NXP Semiconductors
User's Guide

Document Number: KTSDAUG
Rev. 9, 08 April 2019

Kinetis Thread Stack Demo Applications

User's Guide

Contents

Chapter 1
Introduction

This document is an overview of the deployment and operation of wireless network applications created using the Thread network protocol stack running on Kinetis KW41Z.

This document is also a user's guide for the sample applications included with the Kinetis Thread Stack software builds.

1.1 Audience

This document is for firmware and system developers who create Thread-enabled products. The document also provides high level descriptions of Thread application scenarios which can be deployed on Kinetis development boards.

Chapter 2
Thread Stack and Technology Overview

Thread is an IPv6 wireless mesh networking protocol for integration into consumer products. At the link layer, Thread is based on IEEE® 802.15.4 MAC and PHY operating in the 2.4 GHz open radio frequency band.

Thread networks provide a mesh communication fabric for seamless user-to-device and device-to-device application interaction scenarios for the connected home. The technology also provides IP connectivity and addressability to products such as coin-cell battery operated sensors and simple appliances and actuators making them accessible within the mesh network, the home area IP infrastructure, and Internet.

Compared to other wireless protocols, Thread is focused on providing low latency, reliability, redundancy, security, simple and autonomous network configuration, and low-power consumption.

2.1 Thread Device Types

The image below shows the different categories of connected home devices and their respective power profiles.

Figure 1. Connected home devices

The Thread applications provided with the Kinetis Thread stack contain preconfigured stack and application layer configurations organized around the major categories of Thread device types and capabilities reflecting the power and complexity requirements shown in the image above.

Chapter 3
Kinetis Thread Stack Applications Overview

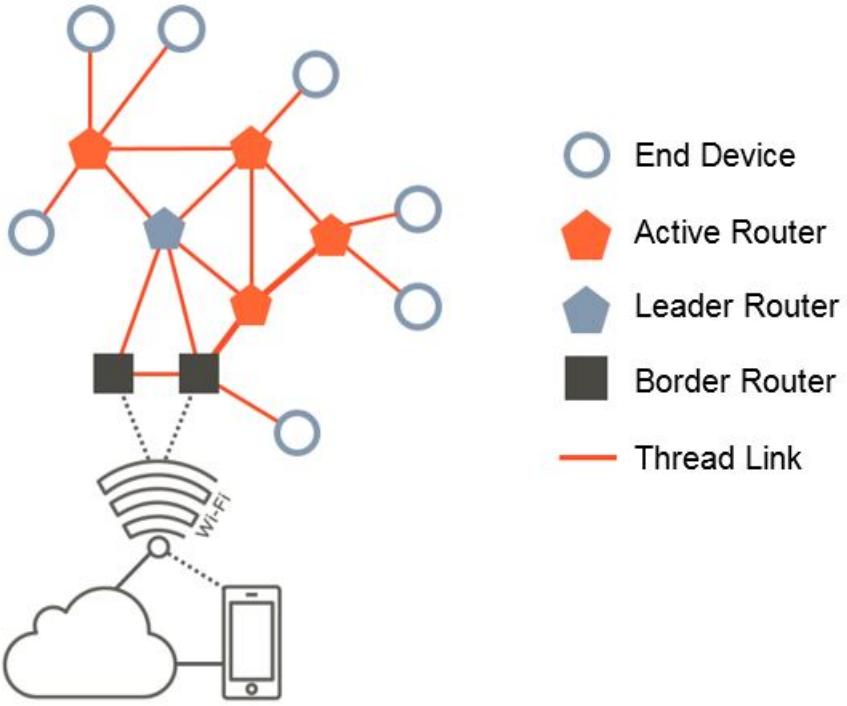


Figure 2. Typical Thread network devices and roles

3.1 Thread router eligible devices

A Router Eligible Device is a node which initially joins the network as an End Device, but can adaptively become a mesh Router. Such a device may also have capabilities to initialize, create, and bootstrap a new Thread Network for the user or a management entity.

The distinct roles this device can play during network operation are:

- Router Eligible End Device(REED)
- Active Router, also referred to as a Promoted Router, or a Router

The two roles are dynamic and can adaptively change from the End Device role when the node does not actively participate in routing, does not forward data transmissions, and cannot accept to be a “parent” for other End Devices to the Active Router role which actively performs all these functionalities.

Typically up to 16-32 Router Eligible Devices on a Thread Network can automatically activate the routing functionality and request a Router Identifier (Router ID). When a Router ID is assigned, the devices become Active Routers.

A Router Eligible Device can also assume a Leader role, acting as a point of decision for Router ID assignments and dissemination of routing and network information for a network partition. A Thread network starts as a single network partition. However, multiple distinct partitions can form if routing segments of the same network become disconnected. The partitions are still part of the same Thread network, and can share network credentials and merge back to a single partition if inter-connectivity between the routing segments is re-established. Each partition has a single Leader router. The Leader role is transitional and can be taken over by any other Active Router in the partition if the current Leader becomes unavailable.

The application example for a Router Eligible Device is a template for developing mains-powered and always-on Thread devices such as smart power outlets, appliances, network range extenders, and control panels.

Other mains-powered devices such as smart light bulbs or lamps are not restricted by power constraints and can act as routers. However, because users can disconnect power from light bulbs using room power switches, the network routing capacity can become diminished when onlylighting fixtures act as routers.

When deploying Thread devices and networks, a base routing infrastructure consisting of always-on Router Eligible Devices is recommended.

3.2 Thread end devices

An End Device is a node which does not have routing eligibility or network creation capabilities. The main characteristic of an End Device is that it always communicates to the rest of the network by having data relayed to and from an Active Router which becomes a "parent" for the End Device. The End Device at its turn acts as the "child" of the Active Router and expects the Active Router to act on its behalf for forwarding data transfers.

An End Device can be battery-powered by using a high-capacity battery. However, the end device is usually a mains powered device which is not expected to be always-on or which otherwise cannot fulfil the Router Eligible role described above due to its limited memory or processing resources available.

3.3 Thread low-power end devices

The Low-power End Device (also called Sleepy End Device (SED)) is an End Device which is meant to remain in a low-power, dormant state for the majority of its lifetime. It is usually battery-powered with a limited battery capacity.

The Low-power End Device in most use cases becomes active for communication only for very short periods, either “waking up” automatically based on a predefined periodic timer or by means of user interaction.

The device can become active, for example when a user presses a button and generates a wake up interrupt. As soon as they become active, Low-power End Devices can poll their parent Active Router for any data addressed to them and transmit any outgoing data they have to send. The devices should return as quickly as it is allowed by the application to the previous inactive radio state to preserve their battery.

Other devices communicating with a Low-power End Device must assume there may be some latency when sending data towards the end device. On the other hand, an interrupt-based wake up and associated transmissions which originate from the Low-power End Device can be very quick. These devices make good candidates for light switches, remote controls, or sensors.

3.4 Thread border routers

A Thread Network is a native IPv6 subnet, where each node has assigned one or more IPv6 addresses as end points and where IP protocols are used for all communications and network management.

Border Routers are Thread devices which have at least another IP interface on board besides the Thread radio transceiver. As a result, Border Routers can internally forward data at the IP layer from the Thread network to the other network interfaces over IP subnet boundaries. This allows end-to-end IP connectivity between Thread devices and application running on computers and smartphones, other devices on home local area Wi-Fi networks and even to an Internet server or a Virtual Private Network (VPN).

The Border Router example templates are recommended for devices meant to have a Thread radio on board as well as Wi-Fi or Ethernet capability and allow inter-connection to the external IP networks and the Internet.

3.5 Thread host-controlled interface

The Kinetis Thread Stack also includes Host-Controlled Interface application templates. This category of firmware is meant to be deployed in a multiple chip system, where a higher level Host application processor is driving the Thread network management and the application operation. Some Host devices can run higher level operating systems (for example, Linux ® OS) but typically do not incorporate an IEEE 802.15.4 radio or the base core Thread stack layers. These functions are still managed by the Kinetis device which acts as a Thread network co-processor.

This firmware implements the Thread Host Control Interface (THCI) serial bus protocol interfaced by default with UART or USB peripherals.

These multiple chip systems regularly posses Wi-Fi or Ethernet capability and, as a result, can fulfill a Border Router network role as described above.

3.6 Demo Applications Overview

Table 1. Demo Applications overview on page 10 shows an overview of the features enabled by default within the demo applications configurations.

Table 1. Demo Applications overview

Example App	Available board functionalities
router_eligible_device (template for **mains powered, always-on products driven entirely by Kinetis**: security control panels, standalone sensor hubs, range extenders, smart plugs, some thermostats, wall light switches, some light fixtures, some appliances)	**CoAP:** led, temp, data sink **UART:** shell **USB:** N/A **Commission:** auto-start collapsed commissioner on leader **Lib capability:** leader, router, reed
end_device (template for **mains powered** or **high-cpapacity battery products driven entirely by Kinetis** which are **NOT intended to be always-on**: light fixtures, appliances, some door locks, some thermostats, some resource constrained devices)	**CoAP:** led, temp, data sink **UART:** shell **USB:** N/A **Other:** rx on ed defaults **Lib capability:** rx on ed
low_power_end_device (template for **low-capacity battery Kinetis products**: sensors, remote controls, door locks)	**CoAP:** temp, data sink **UART:** N/A **USB:** N/A **LP:** LP mode 3 **Lib capability:** sed

Table continues on the next page...

Table 1. Demo Applications overview (continued)

host_controlled_device (template for **products where a Kinetis running the Thread stack is hosted by an application processor** over UART or SPI; use of Host SDK tools is recommended for HLOS UNIX host systems; serves as sub-component for advanced asymmetric multiple chip **border routers**)	**CoAP**: led, temp, data sink **UART**: THCI over FSCI **Serial TUN**: Not enabled **USB**: N/A **Lib capability**: leader, router, reed, rx on ed, ipv4, nd.
ble_thread_host_controlled_device (template for **products where a Kinetis running the dual mode Thread +Bluetooth LE stack is hosted by an application processor** over UART or SPI; use of the Host SDK tools is recommended for HLOS UNIX host systems; serves as sub-component for advanced assimetric multiple chip **border routers**)	**CoAP**: led, temp, data sink **UART**: THCI over FSCI **Serial TUN**: Not enabled **USB**: N/A **Lib capability**: leader, router, reed, rx on ed, ipv4, nd. **Bluetooth LE**: blackbox over FSCI
ble_thread_router_wireless_uart (template for **products where a Kinetis device running the dual mode Thread + Bluetooth LE stack is controlled by a Bluetooth LE controller, such as the Kinetis Bluetooth LE Toolbox Wireless UART** to connect to the Thread stack and perform Thread network management operations)	**CoAP**: led, temp, data sink **Bluetooth LE:** virtual shell **UART**: N/A **USB**: N/A **Lib capability**: leader, router, reed **Commission:** auto-start collapsed commissioner on leader
border_router (template for **products where a Kinetis device is running the IP stack on multiple interfaces, including Thread**)	**CoAP**: led, temp,data sink **UART**: Shell **USB**: RNDIS **Lib capability**: leader, router, reed, rx on ed, nd **Commission**: auto-start collapsed commissioner on leader

CoAP: led -- The app has a CoAP callback which controls the LEDs on the board (also RGB pin mode with FTM on FRDM-KW24D)

• **CoAP: temp --** The app has a CoAP API which sends the chip temperature over the air.

• **CoAP: data sink --** The app has a CoAP API which advertises via multicast the node as a sensor data data sink, enabled LED and temperature data to be then sent unicast to the node.

• **UART: shell --** Shell is enabled over UART (via OpenSDA chip on our boards)

• **USB: shell --** Shell is enabled over direct USB CDC Virtual Serial Port stack on KW2xD

• **UART/USB: N/A --** UART interface (via OpenSDA) or direct USB is disabled

• **ETH: N/A --** Ethernet interface is disabled app.

• **ETH: ND_ROUTER --** Ethernet interface enabled. Ethernet/Wi-Fi IPv6 Neighbor Discovery provides site-local prefix assignment over the Ethernet link.

• **UART: THCI over FSCI --** Thread Host Control Interface (Thread logical host control protocol) over Serial Connectivity interface over UART (on OpenSDA chip)

• **Serial TUN: enabled/disabled --** Serial tunnel feature over THCI/FSCI is disabled/enabled

• **USB: ND_ROUTER over RNDIS, no THCI --** RNDIS (Virtual Ethernet over USB CDC) Interface enabled. Neighbor Discovery provides site-local prefix assignment over the virtual USB Virtual Ethernet (RNDIS) driver. THCI is not carried over RNDIS

• **USB: ND_ROUTER over RNDIS, with THCI --** Same as above, but THCI is carried over a RNDIS subchannel, under IP under a different EtherType – works with the Linux Host SDK

• **Commission: auto-start collapsed commissioner on leader --** A commissioner application auto-starts on the first Leader Router which creates the Thread network. It's called collapsed as the Leader and Commissioner network roles are 'collapsed' on the same node.

• **Lib Capability: leader --** Can become a Leader at runtime

• **Lib Capability: router --** Can become a router at runtime

• **Lib Capability: reed --** Can act as a router eligible End Device at runtime.

• **Lib Capability: polling ed --** Includes the MAC polling mode for an End Device at runtime

• **Lib Capability: sed --** Includes low-power end device capabilities (ability to enter low-power)

• **Lib Capability: rx on ed --** Includes the MAC RXON mode for an end device at runtime

• **Lib Capability: ipv4 --** Includes IPv4 and DHCPv4 client features

• **Lib Capability: nd --** Includes IPv6 Ethernet/Wi-Fi Neighbor Discovery features

• **Other: rx on ed default --** The End device is RXON (non-polling) by default

• **Low-power --** The End device has low-power (deep sleep) enabled by default (users can wake up the End Device via button press).

Chapter 4
Deploying Thread Stack and Applications Software

The Kinetis Thread Stack software package includes the components necessary to begin Thread wireless mesh network application development on Kinetis platforms. These components include:

- Thread Internet Protocol (IP) based mesh network software libraries
- IEEE 802.15.4 Media Access Control (MAC) software libraries and Physical Layer (PHY) drivers
- Example consisting in demo application firmware projects corresponding to different Thread node or device categories
- Helper utilities such as a serial/USB port shell and Linux host enablement software
- Precompiled example firmware, Host Control interface descriptors and other tools and drivers.

The Kinetis Thread Stack software package includes the Kinetis peripheral drivers, platform startup code, RTOS kernel software or other generic Kinetis platform software as provided by the Kinetis SDK (KSDK).

4.1 Downloading the KW41 Connectivity Software

To download the Thread Stack software, perform the following steps:

- In a web browser, access the Thread network technology home page nxp.com/thread.
- Obtain and Download the desired Package
- Copy the package into C:\NXP**SDK_2.2_MKW41Z512xxx4** folder **for example.**

4.2 Supported integrated development environments

The IAR® Embedded Workbench for Arm® (IAR EWARM) Integrated Development Environment (IDE) and MCUXpresso IDE and toolchain are required to customize, compile, and debug the example demo applications included with the Kinetis Connectivity Software installation package. IAR EWARM or MCUX are also required to develop similar applications using the given examples as templates.

To download an evaluation version or to acquire a full version of IAR EWARM, visit the web page iar.com/kinetis.

To download the MCUXpresso Integrated Development Environment (IDE), see the NXP page: MCUX IDE download.

Chapter 5
Kinetis Thread Hardware Platforms

The Kinetis Thread Stack version 1.1.1 supports the following platform which have radio frequency transceiver capabilities compatible with the IEEE 802.15.4 standard used by Thread:

- Kinetis KW41z Wireless MCU family - integrates an Arm Cortex ® -M0+ core with an IEEE 802.15.4 transceiver

Chapter 6
Deploying Applications with IAR EWARM

6.1 Project launch files

Each of the example demo projects provide a set of separate IAR EWARM launch files for each of the supported Kinetis Thread development board and for each RTOS option. The launch files consist in:

- <*.eww> workspace configuration file
- <*.ewp> project configuration file
- <*.ewd> debug configuration file

The launch files for each of the configurations are contained in the folder structure starting at the boards subfolder in the root of the Kinetis Thread Stack installation.

For instance, to access the launch files for the **Thread Border Router** example to deploy on the **USBKW41Z** development platform which uses **FreeRTOS**, navigate to the following subfolder:

<Connectivity_Software_Installation, for example, C:\NXP\SDK_2.2_MKW41Z512xxx4>

\boards\usbkw41z_kw41z\wireless_examples**thread\border_router\freertos\iar**

Each configuration subfolder has the following generic structure:

<Connectivity_Software_Installation, for example, C:\NXP\SDK_2.2_MKW41Z512xxx4>:

\boards**<board>**\wireless_examples\thread\ **<example application>** **<RTOS>**\ **<toolchain>**

Each configuration subfolder contains the set of workspace, project, and debug configuration files, for instance:

- border_router.ewd
- border_router.ewp
- border_router.eww

6.2 Opening a workspace

To open an example Kinetis Thread demo application into IAR EWARM for development: double-click or press Enter to launch the **<*.eww>** workspace file - such as **border_router.eww**

IAR EWARM opens and display the projects referenced in the workspace file.

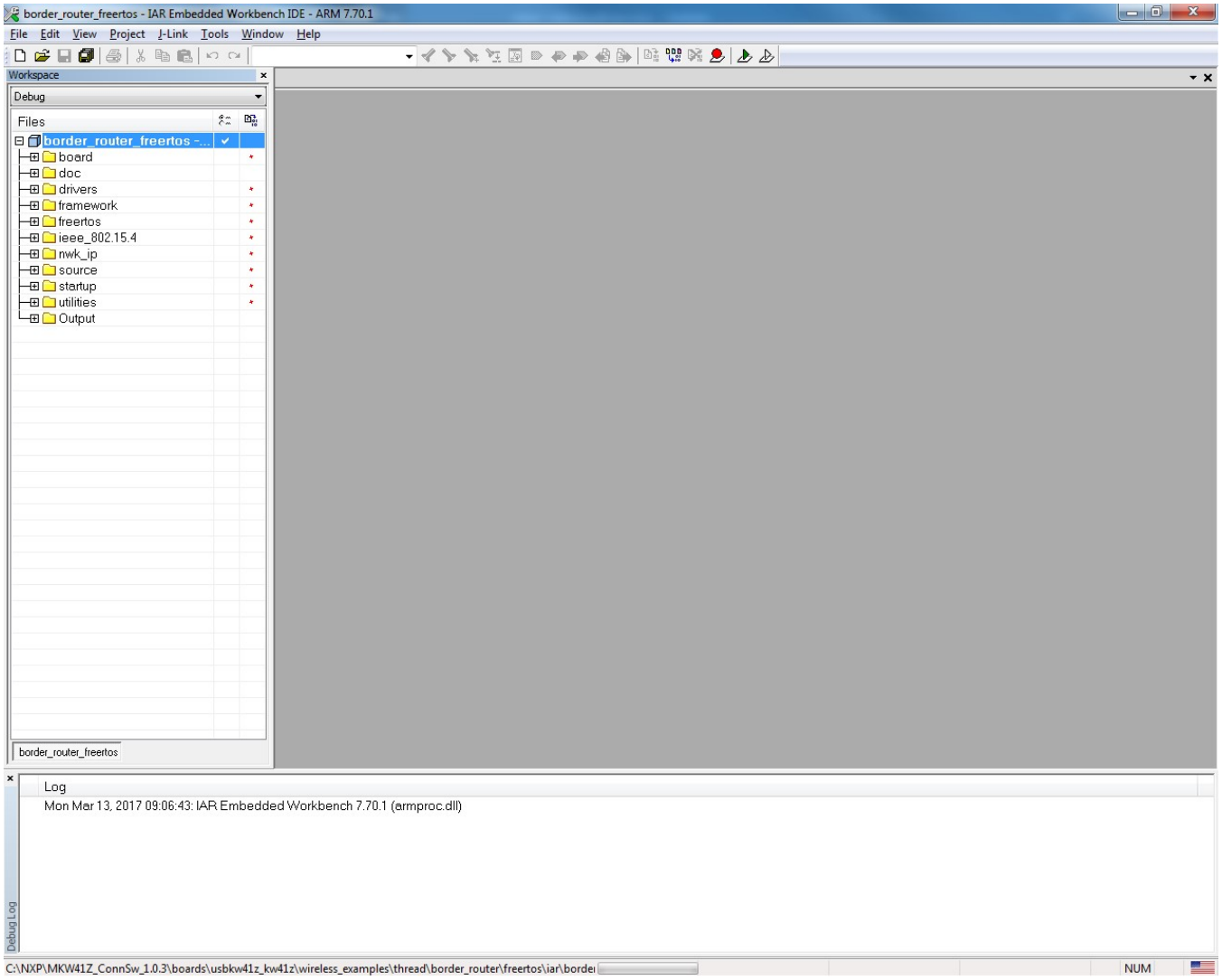


Figure 3. IAR EWARM workspace

6.3 Workspace contents

Once opened, the project for the main application (executable) found in **border_router.ewp** is displayed first.

6.4 Project configurations

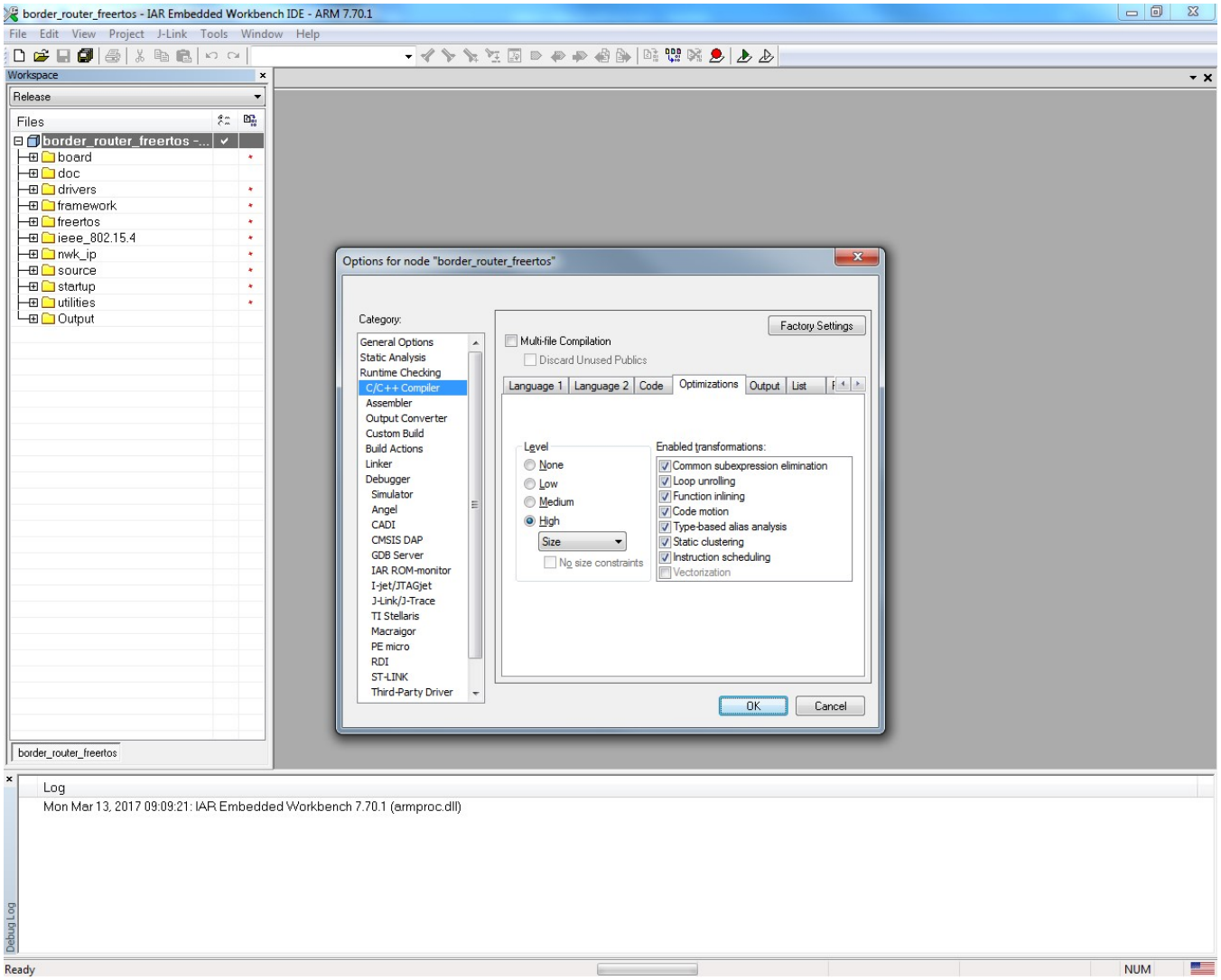


Figure 4. EWARM Release Configuration Options

6.5 Building the application executable

To build the application executable firmware: in the EWARM Workspace navigator, right click the application name and select **Make**.

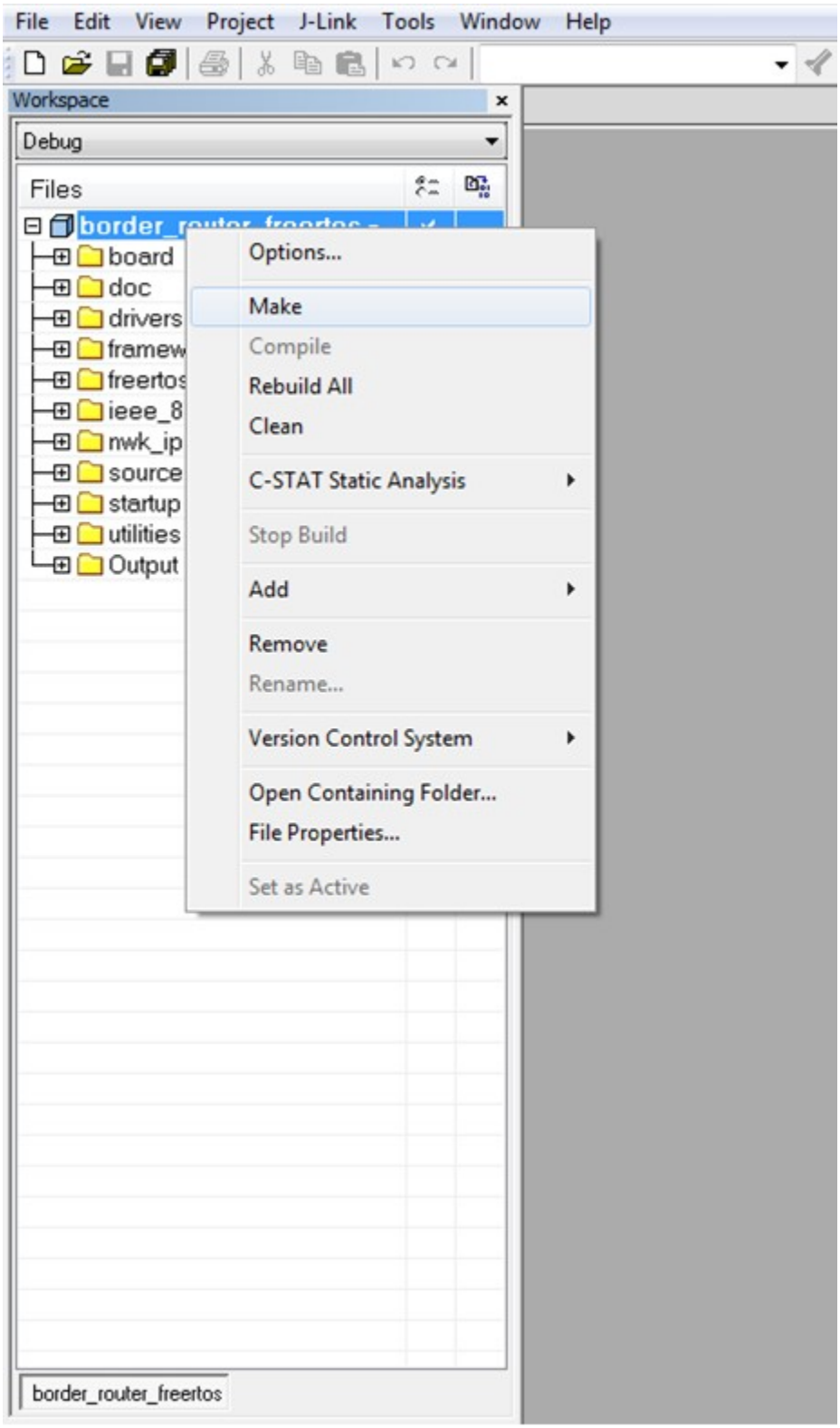


Figure 5. IAR EWARM Building Firmware

6.6 Deploying the firmware using the debugger connection

After building the executable, connect the development board using the OpenSDA USB connection.

Note that the IAR project **Debugger Driver** option matches the board and interface used. To change the Debugger option, right click the application name, then select **Options -> Debugger-> Driver**.

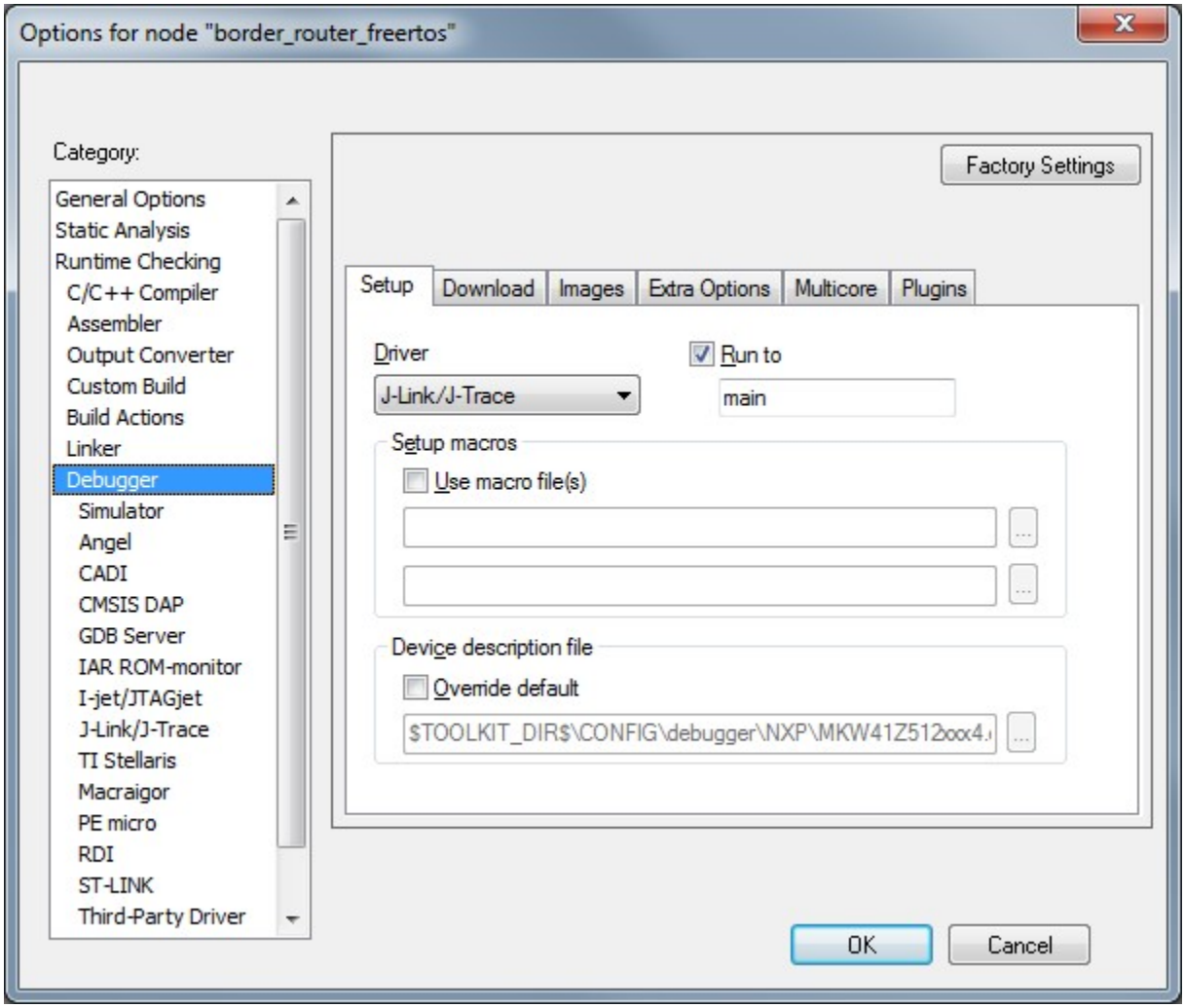


Figure 6. Setting Debugger Options

Table 2. Debugger Options

Development Board	Default Debugger Connection in Example Project	Other debugger options
FRDM-KW41Z	**J-Link/J-Trace** via OpenSDA USB	**CMSIS DAP** via OpenSDA USB (see nxp.com/opensda for reprovisioning)
USB-KW41Z	**J-Link/J-Trace** via OpenSDA USB	CMSIS DAP via OpenSDA USB (see nxp.com/opensda for reprovisioning)

To deploy the application:

- Ensure the application project is active in the Workspace navigator.
- Click **Download and Debug** in the toolbar.

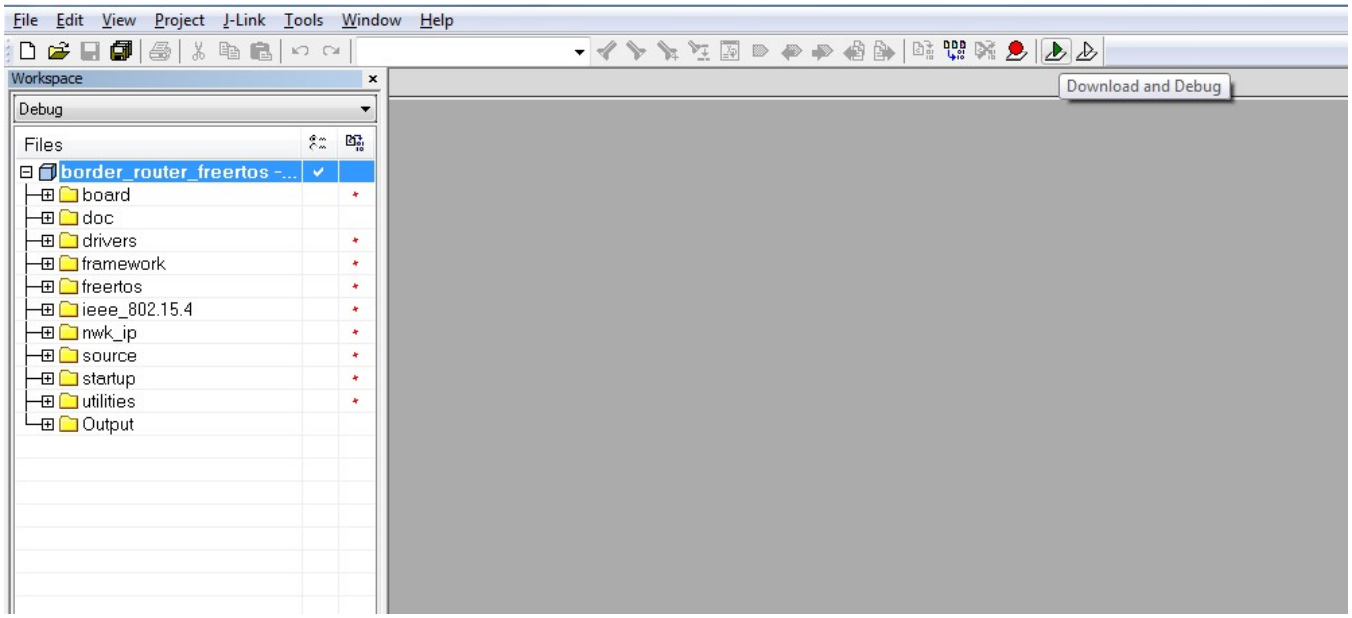


Figure 7. Download and Debug Firmware to Development Board

- Wait for the projects to build if still required.
- Note the specific debugger connection messages and wait for the flash download and verification .

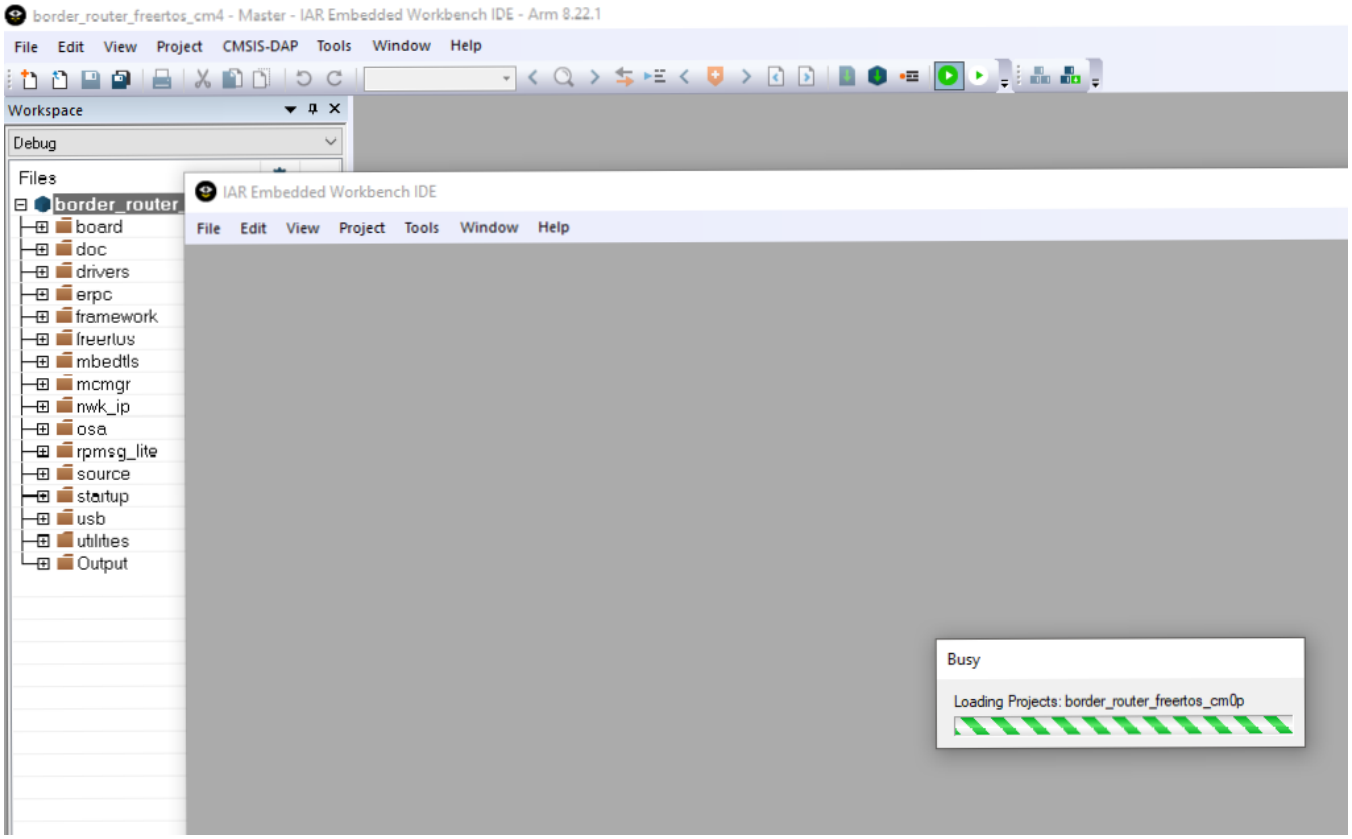


Figure 8. Flash Download Progress

- Wait for the program to run to **main**

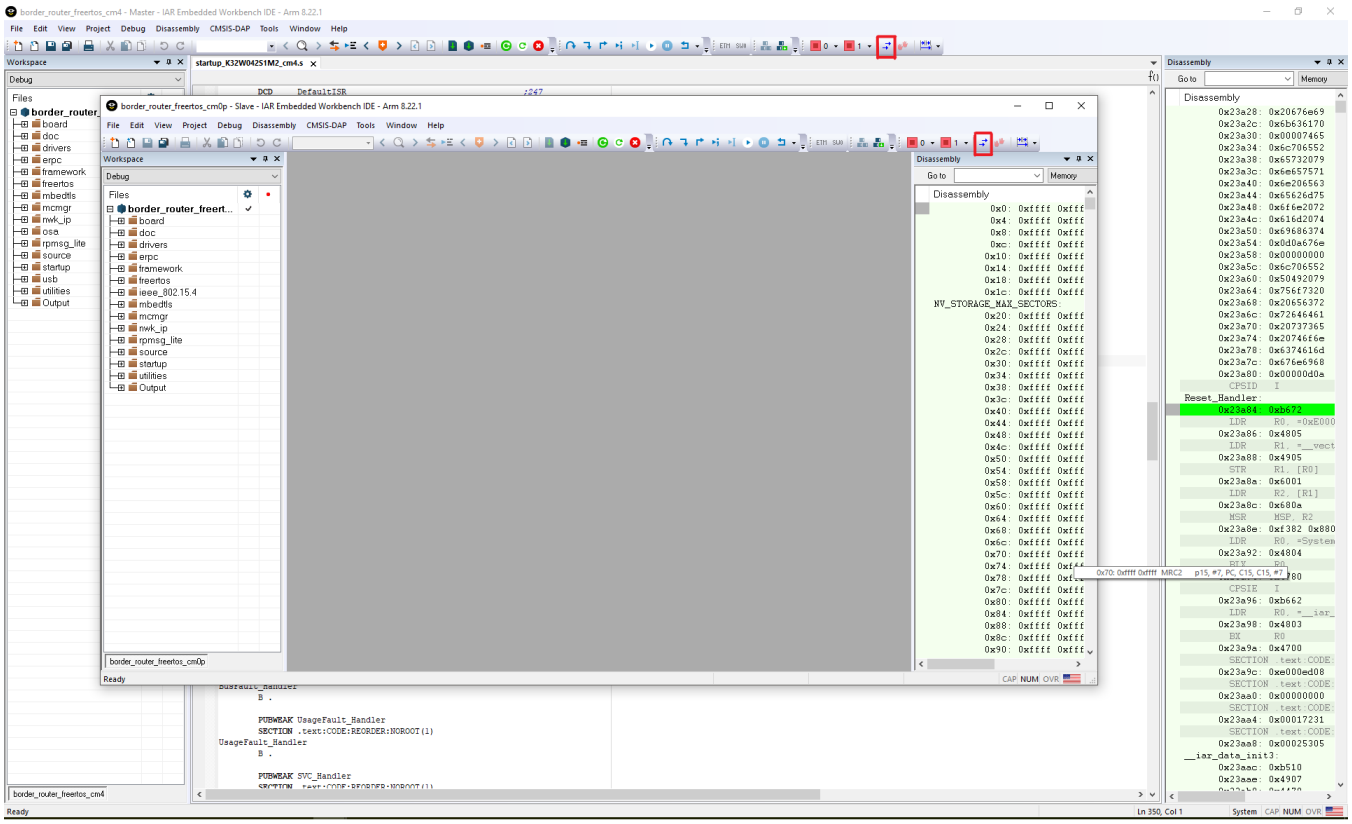


Figure 9. Download and Debug Firmware to Development Board

- Click **Run** in the debugger window to start the application debug
- Click **Stop Debugging** and reset the board if the debugger attachment is not needed

6.7 Using EWARM batch build

EWARM allows users to build multiple projects and configurations in a batch. To use batch build with a Kinetis Thread Stack configuration workspace:

- Press F8
- In the **Batch Build** window select the configurations for batch build: Debug, Release or all (Debug and Release)
- Choose one of the **Make**, or **Rebuild All** options at the bottom of the window to build the selected batch configuration, or **Clean** to remove build artifacts

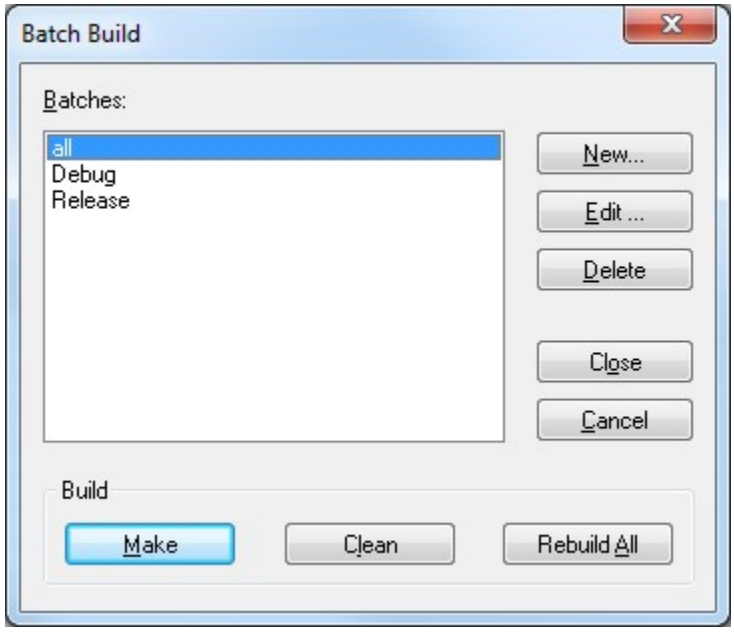


Figure 10. EWARM Batch Build

Chapter 7
Deploying Applications with MCUXpresso IDE

7.1 Project launch files

Each of the example demo projects provides a set of separate MCUX launch files for each of the supported Kinetis Thread development boards and for each RTOS option. The launch files consist of the following.

- <.xml> C Project configuration file
- <.xml> MCUX Project configuration file

The launch files for each of the configurations are contained in the folder structure starting at the **boards** subfolder in the root of the Kinetis Thread Stack installation.

For example, to access the launch files for the **Border Router** example to deploy on the USBKW41Z development platform which uses FreeRTOS OS, navigate to the following subfolder.

<Kinetis Thread_Stack_Installation, for example, <C:\NXP\SDK_2.2_MKW41Z512xxx4>:

\boards**usbkw41z_kw41z**\wireless_examples\thread**border_router****freertos**\

- border_router_freertos.xml

- border_router_freertos.xml
- example.xml

7.2 Updating OpenSDA Serial and Debug Adapter Image Firmware

Download and install the latest OpenSDA firmware image of choice from nxp.com/support/developer-resources/run-time-software/kinetis-developer-resources/ides-for-kinetis-mcus/opensda-serial-and-debug-adapter:OPENSDA.

7.3 Opening a workspace

To open an example Kinetis Thread demo application in the MCUX IDE for development use right click on “Installed SDKs” tab and choose “Import Folder” and select the MCUX SDK to be imported.

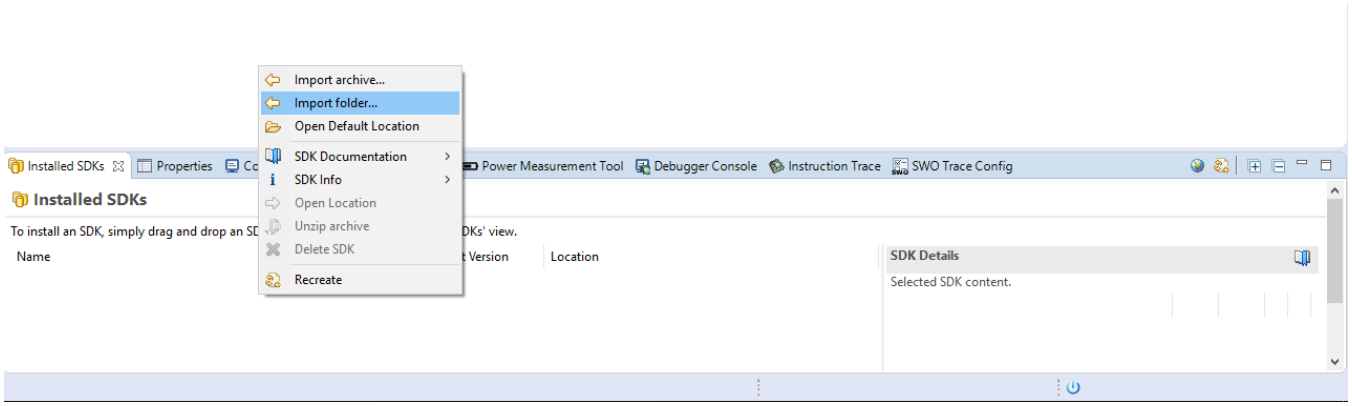


Figure 11. MCUXpresso Import Folder

When the MCUX import, window opens, wait a couple of seconds for selected SDK to be imported.

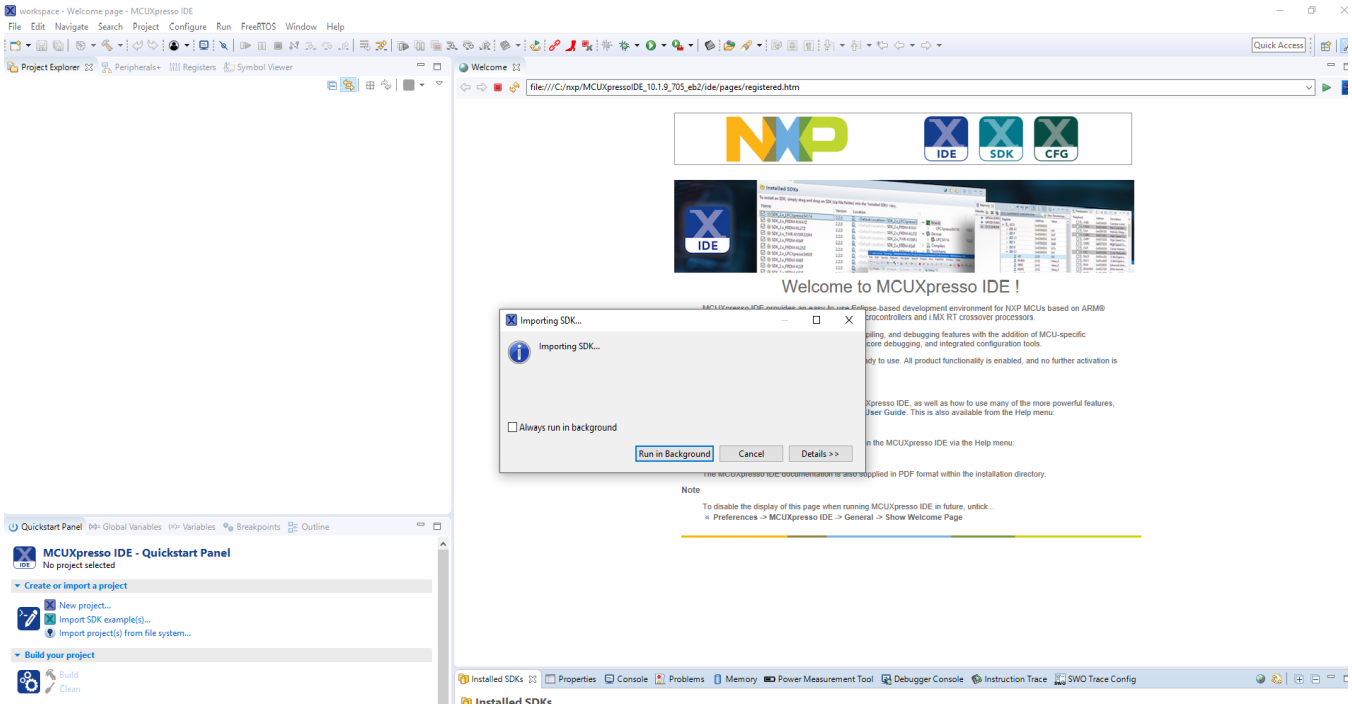


Figure 12. MCUXpresso Importing Project

When the Import SDK files finishes, select “Import SDK example(s)…” menu.

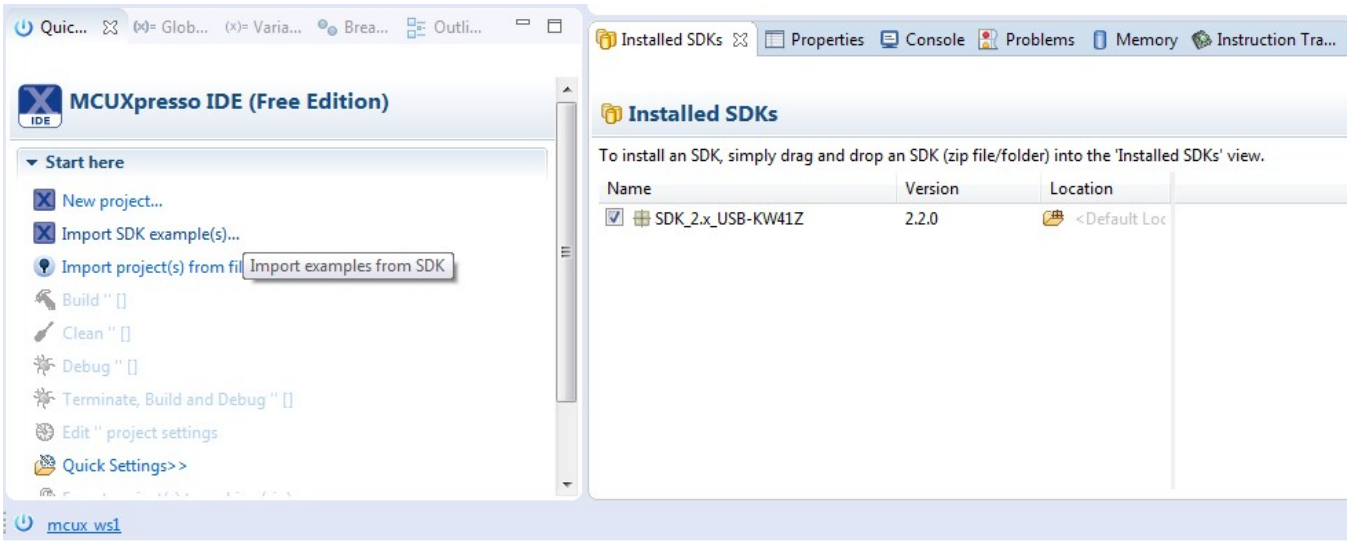


Figure 13. MCUXpresso Import Selected Project

Select the Connectivity Software, for example, <C:\NXP\SDK_2.2_MKW41Z512xxx4>: \boards**usbkw41z_kw41z** \wireless_examples\thread**border_router**\freertos\, then press OK.

Select the desired example (for example, wireless_examples\thread\border_router\freertos).

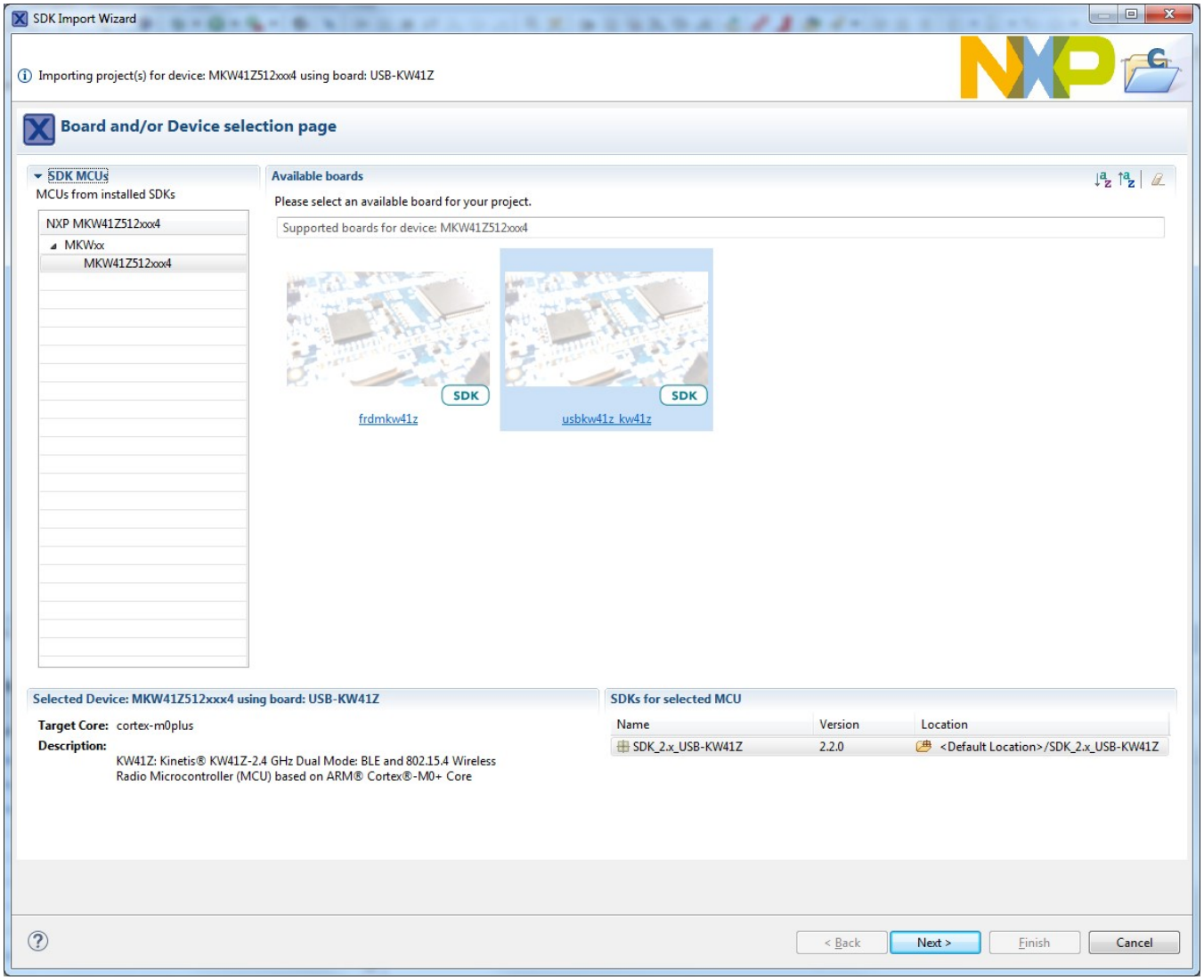


Figure 14. MCUXpresso Select Board

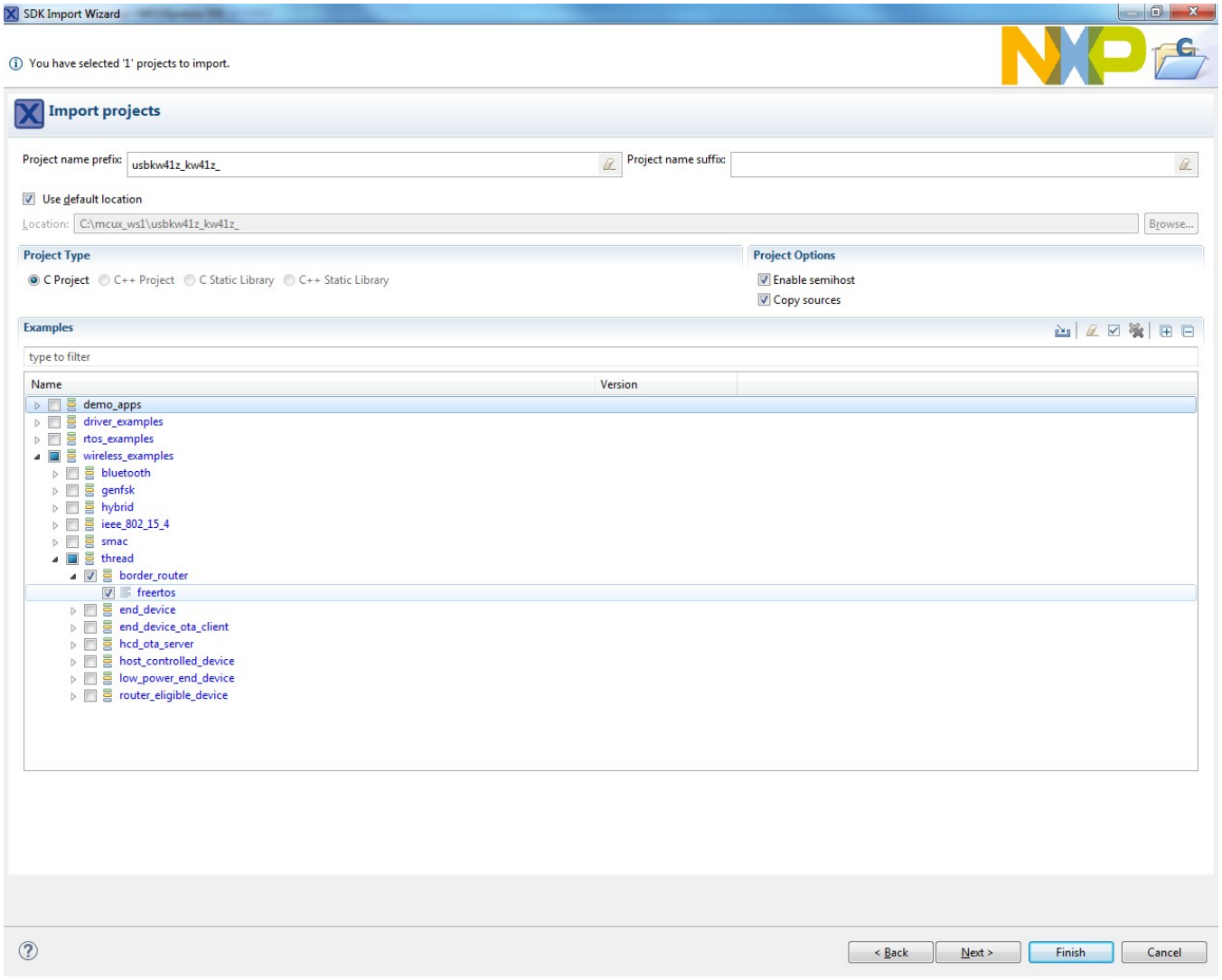


Figure 15. MCUXpresso Selected Project Imported

Click Finish on the Import Projects dialog box. The project is successfully imported into MCUX IDE.

7.4 Workspace contents

Once opened, the project for the main application **border_router_freertos** is displayed first and highlighted.

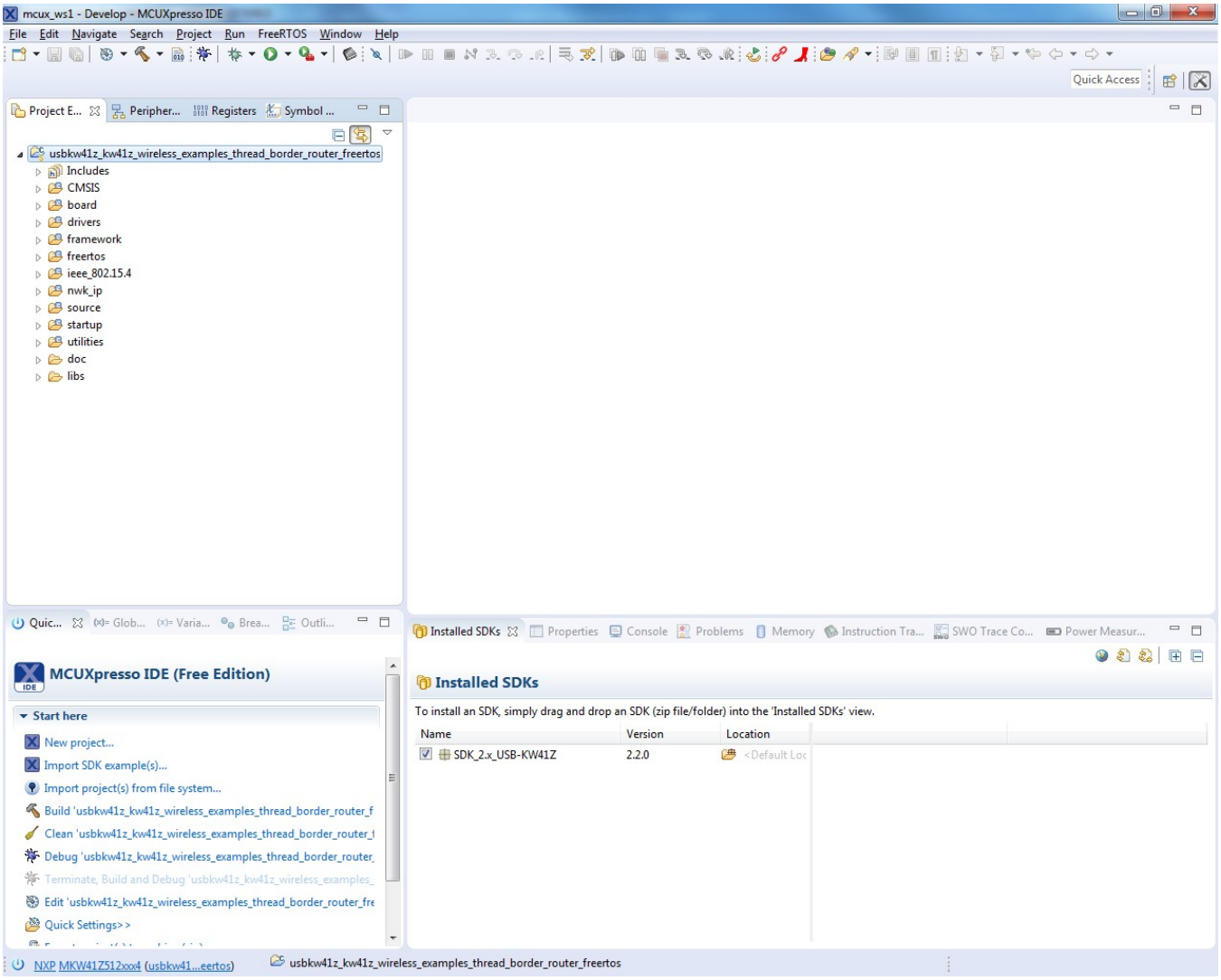


Figure 16. MCUXpresso Workspace

7.5 Project configurations

The optimization level for the Release configuration is set to Size(-O0)

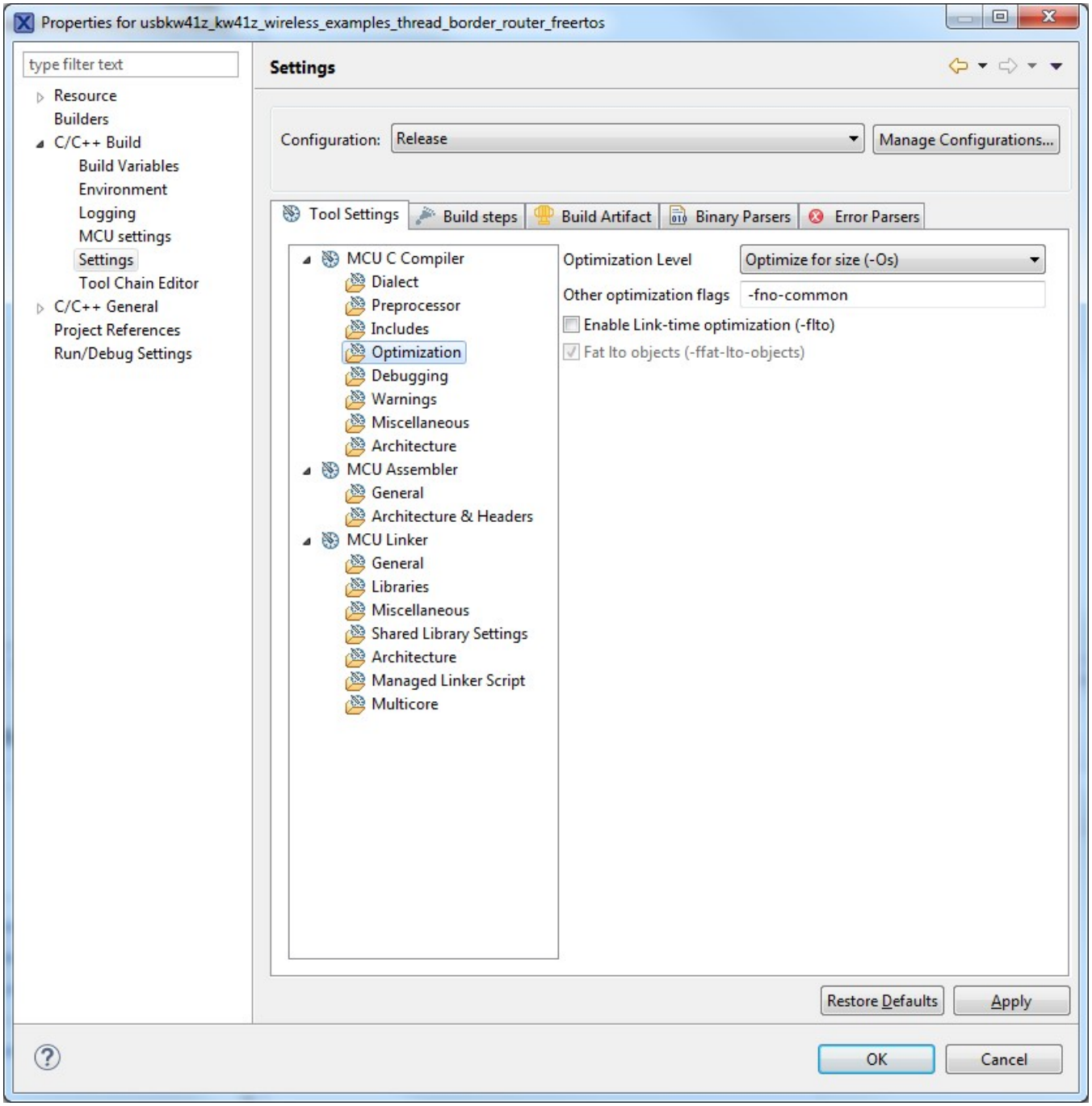


Figure 17. MCUXpresso Release Configuration Options

The optimization level for the Debug configuration is set to Optimize for debug (-Og).

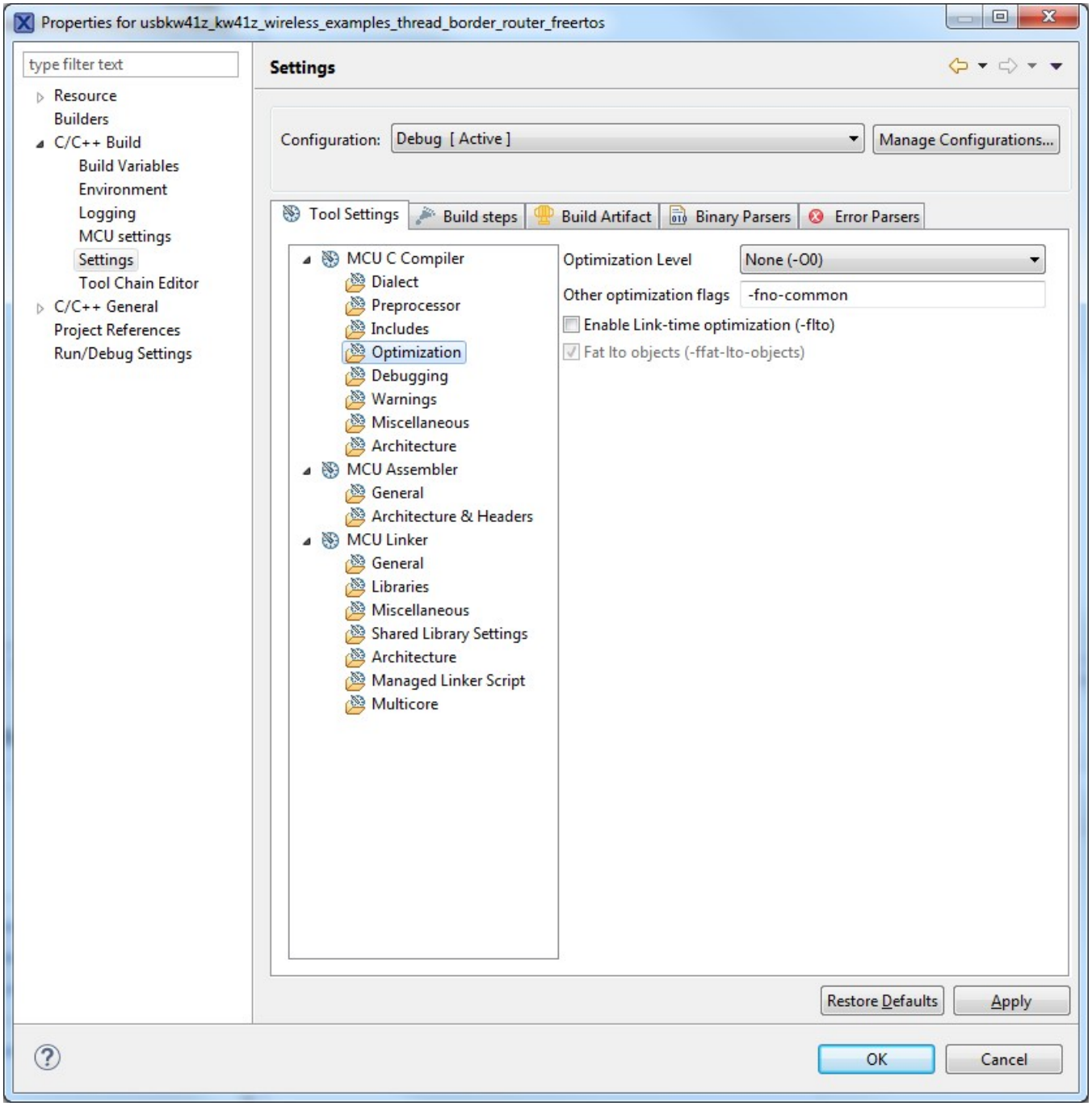


Figure 18. MCUXpresso Debug Configuration Options

7.6 Building the application executable

To build the application executable firmware in the MCUXpresso Workspace navigator, right click the application name and select **Build Project**.

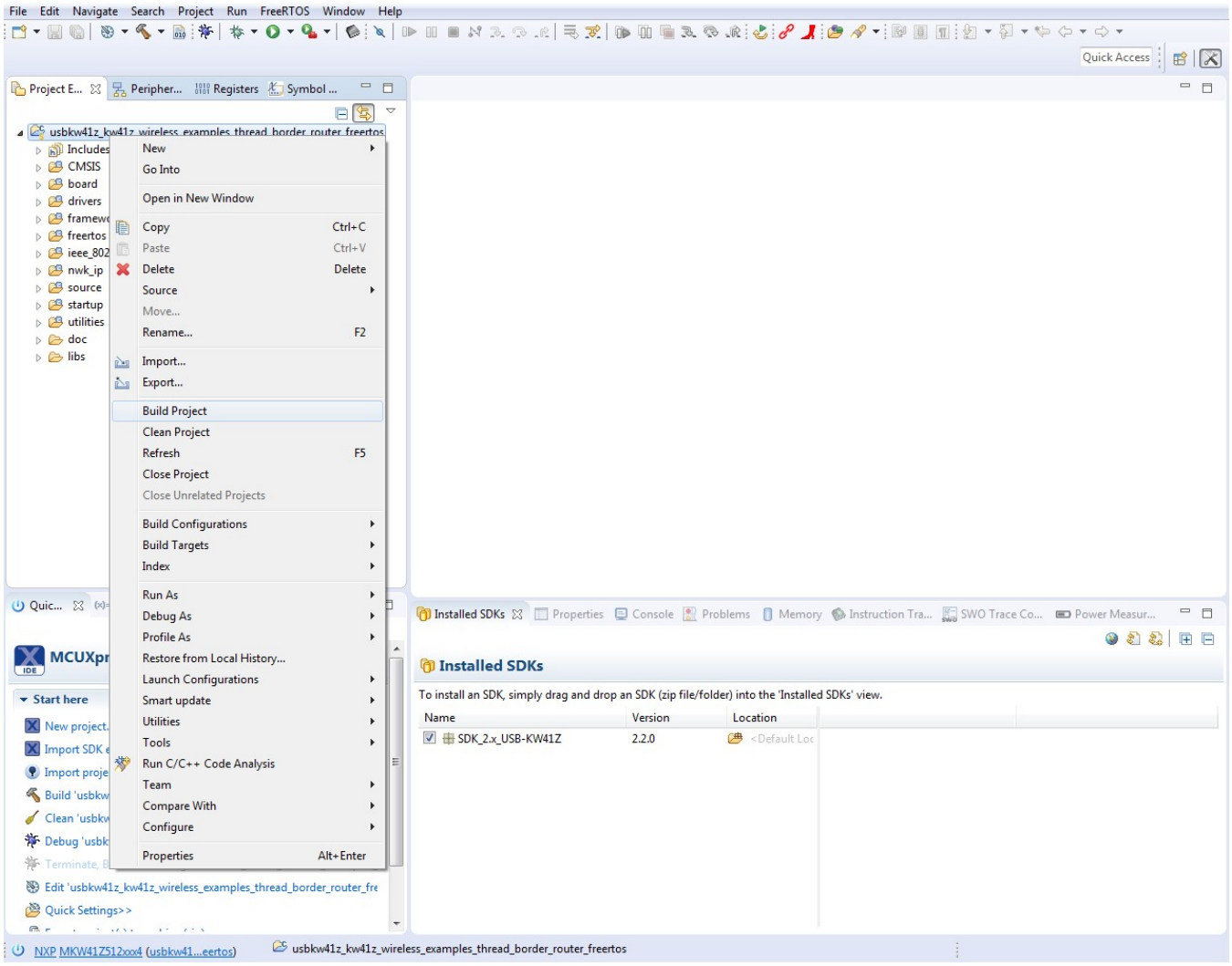


Figure 19. MCUXpresso Building Firmware

According to the selected project configuration (Debug or Release), a .srec file is generated in the corresponding …/Debug/ Release folder, after the project is built.

7.7 Deploying the firmware using the debugger connection

After building the executable, connect the PC to the development board using the OpenSDA USB connection.

The MCUX Debug Configuration dialog box is available using the Run->Debug Configuration menu option.

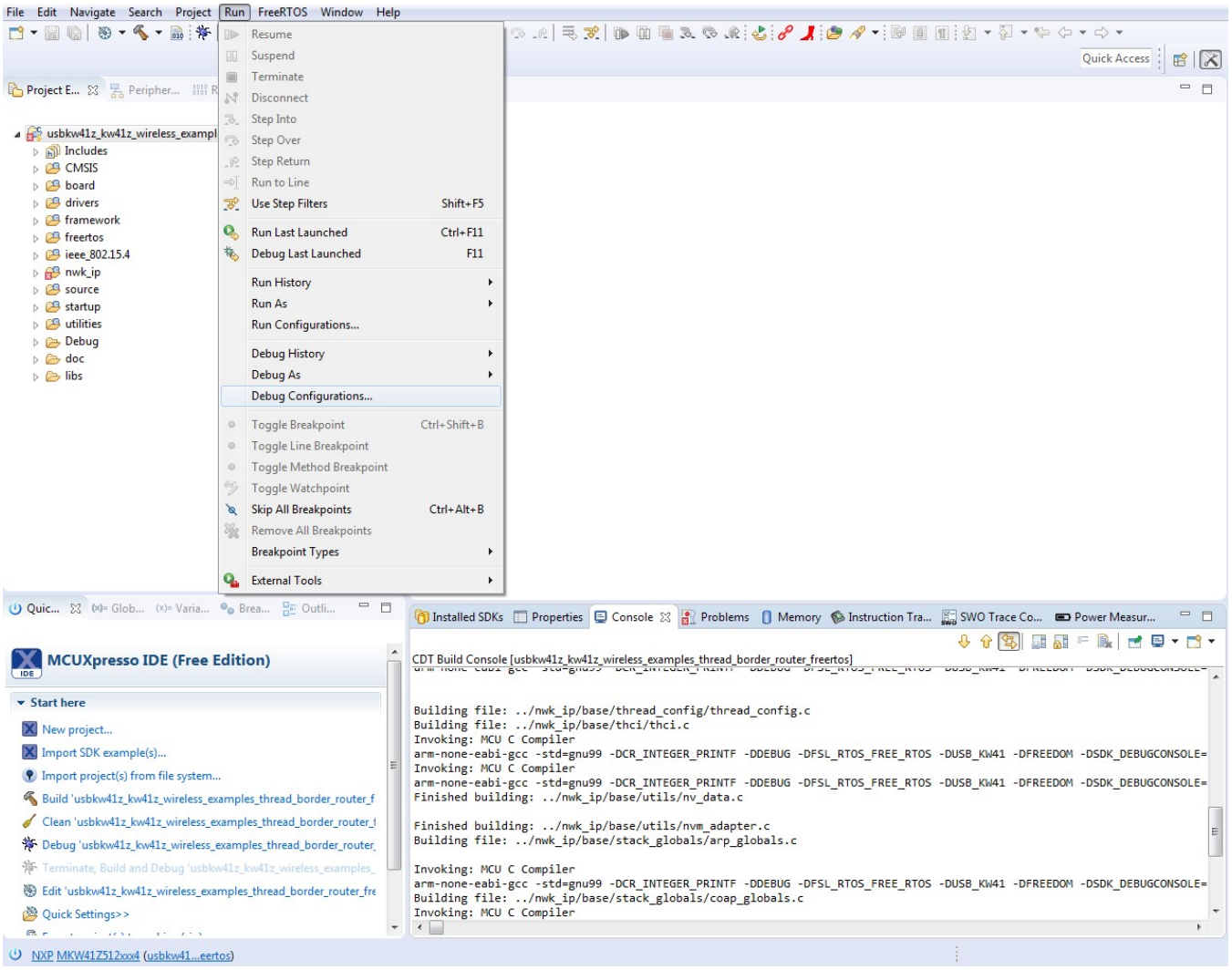


Figure 20. MCUXpresso Debug Confirmation

Then the following dialog box is displayed to select the needed debug options.

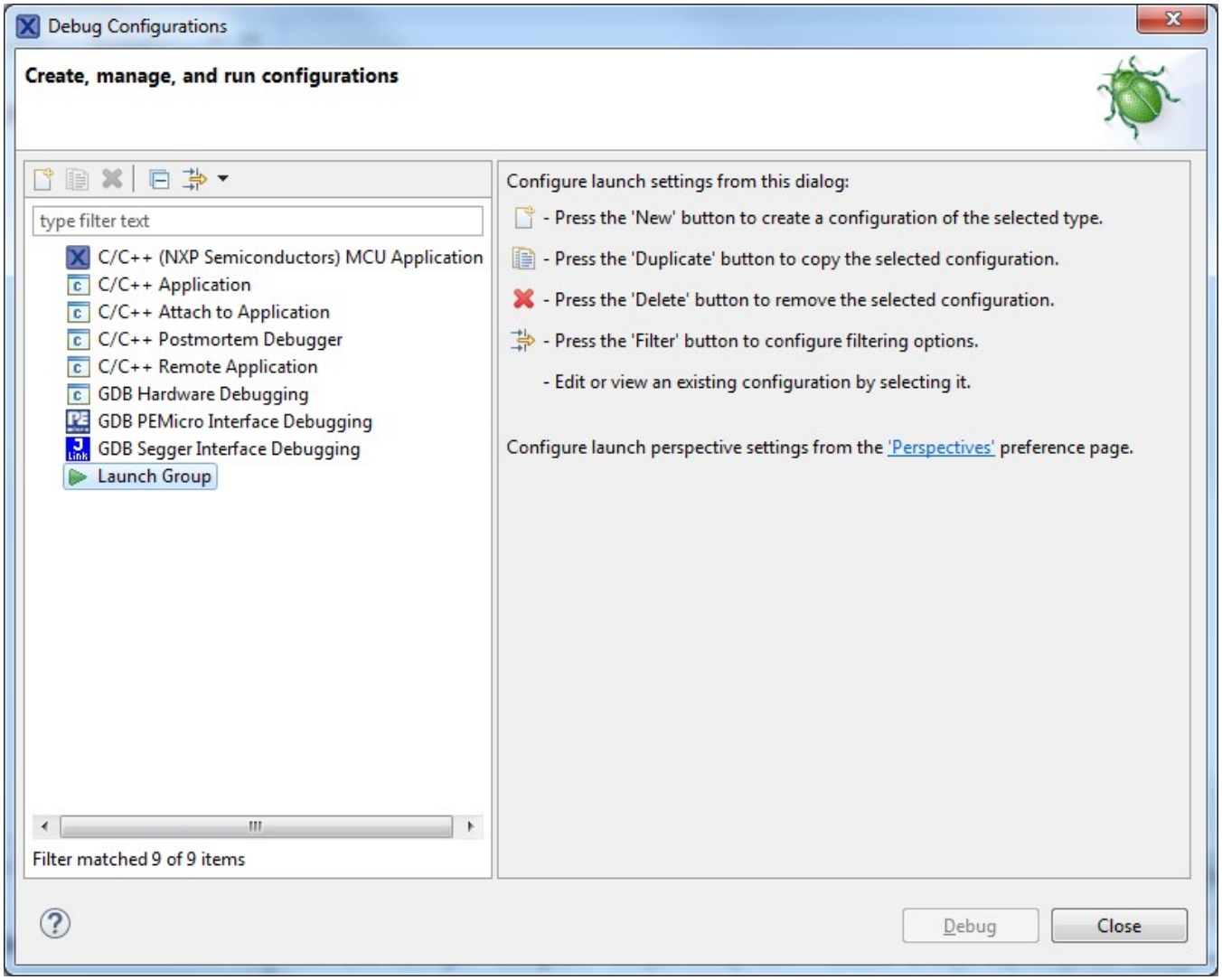


Figure 21. MCUXpresso Setting Debugger Options

To deploy the application, do the following:

- Use the “Drag & Drop” Windows Explorer functionality and copy the files into selected board (MSD functionality)

To deploy and debug the application, do the following:

- Use the “Debug <application_name> [Debug]” from the Quickstart Panel, as shown below.

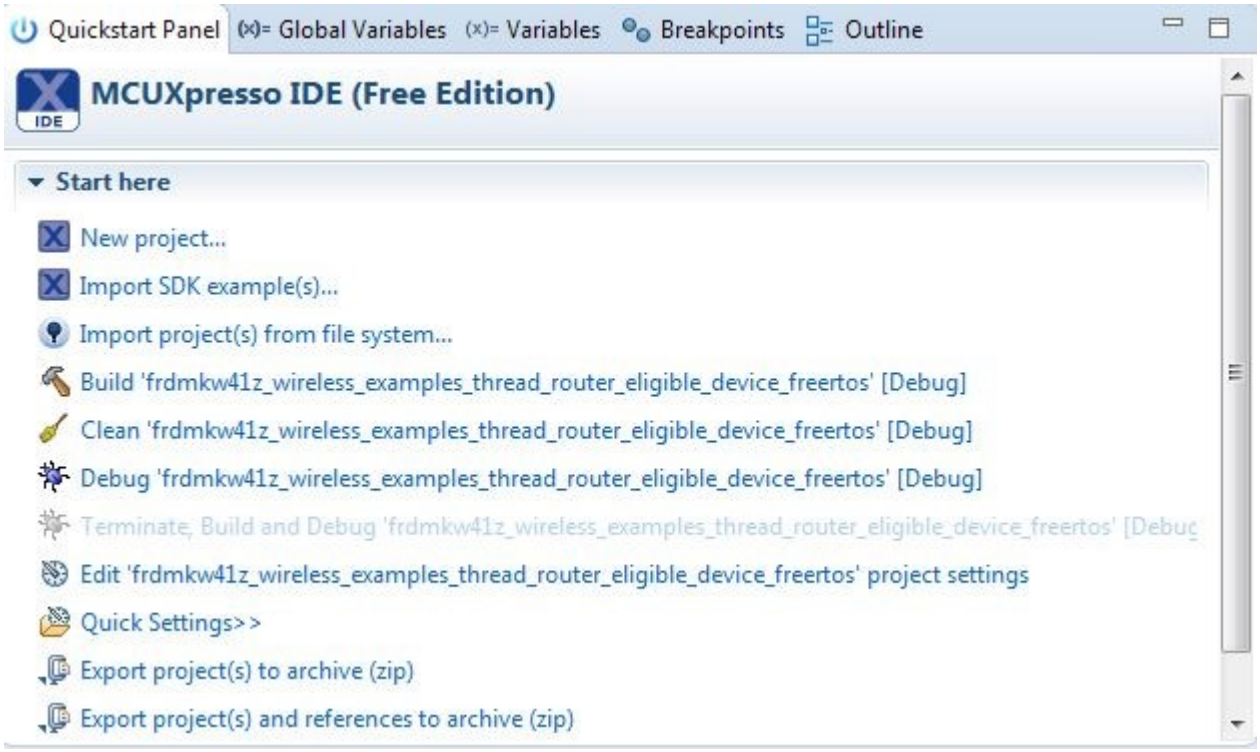


Figure 22. Deploying and debugging the REED application in MCUXpresso

- A new window will appear, requesting the choosing of the board

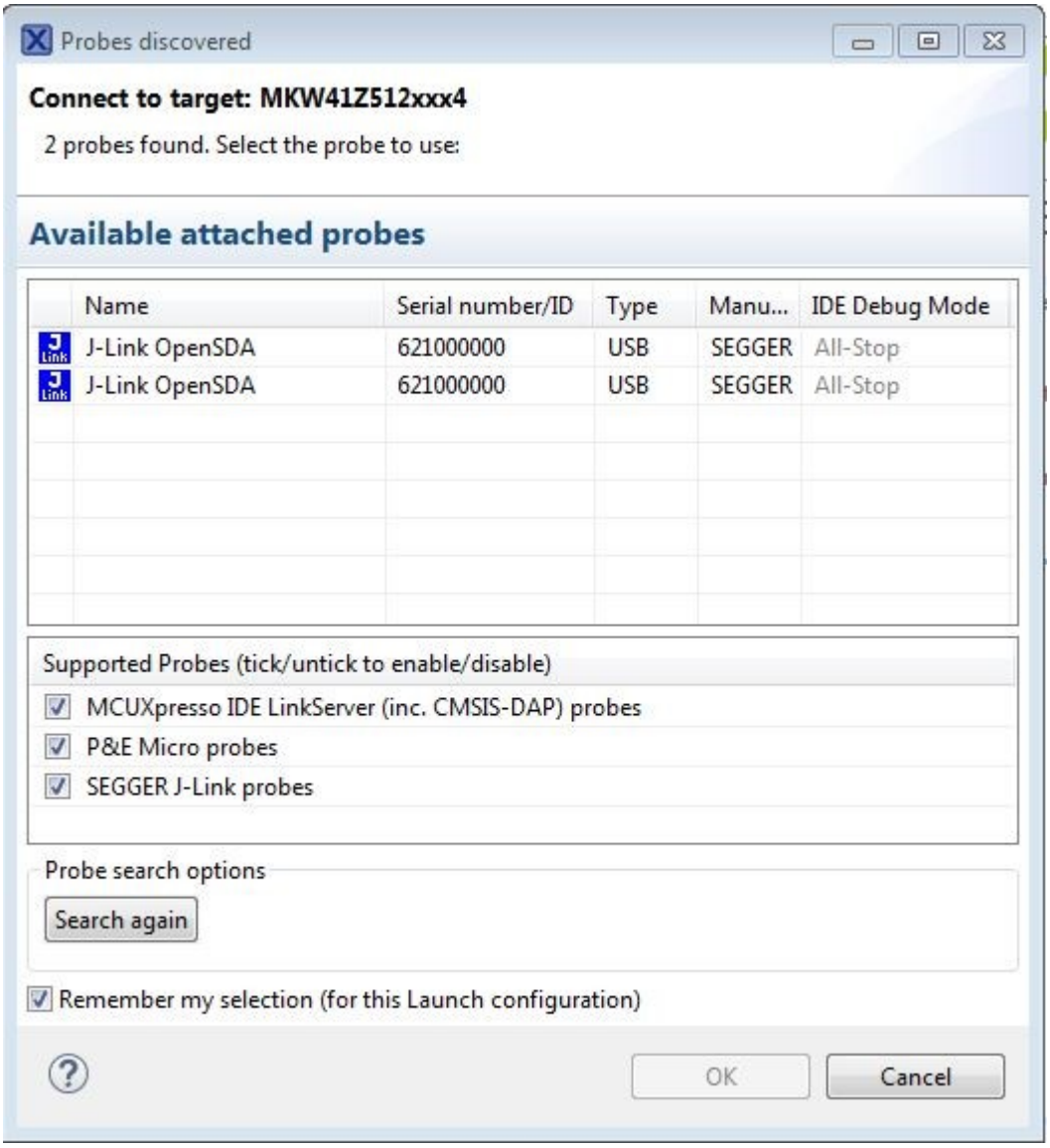


Figure 23. Choosing the board to deploy the application

- After choosing the board, application will be deployed to the board and debugging can commence as shown in figure

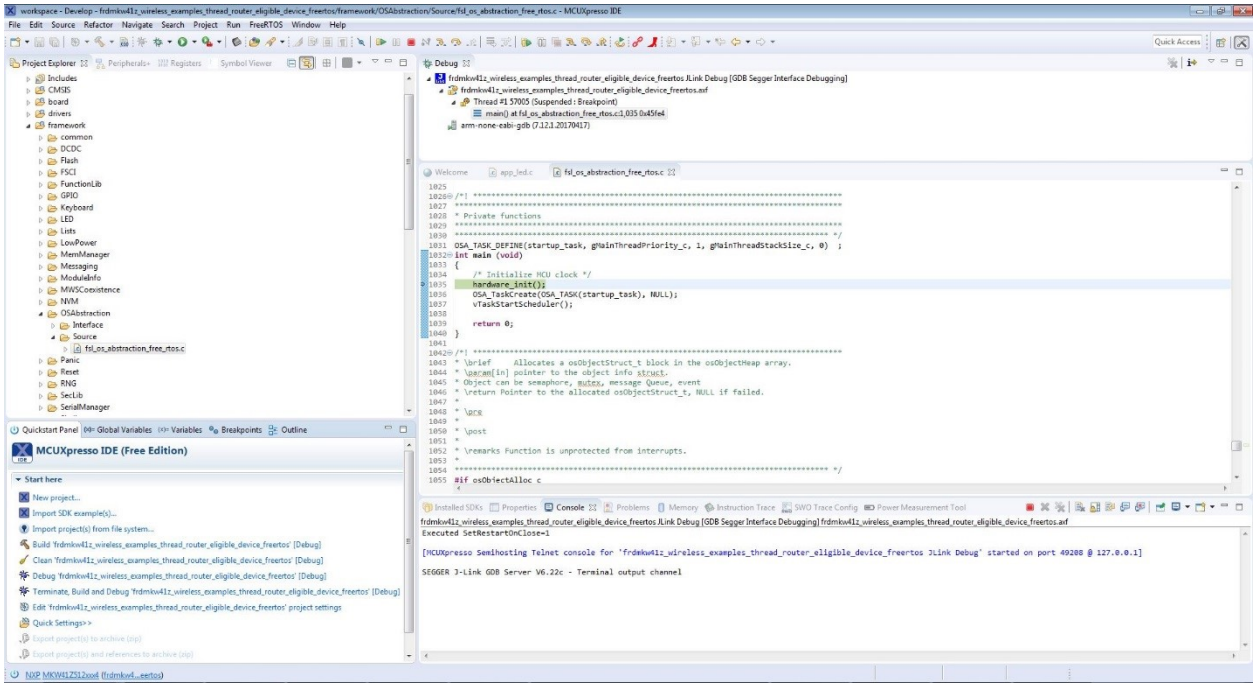


Figure 24. Deployment and debugging in MCUXpresso

After the project is started, the M0+ project can be also debugged in the same way: select the M0+ application from the projects list and then press the "Debug" link in the Quickstart Panel. A new debug session will start and attach to the M0+ core.

Chapter 8
Demo Functionality Overview

The example projects show the following Thread network and application layer functionalities:

- **Creating (bootstrapping) new Thread networks**
- **Joining existing Thread networks** of new devices based on predefined or customized network security
- **Configuring commissioning parameters** for joining of new devices including both the Joiner device configuration as well as the Commissioner configuration
- **Inspecting IPv6 Address Assignment** for Thread device interfaces
- **Using ICMP Ping** to test basic IP layer inter-connectivity
- **Sending Unicast and Multicast Application Data** using the **Constrained Application Protocol (CoAP)** message format over **UDP**
- **Inspecting and configuring network parameters** such as routing and neighbor tables, link-layer addresses, or radio channel parameters
- **IPv6 Border Routing** to applications and devices operating on different IP subnets and link layer technologies
- Noting **the differences in network behavior** between Routers, End Devices, and Low-power End Devices
- Facilitate **integration of Kinetis Thread devices in multiple chip configuration** driven by an external Host application processor

Chapter 9
Running Thread Network Scenarios

In order to build complex network topologies, the details below are referring to other types of boards that can be used in conjunction with the existing release for FRDM-KW41Z, USB-KW41Z_K22F, and USB-KW41Z_KW41Z.

9.1 Board setup and provisioning

The network scenarios described in the following sections go through Thread network use cases and functionalities shown by the demo applications.

These scenarios assume availability of a minimum of 2 (two) Thread development boards in the following supported board set.

- FRDM-KW41Z
- USB-KW41Z (limited board user interface)
  - — USB-KW41Z_K22F (RNDIS interface for Border Router)
  - — USB-KW41Z_KW41Z

Availability of more than 2 boards is useful in verifying application behavior in larger mesh networks consisting in multiple nodes or for deployment of end device multihop scenarios.

Different types of development boards can be mixed within the mesh network depending on availability.

For some scenarios, sequential identification of boards is useful. As such, the boards used in each scenario may be referred to in order based on their device type such as Node A, Node B, Node C, etc., such as below:

Table 3. Node and Application configuration

Node	Application
Node A	Thread Router Eligible Device
Node B	Thread End Device
Node C	Thread Border Router

9.2 Factory default state

9.2.1 Factory default

A device is placed a **Factory Default** state after initial application firmware loading as well as when it has been purposefully reset to this state as shown in the **Factory Reset** section.

In the Factory Default state, the device is **idle** from a Thread network perspective: it is not actively connected to any network and is provisioned with the original default firmware settings. In the Factory Default state, the device is ready for a user initiated action such as creation of a new Thread network or joining an existing network.

The **Factory Default** state is indicated by the development boards **blinking RGB LED blue and red led**.

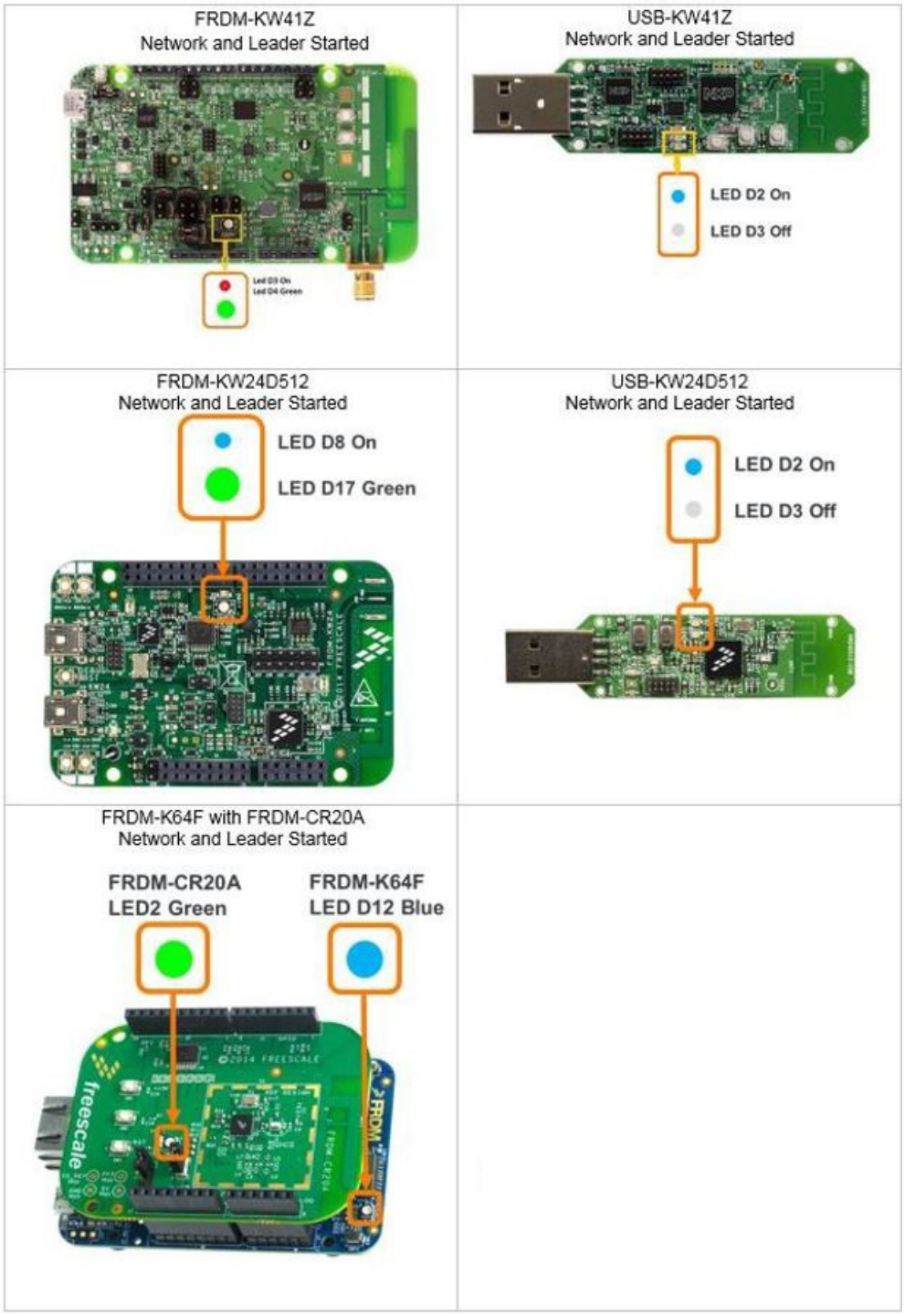


Figure 25. Factory Default State

Once devices create or join a Thread network, they save the network configuration to Non-Volatile Memory (NVM). As a result, the devices maintain their network settings even after a power-off reset cycle.

To ensure consistency, most of the network scenarios in the sections below assume the devices are reset to their Factory Default state before starting to run each scenario.

9.2.2 Factory reset

To reset an application to Factory Default state using board switches.

- **Press and HOLD FOR OVER 8 SECONDS any development board button switch** (with the exception of the Reset/RST switch)

Note

For low-power end devices, before holding any development board button switch for eight seconds, click any button once to wake the device up.

- After the button hold is released, the current Non-Volatile Memory settings are erased and the microcontroller powers-down/powers-up the rest of the system
- Board LEDs blink to show Factory Default state as indicated above

9.3 Overview

9.3.1 Overview

This scenario shows how to create (also known as: start, form, bootstrap) a new Thread network using a board button switch action when a device is in Factory Default mode and has the default firmware settings.

9.3.2 Board and application configuration

Table 4. Board and application configuration

Nodes Needed: at least 1	
Node	Application
Node A	Thread Router Eligible Device

9.3.3 Running the scenario

9.3.3.1 Initiating network creation

To **create (start, form, bootstrap) a new Thread network** with a user-initiated button switch action:

- **Short press any board button switch 2 (two) times** – any button switch initiates the action to create the new network on the double press except for the Reset/RST switch
- The 1st button press initiates network join attempts based on active network discovery requests; the 2nd button press indicates that the application stops the attempts to join a new network with the next opportunity (when the stack indicates a join failure process) and create a new network instead
- Verify the device has started as a network **Leader** role as indicated below

Note: with default settings, the Thread Router Eligible Device demo also auto-starts an initial network **Commissioner** application module once it starts-up as the initial leader. By default the active Commissioner allows joining of any devices which have their **device passphrase (PSKd)** set to value: ***THREAD***.

9.3.3.2 Verifying network creation and leader role

Once the network is created as shown above, the **Router Eligible Device** also becomes an **Active Router** to accept new children nodes to join. At the same time it also self-promotes as a network partition **Leader**.

To verify **the node has created a new network and started successfully as a Leader**, note the state of the board LEDs as shown below:

- **FRDM-KW41Z**
  - LED D3 solid red indicates Active Router role
  - RGB LED D4 solid green indicates Leader role
- **USB-KW41Z**
  - LED D2 solid red indicates only Active Router role
- **FRDM-KW24D512**
  - LED D8 solid blue indicates Active Router role
  - RGB LED D17 solid green indicates Leader role
- **USB-KW24D512**
  - LED D2 solid blue indicates only Active Router role
- **FRDM-K64F with FRDM-MCR20A**
  - FRDM-K64F Motherboard RGB LED D12 solid blue indicates Router role
  - FRDM-MCR20A RGB LED D2 solid green indicates Leader role

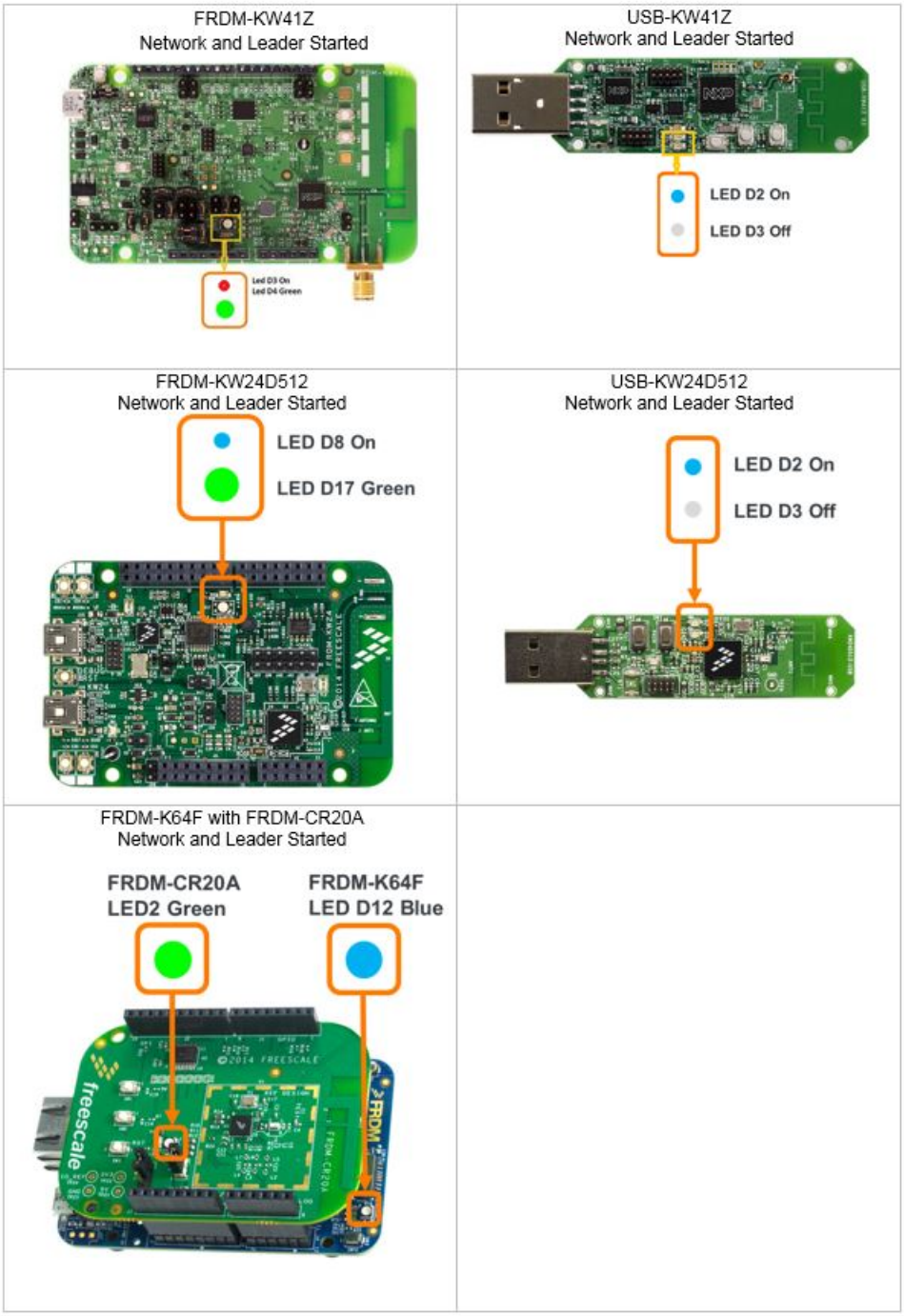


Figure 26. Network Leader Started

9.4 Joining a router eligible device to an existing network

9.4.1 Overview

This scenario shows how to join a Router Eligible Device in Factory Default state to an existing Thread mesh network using a board button switch action.

9.4.2 Board and application configuration

Table 5. Board and application configuration

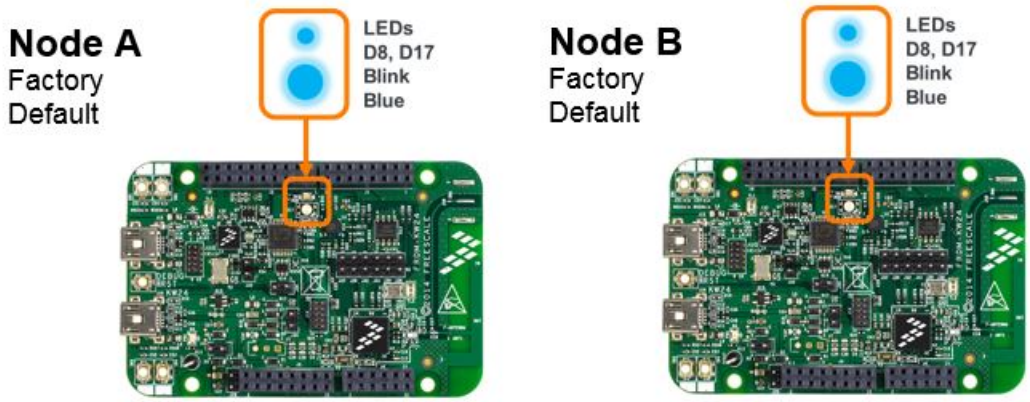
Nodes Needed: at least 2	
Node	Application
Node A	Thread Router Eligible Device
Node B	Thread Router Eligible Device

9.4.3 Steps to create a new network and join a new device

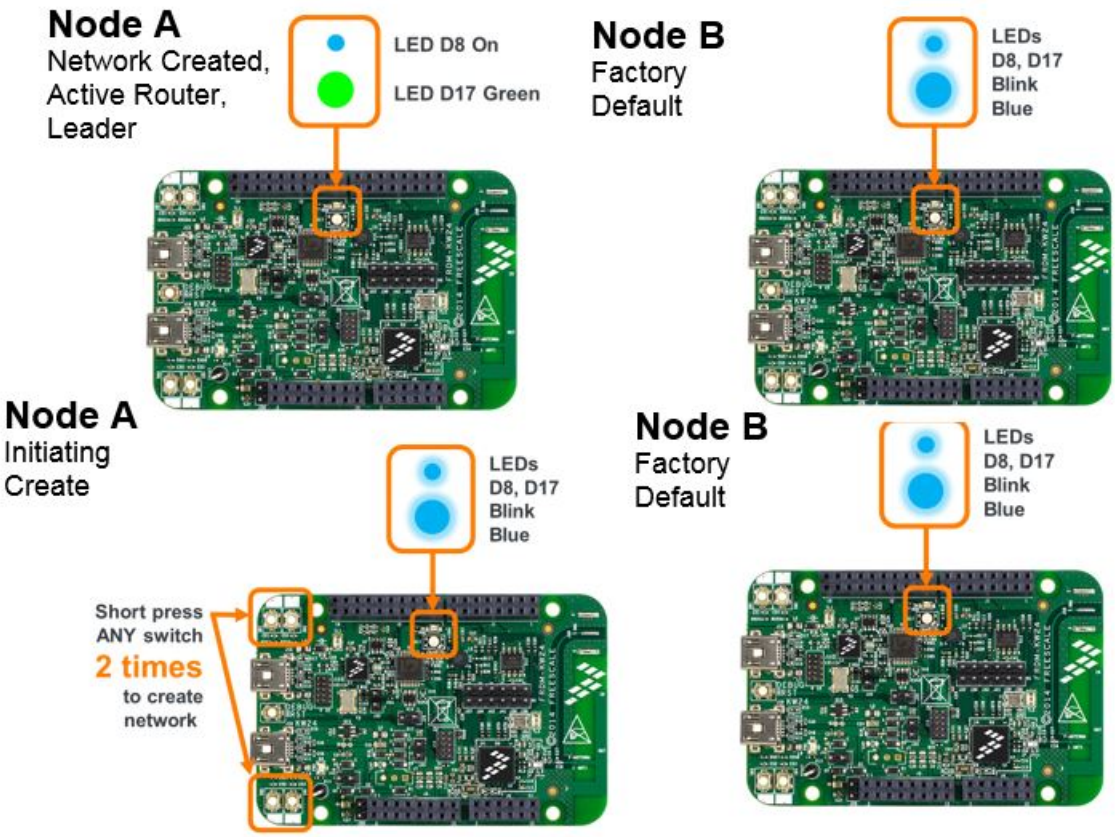
Note: the images used in the detailed steps assume using a FRDM-KW24D512 platform for Node A and Node B.

To **join a new Thread network** with a user initiated button switch action:

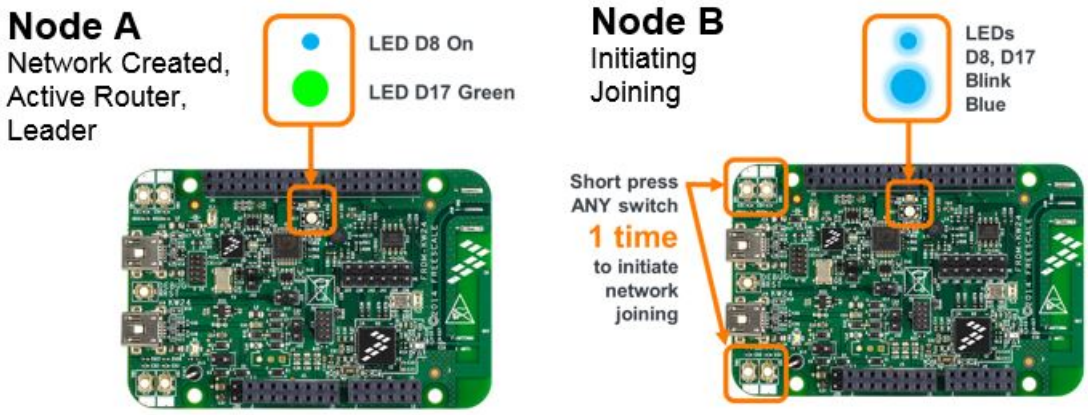
- Generally before joining the device, users must ensure there is an existing Thread network in range having with the following pre-conditions:
  - There is at least 1 (one) **Active Router** in the network allowing new devices to join
  - There is an active **Commissioner** for the network permitting joining from devices provisioned with device password (PSKd) ***THREAD*** and any EUI64 device address
- **Creating a new network usingNode A** and having the default settings of the Router Eligible Device application ensures the conditions above are met: Node A fulfills both the **Active Router** and **Commissioner** roles
- On **Node A** and **Node B**: ensure nodes are in **Factory Default** state (LEDs blink)



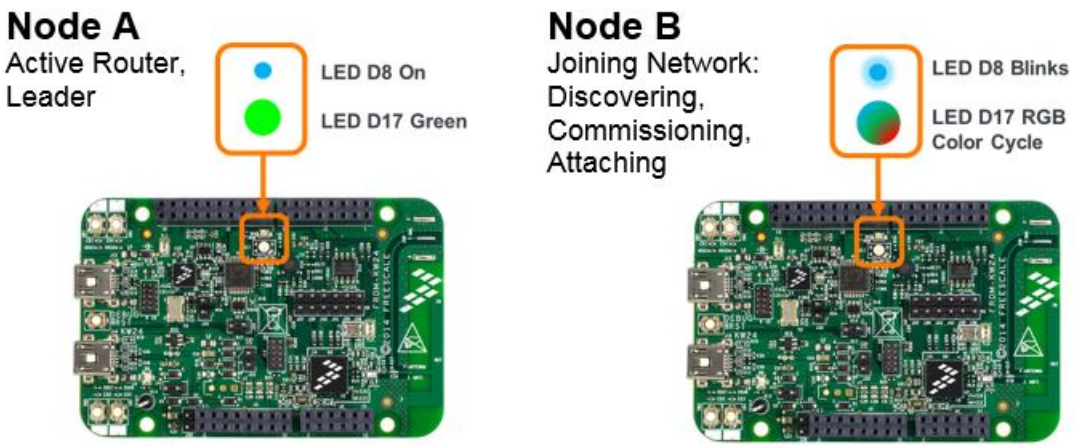
- On **Node A**: as shown in section **Steps to Create a New Thread Network** short press any board button switch **2 (two) times**
- On **Node A**: verify network creation succeeds as shown in section **Verifying Network Creation and Leader Role**



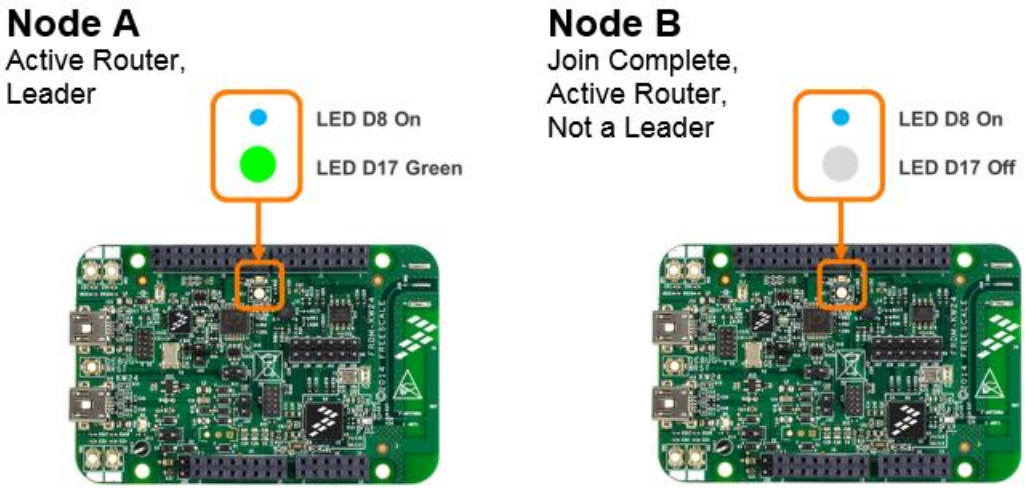
- On **Node B**: **Short press any board button switch 1 (one) time** – any button switch initiates the action to discover and join an existing network



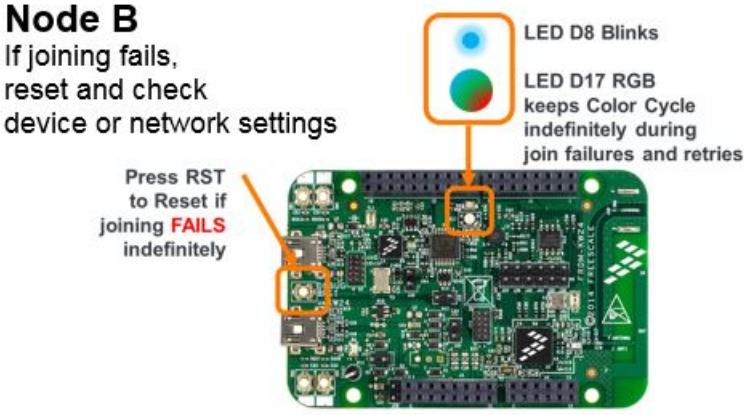
- On **Node B**: as the user initiates joining, the board LED state cycles to display ongoing progress for several seconds as Node B discovers networks in range, negotiates security with the Commissioner and the device attaches to the network after having received its credentials after successful commissioning



- On **Node B**: after a few seconds, if attaching to the network succeeds, Node B starts and requests a Router ID from the network Leader (Node A). If this succeeds and Node B becomes an Active Router in the same network as Node B, its LED state stabilizes



- On **Node B**: if the initial discovery and joining attempt to an existing network does NOT succeed, the node retries joining indefinitely (shown by LED cycle) until a network becomes open for joining or the user resets the node



9.4.4 Joining an end device or low-power end device to an existing network

9.4.4.1 Overview

This scenario shows how to join a Device or a Low-power End Device in Factory Default state to an existing Thread mesh network using a board button switch action.

9.4.4.2 Board and application configuration

Table 6. Board and application configuration

Nodes Needed: at least 2	
Node	Application
Node A	Thread Router Eligible Device
Node B	Thread End Device or Thread Low-power End Device

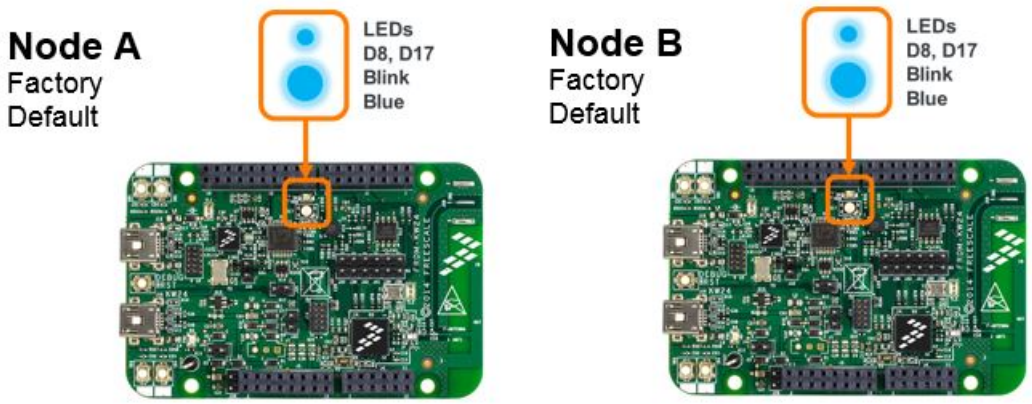
9.4.4.3 Running the scenario

Note: the images used in the detailed steps assume using a FRDM-KW24D512 platform for Node A and Node B.

To **join a Thread network** with a user initiated button switch action:

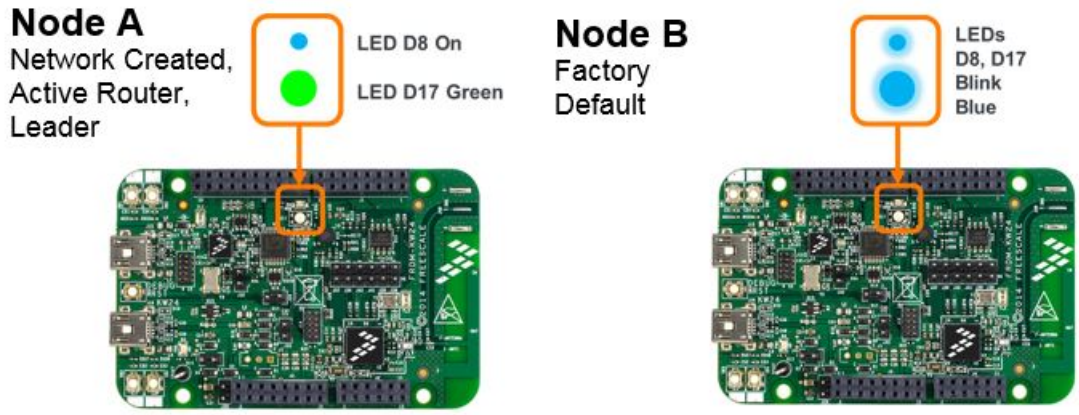
- Generally before joining the device, users must ensure there is an existing Thread network in range having with the following pre-conditions:
  - There is at least 1 (one) **Active Router** in the network allowing new devices to join
  - There is an active **Commissioner** for the network permitting joining from devices provisioned with device password (PSKd) ***THREAD*** and any EUI64 device address
- **Creating a new network using Node A** and having the default settings of the Router Eligible Device application ensures the conditions above are met: Node A fulfills both the **Active Router** and **Commissioner** roles

On **Node A** and **Node B**: ensure nodes are in **Factory Default** state (LEDs blink)

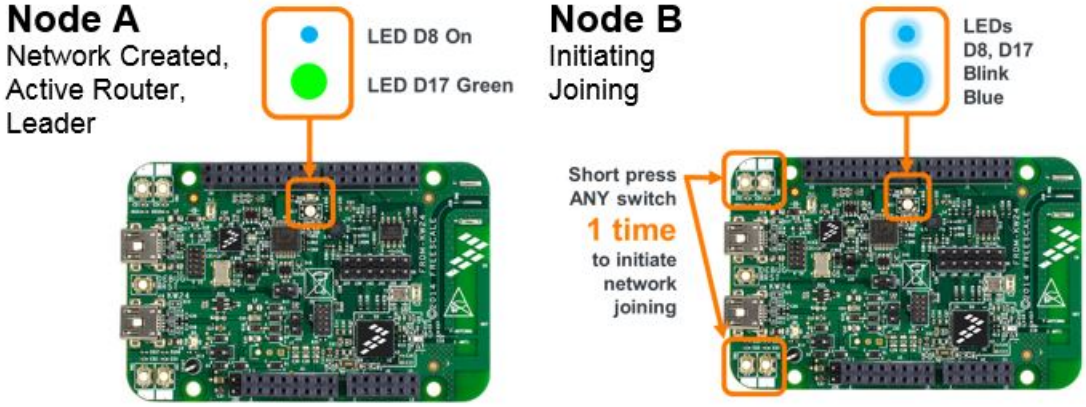


On **Node A**: as shown in section **Steps to Create a New Thread Network** short press any board button switch **2 (two)times**

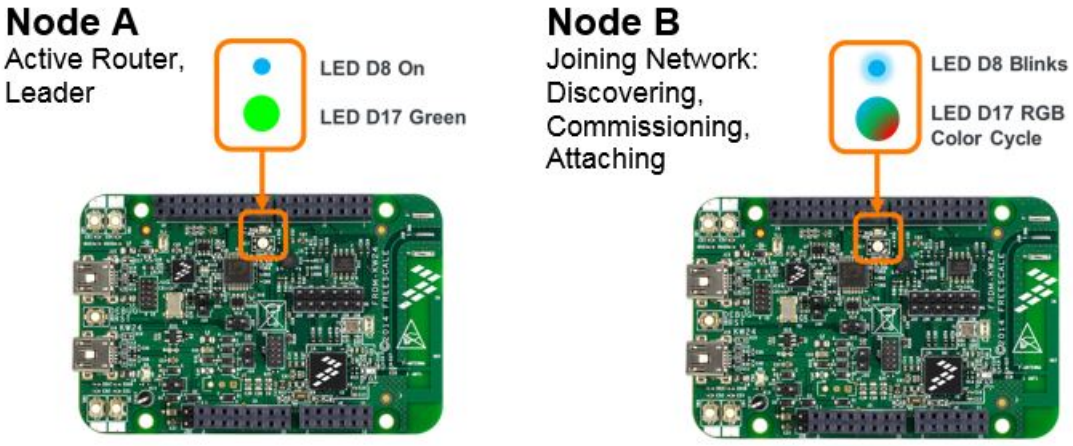
- On **Node A**: verify network creation succeeds as shown in section **Verifying Network Creation and Leader Role**



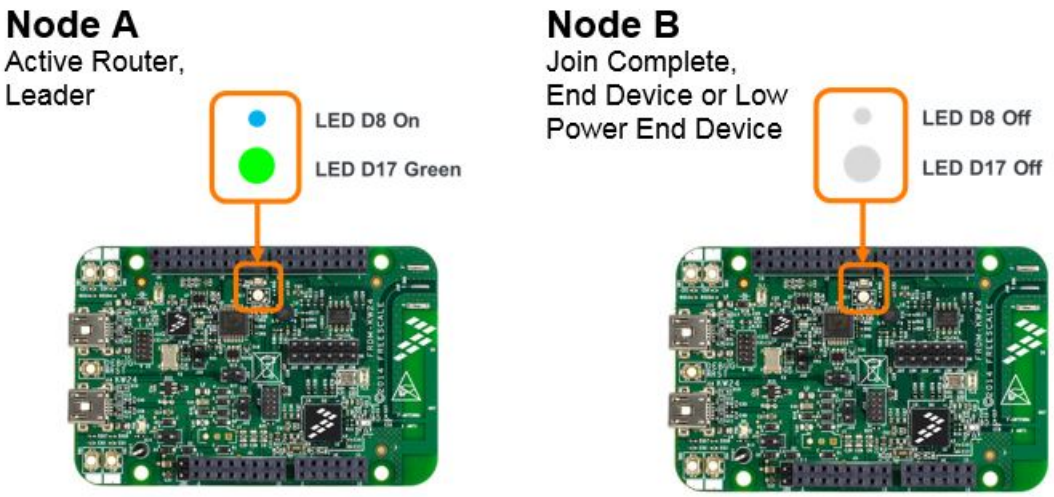
- On **Node B**: **Short press any board button switch 1 (one) time** – any button switch initiates the action to discover and join an existing network



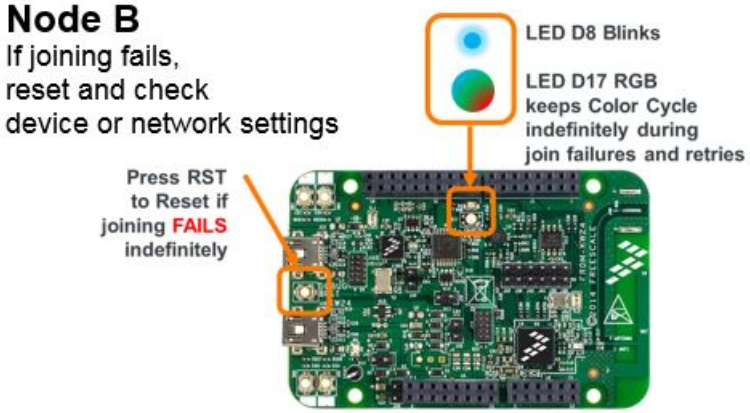
On **Node B**: as the user initiates joining, the board LED state cycles to display ongoing progress for several seconds as Node B discovers networks in range, negotiates security with the Commissioner and the device attaches to the network after having received its credentials after successful commissioning



- On **Node B**: after a few seconds, if attaching to the network succeeds, LEDs turn off.



- On **Node B**: if the initial discovery and joining attempt to an existing network does NOT succeed, the node retries joining indefinitely (shown by LED cycle) until a network becomes open for joining or the user resets the node



9.5 Sending multicast LED control CoAP messages

9.5.1 Overview

This scenario shows how to use board button switch actions to send an **LED ON** or **LED OFF** command to the other devices on the Thread network and notice the LED control taking effect.

The LED control command is:

- encoded using a **CoAP frame format** at the application layer
- carried over **UDP** at the transport layer
- multicast using the **<All Thread Nodes>** realm-local multicast destination address at the IPv6 layer indicating it should be passed on to all nodes
- passed through via **6LoWPAN** and **MPL** by the upper IPv6 protocol stack to and from the link layer
- broadcast over-the-air at the **link layer** for all non-Low-power End Device destinations

- unicast over-the-air at the **link layer** on a MAC Data Request poll sequence for Low-power End Device destinations by the Routers which are respective parents of the End Devices

9.5.2 Board and application configuration

Table 7. Board and application configuration

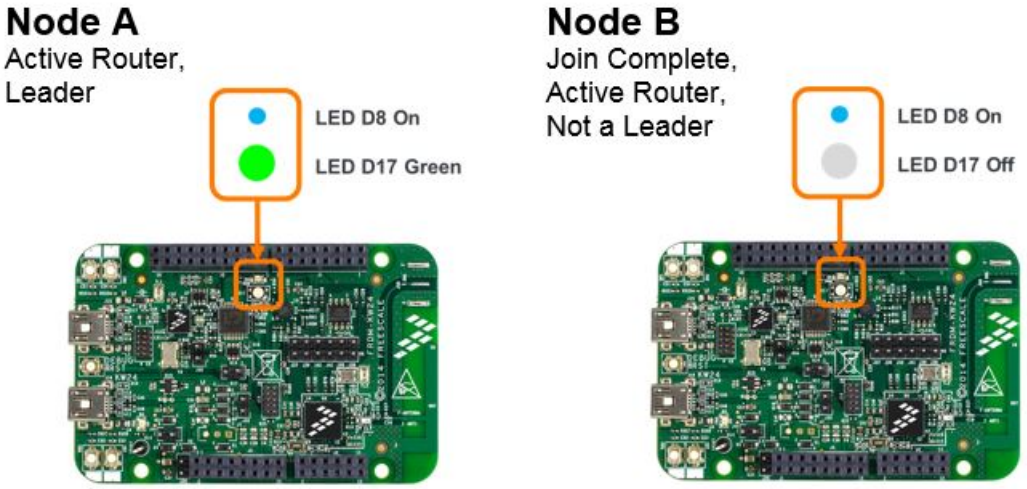
Nodes Needed: at least 2	
Node	Application
Node A	Thread Router Eligible Device
Node B	Thread Router Eligible Device

9.5.3 Steps to send multicast LED control CoAP messages

Note: the images used in the detailed steps assume using FRDM-KW24D512 platform for Node A and Node B. Tables at the end of the section show images of initiating the multicast action also for the other supported boards.

To **send a multicast LED control CoAP message** using a board button switch action:

- LED control works only if devices are already joined to the network. All following steps assume the steps of joining Node B to the network created by Node A have been performed as shown in section **Steps to Create a Network and Joining a New Device.** Node A and Node B should be Active Routers.



- On **Node B: press button switch for LED OFF** (SW4 for FRDM-KW24D512) to ensure the specific LED used to show control actions turns off
- On **Node A**: note that an LED turns off if it was on (the RGB LED D17 turns off on FRDM-KW24D512)
- On **Node B**: **press button switch for LED ON** (SW3 for FRDM-KW24D512)
- On **Node A**: note LED turns on (RGB LED D17 turns on to a random RGB hue on FRDM-KW24D512)
- On **Node B**: **press switch button for LED OFF again** (SW4 for FRDM-KW24D512)
- On **Node A**: note LED turns back off
- On **Node A**: repeat the same steps above using board switches for LED OFF and LED ON and note the effects in the other direction taking place on Node B

- The sender of the control messages generates a different RGB LED hue included when it sends each LED ON message. As a result, nodes receiving the message with RGB LEDs on-board change their LED hue each time the LED ON action is exercised
- If 3 or more boards are used, note the effect of the LED control takes place on all the devices on the network indicating multicast messaging is being used

9.6 Announcing a data sink and sending unicast LED control CoAP messages

This scenario shows how to use board button switch actions to:

- set a node to announce itself as an application **Data Sink –** representing a single concentrator destination node on the network for application messages
- set a node to announce releasing (ceasing) the Data Sink role
- send unicast **LED ON** or **LED OFF** CoAP confirmable commands addressed to the Data Sink node exclusively and notice the LED control taking effect

The commands to **announce setting a Data sink** and **announce releasing a data sink** are sent as multicast CoAP messages. Once a Data Sink is set, LED commands sent to it are sent unicast, on an optimal route and are confirmable, being retransmitted if not acknowledged by the Data Sink node.

9.6.1 Board and application configuration

Table 8. Board and application configuration

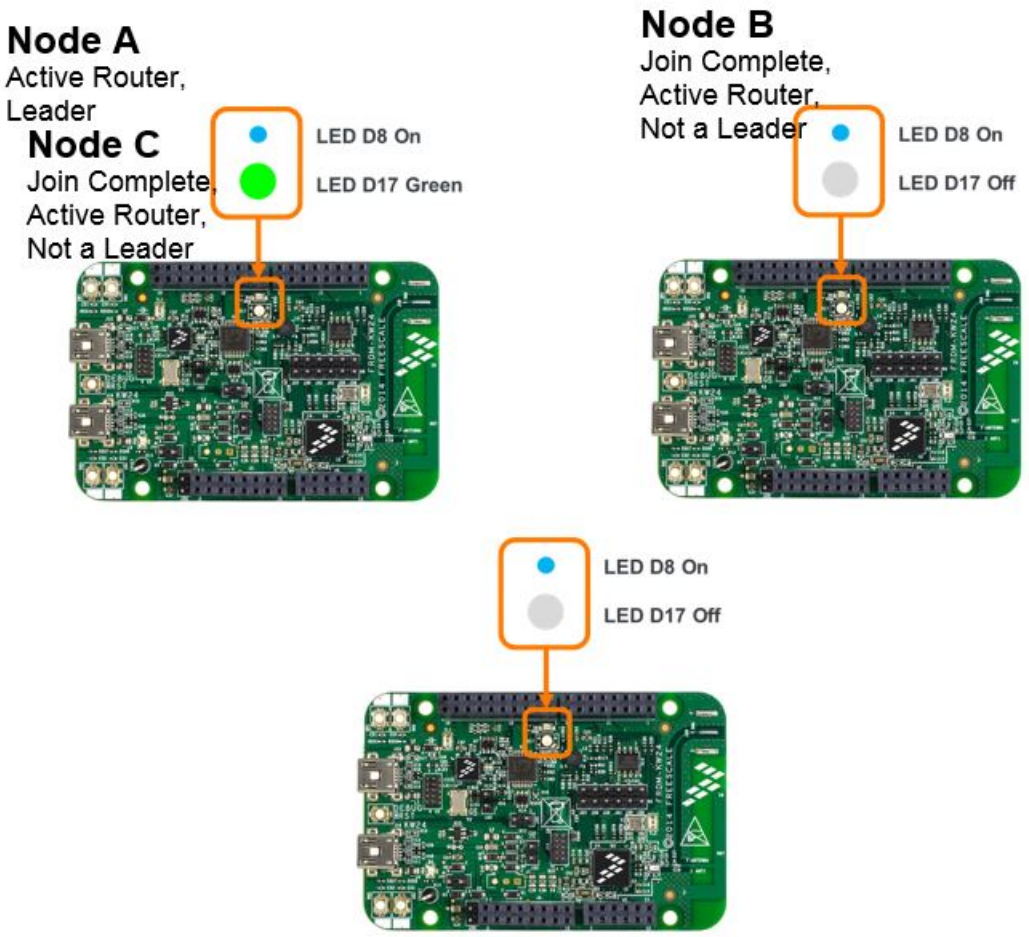
Nodes Needed: at least 3	
Node	Application
Node A	Thread Router Eligible Device
Node B	Thread Router Eligible Device
Node C	Thread Router Eligible Device

9.6.2 Steps to announce and send unicast messages to a data sink

Note: the images used in the detailed steps assume using FRDM-KW24D512 platform for Node A and Node B.

To **announce a node as a Data Sink** using a board button switch action take the following steps:

- Data sink announcements work only if devices are already joined to the network. All following steps assume the steps of joining Node B and Node C to the network created by Node A have been performed as shown in section **Steps to Create a Network and Joining a New Device.** Node A, Node B, and Node C should be Active Routers.



- Check that a data sink is not active and LED control messages are multicast:
- On **Node A: press button switch for LED ON** (SW3 for FRDM-KW24D512)
- On **Node B and C**: note that the LED turns on (the RGB LED D17 turns on FRDM-KW24D512)
- On **Node B: Short press button switch to Create Data Sink** (SW1 for FRDM-KW24D512)
- On **Node A: press button switch for LED OFF** (SW4 for FRDM-KW24D512)
- On **Node B**: note LED has gone off

On **Node C**: note LED has NOT gone off

A Data Sink has been created with Node B as a concentrator. Node C and Node A no longer receive LED control messages from other devices which are now unicast to Node B

- On **Node C**: check that the LED ON and LED OFF buttons only control Node B LED, but not Node A
- On **Node B**: **Long press (press and HOLD 2-3 seconds) switch button to Release Data Sink** (SW1 for FRDM-KW24D512)
- On **Node A** and **Node C:** check that the LED ON and LED OFF buttons now have reverted back to multicast behavior controlling all other nodes.

9.7 Network partitioning and merging

9.7.1 Overview

This scenario shows how to visualize the Thread network states of **partition creation** and **partition merging** using the board LEDs.

A Thread routing segment is a group of Routers and their associated children End Devices which can reach each other on direct radio links or via a multihop path (through messages forwarded through a set of other routers in the segment).

When two or more routing segments of a Thread network become disconnected - for instance due to nodes being moved out of range or intermediary routers being turned off - then each routing segment creates a distinct network partition having its own Leader node.

When Routers detect loss of connectivity to the current Leader node due to that being in a different network partition, they advertise their state and a new Router in the current routing segment emerges as a Leader of the new network partition which got disconnected from the segment containing the previous Leader.

In a worst case scenario, even a single Router can be part of a distinct network partition.

When Routers in distinct partitions come back into connectivity range, their respective partitions merge back into a common partition having a single Leader.

9.7.2 Board and application configuration

Table 9. Board and application configuration

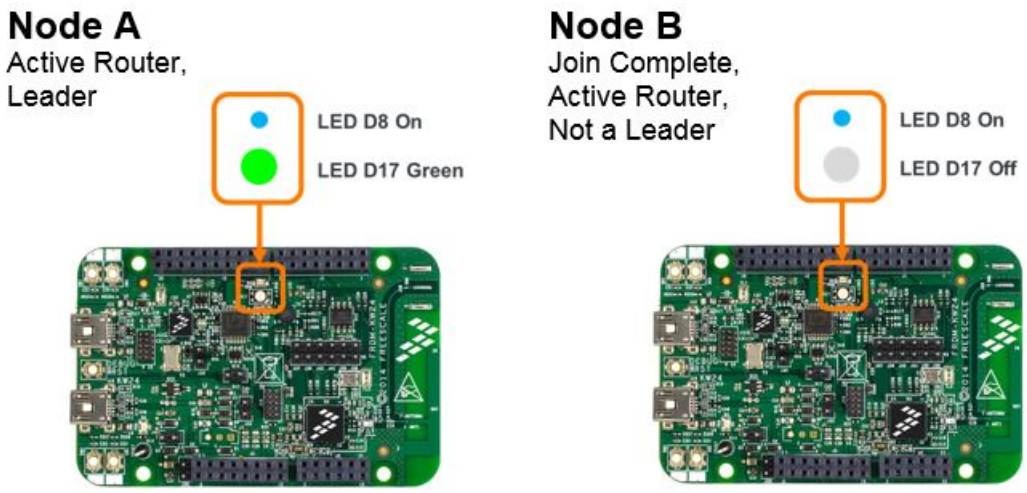
Nodes Needed: at least 2	
Node	Application
Node A	Thread Router Eligible Device
Node B	Thread Router Eligible Device

9.7.3 Using partitioning and merging

Note: the images used in the detailed steps assume using FRDM-KW24D512 platform for Node A and Node B. Tables at the end of the section show images of initiating the multicast action also for the other supported boards.

To **create a partitioning situation**:

- All following steps assume the initial joining of Node B to the network created by Node A has been performed as shown in section **Steps to Create a Network and Joining a New Device.** Node A and Node B should be Active Routers and in connectivity range.



- Create a loss of connectivity situation so that Node B is no longer in radio range with Node A. To achieve that either:
  - **Temporarily power off Node A**
  - **Take one of the nodes physically out of range** from the other – for instance by moving the nodes to different floors for a multistoried building or increasing the physical distance between the nodes significantly (100 feet or longer)
- Note that if **Node B** is not power-off reset, then in 1-2 minutes after losing connectivity, it indicates via its LED it has become a new partition Leader (on FRDM-KW24D512 the RGB LED D17 turns green).

This indicates Node B is part of a distinct network partition.

- If a **power-off reset takes place onNode B** as the nodes are out of range, the partitioning happens more quickly as Node B starts back up as an Active Router without detecting other neighbor Routers.
- **Bring the 2 nodes back into range** (for example, power Node A back on)
- Note the network reverts back to having one Leader, indicating the **distinct partitions have merged**. Note that Node B can also emerge as the single Leader when the partitions are merged back.

Chapter 10
Running Thread Network Scenarios Using the Shell Interface

10.1 Board setup and provisioning for shell usage

The following Kinetis Thread Stack examples are provisioned by default with a shell command line interface accessible via a Terminal application such as PuTTY:

- Thread Router Eligible Device
- Thread End Device
- Thread Border Router

10.2 Shell provisioning in Windows® OS

To connect to a board shell in Windows OS:

- Plug an USB cable attached to a host PC into each device and power on the board.
- Expand section **Ports (COM&LPT)** and check if a COMxx port entry appears as show in the figure below.

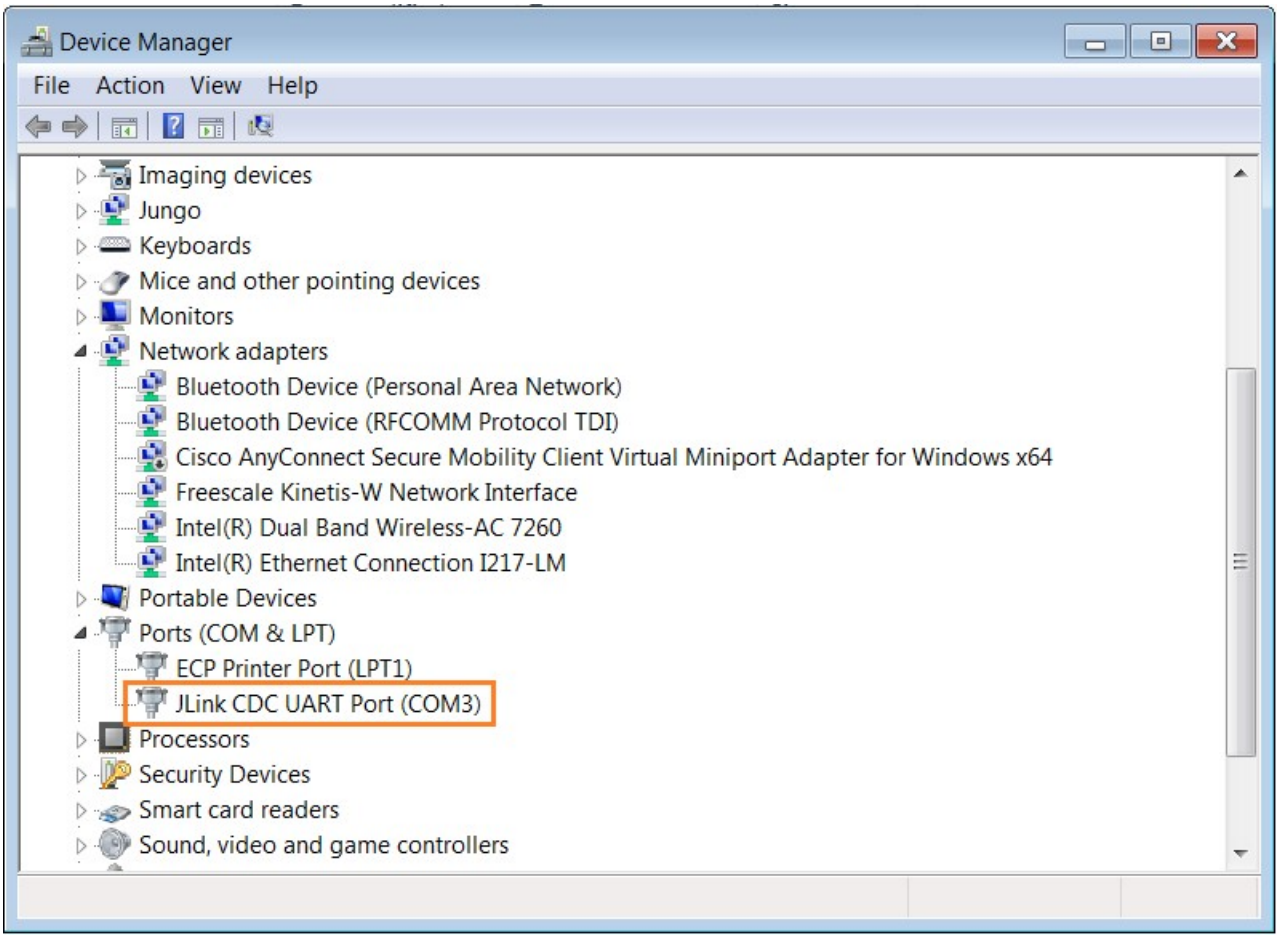


Figure 27. Using Device Manager to determine COM port numbers of USB connections

- To open a shell terminal from the PuTTY application, use the COM port number identified above and a 115200 bps baud rate be used in the port settings

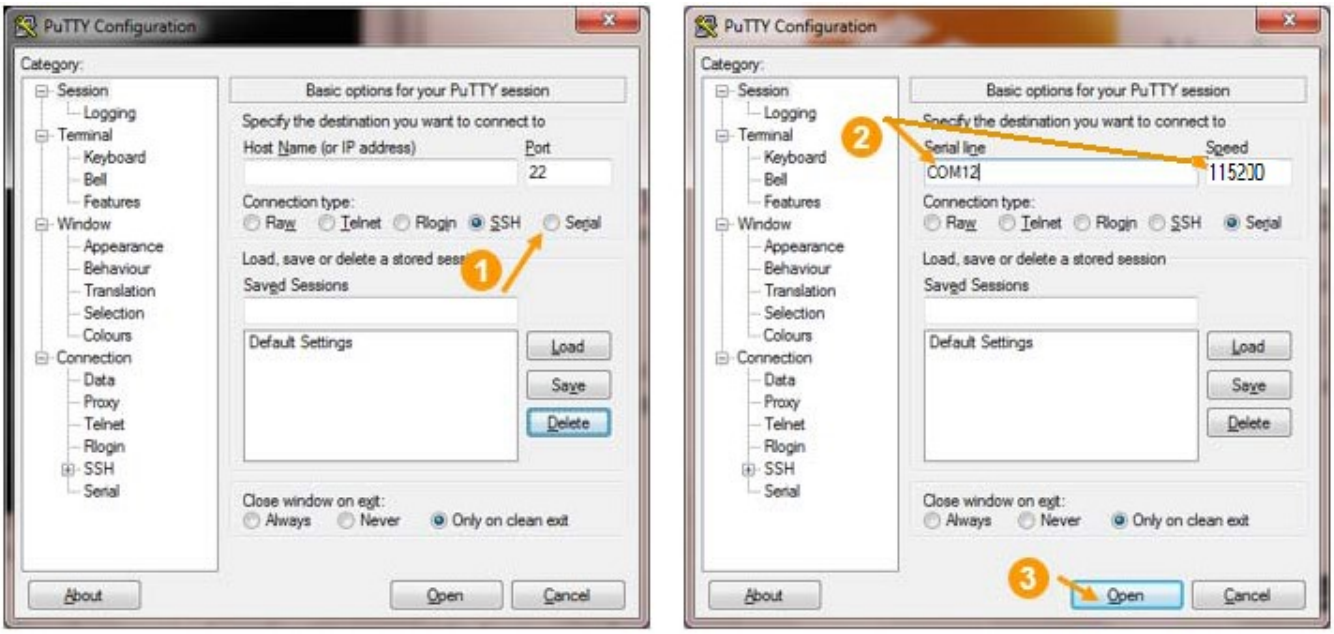


Figure 28. Starting PuTTY

10.3 Shell provisioning in MAC® OS X

To connect to a board shell interface in MAC OS X:

- An USB modem driver should automatically be loaded by MAC OS X when plugging in the board.
- To check the board is recognized by the operating system lunch Spotlight and type: **Terminal**
- Select and launch the terminal application shown in the Spotlight results
- At the terminal prompt type:

ls /dev/tty.*

- A /dev file system entry such as below is shown for the board:

/dev/tty.usbmodem0000001

- The COM port can be accessed via a terminal application such as **screen** with the 115200bps open port baud rate

10.4 Creating a new Thread network and commissioning a device

10.4.1 Overview

This scenario shows how to Factory Reset a device, proceed to create a new Thread network using the shell interface, and join a new device to the network by commissioning it based on a customized device pass phrase (PSKd).

10.4.2 Board and application configuration

Table 10. Board and application configuration

Nodes Needed: at least 2	
Node	**Application**
Node A	Thread Router Eligible Device
Node B	Thread Router Eligible Device

10.4.3 Running the scenario

- On both nodes, factory reset the devices device using the shell by entering:

$ factoryreset

Note: when using the KW2xD USB connection, such as USB-KW24D512, the device needs to be re-inserted and re-opened in PuTTY or the terminal application.

- On **Node A** shell, to create a new Thread network enter:

$ thr create

Note the status messages in the shell indicating the network parameters. Note a Local Commissioner instance also starts on Node A

- On **Node B** shell, set a unique device passphrase (PSKd), for instance:

$ thr set pskd A1B2C3

- On **Node A** shell, replace the “catch-all” device password (accepted for any EUI64 address to the value above)

$ thr joiner add A1B2C3

- On Node B shell initiate joining:

$ thr join

Note the status messages in the shell indicating the network parameters and that the joiner has been accepted.

10.5 Steering and commissioning multiple devices

10.5.1 Overview

This scenario shows how to commission a set of devices based on their EUI64 addressees and a customized device pass phrase (PSKd) in a single steering session.

10.5.2 Board and application configuration

Table 11. Board and application configuration

Nodes Needed: at least 3	
Node	**Application**
Node A	Thread Router Eligible Device
Node B	Thread Router Eligible Device
Node C	Thread Router Eligible Device

10.5.3 Running the scenario

- On all nodes, factory reset the devices device using the shell by entering:

$ factoryreset

- On **Node A** shell, to create a new Thread network enter:

$ thr create

Note the status messages in the shell indicating the network parameters. Note a Local Commissioner instance also starts on Node A

- On **Node A** shell, check the current joiner table:

$thr joiner view

Index EUI PSK

0 0xFFFFFFFFFFFFFFFF THREAD

Note the “catch-all” device password (accepting devices for any EUI64 address if using the PSK value above)

- On **Node A** shell, clear the catch-all entry:

$thr joiner removeall

- On **Node B** shell, set a unique device passphrase (PSKd), for instance:

$ thr set pskd NodeB

- On **Node B** shell, get the device EUI64 (note actual value will be different than listed below):

$ thr get eui

eui: 0x00049F0E45810021

- On **Node C** shell, set a unique device passphrase (PSKd), for instance:

$ thr set pskd NodeC

- On **Node C** shell, get the device EUI64 (note actual value will be different than listed below):

$ thr get eui

eui: 0x00049F0EFD610030

- On Node A shell, add Node B as joiner expected specifically:

$ thr joiner add NodeB 0x00049F0E45810021

- On Node A shell, add Node C as joiner expected specifically:

$ thr joiner add NodeC 0x00049F0EFD610030

- On Node A, check the expected joiner table:

$ thr joiner view

Index EUI PSK

1. **0x00049F0E45810021** NodeB
2. **0x00049F0EFD610030** NodeC
3. On Node A, sync the steering data from the Commissioner layer to the network:

$ thr sync steering

- On Node B shell initiate joining:

$ thr join

Note the status messages in the shell indicating the device is commissioning and joining the network.

- On Node C shell initiate joining:

$ thr join

Note the status messages in the shell indicating the device is commissioning and joining the network.

10.6 Inspecting IP address assignment and testing connectivity

10.6.1 Overview

This scenario shows how to inspect the IP address assigned to the Thread interface and use the ICMP Ping to test basic connectivity.

10.6.2 Board and application configuration requirements

Table 12. Board and application configuration requirements

Nodes Needed: at least 3	
Node	Application
Node A	Thread Router Eligible Device
Node B	Thread End Device
Node C	Thread Router Eligible Device

10.6.3 Running the scenario

- On **All Nodes**, factory reset the devices device using the shell by entering:

$ factoryreset

Note: when using the USB-KW24D512, the device needs to be re-inserted and re-opened in PuTTY or the terminal application.

- On **Node A** shell, to create a new Thread network enter:

$ thr create

Note the status messages in the shell indicating the network parameters. Note a Commissioner instance also starts on Node A

- On **Node B** shell initiate joining with the default PSKd (***THREAD***):

$ thr join

Note the status messages in the shell indicating the joining results.

- On **Node C** shell initiate joining with the default PSKd (***THREAD***):

$ thr join

Note the status messages in the shell indicating the joining results.

- On **Node A**, enter the ifconfig command to view IP address information**:**

$ ifconfig

Interface 1: 6LoWPAN

Link local address (LL64): fe80::7975:973a:a646:ea77

Mesh local address (ML64): fd84:e3f6:590c::81c9:6968:41a6:70c0

Mesh local address (ML16): fd84:e3f6:590c::ff:fe00:0

Link local all Thread Nodes(MCast): ff32:40:fd84:e3f6:590c::01

Realm local all Thread Nodes(MCast): ff33:40:fd84:e3f6:590c::01

Interface 0: Loopback

- On **Node B**, enter the ifconfig command to view IP address information**:**

$ ifconfig

Interface 1: 6LoWPAN

Link local address (LL64): fe80::14f:bb07:baba:3683

Mesh local address (ML64): fd84:e3f6:590c::6548:1d09:a9b:177e

Mesh local address (ML16): fd84:e3f6:590c::ff:fe00:01

Link local all Thread Nodes(MCast): ff32:40:fd84:e3f6:590c::01

Realm local all Thread Nodes(MCast): ff33:40:fd84:e3f6:590c::01

Interface 0: Loopback

- On **Node C**, enter the ifconfig command to view IP address information**:**

$ ifconfig

Interface 1: 6LoWPAN

Link local address (LL64): fe80::e583:30c2:8837:727b

Mesh local address (ML64): fd84:e3f6:590c::2185:81f2:1c61:ac4

Mesh local address (ML16): fd84:e3f6:590c::ff:fe00:400

Link local all Thread Nodes(MCast): ff32:40:fd84:e3f6:590c::01

Realm local all Thread Nodes(MCast): ff33:40:fd84:e3f6:590c::01

Interface 0: Loopback

- Using the mesh local ML64 (ML-EID) addresses of the destination noted via ifconfig command to ping the other nodes over the Thread network.
- For instance, **on Node A**, to ping Node B:

$ ping fd84:e3f6:590c::6548:1d09:a9b:177e

Pinging fd84:e3f6:590c::6548:1d09:a9b:177e with 32 bytes of data:

Reply from fd84:e3f6:590c::6548:1d09:a9b:177e: bytes=32 time=24ms

Reply from fd84:e3f6:590c::6548:1d09:a9b:177e: bytes=32 time=29ms

Reply from fd84:e3f6:590c::6548:1d09:a9b:177e: bytes=32 time=34ms

Reply from fd84:e3f6:590c::6548:1d09:a9b:177e: bytes=32 time=27ms

Reply from fd84:e3f6:590c::6548:1d09:a9b:177e: bytes=32 time=21ms

- On **Node B**, to ping Node C:

$ ping fd84:e3f6:590c::2185:81f2:1c61:ac4

Pinging fd84:e3f6:590c::2185:81f2:1c61:ac4 with 32 bytes of data:

Reply from fd84:e3f6:590c::2185:81f2:1c61:ac4: bytes=32 time=1055ms

Reply from fd84:e3f6:590c::2185:81f2:1c61:ac4: bytes=32 time=44ms

Reply from fd84:e3f6:590c::2185:81f2:1c61:ac4: bytes=32 time=47ms

Reply from fd84:e3f6:590c::2185:81f2:1c61:ac4: bytes=32 time=50ms

Reply from fd84:e3f6:590c::2185:81f2:1c61:ac4: bytes=32 time=52ms

Note the ping times between Node B and Node C are larger as Node B and Node C are not neighbors, the packets being forwarded by Node A as router parent of End Device Node B. Also Node B must first perform address resolution for the Node C ML64 address (large ping time for first packet).

10.7 Sending application data CoAP messages using the shell

10.7.1 Overview

This scenario shows how to send and view the receive indication for CoAP messages using the shell interface.

10.7.2 Board and application configuration

Table 13. Board and application configuration

Nodes Needed: at least 2	
Node	Application
Node A	Thread Router Eligible Device
Node B	Thread Router Eligible Device

10.7.3 Running the scenario

- On All Nodes, factory reset the devices device using the shell by entering:

$ factoryreset

Note: when using the USB-KW24D512, the device needs to be re-inserted and re-opened in PuTTY or the terminal application.

- On **Node A** shell, to create a new Thread network enter:

$ thr create

Note the status messages in the shell indicating the network parameters. Note a Commissioner instance also starts on Node A

- On **Node B** shell initiate joining with the default PSKd (***THREAD***):

$ thr join

Note the status messages in the shell indicating the joining results.

- On **Node A**, enter the ifconfig command to view IP address information**:**

$ ifconfig

Interface 1: 6LoWPAN

Link local address (LL64): fe80::54a0:abf9:c49:8520

Mesh local address (ML64): fd16:e46c:f0e7::f846:14ef:a586:d5d2

Mesh local address (ML16): fd16:e46c:f0e7::ff:fe00:0

Link local all Thread Nodes(MCast): ff32:40:fd16:e46c:f0e7::01

Realm local all Thread Nodes(MCast): ff33:40:fd16:e46c:f0e7::01

Interface 0: Loopback

- On **Node B**, enter the ifconfig command to view IP address information**:**

$ ifconfig

Interface 1: 6LoWPAN

Link local address (LL64): fe80::bc85:e92e:fa9e:a10d

Mesh local address (ML64): fd16:e46c:f0e7::f438:9c8b:c09f:49f0

Mesh local address (ML16): fd16:e46c:f0e7::ff:fe00:400

Link local all Thread Nodes(MCast): ff32:40:fd16:e46c:f0e7::01

Realm local all Thread Nodes(MCast): ff33:40:fd16:e46c:f0e7::01

Interface 0: Loopback

- Use the addresses noted via ifconfig command to send CoAP LED control messages.

For instance on **Node A** enter the command to send to Node B an RGB LED command (each r, g, b flags take values from 000 to 255):

$ coap CON POST fd16:e46c:f0e7::f438:9c8b:c09f:49f0 /led rgb r255 g000 b255

coap rsp from fd16:e46c:f0e7::f438:9c8b:c09f:49f0 ACK

- If Node B has an RGB LED, it turns purple because of the red and blue settings being maximized.
- On **Node A** to send to Node B an RGB LED off command enter:

$ coap CON POST fd16:e46c:f0e7::f438:9c8b:c09f:49f0 /led off

coap rsp from fd16:e46c:f0e7::f438:9c8b:c09f:49f0 ACK

- On **Node B** press “Report Temperature” button switch on Node B (SW2 on FRDM-KW24D512)
- On **Node A** note how the indication of the temperature (in degrees Celsius) is displayed in the shell as coming from the IP address of Node B:

Temp:27.37 From IPv6 Address: fd16:e46c:f0e7::f438:9c8b:c09f:49f0

10.8 Viewing routing and neighbor tables

10.8.1 Overview

This scenario shows how to inspect the routing and neighbor table of a Thread device using the shell interface.

10.8.2 Board and application configuration requirements

Table 14. Board and application configuration requirements

Nodes Needed: at least 3	
Node	Application
Node A	Thread Router Eligible Device
Node B	Thread End Device
Node C	Thread Router Eligible Device

10.8.3 Running the scenario

- On All Nodes, factory reset the devices device using the shell by entering:

$ factoryreset

Note: when using the USB-KW24D512, the device needs to be re-inserted and re-opened in PuTTY or the terminal application.

- On **Node A** shell, to create a new Thread network enter:

$ thr create

Note the status messages in the shell indicating the network parameters. Note a Commissioner instance also starts on Node A

- On **Node B** shell initiate joining with the default PSKd (***THREAD***):

$ thr join

Note the status messages in the shell indicating the joining results.

- On **Node C** shell initiate joining with the default PSKd (***THREAD***):

$ thr join

Note the status messages in the shell indicating the joining results.

- On routers Node A or Node C**,** enter the **get neighbors** command to view neighbor information**:**

$ thr get neighbors

Index Extended Address ShortAddr LastTime LinkMargin Child

0 0x0A438E2BF1EEE179 0x0001 90 45 yes

1 0x86E948900E37AE0F 0x0400 20 44 no

- On routers Node A or Node C**,** enter the **get routes** command to view routing information**:**

$ thr get routes

ID Sequence: 110

Router ID Mask: C000000000000000

RouterID Short Address Next Hop Cost NOut NIn

1 0x0400 0x0400 1 3 3

- On End Device Node B**,** enter the **get parent** command to view routing information**:**

$ thr get parent

Parent short address: 0x0000

Parent extended address: 0x53BE273660D5B757

Chapter 11
Running Border Router Application Scenarios

11.1 Border routers overview

The border router demo applications and network scenarios allows users to establish and verify connectivity to Thread devices over the Thread IP subnet boundary from IP applications or devices on external networks. The following scenarios are described:

- Using the on-chip USB features of KW2xD to emulate a RNDIS “Ethernet over USB” emulated Network Interface Card for a PC running a host OS such as Windows OS. IPv6 capable applications running on the host traverses the Thread network boundary to address Thread network devices end-to end at the IP layer.
- Using the USB-KW41Z board with K22F running `rndis_bridge` application to emulate a RNDIS "Ethernet over USB", emulated a Network Interface Card for a PC running a host OS such as Windows OS. K22F and KW41Z are connected through FSCI using SPI Serial Interface. IPv6 capable applications running on the host traverses the Thread network boundary to address Thread network devices end-to-end at the IP layer. The `rndis_bridge` project can be found in `<Connectivity_Software_Installation>`, for example, `C:\NXP\SDK_2.2_MKW41Z512xxx4>: \boards\usbkw41z_k22f \wireless_examples\framework\rndis_bridge\FreeRTOS\`.
- Using the Ethernet port of the FRDM-K64F motherboard to create a direct Ethernet link to a PC running a host OS such as Windows OS. IPv6 capable applications running on the host traverses the Thread network boundary to address Thread network devices end to end at the IP layer.
- Using the Ethernet port of the FRDM-K64F motherboard to plug into an Ethernet port of a Wi-Fi Home Access Point or Router provisioned with an IPv6 and DHCPv6-PD capable firmware such as OpenWRT. IPv6 capable devices on the local area network and as configured to the Access Point from upstream IP infrastructure provides Thread network external prefix addresses.

11.2 External routing with Ethernet emulation over USB (RNDIS) on Kinetis KW2xD boards

11.2.1 Overview

This scenario shows how to test external IPv6 connectivity through the USB Ethernet Emulation (RNDIS-based) border router.

11.2.2 Board and application configuration requirements

- Thread nodes and boards needed:

Table 15. Board and application configuration requirements

Nodes Needed: at least 2	
Node	Application
Node A	Border Router on FRDM-KW24D512 and USB- KW24D512
Node B	Thread Router Eligible Device

- This scenario also requires a RNDIS and ND capable Host PC, such as one running Windows 7 OS or later.

11.2.3 Running the scenario

- For **Node A**, in order to access RNDIS connect USB interface cables to the PC on:
  - the direct-to-KW24 USB
- For **Node A**, a KINETIS RNDIS device is detected by the Host PC operating system on the KW24 USB port.
  - For **Windows OS** :
    - for driver installation, steer the New Hardware found wizard to the **nxp_rndis.inf** driver in <Kinetis Thread Installation, for example, SDK_2.2_MKW41Z512xxx4\middleware\wireless\framework_5.3.8\SerialManager \Source\USB_VirtualNic\INF\:
- after driver installation, check the device has been detected as a Network Adapter with ID “NXP USB RNDIS” in the Windows Control Panel

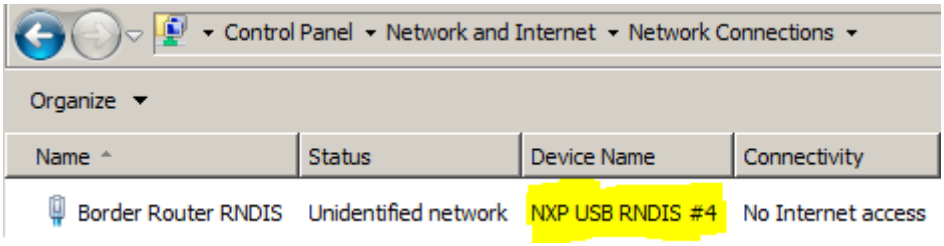


Figure 29. NXP USB RNDIS

- Double click the adapter entry, choose **Details**… and check that **FD01:** prefix IPv6 addresses have been provisioned to the adapter:

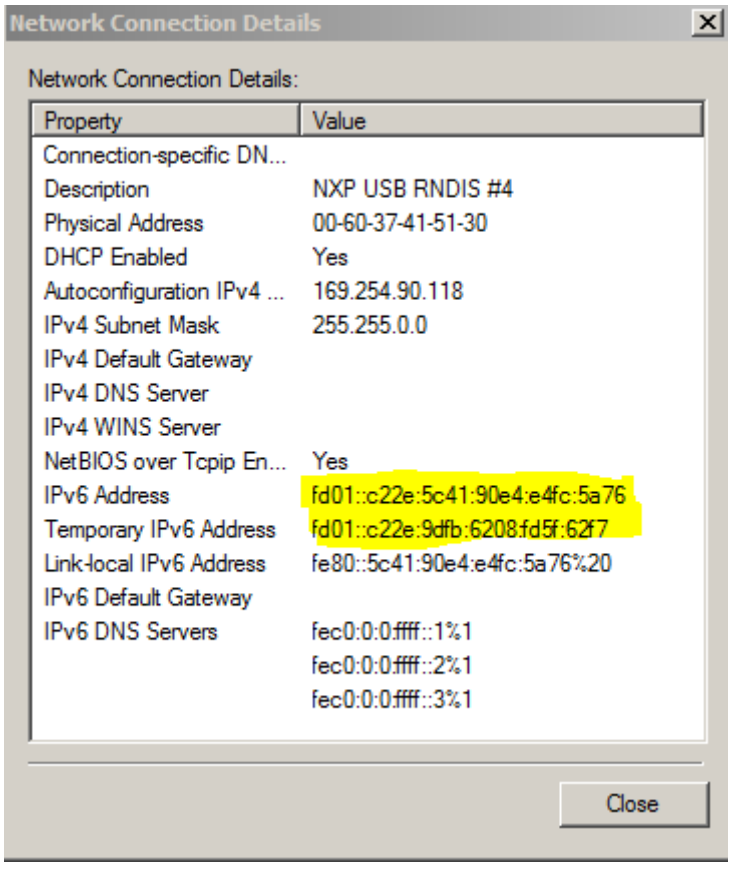


Figure 30. Adapter entry

- If FD01: addresses are not provisioned, try deselecting any 3rd party options such as Virtual Machine drivers in the interface properties
  - For most desktop Linux OS variants should already provisioned with RNDIS drivers through the **usbnet** modules.
  - For **MAC OS X** the RNDIS Host drivers at github.com/jwise/HoRNDIS may need to be deployed.
    - On **Node A**, also open a shell on the OpenSDA / OpenLink USB port, to create a new Thread network enter:

$ thr create

Note the status messages in the shell indicating the network parameters. Note a Commissioner instance also starts on Node A

- On **Node B** shell initiate joining with the default PSKd (***THREAD***):

$ thr join

Note the status messages in the shell indicating the joining results.

- On **Node B**, enter the **ifconfig** command to view IP address information**:**

$ ifconfig

Interface 1: 6LoWPAN

Link local address (LL64): fe80::1885:ba8e:d82a:6fad

Mesh local address (ML64): fd4f:12be:69d2::9c67:fec9:9306:4791

Mesh local address (ML16): fd4f:12be:69d2::ff:fe00:400

Unique local address: fd01::3ead:249f:f277:571a:3109

Link local all Thread Nodes(MCast): ff32:40:fd4f:12be:69d2::01

Realm local all Thread Nodes(MCast): ff33:40:fd4f:12be:69d2::01

Interface 0: Loopback

- On the **Host PC**, launch a terminal window (**cmd** on Windows OS) and ping the Unique local address with prefix FD01::3EAD based address of NodeB:

C:\>ping fd01::3ead:249f:f277:571a:3109

Pinging fd01::3ead:249f:f277:571a:3109 with 32 bytes of data:

Reply from fd01::3ead:249f:f277:571a:3109: time=1024ms

Reply from fd01::3ead:249f:f277:571a:3109: time=31ms

Reply from fd01::3ead:249f:f277:571a:3109: time=22ms

Note applications on the Host PC now have IP layer connectivity to Thread nodes through the border router firmware which assigns Unique Local Addresses (site-local) with prefix FD01:: to all subnets (PC interface and Thread network).

11.3 External routing via Ethernet on FRDM-K64F – ND router mode

11.3.1 Overview

This scenario shows how to test external IPv6 connectivity through the FRDM-K64F Ethernet port on the Thread border router device

11.3.2 Board and application configuration requirements

- Thread nodes and boards needed

Table 16. Board and application configuration requirements

Nodes Needed: at least 2	
Node	Application
Node A	Border Router on FRDM-K64F with FRDM-CR20A
Node B	Thread Router Eligible Device

- This scenario also requires an ND capable Host PC, such as one running Windows 7 OS or later.

11.3.3 Running the scenario

- For **Node A**, connect an Ethernet cable direct to the Ethernet port on the FRDM-K64F
- For **Node A**, check the network card is configured by the
  - check Ethernet card properties
  - in the details window, double click and check that **FD01:** IPv6 addresses have been provisioned to the adapter.
  - If FD01: addresses are not provisioned, try deselecting any 3rd party hooks such as Virtual Machine drivers
- On **Node A**, also open a shell on the OpenSDA / OpenLink USB port, to create a new Thread network enter:

$ thr create

Note the status messages in the shell indicating the network parameters. Note a Commissioner instance also starts on Node A

- On **Node B** shell initiate joining with the default PSKd (***THREAD***):

$ thr join

Note the status messages in the shell indicating the joining results.

- On **Node B**, enter the **ifconfig** command to view IP address information**:**

$ ifconfig

Interface 1: 6LoWPAN

Link local address (LL64): fe80::1885:ba8e:d82a:6fad

Mesh local address (ML64): fd4f:12be:69d2::9c67:fec9:9306:4791

Mesh local address (ML16): fd4f:12be:69d2::ff:fe00:400

Unique local address: fd01:: 3ead:18ef:f459:d754:ae31

Link local all Thread Nodes(MCast): ff32:40:fd4f:12be:69d2::01

Realm local all Thread Nodes(MCast): ff33:40:fd4f:12be:69d2::01

Interface 0: Loopback

- On the **Host PC**, launch a terminal window (**cmd** on Windows OS) and ping the FD01::3EAD based address of NodeB:

C:\>ping fd01::3ead:18ef:f459:d754:ae31

Pinging fd01:: 3ead:18ef:f459:d754:ae31 with 32 bytes of data:

Reply from fd01::3ead:18ef:f459:d754:ae31: time=980ms

Reply from fd01::3ead:18ef:f459:d754:ae31: time=25ms

Reply from fd01::3ead:18ef:f459:d754:ae31: time=20ms

Note applications on the Host PC now have IP layer connectivity to Thread nodes.

11.4 External routing via Ethernet on FRDM-K64F – ND host mode and OpenWrt

11.4.1 Overview

This scenario shows how to test external IPv6 connectivity through the FRDM-K64F Ethernet port on the Thread border router device which is provisioned an external IPv6 prefix by an external OpenWrt based device.

For this scenario, a FRDM-K64F with FRDM-CR20A **border_router_freertos** application configuration is deployed in an external IP network such as an existing home or building IP network where Ethernet ports are available for establishing upstream network connectivity.

The application acts as a Border Router with respect to propagating the network provisioning information to the downstream Thread network (such as IPv6 address assignment based on prefix delegation) and interfacing end-to-end IP layer connectivity to the Thread-only nodes.

To run the application use case, the FRDM-K64F must be connected via the Ethernet port to an existing home or enterprise router or switch which provides IP address and IPv6 prefix provisioning by means of DHCPv6-PD (Prefix Delegation), IPv6 Neighbor Discovery Protocol (ND), and/or DHCPv4. When DHCPv4 only is used, the border router is provisioned with an IPv4 address, however the translation between the IPv4 network and the IPv6 Thread destinations must be configured at the application layer.

11.4.2 Board and application configuration

Table 17. Board and application configuration

Nodes Needed: at least 3	
Node	Application
Node A	Thread Router Eligible Device
Node B	Thread Router Eligible Device
Node C	Thread Router Eligible Device

11.4.3 Running the scenario

- Deploy the application built using **border_router_freertos.eww** workspace found at **boards\frdmk64f_frdmcr20a \wireless_examples\thread\border_router\freertos\iar** into IAR EWARM
- In the `source\config.h`, change the define from `BR_ROUTER_MODE` to `BR_HOST_MODE`.

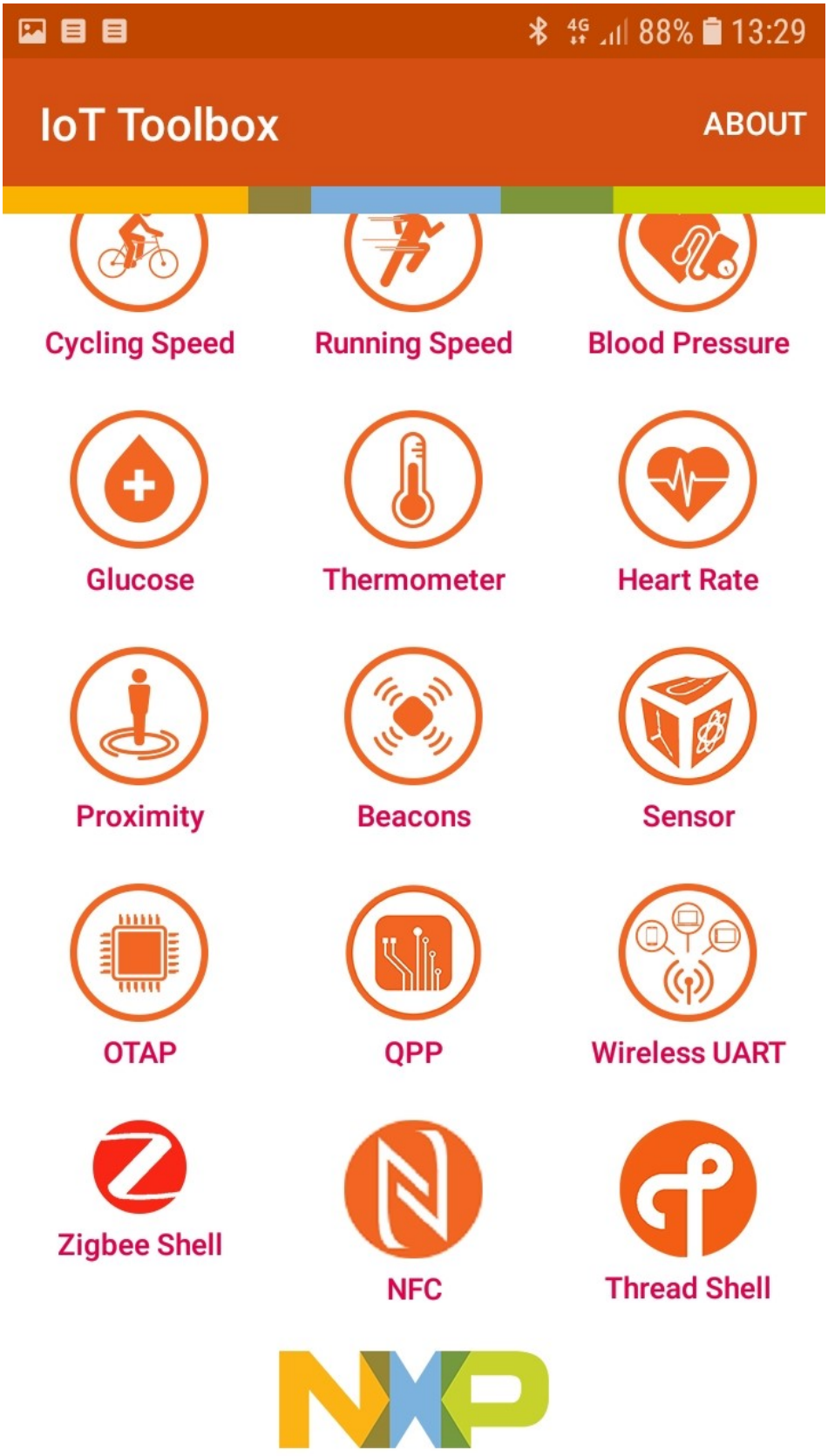


Figure 31. Border Router ND Host Setting

- Deploy **Node B-router_eligible_device_freertos** application to a different board (can be any of the supported development boards)

Before powering on the boards and starting the Thread network connect the **Node A** via Ethernet to the OpenWrt Router as in the image below

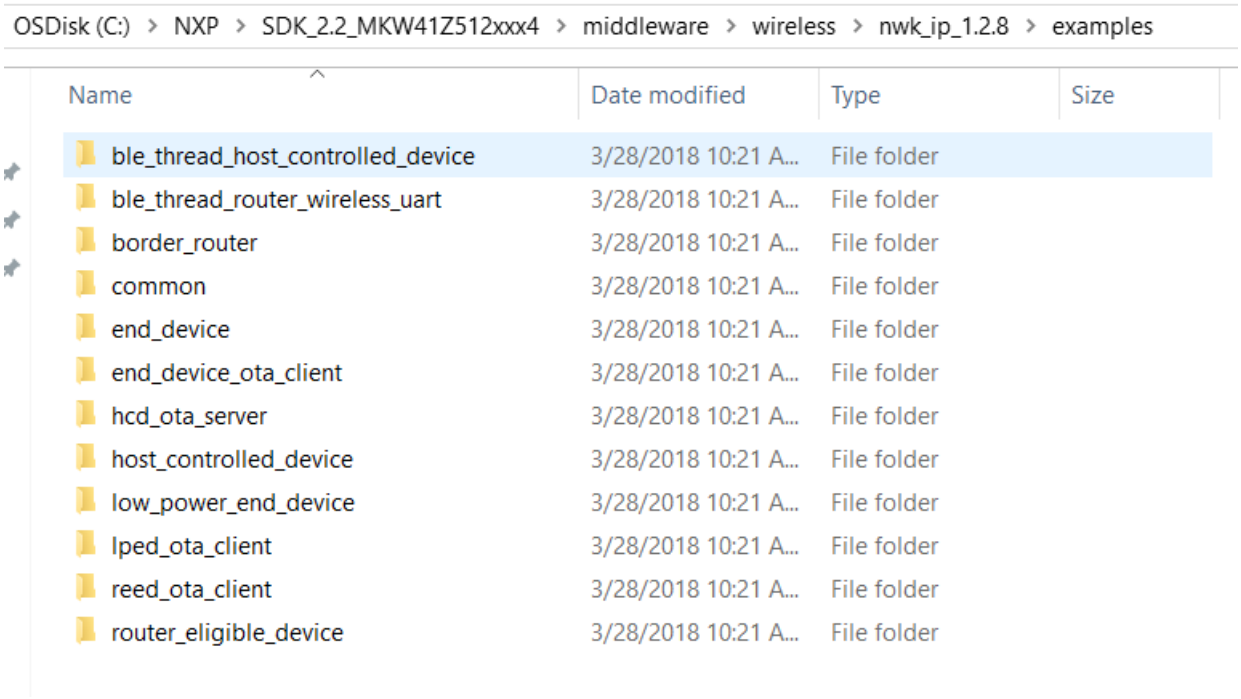


Figure 32. Setup for FRDM-K64F Border Router Application Scenario

- At power up of the FRDM-K64F, the OpenWrt software automatically configures a ULA (Unique Local Address) IPv6 prefix distributed to other devices in the site local networks. This example shows how the site local prefix is distributed via the Border Router to devices in the Thread Network.

- Check the OpenWrt Router provisions an ULA prefix to the home mesh network:

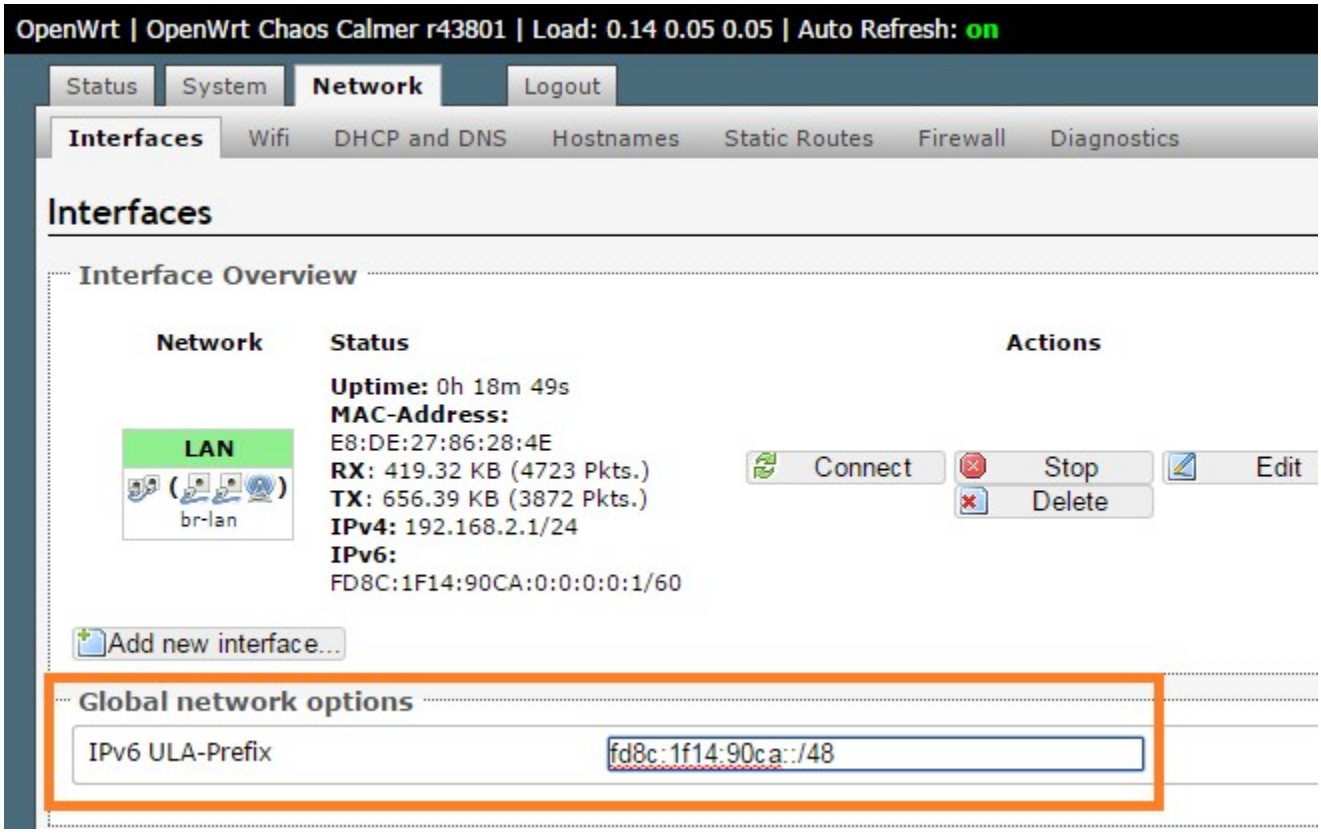


Figure 33. OpenWrt LUCI Web Interface showing IPv6 ULA Prefix Provisioning

- In the image above, the fd8c:1f14:90ca::/48 prefix is being provisioned.
- On **Node A**, also open a shell on the OpenSDA / OpenLink USB port, to create a new Thread network enter:

$ thr create

Note the status messages in the shell indicating the network parameters. Note a Commissioner instance also starts on Node A

- On **Node B** shell initiate joining with the default PSKd (***THREAD***):

$ thr join

Note the status messages in the shell indicating the joining results.

- On **Node A**, enter the **ifconfig** command to view IP address information**:**

$ ifconfig

Note that the Ethernet interface has already been provisioned with the prefix fd8c:1f14:90ca to assign the Unique Local Address. Also note the device has been assigned an IPv4 address via DHCP.

Note that the 6LoWPAN interface has also been provisioned with a subprefix of fd8c:1f14:90ca to assign a Unique Local Address.

- On **Node B**, enter the **ifconfig** command to view IP address information**:**

$ ifconfig

Interface 1: 6LoWPAN

Link local address (LL64): fe80::1885:ba8e:d82a:6fad

Mesh local address (ML64): fd4f:12be:69d2::9c67:fec9:9306:4791

Mesh local address (ML16): fd4f:12be:69d2::ff:fe00:400

Unique local address: fd8c:1f14:90ca:0001:18ef:f459:d754:ae31

Link local all Thread Nodes(MCast): ff32:40:fd4f:12be:69d2::01

Realm local all Thread Nodes(MCast): ff33:40:fd4f:12be:69d2::01

Interface 0: Loopback

Note that the 6LoWPAN interface on the Thread interfaced router has also been provisioned with a subprefix of fd8c:1f14:90ca to assign a Unique Local Address in the site local home network.

- On a PC connected to the same LAN managed as the OpenWrt device open a command terminal prompt
- On the PC, note a routing entry may need to be explicitly added to the host OS routing table.

1. Under Windows OS, assuming the Thread-only device has address fd8c:1f14:90ca:1::1 and the OpenWrt router address is fd8c:1f14:90ca::1

C:\> route ADD fd8c:1f14:90ca:1::1/128 fd8c:1f14:90ca::1

OK!

C:\>ping fd8c:1f14:90ca:0001: 18ef:f459:d754:ae31

Pinging fd8c:1f14:90ca:0001: 18ef:f459:d754:ae31 with 32 bytes of data:

Reply from fd8c:1f14:90ca:0001: 18ef:f459:d754:ae31 : time=932ms

Reply from fd8c:1f14:90ca:0001: 18ef:f459:d754:ae31 : time=24ms

Reply from fd8c:1f14:90ca:0001: 18ef:f459:d754:ae31 : time=22ms

Reply from fd8c:1f14:90ca:0001: 18ef:f459:d754:ae31 : time=11ms

- Under OS X, assuming the Thread-only device has addres fd8c:1f14:90ca:1::1 and the OpenWrt router address is fd8c:1f14:90ca::1

$ sudo route -n add –inet6 fd8c:1f14:90ca:1::1/128 fd8c:1f14:90ca::1

add host fd8c:1f14:90ca:0001: 18ef:f459:d754:ae31 gateway fd8c:1f14:90ca::1

$ ping6 fd8c:1f14:90ca:1::1

PING6(56=40+8+8 bytes) fd8c:1f14:90ca::387b:5427:33fa:b8f5 --> fd8c:1f14:90ca:0001: 18ef:f459:d754:ae31

16 bytes from fd8c:1f14:90ca:0001: 18ef:f459:d754:ae31 , icmp_seq=0 hlim=254 time=1021.479ms

16 bytes from fd8c:1f14:90ca:0001: 18ef:f459:d754:ae31 , icmp_seq=1 hlim=254 time=45.268ms

11.5 External routing via RNDIS-enabled board host mode and OpenWrt

11.5.1 Overview

This scenario shows how to test external IPv6 connectivity through the RNDIS-enabled port on the Thread border router device which is provisioned an external IPv6 prefix by an external OpenWrt-based device. For this scenario, a USB-KW41Z border_router_freertos application configuration must be deployed.

The application acts as a Border Router with respect to propagating the network provisioning information to the downstream Thread network (such as IPv6 address assignment based on prefix delegation) and interfacing end-to-end IP layer connectivity to the Thread-only nodes.

To run the application use case, the RNDIS-enabled board must be connected via the USB port to an existing home or enterprise router or switch which provides IP address and IPv6 prefix provisioning by means of DHCPv6-PD (Prefix Delegation), IPv6 Neighbor Discovery Protocol (ND), and/or DHCPv4. When DHCPv4 only is used, the border router is provisioned with an IPv4 address, however the translation between the IPv4 network and the IPv6 Thread destinations must be configured at the application layer.

11.5.2 Running the scenario

Please read section 11.4.3 that describes the same scenario based on a FRDM-K64F border router. The setup remains the same, except for the following:

- Under source\config.h, change the define from BR_ROUTER_MODE=1 to BR_HOST_MODE=1.
- The OpenWrt router must have USB tethering support (https://wiki.openwrt.org/doc/howto/usb.tethering)

opkg update

opkg install kmod-usb-net kmod-usb-net-rndis kmod-usb-net-cdc-ether usbutils udev

- After the RNDIS-enabled board is plugged to the router, issue `dmesg` and identify the newly created network interface (e.g. eth3).
- This interface (e.g. eth3) must be bridged with the other LAN interfaces to receive management traffic.

The following advisory commands are needed on the OpenWrt router:

ip link set dev eth3 up

brctl addif br-lan eth3

/etc/init.d/odhcpd restart

11.5.3 Re-establishing communication on reset

If the RNDIS-enabled board is reset, the corresponding network interface (e.g. eth3) will be disabled and automatically removed from the bridge. A reissue of the commands presented above to re-establish connectivity is required. The same commands must be applied in the unlikely event of OpenWrt router reset.

Automation of the process at OpenWrt startup, with the following advisory commands added to /etc/rc.local, or directly from the web interface under System -> Startup -> Local Startup:

```
root@OpenWrt:~# cat /etc/rc.local
        # Put your custom commands here that should be executed once
        # the system init finished. By default this file does nothing.

        # wait for br-lan to initialize, assuming ULA prefix is 2001:2002:2003::/48 in this example
        while ! ping6 -c 1 2001:2002:2003::1; do
          sleep 1
        done
        # bridge together the RNDIS interface with the LAN interfaces
            ip link set dev eth3 up
            brctl addif br-lan eth3
        # restart DHCP daemon to trigger Router Advertisements
        /etc/init.d/odhcpd    restart

        exit 0
```

Chapter 12
Host Controlled Interface Applications

12.1 Thread Host controlled interface overview

The Host Controlled Interface firmware exposes a Host to Device control and management API carried over a serial transport over USB and UART and using framing provided by FSCI (Serial Connectivity Interface) transport.

12.2 Exercising the Host controlled interface with Test Tool

12.2.1 Overview

This scenario shows how to exercise the Host Controlled Interface firmware over a USB (UART) connection via the **Test Tool for Connectivity Products** to start a network and add a new device to the network via Commissioning.

12.2.2 Board and application configuration requirements

- Thread nodes and boards configuration

Table 18. Board and application configuration requirements

Nodes Needed: at least 2	
Node	Application
Node A	Host Controlled Device
Node B	Thread Router Eligible Device

- Also needed:
  - **OpenWRT** device with 1 Ethernet LAN port available for the connection to FRDM-K64F. An off-the-shelf home or enterprise router provisioned with **OpenWrt** network software provides such required provisioning. For more information on how to deploy OpenWrt, see openwrt.org. For a list of supported devices to be used with OpenWrt, see the OpenWrt table of hardware at wiki.openwrt.org/toh/start.
  - IPv6 capable PC connected to the OpenWrt device via Ethernet or Wi-Fi

12.2.3 Running the scenario

- To install Test Tool if not present on PC:
  - Go to the KW41Z page
  - Click to go to the **Software & Tools** tab
  - In the **Lab and Test Software** select the link to Download **Test Tool for Connectivity Products**

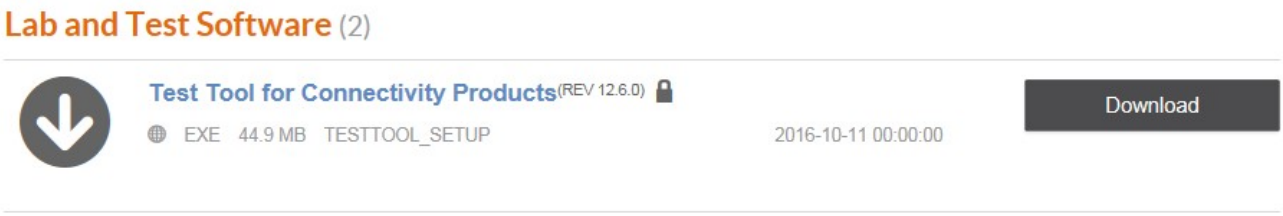


Figure 34. Lab and test software tab

- Sign in or create a nxp.com account and agree to license terms if acceptable
- Wait for the installer to download
- Launch Test Tool Setup and follow the installation wizard to deploy the program
- After Test Tool is installed:
- Copy the **ThreadIP_X.X.X.xml** descriptor file from <Thread Installation>\NXP\SDK_2.2_MKW41Z512xxx4\tools\wireless \xml_fsci to <Test Tool Installation>\Xml (for example, c:\NXP\Test_Tool_12.x.x)
  - Connect **Node A** via USB to the OpenSDA/Openlink port (or the USB port on USB-KW24D512) to the Windows PC
  - Launch Test Tool via Open the application as installed in Windows **Start Menu -> NXP Test Tool -> Test Tool 12**
  - In the tool bar, select **Command Console**
  - The COM port for Node A should be displayed in the panel on the left.
  - **Right-click** the COM port, choose **Configure Device** and change the baud rate to **115200bps**, then click OK

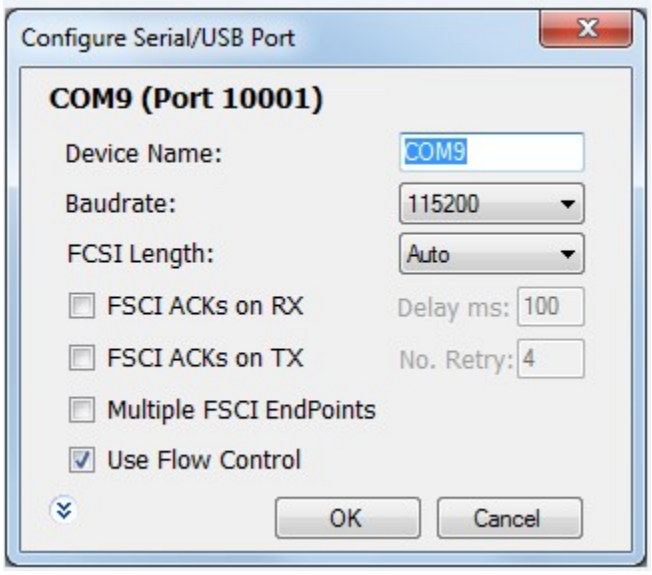


Figure 35. Configuring the serial port in test tool

- Double click the COM port and the Command Console Window opens

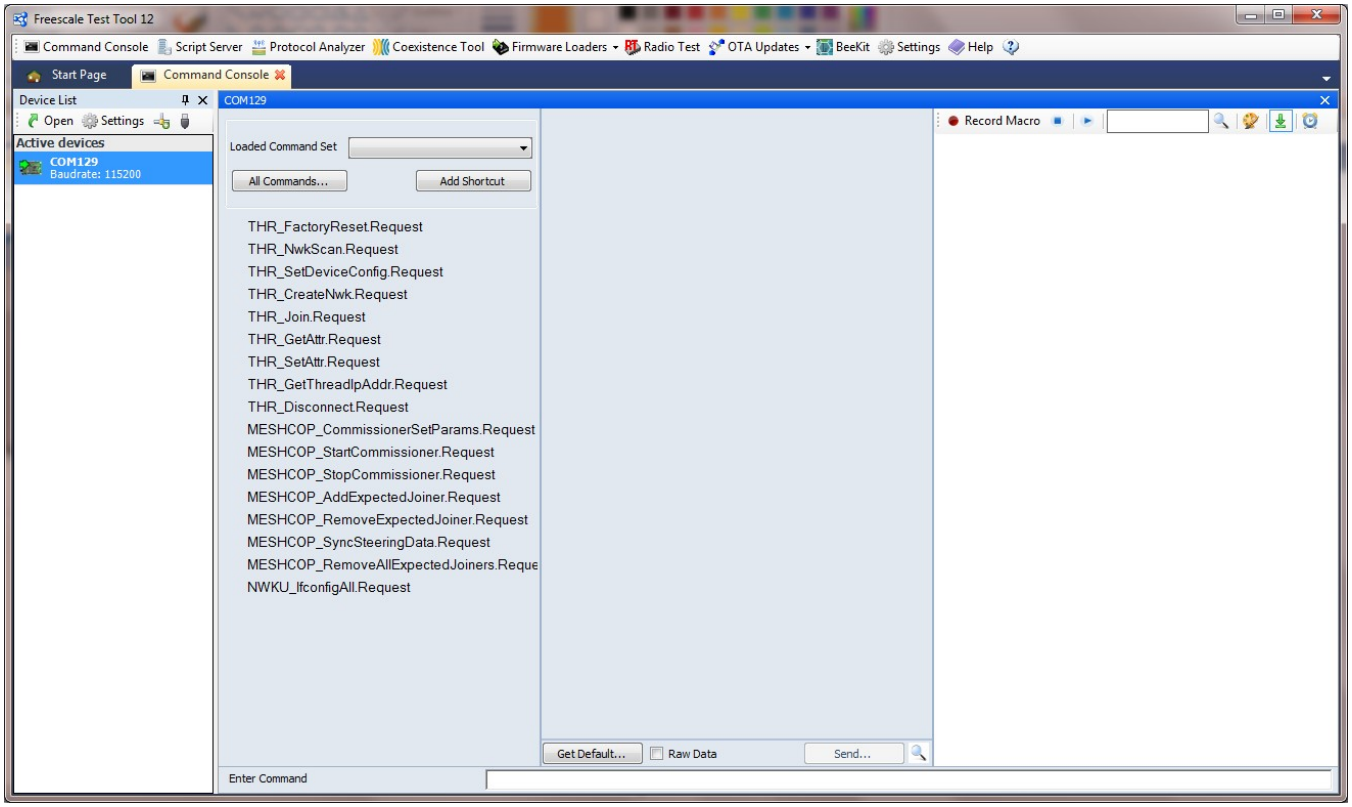


Figure 36. Port opened in command console

- Ensure Node A is in factory reset state (LEDs flashing)
- In the Loaded Command Set select ThreadIP_X.X.X.xml

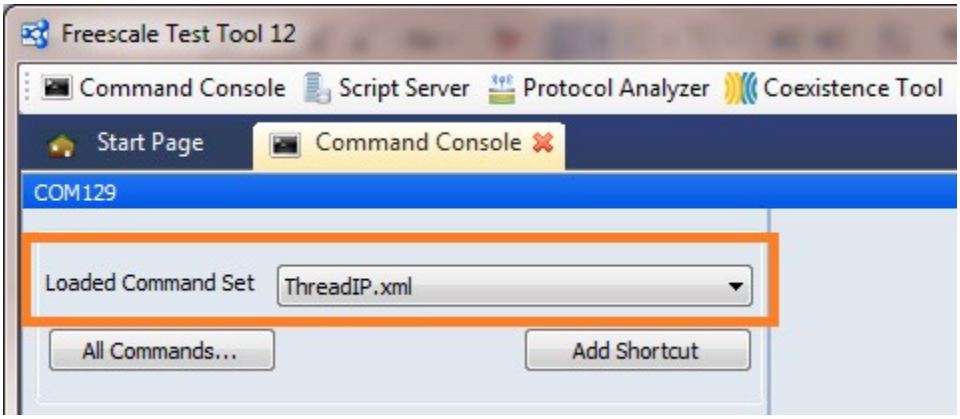


Figure 37. Selecting Thread XML

- Double-click the **THR_CreateNwk.Request** command in the commands panel and note the network creation process displayed in the log

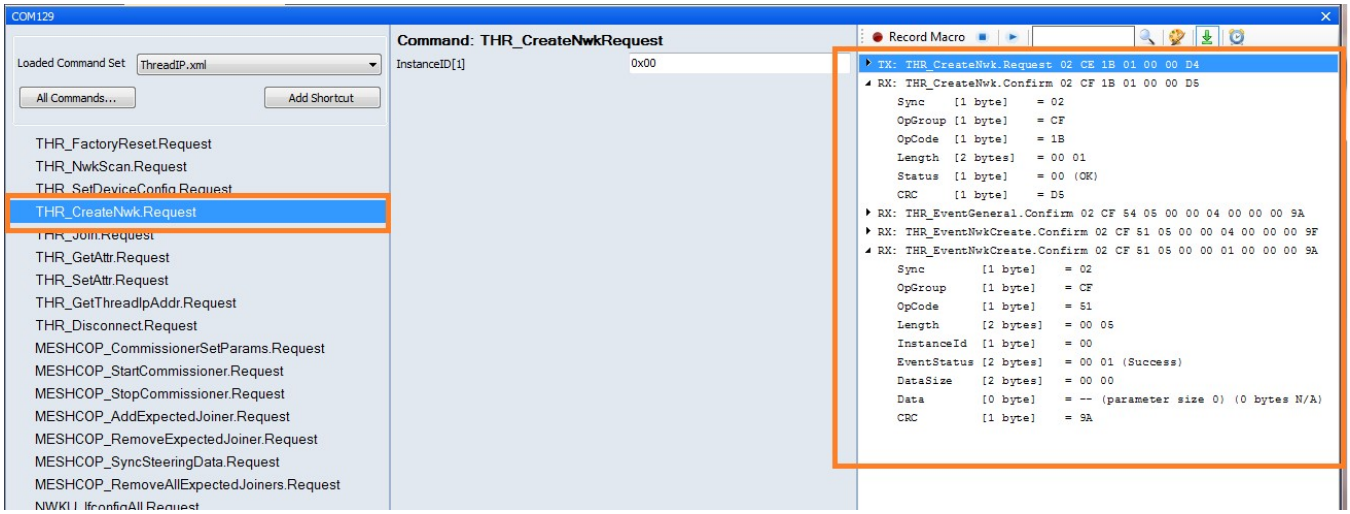


Figure 38. Sending command to Create New Network in Test Tool via THCI

- Double click MESHCOP_StartCommissioner command to start a commissioner on the node with the default settings

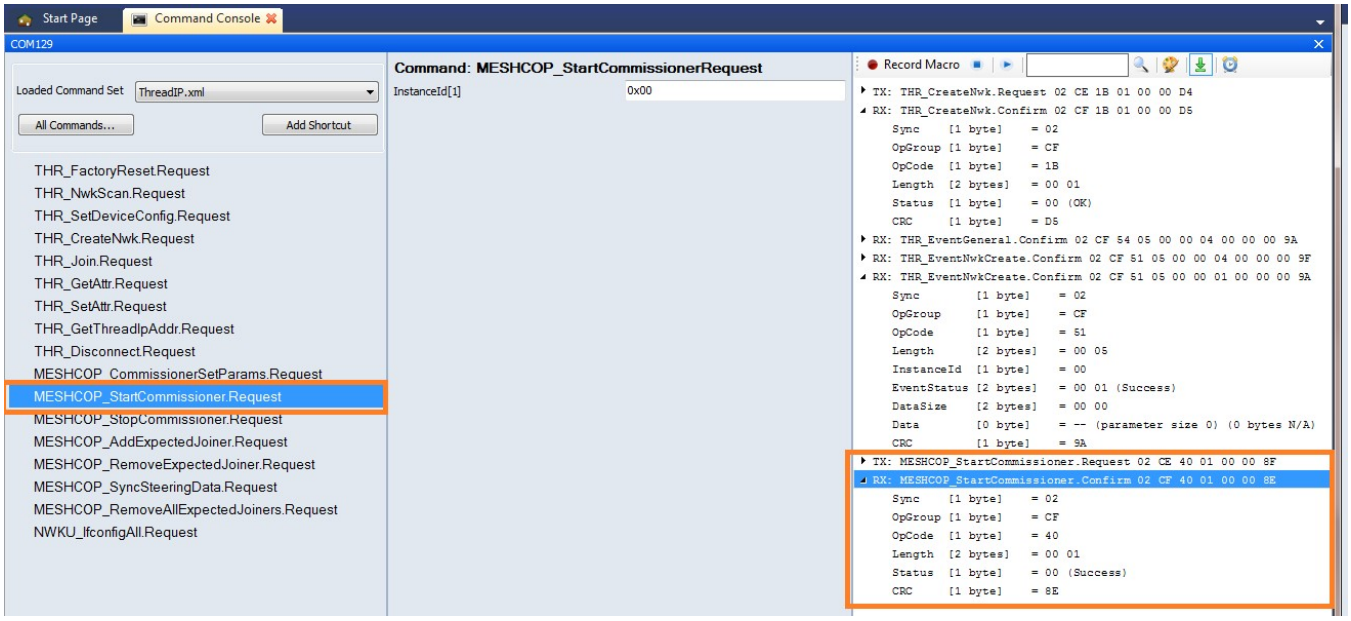


Figure 39. Starting a Commissioner

- Add an expected joiner by clicking command **MESHCOP_AddExpectedJoiner** and filling in the parameters as in the figure below, then click **Send…** at the bottom of the middle panel

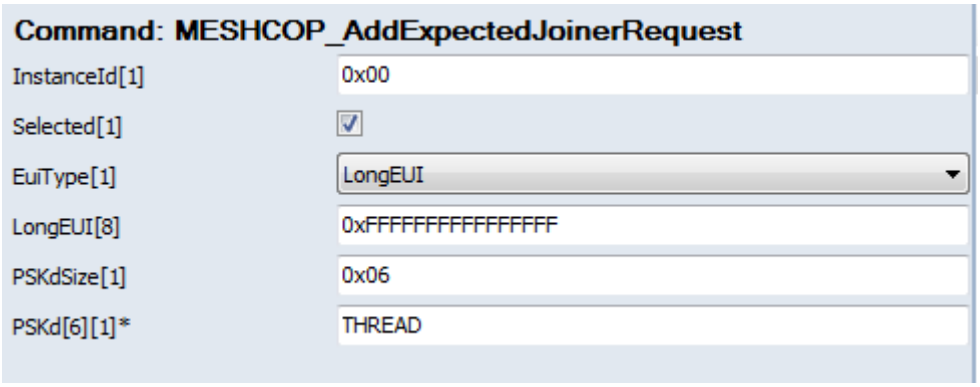


Figure 40. Add Expected Joiner command

```
TX: MESHCOP_AddExpectedJoiner.Request 02 CE 42 12 00 00
 01 01 FF FF FF FF FF FF FF FF 06 54 48 52 45 41 44 96
    Sync        [1 byte]    = 02
    OpGroup     [1 byte]    = CE
    OpCode      [1 byte]    = 42
    Length      [2 bytes]   = 00 12
    InstanceId  [1 byte]    = 00
    Selected    [1 byte]    = 01 (true)
    EuiType     [1 byte]    = 01 (LongEUI)
    LongEUI     [8 bytes]   = FF FF FF FF FF FF FF FF
    PSKdSize    [1 byte]    = 06
    PSKd        [6 bytes]   = 54 48 52 45 41 44 (THREAD)
    CRC         [1 byte]    = 96
RX: MESHCOP_AddExpectedJoiner.Confirm 02 CF 42 01 00 00
 8C
    Sync    [1 byte]    = 02
    OpGroup [1 byte]    = CF
    OpCode  [1 byte]    = 42
    Length  [2 bytes]   = 00 01
    Status  [1 byte]    = 00 (Success)
    CRC     [1 byte]    = 8C
```

Figure 41. Add an Expected Joiner Log

- Steer all nodes to join by clicking **MESHCOP_SyncSteeringData.Request** filling in the parameters as in the figure below, then click **Send…** at the bottom of the middle panel

Figure 42. Steer new devices to allow all joiners

- Initiate **Node B** joining by pressing any switch on the board and check that it joins the network and becomes an Active Router

12.3 Using the Host controlled interface for Linux border router system

For details on integrating the Thread Host Controlled Interface firmware with a Linux OS or OpenWrt border router using the UART or USB transports, see the *Kinetis FSCI Host Application Programming Interface* document.

Chapter 13
Thread and Bluetooth LE Dual Mode Application

13.1 Thread and Bluetooth LE dual mode application overview

The KW41Z Thread Release contains a demo application that allows the control of the dual mode Thread and Bluetooth LE stack via a bluetooth connection from the IoT Toolbox using the Thread Shell application.

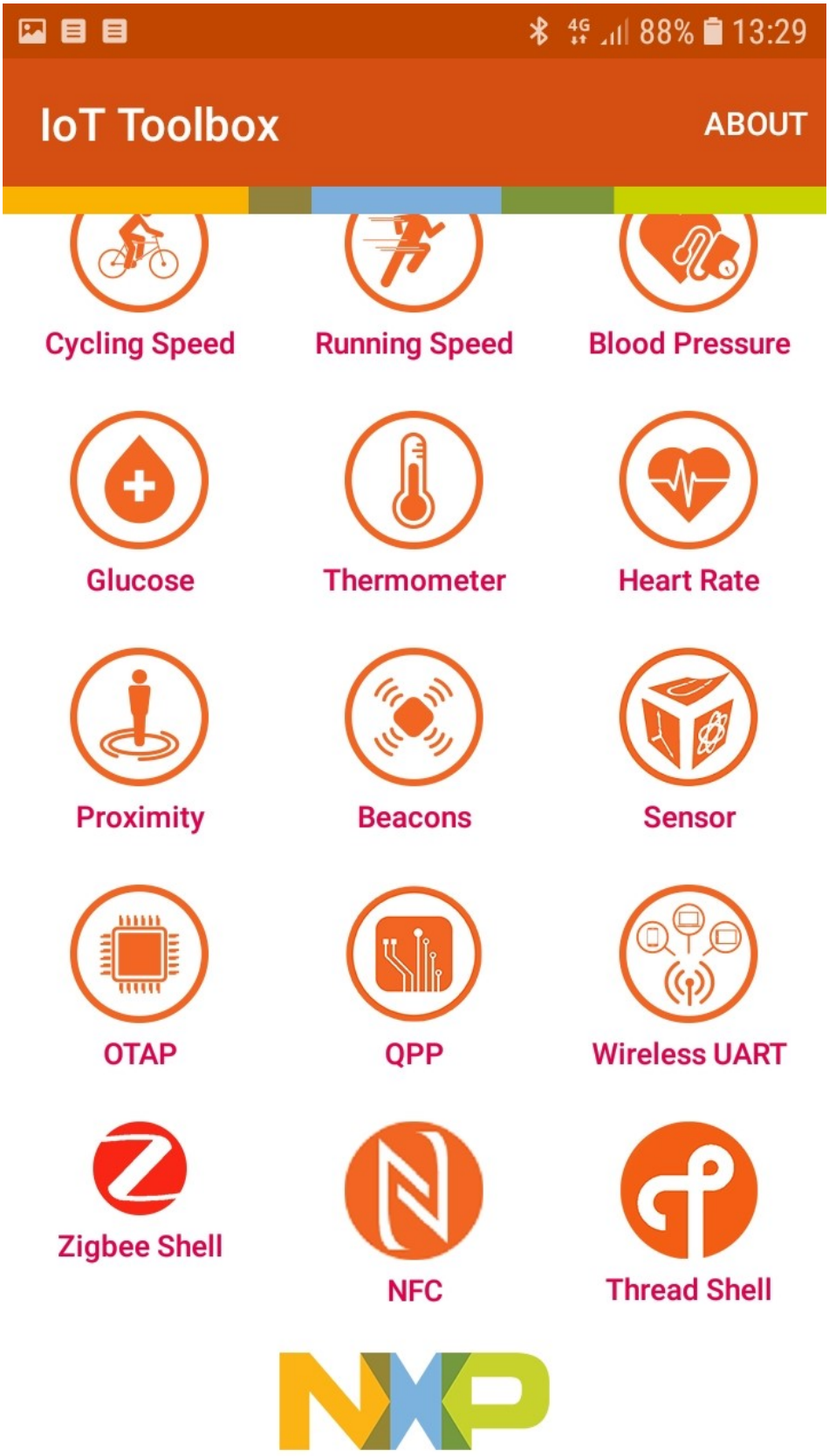


Figure 43. IoT Toolbox Thread Shell

The IoT Toolbox application needs to be downloaded on an Android or iOS-based device from IoT Toolbox – Android or from IoT Toolbox – iOS.

13.2 Thread and Bluetooth LE embedded dual mode application overview

The specific embedded Bluetooth LE application files are located in the <install_directory>\ \middleware\wireless \nwk_ip_1.2.8\examples\ folder.

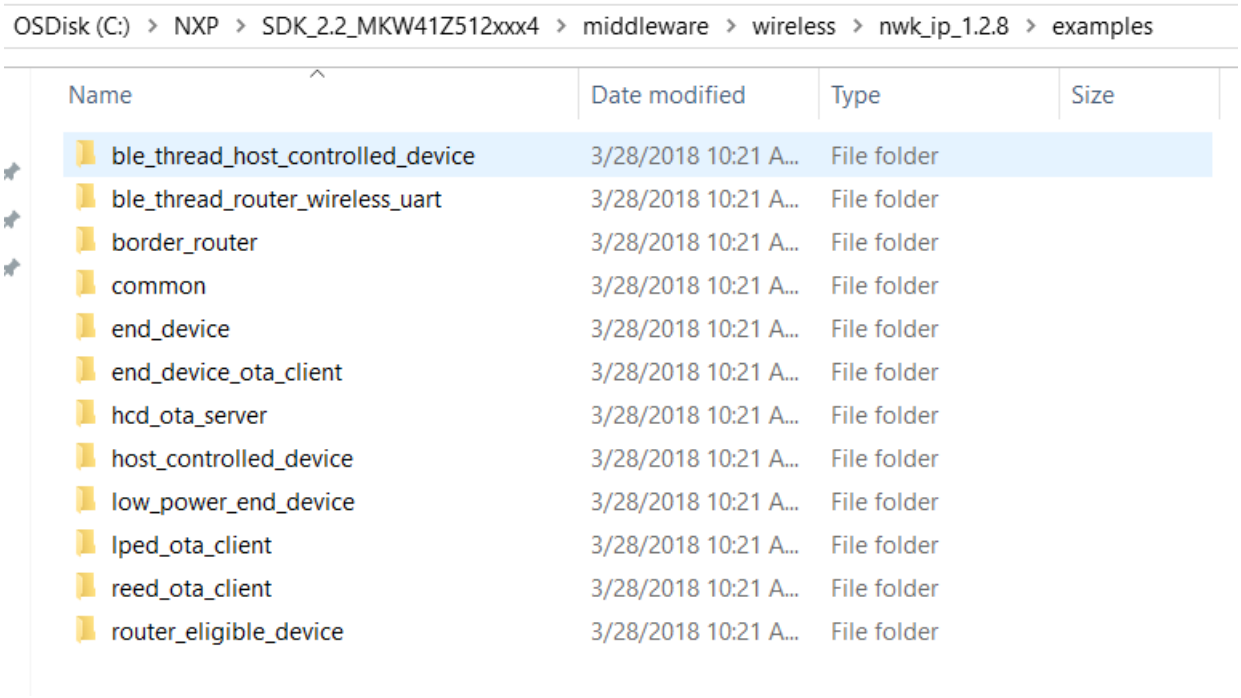


Figure 44. Embedded application files

- <install_directory>\ \middleware\wireless\nwk_ip_1.2.8\examples\ ble_thread_host_controlled_device – contains the specific Bluetooth LE Thread Host Controlled Device application files
- <install_directory>\ \middleware\wireless\nwk_ip_1.2.8\examples\ ble_thread_router_wireless_uart – contains the specific Bluetooth LE Thread Router Wireless UART application files
- <install_directory>\ \middleware\wireless\nwk_ip_1.2.8\examples\common – contains the specific common application files

The embedded Bluetooth LE application is an add-on for the REED application for the ble_thread_router_wireless_uart. ble_thread_host_controlled_device is a black box for both stacks using virtual interfaces on the same serial interface.

The Thread shell application connects the Bluetooth LE stack through the wireless UART profile to a custom interface from the serial manager.

If the device is not connected, the Bluetooth LE stack application remains in advertising mode.

13.3 Steps to connect the IoT Toolbox to the embedded hybrid application

1. Build the `.srec` or `.bin` file for the `ble_thread_router_wireless_uart` embedded project corresponding to the hardware board, using IAR or MCUXpresso IDEs.
2. Download the `ble_thread_router_wireless_uart` image to the board using a Test Tool or a similar flashing tool.
3. Scan using the phone for an active Bluetooth LE connection and select the connection exposed by the `ble_thread_router_wireless_uart` embedded application.
4. Connect the phone to the board.
5. Execute the thread network management commands from the virtual shell HyperTerminal.

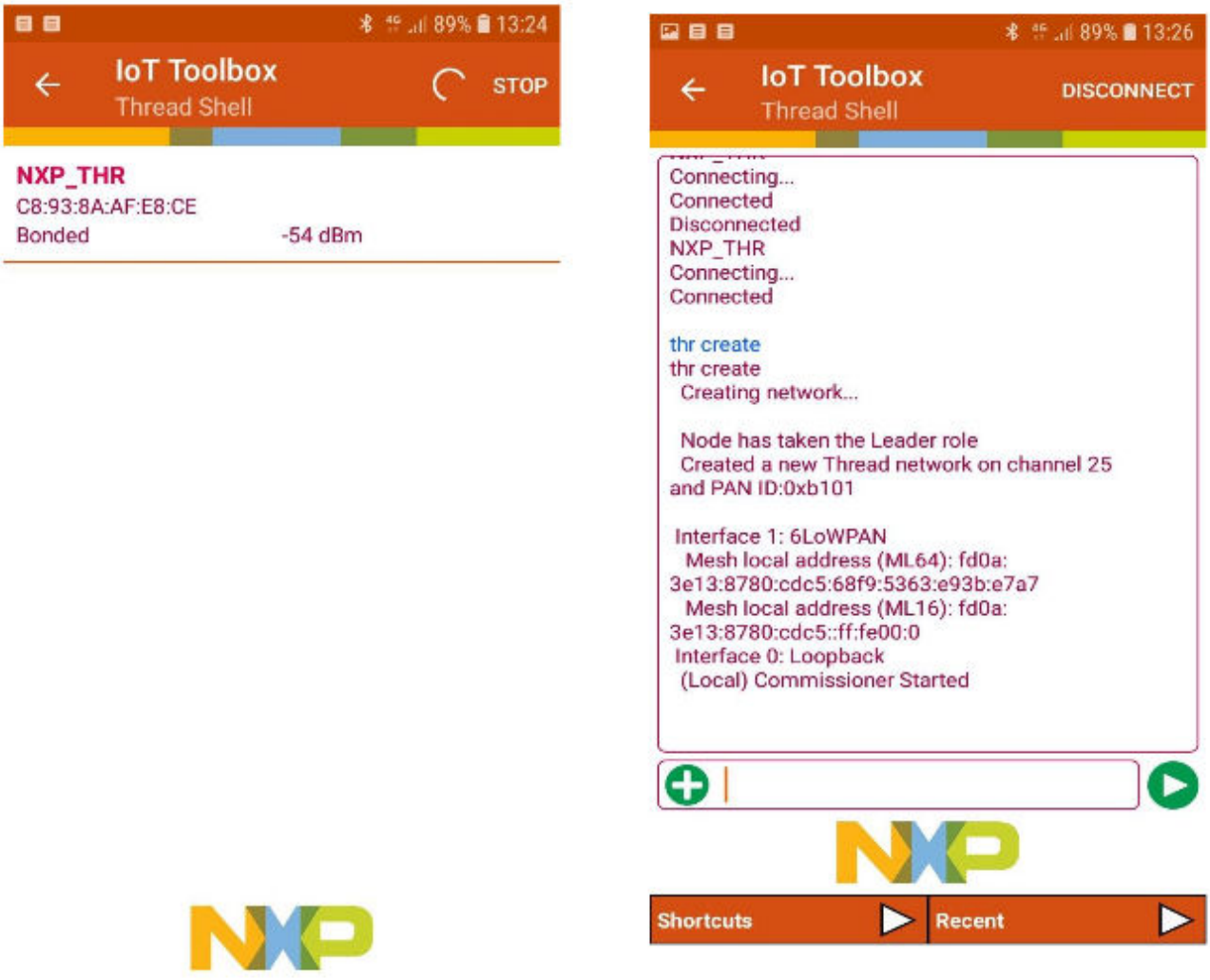


Figure 45. Hybrid Thread/Bluetooth LE device shown in the Thread Shell Bluetooth LE scan interface (left) Executing Thread Network Management Commands (right)

Chapter 14
Development Board User Interface Reference

14.1 Board and application configurations overview

The following sections provide an overview summary of application User Interface feature mapping for each development board platform, including button switch, LED, USB, and Ethernet port settings.

14.2 FRDM-KW24D512 application configurations

14.2.1 FRDM-KW24D512 Thread router eligible device

14.2.1.1 LEDs and switches FRDM-KW24D512 Thread router eligible device

- **Factory Default State**
  - **LEDs**
    - D8 and D17 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1, SW2, SW3, SW4 Short Press Join network
    - SW1, SW2, SW3, SW4 Double Short Press Create network
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - **LEDs**
    - D8 blinking blue Device is in Create/Joining state
    - D17 color cycle Device is in Create/Joining state
  - **Switches**
    - SW1, SW2, SW3, SW4 Short Press Create network
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset
- **Connected State**
  - **LEDs**
    - D8 blinking Device is requesting Router ID to the Leader
    - D8 solid Device is an Active Router
    - D8 off Device is a Router Eligible End Device (REED)
    - D8 toggling briefly Packet activity
    - D17 turns green after Network Create Node has become initial Leader
    - D17 turns off after Network Joining Node is not a Leader

- D17 turns green during Network Operation Router has become a new network partition Leader
- D17 In all other use cases used for CoAP Application LED Control

— **Switches**

- SW1 Short Press Send create data sink multicast announcement
- SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
- SW2 Short Press Send CoAP temperature report multicast or to Data Sink
- SW2 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
- SW3 Short Press Send CoAP LED ON with random RGB values
- SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
- SW4 Short Press Send CoAP LED OFF
- SW4 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
- SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset

14.2.1.2 USB ports USB-KW24D512 Thread router eligible device

- **KW24 USB port:**
  — **Thread Shell Interface** interfaced to KW24D USB CDC Device

14.2.2 FRDM-KW24D512 Thread end device

14.2.2.1 LEDs and switches FRDM-KW24D512 Thread end device

- **Factory Default State**
  — **LEDs**
    - D8 and D17 blinking blue Device is in Factory Default idle state

  — **Switches**
    - SW1, SW2, SW3, SW4 Short Press Join network
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  — **LEDs**
    - D8 blinking blue Device is in Joining state
    - D17 color cycle Device is in Joining state

  — **Switches**
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset
- **Connected State**
  — **LEDs**
    - D8 off Device is an End Device
    - D8 toggling briefly Packet activity
    - D17 In all other use cases used for CoAP Application LED Control

  — **Switches**

- SW1 Short Press Send create data sink multicast announcement
- SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
- SW2 Short Press Send CoAP temperature report multicast or to Data Sink
- SW2 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
- SW3 Short Press Send CoAP LED ON with random RGB values
- SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
- SW4 Short Press Send CoAP LED OFF
- SW4 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
- SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset

14.2.2.2 USB ports FRDM-KW24D512 Thread end device

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via CMSIS-DAP, Segger J-Link:**

Thread Shell Interface interfaced to KW24D UART (115200bps)

- **KW24 USB port**
  - Not available

14.2.3 FRDM-KW24D512 Thread low-power end device

14.2.3.1 LEDs and switches FRDM-KW24D512 Thread low-power end device

- **Factory Default State**
  - **LEDs**
    - D8 and D17 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1, SW2, SW3, SW4 Short Press Join network
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - **LEDs**
    - D8 blinking blue Device is in Joining state
    - D17 color cycle Device is in Joining state
  - **Switches**
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset
- **Low-power State**
  - **LEDs**
    - D8 off Device is an End Device
    - D8 toggling briefly Packet activity on wake up
    - D17 In all other use cases used for CoAP Application LED Control

- **Switches**
  - SW1 Short Press Low-power Wakeup – Wake up state active for 5 seconds

- **Temporary Wake Up State (press SW1 in Low-power State)**
  - **LEDs**
    - D8 off Device is an End Device
    - D8 toggling briefly Packet activity on wake up
    - D17 Used for CoAP Application LED Control
  - **Switches**
    - SW1 Short Press Send create data sink multicast announcement
    - SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
    - SW2 Short Press Send CoAP temperature report multicast or to Data Sink
    - SW2 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
    - SW3 Short Press Send CoAP LED ON with random RGB values
    - SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
    - SW4 Short Press Send CoAP LED OFF
    - SW4 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset

14.2.3.2 USB ports -KW24D512 Thread low-power device

- **KW24 USB port**
  - Not available

14.2.4 FRDM-KW24D512 Thread border router

14.2.4.1 LEDs and Switches FRDM-KW24D512 Thread border router

- **Factory Default State**
  - **LEDs**
    - D8 and D17 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1, SW2, SW3, SW4 Short Press Join network
    - SW1, SW2, SW3, SW4 Double Short Press Create network
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - **LEDs**
    - D8 blinking blue Device is in Create/Joining state
    - D17 color cycle Device is in Create/Joining state
  - **Switches**
    - SW1, SW2, SW3, SW4 Short Press Create network

- SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset

- **Connected State**
  - **LEDs**
    - D8 blinking Device is requesting Router ID to the Leader
    - D8 solid Device is an Active Router
    - D8 off Device is a Router Eligible End Device (REED)
    - D8 toggling briefly Packet activity
    - D17 turns green after Network Create Node has become initial Leader
    - D17 turns off after Network Joining Node is not a Leader
    - D17 turns green during Network Operation Router has become a new network partition Leader
    - D17 In all other use cases used for CoAP Application LED Control
  - **Switches**
    - SW1 Short Press Send create data sink multicast announcement
    - SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
    - SW2 Short Press Send CoAP temperature report multicast or to Data Sink
    - SW2 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
    - SW3 Short Press Send CoAP LED ON with random RGB values
    - SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
    - SW4 Short Press Send CoAP LED OFF
    - SW4 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset

14.2.4.2 USB ports USB-KW24D512 Thread border router

- **KW24 USB port**
  - **Ethernet Emulation over USB CDC (RNDIS)**
    - ND Router with ULA prefix provisioning
    - Thread Host Control Interface (THCI) transportedas raw Ethernet frames

14.2.5 FRDM-KW24D512 Thread Host controlled interface

14.2.5.1 LEDs and switches FRDM-KW24D512 Thread Host controlled interface

- **Factory Default State**
  - **LEDs**
    - D8 and D17 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1, SW2, SW3, SW4 Short Press Join network
    - SW1, SW2, SW3, SW4 Double Short Press Create network

- SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset

- **Network Joining State**
  - **LEDs**
    - D8 blinking blue Device is in Create/Joining state
    - D17 color cycle Device is in Create/Joining state
  - **Switches**
    - SW1, SW2, SW3, SW4 Short Press Create network
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset
- **Connected State**
  - **LEDs**
    - D8 blinking Device is requesting Router ID to the Leader
    - D8 solid Device is an Active Router
    - D8 off Device is a Router Eligible End Device (REED)
    - D8 toggling briefly Packet activity
    - D17 turns green after Network Create Node has become initial Leader
    - D17 turns off after Network Joining Node is not a Leader
    - D17 turns green during Network Operation Router has become a new network partition Leader
    - D17 In all other use cases used for CoAP Application LED Control
  - **Switches**
    - SW1 Short Press Send create data sink multicast announcement
    - SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
    - SW2 Short Press Send CoAP temperature report multicast or to Data Sink
    - SW2 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
    - SW3 Short Press Send CoAP LED ON with random RGB values
    - SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
    - SW4 Short Press Send CoAP LED OFF
    - SW4 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
    - SW1, SW2, SW3, SW4 Very Long Press (> 8 seconds) – Factory reset

14.2.5.2 USB ports FRDM-KW24D512 Thread Host controlled interface

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via CMSIS-DAP, Segger J-Link, P&E Micro:**

Thread Host Control Interface (THCI) interfaced to KW24D UART (115200bps)

- **KW24 USB port**
  - Not available

14.3 USB-KW24D512 application configurations

14.3.1 USB-KW24D512 Thread router eligible device

14.3.1.1 LEDs and switches USB-KW24D512 Thread router eligible device

- **Factory Default State**
  - **LEDs**
    - D2, D3 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1 Short Press Join network
    - SW1 Double Short Press Create network
    - SW1 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - **LEDs**
    - D2, D3 cycle/sequence Device is in Create/Joining state
  - **Switches**
    - SW1 Short Press Create network
    - SW1 Very Long Press (> 8 seconds) – Factory reset
- **Connected State**
  - **LEDs**
    - D2 blinking Device is requesting Router ID to the Leader
    - D2 solid Device is an Active Router
    - D2 off Device is a Router Eligible End Device (REED)
    - D3 Used for CoAP Application LED Control
    - D3 toggling briefly Packet activity
  - **Switches**
    - SW1 Short Press Send create data sink multicast announcement
    - SW1 Long Press (2-3 seconds) Send release data sink multicast announcement

14.3.1.2 USB ports USB-KW24D512 Thread router eligible device

- **KW24 USB port:**
  - **Thread Shell Interface** interfaced to KW24D USB CDC Device

14.3.2 USB-KW24D512 Thread end device

14.3.2.1 LEDs and switches USB-KW24D512 Thread end device

- **Factory Default State**
  - **LEDs**
    - D2, D3 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1 Short Press Join network
    - SW1 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - **LEDs**
    - D2, D3 cycle/sequence Device is in Joining state
  - **Switches**
    - SW1 Very Long Press (> 8 seconds) – Factory reset
- **Connected State**
  - **LEDs**
    - D2 off Device is an End Device
    - D3 Used for CoAP Application LED Control
    - D3 toggling briefly Packet activity
  - **Switches**
    - SW1 Short Press Send create data sink multicast announcement
    - SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
    - SW1 (> 8 seconds) – Factory reset

14.3.2.2 USB ports USB-KW24D512 Thread end device

- **OpenLink/OpenSDA Mini-USB:**
  - **Serial COM port via P&E Micro, CMSIS-DAP, SEGGER J-Link:**

Thread Shell Interface interfaced to KW24D UART (115200bps)

- **KW24 USB port:**
  - **Thread Shell Interface** interfaced to KW24D USB CDC Device

14.3.3 USB-KW24D512 Thread low-power end device

14.3.3.1 LEDs and switches USB-KW41Z Thread low-power end device

- **Factory Default State**
  - **LEDs**
    - D2, D3 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1 Short Press Join network

- SW1 Very Long Press (> 8 seconds) – Factory reset

- **Network Joining State**
  - **LEDs**
    - D2, D3 cycle/sequence Device is in Joining state
  - **Switches**
    - SW1 Very Long Press (> 8 seconds) – Factory reset
- **Low-power State**
  - **LEDs**
    - D2 off Device is an End Device
    - D3 Used for CoAP Application LED Control
    - D3 toggling briefly Packet activity on wake up
  - **Switches**
    - SW1 Low-power Wakeup – Wake up state active for 5 seconds
- **Temporary Wake Up State (press SW1 in Low-power State)**
  - **LEDs**
    - D2 off Device is an End Device
    - D3 Used for CoAP Application LED Control
    - D3 toggling briefly Packet activity on wake up
  - **Switches**
    - SW1 Short Press Send create data sink multicast announcement
    - SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
    - SW1 Very Long Press (> 8 seconds) – Factory reset

14.3.3.2 USB ports -KW24D512 Thread low-power device

- **KW24 USB port**
  - Not available

14.3.4 USB-KW24D512 Thread border router

14.3.4.1 LEDs and switches USB-KW24D512 Thread border router

- **Factory Default State**
  - **LEDs**
    - D2 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1 Short Press Join network
    - SW1 Double Short Press Create network
    - SW1 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**

- **LEDs**
  - D2, D3 cycle/sequence Device is in Create/Joining state
- **Switches**
  - SW1 Short Press Create network
  - SW1 Very Long Press (> 8 seconds) – Factory reset

- **Connected State**
  - **LEDs**
    - D2 blinking Device is requesting Router ID to the Leader
    - D2 solid Device is an Active Router
    - D2 off Device is a Router Eligible End Device (REED)
    - D3 Used for CoAP Application LED Control
    - D3 toggling briefly Packet activity
  - **Switches**
    - SW1 Short Press Send create data sink multicast announcement
    - SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
    - SW1 Very Long Press (> 8 seconds) – Factory reset

14.3.4.2 USB ports USB-KW24D512 Thread border router

- **KW24 USB port**
  - **Ethernet Emulation over USB CDC (RNDIS)**
    - ND Router with ULA prefix provisioning
    - Thread Host Control Interface (THCI) transportedas raw Ethernet frames

14.3.5 USB-KW24D512 Thread Host controlled interface

14.3.5.1 LEDs and switches USB-KW24D512 Thread Host controlled interface

- **Factory Default State**
  - **LEDs**
    - D2 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1 Short Press Join network
    - SW1 Double Short Press Create network
    - SW1 Very Long Press (> 8 seconds) – Factory reset

- **Network Joining State**
  - **LEDs**
    - D2, D3 cycle/sequence Device is in Create/Joining state
  - **Switches**

- SW1 Short Press Create network
- SW1 Very Long Press (> 8 seconds) – Factory reset

- **Connected State**
  - **LEDs**
    - D2 blinking Device is requesting Router ID to the Leader
    - D2 solid Device is an Active Router
    - D2 off Device is a Router Eligible End Device (REED)
    - D3 Used for CoAP Application LED Control
    - D3 toggling briefly Packet activity
  - **Switches**
    - SW1 Short Press Send create data sink multicast announcement
    - SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
    - SW1 Very Long Press (> 8 seconds) – Factory reset

14.3.5.2 USB ports USB-KW24D512 Thread Host controlled interface

- **KW24 USB port**
  - **Thread Host Control Interface (THCI)** interfaced to KW24D USB CDC Device

14.4 FRDM-K64F with FRDM-CR20A application configurations

14.4.1 FRDM-K64F with FRDM-CR20A Thread router eligible device

14.4.1.1 FRDM-K64F with FRDM-CR20A application configurations

- **Factory Default State**
  - **LEDs**
    - FRDM-CR20A RGB LED2 blinking blue Device is in Factory Default idle state
    - FRDM-K64F RGB D12 blinking blue Device is in Factory Default idle state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Short Press Join network
    - FRDM-K64F SW3, SW2 Short Press Join network
    - FRDM-CR20A SW1, SW2 Double Short Press Create network
    - FRDM-K64F SW3, SW2 Double Short Press Create network
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) Factory reset
- **Network Joining State**

- **LEDs**
  - FRDM-K64F RGB D12 blinking blue Device is in Create/Joining state
  - FRDM-CR20A RGB LED 2 color cycle Device is in Create/Joining state
- **Switches**
  - FRDM-CR20A SW1, SW2 Short Press Create network
  - FRDM-K64F SW3, SW2 Short Press Create network
  - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
  - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) Factory reset

- **Connected State**
  - **LEDs**
    - FRDM-K64F RGB D12 blinking blue Device is requesting Router ID to the Leader
    - FRDM-K64F RGB D12 solid blue Device is an Active Router
    - FRDM-K64F RGB D12 off Device is a Router Eligible End Device (REED)
    - FRDM-K64F RGB D12 toggling briefly Packet activity
    - FRDM-CR20A RGB LED2 turns green after Network Create Node has become initial Leader
    - FRDM-CR20A RGB LED2 turns off after Network Joining Node is not a Leader
    - FRDM-CR20A RGB LED2 turns green during Network Operation Router has become a new network partition Leader
    - FRDM-CR20A RGB LED2 In all other use cases used for CoAP Application LED Control
  - **Switches**
    - FRDM-K64F SW3 Short Press Send create data sink multicast announcement
    - FRDM-K64F SW3 Long Press (2-3 seconds) Send release data sink multicast announcement
    - FRDM-K64F SW2 Short Press Send CoAP temperature report multicast or to Data Sink
    - FRDM-K64F SW2 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
    - FRDM-CR20A SW2 Short Press Send CoAP LED ON with random RGB values
    - FRDM-CR20A SW1 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
    - FRDM-CR20A SW1 Short Press Send CoAP LED OFF
    - FRDM-CR20A SW2 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) – Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) – Factory reset

14.4.1.2 USB ports FRDM-K64F with FRDM-CR20A Thread router eligible device

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via CMSIS-DAP, Segger J-Link:**

Thread Shell Interface interfaced to K64F UART (115200bps)

- **K64F USB port:**
  - Not available

14.4.1.3 Ethernet port FRDM-K64F with FRDM-CR20A Thread router eligible device

- Not available

14.4.2 FRDM-K64F with FRDM-CR20A Thread end device

14.4.2.1 LEDs and switches FRDM-K64F with FRDM-CR20A Thread end device

- **Factory Default State**
  - **LEDs**
    - FRDM-CR20A RGB LED2 blinking blue Device is in Factory Default idle state
    - FRDM-K64F RGB D12 blinking blue Device is in Factory Default idle state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Short Press Join network
    - FRDM-K64F SW3, SW2 Short Press Join network
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) Factory reset
- **Network Joining State**
  - **LEDs**
    - FRDM-K64F RGB D12 blinking blue Device is in Joining state
    - FRDM-CR20A RGB LED 2 color cycle Device is in Joining state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) Factory reset
- **Connected State**
  - **LEDs**
    - FRDM-K64F RGB D12 off Device is an End Device
    - FRDM-K64F RGB D12 toggling briefly Packet activity
    - FRDM-CR20A RGB LED2 used for CoAP Application LED Control
  - **Switches**
    - FRDM-K64F SW3 Short Press Send create data sink multicast announcement
    - FRDM-K64F SW3 Long Press (2-3 seconds) Send release data sink multicast announcement
    - FRDM-K64F SW2 Short Press Send CoAP temperature report multicast or to Data Sink
    - FRDM-K64F SW2 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
    - FRDM-CR20A SW2 Short Press Send CoAP LED ON with random RGB values
    - FRDM-CR20A SW1 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
    - FRDM-CR20A SW1 Short Press Send CoAP LED OFF
    - FRDM-CR20A SW2 Long Press (2-3 seconds) Send CoAP LED Blink/Flash

- FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) – Factory reset
- FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) – Factory reset

14.4.2.2 USB ports FRDM-K64F with FRDM-CR20A Thread end device

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via CMSIS-DAP, Segger J-Link:**

Thread Shell Interface interfaced to K64F UART (115200bps)

- **K64F USB port:**
  - Not available

14.4.2.3 Ethernet port FRDM-K64F with FRDM-CR20A Thread end device

- Not available

14.4.3 FRDM-K64F with FRDM-CR20A Thread low-power end device

14.4.3.1 LEDs and switches FRDM-K64F with FRDM-CR20A Thread low-power end device

- **Factory Default State**
  - **LEDs**
    - FRDM-CR20A RGB LED2 blinking blue Device is in Factory Default idle state
    - FRDM-K64F RGB D12 blinking blue Device is in Factory Default idle state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Short Press Join network
    - FRDM-K64F SW3, SW2 Short Press Join network
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) Factory reset
- **Network Joining State**
  - **LEDs**
    - FRDM-K64F RGB D12 blinking blue Device is in Joining state
    - FRDM-CR20A RGB LED 2 color cycle Device is in Joining state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) Factory reset
- **Low-power State**
  - **LEDs**
    - FRDM-K64F RGB D12 off Device is an End Device

- FRDM-CR20A RGB LED 2 Used for CoAP Application LED Control
- FRDM-K64F RGB D12 toggling briefly Packet activity on wake up

— **Switches**

- FRDM-K64F SW3, SW2 Low-power Wakeup – Wake up state active for 5 seconds

- **Temporary Wake Up State (press FRDM-K64F SW2 or SW3 in Low-power State)**

— **LEDs**

- FRDM-K64F RGB D12 blinking blue Device is requesting Router ID to the Leader
- FRDM-K64F RGB D12 solid blue Device is an Active Router
- FRDM-K64F RGB D12 off Device is a Router Eligible End Device (REED)
- FRDM-K64F RGB D12 toggling briefly Packet activity
- FRDM-CR20A RGB LED2 turns green after Network Create Node has become initial Leader
- FRDM-CR20A RGB LED2 turns off after Network Joining Node is not a Leader
- FRDM-CR20A RGB LED2 turns green during Network Operation Router has become a new network partition Leader
- FRDM-CR20A RGB LED2 In all other use cases used for CoAP Application LED Control

— **Switches**

- FRDM-K64F SW3 Short Press Send create data sink multicast announcement
- FRDM-K64F SW3 Long Press (2-3 seconds) Send release data sink multicast announcement
- FRDM-K64F SW2 Short Press Send CoAP temperature report multicast or to Data Sink
- FRDM-K64F SW2 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
- FRDM-CR20A SW2 Short Press Send CoAP LED ON with random RGB values
- FRDM-CR20A SW1 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
- FRDM-CR20A SW1 Short Press Send CoAP LED OFF
- FRDM-CR20A SW2 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
- FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) – Factory reset
- FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) – Factory reset

14.4.3.2 USB Ports FRDM-K64F with FRDM-CR20A Thread low-power end device

- **OpenLink/OpenSDA USB:**

— **Serial COM port via CMSIS-DAP, Segger J-Link:**

- Not available

- **K64F USB port:**

— Not available

14.4.3.3 Ethernet port FRDM-K64F with FRDM-CR20A Thread low-power end device

- Not available

14.4.4 FRDM-K64F with FRDM-CR20A Thread border router device

14.4.4.1 LEDs and switches FRDM-K64F with FRDM-CR20A Thread border router device

- **Factory Default State**
  - **LEDs**
    - FRDM-CR20A RGB LED2 blinking blue Device is in Factory Default idle state
    - FRDM-K64F RGB D12 blinking blue Device is in Factory Default idle state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Short Press Join network
    - FRDM-K64F SW3, SW2 Short Press Join network
    - FRDM-CR20A SW1, SW2 Double Short Press Create network
    - FRDM-K64F SW3, SW2 Double Short Press Create network
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) Factory reset
- **Network Joining State**
  - **LEDs**
    - FRDM-K64F RGB D12 blinking blue Device is in Create/Joining state
    - FRDM-CR20A RGB LED 2 color cycle Device is in Create/Joining state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Short Press Create network
    - FRDM-K64F SW3, SW2 Short Press Create network
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) Factory reset
- **Connected State**
  - **LEDs**
    - FRDM-K64F RGB D12 blinking blue Device is requesting Router ID to the Leader
    - FRDM-K64F RGB D12 solid blue Device is an Active Router
    - FRDM-K64F RGB D12 off Device is a Router Eligible End Device (REED)
    - FRDM-K64F RGB D12 toggling briefly Packet activity
    - FRDM-CR20A RGB LED2 turns green after Network Create Node has become initial Leader
    - FRDM-CR20A RGB LED2 turns off after Network Joining Node is not a Leader
    - FRDM-CR20A RGB LED2 turns green during Network Operation Router has become a new network partition Leader
    - FRDM-CR20A RGB LED2 In all other use cases used for CoAP Application LED Control
  - **Switches**
    - FRDM-K64F SW3 Short Press Send create data sink multicast announcement

- FRDM-K64F SW3 Long Press (2-3 seconds) Send release data sink multicast announcement
- FRDM-K64F SW2 Short Press Send CoAP temperature report multicast or to Data Sink
- FRDM-K64F SW2 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
- FRDM-CR20A SW2 Short Press Send CoAP LED ON with random RGB values
- FRDM-CR20A SW1 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
- FRDM-CR20A SW1 Short Press Send CoAP LED OFF
- FRDM-CR20A SW2 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
- FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) – Factory reset
- FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) – Factory reset

14.4.4.2 USB ports FRDM-K64F with FRDM-CR20A Thread border router

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via CMSIS-DAP, Segger J-Link:**

Thread Shell Interface interfaced to K64F UART (115200 bps)

- **K64F USB port:**
  - Not available

14.4.4.3 Ethernet port FRDM-K64F with FRDM-CR20A Thread border router

- Ethernet enabled
- ND Enet Router with ULA prefix provisioning
- Can be changed to ND Enet Host at compile time

14.4.5 FRDM-K64F with FRDM-CR20A Thread Host controlled interface device

14.4.5.1 LEDs and switches FRDM-K64F with FRDM-CR20A Thread Host controlled interface

- **Factory Default State**
  - **LEDs**
    - FRDM-CR20A RGB LED2 blinking blue Device is in Factory Default idle state
    - FRDM-K64F RGB D12 blinking blue Device is in Factory Default idle state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Short Press Join network
    - FRDM-K64F SW3, SW2 Short Press Join network
    - FRDM-CR20A SW1, SW2 Double Short Press Create network
    - FRDM-K64F SW3, SW2 Double Short Press Create network

    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) Factory reset

- **Network Joining State**
  - **LEDs**
    - FRDM-K64F RGB D12 blinking blue Device is in Create/Joining state
    - FRDM-CR20A RGB LED 2 color cycle Device is in Create/Joining state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Short Press Create network
    - FRDM-K64F SW3, SW2 Short Press Create network
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) Factory reset
- **Connected State**
  - **LEDs**
    - FRDM-K64F RGB D12 blinking blue Device is requesting Router ID to the Leader
    - FRDM-K64F RGB D12 solid blue Device is an Active Router
    - FRDM-K64F RGB D12 off Device is a Router Eligible End Device (REED)
    - FRDM-K64F RGB D12 toggling briefly Packet activity
    - FRDM-CR20A RGB LED2 turns green after Network Create Node has become initial Leader
    - FRDM-CR20A RGB LED2 turns off after Network Joining Node is not a Leader
    - FRDM-CR20A RGB LED2 turns green during Network Operation Router has become a new network partition Leader
    - FRDM-CR20A RGB LED2 In all other use cases used for CoAP Application LED Control
  - **Switches**
    - FRDM-K64F SW3 Short Press Send create data sink multicast announcement
    - FRDM-K64F SW3 Long Press (2-3 seconds) Send release data sink multicast announcement
    - FRDM-K64F SW2 Short Press Send CoAP temperature report multicast or to Data Sink
    - FRDM-K64F SW2 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
    - FRDM-CR20A SW2 Short Press Send CoAP LED ON with random RGB values
    - FRDM-CR20A SW1 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
    - FRDM-CR20A SW1 Short Press Send CoAP LED OFF
    - FRDM-CR20A SW2 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) – Factory reset
    - FRDM-K64F SW3, SW2 Very Long Press (> 8 seconds) – Factory reset

14.4.5.2 USB ports FRDM-KW64F with FRDM-CR20A Thread Host controlled interface

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via CMSIS-DAP, Segger J-Link:**

  - **Thread Host Control Interface (THCI)** interfaced to K64F UART (115200bps)
- **K64F USB port:**
  - Not available

14.4.5.3 Ethernet port FRDM-K64F with FRDM-CR20A Thread Host controlled interface

- Not available

14.5 FRDM-KL46Z with FRDM-CR20A application configurations

14.5.1 FRDM-KL46Z with FRDM-CR20A Thread end device

14.5.1.1 LEDs and switches FRDM-KL46Z with FRDM-CR20A Thread end device

- **Factory Default State**
  - **LEDs**
    - FRDM-CR20A RGB LED2 blinking blue Device is in Factory Default idle state
    - FRDM-CR20A red LED1 in solid state Device is in factory default state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Short Press Join network
    - FRDM-KL46Z SW3, SW1 Short Press Join network
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-KL46Z SW3, SW1 Very Long Press (> 8 seconds) Factory reset
- **Network Joining State**
  - **LEDs**
    - FRDM-CR20A red LED1 in solid state Device is in Joining state
    - FRDM-CR20A RGB LED 2 color cycle Device is in Joining state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-KL46Z SW3, SW1 Very Long Press (> 8 seconds) Factory reset
- **Connected State**
  - **LEDs**
    - FRDM-CR20A red LED1 toggling briefly Packet activity
    - FRDM-CR20A RGB LED2 used for CoAP Application LED Control
  - **Switches**
    - FRDM-CR20A SW2 Short Press Send create data sink multicast announcement

- FRDM-CR20A SW2 Long Press (2-3 seconds) Send release data sink multicast announcement
- FRDM-CR20A SW1 Short Press Send CoAP temperature report multicast or to Data Sink
- FRDM-CR20A SW1 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
- FRDM-KL46Z SW3 Short Press Send CoAP LED ON with random RGB values
- FRDM-KL46Z SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
- FRDM-KL46Z SW1 Short Press Send CoAP LED OFF
- FRDM-KL46Z SW1 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
- FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) – Factory reset
- FRDM-KL46Z SW3, SW1 Very Long Press (> 8 seconds) – Factory reset

14.5.1.2 USB ports FRDM-KL46Z with FRDM-CR20A Thread end device

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via P&E Micro, CMSIS-DAP, SEGGER J-Link:**

Thread Shell Interface interfaced to KL46Z UART (115200bps)

- **KL46Z USB port:**
  - Not available

14.5.2 FRDM-KL46Z with FRDM-CR20A Thread low-power end device

14.5.2.1 LEDs and switches FRDM-KL46Z with FRDM-CR20A Thread low-power end device

- **Factory Default State**
  - **LEDs**
    - FRDM-CR20A RGB LED2 blinking blue Device is in Factory Default idle state
    - FRDM-CR20A red LED1 Device is in Factory Default idle state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Short Press Join network
    - FRDM-KL46Z SW3, SW1 Short Press Join network
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset
    - FRDM-KL46Z SW3, SW1 Very Long Press (> 8 seconds) Factory reset
- **Network Joining State**
  - **LEDs**
    - FRDM-KL46Z D5 blinking red Device is in Joining state
    - FRDM-CR20A RGB LED 2 color cycle Device is in Joining state
  - **Switches**
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) Factory reset

        - FRDM-KL46Z SW3, SW1 Very Long Press (> 8 seconds) Factory reset
- **Low-power State**
  - **LEDs**
    - FRDM-CR20A RGB LED 2 Used for CoAP Application LED Control
    - FRDM-CR20A RGB LED2 toggling briefly Packet activity on wake up
  - **Switches**
    - FRDM-KL46Z SW1 Low-power Wakeup – Wake up state active for 5 seconds
- **Temporary Wake Up State (press FRDM-KL46Z SW1 while in Low-power State)**
  - **LEDs**
    - FRDM-CR20A RGB LED2 toggling briefly Packet activity
    - FRDM-CR20A RGB LED2 used for CoAP Application LED Control
  - **Switches**
    - FRDM-CR20A SW2 Short Press Send create data sink multicast announcement
    - FRDM-CR20A SW2 Long Press (2-3 seconds) Send release data sink multicast announcement
    - FRDM-CR20A SW1 Short Press Send CoAP temperature report multicast or to Data Sink
    - FRDM-CR20A SW1 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
    - FRDM-KL46Z SW3 Short Press Send CoAP LED ON with random RGB values
    - FRDM-KL46Z SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
    - FRDM-KL46Z SW1 Short Press Send CoAP LED OFF
    - FRDM-KL46Z SW1 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
    - FRDM-CR20A SW1, SW2 Very Long Press (> 8 seconds) – Factory reset
    - FRDM-KL46Z SW3, SW1 Very Long Press (> 8 seconds) – Factory reset

14.5.2.2 USB ports FRDM-KL46Z with FRDM-CR20A Thread end device

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via P&E Micro, CMSIS-DAP, SEGGER J-Link:**
    - Not available
- **KL46Z USB port:**
  - Not available

14.6 FRDM-KW41Z application configurations

NOTE

SW2 and SW5 are Touch Pads that are very sensitive and may accidently be triggered while pressing SW3 or SW4 push buttons. Ensure your finger does not inadvertently come in contact with those pads.

14.6.1 FRDM-KW41Z Thread router eligible device

14.6.1.1 LEDs and switches FRDM-KW41Z Thread router eligible device

- **Factory Default State**
  - **LEDs**
    - LED3 and LED4 blinking LED3 is red on REV A2 boards Device is in Factory Default idle state
  - **Switches**
    - SW2, SW3, SW4, SW5 Short Press Join network
    - SW2, SW3, SW4, SW5 Double Short Press Create network
    - SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - **LEDs**
    - LED3 blinking blue Device is in Create/Joining state
    - LED4 color cycle Device is in Create/Joining state
  - **Switches**
    - SW2, SW3, SW4, SW5 Short Press Create network
    - SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset
- **Connected State**
  - **LEDs**
    - LED3 blinking Device is requesting Router ID to the Leader
    - LED3 solid Device is an Active Router
    - LED3 off Device is a Router Eligible End Device (REED)
    - LED3 toggling briefly Packet activity
    - LED4 turns green after Network Create Node has become initial Leader
    - LED4 turns off after Network Joining Node is not a Leader
    - LED4 turns green during Network Operation Router has become a new network partition Leader
    - LED4 In all other use cases used for CoAP Application LED Control
  - **Switches**
    - SW5 Short Press Send create data sink multicast announcement
    - SW5 Long Press (2-3 seconds) Send release data sink multicast announcement
    - SW4 Short Press Send CoAP temperature report multicast or to Data Sink
    - SW4 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
    - SW3 Short Press Send CoAP LED ON with random RGB values
    - SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
    - SW2 Short Press Send CoAP LED OFF
    - SW2 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
    - SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset

14.6.1.2 USB ports FRDM-KW41Z Thread router eligible device

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via CMSIS-DAP, Segger J-Link:**

Thread Shell Interface interfaced to KW24D UART (115200bps)

14.6.2 FRDM-KW41Z Thread end device

14.6.2.1 LEDs and switches FRDM-KW41Z Thread end device

- **Factory Default State**
  - **LEDs**
    - LED3 and LED4 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW2, SW3, SW4, SW5 Short Press Join network
    - SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - **LEDs**
    - LED3 blinking blue Device is in Joining state
    - LED4 color cycle Device is in Joining state
  - **Switches**
    - SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset
- **Connected State**
  - **LEDs**
    - LED3 off Device is an End Device
    - LED3 toggling briefly Packet activity
    - LED4 In all other use cases used for CoAP Application LED Control
  - **Switches**
    - SW5 Short Press Send create data sink multicast announcement
    - SW5 Long Press (2-3 seconds) Send release data sink multicast announcement
    - SW4 Short Press Send CoAP temperature report multicast or to Data Sink
    - SW4 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
    - SW3 Short Press Send CoAP LED ON with random RGB values
    - SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
    - SW2 Short Press Send CoAP LED OFF
    - SW2 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
    - SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset

14.6.2.2 USB ports FRDM-KW41Z Thread end device

- **OpenLink/OpenSDA USB:**

- — **Serial COM port via CMSIS-DAP, Segger J-Link:**

Thread Shell Interface interfaced to KW41Z UART (115200bps)

14.6.3 FRDM-KW41Z Thread low-power end device

14.6.3.1 LEDs and switches FRDM-KW41Z Thread low-power end device

- **Factory Default State**
  - — **LEDs**
    - ◦ LED3 and LED4 blinking blue Device is in Factory Default idle state
  - — **Switches**
    - ◦ SW2, SW3, SW4, SW5 Short Press Join network
    - ◦ SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - — **LEDs**
    - ◦ LED3 blinking blue Device is in Joining state
    - ◦ LED4 color cycle Device is in Joining state
  - — **Switches**
    - ◦ SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset
- **Low-power State**
  - — **LEDs**
    - ◦ LED3 off Device is an End Device
    - ◦ LED3 toggling briefly Packet activity on wake up
    - ◦ LED4 LED toggling when receiving a CoAP toggling message
  - — **Switches**
    - ◦ SW4 Short Press Low-power Wakeup – Wake up state active for 5 seconds
- **Temporary Wake Up State (press SW1 in Low-power State)**
  - — **LEDs**
    - ◦ LED3 off Device is an End Device
    - ◦ LED3 toggling briefly Packet activity on wake up
    - ◦ LED4 LED toggling when receiving a CoAP toggling message
  - — **Switches**
    - ◦ SW5 Short Press Send create data sink multicast announcement
    - ◦ SW5 Long Press (2-3 seconds) Send release data sink multicast announcement
    - ◦ SW4 Short Press Send CoAP temperature report multicast or to Data Sink
    - ◦ SW4 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
    - ◦ SW3 Short Press Send CoAP LED ON with random RGB values
    - ◦ SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)

- SW2 Short Press Send CoAP LED OFF
- SW2 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
- SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset

14.6.3.2 USB ports FRDM-KW41Z Thread low-power end device

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via CMSIS-DAP, Segger J-Link:**

Not enabled

14.6.4 FRDM-KW41Z Thread Host controlled interface

14.6.4.1 LEDs and switches FRDM-KW41Z Thread Host controlled interface

- **Factory Default State**
  - **LEDs**
    - LED3 and LED4 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW2, SW3, SW4, SW5 Short Press Join network
    - SW2, SW3, SW4, SW5 Double Short Press Create network
    - SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - **LEDs**
    - LED3 blinking blue Device is in Create/Joining state
    - LED4 color cycle Device is in Create/Joining state
  - **Switches**
    - SW2, SW3, SW4, SW5 Short Press Create network
    - SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset
- **Connected State**
  - **LEDs**
    - LED3 blinking Device is requesting Router ID to the Leader
    - LED3 solid Device is an Active Router
    - LED3 off Device is a Router Eligible End Device (REED)
    - LED3 toggling briefly Packet activity
    - LED4 turns green after Network Create Node has become initial Leader
    - LED4 turns off after Network Joining Node is not a Leader
    - LED4 turns green during Network Operation Router has become a new network partition Leader
    - LED4 In all other use cases used for CoAP Application LED Control
  - **Switches**

- SW5 Short Press Send create data sink multicast announcement
- SW5 Long Press (2-3 seconds) Send release data sink multicast announcement
- SW4 Short Press Send CoAP temperature report multicast or to Data Sink
- SW4 Long Press (2-3 seconds) Revert local temperature and LED control to use multicast
- SW3 Short Press Send CoAP LED ON with random RGB values
- SW3 Long Press (2-3 seconds) Send CoAP LED Cycle (Color)
- SW2 Short Press Send CoAP LED OFF
- SW2 Long Press (2-3 seconds) Send CoAP LED Blink/Flash
- SW2, SW3, SW4, SW5 Very Long Press (> 8 seconds) – Factory reset

14.6.4.2 USB ports FRDM-KW41Z Thread Host controlled interface

- **OpenLink/OpenSDA USB:**
  - **Serial COM port via CMSIS-DAP, Segger J-Link:**

Thread Host Control Interface (THCI) interfaced to KW41Z UART (115200bps)

14.7 USB-KW41Z application configurations

14.7.1 USB-KW41Z Thread router eligible device

14.7.1.1 LEDs and switches USB-KW41Z Thread router eligible device

- **Factory Default State**
  - **LEDs**
    - D2, D3 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1 Short Press Join network
    - SW1 Double Short Press Create network
    - SW1 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - **LEDs**
    - D2, D3 cycle/sequence Device is in Create/Joining state
  - **Switches**
    - SW1 Short Press Create network
    - SW1 Very Long Press (> 8 seconds) – Factory reset
- **Connected State**
  - **LEDs**
    - D2 blinking Device is requesting Router ID to the Leader
    - D2 solid Device is an Active Router

- D2 off Device is a Router Eligible End Device (REED)
- D3 Used for CoAP Application LED Control
- D3 toggling briefly Packet activity

— **Switches**

- SW1 Short Press Send create data sink multicast announcement
- SW1 Long Press (2-3 seconds) Send release data sink multicast announcement

14.7.1.2 USB ports USB-KW41Z Thread router eligible device

- **OpenLink/OpenSDA USB:**

— **Serial COM port via CMSIS-DAP, Segger J-Link:**

Thread Shell Interface interfaced to KW41Z UART (115200 bps)

14.7.2 USB-KW41Z Thread end device

14.7.2.1 LEDs and switches USB-KW41Z Thread end device

- **Factory Default State**

— **LEDs**

- D2, D3 blinking blue Device is in Factory Default idle state

— **Switches**

- SW1 Short Press Join network
- SW1 Very Long Press (> 8 seconds) – Factory reset

- **Network Joining State**

— **LEDs**

- D2, D3 cycle/sequence Device is in Joining state

— **Switches**

- SW1 Very Long Press (> 8 seconds) – Factory reset

- **Connected State**

— **LEDs**

- D2 off Device is an End Device
- D3 Used for CoAP Application LED Control
- D3 toggling briefly Packet activity

— **Switches**

- SW1 Short Press Send create data sink multicast announcement
- SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
- SW1 (> 8 seconds) – Factory reset

14.7.2.2 USB ports USB-KW41Z Thread end device

- **OpenLink/OpenSDA Mini-USB:**

- — **SSerial COM port via CMSIS-DAP, Segger J-Link:**

Thread Shell Interface interfaced to KW41Z UART (115200bps)

14.7.3 USB-KW41Z Thread low-power end device

14.7.3.1 LEDs and switches USB-KW41Z Thread low-power end device

- **Factory Default State**
  - — **LEDs**
    - ◦ D2, D3 blinking blue Device is in Factory Default idle state
  - — **Switches**
    - ◦ SW1 Short Press Join network
    - ◦ SW1 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - — **LEDs**
    - ◦ D2, D3 cycle/sequence Device is in Joining state
  - — **Switches**
    - ◦ SW1 Very Long Press (> 8 seconds) – Factory reset
- **Low-power State**
  - — **LEDs**
    - ◦ D2 off Device is an End Device
    - ◦ D3 Used for CoAP Application LED Control
    - ◦ D3 toggling briefly Packet activity on wake up
  - — **Switches**
    - ◦ SW1 Low-power Wakeup – Wake up state active for 5 seconds
- **Temporary Wake Up State (press SW1 in Low-power State)**
  - — **LEDs**
    - ◦ D2 off Device is an End Device
    - ◦ D3 Used for CoAP Application LED Control
    - ◦ D3 toggling briefly Packet activity on wake up
  - — **Switches**
    - ◦ SW1 Short Press Send create data sink multicast announcement
    - ◦ SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
    - ◦ SW1 Very Long Press (> 8 seconds) – Factory reset

14.7.3.2 USB ports USB-KW41Z Thread low-power device

- **OpenLink/OpenSDA USB**
  - — Not enabled

14.7.4 USB-KW41Z Thread Host controlled interface

14.7.4.1 LEDs and switches USB-KW41Z Thread Host controlled interface

- **Factory Default State**
  - **LEDs**
    - D2 blinking blue Device is in Factory Default idle state
  - **Switches**
    - SW1 Short Press Join network
    - SW1 Double Short Press Create network
    - SW1 Very Long Press (> 8 seconds) – Factory reset
- **Network Joining State**
  - **LEDs**
    - D2, D3 cycle/sequence Device is in Create/Joining state
  - **Switches**
    - SW1 Short Press Create network
    - SW1 Very Long Press (> 8 seconds) – Factory reset
- **Connected State**
  - **LEDs**
    - D2 blinking Device is requesting Router ID to the Leader
    - D2 solid Device is an Active Router
    - D2 off Device is a Router Eligible End Device (REED)
    - D3 Used for CoAP Application LED Control
    - D3 toggling briefly Packet activity
  - **Switches**
    - SW1 Short Press Send create data sink multicast announcement
    - SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
    - SW1 Very Long Press (> 8 seconds) – Factory reset

14.7.4.2 USB ports USB-KW41Z Thread Host controlled interface

- **OpenLink/OpenSDA USB:**
  - **Thread Host Control Interface (THCI)** interfaced to KW41Z UART (115200bps)

14.7.5 USB-KW41Z Thread border router

14.7.5.1 LEDs and switches USB-KW41Z Thread border router

Factory Default State

LEDs

- D2, D3 blinking blue Device is in Factory Default idle state

Switches

- SW1 Short Press Join network
- SW1 Double Short Press Create network
- SW1 Very Long Press (> 8 seconds) – Factory reset

Network Joining State

LEDs

- D2, D3 cycle/sequence Device is in Create/Joining state

Switches

- SW1 Short Press Create network
- SW1 Very Long Press (> 8 seconds) – Factory reset

Connected State

LEDs

- D2 blinking Device is requesting Router ID to the Leader
- D2 solid Device is an Active Router
- D2 off Device is a Router Eligible End Device (REED)
- D3 Used for CoAP Application LED Control
- D3 toggling briefly Packet activity

Switches

- SW1 Short Press Send create data sink multicast announcement
- SW1 Long Press (2-3 seconds) Send release data sink multicast announcement
- SW1 Very Long Press (> 8 seconds) – Factory reset

14.7.5.2 USB ports USB-KW41Z Thread border router

KW41 USB port

Ethernet Emulation over USB CDC (RNDIS)

- ND Router with ULA prefix provisioning
- Thread Host Control Interface (THCI) transported as raw Ethernet frames

Chapter 15
Revision history

Table 19. Revision history on page 115 summarizes the revisions of this document.

Table 19. Revision history

Revision number	Date	Substantive changes
0	10/2015	Initial release
1	03/2016	Updated for Preview 5 release
2	04/2016	Updated for FRDM-KW41Z
3	08/2016	Updated for Thread KW41 Beta Release
4	09/2016	Updated for Thread KW41 GA Release
5	12/2016	Updated for Thread KW2x GA Release
6	03/2017	Updated for Thread KW41
7	01/2018	Updated for KW41 Maintenance Release
8	05/2018	Updates for K3S Beta Release.
9	04/2019	Updates for KW41Z Maintenance Release 3.

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

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

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

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, All other product or service names are the property of their respective owners. Arm, AMBA, Arm Powered, are registered trademarks of Arm Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.

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

Date of release: 08 April 2019
Document identifier: KTSDAUG

arm