Members: Oliver Zolan, Sutter Reynolds, Bennet Rau, Conner Marks

CPE 301 Final Project : Evaporating Cooler

GITHUB LINK

https://github.com/Zolan-Oliver-1104/CPE301Final

Overview

The goal of this final project is to create a functioning Evaporating Cooler, using the tools and experience from labs reports prior in the year. To define an Evaporating Cooler, it is a function machine that controls temperature by intaking the surrounding air and releasing it back after picking up water evaporation to humidify and cool it. This is done by the functional components being a monitor for water levels, monitor for displaying humidity and temperature, fan motor, adjusting vent output, on/off button, recordings of use. These components will represent the real components of an Evaporating cooler which are the blower motor, evaporative pads, blower, float, duct, recirculating pump, pump screen, and water distribution lines that are normally used in an Evaporating Cooler.

Design

As for the Design aspect, this experiment was done completely through Arduino. This being said, the Arduino Mega 2560 was the main component in controlling the swamp cooler. This was connected to a LCD display, displaying the temperature and humidity on a 2x16 LCD screen. This takes in the inputs from a DHT11 temperature and humidity sensor that returns the temperature and humidity to the LCD, and is recorded through a real-time clock module. The LCD also takes in a water sensor that determines errors based on water levels being too low, and

keeps it active if the water level is adequate. Our project contains two separate buttons, a on/off button and a rotating button for the stepper motor. These buttons are connected to the main Arduino Mega 2560 along with the all functioning components. There is a fan motor connected to the Arduino Mega 2560 that will receive a signal depending on the Arduino's current state to run or not. The states contained within the Arduino, are displayed as blue, green, yellow, and red LEDs each determining what state is currently being run.

The LCD is used to take information from the DHT11 temperature and humidity sensor and display them back to the user in real time. This display is matched with arithmetic to display a celsius temperature back through the display on the top of the 2x16 display, and humidity below it. There is another state to the LCD, which displays depending on an error state being activated. The error state reads "Error" to the display of the LCD, which is in turn due to the water level sensor reading a water level that is below the water threshold. The LCD is manually connected to power and ground, and declared in the Arduino as LiquidCrystal lcd(31, 32, 33, 34, 35) using pins 31, 32, 33, 34, and 35 that receive data from the water level sensor and DHT11.

Breaking down the DHT11 temperature and humidity sensor, its purpose is to specifically record and send data of the temperature and humidity back to the Arduino. The DHT11 is defined as DHT_PIN 7 for the pin and DHT_TYPE DHT11 for the type defined. The temperature is recorded through readTemperature(), and humidity through readHumidity(), through the DTH library. These recordings are sent back through the arduino, to be recorded on Arduino to the real time clock module and sent back to the LCD monitor to be displayed.

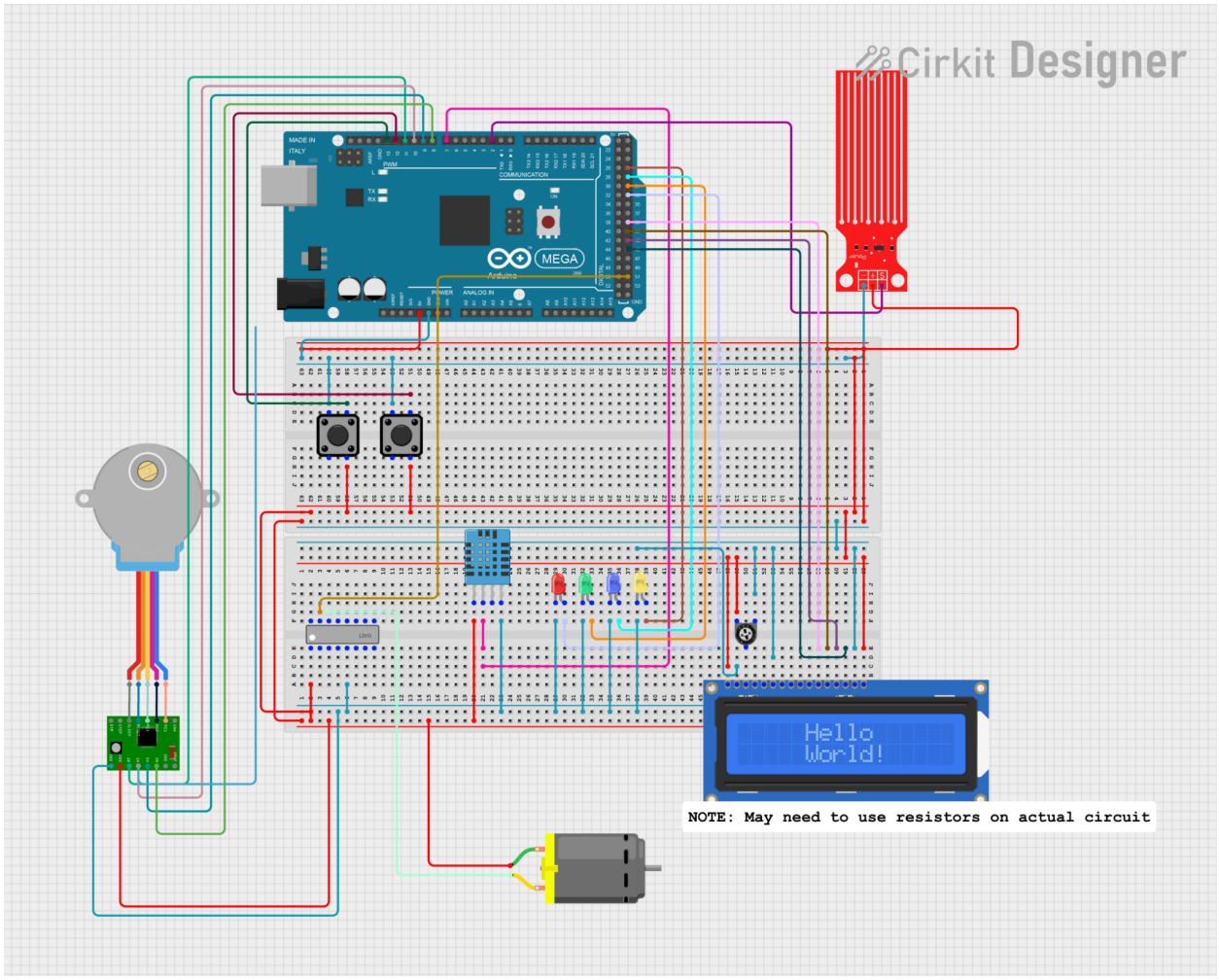
Now, the water level sensor checks the water level continuously, checking if the water level is within the threshold that is determined by the creator. This is done by defining the water threshold as WATER_THRESHOLD. The other definition would be WATER_LEVEL A1, which is the pin associated with the water level sensor. When the water level reads a water level too low for the water threshold, the error state is activated by the water level sensor. Otherwise, if the water level sensor reads a value that is within the threshold, the evaporating cooler functions properly. The function used to check water level is int checkWaterLevel(), this returns the water level back every time it is called.

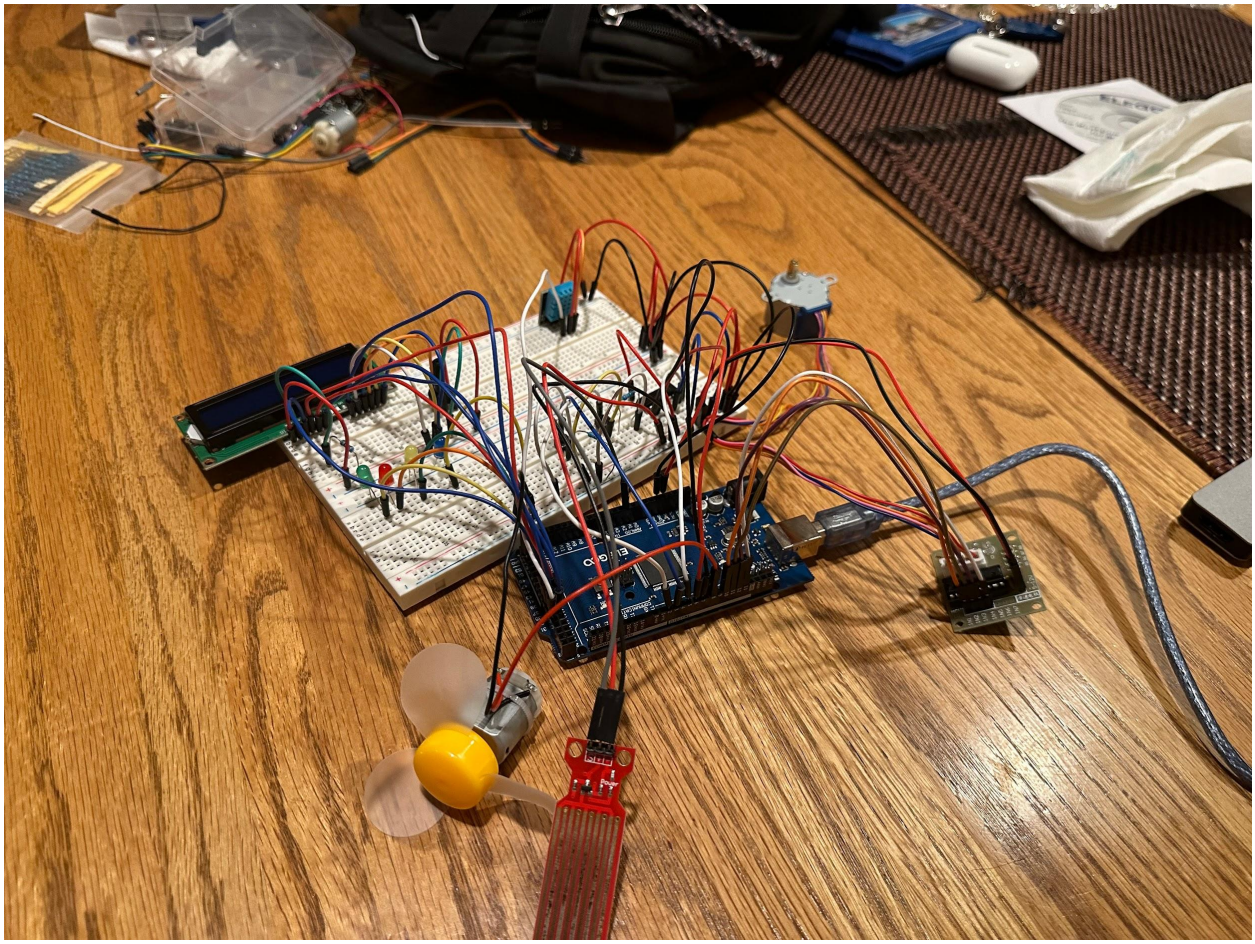
As for our buttons, there is an on/off and rotating stepper button that connects to the Arduino. The on/off button is the const int powerButtonPin 2. And the rotating stepper motor button is the const int stepperButtonPin 3. The on/off determines the on and off state of the arduino, and the rotating stepper determines the direction of the rotating stepper.

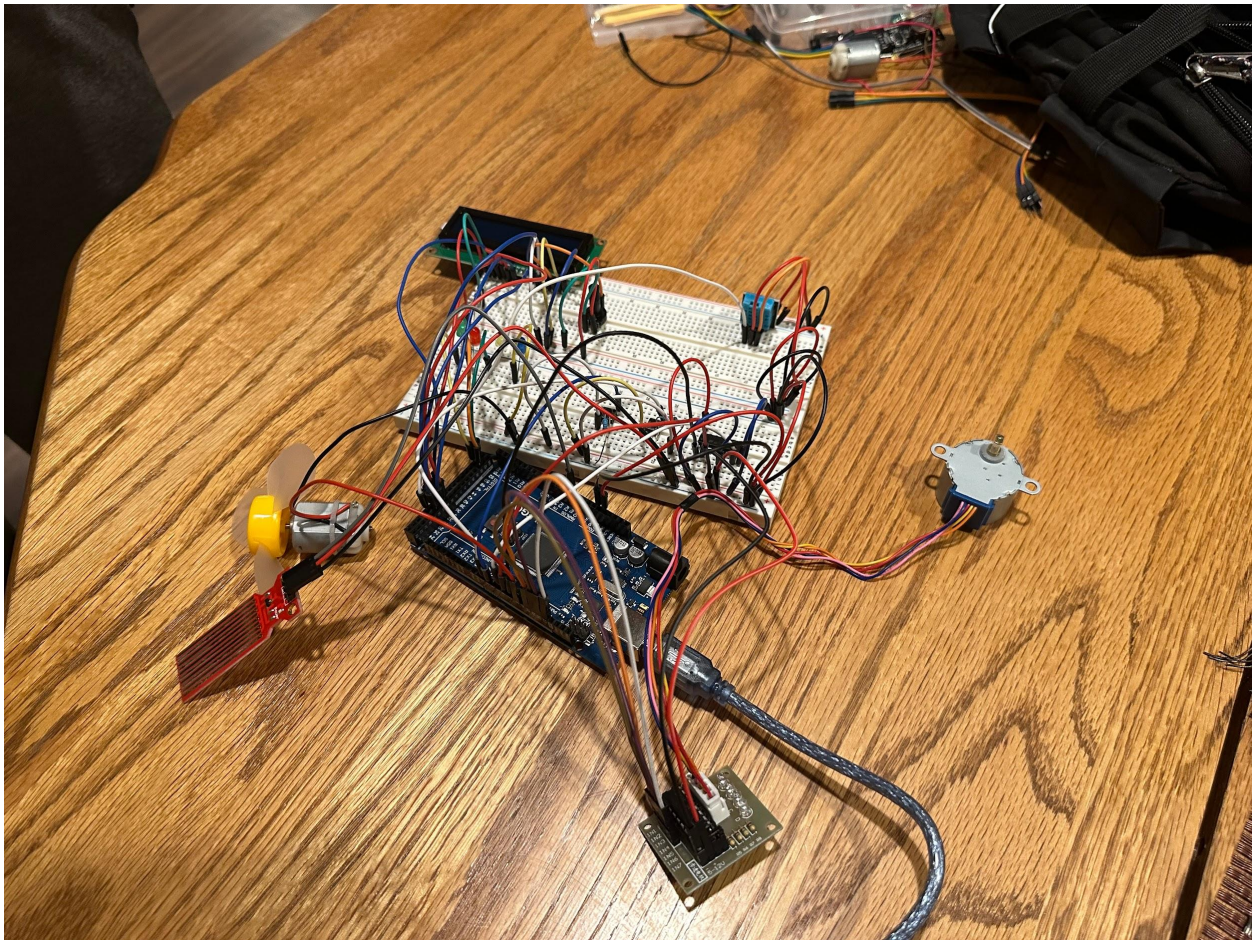
The stepper motor is a motor that functions as the directional motor component to the Evaporating Cooler. This direction being clockwise or counter clockwise is declared as Stepper stepper(10, 11, 12, 13), with the pins being 10, 11, 12 ,13 connected to the Arduino. The direction speed is 200, and the function moveVentClockwise(), and moveVentCoutnerClockwise(). This step didn't work properly, as the stepper motor could’t properly adjust its rotation.

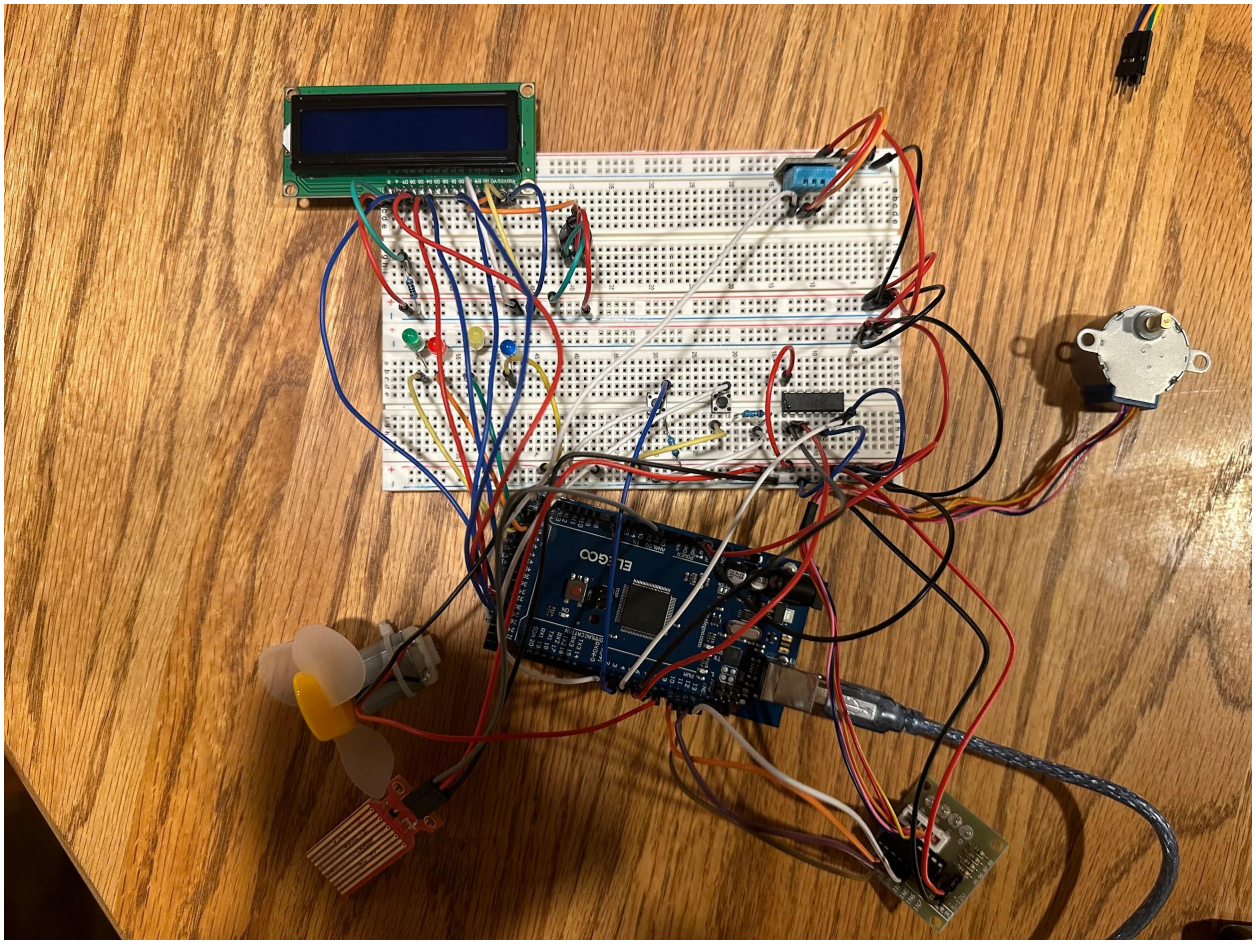
The fan motor functions to spin depending on the currency state of the machine. This fan is activated through the function setFan(), which functions the fan depending on a true or false parameter. This is used to either start or stop the fan. The pin used for the fan in pin 8.

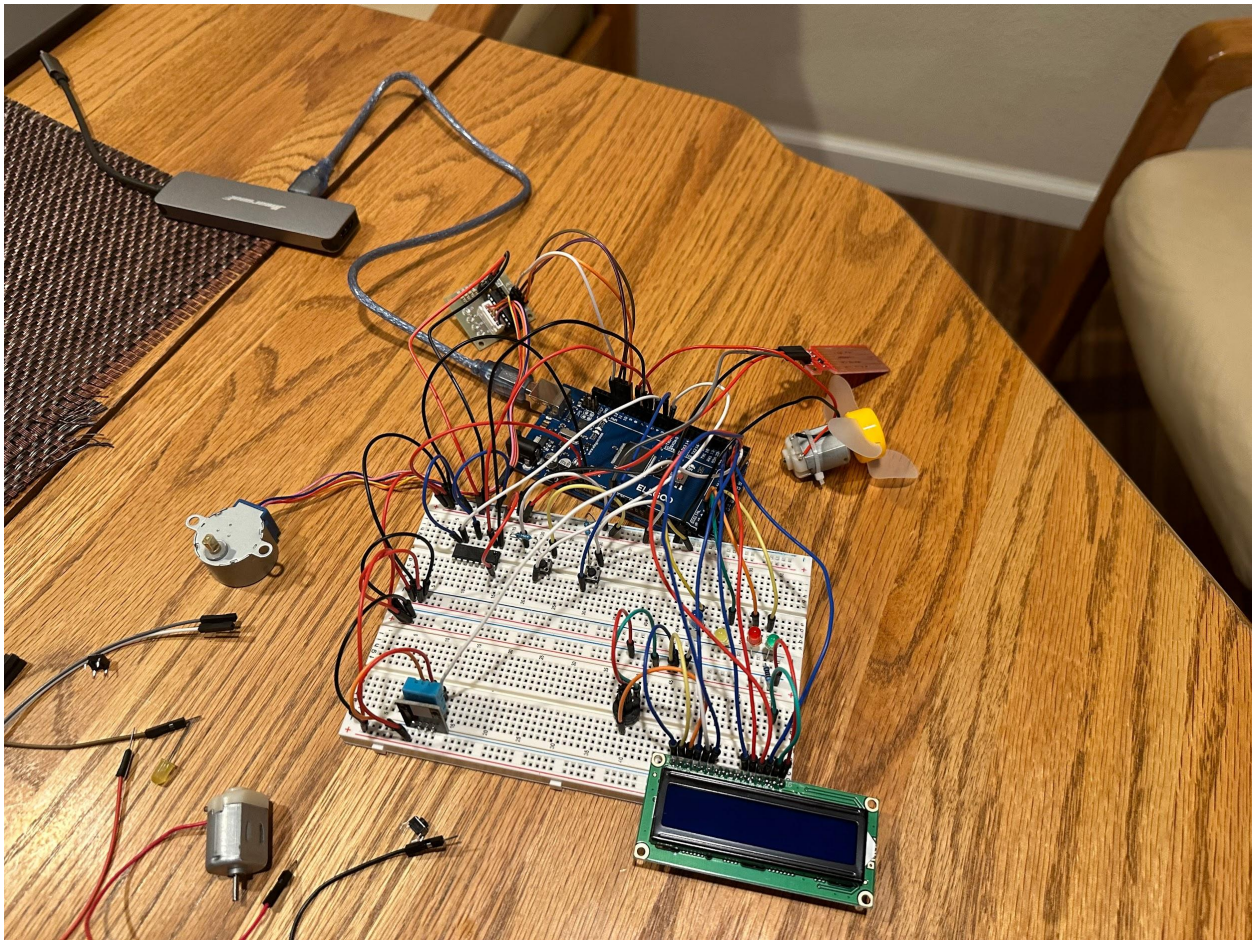
Finally the LEDs are used to display a physical representation of the states to the observers. This is done through the states Disabled(Yellow LED), Idle(Green LED), Error(Red LED), and Running(Blue LED). The Disabled state contains no display on the LCD, monitoring of the start button. The idle state records time stamps, while water level is being monitored incase of error. Error should trigger when water level isn't valid compared to the threshold, and runs an error message to the LCD. Running should start the fan, and check the threshold for triggering the idle. Should also be waiting for an error state if water is too low. These states are responsible for running their functions, based on check functions that are needed for each given state.


Cirkit Designer
MEGA
Hello
World!
NOTE: May need to use resistors on actual circuit









Link to video walk through: https://youtu.be/eDe6ptSnn1Y

Github link: https://github.com/Zolan-Oliver-1104/CPE301Final