Synthesis @ ASU M5StickC Setup & Use Guide

2021, Connor Rawls, Synthesis @ ASU, Arizona State University

Table of Contents

1 Introduction

This guide is meant as a resource for those intending to use the M5Stickc device with the Synthesis @ ASU firmware. Depending on what type of configuration you intend to do, you may want to jump to the later sections of this guide.

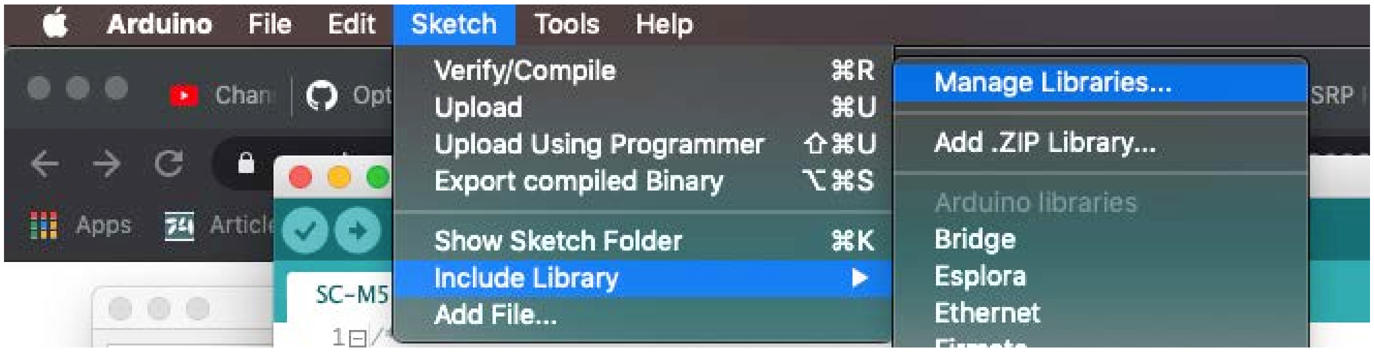
Where to start?

- Freshly bought M5Stickc with factory firmware: **Section 2, Download Device Firmware**
- Device already has the Synthesis @ ASU firmware installed and you want to configure the network settings: **Section 4, Configure Network Settings Via USB**
- Writing your own software to parse output from the device: **Section 5, Data Parsing Explanation and Examples**

2 Download Device Firmware & Example Software

I n this section we will detail which prerequisite software packages and libraries are necessary to update device firmware and configure the network parameters. Additionally, we will direct you to the example source files for handling the data output from the M5Stickc device. For many who are following this guide for the purpose of using Synthesis @ ASU software, these items will be all you need. In the case that you plan on using the M5Stickc device with environments outside of Max/MSP/Jitter, please see Appendix II for links to documentation for handling OSC input in other languages.

The firmware update procedure for the M5Stickc devices requires that users first download the Arduino environment. The Arduino software functions as an Integrated Development Environment for device firmware on the M5Stickc, Arduino's range of microcontrollers, and others. You can download the Arduino software from their website. Once you have the Arduino software downloaded and installed, you will need to download a two libraries, M5StickC & OSC. You can download these libraries by opening the Library Manager in Arduino under "Sketch>Include Library>Manage Libraries...".

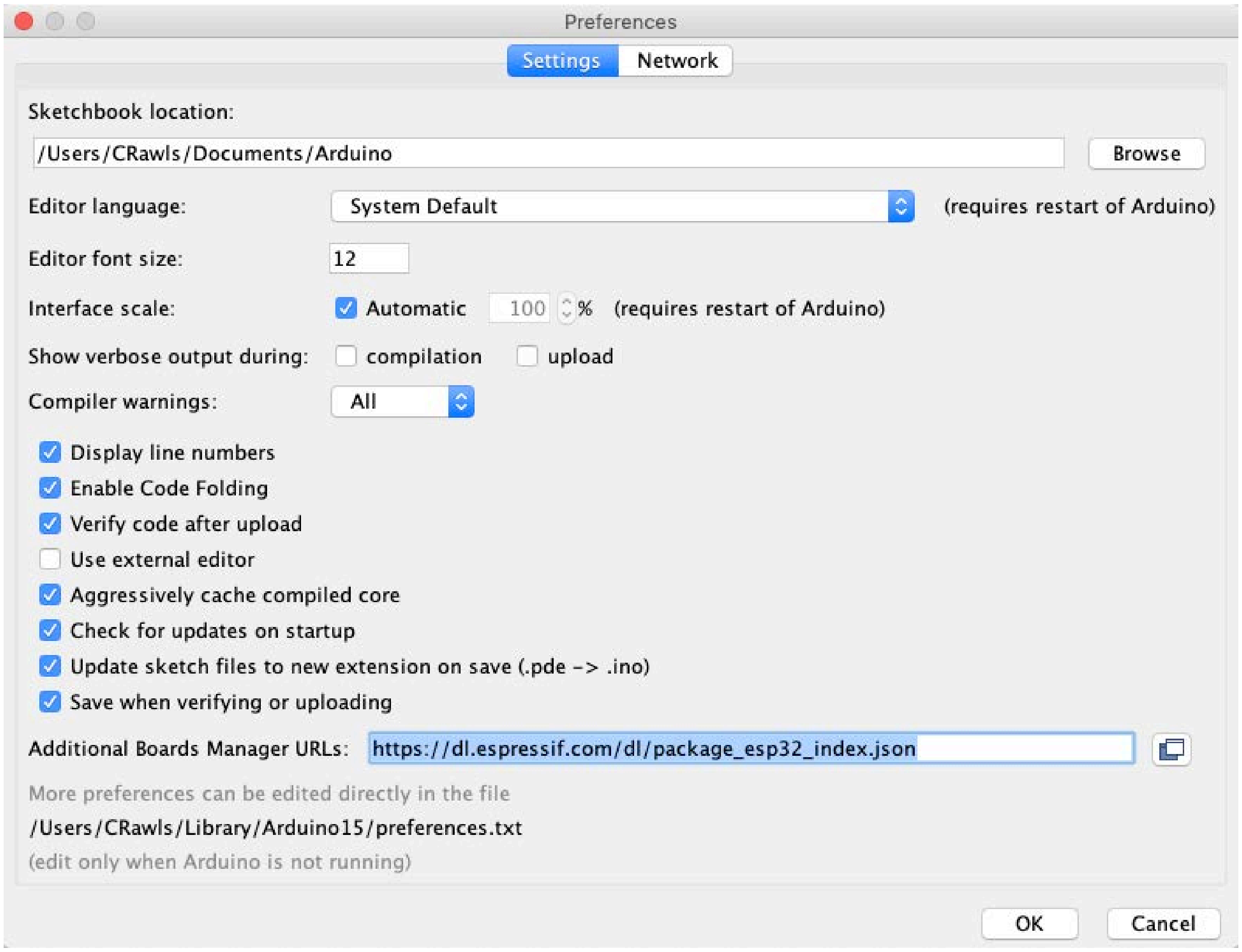


Menu Path in the Mac Environment, Same Path for Windows Users.

In addition to these libraries, you will also need to include the board manager for the M5StickC device. The board manager is the set of instructions that are run when you upload the firmware to the device that ensure that the incoming software is placed correctly in device memory. If incorrectly placed, the device may fail to run or become inoperable; using the correct manager is paramount to proper functioning.

To access the board manager download, you will first need to open the Arduino preferences window and add a URL to the "Additional Boards Manager URLs" box. The URL is for the ESP-32 microprocessor (which the M5 products are based on) which gives you access to the M5Stick-C board manager.

URL: https://dl.espressif.com/dl/package_esp32_index.json



Sketchbook location:
/Users/CRawls/Documents/Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 12

Interface scale: ✓ Automatic 100 % (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings: All

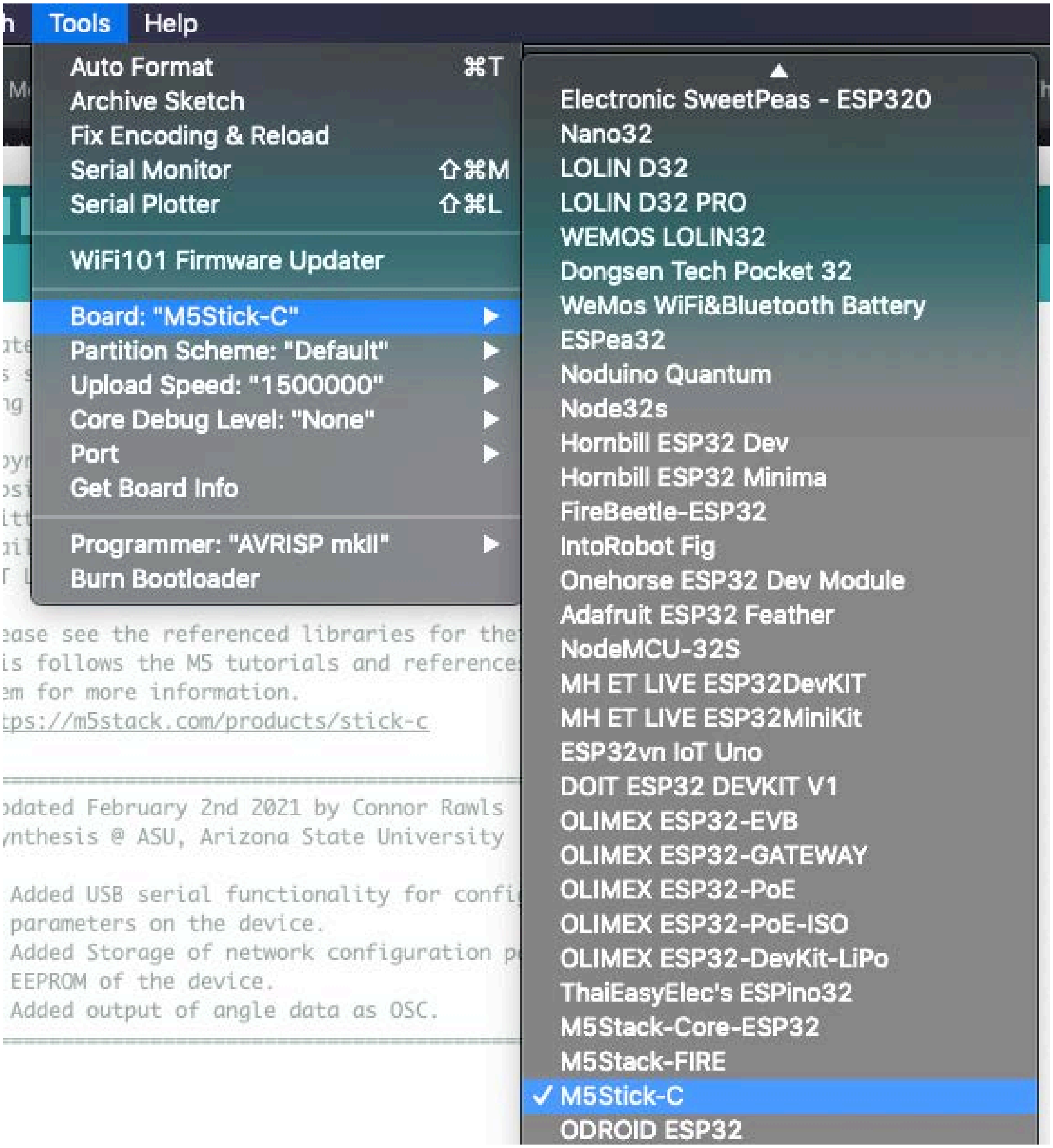
- ✓ Display line numbers
- ✓ Enable Code Folding
- ✓ Verify code after upload
- ☐ Use external editor
- ✓ Aggressively cache compiled core
- ✓ Check for updates on startup
- ✓ Update sketch files to new extension on save (.pde -> .ino)
- ✓ Save when verifying or uploading

Additional Boards Manager URLs: https://dl.espressif.com/dl/package_esp32_index.json

More preferences can be edited directly in the file
/Users/CRawls/Library/Arduino15/preferences.txt
(edit only when Arduino is not running)

Arduino Preferences Window with Board Manager URL highlighted. MacOS look & feel, layout should be similar across operating systems.

Once you have added the URL to the box in the preferences window, accept your changes and close the window. The next step is to download the Board Manager package for the ESP-32 set. You can add managers under "Tools>Board>Boards Manager...". Find the esp32 package in the Boards Manager window. At the time of writing, the version we have tested on is 1.0.4. You can try out newer versions of the esp32 boards manager, but compatibility is not guaranteed. Once the esp32 managers have been downloaded, restart Arduino. You should now have the option under "Tools>Board" to select "M5Stick-C".



Board: "M5Stick-C"
Partition Scheme: "Default"
Upload Speed: "1500000"
Core Debug Level: "None"
Programmer: "AVRISP mkII"
✓ M5Stick-C

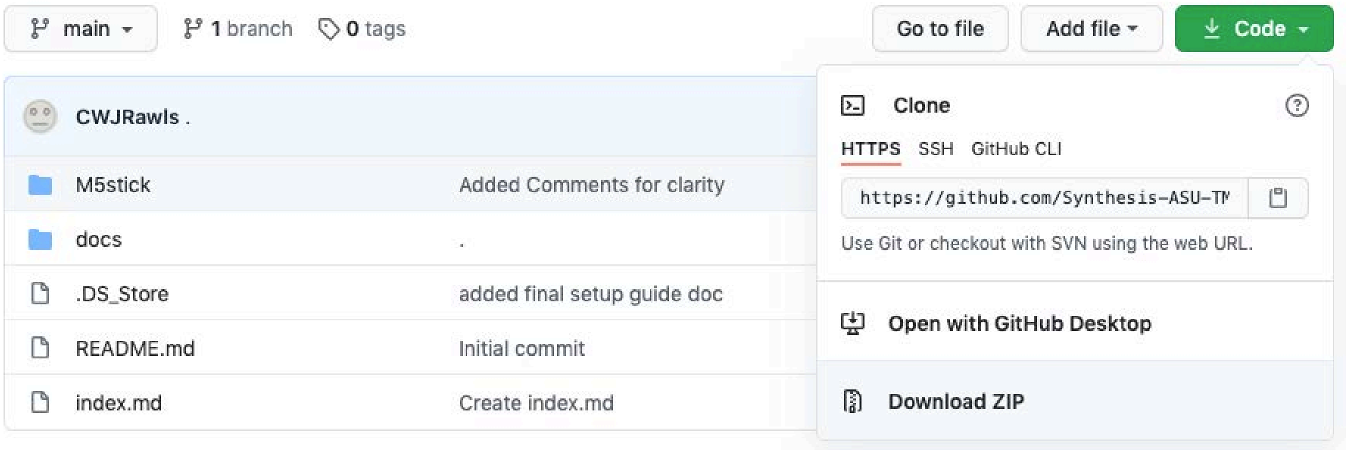
```
ease see the referenced libraries for the
is follows the M5 tutorials and reference
em for more information.
tps://m5stack.com/products/stick-c

===============================================
pdated February 2nd 2021 by Connor Rawls
ynthesis @ ASU, Arizona State University

 Added USB serial functionality for confi
 parameters on the device.
 Added Storage of network configuration p
 EEPROM of the device.
 Added output of angle data as OSC.
===============================================
```

Selecting M5Stick-C Board Manager on MacOS, Windows and Linux users should find the option under the same menu.

Now that Arduino is configured, we will download the Synthesis @ ASU firmware, configuration software, and example Max patches. You can find all of these at https://github.com/Synthesis-ASU-TML/Mated-Objects. We would suggest downloading the entire repository as a ZIP file, and just keeping the M5stick folder once decompressed.



M5stick	Added Comments for clarity
docs	.
.DS_Store	added final setup guide doc
README.md	Initial commit
index.md	Create index.md

Use Git or checkout with SVN using the web URL.

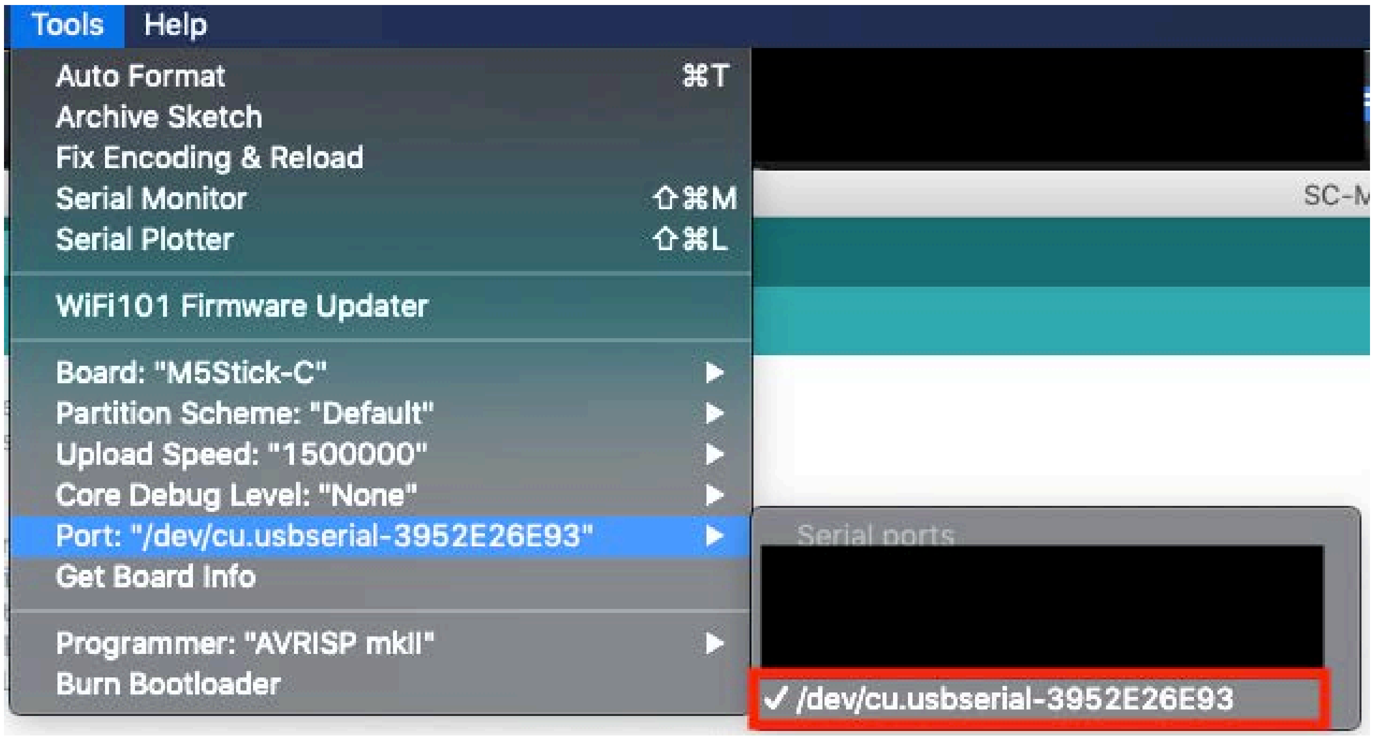
The web view of the Mated-Objects repository with the download menu opened from the green code button.

You can place the M5stick folder wherever is convenient on your file system. We suggest that you keep the contents of the folder together to avoid issues with Arduino compatibility.

3 Installing Firmware to M5StickC Device

Now that the necessary software and packages have been downloaded and installed, we will begin the process of putting the Synthesis @ ASU firmware on the M5StickC device. Start by opening SC-M5-Stick.ino in the Arduino IDE. Next, select the M5Stick-C board manager (see page 6 for details). Most of the programming details should update automatically.

After you have set the board manager to M5Stick-C, you need to select the port of the device. Depending on your operating system, the ports will appear differently in the Arduino IDE. On MacOS the name will look similar to "/dev/cu.usbserial-hexstring". On Windows the device will be listed by COM port number. Since the devices will appear under generic names on both systems, it is recommended to only connect one at a time to avoid confusion.



Board: "M5Stick-C"
Partition Scheme: "Default"
Upload Speed: "1500000"
Core Debug Level: "None"
Port: "/dev/cu.usbserial-3952E26E93"
Programmer: "AVRISP mkII"

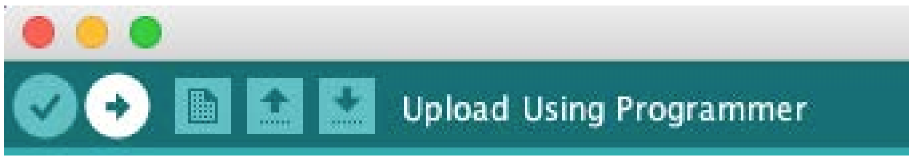
[redacted]

[redacted]

✓ /dev/cu.usbserial-3952E26E93

The view of the "Port" menu in Arduino with the M5StickC device selected. Shown on MacOS. For Windows users, the "Port" menu will give you a list available COM ports. Will appear similar to "COM #".

Once you have you board and port configured, press the right-facing arrow button to compile and upload the firmware to the M5StickC device. This process will take a few minutes and will display "Done Uploading" in the bottom of the window when done.



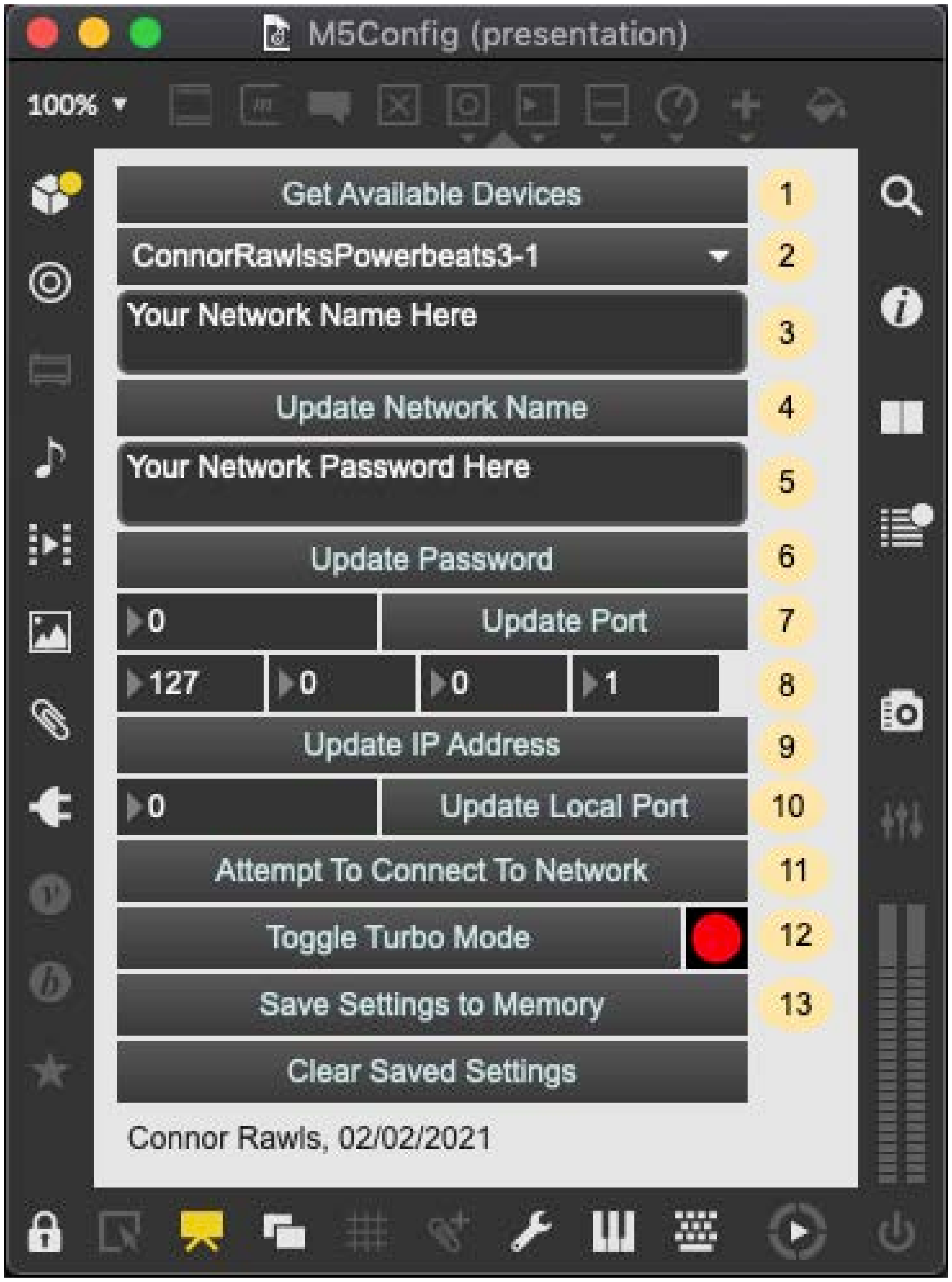
The Upload button in the Arduino IDE highlighted. Shown in MacOS, layout is the same on Windows.

In the case that you encounter an error in uploading, double check that the settings for Port and Board are correct. If you have any other M5StickC devices or other programmable micro-controllers disconnect them to avoid port confusion.

4 Configuring Network Settings

In this section we will cover how to use the provided M5Config Max patch to set how the M5StickC device connects to a WiFi network and setting the IP Address and port for sending data. Anytime you change target computer, or the computer's IP Address changes you will need to repeat at least a portion of this process.

Before opening the M5Config Max patch, make sure to quit the Arduino IDE entirely. Not doing so can result in the IDE blocking communication to the M5StickC device from other applications. Next find M5Config.maxpat in the M5stick folder and open it with Max 8. If you do not have Max installed on your computer, check Appendix I if you want to write your own configuration software. Otherwise, we will provide a compiled version of M5Config in the future. Once the patch has opened, see below for descriptions of each step.


M5Config (presentation)

100%

Get Available Devices 1

ConnorRawlssPowerbeats3-1 2

Your Network Name Here 3

Update Network Name 4

Your Network Password Here 5

Update Password 6

0 Update Port 7

127 0 0 1 8

Update IP Address 9

0 Update Local Port 10

Attempt To Connect To Network 11

Toggle Turbo Mode 12

Save Settings to Memory 13

Clear Saved Settings

Connor Rawls, 02/02/2021

The M5Config Max patch, shown on MacOS layout is the same on Windows systems.

M5Config Descriptions:

1 - Get Available Devices

Updates drop down menu with the updated list of all serial devices currently connected to the computer. Displays the port similarly to the Arduino IDE.

2 - Select Device

Using the drop down menu, select the M5StickC device that you want to configure.

3 - Network Name Box

Enter the name of the WiFi network that you want the M5StickC device to connect to.

4 - Update Network Name Button

Sends the network name to the M5StickC device.

5 - Network Password Box

Enter the password for the WiFi network that you want the M5StickC device to connect to.

6 - Update Network Password Button

Sends the network password to the M5StickC device.

7 - Port Update

Set the port number to use on the left and then press the button on the right to send the new network port to the M5StickC device. We suggest that you use the port 8001, ports above 1000 tend to be open in the case that you want to use a different port.

8 - Enter Target IP Address

Enter the four parts of the IP Address from the computer that you want to send to. See Appendix III for how to find the IP Address of your computer.

9 - Update IP Address Button

Sends the updated target IP Address to the M5StickC device.

10 - Local Port Update

Sets the network port for return traffic to the M5StickC device. In future updates there may be the possibility of configuration over WiFi for some operation parameters. By default this port is set to 7500. You can change this value, but currently has no effect.

11 - Attempt To Connect To Network

Confirm that the connection details are correct before pressing this button. Once pressed, the button tells the M5StickC to attempt to connect to WiFi using the data sent to it. Check the network display screen on the device to see if the M5StickC has been successful in connecting to WiFi.

12 - Toggle Turbo Mode

Pressing the button toggles the device between standard operation and turbo operation. Turbo mode sets the M5StickC to run as fast as possible, otherwise the device runs at 25FPS. When turbo mode is active, the circle to the right of the button will turn green.

13 - Save Settings To Memory

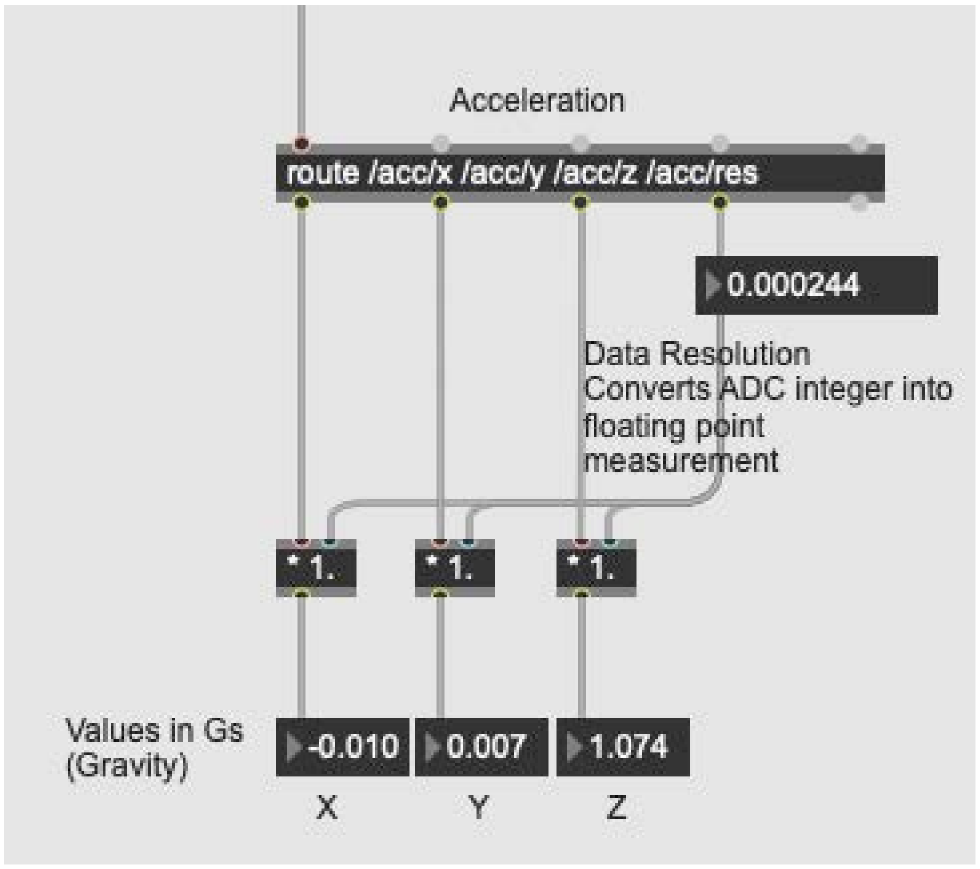
If you are happy with the configuration of the M5StickC, press this button to tell the device to commit your settings to internal memory. Doing this will make the settings available to the device on reset rather than reverting to default values.

Clear Saved Settings

Only use this button if you intend to erase ALL of the previously saved settings. You can overwrite specific values with the controls above.

5 Data Parsing Examples

We have two different Max patches as examples of how to parse the data passed from the M5StickC device. They both show how treat the data to get values that match what is shown on the device display only differing in the set of Max objects used to dispatch the incoming OSC messages.

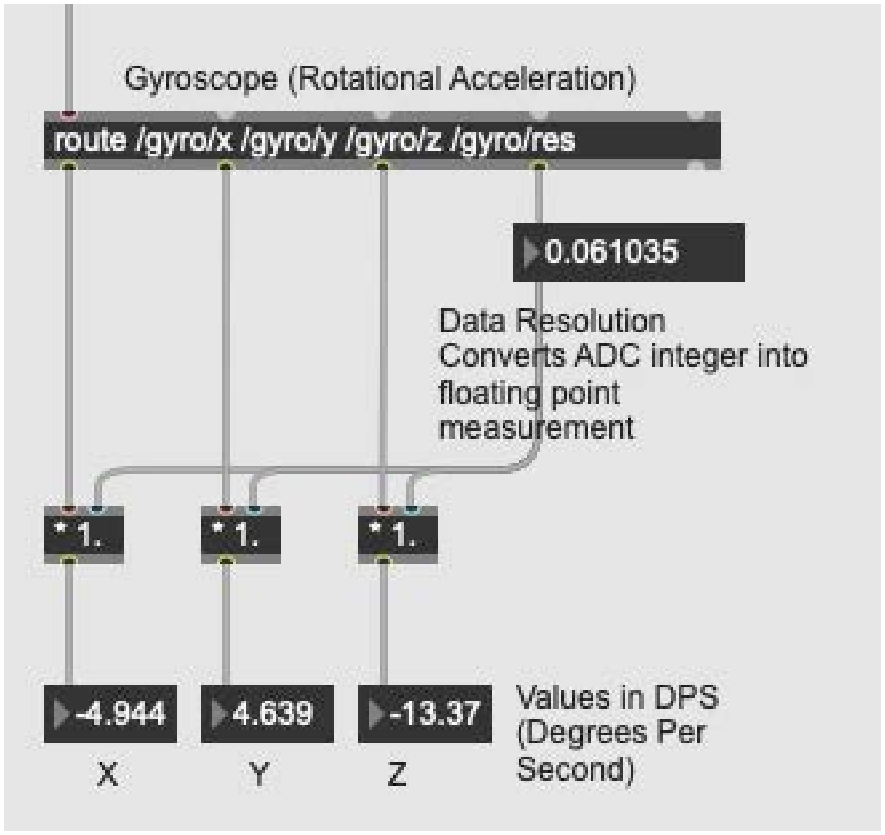


Example of how to route and parse incoming accelerometer data in Max.

Accelerometer Data

The accelerometer in the M5StickC reports sensing data in units of G (or Gravity). A value of 1G is equivalent to the device laying still on a surface, experiencing the normal force of Earth's gravity.

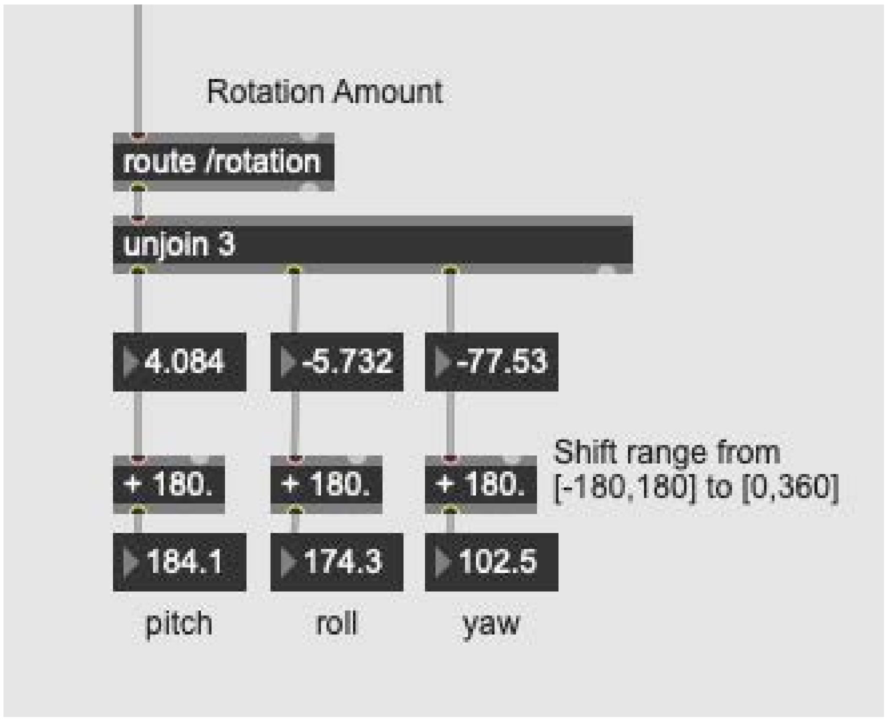
Values greater than 1 are that many times the normal gravity of Earth, meaning that a reading of 2 is 200% the gravitational force of Earth. Values less than 1 but greater than 0 mean the sensor is experiencing less than the normal force of gravity in that direction. Values less than 0 follow the same rules, but indicate that the opposite face of the sensor is pointing towards the direction of motion or the ground. By default the M5StickC accelerometer is set to a range of ±20G.


Gyroscope (Rotational Acceleration)
route /gyro/x /gyro/y /gyro/z /gyro/res
0.061035
Data Resolution
Converts ADC integer into floating point measurement
* 1.
* 1.
* 1.
-4.944
4.639
-13.37
Values in DPS (Degrees Per Second)
X
Y
Z

An example of routing and parsing the gyroscope data in Max.

Gyroscope Data

The gyroscope on the M5StickC reports the rotational acceleration of the device in degrees per second (DPS). This value states the amount in degrees that the device will rotate if continuously spinning for one second. This data is signed, meaning that positive values mean the device is rotating one direction on that specific axis. A negative value means that the device is spinning in the opposite direction on that axis. By default, the M5StickC gyroscope is configured to a sensitivity of ±2000DPS. This limit is unlikely to be reached by humans, but is possible when attached to mechanical objects.



An example of routing and parsing the rotational values reported by the M5StickC.

Rotation Data

The M5StickC is also capable of reporting the estimated current rotation of the device. Unlike the gyroscope, this measure is an estimation by the IMU of rotationally how far from the internal 0 point the device is currently. Since these values are an estimation based on sensor input rather than direct measurements, there is the possibility of drift if any of the sensor elements reports faulty readings. By default these values are reported in the range of ±180 degrees. The example above shows a quick method of shifting the range completely to positive values.

OSC Message Addresses

Below is a listing of the OSC Messages that the data packets will be sent on. You can use this for either writing your own parser with Max or any other language of choice.

Acceleration Data

/acc/x

The X component of the acceleration data. Will arrive as a 16-bit signed integer. Use the data resolution value to parse back Gs.

/acc/y

The Y component of the acceleration data. Will arrive as a 16-bit signed integer. Use the data resolution value to parse back Gs.

/acc/z

The Z component of the acceleration data. Will arrive as a 16-bit signed integer. Use the data resolution value to parse back Gs.

/acc/res

The data resolution value for acceleration data. Use the formula (res * component) to transform the signed component integers to floating point G units.

Gyroscopic Data

/gyro/x

The X component of the gyroscopic data. Will arrive as a 16-bit signed integer. Use the data resolution value to parse back to DPS.

/gyro/y

The Y component of the gyroscopic data. Will arrive as a 16-bit signed integer. Use the data resolution value to parse back to DPS.

/gyro/z

The Z component of the gyroscopic data. Will arrive as a 16-bit signed integer. Use the data resolution value to parse back to DPS.

/gyro/res

The data resolution value for gyroscopic data. Use the formula (res * component) to transform the signed component integers to floating point DPS units.

Rotation Data

/rotation

Reports an array of length 3 of signed floating point values. All the values are in the range of ±180 degrees. There is no data resolution necessary for these values, they are already treated on transmission.

Appendix I: Configuration Serial Controls

In this appendix we will list the serial commands used in the M5Config Max patch to communicate with the M5StickC for configuration. As is standard in serial communication, all data is passed as a series of unsigned 8-bit integers. In this scheme all whitespace is important and all commands with additional arguments require a space character (value 32) between the 3 letter command string and the parameters.

UID <Network Name String>

Used to send the name of the network you want the device to join. Any trailing whitespace characters will be included, null terminators will be ignored by the device.

PWD <Network Password String>

Used to send the password for the network you want the device to join. Any trailing whitespace characters will be included, null terminators will be ignored.

IPA <IP Address String>

Used to send the target IP Address for data transmissions, expects the address string in the form "255.255.255.255" sans the quotations. Currently the string is not error checked on reception and may result in transmission errors in the case of unexpected characters in the string.

PRT <Port# / 255><Port# % 255>

Sends the target network port value to the device. The device will reconstruct the 16-bit unsigned port value on reception. DO NOT PUT WHITESPACE BETWEEN ARGUMENTS! Errant space characters will result in the wrong port being used.

LPT <Port# / 255><Port# % 255>

Sends the local network port value to the device. The device will reconstruct the 16-bit unsigned port value on reception. DO NOT PUT WHITESPACE BETWEEN ARGUMENTS! Errant space characters will result in the wrong port being used.

TRB

Toggles turbo mode, requires no arguments.

ATT

Tells the device to attempt to connect to the network using the current settings. Will write back current connection status if successful.

SAV

Tells the device to write the current settings to internal EEPROM (or non-volatile memory). This command will cause an overwrite of any previously stored settings.

CLR

Clears ALL stored settings from EEPROM. Only use this command in the case that you intend to fully wipe stored configuration parameters.

DSP

Not shown in M5Config patch, causes the device to write back the current stored network name and password. USE RESPONSIBLY

PNG

Not Shown in M5Config patch, causes the device to flash a color on screen. Useful for identifying which unit you are currently configuring in the case that multiple are connected. Also useful when you believe the device has become unresponsive.

Appendix II: Handling OSC

In this appendix we will provide some direction for those seeking to develop with our firmware in languages other than Max.

C++: oscpp

A header only library that is cross-compatible in C++11 environments.

Link: https://github.com/kaoskorobase/oscpp

C#: UnityOSC

A Unity specific implementation of the OSC standard. Built specifically for the Unity engine.

Link: https://github.com/jorgegarcia/UnityOSC

C#: SharpOSC

A general use implementation of the OSC standard. Built with .NET in mind.

Link: https://github.com/ValdemarOrn/SharpOSC

Python: python-osc

A pure Python implementation of OSC, can handle everything from building messages and bundles to running client and server UDP sockets.

Link: https://osc.readthedocs.io/en/latest/

Swift: SwiftOSC

An implementation of OSC as a Swift compatible framework.

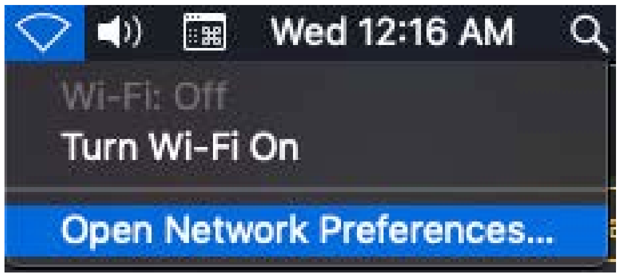
Link: https://github.com/ExistentialAudio/SwiftOSC

Appendix III: Finding Your IP Address

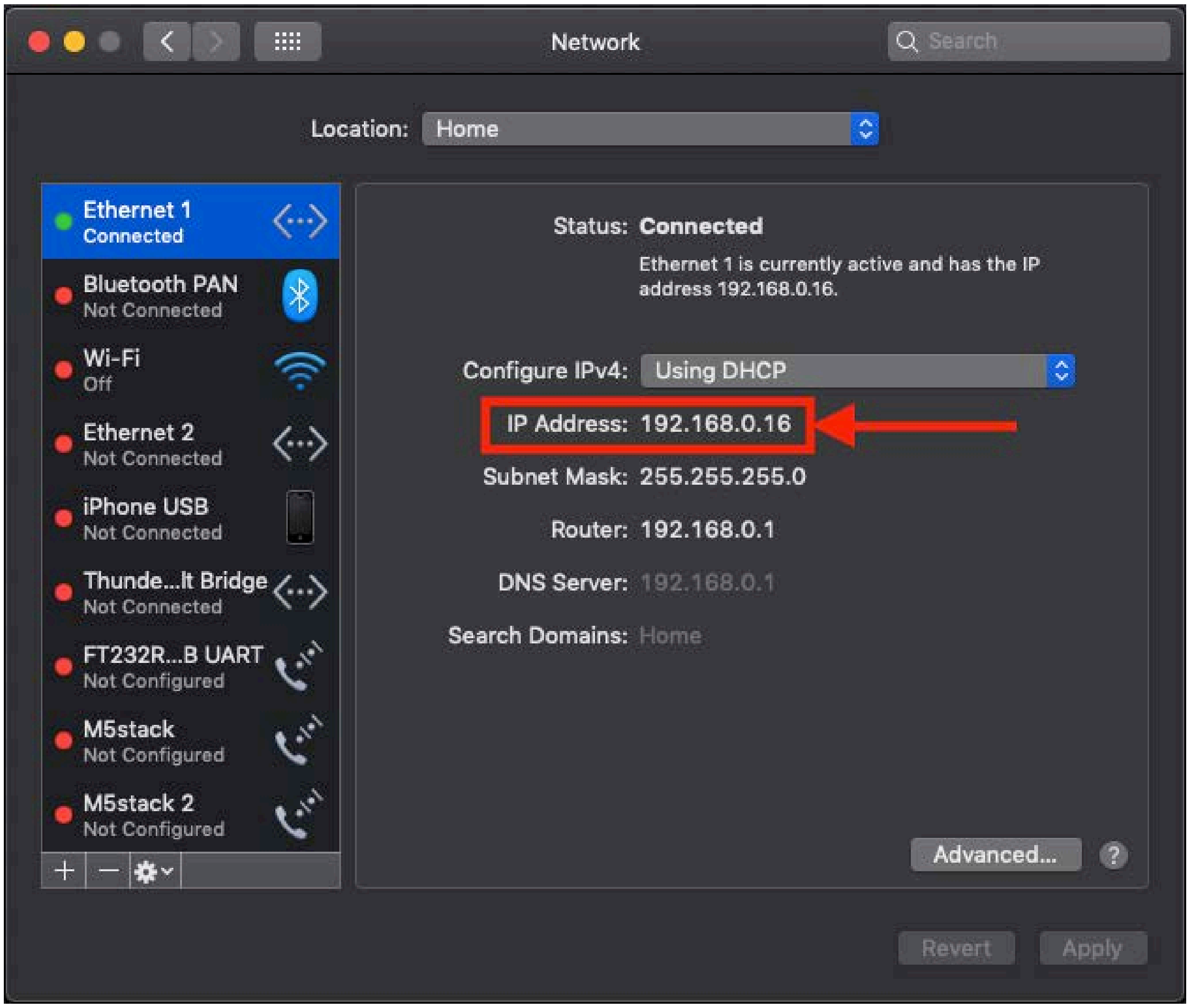
In this appendix we will cover how on MacOS and Windows to find the current IP Address of your computer.

MacOS:

- Access Network Preferences by selecting the WiFi icon in the top-right corner of the screen. In the drop down menu select "Open Network Preferences...".
- Choose the interface that you are connected to on the right of the window. If your network has internet access it will often be moved to the top of the list automatically. In the screenshot below the example computer is connected via ethernet, so those settings are displayed.
- Find the IP Address Line, this section shows you the current address of your computer on the network. Put these numbers into the M5Config patch to set your IP Address.



An example on MacOS of how to quickly access Network Preferences



Location: Home

Ethernet 1	Connected
Bluetooth PAN	Not Connected
Wi-Fi	Off
Ethernet 2	Not Connected
iPhone USB	Not Connected
Thunde...lt Bridge	Not Connected
FT232R...B UART	Not Configured
M5stack	Not Configured
M5stack 2	Not Configured

Status: **Connected**

Ethernet 1 is currently active and has the IP address 192.168.0.16.

Configure IPv4: Using DHCP

IP Address: 192.168.0.16

Subnet Mask: 255.255.255.0

Router: 192.168.0.1

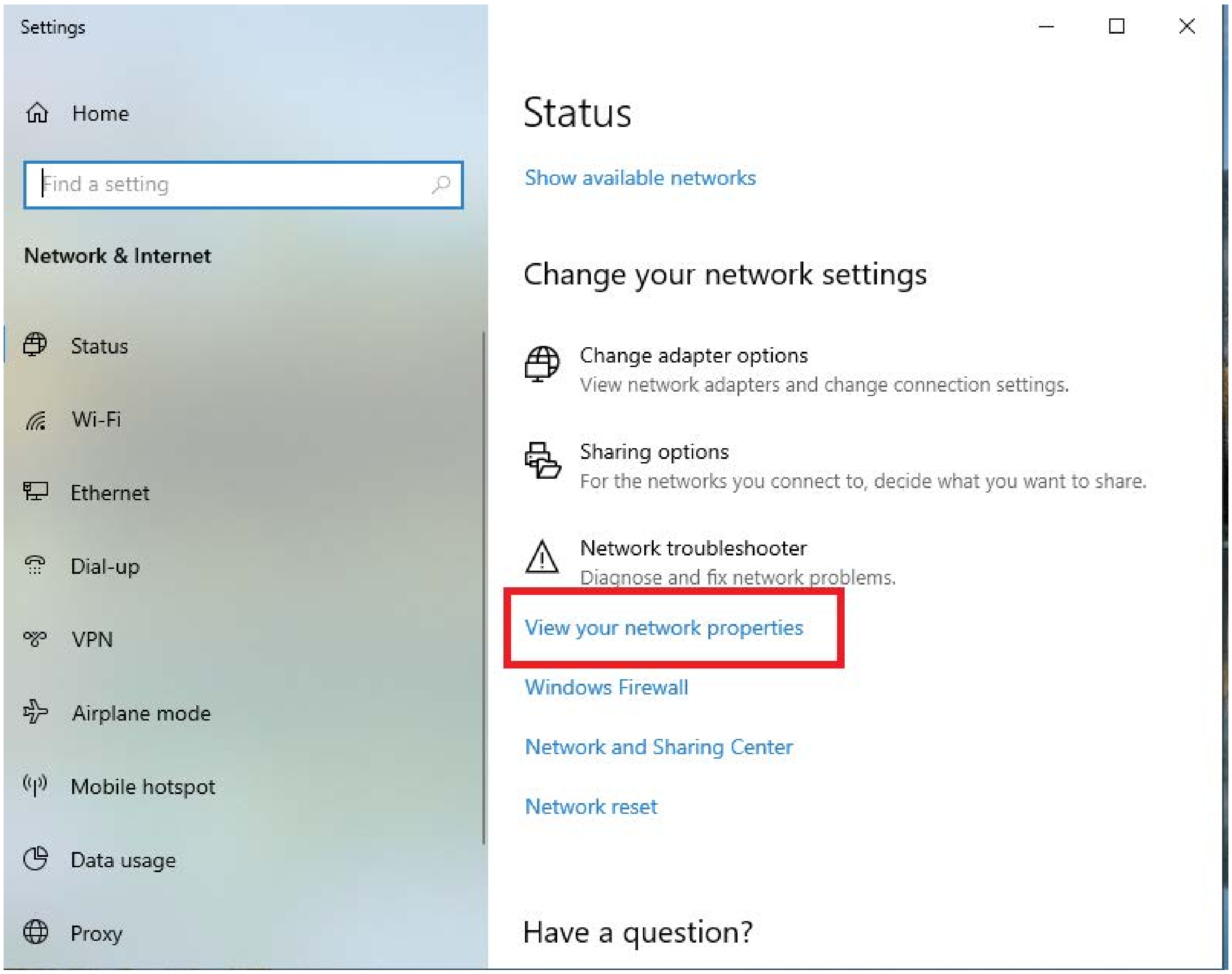
DNS Server: 192.168.0.1

Search Domains: Home

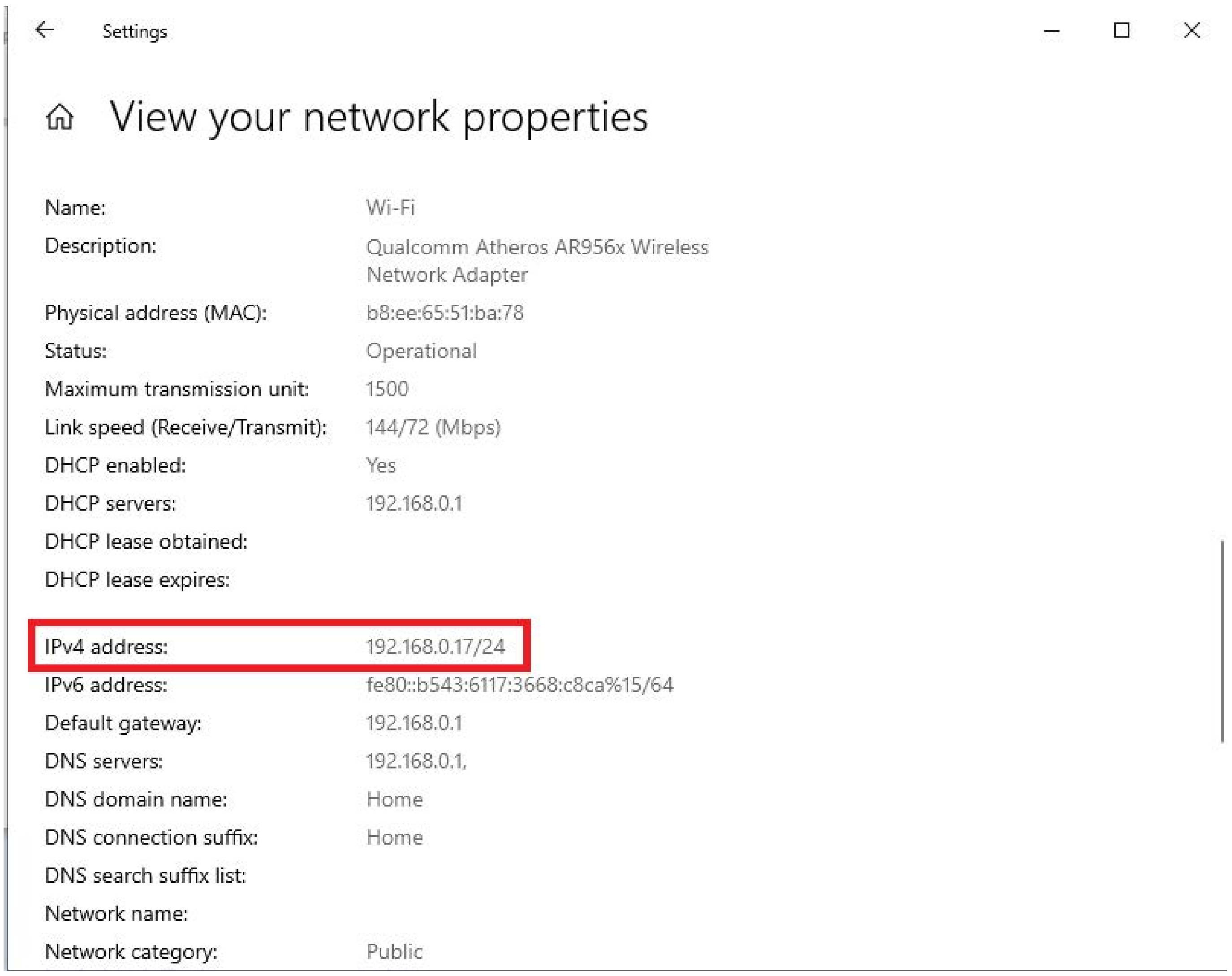
An example of the Network Preferences window on MacOS. The IP Address section is highlighted.

Windows

- Open Control Panel and find "Network Status". Alternatively, use Cortana to search for "Network Status".
- Once you have opened the Network Status panel, click on "View Network Properties" to view the status of your various network interfaces.
- Find the interface you are using to connect to the network and look for the IPv4 Address entry. The address will be shown in a form similar to "255.255.255.255\24". Use the first four numbers in the M5Config patch, the "\24" can be ignored.



Windows Network Status control panel. The "View your network properties" button is highlighted.


Settings

View your network properties

Name:	Wi-Fi
Description:	Qualcomm Atheros AR956x Wireless Network Adapter
Physical address (MAC):	b8:ee:65:51:ba:78
Status:	Operational
Maximum transmission unit:	1500
Link speed (Receive/Transmit):	144/72 (Mbps)
DHCP enabled:	Yes
DHCP servers:	192.168.0.1
DHCP lease obtained:	
DHCP lease expires:	
IPv4 address:	192.168.0.17/24
IPv6 address:	fe80::b543:6117:3668:c8ca%15/64
Default gateway:	192.168.0.1
DNS servers:	192.168.0.1,
DNS domain name:	Home
DNS connection suffix:	Home
DNS search suffix list:	
Network name:	
Network category:	Public

The Windows network properties view. The IPv4 address line is highlighted. This computer is connected via WiFi, and so that interface is shown. The view of the other disconnected interfaces is omitted for clarity.