OpenCTD

Construction and Operation

Andrew Thaler
Kersey Sturdivant
Russell Neches
Ian Black

OpenCTD - 1
1/6/2020

Contents

1.0 Overview

The ocean is not uniform, it its filled with swirling eddies, temperature boundaries, layers of high and low salinity, changing densities, and other physical characteristics invisible to surface observers (Sverdrup et al., 1942). By measuring salinity, temperature, and depth, scientists can unlock ocean patterns hidden beneath the sea's surface. To reveal these patterns, oceanographers use a CTD—an instrument that measures **C**onductivity (used to calculate salinity), **T**emperature, and absolute pressure (used to determine water **D**epth; Thomson and Emery, 2014).

CTDs come in a variety of shapes, sizes, and applications. Most oceanographic research vessels have a CTD connected to a rosette system, which houses other instruments and collects physical water samples in parallel with real-time data. CTDs are also commonly attached to moorings, autonomous underwater vehicles (AUVs), remote-operated vehicles (ROVs), and, on occasion, marine animals (Hooker and Boyd, 2003).

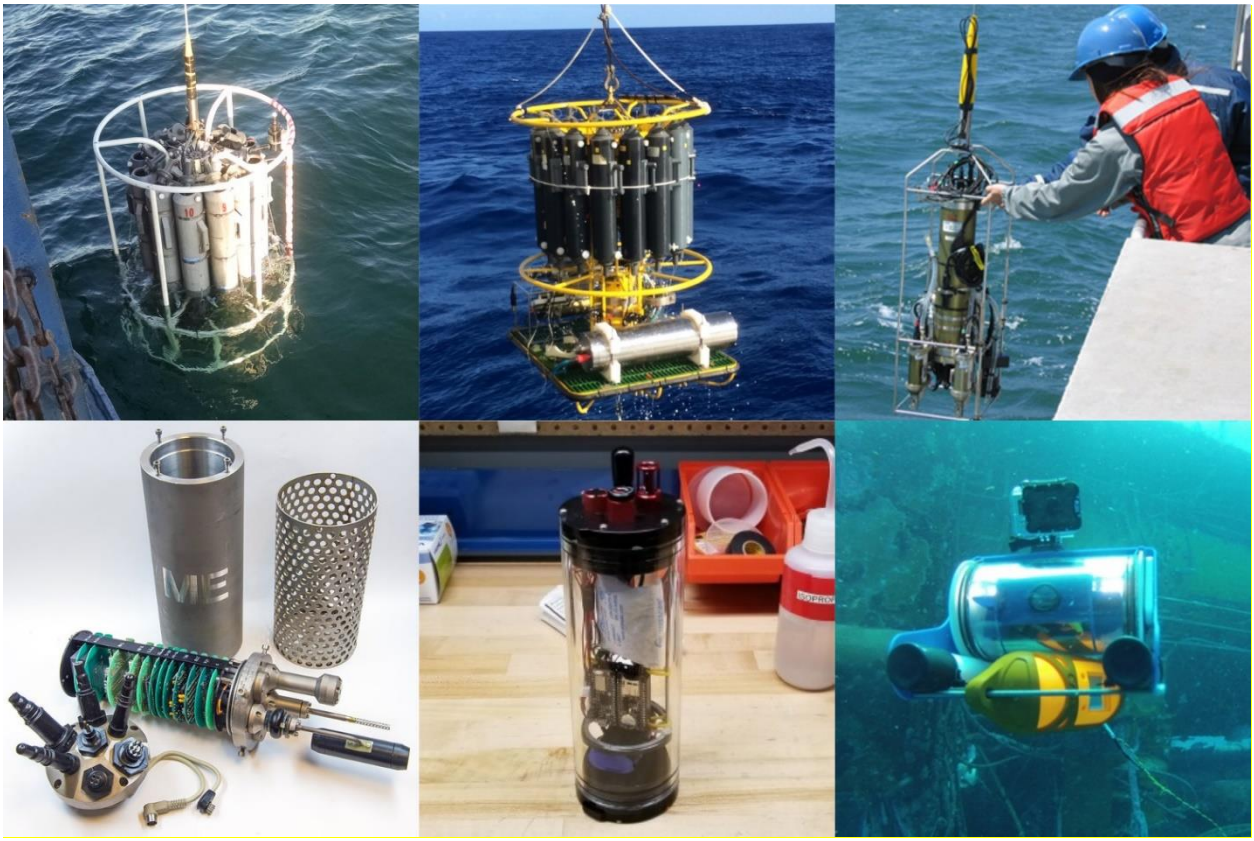


Figure 1. A selection of various CTDs used for field work. Top left: A CTD and rosette deployed from a NOAA boat (WHOI); top center: A CTD and rosette deployed by IFREMER; top right: a student deploys a CTD (MIT); bottom left: an 80s era ME GRISARD CTD; bottom center: the CTDizzle open-source CTD by Ian Black; bottom right: a YSI Castaway affixed to an OpenROV 2.3 (Sofar).

Commercial CTDs are expensive, with the most affordable models costing as much as $6,000.00. For near-shore oceanographic research on the continental shelf, where the water depth rarely exceeds 200 meters, this can be a prohibitive cost. The expense of the CTD is a barrier-to-

entry for formal and informal researchers working with limited budgets, including scientists from developing nations, citizen oceanographers, environmental educators, and students of all levels interested in understanding their local coasts and waterways.

Climate change, ocean acidification, and sea level rise have created an urgent need for extensive measurements of oceanographic conditions both globally and locally (Stammer et al., 2016; Stephen et al., 2016). At the same time, government funding at both state and federal levels for climate change research can be unreliable, if not wholly non-existent. This creates a pressing need for low-cost alternatives to common scientific instruments that allow formal researchers to stretch the impact of extant funding awards while simultaneously enabling informal researchers (such as environmental monitoring organizations and non-governmental organizations, as well as concerned community members) to contribute water-quality measurements.

The OpenCTD is a low-cost, open-source CTD designed intentionally for citizen scientists, educators, and researchers working in nearshore coastal ecosystems, where entire research projects can be conducted for less than the cost of a commercial CTD. It was developed by a core team of marine ecologists in collaboration with a distributed community of scientists, engineers, makers, and conservation practitioners. It is assembled from components commonly available at large hardware stores or through major online retailers. An Arduino-based microcontroller controls a battery of sensors sealed within a PVC pipe. Power is provided by a standard 3.7V lithium polymer battery and data is stored as a tab-delimited text file accessed via SD card. All OpenCTD software is released open source with no restrictions on use.

The OpenCTD is designed to be built by the end-user, providing both access to the tools of oceanography as well as the skills to maintain, repair, and replace OpenCTDs. For educators looking for novel, in-depth, hands-on STEM experience for advanced students, the process of building an OpenCTD can offer a great introduction to coding, 3D-printing, hardware prototyping, and electronics. Construction of an OpenCTD can provide a practical foundation for courses in oceanography and marine or environmental science.

As you build your OpenCTD please reach out to the team at Oceanography for Everyone if you have any questions or feedback about the process. We love to keep up with the community of people constructing and using these devices. During assembly and while using an OpenCTD or other Oceanography for Everyone tools, please consider Tweeting or Instagramming using the hashtag #HackTheOcean so that the community can follow your efforts.

1.1 Skills Needed

To construct an OpenCTD, you will need to be familiar with basic soldering and coding, but this project can also serve as an introduction to those skills. With a few hours of practice, you can develop the soldering, electronics, and coding skills necessary to build an OpenCTD.

For absolute beginners, we recommend working through **Soldering is Easy: Here's How To Do It** by Mitch Altman, Andie Nordgren, and Jeff Keyzer, which is an accessible and approachable introduction to soldering fundamentals:

- **Soldering is Easy: Here's How To Do It: mightyohm.com/files/soldercomic/FullSolderComic_EN.pdf**

The MightyOhm Geiger Counter Kit, the Mega Menorah 9000 Soldering Kit, and the Gikfun MDS-60 Metal Detector Module for Arduino DIY Soldering Practice Kit are great introductory kits to teach soldering and basic electronic principles to students from ages 7 to 70. Preparing the components needed for an OpenCTD will provide most students with sufficient grounding to tackle the more complex elements of constructing an OpenCTD.

- **Geiger Counter Kit - Radiation Sensor: www.adafruit.com/product/483**
- **Evil Mad Scientist Labs Mega Menorah 9000: www.adafruit.com/product/2291**
- **Gikfun Metal Detector Soldering Practice Kit: https://amzn.to/2NlBdX8**

Arduino is a coding environment build on C/C++ and designed for flexibility and ease of use, but it can still seem daunting to first-time users. Practically, the OpenCTD source code is complete and most users will only need to make small alterations to 3 lines of code during assembly and testing, but it is extremely beneficial to understand what the software is doing. Calibration and troubleshooting will require users to load different programs onto the completed OpenCTD. Basic familiarity with the Arduino Integrated Development Environment (IDE) is essential. **Environmental Monitoring with Arduino: Building Simple Devices to Collect Data About the World Around Us** is a short, easy to understand, though dated, introduction to the Arduino ecosystem, and will provide you with many ideas for expanding your inventory of open-source environmental sensors.

- **Environmental Monitoring with Arduino: https://amzn.to/32myCR3**

You will also want to familiarize yourself with resistors and how to read the color code used to demarcate the resistance of a resistor. Arrow provides a good overview of the concept:

- **Cracking the Resistor Color Code: https://www.arrow.com/en/research-and-events/articles/resistor-color-code**

1.2 Project Status and Compatibility

OpenCTD is stable (as of 4/22/2019) on the ATSAMD21G18 ARM Cortex M0 processor. Construction workflow, as well as pinouts, bill of materials, and operation are designed around the Adafruit Adalogger M0. Adjustments may need to be made to adapt to other M0-based microcontrollers.

An older version of OpenCTD is stable on the ATMEGA32U4 processor common to many Arduino systems. Construction workflow, as well as pinouts, bill of materials, and operation are designed around the Qduino Mini. Adjustments may need to be made to adapt to other ATMEGA32U4-based microcontrollers.

1.3 Field Trials

Field trials are critical to assessing the precision, accuracy, durability, and functionality of the OpenCTD. Data from field trials are archived online in the OpenCTD GitHub repository.

Figure 2. The RV Blue Heron *on Lake Superior.*

Lake Superior, *R/V Blue Heron*: Initial field trials were completed in October of 2015 and analysis of the data from the first several casts confirmed the accuracy of OpenCTD compared against a commercial CTD. OpenCTD was successfully deployed to a depth of 140 meters. This initial field trial, supported by a UNOLS grant to Andrew Thaler, tested the earliest version of the OpenCTD, which still used mineral oil for pressure compensation and was conducted in a freshwater lake, so did not test the conductivity probe. All data and field notes from the first Field Trial can be found in the Blue Heron 15-11 repository.

Gloucester Point, VA, Hurricane Hermine: Hurricane Hermine is a tropical cyclone that passed over our field site in coastal Virginia on September 4, 2016. To test the longevity of the batteries and the durability of the OpenCTD during long term deployments, Andrew Thaler placed two units in our Test Ditch (a tidally influence drainage ditch connected to the Mobjack Bay estuary), where they remained for 74 hours. This field trial tested the first generation

OpenCTD based on the Qduino Mini Arduino platform. Notes and data files can be found in the HermineDeployment_9-2016 repository.

Ongoing tests both in the field and laboratory occur on a regular basis throughout the OpenCTD development process and are used to continuously refine and validate new instruments as they are produced.

1.4. Acknowledgements.

From its inception, the OpenCTD and Oceanography for Everyone has benefitted from the support and expertise of an incredible community. Funding to support the OpenCTD was provided by the Bureau of Ocean Energy Management as well as through an initial crowdfunding campaign and an ongoing subscription patronage program through Patreon. Dr. Kim Martini provided exceptional guidance throughout the development process.

We would also like to thank David Lang, Eric Stackpole, Walt Holm, Zack Johnson, and Brian Grau of OpenROV, as well as Dr. Miriam Goldstein, Dr. Amy Freitag, Dr. Craig McClain, Murt Conover, Jake Levenson, Dr. Alex Deghan, Dr. Barbara Martinez, Andrew Quitmeyer, and the participants of GOSH Santiago for their support, advice, and guidance, and one anonymous GitHub contributor who provided significant assistance with early code development.

2.0 Materials and Housing Preparation

The OpenCTD is designed such that it can be assembled entirely from components available from large online electronics vendors such as Adafruit, Sparkfun, Amazon, or AliExpress. Many components can also be purchased directly through their supplier or sourced through auction sites at lower cost. Quality across suppliers for most components has been generally consistent.

Some parts, such as resistors and capacitors, are very cheap but hard to source individually. “Introduction to electronics"-type kits, which include many standard resistors and capacitors in small lots tend to be the most economical choice.

Raw, calibrated absolute pressure sensors are available in both 14-Bar and 30-Bar modules from several vendors. These are surface mounted sensors that require significant soldering skills to mount properly. Beginners may prefer to use the simpler, but more expensive, pressure sensor breakout board. Detailed discussion of the differences and benefits between the two pressure sensor solutions is discussed in Chapter 3.

While we provide a list of recommended parts and their suppliers, these are not necessarily the least expensive option and price can vary significantly depending on local availability and number of units being built. Part availability varies from month to month, and some components may become permanently unavailable or listed under a new manufacturer and product name. We do our best to keep the bill of materials up to date. If you find any discrepancies, please contact us.

When available, we provide Amazon Affiliate links to purchase components. These provide us a small kickback when you buy a component through those links, which helps offset the cost of development and testing for the OpenCTD. We are, however, not endorsing Amazon and encourage users to seek out local sources when possible.

The foundation of the platform is a PVC pipe and a suite of 3D-printed components which provide structure for the rest of the system. It requires about 6 hours to 3D print all parts on a standard consumer 3D printer.

2.1 Bill of Materials

Gray highlights indicate multiple options for a single part. Orange headings indicate multiple solutions for the pressure sensor module.

Item	Links	Price
Adafruit Feather M0 Adalogger	https://www.adafruit.com/product/2796 https://amzn.to/36DrCm4	$19.95
Adafruit Stacking Header	https://www.adafruit.com/product/2830	$1.25
Short Feather Male Headers	https://www.adafruit.com/product/3002	$0.50
Adafruit DS3231 Precision RTC FeatherWing	https://www.adafruit.com/product/3028 https://amzn.to/2PRHmvS	$13.95
CR1220 3V Coin Cell Battery	https://amzn.to/2NYFQpf https://www.adafruit.com/product/380	$6.99
SanDisk MicroSD Memory Card	https://amzn.to/2FTkfKW	$4.99
Half-size breadboard	https://www.adafruit.com/product/64 https://amzn.to/2CmjyIB	$4.99
Screw Terminal Block Extra Long Pins for Breadboarding (3)	https://amzn.to/34nqB0x	$4.49
Adafruit Perma-Proto Small Mint Tin Size Breadboard PCB	https://www.adafruit.com/product/1214	$7.95
Right Angle Female PCB Header	https://amzn.to/2ICUOki	$9.18
10 kΩ Resistor (2)	https://amzn.to/34MSGhd	$6.78
4.7 kΩ Resistor (1)	https://amzn.to/33toes0	$6.78
Expandable braided sleeving	https://amzn.to/2X57FjM	$7.99
Lithium Ion Battery Pack 3.7V 4400 mAh	https://www.adafruit.com/product/354	$19.95

	https://amzn.to/2IfxuIF	
Lithium Ion Battery Pack 3.7V 2000 mAh	https://www.adafruit.com/product/2011 https://amzn.to/2UOLmkq	$12.50
2” PVC	https://amzn.to/2JZnNAm	$5.22
Oatey 270229 End of Pipe Gripper Mechanical Plug	https://amzn.to/2WLCcS7	$5.91
10mm X 3mm Round Rare Earth Magnet (2)	https://amzn.to/2XGbUS3	$9.90
10mm round double-sided adhesive mounting dots (2)	https://amzn.to/2OdU85B	$10.53
DS18B20 Temperature Sensor in Stainless Steel Housing	https://amzn.to/2YOevug	$8.98
Atlas Scientific Conductivity Circuit	https://www.atlas-scientific.com/product_pages/circuits/ezo_ec.html https://amzn.to/2K0Xk5z	$60.00
Atlas Scientific K 1.0 Conductivity Probe	https://www.atlas-scientific.com/product_pages/probes/ec_k1-0.html https://amzn.to/2K0Xk5z	$139.00
Atlas Scientific K 1.0 Mini Conductivity Probe	https://www.atlas-scientific.com/product_pages/probes/ec_k1-0-mini.html	$110.00
Pressure Sensor Option 1		
MS5803-14BA Pressure Sensor Breakout	https://www.sparkfun.com/products/12909 https://amzn.to/2rfyEgA	$59.95
Pressure Sensor Option 2		
MS5803-14BA Pressure Sensor SMC	https://www.newark.com/sensor-solutions-te-connectivity/ms580314ba01-00/pressure-sensor-14bar-digital/dp/03AC1591	$21.82
SOIC to DIP Adapter	https://www.adafruit.com/product/1212	$2.95
10 kΩ Resistor	https://amzn.to/34MSGhd	$6.78
0.1 uF Capacitor	https://amzn.to/32tMBol	$6.14

2.2 Tools and Consumables

Tools for general electronics work: flush cutters, wire stripper, soldering iron, solder tip cleaner, helping hands, needle-nosed pliers, utility knife, solderless breadboard, heat gun.
Consumables for general electronics work: lead-free solder, electrical tape, 22-gauge solid core wire, 20-gauge copper hook-up wire, heat shrink tubing (various small sizes).

Tools for 3D printing: 3D printer, deburring tool.
Consumables for 3D printing: PLA Filament.

Tools for housing and adhesion: Hand saw or PVC pipe cutter, drill with 1" bit or router and jig, 3M epoxy gun for 50ml cartridges (https://amzn.to/2oU32MJ), hot glue gun.
Consumables for housing and adhesion: LOCTITE 2-part Adhesive Epoxy (Hysol EA-90FL), venturi mixing nozzles for 50 mL epoxy cartridges, LOCTITE Epoxy Five Minute Instant Mix, hot glue, LOCTITE Gel Control super glue, sandpaper.

Tools for Arduino programming: Computer capable of running Arduino IDE, micro-USB cable, coffee maker.
Consumables for Arduino programming: Calibration standard for conductivity, coffee.

A note on Epoxies and Glues: We have tested several different epoxies and glues to ensure a generally wide variety of options for sourcing. All superglues should be gel-based with a high viscosity to allow for more control over placement and flow. We find that Loctite Ultra Gel Control Super Glue is ideal for constructing the OpenCTD (https://amzn.to/3106gv5). Hysol EA-90FL has been extensively tested in OpenCTDs as well as other marine hardware, however it is not always easy to source. Other two-part epoxies can also be used for building the sensor pass-though, provided they have a minimum sheer strength of 900 PSI and a medium or high viscosity rating. Any 5-minute marine epoxy is acceptable for sealing the pressure sensor.

Figure 3. Most of the tools needed to construct an OpenCTD.

2.3 3D-Printing

The main support structure as well as the sensor baseplate, magnetic switch and connector reinforcement are 3D-printable objects. While it is possible to construct a fully functional OpenCTD without using these parts, they are designed to make building and operating the device simpler and more intuitive. The primary parts are:

- **Electronics Chassis** which holds the Adalogger and other electronic components, allowing users to easily access the SD card while minimizing strain on the wiring.
- **Sensor Baseplate** which provides a foundation and template for arranging the sensors prior to potting them in epoxy.
- **3-part Magnetic Switch**, which allows users to turn the OpenCTD on and off without opening the housing.
- **Sensor Hub Connector**, which helps relieve strain on the sensor connection when plugging and unplugging the sensors from the electronics.

All parts are designed to be printed in PLA at 100-micron resolution with 10 to 20% infill. Only the electronics chassis and magnetic switch (internal) requires support structures. A second sensor baseplate is also provided for use with the pressure sensor breakout board. If you use the Sparkfun Breakout Board for the pressure sensor, print the OpenCTD_Baseplate_Sparkfun_M0.stl file.

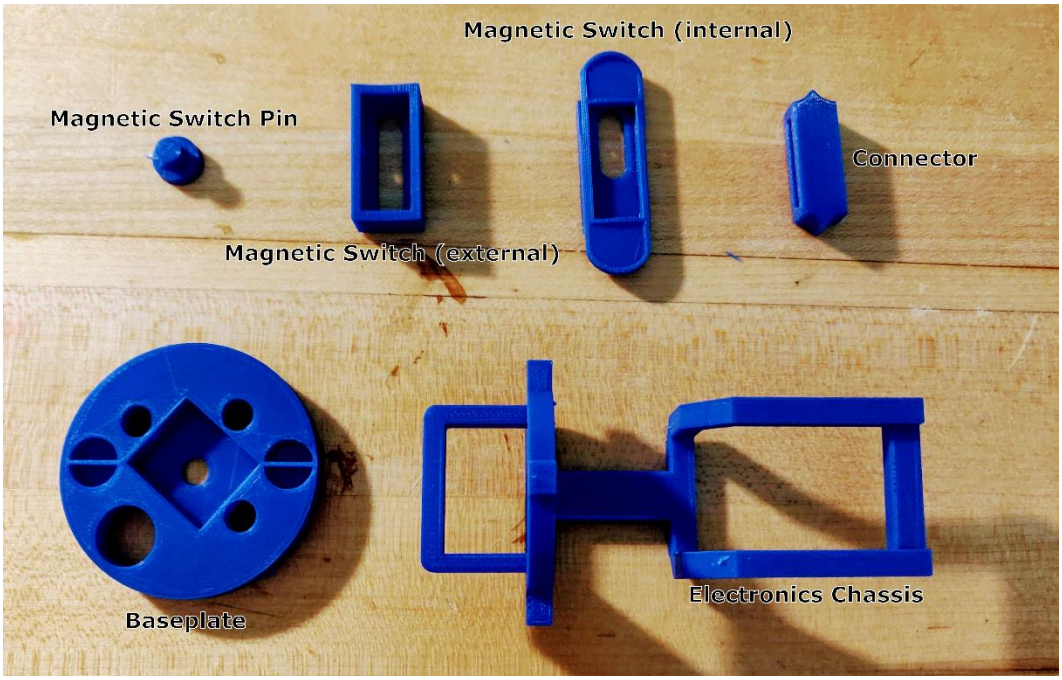


Figure 4: 3D-Printed parts for OpenCTD.

OpenCTD 3D Printed Structural Components:

- **Baseplate:** OpenCTD_Baseplate_M0.stl
  - **[ALT]** OpenCTD_Baseplate_Sparkfun_M0.stl
- **Electronics Chassis:** OpenCTD_ElectronicsChassis_M0.stl
- **Connector:** OpenCTD_Connector_M0.stl
- **Magnetic Switch (external):** OpenCTD_MagneticSwitch_External_M0.stl
- **Magnetic Switch (internal):** OpenCTD_MagneticSwitch_Internal_M0.stl
- **Magnetic Switch Pin:** OpenCTD_MagneticSwitch_Pin_M0.stl

Once the sensor baseplate is complete, use a deburring tool to round of all the edges on both top and bottom as well as inside each hole. Sensors, especially the pressure sensor, should fit snuggly inside their respective hole and the entire baseplate should fit tightly into a 2” PVC pipe.

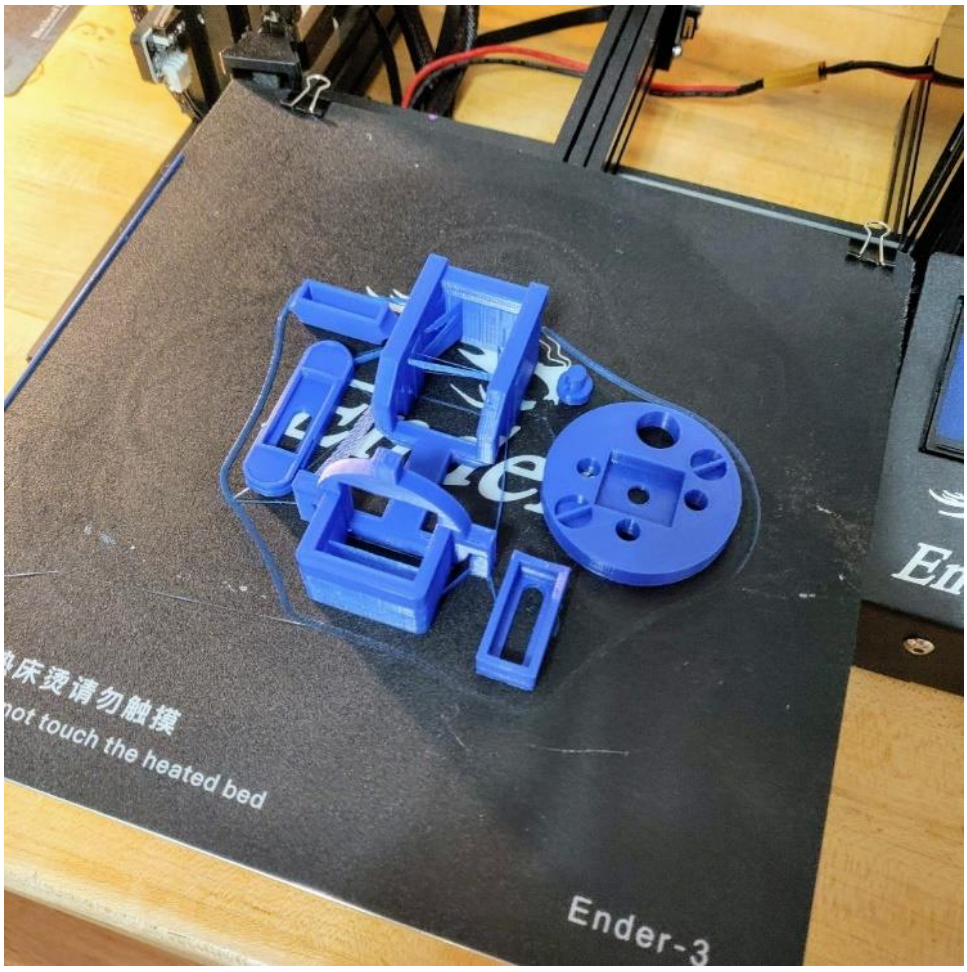


Figure 5. All components can be 3D-printed in a single batch, with support for the chassis and magnetic switch (internal) only.

Additional parts are provided to simplify assembly, including a marking jig for lining up the bottom slot in the PVC and two caps to cover the sensors and top opening when painting the CTD. These auxiliary components are not necessary to complete the build but will simplify some steps.

OpenCTD 3D Printed Auxiliary Components:

- **Jig for aligning notches in PVC pipe:** 2-inch_PVC_NotchJig.stl
- **Cap for sealing the top during painting:** 2-inch_PVC_PaintCap_Top.stl
- **Cap for sealing the bottom and protecting sensors during painting and transport:** 2-inch_PVC_PaintCap_Bottom.stl

2.4 Preparing the housing

Skills: Hand tools
Consumables: N/A
Parts: PVC Pipe, 3D-printed spacing jig (optional)

Parts 1: 12" PVC Pipe, 3D-printed Spacing Jig

To create the housing, cut a short length of schedule-40 2” PVC pipe and add notches to one end of it. This will allow water to flow over the sensors while protecting them from impacts and entanglement.

1. Cut a 12” section of PVC pipe and deburr the edges.

2. Use the 3D-printed jig to mark four spots where the notches will go.

3. If you have access to a router table, cut 4 1” deep notches into one end of the PVC using a ½” bit. If you have access to a drill and bit, cut 3 1” holes centered ¾” from the bottom.

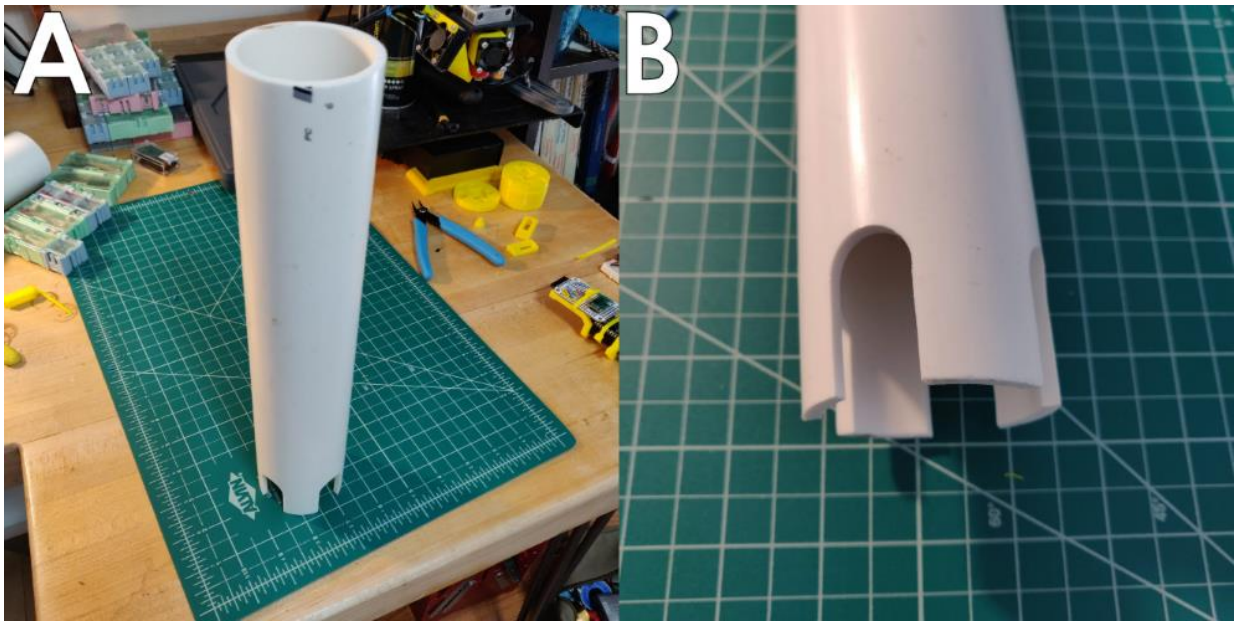


Figure 6. 12" PVC pipe (A) with notches cut at the base (B).

3.0 Arduino Preparation and Software

Before beginning the build familiarize yourself with the electrical components as well as the Arduino Integrated Development Environment (IDE). By ensuring that the software is running and the base components have been assembled and tested, you can save yourself significant frustration down the road. Setting up Arduino from scratch can be a challenging and unintuitive process the first time you do it, so it is a good idea to invest an hour or two into making sure you have compatible, up-to-date libraries and that the OpenCTD code compiles on your computer.

Arduino provides an online development environment if you cannot install software on your personal computer: https://create.arduino.cc/.

3.1 Preparing the Adalogger and RTC

Skills: Soldering
Consumables: Solder
Parts: Adafruit Feather M0 Adalogger, Adafruit DS3231 Precision RTC FeatherWing, Standard Headers, Stacking Headers, CR1220 Coin-cell Battery

Parts 2: Adalogger, Precision RTC FeatherWing, Standard Headers, Stacking Headers, Coin-cell Battery

Both the Adalogger and RTC will need headers. Headers are the male or female pin assemblies that allow you to connect and disconnect electronic components while guaranteeing solid electrical contact. Assembling the Adalogger and Real-Time Clock (RTC) is a great, low-risk, opportunity to practice your soldering skills. The Adalogger will receive stacking headers—headers that contain both male and female ends so that board can be connected to a breadboard while a second device (in this case the RTC) can be mounted on top. The RTC will receive male headers facing down so that they can be connected to the Adalogger.

In electronics, a “pin” refers to any electrical contact point and will generally be used in this guide to refer to the holes in circuit boards through which wires, headers, and other components will be soldered.

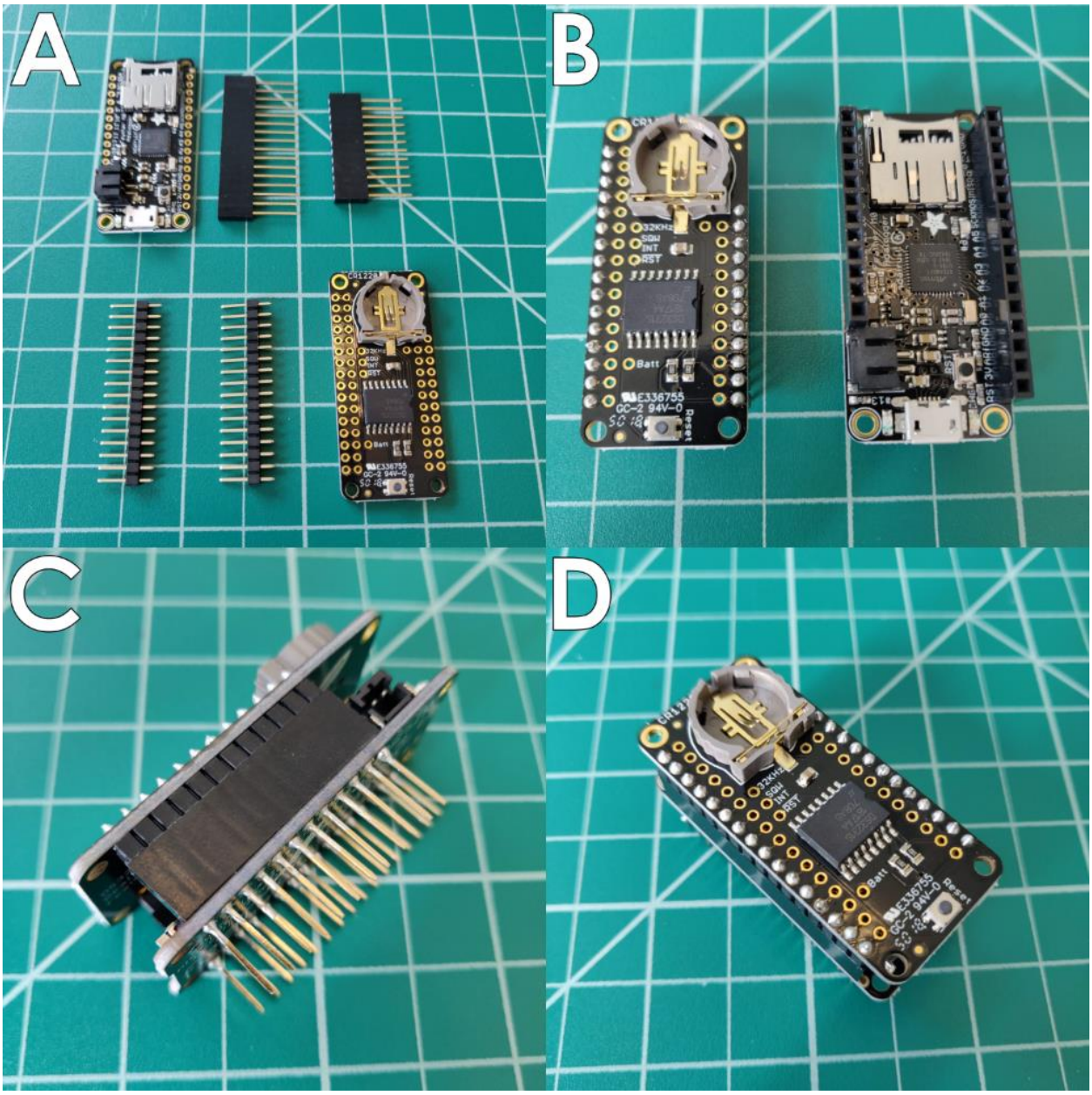


Figure 7. Adalogger and Real-time Clock with associated headers (A); after soldering (B); connected in profile (C); connected from above (D).

1. Slot the stacking headers into the through-holes on the Adalogger, ensuring that the number of headers matches the number of available pins. The female end of the stacking headers should face up. If you are using generic stacking headers, cut them to length with flush cutters so that the number of posts matches the number of pins on each side.

2. Flip the Adalogger over so that male headers are facing up. Ensure that the headers are square and perpendicular to the face of the Adalogger.

3. Solder the four corner pins, double check that the headers are straight, and then solder the remaining pins.

4. Slot the male headers into the outer pins of the RTC such that the short ends pass through the board and the long header pins face down. If you are using generic stacking headers, cut them to length with flush cutters so that the number of posts matches the number of pins on each side.

5. Solder the four corner pins, double check that the male headers are straight and aligned with female headers on the Adalogger, and then solder the remaining pins.

6. Stack the RTC on top of the Adalogger by lining up the male, downward-facing header pins on the RTC with the female, upward facing header pins on the Adalogger.

7. Insert the CR1220 battery into the RTC.

3.2 Software and Libraries

Skills: Software Management
Parts: Your Computer, Adalogger, SD Card, micro-USB cable
Consumables: Coffee

To prepare your computer to talk to the Feather M0 Adalogger, download and install Arduino IDE (https://www.arduino.cc/en/Main/Software). You will need to install additional boards so that Arduino IDE will be able to recognize the Adalogger. For a detailed walkthrough of this process, with screenshots included, please visit: https://learn.adafruit.com/adafruit-feather-m0-adalogger/setup

We use GitHub to host various projects. GitHub can seem a bit daunting to people who aren't familiar with it. At its core, GitHub is a tool to organize and coordinate files and maintain version control (multiple people can edit the same file, while keeping everything in sync). Projects are organized into repositories ('repos') that contain the source code, supporting documents, data files, and other materials needed for each project. When you visit a repo, check the README.md, this will generally describe the project (including the current status), contain necessary guides, and let you know how to contribute.

1. Launch Arduino IDE.

2. Under *preferences*, in the *File* menu, add the following URL to the "Additional Boards Manager URLs" field: https://adafruit.github.io/arduino-board-index/package_adafruit_index.json

3. Click "Ok" and then navigate to *Tools* > *Board* > *Boards Manager*.

4. In *Boards Manager*, search for "Adafruit SAMD Boards" and install the library. Now search for "Arduino SAMD Boards" and install this library as well.

5. Restart Arduino IDE.

6. When Arduino IDE restarts, you should see several new Adafruit options in the *Board* menu. The Adalogger is an Adafruit Feather M0 with an embedded SD card reader. Select "Adafruit Feather M0" from the menu.

7. Create an empty text file call **datalog.txt** on the SD card and insert it into the Adalogger's SD card reader.

8. Connect your Adalogger to your laptop using a micro-USB cable. The necessary drivers should automatically install.

3.3 Required Arduino Libraries

The following Arduino libraries are required for the OpenCTD and come packaged with Arduino IDE by default:

Wire. For communicating with I2C devices like the pressure sensor.
SPI. Serial Peripheral Interface for communicating with a standard SD card reader.
SD. For reading and writing to a standard SD card reader.

You will need to download and install the remaining libraries manually. Please refer to https://www.arduino.cc/en/Guide/Libraries for detailed instructions for installing libraries.

MS5803_14. For the MS5803 14-Bar pressure sensor. Download this library here: https://github.com/millerlp/MS5803_14 (Note: if you decide to use a different pressure sensor, you will need to update this library).
Dallas Temperature Control. Allows you to communicate with the DS18B20 thermometer. Download this library here: https://github.com/milesburton/Arduino-Temperature-Control-Library.
OneWire. For controlling the Dallas 1-wire family of devices, which includes the temperature sensors. Download the library here: https://github.com/PaulStoffregen/OneWire.
RTClib. For controlling and reading the Real-time Clock. Download the library here: https://github.com/adafruit/RTClib.

The final library is a bit more complicated. The standard **SoftwareSerial** library is not compatible with SAMD microcontrollers like the Adafruit Feather M0. An updated library is available from the main Arduino branch, but it also requires modification to work with the Adalogger. The updated, functional library can be found here:

https://github.com/OceanographyforEveryone/OpenCTD/tree/master/OpenCTD_Feather_Adalogger/Support_Code/SoftwareSerial

GitHub does not allow you to download sub-folders within a repository. In order to copy a library from a GitHub Repository, you can either clone the entire repository to your computer and then navigate to the library within the file system or download the entire repository as a .zip file and the extract and navigate to the required library. Confirm that your set-up is correct by loading the OpenCTD_master_m0.ino Arduino sketch into Arduino IDE, setting the board to Adafruit M0, and pressing “Verify”. If the program compiles without error, make sure your Adalogger is connected and press “Upload”.

Benchmark 1: Confirm Software Installation and Adalogger/RTC assembly

Test the Adalogger M0 and real-time clock to confirm that they are working correctly by loading the OpenCTD software onto the Adalogger and using the serial monitor within Arduino IDE to confirm that the software is working correctly. The SD card must be inserted. You should see the date and time, followed by a series of zeros or negative values, since there are no sensors connected to read.

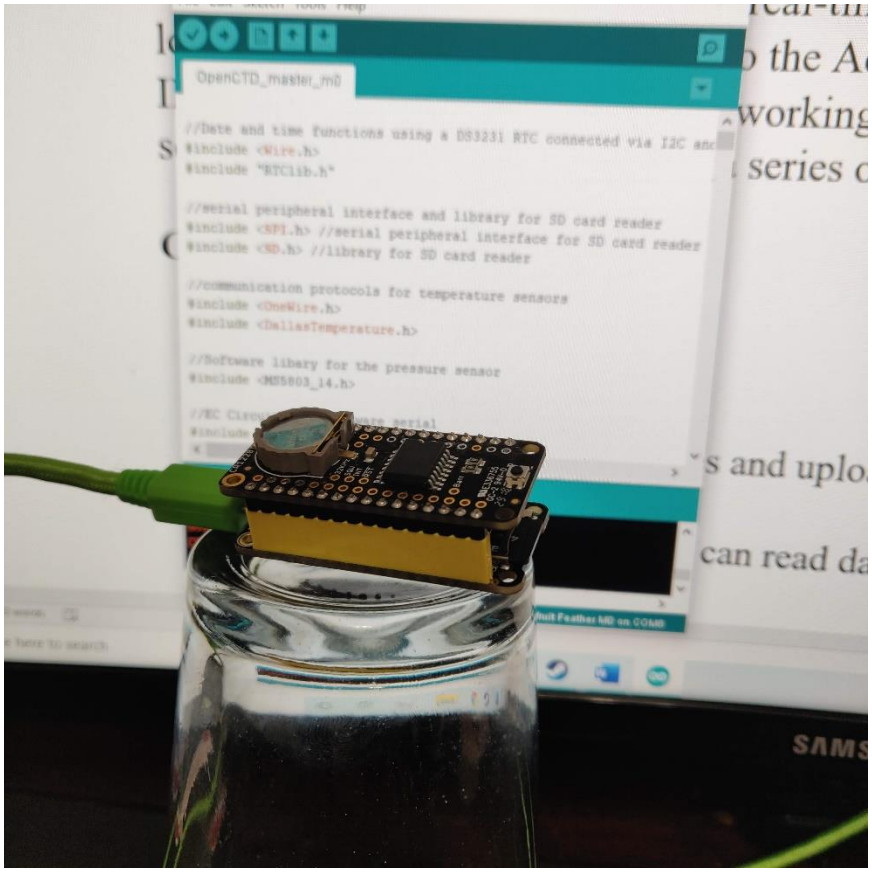


Figure 8. Adalogger connected to PC for software validation.

Checklist:

- Adalogger connects to PC.
- OpenCTD software compiles and uploads to Adalogger.
- Serial monitor connects and can read data.
- Real-time clock shows the correct time.

4.0 Sensor Assembly and Breadboard Testing

The OpenCTD uses three separate types of sensor to accurately and precisely measure the marine environment. The pressure sensor is a MS5803-14BA built by Measurement Specialties and is accurate to 1cm when properly calibrated. The pressure sensor also includes an on-chip thermistor; however, the onboard thermistor is very slow and is unused. For custom builds with extremely limited space that don't need the device to reach equilibrium quickly, users can opt to read temperature directly from the pressure sensor. The standard OpenCTD uses a battery of three DS18B20 thermistors sealed in a stainless-steel housing to take a temperature average of the ambient environment. Finally, the OpenCTD uses a commercially available Atlas EZO conductivity system consisting of a probe and conductivity circuit to determine salinity.

4.1 Pressure Sensor

Skills: Soldering
Consumables: Solder, solid-core wire
Parts: MS5803-14BA, SOIC8-to-DIP adapter, 100nF (104) capacitor, 10kOhm pull-up resistor

Parts 3. MS5803-14BA, SOIC8-to-DIP adapter, 100nF (104) capacitor, 10kOhm pull-up resistor.

Alternative Parts: Sparkfun Pressure Sensor Breakout or OpenROV 30 Bar + IMU

To build the pressure sensor, you will need to surface mount the pressure sensor chip onto an SOIC8-to-DIP adapter. This is the most challenging soldering step in the entire build and requires a steady hand. The pressure chip is able to withstand high heat for a short amount of time, as long as there is no direct contact. Be extremely careful not to touch the gel covering the pressure sensor.

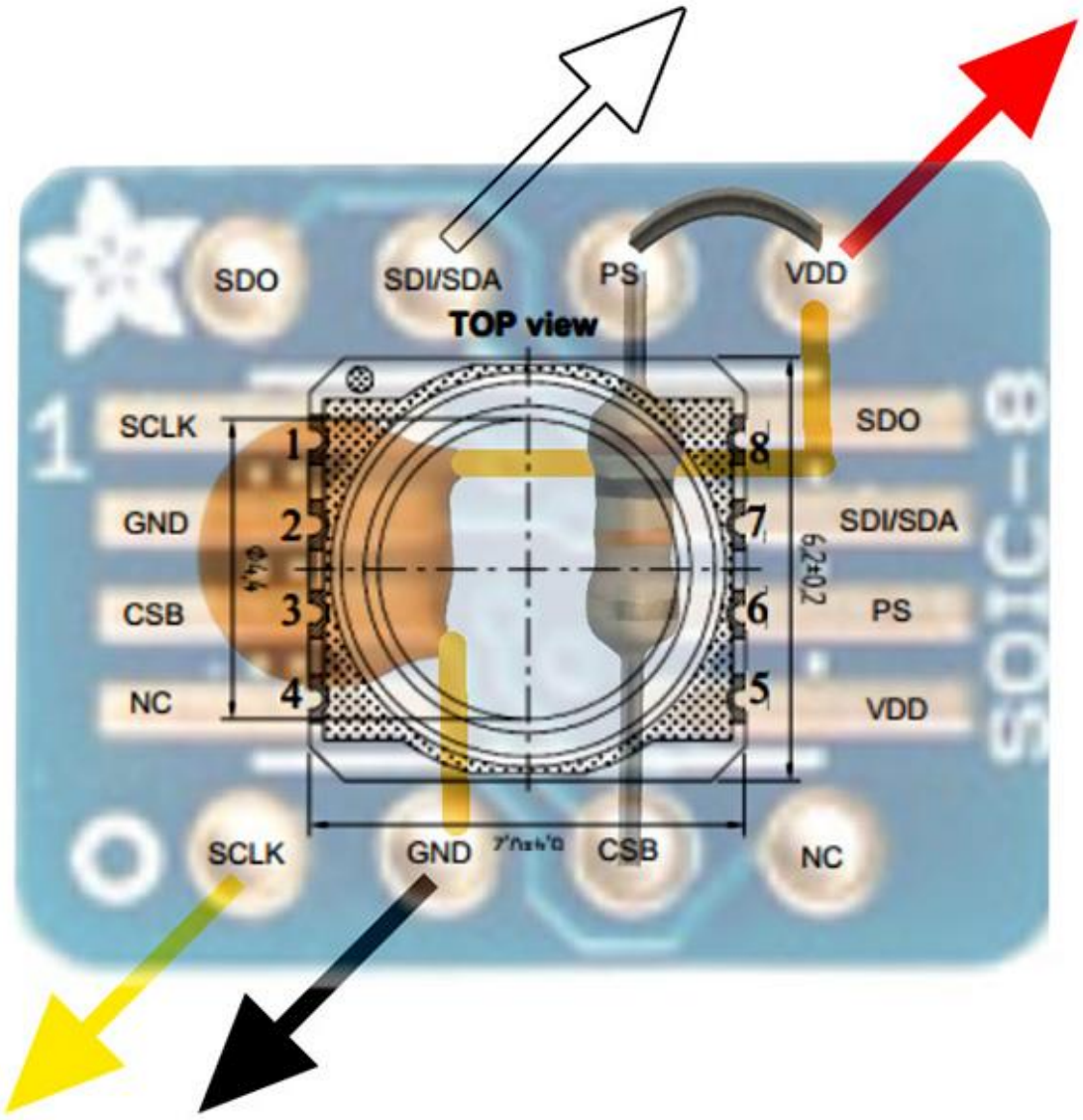


Figure 9. Diagram of pressure sensor chip and component alignment on SOIC-8 to DIP adapter. Note alignment of pin 1 on the chip to the alignment indicator (in this case, the Adafruit star, but on most boards simply a dot). Image courtesy of the Cave Pearl Project, who have an excellent guide to using and soldering these little chips here: https://thecavepearlproject.org/2014/03/27/adding-a-ms5803-02-high-resolution-pressure-sensor/).

1. Make sure that side of the SOIC8-to-DIP adapter with the larger pads is facing up. Add a small amount of solder to each pad. If you have solder rosin, add a dab of rosin to each pad to help the solder flow.

2. Ensure that the dot and notch on the pressure sensor line up with the dot on the SOIC-to-DIP adapter. Solder the sensor in place by applying a small amount of solder to the vertical grooves and drawing it down onto the solder pads. Be careful not to short any connections.

3. Bridge the pads for pin 5 (VDD/VCC) and pin 6 (PS) by drawing a small drop of solder between them.

4. Flip over the adapter and solder a 10kOhm pull-up resistor between pin 3 and 6.

5. Insert the 100nF (104) capacitor between pins 2 and 5 but don't solder yet. Bend the capacitor's legs to hold it in place.

6. Solder a 6 cm length of solid core wire to pin 1 (SCL) and pin 7 (SDA). Keep track of which wire goes to which pin by color-coding each pin (we recommend red for VCC; black for ground; yellow for SCL; and blue for SDA).

7. Solder a 6 cm length of solid core wire to pin 2 (GND) and pin 5 (VDD/VCC) such that the legs of the capacitor are captured in the solder join. You may have to twist the wire a bit to get both the wire and the capacitor leg to fit in a single pin hole. Keep track of which wire goes to which pin by color-coding each pin (we recommend red for VCC; black for ground; yellow for SCL; and blue for SDA).

8. Strip ½ cm of insulation from the exposed ends of each wire.

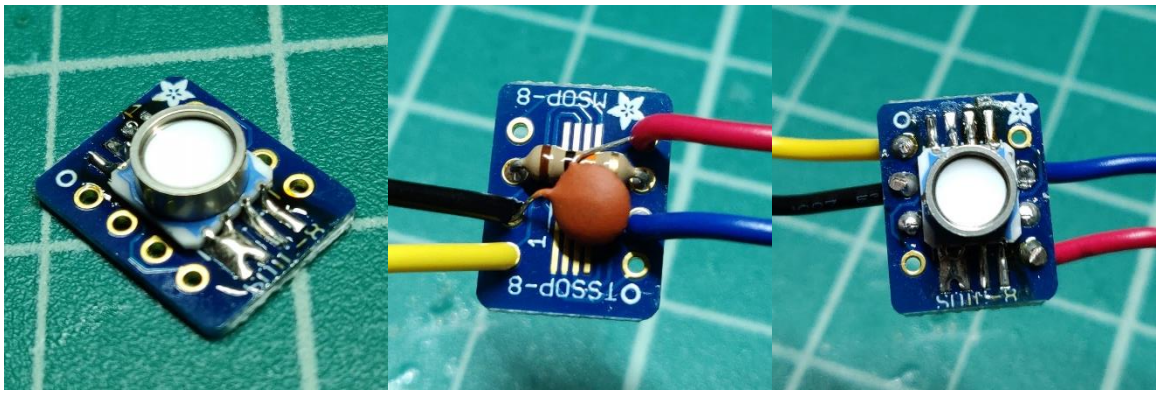


Figure 10. Building the Pressure Sensor unit. A: Components of the sensor. B: Surface mounting the sensor. C: Wiring up the components.

9. **ALTERNATIVE 1:** If you're using the Sparkfun Pressure Sensor Breakout Board, solder a 5 cm length of solid core wire to SCL, GND, VCC, and SDA. Keep track of which wire goes to which pin by color-coding each pin (we recommend red for VCC; black for GND; yellow for SCL; and blue for SDA).

10. **ALTERNATIVE 2:** If you've received an OpenROV IMU with 30Bar pressure chip, you can use that as an alternative for a deeper operating CTD. Solder a 5 cm length of solid core wire to SCL, GND, VCC, and SDA. Keep track of which wire goes to which pin by color-coding each pin (we recommend red for VCC; black for GND; yellow for SCL; and blue for SDA).

4.2 Temperature Sensors

Skills: Soldering
Consumables: Solder, heat shrink tubing, solid-core wire
Parts: DS18B20 thermistors

Parts 4. DS18B20 potted thermistor.

The OneWire protocol allows multiple digital sensors to communicate through a single signal, so we can connect all the thermistors to a shared data pin in the Adalogger. In preparing these sensors, our objective is to join the three units together with as little excess wire as possible but with enough slack that moving and positioning the probes do not put any extra stress on the solder joins. Note, that while there may be variation in the color of the data wire, black will always be ground and red will always be VCC

1. Cut the wire on each temperature probe approximately 5 cm from the probe-end terminus.

2. Strip 2 cm of outer insulation, taking care not to break the inner wires.

3. Strip a centimeter of insulation from each inner wire, exposing the metal strands.

4. Twist all three black wire strands together and solder. Do the same for the red and yellow wires, respectively (the colors may vary based on manufacturer, but the important thing is to ensure that matching colors are bundled in sets of three).

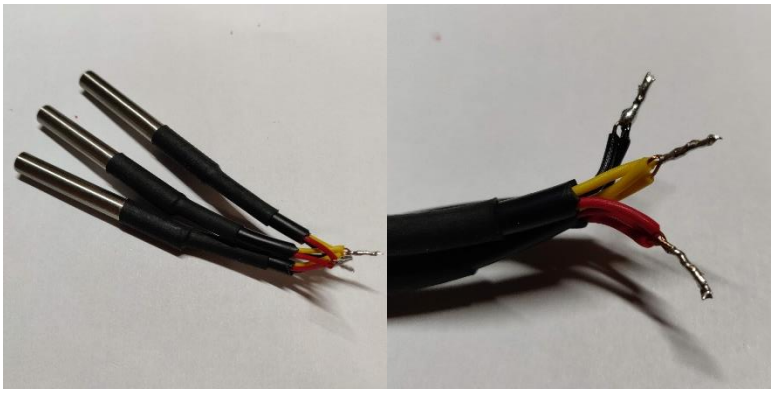


Figure 11. Three potted temperature sensors joined together with a close-up on the tinned tips.

4.3 Conductivity Sensor

Skills: Soldering
Consumables: Solder
Parts: Atlas K 1.0 conductivity probe or Atlas K 1.0 mini conductivity probe

---

Parts 5. Atlas K 1.0 mini conductivity probe

The Atlas conductivity probe and EZO circuit need very little preparation prior to testing. If your Atlas kit came with a BNC connector, no additional preparation is required prior to Benchmark 2, however you will have to do step 1 through 4 after breadboarding.

1. Cut the probe wire about 4 cm above the probe-end terminus.

2. Use a utility knife to strip and expose 2 cm of wire. The wires may be embedded in extruded insulation rather than simply wrapped and will require a bit of care to extract without damaging the internal insulation.

3. Strip half a centimeter of insulation from each inner wire, exposing the metal strands.

4. Tin the exposed wire stands with a small amount of solder.

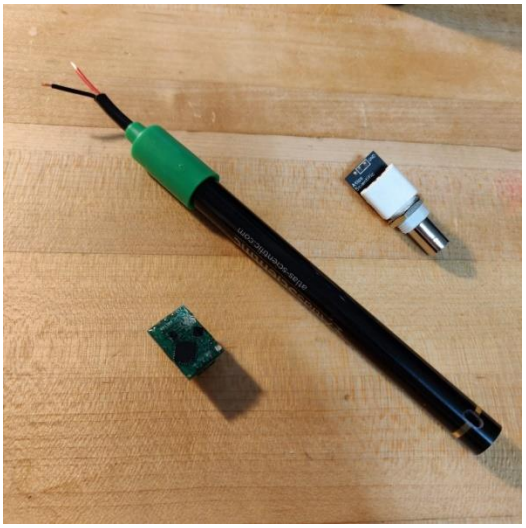


Figure 12. Conductivity probe with stripped lead wires, Atlas EZO conductivity circuit, BNC connector for simplified testing.

Benchmark 2: Breadboarding the Circuit

A breadboard is a tool that allows you to prototype an electronic circuit without permanently affixing components. A breadboard is an electronic pegboard with three major regions. The two strips on either side of the board are the ground and positive rail. They are connected vertically. The rails in between are for affixing components. They are connected horizontally and usually clustered in groups of 5. In the center of the breadboard is a dividing groove. Horizontal rails are not connected across the groove. This allows you to mount large components with pins on either side of the central groove, like, for example, an Adalogger. By breadboarding the circuit first, you can ensure that the system is working properly before continuing.

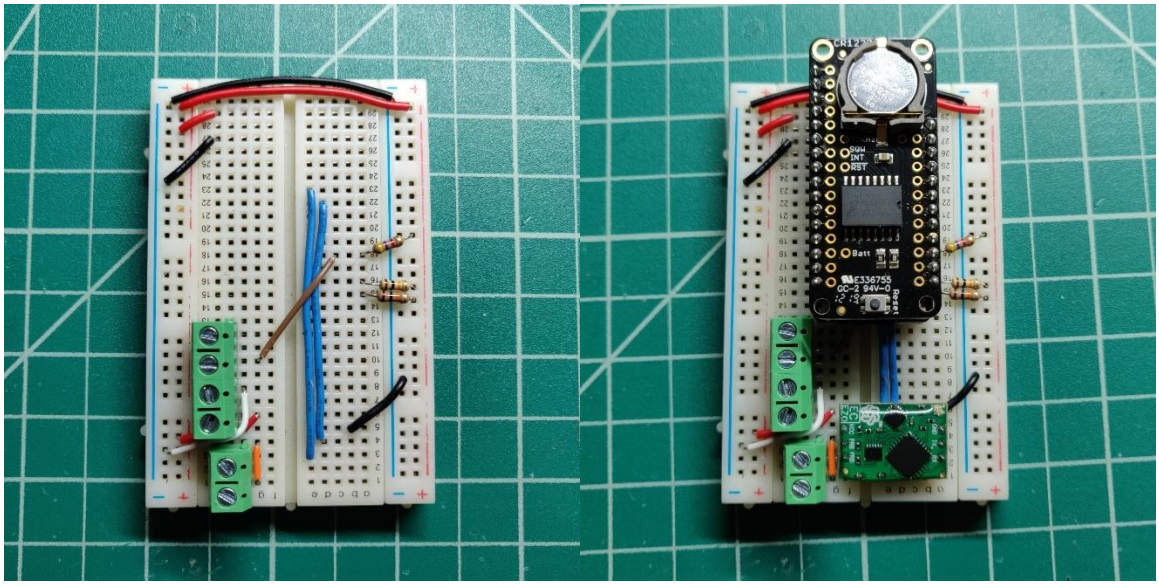


Figure 13. Breadboard with just jumper wires, screw terminals, and resistors attached and with Adalogger/RTC and EZO conductivity circuit.

1. Reference Figure 9 and the photo and pinout guide below. Place Atlas EZO circuit and connect jumper wires in accordance.

2. Connect a 4.7 kΩ resistor between pin D6 and the positive voltage rail.

3. Connect a 10 kΩ resistor between the SDA pin and the positive voltage rail and connect another 10 kΩ resistor between SCL pins and the positive voltage rail.

4. Connect the temperature sensors to digital pin 6, positive, and ground using the screw terminal blocks.

5. Connect the pressure sensor to SCL, SDA, 3.3V, and GND. The solid core wire on the pressure sensor can be loosely slotted directly into the pin holes on the RTC.

6. Connect the temperature probe to the two probe pins on the Atlas EZO using the screw terminal blocks or BNC connector. It does not matter which wire goes to which probe pin. Make sure the conductivity probe itself is submerged or it will read zero.

7. Connect the Adalogger to your PC using a micro-USB cable and open the serial monitor in Arduino IDE.

8. Power the Adalogger down, remove the SD card, and read the data file.

Breadboard Diagram

Note: this diagram does not show the screw terminals. Refer to Figure 13 to see the position of the screw terminals.

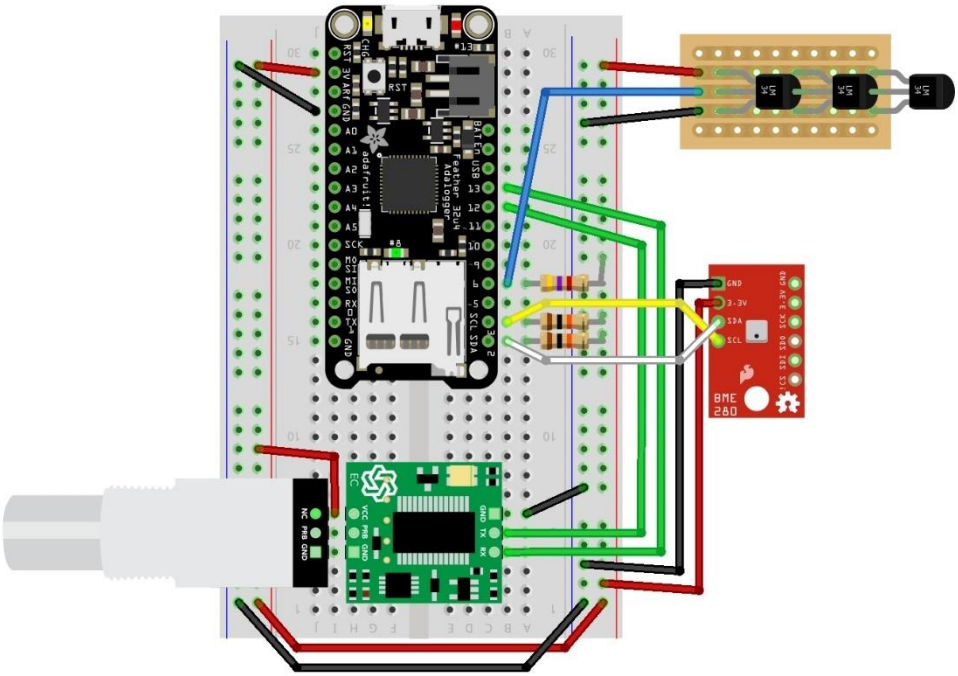


Figure 14. Diagram of breadboard assembly to test completed sensors. Note the presence of BNC connector for the conductivity probe and the use of a barometric pressure sensor to stand in for the OpenCTD pressure sensor.

Checklist

- Atlas EZO circuit is blinking with alternating between green and blue lights.
- Temperature probes are reading ambient temperature and are within 0.5^{o}C of each other.
- Pressure sensor is reading atmospheric pressure (it should be around 1012 mbar if you're near sea level).
- Conductivity sensor is reading the conductivity of whatever fluid it is in.
- Data had been logged to SD Card.

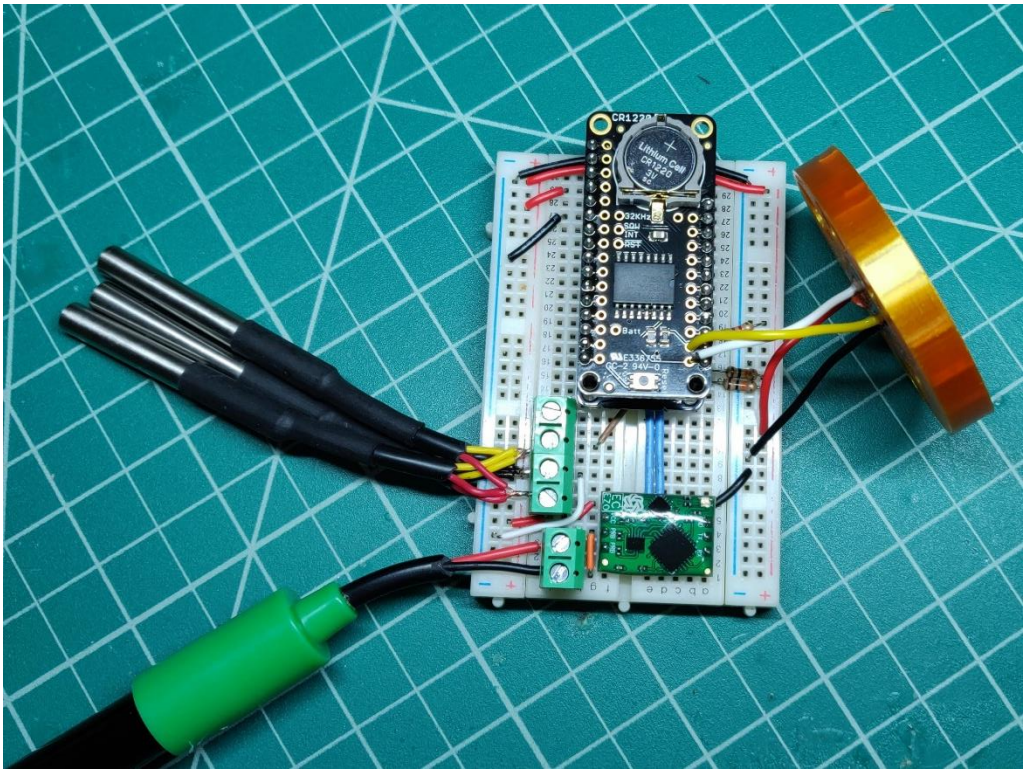


Figure 15. Completed breadboard set-up with all sensors ready for testing.

5.0 Mounting and Potting Sensors

Potting the sensors is the point of no return. Once they are sealed in epoxy, they cannot be removed. It is essential that you confirm all sensors are operating correctly before final potting. If you have cleared the previous benchmarks, you are ready to go.

5.1 Baseplate and Potting the Pressure Sensor

Skills: Epoxy Potting
Consumables: Hot glue, instant epoxy
Parts: Pressure sensor assembly, temperature probe assembly, conductivity probe, 3D-printed baseplate

Parts 6. Pressure sensor assembly, temperature probe assembly, conductivity probe, 3D-printed baseplate

The 3D printed baseplate provides a template for accurately aligning all five sensors as well as an end-stop to prevent epoxy from leaking out during potting. The pressure sensor sits in the center of the template and fits snuggly into the central hole. This sensor will be potted first with instant epoxy to protect the circuits before the full sensor assembly is potted. Be extremely careful not to get any epoxy on the gel of the pressure sensor.

1. Seat the pressure sensor into the central hole on the side with the large square indentation and ensure that the actual sensor sits flush with the bottom face of the baseplate.

2. Once the sensor is correctly seated, apply just enough instant epoxy to fill the entire central rectangular indentation. The epoxy will settle during pouring as it flows under the circuit board.

3. Carefully check that no epoxy is leaking out through the central hole.

4. Let the assembly sit for a 10 to 20 minutes to allow the epoxy to set.

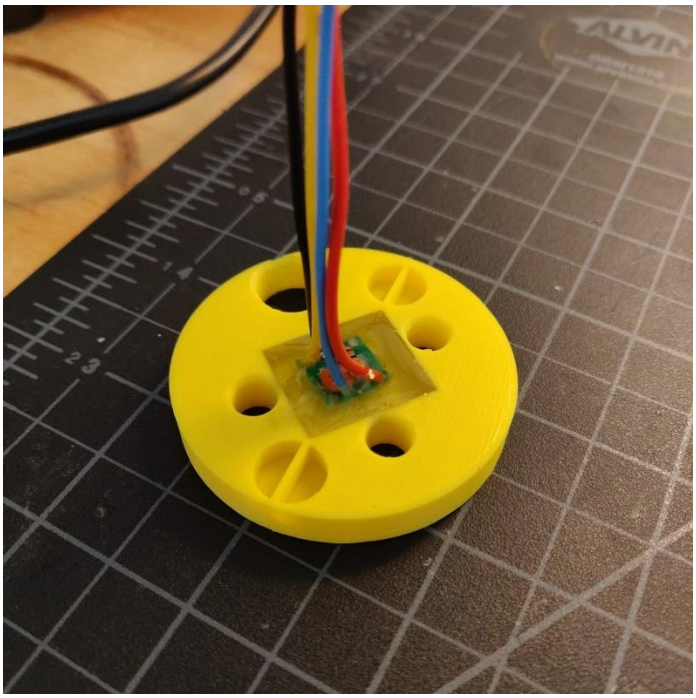


Figure 16. Pressure sensor seated and potted in baseplate.

5. Once the epoxy has set, insert each temperature probe into the three holes such that 1.5 to 2 cm extends out of the bottom of the baseplate.

6. Hot glue each temperature probe into place, sealing any possible holes. The hot glue will prevent epoxy from leaking out through these pass-through holes during potting.

7. Insert the conductivity probe into the large hole such that 1.5 to 2 cm extends out of the bottom of the baseplate.

8. Hot glue the conductivity sensor into place, sealing any possible holes. The hot glue will prevent epoxy from leaking out through these pass-through holes during potting.

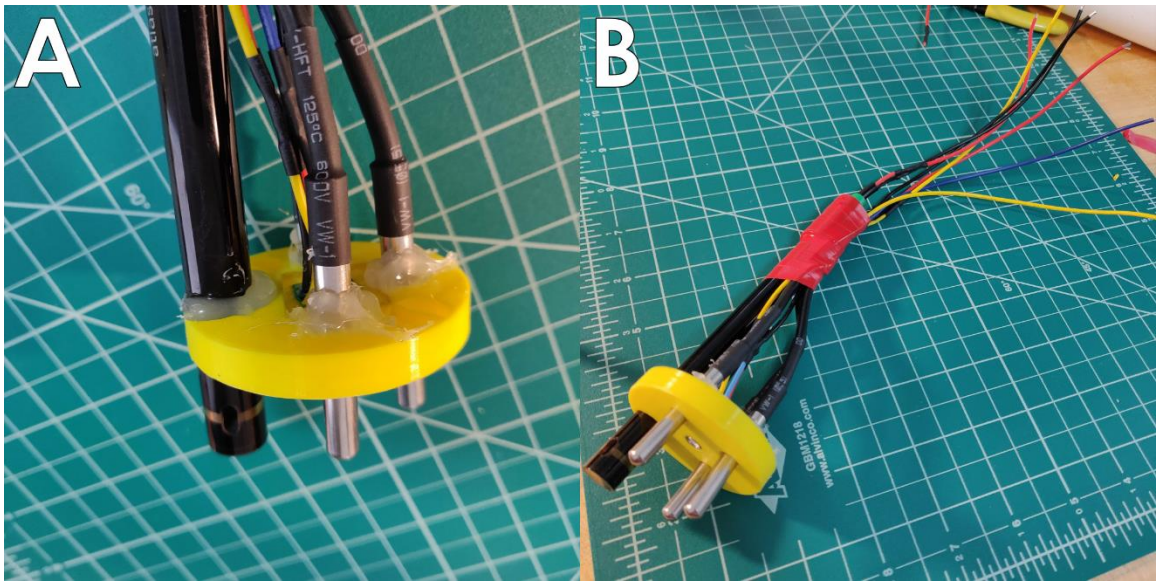
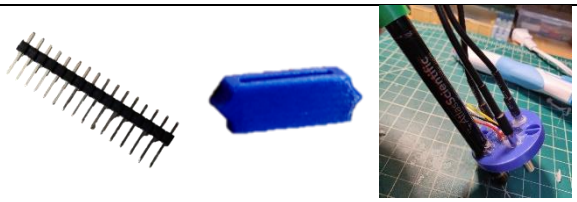


Figure 17. Sensors secured to baseplate with hot glue (A); forward view of sensors passing through baseplate.

5.2 Creating the Connector

Skills: Soldering
Consumables: solder, solid-core wire, 20-gauge wire, super glue, heat shrink tubing
Parts: 8-pin male header, 3D-printed connector housing, sensor assembly



Parts 7. Male header, 3D-printed connector housing, sensor assembly

The 8-pin male connector makes it easy to attach and detach the sensors from the Master Control Unit. At the end, you will have 7 lines coming from the sensors, including a positive and ground line, SDA and SCL from the pressure sensor, data from the temperature probes, and two lines from the conductivity probe.

1. Solder a 5 cm length of 20-gauge wire to the positive (VCC) and ground (GND) wires of the pressure sensors and seal the connection with heat shrink tubing.

2. Solder the positive wires from the pressure and temperature sensors together, add a 15 cm length of wire to each joined end and seal with heat shrink.

3. Solder the ground wires from the pressure and temperature sensors together, add a 15 cm length of wire to each joined end and seal with heat shrink.

4. Attach a 25 cm length of 20-gauge wire to the SDA and SCL wires of the pressure sensor and seal with heat shrink.

5. Attach a 15 cm length of 20-gauge wire to each of the two wires of the conductivity probe and seal with heat shrink.

6. All wires should extend about 1 cm beyond the edge of PVC pipe when the baseplate is dry fit into position. Cut off any excess so that all wires are of equal length. Use electrical tape to tape the free wires to the side of the conductivity probe. This will relieve strain on the solder joins.

7. Strip and tin the ends of the sensor wires. Cut the braided sleeve to length and slide down over the sensor wires. This will provide additional protection to the wires and solder joins.

8. Prepare an 8-pin header by tinning the short ends of each pin.

9. Solder the sensor wires to an 8-pin header. They will align with their respective contacts on the Master Control Unit 8-pin female output. Take care not to apply too much heat to the header or the plastic strip will melt. Seal each pin with heat shrink tubing.

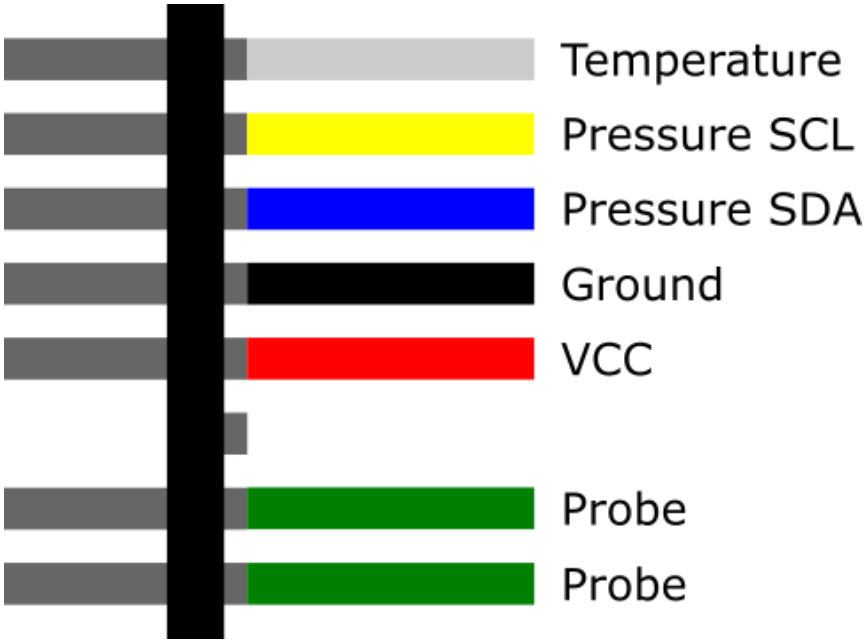


Figure 18. Illustration of connector with wires coming up from the housing. Wires leading to sensors on right, vertical black bar represents 8-pin headers. Pins leading to MCU are on the left.

10. Test the connector by plugging it into the breadboard and reading the sensors. You will have to rearrange the breadboard in order to connect to all seven leads. You can also test each sensor independently using jumper wires.

11. Slide the 3D-printed connector cover over the 8-pin connector.

12. Fill the entire void within the connector with gel-based superglue. This prevents strain on the header and solder joints as well as prevent moisture from entering the connector and prevents short circuits. Allow the glue to cure for 30 minutes.

13. Once cured, test the connection again by plugging it into the breadboard and reading the sensors. You will have to rearrange the breadboard in order to connect to all seven leads. You can also test each sensor independently using jumper wires.

5.3 Potting the OpenCTD

Skills: Epoxy Potting
Consumables: EA-90FL Epoxy (or alternative 2-part epoxy) with 50mL epoxy gun, sandpaper
Parts: 12" PVC Pipe, sensor assembly

Parts 8. 12" PVC Pipe, sensor assembly

It is time to permanently seal the base of the OpenCTD. Once the epoxy has been mixed and injected, it is essential that you do no move or disturb the housing until the epoxy cures. Within the first 20 minutes, you may gently lift the housing vertically to check that no epoxy is leaking out the bottom. After about 2 hours of cure time, the epoxy should have set enough to allow you to continue the build process without disturbing it. Do not submerge the OpenCTD until the epoxy has cured for at least 24 hours.

1. Ensure that you can slide the sensor baseplate up into the bottom of the CTD so that it sits just above the flow-through holes or notches with the sensors protruding out the bottom but protected by the housing. If the fit is too tight, use a deburring tool to knock down the edges of the 3D-printed parts.

2. Gently sand the inner rim of the pipe above where the baseplate will sit to provide extra contact area for the epoxy.

3. Remove the sensor assembly and lay a lip of hot glue around the edge of the PVC pipe just above where the baseplate will sit. This will act as a gasket to prevent epoxy from leaking out onto the sensors.

4. While the glue is still hot, quickly push the baseplate up into position. This will prevent epoxy from leaking out through the bottom of the CTD. If you wait too long and the glue hardens, remove the baseplate, scrape of the glue, and repeat until the baseplate is properly seated.

5. Ensure that the inside of the PVC pipe is clean and free of obstruction. The sensor connector should sit just above the rim of the PVC pipe.

ENSURE EVERYTHING IS SET CORRECTLY
BEFORE PROCEEDING TO THE NEXT STEP.

6. Using the 50mL cartridge gun and venturi mixing nozzle, inject the full 50mL of 2-part epoxy into the OpenCTD housing.

7. Check to ensure that nothing is leaking and let the epoxy cure undisturbed for 24 hours.

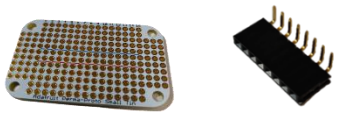
Figure 19. A clear OpenCTD showing the thick epoxy potting layer.

6.0 Assembling the Master Control Unit

The Master Control Unit is the nervous system of the OpenCTD. It handles all the processing and signal routing between the sensors and the Adalogger. The MCU uses a mint-tin sized permanent breadboard, which is arranged similar to a standard breadboard, but with positive and negative rails in the center of the board and free, unattached pins at either end. This Perma-Protoboard allows you to make permanent electronic connections while mimicking the look and set-up of a breadboard.

6.1 Preparation

Skills: Soldering
Consumables: Solder
Parts: Adafruit Perma-proto board, 8-pin right angle female header



Parts 9. Adafruit Perma-proto board, 8-pin right angle female header.

First, we will build the connector that interfaces with the sensors in the OpenCTD housing by attaching an 8-pin, right angle female connector to the end of the Perma-proto board and bridging it to the positive and negative rail with solder.

1. Once the Adalogger has been tested, it no longer needs the long pins that connected it to the breadboard. Snip these pins off at the base of the Adalogger. The Adalogger should now sit flush inside the Electronics Chassis.

2. Cut an 8-pin length from a right-angle female header strip.

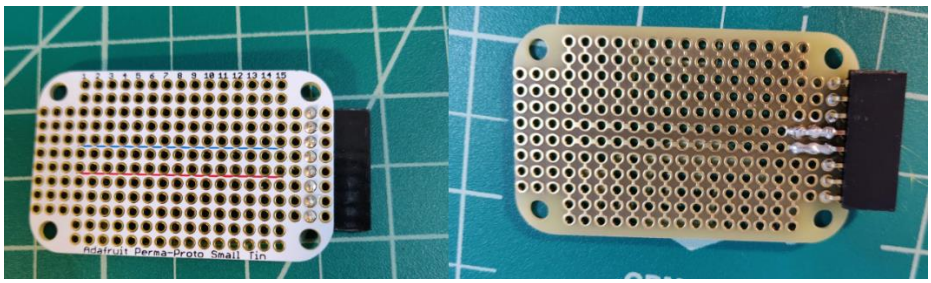


Figure 20. Perma-Protoboard with 8-pin connector top (A) and bottom, with solder bridges (B).

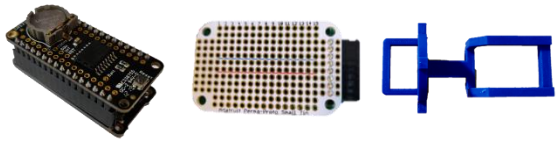
3. Solder the 8-pin connector to the second to last set of through-holes on the right side of the Perma-Protoboard, on the underside such that the pins emerge through the middle column of unconnected through-holes.

4. On the underside of the board, build a solder bridge between the positive and negative strips and their respective, adjacent connector pins. This works best with a soldering iron heated to 750^{o}C. Hotter irons will cause the surrounding solder to melt.

5. **ALTERNATIVE:** If you have trouble building solder bridges, use small lengths of solid core wire to jump the contacts.

6.2 Powering the Protoboard

Skills: Soldering
Consumables: Solder, solid core wire
Parts: Adalogger/RTC, Adafruit Perma-proto board with headers, 3D-printed electronics chassis



Parts 10. Adalogger/RTC, Adafruit Perma-proto board with headers, 3D-printed electronics chassis

Solid core wire will be connected to pass-through holes on the RTC and soldered underneath so that the wires pass between the RTC and the Adalogger and then contact the appropriate pins on the perma-proto board. It is easiest to start with long lengths of wire and cut down to appropriate sizes as you proceed.

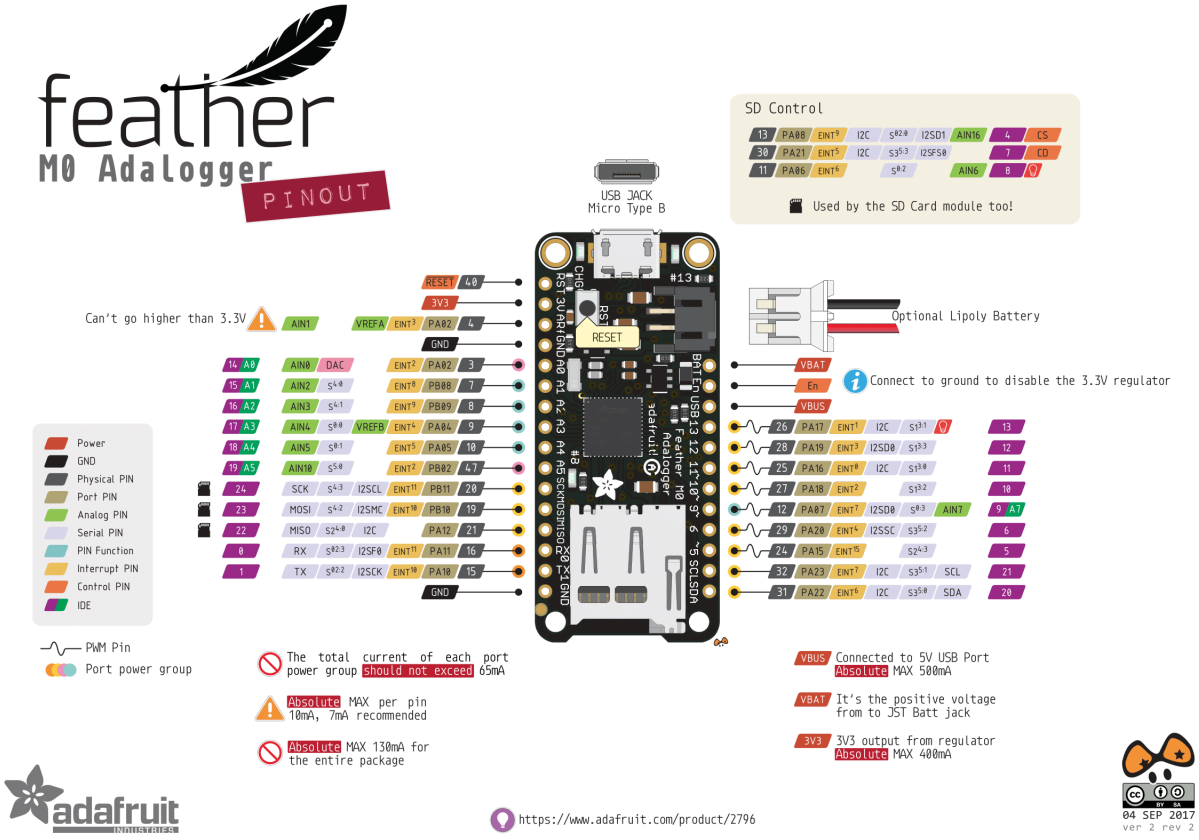


Figure 21. Pinouts for the Adalogger Feather M0. Note, these pinouts also correspond to those of the RTC.

1. Connect red and black solid core wire to the 3.3V and GND pins on the left side of the RTC.

2. Place the complete Adalogger/RTC assembly into the Electronics Chassis and stretch these two wires until the meet the end of the Chassis.

3. Cut the wires near the far end of the zip-tie hole in the e-chassis and strip off half a centimeter of insulation from the leads.

4. Solder these wires to the positive (3.3V) and negative (GND) rails of the Perma-Protoboard along columns 5 or 6.

5. Check the fit by placing the entire unit in the Electronics Chassis. The Adalogger and the Perma-Protoboard should sit flush against their respective support structures.

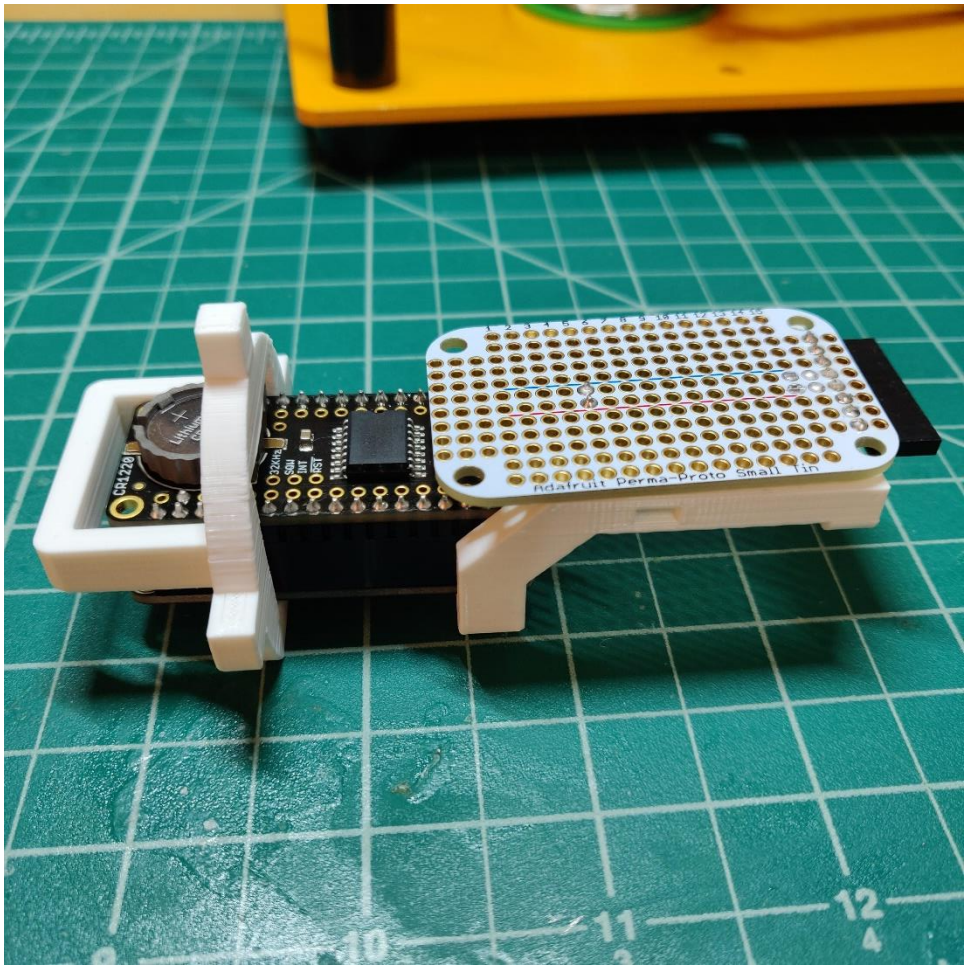
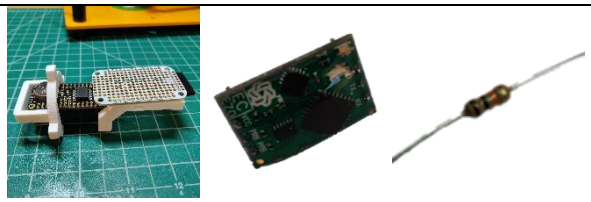


Figure 22. The Adalogger/RTC fit into the electronic chassis.

6.3 Populating the Perma-Protoboard

Skills: Soldering
Consumables: Solder, solid-core wire
Parts: Adalogger/RTC connected to Adafruit Perma-proto board, Atlas EZO circuit, 10 kOhm pull-up resistor (x2), 4.7 kOhm pull-up resistor



Parts 11. Adalogger/RTC connected to Adafruit Perma-proto board, Atlas EZO circuit, resistor

These steps will fully populate the MCU with all necessary components for the OpenCTD. We advise laying out all components first and double checking their placement before soldering. Reference Figures 23 and 24 frequently during assembly.

1. Solder the Atlas EZO circuit to columns 1 through 3 on the Perma-protoboard such that the GND and VCC pins correspond to the correct power rails. Holding the board so that "Adafruit Perma-Proto Small Tin" is correctly oriented, GND/TX/RX should be on top and VCC/PRB/PRB should be below.

2. On the underside of the Perma-proto board, bridge the GND pin of the Atlas EZO circuit to the negative rail with solder.

3. On the underside of the Perma-proto board, bridge the VCC pin of the Atlas EZO circuit to the positive rail with solder.

4. Using solid core wire, connect pin 12 on the RTC to TX on the Atlas EZO.

5. Using solid core wire, connect pin 13 on the RTC to RX on the Atlas EZO.

6. Connect each PRB pin to the through holes adjacent to pins 7 and 8 on the 8-pin right-angle header. It does not matter which PRB pin connects to which of the two header pins.

7. Connect the 4.7 kOhm pull-up resistor to column 8 and the positive (red) rail.

8. Connect a 10 kOhm pull-up resistor to column 10 and the positive (red) rail.

9. Connect a 10 kOhm pull-up resistor to column 11 and the positive (red) rail.

10. Using solid core wire, connect pin 6 on the RTC to upper column 8 on the Perma-proto board.

11. Using solid core wire, connect the SCL pin on the RTC to upper column 10 on the Perma-proto board.

12. Using solid core wire, connect the SDA pin on the RTC to upper column 11 on the Perma-proto board.

13. Using solid core wire, connect upper column 8 on the Perma-proto board to the through hole adjacent to pin 1 on the 8-pin right-angle header.

14. Using solid core wire, connect upper column 10 on the Perma-proto board to the through hole adjacent to pin 2 on the 8-pin right-angle header.

15. Using solid core wire, connect upper column 11 on the Perma-proto board to the through hole adjacent to pin 3 on the 8-pin right-angle header.

16. Starting from the inside pins closest to the positive and negative rails, bridge the 8-pin right-angle header pins to the respective adjacent solid core wires on the perma-proto board with solder.

17. Clip off the one unused pin from the male end of the sensor connector and slot it into the unused female header on the Master Control Unit. This will prevent the connector from being inserted into the MCU backwards, causing a short.

18. Using zip ties, attach the Adalogger/RTC and Perma-proto board to the 3D-printed electronics chassis.

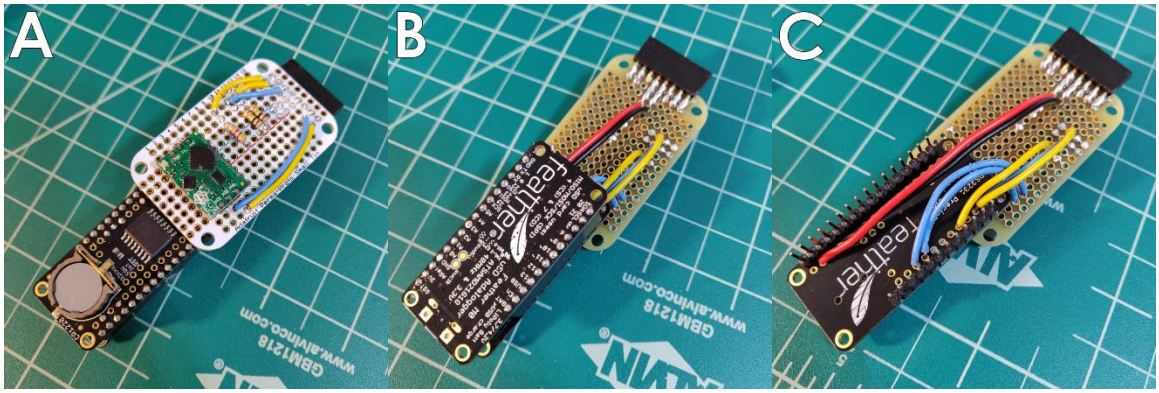


Figure 23. Fully connected MCU shown from the top (A), bottom (B), and bottom but without Adalogger attached (C).

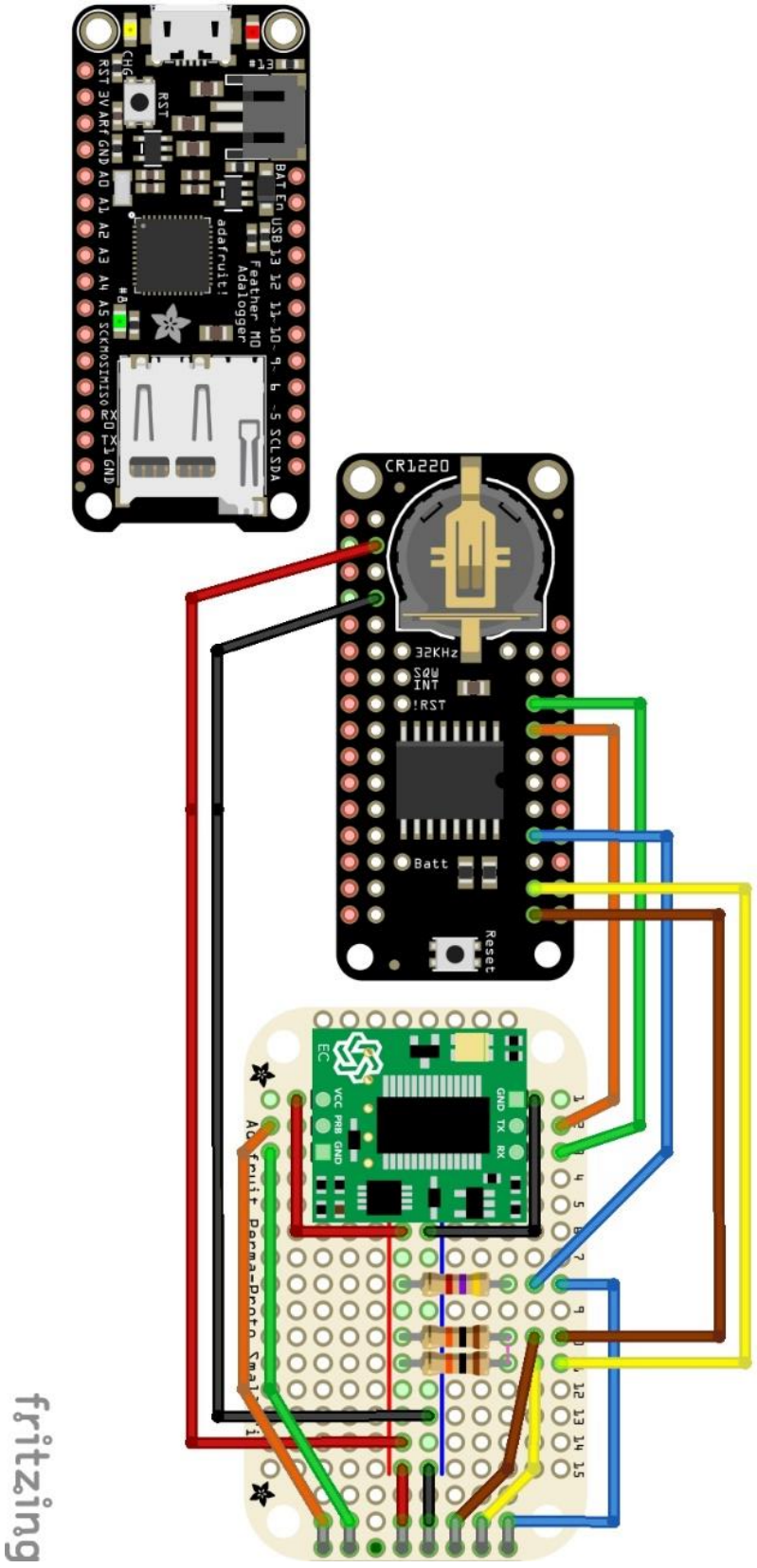


Figure 24. Layout of connections on the Perma-Protoboard.

Benchmark 3 – Testing the Master Control Unit

This test will ensure that the Master Control Unit has been assembled correctly and that all sensors are being read. Once you confirm that the MCU functions you will be ready to move on to the final phase of the OpenCTD build.

1. Holding the Perma-Protoboard face up with the connector on the right, the order of pinouts is, from top to bottom, D6 (Temperature), SCL (Pressure), SDA (Pressure), Ground, 3.3 Volt, UNUSED, Probe (-), Probe (+). Note that, as the conductivity probe functions like a resistor, it does not matter which line from the probe connects to which of the two pins.

2. Connect the MCU to the sensor connector.

3. Connect the Adalogger to your computer and check the serial monitor. If all sensors are reading accurately, you're ready to prepare and pot the sensors.

Checklist

- MCU can read pressure sensor when connected.
- MCU can read temperature sensors when connected.
- MCU can read conductivity sensor when connected.

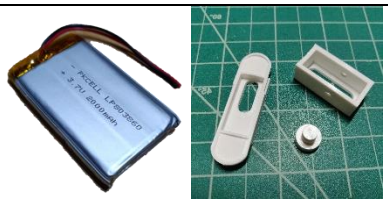
7.0 Final Assembly

The final step in assembly is building the battery switch and inserting the battery into the housing. The Adalogger has an internal 100mA charging unit, which means a standard 2000mAh battery will take 20 hours to charge. Using an external LiPoly battery charger with higher amperage can dramatically speed up the charging process. For charging, we use this 500 mA charger (https://www.adafruit.com/product/1904) which cuts the charging time on a 2000mAh battery down from 20 hours to 4 hours.

Be very careful with heat near a lithium-polymer battery.

7.1 Battery and Switch

Skills: Soldering
Consumables: Solid-core wire, 20-gauge wire, 3M dot adhesive, heat shrink tubing, electrical tape
Parts: LiPo battery, 3D-printed battery switch parts, round rare-earth magnet (2)



Parts 12. LiPo Battery and 3D-printed switch assembly.

The battery assembly is designed to seat independent of the rest of the electronics. This allows the battery leads to pass through a magnetic switch connected to the inner wall of the PVC tube and permits users to activate and deactivate the OpenCTD without opening the housing. The battery assembly is designed as a separate unit from the rest of the electronics to make it easier to replace the battery when necessary and to speed the rate of charging.

1. Assemble the inner switch assembly by running two 5 cm lengths of stripped solid core wire into the associated holes in the 3D-printed assembly.

2. Pinch the wires into place and superglue the outer, insulated wires to the edge of the assembly.

3. Clip off the battery leads about 2 cm from the base. Be careful not to let the leads contact each other.

4. Solder and seal enough wire to the battery leads to allow the battery to sit comfortably in the bottom of the CTD housing (about 10 cm). To reduce strain on the soldered battery leads, tape the wires to the body of the battery.

5. Now solder the wires with the JST connector to the switch assembly above the switch so that they emerge from the top of the CTD. Leave enough wire that you can easily attach and detach the JST connector from the Adalogger.

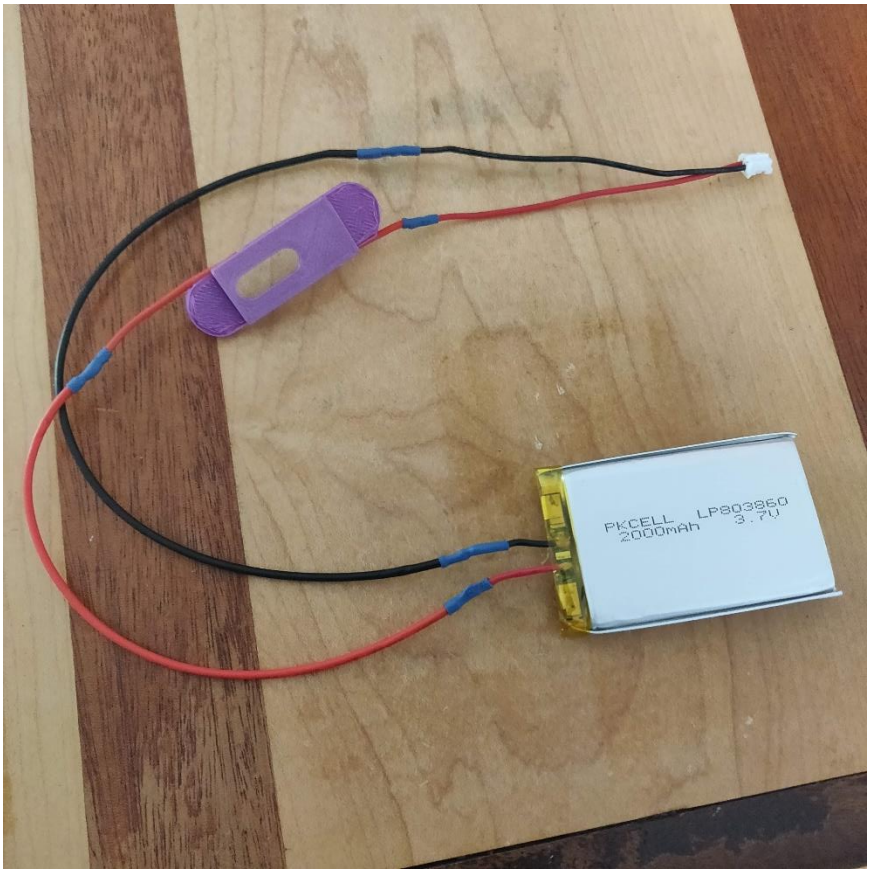


Figure 25. Inner battery switch assembly before battery is taped.

6. Place a 10 mm round magnet inside the housing (you can hold it in place by putting a magnet on the outside, too).

7. Using 10mm adhesive pads, adhere the internal switch component to the wall of the housing. Ensure that the internal magnet moves freely and contacts the two expose wires and that the switch assembly is seated low enough that it does not interfere with the test cap.

8. Stick the switch pin to another magnet using the 10mm double-sided pads.

9. Superglue the external switch assembly to the outside of the PVC pipe such that it traps the pin, is even and parallel with the internal assembly, and the pin can move freely. Line the switch up carefully and tape in place with electrical tape. Take extra caution not to allow the glue to leak into the inner walls of the external switch assembly or it could interfere with movement of the magnet.

10. Connect the MCU to the battery and ensure that the switch works consistently.

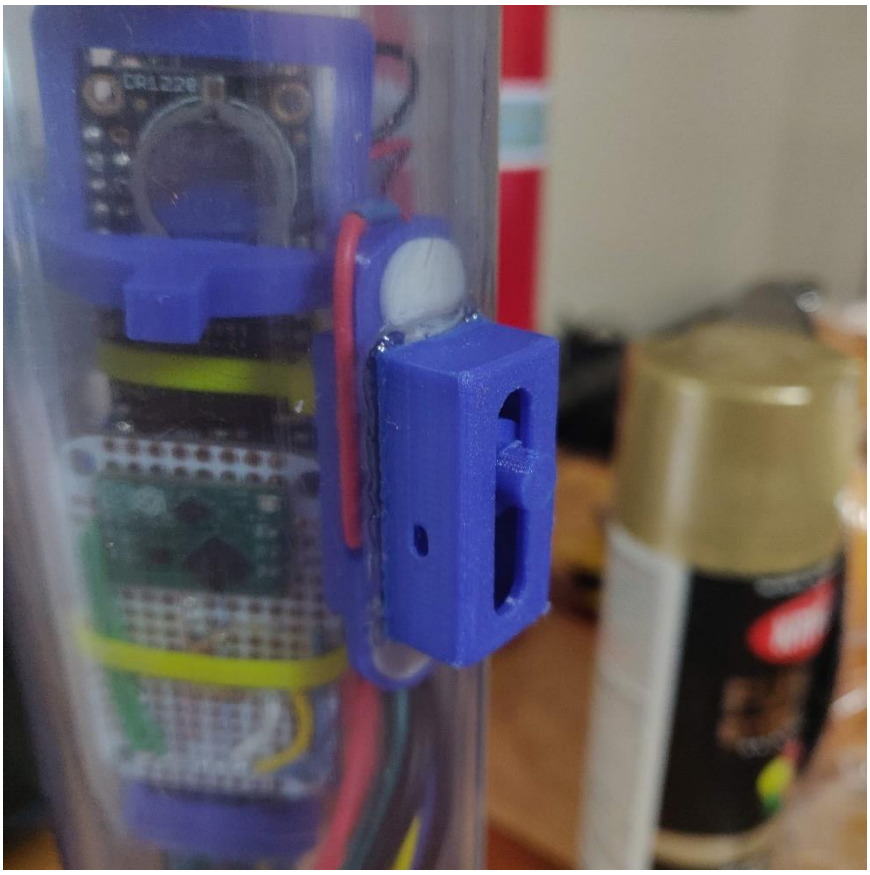


Figure 26. The complete switch assembly in a clear OpenCTD.

7.2 Casting Loop

This is an optional addition for users who want to conduct hand casts with the OpenCTD. As there are as many different configurations for casting as there are use-cases for the OpenCTD, we recommend developing your own solution to mounting or casting. A basic, durable loop for clipping to a handline can be constructed using a short length of polypro rope and a 2” hose clamp, as picture below. For extra security and to avoid snags, we recommend that you also wrap the hose clamp in electrical tape.

Figure 27. Finished OpenCTD with optional casting loop.

8.0 Calibration and Data Management

Comprehensive validation of the OpenCTD requires access to professionally calibrated instruments. These can be commercial CTD, handheld systems, or bench-mounted laboratory systems. As the community grows, validated OpenCTDs can be used to benchmark newly built instruments. It is not necessary to do a full validation test for every instrument, only those for whom a high degree of confidence and extreme precision are needed. For most use cases, an OpenCTD calibrated against salinity standards and general-purpose temperature sensor is adequate. The template OpenCTD_DataConversion_Template.xls available in the OpenCTD GitHub repository will take raw data from the OpenCTD and make the necessary conversions for you.

8.1 Pressure (Depth)

The OpenCTD uses a 14-bar pressure sensitive chip from Measurement Specialties designed, initially, for SCUBA dive watches and depth gauges. This chip contains a pressure sensitive resistor embedded in a gel matrix. It is important while handling not to damage the gel or allow anything to come in contact with it. Sand is particularly destructive, so take extra care when working from a beach. The pressure sensor is rated to 140 meters depth. A 30-bar chip, which can double the operating depth of the OpenCTD, is also available, though requires additional customization.

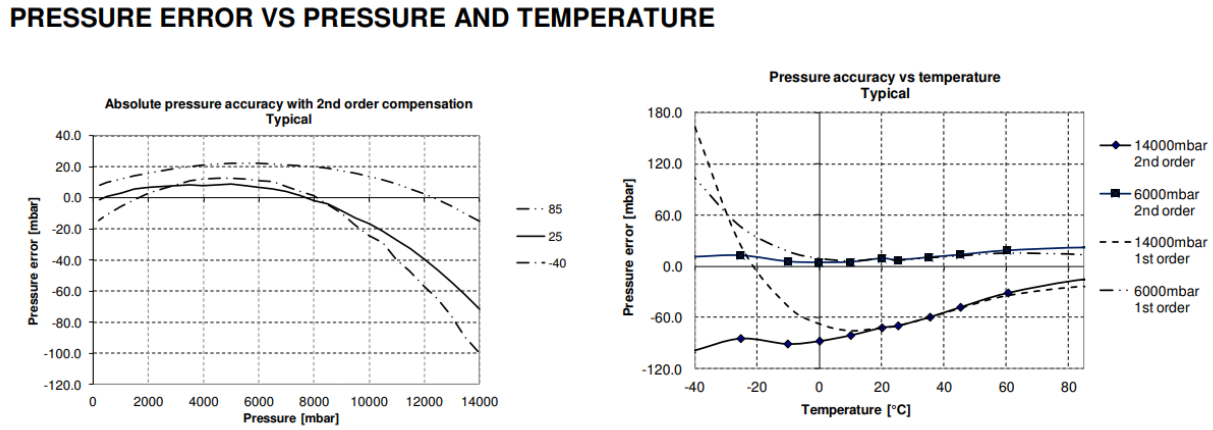


Figure 28. Pressure sensor error versus ambient pressure and temperature for the MS5803-14BA pressure sensor. Provided by manufacturer.

The MS5803 is a factory calibrated pressure sensor capable of outputting absolute pressure up to 0.2 millibar resolution and temperature with a resolution of 0.01 °C. The pressure sensor outputs absolute pressure, which means that it is self-calibrating. At sea level, standard atmospheric pressure should be 1013.5 mbar, though local weather and small changes in altitude will affect this baseline. In order to minimize the amount of processing that happens onboard the

OpenCTD, pressure is measured in millibars (the default output for the current pressure sensor). To convert pressure to depth, use the following equation:

$$D = (P(\text{DEPTH}) - P(\text{SURFACE}) * 100) / (G * 1000)$$

Where D is depth in meters; **P(DEPTH)** is the pressure (in millibars) at depth); **P(SURFACE)** is the pressure at the surface in millibars; and **G** is acceleration due to gravity, which for most field purposes can be assumed as 9.81 m/s^2 (Fofonoff and Millard Jr, 1983). For high-accuracy and high-resolution commercial CTDs, the latitudinal variation in gravity is considered in the pressure to depth conversion as these sensors often offer sub-centimeter accuracy.

8.2 Temperature

The OpenCTD uses a DS18B20 digital thermometer potted in a stainless-steel cladding. These sensors communicate over a 1-wire protocol, which allows us to connect multiple sensors to the same data pin. The sensors have an operating range of -55 to 125°C (with peak accuracy between -10 to 85°C). The advertised accuracy is ±0.5°C, however, in field trials using a battery of 3 sensors, we have seen accuracy as low as ±0.1°C.

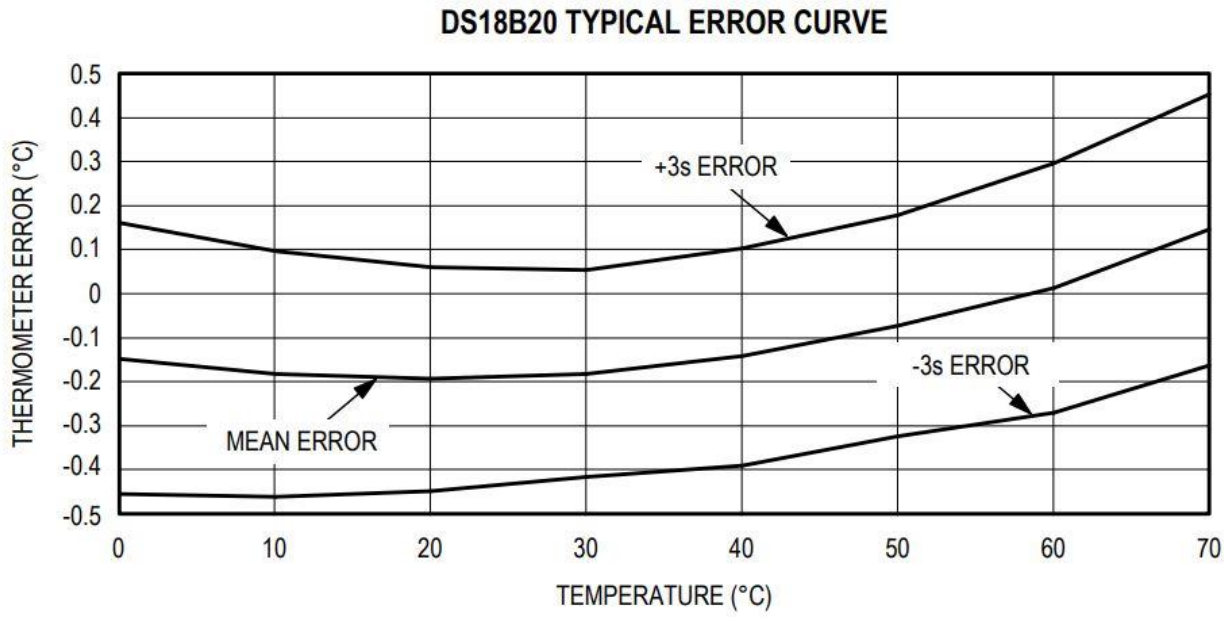


Figure 29. Typical error curve for DS18B20 digital thermometer. Provided by supplier.

Thermal time constant is a measure of how quickly the thermistor to responds to changes in ambient temperature. It is expressed as the time it takes for a thermistor to cool to 63.2% of the total difference between a stable high temperature and a stable low temperature. We

determined the thermal time constant for two different designs of DS18B20 sensors in stainless steel cladding: flat-topped and hemispherical. For sensors clad in a flat top housing, the time constant was 5.7 seconds. For those clad in a hemispherical housing, it was 8.5 seconds.

We also determined equilibration rate for DS18B20 sensors. While equilibration time is not a precise measurement, it is useful for making quick protocol adjustments in the field and gives a basic guideline for how quickly these thermometers react to temperature changes. For transitioning from low temperature to high temperature, the equilibration rate was 1.2 ^{o}C/s. For transitioning from high temperature to low temperature, the equilibration rate was 2.3 ^{o}C/s.

The DS18B20 temperature sensor is not factory calibrated and the user must perform a single-point or two-point calibration in order to achieve precision. The sensor offset is, however, linear, which means that a single calibration constant can be determined to correct the accuracy of the averaged measurement. A calibration constant can be established by plotting the average deviation between OpenCTD temperature sensors and an existing validated instrument or water of known temperature. The OpenCTD does not need to be submerged during this process.

Calibrated instruments should remain steady for many casts, but it is always good practice to check your temperature calibration whenever you suspect that the calibration may have changed. We recommend recalibrating temperature at the beginning of each expedition.

8.3 Conductivity (Salinity)

The OpenCTD uses a graphite conductivity probe manufactured by Atlas Scientific. Electrical conductivity probes are relatively simple devices consisting of two electrodes of known surface area and known distance from each other. As conductive fluid passes between the two electrodes, they measure the resistance of the liquid. The Atlas probe has a response time of 90% within 1 second, a 343-meter maximum operating depth, and a 1 to 110^{o}C temperature range. The K 1.0 probes can measure conductivity from 5 to 200,000 µS/cm.

Salinity is derived following the formulas outlined by the 1978 Practical Salinity Scale (Lewis, 1980). Temperature and pressure are integral parts of this calculation, and as such it is recommended that the user use this method for determining salinity instead of using the salinity value derived by the Atlas-Scientific conductivity circuit. The accuracy of the calculated salinity value ultimately depends on the accuracy of each individual sensor.

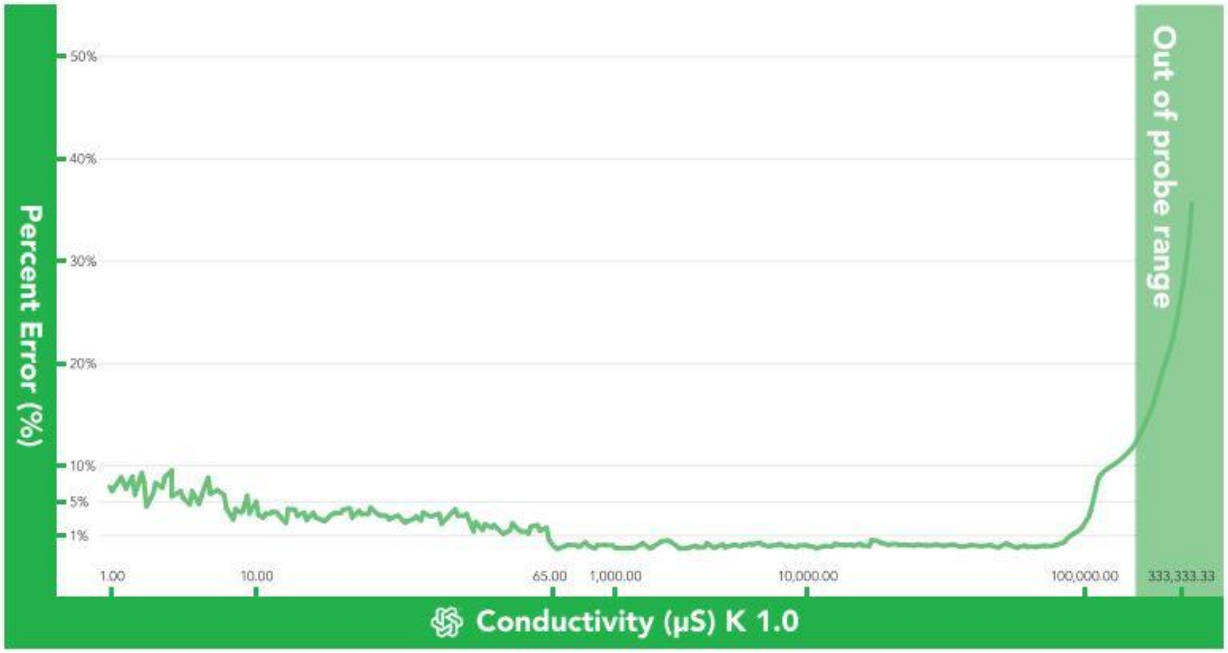


Figure 30. Accuracy of Atlas Scientific K 1.0 Conductivity Probe and EZO circuit. Data provided by Atlas Scientific.

The conductivity circuit requires a two-point calibration using solutions with a known and precise conductivity. Calibration solutions can be pre-mixed or ordered from a scientific supplier. Atlas provides a set of calibration standards for their probes.

Two-point calibration is facilitated using the Serial_for_EC_Calibration_m0.ino software found in the OpenCTD GitHub repository.

1. Flash Serial_for_EC_Calibration_m0.ino to the Adalogger.

2. Open the serial monitor through the Arduino IDE. You should see a stream of unprocessed data. Make sure the dropdown menus in the serial monitor are set to “Carriage return” and “9600 baud”.

3. Enter **c,0** in the command line and hit enter. This will turn off continuous monitoring.

4. Enter **k,?** in the command line and hit enter. This will tell you what the K-value of you probe is set to. It should be set to K=1.0 for most use cases.

5. To change the K-value, enter **k,x** (where x is the new K-value of the probe) in the command line and hit enter.

6. With the probe dry (be sure there is absolutely no water on the electrodes) enter **cal,dry** in the command line and hit enter. This will dry calibrate the probe.

7. For two-point calibration, suspend the probe in the less conductive solution and enter **cal,low,y** (where y is the known value of the standard) in the command line and hit enter.

8. Rinse the probe in fresh water and then suspend the probe in the more conductive solution. Enter **cal,high,z** (where z is the known value of the solution) in the command line and hit enter.

   Alternative: For 1-point calibration, suspend the probe in known calibration solution. Enter **cal,one,y** (where y is the known value of the standard) in the command line and hit enter. Only do this if you do not have two calibration standards.

9. Enter **c,1** in the command line and hit enter to turn continuous monitoring back on.

10. Reflash the OpenCTD software to the Adalogger after calibration is done.

Conductivity should not need to be calibrated again once the OpenCTD is complete, however it is good habit to check calibration at least once per year or whenever components are changed.

8.4 Data Management

The OpenCTD outputs data as a tab delimited text file which can be read by any standard spreadsheet program. It outputs the date and time followed by the pressure sensor in mbar, each individual temperature sensor in ^{o}C, and the conductivity sensor in micro Siemens. In order to covert those values into human-readable oceanographic data, we have produced an Excel template that takes the raw data and outputs depth in meters, average temperature, and practical salinity units. It will also average readings in batches of 60 to simplify data presentation. It has fields for calibration constants so that users can input the absolute pressure at sea level at the time of deployment and the temperature offset of the CTDs temperature probes determined experimentally during the calibration phase.

Although the SD card reader is built into the Adalogger, Arduino microcontrollers do not provide data passthrough, which means the SD card cannot be read directly from the Adalogger without flashing a secondary program to the microcontroller. Data is accessed post deployment by removing the SD card from the MCU and connecting it directly to a computer.

Please note that due to noise in the system, the conductivity circuit can create a bad data entry with additional, non-numerical characters. These are normal and can either be cleaned-up during data processing or discarded.

Importing into Excel: Open the Excel template and navigate to the Data Input sheet in the lower left corner. You can then either use the Get Data function in the Data menu to directly import the entire text file to the sheet or copy-paste the desired data points directly onto the sheet. If you use the Get Data function, Excel will create a new sheet for the data. Regardless of how you import the data, Excel will import it all as a single column. Highlight that column and then, under the Data tab, use the Text to Columns feature to split the data into individual columns by selecting Delimited, clicking Next, checking the boxes for Space and Treat consecutive delimiters as one, and clicking Finish.

The data is now in Excel-readable format and can be pasted into the Paste RAW data from OpenCTD columns on the Calculations sheet.

A Note about Macs: Depending on the version of MacOS, the default text editor may not be able to open the datafile produced by OpenCTD without modifying the editor's settings. We recommend downloading a robust, 3rd-party text editor is you plan to use a Mac to handle OpenCTD data.

8.5 Stewardship

Whenever you introduce a new tool into the marine environment, you run the risk of introducing additional harms to the ecosystem. Though OpenCTDs can serve as a valuable tool for ocean conservation, they also have the potential to negatively impact marine animals and ecosystems. We have developed a set of guidelines for minimizing potential negative impact when introducing new technologies into marine ecosystems (Thaler et al., 2019, 2015). Though these guidelines specifically address small underwater robots, they are equally applicable to OpenCTDs and other instrumentation platforms. We ask that you read the following two short papers and approach your project with an eye towards ensuring that your work will not contribute additional stress to the ecosystems that you are working in.

- Robots as Vectors for Marine Invasions: Best Practices for Minimizing Transmission of Invasive Species Via Observation-Class ROVs (https://journals.sagepub.com/doi/full/10.1177/194008291500800308)
- Bot Meets Whale: Best Practices for Mitigating Negative Interactions Between Marine Mammals and MicroROVs (https://www.frontiersin.org/articles/10.3389/fmars.2019.00506/full)

9.0 The First Deployment

The OpenCTD's maximum operating depth was established both through field trials and tests in a pressure chamber. Maximum operating depth of the sensor system is 140m, however, the housing has only been tested to 100 m depth without leaks or catastrophic failure. Unless you've outfitted it with an exceptionally large battery, a brand new OpenCTD will be positively buoyant. Additional weight can be added into the housing. Lead shot or fishing weights are particularly useful for this. If you are doing a line cast, you will need to secure the CTD to a sturdy line using hose clamps, waterproof tape, or other adhesive solutions.

Ensure that the inside surface of the PVC pipe is clean and free of debris. To ensure no leaks, the test cap used to seal the unit should be fully seated with no obstructions between the pipe wall and the rubber and be hand tight. Seat the test cap onto the housing and tighten the wingnut. The wingnut should be as tight as possible without tool assistance and the cap should not move or turn.

Your OpenCTD is now ready to take the plunge! Gently lower into the water and descend no more than 1 meter per second. The slower the better. For the first cast, descend to 5 meters without the MCU, hold for several minutes, then recover and check that no leakage has occurred. Once it passes the first soak test, you're ready to insert the MCU and collect ocean data.

While calibration plays a significant role in determining that data quality is reasonable, response times and the method of deployment also have an influence. The DS18B20 clad in stainless steel has a time constant of 5.7s. The conductivity sensor maintains a response time of approximately one second for a 90% value (Atlas-Scientific, 2017). The pressure sensor response time is variable between 0.5 to 8.22 m/s depending on the resolution selected by the user (Measurement Specialties, 2012). In order to ensure that the device has reached relative equilibrium with the surrounding water, it is necessary to leave it at station for several minutes before beginning an in-situ deployment. When faster response time is required, the sensors used in the most recent version of the OpenCTD can easily be replaced by more accurate, yet often more expensive alternatives.

Welcome to the Oceanography for Everyone community!

9.0 Works Cited

Fofonoff, N.P., Millard Jr, R.C., 1983. Algorithms for the computation of fundamental properties of seawater.

Hooker, S.K., Boyd, I.L., 2003. Salinity sensors on seals: use of marine predators to carry CTD data loggers. Deep Sea Research Part I: Oceanographic Research Papers 50, 927–939. https://doi.org/10.1016/S0967-0637(03)00055-4

Lewis, E., 1980. The practical salinity scale 1978 and its antecedents. IEEE Journal of Oceanic Engineering 5, 3–8. https://doi.org/10.1109/JOE.1980.1145448

Stammer, D., Balmaseda, M., Heimbach, P., Köhl, A., Weaver, A., 2016. Ocean Data Assimilation in Support of Climate Applications: Status and Perspectives. Annual Review of Marine Science 8, 491–518. https://doi.org/10.1146/annurev-marine-122414-034113

Stephen, R.C., Howard, F.J., Dean, R., Susan, W., Ariel, T., Mathieu, B., Denis, G., Jianping, X., Sylvie, P., Thresher, A., Traon Pierre-Yves, L., Guillaume, M., Birgit, K., Kjell-Arne, M., Velez-Belch Pedro, J., Isabelle, A., Brian, K., Jon, T., Molly, B., Steven, J.R., 2016. Fifteen years of ocean observations with the global Argo array. Nature Climate Change Marine Res Inst 6. https://doi.org/10.1038/NCLIMATE2872

Sverdrup, H.U., Johnson, M.W., Fleming, R.H., 1942. The Oceans: Their Physics, Chemistry, and General Biology. Asia Publishing House.

Thaler, A., Parsons, E.C.M., de Vos, A., Rose, N.A., Smith, C., Fretz, D., 2019. Bot Meets Whale: Best Practices for Mitigating Negative Interactions Between Marine Mammals and MicroROVs. Front. Mar. Sci. 6. https://doi.org/10.3389/fmars.2019.00506

Thaler, A.D., Freitag, A., Bergman, E., Fretz, D., Saleu, W., 2015. Robots as vectors for marine invasions: Best practices for minimizing transmission of invasive species via observation-class ROVs. Tropical Conservation Science 8.

Thomson, R.E., Emery, W.J., 2014. Data Analysis Methods in Physical Oceanography. Newnes.

10.0 Resources and Datasheets

General Guides

Soldering is Easy: Here's How To Do It: mightyohm.com/files/soldercomic/FullSolderComic_EN.pdf

Environmental Monitoring with Arduino: Building Simple Devices to Collect Data About the World Around Us: https://amzn.to/2M17yUi

Pressure Sensor

MS5803-14BA Miniature 14 bar Module: https://cdn.sparkfun.com/datasheets/Sensors/Weather/ms5803_14ba.pdf

Temperature Sensor

DS18B20 1-Wire Digital Thermometer: https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf

Conductivity Sensor

EZO-EC Conductivity Circuit Datasheet – https://www.atlas-scientific.com/_files/_datasheets/_circuit/EC_EZO_Datasheet.pdf

EC K 1.0 Conductivity Probe Datasheet – https://www.atlas-scientific.com/_files/_datasheets/_probe/EC_K_1.0_probe.pdf

Conductivity Accuracy Graph – https://www.atlas-scientific.com/_files/instructions/conductivity_accuracy_graph.pdf