Group: Saransh Beniwal, Michael Nia, and Wyatt Young
CPE 301.1001
Dr. Bashira Akter Anima
July 30, 2023

Swamp Cooler Project: Comprehensive Analysis

Introduction: Setting the Stage

The goal of this project is to create an evaporation cooling system, commonly known as a swamp cooler, which offers a more energy-efficient alternative to traditional air conditioners in dry, hot climates. Evaporative coolers work by pulling in outside air through a water-soaked pad, where the water's evaporation cools and humidifies the air before it is circulated indoors. This cooling concept serves to utilize various components such as a fan motor, temperature and humidity sensor, water level sensor, push buttons, liquid crystal display, and colored LEDs, to achieve precise control and monitoring of the cooling process.

Experimental Design: A Comprehensive Overview

The project aims to utilize an Arduino Mega 2560 for creating a swamp cooler, incorporating a fan motor powered by an extra supply and L293D IC, a DHT11 temperature and humidity sensor, a water level sensor, four push buttons, a liquid crystal display, and four LEDs (green, yellow, blue, and red) to signify various states of the LED: Not in use, disabled, active, and error. The cooler operates by monitoring the water level in a cup, issuing a warning if it gets too low, while also displaying the humidity and air temperature on the LCD panel. The fan can be controlled using an on/off button, and the system records the time and date when the motor starts or stops. Additionally, the vent switch buttons' range is also limited by the system.

The thermistor is employed to detect temperature within the temperature and humidity sensor (DHT11), which combines with a capacitive humidity sensor to measure humidity. Once the readings are recorded, a digital signal is transmitted to the data pin. The sensor is defined with DHT_PIN 7 and DHT_TYPE DHT11. Utilizing the DHT.h library's readTemperature() and readHumidity() functions, the system records temperature and humidity readings from the DHT11 sensor as they are obtained. The temperature reading alternates between the running and idle states of the engine.

The water level sensor monitors the quantity of water it detects, requiring recognition of nearby liquid to switch between idle/running and error states. The system utilizes the adc_read() function to verify the water level after setting the specified pin as WATER_LEVEL. If the value falls below the WATER_THRESH, representative of the water threshold value, the current state is returned as error.

In our design, we incorporated four push buttons, each serving a specific purpose. To toggle between the disabled and enabled states, the BUTTON_STATUS button is utilized. Additionally, we included a second button, BUTTON_RESET, to bring the system back to its default idle state when inactive. Moreover, we added two extra push buttons to control the step motor and adjust

the vent's direction. In order to accommodate these adjustments, limitations were set for the switch button settings, known as SWITCH_STEPPER_HIGH and SWITCH_STEPPER_LOW.

The liquid crystal display shows the temperature and humidity of the surrounding air. Our investigation confirmed a temperature of 20°C, cross-validated using the DS1307 module to ensure precision. If the water level drops too low, the LCD enters an error state and collaborates with the water level sensor to display the message "Error! Water level is low!"

For the display and its pins, we utilize the LiquidCrystal.h library, designating it as lcd. To establish the DateTime() function for the expected results, employ RTClib.h.

The fan motor operates in conjunction with the L293D IC chip only when the system is enabled with proper water levels and temperature values. The FAN_MOTOR_PIN defines the motor's behavior, allowing it to switch between the states of being disabled, idle, and running, depending on the situation.

The motor's direction of rotation can be controlled by using the direction control pins in conjunction with the L293D pinout chart. Furthermore, the motor is supplemented by the power supply, providing an additional direct power source.

In the build, four different colored LEDs are integrated to represent each state of the project's components. The green LED is illuminated when the circuit is not in use, indicating an idle state. The yellow LED signifies that the system is disabled, while the blue LED indicates an active state where all requirements are met. On the other hand, the red LED lights up to indicate an error, signaling that the idle state has been disrupted.

Each LED has a distinct pin assigned to it, and they are designated by their unique colors to differentiate between the various states of the system.

Result:

The experiment was a success as all components operated as intended. The LEDs accurately represented their respective conditions, with each color correctly indicating the system's states - green for "Not in use," yellow for "Disabled," blue for "Active," and red for "Error." While the current machine status and real-time information on the humidity and air temperature were working perfectly and providing to the output, the LCD display had issues displaying them. In terms of functionality, the fan functioned efficiently, responding to the system's control and turning on and off appropriately (Check video in resources for result).

Pictures of Circuit:

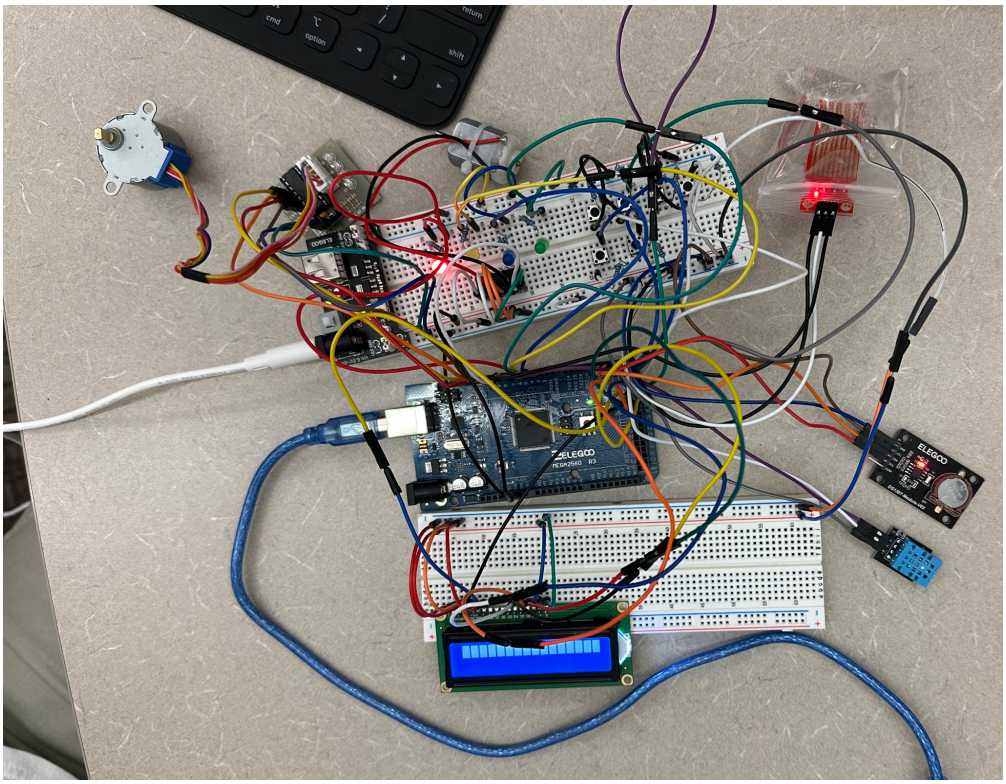

Image 1: Swamp Cooler Circuit picture

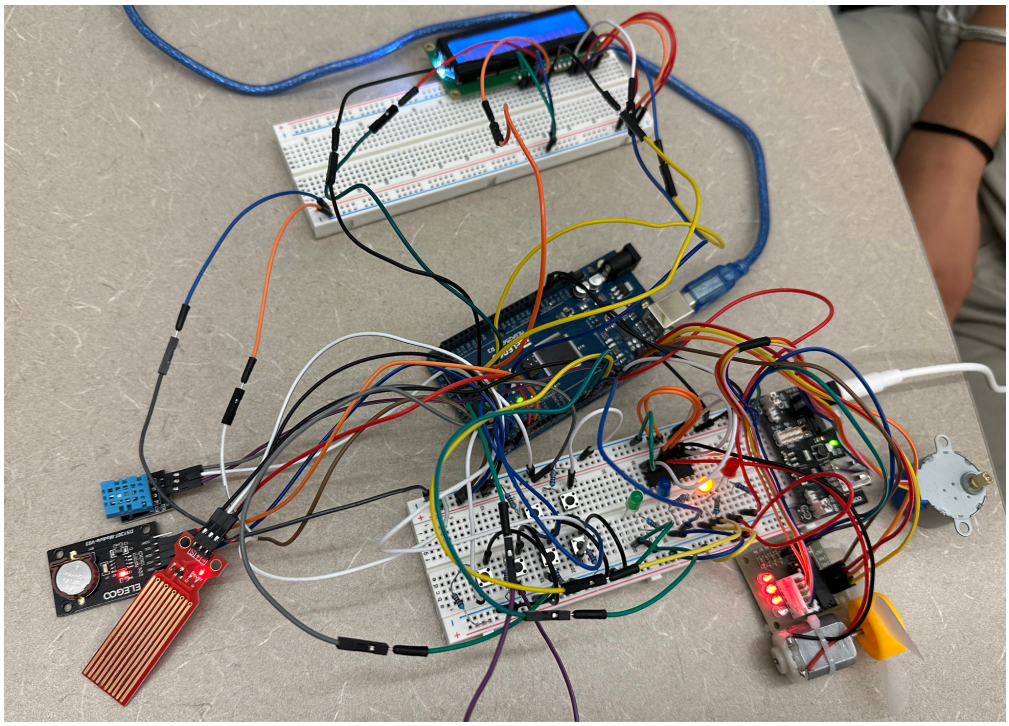

Image 2: Swamp Cooler Circuit picture

Circuit Schematic:

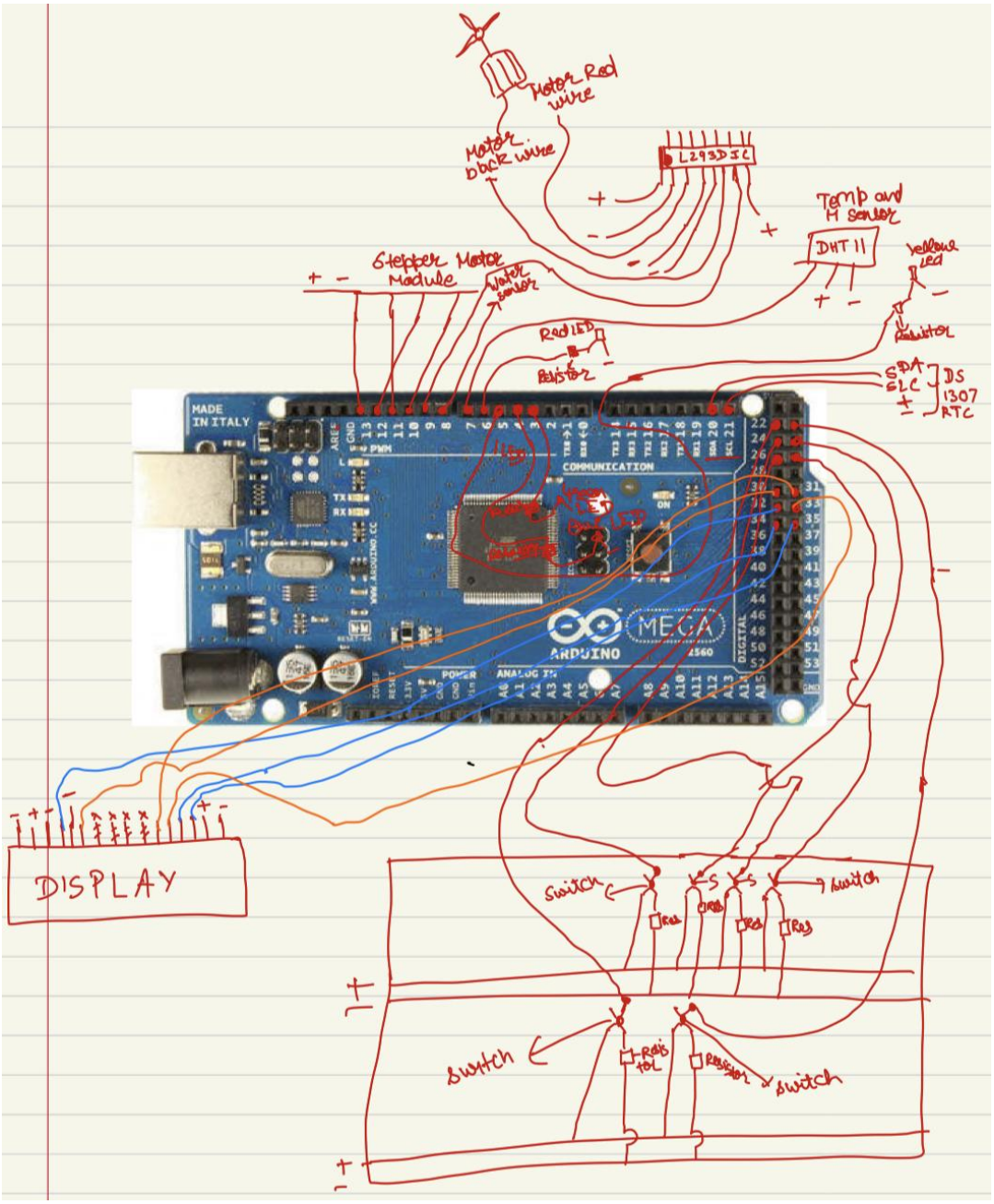

Image 3: Swamp Cooler Circuit schematic

Resources:

GitHub Repository:

https://github.com/Beniwal-Saransh/CPE_301_Final_Project

Video:
[Provided in the github repository]

Additional Resources used:

Arduino
https://ww1.microchip.com/downloads/en/devicedoc/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf

Display

https://toptechboy.com/arduino-tutorial-48-connecting-and-using-an-lcd-display/

clock
https://howtomechatronics.com/tutorials/arduino/arduino-ds3231-real-time-clock-tutorial/

https://www.analog.com/media/en/technical-documentation/data-sheets/ds1307.pdf