NXP Semiconductors
Application Note

Document Number: AN5272
Rev. 1, 09/2016

MKW40Z Power Consumption Analysis

1. Introduction

This document provides information about the power consumption of Kinetis KW40Z wireless MCU, how to design and optimize hardware for low-power operation, and how to configure the software to achieve the best low-power profile. This document provides overview and guidance on achieving the best low-power profile while maintaining the high performance of the system. The setup and procedures to measure the current consumption of the MKW40Z MCU are also described.

The power consumption of wireless devices is a critical requirement for the quickly growing Internet of Things (IoT) world. As a result, the hardware is gradually improved and optimized from the power consumption viewpoint and new communication standards are developed. Bluetooth® Smart (also known as Bluetooth Low-Energy or BLE) and IEEE 802.15.4 are two of these new standards that are developed for long-term (several years) battery operation.

The MKW40Z MCU is a dual-mode radio wireless device that supports both the BLE v4.1 and IEEE 802.15.4-2011 protocols.

To understand this document, the reader must have a good knowledge of the Bluetooth Smart and IEEE 802.15.4 protocols, as well as basic knowledge about the ARM® Cortex® MCU architecture and radio communication.

Contents

2. Bluetooth Smart Power Metrics

Defining a metric related to power consumption in the embedded wireless MCU world is not as simple as it might seem. Every low-power wireless MCU usually has this operational cycle:

- Both the MCU and the radio are in some kind of sleep or deep-sleep modes. The time spent in the sleep or deep-sleep modes is the longest when compared to the other operation modes.
- The MCU wakes up and performs the system-initialization and pre-processing tasks.
- The transceiver wakes up and is ready to operate. The MCU may enter the STOP mode if the software allows it.
- The transceiver performs one or more RX/TX sequences.
- The MCU processes the received and transmitted packets.
- The transceiver goes back to the sleep mode.
- The MCU enters a low-power mode (sleep or deep-sleep mode).

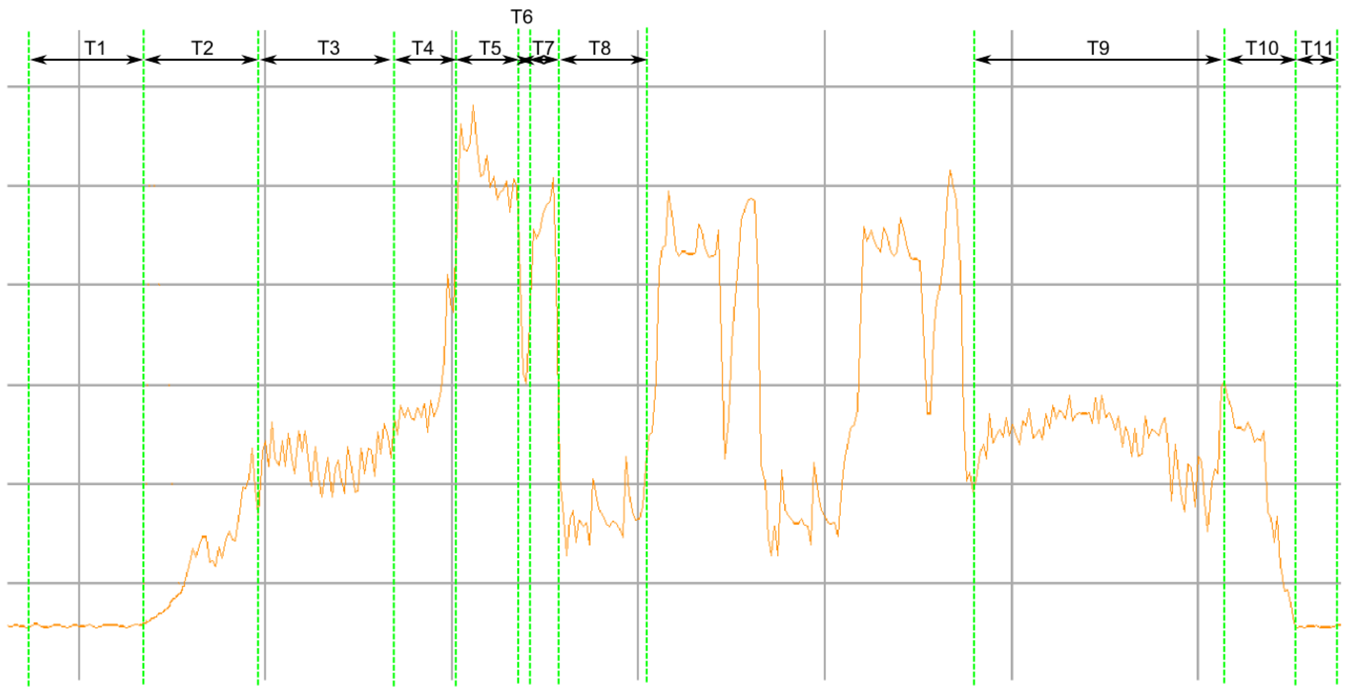


Figure 1. Typical operation cycle of a wireless low-power end device

Figure 1 shows how the current consumption varies over time for each operation cycle of the device.

At power-up, the system performs the so-called power-on reset and system initialization. After the initialization is completed, the system enters a low-power mode. There are several low-power modes available for both the MCU and the radio, but usually the software defines only the most suitable combinations of the MCU and XCVR low-power modes (for example, LLS3 for MCU and DSM for LL)

All the timings from Figure 1 are explained in this table:

Table 1. Timings of a typical low-power device

T1	SoC in SLEEP mode
T2	SoC awakes from the SLEEP mode
T3	MCU RUN: pre-processing
T4	XCVR warm-up
T5	XCVR RX operation
T6	RX to TX turnaround
T7	TX operation
T8	MCU STOP mode
T9	MCU RUN: post-processing
T10	SoC goes back to the SLEEP mode
T11	SoC in the SLEEP mode

The time it takes the transceiver to switch from RX to TX is called the RX-to-TX turnaround time, and it is an important parameter of the transceiver.

NOTE

When the radio is operational, the MCU may also perform various tasks, such as serving interrupts or controlling various peripherals.

The best metric to be applied is the current consumption over time, considering the average current of all entities that are implied.

2.1. Bluetooth Smart (BLE)

Bluetooth Smart (BLE) is one of the most promising low-power communication and a good candidate for the Internet of Things (IoT) deployments. BLE operates in the 2.4 GHz ISM band and uses GFSK modulation. The bandwidth bit period product is 0.5 and the modulation index is 0.5.

BLE uses forty 2 MHz wide channels, each separated by 2 MHz, three channels for the advertising packets, and 37 channels for data exchange. The channels are numbered from 0 to 39.

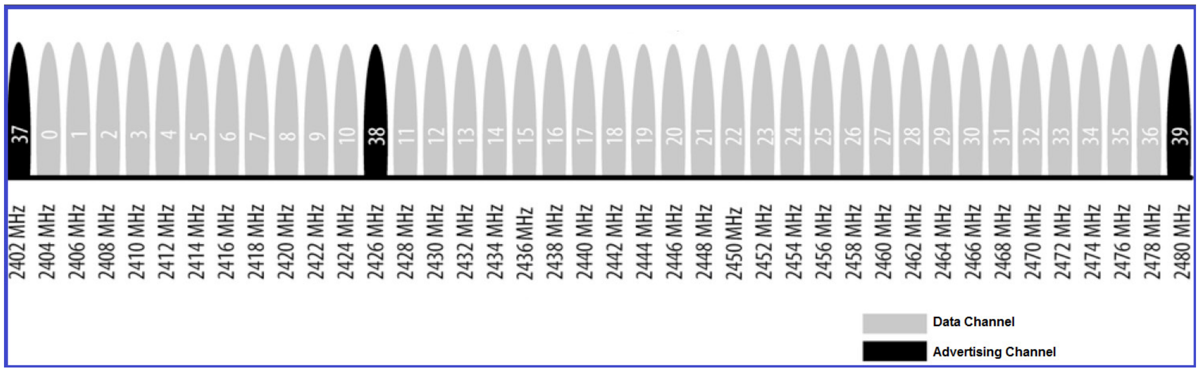


Figure 2. BLE radio channels

To ensure robustness and co-existence with the Wi-Fi channels, an adaptive frequency-channel-hopping mechanism is implemented for all data channels. At the Link Layer level, the longest BLE v4.1 packet has 47 bytes from which 39 bytes is the maximal PDU length. The PDU is different for the advertising channels and for the data channels.

The low energy is achieved by having a low duty cycle of transmission and/or reception of data and by using short advertising and data packets. The asynchronous and connection-less Link Layer ensures low latency and fast transactions.

At the GAP layer level, the roles that the BLE devices may have are the Central and the Peripheral, as shown in this figure:

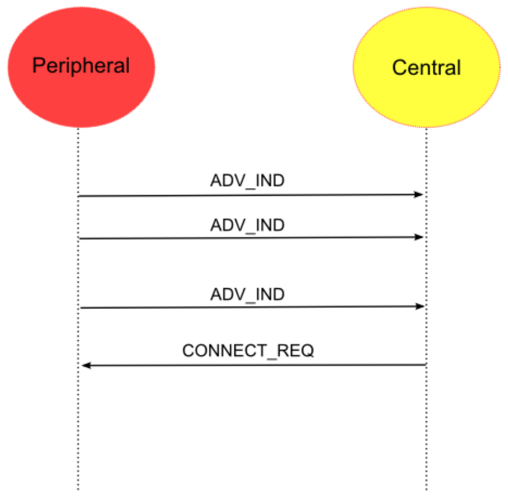


Figure 3. GAP Peripheral and GAP Central

The Peripheral starts sending the advertising data to the Central. If the Central is willing to establish a connection with the Peripheral, it sends a connection request back to the advertiser. After the connection is established, data exchange can start.

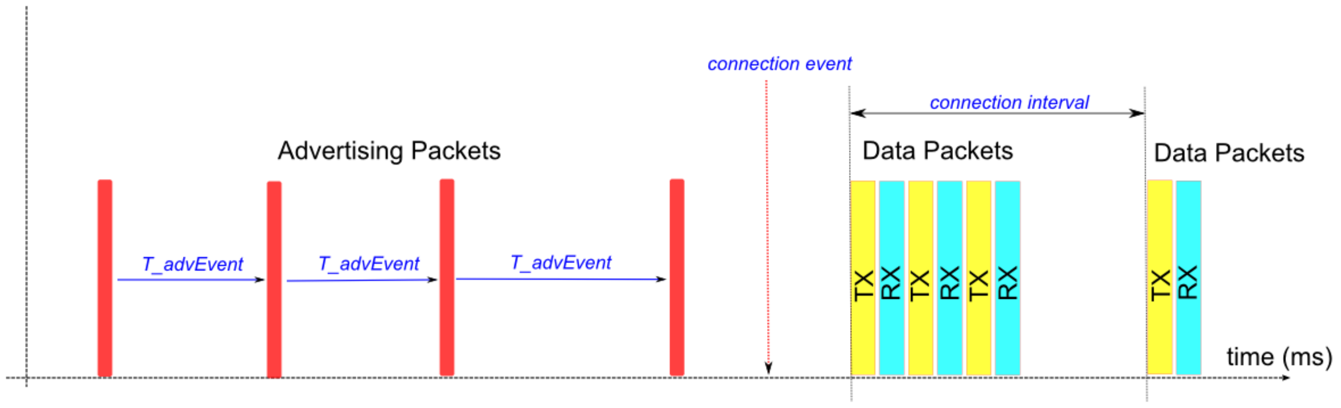


Figure 4. BLE advertising and connection events

According to the Bluetooth LE specifications, there are these four types of advertising packets:

- ADV_IND—connectable undirected advertising

- ADV_DIRECT_IND—connectable directed advertising
- ADV_NONCONN_IND—non-connectable undirected advertising
- ADV_SCAN_IND—scannable undirected adverising (formerly known as ADV_DISCOVER_IND)

All of the above types (except for the non-connectable advertising) use a TX sequence followed by an RX sequence, as shown in the following figure. This is due to the fact that the device is waiting for a scan request or a connect request from a peer device (if any) after sending the advertising packet.

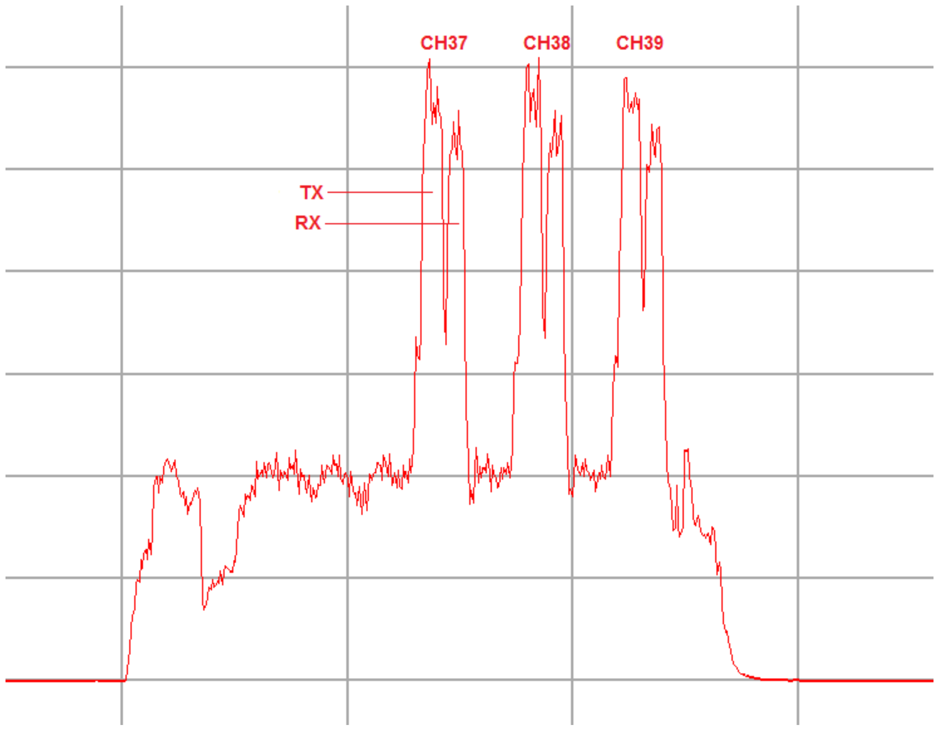


Figure 5. Current consumption vs. time in ADV phase for BLE peripheral device

The current variation in time when the system is in a typical advertising event is presented in Figure 5. There are current peaks and spikes that are present only for very short periods of time—the duty cycle still remains very low.

Another noteworthy feature of BLE is that the advertising events have a random temporal component, according to the BLE specifications.

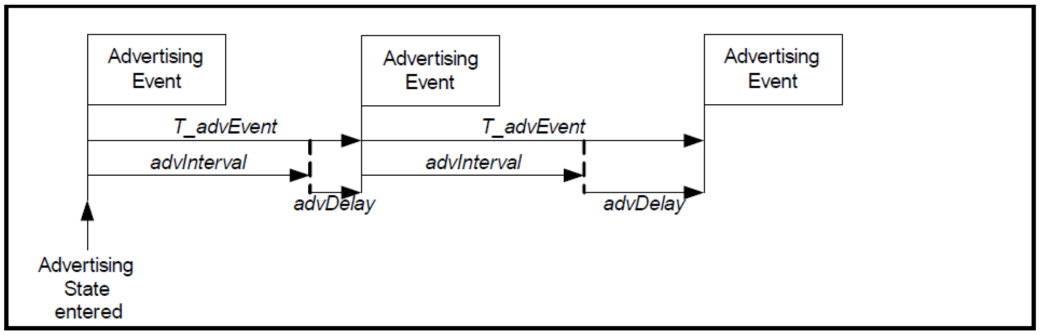


Figure 6. Advertising events occurring at unequal intervals

As shown in Figure 6, *T_advEvent = advInterval + advDelay*, where *advInterval* is an integer multiple of 0.625 ms, with a range from 20 ms to 10.24 s. *advDelay* is a pseudo-random value generated by the Link Layer that ranges from 0 to 10 ms. The minimum advertising event interval is 20 ms and the maximum interval is 10.25 s.

Bluetooth LE is designed and implemented for ultra-low-power battery-operated devices, but the actual power consumption of a real BLE device strongly depends on:

- The BLE application profile.
- The application duty cycle.
- The TX power.
- The software management of low-power modes.
- The board design and layout.

3. MKW40Z Low-Power Features

The KW40Z is an ultra-low-power, highly integrated single-chip device that enables Bluetooth Low Energy (BLE) v4.1 or IEEE Std. 802.15.4/ZigBee RF connectivity for portable, extremely low-power embedded systems. The applications include portable health care devices, wearable sports and fitness devices, AV remote controls, computer keyboards and mice, gaming controllers, access control, security systems, smart energy, and home-area networks.

The KW40Z SoC integrates a radio transceiver operating in the 2.36 GHz to 2.48 GHz range, supporting a range of FSK/GFSK and O-QPSK modulations, an ARM Cortex-M0+ CPU, 160 KB flash and 20 KB SRAM, BLE Link Layer hardware, IEEE 802.15.4 packet processor hardware, and peripherals optimized to meet the requirements of the target applications.

The KW40Z’s radio frequency transceiver complies with Bluetooth version 4.1 for Low Energy (aka Bluetooth Smart), the IEEE 802.15.4-2011 standard using O-QPSK in the 2.4 GHz ISM band, and the IEEE 802.15.4j MBAN frequency range spanning from 2.36 GHz to 2.40 GHz. In addition, the KW40 enables the Bluetooth Low Energy protocol to be used in the MBAN frequency range for proprietary applications.

An extremely long battery life is achieved by the efficiency of code execution in the Cortex-M0+ CPU core and the multiple low-power operating modes of the KW40Z. An integrated DC/DC converter enables a wide operating range from 0.9 V to 4.2 V. The DC/DC in the buck mode enables the KW40Z to operate from a single coin-cell battery with a significant reduction of peak RX and TX current consumption. The DC/DC in the boost mode enables a single alkaline battery to be used throughout its entire useful voltage range of 0.9 V to 1.795 V.

3.1. MKW40Z hardware support for low-power operation

The MKW40Z SoC is designed and built with hardware features that enable the chip to operate in various low-power modes. The noteworthy features include:

- Multiple MCU power modes including low leakage with memory retention modes.
- BLE Link Layer sleep mode support.
- Peripheral modules clock gating.

- Several peripheral doze modes.
- DC/DC converter.
- Transceiver Sequence Manager (TSM) that assures that the transceiver analog and digital blocks do not consume power when no RX/TX sequence is in progress.
- Dedicated Power Management Controller (PMC).
- Low-power peripherals (LPTMR, LPUART) that you can configure as wake-up sources to exit a particular low-power state.

The software is responsible for configuring all the hardware to achieve the best power scheme required by the applications. As described in the following sections, the chip low-power modes are combinations of the MCU and LL/Packet Processor sleep modes. The clock gating of peripherals as well as the GPIO states before entering the low-power mode are in charge of the application developer. The Connectivity Software package provides callbacks that are called before entering the low-power mode and after exiting the low-power mode. The system must enter the low-power mode when it is idle and all software layers agree on that. The system must exit the low-power mode each time a synchronous or asynchronous event appears and requires to be processed.

3.1.1. KW40Z MCU power modes

The PMC module provides a variety of power options to enable you to optimize and personalize the power consumption with respect to the level of functionality that the application requests. Based on the ARM architecture power modes, there are two power modes defined: the sleep mode and the deep-sleep mode. From the Connectivity Software viewpoint, only the deep-sleep modes are of interest.
The deep-sleep modes are:

- STOP
- LLS3
- VLLS0
- VLLS1
- VLLS2

The CPU recovery method uses the wakeup interrupt for the stop and low-leakage stop modes, and the wakeup reset for the very-low-leakage stop modes. All power modes are listed in the *MKW40Z Reference Manual* (document MKW40Z160RM).

3.1.2. KW40Z Link Layer power modes

The Bluetooth Link Layer (BTLL) has these power modes:

- IDLE
- RUN
- DSM (Deep-Sleep Mode)

For BLE, the Connectivity Software package implements six low-power modes for the KW40Z SoC, as shown in this table:

Table 2. KW40Z low-power modes for BLE applications

Low-power mode	Required state		Wakeup sources				
	MCU	BTLL	GPIO	BTLL	LPTMR	DCDC[1]	UART
1	LLS3	DSM	x	x	x		
2	LLS3	IDLE	x	x			
3	LLS3	IDLE	x		x	x	
4	VLLS0/1[2]	IDLE	x			x	
5	VLLS2	IDLE	x			x	
6	STOP	IDLE/RUN	x	x	x	x	x

1. Only available in the buck mode. For the boost mode, the PSWITCH is hardwired to VDCDC_IN and therefore not available.
2. VLLS0 if DCDC in the bypass mode, VLLS1 otherwise.

NOTE

The BLE and IEEE 802.15.4 use a common radio transceiver digital block (the TSM that is used to sequence the analog regulators and circuits needed for the RX/TX operations on or off) so that these circuits only consume power during RX/TX.

3.1.3. KW40Z XCVR power modes

The KW40Z transceiver is tightly coupled with the MCU. Therefore, the transceiver analog regulators are powered off whenever the MCU enters a low-power mode. Depending on the low-power mode, the transceiver digital logic is power-gated or has its state retained.

3.1.4. KW40Z DC/DC converter

The DCDC module is a Switched Mode Power Supply (SMPS) DC/DC converter that has these three operation modes:

- Buck—V_{in} = 2.1 V to 4.2 V.
- Boost—V_{in} = 0.9 V to 1.795 V.
- Bypass—DC/DC is disabled, the required supplies must be provided externally.

The module is configurable through internal registers to operate in the continuous or pulsed modes and it provides two voltage outputs in both the buck and boost modes: 1.45 V (25 mA) and 1.8 V (42.5 mA). The efficiency of the converter is around 90 %. The operation modes are selected by setting the DCDC_CFG pin. Start the converter using the PSWITCH pin or set it to the auto-start mode.

For detailed information about the DC/DC converter, see *MKW40 Power Management* (document AN5025).

3.1.5. GPIO, analog pins, and clock gating

The clock gating mechanism is implemented to reduce power dissipation, in other words, whenever a peripheral is not used, you can turn it off using the SCGCx registers in the SIM module. Clock gating applies to each peripheral, including the GPIO module. Pruning the clock to a peripheral assures that the peripheral internal circuitry does not have switch states (and therefore no power consumption), except for the leakage currents.

After the reset, the clock gating bits are cleared. Before using any peripheral, set the corresponding clock gating bit, otherwise any access to the peripheral registers causes a hardware fault. To turn the peripheral clock off (gate off), turn the peripheral off before the clock.

The application must control and set the state of the GPIO ports before the device goes to sleep, as well as after the device exits the low-power state. The Connectivity Software package provides callback functions that are called before the device enters the low-power state and after it wakes up.

Related to the analog pins, the device has several analog blocks that have selectable reference voltages. The main blocks are the 16-bit SAR ADC, the 12-bit DAC, and the CMP. The board design must take the chip analog pins into account and use them appropriately.

The external analog inputs are usually shared with the digital I/O. To improve the performance in the presence of noise or when the source impedance is high, use the capacitors on these inputs. Place the capacitors as close to the chip analog pins as possible.

For more details, see the reference designs and the reference manual for KW40Z.

3.2. Software configuration for low-power operation

3.2.1. DC/DC converter software setup

To configure the DC/DC converter using the application software, see *MKW40 Power Management* (document AN5025).

3.2.2. Bluetooth Smart applications configuration

The Connectivity Software package offers a variety of BLE demo projects. These projects are located at this relative path: *\ConnSw\examples\bluetooth\*

The Heart Rate Sensor is selected for power measurements because it is an application that is typically optimized for low power consumption and it is a commonly used BLE profile. The relative project path is:

\ConnSw\examples\bluetooth\heart_rate_sensor\frdmkw40z\FreeRTOS\build\iar\heart_rate_sensor.eww

The application starts by initializing the MCU hardware, peripherals, Connectivity Framework, OS tasks, timers and queues, transceiver, and finally the BLE host stack.

On the idle task (if device is eligible for it), the device enters the low-power mode 3 (MCU→LLS3, LL→IDLE, Wakeup Source→GPIO/LL).

The device exits the low-power mode and the BLE application starts when you press the SW3 push-button (*app.c* source file).

```
void BleApp_HandleKeys(key_event_t events)
{
    switch (events)
    {
        case gKBD_EventPressPB1_c:
```

```
        {
            if (mPeerDeviceId == gInvalidDeviceId_c)
            {
                BleApp_Start();
            }
        }
```

After the wakeup, the device starts to advertise. The advertising parameters are defined in the *gap_types.h* header file:

```
typedef struct gapAdvertisingParameters_tag {
       uint16_t                            minInterval;
       uint16_t                            maxInterval;
       bleAdvertisingType_t                advertisingType;
       bleAddressType_t                    ownAddressType;
       bleAddressType_t                    directedAddressType;
       bleDeviceAddress_t                  directedAddress;
       gapAdvertisingChannelMapFlags_t     channelMap;
       gapAdvertisingFilterPolicyFlags_t   filterPolicy;
} gapAdvertisingParameters_t;
```

The default advertising parameters values are also defined in the *gap_types.h* header file:

```
#define gGapDefaultAdvertisingParameters_d \
{ \
    /* minInterval */         gGapAdvertisingIntervalDefault_c, \
    /* maxInterval */         gGapAdvertisingIntervalDefault_c, \
    /* advertisingType */     gAdvConnectableUndirected_c, \
    /* addressType */         gBleAddrTypePublic_c, \
    /* directedAddressType */ gBleAddrTypePublic_c, \
    /* directedAddress */     {0, 0, 0, 0, 0, 0}, \
    /* channelMap */          (gapAdvertisingChannelMapFlags_t) (gAdvChanMapFlag37_c |
gAdvChanMapFlag38_c | gAdvChanMapFlag39_c), \
    /* filterPolicy */        gProcessAll_c \
}
```

The actual values of the advertising parameters are set in the *app_config.c* file and you can change the values at runtime:

```
gapAdvertisingParameters_t gAdvParams = {
    /* minInterval */         gGapAdvertisingIntervalDefault_c, \
    /* maxInterval */         gGapAdvertisingIntervalDefault_c, \
    /* advertisingType */     gAdvConnectableUndirected_c, \
    /* addressType */         gBleAddrTypePublic_c, \
    /* directedAddressType */ gBleAddrTypePublic_c, \
    /* directedAddress */     {0, 0, 0, 0, 0, 0}, \
    /* channelMap */          (gapAdvertisingChannelMapFlags_t)
(gGapAdvertisingChannelMapDefault_c), \
    /* filterPolicy */        gProcessAll_c \
};
```

The default advertising interval is 1.28 s, but it is overwritten by the application profile within the *app.h* file:

```
#define gFastConnMinAdvInterval_c       32 /* 20 ms */
#define gFastConnMaxAdvInterval_c       48 /* 30 ms */

#define gReducedPowerMinAdvInterval_c   1600 /* 1 s */
#define gReducedPowerMaxAdvInterval_c   4000 /* 2.5 s */
```

There are two types of advertising: the fast advertising interval and the reduced power (slow) advertising interval. Both have the minimum and maximum values set in the BLE controller. The controller itself chooses the advertising interval, taking the interval set and the random advertising delay into account. These values are specific to a profile and it is not recommended to modify them. Each type of advertising (fast, slow) has a defined timeout. If no connection request is received during the advertising and the timeout occurs, the advertising type is switched (fast to slow and slow to fast). These timeouts are defined in the *app.h* file:

```
#define gFastConnAdvTime_c              30  /* s */
#define gReducedPowerAdvTime_c          300 /* s */
```

There is an advertising callback implemented within the *app.c* file (`BleApp_AdvertisingCallback`) that is called by the host stack each time an advertising event occurs. When the advertising starts, the application code starts the so-called advertising timer that is used to count down the timeouts mentioned earlier:

```
PWR_ChangeDeepSleepMode(1);
/* Start advertising timer */
TMR_StartLowPowerTimer(mAdvTimerId,gTmrLowPowerSecondTimer_c,
              TmrSeconds(mAdvTimeout), AdvertisingTimerCallback, NULL);
Led1On();
```

After the advertising packet is sent (according to the advertising channel mask), the same callback is called again and the application code puts the device into power mode 1 (as defined in Table 1):

```
Led1Off();
PWR_ChangeDeepSleepMode(1);
PWR_SetDeepSleepTimeInMs(cPWR_DeepSleepDurationMs);
PWR_AllowDeviceToSleep();
```

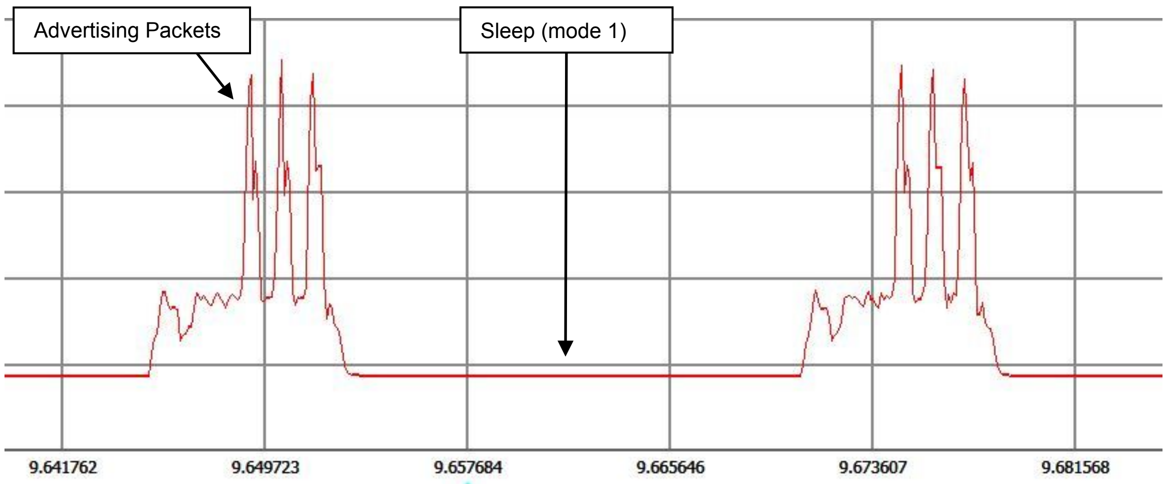


Figure 7. BLE ADV packets

When a connection is established with the Central, the advertising stops and the Peripheral starts to report. In this particular case, there are two characteristics that are reported: the heart rate value and the battery level. The report interval is defined in the *app.c* file:

```
#define mHeartRateReportInterval_c      (1) /* heart rate report interval in seconds  */
#define mBatteryLevelReportInterval_c   (10) /* battery level report interval in seconds  */
```

By changing these values, you can modify the device duty cycle (and therefore also its power consumption).

The connection interval is negotiated and set by the Central, but the Peripheral can send a connection parameter update request that the Central can accept or reject.

Another important connection parameter is the Slave latency (the Peripheral can choose not to answer when the Central asks for data up to the Slave latency number of times).

Between the connection events, the device enters the low-power mode 1 (MCU→LLS3, LL→DSM, Wakeup sources→GPIO/LL/LPTMR).

```
PWR_SetDeepSleepTimeInMs(900);
PWR_ChangeDeepSleepMode(1);
PWR_AllowDeviceToSleep();
```

To maintain the connection, the Peripheral may send empty packets (keep alive) to the Central but this happens only at the Link Layer level (the MCU does not leave the sleep mode).

The connection can be ended by the Central or by the Peripheral by long-pressing the SW3 push-button. When the disconnect event is received, the application code (*app.c*, *BleApp_ConnectionCallback*) puts the device into power mode 3 (as defined in Table 1, MCU→LLS3, LL→IDLE, Wakeup Sources→GPIO/LL).

```
/* Go to sleep */
PWR_ChangeDeepSleepMode(3); /* MCU=LLS3, LL=IDLE, wakeup on switches/LL */
PWR_SetDeepSleepTimeInMs(cPWR_DeepSleepDurationMs);
PWR_AllowDeviceToSleep();
```

The *app_preinclude.h* header file is a very important configuration file. The macros defined in this file overwrite all the macros with the same name within the entire project. Related to the low-power functionalities, the macros in the following table are implied. The settings are for the Heart Rate Sensor Project only. The other projects may have different low-power settings.

Table 3.

Name	Value for low-power enabled	Description
gTMR_EnableLowPowerTimers	1	Enables/disables the Low-Power Timer.
cPWR_UsePowerDownMode	1	Enables/disables the PowerDown functionality in the PwrLib software module.
cPWR_BLE_LL_Enable	1	Enables/disables the BLE Link Layer deep-sleep mode.
cPWR_DeepSleepMode	3	Default power mode: MCU=LLS3, LL=IDLE.
APP_DISABLE_PINS_IN_LOW_POWER	1	Disables all pins when entering the low-power mode.
cPWR_DeepSleepDurationMs	30000	Default deep sleep duration [ms].
cPWR_BLE_LL_OffsetToWakeupInstant	3	Number of slots (1 slot = 625 μs) before the wakeup instant, before which the hardware must exit from the deep-sleep mode. This parameter provides the “flat zone” before the first advertising event.
gDCDC_Enabled_d	1	Enables/disables the DC/DC platform component.
APP_DCDC_MODE	gDCDC_Mode_Buck_c	The default DC/DC mode used by the application.

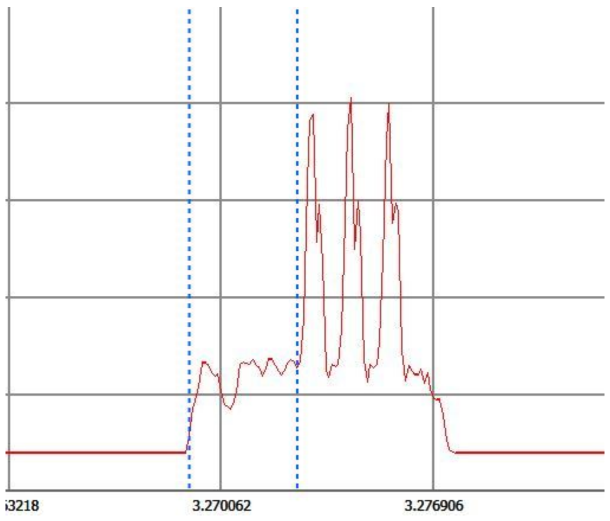


Figure 8. Offset to wakeup instant

Another important thing related to the low-power functionalities is that you can register the callback functions that are called from the low-power software module, just before entering the low-power state and after exiting the low-power state. These callbacks are registered in the *board.c* source file:

```
void BOARD_InstallLowPowerCallbacks()
{
#if cPWR UsePowerDownMode
  PWR_RegisterLowPowerEnterCallback((pfPWRCallBack_t)BOARD_EnterLowPowerCb);
  PWR RegisterLowPowerExitCallback((pfPWRCallBack_t)BOARD_ExitLowPowerCb);
#endif
}
```

In the particular case of the Heart Rate Sensor, the `BOARD_EnterLowPowerCb()` callback disables all used GPIOs and changes the DC/DC converter to the pulsed-mode operation. After exiting the low-power mode, the SWD and LED pins are enabled and the DC/DC converter changes to the continuous mode:

```
void BOARD_EnterLowPowerCb()
{
#if APP_DISABLE_PINS_IN_LOW_POWER
    BOARD_TogglePins(TRUE);
#endif

#if gDCDC Enabled d
    DCDC_BWR_REG0_DCDC_VBAT_DIV_CTRL(DCDC_BASE_PTR, 0);
    DCDC_PrepareForPulsedMode();
#endif
}
void BOARD_ExitLowPowerCb()
{
#if APP_DISABLE_PINS_IN_LOW_POWER
  BOARD_TogglePins(FALSE);
#endif

#if gDCDC_Enabled_d
    DCDC_PrepareForContinuousMode();
#endif
}
```

4. Power Measurements and Analysis

4.1. Setting up test environment and DUT

This section describes how to set up the test environment, what hardware tools and boards are required, and all the necessary operations that must be done before performing the measurements.

All the measurements are performed using the Keysight N6705A power analyzer.

Figure 9. Keysight N6705A power analyzer

Analyzer module 2 is used as a power supply while module 3 is used as an ammeter. The power supply is set to provide 3.0 V DC. Two pairs of high-quality cables are required; one for supplying the board and one for the current measurement. The connections between the power analyzer and the Freedom board must be perfect to avoid unwanted spikes, power losses, or board resets.

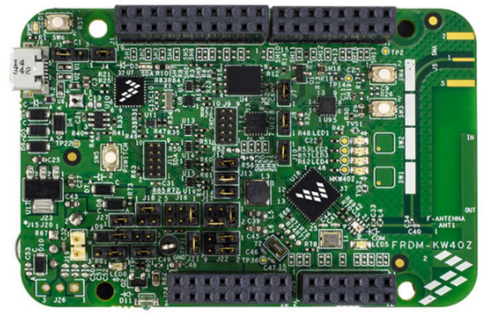


Figure 10. FRDM-KW40Z board, rev. C

Supply the FRDM-KW40Z board using the J19 header connector. Shunt the jumper J23 at the 2-3 position to enable the external power supply (via J19).

The current measurement is performed at the J34 pads. When programming the board, J18 has the jumper at the 2-4 position. After programming the board, remove the jumper.

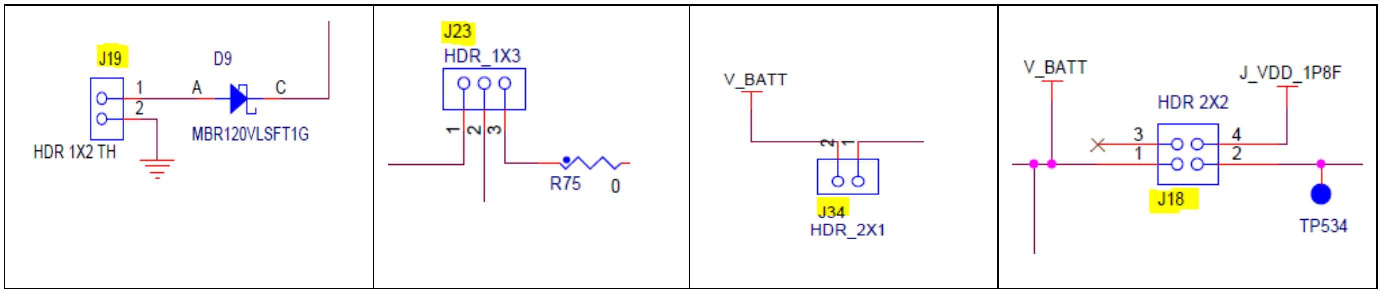


Figure 11. FRDM-KW40Z board setup

There are two ways to perform the measurements:

- Using the power analyzer's built-in display and a USB flash memory stick to save the results.
- Using Keysight 14585A Control and Analysis Software (requires a USB connection between the power analyzer and the host PC).

The measurements presented in this application note use the second option (Keysight 14585A Control and Analysis Software).

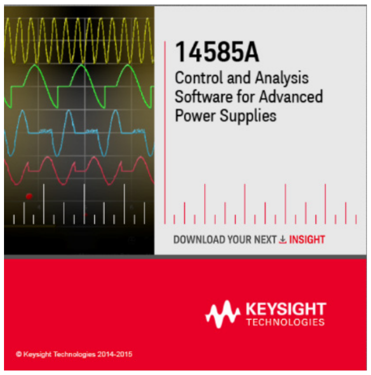

14585A
Control and Analysis Software for Advanced Power Supplies
DOWNLOAD YOUR NEXT INSIGHT
KEYSIGHT TECHNOLOGIES
© Keysight Technologies 2014-2015

Figure 12. Keysight 14585A Control and Analysis Software

The FRDM-KW40Z board is programmed with the Heart Rate Sensor binary using J-Link for ARM as the programming/debugging tool[1].

[1] J-Link Lite can't be used when the DC/DC converter is configured in the buck mode, because the FRDM-KW40Z SWD pins' voltage levels are not compatible with J-Link Lite.

Figure 13. J-Link for ARM programmer/debugger

The relative project path is:

\ConnSw\examples\bluetooth\heart_rate_sensor\frdmkw40z\FreeRTOS\build\iar\heart_rate_sensor.eww

The DC/DC converter is configured to operate in the buck mode. If the board has a SMA connector and the C34 capacitor is populated, then a SMA antenna is required to be connected to the board.

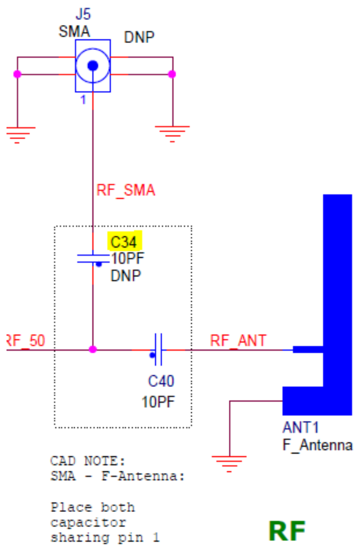


Figure 14. SMA configuration

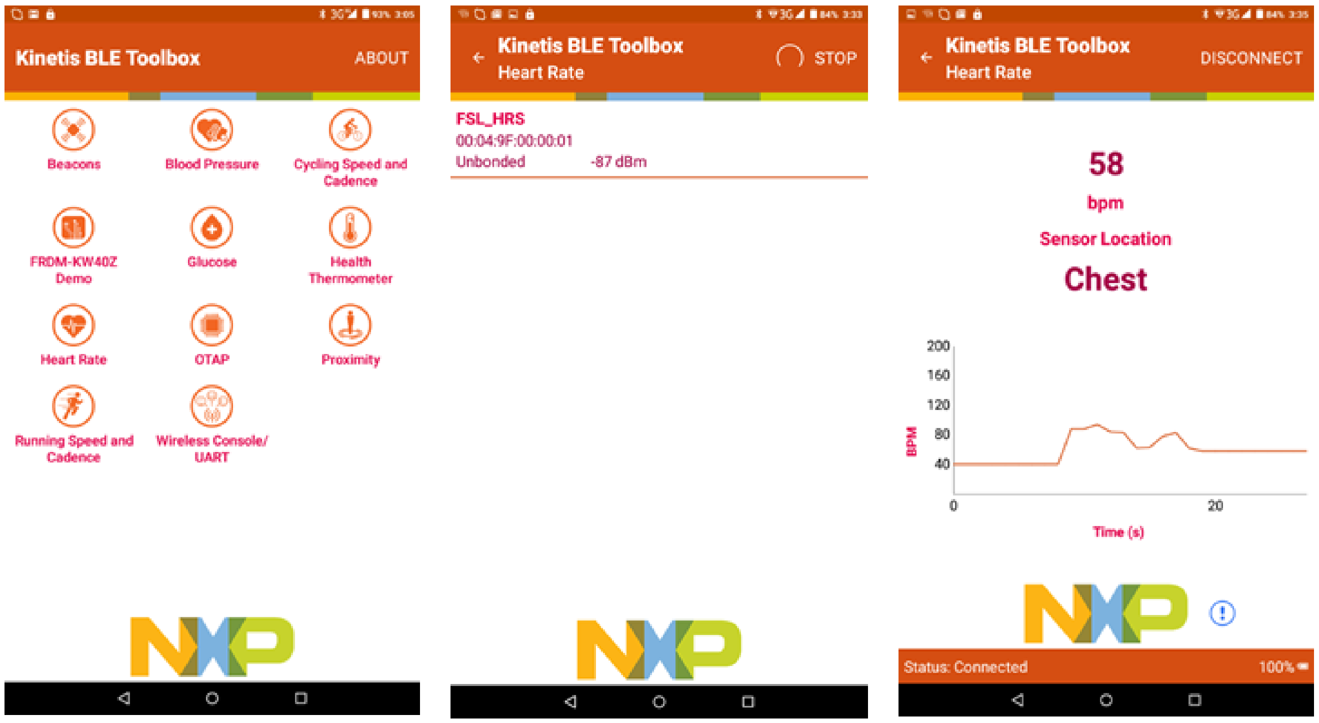


Figure 15. Kinetis BLE toolbox

On the Central side (in this case a mobile phone or a tablet with the BLE available), install the Kinetis BLE Toolbox application (available from Google Play™ or iTunes®).

Use the Heart Rate Monitor application. There is no need for the connecting device to measure the advertising events, but it is mandatory to measure the connection events. To connect to the FRDM-KW40Z board, the procedure is simple and straightforward:

- Open Kinetis BLE Toolbox.
- Press the “HEART RATE” button.
- Power-up the FRDM-KW40Z board and press the SW3 push-button to start advertising.
- On the Android™ application, the FSL_HRS is reported at the scan phase.
- Connect to the FSL_HRS peripheral.
- Wait for the measurements.

4.2. Measuring power consumption

All the measurements within this sub-section are performed at 0 dBm (1 mW) (power level 5, see the *controller_interface.h* header file). This document does not describe how to use the power analyzer or the Control and Analysis Software.

Use the steps provided in the previous section to capture a full BLE scenario (as shown in the following figure). The main events and phases are documented within the capture. All the plots that follow depict the current consumption (y-axis) in regards to time (x-axis).

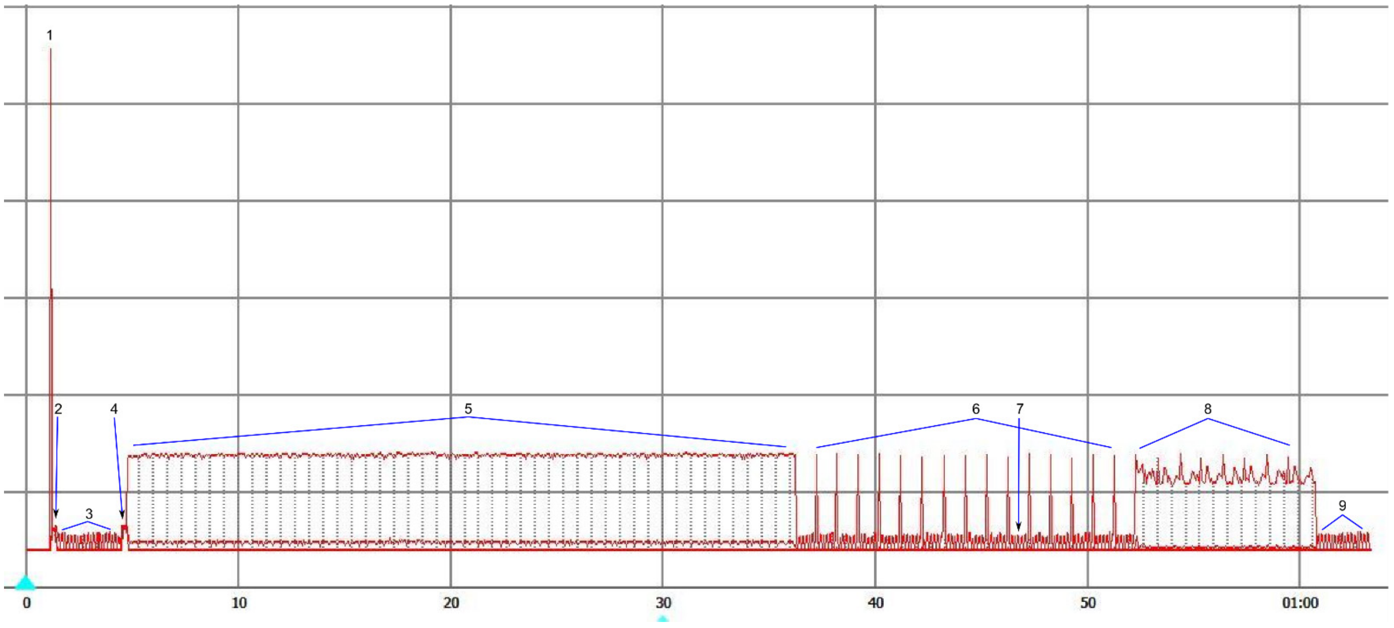


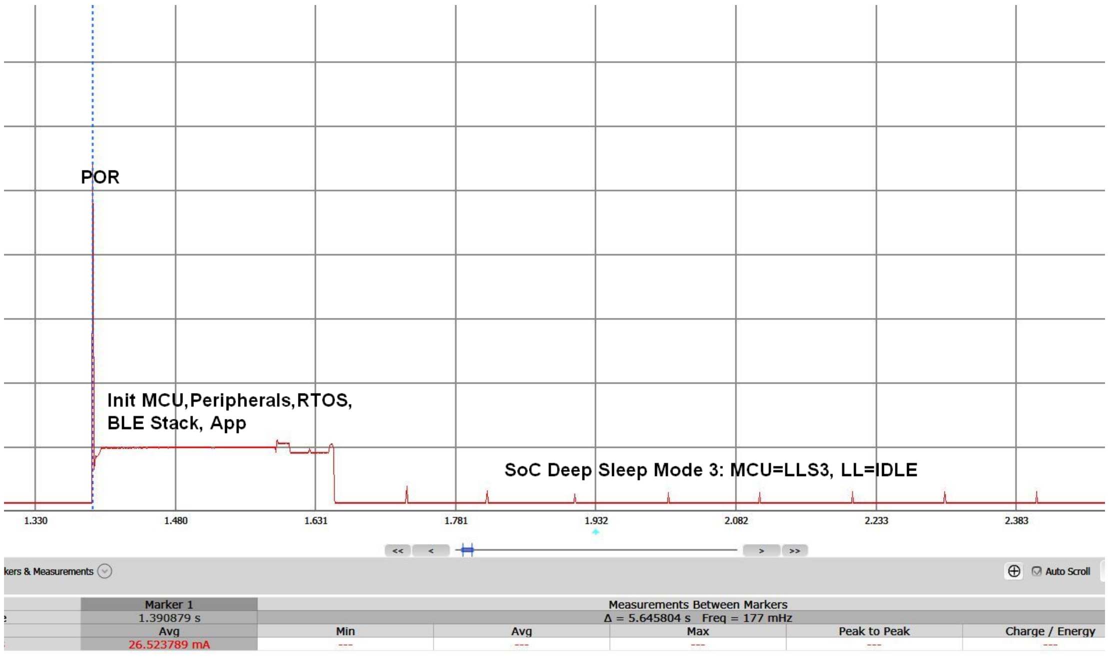
Figure 16. KW40Z SoC current consumption

The above figure shows the current consumption of the KW40Z SoC during different operational phases, as noted in this table:

Table 4.

Phase number	Description
1	Power On Reset (POR), just after connecting the SoC to a power supply. The spike in the figure is about 26 mA and is caused by the coupling capacitors as well as the SoC internal circuitry (regulators, clock oscillators, MCU, radio digital, radio analog, and so on).
2	The MCU is initialized in all the software: low-level drivers, framework, RTOS, BLE stack, application.
3	The system enters the low-power mode 3 (MCU = LLS3, LL = IDLE). The spikes are caused by the DC/DC converter that operates in the pulsed mode.
4	The MCU leaves the low-power mode 3 (by pushing the SW3 push-button) and resumes its execution. The BLE Link Layer goes to the RUN state.
5	Fast advertising starts. Between the advertising events, the system enters the low-power mode 1 (MCU = LLS3, LL = DSM).
6	If no connection request occurs during the fast advertising period and the timeout expires, the slow advertising starts.
7	Between the advertising events, the system enters the power mode 1 (MCU = LLS3, LL = DSM).
8	If a connection request is received, the advertising stops and the connection events start.
9	After disconnecting, the SoC enters the power mode 3 (MCU = LLS3, LL = IDLE).

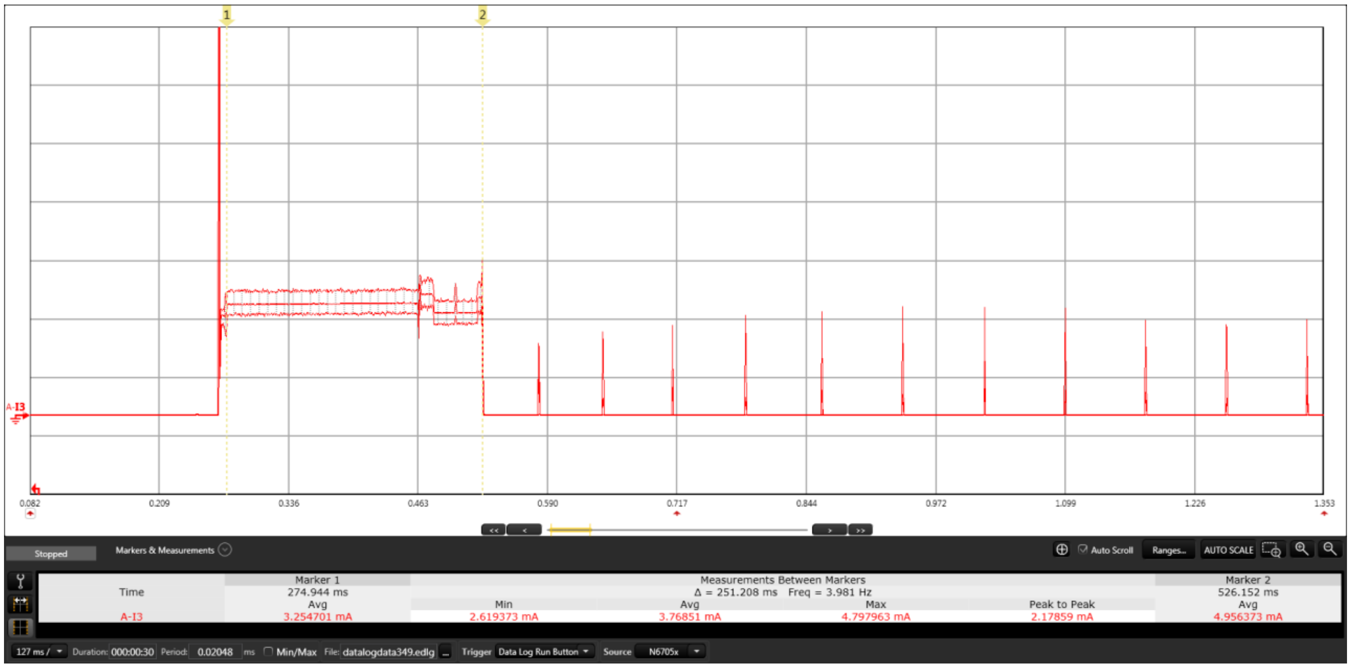
All of the above phases are analyzed and measured in the following paragraphs.



	Marker 1	Measurements Between Markers				
	1.390879 s	Δ = 5.645804 s Freq = 177 mHz				
	Avg	Min	Avg	Max	Peak to Peak	Charge / Energy
	26.523789 mA	---	---	---	---	---

Figure 17. KW40 POR and initialization

When the SoC is connected to the power supply, a power-up spike occurs due to coupling of the board to the power supply. After the MCU POR, the software execution begins, the clocks and peripherals are enabled and configured, the Connectivity Framework is initialized, RTOS tasks are initialized and started, the BLE stack is up and running, and the BLE application is started. After all of this is completed, the system enters deep sleep mode 3 (MCU=LLS2, LL=IDLE). The initialization phase (before the system enters the deep sleep) takes about 250 ms with an average current consumption about 3.8 mA.

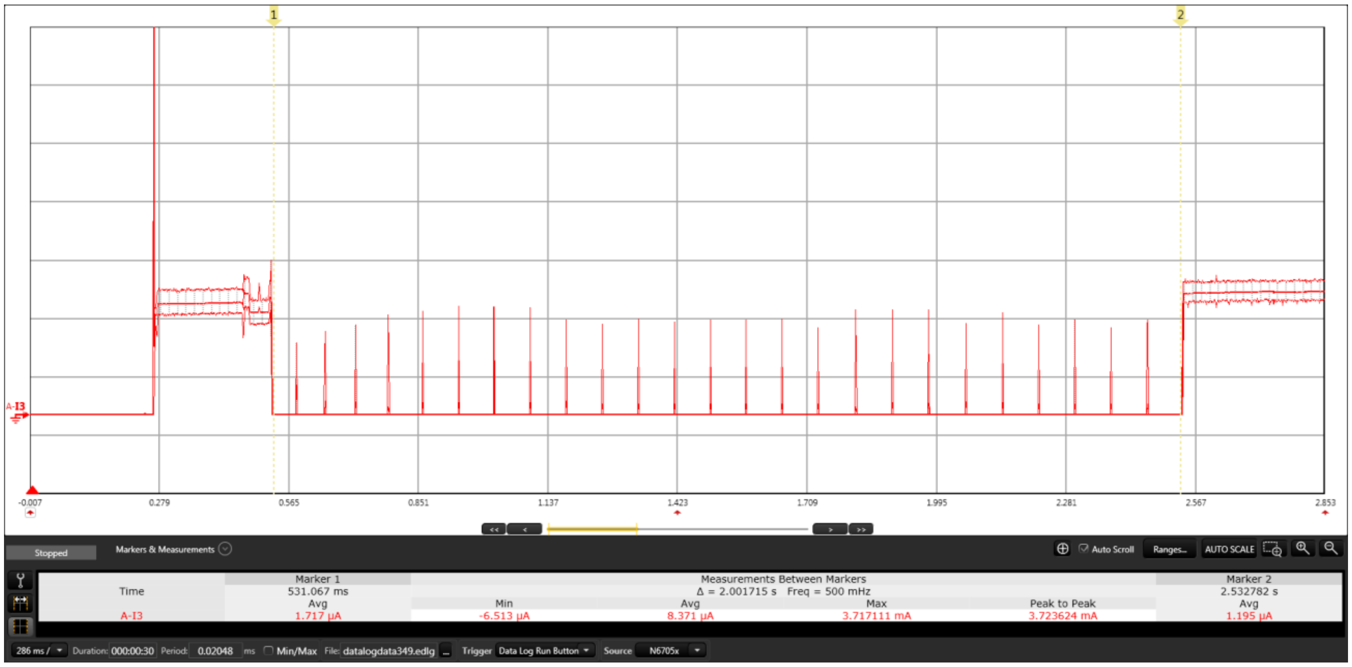


	Marker 1	Measurements Between Markers				Marker 2
Time	274.944 ms	Δ = 251.208 ms Freq = 3.981 Hz				526.152 ms
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	3.254701 mA	2.619373 mA	3.76851 mA	4.797963 mA	2.17859 mA	4.956373 mA

127 ms / Duration: 000:00:30 Period: 0.02048 ms Min/Max File: datalogdata349.edlg Trigger Data Log Run Button Source N6705x

Figure 18. System initialization after POR

The device operates in the deep sleep mode 3 until the SW3 key is pressed. By pressing the SW3, the system wakes up because the GPIO associated to the SW3 is configured as interrupt source in the LLWU (Low-Leakage Wake-up Unit) module. While in deep sleep mode 3, the module has an average consumption of 8.4 µA (Δ = 2 s).

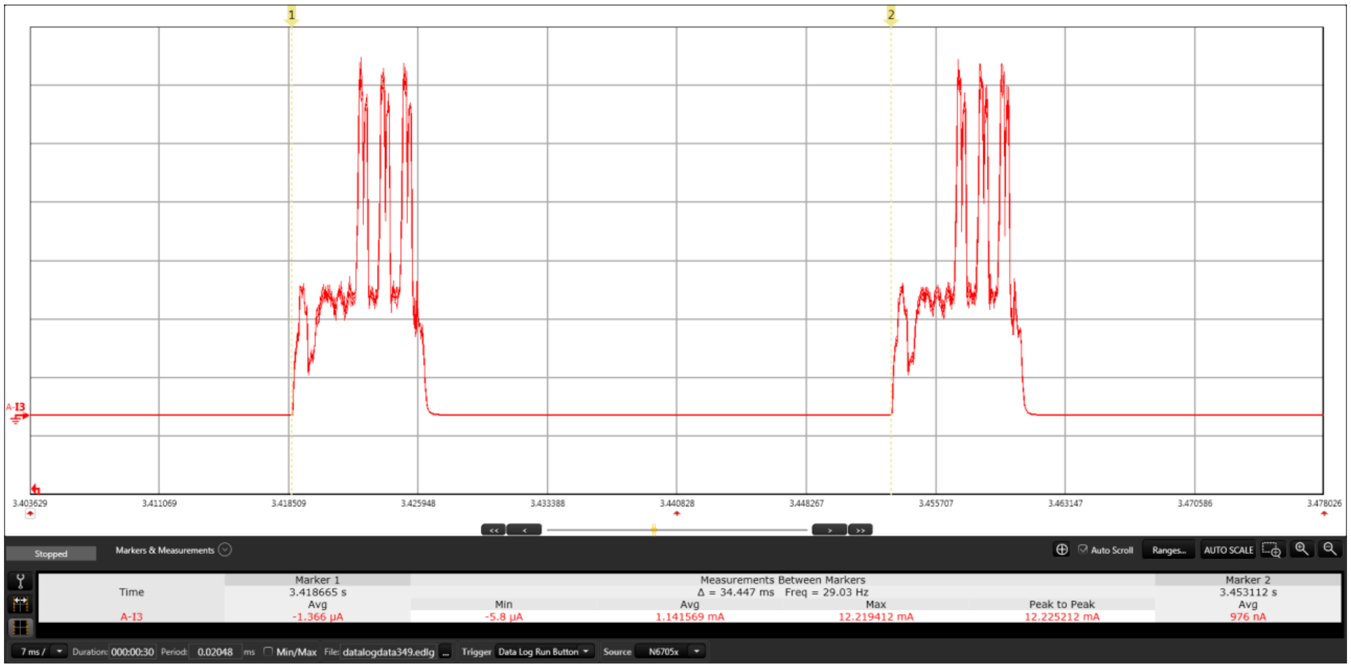


Time	Marker 1	Measurements Between Markers				Marker 2
	531.067 ms	Δ = 2.001715 s Freq = 500 mHz				2.532782 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	1.717 µA	-6.513 µA	8.371 µA	3.717111 mA	3.723624 mA	1.195 µA

286 ms / Duration: 000:00:30 Period: 0.02048 ms Min/Max File: datalogdata349.edlg Trigger Data Log Run Button Source N6705x

Figure 19. KW40Z current consumption in deep sleep mode 3

The next figure captures the current consumption between two fast advertising events.



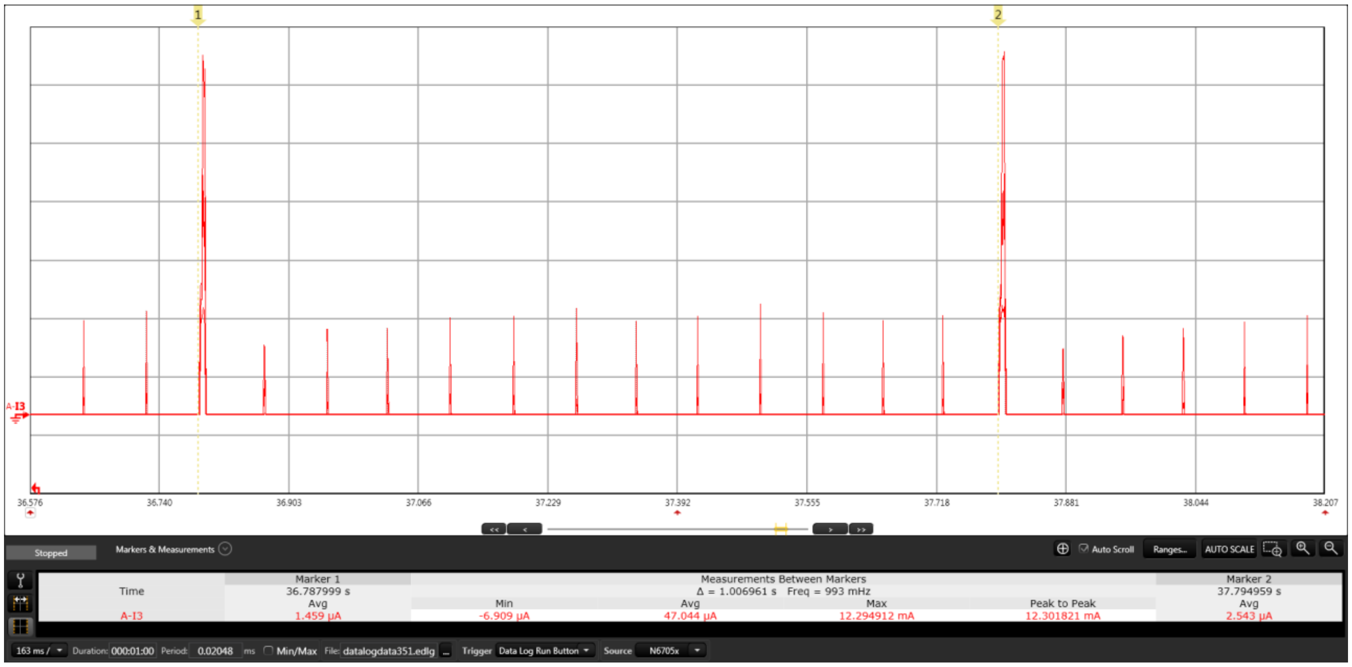
Time	Marker 1	Measurements Between Markers				Marker 2
	3.418665 s	Δ = 34.447 ms Freq = 29.03 Hz				3.453112 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	-1.366 µA	-5.8 µA	1.141569 mA	12.219412 mA	12.225212 mA	976 nA

7 ms / Duration: 000:00:30 Period: 0.02048 ms Min/Max File: datalogdata349.edlg Trigger Data Log Run Button Source N6705x

Figure 20. Current consumption in fast advertising event period

The advertising event peak current is 12.2 mA and the average current between two fast advertise events (including the advertising event itself) is 1.1 mA.

Figure 21 shows similar information, but for the slow advertising packets. The slow advertising interval is about 1 s, the peak current is 12.3 mA and the average current is 47 µA.

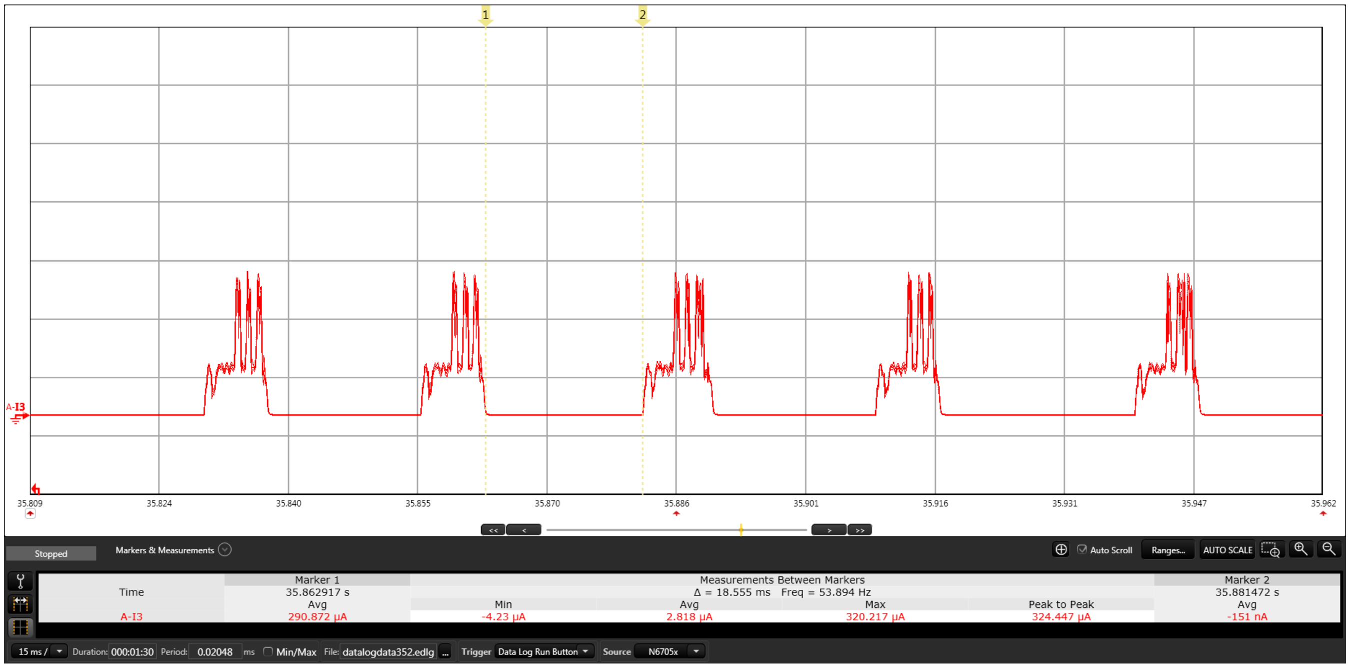


Time	Marker 1 36.787999 s	Measurements Between Markers Δ = 1.006961 s Freq = 993 mHz				Marker 2 37.794959 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	1.459 µA	-6.909 µA	47.044 µA	12.294912 mA	12.301821 mA	2.543 µA

163 ms / Duration: 000:01:00 Period: 0.02048 ms Min/Max File: datalogdata351.edlg Trigger Data Log Run Button Source N6705x

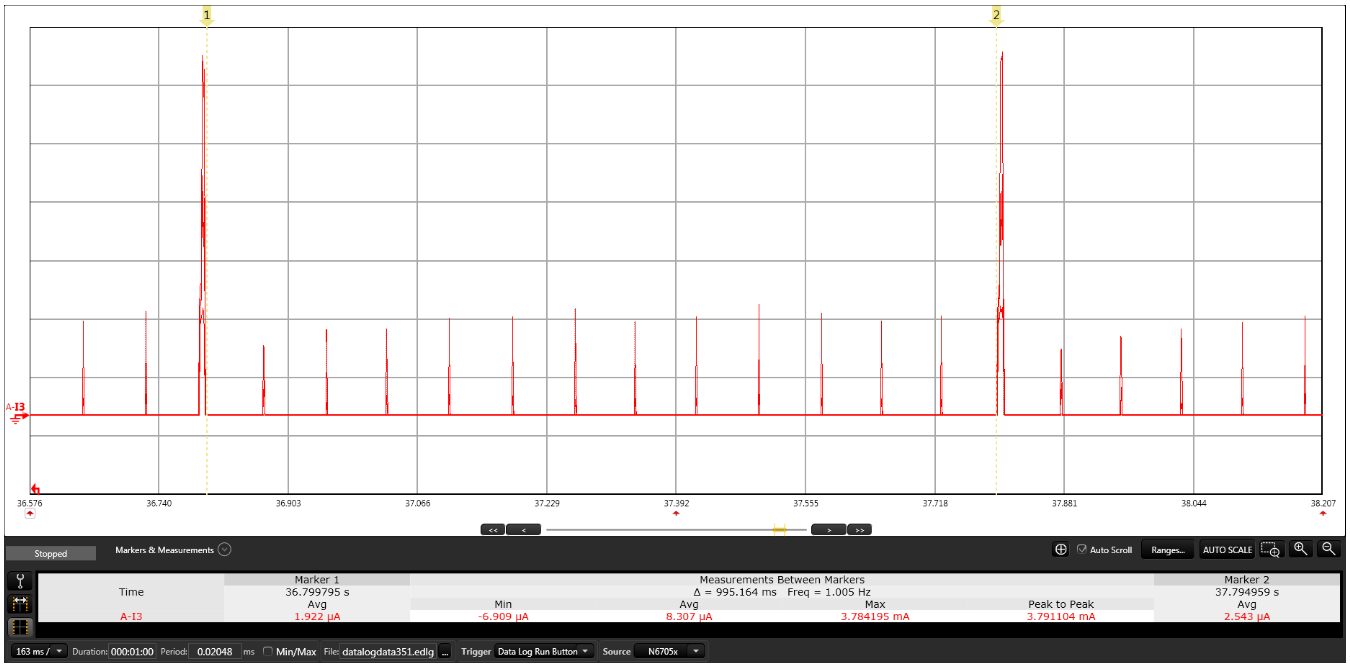
Figure 21. Current consumption over slow advertising event period

Between the advertising events, the system is put to low-power mode 1 (MCU=LLS3, LL=DSM). The low-leakage average current between the fast advertising events is ~ 3 µA, and, between slow advertising events, the leakage currrent has an average of 8.3 µA.



	Marker 1	Measurements Between Markers				Marker 2
Time	35.862917 s	Δ = 18.555 ms Freq = 53.894 Hz				35.881472 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	290.872 µA	-4.23 µA	2.818 µA	320.217 µA	324.447 µA	-151 nA

Figure 22. Current consumption in low-power mode 1 (fast ADV)

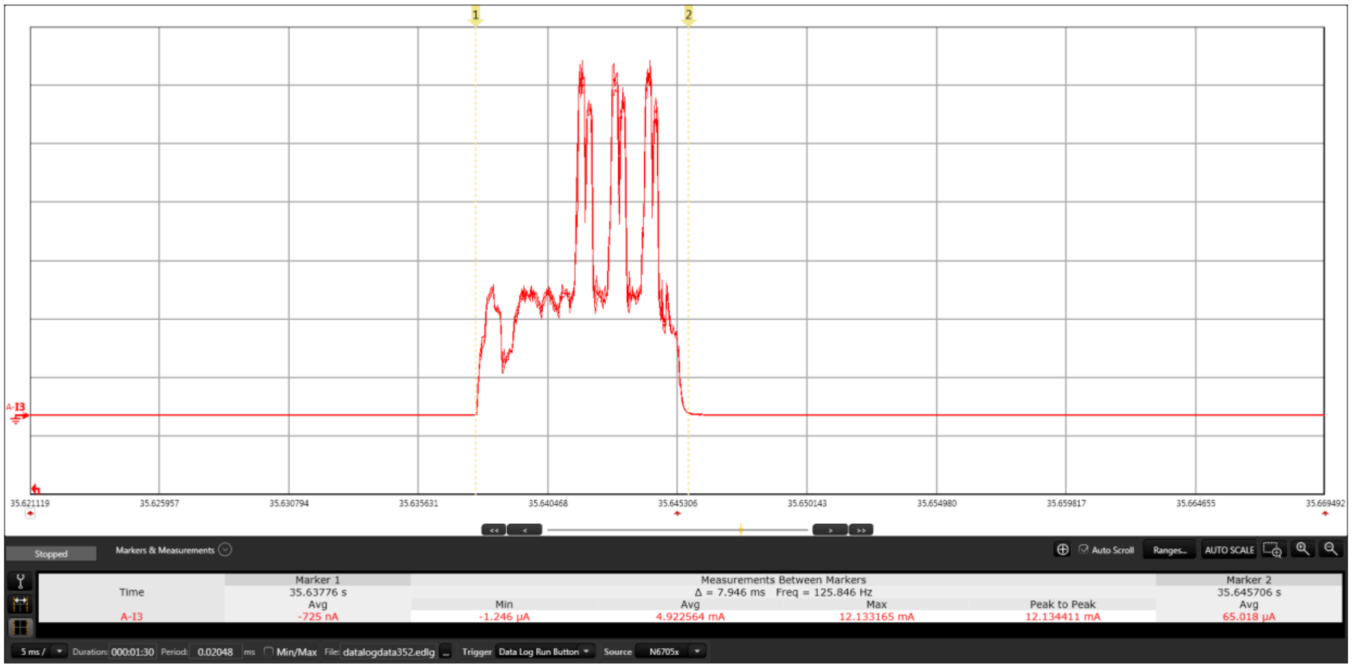


	Marker 1	Measurements Between Markers				Marker 2
Time	36.799795 s	Δ = 995.164 ms Freq = 1.005 Hz				37.794959 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	1.922 µA	-6.909 µA	8.307 µA	3.784195 mA	3.791104 mA	2.543 µA

Figure 23. Current consumption in low-power mode 1 (slow ADV)

The current consumption when the system is in the low-power mode is lower for the fast advertising events than for the slow advertising events. This is caused by the DCDC converter. For the fast advertising events, the time between the wake-up events is so short that the DCDC doesn't have to refresh. For the slow advertising events, the time between the wake-up events is about 1 s and the DCDC has to refresh periodically. The small spikes shown in Figure 23 happen when the DCDC refreshes. This results in a higher average current consumption.

The following figures show the current consumption for the advertising and connection events (both are chosen randomly). The average current for the advertising event is 4.9 mA and the peak current is 12.1 mA.

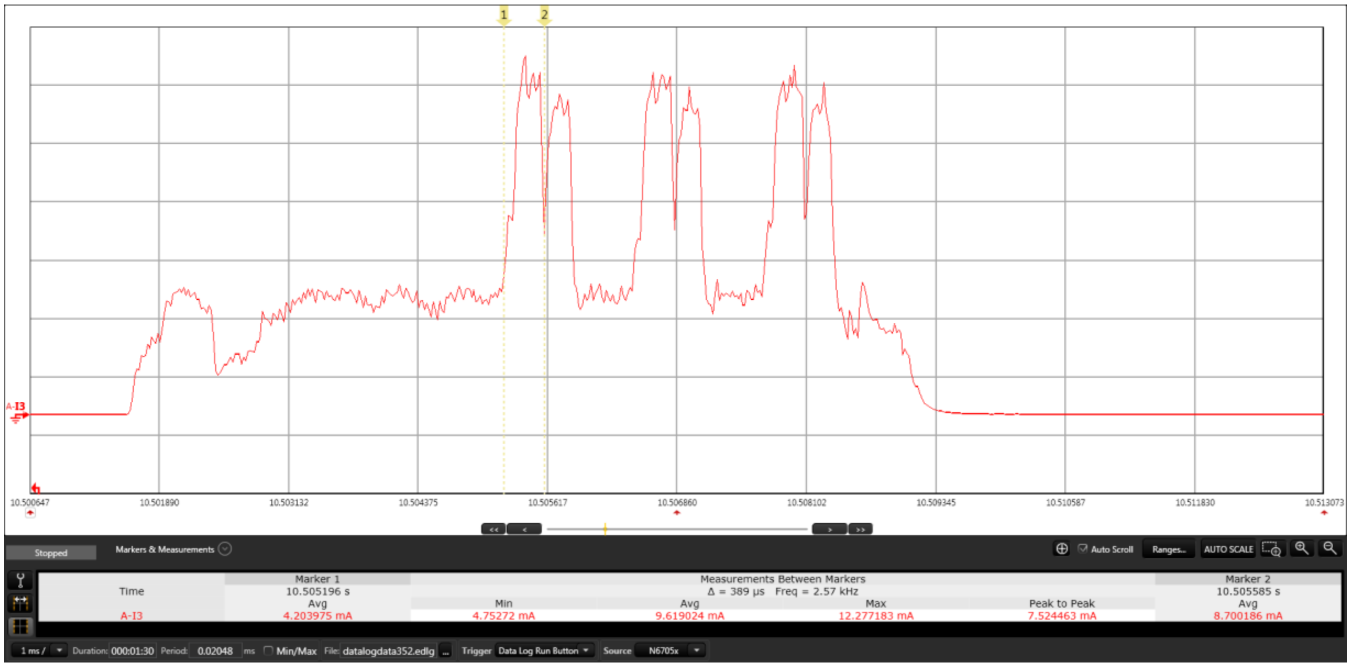


	Marker 1	Measurements Between Markers				Marker 2
Time	35.63776 s	Δ = 7.946 ms Freq = 125.846 Hz				35.645706 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	-725 nA	-1.246 µA	4.922564 mA	12.133165 mA	12.134411 mA	65.018 µA

5 ms / Duration: 000:01:30 Period: 0.02048 ms Min/Max File: datalogdata352.edlg Trigger Data Log Run Button Source N6705x

Figure 24. Advertising event current consumption

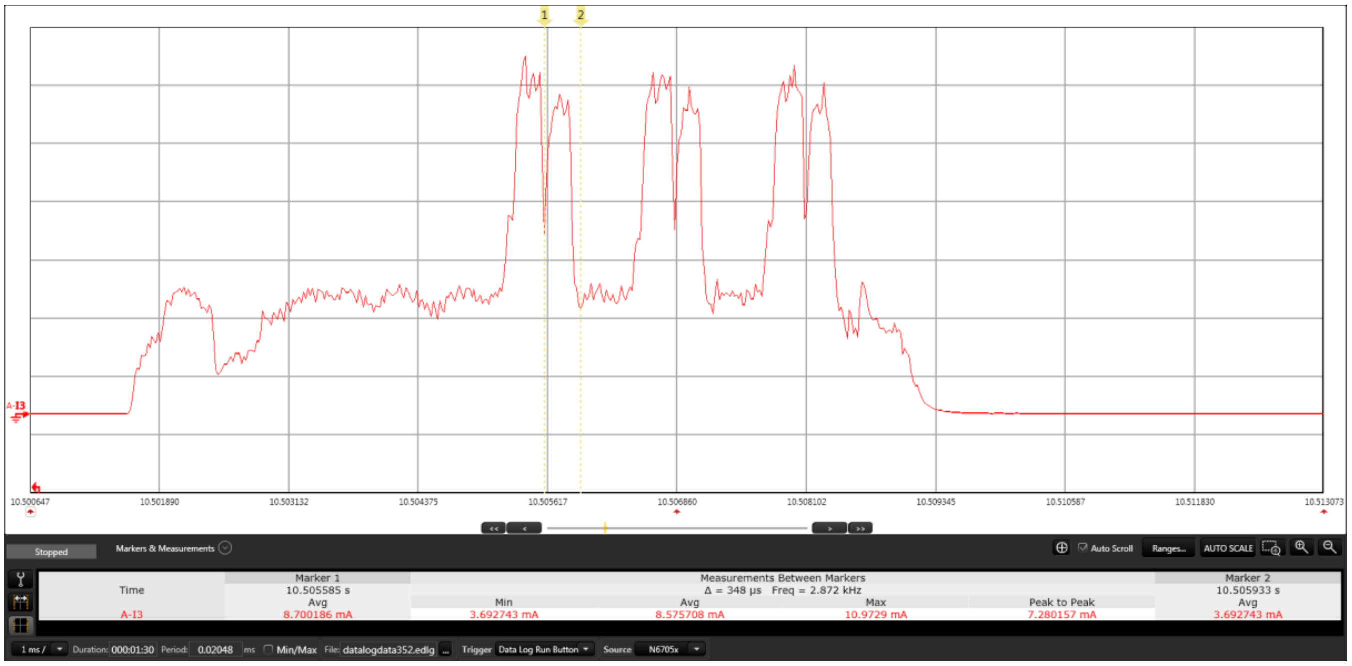
The advertising event consists in a TX-RX sequence for every channel selected in the channel map. In the above figure, all three advertising channels are used.



	Marker 1	Measurements Between Markers				Marker 2
Time	10.505196 s	Δ = 389 µs Freq = 2.57 kHz				10.505585 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	4.203975 mA	4.75272 mA	9.619024 mA	12.277183 mA	7.524463 mA	8.700186 mA

1 ms / Duration: 000:01:30 Period: 0.02048 ms Min/Max File: datalogdata352.edlg Trigger Data Log Run Button Source N6705x

Figure 25. ADV TX phase



	Marker 1	Measurements Between Markers				Marker 2
Time	10.505585 s	Δ = 348 µs Freq = 2.872 kHz				10.505933 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	8.700186 mA	3.692743 mA	8.575708 mA	10.9729 mA	7.280157 mA	3.692743 mA

```
1 ms /   Duration: 000:01:30   Period: 0.02048 ms   Min/Max   File: datalogdata352.edlg   Trigger Data Log Run Button   Source N6705x
```

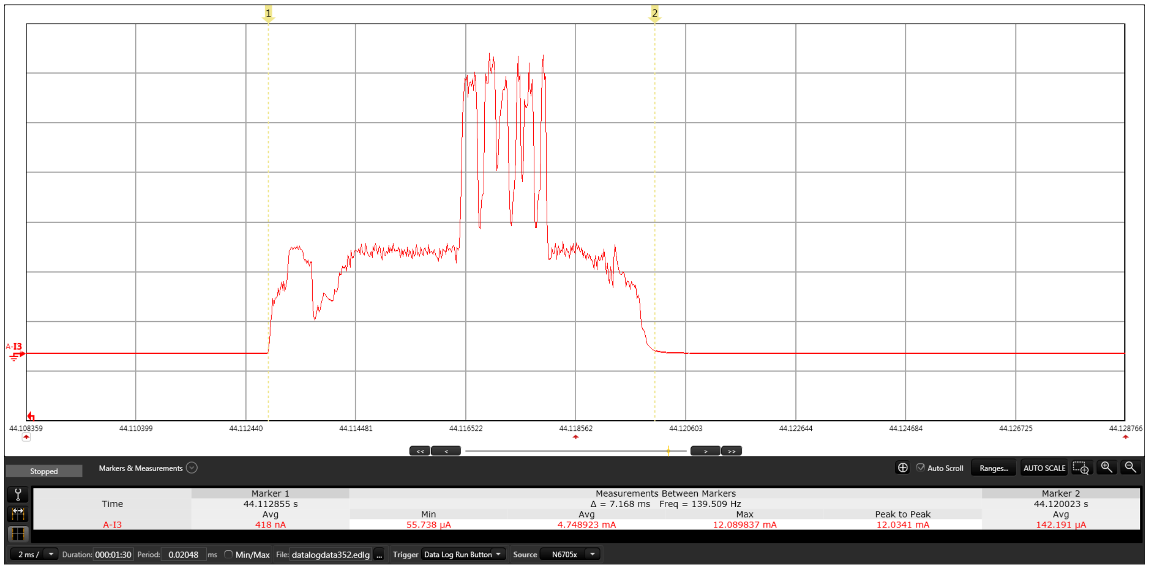
Figure 26. ADV RX phase

The current consumption is summarized in this table:

Table 5. ADV RX and TX currents

—	AVERAGE	PEAK
ADV TX	9.6 mA	12.2 mA
ADV RX	8.5 mA	10.9 mA

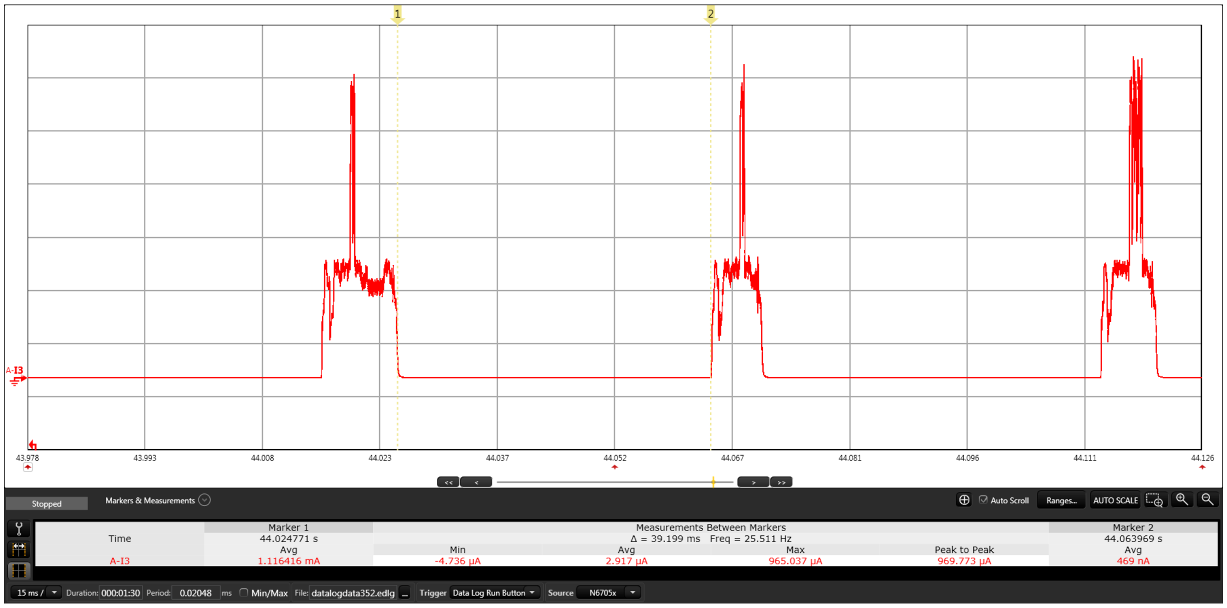
The connection event has an average current of 4.7 mA and a peak current of 12 mA.



	Marker 1	Measurements Between Markers				Marker 2
Time	44.112855 s	Δ = 7.168 ms Freq = 139.509 Hz				44.120023 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	418 nA	55.738 µA	4.748923 mA	12.089837 mA	12.0341 mA	142.191 µA

```
2 ms /   Duration: 000:01:30   Period: 0.02048 ms   Min/Max   File: datalogdata352.edlg   Trigger Data Log Run Button   Source N6705x
```

Figure 27. Connection Event current consumption

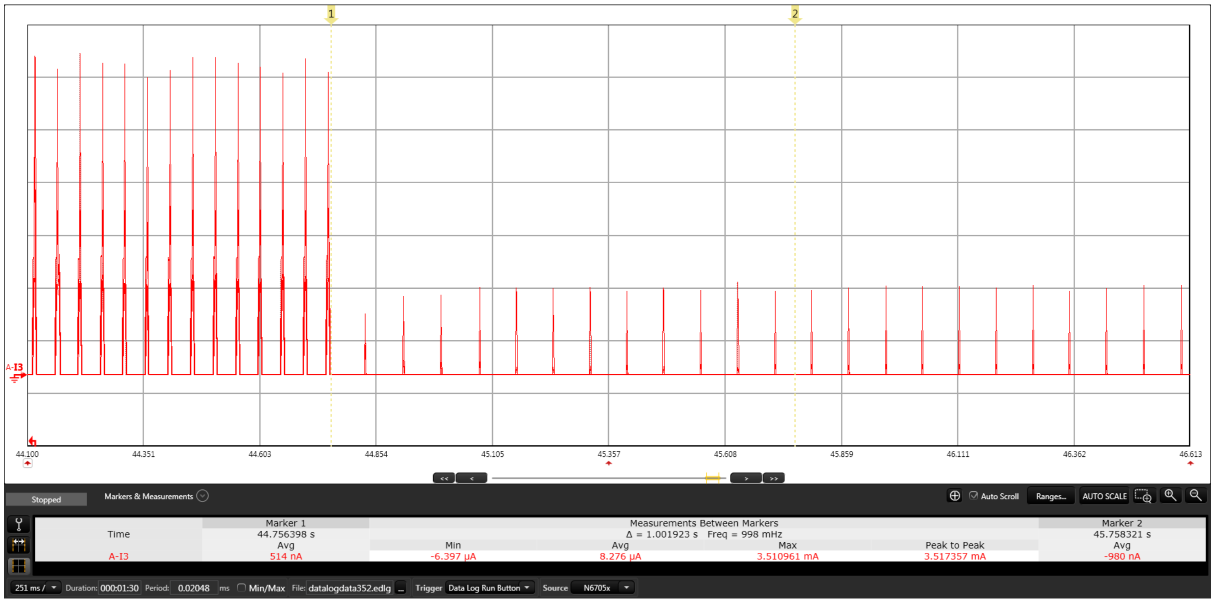


	Marker 1	Measurements Between Markers				Marker 2
Time	44.024771 s	Δ = 39.199 ms Freq = 25.511 Hz				44.063969 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	1.116416 mA	-4.736 µA	2.917 µA	965.037 µA	969.773 µA	469 nA

Figure 28. Current consumption between connection events

Between the connection events, the system enters the low-power mode 1 (MCU=LLS3, LL=DSM) and has an average consumption of ~3 µA.

After disconnecting, the low-power mode is mode 3 (MCU=LLS3, LL=IDLE). In this mode, the average current is 8.3 µA (Δ = 1 s).



	Marker 1	Measurements Between Markers				Marker 2
Time	44.756398 s	Δ = 1.001923 s Freq = 998 mHz				45.758321 s
	Avg	Min	Avg	Max	Peak to Peak	Avg
A-I3	514 nA	-6.397 µA	8.276 µA	3.510961 mA	3.517357 mA	-980 nA

Figure 29. Average current after disconnecting

NOTE

In the connection mode, the BLE LL periodically sends empty packets to maintain the connection with the Central.

4.3. Reports

This table summarizes the measurement performed in the previous sub-section, to make it easier to observe and correlate the numbers:

Table 6. Measurements summary

Measurement	Duration/ timeframe	Average current	Peak current	Reference
System initialization	251 ms	3.76 mA	4.79 mA	Figure 18
Sleep mode 3 (post initialization)	2 s	8.37 µA	3.7 mA	Figure 19
Fast ADV (full period)	34 ms	1.14 mA	12.22 mA	Figure 20
Slow ADV (full period)	1 s	47 µA	12.29 mA	Figure 21
Sleep mode 1 (between fast ADV events)	18.5 ms	2.81 µA	320.2 µA	Figure 22
Sleep mode 1 (between slow ADV events)	995 ms	8.3 µA	3.78 mA	Figure 23
ADV event	7.9 ms	4.92 mA	12.2 mA	Figure 24
ADV TX	389 µs	9.61 mA	12.2 mA	Figure 25
ADV RX	348 µs	8.57 mA	10.97 mA	Figure 26
CONN Event	7.1 ms	4.74 mA	12 mA	Figure 27
Sleep mode 1 (between CONN events)	39 ms	2.91 µA	965 µA	Figure 28
Disconnect	1 s	8.2 µA	3.51 mA	Figure 29

5. Coin-Cell Battery Life Power Profile

Within a wireless network, most of the devices are battery powered while only few are supplied from the mains (such as network coordinators, gateways, or routers—devices with high processing power that must be fully operational all of the time).

In the case of battery-powered devices, there are many battery types that you can use: alkaline, lithium, manganese-lithium, lithium-ion, flow, lead-acid, nickel-oxy-hydroxide, zinc-carbon, nickel-cadmium, nickel-zinc, polymer-based, and other types.

Nowadays, the lithium-based batteries start to be the common choice of customers. Lithium batteries (not lithium-ion) are disposable batteries that have a high charge density (long life) and typically provide voltages from 1.5 V to 3.7 V. The manganese-lithium batteries are also rechargeable.

One of the main advantages of the lithium batteries is that they have an extremely flat discharge curve due to a very low self-discharge rate. They have high efficiency, long cycle, high load capabilities, and require no maintenance. However, the effective capacity of the battery cell is reduced if it is discharged at very high rates. The peak current that the battery can deliver is directly proportional to the internal ESR. A high peak current reduces the battery terminal voltage by an amount equal to the ESR multiplied by the peak current:

$$V_{loss} = I_{peak} \times ESR$$

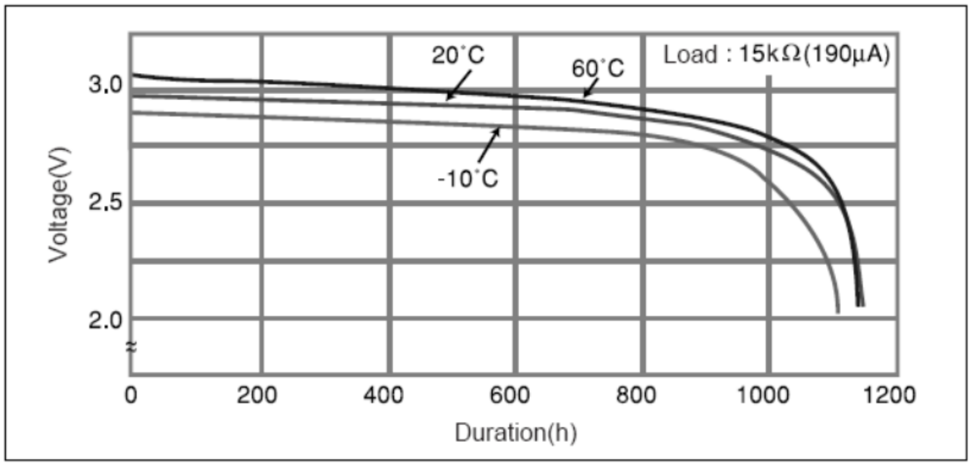


Figure 30. VARTA CR2032 battery discharge rate

The FRDM-KW40Z board is equipped with a CR2032 battery holder on its bottom side. The CR2032 lithium battery is selected for the analysis, calculations, and measurements within this sub-section.

The selected battery has a nominal voltage of 3 V and a typical capacity of 230 mAh. This means that a 230 mA current drains the battery in one hour. To evaluate the battery lifetime, compute the average current consumption of the given application scenario.

5.1. Bluetooth Smart application

5.1.1. Heart Rate Sensor

The scenario for this application is:

- Kinetis BLE Toolkit application starts, Heart Rate Sensor is running (the phone/tablet is in the scan mode).
- A CR2032 coin-cell battery is pushed into the FRDM-KW40Z battery holder.
- After three seconds, the SW3 push-button is pressed. The device starts to advertise on the advertising channels.
- When the device is listed on the phone/tablet, the user connects to it (advertising ends, connection events start).
- The heart rate measurement is reported once per second, the battery level is reported every 10 seconds.
- The setup is left to run indefinitely (until the battery sensor is exhausted).

The PDU length of packets is:

- Advertising packet—24 B.
- Heart rate reporting—17 B.
- Battery level (read)—9 B.
- Battery level (response)—8 B.

To estimate the battery life, compute the energy consumed for the above phases.

- System initialization:

$$43.76 \text{ mA} \times 251 \text{ ms} = 3.76 \text{ mA} \times 0.\ 000069722 \text{ h} = 0.0002621472 \text{ mAh}$$

- First deep sleep (just after the initialization):

$$8.37 \text{ µA} \times 2 \text{ s} = 0.00837 \text{ mA} \times 0.000555555 \text{ h} = 0.00000464999535 \text{ mAh}$$

- During wakeup (before the advertising):

$$4 \text{ mA} \times 150 \text{ ms} = 4 \text{ mA} \times 0.000041667 \text{ h} = 0.000166668 \text{ mAh}$$

- During one advertising period (random sample):

$$(4.9 \text{ mA} \times 8 \text{ ms}) + (2.8 \text{ µA} \times 26 \text{ ms}) = (4.9 \text{ mA} \times 0.000002222 \text{ h}) + (0.0028 \text{ mA} \times 0.000007222 \text{ h}) =$$

$$0.0000108878 + 0.0000000202216 = 0.0000109080216 \text{ mAh}$$

- During five seconds of advertising (extrapolation):

$$(5000 \text{ ms} / 34 \text{ ms}) \times 0.0000109080216 \text{ mAh} = 0.001604 \text{ mAh}$$

Before connection, the system consumed 0.002037 mAh of the battery energy. The remaining battery energy is:

$$230 \text{ mAh} - 0.002037 \text{ mAh} = 229.9979 \text{ mAh}$$

Supposing that all the remaining energy is consumed for reporting the heart rate value, the math is as follows:

- The average current consumption during a connection event (including LL keep-alive transmissions) is:

$$(4.7 \text{ mA} \times 7.1 \text{ ms}) + (2.9 \text{ µA} \times 39 \text{ ms}) / 46.1 \text{ ms} = 0.7263 \text{ mA}$$

- The estimated battery life is:

$$229.9979 \text{ mAh} / 0.7263 \text{ mA} = 316.6 \text{ hours}$$

Consider that the result above is valid for the continuous connection mode.

Based on the numbers provided in Table 4, the battery lifetime can be determined for a user-defined scenario. For most of the real-life scenarios, the battery life is extended to years.

6. Acronyms and Abbreviations

Table 7. Acronyms and abbreviations

Acronym/abbreviation	Description
ADC	Analog to Digital Converter
ARM	Advanced RISC Machine (RISC – Reduced Instruction Set Computer)
BLE	Bluetooth Low-Energy (Bluetooth Smart)
BPSK	Binary Phase Shift Keying
BTLL	Bluetooth Link Layer

Table 7. Acronyms and abbreviations

Acronym/abbreviation	Description
CMP	Comparator module
DAC	Digital to Analog Converter
DC	Direct Current
DSM	Deep-Sleep Mode
DUT	Device Under Test
ESR	Equivalent Series Resistance
FRDM	Freedom board
GAP	Generic Access Profile
GFSK	Gaussian Frequency Shift Keying
GPIO	General-Purpose Input/Output
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
ISM	Industrial, Scientific, and Medical bands
LE	Low Energy
LL	Link Layer
LLS	Low-Leakage Stop
LLWU	Low-Leakage Wake-up Unit
LPTMR	Low-Power Timer
LPUART	Low-Power UART
MBAN	Medical Body Area Network
MCU	Microcontroller Unit
O-QPSK	Offset Quadrature Phase Shift Keying
PC	Personal Computer
PDU	Protocol Data Unit
PMC	Power Management Controller
RX	Reception
SAR	Successive Approximation Register ADC
SCGC	System Clock Gating Control register
SIM	System Integration Module
SMPS	Switched Mode Power Supply
SRAM	Static Random Access Memory
TMR	Timer
TSM	Transceiver Sequence Manager
TX	Transmission
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Bus
VLLS	Very Low Leakage Stop
XCVR	Transceiver

7. Revision History

Revision number	Date	Substantive changes
0	03/2016	Initial release.
1	09/2016	Measurements done at 3.6 V instead of 3 V. Updated low-leakage currents. Added the ADV TX and ADV RX current measurements. Updated the battery life power profile example. Updated the Android application screenshots.

Document Number: AN5272
Rev. 1
09/2016