NXP Semiconductors
User's Guide

Document Number: MKW41ZZIGBEESW606QSG
Rev. 1,
11/2018

Kinetis MKW41Z Zigbee 3.0 Software

Quick Start Guide

Introduction

This document is a brief presentation of the NXP Zigbee 3.0 Software for the MKW41Z wireless microcontroller platforms, version 6.0.8. This software package is built using the Kinetis Software Development Kit (KSDK) version 2.2. This document covers hardware setup, build and usage of the provided demo applications.

Contents

1 Download

Navigate to https://mcuxpresso.nxp.com/en/welcome and select the development board:

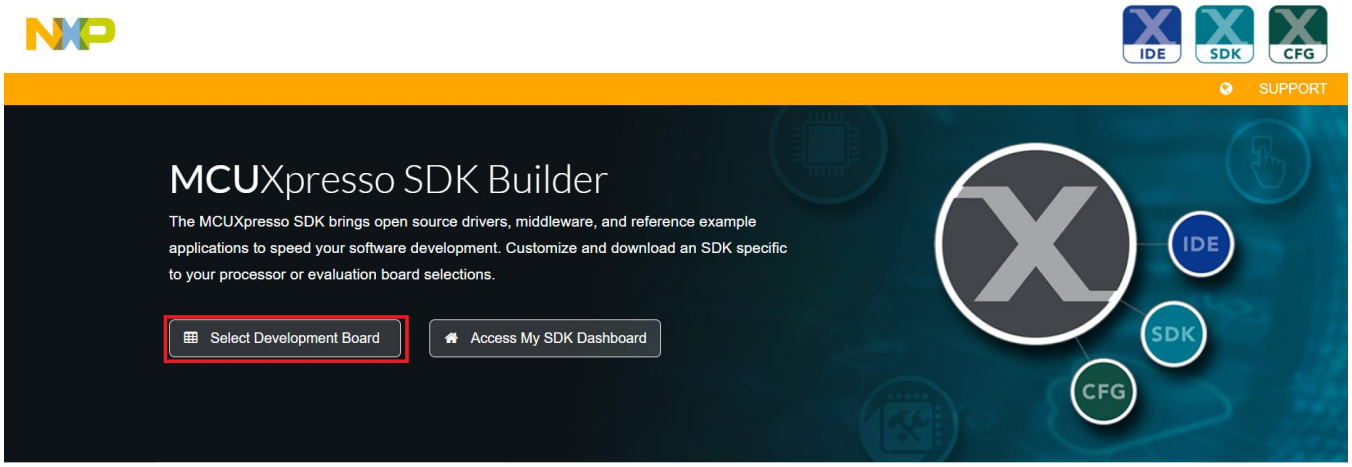

Figure 1: The MCUXpresso welcome screen

Type the desired platform or select from the list, then press “Build MCUXpresso SDK”:

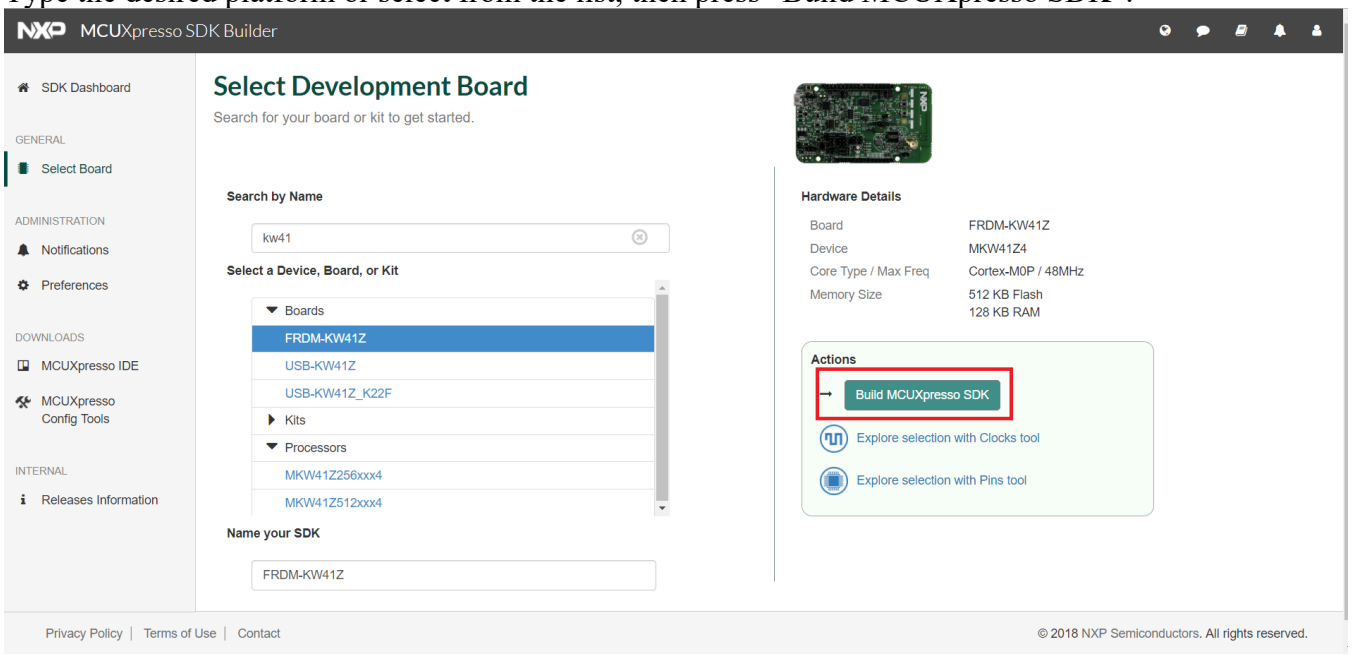

Figure 2: Available configurations

Select the desired toolchain(s) and middleware (by pressing “Add software components”):

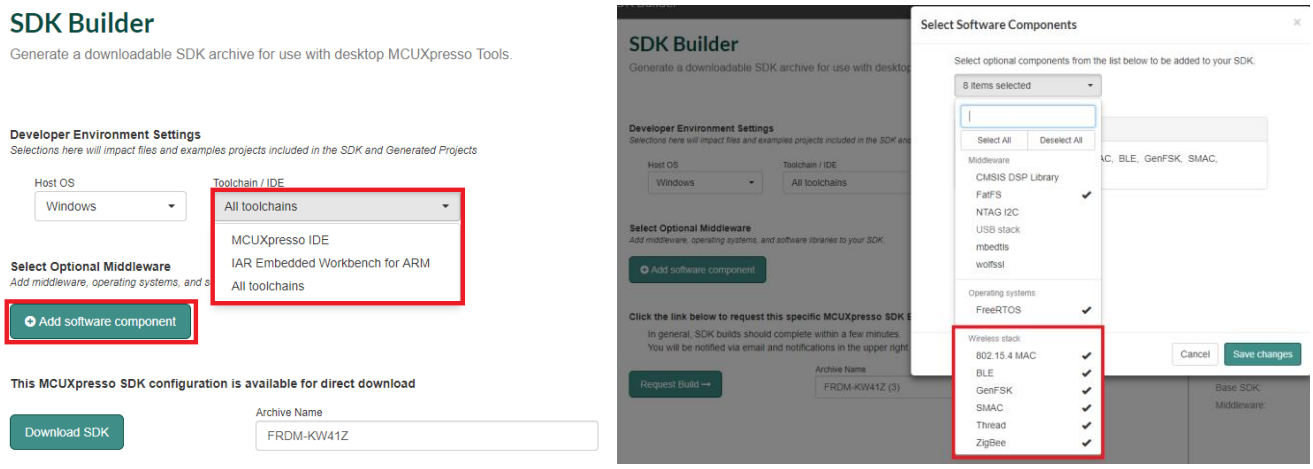

Figure 3: Select toolchain(s) and middleware

If the package is already built, the “Download SDK” button will appear, else the “Request Build” button will appear.
If you request a new build, you will receive a notification when process ends. To download the archive, access the SDK Dashboard:

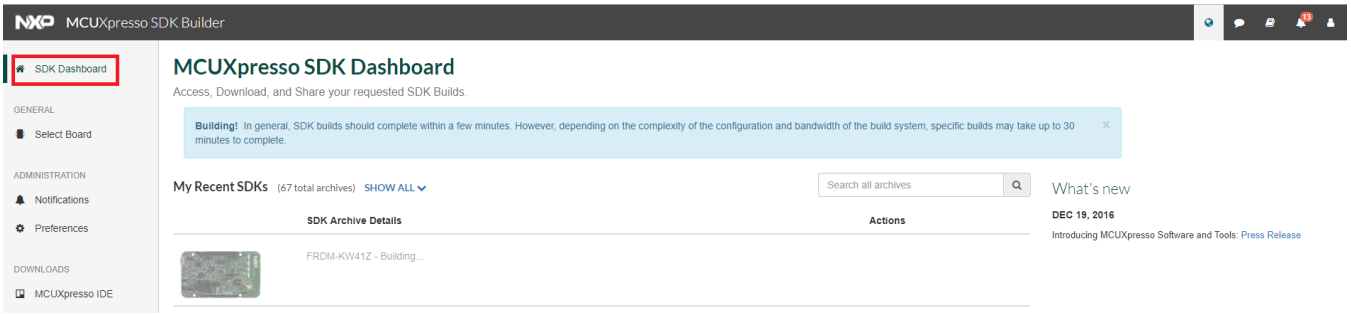

Figure 4: MCUXpresso SDK Dashboard

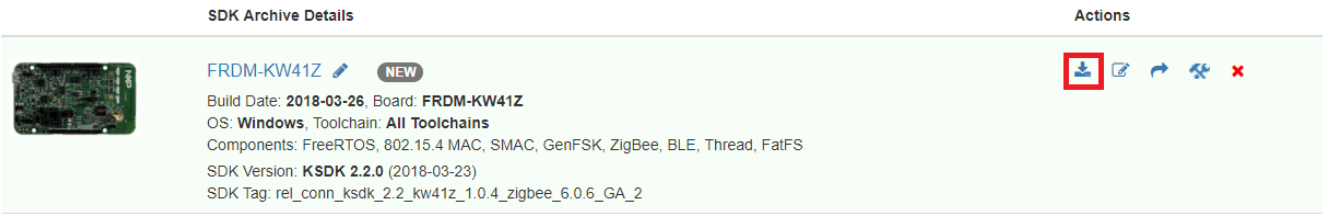

Figure 5: Download the archive

2 Building the Binaries

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

NOTE

To be able to build any of these packages you need IAR Embedded Workbench for ARM® version 8.32.1 or higher or MCUXpresso IDE version 10.2.1 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 *ble_zigbee_coordinator* 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 “frdmkw41z”
- <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\zigbee_3_0\<demo_app>\<RTOS>\<IDE>\

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

Kinetis Zigbee 3.0 Software Demo Application Build Example

Selected app: ble_zigbee_coordinator

Board: frdmkw41z

RTOS: FreeRTOS

Resulting location:

<connectivity_path>\boards\frdmkw41z\wireless_examples\hybrid\
ble_zigbee_coordinator\freertos\ <IDE>

NOTE

If your FRDM-KW41Z board is configured for the buck or boost modes of the KW41Z 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 config.h header file.

2.1 Building and Flashing the Zigbee 3.0 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):

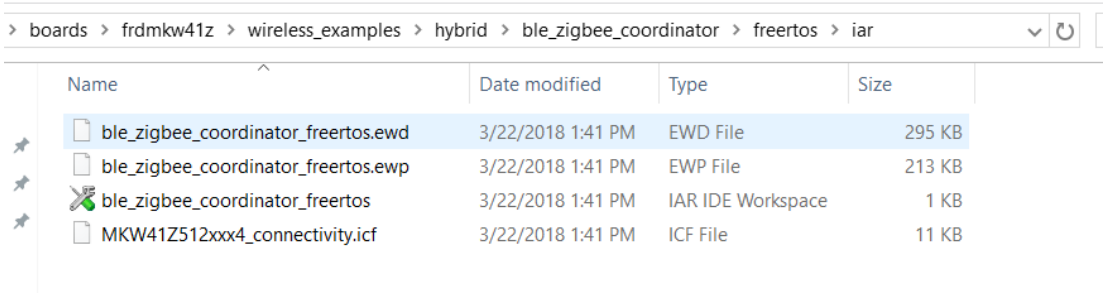

Figure 6: ble_zigbee_coordinator IAR demo project location

Step 3:

Select the ble_zigbee_coordinator project.

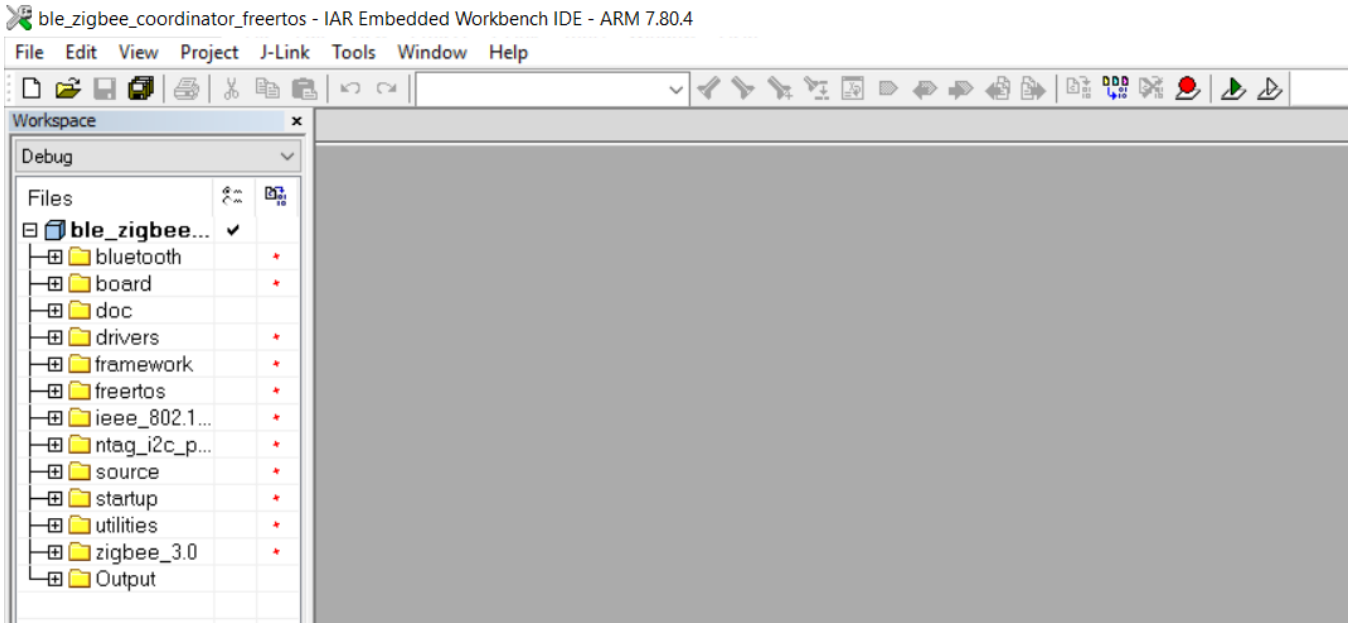

Figure 7: ble_zigbee_coordinator FreeRTOS IAR project

Step 4:

Build the ble_zigbee_coordinator project.

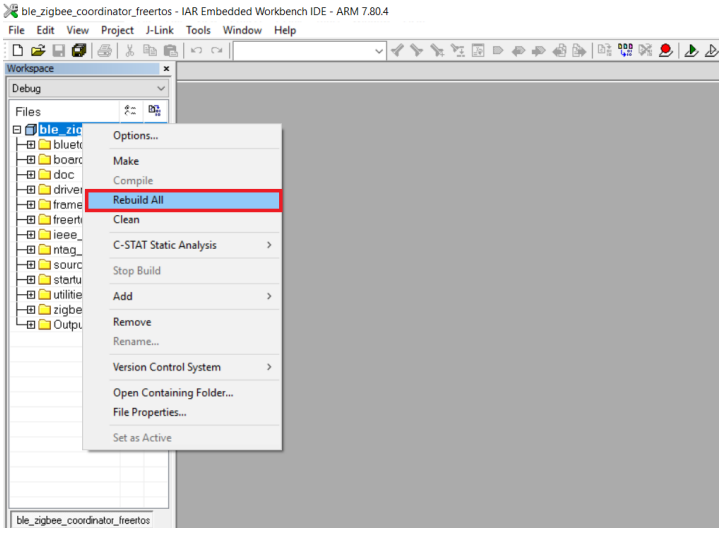

Figure 8: ble_zigbee_coordinator build

Step 5

Make the appropriate debugger settings in the project options window:

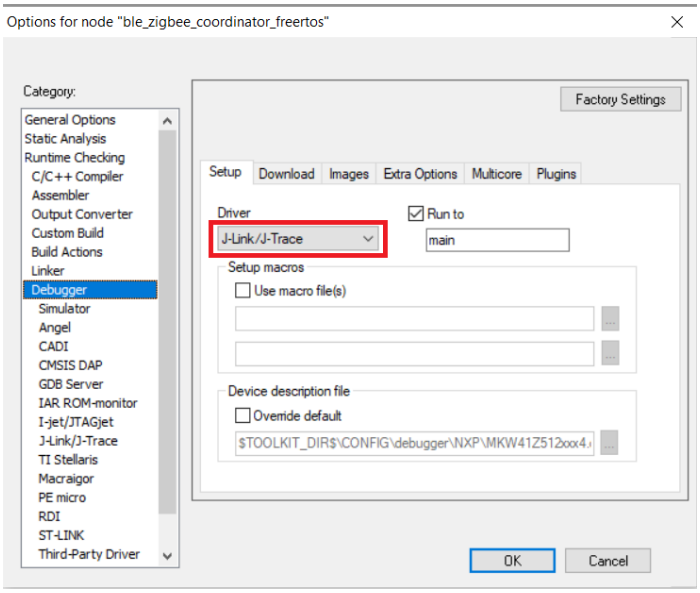

Figure 9: Debugger Settings

Step 6:

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

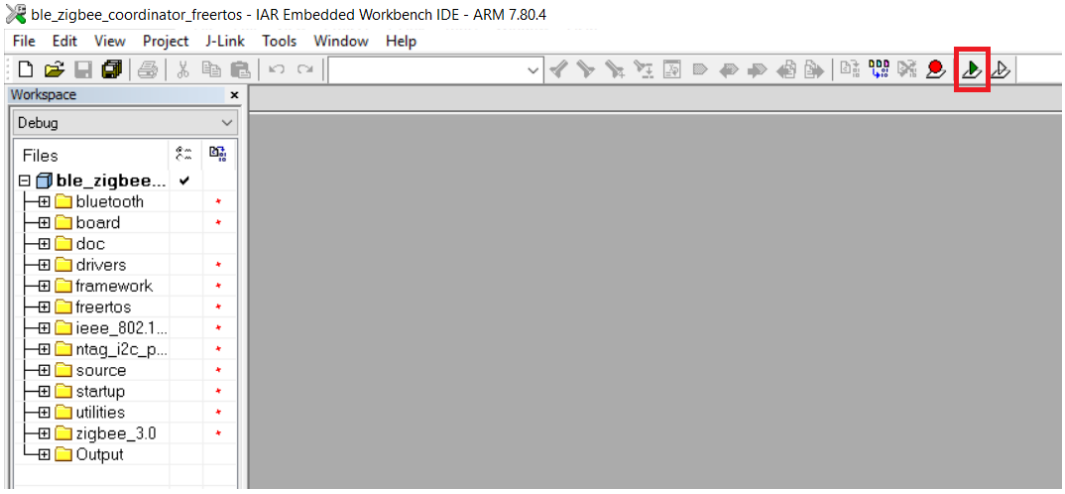

Figure 10: ble_zigbee_coordinator Download and Debug

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 Segger 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 Zigbee 3.0 Software Demo Applications using MCUXpresso IDE

Step 1:

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

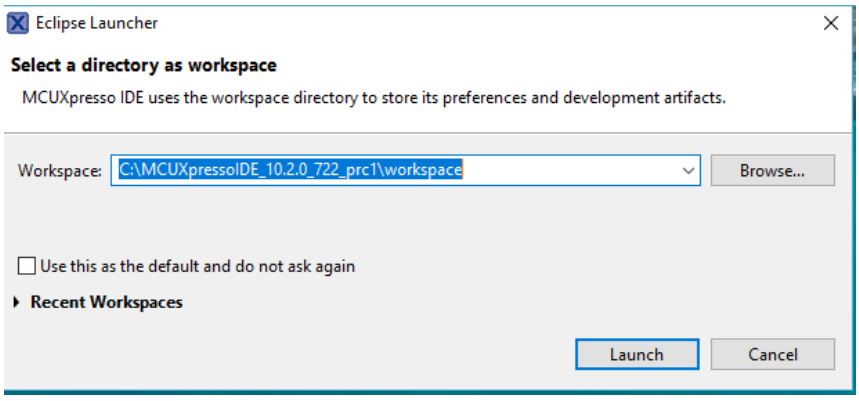

Figure 11: Select a MCUXpresso IDE workspace location

Step 2:

Drag and drop the package archive into the MCUXpresso Installed SDKs area in the lower right of the main window.

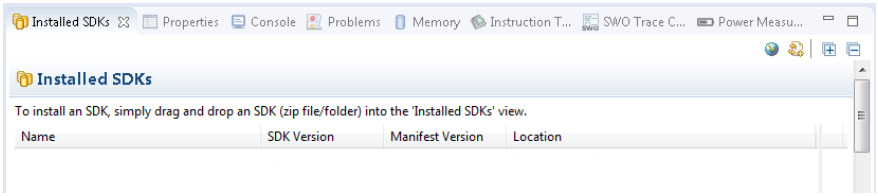

Figure 12: MCUXpresso IDE Installed SDKs area

Step 3:

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

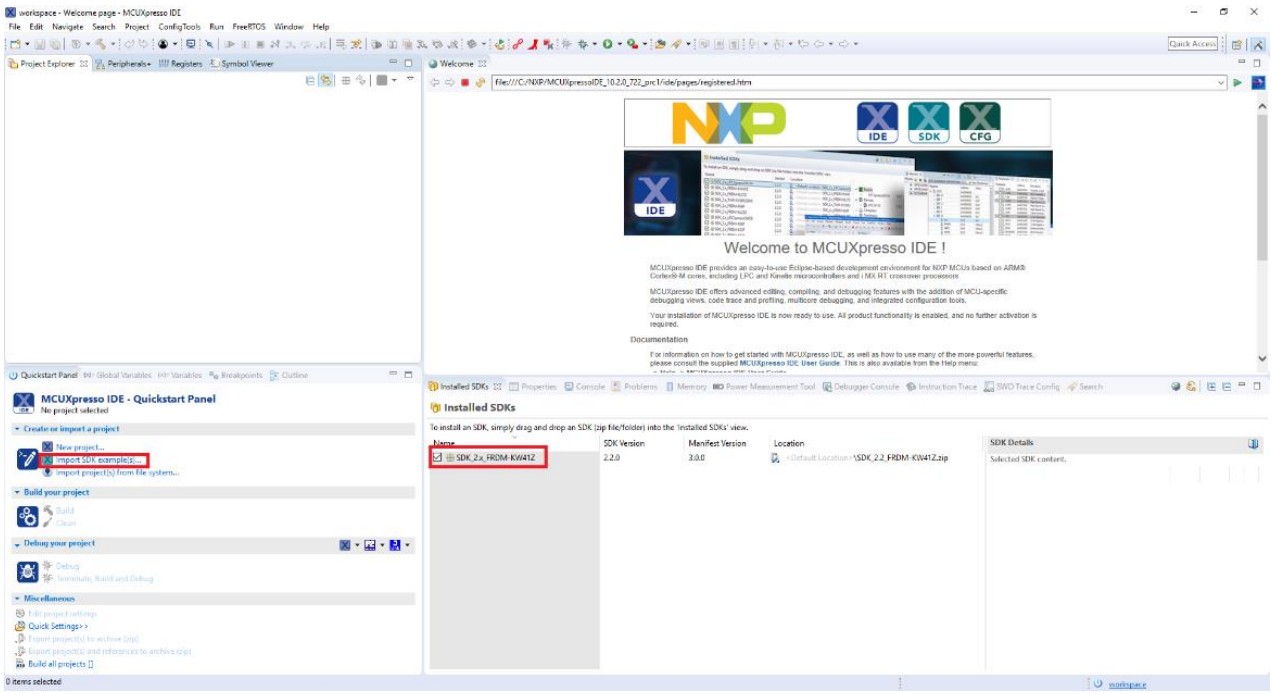

Figure 13: Installed SDKs

Step 4:

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

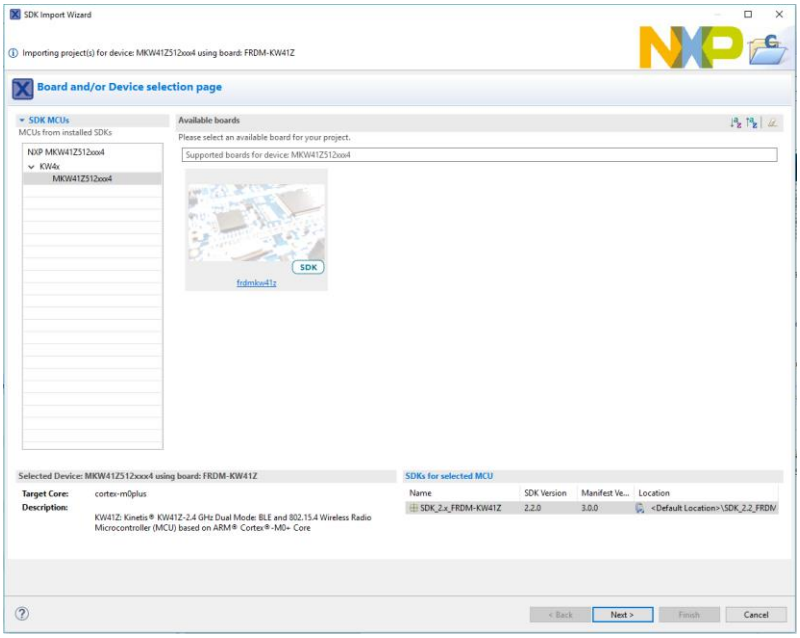

Figure 14: Select the board

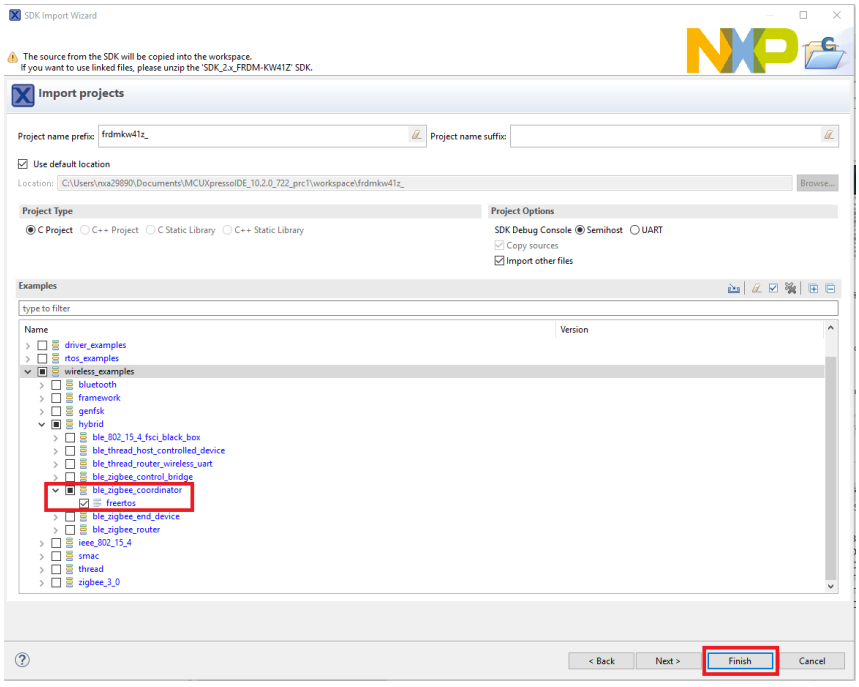

Figure 15: Select the example(s)

Step 5:

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

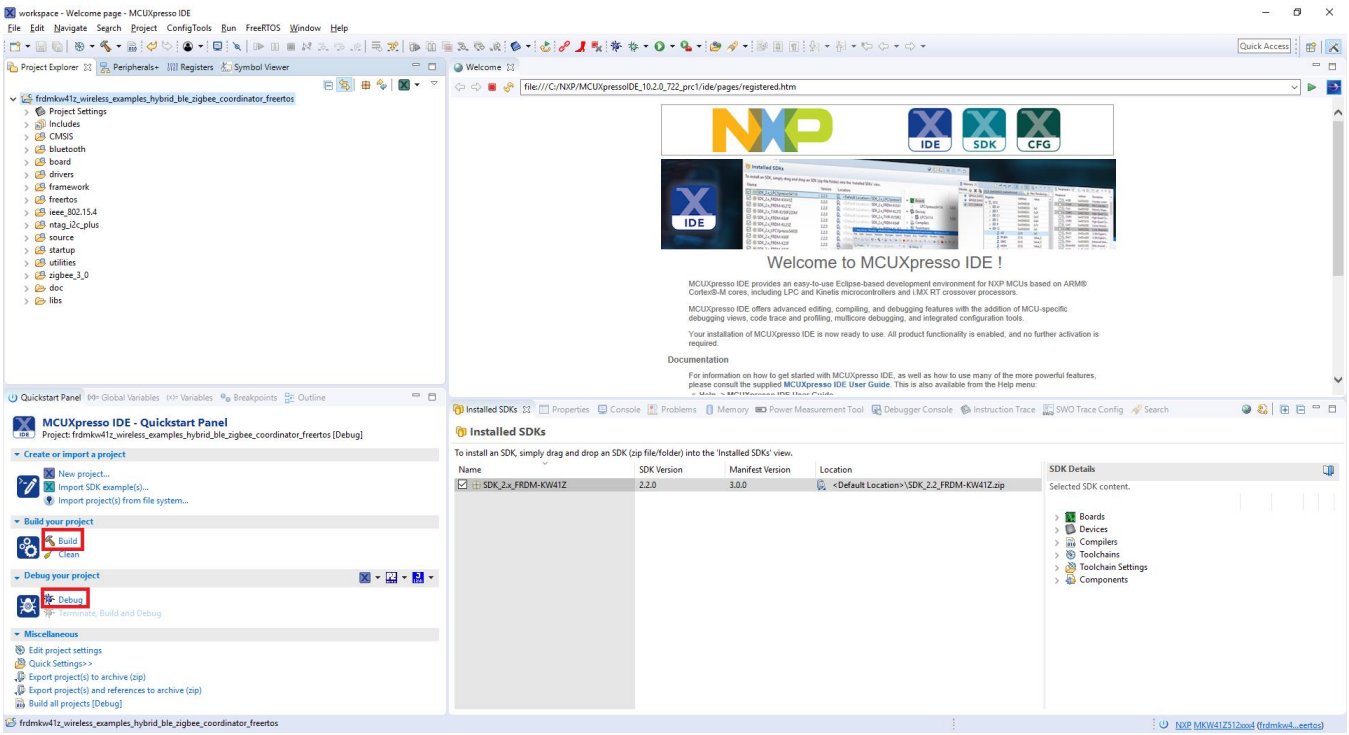

Figure 16: Build, download and debug the ble_zigbee_coordinator FreeRTOS project

3 Hardware Setup

The hardware setup in this example uses a FRDM-KW41Z development platform, shown in the figure below:

Figure 17: FRDM-KW41Z

The FRDM-KW41Z 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.

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

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

J-Link/J-Trace is the default interface selected in the IAR Embedded Workbench for ARM® projects with FRDM-KW41Z 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 18: FRDM-KW41Z Jumper Configuration for DCDC Modes

4 Running the BLE Zigbee Coordinator 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:

Build the .srec or .bin file for the ble_zigbee_coordinator embedded project corresponding to the hardware board, 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 Zigbee Shell icon.

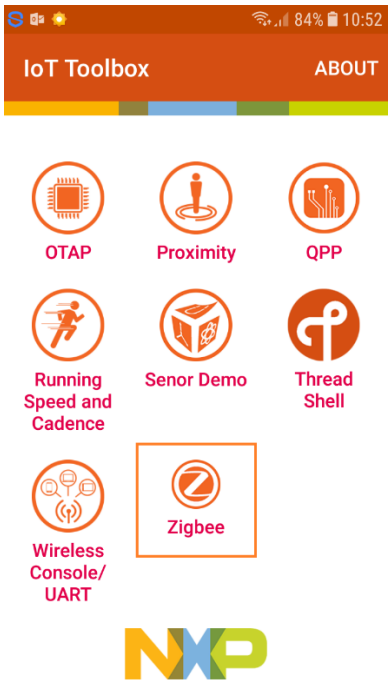

Figure 19: IoT Toolbox - opening the Zigbee Shell application

Step 4:

Connect to the board and execute Zigbee shell commands.

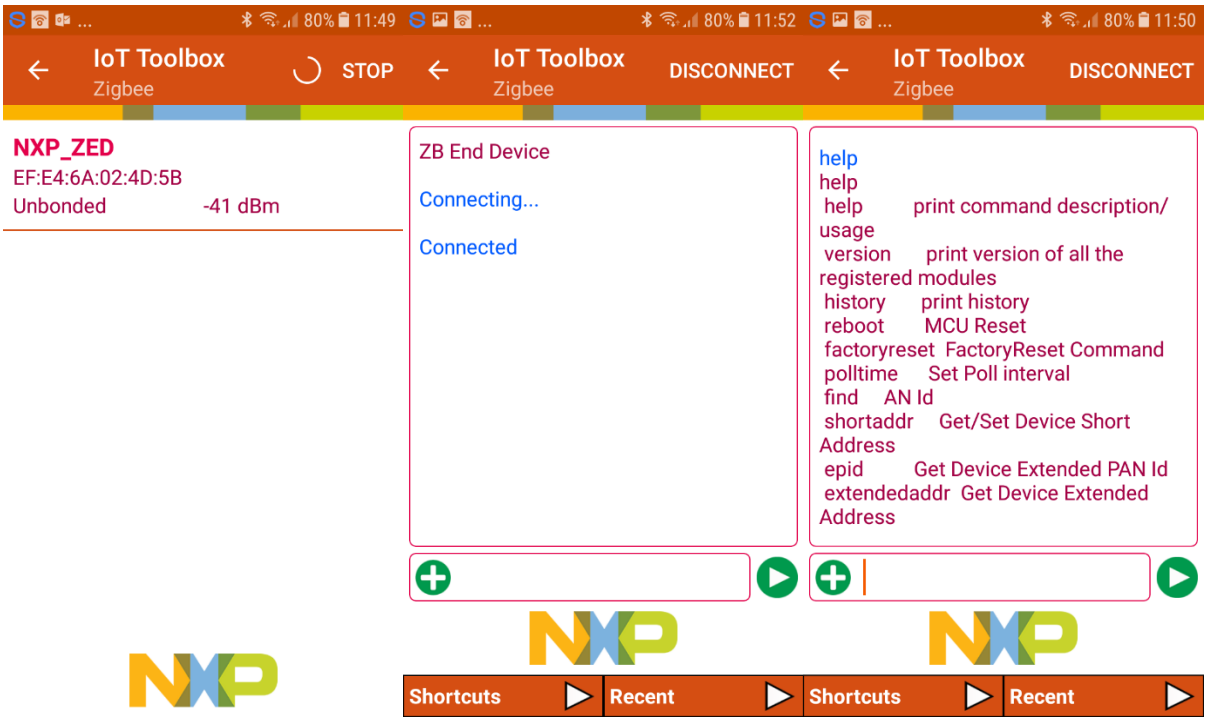

Figure 20: Selecting the scanned device and executing Zigbee shell commands

www.nxp.com/support

Document Number: MKW41ZZIGBEESW606QSG
Rev. 1
11/2018