Yun-Ting Tsai

Professor Eric Schweitzer

CSCI 267

21 Dec 2022

Final Paper

Heat Index Controls Fan's Speed

I created a fan controller with a temperature & humidity sensor. I decided to use the heat index as the measurement instead of just the temperature or humidity because the heat index can more accurately detect how people "feel" about temperature.

The components and supplies needed for this project include

- Arduino UNO Board x 1
- USB A to B Cable x 1 // connect to the Arduino UNO board
- Wall Adapter Power Supply (12VDC 2A) x 1 // give the extra power to the 12V fan
- DHT22 Temperature & Humidity Sensor (4 pins) x 1 // detect temperature and humidity
- LCD Display Screen (16x2) 12C x 1 // display "Curr HI" and "Set Value"
- RGB LED light bulb (Common Anode) x 1
- 12V 2 Pin DC Fan (Size: 40x40x10) x 1
- N-Channel MOSFET (30V 60A) x 1 // electrical switch for turning on or off the fan
- Toggle Slide Switch (SPDT) x 1 // for turning on and off the machine
- Mini Push Button Switch x 4 // the "▲", "▼", "R", and "S" buttons
- 100 Ohm Resistor x 2 // for the LED light bulb (green and blue channel)
- 220 Ohm Resistor x 1 // for the LED light bulb (red channel)
- 10K Ohm Resistor x 7 // for push buttons, sensor, and MOSFET

- Jumper Wires (Male to Male) x 1 Pack (40 pins)
- Jumper Wires (Male to Female) x 1 Pack (40 pins)
- Full-Size Solderless Breadboard x 1

*** I purchased all materials on Amazon and Tinkersphere

How does this fan controller work? After connecting jumper wires with other components by following the circuit diagram, the next step is to plug the wall adapter into the Arduino UNO board. We need an extra power supply because Arduino can only supply 5V, but the fan used in this project needs 12V to operate. Use the USB A to B cable to transfer the code to the Arduino board and everything will work perfectly.

There is a toggle switch for turning the machine on and off. After turning on the machine, the LED light bulb will start blinking in blue color, and the LCD display screen will show the Current Heat Index and the Set Value that can be inputted by the user. There are 4 buttons on the breadboard, which are "▲" (increase the input value), "▼" (decrease the input value), "R" (reset the input value to default), and "S" (confirm the input value and start running the fan) buttons. Compared to the original proposal, I removed the "U" (change the unit of the heat index and input value from Fahrenheit to Celsius) button after consideration. Since people living in the US use Fahrenheit as the measurement, I don't think it's necessary to change the unit. Of course, after this semester, I can continue to develop this fan controller and build more functions.

The preset input value is 100°F. Users can set their value by pressing the "▲" and "▼" buttons. This value is the highest apparent temperature the user wants the fan to run at its maximum speed. Users can press the "R" button if they want to reset the value to default. After users confirm the set value and want to turn on the fan, they can press the "S" button. The LED bulb will stop blinking and stay on depending on the current heat index. There are 4 light colors

according to the heat index classification. The green light will stay on if the heat index is under 90°F. The yellow light will stay on if the heat index is between 90°F and 103°F. The orange light will stay on if the heat index is between 103°F and 125°F. The red light will stay on if the heat index is over 125°F.

I will create a 40°F interval for the fan to accelerate from speed 155 to max speed (255). My original proposal was to create a 20°F interval, but I realized the interval is too small, so I decided to make it larger. If the current index heat is higher than the user input value, the fan will run at the maximum speed. On the other hand, if the index heat is under the user input value minus 40, the fan will stop running. For example, when the user inputs 105°F, the fan will automatically adjust its speed if the heat index is between 65°F and 105°F. It will operate at the maximum speed if the heat index is over 105°F and turn off if the heat index is below 65°F.

Users can simply slide the toggle switch to turn off the machine. When the fan controller is turned off, the LED light bulb, the LCD display screen, and the fan are also turned off. For a better user experience, the machine will store the user's previous input value. Therefore, when users turn on the machine, they don't need to reset the value again. After the machine is turned on, the LED bulb will start blinking again and the fan will be turned off. The user has to press the "S" button again to stop the bulb from blinking and to run the fan.

For my code, I found a lot of tutorials and other people's programs online as references and combined their codes to create mine. To use DHT22 (temperature & humidity sensor), we need to download the "dht-sensor-library" library on Arduino, reading the documentation on https://www.arduino.cc/reference/en/libraries/dht-sensor-library/. To use LCD display screen 12C, we need to download the "liquidcrystal-i2c" library on Arduino, reading the documentation on https://www.arduino.cc/reference/en/libraries/liquidcrystal-i2c/. In these libraries, there are

example codes for testing our devices and components. I also got the basic code structures for the sensor and display screen from their examples. I started building my codes by understanding how the common anode RGB LED light bulb works, which I learned from the RGB LED tutorial (https://create.arduino.cc/projecthub/muhammad-aqib/arduino-rgb-led-tutorial-fc003e). For the next step, I connected the RGB LED with a switch by reading the "LED and Switch with Arduino Uno" tutorial (https://binaryupdates.com/led-and-switch-with-arduino-uno/). After the LED was working, I focused on the LCD display screen, the DHT22 sensor, and the fan. I created the "Set Value" on the screen and let the user can change the input value by following the "Arduino Counter with LCD display and Push button" tutorial on Youtube (https://www.youtube.com/watch?v=1cg9mXA2XRE). This tutorial is pretty helpful and also my main reference for my project because I can reuse its code structure to create other functions. To get a better understanding of how the temperature and humidity sensor works with the PWN fan, I watched many people's projects for references, but I mainly focused on the "Arduino Temperature Controlled Fan Speed" video (https://www.youtube.com/watch?v=DAn4UguyzfE) and the "Controlling fan speed with MOSFET and Arduino" video (https://www.youtube.com/watch?v=Pw1kSS_FIKk). The last step in completing my project is the functions after pressing the "Start" button. I read the "Temperature Based Fan Speed Control & Monitoring With Arduino" project (https://how2electronics.com/temperature-fan-speed-control-arduino/) to learn how to adjust the fan speed based on the temperature. I have trouble stopping the blinking LED and getting the light to stay on after pressing the "Start" button. Therefore, I found a tutorial to solve this problem (https://forum.arduino.cc/t/start-and-stop-a-blink-led-with-two-pushbuttons/697761/9).

This is my final project. Hope you find it interesting.

Circuit Diagram