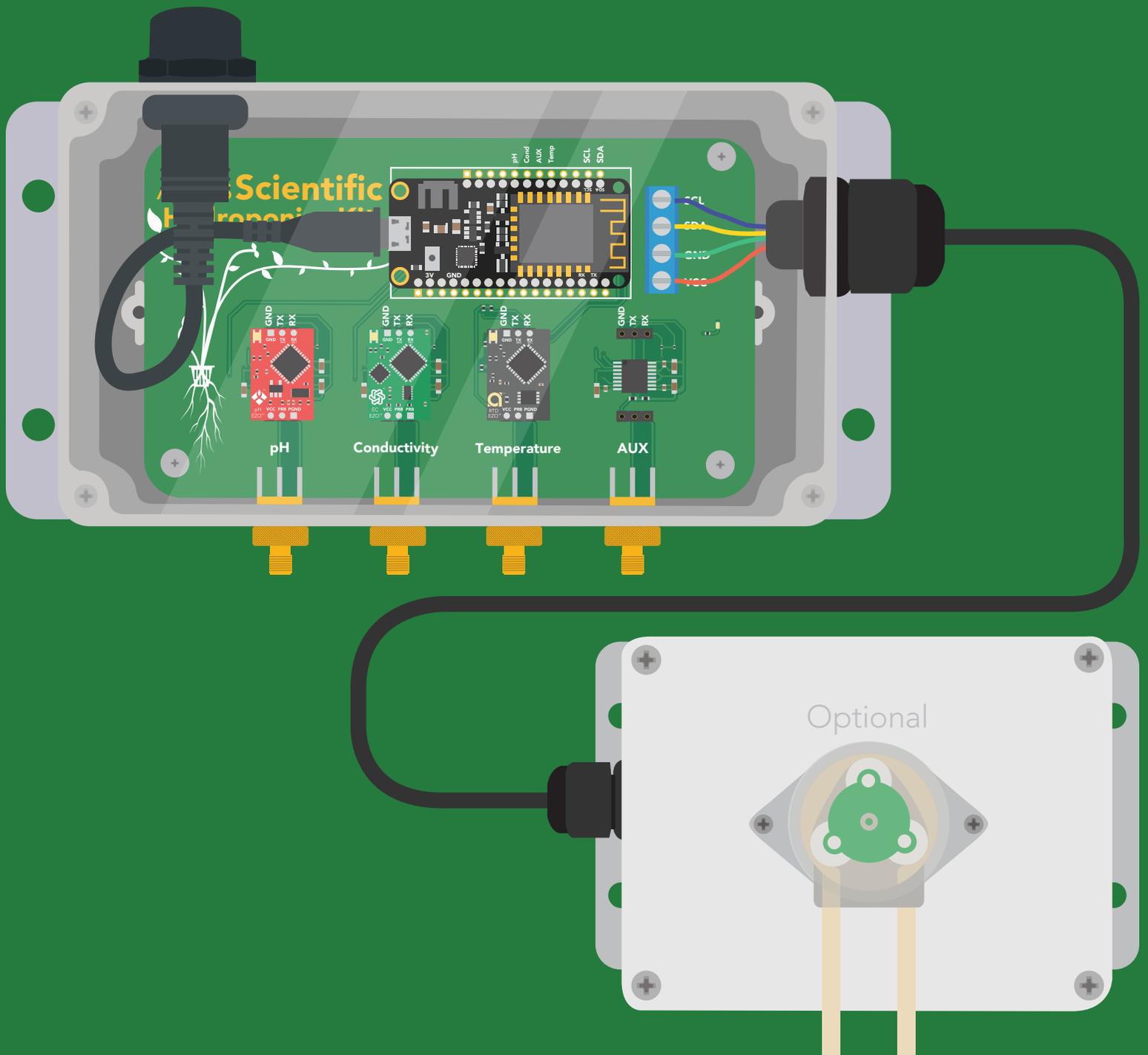


Wi-Fi Hydroponics Kit

Datasheet

V 2.0

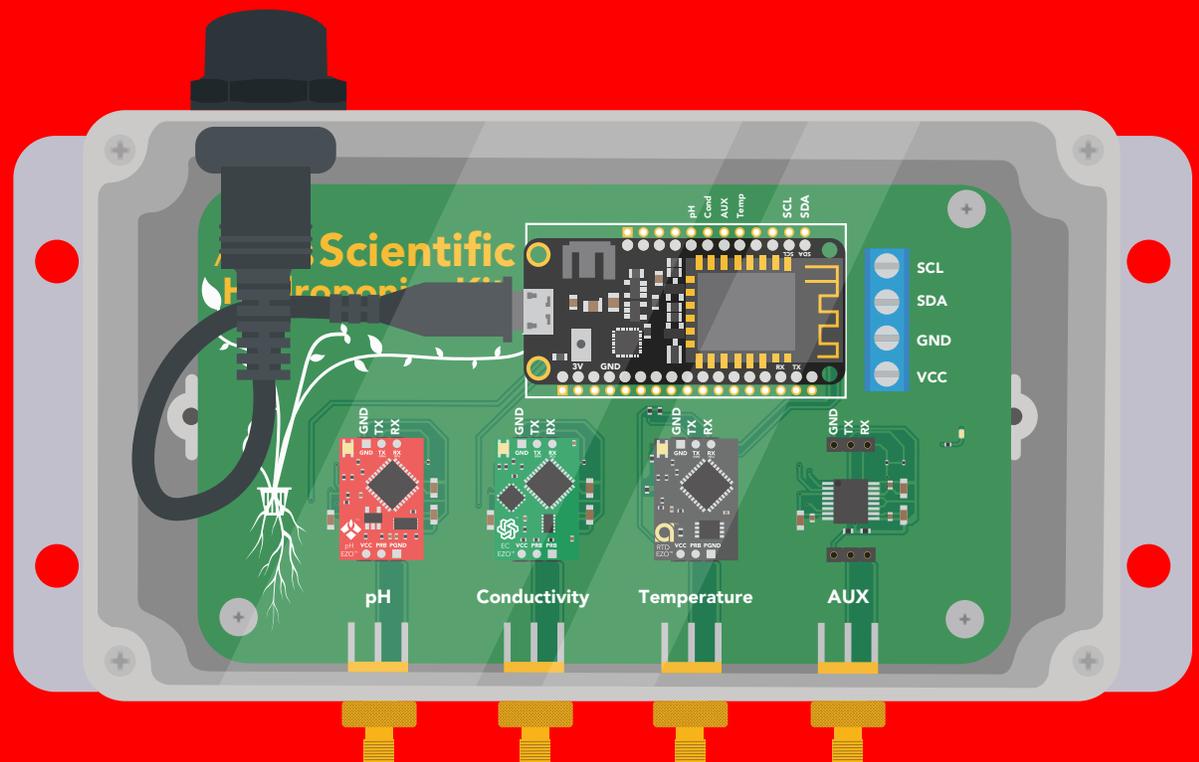


STOP

Atlas Scientific does not make consumer electronics.

This equipment is intended for electrical engineers. If you are not familiar with electrical engineering or embedded systems programming, this product may not be for you.

This device was developed and tested using a Windows computer. It was not tested on Mac, Atlas Scientific does not know if these instructions are compatible with a Mac system.



IP64

(dust and water splash proof)

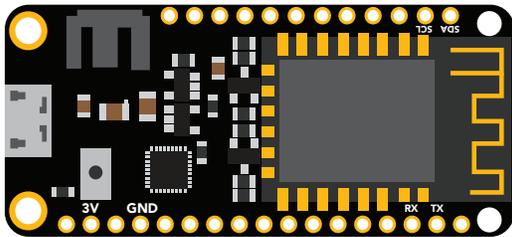
Operating principle

The Wi-Fi hydroponics kit has been designed to provide the engineer with a simple way of remotely monitoring and controlling a hydroponics system's chemistry. Sensor data is uploaded to ThingSpeak™, a free, cloud-based data acquisition and visualization platform. The Wi-Fi hydroponics kit has also been designed to be easily modified by the engineer. Feel free to change the sensors or functionality of the device to meet your specific needs.

Overview

CPU

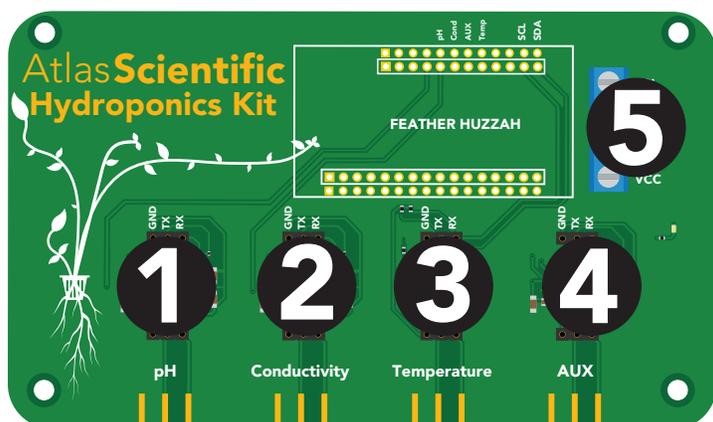
The Wi-Fi hydroponics kit is controlled using an Adafruit HUZZAH32 as its CPU. The HUZZAH is programmed using the Arduino IDE and uses an onboard ESP32 as its Wi-Fi transmitter. [Adafruit HUZZAH32 datasheet.](#)



Sensor ports

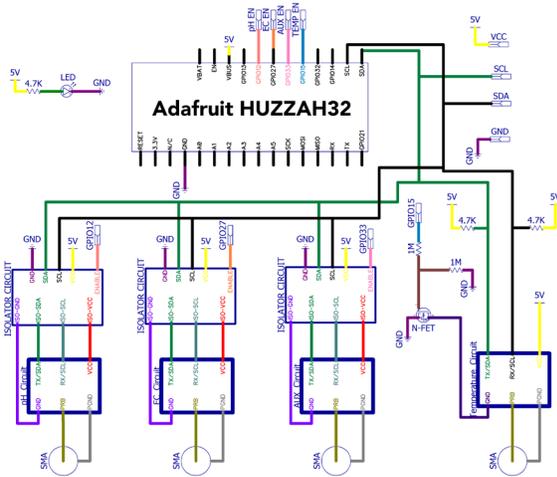
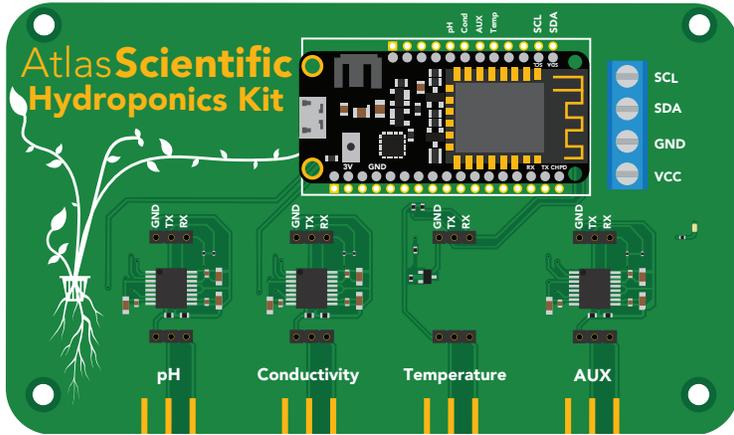
The Wi-Fi hydroponics kit PCB has 5 sensor ports. Three of the ports are electrically isolated. The isolated ports are marked pH, Conductivity, and AUX. The isolated ports are needed to take noise-free electrochemical readings. Because the sensing element of a temperature sensor is never in direct contact with the water, electrical isolation is not needed for temperature sensing.

The AUX port can be used to add an additional sensor of your choice. The terminal block marked Port 5 has been designed to connect one or more dosing pumps to the device. However, the port could also be used to connect a gas sensor.

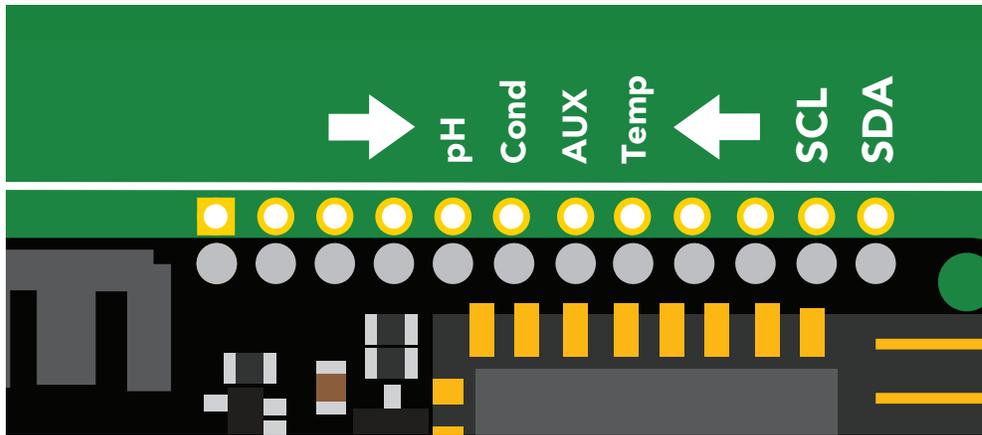


PCB

The overall design of the PCB is quite simple. The CPU is powered and programmed through the panel-mount USB connector. The CPU's USB pin supplies the board's power bus with 5V.



Each of the four main sensor ports have an enable pin, which must be set correctly to power the sensor. The enable pins are found here:



The first three pins (pH, Cond and Aux) must be set low to power on the sensor. The last pin (Temp) must be set high to power on the sensor.

Truth table

Pin	Adafruit Huzzah32 GPIO	State	Sensor Power
pH	12	LOW	ON
Cond	27	LOW	ON
Aux	33	LOW	ON
Temp	15	HIGH	ON

Sensor port 5 (the terminal block) does not have an enable pin and can not be turned off.

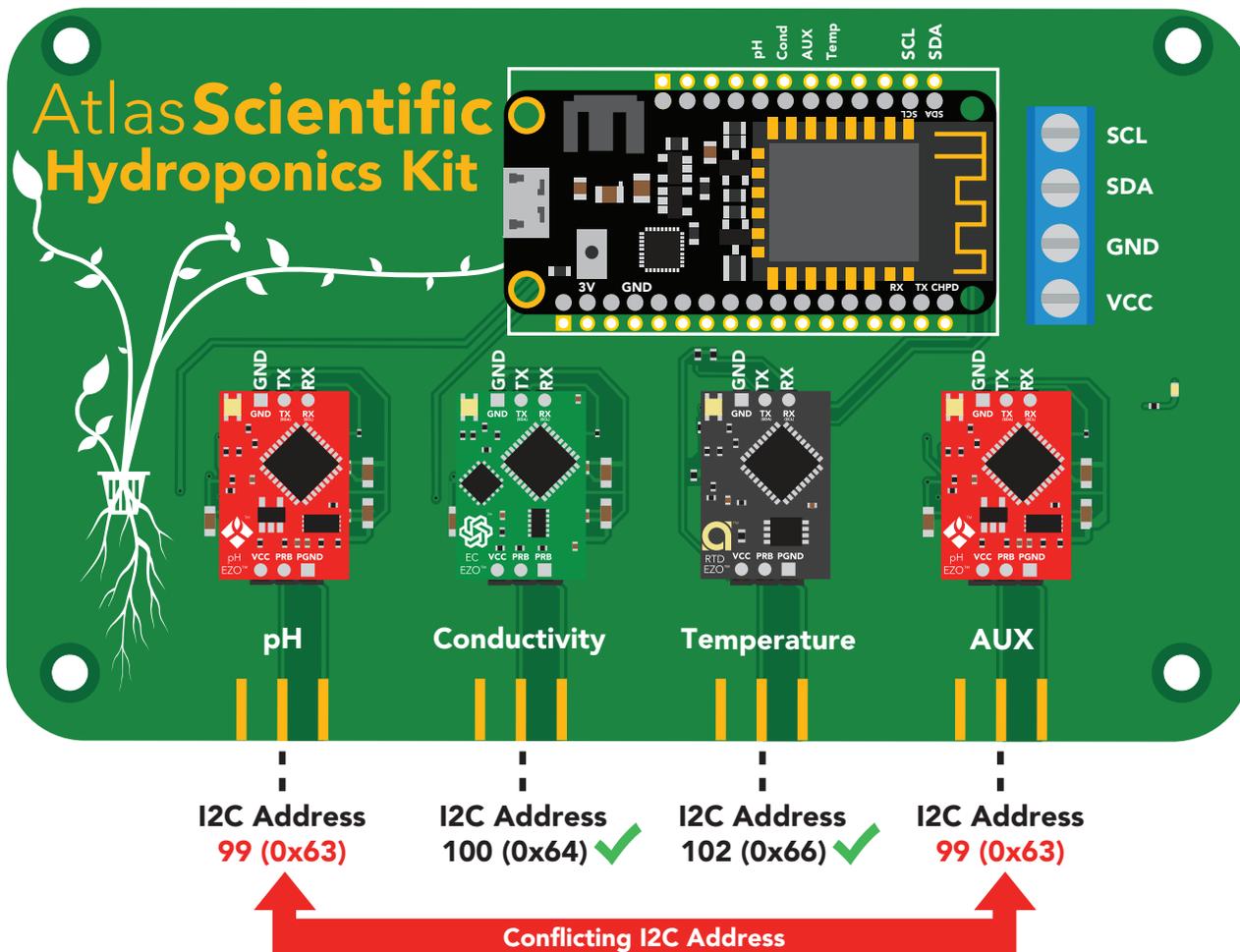
Data protocol

The CPU communicates with all peripheral sensors using the I2C data protocol. All data lines are directly connected to the CPU's I2C port. Using a different data protocol with this circuit board is not possible.

It is important to keep in mind that all Atlas Scientific components default to UART mode. When adding a new Atlas Scientific component to the kit, it must first be put into I2C mode. Refer to the component's datasheet for instructions on how to switch it over.

Adding more of the same sensor or component type

Adding additional components of the same type, such as an additional pH or conductivity sensor, is not hard to do. As mentioned above, you must set the device to I2C mode, and you must make sure that its I2C address is not the same as the already existing component.

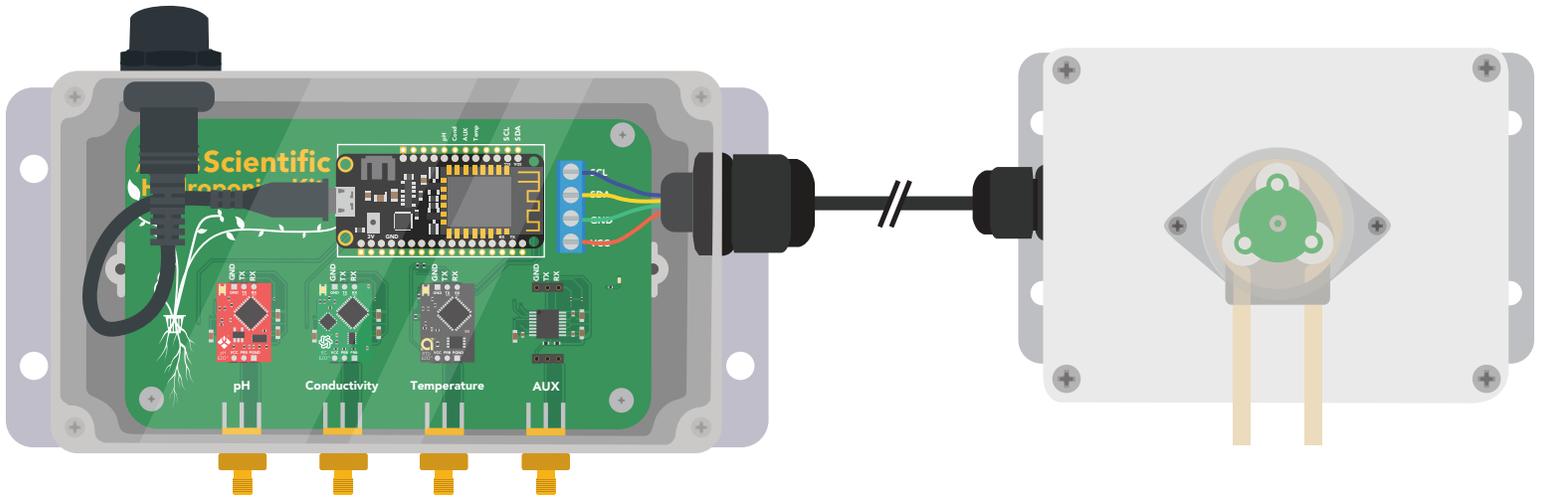


This table lists the default I2C address of components commonly added to this kit.

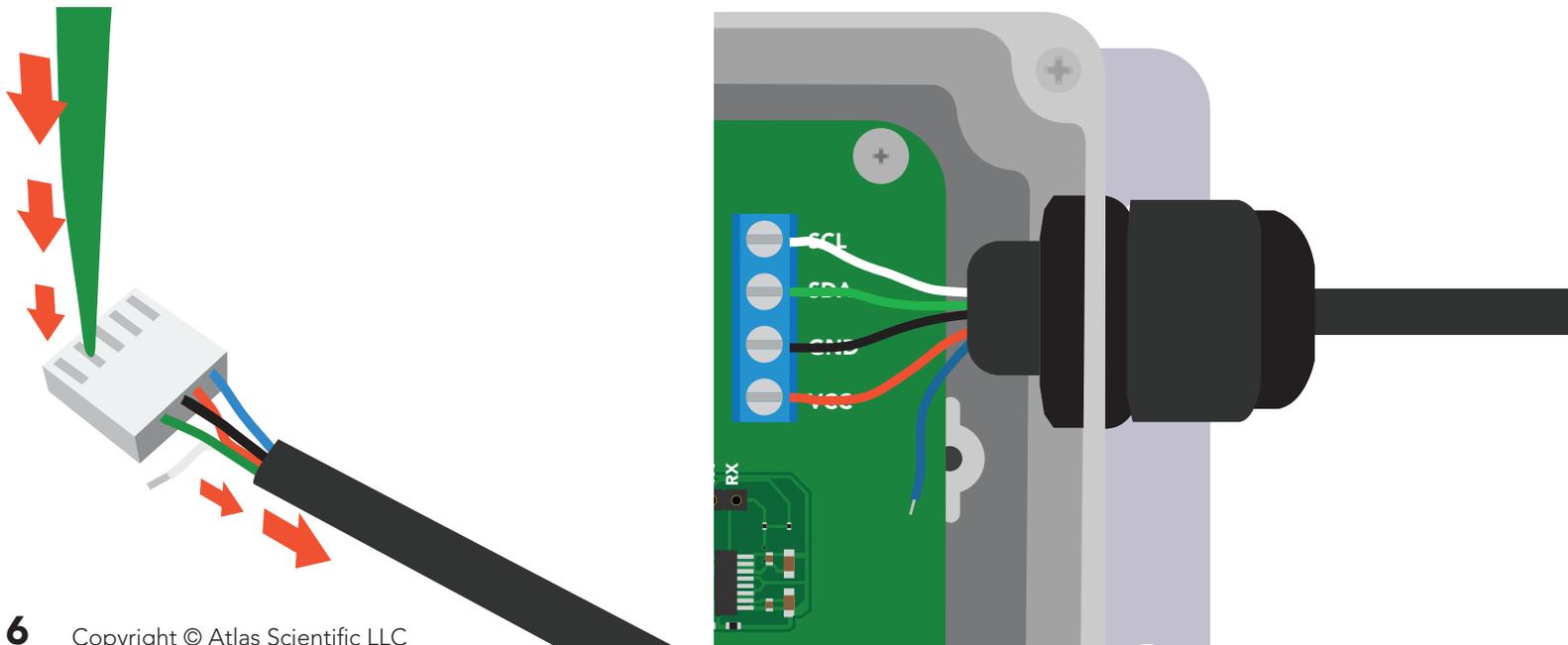
Device	I2C Address	Device	I2C Address
EZO pH	99 (0x63)	EZO EC	100 (0x64)
EZO ORP	98 (0x62)	EZO RTD	102 (0x66)
EZO DO	97 (0x61)	EZO PMP	103 (0x67)

Dosing pump

An optional external dosing pump can be added to the Wi-Fi hydroponics kit. Using the [SGL-PMP-BX](#) is the simplest way to add on a dosing pump.



A stand-alone EZO-PMP can be used instead of the expansion pump kit; however, you must manually put the pump in I2C mode and remove the data cable connector.



Uploading sensor data to the cloud

The Atlas-Scientific Wi-Fi hydroponics kit has been designed to upload sensor data to ThingSpeak™, a free, cloud-based data acquisition and visualization platform. You will be required to set up a free account with ThingSpeak™ to upload and visualize the data. With a free account, you can upload data once every 15 seconds. A paid account lets you upload data once per-second; look [here](#) for more info about various ThingSpeak™ services.

Atlas Scientific has no business relationship with ThingSpeak™; we just like how it works. If you want to use a different service, modify the device as you see fit.

Setting up your Wi-Fi kit

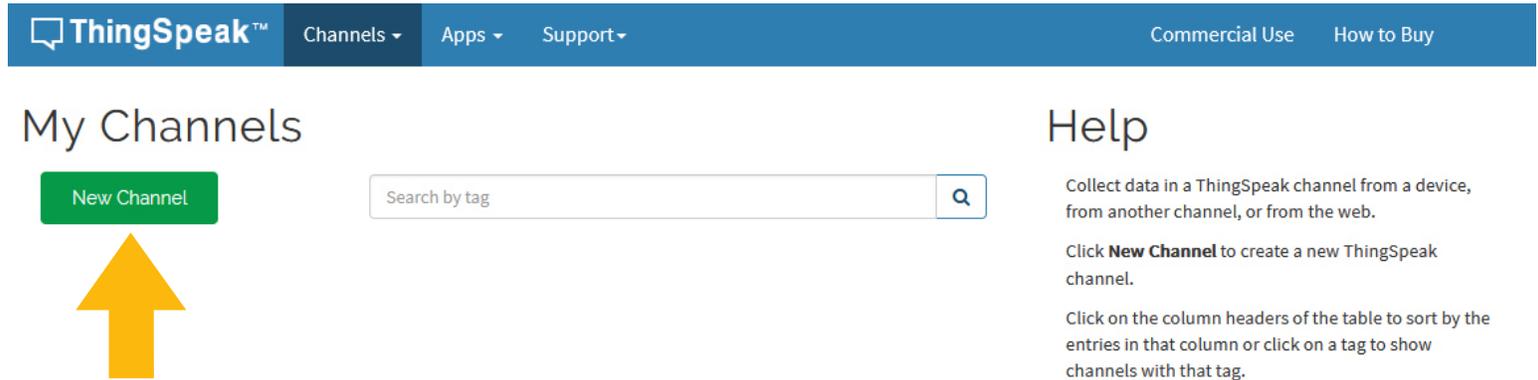
Step 1 Setup a ThingSpeak Account

Because the sensor data is stored / viewed on ThingSpeak, you will need to setup a ThingSpeak account. Create your ThingSpeak account by clicking [HERE](#).

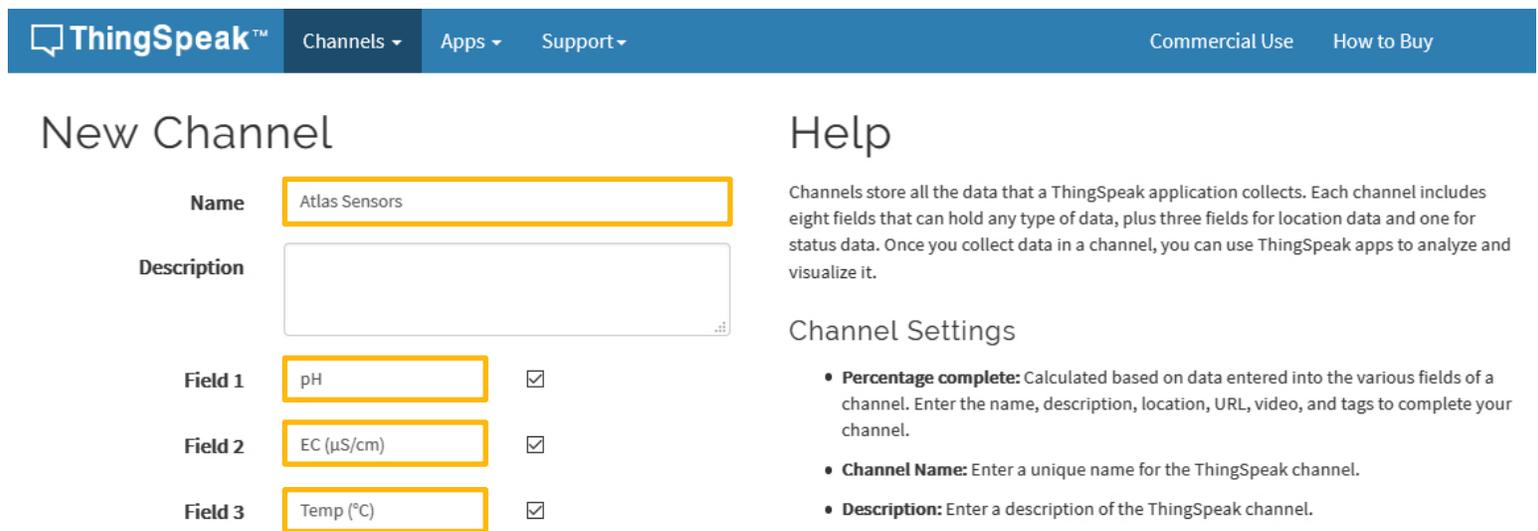
The image shows a screenshot of the ThingSpeak website's account creation page. The page has a blue header with the ThingSpeak logo and navigation links: Channels, Apps, Support, Commercial Use, and How to Buy. Below the header, there is a text block explaining that users must sign in with an existing MathWorks account or create a new one. It also mentions that non-commercial users can use ThingSpeak for free, while commercial users are eligible for a time-limited free evaluation. A link to 'paid license options' is provided. Below this text is a form with the MathWorks logo, an 'Email' input field, and a 'Create one' button. A 'Next' button is also visible. To the right of the form is a diagram illustrating the data flow: 'SMART CONNECTED DEVICES' send data to a cloud labeled 'DATA AGGREGATION AND ANALYTICS ThingSpeak™'. From the cloud, data is sent to a 'MATLAB' computer monitor labeled 'ALGORITHM DEVELOPMENT SENSOR ANALYTICS'.

Step 2 Create a Channel

Your data is uploaded to ThingSpeak through a 'Channel.' Select **New Channel**



The screenshot shows the 'My Channels' page in the ThingSpeak interface. At the top, there is a navigation bar with 'ThingSpeak™', 'Channels', 'Apps', and 'Support' menus, along with links for 'Commercial Use' and 'How to Buy'. Below the navigation bar, the page is titled 'My Channels'. On the left, there is a green 'New Channel' button, which is pointed to by a large yellow arrow. To the right of the button is a search bar labeled 'Search by tag'. On the right side of the page, there is a 'Help' section with text explaining how to collect data and how to use the 'New Channel' button.



The screenshot shows the 'New Channel' page in the ThingSpeak interface. At the top, there is a navigation bar with 'ThingSpeak™', 'Channels', 'Apps', and 'Support' menus, along with links for 'Commercial Use' and 'How to Buy'. Below the navigation bar, the page is titled 'New Channel'. The form has several fields: 'Name' (containing 'Atlas Sensors'), 'Description' (empty), 'Field 1' (containing 'pH'), 'Field 2' (containing 'EC (μS/cm)'), and 'Field 3' (containing 'Temp (°C)'). The 'Name', 'Field 1', 'Field 2', and 'Field 3' input boxes are highlighted with yellow borders. To the right of each field is a checkbox, which is checked for 'Field 2' and 'Field 3'. On the right side of the page, there is a 'Help' section explaining channels and a 'Channel Settings' section with a list of settings.

Fill out the highlighted boxes. (Be sure to click on the checkboxes to enable **field 2** and **3**)
For reference, this is what we entered.

Name **Atlas Sensors**
Field 1 **pH**
Field 2 **EC (μS/cm)**
Field 3 **Temp (°C)**

Scroll to the bottom of the page and click **Save Channel**.

Step 3 Get ThingSpeak API keys

After you saved your channel settings, you will be redirected to your channel page. Click on **API keys**.

The screenshot shows the 'My Channels' page in the ThingSpeak interface. At the top, there is a navigation bar with 'ThingSpeak™', 'Channels', 'Apps', and 'Support' menus, along with 'Commercial Use' and 'How to Buy' links. Below the navigation bar, the 'My Channels' section features a 'New Channel' button and a search bar. A table lists channels, with the first entry being 'Atlas Sensors' (created 2020-02-14, updated 2020-05-11 23:04). Below the table, there are tabs for 'Private', 'Public', 'Settings', 'Sharing', 'API Keys', and 'Data Import / Export'. A large yellow arrow points to the 'API Keys' tab. To the right, a 'Help' section provides instructions on how to collect data and create channels.

The screenshot shows the 'Atlas Sensors' channel page in the ThingSpeak interface. The navigation bar is the same as in the previous screenshot. The page title is 'Atlas Sensors'. Below the title, the 'Channel ID: xxxxxx' is highlighted with a yellow box. The 'Author:' and 'Access: Private' information is also visible. A navigation bar below the title includes 'Private View', 'Public View', 'Channel Settings', 'Sharing', 'API Keys', and 'Data Import / Export'. The 'API Keys' tab is selected. Below this, the 'Write API Key' section is highlighted with a yellow box. It shows a 'Key' field containing 'XXXXXXXXXXXXXXXXXXXX' and a 'Generate New Write API Key' button. To the right, a 'Help' section explains that API keys enable writing data to a channel or reading data from a private channel. Below that, 'API Keys Settings' are listed, including 'Write API Key' and 'Read API Keys'.

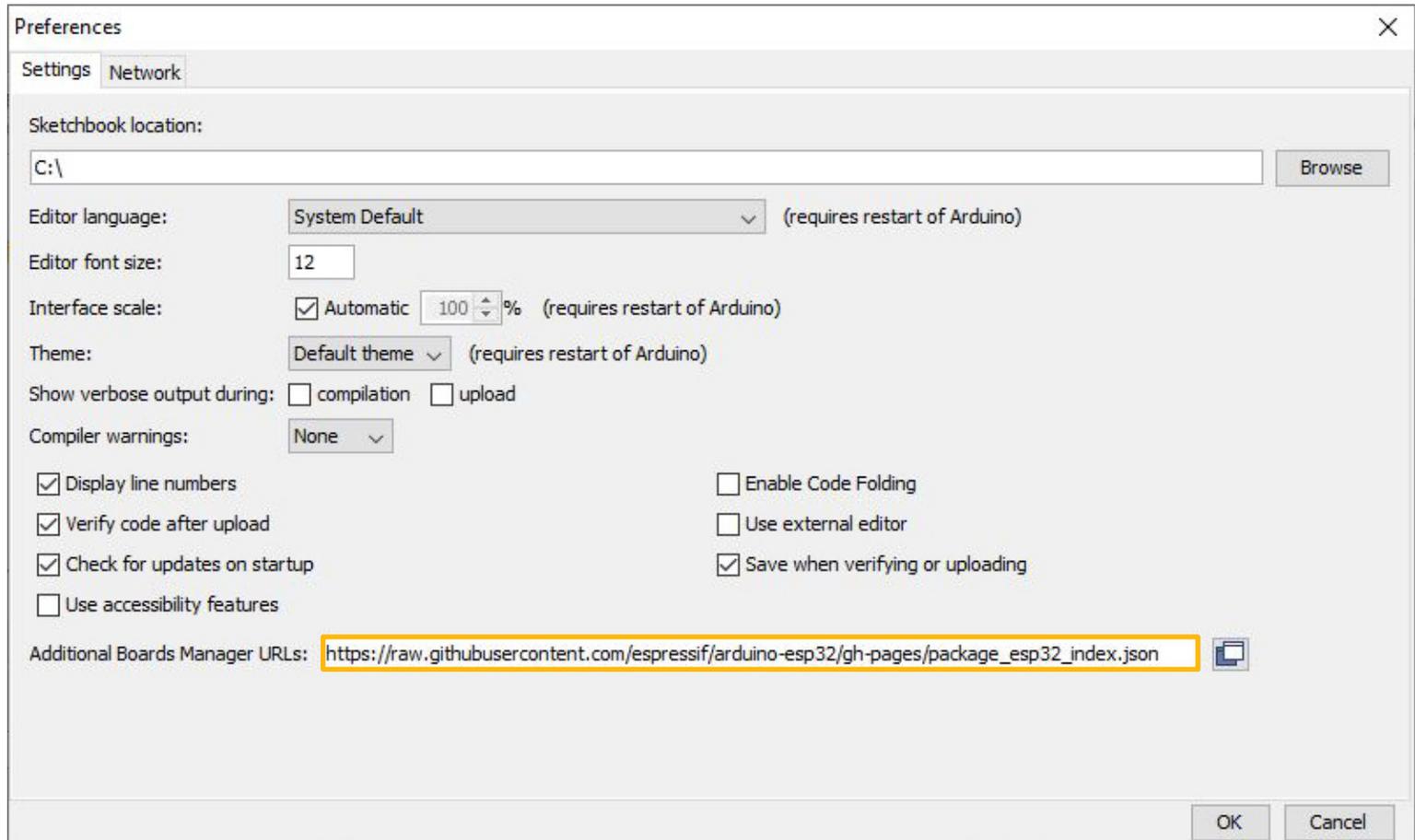
Be sure to save your **Channel ID** and **Write API Key** we are going to need these, in the next few steps.

Step 4 Make sure your Arduino IDE libraries are up to date

A Make sure you have the correct path for the Esp32 Library

In the IDE, go to **File > Preferences**

Locate the **Additional Boards Manager URLs** text box.



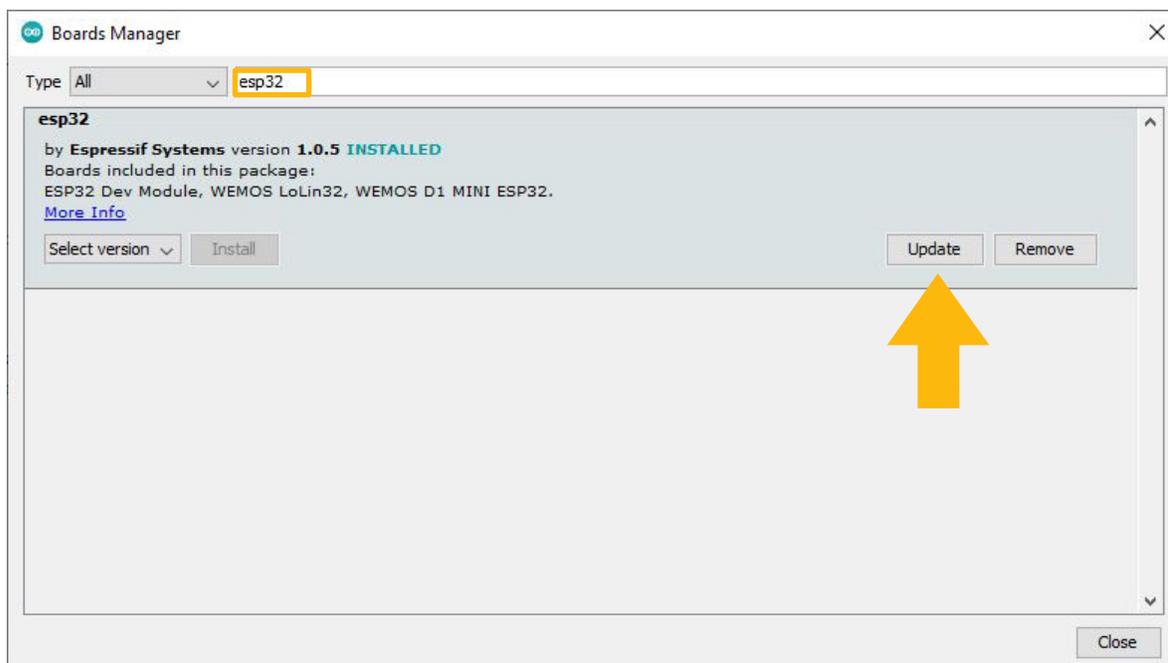
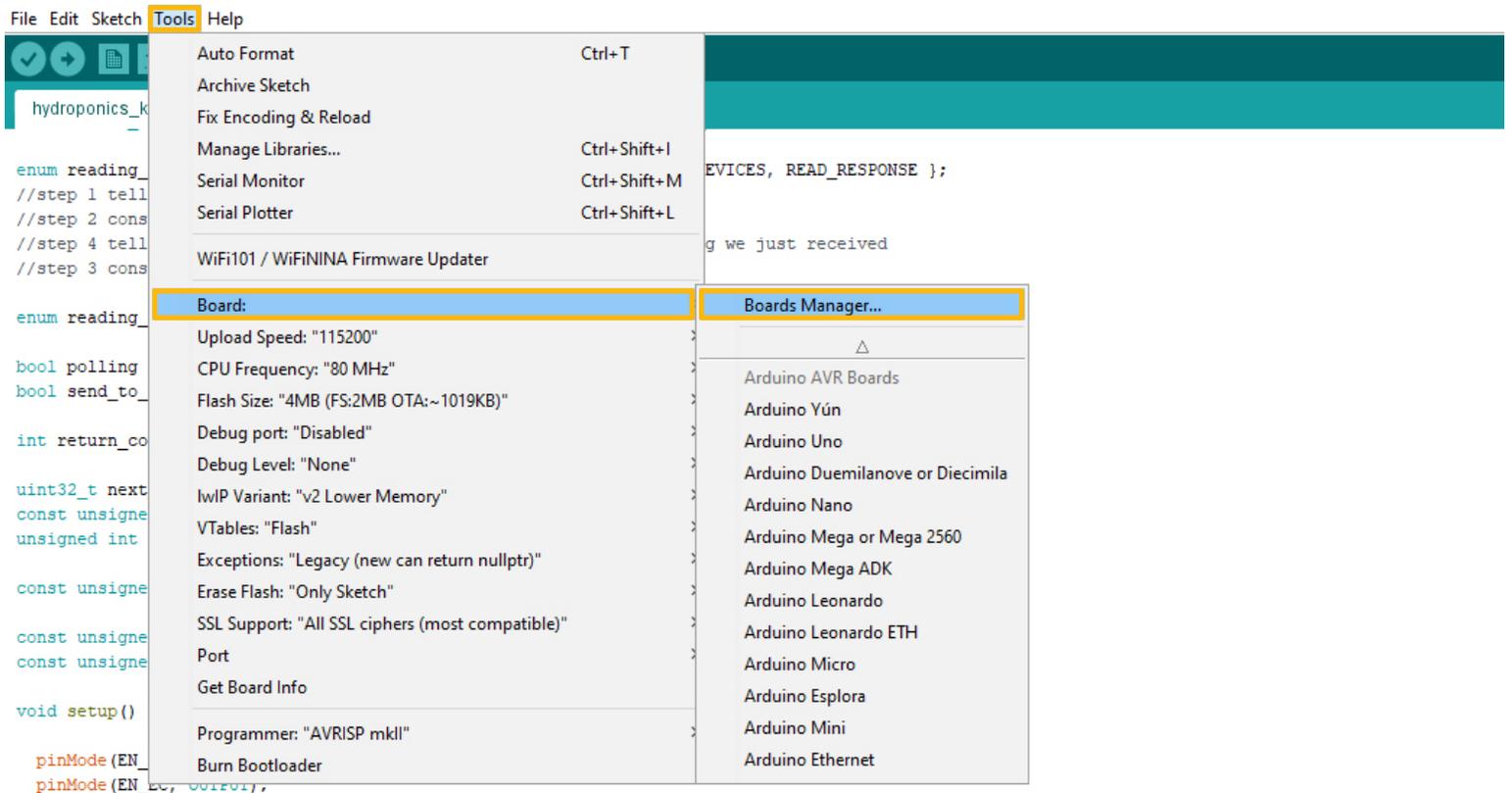
Make sure this URL is in the textbox

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

Click **OK**.

B Update the esp32 board

In the IDE, go to **Tools > Board > Boards Manager**



In the search bar of the Boards Manager, lookup **esp32**.
Update to the most recent version if you don't already have it.

(Version 1.0.5 in not the most recent version)

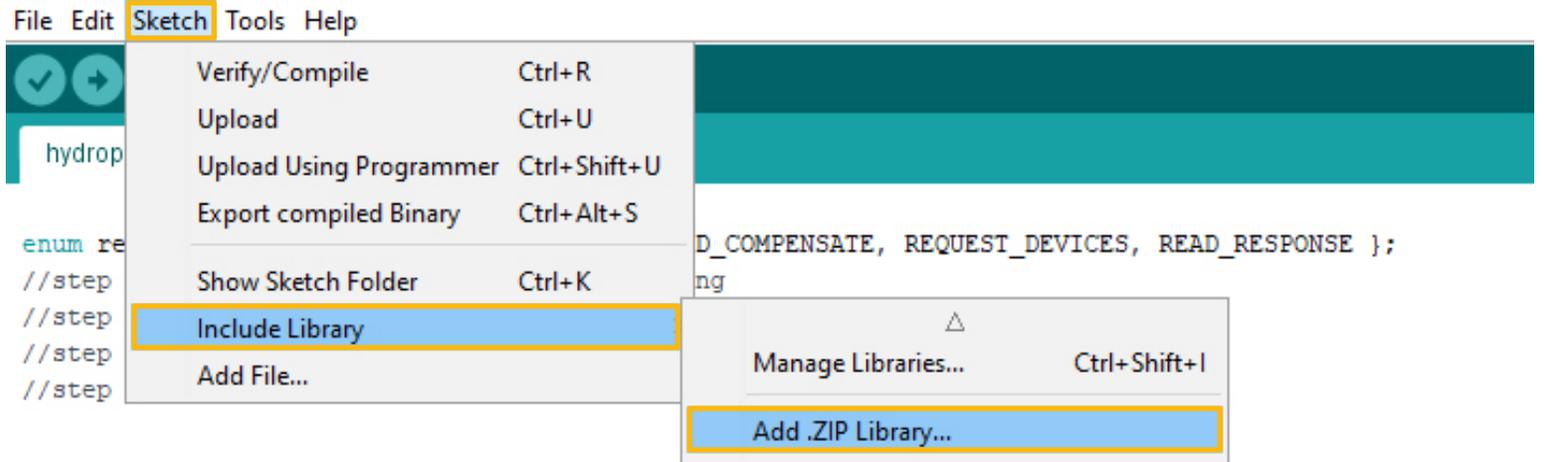
C Download the ThingSpeak library for Arduino

Click [HERE](#) to download the latest version of the ThingSpeak library.

Don't unzip it!

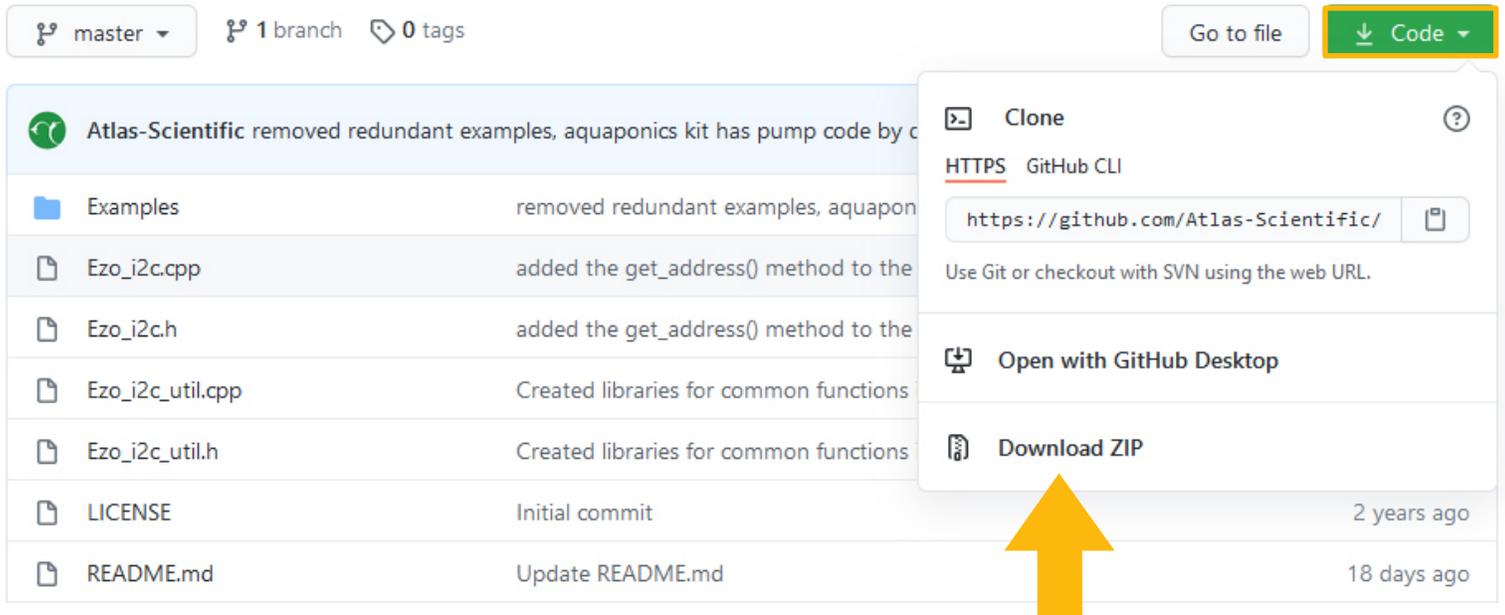
Import the .ZIP file into your Arduino IDE.

To import the .ZIP file go to **Sketch > Include Library > Add .ZIP Library**



D Add the EZO I2C Library

To download the Ezo_I2c library file, click [HERE](#).



Don't unzip it!

Import the .ZIP file to your Arduino IDE.

To import the .ZIP file go to **Sketch > Include Library > Add .ZIP Library**

Step 5 Flash the Hydroponics meter with the correct code

A Select, open and adjust the code you want to use for your Wi-Fi Kit

File> Examples> EZO_I2C_lib-master> Examples> IOT_kits> hydroponics_kit

The screenshot shows the Arduino IDE interface. The File menu is open, and the path File > Examples > EZO_I2C_lib-master > Examples > IOT_kits > hydroponics_kit is highlighted. The code editor displays the following code:

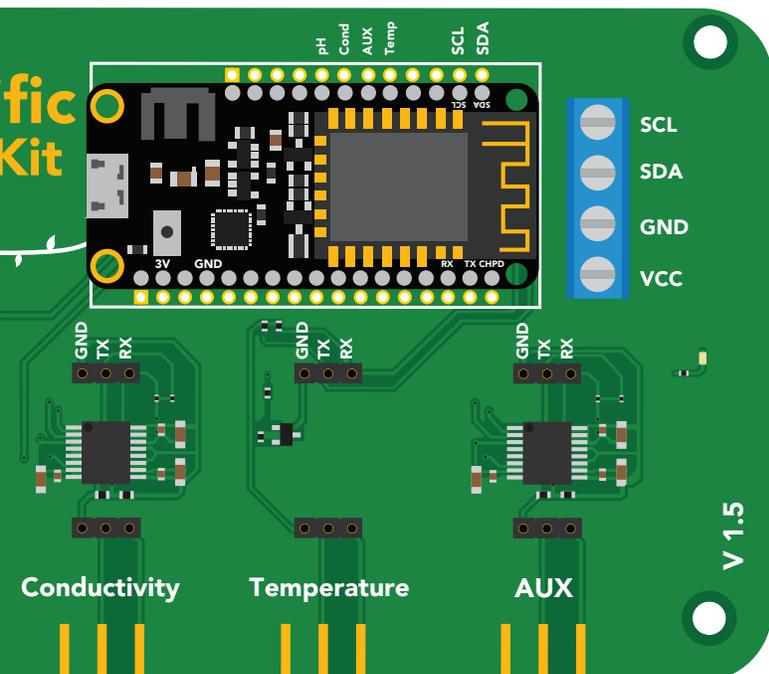
```
11 // This sketch demonstrates how to use the I2C library to control a hydroponics kit that uses the Adafruit huzzah32 as its computer.
12 WiFiClient client; // Instance of the WiFiClient library
13
14 //-----
15 const String ssid = "XXXXXXXX"; //The name of the Wi-Fi network you are connecting to
16 const String password = "XXXXXXXX"; //Your WiFi network password
17 const long myChannel = 1234; //Your Thingspeak channel number
18 const char * myWriteKey = "XXXXXXXX"; //Your ThingSpeak Write API Key
19 //-----
20
21 Ezo_board PH = Ezo_board(99, "PH"); //create a PH circuit object, who's address is 99 and name is "PH"
22 Ezo_board EC = Ezo_board(100, "EC"); //create an EC circuit object who's address is 100 and name is "EC"
23 Ezo_board RTD = Ezo_board(102, "RTD"); //create an RTD circuit object who's address is 102 and name is "RTD"
24 Ezo_board PMP = Ezo_board(103, "PMP"); //create an PMP circuit object who's address is 103 and name is "PMP"
25
26 Ezo_board deviceList[] = {
27   PH,
28   EC,
29   RTD,
30   PMP
31 };
32
33 Ezo_board* defaultBoard = PH; //used to default to a board if no board is specified
34
35 //gets the length of the array
36 const uint8_t deviceListLength = sizeof(deviceList);
```

B Fill in your Wi-Fi / ThingSpeak credentials

Fill in your Wi-Fi name and Password, along with the Channel ID and Write API Key to the code. (see step 3)

```
3 #include <iot_cmd.h>
4 #include <WiFi.h> //include wifi library
5 #include "ThingSpeak.h" //include thingspeak library
6 #include <sequencer4.h> //imports a 4 function sequencer
7 #include <sequencer1.h> //imports a 1 function sequencer
8 #include <Ezo_i2c_util.h> //brings in common print statements
9 #include <Ezo_i2c.h> //include the EZO I2C library from https://github.com/Atlas-Scientific/Ezo\_I2c\_lib
10 #include <Wire.h> //include arduinos i2c library
11
12 WiFiClient client; //declare that this device connects to a Wi-Fi network,c
13
14 //-----Fill in your Wi-Fi / ThingSpeak Credentials-----
15 const String ssid = "Wifi Name"; //The name of the Wi-Fi network you are connecting to
16 const String pass = "Wifi Password"; //Your WiFi network password
17 const long myChannelNumber = 1234566; //Your ThingSpeak channel number
18 const char * myWriteAPIKey = "XXXXXXXXXXXXXXXXXX"; //Your ThingSpeak Write API Key
19 //-----
```

C Choose enable pins



Check here to see which version you have.

```
38 //-----For version 1.4 use these enable pins for each circuit-----
39 //const int EN_PH = 13;
40 //const int EN_EC = 12;
41 //const int EN_RTD = 33;
42 //const int EN_AUX = 27;
43 //-----
44
45 //-----For version 1.5 use these enable pins for each circuit-----
46 const int EN_PH = 12;
47 const int EN_EC = 27;
48 const int EN_RTD = 15;
49 const int EN_AUX = 33;
50 //-----
```

**If version 1.4
use these enable pins.**

**If version 1.5
use these enable pins.**

D Setting up your pump

If you do not have a pump attached, you can just skip this part. The code is rather self explanatory. You set what parameters will trigger the pump to engage.

```
57 //parameters for setting the pump output
58 #define PUMP_BOARD      PMP      //the pump that will do the output (if theres more than one)
59 #define PUMP_DOSE      -0.5      //the dose that the pump will dispense in milliliters
60 #define EZO_BOARD      EC        //the circuit that will be the target of comparison
61 #define IS_GREATER_THAN true     //true means the circuit's reading has to be greater than the comparison value,
62 #define COMPARISON_VALUE 1000    //the threshold above or below which the pump is activated
```

Step 6

Setting up the HUZDAH board

A Set the target CPU to flash

Tools > Board > ESP32 Arduino > Adafruit ESP32 Feather

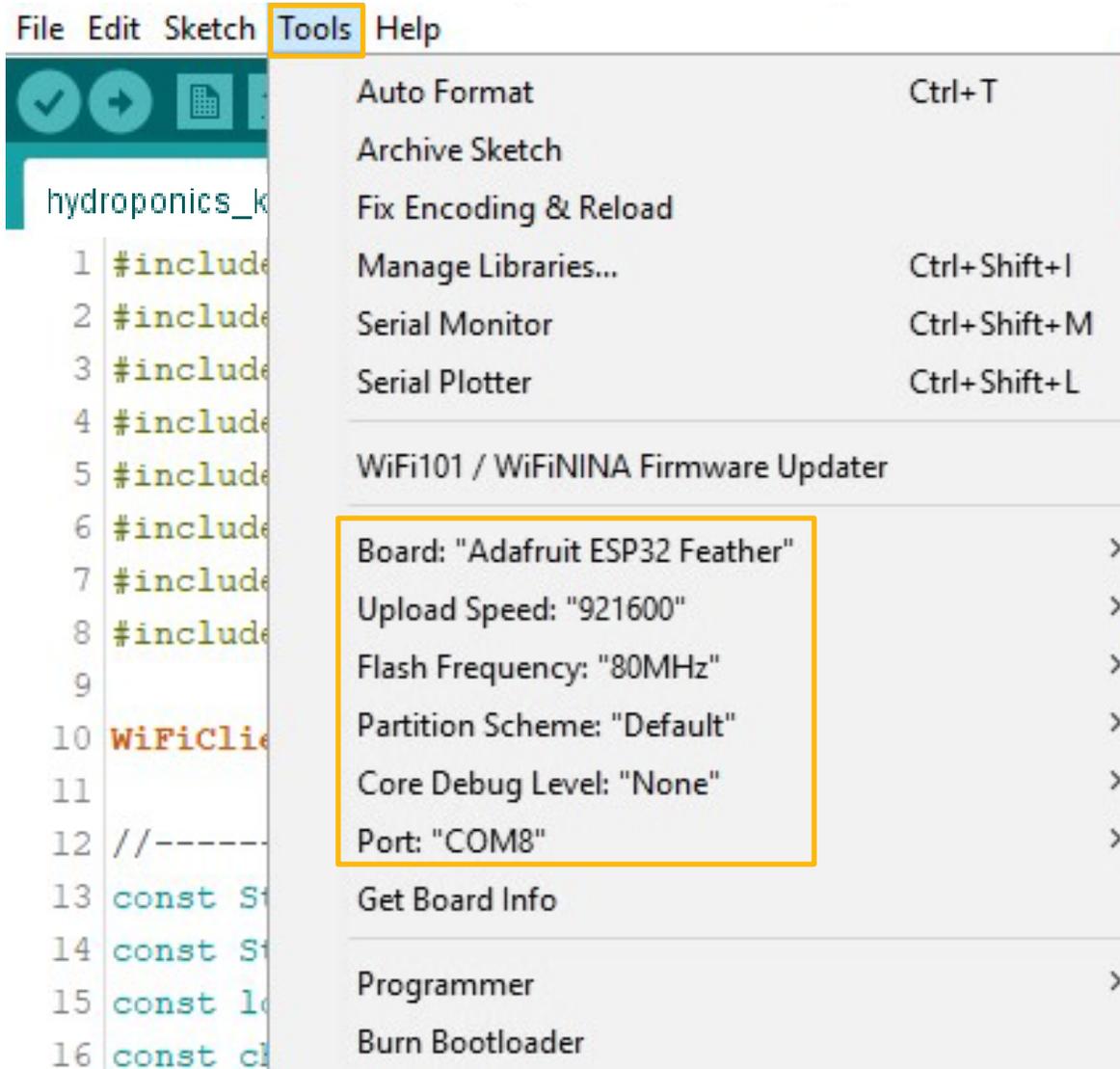
The screenshot shows the Arduino IDE interface with the 'Tools' menu open. The 'Board' submenu is also open, showing 'ESP32 Arduino' selected. The 'ESP32 Arduino' submenu is open, showing 'Adafruit ESP32 Feather' selected. The background shows a code editor with a sketch for a hydroponics kit.

```
File Edit Sketch Tools Help
hydroponics
1 //This
2
3 #includ
4 #includ
5 #includ
6 #includ
7 #includ
8 #includ
9 #includ
10 #includ
11
12 WiFiCl
13
14 //----
15 const String ssid = "Wifi Name";
16 const String pass = "Wifi Password";
17 const long myChannelNumber = 1234566;
18 const char * myWriteAPIKey = "XXXXXXXXXXXXXXXXXXXX";
19 //-----
20
21 Ezo_board PH = Ezo_board(99, "PH"); //create a PH circ
22 Ezo_board EC = Ezo_board(100, "EC"); //create an EC cir
23 Ezo_board RTD = Ezo_board(102, "RTD"); //create an RTD ci
24 Ezo_board PMP = Ezo_board(103, "PMP"); //create an PMP ci
//include wifi libra
//include thingspea
//imports a 4 functi
```

B Adjust CPU Settings

Make sure the CPU settings on the Adafruit HUZZAH32 are correct. To adjust the CPU settings, click **Tools**.

For reference, this is what Atlas Scientific set the CPU settings to. (your options may not be exactly the same, just try and match them as closely as possible. Don't forget to set the correct com port for your device.)



C Compile and upload

hydroponics_kit | Arduino 1.8.13

File Edit Sketch Tools Help

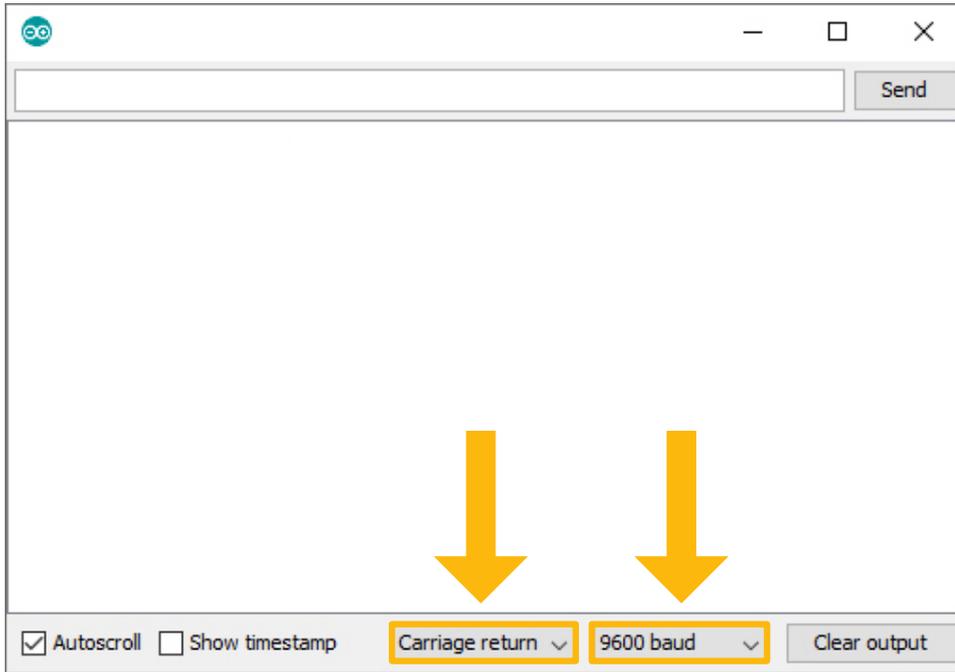


Compile and upload the code.

Step 7 See the readings

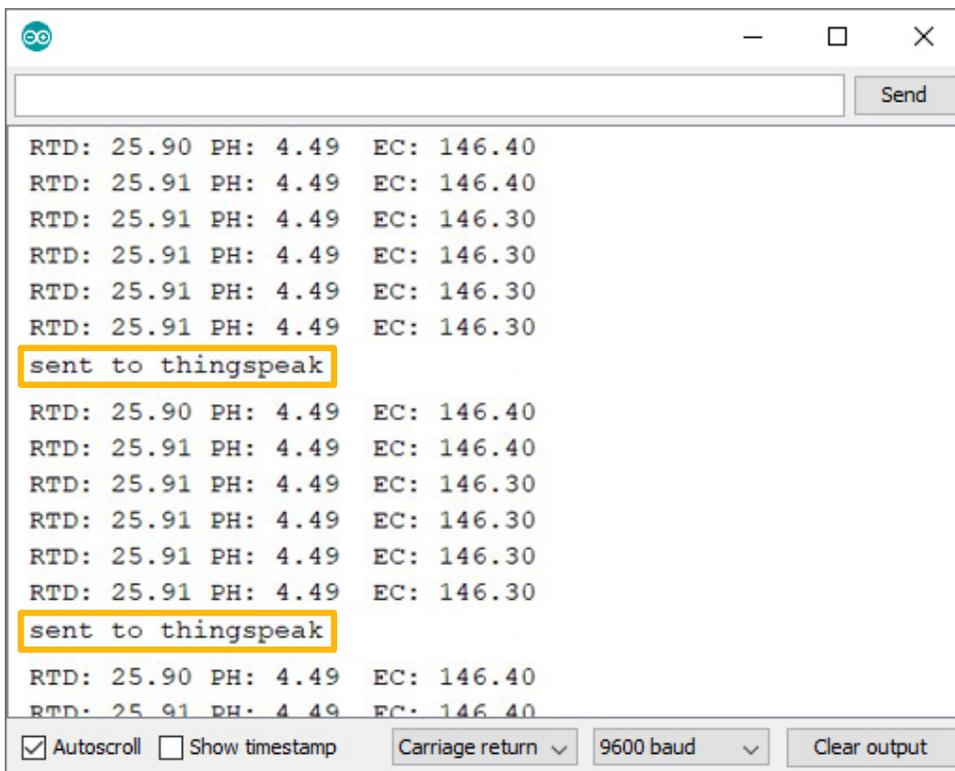
Open your Arduino serial monitor.

(You must have the serial monitor set to the com port from the Adafruit HUZZAH32.)

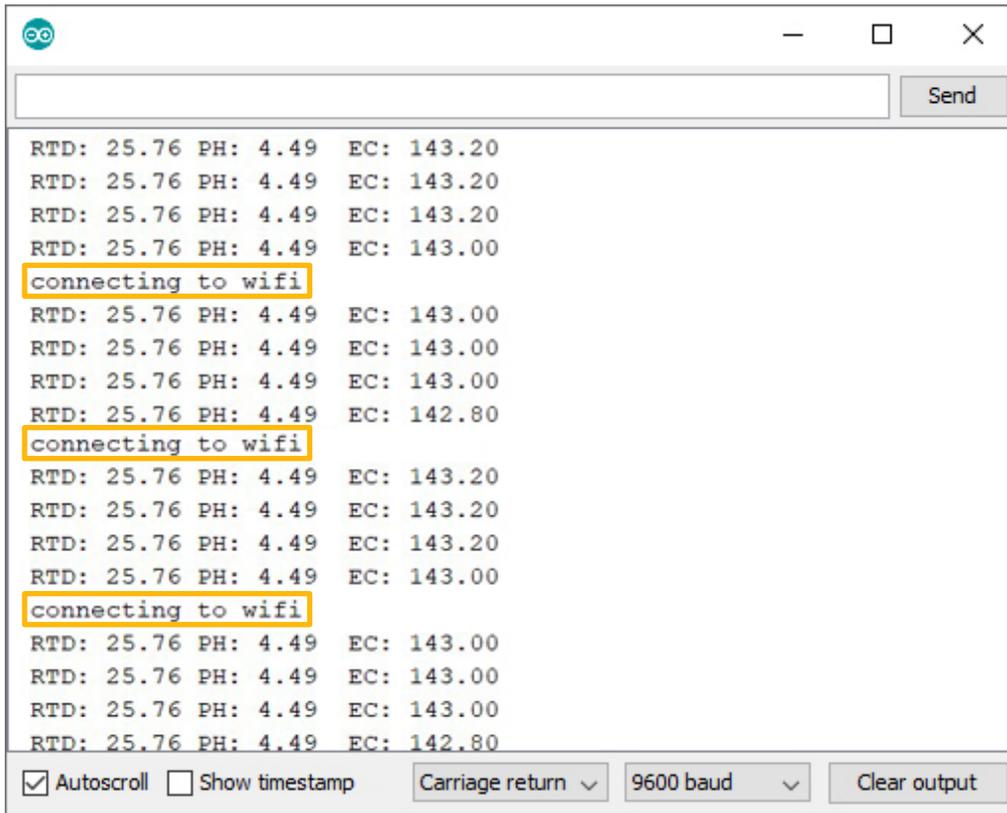


Set to **carriage return** and **9600 baud**.

The Wi-Fi Hydroponics Meter will always attempt to connect to ThingSpeak on bootup.



If it cannot connect to your Wi-Fi you will see this:

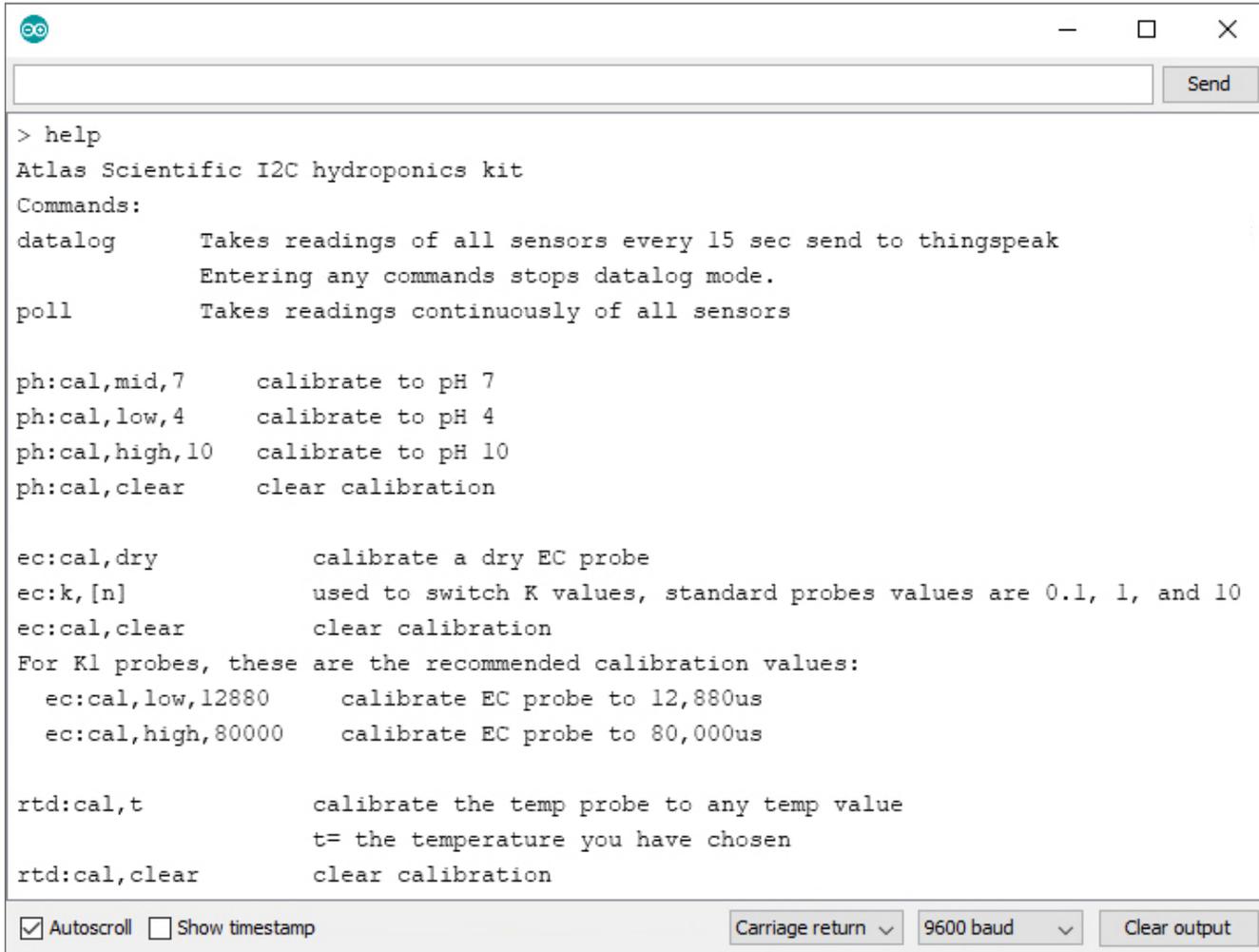


The screenshot shows a terminal window with a title bar containing a logo, a minimize button, a maximize button, and a close button. Below the title bar is a text input field and a 'Send' button. The main area of the terminal displays a series of sensor readings: 'RTD: 25.76 PH: 4.49 EC: 143.20'. These readings are repeated several times. Interspersed among these readings are three lines of the text 'connecting to wifi', each highlighted with a yellow box. At the bottom of the terminal window, there is a control bar with several options: a checked checkbox for 'Autoscroll', an unchecked checkbox for 'Show timestamp', a dropdown menu for 'Carriage return', a dropdown menu for '9600 baud', and a 'Clear output' button.

Entering the **poll** command will stop the Wi-Fi Hydroponics Meter from uploading the readings to thingspeak, while you debug your Wifi problems.

Step 8 Sensor Calibration

Atlas Scientific created a list of calibration commands that are built into the library. Type in **help** to see a list of commands.



```
> help
Atlas Scientific I2C hydroponics kit
Commands:
datalog      Takes readings of all sensors every 15 sec send to thingspeak
              Entering any commands stops datalog mode.
poll         Takes readings continuously of all sensors

ph:cal,mid,7  calibrate to pH 7
ph:cal,low,4  calibrate to pH 4
ph:cal,high,10 calibrate to pH 10
ph:cal,clear  clear calibration

ec:cal,dry    calibrate a dry EC probe
ec:k,[n]      used to switch K values, standard probes values are 0.1, 1, and 10
ec:cal,clear  clear calibration
For K1 probes, these are the recommended calibration values:
  ec:cal,low,12880    calibrate EC probe to 12,880us
  ec:cal,high,80000  calibrate EC probe to 80,000us

rtd:cal,t     calibrate the temp probe to any temp value
              t= the temperature you have chosen
rtd:cal,clear clear calibration

 Autoscroll  Show timestamp
Carriage return 9600 baud Clear output
```

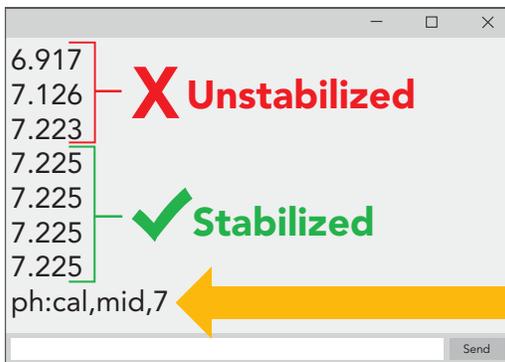
A The poll command

Send the command **poll**; This will let you see the readings once per second and it will stop uploading to ThingSpeak while you calibrate.

B Calibrate pH

When calibrating pH, you must always calibrate to pH 7 first.

Remove the soaker bottle and rinse off the pH probe. Remove the top of the pH 7.00 calibration solution pouch. Place the pH probe inside the pouch and let the probe sit in the calibration solution until the readings stabilize. This will take about 1 – 2 mins.



Once the readings have stabilized, issue the Mid point calibration command. **ph:cal,mid,7**

After 20 mins, the calibration solution inside an open pouch is no longer considered accurate.

Dispose of the unused solution, after calibration.

Rinse off the probe and repeat this process for both **pH 4.00** and **pH 10.00**.

C Calibrate Conductivity

Setting the Conductivity probe type

If your probe \neq K 1.0 (default), then set the probe type by using the **ec:k,n** command. (where n = K value of your probe)

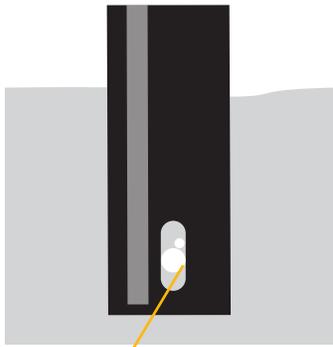
Example, if you have a K 0.1 conductivity probe issue the command **ec:k,0.1**

When calibrating Conductivity, you must always calibrate a dry probe first.

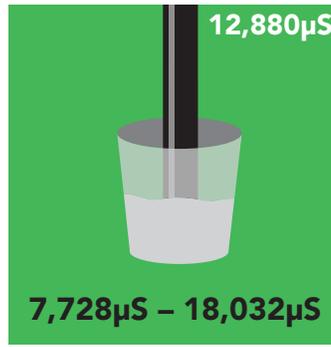


Make sure that the probe is dry before issuing this command, **ec:cal,dry**

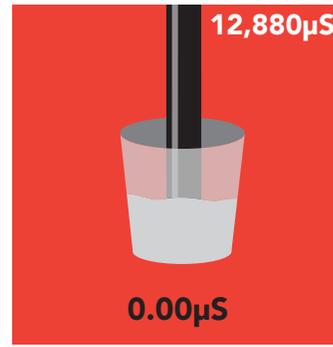
Once the dry calibration has been completed, place the probe into a small cup of the low point calibration solution. Shake the probe to make sure you do not have trapped air bubbles in the sensing area. You should see readings that are off by **1 – 40%** from the stated value of the calibration solution. Wait for readings to stabilize.



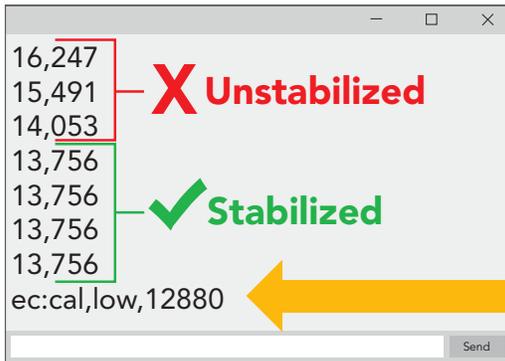
Trapped air in sensing area (shake to remove)



+/- 40%

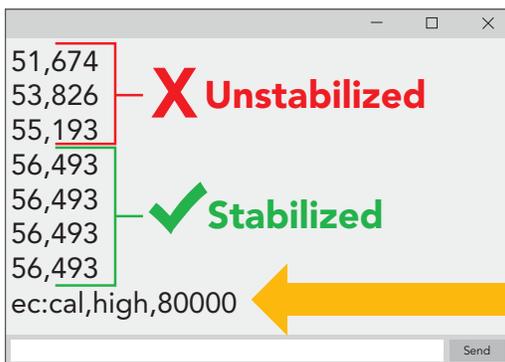


check probe connection, you cannot calibrate to 0.



Once the readings stabilize, issue the low point calibration command. **ec:cal,low,12880**
(Readings will **NOT** change)

Rinse off the probe before calibrating to the high point. Pour a small amount of the high point calibration solution into a cup. Shake the probe to remove trapped air. Again, the readings may be off by **1 – 40%** Wait for readings to stabilize.

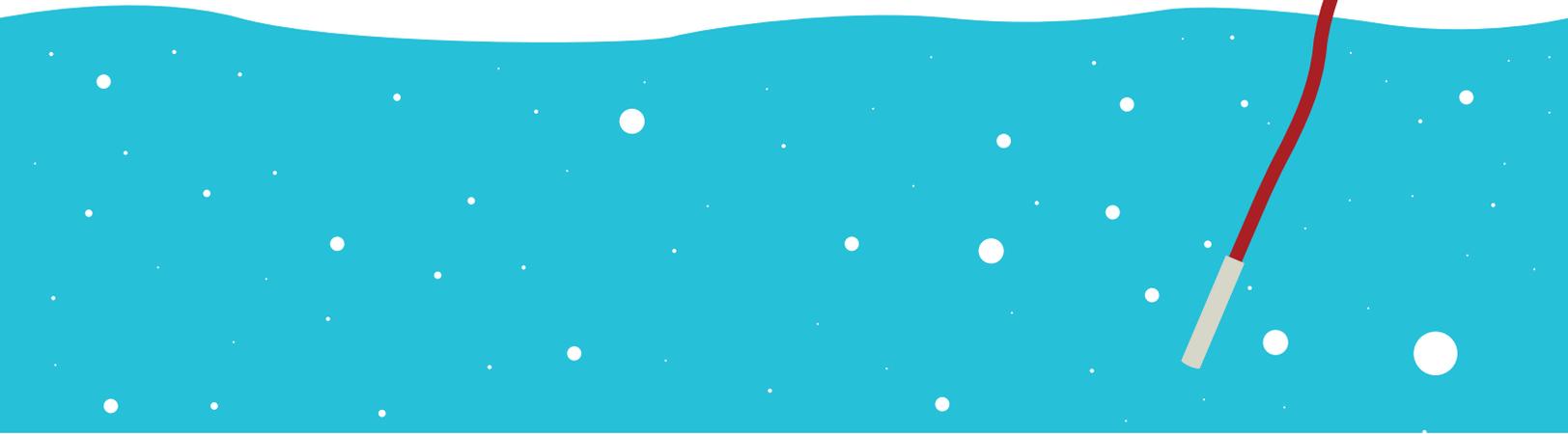


Once the readings stabilize, issue the high point calibration command. **ec:cal,high,80000**
(Readings **will** change, calibration complete).

D Calibrate Temperature

Calibrating the PT-1000 temperature probe is not required. However, if you want to, a simple method to calibrate the probe is to place the PT-1000 into boiling water. Then issue command **rtd:cal,t**

100 °C

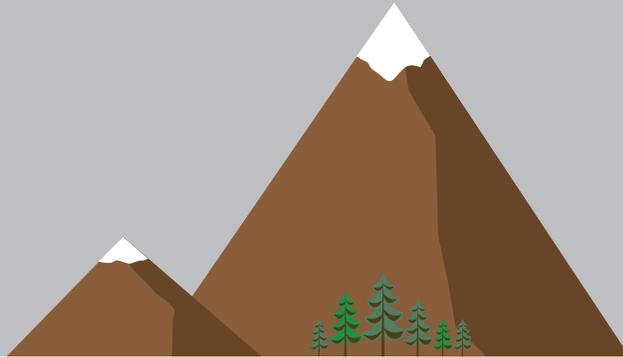


Elevation in meters

305
229
152
76
0
-76
-152

Boiling point

98.9 °C
99.2 °C
99.5 °C
99.7 °C
100 °C
100.3 °C
100.5 °C



Calibration Complete

Step 9 **Almost done!**

Once you are finished with calibration, issue the **datalog** command to resume taking a reading every 15 seconds and uploading it to thingspeak.

To see the data on your phone, download the ThingSpeak app.



Setup Complete!