Guidebook

For troubleshooting issues and better understanding of the Purdue EPICS Hydroponics Project

Presented by: Purdue EPICS, Hydroponics Team
epics-asme@ecn.purdue.edu

Last Revised: November 2, 2025

Section	Table of Contents	Page
Design	Design Specifications	3
	Design	4
Construction	Constructing the Frame	7
	Constructing the Manifold	8
	Adding the Pump to the Reservoir	12
	Constructing Grow Beds	13
	Integrating Mylar	20
	Constructing Nutrient Distributor	23
Standard Use	Planting in the System	27
	Cleaning the System	30
	Start of Year Maintenance	31
	End of Year Maintenance	42
Software	Downloading Arduino IDE	48
	Introduction to Arduino	50
	Coding in Arduino IDE	57
	Display Code	64
	Setting Customization	66
Appendix	Troubleshooting	69
	Glossary	71
	Links	72

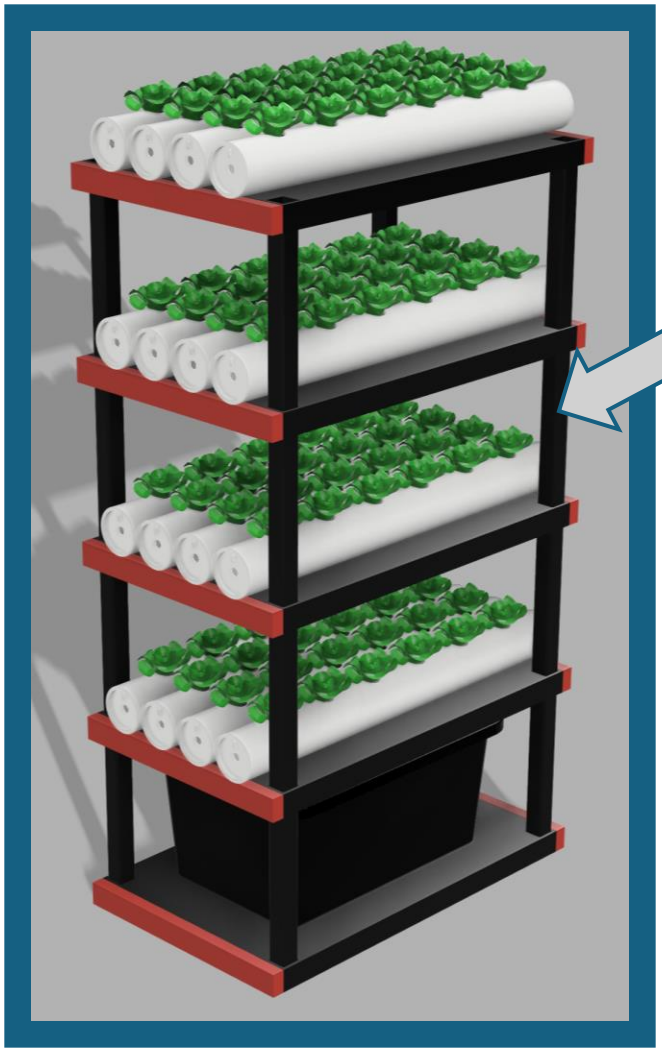
Design Specifications

Things to consider:

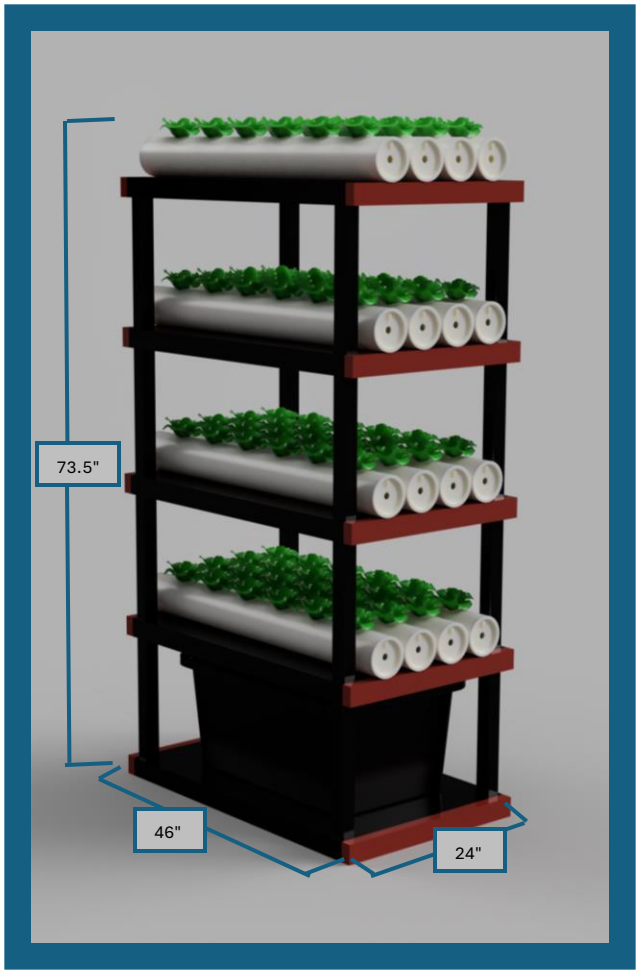
- Number of shelves
- Number of beds per level
- Height of plants
- Number of plants
- Modularity of design

These specifications affect the design of the project as well as the plants that can be grown.

Design



3D structural modeling of the hydroponics system shell



Materials List

Structure:

- Plastic Shelf kit
- Reservoir
- 3x ½” T-shape PVC pieces
- 2x ½” L-shape PVC pieces
- ½” Thick SCH 40 PVC pipe
- 12x 40" 4" Diameter SCH 10 (Drainage) PVC Pipe
- 24x 4" End Caps
- 24x ½" Bulkhead Fittings
- 24x ½" 90-degree Hose Barb Fitting to Male Thread Adapters
- Silicone Food Safe Caulk
- Mylar Sheets
- Vinyl Piping
- Tape
- ~150 GPH Fountain Pump
- PH Control Solutions
- Plant Growth Nutrients
  - FloraGro
  - FloraMicro

Electronics:

- Arduino Giga R1
- Screw Block Breakout
- Arduino Giga Display Shield
- USB Flash Drive
- PH Sensor Kit
- EC Sensor Kit
- 4x Water Level Sensors
- Reservoir Water Level Sensor
- Water Sealed Container
- Relays
- 4x Peristaltic Pump
- Wires
- ~6x 4' LED Grow Lights

Safety

- Be careful when lifting the reservoir
- Pour water only into the tank
- Clean up spilled water
- Wear gloves while handling rockwool

- Drink the water
- Work on electronics while powered
- Clean the tubes while the pump is on
- Climb on the system

Constructing the Frame

Notes

Plastic shelves

- Follow instructions given by supplier
- Hydroponics system can be implemented on any kind of shelf

Why plastic?

- Very durable
- Simple
- Cheap

Materials List

- Plastic shelf kit
- Mallet

Checklist

- Shelf kit contains 4+ shelves
- Shelf kit is greater than 40" in length and 16" in width

Additional Notes

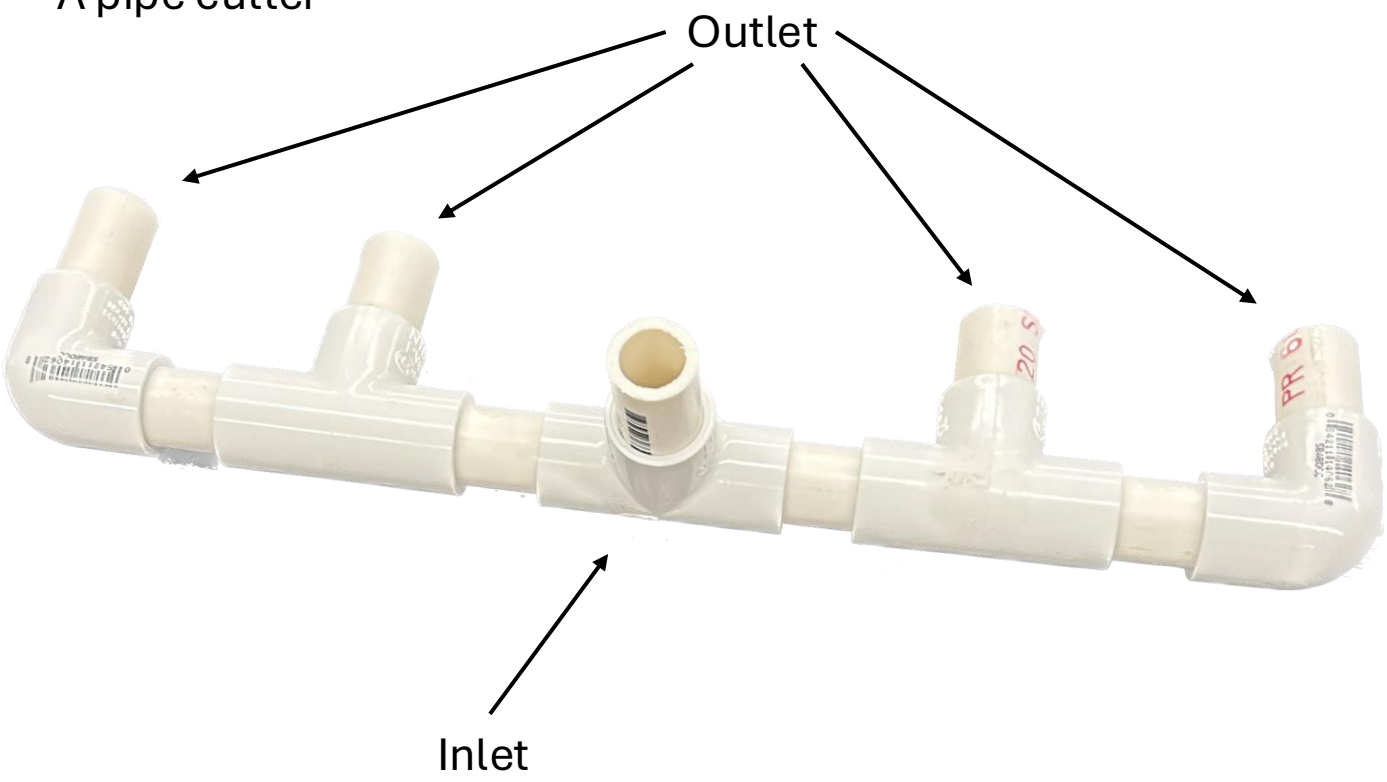
Any plastic shelf kit will work for this system as long as it is 4+ levels and large enough to support the grow beds, reservoir, and nursery.

Constructing the Manifold

Materials

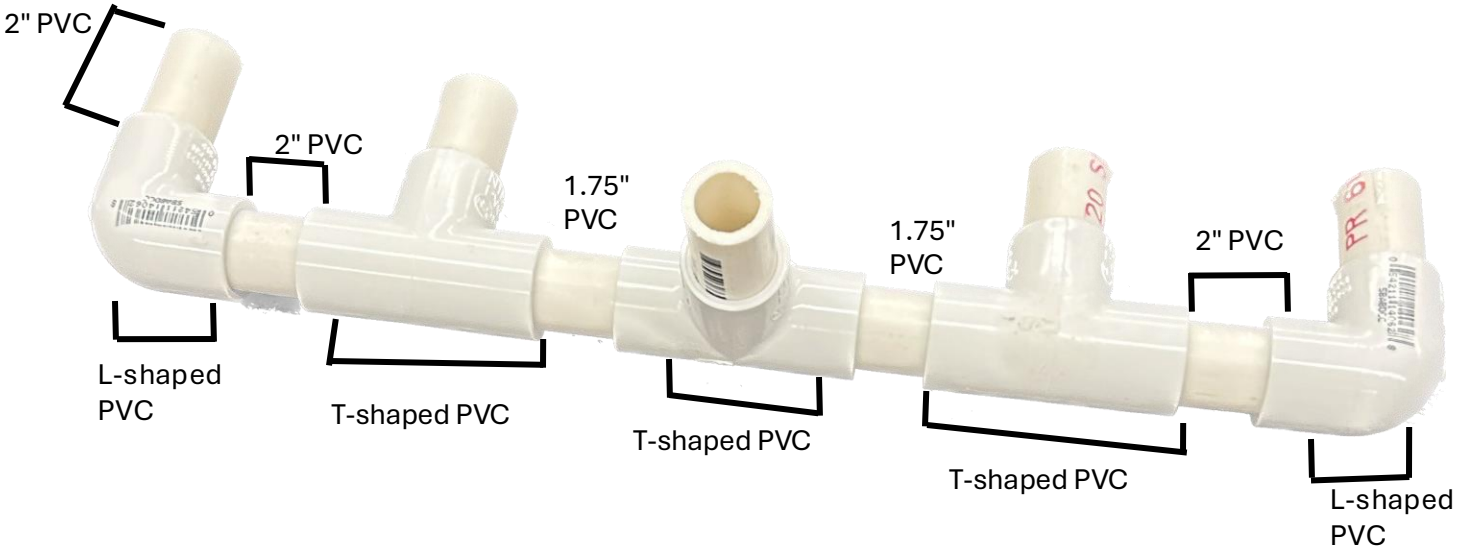
- Three T-shaped PVC pieces
- Two L-shaped PVC pieces
- ½ inch thick PVC pipe
  - Seven 2-inch-long pieces
  - Two 1.75-in-long pieces
- Four barbed connectors
- A mallet
- A pipe cutter

Barbed connector for each outlet



Step 1

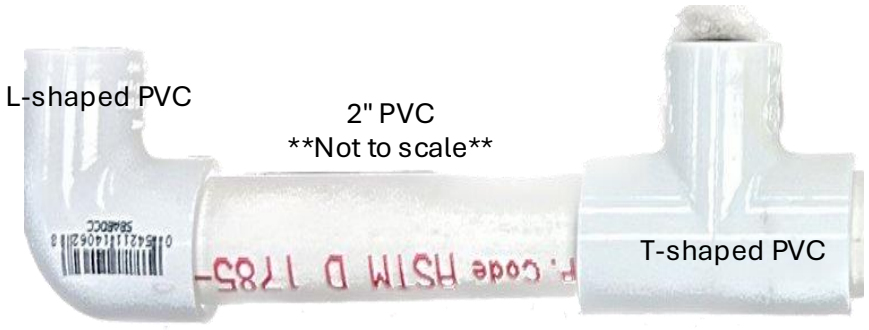
- Cut ½” PVC to the following lengths:
  - 2x – 1.75 in.
  - 7x – 2.00 in.
- Recommended to use pipe cutters similar to the ones pictured at the bottom of this page for safe and clean cutting

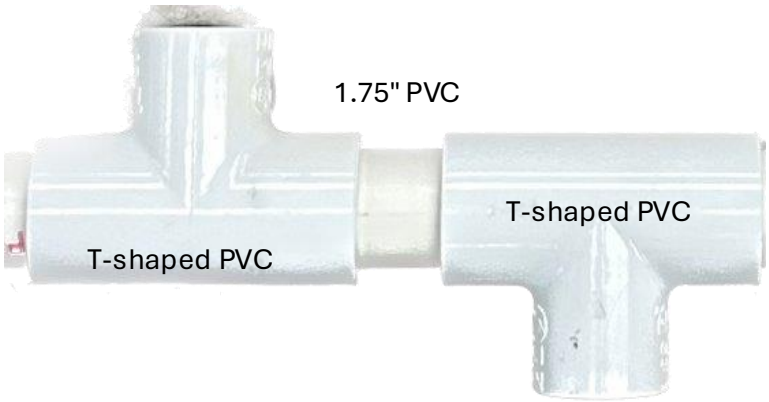


An example of pipe cutters that could be used

Step 2

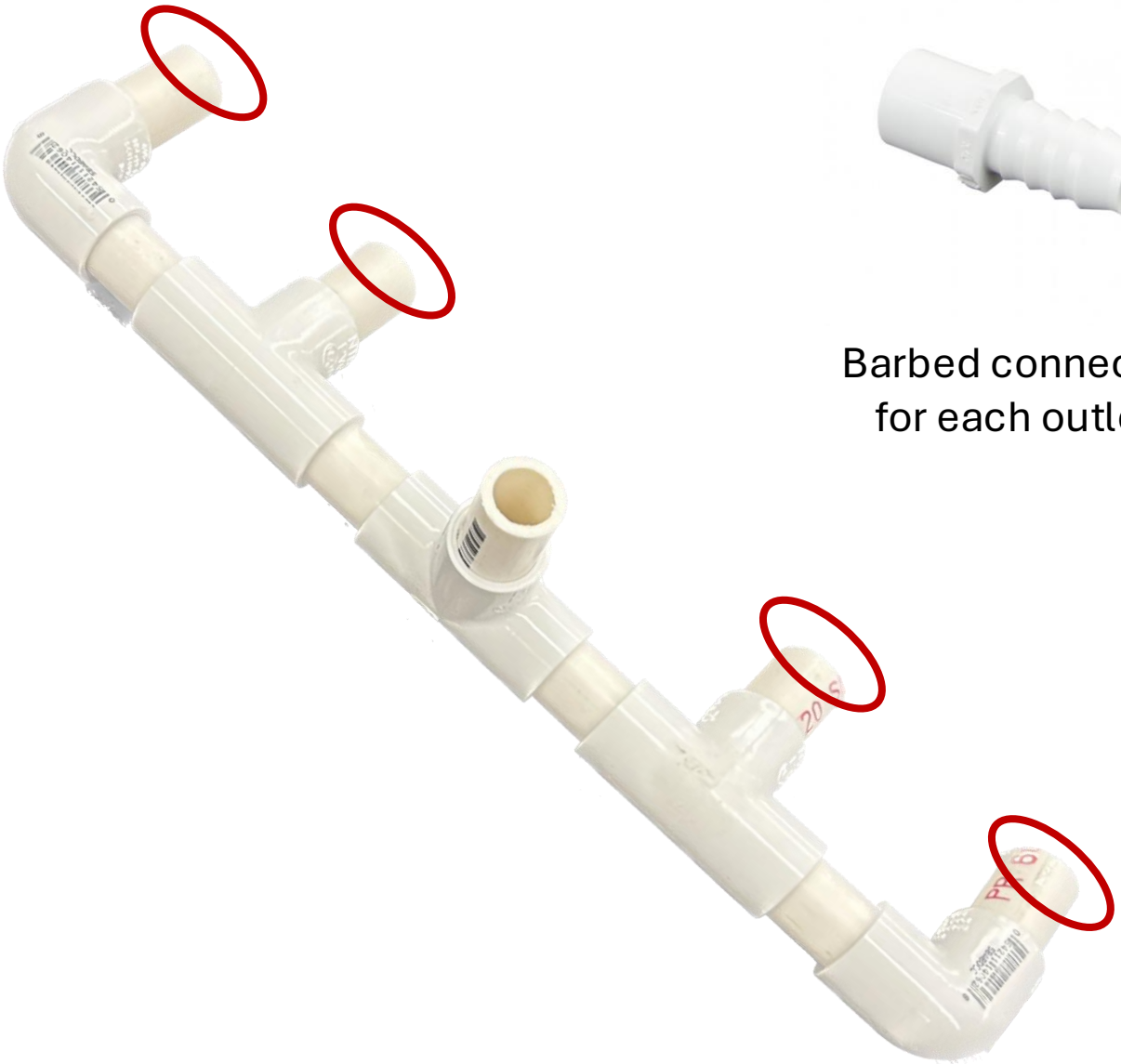
- Connect the 2" PVC in between the T and L-shaped PVC pieces
- Connect the 1.75" PVC in between the T-shaped PVC pieces
- Use a mallet to dry fit straight sections into the T- and L-shaped PVCs
    - Make sure to bottom them
    - Check to ensure they are water-tight





Step 3

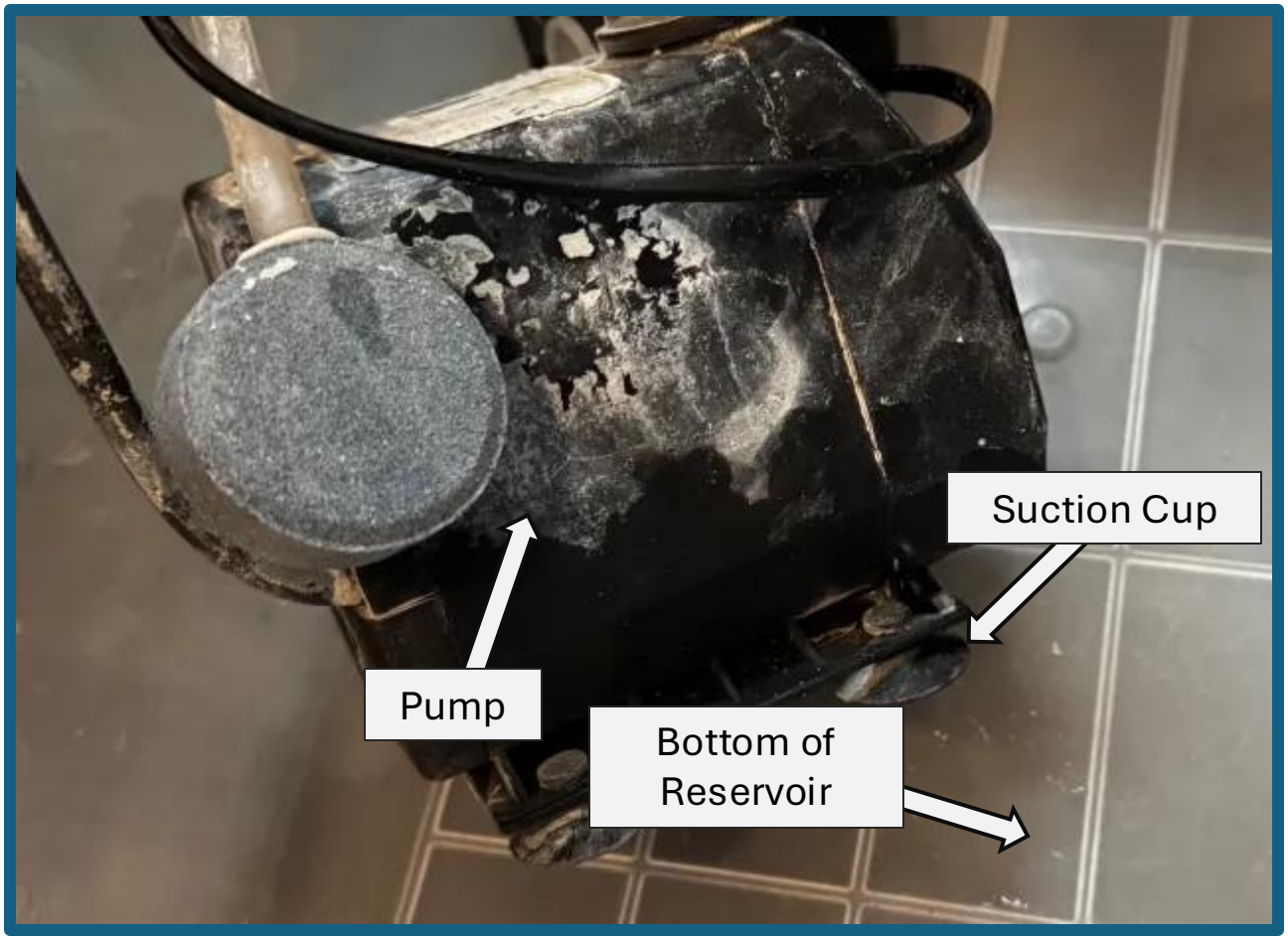
- Attach vinyl tubing connectors to PVC at all open holes



Barbed connector for each outlet

Adding the Pump to the Reservoir

Using the suction cups on the bottom of the fountain pump, place it in the reservoir



Constructing Grow Beds

Materials

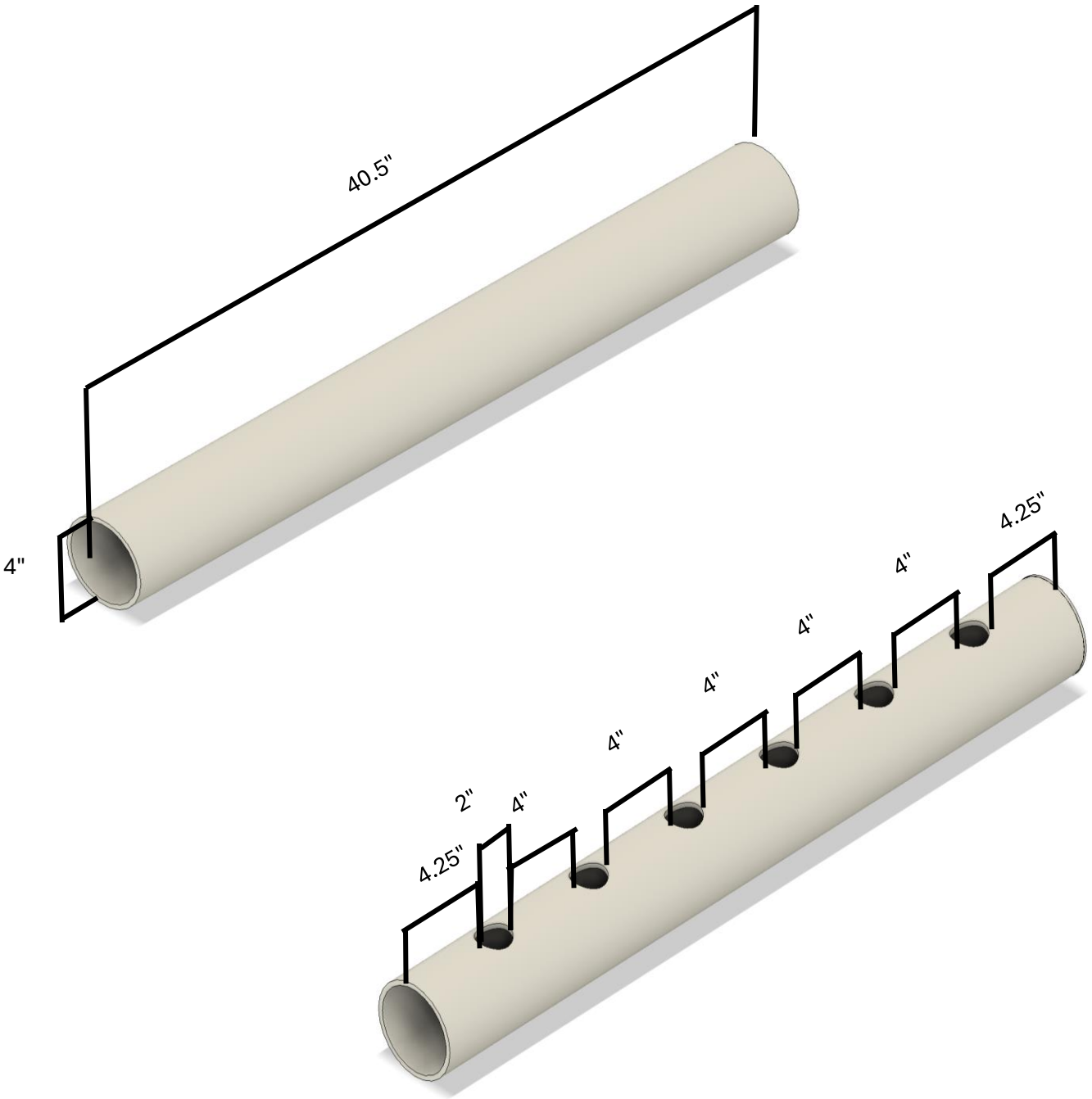
- 1x 40" 4" Diameter SCH 10 (Drainage) PVC Pipe
- 2x 4" End Caps
- 2x ½" Bulkhead Fittings
- 2x ½" Hose Barb Fitting to Male Thread Adapters
- Silicon Food Safe Caulk
- A Drill Press
  - Hand drilling holes on top could result in injury
- Appropriate Hole Saw Drill Bits
  - 2" step bit to drill holes on top to insert net cups
  - 5/8" diameter bit to drill holes into end caps
- Caulk Gun

Additional Notes

- Measure the space available (between the shelf posts) on each shelf to determine how many grow beds can fit.
- Constructing one grow bed will take roughly 45 minutes.

Step 1

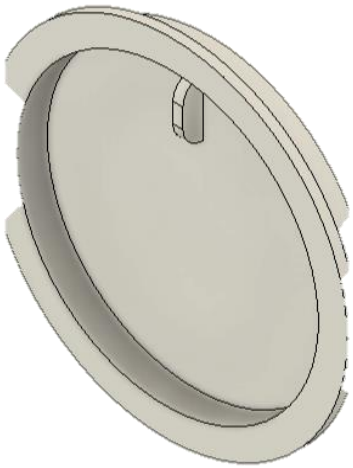
Ensure each grow bed is 40 inches in length. The first hole center should be punched 5.25 inches from the end, with each subsequent hole center punched 6" away from the previous center (4" between the edges of any two holes).



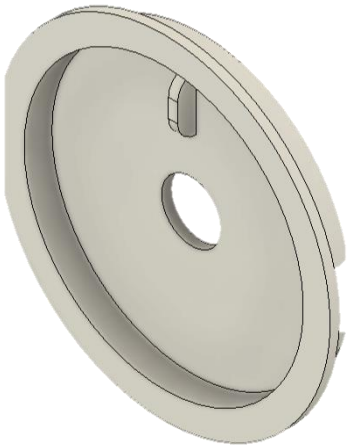
Step 2

Follow the steps below to prepare the grow bed caps. Do the entire process described on this slide twice, resulting in one cap for each end of the grow bed.

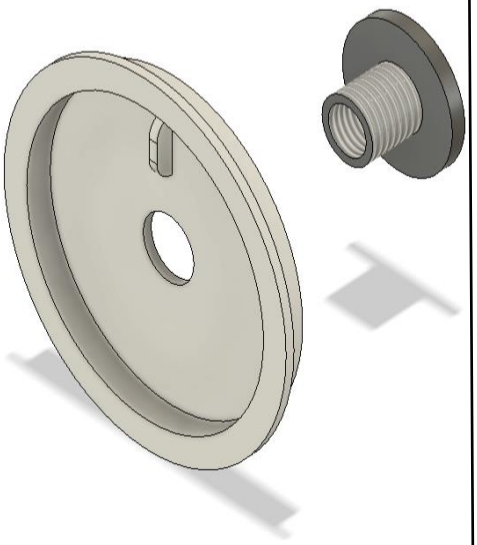
1.



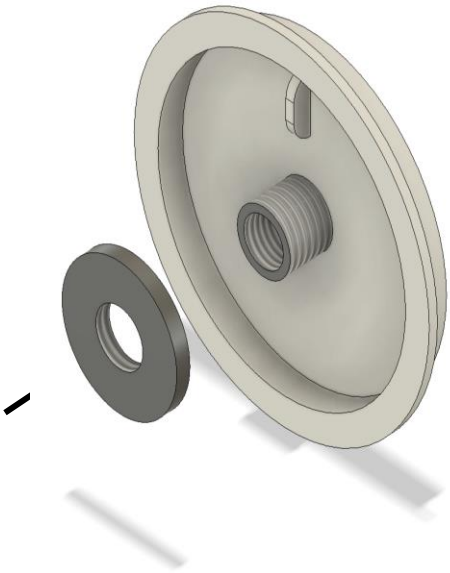
2.



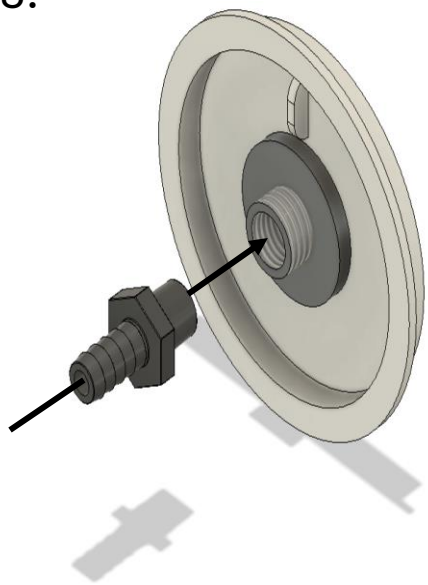
3.



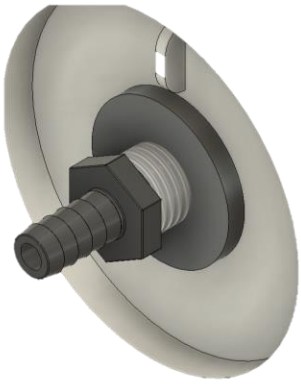
4.



5.

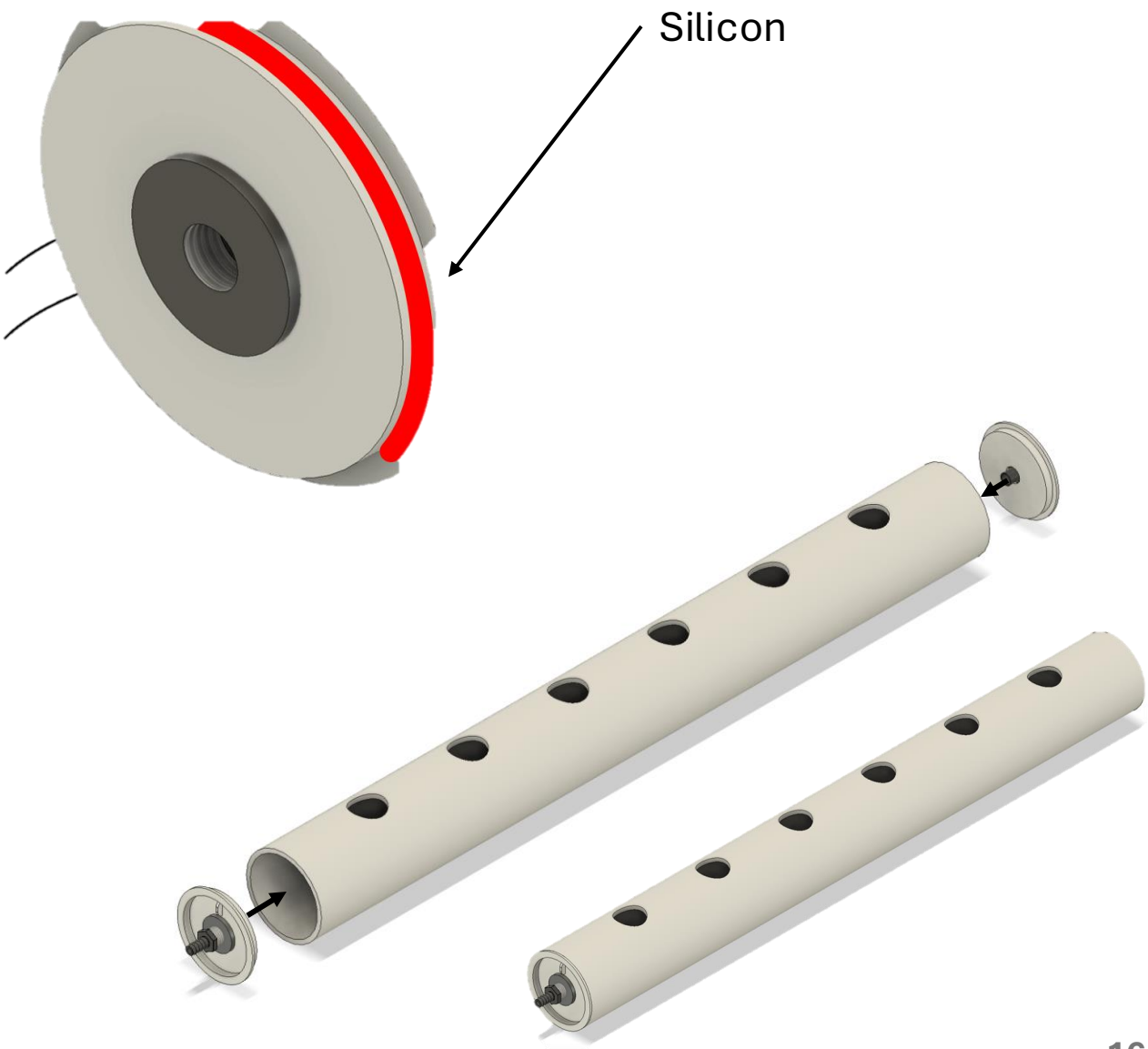


6.



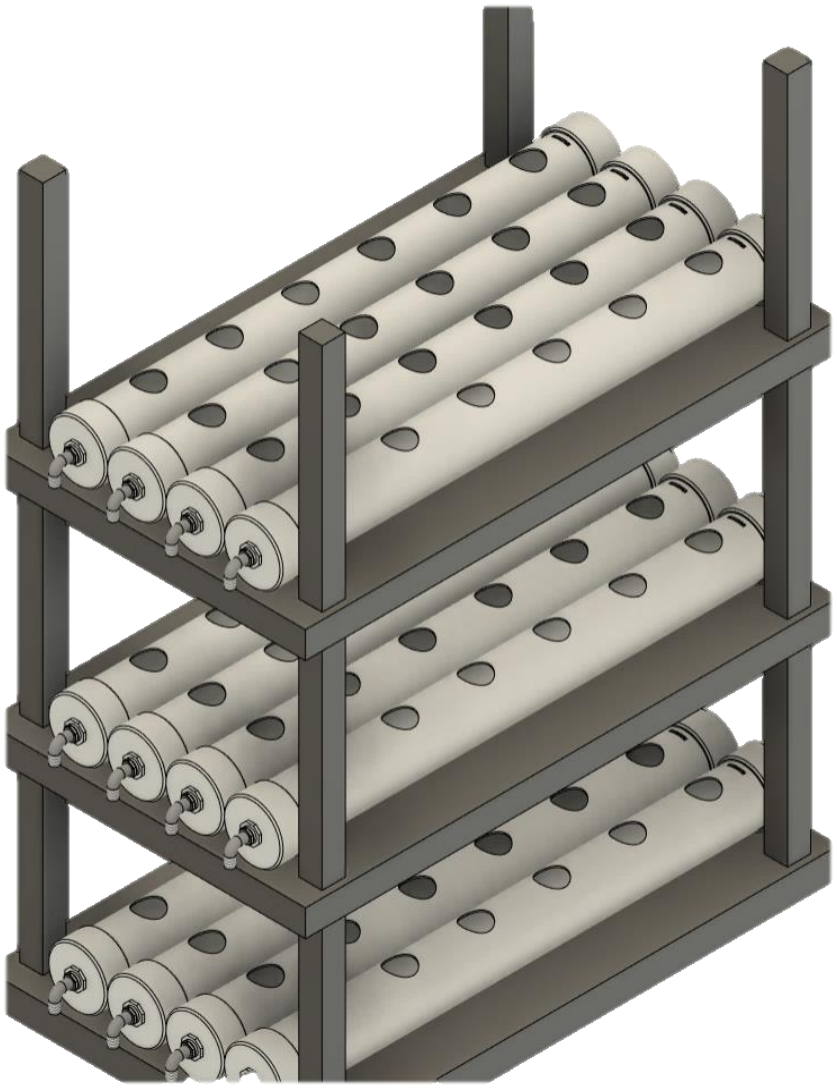
Step 3

Apply the Silicon Caulk to the back of the end caps (as shown in the photo on the left) and attach the caps to the ends of the grow bed. Wait 24 hours for a good seal; then the grow bed will be complete.



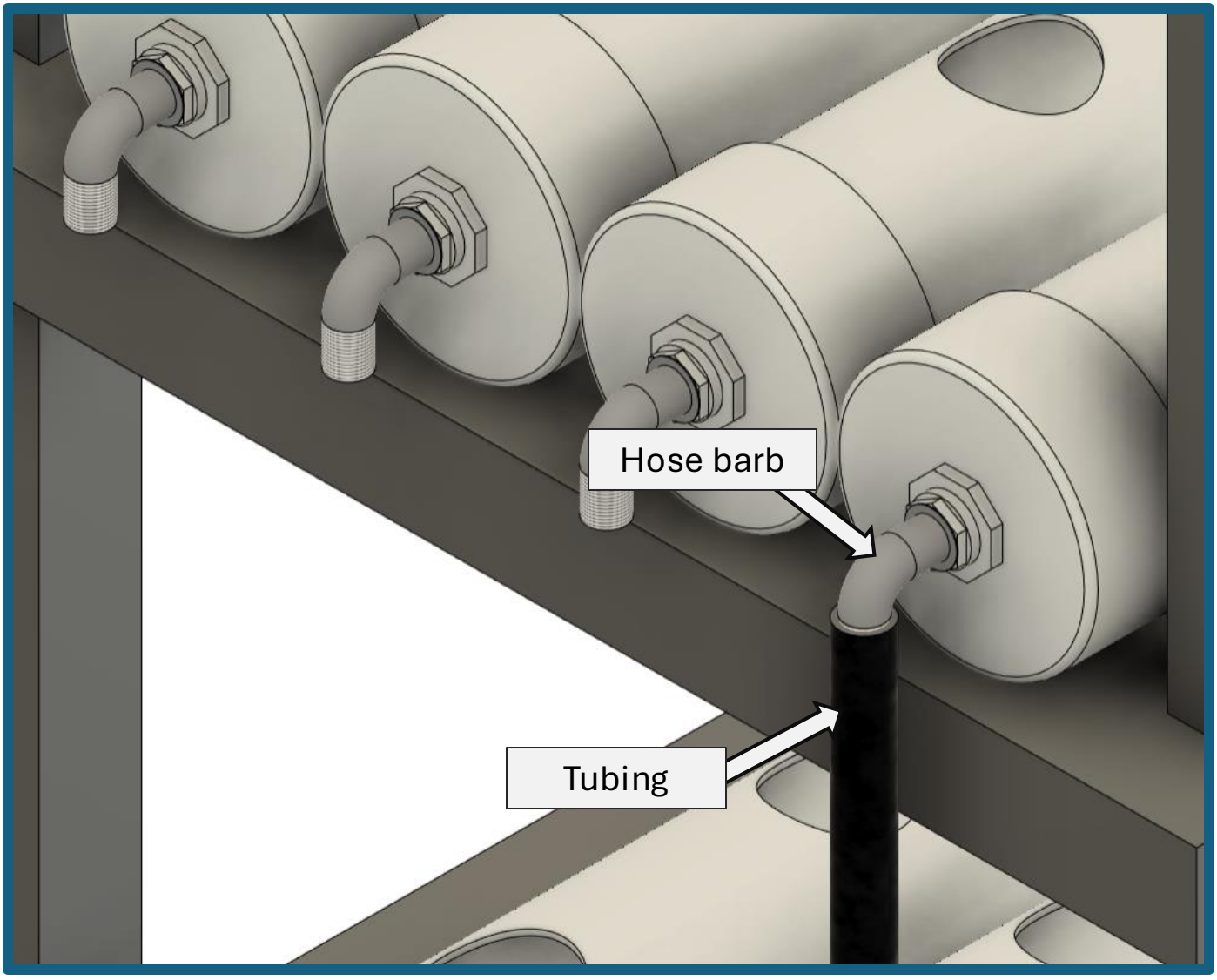
Integrating Grow Beds

- Place four grow beds next to each other
- Repeat this process for all three layers



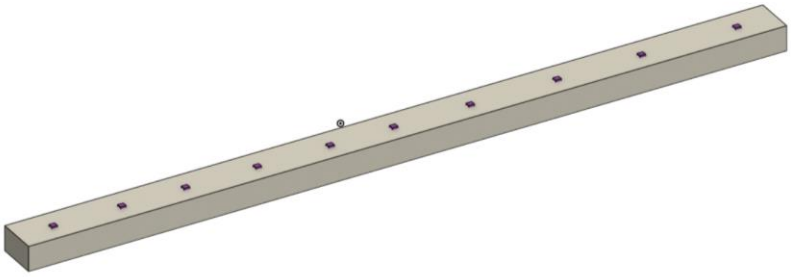
Piping the System

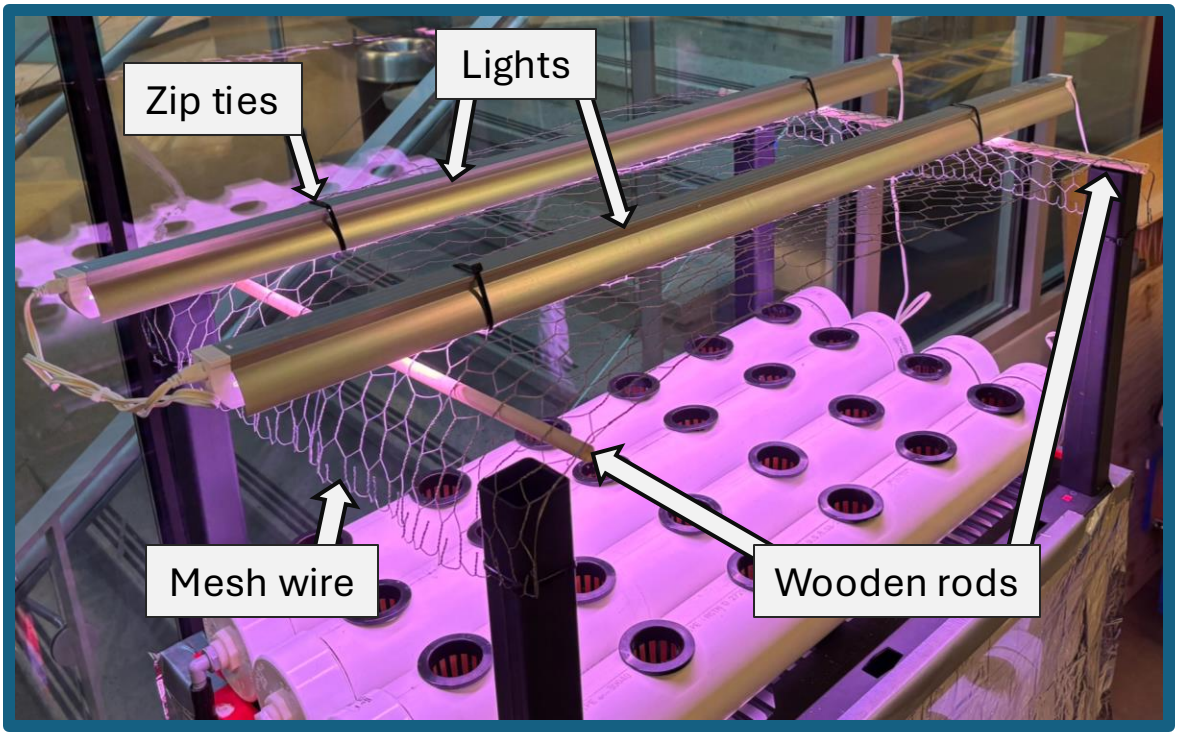
- Using a heat gun, lightly heat the end of the tubing before sliding it over the hose barb to ensure a tight fit
- Repeat this process for all 24 hose barbs



Adding Lights

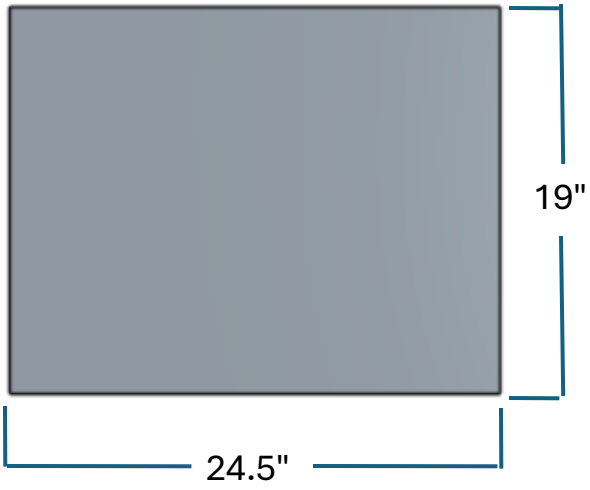
- The LED grow lights are 46", and the lights for each layer should be securely attached to the bottom of the layer above it.
- For the top layer, the lights should be securely placed on wooden rods that are attached to the beams extending up from the top layer. Mesh wire sheets and zip ties may be used to secure the lights more effectively.
- The lights must shine down on the plants from above, and they should emit red and blue light for most efficient energy usage (plants reflect green wavelengths)

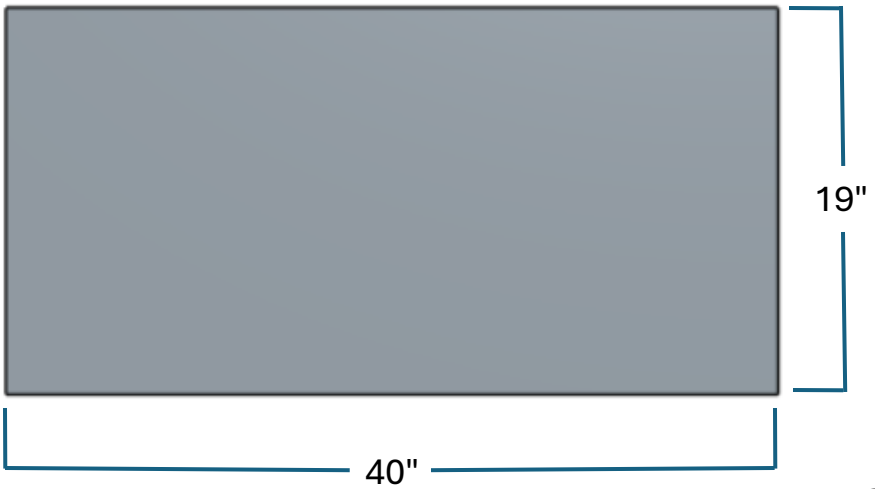




Integrating Mylar

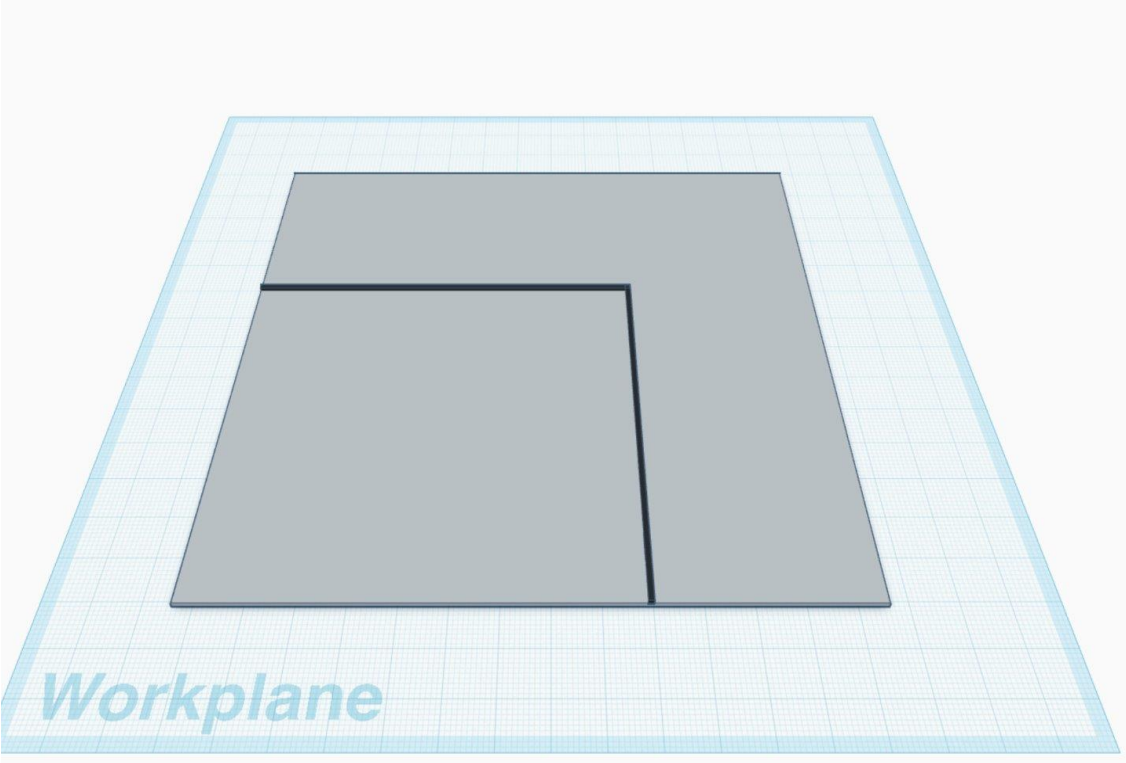
- Add reflective mylar with a thickness of approximately 0.01 mm to the sides of the system. This reflects the escaping light back into the system, increasing yield.
- Cut the mylar sheets into four 40“ x 19" sheets and four 24.5“ x 19" sheets
- Next, place the sheets on the sides of the grow bed layers





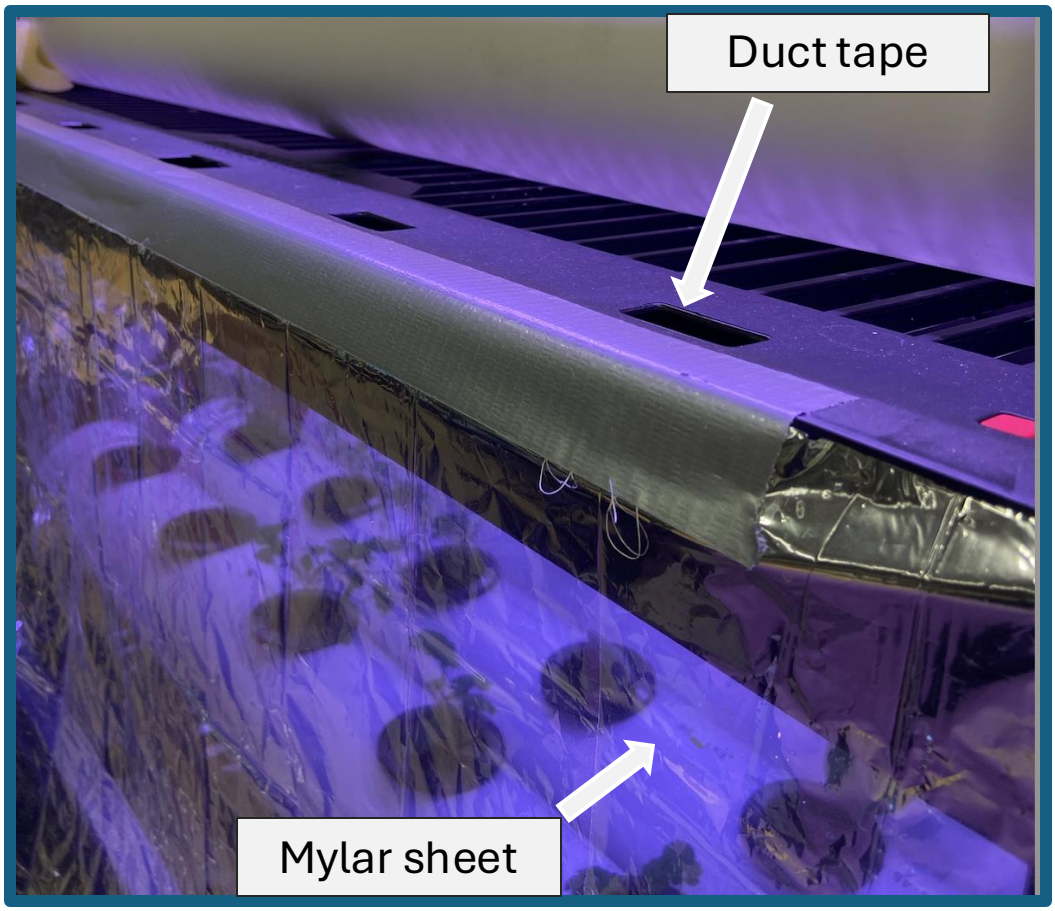
Sketch Out Dimensions

- Stretch out the mylar on a surface and mark dimensions
- 40” x 19” inches for front and back of the system
- 24.5” x 19” inches for sides of the system
- Use a pair of scissors or another tool to cut out the piece.



Attach the Mylar

- Using tape, glue or another medium, integrate the mylar along the sides of the structure
- Make sure that the bottom edges of the mylar sheets are not glued so that they can be lifted to access the interior where the plants are growing
- In the original system setup, the two middle layers are covered with mylar, while the top layer is left open with no mylar for experimental purposes. This setup can be modified for further experimentation or based on personal preference.



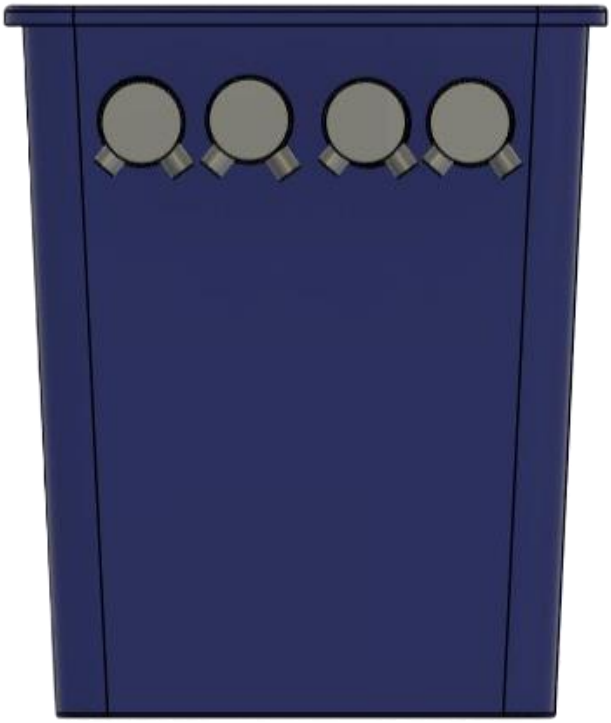
Constructing Nutrient Distributor

Materials List (Per Grow Bed)

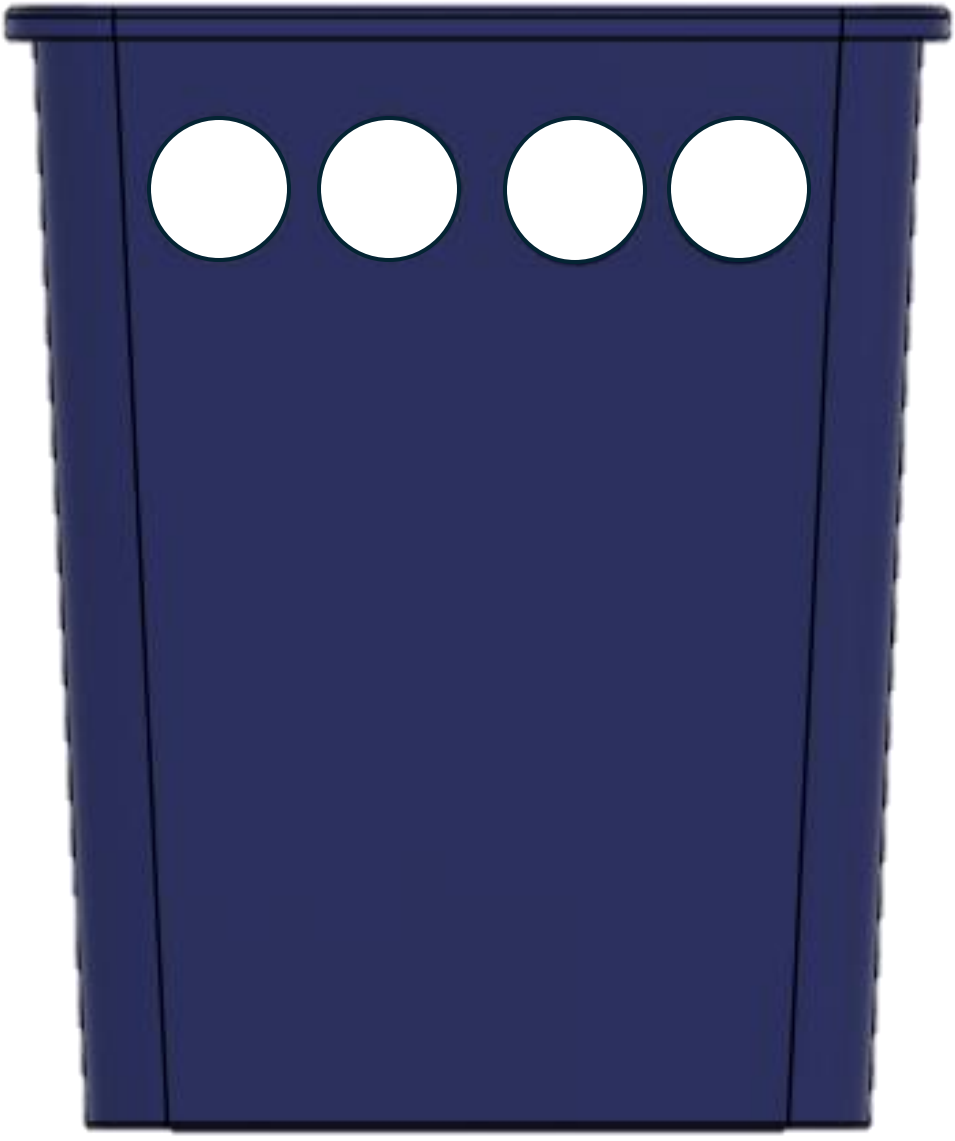
- 4 x peristaltic pumps
- A bin or container big enough to store the nutrient solution and pH controller. We used a spare recycling bin.
- ~6ft of piping, making sure the pipes fit on the motors
- 1 x General Hydroponics FloraMicro
- 1 x General Hydroponics FloraGro
- 1 x General Hydroponics pH UP
- 1 x General Hydroponics pH down

Notes

- Ensure piping is properly connected to each bottle of nutrient solution or pH controller and the motors
- Check the proper bottle is connected to the associated motor
- **Pipes can be fragile, so be careful when adjusting the piping of the system**

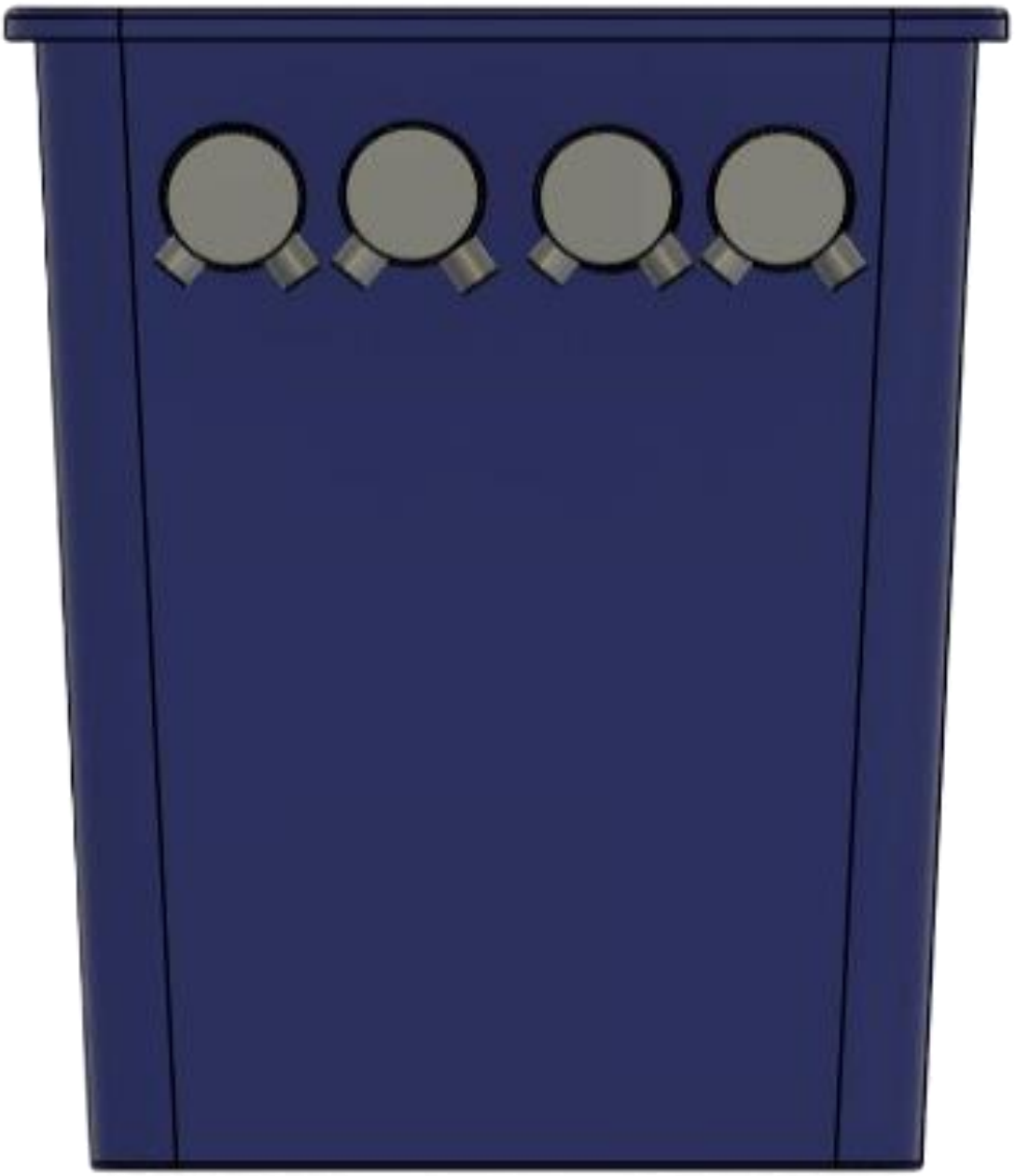


1 Drill four holes into one side of the plastic bucket with diameter equal to your pump diameter. One side of the bucket has been removed in the diagram below for clarity.



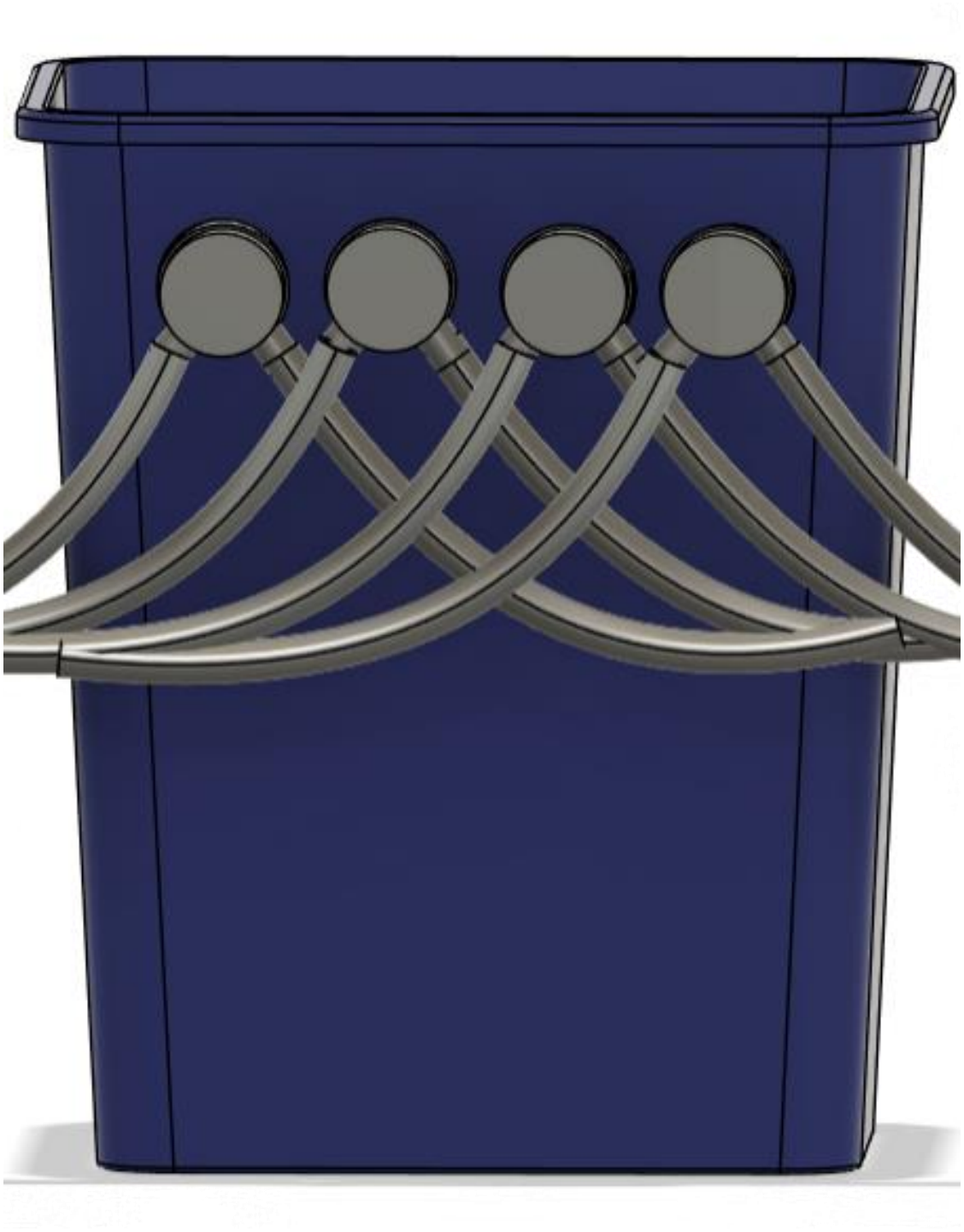
2

Install all 4 water pumps with pipe connectors facing downwards. Your motors should be on the outside of the bucket, with piping on the inside.



3

Connect rubber tubing to the pipe connectors by inserting tubing into the open slots in the connector.



Planting in the System

Materials

- 2" plastic net cups
- Rockwool Cubes
- Seeds of the desired plants (about 3 seeds each cup for large seeds and about 5 seeds each cup for smaller seeds)
- A pencil or pen (or something with a pointy edge)
- Gloves
- Properly functioning hydroponics system
    - **Make sure water will not overflow**
- Nutrient solution has been connected to electronics system
- Adequate lighting/ lighting has been turned on
    - **Make sure to wear gloves when handling Rockwool**

Additional Notes

When choosing seeds to plant in the system, consider whether the plant can properly survive within the conditions of the hydroponics system. Additionally, ensure each species grows within the same system settings. It takes an estimated 15 minutes to finish planting a layer.

Instructions

- Rockwool, desired seeds, gloves, a pencil/pen (something with a pointy tip), net cup
- Take a cube of rockwool and place it at the bottom of the net cup
- Take the pencil and make sure the pre-made hole is open
- Place the seeds inside the hole and use pencil/pen to scratch the rockwool surface to close the hole up
- Place the net cup into the system, making sure the rockwool is touching the water.
- Your plants are now ready to start growing!

Harvesting

In order to harvest the system between plant cycles,

- You can use gardening shears or scissors to cut the leaves.
- You can also harvest by cutting at the stem, but it is not advised as will take longer and be less effective.
- You can expect around 5-6 plant harvests before re-planting and cleaning the system.

Cleaning the System

To clean the system between plant cycles:

- **TURN PUMPS OFF BEFORE CLEANING**
- Take out all plants/net cups
- Use either Hydrogen peroxide or Vinegar (Hydrogen peroxide is more commonly used to clean out algae, but vinegar can be used as well.)
- 1:1 solution of 3% concentration of Hydrogen peroxide and water
- 1:1 solution of 5% concentration of white vinegar and water

Start of Year Maintenance

This section will go into the steps needed to set the system up at the beginning of the year to prepare for the next grow cycle. If you're looking for troubleshooting advice, that can be found in a later section. (This will also be noted throughout, however, as you complete these steps keep in mind: for experimental purposes, the pH and nutrient levels are customizable. Listed in these instructions are a good setup for a control group for standard plants such as lettuce and various herbs.)

1 Preventative Leak Testing

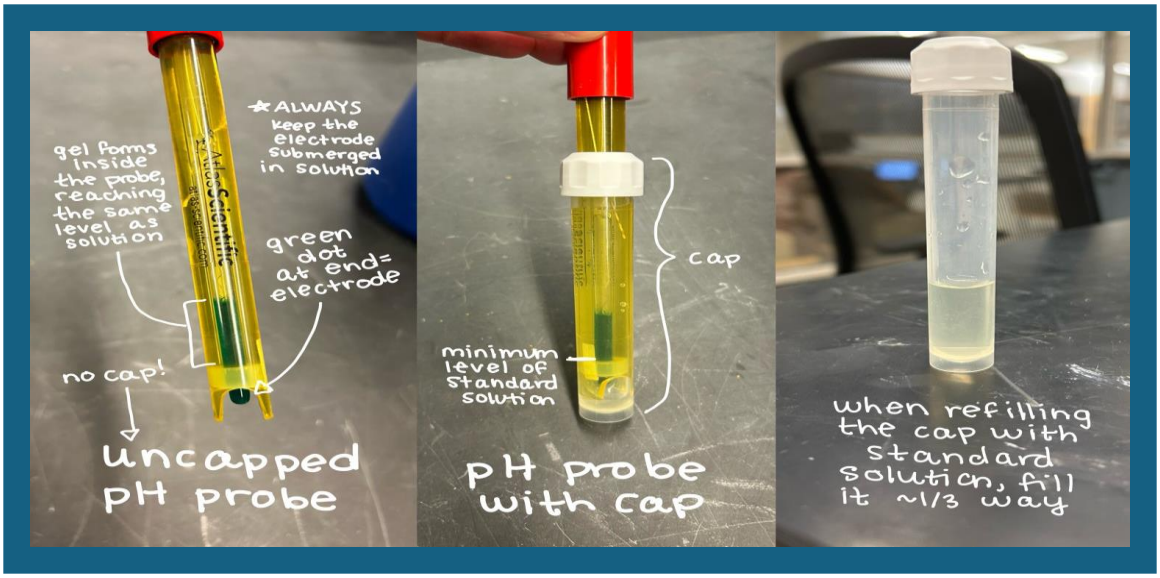
- Detach a grow bed from the system by pulling the tubing out of the connector gently (it may require a little bit of force).
- Take the grow bed to a sink and place a small amount of water inside the tube, slowly tilting the grow bed back and forth to see if any leaks are present. Ensure the large holes at the top remain upwards.
- Repeat with all grow beds. If there are any leaks, see the troubleshooting portion of this manual. If there are no leaks, reattach the grow bed to the tubing, double-checking for a strong connection to prevent leaks at the source.

2 Fill the Reservoir with Water

- Remove debris and dirt from the reservoir basin as needed.
- Plug the manifold into the water pump and ensure all outflow pipes from the lowest grow bed are within the reservoir.
- Gently pour water from a separate container into the reservoir until it is about ¾ of the way full.

3 Sensor Testing

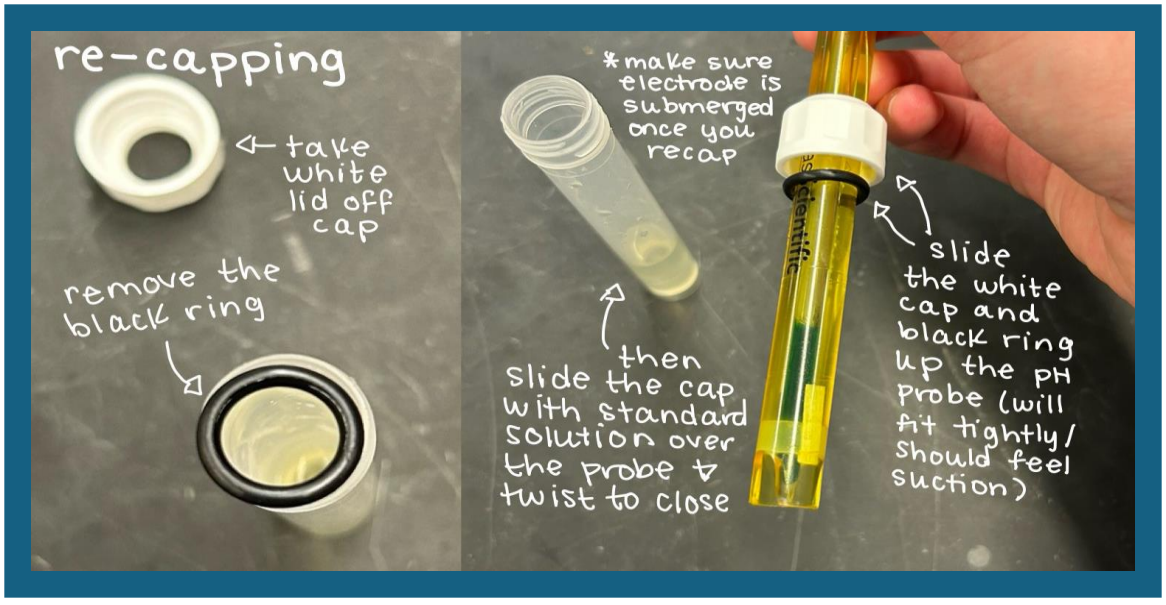
- Open the housing box and ensure there is no debris or liquid inside of the box
- Plug in the housing box and make sure the Arduino Giga display turns on and starts displaying values
- Using a pH strip, test the pH of the water in the system
- Uncap the pH probe slowly by gently twisting and pulling on the suctioned out covering
- Place the pH and conductivity probe in the reservoir
- Observe the displayed values, if they meet the following criteria, you do not need to recalibrate the sensors:
  - The displayed pH matches, or is within 0.5 of the measured pH using the test strip
  - The displayed nutrient values are between 30 and 1500
- If one of more of the criteria above is not met, proceed to the second part of this section (if everything is met, move to step 5)



Calibration: pH Sensor

- Some brief notes beforehand:
  - When the pH probe is not taking measurements in the reservoir, make sure it is always capped (shown in the diagram above), and the cap is filled enough to submerge the tip of the probe in standard solution to keep it healthy.
  - If the probe has not been properly soaking, refill the cap with standard solution and let the probe soak in the fluid for at least 4 hours.
  - Always check the expiration dates of your test solutions prior to completing a calibration.
  - When rinsing the probe, do not dip the probe into clean water to wash it off. Pour clean water over the probe into a waste container.
  - Calibration can be a two-person process, so be sure to have extra hands available.

Complete the steps carefully! It's a much easier process if you don't spill solutions or damage the sensors.



To calibrate the pH sensor, you will need to complete the midpoint, low point, and high point calibrations separately. This process works best if one person oversees holding the sensor/calibration packets and preparing the calibrations, while the other types commands into the computer.

- Secure a charged laptop/computer with an available USB-C port and obtain a USB-C cord:
    - Plug the USB-C cable into the Giga, as shown on the next page and open Arduino IDE
- Gather the following materials:
    - Deionized or filtered water
    - 2-3 small containers
    - Paper towels
    - Calibration solutions (Mid: 7, Low: 4, High: 10)
- Calibration:

1. Label one of the empty containers as “waste”, and the one filled with water as “clean”.

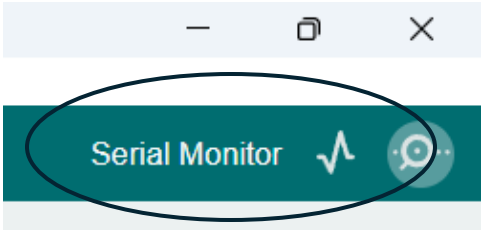
2. Gently take off the cap of the pH probe. The cap will already be removed if the probe was taken from the reservoir.

3. Rinse it with a small amount of clean water and blot (gently press) the electrode tip and rest of probe with a paper towel.

4. Open the midpoint calibration solution with a pair of scissors and submerge the relatively dry probe in the pouch.

5. Swirl gently and then allow the probe to sit until the reading on the computer stabilizes.

6. Once the reading is stable, calibrate the sensor to the correct pH by opening the “serial monitor” tab shown above.



Calibration Commands:

- Mid Point: “ph,cal,mid”
- Low Point: “ph,cal,low”
- High Point: "ph,cal,high"

7. Type the corresponding calibration command from the next page into the white box beneath your code in the serial monitor command window.

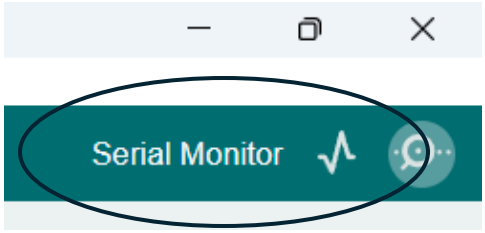
8. Enter the calibration command for whichever calibration you are currently performing, and wait for a response back, stating that the command has been accepted and ensure that the reading now matches the command you just entered.

9. Repeat steps 3-8 for the low and high point solutions to complete calibration.

Troubleshooting Sending General Calibration Commands

- If you forget the calibration command, or are unsure what commands you have available, type “help” into the serial monitor
- If the calibration command doesn’t go through, an error message will be thrown:
  - Follow instructions in the error message to fix your command or reupload your code if it is not specific enough.
  - If the Arduino starts flashing red, press the button on the inside of the housing to restart it.
  - If the calibration commands/code is still not functioning, restart your computer.
- If you accidentally type the wrong calibration command, you will need to restart the calibration:
  - Type: **“ph,cal,clear”** into the serial monitor.
  - Wash off the probe and restart calibration.

  For any other issues, please refer to the Atlas Scientific website linked at the end of this guidebook.



Calibration Commands:

- Low Point: “ec,cal,low”
- High Point: “ec,cal,high”

Calibration: Nutrient Sensor

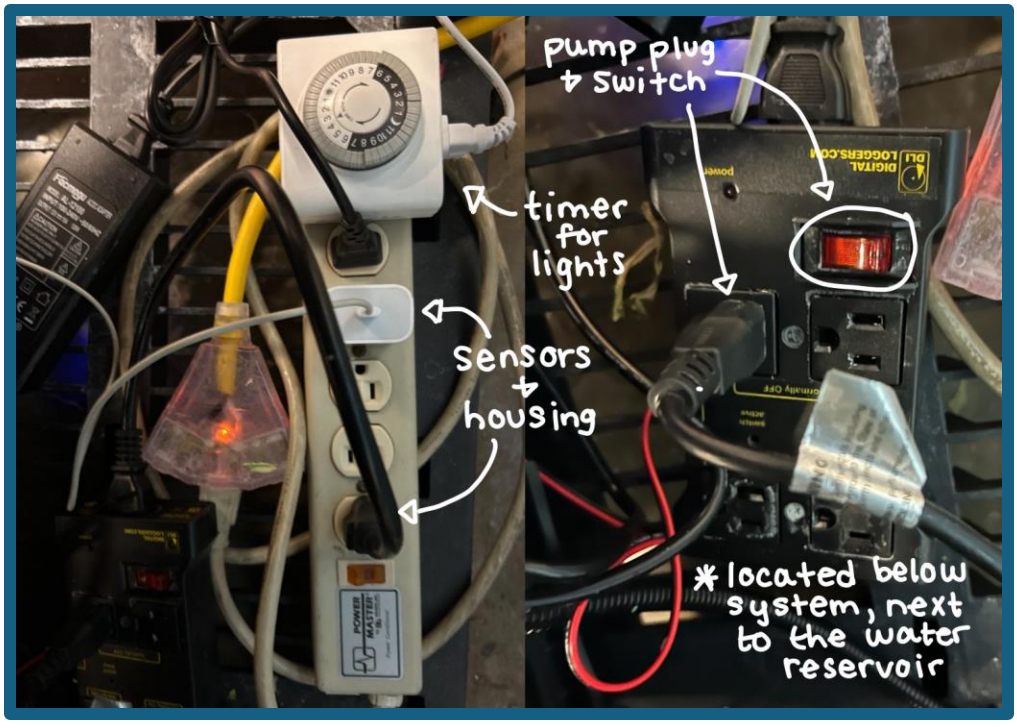
- Secure a charged laptop/computer with an available USB-C port and obtain a USB-C cord:
  - Plug the USB-C cable into the Giga and open the serial monitor in Arduino IDE to read values from the sensor.
- Gather the following materials:
  - Conductivity calibration solutions (12,880 µS and 80,000 µS)
  - Two medium-sized containers, fill one with clean water, and label the other one as waste
  - Paper towels
- Calibration:

1. Pour clean water onto the probe to gently rinse it.
2. Gently blot dry with a paper towel and let it sit on a clean, dry surface. Allow the probe to stabilize.
3. Once the probe has stabilized, begin calibrating with the 12,880 µS solution.
4. Open the solution pouch and submerge the sensor in the solution. Wait for readings to stabilize.
5. Enter the calibration command for the low point.
6. Repeat steps for the high point calibration.

5 Allow the System to Run

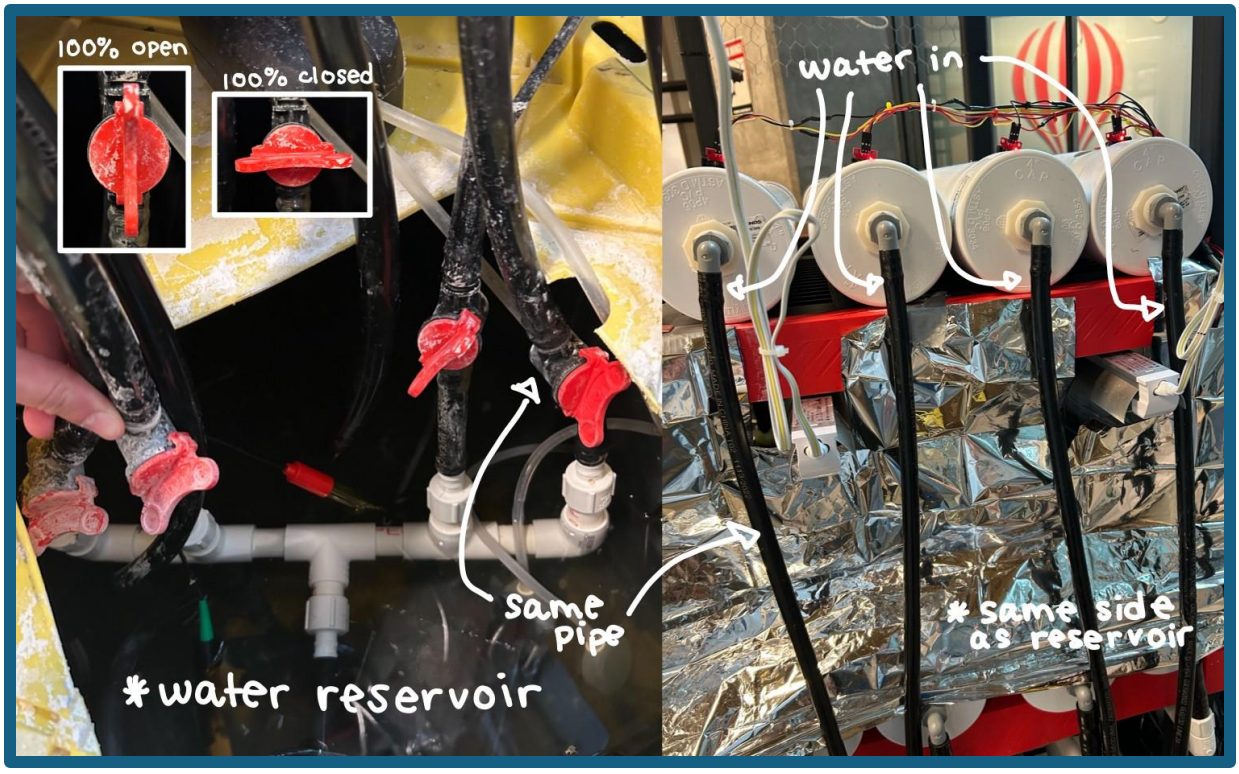
To begin, plug the main surge protector into an outlet. This should turn on the system's pump.

- Allow the system to run for a couple hours, watching closely as each layer fills:
  - If the pump does not run, check underneath the system (shown below), to make sure it is plugged in and the surge protector is switched on.
  - Make sure the light timer control and system electronics are plugged in and set to the desired setting.
- If a layer is doing any of the following, consult the trouble shooting guide on the next page:
  - Filling unevenly (one is filling much quicker than the others)
  - On the verge of overflowing
  - Not filling at all



Troubleshooting Pump Problems:

- If the layers are filling unevenly or not at all, check the valves inside the reservoir pictured below):
  - Adjust the power as needed by turning the valve in either direction.
  - Turning the valve closer to horizontal will reduce the flow of the water and turning it to be more vertical will increase the flow of the water.
- If you turn the valve and the water flow does not change in the respective grow bed layer, there could be something blocking the tubing/stuck in the grow bed layer:
  - This could possibly be due to a buildup of algae. Refer to end of year maintenance on guidelines for cleansing and removal.
- If no water is flowing through the system at all, double check to make sure the pump is plugged in.



6 Get Planting!

Before you begin planting your seeds, ensure that they are suitable for growing in a hydroponics system, keeping in mind the space and sunlight restrictions of the system.

IMPORTANT: make sure the water level in the system has stabilized completely and remains unchanged for at least 10-15 minutes.

- While you wait, gather the following materials:
  - 72 net cups
  - 72 rockwool clumps
  - Seeds of your choice (can be many different kinds, as long as they are all within the same pH growing range
  - A pencil or other thin, slightly sharp object
  - Gloves (gardening or lab safety)
- Ideally, it is best to let your rockwool soak in slightly acidic solution for about an hour, so you that they are fully moistened before you place the seeds:
  - Find a large bucket and fill with tap water, placing the rockwool pieces into the water using gloves.
- One the rockwool has soaked, place each clump into a net cup, using the pencil to lightly push it down until the rockwool is resting on the bottom.
- Next, poke a small hole with the pencil, reaching about halfway down the rockwool.
- Place 3-5 seeds in the small hole, using the pencil to ensure they have found the bottom of the hole.
  - You can place more seeds if they are small or have low germination rates.

- Lastly, scratch the rockwool above the seeds (do not push them down any further), using the pencil.
    - Be sure to label your net cups after you've planted a seed.
    - Your net cup is now good to go and can be placed in the system in one of the 72 open holes!

***Pictures corresponding with each step can also be found on the next page for further clarification**

Visual Guide

End of Year Maintenance

This section will go into the steps needed to clean and shut off the system for an extended period (such as summer break) after all desired plants have been harvested. End of year maintenance is especially important as it makes set up the following fall much easier and prevents mold from growing in the system while it is not in use.

1 Remove the Plants and Net Cups

- After harvesting all desired plants, begin removing the remaining net cups from the system
  - Rockwool can be reused but leaving it in the system unattended can promote mold growth. For more information on how to properly reuse rockwool, consult the back of the manual.
- Dispose of the rockwool inside of the net cups and gently rinse the net cups to get rid of any remaining nutrient or pH altering chemicals.
- Allow net cups to dry, then store.

2 Inspect the Grow Beds

- Using a flashlight, check the grow beds for any signs of chemical build-up, algae or mold:
  - Algae is the most common buildup within the system.

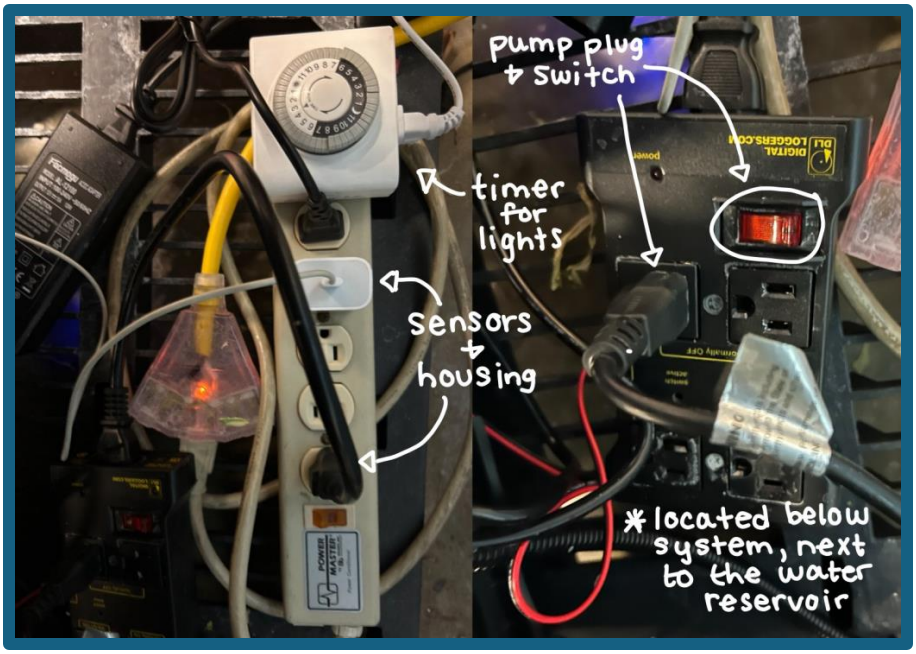
- If white specks are present in the water in the water during maintenance or the grow cycle, it is likely due to a chemical imbalance of FloraMicro and FloraGrow and can be remedied by decreasing the amount of nutrients being pumped into the system.

- If you spot any algae build-up, you can treat it in one of two ways (for cleanup purposes, you'll only need to use the first)
  - Measure out slightly over a liter of 35% food grade hydrogen peroxide and pour into the reservoir
  - For preventative measures, that is if you notice algae growing during the middle of a grow cycle, you can add a small amount of hydrogen peroxide to the (just be sure to monitor the vitals of the system for the next couple hours)
  - After both methods, allow at least 48 hours of running the hydrogen peroxide through the system to see results. If you are cleaning the system for end of the year maintenance, be sure to remove any remaining algae with a scrub brush AFTER turning off the pumps.

3 Power Off System Components

- Turn off the pump using the switch, or simply unplug it
- Unplug the main power strip from the wall (this will power off the housing, the lights and any other components that are supplied power from the housing box
  - Arduino should power off, along with display

- Make sure all water in the system has stopped flowing before proceeding with any further cleaning steps to avoid overflowing the grow beds



4 Clean the Grow Beds

- Once the pumps and electronic components are all turned off, begin draining the grow beds
  - Make sure that all tube connections are secure before proceeding
  - Allow each grow bed to drain until no water is flowing into the reservoir
  - Empty the reservoir first (take water out of the reservoir using a pitcher, until you can comfortably carry it to a sink to empty the rest of the way)
  - Once the reservoir is empty, begin tilting the top grow bed layer towards the drain tubes on the right of the system, causing the water to flow down to the second grow bed (monitor water level as to not flood the second grow bed layer)

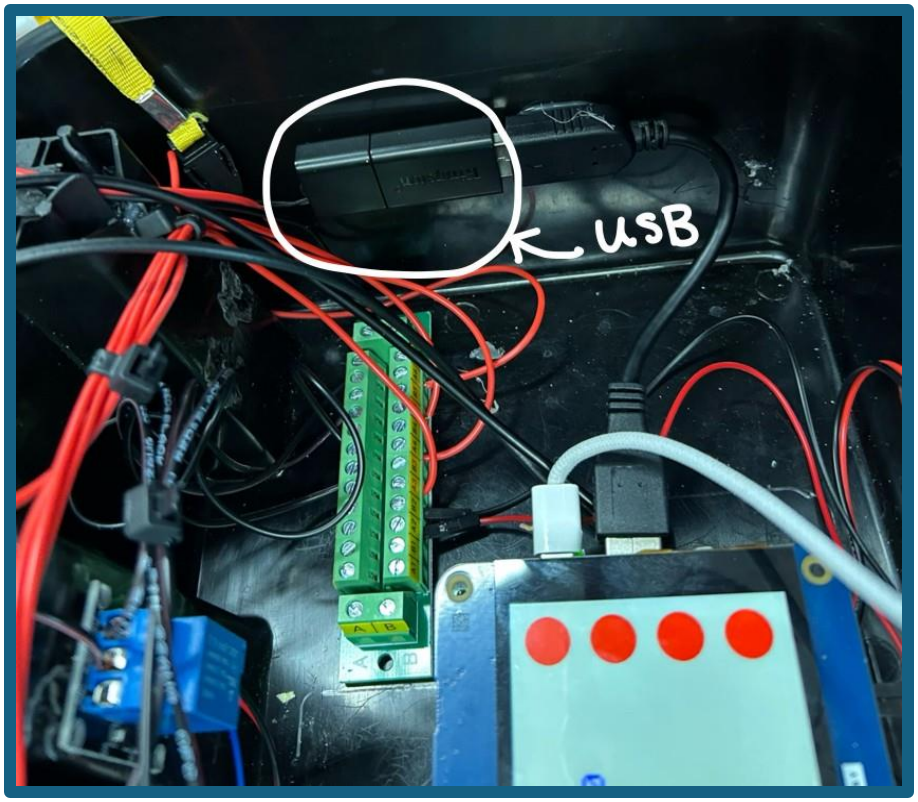
    - Tilt the second grow bed layer towards the drain tube on the left, causing water to flow to the first layer
    - Repeat, tilting the first grow bed layer to the drain tubes on the left causing the water to then flow back into the reservoir
    - Continue this process until there is not much water left in all the grow bed layers
- Remove the grow bed layers one by one to empty out the remaining water in the system, and then empty the reservoir the rest of the way
- Gather a scrub brush, paper towels and hydrogen peroxide diluted to 3% concentration (you can consult an online dilution calculator to figure out how much water you need to add, or turn it into a fun learning experience doing the calculations by hand)
    - Bring each grow bed layer over to a sink or other water source
    - Using the scrub brush, remove any debris or algae
    - Use the diluted hydrogen peroxide to remove any “gunk” that may still be stuck to the pipes
    - Rinse thoroughly with water and place with the holes over a sink (or slightly elevated on a clean surface) to dry completely
- Place grow bed layers back on the system
- Dispose of hydrogen peroxide down a sink drain, being sure to run water as you pour to dilute the solution further

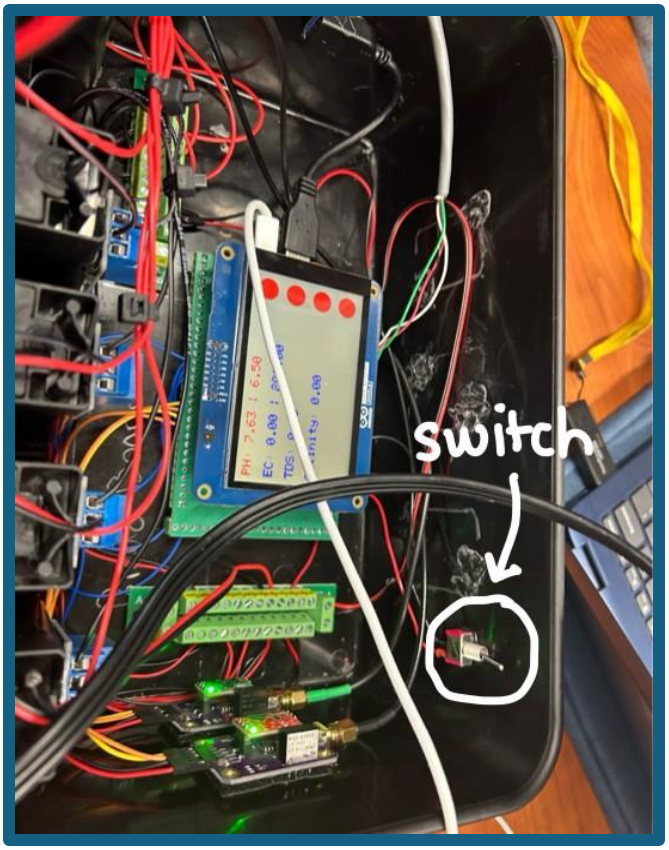
5 Turn Off Electronics

- Unplug the USB from them port (located on the left-hand side of the housing wall) and insert into computer, a file should then immediately open with all the data stored on the USB from the current grow cycle
- Recommended: save the CSV file using whatever software you have access to (likely Google Sheets, you may have to install the desktop version/app for this to work), once saved, you can then delete the data off the USB and reuse it next grow cycle
- Store the USB in a safe space where you'll be able to locate it again next school year
- Once you've ensured the data is saved, turn off the display using the switch and unplug the housing from the wall outlet, and double check to make sure that the lights on the Arduino Giga have turned off
- Allow the housing to cool for a brief moment (~ 5 minutes, if it doesn't feel warm at all, don't worry about letting it sit), and then gently dust off the components and the inside of the housing box with a microfiber cloth, if there is any debris or contaminants, clean the affected area with a cotton swab/paper towel and concentrated isopropyl alcohol (NEVER use a damp cloth or water-based solution, it will damage the parts)
- Close the housing box and place in a room temperature, dry area

***Visual Guide for the USB and Switch are on the following page**

Visual Guide





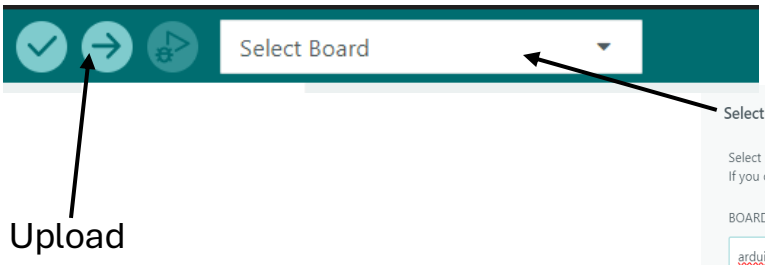
Downloading Arduino IDE

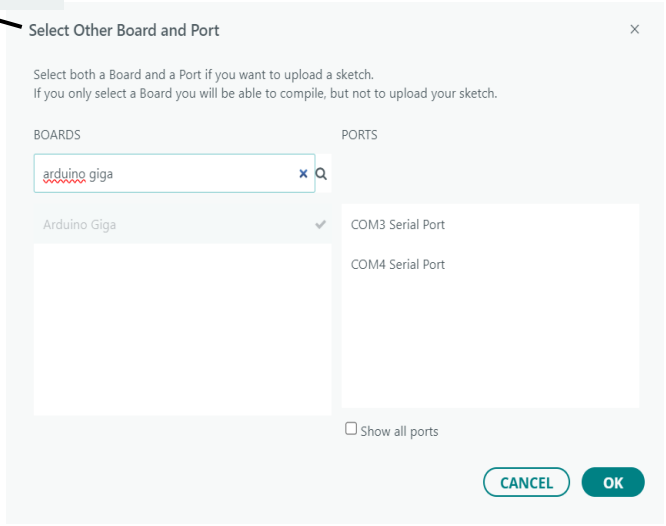
1 How to download Arduino IDE

- ❑ https://www.arduino.cc/en/software/
- ❑ Choose your platform:
  - ❑ Windows: Download the installer or the ZIP file
  - ❑ Mac OS: Download the zip file and move it to applications
- ❑ Follow the instructions on the screen

2 Connecting to Arduino

- ❑ Plug in your Arduino using a USB cable
- ❑ Open Arduino IDE
- ❑ Go to Tools > Port and select the correct port
- ❑ Select the correct board under Tools > Board
- ❑ Click Upload (→) to send your code



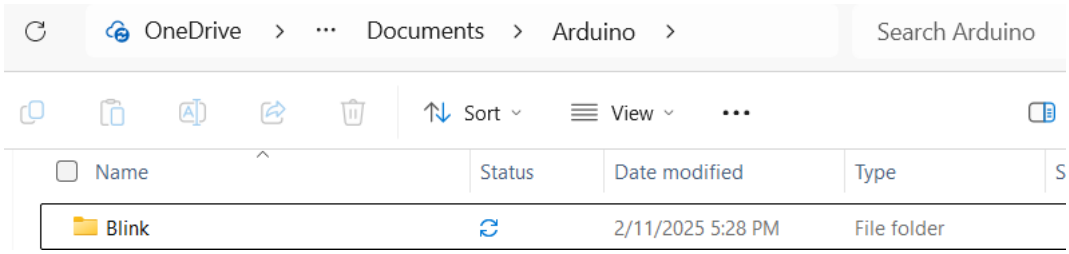


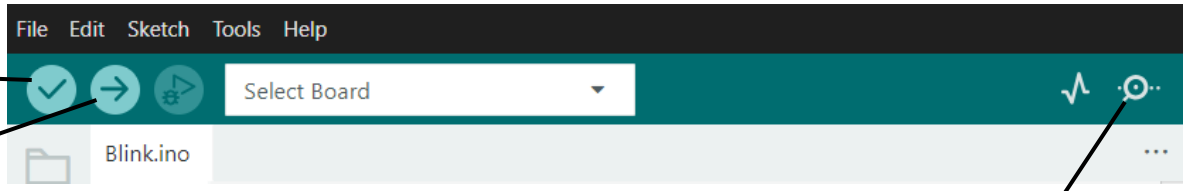
3 IDE functions

Function | Description

1. Sketches | Found in the Sketchbook folder (usually Documents > Arduino). These are .ino files written in the editor.
2. Verify | Checks your code for errors and shows them in the message area and console.
3. Uploading | Sends the compiled sketch to your connected Arduino board via USB.
4. Serial Monitor | Opens a window to communicate with the board and view data sent via serial.

The sketches are saved in folders in the Arduino Folder





Verify

Upload

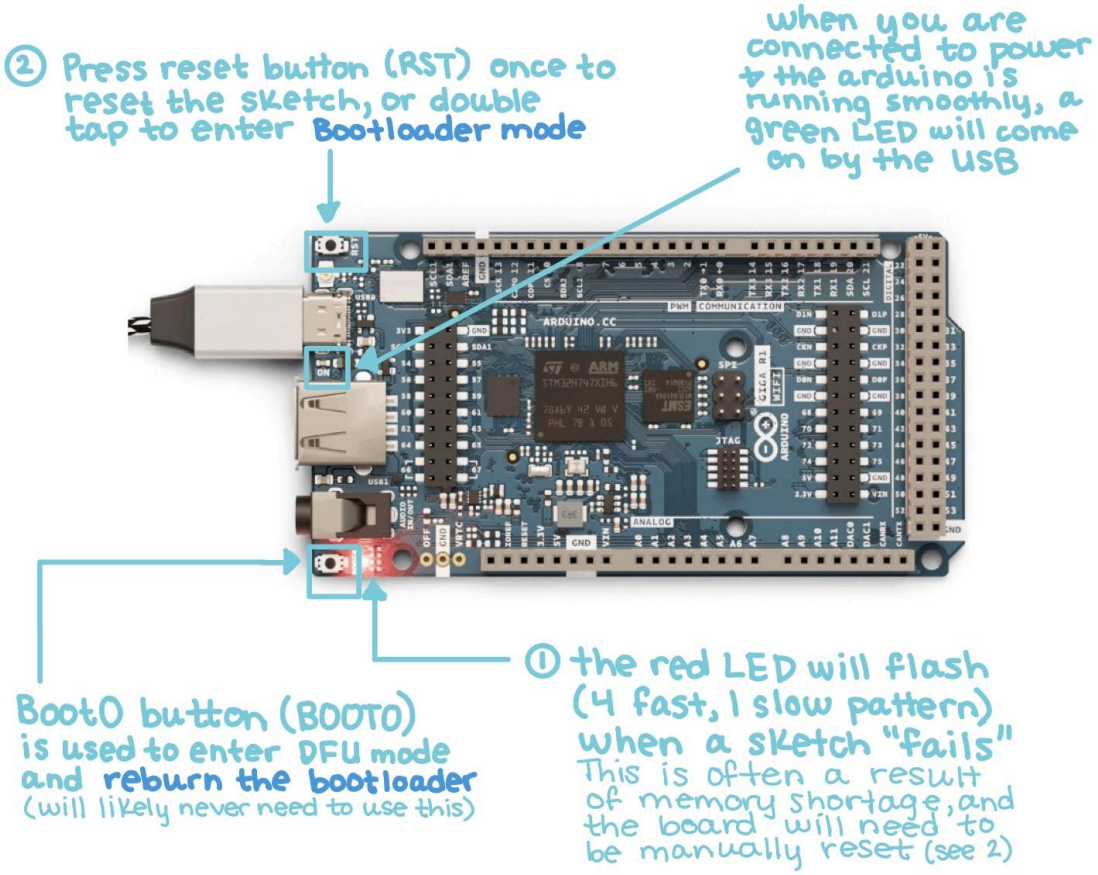
Serial Monitor

Introduction to Arduino

This section should provide you with everything you need to know about Arduino microcontrollers, specifically the Arduino Giga. If you've never had any experience working with Arduinos, they are programmable circuit boards that are used to transmit and receive information between a computer (sometimes referred to as a "serial monitor") and, in our case, Atlas Scientific sensors.

Essentially, it is like a secretary who reroutes phone calls. You call in by sending commands, and then then the Arduino takes down messages with important information. These messages are then forwarded to the correct people, with information that will make sense to them. In this case, the recipients are the sensors that monitor our system. The messages that are sent to the Arduino are then converted into simpler messages that the sensors can read, such as binary, or voltage signals, but we'll get deeper into that later. For this project, we used an Arduino Giga, the latest model of microcontrollers as of 2024, and throughout the guidebook, we may refer to it as just "the Giga."

To begin, we'll start with a brief overview of the hardware at hand: the external, physical components of the electronic systems. Below is a basic overview of the Giga and its various components:



The Giga also has a number of different pins, which are labeled on the sides of the board. These small slots are where the male end of a wire (the part where the actual wire is visible/sticking out) will go in order to connect various elements to the board. They all serve a unique purpose, as described below:

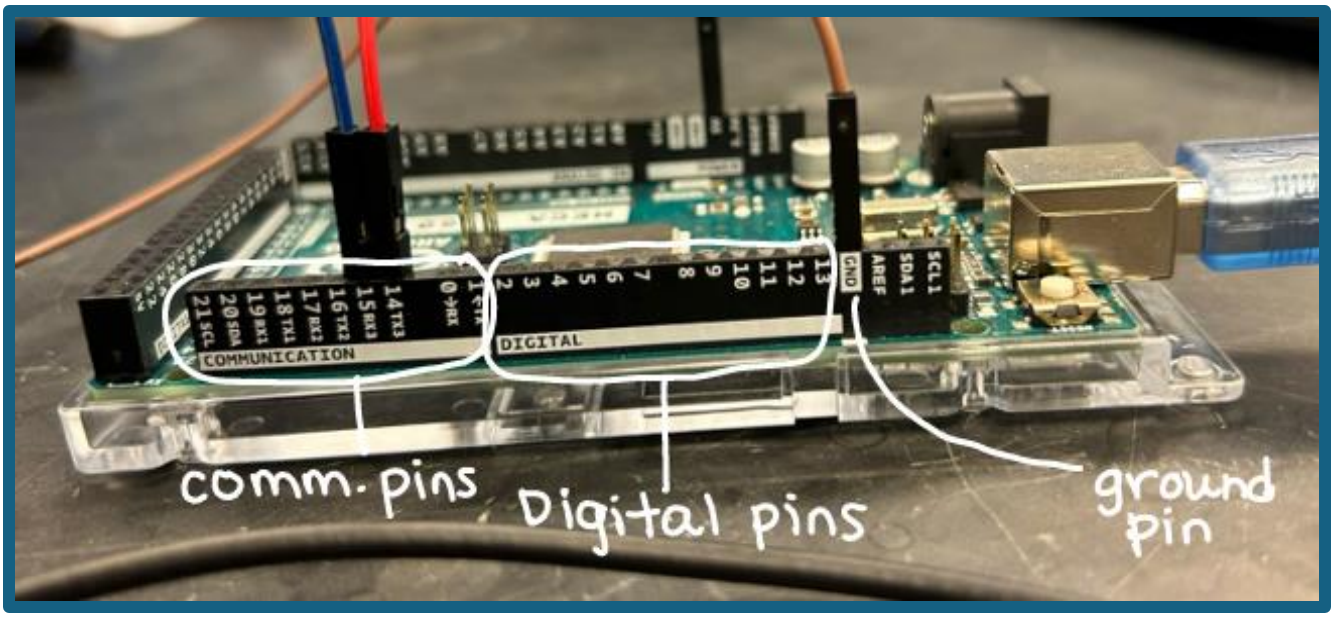
Note: Often, we use the term "mapping" to describe how the software (the serial number defined in our code) is related to the physical board. For example, if a sensor in our code is defined as Serial 3, it would be mapped to pins 19 and 18.

Communication Pins — used to send and receive data between the Giga, the sensors and the computer, among these are the "serial pins," which will be referred to later in an explanation of the software. Below are all the UART serial ports and the pins on the actual board that they connect to.

- Serial 0 corresponds with pins 0 (RX) and 1 (TX)
- Serial 1 corresponds with pins 19 (RX) and 18 (TX)
- Serial 2 corresponds with pins 17 (RX) and 16 (TX)
- Serial 3 corresponds with pins 15 (RX) and 14 (TX)

Digital Pins — read/send binary signals; can determine if a condition is "present" or not (basically only interprets something as "on" or "off")

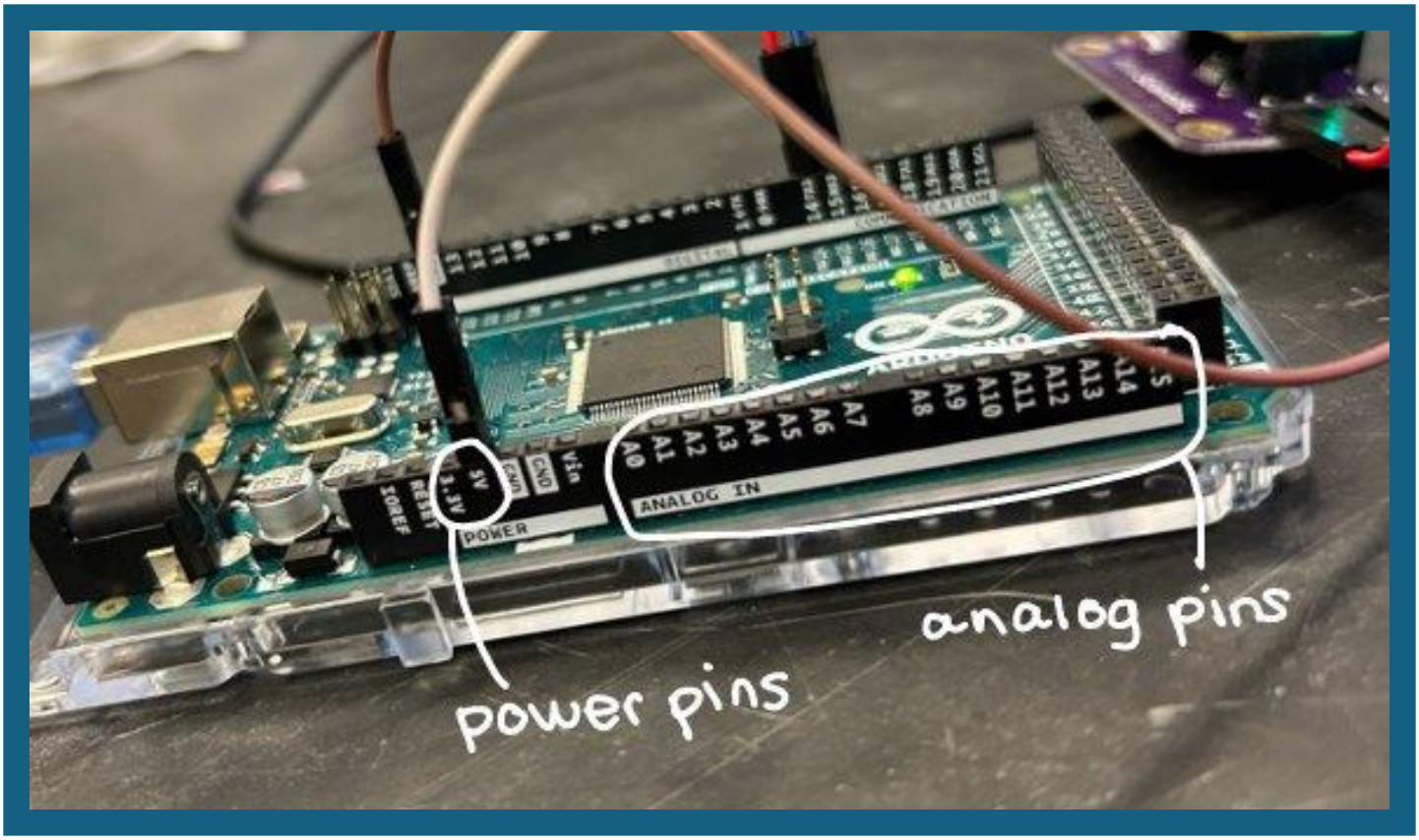
Ground Pin — Provides a route for current to flow through a circuit; all power to any component must be connected to a voltage source and ground.



Analog Pins — similar to communication pins, but they are used for receiving voltage readings from sensors; used for simple binary commands/messages

- Limited applications
- Some sensors can send their readings using changes in voltage, which the Arduino is able to convert to legible, digital messages

Power Pins — allow current to flow to other components and sensors

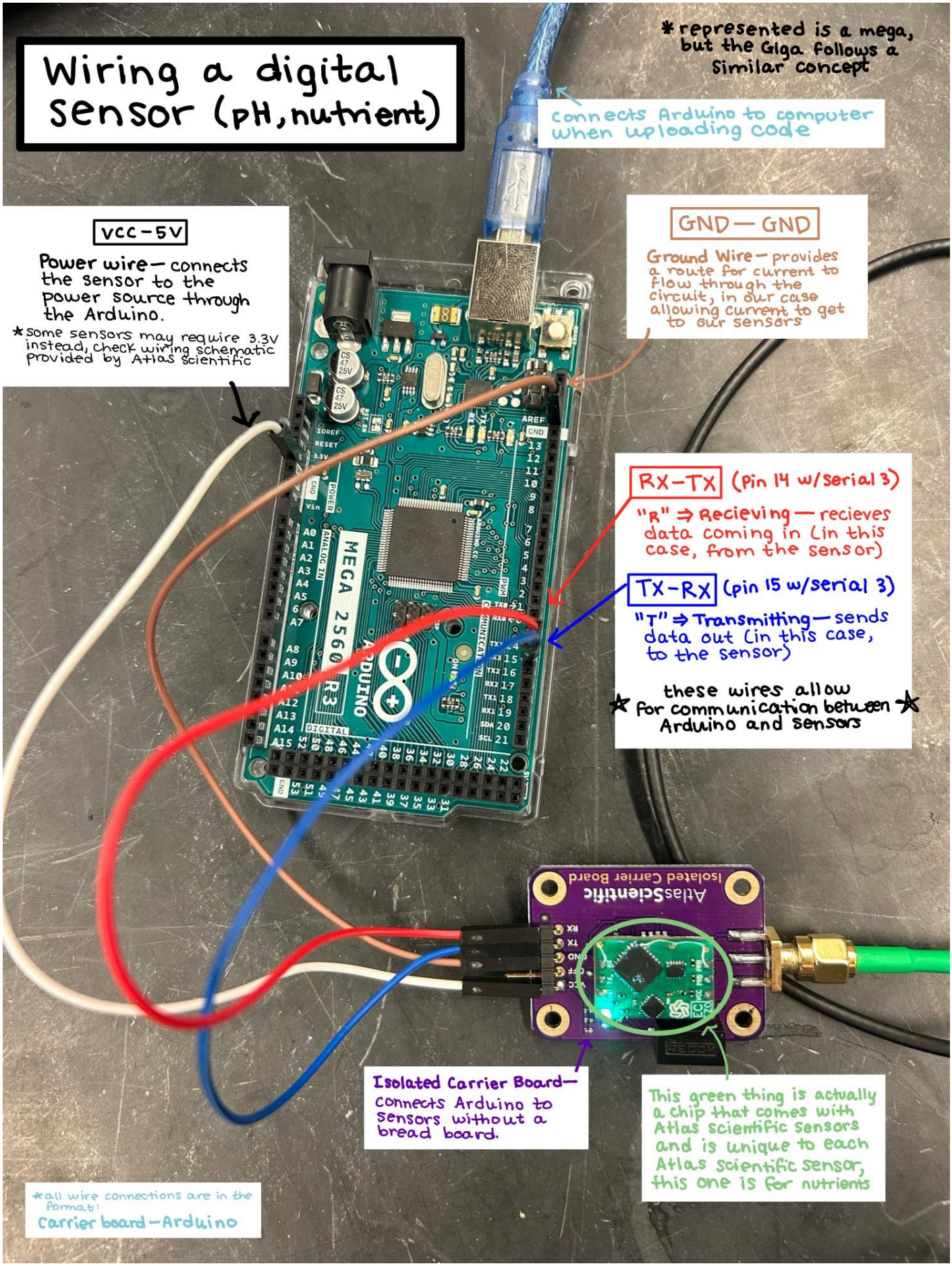


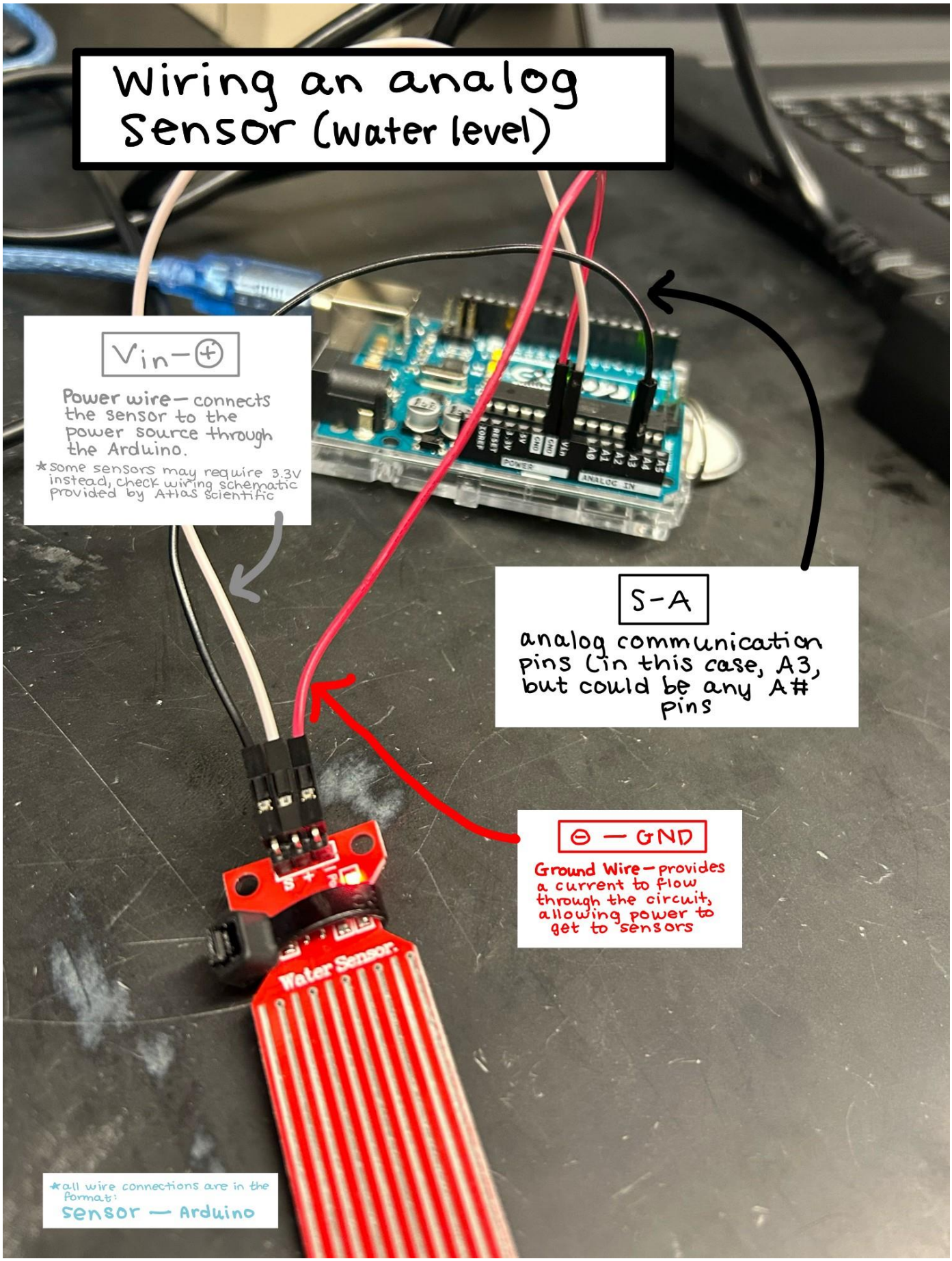
The other components that interact with the Giga are a bit more complicated to describe, so instead we'll begin with a visual representation. On the following pages, there will be diagrams detailing how to wire a digital and analog sensor.

Digital Sensors: digital sensors are able to communicate back and forth with the computer, sending data and receiving commands via the Arduino; the perform a measurement and report back a specific number.

Analog Sensors: analog sensors take a continuous range of data and report back a reflection of whether a condition is met or not (usually 0 or 1); they do not report back the specific measurements, but measurements can be extrapolated from voltage readings.

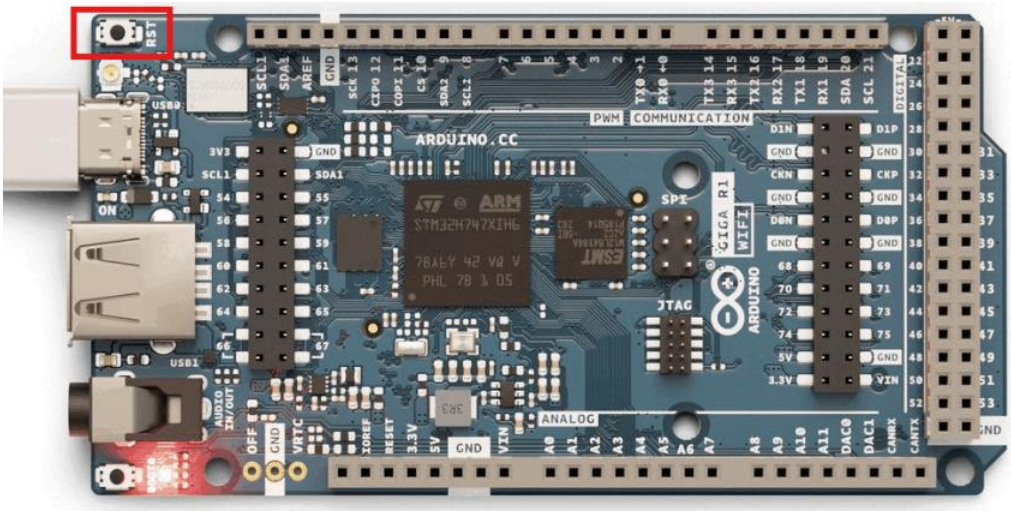
(all of the components involved with wiring are also labeled and described in the following diagrams)


Wiring a digital sensor (pH, nutrient)
*represented is a mega, but the Giga follows a similar concept
connects Arduino to computer when uploading code
VCC-5V
Power wire — connects the sensor to the power source through the Arduino.
*some sensors may require 3.3V instead, check wiring schematic provided by Atlas Scientific
GND — GND
Ground Wire — provides a route for current to flow through the circuit, in our case allowing current to get to our sensors
RX-TX (pin 14 w/Serial 3)
"R" ⇒ Recieving — recieves data coming in (in this case, from the sensor)
TX-RX (pin 15 w/Serial 3)
"T" ⇒ Transmitting — sends data out (in this case, to the sensor)
these wires allow for communication between Arduino and sensors
Isolated Carrier Board — connects Arduino to sensors without a bread board.
This green thing is actually a chip that comes with Atlas scientific sensors and is unique to each Atlas scientific sensor, this one is for nutrients
*all wire connections are in the format:
Carrier board — Arduino

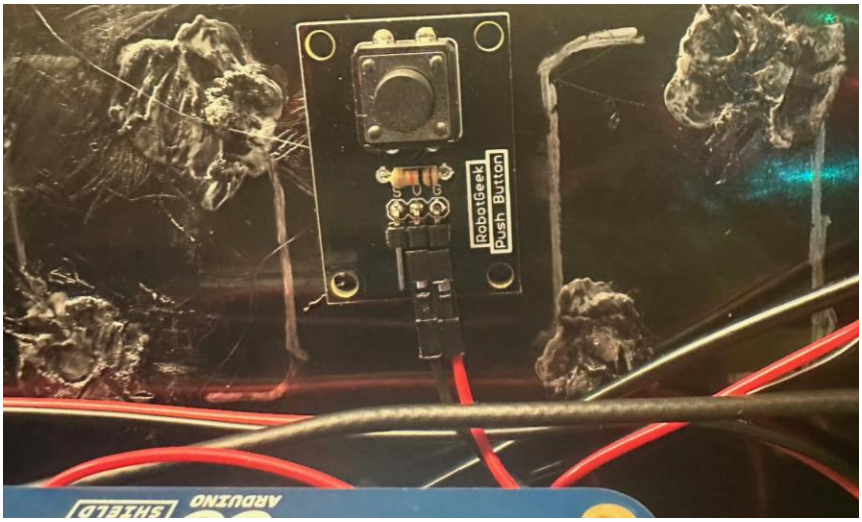

Wiring an analog Sensor (water level)
Vin — ⊕
Power wire — connects the sensor to the power source through the Arduino.
*some sensors may require 3.3V instead, check wiring schematic provided by Atlas Scientific
S-A
analog communication pins (in this case, A3, but could be any A# pins
⊖ — GND
Ground Wire — provides a current to flow through the circuit, allowing power to get to sensors
Water Sensor
*all wire connections are in the format:
sensor — Arduino

The Arduino Giga utilizes an operating system called Mbed OS. This operating system is used due to several of its features including real-time operating system capabilities, power management, and large range of libraries. However, occasionally the operating system will crash due to issues such as a memory shortage. If this occurs, reset the system using the reset button to fix it.

When the Mbed OS operating system crashes, the Giga will start flashing a red light. You can either press the reset button once, which will reset the sketch, or tap the button twice to enter bootloader mode, allowing you to reprogram the board.



Reset Button Relocation — To improve the accessibility of our Arduino Giga's reset switch, our team connected a new reset button. This button is mounted on the wall of the electronics box, opposite of the wire exit holes.



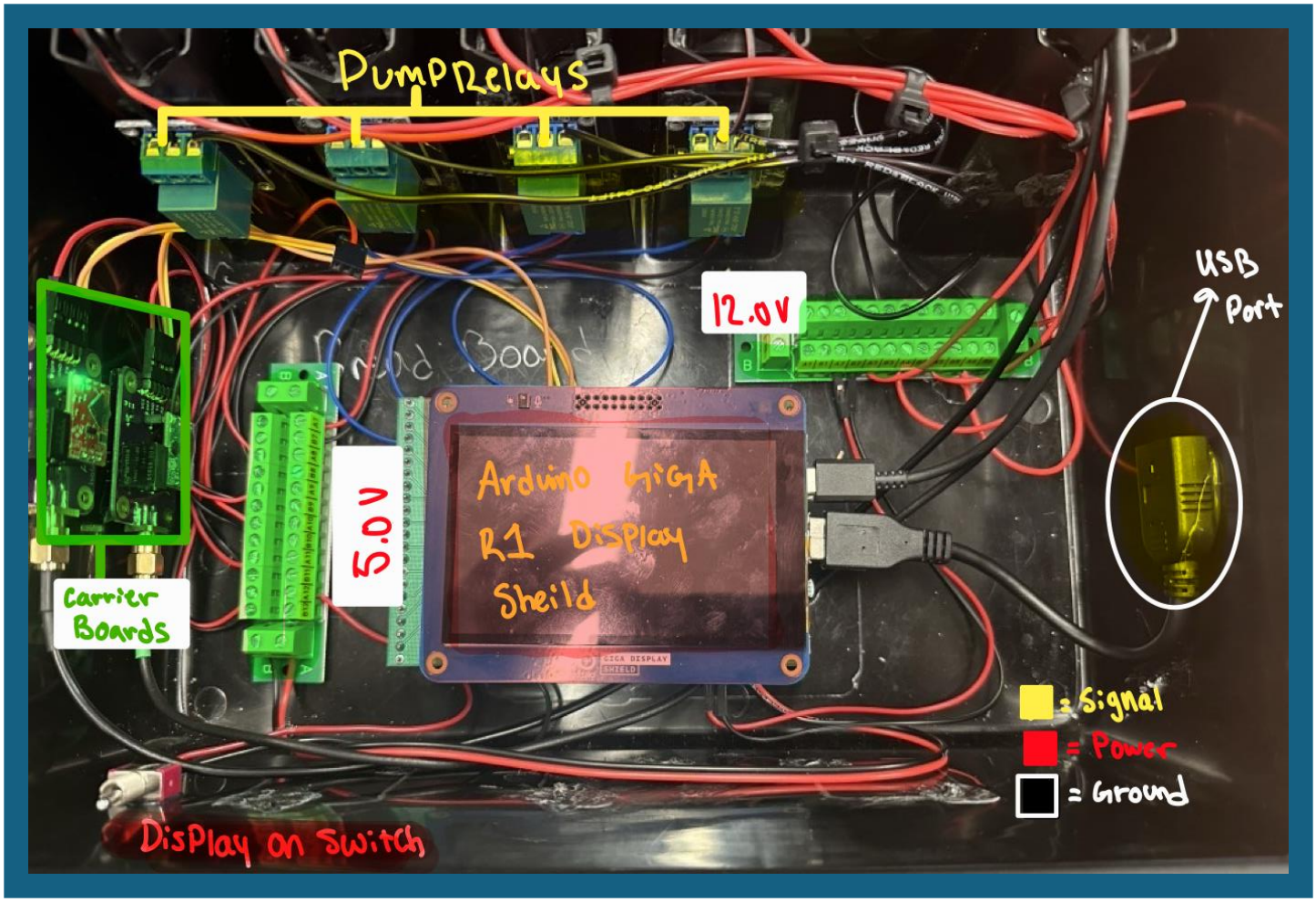
Housing

Introduction

This section is about the housing for the electronics system of the hydroponics project. The housing is the brain of the system holding the Arduino Giga R1, relays, carrier boards, and wiring.

Housing Function

The function of the housing it protect the most vulnerable components from flooding due to it positioning at the bottom of the system. Continually, the overall housing has the power buses to power the Arduino and relays for the pumps. The housing due to writing also is connected to the pump housing.



Coding in Arduino IDE

Introduction

This section should provide you with a basic understanding of coding in Arduino IDE. Arduino IDE is a software development platform designed specifically for interfacing with Arduino microcontrollers. The primary programming language in Arduino IDE is similar to the C and C++ programming languages, but with a few tweaks to cater to the use of Arduino boards and simplify the learning process for beginners. Within your Arduino IDE code file, you will most likely use these two functions: "void setup()" and "void loop()".

Setup Function

The first function, "void setup()", is used to prepare your Arduino microcontrollers for all of the processes within your program. This function includes actions such as setting the baud rate for each serial line (setting the communication speed), allocating memory space for temporary data storage (reserving storage for temporary memory), defining pins as input or output, and any other necessary setup procedures.

The second integral function within any Arduino IDE program is "void loop()". This function will contain most, if not all, of the processes in your program. This function is a time loop, repeating actions and communications indefinitely until the program is stopped or updated.

Below is an example of a program given by Arduino in which the user is controlling a led with an Arduino board. The program utilizes the "void setup()" function to define the LED_BUILTIN pin as an output. Then, it uses the "void loop()" function to perform the actions necessary for a repeatedly blinking LED.

```
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
```

void processInput()

```
void processInput() {
  int first_comma = inputstring.indexOf(',');

  if (inputstring.startsWith("PH")) {
    PH_Serial.print(inputstring.substring(first_comma + 1));
    PH_Serial.print('\r');
  }
```

- Routes commands for pH and EC sensors.
- Handles status requests ("OUTPUT") by printing current values.
- Uses commas to separate commands from parameters.

User Input	What Happens
"PH, ..."	Sends to the pH sensor
"EC, ..."	Sends to the EC sensor
"OUTPUT"	Prints pH, EC, TDS, and Salinity values

```
// If a command was unrecognized a list of possible commands will be shown
else {
  Serial.println("\nHere are the available commands:");
  Serial.println("To set the ideal PH type:");
  Serial.println("SETPH,(ideal PH value)");
  Serial.println("To set the ideal EC type:");
  Serial.println("SETEC,(ideal EC value)");
  Serial.println("To set the clock type:");
  Serial.println("SETTIME,[day],[hour]");
  Serial.println("To get a system update of all current values type:");
  Serial.println("OUTPUT");
  Serial.println("To send commands to the PH sensor (found in the guidebook or atlas scientific pdf) type:");
  Serial.println("PH,(comamnd)");
  Serial.println("To send commands to the EC sensor (found in the guidebook or atlas scientific pdf) type:");
  Serial.println("EC,(comamnd)\n");
}
```

void updatepH() & voidupdateEC()

```
// Updates the value of the PH variable, or displayed the error message
void updatePH() {
  delay(10);
  //Serial.println(PH_sensorstring); // Uncomment this line to see the message from the sensor

  if (isdigit(PH_sensorstring[0])) {  // If the first character in the string is a digit
    PH = PH_sensorstring.toFloat();   // Update the PH float variable
  } else {
    Serial.println(PH_sensorstring);  // If its a message from the sensor (not a number) then print it to the user
  }
  PH_sensorstring = "";  // Clear the string for the next input
}
```

void updatePH()

- Extracts pH readings from the sensor's response.
- Filters out error messages and ensures only valid numerical data.

Sensor Response	What Happens
"6.8"	PH = 6.8 (Stores the value)
"Error: Sensor Not Found"	Prints "Error: Sensor Not Found" to the serial monitor

```
void updateEC() {
  delay(10);
  //Serial.println(EC_sensorstring);

  int first_comma = EC_sensorstring.indexOf(',');
  int second_comma = EC_sensorstring.lastIndexOf(',');
  if (isdigit(EC_sensorstring[0])) {  // If the first character in the string is a digit
    EC = EC_sensorstring.substring(0, first_comma + 1).toFloat();
    TDS = EC_sensorstring.substring(first_comma + 1, second_comma + 1).toFloat();
    Salinity = EC_sensorstring.substring(second_comma + 1).toFloat();
  } else {
    Serial.println(EC_sensorstring);  // If its a message from the sensor (not a number) then print it to the user
  }
  EC_sensorstring = "";  // Clear the string for the next input
}
```

void updateEC()

- Extracts EC (Electrical Conductivity), TDS (Total Dissolved Solids), and Salinity readings from the sensor's response.
- Filters out non-numerical messages and ensures only valid numerical data is stored.

Sensor Response	What Happens
"764.5,481.2,376.8"	EC = 764.5, TDS = 481.2, Salinity = 376.8 (Values are parsed and stored as floats)

void checkFlooding()

```
//check if they are flooding
bool tempFloodFlag = false;
for (int i = 1; i < 5; i++) {   // Check if each sensor is flooding
  if (sensorValues[i] > thresholdValues[i]) {
    tempFloodFlag = true;
  }
}
if (tempFloodFlag) {
  Flooding = true;
} else {
  Flooding = false;
}
```

The "void checkFlooding()" function's main purpose is to monitor the water level within the system. In the code snippet above, the program is checking that the water level sensor values are below the threshold values.

```
// Turn off pump if flooding
// if (Flooding || !digitalRead(reservoirSensor)) {
if (Flooding) {
  Serial.print("\nSystem Flooding\n");
  digitalWrite(pumpPin, 0);
} else {
  digitalWrite(pumpPin, 1);
}
```

Further within the function, the program analyzes a boolean variable that indicates whether flooding has occurred. If there has been flooding, then the water pump will be shut off until the water level has returned to normal.

void regulateLevels()

```
// Triggers relays for the pumps to turn on/off do distribute chemicals
void regulateLevels() {
  //regulating ph section
  if (PH < (idealPH - .5 * idealPHThreshold)) {
    Serial.print("\nIncreasing PH");
    digitalWrite(PHUpPin, HIGH);
    delay(pumpTime);
    digitalWrite(PHUpPin, LOW);
  } else if (PH > (idealPH + .5 * idealPHThreshold)) {
    Serial.print("\nDecreasing PH");
    digitalWrite(PHDownPin, HIGH);
    delay(pumpTime);
    digitalWrite(PHDownPin, LOW);
  }
```

The function "void regulateLevels()" controls the addition of various chemicals within the system using relays. In the snippet above, the program regulates the pH by adding different chemicals to increase or decrease the pH.

```
//regulating EC section
if (EC < idealEC) {
  Serial.print("\nAdding nutrients");
  digitalWrite(NutrientsPin, HIGH);
  delay(pumpTime);
  digitalWrite(NutrientsPin, LOW);
}
```

Later within the function, the program regulates the EC to ensure the proper electrical conductivity within the system.

void updateDisplay()

```
void updateDisplay() {  //screen resolution 480x800 + default text size is 7 pixels tall
  display.fillScreen(TAN);
  display.setTextSize(5);
  //ph
  display.setCursor(25, 25);
  display.setTextColor(RED);
  display.print("PH: ");
  display.print(PH);
  display.print(" | ");
  display.print(idealPH);
  //ec
  display.setCursor(25, 111);
  display.setTextColor(BLUE);
  display.print("EC: ");
  display.print(EC);
  display.print(" | ");
  display.print(idealEC);
  //tds
  display.setCursor(25, 197);
  display.setTextColor(BLUE);
  display.print("TDS: ");
  display.print(TDS);
  //salinity
  display.setCursor(25, 283);
  display.print("Salinity: ");
  display.print(Salinity);
  //time
  display.setTextSize(3);
  display.setTextColor(BLACK);
  display.setCursor(25, 434);
  display.print("Elapsed data logs: ");
  display.print(logCount);
```

The void updateDisplay() function updates the visual display of the most current sensor readings for pH, EC, TDS, and Salinity. It also displays the log count allowing users to see how long the system has been running.

```
//overflow
for (int i = 1; i < 5; i++) {
  if (sensorValues[i] <= thresholdValues[i]) {
    display.fillCircle(735, 65 + (i - 1) * 115, 40, GREEN);  //was 117
  } else {
    display.fillCircle(735, 65 + (i - 1) * 115, 40, RED);
  }
```

This allows users to visualize whether each of the 4 water level sensors is within safe range (GREEN) or indicating a flood condition (RED).

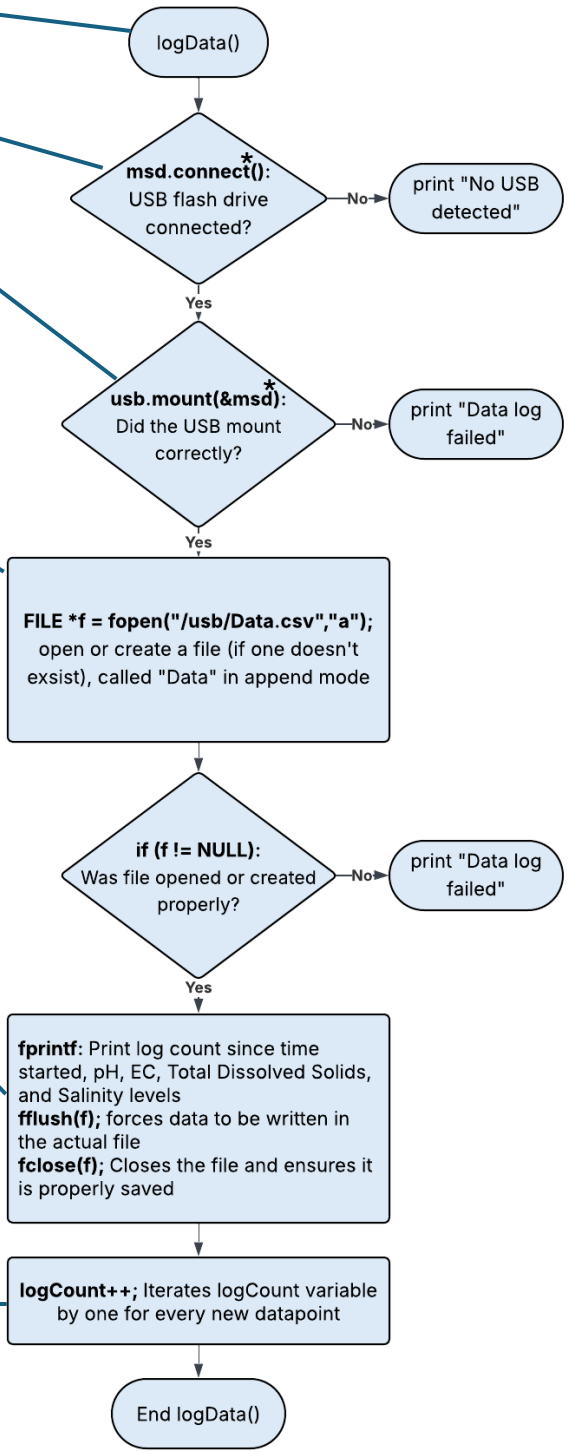
void logData()

Collects data from sensors over running time

```
void logData() {
  if (msd.connect()) {
    if (usb.mount(&msd)) {

      FILE *f = fopen("/usb/Data.csv", "a");
      if (f != NULL) {
          fprintf(f, "%d", days);
          fprintf(f, "%s", ",");
          fprintf(f, "%d", hours);
          fprintf(f, "%s", ",");
          fprintf(f, "%f", PH);
          fprintf(f, "%s", ",");
          fprintf(f, "%f", EC);
          fprintf(f, "%s", ",");
          fprintf(f, "%f", TDS);
          fprintf(f, "%s", ",");
          fprintf(f, "%f", Salinity);
          fprintf(f, "%s", "\n");

          fflush(f);
          fclose(f);
          logCount++;
      } else {
        Serial.print("\nData log failed");
      }
  } else {
    Serial.print("\nNo USB detected!");
  }
}
```



***msd.connect():** function part of msd (mass storage device) library that establishes connection with USB flash drive

***usb.mount(&msd):** "mounts" USB flash drive so that you can read/write files on it

Display Code

How the Display Turns On

Function | Description

- ❑ Include Display Library | Adds the library Arduino_GigaDisplay_GFX.h, which provides functions for controlling the display.
- ❑ Create Display Object | GigaDisplay_GFX display; initializes the screen so you can start interacting with it.

```
2   #include "Arduino_GigaDisplay_GFX.h"
3
4   GigaDisplay_GFX display;  // create the object
```

- ❑ Define Pin & Frequency |
  - ❑ displaySwitchPin = 50; sets which pin controls display power.
  - ❑ displayFrequency = 2 * 1000; determines how often the display updates (in milliseconds).

```
16  int displaySwitchPin = 50;
17  int displayFrequency = (2 * 1000);  //how often the display is updated [ms]
```

- ❑ Start Display | display.begin(); runs once to initialize the display.

```
98      display.begin();
```

- ❑ Update Condition | The if statement checks:

```
146 if ((millis() - displayTempTime) > displayFrequency) {
147   if (digitalRead(displaySwitchPin)) {  //if the switch is on power on and update the screen
148     digitalWrite(D74, HIGH);
149     updateDisplay();
150   } else {
151     digitalWrite(D74, LOW);             //if the switch is off power down the screen
152   }
153   displayTempTime = millis();
154 }
```

- ❑ If enough time has passed (based on frequency)
- ❑ If the switch pin is ON → turns on display and calls updateDisplay()
- ❑ Else turns OFF the display

What is Being Displayed

Function | Description

- ❑ void updateDisplay | function declaration
- ❑ display.fillScreen | sets background color from the defined colors above

```
355   void updateDisplay() {
356     //screen resolution 480x800
357     //default text size is 7 pixels tall
358     display.fillScreen(TAN);
359
360     display.setTextSize(5);
```

- ❑ In order to display text you have to set the cursor to then print the text at that position

```
370     //ec
371     display.setCursor(25, 111);
372     display.setTextColor(BLUE);
373     display.print("EC: ");
374     display.print(EC);
375     display.print(" | ");
376     display.print(idealEC);
```

- ❑ To display the overflow circles we check for how much it has overflowed to determine how many circles to display
- ❑ We check for if the sensor value is less than the threshold value and if it is we change the color of the circle

```
401     //overflow
402     for (int i = 1; i < 5; i++) {
403       if (sensorValues[i] <= thresholdValues[i]) {
404         display.fillCircle(735, 65 + (i - 1) * 115, 40, GREEN);
405       } else {
406         display.fillCircle(735, 65 + (i - 1) * 115, 40, RED);
407       }
408     }
409   }
```

Setting Customization

Open Arduino IDE on a computer that has all the proper software installed on it (detailed in the "How to Download Arduino IDE" section). The following commands will allow you to make changes to the current state of the system. If you forget a command or would like to view a list of all your available options, you can also just type "HELP". The values enclosed by [ ] represent the number that you should input after the command, this value does not need to be in brackets when you type the command. You can also adjust the amount of time the lights are on using the dial plugged in.

Command	Function	Uses
SETPH, [pH value]	Allows you to set a target pH value (the system will then adjust to maintain this selected value)	Good for running different experiments to determine how pH affects plant growth Growing plants with different pH requirements
SETEC, [EC value]	Allows you to set a target EC value (the system will then adjust to maintain this selected value)	Good for running different experiments to determine how varied nutrient levels affect plant growth
SETTIME, [day],[hour]	Allows you to the set the time and date to log data	Allows you to input the real date at which you begin growing plants, so that you can track overall changes to the system over the entire grow cycles (data collection)

Command	Function	Uses
PUMP, MAIN, [1 or 0]	Turns pump on (1) or off (0)	Turning on the pump if it will not turn on automatically Manual pump shut off
PUMP, PH, DOWN	Temporarily pumps pH down into the system	Quickly decreasing the pH of the system to see how electronics react Testing to see if probes are changing values/dosing is causing change
PUMP, PH, UP	Temporarily pumps pH up into the system	Quickly increasing the pH of the system to see how electronics react Testing to see if probes are changing values/dosing is causing change
PUMP, NUTRIENTS	Temporarily pumps FloraMicro and FloraGrow into the system	Quickly inserting nutrients into the system to see how electronics react Testing to see if probes are changing values/dosing is causing change

Command	Function	Uses
OUTPUT	Will output current system readings	Checking to ensure display is accurately updating

Changing the Interval of Data Logging:

- ❑ Open the Arduino code file on a device that has all the proper libraries installed on it
  - Search for "logFrequency" using CTRL+F
  - Replace the current time interval located within logFrequency with desired time interval
  - **NOTE:** Arduino uses microseconds to keep time, therefore the current line of code causes a new data point to be logged every 12 minutes (will update the time, and record the values of all sensors in the system at this time)
- ❑ If you would like to look at certain parts of a grow cycle very closely (to determine volume of nutrients needed to sustain system, etc.) you can decrease the time interval.
- ❑ If you are mostly focused on tracking long term trends or data, you can use a much greater time interval, such as daily or semi-daily.

```
27    int logCount = 0;
28    int days = 0;
29    int hours = 0;
30    int logFrequency = (12 * 60 * 1000);
```

Troubleshooting

This section is dedicated to major issues you hopefully will not encounter, but should be prepared for, nonetheless. This section will also cover short-term fixes for problems that require our service, allowing the system to keep running until we are able to help.

Dysfunctional or broken probe (or any electronic component besides the pump)

- ❑ pH Probe
  - Attempt to recalibrate the pH probe using the instructions in the system setup portion of the guidebook
  - If readings remain unreliable (inconsistent, causing the pH to dip 1-2 below or above the stated target value for multiple hours), disconnect the pH probe from the housing system and recap in standard solution
  - In the meantime, monitor the pH of the system at the end of each day, and add a small amount of pH up or down as needed until a desired pH is reached
  - In general, the pH of the system should not fluctuate that much unmonitored, and a little pH up or down usually goes a long way

- ❑ Nutrient Sensor
  - Try recalibrating the sensor using the instructions found in the setup section
  - If readings remain inaccurate or unstable, disconnect the nutrient sensor and add the recommended dosage of nutrients (listed on the back of the FloraMicro and FloraGrow bottles) to the system every 1-2 weeks, depending on how the plants respond
  - If they appear yellowish or lighter than normal, but still grow rapidly, decrease dosage to once every 2 weeks
  - If growth is stunted and the plants begin to shrivel or die, increase the dosage to once every week

- ❑ Other electronic components
  - For errors with the Arduino Giga or other Arduino products contact Arduino customer support
  - It may take a while for them to get back to you, so feel free to contact us as well

Consistent flooding

- ❑ If possible, check to see if there are any large pieces of debris blocking the ends of each grow bed and remove them
    - If you are unable to get a clear look, drain the system using the instructions in the end of year maintenance section
    - Clean around the ends of each grow bed, and allow the system to fill, checking to see if the issue is specific to a certain layer (one remaining empty, one always overflowing, etc.)
    - If you can narrow down the issue to a specific layer, try applying pressure to the tubes going in and out of the layer to see if there is any debris stuck on the inside

Glossary

Term	Definition
Electrical Conductivity (EC)	A quantitative measurement of a material's (in our case, a liquid's) ability to carry electrical current.
Peristaltic Pump	A positive displacement pump which moves fluids through a flexible tube by compressing and releasing the tube using rollers or shoes.
pH	A quantitative measurement of the acidity or basicity of a solution, generally ranging from 0 to 14. Pure water has a neutral pH of 7. A solution with a pH less than 7 is considered acidic; a solution with a pH greater than 7 is considered basic, or alkaline.
Salinity	The amount of dissolved salts present in water.
Total Dissolved Solids (TDS)	The total concentration of dissolved substances in water, including inorganic salts and organic matter.

Links

Links to Electronic Components

pH Probe - Atlas Scientific Lab Grade pH Probe #ENV-40-pH-8cm

pH Circuit (chip) - Atlas Scientific EZO™ pH Circuit #EZO-pH

Nutrient Sensor - Atlas Scientific Conductivity Probe K 10 #ENV-40-EC-K10

Nutrient Sensor Circuit (chip) - Atlas Scientific EZO™ Conductivity Circuit #EZO-EC

Isolated Carrier Board - Atlas Scientific Gen 2 Electrically Isolated USB EZO™ Carrier Board #G2-USB-ISO

Links to Helpful Videos/Websites

pH Kit | Atlas Scientific - Atlas Scientific pH Kit #KIT-101P

How does a peristaltic pump work?

A Guide To Reusing Rockwool in Hydroponics – FloraFlex