RL78/G1D Beacon Stack

Basic Operation Sample Program

R01AN3045EJ0220
Rev.2.20
Jun 19, 2020

Introduction

This Sample Program runs on Bluetooth® Low Energy microcontroller RL78/G1D device and includes applications such as Advertising as a beacon to broadcast information, Scanning to receive Advertising packets, and evaluating RF characteristic with Direct Test Mode (DTM).

By using DIP switch on RL78/G1D Evaluation Board, select to execute among on Beacon Application, Scan Application, or DTM Application. The Beacon Application transmits either Non-connectable Undirected Advertising packets or Scannable Undirected Advertising packets alternately by pushing switch on RL78/G1D Evaluation Board. Or, send ASCII-format UART commands from the host machine to control advertising packets. The Scan Application receives Advertising packets and forward through UART. The DTM Application executes either RF Receiver Test or RF Transmitter Test with respect to UART commands.

Target Device

RL78/G1D Evaluation Board (RTK0EN0001D01001BZ)

Related documents

Document Name		Document No.
RL78/G1D		
	User's Manual: Hardware	R01UH0515E
RL78/G1D Evaluation Board		
	User's Manual	R30UZ0048E
E1 Emulator		
	User's Manual	R20UT0398E
	Additional Document for User's Manual (Notes on Connection of RL78)	R20UT1994E
Renesas Flash Programmer V3.02 Flash memory programming software		
	User's Manual	R20UT3841E
CC-RL Compiler		
	User's Manual	R20UT3123E
RL78/G1D Beacon Stack		
	User's Manual	R01UW0171E

Contents

1. Overview

Figure 1-1 shows the architecture of the Sample Program, which consists of Beacon Application, Scan Application, DTM Application, and Beacon Stack. And the Sample Program works on RL78/G1D Evaluation Board.

- Beacon Application : executes Advertising for broadcasting information.
- Scan Application : executes Scanning for receiving Advertising packets.
- DTM Application : executes Direct Test Mode for evaluating RF characteristic.
- Beacon Stack : provides APIs to execute Advertising, Scanning, and Direct Test Mode.

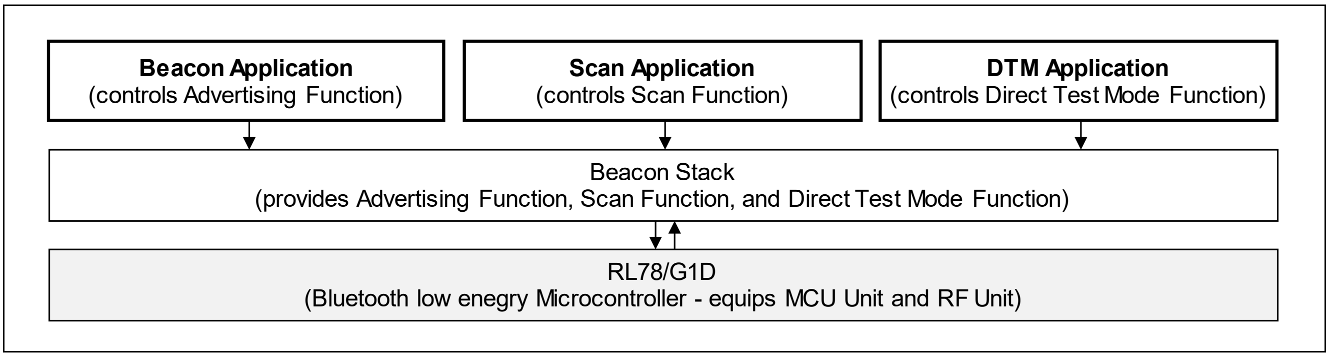

Beacon Application
(controls Advertising Function)
Scan Application
(controls Scan Function)
DTM Application
(controls Direct Test Mode Function)
Beacon Stack
(provides Advertising Function, Scan Function, and Direct Test Mode Function)
RL78/G1D
(Bluetooth low enegry Microcontroller - equips MCU Unit and RF Unit)

Figure 1-1 Architecture of Sample Program

Regarding to the specification of Beacon Application, Scan Application and DTM Application, refer to chapter 5 "Specification" in this document.

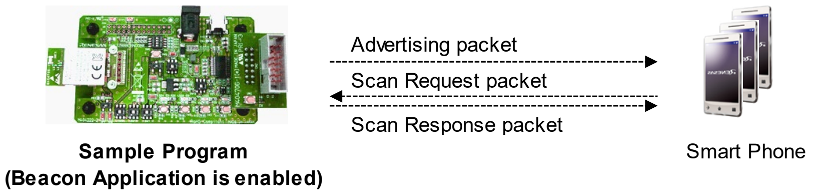
Regarding to the specification of Beacon Stack, refer to RL78/G1D Beacon Stack User's Manual (R01UW0171).

Regarding to the details about the evaluation board, refer to RL78/G1D Evaluation Board User's Manual (R30UZ0048).

1.1 Beacon Application

Figure 1-2 shows the overview of Beacon Application operating with the evaluation board and smart phone. To start Beacon Application, switch both DIP switch SW6 position-1 and position-4 to OFF and then power up.

After power up, Beacon Application starts transmitting Non-connectable Undirected Advertising packets. Pushing switch SW2 stops transmitting Non-connectable Undirected Advertising packets. Again, pushing switch SW2 in this state starts transmitting Scannable Undirected Advertising packets. Pushing switch SW2 in this state stops transmitting Scannable Undirected Advertising packets. Thus using switch SW2, it is possible to start and stop transmitting with different Advertising packets: Non-connectable Undirected Advertising packets or Scannable Undirected Advertising packets.

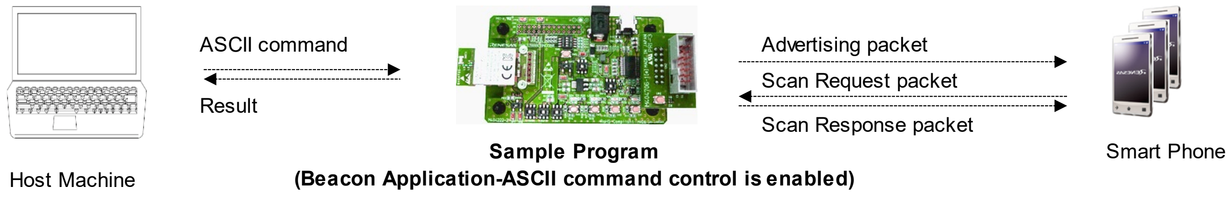


Beacon Application:
- Application starts Advertising automatically, then transmitts Non-connectable Undirected Advertising packet or Scannable Undirected Advertising packet periodically.

Figure 1-2 Overview of Beacon Application

1.2 Beacon Application-ASCII command control

Figure 1-3 shows the overview of Beacon Application-ASCII command control operating with the host machine, the evaluation board and smart phone. The Beacon Application-ASCII command control waits for a command from the host machine after starting. The beacon application is controlled by entering ASCII commands from the host machine. It is possible to send Non-connectable Undirected Advertising packets, send Scannable Undirected Advertising packets, and set parameters for Advertising operation. The Beacon Application-ASCII command control is selected with the APP_SELECT macro. For details on the APP_SELECT macro, refer to subsection 6.2.1 "Application Selection configuration".



Beacon Application:
- Non-connectable Undirected Advertising packet or Scannable Undirected Advertising packet is sent periodically by the control command received from the host machine.

Figure 1-3 Overview of Beacon Application-ASCII command control

1.3 Scan Application

Figure 1-4 shows the overview of Scan Application operating with beacon device, the evaluation board, and Host machine. Scan Application can communicate through UART in two formats: ASCII-format and Binary-format.

To start Scan Application and communicate through UART in ASCII-format, switch DIP switch SW6 position-1 to ON and position-4 to OFF and then power up. To start Scan Application and communicate through UART in Binary-format, switch both DIP switch SW6 position-1 and position-4 to ON and then power up.

ASCII-format UART Scan Application starts Scanning for receiving Advertising packets automatically after power up. When receive Advertising packet from beacon device, Scan Application sends ASCII-format report through UART. And, it is also possible to control operation of Scan Application by sending ASCII-format command through UART.

Binary-format UART Scan Application starts Scanning for receiving Advertising packets after receiving Start Scan command through UART. When receive Advertising packet from beacon device, Scan Application sends Binary-format report through UART. And, it is also possible to control operation of Scan Application by sending Binary-format command through UART.

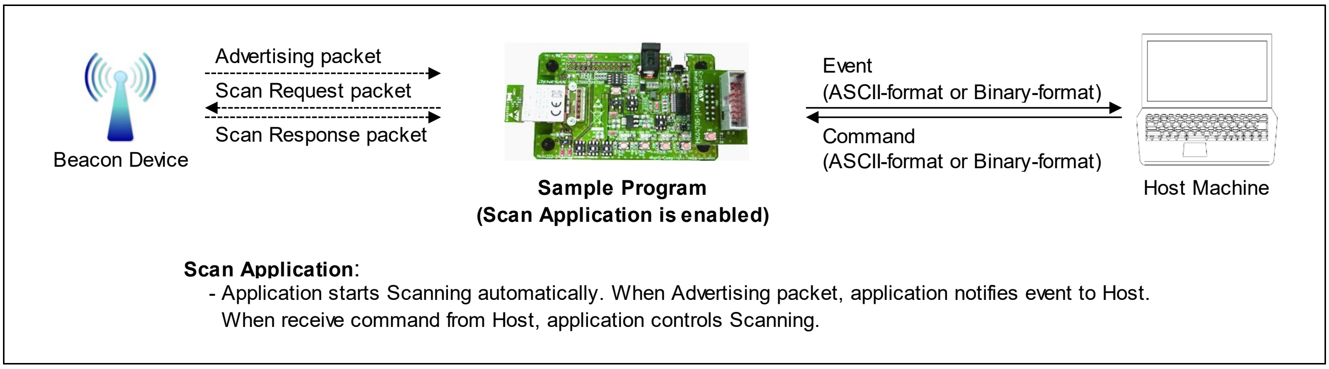

Advertising packet
Scan Request packet
Scan Response packet
Beacon Device
Sample Program
(Scan Application is enabled)
Event
(ASCII-format or Binary-format)
Command
(ASCII-format or Binary-format)
Host Machine
Scan Application:
- Application starts Scanning automatically. When Advertising packet, application notifies event to Host.
When receive command from Host, application controls Scanning.

Figure 1-4 Overview of Scan Application

1.4 DTM Application

Figure 1-5 shows the overview of DTM Application operating with the evaluation board and RF Tester.

To start DTM Application, switch DIP switch SW6 position-1 to OFF and position-4 to ON and then power up. By receiving RF Test command from RF Tester through UART, DTM Application starts Direct Test Mode. Then DTM Application sends test result as RF Test event through UART.

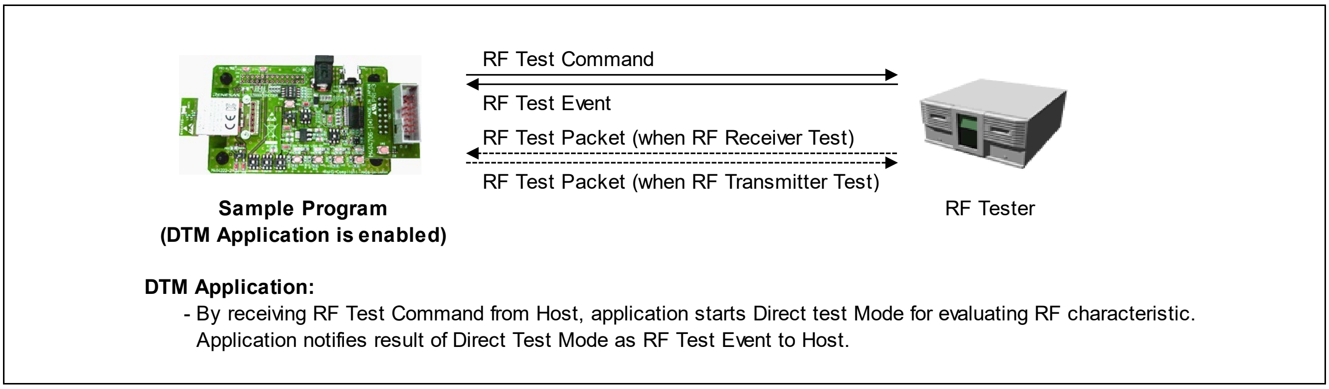

RF Test Command
RF Test Event
RF Test Packet (when RF Receiver Test)
RF Test Packet (when RF Transmitter Test)
Sample Program
(DTM Application is enabled)
RF Tester
DTM Application:
- By receiving RF Test Command from Host, application starts Direct test Mode for evaluating RF characteristic.
Application notifies result of Direct Test Mode as RF Test Event to Host.

Figure 1-5 Overview of DTM Application

2. Environment

For compiling and evaluating the Sample Program, the necessary hardware and software environment is as follow:

- Hardware Environment
  - Host
    - PC/AT^{TM} compatible computer
    - Processor : at least 1.6GHz
    - Main memory : at least 1Gbyte
    - Interface : USB2.0 (for connecting E1 Emulator and RL78/G1D Evaluation Board)
  - Device
    - RL78/G1D Evaluation Board (RTK0EN0001D01001BZ) : 2 boards
    - USB cable (A type male / mini-B type male) : 2 cables
    - iOS device or Android device

    Note: Two evaluation boards are needed when evaluate transmitting Advertising packets by an evaluation board and receiving the packets by other evaluation board.
  - Tool
    - Renesas On-chip Debugging Emulator E1 (R0E000010KCE00)
- Software Environment
    - Windows®7 Service Pack1 or later
    - Renesas CS+ for CC V8.03.00 / Renesas CC-RL V1.09.00
      or Renesas e^{2} studio Version 7.8.0 / Renesas CC-RL V1.09.00
    - Renesas Flash Programmer v3.06.01
    - Tera Term Pro (or Terminal software which can connect to serial port)
    - UART-USB conversion device driver

    Note: It may be that device driver for UART-USB conversion IC "FT232RL" is requested when connect first time RL78/G1D Evaluation Board to Host. In this case, you can get the device driver from below website.

    - FTDI (Future Technology Devices International) - Drivers
      http://www.ftdichip.com/Drivers/D2XX.htm

3. File Composition

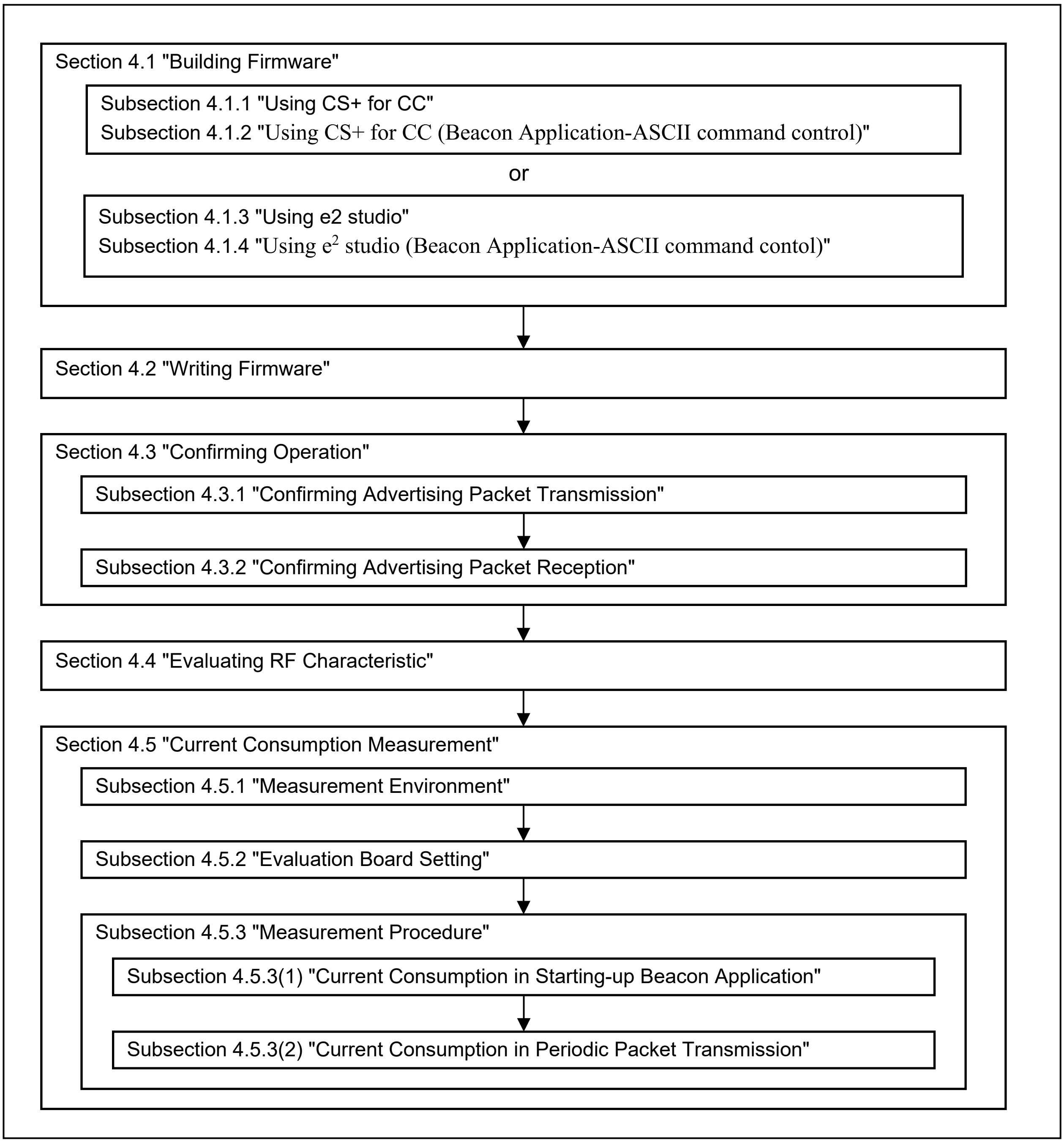
The Sample Program includes Beacon Stack library as well as the source code of Beacon Application, Beacon Application-ASCII command control, Scan Application and DTM Application.

In the release package, file and folder composition of the Sample Program is as shown below.

```
RL78G1D_Beacon
  ├ROM_File
  │     R5F11AGJ_Beacon.hex                    Sample Program - firmware file (R5F11AGJ)
  │                                            (all Application, switched by DIP SW)
  │     R5F11AGJ_Beacon(beacon).hex            Sample Program - firmware file (R5F11AGJ)
  │                                            (Beacon Application only)
  │     R5F11AGJ_Beacon(scan_ascii).hex        Sample Program - firmware file (R5F11AGJ)
  │                                            (ASCII-format Scan Application only)
  │     R5F11AGJ_Beacon(scan_bin).hex          Sample Program - firmware file (R5F11AGJ)
  │                                            (Binary-format Scan Application only)
  │     R5F11AGJ_Beacon(dtm).hex               Sample Program - firmware file (R5F11AGJ)
  │                                            (DTM Application only)
  │     R5F11AGJ_Beacon(beacon_ascii).hex      Sample Program - firmware file (R5F11AGJ)
  │                                            (Beacon Application-ASCII command control only)
  ├RUC_File
  │     r5f11agg_syscfg.ruc                    System Configuration - unique code file (R5F11AGG)
  │     r5f11agh_syscfg.ruc                    System Configuration - unique code file (R5F11AGH)
  │     r5f11agj_syscfg.ruc                    System Configuration - unique code file (R5F11AGJ)
  │     r5f11agg_syscfg.ruc                    System Configuration - unique code file (R5F11AGG)
  │                                            (for Beacon Application-ASCII command control)
  │     r5f11agh_syscfg.ruc                    System Configuration - unique code file (R5F11AGH)
  │                                            (for Beacon Application-ASCII command control)
  │     r5f11agh_syscfg.ruc                    System Configuration - unique code file (R5F11AGJ)
  │                                            (for Beacon Application-ASCII command control)
  └Project_Source
      ├library
      │  │  r_arch.h                           architecture - header file
      │  │  r_compiler.h                       compiler dependent part - header file
      │  │  r_iodefine.h                       SFR definition - header file for CC-RL
      │  │  r_ll.h                             low level built-in function - header file
      │  │  r_port.h                           port access - header file
      │  │  r_bcn_api.h                        Beacon Stack API - header file
      │  │  BLE_BEACON_CC.lib                  Beacon Stack - library for CC-RL
      │  └codeflash
      │     fsl.h                              Flash self-programming library - header file
      │     fsl_types.h                        Flash self-programming library - header file
      │     fsl.lib                            Flash self-programming library - library for CC-RL
      └application
          ├src
          │  │  cstart.asm                     start-up - assembly file for CC-RL
          │  │  r_config.h                     configuration - header file
          │  │  r_main.c                       entry point - header file
          │  │  r_interrupt.c                  interrupt - code file
          │  │  r_beacon.h                     Beacon Application - header file
          │  │  r_beacon.c                     Beacon Application - code file
          │  │  r_beacon_ascii.c               Beacon Application-ASCII command control - code file
          │  │  r_scan.h                       Scan Application - header file
          │  │  r_scan_ascii.c                 Scan Application (ASCII-format UART) - code file
          │  │  r_scan_bin.c                   Scan Application (Binary-format UART) - code file
          │  │  r_dtm.h                        DTM Application - header file
          │  │  r_dtm.c                        DTM Application - code file
          │  └driver
          │      ├plf
          │      │  r_plf.h                    platform driver - header file
          │      │  r_plf.c                    platform driver - code file
          │      ├input
          │         │   r_input.h                     external interrupt input driver - header file
          │         │   r_input.c                     external interrupt input driver - code file
          │         ├uart
          │         │   r_uart.h                      UART driver - header file
          │         │   r_uart.c                      UART driver - code file
          │         ├led
          │         │   r_led.h                       LED driver - header file
          │         │   r_led.c                       LED driver - code file
          │         └codeflash                        LED driver - code file
          │             r_codeflash.h                 codeflash driver - header file
          │             r_codeflash.c                 codeflash driver - code file
          ├project
          │    ├cs_cc
          │    │    └BLE_Software
          │    │         │   BLE_Software.mtpj        project file for CS+ for CC
          │    │         ├R5F11AGG_Beacon
          │    │         │       R5F11AGG_Beacon.mtsp subproject file for CS+ for CC (R5F11AGG)
          │    │         ├R5F11AGH_Beacon
          │    │         │       R5F11AGH_Beacon.mtsp subproject file for CS+ for CC (R5F11AGH)
          │    │         └R5F11AGJ_Beacon
          │    │                 R5F11AGJ_Beacon.mtsp subproject file for CS+ for CC (R5F11AGJ)
          │    └e2_cc
          │         └BLE_Software
          │              ├R5F11AGG_Beacon
          │              │   .project                 project composition file for e2 studio (R5F11AGG)
          │              │   .cproject                project configuration file for e2 studio (R5F11AGG)
          │              ├R5F11AGH_Beacon
          │              │   .project                 project composition file for e2 studio (R5F11AGH)
          │              │   .cproject                project configuration file for e2 studio (R5F11AGH)
          │              └R5F11AGJ_Beacon
          │                  .project                 project composition file for e2 studio (R5F11AGJ)
          │                  .cproject                project configuration file for e2 studio (R5F11AGJ)
          │
          └project_beacon_ascii                       project for the Beacon Application
               ├cs_cc                                 -ASCII command control
               │    └BLE_Software
               │         │   BLE_Software.mtpj        project file for CS+ for CC
               │         ├R5F11AGG_Beacon
               │         │       R5F11AGG_Beacon.mtsp subproject file for CS+ for CC (R5F11AGG)
               │         ├R5F11AGH_Beacon
               │         │       R5F11AGH_Beacon.mtsp subproject file for CS+ for CC (R5F11AGH)
               │         └R5F11AGJ_Beacon
               │                 R5F11AGJ_Beacon.mtsp subproject file for CS+ for CC (R5F11AGJ)
               └e2_cc
                    └BLE_Software
                         ├R5F11AGG_Beacon
                         │   .project                 project composition file for e2 studio (R5F11AGG)
                         │   .cproject                project configuration file for e2 studio (R5F11AGG)
                         ├R5F11AGH_Beacon
                         │   .project                 project composition file for e2 studio (R5F11AGH)
                         │   .cproject                project configuration file for e2 studio (R5F11AGH)
                         └R5F11AGJ_Beacon
                             .project                 project composition file for e2 studio (R5F11AGJ)
                             .cproject                project configuration file for e2 studio (R5F11AGJ)
```

4. Operating Procedure

This chapter describes operating procedure to evaluate the Sample Program. The operating procedure consists of five steps: Building Firmware, Writing Firmware, Confirming Operation, Evaluating RF Characteristic, and Current Consumption Measurement.


Section 4.1 "Building Firmware"
Subsection 4.1.1 "Using CS+ for CC"
Subsection 4.1.2 "Using CS+ for CC (Beacon Application-ASCII command control)"
or
Subsection 4.1.3 "Using e2 studio"
Subsection 4.1.4 "Using e^2 studio (Beacon Application-ASCII command contol)"
Section 4.2 "Writing Firmware"
Section 4.3 "Confirming Operation"
Subsection 4.3.1 "Confirming Advertising Packet Transmission"
Subsection 4.3.2 "Confirming Advertising Packet Reception"
Section 4.4 "Evaluating RF Characteristic"
Section 4.5 "Current Consumption Measurement"
Subsection 4.5.1 "Measurement Environment"
Subsection 4.5.2 "Evaluation Board Setting"
Subsection 4.5.3 "Measurement Procedure"
Subsection 4.5.3(1) "Current Consumption in Starting-up Beacon Application"
Subsection 4.5.3(2) "Current Consumption in Periodic Packet Transmission"

4.1 Building Firmware

Building Sample Program firmware can be used either CS+ for CC or e^2 studio as IDE (Integrated Development Environment).

By default settings, building the Sample Program generates the firmware R5F11AGJ_Beacon.hex, which is the same HEX file included in release package.

For evaluation, you can skip below building procedures if use HEX file included in release package.

The Beacon Application-ASCII command control is selected with the APP_SELECT macro. For details on the APP_SELECT macro, refer to subsection 6.2.1 "Application Selection configuration".

4.1.1 Using CS+ for CC

1. Start CS+ for CC and open the project named "BLE_Software.mtpj" from below folder by selecting Open File from File menu bar: [File]→[Open File].
    - Project_Source\application\project\cs_cc\BLE_Software\
2. Select Rebuild project from Build menu: [Build]→[Rebuild project], and confirm that successful compilation.
3. Confirm that the below folder contain the firmware R5F11AGJ_Beacon.hex.
    - Project_Source\application\project\cs_cc\BLE_Software\R5G11AGJ_Beacon\DefaultBuild\

4.1.2 Using CS+ for CC (Beacon Application-ASCII command control)

When building the Beacon Application-ASCII command contol, set the APP_SELECT macro to the following. For details on the APP_SELECT macro, refer to subsection 6.2.1 "Application Selection configuration".

- #define APP_SELECT (5)

Also use the project file in the following path.

- Project_Source\application\project_beacon_ascii\cs_cc\BLE_Software\

For the build procedure, refer to Section 4.1.1.

4.1.3 Using e^2 studio

1. Start Renesas e^2 studio and select below path as a workspace.

   - Project_Source\

2. Select Import from File menu bar: [File]→[Import] to open Import dialog.

3. Select Existing Project into Workspace from General: [General]→[Existing Project into Workspace] and click [Next] button.

4. Select below path as a root folder and confirm R5F11AGJ_Beacon that selected in [Projects].

   - Project_Source\

5. Click [Finish] button to close Import dialog.

6. Close [Welcome].

7. Select R5F11AGJ_Beacon in the Project Explorer.

8. Select Build Project from Project menu: [Project]→[Build Project], and confirm that successful compilation.

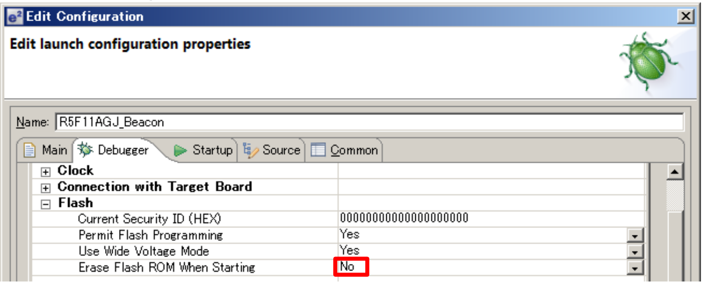
9. Confirm that the firmware R5F11AGJ_Beacon.hex is generated in the place of below path.

   - Project_Source\application\project\e2_cc\BLE_Software\R5F11AGJ_Beacon\DefaultBuild\

Note: By default, debugger setting of e^2 studio erases Flash memory before writing firmware.

When developing by using e^2 studio, change the debugger setting before starting debugging, to avoid erasing Shipping Checking Flag and Device Address which already been written in RL78/G1D Module. Disconnect the E1 Emulator from RL78/G1D Module when changing the debugger setting.

- Select [Debugger] tab in [Edit launch configuration properties] dialog, and set [No] in [Erase Flash ROM When Starting].



Edit launch configuration properties

Name: R5F11AGJ_Beacon

⊞ Clock	
⊞ Connection with Target Board	
⊟ Flash	
Current Security ID (HEX)	00000000000000000000
Permit Flash Programming	Yes
Use Wide Voltage Mode	Yes
Erase Flash ROM When Starting	No

4.1.4 Using e^2 studio (Beacon Application-ASCII command contol)

When building the Beacon Application-ASCII command contol, set the APP_SELECT macro to the following. For details on the APP_SELECT macro, refer to subsection 6.2.1 "Application Selection configuration".

- #define APP_SELECT (5)

Also use the project file in the following path.

- Project_Source\application\project_beacon_ascii\e2_cc\

For the build procedure, refer to Section 4.1.3.

4.2 Writing Firmware

To evaluate transmitting Advertising packets and receiving the packets by using by evaluation board, it is necessary to write two evaluation boards with Sample Program firmware, which described the building process in section 4.1.

For writing the Sample Program firmware, use Host machine and E1 Emulator as shown in **Figure 4-1**.

Regarding to the details of E1 Emulator, refer to E1 Emulator User's Manual (R20UT0398) and E1 Emulator Additional Document for User's Manual (Notes on Connection of RL78) (R20UT1994).

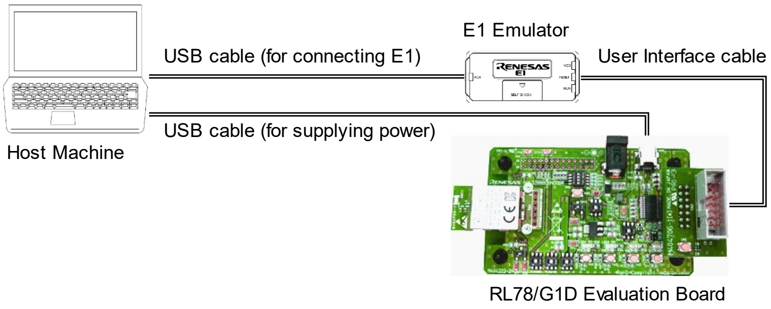

E1 Emulator
USB cable (for connecting E1)
User Interface cable
USB cable (for supplying power)
Host Machine
RL78/G1D Evaluation Board

Figure 4-1 Board Operating for Writing Firmware

Table 4-1 shows the slide switch settings for evaluating the Sample Program.

Table 4-1 Slide Switch Settings for evaluating the Sample Program

Switch	Setting	Description
SW7	2-3 connected (right) (default setting)	Power is supplied from the AC Power Supply Adapter or USB via a regulator.
SW8	2-3 connected (right)	Power is supplied from USB. If it is necessary to supply from AC Power Supply Adapter, set 1-2 connected (left).
SW9	2-3 connected (right) (default setting)	Connect to a USB device.
SW10	1-2 connected (left) (default setting)	Power is supplied to the module.
SW11	2-3 connected (right) (default setting)	Power is supplied from a source other than the E1 Debugger
SW12	2-3 connected (right) (default setting)	(Fixed)
SW13	1-2 connected (left) (default setting)	USB interface is connected.

Procedure of writing the Sample Program firmware to RL78/G1D Evaluation Board is as below:

For writing the Sample Program firmware, you can use Renesas Flash Programmer (RFP).

1. Set the slide switches on the evaluation board according to **Table 4-1** settings.
2. Connect E1 Emulator to the evaluation board as well as E1 Emulator to Host machine.
3. Connect the evaluation board to Host machine or AC-USB power supply adapter to supply power.
4. Start RFP, and create a project according to the below procedure.

   Note: Once creating project can skip above procedures for subsequence usages.

   4-1. Select [File]→[Create a new project].

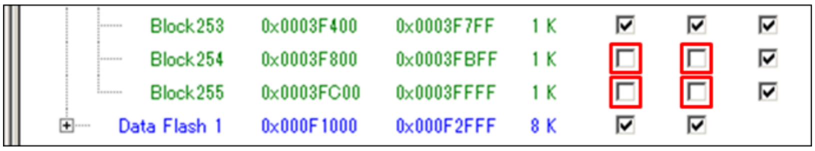
   4-2. Select [RL78] as a Microcontroller, input a project name and click [Connect] in [Create New Project] dialog.

   4-3. Confirm [Operation completed] message in Log output panel.
5. Select the firmware R5F11AGJ_Beacon.hex at [Program File].
6. Prevent erasing Block 254, 255 in Code Flash memory according to the below procedure.

   Note that Shipping Check Flag is written in Block 254 and Device Address is written in Block 255 in the case of using RL78/G1D Module.

   6-1. Select [Operation Setting] tab, and select [Erase Selected Blocks] at [Erase Option].

   6-2. Select [Block Setting] tab, and uncheck each [Erase], [P.V] of Block254, 255.



Block253	0x0003F400	0x0003F7FF	1 K	☑	☑	☑
Block254	0x0003F800	0x0003FBFF	1 K	□	□	☑
Block255	0x0003FC00	0x0003FFFF	1 K	□	□	☑
Data Flash 1	0x000F1000	0x000F2FFF	8 K	☑	☑	

7. Click [Start] button to start writing the firmware, and confirm [Operation completed] message.
8. Disconnect E1 Emulator and Power Supply from the evaluation board.

In addition, by using Unique Code Embedding Function of RFP, it is possible that same firmware can be written each time along with different System Configuration.

If write System Configuration, before clicking [Start] button by above procedure 7, set unique code file according to the below procedure.

1. Select [Unique Code] tab.
2. Check [Enable].
3. Select the below unique code file at [Unique Code File].
   - RUC_File\r5f11agj_syscfg.ruc
4. Go back to [Operation] tab.

Regarding to System Configuration, refer to subsection 5.5.1 "Accessing to Code Flash memory" in this document.

Regarding to the Unique Code Embedding Function, refer to subsection 2.3.6 "[Unique Code] Tabbed Page" in Renesas Flash Programmer V3.06 Flash memory programming software User's Manual (R20UT4540).

4.3 Confirming Operation

For confirming operation of the Sample Program, one or two the evaluation boards are used.

The switches and LED indicators are used as a user interface. For determining which application to be executed, the Sample Program checks the setting of DIP switch SW6 position-1 and position-4. Beacon Application and Scan Application also use push switch SW2 as external interrupt input trigger.

The Beacon Application-ASCII command control cannot be selected by DIP switch. Use the sample program selected and built with the APP_SELECT macro. For details on the APP_SELECT macro, refer to subsection 6.2.1 "Application Selection configuration".

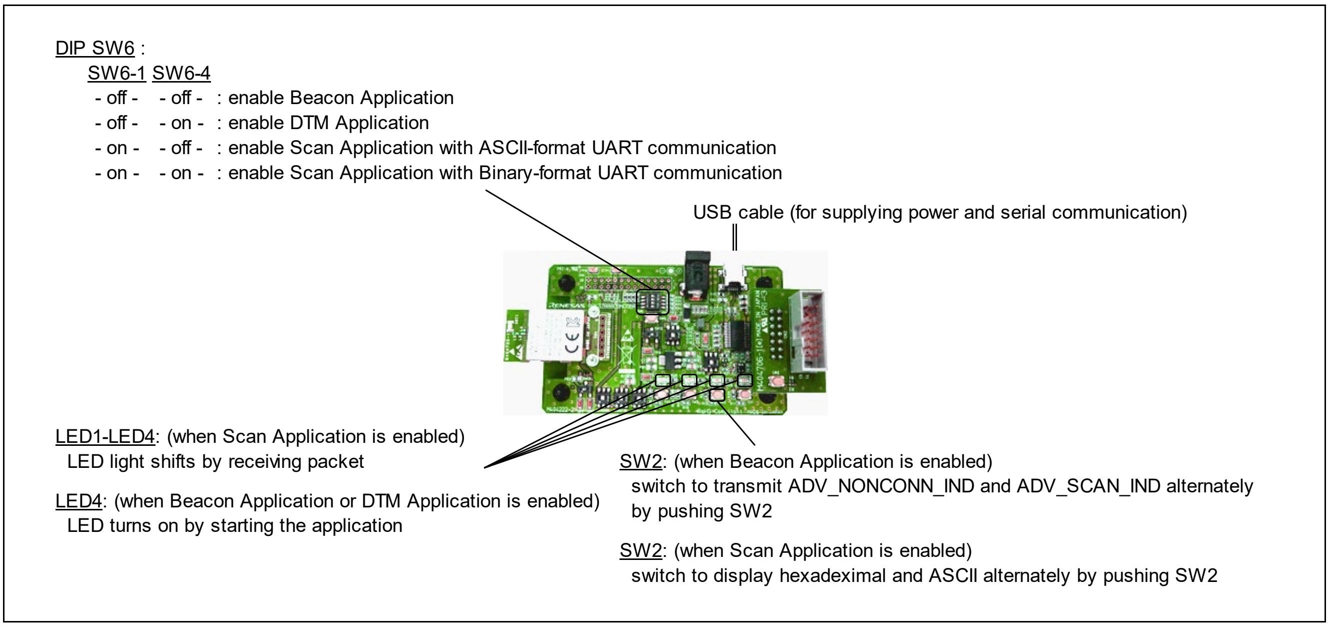
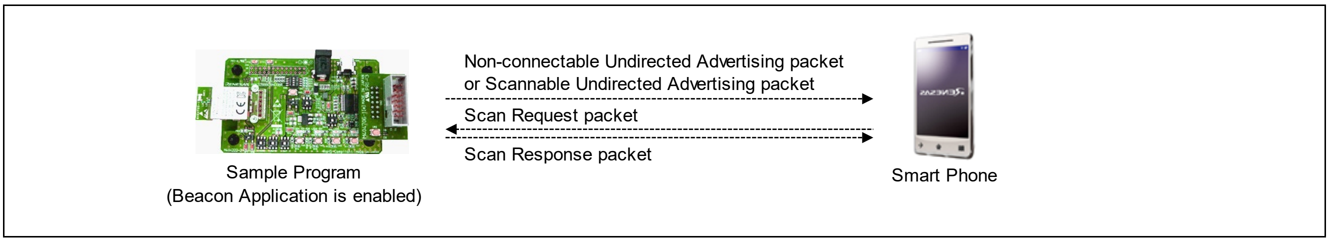

DIP SW6 :
SW6-1 SW6-4
- off - - off - : enable Beacon Application
- off - - on - : enable DTM Application
- on - - off - : enable Scan Application with ASCII-format UART communication
- on - - on - : enable Scan Application with Binary-format UART communication
USB cable (for supplying power and serial communication)
LED1-LED4: (when Scan Application is enabled)
LED light shifts by receiving packet
LED4: (when Beacon Application or DTM Application is enabled)
LED turns on by starting the application
SW2: (when Beacon Application is enabled)
switch to transmit ADV_NONCONN_IND and ADV_SCAN_IND alternately by pushing SW2
SW2: (when Scan Application is enabled)
switch to display hexadeximal and ASCII alternately by pushing SW2

Figure 4-2 Board Operating for Confiming Operation of the Sample Program

Regarding to the slide switch settings for confirming operation of the Sample Program, refer to section 4.2 **Table 4-1** in this document.

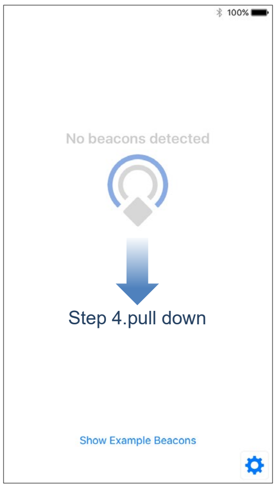
4.3.1 Confirming Advertising Packet Transmission

This subsection describes procedure for confirming that Beacon Application transmits Advertising packets by using Smart Phone.

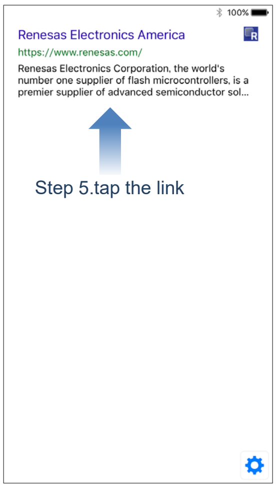

Non-connectable Undirected Advertising packet
or Scannable Undirected Advertising packet
Scan Request packet
Scan Response packet
Sample Program
(Beacon Application is enabled)
Smart Phone

To run Beacon Application, switch both DIP switch SW6 position-1 and position-4 to OFF, and supply power via either DC jack (J1) or USB interface (CN3). After power up, Beacon Application transmits Eddystone-URL packet by default. Using smart phone application, you can receive the advertising packet. The smart phone application procedure is very much similar for both iOS device and Android device.

1. To receive Eddystone-URL packet, install below application to smart phone.
   - for Android device, Physical Web - Google Play
     https://play.google.com/store/apps/details?id=com.physicalweb
   - for iOS device, Physical Web - App Store
     https://itunes.apple.com/app/physical-web/id927653608?mt=8
2. Switch DIP switch SW6 position-1 and position-4 to OFF on the evaluation board.
3. Start supplying power to the evaluation board, and then Beacon Application starts running.
4. Run the smart phone application and search the Eddystone beacons by pulling down the display.
5. When receive the Eddystone-URL packet from the Sample Program, below URL is displayed to link the web page.
   - Renesas Electronics
     https://www.renesas.com/


No beacons detected

Step 4.pull down


Renesas Electronics America

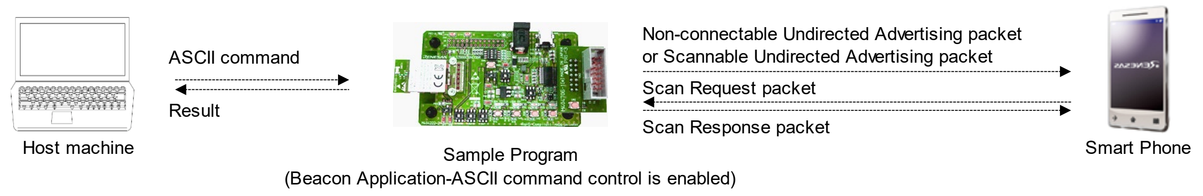
https://www.renesas.com/

Renesas Electronics Corporation, the world's number one supplier of flash microcontrollers, is a premier supplier of advanced semiconductor sol...

Step 5.tap the link

4.3.2 Confirming Advertising Packet Transmission (Beacon Application-ASCII command control)

This subsection describes procedure for confirming that Beacon Application-ASCII command control transmits Advertising packets by using Smart Phone.


Host machine
ASCII command
Result
Sample Program
(Beacon Application-ASCII command control is enabled)
Non-connectable Undirected Advertising packet
or Scannable Undirected Advertising packet
Scan Request packet
Scan Response packet
Smart Phone

The Beacon Application-ASCII command control is executed when power is supplied from the DC jack (J1) or USB interface (CN3) of the evaluation board that contains the sample program firmware. The evaluation board waits for a control command from the host machine.

The Beacon Application-ASCII command control can send Eddystone-URL by default setting. You can receive this packet by using a smartphone. Using smart phone application, you can receive the advertising packet. The smart phone application procedure is very much similar for both iOS device and Android device.

1. To receive Eddystone-URL packet, install below application to smart phone.
   - for Android device, Physical Web - Google Play
     https://play.google.com/store/apps/details?id=com.physicalweb
   - for iOS device, Physical Web - App Store
     https://itunes.apple.com/app/physical-web/id927653608?mt=8
2. Connect evaluation board that used as scanner device to Host machine with USB cable. The evaluation board waits for a control command from the host machine.
3. Start terminal software (e.g. Tera Term) on Host machine. Then set serial communication setting according to **Table 4-2** settings

Table 4-2 UART Settings for Terminal software

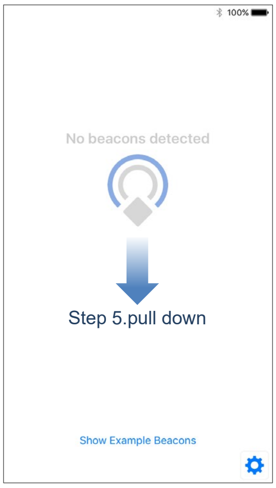
Setting Item		Setting Value
Serial Port	Port	USB Serial Port Note that COM number is different from each evaluation board.
	Baud rate	4,800bps
	Data bit length	8bit
	Parity	None
	Stop bit length	1bit
	Flow control	None
New Line	Receive	LF
	Transmit	LF
Terminal Size	Horizontal	over than 128 characters

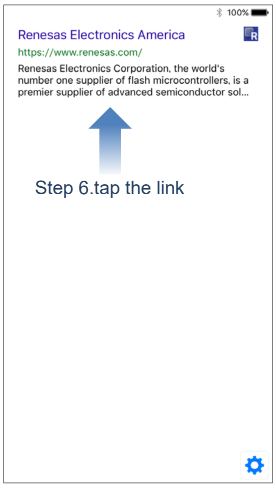
4. Enter "adstart n" in the terminal software and press the Enter key to send a Non-connectable Undirected Advertising packet. For the ASCII command specifications, refer to section 5.2 "Beacon Application-ASCII command control".

5. Launch the smart phone application and use the pull-down on the screen to search for Eddystone beacons.

6. When receive the Eddystone-URL packet from the Sample Program, below URL is displayed to link the web page.

   − Renesas Electronics
     https://www.renesas.com/



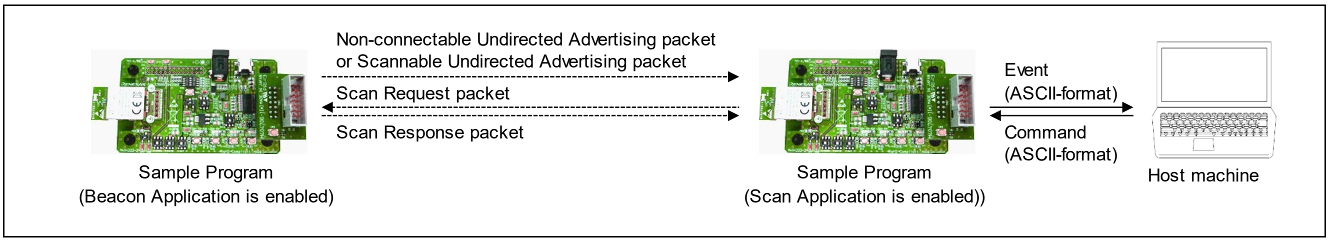


Renesas Electronics America
https://www.renesas.com/
Renesas Electronics Corporation, the world's number one supplier of flash microcontrollers, is a premier supplier of advanced semiconductor sol...

4.3.3 Confirming Advertising Packet Reception

This subsection describes procedure for comfirming that Scan Application receives Advertising packets.

After power up, Beacon Application transmits Advertising packets periodically, and Scan Application executes Scanning indefinitely. When receive Advertising packets, Scan Application notifies events to Host machine.


Non-connectable Undirected Advertising packet
or Scannable Undirected Advertising packet
Scan Request packet
Scan Response packet
Sample Program
(Beacon Application is enabled)
Sample Program
(Scan Application is enabled))
Event
(ASCII-format)
Command
(ASCII-format)
Host machine

To run Beacon Application, switch both DIP switch SW6 position-1 and position-4 to OFF, and supply power via DC jack (J1) or USB interface (CN3). Similarly, to run Scan Application, switch DIP switch SW6 position-1 to ON and position-4 to OFF, and supply power via USB interface from Host machine.

1. Switch DIP switch SW6 position-1 to ON and position-4 to OFF on evaluation board used as scanner device.
2. Connect evaluation board that used as scanner device to Host machine with USB cable. By supplying power, Scan Application starts running.
3. Start terminal software (e.g. Tera Term) on Host machine. Then set serial communication setting according to **Table 4-3** settings.
4. Push RESET switch SW5 of evaluation board that used as scanner device, and confirm below message is displayed in terminal software. This message is displayed when Scan Application starts Scanning.

```
Start Scan :OK
```

5. Switch DIP switch SW6 position-1 to and position-4 to OFF on evaluation board that used as beacon device.
6. Supply power to evaluation board that used as beacon device. Beacon Application starts running and transmitting Advertising packets.
7. Confirm that Scan Application receives Advertising packet and below message is displayed on terminal software. The information displays the received Advertising packet, which consists of PDU type of Advertising packet, device address type, device address, RSSI, payload data size and payload data, is displayed.

```
ADV_NONCONN_IND PUBLIC 12:34:56:78:9A:B1 37ch -46dBm 27byte
                                    0201060303AAFE1316AAFE10EE02676F6F2E676C2F35774B6B524B
```

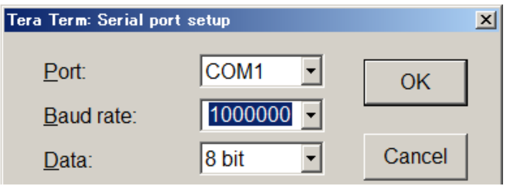
If there are another beacon devices, which transmit Advertising packets, the information of the received Advertising packets from those devices are displayed as well.

By using commands, it is possible to start or stop Scanning as well as to change the configuration. Regarding to the specification of commands, refer to subsection 5.3.1 "ASCII-format UART communication" in this document.

Table 4-3 UART Settings for Terminal software

Setting Item		Setting Value
Serial Port	Port	USB Serial Port Note that COM number is different from each evaluation board.
	Baud rate	1,000,000bps
	Data bit length	8bit
	Parity	None
	Stop bit length	1bit
	Flow control	None
New Line	Receive	LF
	Transmit	LF
Terminal Size	Horizontal	over than 128 characters

When Tera Term is used as terminal software, there is no "1,000,000bps" in the drop-down list of Baud Rate. Thus, it is necessary to enter "1000000" to the input box of Baud Rate directly.



Tera Term: Serial port setup

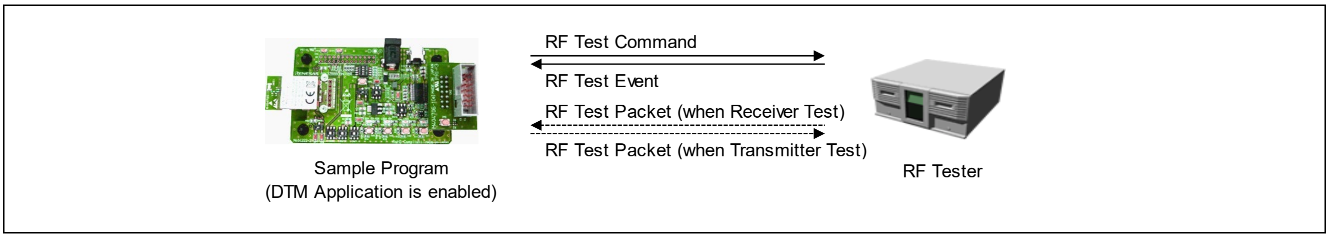
Port: COM1

Baud rate: 1000000

Data: 8 bit

4.4 Evaluating RF Characteristic

This subsection describes procedure for enabling DTM Application and confirming RF characteristic by executing Direct Test Mode.


RF Test Command
RF Test Event
RF Test Packet (when Receiver Test)
RF Test Packet (when Transmitter Test)
Sample Program
(DTM Application is enabled)
RF Tester

To run DTM Application, switch DIP switch SW6 position-1 to OFF and position-4 to ON, and supply power through DC jack (J1) or USB interface (CN3). DTM Application executes Direct Test Mode by receiving RF Test commands through UART. By connecting RL78/G1D Evaluation Board to RF Tester, it is possible to evaluate RF characteristic of RL78/G1D.

1. Set DIP switch SW6 position-1 to OFF and position-4 to ON on the evaluation board.
2. Connect TxD0 pin, RxD0 pin and GND pin of the evaluation board to RF Tester. If logic level of the signal is different between RL78/G1D and RF Tester, connect through logic level converter.
3. Start supplying power to the evaluation board, and DTM Application starts running.
4. Refer to respective manuals of RF Tester to set UART settings according to **Table 4-4** settings.
5. Refer to respective manuals of RF Tester to start Direct Test Mode.

Table 4-4 UART Settings for RF Tester

Setting	Value
Baud rate	9600bps
Data bit length	8bit
Parity	None
Stop bit length	1bit
Flow control	None

4.5 Current Consumption Measurement

This section describes current consumption measurement for using RL78/G1D Evaluation Board. Regarding to the details of RL78/G1D Evaluation Board (RTK0EN0001D01001BZ), refer to RL78/G1D Evaluation Board User's Manual (R30UZ0048).

4.5.1 Measurement Environment

Table 4-5 shows the necessary equipments for current consumption measurement. Regarding to the details on how to use each equipment, refer to respective manuals of each equipment.

Table 4-5 Necessary Equipments for Current Consumption Measurement

Equipment	Role	Example Equipment
Power Source	Supplying power to RL78/G1D	Stabilized power supply or Battery Note that supply voltage shall be in the range of the RL78/G1D operation voltage
Measurement Equipment	Indicating and logging the result of measurement	Oscilloscope
Voltage Detector	Detecting the operation voltage of RL78/G1D	Voltage Probe
Current Detector	Detecting the current consumption of RL78/G1D	Current Probe with clamp, or combination of Shunt Resistor and Voltage Probe Note that recommended resistor is 10 ohm.

Figure 4-3 shows the measurement environment which uses current probe as current detector. In this environment, the current consumption of RL78/G1D is the result of measuring between terminal TP7 and TP8 of the evaluation board by current probe.

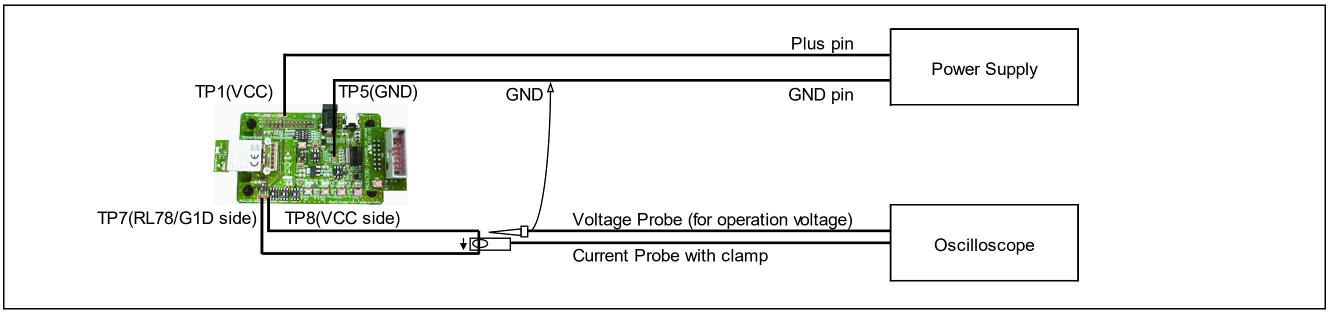


Figure 4-3 Measurement Environment which uses Current Probe

Figure 4-4 shows the measurement environment which uses the combination of shunt resistor and voltage probe as a current detector. In this environment, the resistor is inserted between terminal TP7 and TP8 of the evaluation board, and voltage drop at the resistor is measured by using two voltage probes.

Voltage drop *dV* by the resistor is difference of two voltages measured by individual voltage probe. The current consumption of RL78/G1D is the result of calculating with formula *I*=*dV* / *R*, where *I* is current; *dV* is voltage drop by the resistor; and *R* is resistance value.

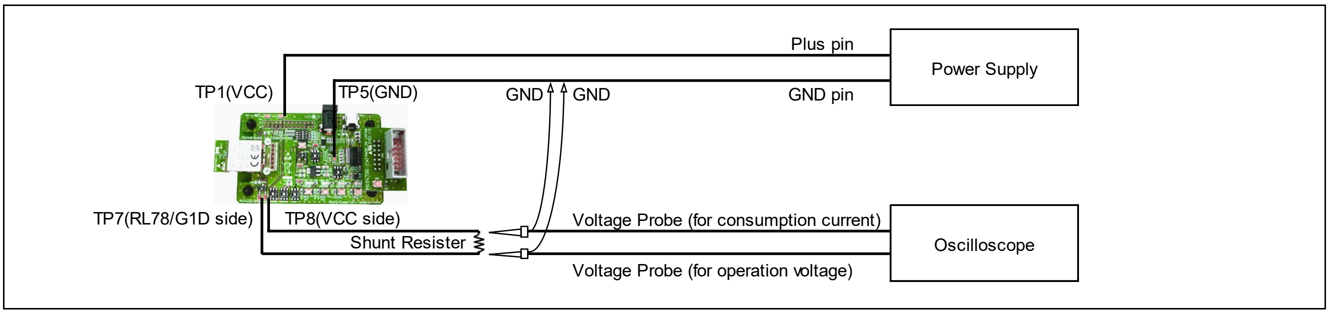


Figure 4-4 Measurement Environment which uses the combination of Shunt Resistor and Voltage Probe

4.5.2 Evaluation Board Setting

Table 4-6 shows the slide switch settings of the evaluation board for current consumption measurement.

Table 4-6 Slide Switch Settings for current consumption measurement

Switch	Setting	Description
SW7	1-2 connected (left)	Power is directly supplied from external power source (not via a regulator). If it is necessary to supply from USB, set 2-3 connected (right).
SW8	1-2 connected (left)	Power is supplied from TP1, TP5 pin or AC Power Supply Adapter. If it is necessary to supply from USB, set 2-3 connected (right).
SW9	1-2 connected (left)	Connect to an external extension interface.
SW10	2-3 connected (right)	The power supply line is left open.
SW11	2-3 connected (right) (default setting)	Power is supplied from a source other than the E1 debugger.
SW12	2-3 connected (right) (default setting)	(Fixed).
SW13	2-3 connected (right)	USB interface is disconnected.

4.5.3 Measurement Procedure

Current consumption measurement procedures are described in below steps. Note that the procedure is reference for only measuring current consumption of Beacon Application with default setting.

Regarding to the details of how to set each equipment settings, refer to the respective manuals.

(1) **Current Consumption in Starting-up Beacon Application**

1. Connect the evaluation board to Power Supply at power off stage.
2. Set below settings to Oscilloscope by referring to **Figure 4-5**.
   - Capture Trigger : about 1.6V in Supply Voltage
   - Current Measurement Range : about 10mA
   - Measurement Period : about 40msec from capture trigger
3. Start supplying power to the evaluation board within the RL78/G1D operation voltage range.
4. Start measuring by Oscilloscope by detecting the rising voltage.

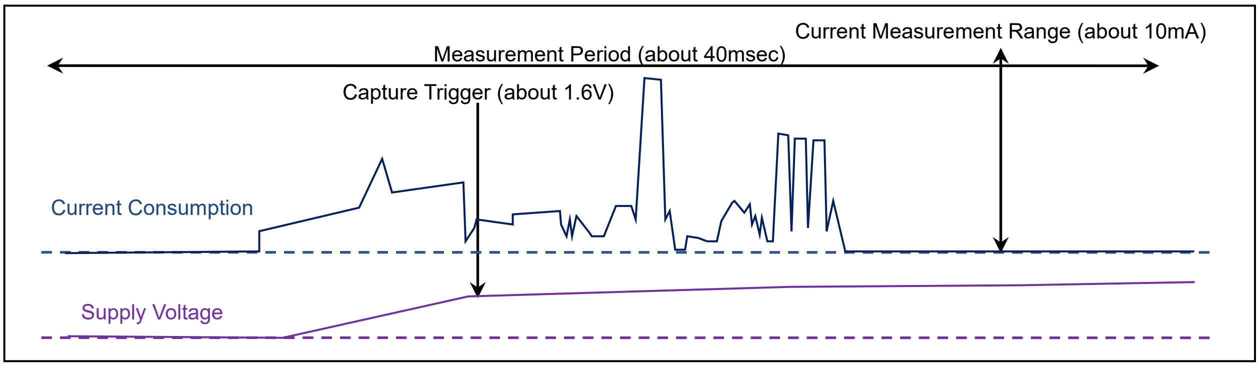

Current Measurement Range (about 10mA)
Measurement Period (about 40msec)
Capture Trigger (about 1.6V)
Current Consumption
Supply Voltage

Figure 4-5 Measuring Current Consumption in Starting-up Beacon Application

(2) **Current Consumption in Periodic Packet Transmission**

1. Start supplying power and start Beacon Application.
2. Set below settings to Oscilloscope by referring to **Figure 4-6**.
   - Capture Trigger : about 0.5mA in current consumption
   - Current Measurement Range : about 10mA
   - Measurement Period : about 10msec from capture trigger
3. Start measuring by Oscilloscope by detecting the current of periodic transmitting.

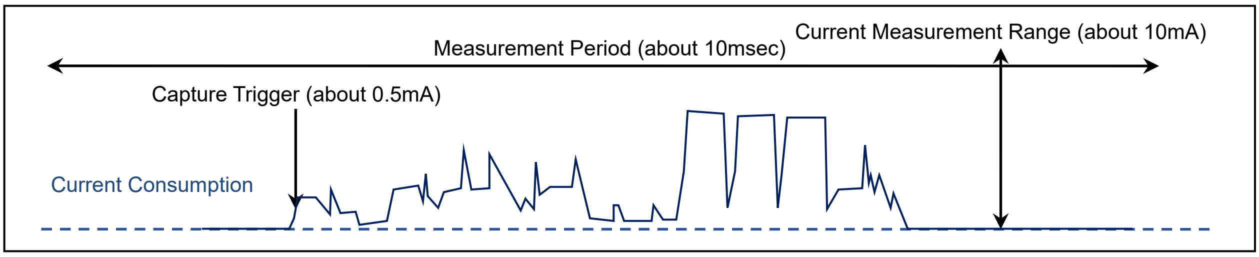

Current Measurement Range (about 10mA)
Measurement Period (about 10msec)
Capture Trigger (about 0.5mA)
Current Consumption

Figure 4-6 Measuring Current Consumption in Periodic Packet Transmission

5. Specification

5.1 Beacon Application

Beacon Application loads Advertising Information and Advertising Data from the system configuration, which stored in Code Flash memory. Then starts transmitting Non-connectable Undirected Advertising packet for broadcasting information. Peer Device, like a Smart Phone, receives Advertising packet and provides each service related to the Advertising Data.

When switch SW2 of the evaluation board is pushed, Beacon Application stops transmitting Non-connectable Undirected Advertising packet. When switch SW2 is pushed again, Beacon Application starts transmitting Scannable Undirected Advertising packet.

5.1.1 Default Advertising Configuration

Table 5-1 shows the default Advertising configuration of Beacon Application. The Default Advertising Data is different between transmitting Non-connectable Undirected Advertising packet and transmitting Scannable Undirected Advertising packet.

Table 5-1 the Default Advertising Configuration of Beacon Application

Advertiser Address	Public Device Address 12:34:56:78:9A:B1
Advertising Type	Non-connectable Undirected Advertising (ADV_NONCONN_IND) or Scannable Undirected Advertising(ADV_SCAN_IND)
Advertising Interval	100msec
Advertising Interval Delay	add random delay to Advertising interval
Advertising Channel Map	All channels (37ch,38ch,39ch)
Advertising Count Limitation	transmitting indefinitely
Advertising Transmit Power	0dBm at ANT pin of RL78/G1D
Advertising Data Count	the number of Advertising data is 1, if Advertising Type is ADV_NONCONN_IND the number of Advertising data is 2, if Advertising Type is ADV_SCAN_IND
Advertising Data [0] to [9]	Regarding to the default Advertising Data for ADV_NONCONN_IND packet, refer to **Table 5-2**. Regarding to the default Advertising Data for ADV_SCAN_IND packet and SCAN_RSP packet, refer to **Table 5-3**.
Advertising Event Permission	notify All Advertising Event
Use White List	not use White List

Table 5-2 shows the default data for Non-connectable Undirected Advertising packet.

Table 5-2 the Default Data for Non-connectable Undirected Advertising packet

Advertising Data [0] to [9]	Advertising data[0] (ADV_NONCONN_IND payload data)	
	Length	2byte
	AD Type	<<Flags>> (0x01)
	AD Data	LE General Discoverable Mode (bit1) BR/EDR Not Supported (bit2)
	Length	3byte
	AD Type	<<Complete List of 16-bit Service Class UUIDs>> (0x03)
	AD Data	Eddystone (0xFEAA)
	Length	19byte
	AD Type	<<Service Data>> (0x16)
	AD Data	Eddystone-URL : https://goo.gl/5wKkRK
	Advertising data[1] to [9] are empty	

Table 5-3 shows the default data for Scannable Undirected Advertising packet.

Table 5-3 the Default Data for Scannable Undirected Advertising packet and Scan Response packet

Advertising Data [0] to [9]	Advertising data[0] (ADV_SCAN_IND payload data)	
	Length	2byte
	AD Type	<<Flags>> (0x01)
	AD Data	LE General Discoverable Mode (bit1) BR/EDR Not Supported (bit2)
	Length	3byte
	AD Type	<<Complete List of 16-bit Service Class UUIDs>> (0x03)
	AD Data	Eddystone (0xFEAA)
	Length	19byte
	AD Type	<<Service Data>> (0x16)
	AD Data	Eddystone-URL : http://goo.gl/JQh3fQ
	Advertising data[1] (SCAN_RSP payload data)	
	Length	24byte
	AD Type	<<Complete Local Name>> (0x09)
	AD Data	"Renesas RL78/G1D Beacon"
	Advertising data[2] to [9] are empty	

Regarding to the specification of Eddystone and Eddystone-URL, refer to below website.

- Specification for Eddystone, an open beacon format from Google
  https://github.com/google/eddystone
- Specification for Eddystone, an open beacon format from Google - Eddystone-URL
  https://github.com/google/eddystone/tree/master/eddystone-url

5.2 Beacon Application-ASCII command control

The Beacon Application-ASCII command control can send Advertising packet or set Advertising configuration by sending ASCII command from the host machine.

5.2.1 Default Advertising Configuration

See below for the default settings of the Beacon Application-ASCII command control.

- The default advertising configuration: **Table 5-1**
- The default data for Non-connectable Undirected Advertising packet: **Table 5-2**
- The default data for Scannable Undirected Advertising packet and Scan Response packet: **Table 5-3**
- The default data for White List: **Table 5-4**

Table 5-4 the Default Data for White List

White List [0] to [0xF]	White List[0]	
	Device Address Type	Public
	reserve	-
	Device Address	12:34:56:78:9A:B1
	White List[1] to [0xF] are empty.	

5.2.2 ASCII-format UART communication commands List

Table 5-5 ASCII-format UART commands

Command	Description
pwup	Start power supply to the RF unit. / Display status.
pwdn	Stop power supply to the RF unit.
adstart	Start advertising. / Display status.
adstop	Stop advertising.
aditvl	Set advertising interval. / Display setting.
addly	Set advertising interval delay. / Display setting.
adch	Set advertising channel. / Display setting.
adloop	Set advertising count limitation. / Display setting.
adtxpw	Set advertising transmit power. / Display setting.
adadr	Set own device address type and own device address. / Display setting.
adevt	Set advertisng event. / Display setting.
addtype	Set advertising data type. / Display setting.
addata	Set advertising data. / Display setting.
wlist	Set white list. / Display setting.
save	Save advertising configuration, advertising data and white list to system configuration area in code flash.
erase	Erase system configuration area in code flash.
echo	Set echo back of command input.

5.2.3 ASCII-format UART communication commands

Shows the ASCII format command specifications of the Beacon Application-ASCII command control. The beacon application can be operated by connecting the host machine to the evaluation board with USB and using the terminal software installed on the host machine to input commands.

For the operation confirmation procedure, refer to Section 4.3.2, "Confirming Advertising Packet Transmission (Beacon Application-ASCII command control)".

(1) Start power supply to the RF unit command

Syntax	**pwup**
Operation	Power supply to the RF unit is started and the beacon stack is initialized. Note: Immediately after starting the Beacon Application-ASCII command control, the power supply to the RF section is started.
Condition	-
Error	OK : Success ERR_PWRUP : Power supply to RF unit
Example	`pwup` `OK`

Syntax	**pwup?**
Operation	Display the status of power supply to the RF unit.
Display Format	pwup? <power state> OK <power state>: POWER UP : Power supply to RF unit POWER DOWN : Power supply to RF unit is stopped
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`pwup?`
	`POWER UP`
	`OK`

(2) Stop power supply to the RF unit command

Syntax	**pwdn**
Operation	Power supply to the RF unit is stopped.
Condition	-
Error	OK : Success ERR_PARAM : Illigal parameter
Example	`pwdn` `OK`

(3) Start advertising command

Syntax	**`adstart <adv_type>`**
Operation	Start advertising specified by <adv_type>. <adv_type>: n : Non-connectable Undirected Advertising s : Scannable Undirected Advertising
Condition	Power supply to RF unit. Advertising event "RBLE_EVT_PERMIT_ADV_STOP" is enabled.
Error	OK : Success ERR_PARAM : Illigal parameter ERR_PWRDOWN : Power supply to RF unit is stopped

Example	`adstart n // Start Non-connectable Undirected Advertising`
	`OK`

Syntax	**`adstart?`**
Operation	Display the operating status of advertising.
Display Format	adstart? <advertising state> OK <advertising state>: NON-CONNECTABLE : Non-connectable Undirected Advertising SCANNABLE : Scannable Undirected Advertising STOP : Advertising is stopped
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adstart?`
	`NON-CONNECTABLE`
	`OK`

(4) Stop advertising command

Syntax	**`adstop`**
Operation	Stop advertising.
Condition	Power supply to RF unit.
Error	OK : Success ERR_PARAM : Illigal parameter ERR_PWRDOWN : Power supply to RF unit is stopped ERR_STOP : Advertising is stopped
Example	`adstop` `OK`

(5) Set advertising interval command

Syntax	**`aditvl <interval>`**
Operation	Set Advertising interval to the time specified by <interval>. When advertising packet is being sent, the setting will be applied immediately. <interval>: 2 to 30720 : 2msec to 30.72s Note: Specify with units of msec. The decimal point cannot be specified.
Condition	Power supply to RF unit. Advertising event "RBLE_EVT_PERMIT_ADV_STOP" is enabled.
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`aditvl 1000 // Set advertising interval to 1 sec`
	`OK`

Syntax	**`aditvl?`**
Operation	Display the advertising interval.
Display Format	aditvl? <interval> OK <interval>: 2 to 30720 : 2msec to 30.72s Note: Display with units of msec.
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`aditvl?`
	`1000`
	`OK`

(6) Set advertising interval delay command

Syntax	**`addly <delay>`**
Operation	Set advertising interval delay to <delay>. When advertising packet is being sent, the setting will be applied immediately. <delay>: 0 : Disable advertising interval delay 1 : Enable advertising interval delay
Condition	-
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`addly 1 // Enable advertising interval delay`
	`OK`

Syntax	**`addly?`**
Operation	Display the advertising interval delay setting.
Display Format	addly? <delay> OK <delay>: 0 : Disable advertising interval delay 1 : Enable advertising interval delay
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`addly?`
	`1`
	`OK`

(7) Set advertising channel command

Syntax	**adch <ch_map>**
Operation	Set advertising channel to <ch_map>. When advertising packet is being sent, the setting will be applied immediately. <ch_map>: 37 : 37ch 38 : 38ch 39 : 39ch all : All channel Note: When specifying multiple channels, separate them with ','.
Condition	-
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adch 37,39          // Set advertising channel to 37ch and 39ch`
	`OK`

Syntax	**adch?**
Operation	Display the advertising channel setting.
Display Format	adch? <ch_map> OK <ch_map>: 37 : 37ch 38 : 38ch 39 : 39ch all : All channel
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adch?`
	`37,39`
	`OK`

(8) Set advertising count limitation command

Syntax	**`adloop <loop_cnt>`**
Operation	Set advertising count limitation to <loop_cnt>. When advertising packet is being sent, the setting will be applied immediately. <loop_cnt>: 0 to 255 Note: Specify with decimal. Note: When set with 0, Advertising is executed indefinitely.
Condition	-
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adloop 10 // Set advertising count limitation to 10`
	`OK`

Syntax	**`adloop?`**
Operation	Display the advertising count limitation setting.
Display Format	adloop? <loop_cnt> OK <loop_cnt>: 0 to 255 Note: Display with decimal.
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adloop?`
	`10`
	`OK`

(9) Set advertising transmit power command

Syntax	**adtxpw <tx_pwr>**
Operation	Set advertising transmit power to <tx_pwr>. When advertising packet is being sent, the setting will be applied immediately. <tx_pwr>: 1 : -15dBm 2 : -10dBm 3 : -7dBm 4 : -2dBm 5 : (reserved) 6 : (reserved) 7 : -1dBm 8 : (reserved) 9 : 0dBm Note: 5, 6 and 8 are reserved. Specify with one numeric character.
Condition	-
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adtxpw 2 // Set advertising transmit power to 2`
	`OK`

Syntax	**adtxpw?**
Operation	Display the advertising transmit power setting.
Display Format	adtxpw? <tx_pwr> OK <tx_pwr>: 1 to 9
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adtxpw?`
	`2`
	`OK`

(10) Set own device address type and own device address command

Syntax	**adadr <own_addr_type><own_addr>**
Operation	Set own device address type to <own_addr_type>. Set own device address to <own_addr>. When advertising packet is being sent, the setting will be applied immediately. <own_addr_type>: pub : Public device address rnd : Random device address <own_addr>: Specify with 12 hexadecimal characters.
Condition	-
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adadr pub74905000C991     // Set device addres to public and 74:90:50:00:C9:91`
	`OK`

Syntax	**adadr?**
Operation	Display own device address type and own device address.
Display Format	adadr? <own_addr_type><own_addr> OK <own_addr_type>: pub : Public device address rnd : Random device address <own_addr>: Display with 12 hexadecimal characters.
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adadr?`
	`pub74905000c991`
	`OK`

(11) Set advertising event command

Syntax	**`adevt <evt_permit>`**
Operation	Set advertising event to <evt_permit>. When advertising packet is being sent, the setting will be applied immediately. <evt_permit>: none : Not permit to notify event tx : Permit to notify RBLE_EVT_ADV_TX_IND event stop : Permit to notify RBLE_EVT_ADV_STOP_CMP event rx : Permit to notify RBLE_EVT_SCANREQ_RX_IND event all : Permit to notify All Advertising event Note: When specifying multiple event, separate them with ','.
Condition	-
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adevt tx,stop // Set permission for RBLE_EVT_ADV_TX_IND event notification`
	`OK            // and RBLE_EVT_ADV_STOP_CMP event notification`

Syntax	**`adevt?`**
Operation	Display advertising event setting.
Display Format	adevt? <evt_permit> OK <evt_permit>: none : Not permit to notify event tx : Permit to notify RBLE_EVT_ADV_TX_IND event stop : Permit to notify RBLE_EVT_ADV_STOP_CMP event rx : Permit to notify RBLE_EVT_SCANREQ_RX_IND event all : Permit to notify All Advertising event
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`adevt?`
	`tx,stop`
	`OK`

(12) Set advertising data type command

Syntax	**addtype <adv_type>**
Operation	Set advertising data array type to <adv_type>. Specify the array type of Advertising data set by the Set advertising data command (addata) when Advertising is stopped. <adv_type>: n : Non-connectable Undirected Advertising data array s : Scannable Undirected Advertising data array
Condition	Advertising is stopped.
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`addtype n    // Set data array type to Non-connectable Undirected Advertising`
	`OK`

Syntax	**addtype?**
Operation	Display advertising data array type.
Display Format	addtype? <adv_type> OK <adv_type>: NON-CONNECTABLE SCANNABLE
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`addtype?`
	`NON-CONNECTABLE`
	`OK`

(13) Set advertising data command

Syntax	**`addata <data_idx>,<data>`**
Operation	Set <data> to element <data_idx> of advertising data array. Before setting Advertising data, specify the type of Advertising data array (Non-Connectable Undirected Advertising or Scannable Undirected Advertising) with the Set advertising data type command (addtype). However, subject to the following conditions. When advertising is stopped, set the data to the advertising data array specified by the Set advertising data type command (addtype). When advertising is started, the advertising data array type specified by the Set advertising data type command (addtype) is ignored and the advertising data array in operation is immediately updated. Note: In the initial settings of the Beacon Application-ASCII command control, the element <data_idx> of Advertising data that can be set is as follows. Non-connectable Undirected Advertising : <data_idx>=0 Scannable Undirected Advertising : <data_idx>=0,1 To increase the number of configurable advertising data elements, refer to subsection 6.2.5 "Advertising Data" and add Advertising data to adv_nonconn_data structure and adv_scan_data structure. <data_idx>: 0 to 9 : Element of advertising data array Note: Specify with one numeric character. <data>: Advertising data length + Advertising data string Note: Advertising data length represents the data length of Advertising data character string, and specify the value of "(Advertising data character string length) / 2". Note: Specify with hexadecimal string.
Condition	-
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`addata 0,0C020106080952656E65736173`
	`OK`

Syntax	**`addata?`**
Operation	Display advertising data.
Display Format	addata? <adv_type> : Advertising data array type <data_idx>,<data> : Advertising data OK <adv_type>: NON-CONNECTABLE SCANNABLE <data_idx>: 0 to 9 : Element number of advertising data Note: Display with one numeric character. <data>: Advertising data length + Advertising data string Note: Display as hexadecimal string.
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`addata?`
	`SCANNABLE`
	`0,0C020106080952656E65736173`
	`1,19180952656e6573617320524c37382f47314420426561636f6e`
	`OK`

(14) Set white list command

Syntax	**`1. wlist <wl_idx>,<addr_type><addr>`**
	`2. wlist flush`
	`3. wlist <use_wl>`

Operation	1. Set address <addr> of address type <addr_type> to element <wl_idx> of White List array. <wl_idx>: 0 to F : Element number of white list Note: Specify as 1 hexadecimal characters. <addr_type>: pub : Public device address rnd : Random device address <own_addr>: Specify with 12 hexadecimal characters. Note: In the initial settings of the Beacon Application-ASCII command control, the element <data_idx> of White List that can be set is as follows. <data_idx>=0 To increase the number of configurable whtie list elements, refer to subsection 6.2.7 "White List configuration" and add device address to wl_info structure. 2. Clear the white wist and set disnable white list. Note: By executing the save command, the white list saved in the system configuration area of the code flash can be cleared. After the system is restarted, the initial values of RBLE_DEV_INFO structure (White List) set by the program are used. 3. Set white list operation with <use_wl>. When advertising packet is being sent, the setting will be applied immediately. <use_wl>: 0 : Disable white list 1 : Enable white list Note: White List is valid for Scannable Undirected Advertising.
Condition	-
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`1. wlist 0,pub74905000C991`
	`OK`
	
	`2. wlist flush`
	`OK`
	
	`3. wlist e`
	`OK`

Syntax	**wlist?**
Operation	Display white list.
Display Format	wlist? <use_wl> <wl_idx>,<addr_type><addr> OK <use_wl>: 0 : Disable white list 1 : Enable white list <wl_idx>: 0 to f : Element number of white list Note: Display with 1 hexadecimal characters. <addr_type>: pub : Public device address rnd : Random device address <addr>: Display with 12 hexadecimal characters.
Error	OK : Success ERR_PARAM : Illigal parameter

Example	`wlist?`
	`0`
	`0,pub74905000c991`
	`OK`

(15) Save advertising data command

Syntax	**save**
Operation	Save the current advertising operation settings in the system configuration area of the code flash.
Condition	Advertising is stopped.
Error	OK : Success ERR_START : Advertising is started ERR_SAVE : Save failed
Example	`save` `OK`

(16) Erase advertising data command

Syntax	**erase**
Operation	Erase the system configuration area of the code flash. After the system is restarted, the initial values of RBLE_ADV_INFO structure (Advertising information), RBLE_ADV_DATA structure (Advertising data), and RBLE_DEV_INFO structure (White List) set by the program are used.
Condition	Advertising is stopped.
Error	OK : Success ERR_START : Advertising is started
Example	`erase` `OK`

(17) Set echo back command

Syntax	**echo**
Operation	Set echo back mode.
Display Format	echo <echo_back> OK <echo_back>: 0 : Disable echo back 1 : Enable echo back
Condition	-
Error	OK : Success ERR_START : Advertising is started

Example	`echo`
	`0`
	`OK`

5.3 Scan Application

Scan Application executes Scanning for receiving Advertising packets. When receive Advertising packet from beacon device, Scan Application sends the information such as received channel, RSSI, and payload data through UART. By receiving the commands through UART, Scan Application can start and stop Scanning as well as changes its configuration.

Scan Application can execute Duplicate Filter and RSSI Filter. When Duplicate Filter is enabled, Scan Application does not notify to host about the packet if it receives the same Advertising packet again which has same Advertiser's Address and Advertising Data as the received packet before. When RSSI Filter is enabled, Scan Application does not notify to host about the packet if RSSI of received Advertising packet is lower than threshold.

For notifying result of Scanning and controlling Scanning, there are two formats, which are implemented with ASCII-format UART communication and Binary-format UART communication in Scan Application.

5.3.1 ASCII-format UART communication commands

This subsection describes specification of ASCII-format UART communication, which is implemented in Scan Application. This UART communication is used for connecting between the evaluation board and Host machine through virtual COM port using USB cable, and for operating by entering commands manually on terminal software.

Regarding to the procedure for evaluating operation, refer to subsection 4.3.3 "Confirming Advertising Packet Reception" in this document.

(1) **Start Scan command**

Syntax	`(Key [Enter] only)`
Operation	This command starts Scanning. When starting Scanning succeeds, the message "`Start Scan :OK`" is displayed. When starting Scanning fails, "`Start Scan :ER`" is displayed.
Condition	Scanning is stopped
Example	`Key [Enter]` // start Scanning `Start Scan :OK`

(2) **Stop Scan command**

Syntax	`(Key [Enter] only)`
Operation	This command stops Scanning. When stopping Scanning succeeds, the message "`Stop Scan :OK`" is displayed. When stopping Scanning fails, "`Stop Scan :ER`" is displayed.
Condition	Scanning is executed
Example	`Key [Enter]` // stop Scanning `Stop Scan :OK`

(3) Set Scan Type command

Syntax	`type active`
	`type passive`

Operation	This command sets Scan type. When set Scan type to Active Scan, enter "`type active`". When set Scan type to Passive Scan, enter "`type passive`". All this command characters should be entered with lower case. When setting Active Scan succeeds, the message "`Set Active :OK`" is displayed. When setting Passive Scan succeeds, the message "`Set Passive :OK`" is displayed. When there is something wrong with command, nothing is displayed.
Condition	Scanning is stopped

Example	`type active` // set Active Scan
	`Set Active :OK`
	`type passive` // set Passive Scan
	`Set Passive :OK`

(4) Set Scan Channel command

Syntax	`ch 37`
	`ch 38`
	`ch 39`
	`ch all`

Operation	This command sets Scan channel. When set Scan channel to 37 channel, enter "`ch 37`". When set Scan channel to 38 channel, enter "`ch 38`". When set Scan channel to 39 channel, enter "`ch 39`". When set Scan channel to All channels (37,38,39), enter "`ch all`". All this command characters should be entered with lower case. When setting 37 channel succeeds, the message "`Set Channel 37 :OK`" is displayed. When setting 38 channel succeeds, the message "`Set Channel 38 :OK`" is displayed. When setting 39 channel succeeds, the message "`Set Channel 39 :OK`" is displayed. When setting All channels succeeds, the message "`Set Channel All :OK`" is displayed. When there is something wrong with command, nothing is displayed.
Condition	Scanning is stopped

Example	`ch 37` // set 37Channel
	`Set Channel 37 :OK`
	`ch all` // set all Channel (37,38,39)
	`Set Channel All :OK`

(5) Set Scan Interval command

Syntax	`itvl <Scan Interval>`
Operation	This command sets Scan interval. Scan interval (unit: 0.625msec) is specified with hexadecimal number (`XXXX`), enter "`itvl XXXX`". All this command characters and hexadecimal numbers should be entered with lower case. When setting Scan interval succeeds, the message "`Set Interval :OK`" is displayed. When there is something wrong with command, nothing is displayed.
Condition	Scanning is stopped
Example	`itvl a0` // set 100msec(0x00A0) `Set Interval :OK` `itvl 640` // set 1sec(0x0640) `Set Interval :OK`

(6) Flush White List command

Syntax	`wlist flush`
Operation	This command flushes White List. All this command characters should be entered with lower case. When flushing White List succeeds, the message "`Flush White List :OK`" is displayed. When there is something wrong with command, nothing is displayed.
Condition	Scanning is stopped
Example	`wlist flush` // flush White List `Flush White List :OK`

(7) Add White List command

Syntax	`wlist <Device Address Type><Device Address>`
Operation	This command adds Device Address Type and Device Address to White List. When Device Address Type is Public, specify "`pub`", or when Device Address Type is Random, specify "`rnd`". Device Address is specified by 12 digits hexadecimal (`XXXXXXXXXXXX`), enter "`wlist pubXXXXXXXXXXXX`" or enter "`wlist rndXXXXXXXXXXXX`". All this command characters and hexadecimal numbers should be entered with lower case. When adding Device Address to White List succeeds, the message "`Add White List :OK`" is displayed. When there is something wrong with command, nothing is displayed.
Condition	Scanning is stopped
Example	`wlist pub123456789abc` // add Public Device Address 12:34:56:78:9A:BC to White List `Add White List :OK` `wlist rnd47f2bb2c2a79` // add Random Device Address 47:F2:BB:2C:2A:79 to White List `Add White List :OK`

(8) Duplicate Filter command

Syntax	`dup en`
	`dup dis`

Operation	This command enables or disables Duplicate Filter for Advertiser's Address and Advertising data. When enable Duplicate Filter, enter "`dup en`". When disable Duplicate Filter, enter "`dup dis`". All this command characters should be entered with lower case. When enabling Duplicate Filter succeeds, the message "`Enable Dup Filter :OK`" is displayed. When disabling Duplicate Filter succeeds, the message "`Disable Dup Filter :OK`" is displayed. When there is something wrong with command, nothing is displayed.
Condition	Scanning is stopped

Example	`dup en` // enable Duplicate Filtering
	`Enable Dup Filter :OK`
	`dup dis` // disable Duplicate Filtering
	`Disable Dup Filter :OK`

(9) RSSI Filter command

Syntax	`rssi en <RSSI Threshold>`
	`rssi dis`

Operation	This command enables or disables RSSI Filter for packet. When enable RSSI Filter, RSSI threshold -128dBm to 127dBm is specified by decimal number (`XX`), enter "`rssi en XX`". When disable RSSI Filter, enter "`rssi dis`". All this command characters should be entered with lower case. When enabling RSSI Filter succeeds, the message "`Enable RSSI Filter :OK`" is displayed. When disabling RSSI Filter succeeds, the message "`Disable RSSI Filter :OK`" is displayed. When there is something wrong with command, nothing is displayed.
Condition	Scanning is stopped

Example	`rssi en -70` // enable RSSI Filtering and set RSSI Threshold to -70dBm
	`Enable RSSI Filter :OK`
	`rssi dis` // disable RSSI Filtering
	`Disable RSSI Filter :OK`

Scan Application starts Scanning automatically. When key in "Enter" on terminal software, Scan Application stops Scanning. When key in "Enter" again, Scan Application restarts Scanning.

While executing Scanning, Scan Application displays the result of receiving Advertising packet on terminal software when RL78/G1D receives Advertising packet.

While stopping Scanning, Scan Application changes Scan type, Scan channel, Scan interval, White List, Duplicate Filter, and RSSI Filter with respect to the commands keyed in on terminal software.

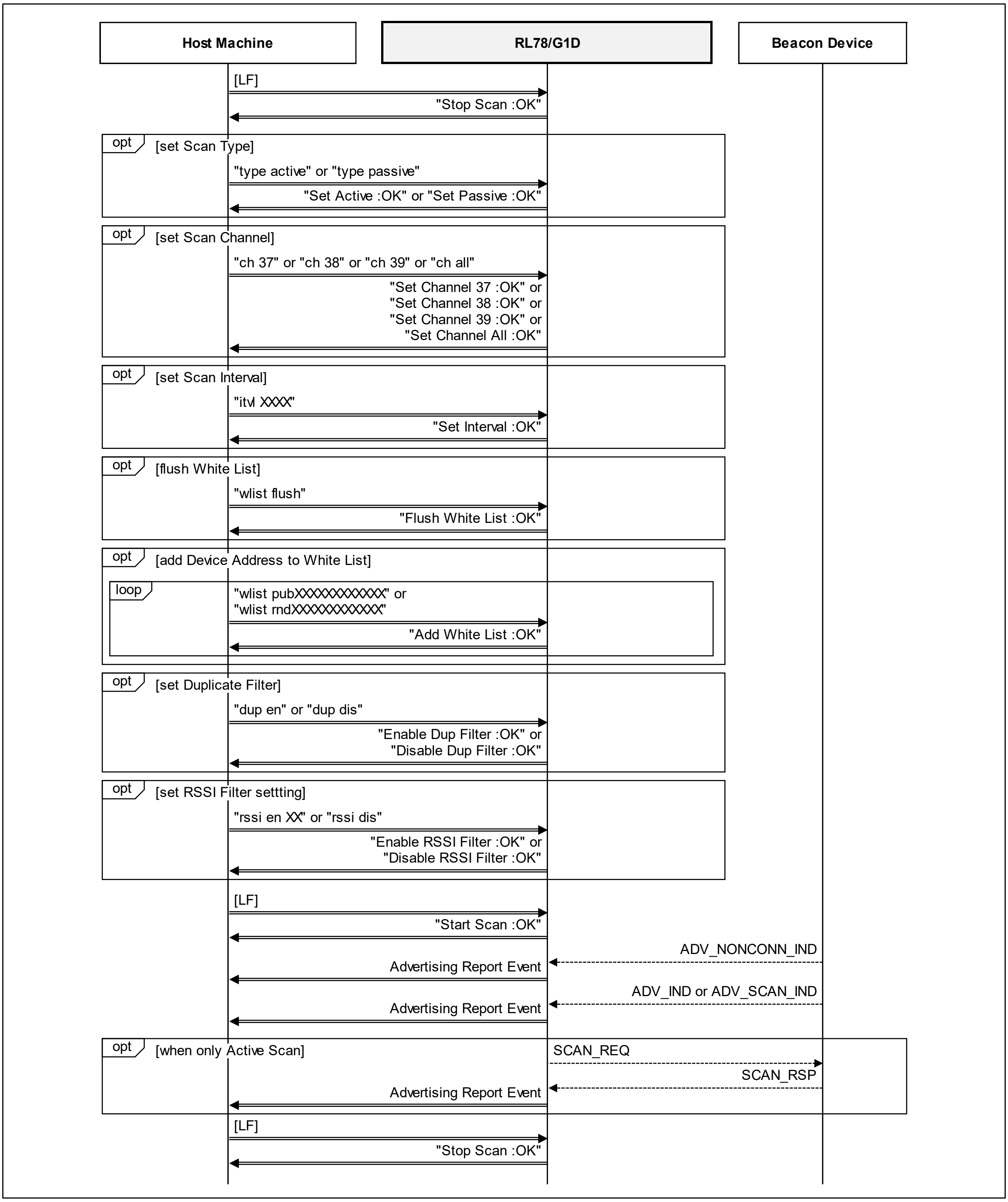


Figure 5-1 ASCII-format UART communication Sequence

5.3.2 Binary-format UART communication commands

This subsection describes specification of Binary-format UART communication, which is implemented in Scan Application. This UART communication is used for connecting between RL78/G1D device and Host MCU.

Table 5-6 shows UART settings for the Binary-format UART communication.

Table 5-6 UART Settings for the Binary-format UART communication

Setting Item	Setting Value
Baud rate	1,000,000bps
Data bit length	8bit
Parity	None
Stop bit length	1bit
Flow control	None

Figure 5-2 shows packet format of the Binary-format UART communication. Regarding to the details of format, refer to the following pages.

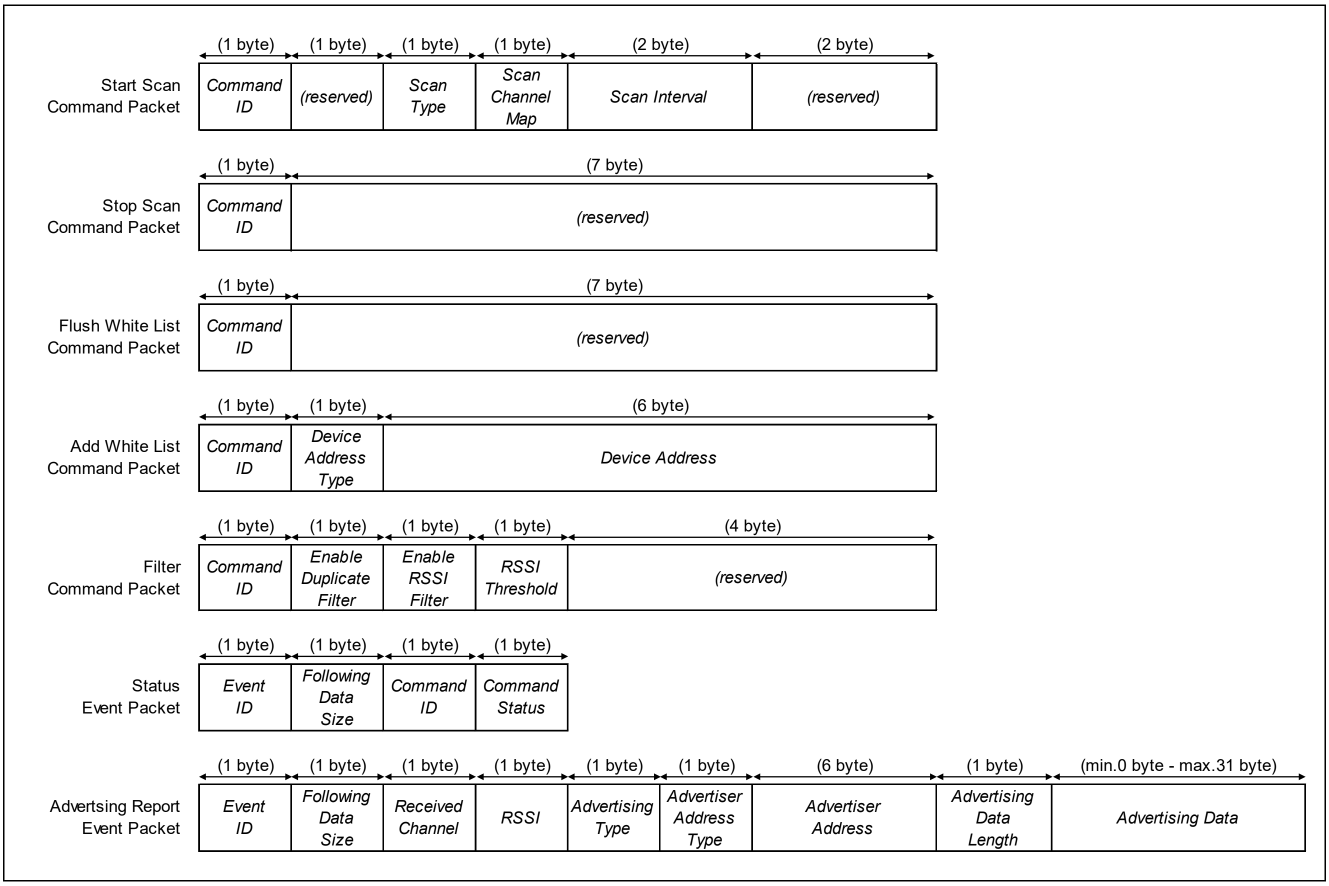

Start Scan Command Packet
(1 byte) Command ID
(1 byte) (reserved)
(1 byte) Scan Type
(1 byte) Scan Channel Map
(2 byte) Scan Interval
(2 byte) (reserved)
Stop Scan Command Packet
(1 byte) Command ID
(7 byte) (reserved)
Flush White List Command Packet
(1 byte) Command ID
(7 byte) (reserved)
Add White List Command Packet
(1 byte) Command ID
(1 byte) Device Address Type
(6 byte) Device Address
Filter Command Packet
(1 byte) Command ID
(1 byte) Enable Duplicate Filter
(1 byte) Enable RSSI Filter
(1 byte) RSSI Threshold
(4 byte) (reserved)
Status Event Packet
(1 byte) Event ID
(1 byte) Following Data Size
(1 byte) Command ID
(1 byte) Command Status
Advertsing Report Event Packet
(1 byte) Event ID
(1 byte) Following Data Size
(1 byte) Received Channel
(1 byte) RSSI
(1 byte) Advertising Type
(1 byte) Advertiser Address Type
(6 byte) Advertiser Address
(1 byte) Advertising Data Length
(min.0 byte - max.31 byte) Advertising Data

Figure 5-2 Packet Format of the Binary-format UART communication

(1) Start Scan command

This command starts Scanning. **Table 5-7** shows the Start Scan command format.

Table 5-7 Start Scan command format

Offset	Size	Data	Value
0x00	0x01	Command ID	0x01: Start Scan
0x01	0x01	(reserved)	-
0x02	0x01	Scan Type	0x00: Passive Scan, 0x01: Active Scan
0x03	0x01	Scan Channel Map	0x01: 37ch, 0x02: 38ch, 0x04: 39ch, 0x07: All ch
0x04	0x02	Scan Interval	min.0x0004 (2.5msec) to max.0xC000 (30.72sec) Byte Order : Least Significant Byte First
0x06	0x02	(reserved)	-
0x08	-	-	-

Example Start Scan command

```
        +0  +1  +2  +3  +4  +5  +6  +7
  0000   01  00  01  07  A0  00  00  00
```

(2) Stop Scan command

This command stops Scanning. **Table 5-8** shows the Stop Scan command format.

Table 5-8 Stop Scan command format

Offset	Size	Data	Value
0x00	0x01	Command ID	0x02: Stop Scan
0x01	0x07	(reserved)	-
0x08	-	-	-

Example Stop Scan command

```
        +0  +1  +2  +3  +4  +5  +6  +7
  0000   02  00  00  00  00  00  00  00
```

(3) Flush White List command

This command flushes White List. **Table 5-9** shows the Flush Write List command format.

Table 5-9 Flush White List command format

Offset	Size	Data	Value
0x00	0x01	Command ID	0x03: Flush White List
0x01	0x07	(reserved)	-
0x08	-	-	-

Example Flush White List command

```
        +0  +1  +2  +3  +4  +5  +6  +7
  0000   03  00  00  00  00  00  00  00
```

(4) Add White List command

This command adds Device Address Type and Device Address to White List. **Table 5-10** shows the Add White List command format.

Table 5-10 Add White List command format

Offset	Size	Data	Value
0x00	0x01	Command ID	0x04: Add White List
0x01	0x01	Device Address Type	0x00: Public, 0x01: Random
0x02	0x06	Device Address	Byte Order : Least Significant Byte First e.g.) if 12:34:56:78:9A:BC, Device Address is notified in order of 0xBC,0x9A,0x78,0x56,0x34,0x12
0x08	-	-	-

Example Add White List command

```
       +0  +1  +2  +3  +4  +5  +6  +7
0000   04  00  BC  9A  78  56  34  12
```

(5) Filter command

This command sets the filtering configuration. **Table 5-11** shows the Filter command format.

Table 5-11 Filter command format

Offset	Size	Data	Value
0x00	0x01	Command ID	0x05: Filter
0x01	0x01	Enable Duplicate Filter	0x00: Disable, 0x01: Enable
0x02	0x01	Enable RSSI Filter	0x00: Disable, 0x01: Enable
0x03	0x01	RSSI Threshold	min.0x80(-128dBm) to max.0x7F(127dBm)
0x04	0x04	(reserved)	-
0x08	-	-	-

Example Filter command

```
       +0  +1  +2  +3  +4  +5  +6  +7
0000   05  01  01  D3  00  00  00  00
```

(6) Status event

This event notifies the result of command executed. **Table 5-12** shows the Status event format.

Table 5-12 Status event format

Offset	Size	Data	Value
0x00	0x01	Event ID	0x10: Status
0x01	0x01	Following Data Size	0x02(byte)
0x02	0x01	Command ID	0x01: Start Scan 0x02: Stop Scan 0x03: Flush White List 0x04: Add White List 0x05: Filter
0x03	0x01	Command Status	Refer to subsection 4.2.1 "Status macro" in RL78/G1D Beacon Stack User's Manual (R01UW0171)
0x04	-	-	-

Example Status event format

```
        +0  +1  +2  +3
0000    10  02  01  00
```

(7) Advertising Report event

This event reports the information of Advertising packet received by Scanning. **Table 5-13** shows the Advertising Report event format.

Table 5-13 Advertising Report event format

Offset	Size	Data	Value
0x00	0x01	Event ID	0x20: Advertising Report
0x01	0x01	Following Data Size	min.0x0B(byte) to max.0x2A(byte)
0x02	0x01	Received Channel	0x25: 37ch, 0x26: 38ch, 0x27: 39ch
0x03	0x01	RSSI	min.0x80(-128dBm) to max.0x7F(127dBm)
0x04	0x01	Advertising Type	0x00: ADV_IND 0x01: ADV_DIRECT_IND 0x02: ADV_NONCONN_IND 0x04: SCAN_RSP 0x06: ADV_SCAN_IND
0x05	0x01	Advertiser Address Type	0x00: Public, 0x01: Random
0x06	0x06	Advertiser Address	Byte Order : Least Significant Byte First e.g.) if 12:34:56:78:9A:BC, Advertiser Address is notified in order of 0xBC,0x9A,0x78,0x56,0x34,0x12
0x0C	0x01	Advertising Data Length	min.0x00(byte) to max.0x1F(byte)
0x0D	0x00-0x1F	Advertising Data	-
0x0D-0x2C	-	-	-

Example Advertising Report event

```
        +0  +1  +2  +3  +4  +5  +6  +7  +8  +9  +A  +B  +C  +D  +E  +F
0000    20  26  25  DB  02  00  B1  9A  78  56  34  12  1B  02  01  06
0010    03  03  AA  FE  13  16  AA  FE  10  EE  02  67  6F  6F  2E  67
0020    6C  2F  35  77  4B  6B  52  4B
```

When Host MCU issues Start Scan command, RL78/G1D device starts Scanning. Similarly, when Host MCU issues Stop Scan command, RL78/G1D device stops Scanning.

While executing Scanning, Scan Application notifies Advertising Report event to Host MCU when RL78/G1D device received Advertising packet.

While stopping Scanning, Scan Application updates White List and Filters setting when receive Flush White List command or Add White List command from Host MCU respectively.

After issuing the command to RL78/G1D device, Host MCU should confirm the Status event whether acknowledged by RL78/G1D device or not. If RL78/G1D device does not acknowledge the Status event, Host MCU should issues same command again.

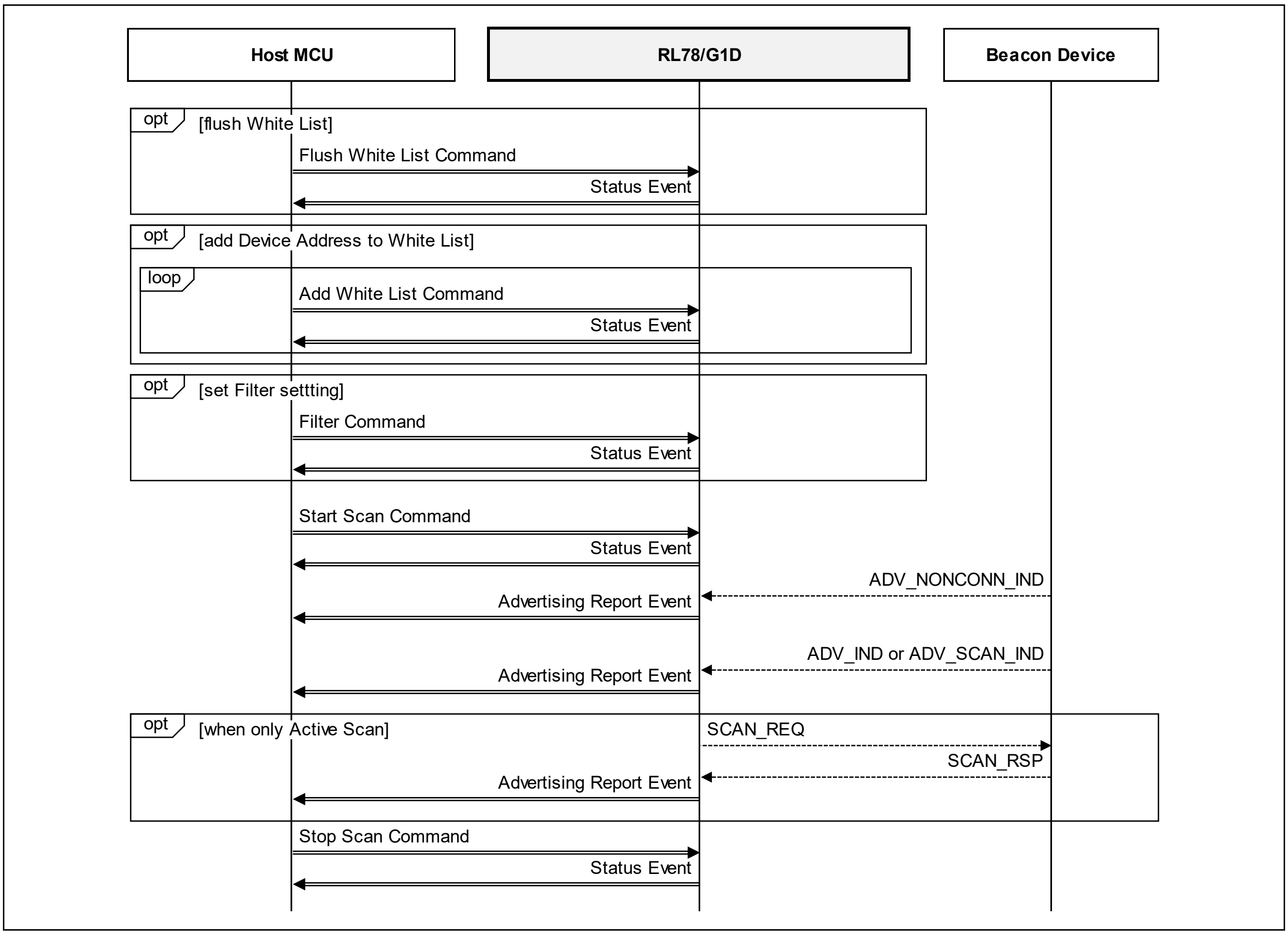


Figure 5-3 Binary-format UART communication sequence

5.4 DTM Application

DTM Application executes Direct Test Mode for evaluating RF characteristic. By connecting the evaluation board to RF Tester with UART, and communicating RF Test packets through UART, it is possible to control Direct Test Mode.

5.4.1 Direct Test Mode

Figure 5-4 shows RF Test packet format.

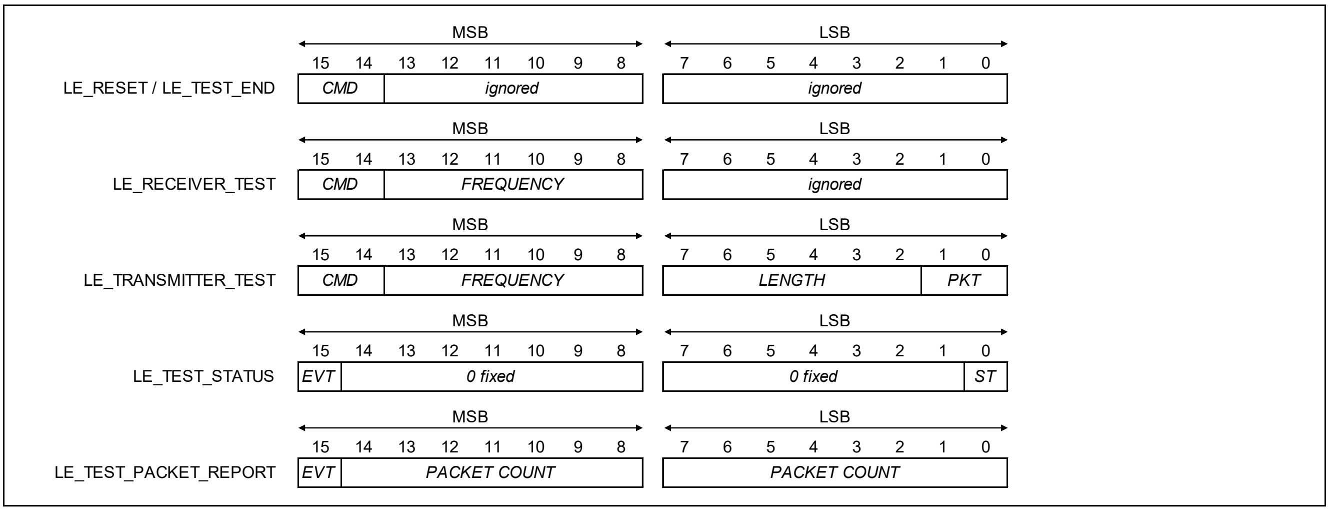


Figure 5-4 RF Test packet format

Table 5-14 shows RF test command packet format for executing Direct Test Mode.

Table 5-14 RF Test command packet format

RF Test Command	Parameters		
LE_RESET	(not referred)		
LE_RECEIVER_TEST	FREQUENCY	Rx Frequency : (2*FREQUENCY+2402) MHz FREQUENCY=0x00(2402MHz) to 0x27(2480MHz)	
LE_TRANSMITTER_TEST	FREQUENCY	Tx Frequency : (2*FREQUENCY+2402) MHz FREQUENCY=0x00(2402MHz) to 0x27(2480MHz)	
	LENGTH	Tx Packet Payload Length LENGTH=0x00-0x25 Bytes	
	PKT	Tx Packet Payload Type	
		0	9-bit pseudo-random sequence (PRBS9)
		1	b'11110000 bits sequence
		2	b'10101010 bits sequence
		3	15-bit pseudo-random sequence (PRBS15)
		4	b'11111111 bits sequence
		5	b'00000000 bits sequence
		6	b'00001111 bits sequence
		7	b'01010101 bits sequence
LE_TEST_END	(not referred)		

Table 5-15 shows RF Test event format for responding a result of Direct Test Mode.

Table 5-15 RF Test event packet format

RF Test Event	Parameters		
LE_TEST_STATUS	ST	Status	
		0	Success
		1	Error
LE_TEST_PACKET_REPORT	PACKET COUNT	Rx Packet Count : 0-32767	

When receive LE_RECEIVER_TEST command from RF Tester, RL78/G1D device starts RF Receiver Test and receives LE Test packets from RF Tester. When receive LE_TEST_END command from RF Tester, RL78/G1D device transmits LE_PACKET_REPORT event and notifies the number of received packets.

When receive LE_TRANSMITTER_TEST command from RF Tester, RL78/G1D device starts RF Transmitter Test and transmits LE Test packets to RF Tester. When receive LE_TEST_END command from RF Tester, RL78/G1D device stops to transmit LE Test packets and transmits LE_PACKET_REPORT event.

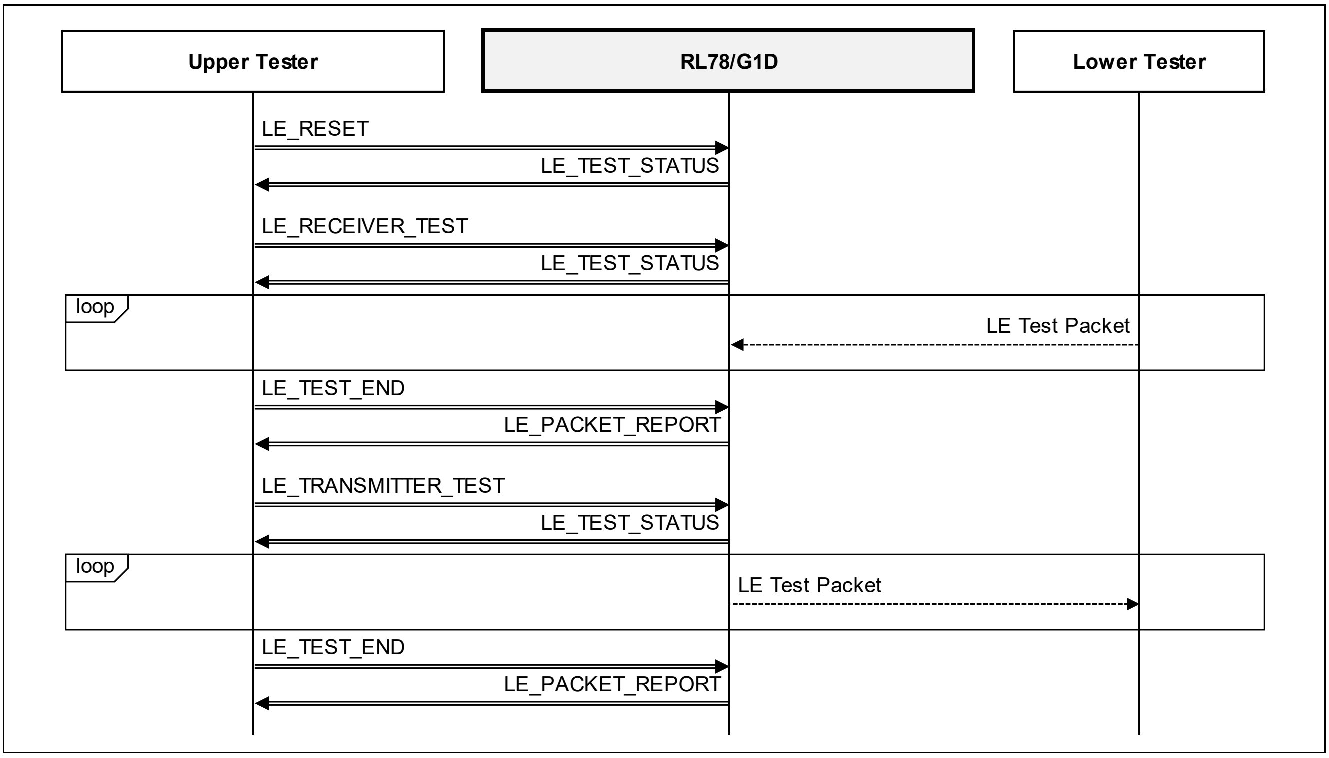


Figure 5-5 2-wire UART Direct Test Mode communication sequence

Regarding to the specification of Direct Test mode, refer to [Vol. 6, Part F] Section 3.3, Bluetooth Core Specification v4.2.

5.5 Accessing to Flash memory

5.5.1 Accessing to Code Flash memory

To store parameters as system configuration, Beacon Application and Scan Application use part of Code Flash memory, which is allocated outside of the Sample Program firmware memory range. The system configuration is used to store parameters, which are needed to be different from each and individual device.

Beacon Application loads Device Address, Device Address Type, and Advertising Information from the system configuration. Then start Advertising.

Scan Application loads Device Address and Device Address Type from the system configuration. Then start Scanning.

Table 5-16 shows the specification of the system configuration. Regarding to the location of the system configuration, refer to section 5.10 "Address Map" in this document.

Table 5-16 System Configuration in Code Flash memory

offset	data		size	read (YES:read, NO:not read)	
				Beacon Application	Scan Application
0x00	Device Address (RBLE_BD_ADDR structure)		6 byte	YES	YES
0x06	Device Address Type 0x00: public, 0x01: random (uint8_t type)		1 byte	YES	YES
0x07	(reserved)		1 byte	NO	NO
0x08	Advertising Information (RBLE_ADV_INFO structure)		18 byte		
		interval	2 byte	YES	NO
		delay	1 byte	YES	NO
		ch_map	1 byte	YES	NO
		loop_cnt	1 byte	YES	NO
		tx_pwr	1 byte	YES	NO
		own_addr	6 byte	NO	NO
		own_addr_type	1 byte	NO	NO
		data_cnt	1 byte	NO	NO
		data	2 byte	NO	NO
		evt_permit	1 byte	NO	NO
		use_wl	1 byte	NO	NO
0x1A	Non-connectable Undirected Advertising packet data (RBLE_ADV_DATA structure)		32 byte		
		len	1 byte	YES	NO
		data	31 byte	YES	NO
0x3A	Scannable Undirected Advertising packet data (RBLE_ADV_DATA structure)		32 byte		
		len	1 byte	YES	NO
		data	31 byte	YES	NO
0x5A	Scan Response packet data (RBLE_ADV_DATA structure)		32 byte		
		len	1 byte	YES	NO
		data	31 byte	YES	NO
0x7A	-		-	-	

5.5.2 Accessing to Code Flash memory (Beacon Appliation-ASCII command control)

To store parameters as system configuration, Beacon Application - ASCII command contol use part of Code Flash memory, which is allocated outside of the Sample Program firmware memory range. The system configuration is used to store parameters, which are needed to be different from each and individual device.

The Beacon Application-ASCII command control reads device address, device address type, Advertising information, Advertising packet data and White List when the setting is saved in the system configuration area. If not saved, use the application's default value.

Table 5-17 shows the specification of the system configuration. Regarding to the location of the system configuration, refer to subsection 6.1.8 "System Configuration Address" in this document.

Table 5-17 System Configuration in Code Flash memory (Beacon Application-ASCII command control)

offset	data		size		note
0x00	Device Address (RBLE_BD_ADDR structure)		6 byte		Set the Device Address and the Device Address Type to own_addr and own_addr_type of advertising information at application startup.
0x06	Device Address Type 0x00: public, 0x01: random (uint8_t type)		1 byte		
0x07	(reserved)		1 byte		-
0x08	Advertising Information (RBLE_ADV_INFO structure)		18 byte		-
		interval		2 byte	
		delay		1 byte	
		ch_map		1 byte	
		loop_cnt		1 byte	
		tx_pwr		1 byte	
		own_addr		6 byte	Device address (offset: 0x00) and Device Address Type (offset: 0x06) are set.
		own_addr_type		1 byte	
		data_cnt		1 byte	The advertising packet data specified at the start of Advertising is set.
		data		2 byte	
		evt_permit		1 byte	-
		use_wl		1 byte	
0x1A	Non-connectable Undirected Advertising packet data (RBLE_ADV_DATA structure)		320 byte		It can save up to 10 Non-connectable Undirected Advertising datas.
		len		1 byte	
		data		31 byte	
		:		-	
0x15A	Scannable Undirected Advertising packet data (RBLE_ADV_DATA structure)		320 byte		It can save up to 10 Scannable Undirected Advertising packett datas. (5 Advertising Datas and 5 Scan Response datas.)
		len		1 byte	
		data		31byte	
		:		-	
0x29A	White List (RBLE_DEV_INFO structure)		128 byte		It can save up to 16 White Lists.
		dev_type		1 byte	
		reserved		1 byte	
		RBLE_BD_ADDR		6 byte	
		:		-	

5.6 Hardware Resources used

Hardware resources, which used by the Sample Program with default settings, are as shown below.

RL78/G1D MCU Unit		
	Clock generator	Common • use 8MHz from High-speed On-chip Oscillator as MCU main system clock
		Common • not use XT1 oscillator (use RF on-chip oscillator for generating RF slow clock)
	Clock output/buzzer output	Common • not output clock generated XT1 oscillation from PCLBUZ0 pin
	Timer Array Unit	Beacon Stack • use TM00, and set operation clock CK00 to 1MHz Scan Application and DTM Application • use TM01, and set operation clock CK01 to 125kHz
	Serial array unit	Beacon Stack • use CSI21 Scan Application, DTM Application and Beacon Application-ASCII command control • use UART0
	DMA controller	Beacon Stack • use DMA2 and DMA3 Scan Application and DTM Application • use DMA0 and DMA1
	Interrupt	Beacon Stack • use INTRF, INTDMA2, INTDMA3, and INTTM00 Beacon Application • use INTP5 Scan Application • use INTP5, INTDMA0, INTDMA1, INTSR0, INTSRE0, INTST0, and INTTM01 DTM Application • use INTDMA0, INTDMA1, INTSR0, INTSRE0, INTST0, and INTTM01
	Port	Common • use P10, for DIP switch SW6 position-1 input on the evaluation board • use P02, for DIP switch SW6 position-4 input on the evaluation board • use P16, for switch SW2 input on the evaluation board • use P120,P147, P03, and P60, for controlling LED1, 2, 3, and 4 on the evaluation board
RL78/G1D RF Unit		
	DC-DC Converter	use RF on-chip DC-DC converter
	Oscillator for RF slow clock	use RF on-chip oscillator
	Clock Output	clock output from CLKOUT_RF pin is disabled
RL78/G1D Evaluation Board		
	USB communication	Scan Application • use USB interface for communication to Host machine
	Input functions	Common • use DIP switch SW6 position-1 and position-4, for selecting application Beacon Application • use push switch SW2, for switching Advertising Type ASCII-format UART Scan Application • use push switch SW2, for switching display format of packet data
	Display	Beacon Application and DTM Application • use LED4, for indicating that Application is started Scan Application • use from LED1 to LED4, for indicating that packet is received

5.7 Compiler

The library of Beacon Stack is generated by below compiler. It is necessary to use CC-RL compiler for developing application, which uses Beacon Stack.

Compiler	:	Renesas CC-RL V1.09.00

5.8 Memory Model

The memory model of Beacon Stack is medium model. It is necessary to set below option in the compile option of application which uses Beacon Stack.

Memory Model	:	-memory_model=medium

5.9 Program Size

Table 5-18 shows the total memory usage in the Sample Program.

Target Device	:	R5F11AGJ
Compiler	:	Renesas CC-RL V1.09.00
Compile Configuration	:	default configuration of the Sample Program released

Table 5-18 Sample Program Total Program Size

ROM SIZE	26,687byte PROGRAM SECTION + ROMDATA SECTION
RAM SIZE	8,468byte RAMDATA SECTION (not included stack memory which program consumes for calling functions and allocating auto variables)

Table 5-19 shows the total memory usage in the Sample Program.

Target Device	:	R5F11AGJ
Compiler	:	Renesas CC-RL V1.09.00
Compile Configuration	:	default configuration of the Beacon Application-ASCII command control

Table 5-19 Beacon Application-ASCII command control Total Program Size

ROM SIZE	37,421byte PROGRAM SECTION + ROMDATA SECTION
RAM SIZE	5,566byte RAMDATA SECTION (not included stack memory which program consumes for calling functions and allocating auto variables)

Regarding to the section specification, refer to chapter 6 "SECTION SPECIFICATIONS"(R20UT3123) in CC-RL Compiler User's Manual.

5.10 Address Map

Figure 5-6 shows the address map of the Sample Program for RL78/G1D (R5F11AGG) device. (excluding Beacon Application-ASCII command control.)

Under-lined values are different for R5F11AGH and R5F11AGJ devices.

Address	Area Size	Section	Section Name
	131,072byte	Code Flash memory	
0x00000	128byte	Vector table area	.vect
0x00080	64byte	CALLT table area	.callt0
0x000C0	4byte	Option byte area	.option_byte
0x000C4	10byte	Security ID setting area	.security_id
0x000CE	129,842byte	Program area (below sections are described in no particular order)	
		OCD monitor	.monitor1
		Startup	BOOT0_TEXT
		Runtime library	.RLIB
		Standard library	.SLIB
		Beacon Stack	BCN_CONST, BCN_TEXT
		Application	.const, .constf, .data, .text, .textf
		Unused area	-
0x1FC00	122byte	System Configuration area	
0x1FE00		.monitor2	
0x20000		Reserved	
0xF0000	2048byte	Special function register(2nd SFR)	
0xF0800		Reserved	
0xF1000	8192byte	DataFlash memory	
0xF3000	40,704byte	Mirror area	
0xFCF00	12,064byte	RAM area	
		Program Resource area (below sections are described in no particular order)	
		Application	.bss, .dataR
		Beacon Stack	BCN_BSS
		Unused area	-
		Stack area	-
0xFFEE0	32byte	General-purpose register	
0xFFF00	256byte	Special function register(SFR)	
0xFFFFF			

Figure 5-6 Address Map (R5F11AGG)

Figure 5-7 shows the address map of the Sample Program for RL78/G1D (R5F11AGH) device. (excluding Beacon Application-ASCII command control.)

Under-lined values are different for R5F11AGG and R5F11AGJ devices.

Address	Area Size	Section	Section Name
	196,608byte	Code Flash memory	
0x00000	128byte	Vector table area	.vect
0x00080	64byte	CALLT table area	.callt0
0x000C0	4byte	Option byte area	.option_byte
0x000C4	10byte	Security ID setting area	.security_id
0x000CE	195,378byte	Program area (below sections are described in no particular order)	
		OCD monitor	.monitor1
		Startup	BOOT0_TEXT
		Runtime library	.RLIB
		Standard library	.SLIB
		Beacon Stack	BCN_CONST, BCN_TEXT
		Application	.const, .constf, .data, .text, .textf
		Unused area	-
0x2FC00	122byte	System Configuration area	
0x2FE00		.monitor2	
0x30000		Reserved	
0xF0000	2048byte	Special function register(2nd SFR)	
0xF0800		Reserved	
0xF1000	8192byte	DataFlash memory	
0xF3000	36,608byte	Mirror area	
0xFBF00	16,160byte	RAM area	
		Program Resource area (below sections are described in no particular order)	
		Application	.bss, .dataR
		Beacon Stack	BCN_BSS
		Unused area	-
		Stack area	-
0xFFEE0	32byte	General-purpose register	
0xFFF00	256byte	Special function register(SFR)	
0xFFFFF			

Figure 5-7 Address Map (R5F11AGH)

Figure 5-8 shows the address map of the Sample Program for RL78/G1D (R5F11AGJ) device. (excluding Beacon Application-ASCII command control.)

Under-lined values are different for R5F11AGG and R5F11AGH devices.

Address	Area Size	Section	Section Name
	262,144byte	Code Flash memory	
0x00000	128byte	Vector table area	.vect
0x00080	64byte	CALLT table area	.callt0
0x000C0	4byte	Option byte area	.option_byte
0x000C4	10byte	Security ID setting area	.security_id
0x000CE	258,866byte	Program area (below sections are described in no particular order)	
		OCD monitor	.monitor1, .monitor2
		Startup	BOOT0_TEXT
		Runtime library	.RLIB
		Standard library	.SLIB
		Beacon Stack	BCN_CONST, BCN_TEXT
		Application	.const, .constf, .data, .text, .textf
		Unused area	-
0x3F400	122byte	System Configuration area	
0x3F47A		Unused area	
0x3F800	512byte	Reserved area (RL78/G1D Module only)	
0x3FC00	6byte	User Information area	
0x3FC06		Unused area	
0x40000		Reserved	
0xF0000	2048byte	Special function register(2nd SFR)	
0xF0800		Reserved	
0xF1000	8192byte	DataFlash memory	
0xF3000	32,512byte	Mirror area	
0xFAF00	1024byte	Self RAM area (R5F11AGJ only)	
0xFB300	20,447byte	RAM area	
		Program Resource area (below sections are described in no particular order)	
		Application	.bss, .dataR
		Beacon Stack	BCN_BSS
		Unused area	-
		Stack area	-
0xFFEE0	32byte	General-purpose register	
0xFFF00	256byte	Special function register(SFR)	
0xFFFFF			

Figure 5-8 Address Map (R5F11AGJ)

6. Configuration

This chapter describes configurations for hardware and application of the Sample Program.

Regarding to the hardware resources used by Beacon Stack, refer to section 2.1 "Hardware Resources used" in RL78/G1D Beacon Stack User's Manual (R01UW0171). Regarding to the specification of Beacon Stack API, refer to chapter 4 "API" in RL78/G1D Beacon Stack User's Manual (R01UW0171).

6.1 Hardware configuration

For using Beacon Stack, major hardware configurations are arranged to macro definitions in r_config.h file. Regarding to the details about macro definitions, refer to following subsections.

r_config.h, line 34-75

```
34:   /*
35:    * CONFIGURATION (NEED TO CHANGE BELOW DEFINES AS NECESSARY)
36:    ******************************************************************************************
37:    */
38:   /* MCU Main System Clock (either clock frequency of 4MHz,8MHz,16MHz,32MHz)          */
39:   /*  Note: It is necessary to set Option Bytes Value at Device Setting of Linker Option */
40:   #define MCU_HOCO_CLK         (8)
41:
42:   /* RF Operation (0:enable both Tx and Rx, 1:enable Tx only) */
43:   #define RF_TX_ONLY          (0)
44:
45:   /* RF DC-DC Converter (0:disable DC-DC, 1:enable DC-DC) */
46:   #define RF_DCDC_EN          (1)
47:
48:   /* RF Slow Clock Source (0:RF On-Chip Oscillator, 1:MCU XT1 Oscillator) */
49:   #define RF_SLK_XT1          (0)
50:
51:   /* RF Slow Clock Calibration (0:not execute, 1:execute)  */
52:   /*  Note: Calibration is only for RF-On_Chip_Oscillator */
53:   #define RF_SLK_CAL          (1)
54:
55:   /* RF High-speed clock output from CLKOUT_RF (0:not output, 4:4MHz, 8:8MHz, 16:16MHz) */
56:   #define RF_CLKOUT           (0)
57:
58:   /* RF 32MHz Oscillation Stabilization Time (usec, at least 550usec)    */
59:   /*  Note: Stabilization Time needs to be optimized for 32MHz resonator */
60:   #define RF_32MHZ_WAIT        (1000)
61:
62:   /* System Configuration Address in CodeFlash memory */
63:   #if defined(_USE_R5F11AGG)
64:      /* System Configuration is located the last block */
65:      #define SYSCFG_ADDR      (0x1FC00)
66:   #elif defined(_USE_R5F11AGH)
67:      /* System Configuration is located the last block */
68:      #define SYSCFG_ADDR      (0x2FC00)
69:   #elif defined(_USE_R5F11AGJ)
70:      /* System Configuration is located the third last block */
71:      /* by taking into account the location of RL78/G1D module (RY7011) */
72:      #define SYSCFG_ADDR      (0x3F400)
73:      /* In the case of RL78/G1D Module (RY7011), Device Address is located the last block */
74:      #define MODCFG_ADDR      (0x3FC00)
75:   #endif
```

6.1.1 MCU main system clock frequency

Clock generated by Hi-speed On-chip Oscillator is used as MCU main system clock, and selectable frequencies of MCU main system clock are 4, 8, 16 and 32MHz. In the Sample Program, frequency of MCU main system clock is defined by the macro MCU_HOCO_CLK and Option Bytes. The default setting of clock frequency is 8 (MHz).

If need to change the frequency of MCU main system clock, change the macro value to one of the values: 4 (MHz), 8 (MHz), 16 (MHz), or 32 (MHz).

r_config.h, line 39-40

```
38:   /* MCU Main System Clock (either clock frequency of 4MHz,8MHz,16MHz,32MHz)         */
39:   /*  Note: It is necessary to set Option Bytes Value at Device Setting of Linker Option */
40:   #define MCU_HOCO_CLK         (8)
```

Option Bytes is set to the linker option "-user_opt_byte". Regarding to the value of Option Bytes, refer to **Table 6-1**.

Table 6-1 Option Bytes value setting

Option Bytes setting			Clock frequency	Flash Operation Mode
000C0	000C1	000C2		
(any)	(any)	2B	4MHz	low-voltage main mode
		AA	8MHz	low-speed main mode
		E9	16MHz	high-speed main mode
		E8	32MHz	

Regarding to the details about Option Bytes, refer to chapter 25 "OPTION BYTE" in RL78/G1D User's Manual: Hardware (R01UH0515).CPU operation voltage varies with respect to CPU clock frequency. Regarding to the operation voltage, refer to section 30.2 "Operating Voltage" in RL78/G1D User's Manual: Hardware (R01UH0515).

(1) Using CS+ for CC

In the case of using CS+ for CC on how to set Option Bytes, follow the below steps.

1 Right-click to [CC-RL (Build Tool)] of the subproject "R5F11AGJ_Beacon" in the project tree.

2 Select [Property] in right click menu.

3 Set the Option Bytes at the [Device]→[User option byte value] of [Link Options] tab.

(2) Using e^2 studio

In the case of using e^2 studio on how to set Option Bytes, follow the below steps.

1 Right-click to "R5F11AGJ_Beacon" project.

2 Select [Renesas Tool Settings] in right click menu.

3 Set the Option Bytes at the [Linker]→[Device]→[User option byte value] of [Tool Settings] tab.

6.1.2 RF Operation

It is possible to select whether to enable both Tx and Rx or only Tx. When enabling only Tx is selected, RF initialization time is shortened. In the Sample Program, whether to enable both Tx and Rx or only Tx is defined by the macro RF_TX_ONLY. The default setting is 0, which means that RF operation is enabled both Tx and Rx.

If need to enable only Tx, change the macro value to 1.

r_config.h, line 42-43

```
42:   /* RF Operation (0:enable both Tx and Rx, 1:enable Tx only) */
43:   #define RF_TX_ONLY          (0)
```

6.1.3 RF on-chip DC-DC converter

In the Sample Program, whether to use RF on-chip DC-DC converter is defined by the macro RF_DCDC_EN. Thus, it is possible to select whether to use RF on-chip DC-DC converter or not. The default setting is 1, which means that RF on-chip DC-DC converter is used.

If need not to use RF on-chip DC-DC converter, change the macro value to 0.

r_config.h, line 45-46

```
45:   /* RF DC-DC Converter (0:disable DC-DC, 1:enable DC-DC) */
46:   #define RF_DCDC_EN          (1)
```

6.1.4 RF slow clock source

RF slow clock is needed to RF unit for counting the period, and it is possible to select as a source of RF clock from either RF on-chip oscillator or MCU unit XT1 oscillator. In the Sample Program, RF slow clock source is defined by the macro RF_SLK_XT1. The default setting is 0, which means that RF on-chip oscillator is selected as a source for RF slow clock.

If need to change RF slow clock source to MCU unit XT1 oscillator, change the macro value to 1. By changing the macro to 1, clock generated by MCU unit XT1 oscillator is supplied to RF unit through EXSLK_RF pin.

r_config.h, line 48-49

```
48:   /* RF Slow Clock Source (0:RF On-Chip Oscillator, 1:MCU XT1 Oscillator) */
49:   #define RF_SLK_XT1          (0)
```

6.1.5 RF on-chip oscillator calibration

In the case of using RF on-chip oscillator as a source of RF slow clock, calibrating accuracy of clock generated by RF on-chip oscillator is always executed when Protocol Stack works. But it is possible to select whether to execute calibration or not when Beacon Stack works. In the Sample Program, whether to execute calibration is defined by the macro RF_SLK_CAL. The default setting is 1, which means that the calibration is executed.

If not executing calibration, change the macro value to 0.

r_config.h, line 51-53

```
51:   /* RF Slow Clock Calibration (0:not execute, 1:execute) */
52:   /*  Note: Calibration is only for RF-On_Chip_Oscillator */
53:   #define RF_SLK_CAL          (1)
```

6.1.6 Output frequency-divided clock of RF base clock

It is selectable whether to output frequency-divided clock (4, 8, 16 or 32MHz) of RF base clock from CLKOUT_RF pin or not. In the Sample Program, whether to output frequency-divided clock or not is defined by the macro RF_CLKOUT. The default setting is 1, which means that no frequency-divided clock output. Note that it is impossible to use frequency-divided clock of RF base clock for MCU main system clock.

If need to change Frequency-divided clock of RF base clock configuration, change the macro value to 0, 4(MHz), 8(MHz) or 16(MHz).

r_config.h, line 55-56

```
 55:   /* RF High-speed clock output from CLKOUT_RF (0:not output, 4:4MHz, 8:8MHz, 16:16MHz) */
 56:   #define RF_CLKOUT           (0)
```

6.1.7 RF base clock oscillation stabilization time

In the Sample Program, the oscillation stabilization time is defined by the macro RF_32MHZ_WAIT. Thus, it is necessary to optimize the oscillation stabilization time of XTAL_RF oscillator for using RF base clock, which is depending on the 32MHz resonator connected to XTAL1_RF and XTAL2_RF pin. The default setting is 1000 (usec) which is suitable for the particular RL78/G1D Evaluation Board.

If need to change the oscillation stabilization time, change the macro value to the time as minimum 550 (usec).

r_config.h, line 58-60

```
 58:   /* RF 32MHz Oscillation Stabilization Time (usec, at least 550usec)    */
 59:   /*  Note: Stabilization Time needs to be optimized for 32MHz resonator */
 60:   #define RF_32MHZ_WAIT        (1000)
```

6.1.8 System Configuration Address

In the Code Flash memory, it is possible to store information as system configuration outside of the firmware memory range. By setting each different system configuration to different devices, it is possible to configure the information without rebuilding firmware. For example, this information includes device address, advertising data, etc. In the Sample Program, the address of system configuration is defined by the macro SYSCFG_ADDR.

If needed to re-assign the address map, change the macro value to new address.

r_config.h, line 62-75

```
 62:   /* System Configuration Address in CodeFlash memory */
 63:   #if defined(_USE_R5F11AGG)
 64:      /* System Configuration is located the last block */
 65:      #define SYSCFG_ADDR       (0x1FC00)
 66:   #elif defined(_USE_R5F11AGH)
 67:      /* System Configuration is located the last block */
 68:      #define SYSCFG_ADDR       (0x2FC00)
 69:   #elif defined(_USE_R5F11AGJ)
 70:      /* System Configuration is located the third last block */
 71:      /* by taking into account the location of RL78/G1D module (RY7011) */
 72:      #define SYSCFG_ADDR       (0x3F400)
 73:      /* In the case of RL78/G1D Module (RY7011), Device Address is located the last block */
 74:      #define MODCFG_ADDR       (0x3FC00)
 75:   #endif
```

Regarding to the details about System Configuration, refer to subsection 5.5.1 "Accessing to Code Flash memory" in this document.

6.1.9 Hardware configuration for Energy Harvesting

To transmit Advertising packets from Beacon Application by using limited energy, which is generated by such as energy harvesting, example configuration for reducing power consumption is as shown below.

To shorten RF initialization time, select only Tx as RF Operation configuration. Regarding to the details about this configuration, refer to subsection 6.1.2 "RF Operation" in this document.

r_config.h, line 42-43

```
42:   /* RF Operation (0:enable both Tx and Rx, 1:enable Tx only) */
43:   #define RF_TX_ONLY          (1)
```

To reduce RF transmission current, select to use RF on-chip DC-DC converter. Regarding to the details about this configuration, refer to subsection 6.1.3 "RF on-chip DC-DC converter" in this document.

r_config.h, line 45-46

```
45:   /* RF DC-DC Converter (0:disable DC-DC, 1:enable DC-DC) */
46:   #define RF_DCDC_EN          (1)
```

To omit stabilization time of MCU unit XT1 oscillator, select to use RF on-chip oscillator. Regarding to the details about this configuration, refer to subsection 6.1.4 "RF slow clock source" in this document.

r_config.h, line 48-49

```
48:   /* RF Slow Clock Source (0:RF On-Chip Oscillator, 1:MCU XT1 Oscillator) */
49:   #define RF_SLK_XT1          (0)
```

To omit accuracy calibration of RF on-chip oscillator clock, select not to execute calibration. Regarding to the details about this configuration, refer to subsection 6.1.5 "RF on-chip oscillator calibration" in this document.

r_config.h, line 51-53

```
51:   /* RF Slow Clock Calibration (0:not execute, 1:execute)  */
52:   /*  Note: Calibration is only for RF-On_Chip_Oscillator */
53:   #define RF_SLK_CAL          (0)
```

6.2 Application configuration

6.2.1 Application Selection configuration

It is possible to select application either by switching DIP switch of RL78/G1D Evaluation board before power up a firmware, or by setting macro before building a firmware. In the Sample Program, Application Selection is defined by the macro APP_SELECT. The default setting is 0, which means that application is selected by switching DIP switch of RL78/G1D Evaluation Board.

If need to use no DIP switch and enable Beacon Application, change the macro value to 1. Similary, if need to enable ASCII-format Application, change the macro value to 2. If need to enable Binary-format Application, change the macro value to 3. If need to enable DTM Application, change the macro value to 4.

If need to enable Beacon Application-ASCII command control, change the macro value to 5. It cannot be selected by the DIP switch on the evaluation board.

r_config.h, line 62-69

```
62:   /* Application Selection                                    */
63:   /*  0: select by DIP SW6-1 and SW6-4 before start up firmware */
64:   /*  1: enable Beacon Application only                       */
65:   /*  2: enable UART-ASCII  Scan Application only             */
66:   /*  3: enable UART-Binary Scan Application only             */
67:   /*  4: enable DTM Application only                          */
68:   /*  5: enable UART-ASCII  Beacon Application only           */
69:   #define APP_SELECT          (0)
```

As a sample, the release package includes below firmware files.

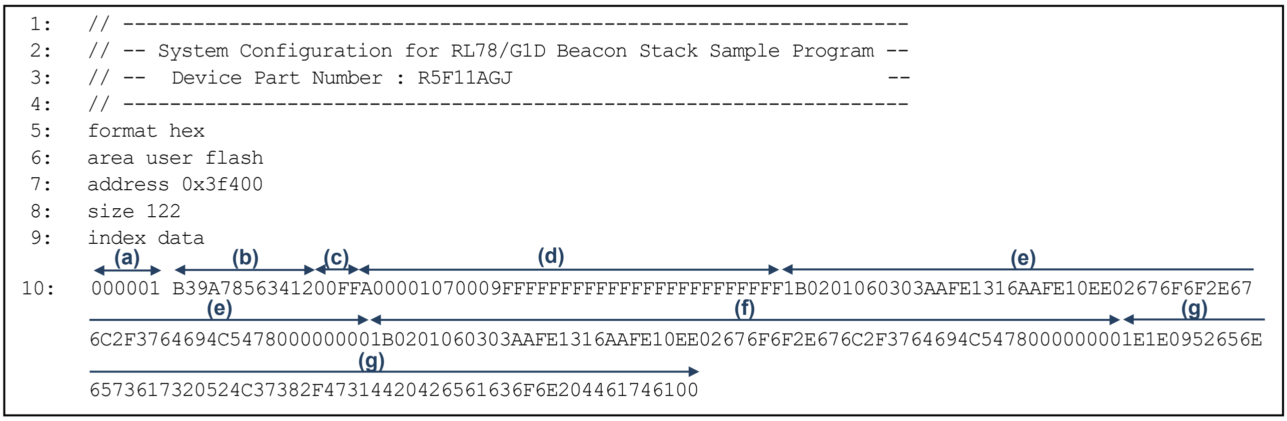
- firmware files built with APP_SELECT=0 : R5F11AGJ_Beacon.hex
- firmware files built with APP_SELECT=1 : R5F11AGJ_Beacon(beacon).hex
- firmware files built with APP_SELECT=2 : R5F11AGJ_Beacon(scan_ascii).hex
- firmware files built with APP_SELECT=3 : R5F11AGJ_Beacon(scan_bin).hex
- firmware files built with APP_SELECT=4 : R5F11AGJ_Beacon(dtm).hex
- firmware files built with APP_SELECT=5 : R5F11AGJ_Beacon(beacon_ascii).hex

6.2.2 System Configuration

System configuration is allocated outside of the Sample Program firmware in the Code Flash memory. Thus, it is possible to write System configuration and firmware at the same time, by using Unique Code Embedding Function of Renesas Flash Programmer.

As a sample, the release package includes unique code file for system configuration. Regarding to the details about system configuration defined by Sample Program, refer to subsection 5.5.1 "Accessing to Code Flash memory" in this document.

r5f11agj_syscfg.ruc, line 1-10



```
 1:   // -------------------------------------------------------------------
 2:   // -- System Configuration for RL78/G1D Beacon Stack Sample Program --
 3:   // --  Device Part Number : R5F11AGJ                                 --
 4:   // -------------------------------------------------------------------
 5:   format hex
 6:   area user flash
 7:   address 0x3f400
 8:   size 122
 9:   index data
10:   000001 B39A7856341200FFA00001070009FFFFFFFFFFFFFFFFFFFFFFFF1B0201060303AAFE1316AAFE10EE02676F6F2E67
      6C2F3764694C5478000000001B0201060303AAFE1316AAFE10EE02676F6F2E676C2F3764694C5478000000001E1E0952656E
      6573617320524C37382F47314420426561636F6E204461746100
```

The sample unique code file for R5F11AGJ device describes as below.

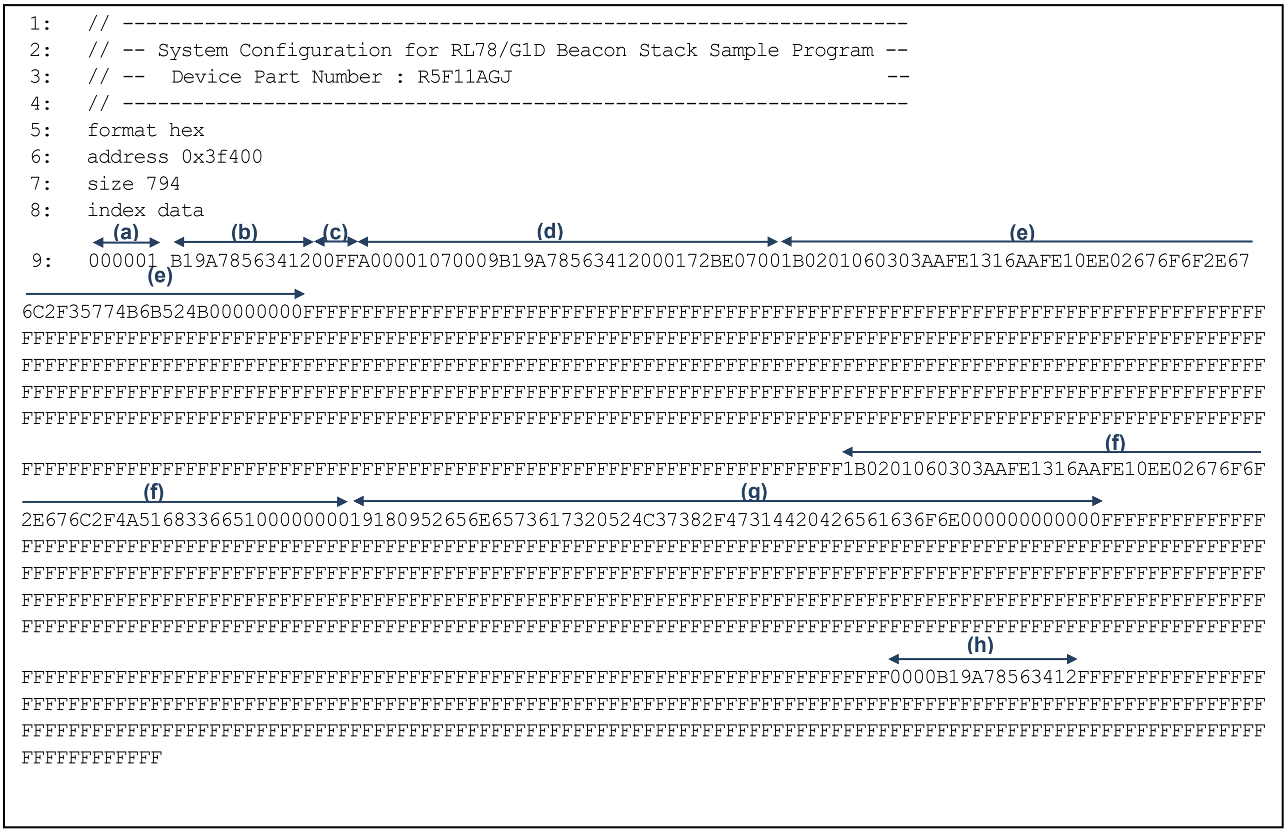
line 1-4 : The lines starting with // are comment line.
line 5 : specifies the format as hexadecimal format
line 6 : specifies the area as User area
line 7 : specifies the address as 0x3F400 (block 253)
line 8 : specifies the size 122 byte
line 9 : declares the unique code data starts at the next line
line10 : specifies the index and unique code
(a): index of unique code data
(b): device address (6byte)
(c): device address type (1byte), padding (1byte)
(d): advertising information (18byte)
(e): Non-connectable Undirected advertising data (32byte)
(f): Scannable Undirected advertising data (32byte)
(g): Scan Response data (32byte)

6.2.3 System Configuration (Beacon Application-ASCII command control)

System configuration is allocated outside of the Beacon Application-ASCII command control in the Code Flash memory. Thus, it is possible to write System configuration and firmware at the same time, by using Unique Code Embedding Function of Renesas Flash Programmer.

As a sample, the release package includes unique code file for system configuration. Regarding to the details about system configuration defined by Sample Program, refer to subsection 5.5.2 "Accessing to Code Flash memory (Beacon Appliation-ASCII command control)" in this document.

r5f11agj_syscfg(beacon_ascii).ruc, line 1-9



```
 1:   // -------------------------------------------------------------------
 2:   // -- System Configuration for RL78/G1D Beacon Stack Sample Program --
 3:   // --  Device Part Number : R5F11AGJ                                --
 4:   // -------------------------------------------------------------------
 5:   format hex
 6:   address 0x3f400
 7:   size 794
 8:   index data
 9:   000001 B19A7856341200FFA00001070009B19A78563412000172BE07001B0201060303AAFE1316AAFE10EE02676F6F2E67
6C2F35774B6B524B00000000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF1B0201060303AAFE1316AAFE10EE02676F6F
2E676C2F4A51683366510000000019180952656E6573617320524C37382F47314420426561636F6E000000000000FFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF0000B19A78563412FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFF
```

The sample unique code file for R5F11AGJ device describes as below.

line 1-4 : The lines starting with // are comment line.
line 5 : specifies the format as hexadecimal format
line 6 : specifies the address as 0x3F400 (block 253)
line 7 : specifies the size 794 byte
line 8 : declares the unique code data starts at the next line
line 9 : specifies the index and unique code

(a): index of unique code data
(b): device address (6byte)
(c): device address type (1byte), padding (1byte)
(d): advertising information (18byte)
(e): Non-connectable Undirected advertising data (32byte)
(f): Scannable Undirected advertising data (32byte)
(g): Scan Response data (32byte)
(h): White List data (8byte)

6.2.4 Advertising Configuration

The default Advertising configuration of Beacon Application and Beacon APPlication-ASCII command control are defined in r_beacon.c file and r_beacon_ascii.c file. Regarding to the specification of macros and structures, refer to chapter 4, "API" in RL78/G1D Beacon Stack User's Manual (R01UW0171).

Advertising type is set to the variable adv_type, Advertising interval, Advertising channel, and etc. are set to the structure adv_info. If need to change Advertising configuration of Beacon Application, change the setting of the variable adv_type and the structure adv_info.

r_beacon.c, line 99-120

```
 99:  /* Advertising packet type, ADV_NONCONN_IND or ADV_SCAN_IND */
100:  static uint8_t adv_type = RBLE_PDU_ADV_NONCONN_IND;
101:
102:  /* Advertising Information */
103:  static RBLE_ADV_INFO adv_info =
104:  {
105:     0x00A0,                                         /* Advertising interval         */
106:     true,                                           /* Advertising interval delay   */
107:     RBLE_ADV_ALL_CHANNELS,                          /* Advertising channel map      */
108:     0x00,                                           /* Advertising count limitation */
109:     RBLE_TXPW_LV9,                                  /* Advertising transfer power   */
110:     { 0xB1, 0x9A, 0x78, 0x56, 0x34, 0x12 },         /* Own device address           */
111:     RBLE_ADDR_PUBLIC,                               /* Own device address type      */
112:     sizeof(adv_nonconn_data) / sizeof(RBLE_ADV_DATA),  /* Advertising data count    */
113:     &adv_nonconn_data[0],                           /* Advertising data             */
114:     RBLE_EVT_PERMIT_ADV_ALL,                        /* Advertising event permission */
115:     #if WLIST_EN
116:     true                                            /* Use White List               */
117:     #else
118:     false                                           /* Use White List               */
119:     #endif
120:  };
```

Note: For the Beacon Application-ASCII command control, refer to line 191 to line 208 of the r_beacon_ascii.c file.

6.2.5 Advertising Data

The default data for Non-connectable Undirected Advertising packet of Beacon Application and Beacon Application-ASCII command control are set to the structure adv_nonconn_data.

r_beacon.c, line 54-70

```
54:  /* Advertising Data Array for ADV_NONCONN_IND */
55:  static RBLE_ADV_DATA adv_nonconn_data[] =
56:  {
57:     /* Advertising Data[0] */
58:     /* Eddystone-URL: https://goo.gl/5wKkRK -> https://www.renesas.com/ */
59:     {
60:        /* Advertising data length */
61:        27,
62:        /* Advertising data <<Flags>> */
63:        0x02, 0x01, 0x06,
64:        /* Advertising data <<Complete List of 16-bit Service Class UUIDs>> */
65:        0x03, 0x03, 0xAA, 0xFE,
66:        /* Advertising data <<Service Data>> */
67:        0x13, 0x16, 0xAA, 0xFE, 0x10, 0xEE, 0x02,
68:        'g', 'o', 'o', '.', 'g', 'l', '/', '5', 'w', 'K', 'k', 'R', 'K'
69:     },
70:  };
```

Note: For the Beacon Application-ASCII command control, refer to line 146 to line 162 of the r_beacon_ascii.c file.

The default data for Scannable Undirected Advertising packet and Scan Response packet of Beacon Application and Beacon Application-ASCII command control are set to the structure adv_scan_data.

r_beacon.c, line 72-97

```
72:   /* Advertising Data Array for ADV_SCAN_IND */
73:   static RBLE_ADV_DATA adv_scan_data[] =
74:   {
75:      /* Advertising Data[0] */
76:      /* Eddystone-URL: http://goo.gl/JQh3fQ ->
                                          https://github.com/google/eddystone/tree/master/eddystone-url */
77:      {
78:         /* Advertising data length */
79:         27,
80:         /* Advertising data <<Flags>> */
81:         0x02, 0x01, 0x06,
82:         /* Advertising data <<Complete List of 16-bit Service Class UUIDs>> */
83:         0x03, 0x03, 0xAA, 0xFE,
84:         /* Advertising data <<Service Data>> */
85:         0x13, 0x16, 0xAA, 0xFE, 0x10, 0xEE, 0x02,
86:         'g', 'o', 'o', '.', 'g', 'l', '/', 'J', 'Q', 'h', '3', 'f', 'Q'
87:      },
88:      /* Scan Response Data[0] */
89:      {
90:         /* Scan Response data length */
91:         25,
92:         /* Scan Response data <<Complete local name>> */
93:         0x18, 0x09,
94:         'R','e','n','e','s','a','s',' ','R','L','7','8','/','G','1','D',
95:         ' ','B','e','a','c','o','n'
96:      },
97:   };
```

Note: For the Beacon Application-ASCII command control, refer to line 164 to line 189 of the r_beacon_ascii.c file.

If transmitting multiple advertising data repeatedly, increase the number of RBLE_ADV_DATA structure array.

Example Code for transmitting multiple advertising data

```
/* Advertising Data Array */
static RBLE_ADV_DATA adv_data[] =
{
   /* Advertising data No.1 */
   {
      /* Advertising data length */
      … ,
      /* Advertising data */
      …
   },
   /* Advertising data No.2 */
   {
      /* Advertising data length */
      … ,
      /* Advertising data */
      …
   }
   :
};
```

6.2.6 Updating Advertising Data

Beacon Application can update Advertising data without stopping Advertising. If enabling to update Advertising data, change the macro UPDATE_EN value, which is defined in r_beacon.c file, to 1.

r_beacon.c, line 44-45

```
44:   /* Update Advertising Data (0:disable Update, 1:enable Update) */
45:   #define UPDATE_EN          (0)
```

Note: Beacon application-ASCII command control does not have this macro. Please use "Set advertising data command" to update Advertising data.

When changed the macro UPDATE_EN value to 1, Beacon Application updates Advertising data at end of each transmitting Advertising packet. By using Scan Application, it is possible to confirm that Advertising data is updated.

Result of receiving Advertising Data Updated

```
Start Scan :OK
ADV_NONCONN_IND PUBLIC 12:34:56:78:9A:B1 39ch -37dBm 30byte .!.PV.A.&.$J......RL78/G1D  00
ADV_NONCONN_IND PUBLIC 12:34:56:78:9A:B1 39ch -37dBm 30byte .!.PV.A.&.$J......RL78/G1D  01
ADV_NONCONN_IND PUBLIC 12:34:56:78:9A:B1 37ch -37dBm 30byte .!.PV.A.&.$J......RL78/G1D  02
ADV_NONCONN_IND PUBLIC 12:34:56:78:9A:B1 37ch -36dBm 30byte .!.PV.A.&.$J......RL78/G1D  03
ADV_NONCONN_IND PUBLIC 12:34:56:78:9A:B1 37ch -37dBm 30byte .!.PV.A.&.$J......RL78/G1D  04
ADV_NONCONN_IND PUBLIC 12:34:56:78:9A:B1 37ch -37dBm 30byte .!.PV.A.&.$J......RL78/G1D  05
```

6.2.7 White List configuration

Beacon Application can use White List. By using White List in transmitting Scannable Undirected Advertising packet, it is possible to filter Scan Request packet by device address. If using White List, change the macro WLIST_EN value, which is defined in r_beacon.c file, to 1.

r_beacon.c, line 47-48

```
47:   /* Enable White List for Filtering Scan Request packets */
48:   #define WLIST_EN           (0)
```

Note: Beacon application-ASCII command control does not have this macro. Please use "Set white list command" to update Advertising data.

When changed the macro WLIST_EN value to 1, defining structure wl_info is enabled, and the structure is register to Beacon Stack. By default setting, only device address of Scan Application is registered, which means that Scan Request packet from other than the device of Scan Application is not received.

r_beacon.c, line 133-139

```
133:   #if WLIST_EN
134:   /* White List for Filtering Scan Request packets */
135:   static RBLE_DEV_INFO wl_info[] =
136:   {
137:      {RBLE_ADDR_PUBLIC, 0, {0xB2, 0x9A, 0x78, 0x56, 0x34, 0x12}},
138:   };
139:   #endif
```

Note: For the Beacon Application-ASCII command control, refer to line 210 to line 214 of the r_beacon_ascii.c file.

7. Functions

This chapter describes major functions implemented in the Sample Program.

7.1 Function List

7.1.1 Beacon Application

Table 7-1 shows the functions of Beacon Application.

Table 7-1 Beacon Application Functions

file	function	description
r_beacon.c	R_BEACON_Main	initializes and executes main loop of Beacon Application
	beacon_eventhandler	Beacon Stack event handler
	beacon_input_callback	External Interrupt Input callback starts and stops Advertising

7.1.2 Scan Application

Table 7-2 shows the functions of Scan Application.

Table 7-2 Scan Application Functions

file	function	description
r_scan_ascii.c / r_scan_bin.c	R_SCAN_ASCII_Main / R_SCAN_BINARY_Main	initializes and executes main loop of Scan Application
	scan_eventhandler	Beacon Stack event handler
	uart_rx_complete	UART received interrupt callback parses UART commands, starts and stops Scanning
	uart_tx_complete	UART transmitted interrupt callback confirms transmitting event packet
	uart_rx_error	UART Rx Error Interrupt callback restarts receiving UART commands
r_scan_ascii.c	scan_input_callback	External Interrupt Input callback changes display format of received Advertising packet

7.1.3 DTM Application

Table 7-3 shows the functions of DTM Application.

Table 7-3 DTM Application Functions

file	function	description
r_dtm.c	R_DTM_Main	initializes and executes main loop of DTM Application
	dtm_eventhandler	Beacon Stack event handler
	dtm_rx_complete	UART received interrupt callback parses RF Test commands, starts and stops DTM
	dtm_tx_complete	UART transmitted interrupt callback confirms transmitting RF Test event packet
	dtm_rx_error	UART Rx Error Interrupt callback restarts receiving RF Test commands

7.1.4 Beacon Application-ASCII command control

Table 7-4 shows the functions of Beacon Application-ASCII command control.

Table 7-4 Beacon Application-ASCII command control Functions

file	function	description
r_beacon_ascii.c	R_BEACON_ASCII_Main	Initializes and executes main loop of Beacon Application-ASCII command control
	beacon_eventhandler	Beacon Stack event handler
	beacon_read_advcfg	Read system configuration
	beacon_send_<***>Note	Display command processing results to terminal software
	uart_cmdfunc	Identification of the command entered
	uart_cmd_<***>Note	Command processing

Note: Please refer to r_beacon_ascii.c for each function represented by <***>.

8. Operation

8.1 State Transition

The Sample Program consists of three applications: Beacon Application, Scan Application, DTM Application and Beacon Application-ASCII command control. This section describes state transition of each application.

8.1.1 Beacon Application

Figure 8-1 shows the state transition of Beacon Application.

It starts with Initializing state and follows by Advertising state. In the Advertising state, Beacon Application transmits Non-connectable Undirected Advertising packets.

By pushing switch SW2, Beacon Application stops Advertising and goes to Idling state as path 1 in below figure. By pushing switch SW2, Beacon Application transmits Scannable Undirected Advertising packet and goes to Advertising state as path 2 in below figure. By pushing switch SW2, Beacon Application stops Advertising and goes to Idling state as path 3 in below figure. By pushing switch SW2, Beacon Application transmits Non-connectable Undirected Advertising packet again as path 4 in below path.

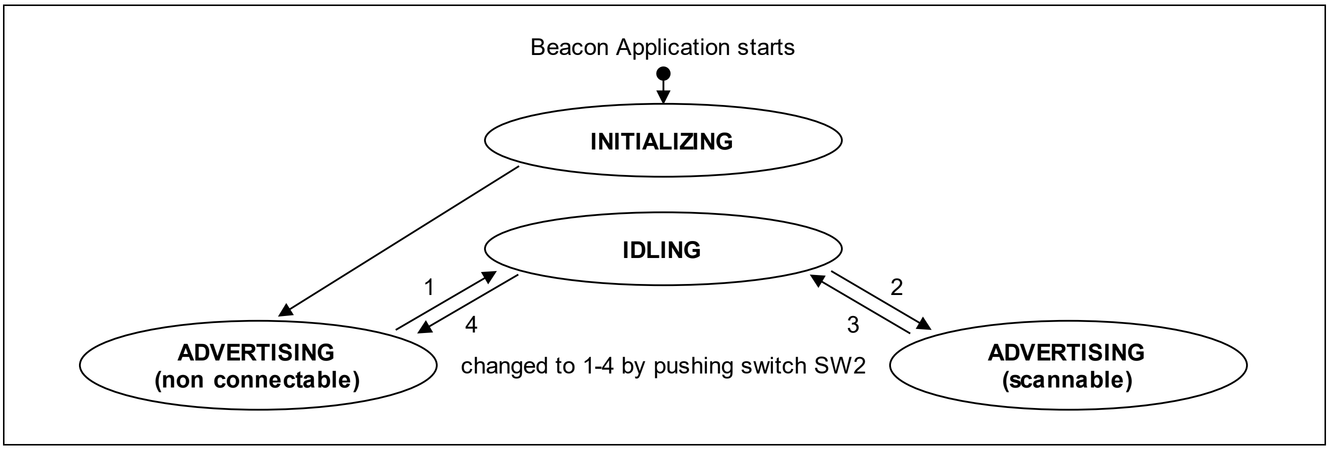

Beacon Application starts
INITIALIZING
IDLING
1
2
4
3
ADVERTISING (non connectable)
changed to 1-4 by pushing switch SW2
ADVERTISING (scannable)

Figure 8-1 State Transition of Beacon Application

8.1.2 Scan Application

Figure 8-2 shows the state transition of Scan Application.

It starts with Initializing state and follows by Scanning state when use ASCII-format UART communication, or Idling state when use Binary-format UART communication.

In the Scanning state, Scan Application starts Scanning. When receive Stop Scan command through UART, Scan Application stops Scanning and goes to Idling state.

In the Idling state, by receiving configuration change command through UART, Scan Application changes Scanning configuration. By receiving Start Scan command through UART, Scan Application restarts Scanning and goes to Scanning state again.

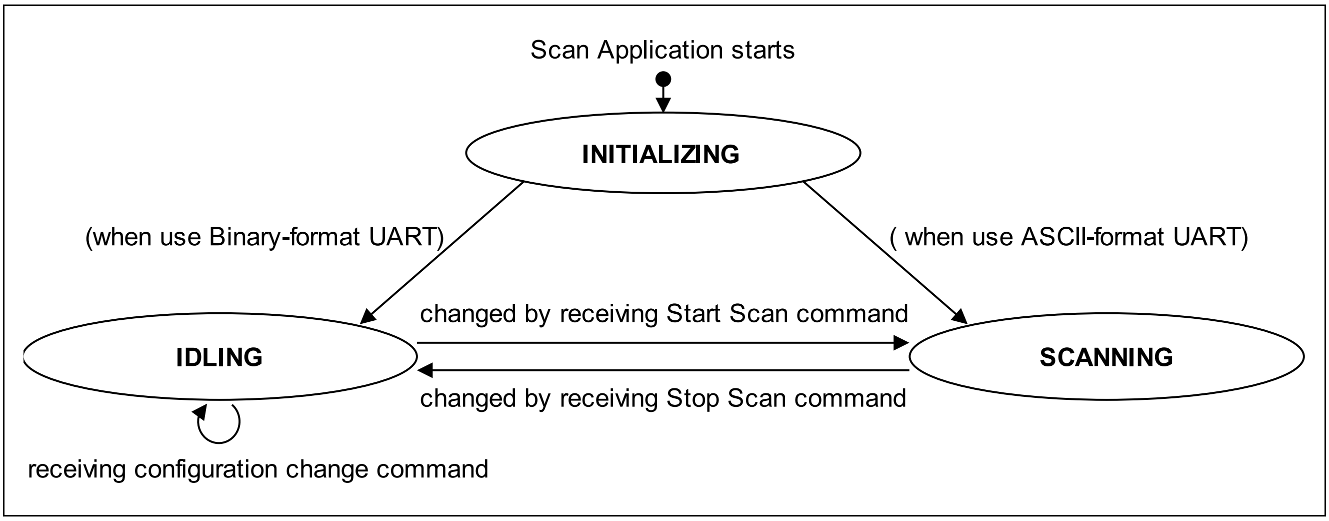

Scan Application starts
INITIALIZING
(when use Binary-format UART)
(when use ASCII-format UART)
changed by receiving Start Scan command
IDLING
SCANNING
changed by receiving Stop Scan command
receiving configuration change command

Figure 8-2 State Transition of Scan Application

8.1.3 DTM Application

Figure 8-3 shows the state transition of DTM Application.

It starts with Initializing state and follows by Idling state.

In the Idling state, DTM Application accepts LE_RESET, LE_TRANSMITTER_TEST and LE_RECEIVER_TEST commands. By receiving LE_TRANSMITTER_TEST command through UART, DTM Application starts RF Transmitter Test and goes to Transmitter Test state. By receiving LE_RECEIVER_TEST command through UART, DTM Application starts RF Receiver Test and goes to Receiver Test state. By receiving LE_TEST_END command through UART, DTM Application stops RF Test and goes to Idling state.

If receive LE_RESET command through UART in Transmitter Test state or Receiver Test, DTM Application stops RF Test and goes to Idling state.

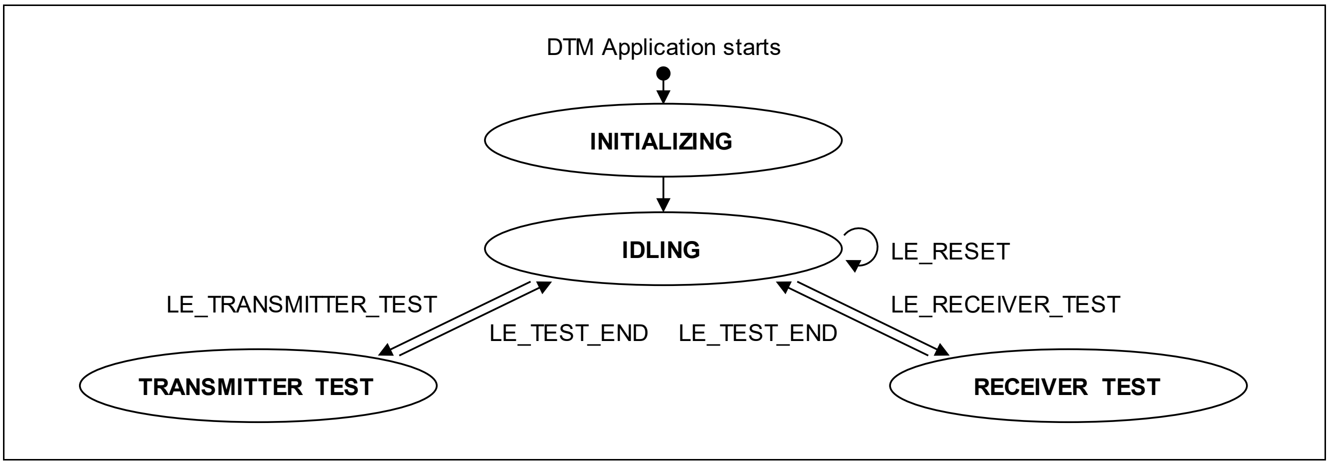


Figure 8-3 State Transition of DTM Application

8.1.4 Beacon Application-ASCII command control

Figure 8-4 shows the state transition of Beacon Application.

Beacon application starts operation in Initializing state and goes to Idling state. Then wait for ASCII command input.

When the "adstart n" command is input from the host machine, it goes to Advertising state in which Non-connectable Undirected Advertising packets are transmitted (4 in the figure), and when the "adstop" command is input, it goes to Idling state (1 in the figure).

When the"adstart s" command is input from the host machine, it goes to the Advertising state in which Scannable Undirected Advertising packets are transmitted (2 in the figure), and when the "adstop" command is input, it goes to the Idling state (3 in the figure).

When the "pwdn" command is input from the host machine,it goes to the RF Power down state (5 in the figure), and when the "pwup" command is input, it goes to the Idling state (6 in the figure).

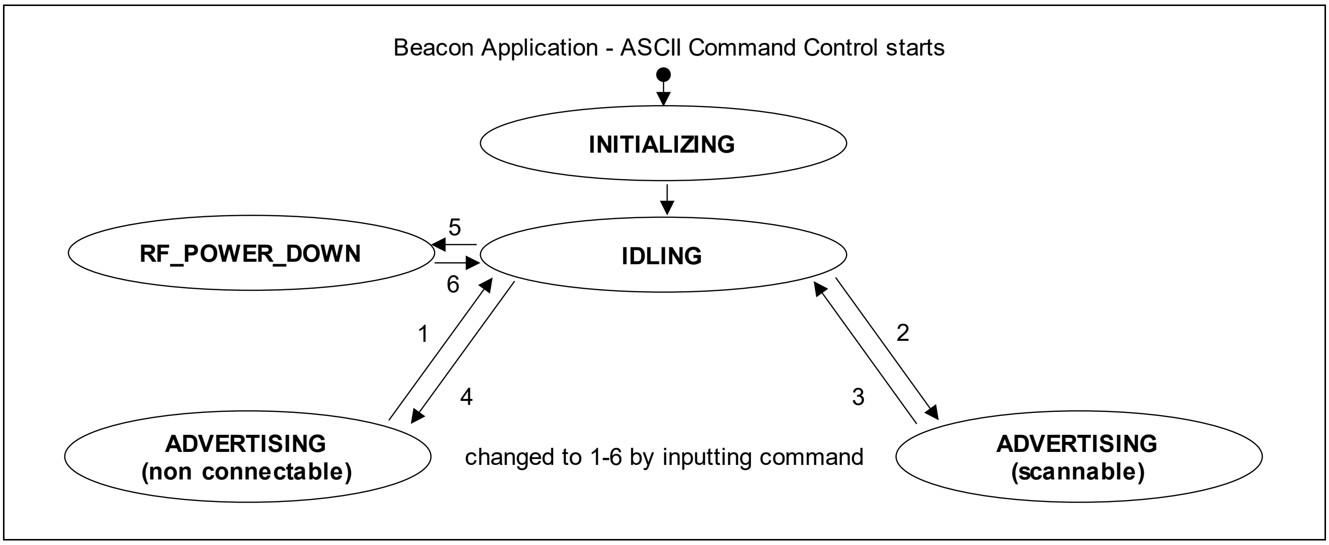

Beacon Application - ASCII Command Control starts
INITIALIZING
RF_POWER_DOWN
5
IDLING
6
1
2
4
3
ADVERTISING (non connectable)
changed to 1-6 by inputting command
ADVERTISING (scannable)

Figure 8-4 State Transition of Beacon Application-ASCII command control

8.2 Sequence

This section describes sequence of RF Initialization, Beacon Application, Scan Application, and DTM Application. The Sample Program uses Beacon Stack API.

Regarding to the specification of Beacon Stack API, refer to chapter 4 "API" in RL78/G1D Beacon Stack User's Manual (R01UW017).

8.2.1 RF Initialization

Figure 8-5 shows RF Initialization Sequence.

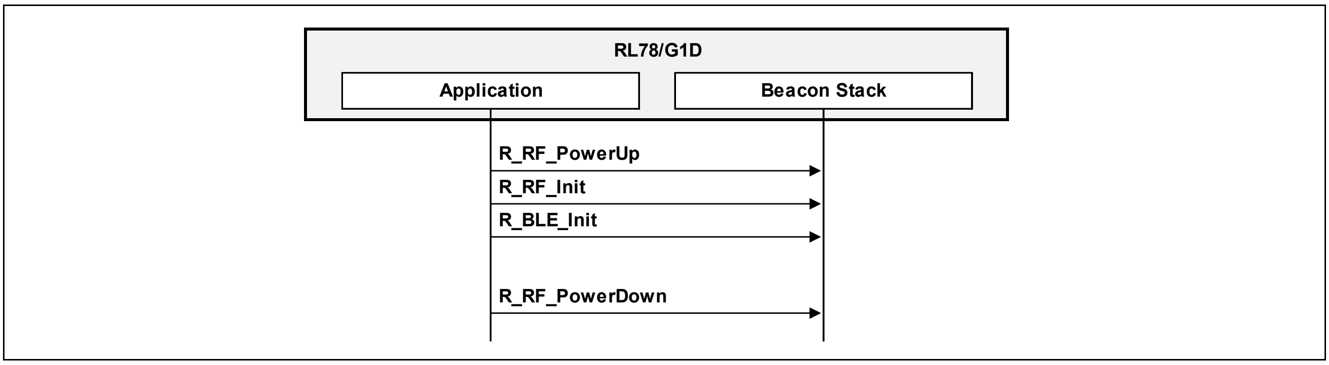


Figure 8-5 RF Initialization Sequence

8.2.2 Beacon Application

Figure 8-6 shows Beacon Application Sequence.

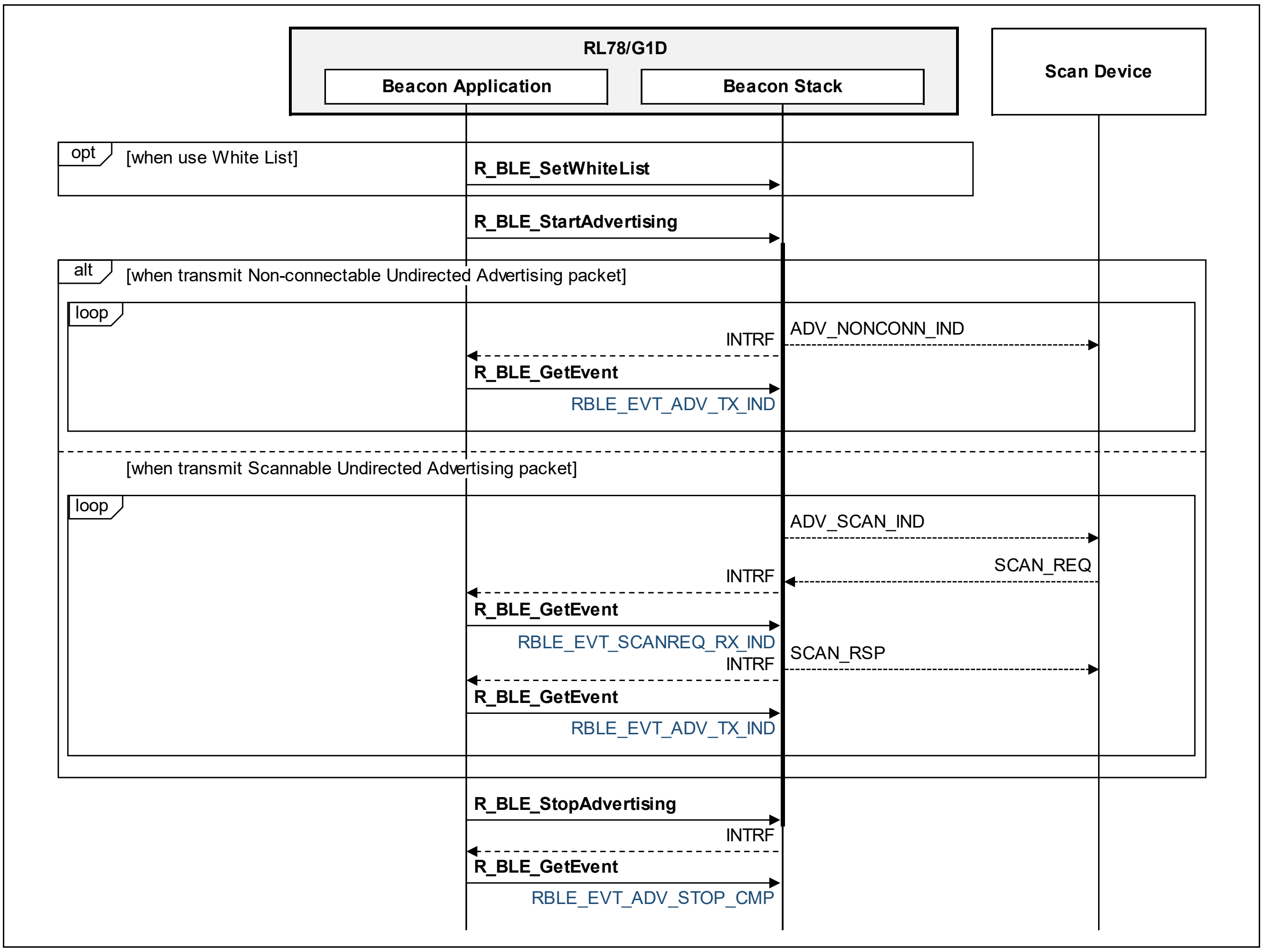


Figure 8-6 Beacon Application Sequence

8.2.3 Scan Application

Figure 8-7 shows Scan Application Sequence.

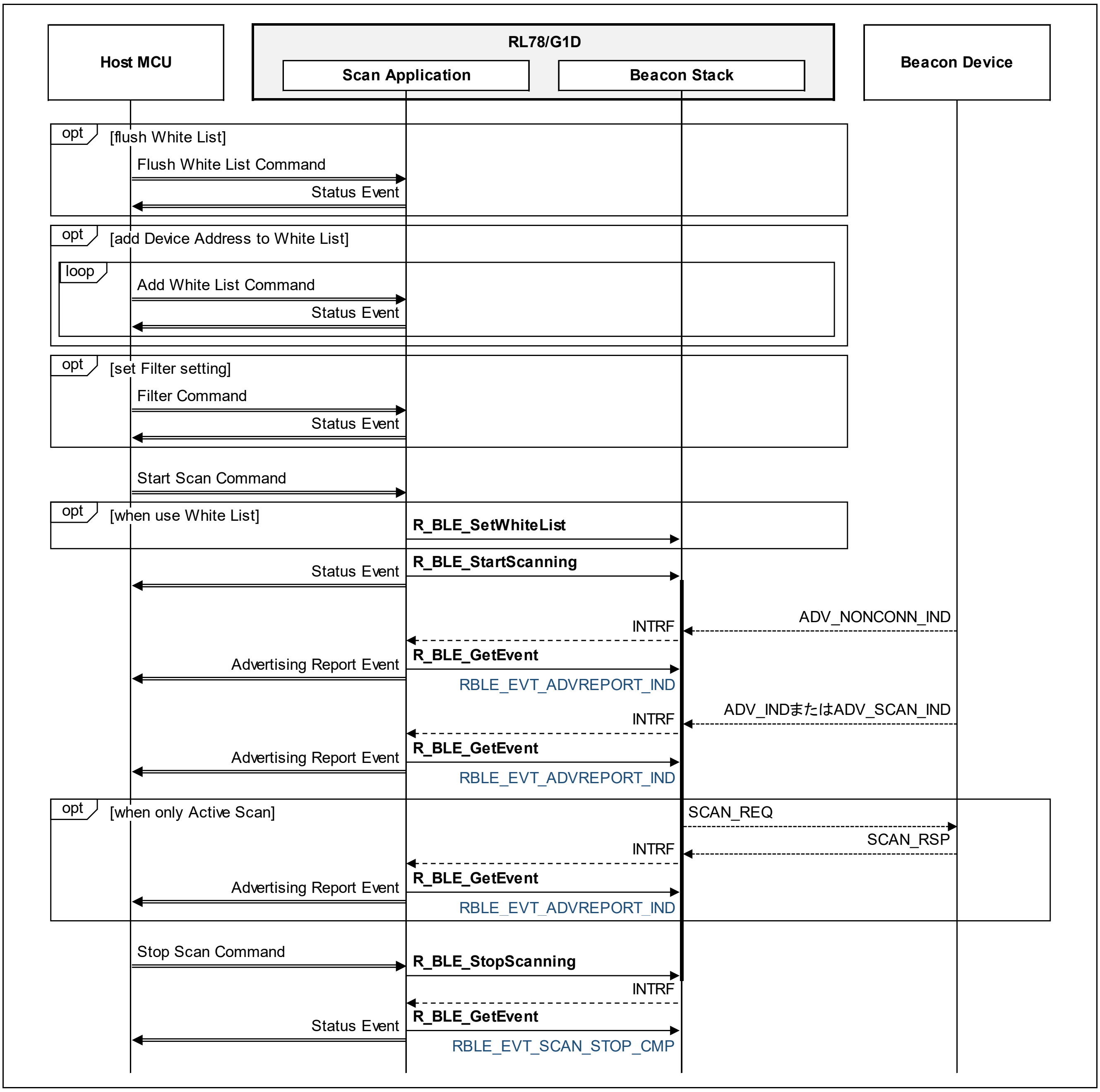


Figure 8-7 Scan Application Sequence

8.2.4 DTM Application

Figure 8-8 shows DTM Application sequence.

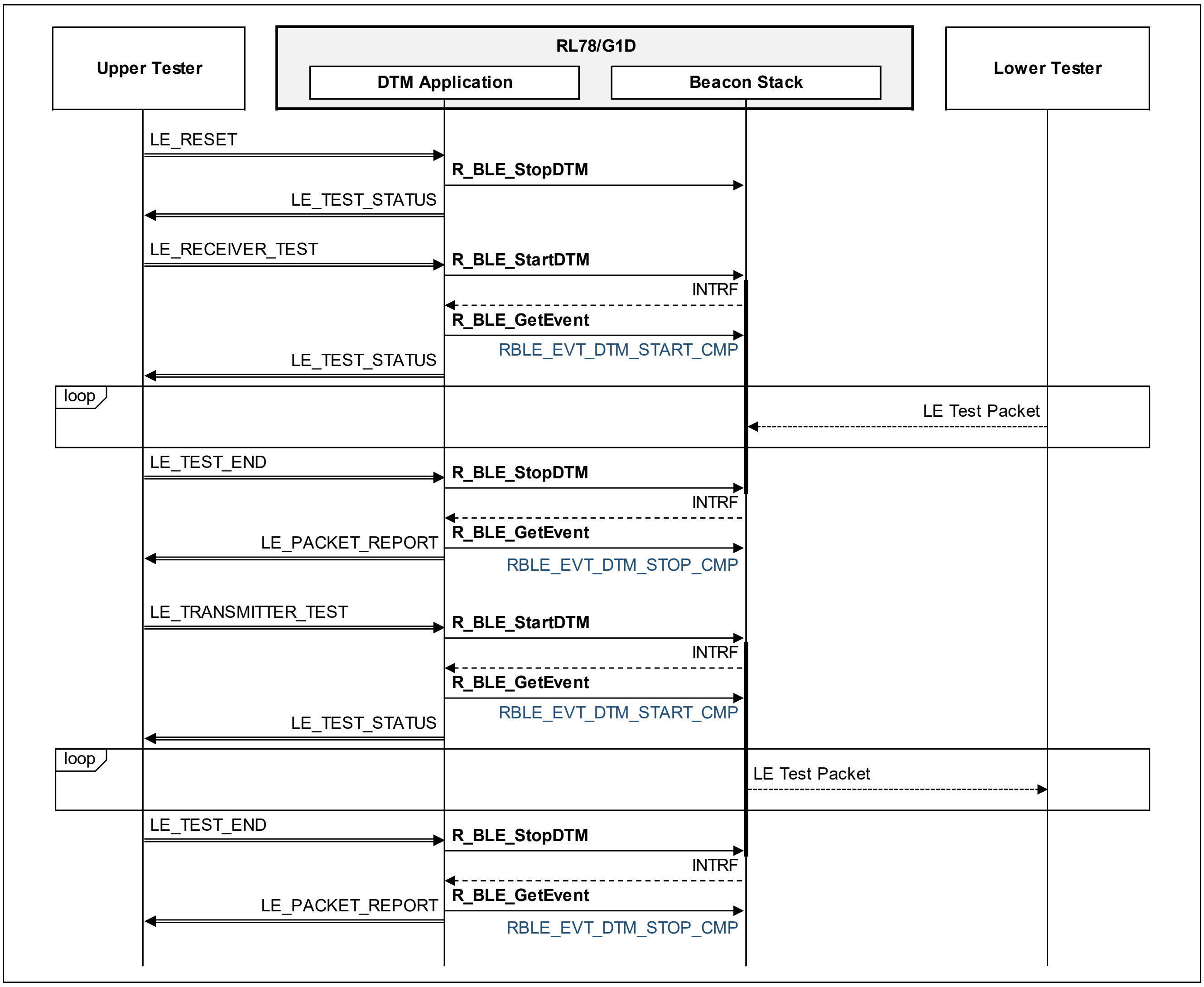


Figure 8-8 DTM Application Sequence

8.2.5 Beacon Application-ASCII command control

The Beacon Application-ASCII command control mainly uses the following API. Regarding to the specification of Beacon Stack API, refer to chapter 4 "API" in RL78/G1D Beacon Stack User's Manual (R01UW0171).

- R_BLE_StartAdvertising
- R_BLE_StopAdvertising
- R_BLE_UpdateAdvInfo
- R_BLE_UpdateAdvData
- R_BLE_SetWhiteList
- R_BLE_GetEvent

9. Appendix

9.1 Current Consumption of Beacon Stack

Regarding to current consumption in starting-up and periodic advertising packet transmission of Beacon Stack, the measurement conditions and the measurement results in our environment are shown.

9.1.1 Notice

The results of current consumption measurement shown in this section is NOT guaranteed performance but for reference only. It is recommended to measure by oneself with the conditions of actual use case.

9.1.2 Measurement Environment

Table 9-1 shows the equipments for current consumption measurement. Regarding to the slide switch settings of the evaluation board, refer to subsection 4.5.2 "Evaluation Board Setting" in this document. Regarding to the measurement procedure, refer to subsection 4.5.3 "Measurement Procedure" in this document.

Table 9-1 Equipments for Current Consumption Measurement

Category	Condition
Target Device	RL78/G1D(R5F11AGJ)
Target Board	RL78/G1D Evaluation Board(RTK0EN0001D01001BZ)
Current Measurement Equipment	Keysight Technologies DC Power Analyzer Mainframe(N6705B) 2-Quadrant Source/Measurement Unit for Battery Drain Analysis(N6781A)

Figure 9-1 shows the current consumption measurement environment.

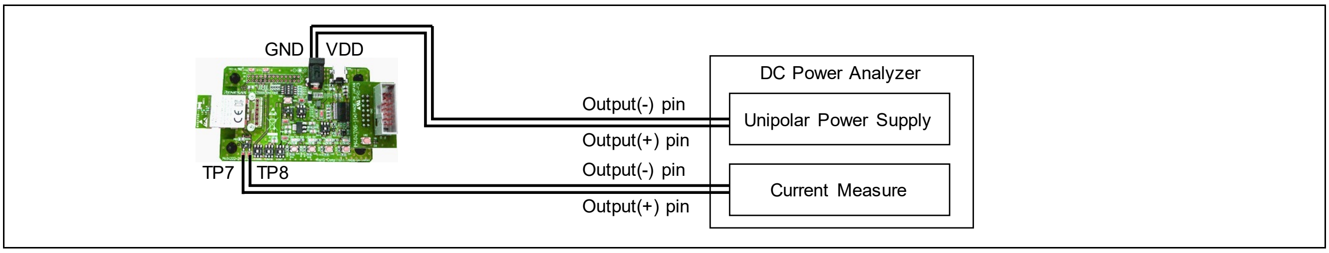


Figure 9-1 Current Consumption Measurement Environment

9.1.3 Measurement Results of Starting-up Beacon Stack

(1) MCU Clock Frequency 4MHz

Table 9-2 shows the operation settings for measuring starting-up current waveform at MCU clock frequency 4MHz.

Table 9-2 Operation Settings for Measuring Current Waveform of Starting-up Beacon Stack

Category	Condition
Compiler	CC-RL V1.04.00
MCU Clock Frequency	4MHz
Supply Voltage	3V
DC-DC Converter	not use RF on-chip DC-DC converter
RF Slow Clock Source	use RF on-chip oscillator, execute calibration
RF Operation	enable only Tx and Rx (RFCFG_TX)
Advertising Transmit Power	0dBm
Advertising Channels	3ch channels (37,38,39ch)
Advertising Type	ADV_NONCONN_IND
Advertising Data Length	31byte
Advertising Data Count	1
Advertising Interval	100msec

Current waveforms of starting-up Beacon Stack in the case of MCU clock frequency 4MHz is shown in **Figure 9-2**.

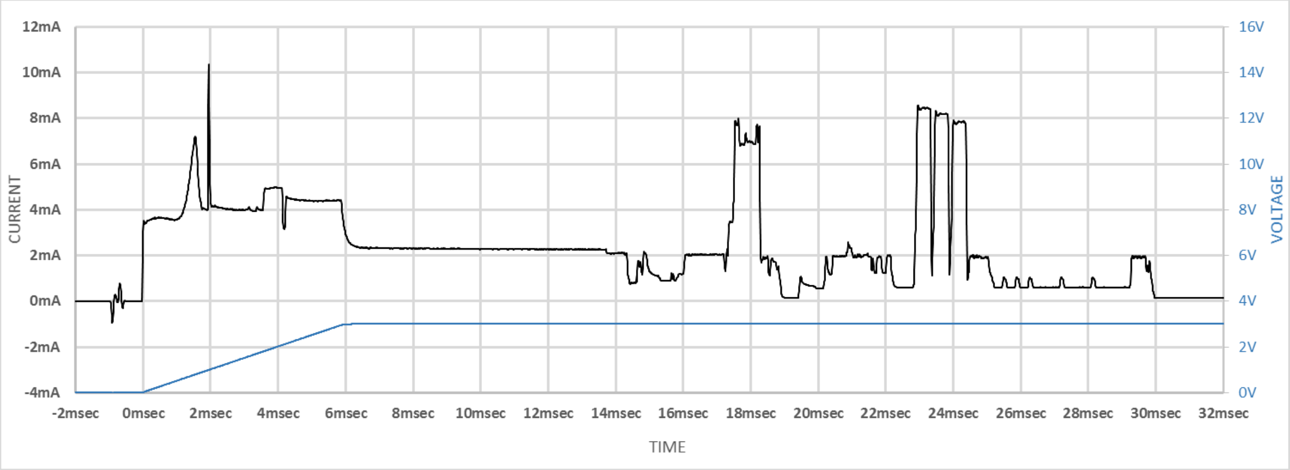


Figure 9-2 Current Waveform of Starting-up Beacon Stack at MCU Clock Frequency 4MHz

(2) MCU Clock Frequency 8MHz

Table 9-3 shows the operation settings for measuring starting-up current waveform at MCU clock frequency 8MHz.

Table 9-3 Operation Settings for Measuring Current Waveform of Starting-up Beacon Stack

Category	Condition
Compiler	CC-RL V1.04.00
MCU Clock Frequency	8MHz
Supply Voltage	3V
DC-DC Converter	not use RF on-chip DC-DC converter
RF Slow Clock Source	use RF on-chip oscillator, execute calibration
RF Operation	enable only Tx and Rx (RFCFG_TX)
Advertising Transmit Power	0dBm
Advertising Channels	3ch channels (37,38,39ch)
Advertising Type	ADV_NONCONN_IND
Advertising Data Length	31byte
Advertising Data Count	1
Advertising Interval	100msec

Current waveforms of starting-up Beacon Stack in the case of MCU clock frequency 8MHz is shown in **Figure 9-3**.

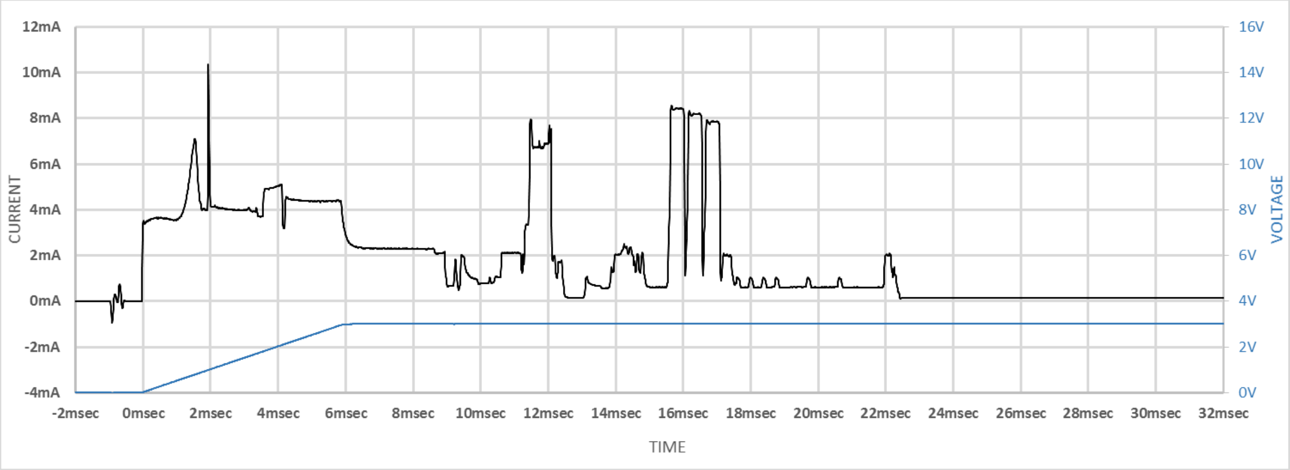


Figure 9-3 Current Waveform of Starting-up Beacon Stack at MCU Clock Frequency 8MHz

(3) MCU Clock Frequency 16MHz

Table 9-4 shows the operation settings for measuring starting-up current waveform at MCU clock frequency 16MHz.

Table 9-4 Operation Settings for Measuring Current Waveform of Starting-up Beacon Stack

Category	Condition
Compiler	CC-RL V1.04.00
MCU Clock Frequency	16MHz
Supply Voltage	3V
DC-DC Converter	not use RF on-chip DC-DC converter
RF Slow Clock Source	use RF on-chip oscillator, execute calibration
RF Operation	enable only Tx and Rx (RFCFG_TX)
Advertising Transmit Power	0dBm
Advertising Channels	3ch channels (37,38,39ch)
Advertising Type	ADV_NONCONN_IND
Advertising Data Length	31byte
Advertising Data Count	1
Advertising Interval	100msec

Current waveforms of starting-up Beacon Stack in the case of MCU clock frequency 16MHz is shown in **Figure 9-4**.

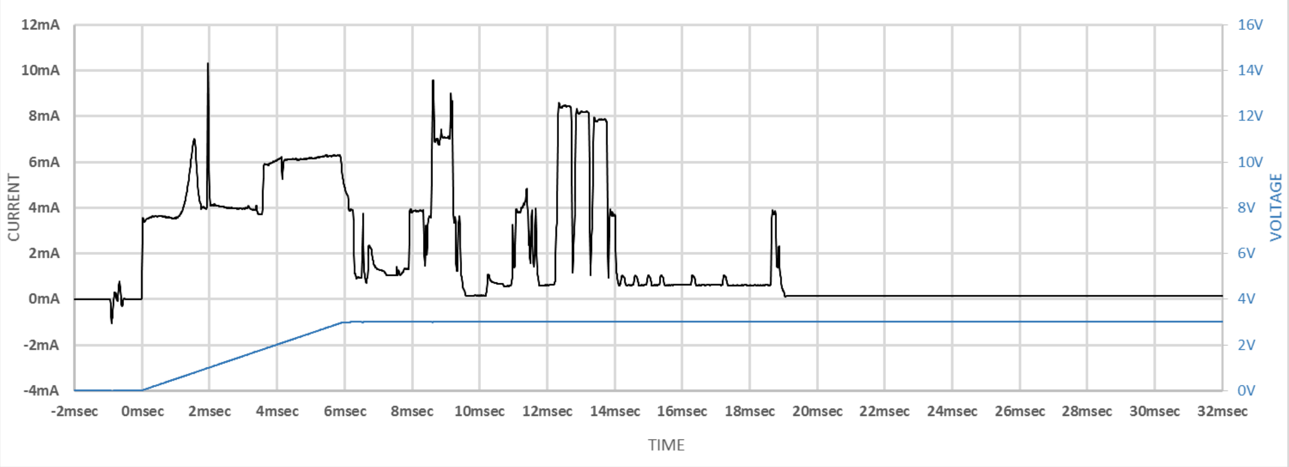


Figure 9-4 Current Waveform of Starting-up Beacon Stack at MCU Clock Frequency 16MHz

(4) MCU Clock Frequency 32MHz

Table 9-5 shows the operation settings for measuring starting-up current waveform at MCU clock frequency 32MHz.

Table 9-5 Operation Settings for Measuring Current Waveform of Starting-up Beacon Stack

Category	Condition
Compiler	CC-RL V1.04.00
MCU Clock Frequency	32MHz
Supply Voltage	3V
DC-DC Converter	not use RF on-chip DC-DC converter
RF Slow Clock Source	use RF on-chip oscillator, execute calibration
RF Operation	enable only Tx and Rx (RFCFG_TX)
Advertising Transmit Power	0dBm
Advertising Channels	3ch channels (37,38,39ch)
Advertising Type	ADV_NONCONN_IND
Advertising Data Length	31byte
Advertising Data Count	1
Advertising Interval	100msec

Current waveforms of starting-up Beacon Stack in the case of MCU clock frequency 32MHz is shown in **Figure 9-5**.

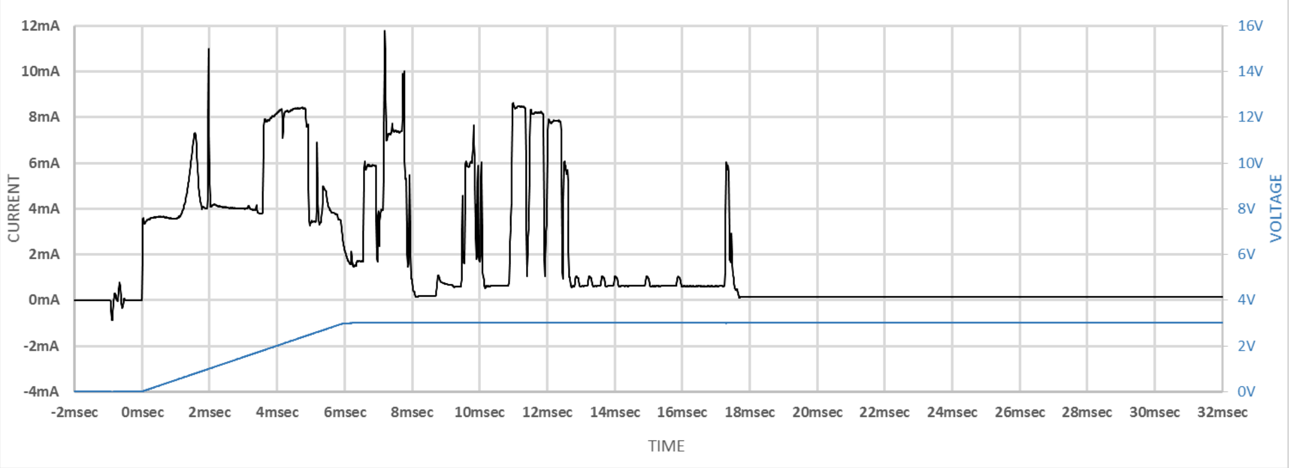


Figure 9-5 Current Waveform of Starting-up Beacon Stack at MCU Clock Frequency 32MHz

9.1.4 Measurement Results of Periodic Advertising packet transmission

(1) MCU Clock Frequency 4MHz

Table 9-6 shows the operation settings for measuring average current consumption at MCU clock frequency 4MHz.

Table 9-6 Operation Settings for Measuring Average Current Consumption

Category	Condition
Compiler	CC-RL V1.04.00
MCU Clock Frequency	4MHz
Supply Voltage	3V
DC-DC Converter	use RF on-chip DC-DC converter / not use RF on-chip DC-DC converter
RF Slow Clock Source	use RF on-chip oscillator, execute calibration
RF Operation	enable both Tx and Rx (RFCFG_TXRX)
Advertising Transmit Power	0dBm
Advertising Channels	3ch channels (37,38,39ch)
Advertising Type	ADV_NONCONN_IND / ADV_SCAN_IND
Advertising Data Length	31byte
Advertising Data Count	1
Advertising Interval	10msec / 100msec / 1sec Note: Interval range defined by Bluetooth Core Specification is 100msec to 10.24sec.

Measurement results of average current consumption at MCU clock frequency 4MHz are shown in **Table 9-7** and **Table 9-8**.

Table 9-7 Measurement result at MCU Clock Frequency 4MHz (when transmit ADV_NONCONN_IND)

Advertising Interval	Measurement Result of Average Current Consumption (transmit ADV_NONCONN_IND)	
	not use DC-DC converter	use DC-DC converter
10msec	1540.8uA	1041.1uA
100msec	172.2uA	126.4uA
1sec	18.8uA	14.2uA

Table 9-8 Measurement result at MCU Clock Frequency 4MHz (when transmit ADV_SCAN_IND)

Advertising Interval	Measurement Result of Average Current Consumption (transmit ADV_SCAN_IND, receive no SCAN_REQ)	
	not use DC-DC converter	use DC-DC converter
10msec	1806.1uA	1211.8uA
100msec	217.3uA	160.3uA
1sec	23.2uA	17.6uA

Current consumption waveforms of transmitting ADV_NONCONN_IND packet at Advertising interval 1sec, MCU clock frequency 4MHz are shown in **Figure 9-6** and **Figure 9-7**.

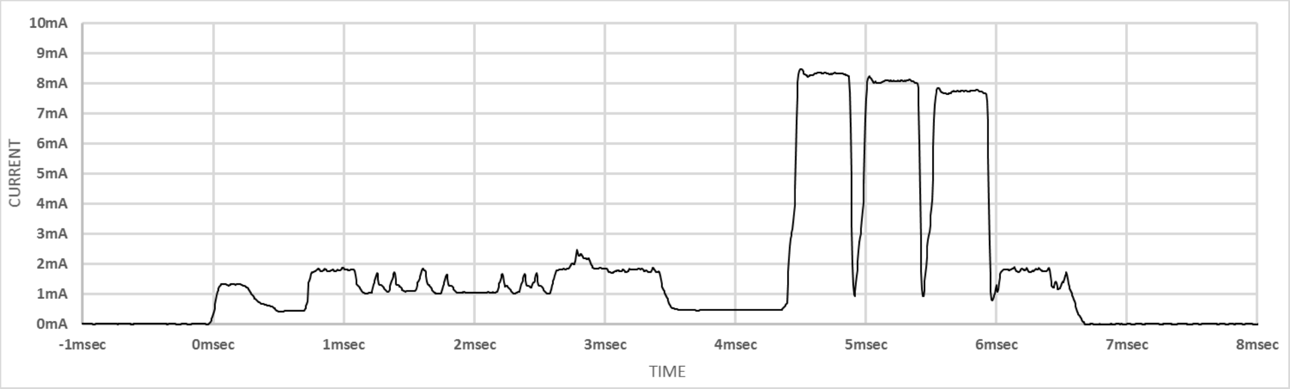


Figure 9-6 Current Consumption Waveform of ADV_NONCONN_IND at MCU Clock Frequency 4MHz (not use DC-DC converter)

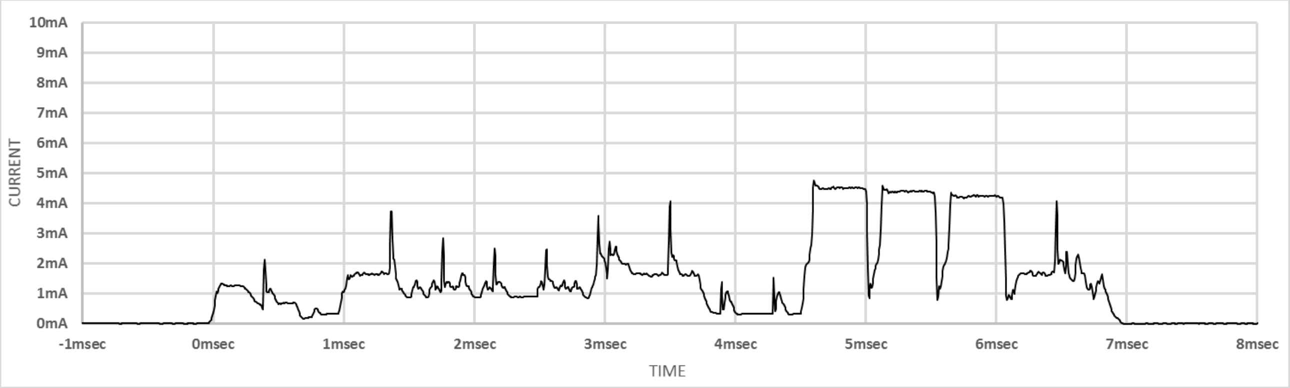


Figure 9-7 Current Consumption Waveform of ADV_NONCONN_IND at MCU Clock Frequency 4MHz (use DC-DC converter)

Current consumption waveforms of transmitting ADV_SCAN_IND packet and receiving no SCAN_REQ packet at Advertising interval 1sec, MCU clock frequency 4MHz are shown in **Figure 9-8** and **Figure 9-9**.

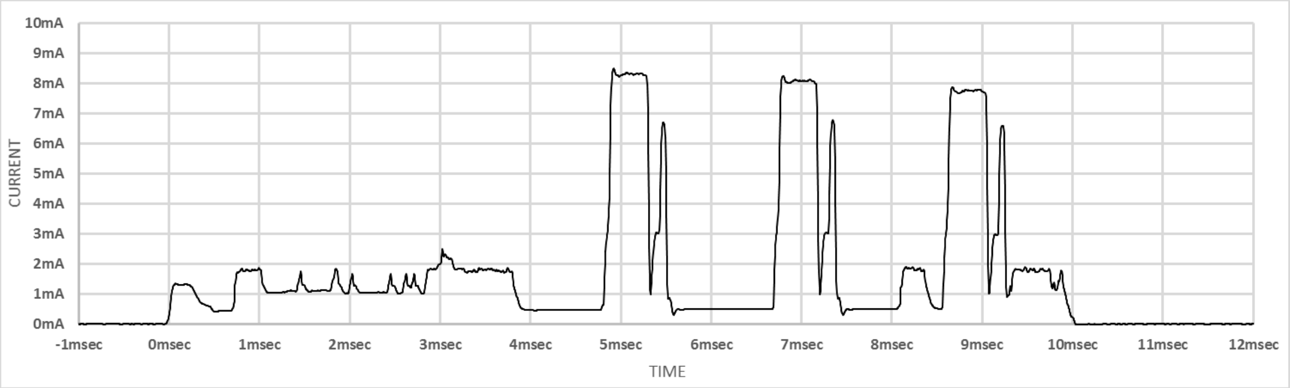


Figure 9-8 Current Consumption Waveform of ADV_SCAN_IND at MCU Clock Frequency 4MHz (not use DC-DC converter)

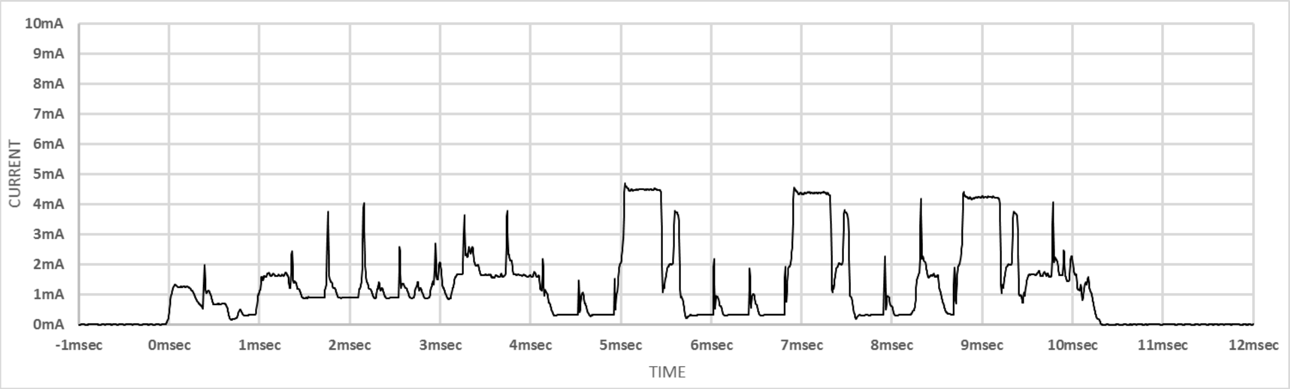


Figure 9-9 Current Consumption Waveform of ADV_SCAN_IND at MCU Clock Frequency 4MHz (use DC-DC converter)

(2) MCU Clock Frequency 8MHz

Table 9-9 shows the operation settings for measuring average current consumption at MCU clock frequency 8MHz.

Table 9-9 Operation Settings for Measuring Average Current Consumption

Category	Condition
Compiler	CC-RL V1.04.00
MCU Clock Frequency	8MHz
Supply Voltage	3V
DC-DC Converter	use RF on-chip DC-DC converter / not use RF on-chip DC-DC converter
RF Slow Clock Source	use RF on-chip oscillator, execute calibration
RF Operation	enable both Tx and Rx (RFCFG_TXRX)
Advertising Transmit Power	0dBm
Advertising Channels	3ch channels (37,38,39ch)
Advertising Type	ADV_NONCONN_IND / ADV_SCAN_IND
Advertising Data Length	31byte
Advertising Data Count	1
Advertising Interval	10msec / 100msec / 1sec Note: Interval range defined by Bluetooth Core Specification is 100msec to 10.24sec.

Measurement results of average current consumption at MCU clock frequency 8MHz are shown in **Table 9-10** and **Table 9-11**.

Table 9-10 Measurement result at MCU Clock Frequency 8MHz (when transmit ADV_NONCONN_IND)

Advertising Interval	Measurement Result of Average Current Consumption (transmit ADV_NONCONN_IND)	
	not use DC-DC converter	use DC-DC converter
10msec	1518.8uA	1015.9uA
100msec	148.5uA	101.8uA
1sec	16.5uA	11.8uA

Table 9-11 Measurement result at MCU Clock Frequency 8MHz (when transmit ADV_SCAN_IND)

Advertising Interval	Measurement Result of Average Current Consumption (transmit ADV_SCAN_IND, receive no SCAN_REQ)	
	not use DC-DC converter	use DC-DC converter
10msec	1751.7uA	1178.5uA
100msec	187.5uA	132.7uA
1sec	20.4uA	14.7uA

Current consumption waveforms of transmitting ADV_NONCONN_IND packet at Advertising interval 1sec, MCU clock frequency 8MHz are shown in **Figure 9-10** and **Figure 9-11**.

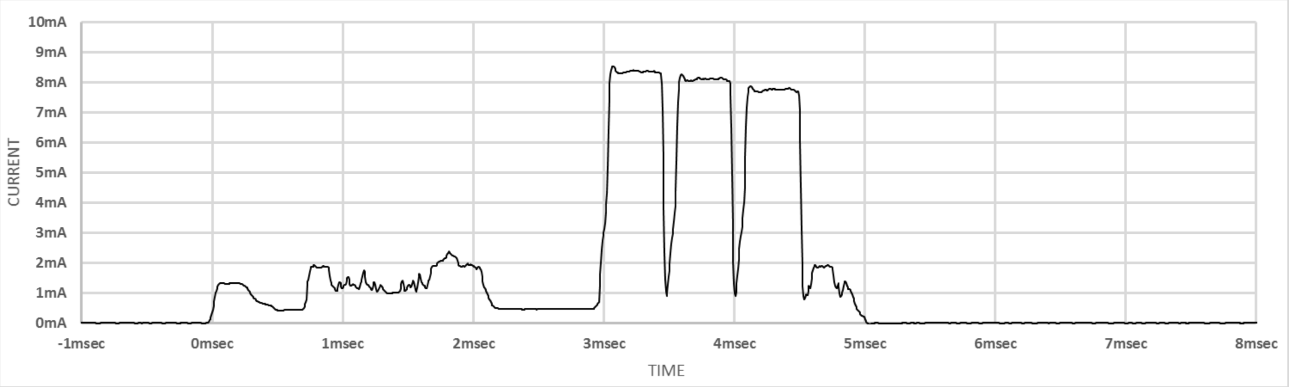


Figure 9-10 Current Consumption Waveform of ADV_NONCONN_IND at MCU Clock Frequency 8MHz (not use DC-DC converter)

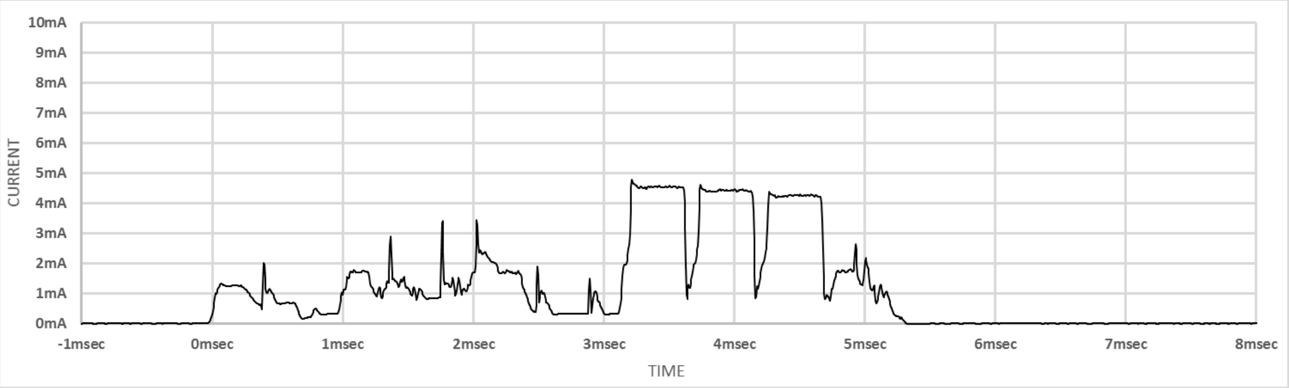


Figure 9-11 Current Consumption Waveform of ADV_NONCONN_IND at MCU Clock Frequency 8MHz (use DC-DC converter)

Current consumption waveforms of transmitting ADV_SCAN_IND packet and receiving no SCAN_REQ packet at Advertising interval 1sec, MCU clock frequency 8MHz are shown in **Figure 9-12** and **Figure 9-13**.

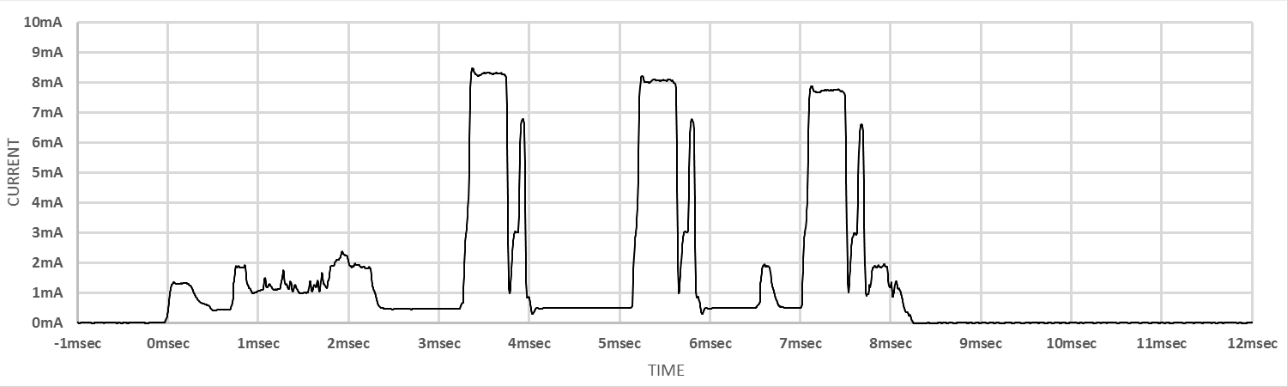


Figure 9-12 Current Consumption Waveform of ADV_SCAN_IND at MCU Clock Frequency 8MHz (not use DC-DC converter)

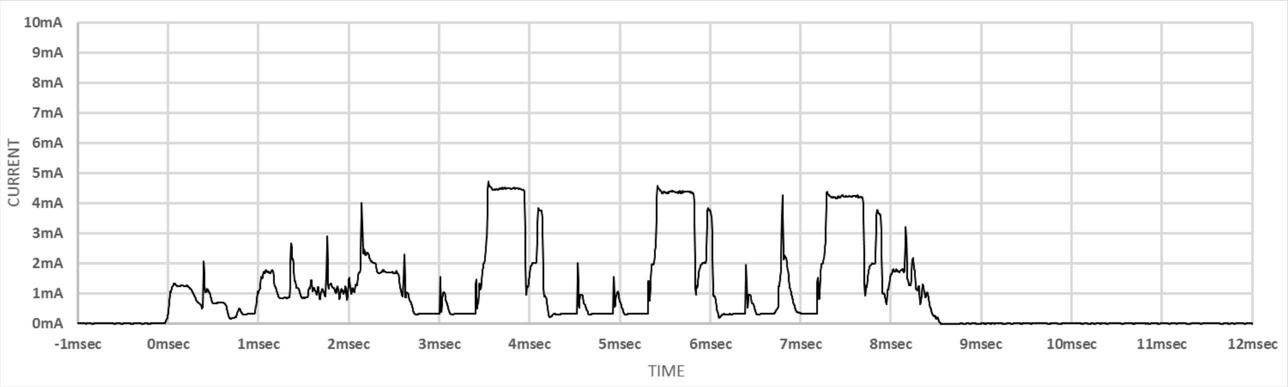


Figure 9-13 Current Consumption Waveform of ADV_SCAN_IND at MCU Clock Frequency 8MHz (use DC-DC converter)

(3) MCU Clock Frequency 16MHz

Table 9-12 shows the operation settings for measuring average current consumption at MCU clock frequency 16MHz.

Table 9-12 Operation Settings for Measuring Average Current Consumption

Category	Condition
Compiler	CC-RL V1.04.00
MCU Clock Frequency	16MHz
Supply Voltage	3V
DC-DC Converter	use RF on-chip DC-DC converter / not use RF on-chip DC-DC converter
RF Slow Clock Source	use RF on-chip oscillator, execute calibration
RF Operation	enable both Tx and Rx (RFCFG_TXRX)
Advertising Transmit Power	0dBm
Advertising Channels	3ch channels (37,38,39ch)
Advertising Type	ADV_NONCONN_IND / ADV_SCAN_IND
Advertising Data Length	31byte
Advertising Data Count	1
Advertising Interval	10msec / 100msec / 1sec Note: Interval range defined by Bluetooth Core Specification is 100msec to 10.24sec.

Measurement results of average current consumption at MCU clock frequency 16MHz are shown in **Table 9-13** and **Table 9-14**.

Table 9-13 Measurement result at MCU Clock Frequency 16MHz (when transmit ADV_NONCONN_IND)

Advertising Interval	Measurement Result of Average Current Consumption (transmit ADV_NONCONN_IND)	
	not use DC-DC converter	use DC-DC converter
10msec	1536.7uA	1031.3uA
100msec	157.9uA	111.5uA
1sec	17.4uA	12.8uA

Table 9-14 Measurement result at MCU Clock Frequency 16MHz (when transmit ADV_SCAN_IND)

Advertising Interval	Measurement Result of Average Current Consumption (transmit ADV_SCAN_IND, receive no SCAN_REQ)	
	not use DC-DC converter	use DC-DC converter
10msec	1786.0uA	1191.9uA
100msec	199.2uA	143.2uA
1sec	21.6uA	16.8uA

Current consumption waveforms of transmitting ADV_NONCONN_IND packet at Advertising interval 1sec, MCU clock frequency 16MHz are shown in **Figure 9-14** and **Figure 9-15**.

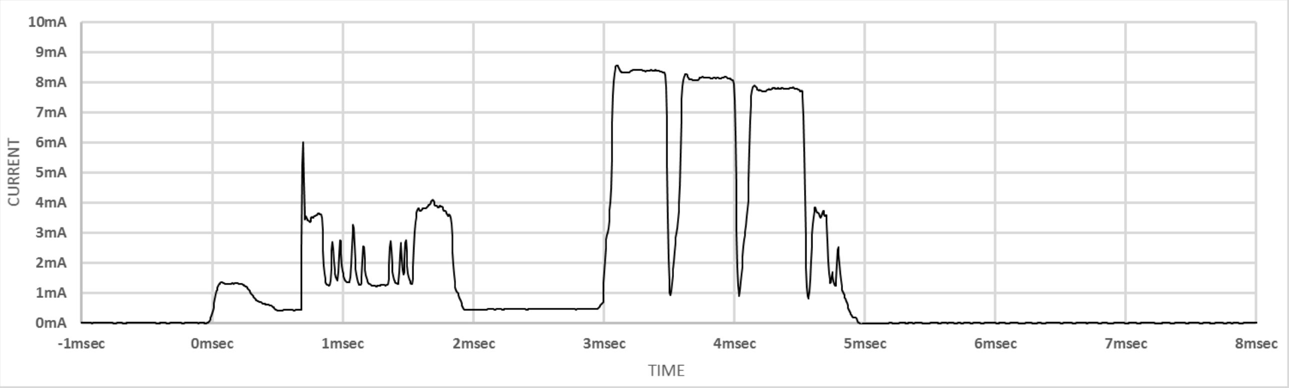


Figure 9-14 Current Consumption Waveform of ADV_NONCONN_IND at MCU Clock Frequency 16MHz (not use DC-DC converter)

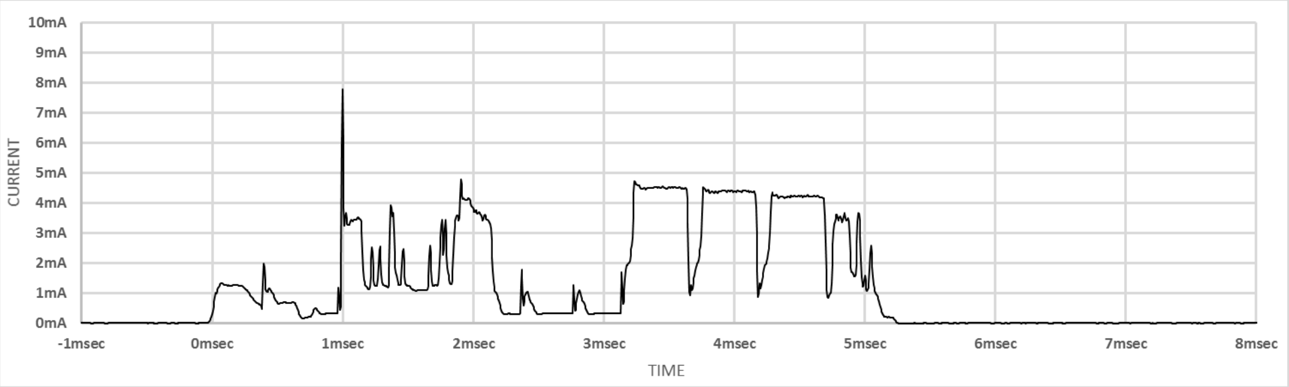


Figure 9-15 Current Consumption Waveform of ADV_NONCONN_IND at MCU Clock Frequency 16MHz (use DC-DC converter)

Current consumption waveforms of transmitting ADV_SCAN_IND packet and receiving no SCAN_REQ packet at Advertising interval 1sec, MCU clock frequency 16MHz are shown in **Figure 9-16** and **Figure 9-17**.

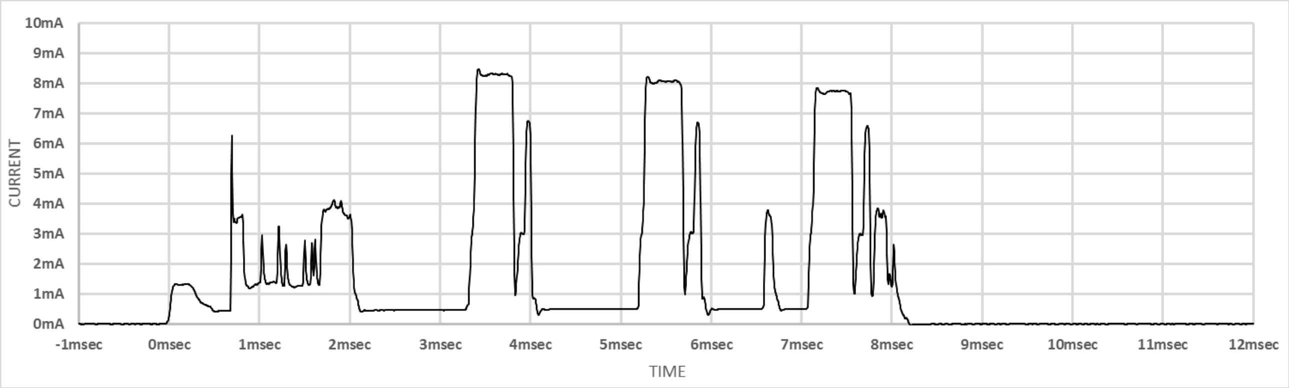


Figure 9-16 Current Consumption Waveform of ADV_SCAN_IND at MCU Clock Frequency 16MHz (not use DC-DC converter)

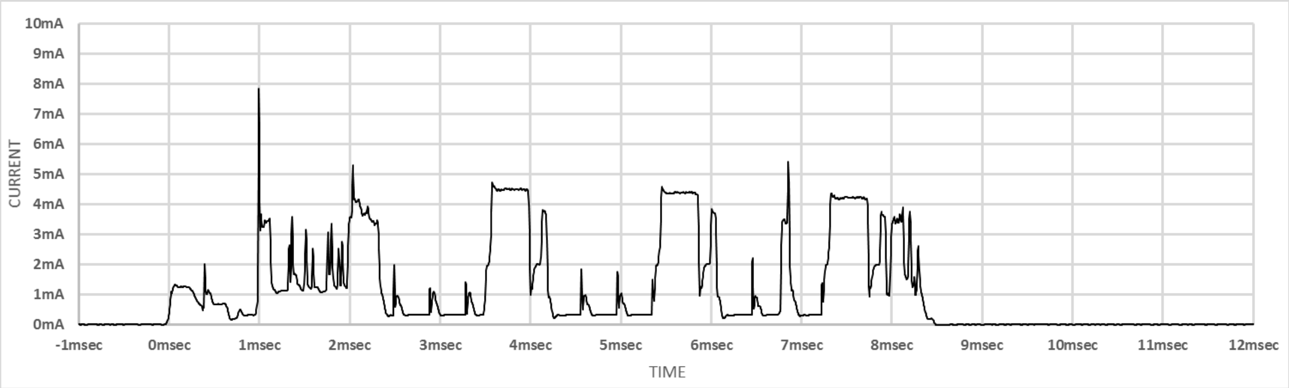


Figure 9-17 Current Consumption Waveform of ADV_SCAN_IND at MCU Clock Frequency 16MHz (use DC-DC converter)

(4) MCU Clock Frequency 32MHz

Table 9-15 shows the operation settings for measuring average current consumption at MCU clock frequency 32MHz.

Table 9-15 Operation Settings for Measuring Average Current Consumption

Category	Condition
Compiler	CC-RL V1.04.00
MCU Clock Frequency	32MHz
Supply Voltage	3V
DC-DC Converter	use RF on-chip DC-DC converter / not use RF on-chip DC-DC converter
RF Slow Clock Source	use RF on-chip oscillator, execute calibration
RF Operation	enable both Tx and Rx (RFCFG_TXRX)
Advertising Transmit Power	0dBm
Advertising Channels	3ch channels (37,38,39ch)
Advertising Type	ADV_NONCONN_IND / ADV_SCAN_IND
Advertising Data Length	31byte
Advertising Data Count	1
Advertising Interval	10msec / 100msec / 1sec Note: Interval range defined by Bluetooth Core Specification is 100msec to 10.24sec.

Measurement results of average current consumption at MCU clock frequency 32MHz are shown in **Table 9-16** and **Table 9-17**.

Table 9-16 Measurement result at MCU Clock Frequency 32MHz (when transmit ADV_NONCONN_IND)

Advertising Interval	Measurement Result of Average Current Consumption (transmit ADV_NONCONN_IND)	
	not use DC-DC converter	use DC-DC converter
10msec	1542.2uA	1037.3uA
100msec	165.4uA	119.8uA
1sec	18.2uA	13.6uA

Table 9-17 Measurement result at MCU Clock Frequency 32MHz (when transmit ADV_SCAN_IND)

Advertising Interval	Measurement Result of Average Current Consumption (transmit ADV_SCAN_IND, receive no SCAN_REQ)	
	not use DC-DC converter	use DC-DC converter
10msec	1812.4uA	1216.8uA
100msec	209.2uA	153.1uA
1sec	22.5uA	16.9uA

Current consumption waveforms of transmitting ADV_NONCONN_IND packet at Advertising interval 1sec, MCU clock frequency 32MHz are shown in **Figure 9-18** and **Figure 9-19**.

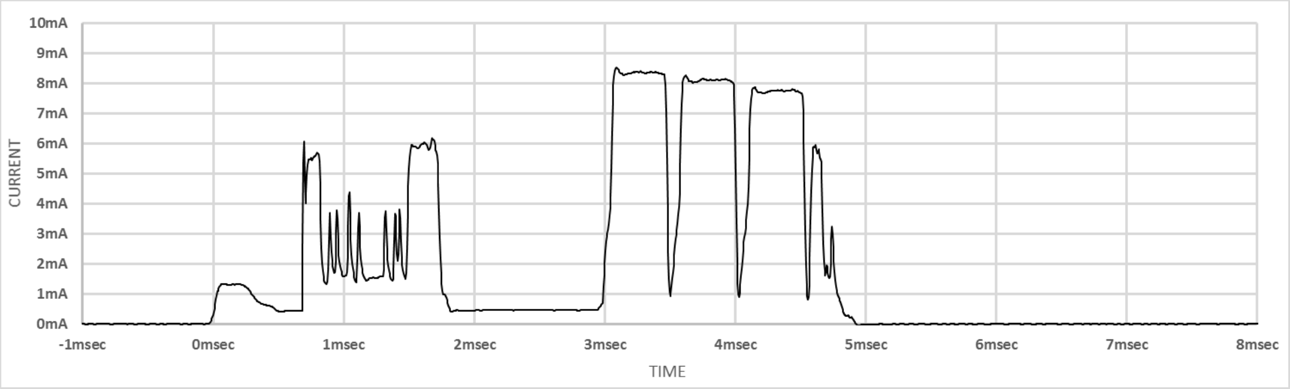


Figure 9-18 Current Consumption Waveform of ADV_NONCONN_IND at MCU Clock Frequency 32MHz (not use DC-DC converter)

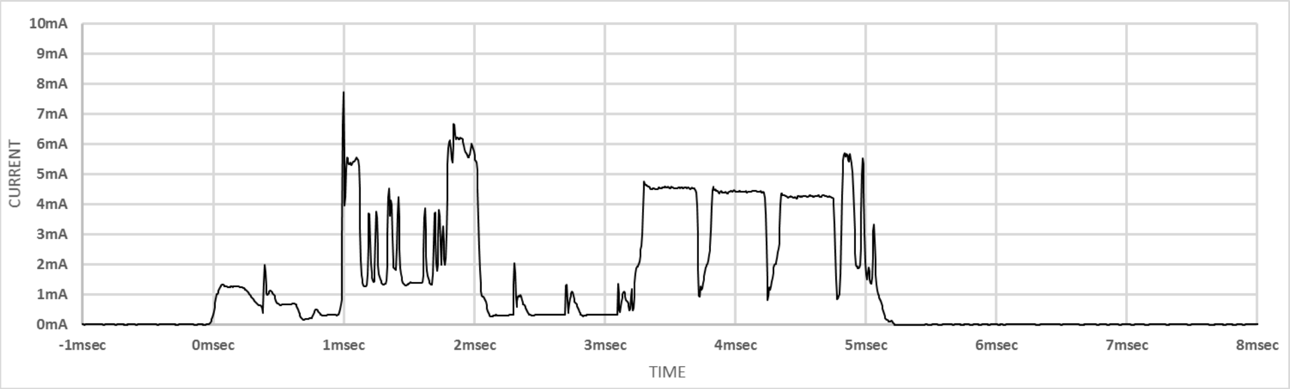


Figure 9-19 Current Consumption Waveform of ADV_NONCONN_IND at MCU Clock Frequency 32MHz (use DC-DC converter)

Current consumption waveforms of transmitting ADV_SCAN_IND packet and receiving no SCAN_REQ packet at Advertising interval 1sec, MCU clock frequency 32MHz are shown in **Figure 9-20** and **Figure 9-21**.

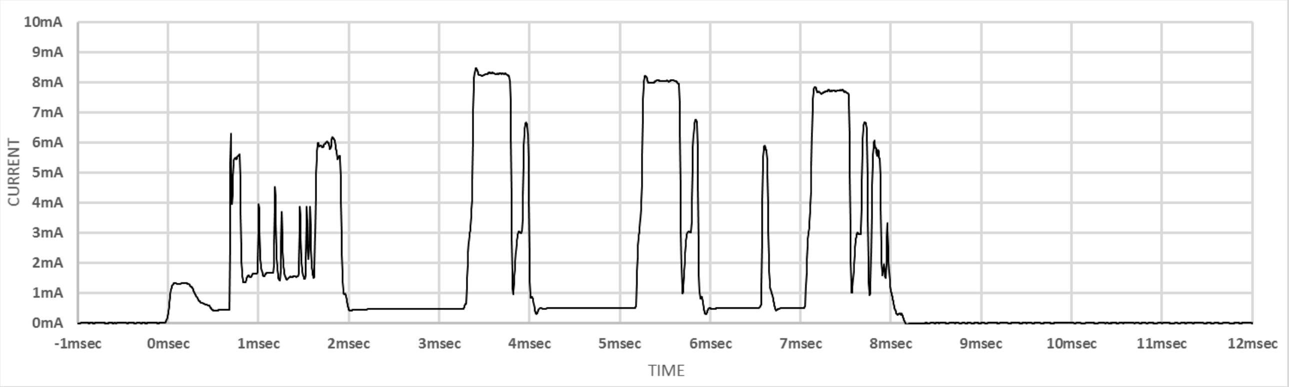


Figure 9-20 Current Consumption Waveform of ADV_SCAN_IND at MCU Clock Frequency 32MHz (not use DC-DC converter)

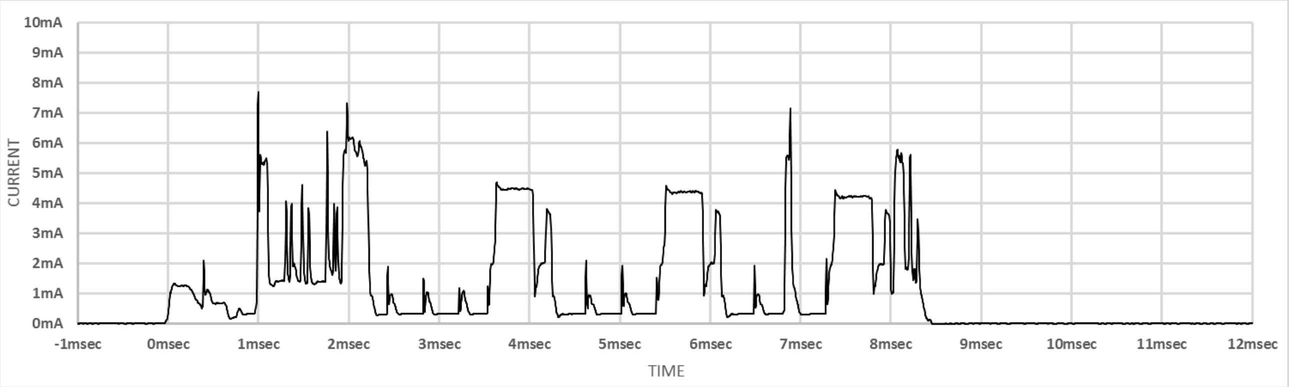


Figure 9-21 Current Consumption Waveform of ADV_SCAN_IND at MCU Clock Frequency 32MHz (use DC-DC converter)

9.2 Device Address

Device Address is 48-bit value for identifying each device. Device Address Types defined by Bluetooth Core Specification are shown as below.

- Public Device Address
  - ■ Public device address shall be created in accordance with section "48-bit universal LAN MAC addresses" of the IEEE 802-2001 standard and using a valid Organizationally Unique Identifier (OUI) obtained from the IEEE Registration Authority.
- Random Device Address
  - Static Device Address
    - ■ Static Device Address is a 48-bit randomly generated. Device may choose to initialize its address to a new value after each power cycle. And device shall not change its address value once initialized until the device is power cycled.
  - Private Device Address
    - Non-resolvable Private Address
      - ■ Non-resolvable Private Address is a 48-bit randomly generated. Its address should be changed over a period of time (recommended value of Bluetooth Core Specification is 15mins) for reducing the ability to track by other devices.
    - Resolvable Private Address
      - ■ Resolvable Private Address contains 24-bit randomly generated number and 24-bit hash generated with randomly generated number and Identity Resolving Key (IRK). Its address should be changed over a period of time (recommended value of Bluetooth Core Specification is 15mins) for reducing the ability to track by other devices.

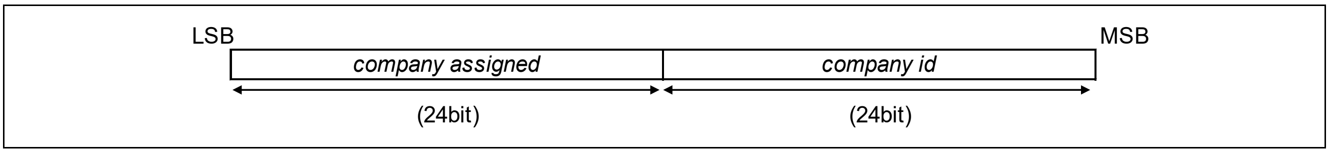


Figure 9-22 Public Device address format

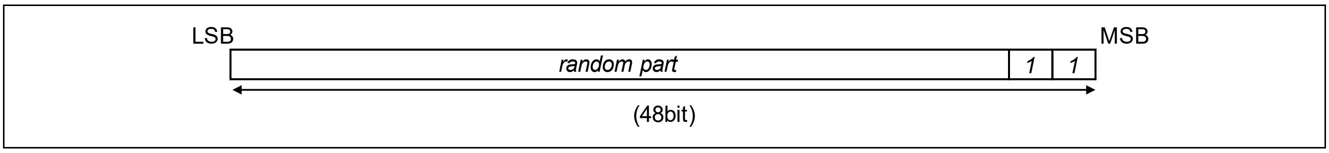


Figure 9-23 Static Device Address format

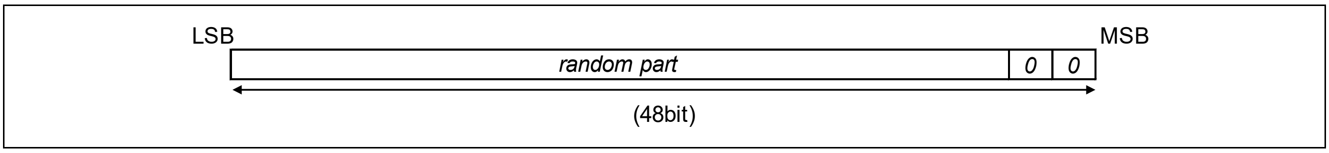


Figure 9-24 Non-resolvable Private Device Address format

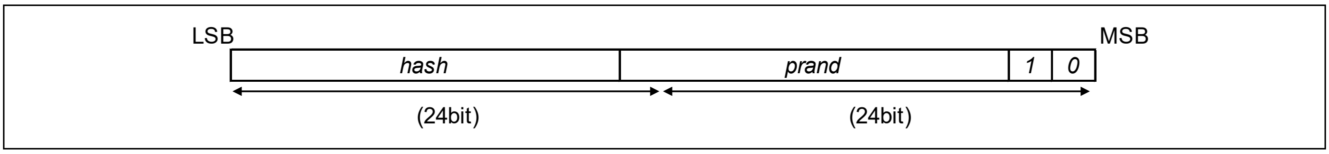


Figure 9-25 Resolvable Private Device Address

Regarding to the specification of Device Address, refer to [Vol. 6, Part B] Section 1.3, Bluetooth Core Specification v4.2.

9.3 Advertising Packet Format

Beacon Application transmits Non-connectable Undirected Advertising packet or Scannable Undirected Advertising packet. The packet format is common, and it is shown in **Figure 9-26**.

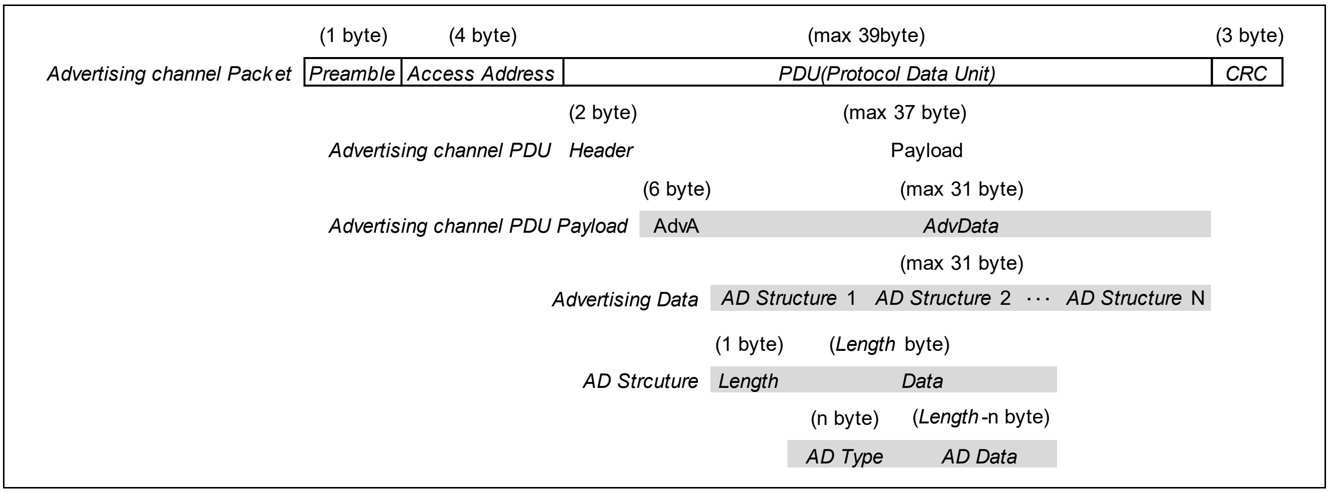


Figure 9-26 Advertising packet format

The specification of advertising packet is as shown below.

- advertising channel packet
  - Preamble : fixed 10101010b
  - Access Address : fixed 0x8E89BED6
  - Advertising channel PDU : Header and Payload
  - CRC : 24bits

Below fields are set by application.

- advertising channel PDU Payload
  - AdvA : Advertiser's Address is placed in
  - AdvData (Advertising data) : multiple AD structures are placed in, and muximum size is 31 bytes
    - AD structure : 1 byte part of Length information and Length bytes part of Data
      - Data : n bytes part of AD Type and (Length-n) bytes part of AD Data

Regarding to the details, refer to below specifications respectively.

- advertising packet format : [Vol. 6, Part B] Section 2.1, Bluetooth Core Specification v4.2
- advertising channel PDU format : [Vol. 6, Part B] Section 2.3, Bluetooth Core Specification v4.2
- advertising data format : [Vol. 3, Part C] Section 11, Bluetooth Core Specification v4.2
- AD Type : Part A, Supplement to the Bluetooth Core Specification v6.

Regarding to the definitions of AD Type, refer to below website.

- Bluetooth SIG Home > Specification > Assigned Numbers > Generic Access Profile
  https://www.bluetooth.com/specifications/assigned-numbers/generic-access-profile

Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

Revision History of Preceding Editions

Rev.	Date	Description	
1.00	Apr 13, 2016	First edition issued	
1.01	Jun 21, 2016	Chapter 3 Beacon Application	
		-	The version of Renesas Flash Programmer is updated in section 3.7
			The procedure of Renesas Flash Programmer is added in section 3.8
		-	SW7 setting when supplying from USB is added in Table 3-10
		Chapter 4 Appendix	
		-	This chapter is added
		-	Section 4.1 "Current consumption in periodic advertising packet transmission" is added
2.00	Oct 26, 2016	Section which describes Specification, Function and API of Beacon Stack are separated from this application note and issued as "RL78/G1D Beacon Stack Application User's Manual" (R01UW0171).	
		Chapter 5 Specification	
		P.44	Section 5.3 "Scan Application" is added
		P.54	Section 5.4 "DTM Application" is added
		Chapter 7 Functions	
		P.74	Subsection 7.1.2 "Scan Application" is added
		P.74	Subsection 7.1.3 "DTM Application" is added
		Chapter 8 Operation	
		P.77	Supsection 8.1.2 "Scan Application" is added
		P.78	Supsection 8.1.3 "DTM Application" is added
		P.80	Section 8.2 "Sequence" is added
		Chapter 9 Appendix	
		P.102	Section 9.2 "Device Address" is added
2.01	Jan 12, 2017	Chapter 4 Evaluation	
		P.14	Slide switch SW9 setting is corrected in Table 4-1
		Chapter 5 Specification	
		P.58	Resource used by the sample program is added in table of the Hardware Resouce used in Section 5.6
2.10	Mar 09, 2017	Chapter 2 Environment	
		P.8	Compiler and IDEs are updated in Software Environment
		Chapter 3 File Composition	
		P.9	Firmware files area added in File Composition
		Chapter 5 Specification	
		P.49	Example binary of command are added in Subsection 5.3.2
		P.56	Parameters read by application are added in Table 5-16
		P.59	Compiler version is updated in Section 5.7
		Chapter 6 Configuration	
		P.67	Subsection 6.1.9 "Hardware configuration for Energy Harvesting" is added
		P.68	Subsection 6.2.1 "Application Selection configuration" is added
2.11	Oct 12, 2017	Chapter 4 Operating Procedure	
		P.14	Slide switch SW9 setting is corrected in Table 4-1
		P.15	Unique code file setting is separated from wrting procedure in section 4.2
		Chapter 6 Configuration	
		P.68	Condition of building firmware files are added in subsection 6.2.1
		Chapter 9 Appendix	
		P.103	Description of advertising packet filelds is updated in section 9.3
		Overall	

		-	Newline character is changed to "LF"
		-	Some of chapter name and subsection name are changed
2.20	Jun 19, 2020	Introduction	
		P.1	About ASCII command transmission of beacon application is added.
		Chapter 1 Overview	
		P.6	Overview of Beacon Application-ASCII command control is added in section 1.2.
		Chapter 2 Environment	
		P.8	Software environment is updated.
		Chapter 3 File Composition	
		P.9	File composition is updated.
		Chapter 4 Operating Procedure	
		P.12	Build of Beacon Application-ASCII command control is added in section 4.1.
		P.16	Choise of Beacon Application-ASCII command control is added in section 4.3.
		P.18	Confirming operation of Beacon Application-ASCII command control in subsection 4.3.2.
		Chapter 5 Specification	
		P.28	Command specification of Beacon Application-ASCII command control in section 5.2.
		P.57	Accessing to code flash memory of Beacon Application-ASCII command control in subsection 5.5.2.
		P.58	Hardware resources of Beacon Application-ASCII command control is added in section 5.6.
		P.59	Compiler version is updated in section 5.7.
		P.59	Program size of Beacon Application-ASCII command control is added in section 5.9.
		P.60	Address map of Beacon Application-ASCII command control is updated in seciton 5.10.
		Chapter 6 Configuration	
		P.68	Application selection configuration of Beacon Application-ASCII command control is updated in subsection 6.2.1.
		P.70	System configuration of Beacon Application-ASCII command control is added in subsection 6.2.3.
		Chapter 7 Functions	
		P.75	Functions of Beacon Application-ASCII command control is added in subsection 7.1.4.
		Chapter 8 Operation	
		P.79	State transition figure of Beacon Application-ASCII command control is added in subsection 8.1.4.

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. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   ¾ 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 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. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   ¾ The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   ¾ The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   ¾ 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. Moreover, 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.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   ¾ The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the 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.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 7.1