CPE 301

Final Project

Evaporation Cooling System

Jennifer Hogueison

Enekoitz Irujo Urdiroz

**Project Description:**

This lab consists of us designing and implementing a working evaporation cooler with the Arduino kit. The main purpose for the cooler will be to continuously check the temperature and if it goes past the predetermined temperature it would activate its cooling system and begin functioning. Once powered the cooling system has four states, disabled, idle, running, and error. In the disabled state, the system is powered, but it does not perform any actions. When the start button is pressed, it will move to the idle state, where it will check the temperature every minute until it detects that the temperature has passed the threshold value and it proceeds to switch the system to its running state. While in the running state, the cooling system will be turned on until it detects that the temperature has fallen below the threshold, at which point it would be set to idle again. The last state that the system will have is error, this state will only be triggered when the water level drops below the predetermined water level limit. If that happens, the system will send an error signal and most of the systems will be turned off. To build the cooling system we implemented a variety of components from the Arduino kit, including a DC motor for the fan, a water level sensor, a stepper motor for the fan vent, a RTC module, four LEDs of different colors, buttons for moving the vent and the start and reset functions, LCD display for showing messages, the DHT11 humidity sensor to check for temperature and humidity.

**Component Details:**

**DC motor:**

We used the DC motor to power the fan of the system. The Implementation for it was very simple as we had to conly power it once the system had entered its running state.

**Water Level Detection Module:**

The water detection sensor was used to trigger our error state. Whenever it detected that the water level in the water reservoir was too low it would trigger the system to switch to that state.

**Stepper Motor:**

We used a stepper motor to move the vents on the fan. The way we decided to implement them is to have the motor move the vents 30 degrees up or down each time the corresponding button was pressed.

**RTC Module:**

We used the RTC module to be able to keep track of the times that the motor was turned on and off. The implementation of this turned out to be more complicated than expected, we ran into issues of the proper date and time not appearing or just having random characters be outputted.

**LEDs:**

We used 4 LEDs to show what state the cooling system was on, yellow for disabled, greed for idle, blue for running, and red for error.

**Buttons:**

We used a few buttons in system 2 for moving the vents up and down, another was implemented as a start, and the last is a reset to get out of the error state once the water has been refilled.

**LCD Display:**

The display is mostly used to display the current temperature and humidity in the area, as well as to display the error message whenever the system enters the error state.

**DHT11 Humidity Sensor:**

The humidity sensor does exactly what it’s named for, it detects the humidity, as well as the temperature of the air, and sends that information to the system. This is what mostly determines what state the system will be in. If the temperature gets too high the system will switch to the running state and if it falls below that point it goes back to the idle state.

**Circuit Image:**

**Schematic Diagram:**

**System Demonstration:**

**Submission Links:**