UG10194

K32W1/MCXW71/MCXW72 ZigBee Demo Applications User Guide

Rev. 4.0 — 28 November 2024 **User guide**

Document information

Information	Content
Keywords	UG10194, K32W148-EVK board, FRDM-MCXW71 board, MCXW72-EVK board ZigBee devices, ZigBee 3.x network, ZigBee sample applications, MCUXpresso Integrated Design Environment (IDE), K32W1/MCXW71/MCXW72 Software Development Kit (SDK), ZigBee over-the-air (OTA), FRDM-MCXW7x board
Abstract	This document describes ZigBee sample applications to demonstrate the features and operation of the Base Device in a ZigBee 3.x network employing K32W1/MCXW71/MCXW72 device based microcontrollers.

1 Overview

This document applies to the K32W1/MCXW71/MCXW72 device ZigBee 3.x wireless microcontrollers, referred to as ZigBee devices throughout this document.

This document provides details of the ZigBee example applications. These examples demonstrate the features and operation of the base device in a ZigBee 3.x network that uses the NXP K32W1_MCXW71_MCXW72_ZDAUG microcontroller. The examples can be a starting point for developing real-world devices.

1.1 Related documentation

Table 1 lists the documents that can be referred for more information and developing custom applications based on this user guide.

Table 1. Related documents

Documents	Description	Link/how to obtain
ZigBee Base Device Behavior Specification Version 1.0	It provides a definition for the base device behavior specification, for devices operating on the ZigBee-PRO stack.	ZigBee Base Device Behavior Specification
Getting Started with the K32W148 Development Platform	It provides detailed instructions for installing the MCUXpresso SDK for the K32W148-EVK board.	Getting Started with the K32W148 Development Platform
Getting Started with the FRDM-MCXW71	It provides detailed instructions for installing the MCUXpresso SDK for the FRDM-MCXW71 board.	Getting Started with FRDM-MCXW71
Getting Started with the MCXW72-EVK	It provides detailed instructions for installing the MCUXpresso SDK for the MCXW72-EVK board.	Getting Started with the MCXW72 Development Platform
ZigBee 3.0 Stack User Guide (JN-UG-3130)	It provides information relating to the ZigBee 3.0 wireless networking protocol and its associated stack for implementation on NXP microcontrollers.	Contact an NXP field applications engineer (FAE) or sales representative
ZigBee 3.0 Devices User Guide (JN-UG-3131)	It introduces and provides details of the ZigBee Base Devices.	Contact an NXP field applications engineer (FAE) or sales representative
ZigBee 3.0 Cluster Library User Guide (JN-UG-3132)	It describes the NXP implementation of the ZigBee Cluster Library (ZCL) for the ZigBee 3.0 standard.	Contact an NXP field applications engineer (FAE) or sales representative
Core Utilities User Guide [JN-UG-3133]	It describes the device Core Utilities (JCU) that is used in wireless network applications for the NXP device-based microcontrollers.	Contact an NXP field applications engineer (FAE) or sales representative
ZigBee 3.0 Green Power User Guide [JN-UG-3134]	It describes the use of the NXP implementation of the Green Power feature for ZigBee 3.0 applications.	Contact an NXP field applications engineer (FAE) or sales representative

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

2 Introduction

A ZigBee 3.x wireless network comprises various ZigBee software devices that are implemented on hardware platforms to form nodes. These ZigBee examples are concerned with implementing the ZigBee Base device on the NXP ZigBee device (K32W1/MCXW71/MCXW72).

This document provides example implementations of the following ZigBee logical device types:

- Coordinator
- Router
- End device with Receiver always On
- End device with Receiver Off

The examples of the above device types are not real-world devices but provide the basic behavior required by the ZigBee Base Device Behavior Specification. These examples must serve as base templates for further development into real physical devices. The ZigBee Base Device is introduced and detailed in the *ZigBee 3.0 Devices User Guide* (document JN-UG-3131).

The ZigBee Base Device Behavior Specification provides definitions, procedures and methods for forming, joining, and maintaining ZigBee 3.x networks. It also defines the method for service discovery, which binds a client and server of an operational cluster to achieve the functionality of the physical devices. For more information on ZigBee 3.x networks general introduction, refer to the *ZigBee 3.0 Stack User Guide* (document JN-UG-3130).

3 Development environment

This section includes software and hardware requirements:

- Software
- Hardware

3.1 Software

To use the ZigBee examples, install the following software:

- MCUXpresso Integrated Design Environment (IDE)
- K32W1/MCXW71/MCXW72 ZigBee 3.0 Software Development Kit (SDK)
- Python 3 and lxml module
- SPSDK version 1.11.0 and crccheck module
  Note: *Both SPSDK version 1.11.0 and crccheck modules are required to generate OTA images using the NXP ZB OTA tool (`nxpzbota.py`).*

The MCUXpresso software and installation instructions are described in Getting Started with the K32W148 Development Platform.

Support for Zigbee packet sniffing is provided by using the `Sniffer_1000000baud_8N1_NoFlowControl` sniffer binary available in the `tools/sniffer` K32W061 SDK. The binary must be flashed on a K32W0 board using either of the following two methods:

- Using the *Getting Started with MCUXpresso SDK for K32W061* (document MCUXSDKK32W061GSUG). For more information, see the section "Building and Flashing the Application". In this case, the user must also install the K32W061 SDK.
- Using the firmware loader from NXP Test Tool.

Also, ensure to install the following tools:

- Kinetis Protocol Analyzer Adapter 2.0.3.1 or newer and Wireshark

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

- J-Link software, which can be downloaded from J-Link / J-Trace Downloads

The wireless microcontroller-specific resources and documentation are available via the MCUXpresso to authorized users.

3.2 Hardware

NXP enables the development of ZigBee 3.x applications by providing supported hardware kits. The following boards provide a platform for running the Zigbee applications:

- K32W148-EVK board
- FRDM-MCXW71 board
- MCXW72-EVK board

4 Examples

Table 2 lists the example applications provided.

Table 2. Example applications and device types

Application	Device type
Coordinator	Coordinator/Trust center
Router	Router
End Device RX On	End Device (Non-sleeping)
End Device RX Off when Idle	End Device (Sleeping)

The examples are provided as part of the wireless examples for ZigBee. The binaries generated by these examples are targeted to run on the K32W148-EVK board, FRDM-MCXW71 board, or MCXW72-EVK board.

4.1 K32W148-EVK board

The K32W148-EVK board is used for the ZigBee examples described in this document. The important LEDs and user control buttons are highlighted, as shown in Figure 1.

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

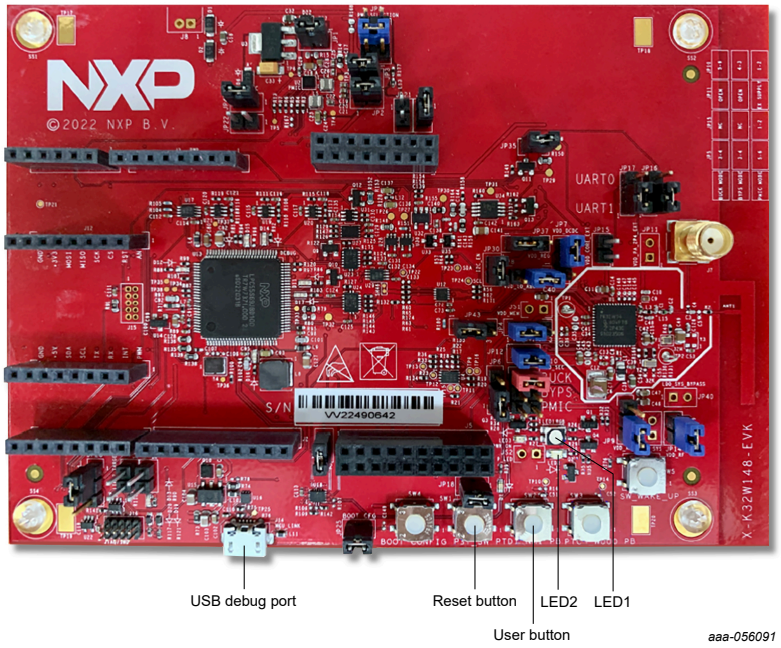

Figure 1. K32W148-EVK board

4.2 FRDM-MCXW7x board

The FRDM-MCXW7x board is used for the ZigBee examples described in this document. The important LEDs and user control buttons are highlighted, as shown in Figure 2.

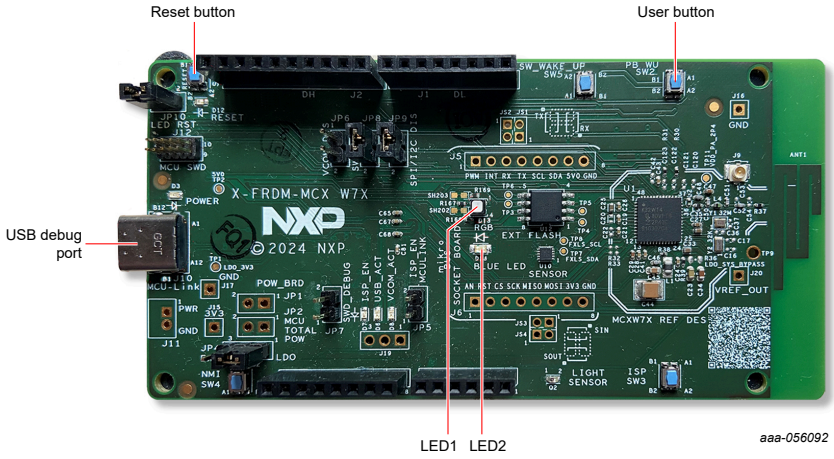

Figure 2. FRDM-MCXW7x board

4.3 FRDM-MCXW71/2 board

The FRDM-MCXW71/2 board is used for the ZigBee examples described in this document. The important LEDs and user control buttons are highlighted, as shown in Figure 3.

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

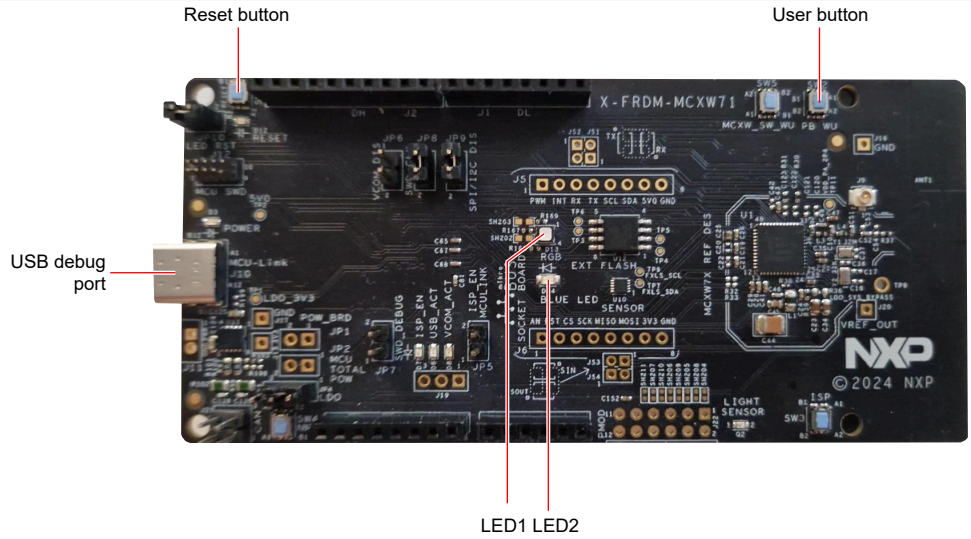

Figure 3. FRDM-MCXW71/2 board

5 Running the demonstration examples

This section describes how to use the supplied pre-built binaries to run the example applications on a ZigBee device. All the applications run on the K32W148-EVK board or FRDM-MCXW71 board, or MCXW72-EVK board. The examples do not run on other development kits. All the ZigBee wireless demo applications are configured to use a default channel 12. The pre-processor configuration uses the setting `SINGLE_CHANNEL=12`, which can be overwritten to change it to a different channel.

The ZigBee examples work in the Centralized (Trust center) network only, and all devices are expected to join with the ZigBee09 key. Once joined, the devices then automatically negotiate a new TCLK key.

For details of the differences between a Centralized (Trust center) network and a Distributed network, see the *ZigBee Devices User Guide* (document JN-UG-3114).

5.1 Loading the applications

Table 3 lists the binaries generated by the examples.

Table 3. Application binaries and hardware components

Hardware platform	Binary files
K32W148-EVK board	• `k32w148evk_zigbee_coordinator_bm.axf` • `k32w148evk_zigbee_router_bm.axf` • `k32w148evk_zigbee_ed_rx_on_bm.axf` • `k32w148evk_zigbee_ed_rx_off_bm.axf` • `k32w148evk_zigbee_coordinator_freertos.axf` • `k32w148evk_zigbee_router_freertos.axf` • `k32w148evk_zigbee_ed_rx_on_freertos.axf` • `k32w148evk_zigbee_ed_rx_off_freertos.axf`
FRDM-MCXW71 board	• `frdmmcxw71_zigbee_coordinator_bm.axf` • `frdmmcxw71_zigbee_router_bm.axf` • `frdmmcxw71_zigbee_ed_rx_on_bm.axf` • `frdmmcxw71_zigbee_ed_rx_off_bm.axf` • `frdmmcxw71_zigbee_coordinator_freertos.axf`

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

Table 3. Application binaries and hardware components*...continued*

Hardware platform	Binary files
	• frdmmcxw71_zigbee_router_freertos.axf • frdmmcxw71_zigbee_ed_rx_on_freertos.axf • frdmmcxw71_zigbee_ed_rx_off_freertos.axf
MCXW72-EVK board	• mcxw72evk_zigbee_coordinator_freertos.axf • mcxw72evk_zigbee_router_freertos.axf • mcxw72evk_zigbee_ed_rx_on_freertos.axf • mcxw72evk_zigbee_ed_rx_off_freertos.axf

To write the images to the board, perform the following steps:

1. Use the J-Link utility.
2. Download the J-Link from J-Link / J-Trace Downloads.
3. Plug the K32W148-EVK board (or FRDM-MCXW71 board, or MCXW72-EVK board) to the USB port (do not keep the **SW4** button pressed while doing this step).
4. Create a `commands_script` file with the following content (change the application name as necessary):

```
Reset
Halt
Erase
LoadFile <PATH_TO_AXF>
Reset
Go
Quit
```

5. Copy the application and `commands_script` in the same folder where the J-Link executable is placed.
6. Run the following code:

```
JLink.exe/JLinkExe (linux) -<DEVICE_NAME> -if SWD -speed 4000 -autoconnect 1
 -CommanderScript commands_script
```

where `DEVICE_NAME` can be:
- K32W1480 for K32W1
- MCXW716 for MCXW71
- MCXW727C_CORE0 for MCXW72

After completing the above steps, reset the board or module to run the application.

5.2 Coordinator functionality

The functionality of the Coordinator application is described as follows:

- The Coordinator is responsible for initially forming the network. It manages other devices that can join the network via the Trust center functionality. It distributes security materials to those devices that are allowed to join. The Coordinator supports the mandatory clusters and features of the Base Device as defined in the *ZigBee Base Device Behavior Specification*.
- The Coordinator also supports the On/Off Cluster as a client for demonstrating the "Finding and Binding" functionality.
- The serial commands issued from a terminal program control the Coordinator. The terminal program runs on a PC connected to the Zigbee device through a USB connection. The Coordinator application is configured to communicate with the following configuration:
  - Baud rate = 115200
  - Data = 8 bits
  - Stop = 1 bit

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

– Parity = None
– Flow control = None

The serial interface is not case-sensitive. For a summary of the serial interface, refer to Summary of Serial Interface Commands.

5.2.1 Forming a network

A network can be formed from a factory-new Coordinator (Network Steering while not on a network) as follows:

1. Enter the `form` on the serial interface. The Coordinator then starts a network.
2. Using a ZigBee packet sniffer (running separately on a USB Dongle), validates the user regarding network status. The periodic "link status" messages must be present on the operational channel. This step is optional.

5.2.2 Allowing other nodes to join (Network Steering)

Once a network has been formed, it must be opened to allow other devices to join it, referred to as Network Steering while on a network. To initiate Network Steering, perform the following steps:

1. Enter the `steer` on the serial interface (Dongle or Carrier Board).
2. Then, the Coordinator broadcasts a Management Permit Join Request to the network to open the "permit join" window for 180 seconds.
3. The Network Steering process (for devices not on a network) can now be triggered on the devices that are to join the network.

5.2.3 Operating the device (Coordinator)

The operational functionality of this device in this demonstration is provided by the On/Off cluster. Enter the `toggle` in the serial interface (Carrier Board) to send an OnOff Toggle command to the bound device (in the Binding table).

5.2.4 Rejoining a network (Coordinator)

As a Coordinator, when this device is restarted in a state that is not factory-new, it resumes operation in its previous state. All applications, bindings, groups, and network parameters are preserved in non-volatile memory.

5.2.5 Performing a factory reset (Coordinator)

The Coordinator can be returned to its factory-new state, which erases all persistent data except the outgoing network frame counter.

To perform a factory reset, enter the following command on the serial interface:

```
factory reset
```

5.2.6 Summary of serial interface commands

The serial port connection to the Coordinator application is set up with the following configuration settings:

- Baud rate = 115200
- Data = 8 bits
- Stop = 1 bit
- Parity = None

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

The serial commands are not case-sensitive.

Table 4. Serial interface commands

Serial command	Action
`toggle`	Sends an On/Off Toggle command to bound devices
`steer`	Triggers Network Steering for a device on the network
`form`	Triggers network formation for a device not on a network
`find`	Triggers Finding and Binding as an initiator
`factory reset`	Factory resets the device, erasing persistent data
`soft reset`	Triggers a software reset (no loss of data)

5.3 Router functionality

For demonstrating the "Finding and Binding" functionality, the Router also supports the On/Off Cluster as a server.

5.3.1 Forming or joining a network (router)

The router can only join an existing network. If it does not find the network, it continues discovering the network until it can find a network to join.

5.3.2 Joining an existing network using network steering (Router)

A factory-new Router can join an existing ZigBee only when the network is opened to accept new joiners (Network Steering for a device on a network).

Joining an existing network using Network Steering is achieved as follows:

1. Trigger Network Steering on one of the devices already on the network (Coordinator or another Router in the same ZigBee network).
2. Then reset using the **RESET** button or power on the joining Router device.
3. As a result, the Router starts a network discovery and the associate process. Association is followed by an exchange of security materials and an update of the Trust center link key (if joining a Centralized Trust center network).
4. By power cycling, the join can be retried if it fails.

5.3.3 Allowing other devices to join the network (Router)

Once the Router is part of a network, the network must be opened to allow other devices to join (Network Steering while on a network).

To allow other devices to join, perform the following steps:

1. Press the **USER** button on the K32W148-EVK board or FRDM-MCXW71 board, or MCXW72-EVK board. The same button is also used to start "Finding and Binding", described in Binding devices.
2. Then, the Router broadcasts a Management Permit Join Request to the network to open the "permit join" window for 180 seconds. The Network Steering process (for devices not on a network) can now be triggered on the devices that are to join the network.

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

5.3.4 Operating the device (router)

The operational functionality of this device in this demonstration is provided by the On/Off cluster. As the device supports the On/Off cluster server, its operation is passive, and it responds to commands sent by bound devices. It responds to an OnOff Toggle command from a bound controller device, by toggling the LED1 on the K32W148-EVK board or FRDM-MCXW71 board, or MCXW72-EVK board.

5.3.5 Rejoining a network (router)

As a Router, when this device is restarted in a state, which is not factory-new, it resumes operation in its previous state. All applications, binding, group, and network parameters are preserved within the non-volatile memory of the device.

5.3.6 Performing a factory reset (router)

The Router can be returned to its factory-new state (erasing all persistent data except the outgoing network frame counter) as follows:

- Hold down the **USER** button and press the **RESET** button on the K32W148-EVK board or FRDM-MCXW71 board, or MCXW72-EVK board.

The Router then broadcasts a Leave Indication on the old network. It also deletes all persistent data (except the outgoing network frame counter) and performs a software reset.

The two supported over-the-air commands for removing a device from the network are as follows:

- Network Leave Request without rejoin
- ZDO Management Network Leave Request without rejoin

The Reset command of the Basic cluster causes the ZCL to be reset to its factory-new defaults, resetting all attributes and configured reports. It does not remove the device from the network. Therefore, all network parameters, groups, and bindings remain in place.

5.4 End Device functionality

The End Device is not capable of either forming a network or being a parent to other devices joining the network.

Two types of End Device are as follows:

- "RX On" End Devices, which are always ready to communicate in the network
- Sleepy "RX Off when Idle" End Devices, which can sleep for periods of time during which it cannot communicate

The End Device supports the mandatory clusters and features of the Base Device as defined in ZigBee Base Device Behavior Specification.

For demonstrating the "Finding and Binding" functionality, the End Device also supports the On/Off cluster as a client.

All communications to/from the End Device are passed through its parent Coordinator or Router. For an RX Off with the Idle End Device, communication is initiated from the End Device through Poll Requests. The parent device then buffers data for the child End Device for some time. During this period, the End Device must send periodic Poll Requests to its parent to receive any messages that are waiting for it.

For the RX On device, no poll is required and messages are sent directly from the parent. However, regular messages must be sent. Otherwise, the device can be timed out.

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

5.4.1 Joining an existing network using network steering (ED)

A factory-new End Device can join an existing network once the network opens to accept new joiners (Network Steering for a device on a network).

Joining an existing network using Network Steering is achieved as follows:

1. Trigger Network Steering on one of the devices already on the network.
2. Then, reset using the **RESET** button or power on the End device.

5.4.2 Operating the device (ED)

The operational functionality of this device in this demonstration is provided by the On/Off cluster. The device supports the On/Off cluster server. Therefore, its operation is passive and it responds to commands sent by bound devices. It responds to an OnOff Toggle command from a bound controller device, by toggling the **LED1** on the K32W148-EVK board or FRDM-MCXW71 board, or MCXW72-EVK board.

5.4.3 Rejoining a network (ED)

An End Device can be restarted in a state, which is not factory-new. In such a case, it automatically sends a Network Rejoin Request to re-establish contact with its previous parent. If this fails, it then tries to join any Router on the network that can host it. The End Device attempts to rejoin when powered on and after it wakes from the deep sleep state. All the application, binding, group, and network parameters are preserved in non-volatile memory.

5.4.4 Performing a factory reset (ED)

The End Device can be returned to its factory-new state (erasing all persistent data except the outgoing network frame counter) as follows:

- Hold down the **USER** button and press the **RESET** button on the K32W148-EVK board or FRDM-MCXW71 board, or MCXW72-EVK board.

The End Device then unicasts a Leave Indication to its parent. The parent then broadcasts this message again to the old network. The End Device deletes all persistent data (other than the outgoing network frame counter) and performs a software reset.

There are two supported over-the-air commands for removing a device from the network as follows:

- Network Leave Request without rejoin
- ZDO Management Network Leave Request without rejoin

The Reset command of the Basic cluster causes the ZCL to be reset to its factory-state defaults. It also resets all attributes and configured reports. This step does not remove the device from the network and all network parameters, groups, and bindings remain in place.

5.5 Binding devices

The Router and End Device support the On/Off cluster as a server and implement the "Finding and Binding" process as a target.

To trigger "Finding and Binding" as a target, perform the following steps:

1. Press the **USER** button on the K32W148-EVK board or FRDM-MCXW71 board, or MCXW72-EVK board of the target device. The same button is used to start Network Steering, described in Allowing Other Devices to Join the Network.
2. Start "Finding and Binding" on the initiator device.

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

This step causes the End Device or Router to self-identify for 180 seconds. In this duration, the initiator tries to find the identifying devices, queries their capabilities, and creates bindings on the devices with matching operational clusters. As part of this process, the Route or End Device can receive an `Add Group` command and/or a `Binding Request` command.

Reporting is a mandatory feature in ZigBee 3.x. The Router and End Device supports the On/Off cluster as a server and the OnOff attribute of this cluster is a reportable attribute as defined in ZigBee Base Device Behavior Specification. The Router and End Device hold a default configuration for reporting the state of the OnOff attribute. Once a device wishing to receive these periodic and on-change reports creates a remote binding, the Router starts to send reports to this bound device. The frequency of the reports depends on the default report configuration of the individual target device; 60 seconds in this case. The device receiving the reports can request the change by sending a `Report Configuration` command.

6 ZigBee over-the-air upgrade

An over-the-air (OTA) upgrade involves transferring a new firmware image to a device already installed and operational within a ZigBee network. This functionality is provided by the OTA upgrade cluster. To upgrade the devices on a network, two functional elements are required as follows:

- **OTA Server:** First, the network must host an OTA server, which receives new OTA images from manufacturers and advertise the OTA image details to the network. Then, it must deliver the new image to the requested devices.
- **OTA Clients:** The second requirement is for OTA clients, which are on the network devices that can be updated. These devices periodically interrogate the OTA server for details of the firmware images available. If a client finds a suitable upgrade image on the server, it starts to request this image, storing each part as it is received. Once the full image has been received, it validates, and the device boots to run the new image.

The clients pull down the new images, requesting each block in turn and filling in the gaps. The server never pushes the images onto the network.

6.1 Overview

Support for the OTA upgrade cluster as a client has been included for the Router and End Device.

Note: *By default, all the devices only support encrypted OTA.*

The internal flash memory is used to store the upgraded image by default.

Table 5 shows that the initial client binaries to be programmed into the K32W1, MCXW71, and MCXW72 devices are version 1 files.

Table 5. Version 1 files

Hardware platform	Binary files
K32W148-EVK board	• `k32w148evk_zigbee_router_bm_v1.axf` • `k32w148evk_zigbee_ed_rx_on_bm_v1.axf` • `k32w148evk_zigbee_ed_rx_off_bm_v1.axf` • `k32w148evk_zigbee_router_freertos_v1.axf` • `k32w148evk_zigbee_ed_rx_on_freertos_v1.axf` • `k32w148evk_zigbee_ed_rx_off_freertos_v1.axf`
FRDM-MCXW71 board	• `frdmmcxw71_zigbee_router_bm_v1.axf` • `frdmmcxw71_zigbee_ed_rx_on_bm_v1.axf` • `frdmmcxw71_zigbee_ed_rx_off_bm_v1.axf` • `frdmmcxw71_zigbee_router_freertos_v1.axf` • `frdmmcxw71_zigbee_ed_rx_on_freertos_v1.axf`

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

Table 5. Version 1 files...*continued*

Hardware platform	Binary files
	• frdmmcxw71_zigbee_ed_rx_off_freertos_v1.axf
MCXW72-EVK board	• mcxw72evk_zigbee_router_freertos_v1.axf • mcxw72evk_zigbee_ed_rx_on_freertos_v1.axf • mcxw72evk_zigbee_ed_rx_off_freertos_v1.axf

Table 6 shows that the OTA images are the V2/V3.ota files.

Table 6. V2/V3 OTA files

Hardware platform	OTA files
K32W148-EVK board	• k32w148evk_zigbee_router_bm_v2.ota • k32w148evk_zigbee_router_bm_v3.ota • k32w148evk_zigbee_ed_rx_on_bm_v2.ota • k32w148evk_zigbee_ed_rx_on_bm_v3.ota • k32w148evk_zigbee_ed_rx_off_bm_v2.ota • k32w148evk_zigbee_ed_rx_off_bm_v3.ota • k32w148evk_zigbee_router_freertos_v2.ota • k32w148evk_zigbee_router_freertos_v3.ota • k32w148evk_zigbee_ed_rx_on_freertos_v2.ota • k32w148evk_zigbee_ed_rx_on_freertos_v3.ota • k32w148evk_zigbee_ed_rx_off_freertos_v2.ota • k32w148evk_zigbee_ed_rx_off_freertos_v3.ota
FRDM-MCXW71 board	• frdmmcxw71_zigbee_router_bm_v2.ota • frdmmcxw71_zigbee_router_bm_v3.ota • frdmmcxw71_zigbee_ed_rx_on_bm_v2.ota • frdmmcxw71_zigbee_ed_rx_on_bm_v3.ota • frdmmcxw71_zigbee_ed_rx_off_bm_v2.ota • frdmmcxw71_zigbee_ed_rx_off_bm_v3.ota • frdmmcxw71_zigbee_router_freertos_v2.ota • frdmmcxw71_zigbee_router_freertos_v3.ota • frdmmcxw71_zigbee_ed_rx_on_freertos_v2.ota • frdmmcxw71_zigbee_ed_rx_on_freertos_v3.ota • frdmmcxw71_zigbee_ed_rx_off_freertos_v2.ota • frdmmcxw71_zigbee_ed_rx_off_freertos_v3.ota
MCXW72-EVK board	• mcxw72evk_zigbee_router_freertos_v2.ota • mcxw72evk_zigbee_router_freertos_v3.ota • mcxw72evk_zigbee_ed_rx_on_freertos_v2.ota • mcxw72evk_zigbee_ed_rx_on_freertos_v3.ota • mcxw72evk_zigbee_ed_rx_off_freertos_v2.ota • mcxw72evk_zigbee_ed_rx_off_freertos_v3.ota

6.2 OTA upgrade operation

To add an image to the coordinator, the OTA images must be programmed. To program the OTA images, perform the following steps:

1. Use the J-Link utility.
2. Download J-Link from J-Link / J-Trace Downloads.

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

3. Plug the K32W148-EVK board or FRDM-MCXW71 board, or MCXW72-EVK board to the USB port (no need to keep the **SW4** button pressed while doing this step).
4. Create a `commands_script` file with the following content (change the application name as necessary):

```
Reset
Halt
LoadBin <OTA_ADDRESS> 0x7A000
Reset
Go
Quit
```

   Where `OTA_ADDRESS` value:
   - for K32W1/FRDM-MCXW71 OTA_ADDRESS = 0x7A000
   - for MCXW72-EVK OTA_ADDRESS = 0xFA000

   Note: *If J-Link fails to recognize the `.ota` file, rename it to `.bin` and retry.*
5. Copy the application and `commands_script` in the same folder where the J-Link executable is placed.
6. Run the following code:

```
JLink.exe/JLinkExe (linux) -<DEVICE_NAME> -if SWD -speed 4000 -autoconnect 1
 -CommanderScript commands_script
```

   Where `DEVICE_NAME` can be:
   - K32W1480 for K32W1
   - MCXW716 for MCXW71
   - MCXW727C_CORE0 for MCXW72

When adding an image to a non-factory new coordinator, care must be taken not to use the Erase command to erase the flash.

Any devices with OTA clients in the network periodically send match descriptor requests to find an OTA server. Once a server responds, it then sends an IEEE address request to confirm the address details. The clients then periodically send OTA Image Requests to determine whether the server is hosting an image for that client device. In response to the Image Request, the server returns details of the image that it currently hosts: Manufacturer code, Image tag, and Version number. The client checks these credentials and decides whether it requires this image. If it does not, it queries the server again at the next query interval. If the client does require the image, it starts to issue Block Requests to the server to get the new image. Once all blocks of the new image have been requested and received, the new image is verified. The older image is invalidated, the device reboots, and runs the new image. The client resumes periodically querying the server for new images.

The End Device, which is RX Off, is allowed to enter Sleep mode. It stays awake for 5 seconds and then sleeps for 1 second when not performing "Finding and Binding".

6.3 Image credentials

Four main elements of the OTA header are used to identify the image to enable the OTA client to decide whether it must download the image.

- **Manufacturer code:** This element is a 16-bit number that is a ZigBee-assigned identifier for each member company. In this application, this number has been set to 0x1037, which is the identifier for NXP. In the final product, this number must be changed to the identifier of the manufacturer. The OTA client compares the Manufacturer code in the advertised image with its own and the image downloads only if they match.
- **Image type:** This element is a manufacturer-specific 16-bit number in the range 0x0000 to 0xFFBF. It is used by the manufacturer to distinguish between devices. In this application, the Image type is normally set to the ZigBee Device Type. However, this application uses 0x0003 for the Router and End Device. The OTA client compares the advertised Image type with its own Image type. If the Image type matches, then the image is downloaded. The product designers are entirely free to implement an identification scheme of their own.

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

- **File version:** This element is a 32-bit number representing the version of the image. The OTA client compares the advertised version with its current version before deciding whether to download the image.
- **OTA header string:** This element is a 32-byte character string and its use is manufacturer-specific. In this application, the OTA client compares the string in the advertised image with its own string before accepting an image for download. If the strings match, then the image is accepted. In this way, the string can be used to provide extra detail for identifying images, such as hardware subtypes.

6.4 Upgrade and downgrade

The decision to accept an image following a query response is under the control of the application. The code, as supplied, accepts an upgrade or a downgrade. As long as the notified image has the right credentials and a version number, which is different from the current version number, the image is downloaded.

For example, if a client is running a v3 image and a server is loaded with a v2 image then the v2 image is downloaded. The application callbacks the function responsible for handling the image in the following two scenarios:

- When the client is required to accept only the upgraded images (v2 > v3 > v5)
- When the client is required to accept only the sequential upgrade images (v2 > v3 > v4 > v5)

.

7 LED indication table

This section includes the LED states for the following device types:

- Coordinator
- Router
- End device RX on
- End device RX off (sleepy device)

7.1 Coordinator

Table 7 lists the LED states for Coordinator.

Table 7. Coordinator

LED1	LED2	NOTES
OFF	OFF	The device is not on the network
OFF	BLINKING ON/OFF every 500 ms	Network Steering/permit join is active
OFF	BLINKING ON/OFF every 1 second	Find and Bind initiated and is still active
OFF	ON	The device is active
OFF	BLINKING ON/OFF every 250 ms	Both Network Steering/permit join and find and bind are active

7.2 Router

Table 8 lists the LED states for Router device type.

Table 8. Router

LED1	LED2	NOTES
OFF	OFF	The device is not on the network

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

Table 8. Router*...continued*

LED1	LED2	NOTES
OFF/ON (Current ON/OFF cluster status) or BLINKING ON/OFF (Identifying)	BLINKING ON/OFF every 250 ms	Network Steering/permit join is active
OFF/ON (Current ON/OFF cluster status) or BLINKING ON/OFF (Identifying)	BLINKING ON/OFF every 2 seconds	OTA aborted or failed
OFF/ON (Current ON/OFF cluster status) or BLINKING ON/OFF (Identifying)	BLINKING ON/OFF every 500 ms	OTA in progress

7.3 End Device RX On

Table 9 lists the LED states for End Device RX On.

Table 9. End Device RX On

LED1	LED2	NOTES
OFF	OFF	The device is not on the network
OFF/ON (Current ON/OFF cluster status) or BLINKING ON/OFF (Identifying)	BLINKING ON/OFF every 250 ms	Find and Bind active
OFF/ON (Current ON/OFF cluster status) or BLINKING ON/OFF (Identifying)	BLINKING ON/OFF every 2 seconds	OTA aborted or failed
OFF/ON (Current ON/OFF cluster status) or BLINKING ON/OFF (Identifying)	BLINKING ON/OFF every 500 ms	OTA in progress

7.4 End Device RX Off (sleepy device)

Table 10 lists the LED states for End device RX Off.

Table 10. End Device RX Off (sleepy device)

LED1	LED2	NOTES
OFF	OFF	The device is not on the network
OFF/ON (Current ON/OFF cluster status) or BLINKING ON/OFF (Identifying)	ON	The device is not sleeping and is active
OFF/ON (Current ON/OFF cluster status) or BLINKING ON/OFF (Identifying)	BLINKING ON for 5 seconds and OFF for 1 second	The device is going through a sleep and wake cycle

8 Note about the source code in the document

Example code shown in this document has the following copyright and BSD-3-Clause license:

Copyright 2024 NXP Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials must be provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

9 Revision history

Table 11 summarizes revisions to this document.

Table 11. Revision history

Document ID	Release date	Description
UG10194 v.4.0	28 November 2024	• Updated document ID to UG10194 • Updated document title to K32W1/MCXW71/MCXW72 ZigBee Demo Applications User Guide • Updated the document for FRDM-MCXW71 board and MCXW72-EVK board reference • Updated Table 1 for FRDM-MCXW71 board and MCXW72-EVK board • Updated Hardware for FRDM-MCXW71 board and MCXW72-EVK board • Updated Software for K32W1/MCXW71/MCXW72 • Added FRDM-MCXW71/2 board • Updated Table 3 in Loading the applications • Updated Table 5 and Table 6 in Overview
K32W1_ZDAUG v.3.0	4 June 2024	• Updated the document with FRDM-MCXW7x board reference • Added FRDM-MCXW7x board • Updated Table 3 in Loading the applications • Updated Overview for "FRDM-MCXW7x"
K32W1_ZDAUG v.2.0	27 October 2023	• Added ZigBee over-the-air upgrade and LED indication table • Document updated to new style guide • Overall improvement performed for the document
K32W1_ZDAUG v.1.1	16 June 2023	Added support for "End Device with Receiver Off" in Table 2
K32W1_ZDAUG v.1.0	22 May 2023	Updated Software
K32W1_ZDAUG v.0	27 February 2023	Initial release for K32W1 platform

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

All information provided in this document is subject to legal disclaimers.

© 2024 NXP B.V. All rights reserved.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

J-Link — is a trademark of SEGGER Microcontroller GmbH.

Kinetis — is a trademark of NXP B.V.

Matter, Zigbee — are developed by the Connectivity Standards Alliance. The Alliance's Brands and all goodwill associated therewith, are the exclusive property of the Alliance.

Contents

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

© 2024 NXP B.V. **All rights reserved.**