Group Members: Bill Tong, Clifford Lewis, Jayden Davis
Date: Dec 3, 2021
Subject: CPE 301 Semester Project, Fall 2021

Introduction

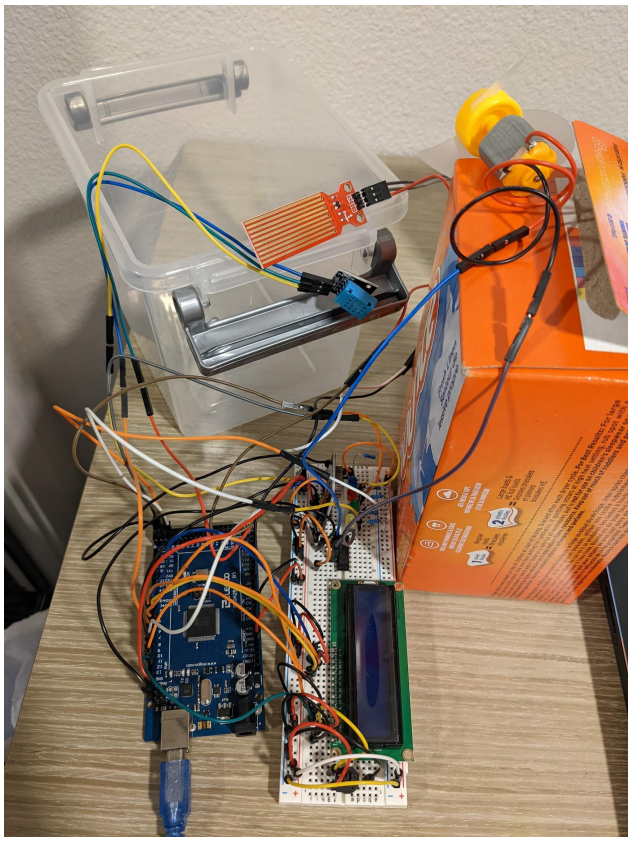
The purpose of this project was to expand our limited knowledge of embedded systems by collaborating as a team to design and implement an evaporation cooling system. For this particular swamp cooler model, the following materials were used:

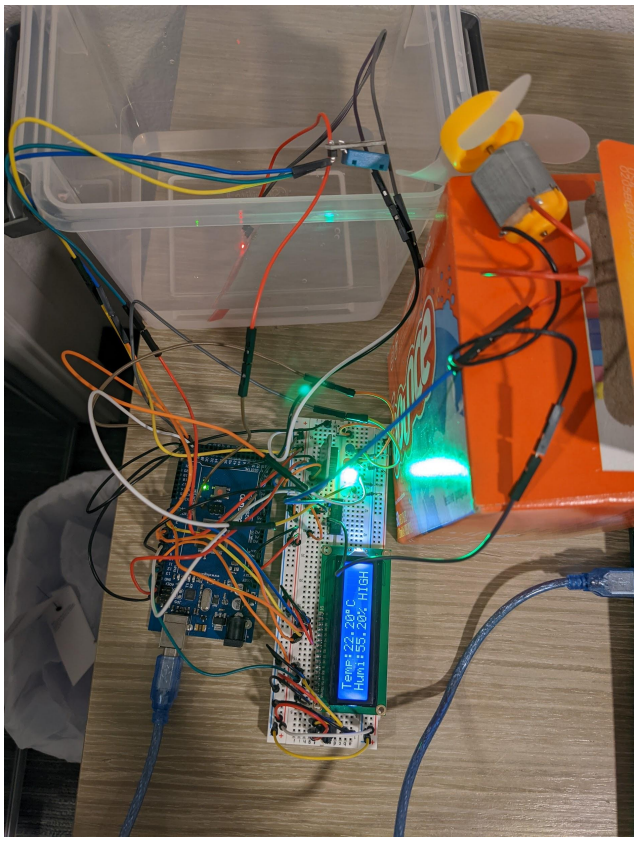
Hardware: Arduino Mega2560, LCD1602 Module, Water Level Detection Sensor Module, DS1307 RTC Module, DHT11 Temp/Humidity Module, Potentiometer 10K, L293D, Fan & 3-6V Motor, Breadboard, Jumper wires, USB cable, 220Ω resistors, 2kΩ resistor, button, LEDs(Yellow, Green, Red, Blue).

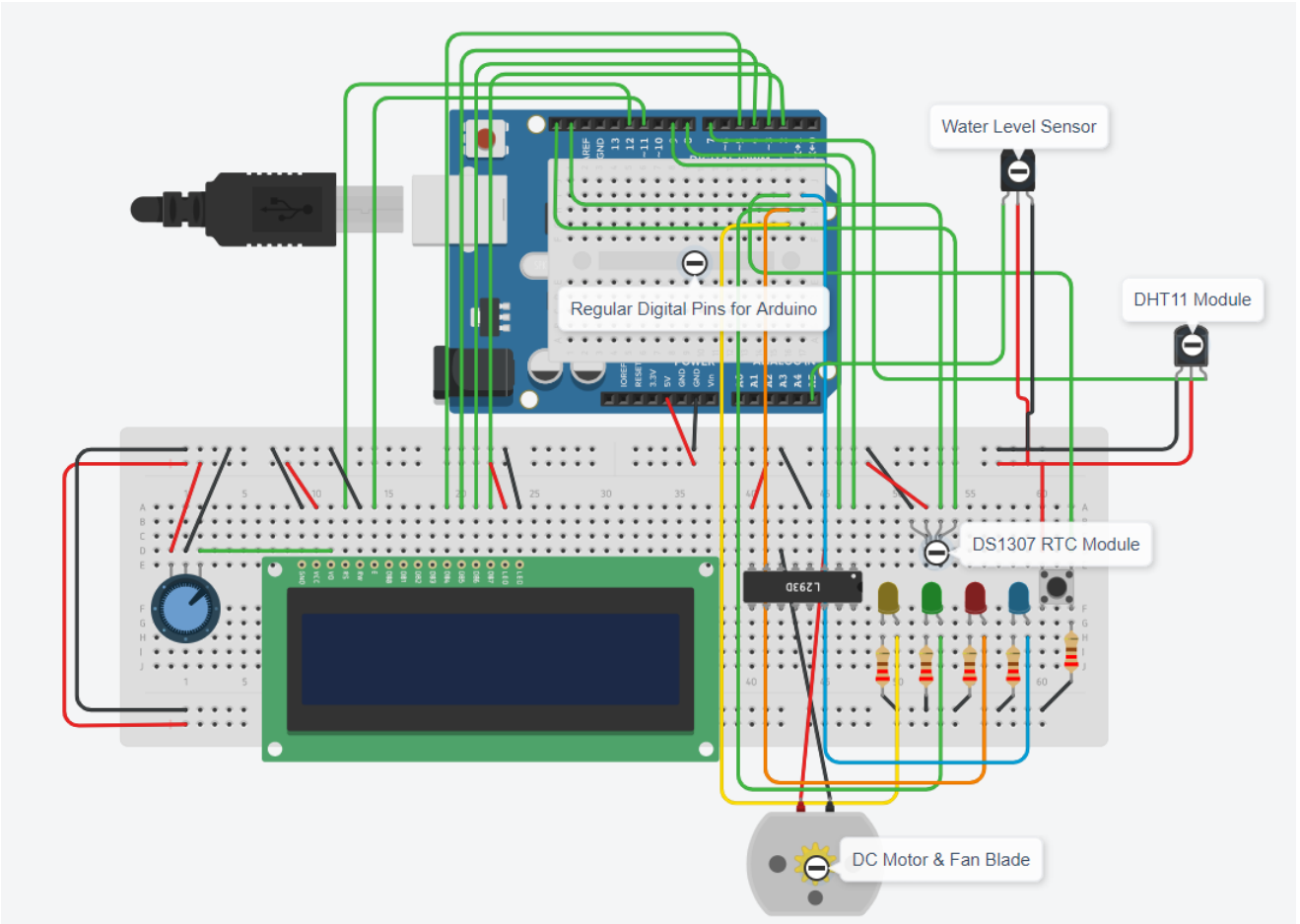
Software: Arduino IDE, TinkerCad, IMovie.

Design

This project was built in four separate stages instead of all at once, in order to minimize wiring errors, and maintain a compact system. In doing so, individual components could be tested and malfunctioning components could easily be pinpointed and corrected, saving time. Keep in mind that while everyone was given the same requirements, there are still multiple ways to implement the cooler. The first stage involved hooking the LCD to the arduino and displaying text in preparation for receiving water level, temperature, and humidity information. As this was the first component, the digital pins(PWM) were used. Next, the DHT11 and Water Level Detection Sensor Module were installed. These were responsible for collecting temperature in Celsius(℃), humidity in percentage out of 100, and water level of either empty(EMPT), low, medium(MED), or high. For these components, both a digital and analog pin were used. Then came the motor fan, which also needed to be connected through a L293D motor driver and used another PWM pin. The fourth stage involved setting up the RTC module for time/date recording, LEDs to indicate which state is operating at any moment, and a push button. The RTC module used SDA and SCL pins, while the cathode of the LEDs had 220Ω resistors connected to the ground, and the anodes using the regular digital pins. As for the push button, a regular digital pin was also occupied. Note that for the software portion, code was first assembled using non-approved library functions such as pinMode() and analogRead(), before transitioning to registers. This was done to make sure that the components worked individually and as a whole system before moving on, so that there would be no need to reconfigure the hardware setup later on. In regards to general libraries, <LiquidCrystal.h>, “DHT.h”, and “RTClib.h” were utilized. These were for controlling the LCD screen, DHT11, and RTC module respectively, and needed to be installed manually through the Arduino IDE library manager. Serial library methods were also implemented within the code, as it was also approved. Below are pictures detailing the hardware setup, along with the complete schematic. Both the working code and a video will be provided as well.




Temp: 22.20°C
Humi: 55.20% HIGH


Water Level Sensor
DHT11 Module
Regular Digital Pins for Arduino
DS1307 RTC Module
L293D
DC Motor & Fan Blade

Constraints

While this evaporation cooling system does present a somewhat realistic representation of large scale swamp coolers, several limitations apply, including operating temperatures, power requirements, water usage, and physical interference with the components. To begin, this cooler has been programmed to activate the motor fan and cool the area once the temperature has been detected to be 23.5℃(74.3℉), which is just above the conventionally room temperature of 21-22℃(70-72℉). This was due to our working environment, which was found to be always around that range, so we had to introduce a heat source such as our finger or blowing hot air into the DHT11 in order to make the fan run. Because of this outcome, we can confirm that the system needs an environment that is dry and at the same time hotter. In addition, the DHT11 is also a point of potential failure since the motor relies on the reading from this component to operate. If this component does not update as frequently as desired, or encounters some sort of error, then the whole system will fail as well. The vast majority of swamp coolers require some source of electricity or alternative power, and this one is no exception. To power it, we used the provided USB cable, which served as a source of power to the arduino and a way to execute code. An external battery pack could be a replacement for this, but it would add to the overall size of the cooler. A power supply Module could also help feed the power consumption of the fan and the LEDs, however, the system worked fine without it, so we opted to not include it in favor of conserving space. Next is water usage, a staple in all swamp coolers. This system needs a constant supply of water, since a swamp cooler without water would only be circulating hot air. The water essentially extracts heat from the hot air, and evaporates, which leaves behind air that is cool and moist. Lastly, through experience and troubleshooting, our group has found that a major constraint of this cooling model is that it can't be moved around while in operation. The wires connecting each component to their respective pin location on the arduino/breadboard are quite delicate, and accidentally moving them can cause them to malfunction. One example is the LCD, which starts reading gibberish when one of its wires is displaced slightly, or the motor deactivating if one of its wires connecting to the L293D driver is not tightly fit in the breadboard pin. Of course some of these constraints are not applicable to large scale coolers, but it's important to acknowledge them so that future designs can implement preventative measures or be revamped to completely eliminate them.

References

- https://content.arduino.cc/assets/Pinout-Mega2560rev3_latest.pdf
- https://www.instructables.com/Arduino-Button-Tutorial/
- https://images-na.ssl-images-amazon.com/images/I/D1oC-c3G5TS.pdf
- https://ww1.microchip.com/downloads/en/devicedoc/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf

Links

- https://github.com/unr-f21/cpe-301-semester-project-group-24.git
- https://www.tinkercad.com/things/6LmGBBHP0Ix

Note: Video was included in Canvas Submission as we encountered problems with video links.