CPE 301 Embedded Systems Design Final Project

Mason Haines, Austin Jarolimek, Samuel Mouradian

May 7, 2024

– Introduction

This lab project focuses on developing a cooling system, also known as a swamp cooler, using the Arduino 2560 microcontroller and sensors from a standard Arduino kit. The goal is to build a sudo-functional cooler capable of cooling and humidifying air. To automate cooling using an Arduino Mega 2560, integrate sensors like a water level sensor and a DHT11 for environmental monitoring, along with actuators such as a stepper motor for vent control and a fan for cooling. Use Arduino libraries and directly manipulate register values for precise control and display information on an LCD screen. Include a real-time clock module for logging events. System states manage continuous monitoring, vent control, and motor activation, transitioning between IDLE, ERROR (for low water levels), and RUNNING (during cooling). This setup optimizes automation and control for efficient cooling processes.

Equipment

- Arduino Mega 2560
- USB programming cable
- Resistors
- Breadboards
- Jumper Kit
- Female to Male Wires
- LCD Screen
- DHT11 Temp Humidity Sensor
- Water Level Detection Sensor
- LEDs
- Potentiometer
- Buttons
- Power Supply Module
- 9 Volt Battery
- 3-6 V DC Motor
- L293D Motor Driver
- Stepper Moter (28BY J-48)
- Stepper Motor Driver Module
- Fan

– Component Experimental Design Requirements

Design Requirements

1. **Water Level Sensor:**
   - Utilize the water level sensor from the kit for monitoring water levels. Threshold detection can be achieved using either interrupt-based comparator methods or ADC sampling without relying on the ADC library.
2. **Stepper Motor:**
   - Control the vent direction using a stepper motor, which can be operated via buttons and/or a potentiometer to adjust the vent's orientation. Leverage Arduino libraries for stepper motor control.
3. **LCD Screen:**
   - Display essential messages on the LCD screen using the Arduino library.
4. **RTC Module:**
   - Employ the real-time clock module for event reporting purposes, integrating Arduino library functionalities.
5. **DHT11 temperature/humidity sensor:**
   - Integrate the DHT11 temperature/humidity sensor for real-time environmental monitoring, utilizing the Arduino library for sensor data acquisition.
6. **Fan and Motor:**
   - Use the provided kit motor and fan blade for the fan motor. Ensure to connect these components using the separate power supply board to prevent damage to the Arduino's output circuitry.

– State Descriptions

- **All States:**
  - Utilize the real-time clock to log state transitions and vent position changes via the Serial port.
- **All States except DISABLED:**
  - Continuously monitor and display humidity and temperature readings on the LCD screen, updating once per minute.
  - Respond to changes in vent position control.
  - The stop button should deactivate the fan motor (if running) and transition the system to the DISABLED state.
- **DISABLED:**
  - Illuminate the YELLOW LED.
  - Cease temperature and water level monitoring.
  - Utilize an ISR to monitor the start button.
- **IDLE:**
  - Log precise transition timestamps using the real-time clock.
  - Continuously monitor water levels and transition to an error state if levels are insufficient.

- Activate the GREEN LED.

- **ERROR:**
  - Deactivate the motor and prevent activation regardless of temperature.
  - Use a reset button to return to the IDLE state if water levels are restored above the threshold.
  - Display an error message on the LCD screen.
  - Illuminate the RED LED while deactivating other LEDs.

- **RUNNING:**
  - Activate the fan motor.
  - Transition to the IDLE state once temperatures drop below the set threshold.
  - Transition to the ERROR state if water levels become critically low.
  - Illuminate the BLUE LED.

– Final Project Pictures

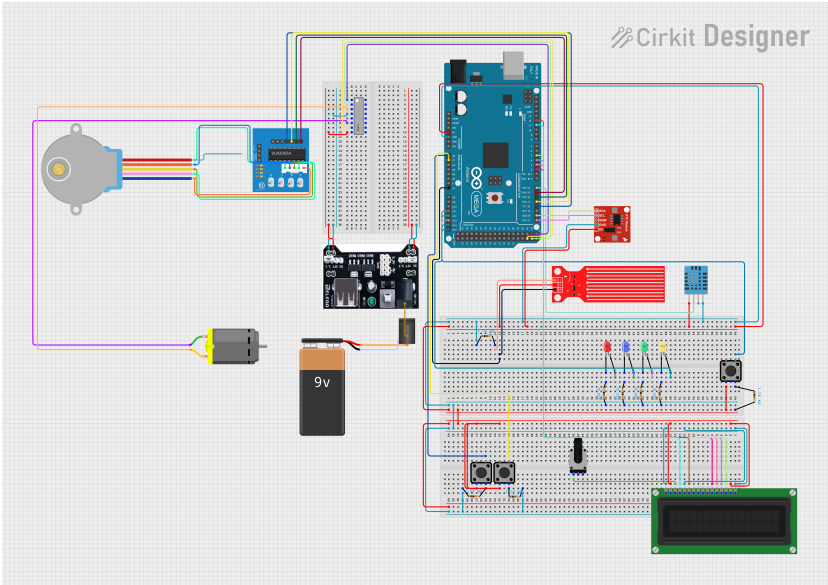

Figure 1: Circuit design schematic built on Cirkit Designer.

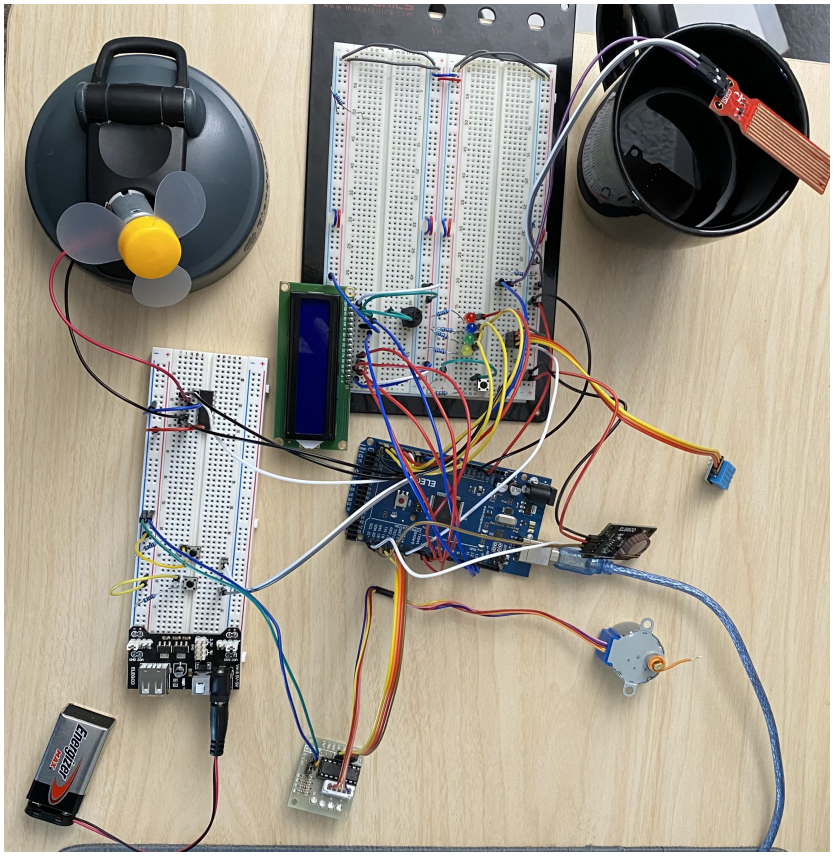

Figure 2: Final circuit setup.

– Contributions

- **Mason Haines:**
  - Fan motor code/circuit
  - Stepper motor code/circuit
  - Implemented code modules into driver
- **Austin Jarolimek:**
  - Temperature/Humidity sensor code/circuit
  - Water level sensor code/circuit
  - Lab write-up and documentation
- **Samuel Mouradian:**
  - My delay code
  - RTC code/circuit
  - Circuit design schematic
- **All:**
  - Circuit implementation
  - Final driver debug
  - ISR implementation
  - GPIO research

– References

- **Component Links:**
  - **DHT11 Temp/Humidity Sensor:** https://www.circuitbasics.com/how-to-set-up-the-dht11-humidi
  - **Water Level Sensor:** https://arduinogetstarted.com/tutorials/arduino-water-sensor
  - **LCD Display:** https://arduinogetstarted.com/tutorials/arduino-lcd
  - **Stepper Motor:** https://lastminuteengineers.com/28byj48-stepper-motor-arduino-tutorial/
  - **DC Motor:** https://toptechboy.com/arduino-tutorial-37-understanding-how-to-control-dc-motor
  - **Real-Time Clock:** https://howtomechatronics.com/tutorials/arduino/arduino-ds3231-real-time-c
    #google_vignette
- **Github Link:**
  - **Github Repo:** https://github.com/masonhaines/CPE-301-Final-Project-