5-2024

Temperature Controlled Box

Dennis Butts
Trinity University

Gabriel Alford
Trinity University

John Hawes
Trinity University

Holden Soderstrom
Trinity University

TRINITY UNIVERSITY

Temperature Controlled Box

Instructor: Dr. Kevin Nickels

Course: ENGR 4367-1

Group Number: 3
Pledged: Dennis Butts, Gabriel Alford, John Hawes, Holden Soderstrom
5/3/2024

Table of Contents

Design Summary

The temperature-controlled box is a project developed in the Spring of 2024. Its main goal is to maintain a setpoint temperature by switching both a heating element on for heating and a fan on for cooling. Figure 1 shows the final product for the temperature-controlled box.

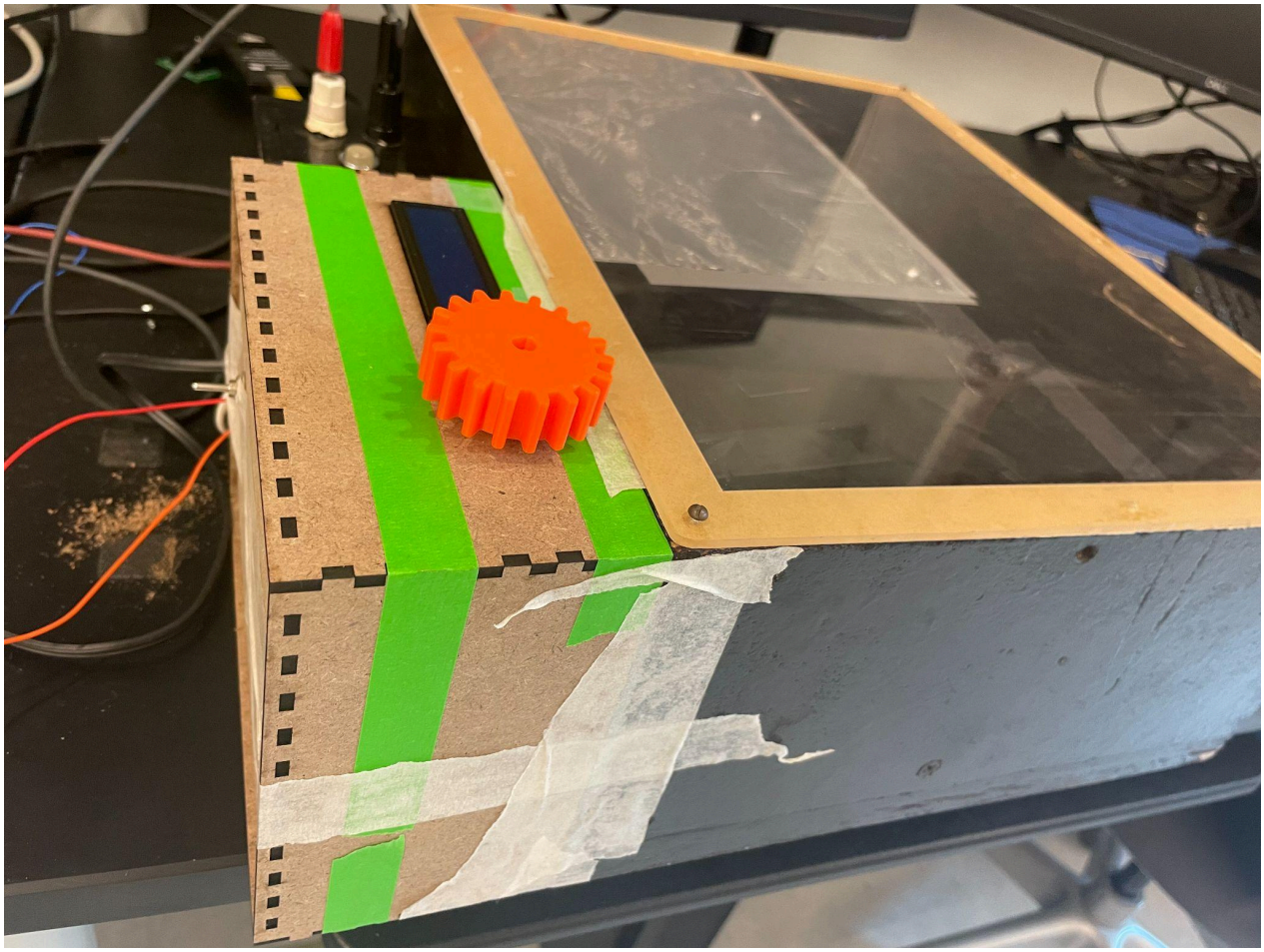


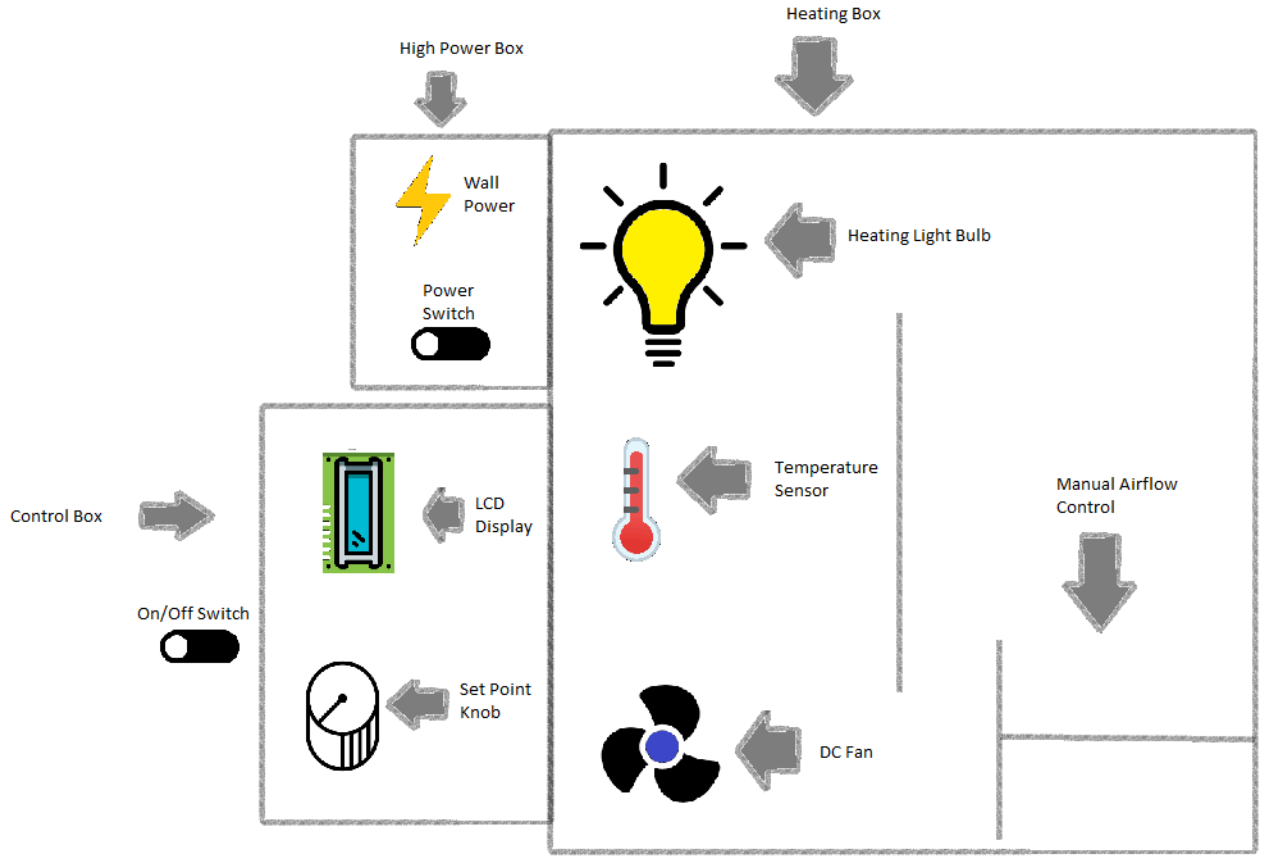
Figure 1: The final temperature-controlled box

The orange knob on the brown box in Figure 1 was a potentiometer used to set the setpoint temperature of the box. The PICs would use this setpoint temperature to determine whether to heat or cool the box. The temperature of the box was recorded using an LM35 temperature sensor. With some math, the voltage signal on the temperature sensor was converted to degrees Celsius. If the temperature was too far below the setpoint temperature set by the potentiometer, a lightbulb would turn on and heat the black box. If the temperature measured by the LM35 was higher than the setpoint temperature, the fan would turn on. A piezo buzzer would

turn on at 56℃ to indicate that the box would get too hot, and a switch was used to turn the controller on and off.

System Details

The temperature-controlled box would, of course, have a box to contain the heating element, but it also had attached modules to control the high-power elements and house the controller and interfacing elements. A basic sketch of these boxes and the visible components is shown in Figure 2.



The main *Heating Box* contained the light bulb that generates heat, a fan to dissipate heat, and a temperature sensor. The box also included a piston-like mechanism that was used to manually control the air circulation possible for the box. The *High Power Box* contains the elements that connect and distribute the high voltage from regular wall power. The light bulb would need to be powered from this box but controlled with the logic and program of the *Control Box*. The *Control Box* housed the microcontroller and other circuitry necessary for the user's

inputs to dictate the temperature inside the *Heating Box*. The *LCD Display* listed the setpoint and current temperatures, the *Set-Point Knob* could be turned to adjust the setpoint for the temperature, and the *On/Off Switch* turned the control system on and off.

Inside the *Control Box* is the majority of the design for the whole project; It contains two PIC microcontrollers, a signal amplifier, a buzzer, and a motor driving circuit. Figure 3 displays how all the components are interfaced and demonstrates the flow from inputs to outputs for the design.

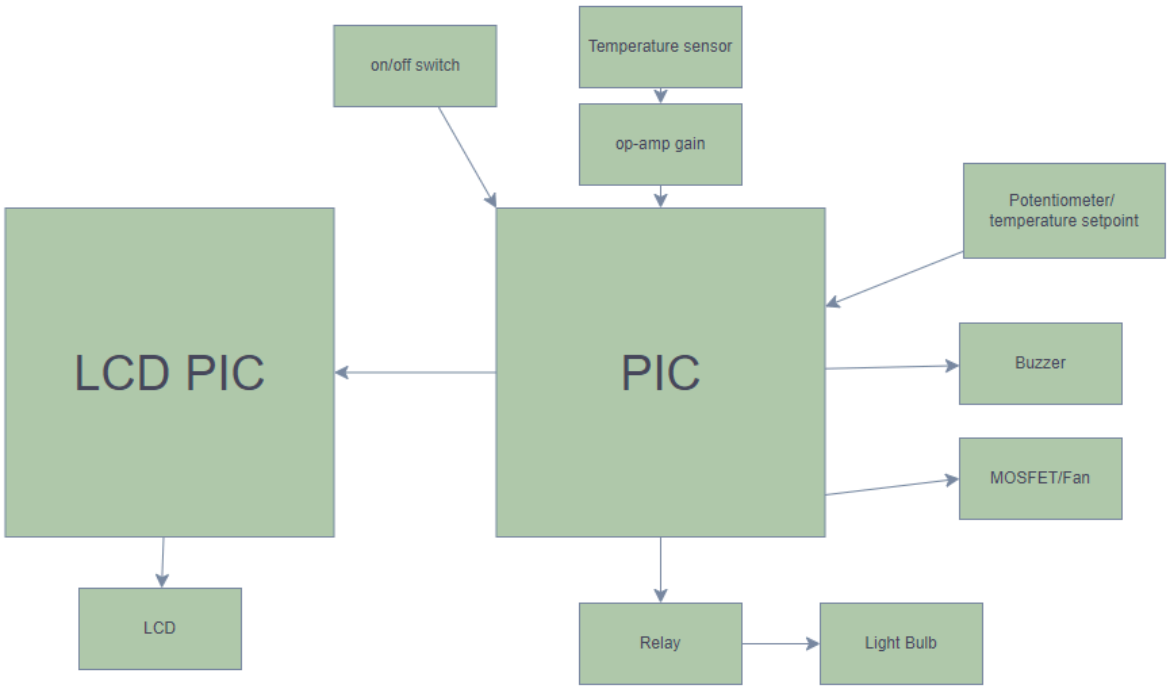


Figure 3: Functional diagram for all design elements

As depicted by Figure 3, the majority of the control is done from one PIC microcontroller. The PIC needs the on/off switch to be on so it can read the setpoint potentiometer and the temperature sensor. It uses this to determine the output for the LCD, the buzzer, the fan, and the light bulb. It controls the LCD by communicating with another PIC, which drives the LCD. For the fan, it uses a MOSFET to drive the fan connected to a higher voltage source. The buzzer is

driven directly with a square wave. Finally, the lightbulb is controlled via a relay which is connected to the wall power. Figure 4 and 5 depict the whole schematic of this design and exactly how each element is connected.

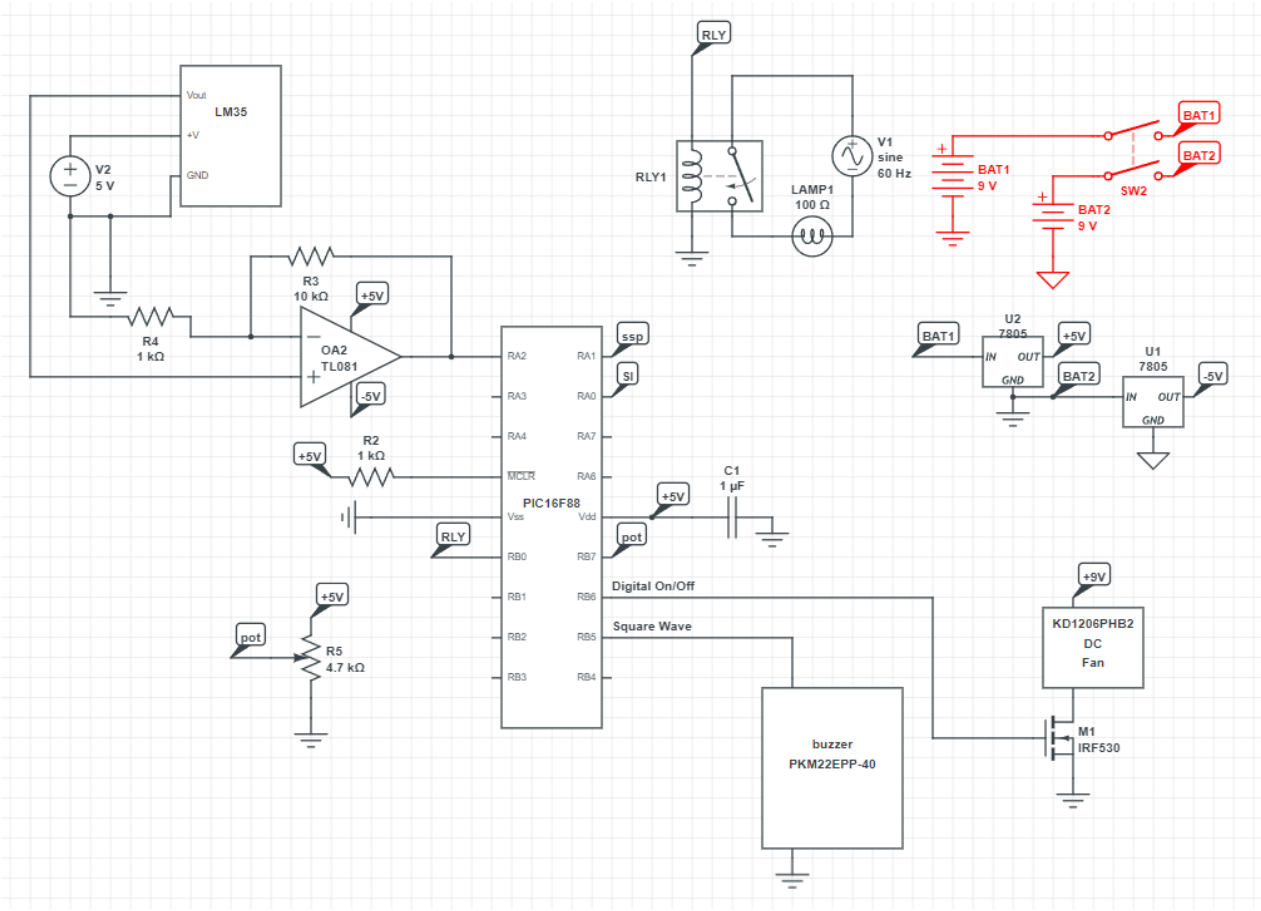


Figure 4: Final circuit without LCD display

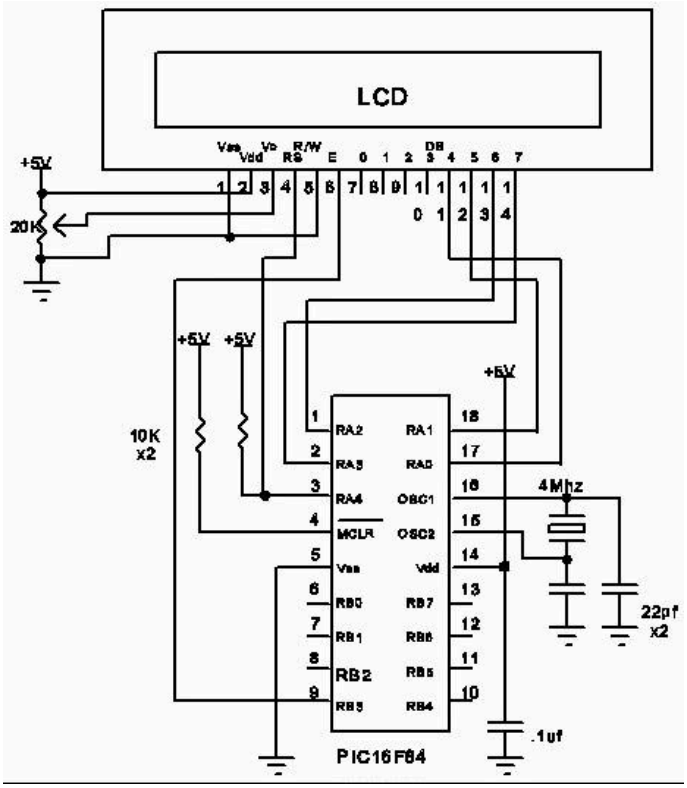


Figure 5: Circuit for LCD display

As shown in the circuits, the on/off switch directly closes the batteries into the loop and so when it is open, nothing is powered. The two batteries feed two voltage regulators that create positive and negative 5 V lines for the rest of the circuit. The -5 V is needed for the op-amp circuit which amplifies the signal from the temperature sensor. The programming of the PIC does the rest.

The PIC's program is designed to keep the box at a set temperature. Figure 6 shows the flowchart of the program.

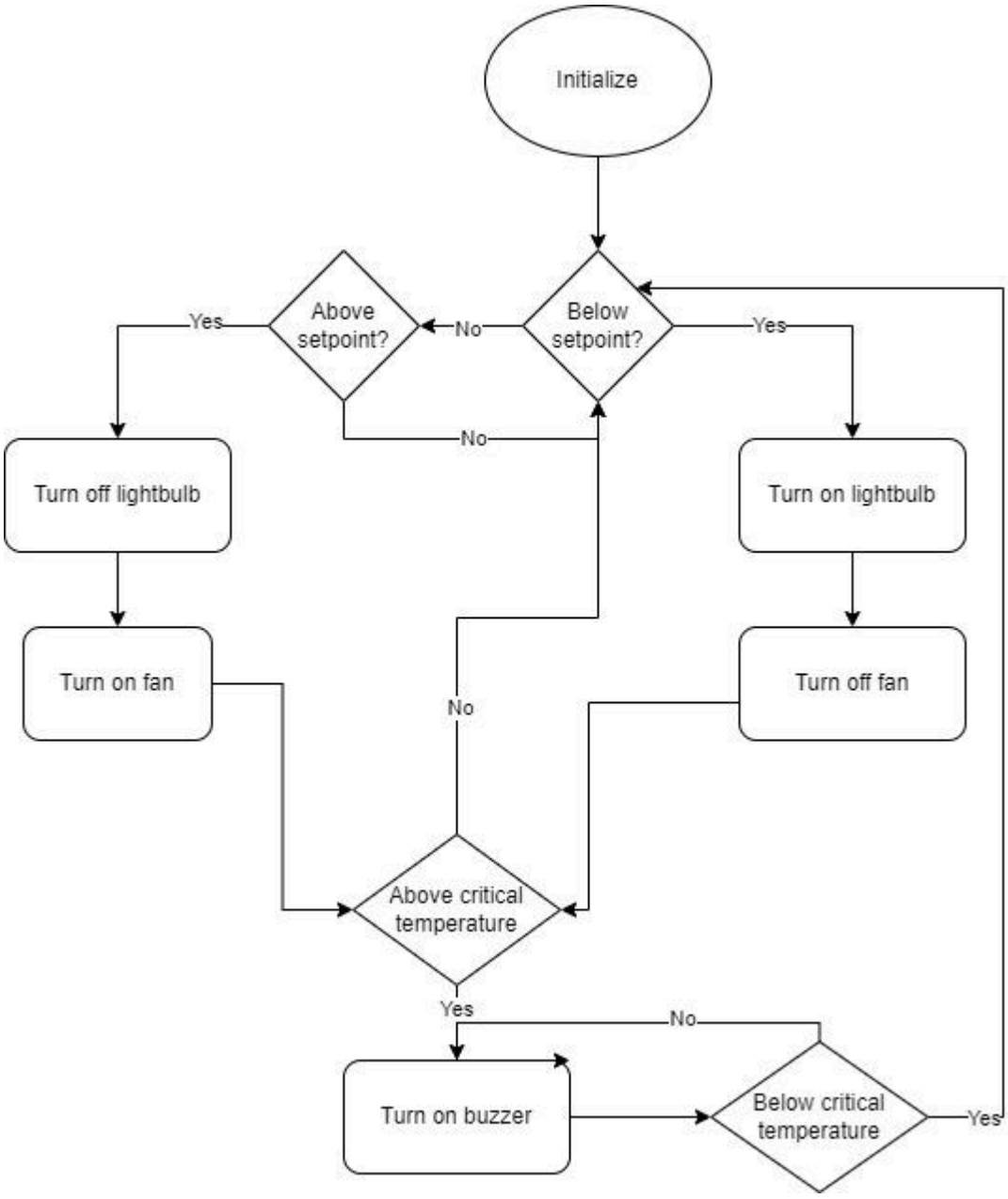


Figure 6: Software Flow Chart

Its first if statement is based on whether the temperature reading is above or below the setpoint. If it is below, it turns the light on and makes sure the fan is off. Otherwise, if it is too high, it does the opposite: turns the light off and the fan on. If the temperature is above the *critical temperature*, the buzzer turns on until the temperature lowers.

Design Evaluation

There are six elements that our design needed to have: an output display, an audio output device, a manual user input, an automatic sensor, an actuator or another type of mechanism, and some sort of logic, processing, and control. Our design project was able to meet all of these requirements, but for the most part, it did nothing truly extraordinary, merely what we needed it to do.

For our output display, we had an LCD display that displayed the current temperature of the system along with the temperature the system was attempting to reach (hereafter referred to as the setpoint temperature). This required some serial communication between PICs since a single PIC cannot run an LCD and all of the functions needed for our device, which somewhat increased the complexity of implementation. We believe that this aspect of our design earns a 6, as it consistently works and displays what it needs to whenever it is activated, but the process for making the LCD work is explained in detail in the textbook, and thus required no substantial research on our part.

For our audio output device, we used a piezo disk that would buzz whenever the temperature inside our system exceeded a set safety limit of 56 ℃. We believe that this aspect of our design project earns a 6 for similar reasons to our LCD display. The audio output works consistently, but a piezo disc is an easy audio output to use, so it required no real extra effort on our part to implement. Additionally, programming the disc to behave the way we desired wasn't terribly difficult, though there were some hiccups during troubleshooting.

For our manual user inputs, we had both a DPDT as an on-switch for the entire device and a potentiometer that acted as a means to control the setpoint. This section, we believe, also earns a 6, as neither of these devices was particularly challenging to implement, simply requiring the PIC to read an input for the potentiometer and a little bit of wiring for the switch. However, they do consistently function the way they're intended with repeated use.

For our automatic sensor, we used an LM35 temperature sensor in order to detect the current temperature inside the system. This specific device required a bit more external research than the others, as this type of sensor is an integrated circuit rather than a thermistor, thermocouple, or any of the more standard temperature devices brought up during class.

However, while there was a bit of research required to understand the device, it ultimately wasn't too difficult to get working and measuring properly and didn't require an insane level of research. Therefore, we believe that this section should earn an 8.

We had two actuators for this project: a heating actuator and a cooling actuator. The heating actuator was a relay that when energized, connected power to a light bulb, which would begin to warm the system. The cooling actuator was a fan controlled by a MOSFET and a digital output from the PIC which activated it when the setpoint temperature was below the current temperature. These actuators work perfectly, raising and lowering the temperature respectively, albeit at very slow speeds. However, they aren't really complicated and didn't require any real external research for us to make them function, so we believe that they earn a 6.

Lastly, our logic, processing, and control was entirely handled by the PIC. This required a bit of research in order to function properly, but it was mostly taken care of when trying to make the other components work individually. Aside from that, the control really boils down to an on-off controller for our various components, which isn't a particularly complicated bit of logic. Thus, we believe that this section also earns a 6.

Bill of Materials

Part	Model Number	Price	Source
MicroController x2	PIC16F88	Free	Electronics shop
Temperature Sensor	LM35	Free	Electronics shop
9 Volt Battery x2	6LR61	Free	Electronics shop
LCD display	HM1602A-4	Free	Electronics shop
DC Fan	MFD8025	Free	Electronics shop
Piezo Buzzer	AC2208	Free	Electronics shop
150 W Light Bulb		Free	Electronics shop
Relay	rssdn-10a	Free	Electronics shop

Lessons Learned

The team learned that integrating hardware and software systems is much more difficult than it first may seem. The biggest problem that the team faced on this project was implementing the LCD. At first, they tried to implement the I2C LCD screens. However, they found it easier to use the regular LCDs instead. They found that the LCD screen was the easiest to implement due to the existing functionality in the PIC framework. The team did find afterward that a library for I2C LCDs does exist, which would have made life much easier. Another hardware problem the team faced was when they implemented an op-amp gain for the temperature sensor reading. They had some difficulty implementing a non-inverting amplifier configuration. The op-amp kept on clipping despite the resistor values being correct. In the end, the op-amp configuration ended up working when the team switched to a different op-amp. Another problem that the team faced was when the team integrated the op-amp to the PIC. For some reason, not specifying the op-amp output as an input to the PIC caused the PIC to distort the output of the op-amp. All the team had to do to fix this was specify the PIC input as an input on the PIC Basic Pro code. It was an easy fix, but the team still had to spend some time to figure out the issue. The main takeaway from this is to allow for ample time to debug hardware because things almost never work theoretically as expected.

The second difficulty that the team encountered was putting the entire system together. They had components that kept on shorting together and disconnecting on them. Despite having an easier project, they still had issues with the amount of wiring due to the amount of requirements that exist in the project. It is important to make sure to allow for time to integrate components and wiring together seamlessly since things do go wrong once everything is put together.

Appendix

The team has no appendix section.