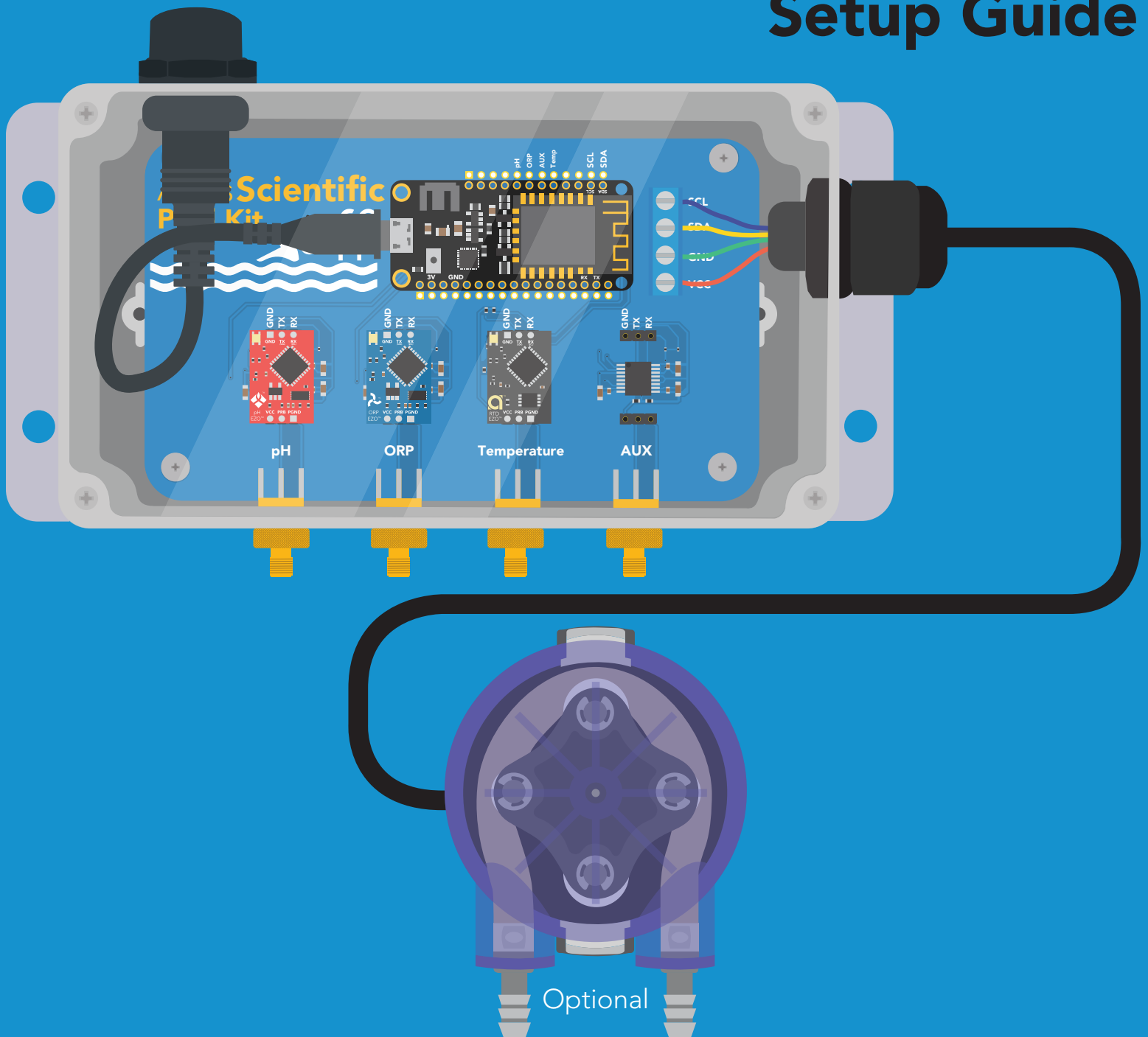


Wi-Fi Pool Kit

Setup Guide

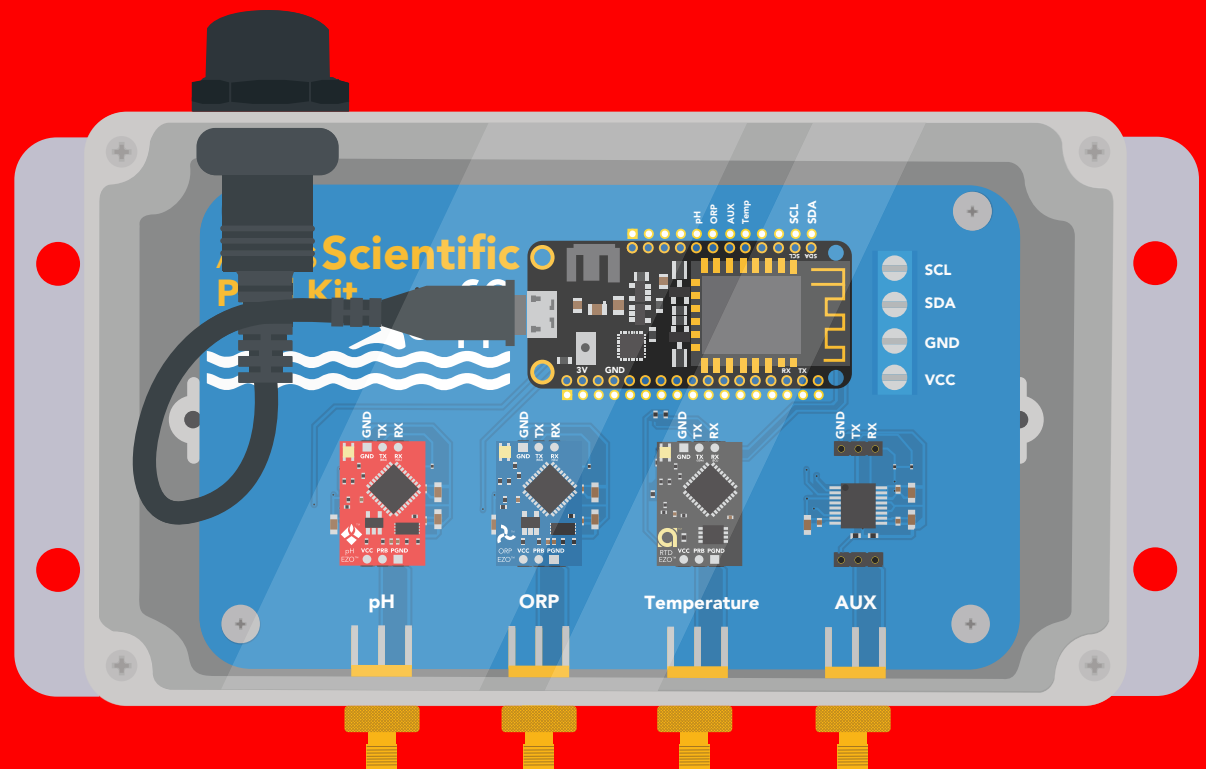


STOP

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)

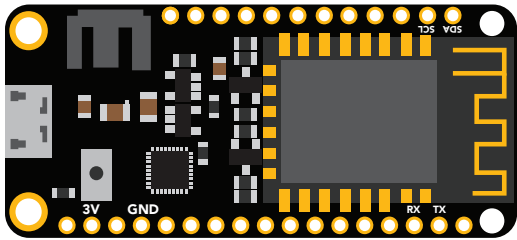
Operating principle

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

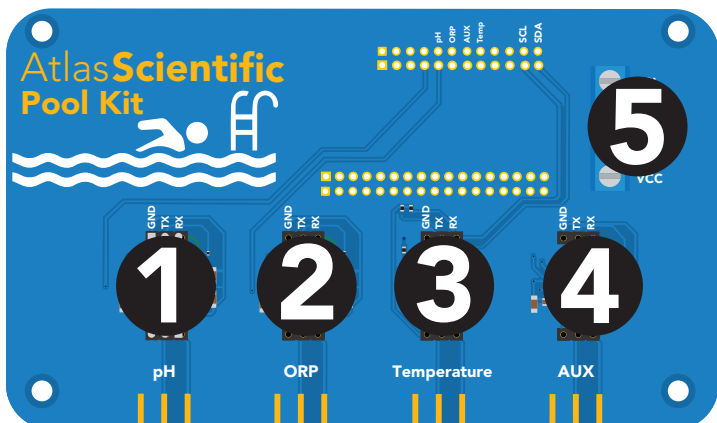
The Wi-Fi Pool Kit is controlled using an Adafruit Feather HUZZAH32 as its CPU. The HUZZAH is programmed using the Arduino IDE and uses an onboard ESP32 as its Wi-Fi transmitter. [Adafruit HUZZAH32 datasheet](#).



Sensor ports

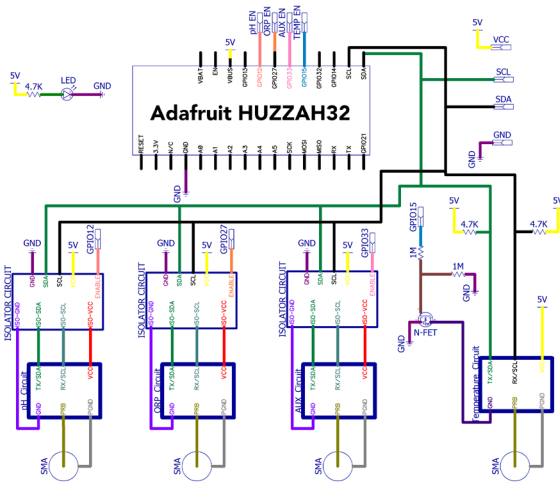
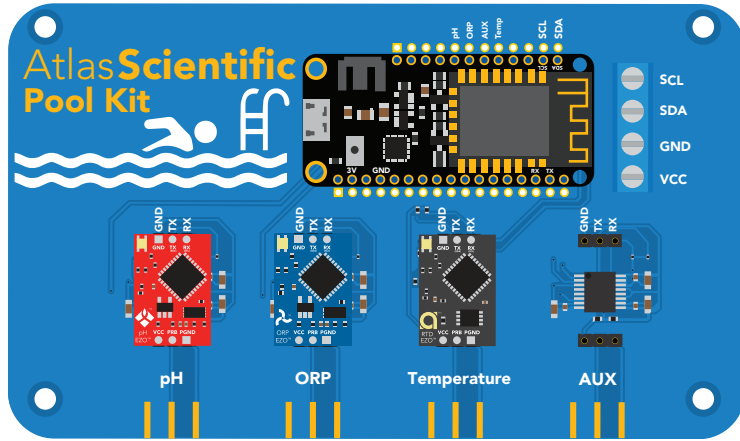
The Wi-Fi Pool Kit PCB has 5 sensor ports. Three of the ports are electrically isolated. The isolated ports are marked pH, ORP, and AUX. The isolated ports are needed to take noise-free electrochemical readings. Because the sensing element of a temperature sensor is never in direct contact with the water, 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.

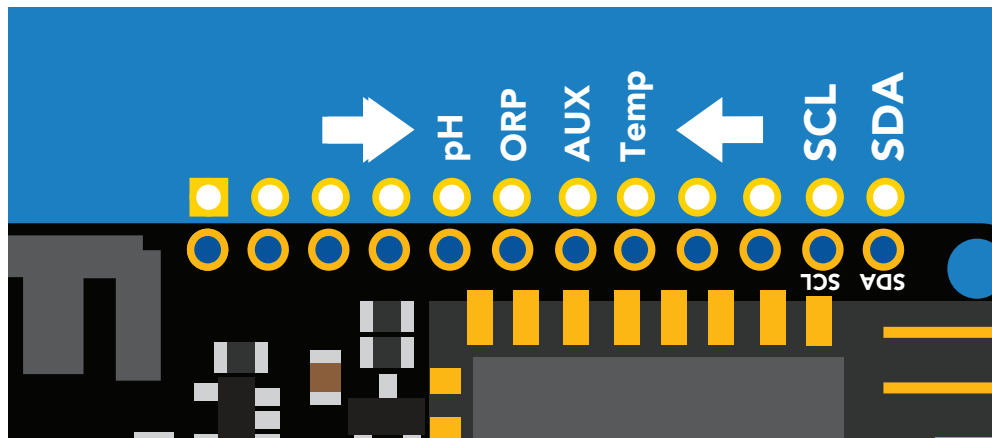


PCB

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, ORP 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	State	Sensor Power
pH EN	LOW	ON
ORP EN	LOW	ON
Aux EN	LOW	ON
Temp EN	HIGH	ON

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

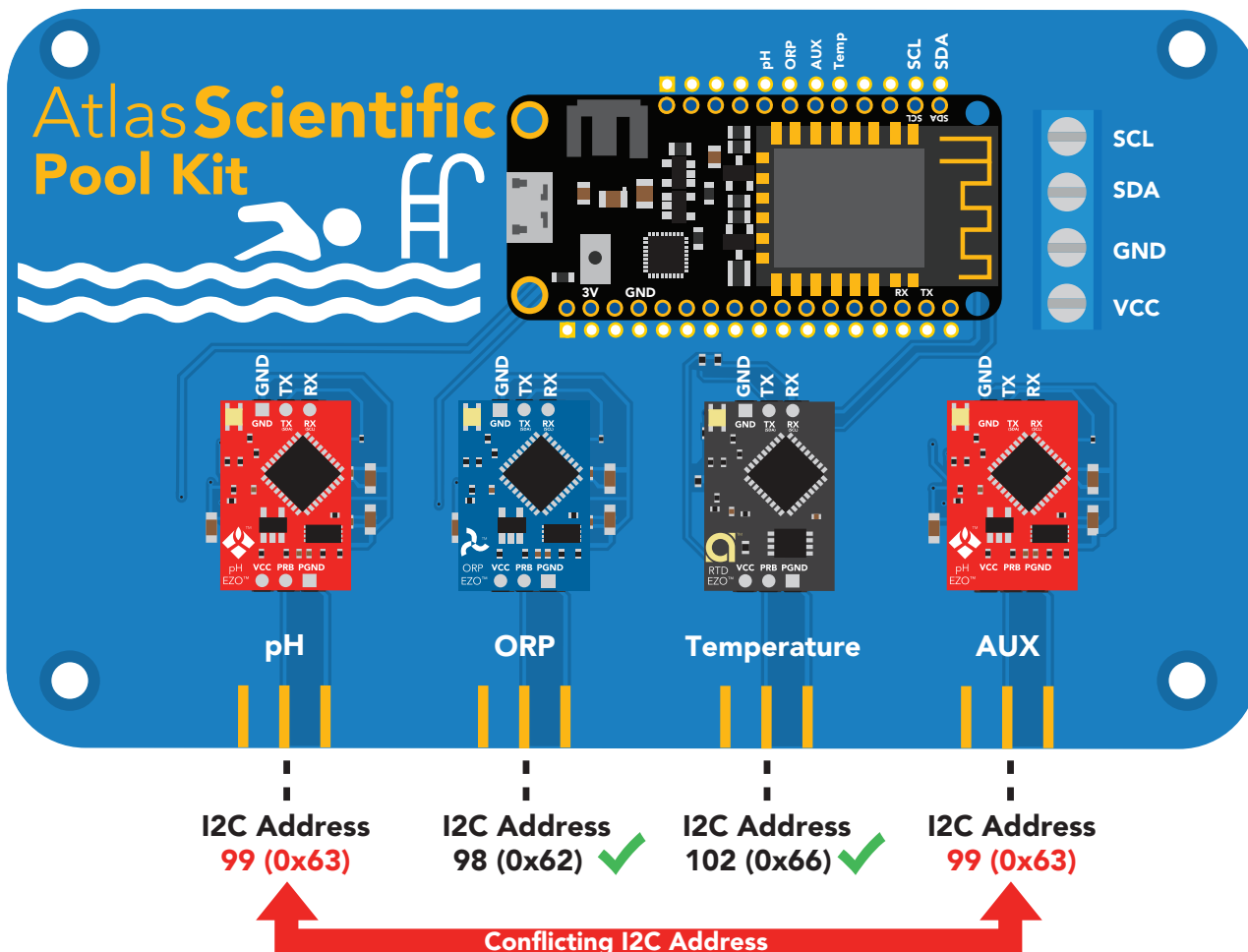
Data protocol

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 ORP 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.

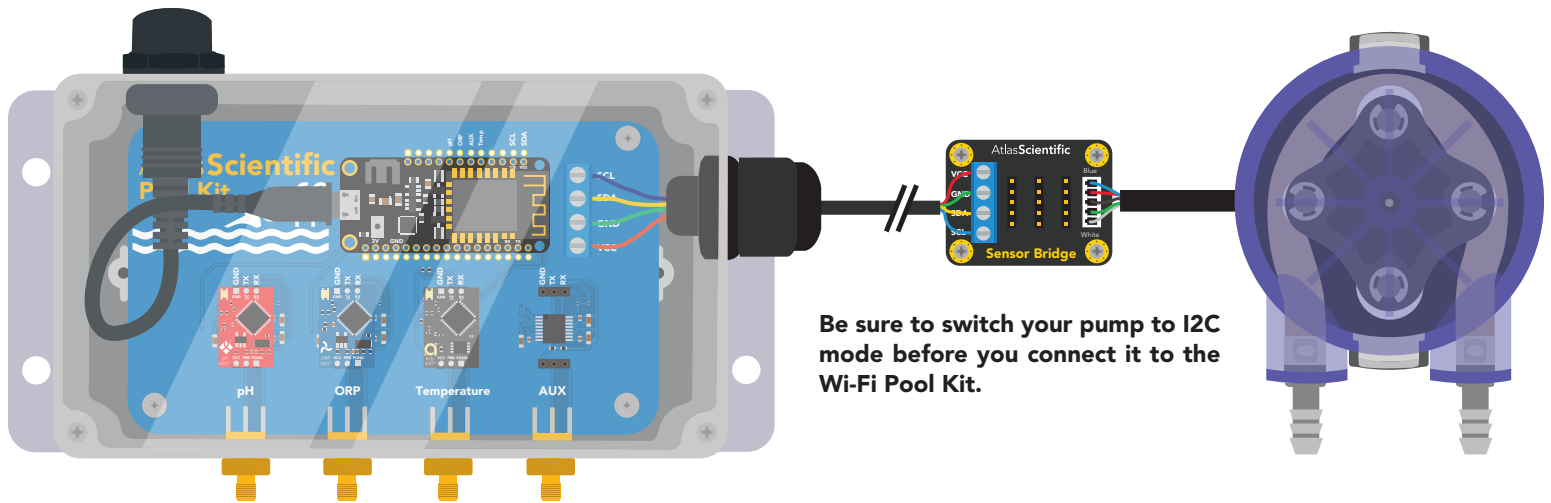


This table lists the default I2C address of components commonly added to this kit.

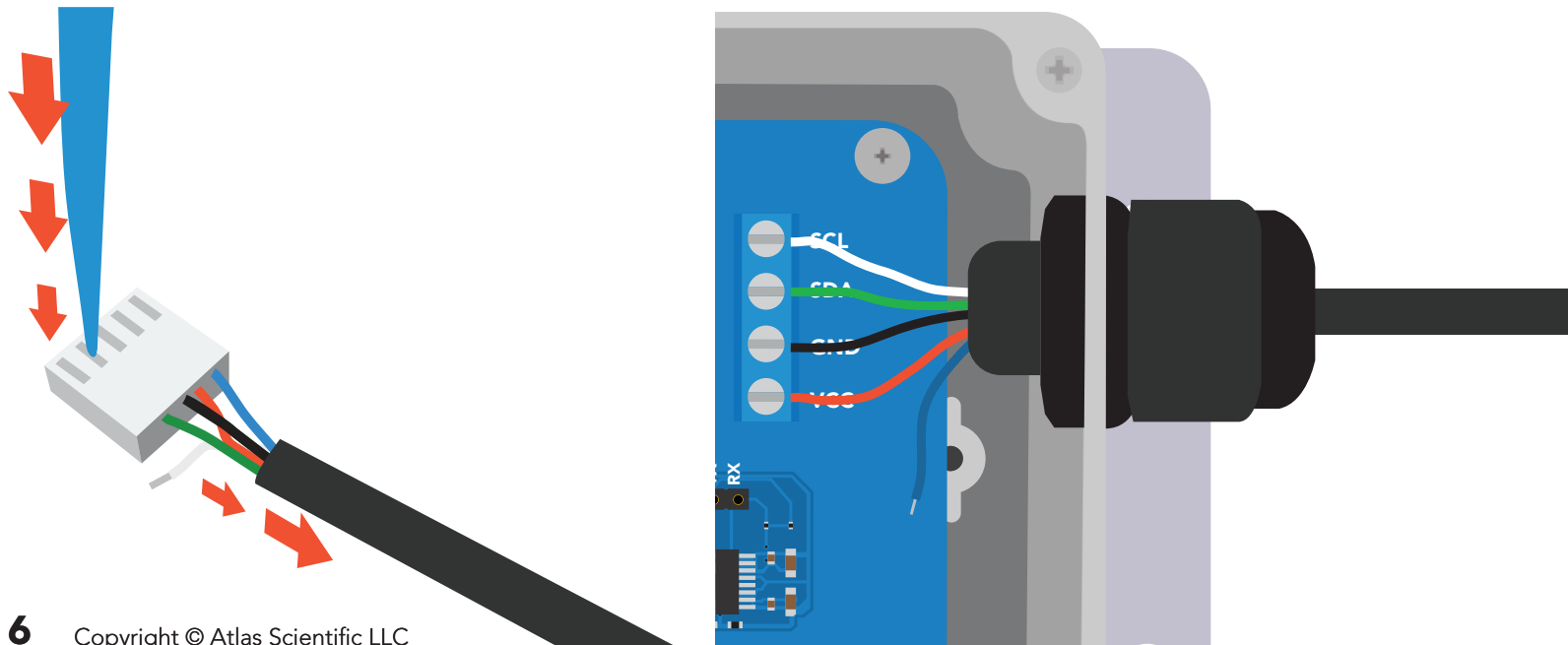
Device	I2C Address	Device	I2C Address
EZO pH	99 (0x63)	EZO EC	100 (0x64)
EZO ORP	98 (0x62)	EZO RTD	102 (0x66)
EZO DO	97 (0x61)	EZO PMP-L	109 (0x6D)

Dosing pump

An optional dosing pump can be added to the Wi-Fi Pool Kit. Using both the [Large Embedded Dosing Pump](#) and our sensor bridge is the simplest way to add on a dosing pump.



You can directly connect an EZO Pump to the Wi-Fi Pool Kit without the sensor bridge however you must remove the data cable connector and manually put the pump into I2C mode.



Uploading sensor data to the cloud

The Atlas-Scientific Wi-Fi Pool 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 a 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**

My Channels

New Channel

Search by tag

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.

New Channel

Name Atlas Sensors

Description

Field 1 pH

Field 2 ORP (mV)

Field 3 Temp (°C)

Help

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

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.

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 **ORP (mV)**
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.

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. 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. Below that, the 'Author:' and 'Access: Private' are listed. There are tabs for 'Private View', 'Public View', 'Channel Settings', 'Sharing', 'API Keys', and 'Data Import / Export'. The 'API Keys' tab is selected. Below the tabs, the 'Write API Key' section is highlighted with a yellow box. It contains a 'Key' field with the value 'XXXXXXXXXXXXXXXXXXXX' and a 'Generate New Write API Key' button.

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

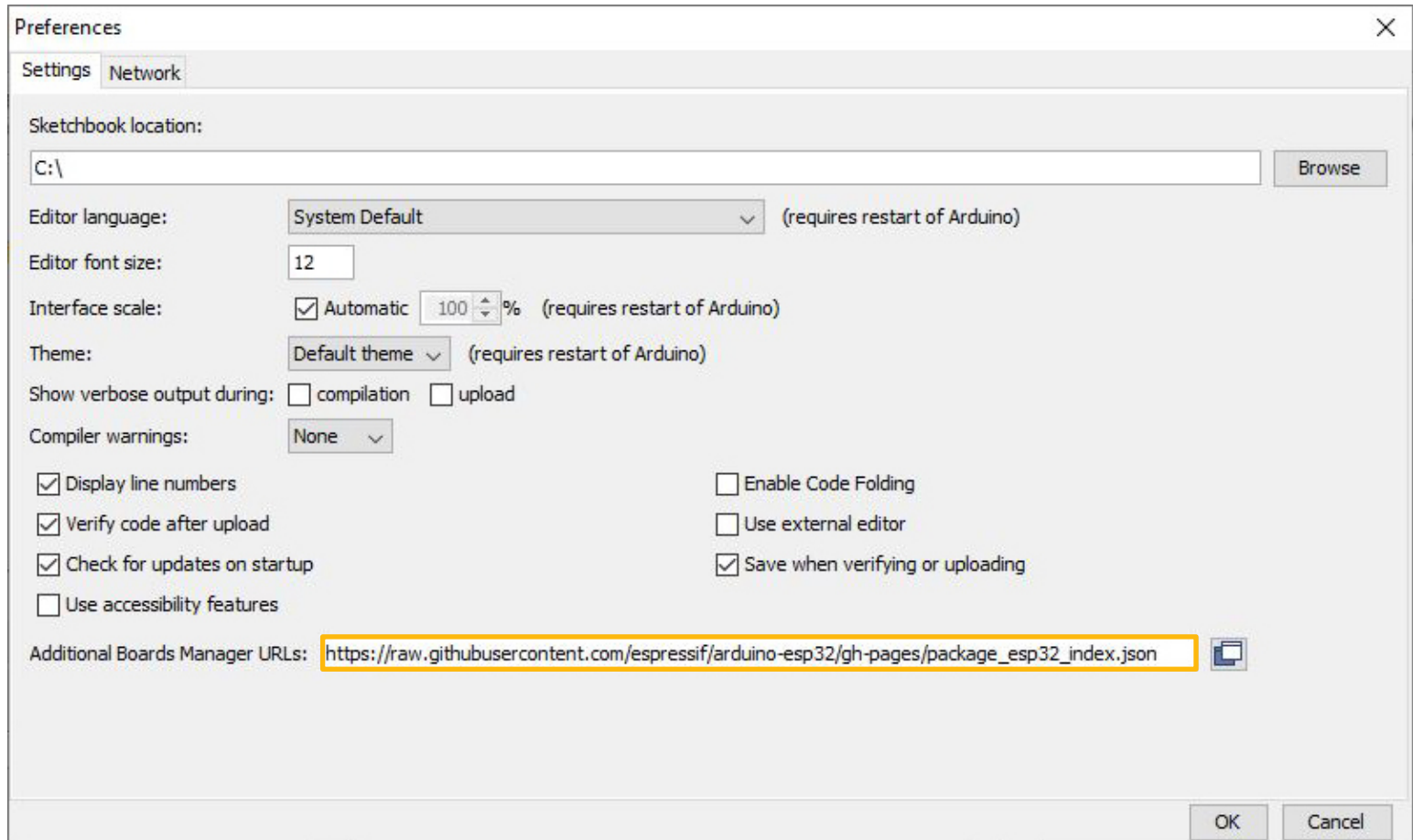
Be sure to save your **Channel ID** and **Write API Key** we are going to need these, in the next few steps.

Step 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.



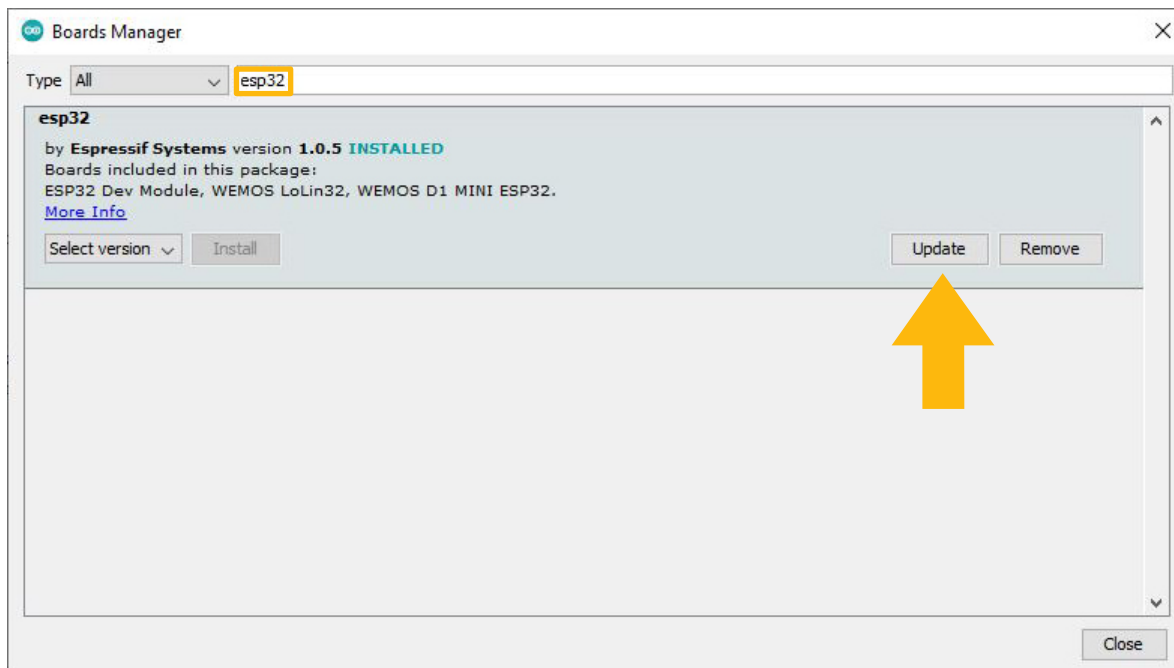
Make sure this URL is in the textbox

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)

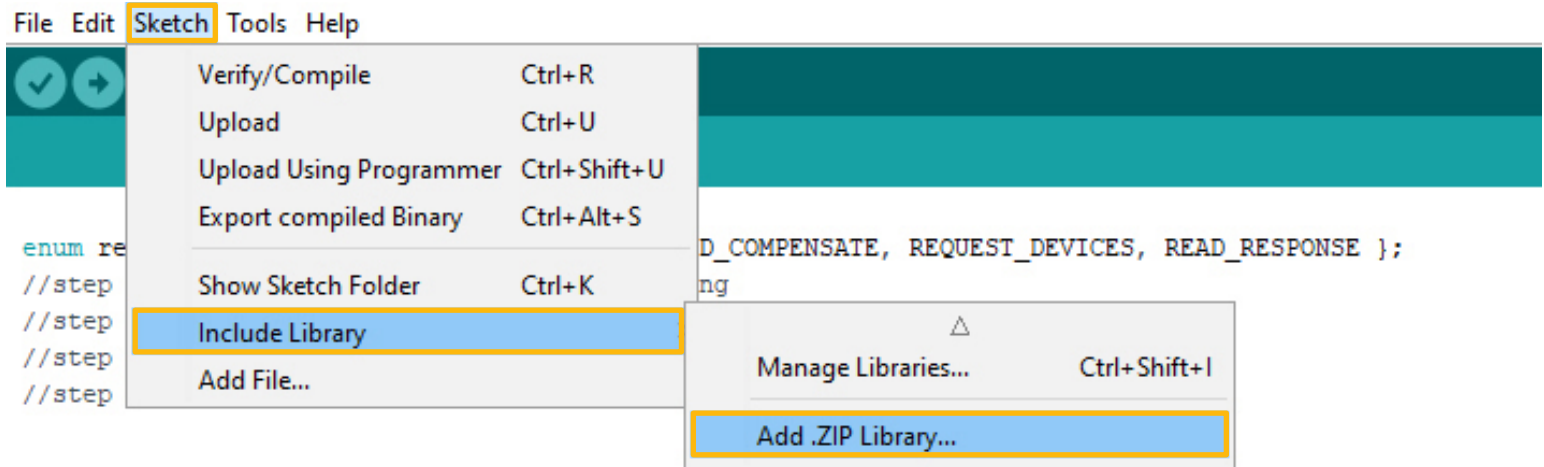
C Download the ThingSpeak library for Arduino

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

Don't unzip it!

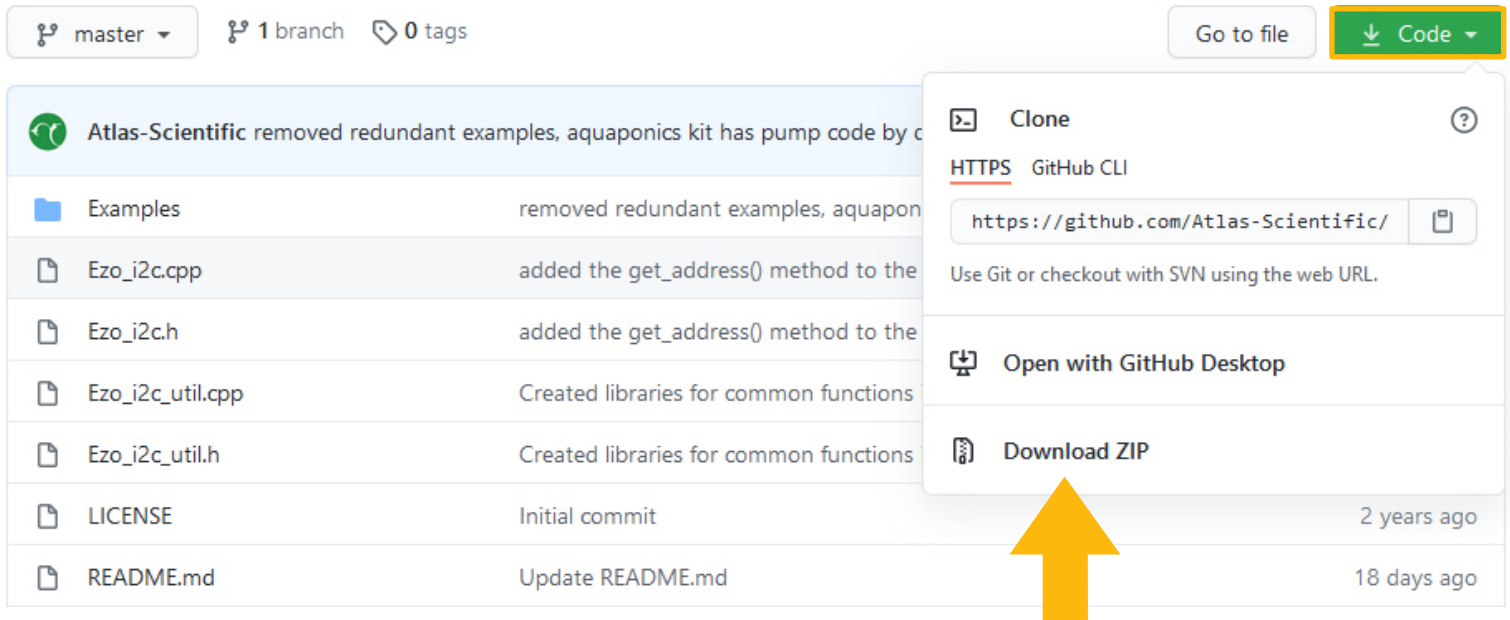
Import the .ZIP file into your Arduino IDE.

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 Pool kit 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 > pool_kit

The screenshot shows the Arduino IDE interface with the following elements:

- File Menu:** Open, Examples, Close, Save, Save As..., Page Setup, Print, Preferences, Quit.
- Examples Menu:** HTTPClient, HTTPUpdate, HTTPUpdateServer, LittleFS, NetBIOS, Preferences, SD, SD_MMC, SimpleBLE, SPI, SPIFFS, Ticker, Update, USB, WebServer, WiFi, WiFiClientSecure, WiFiProv.
- Examples from Custom Libraries:** ACROBOTIC SSD1306, Adafruit BME280 Library, Adafruit BME680 Library, Adafruit ESP8266, Adafruit HTU21DF Library, Adafruit LiquidCrystal, Adafruit SHT31 Library, Adafruit SI7021 Library, atlas_gravity, EspSoftwareSerial.
- Ezo_I2C_lib-master > Examples > I2C_lib_examples > IOT_kits > pool_kit** (highlighted path).
- Code Editor:** Contains C++ code for a Wi-Fi pool kit. Key comments include: "wifi pool kit that uses the Adafruit huzzah32 as its computer.", "//include wifi library", "//include thingspeak library", "//imports a 4 function sequencer", "//imports a 1 function sequencer", "//brings in common print statements", "I2C library from https://github.com/Atlas-Scientific/Ezo_I2c_lib", "I2C library", "//declare that this device connects to a Wi-Fi network, create a connect", "ThingSpeak Credentials-----", "//The name of the Wi-Fi network you are connecting to", "//Your WiFi network password", "//Your Thingspeak channel number", "//Your ThingSpeak Write API Key", "XXXXX";", "-----", "//create a PH circuit object, who's address is 99 and name is \"PH\"", "//create an ORP circuit object who's address is 98 and name is \"ORP\"", "//create an RTD circuit object who's address is 102 and name is \"RTD\"", "//create an PMPL circuit object who's address is 109 and name is \"PMPL\"", "of boards used for sending commands to all or specific boards".
- Status Bar:** "Library added to your library".

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)

```
pool_kit | Arduino 1.8.13
File Edit Sketch Tools Help

pool_kit
1 #include <iot_cmd.h>
2 #include <ESP8266WiFi.h> //include esp8266 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 a Wi-Fi network,
11
12 //-----Fill in your Wi-Fi / ThingSpeak Credentials-----
13 const String ssid = "Wifi Name"; //The name of the Wi-Fi network you are connecting to
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.

```
48 //parameters for setting the pump output
49 #define PUMP_BOARD PMP1 //the pump that will do the output (if theres more than one)
50 #define PUMP_DOSE 10 //the dose that the pump will dispense in milliliters
51 #define EZO_BOARD PH //the circuit that will be the target of comparison
52 #define IS_GREATER_THAN true //true means the circuit's reading has to be greater than the comparison
53 #define COMPARISON_VALUE 7 //the threshold above or below which the pump is activated
```

Step 6 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 'Tools' menu open. The path 'Tools > Board > ESP32 Arduino > Adafruit ESP32 Feather' is highlighted. The background code is a sketch for a pool kit that uses the Adafruit Huzzah32 as a target.

```
pool_kit | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
pool_kit
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
22 Ezo_board ORP = Ezo_board(98, "ORP"); //create an ORP
23 Ezo_board RTD = Ezo_board(102, "RTD"); //create an RTD
24 Ezo_board PMPL = Ezo_board(109, "PMPL"); //create an PMPL
25
26 Ezo_board device_list[] = { //an array of boards used for
27 PH,
28 ORP,
29 RTD,
30 PMPL
```

The Tools menu options are:

- Auto Format (Ctrl+T)
- Archive Sketch
- Fix Encoding & Reload
- Manage Libraries... (Ctrl+Shift+I)
- Serial Monitor (Ctrl+Shift+M)
- Serial Plotter (Ctrl+Shift+L)
- WiFi101 / WiFiNINA Firmware Updater
- Board: "Adafruit ESP32 Feather"**
- Upload Speed: "921600"
- Flash Frequency: "80MHz"
- Partition Scheme: "Default"
- Core Debug Level: "None"
- Port
- Get Board Info
- Programmer
- Burn Bootloader

The Boards Manager sub-menu is open, showing:

- Boards Manager...
- Arduino AVR Boards
- ESP32 Arduino**
- ESP8266 Boards (3.0.2)

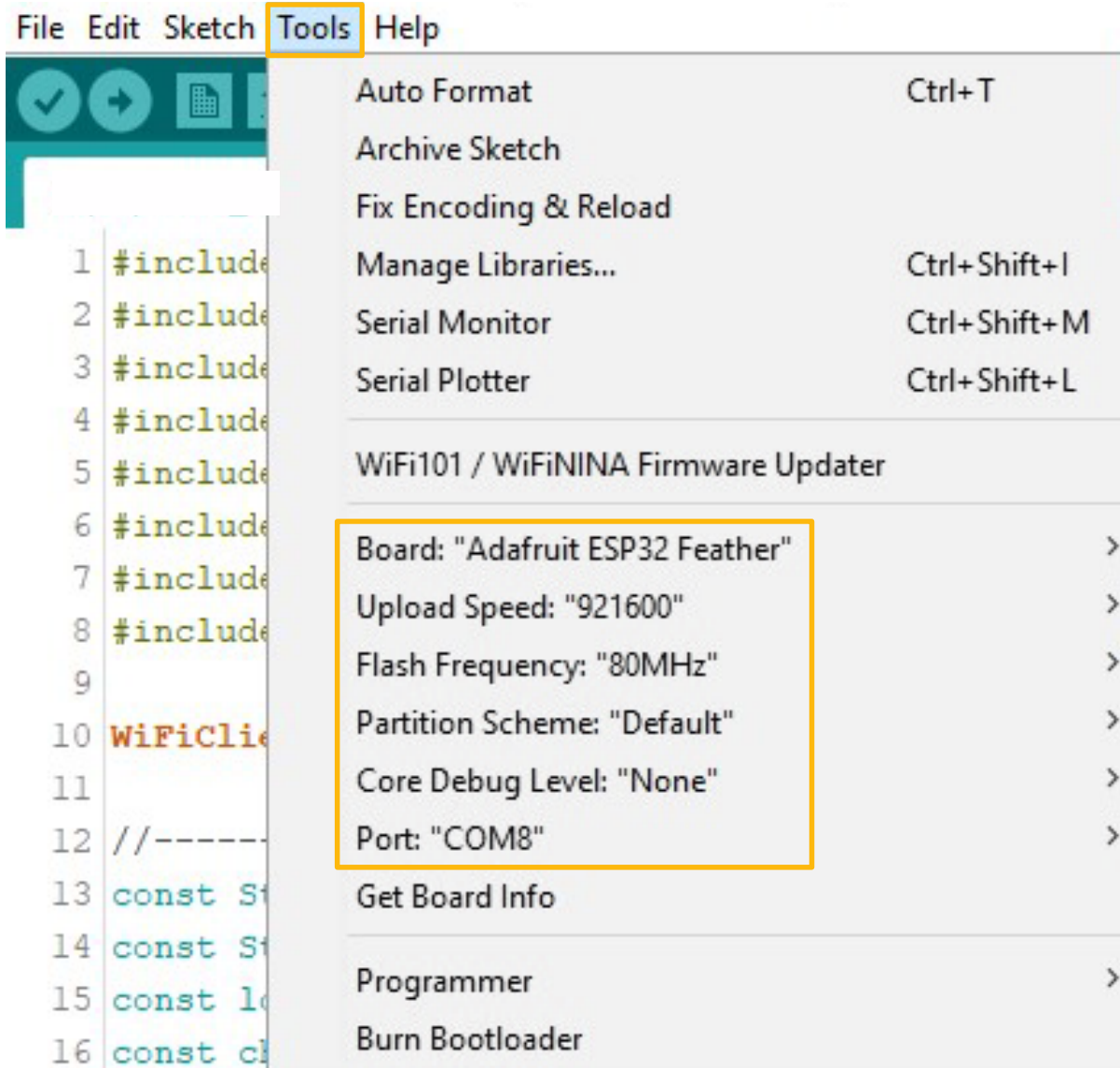
The ESP32 Arduino sub-menu is open, showing:

- FireBeetle-ESP32
- IntoRobot Fig
- Onehorse ESP32 Dev Module
- Adafruit ESP32 Feather**
- Adafruit Metro ESP32-S2
- Adafruit MagTag 2.9"
- Adafruit FunHouse
- Adafruit Feather ESP32-S2 (no PSRAM)
- NodeMCU-32S
- MH ET LIVE ESP32DevKIT
- MH ET LIVE ESP32MiniKit
- ESP32vn IoT Uno
- DOIT ESP32 DEVKIT V1
- DOIT ESPduino32
- OLIMEX ESP32-EVB
- OLIMEX ESP32-GATEWAY
- OLIMEX ESP32-PoE
- OLIMEX ESP32-PoE-ISO
- OLIMEX ESP32-DevKit-LiPo
- ThaiEasyElec's ESPino32
- M5Stack-Core-ESP32
- M5Stack-FIRE

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

pool_kit | Arduino 1.8.13

File Edit Sketch Tools Help

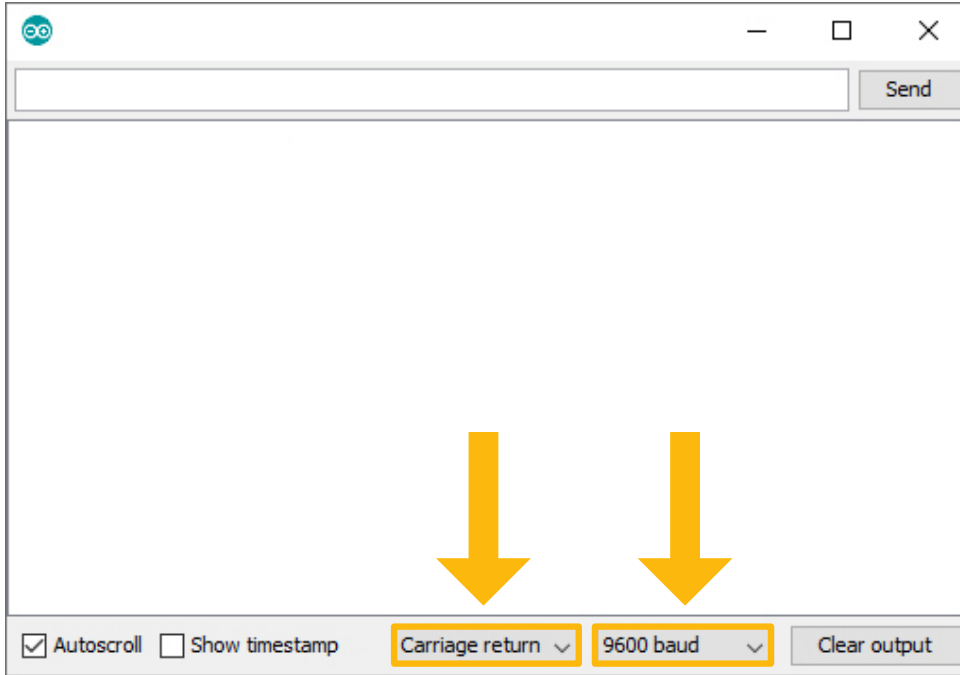


Compile and upload the code.

Step 7 See the readings

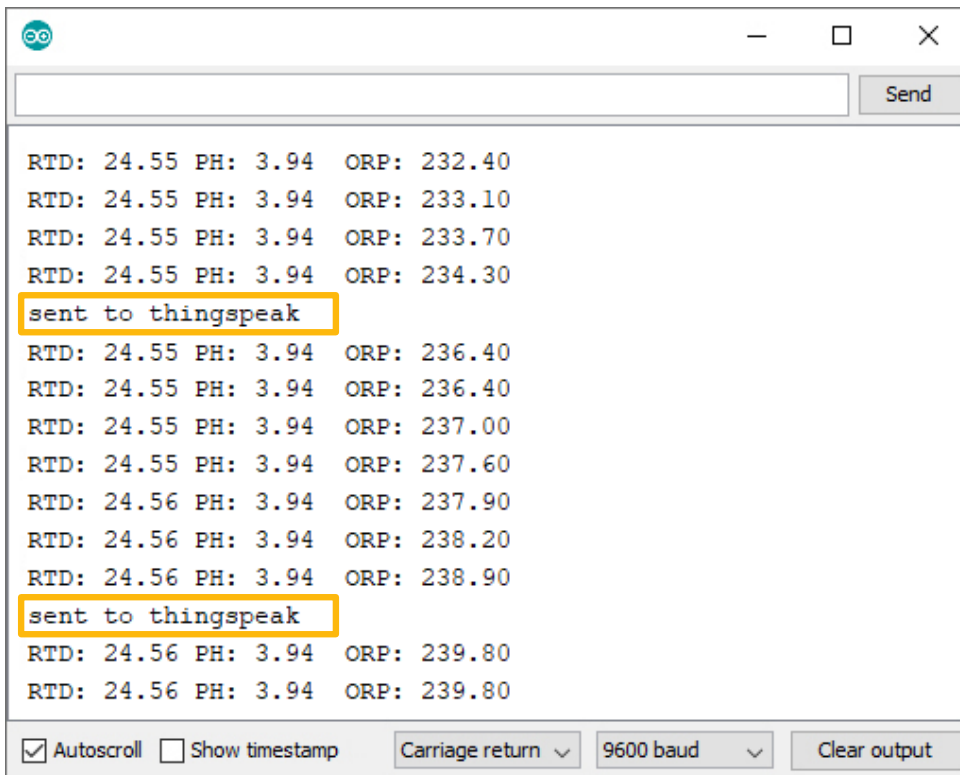
Open your Arduino serial monitor.

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

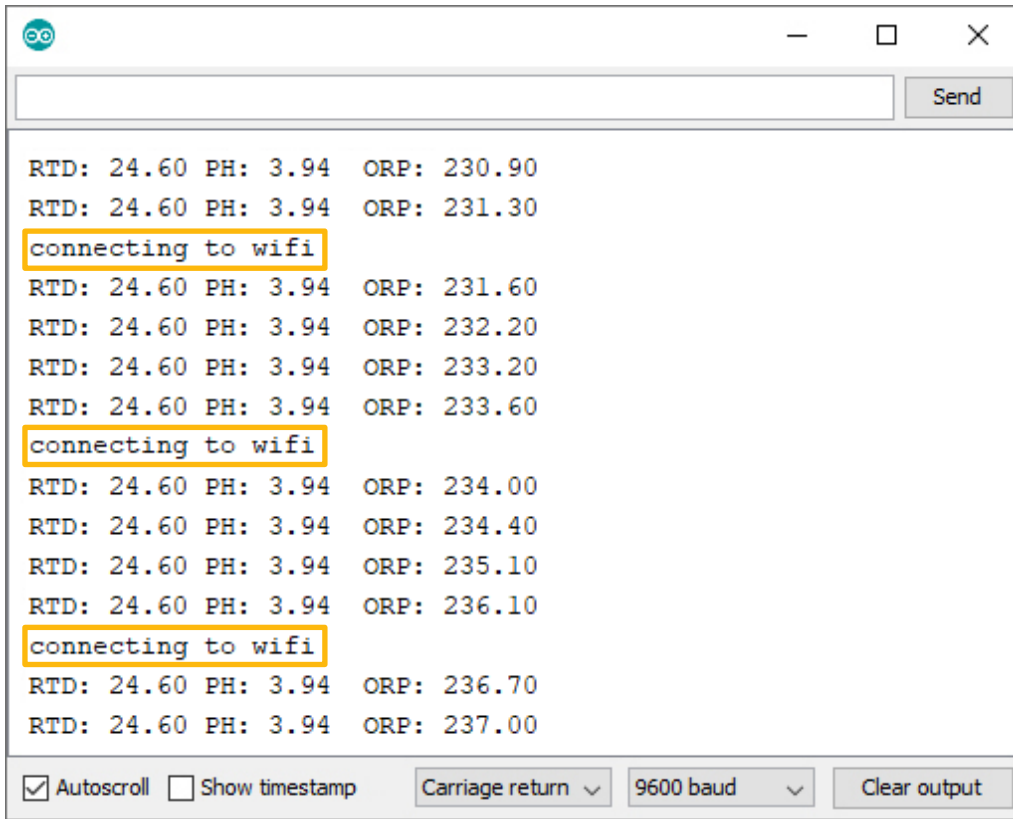


Set to **carriage return** and **9600 baud**.

The Wi-Fi Pool Kit will always attempt to connect to ThingSpeak on bootup.



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



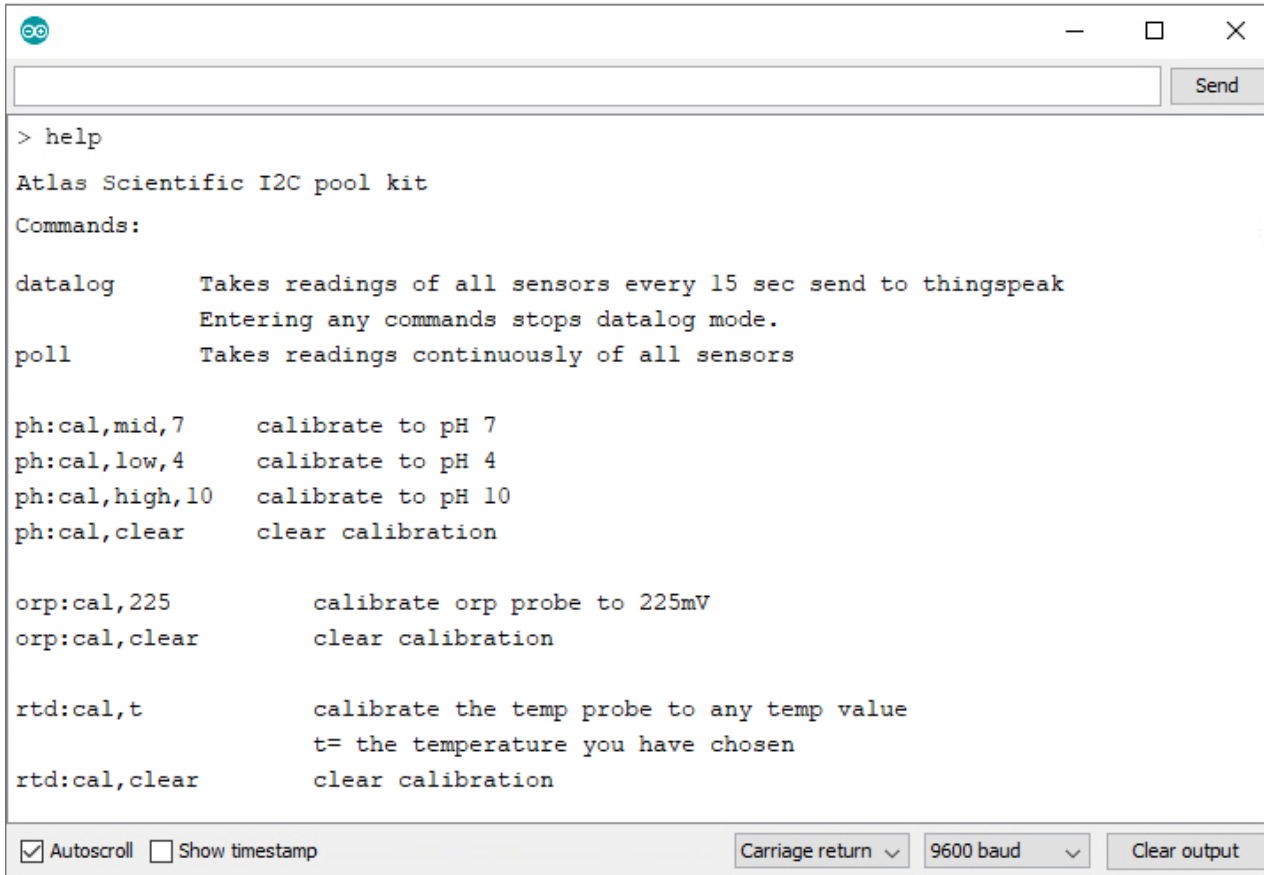
The screenshot shows a terminal window with a title bar containing a logo, a close button, and a maximize button. The terminal output consists of several lines of sensor data: RTD: 24.60 PH: 3.94 ORP: 230.90, RTD: 24.60 PH: 3.94 ORP: 231.30, RTD: 24.60 PH: 3.94 ORP: 231.60, RTD: 24.60 PH: 3.94 ORP: 232.20, RTD: 24.60 PH: 3.94 ORP: 233.20, RTD: 24.60 PH: 3.94 ORP: 233.60, RTD: 24.60 PH: 3.94 ORP: 234.00, RTD: 24.60 PH: 3.94 ORP: 234.40, RTD: 24.60 PH: 3.94 ORP: 235.10, RTD: 24.60 PH: 3.94 ORP: 236.10, RTD: 24.60 PH: 3.94 ORP: 236.70, and RTD: 24.60 PH: 3.94 ORP: 237.00. Three instances of the text 'connecting to wifi' are interspersed among the data lines and are highlighted with a yellow box. At the top of the terminal is an input field with a 'Send' button. At the bottom, there are control buttons: a checked 'Autoscroll' checkbox, an unchecked 'Show timestamp' checkbox, a 'Carriage return' dropdown menu, a '9600 baud' dropdown menu, and a 'Clear output' button.

Entering the **poll** command will stop the Wi-Fi Pool Kit 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 pool 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

orp:cal,225   calibrate orp probe to 225mV
orp:cal,clear clear calibration

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

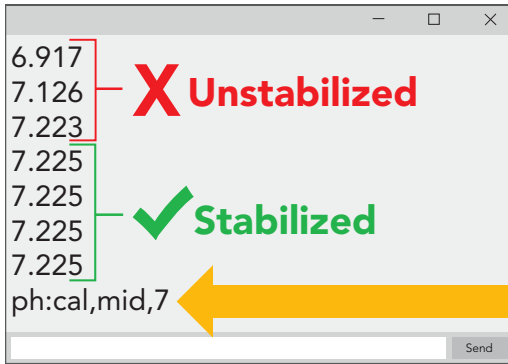
A The poll command

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**

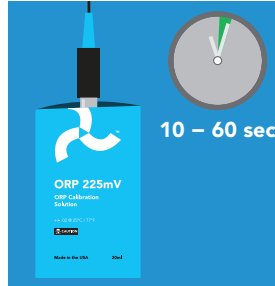
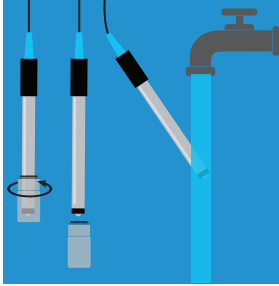
After 20 mins, the calibration solution inside an open pouch is no longer considered accurate.

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 ORP

Remove the soaker bottle and rinse off the ORP probe. Remove the top of the **ORP 225mV** calibration solution pouch. Insert the ORP probe directly into the pouch, and let the probe sit in the calibration solution until the readings stabilize (*small movement from one reading to the next is normal*).



```
342.0  
315.2  
268.7  
240.1  
240.1  
240.1  
240.1  
orp:cal,225
```

X Unstabilized

✓ Stabilized

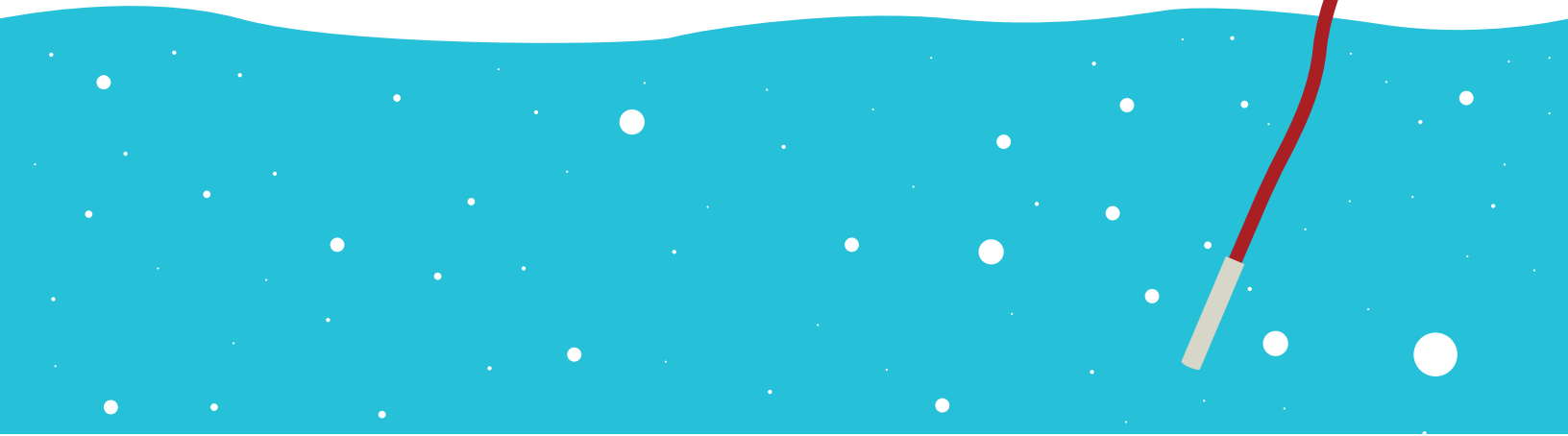
Send

Once the readings have stabilized, issue the calibration command. In this case **orp:cal,225**

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

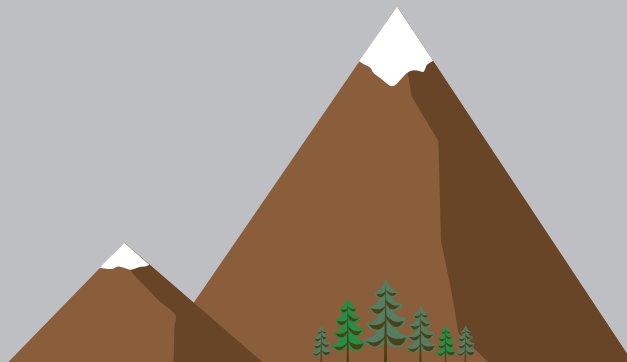


Elevation in meters

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!