Final Project

University of Nevada, Reno
CPE 301 Embedded System Design
Instructor: Shawn Ray

Due December 13th 2022

Contents

1 Overview

The goal is to create an evaporation cooling system (aka a swamp cooler). In dry, hot climates, evaporation coolers provide a more energy efficient alternative to air conditioners. Air is pulled in from the outside through a pad that is soaked in water. The evaporation of the water cools and humidifies the air. As they rely on evaporation, they do not work in humid climates.

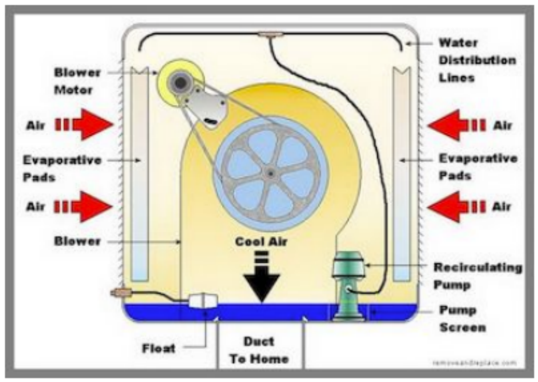

Figure 1: The general operation of the swamp cooler.

Figure 2: An image of a cooler without external panels attached so that the pad and fan are visible.

2 Project Requirements

Your team is to build a working cooler using the Arduino 2560 and sensors from the Arduino kit that we use for labs.

The completed project will

- Monitor the water levels in a reservoir and print an alert when the level is too low
- Monitor and display the current air temp and humidity on an LCD screen
- Start and stop a fan motor as needed when the temperature falls out of a specified range (high or low)
- Allow a user to use a control to adjust the angle of an output vent from the system
- Allow a user to enable or disable the system using an on/off button
- Record the time and date every time the motor is turned on or off. This information should be transmitted to a host computer (over USB)

3 Use of the Arduino Library

Unless it is specifically mentioned as being allowed, you may not use library functions such as pinMode, etc. You may use the predefined macros for registers and pin positions.

4 Component Selection / Design Requirements

- Water level monitoring must use the water level sensor from the kit. Threshold detection can use either an interrupt from the comparator or via a sample using the ADC.
  - You may NOT use the ADC library to perform the sampling
- The vent direction control must be implemented using the stepper motor. You can use either buttons or a potentiometer to control the direction of the vent
  - You may use the Arduino libraries for the stepper motor
- The LCD display must be used for the required messages (defined below).
  - You may use the Arduino library for the LCD.
- The real-time clock module must be used for event reporting.
  - You may use the Arduino library for the clock
- The temp/humidity sensor DHT11 must be used for the temp and humidity readings
  - You may use the Arduino library for this sensor.
- The kit motor and fan blade must be used for the fan motor.
  - Be sure to use the included separate power supply board! Connecting the fan directly to the Arduino can result in damage to the Arduino output circuitry.

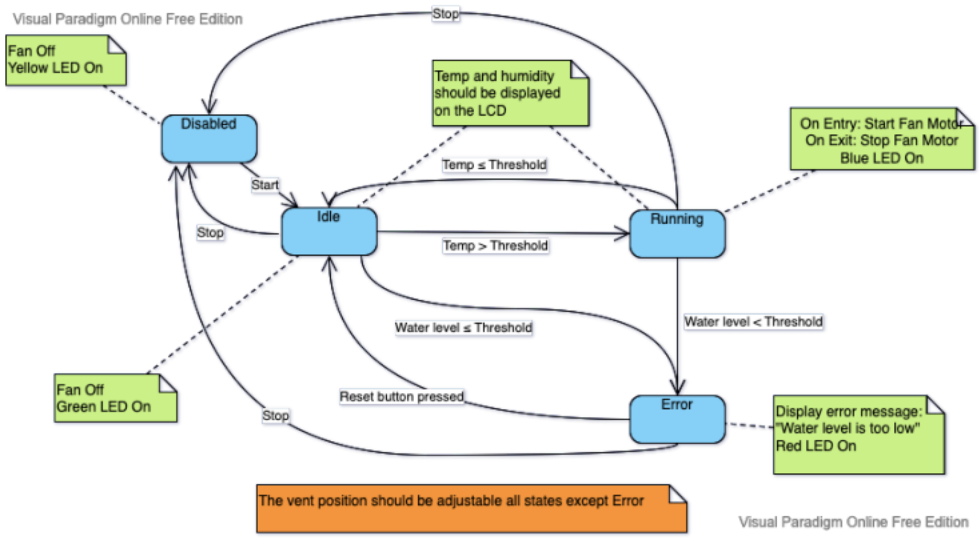

Figure 3: The state diagram for the operations of the swamp cooler.

4.1 Cooler States

The cooler continuously cycles through a number of states, with state transitions triggered by the user or by events such as temperature changes. The state diagram is shown below.

4.2 State Descriptions

Note: Some states include specific implementation requirements, such as requiring the use of an ISR.

- All States
  - The realtime clock must be used to report (via the Serial port) the time of each state transition, and any changes to the stepper motor position for the vent.
- All states except DISABLED
  - Humidity and temperature should be continuously monitored and reported on the LDC screen. Updates should occur once per minute.
  - System should respond to changes in vent position control
  - Stop button should turn fan motor off (if on) and system should go to DISABLED state
- DISABLED
  - YELLOW LED should be ON
  - No monitoring of temperature or water should be performed

Figure 4: An image of a cooler without external panels attached so that the pad and fan are visible.

  - Start button should be monitored using an ISR
- IDLE
  - Exact time stamp (using real time clock) should record transition times
  - Water level should be continuously monitored and state changed to error if level is too low
  - GREEN LED should be ON
- ERROR
  - Motor should be off and not start regardless of temperature
  - A reset button should trigger a change to the IDLE stage if the water level is above the threshold
  - Error message should be displayed on LCD
  - RED LED should be turned on (all other LEDs turned off)
- RUNNING
  - Fan motor should be on
  - System should transition to IDLE as soon as temperature drops below threshold
  - System should transition to ERROR state if water becomes too low
  - BLUE LED should be turned on (all other LEDs turned off)

5 Deliverables

5.1 Project Overview

- This document is a single PDF containing the following:
  - An overview of the design and any constraints on the system (example: operating temperatures, power requirements, etc)
  - Pictures of the final system and a link to a video of the system in operation
  - A complete schematic, and links to all relevant specification sheets for the components used
  - A link to the Github repository

5.2 Github Repository

The repository link must be turned in to WebCampus. The README for the project must include the group name and the names of all team members. The commits to Github will be reviewed, and will be assessed based on the comments (no "asdf" comments!) and the contributions from all team members.

5.3 Video Of Operation

A short video must be turned in showing the system in operation. There should be some narration as the project is demonstrated.