**Project Overview:**

Objective:

This project required us to simulate the functions of a swamp cooler using Arduino AtMega2560 board. We needed to implement four different states for the machine to exist in, as well as build a moving fan component, a temperature and humidity sensor, and a water level sensor. We also needed to display information to the user through an LCD screen, all while using interrupts to change the state of the machine.

We decided that a state machine would be the best option for the task at hand, giving us four states to build for. Not all of the problems were well suited for the use of interrupts, many were much more logical to implement as polling, as seen in monitoring water levels and temperature. But the servo and the on/off button were both implemented using interrupts in order to get at one of the large aims of the project.

Challenges:

We largely found issue on the wiring and debugging side of the project, as the concept itself was not too complex. Our primary problem came with getting our interrupt buttons to function correctly. We had to implement a “debounce” routine in our interrupts to insure that we were getting accurate readings. This coupled with a pull-down resistor allowed us to get accurate readings consistently from our buttons. This primary problem had made quite a number of smaller “symptom problems” by misleading our debugging work down our code for the issue and not our wiring.

Besides this, we found difficulty in deciding which parts of the project to be polling. We decided to make two button interrupts, to move the fan and turn the machine on and off, and have the sensors work off of polling.

Implementation:

The machine will boot through its setup function, which calls a number of smaller “initialize” functions for the LCD, Servo, Sensors, interrupts, and what will be called the state. The state determines what operations the machine will be performing on any given loop and is initially set to 0. After the machine gives a brief greeting, it goes into the “off” state, in which the LCD displays “OFF” and no polling is done and the interrupt for the servo is disabled.

Once the on/off button has been pressed, the system moves into the “idle” state; here, the servo can be freely moved, the temperature and humidity are monitored along with the water level (incase the water level drops while in this state), and the LCD displays the current readings.

In this state, if the temperature goes above 26, the fan will turn on and the state will transition into the “running” state. Here the temperature and water level are both still monitored, because the water level holds a higher precedence in the state machine. If the temperature falls back into designated zones, the fan will turn off and the machine returns to idle state.

But if the water level becomes too low, even if the machine is in running state, it will override and go into the “error” state, where the fan is off and cannot be moved, the only thing polled for is the water level until it reaches designated zones. After which it will return to idle or running based on given parameters.

Test Plan:

1. Turn on system
2. Press the state change button and wait for the state to change to green
3. Take water sensor out of water and wipe clean to enter the error state
4. Raise temperature to show that it will not change into the running state while in the error state
5. Resubmerge the water level sensor and return to idle state
6. Raise temperature to enter the running state
7. While in this state unsubmerged the water sensor and wipe off and show transition to error state
8. Submerge sensor
9. Return to idle
10. Press state button to disable state

Link to GitHub: https://github.com/mryamz/-Evaporation-Cooler