Morse Micro
reaching farther™

MM6108-EKH05

USER GUIDE

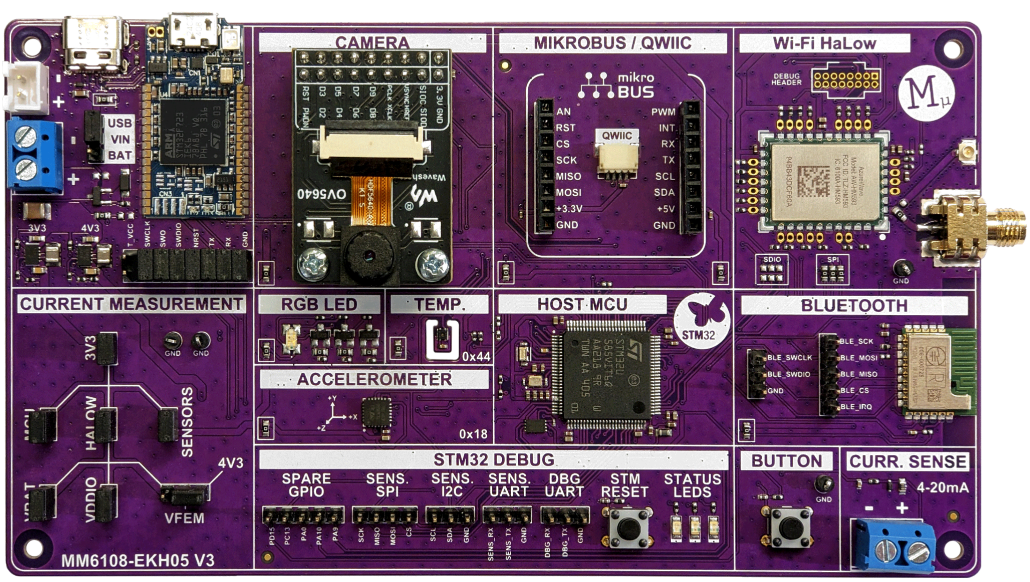
Table of Contents

1 Introduction

The MM6108-EKH05 evaluation kit is a fully integrated Wi-Fi HaLow development platform, designed for a wide range of IoT applications—from smart home devices to industrial automation systems. It features an Azurewave AW-HM593 module, incorporating Morse Micro's MM6108 Wi-Fi HaLow low-power SoC, alongside the STM32U585 low-power microcontroller (MCU) and the BlueNRG-M2 Bluetooth® SoC, the board provides robust wireless connectivity, low power consumption, and an extensive range of programmable interfaces and sensors. This versatile platform is ideal for software engineers developing energy-efficient IoT solutions.

This user guide provides a comprehensive overview of the MM6108-EKH05 evaluation kit, offering step-by-step instructions for setup, configuration, and programming. It covers all aspects of the board's capabilities, including the integration of Wi-Fi HaLow and Bluetooth® connectivity, utilization of the STM32U585 MCU, and leveraging the onboard sensors and interfaces. Detailed examples and practical use cases are provided to help developers quickly prototype and validate IoT applications, ensuring they can harness the full potential of this evaluation kit. The guide also includes troubleshooting tips, optimization techniques for low-power operation, and insights into best practices for IoT system design. Whether the user is new to IoT development or an experienced engineer, this guide is structured to help efficiently create innovative, connected solutions.

Note: At this time, a HaLow access point is required in addition to the MM6108-EKH05. Refer to Section 4 for setup details.

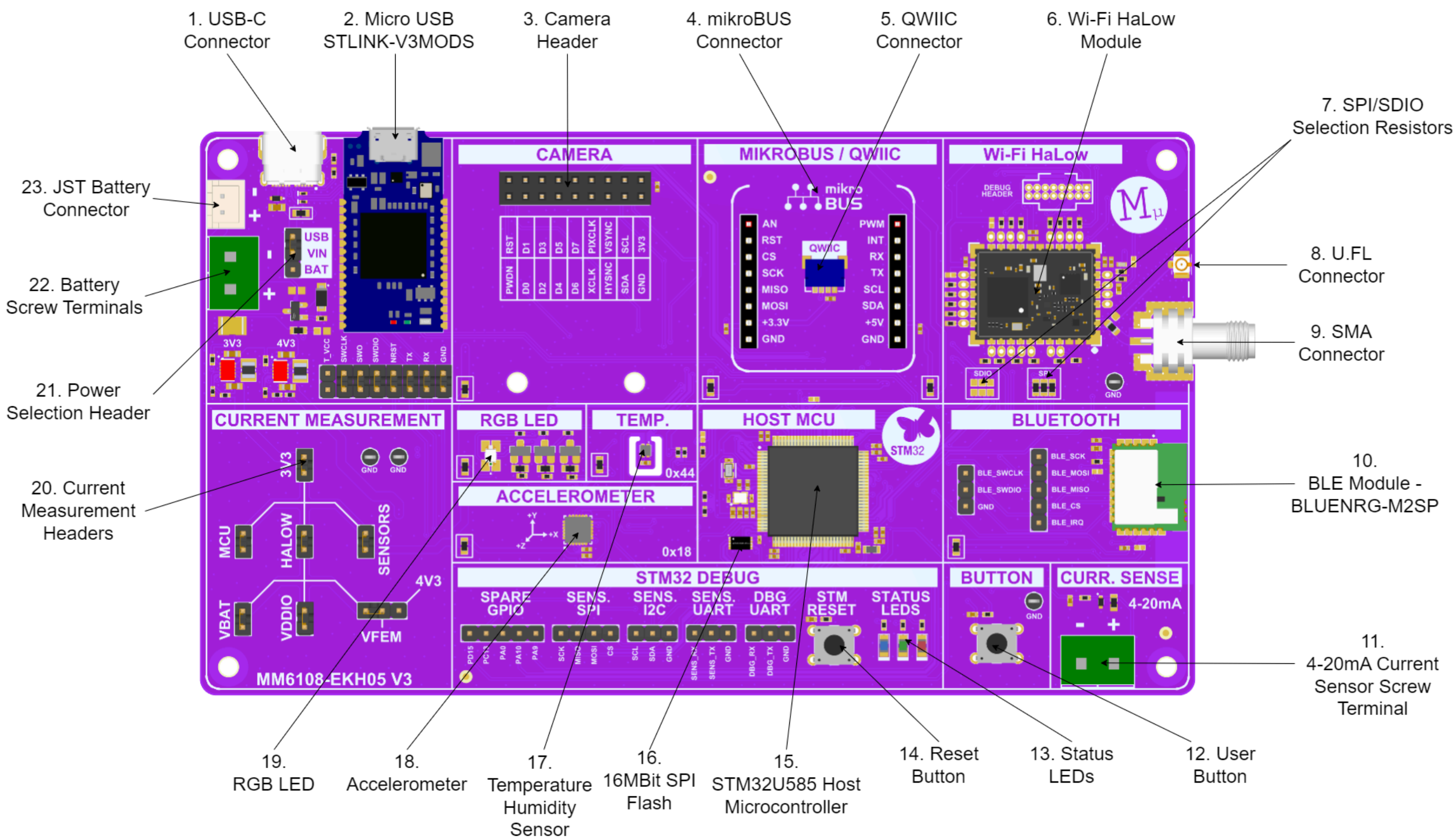


2 Features

This section highlights the key features of the MM6108-EKH05 Wi-Fi HaLow evaluation kit, showcasing the advanced capabilities that make it a versatile platform for IoT development.

Key features:

- Wi-Fi HaLow support for long-range, low-power wireless connectivity.
- STM32U585 microcontroller with energy-efficient architecture.
- Integrated programmable interface for flexible development.
- Integrated sensors for enhanced IoT applications.
- Multiple power input sources, including battery connectors.



#	Item	Description
1.	USB-C Connector	5V power supply. USB data lines connecting to STM32U585 host microcontroller.
2.	Micro USB STLINK-V3MODS	5V power supply. Programming and debugging access to STM32U585 host microcontroller. Access to STM32U585 host microcontroller UART. Note: The header pins below the STLINK-V3MODS module allow for disconnection from the host microcontroller for low-power testing.
3.	Camera Header	Camera connector to suit a DCMI (Digital Camera Interface) camera module. Connects to DCMI interface on STM32U585 host microcontroller. Note: A Camera module is provided in the MM6108-EKH05 kit.
4.	mikroBUS Connector	Standard connector for mikroBUS modules. Interfaces connect to STM32U585 host microcontroller.
5.,	QWIIC Connector	Standard connector for QWIIC modules. Interface connects to STM32U585 host microcontroller.
6.	HaLow Module	Azurewave AW-HM593 module with the MM6108 Morse Micro chipset. Connects to SPI interface (by default) on STM32U585 host microcontroller.
7.	SPI/SDIO Selection Resistors	Option to populate 0-ohm resistors for either SPI (default) or SDIO operation to STM32U585 host microcontroller. Note: Changing this setting also requires software modifications.
8.	U.FL Connector	Antenna output for U.FL connections.

		Note: SMA Connector is set by default. 0-ohm resistor change required to use U.FL connector. Instructions available in a chapter below.
9.	**SMA Connector**	Antenna output for SMA connections. Note: SMA is the default antenna output.
10.	**BLUENRG-M2SP Module**	ST BLUENRG-M2SP BLE module for provisioning devices. Connects to SPI interface on STM32U585 host microcontroller. Note: Access to SPI data lines and SWD data lines through header pins beside the BLUENRG module.
11.	**4-20mA Current Sensor Screw Terminal**	Connects to an ADC pin on the STM32U585 host microcontroller for 4-20mA operation.
12.	**User Button**	Programmable button for user applications. Connects to GPIO on STM32U585 host microcontroller.
13.	**Status LEDs**	Individual red, green, and blue LEDs for user applications. Connects to GPIO on STM32U585 host microcontroller.
14.	**Reset Button**	Resets the STM32U585 host microcontroller.
15.	**STM32U585 host microcontroller**	Ultra-low-power MCU with FPU ARM Cortex-M33 MCU with TrustZone, 160 MHz with 2 MBytes of Flash memory.
16.	**16MBit SPI Flash**	Winbond W25Q16JV SPI flash. Available as extra memory for user applications on STM32U585 host microcontroller
17.	**Temperature Humidity Sensor**	Sensirion SHT40 Ultra-Low-Power, 16-bit relative humidity and temperature sensor. Connects to I2C Interface on STM32U585 host microcontroller

18.	**Accelerometer**	ST IIS328DQ Ultra low-power 3-axis accelerometer with digital output. Connects to I2C Interface on STM32U585 host microcontroller.
19.	**RGB LED**	RGB LED connected to PWM interface on STM32U585 host microcontroller.
20.	**Current Measurement Headers**	Headers for power measurement. Follows a tree structure allowing measurement of either the entire system or individual blocks.
21.	**Power Selection Header**	Header to select system power source. Options: 1. USB power from USB-C or STLINK 2. Battery JST Connector or Battery screw terminals
22.	**Battery Screw Terminals**	Option for connecting a battery to the system through screw terminals.
23.	**JST Battery Connector**	Option for connecting a battery to a system through a standard JST connector.

3 Development Environment

3.1 System Environment

All major operating systems are supported:

- Windows 10, Windows 11
- Linux 64-bit
- macOS (macOS 14 and below.)

The development options are described in the following section.

3.2 Development Options

Morse Micro provides three development options for the MM6108-EKH05. The CMSIS Pack with STM32CubeIDE fully supports all hardware and includes pre-loaded example applications for immediate, out-of-box use. The other development options are available but have limited support for the hardware peripherals of the MM6108-EKH05.

3.2.1 CMSIS Pack

A CMSIS (Cortex Microcontroller Software Interface Standard) pack is a standardized software package format defined by ARM to facilitate the development of software for ARM Cortex-based microcontrollers. It bundles together all necessary components, such as device drivers, middleware, libraries, and configuration files, into a single package that can be easily distributed and integrated into development environments. The Morse Micro CMSIS pack includes everything needed to integrate Wi-Fi HaLow applications using this standardized package.

Morse Micro CMSIS pack releases can be found via:

https://www.morsemicro.com/downloads-dashboard/#/software-releases/iot/cmsis-pack

The instructions for using this package with STM32CubeIDE as the supported IDE are included in section 5 of this document.

3.2.2 MM IoT SDK

The Morse Micro IoT SDK provides a framework for developing embedded applications that include connectivity provided by Morse Micro MM6108 chip. This SDK package contains the software components to build the application firmware for the STM32 processor. These components include the RTOS (FreeRTOS), network stack (LwIP or FreeRTOS + TCP), Morse Micro host software (morselib), and the platform-specific board support package (BSP).

The MM IoT SDK release packages can be found via:
https://www.morsemicro.com/downloads-dashboard/#/software-releases/iot

The documentation for this package is included in each release.

3.2.3 PlatformIO + MM IoT SDK

PlatformIO is an open source multi-platform build system with integration available for various IDEs. It has powerful built-in features such as debugging, unit testing, and static code analysis and it supports a multitude of hardware platforms. PlatformIO also has a vast amount of libraries and example code available for various sensors and actuators used in IoT applications. The MM IoT SDK provides an integration for PlatformIO that supports multiple development boards from ST Microelectronics combined with Morse Micro extension board for developing Wi-Fi HaLow enabled applications.

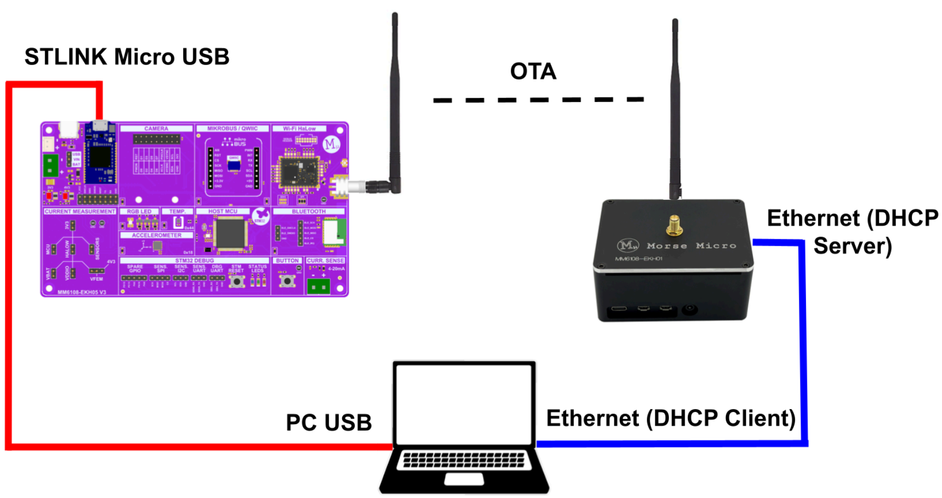
The user guide for the PlatformIO can be found from the following link on the Morse Micro support portal:
https://www.morsemicro.com/download/ug-platform-io-mm-iot-sdk-user-guide/.

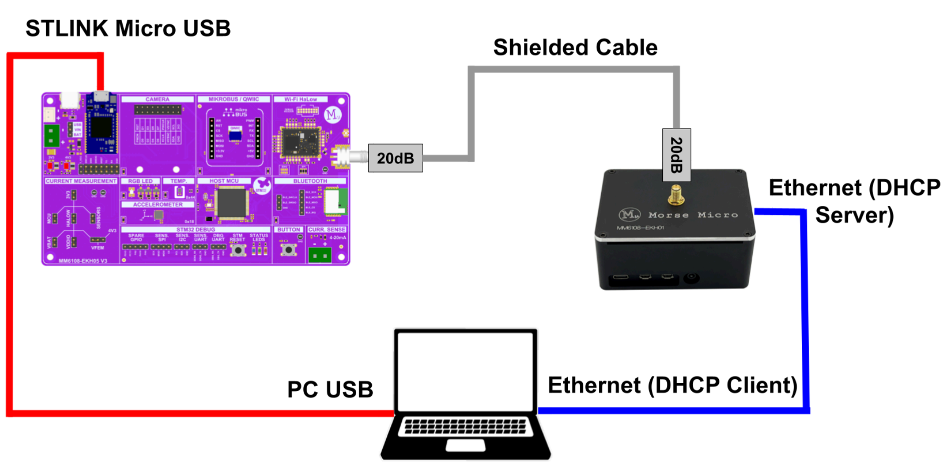
4 Getting Started

The most straightforward setup involves connecting a host PC to both the MM6108-EKH05 as a station (STA), and a separate Wi-Fi HaLow access point (AP). The recommended AP is the MM6108-EKH01-05US evaluation kit, which will be referenced throughout this document.

Note: At this time, the MM6108-EKH05 **cannot** operate as an AP.

The MM6108-EKH05 connects through the STLINK Micro USB connector, and the AP connects through an Ethernet cable. The Host PC must set the Ethernet interface to a DHCP client - see [AP Device Settings](#) . The HaLow connection can either be Over The Air (OTA) through antennas, or cabled. For cabled, it is recommended to use 40 dB of external attenuation as shown below. The device can be damaged if connected directly without attenuation.

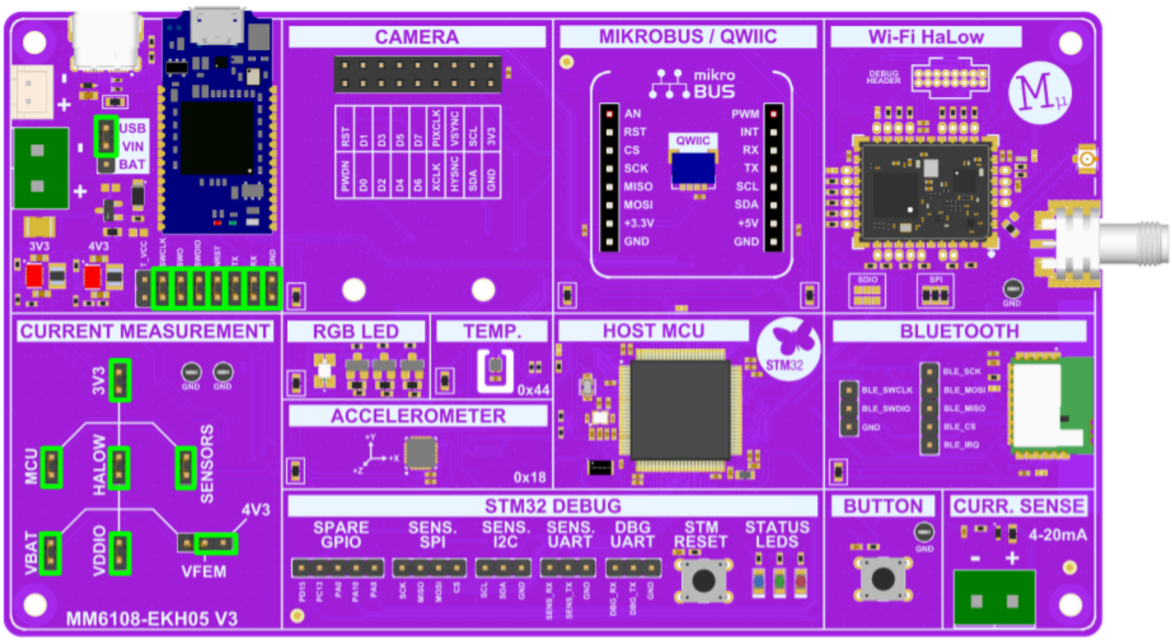




4.1 Default Jumper Configuration

By default, the following headers are populated:

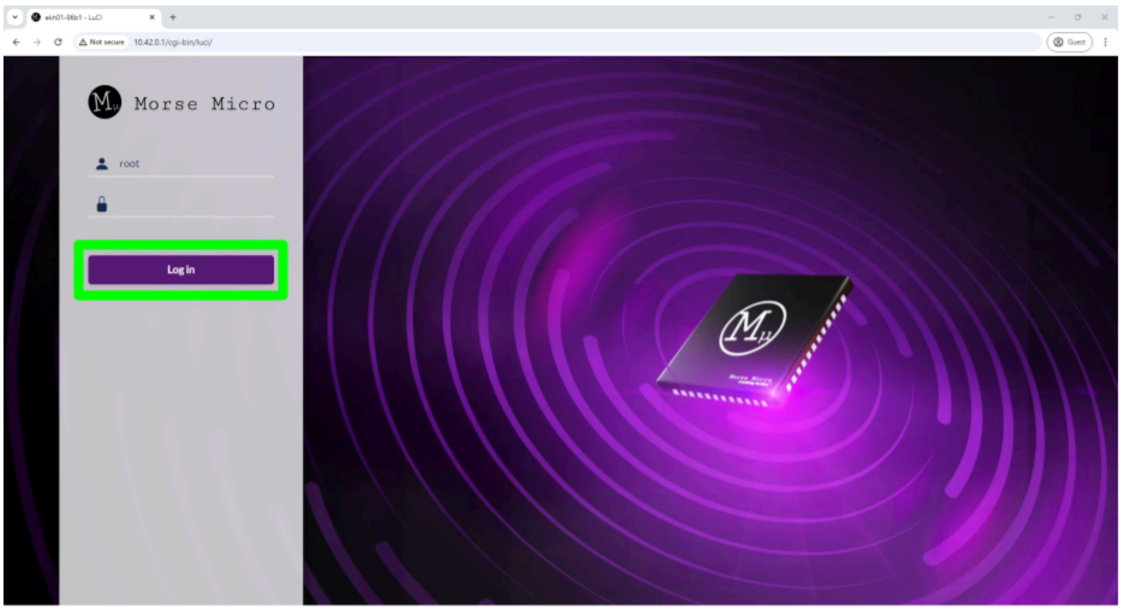
1. USB power selection header
2. All STLINK headers (except for T_VCC)
3. Current measurement headers



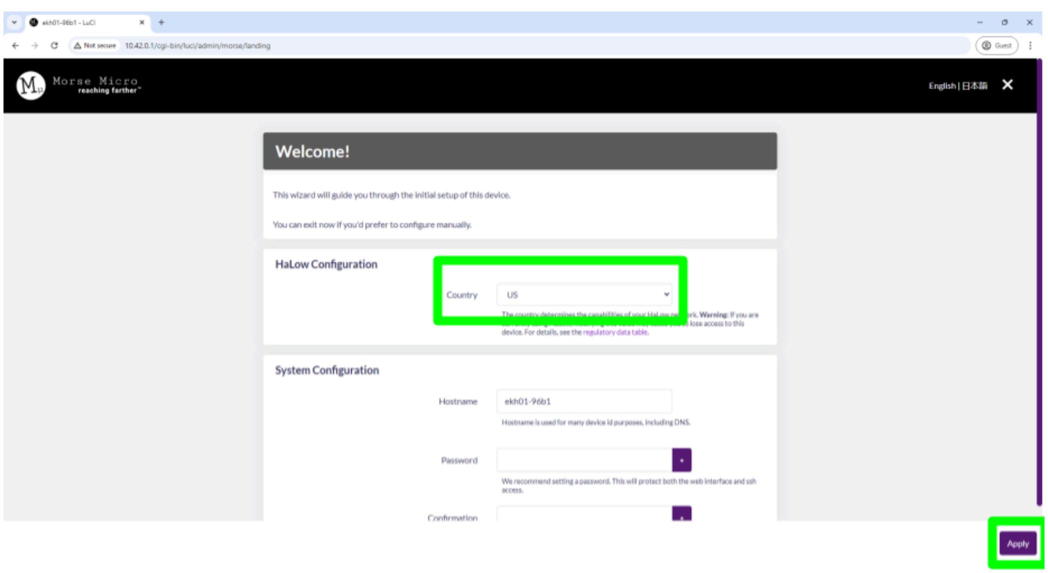
4.2 AP Device Settings

This section describes how to set up a MM6108-EKH01-05US evaluation kit as an access point (AP). These steps assume the EKH01 AP has been freshly flashed with a OpenWRT 2.6.6 image. For more information, the User Guide for the EKH01 can be found on the Customer Portal.

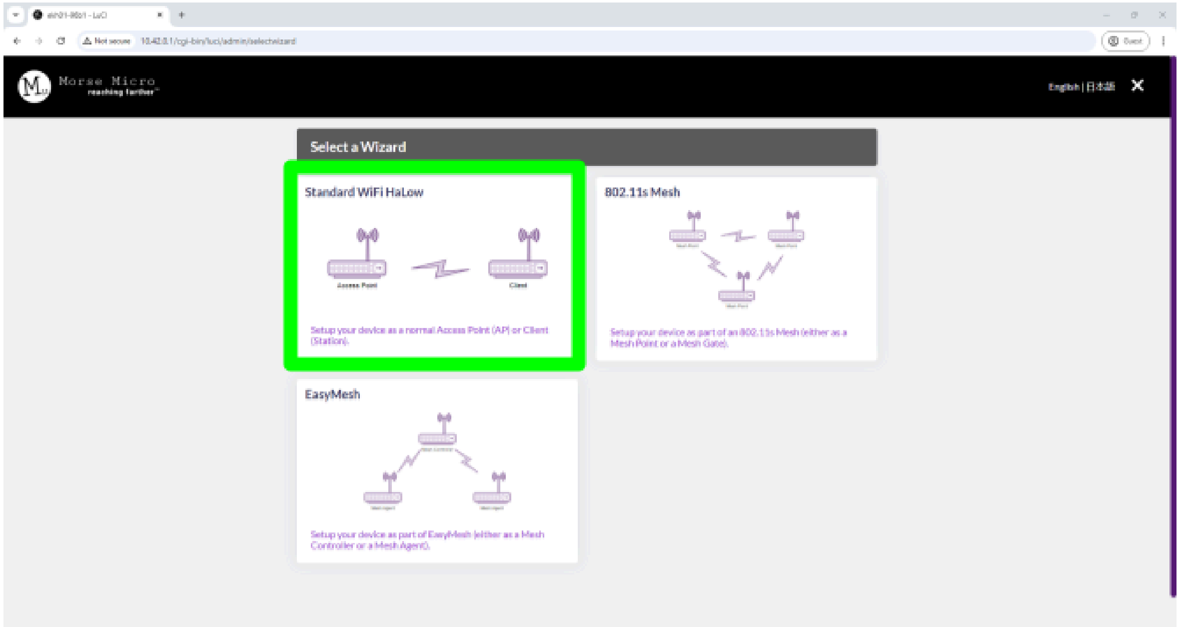
Navigate to **10.42.0.1** in a web browser. Click **Log in**.



Select which **Country**, and optionally, there is the ability to set a new hostname and passphrase. Click **Apply** after making changes.



Select **Standard Wi-Fi HaLow**.



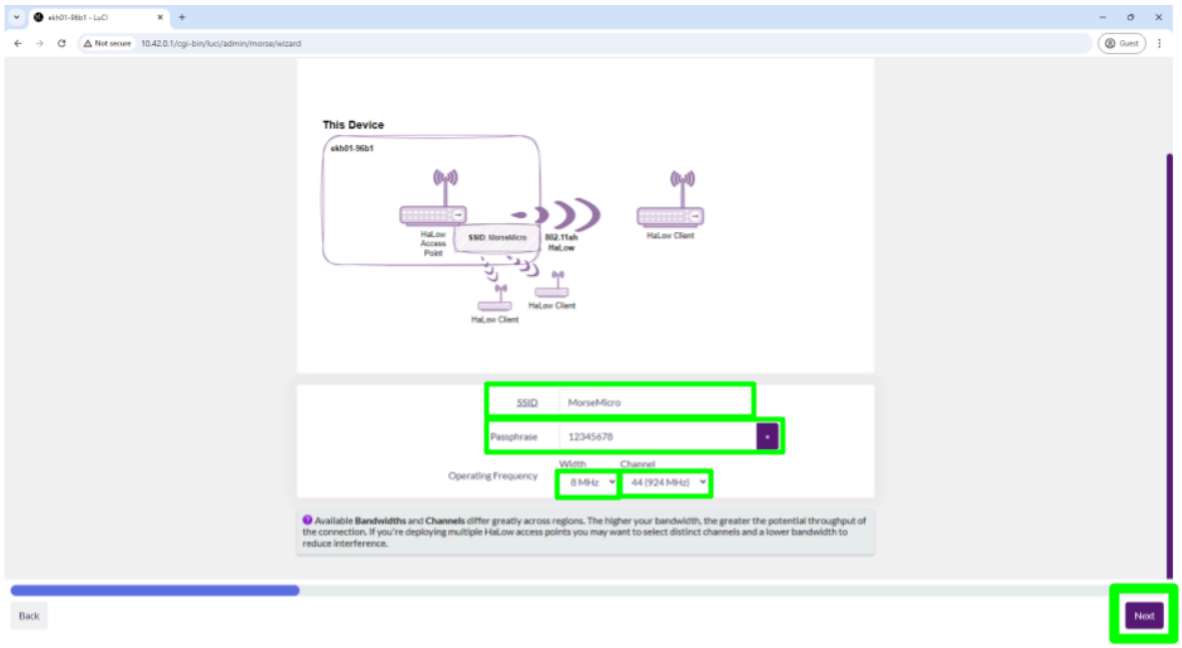
Select **Access Point**, and click **Next**.

Here is where the **SSID** and **Passphrase**, **Bandwidth** and **Channel** can be set. Default SSID and passphrases for Morse Micro example applications are:

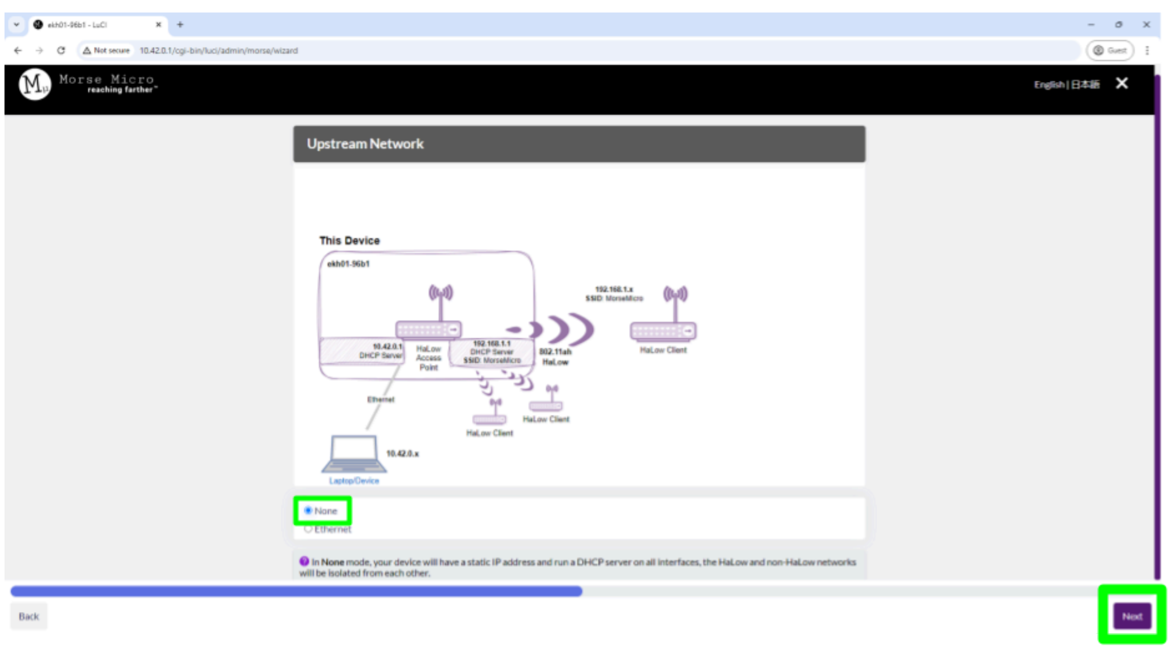
SSID: MorseMicro

Passphrase: 12345678

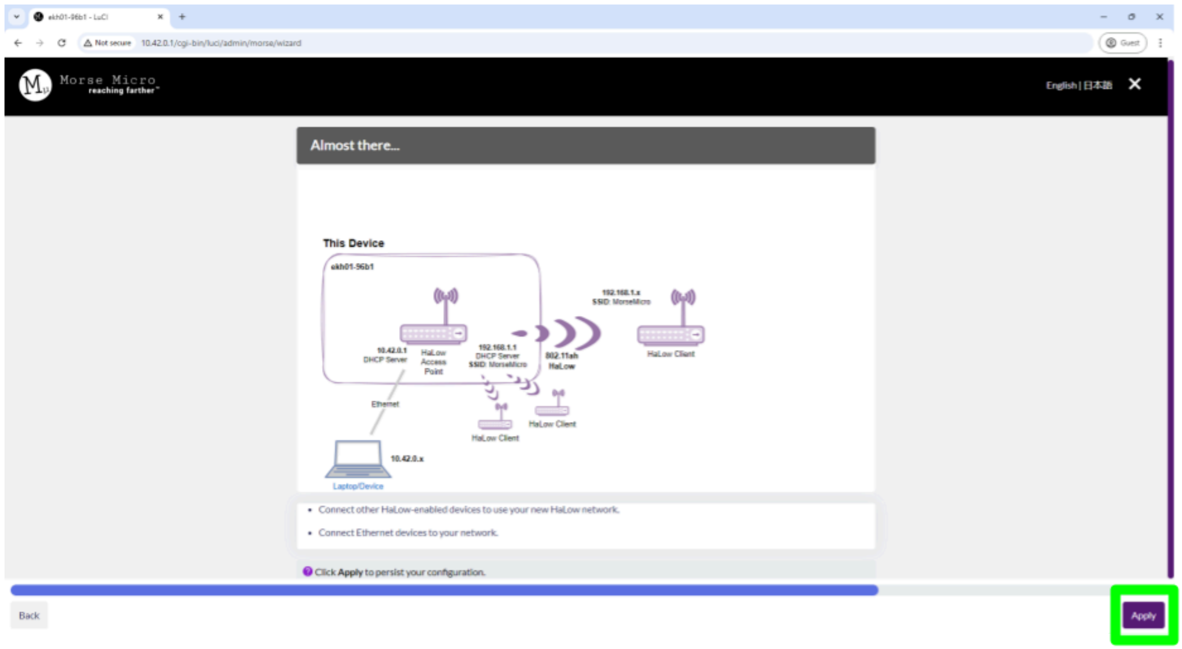
After configuring, click **Next**. These parameters can be changed later if needed.



Set the Upstream Network to **None**, and click **Next**.



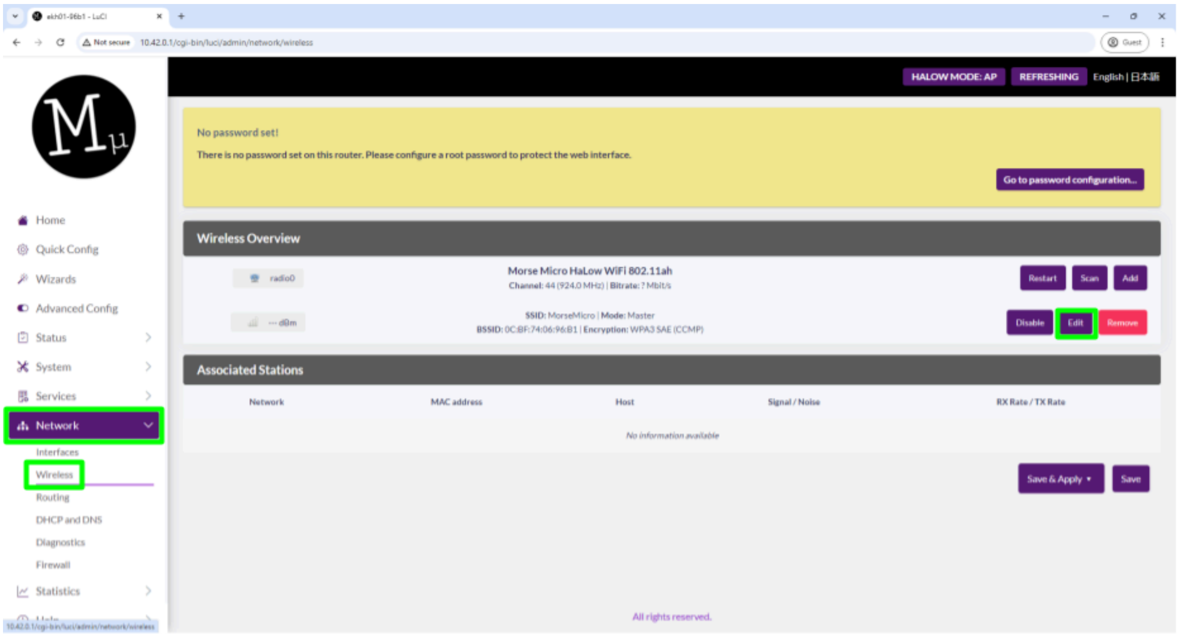
Click **Apply**.



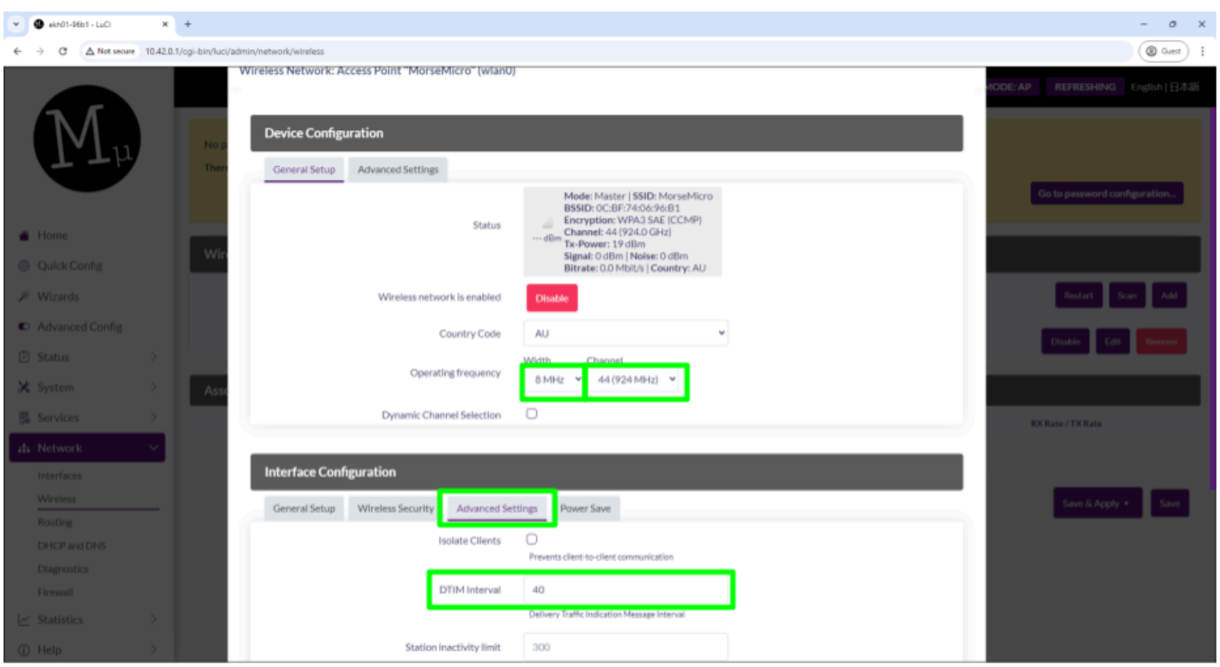
The device is now set up as an AP.

4.2.1 Changing Channel, Bandwidth, DTIM Period

After initial configuration, the channel, bandwidth, and DTIM period can all be changed by navigating to the **Network → Wireless** page, and then clicking **Edit** on the HaLow radio.



The **Bandwidth** and **Channel** can then be changed from the **Device Configuration** menu. The **DTIM Interval** can then be changed by selecting **Advanced Settings** from the **Interface Configuration** menu.



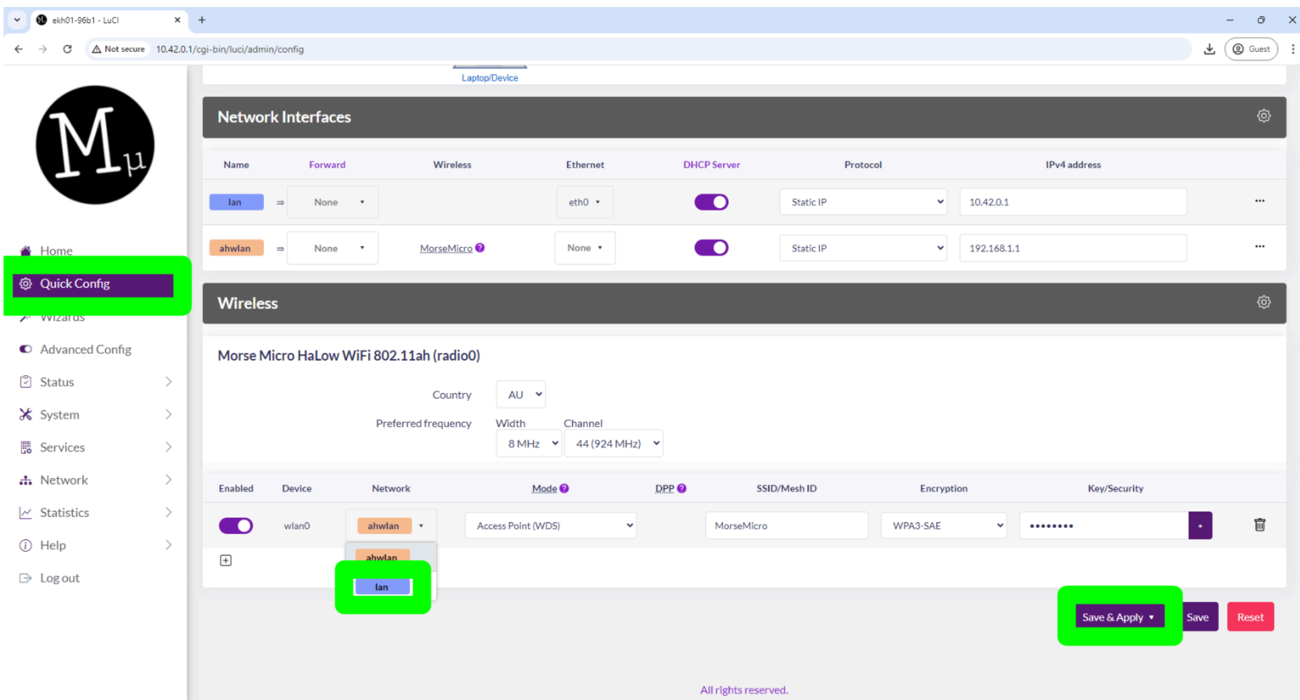
4.3 Software Examples

The following table will describe the list of supported software examples that are included with the CMSIS pack for the MM6108-EKH05. The "EKH05 Demo" example is loaded on the device out of the box.

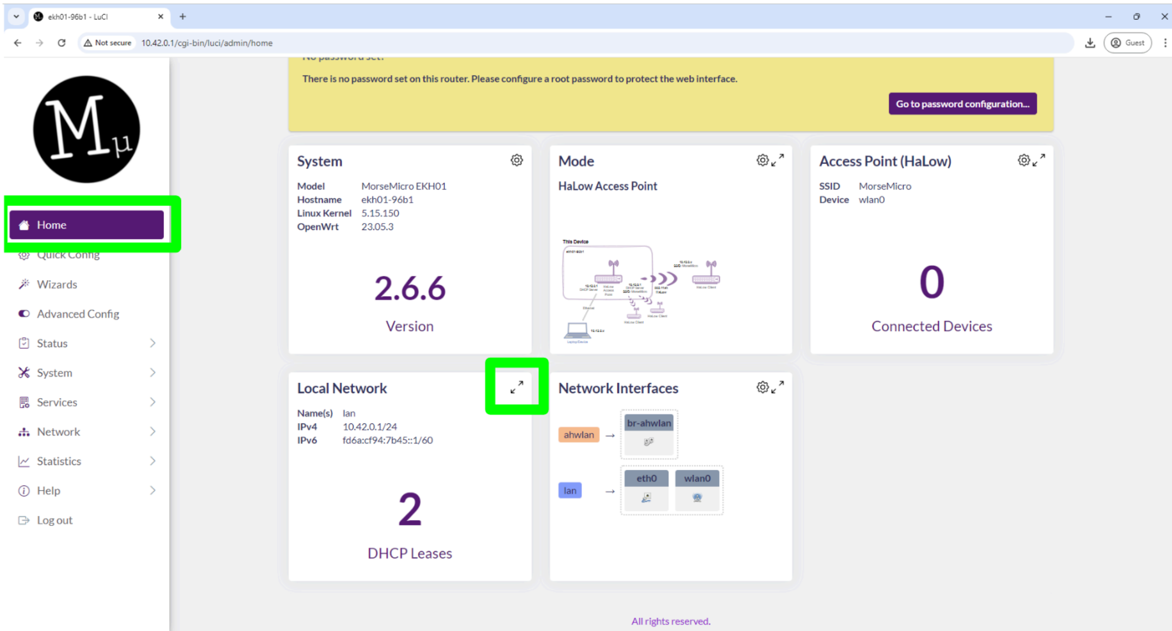
Software Example	Description
EKH05 Demo	As the default out-of-box example, the EKH05 Demo showcases the operation of all on-board peripherals of the MM6108-EKH05. A HTTP server runs on the device and displays images from the camera, temperature, humidity and accelerometer values. The BLE module is also activated and acts as a sensor.
Ping	Simple ping demonstration. Also used to stay connected to an AP while idling to measure DTIM sleep currents.
WNM Sleep	Example utilizing WNM sleep to conserve power in between periodic transmissions.
iPerf	Example showing how to perform throughput measurements using iPerf.
AWS IoT	AWS IoT example to demonstrate connecting to AWS Shadow service and demonstrate a simple light bulb example.
Scan	Example application to demonstrate scanning for Wi-Fi HaLow networks.

4.4 Viewing MM6108-EKH05 Demo HTTP Server

To view the HTTP server running on the MM6108-EKH05 device from a PC, the AP will need to bridge the HaLow and Ethernet interfaces together. To do this, first click **Quick Config** from the left hand panel. Scroll down to the **Wireless** section, click the drop down arrow under **Network**, and select **lan**. Click **Save & Apply** for the changes to take effect.



The MM6108-EKH05 device needs to be restarted at this point to obtain the new DHCP lease. To determine the new IP address it has taken, navigate back to the **Home** section, and expand the **Local Network** tile.



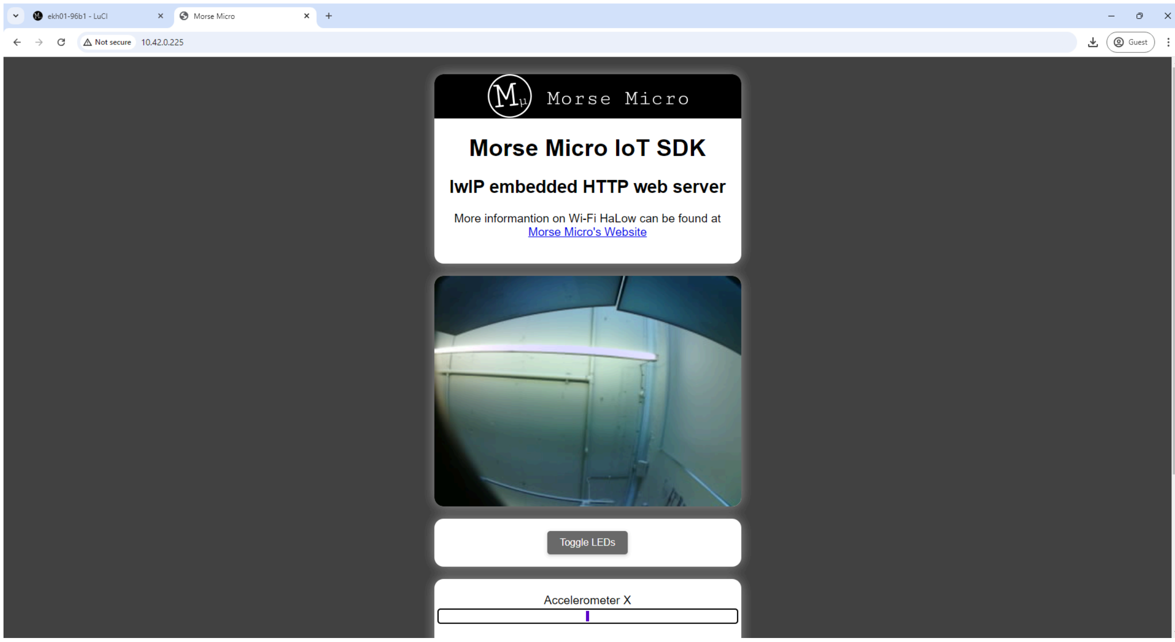
Here is where to find the DHCP leases the AP has distributed. The first lease given out is the HaLow connection at 10.42.0.225, and the second is the lease given to the host PC.

Local Network

IPv4 10.42.0.1/24
IPv6 fd6a:cf94:7b45::1/60

MAC Address	Hostname	IPv4	Expiry	IPv6	IPv6 Expiry
02:00:73:64:4D:79		10.42.0.225	720 min(s)		
00:E0:4D:6C:88:9A	jono_xps.lan	10.42.0.209	686 min(s)		

Navigate to the IP address of the MM6108-EKH05 (10.42.0.225 in this case) in a web browser to view the contents of the HTTP server.



Important: This will change the HaLow IP address of the AP to be on the same 10.42.0.x IP range. All other example applications such as Ping default to using 192.168.1.1 as the AP IP address. To ensure other example applications still function as expected, the IP address must either be changed in the example applications, or the bridging step can be reversed by selecting **ahwlan** under the **Network** drop down menu. **Save & Apply** must once again be pressed for this change to take effect.

5 Software Development

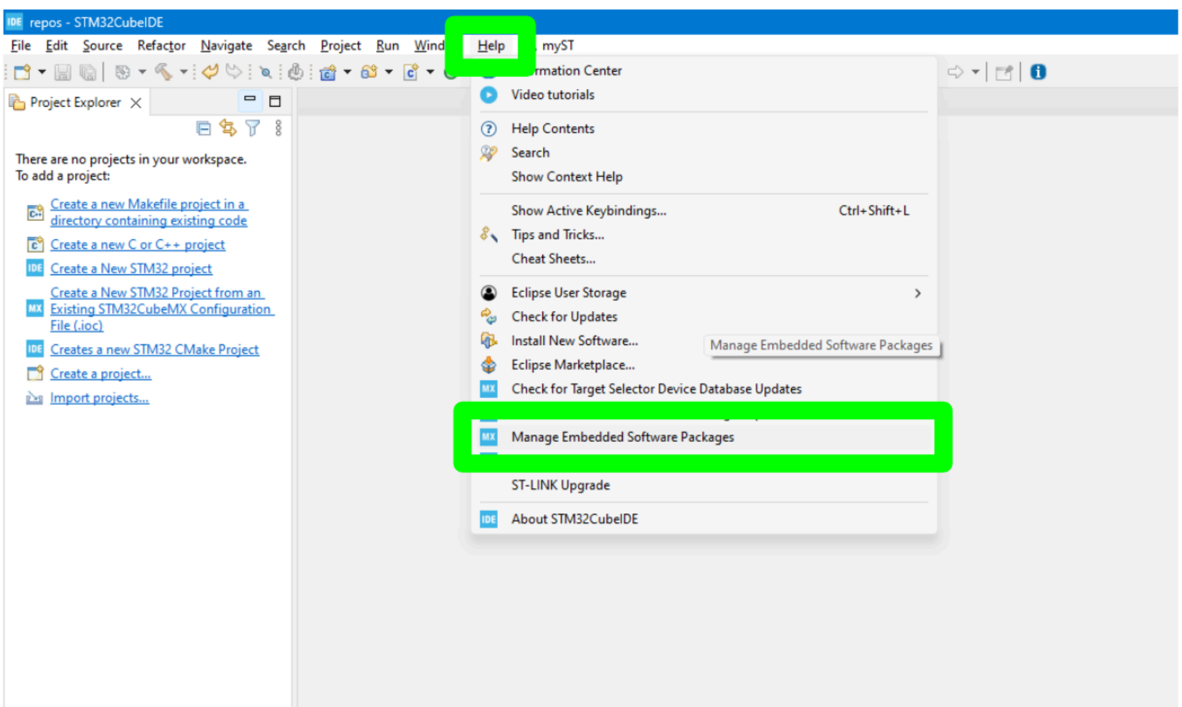
This section will describe how to install the CMSIS package, how to build and run Morse Micro example applications within the STM32CubeIDE and how to update the firmware on the BLUENRG-M2SP module.

STM32CubeIDE can be downloaded from the ST website - https://www.st.com/en/development-tools/stm32cubeide.html.

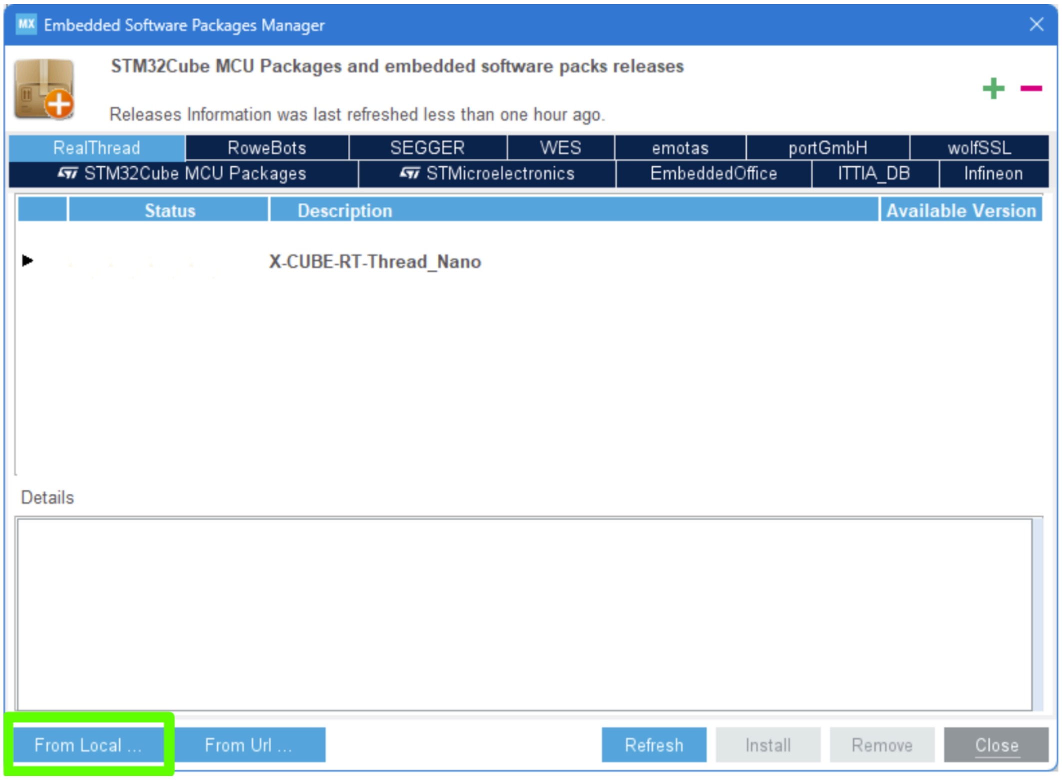
The CMSIS package can be downloaded from the Morse Micro customer portal - https://www.morsemicro.com/downloads-dashboard/#/software-releases/iot/cmsis-pack.

5.1 Installing CMSIS Package

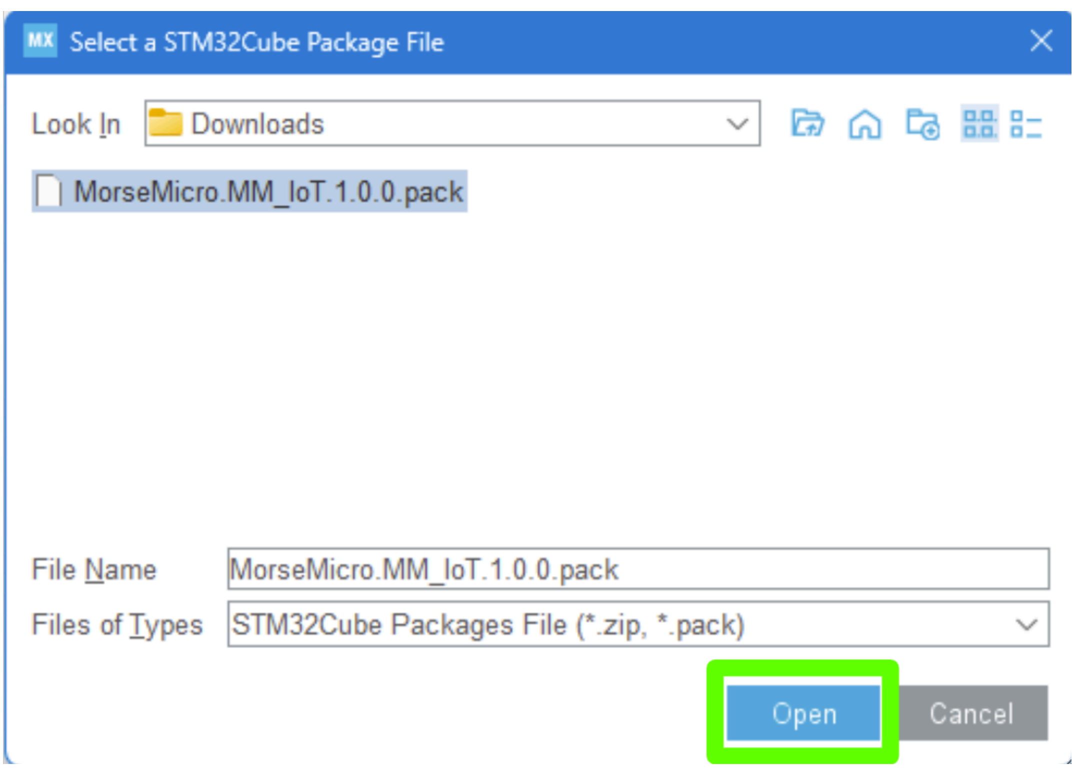
After Starting STM32CubeIDE, click on **Manage Embedded Software Packages** from the **Help** menu:



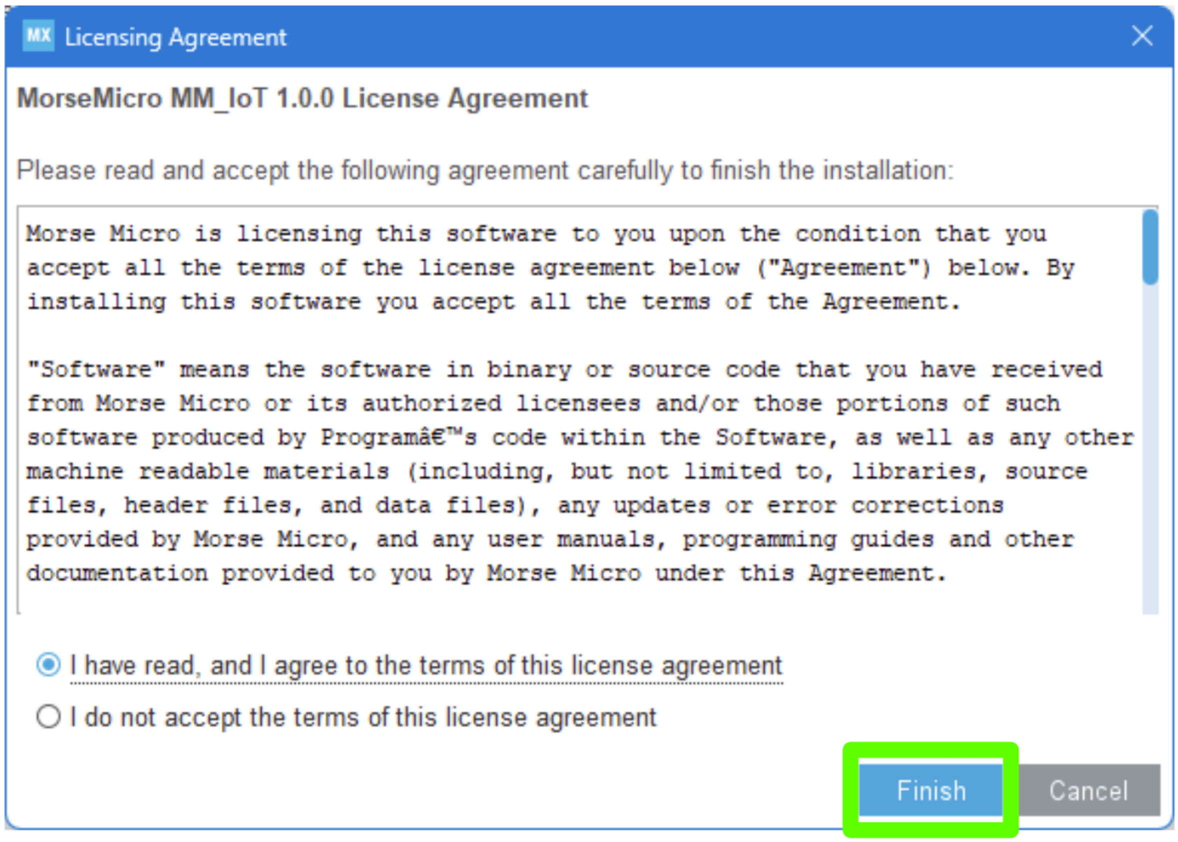
Click on **From Local ...**:



Navigate to where the Morse Micro CMSIS package is downloaded, and click **Open**:



Accept the license agreement, and click **Finish**:

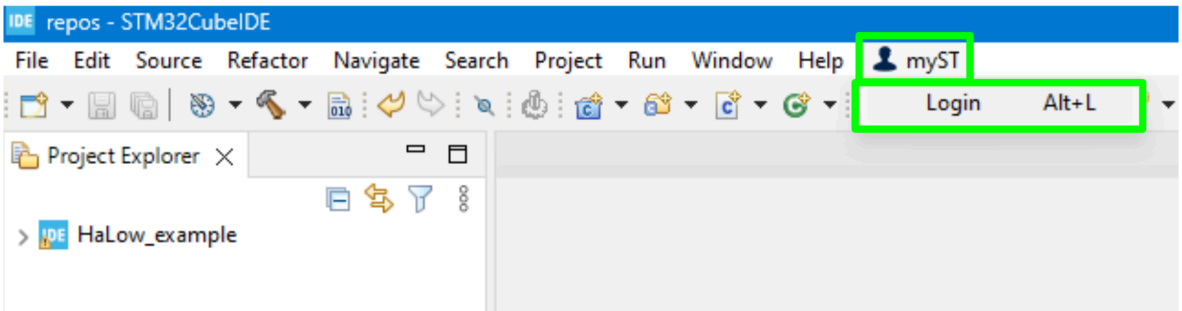


The package is now installed.

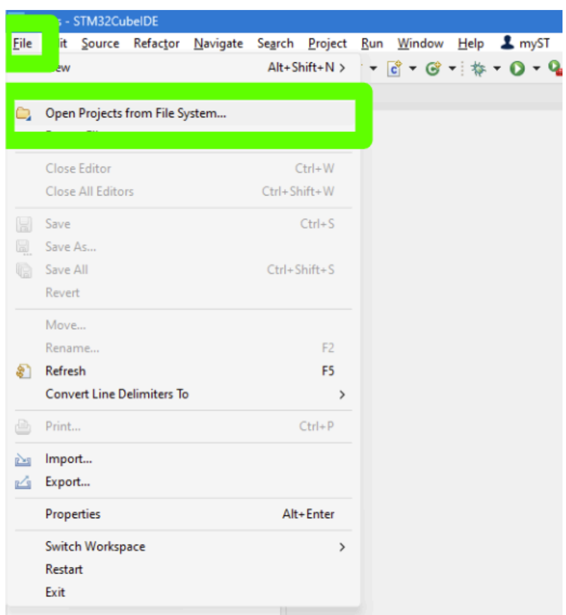
5.2 Build and Run Example Applications

After installing the Morse Micro CMSIS package, an example application can be run. As an example, this will run the HaLow Example application.

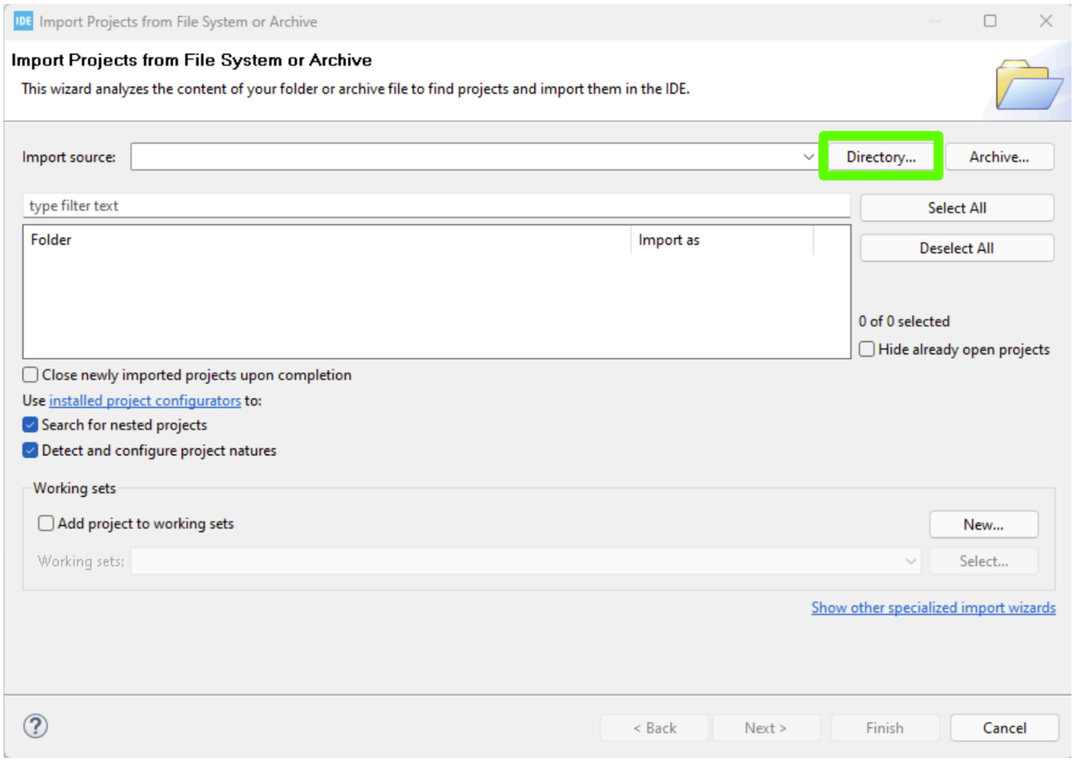
First, create and log into a myST account. This is needed to install other packages included within the HaLow example.



From **File**, click **Open Projects from File System**:



Click on **Directory...**:



Navigate to the default STM32 package location, and then find the HaLow_example application.

On Windows, the default path is:

```
C:\Users\USER_NAME\STM32Cube\Repository\Packs\MorseMicro\MM_IoT\1.3.3\HaLow_example
```

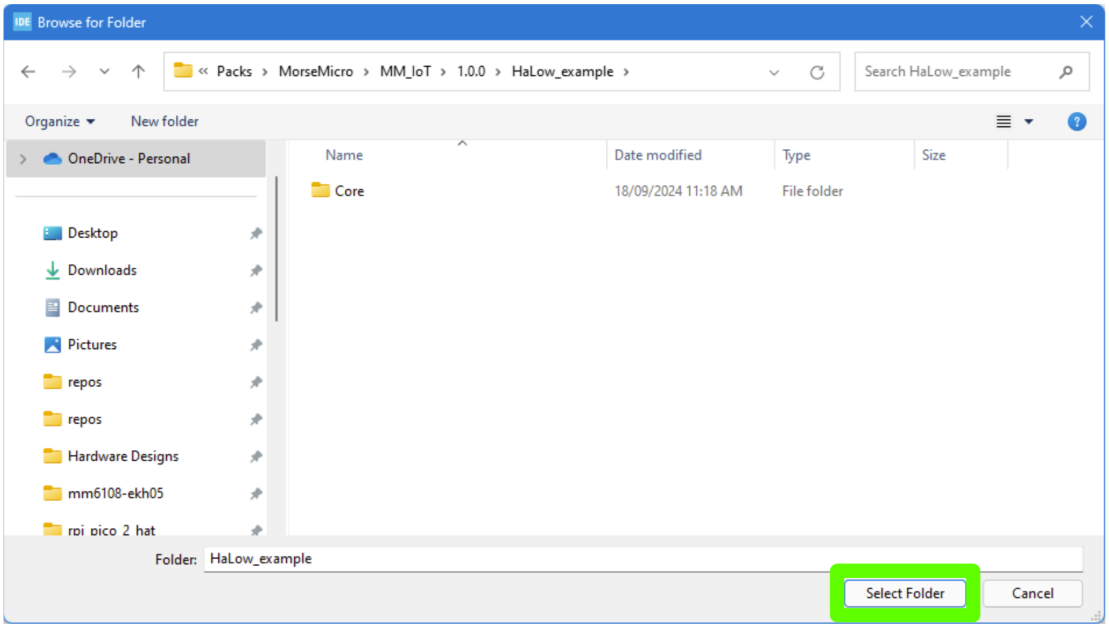
On Linux, the default path is:

```
/home/USER_NAME/STM32Cube/Repository/Packs/MorseMicro/MM_IoT/1.3.3/HaLow_example
```

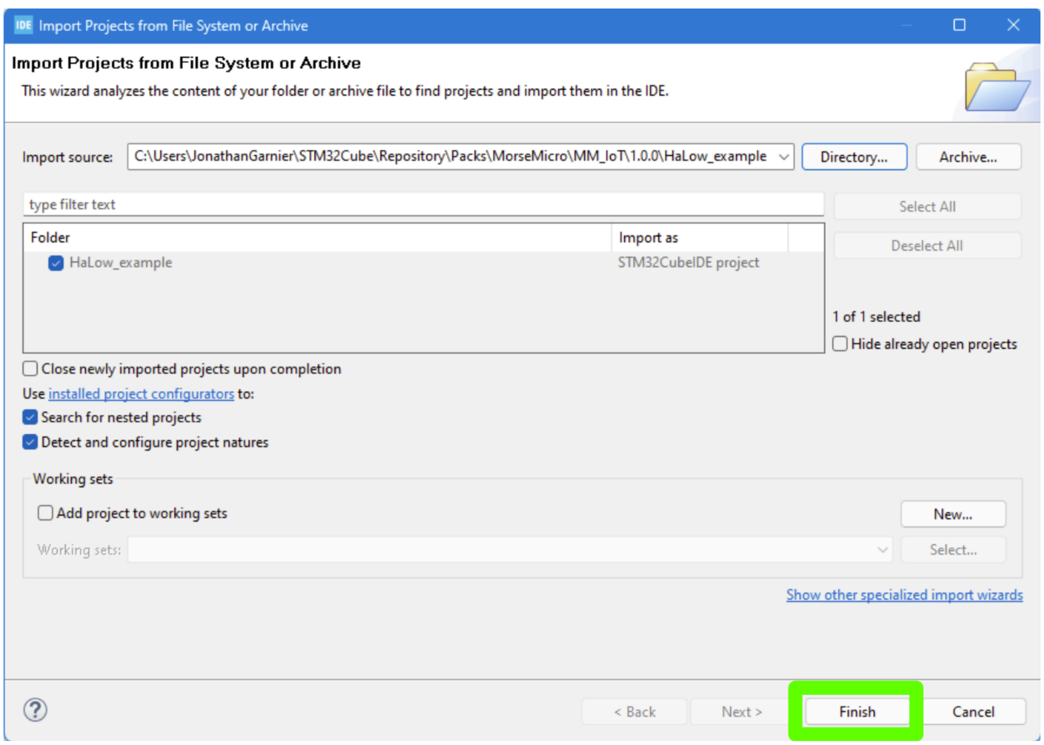
On Mac, the default path is:

```
/Users/USER_NAME/STM32Cube/Repository/Packs/MorseMicro/MM_IoT/1.3.3/HaLow_example
```

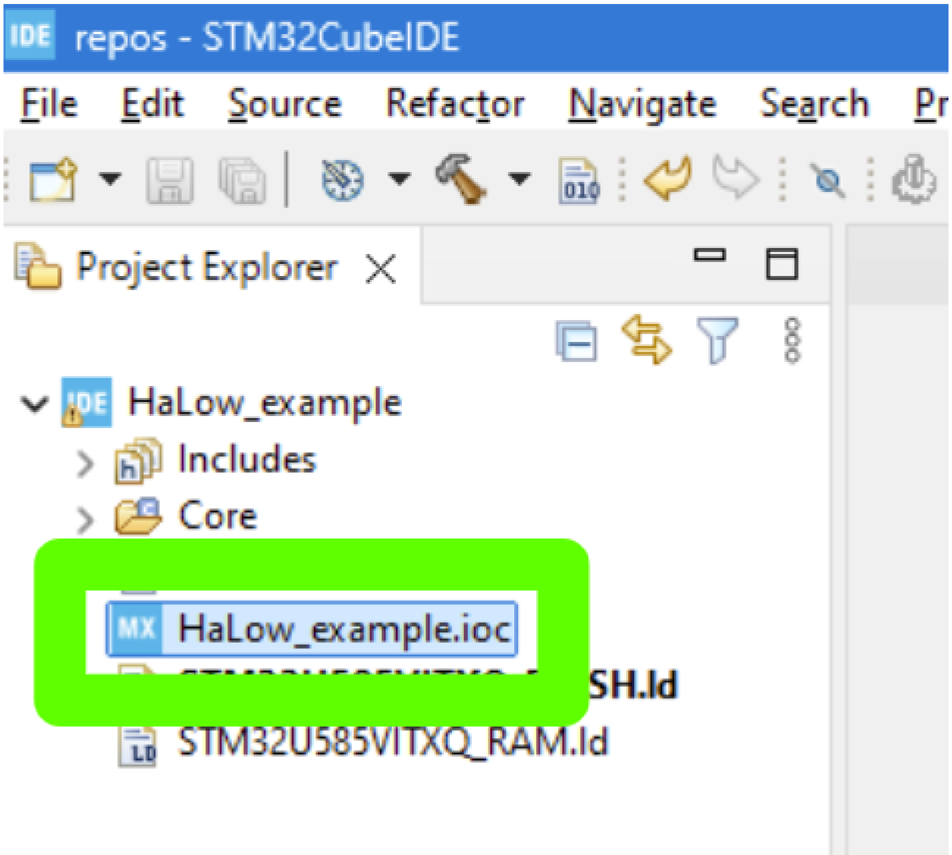
Note that the version number may change depending on which version of the CMSIS pack is downloaded.



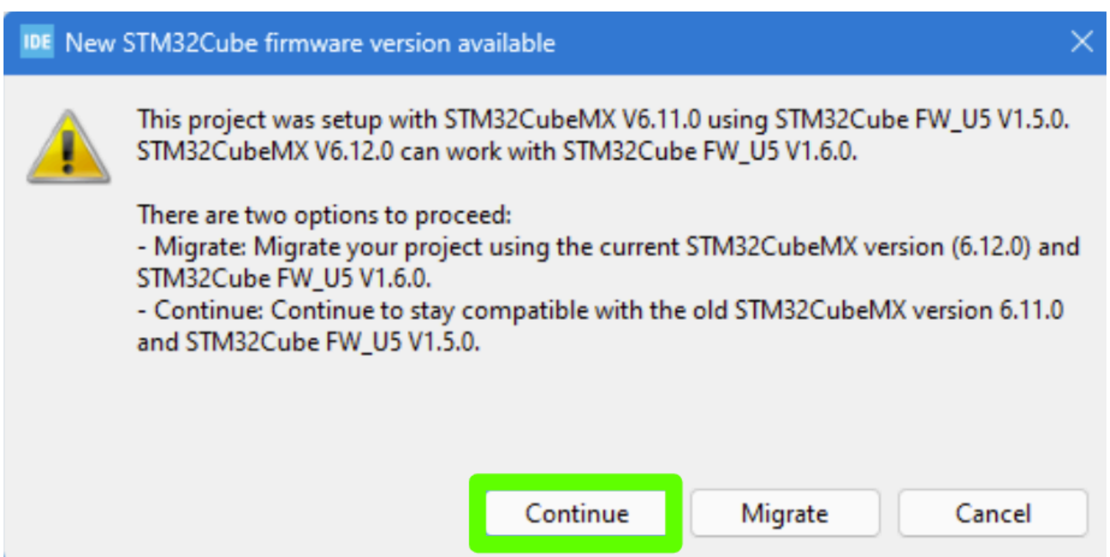
Click **Finish**:



Expand the Halow_example, and double click on the .ioc file:

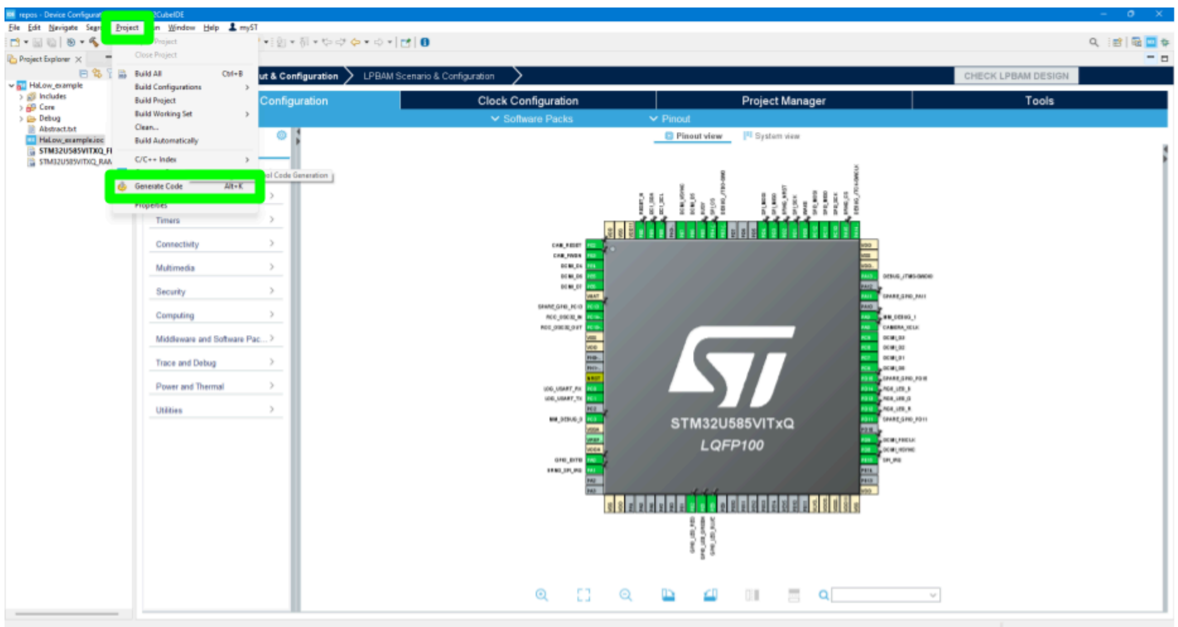


Click **Continue**.

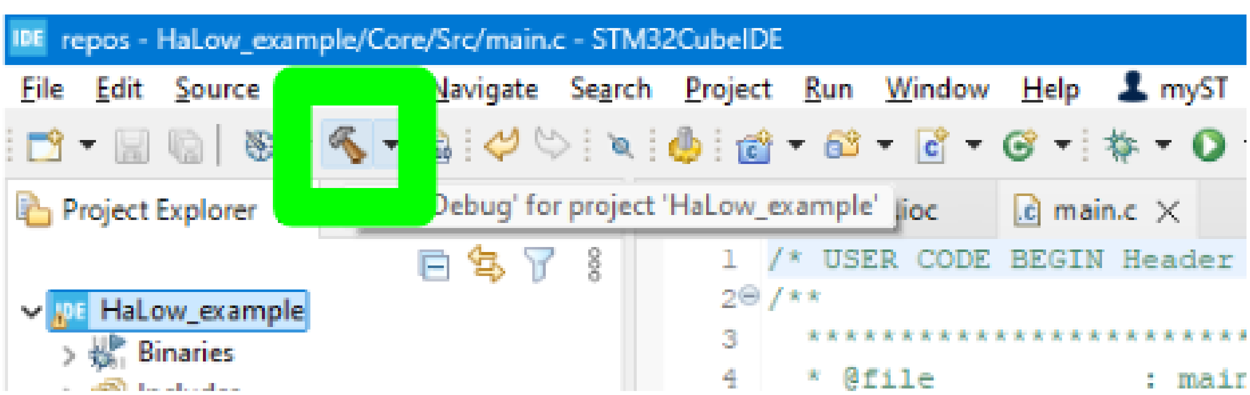


There may be additional software packs that need to be installed including STMicroelectonics.X-CUBE-FREERTOS.1.2.0, and STMicroelectronics.X-CUBE-BLE2.3.3.0. Download these items in the subsequent pop-up windows.

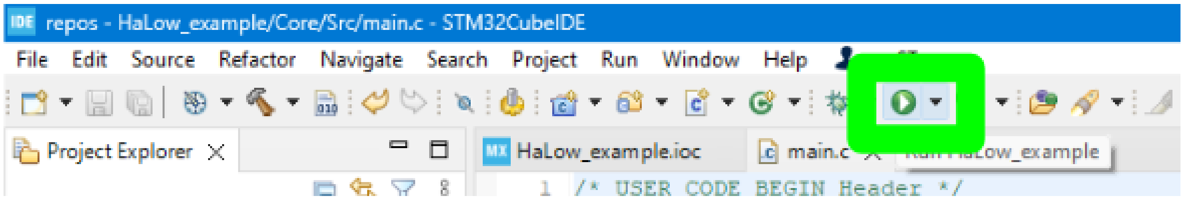
Click **Project**, **Generate Code**. This may also start a download of the STM32U5 firmware if this is not downloaded already.



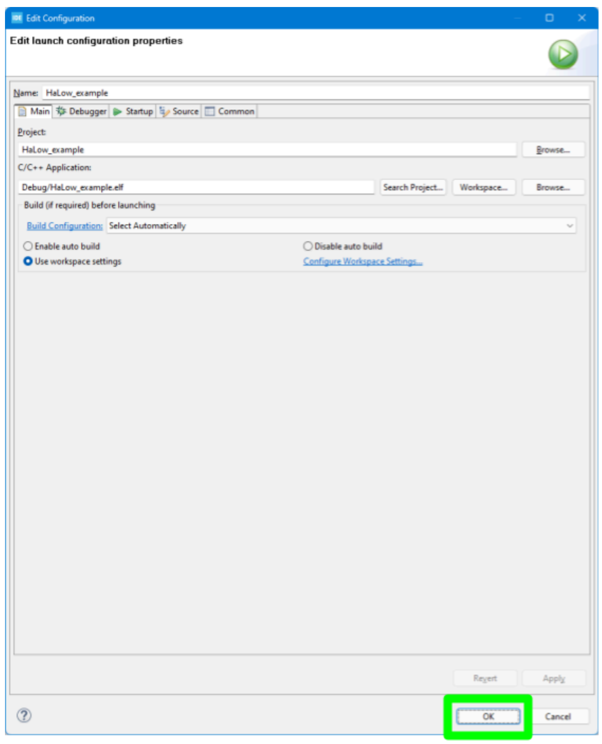
The example can then be built by clicking the hammer icon in the toolbar. If successful, there should be no errors in the Console window:



The program can now be run by clicking on the green Run icon:



This will then bring up a window for the Run Configuration. All settings can be left as default. Click **Ok**:



The example program will now build again if not built already, and then be flashed to the device.

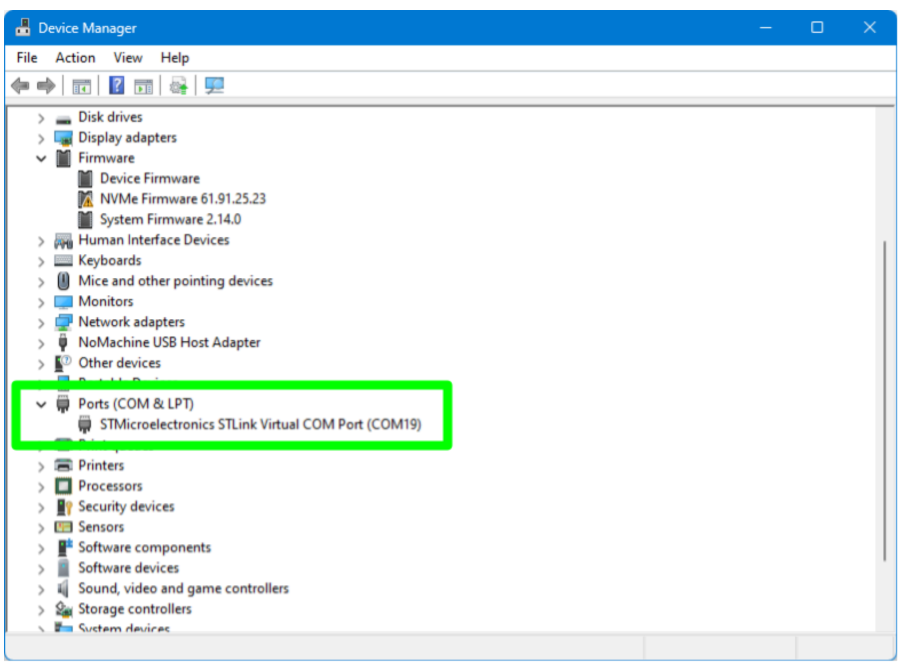
5.2.1 UART Output

The UART output can be monitored by opening a serial monitor with the following settings:

Item	Setting
Baud Rate	115200
Hardware Flow Control	None
Data Bits	8
Stop Bits	1
Parity	None

5.2.1.1 Windows

The STLINK-V3MODS will show up as a Virtual COM port on windows machines.



This COM port can then be opened with tools such as PuTTY.

```
putty -serial COM10 -sercfg 115200
```

5.2.1.2 Linux

On Linux machines, the device can be detected with the following command:

```
ls /dev/serial/by-id/usb-STMicroelectronics_STLINK-V3_*
```

And a serial terminal can be opened with tools such as minicom:

```
minicom -D /dev/serial/by-id/usb-STMicroelectronics_STLINK-V3_*
-b 115200
```

5.2.1.3 Mac OS

On Mac machines, the device can be detected with the following command:

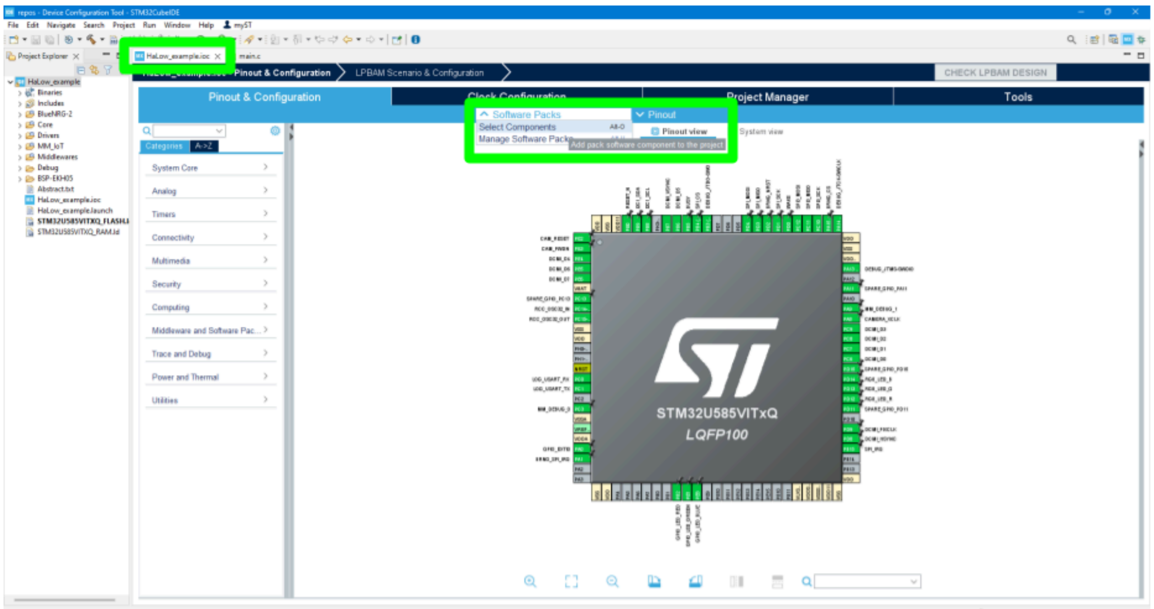
```
ls /dev/tty.*
```

And similarly, a tool such as minicom can be used to open the serial terminal:

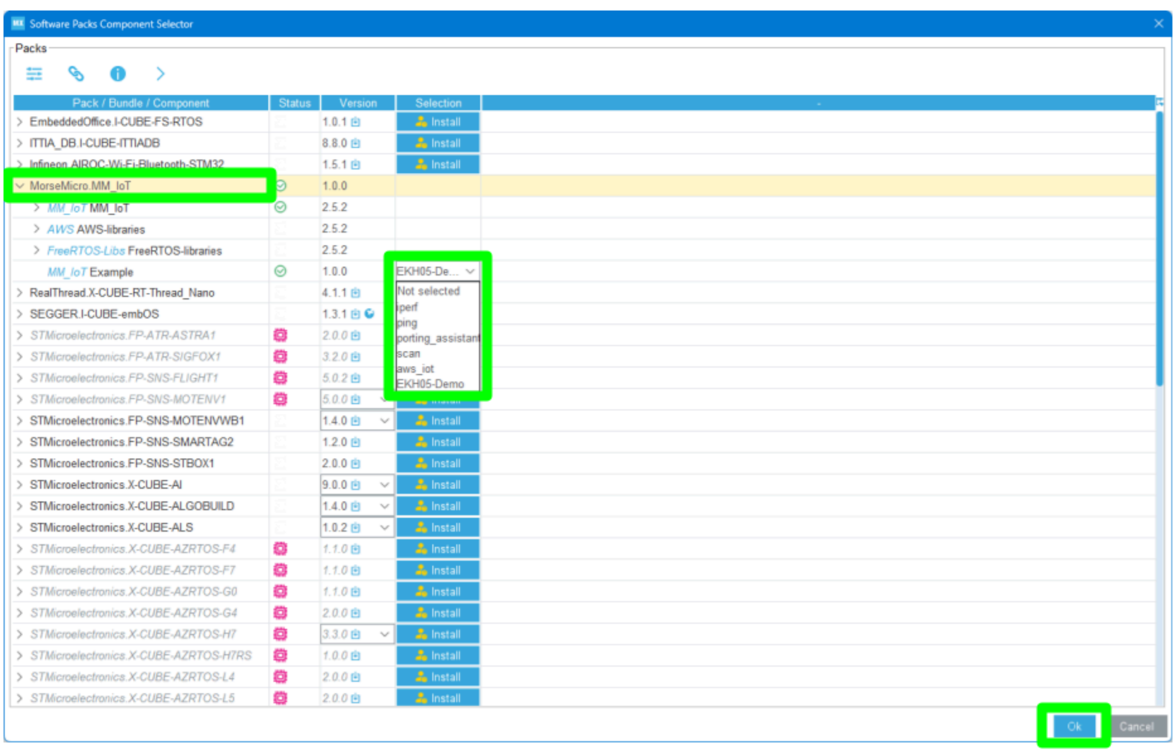
```
minicom -D /dev/tty<YOUR-DEVICE-HERE> -b 115200
```

5.3 Changing Example Application

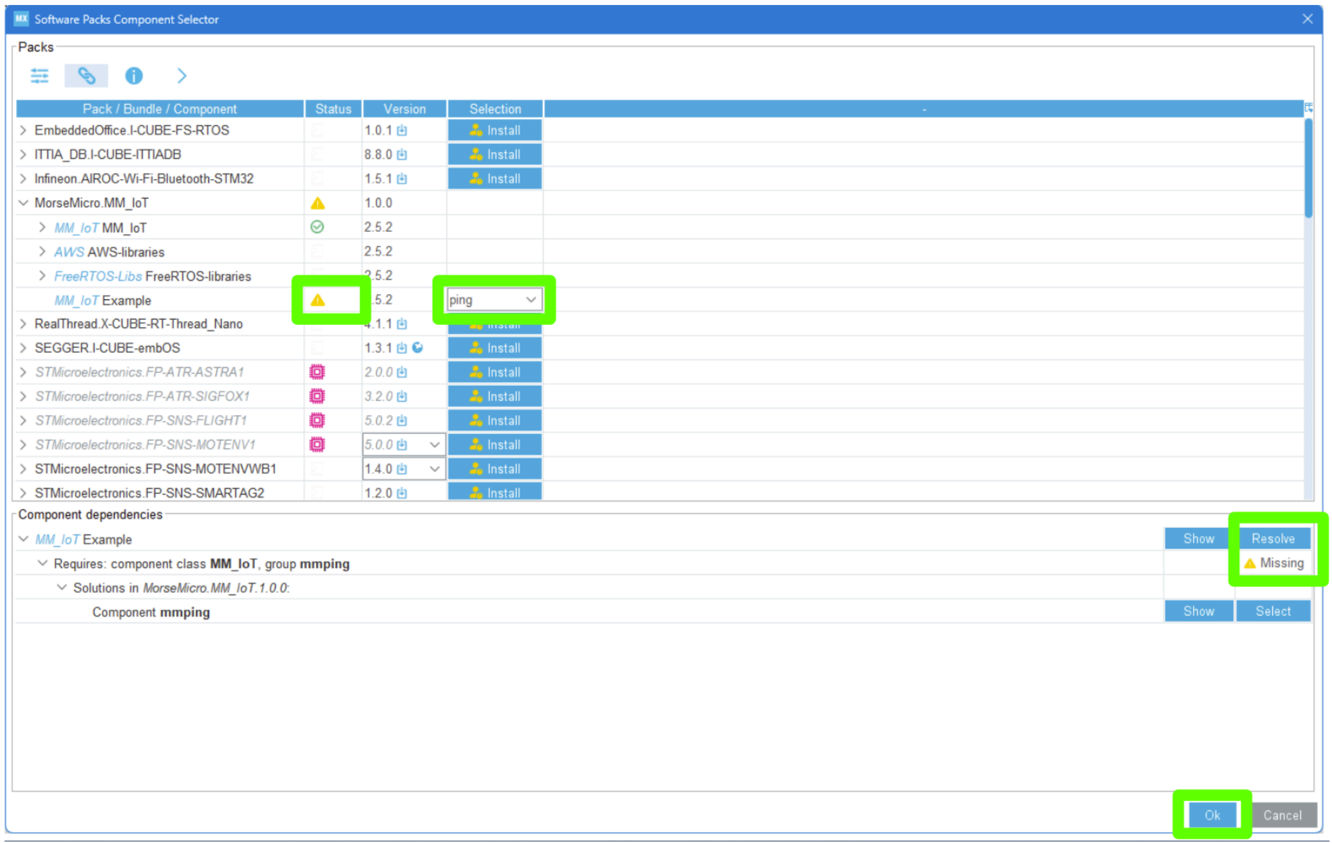
To change the example application, open the HaLow_example.ioc file and click **Select Components** from the **Software Packs** menu.



Click on the MorseMicro.MM_IoT arrow, and then select the desired application from the drop down list.



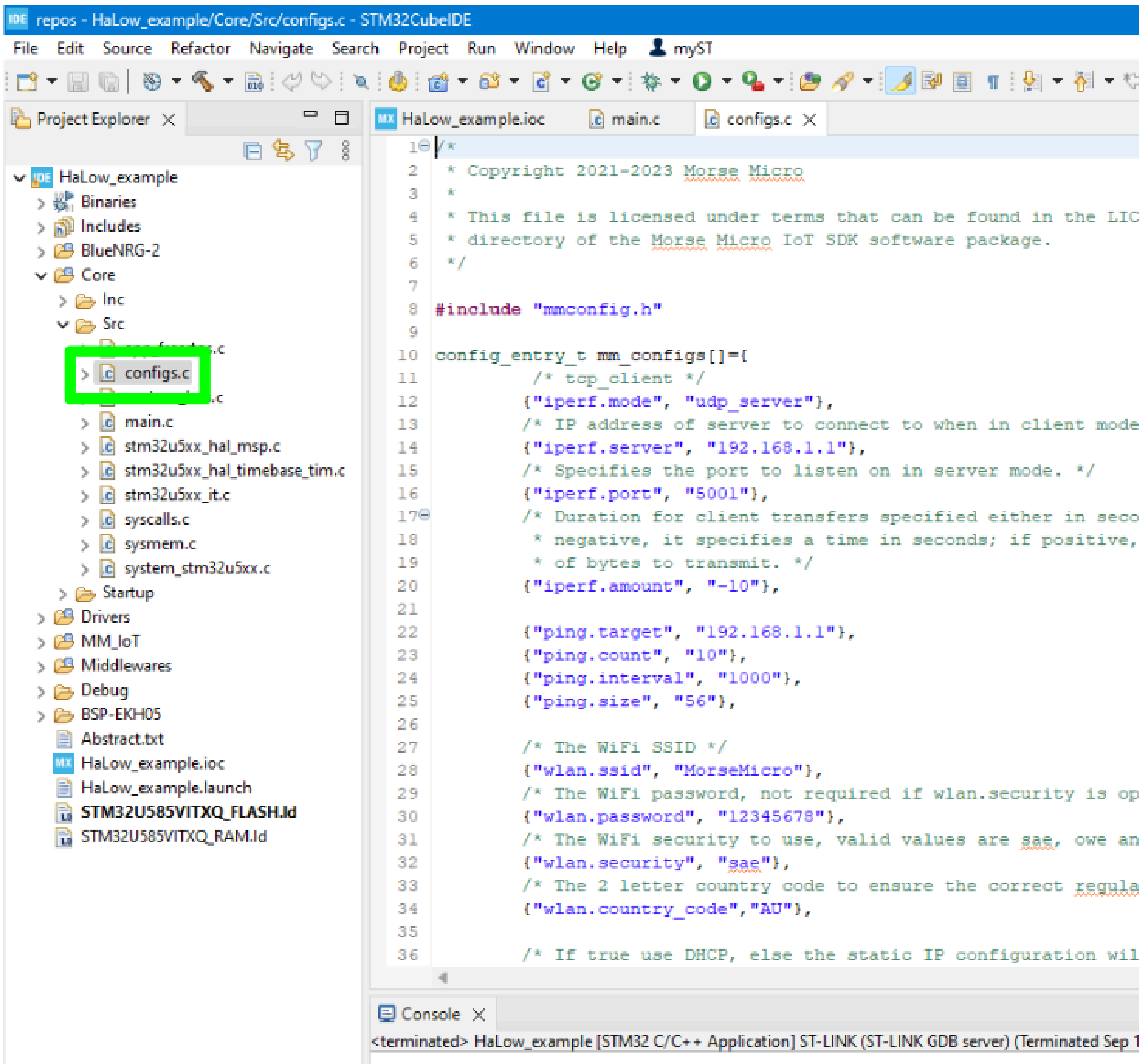
For some example applications, there may need to be an extra component added. There will be a yellow warning symbol in the **Status** column. Clicking on the yellow warning symbol will bring up a menu detailing the missing component. Click on the **Resolve** button to add the component, and then click **Ok**.



Click **Generate Code** from the **Project** menu again. The **Run** button can then be clicked again to build and run the new example.

5.4 Changing Example Configurations

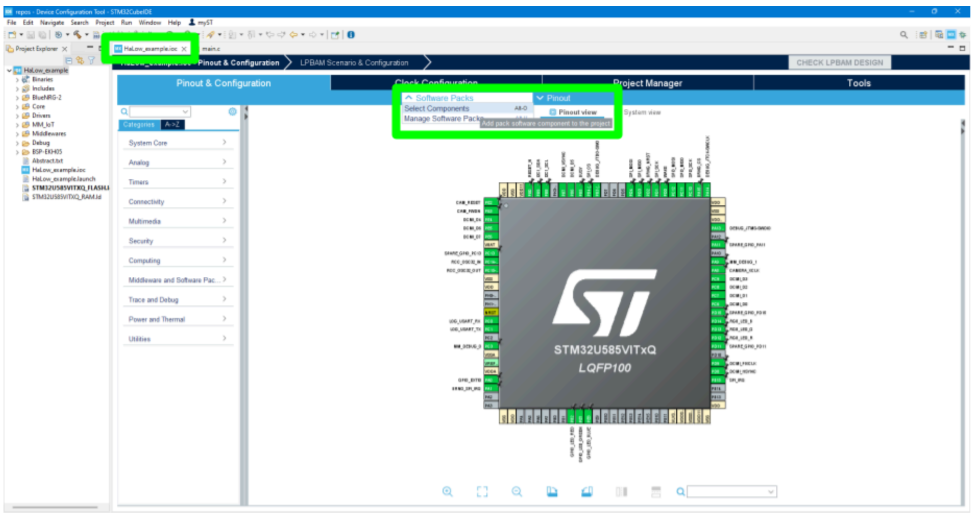
Parameters like country code, SSID, passphrase, IP addresses, ping count etc can be changed by navigating to the **Core → Src → configs.c** file. Replace the parameter and save the file. The changes will take effect upon the next build of the firmware.



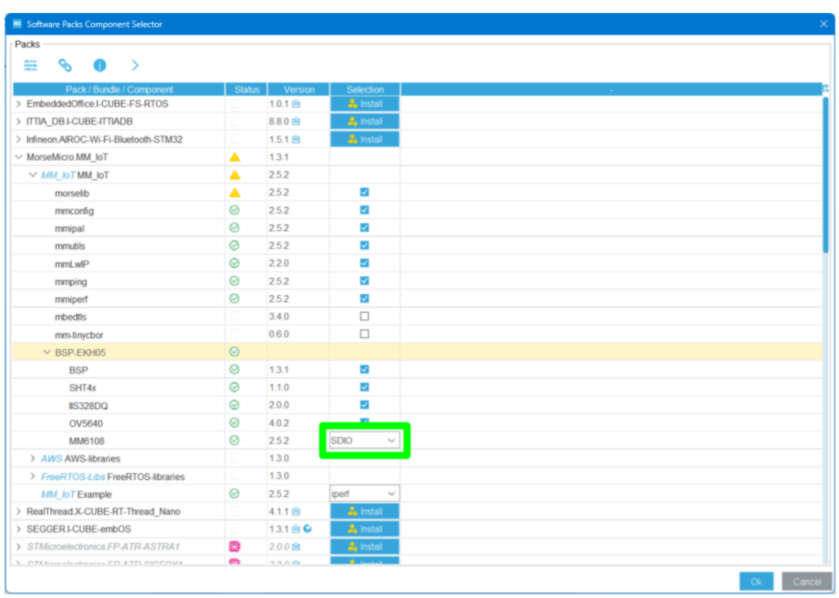
5.5 Changing Between SPI and SDIO

This section will document how to change between using SPI and SDIO as the communication protocol between the STM32U585 and the HaLow module. Note that hardware changes are also required to function properly. These changes are described in section 6.4.

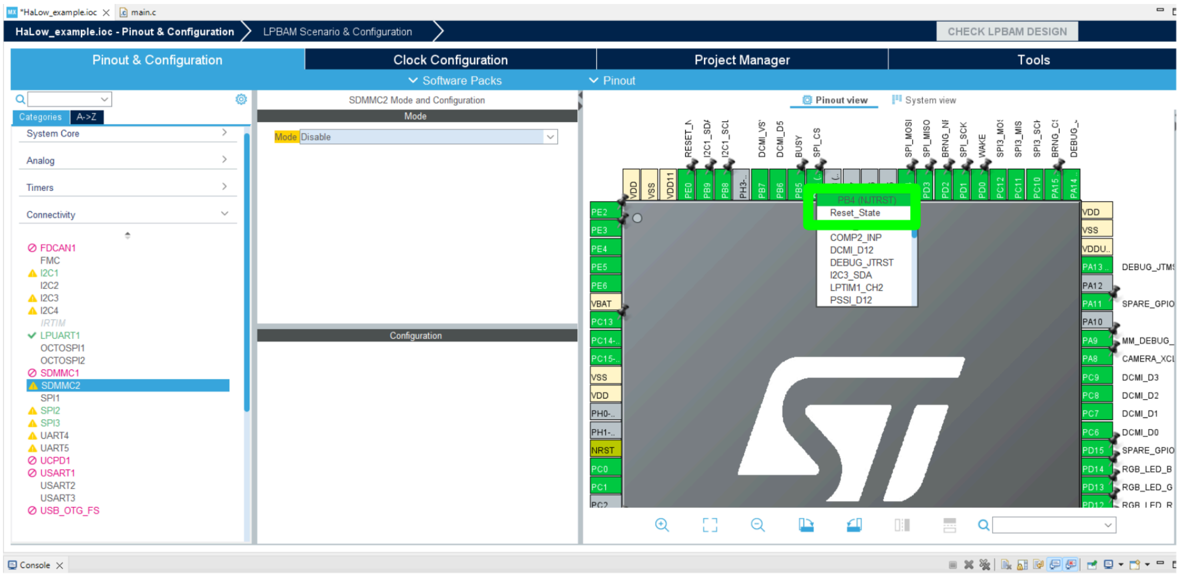
Open the HaLow_example.ioc file and click **Select Components** from the **Software Packs** menu.



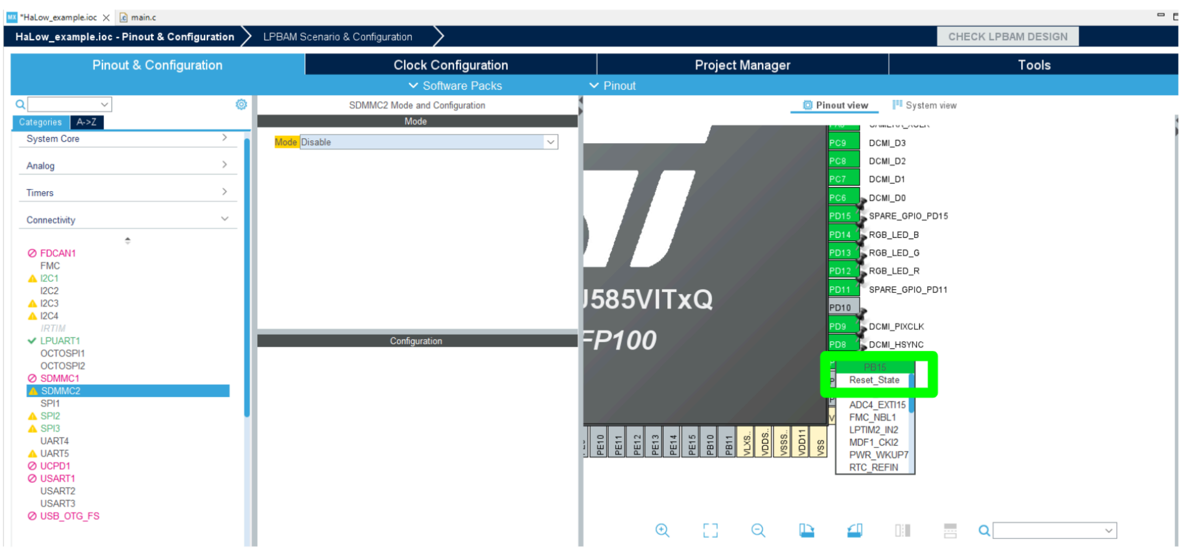
Select the **MM_IoT** and **BSP-EKH05** drop down menus, and then select the desired communication protocol from the **MM6108** drop down box. When using SDIO, there will be a yellow warning label next to morselib. This can be ignored.



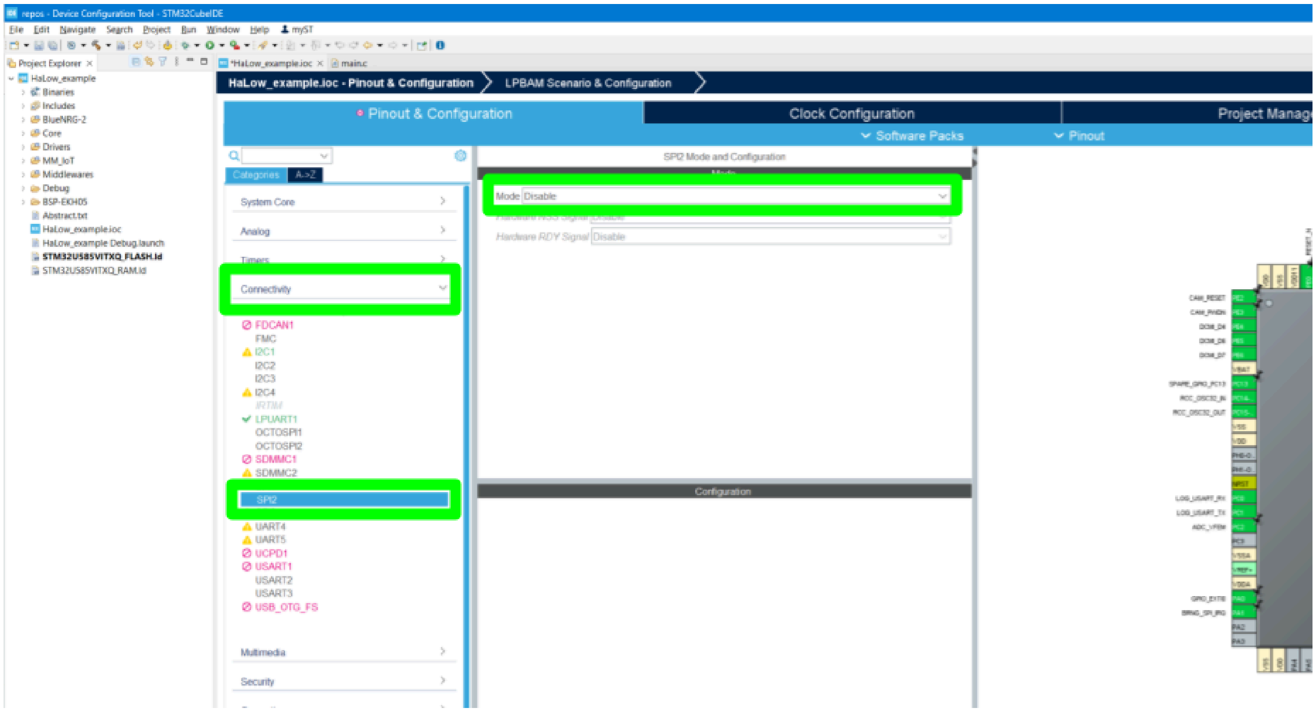
Click on the **PB4** pin and click **Reset State**.



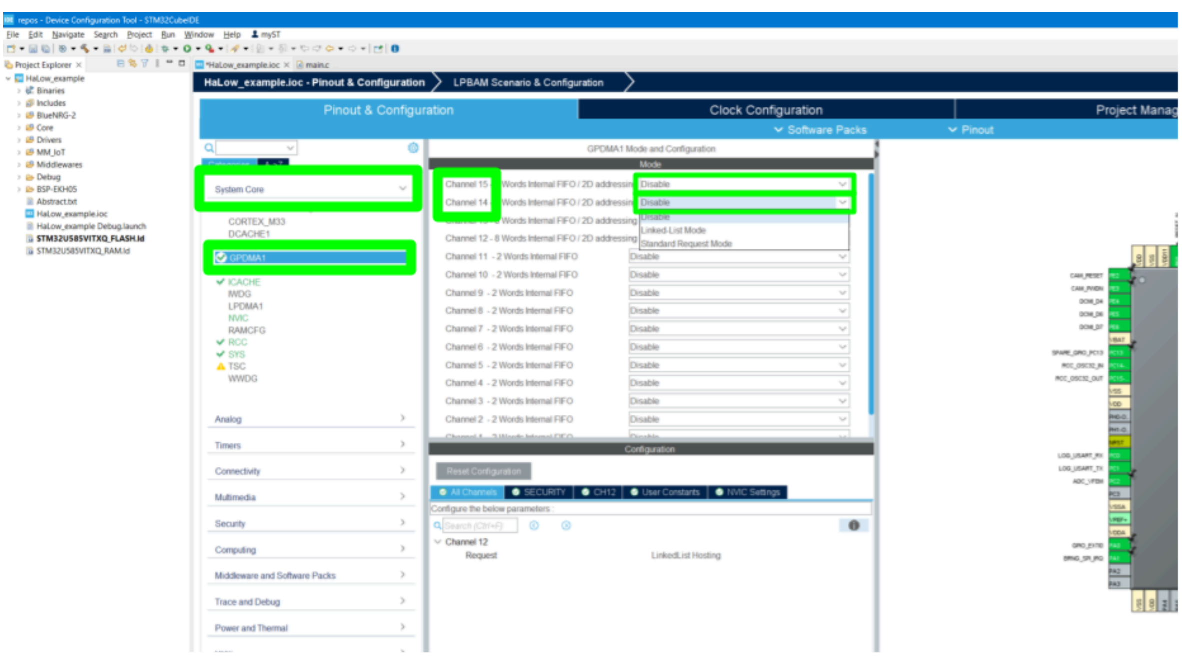
Click on the **PB15** pin and click **Reset State**.



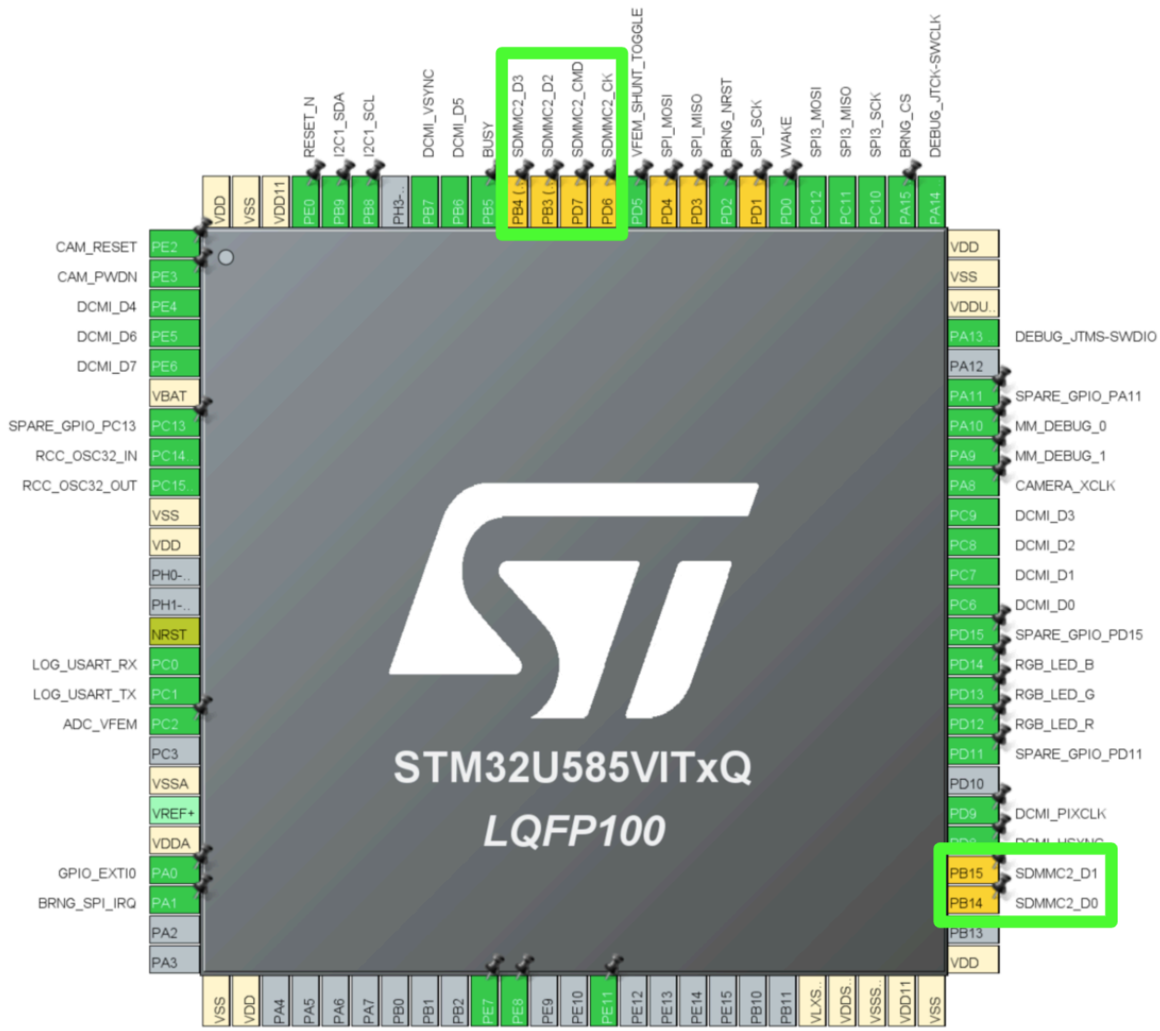
Click on **Connectivity**, and **SPI2** from the left hand panel. Select, **Disable**.



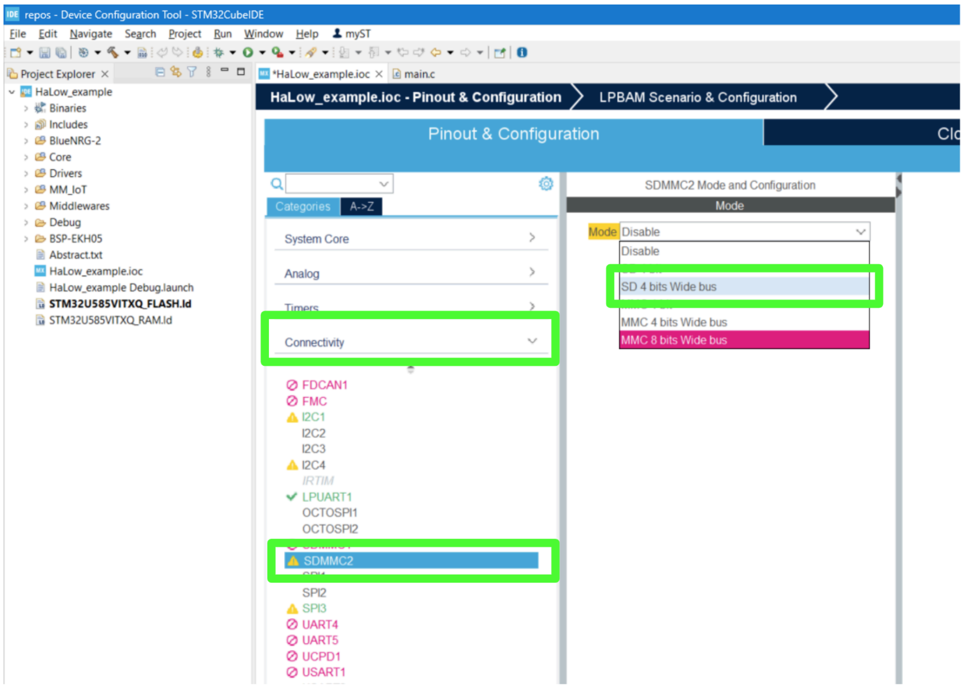
Click on **System Core**, and **GPDMA1** from the left-hand panel. Select, **Disable** for channels 14 and 15.



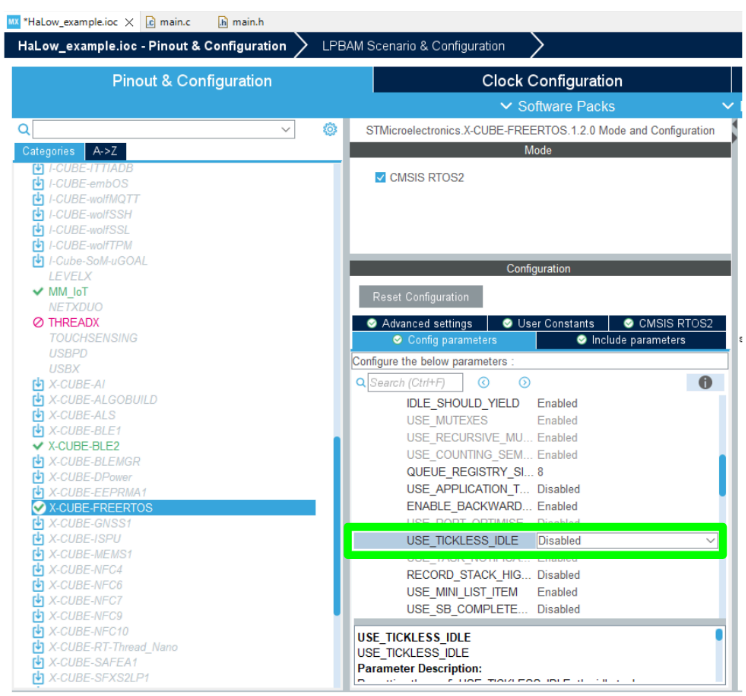
For the pins PD6, PD7, PB3, PB4, PB14 and PB15, click on them, and select the SDMMC2 pin functions.



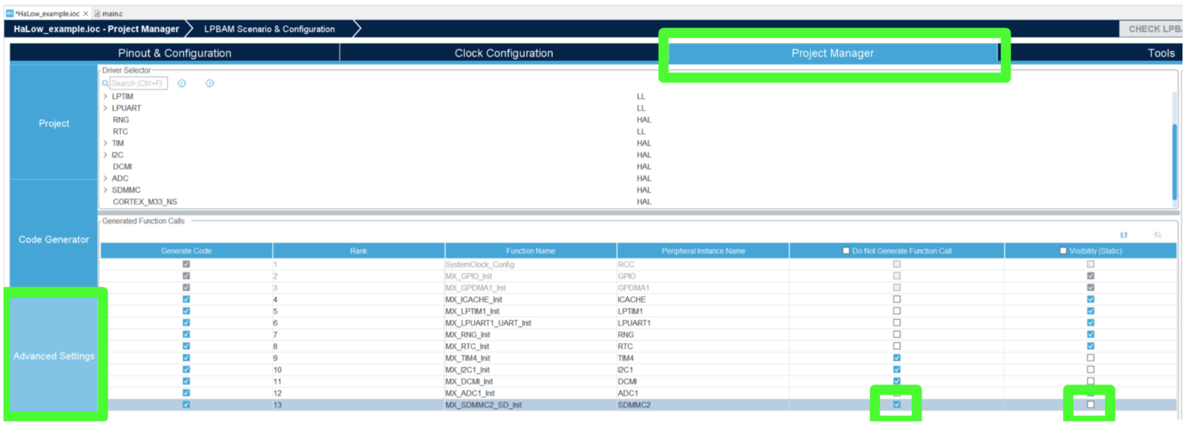
Click on **Connectivity**, and **SDMMC2** from the left-hand panel. Select, **SD 4 bit Wide bus**.



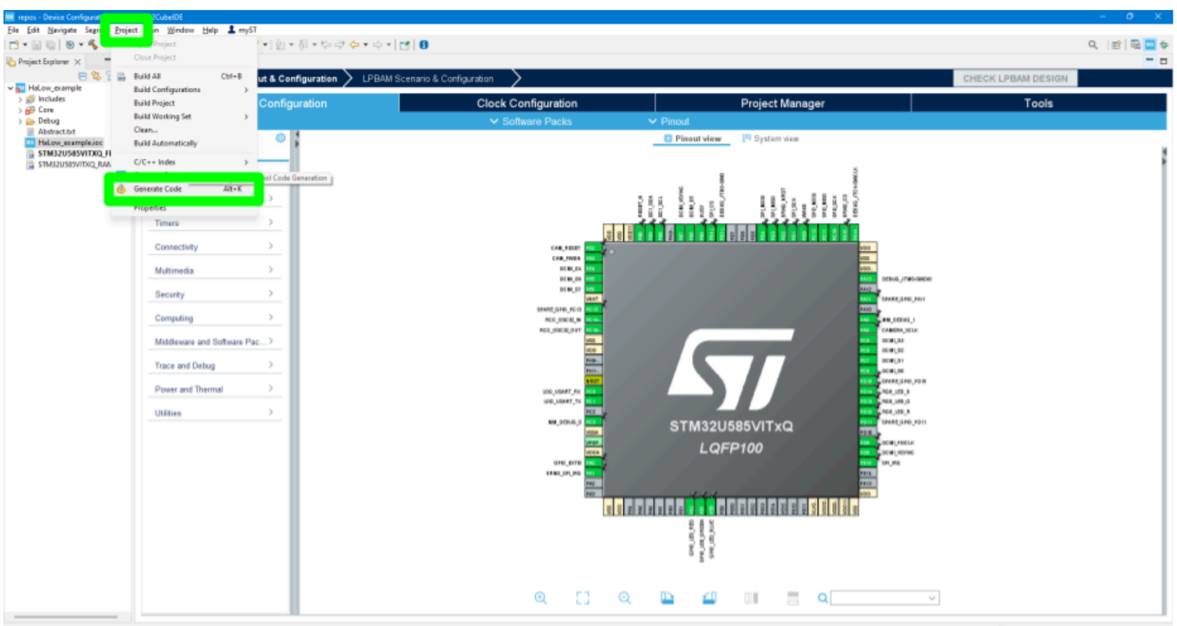
Click on **Middleware and Software Packs** from the left-hand panel, and then **X-CUBE-FREERTOS**. Set **USE_TICKLESS_IDLE** to **Disabled**.



Next, click on **Project Manager**, **Advanced Settings**. Untick the checkbox under **Visibility (Static)** and tick the checkbox under **Do Not Generate Function Call** for SDMMC2.



Click **Generate Code** from the **Project** menu to regenerate the code.

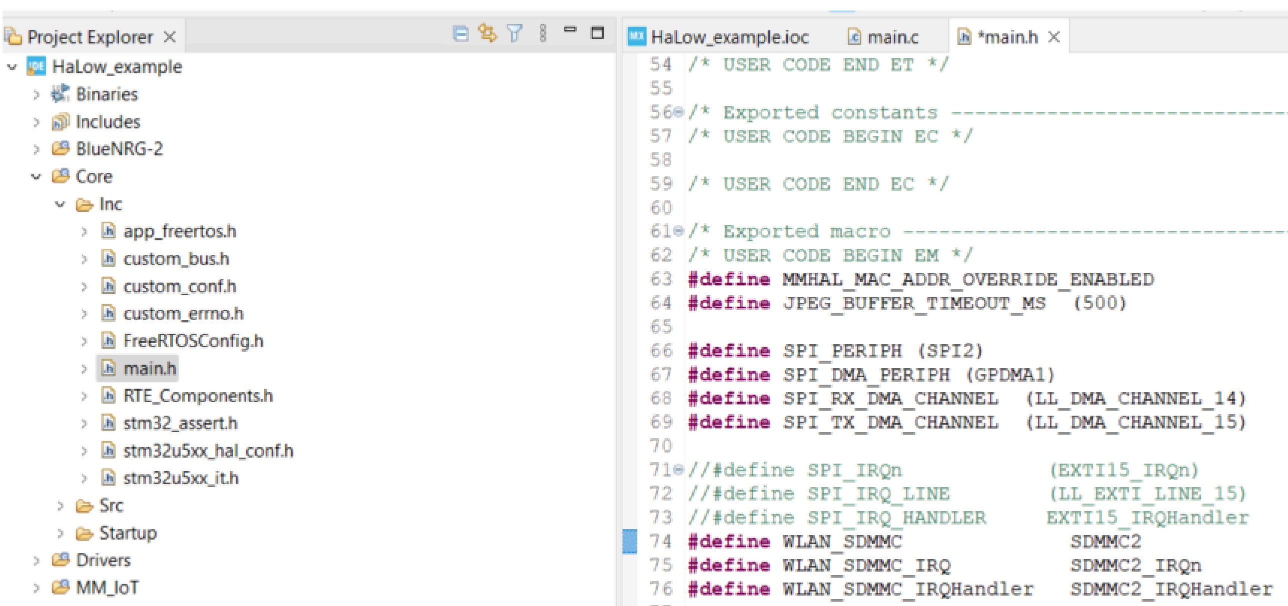


Next, open the **Core** → **Inc** → **main.h** file, and delete or comment out the following lines:

```
#define SPI_IRQn             (EXTI15_IRQn)
#define SPI_IRQ_LINE         (LL_EXTI_LINE_15)
#define SPI_IRQ_HANDLER      EXTI15_IRQHandler
```

Replace them with:

```
#define WLAN_SDMMC              SDMMC2
#define WLAN_SDMMC_IRQ          SDMMC2_IRQn
#define WLAN_SDMMC_IRQHandler   SDMMC2_IRQHandler
```

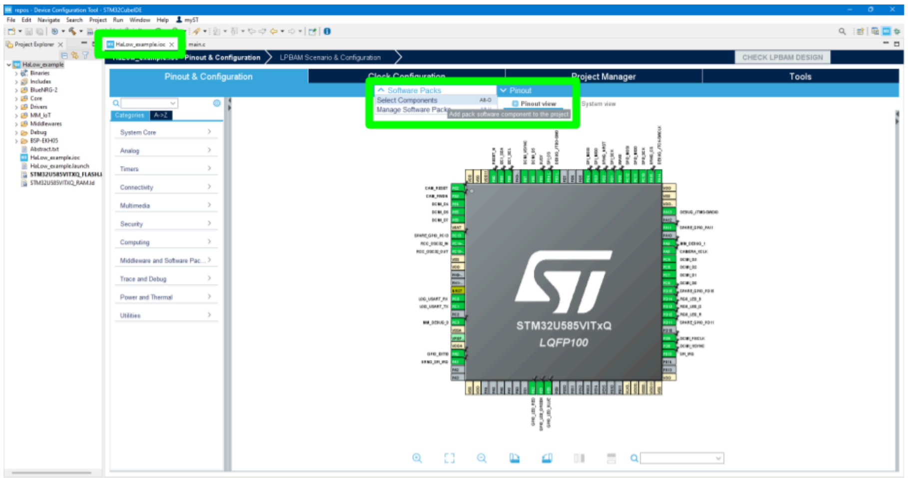


The **Run** button can now be clicked again to build and run the example using SDIO.

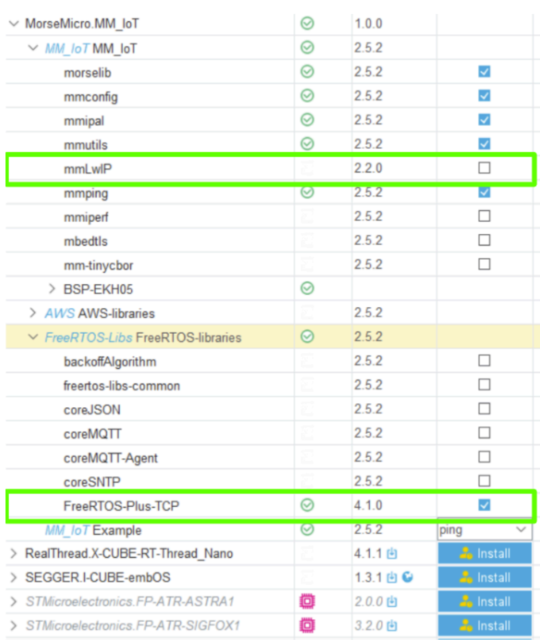
5.6 Changing Network Stacks

This section will document how to change between using LwIP and FreeRTOS+TCP as the network stack. For almost all situations, LwIP should be used. FreeRTOS+TCP should only be used when validating the best low-power performance.

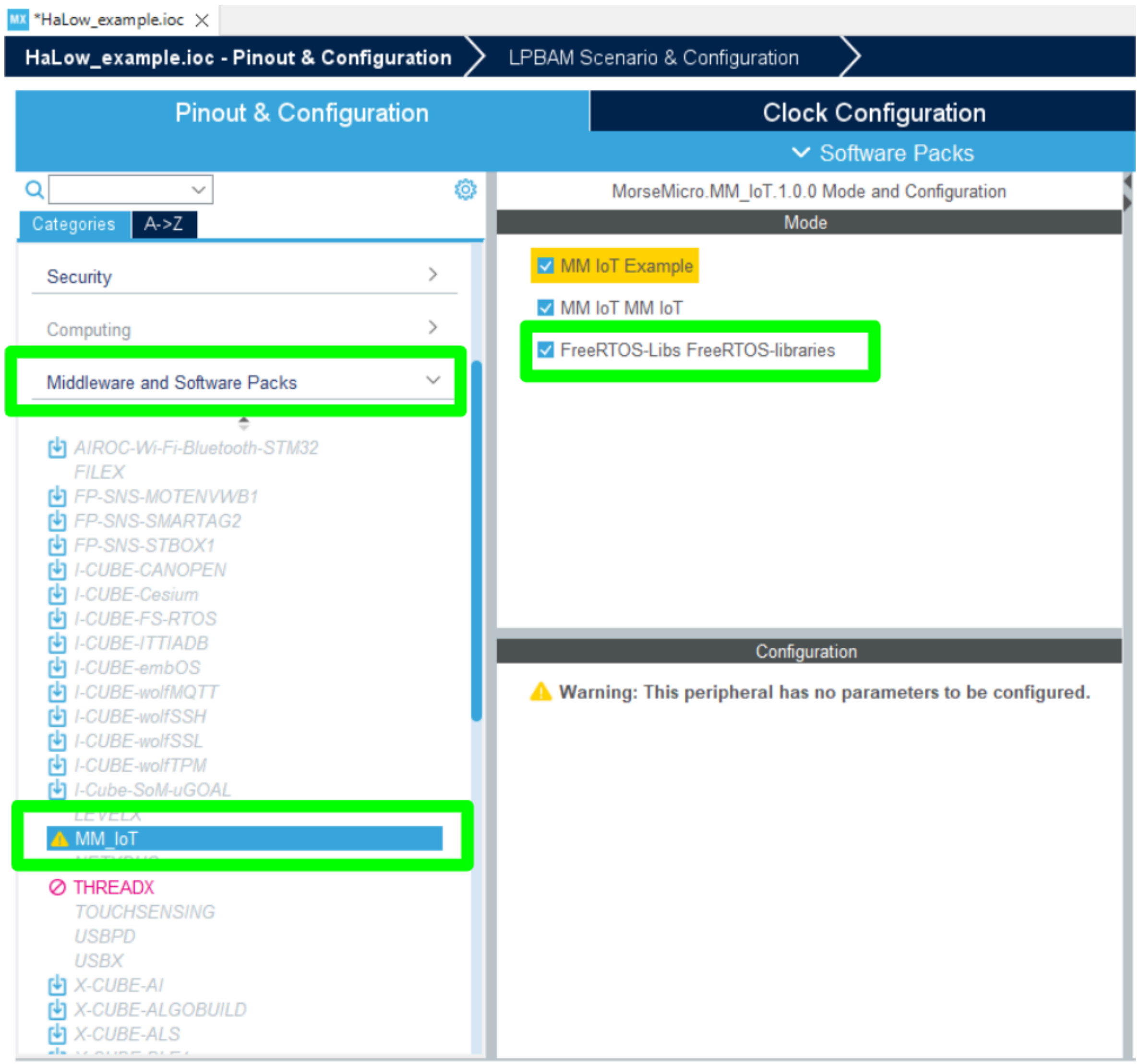
Open the HaLow_example.ioc file and click **Select Components** from the **Software Packs** menu.



Select the **MM_IoT** and **FreeRTOS-Libs** drop down menus. Select the desired network stack and ensure the other is deselected. There should be all green ticks if there are no issues.



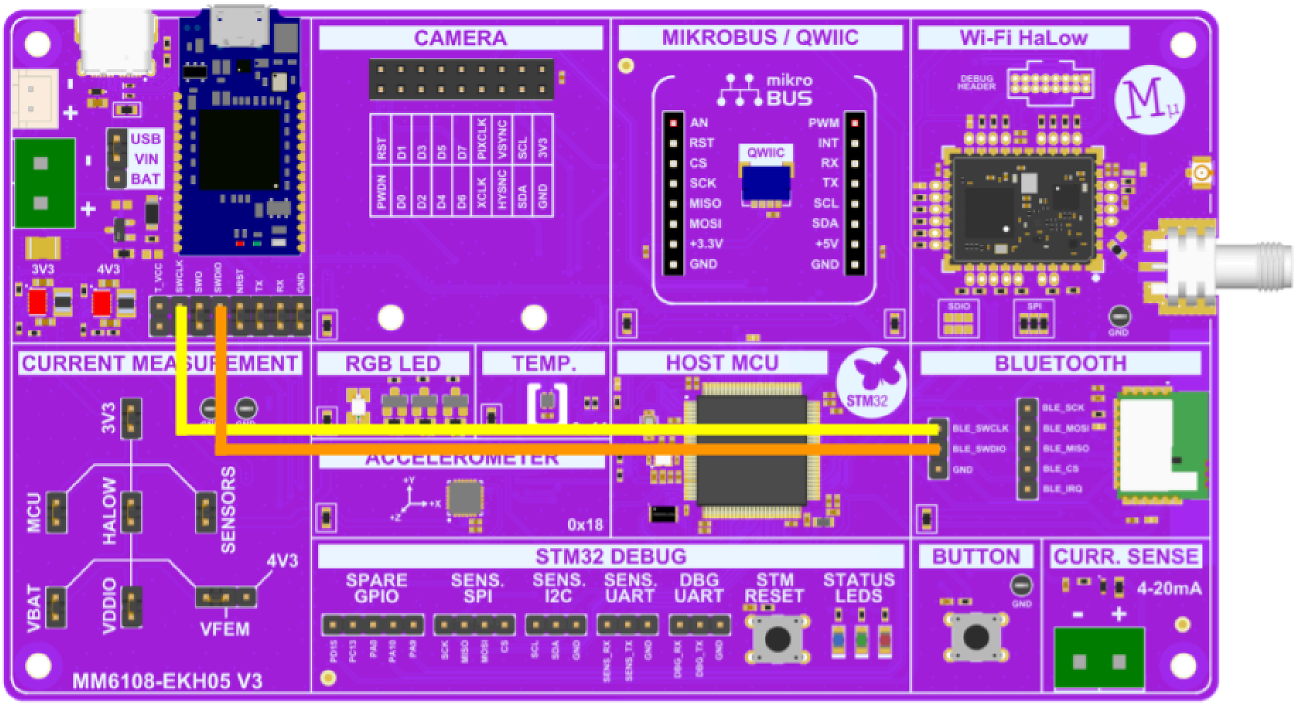
Next, click the **Middleware and Software Packs** menu from the .ioc file. Select **MM_IoT** and ensure the **FreeRTOS-Libs FreeeRTOS-libraries** checkbox is ticked.



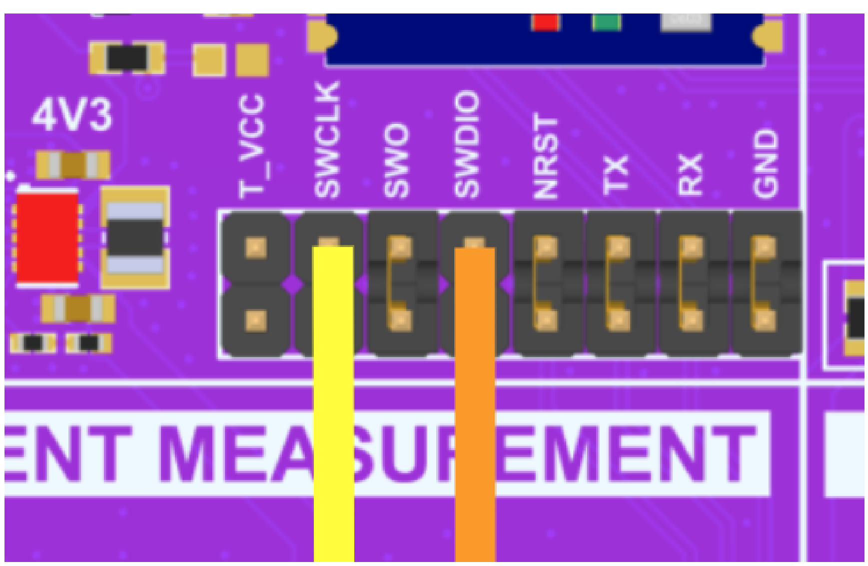
The code will need to be regenerated, and the firmware rebuilt and uploaded to the device for the changes to take effect.

5.7 Updating BLUENRG-M2SP Module Firmware

The most straightforward way to flash the firmware on the BLUENRG-M2SP module is to use the on board STLINK V3. The SWCLK and SWDIO jumpers can be removed, and wires connecting to the BLE SWD headers can be attached as shown below.



Ensure the wires are attached to the top pins of the header as these are connected to the STLINK V3.

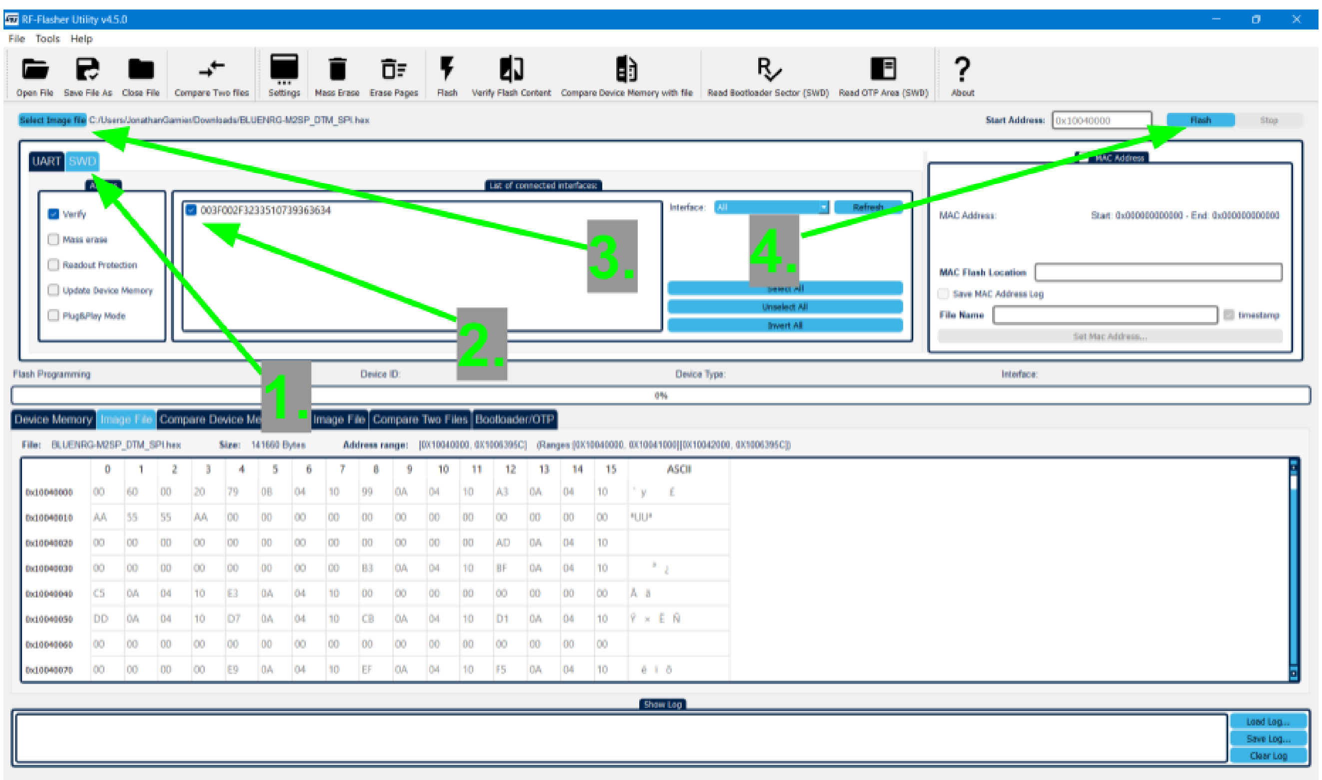


Before doing this, the BLE RESET line must be held high by the STM32U585. Please ensure that pin PD2 is configured as an output, and is set to high. The BLUENRG-M2SP module will be held in reset if this is not done, and will not be able to be programmed.

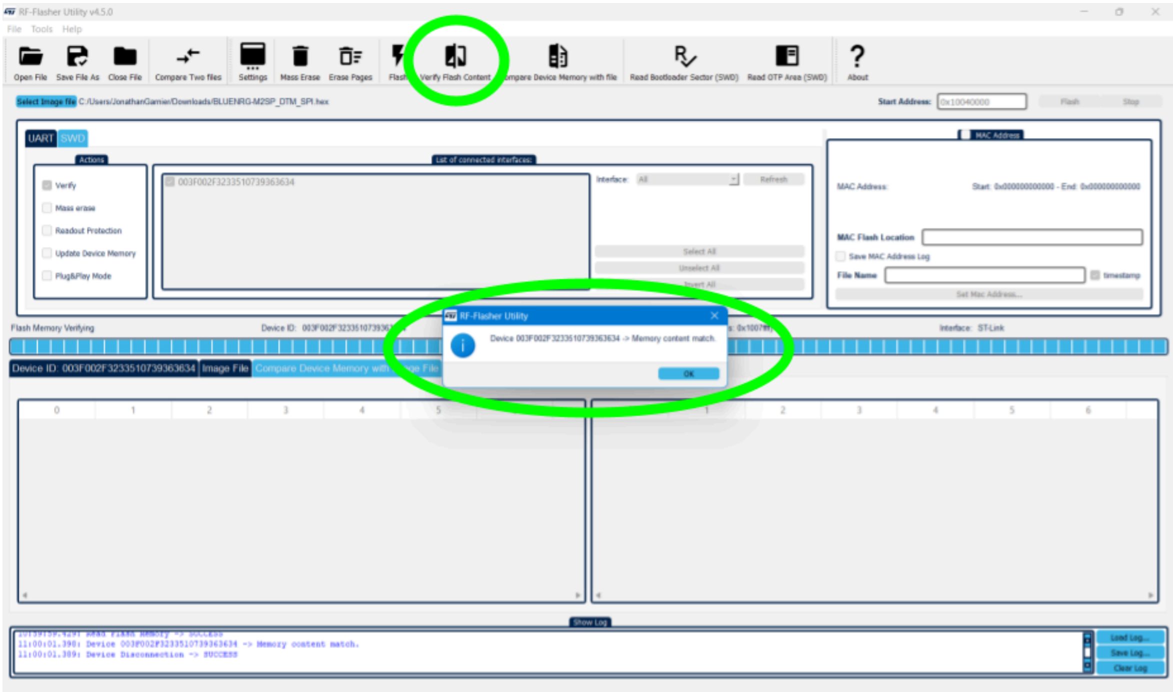
The next step is to download the RF-Flasher Utility from ST. This can be found here - https://www.st.com/en/embedded-software/stsw-bnrgflasher.html

Once installed, open the program and do the following:

1. Click on the **SWD** tab.
2. Select your connected STLINK device
3. Click **Select Image File**. Select the firmware image file to load.
4. Click **Flash**.



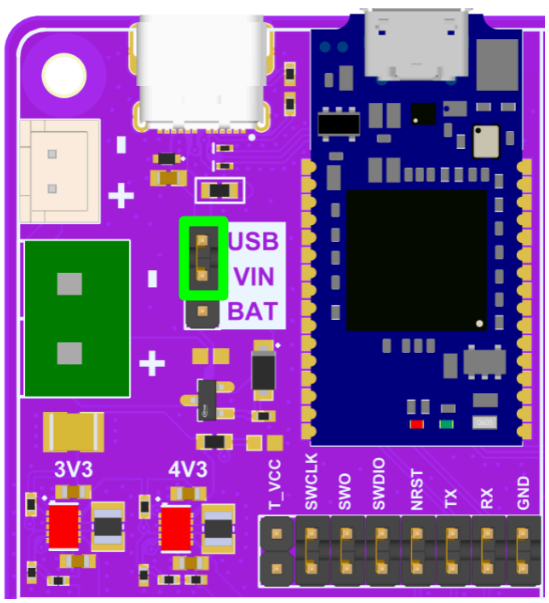
To verify this operation was successful, click the **Verify Flash Content** button. A popup window will appear notifying if the memory content matches the file or not.



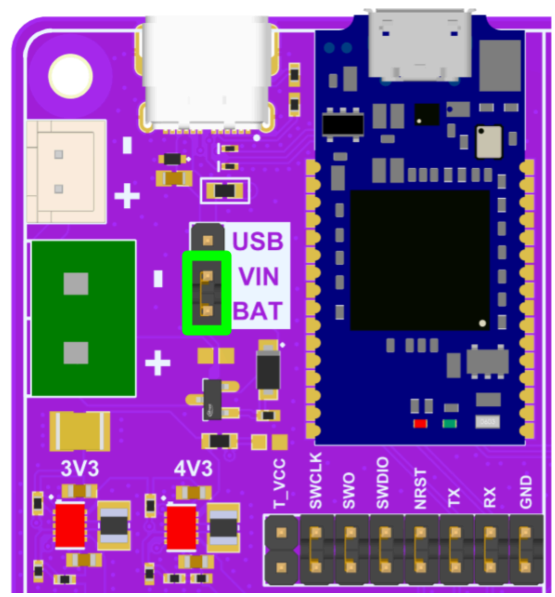
6 Hardware Layout and Configuration

6.1 Power Selection

The voltage supply input to the buck-boost regulators can come from multiple sources. If the jumper is placed on the USB and VIN header, the supply can come from either the USB-C or STLINK Micro USB connectors. If the jumper is placed on the BAT and VIN header, the supply will come from the battery connections.



Powered From Either USB-C or STLink Micro USB Connectors

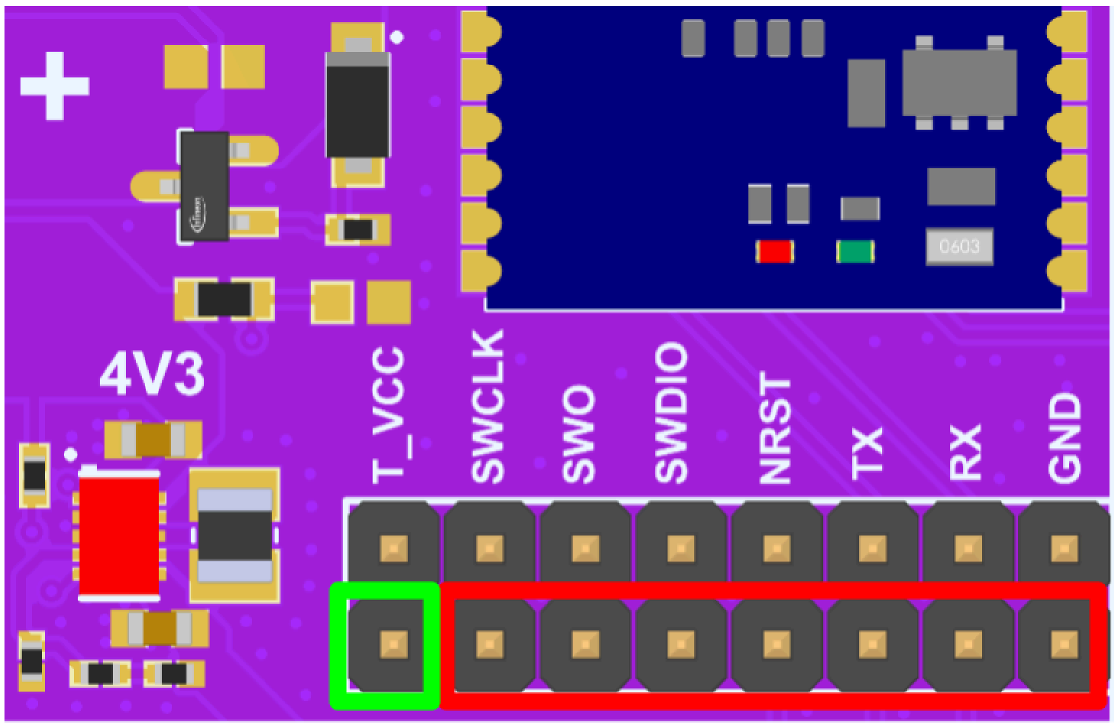


Powered From Either JST Battery Connector or Screw Terminal Battery Connector

6.2 Using an External Debugger/Programmer

The STLINK V3 MODS is a modular in-circuit debugger and programmer for STM32 and STM8 microcontrollers. It programs the device through a Single Wire Debug (SWD) interface, and connects to a UART interface on the STM32U585 microcontroller in the one cable.

Developers can opt to use their own external debug tool if desired. The jumpers below the module can be removed, and the bottom side of the headers can be attached to with an external debug tool to connect to the relevant pins on the STM32U585 microcontroller.



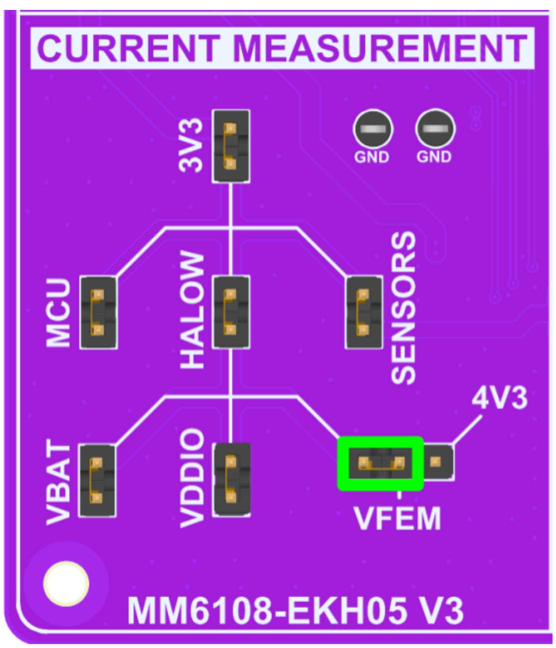
Connects to 3.3V. Attach target voltage pin of external debugger here.

This side connects to STM32U585 Microcontroller. Connect external debug tool to this side of header.

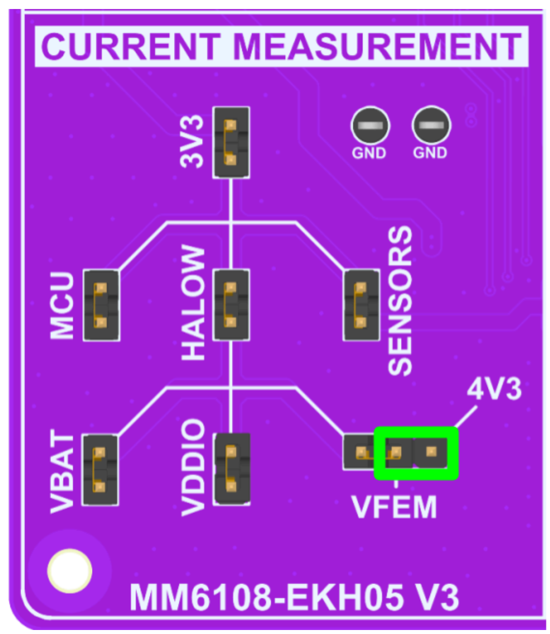
6.3 Changing VFEM Voltage

The voltage supplied to the Front End Module (FEM) on the HaLow module can be changed between 3.3V and 4.3V. Setting to 4.3V will increase the power output from the Power Amplifier, and increase the maximum range. This will also increase power consumption of the VFEM rail when actively transmitting. Note that different Board Configuration Files (BCF) are required for each of the settings. In the Morse Micro examples, an ADC from the STM32U585 is connected to VFEM and will automatically select the correct BCF. For user applications, please take care to ensure the correct BCF is being used.

To switch between the two, the jumper can be placed on the leftmost header and VFEM (3.3V), or the rightmost header and VFEM (4.3V). Power cycle on change.



VFEM set to 3.3V.

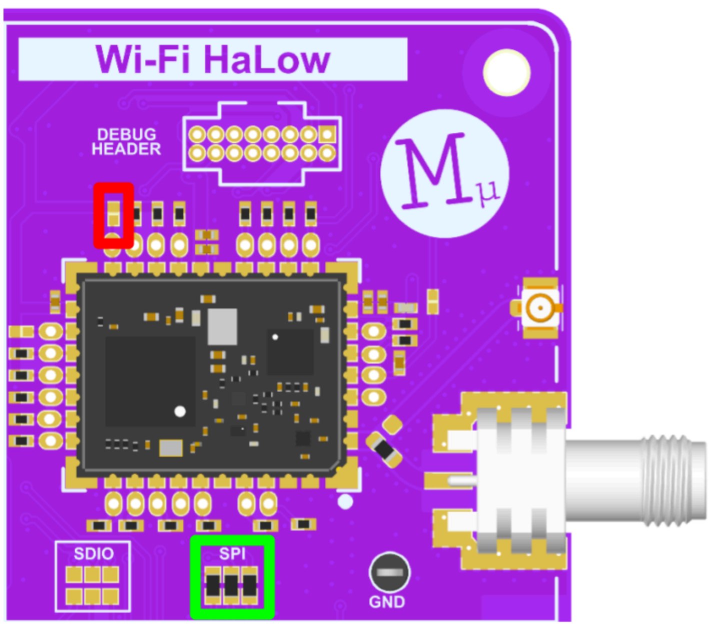


VFEM set to 4.3V (default)

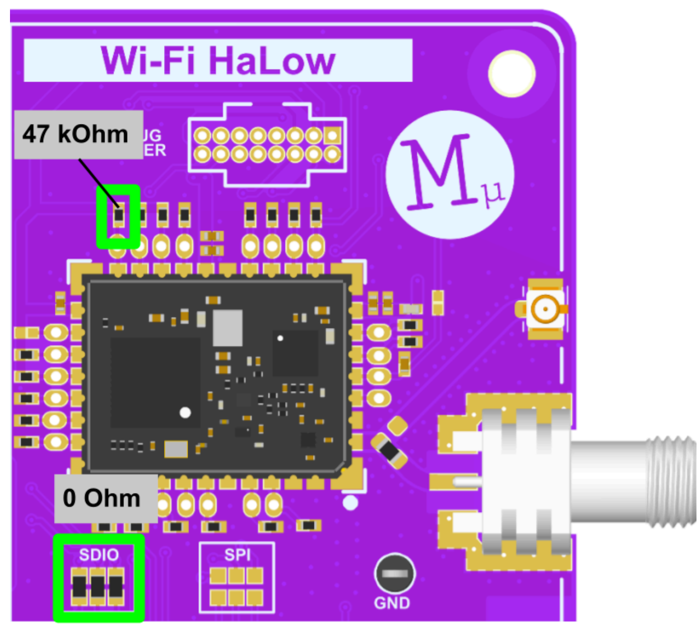
6.4 Switching Between SDIO and SPI

The MM6108-EKH05 gives users the option to interface the STM32U585 Microcontroller with the HaLow module either through SDIO or SPI. SPI is the default and has been optimised for lowest power consumption. If using SDIO however, the 47 kOhm pull-up resistor must be populated as pointed out by the arrow below. For SPI, this pull up resistor will cause extra leakage current on the VDDIO rail and should be removed.

A different pin configuration in the software is also needed when changing between the two. Section 5.6 details how to configure this.



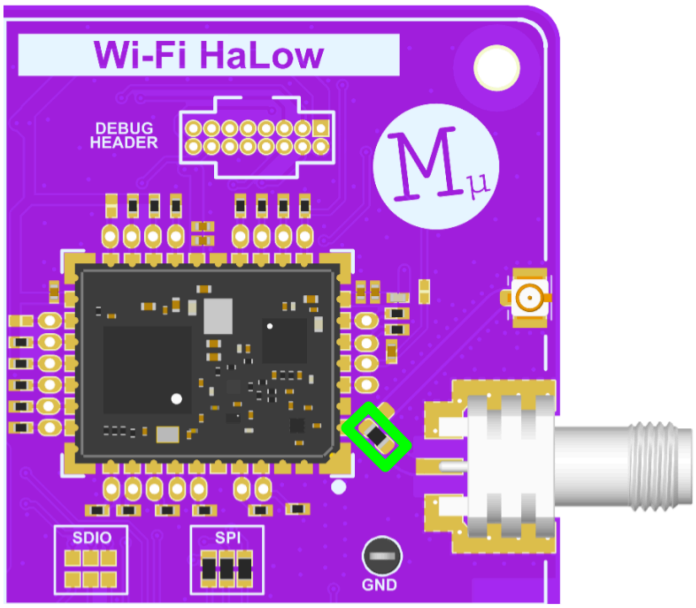
SPI Configuration



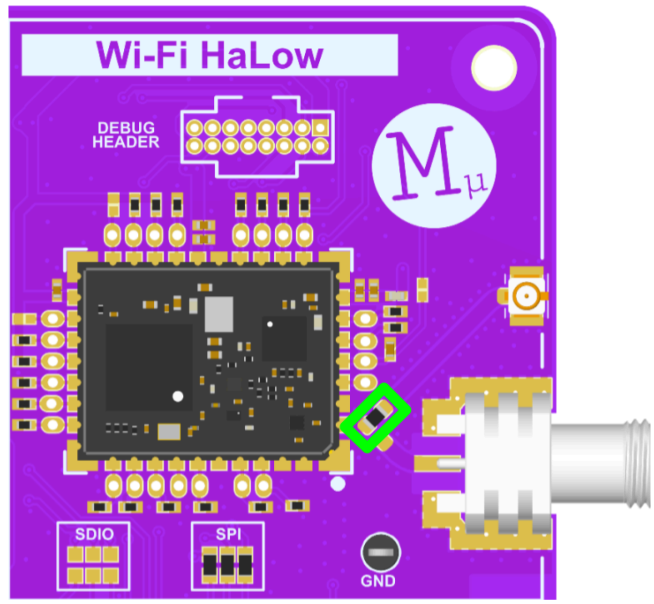
SDIO Configuration

6.5 Switching Between SMA and U.FL Connector

The user has the option to use either a SMA or U.FL connector for the HaLow antenna. The two can be switched between by changing the orientation of a 0 ohm resistor.



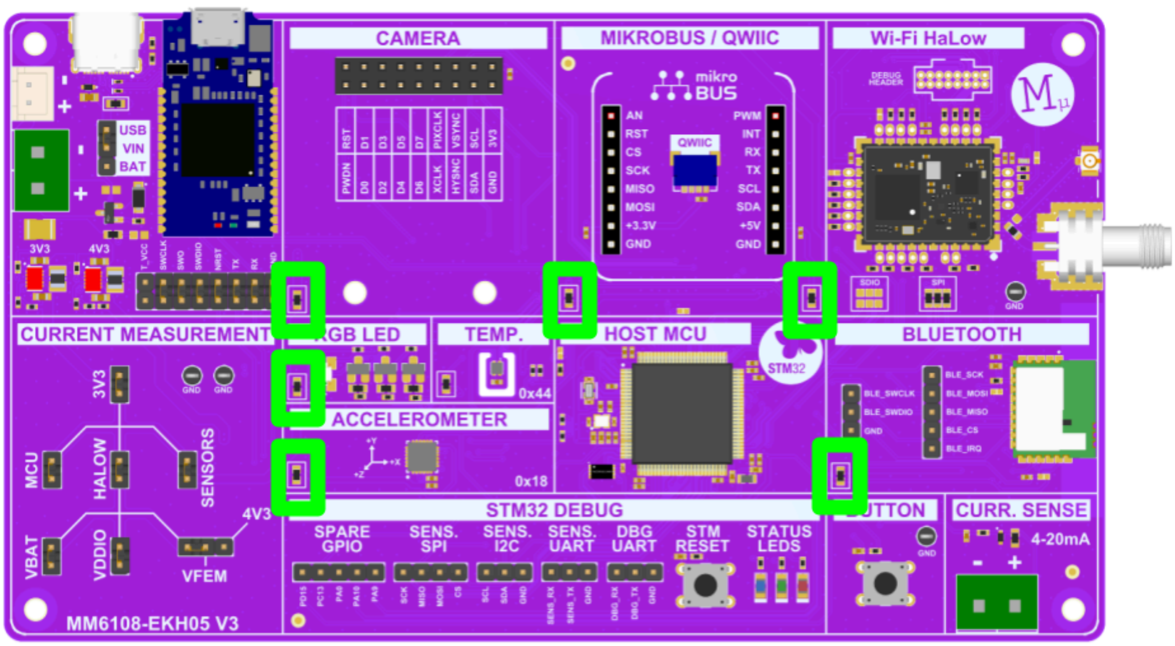
SMA Antenna



U.FL Antenna

6.6 Disconnecting Sensors

The sensor blocks on the MM6108-EKH05 feature a 0 ohm resistor that supplies power from the SENSOR header in the power tree to each individual sensor. These sensors include the temperature and humidity sensor, accelerometer, RGB LED, camera header, mikroBUS connector (3.3V on the left, 5V on the right), QWIIC connector, and the BLE module. By removing these 0 ohm resistors, the power can be cut off to the corresponding sensor individually. All these resistors are populated by default.

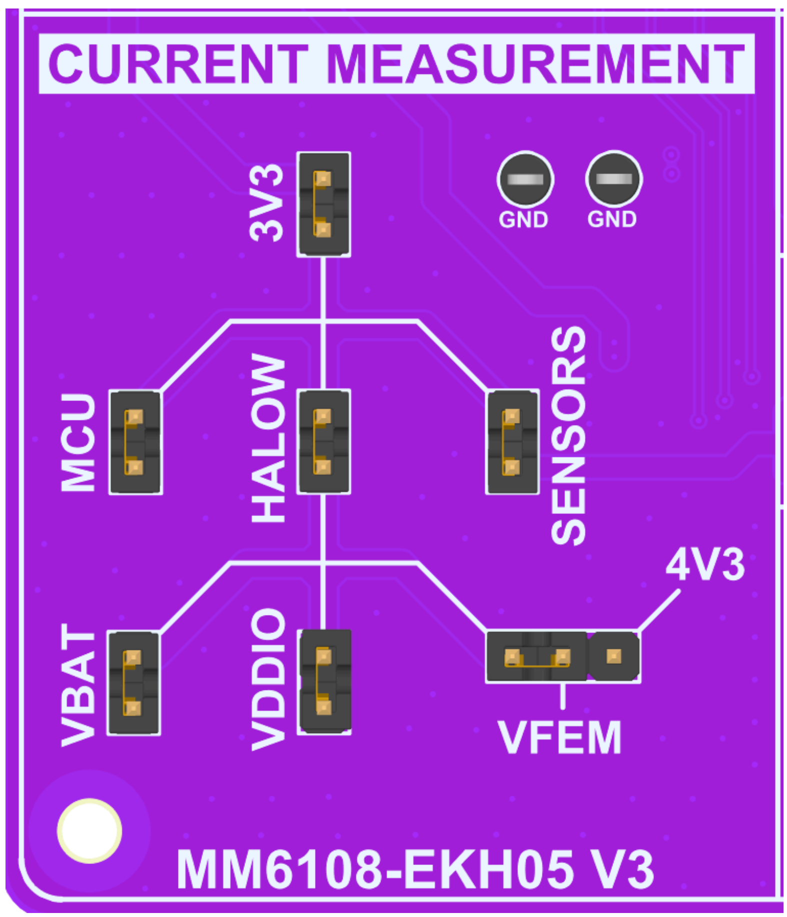


7 Power Consumption Measurements

7.1 Power Consumption Measurement Points

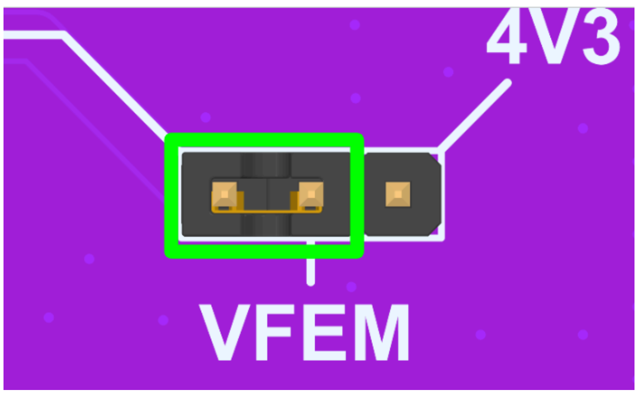
7.1.1 General Structure

The MM6108-EKH05 is designed to make key power consumption measurements straightforward. Its power distribution follows a tree structure, with each sub-power section branching from the main supply. For instance, to measure the total current draw on the 3.3V rail, an ammeter can be connected across the 3V3 header. If the user wishes to measure only the power consumption of the MCU for example, the ammeter can instead be placed across the MCU header.

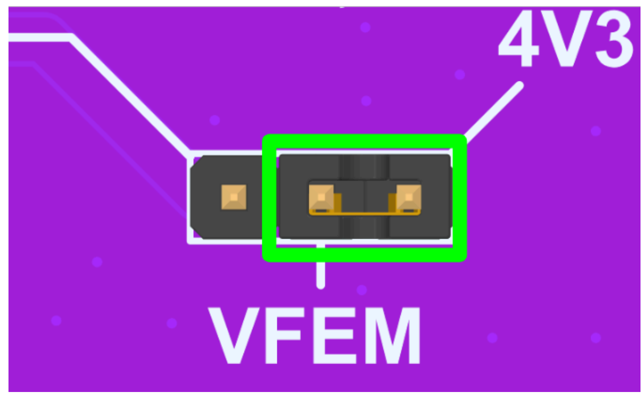


7.1.2 HaLow and VFEM

For measuring HaLow power consumption, all HaLow power rails can be assessed together from the HALOW header when VFEM is set to 3.3V. Each individual HaLow supply can also be measured separately from the corresponding header. Keep in mind that VFEM can be configured to either 3.3V or 4.3V. When set to 4.3V, VFEM is disconnected from the rest of the power tree, meaning its current consumption is not included when measuring from the 3V3 or HALOW headers. To measure power consumption at 4.3V, a separate measurement is required.



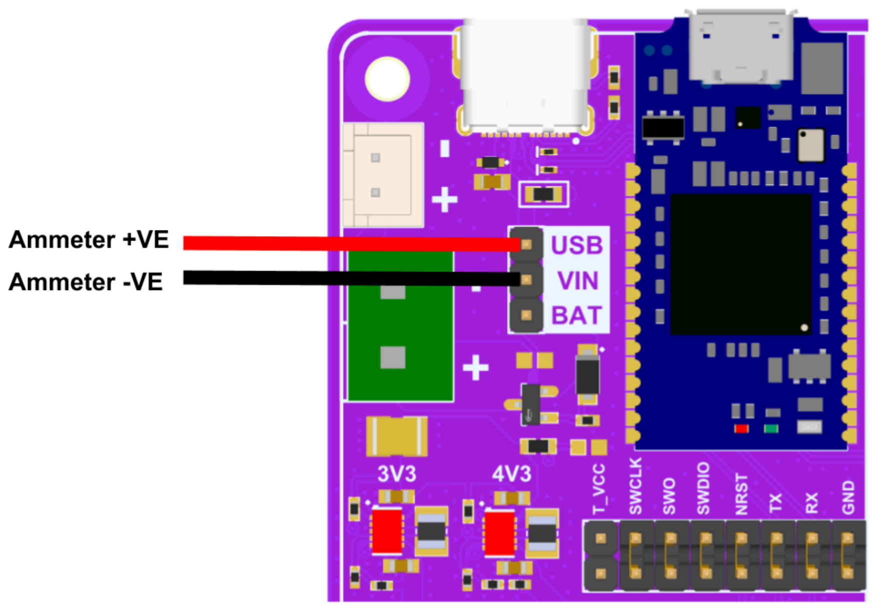
VFEM set to 3.3V. VFEM will be included in power consumption measurements from 3V3 and HALOW headers.



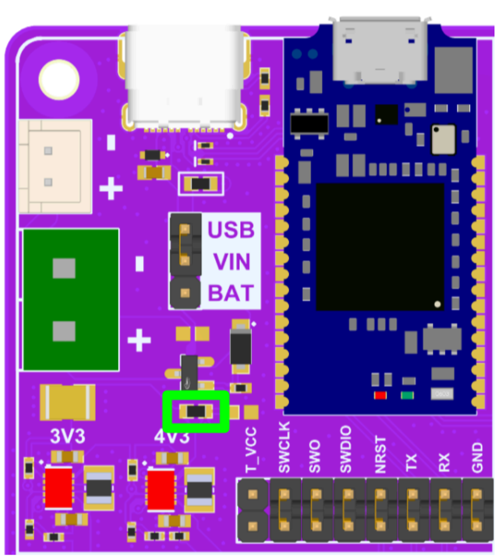
VFEM set to 4.3V. VFEM will not be included in power consumption measurements from 3V3 and HALOW headers. Separate measurement is needed.

7.1.3 Whole System Power Consumption

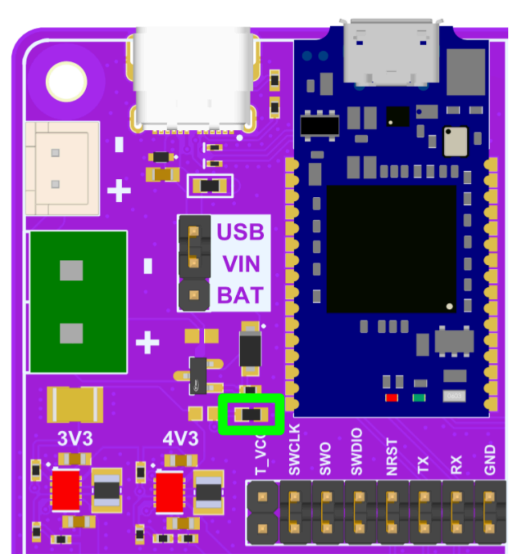
Note that the 3V3 header in the current measurement tree is the output from the onboard buck-boost regulators and does not include the power consumption from the regulators themselves. To measure the entire system including the 3.3V and 4.3V buck-boost regulators, an ammeter can be attached in the following position.



If using only 3.3V for VFEM, the user can disable the 4.3V buck-boost regulator by moving the 0 ohm resistor as shown below. This connects the enable line of the regulator to GND, reducing its power consumption to negligible amounts.



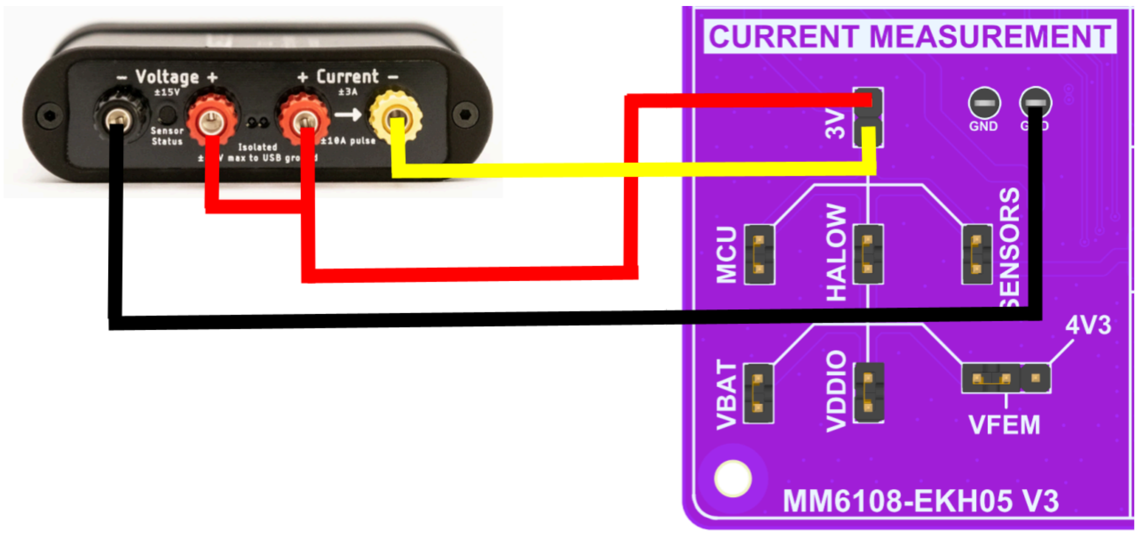
4.3V Regulator Enabled



4.3V Regulator Disabled

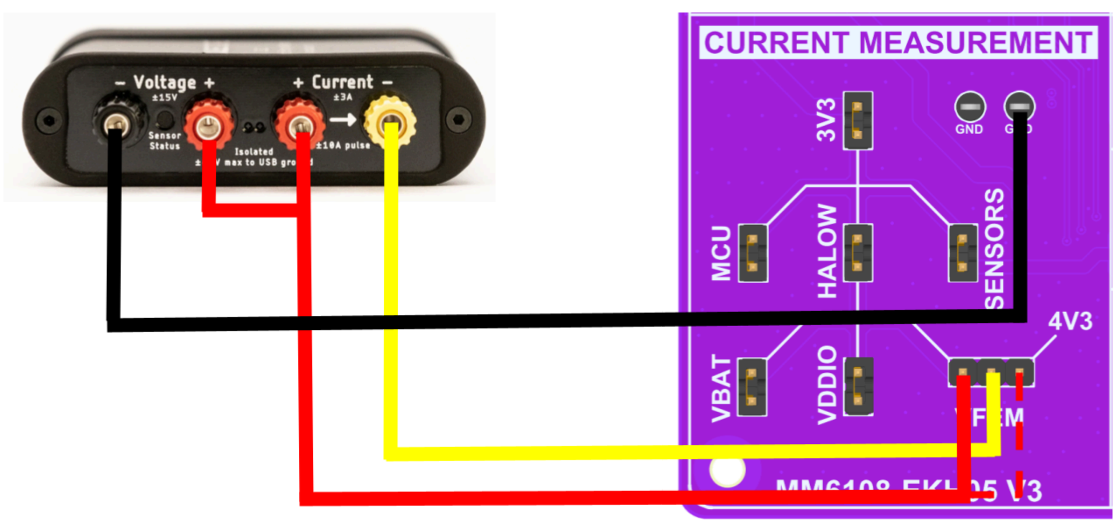
7.2 Power Consumption Measurement Procedure

A high-precision power analyzer, such as the Joulescope [JS220](JS220), is recommended for low power measurements. This device can measure both voltage and current, allowing for a precise calculation of overall power. The suggested configuration is shown below.



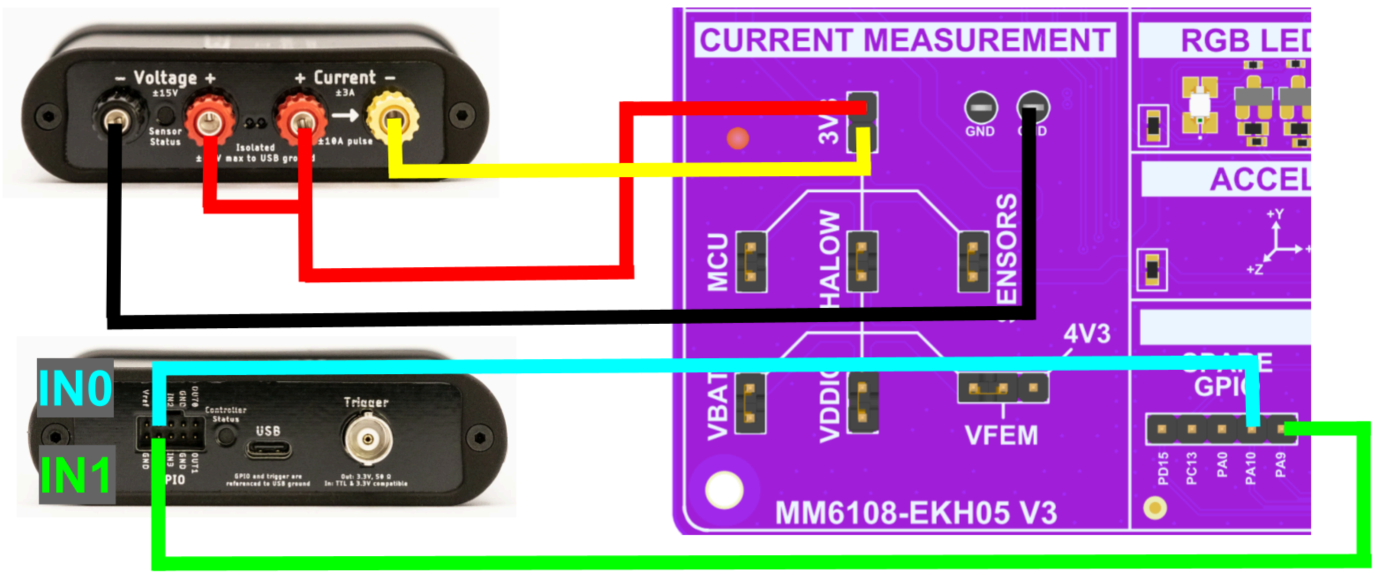
This configuration can be used for the 3V3, MCU, HALOW, SENSORS, VBAT and VDDIO headers, with the positive ammeter position at the top pin of the header, and the negative at the bottom.

For VFEM, the positive ammeter position should be placed on either the leftmost header (3.3V), or the rightmost header (4.3V) depending on which voltage is desired for VFEM. The negative ammeter position should always be connected to the middle header as shown below.

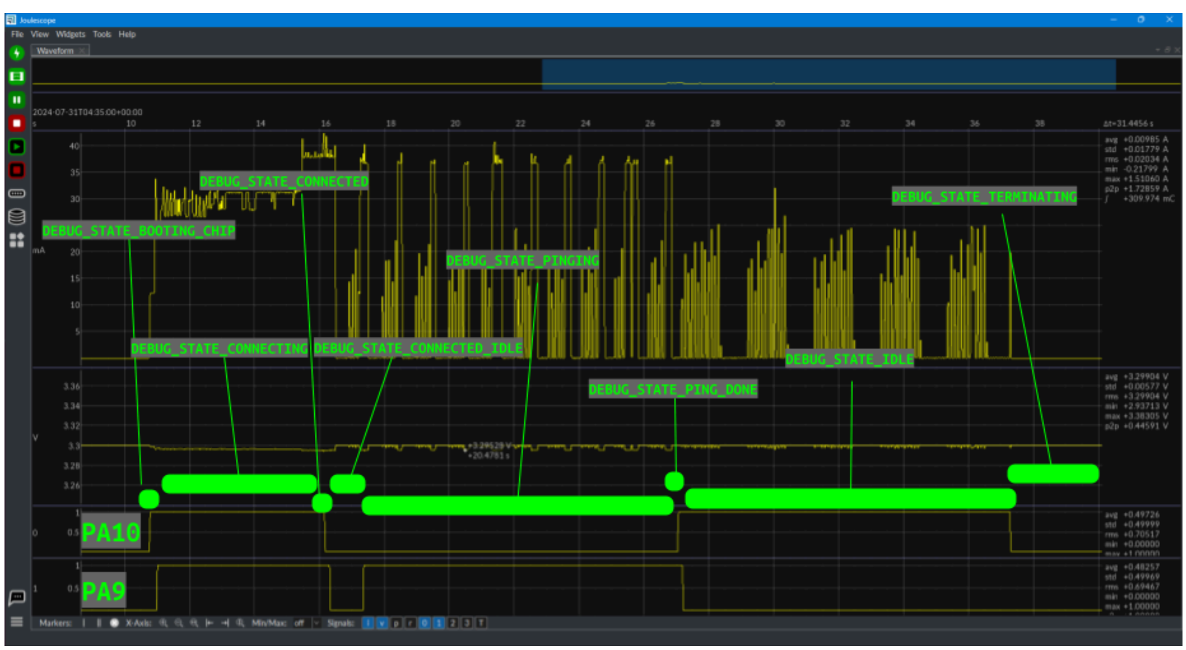


7.3 Optional Debug GPIO Connections

Certain example applications utilise PA9 and PA10 of the STM32U585 as debug GPIOs. These debug GPIOs will toggle when the device is transitioning between different states and can be useful for measuring different parts of the example applications as they execute. These GPIOs are broken out to the SPARE GPIO header on the MM6108-EKH05 and can be connected to the Joulescope as follows.



On the Joulescope UI, the toggling debug GPIOs can be seen at the bottom and are annotated over to show the different states in a ping example application.



Definitions for each debug state can be found in the example folder under each application of interest. For example, the definitions for the ping application can be found in the following locations.

Windows:

```
C:\Users\USER_NAME\STM32Cube\Repository\Packs\MorseMicro\MM_IoT\1.3.3\E
xample\ping\ping.c
```

Linux:

```
/home/USER_NAME/STM32Cube/Repository/Packs/MorseMicro/MM_IoT/1.3.3/Exam
ple/ping/ping.c
```

Mac:

```
/Users/USER_NAME/STM32Cube/Repository/Packs/MorseMicro/MM_IoT/1.3.3/Exa
mple/ping/ping.c
```

A search for the **enum** of **debug_state** will show the definitions. The function **set_debug_state** is used to set these throughout the application.

7.4 Measuring DTIM Power

To measure DTIM power consumption, use the ping application. Make sure to use the correct country code, BCF, SSID, and passphrase to connect the device to the AP. The DTIM period and bandwidth/channel can also be changed by following the steps in Section 4.2.1. To get an accurate measurement for just the HaLow and MCU power consumption, it is recommended to remove the SENSORS header on the power tree.

The ping application will perform the following actions:

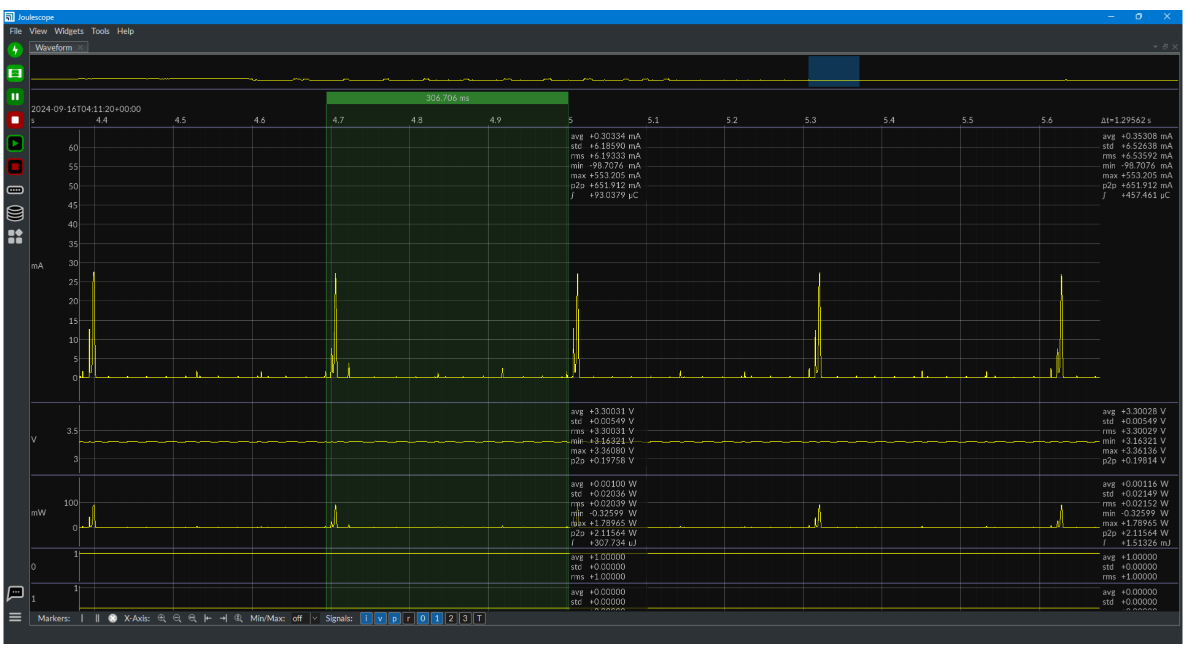
- Boot the MM6108 chip.
- Connect to the AP.
- Ping the AP 10 times.
- Remain connected in power save mode while idling for 60 seconds.
- Disconnect from the AP.

Before starting, ensure that the ping application is loaded onto the device and that the device is properly connected to the AP. This can be confirmed by monitoring the UART output. Note that the UART baud rate is set to 115200.

```
Morse Ping Demo (Built Sep 16 2024 13:44:00)

-----------------------------------
  BCF API version:         6.7.0
  BCF build version:       fbed8cb 244ffd771b
  BCF board description:   BAILEY_V12
  Morselib version:        998b9a497_NFP
  Morse firmware version:  1.13.0
  Morse chip ID:           0x0306
-----------------------------------
Initialize IPv4 with static IP: 192.168.1.2...
Initialize IPv6 using Autoconfig...
Morse FreeRTOS+ TCP interface initialised. MAC address 94:bb:43:dc:f9:52
Attempting to connect to MorseMicro with passphrase 12345678
This may take some time (~30 seconds)
WLAN STA connecting
WLAN STA connected
DNS server 0: 192.168.1.1
Link is up. Time: 7068 ms, IP: 192.168.1.2, Netmask: 255.255.255.0, Gateway: 192.168.1.1, D1


Ping 192.168.1.1 56(64) bytes of data.
(192.168.1.1) packets transmitted/received = 2/2, round-trip min/avg/max = 12/12/12 ms
(192.168.1.1) packets transmitted/received = 3/3, round-trip min/avg/max = 11/11/12 ms
(192.168.1.1) packets transmitted/received = 4/4, round-trip min/avg/max = 11/11/12 ms
(192.168.1.1) packets transmitted/received = 5/5, round-trip min/avg/max = 11/11/12 ms
(192.168.1.1) packets transmitted/received = 6/6, round-trip min/avg/max = 11/11/13 ms
(192.168.1.1) packets transmitted/received = 7/7, round-trip min/avg/max = 11/11/13 ms
(192.168.1.1) packets transmitted/received = 8/8, round-trip min/avg/max = 11/11/13 ms
(192.168.1.1) packets transmitted/received = 9/9, round-trip min/avg/max = 11/11/13 ms
(192.168.1.1) packets transmitted/received = 10/10, round-trip min/avg/max = 11/11/13 ms

--- 192.168.1.1 ping statistics ---
10 packets transmitted, 10 packets received, 0.000% packet loss
round-trip min/avg/max = 11/11/13 ms
WLAN STA disabled
Link is down. Time: 27868 ms
```

The below screenshot shows the power consumption profile as the program executes.

The green highlighted section below illustrates the measurement of a DTIM period, zoomed in from the DTIM Wakeups section displayed above. In this example, the measurement is for a DTIM3 with an 8 MHz channel and a 102.4 ms beacon interval. Adjusting these parameters will impact the measured power consumption.



7.5 Measuring WNM Sleep Power

WNM Sleep provides a more sophisticated sleep mechanism, where the sleep schedule is explicitly negotiated between the station and the AP. This allows for longer sleep intervals and more efficient data retrieval when the client wakes up.

To measure WNM sleep power consumption, use the wnm_sleep application. Ensure that the correct country code, BCF, SSID, and passphrases are configured so the device can connect to the AP. To get an accurate measurement for just the HaLow and MCU power consumption, remove the SENSORS header on the power tree.

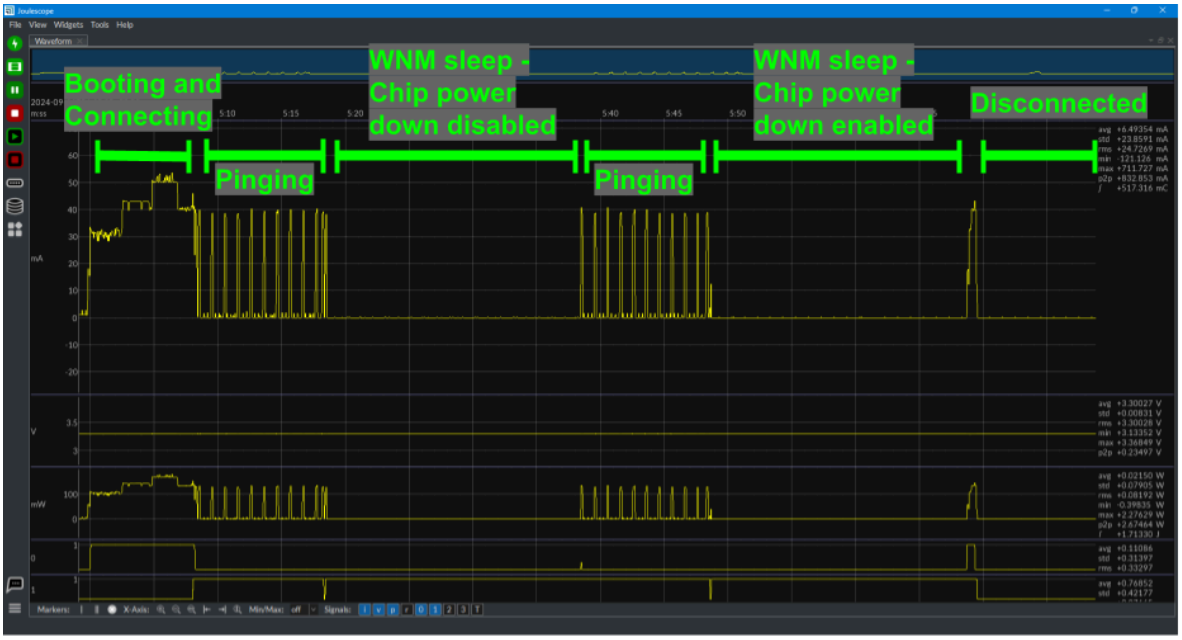
The wnm_sleep application will perform the following actions:

- Boot the MM6108 chip.
- Connect to the AP.
- Ping the AP 10 times.
- Enter WNM sleep for 20 seconds.
- Exit WNM sleep.
- Ping the AP another 10 times.
- Enter WNM sleep with chip in shutdown mode for 20 seconds.
- Exit WNM sleep.
- Disconnect from the AP.

As with DTIM measurements, start by verifying that the wnm_sleep example is correctly loaded and that the device can successfully associate with the AP via the MM6108-EKH05's UART. Below are screenshots of the complete UART output from the program, along with an annotated power trace.

```
Morse WNM Sleep Demo (Built Sep 16 2024 14:15:05)

-----------------------------------
  BCF API version:         6.7.0
  BCF build version:       fbed8cb 244ffd771b
  BCF board description:   BAILEY_V12
  Morselib version:        998b9a497_NFP
  Morse firmware version:  1.13.0
  Morse chip ID:           0x0306
-----------------------------------
Initialize IPv4 with static IP: 192.168.1.2...
Initialize IPv6 using Autoconfig...
Morse FreeRTOS+ TCP interface initialised. MAC address 94:bb:43:dc:f9:52
Attempting to connect to MorseMicro with passphrase 12345678
This may take some time (~30 seconds)
WLAN STA connecting
WLAN STA connected
DNS server 0: 192.168.1.1
Link is up. Time: 8176 ms, IP: 192.168.1.2, Netmask: 255.255.255.0, Gateway: 192.168.1.1, D1


Ping 192.168.1.1 56(64) bytes of data.
(192.168.1.1) packets transmitted/received = 2/2, round-trip min/avg/max = 12/12/12 ms
(192.168.1.1) packets transmitted/received = 3/3, round-trip min/avg/max = 12/12/12 ms
(192.168.1.1) packets transmitted/received = 4/4, round-trip min/avg/max = 11/11/12 ms
(192.168.1.1) packets transmitted/received = 5/5, round-trip min/avg/max = 11/11/12 ms
(192.168.1.1) packets transmitted/received = 6/6, round-trip min/avg/max = 11/12/14 ms
(192.168.1.1) packets transmitted/received = 7/7, round-trip min/avg/max = 11/12/14 ms
(192.168.1.1) packets transmitted/received = 8/8, round-trip min/avg/max = 11/12/14 ms
(192.168.1.1) packets transmitted/received = 9/9, round-trip min/avg/max = 11/11/14 ms
(192.168.1.1) packets transmitted/received = 10/10, round-trip min/avg/max = 11/11/14 ms

--- 192.168.1.1 ping statistics ---
10 packets transmitted, 10 packets received, 0.000% packet loss
round-trip min/avg/max = 11/11/14 ms

Entering WNM sleep with chip power down disabled
Expected sleep time 20000ms.

Enter WNM sleep took 16 ms.

Exit WNM sleep took 17 ms.
```

```
Ping 192.168.1.1 56(64) bytes of data.
(192.168.1.1) packets transmitted/received = 2/2, round-trip min/avg/max = 3/7/11 ms
(192.168.1.1) packets transmitted/received = 3/3, round-trip min/avg/max = 3/9/13 ms
(192.168.1.1) packets transmitted/received = 4/4, round-trip min/avg/max = 3/9/13 ms
(192.168.1.1) packets transmitted/received = 5/5, round-trip min/avg/max = 3/14/35 ms
(192.168.1.1) packets transmitted/received = 6/6, round-trip min/avg/max = 3/14/35 ms
(192.168.1.1) packets transmitted/received = 7/7, round-trip min/avg/max = 3/14/35 ms
(192.168.1.1) packets transmitted/received = 8/8, round-trip min/avg/max = 3/14/35 ms
(192.168.1.1) packets transmitted/received = 9/9, round-trip min/avg/max = 3/14/35 ms
(192.168.1.1) packets transmitted/received = 10/10, round-trip min/avg/max = 3/14/35 ms

--- 192.168.1.1 ping statistics ---
10 packets transmitted, 10 packets received, 0.000% packet loss
round-trip min/avg/max = 3/14/35 ms

Entering WNM sleep with chip power down enabled.
Expected sleep time 20000ms.

Enter WNM sleep took 13 ms.

Exit WNM sleep took 623 ms.
WLAN STA disabled
Link is down. Time: 69345 ms
```

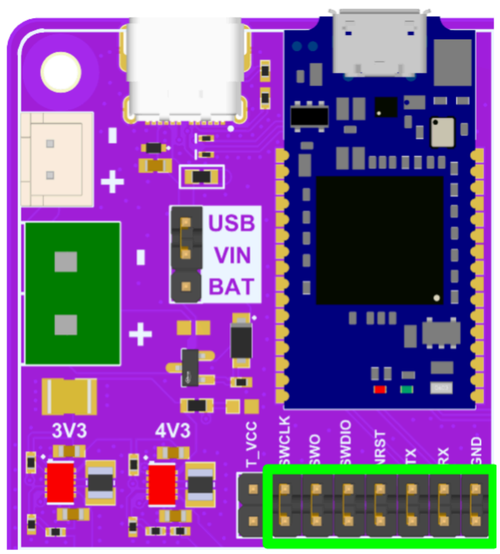

Booting and Connecting
Pinging
WNM sleep - Chip power down disabled
Pinging
WNM sleep - Chip power down enabled
Disconnected

8 Troubleshooting

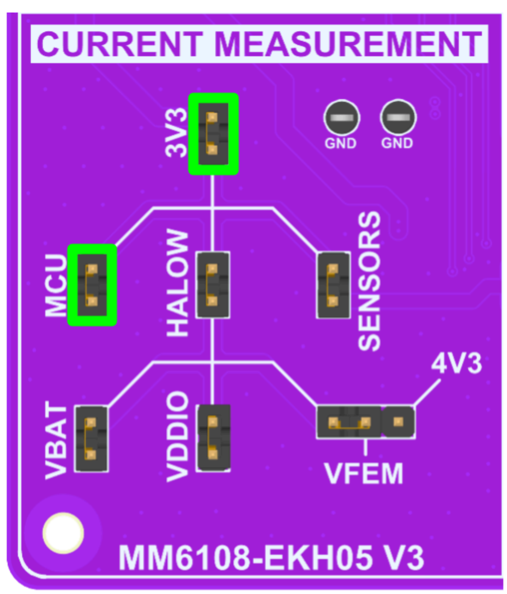
8.1 Device Is Not Programming

If the STLINK V3-MODS is unable to connect to the STM32U585, please check the following steps.

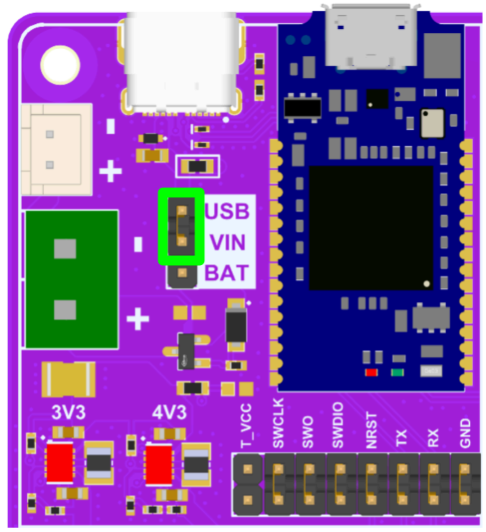
1. Ensure the STLINK jumpers are populated. Without these, the programmer will be disconnected from the microcontroller.



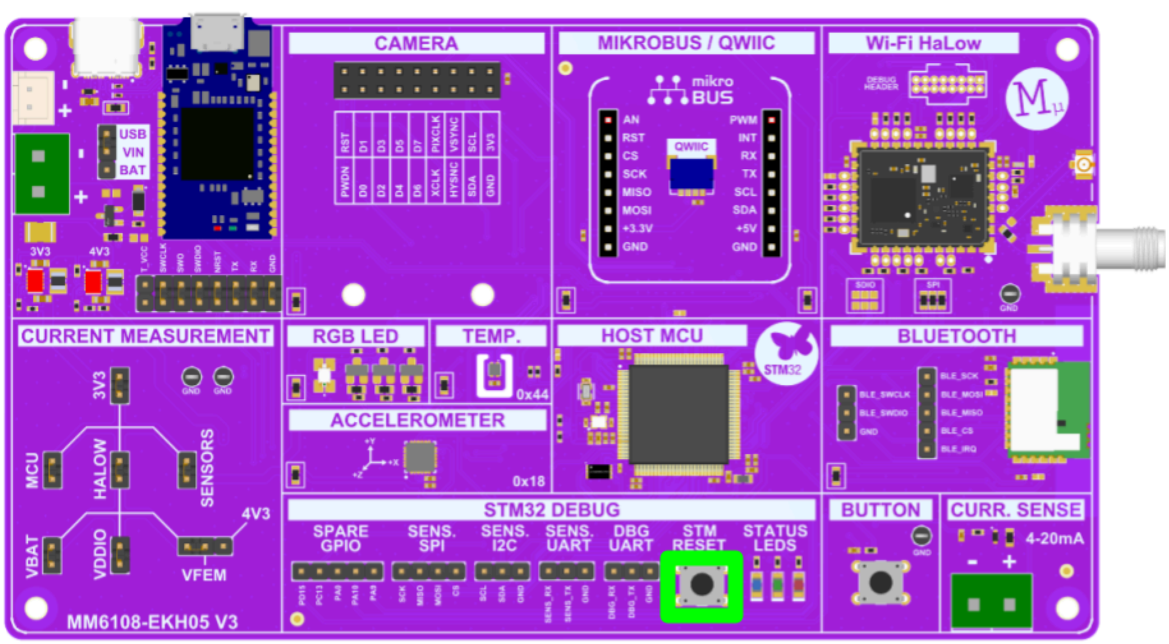
2. Ensure the 3V3 and MCU jumpers are populated. The microcontroller will not be powered without these.



3. Ensure the power select jumper is on USB. If using batteries, check VIN voltage to ensure the battery is functioning normally.



4. Ensure the device is not in a low power state. If the microcontroller is in certain deep sleep states, the programmer will not be able to attach. To recover in this situation, hold the STM RESET button as the device is attempting to connect, and release. This will ensure the microcontroller can be attached to.



8.2 Poor HaLow Performance

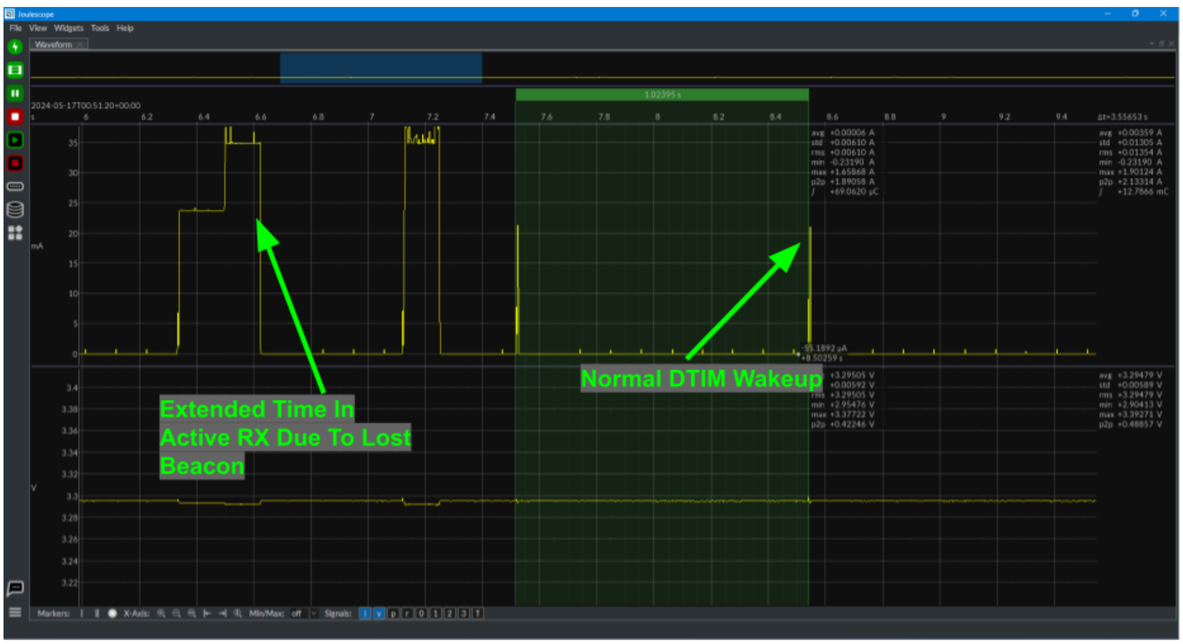
If the HaLow performance is poor, the following items should be checked.

1. Confirm the BCF loaded on to the device matches the VFEM setting. The bcf_aw_hm593.bin file should be used for 3.3V, and the bcf_aw_hm593_4v3.bin file should be used for 4.3V. If using a Morse Micro example from the CMSIS pack, the software will automatically detect which setting is used and select the correct BCF. If this is the case, the device might just need to be reset with the STM RESET button.
2. Ensure the AP configuration is expected. I.e. the correct channel and bandwidth are configured.
3. Ensure there are no other HaLow devices on the same channel causing in-band interference.
4. Ensure antenna connection is on securely.
5. If cabled, ensure there is sufficient attenuation (~40 dB) between the devices. If the power is too high, it will saturate the LNA.

8.3 High Power Consumption

There are many reasons that high power consumption may be measured. Please check the following:

1. If the T_VCC jumper is populated, this will add ~300uA to the 3V3 power consumption.
2. When measuring just HaLow and MCU power consumption, ensure the SENSORS jumper is not populated.
3. To replicate Morse Micro DTIM power consumption numbers, the devices should be placed inside a shielded box away from interference. If the device fails to receive the beacon from the AP, it will spend a longer time in active RX mode and consume more power resulting in misleading DTIM measurements.



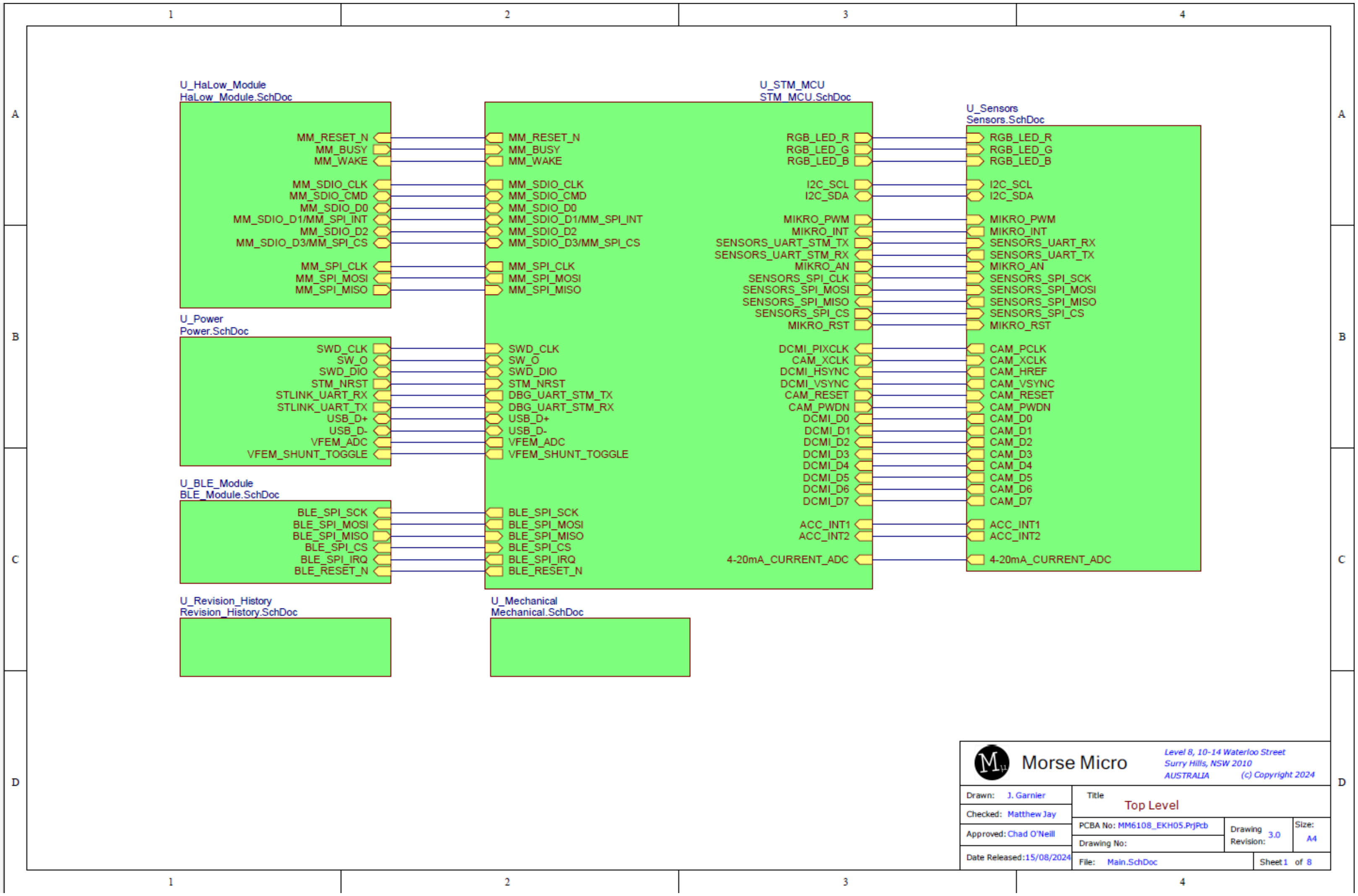
4. The temperature plays a big factor in power consumption. All testing should be done at room temperature of ~77 degree Fahrenheit to match Morse Micro DTIM power consumption numbers.
5. Ensure the MM6108-EKH05 is associated with the AP. Double check SSID and passphrase match.

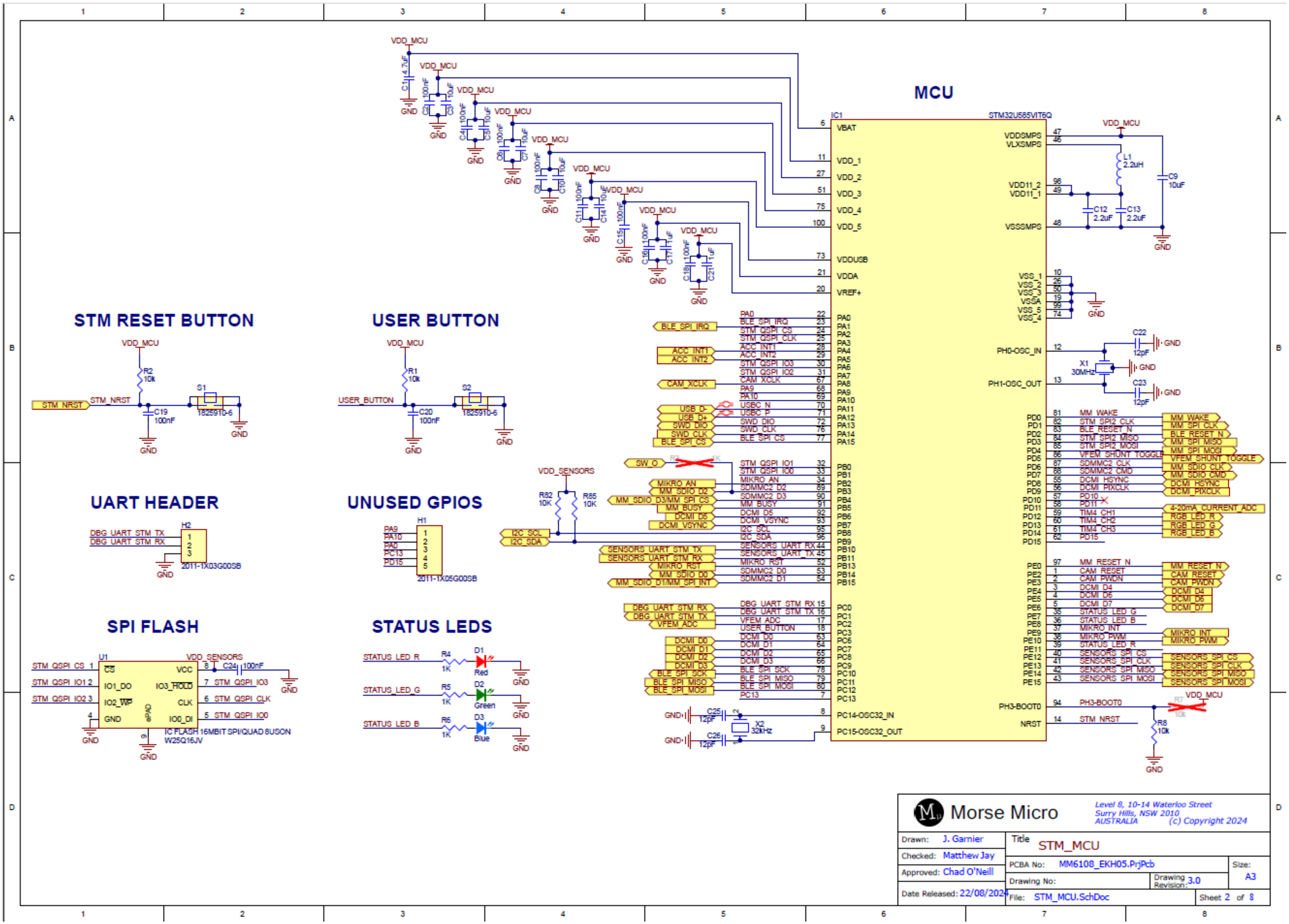
9 Errata

10 Revision History

Release Number	Release Date	Release Notes
01	19/11/2024	Initial release.

Appendix A: Schematics




MCU
STM32U585VIT6Q
STM RESET BUTTON
USER BUTTON
UART HEADER
UNUSED GPIOS
SPI FLASH
STATUS LEDS
Morse Micro
Level 8, 10-14 Waterloo Street
Surry Hills, NSW 2010
AUSTRALIA
(c) Copyright 2024
Drawn: J. Garnier
Checked: Matthew Jay
Approved: Chad O'Neill
Date Released: 22/08/2024
Title STM_MCU
PCBA No: MM6108_EKH05.PrjPcb
Drawing No:
Drawing Revision: 3.0
Size: A3
File: STM_MCU.SchDoc
Sheet 2 of 8

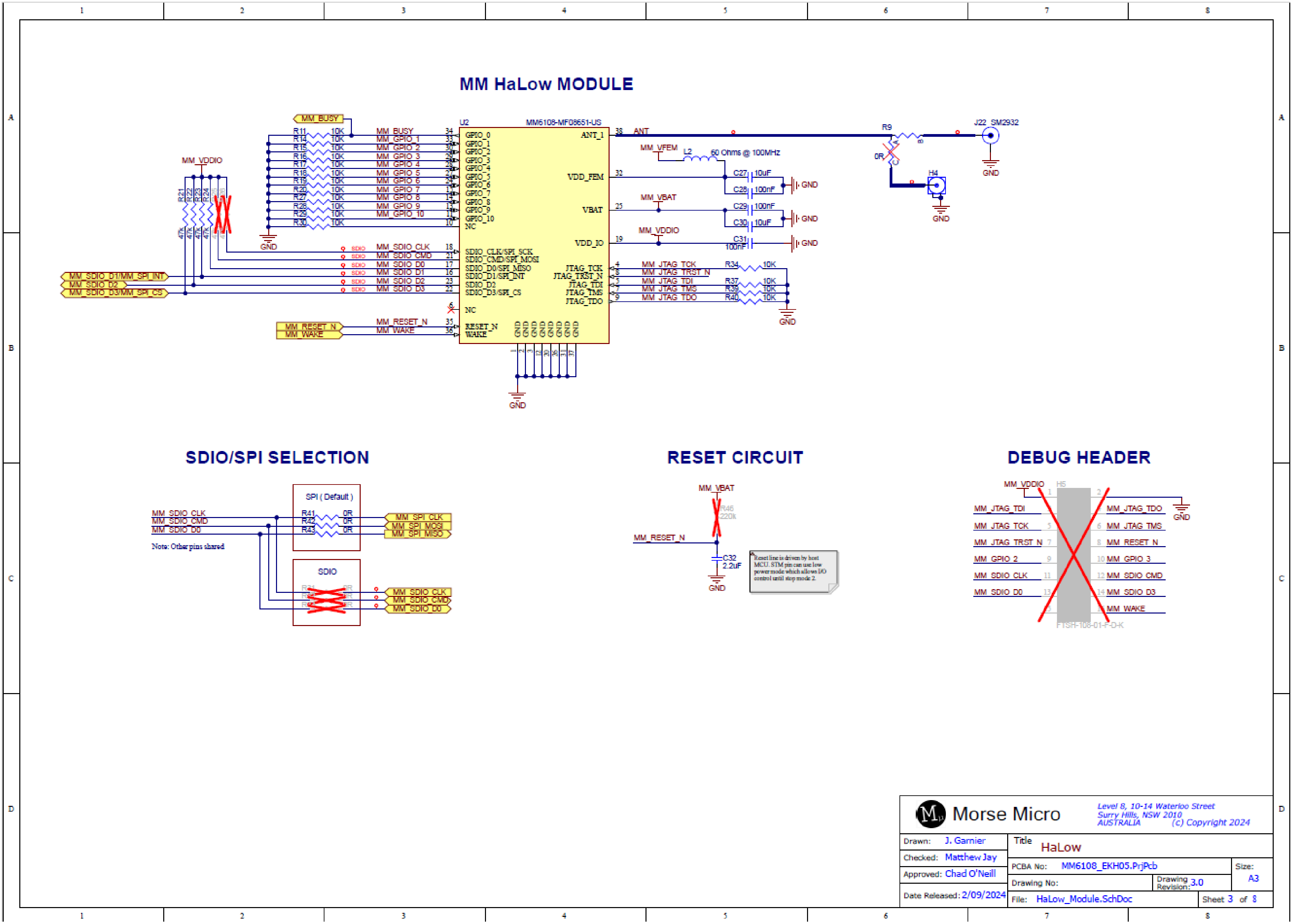
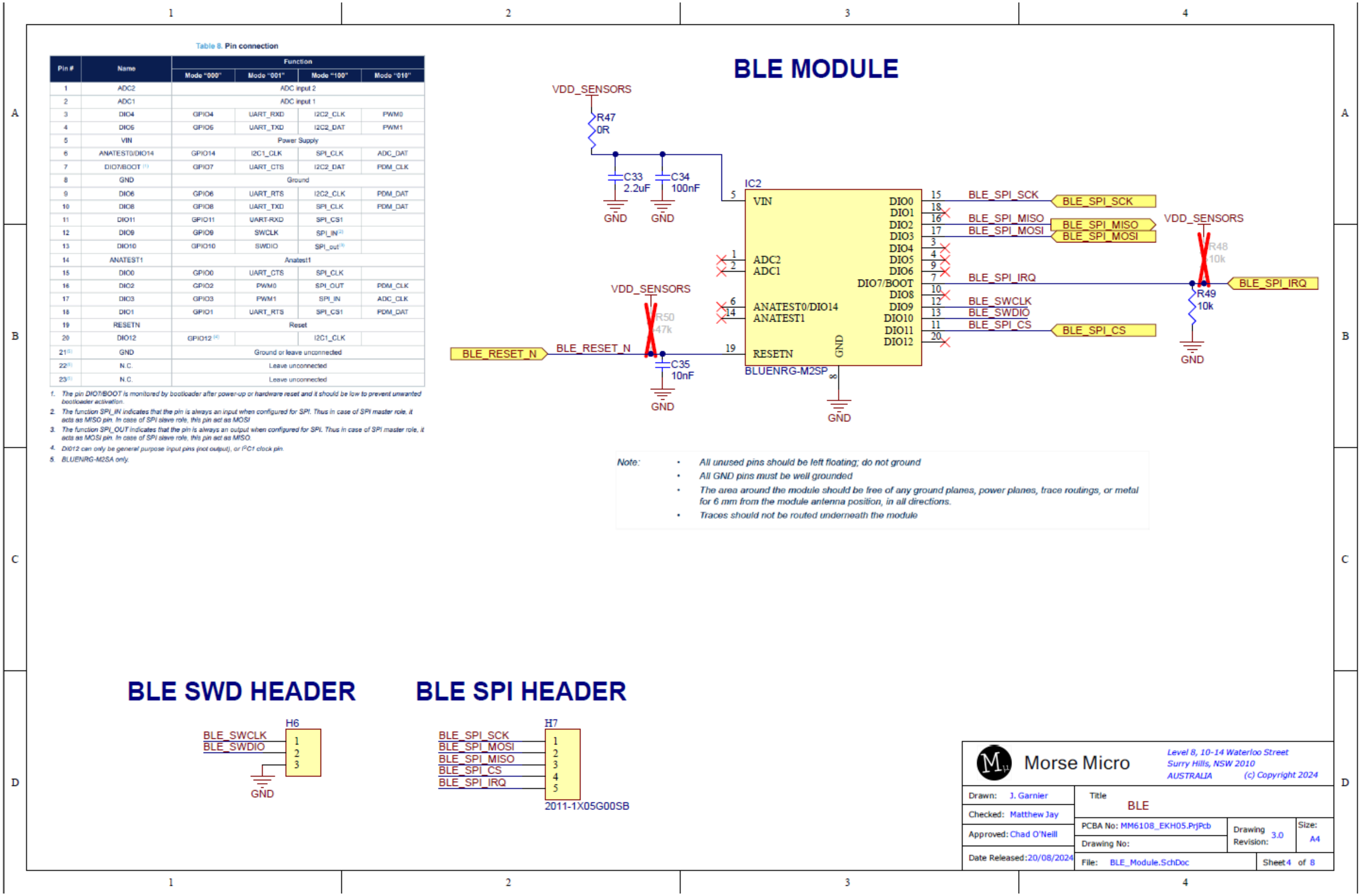
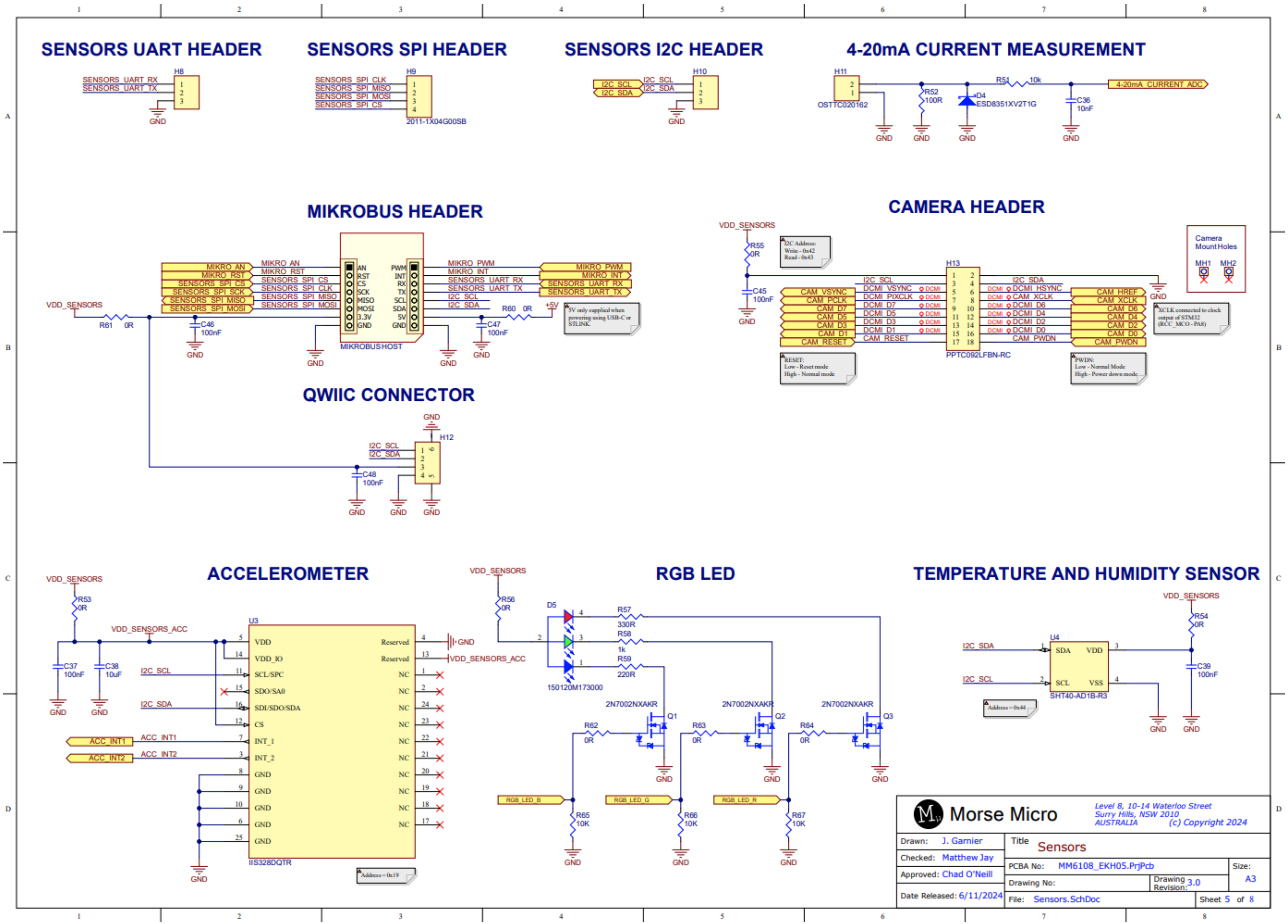

MM HaLow MODULE
U2
MM6108-MF08651-US
ANT_1
ANT
VDD_FEM
VBAT
VDD_IO
MM_VFEM
MM_VBAT
MM_VDDIO
L2
60 Ohms @ 100MHz
C27 10uF
C28 100nF
C29 100nF
C30 10uF
C31 100nF
GND
R9
0R
J22 SM2932
H4
MM_BUSY
MM_GPIO_1
MM_GPIO_2
MM_GPIO_3
MM_GPIO_4
MM_GPIO_5
MM_GPIO_6
MM_GPIO_7
MM_GPIO_8
MM_GPIO_9
MM_GPIO_10
MM_SDIO_CLK
MM_SDIO_CMD
MM_SDIO_D0
MM_SDIO_D1
MM_SDIO_D2
MM_SDIO_D3
MM_SDIO_D1/MM_SPI_INT
MM_SDIO_D2
MM_SDIO_D3/MM_SPI_CS
MM_RESET_N
MM_WAKE
MM_JTAG_TCK
MM_JTAG_TRST_N
MM_JTAG_TDI
MM_JTAG_TMS
MM_JTAG_TDO
SDIO/SPI SELECTION
SPI (Default)
SDIO
R41 0R
R42 0R
R43 0R
MM_SPI_CLK
MM_SPI_MOSI
MM_SPI_MISO
Note: Other pins shared
RESET CIRCUIT
R46 220k
C32 2.2uF
Reset line is driven by host MCU. STM pin can use low power mode which allows I/O control until stop mode 2.
DEBUG HEADER
FTSH-108-01-F-D-K
Morse Micro
Level 8, 10-14 Waterloo Street
Surry Hills, NSW 2010
AUSTRALIA
(c) Copyright 2024
Drawn: J. Garnier
Checked: Matthew Jay
Approved: Chad O'Neill
Date Released: 2/09/2024
Title HaLow
PCBA No: MM6108_EKH05.PrjPcb
Drawing No:
Drawing Revision: 3.0
Size: A3
File: HaLow_Module.SchDoc
Sheet 3 of 8

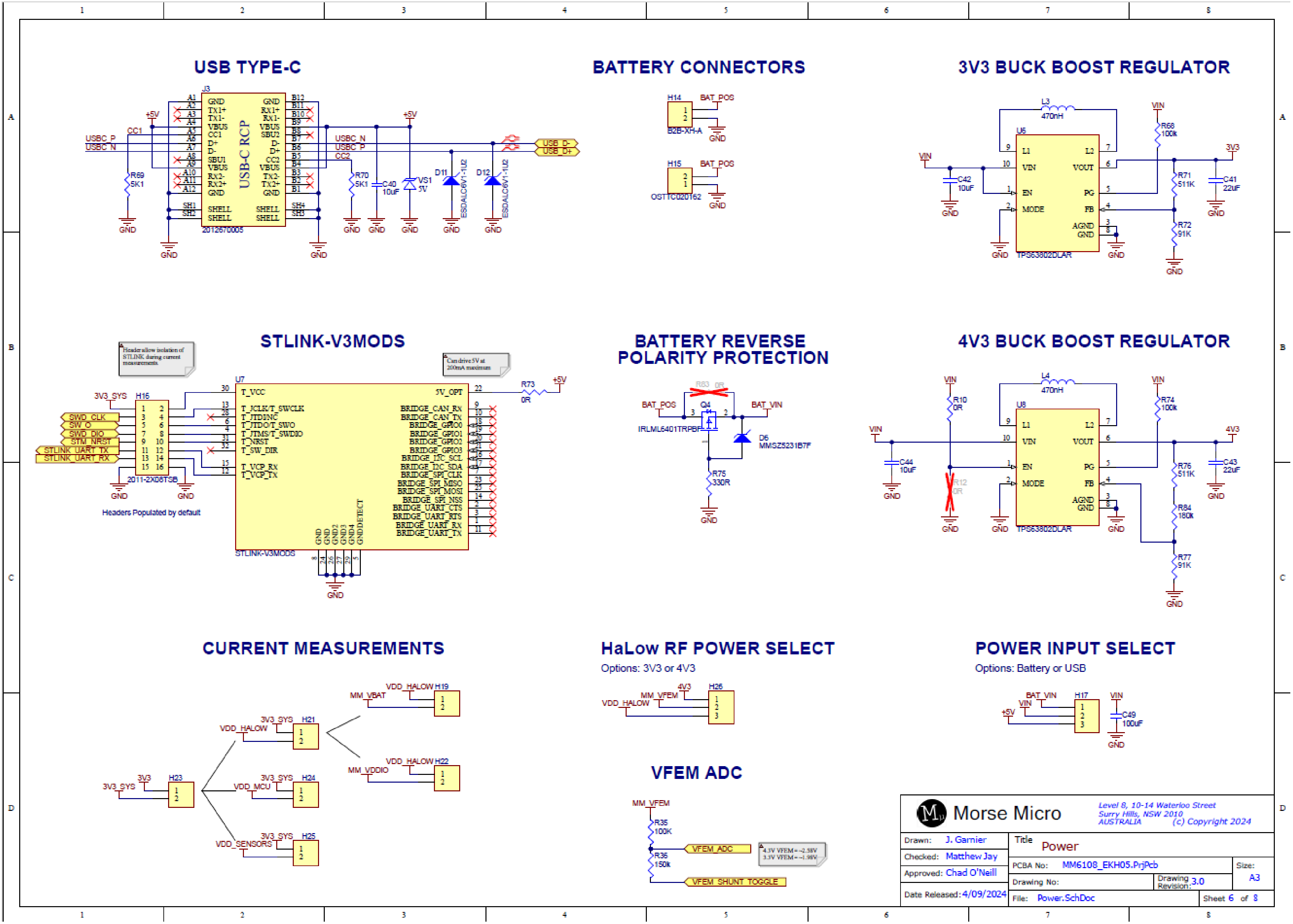
Table 8. Pin connection

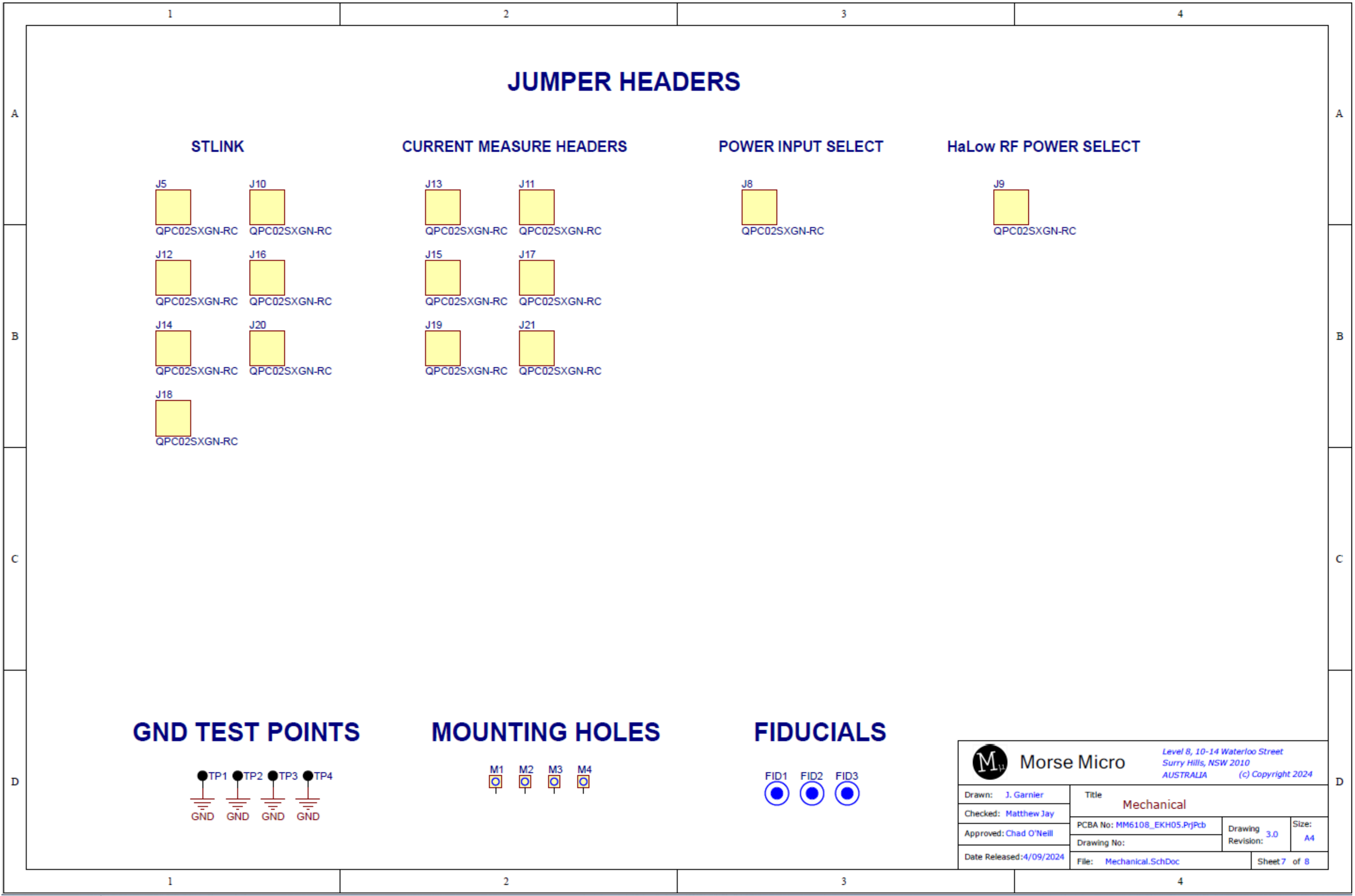
Pin #	Name	Function			
		Mode "000"	Mode "001"	Mode "100"	Mode "010"
1	ADC2	ADC input 2			
2	ADC1	ADC input 1			
3	DIO4	GPIO4	UART_RXD	I2C2_CLK	PWM0
4	DIO5	GPIO5	UART_TXD	I2C2_DAT	PWM1
5	VIN	Power Supply			
6	ANATEST0/DIO14	GPIO14	I2C1_CLK	SPI_CLK	ADC_DAT
7	DIO7/BOOT [(1)]	GPIO7	UART_CTS	I2C2_DAT	PDM_CLK
8	GND	Ground			
9	DIO6	GPIO6	UART_RTS	I2C2_CLK	PDM_DAT
10	DIO8	GPIO8	UART_TXD	SPI_CLK	PDM_DAT
11	DIO11	GPIO11	UART-RXD	SPI_CS1	
12	DIO9	GPIO9	SWCLK	SPI_IN[(2)]	
13	DIO10	GPIO10	SWDIO	SPI_out[(3)]	
14	ANATEST1	Anatest1			
15	DIO0	GPIO0	UART_CTS	SPI_CLK	
16	DIO2	GPIO2	PWM0	SPI_OUT	PDM_CLK
17	DIO3	GPIO3	PWM1	SPI_IN	ADC_CLK
18	DIO1	GPIO1	UART_RTS	SPI_CS1	PDM_DAT
19	RESETN	Reset			
20	DIO12	GPIO12 [(4)]		I2C1_CLK	
21[(5)]	GND	Ground or leave unconnected			
22[(5)]	N.C.	Leave unconnected			
23[(5)]	N.C.	Leave unconnected			

1. *The pin DIO7/BOOT is monitored by bootloader after power-up or hardware reset and it should be low to prevent unwanted bootloader activation.*
2. *The function SPI_IN indicates that the pin is always an input when configured for SPI. Thus in case of SPI master role, it acts as MISO pin. In case of SPI slave role, this pin act as MOSI*
3. *The function SPI_OUT indicates that the pin is always an output when configured for SPI. Thus in case of SPI master role, it acts as MOSI pin. In case of SPI slave role, this pin act as MISO.*
4. *DI012 can only be general purpose input pins (not output), or I^2C1 clock pin.*
5. *BLUENRG-M2SA only.*




SENSORS UART HEADER
SENSORS SPI HEADER
SENSORS I2C HEADER
4-20mA CURRENT MEASUREMENT
MIKROBUS HEADER
CAMERA HEADER
QWIIC CONNECTOR
ACCELEROMETER
RGB LED
TEMPERATURE AND HUMIDITY SENSOR
SENSORS_UART_RX
SENSORS_UART_TX
SENSORS_SPI_CLK
SENSORS_SPI_MISO
SENSORS_SPI_MOSI
SENSORS_SPI_CS
2011-1X04G00SB
I2C_SCL
I2C_SDA
OSTTC020162
R52 100R
D4 ESD8351XV2T1G
R51 10k
C36 10nF
4-20mA_CURRENT_ADC
MIKRO_AN
MIKRO_RST
MIKRO_PWM
MIKRO_INT
MIKROBUSHOST
VDD_SENSORS
R61 0R
C46 100nF
C47 100nF
R60 0R
+5V
5V only supplied when powering using USB-C or STLINK.
R55 0R
C45 100nF
I2C Address: Write - 0x42 Read - 0x43
PPTC092LFBN-RC
CAM_VSYNC
CAM_PCLK
CAM_RESET
CAM_HREF
CAM_XCLK
CAM_PWDN
RESET: Low - Reset mode High - Normal mode
PWDN: Low - Normal Mode High - Power down mode
XCLK connected to clock output of STM32 (RCC_MCO - PA8)
Camera Mount Holes
MH1
MH2
C48 100nF
H12
GND
VDD_SENSORS_ACC
R53 0R
C37 100nF
C38 10uF
ACC_INT1
ACC_INT2
IIS328DQTR
Address = 0x19
R56 0R
D5
150120M173000
R57 330R
R58 1k
R59 220R
2N7002NXAKR
Q1
Q2
Q3
R62 0R
R63 0R
R64 0R
RGB_LED_B
RGB_LED_G
RGB_LED_R
R65 10K
R66 10K
R67 10K
R54 0R
U4
SHT40-AD1B-R3
C39 100nF
Address = 0x44
Morse Micro
Level 8, 10-14 Waterloo Street
Surry Hills, NSW 2010
AUSTRALIA
(c) Copyright 2024
Drawn: J. Garnier
Checked: Matthew Jay
Approved: Chad O'Neill
Date Released: 6/11/2024
Title Sensors
PCBA No: MM6108_EKH05.PrjPcb
Drawing No:
Drawing Revision: 3.0
Size: A3
File: Sensors.SchDoc
Sheet 5 of 8


USB TYPE-C
BATTERY CONNECTORS
3V3 BUCK BOOST REGULATOR
STLINK-V3MODS
BATTERY REVERSE POLARITY PROTECTION
4V3 BUCK BOOST REGULATOR
CURRENT MEASUREMENTS
HaLow RF POWER SELECT
Options: 3V3 or 4V3
POWER INPUT SELECT
Options: Battery or USB
VFEM ADC
Headers Populated by default
Morse Micro
Level 8, 10-14 Waterloo Street
Surry Hills, NSW 2010
AUSTRALIA
(c) Copyright 2024
Drawn: J. Garnier
Checked: Matthew Jay
Approved: Chad O'Neill
Date Released: 4/09/2024
Title Power
PCBA No: MM6108_EKH05.PrjPcb
Drawing No:
Drawing Revision: 3.0
Size: A3
File: Power.SchDoc
Sheet 6 of 8


JUMPER HEADERS
STLINK
CURRENT MEASURE HEADERS
POWER INPUT SELECT
HaLow RF POWER SELECT
J5 QPC02SXGN-RC
J10 QPC02SXGN-RC
J12 QPC02SXGN-RC
J16 QPC02SXGN-RC
J14 QPC02SXGN-RC
J20 QPC02SXGN-RC
J18 QPC02SXGN-RC
J13 QPC02SXGN-RC
J11 QPC02SXGN-RC
J15 QPC02SXGN-RC
J17 QPC02SXGN-RC
J19 QPC02SXGN-RC
J21 QPC02SXGN-RC
J8 QPC02SXGN-RC
J9 QPC02SXGN-RC
GND TEST POINTS
TP1 TP2 TP3 TP4
GND GND GND GND
MOUNTING HOLES
M1 M2 M3 M4
FIDUCIALS
FID1 FID2 FID3
Morse Micro
Level 8, 10-14 Waterloo Street
Surry Hills, NSW 2010
AUSTRALIA
(c) Copyright 2024
Drawn: J. Garnier
Checked: Matthew Jay
Approved: Chad O'Neill
Date Released: 4/09/2024
Title
Mechanical
PCBA No: MM6108_EKH05.PrjPcb
Drawing No:
Drawing Revision: 3.0
Size: A4
File: Mechanical.SchDoc
Sheet 7 of 8

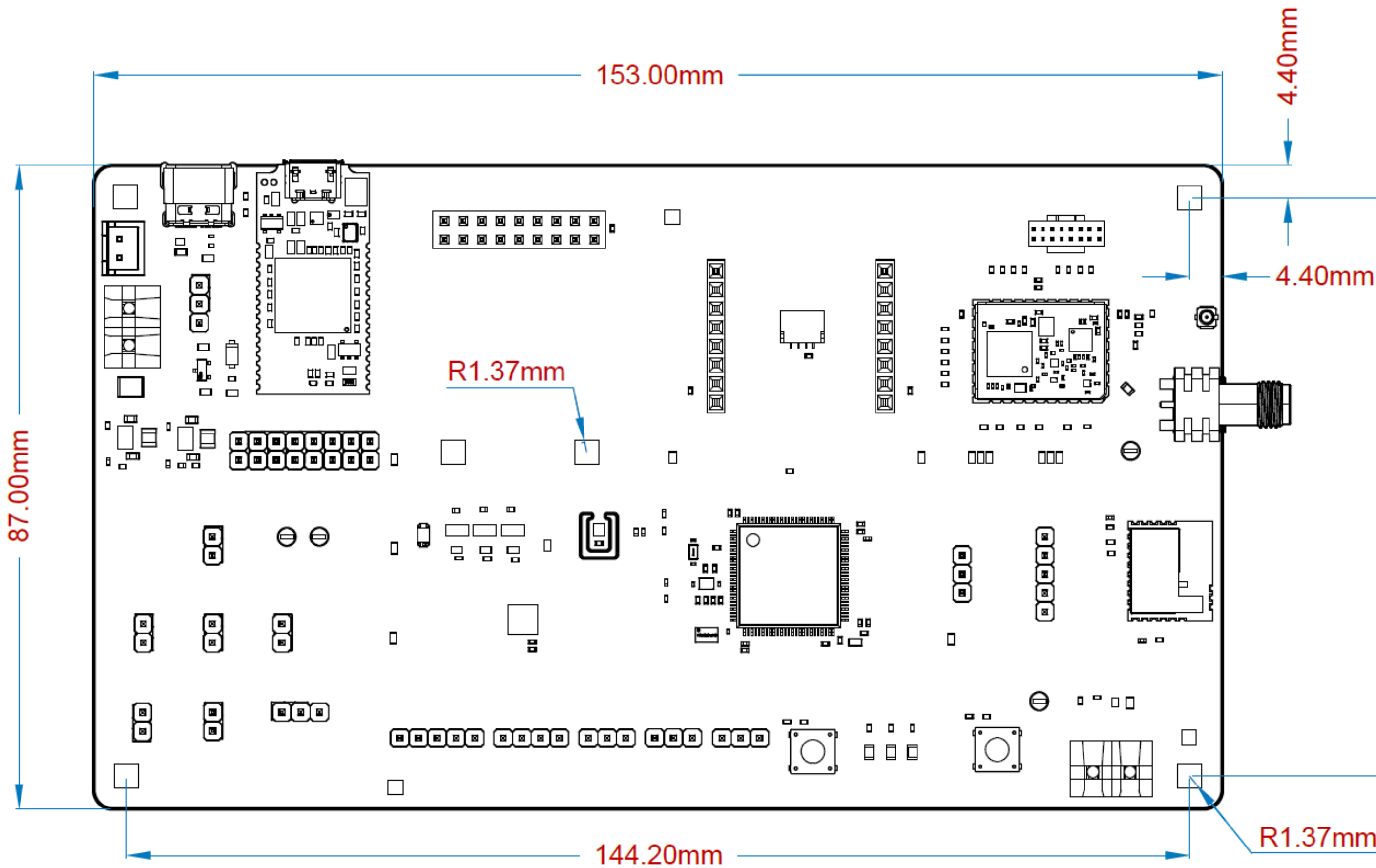
Revision History	
1.0	- Initial release.
2.0	- Connect USB data to STM - Connect RTS/CTS to BLE Module. - Swap UART pins of BLE Module. - Update buck boost output cap to 22uF - Reversed UART RX/TX connections on STLINK - Moved power supply for QSPI Flash to VDD_SENSORS - Updated battery connector to JST - Added bulk cap on Power select - Removed load switch for sensors power supply - Added option to change VFEM to 4.3V
3.0	- Updated SMA footprint. - Updated BLE module communication protocol from UART to SPI. Updated pinout as needed. - Removed 0ohm links to unneeded SPI/SDIO connections. - Combined the MM_SPI_CS and MM_SPI_INT pins with SDIO - Pin assignment updates to red status LED, MIKRO_AN and MIKRO_PWM - Updated VFEM High Power Select to 4.3V - Removed Vin & VFEM from power measure tree

Morse Micro

Level 8, 10-14 Waterloo Street
Surry Hills, NSW 2010
AUSTRALIA

Drawn: J. Garnier	Title: Revision History		
Checked: Matthew Jay	PCBA No: MM6108_EKH05.PrjPcb	Drawing Revision: 3.0	Size: A4
Approved: Chad O'Neill	Drawing No:		
Date Released: 2/09/2024	File: Revision_History.SchDoc		Sheet 8 of 8

Appendix B: Mechanical Drawing



Morse Micro Pty. Ltd. Corporate Headquarters
Level 8, 10-14 Waterloo Street, Surry Hills, NSW 2010,
AUSTRALIA

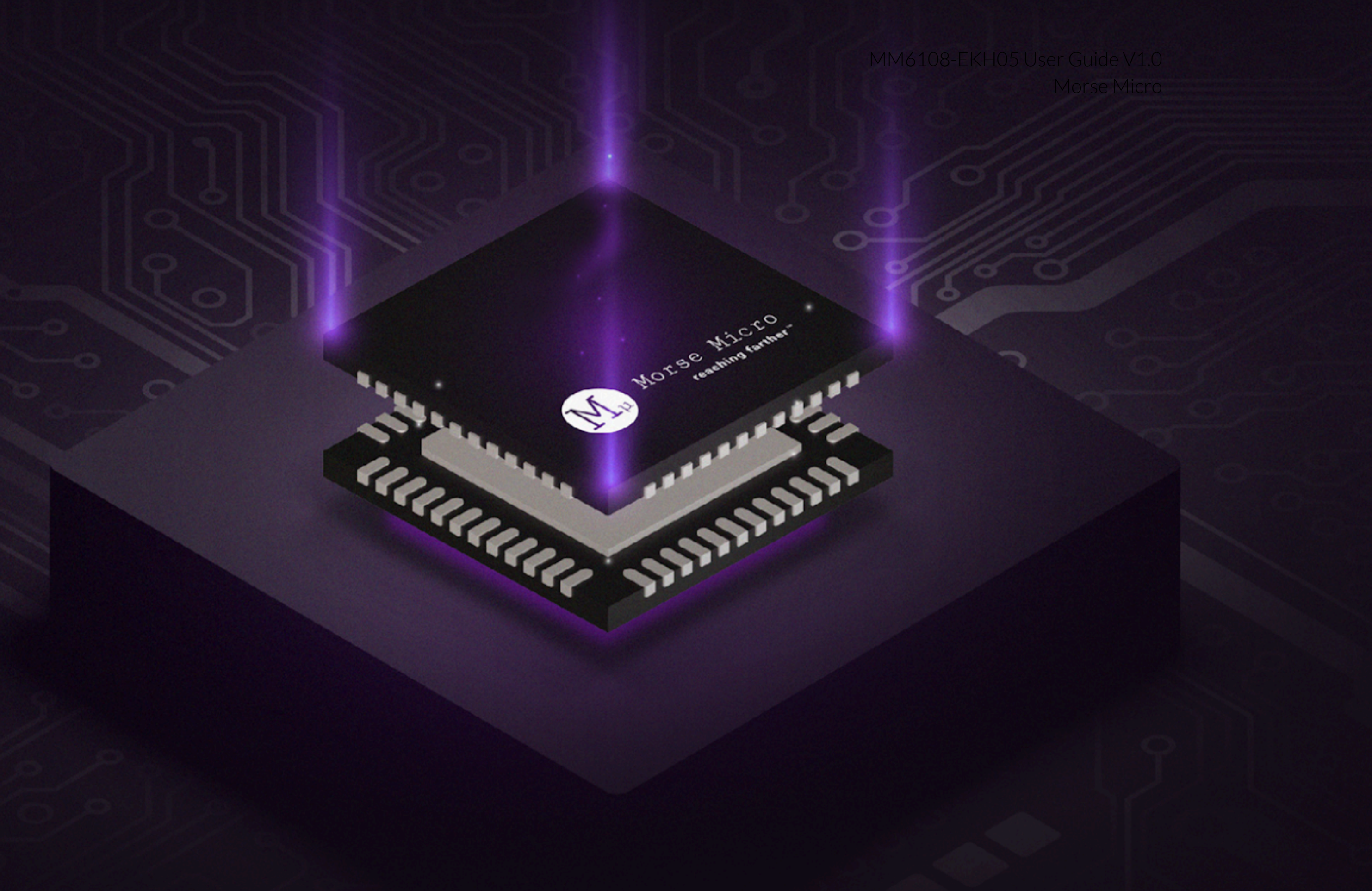
Morse Micro Inc. – USA
40 Waterworks Way
Irvine, CA 92618, UNITED STATES

Morse Micro – China
Floor 7, Room 08-210, Wework office 292 Yan'an Rd,
Shangcheng District, Hangzhou 310003

USA: +1.949.501.7080
Australia: +61.02.905.45922
China: +86.571.229.30277
Email: sales@morsemicro.com
www.morsemicro.com

About Morse Micro

Morse Micro is producing IEEE 802.11ah / Wi-Fi HaLow solutions for Internet of Things (IOT) - based on a newly certified Wi-Fi standard called HaLow. Morse Micro is a VC-backed Startup headquartered in Sydney, Australia. Learn more at www.morsemicro.com



Morse Micro
reaching farther™