

# Kinetis MKW34/MKW35/MKW36 Bluetooth®

## Low Energy Software

### Quick Start Guide

#### Introduction

This document is a brief description of the NXP Bluetooth® Low Energy Software for the MKW34/MKW35/MKW36 wireless microcontroller platforms, version 1.3.8. This software package is built using the Kinetis Software Development Kit (KSDK) version 2.2.4. This document covers hardware setup, build and usage of the provided demo applications.

#### Contents

<b>Introduction</b>	1
1 Installing the Connectivity Package	2
2 Building the Binaries	3
2.1 Building and Flashing the BLE Software Demo Applications using IAR.....	5
2.2 Building and Flashing the BLE Software Demo Applications using MCUXpresso IDE...	7
3 Hardware Setup	12
4 Running the Heart Rate sensor application using <i>IoT Toolbox</i> mobile application	14



# 1 Installing the Connectivity Package

This section details the steps to install the connectivity package.

First, configure and download the package archive from the staging system on the <https://mcuxpresso.nxp.com> website or simply download the pre-created package archive if it is available on the same website.

Unpack the contents of the archive to a folder on the local disk.

## NOTE

It is recommended to use the default location for the package (C:\NXP) and create a subfolder there specific to each device and release.

## 2 Building the Binaries

This section details the required steps for creating the binary files to be downloaded to the boards.

### NOTE

To be able to build any of the demo applications you need one of the following toolchains:

- IAR Embedded Workbench for ARM® (version 8.32.4 or higher)
- MCUXpresso IDE (version 10.3.1 or higher)

This connectivity software package does not include support for any other toolchains.

The packages must be built with the debug configuration to enable debugging information.

This package includes various demo applications that can be used as a starting point.

The next section presents the steps required for building the *heart\_rate\_sensor* application. All applications can be found using the following placeholders for text:

- <connectivity\_path>: represents the root path for the SDK.
- <board>: represents the target board for the demo app, can be “frdmkw36”
- <RTOS>: represents the scheduler or RTOS used by the app; it can be either “bm” or “freertos”
- <demo\_app>: represents the demo app name
- <IDE>: represents the integrated development environment used to build projects and can be “iar”.

The demo applications general folder structure is the following:

```
<connectivity_path>\boards\<board>\wireless_examples\bluetooth\<demo_app>\<RTOS>\<IDE>\
```

---

### Kinetis Bluetooth Low Energy Software Demo Application Build Example

---

Selected app: heart\_rate\_sensor

Board: frdmkw36

RTOS: FreeRTOS

Resulting location:

```
<connectivity_path>\boards\frdmkw36\wireless_examples\bluetooth\heart_rate_sensor\freertos\  
<IDE>
```

## NOTE

If your FRDM-KW36 board is configured for the buck or boost modes of the KW36Z DCDC converter, please note that the following defines need to be set: *gDCDC\_Enabled\_d* to 1 and *APP\_DCDC\_MODE* to *gDCDC\_Mode\_Buck\_c* or *gDCDC\_Mode\_Boost\_c* respectively, in the *app\_preinclude.h* header file.

The Heart Rate sensor application is configured for low power operation by default. In this case, when the core enters in sleep mode, the debug probe will fail. If you want to disable the low power functionality, the *cPWR\_UsePowerDownMode* preprocessor definition needs to be set to 0 in the *app\_preinclude.h* header file.

## 2.1 Building and Flashing the BLE Software Demo Applications using IAR

### Step 1:

Navigate to the resulting location in the SDK root directory.

### Step 2:

Open the highlighted IAR workspace file (\*.eww file format):

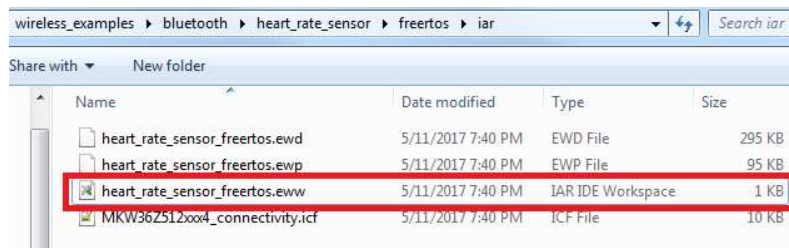


Figure 1: “Heart Rate Sensor” IAR demo project location

### Step 3:

Select the Heart Rate Sensor project.

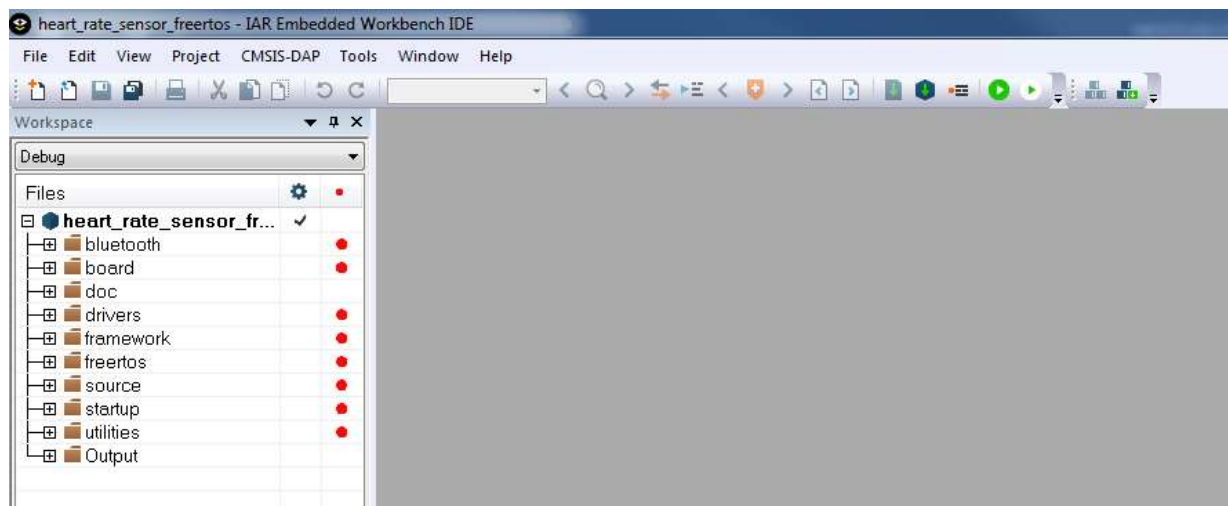


Figure 2: Heart Rate Sensor FreeRTOS IAR project

#### Step 4:

Build the Heart Rate Sensor project.

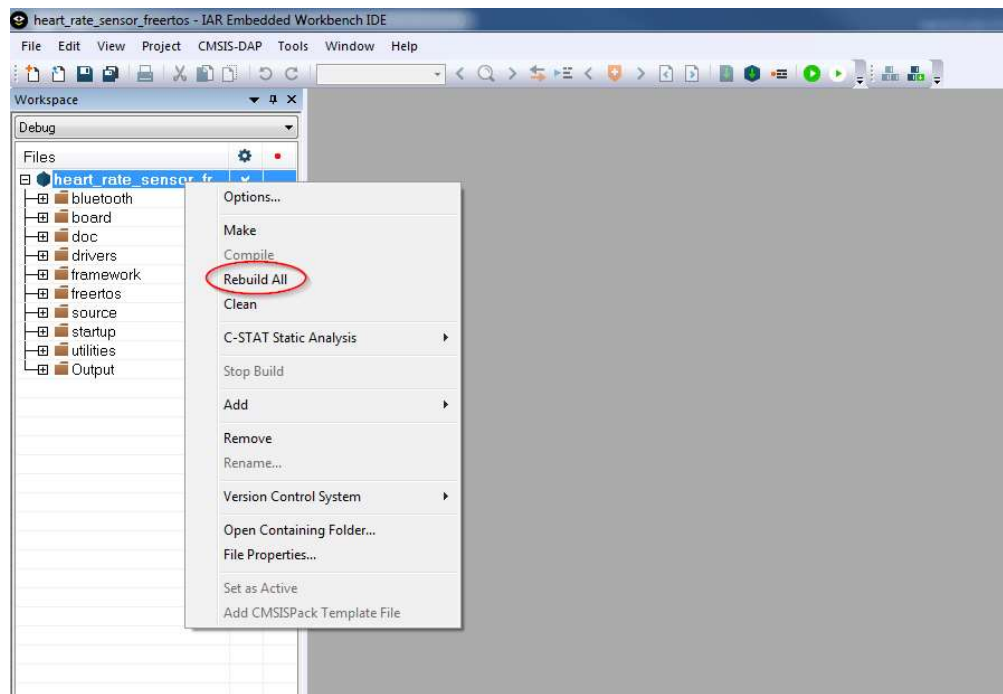


Figure 3: Heart Rate Sensor build

#### Step 5

Make the appropriate debugger settings in the project options window:

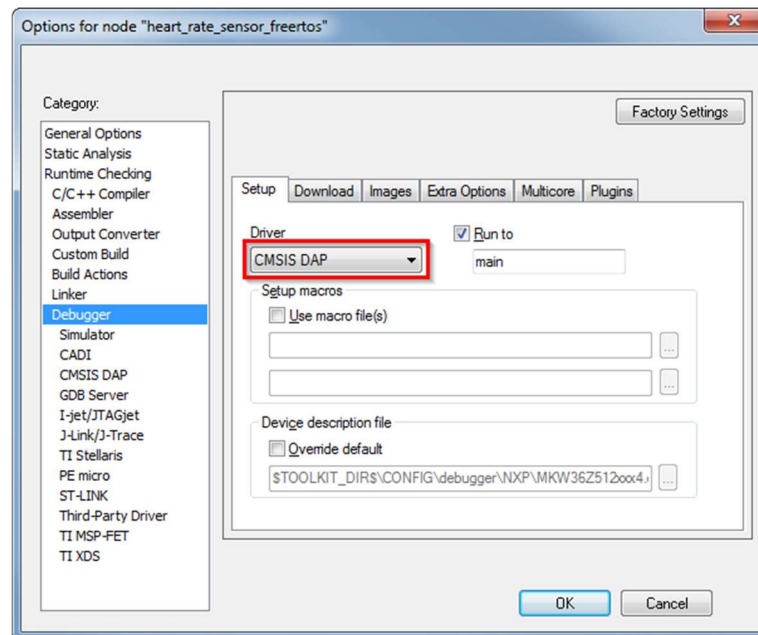
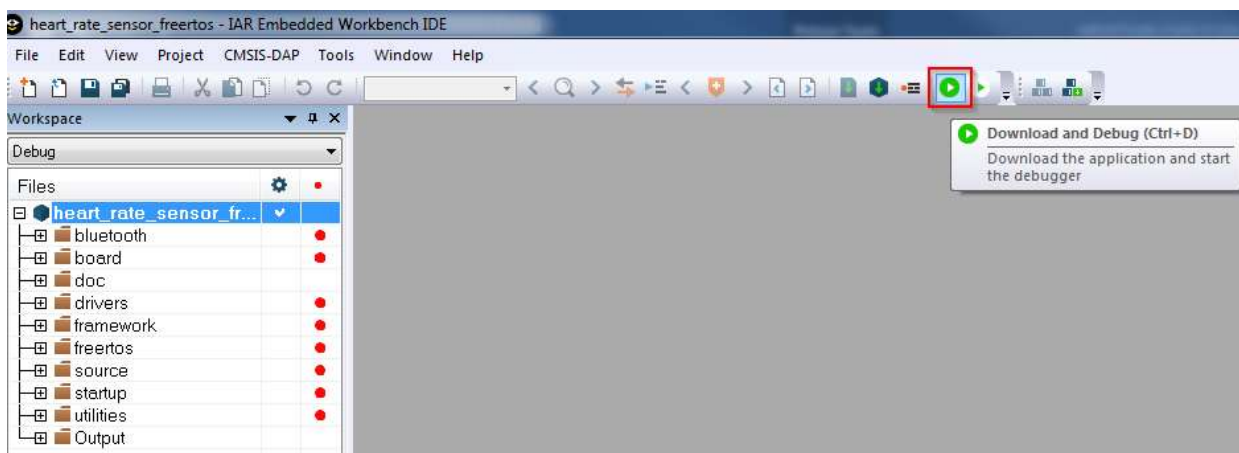


Figure 4: Debugger Settings

## Step 6:

Click the “Download and Debug” button to flash the executable onto the board.



**Figure 5: Heart Rate Sensor Download and Debug**

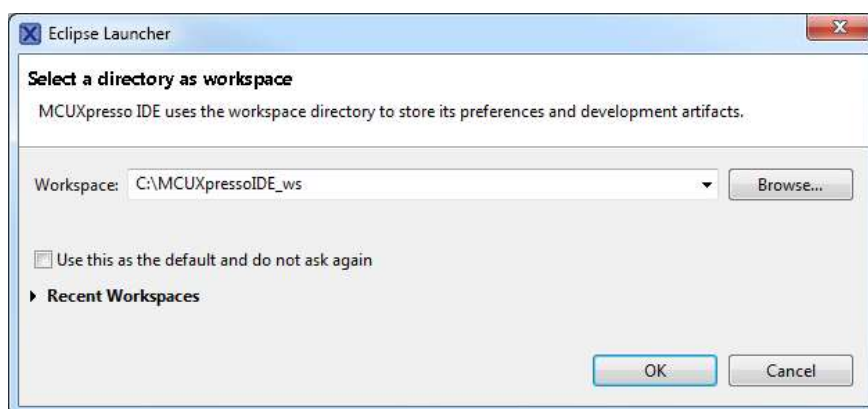
### NOTE

The projects are configured to use “CMSIS DAP” as the default debugger. Please make sure that your board’s OpenSDA chip contains a CMSIS DAP firmware or that the debugger selection corresponds to the physical interface used to interface to the board. See the section below for more information.

## 2.2 Building and Flashing the BLE Software Demo Applications using MCUXpresso IDE

### Step 1:

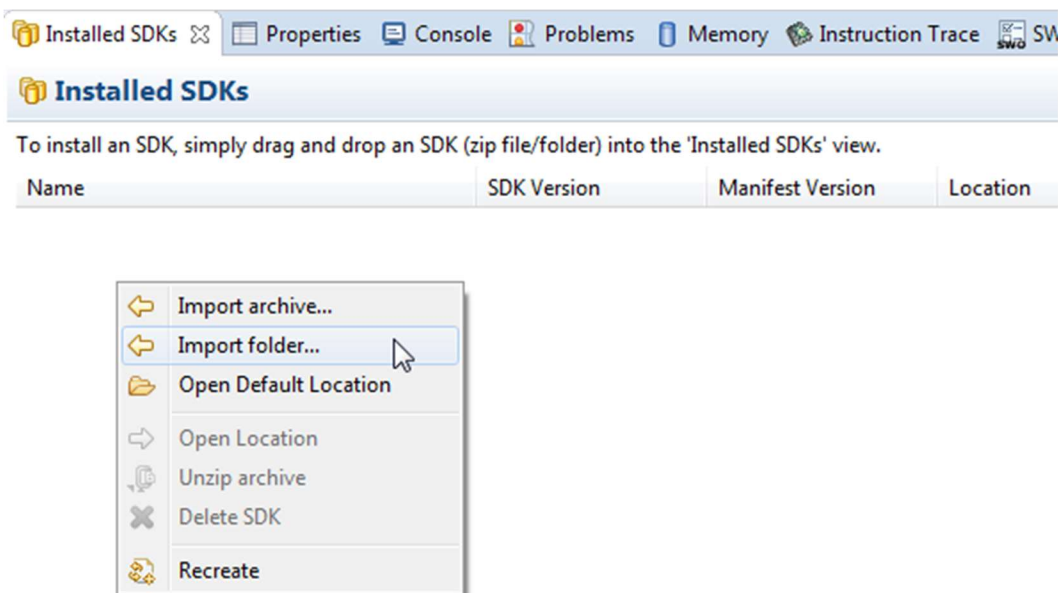
Open MCUXpresso IDE and open an existing or new workspace location.



**Figure 6: Select a MCUXpresso IDE workspace location**

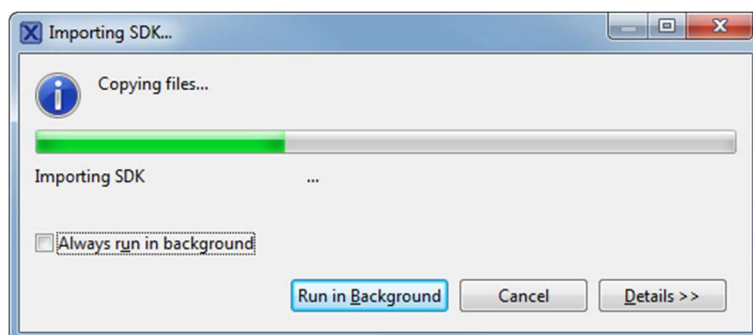
## Step 2:

Right click in the “*Installed SDKs*” tab and then click “*Import Folder*”.



**Figure 7: Import a new SDK from a folder**

When the Browse dialog appears, go to the location of the unpacked archive and select the folder. The SDK will be imported.

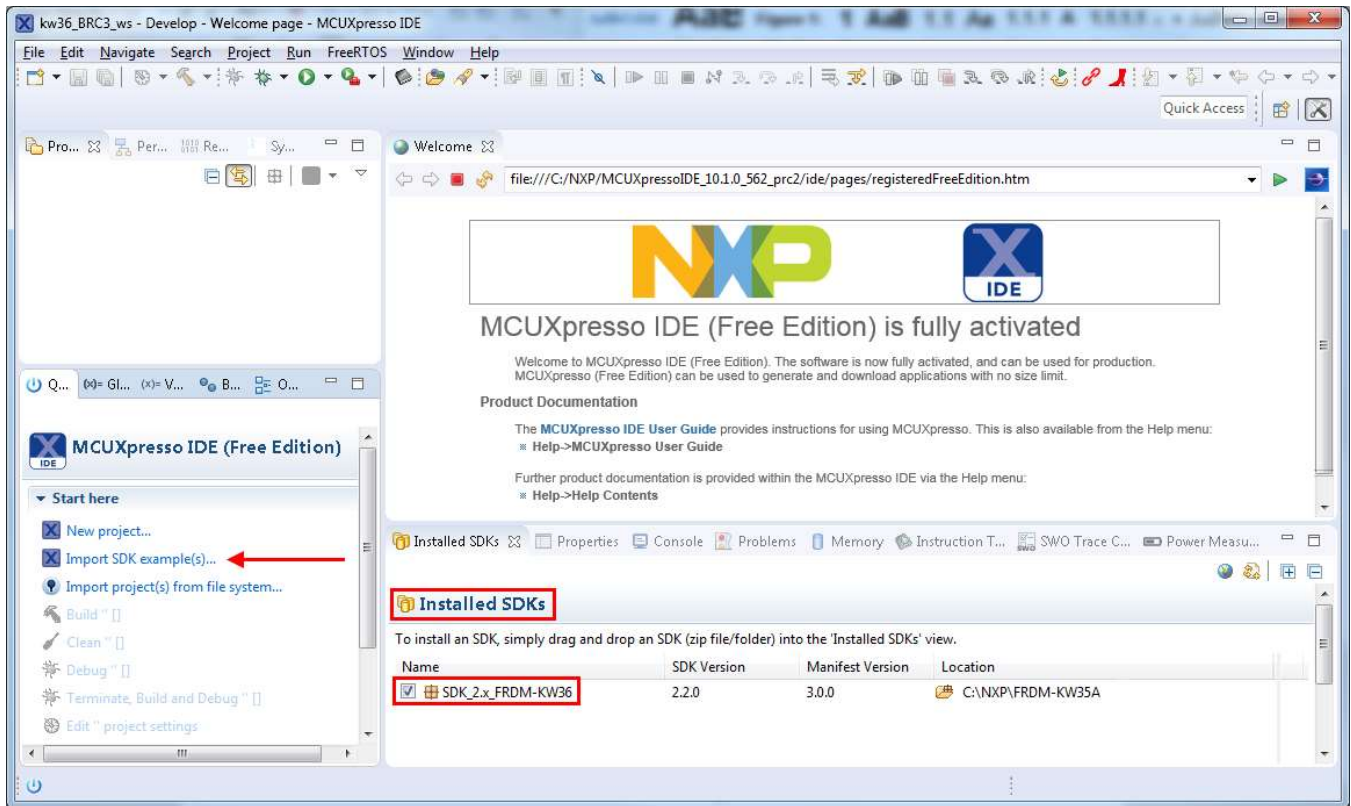


**Figure 8: Importing the SDK**

## Step 3:

After the SDK is loaded successfully elect the Import the SDK examples(s)...” option to add examples to your workspace.





**Figure 9: Installed SDKs**

#### Step 4:

Select the board, then the desired example(s):

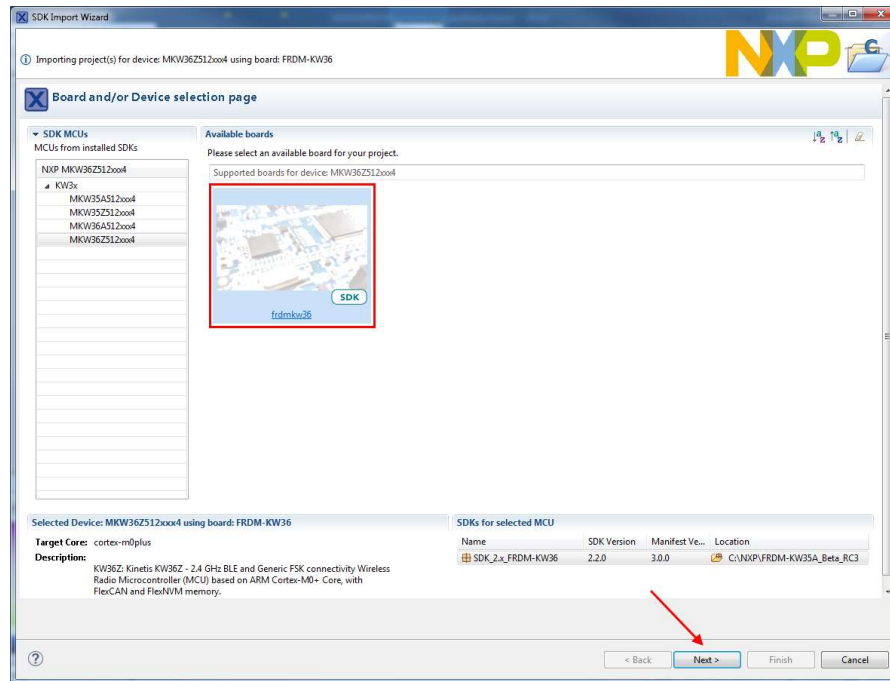


Figure 10: Select the board

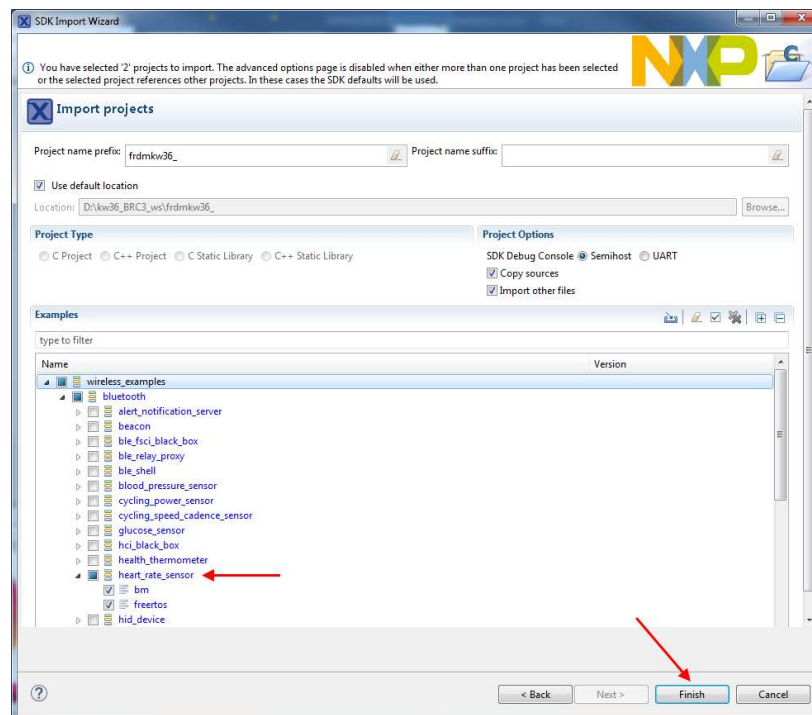


Figure 11: Select the example(s)

## Step 5:

Build the heart\_rate\_sensor project.

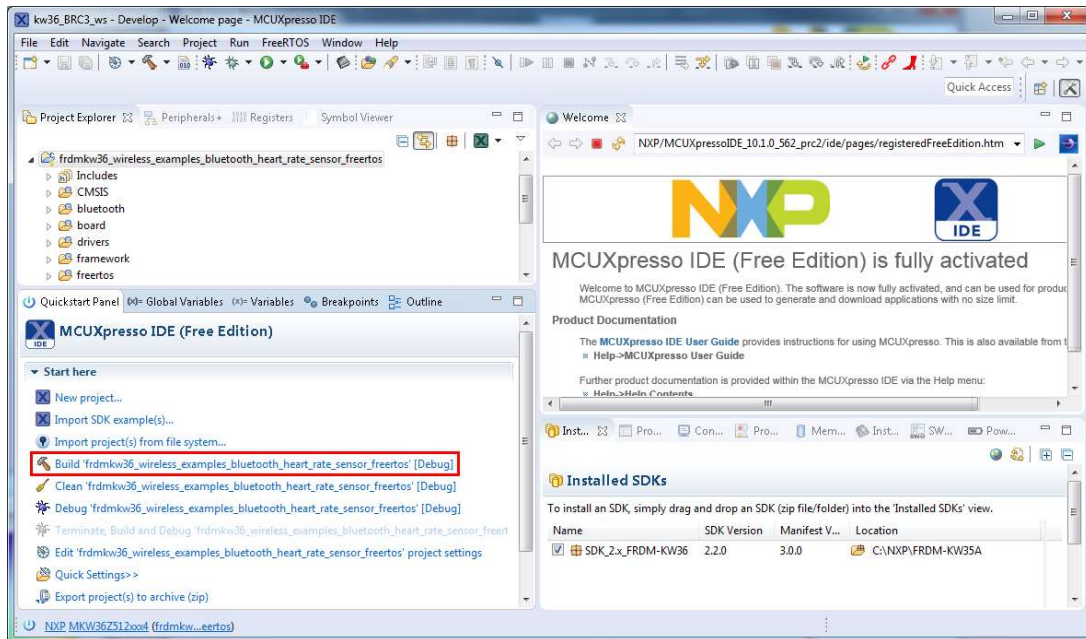


Figure 12: Build the “heart\_rate\_sensor” FreeRTOS project

## Step 6:

Click the “Debug” button to download the executable onto the board.

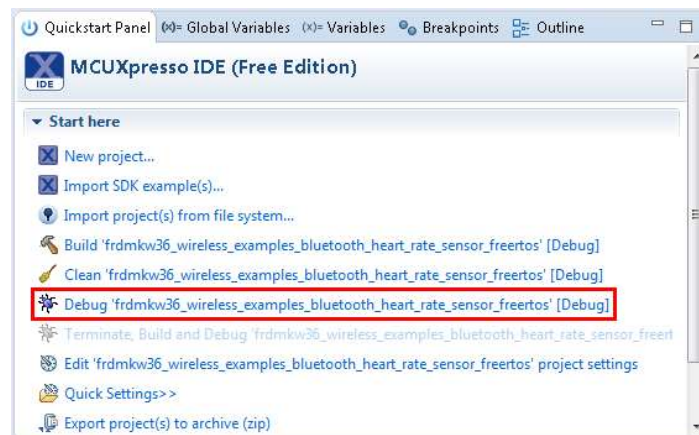


Figure 13: Download and debug the “heart\_rate\_sensor” FreeRTOS project

### 3 Hardware Setup

The hardware setup in this example uses a FRDM-KW36 development platform, shown in the figure below:



**Figure 14: FRDM-KW36**

The FRDM-KW36 board should have the OpenSDA USB port connected to a Windows PC. The OpenSDA chip on the board should have appropriate firmware flashed, with debugging and virtual serial COM port capabilities. For more information on OpenSDA please refer to the following webpage: [www.nxp.com/opensda](http://www.nxp.com/opensda).

Variants of embedded firmware for the OpenSDA chip can be downloaded from:

<https://github.com/mbedmicro/CMSIS-DAP>  
<https://www.segger.com/opensda.html>

CMSIS DAP is the default interface selected in the IAR Embedded Workbench for ARM® projects with FRDM-KW36 included in this release.

The FRDM-KW36 board can be configured via jumpers to be in the two modes of the DCDC converter inside the KW36Z microcontroller or to bypass it entirely, as shown in the figure below:

## Power Configuration

Default: Buck Mode (auto start).

	PSW_CFG J38	REG_CFG J27, J30
<b>Bypass Mode (auto start)</b> VDCDC_IN (1.71 to 3.6V) Operation 1.8V - 3.6 V	1~2	J27-on J30-on
<b>Buck Mode (manual start)</b> VDCDC_IN (1.8V to 4.2V) Coin Cell Battery Operation	1~2 press SW4 to start	J27-off J30-off
<b>Buck Mode (auto start)</b> VDCDC_IN (1.8V to 4.2V) Coin Cell Battery Operation	2~3	J27-off J30-off

Figure 15: FRDM-KW36 Jumper Configuration for DCDC Modes

## 4 Running the Heart Rate sensor application using *IoT Toolbox* mobile application

This section does not cover the installation of the *IoT Toolbox* mobile application and it assumes you have already installed the latest version from the application store.

### Step 1:

Flash the board with the Heart Rate sensor application as previously described. If low power support is enabled, no LED's will be flashing.

### Step 2:

Make sure your phone's Bluetooth Device is enabled.

### Step 3:

Open the *IoT Toolbox* application and select the *Heart Rate* icon.



Figure 16: Kinetis BLE Toolbox - opening Heart Rate application

### Step 4:

Press **SW2** on the FRDM-KW36 board to start advertising. The device should become visible for the Heart Rate application.

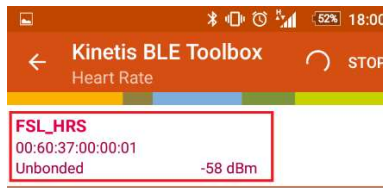


Figure 17: Selecting the scanned device

#### Step 5:

Select the device that appears in the *Heart Rate* tab to connect to it. After connecting, the mobile application starts plotting the randomly generated heart rate values sent by the FRDM-KW36 device.

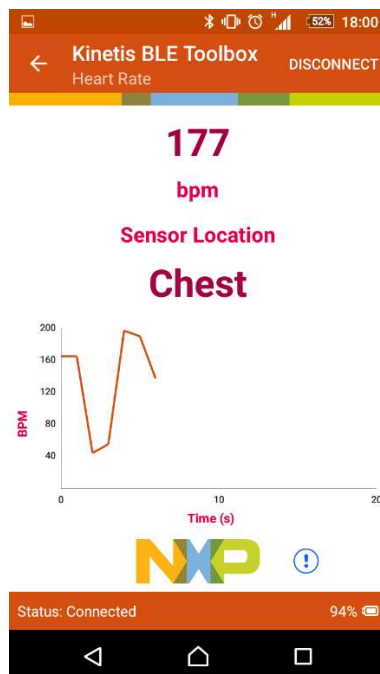


Figure 18: Heart Rate sensor device functionality in connected mode



#### How to Reach Us:

##### Home Page:

[www.nxp.com](http://www.nxp.com)

##### Web Support:

[www.nxp.com/support](http://www.nxp.com/support)

Information in this document is provided solely to enable system and software implementers to use Freescale 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. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale 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 Freescale 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. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: [nxp.com/SalesTermsandConditions](http://nxp.com/SalesTermsandConditions).

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, Altivec, C-5, CodeTest, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, and UMEMS are trademarks of NXP B.V. All other product or service names are the property of their respective owners.

Bluetooth® low energy is a trademark of the Bluetooth Special Interest Group (SIG). This product is not endorsed or approved by the Bluetooth SIG. All other product or service names are the property of their respective owners. ARM, the ARM powered logo, and Cortex are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere..

© 2020 NXP B.V.

