User's Manual

RL78/G1D Beacon Stack

User's Manual

Software Library

Target Device

RL78/G1D

Renesas Electronics
www.renesas.com

Rev.2.10 Mar 2017

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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of 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. 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 should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The recommended 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; and industrial robots etc.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor 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 damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate 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. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may 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 should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be 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, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

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

(2012.4)

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.

How to Use This Manual

Purpose and Target Readers

This manual describes specification, functions, and API of RL78/G1D Beacon Stack, which is used for developing Bluetooth applications that incorporate with Bluetooth low energy microcontroller RL78/G1D device. This is intended to users who design the application systems incorporating with this software. In order to use this manual, basic knowledge of microcontrollers and Bluetooth low energy is necessary.

Related documents

The related documents are listed in below table. Make sure to refer to the latest versions of these documents. The latest versions of the documents might be obtained from the Renesas Electronics Web site.

Document Name	Document No.
RL78/G1D	
User's Manual: Hardware	R01UH0515E
RL78/G1D Evaluation Board	
User's Manual	R30UZ0048E
Renesas Flash Programmer V3.02 Flash memory programming software	
User's Manual	R20UT3841E
CC-RL Compiler	
User's Manual	R20UT3123E
RL78/G1D Beacon Stack Sample Program	
Basic Operation Application Note	R01AN3045E
Connecting and Updating Beacon Data Application Note	R01AN3313E

Contents

1. Outline

RL78/G1D Beacon Stack is software library, which runs on microcomputer supporting Bluetooth® Low Energy.

The RL78/G1D Beacon Stack provides functions such as RF Control, Advertising, Scanning and Direct Test Mode for using with RL78/G1D device. By executing with those restricted functions only, the Beacon Stack can execute Initialization processing and RF transmission/reception processing quickly, and consume lower power than Bluetooth Low Energy Protocol Stack.

1.1 Features

1.1.1 RF Control Function

RF Control Function initializes RF unit of RL78/G1D device and controls RF modes. It is possible to select below configuration for RF operation depending on the Beacon Stack' functions in which application uses.

- Enable both Tx and Rx
- Enable only Tx

When both Tx and Rx are enabled, all Beacon Stack's functions are available to Advertising Function, Scanning Function, and Direct Test Mode Function.

When only Tx is enabled, RF initialization time is shortened. On the other hand, the available Beacon Stack's functions are limited to transmitting Non-connectable Undirected Advertising packet of Advertising Function and RF Transmitter Test.

By enabling only Tx and setting short Advertising Interval, it is possible to transmit Advertising packets as many as possible with a very little energy generated by energy harvesting device.

Major configurations of RF Control Function are as shown below.

- RF on-chip DC-DC Converter : whether use RF on-chip DC-DC converter or not is selectable
- RF on-chip Oscillator : whether use RF on-chip oscillator or not is selectable

1.1.2 Advertising Function

Advertising Function is used for transmitting Advertising packets. Beacon Stack can execute below Advertising types.

- Non-connectable Undirected Advertising (ADV_NONCONN_IND)
- Scannable Undirected Advertising (ADV_SCAN_IND)

Regardless of the Advertising packet types, maximum 31 bytes data can be broadcasted in each transmission. While transmitting Scannable Undirected Advertising packet and receive Scan Request packet from peer device, additional maximum 31 bytes data is transmitted to peer device by transmitting Scan Response packet.

By using device filtering called White List, Scan Response packets are transmitted to only devices that are in the White List.

Major configurations of Advertising Function are as shown below.

- Advertising Channel : any 1 to 3 channels of 37,38,39ch can be selectable

- Advertising Interval

  ADV_NONCONN_IND (for 1ch) : 1.250msec to 30.72sec can be configurable in units of 625usec

  ADV_NONCONN_IND (for 2,3ch) : 2.500msec to 30.72sec can be configurable in units of 625usec

  ADV_SCAN_IND (for 1ch) : 2.500msec to 30.72sec can be configurable in units of 625usec

  ADV_SCAN_IND (for 2,3ch) : 7.500msec to 30.72sec can be configurable in units of 625usec

  Note the range of Advertising Interval defined by Bluetooth Core Specification is from 100msec to 10.24sec.

- Advertising Data : max 10 data can be configurable

1.1.3 Scanning Function

Scanning Function is used for receiving Advertising packets. Beacon Stack can execute below Scanning types.

- Passive Scan
- Active Scan

Both Passive Scan and Active Scan receive Advertising packets, and the difference is that Active Scan can request Scan Response packet by transmitting Scan Request packet. The data of received Advertising packet and Scan Response packet are notified as events.

By using device filtering known as White List, only Advertising packets and Scan Response packets from devices are in the White List.

Major configurations of Scanning Function are as shown below. Scan Interval can be selected from 1 to 3 channels, so it is possible to receive specified channel only.

- Scan Channel : any 1 to 3 channels of 37,38,39ch can be selectable
- Scan Interval : 2.500msec to 30.72sec can be configurable in units of 625usec

1.1.4 Advertising and Scan Switching Function

Advertising and Scan Switching Function is function which executes Advertising and Scanning alternately.

Major configurations in Advertising and Scan Switching are as shown below.

- Advertising Type : only Non-connectable Undirected Advertising(ADV_NONCONN_IND)
- Scan Type : only Passive Scan
- Advertising and Scan Channel : only 1 channel of 37,38,39ch can be selectable
- Advertising Interval : 2.500msec to 30.72sec can be configurable in units of 625usec

  Note the range of Advertising Interval defined by Bluetooth Core Specification is from 100msec to 10.24sec.

1.1.5 Direct Test Mode Function

Direct Test Mode Function is used for evaluating RF characteristic of RL78/G1D device. Beacon Stack can executes below RF tests.

- RF Transmitter Test
- RF Receiver Test

Major configurations of Direct Test Mode Function are as shown below.

- Tx and Rx Frequency : 2402MHz to 2480MHz can be configurable

- Tx Packet Data Length : 0 to 37 bytes can be configurable
- The number of Tx Packet : no limitation or 1 to 65535 packets can be configurable
- Packet Tx and Rx Mode : Transmit or Receive packets every 625usec for Direct Test Mode
- Infinite Tx and Rx Mode : Activate Transmitting or Receiving constantly for measuring RF current
- Continuous Wave (CW) Mode : Output Continuous Wave (CW) for technology conformance inspection.

2. Specification

2.1 Hardware Resources used

MCU Unit		
	Clock generator	MCU main system clock (f_{MAIN}) can be selected from below frequencies of High-speed on-chip oscillator clock (f_{IH}). • 4MHz • 8MHz • 16MHz • 32MHz Note: Only High-speed on-chip oscillator clock can be used as MCU main system clock. External main system clock (f_{EX}) can not be used.
		To be selected whether to use XT1 oscillator clock (f_{XT}) or not. • use XT1 oscillator • not to use XT1 oscillator (use RF on-chip oscillator) When RF on-chip oscillator is not used, it is necessary to generate clock for RF slow clock by using XT1 oscillator, set the clock output from PCLBUZ0 pin, and the clock input to EXSLK_RF pin. XT1 oscillator needs 32.768kHz crystal resonator connected at XT1, XT2 pin.
	Clock output/buzzer output	To be selected whether to output clock from PCLBUZ0 pin for RF slow clock or not. • output 16.384kHz clock from PCLBUZ0 pin (XT1 oscillator is needed) • output 32.768kHz clock from PCLBUZ0 pin (XT1 oscillator is needed) • not to use PCLBUZ0 When PCLBUZ0 is not used, it is necessary to use RF on-chip oscillator.
	Timer Array Unit	use TM00, and set operation clock CK00 to 1MHz
	Serial array unit	use CSI21
	DMA controller	use DMA2, DMA3
	Interrupt	use INTRF, INTDMA2, INTDMA3, INTTM00
RF Unit		
	DC-DC Converter	To be selected whether to use RF on-chip DC-DC converter or not. • use RF on-chip DC-DC converter • not to use RF on-chip DC-DC converter When DC-DC converter is used, DC-DC converter needs external feedback circuit.
	Oscillator for RF slow clock	To be selected whether to use RF on-chip oscillator or not. • use RF on-chip oscillator • not to use RF on-chip oscillator When RF on-chip oscillator is not used, needed to supply 16.384kHz clock or 32.768kHz clock to EXSLK_RF pin.
	Clock Output	Frequency-divided clock of RF base clock (32MHz) can be output from CLKOUT_RF pin. • no output • output 16MHz clock • output 8MHz clock • output 4MHz clock When no output is selected, CLKOUT_RF pin is set as input.

2.2 Compiler

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

Compiler : Renesas CC-RL V1.04.00

2.3 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 with Beacon Stack.

Memory Model : -memory_model=medium

2.4 Section Size

Table 2-1 shows the section size of Beacon Stack.

Table 2-1 Beacon Stack Section Size

Relocation Attribute for CC-RL	Section Name	Section Size
CALLT0	.callt0	12byte
BSS	BCN_BSS_n	2,506byte
DATA	-	-
DATAF	-	-
CONST	BCN_CONST_n	328byte
CONSTF	-	-
TEXT	BCN_TEXT_n	1,080byte
TEXTF	BCN_TEXT_f	13,839byte

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

3. Functions

3.1 RF Control Function

RF unit has several modes, which are independent of MCU modes. Thus, current consumption is different in each RF mode. The correlation between current consumption and RF modes is as shown below.

POWER_DOWN = RESET_RF < DEEP_SLEEP < STANDBY_RF < IDLE_RF < SETUP_RF < TX, RX

By calling R_RF_PowerUp and R_RF_Init of Beacon Stack API, RF unit is initialized and RF mode is changed to either IDLE_RF or DEEP_SLEEP, which is available Advertising, Scanning and DTM.

When start Advertising, Beacon Stack changes RF mode to IDLE_RF. After transmitting Advertising packet, Beacon Stack changes RF mode to DEEP_SLEEP until next transmission timing, which defined according to Advertising interval. When stop Advertising, Beacon Stack changes RF mode to DEEP_SLEEP.

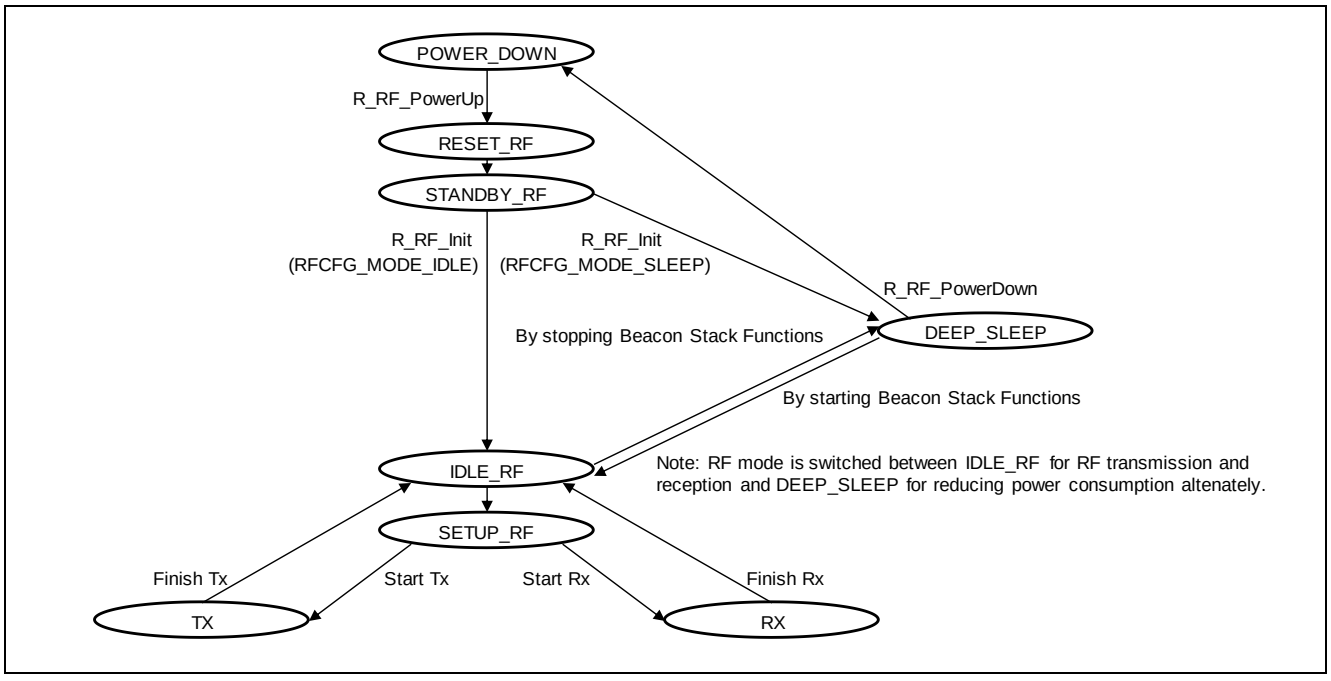
When start Scanning, Beacon Stack changes RF mode to IDLE_RF and starts receiving packets. When stop Scanning, Beacon Stack changes RF mode to DEEP_SLEEP.

When start Advertising and Scan Switching, Beacon Stack changes RF mode to IDLE_RF. After transmitting Advertising packet and executing Scannning, Beacon Stack changes RF mode to DEEP_SLEEP until next executing timing, which defined according to Advertising interval. When stop Advertising and Scan Switching, Beacon Stack changes RF mode to DEEP_SLEEP.

When start DTM, Beacon Stack changes RF mode to IDLE_RF, and start transmitting or receiving. When stop DTM, Beacon Stack changes RF mode to DEEP_SLEEP.

If none of the Advertising, Scanning, and DTM is executed for a long time, it is possible to reduce further current consumption by calling R_RF_PowerDown. Thereafter, it is necessary to initialize RF unit again by calling R_RF_PowerUp and R_RF_Init when restart one of the Advertising, Scanning, and DTM.

Figure 3-1 RF Mode Transition



Regarding to the details of RF modes, refer to section 15.4 "RF modes" in RL78/G1D User's Manual: Hardware (R01UH0515).

3.2 Advertising Function

3.2.1 Non-connectable Undirected Advertising

By calling Beacon Stack API "R_BLE_StartAdvertising" with RBLE_PDU_ADV_NONCONN_IND in argument adv_type, Beacon Stack starts to transmit Non-connectable Undirected Advertising (ADV_NONCONN_IND) packets. By calling Beacon Stack API "R_BLE_StopAdvertising", Beacon Stack stops Advertising.

Figure 3-2 Non-connectable Undirected Advertising in the case of 3channels

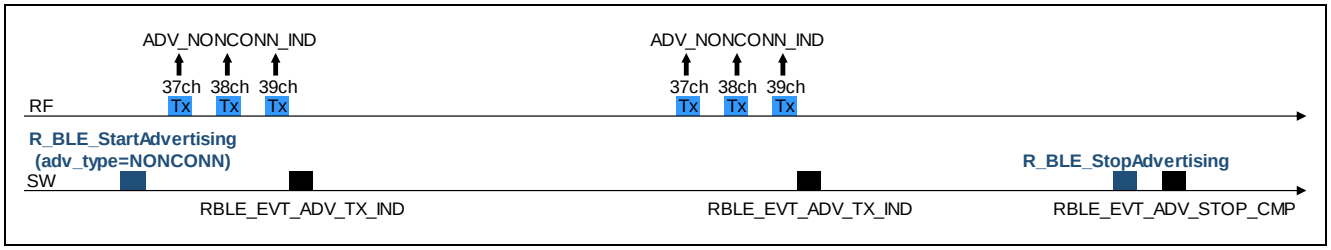
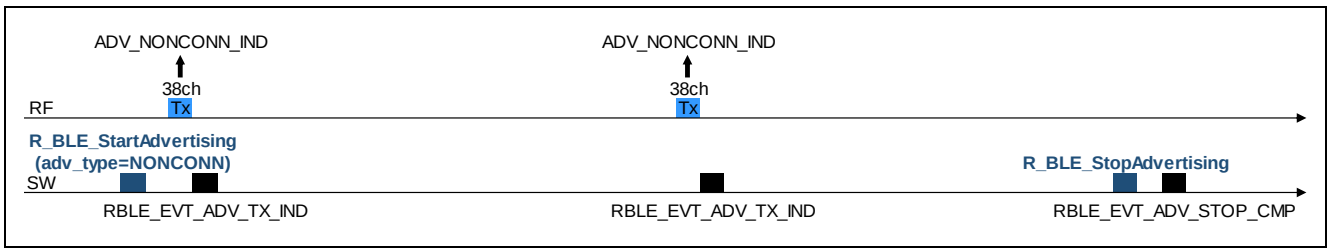


Figure 3-3 Non-connectable Undirected Advertising in the case of only 1channel



3.2.2 Scannable Undirected Advertising

By calling Beacon Stack API "R_BLE_StartAdvertising" with RBLE_PDU_ADV_SCAN_IND in argument adv_type, Beacon Stack starts to transmit Scannable Undirected Advertising (ADV_SCAN_IND) packets. By calling Beacon Stack API "R_BLE_StopAdvertising", Beacon Stack stops Advertising.

Figure 3-4 Scannable Undirected Advertising in the case of 3channels

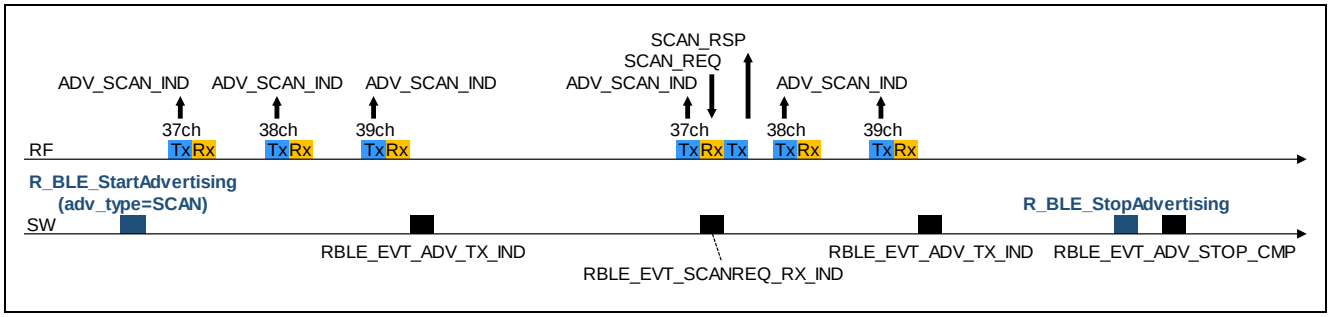
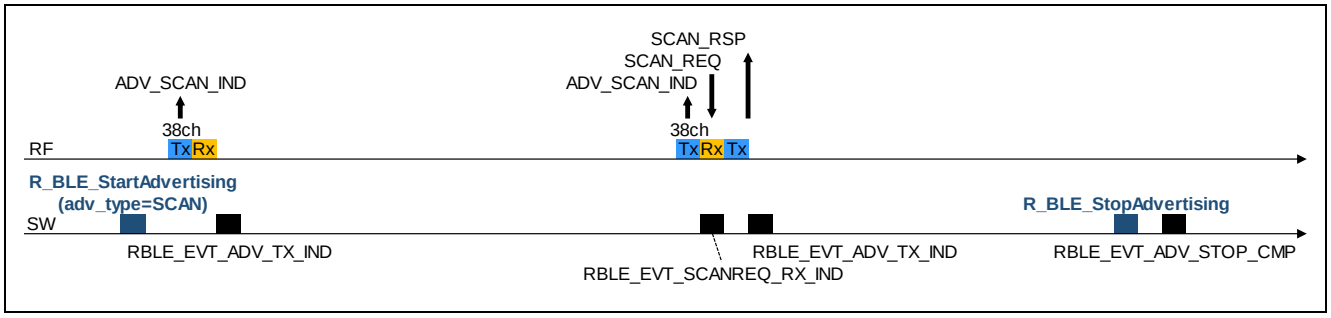


Figure 3-5 Scannable Undirected Advertising in the case of only 1channel



3.3 Scanning Function

3.3.1 Passive Scan

By calling Beacon Stack API "R_BLE_StartScanning" with RBLE_SCAN_PASSIVE in argument scan_type, Beacon Stack starts Passive Scan. By calling Beacon Stack API "R_BLE_StopScanning", Beacon Stack stops Scanning.

Figure 3-6 Passive Scan in the case of 3channels

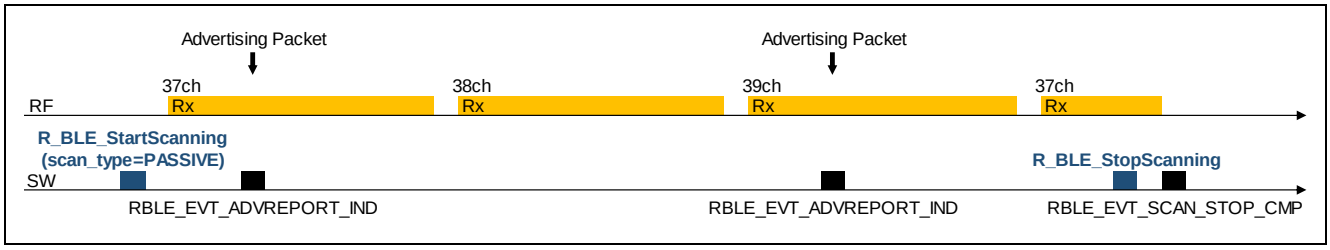
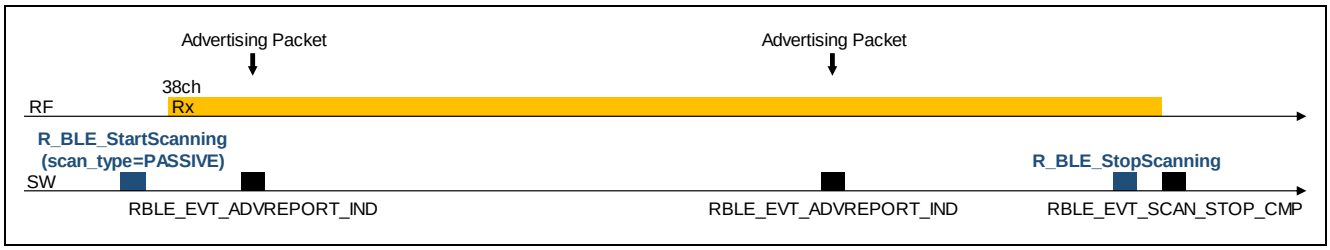


Figure 3-7 Passive Scan in the case of only 1channel



3.3.2 Active Scan

By calling Beacon Stack API "R_BLE_StartScanning" with RBLE_SCAN_ACTIVE in argument scan_type, Beacon Stack starts Active Scan. By calling Beacon Stack API "R_BLE_StopScanning", Beacon Stack stops Scanning.

Figure 3-8 Active Scan in the case of 3channels

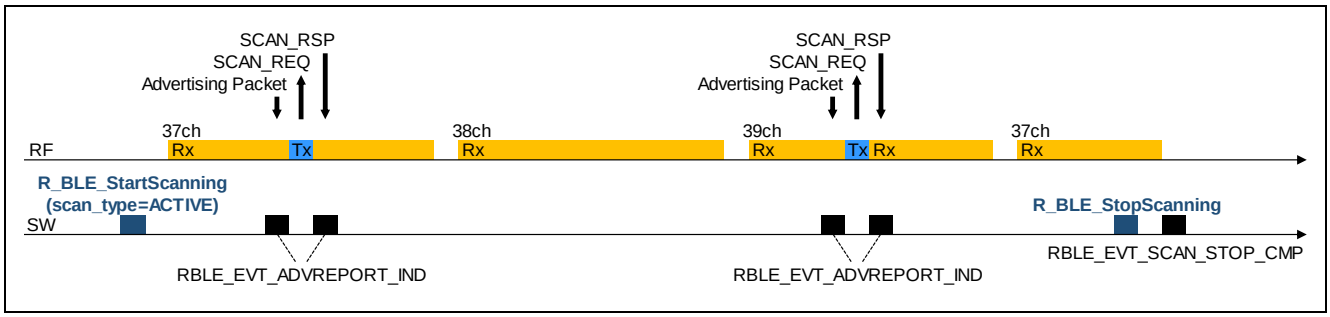
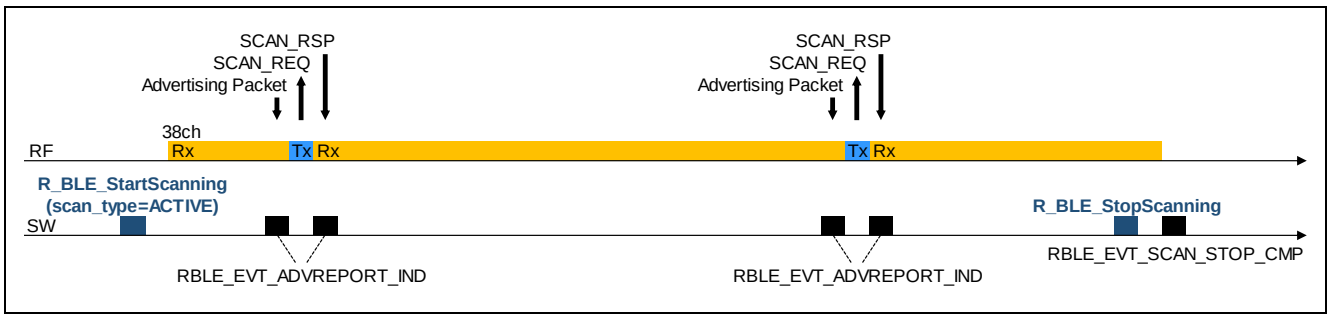


Figure 3-9 Active Scan in the case of only 1channel



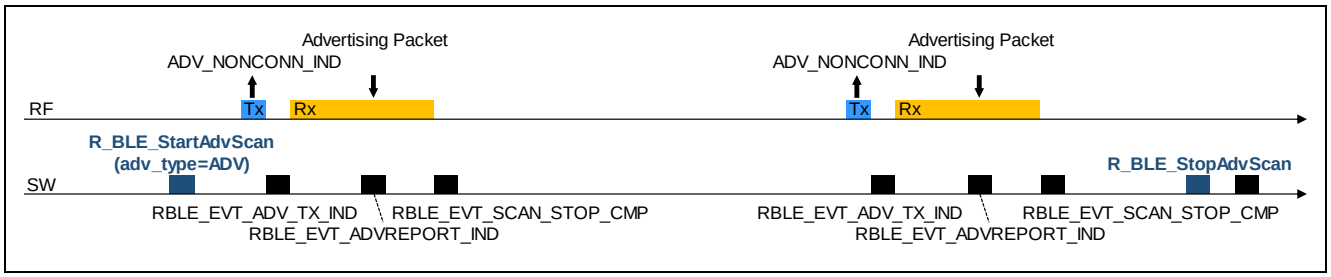
3.4 Advertising and Scan Switching Function

3.4.1 Starting with Advertising

By calling Beacon Stack API "R_BLE_StartAdvScan" with RBLE_PDU_ADV_NONCONN_IND in argument adv_type, Beacon Stack transmits Non-connectable Undirected Advertising (ADV_NONCONN_IND) packet. After transmitting, Beacon Stack executes Scanning within specified period.

Beacon Stack executes Advertising and Scanning periodically. By calling Beacon Stack API "R_BLE_StopAdvScan", Beacon Stack stops operation.

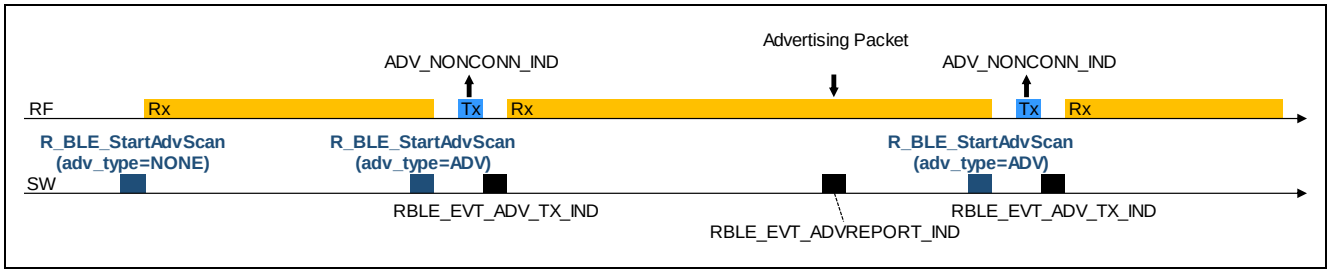
Figure 3-10 Advertising and Scan Switching, starting with Advertising



3.4.2 Starting with Scanning

By calling Beacon Stack API "R_BLE_StartScanning" with RBLE_PDU_NO_TYPE in argument adv_type, Beacon Stack executes Scanning only. By calling Beacon Stack API "R_BLE_StartAdvScan" with argument RBLE_PDU_ADV_NONCONN_IND in adv_type at an optional timing, Beacon Stack suspends Scanning and transmits Non-connectable Undirected Advertising (ADV_NONCONN_IND) packet. After transmitting, Beacon Stack restarts Scanning.

Figure 3-11 Advertising and Scan Switching, starting with Scanning

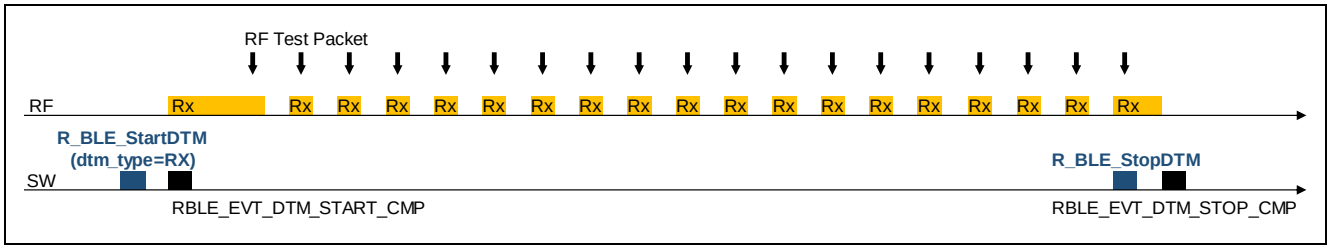


3.5 Direct Test Mode Function

3.5.1 RF Receiver Test

By calling Beacon Stack API "R_BLE_StartDTM" with RBLE_DTM_RX in argument dtm_type, Beacon Stack starts RF Receiver Test. By calling Beacon Stack API "R_BLE_StopDTM", Beacon Stack stops RF Receiver Test.

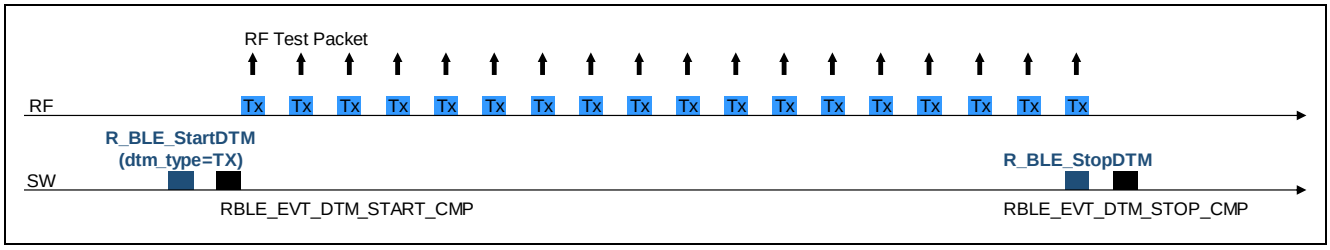
Figure 3-12 RF Receiver Test



3.5.2 RF Transmitter Test

By calling Beacon Stack API "R_BLE_StartDTM" with RBLE_DTM_TX in argument dtm_type, Beacon Stack starts RF Transmitter Test. By calling Beacon Stack API "R_BLE_StopDTM", Beacon Stack stops RF Transmitter Test.

Figure 3-13 RF Transmitter Test



4. API

4.1 Type

Table 4-1 shows list of the types, which is defined by Beacon Stack.

Table 4-1 Type

Type Name	Standard Type	Description
uint8_t	unsigned char	un-signed 8bit integer
uint16_t	unsigned short	un-signed 16bit integer
uint32_t	unsigned long	un-signed 32bit integer
int8_t	signed char	signed 8bit integer
int16_t	signed short	signed 16bit integer
int32_t	signed long	signed 32bit integer
bool	unsigned char	boolean
int_t	signed int	signed integer
uint_t	unsigned int	un-signed integer
char_t	char	character
RBLE_STATUS	unsigned char	return value of Beacon Stack function

4.2 Macros

4.2.1 Status macro

Table 4-2 shows the list of status macro, which returns Beacon Stack.

Table 4-2 Status macro

Macro Name	Value	Description
RBLE_OK	0x00	Success
RBLE_ERR_PARAM	0x01	Error : invalid parameter
RBLE_ERR_WL	0x02	Error : White List is empty
RBLE_ERR_PWRDOWN	0x03	Error : supplying power to RF unit is stopped
RBLE_ERR_PWRUP	0x04	Error : supplying power to RF unit is already started
RBLE_ERR_RFRX	0x05	Error : RF unit Rx is disabled
RBLE_ERR_START	0x06	Error : RF operation is already started
RBLE_ERR_STOP	0x07	Error : RF operation is stopped
RBLE_ERR_HW_STANDBY	0x08	Error : RF unit abnormality in STANDBY_RF
RBLE_ERR_HW_STANDBYRX	0x09	Error : RF unit abnormality in STANDBY_RF and enabling Rx
RBLE_ERR_HW_IDLE	0x0A	Error : RF unit abnormality in IDLE_RF

4.2.2 Event macro

Table 4-3 shows the list of event macro, which is notified by Beacon Stack. To get an event, use R_BLE_GetEvent.

Table 4-3 Event macro

Macro Name	Value	Description
RBLE_EVT_ADV_TX_IND	0x01	Advertising event : Advertising packet is transmitted
RBLE_EVT_SCANREQ_RX_IND	0x02	Advertising event : Scan Request packet is received
RBLE_EVT_ADV_STOP_CMP	0x03	Advertising event : Advertising is stopped
RBLE_EVT_ADVREPORT_IND	0x04	Scanning event : information about Advertising packet received
RBLE_EVT_SCAN_STOP_CMP	0x05	Scanning event : Scanning is stopped
RBLE_EVT_DTM_START_CMP	0x06	DTM event : Direct Test Mode is started
RBLE_EVT_DTM_STOP_CMP	0x07	DTM event : Direct Test Mode is stopped

4.2.3 RF Initialization Configuration macro

Table 4-4 shows the list of RF initialization configuration macro that used for the argument rf_flg of Beacon Stack API "R_RF_PowerUp". Macros shown in bold are default configuration of sample program.

Table 4-4 RF Initialization Configuration macro

Macro Name	Value	Description
RFCFG_TX	0x0000	enable only Tx (Rx is disabled, but RF initialization time is shortened)
RFCFG_TXRX	0x0001	enable both Tx and Rx
RFCFG_DCDC_ON	0x0000	use RF on-chip DC-DC converter
RFCFG_DCDC_OFF	0x0002	not use RF on-chip DC-DC converter
RFCFG_INT_32KHZ	0x0000	use RF on-chip oscillator clock as RF slow clock (32.768kHz) not execute RF on-chip oscillator calibration
RFCFG_INT_32KHZCAL	0x0010	use RF on-chip oscillator clock as RF slow clock (32.768kHz) execute RF on-chip oscillator calibration
RFCFG_EXT_32KHZ	0x0020	use MCU crystal oscillator clock as RF slow clock (32.768kHz)
RFCFG_EXT_16KHZ	0x0040	use MCU crystal oscillator clock as RF slow clock (16.384kHz)
RFCFG_MODE_IDLE	0x0000	after RF initialization, RF mode is changed to IDLE_RF
RFCFG_MODE_SLEEP	0x0080	after RF initialization, RF mode is changed to DEEP_SLEEP
RFCFG_OUT_NONE	0x0000	not output clock from RF unit
RFCFG_OUT_16MHZ	0x0300	output clock from RF unit (16MHZ)
RFCFG_OUT_8MHZ	0x0400	output clock from RF unit (8MHZ)
RFCFG_OUT_4MHZ	0x0500	output clock from RF unit (4MHZ)

4.2.4 Device Address Type macro

Table 4-5 shows the list of device address type macro. This macro is used for setting own device address type in Advertising or Scanning, as well as setting device address type to White List. Beacon Stack uses the macro to notify device address type of peer device, which transmits the received packet in Advertising and Scanning.

Table 4-5 Device Address Type macro

Macro Name	Value	Description
RBLE_ADDR_PUBLIC	0x00	Public Device Address
RBLE_ADDR_RANDOM	0x01	Random Device Address

4.2.5 Advertising Channel macro

Table 4-6 shows the list of Advertising channel macro. This macro is used for setting channel map of Advertising and Scanning. Beacon Stack uses the macro to notify channel of received Scan Request packet in Advertising as well as received Advertising packet or Scan Response packet in Scanning.

Table 4-6 Advertising Channel macro

Macro Name	Value	Description
RBLE_ADV_CHANNEL_37	0x01	37 channel
RBLE_ADV_CHANNEL_38	0x02	38 channel
RBLE_ADV_CHANNEL_39	0x04	39 channel
RBLE_ADV_ALL_CHANNELS	0x07	All channels (37, 38, 39 channel)

4.2.6 Tx Power macro

Table 4-7 shows the list of Tx power macro that used for setting Tx power level to the packet in Advertising, Scanning, and Direct Test Mode.

Table 4-7 Tx Power macro

Macro Name	Value	Description
RBLE_TXPW_LV1	0x01	Tx Power Level 1 (-15dBm)
RBLE_TXPW_LV2	0x02	Tx Power Level 2 (-10dBm)
RBLE_TXPW_LV3	0x03	Tx Power Level 3 (-7dBm)
RBLE_TXPW_LV4	0x04	Tx Power Level 4 (-2dBm)
RBLE_TXPW_LV5	0x05	(reserved)
RBLE_TXPW_LV6	0x06	(reserved)
RBLE_TXPW_LV7	0x07	Tx Power Level 7 (-1dBm)
RBLE_TXPW_LV8	0x08	(reserved)
RBLE_TXPW_LV9	0x09	Tx Power Level 9 (0dBm)

4.2.7 PDU Type macro

Table 4-8 shows the list of Protocol Data Unit (PDU) type macro that used for setting packet type in Advertising. This macro is also used for notifying packet type, which is received in Scanning.

Table 4-8 PDU Type macro

Macro Name	Value	Description
RBLE_PDU_ADV_IND	0x00	Connectable Undirected Advertising (ADV_IND)
RBLE_PDU_ADV_DIRECT_IND	0x01	Connectable Directed Advertising (ADV_DIRECT_IND)
RBLE_PDU_ADV_NONCONN_IND	0x02	Non-connectable Undirected Advertising (ADV_NONCONN_IND)
RBLE_PDU_SCAN_REQ	0x03	Scan Request (SCAN_REQ)
RBLE_PDU_SCAN_RSP	0x04	Scan Response (SCAN_RSP)
RBLE_PDU_CONNECT_REQ	0x05	Connect Request (CONNECT_REQ)
RBLE_PDU_ADV_SCAN_IND	0x06	Scannable Undirected Advertising (ADV_SCAN_IND)
RBLE_PDU_NO_TYPE	0x0F	Not Specified Type

4.2.8 Advertising Event Permission macro

Table 4-9 shows the list of Advertising Event Permission macro that used for setting whether Advertising event is notified or not by Beacon Stack in Advertising.

Table 4-9 Advertising Event Permission macro

Macro Name	Value	Description
RBLE_EVT_PERMIT_NONE	0x00	not permit to notify event
RBLE_EVT_PERMIT_ADV_TX	0x01	permit to notify RBLE_EVT_ADV_TX_IND event
RBLE_EVT_PERMIT_ADV_STOP	0x02	permit to notify RBLE_EVT_ADV_STOP_CMP event
RBLE_EVT_PERMIT_ADV_RXREQ	0x04	permit to notify RBLE_EVT_SCANREQ_RX_IND event
RBLE_EVT_PERMIT_ADV_ALL	0x07	permit to notify All Advertising event

4.2.9 Scan Type macro

Table 4-10 shows the list of Scan type macro that used for setting Scan type in Scanning.

Table 4-10 Scan Type macro

Macro Name	Value	Description
RBLE_SCAN_PASSIVE	0x00	Passive Scan
RBLE_SCAN_ACTIVE	0x01	Active Scan

4.2.10 Direct Test Mode Type macro

Table 4-11 shows the list of Direct Test Mode macro that used for setting Direct Test Mode type in DTM.

Table 4-11 Direct Test Mode Type macro

Macro Name	Value	Description
RBLE_DTM_RX	0x00	RF Receiver Test
RBLE_DTM_TX	0x01	RF Transmitter Test

4.2.11 Direct Test Mode Modulation Configuration macro

Table 4-12 shows the list of Direct Test Mode Modulation Configuration macro that used for setting Direct Test Mode Modulation Configuration in DTM.

Table 4-12 Direct Test Mode Modulation Configuration macro

Macro Name	Value	Description
RBLE_DTM_MODON_PACKET	0x00	Modulation ON, Transmit or Receive packets every 625usec
RBLE_DTM_MODON_INFINITE	0x01	Modulation ON, Activate Transmitting or Receiving constantly
RBLE_DTM_MODOFF_CW	0x02	Modulation OFF, Continuous Wave(CW)

4.2.12 Direct Test Mode Payload macro

Table 4-12 shows the list of Direct Test Mode payload macro that used for setting the payload data in RF Transmitter Test of DTM.

Table 4-13 Direct Test Mode Payload macro

Macro Name	Value	Description
RBLE_PAYLOAD_PRBS9	0x00	9-bit pseudorandom binary sequence (PRBS9)
RBLE_PAYLOAD_11110000	0x01	b’11110000 bits sequence
RBLE_PAYLOAD_10101010	0x02	b’10101010 bits sequence
RBLE_PAYLOAD_PRBS15	0x03	15-bit pseudorandom binary sequence (PRBS15)
RBLE_PAYLOAD_ALL_1	0x04	b’11111111 bits sequence
RBLE_PAYLOAD_ALL_0	0x05	b’00000000 bits sequence
RBLE_PAYLOAD_00001111	0x06	b’00001111 bits sequence
RBLE_PAYLOAD_01010101	0x07	b’01010101 bits sequence

4.3 Structures

4.3.1 Device Address structure

Macro Name	Type	Offset	Description
struct RBLE_BD_ADDR			
addr	uint8_t[6]	0	Device Address

4.3.2 Device Information structure

Macro Name	Type	Offset	Description
struct RBLE_DEV_INFO			
dev_type	uint8_t	0	Device Address Type
reserved	uint8_t	1	(reserved)
dev_addr	RBLE_BD_ADDR	2	Device Address

4.3.3 Version structure

Macro Name	Type	Offset	Description
struct RBLE_VERSION			
major	uint8_t	0	Major Version
minor	uint8_t	1	Minor Version

4.3.4 Advertising Data structure

Macro Name	Type	Offset	Description
struct RBLE_ADV_DATA			
len	uint8_t	0	Advertising Data Length
data	uint8_t[31]	1	Advertising Data

4.3.5 Advertising Information structure

Macro Name	Type	Offset	Description
struct RBLE_ADV_INFO			
interval	uint16_t	0	Advertising Interval
delay	bool	2	Add Random Delay to Advertising Interval
ch_map	uint8_t	3	Advertising Channel Map
loop_cnt	uint8_t	4	Advertising Count Limitation
tx_pwr	uint8_t	5	Advertising Tx Power
own_addr	RBLE_BD_ADDR	6	Own Device’s Device Address
own_addr_type	uint8_t	12	Own Device’s Device Address Type
data_cnt	uint8_t	13	the number of Advertising Data
data	*RBLE_ADV_DATA	14	Advertising Data Array
evt_permit	uint8_t	16	Advertising Event Permission
use_wl	bool	17	use White List

4.3.6 Scanning Information structure

Macro Name	Type	Offset	Description
struct RBLE_SCAN_INFO			
interval	uint16_t	0	Scan Interval
ch_map	uint8_t	2	Scan Channel Map
tx_pwr	uint8_t	3	Scan Request Tx Power
own_addr	RBLE_BD_ADDR	4	Own Device's Device Address
own_addr_type	uint8_t	10	Own Device's Device Address Type
use_wl	bool	11	use White List

4.3.7 Advertising and Scan Information structure

Macro Name	Type	Offset	Description
struct RBLE_ADVSCN_INFO			
interval	uint16_t	0	Advertising Interval
delay	bool	2	Add Random Delay to Advertising Interval
ch_map	uint8_t	3	Advertising and Scan Channel Map
loop_cnt	uint8_t	4	Advertising Count Limitation
tx_pwr	uint8_t	5	Packet Tx Power
own_addr	RBLE_BD_ADDR	6	Own Device's Device Address
own_addr_type	uint8_t	12	Own Device's Device Address Type
data_cnt	uint8_t	13	the number of Advertising Data
data	*RBLE_ADV_DATA	14	Advertising Data
evt_permit	uint8_t	16	Advertising Event Permission
offset	uint8_t	17	Scan Window Offset
window	uint16_t	18	Scan Window Size
continuous	bool	20	Continuous Execution
use_wl	bool	21	use White List

4.3.8 Direct Test Mode Information structure

Macro Name	Type	Offset	Description
struct RBLE_DTM_INFO			
mod	uint8_t	0	Modulation Configuration
freq	uint8_t	1	Frequency
reserved	uint8_t	2	(reserved)
tx_pwr	uint8_t	3	Tx Power
tx_datalen	uint8_t	4	Tx Data Length
tx_payload	uint8_t	5	Tx Data Payload Type
tx_num	uint16_t	6	the number of Tx Packet

4.3.9 Advertising Tx Event structure

Macro Name	Type	Offset	Description
struct RBLE_ADVTX_IND			
tx_cnt	uint16_t	0	Advertising transmitted count
data_idx	uint8_t	2	Advertising data buffer index
reserved	uint8_t	3	(reserved)

4.3.10 Scan Request Rx Event structure

Macro Name	Type	Offset	Description
struct RBLE_SCANREQ_IND			
ch_idx	uint8_t	0	Rx channel
rssi	int8_t	1	RSSI
dev_addr	RBLE_BD_ADDR	2	Device Address

4.3.11 Advertising Stop Complete Event structure

Macro Name	Type	Offset	Description
struct RBLE_ADVSTOP_CMP			
tx_cnt	uint16_t	0	Advertising transmitted count

4.3.12 Advertising Report Event structure

Macro Name	Type	Offset	Description
struct RBLE_ADV_REPORT			
adv_type	uint8_t	0	PDU Type
ch_idx	uint8_t	1	Rx Channel
rssi	int8_t	2	RSSI
adv_addr_type	uint8_t	3	Advertiser’s Device Address Type
adv_addr	RBLE_BD_ADDR	4	Advertiser’s Device Address
data	uint8_t[RBLE_ADVDATA_LEN]	10	Advertising data
data_len	uint8_t	41	the number of Advertising data

4.3.13 Scanning Stop Complete Event structure

Macro Name	Type	Offset	Description
struct RBLE_SCANSTOP_CMP			
status	uint8_t	0	Status
exe_cnt	uint8_t	1	Execution Count

4.3.14 Direct Test Mode Start Complete Event structure

Macro Name	Type	Offset	Description
struct RBLE_DTMSTART_CMP			
status	uint8_t	0	Status
reserved	uint8_t	1	(reserved)

4.3.15 Direct Test Mode Stop Complete Event structure

Macro Name	Type	Offset	Description
struct RBLE_DTMSTOP_CMP			
status	uint8_t	0	Status
dtm_type	uint8_t	1	Direct Test Mode Type
rx_num	uint16_t	2	the number of Received Packets

4.3.16 Event structure

Macro Name	Type	Offset	Description
struct RBLE_EVT			
type	uint8_t	0	Event Type
reserved	uint8_t	1	(reserved)
union param			
advtx	RBLE_ADVTX_IND	2	Advertising Tx Event
reqrx	RBLE_SCANREQ_IND	2	Scan Request Rx Event
advstop	RBLE_ADVSTOP_CMP	2	Advertising Stop Complete Event
advreport	RBLE_ADV_REPORT	2	Advertising Report Event
scanstop	RBLE_SCANSTOP_CMP	2	Scanning Stop Complete Event
dtmstart	RBLE_DTMSTART_CMP	2	Direct Test Mode Start Complete Event
dtmstop	RBLE_DTMSTOP_CMP	2	Direct Test Mode Stop Complete Event

4.4 Functions

Application executes RF Control, Advertising, Scanning and Direct Test Mode by calling Beacon Stack Functions. Beacon Stack Functions are listed in **Table 4-14**.

Table 4-14 Beacon Stack Functions

Function	Operation
R_RF_PowerUp	start supplying power to RF unit.
R_RF_Init	initialize RF unit.
R_RF_PowerDown	stop supplying power to RF unit.
R_BLE_Init	initialize Beacon Stack.
R_BLE_GetEvent	return Beacon Stack event.
R_BLE_GetVersion	return Beacon Stack version.
R_BLE_StartAdvertising	start Advertising.
R_BLE_UpdateAdvInfo	update Advertising Information in Advertising.
R_BLE_UpdateAdvData	update Advertising Data in Advertising.
R_BLE_StopAdvertising	stop Advertising.
R_BLE_StartScanning	start Scanning.
R_BLE_StopScanning	stop Scanning.
R_BLE_StartAdvScan	start Advertising and Scanning Switching
R_BLE_StopAdvScan	stop Advertising and Scanning Switching
R_BLE_SetWhiteList	set White List.
R_BLE_StartDTM	start Direct Test Mode.
R_BLE_StopDTM	stop Direct Test Mode.

Regarding to the specification of Beacon Stack Functions, refer to the following pages.

4.4.1 R_RF_PowerUp

RBLE_STATUS R_RF_PowerUp(uint16_t rf_flg, uint16_t osc_usec);

This function starts supplying power to RF unit.
After calling this function, RF mode is changed to STANDBY_RF.

It is necessary to execute R_RF_Init after executing this function.
This function shall not be called inside interrupt handler.

Parameters:

rf_flg	RF Initialization Configuration Set for RF Operation, RF on-chip DC-DC converter, RF slow clock source, RF mode after RF initialization, clock output. Regarding to the setting, refer to subsection 4.2.3 "RF Initialization Configuration macro".
osc_usec	32MHz Oscillation Stabilization Time (usec) It is necessary to set more than 550usec and recommended time by manufacturer for the connected crystal resonator.

Return:

RBLE_OK	Success
RBLE_ERR_PWRUP	Error : supplying power to RF unit is already started
RBLE_ERR_PARAM	Error : parameter is invalid

Supplementation:

Example code for setting below configuration to argument rf_flg is as shown below.
When [enable both Tx and Rx as RF operation], [use RF on-chip DC-DC converter], [RF slow clock source is RF on-chip oscillator (not execute calibration)], [RF mode after RF Initialization is IDLE_RF] are set to rf_flg.

```
rf_flg = (RFCFG_TXRX | RFCFG_DCDC_ON | RFCFG_INT_32KHZ | RFCFG_MODE_IDLE)
```

4.4.2 R_RF_Init

RBLE_STATUS R_RF_Init(void);

This function initializes RF unit.
After calling this function, RF mode is changed to IDLE_RF or DEEP_SLEEP.

It is necessary to execute R_RF_PowerUp before executing this function.
This function shall not be called inside interrupt handler.

Parameters:

None

Return:

RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_HW_STANDBY	Error : RF unit abnormality in STANDBY_RF
RBLE_ERR_HW_STANDBYRX	Error : RF unit abnormality in STANDBY_RF and enabling Rx
RBLE_ERR_HW_IDLE	Error : RF unit abnormality in IDLE_RF

Supplementation:

While executing RF Initialization, MCU mode is changed to STOP mode temporarily.

4.4.3 R_RF_PowerDown

RBLE_STATUS R_RF_PowerDown(void);

This function stops supplying power to RF unit.
After calling this function, RF mode is changed to POWER_DOWN.

When reusing RF unit after executing this function, it is necessary to execute R_RF_PowerUp and R_RF_Init.

Parameters:

None

Return:

RBLE_OK	Success

4.4.4 R_BLE_Init

void R_BLE_Init(void);	
This function initializes Beacon Stack. No event is notified by calling this function. Before Advertising, Scanning and DTM, it is necessary to execute this function.	
Parameters:	
None	
Return:	
None	

4.4.5 R_BLE_GetEvent

RBLE_EVT* R_BLE_GetEvent(void);		
This function returns Beacon Stack event one by one. When there is one or more events, this function returns other than NULL. When there is no event, this function returns NULL. Regarding to the specification of Beacon Stack events returned by this function, refer to section 4.6 "Events".		
Parameters:		
None		
Return:		
not NULL	returned value is the pointer to the event buffer	
NULL	there is no notified event	
Supplementation:		
Event buffering method is FIFO (First-In First-Out) and the maximum stored number is 31 events. If event buffer is full, Beacon Stack stores new event after deleting oldest event. In order to get all events in event buffer, it is necessary to execute this function repeatedly until returns NULL. If this function returns NULL, it is possible to change MCU mode to STOP mode by calling STOP instruction. Example code for getting events is as shown below.		

```
{
  RBLE_EVT* evt = R_BLE_GetEvent();

  while (evt != NULL)
  {
    switch (evt->type)
    {
      /* add post processing correspond to each event */
    }
    evt = R_BLE_GetEvent();
  }
}
```

4.4.6 R_BLE_GetVersion

RBLE_VERSION R_BLE_GetVersion(void);
This function returns Beacon Stack version.
Parameters:
None
Return:
Version

4.4.7 R_BLE_StartAdvertising

RBLE_STATUS R_BLE_StartAdvertising(uint8_t adv_type, const RBLE_ADV_INFO* adv_info);

This function starts Advertising, and transmits Non-connectable Undirected Advertising (ADV_NONCONN_IND) packets or Scannable Undirected Advertising (ADV_SCAN_IND) packets.
To stop Advertising, limit Advertising count by the argument adv_info->loop_cnt, or execute R_BLE_StartAdvertising.
To update Advertising information in Advertising, execute R_BLE_UpdateAdvInfo.
To update Advertising data in Advertising, execute R_BLE_UpdateAdvData.

Whenever transmit Advertising packet, Beacon Stack notifies RBLE_EVT_ADV_TX_IND event.
Whenever receive Scan Request packet, Beacon Stack notifies RBLE_EVT_SCANREQ_RX_IND event.
When stop Advertising, Beacon Stack notifies RBLE_EVT_ADV_STOP_CMP event.

Regarding to the operation of Advertising and Scan Switching, refer to subsection 4.7.1 "Advertising".

When transmit Scannable Undirected Advertising packet, it is necessary to set RFCFG_TXRX to the argument rf_flg of R_RF_Init.

Parameters:

adv_type	Advertising Type	
	RBLE_PDU_ADV_NONCONN_IND	ADV_NONCONN_IND
	RBLE_PDU_ADV_SCAN_IND	ADV_SCAN_IND

*adv_info	*interval*	Advertising Interval=N×0.625msec when ADV_NONCONN_IND : 1 channel is selected : N=0x0002 to 0xC000(1.250msec to 30.72sec) 2 or 3 channels are selected : N=0x0004 to 0xC000(2.500msec to 30.72sec) when ADV_SCAN_IND : 1 channel is selected : N=0x0004 to 0xC000(2.500msec to 30.72sec) 2 or 3 channels are selected : N=0x000C to 0xC000(7.500msec to 30.72sec) Note that interval range defined by Bluetooth specification is below. N=0x00A0 to 0x4000(100msec to 10.24sec) Advertising Interval Accuracy When RF on-chip oscillator is used : with no calibration is selected : error range about ±50% with calibration is selected : error range about ±6% When XT1 oscillator is used: depends on the accuracy of XT1 oscillator clock
	delay	Whether to add Random Delay to Advertising Interval or not Range of Random Delay is 0 to 10msec by increment of 0.625msec
		true : add random delay
		false : not add random delay
	ch_map	Advertising Channel Regarding to the setting, refer to subsection 4.2.5 "Advertising Channel macro".
	loop_cnt	Advertising Count Limitation(0x01 to 0xFF) When multiple Advertising data are set, each transmitting all Advertising data is counted as one. When set with 0, Advertising is executed indefinitely.
	tx_pwr	Advertising Packet Transmit Power (at ANT pin of RL78/G1D device) Regarding to the setting, refer to subsection 4.2.6 "Tx Power macro".
	own_addr	Own Device Address
	own_addr_type	Own Device Address Type Regarding to the setting, refer to subsection 4.2.4 "Device Address Type macro".
	data_cnt	the number of Advertising Data and Scan Response Data (1 to 10) When use ADV_SCAN_IND, it is necessary to set even number.

RBLE_STATUS R_BLE_StartAdvertising(uint8_t adv_type, const RBLE_ADV_INFO* adv_info);

*data	Advertising Data and Scan Response Data array The number of array is specified by data_cnt. When multiple Advertising data are set, all Advertising data from start to end of the array are transmitted repeatedly. When transmitting ADV_SCAN_IND packets, it is necessary to set ADV_SCAN_IND Data and SCAN_RSP Data alternately.	
evt_permit	Advertising Event Permission Regarding to the setting, refer to subsection 4.2.8 "Advertising Event Permission macro".	
use_wl	When receive Scan Response packet, check whether to filter by White List or not When use White List, it is necessary to set White List by calling R_BLE_SetWhiteList before calling this function. When use ADV_NONCONN_IND, the function does not refer to this parameter.	
	true	use White List
	false	not to use White List

Return:

RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_RFRX	Error : RF unit Rx is disabled
RBLE_ERR_START	Error : Advertising or any function is already started
RBLE_ERR_PARAM	Error : invalid parameter
RBLE_ERR_WL	Error : White List is empty

Supplementation:

Example code for setting Advertising data and Scan Response data when transmitting Scannable Undirected Advertising packet is as shown below. Each Advertising data are transmitted repeatedly in order, and each Scan Response data are transmitted only by when receive Scan Response packet.

```
static RBLE_ADV_DATA adv_scan_data[] =
{
   /* 1st Advertising Data */
   {
      /* Advertising data length */
      27,
      /* Advertising data */
      …
   },
   /* 1st Scan Response Data */
   // After transmitted 1st Advertising Data, this data is transmitted only when received Scan Request
   {
      /* Scan Response data length */
      25,
      /* Scan Response data */
      …
   },
   /* 2nd Advertising Data */
   {
      /* Advertising data length */
      31,
      /* Advertising data */
      0x02, 0x01, 0x04, …
   },
   /* 2nd Scan Response Data */
   // After transmitted 2nd Advertising Data, this data is transmitted only when received Scan Request
   {
      /* Scan Response data length */
      12,
      /* Scan Response data */
      …
   },
};
```

4.4.8 R_BLE_UpdateAdvInfo

RBLE_STATUS R_BLE_UpdateAdvInfo(const RBLE_ADV_INFO* adv_info);			
This function updates Advertising Information in Advertising.			
Parameters:			
	*adv_info	*interval*	refer to the specification of R_BLE_StartAdvertising
		delay	refer to the specification of R_BLE_StartAdvertising
		ch_map	refer to the specification of R_BLE_StartAdvertising
		loop_cnt	refer to the specification of R_BLE_StartAdvertising
		tx_pwr	refer to the specification of R_BLE_StartAdvertising
		own_addr	refer to the specification of R_BLE_StartAdvertising
		own_addr_type	refer to the specification of R_BLE_StartAdvertising
		data_cnt	not referred
		**data*	not referred
		evt_permit	refer to the specification of R_BLE_StartAdvertising
		use_wl	not referred

Return:	
RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_STOP	Error : Advertising is stopped
RBLE_ERR_PARAM	Error : invalid parameter
RBLE_ERR_WL	Error : White List is empty

4.4.9 R_BLE_UpdateAdvData

RBLE_STATUS R_BLE_UpdateAdvData(uint8_t data_idx, const RBLE_ADV_DATA* adv_data);	
This function updates Advertising Data in Advertising.	
Parameters:	
data_idx	Advertising Data Array Index Set within the range of Advertising Data array size, which is specified by the argument adv_info->data of R_BLE_StartAdvertising.
**data*	Advertising Data or Scan Response Data

Return:	
RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_STOP	Error : Advertising is stopped
RBLE_ERR_PARAM	Error : invalid parameter

4.4.10 R_BLE_StopAdvertising

RBLE_STATUS R_BLE_StopAdvertising(void);	
This function stops Advertising. When stop Advertising, Beacon Stack notifies RBLE_EVT_ADV_STOP_CMP event.	
Parameters:	
None	

Return:	
RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_STOP	Error : Advertising is stopped

4.4.11 R_BLE_StartScanning

RBLE_STATUS R_BLE_StartScanning(uint8_t scan_type, const RBLE_SCAN_INFO* scan_info);

This function starts Scanning, and receives Advertising packets by Active Scan or Passive Scan.
To stop Scanning, execute R_BLE_StopScanning.

Whenever receive Advertising packet, Beacon Stack notifies RBLE_EVT_ADVREPORT_IND event.

Regarding to the operation of Scanning, refer to subsection 4.7.2 "Scanning".

When execute this function, it is necessary to set RFCFG_TXRX to the argument rf_flg of R_RF_Init.

Parameters:

scan_type		Scan Type	
		RBLE_SCAN_PASSIVE	Passive Scan
		RBLE_SCAN_ACTIVE	Active Scan
**scan_info*	*interval*	Scan Interval=N×0.625msec 1 channels is selected : N=0x0004 to 0xC000 (2.500msec to 30.72sec) 2 or 3 channel are selected : not referred Note that interval range defined by Bluetooth specification is below. N=0x0004 to 0x4000(2.500msec to 10.24sec) Scan Window=(N-2)×0.625msec Scan Interval Accuracy depends on the accuracy of 32MHz clock.	
	ch_map	Scan Channel Regarding to the setting, refer to subsection 4.2.5 "Advertising Channel macro".	
	tx_pwr	Scan Request Packet Transmit Power (at ANT pin of RL78/G1D device) Regarding to the setting, refer to subsection 4.2.6 "Tx Power macro".	
	own_addr	Own Device Address	
	own_addr_type	Own Device Address Type Regarding to the setting, refer to subsection 4.2.4 "Device Address Type macro".	
	use_wl	Weather to use White List or not When use White List, it is necessary to set White List by calling R_BLE_SetWhiteList before calling this function.	
		true	use White List
		false	not to use White List

Return:

RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_RFRX	Error : RF unit Rx is disabled
RBLE_ERR_START	Error : Scanning or any function is already started
RBLE_ERR_PARAM	Error : invalid parameter
RBLE_ERR_WL	Error : White List is empty

4.4.12 R_BLE_StopScanning

RBLE_STATUS R_BLE_StopScanning(void);

This function stops Scanning.
When stop Scanning, Beacon Stack notifies RBLE_EVT_SCAN_STOP_CMP event.

Parameters:

None

Return:

RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_STOP	Error : Scanning is stopped

4.4.13 R_BLE_StartAdvScan

RBLE_STATUS R_BLE_StartAdvScan(uint8_t adv_type, uint8_t scan_type, const RBLE_ADVSCN_INFO* advscan_info);

This function starts Advertising, and then executes Scannning within specified period.

To stop Advertising and Scan Switching, set Advertising Count Limitation by the argument advscn_info->loop_cnt, or execute R_BLE_StopAdvScan.
To restart Advertising at an optional timing, call this function again.
To update Advertising data, execute R_BLE_UpdateAdvData.

Whenever transmit Advertising packet, Beacon Stack notifies RBLE_EVT_ADV_TX_IND event.
Whenever receive Advertising packet, Beacon Stack notifies RBLE_EVT_ADVREPORT_IND event.
When stop Scanning, Beacon Stack notifies RBLE_EVT_SCAN_STOP_CMP event.

Regarding to the operation of Advertising and Scan Switching, refer to subsection 4.7.3 "Advertising and Scan Switching".

When use this function, it is necessary to set RFCFG_TXRX to the argument rf_flg of R_RF_Init.

Parameters:

adv_type		Advertising Type	
		RBLE_PDU_ADV_NONCONN_IND	ADV_NONCONN_IND
		RBLE_PDU_NO_TYPE	not execute Advertising
scan_type		Scan Type	
		RBLE_SCAN_PASSIVE	Passive Scan
*advscn_info	*interval*	Advertising Interval=N×0.625msec N=0x0004 to 0xC000(2.500msec to 30.72sec) Note that Advertising interval range defined by Bluetooth specification is below. N=0x00A0 to 0x4000(100msec to 10.24sec) When loop_cnt is 1, this parameter is not referred. Advertising Interval Accuracy When RF on-chip oscillator is used : with no calibration is selected : error range about ±50% with calibration is selected : error range about ±6% When XT1 oscillator is used: depends on the accuracy of XT1 oscillator clock	
	delay	Whether to add Random Delay to Advertising Interval or not Range of Random Delay is 0 to 10msec by increment of 0.625msec	
		true	add random delay
		false	not add random delay
	ch_map	Advertising and Scan Channel Map Only one channel can be selected Regarding to the setting, refer to subsection 4.2.5 "Advertising Channel macro".	
	loop_cnt	Advertising Count Limitation (0x01 to 0xFF) When set with 0, Advertising and Scan Switching is executed indefinitely. When adv_type is RBLE_PDU_NO_TYPE, set 1 to this parameter.	
	tx_pwr	Packet Transmit Power (at ANT pin of RL78/G1D device) Regarding to the setting, refer to subsection 4.2.6 "Tx Power macro".	
	own_addr	Own Device Address	
	own_addr_type	Own Device Address Type Regarding to the setting, refer to subsection 4.2.4 "Device Address Type macro".	
	data_cnt	the number of Advertising Data (1 only) When adv_type is RBLE_PDU_NO_TYPE, this parameter is not referred	
	**data*	Advertising Data When adv_type is RBLE_PDU_NO_TYPE, this parameter is not referred	
	evt_permit	Advertising Event Permission Regarding to the setting, refer to subsection 4.2.8 "Advertising Event Permission macro".	

RBLE_STATUS R_BLE_StartAdvScan(uint8_t adv_type, uint8_t scan_type, const RBLE_ADVSCN_INFO* advscan_info);				
		offset	Scan Window Offset=N×0.625msec N=0x01 to 0x08(0.625msec to 5.000msec) time from the start of Advertising to the start of Scan Note that offset shall be less than or equal to (interval - 3). When adv_type is RBLE_PDU_NO_TYPE, this parameter is not referred	
		window	Scan Window Size=N×0.625msec N=0x0001 to 0xC000(0.625msec to 30.72sec) Note that if window shall be greater than or equal to (interval - offset), Scanning stops suspended before next Advertising. When loop_cnt is 1, by setting 0 to this parameter, Scan Window Size is indefinite.	
		continuous	Continuous Execution Flag Regarding to the operation of Continuous Execution, refer to subsection 4.7.3(3) "Continuous Execution of Advertising and Scan Switching".	
			false	not execute continuously After Scanning, RF state goes to DEEP_SLEEP as soon as possible. If call this function after Scanning, Beacon Stack needs to executes resuming processing before starting Advertising.
			true	execute continuously After Scanning, RF state goes to IDLE_RF and then goes to DEEP_SLEEP after 10msec. If call this function before 10msec expires, Beacon Stack starts Advertising without resuming processing.
		use_wl	In Scanning, check whether to filter by White List or not When use White List, it is necessary to set White List by calling R_BLE_SetWhiteList before calling this function.	
			true	use White List
			false	not to use White List

Return:

RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_RFRX	Error : RF unit Rx is disabled
RBLE_ERR_START	Error : Other than Scanning is already started
RBLE_ERR_PARAM	Error : invalid parameter
RBLE_ERR_WL	Error : White List is empty

4.4.14 R_BLE_StopAdvScan

RBLE_STATUS R_BLE_StopAdvScan(void);

This function stops Advertising and Scan Switching.
When stop Advertising and Scan Switching, Beacon Stack notifies RBLE_EVT_SCAN_STOP_CMP event.

Parameters:

None

Return:

RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_STOP	Error : Advertising and Scan Switching is stopped

4.4.15 R_BLE_SetWhiteList

RBLE_STATUS R_BLE_SetWhiteList(uint8_t wl_cnt, const RBLE_DEV_INFO* wl);		
This function sets White List.		
Parameters:		
	wl_cnt	the number of Devices in White List (0 to 16)
	**wl*	White List
Return:		
	RBLE_OK	Success
	RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
	RBLE_ERR_START	Error : any function is already started
	RBLE_ERR_PARAM	Error : invalid parameter

4.4.16 R_BLE_StartDTM

RBLE_STATUS R_BLE_StartDTM(uint8_t dtm_type, const RBLE_DTM_INFO* dtm_info);

This function starts Direct Test Mode, and executes RF Receiver Test or RF Transmitter Test.
To stop RF Receiver Test, execute R_BLE_StopDTM.
To stop RF Transmitter Test, limit transmitting packets by the argument dtm_info->tx_num, or execute R_BLE_StopDTM.

When start Direct Test Mode, Beacon Stack notifies RBLE_EVT_DTM_START_CMP event.

Regarding to the operation of Direct Test Mode, refer to subsection 4.7.4 "Direct Test Mode".

When execute RF Receiver Test, it is necessary to set RFCFG_TXRX to the argument rf_flg of R_RF_Init.

Parameters:

dtm_type	Direct Test Mode Type	
	RBLE_DTM_RX	RF Receiver Test
	RBLE_DTM_TX	RF Transmitter Test

*dtm_info	*mod*	Modulation Configuration Regarding to the setting, refer to subsection 4.2.11 "Direct Test Mode Modulation Configuration macro".
	freq	Frequency : (2*N+2402) MHz N=0x00(2402MHz) to 0x27(2480MHz)
	tx_pwr	Packet Transmit Power of RF Transmitter Test (at ANT pin of RL78/G1D) Regarding to the setting, refer to subsection 4.2.6 "Tx Power macro". When start RF Receiver Test, the function does not refer to this parameter.
	tx_datalen	Packet Data Length of RF Transmitter Test (0 to 37byte) When start RF Receiver Test, the function does not refer to this parameter.
	tx_payload	Packet Payload Type of RF Transmitter Test Regarding to the setting, refer to subsection 4.2.10 "Direct Test Mode Type macro". When start RF Receiver Test, the function does not refer to this parameter.
	tx_num	the number of transmitting RF Test Packets (0x0001 to 0xFFFF) When set with 0, RF Transmitter Test is executed indefinitely When start RF Receiver Test, the function does not refer to this parameter.

Return:

RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_RFRX	Error : RF unit Rx is disabled
RBLE_ERR_START	Error : Direct Test Mode or any function is already started
RBLE_ERR_PARAM	Error : invalid parameter

4.4.17 R_BLE_StopDTM

RBLE_STATUS R_BLE_StopDTM(void);

This function stops Direct Test Mode.
When stop Direct Test Mode, Beacon Stack notifies RBLE_EVT_DTM_STOP_CMP event.

Parameters:

None

Return:

RBLE_OK	Success
RBLE_ERR_PWRDOWN	Error : supplying power to RF unit is stopped
RBLE_ERR_STOP	Error : Direct Test Mode is stopped

4.5 Interrupt Processing

When execute RF Control, Advertising, Scanning or Direct Test Mode, Beacon Stack executes processing by interrupts. Beacon Stack Interrupt Processing are listed in **Table 4-15**.

Table 4-15 Beacon Stack Interrupt Processing

Interrupt Processing	Interrupt Source	Operation
R_INTRF_isr	INTRF	Tx or Rx processing and RF control processing
R_INTTM00_isr	INTTM00	Wait processing
R_INTDMA2_isr	INTDMA2	RF register transfer processing.
R_INTDMA3_isr	INTDMA3	RF register transfer processing.

Application shall implement interrupt handlers for executing Beacon Stack Interrupt Processing when interrupt occurs.

Example code for implementation of Beacon Stack Interrupt Processing is as shown below.

Example Code for implementation of Beacon Stack Interrupt Processing

```
#pragma interrupt rf_interrupt   (vect=INTRF)
#pragma interrupt dma2_interrupt (vect=INTDMA2)
#pragma interrupt dma3_interrupt (vect=INTDMA3)
#pragma interrupt tm00_interrupt (vect=INTTM00)

__interrupt static void rf_interrupt(void)
{
   R_INTRF_isr();
}

__interrupt static void dma2_interrupt(void)
{
   R_INTDMA2_isr();
}

__interrupt static void dma3_interrupt(void)
{
   R_INTDMA3_isr();
}

__interrupt static void tm00_interrupt(void)
{
   R_INTTM00_isr();
}
```

Regarding to the specification of Beacon Stack Interrupt Processing, refer to the following page.

4.5.1 R_INTRF_isr

void R_INTRF_isr(void);
By occurring RF interrupt (INTRF), this function executes Tx or Rx processing and RF control processing. In order to execute this function by occurring RF interrupt, application shall define INRF interrupt handler which calls this function and register to vector table.
Parameters: None
Return: None

4.5.2 R_INTTM00_isr

void R_INTTM00_isr(void);
By occurring TM00 expired interrupt (INTTM00), this function executes wait processing. In order to execute this function by occurring TM00 expired interrupt, application shall define INTTM00 interrupt handler which calls this function and register to vector table.
Parameters: None
Return: None

4.5.3 R_INTDMA2_isr

void R_INTDMA2_isr(void);
By calling DMA2 transfer end interrupt (NTDMA2), this function executes RF register transfer processing. In order to execute this function by occurring DMA2 transfer end interrupt, application shall define INTDMA2 interrupt handler which calls this function and register to vector table.
Parameters: None
Return: None

4.5.4 R_INTDMA3_isr

void R_INTDMA3_isr(void);
By calling DMA3 transfer end interrupt (NTDMA3), this function executes RF register transfer processing. In order to execute this function by occurring DMA3 transfer end interrupt, application shall define INTDMA3 interrupt handler which calls this function and register to vector table.
Parameters: None
Return: None

4.6 Events

Beacon stack notifies application by Events that Advertising, Scanning or Direct Test Mode is started, stopped, transmitted or received. Application gets Event by executing R_BLE_GetEvent. Beacon Stack Events are listed in **Table 4-16**.

Table 4-16 Beacon Stack Events

Event	Timing Notified
RBLE_EVT_ADV_TX_IND	When Advertising packet is transmitted in Advertising
RBLE_EVT_SCANREQ_RX_IND	When Scan Request is received in Advertising
RBLE_EVT_ADV_STOP_CMP	When Advertising is stopped completely
RBLE_EVT_ADVREPORT_IND	When Advertising packet or Scan Response packet received in Scanning
RBLE_EVT_SCAN_STOP_CMP	When Scanning is stopped completely
RBLE_EVT_DTM_START_CMP	When Direct Test Mode is started completely
RBLE_EVT_DTM_STOP_CMP	When Direct Test Mode is stopped completely

Regarding to the specification of Beacon Stack Events, refer to the following pages.

Regarding to implementation for getting Beacon Stack Event, refer to subsection 4.4.5 "R_BLE_GetEvent" in this document.

4.6.1 RBLE_EVT_ADV_TX_IND

RBLE_EVT_ADV_TX_IND
This event notifies that Advertising packet is transmitted in Advertising. This event occurs after Advertising packet is transmitted to all channels. Whether to notify this event or not is set by the argument adv_info->evt_permit of R_BLE_StartAdvertising. In order to get an event, use R_BLE_GetEvent. Parameter tx_cnt is the count of transmission since R_BLE_StartAdvertising is called. The number of transmitted packets can be calculated by the parameter tx_cnt. [the number of transmitted packets] = [the number of transmitting channel] × tx_cnt
Parameters:

tx_cnt	Advertising Transmitted Count
data_idx	Transmitted Advertising Data Array Index

4.6.2 RBLE_EVT_SCANREQ_RX_IND

RBLE_EVT_SCANREQ_RX_IND
This event notifies that Scan Request is received in Advertising. This event occurs only when transmitting Scannable Undirected Advertising packet. Whether to notify this event or not is set by the argument adv_info->evt_permit of R_BLE_StartAdvertising. In order to get an event, use R_BLE_GetEvent.
Parameters:

ch_idx	Scan Request Packet Received Channel
rssi	Scan Request Packet RSSI
dev_addr	Peer Device's Device Address

4.6.3 RBLE_EVT_ADV_STOP_CMP

RBLE_EVT_ADV_STOP_CMP
This event notifies that Advertising is stopped completely. This event occurs after Advertising is stopped by calling R_BLE_StopAdvertising or by executing the number of transmission, which is specified by the argument adv_info->loop_cnt of R_BLE_StartAdvertising. Whether to notify this event or not is set by the argument adv_info->evt_permit of R_BLE_StartAdvertising. In order to get an event, use R_BLE_GetEvent. Parameter tx_cnt is the count of transmission since R_BLE_StartAdvertising is called. The number of transmitted packets can be calculated by the parameter tx_cnt. [the number of transmitted packets] = [the number of transmitting channel] × tx_cnt
Parameters:

tx_cnt	Advertising Transmitted Count

4.6.4 RBLE_EVT_ADVREPORT_IND

RBLE_EVT_ADVREPORT_IND	
This event reports the information of Advertising packet or Scan Response packet received in Scanning. In order to get an event, use R_BLE_GetEvent.	
Parameters:	
adv_type	Received Packet Type Regarding to the returned value, refer to subsection 4.2.7 "PDU Type macro".
ch_idx	Packet Received Channel
rssi	Packet RSSI
adv_addr_type	Device Address Type Regarding to the returned value, refer to subsection 4.2.4 "Device Address Type macro".
adv_addr	Device Address
data	Received Advertising Data or Received Scan Response Data
data_len	Received Data Length

4.6.5 RBLE_EVT_SCAN_STOP_CMP

RBLE_EVT_SCAN_STOP_CMP	
This event notifies that Scanning is stopped completely. This event occurs after Scanning is stopped by calling R_BLE_StopScanning, or stopping Advertising and Scan Switching. In order to get an event, use R_BLE_GetEvent.	
Parameters:	
status	Status Regarding to the returned value, refer to subsection 4.2.1 "Status macro".
exe_cnt	Execution Count In Scanning : This parameter indicates the execution count of R_BLE_StartScanning. In Advertising and Scan Switching : This parameter indicates the execution count of R_BLE_StartAdvScan.

4.6.6 RBLE_EVT_DTM_START_CMP

RBLE_EVT_DTM_START_CMP	
This event notifies that Direct Test Mode is started completely. This event occurs after Direct Test Mode is started by calling R_BLE_StartDTM. In order to get an event, use R_BLE_GetEvent.	
Parameters:	
status	Status Regarding to the returned value, refer to subsection 4.2.1 "Status macro".

4.6.7 RBLE_EVT_DTM_STOP_CMP

RBLE_EVT_DTM_STOP_CMP	
This event notifies that Direct Test Mode is stopped completely. This event occurs after Direct Test Mode is stopped by calling R_BLE_StopDTM or by transmitting the number of packets, which is specified by the argument dtm_info->tx_num of R_BLE_StartDTM. In order to get an event, use R_BLE_GetEvent.	
Parameters:	
status	Status Regarding to the returned value, refer to subsection 4.2.1 "Status macro".
dtm_type	Direct Test Mode Type Regarding to the returned value, refer to subsection 4.2.10 "Direct Test Mode Type macro".
rx_num	the number of Receiver Packets

4.7 Operation

4.7.1 Advertising

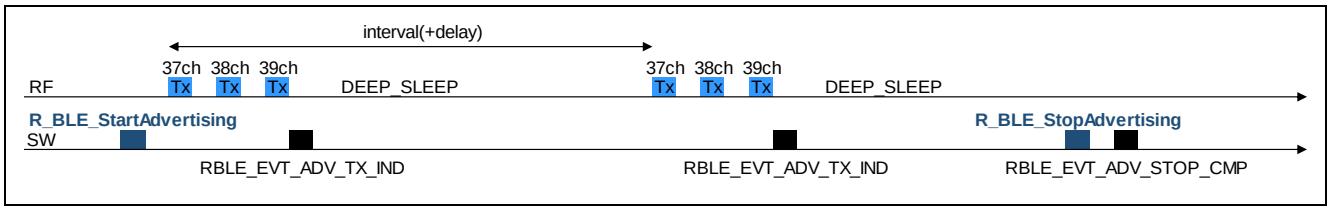
(1) Advertising, operating indefinitely

By calling R_BLE_StartAdvertising with 0 in argument adv_info->loop_cnt, Beacon Stack transmits Advertising packets indefinitely. Beacon Stack transmits Advertising pakets and then notifies RBLE_EVT_ADV_TX_IND event in the period specified by argument adv_info->interval. By calling R_BLE_StopAdvertising, Beacon Stack stops Advertising and then notifies RBLE_EVT_ADV_STOP_CMP event.

Figure 4-1 shows operation in the case of below example parameters for R_BLE_StartAdvertising.

- adv_type = RBLE_PDU_ADV_NONCONN_IND
- adv_info->loop_cnt = 0;
- adv_info->ch_map = RBLE_ADV_ALL_CHANNELS;
- adv_info->permit = RBLE_EVT_PERMIT_ADV_ALL;

Figure 4-1 Advertising, operating indefinitely



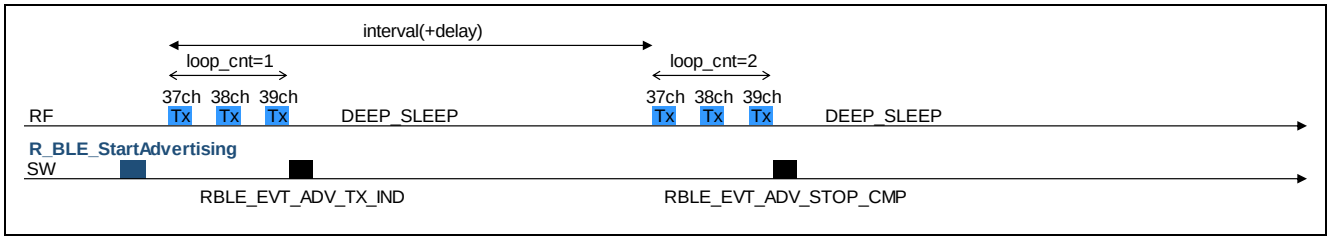
(2) Advertising, oprating with Count Limitation

By calling R_BLE_StartAdvertising with value from 0x01 to 0xFF in argument adv_info->loop_cnt, Beacon Stack transmits Advertising packets in the specified number of times only. Beacon Stack transmits Advertising pakets and then notifies RBLE_EVT_ADV_TX_IND event in the period specified by argument adv_info->interval. After transmit Advertising packets in the specified number of times, Beacon Stack stops Advertising automatically and then notifies RBLE_EVT_ADV_STOP_CMP event. If need to stop operation before executing in the specified number of times, call R_BLE_StopAdvertising.

Figure 4-2 shows operation in the case of below example parameters for R_BLE_StartAdvertising.

- adv_type = RBLE_PDU_ADV_NONCONN_IND
- adv_info->loop_cnt = 2;
- adv_info->ch_map = RBLE_ADV_ALL_CHANNELS;
- adv_info->permit = RBLE_EVT_PERMIT_ADV_ALL;

Figure 4-2 Advertising, operating with Count Limitation



4.7.2 Scanning

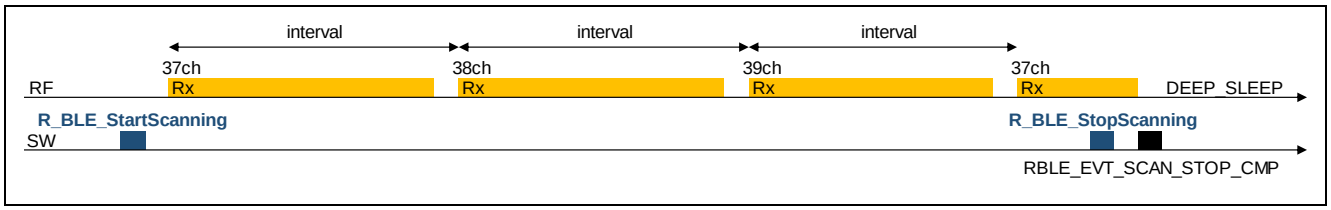
(1) Multiple Channels Scanning

By calling R_BLE_StartScanning with 2 or 3 channels in argument scan_info->ch_map, Beacon Stack executes Scanning for each channel in the period specified by argument scan_info->interaval. By calling R_BLE_StopScanning, Beacon Stack stops Scanning and then notifies RBLE_EVT_SCAN_STOP_CMP event.

Figure 4-3 shows operation in the case of below example parameters for R_BLE_StartScanning.

- scan_type = RBLE_SCAN_PASSIVE;
- scan_info->ch_map = RBLE_ADV_ALL_CHANNELS;

Figure 4-3 Multiple Channels Scanning



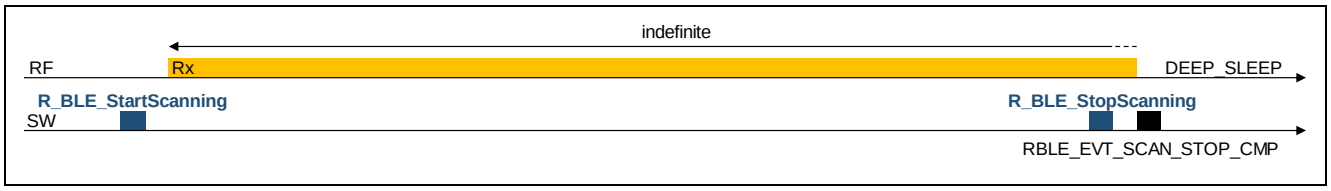
(2) 1 Channel Scanning

By calling R_BLE_StartScanning with only 1channel in argument scan_info->ch_map, Beacon Stack executes Scanning indenitely. By calling R_BLE_StopScanning, Beacon Stack stops Scanning and then notifies RBLE_EVT_SCAN_STOP_CMP event.

Figure 4-4 shows operation in the case of below example parameters for R_BLE_StartScanning.

- scan_type = RBLE_SCAN_PASSIVE;
- scan_info->ch_map = RBLE_ADV_CHANNEL_37;

Figure 4-4 1 Channel Scanning



4.7.3 Advertising and Scan Switching

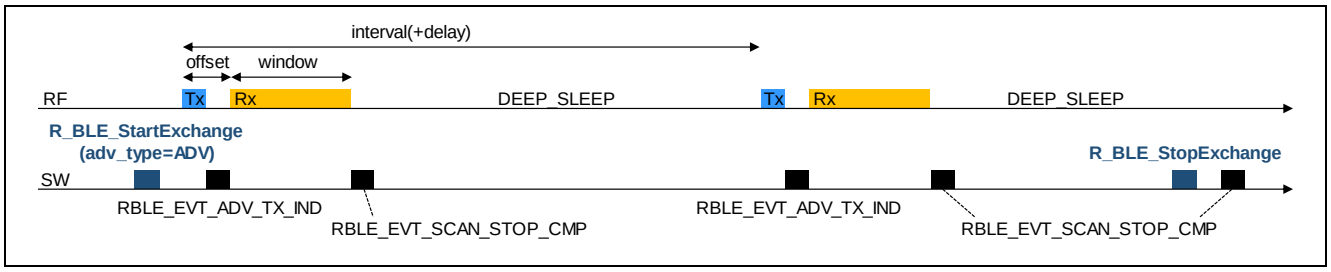
(1) Advertising and Scan Switching, operating indefinitely

By calling R_BLE_StartAdvScan with RBLE_PDU_ADV_NONCONN_IND in argument adv_type and 0 in argument adv_info->loop_cnt, Beacon Stack executes Advertising and Scan Switching indefinitely. Beacon Stack transmits Advertising packet and executes Scanning alternately. Beacon Stack starts Scanning from offset time after Advertising specified by argument advscn_info->offset and within argument advscn_info->window. By calling R_BLE_StopAdvScan, Beacon Stack stops operation and then notifies RBLE_EVT_SCAN_STOP_CMP event.

Figure 4-5 shows operation in the case of below example parameters for R_BLE_StartAdvScan.

- adv_type = RBLE_PDU_ADV_NONCONN_IND;
- advscn_info->loop_cnt = 0;
- advscn_info->continuous = false;
- advscn_info->permit = RBLE_EVT_PERMIT_ADV_ALL;

Figure 4-5 Advertising and Scan Switching, operating indefinitely



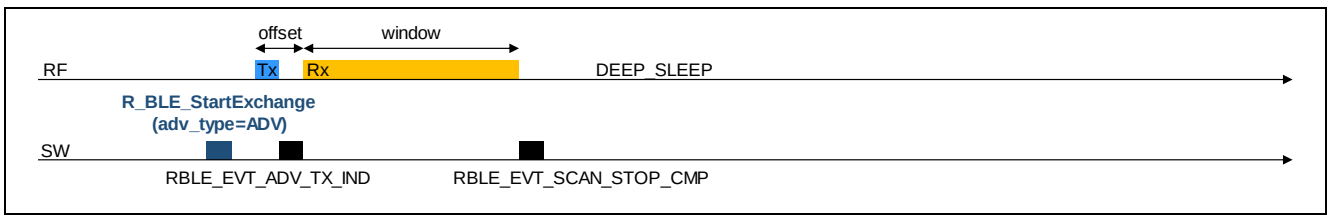
(2) Advertising and Scan Switching, operating with Count Limitation

By calling R_BLE_StartAdvScan with RBLE_PDU_ADV_NONCONN_IND in argument adv_type and value from 0x01 to 0xFF in argument advscn_info->loop_cnt, Beacon Stack executes Advertising and Scan Switching in the period specified by argument advscn_info->interval and in the specified number of times only. Beacon Stack transmits Advertising packet and executes Scanning alternately. After executes Advertising and Scan Switching in the specified number of times, Beacon Stack stops operation automatically. If need to stop operation before executing in the specified number of times, call R_BLE_StopAdvScan.

Figure 4-6 shows operation in the case of below example parameters for R_BLE_StartAdvScan.

- adv_type = RBLE_PDU_ADV_NONCONN_IND;
- advscn_info->loop_cnt = 1;
- advscn_info->continuous = false;
- advscn_info->permit = RBLE_EVT_PERMIT_ADV_ALL;

Figure 4-6 Advertising and Scan Switching, operating with Count Limitation



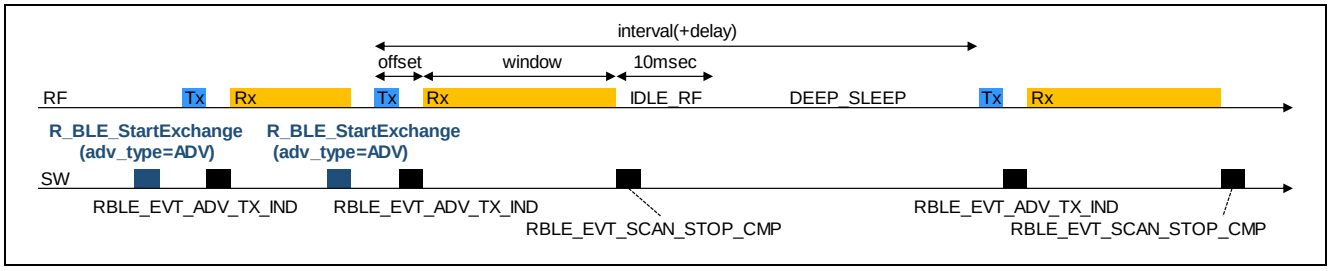
(3) Continuous Execution of Advertising and Scan Switching

By calling R_BLE_StartAdvScan with RBLE_PDU_ADV_NONCONN_IND in argument adv_type, 0 in argument adv_info->loop_cnt, and true in argument advscn_info->continuous, Beacon Stack executes Advertising and Scan Switching indefinitely. Beacon Stack transmits Advertising packet and executes Scanning alternately, and keeps IDLE_RF state 10msec after Scanning. By calling R_BLE_StartAdvScan again, Beacon Stack re-transmits Advertising packet as soon as possible. By calling R_BLE_StopAdvScan, Beacon Stack stops operation and then notifies RBLE_EVT_SCAN_STOP_CMP event.

Figure 4-7 shows operation in the case of below example parameters for R_BLE_StartAdvScan.

- adv_type = RBLE_PDU_ADV_NONCONN_IND;
- advscn_info->loop_cnt = 0;
- advscn_info->continuous = true;
- advscn_info->permit = RBLE_EVT_PERMIT_ADV_ALL;

Figure 4-7 Continuous Execution of Advertising and Scan Switching



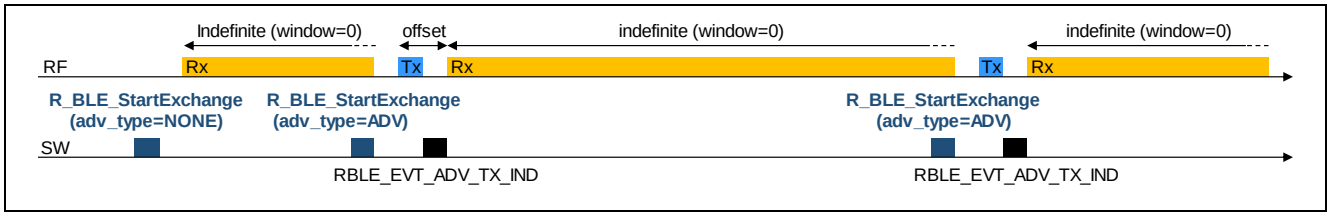
(4) Advertising and Scan Switching, starting with Scanning

By calling R_BLE_StartAdvScan with RBLE_PDU_NO_TYPE in argument adv_type, 1 in argument advscn_info->loop_cnt, and 0 in argument advscn_info->window, Beacon Stack executes Scanning indefinitely. By calling R_BLE_StartAdvScan with RBLE_PDU_ADV_NONCONN_IND in argument adv_type, 1 in argument advscn_info->loop_cnt, and 0 in argument advscn_info->window, Beacon Stack executes transmits one Advertising packet and re-executes Scanning indefinitely.

Figure 4-8 shows operation in the case of below example parameters for R_BLE_StartAdvScan.

- adv_type = RBLE_PDU_NO_TYPE; for the first time only
- adv_type = RBLE_PDU_ADV_NONCONN_IND; for transmitting Advertising packet when receive packet
- advscn_info->loop_cnt = 0;
- advscn_info->continuous = false;
- advscn_info->permit = RBLE_EVT_PERMIT_ADV_ALL;

Figure 4-8 Advertising and Scan Switching, starting with Scanning



4.7.4 Direct Test Mode

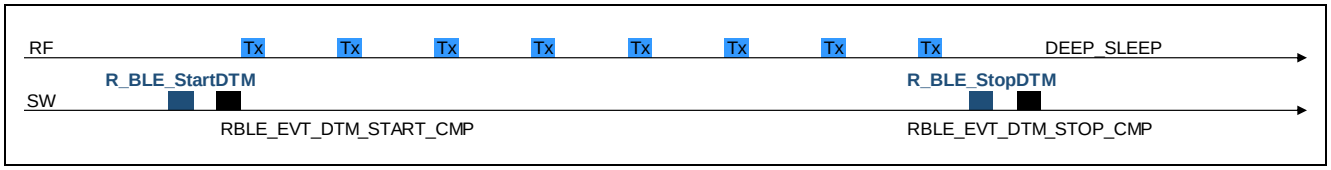
(1) RF Transmitter Test Stopped by calling R_BLE_StopDTM

By calling R_BLE_StartDTM with RBLE_DTM_TX in argument dtm_type and 0 in argument dtm_info->tx_num, Beacon Stack starts RF Transmitter Test and transmits packets indefinitely. By calling R_BLE_StopDTM, Beacon Stack stops RF Transmitter Test and then notifies RBLE_EVT_DTM_STOP_CMP event.

Figure 4-9 shows operation in the case of below example parameters for R_BLE_StartDTM.

- dtm_type = RBLE_DTM_TX;
- dtm_info->mod = RBLE_DTM_MODON_PACKET;
- dtm_info->tx_num = 0;

Figure 4-9 RF Transmitter Test Stopped by calling R_BLE_StopDTM



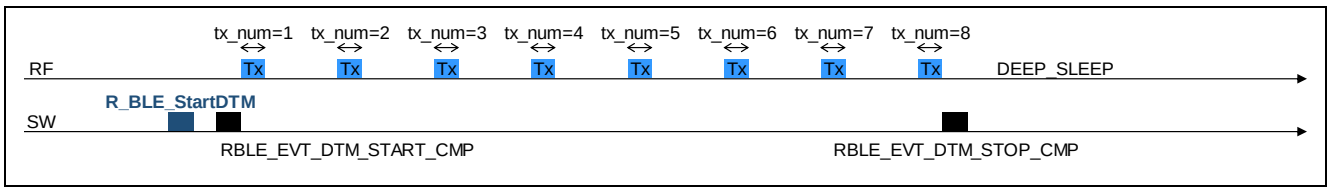
(2) RF Transmitter Test Stopped by Limiting Transmit Count

By calling R_BLE_StartDTM with RBLE_DTM_TX in argument dtm_type and value from 0x01 to 0xFF in argument dtm_info->tx_num, Beacon Stack starts RF Transmitter Test and transmits only the specified number of packets. After transmit the specified number of packets, Beacon Stack stops RF Transmitter Test automatically and then notifies RBLE_EVT_DTM_STOP_CMP event.

Figure 4-10 shows operation in the case of below example parameters for R_BLE_StartDTM.

- dtm_type = RBLE_DTM_TX;
- dtm_info->mod = RBLE_DTM_MODON_PACKET;
- dtm_info->tx_num = 8;

Figure 4-10 RF Transmitter Test Stopped by Limiting Transmit Count



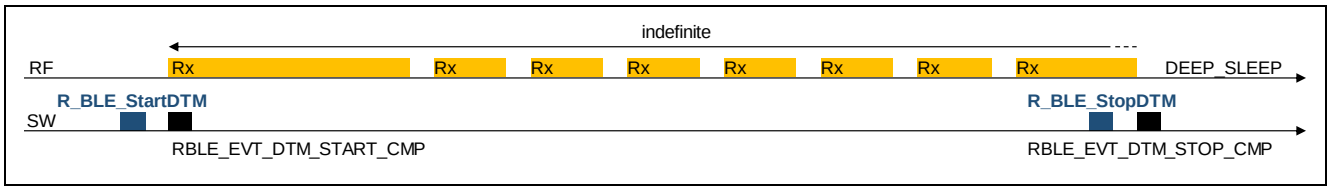
(3) RF Receiver Test

By calling R_BLE_StartDTM with RBLE_DTM_RX in argument dtm_type, Beacon Stack starts RF Receiver Test indefinitely. By calling R_BLE_StopDTM, Beacon Stack stops RF Receiver Test and then notifies RBLE_EVT_DTM_START_CMP event.

Figure 4-11 shows operation in the case of below example parameters for R_BLE_StartDTM.

- dtm_type = RBLE_DTM_RX;
- dtm_info->mod = RBLE_DTM_MODON_PACKET;

Figure 4-11 RF Receiver Test



Revision History of Preceding Editions

Rev.	Date	Summary
2.00	Oct 26, 2016	Section which describes Specification, Function and API of Beacon Stack are separated from RL78/G1D Beacon Stack Application Note and issued as this user's manual. 3.3.Scanning Function : new added 3.4.DTM Function : new added 4.2.7.PDU Type macro : new added 4.2.9.Scan Type macro : new added 4.2.10.Direct Test Mode Type macro : new added 4.2.11. Direct Test Mode Modulation Configuration macro : new added 4.2.12. Direct Test Mode Payload macro : new added 4.3.6.Scanning Information structure : new added 4.3.7.Direct Test Mode Information structure : new added 4.3.9.Scan Request Rx event structure : new added 4.3.11.Advertising Report Event structure : new added 4.3.12.Scanning Stop Complete Event structure : new added 4.3.13.Direct Test Mode Start Complete Event structure : new added 4.3.14.Direct Test Mode Stop Complete Event structure : new added 4.4.15.R_BLE_StartScanning : new added 4.4.16.R_BLE_StopScanning : new added 4.4.17.R_BLE_SetWhiteList : new added 4.4.18.R_BLE_StartDTM : new added 4.4.19.R_BLE_StopDTM : new added 4.5.2.RBLE_EVT_SCANREQ_RX_IND : new added 4.5.4.RBLE_EVT_ADVREPORT_IND : new added 4.5.5.RBLE_EVT_SCAN_STOP_CMP : new added 4.5.6.RBLE_EVT_DTM_START_CMP : new added 4.5.7.RBLE_EVT_DTM_STOP_CMP : new added
2.10	Mar 09, 2017	1.1.4.Advertising and Scan Switching Function : new added 2.2.Compiler : compiler is updated 3.4.Advertising and Scan Switching Function : new added 4.2.7.PDU Type macro : RBLE_PDU_NO_TYPE macro is added 4.3.7.Advertising and Scan Switching Information structure : new added 4.3.13.Scanning Stop Complete Event structure : exe_cnt parameter is added 4.4.13.R_BLE_StartAdvScan : new added 4.4.14.R_BLE_StopAdvScan : new added 4.6.5.RBLE_EVT_SCAN_STOP_CMP : exe_cnt parameter is added 4.7.Operation : new added

RL78/G1D Beacon Stack
User's Manual

Publication Date : Rev.2.00 Oct 26, 2016
Rev.2.10 Mar 09, 2017

Published by : Renesas Electronics Corporation

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.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

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: +44-1628-585-100, Fax: +44-1628-585-900

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, P. R. China 200333
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 II Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 4.0

RL78/G1D Beacon Stack

Renesas Electronics Corporation

R01UW0171EJ0210