blue_eye

A Low Cost Video System for Deep-Sea Exploration

Assembly Guide 1.0

November 2022

Pau Anta, www.pauanta.bio
pauanta@icloud.com
https://github.com/PauAntaBio/deep-sea-drop-cam

Index

1. Introduction

Welcome aboard! This document describes how to build and operate a low-cost video system capable of descending up to 500 meters in the sea, ideal for exploring epipelagic and mesopelagic zones. The device, which is also known as a deep-sea dropcam, is called the blue_eye.

Why did I develop this device? In November 2018, my father took me with him to Boston to attend All Hands on Deck, a fantastic event co-organized by the MIT Media Lab and NOAA Office of Ocean Exploration and Research, in which policy makers, scientists and activists met to imagine new ways to empower an open and inclusive global community of ocean explorers.

During one of the event's presentations, I learned that 70 percent of countries in the world have deep-sea environments but only 16 percent of them can explore such environments. The scarcity of technological capability and knowledge leads to a lack of exploration, inappropriate or inadequate management decisions, and unaware populations, and this scarcity happens mostly in developing countries and small island states (source: MIT Media Lab My Deep Sea, My Backyard project).

At the same time, during the exhibitions, I was very impressed by the deep-sea dropcam developed by the Exploration Technology Lab at National Geographic, which had been deployed in the Marianas Trench at 10,641 meters of depth, and had an estimated cost of U$10,000. In that contagious atmosphere of willingness to contribute somehow to ocean exploration I decided to build a low-cost deep-sea dropcam and make it open source, so others could build their own dropcams for a reasonable amount of money. I was also inspired by my love for the Mediterranean, a beautiful sea threatened by overfishing, pollution, and mass tourism.

My original goal was to develop an Arduino-controlled dropcam capable of submerging to 500 meters at a maximum cost of US$500. I started to design and develop the blue_eye in the last days of 2018 and tested it during the summer of 2019 in Menorca at 60 meters of depth. Everything worked well except for the burnwire, and I was unable to further test in deeper waters due to a lack of access to the necessary depths. I put this project on hold during 2020 and 2021 to pursue other projects. In the Spring of 2022, I went back to work on this project, aiming to improve several features. I solved the burnwire mechanism (now, it works!), but I was still unable to test it at 500 meters. The cost of the system presented in this document is U$1,400, far more expensive than the original goal, but I still consider it to be a great low cost device for ocean exploration that could be even cheaper if certain non-critical elements are excluded.

Special thanks to Dr. Brennan Phillips, Assistant Professor of Ocean Engineering at the University of Rhode Island, for sharing his knowledge, passion, and enthusiasm with me. His advice was key for building the blue_eye (he knows everything, but if he doesn't know, he knows who does).

I want to dedicate this project to Dr. Katy Croff Bell, former director of the Open Ocean Initiative at the MIT Media Lab, and now founder and president of the Ocean Discovery League, and to the whole team behind the My Deep Sea, My Backyard project.

And a big thank you to my dad for supporting me all along this project and helping me in the development of this assembly guide.

Together with this assembly guide, you will find other key materials in my GitHub repository under github.com/PauAntaBio/deep-sea-drop-cam:

- Bill of materials - BOM.xlsx
- Code – code/blue_eye_v10
- 3D parts

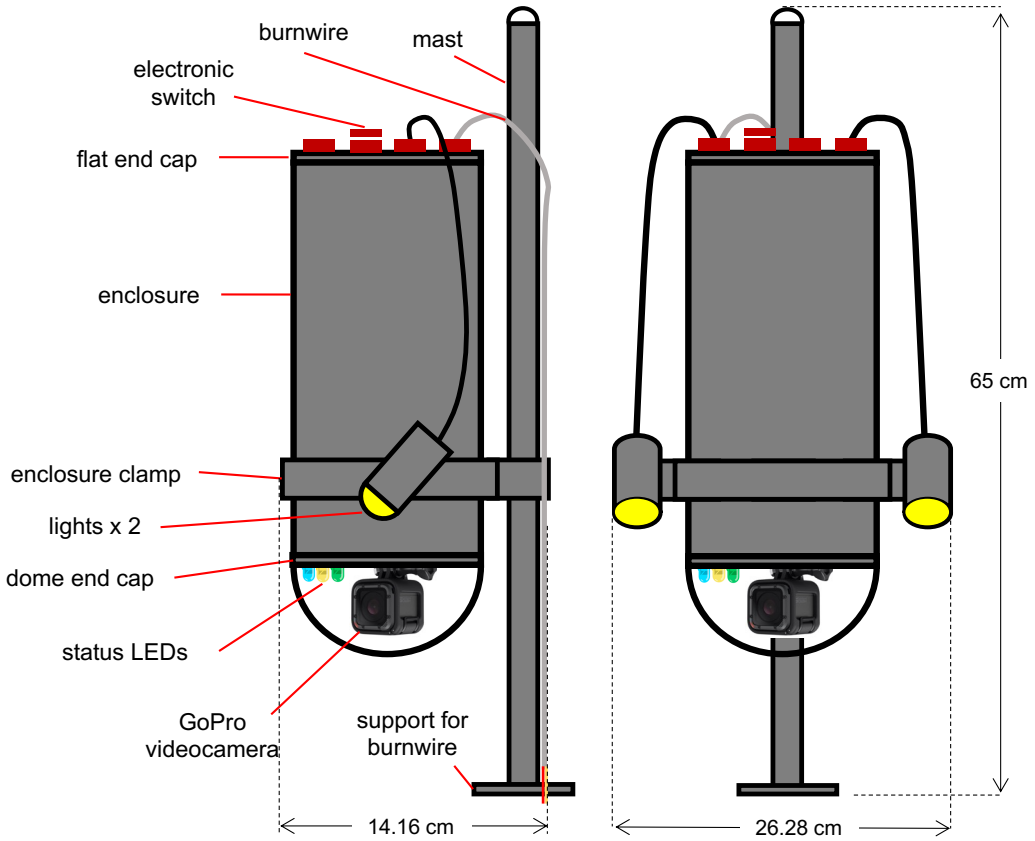
In this link https://youtu.be/udxaxlVZxtM, you can see a short video of the process of building and testing the blue_eye.

2. Technical Architecture

The blue_eye is a programmable video system built inside a water tight enclosure attached to a mast, that records 4K video with a GoPro HERO 5 Session, controlled by an Arduino MKR WiFi 1010. The main characteristics are of this system are:

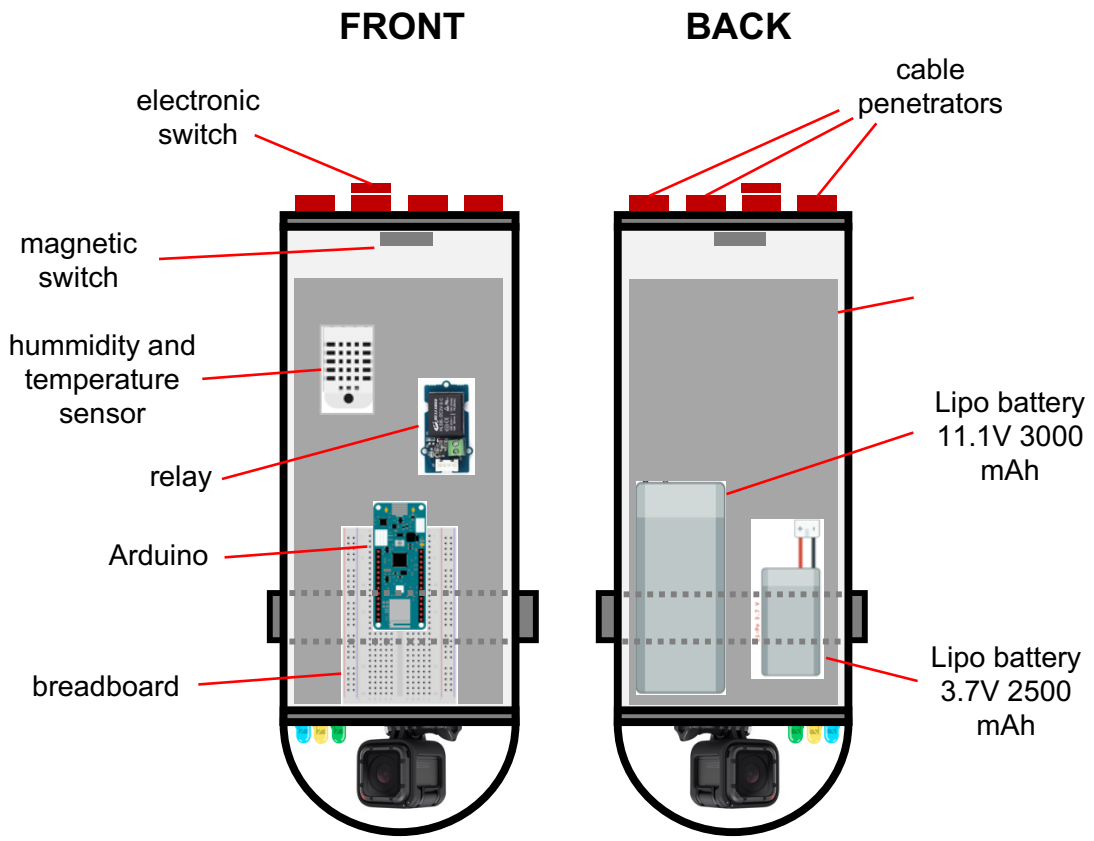
- Two Li-Po batteries: a 3.7V for the motherboard and a 11.1V for lights and burnwire.
- Electronic switch for turning on/off the motherboard.
- Magnetic switch for activating the exploration mission (activates the control of the GoPro).
- LED-based interface that signal the status of the device: power on/off (blue), Arduino connected to GoPro (green) and exploration activated (yellow).
- Two subsea LED lights with 1500 lumens each.
- Internal humidity and temperature sensor (optional).
- Primary release mechanism based on a burnwire that produces electrolytic erosion to cut the line with the sandbag ballast.
- Secondary release mechanism based on a galvanic time release (GTR), as a back up in case the burnwire doesn't work as expected.
- A carbon mast to attach the elements that keep the device vertical (a float and a weight), the burnwire, the sandbag ballast and a bait bag.

In the following schematic you can see some of the main components of the blue_eye:



The enclosure is built out of parts used for manufacturing the underwater remotely operated vehicle BlueROV2, supplied by the company Blue Robotics.

In the schematic below you can see the disposition of the different elements inside the device (didn't include wires, because it would be a mess), separated in half by an "electronic tray", a flat support in which you can install all the electronics:

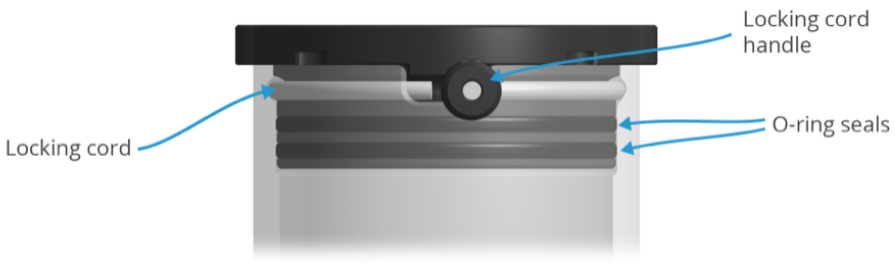


One of the main technical decisions during the design was the selection of the video camera. I decided to use a GoPro HERO 5 Session because it records high-definition video, it can be controlled wirelessly by an Arduino, and because I already had it and wanted to keep my budget as low as possible. If you don't have a GoPro, you can explore other options that could be controlled by an Arduino or similar microcontroller.

Up until recently, using the GoPro for the purpose of the blue_eye had a drawback: in the preparation process to drop it overboard, you needed to open the enclosure to get access to the GoPro and be able to turn it on manually, and this was not ideal because closing the device is always a critical step as you must make sure it is sealed and watertight.

However, in early 2022, Blue Robotics launched a new generation of parts (O-ring flange and aluminum tube) with a locking cord handle that makes the open/close of the enclosure much

easier (see image below). So, if you are going to build a blue_eye, you will use the new generation of parts that will allow you to open and close the device easily and safely.



Tube and flange with the locking cord installed (source: Blue Robotics)

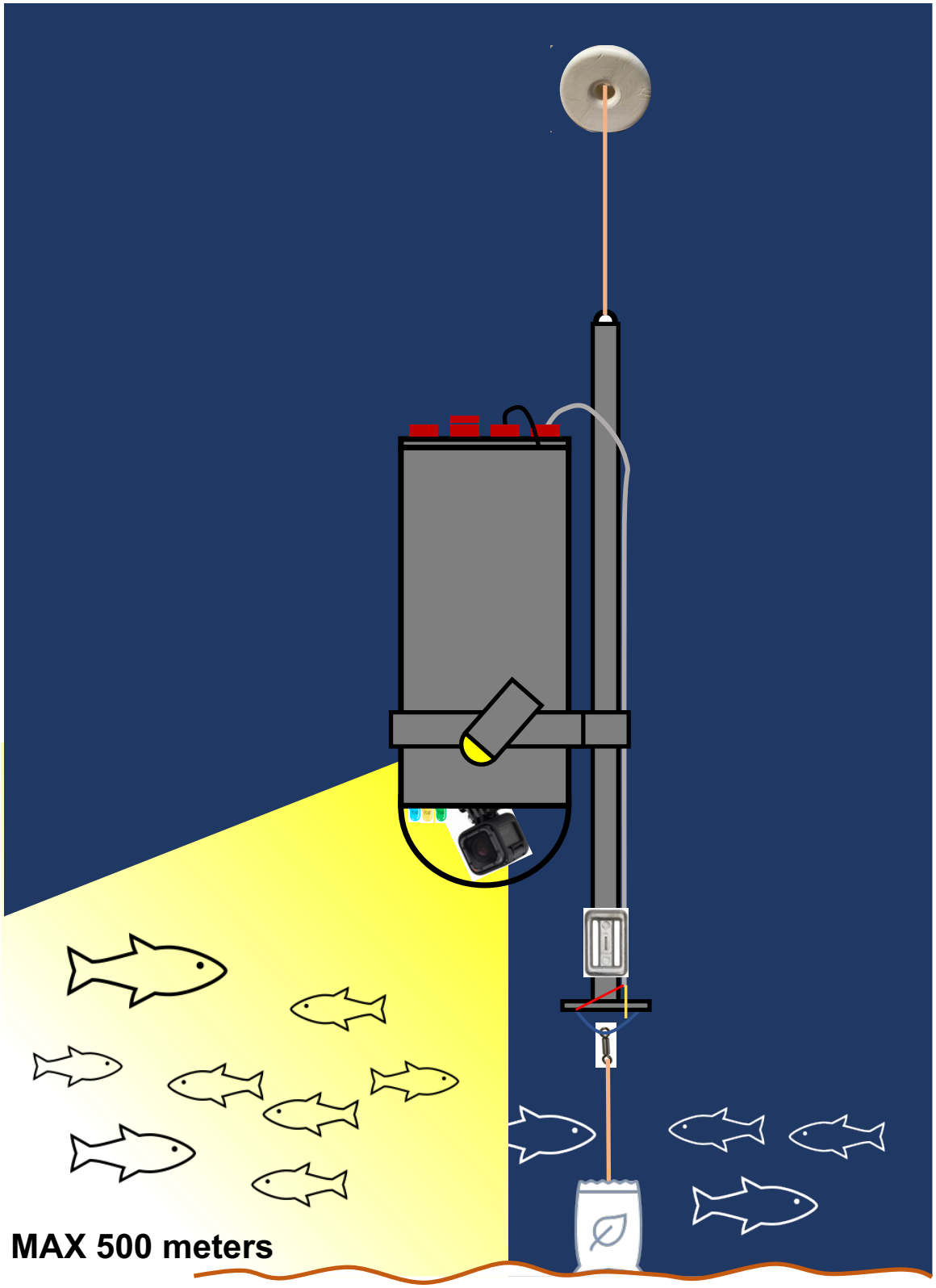
With the Arduino inside the blue_eye, you can control the following:

- Timers (drop time, recording time)
- Start/stop video recording
- Turn on/off the lights
- Control lights brightness
- Activate burnwire

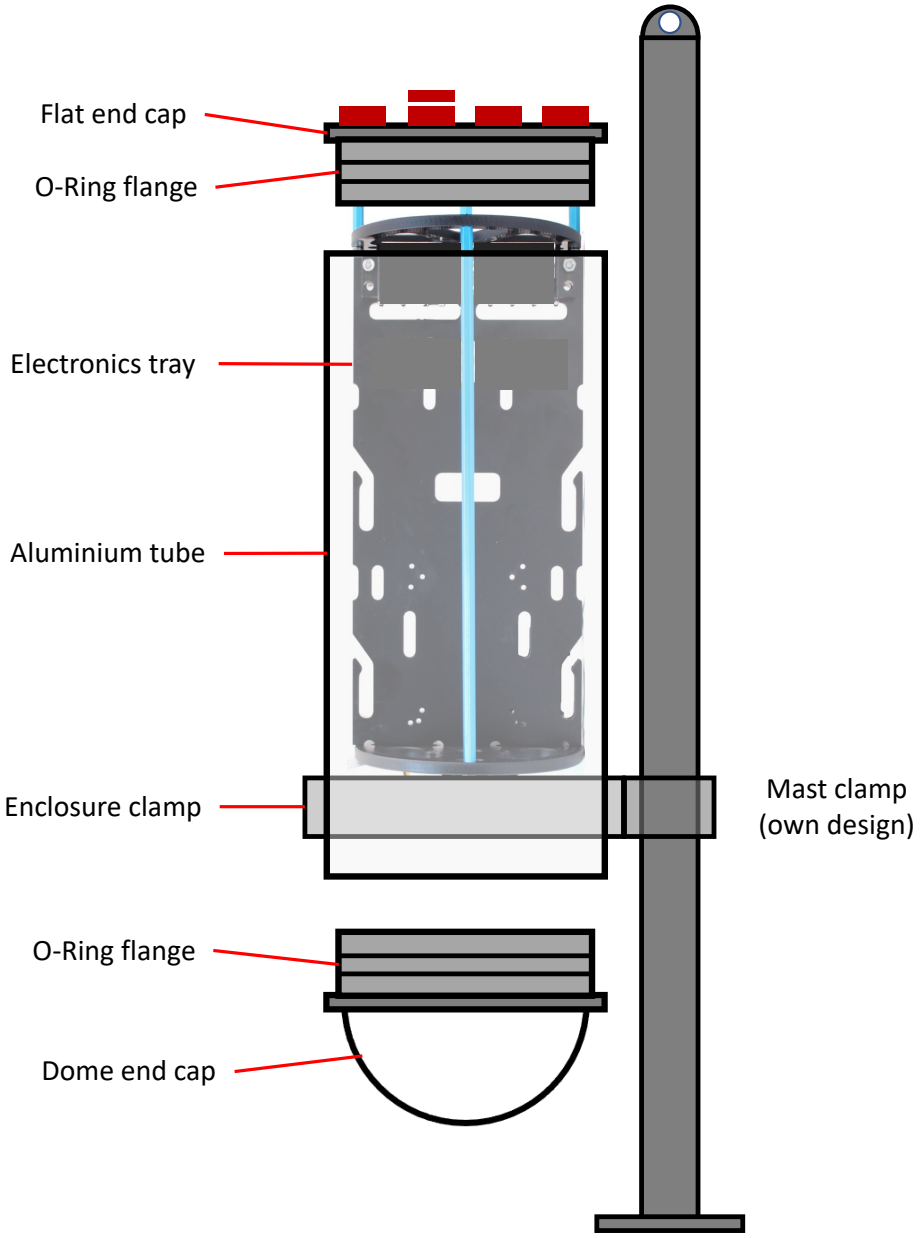
Main specs:

- Size: 65 cm x 26.28 cm x 14.16 cm
- Rated for 500 m (epipelagic and mesopelagic zones)
- Weight in air: 2.85 kg
- Cost: ~ $1,400 USD

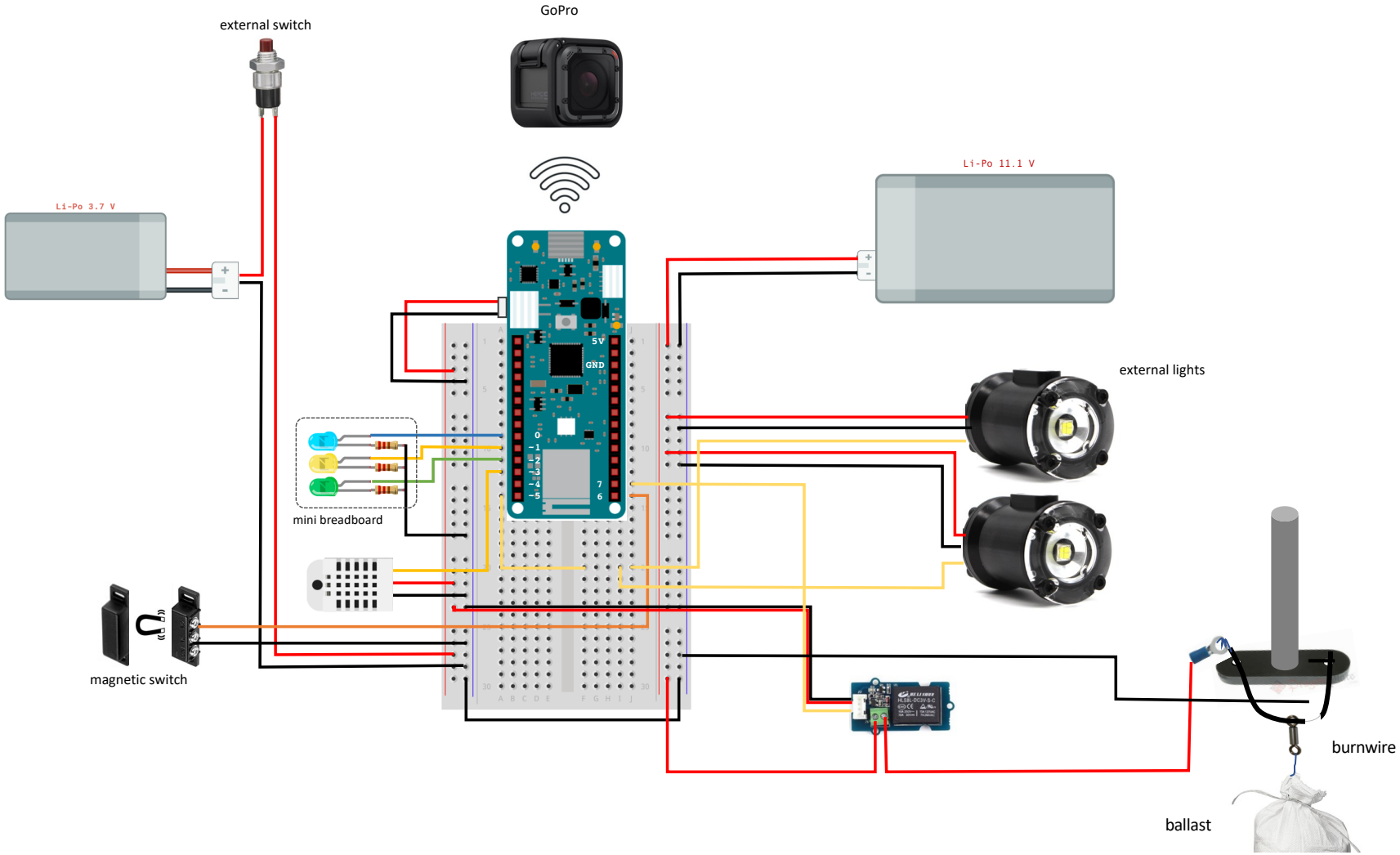
The schematic below represents the blue_eye deployed with all its components:



The schematics below shows the different elements of the physical structure of the blue_eye:



The following schematics shows all the electronic elements and the wiring of the motherboard:



3. How to build it

3.1 Motherboard

Parts

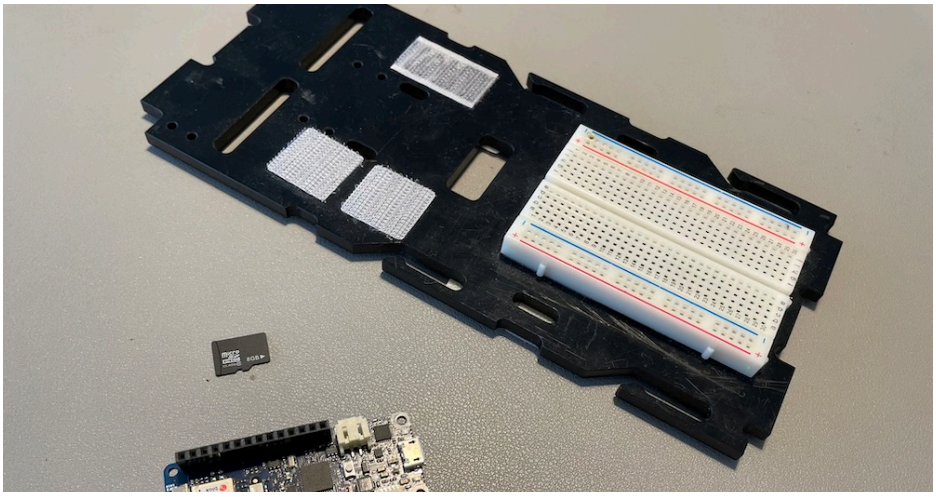
1 Electronics tray 4”
1 Half sized breadboard
1 PCB breadboard
1 Digital temperature and humidity sensor
3 LEDs (green, yellow, blue)
3 Resistors 220Ω
1 Microcontroller Arduino MKR 1010 WiFi
1 Memory Shield
1 microSD card
1 Relay
1 Magnetic switch
1 Electronic switch
1 Battery 11.1V 3000mAh (lights and burn-wire)
1 T-plug connector
1 Battery 3.7V 2500mAh (brains)
1 Connector cable
2 Ring terminals
1 flat end cap
6 M3-12mm screws
1 T-Plug to Male Adapter
1 O-Ring flange
Solid wires (red, green, yellow, blue, and black)
Lithium Ion Polymer Battery 3.7v 2500mAh
3S Lipo Battery 11.1V 30C 3000mAh

Tools

Adjustable helping hands
Wire strippers
Terminal crimper tool
Solder and solder wire
Velcro strip
Scissors

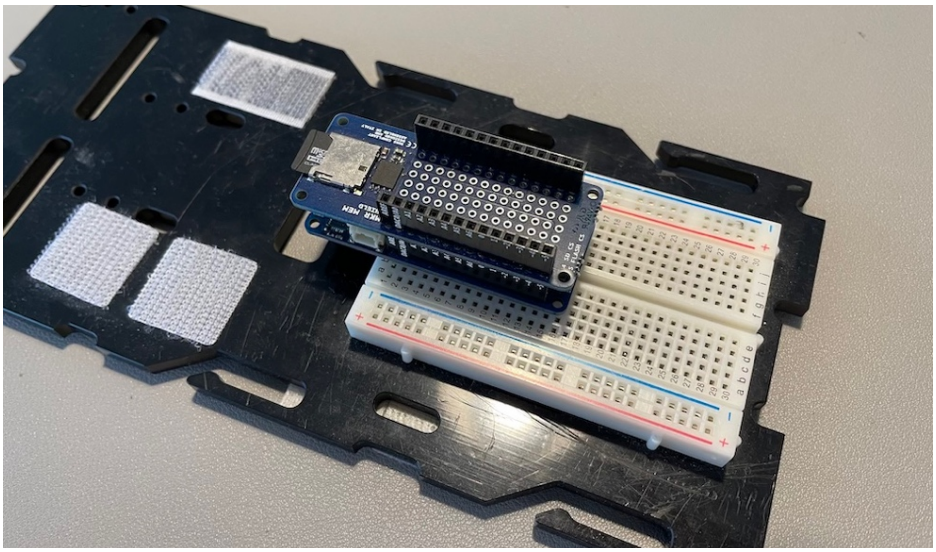
Instructions

1. **Electronics tray**. All the electronics inside the blue_eye are attached to a flat panel. Remove the protective paper from the back of the half sized breadboard and stick it on the electronics tray, bottom-centered, as shown in the picture. I suggest you to read this tutorial to get familiar about the installation of the tray: https://bluerobotics.com/learn/4-series-electronics-tray-assembly/

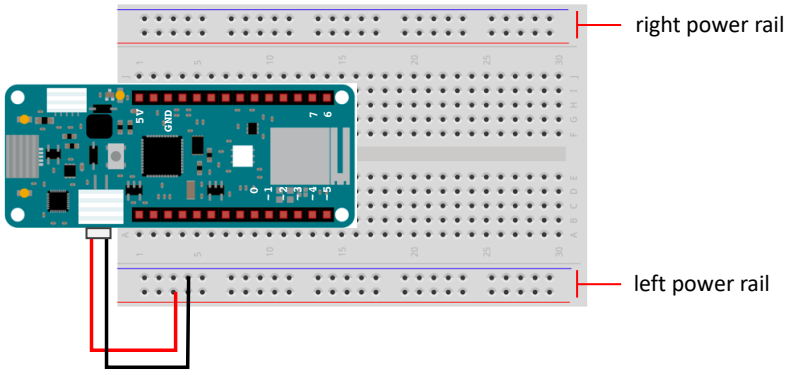


2. **Microcontroller.** Insert the Arduino MKR 1010 in the breadboard, with pin AREF (tie point 1-b in the breadboard).

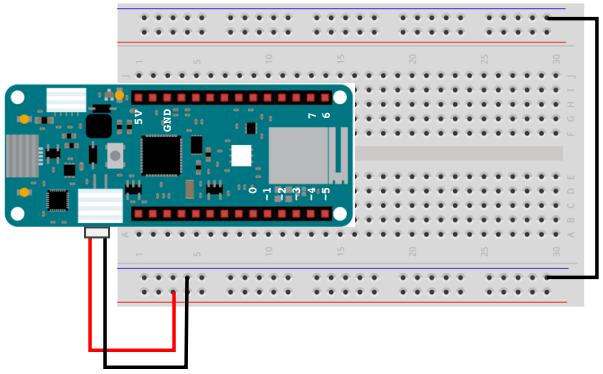
   OPTIONAL: if you want to keep a log of the activity of the blue_eye, then insert a microSD card inside the memory shield, and add the memory shield on top of the Arduino. I use this feature and it was critical during the development and testing.

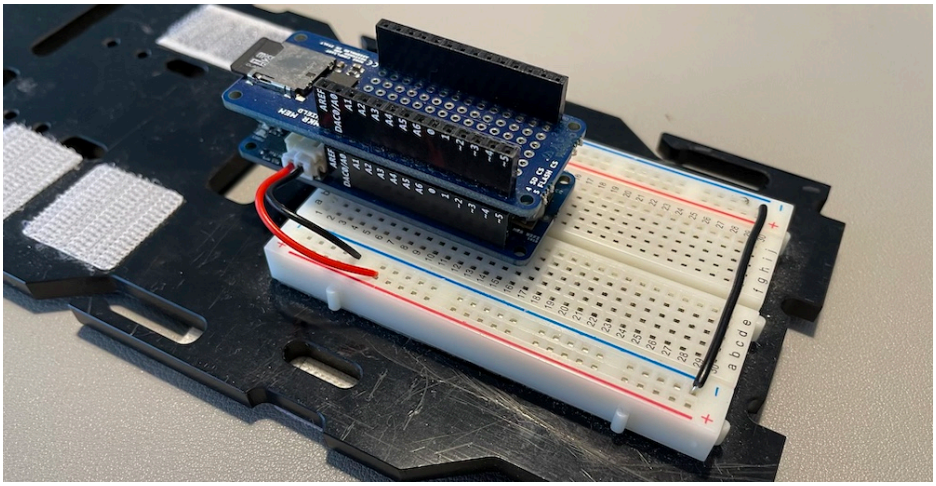


a. Plug a connector cable inside the battery charging port of the Arduino. Now, insert the red wire of the connector cable in any tie of the + row and the black wire in any tie of the – row (ground) of the left power rail of the breadboard.



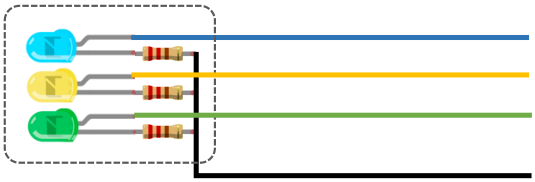
b. As you can see, the breadboard has two power rails: the left one will be powered by a 3.7V Li-Po battery and the right one by a 11.1V Li-Po battery, and you need to connect the ground of both power rails. To do that, cut a few centimeters of black wire, peel 4mm of each end, and use it to connect the – rows of the two power rails.





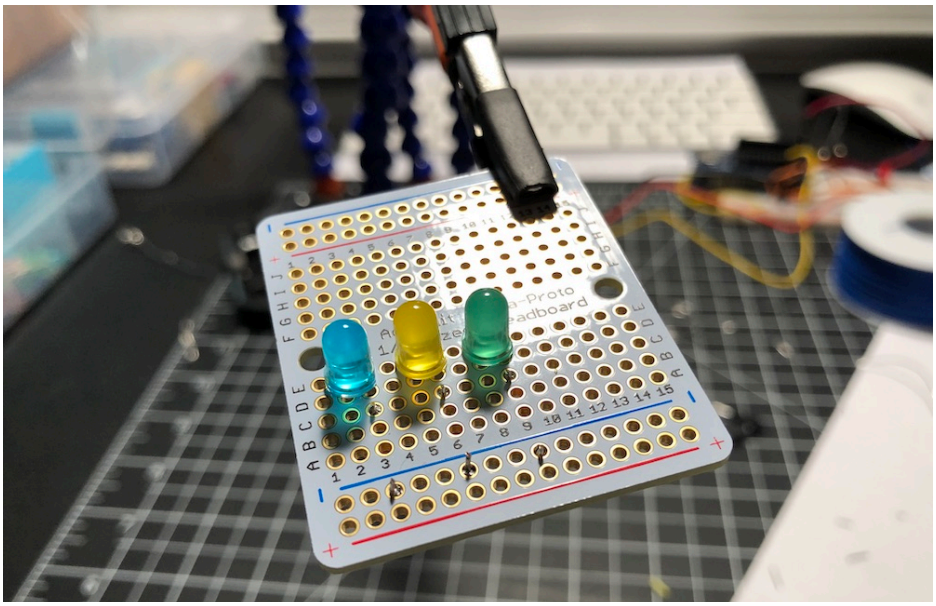
3. **LED status indicators**. The blue_eye uses three LEDs as status indicators, which are the actual user interface for the operation the device:
   - A solid blue LED indicates that the motherboard of the blue_eye is on.
   - A blinking green LED indicates that the Arduino is trying to connect to the GoPro's WiFi network and a solid green LED indicates that the Arduino is connected to the GoPro.
   - A solid yellow LED signals that the magnetic switch has been activated.

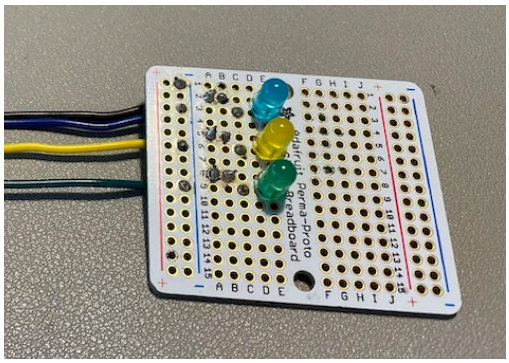
   The schematics below shows the circuit for the LEDs.

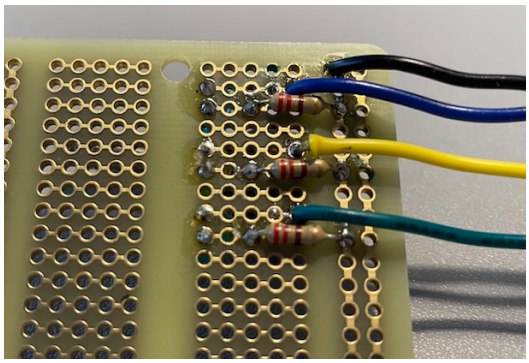


   Now, cut a few centimeters of wires: blue, yellow, green and black, to match the colours of the LEDs and ground, and peel the ends of each wire.

   a. Insert the anode and cathode of the LEDs into the PCB breadboard and solder them from the bottom. You can cut excess of the anodes and cathodes.
   b. Solder a wire of the same color to the anode of each LED (blue, yellow and green).
   c. Solder a 220Ω resistor between the cathode of each LEDs and the ground row of the PCB breadboard. Cut excess of the connecting leads of the resistors.
   d. Solder a black wire to the ground row of the power rail of the PCB breadboard.

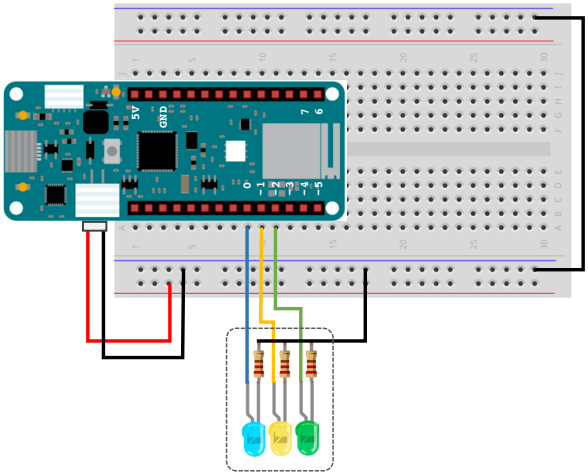




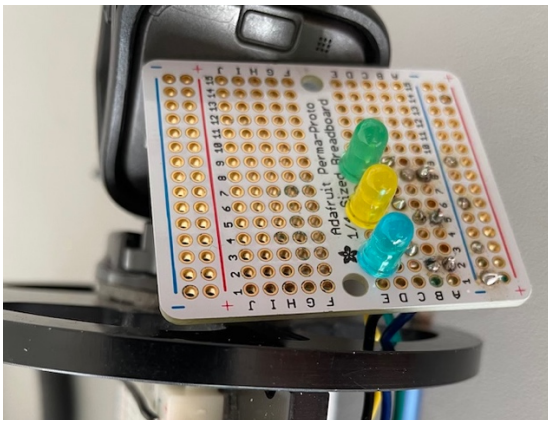


Now, you connect the LED wires to the Arduino, as follows:

e. Blue wire (blue LED) in point 9-A, connected to pin 0.
f. Yellow wire (yellow LED) in 10-A, connected to pin 1.
g. Green wire (green LED) in 11-A, connected to pin 2.
h. Ground wire to ground row of left power rail.

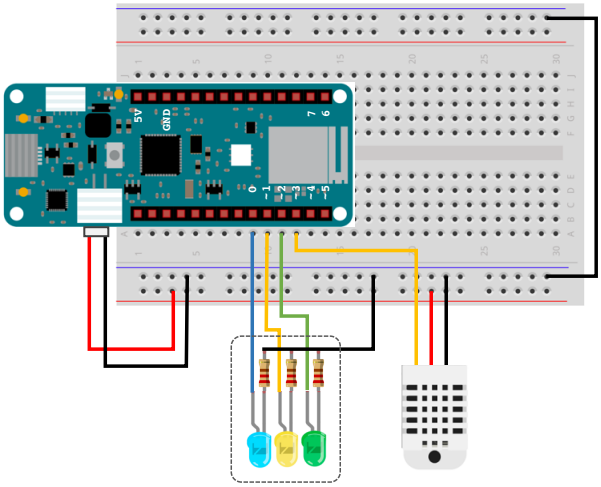


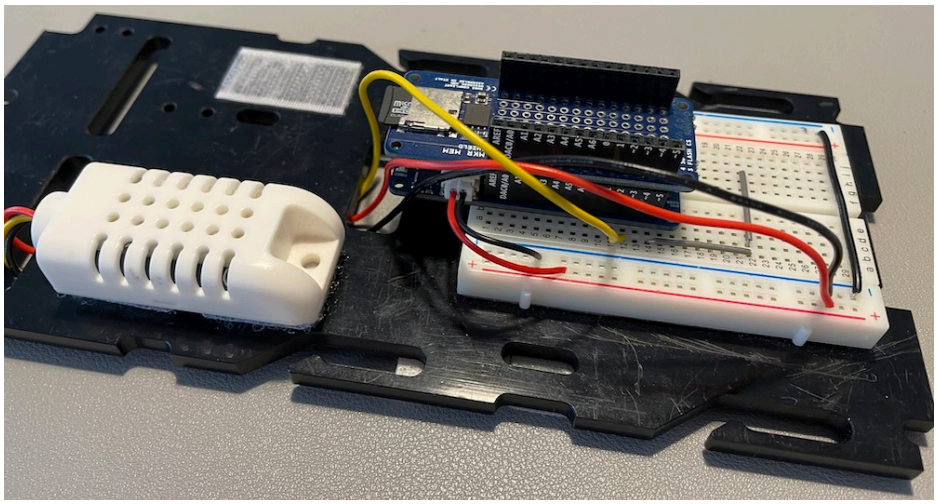
The PCB breadboard with the LEDs will be installed next to or behind the GoPro, in a place where you can see the status when the enclosure is closed and you are about to drop it (see pictures below).



4. **Temperature and humidity sensor (OPTIONAL)**: the installation of a digital temperature and humidity (DHT) sensor inside the blue_eye is optional. It can provide very interesting data that you can store in a log, and you can also use the humidity sensor to estimate inlets of water inside the enclosure and automate the activation of the burnwire (the pseudocode would be something like: if humidity is higher than 80%, then activate burnwire). Cut 3cm of velcro strip, attach one side to the back of the DHT sensor and the other side to the electronics tray. Now you can install the DHT sensor on the electronics tray and connect the DHT sensor to the Arduino as follows:

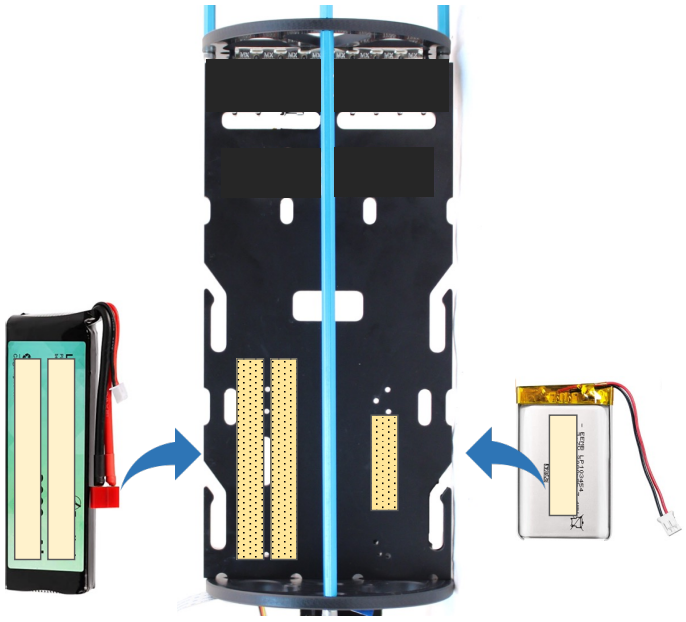
    a. Yellow wire to tie point 12-A, connected to pin 3.
    b. Red wire to + row of left power rail.
    c. Black wire to ground row of left power rail.



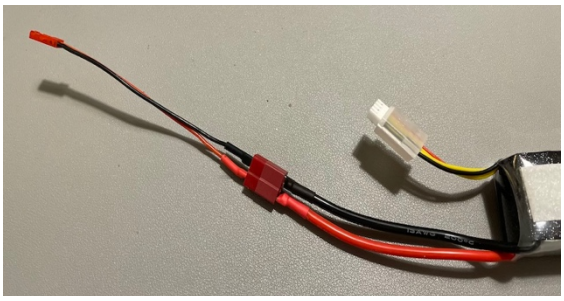


5. **Batteries**. As it was described in chapter 2, the blue_eye has two Li-Po batteries: a 3.7V 2500mAh to power the motherboard and an 11.1V 3000mAh for the lights and burnwire.

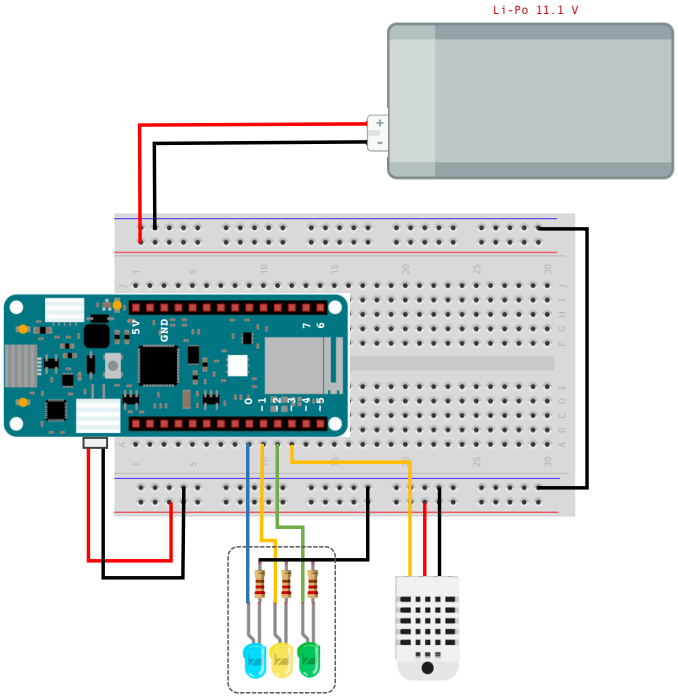
   a. Cut a 4cm Velcro strip. Attach one side to the 3.7V battery and the other side to the electronic tray, in the opposite side where you have the breadboard (remember figure in page 7).
   b. Attach the 3.7V battery to the electronics tray by the Velcro.
   c. Cut two Velcro strips of 10cm each. Attach one side of each strip to the 11.1V battery and the other two sides to the electronic tray.
   d. Attach the 11.1V battery to the electronics tray by the Velcro.



   e. Attach the T-plug male connector to the female power cable of the 11.1V battery.



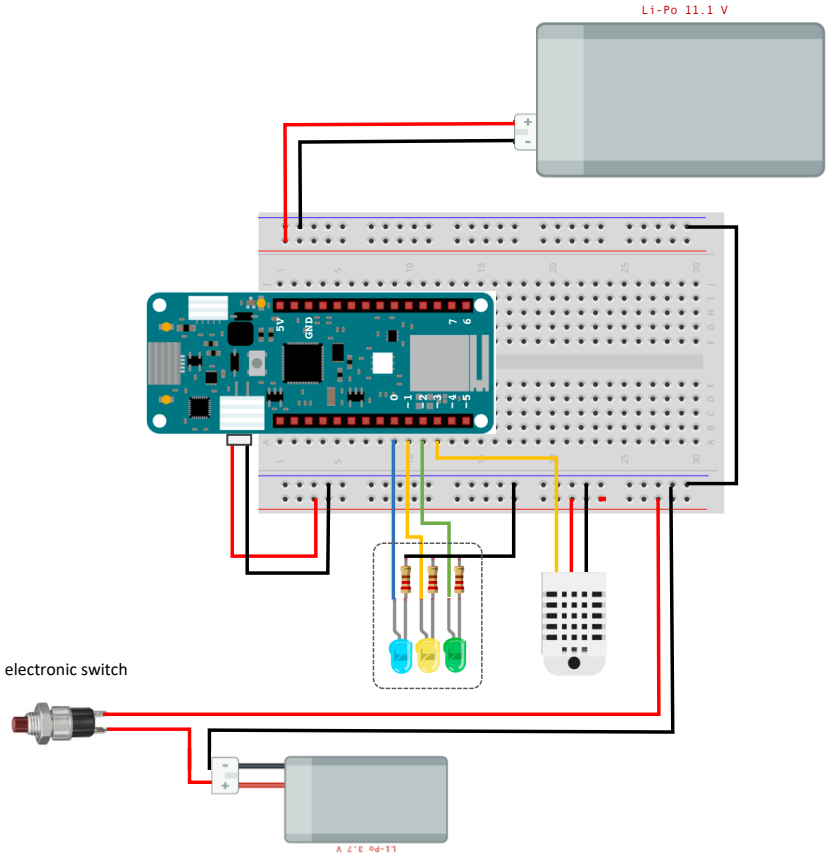
f. Cut two wires of 12 cm each, one red and one black.
g. Connect one end of the red wire to the red tie of the power cable of the 11.V battery, and the other end to the + row of the right power rail of the breadboard.
h. Connect one end of the black wire to the black tie of the power cable of the 11.V battery, and the other end to the - row of the right power rail of the breadboard.



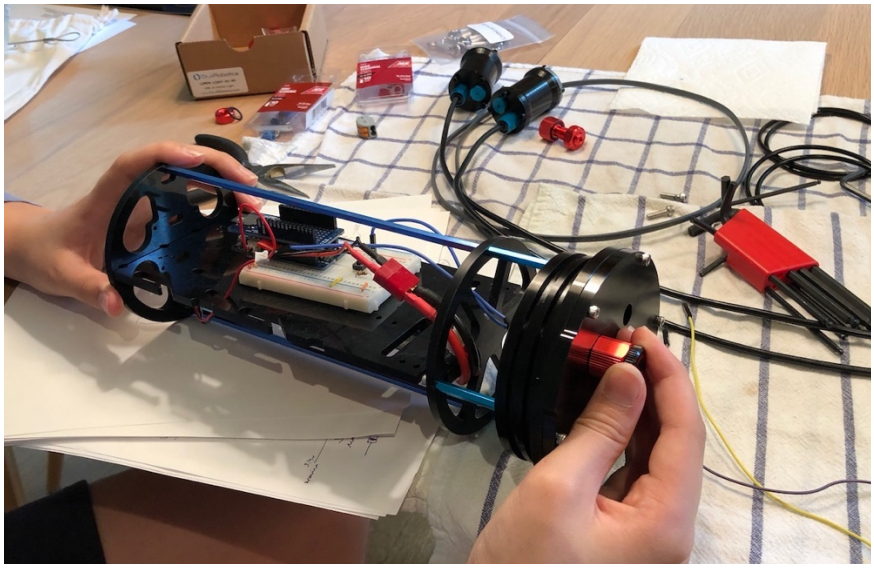
6. **Electronic switch**. This is the switch that you will use to turn on/off the motherboard of the blue_eye, powered by the 3.7V battery. This switch has two pins, and it comes with two blue wires (pictures from Blue Robotics).

a. Assemble the different elements of the electronic switch, following the instructions provided by Blue Robotics.

b. Install the flat end cap on top of one O-Ringe flange, using six socket head cap screws, following the tutorial provided by Blue Robotics: https://bluerobotics.com/learn/watertight-enclosure-assembly-guide/
c. Install the switch bulkhead and plug in any of the holes of the flat end cap.
d. Connect the two wires to the pins of the 30-3UL button.
e. Connect one of the wires to the positive tie (red) of the 3.7V battery.
f. Connect the other wire to the + row of the left power rail of the breadboard. With these two connections, when you turn the switch on, you allow the flow of electricity from the battery to the breadboard. In the schematics below, both wires are red (to make it easier to remember that they carry electricity), but they are actually blue.
g. Cut 20 cm of black wire. Connect one end to the ground tie of the 3.7V Li-Po battery and the other end to the ground row of the left power rail of the breadboard.

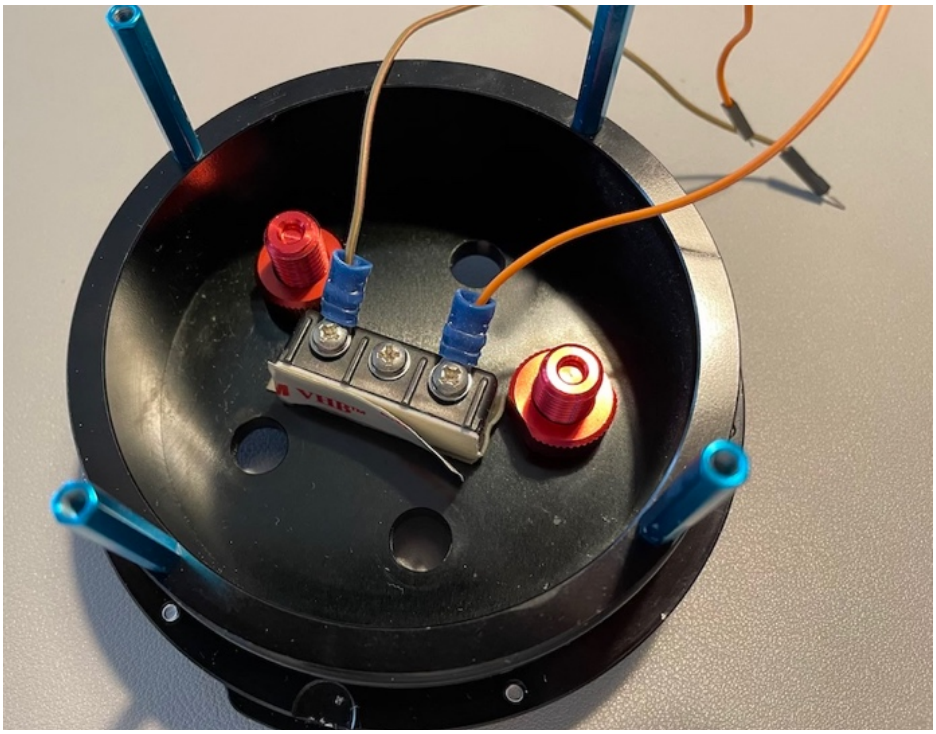


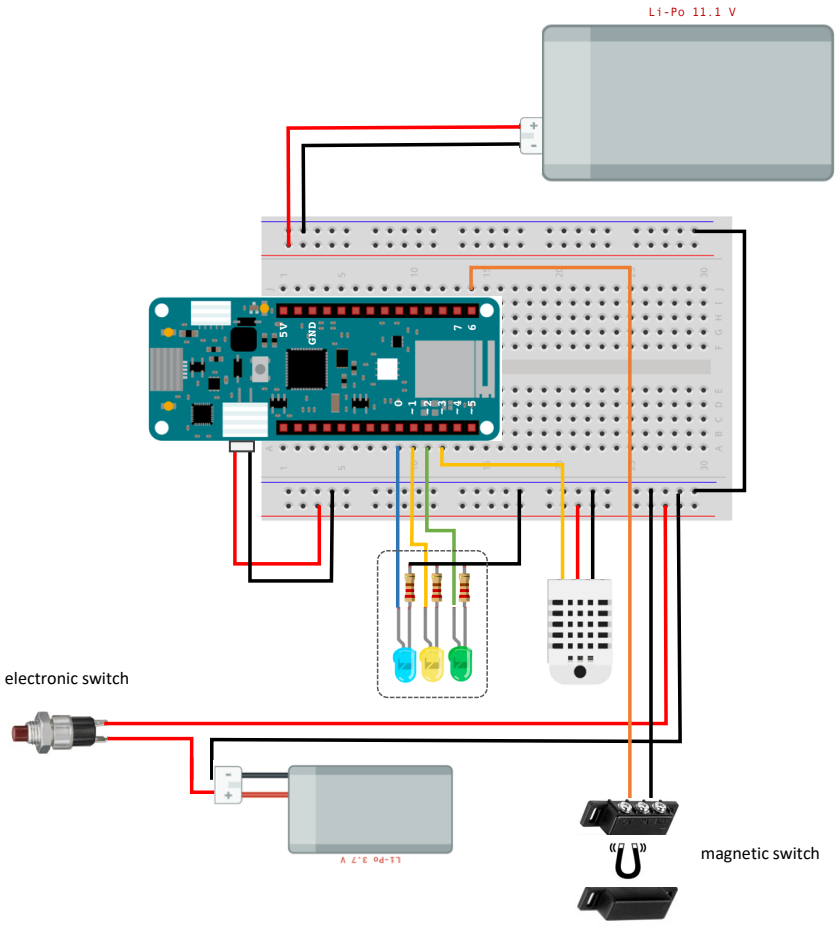
Now, you may want to test that the electronic switch works properly and powers the Arduino with the 3.7V battery.



7. **Magnetic switch**. This is the switch that allows you to start the exploration mission of the blue_eye. When you turn it on, you are telling the Arduino that you are dropping the device overboard and it starts the timer of the mission. The magnetic switch, as its name suggests, has two parts: one is the switch and the other is a magnet (make sure you always keep it on you when you are going to operate the blue_eye).

    a. Cut two wires of 20cm each: one black for ground, and any color for the other one.
    b. Install a ring terminal to one end of each wire, using the terminal crimper tool.
    c. Install the ring terminal of the ground wire in the NO terminal of the switch, and the end of the wire in the ground row of the left power rail of the breadboard.
    d. Install the ring terminal of the other wire in the COM terminal and the end of the wire in 14-j, connected to pin 6 of the Arduino.
    e. Attach the switch in the internal part of the flat end cap, using VHB tape. This part has its own VHB tape and you could use it, but I preferred to attach it in a way that I can access the screws in case I need to change anything.




Li-Po 11.1 V
electronic switch
magnetic switch

3.2 Lights

Parts

2 Lumen Subsea Light 1500 lumens
2 Cable penetrators
1 Cable penetrator blank

Materials

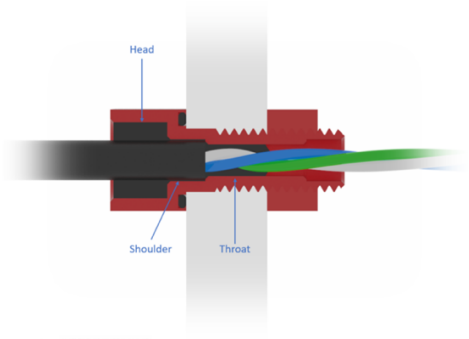
Potting Compound: WetLink Thixotropic 80A
Potting kit
Acetone cable preparation wipes

Tools

Potting dispenser
Adjustable helping hands
Clamp on vise
Hex key set
Wire strippers

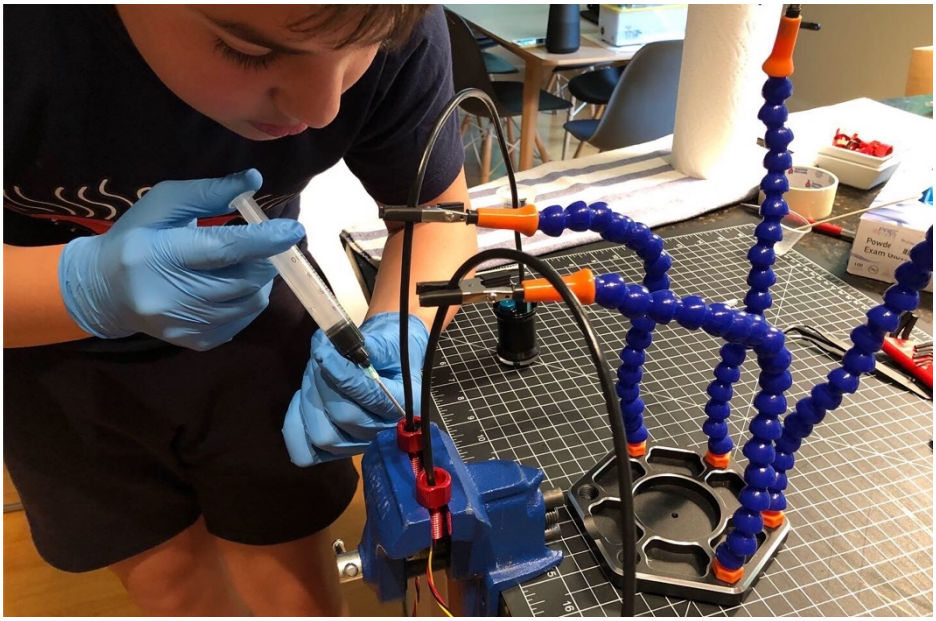
Instructions

1. **Potted cable penetrators**. The flat end cap of the blue_eye has five holes, which allow for five elements on it (cables or buttons). The blue_eye needs four holes (two for the lights, one for the burnwire and one for the electronic switch) and you will use the fifth for a vacuum test or future needs (for example, you might want to add a third light, a temperature sensor, or an electrical conductivity sensor). For connecting the lights and the burnwire to the motherboard, we use potted cable penetrators, which offer a waterproof, high-pressure seal to pass cables into a watertight enclosure (picture below from Blue Robotics).

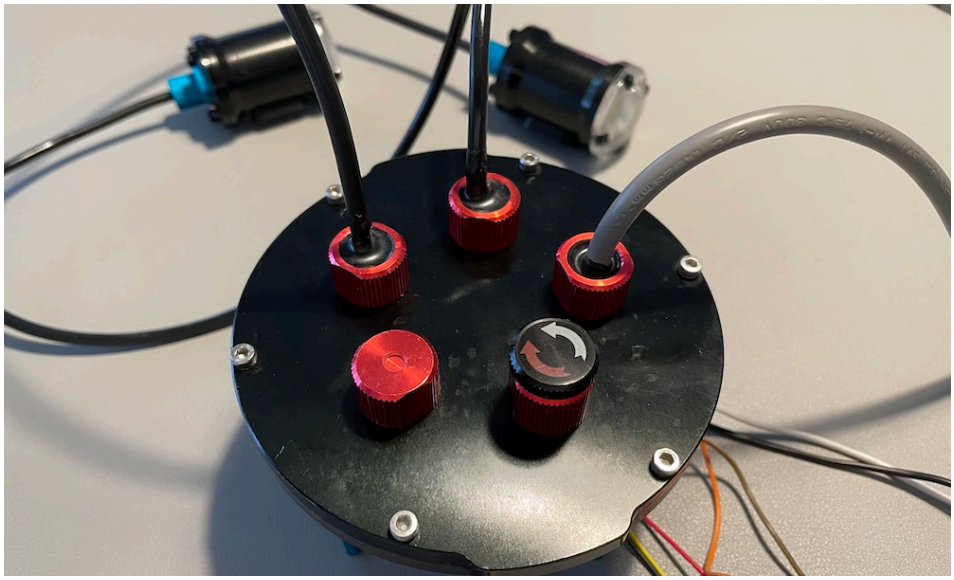


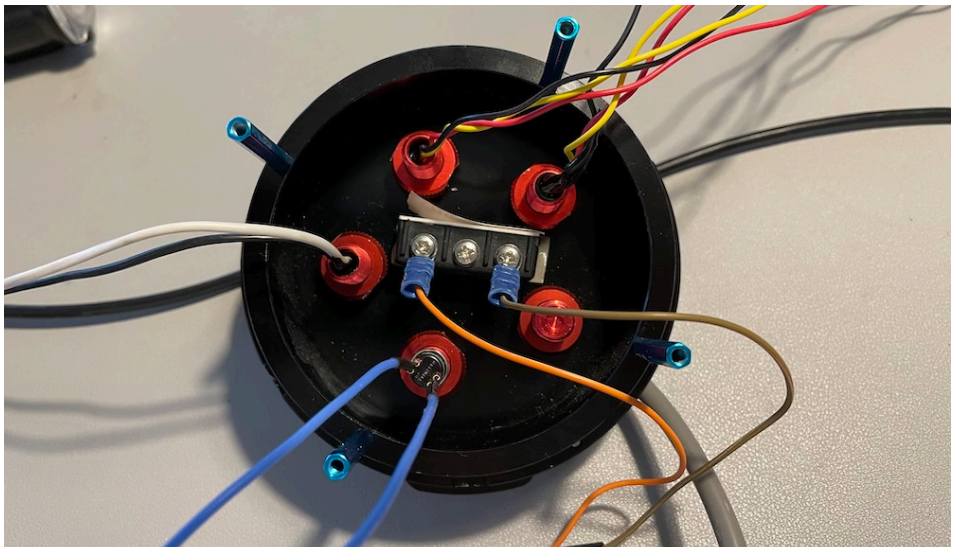
REMEMBER - You will need to pot three cable penetrators: two for the lights and one for the burnwire. I recommend that you do this operation for the three cable penetrators at once, so you can use the potting compound in one round.

a. Pot two cable penetrators with the cables for the lights. I recommend you watch this video https://www.youtube.com/watch?v=mKaJLWv1SCw, which explains how to pot cable penetrators very well.

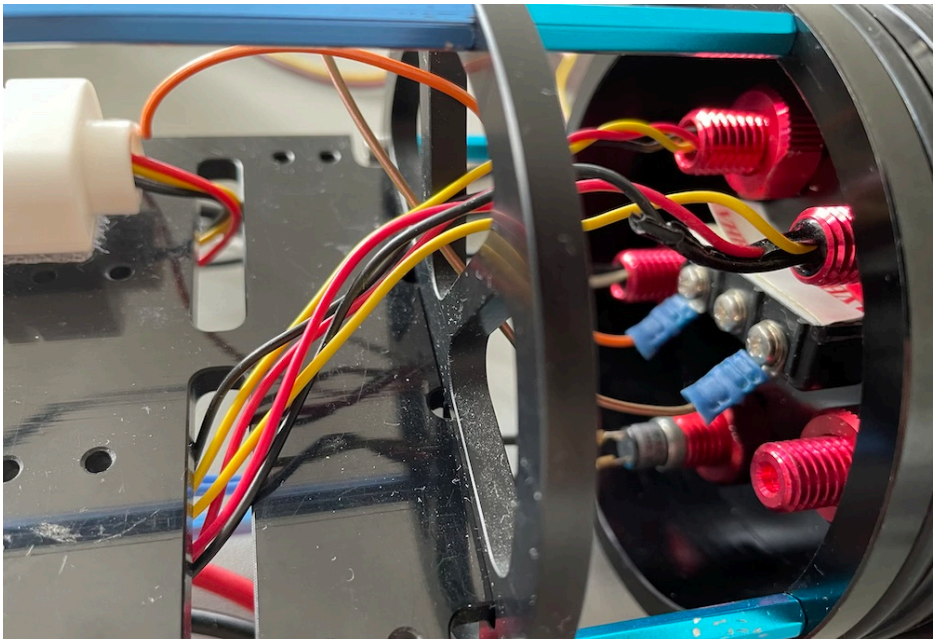


b. Install the two potted cable penetrators and a cable penetrator blank in the flat end cap. In the pictures below, you can see the two potted cable penetrators for the lights (black cables), and also the potted burnwire cable (gray), the electronic switch, and the cable penetrator blank.





c. Now, attach the flat end cap to the electronics tray, and guide the wires of the light cables through the routing slots in order to be able to connect them to the breadboard in the next step.

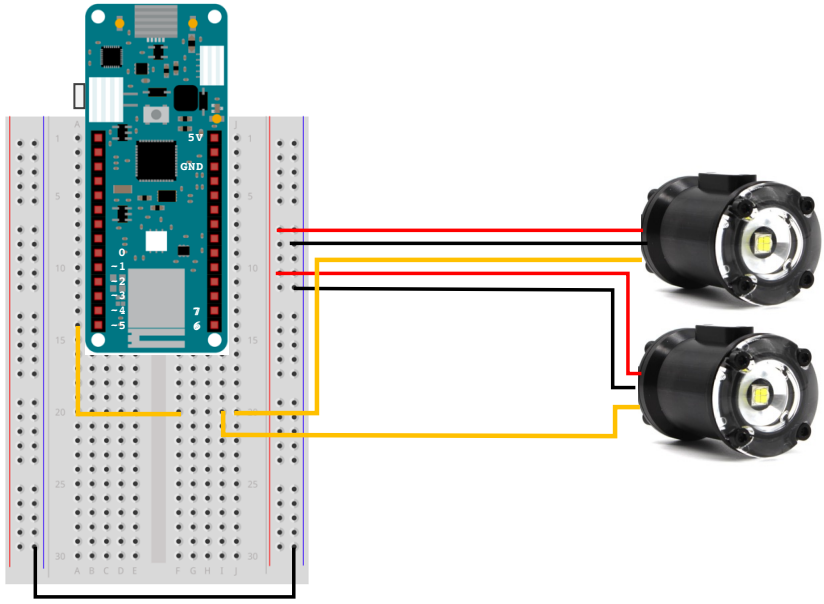


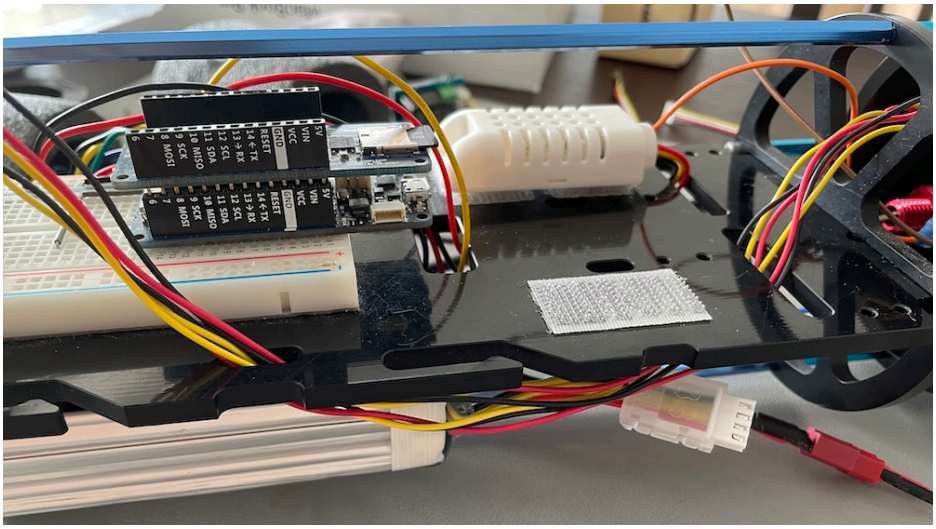
2. **Connect lights to the breadboard**. Each light cable has three wires: power (red), ground (black) and signal or control (yellow).

    a. Connect the power wires to the + row of the right power rail of the breadboard.
    b. Connect the ground wires to the – row of the right power rail.
    c. Connect the signal wire to the tie points 20-I and 20-J.

d. Cut 4 or 5cm of yellow wire, insert one end in tie point 14-A, connected to pin 5 of the Arduino, and the other end in tie point 20-F. Connecting pin 5 of the Arduino to the two signal wires will allow you to control them (turn them on/off and change the brightness).

For simplicity, the schematics just shows the wiring of the lights in the breadboard.





3.3 Burnwire

Parts

1 Grove relay

80cm Electric cable with two wires
1 Cable penetrator
2 x female wire connectors
2 splicing connectors (2 ports)
1 Ring terminal
Solid wires

Tools

Screwdriver
Hex key set
Wire strippers
Terminal crimper tool
Velcro strip
Scissors

Instructions

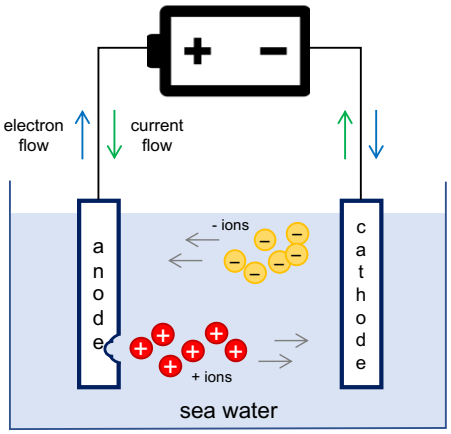
This was perhaps the most complex feature in the whole project: developing a reliable mechanism that allows the device to “cut” ties with the ballast, so it can come up to the surface. It took me a lot of time to solve two issues:

a. The 3.7V-powered motherboard needs a relay to control the burnwire, a circuit powered by a 11.1V battery. Almost all relays are triggered by 5V but Arduino’s output pins only provide 3.27V. The challenge here was to find a relay that could be activated by 3V, but after a lot of searching, I found it.
b. The other challenge was the approach to “burning” the wire. After testing different approaches, I realized that electrolytic erosion was the best way to cut the nichrome wire. This took a lot of time and testing.

The burnwire consists of an electric cable with two wires:

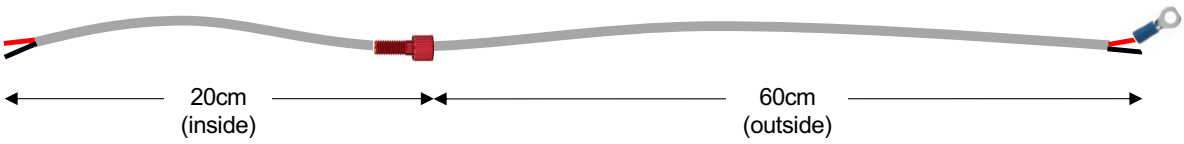
- Anode: one wire has the function of an anode. One end is inside the watertight enclosure, connected to the relay, and the other is outside, connected to a nichrome wire. When the blue_eye wants to activate the burnwire, Arduino’s pin 7 triggers the relay, which closes its circuit to power the this wire with 11.1V.
- Cathode: the other wire is the ground wire and functions as a cathode. One end is inside the watertight enclosure, connected to the ground row in the right power rail and the other end is outside, close to the nichrome wire.

The flow of current underwater between the anode and the cathode will cause electrolytic erosion of the nichrome wire in the part that is not coated. When this wire breaks, the blue_eye detaches from the sandbag ballast.



1. **Potted cable penetrator.** We need to pot a cable penetrator for this electric cable. Follow the steps described in the previous section 3.2.1.

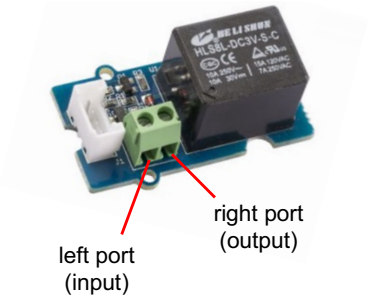
   Install a ring terminal to the positive wire of the electronic cable on the end that will be underwater, using the terminal crimper tool.



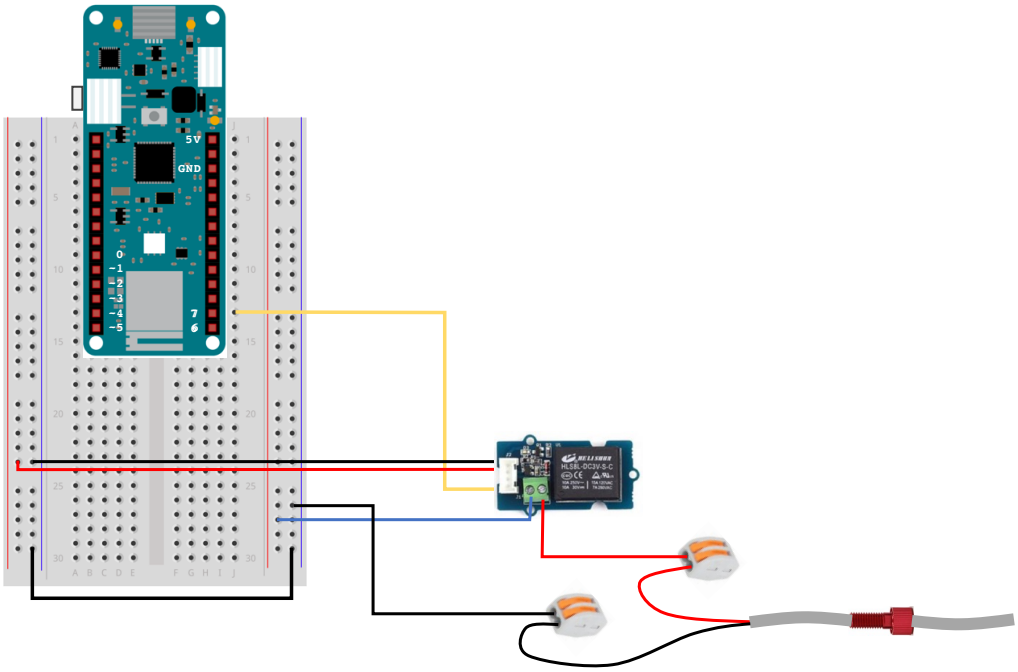
2. **Relay**. The relay comes with a grove 4 pin cable, as you can see in the picture below.

   a. Cut a 4cm Velcro strip. Stick one side of the Velcro under the relay and the other side on the electronics tray. I put it next to the DHT sensor.
   b. Install the relay on the electronics tray, using the Velcro.
   c. Insert one end of the grove 4 pin cable in the 4 pin socket connector of the relay.
   d. Cut the connector of the other end and separate the wires so you can manipulate them more easily. The black wire is for ground, the red wire is for power and the yellow wire is for the control signal, to trigger the circuit relay. You can cut the white wire.

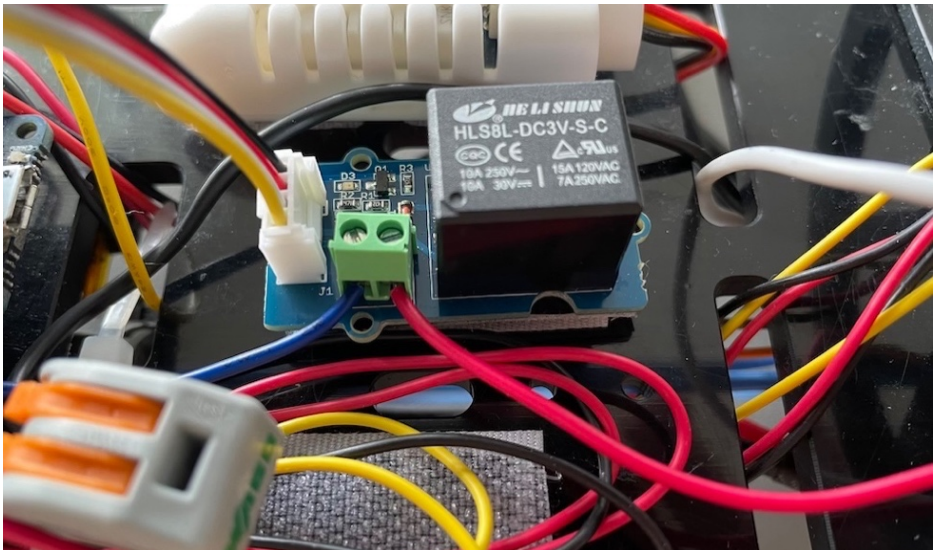
e. Now, let's connect the relay to the motherboard. Insert the end of the red wire in the + row and the end of the black wire in the ground row of the left power rail of the breadboard.
f. The relay is controlled by pin 7 of Arduino. Insert the end of the yellow wire in tie point 13-J of the breadboard.
g. Cut 10cm of blue wire. Install one end in the left port of the green socket in the relay PCB, and install the other end in the + row of the right power rail of the breadboard which provides 11.1V.

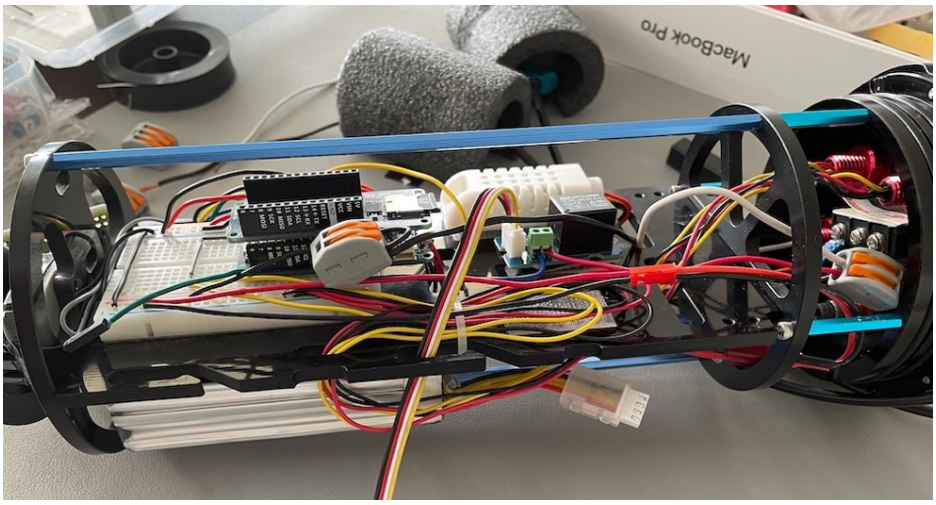


h. Cut 10cm of red wire. Install one end in the right port of the green socket in the relay PCB and install the other end in one of the ports of a Wago splicing connector. Install the red wire of the electric cable in the other port of the splicing connector.
i. Cut 10cm of black wire. Install one end in the ground row of the power rail of the breadboard, and the other end in one of the ports of a Wago splicing connector. Install the ground wire of the electric cable in the other port of the splicing connector.



At this point of the assembling, there are so many wires that my pictures may not help much and could even confuse you. It is critical that you are very clear about the wiring color code during the whole installation.





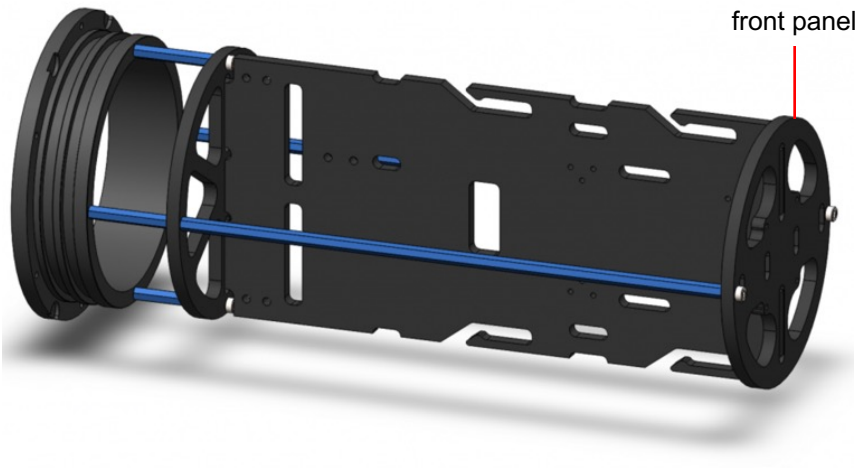
3.4 Video camera

Parts

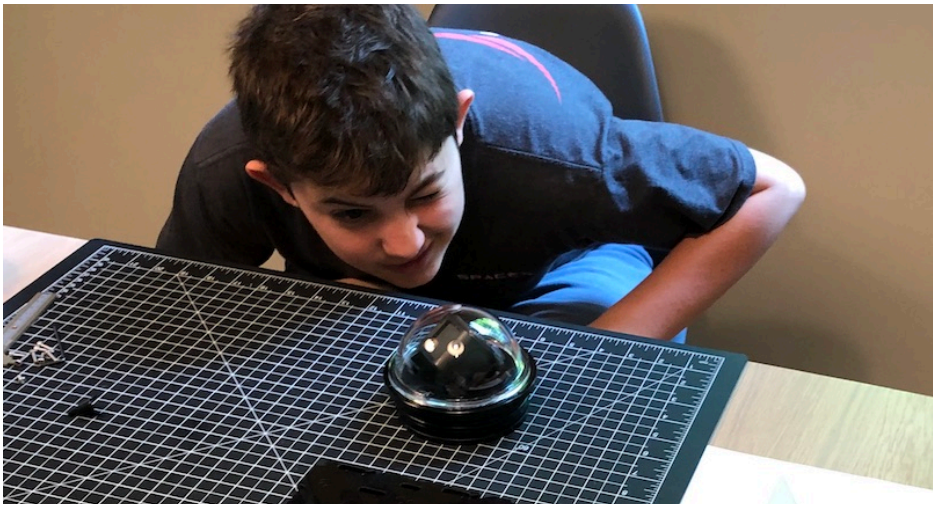
1 GoPro frame mount housing
1 Flat adhesive mount (an accessory that comes with the GoPro in the box)

Instructions

1. **Install mount housing**. You will install the GoPro on the front panel of the electronics tray, which faces down when you deploy the blue_eye.



To do that, you need two accessories that come with the GoPro bundle: a frame mount housing and a flat adhesive mount, which will be attached to the front panel of the electronics tray. Bear in mind that you need enough space to install the GoPro at the desired angle (for example, 45 degrees) and there is little space between the front panel and the dome end cap. To do that, I suggest you place the front panel on a table, put the flat adhesive mount with the frame mount housing on top, in a 45 degree angle, and put the dome end cap on top, as shown in the picture below.



By doing this, you will be able to find the right place to stick the flat adhesive mount on the front panel, which is somewhere near the bottom-center in the schematics below:

Frame mount housing attached to the flat adhesive mount:

3.5 Watertight enclosure

Parts

1 Aluminum tube
1 Dome end cap
6 M3-12mm screws
1 O-Ring flange
1 Enclosure clamp
4 M3-10mm screws
2 Light mounts (custom 3D parts)
4 M3-5mm screw (for light supports)

Tools

Vent plug
Vacuum plug
Vacuum pump
Silicone
Hex key set

Instructions

1. **Insert aluminum tube**. When you arrive to this step, the flat end cap has already been attached to a O-ring flange, which has the electronics tray attached, as shown in the following picture.

Apply silicone to the two rubber bands of the O-ring flange to make it easier to insert the O-ring inside the aluminum tube. A video tutorial from Blue Robotics explains these two tasks very well: https://www.youtube.com/watch?v=G6PqEsKjxHM.

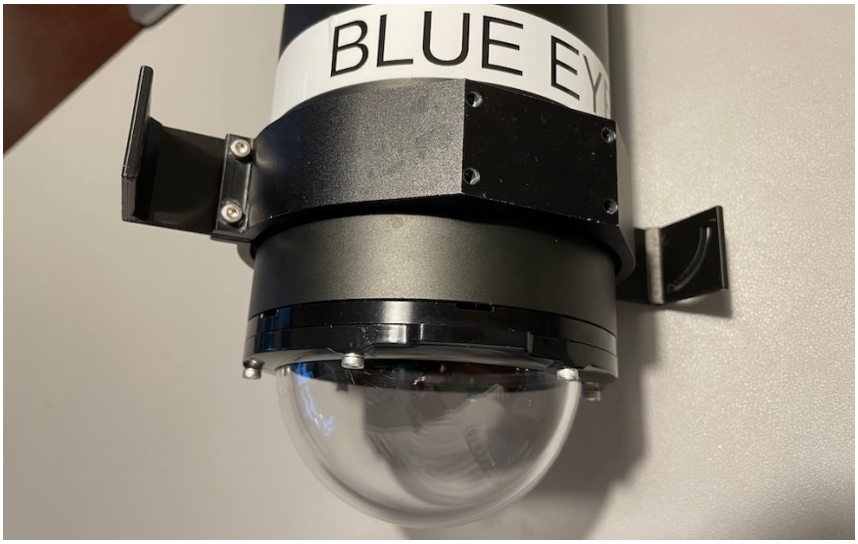
Now, install the aluminum tube on top of the O-Ring, by applying pressure. As you can see in the previous picture, I put two piles of magazines under the flat end cap, making sure the potted cable penetrators and the cables do not touch anything, so I can press down on the aluminum tube towards the O-Ring.

2. **Install dome end cap**. Attach the dome end cap on top of the O-ring flange, using 6 M3-12mm screws. First, apply a drop of silicone to each of the three rubber bands to seal the attachment and make it easier to insert the O-ring flange inside the aluminum tube. Insert the O-ring flange with the dome end cap inside the aluminum tube, by applying pressure.

3. **Vacuum test.** You need to verify that the enclosure is sealed and watertight. This is a critical step in the assembly process and to do so, you need a vacuum pump and a vacuum test plug. Now is when you will use the fifth hole of the flat end cap that you haven't used yet (now, it has a cable penetrator blank). In this web page, https://bluerobotics.com/learn/using-the-vacuum-test-plug/, Blue Robotics explains very well how to perform the vacuum test.

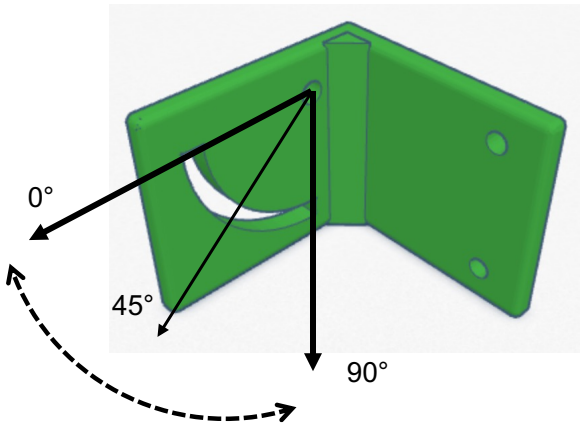
4. **Install light mounts**. Now install the light mounts, between the joints of the enclosure clamp around the aluminum tube (picture from Blue Robotics), using 4 M3-10mm screws.

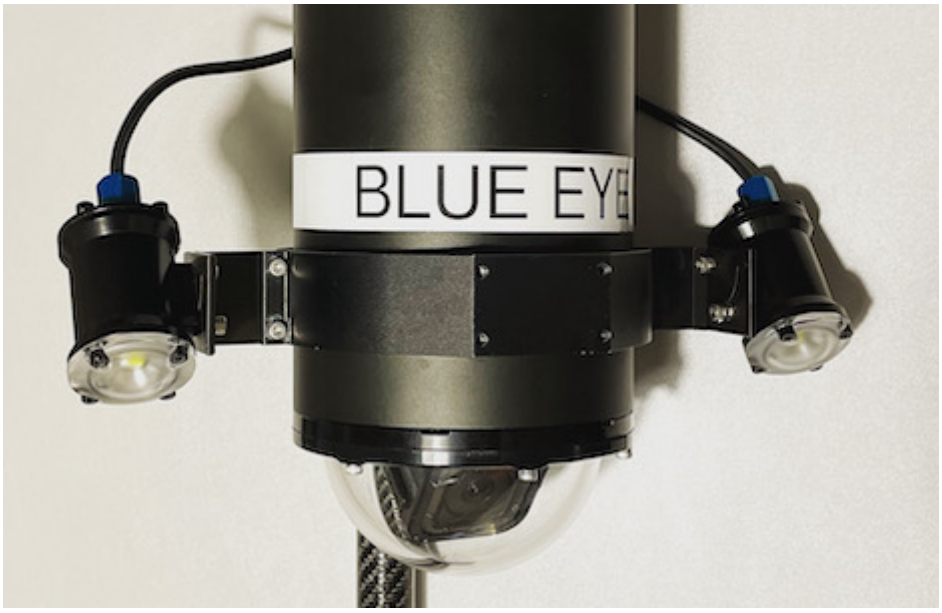
As you need to install the lights as close as possible to the GoPro, you will attach the enclosure clamp around the bottom part of the tube, close to the dome end cap, as you can see in the picture below.



Make sure you install the clamp in a way that the light mounts are aligned horizontally with the lens of the GoPro: each light is on each side of the lens

5. **Install lights**. Attach the lights to the light mounts, using 2 M3-5mm screws in each one. The light mounts have a design that offer 90 degrees of freedom to align the direction of the light with the direction of the GoPro.





3.6 Mast

Parts

1 Carbon fiber tube 24”
1 Top modular connector
1 Bottom modular connector (threaded end)
1 Link bottom modular connector
Epoxy adhesive

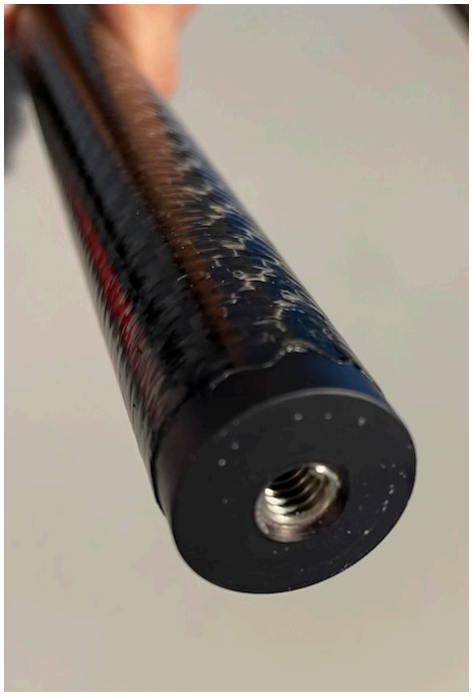
1 Mast clamp (custom 3D part)
1 Stainless Steel Black Oxide Bolt - 0.75" x 1/4"-20 thread
4 M4-10mm screws
2 M3-12mm screws and nuts

Tools

Hex key set

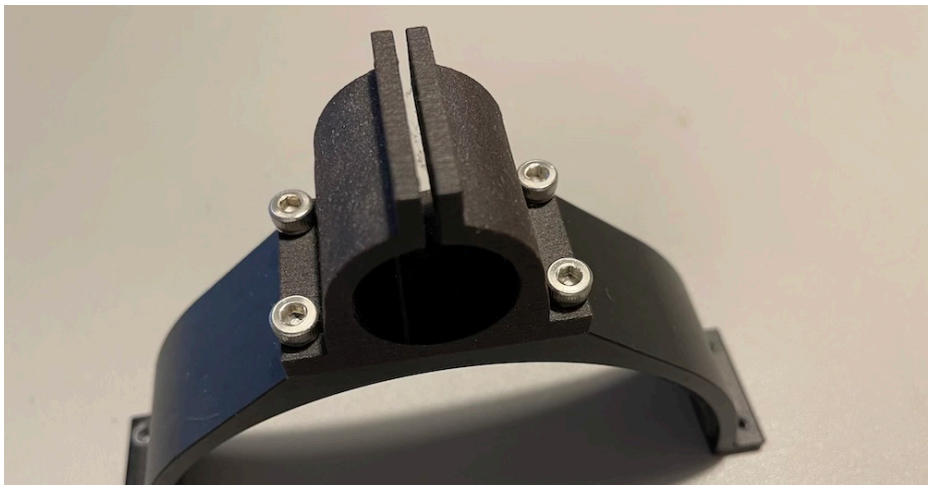
Instructions

1. **Build mast**. The blue_eye has a mast attached to its enclosure, as a versatile platform for installing and tying several elements: the burnwire, the sandbag ballast, the bait bag, a float and a weight. You could also install a flag to make it easier to see the device when it comes back to the surface, just like the dropcam developed by National Geographic. For building the mast, you must connect three parts: the top modular connector to one end of the tube and the bottom modular connector to the other end of the tube using epoxy adhesive. Follow the instructions provided by Dragon Plate, the supplier of the epoxy and these three parts.

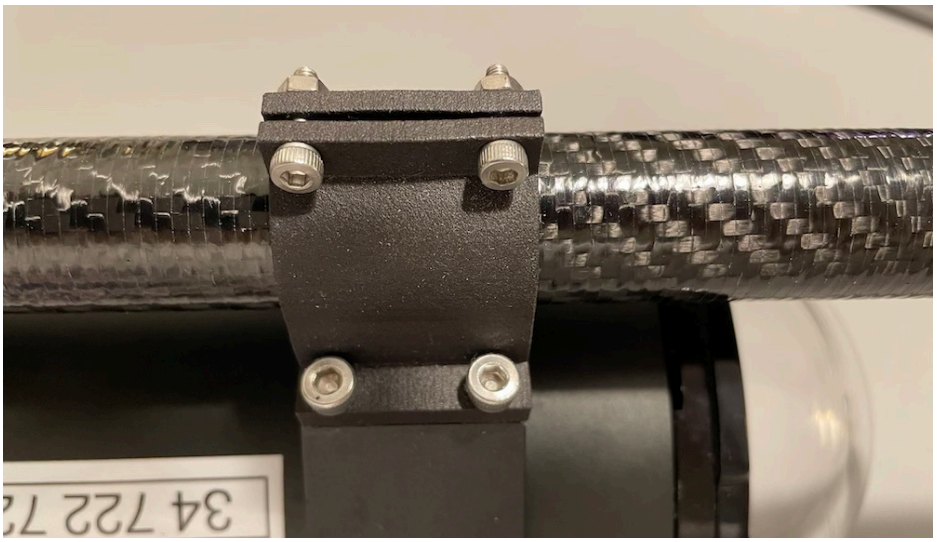


Once the three parts are bonded and the epoxy cured, you can attach the link bottom modular connector to the bottom connector, using the black stainless bolt through the hole in the middle of the connector.

2. **Attach mast**. First, attach the mast clamp to the enclosure clamp, using 4 M4-10mm screws.



Now insert the mast through the mast clamp and tight the clamp using 2 M3-12mm screws and nuts.

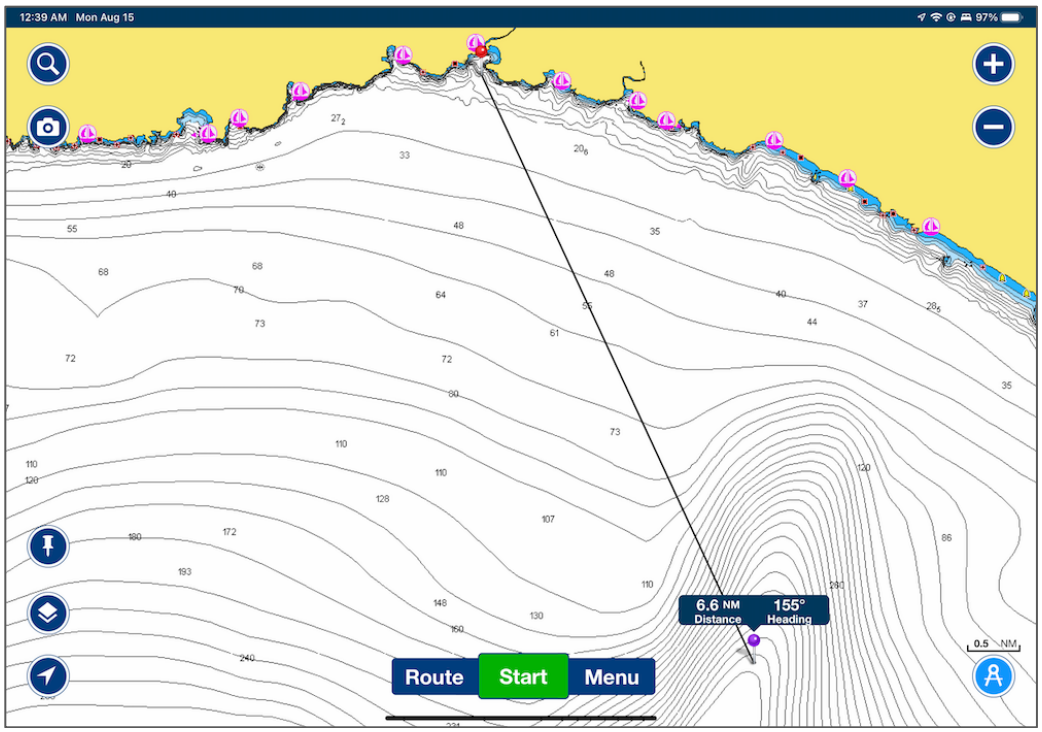


3. **Label**. At this point, you have built your blue_eye. I suggest you put a sticker with your name, email and phone number, and a short description of what this device is, so if something happens in a deployment and you lose it, you still have a chance of recovering it if someone finds it. It may sound unlikely, but it can happen. According to my dad, a National Geographic team lost one of their dropcams in the Atlantic, but a few months later, a man contacted them from France, saying that he had found it floating next to his sailboat. They recovered it!

4. How to use the blue_eye

4.1. Decide on the location and get its depth. First of all, you need to know where you want to drop the blue_eye in latitude and longitude coordinates. You can use the GPS of your boat or the GPS in your smartphone (see screenshot below). You will use these coordinates to make sure you are at the same place you dropped the blue_eye to recover it when it returns to the surface.

You also need to know the depth of your selected location for two reasons. The first reason is that the location should not be deeper than 500 meters, which is the maximum depth that the blue_eye can hold (the tube and the flat end cap are rated for 1,000 meters but the dome end cap, an optically clear acrylic dome, is rated for 500 meters). The second reason is that you can use the depth to (a) estimate how long it will take the blue_eye to reach the ocean floor, (b) set up when you would like it to start recording, and (c) estimate how long it will take to get back to the surface. You can use the probe on your boat or the bathymetry of navigation charts such as Navionics (see screenshot below).



4.2. **Video resolution and recording time**. You also need to decide the quality of the video and the recording time in minutes. Bear in mind that recording time depends on three variables: the video resolution, frames per second (FPS), and the size of the microSD card. My GoPro has a microSD card of 32GB and it can record 48 minutes of video in 4K resolution at 30 FPS. However, the camera can also accept a microSD card of 128GB, which offers up to 3h15min of 4K video resolution at 30 FPS. When the GroPro records in 4K resolution, it stores the videos in files of 8 minutes 53 seconds, which have a size of 4GB. In order to set up the video resolution in the GoPro, check pages 36 and 37 of the GoPro User Manual.

4.3. **Batteries and memory.** Another step in the preparation of the blue_eye for an exploration mission is to charge the different batteries:

- 3.7V battery for the motherboard (I use Adafruit Micro-Lipo Charger for LiPo/LiIon Batt w/MicroUSB Jack),
- 11.1V battery for lights and burnwire (I use LiPo Charger RC Balance Fast Charger Discharger),
- GoPro's battery (I use the USB cable that comes with it).

REMEMBER: make sure the microSD card of the GoPro is empty, so it can record as much time as you want (you can empty the microSD card directly from GoPro's menu or from a computer).

4.4. **Setting up the blue_eye**. You need to set up the blue_eye to the specific needs and conditions of your exploration, and you do this in the code that will run the Arduino

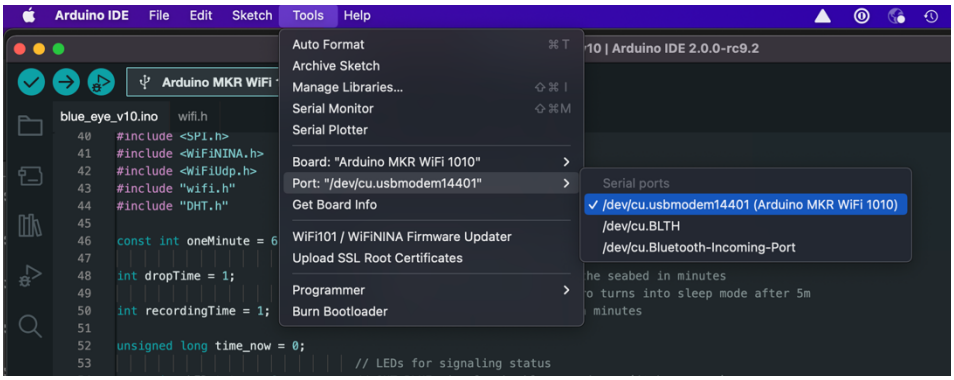
microcontroller. It's not essential that you know how to code Arduino, but you should have at least a basic understanding of the Arduino IDE and how Arduino works. Please follow the next steps:

1. **Connect your Arduino to a computer** with a USB cable.

2. **Open Arduino IDE** in your computer (I was using IDE's version 2.0.0 on a MacBook Pro with macOS Monterey 12.2.1 when writing this guide).

3. **Select microcontroller.** Go to menu Tools > Board > Arduino SAMD Boards (32 bits) > Arduino MKR WiFi 1010.

4. **Select the port** you are using to connect the microcontroller to your computer to the (USB port in my case). Menu Tools > Port > /dev/cu.usbmodem14401



5. **Open the software of the the blue_eye.** First, you download the file blue_eye_v10.ino from the GitHub repository to your computer. Then, you can open the file through Menu > File > Open > (select the folder where you downloaded the code) > blue_eye_v10.ino.

   Once you are here, it is important to understand a key feature of the GoPro Hero 5 Session for its interaction with the microcontroller: once you turn on the GoPro, it will "awake" for 5 minutes, ready to record videos or shoot pictures, and if nothing has happened during these 5 minutes, it automatically turns to "sleep" mode to save energy. This means that once you turn on the GoPro, you have 5 minutes to do the following two things:

   a. **Turn on the blue_eye**, using the electronic switch (see figure X). This will turn on the microcontroller, the brain of the dropcam, and you will know if it has been activated because it turns the blue LED on. The very first thing that will happen is that the Arduino will search for and connect to the GoPro's wifi network (this operation takes between 6 and 8 seconds). When the Arduino is connected to the wifi, it will turn the green LED on.

   b. **Initialize the exploration mission** using the magnetic switch, which means that you are ready to drop the blue_eye overboard. After you activate the magnetic switch, the Arduino will turn the yellow LED on, which means that you are dropping the blue_eye right now.

   In order to explain these two actions in detail, consider the following timing variables:

   - T_0 – The time when you turn on the GoPro.
   - T_1 – The time when you turn the electronic switch (ES) on. You should turn on the ES as soon as you turned on the GoPro, but you have a buffer of two minutes (remember you have to do two things in 5 minutes). $T_1 \leq T_0 + 2$ minutes
   - T_2 - The time when Arduino connects to GoPro's wifi. $T_2 = T_1 + 6$ seconds approx.

- T_3 – The time when you turn the magnetic switch (MS) on. The Arduino “wakes up” the GoPro, which means that the GoPro will be waiting for another 5 minutes. $T_3 \leq T_2 + 3$ minutes
- $T_{droptime}$ – The time you estimate the blue_eye needs to descend to the bottom of the ocean. This timer variable starts counting as soon as you activate the MS and its maximum value is 5 minutes.
- T_4 – time when GoPro starts recording. $T_4 = T_3 + T_{droptime}$

Within the code, I defined the variable `dropTime` (in minutes) as the time it takes for the blue_eye to reach the seabed, after which the Arduino will start recording with the GoPro. The blue_eye descends at a speed of 1.5 meter per second approx. (although it could travel faster with a heavier sandbag). For example, if the depth of the location is 400 meters, it will need 4.4 minutes (266 seconds) to reach the seabed. So you can set `dropTime` = 4.4. However, if you want to record the descent to the sea floor, then you can set `dropTime` = 0. REMEMBER: `dropTime` can’t be longer than 5 minutes, otherwise the GoPro will switch to sleeping mode and the microcontroller won’t be able to activate the recording in the GoPro.

I also defined the variable `recordingTime` (in minutes). In my case, since my microSD card has 32GB of memory, I can record up to 48 minutes of 4K resolution, so I use 48 minutes. Go ahead and edit the desired values of these two variables: `dropTime` and `recordingTime` in the code.

```
int dropTime = 3;              // estimated time to get to the seabed in minutes
int recordingTime = 25;        // duration of video recording in minutes
```

If you want to play with the lighting of the blue_eye (for example, turn them on and off several times, or change their brightness), you can to do so inside the loop of the program, between the instructions `StartRecording()` and `StopRecording()`. See code below:

```
void loop() {

  MagneticSwitch();

  WakeupGoPro();

  time_now = millis();

  printInLog("blue_eye on its way to the deep sea");

  while(millis() < time_now + (dropTime * oneMinute)){
     readDHT22Values();
     delay(oneMinute);
  }
```

```
  printInLog("blue_eye arrived to the seabed");

  printInLog("Turning lights on");

  // turns on lights gradually
  for(int i=0; i<255; i++){
    analogWrite(lightsPin, i);
    delay(5);
  }

  StartRecording();

  time_now = millis();

  // during recording time, checks humidity and temperature every minute
  while(millis() < time_now + (recordingTime * oneMinute)){
    readDHT22Values();

  // if you want to play with lights intensity, on or off, this is the place to do it

    delay(oneMinute);
  }

  StopRecording();

  // turns off lights gradually
  for(int i=255; i>0; i--){
    analogWrite(lightsPin, i);
    delay(50);
  }

  digitalWrite(lightsPin, LOW);

  printInLog("Turning lights off");

  BurnWire();

  DisconnectFromGoPro();

  digitalWrite (LEDmagswitch, LOW);

  while(justWait == true) {
  }

}
```

6. **Set up the wifi network** (ONLY the first time). The code file **blue_eye_v10.ino** has a second file attached to it called **wifi.h** with the two parameters of your GoPro's wifi network: the network name (Service Set Identifier or SSID) and its password. You need to change these two parameters using the values of your GoPro:

```
#define SECRET_SSID "GP5XXXX"
#define SECRET_PASS "bicycle"
```

In order to get the network name and password of your GoPro, you may follow these steps (picture from GoPro User Manual):

1. Press the **Menu** button to turn on the status screen.
2. Press the **Menu** button repeatedly to get to Connection Settings.
3. Press the **Shutter** button [◯], and then press the **Menu** button to get to Camera Info.
4. Press the **Shutter** button to display your camera's username (ID) password (pw).

You only need to do this the very first time that you are going to connect the Arduino to your GoPro.

7. **Dry test**. Once you have set up the variables in blue_eye_v10.ino and the wifi settings in wifi.h, you may want to make sure everything functions: the electronic switch works well, the Arduino connects to the GoPro, the magnetic switch works as expected, the Arduino wakes up the GoPro, sends a message to start recording after `dropTime`, sends a message to stop recording after `recordingTime`, the lights and burnwire work (burnwire is described in section 4.8, you will need a bucket of salt water). For a dry test, I suggest you use shorter times for `recordingTime`, for example, 5 or 10 minutes.

   In order to do a test, you need to do the following:

   1. Upload software to the Arduino: click the button "Upload" in Arduino IDE's window.
   2. Disconnect Arduino from computer.
   3. Turn the GoPro on.
   4. Activate the magnetic switch, and see how the blue_eye turns on the different LEDs and performs each instruction.

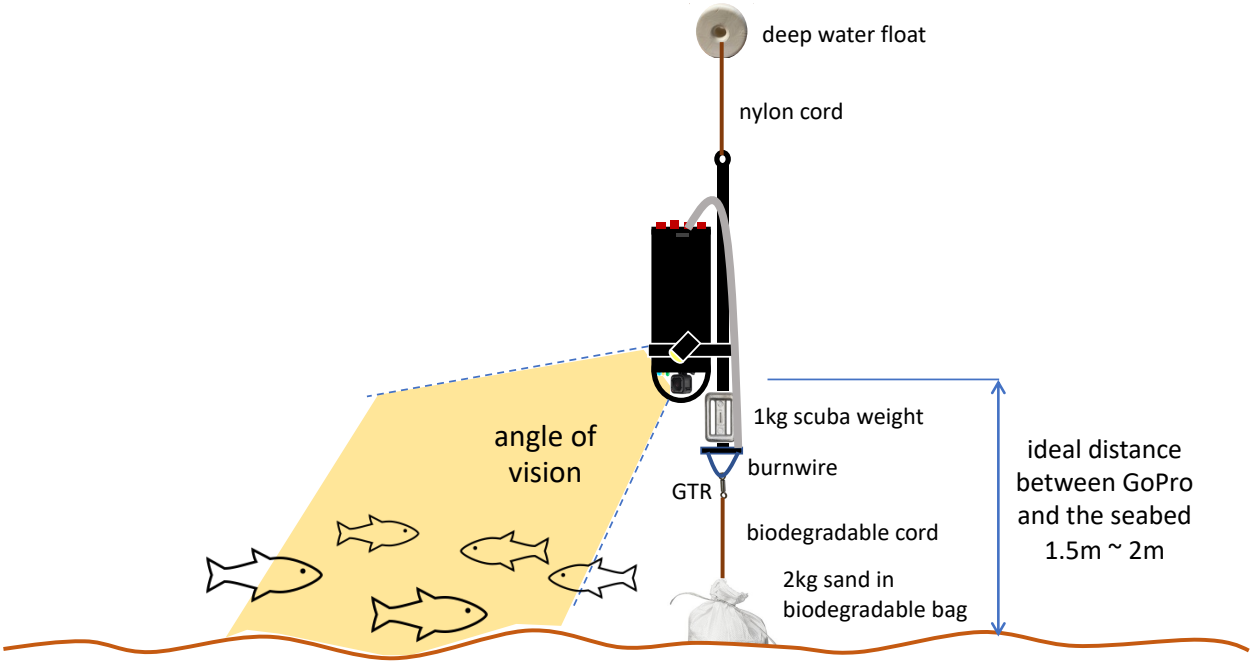
4.5. Prepare for actual drop in the sea. When the dry test works as expected, you are ready to set up the blue_eye for the actual drop in the ocean. You just need to follow these three actions:

a. Assign the desired values to `dropTime` and `recordingTime` in minutes.
b. Connect the Arduino to your computer and upload the software.
c. Make sure the two batteries and the GoPro are fully charged.

Wether you performed tasks 1 to 7 from your home, school, or Q Division, you are now aboard a boat, heading to the location you want to explore the seabed. REMEMBER: Don't forget to have the magnet part of the magnetic switch with you, so you will be able to activate the switch. While you are sailing to the desired location, you will execute the following tasks with the specified materials:

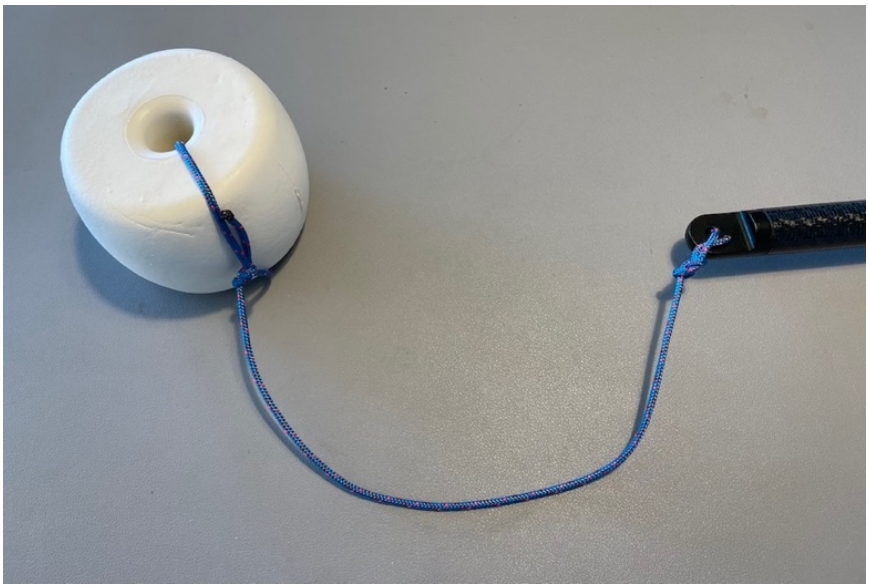
Parts

- Deep water float
- 3mm nylon cord
- 3mm biodegradable cord
- Scuba weight 1kg
- Burnwire
- Cable zip ties
- Electric tape
- Nichrome wire
- Sand 2kg
- Biodegradable bags
- Bait
- Bait bag
- Galvanic time release

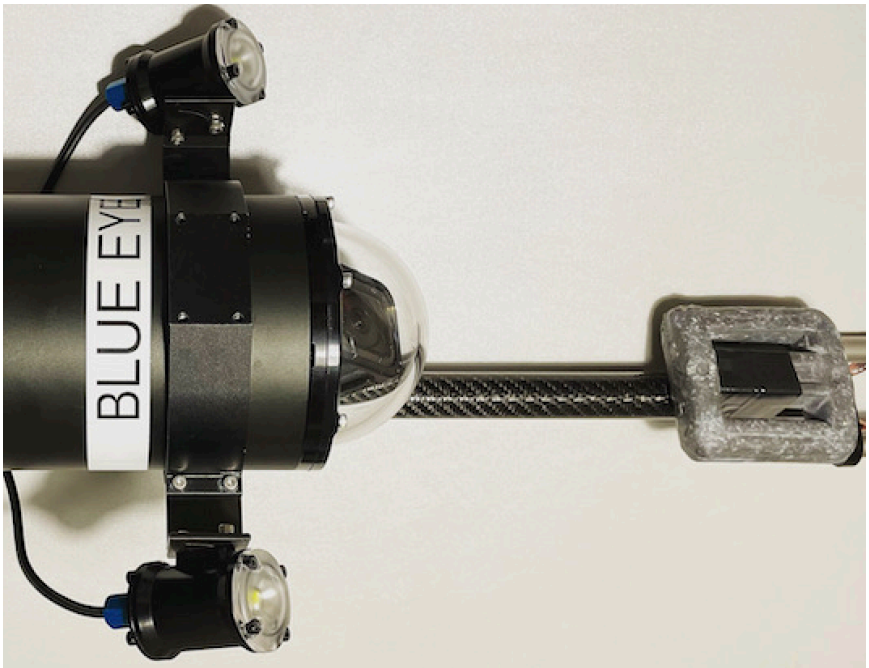


Note: in all my deployments, I had the GoPro at 1.5 meters from the seabed but I understand the “ideal” distance will depend on the purpose of your exploration, so it could be lower or higher.

4.6. Tie the deep water float to the top of the mast, using the 3mm nylon cord (distance between the float and the top of the mast is 0.5m approx.).



4.7. Attach a 1kg scuba lead weight to the bottom of the mast. I use black electric tape, as you can see in the picture, but you can also use a 3mm cord. This weight together with the float will keep the blue_eye vertical during the deployment and recovery, and the float will push the blue_eye to the surface after the burnwire cuts the line to the ballast.



RECOMMENDATION: As you can see at this point of the guide, there are several elements that have to work perfectly in order to have a successful recovery of the blue_eye. I recommend you use a second float during the first deployments of the blue_eye in shallower waters, as another backup recovery system in case something goes wrong with the burnwire and the blue_eye can’t go up to the surface at the scheduled

time. If the burnwire doesn't work as expected and you don't have this second float, you will still be able to recover the blue_eye because you have the GTR. However, the GTR has a release time of 1 day, which may vary depending on the temperature and salinity of the water (It may differ by -2 to +2 hours). In this case, you should need to be at the location of the drop for a few hours, waiting for the GTR to disintegrate and recover the blue_eye after it reaches the surface.

This second float also helps you see where the blue_eye is. You will need a cord almost as long as the depth of the location of the drop. This works perfect in shallow waters and even in depths of 200m or 300m, but it is not convenient for greater depths. In the following picture, you can see my dad and I pointing to the second float, which marks the location where we dropped the blue_eye for testing.

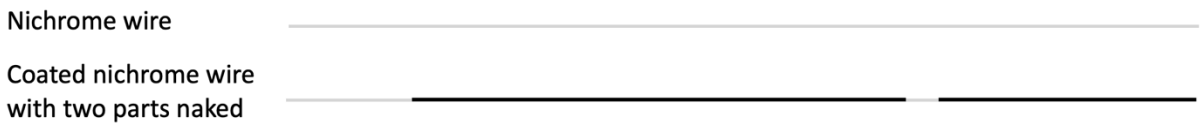
I find the burnwire mechanism is a very elegant engineering solution for the recovery of a device that you drop to the bottom of the ocean and it gives you the freedom to leave and come back to recover the blue_eye. However, the safest way to recover a dropcam will always be always with a tethered rope. You don't need to rely on batteries, wires, and software to produce the electrolytic erosion of a nichrome wire. Rather, the deep-sea drop-cam would be operated with an electric reel to make the recovery easier and convenient. It's not very realistic to recover a device from 1,000 m of depth by pulling a rope by hand.

The only downside of this mechanism (it is only a minor detail, nothing dramatic) is that you must wait at the location of the drop for the time you want to explore the bottom of the ocean. The leader in these types of reel-based drop-cams is Dr. Brennan Phillips, Assistant Professor in Ocean Engineering at the University of Rhode Island, and Principal

Investigator of the Undersea Robotics and Imaging Laboratory (URIL) who helped me a lot during this project (many thanks Brennan!).

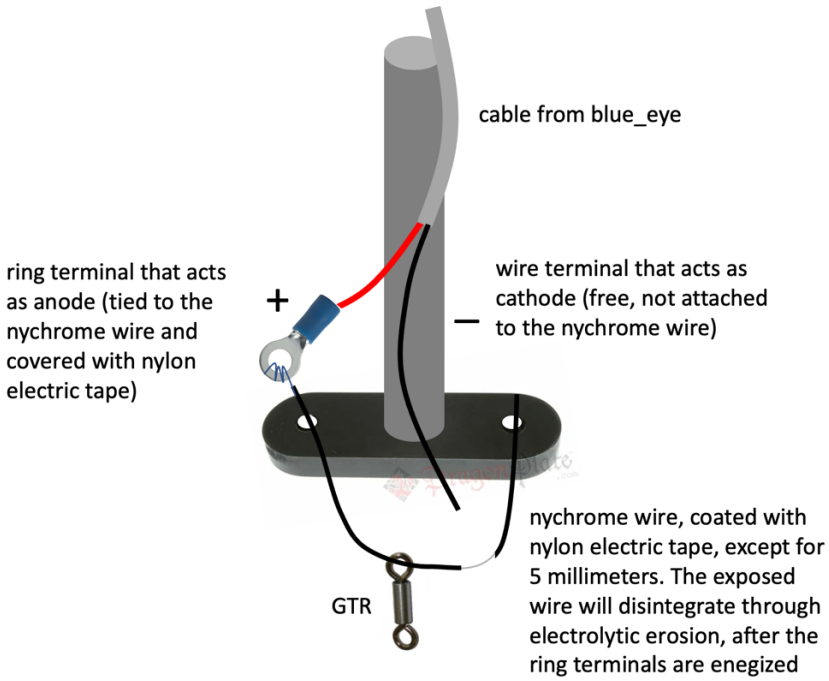
4.8. Attach the burnwire to the mast using two or three cable zip ties.

4.9. **Install the burnwire and GTR** as shown in the following diagram. I don't have actual pictures of this installation, so the diagram helps illustrate the details. Cut 12 cm of a nichrome wire (I use 24 gauge) and use electronic tape to coat two parts of the wire, leaving 2 cm naked on one end and 5 mm in between, as follows:



Pass the left side of the wire through the left hole of the modular connector and tie it to the positive ring terminal (anode). Now cover the ring terminal and the naked end of the wire with electronic tape too.

Insert the right side of the wire through one of the holes of the GTR and tie the wire to the right hole of the modular connector. The only part of the nichrome wire exposed to salt water will be the 5 millimeters. The following diagram illustrated the details:



When the Arduino activates the burnwire, it takes about a minute to disintegrate a wire of 24 gauge (0.51 milimeters). If you use a lower gauge (a thicker wire) it will take longer to disintegrate.

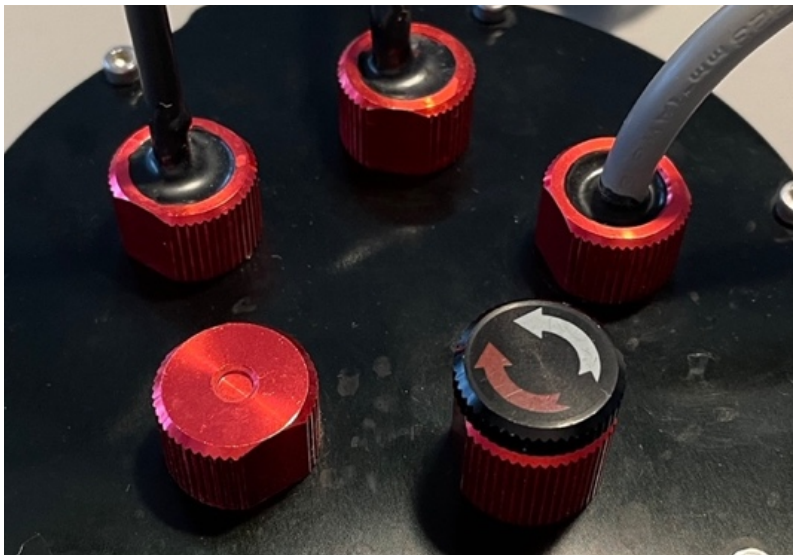
4.10. Tie the ballast, a cotton bag with 2kg of sand, to the free hole of the GTR with a biodegradable cord. It is critical that you use a biodegradable bag and cord because they will be left in the ocean after the burnwire disintegrates the nichrome wire and you don't want to leave any residue or pollutant behind.

IMPORTANT: The distance between the GoPro and the seabed is very important for the quality of the videos that you want to record, as you want the capture images with good perspective and proximity to the seabed. If the seabed is rocky or sandy, you may want to have a distance of 1.5 meters, but if the seabed is a meadow of algae, you may want to have a distance of 2 meters (remember schematics in page 44).

4.11. Tie a bait bag (OPTIONAL). You may want to attach a bait bag to help attract marine life to the blue_eye. In this case, fill the bag with parts of fish and tie it around the mast clamp with a rope measured to keep the bait bag between the GoPro and the ballast. This bag will remain at the back of the GoPro, outside of its focus.

4.12. Turn on the GroPro. Now that everything has been set up outside the blue_eye (water float, burnwire, GTR, ballast and optionally, bait bag), it is time to turn on the blue_eye. Open the dome end cap to have access to the GoPro. The newest Blue Robotics' O-Ring flanges have a locking cord handle that makes this operation easier and faster. Once you have access to the GoPro, turn on the camera by pressing the button at the back.

4.13. Turn on the blue_eye, turning the knob of the electronic switch clockwise.



This switch will allow the 3.7V battery to power the microcontroller, the brain of the blue_eye, and will turn on the blue LED to signal that the whole system is on.

The very first thing that will happen is that the Arduino will search for and connect to the GoPro's wifi network. This operation takes between 6 and 8 seconds, during which the green LED is blinking. When the Arduino stablishes a connection to the wifi, then the green LED is permanently on. On very few occasions (and I still don't know why), the Arduino has trouble trying to connect to the GoPro, which you can notice because the green LED will keep blinking after 8 seconds and never stays on permanently. If this happens, my recommendation is to turn off the GroPro and the electronic switch and repeat these two steps again (4.11 and 4.12).

4.14. Close the dome end cap. Once the Arduino is connected to the GoPro's wifi, it can control the video camera by sending commands to start and stop recording. Close the dome end cap of the blue_eye, making sure that it is perfectly attached to the aluminum tube. If it is not well attached, there is the risk of water leakage, which could be a disaster for the whole device.

4.15. Activate the magnetic switch. Activate the magnetic switch to initialize the exploration mission, which means that you are ready to drop the blue_eye immediately. **You have a maximum of 5 minutes from the time the Arduino establishes a connection to the GoPro's wifi until you turn on the magnetic switch**. If you don't turn on this switch before 5 minutes, the GoPro will turn to sleep mode to save battery.

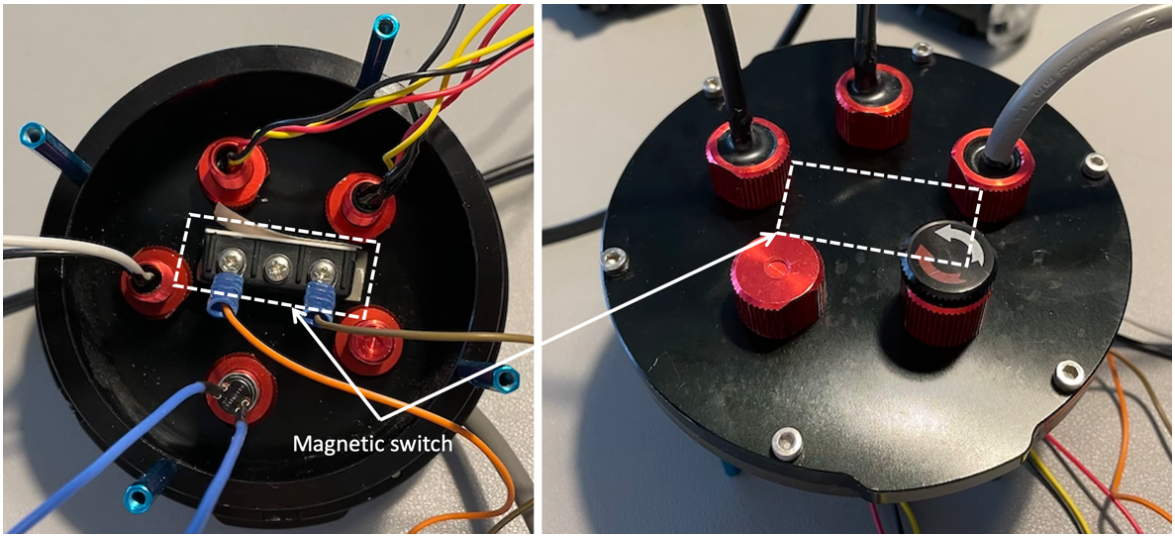
```
void MagneticSwitch(){
  digitalWrite(LEDmagswitch, LOW);
  switchState = digitalRead(magswitchPin);
```

```
  while (switchState == HIGH){
     switchState = digitalRead(magswitchPin);
   }
  digitalWrite(LEDmagswitch, HIGH);
  printInLog("Magnetic switch activated");
}
```

Once the magnetic switch is activated, the Arduino wakes up the GoPro to be ready to start recording after `dropTime`.

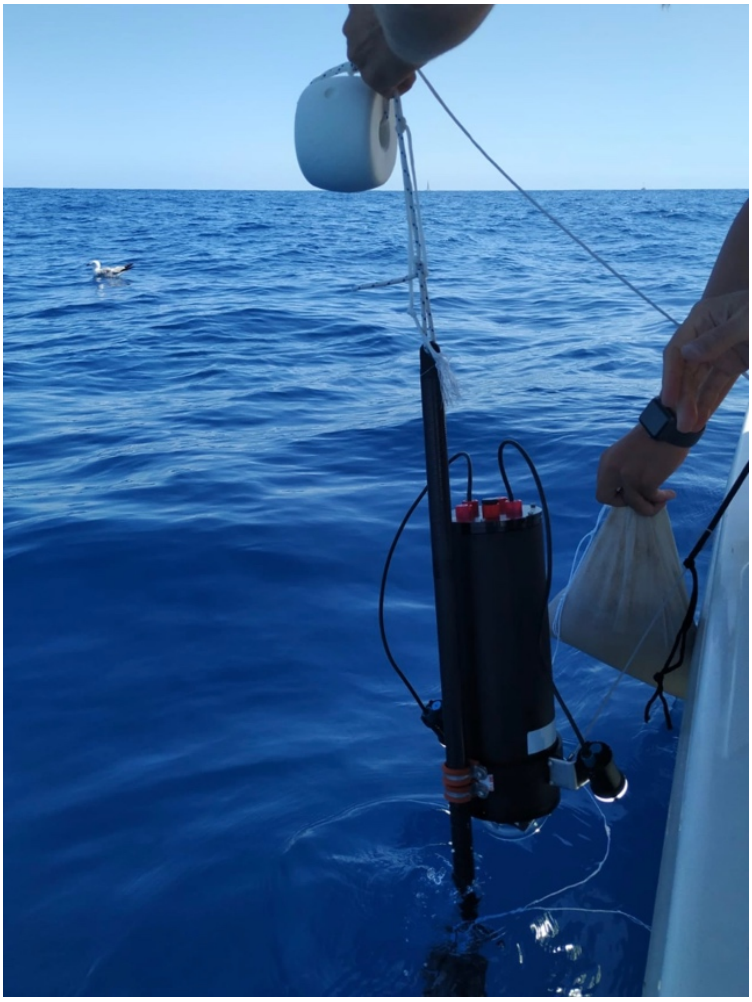
```
void loop() {
  MagneticSwitch();
  WakeupGoPro();
  time_now = millis();
  printInLog("blue_eye on its way to the deep sea");
```

Remember that the magnetic switch has two parts: one is a reed switch, which is installed below the flat end cap (see pictures below), and the other part is a magnet. When the magnet is less than 13 mm away of the reed switch, it closes and sends an activation signal to Arduino (`switchState == LOW`).



In order to activate the magnetic switch, you need to bring the magnet to the flat end cap, close to where the reed switch is located. When the magnetic switch is activated, the Arduino turns on the yellow LED , wakes up the GoPro, and starts to count towards the `dropTime`.

4.16. Drop it overboard! Now you can drop (carefully) the blue_eye with its ballast and float overboard. Take note of what time it is, so you can calculate when the device should be back on the surface.



You can stay there or leave and come back later to recover it. After it finishes the recording time, the burnwire disintegrates the nichrome wire and the device ascends to the surface. For example:

Location	40°4’31’’ N 4°7’9’’ E
Time when you drop the blue_eye	10:15 am
Depth	360 meters
A. dropTime	2 minutes
B. recordingTime	40 minutes
C. Disintegration of nichrome wire	1 minutes
D. Ascension to the surface	6 minutes
Duration = A + B + C + D	49 minutes
Time when the blue_eye should resurface	11:04 am

4.17. Recover it! Once you recover the blue_eye, you turn it off by turning the knob of the electronic switch counterclockwise. Once you go back to port the first thing you may want to do is to rinse the device with freshwater to remove the salt from its surface.

4.18. Enjoy the videos. Now, you can remove the dome end cap and gain access to the GoPro, get its microSD card, and insert it into your computer to enjoy the videos. That's all! If you want to do a new exploration mission, go back to the first task of this chapter (4.1).

4.19. If you want to read the log, then you need to get access to the Arduino. To do that, you need to:

- Remove the cable zip ties that attach the burnwire to the mast.
- Uninstall the lights from the light mounts.
- Separate the flat end cap from the aluminum tube.
- Get the microSD card from the memory shield, where the Arduino stores the log.

Here is an example of a log from one of my tests. The enclosure was not perfectly watertight and a few drops of sea water made its way inside the blue_eye; you can see that the values of humidity went up during the test: