Bluetooth® Low Energy Protocol Stack

Fast Prototyping Board BLE & LoRaWAN® Application

Introduction

Sensor devices that support LoRaWAN® communication are required to be easily settable from smartphones using Bluetooth® Low Energy. For example, information such as encryption key and server ID required for LoRaWAN communication, management number and installation location required for sensor device application, etc.

The following are application examples of combining sensor devices that support LoRaWAN communication with Bluetooth Low Energy.

- Update the device firmware.
- Check current sensor data directly from the device.
- The device diagnostics.
- Manage the device position with a smartphone in cooperation with BLE.
- Manage the Application EUI/Key of device with a smartphone in cooperation with BLE.

This application note explains how to implement IoT wireless communication using the "RL78/G14 Fast Prototyping Board" as the main MCU, the "RL78/G1D BLE Module Expansion Board" for BLE communication, the "LoRaWAN stack sample application board" for LoRaWAN communication, and the sensor "TE Connectivity PMOD_MS8607". Set the parameters necessary for LoRaWAN communication with BLE communication, and send sensor data to LoRaWAN Gateway with LoRaWAN communication.

Target Device

RL78/G14 Fast Prototyping Board (Parts Number: RTK5RLG140C00000BJ)

RL78/G1D BLE Module Expansion Board (Parts Number: RTKYRLG1D0B00000BJ)

LoRaWAN stack sample application board

Related Documents

Document Name	Document No.
Bluetooth Low Energy Protocol Stack	
User's Manual	R01UW0095E
API Reference Manual: Basic	R01UW0088E
Application Note: Sample Program	R01AN1375E
Application Note: rBLE Command Specification	R01AN1376E
RL78/G1D	
User's Manual: Hardware	R01UH0515E
RL78/G1D Module	
User's Manual: Hardware	R02UH0004E
User's Manual: Firmware	R01UW0160E
Application Note: Module Control Software	R01AN3362E
LoRaWAN	
LoRaWAN Stack Sample Application Command Reference	R11AN0231E
LoRaWAN IoT Demo LPWA IoT Solution with Cloud	R11AN0412E
RL78/G14	
User's Manual: Hardware	R01UH0186E

Contents

1. Overview

Sensor devices that support LoRaWAN® communication are required to be easily settable from smartphones using Bluetooth® Low Energy. For example, information such as encryption key and server ID required for LoRaWAN communication, management number and installation location required for sensor device application, etc.

The following are application examples of combining sensor devices that support LoRaWAN communication with Bluetooth Low Energy.

- Update the device firmware.
- Check current sensor data directly from the device.
- The device diagnostics.
- Manage the device position with a smartphone in cooperation with BLE.
- Manage the Application EUI/Key of device with a smartphone in cooperation with BLE.

This application note explains how to implement IoT wireless communication using the "RL78/G14 Fast Prototyping Board" as the main MCU, the "RL78/G1D BLE Module Expansion Board" for BLE communication, the "LoRaWAN stack sample application board" for LoRaWAN communication, and the sensor "TE Connectivity PMOD_MS8607". Set the parameters necessary for LoRaWAN communication with BLE communication, and send sensor data to LoRaWAN Gateway with LoRaWAN communication.

RL78/G1D BLE Module Expansion Board and LoRaWAN stack sample application board are connected to the two Pmod interfaces of RL78/G14 Fast Prototyping Board, and TE Connectivity PMOD_MS8607 is connected to the MCU header. Set the parameters required for LoRaWAN communication via BLE communication with a smartphone, and send sensor data to LoRaWAN Gateway via LoRaWAN communication. The sensor data sent to LoRaWAN Gateway is confirmed by LoRaWAN Network Server.

- The RL78/G14 Fast Prototyping Board is an evaluation board specializing in prototype development of applications incorporating the RL78/G14 microcomputer. Built-in emulator circuit equivalent to E2 Emulator Lite enables programming/debugging without additional tools.
  Hereinafter referred to as "FPB".
- The RL78/G1D BLE Module Expansion Board is a Pmod interface evaluation board equipped with the RL78/G1D module (RY7011). Firmware for checking operation is written and can be controlled from the Host MCU program with BLE protocol stack/modem configuration.
  Hereinafter referred to as "BLE module".
- LoRaWAN stack sample application board is a Pmod interface evaluation board in which LoRaWAN stack and Sample application are written. It can be controlled by AT command from Host MCU program.
  Hereinafter referred to as "LoRaWAN module".
- TE Connectivity PMOD_MS8607 is a Pmod interface evaluation board that can measure temperature, pressure and relative humidity.
  Hereinafter referred to as "Sensor module".

2. Development Environment

Describes the build environment and the development environment used to operation confirmation.

2.1 Hardware Environment

— Host
- PC/AT™ compatible computer
- Processor: 1GHz or faster (with support for hyper threading and multicore CPUs)
- Main Memory: We recommend 2GB or more.
- Display: Graphics resolution should be at least 1024 x 768, and the mode should display at least 65,536 colors.
- Interface: USB2.0 (connection with Fast Prototyping Board)

— Development Board
- RL78/G14 Fast Prototyping Board (RTK5RLG140C00000BJ)
- RL78/G1D BLE Module Expansion Board (RTKYRLG1D0B00000BJ)
- LoRaWAN stack sample application board
- TE Connectivity PMOD_MS8607 (DPP901Z000) (pressure, humidity, temperature)

— Smartphone
- Android device or iOS device

— LoRaWAN Communication Device
- Gateway: kerlink Wirnet™ iFemtoCell 923
- Network Server: LORIOT LoRaWAN Network Server
  (Browse from the host machine's web browser)

2.2 Software Environment

— OS
- Windows7 or later

— Web Browser
Use one of the following web browser.
- Microsoft Edge
- Google Chrome

— Integrated Development Environment/Compiler
Use one of the following integration environment and compiler combinations.
- CS+ for CC V8.02.00/CC-RL V1.08.00
- e^2 studio V7.60/CC-RL V1.08.00

3. Application Configuration

3.1 System Configuration

The system configuration figure used in this application note is shown in "Figure 3-1 System Configuration". Connect the RL78/G1D BLE Module Expansion Board for BLE communication and the LoRaWAN stack sample application board for LoRaWAN communication to Pmod I/F of the FPB. Connect the TE Connectivity PMOD_MS8607 to IICA0 pin of MCU header J1.

The FPB applications are mainly composed of programs with the following roles.

- The Host MCU program of the BLE protocol stack modem configuration Note that controls the BLE module.
- The program to control the LoRaWAN module with AT command.
- The program to read the data measured by controlling the Sensor module.

The FPB application controls the BLE module, communicates with the smartphone, and receives the LoRaWAN module configuration parameters. The FPB application sets the received parameters in the LoRaWAN module using AT commands, and sends the data measured by the Sensor module from the LoRaWAN module to the LoRaWAN Gateway. It can check the sensor data of the Sensor module from the LoRaWAN Gateway to the LoRaWAN Network Server using a PC web browser.

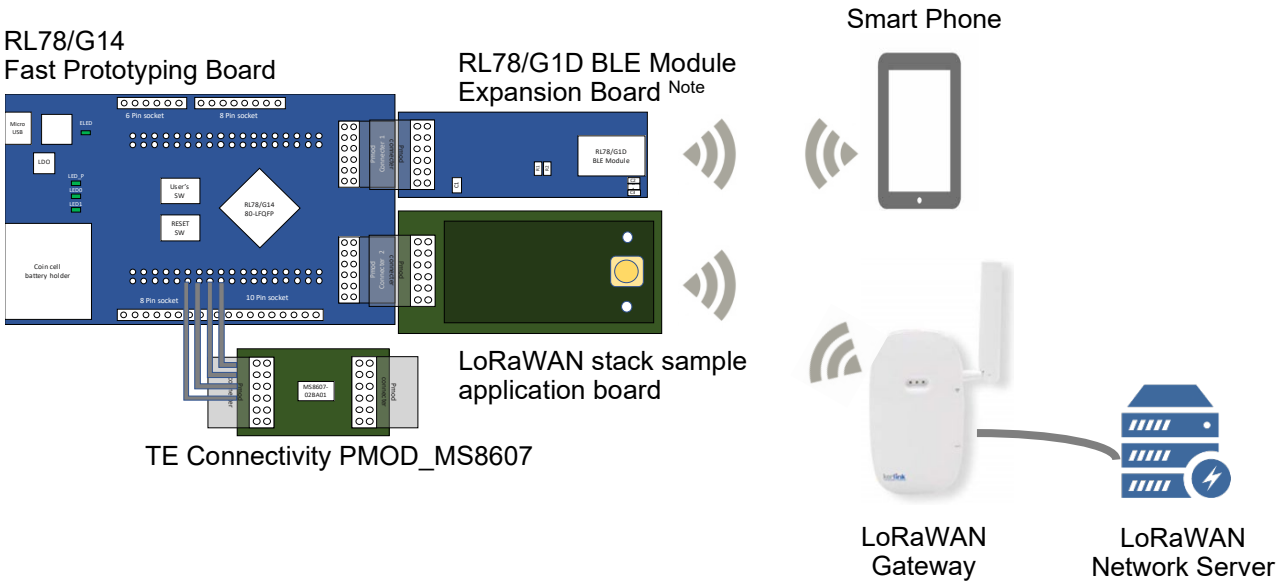


Figure 3-1 System Configuration

Note: The FPB and the BLE module use the BLE protocol stack modem configuration, and communication method is the UART 2-wire branch connection method. Refer to "3.2.1 BLE module Configuration" for Modem configuration and "6.5 BLE Modem Structure - UART 2-wire with branch connection" for UART 2-wire branch connection method.

3.1.1 FPB Connection PIN

The Pmod interface connector pin assignments and the MCU header J1 pin assignments shown in "Figure 3-1 System Configuration" are shown. The functions used for each pin are shown in blue.

(1) PMOD1

Table 3-1 Pin Assignment of PMOD1

RL78/G14 Fast Prototyping Board		Pmod	RL78/G1D BLE Module Expansion Board	
Function Name	Pin Number	Pin Number	Pin Number	Function Name
P74[Note]/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}

Note: The two boards are connected via the Pmod interface, the FPB TxD line required for the UART 2-wire branch connection method cannot be branched and input to INTP3 of the BLE module. In this application enables UART 2-wire branch connection communication by inputting a low level from P74 of FPB to INTP3 of BLE module.

(2) PMOD2

Table 3-2 Pin Assignment of PMOD2

RL78/G14 Fast Prototyping Board		Pmod	LoRaWAN stack sample application board
Function Name	Pin Number	Pin Number	Function Name
P16/TI01/TO01/INTP5/TRDIOC0/IVREF0/(SI00)/(RxD0)	48	1	NC
P13/**TxD2**/SO20/TRDIOA1/IVCMP1	51	2	**RxD**
P14/**RxD2**/SI20/SDA20/TRDIOD0/(SCLA0)	50	3	**TxD**
P15/SCK20/SCL20/TRDIOB0/(SDAA0)	49	4	NC
GND	-	5	GND
VCC	-	6	V_{DD}
P141/PCLBUZ1/INTP7	1	7	NC
P110/(INTP11)	55	8	**RESET#**
P17/TI02/TO02/TRDIOA0/TRDCLK/IVCMP0/(SO00)/(TxD0)	47	9	NC
P111	56	10	NC
GND	-	11	GND
VCC	-	12	V_{DD}

(3) MCU Header J1

Table 3-3 Pin Assignment of MCU Header J1

RL78/G14 Fast Prototyping Board		TE Connectivity PMOD_MS8607	
Function Name	J1 Pin Number	Pmod Pin Number	Function Name
-	-	1	NC
-	-	2	NC
P60/**SCLA0**	21	3	**SCL**
P61/**SDAA0**	22	4	**SDA**
EV_{SS0}	17	5	**GND**
EV_{DD0}	19	6	**V_{DD}**
-	-	7	NC
-	-	8	NC
-	-	9	SCL
-	-	10	SDA
-	-	11	GND
-	-	12	V_{DD}

3.2 Software Configuration

3.2.1 BLE module Configuration

The software configuration of RL78/G14 which is a Host MCU and RY7011 which is a BLE MCU is shown. Firmware for operation check is written in RY7011 at the time of shipment, and it supports some profiles. In this application, General Purpose Communication Profile (GPCP) is used. For other profiles, refer to "7. Profile" in the "RL78/G1D Module Firmware User's Manual" (R01UW0160).

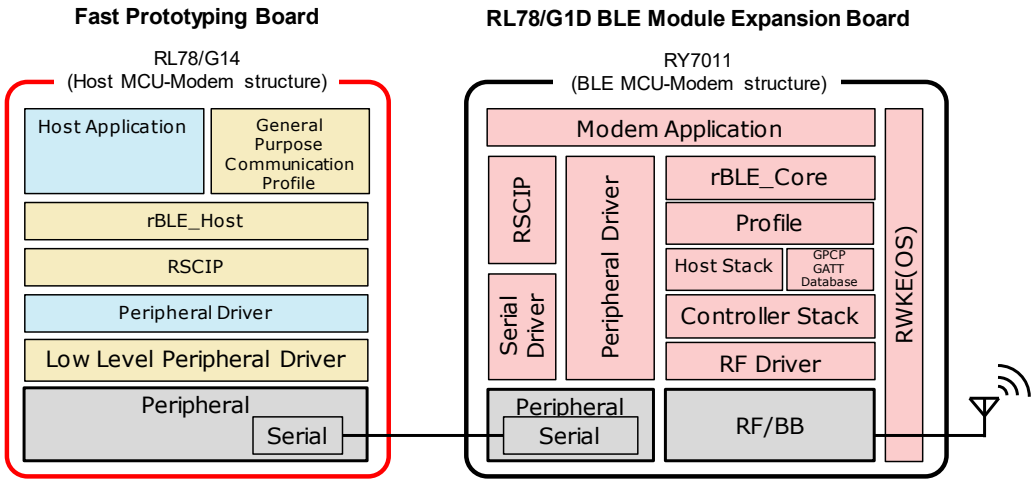


Figure 3-2 BLE module communication software configuration

The software of the Host MCU consists of low level peripheral drivers and the peripheral drivers which controls MCU peripheral hardware, the RSCIP (Renesas Serial Communication Interface Protocol), the rBLE_Host which provides rBLE APIs, the host application which controls the system, and the General Purpose Communication Profile (GPCP) using the GATT API.

The low level peripheral driver code is generated by the Code Generator. The RSCIP and the rBLE_Host are included in the BLE protocol stack package and provided code. When developing software, it is necessary to use the latest code which is provided by the BLE protocol stack package.

Table 3-4 Host MCU Software Configuration

Software	Functions	When developing software
Host Application	Initializing rBLE Scheduling rBLE command execution Registering rBLE event callbacks	Need to be coded
General Purpose Communication Profile (GPCP)	Custom Profile using GATT APIs	No need to be coded (provided by package) Note1
rBLE_Host	Providing rBLE APIs Executing rBLE event callbacks	No need to be coded (provided by package) Note1
RSCIP (Renesas Serial Comm	Controlling serial communication	No need to be coded (provided by package) Note1
Peripheral Driver	Controlling Host MCU peripheral hardware	Need to be coded
Low Level Peripheral Driver	Controlling Host MCU peripheral hardware primitively	No need to be coded (generated by tool) Note2

Notes: 1. Code files for software development are provided by BLE protocol stack package.
2. Code files for software development are generated by the Code Generator.

The software of the BLE MCU consists of RF driver which controls RF/BB, Host/Controller stacks, Profiles, rBLE_Core, Serial Driver and RSCIP for communicating with the Host MCU, RWKE (Renesas Wireless Kernel Extension) which manages the system and the Modem application. The build environment is provided by "RL78/G1D Module Control Software" (R01AN3362).

Table 3-5 BLE MCU Software Configuration

Software	Functions
Modem Application	Controlling RSCIP and rBLE
RWKE	Managing the whole system schedule and memory resource.
RSCIP	Controlling serial communication
Peripheral Driver/Serial Driver	Controlling BLE MCU peripheral hardware
rBLE_Core	Providing rBLE APIs
Profile	Providing Profiles functions
Host Stack	Providing GAP, GATT, SM, L2CAP functions
GSCP GATT Database	GATT Database of General Purpose Communication Profile
Controller Stack	Providing LL functions

3.2.2 LoRaWAN module Configuration

The software configuration of the RL78/G14 of the FPB and the LoRaWAN module are shown below. The LoRaWAN module consists of LoRaWAN stack Note1, peripheral driver, sample application Note2 that controls LoRaWAN stack with AT command. FPB consists of a LoRaWAN module application for controlling the LoRaWAN module by AT commands and a Sensor control application for controlling the Sensor module and reading the measured data.

Notes: 1. Refer to "LoRaWAN stack reference guide" (R11AN0228)
2. Refer to "LoRaWAN Stack Sample Application Command Reference" (R11AN0231)

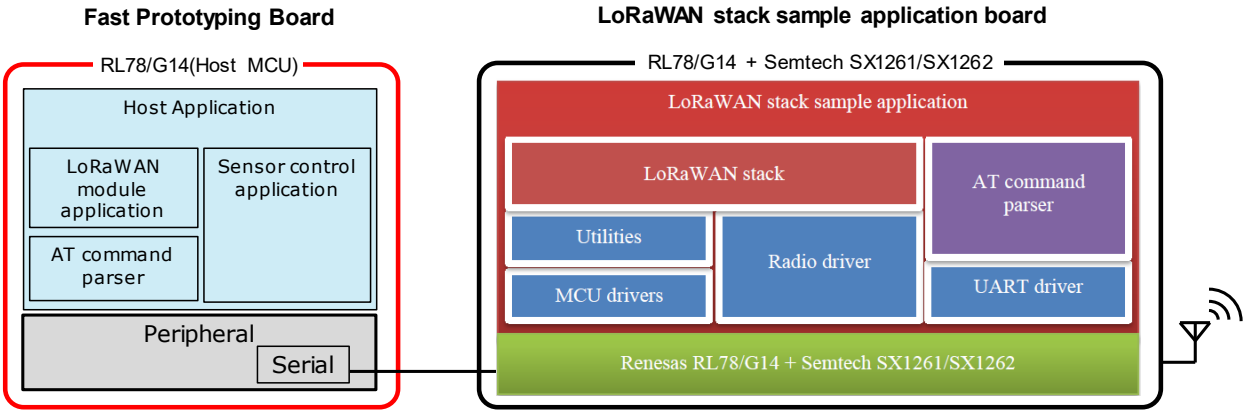


Figure 3-3 LoRaWAN module communication software configuration

3.3 Peripheral Configuration

The peripheral functions of The RL78/G14 used in the FPB applications are shown below.

The Peripheral functions that are necessary for applications to use the BLE module or the LoRaWAN module are defined as "Mandatory" and others as "Optional".

Table 3-6 Peripheral Functions

Peripheral Function	Purpose		Necessity
Clock Generator	Used for operating frequency of the RL78/G14.		Mandatory
Port	P74	Used for a WAKEUP pin for UART 2-wire branch connection method in UART communication with the BLE module.	Mandatory
	P130	Used for reset release pin of the BLE module.	Mandatory
	P110	Used for reset release pin of the LoRaWAN module.	Mandatory
	P43	Used for LED0 on the FPB.	Optional
	P44	Used for LED1 on the FPB.	Optional
	P137	Used for SW_USR on the FPB.	Optional
Serial-UART0	Used for UART communication with the BLE module.		Mandatory
Serial-UART2	Used for UART communication with the LoRaWAN module.		Mandatory
Serial-IICA0	Used for IICA communication with the Sensor module.		Optional
Timer-TAU0	Used for A/D converter wait with the Sensor module.		Optional
12-Bit Interval Timer	Used for timer function used by rBLE_Host and RSCIP of BLE Host MCU program.		Mandatory

3.4 File Configuration

File configuration of this application is shown below.

The (R) mark of file configuration indicates that the file is included in the BLE protocol stack package. When developing software, it is necessary to use the latest code which is provided by BLE protocol stack package.

```
RL78G14_FPB_LoRaWAN_BLE_Application
 ├──ROM_File                                              Execution file (HEX File)
 ├──HostSample
 │   ├──Platform
 │   │   ├──driver
 │   │   │   ├──dataflash
 │   │   │   │   │   dataflash.c                         Data Flash driver code file
 │   │   │   │   │   dataflash.h                         Data Flash driver header file
 │   │   │   │   │   eel_descriptor.c                   EEPROM Emulation Library code file
 │   │   │   │   │   eel_descriptor.h                   EEPROM Emulation Library header file
 │   │   │   │   │   fdl_descriptor.c                   Data Flash Access Library code file
 │   │   │   │   │   fdl_descriptor.h                   Data Flash Access Library header file
 │   │   │   │   │
 │   │   │   │   └──cc_rl
 │   │   │   │          eel.h                           EEPROM Emulation Library header file
 │   │   │   │          eel.lib                         EEPROM Emulation Library file
 │   │   │   │          eel_types.h                     EEPROM Emulation Library header file
 │   │   │   │          fdl.h                           Data Flash Access Library header file
 │   │   │   │          fdl.lib                         Data Flash Access Library file
 │   │   │   │          fdl_types.h                     Data Flash Access Library header file
 │   │   │   │
 │   │   │   ├──ms8607
 │   │   │   │      r_ms8607.c                         MS8607 Sensor driver code file
 │   │   │   │      r_ms8607.h                         MS8607 Sensor driver header file
 │   │   │   │
 │   │   │   ├──serial
 │   │   │   │      r_uart_ble_modem.c                 BLE UART driver code file
 │   │   │   │      r_uart_ble_modem.h                 BLE UART driver header file
 │   │   │   │      r_uart_lora.c                      LoRaWAN UART driver code file
 │   │   │   │      r_uart_lora.h                      LoRaWAN UART driver header file
 │   │   │   │
 │   │   │   └──timer
 │   │   │          timer.c                           timer driver code file
 │   │   │          timer.h                           timer driver header file
 │   │   │
 │   │   └──include
 │   │          arch.h                          (R)   architecture header file
 │   │          compiler.h                      (R)   compiler header file
 │   │          ll.h                            (R)   low level macro header file
 │   │          rscip_api.h                     (R)   RSCIP callback header file
 │   │          types.h                         (R)   type definition header file
 │   │
 │   └──rBLE
 │       ├──host
 │       │   │  rble_host.c                      (R)   rBLE_Host code file
 │       │   │  rble_if_api_cb.c                 (R)   rBLE API callback code file
 │       │   │
 │       │   ├──gap
 │       │   │      rble_api_gap.c               (R)   GAP API code file
 │       │   │
 │       │   ├──gatt
 │       │   │      rble_api_gatt.c              (R)   GATT API code file
 │       │   │
 │       │   ├──sm
 │       │   │      rble_api_sm.c                (R)   SM API code file
 │       │   │
```

```
|   |   └─vs
|   |       rble_api_vs.c                                   (R)  VS API code file
|   |
|   ├─include
|   |   |   db_handle.h                                     (R)  data base handle header file
|   |   |   prf_sel.h                                       (R)  profile select header file
|   |   |   rble.h                                          (R)  rBLE macro header file
|   |   |   rble_api.h                                      (R)  rBLE API header file
|   |   |   rble_app.h                                      (R)  rBLE SCP API header file
|   |   |   rble_trans.h                                    (R)  rBLE communication header file
|   |   |
|   |   └─host
|   |       rble_host.h                                     (R)  rBLE_Host  header file
|   |
|   ├─rscip
|   |   rscip.c                                             (R)  RSCIP code file
|   |   rscip.h                                             (R)  RSCIP header file
|   |   rscip_cntl.c                                        (R)  RSCIP control code file
|   |   rscip_cntl.h                                        (R)  RSCIP control header file
|   |   rscip_ext.h                                         (R)  RSCIP external callback header file
|   |   rscip_uart.c                                        (R)  RSCIP serial communication code file
|   |   rscip_uart.h                                        (R)  RSCIP serial communication header file
|   |
|   ├─sample_app
|   |   r_app_ble.c                                              BLE Application code file
|   |   r_app_lora.c                                             LoRaWAN Application code file
|   |   r_app_lora.h                                             LoRaWAN Application header file
|   |   r_app_ms8607.c                                           ms8607 Sensor Application code file
|   |   r_app_ms8607.h                                           ms8607 Sensor Application header file
|   |
|   └─sample_profile
|     └─vuart
|         vuart.h                                                General purpose communication header file
|         vuarts.c                                               General purpose communication server code file
|         vuarts.h                                               General purpose communication server header file
|
└─project
  ├─cs_cc
  |   |   cstart.asm                                           startup routine
  |   |   iodefine.h                                           I/O header file
  |   |   RL78G14_FPB_LoRaWAN_BLE_Applicaiotn.mtpj             CS+ for CC project file
  |   |   RL78G14_FPB_LoRaWAN_BLE_Application.rcpe             same as above
  |   |   stkinit.asm                                          stack area initialize routine
  |   |
  |   └─src
  |       r_cg_cgc.c                                           clock generator driver code file
  |       r_cg_cgc.h                                           clock generator driver header file
  |       r_cg_cgc_user.c                                      clock generator driver user code file
  |       r_cg_intc.c                                          interrupt driver code file
  |       r_cg_intc.h                                          interrupt driver header file
  |       r_cg_intc_user.c                                     interrupt driver user code file
  |       r_cg_it.c                                            interval timer driver code file
  |       r_cg_it.h                                            interval timer driver header file
  |       r_cg_it_user.c                                       interval timer driver user code file
  |       r_cg_macrodriver.h                                   macro header file
  |       r_cg_macrodriver_hostsample.h                        macro header file (backup)
  |       r_cg_port.c                                          port driver code file
  |       r_cg_port.h                                          port driver header file
  |       r_cg_port_user.c                                     port driver user code file
  |       r_cg_serial.c                                        serial driver code file
  |       r_cg_serial.h                                        serial driver header file
  |       r_cg_serial_user.c                                   serial driver user code file
```

r_cg_timer.c	timer array unit code file
r_cg_timer.h	timer array unit header file
r_cg_timer_user.c	timer array unit user code file
r_cg_userdefine.h	user defined macro header file
r_main.c	main loop code file
r_systeminit.c	peripheral initialization code file
e2studio	
.cproject	e2studio project file
.project	same as above
RL78G14_FPB_LoRaWAN_BLE_Application HardwareDebug.launch	same as above
r_option_cc.txt	same as above
.settings	
com.renesas.tools.misrac.prefs	e2studio project file
CoverageSetting.xml	same as above
Dependency_Scan_Preferences.prefs	same as above
e2studio_project.prefs	same as above
renesasPGModel.xml	same as above
CodeGenerator	
cgproject.cgp	e2studio project file
cgprojectDatas.datas	same as above
generate	
cstart.asm	startup routine
iodefine.h	I/O header file
stkinit.asm	stack area initialize routine
src	
r_cg_cgc.c	clock generator driver code file
r_cg_cgc.h	clock generator driver header file
r_cg_cgc_user.c	clock generator driver user code file
r_cg_intc.c	interrupt driver code file
r_cg_intc.h	interrupt driver header file
r_cg_intc_user.c	interrupt driver user code file
r_cg_it.c	interval timer driver code file
r_cg_it.h	interval timer driver header file
r_cg_it_user.c	interval timer driver user code file
r_cg_macrodriver.h	macro header file
r_cg_macrodriver_hostsample.h	macro header file (backup)
r_cg_port.c	port driver code file
r_cg_port.h	port driver header file
r_cg_port_user.c	port driver user code file
r_cg_serial.c	serial driver code file
r_cg_serial.h	serial driver header file
r_cg_serial_user.c	serial driver user code file
r_cg_timer.c	timer array unit code file
r_cg_timer.h	timer array unit header file
r_cg_timer_user.c	timer array unit user code file
r_cg_userdefine.h	user defined macro header file
r_main.c	main loop code file
r_systeminit.c	peripheral initialization code file

4. How to Build

The project and build procedure for building the FPB BLE・LoRaWAN Application are shown below.

Table 4-1 FPB BLE・LoRaWAN Application project

CS+ for CC		
	Project File	RL78G14_FPB_BLE_LoRaWAN_Application\project\cs_cc\ RL78G14_FPB_BLE_LoRaWAN_Applicaiotn.mtpj
	HEX File	RL78G14_FPB_BLE_LoRaWAN_Application\project\cs_cc\DefaultBuild\ RL78G14_FPB_BLE_LoRaWAN_Applicaiotn.hex
e² studio		
	Project Folder	RL78G14_FPB_BLE_LoRaWAN_Application\project\e2studio
	HEX File	RL78G14_FPB_BLE_LoRaWAN_Application\project\e2studio\HardwareDebug\ RL78G14_FPB_BLE_LoRaWAN_Application.hex

4.1 CS+ for CC

1. Double click the project file shown in Project File of "Table 4-1 FPB BLE・LoRaWAN Application project".
2. Right click "RL78G14_FPB_BLE_LoRaWAN_Application (Project)" in the "Project Tree" and select "Build RL78G14_FPB_BLE_LoRaWAN_Application" from the dropdown menu to start the build.
3. The HEX file is generated in the path shown in the HEX File column of CS+ for CC in "Table 4-1 FPB BLE・LoRaWAN Application project".

4.2 e2 studio

1. Launch e²studio.
2. Right click on the "Project Explorer" and select "Import" from the displayed menu.
3. The "Import" window will be displayed. Select "Existing project to workspace" and click "Next".
4. In the "Select root directory" form, select the project folder shown in the Project Folder of e² studio in "Table 4-1 FPB BLE・LoRaWAN Application project". After selection, confirm that the specified project is displayed in "Project" and click "Finish". Then the "Import" window is closed.
5. Right click on the project displayed on the "Project Explorer" and select "Build Project" to start the build.
6. The HEX file is generated in the path shown in the HEX File of e² studio in "Table 4-1 FPB BLE・LoRaWAN Application project".

5. Execute Application

Check the operation of BLE communication and LoRaWAN communication with the configuration shown in "Figure 5-1 Application Execution Environment".

5.1 Install GATTBrowser

Install GATTBrowser to a smartphone.

- GATTBrowser for Android

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

- GATTBrowser for iOS

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

5.2 Prepare LORIOT LoRaWAN Network Server

Refer to "3. LoRaWAN Network Server" in "LoRaWAN® IoT Demo LPWA IoT Solution with Cloud" (R11AN0412) for setup.

5.3 Execution Environment

Build the program referring to "4. How to Build". Then connect the FPB and PC with a USB cable and download the program.

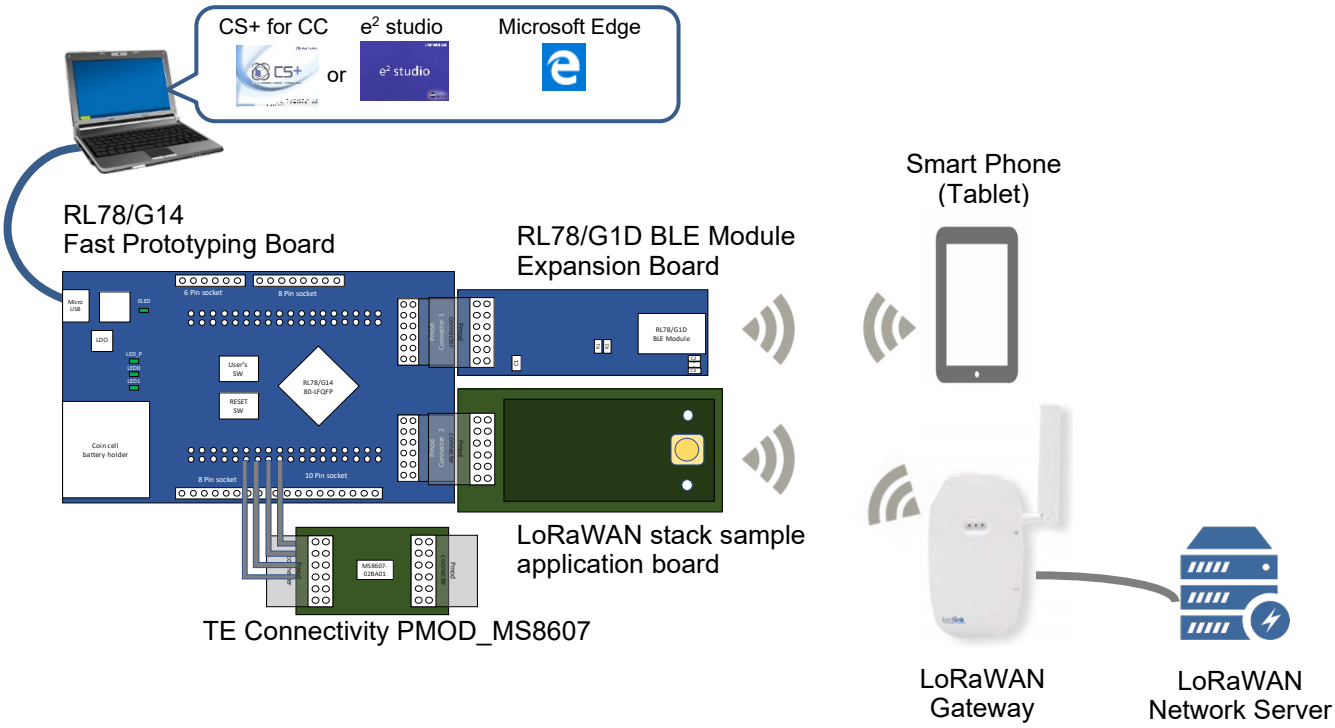


Figure 5-1 Application Execution Environment

5.4 Execution Process

The overall operation of the application is shown in "Figure 5-2 Application Flow".

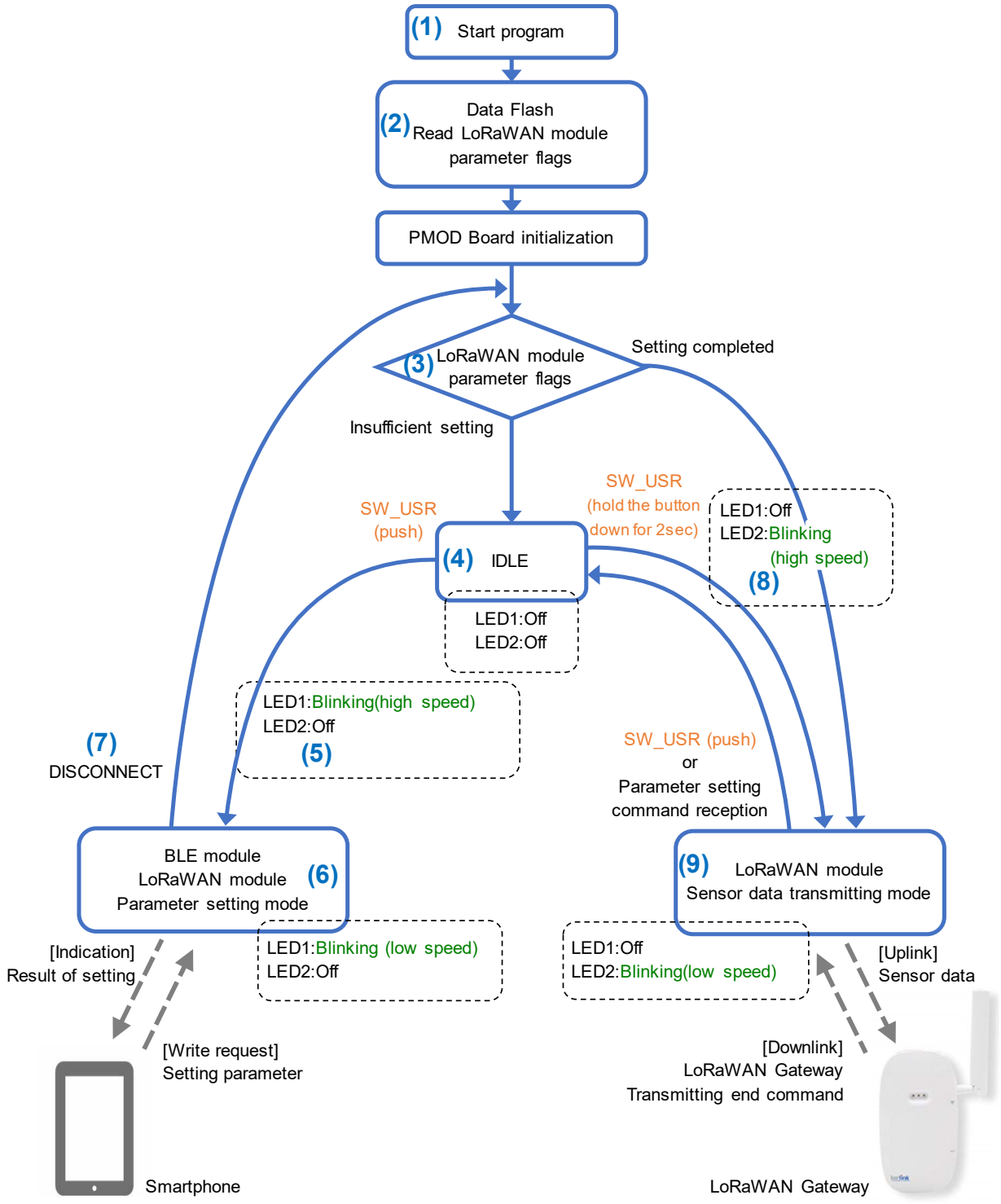


Figure 5-2 Application Flow

[Start Program]

(1) Start Program.

(2) Read the LoRaWAN module parameter flags from the FPB data flash.

(3) Check whether LoRaWAN module parameter flags are set.

- All parameters required for LoRaWAN module communication are set.
  ➔ Go to [LoRaWAN module: Sensor data transmitting mode]

- Parameters required for LoRaWAN module communication are not set.
  ➔ Go to [IDLE]

[IDLE]

(4) IDLE state. Wait for the push switch (SW_USR) to be pushed.
(LED1: Off, LED2: Off)

- Push SW_USR
  (Parameters required for LoRaWAN module communication are not set.)
  ➔ Go to [BLE module: LoRaWAN parameter setting mode]

- Push SW_USR (hold the button down for 2sec)
  (All parameters required for LoRaWAN module communication are set.)
  ➔ Go to [LoRaWAN module: Sensor data transmitting mode]

[BLE module: LoRaWAN parameter setting mode]

(5) The BLE module starts advertising.
(LED1: Blinking (high speed), LED2: Off)

(6) Connect to the BLE module with a smartphone and set the LoRaWAN module parameters from the smartphone.
(LED1: Blinking (low speed), LED2: Off)

For smartphone operations, refer to "5.4.1 LoRaWAN module parameters setting by Android device" for Android, and "0

LoRaWAN module parameters setting by iOS device" for iOS.

(7) After setting all the LoRaWAN module parameters, disconnect BLE connection from the smartphone. When the BLE module detects a disconnection, it saves the parameter flag set to the LoRaWAN module to the FPB data flash. If you turn off the power or disconnect E2Lite without disconnecting from the smartphone, the parameter flag is not saved in the FPB data flash, so set it again.

- All parameters required for LoRaWAN module communication are set.

  ➔ Go to [LoRaWAN module: Sensor data transmitting mode]

- Parameters required for LoRaWAN module communication are not set.

  ➔ Go to [IDLE]

[LoRaWAN module: Sensor data transmitting mode]

(8) Start LoRaWAN module communication. First, activate to LoRaWAN Gateway.
(LED1: Off, LED2: Blinking (high speed))

(9) When the LoRaWAN Gateway is accepted, the measured data is read from the Sensor module at 1-minute intervals and sent to the LoRaWAN Gateway.
(LED1: Off, LED2: Blinking (low speed))

Check the LORIOT Network Server site with a web browser and confirm that data is being sent from the LoRaWAN module to the LoRaWAN Network Server. Refer to “5.4.3.1 Check Sensor Data”.

There are two ways to interrupt LoRaWAN module communication and return to IDLE.

- LoRaWAN module receives communication termination command sent from LoRaWAN Network Server. Refer to "5.4.3.2 LoRaWAN Gateway Communication End Command Transmitting".

  ➔ Go to [IDLE]

- Push SW_USR

  ➔ Go to [IDLE]

5.4.1 LoRaWAN module parameters setting by Android device

1. Start GATTBrowser installed on the Android device.
2. Connect to the device displayed as RTK5RL140C from the scan results. (Arrow (1) in Figure A1)
3. When connected, a list of services is displayed. Scroll down to the bottom and select "Indication Characteristic" of "Renesas Virtual UART Service". (Arrow (2) in Figure A2)
4. Tap "Indication Off" to "Indication On". (Arrow (3) in Figure A3)
5. Return to the list of services. (Arrow (4) in Figure A3)

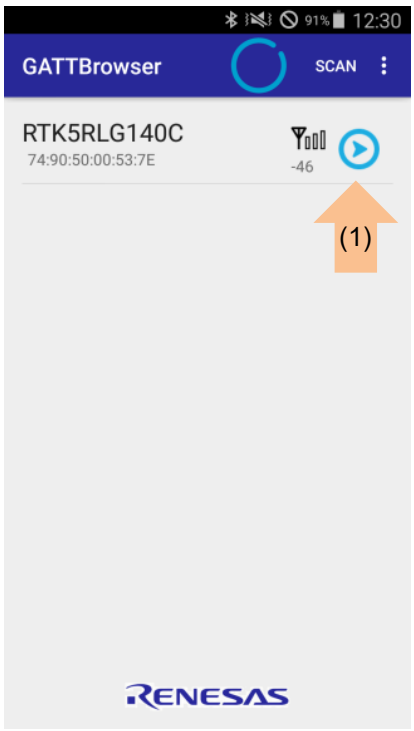


Figure A1

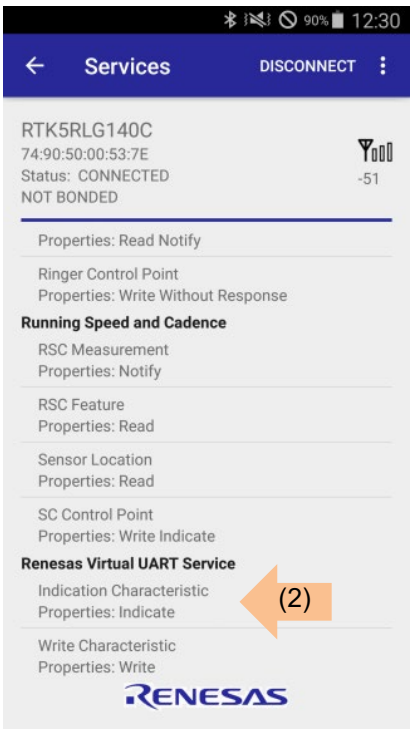


Figure A2

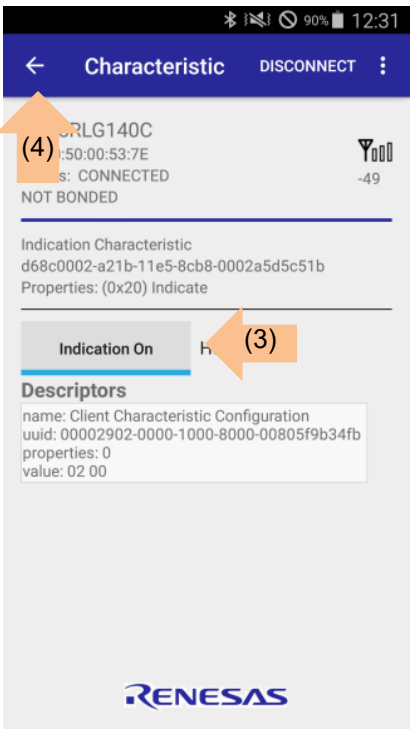


Figure A3

6. Select "Write Characteristic". (Arrow (5) in Figure A4)
7. Select "Hex". (Arrow (6) in Figure A5)
8. Input LoRaWAN setting parameter. (Arrow (7) in Figure A5)
9. Tap "Write" to send. (Arrow (8) in Figure A5)
10. Return to the list of services. (Arrow (9) in Figure A5)

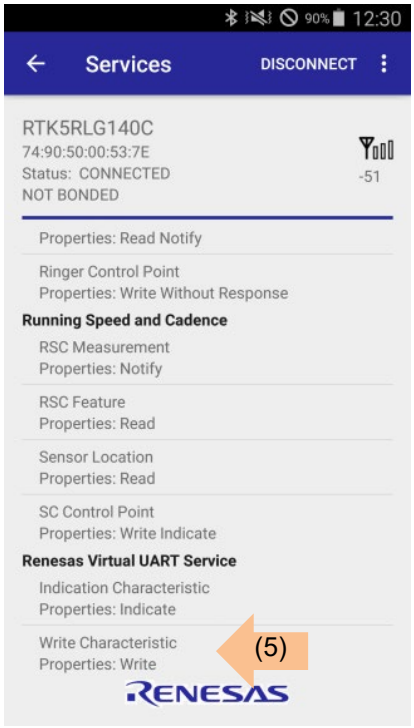


Figure A4

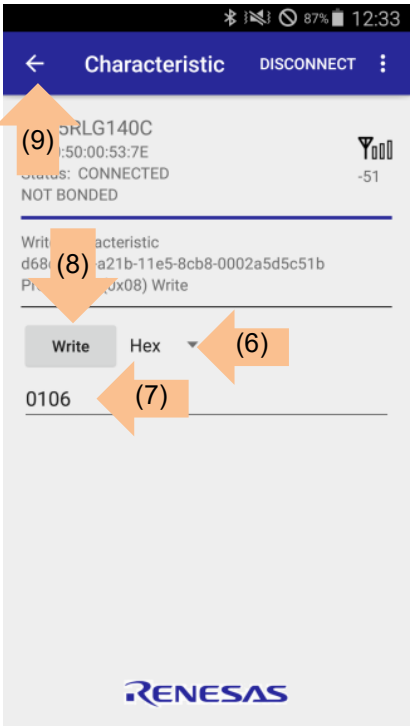


Figure A5

11. Select "Indication Characteristic". (Arrow (10) in Figure A6)
12. The LoRaWAN module parameter setting result is displayed. (Arrow (11) in Figure A7)
    Refer to "6.4.1.2 AT Command Result Code Format" for the result format.
13. After all parameters have been set, select "DISCONNECT". (Arrow (12) in Figure A7)
    To determine whether the setting is complete, check that 0x7F is displayed in the third byte (parameter flag) of the AT command result code format.

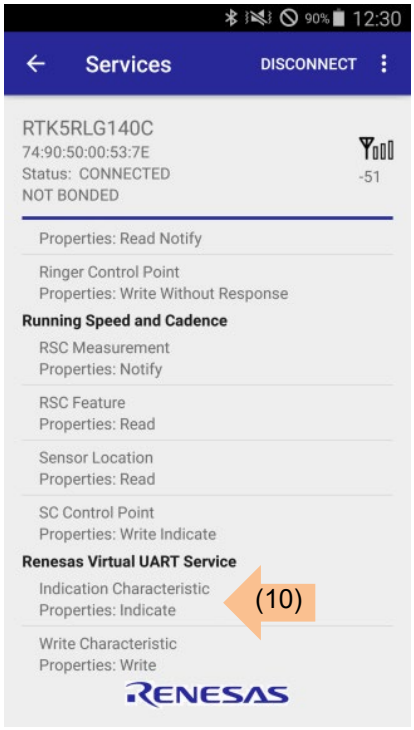


Figure A6

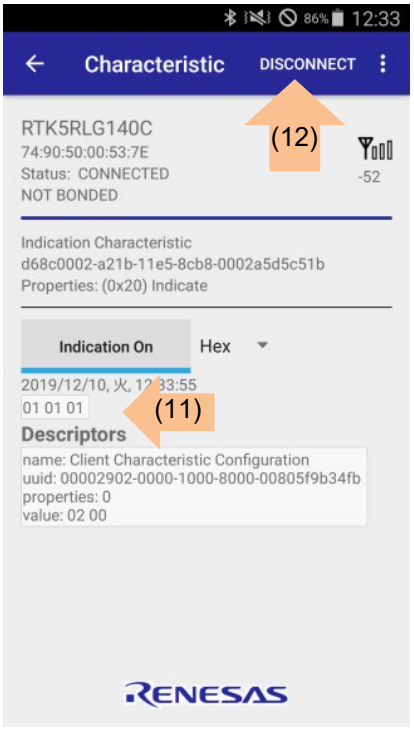


Figure A7

5.4.2 LoRaWAN module parameters setting by iOS device

1. Start GATTBrowser installed on the iOS device.
2. Connect to the device displayed as RTK5RL140C from the scan results. (Arrow (1) in Figure B1)
3. When connected, a list of services is displayed. Scroll down to the bottom and select "Indication Characteristic" of "Renesas Virtual UART Service". (Arrow (2) in Figure B2)
4. Tap "Enable Indication" to "Disable Indication". (Arrow (3) in Figure B3)
5. Return to the list of services. (Arrow (4) in Figure B3)

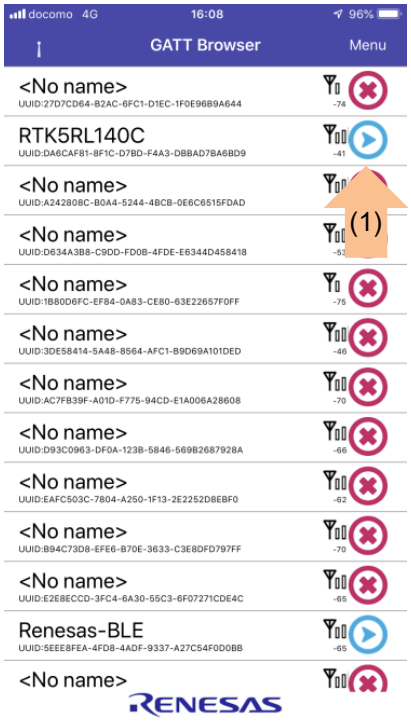


Figure B1

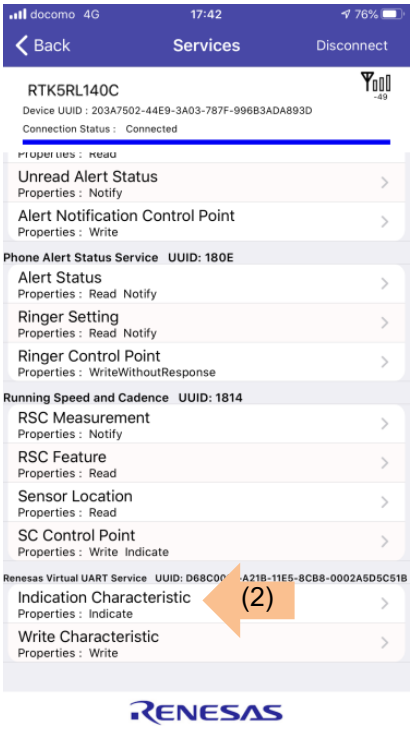


Figure B2

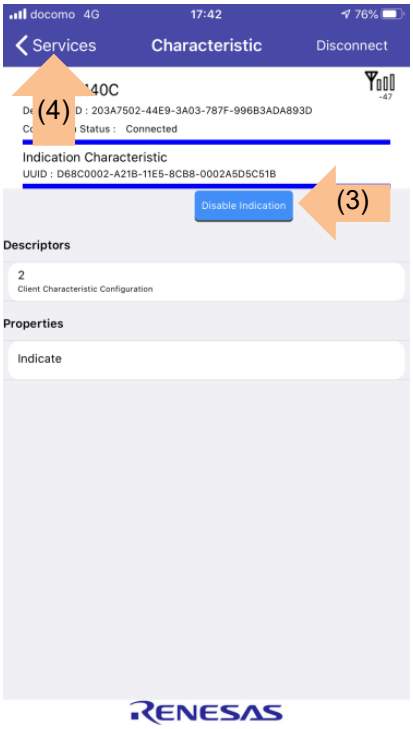


Figure B3

6. Select "Write Characteristic". (Arrow (5) in Figure B4)
7. Select "Hex". (Arrow (6) in Figure B5)
8. Input LoRaWAN setting parameter. (Arrow (7) in Figure B5)
9. Tap "Write" to send. (Arrow (8) in Figure B5)
10. Return to the list of services. (Arrow (9) in Figure B5)

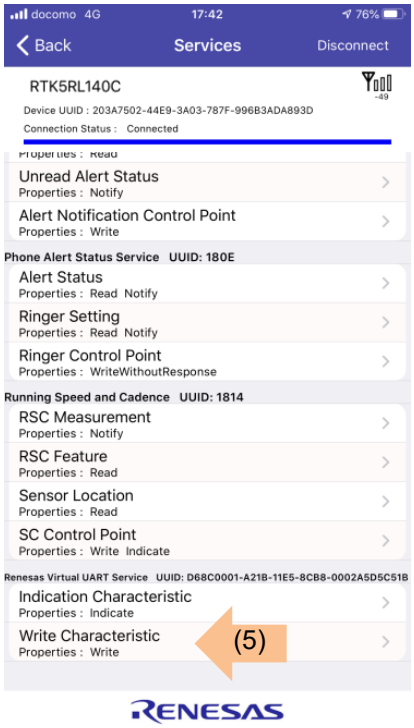


Figure B4

Figure B5

14. Select "Indication Characteristic". (Arrow (10) in Figure B6)
15. The LoRaWAN module parameter setting result is displayed. (Arrow (11) in Figure B7)
    Refer to "6.4.1.2 AT Command Result Code Format" for the result format.
16. After all parameters have been set, select "DISCONNECT". (Arrow (12) in Figure B7)
    To determine whether the setting is complete, check that 0x7F is displayed in the third byte (parameter flag) of the AT command result code format.

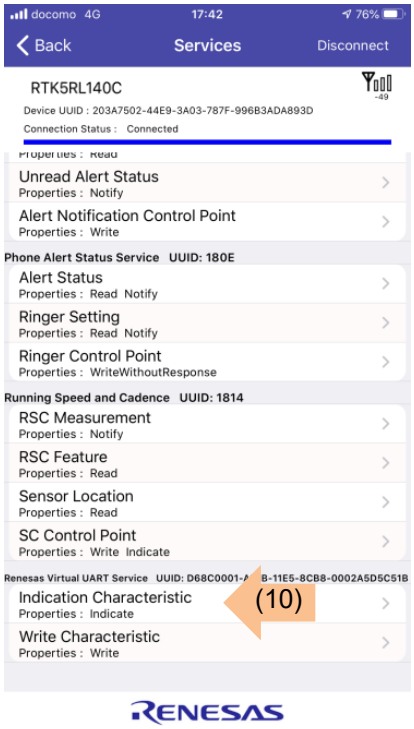


Figure B6

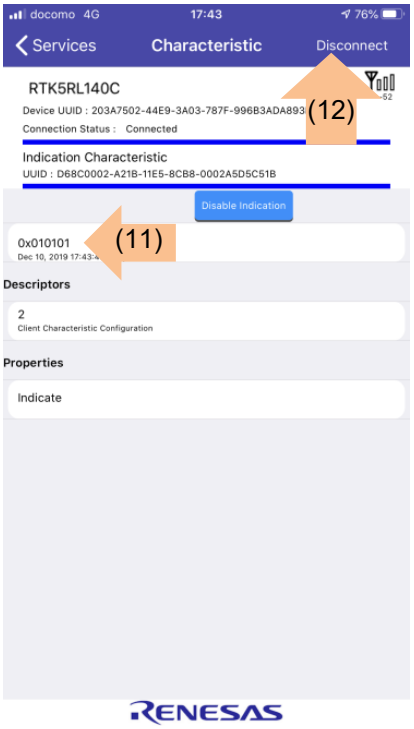


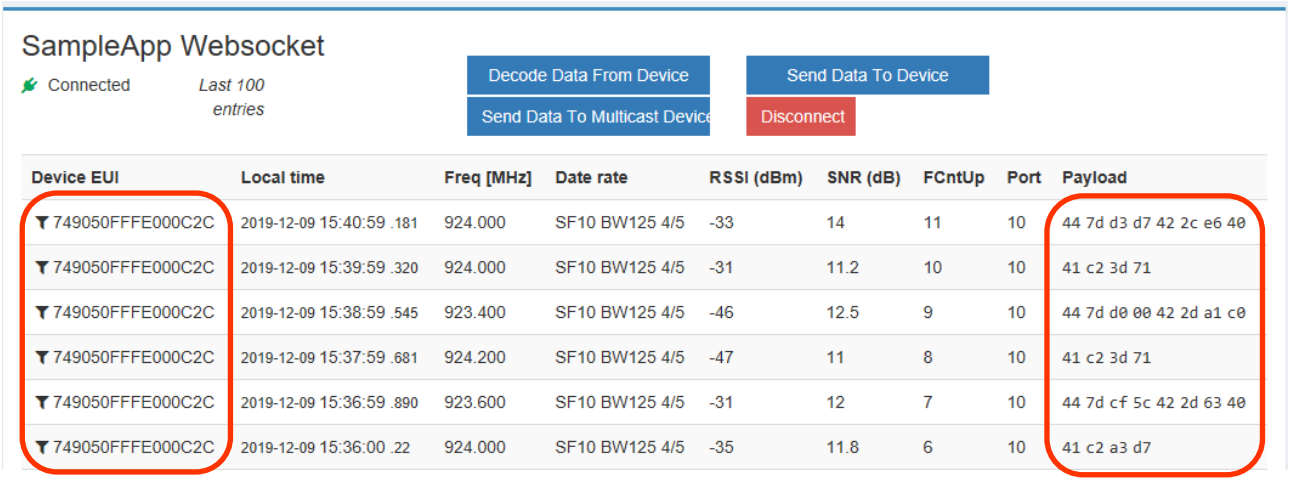
Figure B7

5.4.3 LoRaWAN Network Server

5.4.3.1 Check Sensor Data

The sensor data sent from the LoRaWAN module is checked by the LoRaWAN Network Server operated by LORIOT.

You can confirm that sensor data (Payload) is received from LoRaWAN module with Device EUI of 0x749050FFFE000C2C.



Device EUI	Local time	Freq [MHz]	Date rate	RSSI (dBm)	SNR (dB)	FCntUp	Port	Payload
749050FFFE000C2C	2019-12-09 15:40:59 .181	924.000	SF10 BW125 4/5	-33	14	11	10	44 7d d3 d7 42 2c e6 40
749050FFFE000C2C	2019-12-09 15:39:59 .320	924.000	SF10 BW125 4/5	-31	11.2	10	10	41 c2 3d 71
749050FFFE000C2C	2019-12-09 15:38:59 .545	923.400	SF10 BW125 4/5	-46	12.5	9	10	44 7d d0 00 42 2d a1 c0
749050FFFE000C2C	2019-12-09 15:37:59 .681	924.200	SF10 BW125 4/5	-47	11	8	10	41 c2 3d 71
749050FFFE000C2C	2019-12-09 15:36:59 .890	923.600	SF10 BW125 4/5	-31	12	7	10	44 7d cf 5c 42 2d 63 40
749050FFFE000C2C	2019-12-09 15:36:00 .22	924.000	SF10 BW125 4/5	-35	11.8	6	10	41 c2 a3 d7

Figure 5-3 Confirmation of sensor data with LoRaWAN Network Server

5.4.3.2 LoRaWAN Gateway Communication End Command Transmitting

(1) To check the sensor data, press the “Send Data To Device” button at the top of the page to display the "Send Data To Device" dialog.

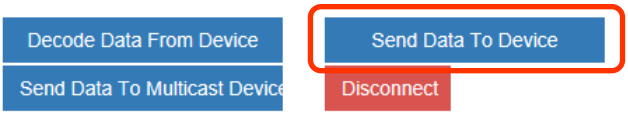


Figure 5-4 Send Data To Device Button

(2) Enter the parameters set in the LoRaWAN module and the communication end command and press the “Send to device” button.

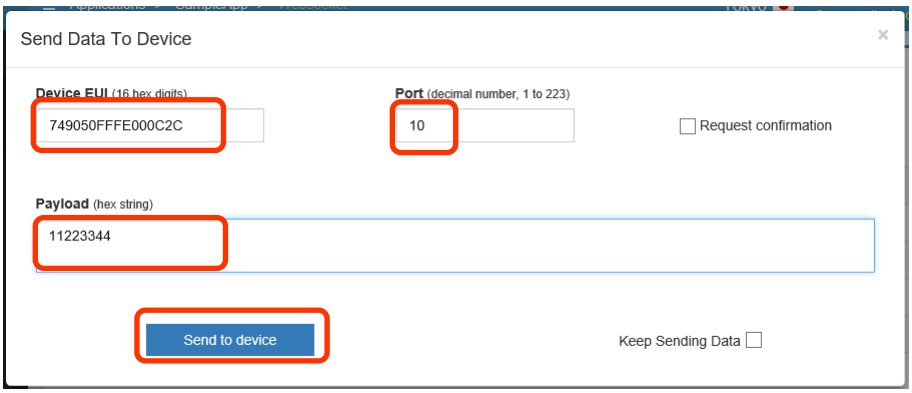


Figure 5-5 Send Data To Device Dialog

(3) The communication end command is held in the transmission queue.

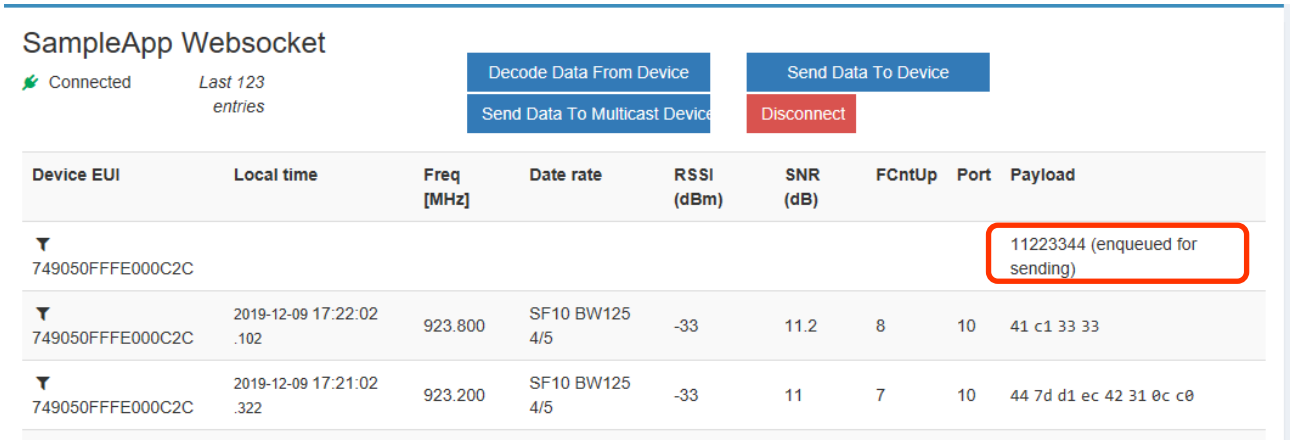


Figure 5-6 Enqueued For Sending

(4) Sends a communication end command triggered by sensor data reception from LoRaWAN module.

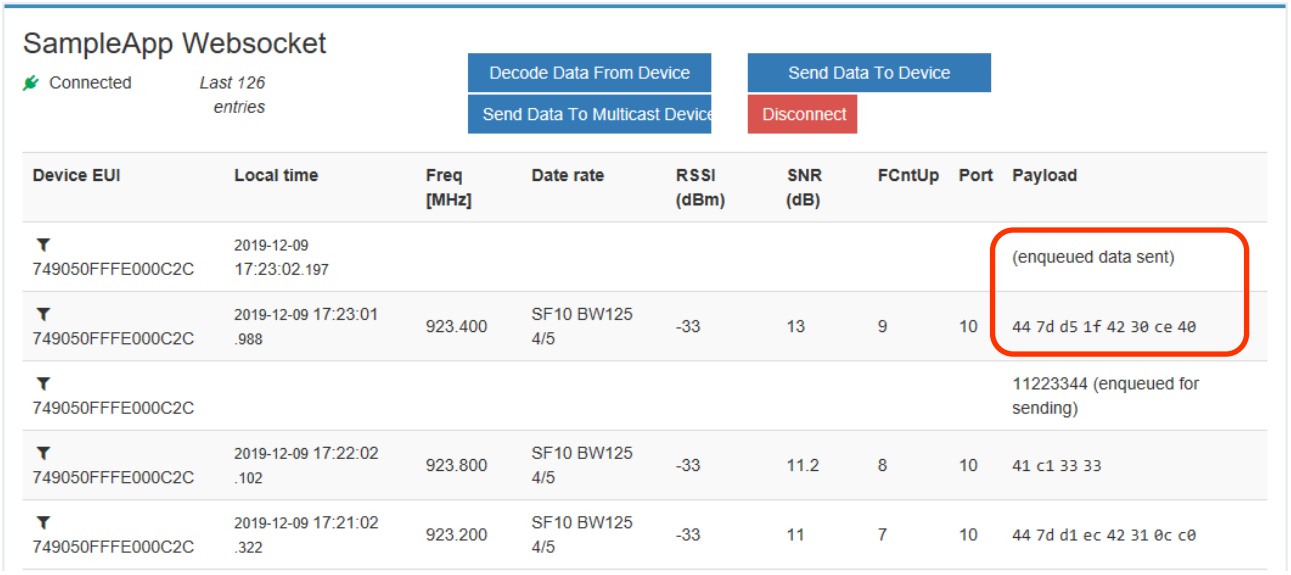


Figure 5-7 Enqueued For Sending

6. Processing Flow

This section describes the Host MCU application processing shown in "Figure 3-2 BLE module communication software configuration" and "Figure 3-3 LoRaWAN module communication software configuration".

6.1 main loop

The main loop of this application consists of two processing units, the BLE module communication unit and the LoRaWAN module communication unit, as shown in "Figure 6-1 Main Loop Processing".

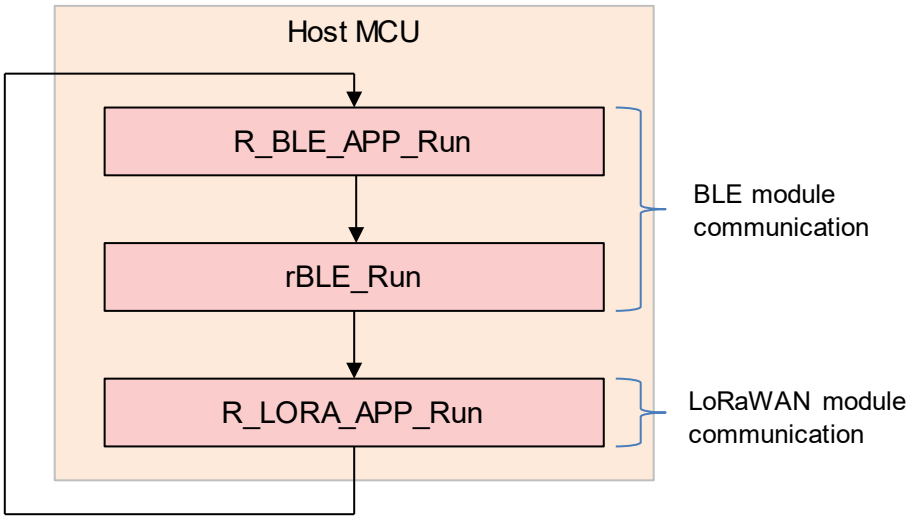


Figure 6-1 Main Loop Processing

The BLE module communication unit is executed by the R_BLE_APP_Run function and the rBLE_Run function. The R_BLE_APP_Run function is an application part that connects to a smartphone to receive LoRaWAN module setting parameters, converts them to AT commands, and sets parameters in the LoRaWAN module. The rBLE_Run function performs the process of sending the rBLE API called in the application part to the BLE module and the process of receiving the rBLE event from the BLE module and notifying the callback function of the application part.

The LoRaWAN module communication part is executed by R_LORA_APP_Run function. The sensor data measured from the Sensor module is read out and sent to the LoRaWAN module using AT commands, and the communication termination command received from the LoRaWAN Gateway is processed.

6.2 BLE module Communication Flow

The flow for setting the parameters received from the smartphone to the LoRaWAN module is explained in "Figure 6-2 BLE module Communication Process".

6.2.1 Connection with Smartphone

The BLE module communication is executed by the R_BLE_APP_Run function and rBLE_Run function in the main loop.

The R_BLE_APP_Run function is an application part that connects to a smartphone, receives LoRaWAN module setting parameters, and sets parameters to the LoRaWAN module with AT commands. The rBLE_Run function performs the process of sending the rBLE API called in the application part to the rBLE module and the process of receiving the rBLE event from the BLE module and passing it to the callback function of the application part.

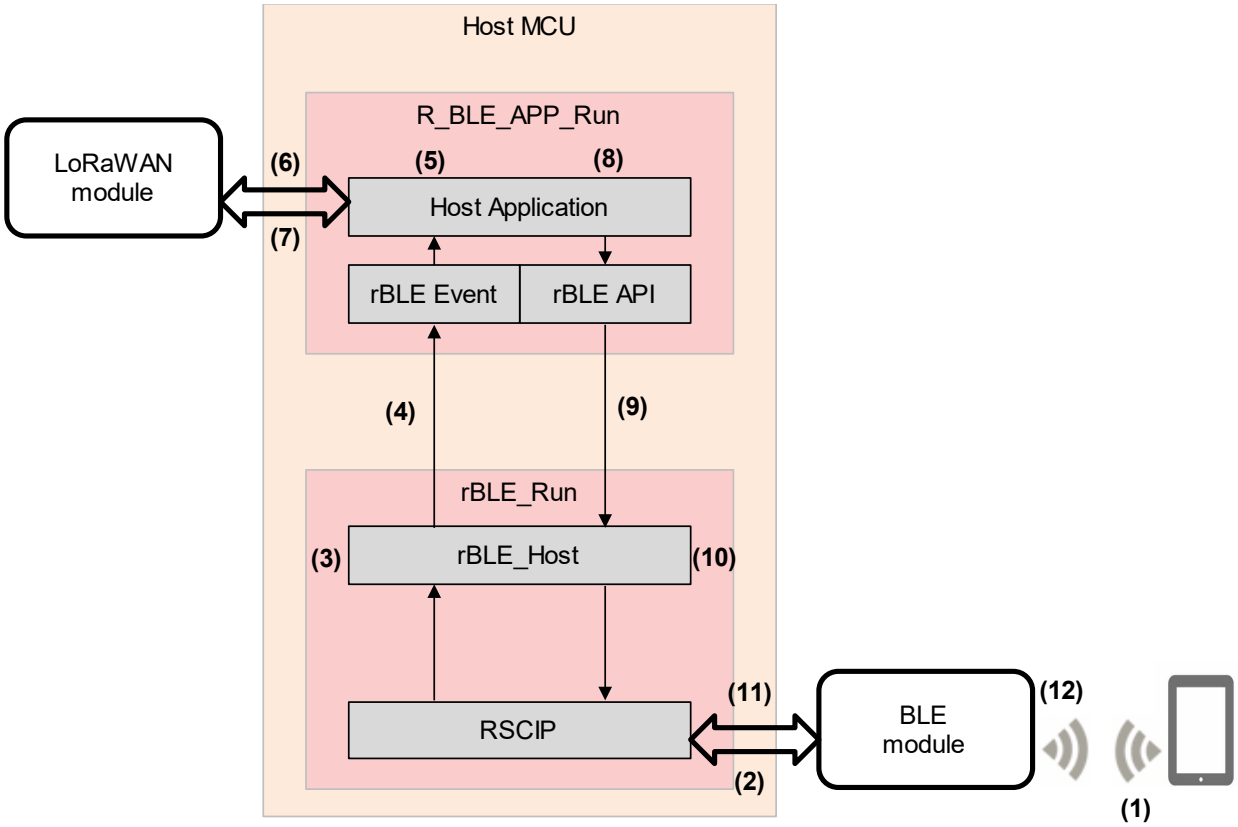


Figure 6-2 BLE module Communication Process

(1) The smartphone sends LoRaWAN configuration parameters.

(2) When the BLE module receives the LoRaWAN configuration parameters from the smartphone, it sends the rBLE event format via RSCIP and is passed from the Host MCU RSCIP to rBLE_Host.

(3) rBLE_Host converts rBLE event format to rBLE event.

(4) rBLE_Host notifies the rBLE event by calling the callback function of Host Application.

(5) Host Application extracts the hexadecimal LoRaWAN module configuration parameter from the rBLE event and converts it to an AT command string.

(6) Host Application sends AT command to LoRaWAN module to set parameters.

(7) Host Application receives the AT command execution result string from the LoRaWAN module.

(8) Host Application converts the AT command execution result string to hexadecimal format.

(9) Host Application calls rBLE API.

(10) rBLE_Host converts rBLE API and parameters specified by API into rBLE command format.

(11) RSCIP sends the rBLE command format to the BLE module.

(12) The AT command execution result is sent from the BLE module to the smartphone.

6.2.2 Disconnection with Smartphone

When the BLE module detects that the smartphone has been disconnected, the flag indicating the parameter set in the LoRaWAN module is saved in the FPB data flash. If the FPB power is turned off or E2Lite is disconnected without executing the disconnection from the smartphone, the parameter flag is not saved in the FPB data flash. Connect to the smartphone and set the parameters again.

6.3 LoRaWAN module Communication Flow

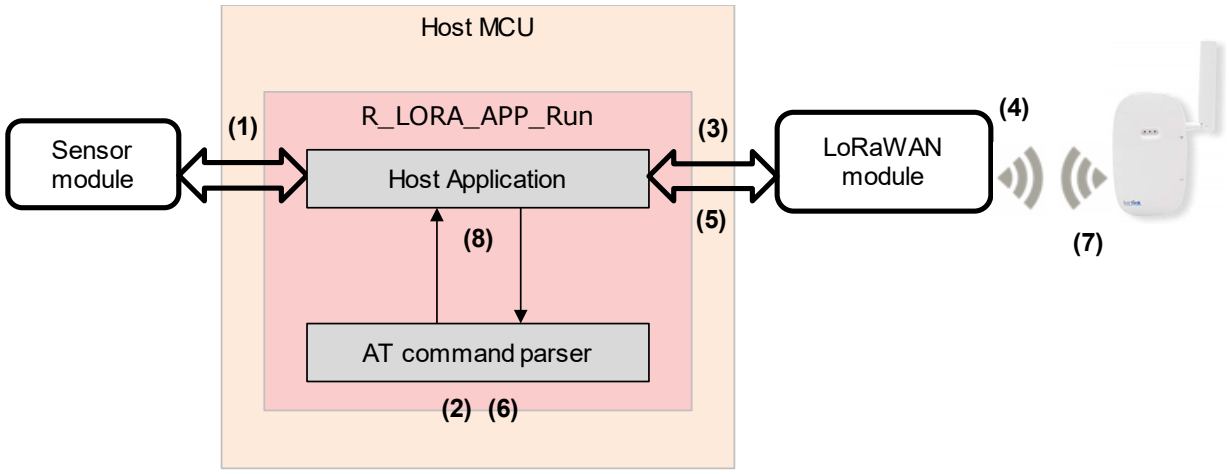


Figure 6-3 LoRaWAN module Communication Process

[Send a sensor data - Uplink]

(1) Host Application reads the sensor data from the Sensor module and passes it to the AT command parser.

(2) The AT command parser generates an AT command that includes sensor data read by the Host Application and passes it to the Host Application.

(3) Host Application sends AT command to LoRaWAN module.

(4) LoRaWAN module sends sensor data to LoRaWAN Gateway.

(5) Host Application receives the sensor data transmission result and passes it to AT command parser.

(6) The AT command parser analyzes the sensor data transmission result and notifies the Host Application.

[Receive the LoRaWAN Gateway communication end command - Downlink]

(7) When LoRaWAN Gateway communication termination command is sent from LoRaWAN Network Server, LoRaWAN module is received through LoRaWAN Gateway and passed to Host Application.

(8) Host Application analyzes using AT command parser and terminates communication with LoRaWAN Gateway.

6.4 LoRaWAN module Commands

This section describes the AT command communication format specification used to set parameters NOTE for the LoRaWAN module and the command specification for end communication with the LoRaWAN Gateway.

In this application, 7 AT commands are used to set parameters in LoRaWAN module. And, one unique command is used to end communication with the LoRaWAN Gateway.

Refer to “LoRaWAN Stack Sample Application Command Reference” (R11AN0231) for detailed explanation of AT commands.

Note: In this application note, parameters such as Device EUI are set because the LoRaWAN evaluation board (LoRaWAN stack sample application board) is used. Usually, two parameters that need to be set are Application Key and Application EUI. Other parameters are often written as device-specific parameters.

Table 6-1 LoRaWAN module Setting AT Commands

AT Command	Description
AT+REGION	Set Region
AT+DEVEUI	Set Device EUI
AT+CLASS	Set Device class
AT+APPEUI	Set Application identifier
AT+APPKEY	Set Application key
AT+ACTMODE	Set Activation mode
AT+FPORT	Set Port number

6.4.1 Parameter Setting AT Commands

This section describes the format of the AT command sent from the smartphone, the format of the AT command execution result sent to the smartphone, and the parameter flag format for storing the set parameters in the FPB (RL78/G14) data flash.

6.4.1.1 AT Command Format

The parameters set for the LoRaWAN module are sent from the smartphone.

The AT command and parameters sent to the LoRaWAN module are ASCII character strings, but the character strings may be longer depending on the parameters. Therefore, use a hexadecimal numeric string that contains the number and parameters assigned to each AT command, not an ASCII string.

Hexadecimal parameters received by the BLE module are converted to ASCII string AT commands by the FPB Host Application and sent to the LoRaWAN module.

The table below shows the correspondence between the hexadecimal AT command format sent from the smartphone and the character string to be converted after being received by the BLE module.

Hexadecimal AT command format: <AT command number> <Parameter>

AT command number: 1 byte

Parameter: Variable length

Table 6-2 AT Commands Number Correspondence Table

AT Command Number	AT Command
01	AT+REGION
02	AT+DEVEUI
03	AT+CLASS
04	AT+APPEUI
05	AT+APPKEY
06	AT+ACTMODE
07	AT+FPORT

Table 6-3 Correspondence Table between AT Command Numeric String and AT Command String

AT Command numeric string Note1	AT Command string
0106	AT+REGION=06
02749050FFFE000C2C	AT+DEVEUI=749050FFFE000C2C
0300	AT+CLASS=00
040123456701234567	AT+APPEUI=0123456701234567
055555555555555555AAAAAAAAAAAAAAAA	AT+APPKEY=55555555555555555AAAAAAAAAAAAAAAA
0601	AT+ACTMODE=01
0710	AT+FPORT=10

Notes: 1. The first byte is the AT command number, and the rest is the parameter. The parameters vary depending on the communication environment with LoRaWAN Gateway.

6.4.1.2 AT Command Result Code Format

When AT command is sent to LoRaWAN module, the execution result is returned as ASCII character string. The FPB Host Application converts the execution result from an ASCII character string to a hexadecimal numeric string and sends it from the BLE module to the smartphone.

The format of the AT command execution result sent from the smartphone is shown below.

Hexadecimal AT command result format:

<AT command number Note1> <AT command execution result Note2> <Parameter flag Note3>

Notes: 1. Refer to "Table 6-2 AT Commands Number Correspondence Table".
2. Refer to "Table 6-4 AT Command Execution Result".
3. Refer to "Table 6-5 AT Command Parameter Flag".

Table 6-4 AT Command Execution Result

AT Command execution result (ASCII)	AT Command execution result (Hexadecimal)
OK	01
ERROR	00

6.4.1.3 AT Command Parameter Flag

The AT command execution result set to the LoRaWAN module is saved in the FPB (RL78 / G14) data flash as a bit field parameter flag. The parameter flag is read at the start of the program to determine whether LoRaWAN module communication is possible. The AT command parameters set from the smartphone via BLE module communication are saved to the FPB data flash after the BLE module detects disconnection from the smartphone. If the FPB power is turned off or E2Lite is disconnected without disconnecting from the smartphone, the parameter flag is not saved in the FPB data flash, so set it again.

Table 6-5 AT Command Parameter Flag

bit	bit name
0	region
1	deveui
2	class
3	appeui
4	appkey
5	actmode
6	fport
7	-

6.4.2 LoRaWAN Gateway End of Communication Command

This command is sent from LoRaWAN Network Server to terminate sensor data communication with LoRaWAN Gateway. When LoRaWAN module receives the command, Host Application finishes sending sensor data and enters IDLE state.

Communication end command: 11223344

6.5 BLE Modem Structure - UART 2-wire with branch connection

This section describes the UART 2-wire branch connection method used for serial communication with the BLE module. Refer to "Table 3-1 Pin Assignment of PMOD1" for connection of FPB and BLE module using UART 2-wire branch connection method.

6.5.1 Transmission Process

To perform transmission from the FPB (Host MCU) to the BLE module (BLE MCU), handshake is required. A handshake is performed by send REQ byte (0xC0) from the Host MCU and send ACK byte (0x88) or RSCIP packet from the BLE MCU. When handshaking is performed, monitoring is performed by a timer, and when a timeout occurs, handshaking is re-executed. The Host MCU of UART driver for performing a handshake, it has a 5 state by the transmission status.

Table 6-6 UART driver transmission state

STATE	Description
T_IDLE	Initialize UART driver. RSCIP packet transmission completion.
T_REQUESTING	During REQ byte transmission.
T_RCV_BF_REQUESTED	Receive RSCIP packet from the module instead of ACK bytes.
T_REQUESTED	REQ byte transmission completion. (Wait for the ACK byte from the module)
T_ACTIVE	During RSCIP packet transmission.

Transmission from the Host MCU to the BLE MCU, always start with REQ byte. After sending the REQ byte, Host MCU branches to one of the following operations by the receiving state.

(a) The Host MCU has not received RSCIP packet from the BLE MCU (Figure 6-4)

(b) The Host MCU is receiving RSCIP packet from the BLE MCU (Figure 6-5)

(c) ACK byte reception time-out (Figure 6-6)

(a) The Host MCU has not received RSCIP packet from the BLE MCU

This state is RSCIP packet has not been transmitted from the BLE MCU, after sending the REQ byte from the Host MCU, the Host MCU is waiting to receive an ACK byte. The BLE MCU receives REQ byte and sends ACK byte. The Host MCU which received ACK byte sends RSCIP packet to the BLE MCU.

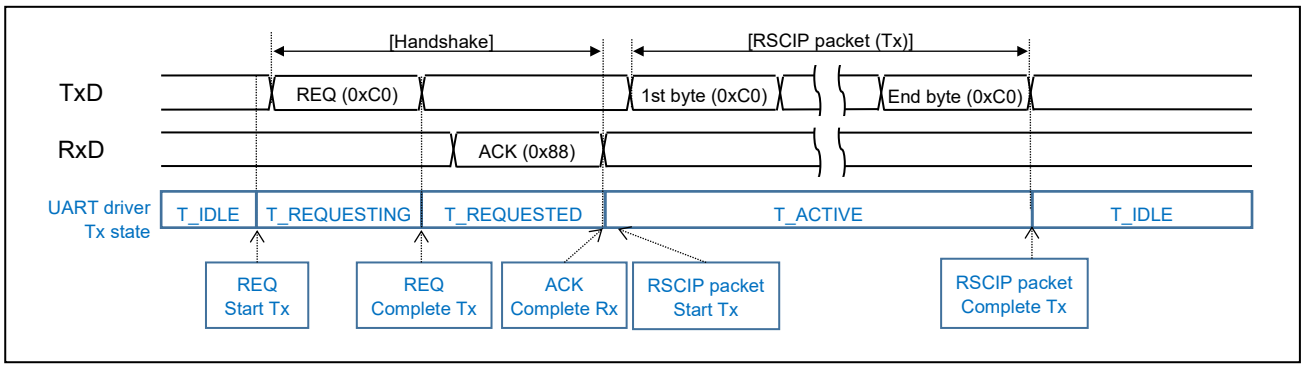


Figure 6-4 The Host MCU has not received RSCIP packet from the BLE MCU

(b) The Host MCU is receiving RSCIP packet form the BLE MCU

In this state, the BLE MCU is sending RSCIP packets, and the Host MCU is receiving RSCIP packets. Even if a module receives REQ, ACK byte isn't returned. RSCIP packet which is being sent is made a substitute of ACK byte. The Host MCU regards RSCIP packet from the BLE MCU as a substitute of ACK byte. And RSCIP packet is sent to the BLE MCU.

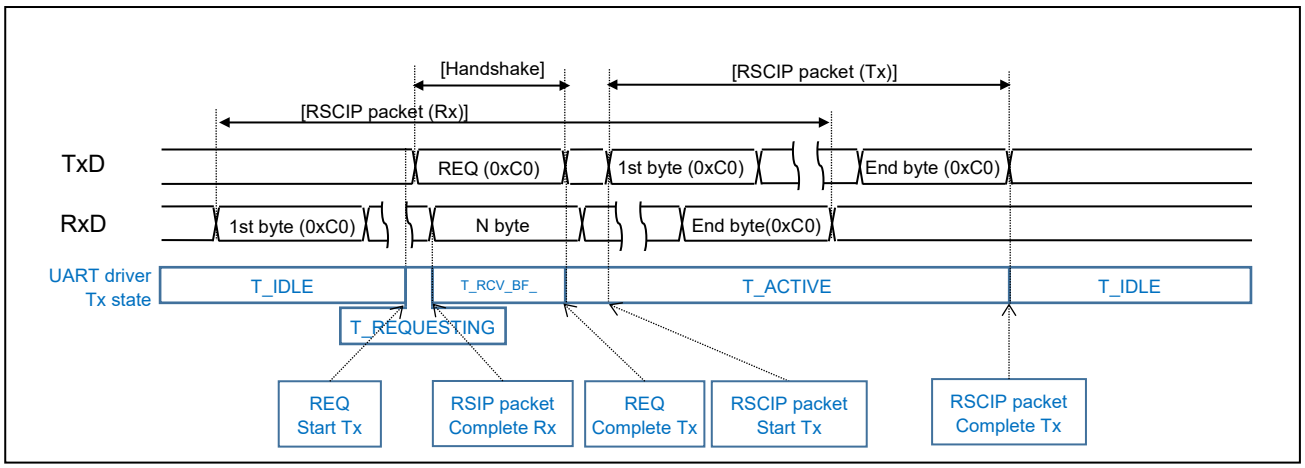


Figure 6-5 The Host MCU is receiving RSCIP packet from the module

(c) ACK byte reception time-out

After sending REQ byte, the Host MCU starts a timeout timer. If the ACK byte is not received for a certain time, and then the REQ byte is retransmitted.

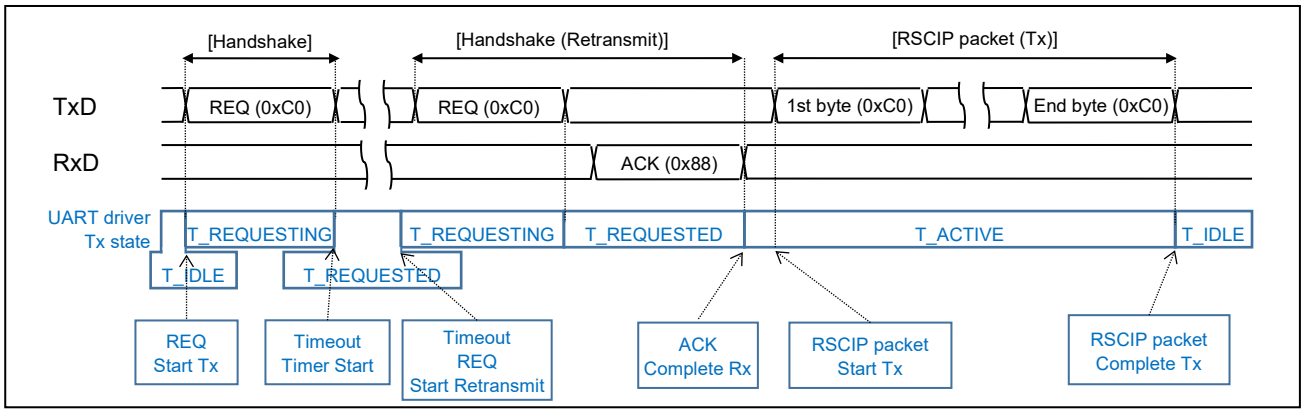


Figure 6-6 ACK byte reception time-out

6.5.2 Reception Process

There is no state transition of a UART driver at the reception. In order to receive the data from the BLE MCU, it listens for RSCIP packet from the BLE MCU in the specified number of bytes from the rBLE_Host.

7. BLE Sequence Chart

The communication sequence of Local Device consisting of Host MCU and BLE MCU and Remote Device such as smartphone is shown.

7.1 Main sequence chart

In the Main Sequence Chart, the processing blocks of 10 steps are shown. The detail of each processing block is shown in following sections.

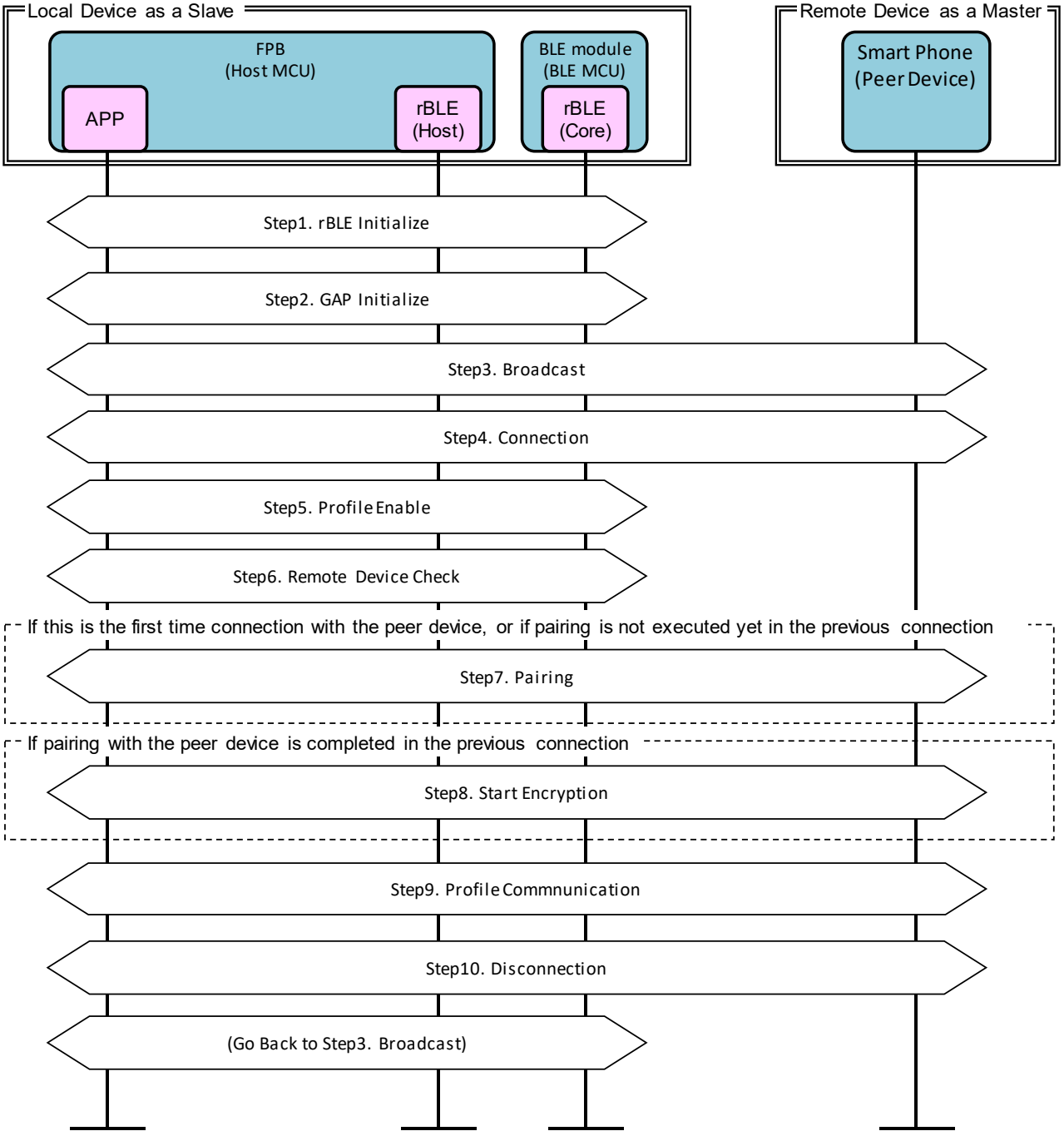


Figure 7-1 Main sequence chart

7.2 Step1. rBLE Initialize sequence

APP calls "RBLE_Init" function to initialize rBLE (rBLE_Host and rBLE_Core). After initializing rBLE and establishing communication to BLE MCU, rBLE informs "RBLE_MODE_ACTIVE" event.

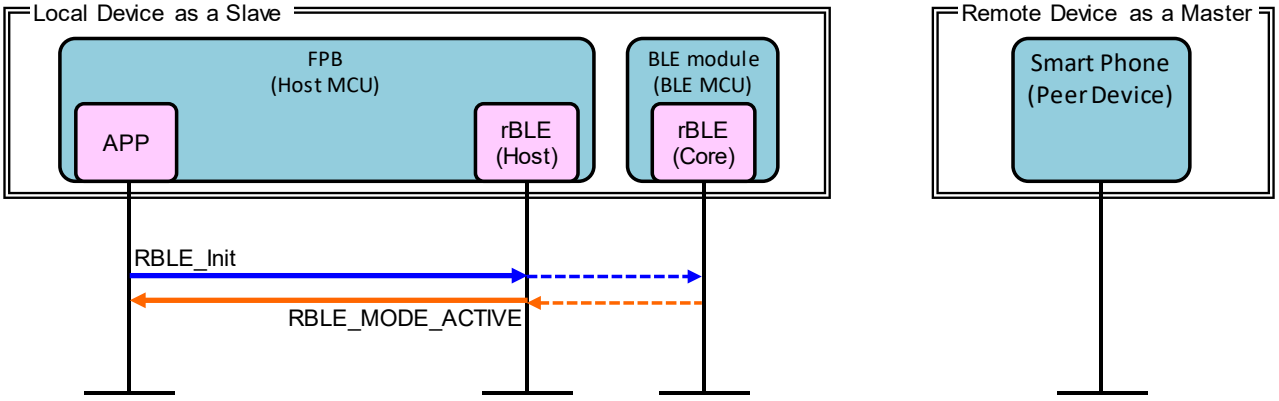


Figure 7-2 rBLE Initialize sequence chart

7.3 Step2. GAP Initialize sequence

APP calls "RBLE_GAP_Reset" function to reset GAP. After resetting rBLE, rBLE informs "RBLE_GAP_EVENT_RESET_RESULT" event.

APP calls "RBLE_GAP_Set_Bonding_Mode" function to permit the bonding with remote device. After setting the permission, rBLE informs "RBLE_GAP_EVENT_SET_BONDING_MODE_COMP" event.

APP calls "RBLE_GAP_Set_Security_Request" function to set security level. After setting security level, rBLE informs "RBLE_GAP_EVENT_SET_SECURITY_REQUESET_COMP" event.

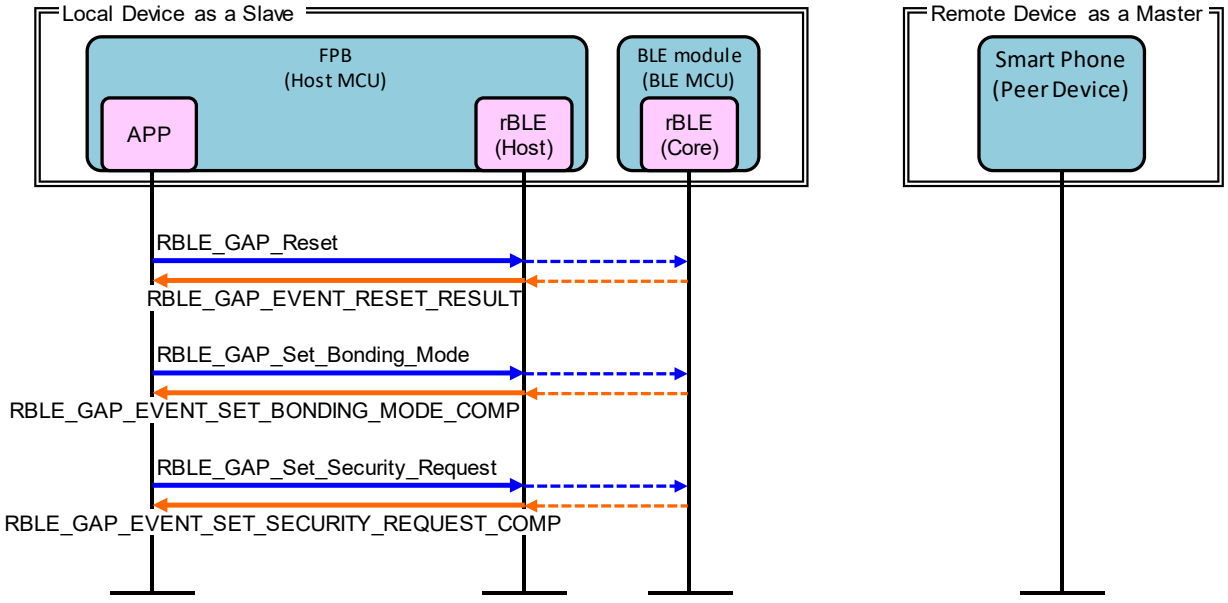


Figure 7-3 GAP Initialize sequence chart

7.4 Step3. Broadcast sequence

Local Device starts broadcasting to establish connection as a slave.

APP calls "RBLE_GAP_Broacast_Enable" function to start broadcasting. After starting the broadcast, rBLE informs "RBLE_GAP_EVENT_BROADCAST_ENABLE_COMP" event.

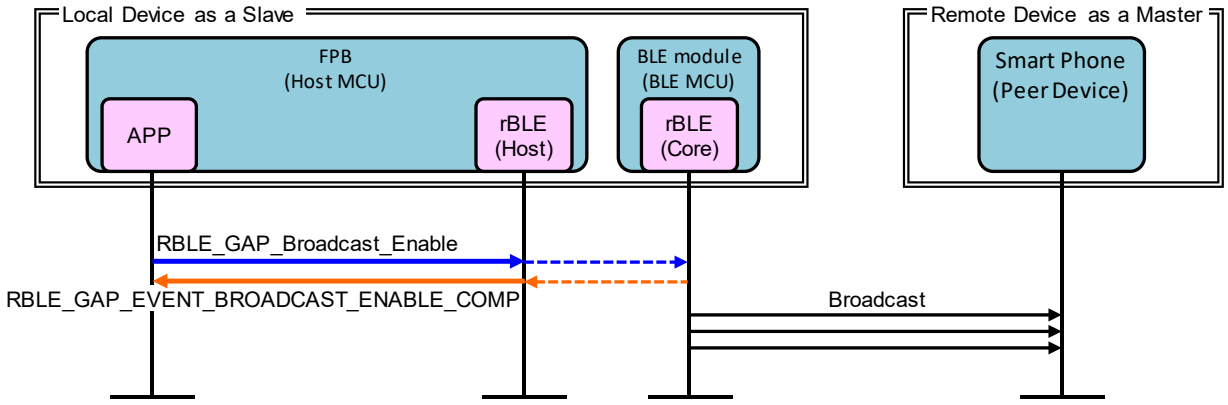


Figure 7-4 Broadcast sequence chart

7.5 Step4. Connection sequence

Remote Device receives the broadcast and requests to establish connection with Local Device.

If the connection between Remote Device and Local Device is established by receiving Connection Request from Remote Device, rBLE informs "RBLE_GAP_EVENT_CONNECTION_COMP" event.

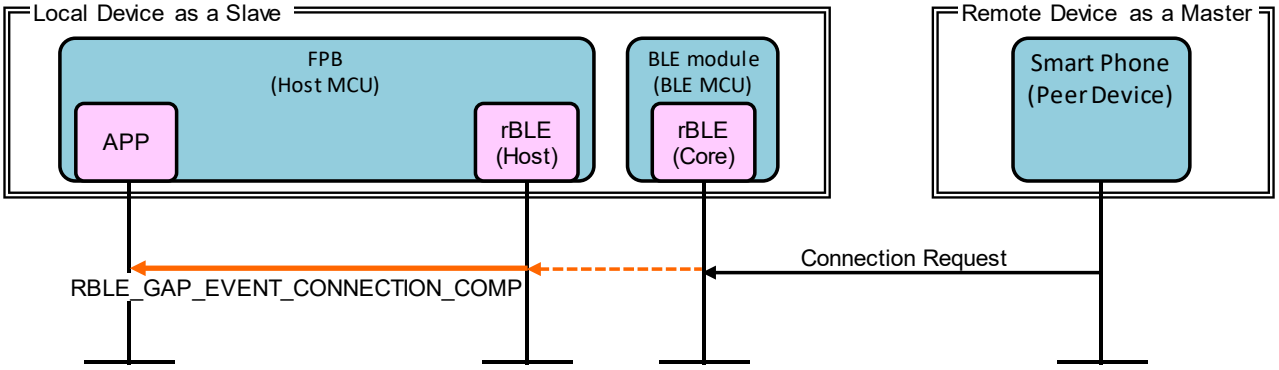


Figure 7-5 Connection sequence chart

7.6 Step5. Profile Enable sequence

Local Device enables GPCP (General Purpose Communication Profile) to send data.

APP calls "RBLE_VUART_Server_Enable" function to enable GPCP. Enabling is complete when the Remote Device sends a Write Client Characteristic Configuration that allows Indication. Refer to "Figure 7-10 Profile Communication sequence chart".

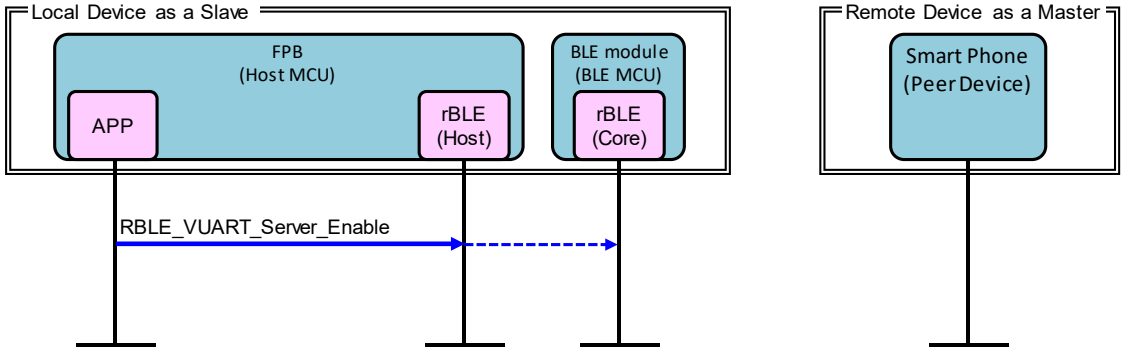


Figure 7-6 Profile Enable sequence chart

7.7 Step6. Remote Device Check sequence

Local Device confirms security information about Remote Device.

If the device address of Remote Device is public address or if it is random address except resolvable private address, rBLE informs "RBLE_SM_CHK_BD_ADDR_REQ" event to acquire security information about Remote Device. APP calls "RBLE_SM_Chk_Bd_Addr_Req_Resp" function to inform security information.

If the device address of Remote Device is resolvable private address, rBLE informs "BLE_SM_IRK_REQ_IND" event to acquire IRK (Identify Resolving Key) which is used for resolving address. APP calls "RBLE_SM_Irk_Req_Resp" function to informs whether to have IRK or not and informs IRK. If resolving address is success, rBLE informs "RBLE_GAP_EVENT_RPA_RESOLVED" event. If resolving address is failed, rBLE informs "RBLE_SM_IRK_REQ_IND" event repeatedly until it is successful or until all of IRK which APP possess is checked.

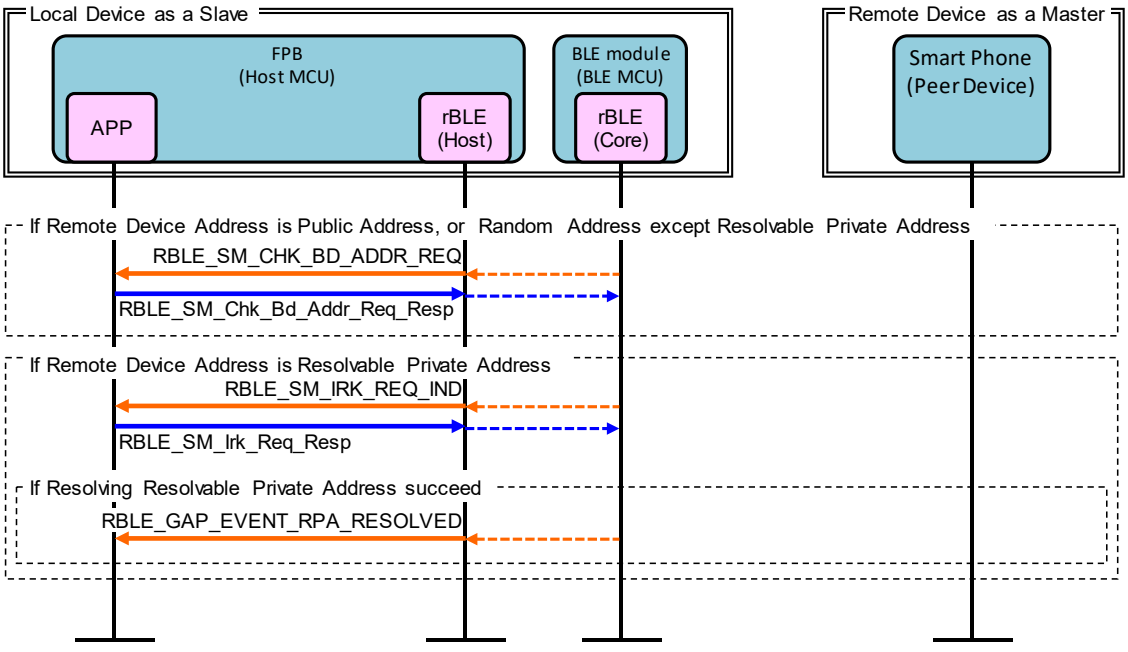


Figure 7-7 Remote Device Check sequence chart

7.8 Step7. Pairing sequence

If the connection with the Remote Device is first time or if pairing is not executed in previous connection, Local Device starts pairing sequence by request from Remote Device. Pairing sequence is consisted of PHASE1, PHASE2, starting encryption and PHASE3.

PHASE1 is for exchanging the pairing features between Local Device and Remote Device.

If Local Device receives Pairing Request from Remote Device, rBLE informs "RBLE_GAP_EVENT_BONDING_REQ_IND" event. APP calls "RBLE_GAP_Bonding_Response" function to send Pairing Response.

PHASE2 is for generating STK (Short Term Key).

rBLE informs "RBLE_SM_TK_REQ_IND" event to acquire TK (Temporary Key). APP calls "RBLE_SM_Tk_Req_Resp" function to inform TK. After generating STK by BLE_MCU, Local Device and Remote Device start encrypting the contents of communication.

PHASE3 is for distributing encryption keys of Local Device and Remote Device.

rBLE informs "RBLE_SM_LTK_REQ_IND" event to acquire LTK (Long Term Key). APP calls "RBLE_SM_Ltk_Req_Resp" function to inform LTK and send Encryption Information (LTK).

By receiving Encryption Information (LTK) from Remote Device, rBLE informs "RBLE_SM_KEY_IND" event.

By receiving Identity Information (IRK) from Remote Device, rBLE informs "RBLE_SM_KEY_IND" event.

If pairing sequence is success, rBLE informs "RBLE_GAP_EVENT_BONDING_COMP" event.

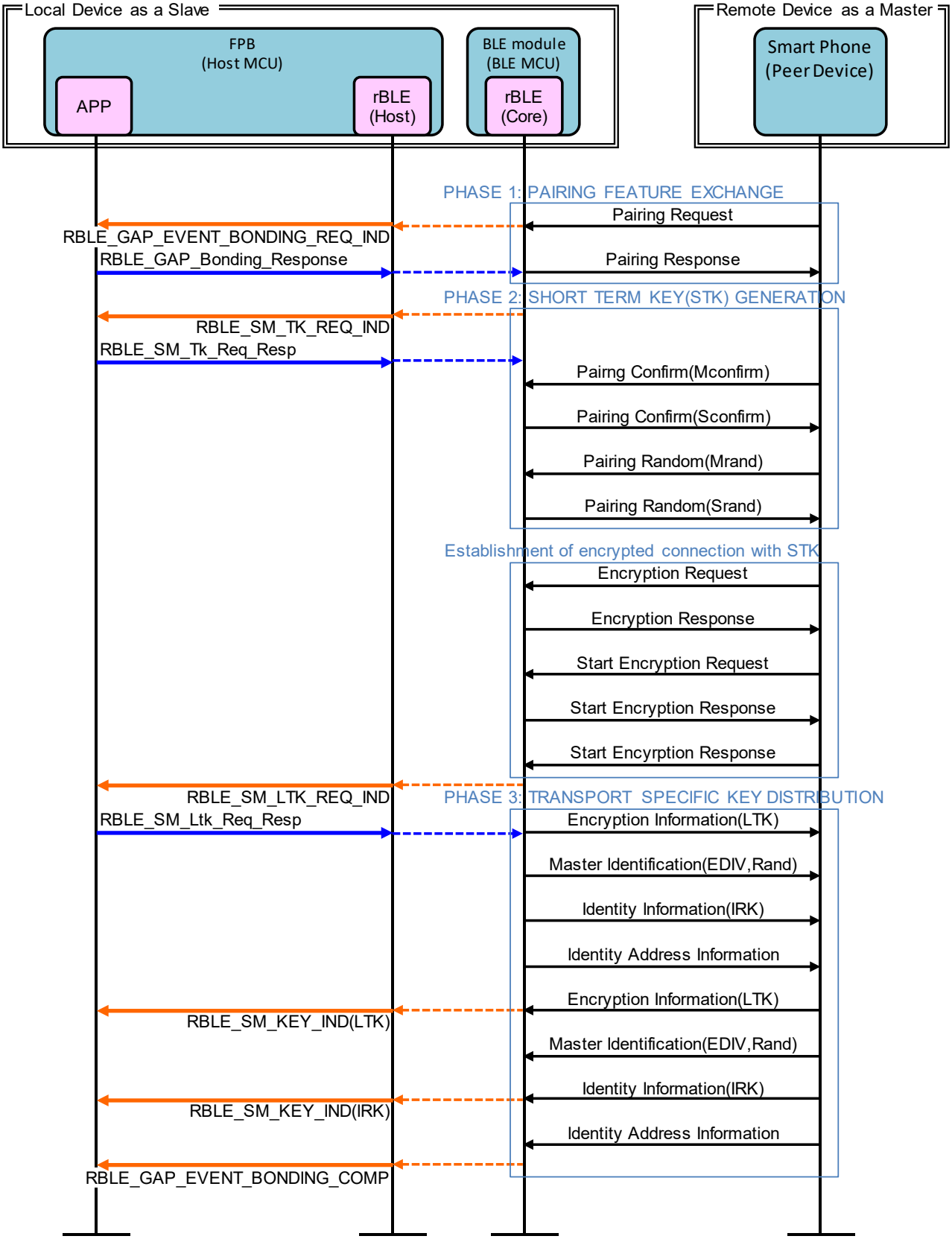


Figure 7-8 Pairing sequence chart

7.9 Step8. Start Encryption sequence

If pairing is success in previous connection, Local Device starts encryption sequence with LTK (Long Term Key) by request from Remote Device.

By receiving Encryption Request from Remote Device, rBLE informs "RBLE_SM_LTK_REQ_FOR_ENC_IND" event. APP calls "RBLE_SM_Ltk_Req_Resp" function to inform LTK and send Encryption Response.

By receiving Start Encryption Request, BLE MCU of Local Device sends Start Encryption Response.

If start encryption sequence is success, rBLE informs "RBLE_SM_ENC_START_IND" event.

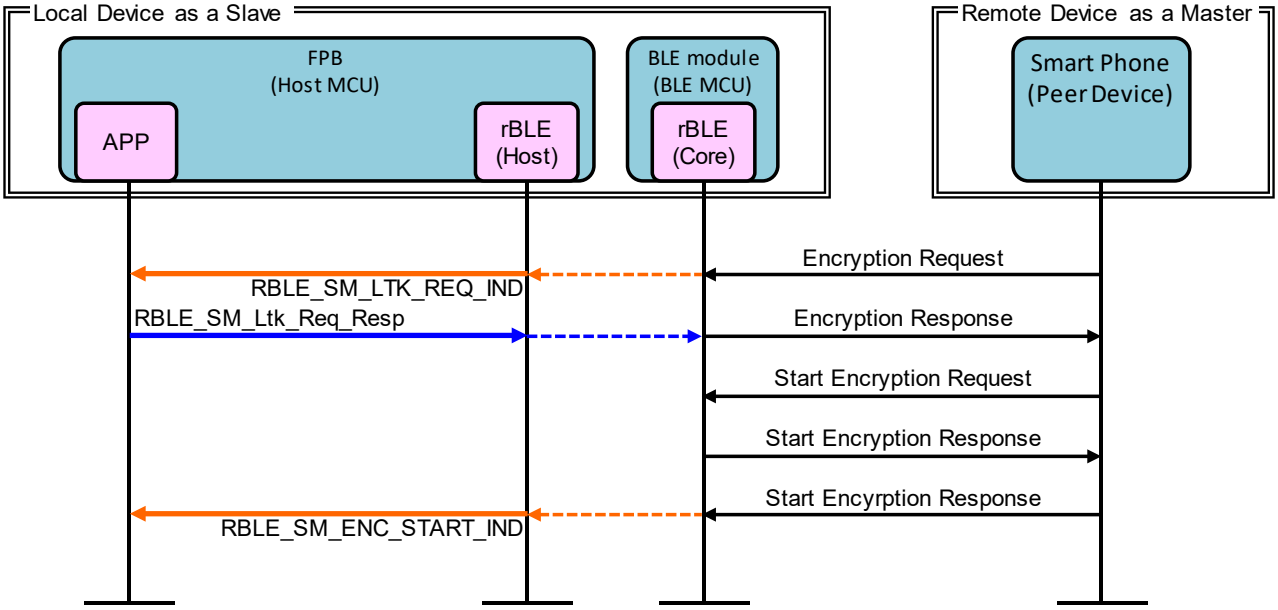


Figure 7-9 Start Encryption sequence chart

7.10 Step9. Profile Communication sequence

Use GPCP (General Purpose Communication Profile) to send parameters to be set in the LoRaWAN module.

When the Write Client Characteristic Configuration that allows indication from Remote Device is sent, informs RBLE_VUART_EVENT_SERVER_ENABLE_COMP event from rBLE(Host).

When a parameter to be set to LoRaWAN module is sent from Remote Device, RBLE_VUART_EVENT_SERVER_WRITE_REQ is generated and received by Local Device. Local Device set the parameter to LoRaWAN module.

The parameter setting result is sent as Indication data to the Remote Device. When Remote Device receives Indication, it sends Confirmation. When Local Device receives Confirmation, RBLE_VUART_EVENT_SERVER_INDICATION_CFM event is notified.

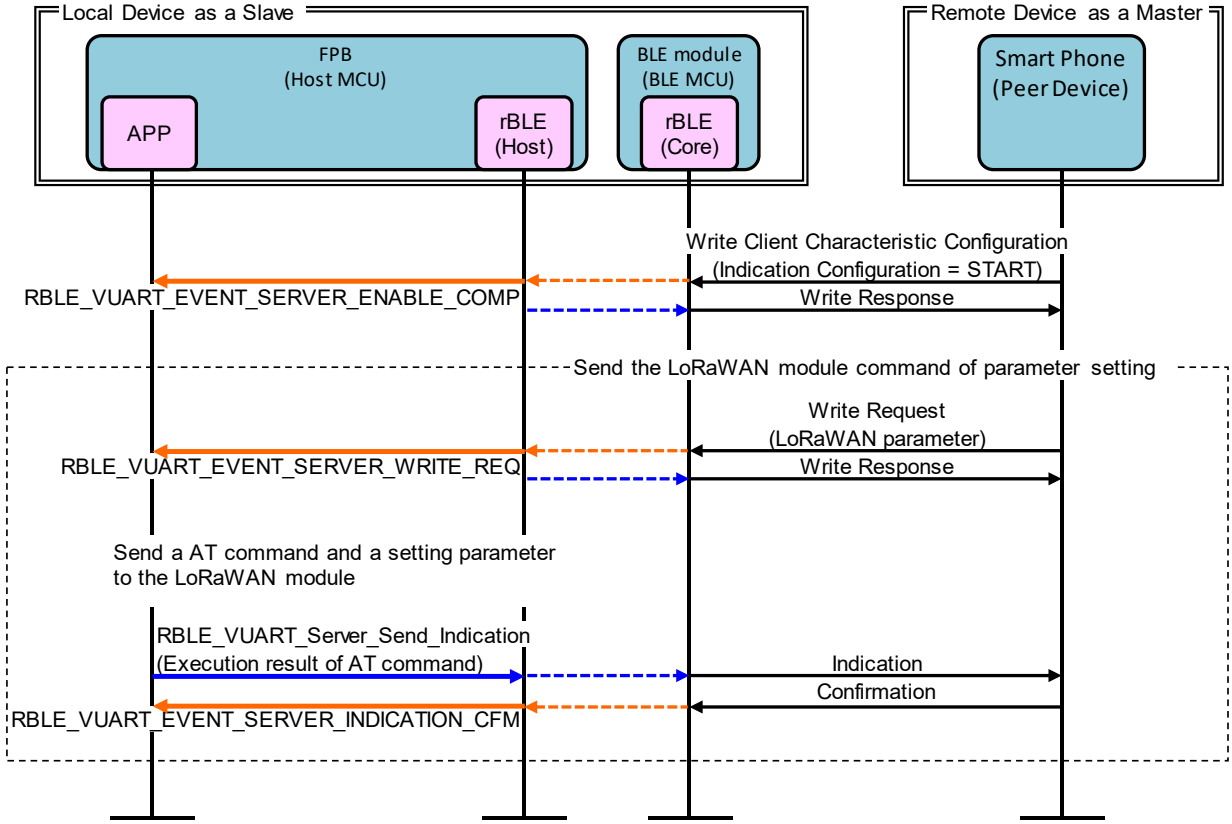


Figure 7-10 Profile Communication sequence chart

7.11 Step10. Disconnection sequence

By receiving Disconnect from Remote Device, rBLE disconnects connection and informs "RBLE_GAP_EVENT_DISCONNECT_COMP" event.

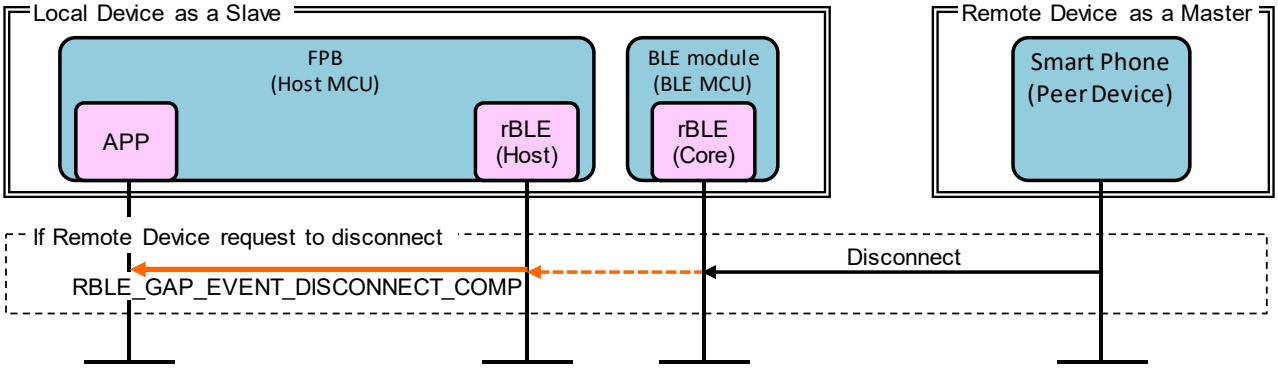


Figure 7-11 Disconnection sequence chart

8. LoRaWAN module AT Command Sequence Chart

This chapter describes the AT command sequence that communicates with FPB Host Application and LoRaWAN module.

8.1 Main Sequence

The AT command sequence is divided into processing blocks Steps 1 to 3.

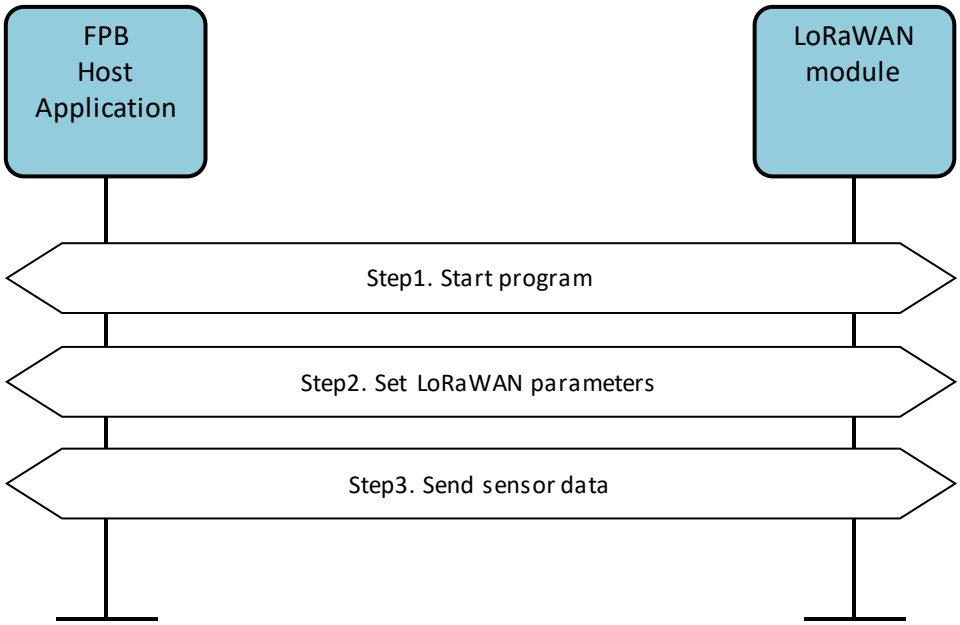


Figure 8-1 AT Command Main Sequence

8.2 Step1. Start program

Echo back from the LoRaWAN module is prohibited immediately after starting the program.

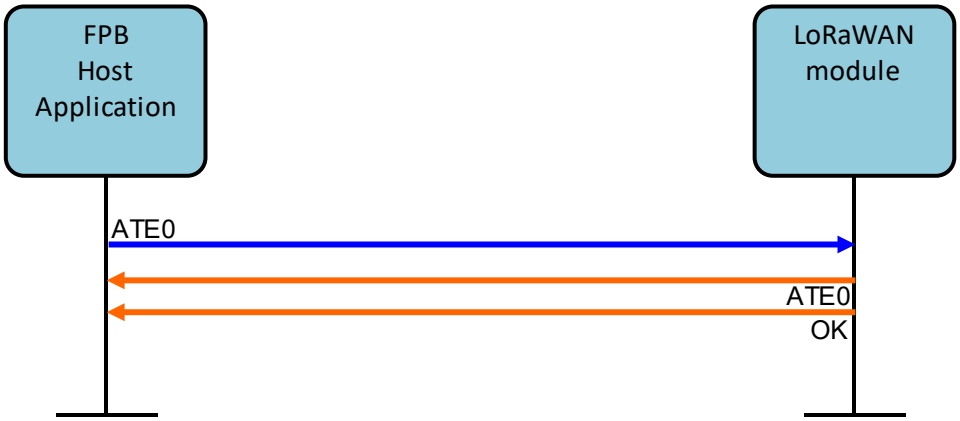


Figure 8-2 Start program

8.3 Step2. Set LoRaWAN parameters

Set parameters to LoRaWAN module with AT command.

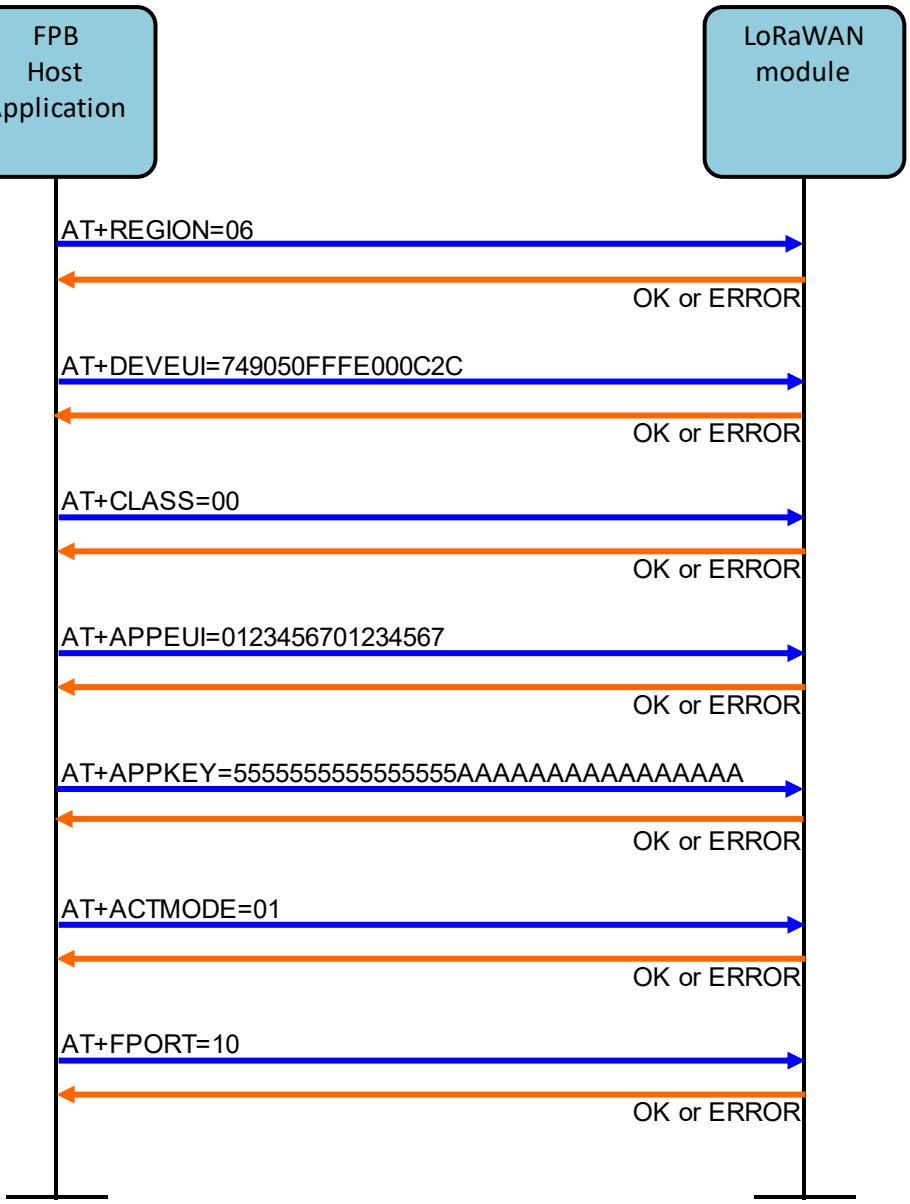


Figure 8-3 Set LoRaWAN parameters

8.4 Step3. Send sensor data

The AT command sequence for communicating with LoRaWAN Gateway is shown below. First, activate the LoRaWAN Gateway and start sending sensor data. To stop transmission, send a communication termination command from LoRaWAN Gateway.

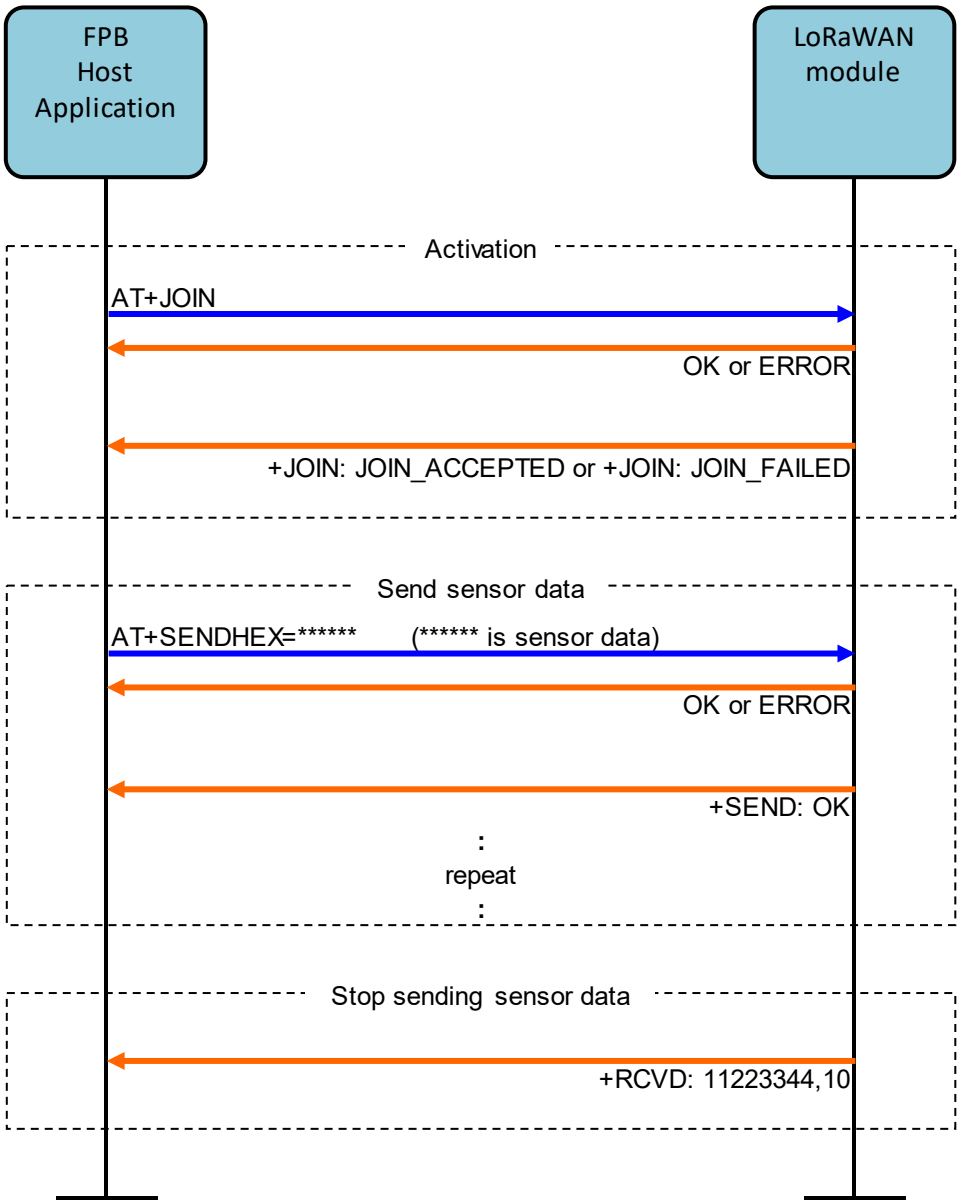


Figure 8-4 Send sensor data

9. Appendix

9.1 ROM size, RAM size

The ROM size and the RAM size which is used by this application is shown in "Table 9-1 ROM size, RAM size".

Table 9-1 ROM size, RAM size

Compiler	ROM (bytes)	RAM (bytes)
CC-RL V1.08	46,608	5,195

9.2 References

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

Revision History

Rev.	Date	Description	
		Page	Summary
1.0	Dec.25.19	-	First edition issued

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins
   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.