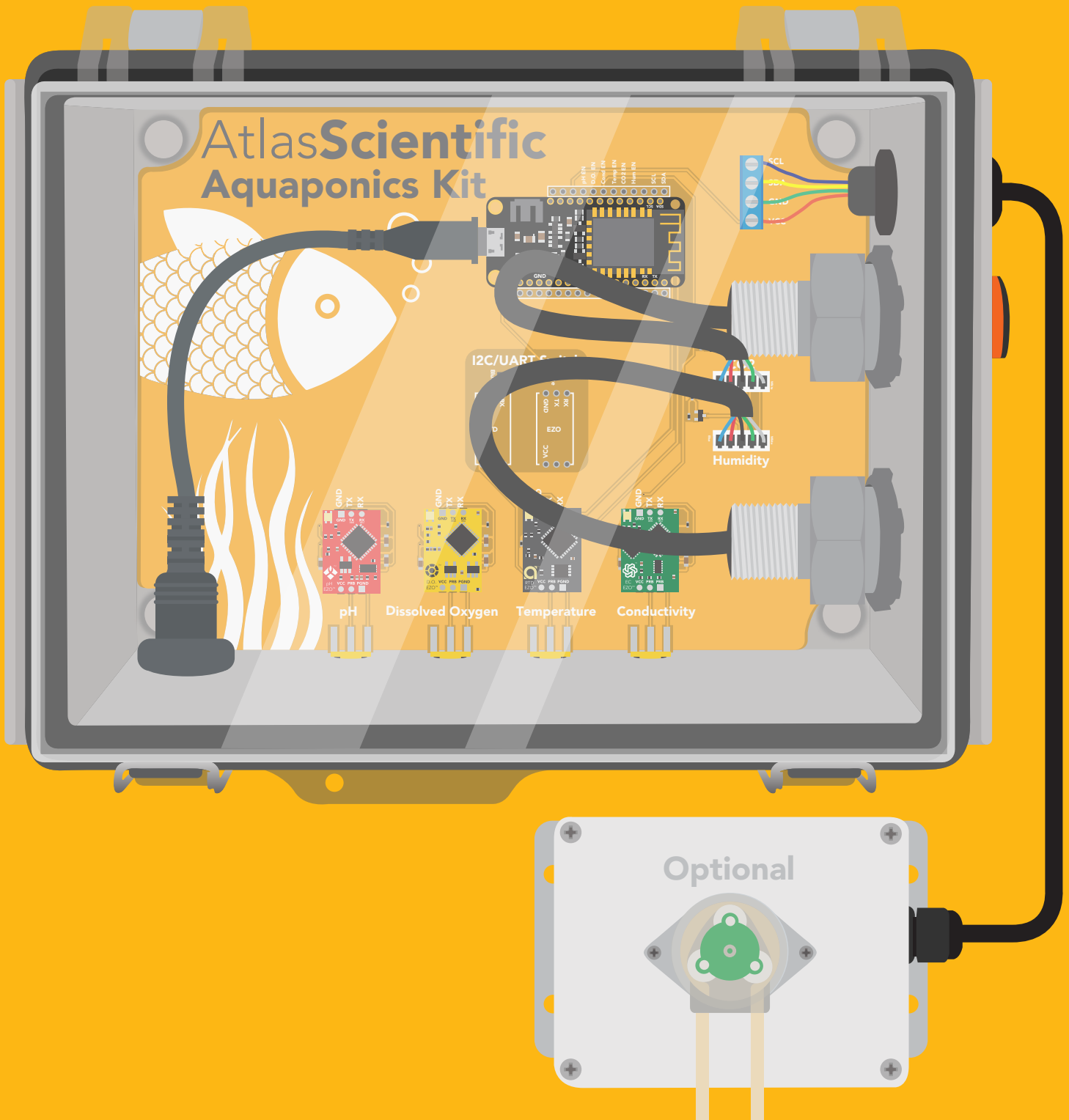


Wi-Fi Aquaponic Kit

Datasheet

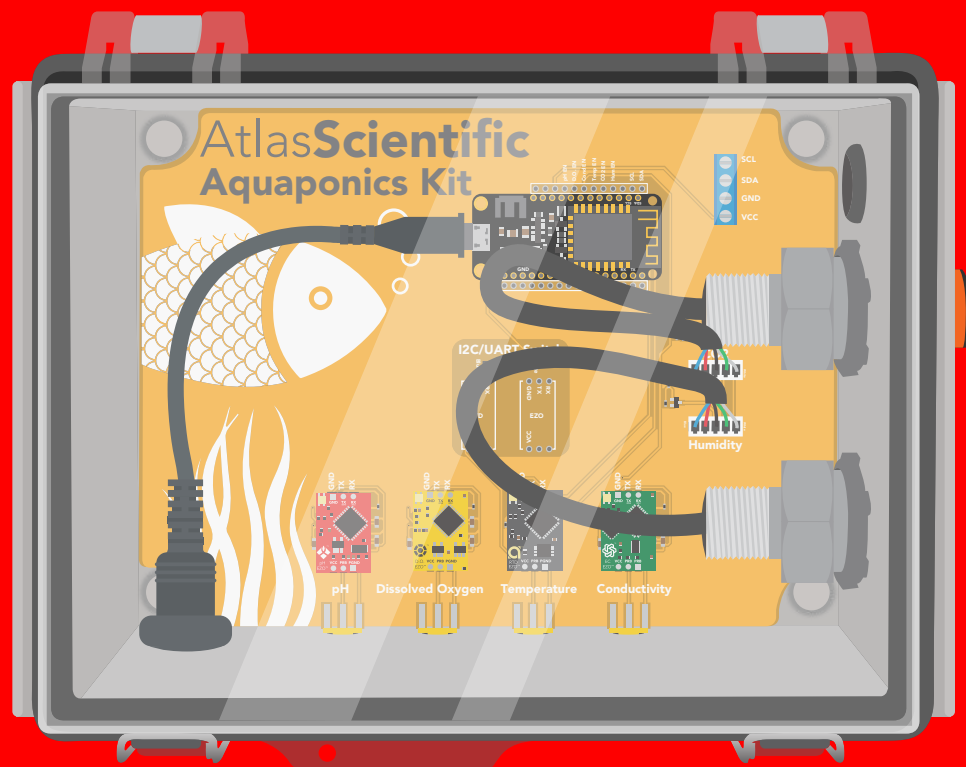


STOP

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)

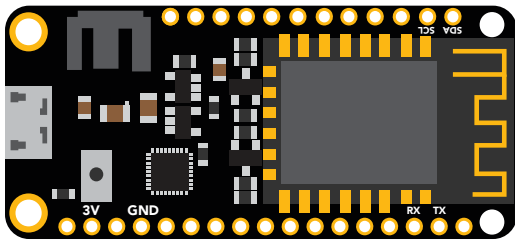
Operating principle

The Wi-Fi Aquaponics kit has been designed to provide the engineer with a simple way of remotely monitoring and controlling an aquaponics system's chemistry. Sensor data is uploaded to ThingSpeak™, a free, cloud-based data acquisition and visualization platform. The Wi-Fi Aquaponics 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

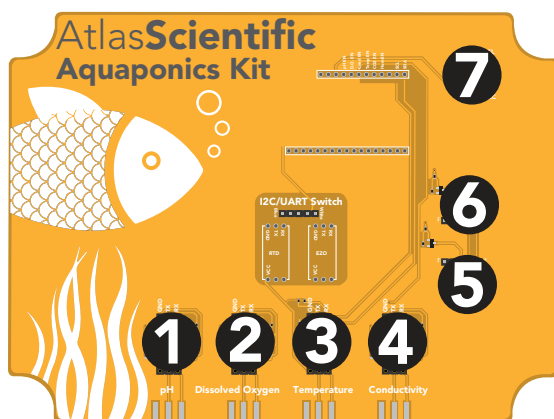
The Wi-Fi Aquaponics kit is controlled using an Adafruit HUZZAH32 as its CPU. The HUZZAH is programmed using the Arduino IDE and uses an onboard ESP32 as its Wi-Fi transmitter. [Adafruit HUZZAH32 datasheet.](#)



Sensor ports

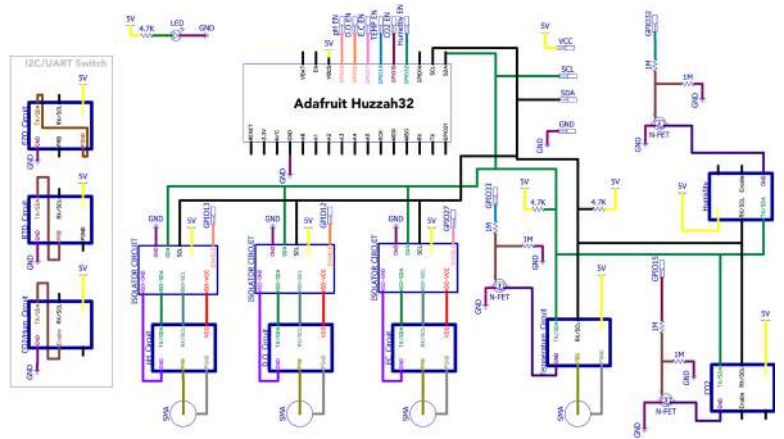
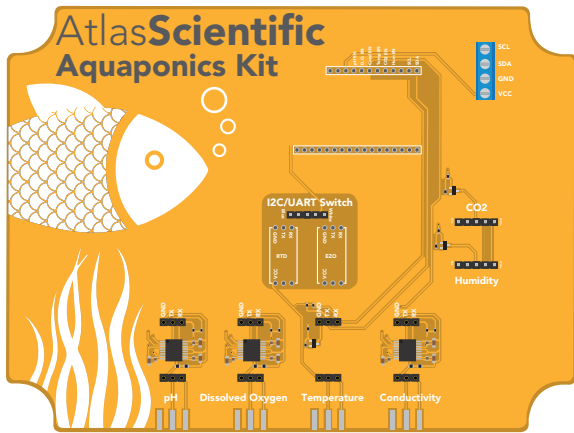
The Wi-Fi Aquaponics kit PCB has 7 sensor ports. Three of the ports are electrically isolated. The isolated ports are marked pH, Dissolved Oxygen, and Conductivity. The isolated ports are needed to take noise-free electrochemical readings. Because the sensing element of a temperature sensor is never in direct contact with the water, electrical isolation is not needed for temperature sensing.

Port 5 and 6 are marked Humidity and CO2. The terminal block marked Port 7 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.

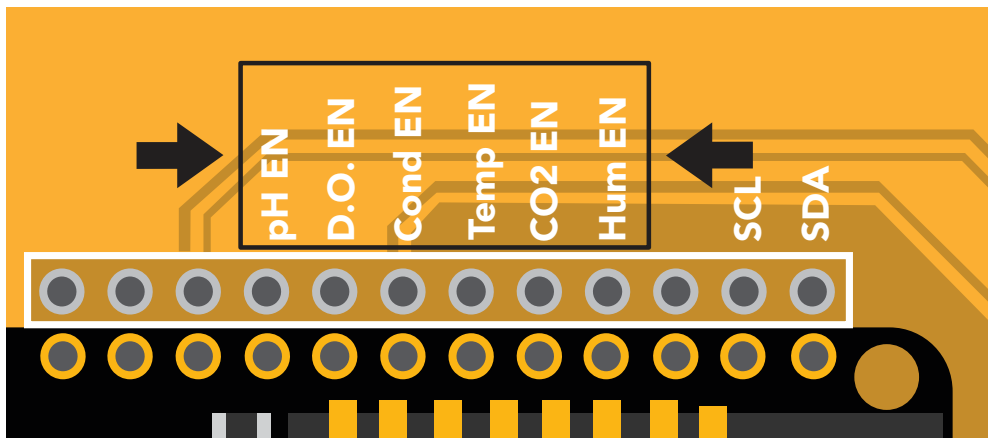


PCB

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 six 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, D.O and Cond) must be set low to power on the sensors. While the Temp, CO2 and Hum pins must be set high to power on the sensors.

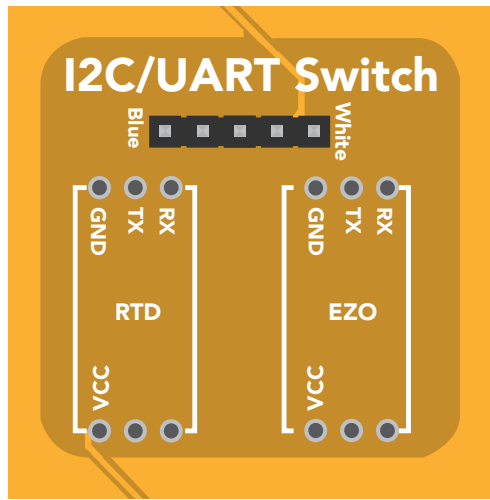
Truth table

Pin	Adafruit Huzzah32 GPIO	State	Sensor Power
pH EN	13	LOW	ON
D.O. EN	12	LOW	ON
Cond EN	27	LOW	ON
Temp EN	33	HIGH	ON
CO2 EN	15	HIGH	ON
Hum EN	32	HIGH	ON

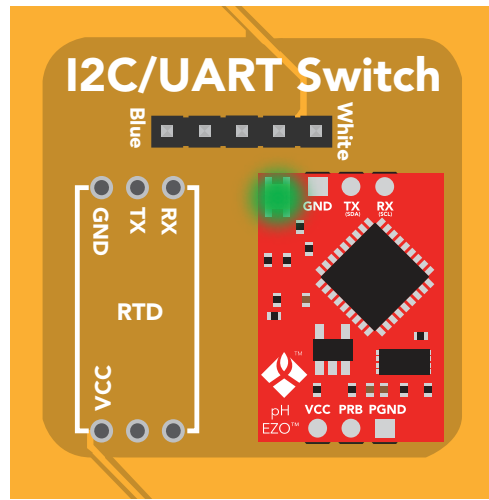
Sensor port 7 (the terminal block) does not have an enable pin and can not be turned off.

On Board I2C/UART Switch

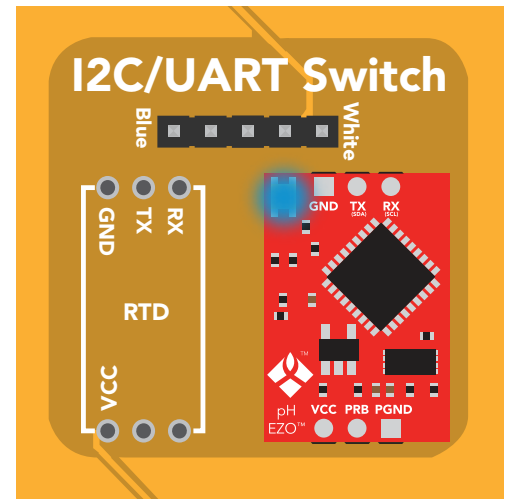
The PCB has a built in protocol toggler which can be used to switch each of the sensors between I2C/UART modes.



Place an EZO circuit onto the I2C/UART switch pins.



The EZO circuit in UART mode...



...is now in I2C mode.

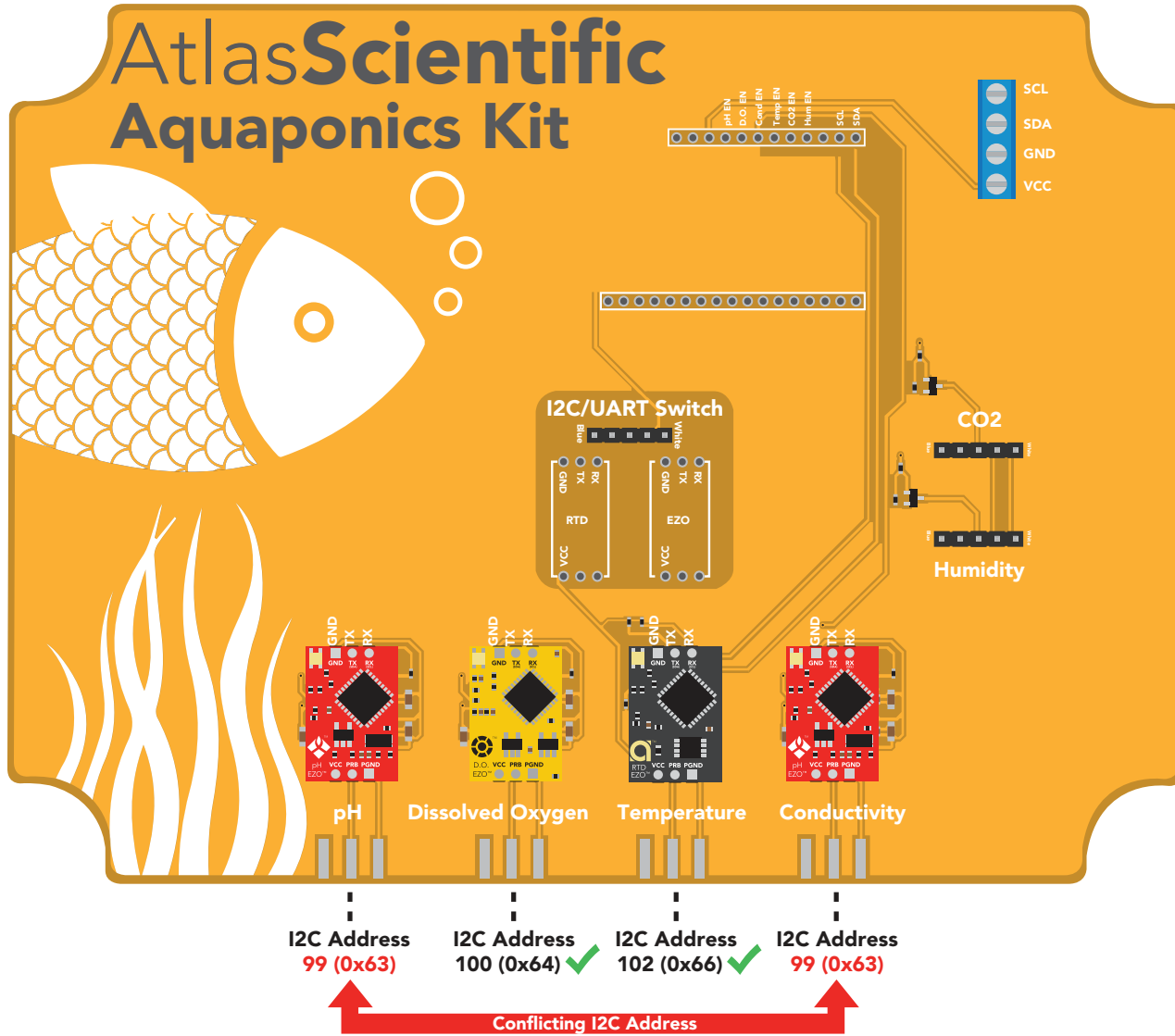
Data protocol

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, or use the on board protocol toggler mentioned above.

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.

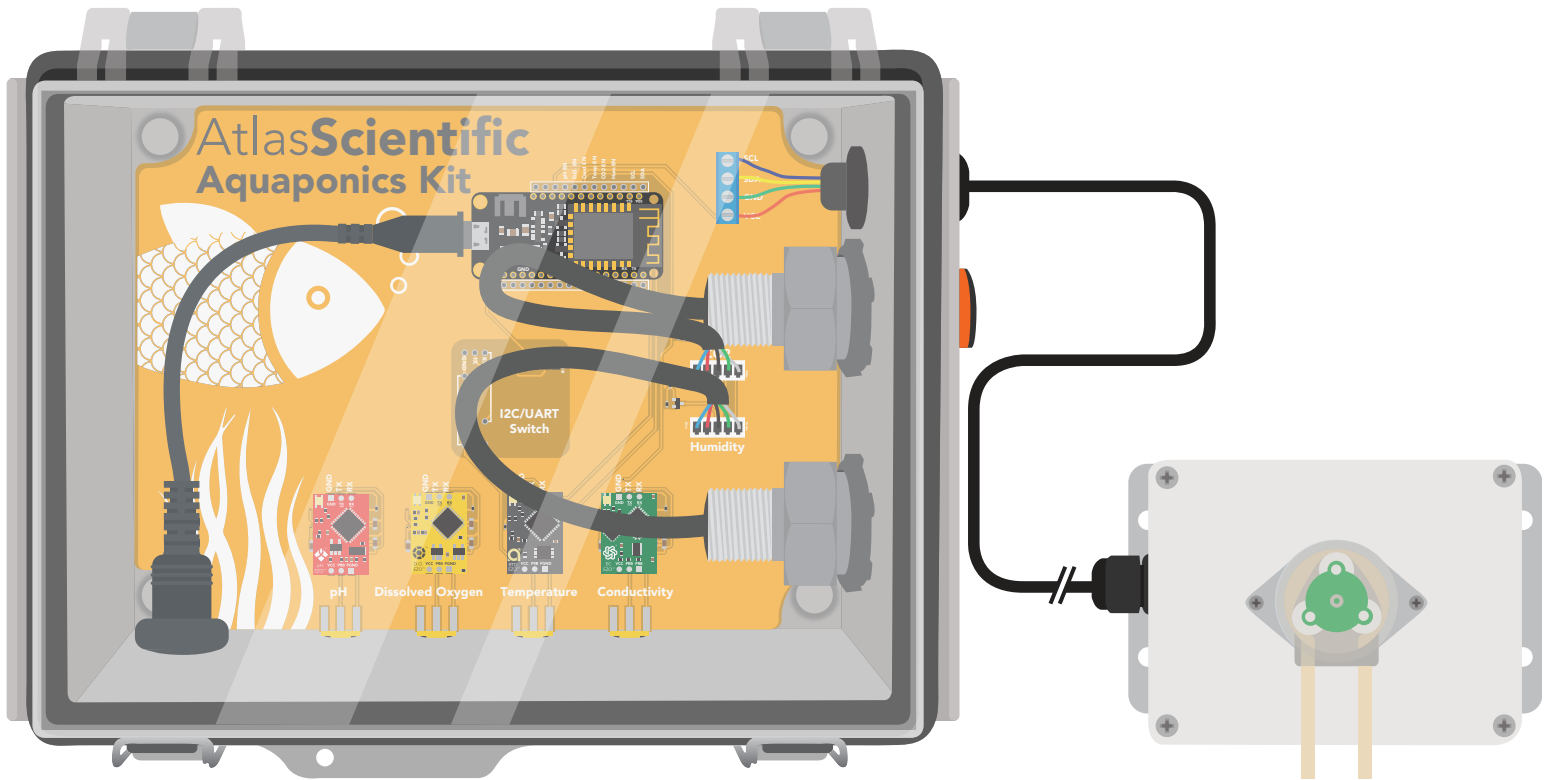


This table lists the default I2C address of components commonly added to this kit.

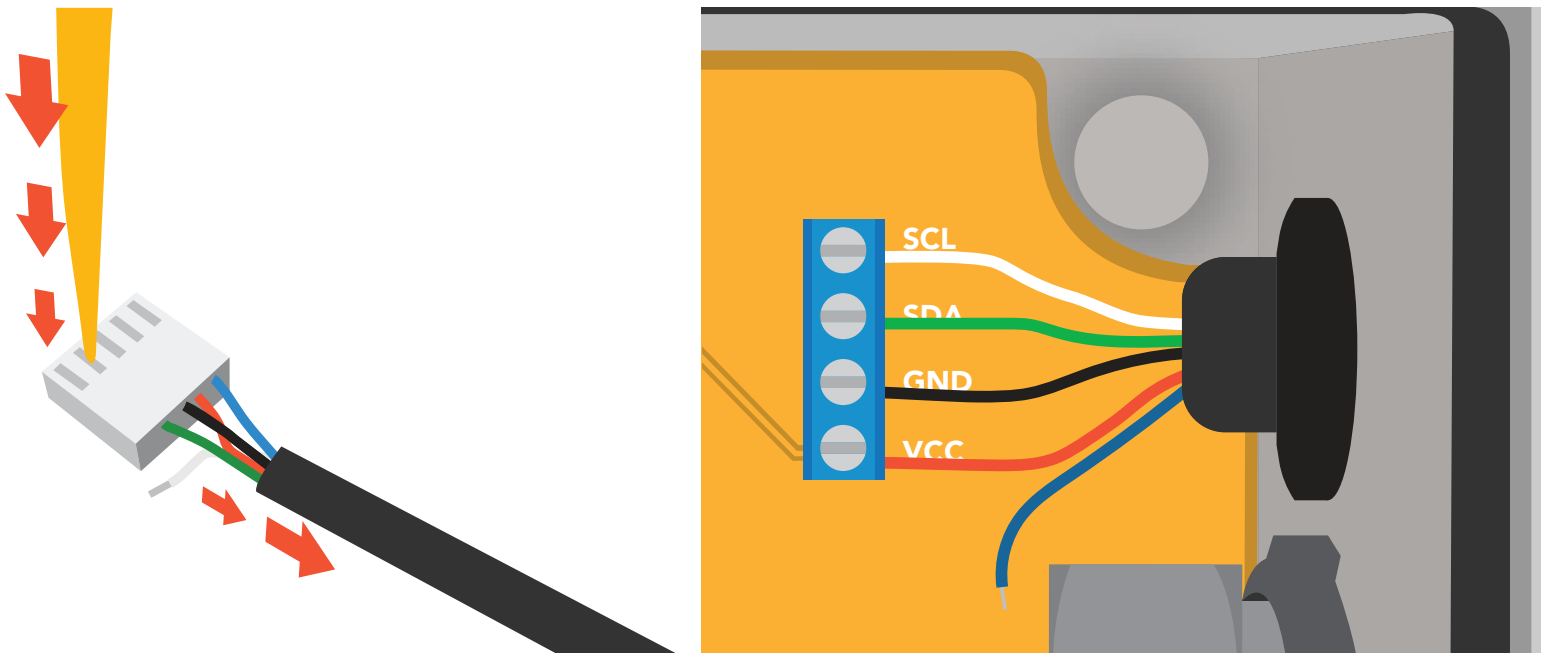
Device	I2C Address	Device	I2C Address	Device	I2C Address
EZO pH	99 (0x63)	EZO EC	100 (0x64)	EZO CO2	105 (0x69)
EZO ORP	98 (0x62)	EZO RTD	102 (0x66)	EZO HUM	111 (0x6F)
EZO DO	97 (0x61)	EZO PMP	103 (0x67)	EZO PMP-L	109 (0x6D)

Dosing pump

An optional external dosing pump can be added to the Wi-Fi Aquaponics 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

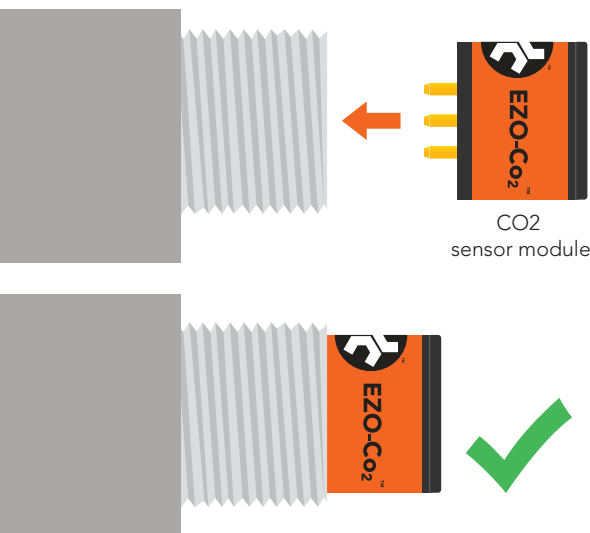
The Atlas-Scientific Wi-Fi hydroponics kit has been designed to upload sensor data to ThingSpeak™, a free, cloud-based data acquisition and visualization platform. You will be required to set up a free account with ThingSpeak™ to upload and visualize the data. With a free account, you can upload data 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 **Connect Co2 sensor module**

The CO2 sensor module has been removed to protect it during transport. Before you begin setting up the aquaponics kit, plug it back into position.



Warning:

Do not mix up sensor modules as each is calibrated to a specific EZO-CO2 device.

Step 2 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 screenshot shows the ThingSpeak login interface. At the top, there is a navigation bar with 'ThingSpeak™', 'Channels', 'Apps', 'Support', 'Commercial Use', and 'How to Buy'. Below the navigation bar, there is a message: 'To use ThingSpeak, you must sign in with your existing MathWorks account or create a new one.' This is followed by two paragraphs of text explaining account types and data transfer options. The main content area features the MathWorks logo, an 'Email' input field, a 'No account? Create one!' link, and a 'Next' button. To the right of the login 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 for 'ALGORITHM DEVELOPMENT SENSOR ANALYTICS'.

Step 3 Create a Channel

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

The screenshot shows the 'My Channels' page on ThingSpeak. The navigation bar is identical to the previous screenshot. Below the navigation bar, there is a 'My Channels' section with a search bar labeled 'Search by tag' and a magnifying glass icon. A prominent green button labeled 'New Channel' is highlighted with a large red arrow pointing upwards. To the right of the 'My Channels' section is a 'Help' section with text explaining how to collect data and sort channels.

New Channel

Name	<input type="text" value="Atlas Sensors"/>
Description	<input type="text"/>
Field 1	<input type="text" value="pH"/> <input checked="" type="checkbox"/>
Field 2	<input type="text" value="DO (mg/L)"/> <input checked="" type="checkbox"/>
Field 3	<input type="text" value="Temp (°C)"/> <input checked="" type="checkbox"/>
Field 4	<input type="text" value="EC (µS/cm)"/> <input checked="" type="checkbox"/>
Field 5	<input type="text" value="Humidity (%)"/> <input checked="" type="checkbox"/>
Field 6	<input type="text" value="CO2 (ppm)"/> <input checked="" type="checkbox"/>

Help

Channel Settings

- **Percentage complete:** Calculated based on data entered into the various fields of a channel. Enter the name, description, location, URL, video, and tags to complete your channel.
- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Field#:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Link to External Site:** If you have a website that contains information about your ThingSpeak channel, specify the URL.
- **Show Channel Location:**
 - **Latitude:** Specify the latitude position in decimal degrees. For example, the latitude of the city of London is 51.5072.

Fill out the highlighted boxes. (Be sure to click on the checkboxes to enable **fields 2 – 6**)
For reference, this is what we entered.

Name **Atlas Sensors**
Field 1 **pH**
Field 2 **DO (mg/L)**
Field 3 **Temp (°C)**
Field 4 **EC (µS/cm)**
Field 5 **Humidity (%)**
Field 6 **CO2 (ppm)**

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

Step 4 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 on ThingSpeak. At the top, there is a navigation bar with 'ThingSpeak™', 'Channels', 'Apps', and 'Support'. On the right, there are links for 'Commercial Use' and 'How to Buy'. Below the navigation bar, the 'My Channels' section features a 'New Channel' button and a search bar. A table lists channels, with 'Atlas Sensors' highlighted. Below the table, there are tabs for 'Private', 'Public', 'Settings', 'Sharing', 'API Keys', and 'Data Import / Export'. A large red arrow points to the 'API Keys' tab.

Name	Created	Updated
Atlas Sensors	2020-02-14	2020-05-11 23:04

Help

Collect data in a ThingSpeak channel from a device, from another channel, or from the web.

Click **New Channel** to create a new ThingSpeak channel.

Click on the column headers of the table to sort by the entries in that column or click on a tag to show channels with that tag.

Learn to [create channels](#), explore and transform data.

Learn more about [ThingSpeak Channels](#).

The screenshot shows the 'Atlas Sensors' channel page on ThingSpeak. The navigation bar is the same as in the previous screenshot. Below it, the channel name 'Atlas Sensors' is displayed. A red box highlights the 'Channel ID: xxxxxx' and the 'Author:' and 'Access: Private' information. Below this, there are tabs for 'Private View', 'Public View', 'Channel Settings', 'Sharing', 'API Keys', and 'Data Import / Export'. The 'API Keys' tab is selected, and a red box highlights the 'Write API Key' section, which contains a text input field with the key 'XXXXXXXXXXXXXXXXXX' and a 'Generate New Write API Key' button. To the right, there is a 'Help' section and 'API Keys Settings'.

Channel ID: xxxxxx

Author:

Access: Private

Private View Public View Channel Settings Sharing API Keys Data Import / Export

Write API Key

Key: XXXXXXXXXXXXXXXXXXXX

Generate New Write API Key

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

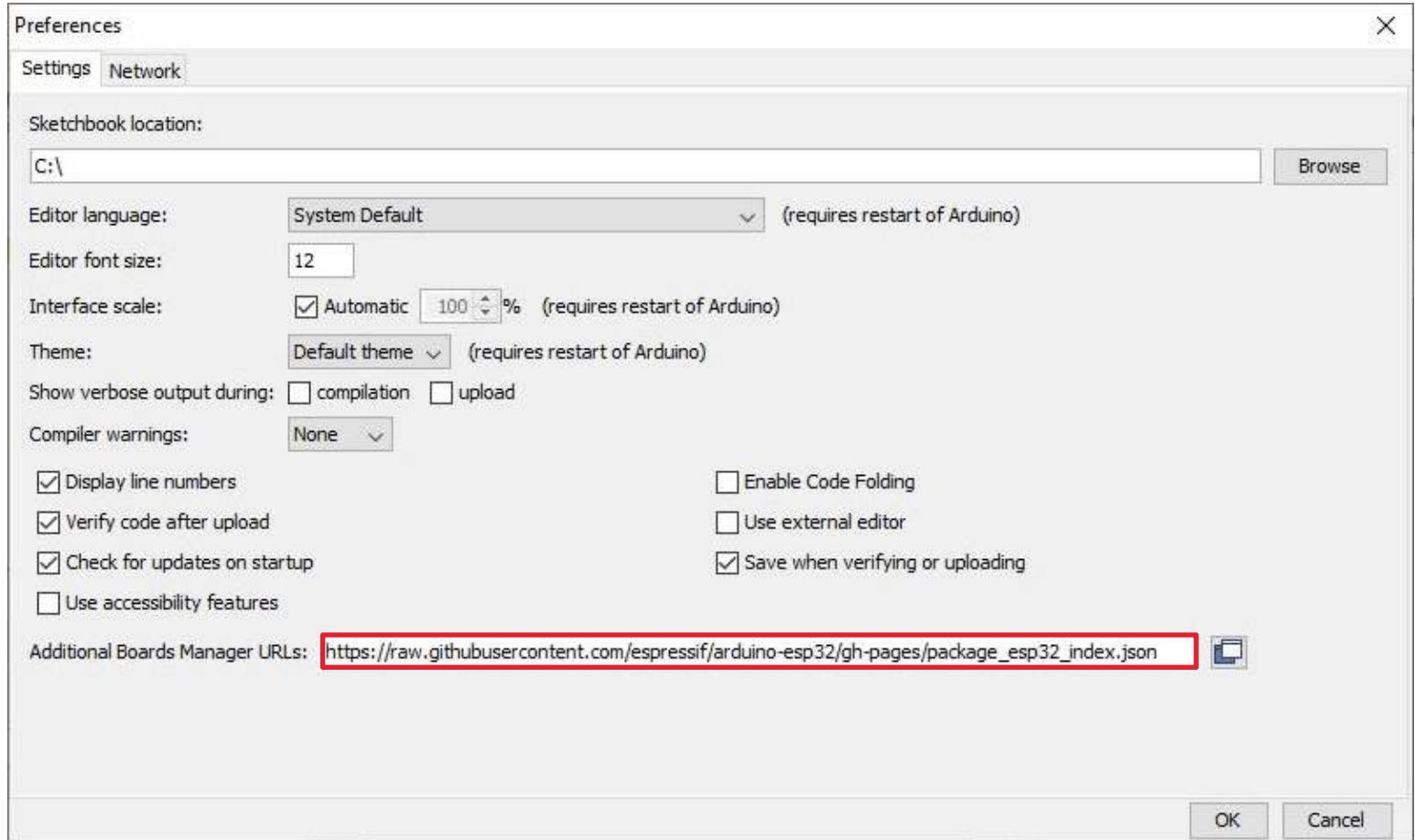
- **Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key**.
- **Read API Keys:** Use this key to allow other people to view your private channel

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

Step 5 Make sure your Arduino IDE libraries are up to date

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

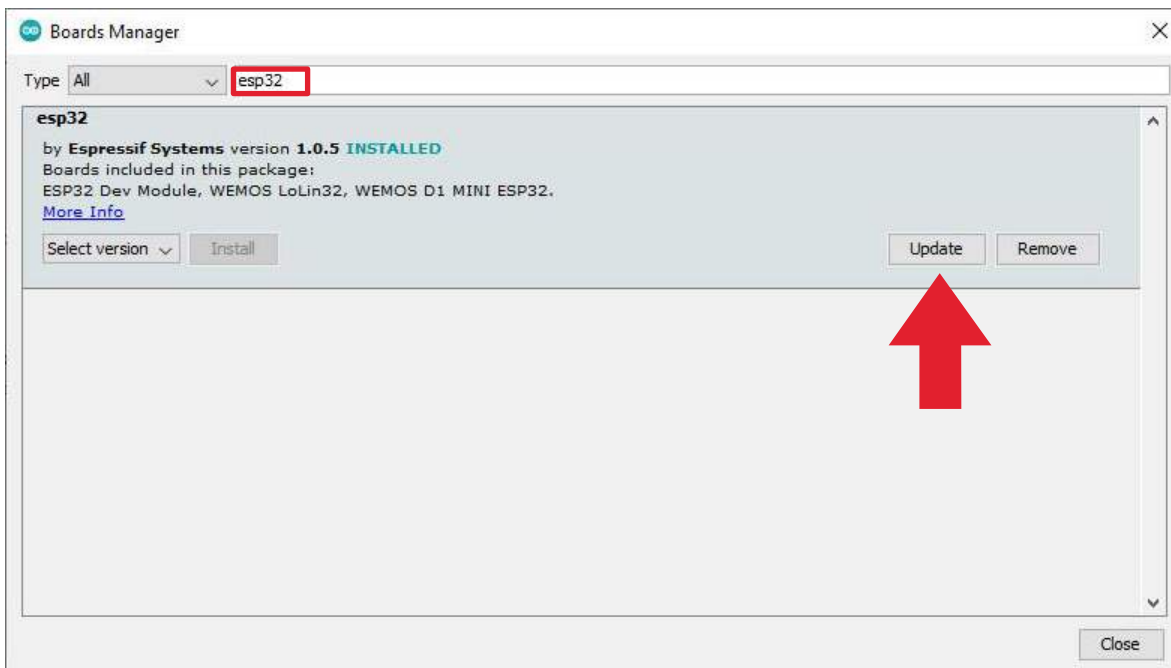
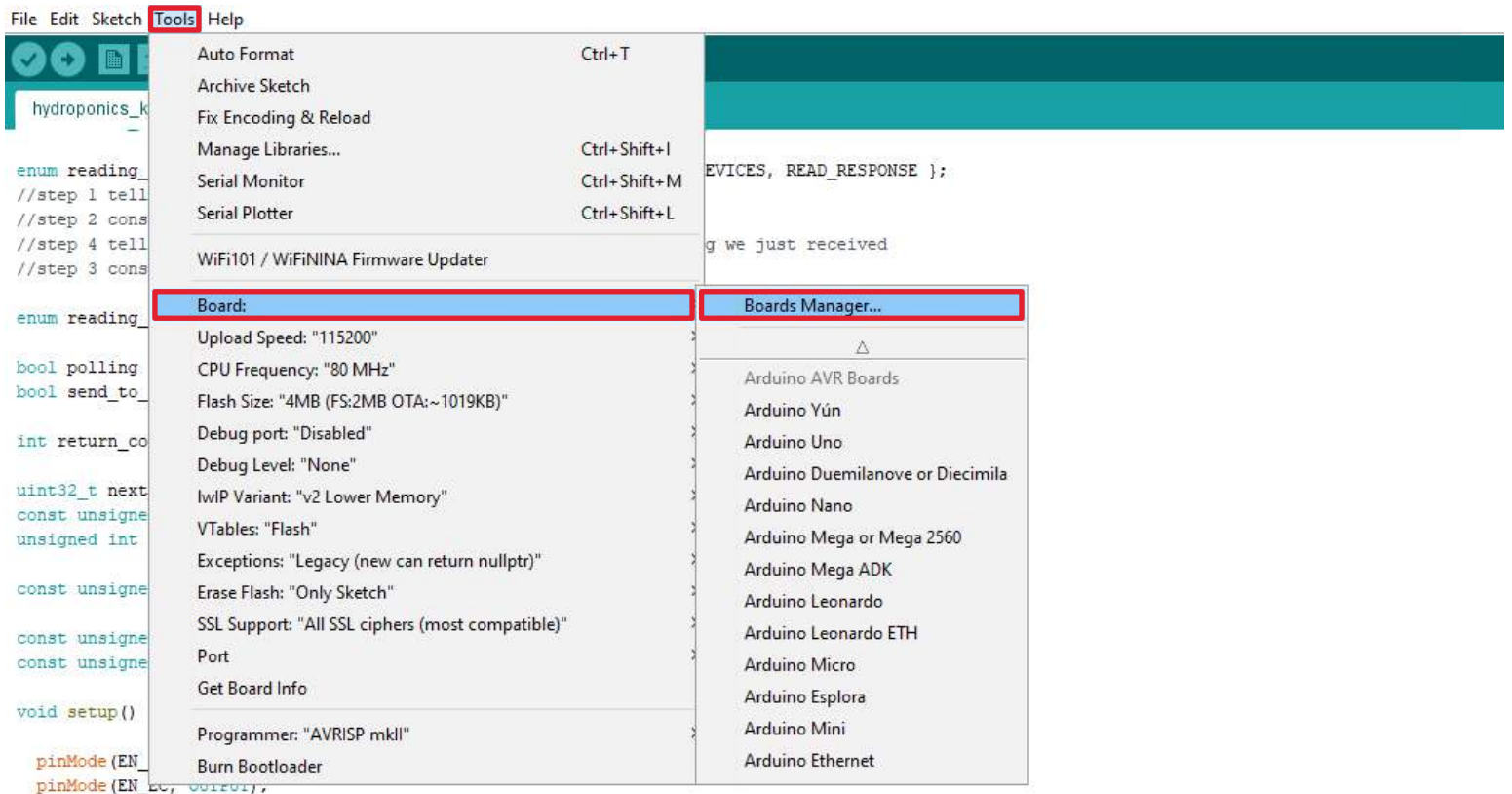
In the IDE, go to **File > Preferences**
Locate the **Additional Boards Manager URLs** text box.



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)

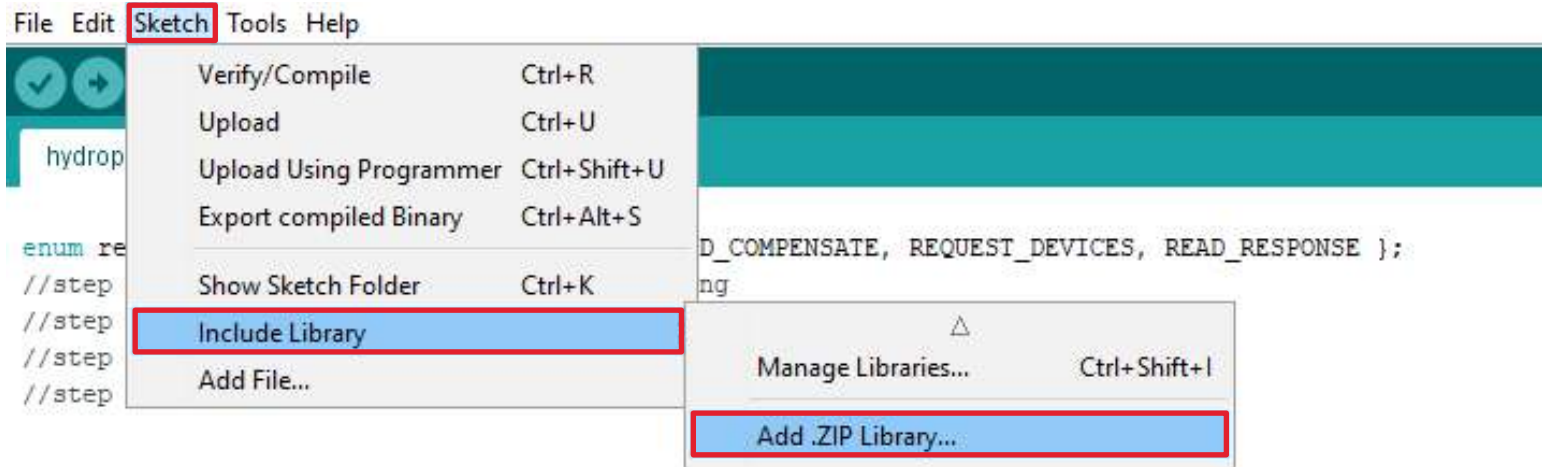
C Download the ThingSpeak library for Arduino

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

Don't unzip it!

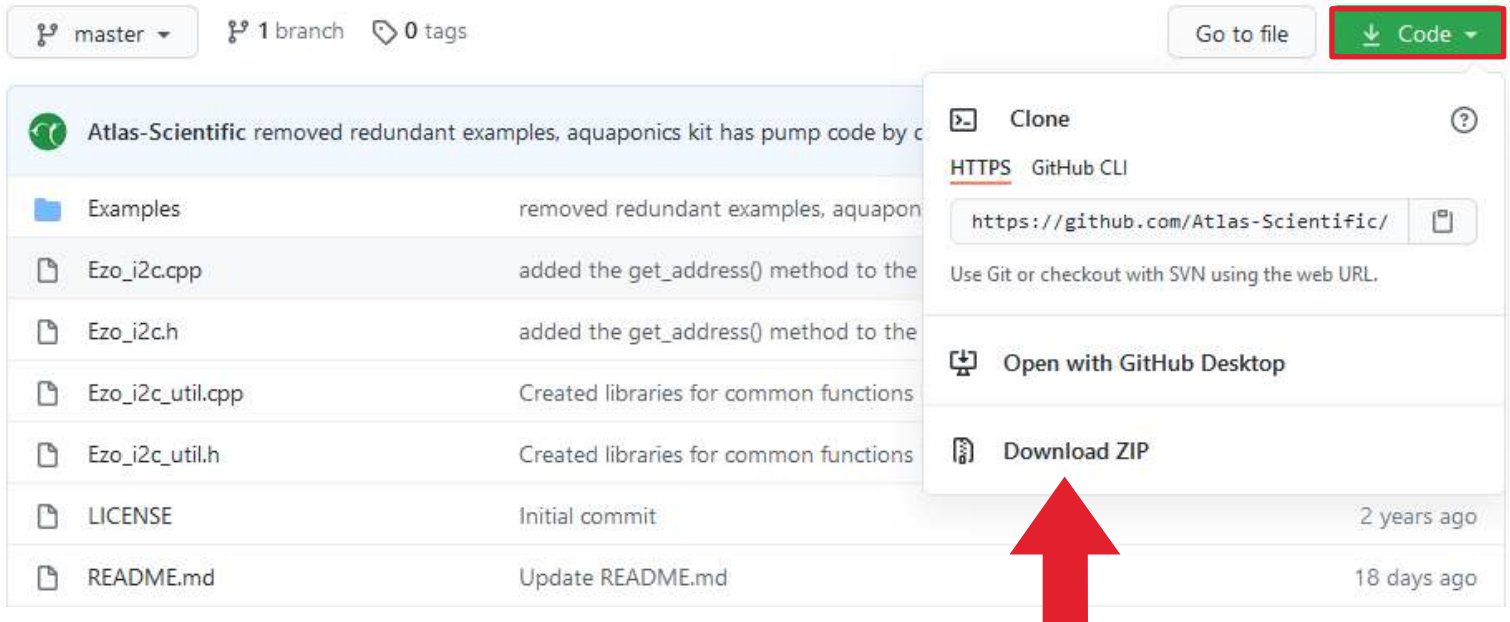
Import the .ZIP file into your Arduino IDE.

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!

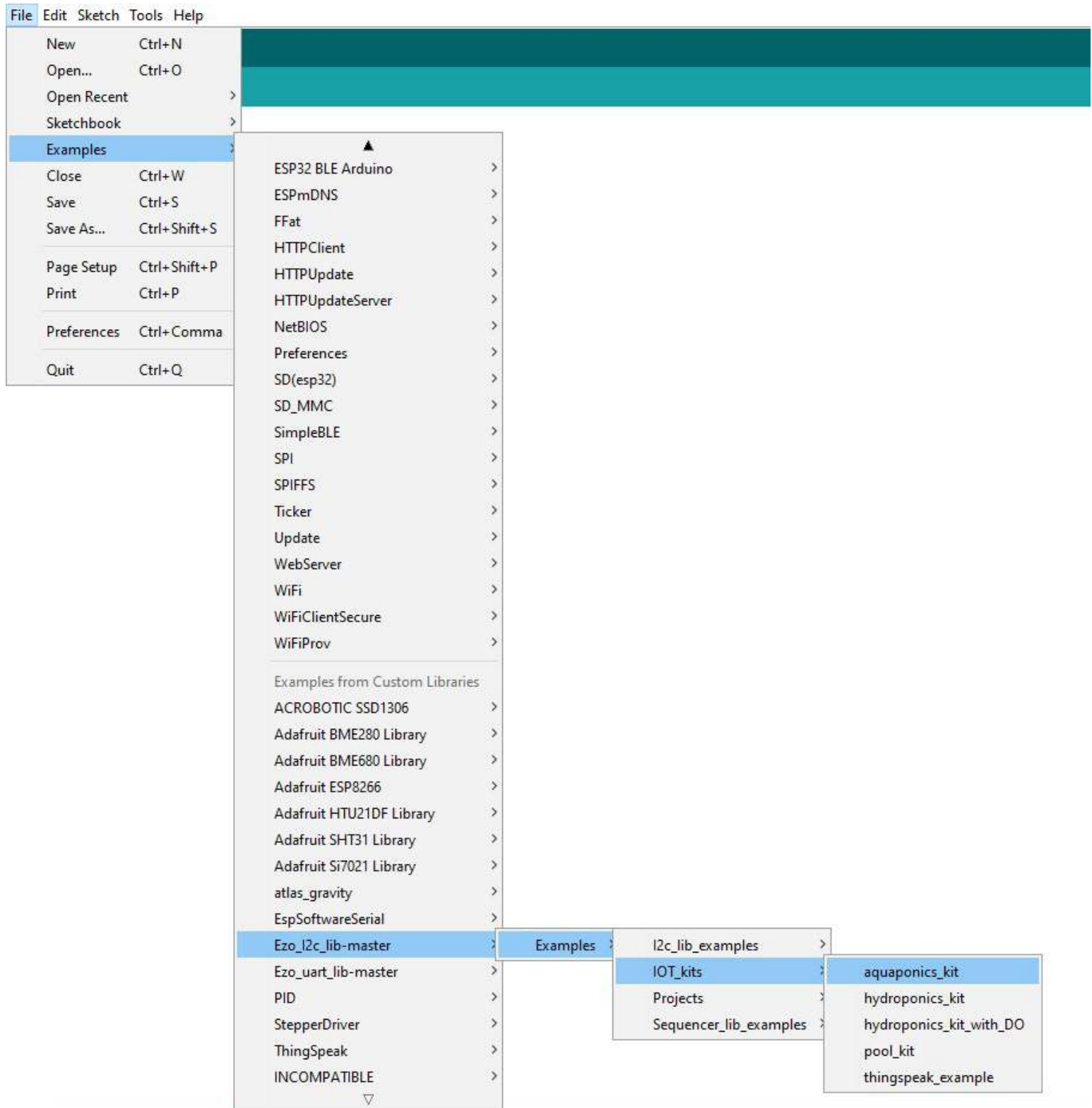
Import the .ZIP file to your Arduino IDE.

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

Step 6 Flash the Aquaponics 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 > aquaponics_kit



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)

```
1 #include <iot_cmd.h>
2 #include <WiFi.h> //include wifi library
3 #include "ThingSpeak.h" //include thingspeak library
4 #include <sequencer4.h> //imports a 4 function sequencer
5 #include <sequencer1.h> //imports a 1 function sequencer
6 #include <Ezo_i2c_util.h> //brings in common print statements
7 #include <Ezo_i2c.h> //include the EZO I2C library from https://github.com/Atlas-Scientific/Ezo_I2c_lib
8 #include <Wire.h> //include arduinos i2c library
9
10 WiFiClient client; //declare that this device connects to
11
12 //-----Fill in your Wi-Fi / ThingSpeak Credentials-----
13 const String ssid = "Wifi Name"; //The name of the Wi-Fi network you
14 const String pass = "Wifi Password"; //Your WiFi network password
15 const long myChannelNumber = 1234566; //Your Thingspeak channel number
16 const char * myWriteAPIKey = "XXXXXXXXXXXXXXXXXX"; //Your ThingSpeak Write API Key
17 //-----
```

C 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.

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


Step 7 Setting up the HUZZAH board

A Set the target CPU to flash

Tools > Board > ESP32 Arduino > Adafruit Esp32 Feather

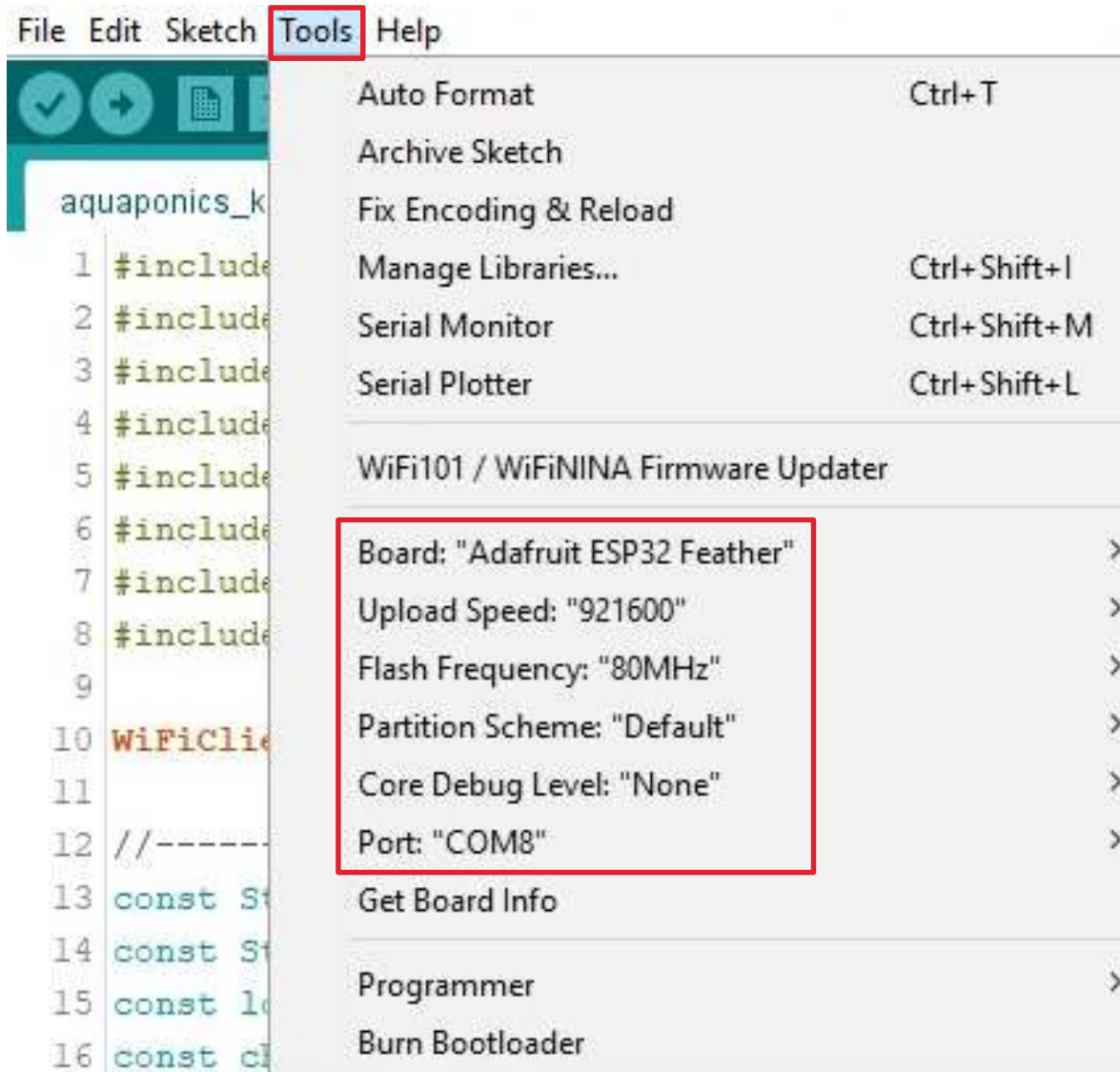
The screenshot shows the Arduino IDE interface with the following menu path highlighted: **Tools > Board > ESP32 Arduino > Adafruit Esp32 Feather**. The Boards Manager window is open, showing a list of boards. The 'Adafruit ESP32 Feather' board is selected and highlighted in blue. The background shows a sketch with code for an Ezo_board, including comments like '//create a PH circuit object' and '//an array of boards used'.

```
1 #include <Ezo_board.h>
2 #include <WiFi.h>
3 #include <ThingSpeak.h>
4 #include <sequencer.h>
5 #include <sequencer.h>
6 #include <Ezo_board.h>
7 #include <Ezo_board.h>
8 #include <Ezo_board.h>
9 #include <Ezo_board.h>
10 #include <Ezo_board.h>
11 #include <Ezo_board.h>
12 //-----
13 const char ssid[] = "Ezo";
14 const char pass[] = "12345678";
15 const int ledPin = 13;
16 const int ledPin2 = 14;
17 //-----
18 Ezo_board PH = Ezo_board(99, "PH"); //create a PH circuit object
19 Ezo_board DO = Ezo_board(97, "DO"); //create a DO circuit object
20 Ezo_board RTD = Ezo_board(102, "RTD"); //create an RTD circuit object
21 Ezo_board EC = Ezo_board(100, "EC"); //create an EC circuit object
22 Ezo_board PMP = Ezo_board(103, "PMP"); //create an PMP circuit object
23 Ezo_board HUM = Ezo_board(111, "HUM"); //create a HUM circuit object
24 Ezo_board CO2 = Ezo_board(105, "CO2"); //create a CO2 circuit object
25
26 Ezo_board device_list[] = { //an array of boards used
27   PH,
28   DO,
29   RTD,
30   EC,
31   PMP,
32   HUM,
33   CO2
34 };
35
36 Ezo_board* default_board = &device_list[0]; //used to store the board
37
38 //gets the length of the array automatically so we dont have to change it
39 const uint8_t device_list_len = sizeof(device_list) / sizeof(Ezo_board);
40
```

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.)



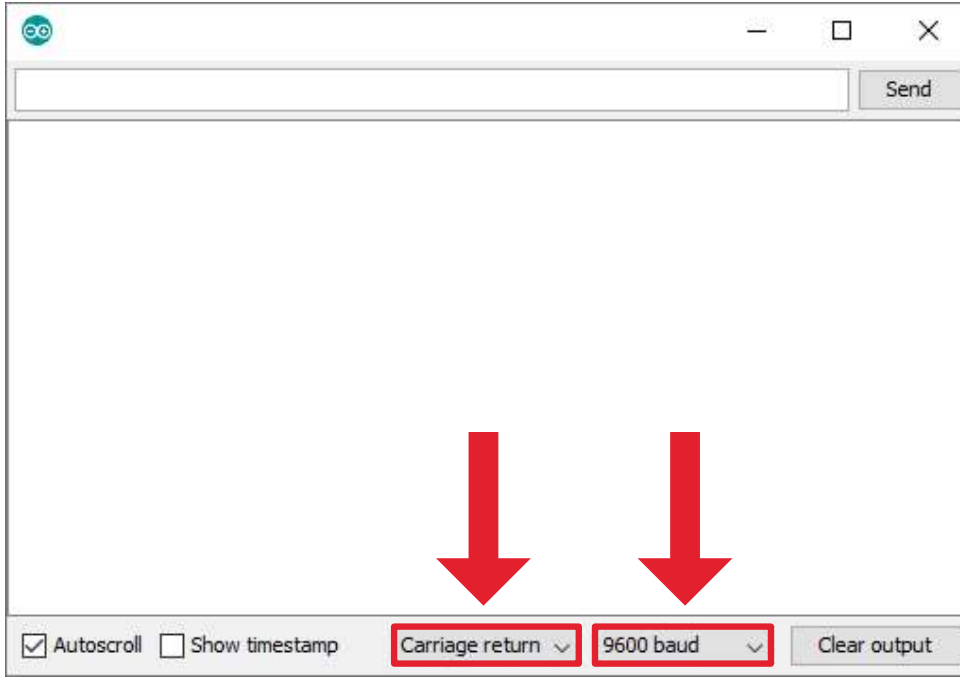
C Compile and upload



Compile and upload the code.

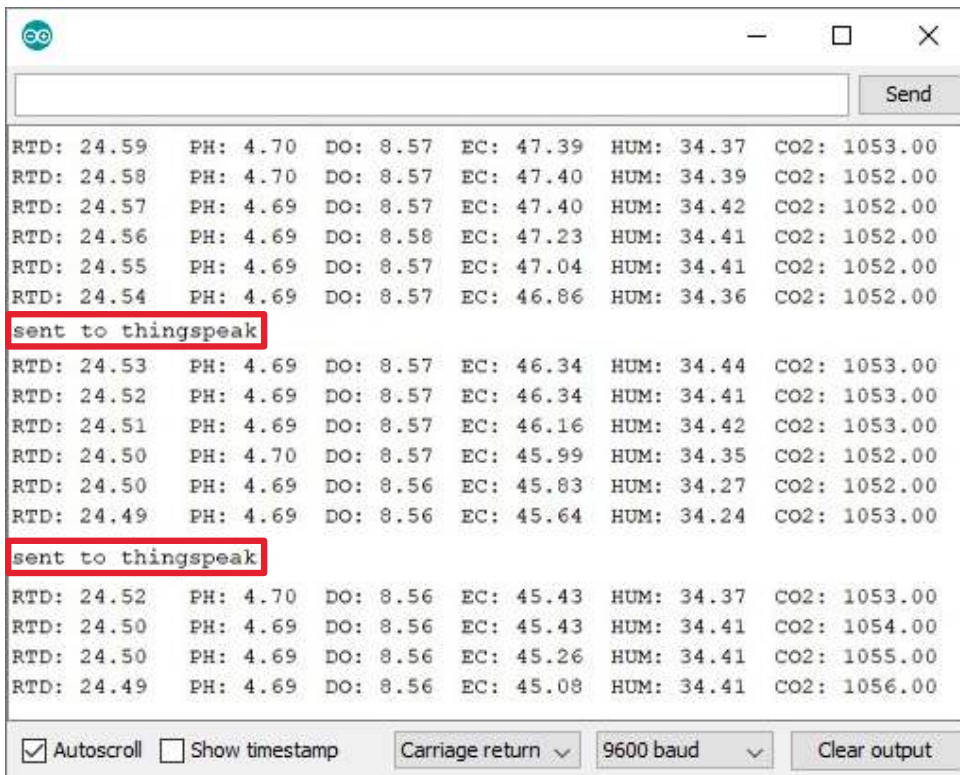
Step 8 See the readings

Open your Arduino serial monitor.
(You must have the serial monitor set to the com port from the Adafruit HUZAZH32.)

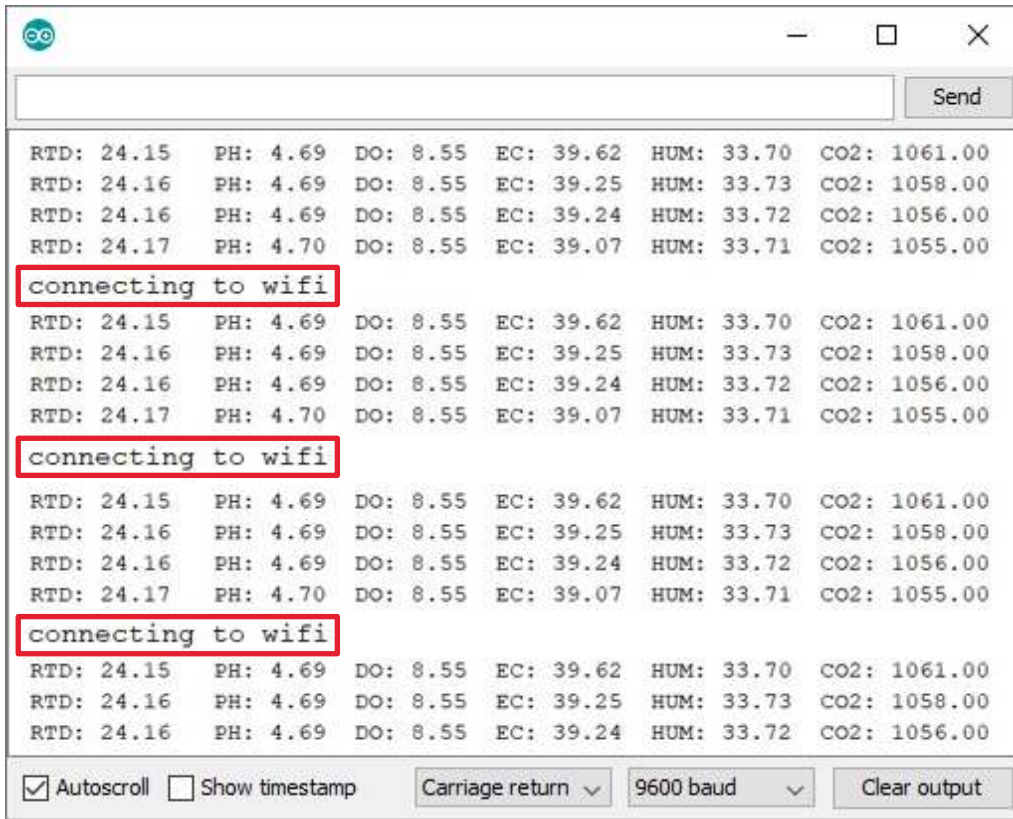


Set to **carriage return** and **9600 baud**.

The Wi-Fi Aquaponics 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 'Send' button at the top right. The output consists of four identical blocks of sensor data, each preceded by a red-bordered box containing the text 'connecting to wifi'. The sensor data for each block is as follows:

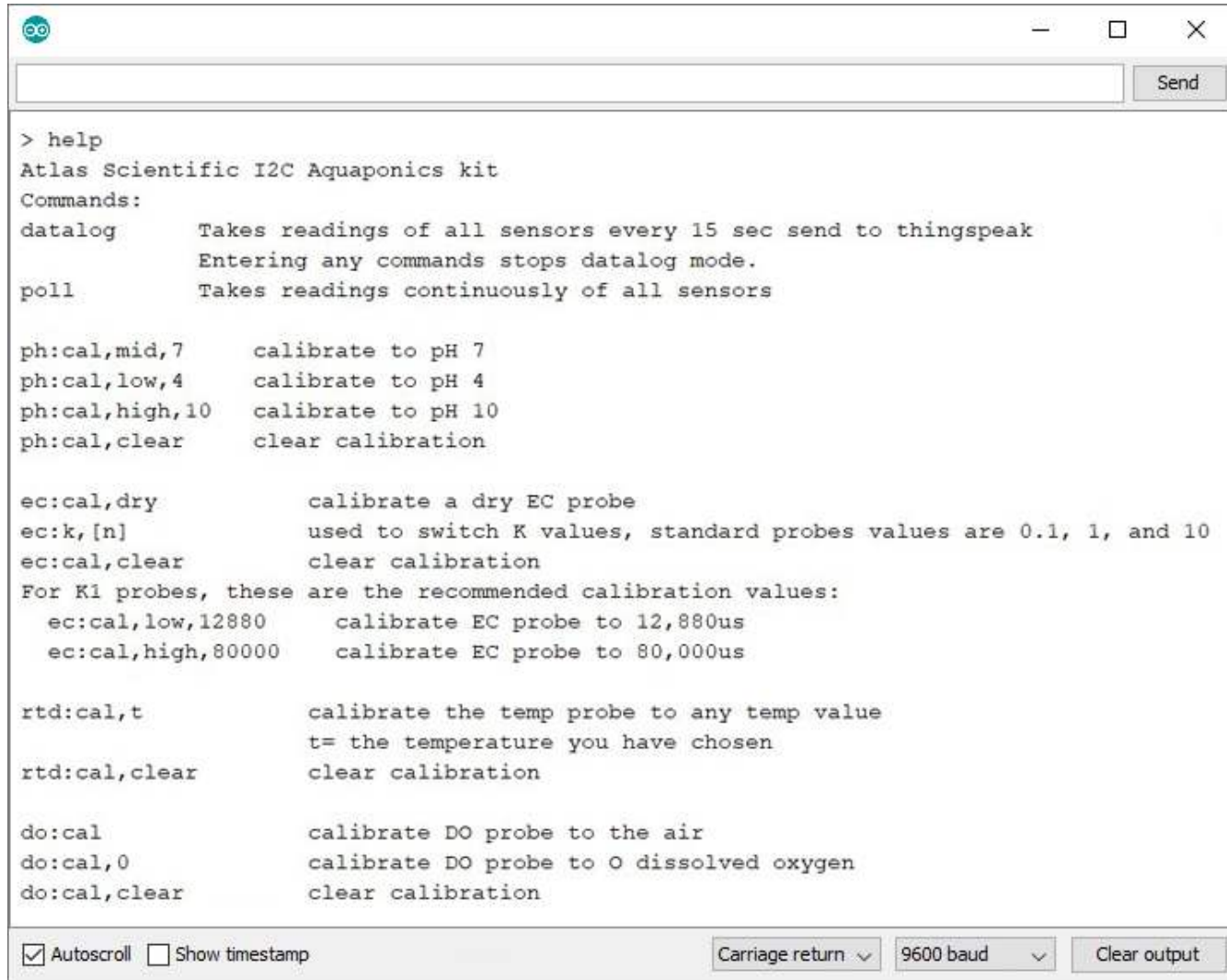
RTD: 24.15	PH: 4.69	DO: 8.55	EC: 39.62	HUM: 33.70	CO2: 1061.00
RTD: 24.16	PH: 4.69	DO: 8.55	EC: 39.25	HUM: 33.73	CO2: 1058.00
RTD: 24.16	PH: 4.69	DO: 8.55	EC: 39.24	HUM: 33.72	CO2: 1056.00
RTD: 24.17	PH: 4.70	DO: 8.55	EC: 39.07	HUM: 33.71	CO2: 1055.00

At the bottom of the terminal window, there are control options: Autoscroll, Show timestamp, a dropdown menu set to 'Carriage return', a dropdown menu set to '9600 baud', and a 'Clear output' button.

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

Step 9 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 Aquaponics 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

do:cal        calibrate DO probe to the air
do:cal,0      calibrate DO probe to 0 dissolved oxygen
do:cal,clear  clear calibration
```

Autoscroll Show timestamp Carriage return 9600 baud Clear output

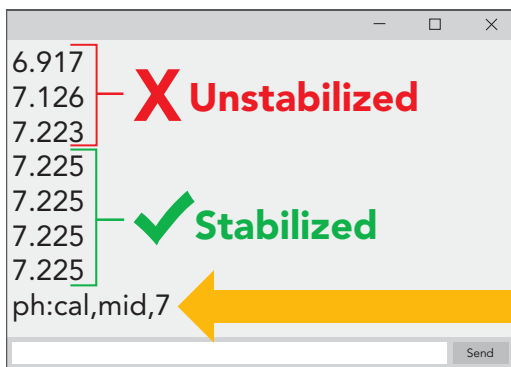
A The poll command

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)

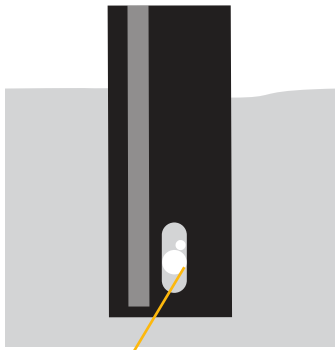
Example, if you have a K 0.1 conductivity probe issue the command **ec:k,0.1**

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

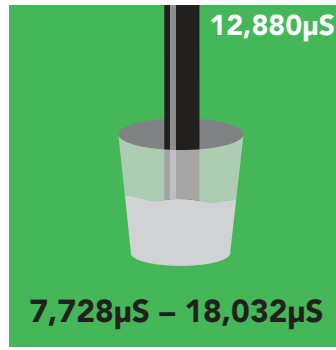


Make sure that the probe is dry before issuing this command, **ec:cal,dry**

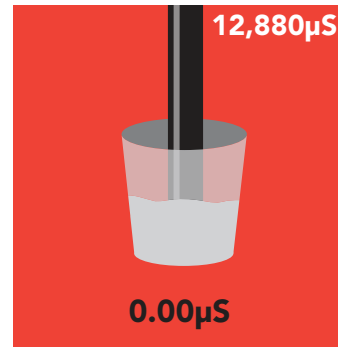
Once the dry calibration has been completed, place the probe into a small cup of the low point calibration solution. Shake the probe to make sure you do not have trapped air bubbles in the sensing area. You should see readings that are off by **1 – 40%** from the stated value of the calibration solution.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



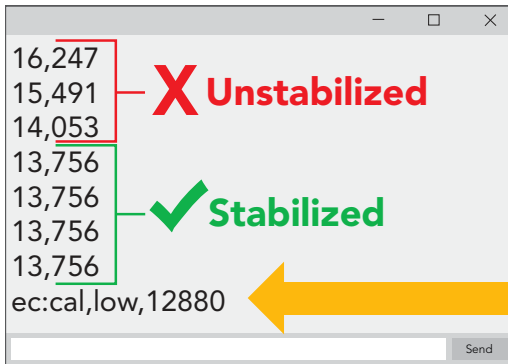
Trapped air in sensing area (shake to remove)



+/- 40%

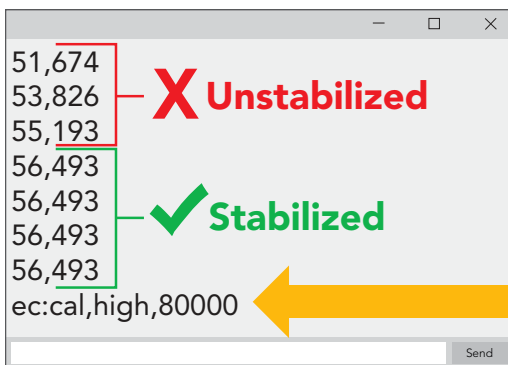


check probe connection, you cannot calibrate to 0.



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

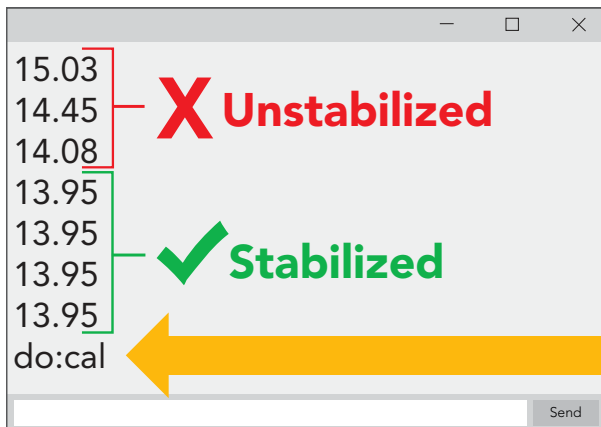
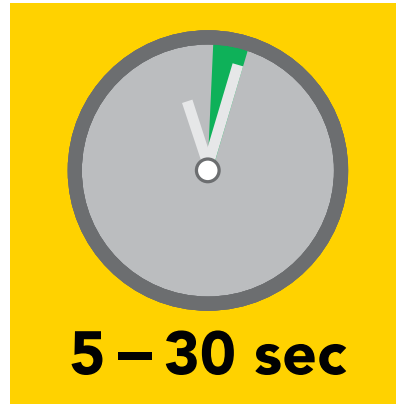
Rinse off the probe before calibrating to the high point. Pour a small amount of the high point calibration solution into a cup. Shake the probe to remove trapped air. Again, the readings may be off by **1 – 40%** Wait for readings to stabilize.



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

D Calibrate Dissolved Oxygen

Let the Dissolved Oxygen probe sit, exposed to air until the readings stabilize. (*small movement from one reading to the next is normal*).



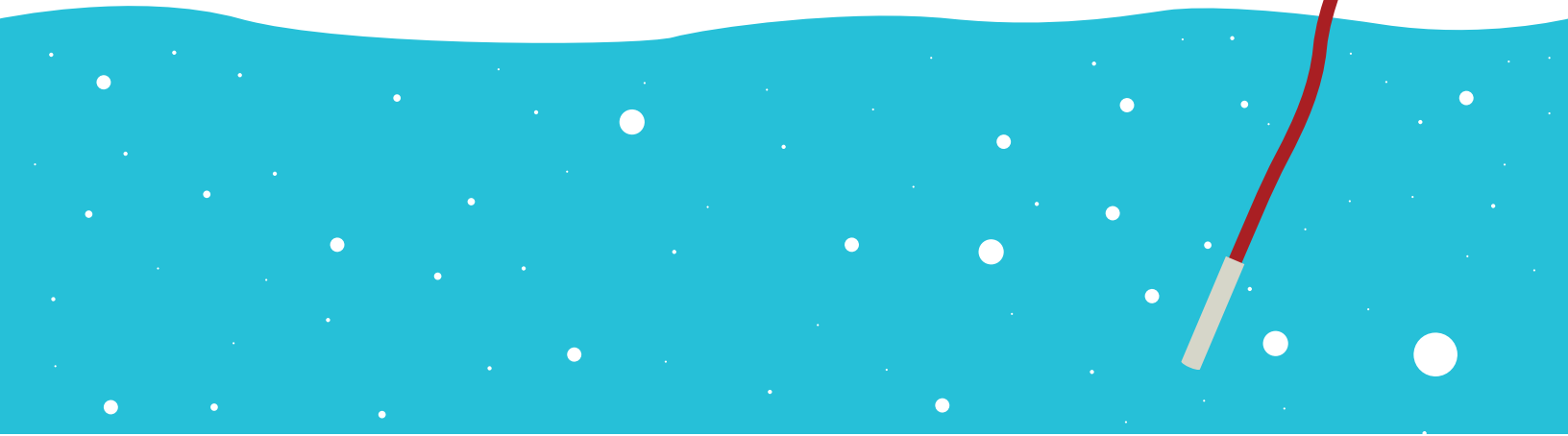
Once the readings have stabilized, issue the calibration command.
do:cal

After calibration is complete, you should see readings between **9.09 – 9.1X mg/L**.
(*only if temperature, salinity and pressure compensation are at default values*)

E 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

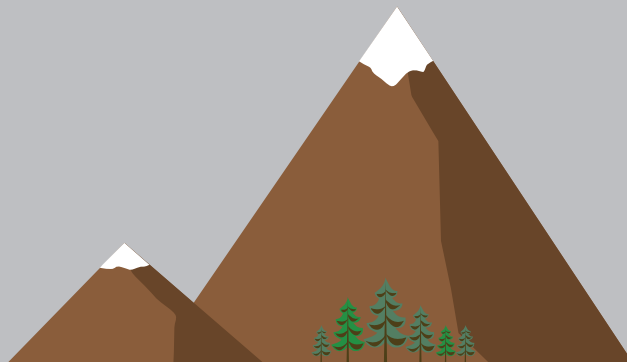


Elevation in meters

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

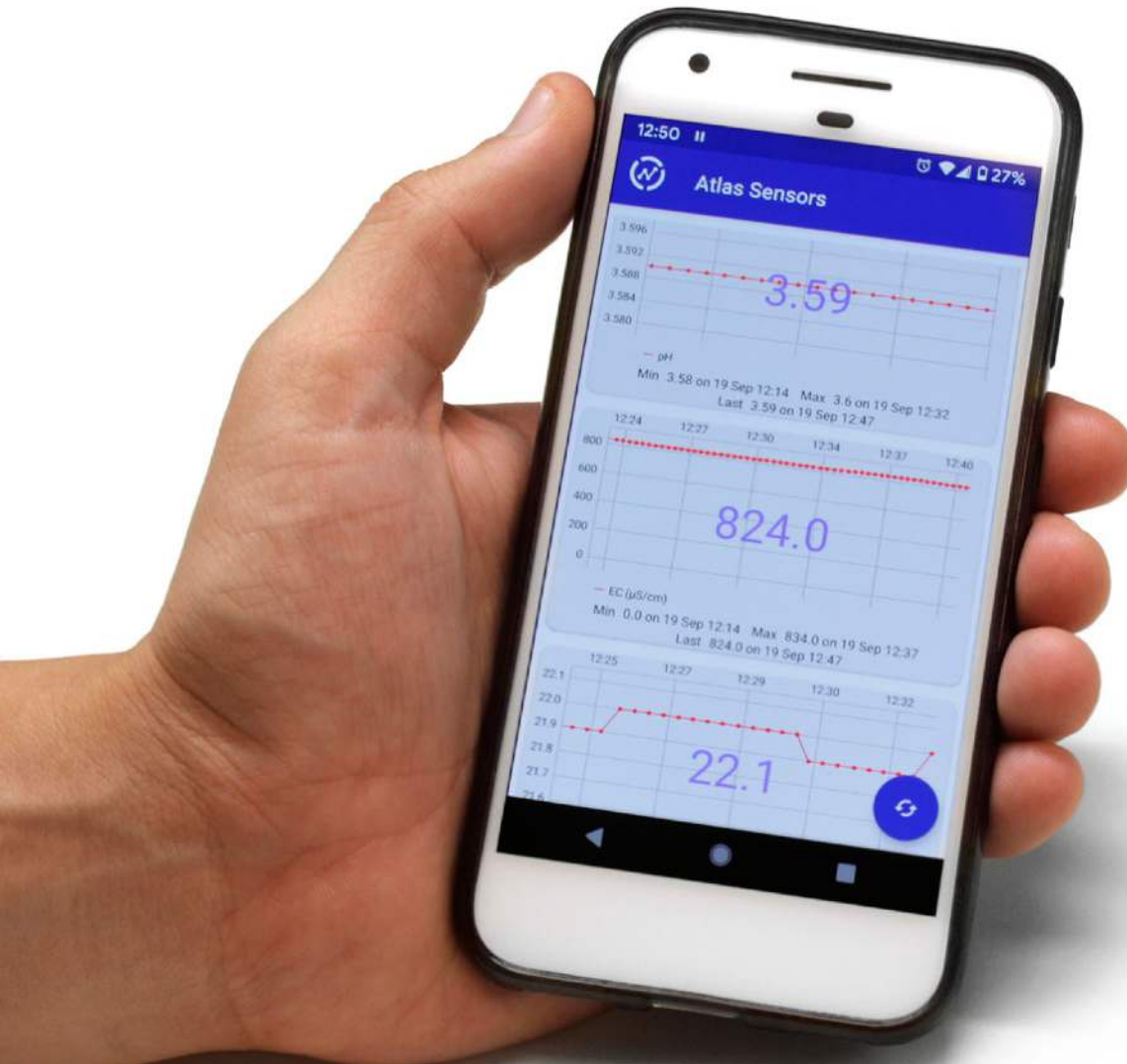


Calibration Complete

Step 10 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!