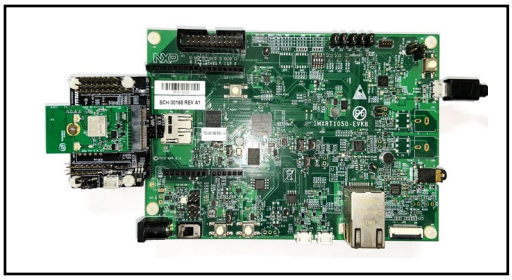
E2B-07-1399

Murata Wi-Fi/BT (NXP) Solution for i.MX

FreeRTOS User Guide



Revision History

Revision	Date	Author	Change Description
1.0	August 30, 2021	TF	Initial Release
1.1	Sept 3, 2021	TF	Fix Murata MCUXpresso 2.10.0 SDK patch link.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1 Introduction

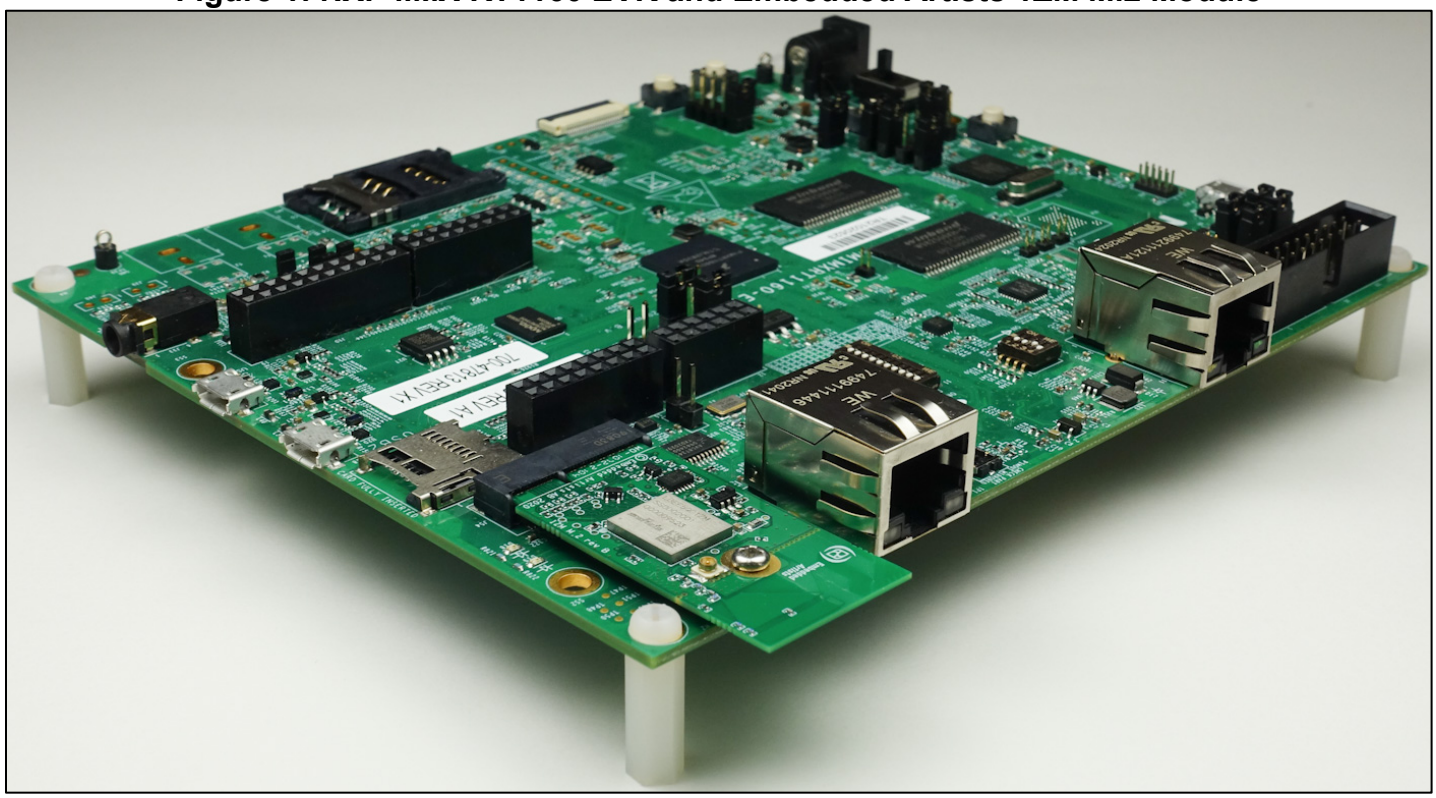
This document details enabling Murata's (NXP-based) Wi-Fi/Bluetooth modules (2DS/1XK/1ZM) on NXP i.MX RT Evaluation Kits (RT500/600/1050/1060/1064/1160/1170) and Embedded Artists i.MX RT Developer Kits (RT1062/1064/1176), using Embedded Artists' Wi-Fi/BT M.2 modules.

Murata supports several NXP i.MX RT EVK's and Wi-Fi/Bluetooth M.2 modules with a new MCUXpresso SDK 2.10 patch – downloadable from Murata's Community Forum. Embedded Artists provides their own drop-in patch for their i.MX RT Developer Kits. Note that Wi-Fi support is consistent throughout. However, BT/BLE support is limited to specific NXP i.MX RT platforms such as RT1060 and RT1170 on 1XK/1ZM modules.

NXP's MCUXpresso IDE and SDK solution provides strong support to customers looking to get their wireless-enabled i.MX RT product to market quickly and easily. Embedded Artists provides drop-in hardware interconnect with the M.2 modules in addition to their own enhanced i.MX RT hardware solutions. Murata's Wi-Fi/Bluetooth modules are best of breed with comprehensive hardware, software, and regulatory support. Murata's Community Forum provides easy access to customers needing hands-on support.

Figure 1 below shows one such example – NXP i.MX RT1160 EVK with Type 1XK module (enabled by Embedded Artists' 1XK M.2 module).

Figure 1: NXP i.MX RT1160 EVK and Embedded Artists 1ZM M.2 Module



1.1 Acronyms

Table 1: Acronyms used in Quick Start Guide

Acronym	Meaning
BT	Bluetooth
EA	Embedded Artists designs, manufactures and distributes current Wi-Fi/BT M.2 EVB's (link here). EA also have enhanced i.MX developer kits which provide comprehensive support for Murata modules (link here).
EULA	End User License Agreement
EVB	Evaluation Board (Embedded Artists' Wi-Fi/BT module)
EVK	Evaluation Kit
FTDI	Future Technology Devices International
IDE	Integrated Development Environment
JTAG	Joint Test Action Group
M.2	Formerly known as the Next Generation Form Factor (NGFF), is a specification for internally mounted computer expansion cards and associated connectors. The M.2 specification is defined by PCI-SIG (www.pcisig.com).
PC	Personal Computer
RF	Radio Frequency
RTOS	Real-time Operating System
RX	Receive
SD	Secure Digital
SDIO	Secure Digital Input Output
SDK	Software Development Kit
TX	Transmit
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
uSD	Micro SD
uSD-M.2	Micro SD to M.2 Adapter
Wi-Fi	Wireless LAN: "Wi-Fi" is a registered trademark of Wi-Fi Alliance
WLAN	Wireless Local Area Network

1.1 References

1.1.1 Murata Wi-Fi/BT (NXP) Solution for i.MX FreeRTOS Quick Start Guide

This Quick Start Guide provides quick steps to get started with Murata Wi-Fi/BT NXP chipset-based solution with the help of an example.

1.1.2 Murata Wi-Fi/BT Solution for i.MX Hardware User Manual

This manual describes the Murata uSD-M.2 Adapter hardware. All interface signals to the NXP i.MX RT EVK's are described. Specifics on interfacing each i.MX EVK to Murata uSD-M.2 Adapter are provided.

1.1.3 Murata's Community Forum Support

Murata's Community provides online support for the Murata Wi-Fi/Bluetooth modules on various i.MX platforms. Refer to this link for existing support threads on i.MX RT MCUXpresso.

1.1.4 Murata uSD-M.2 Adapter Datasheet (Rev B1)

This datasheet documents the current version of the Murata' uSD-M.2 adapter hardware and its interfacing options.

1.1.5 Murata's uSD-M.2 Adapter Landing Page

This website landing page provides latest/comprehensive information on Murata's adapter including links to where it can be purchased.

1.1.6 Murata Patching Solution

This archive file contains the files necessary to enable the Murata modules for the Wi-Fi/BT examples on MCUXpresso SDK 2.10 for NXP i.MX RT EVKs.

1.1.7 Murata's i.MX Wireless Solutions Landing Page

This website landing page provides latest/comprehensive information on Murata's i.MX Wireless solutions which use the uSD-M.2 Adapter as a key enabler so customers can easily evaluate Murata's modules on i.MX processors.

1.1.8 Embedded Artists' Reference Documentation

Embedded Artists designed the 2DS/1XK/1ZM/1YM M.2 EVB's in close collaboration with Murata. It is ***important to note*** that Embedded Artists manufactures and distributes the Wi-Fi/BT M.2 EVB's. Refer to this main landing page for more information: www.embeddedartists.com/m2. **Table 2** lists some relevant documents published by Embedded Artists.

Table 2: Embedded Artists Documentation Listing

Documentation Filename	Note
Wi-Fi/BT M.2 EVB Primer	Introduction and drill-down on M.2 interface
M.2 SDIO Interface Schematic	Reference schematic for customers designing in WLAN-SDIO M.2 EVB.
M.2 PCIe Interface Schematic	Reference schematic for customers designing in WLAN-PCIe M.2 EVB.
2DS M.2 Module Datasheet	Comprehensive details on 2DS Wi-Fi/BT M.2 Module.
1XK M.2 Module Datasheet	Comprehensive details on 1XK Wi-Fi/BT M.2 Module.
1ZM M.2 Module Datasheet	Comprehensive details on 1ZM Wi-Fi/BT M.2 Module.
1YM M.2 Module Datasheet	Comprehensive details on 1YM Wi-Fi/BT M.2 Module.

1.2 Hardware Options

This section describes how to configure the hardware correctly, for both NXP and Embedded Artists i.MX RT series platforms. The following table lists the different platforms supported by Murata modules.

Table 3: Murata module support on NXP and Embedded Artists' platforms

i.MX EVK / Dev kit	Manufacturer	Part number	Murata modules supported	Interconnect
i.MX RT1170 EVK	NXP	MIMXRT1170-EVK	2DS, 1XK, 1ZM	M.2, uSD-M.2 Adapter
i.MX RT1160 EVK	NXP	MIMXRT1160-EVK	2DS, 1XK, 1ZM	M.2, uSD-M.2 Adapter
i.MX RT1064 EVK	NXP	MIMXRT1064-EVK	2DS, 1XK, 1ZM	uSD-M.2 Adapter
i.MX RT1060 EVK	NXP	MIMXRT1060-EVK	2DS, 1XK, 1ZM	uSD-M.2 Adapter
i.MX RT1050 EVK	NXP	IMXRT1050-EVKB	2DS, 1XK, 1ZM	uSD-M.2 Adapter
i.MX RT595 EVK	NXP	MIMXRT595-EVK	2DS, 1XK, 1ZM	uSD-M.2 Adapter
i.MX RT685 EVK	NXP	MIMXRT685-EVK	2DS, 1XK, 1ZM	uSD-M.2 Adapter
i.MX RT1176 Dev Kit	Embedded Artists	EAK00380	2DS, 1XK, 1ZM	M.2
i.MX RT1064 Dev Kit	Embedded Artists	EAC00375	2DS, 1XK, 1ZM	M.2
i.MX RT1062 Dev Kit	Embedded Artists	EAK00310	2DS, 1XK, 1ZM	M.2

1.3 Software Options

Several toolchains are supported by NXP as below, but MCUXpresso IDE is the primary focus in this document:

- NXP supports MCUXpresso IDE
- GNU toolchain for Arm® Cortex® -M with Cmake build system
- IAR Embedded Workbench
- Keil™ MDK-Arm

2 Hardware Setup for NXP EVKs with uSD-M.2 Adapter

To enable Murata's wireless solution on NXP's i.MX RT 1020/1050/1060/1064/595/685 Evaluation Kits, Embedded Artists' Wi-Fi/BT M.2 EVB's (Murata module onboard) must be connected to Murata's uSD-M.2 Adapter. The on-board debug adapter is supported. Refer to **Figure 2** for example of i.MX RT1050 EVK. Murata's uSD-M.2 Adapter plugs in directly to the EVK's microSD connector. The micro USB connector (J28) is used for USB-UART/JTAG. The NXP i.MX RT 1160/1170 EVKs have on-board M.2 slot that can be used, in addition to the uSD-M.2 adapter option.

Refer to **Section 9** on how to correctly connect Embedded Artists' Wi-Fi/BT M.2 EVB to the Murata Adapter and how to properly jumper the Adapter for default 1.8V VIO operation (not 3.3V override mode). Now insert the Murata Adapter into the microSD slot (J20) until you hear the click sound (push-push connector). Per **Section 9.3**, it is best to tape the uSD Adapter-microSD connection. Make sure the green LED (LED1) on the adapter board is illuminated when powered. Also, the blue LED (LED2) should **not be** illuminated. Repeating the Murata uSD-M.2 Adapter jumper settings:

- For rev B1 adapter, J12 is in 1-2 pos & J13 is in 1-2 pos.
- For (legacy) rev A adapter, J12 is open.

Additional connections between the EVK and the uSD-M.2 adapter are required for full functionality, using male-to-female jumper cables. Please refer to **Table 4** and **Table 5** for additional connection requirements on the various NXP i.MX RT EVKs.

Figure 2: Connecting the EVB to the EVK

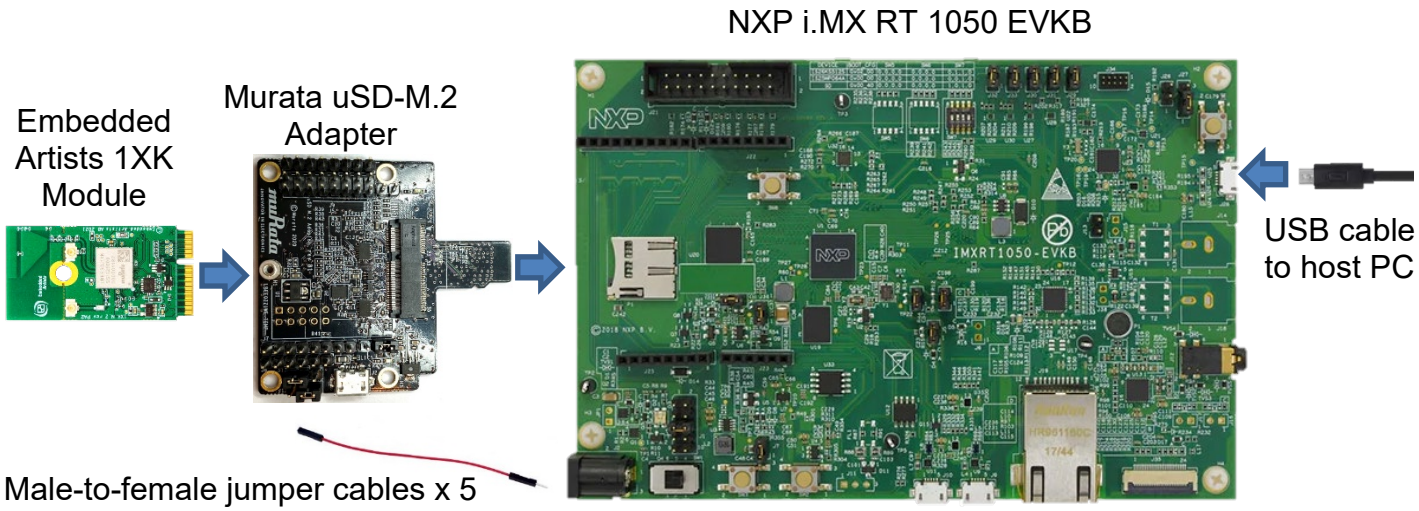


Table 4: Additional cabling for NXP i.MX RT EVK 1050/1060/1064

Pin name	uSD-M.2 adapter pin	i.MX RT 10XX pin	Pin name of RT10XX	GPIO name of RT10XX
BT_UART_TXD_HOST	J9 (pin 1)	J22 (pin 1)	LPUART3_RXD	GPIO_AD_B1_07
BT_UART_RXD_HOST	J9 (pin 2)	J22 (pin 2)	LPUART3_TXD	GPIO_AD_B1_06
RADIO_ENABLE	J9 (pin 3)	J22 (pin 3)	GPIO1_IO11	GPIO_AD_B0_11
BT_UART_RTS_HOST	J8 (pin 3)	J23 (pin 3)	LPUART3_CTS	GPIO_AD_B1_04
BT_UART_CTS_HOST	J8 (pin 4)	J23 (pin 4)	LPUART3_RTS	GPIO_AD_B1_05

Table 5: Additional cabling for NXP i.MX RT EVK 1160/1170

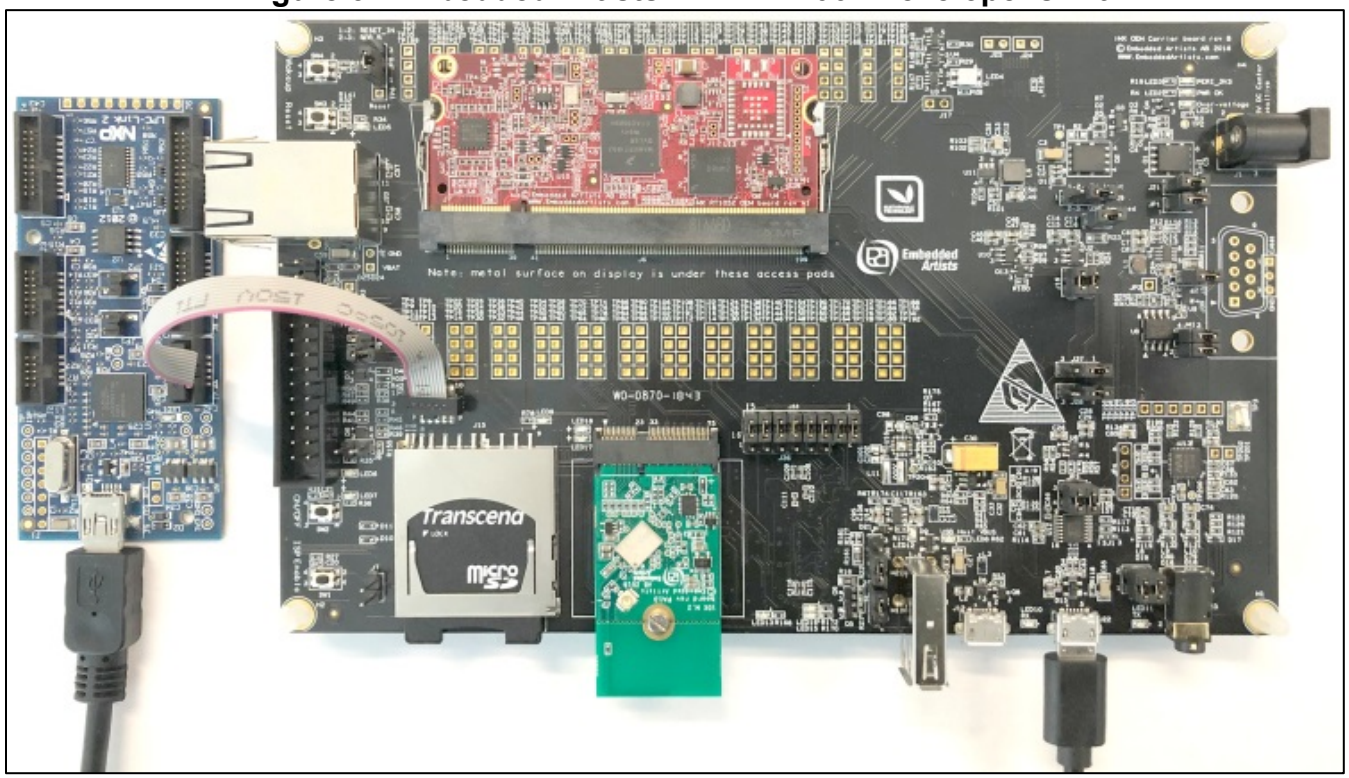
Pin name	uSD-M.2 adapter pin	i.MX RT 10XX pin	Pin name of RT11XX	GPIO name of RT11XX
BT_UART_TXD_HOST	J9 (pin 1)	J25 (pin 13)	LPUART7_RXD	GPIO_AD_01
BT_UART_RXD_HOST	J9 (pin 2)	J25 (pin 15)	LPUART7_TXD	GPIO_AD_00
BT_UART_RTS_HOST	J8 (pin 3)	J25 (pin 11)	LPUART7_CTS	GPIO_AD_02
BT_UART_CTS_HOST	J8 (pin 4)	J25 (pin 9)	LPUART7_RTS	GPIO_AD_03

3 Hardware Setup for Embedded Artists Dev Kits via M.2 interface

Embedded Artists' i.MX RT Developer Kits have a M.2 connector onboard for direct connection to the M.2 EVB (no adapter required). LPC-Link2 is recommended for the debug adapter. Embedded Artists' website provides support package. **Figure 3** shows the full connection of developer's kit with M.2 and debug probe. Micro USB connector (J22) is used for USB-UART. J10 is used for LPC-Link2 connection.

NOTE: The red line in the flex cable used to connect the debug probe to the developer's kit should align with the arrow at J10 (i.e., pin 1 of J10).

Figure 3: Embedded Artists i.MX RT 1062 Developer's Kit



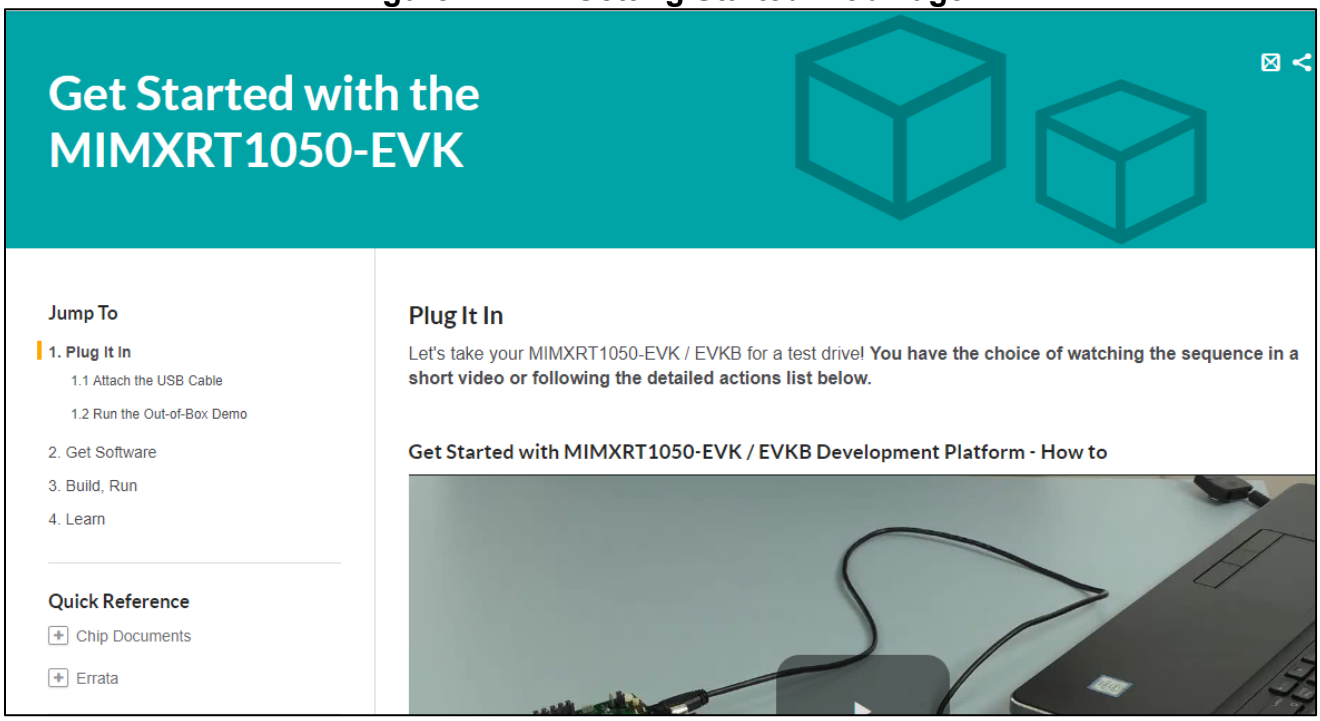
4 Software Setup for NXP EVKs

If you are using Embedded Artists Board, please skip this section and go to **Section 5** of this document.

Click here to go the NXP landing page as shown in **Figure 4**. Follow the steps described in NXP web to install these tools:

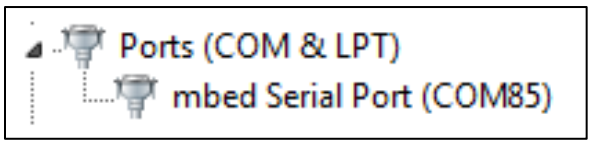
- MCUXPresso IDE
- mbed Virtual COM Port Driver
- Terminal application (TeraTerm, Putty, etc.)

Figure 4: NXP Getting Started Web Page



You should now be able to see mbed Serial Port on the Device Manager as shown in **Figure 5** if you completed software installation successfully.

Figure 5: COM Port of NXP IMXRT1050-EVKB



For installing the required SDK(s) for MCUXpresso IDE, please refer to **Section 6**.

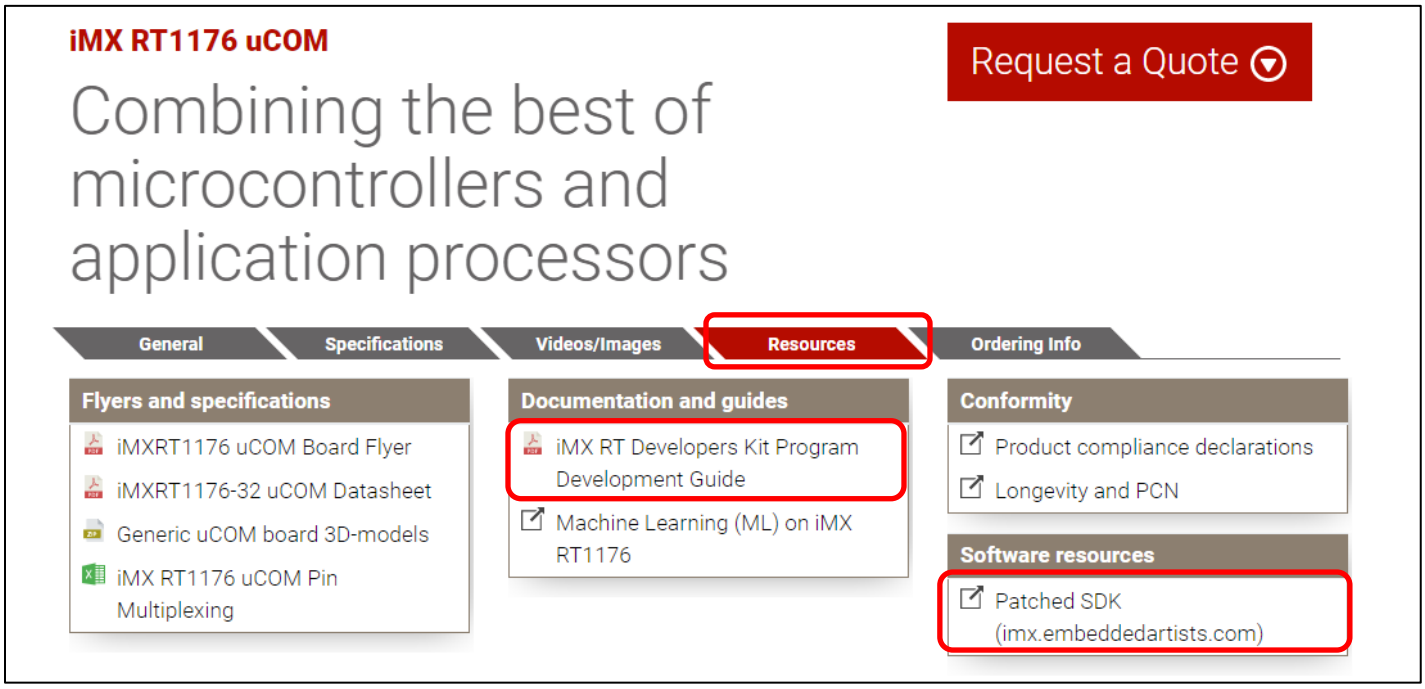
5 Software Setup for Embedded Artists Dev Kits

If you are using NXP Board, please skip this section and go back to **Section 4** of this document.

Click here to go the EA landing page for i.MX RT 1176 as shown in **Figure 6**. Download the document iMX RT Developers Kit Program Development Guide from the resource tab for detailed instructions about downloading and setting up the SDK. The patched SDK is also accessible from this page that enables the EA Developers Kits on MCUXpresso. You can also download Getting Started with M.2 modules and i.MX RT to run the sample examples. Also install all the tools mentioned below:

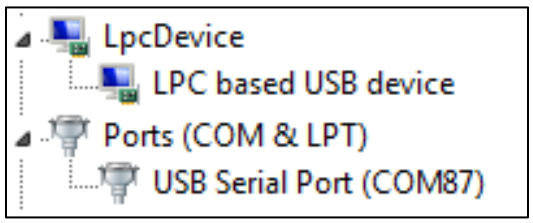
- MCUXpresso IDE
- FTDI Virtual COM Port Driver
- Terminal application (TeraTerm, Putty, etc.)

Figure 6: Embedded Artists Resource Web



You should see LPC based USB device and USB Serial Port on the Device Manager as below if you completed software installation successfully. The driver for LPC-Link2 is included in the MCUXpresso.

Figure 7: COM Port and JTAG of Embedded Artists Kit

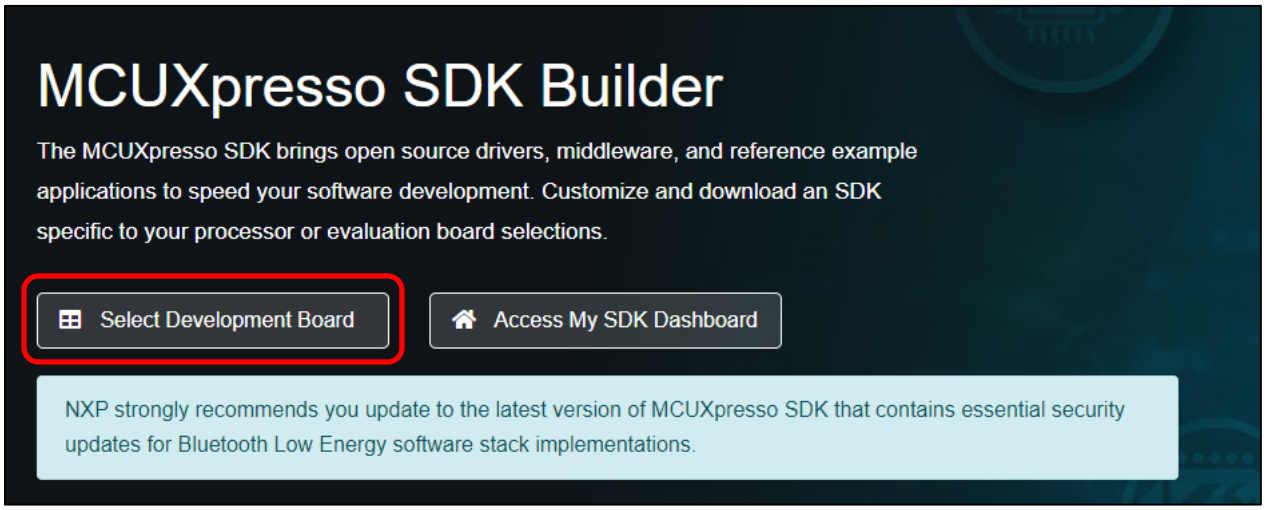


6 SDK Setup for MCUXpresso

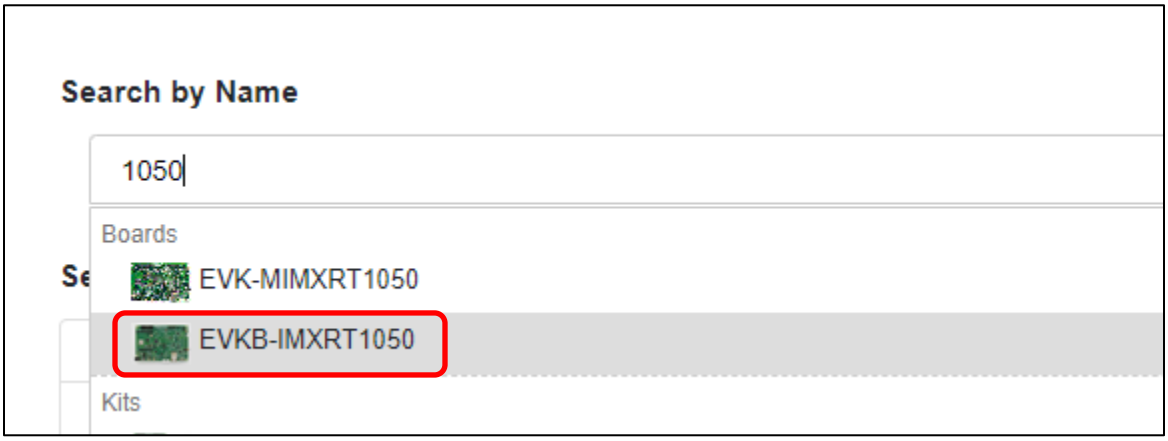
MCUXpresso supports various processors, so it requires appropriate SDK for i.MX RT. To support Embedded Artists' Wireless M.2 Modules, additional components (NXP Wi-Fi and other related components) are required. There are two ways to install SDK in the new MCUXpresso IDE. Follow the steps below carefully to install right components.

6.1 Drag and drop SDK in the IDE

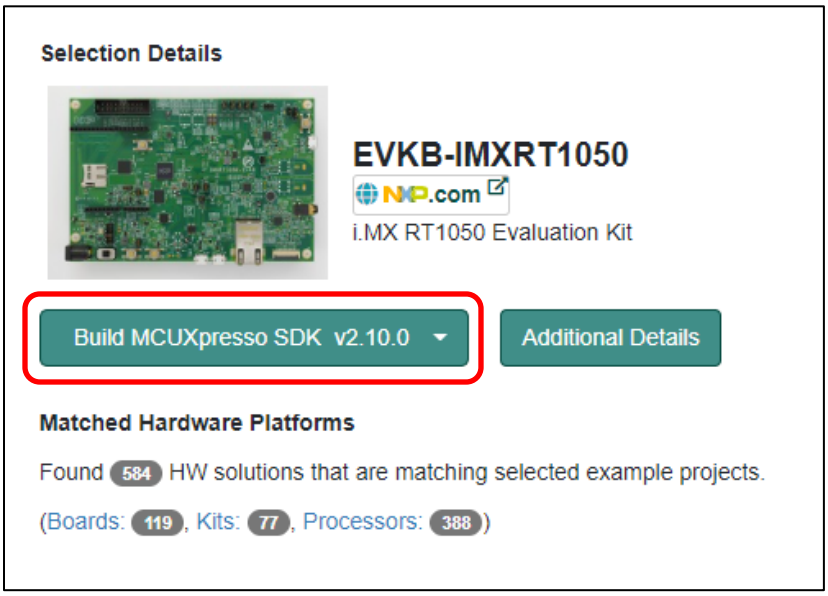
1. First download the MCUXpresso SDK by following this URL: https://mcuxpresso.nxp.com. You will need to login to your NXP account and then click "Select Development Board".



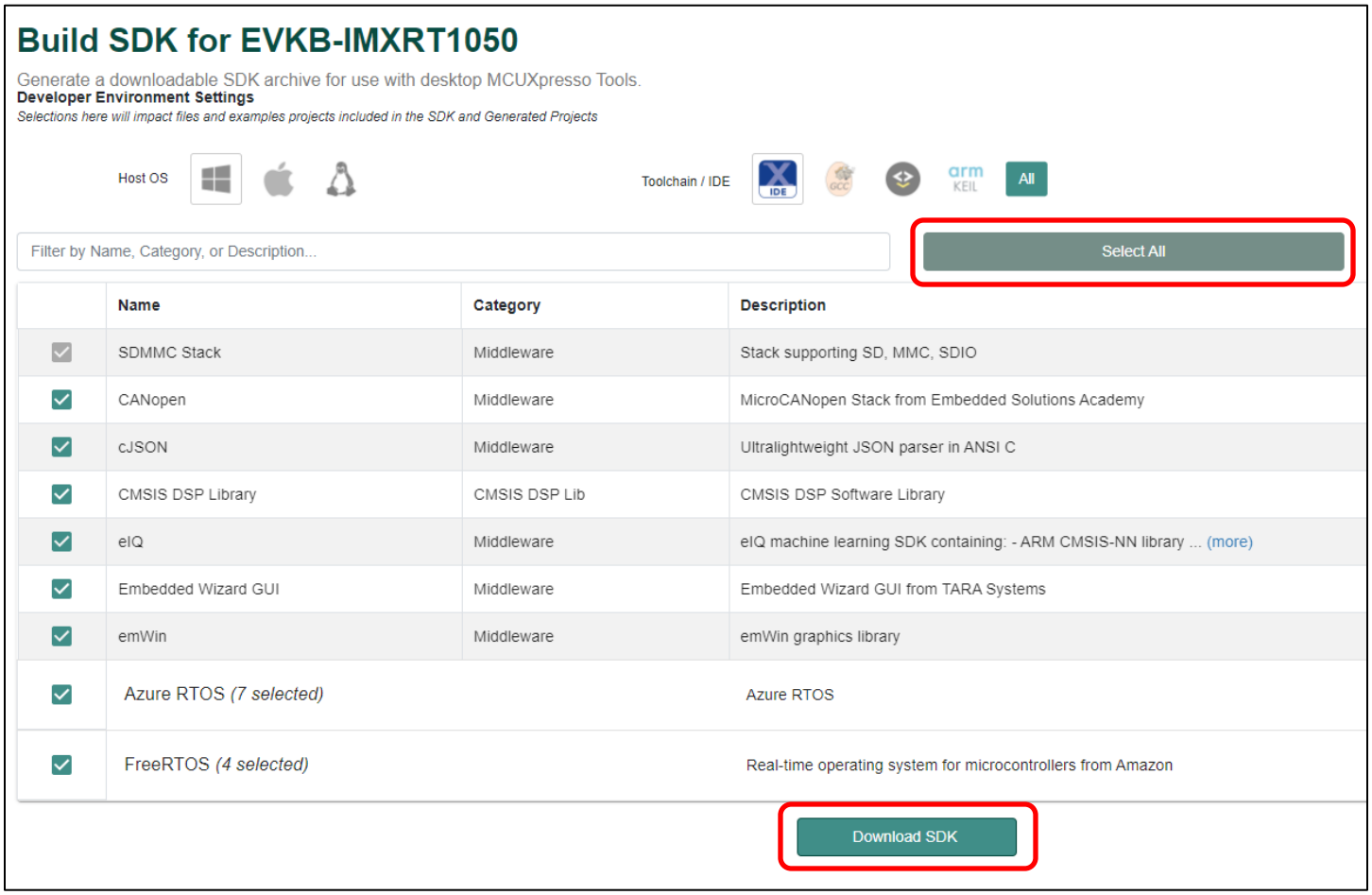
2. Type "1050", then select "EVKB-IMXRT1050" for i.MX RT 1050 EVKB.



3. On the right, then click “Build MCUXpresso SDK”

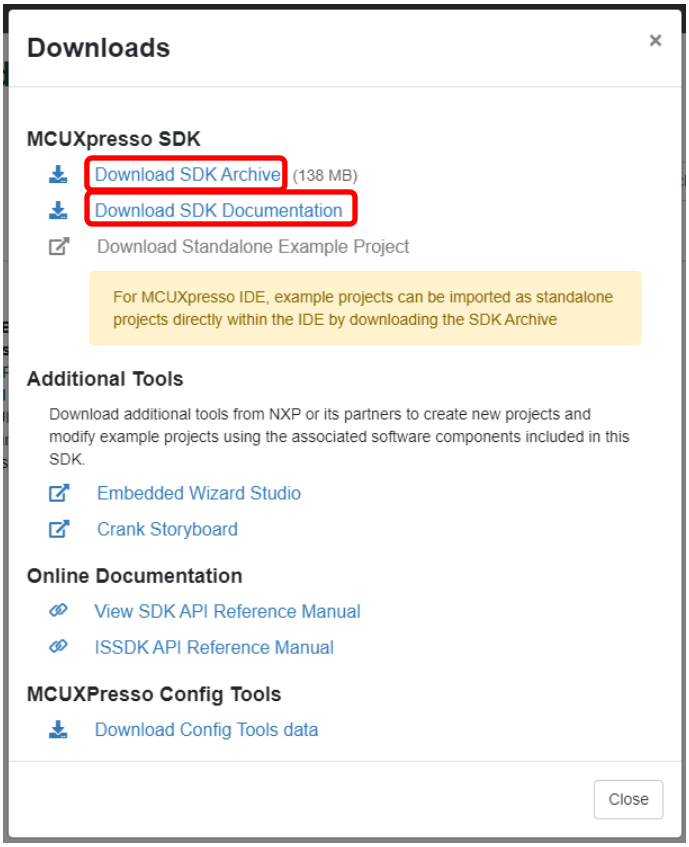


4. Click “Select All” and then select “Download SDK” to download the SDK.

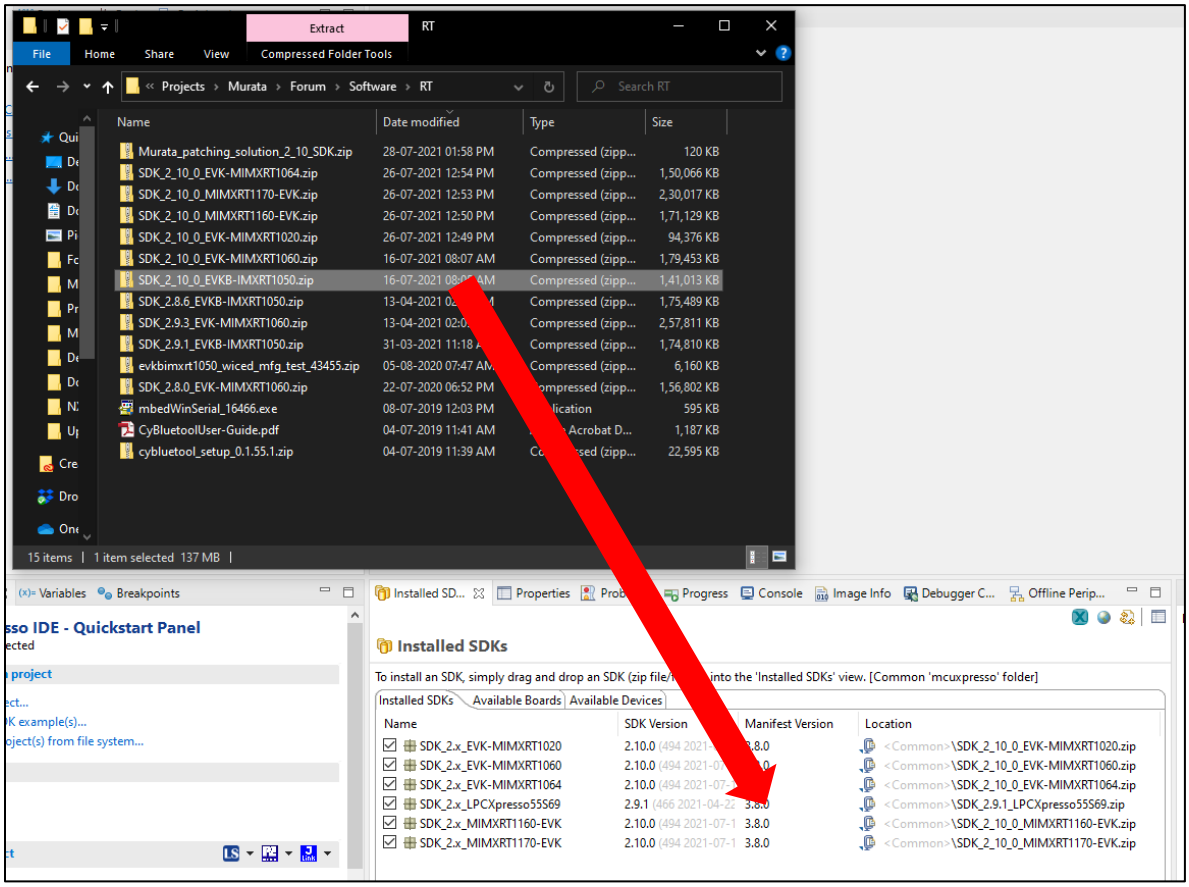


5. Agree to the EULA.

6. Click “Download SDK Archive” if download does not start automatically. You can also download SDK Documentation, then click “Close”.

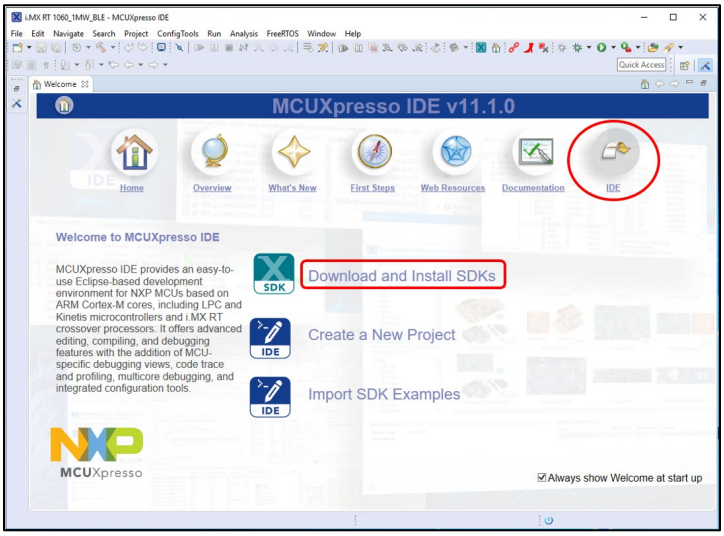


7. To install the SDK in the MCUXpresso, drag and drop the SDK Archive file on “Installed SDKs window”.

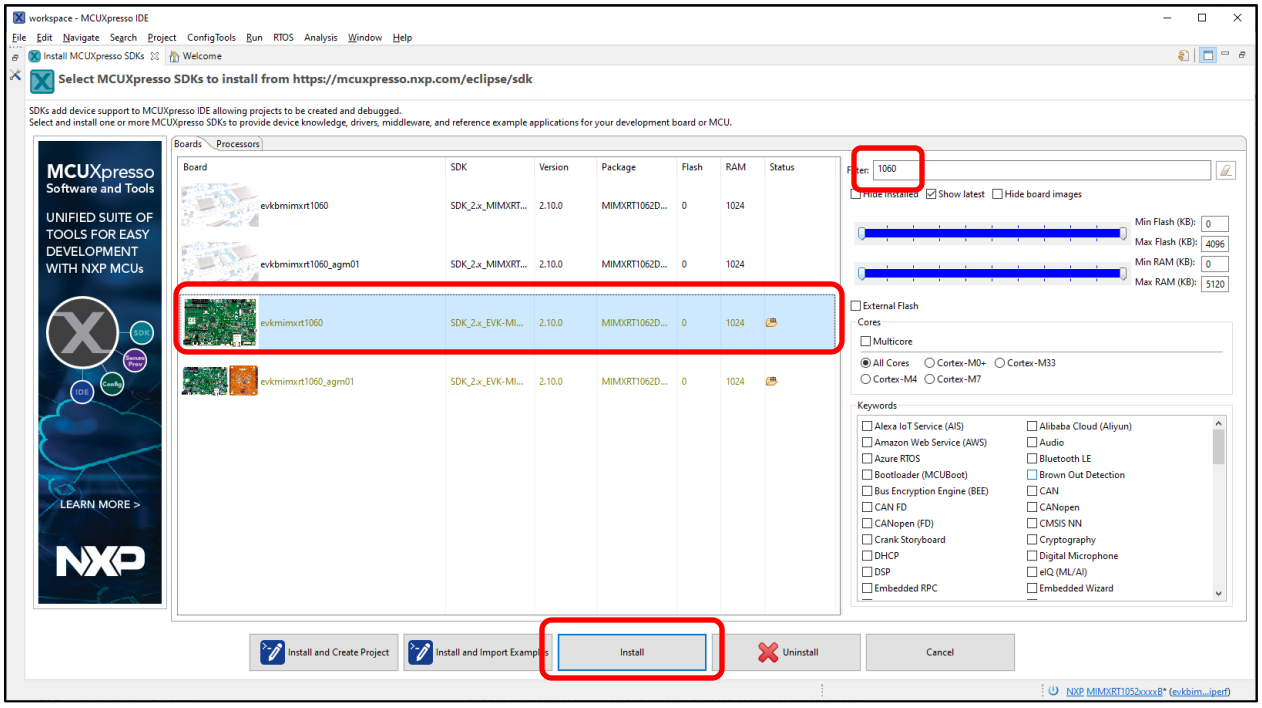


6.2 Install SDK directly from MCUXpresso IDE

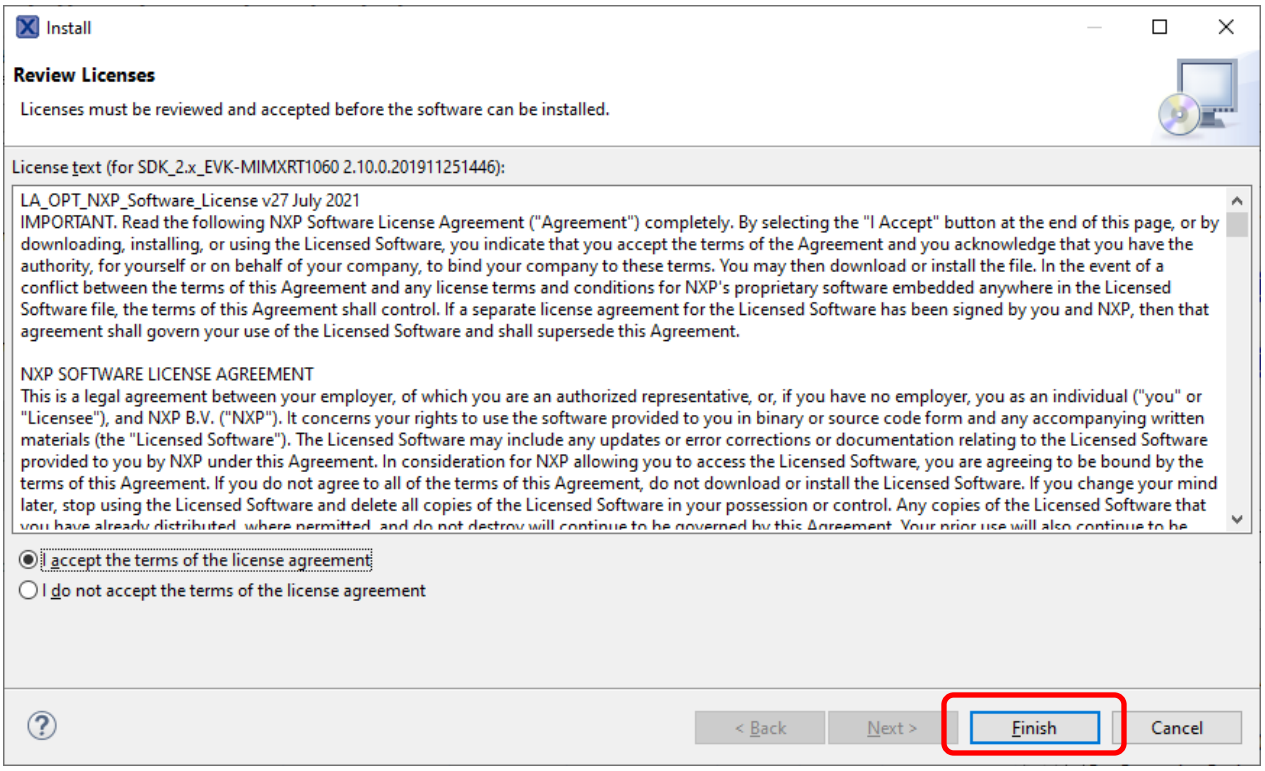
1. To install the SDK in the MCUXpresso IDE, click on “Download and Install SDKs”. To switch to normal IDE, click on IDE.



2. Type “1060” in the filter box. Click on “evkmimxrt1060” and click on “install” to download and install the SDK for i.MX RT 1060 EVK.



3. Accept the license agreement and click on ‘Finish’ to start the download.



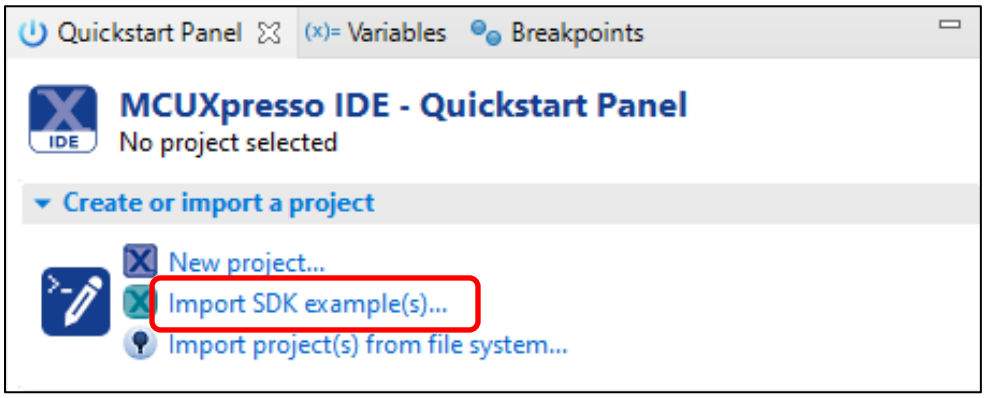
7 Patching the examples

The examples provided with MCUXpresso SDK (2.10 or later) does not support Murata modules out-of-the-box. Some of the example files need to be modified to enable the Murata modules. Murata has simplified this “patching” process by providing the modified files and a script based one-click patching mechanism.

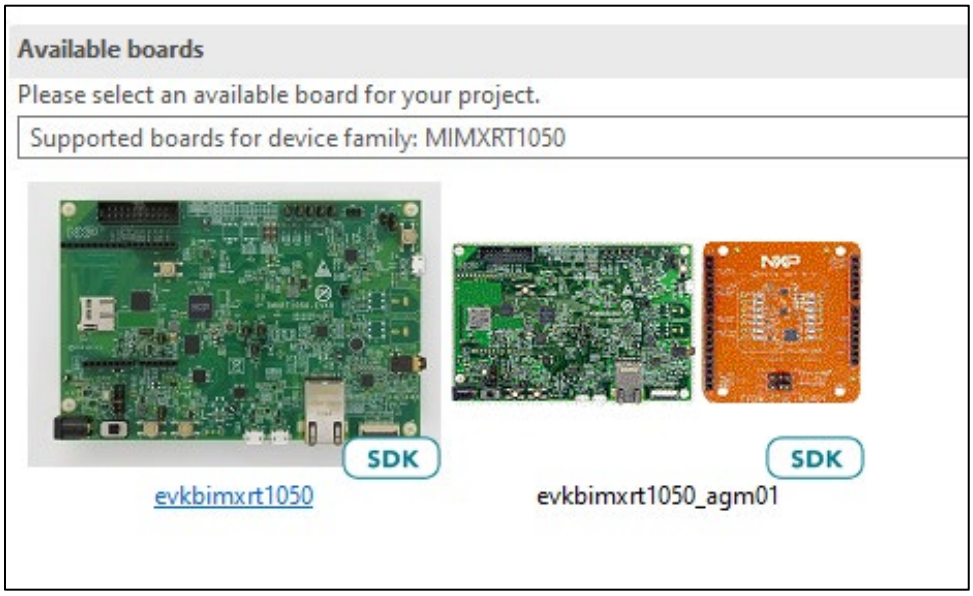
Note: Before executing the following steps, import all the exampled that needs to be patched for Murata module support. The following mechanism only modifies the examples currently present in the user workspace.

7.1 Import example(s)

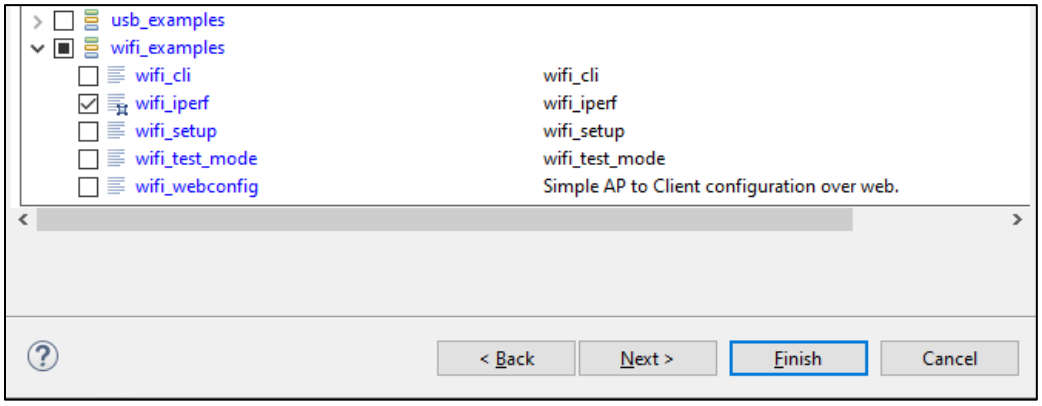
1. Click on **Import SDK example(s)…** in the Quickstart Panel. (If the Welcome page is still open, either close it, or click on the **IDE** icon).



2. Select the target EVK (e.g **evkbimxrt1050**) and click the **Next** button.

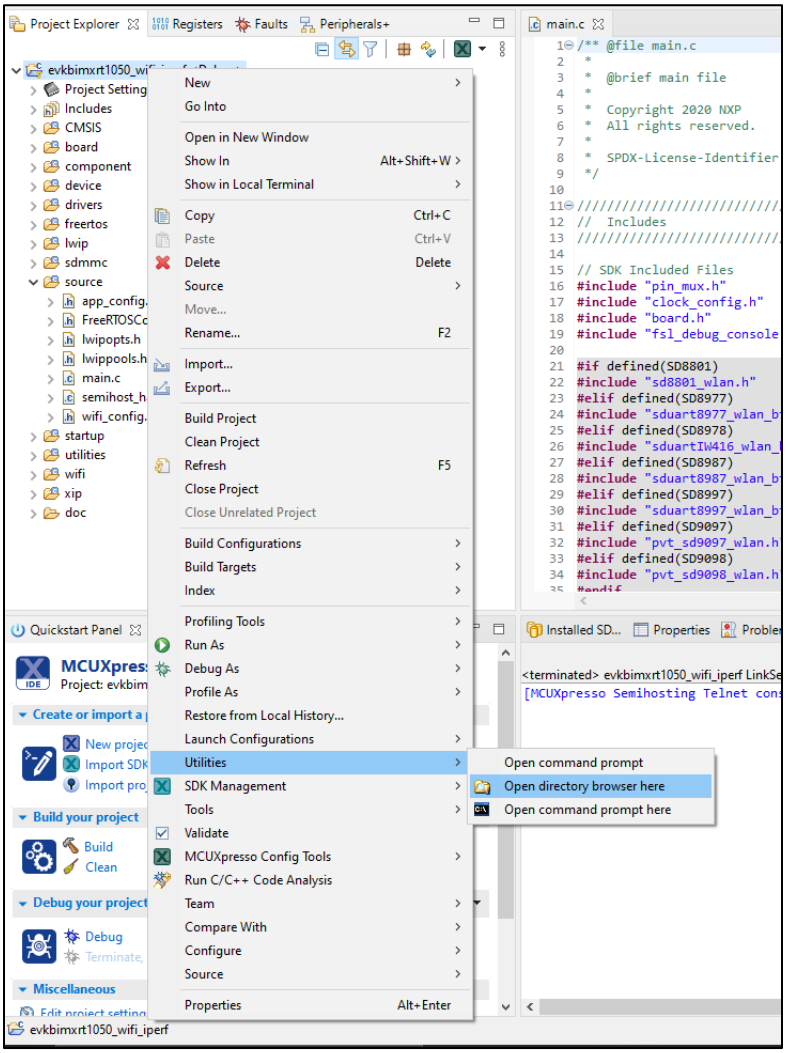


3. Expand **wifi_examples** and select one or more examples to import. Click **Finish**. Ensure **SDK Debug Console** is set as **UART** in **Project Options**.

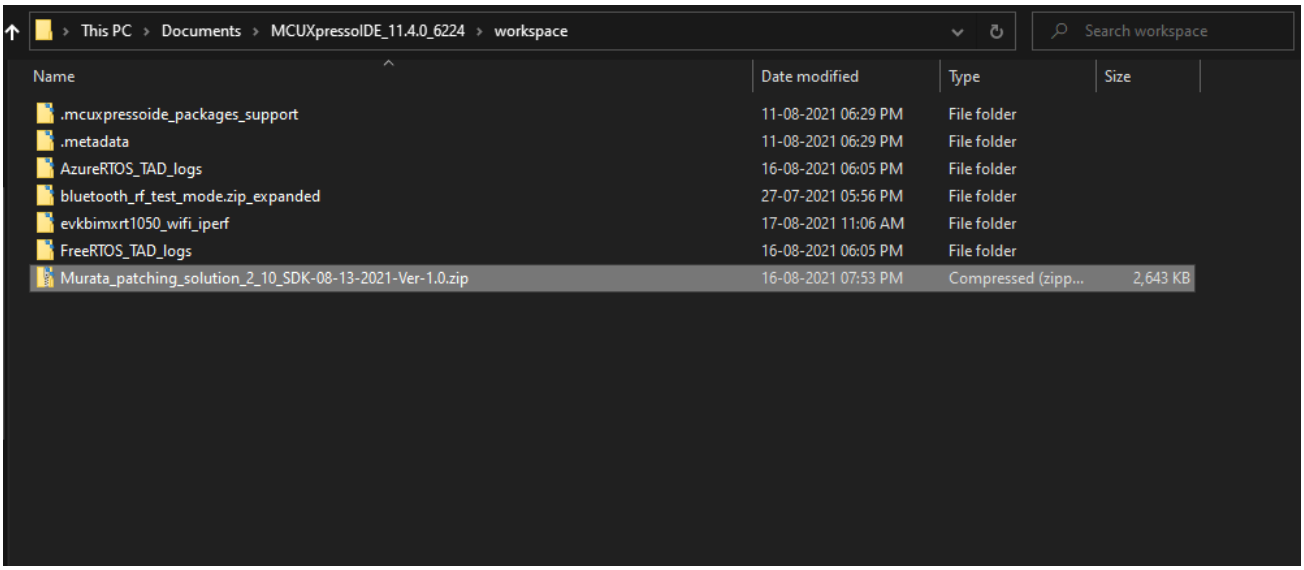


7.2 Modify the imported examples

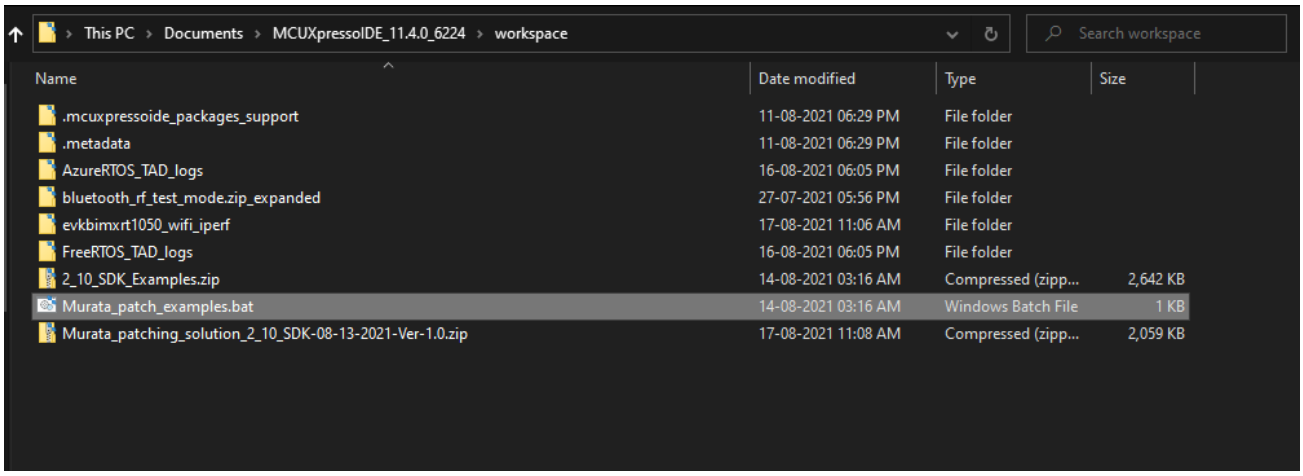
1. Right click on the imported project in the Project Explorer and select **Utilities -> Open directory browser here**.



2. The project source folder will open. Go to the parent folder (one level up). This is the workspace folder used by MCUXpresso. Download the Murata patch file.



3. Extract the downloaded file. Double click on the Murata_patch_example.bat file. This will replace all the example files necessary to enable Murata modules.



Note: This will patch all the valid examples in the workspace, including ones that may have been imported previously. The script provides a warning to the user to this effect and gives a chance to make backups if required.

For SDK 2.10, the following files are replaced (not all files apply to all examples) –

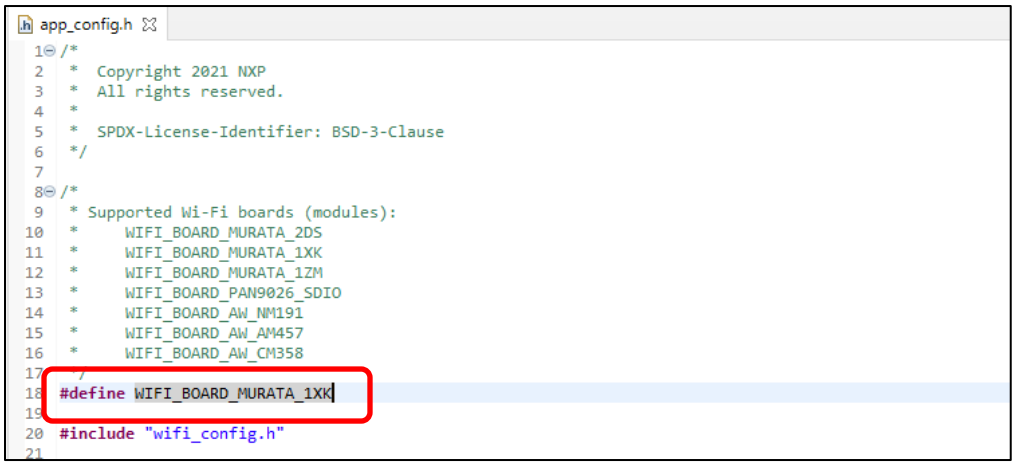
Example source root/

- **board**
  - pin_mux.c
  - pin_mux.h
- **doc**
  - readme.txt
- **source**
  - app_config.h
  - main.c
  - wifi_config.h
- **wifi**
  - **incl**
    - **wlcmgr**
      - wlan_enhanced_tests_murata_1XK_CA.h
      - wlan_enhanced_tests_murata_1XK_EU.h
      - wlan_enhanced_tests_murata_1XK_JP.h
      - wlan_enhanced_tests_murata_1XK_US.h
      - wlan_enhanced_tests_murata_1XK_WW.h
      - wlan_enhanced_tests_murata_1ZM_CA.h
      - wlan_enhanced_tests_murata_1ZM_EU.h
      - wlan_enhanced_tests_murata_1ZM_JP.h
      - wlan_enhanced_tests_murata_1ZM_US.h
      - wlan_enhanced_tests_murata_1ZM_WW.h
      - wlan_enhanced_tests_murata_2DS.h
      - wlan_enhanced_tests_murata_2DS_WW.h
      - wlan_txpwrlimit_cfg_murata_1XK_CA.h
      - wlan_txpwrlimit_cfg_murata_1XK_EU.h
      - wlan_txpwrlimit_cfg_murata_1XK_JP.h
      - wlan_txpwrlimit_cfg_murata_1XK_US.h
      - wlan_txpwrlimit_cfg_murata_1XK_WW.h
      - wlan_txpwrlimit_cfg_murata_1ZM_CA.h
      - wlan_txpwrlimit_cfg_murata_1ZM_EU.h
      - wlan_txpwrlimit_cfg_murata_1ZM_JP.h
      - wlan_txpwrlimit_cfg_murata_1ZM_US.h
      - wlan_txpwrlimit_cfg_murata_1ZM_WW.h
      - wlan_txpwrlimit_cfg_murata_2DS_CA.h
      - wlan_txpwrlimit_cfg_murata_2DS_EU.h
      - wlan_txpwrlimit_cfg_murata_2DS_JP.h
      - wlan_txpwrlimit_cfg_murata_2DS_US.h
      - wlan_txpwrlimit_cfg_murata_2DS_WW.h
  - **wlcmgr**
    - wlan.c
    - wlan_enhanced_tests.c
    - wlan_txpwrlimit_cfg.c

7.3 Selecting the module

Enable the correct compiler flag in source/app_config.h file to select the Murata module - **WIFI_BOARD_MURATA_1XK / WIFI_BOARD_MURATA_2DS / WIFI_BOARD_MURATA_1ZM** (for connecting via uSD-M.2 adapter, valid for all EVKs)

or **WIFI_BOARD_MURATA_1XK_M2 / WIFI_BOARD_MURATA_2DS_M2 / WIFI_BOARD_MURATA_1ZM_M2** (for directly connecting via M.2 interface, valid for only i.MX RT 1160 and 1170 EVKs)

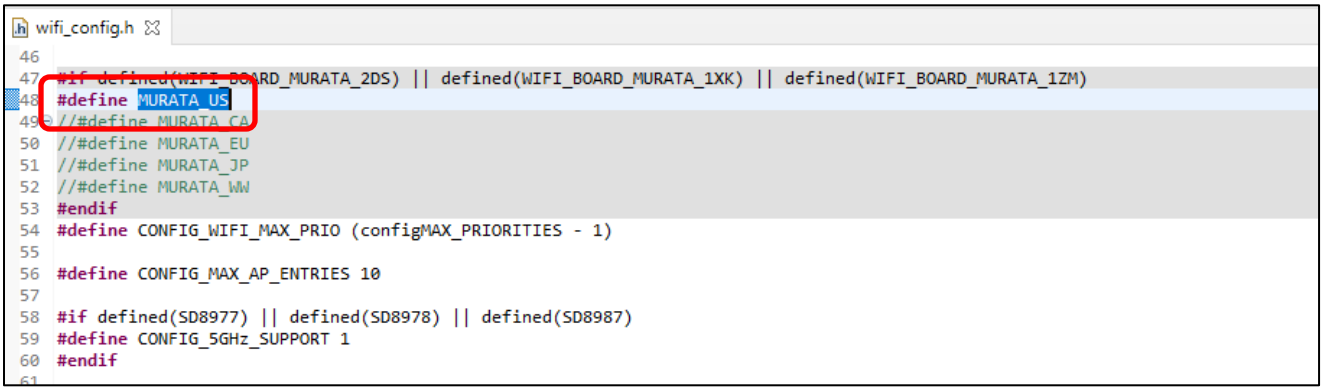


7.4 Selecting the region

By default, US region (FCC) is used for TX power limit values in Wi-Fi. However, the following regions are supported by the Murata modules.

- US (FCC)
- Canada (IC)
- European Union
- Japan
- Worldwide Safe List

Edit the file source/wifi_config.h and select the required macro from the selection (**MURATA_US**, **MURATA_EU**, **MURATA_CA**, **MURATA_JP, MURATA_WW**).



8 Running Sample Applications

Various sample applications are provided by the SDK. For example, there are seven Wi-Fi examples included in the 2.10 SDK for i.MX RT 1060 EVK:

- **evkmimxrt1060_wifi_cert**: Provides CLI access to common and advanced Wi-Fi certification test operations.
- **evkmimxrt1060_wifi_cli**: Provides CLI access to common Wi-Fi operations, including throughput measurements.
- **evkmimxrt1060_wifi_cli_fw_dump**: Provides Wi-Fi CLI Firmware dump example to demonstrates the ability to capture a FW memory dump on a micro-USB memory device connected to the host platform.
- **evkmimxrt1060_wifi_iperf**: Provides RF measurement tests.
- **evkmimxrt1060_wifi_setup**: Provides a simple Wi-Fi setup demo (scan-connect-ping).
- **evkmimxrt1060_wifi_test_mode**: Provides CLI access to common Wi-Fi certification test operations.
- **evkmimxrt1060_wifi_webconfig**: Provides a STA + SoftAP test with an embedded web server.

Additionally, there are 15 Bluetooth examples included in the 2.10 SDK for i.MX RT 1060 EVK.

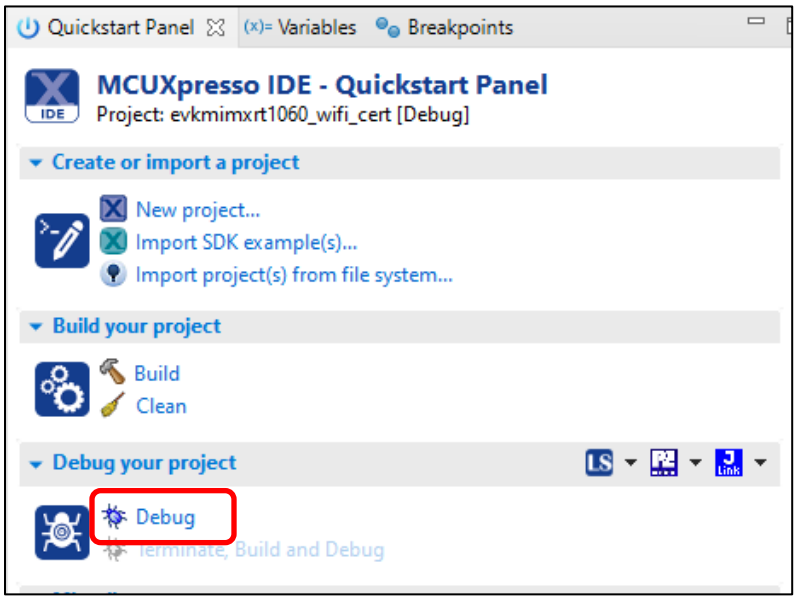
- **evkmimxrt1060_a2dp_sink**: Demonstrates how to use the a2dp sink feature.
- **evkmimxrt1060_a2dp_source**: Demonstrates how to use the a2dp source feature.
- **evkmimxrt1060_audio_profile**: Demonstrates an audio control application using AWS cloud.
- **evkmimxrt1060_central_hpc**: Demonstrates basic BLE Central role functionality, connecting to a HPS Server.
- **evkmimxrt1060_central_ht**: Demonstrates basic BLE Central role functionality, connecting to a health thermometer sensor.
- **evkmimxrt1060_central_ipsp**: Demonstrates basic BLE Central role functionality, connecting to an IPSP Service.
- **evkmimxrt1060_central_pxm**: Demonstrates basic BLE Central role functionality, connecting to a Proximity Reporter.
- **evkmimxrt1060_handsfree**: Demonstrates the HFP HF basic functionality.
- **evkmimxrt1060_handsfree_ag**: Demonstrates the HFP Ag basic functionality.
- **evkmimxrt1060_peripheral_hps**: Demonstrates basic BLE Peripheral role functionality, exposing an HTTP Proxy GATT Service.
- **evkmimxrt1060_peripheral_ht**: Demonstrates basic BLE Peripheral role functionality, exposing a Health Thermometer GATT Service.
- **evkmimxrt1060_peripheral_ipsp**: Demonstrates basic BLE Peripheral role functionality, exposing an Internet Protocol Support GATT Service.
- **evkmimxrt1060_peripheral_pxr**: Demonstrates basic BLE Peripheral role functionality, exposing a Proximity Reporter GATT Service.
- **evkmimxrt1060_spp**: Demonstrates how to use the SPP feature.
- **evkmimxrt1060_wifi_provisioning**: Demonstrates how the EVK Wi-Fi can be configured by Android mobile application, via Bluetooth.

The following sections detail the process of running some of the examples on i.MX RT 1060 EVK. The procedures will be similar in other EVKs as well, for the examples available for them.

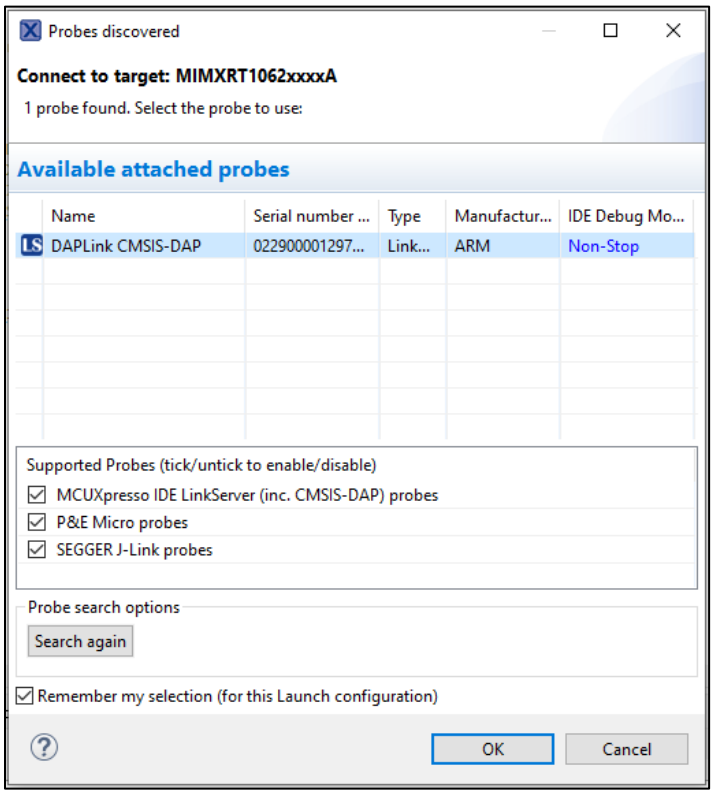
8.1 Example evkmimxrt1060_wifi_cert

1. Refer to **Section 7** to import and patch the evkmimxrt1060_wifi_cert example, and select the current module (2DS, 1XK or 1ZM).

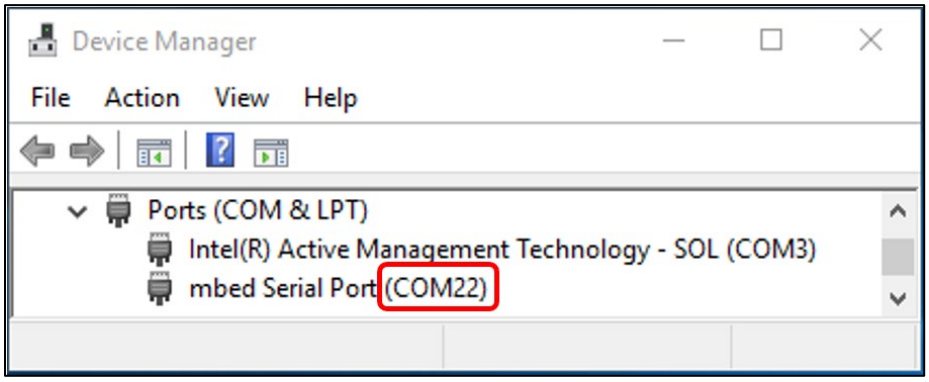
2. Click Debug in the QuickStart Panel.



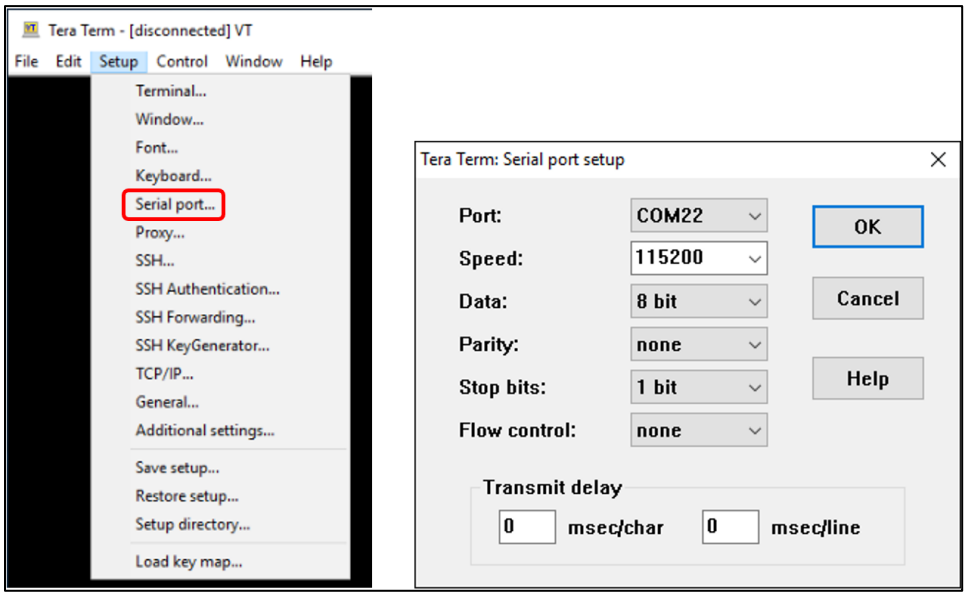
3. For the very first attempt, you need to select the appropriate JTAG adapter. Select the adapter and click OK, and then wait for a while for the build. You can see the log on console window.



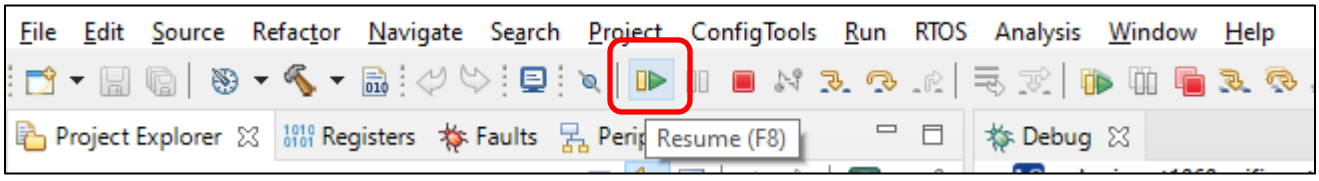
4. After the Debug process is complete, open the “Device Manager” in windows and go to “Ports (COM & LPT)” to check the mbed COM port number. In this case, COM 22 is our COM port number.



5. Now the example is ready to run. Open Tera Term on the appropriate COM port (i.e. COM 22 in this case). Configure port for 115200 bps, 8 bits data, no parity, and 1 stop bit (115200/8/N/1).



6. Click resume button in MCUXpresso.



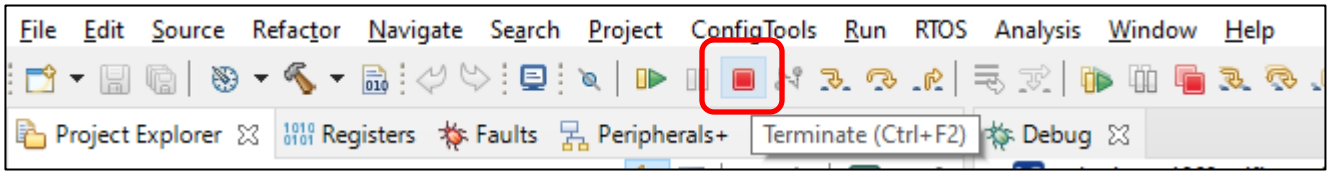
7. You should see this output from i.MX RT.

```
=======================================
app_cb: WLAN initialized
=======================================
WLAN CLIs are initialized
=======================================
ENHANCED WLAN CLIs are initialized
=======================================
CLIs Available:
=======================================

help
wlan-version
wlan-mac
wlan-scan
wlan-scan-opt ssid <ssid> bssid ...
wlan-add <profile_name> ssid <ssid> bssid...
wlan-remove <profile_name>
wlan-list
wlan-connect <profile_name>
wlan-start-network <profile_name>
wlan-stop-network
wlan-disconnect
wlan-stat
wlan-info
wlan-address
wlan-get-uap-channel
wlan-get-uap-sta-list
wlan-ieee-ps <0/1>
wlan-deep-sleep-ps <0/1>
wlan-set-regioncode <region-code>
wlan-get-regioncode
wlan-get-txpwrlimit <subband>
wlan-set-txpwrlimit
wlan-set-chanlist-and-txpwrlimit
wlan-set-chanlist
wlan-get-chanlist
wlan-set-txratecfg <format> <index> <nss>
wlan-get-txratecfg
wlan-get-data-rate
wlan-set-pmfcfg <mfpc> <mfpr>
wlan-get-pmfcfg
wlan-set-antcfg <ant mode> [evaluate_time]
wlan-get-antcfg
wlan-set-ed-mac-mode <ed_ctrl_2g> <ed_offset_2g> <ed_ctrl_5g> <ed_offset_5g>
wlan-get-ed-mac-mode
ping [-s <packet_size>] [-c <packet_count>] [-W <timeout in sec>] <ip_address>
iperf [-s|-c <host>|-a|-h] [options]
dhcp-stat
=======================================

#
```

8. Type any of the command(s) and press enter to execute.

```
# wlan-version
WLAN Version : w8987o-V0, RF878X, FP91, 16.91.10

# wlan-info
Station not connected
uAP not started

# wlan-mac
MAC address
D4:53:83:BE:4A:9E

#
```

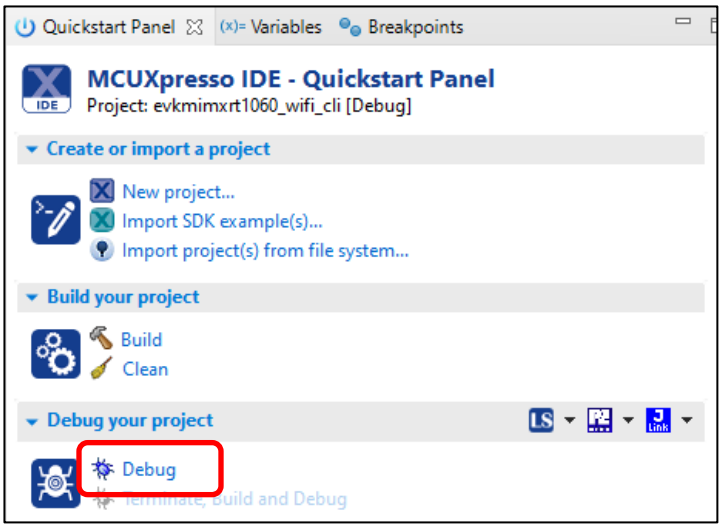
9. Click Terminate button in MCUXpresso to stop the test.



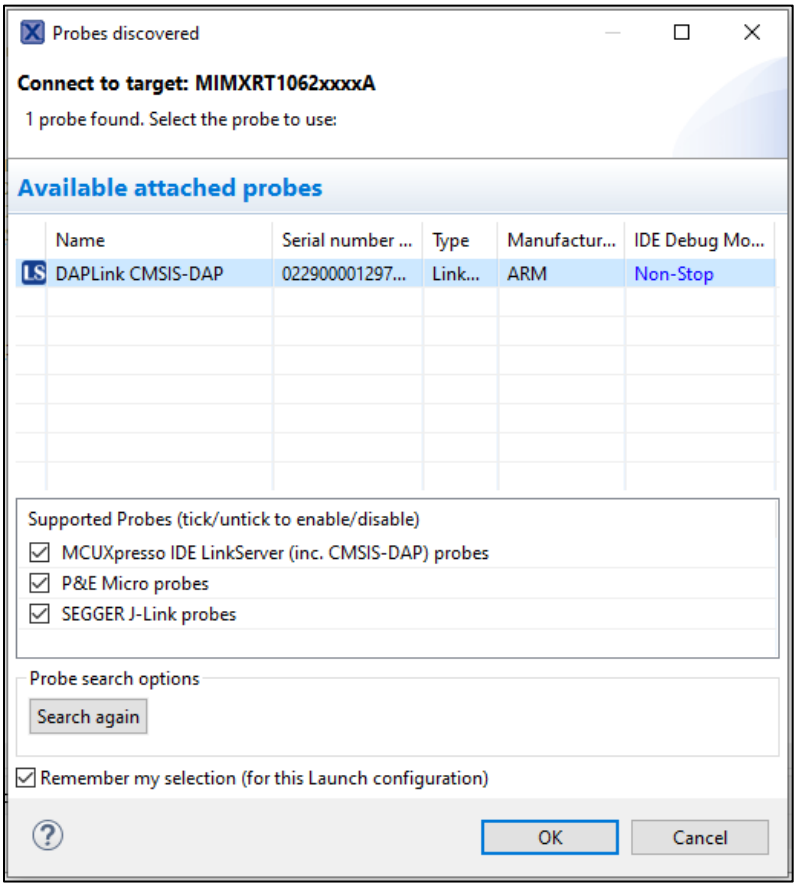
8.2 Example evkmimxrt1060_wifi_cli

1. Refer to **Section 7** to import and patch the evkmimxrt1060_wifi_cli example, and select the current module (2DS, 1XK or 1ZM).

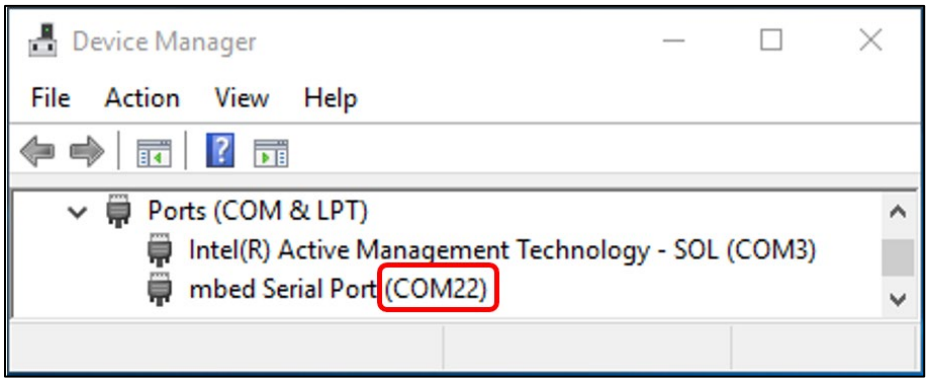
2. Click Debug in the QuickStart Panel.



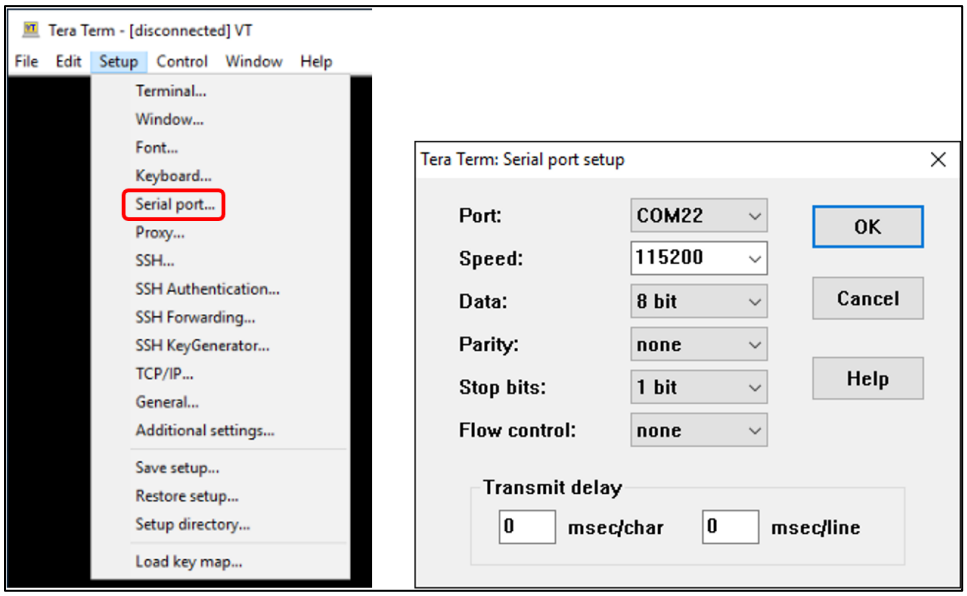
3. For the very first attempt, you need to select the appropriate JTAG adapter. Select the adapter and click OK, and then wait for a while for the build. You can see the log on console window.



4. After the Debug process is complete, open the “Device Manager” in windows and go to “Ports (COM & LPT)” to check the mbed COM port number. In this case, COM 22 is our COM port number.



5. Now the example is ready to run. Open Tera Term on the appropriate COM port (i.e. COM 22 in this case). Configure port for 115200 bps, 8 bits data, no parity, and 1 stop bit (115200/8/N/1).



6. To download the iperf, follow the following link:

For Windows: https://iperf.fr/download/windows/iperf-2.0.5-win32.zip
For Linux: Follow the steps to install iperf.

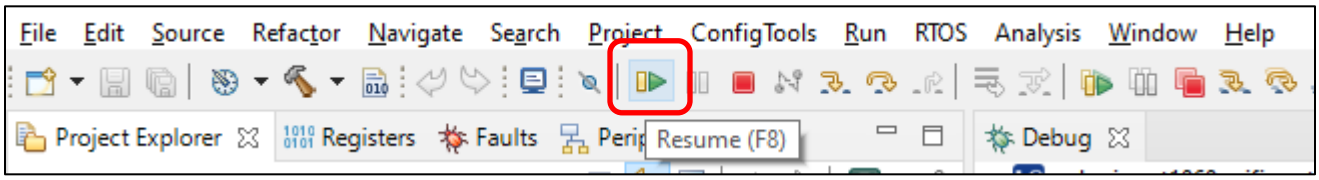
1. Download the .deb file for iperf 2.05 via https://iperf.fr/download/ubuntu/iperf_2.0.5+dfsg1-2_amd64.deb

2. Change directory to where the .deb file is.

3. Run the following commands to install it.

```
$ dpkg -I iperf_2.0.5+dfsg1-2_amd64.deb
$ sudo dpkg -i iperf_2.0.5+dfsg1-2_amd64.deb
$ sudo apt install iperf
```

7. Click resume button in MCUXpresso.



8. You should see this output from i.MX RT.

```
=========================================
wifi cli demo
=========================================
Initialize CLI
=========================================
Initialize WLAN Driver
=========================================
MAC Address: D4:53:83:BE:4A:9E
[net] Initialized TCP/IP networking stack
=========================================
app_cb: WLAN: received event 10
=========================================
app_cb: WLAN initialized
=========================================
WLAN CLIs are initialized
=========================================
CLIs Available:
=========================================

help
wlan-version
wlan-mac
wlan-scan
wlan-scan-opt ssid <ssid> bssid ...
wlan-add <profile_name> ssid <ssid> bssid...
wlan-remove <profile_name>
wlan-list
wlan-connect <profile_name>
wlan-start-network <profile_name>
wlan-stop-network
wlan-disconnect
wlan-stat
wlan-info
wlan-address
wlan-get-uap-channel
wlan-get-uap-sta-list
wlan-ieee-ps <0/1>
wlan-deep-sleep-ps <0/1>
ping [-s <packet_size>] [-c <packet_count>] [-W <timeout in sec>] <ip_address>
iperf [-s|-c <host>|-a|-h] [options]
dhcp-stat
=========================================
```

9. Type any of the command(s) and press enter to execute.

```
# wlan-version
WLAN Driver Version   : v1.3.r33.p2
WLAN Firmware Version : w8987o-V0, RF878X, FP91, 16.91.10.p200, WPA2_CVE_FIX 1, PVE_FIX 1

# wlan-ieee-ps 1
Turned on IEEE Power Save mode
#
```

10. Some common actions are:

    a. Scan Wi-Fi networks

```
# wlan-scan
```

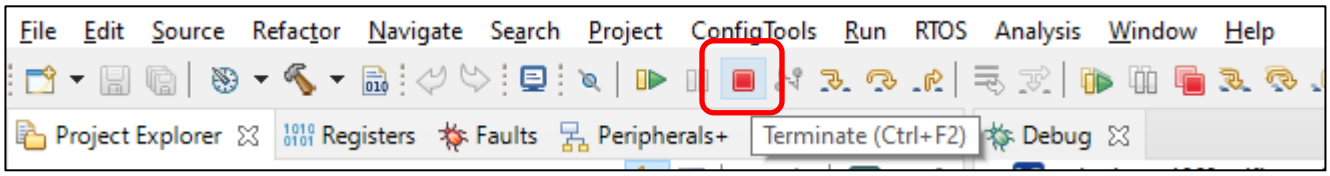
    b. Connect to an Open AP

```
# wlan-add test_network ssid <AP SSID>
# wlan-connect test_network
```

    c. Ping host after connection

```
 # ping <target host IP>
```

11. Click Terminate button in MCUXpresso to stop the test.



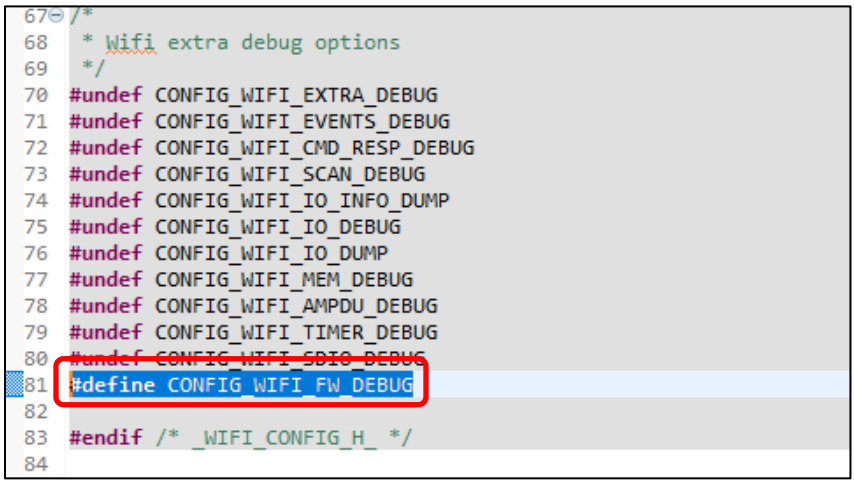
8.3 Example evkmimxrt1060_wifi_cli_fw_dump

Note: This example is almost similar to the evkmimxrt1060_wifi_cli example in terms of output and behavior. However, this example has the added capability of storing a FW memory dump on an external mass storage device in case of any errors.

1. Refer to **Section 7** to import and patch the evkmimxrt1060_wifi_cli_fw_dump example, and select the current module (2DS, 1XK or 1ZM).

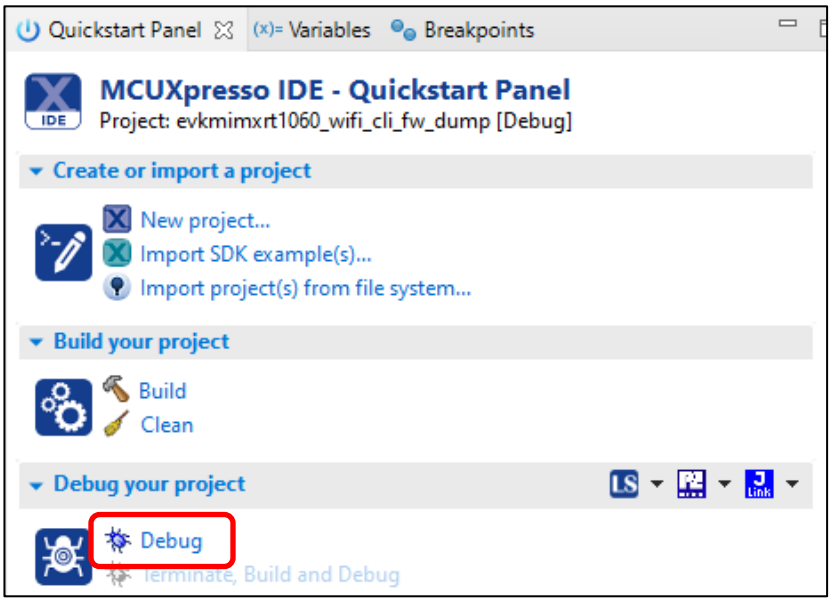
2. Connect a micro–USB Mass Storage Device to the host platform's USB OTG (J9) slot. Make sure the mass storage device has an USB 2.0 interface and is formatted as FatFS.

3. Edit the file source/wifi_config.h and enable the compiler flag **CONFIG_WIFI_FW_DEBUG** (line 81). Save the changes.

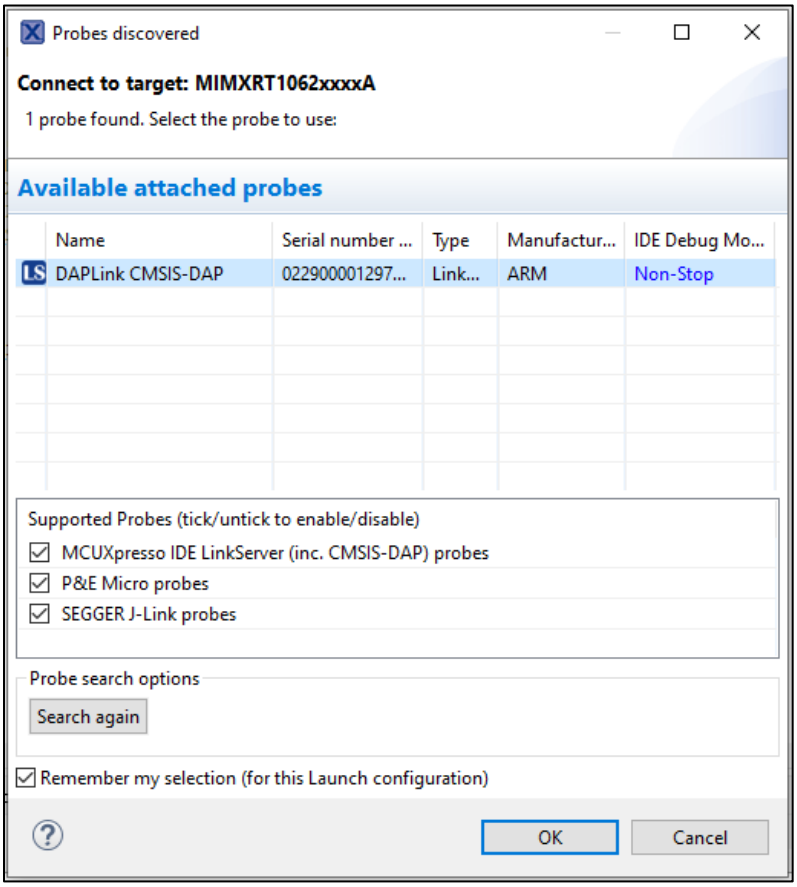


```
67 /*
68  * Wifi extra debug options
69  */
70 #undef CONFIG_WIFI_EXTRA_DEBUG
71 #undef CONFIG_WIFI_EVENTS_DEBUG
72 #undef CONFIG_WIFI_CMD_RESP_DEBUG
73 #undef CONFIG_WIFI_SCAN_DEBUG
74 #undef CONFIG_WIFI_IO_INFO_DUMP
75 #undef CONFIG_WIFI_IO_DEBUG
76 #undef CONFIG_WIFI_IO_DUMP
77 #undef CONFIG_WIFI_MEM_DEBUG
78 #undef CONFIG_WIFI_AMPDU_DEBUG
79 #undef CONFIG_WIFI_TIMER_DEBUG
80 #undef CONFIG_WIFI_SDIO_DEBUG
81 #define CONFIG_WIFI_FW_DEBUG
82
83 #endif /* _WIFI_CONFIG_H_ */
84
```

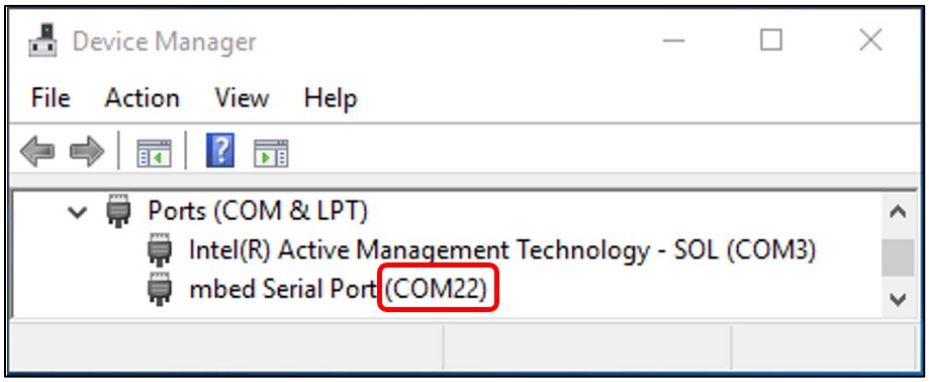
4. Click Debug in the QuickStart Panel.



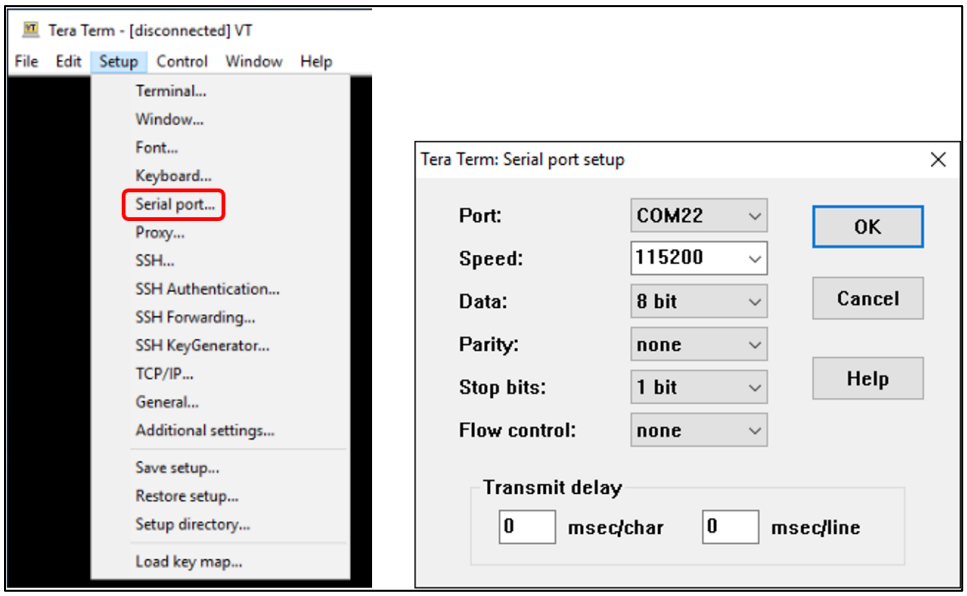
5. For the very first attempt, you need to select the appropriate JTAG adapter. Select the adapter and click OK, and then wait for a while for the build. You can see the log on console window.



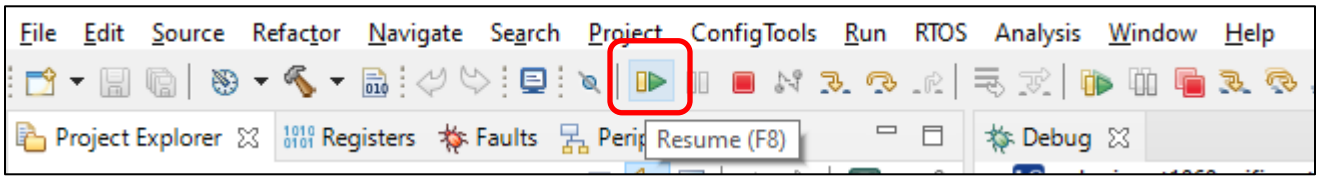
6. After the Debug process is complete, open the “Device Manager” in windows and go to “Ports (COM & LPT)” to check the mbed COM port number. In this case, COM 22 is our COM port number.



7. Now the example is ready to run. Open Tera Term on the appropriate COM port (i.e. COM 22 in this case). Configure port for 115200 bps, 8 bits data, no parity, and 1 stop bit (115200/8/N/1).



8. Click resume button in MCUXpresso.



9. You should see this output from i.MX RT.

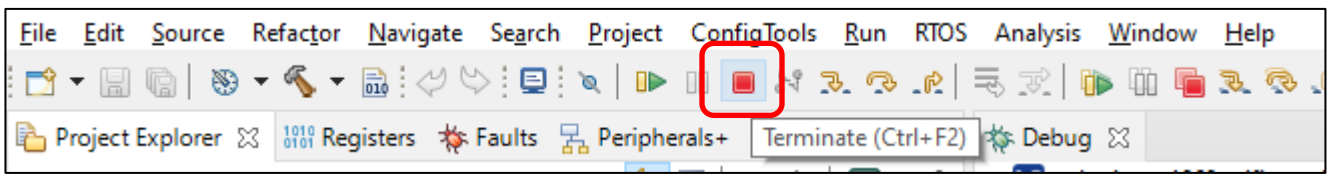
```
=========================================
wifi cli demo
=========================================
Initialize CLI
=========================================
Initialize WLAN Driver
=========================================
MAC Address: D4:53:83:BE:4A:9E
[net] Initialized TCP/IP networking stack
=========================================
app_cb: WLAN: received event 10
=========================================
app_cb: WLAN initialized
=========================================
WLAN CLIs are initialized
=========================================
CLIs Available:
=========================================

help
wlan-version
wlan-mac
wlan-scan
wlan-scan-opt ssid <ssid> bssid ...
wlan-add <profile_name> ssid <ssid> bssid...
wlan-remove <profile_name>
wlan-list
wlan-connect <profile_name>
wlan-start-network <profile_name>
wlan-stop-network
wlan-disconnect
wlan-stat
wlan-info
wlan-address
wlan-get-uap-channel
wlan-get-uap-sta-list
wlan-ieee-ps <0/1>
wlan-deep-sleep-ps <0/1>
ping [-s <packet_size>] [-c <packet_count>] [-W <timeout in sec>] <ip_address>
iperf [-s|-c <host>|-a|-h] [options]
dhcp-stat
=========================================
```

10. Type any of the command(s) and press enter to execute.

```
# wlan-version
WLAN Driver Version   : v1.3.r33.p2
WLAN Firmware Version : w8987o-V0, RF878X, FP91, 16.91.10.p200, WPA2_CVE_FIX 1, PVE_FIX 1

# wlan-ieee-ps 1
Turned on IEEE Power Save mode
#
```

11. Whenever a Wi-Fi firmware or SDIO communication failure occurs, the FW dump will be stored on the connected mass storage device.

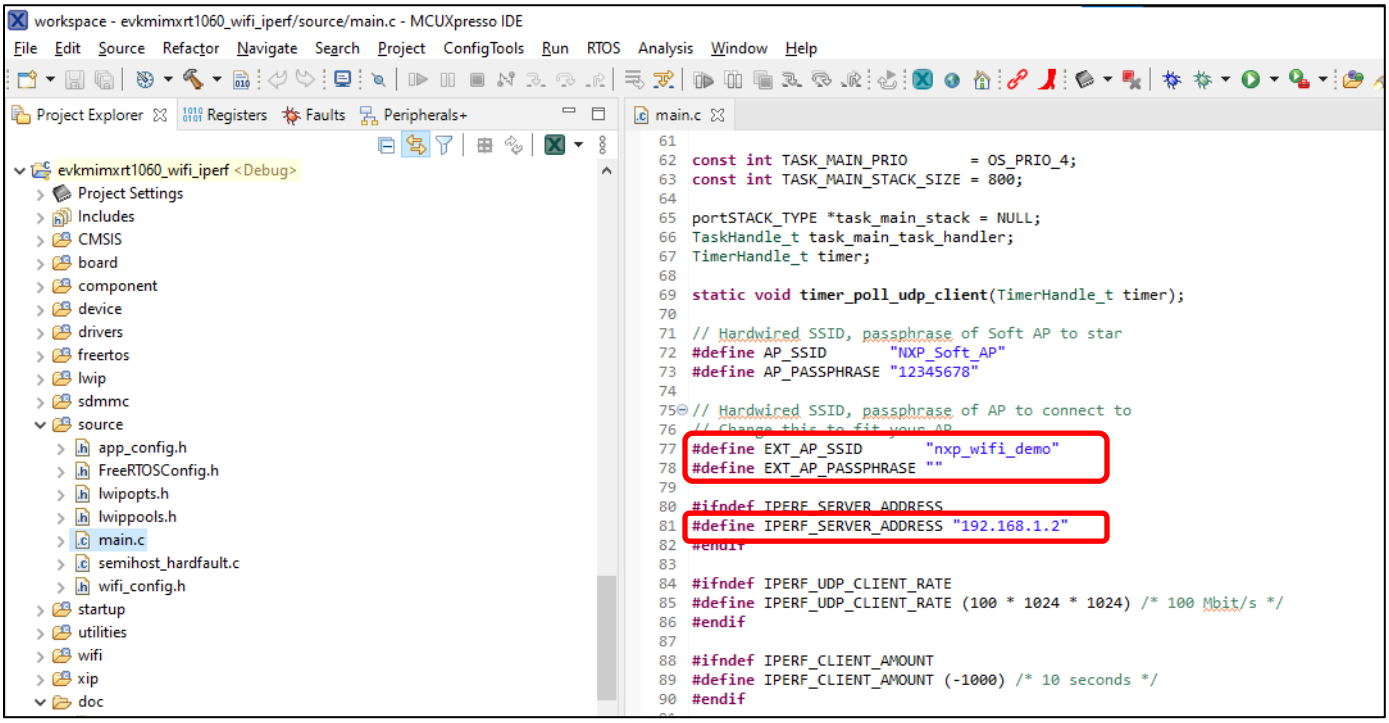
12. Click Terminate button in MCUXpresso to stop the test.



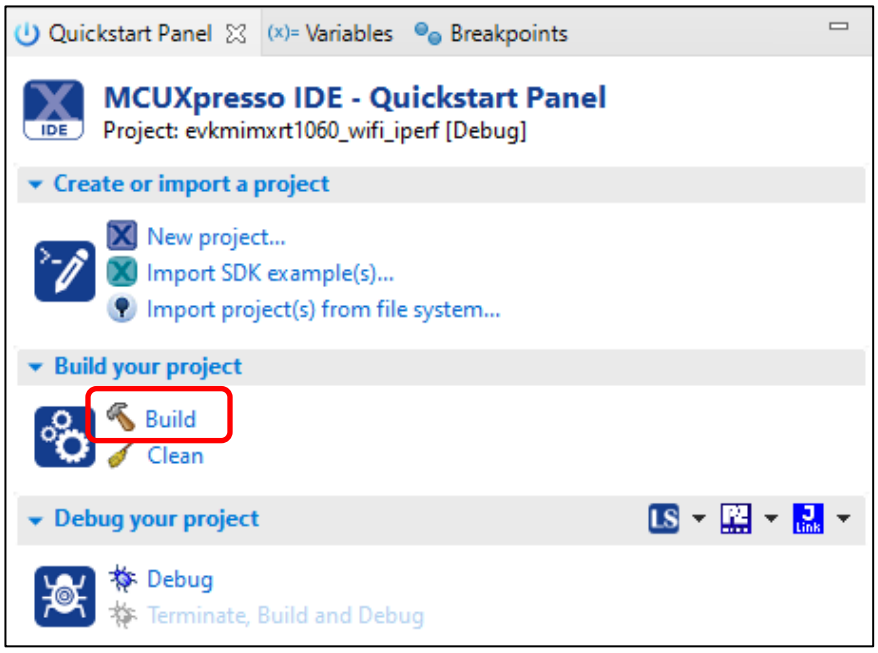
8.4 Example evkmimxrt1060_wifi_iperf

1. Refer to **Section 7** to import and patch the evkmimxrt1060_wifi_iperf example, and select the current module (2DS, 1XK or 1ZM).

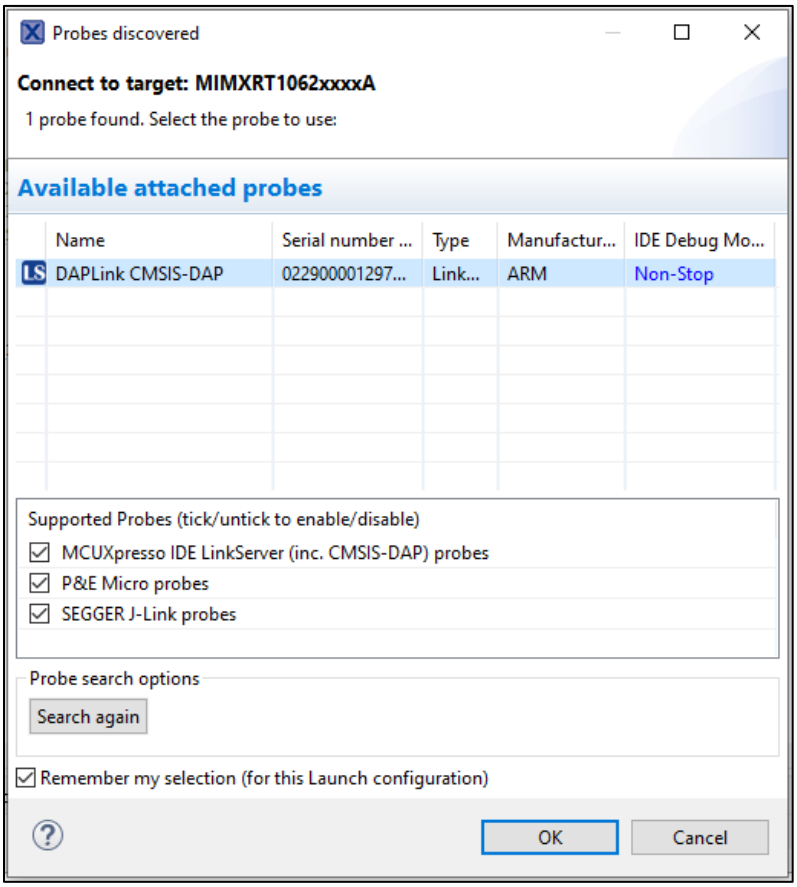
2. Edit the source/main.c file to modify the EXT_AP_SSID, EXT_AP_PASSPHRASE and IPERF_SERVER_ADDRESS to match the test setup.



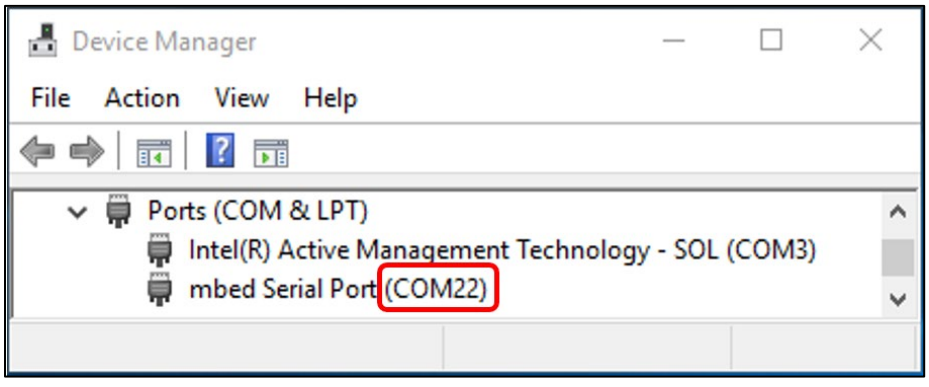
3. Click Debug in the QuickStart Panel.



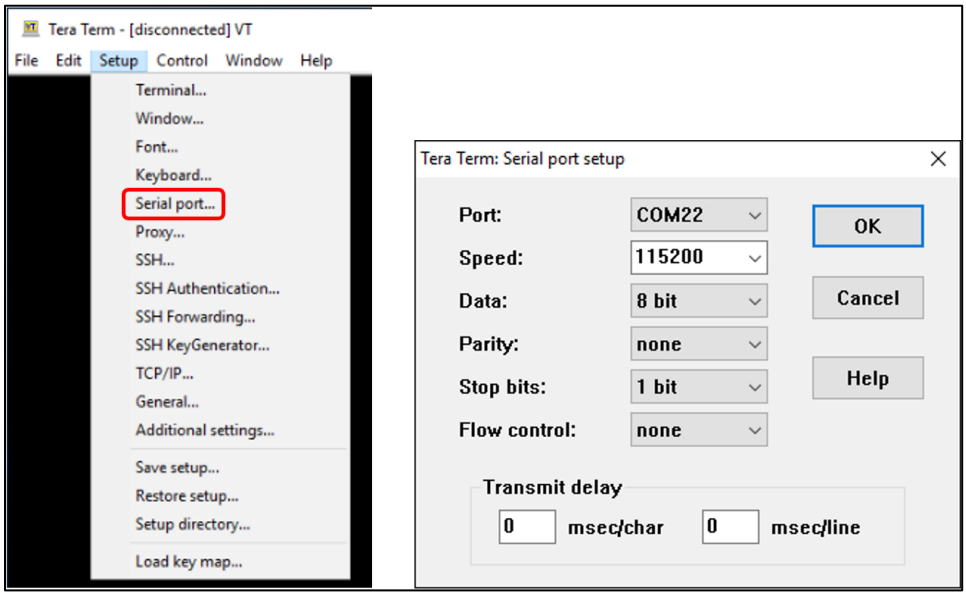
4. For the very first attempt, you need to select the appropriate JTAG adapter. Select the adapter and click OK, and then wait for a while for the build. You can see the log on console window.



5. After the Debug process is complete, open the “Device Manager” in windows and go to “Ports (COM & LPT)” to check the mbed COM port number. In this case, COM 22 is our COM port number.



6. Now the example is ready to run. Open Tera Term on the appropriate COM port (i.e. COM 22 in this case). Configure port for 115200 bps, 8 bits data, no parity, and 1 stop bit (115200/8/N/1).



7. To download the iperf, follow the following link:

For Windows: https://iperf.fr/download/windows/iperf-2.0.5-win32.zip
For Linux: Follow the steps to install iperf.

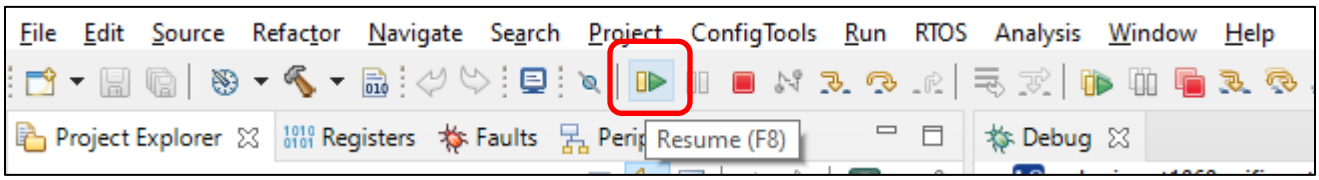
1. Download the .deb file for iperf 2.05 via https://iperf.fr/download/ubuntu/iperf_2.0.5+dfsg1-2_amd64.deb

2. Change directory to where the .deb file is.

3. Run the following commands to install it.

```
$ dpkg -I iperf_2.0.5+dfsg1-2_amd64.deb
$ sudo dpkg -i iperf_2.0.5+dfsg1-2_amd64.deb
$ sudo apt install iperf
```

8. Click resume button in MCUXpresso.



9. You should see this output from i.MX RT.

```
=======================================
wifi iperf demo
=======================================
Initialize WLAN Driver
Setting up new cal data
MAC Address: 2C:4C:C6:F4:D4:40
=======================================
For Soft AP demonstration
Start a Soft AP using option "A" in WPA2 security mode from menu
This also starts DHCP Server with IP 192.168.10.1, NETMASK 255.255.255.0
=======================================
For Station demonstration
Start an External AP with SSID as "nxp_wifi_demo" in Open mode
Start DHCP Server on External AP
Station network is configured with Dynamic address assignment
Application provides IPerf support
Set IPERF_SERVER_ADDRESS while using as IPerf Client
=======================================
  A  Start Soft AP
  S  Stop Soft AP
  s  Start Scan for external APs
  c  Connect to External AP (SSID='nxp_wifi_demo')
  D  Disconnect from External AP
  I  Enable IEEE PS on Station
  i  Disable IEEE PS on Station
  d  Enable Deep sleep on Station
  e  Disable Deep sleep on Station
  p  Print All Network info
  P  Print DHCP Server info
  1  TCP server mode (RX only test)
  2  TCP client mode (TX only test)
  3  TCP client dual mode (TX and RX in parallel)
  4  TCP client tradeoff mode (TX and RX sequentially)
  5  UDP server mode (RX only test)
  6  UDP client mode (TX only test)
  7  UDP client dual mode (TX and RX in parallel)
  8  UDP client tradeoff mode (TX and RX sequentially)
  h  Help (print this menu)
  H  Print extended help
[net] Initialized TCP/IP networking stack
=======================================
app_cb: WLAN: received event 10
=======================================
app_cb: WLAN initialized
=======================================
WLAN Driver Version   : v1.3.r33.p2
WLAN Firmware Version : IW416-V0, RF878X, FP91, 16.91.10.p214, WPA2_CVE_FIX 1, PVE_FIX 1
=======================================
```

10. Type ‘c’ to connect.

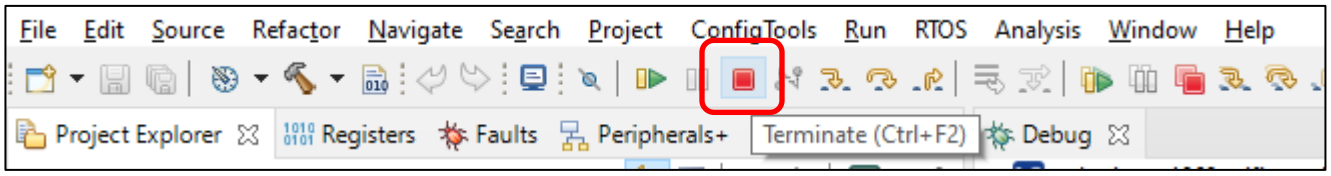
```
=======================================
Key 'c': Connect to External AP (SSID='Murata_5')
Connecting to Murata_5 .....=======================================
app_cb: WLAN: received event 0
=======================================
app_cb: WLAN: connected to network
Connected to following BSS:
SSID = [Murata_5], IP = [192.168.1.142]
=======================================
```

11. Type ‘4’ to perform a TCP bidirectional iPerf test. The available test options are:

    1: TCP server mode (RX only test)
    2: TCP client mode (TX only test)
    3: TCP client dual mode (TX and RX in parallel)
    4: TCP client tradeoff mode (TX and RX sequentially)
    5: UDP server mode (RX only test)
    6: UDP client mode (TX only test)
    7: UDP client dual mode (TX and RX in parallel)
    8: UDP client tradeoff mode (TX and RX sequentially)

```
Key '4': TCP client tradeoff mode (TX and RX sequentially)
------------------------------------------------
 TCP_DONE_CLIENT (TX)
 Local address : 192.168.1.142  Port 49153
 Remote address : 192.168.1.147  Port 5001
 Bytes Transferred 27894784
 Duration (ms) 10000
 Bandwidth (Mbitpsec) 22

New TCP client (settings flags 0x30313233)

------------------------------------------------
 TCP_DONE_SERVER (RX)
 Local address : 192.168.1.142  Port 5001
 Remote address : 192.168.1.147  Port 42654
 Bytes Transferred 35650148
 Duration (ms) 10037
 Bandwidth (Mbitpsec) 28
```

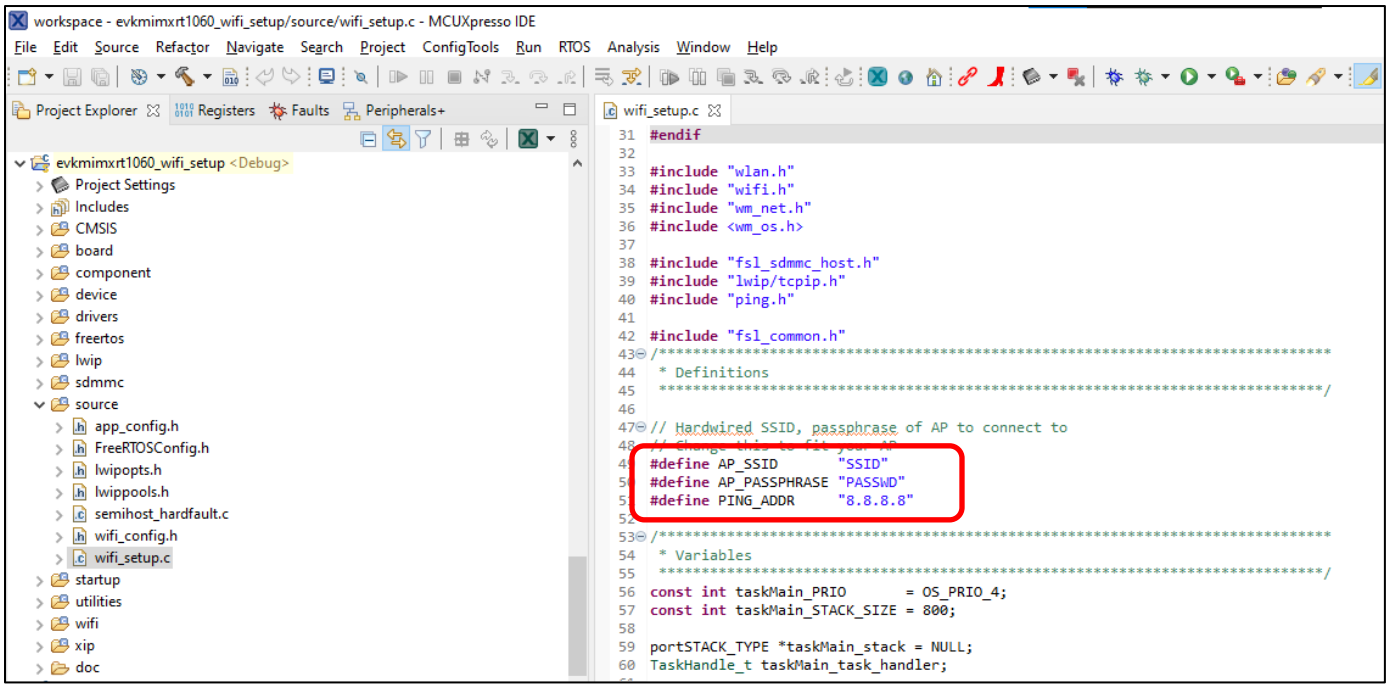
12. Click Terminate button in MCUXpresso to stop the test.



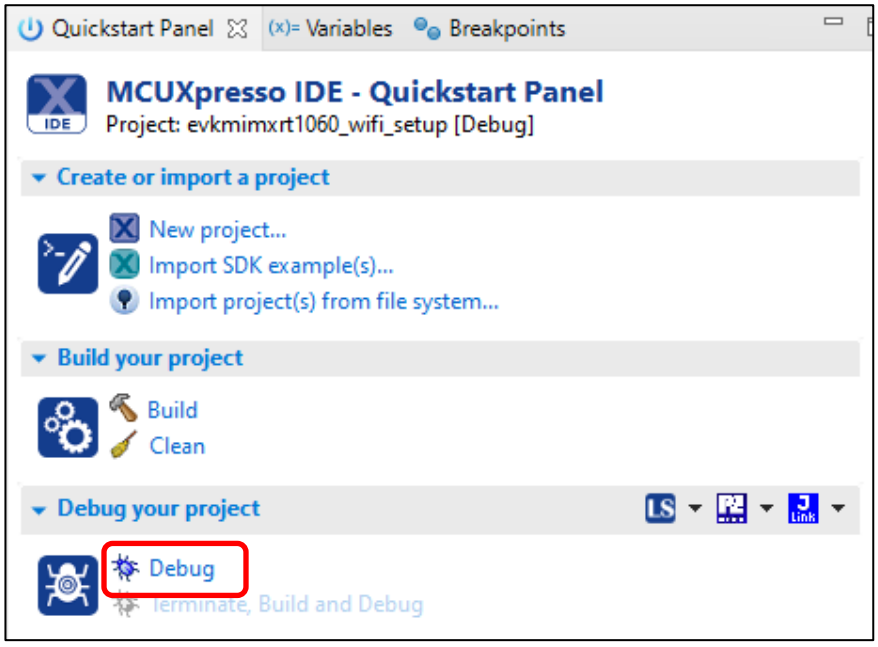
8.5 Example evkmimxrt1060_wifi_setup

1. Refer to **Section 7** to import and patch the evkmimxrt1060_wifi_setup example, and select the current module (2DS, 1XK or 1ZM).

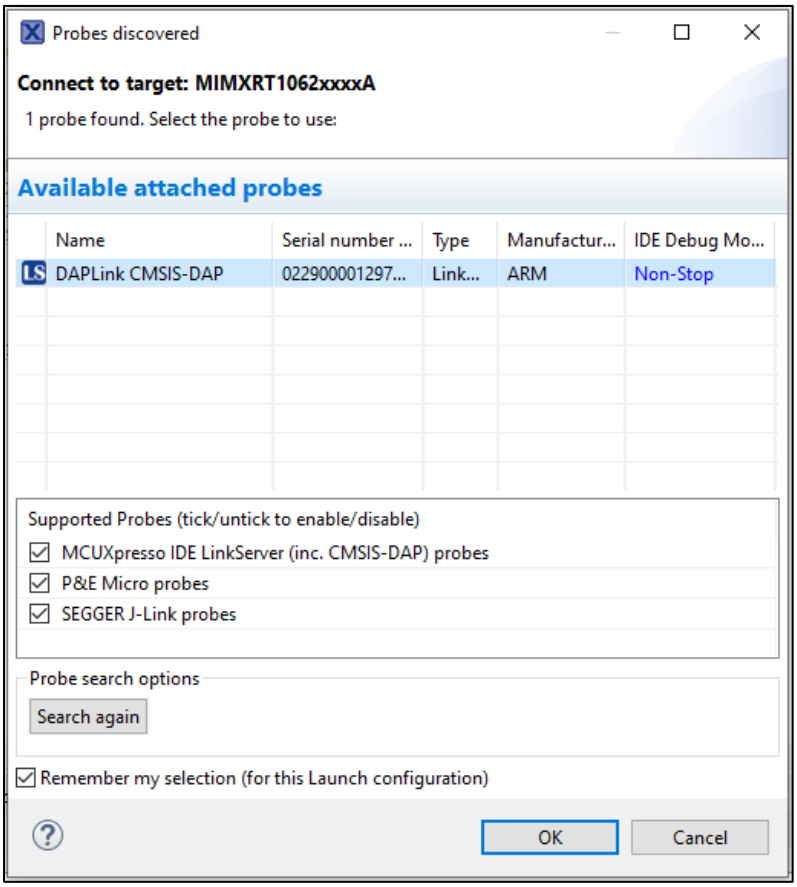
2. Edit the source/wifi_setup.c file to modify the AP_SSID, AP_PASSPHRASE and PING_ADDR to match the test setup.



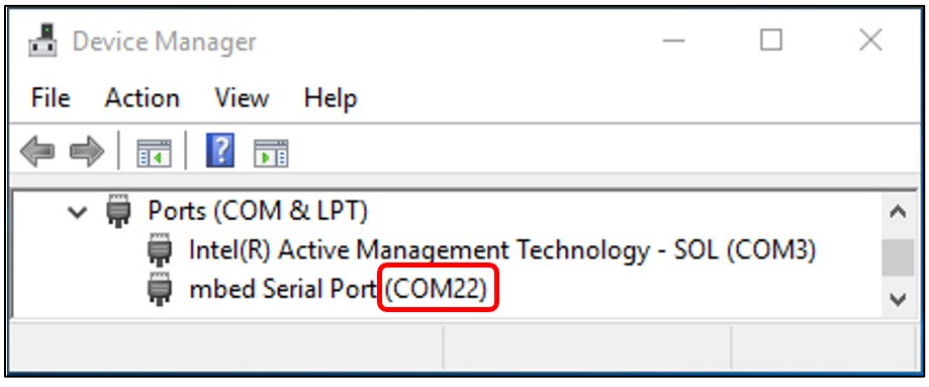
3. Click Debug in the QuickStart Panel.



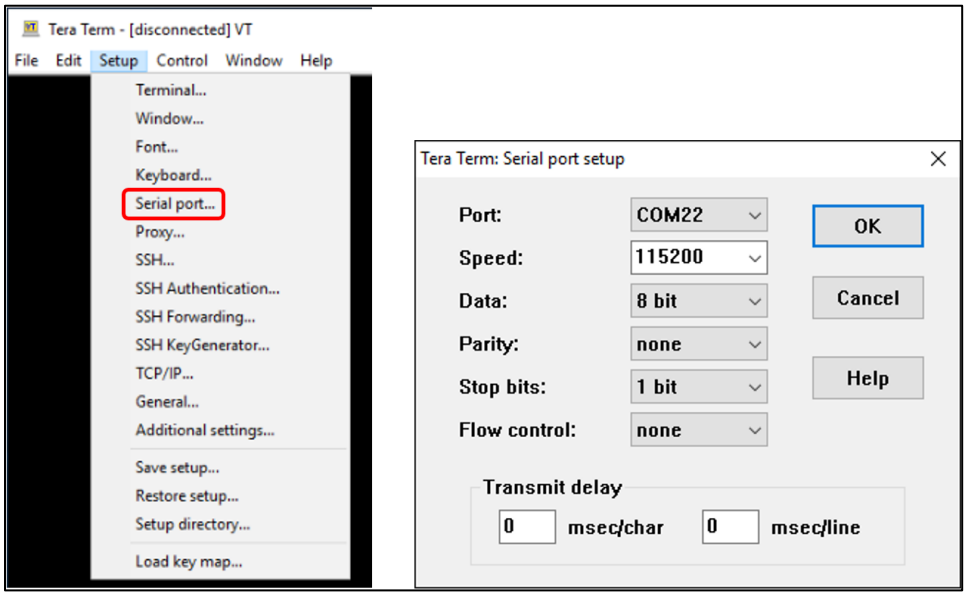
4. For the very first attempt, you need to select the appropriate JTAG adapter. Select the adapter and click OK, and then wait for a while for the build. You can see the log on console window.



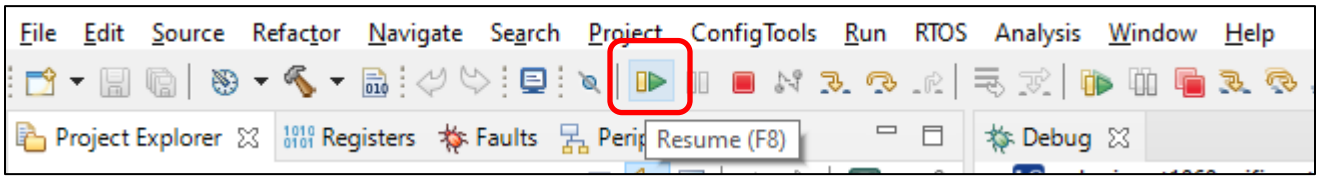
5. After the Debug process is complete, open the “Device Manager” in windows and go to “Ports (COM & LPT)” to check the mbed COM port number. In this case, COM 22 is our COM port number.



6. Now the example is ready to run. Open Tera Term on the appropriate COM port (i.e. COM 22 in this case). Configure port for 115200 bps, 8 bits data, no parity, and 1 stop bit (115200/8/N/1).



7. Click resume button in MCUXpresso.



8. You should see this output from i.MX RT. The example will automatically perform a scan, connect to the AP specified (in step 2) and ping the host address specified (in step 2).

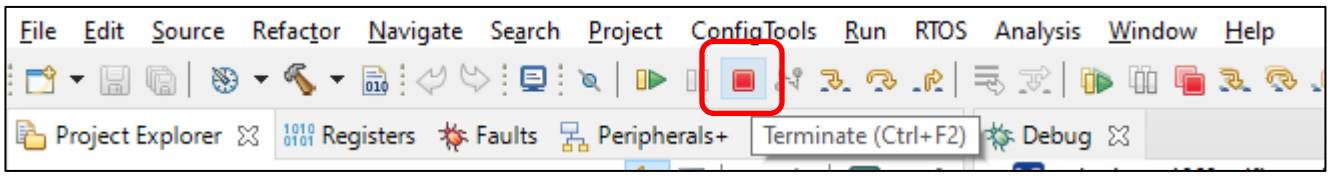
```
Wifi setup example
Initialize WLAN Driver
Setting up new cal data
MAC Address: 2C:4C:C6:F4:D4:40
[net] Initialized TCP/IP networking stack
Scan scheduled...
app_cb: WLAN: received event 10
app_cb: WLAN initialized
WLAN Driver Version    : v1.3.r33.p2
WLAN Firmware Version : IW416-V0, RF878X, FP91, 16.91.10.p214, WPA2_CVE_FIX 1, PVE_FIX 1
5 networks found:
BC:62:D2:6A:A5:50  "Soumya -2.4.ghz"
        channel: 4
        rssi: -92 dBm
        security: WPA/WPA2 Mixed
        WMM: YES
30:49:50:29:5A:51  "RNMKR_2"
        channel: 11
        rssi: -55 dBm
        security: WPA2
        WMM: YES
30:49:50:29:5A:52  "RNMKR_5"
        channel: 44
        rssi: -37 dBm
        security: WPA2
        WMM: YES
32:49:50:19:5A:52  (hidden)
        channel: 44
        rssi: -37 dBm
        security: WPA2
        WMM: YES
60:38:E0:9A:A3:9C  "Murata_5"
        channel: 161
        rssi: -34 dBm
        security: OPEN
        WMM: YES
Connecting to Murata_5 ....app_cb: WLAN: received event 0
Connected to [Murata_5]
ping: send
192.168.1.147


ping: recv
192.168.1.147
 3 ms

ping: send
192.168.1.147
```

Scanning

Connecting to AP

Pinging host

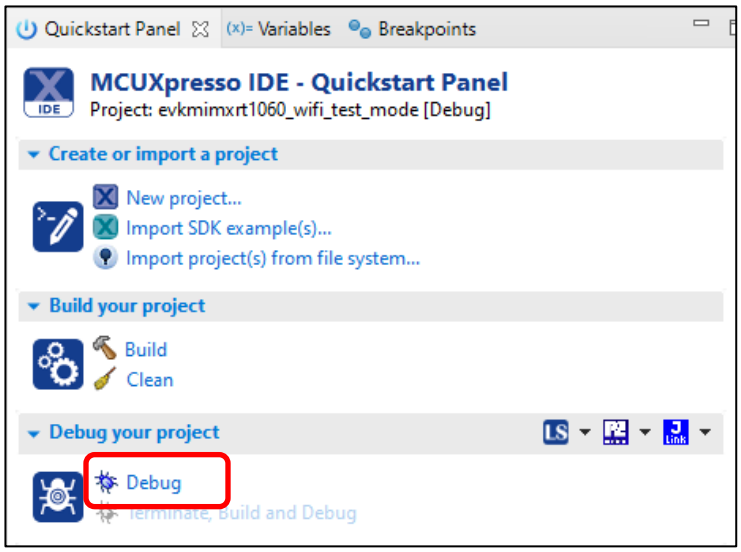
9. Click Terminate button in MCUXpresso to stop the test.



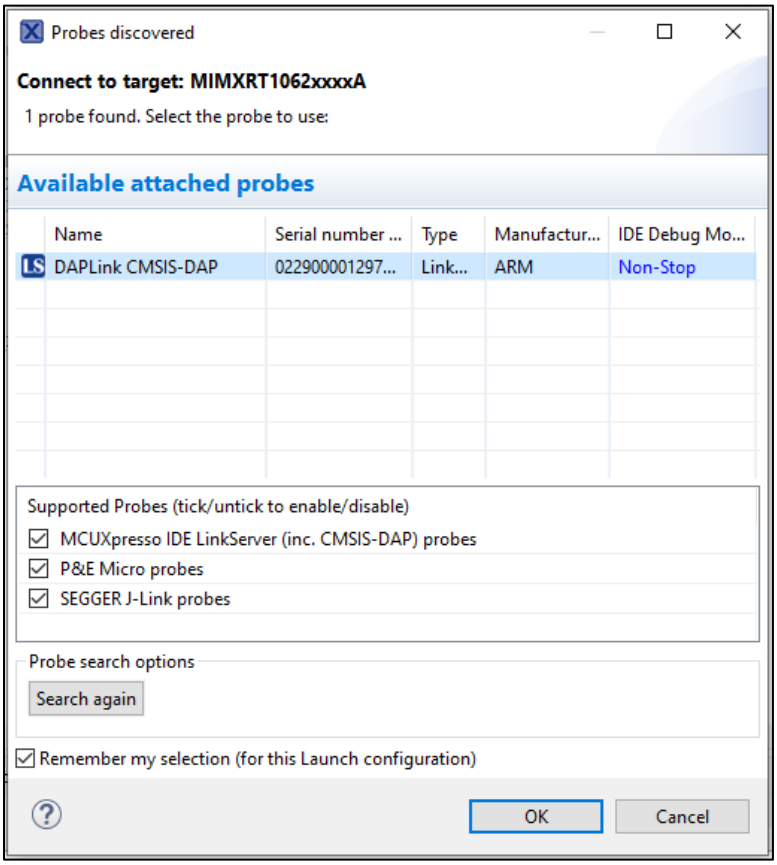
8.6 Example evkmimxrt1060_wifi_test_mode

1. Refer to **Section 7** to import and patch the evkmimxrt1060_wifi_test_mode example, and select the current module (2DS, 1XK or 1ZM).

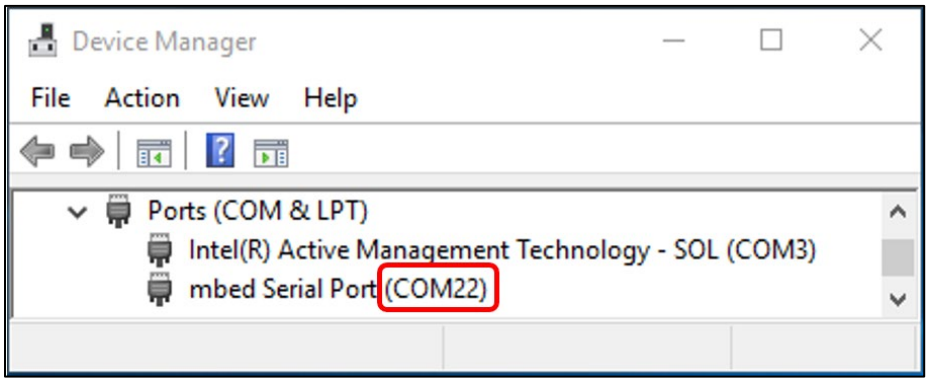
2. Click Debug in the QuickStart Panel.



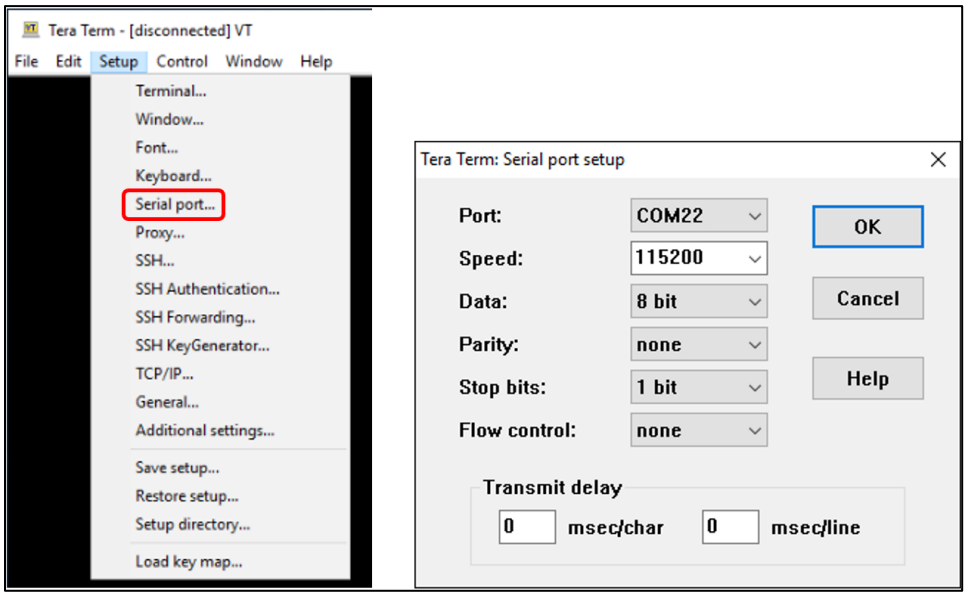
3. For the very first attempt, you need to select the appropriate JTAG adapter. Select the adapter and click OK, and then wait for a while for the build. You can see the log on console window.



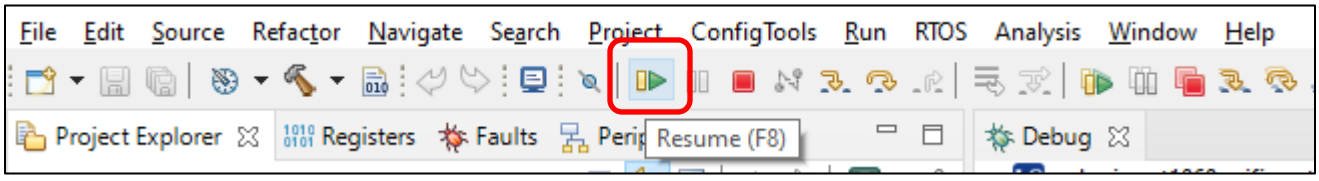
4. After the Debug process is complete, open the “Device Manager” in windows and go to “Ports (COM & LPT)” to check the mbed COM port number. In this case, COM 22 is our COM port number.



5. Now the example is ready to run. Open Tera Term on the appropriate COM port (i.e. COM 22 in this case). Configure port for 115200 bps, 8 bits data, no parity, and 1 stop bit (115200/8/N/1).



6. Click resume button in MCUXpresso.



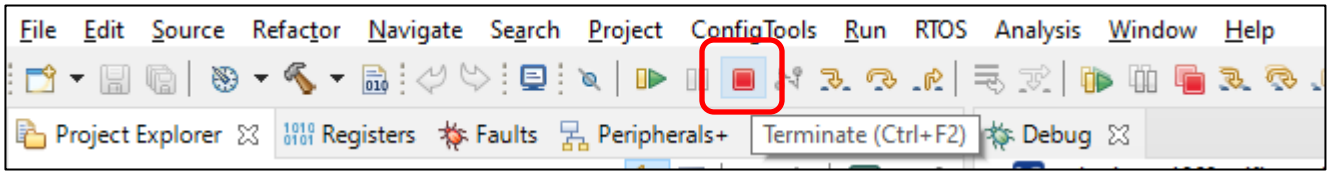
7. You should see this output from i.MX RT.

```
=======================================
wifi test mode demo
=======================================
Initialize CLI
=======================================
Initialize WLAN Driver
=======================================
Setting up new cal data
MAC Address: 2C:4C:C6:F4:D4:40
[net] Initialized TCP/IP networking stack
=======================================
app_cb: WLAN: received event 10
=======================================
app_cb: WLAN initialized
=======================================
WLAN Test Mode CLIs are initialized
=======================================
CLIs Available:
=======================================

help
wlan-version
wlan-mac
wlan-set-rf-test-mode
wlan-set-rf-tx-antenna <antenna>
wlan-get-rf-tx-antenna
wlan-set-rf-rx-antenna <antenna>
wlan-get-rf-rx-antenna
wlan-set-rf-band <band>
wlan-get-rf-band
wlan-set-rf-bandwidth <bandwidth>
wlan-get-rf-bandwidth
wlan-set-rf-channel <channel>
wlan-get-rf-channel
wlan-set-rf-tx-power <tx_power> <modulation> <path_id>
wlan-set-rf-tx-cont-mode <enable_tx> <cw_mode> <payload_pattern> <cs_mode> <act_sub_ch> <tx_rate>
wlan-set-rf-tx-frame <start> <data_rate> <frame_pattern> <frame_len> <adjust_burst_sifs> <burst_sifs_in_us> <
short_preamble> <act_sub_ch> <short_gi> <adv_coding> <tx_bf> <gf_mode> <stbc> <bssid>
wlan-get-and-reset-rf-per
=======================================
```

8. Type any of the command(s) and press enter to execute.

```
=======================================
wlan-version
WLAN Driver Version   : v1.3.r33.p2
WLAN Firmware Version : IW416-V0, RF878X, FP91, 16.91.10.p214, WPA2_CVE_FIX 1, PVE_FIX 1

# wlan-set-rf-test-mode
RF Test Mode configuration successful

# wlan-set-rf-rx-antenna 1
Rx Antenna configuration successful

# wlan-get-rf-rx-antenna
Configured Rx Antenna is: Main
```

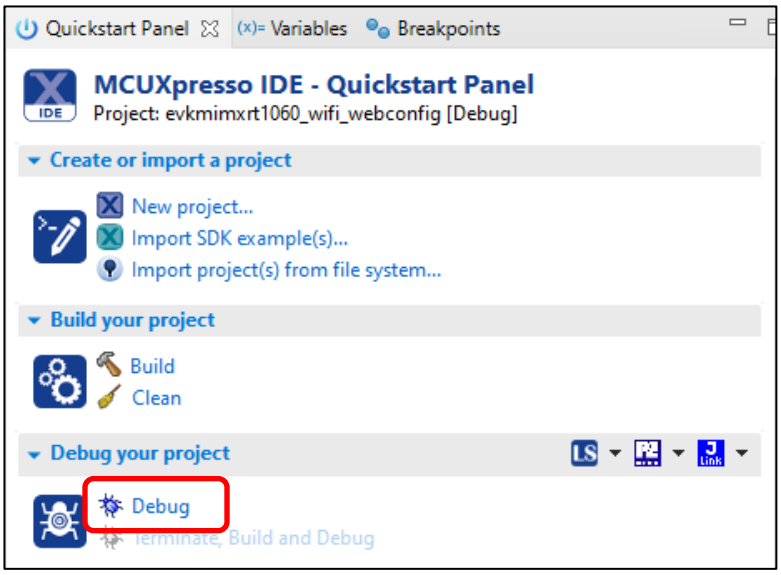
9. Click Terminate button in MCUXpresso to stop the test.



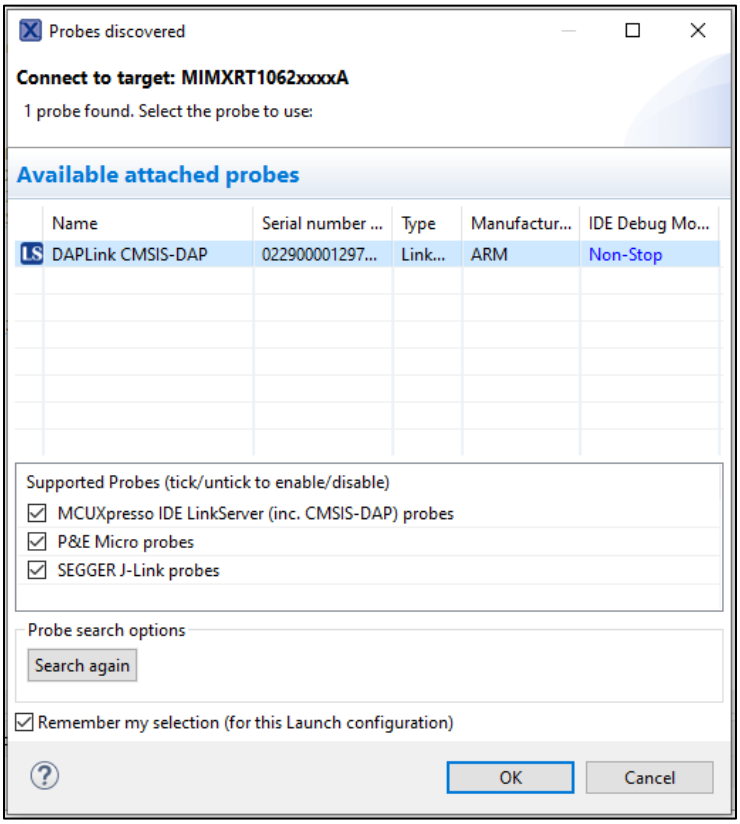
8.7 Example evkmimxrt1060_wifi_webconfig

1. Refer to **Section 7** to import and patch the evkmimxrt1060_wifi_webconfig example, and select the current module (2DS, 1XK or 1ZM).

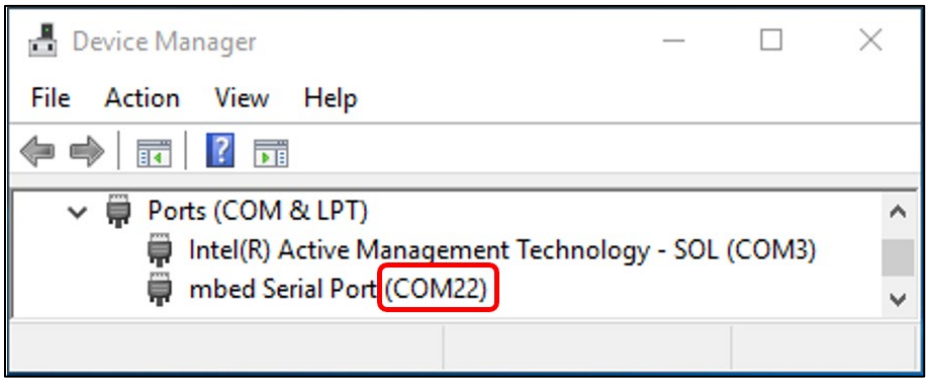
2. Click Debug in the QuickStart Panel.



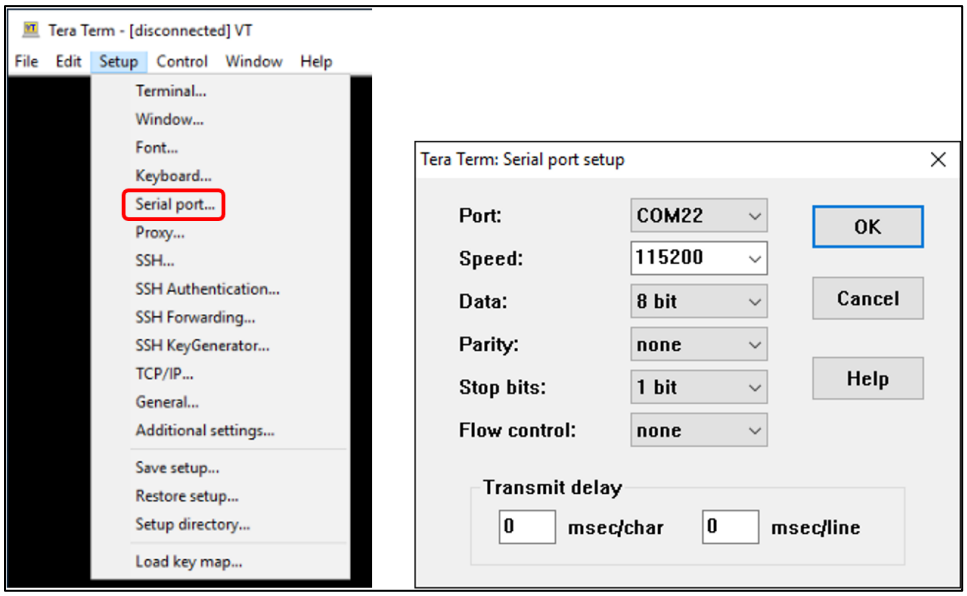
3. For the very first attempt, you need to select the appropriate JTAG adapter. Select the adapter and click OK, and then wait for a while for the build. You can see the log on console window.



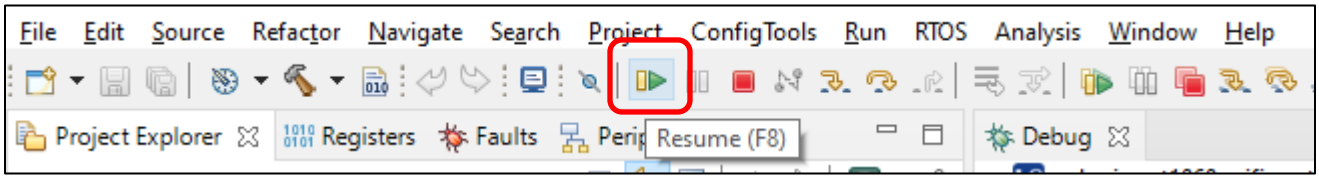
4. After the Debug process is complete, open the “Device Manager” in windows and go to “Ports (COM & LPT)” to check the mbed COM port number. In this case, COM 22 is our COM port number.



5. Now the example is ready to run. Open Tera Term on the appropriate COM port (i.e. COM 22 in this case). Configure port for 115200 bps, 8 bits data, no parity, and 1 stop bit (115200/8/N/1).



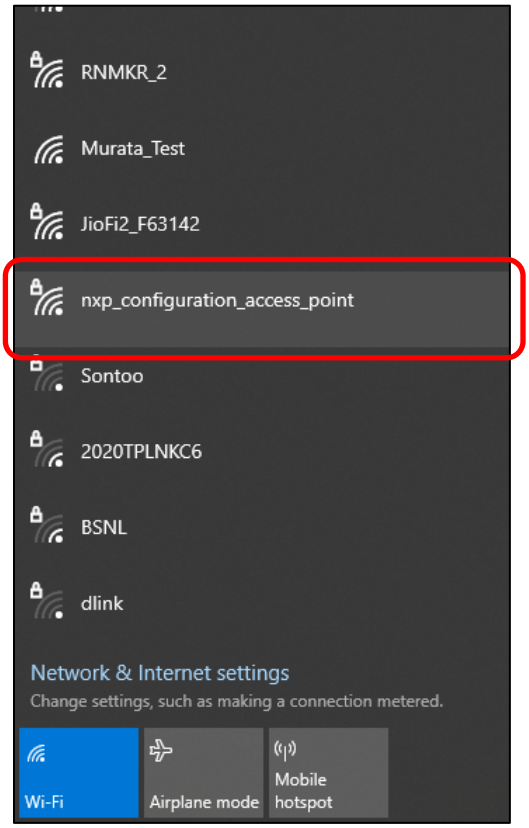
6. Click resume button in MCUXpresso.



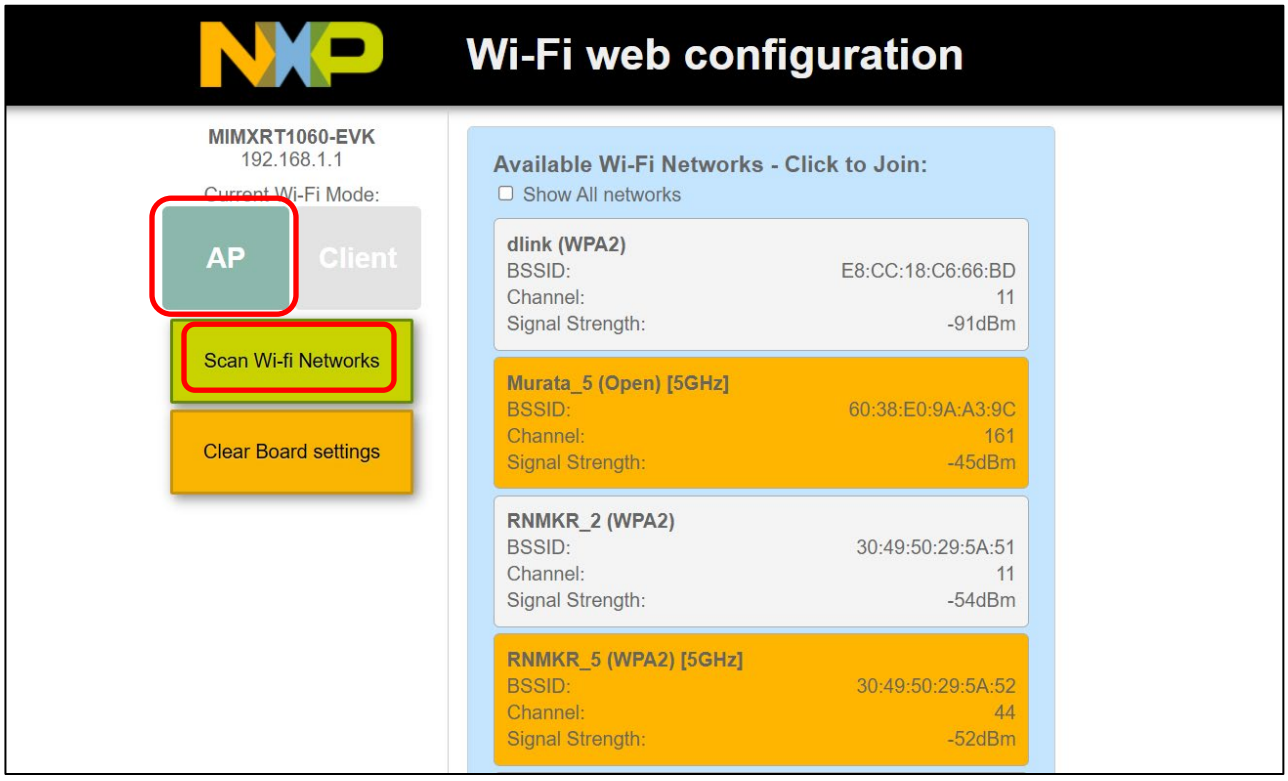
7. You should see this output from i.MX RT on Tera Term.

```
Starting webconfig DEMO
[i] Trying to load data from mflash.
[i] Nothing stored yet
[i] Initializing WiFi connection...
Setting up new cal data
MAC Address: 2C:4C:C6:F4:D4:40
[net] Initialized TCP/IP networking stack
WLAN initialized
WLAN FW Version: IW416-V0, RF878X, FP91, 16.91.10.p214, WPA2_CVE_FIX 1, PVE_FIX 1
[i] Successfully initialized WiFi module
Starting Access Point: SSID: nxp_configuration_access_point, Chnl: 1
[wlcm] Warn: NOTE: uAP will automatically switch to the channel that station is on.
Soft AP started successfully
This also starts DHCP Server with IP 192.168.1.1
 Now join that network on your device and connect to this IP: 192.168.1.1
```

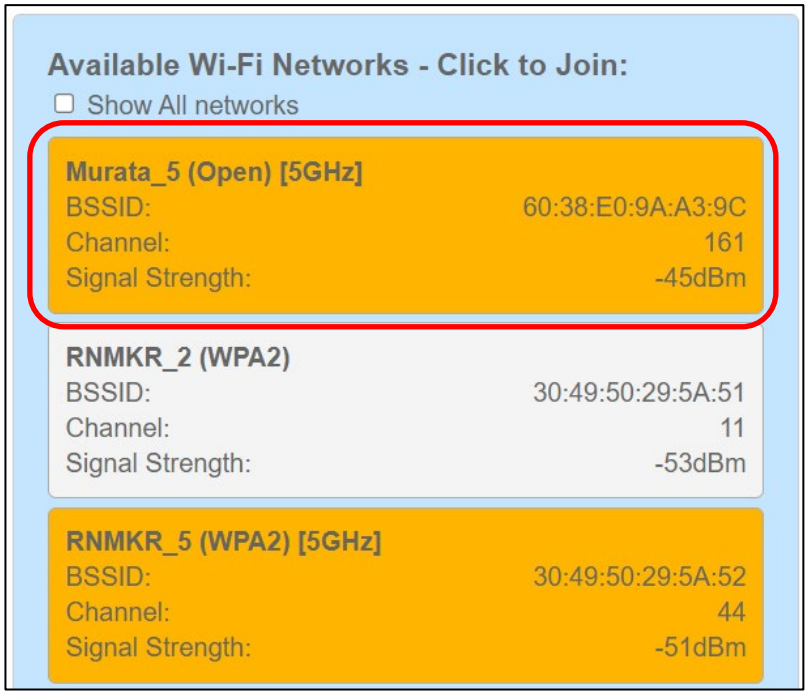
8. From a wireless client device (can be a laptop, or a phone), search for available wireless networks and connect to “nxp_configuration_access_point” SSID. The password is “NXP0123456789”.



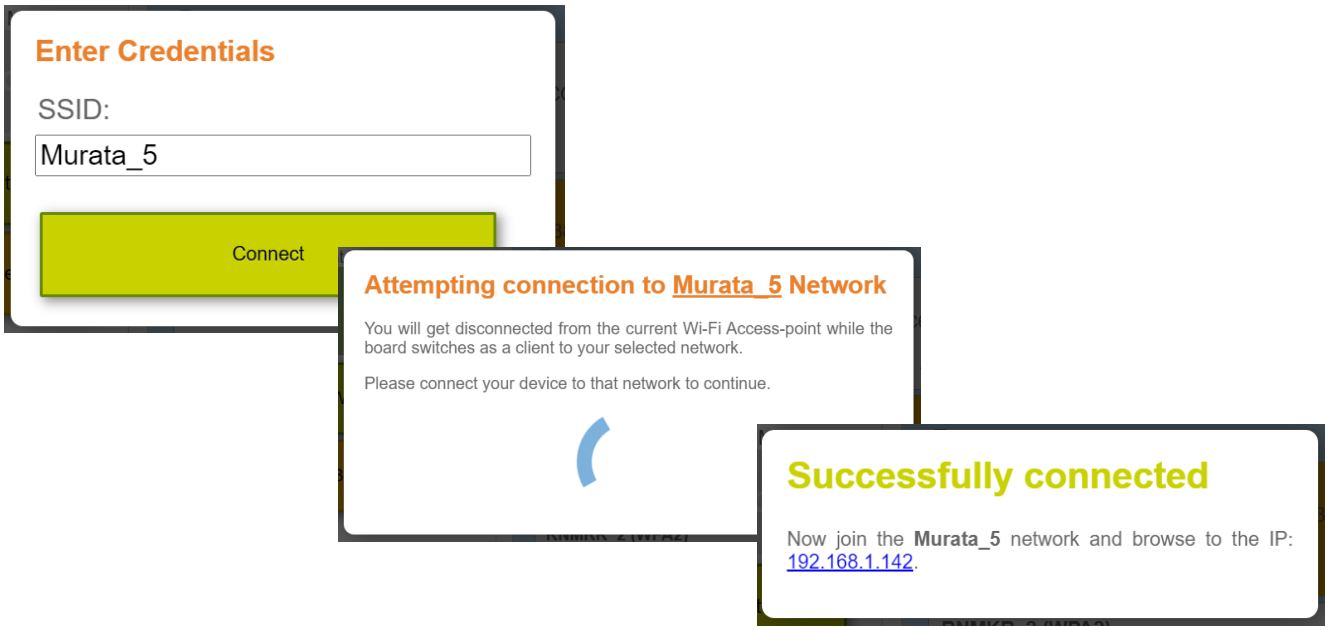
9. Open the web browser on the client device (Microsoft Internet Explorer is not supported) and go to “192.168.1.1” IP address. The wifi_webconfig example creates a web-based configuration interface to set up the Wi-Fi client configurations here. The EVK is currently set up as an AP, as can be seen in the UI. You can scan for available networks here.



10. Click on a network of your selection to connect to it.



11. Enter the network password (if required) and click on connect.



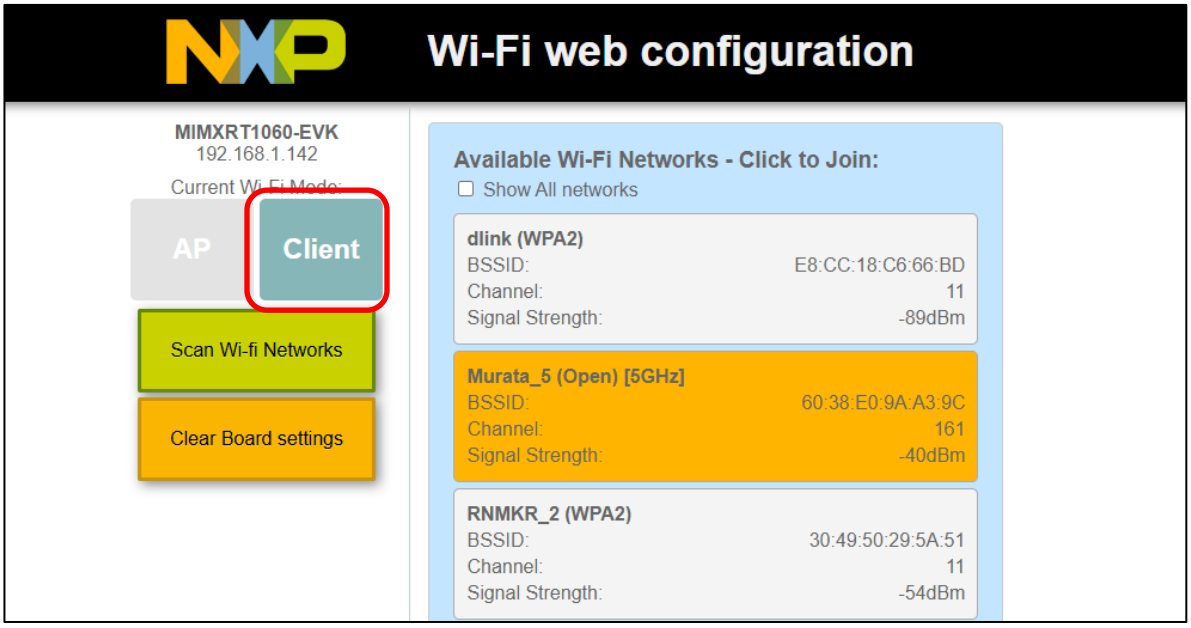
12. If connection is successful, the credential will be saved on the EVK mflash and will be used automatically after the EVK reboots. The AP will be turned off. You should see this output on Tera Term. Note the IP address shown.

```
Client => 28:3A:4D:36:5B:6D Associated with Soft AP

Initiating scan...

                RNMKR_2
     BSSID         : 30:49:50:29:5A:51
     RSSI          : -53dBm
     Channel       : 11

     BSSID         : 32:49:50:19:5A:51
     RSSI          : -54dBm
     Channel       : 11
RNMKR_5
     BSSID         : 30:49:50:29:5A:52
     RSSI          : -51dBm
     Channel       : 44

     BSSID         : 32:49:50:19:5A:52
     RSSI          : -51dBm
     Channel       : 44
Murata_5
     BSSID         : 60:38:E0:9A:A3:9C
     RSSI          : -45dBm
     Channel       : 161
[i] Chosen ssid: Murata_5
[i] Chosen passphrase: ""
[i] Joining: Murata_5
Connected to following BSS:SSID = [Murata_5], IP = [192.168.1.142]
[i] Successfully joined: Murata_5
 Now join that network on your device and connect to this IP: 192.168.1.142
[i] mflash_save_file success
Client => 28:3A:4D:36:5B:6D Associated with Soft AP
[i] Stopping AP!
Soft AP stopped successfully
```

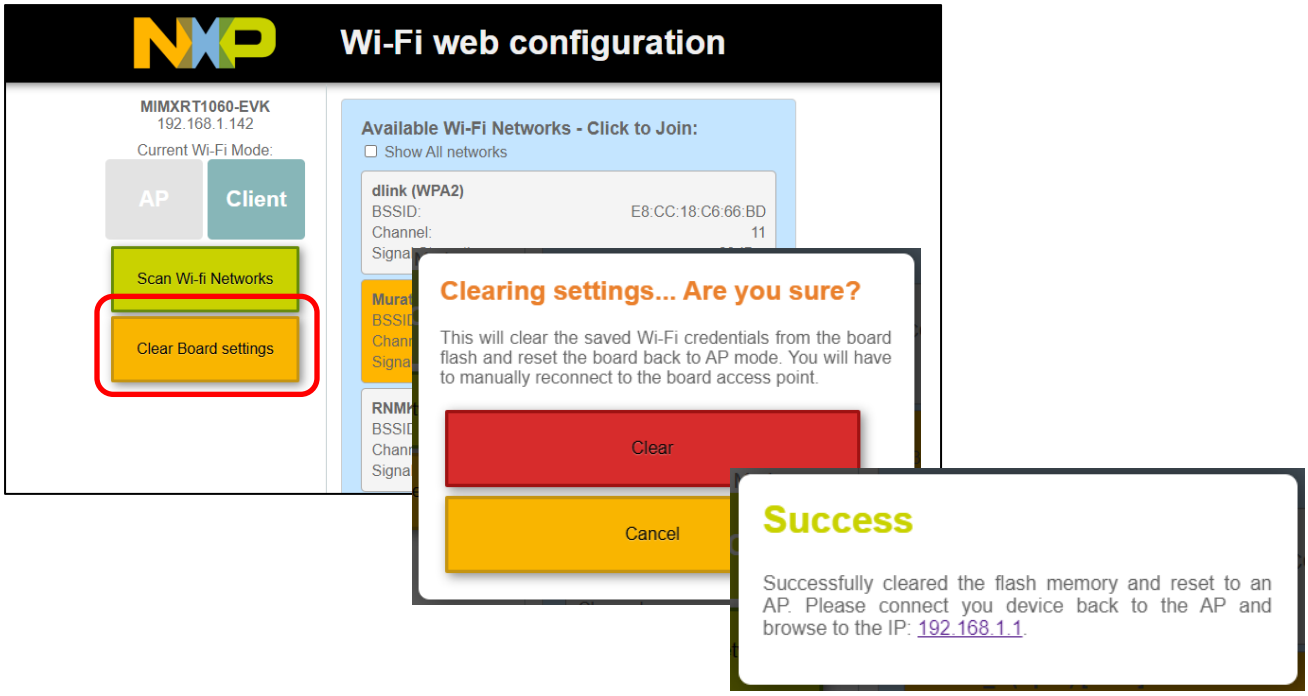
Upon reboot, the EVK will henceforth automatically connect to the saved network.

13. Connect the wireless client device to the same network as the EVK, open the web browser and go to the IP address shown on the Tera Term window (192.168.1.142 in this example).

14. The web-based configuration interface will be accessible here. The EVK is currently set up as a Wi-Fi client, as can be seen in the UI.



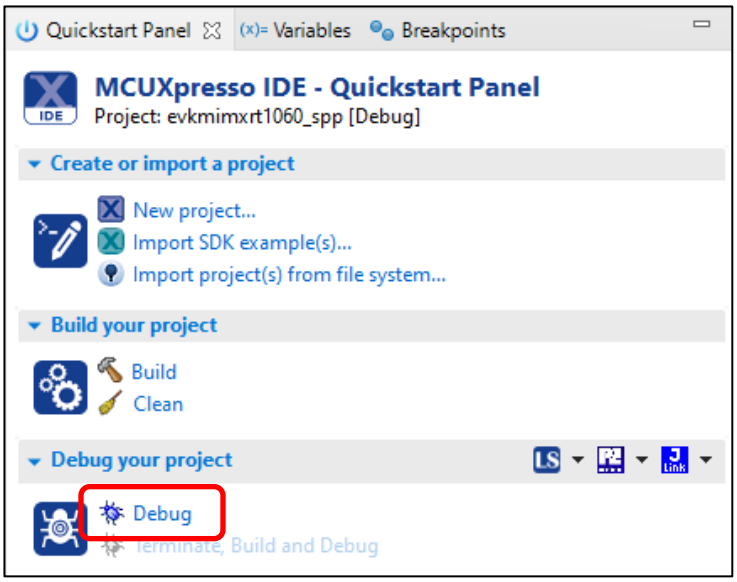
15. You can use the “Clear Board settings” button on the interface to remove the saved network settings.



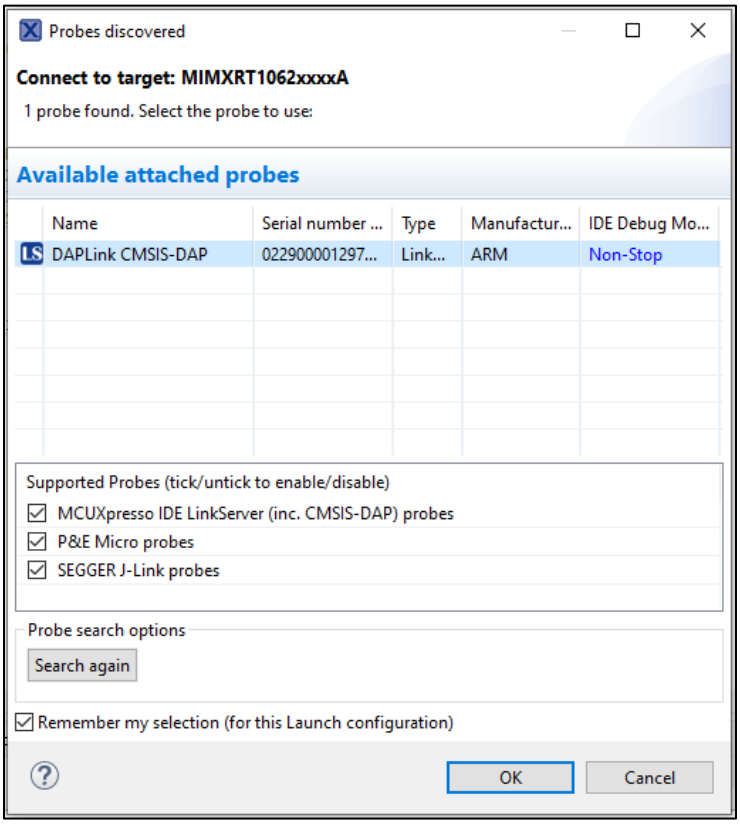
8.8 Example evkmimxrt1060_spp

1. Refer to **Section 7** to import and patch the evkmimxrt1060_spp example, and select the current module (2DS, 1XK or 1ZM).

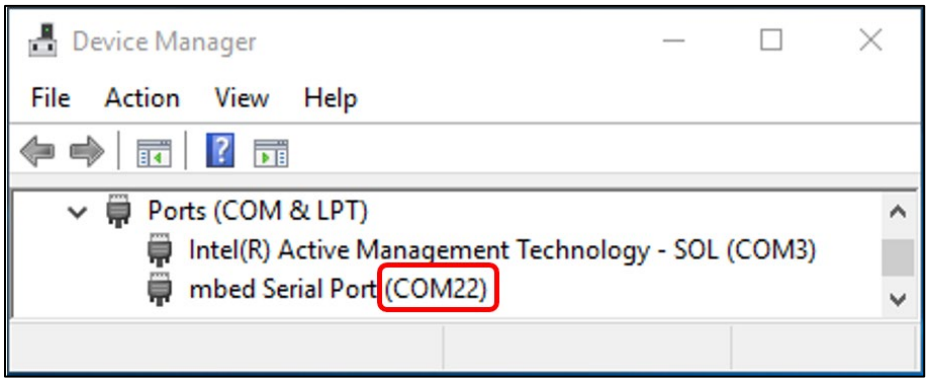
2. Click Debug in the QuickStart Panel.



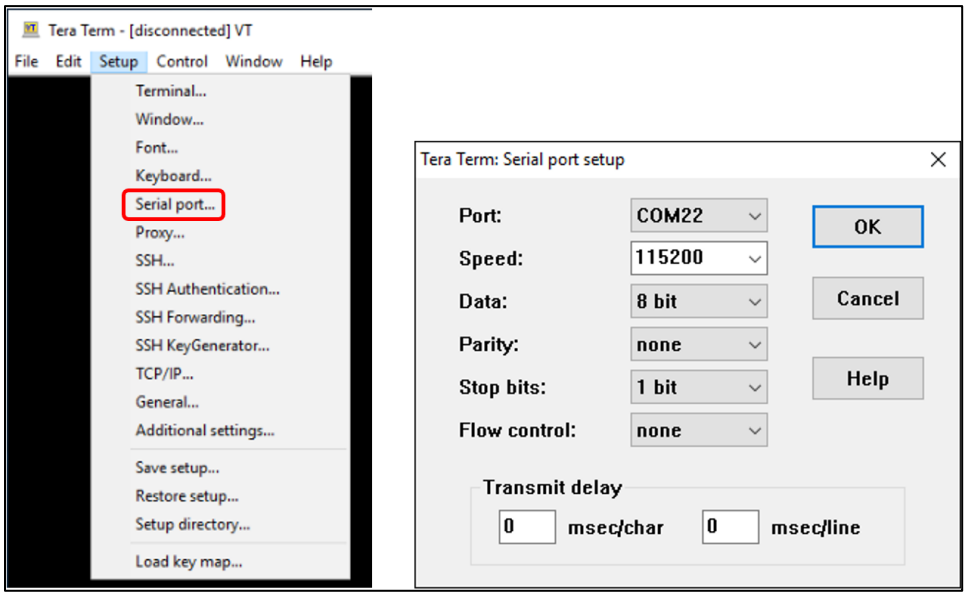
3. For the very first attempt, you need to select the appropriate JTAG adapter. Select the adapter and click OK, and then wait for a while for the build. You can see the log on console window.



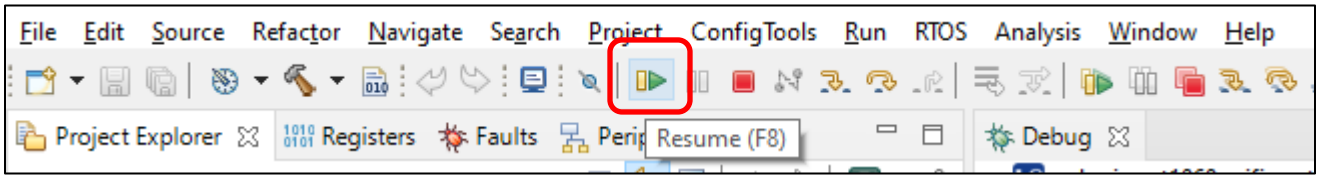
4. After the Debug process is complete, open the “Device Manager” in windows and go to “Ports (COM & LPT)” to check the mbed COM port number. In this case, COM 22 is our COM port number.



5. Now the example is ready to run. Open Tera Term on the appropriate COM port (i.e. COM 22 in this case). Configure port for 115200 bps, 8 bits data, no parity, and 1 stop bit (115200/8/N/1).



6. Click resume button in MCUXpresso.

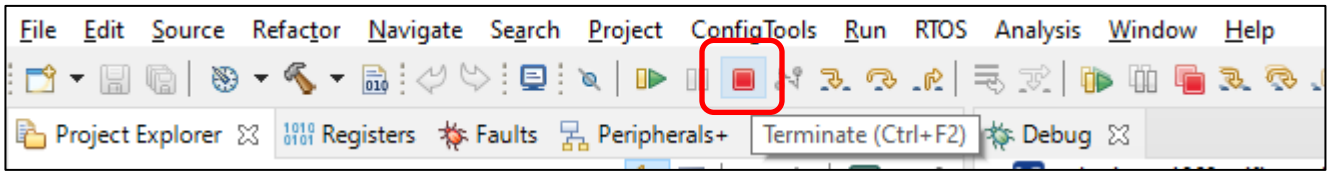


7. You should see this output from i.MX RT on Tera Term.

```
Bluetooth initialized
                    BR/EDR set connectable and discoverable done

Copyright  2020  NXP

>>
>> help

"help": List all the registered commands

"exit": Exit program

"bt": BT related function
  USAGE: bt [discover|connect|disconnect|delete]
    bt discover    start to find BT devices
    bt connect     connect to the device that is found, for example: bt connectdevice n (from 1)
    bt disconnect  disconnect current connection.
    bt delete      delete all devices. Ensure to disconnect the HCI link connection with the peer device before attempting to delete the bonding information.

"spp": SPP related function
  USAGE:
    spp register [5|3]    register a spp server channel
    spp discover          discover spp server channel on peer device
    spp connect [channel] create spp connection
    spp disconnect        disconnect current spp connection.
    spp send [1|2|3|4]    send data over spp connection.
>>
```

8. Given below are the list of supported commands.
    - bt discover: Start scan to find BT devices
    - bt connect: Connect to one of the found devices. For example: bt connectdevice n (indexing starts from 1)
    - bt disconnect: Disconnect current connection
    - bt delete: Delete all devices. Ensure to disconnect the HCI link connection with the peer device before attempting to delete the bonding information
    - spp register [5|3]: Register an SPP server channel
    - spp discover: Discover SPP server channel on peer device
    - spp connect [channel]: Create SPP connection
    - spp disconnect: Disconnect current SPP connection
    - spp send [1|2|3|4]: Send data over SPP connection

9. Click Terminate button in MCUXpresso to stop the test.

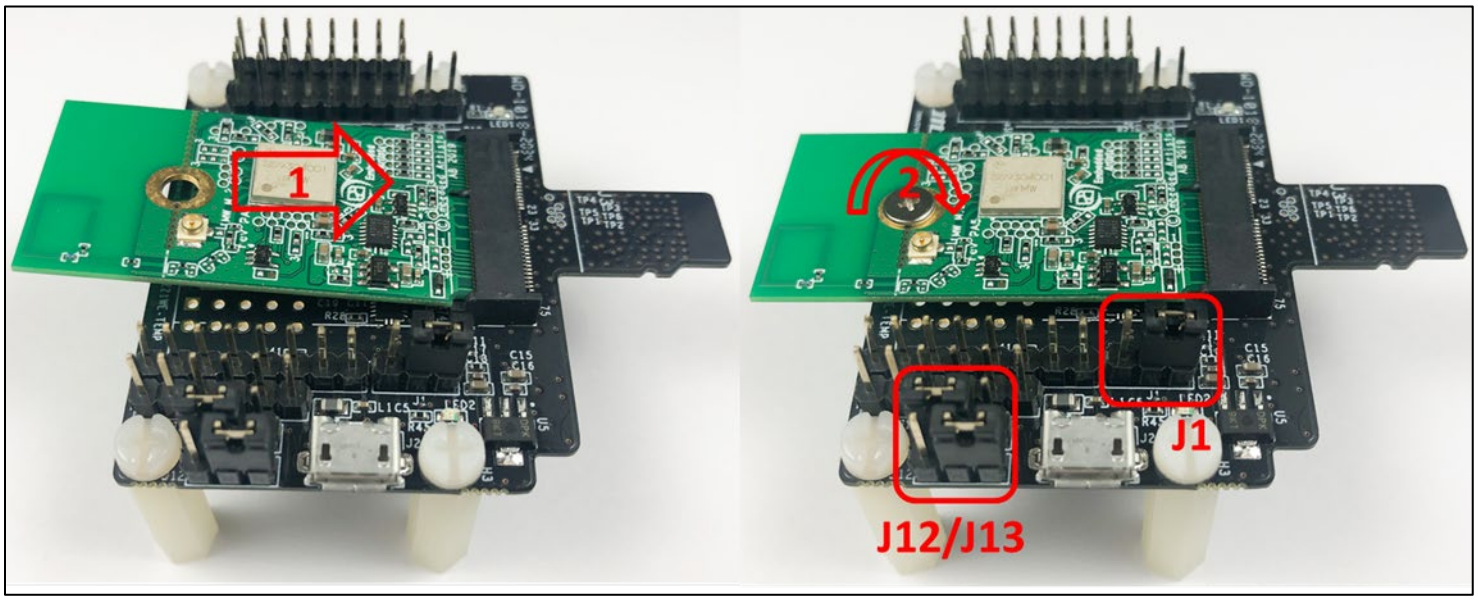


9 Murata's uSD-M.2 Adapter

9.1 Connecting the Wi-Fi/BT M.2 EVB to uSD-M.2 Adapter

When connecting the Wi-Fi/BT M.2 EVB to uSD-M.2 Adapter Rev B1 (**Figure 8**), make sure to (#1) firmly insert it before using M.2 screw to (#2) secure it in place. Important Jumpers (J12, J13, and J1) are highlighted.

Figure 8: Connecting the Wi-Fi/BT M.2 EVB to uSD-M.2 Adapter

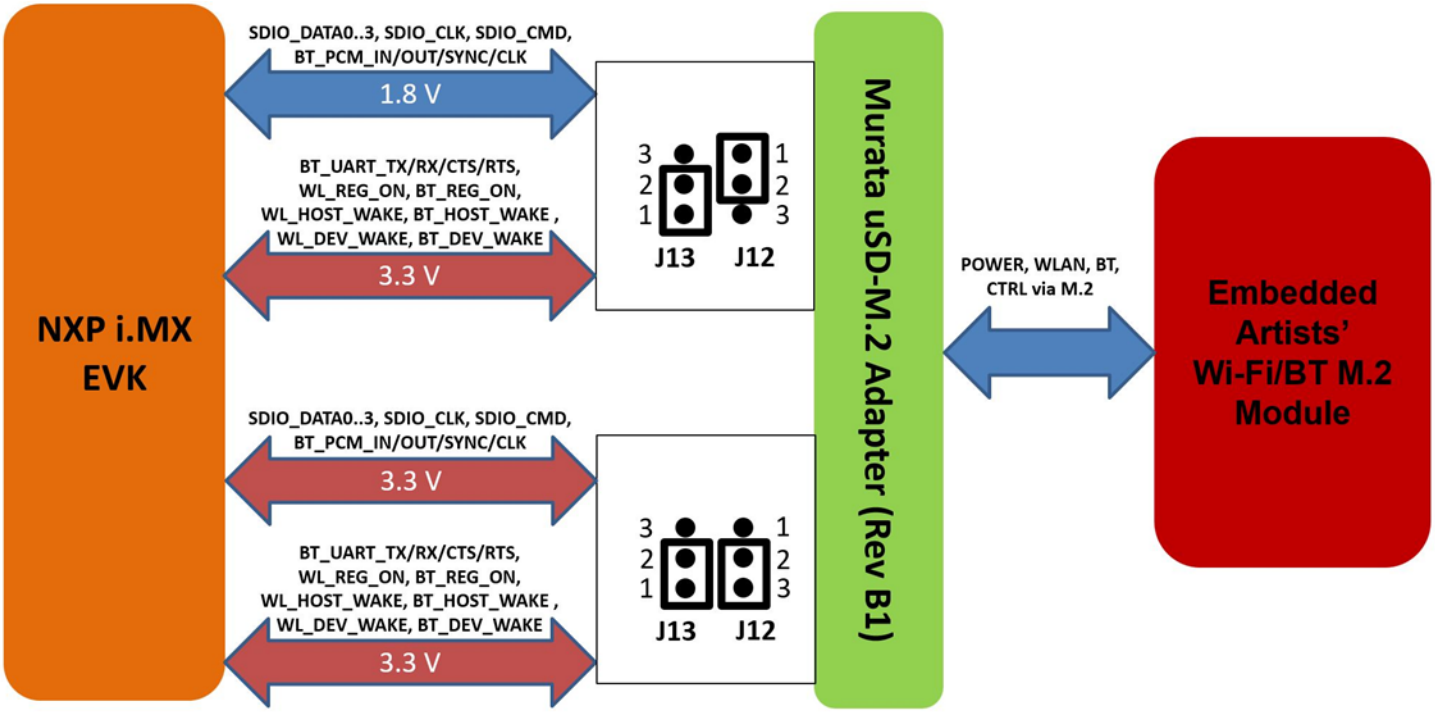


9.2 Configuring uSD-M.2 Adapter Jumpers for Correct VIO Signaling

Figure 9 shows a block diagram highlighting the Host (i.MX EVK) and Wi-Fi/BT M.2 EVB VIO signaling voltages. The legacy i.MX 6 EVK's (excluding the i.MX 6UL(L) EVK's) have J13/J12 set to 1-2/2-3 positions respectively for the 3.3V VIO override mode setting (WLAN-SDIO, BT-UART, and WLAN/BT control signals all at 3.3V VIO).

All other i.MX EVK's (i.MX 8M Mini/Nano & i.MX 6UL(L) EVK's) have J13/J12 set to 1-2/1-2 positions respectively for the 1.8V VIO default configuration (WLAN-SDIO VIO at 1.8V VIO; BT-UART and WLAN/BT control signals at 3.3V VIO).

Figure 9: Host/M.2 IO Voltage Level Shift Options on Rev B1 Adapter

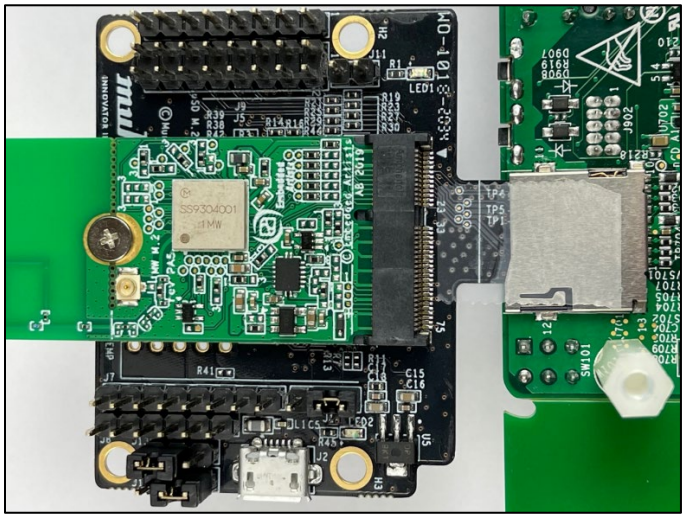


9.3 Securing uSD-M.2 Adapter to NXP i.MX EVK

On NXP i.MX RT EVK's, a common issue that customers run into is an unreliable uSD electrical connection when using Murata's uSD-M.2 Adapter. The poor interconnect is caused by the push-push (micro) SD card connectors on NXP i.MX EVK's.

To properly secure the uSD-M.2 Adapter interconnect on the i.MX RT EVK's, Murata ***strongly recommends*** to simply tape the uSD Adapter-EVK connection as shown in **Figure 10**. Note that taping the uSD Adapter-EVK connection makes the platform a little less flexible to work with. However, removing and re-applying clear tape is straightforward.

Figure 10: Securing uSD Connection



9.4 uSD-M.2 Adapter High-Level Description

Figure 11 and **Figure 12** show the features on the uSD-M.2 Adapter; with text explanation in **Table 6.** The uSD-M.2 Adapter supports additional signals to WLAN-SDIO using either Arduino headers (J5, J8, and J9) or 20 pin FFC connector (J6). For more details on Murata's uSD-M.2 Adapter, refer to the Adapter Datasheet or Hardware User Manual.

Table 6: uSD-M.2 Adapter Features

Char	Description
A	microSD connector provides Power (VBAT, GND) and WLAN-SDIO
B	SDIO bus test points (CLK, CMD, DAT0, DAT1, DAT2, DAT3)
C	Power LED Indicator (green): if not illuminated then no power applied to M.2 EVB
D	J11 = Optional BT Disable Jumper (not used)
E	J9 = BT UART TX/RX and WLAN/BT Control Signals (8 pin header)
F	J5 = Optional BT PCM and WLAN/BT Debug Signals (2x8 pin header)
G	Threaded mount for M.2 screw: 30mm distance from M.2 connector
H	Regulator to step down optional 5V VBAT from USB or Arduino header to 3.3V
I	External sleep clock input (32.768kHz)
J	J7 = Optional Arduino Header Power Supply (8 pin header; 5V or 3.3V VBAT)
K	J8 = BT UART RTS/CTS Signals (6 pin header)
L	J13 = Host IO Voltage: J13 in 1-2 pos for 3.3V VDDIO (default); J13 in 2-3 pos for 1.8V
M	J12 = M.2 IO Voltage: J12 in 1-2 pos for 1.8V VDDIO (default); J12 in 2-3 pos for 3.3V
N	J2 = Optional 5V USB Power Supply via Micro-AB USB Connector
O	LED2 = 3.3V M.2 IO Voltage Indicator (Blue) – not illuminated in default configuration
P	Regulator to provide optional 1.8V VIO to M.2 interface (M.2 EVB's have own 1.8V onboard)
Q	J1 = Power Supply Selector Jumper must be installed to power Adapter (unless J5 Arduino Header Pins #15/16 are connected to external GND/3.3V VBAT). **Position 1-2:** 5V/3.3V VBAT supply from micro-USB (J2); or Arduino (J7) **Position 2-3:** VBAT supply (typical 3.1~3.3V) from microSD connector
R	M.2 Connector: type 2230-xx-E
S	microSD connector pins: provides Power (VBAT, GND) and WLAN-SDIO
T	WLAN JTAG header (header pins not populated)
U	20 pin FFC connector (BT UART, BT PCM, WLAN/BT Control signals)
V	Additional test points from 20pin flat/flex connector

Figure 11: uSD-M.2 Adapter Features (Top View)

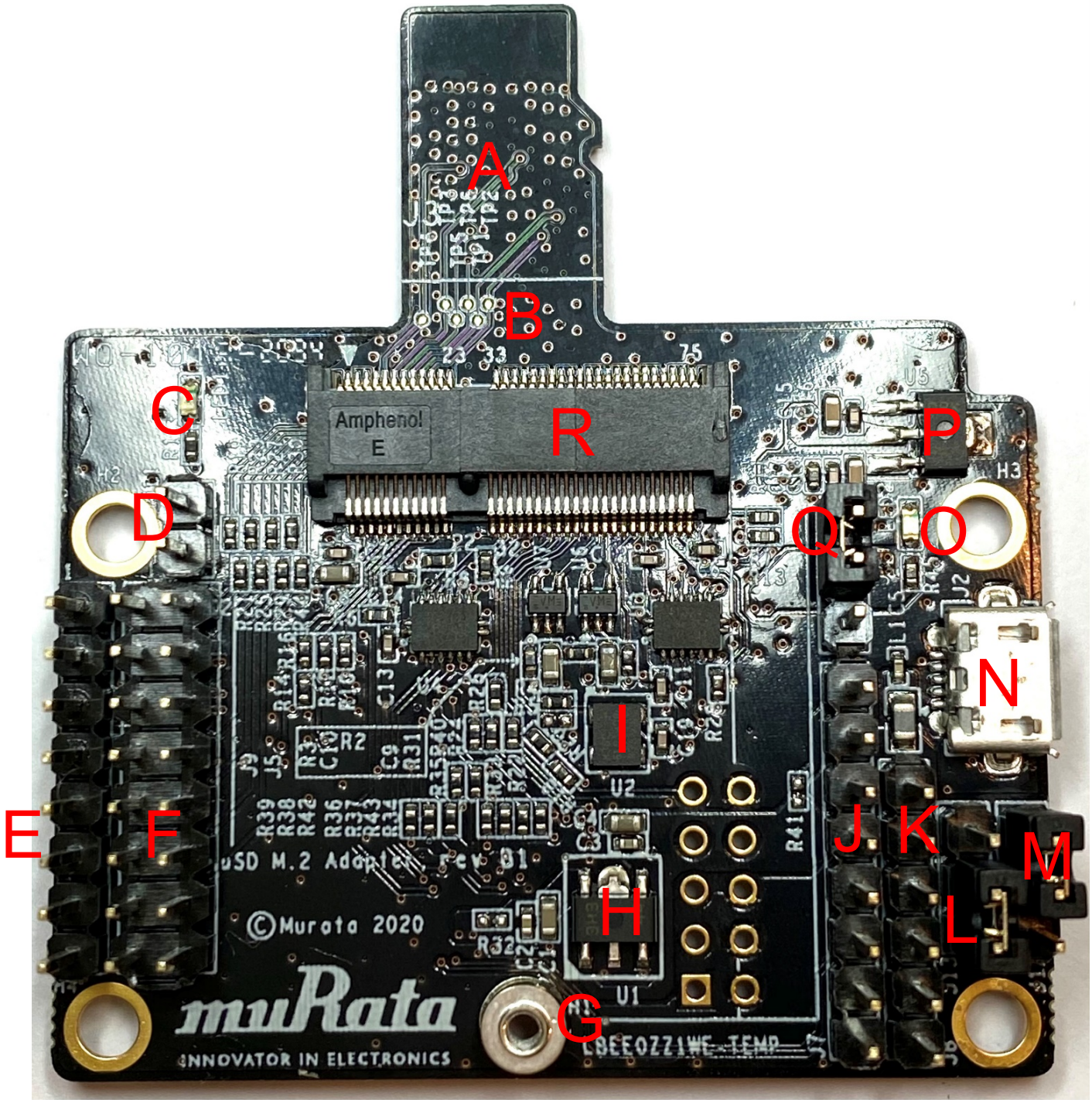
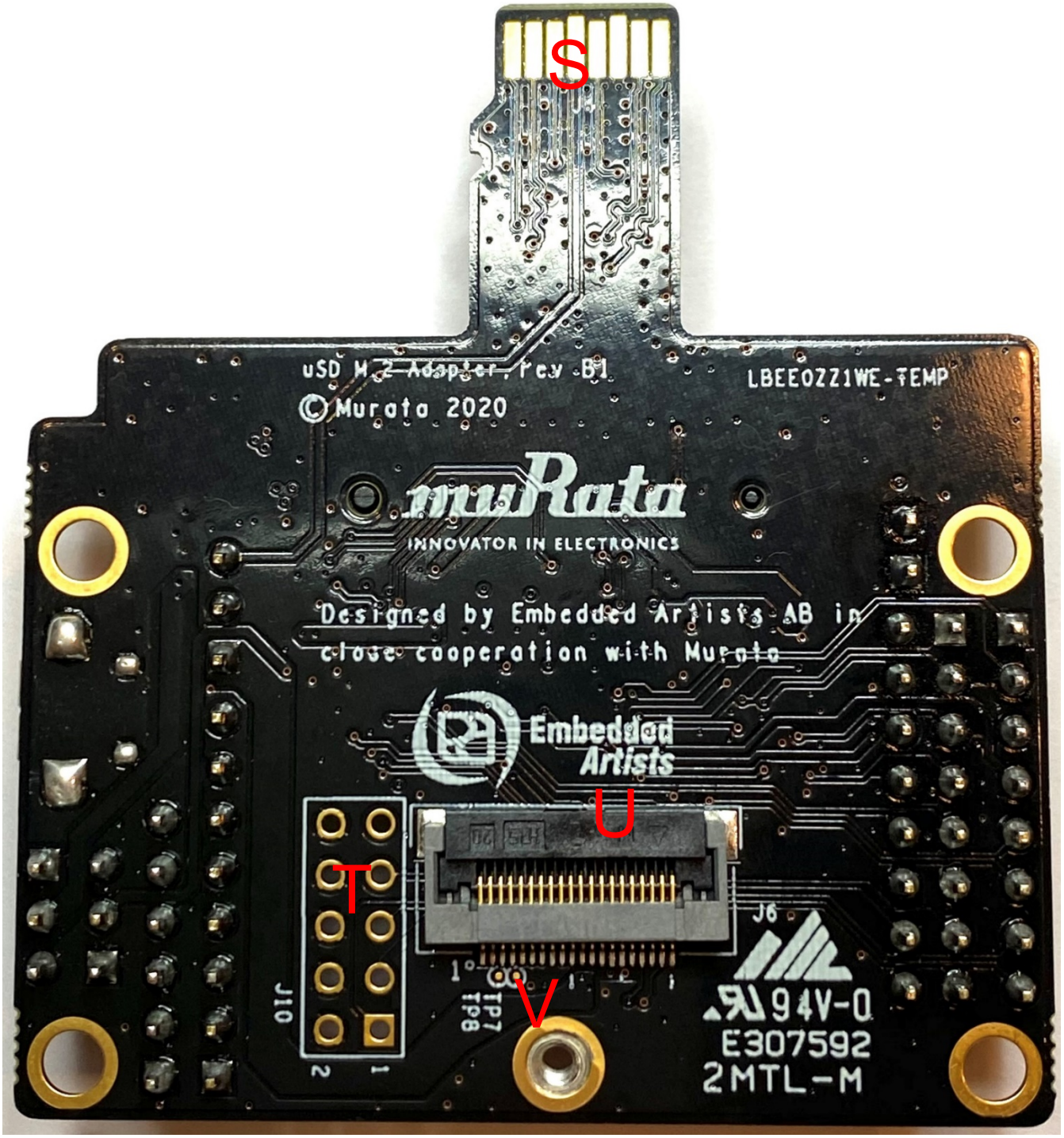


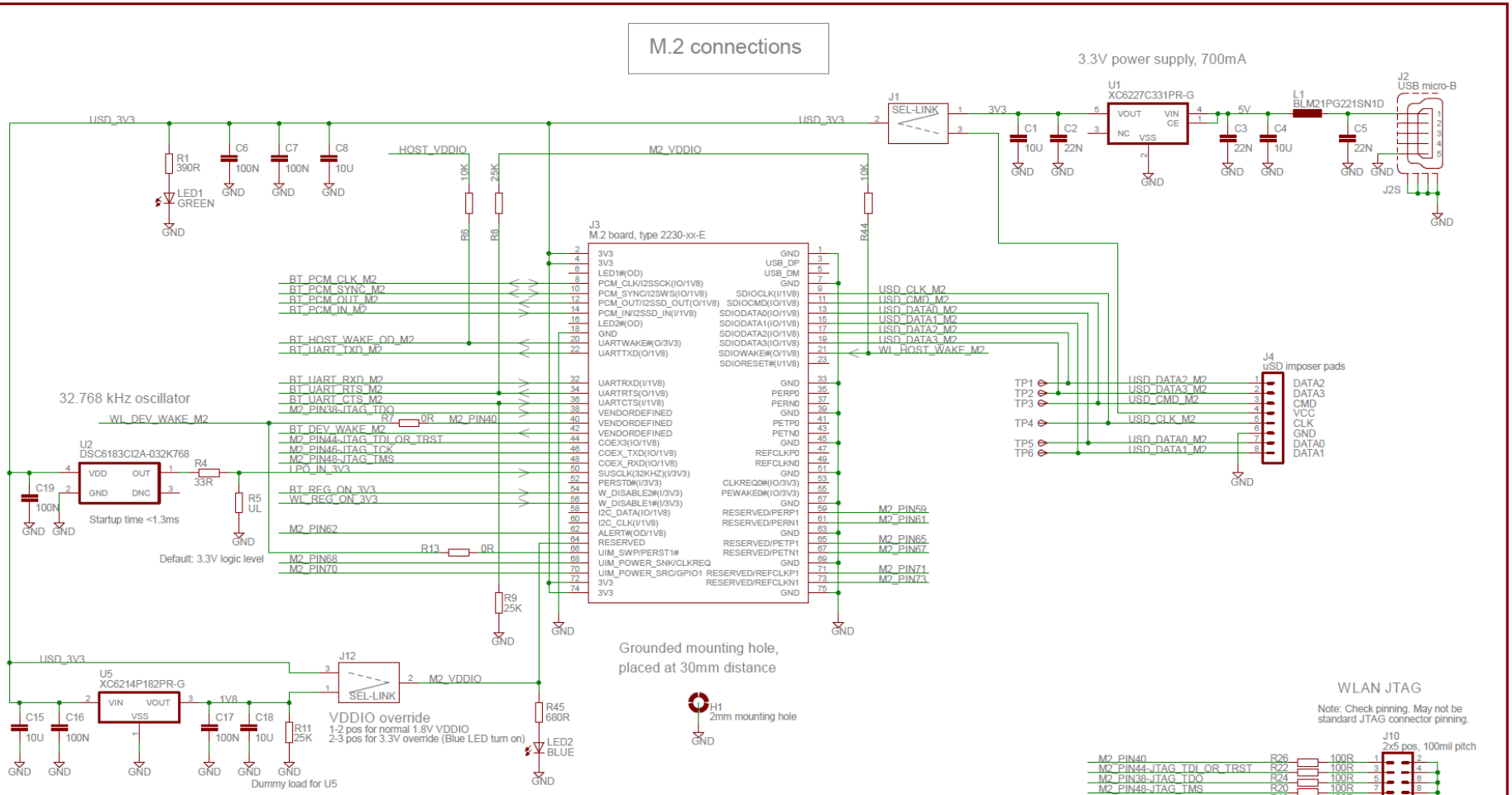
Figure 12: uSD-M.2 Adapter Features (Bottom View)



9.5 uSD-M.2 Adapter Schematic and Layout

For more specifics on adapter circuit and layout refer to **Figure 13**, **Figure 14**, and **Figure 15**.

Figure 13: uSD-M.2 schematic



IO Voltage Levels Settings

Host IO Voltage	M.2 IO Voltage	SDIO voltage	UART/Ctrl signal voltage	All other signals voltage	Notes and Explanation
3.3V (J13 in 1-2 pos)	1.8V (J12 in 1-2 pos)	1.8V	3.3V	1.8V	Voltage levels to M.2 module according to standard. 3.3V on UART and main control signals, but some direct M.2 signals have 1.8V voltage level.
3.3V (J13 in 1-2 pos)	3.3V (J12 in 2-3 pos) Blue LED is on.	3.3V	3.3V	3.3V	"3.3V override mode". 3.3V on SDIO and all GPIOs. Note that all M.2 modules do not support 3.3V override mode.
1.8V (J13 in 2-3 pos)	1.8V (J12 in 1-2 pos)	1.8V	1.8V	1.8V	Voltage levels to M.2 module according to standard. Host processor has 1.8V IO voltage.
1.8V (J13 in 2-3 pos)	3.3V (J12 in 2-3 pos) Blue LED is on.				Do not select. Not a valid combination.

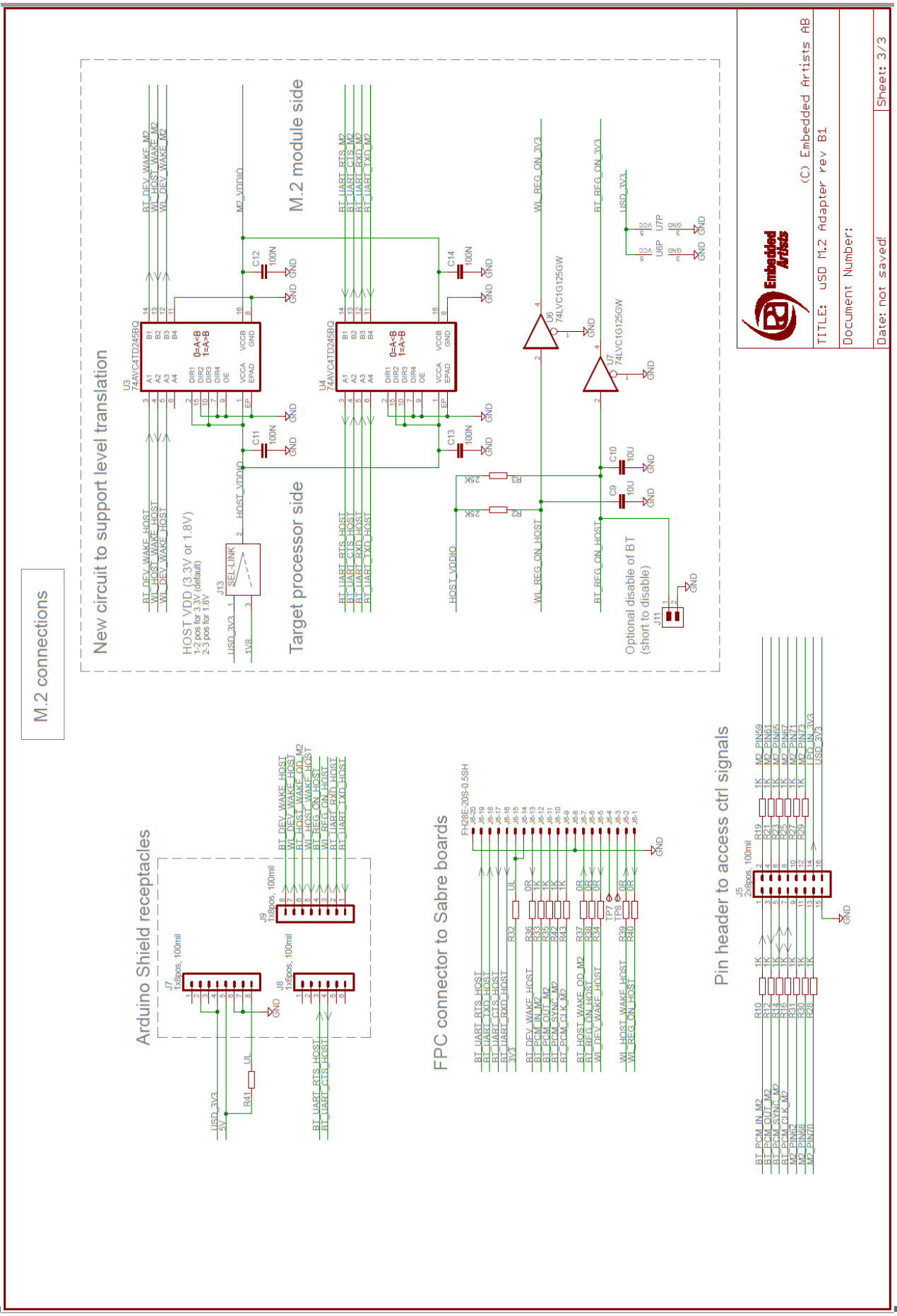

M.2 connections
New circuit to support level translation
M.2 module side
Target processor side
HOST VDD (3.3V or 1.8V)
1-2 pos for 3.3V (default)
2-3 pos for 1.8V
Optional disable of BT
(short to disable)
Arduino Shield receptacles
FPC connector to Sabre boards
Pin header to access ctrl signals
(C) Embedded Artists AB
TITLE: uSD M.2 Adapter rev B1
Document Number:
Date: not saved!
Sheet: 3/3

Figure 14: uSD-M.2 Adapter Layout (top)

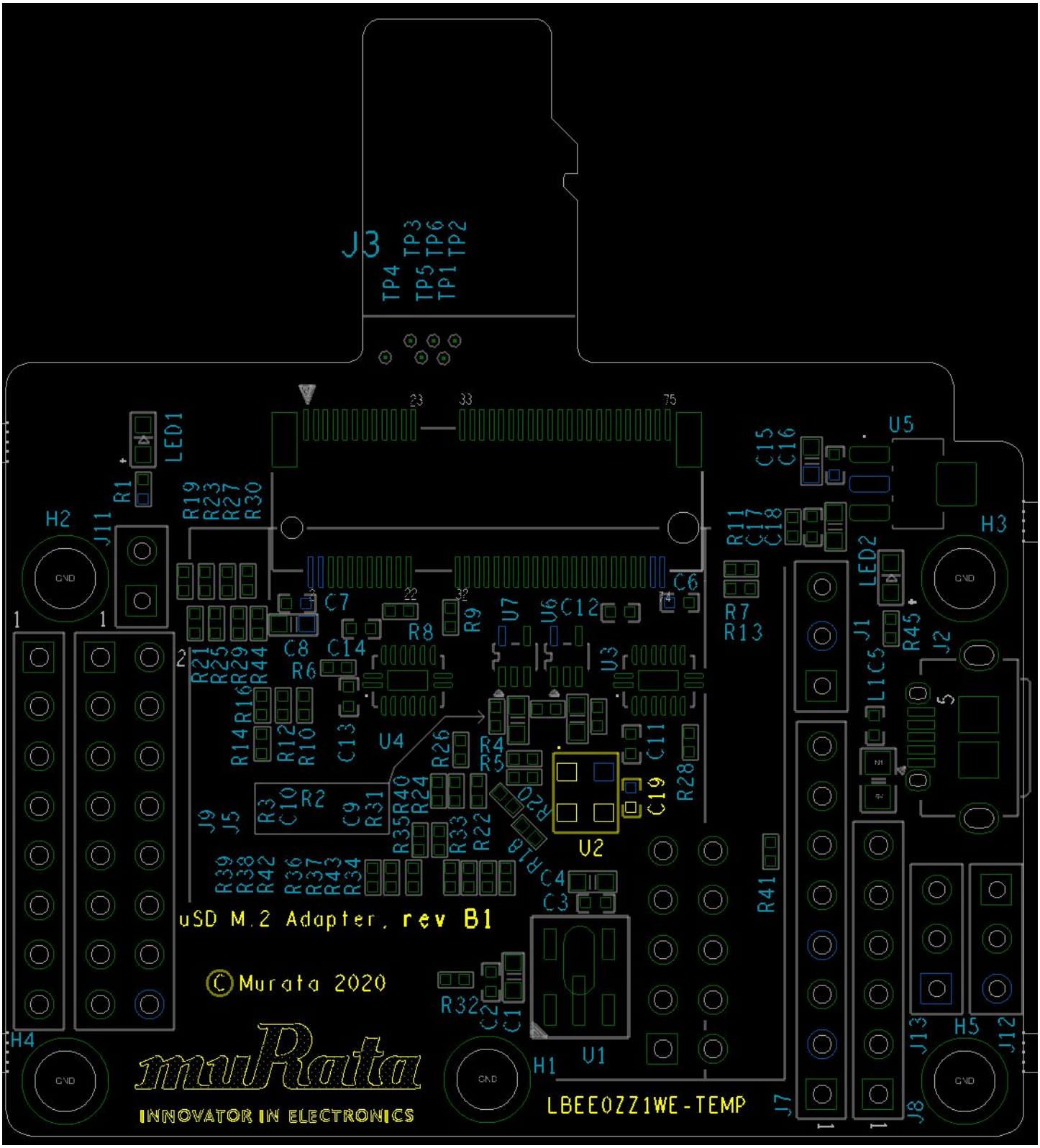
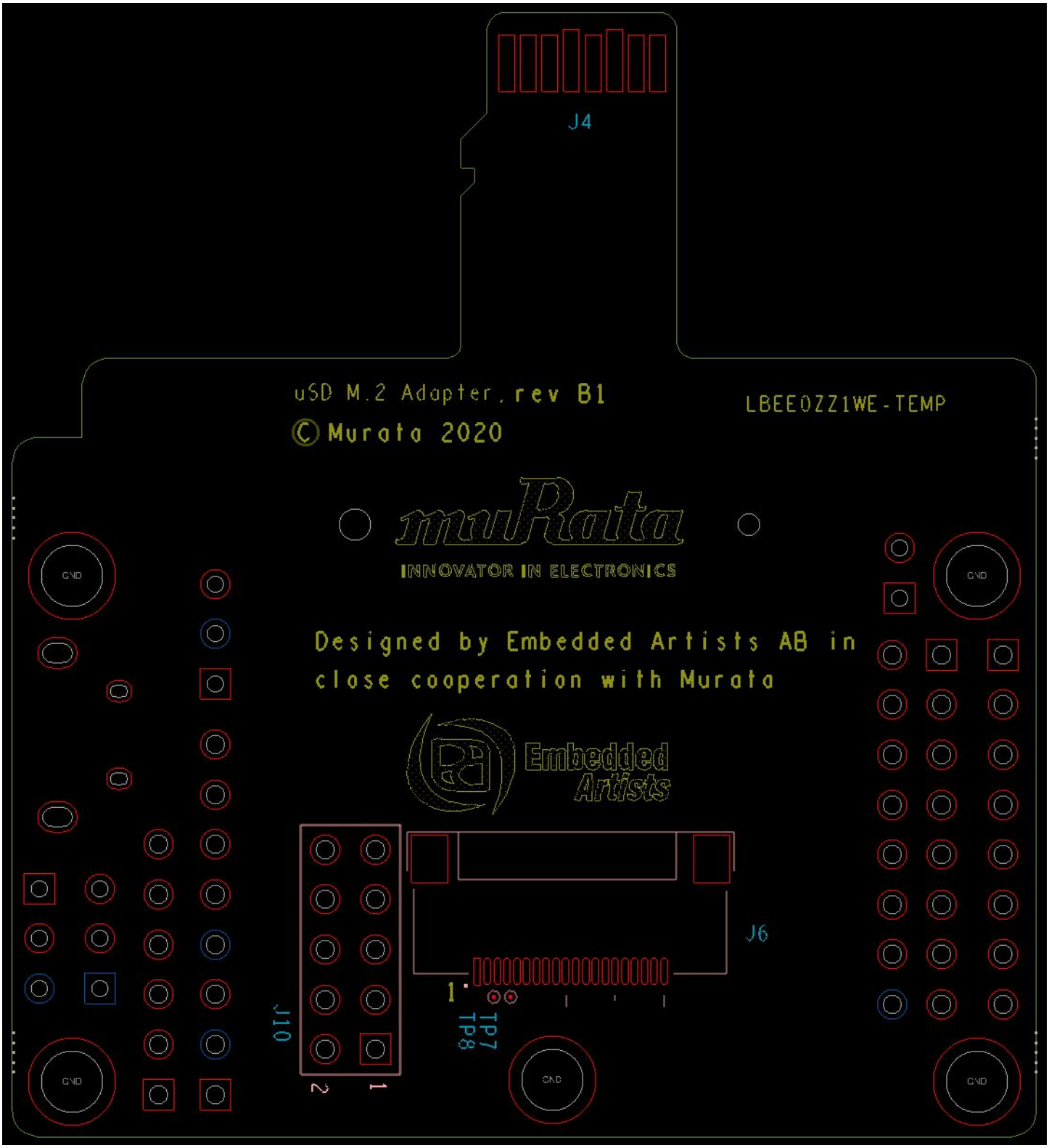


Figure 15: uSD-M.2 Adapter Layout (bottom)



10 Technical Support Contact

Table 7 lists all the support resources available for the Murata Wi-Fi/Bluetooth solution.

Table 7: List of Support Resources

Support Site	Notes
Murata Community Forum	**Primary support point for technical queries.** This is an open forum for all customers. Registration is required.
Murata i.MX Landing Page	No login credentials required. Murata documentation covering hardware, software, testing, etc. is provided here.
Murata uSD-M.2 Adapter Landing Page	Landing page for uSD-M.2 Adapter. In conjunction with Murata i.MX Landing Page, this should provide the user with comprehensive getting started documentation.
Murata Module Landing Page	No login credentials required. Murata documentation covering all Cypress-based Wi-Fi/BT modules is provided here.

11 Additional Useful Links

In addition to **Table 7** listings of support resources, **Table 8**, **Table 9**, **Table 10** and **Table 11** provides some useful links.

Table 8: NXP links

Link	Notes
MCUXpresso IDE	Landing page to download MCUXpresso IDE
MCUXpresso SDK	Comprehensive information of MCUXpresso SDK
MCUXpresso SDK Builder	Customize and build MCUXpresso SDKs
LPC-Link 2	Landing page of the debug probe for i.MX RT EVKs
i.MX RT 1050 EVK	Landing page of the i.MX RT 1050 EVK
i.MX RT 1060 EVK	Landing page of the i.MX RT 1060 EVK
i.MX RT 1064 EVK	Landing page of the i.MX RT 1064 EVK
i.MX RT 1160 EVK	Landing page of the i.MX RT 1160 EVK
i.MX RT 1170 EVK	Landing page of the i.MX RT 1170 EVK
i.MX RT 595 EVK	Landing page of the i.MX RT 595 EVK
i.MX RT 685 EVK	Landing page of the i.MX RT 685 EVK
i.MX RT 1050 Getting Started	Getting started guide for the i.MX RT 1050 EVK
i.MX RT 1060 Getting Started	Getting started guide for the i.MX RT 1060 EVK
i.MX RT 1064 Getting Started	Getting started guide for the i.MX RT 1064 EVK
i.MX RT 1160 Getting Started	Getting started guide for the i.MX RT 1160 EVK
i.MX RT 1170 Getting Started	Getting started guide for the i.MX RT 1170 EVK

Table 9: Embedded Artists' Landing Pages

Landing Pages	Notes
Embedded Artists' Website	The Art of Embedded Systems Development – made EASY™
i.MX RT COM Boards	Listing of Computer-on-Module boards.
i.MX RT COM Carrier Board V2	Main baseboard which all the COM boards plug into.
M.2 Module Family	Top level listing of 1ZM, 1YM, 1XK, 2DS M.2 EVBs.

Table 10: Embedded Artists' Datasheets and Schematics

Datasheets and Schematics	Notes
i.MX RT COM Carrier Board V2 Datasheet	Comprehensive definition of COM Carrier (baseboard).
i.MX RT COM Carrier Board V2 Schematics	Complete schematics including clear definition of uSD-M.2 Adapter.
M.2 SDIO Interface Schematic	Reference schematic for customers designing in WLAN-SDIO M.2 EVB.
M.2 PCIe Interface Schematic	Reference schematic for customers designing in WLAN-PCIe M.2 EVB.
EACOM Board Specification Guide	Comprehensive definition of Embedded Artists' Computer-On-Module's.
1ZM M.2 Module Datasheet	Comprehensive details on 1ZM Wi-Fi/BT M.2 Module.
1YM M.2 Module Datasheet	Comprehensive details on 1YM Wi-Fi/BT M.2 Module.
1XK M.2 Module Datasheet	Comprehensive details on 1XK Wi-Fi/BT M.2 Module.
2DS M.2 Module Datasheet	Comprehensive details on 2DS Wi-Fi/BT M.2 Module.

Table 11: Embedded Artists' User Manuals and Software

User Manuals and Software	Notes
Getting Started with M.2 modules and i.MX RT	How to bring up Embedded Artists i.MX RT Dev Kits.
Wi-Fi/BT M.2 EVB Primer	Introduction and drill-down on M.2 interface.

12 Appendix: HOST/M.2 VDDIO Voltage Settings (Rev B1 vs. Rev A)

Table 12 summarizes J13/J12 jumper settings for **Rev B1 of the uSD-M.2 Adapter**, indicating what Host and M.2 VIO voltages are being configured. The default configuration for J13/J12 (Host/M.2 VIO) is setting both jumpers in 1-2 position. This configures the M.2 VIO for WLAN-SDIO (and optional PCM) at 1.8 volts. The BT-UART and select WLAN-BT CTRL signals are level shifted from Host 3.3V to M.2 1.8V as necessary to adhere to the M.2 specification.

The "**3.3V Override**" configuration is used when the Host MPU/MCU platform **can only support 3.3V VIO signaling on WLAN-SDIO interface**. This override feature **only** works with M.2 EVB's that support 3.3V VIO signaling (currently 1DX and 1MW). The J13/J12 settings for this override mode are 1-2/2-3 respectively as shown in the block diagram.

Revision A of the uSD-M.2 Adapter **does not support level shifting** on BT-UART nor on select WLAN/BT CTRL signals. The limitation with the Rev A adapter is that the Host and/or M.2 interface may over-drive certain pins at 3.3V VIO which are configured for 1.8V input. This limitation has been **corrected** with Revision B1. Note the Rev A of the uSD-M.2 Adapter "3.3V Override" configuration is configured by connecting Jumper J12.

Table 12: **HOST/M.2 IO Voltage Levels Settings Figure** 16: Rev A (left) and Rev B1 (right) Adapter configured for default 1.8V VIO

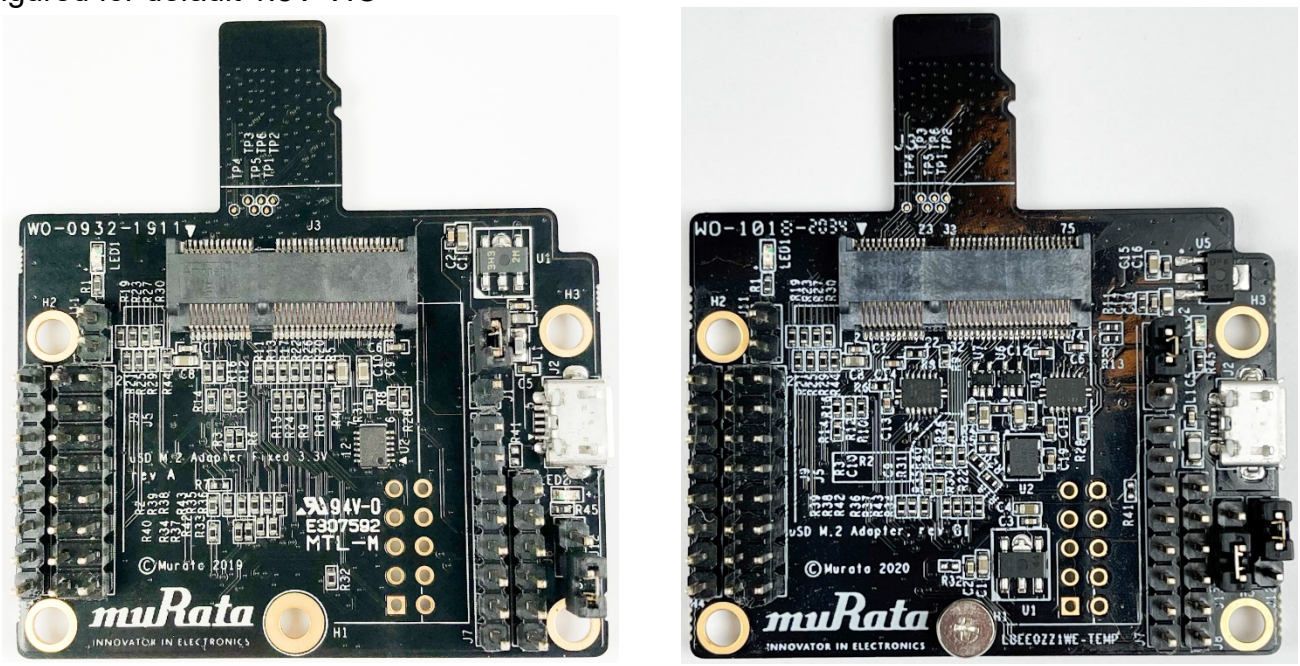


Figure 17: Rev A (left) and Rev B1 (right) Adapter configured for default 3.3V VIO Override

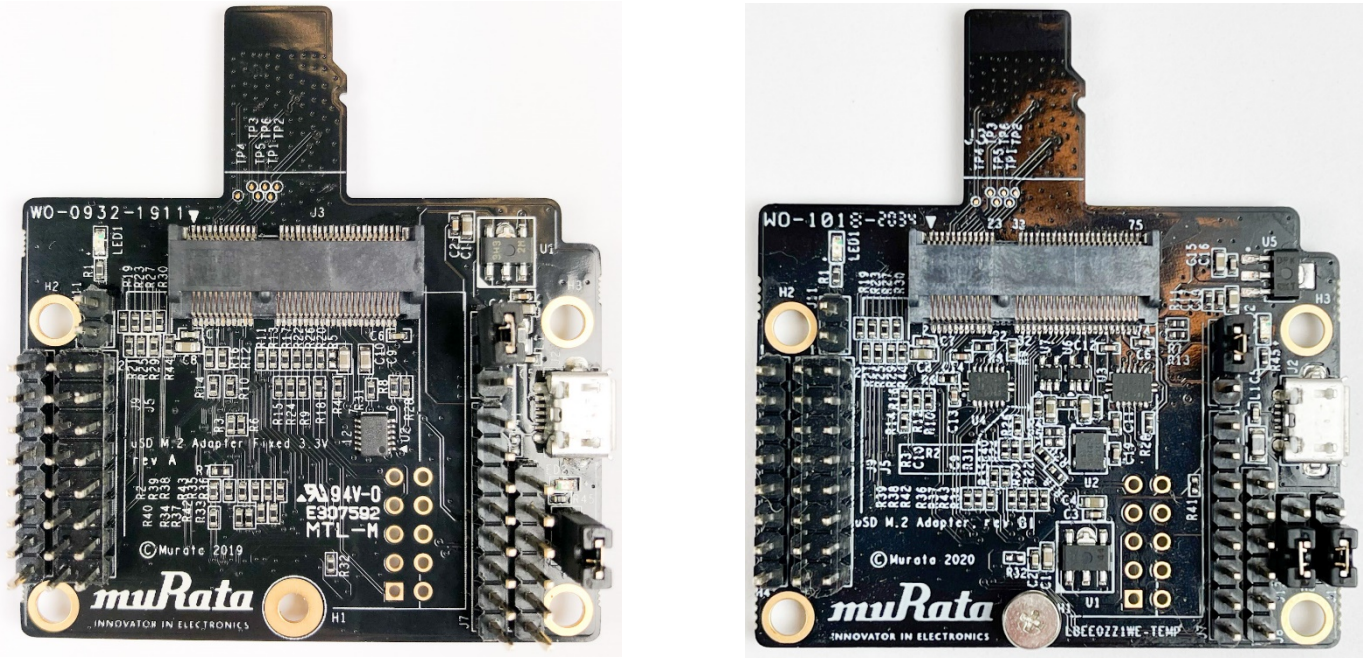


Figure 16 and **Figure 17** shows comparison between jumper J12 setting in between rev A and rev B1 adapter.

Figure 16: Rev A (left) and Rev B1 (right) Adapter configured for default 1.8V VIO

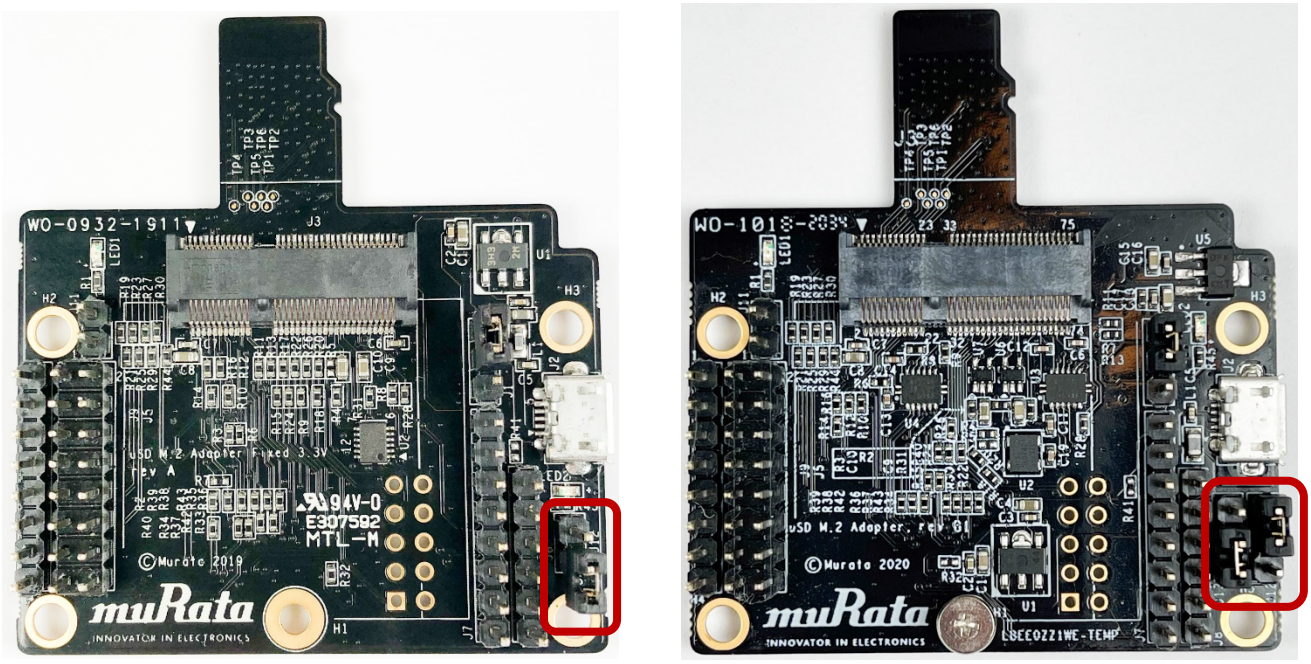
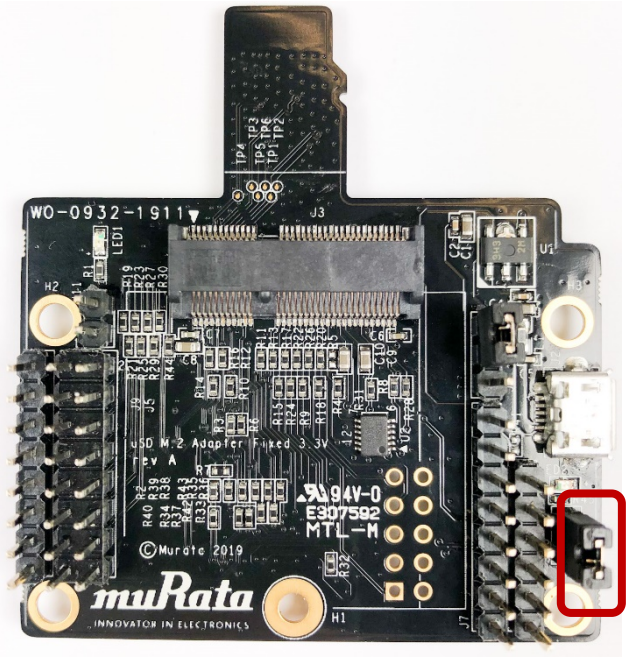
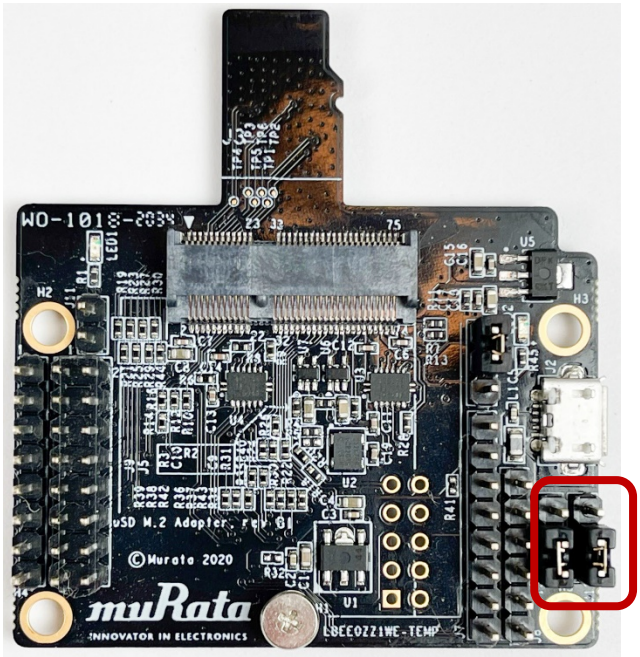


Figure 17: Rev A (left) and Rev B1 (right) Adapter configured for default 3.3V VIO Override





To configure 1.8V VIO (default) mode (blue LED2 is not illuminated):

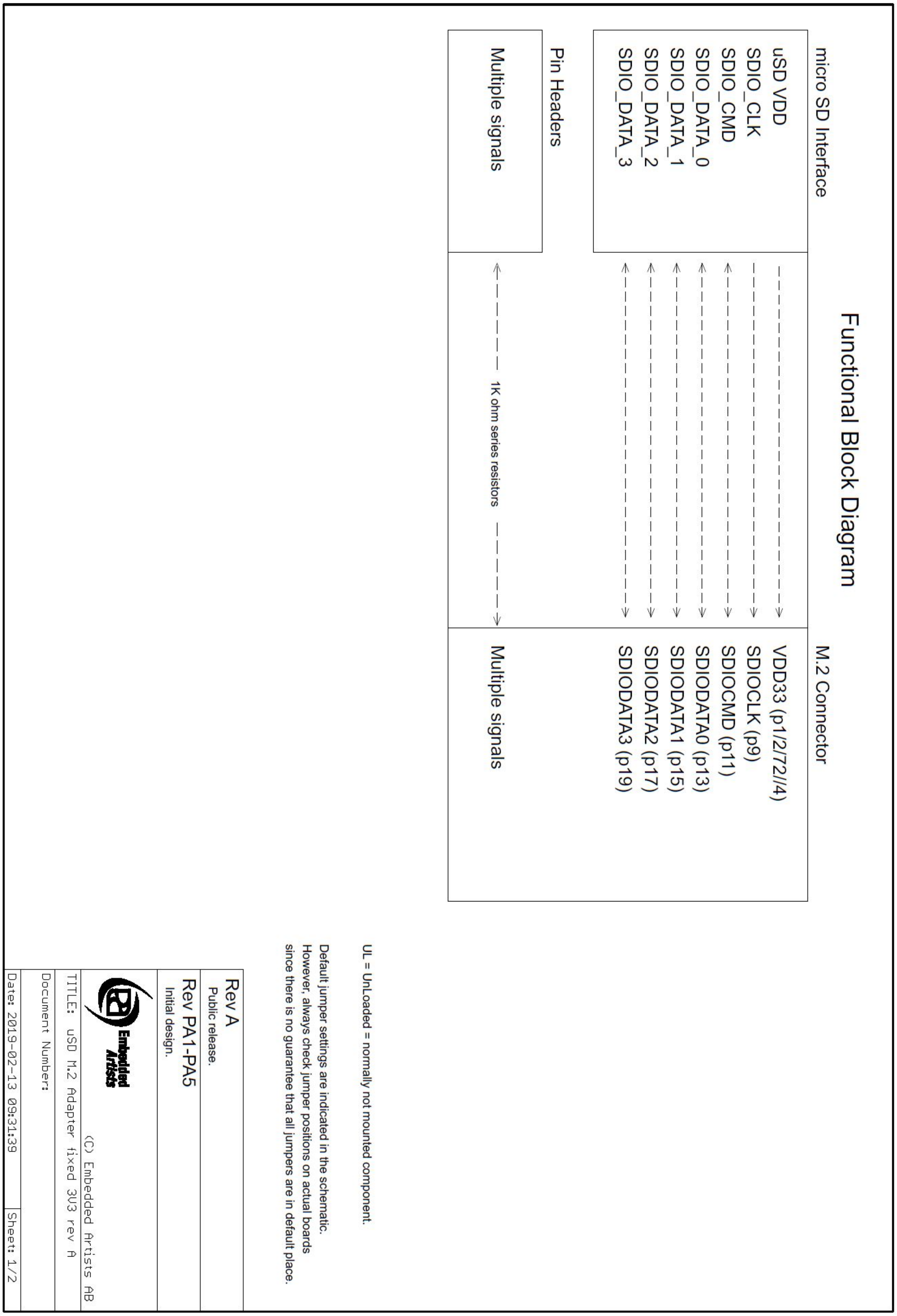
- For Rev A uSD-M.2 Adapter: Jumper J12 is removed/open.
- For Rev B1 uSD-M.2 Adapter:
  - Jumper J12 is installed in 1-2 position.
  - Jumper J13 is installed in 1-2 position.

To configure 3.3V VIO Override mode (blue LED2 illuminated):

- For Rev A uSD-M.2 Adapter: Jumper J12 is installed/shorted.
- For Rev B1 uSD-M.2 Adapter:
  - Jumper J12 is installed in 2-3 position.
  - Jumper J13 is installed in 1-2 position.

For more specifics on rev A adapter circuit and layout refer to **Figure 18: rev A uSD-M.2 Adapter Schematic**, **Figure 19: rev A uSD-M.2 Adapter Layout (top)**, and **Figure 20: rev A uSD-M.2 Adapter Layout (bottom)**.

Figure 18: rev A uSD-M.2 Adapter Schematic



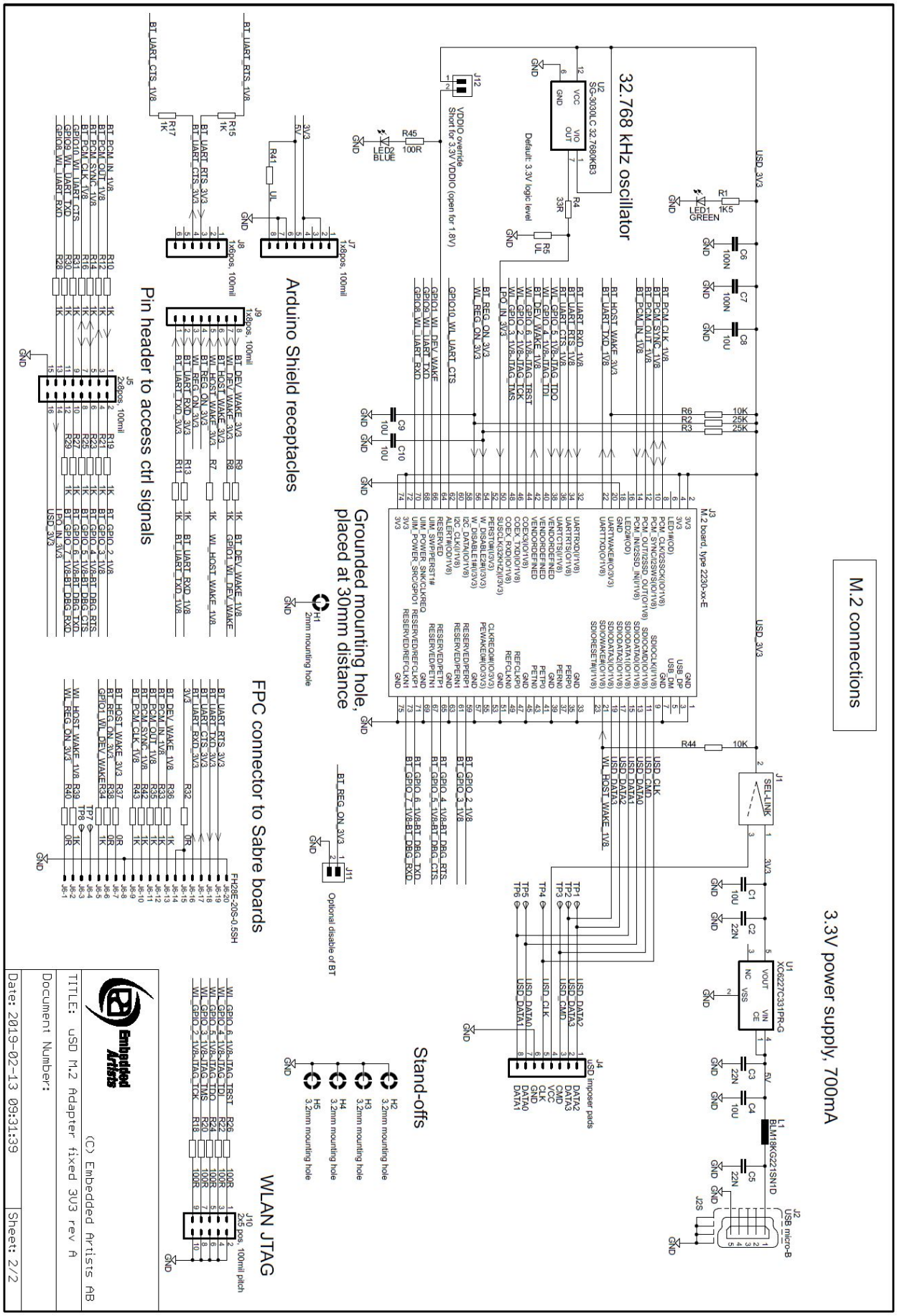

M.2 connections
3.3V power supply, 700mA
32.768 kHz oscillator
Default: 3.3V logic level
Grounded mounting hole, placed at 30mm distance
Arduino Shield receptacles
Pin header to access ctrl signals
FPC connector to Sabre boards
Stand-offs
WLAN JTAG
Optional disable of BT
VDDIO override
Short for 3.3V VDDIO (open for 1.8V)
M.2 board type 2230-xx-E
USD imposer pads
USD micro-B
SEL-LINK
Embedded Artists
(C) Embedded Artists AB
TITLE: uSD M.2 adapter fixed 3V3 rev A
Document Number:
Date: 2019-02-13 09:31:39
Sheet: 2/2

Figure 19: rev A uSD-M.2 Adapter Layout (top)

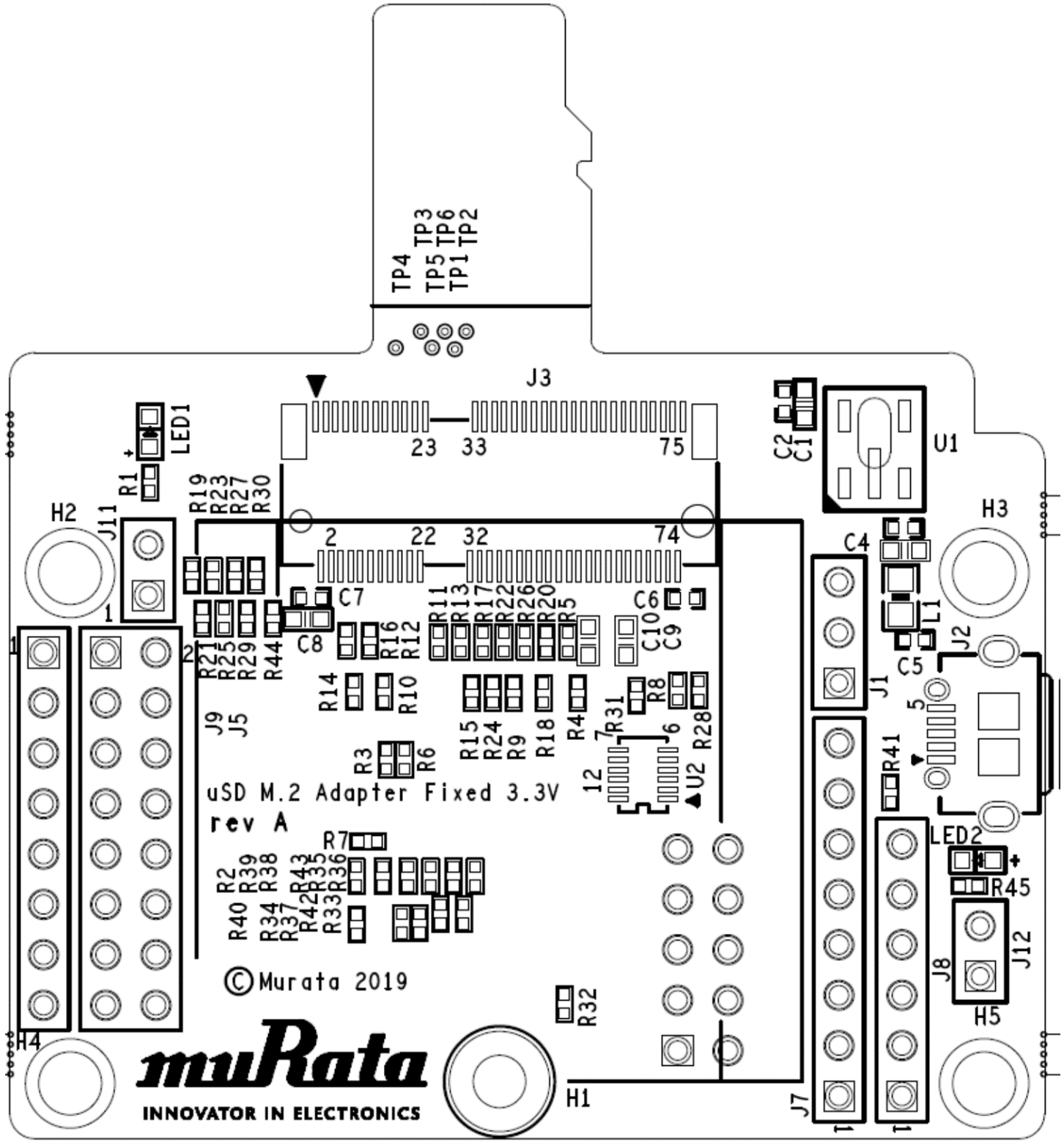


Figure 20: rev A uSD-M.2 Adapter Layout (bottom)