# Final Project Design Document

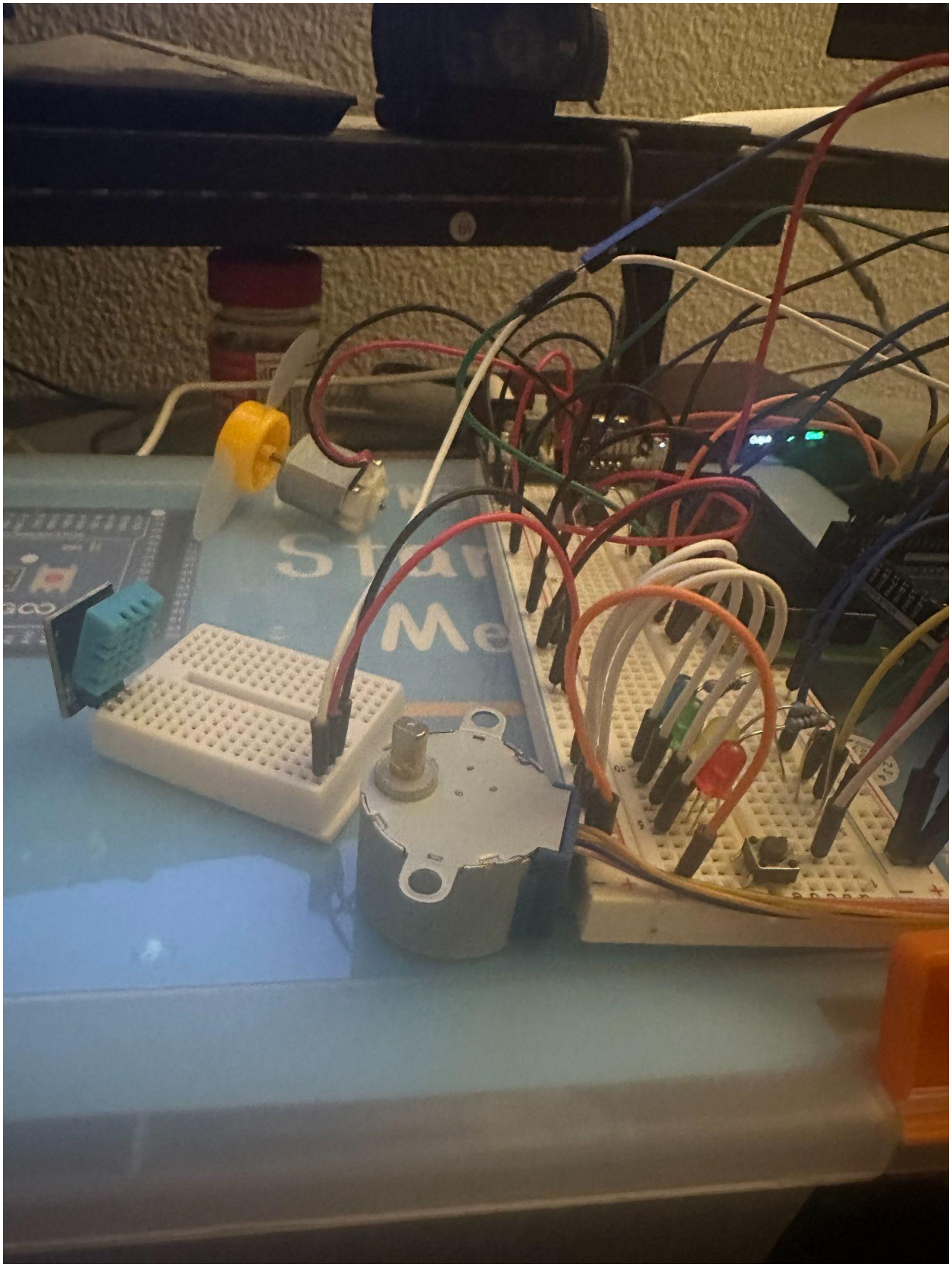
This project serves as the synthesis of everything that has been taught and presented to the CPE 301 Fall Semester class where the task of creating a circuit for a swamp cooler was given. The main controller that handles all of the computational tasks is an Arduino 2560 Mega that came with the Elegoo Starter Kit, alongside many other peripherals. The peripherals used in this project were a DH11 temperature and humidity sensor, a DC motor for the fan, a Stepper motor for the vent control, and a water level sensor. The libraries used were LiquidCrystal, RTClib, and Stepper. The system can be in four states, each represented by a different LED bulb and they are as follows - Idle (green), Error (red), Yellow(Fan off), Blue (Fan on).The crux of the circuit relies on the relationship between the fan and the water. If the temperature got too hot, the fan would begin to circulate and make use of the thermodynamic relationship between the water and the air to begin to lower the temperature of the inside of the swamp cooler (had there been a physical casing for such a cooler). The first issue that was found while completing this project was that the temperature sensor appeared to be broken. Upon removing it from the kit, the sensor was leaning off of the metal plate and was barely connected so it made managing the states of the system extremely difficult. Because of this, the demonstration video shows what can be demonstrated without the relationship between the humidity and the fan - the water level sensor. The beauty of the water level sensor is that it relies on no library function and has its analog signal converted via bitwise operations instead of the library function that would normally be used. Additionally, every single register change was done via bitwise operations as well. With the brief introduction out of the way, the rest of the project will be discussed in depth.

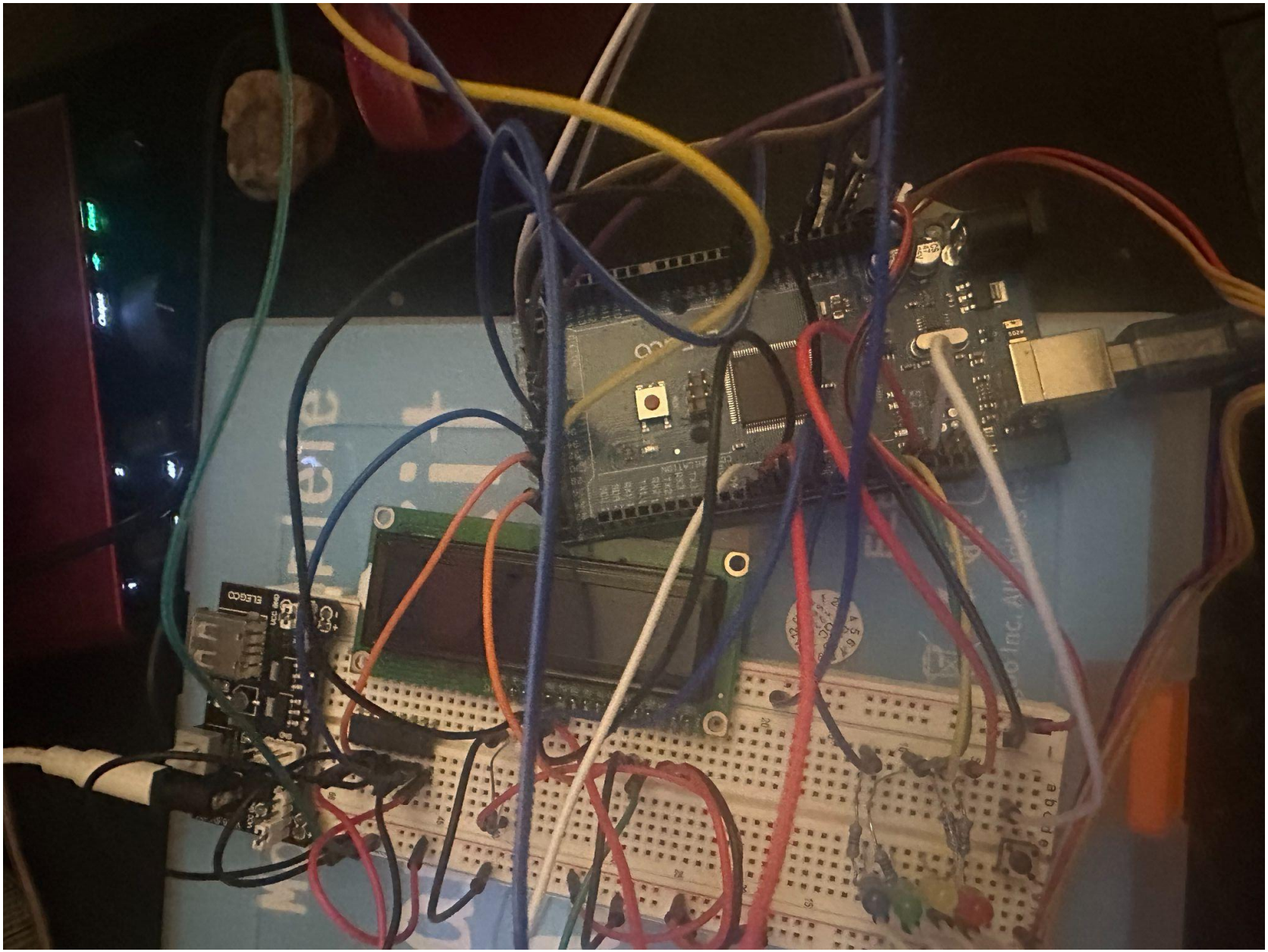
The water threshold, since it was one of the few things that actually worked, was the first thing to be implemented. A lot of time was spent just paying attention to how the value of the sensor changed with the addition and removal of water. It seemed to have a value of around 287 when it was a little wet, and averaged closer to 250 when it was dry. There was also a weird thing where depending on how the ground wire from the arduino was wired to the circuit board would affect the value of the water level that was returned. Again, the water level value was found by using the adc_read() function written in a previous lab. The water level was responsible for two states - idle and error. If the water level was above the threshold then the system was in the "Idle" state barring any other issues occurring as well. If the water level was below the threshold then it would be in the "Error" state. These states were represented by the Green and Red LED lights respectively.

The DHT11 (Three-pronged sensor) caused the most issues with this implementation, only returning "NaN" values no matter how it was wired. Upon further inspection it appears that the chassis has sustained some sort of damage which may have caused the issue, or the issue lies within miswiring. Regardless, the thought behind the implementation of the code was that the output from the DHT11 was sent to an analog pin (in this case, the second pin) to be converted into a digital signal. This signal would then be read, converted into an integer, then placed into the respective variables (temperature and humidity). If the temperature got too high, the fans would be triggered and the vents could then be adjusted. If the temperature got too low,

the fans would be turned off and the vents would not be adjustable. The code implementation relied on the DHT.h library

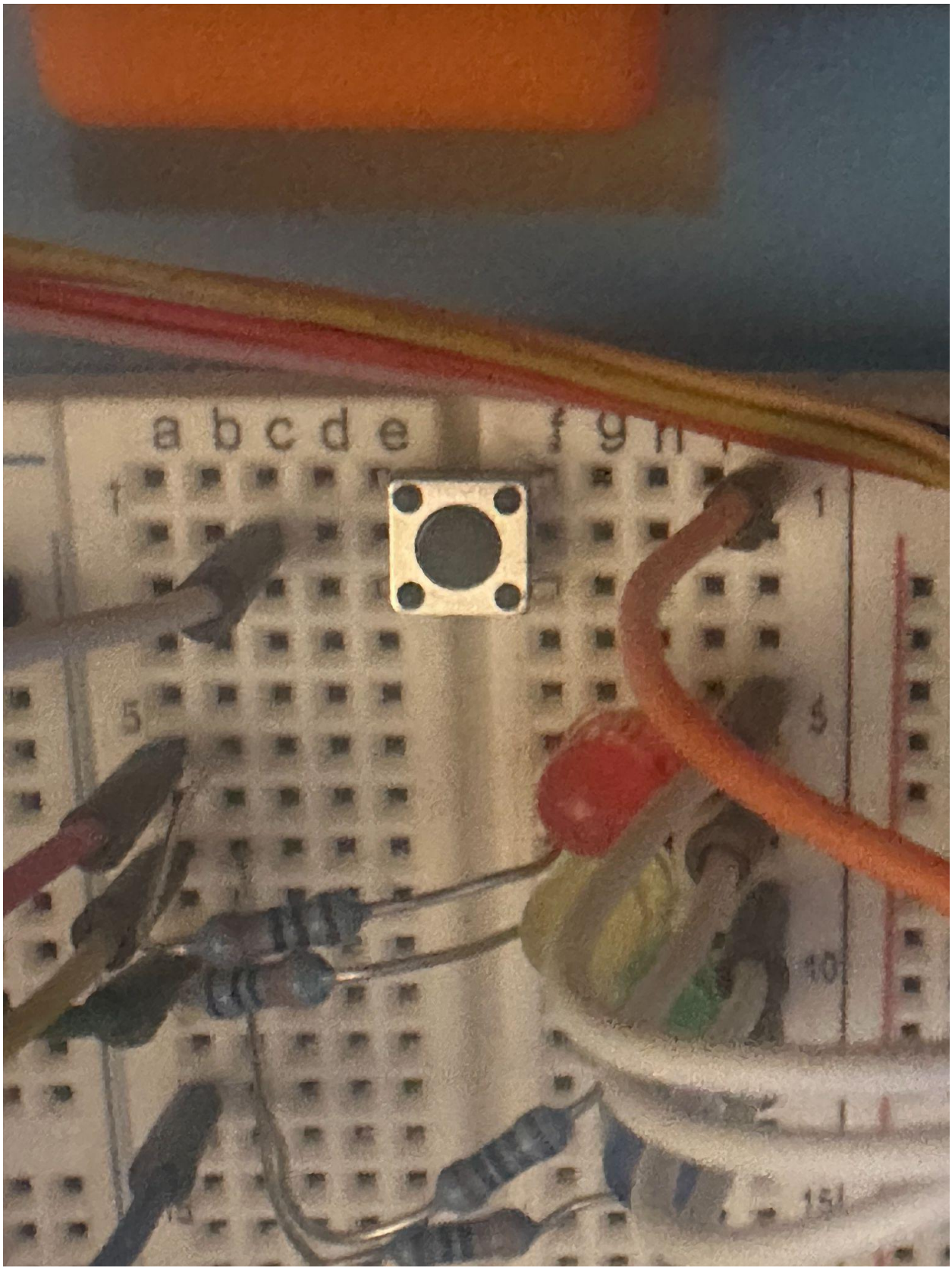
The motors (DC for fan and Stepper for vents) required additional apparatuses to actually work. The DC motor required power directly from the peripheral that put power directly into the board to avoid damaging the arduino's circuitry. On top of that, the DC motor needed an L293D chip to actually operate. Only the top inputs were used, Input 3 and 4 for clockwise and counterclockwise motions, in addition to the power and ground inputs with two additional outputs running from the arduino to the chip to pass along instructions. The stepper motor had a similar apparatus however there were only four inputs responsible for the stepper motor on the driver.






Caution
HOT!
Precaución
¡CALIENTE!
Cream
Sugar

a b c d e
35
30
25
20
15
10
5
1


a b c d e
f g h
1
5
10
15

Datasheets:

- https://docs.arduino.cc/resources/datasheets/A000067-datasheet.pdf
- https://www.electronicoscaldas.com/datasheet/ULN2003A-PCB.pdf
- https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf
- https://asset.conrad.com/media10/add/160267/c1/-/en/001485323DS01/datasheet-1485323-iduino-moisture-sensor-module-1-pcs-se045.pdf

Git Repository: https://github.com/stjohn-julian/CPE301_Final_Project