AN12561

Kinetis KW38 Bluetooth Low Energy 5.0 Advertising Extension Enablement in Legacy Bluetooth Low Energy Examples

Rev. 0 — 29 August 2019

Application Note

Contents

1 Introduction..1

2 Hardware setup..................................1

2.1 Use FRDM development board.......................................1

2.2 Program the FRDM board.......2

3 Software enablement..........................2

3.1 Bluetooth LE shell demo example..................................2

3.2 Enable advertising extension in legacy Bluetooth LE examples...........2

3.2.1 Include the advertising extension library..........2

3.2.2 Configure the advertising extension.....................3

3.2.3 Configure the advertising data...........5

3.2.4 Start extended advertising..................5

3.2.5 Testing.....................6

4 Revision history..................................7

1 Introduction

This document provides instructions on how to enable new Bluetooth Low Enegy 5.0 Advertising extension features in a legacy Bluetooth Low Energy example, which means demo examples such as Bluetooth Low Energy beacon, and other examples except BLE_shell, in a software application point of view.

NOTE

This user guide is only available for Kinetis KW37/38/39 devices.

The Bluetooth LE 5.0 specification introduced new features of Advertising Extensions, such as extended advertising, periodic advertising, Long Range, High Speed etc.

Kinetis KW37/38/39 MCU series are radio wireless MCUs that support Bluetooth LE 5.0 protocol.

The prerequisites for understanding this document are that the reader has basic knowledge about Bluetooth LE protocol, as well as basic knowledge about Arm MCU architecture and radio communication basics.

2 Hardware setup

- Use FRDM development board
- Program the FRDM board

2.1 Use FRDM development board

This document takes the FRDM-KW38 board as example

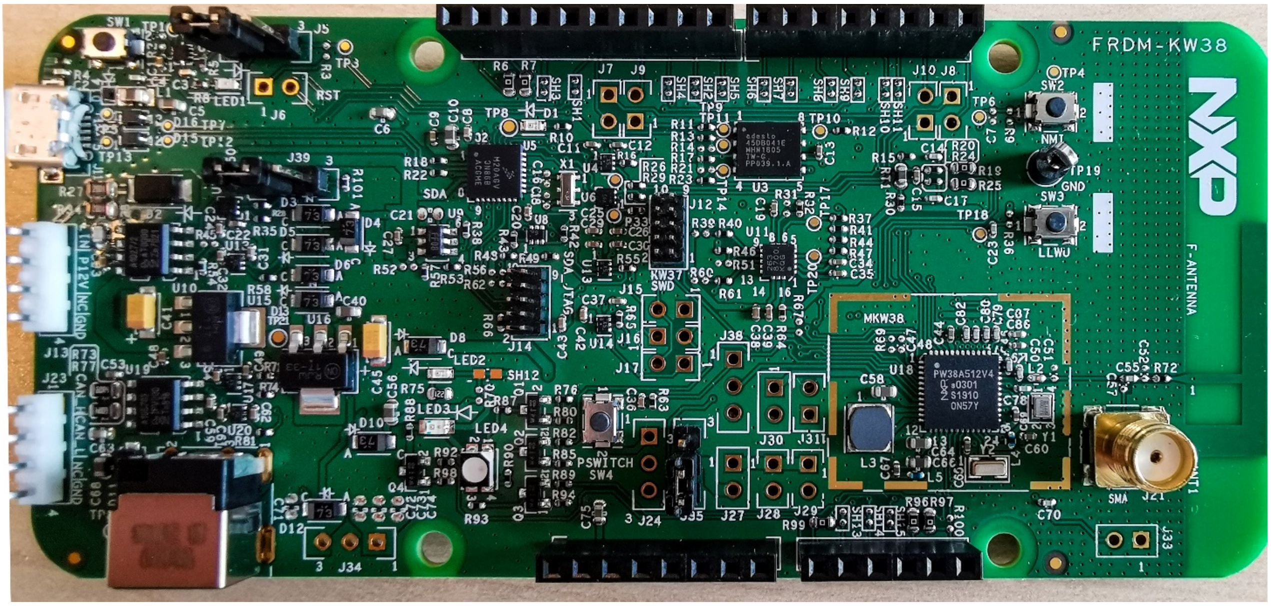


Figure 1. FRDM-KW38 development board

The FRDM boards offer possibilities to change certain configurations of the MCU and offers a debug interface to program it. To enable the Bluetooth LE 5.0 features, no hardware change is required.

2.2 Program the FRDM board

The demo application example supports the IAR embedded workbench for Arm. The FRDM board supports OpenSDA. After choosing CMSIS-DAP as debugger interface, user's can build and program the board by simply connecting the board to the PC via a USB cable. In addition, simply drag-and-drop the generated .bin file to the Kinetis device (displayed as a USB flash driver) is also an option to program the board.

3 Software enablement

- Bluetooth LE shell demo example
- Enable advertising extension in legacy Bluetooth LE examples

3.1 Bluetooth LE shell demo example

The KW38 SDK comes with a Bluetooth LE shell demo example which supports and demonstrate natively the Advertising Extension feature. For details on how to use this example to configure and advertise with new features, see the Bluetooth LE Demo Application User’s Guide.

3.2 Enable advertising extension in legacy Bluetooth LE examples

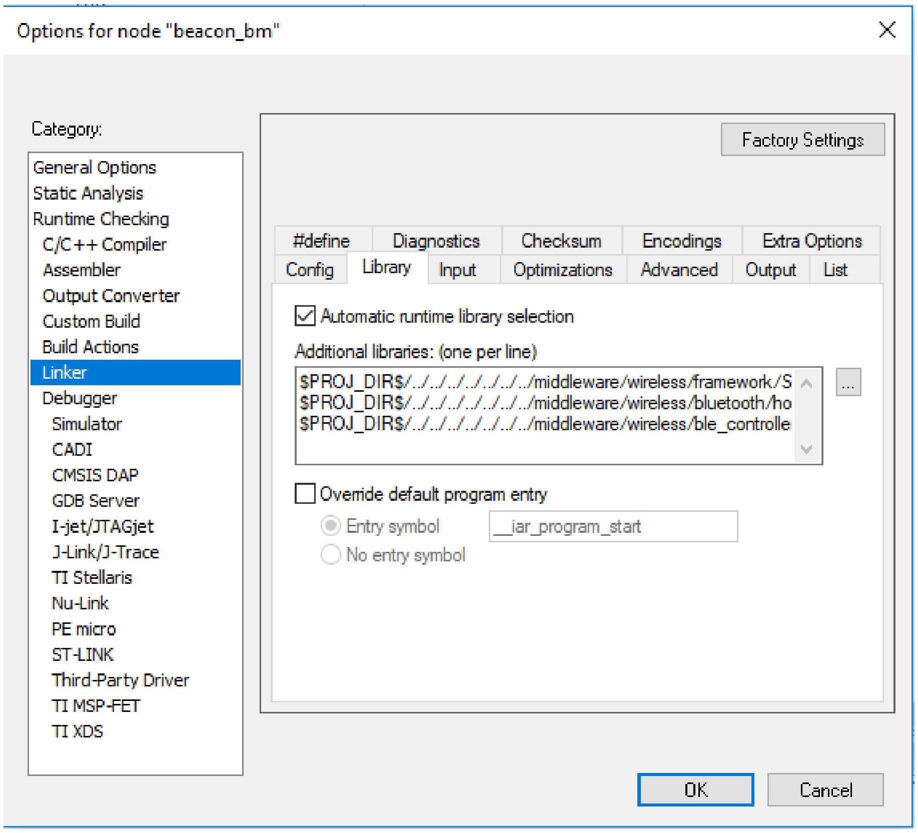
The rest of the demo examples which have the GAP peripheral role, including low-power reference design example, are called “Bluetooth LE legacy examples”. By default, these examples do not support the Advertising Extension. In this document, the Bluetooth LE Beacon is taken as the example to enable the Advertising Extension. However, the same procedure is applicable to any of the other demo examples.

3.2.1 Include the advertising extension library

By default, the Bluetooth LE Beacon example (as well as other demo examples which have the GAP peripheral role) uses the library “lib_ble_5-0_host_peripheral_cm0p.a”, which does not support the Advertising Extension.

To steps include:

1. Replace this library with “lib_ble_5-0_AE_host_cm0p.a” in the linker configurations.
2. Importing the project into IAR embedded workbench:
   a. Right-click on the project.
   b. Select **Options**.
   c. Go to **Linker > Library**.



☑ Automatic runtime library selection

Additional libraries: (one per line)

```
$PROJ_DIR$/../../../../../../../middleware/wireless/framework/S
$PROJ_DIR$/../../../../../../../middleware/wireless/bluetooth/ho
$PROJ_DIR$/../../../../../../../middleware/wireless/ble_controlle
```

□ Override default program entry

● Entry symbol __iar_program_start

○ No entry symbol

Figure 2. Options for node

3. Replace the second library “*$PROJ_DIR$/../../../../../../../middleware/wireless/bluetooth/host/lib/lib_ble_5-0_host_peripheral_cm0p.a*” with “*$PROJ_DIR$/../../../../../../../middleware/wireless/bluetooth/host/lib/lib_ble_5-0_AE_host_cm0p.a*”.

The consequence of this change is the increase of memory usage. lib_ble_5-0_host_peripheral_cm0p.a library uses 51934 bytes of flash memory, and 1523 bytes of RAM, whereas lib_ble_5-0_AE_host_cm0p.a uses 78070 bytes of Flash, and 1717 bytes of RAM.

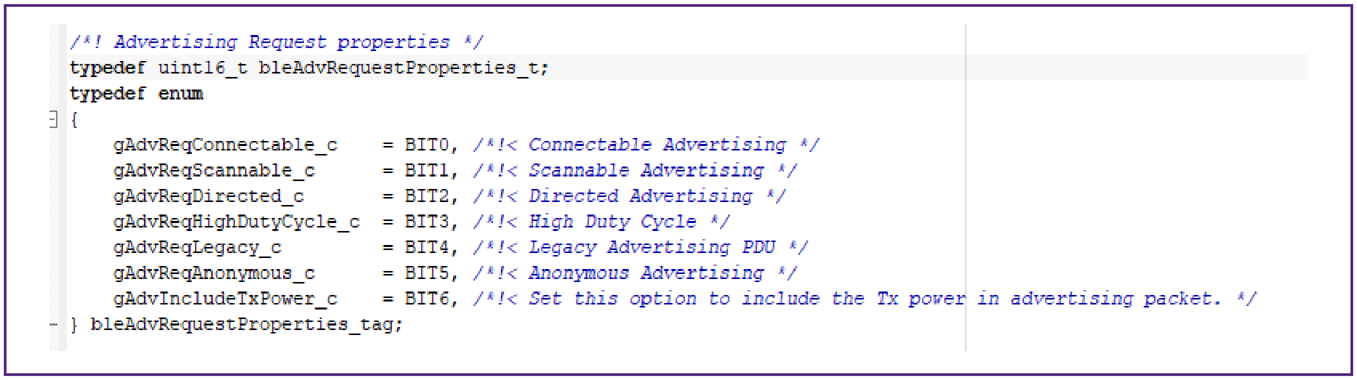
3.2.2 Configure the advertising extension

The next step is to configure the extended advertising parameters. To extend advertising, add the definition below in the app_config.c file,

```
const gapExtAdvertisingParameters_t gAppExtAdvParams =
{
    /* SID */                        1, \
    /* handle */                     1, \
    /* minInterval */                1600 /* 1 s */, \
    /* maxInterval */                3200 /* 2 s */, \
    /* ownAddrType */                gBleAddrTypePublic_c,\
    /* ownAddress */                    {0, 0, 0, 0, 0, 0}, \
    /* peerAddrType */                  gBleAddrTypePublic_c,\
    /* peerAddress */                      {0, 0, 0, 0, 0, 0}, \
    /* channelMap */                       (gapAdvertisingChannelMapFlags_t) (gAdvChanMapFlag37_c |
gAdvChanMapFlag38_c | gAdvChanMapFlag39_c), \
    /* filterPolicy */                  gProcessAll_c, \
    /* extAdvProperties */              (bleAdvRequestProperties_t)gAdvIncludeTxPower_c, \
    /* TxPower */                          5, \
    /* primaryPHY */                      (gapLePhyMode_t)gLePhy1M_c, \
    /* secondaryPHY */                 (gapLePhyMode_t)gLePhy1M_c, \
    /* secondaryAdvMaxSkip */             0, \
    /* enableScanReqNotification*/        FALSE \
};
```

This is the default parameters for the extended advertising. The application developer can adapt these parameters to their needs. Some details about the above parameters.

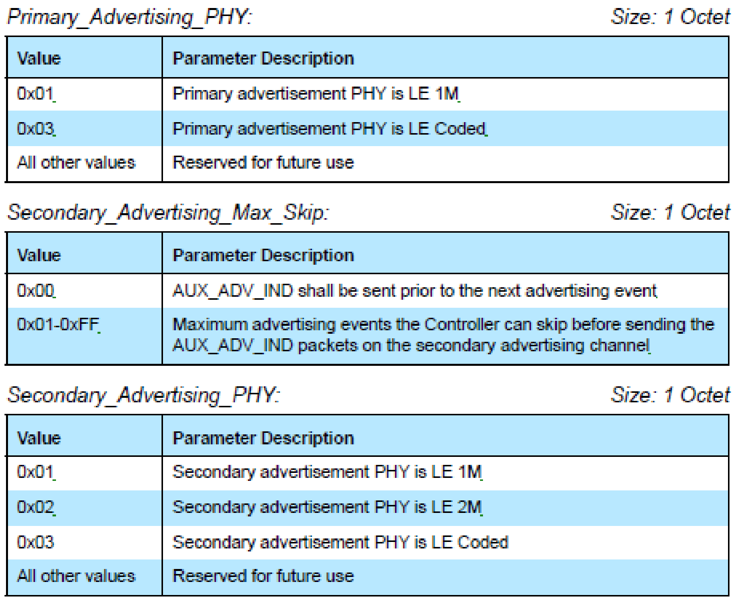
- For the handle of the Ext Adv message, it should be in the range of 0≤ handle ≤ (gLlMaxUsedAdvSet_c-1).
- For the interval definitions, one LSB is 625 µs.
- For the extAdvProperties, it is a 2 octets value.



```
/*! Advertising Request properties */
typedef uint16_t bleAdvRequestProperties_t;
typedef enum
{
    gAdvReqConnectable_c    = BIT0, /*!< Connectable Advertising */
    gAdvReqScannable_c      = BIT1, /*!< Scannable Advertising */
    gAdvReqDirected_c       = BIT2, /*!< Directed Advertising */
    gAdvReqHighDutyCycle_c  = BIT3, /*!< High Duty Cycle */
    gAdvReqLegacy_c         = BIT4, /*!< Legacy Advertising PDU */
    gAdvReqAnonymous_c      = BIT5, /*!< Anonymous Advertising */
    gAdvIncludeTxPower_c    = BIT6, /*!< Set this option to include the Tx power in advertising packet. */
} bleAdvRequestProperties_tag;
```

The developer can simply set the desired bits.

- For the Tx Power, it is a signed value in dBm. Its range can be between -127 and +20. 127 is also a valid value, it set the Tx Power to its default value. However, the KW38 is only capable of delivering a Tx Power of -32 dBm to 5 dBm, so any value configured out of this range is considered as the lower or the upper limit. In addition, the granularity is not 1 dBm, the Tx Power sets to the closest smaller value available.
- For Primary PHY and secondary PHY, see the table below.



Primary_Advertising_PHY: *Size: 1 Octet*

Value	Parameter Description
0x01	Primary advertisement PHY is LE 1M
0x03	Primary advertisement PHY is LE Coded
All other values	Reserved for future use

Secondary_Advertising_Max_Skip: *Size: 1 Octet*

Value	Parameter Description
0x00	AUX_ADV_IND shall be sent prior to the next advertising event
0x01-0xFF	Maximum advertising events the Controller can skip before sending the AUX_ADV_IND packets on the secondary advertising channel

Secondary_Advertising_PHY: *Size: 1 Octet*

Value	Parameter Description
0x01	Secondary advertisement PHY is LE 1M
0x02	Secondary advertisement PHY is LE 2M
0x03	Secondary advertisement PHY is LE Coded
All other values	Reserved for future use

NOTE

The Primary PHY cannot be LE 2 M.

3.2.3 Configure the advertising data

The extended advertising data can be set as for a normal advertising data, in app_config.c file. For legacy Bluetooth LE examples, it is recommended to redefine the “Bluetooth Low Energy LL Configuration” section in the app_preinclude.h file, in order to configure the Extended Advertising according to the application need. Remove the definition “gLlExtAdvWithLegacyAdv_d” as it is redundant.

The maximum allowed length for an extended advertisement is 1650 bytes, setting this value will however increase the memory usage by 6.6kb comparing to the case where only 31 bytes of advertising message data is allowed.. The additional RAM used is as: gLlMaxUsedAdvSet_c × (gLlMaxExtAdvDataLength_c × 2).

3.2.4 Start extended advertising

The way to start extended advertising is almost the same for the legacy BLE advertising. In this example, the file beacon.c has to be changed.

Firstly, for the function static void BleApp_Advertise(void), it is necessary to replace the call of Gap_SetAdvertisingParameters(&gAppAdvParams) with the call of Gap_SetExtAdvertisingParameters(&gAppExtAdvParams), to set the Extended advertising parameters which are defined in previous step (3.2.2). In addition, gAppExtAdvParams must be declared in beacon.h file as extern const.

Secondly, once the Extended Adv parameters are set, the gExtAdvertisingParametersSetupComplete event will be triggered. We should then set the Extended Advertising data. So, in function BleApp_GenericCallback function, the following lines have to be added in the switch (pGenericEvent->eventType) loop:

```
case gExtAdvertisingParametersSetupComplete_c:
          {
               (void)Gap_SetExtAdvertisingData(1,&gAppAdvertisingData, NULL);
          }
break;
```

The first argument of Gap_SetExtAdvertisingData is the ID of the advertising set, the second argument is the pointer to gapAdvertisingData_t structure, and the third argument is the pointer to gapScanResponseData_t structure.

The previous step sets the Extended Advertising data, and it triggers gExtAdvertisingDataSetupComplete event when finished. So the following lines need to be added in the switch loop as well:

```
case gExtAdvertisingDataSetupComplete_c:
        {
            (void)Gap_StartExtAdvertising(BleApp_AdvertisingCallback, NULL,1,0,0);
        }
break;
```

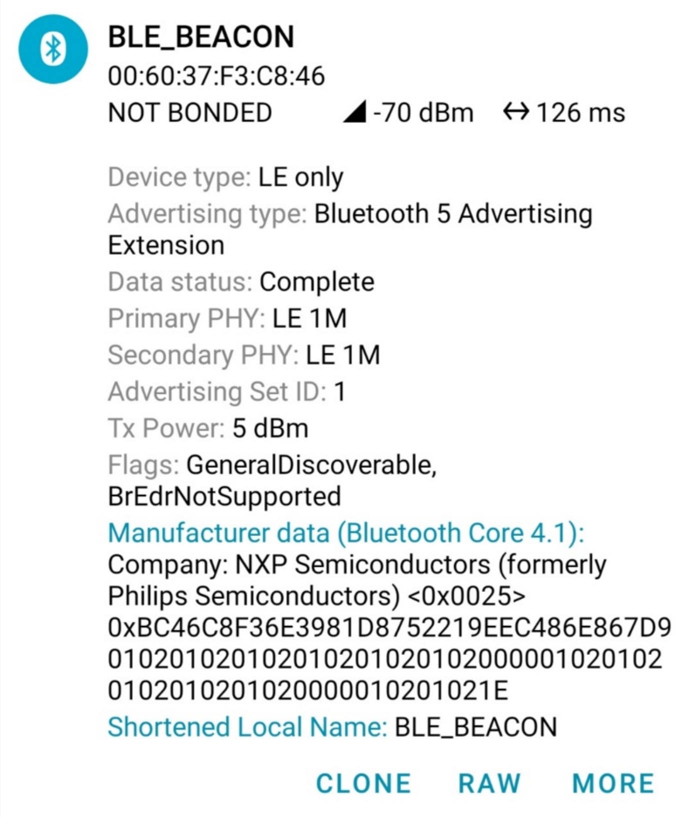
Handle is the ID of the ADV set. If duration is set to 0, it means that the ADV never stops until it receives the GAP_StopExtAdvertising command.

```
bleResult_t Gap_StartExtAdvertising
(
    gapAdvertisingCallback_t    advertisingCallback,
    gapConnectionCallback_t     connectionCallback,
    uint8_t                     handle,
    uint16_t                    duration,
    uint8_t                     maxExtAdvEvents
);
```

After this call, the device will start the extended advertising.

3.2.5 Testing

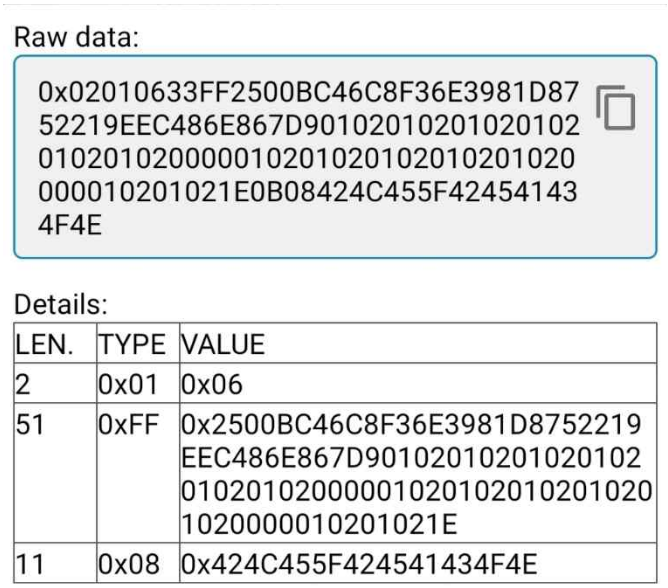
As shown below, the extended ADV is visible for smartphones compatible with Bluetooth LE 5.0 or higher, and can be read correctly:



BLE_BEACON
00:60:37:F3:C8:46
NOT BONDED ◢ -70 dBm ↔ 126 ms

Device type: LE only
Advertising type: Bluetooth 5 Advertising Extension
Data status: Complete
Primary PHY: LE 1M
Secondary PHY: LE 1M
Advertising Set ID: 1
Tx Power: 5 dBm
Flags: GeneralDiscoverable, BrEdrNotSupported
Manufacturer data (Bluetooth Core 4.1):
Company: NXP Semiconductors (formerly Philips Semiconductors) <0x0025>
0xBC46C8F36E3981D8752219EEC486E867D9
010201020102010201020102000001020102
0102010201020000010201021E
Shortened Local Name: BLE_BEACON

The raw data shows as well that the ADV data can exceed the limit of 31 bytes:



Raw data:

```
0x02010633FF2500BC46C8F36E3981D87
52219EEC486E867D90102010201020102
01020102000001020102010201020
000010201021E0B08424C455F42454143
4F4E
```

Details:

LEN.	TYPE	VALUE
2	0x01	0x06
51	0xFF	0x2500BC46C8F36E3981D8752219 EEC486E867D90102010201020102 0102010200000102010201020102 10200000102010 21E
11	0x08	0x424C455F424541434F4E

4 Revision history

Table 1 summarizes the changes done to this document since the initial release.

Table 1. Revision history

Revision number	Date	Substantive changes
0	08/2019	Initial release

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer’s applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 29 August 2019
Document identifier: AN12561