NXP Semiconductors
User's Guide

Document Number: MKW41ZBLESW122QSG
Rev. 0,
09/2016

Kinetis MKW41Z Bluetooth® Low Energy Software

Quick Start Guide

Introduction

This document is a brief presentation of the NXP Bluetooth® Low Energy Software for the KW41Z wireless microcontroller platforms version 1.2.2. This document covers installation of the software packages, hardware setup, build and usage of the provided demo applications.

Contents

1 Installation

This section covers the steps for a successful installation of the connectivity software.

1.1 MKW41Z Bluetooth® Low Energy Software Installation

Execute the installer and follow the steps presented in the example.

NXP MKW41Z BLUETOOTH® LOW ENERGY Software Installation Example

The first page is just a preamble for the installation. Choose next to continue.

Figure 1: BLE software installation wizard first screen

The next page is the license agreement. If you accept the terms and conditions, please select “I Agree” to continue the installation.

Figure 2: BLE software installation wizard license screen

The next step is to select the install location for the NXP BLE software. By default, the installer uses “C:\NXP\MKW41Z_ConnSw_1.0.2”, but this may be changed, depending on your needs.

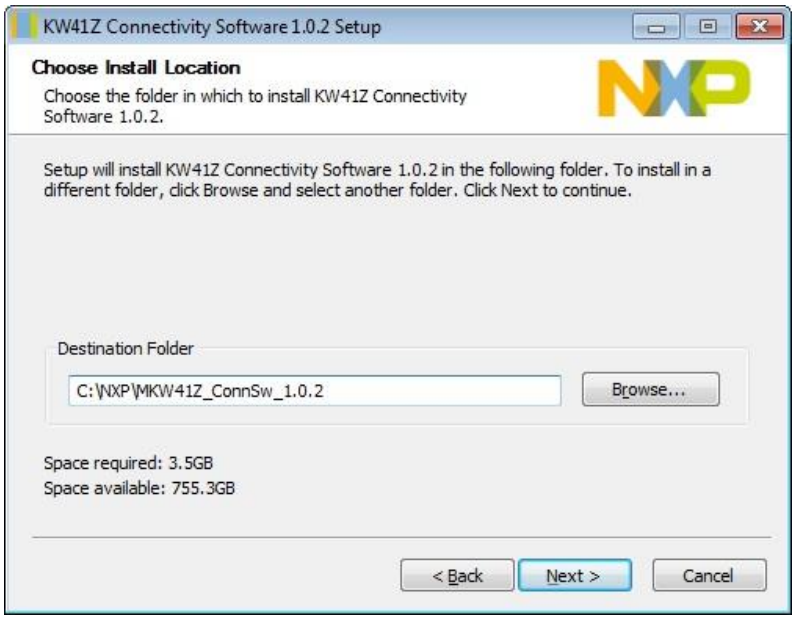

Figure 3: BLE software installation wizard location selection

In the next step you have to check the components you want to install and uncheck the components you don't want to install. In order to install BLE Software be sure that the corresponding checkbox is enabled as shown in the next figure.

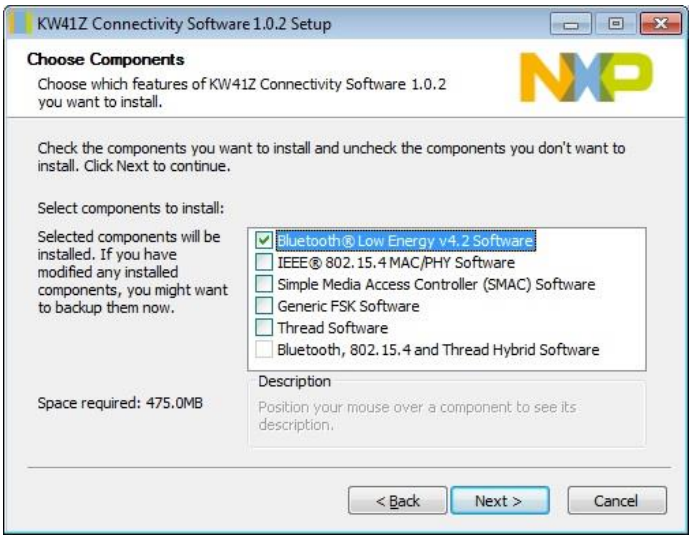

Figure 4: BLE Software checkbox enable

The last step is to choose whether you wish to create shortcuts for the BLE software installation.

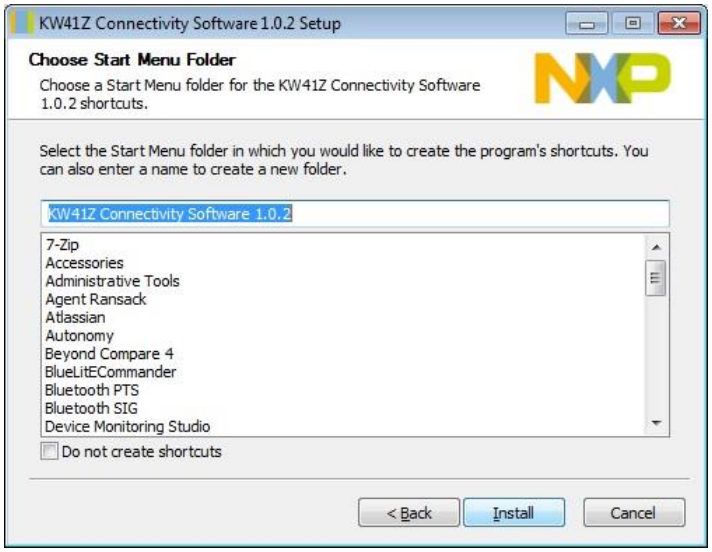

Figure 5: BLE software install wizard creates shortcuts

NOTE

The MKW41Z software installer will also contain the KSDK version 2.0 drivers and RTOS enablement for KW41Z/KW31Z/KW21Z. The BLE software does not need, use or affect any previous KSDK installed on your machine.

2 Cloning a project

Navigate to the KW41Z Connectivity Software installation folder and run the Project Cloner application (**MKW41Z_ConnSw_1.0.2\tools\wireless\ProjectCloner\ProjectCloner.exe**).

At the first run, the Project Cloner will search for the location of the KW41Z Connectivity Software installation folder. This location can be modified at any time.

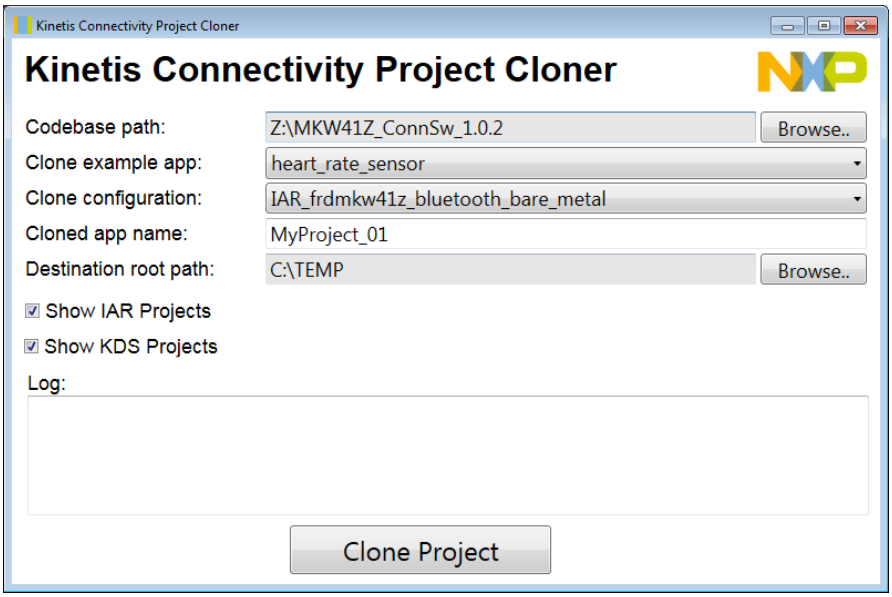

Figure 6: Connectivity Software example application

After the **Codebase path** has been selected correctly, the Project Cloner will display all the Connectivity Software example applications.

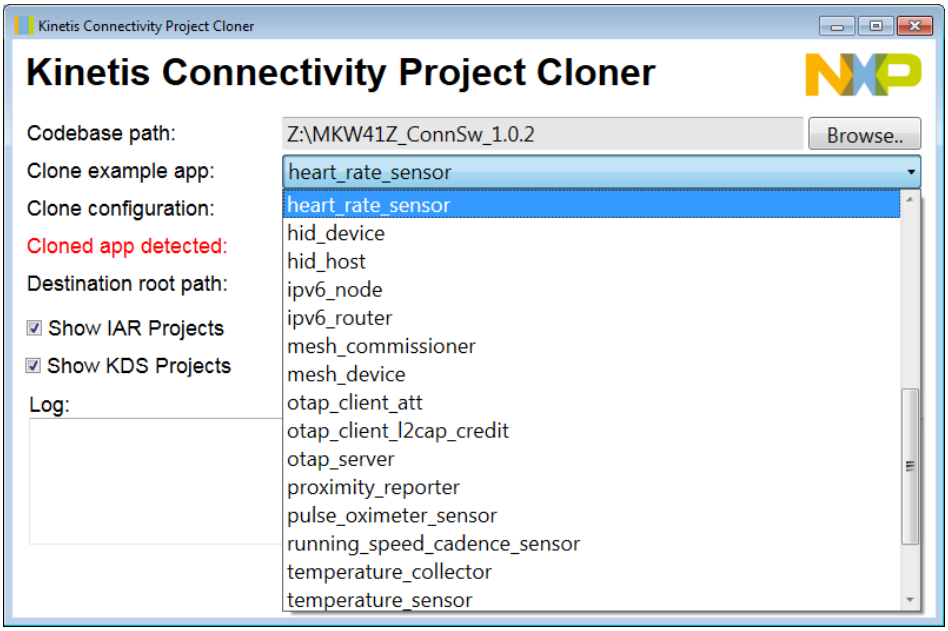

Figure 7: Connectivity Software example applications

Next select the example application to be cloned (**Clone example app**), and the desired configuration (**Clone Configuration**).

By default both IAR and KDS projects are shown in the **Clone configuration** dropdown menu. This can be modified by deselecting one of the two corresponding checkboxes.

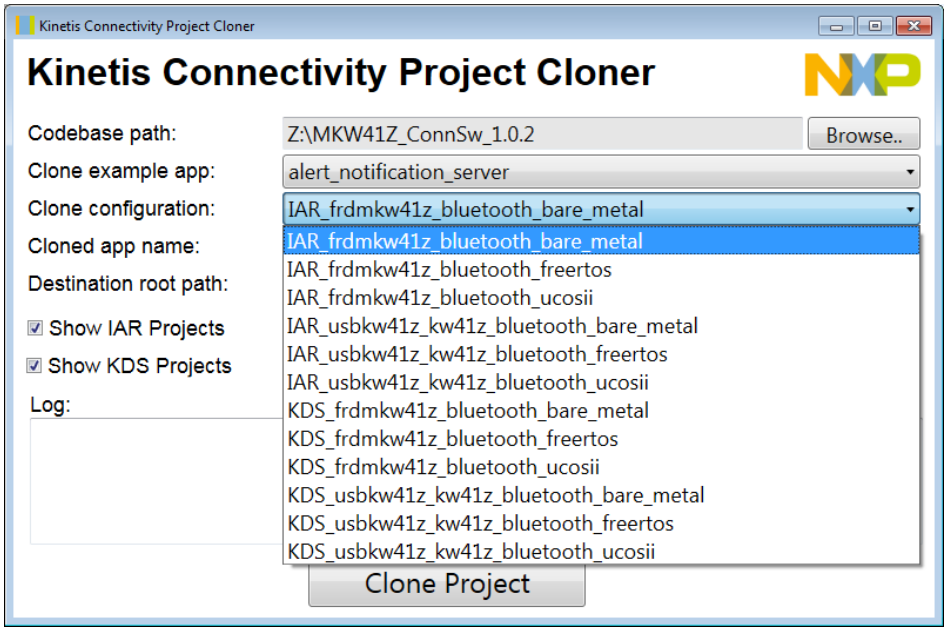

Figure 8: Available configurations for the selected example application

After this, the **Cloned app name** text box will contain a default name for the selected application. This name can be modified to any value.

The default **Destination root path** for the cloned application is the “C:/Temp” folder. To change this path, click the **Browse** button to select a new location.

Now press the **Clone Project** button to start the cloning process. The log window will display “Cloning completed” when the process ends.

The Project Cloner will clone all files referenced by the demo application project. The cloned application can be relocated anywhere since it has no dependencies of the original codebase folder.

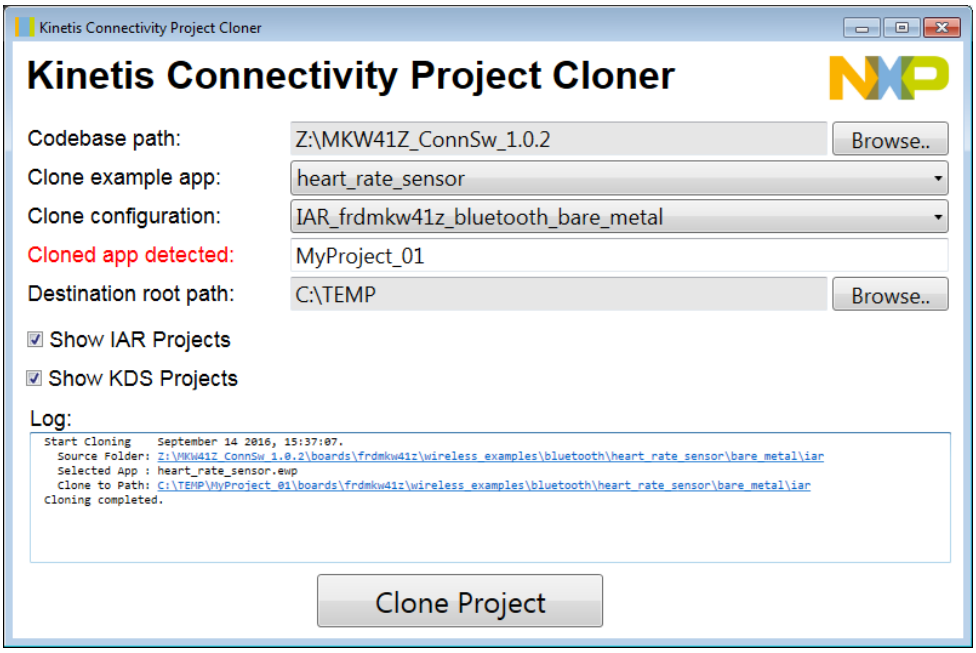

Figure 9: Application cloning done

3 Building the Binaries

This section details the required steps for obtaining the binary files for usage with the boards.

NOTE

In order to be able to build any of these packages you need a copy of the IAR Embedded Workbench for ARM® version 7.70.1 or higher or Kinetis Design Studio Integrated Development Environment version 3.2.0 or higher. This connectivity software package does not include support for any other toolchains.

The packages must be built with the debug configuration in order 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. All applications can be found using the following placeholders for text:

- <connectivity_path> : represents the root path for the BLE software package
- <board> : represents the target board for the demo app, can be either “frdmkw41z” or “usbkw41z”
- <RTOS> : represents the scheduler or RTOS used by the app and can be “bare_metal”, “FreeRTOS” or “uCOSII”
- <demo_app> : represents the demo app name
- <IDE> : represents the integrated development environment used to build projects and can be “iar” or “kds”

The demo applications general folder structure is the following:

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

NOTE

If your FRDM-KW41Z board is configured for the buck or boost modes of the DCDC converter inside the KW41Z microcontroller, 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.

Selected app: Heart Rate Sensor

Board: frdmkw41z

RTOS: bare-metal scheduler

Resulting location in the connectivity software installation directory:
<connectivity_path>boards\frdmkw41z\wireless_examples\bluetooth\heart_rate_sensor\bare_metal\<IDE>

Cloned application location:

Destination_Root_Path\MyProject_01\boards\frdmkw41z\wireless_examples\bluetooth\heart_rate_sensor\bare_metal\<IDE>

NOTE

The Heart Rate sensor application has the low power mode enabled 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 *gPWR_UsePowerDownMode* define need to be set to 0 in the app_preinclude.h header file.

3.1 Building and Flashing the BLE Software Demo Applications using KDS

Step 1:

Open the KDS IDE and create a new workspace.

Step 2:

Import the project into Workspace: File -> Import -> General -> Existing Projects into Workspace.

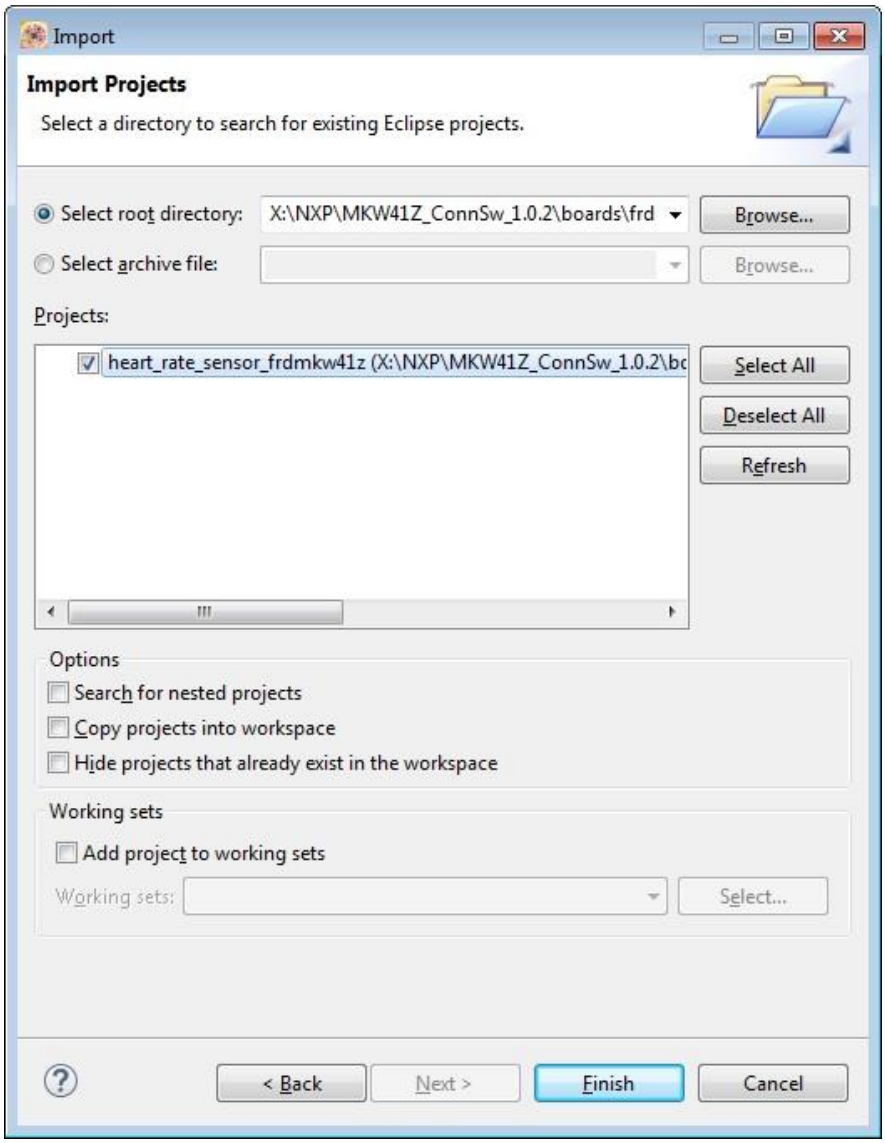

Figure 10: "Heart Rate Sensor" import project

Step 3:

Select the Heart Rate Sensor project.

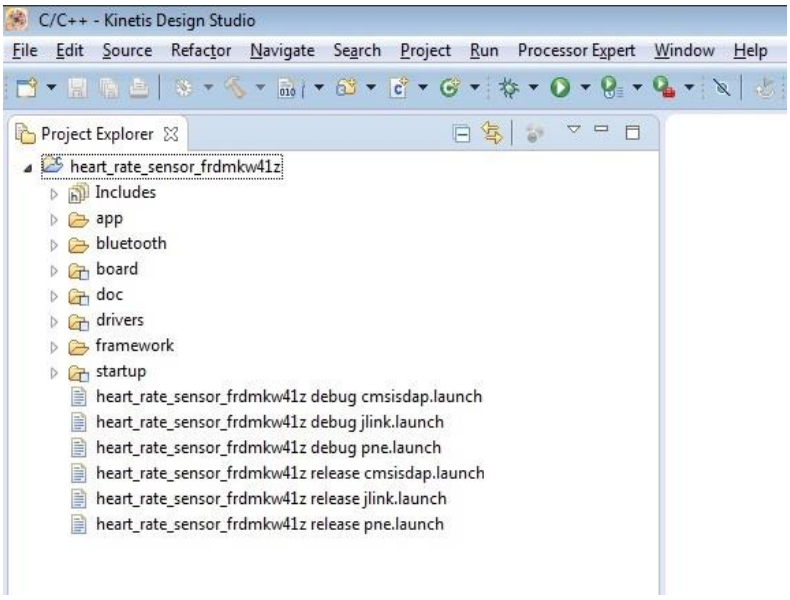

Figure 11: "Heart Rate Sensor" Bare Metal KDS project

Step 4:

Build the Heart Rate Sensor project.

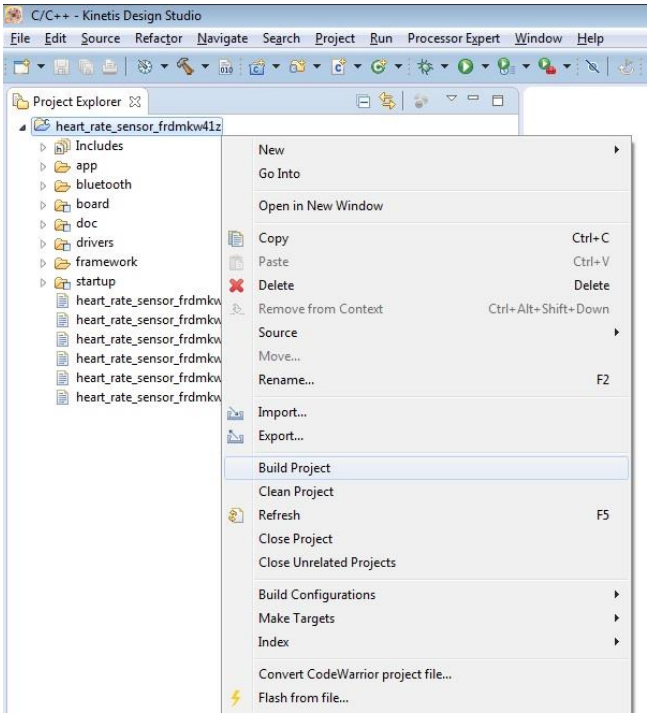

Figure 12: "Heart Rate Sensor" Bare Metal build

Step 5:

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

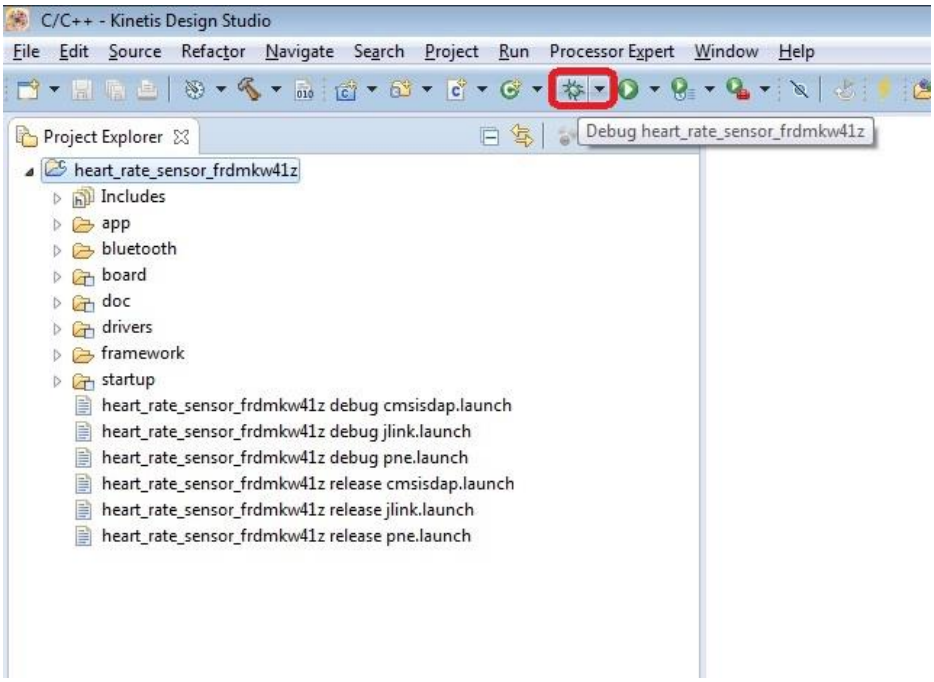

Figure 13: “Heart Rate Sensor” Debug

NOTE

Please make sure that you install the latest J-Link driver and associate it with KDS. To do this, download the driver from https://www.segger.com/jlink-software.html and install it. When you are asked to associate the applications which use J-Link driver with this version of driver you only have to check the box near Kinetis Design Studio and click Ok.

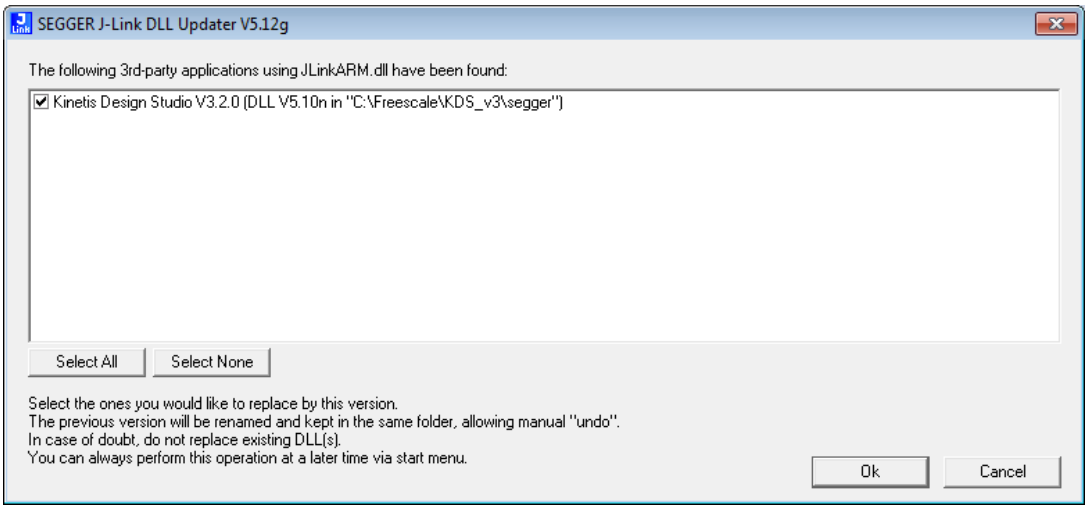

Figure 14: Update KDS J-Link Driver

Step 6:

Select the J-Link debug configuration option when asked for Launch Configuration.

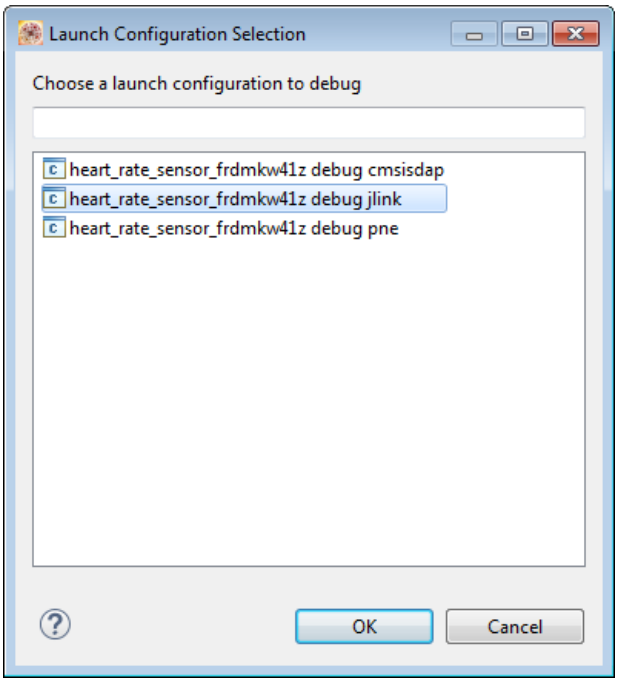

Figure 15: "Heart Rate Sensor" debug configuration

NOTE

The projects are configured to use "J-Link / J-Trace" as the default debugger. Please make sure that your board's OpenSDA chip contains a J-Link firmware or that the debugger selection corresponds to the physical interface used to interface to the board. See the section 3.3 for more information.

3.2 Building and Flashing the BLE Software Demo Applications using IAR

Step 1:

Navigate to the resulting location in either the connectivity software installation directory or the cloned application root directory.

Step 2:

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

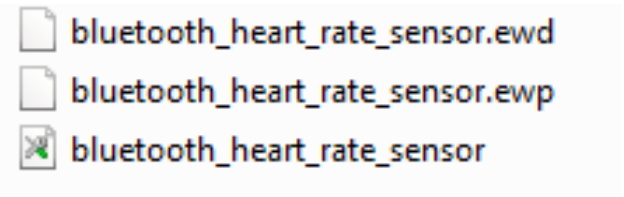

Figure 16: "Heart Rate Sensor" IAR demo project location

Step 3:

Select the Heart Rate Sensor project.

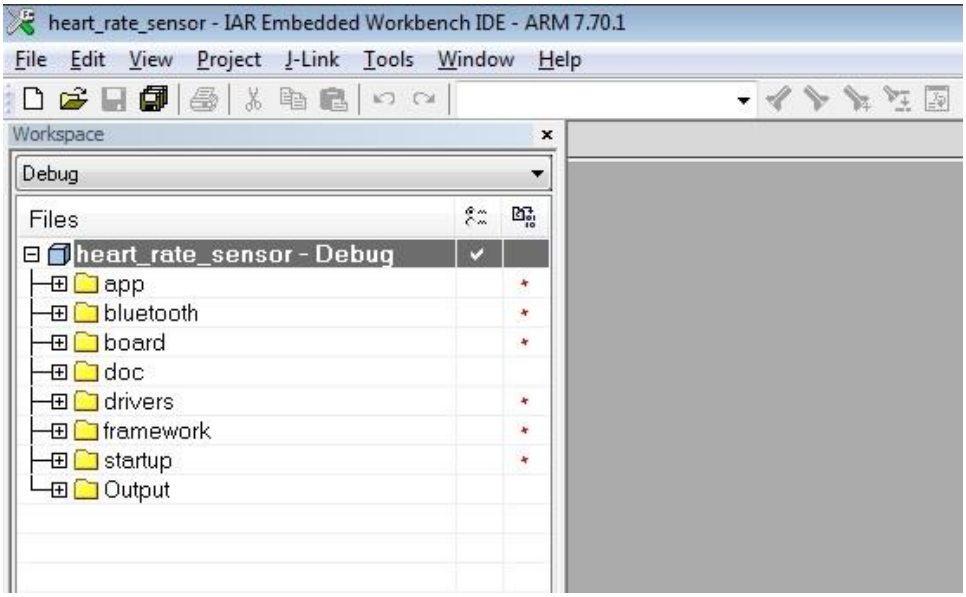

Figure 17: Heart Rate Sensor Bare Metal IAR project

Step 4:

Build the Heart Rate Sensor project.

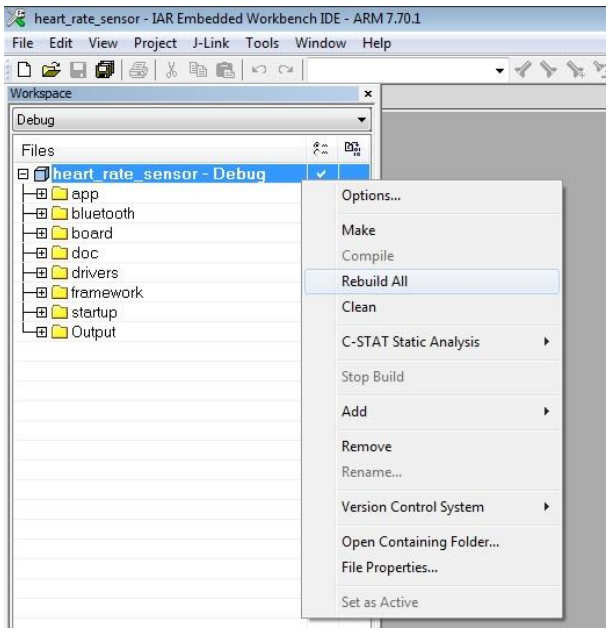

Figure 18: Heart Rate Sensor Bare Metal build

Step 5

Make the appropriate debugger settings in the project options window:

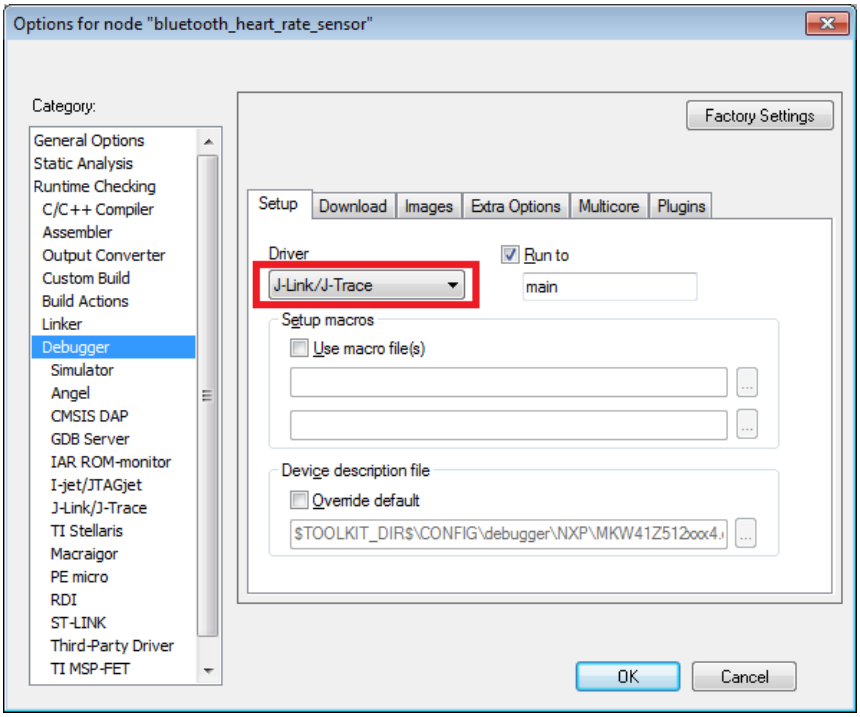

Figure 19: Debugger Settings

Step 6:

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

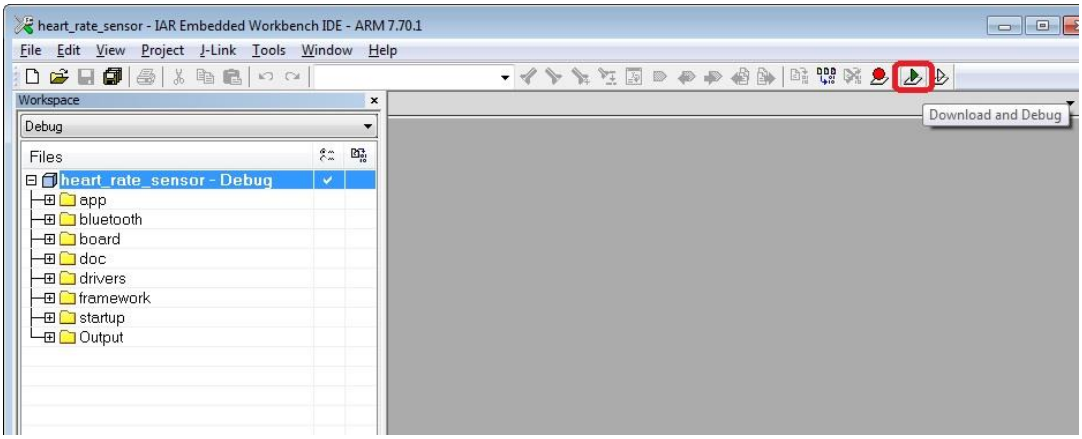

Figure 20: Heart Rate Sensor Download and Debug

NOTE

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

3.3 Flashing a Binary Image File Without Using an IDE

The MKW41Z connectivity software package contains in the MKW41Z_ConnSw_1.0.2\tools\wireless\binaries folder a series of pre-compiled binary applications that can be flashed onto a development board.

In order to flash the corresponding binaries to the FRDMKW41Z board, the best approach is to use the OpenSDA on-board interface J-Link Mass Storage Device functionality, by simply dragging and dropping the binary image in the mass storage drive exposed by this OpenSDA firmware.

In order to flash the firmware on the USBKW41Z, a J-Link probe is needed along with the latest J-Link software from www.segger.com.

Run the *jlink.exe* executable provided in the J-Link software installation follow the steps below for flashing the image on the microcontroller. Make sure that the binary file is in the same folder with the *jlink.exe* executable, or specify the absolute path to the file.

Step 1: Select MKW41Z512xxx4 device.

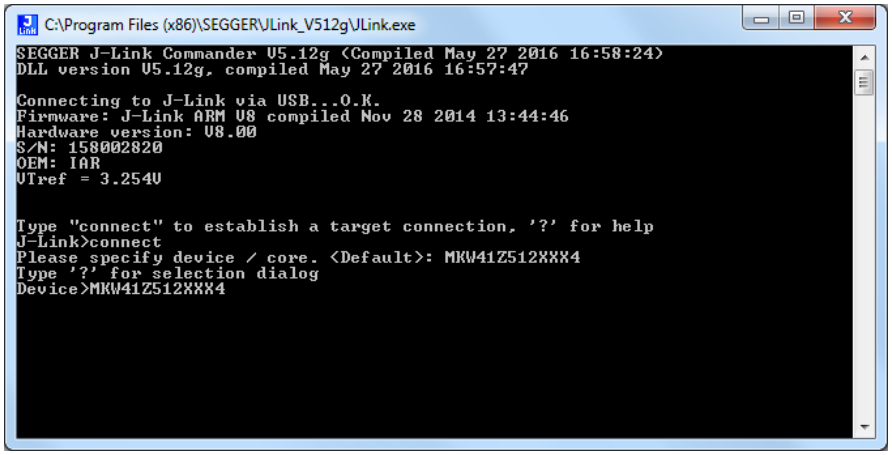

Figure 21: MKW41Z512xxx4 device selection

Step 2: Select SWD target interface.

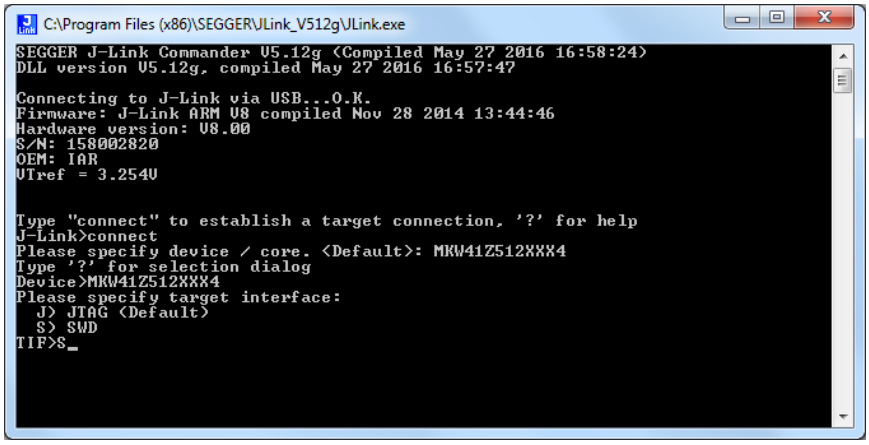

Figure 22: SWD interface selection

Step 3: Press “Enter” to select the default interface speed.

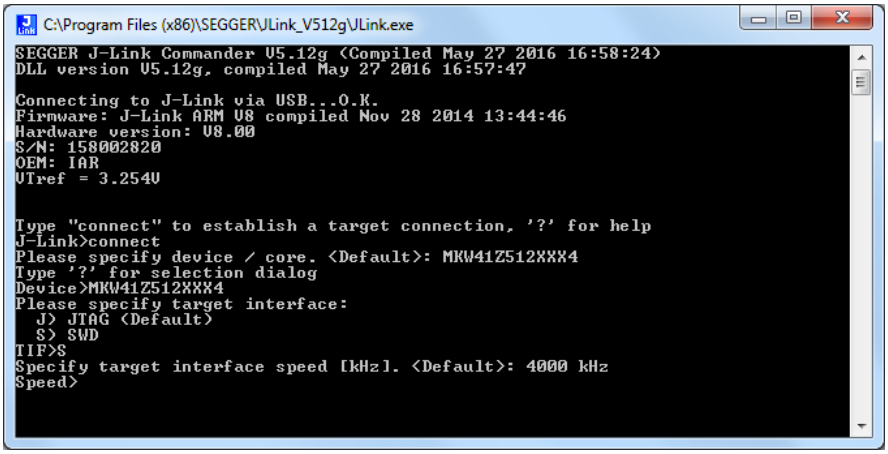

Figure 23: SWD interface speed selection

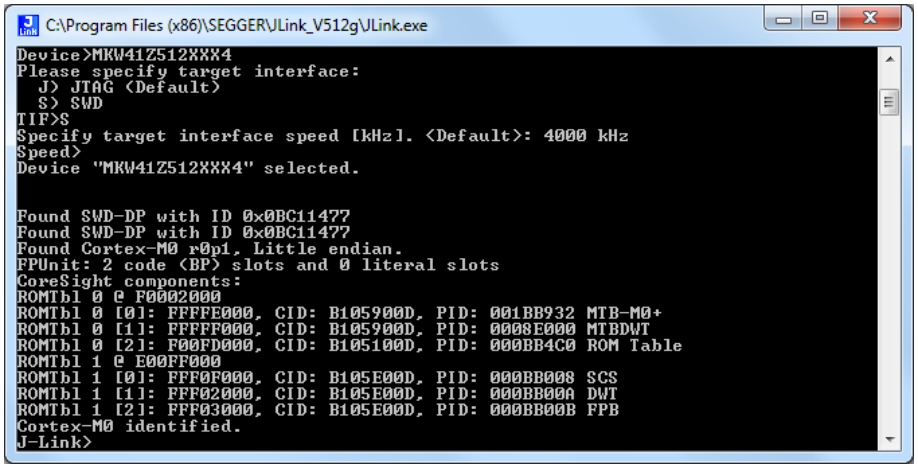

Figure 24: Cortex-M0 identified

Step 4: Type **loadbin app.bin 0** in order to flash the binary file (assuming application name is *app.bin*).

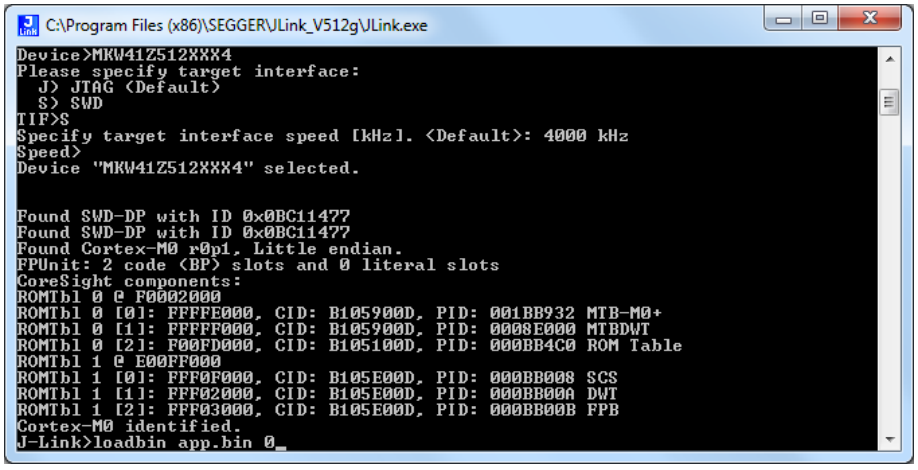

Figure 25: Load binary file

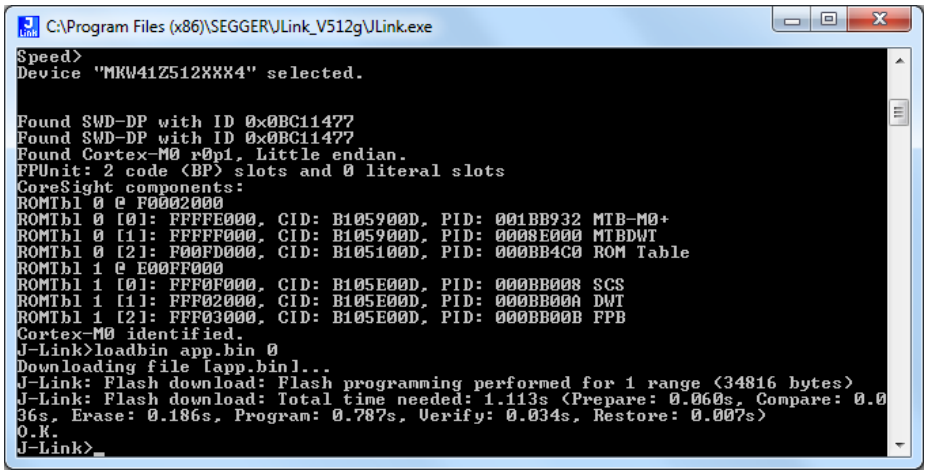

Figure 26: Download completed successfully

4 Hardware Setup

The hardware setup in this example uses either a FRDMKW41Z or USBKW41Z development platform, shown in the figure below:

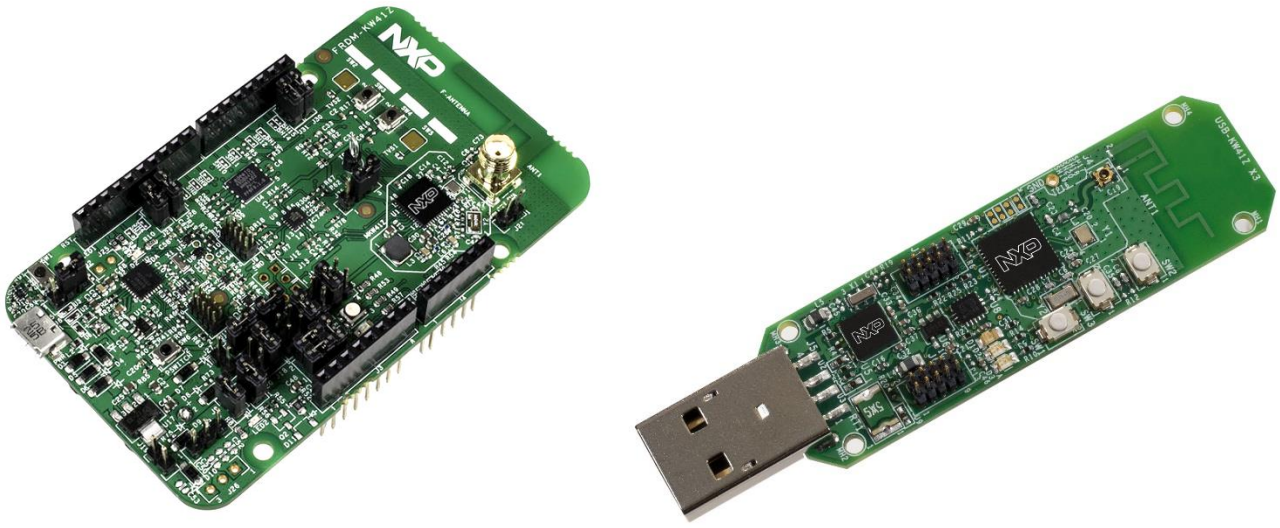

Figure 27: FRDMKW41Z and USBKW41Z

The FRDMKW41Z and USBKW41Z boards should have their OpenSDA USB ports connected to a Windows PC. The OpenSDA chip on the boards 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.

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

https://github.com/mbedmicro/CMSIS-DAP
https://www.segger.com/opensda.html
http://www.pemicro.com/opensda/

J-LINK/J-TRACE is the default interface selected in the IAR Embedded Workbench for ARM® projects with FRDMKW41Z and USBKW41Z included in this release.

The FRDM-KW41Z board can be configured via jumpers to be in the two modes of the DCDC converter inside the KW41Z microcontroller or to bypass it entirely, as shown in the figure below:

Power Configuration

	PWR_CFG J18	PSW_CFG J16	DCDC_CFG J17	REG_CFG J22
Bypass Mode (auto start) VDCDC_IN (1.71 to 3.6V) Operation 1.8V - 3.6 V	1-2	1-2	3-4	1-3 2-4 5-6
Boost Mode (auto start) VDCDC_IN (0.9V to 1.8V) Single Battery Operation	2-4	3-4	1-2 5-6	5-6
Buck Mode (manual start) VDCDC_IN (1.8V to 4.2V) Coin Cell Battery Operation	2-4	5-6 press SW6 to start	3-4	5-6
Buck Mode (auto start) VDCDC_IN (1.8V to 4.2V) Coin Cell Battery Operation	2-4	3-4	3-4	5-6

Figure 28: FRDM-KW41Z Jumper Configuration for DCDC Modes

How to Reach Us:

Home Page:
www.nxp.com

Web Support:
www.nxp.com/support

.

Document number: MKW41ZBLESW122QSG
Rev. 0
09/2016