QSG185: Z-Wave 800 Dev Kit Thunderboard Quick Start Guide with Unify

The Z-Wave 800 Thunderboard BRD2603A Radio Board is an excellent starting point to get familiar with the ZGM230S Z-Wave 800 SiP Module. The board is a small and cost-effective, feature-rich prototype and development platform based on the ZGM230S SiP Module.

It has been designed to inspire customers to make energy-friendly, battery-operated and secure connected IoT devices using Z-Wave MESH and Z-Wave Long Range protocols. Customers can develop wide range command, control, and monitoring applications utilizing numerous sensors, buttons, and LEDs on this board. A built-in SEGGER J-Link debugger ensures easy debugging using a USB Type-C cable.

The Z-Wave 800 Dev Kit is supported in Simplicity Studio and it ships with Unify SDK, a portable environment that works with any Z-Wave end device and provides a flying start, out-of-box experience for Z-Wave applications. This Quick Start Guide describes the steps to get up and running quickly with the Dev Kit with Unify environment. The portable runtime comes as an executable CLI, together with Unify, a docker-compose file, and a tool which can map a Silicon Labs device with a docker environment on Windows, MacOS, or Linux.

KEY FEATURES

- ZGM230S Z-Wave SiP Module with 512 kB Flash, 64 kB RAM
- +14 dBm TX power
- SMA antenna connector (863-925 MHz)
- Advanced Energy Monitor
- Virtual COM port
- SEGGER J-Link on-board debugger
- RGB LED, two single color LEDs, and two push buttons
- CR2032 coin cell battery support
- 20-pin 2.54 mm breakout pads
- Mini Simplicity connector

ON-BOARD SENSORS

- Relative humidity and temperature sensor
- Passive Infrared sensor
- Hall effect sensor
- Ambient light sensor
- Pressure sensor
- LESENSE LC-sensor
- 6-axis Inertial sensor

SOFTWARE SUPPORT

- Simplicity Studio
- Unify SDK Z-Wave Protocol Controller

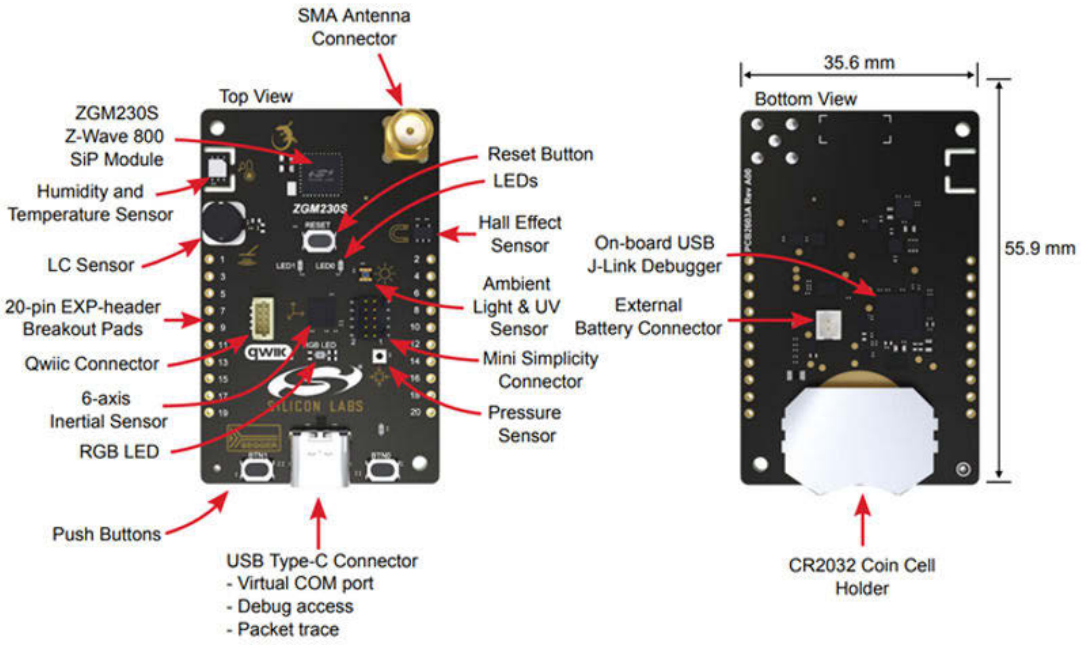
REQUIRED COMPONENTS

- Two Z-Wave Thunderboard Kits pre-programmed and labelled with "Z-Wave - NCP Serial API Controller"
- USB-C cable for controller (not included)
- Optional coin cell battery for end device (not included)
- Docker version > 20.10.12 installed.
- Docker-compose version > 2.5.1

1. Setting Up Your Z-Wave 800 Dev Kit

Step 1: Unboxing

1. Unbox Z-Wave 800 Dev Kit.



2. Connect the USB Type-C connector to the PC using the USB-C cable.

Step 2: Software Preparation

1. Download and install Docker Desktop:
   - Windows - https://docs.docker.com/desktop/windows/install/
   - MAC - https://docs.docker.com/desktop/mac/install/
   - Linux - https://docs.docker.com/desktop/linux/install/
2. Download and unzip the Unify Portable Environment, which enables the Z-Wave demo by running sample applications on 800 Dev Kit devices. This is a containerized Unify environment that can manage a Z-Wave controller, form a network with end devices, and provide a way to try the features of the applications.
   - Windows: http://www.silabs.com/documents/public/software/portable_runtime_windows.zip
   - Linux: http://www.silabs.com/documents/public/software/portable_runtime_linux.zip
   - Mac OS: http://www.silabs.com/documents/public/software/portable_runtime_macos.zip

Step 3: Flash Application

The development kit devices are shipped with the NCP Serial API Controller firmware by default. To try the demo, at least one controller (SerialAPI) and one end device application is needed. Therefore, one of the devices must be re-programmed before first use. The Unify Portable Runtime Environment package contains six built-in sample applications:

- SerialAPI – Controller NCP application
- SwitchOn/Off – End device application
- Multilevel Sensor – End device application
- SensorPIR – End Device application
- WallController – End Device application
- PowerStrip – End Device application

Follow these steps to flash the new application firmware to the dev kit:

1. Connect the device to the PC via USB-C cable.
2. Open a **new** terminal window in the unzipped **Unify Portable Environment** folder.
3. Run this command to flash Serial API Controller NCP application:

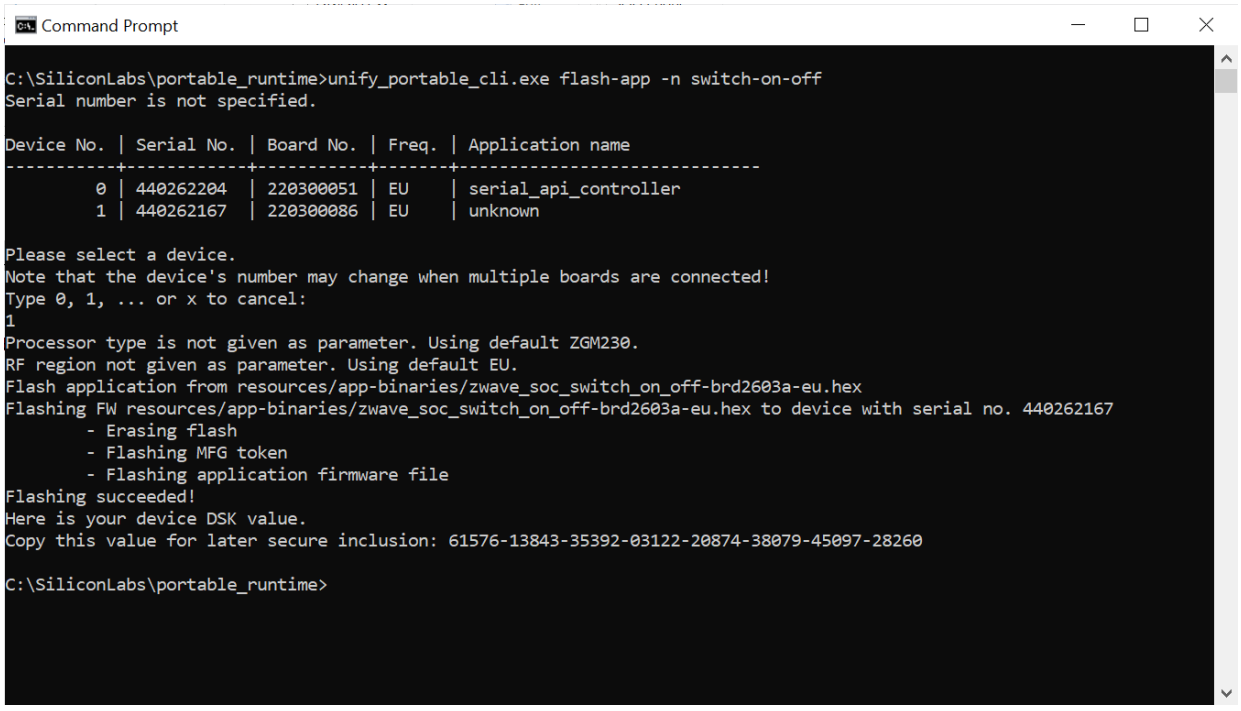
```
unify_portable_cli.exe flash-app -n serial-api
```

or this command to flash SwitchOn/Off application:

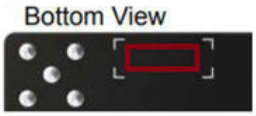
```
unify_portable_cli.exe flash-app -n switch-on-off
```

The environment performs some actions in the background, like resetting the device, flashing the new firmware, and setting the regional radio frequency.

If available, the Device Specific Key will be displayed at the end of the process.



If there are multiple devices connected via USB, you will be prompted to select one of the available devices for flashing. You can match the listed devices with your physical boards by examining the board number printed on the bottom of the Thunderboard.



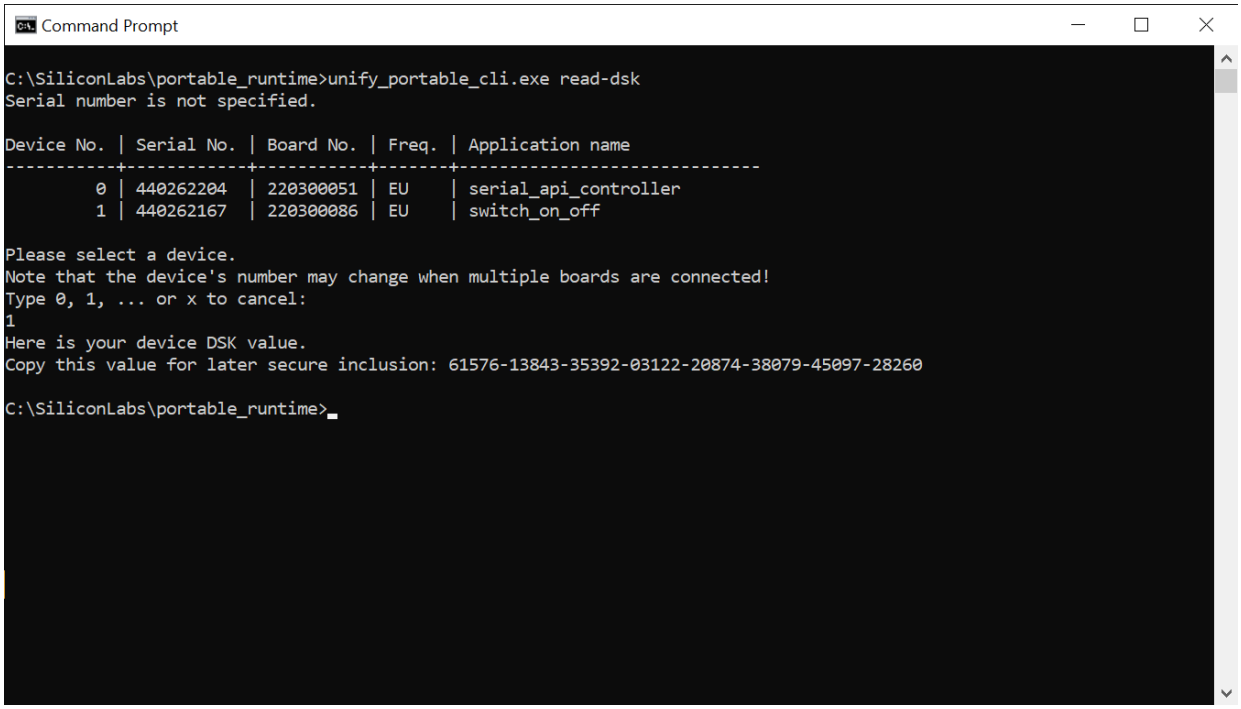
Alternatively, you can select a device directly in the command by specifying its J-Link serial number.

Step 3.1: Read Device Specific Key

To perform a secure inclusion, it is necessary to provide the device specific key (DSK) of the end device. It can be read by running the following command:

```
unify_portable_cli.exe read-dsk
```

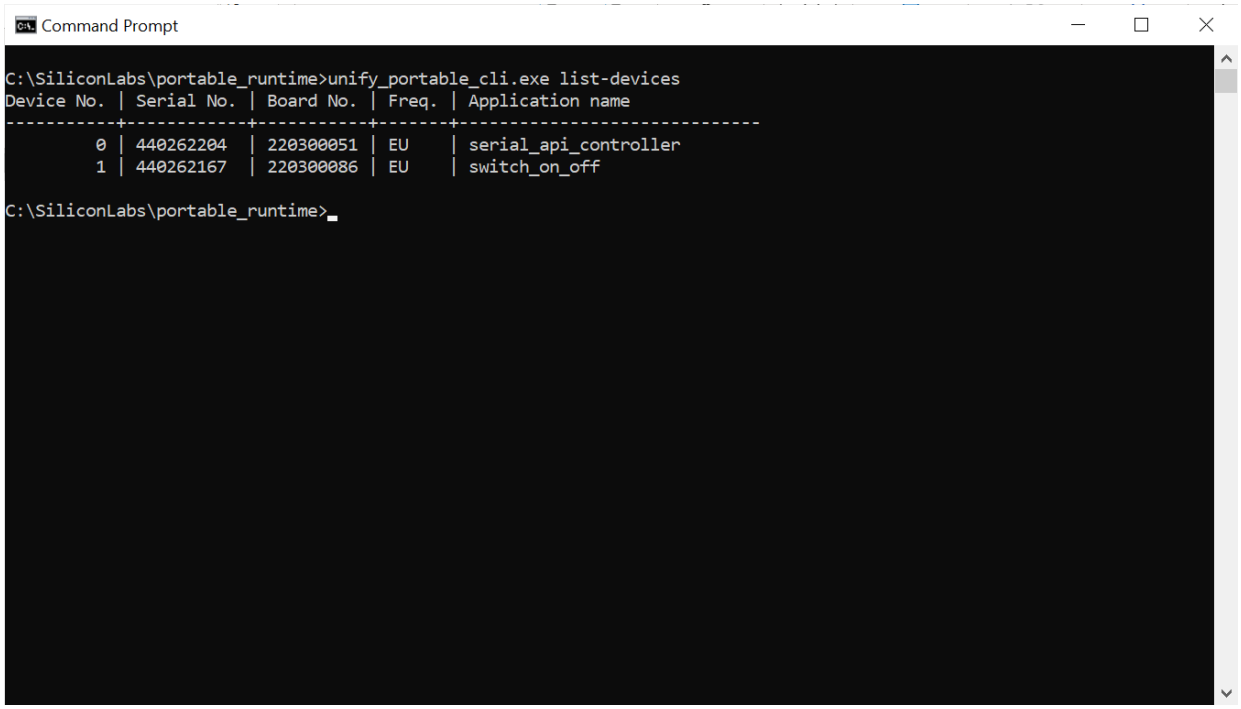
Select the end device from the list. The board can be identified by its board number.



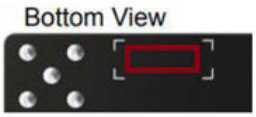
Step 3.2: List Connected Devices

To list the connected devices, run the following command:

```
unify_portable_cli.exe list-devices
```

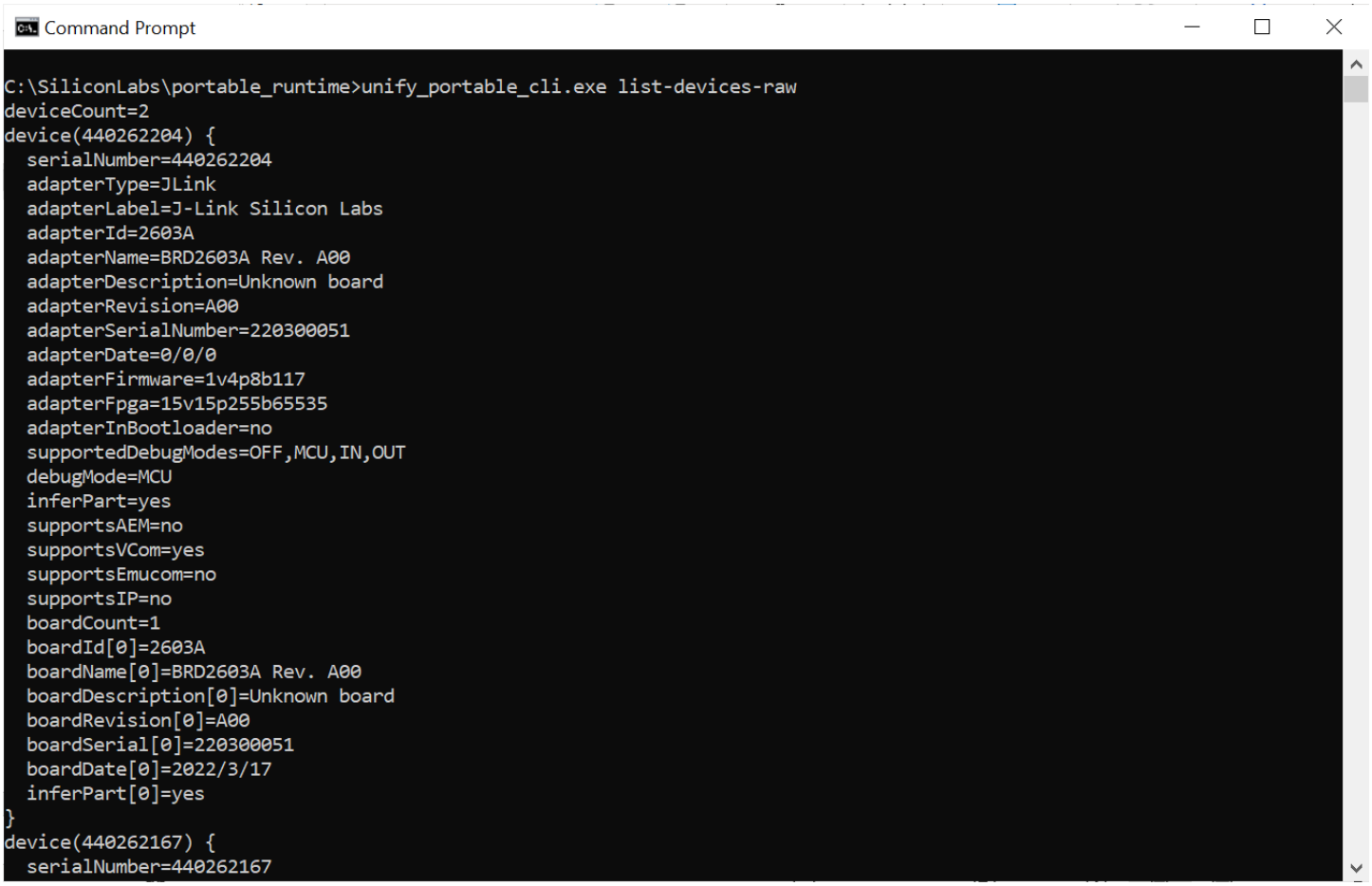


The output contains the Serial Number, Board Number, configured transmit region frequency, and application name of each device. The Board Number is also printed on the back of the development kit, next to the SMA antenna. The connected boards can be distinguished from each other by this number.



To get more information about the connected devices, run the following command:

```
unify_portable_cli.exe list-devices-raw
```

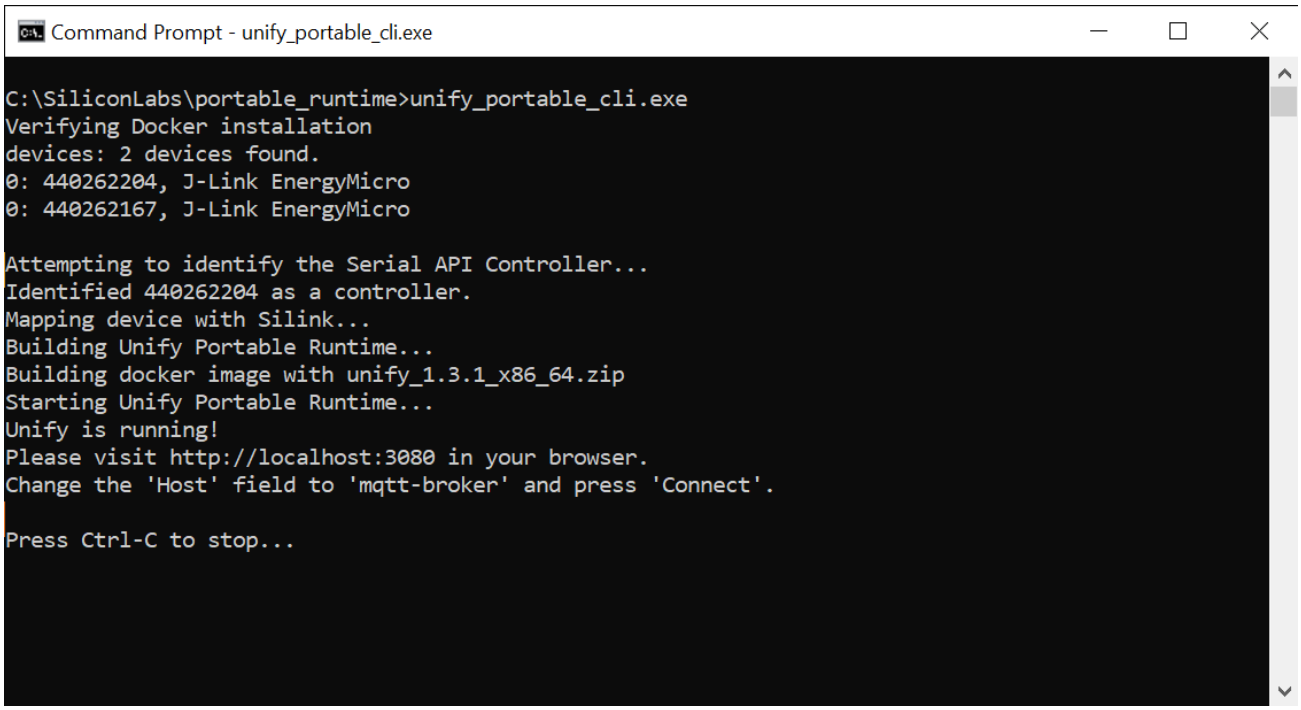

Command Prompt
C:\SiliconLabs\portable_runtime>unify_portable_cli.exe list-devices-raw
deviceCount=2
device(440262204) {
serialNumber=440262204
adapterType=JLink
adapterLabel=J-Link Silicon Labs
adapterId=2603A
adapterName=BRD2603A Rev. A00
adapterDescription=Unknown board
adapterRevision=A00
adapterSerialNumber=220300051
adapterDate=0/0/0
adapterFirmware=1v4p8b117
adapterFpga=15v15p255b65535
adapterInBootloader=no
supportedDebugModes=OFF,MCU,IN,OUT
debugMode=MCU
inferPart=yes
supportsAEM=no
supportsVCom=yes
supportsEmucom=no
supportsIP=no
boardCount=1
boardId[0]=2603A
boardName[0]=BRD2603A Rev. A00
boardDescription[0]=Unknown board
boardRevision[0]=A00
boardSerial[0]=220300051
boardDate[0]=2022/3/17
inferPart[0]=yes
}
device(440262167) {
serialNumber=440262167

Step 4: Start Unify Portable Environment

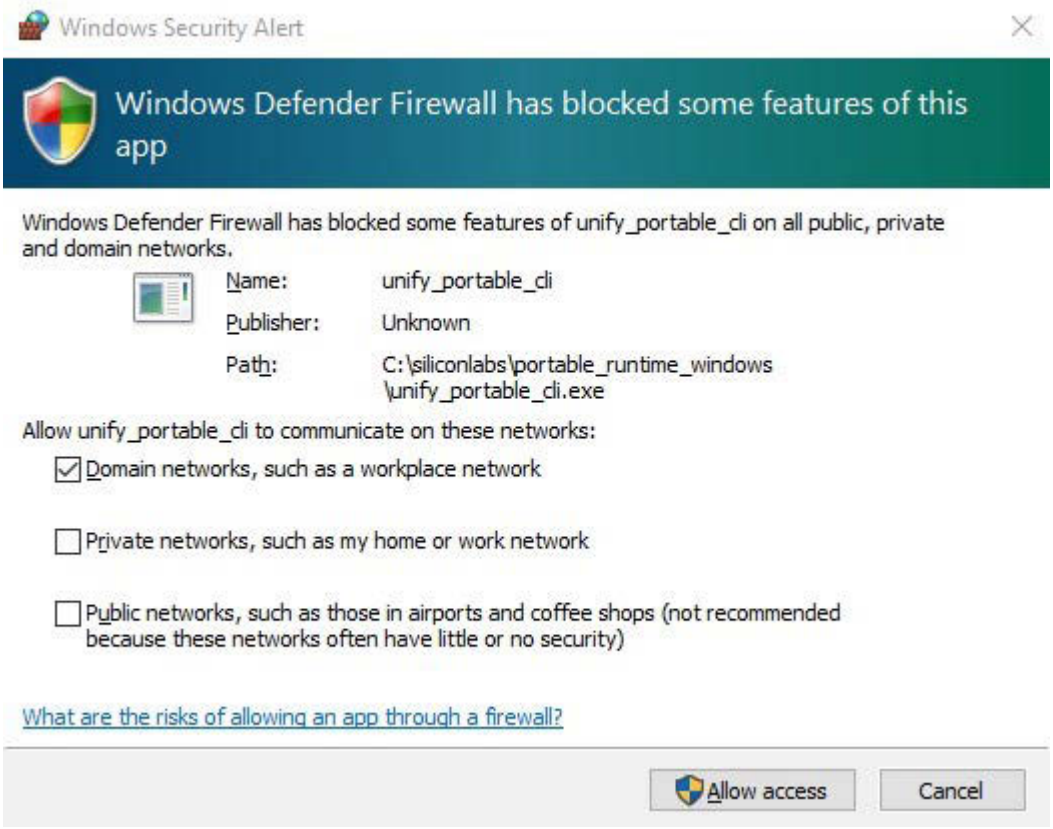
1. Connect the Controller device (Dev Kit with the Z-Wave - NCP Serial API Controller firmware) to the PC.
2. Run `unify_portable_cli.exe` in a terminal inside folder where you have extracted the Unify Portable Environment. It will start the environment with the default frequency for the EU region.

   Note: You may need to run the executable from an **elevated** terminal.

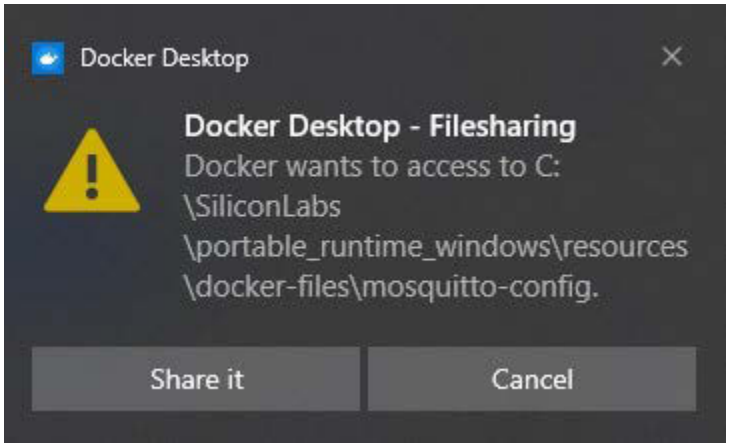
3. If there are more devices connected via USB, the controller will be automatically identified based on its application name or, failing that, you will be prompted to select it from the list. Alternatively, the serial number of the controller can be specified directly in the command: `unify_portable_cli.exe -s <SERIAL_NO>`
4. The serial number of the devices can be identified based on Step 3.2 List Connected Devices.



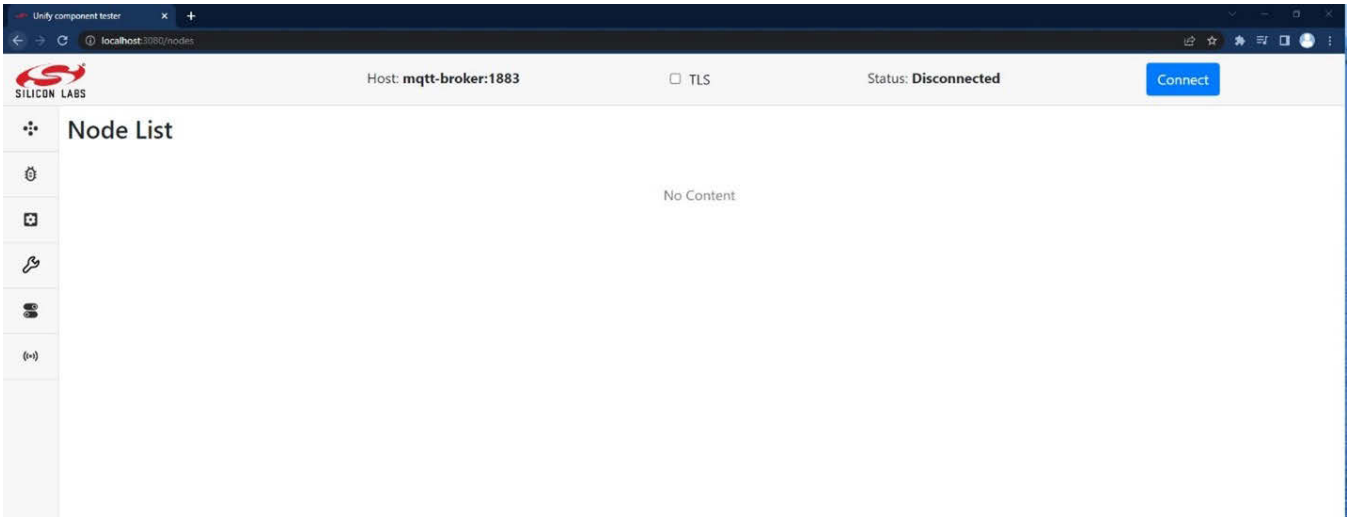
5. Firewall access must be granted for the application, Docker, and USB connection.



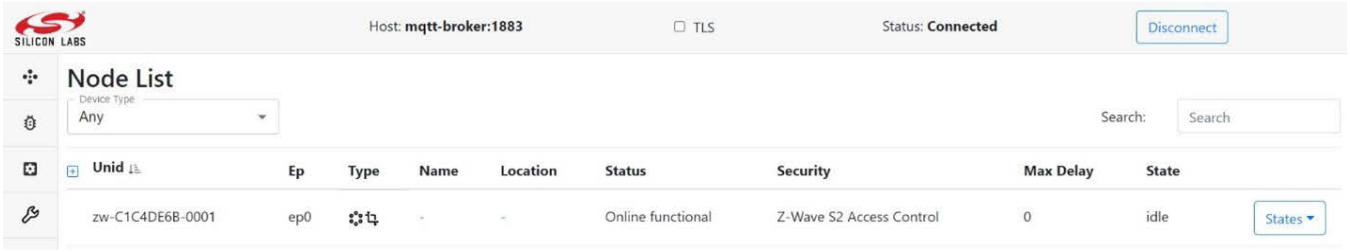
6. Allow Docker access to the required files (mosquitto-config location) if prompted in the Action Center.



7. Launch the GUI in your browser using the URL http://localhost:3080.

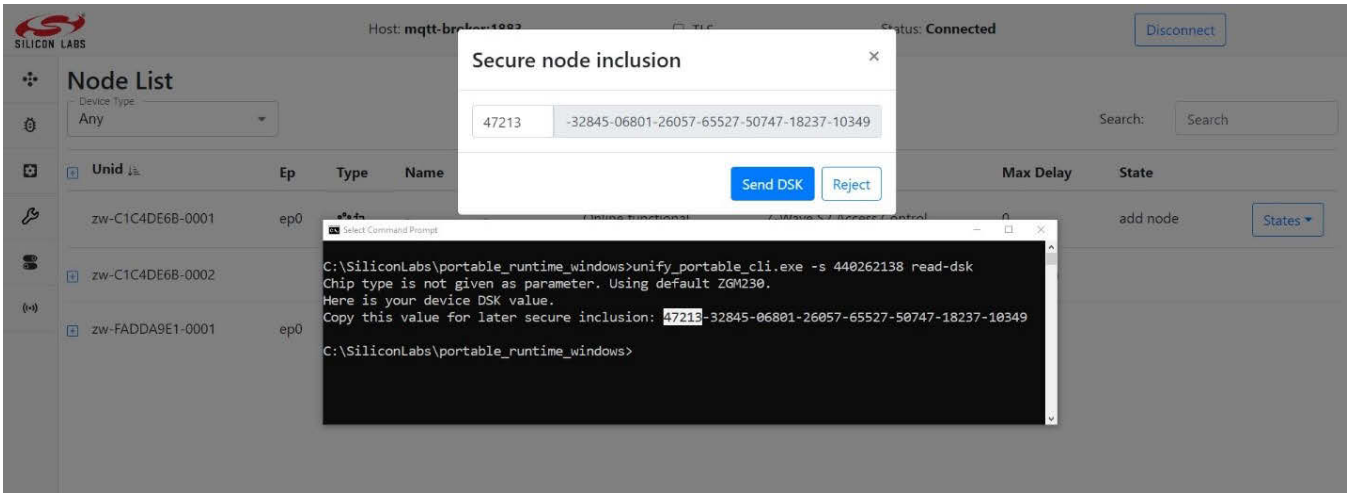


8. Click the header at the top of the page to enter your credentials. Set the **Host** to **mqtt-broker** and the **Port** to **1883** and click **Connect.**
9. In the Node List, the Controller device should now appear.



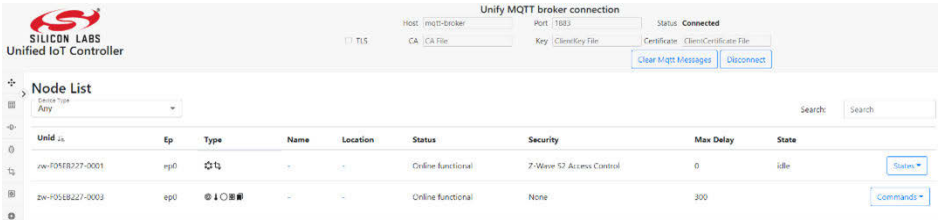
Step 5: Include the End Device in the Z-Wave Network

1. Power on the end device by connecting it via a USB-C cable or by inserting a battery.
2. In the Node list, click the **States** button on the right side of the Controller's row and select **Add Node** from the drop-down list.
3. Activate inclusion mode on the end device by pressing the button labelled BTN1 on the Thunderboard.
4. Input the first five digits of the DSK key from the label in the pop-up window or click **Reject** to perform an insecure inclusion.



5. A popup warning message is displayed in the lower right corner when a non-secure inclusion is performed:

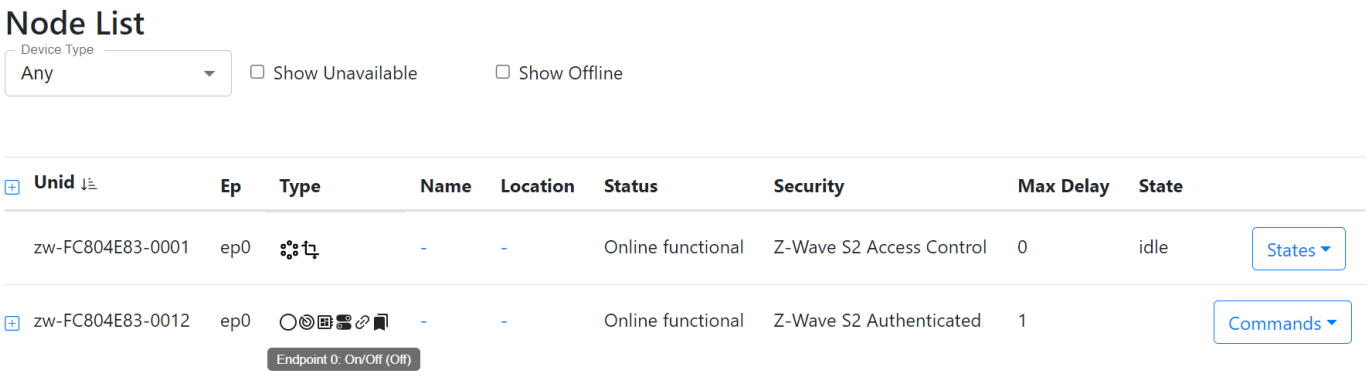
6. After a successful inclusion, the end device appears below the Controller row.



The icons under the **Type** category indicate the supported functionalities of the nodes.

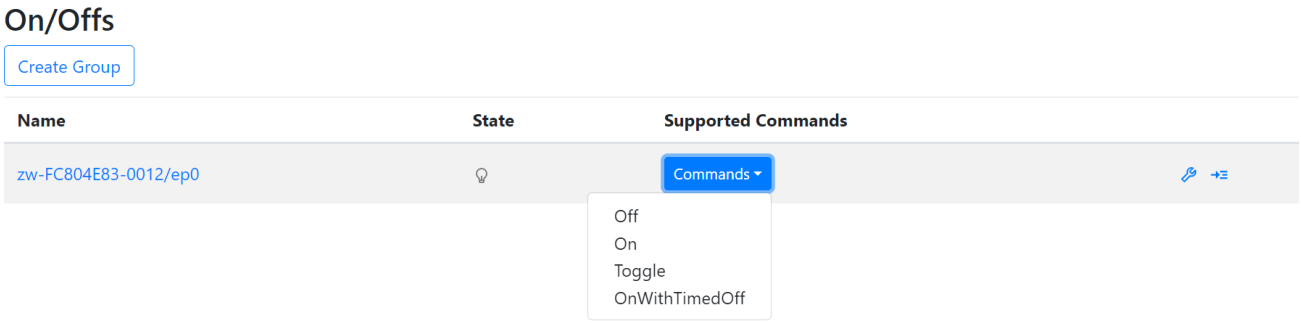
Step 6: Try the Application

- For the **"Z-Wave - SoC Switch On/Off"** end device, the capabilities are as shown below:

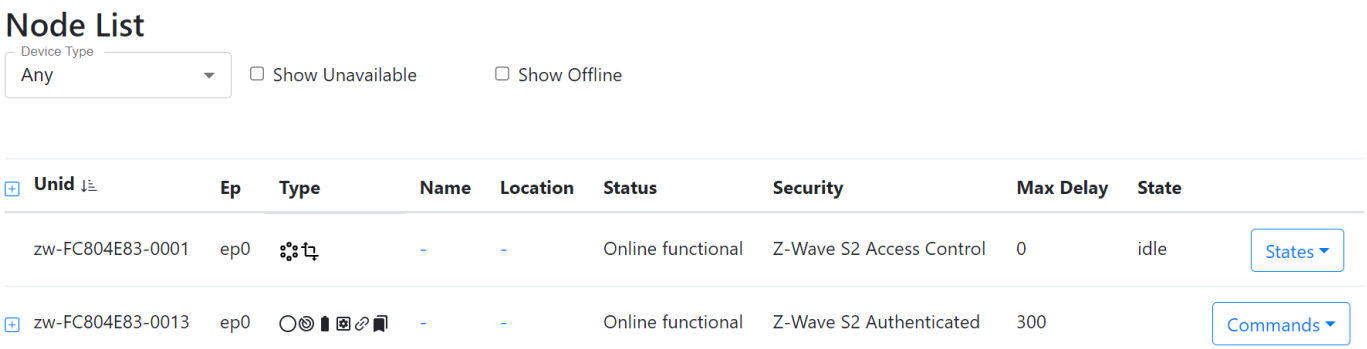


- The capabilities of the connected node can be seen by clicking the icons:

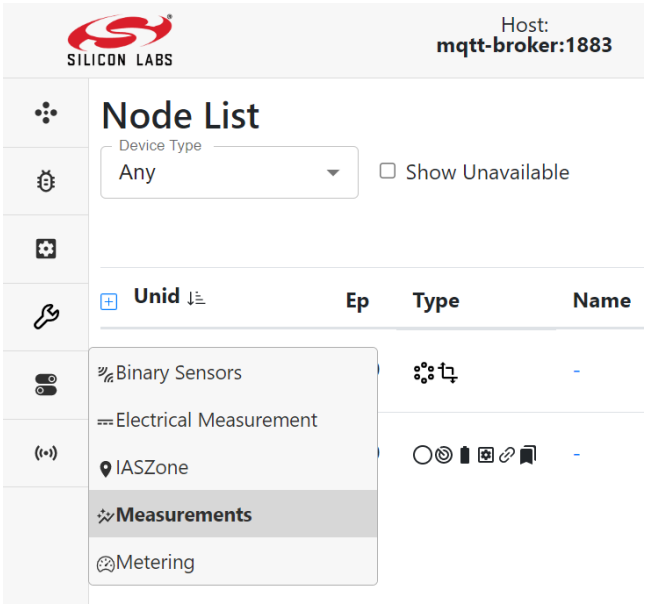
- Clicking the On/Off icon navigates to a new view where the LED can be controlled on the Thunderboard:



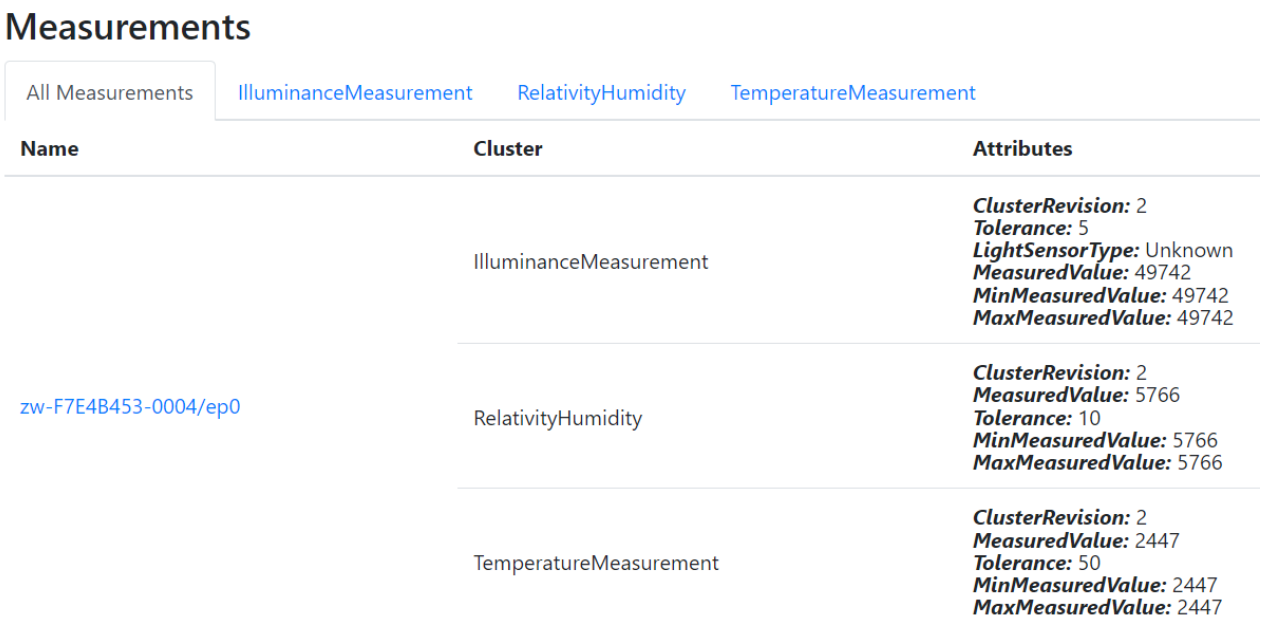
- For the **"Z-Wave - SoC Multilevel Sensor"** end device, the capabilities are:



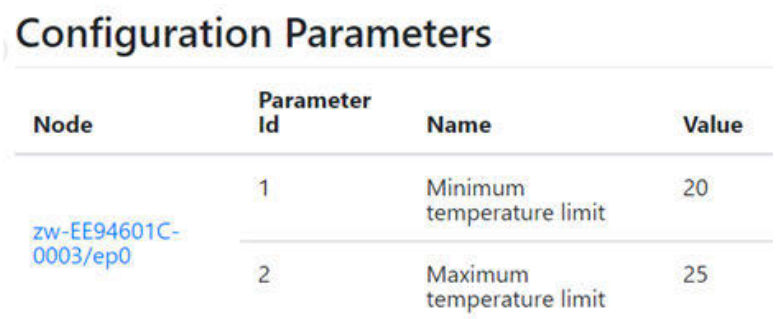
- To access the values measured by the device, go to the Measurements view.



- Here, the temperature value measured by the onboard thermostat can be read out.



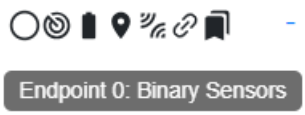
- In the Configuration Parameters window, a minimum and maximum temperature threshold can be set:



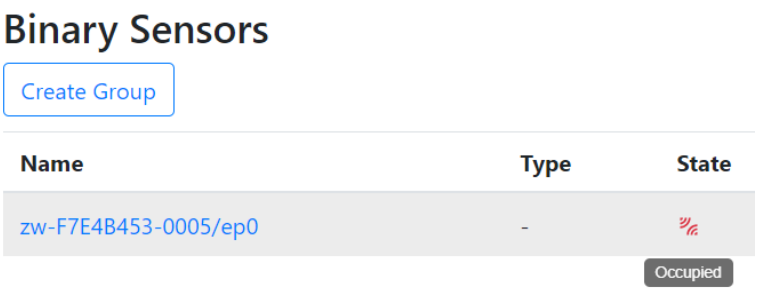
Configuration Parameters

Node	Parameter Id	Name	Value
zw-EE94601C-0003/ep0	1	Minimum temperature limit	20
	2	Maximum temperature limit	25

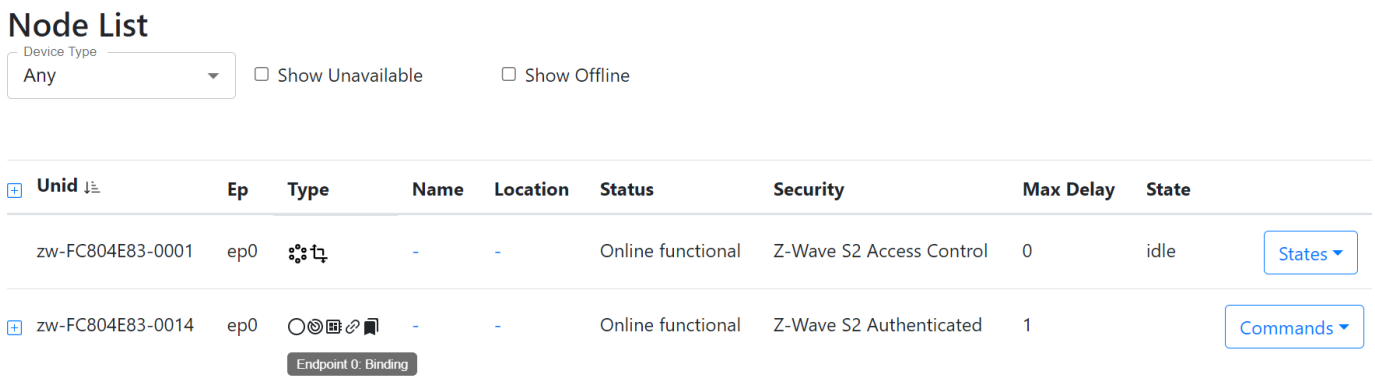
- For the **"Z-Wave - SoC Sensor PIR"** End Device, the capabilities are as shown below:



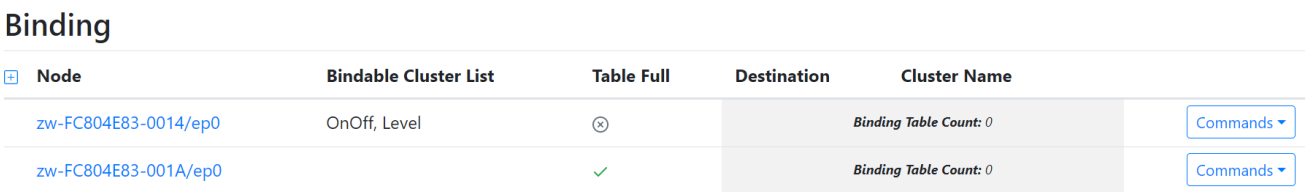
  - To view the state of the passive infrared sensor, click on the Binary Sensors button.
  - The sensor will be briefly shown as “Occupied” when you hold and release BTN1.



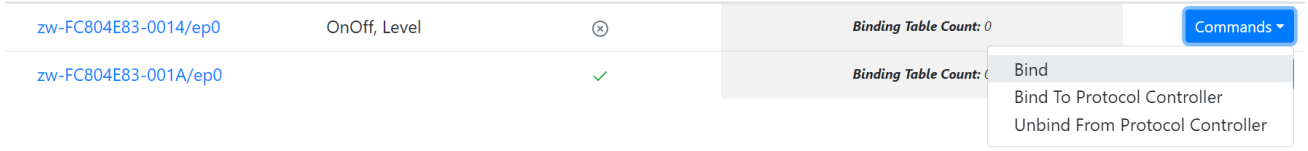
- For the **"Z-Wave - SoC Wall Controller"** End Device, the capabilities are as shown below:



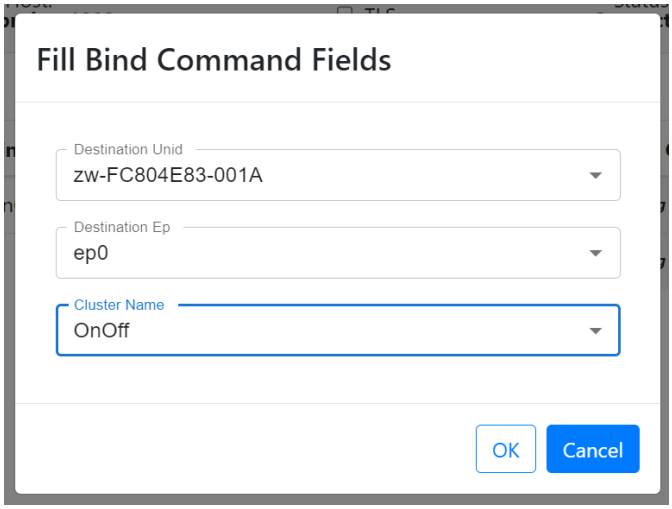
- Clicking the Binding button navigates to a new view, where associations can be made between nodes.



- For example, we can set up the Wall Controller to control a Switch On/Off device. To do this, first click on the Commands button and select Bind.

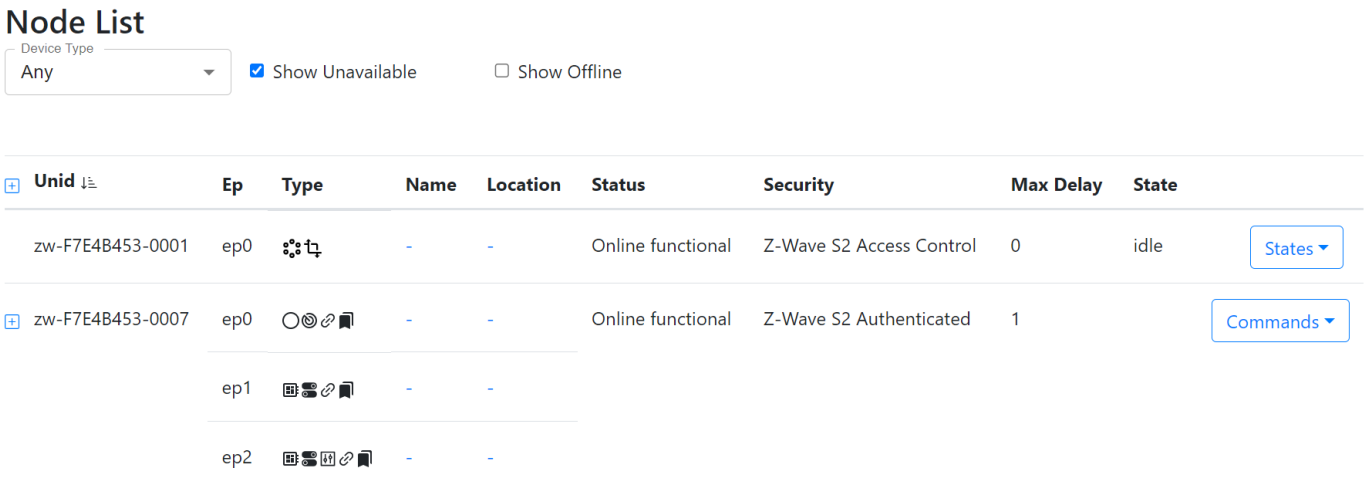


- In the pop-up dialog, select the device to be controlled (Destination Unid), the desired endpoint (Destination Ep) and the Command Class (Cluster Name), then click OK.

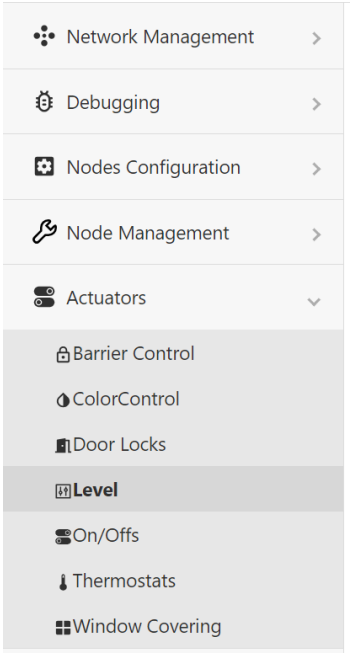


- Now that the binding has been created, click BTN0 on the Wall Controller to toggle LED0 on the associated Switch On Off.

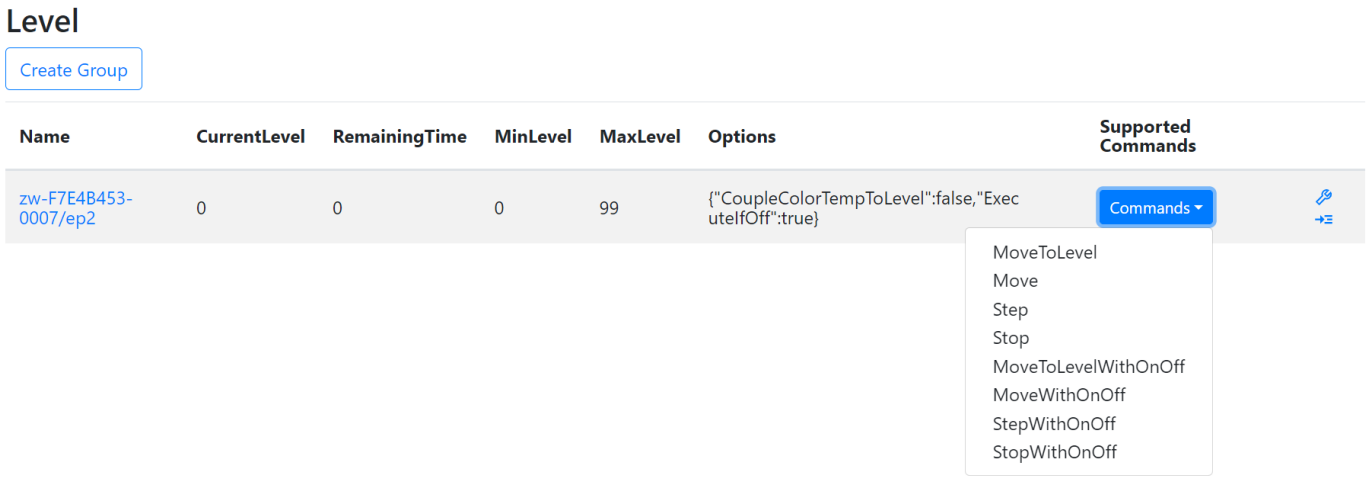
- For the **"Z-Wave - SoC Power Strip"** End Device, the capabilities are as shown below:



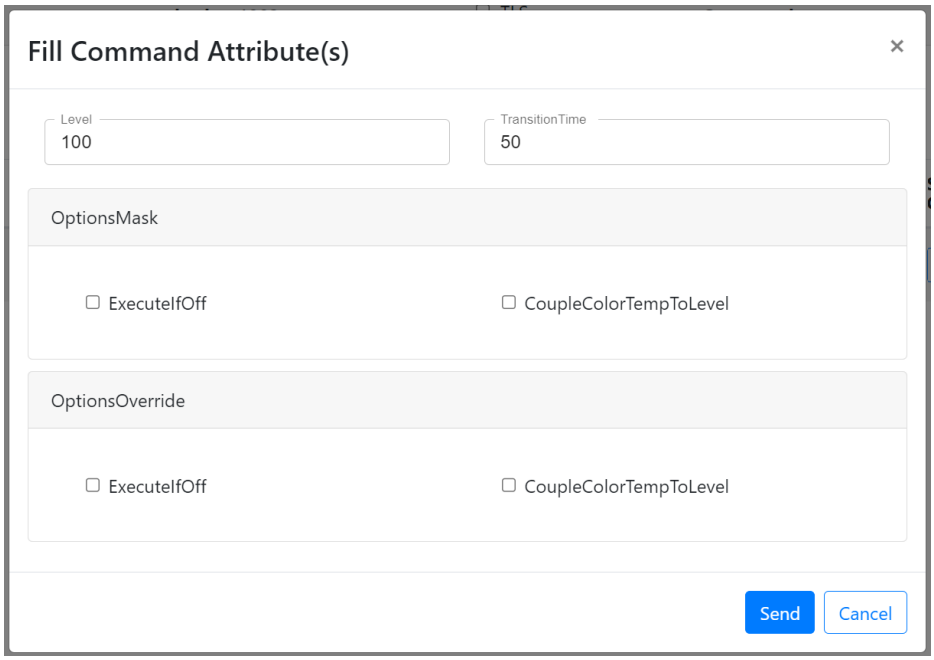
- To access the controls of the multilevel switch, select “Level” from the “Actuators” menu.



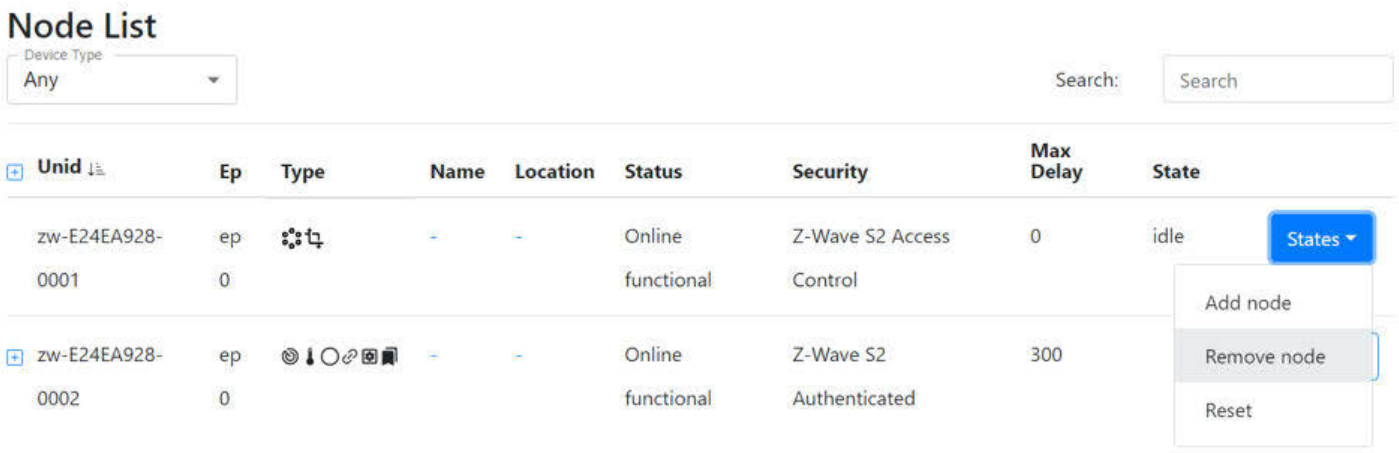
- In the Level view, click on the Commands button, then select “MoveToLevel”.



- Set “Level” to 100 and “TransitionTime” to 50, then click “Send”. LED1 transitions from dark to full brightness over 5 seconds.



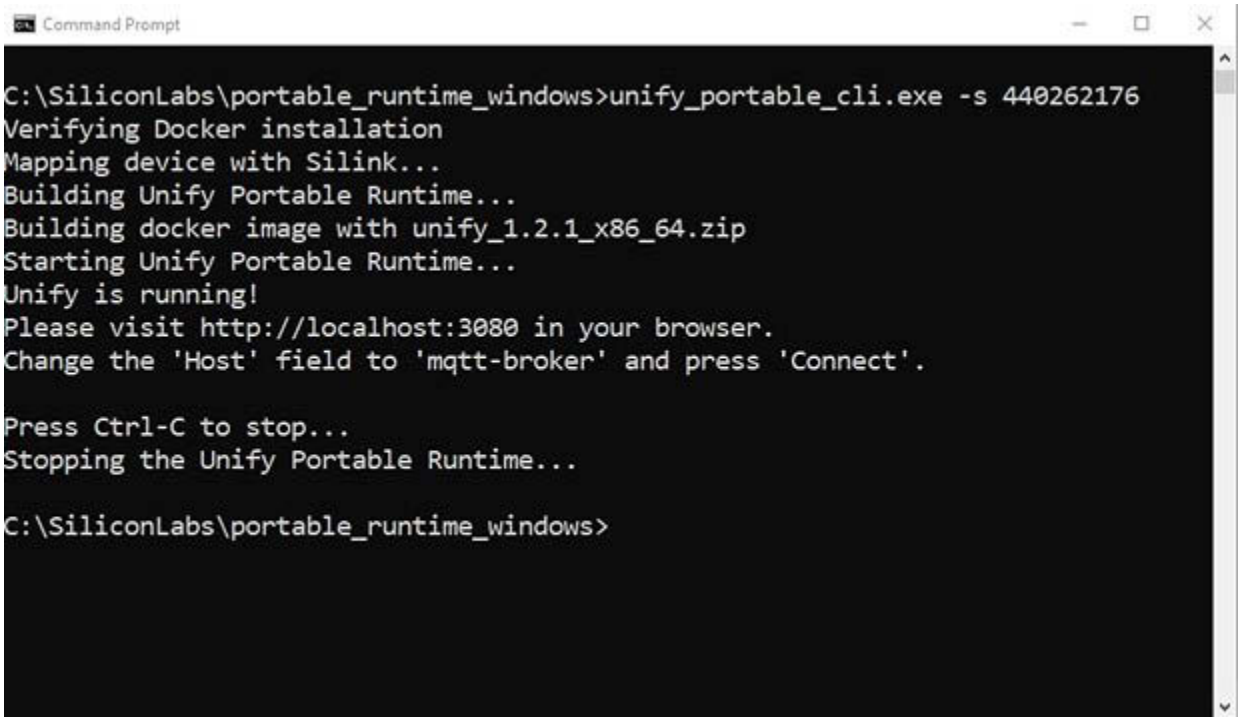
Step 7: Exclusion of the Device



1. Press the **States** button on the right side of the controller's row and select **Remove node** in the drop-down list. Next, press the button labelled **BTN1** on the end device's board.
2. If using a Multilevel Sensor application, the device may be in a deep-sleeping state after the **Remove node** has been pressed. If this is the case, press the **RESET** button on the board to wake it up, then **BTN1** to perform the exclusion.

Step 8: Stop the Unify Portable Runtime Environment

1. On the web-based GUI, click the **Disconnect** button.
2. To stop the Unify Portable Environment, go back to the terminal and press Ctrl+C. The terminal will show the default prompt when the shut-down process is complete.



2. Advanced Options

Descriptions for all the available commands and subcommands in the Unify Portable CLI can be accessed by using the **-h** option.

```
unify_portable_cli.exe -h
unify_portable_cli.exe flash-app -h
unify_portable_cli.exe read-dsk -h
unify_portable_cli.exe list-devices -h
unify_portable_cli.exe list-devices-raw -h
```

Some additional options are described in the following subsections.

2.1 Using Custom Region Frequency

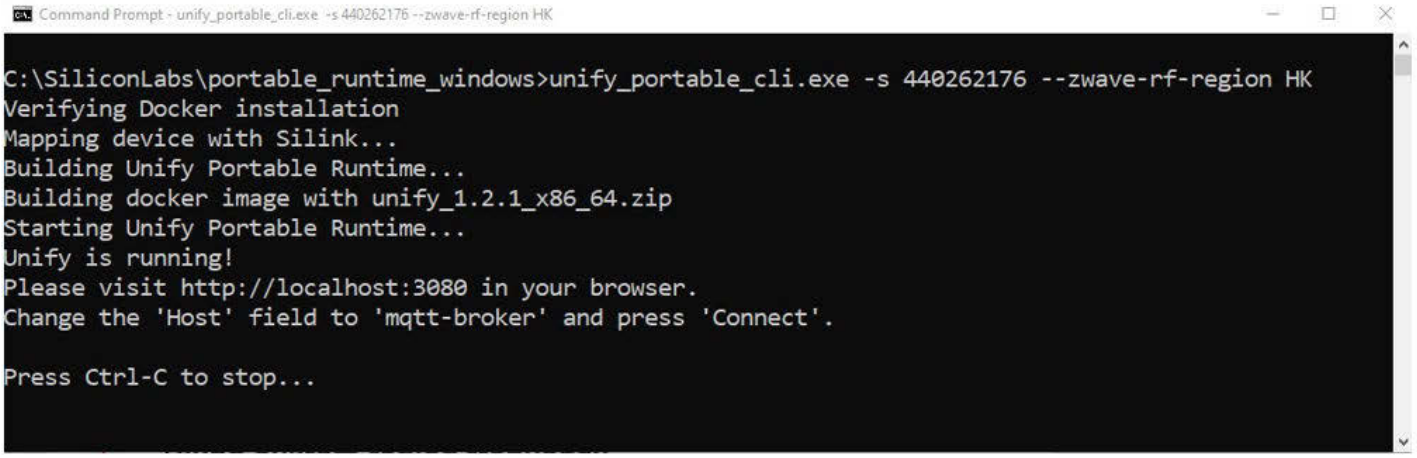
The Quick Start Guide assumes that the EU Z-Wave frequency region is used. It is possible to use different region frequencies by re-flashing the end device and starting the Unify Portable Environment with a specified region.

2.1.1 Start Unify Portable Environment with Specified Region Frequency

When no region frequency is specified, the Unify Environment sets the controller to use the EU region frequency by default. It is possible to specify the region by using **-zwave-rf-region** option.

```
unify_portable_cli.exe -s <SERIAL_NO> --zwave-rf-region <REGION>
```

The supported options for <REGION> are EU, US, US_LR, ANZ, HK, MA, IN, IS, RU, CN, JP, and KR.



2.1.2 Flashing Device with Specified Region Frequency

When no region frequency is specified, the end devices are flashed with the EU region frequency by default. It is possible to specify the region to be used on end devices by using the **-zwave-rf-region** option with the **flash-app** subcommand.

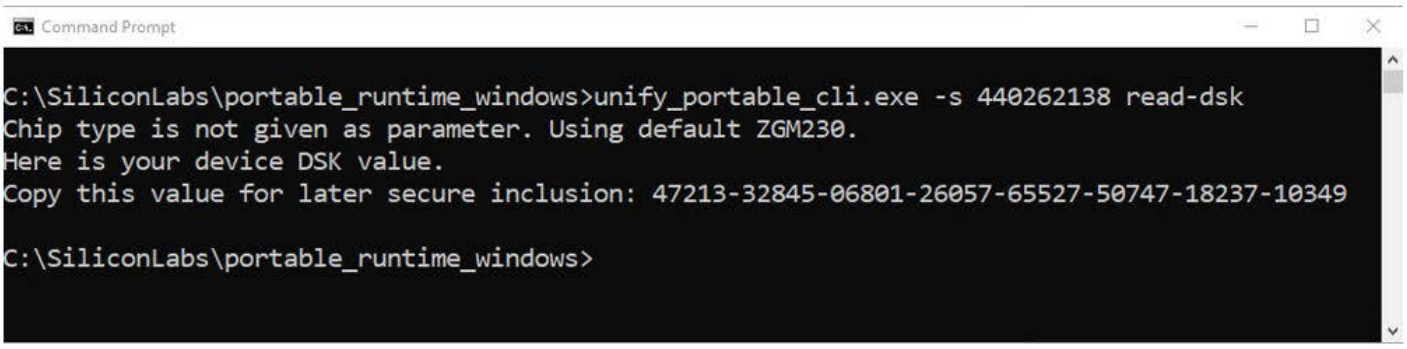
```
unify_portable_cli.exe --zwave-rf-region <REGION> flash-app -n <APP_NAME>
```

The supported options for <REGION> are EU, US, US_LR, ANZ, HK, MA, IN, IS, RU, CN, JP, and KR.

2.2 Read DSK Value

To perform a secure inclusion, it is necessary to provide the DSK of the end device. It can be read by running the following command:

```
unify_portable_cli.exe -s <SERIAL_NO> read-dsk
```



where the SERIAL_NO can be determined using the *list-devices* subcommand. By omitting the -s parameter, you will be prompted to select one of the connected devices from the list instead.

2.3 Flash Custom Application Firmware

It is possible to flash custom-built firmware to the Thunderboard. Currently, the Thunderboard supports six applications:

- SerialAPI – Controller NCP application
- SwitchOn/Off – End Device application
- Multilevel Sensor – End Device application
- SensorPIR – End Device application
- WallController – End Device application
- PowerStrip – End Device application

It is possible to build these applications using Simplicity Studio. Specify the path to the custom-built application's binary by using the **-p** switch.

```
unify_portable_cli.exe flash-app -s <SERIAL_NO> -p custom_firmware.hex
```

2.4 Flash to Other Silicon Labs Z-Wave Board

The Unify Portable Environment can also be used to control and flash other types of Silicon Labs Z-Wave compatible development kits and radio boards.

If the processor type is not specified with the flash-app subcommand, the default parameter is ZGM230. If the processor type on the desired target is different, it can be specified with the **-c** option.

```
unify_portable_cli.exe -s <SERIAL_NUMBER> flash-app -p <FIRMWARE_PATH> -c <PROCESSOR_TYPE>
```

For different boards, the corresponding application firmware should be downloaded and used instead of the built-in app firmware. See the "Flash custom application firmware" section.

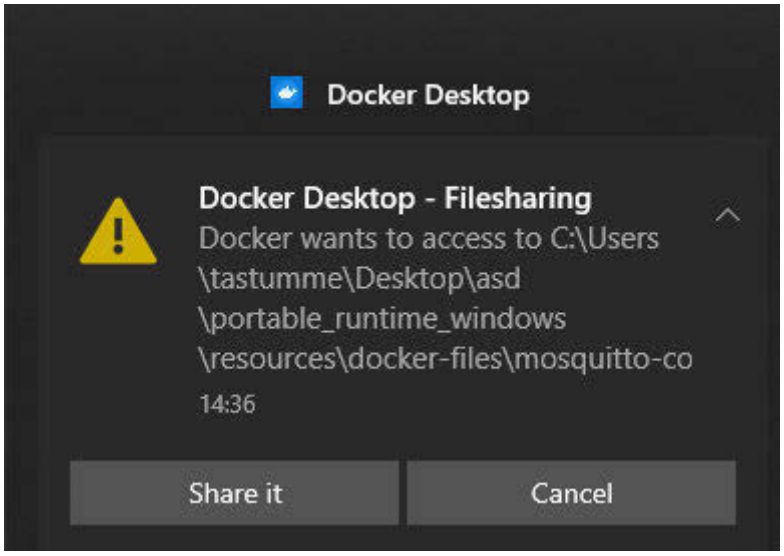
As with the other commands, specifying the serial number is not necessary. If it is omitted, the target board can be selected from the list of connected devices.

3. Troubleshooting

3.1 Docker Settings – Filesharing to Mosquitto Configuration

After starting the Unify Portable Environment for the first time, Docker needs access to the mosquito-config folder. This must be granted.

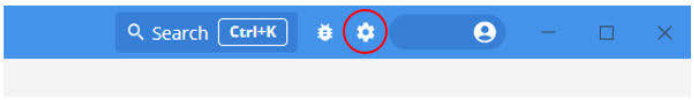
If you do not see a pop-up window requesting access to this folder, try looking for this notification in the Action Center and grant the access.

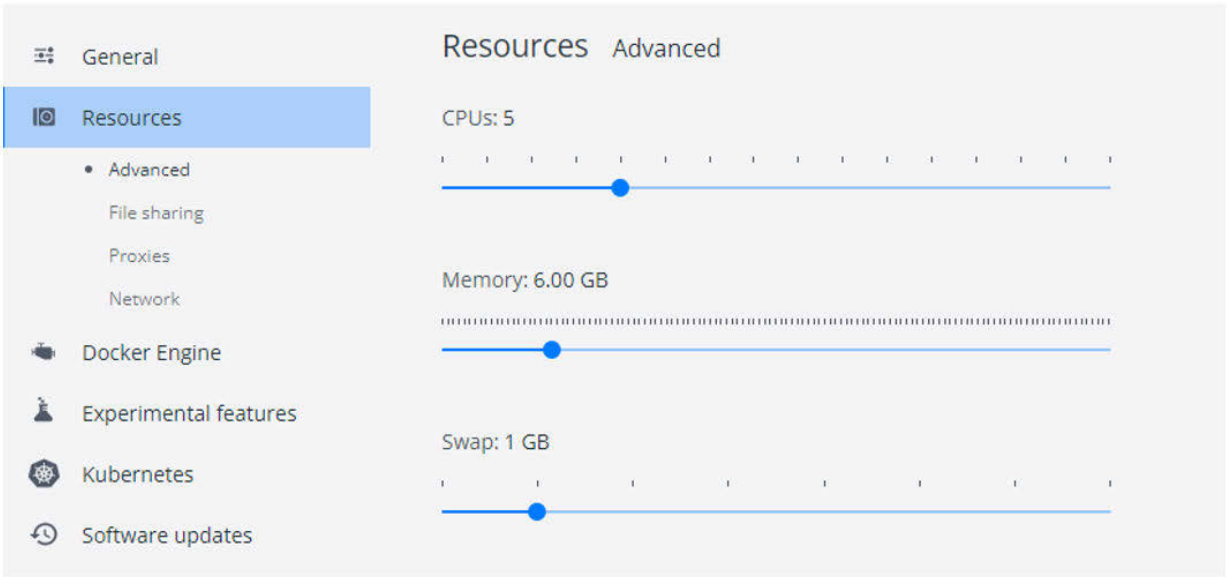


3.2 Docker Settings – Disk Space

The portable environment may fail to start properly when it cannot allocate enough resources for itself.

If this is the case, open Docker Desktop's settings by clicking the gear icon on the top right. In this window, select the **Resources** view on the left, and increase the availability of resources to the minimum recommended values, as shown on the screenshots below.





3.3 Devices Not Detected by Unify

If the Unify GUI is not able to detect the connected devices, try the following:

1. Press **Disconnect** in the web GUI.
2. Stop the Portable Environment.
3. Press the **RESET** button on the devices.
4. Start the Portable Environment again.

3.4 The Environment Freezes

If the Unify Portable Environment becomes unresponsive, try the following:

1. Open Docker Desktop.
2. In the **Containers** view, **Stop** the environment's container if it is still running.
3. Select the **Volumes** view and **Delete** the volumes related to the portable environment.

3.5 Application Name is "unknown"

If the Application name field shows up as "unknown" in the device list when issuing the list-devices command, try the following:

1. Press the RESET button on the device and try again.
2. Make sure a supported firmware has been flashed to the device with the Unify Portable Environment CLI.
3. If you are using a custom firmware, make sure the `zaf_appname_nvm` component is enabled for the project.

Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support & Community
www.silabs.com/community

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

www.silabs.com