Barn Livestock Alarm System

Official Report

ENEL 387

Daris Lychuk

SID: 200361245

April 2019

University of Regina

Table of Contents

1.0 Design Process

1.1 Hardware Description

The design process I took worked best for me I believe. I gathered most of my parts before starting to design my project. Once most of the parts were gathered, and the remaining parts were ordered, I began to design my schematic roughly on paper. Due to forgetting how certain electrical components work, I asked how to set up my relay and transistor to power my halogen lamp. Later on, I also asked what would be the best way to set up my photo resistors, as well as a few other small things. Once I had multiple little schematics drawn on paper to go off of, I began building the actual design on a breadboard, which mainly consisted of jumper cables to connect everything. Jumpers are not the best solution due to the fact that they can easily be pulled out, however, I figured because I was using a breadboard it was the best I had. To make it easier, I labelled any jumper wire with important information; such as, which port it was connected to.

1.2 Software Description

Unlike some may have done, I did not begin coding anything until all of my components were in my board. When beginning to code, I tried to utilize as much code as I could from previous labs to get my design working. I wanted to start with my UART LCD because for whatever reason I thought it was going to be very easy to get it to work. I also knew the need for either this LCD or the TTL one on our 384 board as an output to the user. I managed to get characters printed to my UART LCD, however, they were the wrong characters. Additionally, I could not figure out how to properly reset the cursor and clear the entire screen. I was able to clear one space at a time on the UART LCD wherever the cursor was placed at that moment. This obviously did not go as planned as you will read further about in my deviation from circuit operation section. With spending a fair amount of time reading the SPM32 manual as well as other guides, I decided to move on and get everything else working with my TTL LCD instead and then go back to my UART LCD if I had enough time. Right away, I knew the one downside of using my TTL LCD however, is that I do not have enough analog input ports into my microprocessor due to taking up seven of the fifteen ports. With this issue, I decided to use both of my photo resistors as inputs. However, instead of using my external temperature sensors corresponding to each photo resistor, I used the temperature sensor already on the 384 board as one unison temperature for both photo resistors. I figured having it print the proper information to my TTL LCD would be better than the improper information to my UART LCD, therefore continued with this design. With this in mind, I planned on getting one of my photo resistors working, then print for the specific photo resistor along with displaying the temperature in hexadecimal. Once this was done, I planned to get my halogen lamp as well as my buzzer to activate at the appropriate times. Finally, I planned to get my other photo resistor working along with the temperature again.

2.0 Testing Strategies & Procedures

2.1 Hardware

For my project I did quite a bit of testing before throwing it all together. I first wanted to see if each of my individual parts were working. This is because I have had broken components before without realizing until a lot of time is wasted. To test my parts that had a visual/audio output, I put them in my breadboard by themselves and applied the proper voltage to them. I first tested my photo resistors by putting an LED after them and applying voltage, then putting my finger over the sensor and watching the LED dim. Secondly, I tested out my buzzer by applying the appropriate voltage to it; it worked perfectly. I then tried to get my halogen light to work by using alligator clips and applying voltage, however it did not work. I finally realized it was because of the alligator clips so I plugged it in to my breadboard and it worked. Lastly, I tested out my relay by connecting it to a 9V battery and I was able to hear it activating. The only component I was unable to check using audio/visual senses was my temperature sensor.

Once I knew my most of my components worked by themselves, I put them into my breadboard based on the schematics I had and then tested them again to make sure I had everything wired properly. I tested my relay along with my halogen light to see if it would activate when a load was applied to the transistor; it worked the first time.

I then tested my photo resistors and temperature sensors by measuring the voltage coming out of my Op Amp IC's. I was able to tell they were all working due to the voltages increasing or decreasing depending on whether I was covering the photo resistor or not, and depending on what temperature was being read. To help myself, I tried to use jumper cables in colour groupings on what their purpose was such as white to ground, red was 5V, etc. Also, as mentioned earlier, to have it organized to my liking, I labelled any wire that was outgoing to an external power source, or to my STM32 chip.

2.2 <u>Software</u>

Once I knew everything was hooked up and wired properly, I began by trying to get one component working at a time instead of everything all at once. Like we are always told, "code in chunks, and test it". I started by trying to code my UART LCD to work as mentioned previously. I was able to get it to start printing certain values almost immediately, however it was not clearing and resetting the cursor on the LCD. Therefore, I attempted to read the STM32 reference manual as well as other guides to solve this problem. As mentioned in 1.2 Software Description, I was unable to get it to work within a fair amount of time and therefore decided to get everything else working first. I first created a function for my photo resistors, then had it print out information based on which one it was. I then got my temperature sensor to work. Once that was working, I got my halogen lamp to light up at the appropriate time and then my buzzer, as described in 1.2 Software Description. If I ever ran into any problems, I would simply debug and go through my code line by line to see exactly what was happening.

3.0 Deviations

I unfortunately have had to deviate from my original proposal and functional specifications document. I am using a majority of the same components proposed in my original design, however there is some deviation. First of all, I made a minor change for the alarm cut-off. Instead of having a pushbutton, I decided to use a SPST switch instead. My original plan was to use a normal pushbutton and have my STM chip detect as soon as I push it in order to cut off power to my alarm. I decided this would just be tedious however and decided to use a switch that I had laying around. Another deviation to my design would be using my TTL LCD instead of my UART LCD. Due to being unable to get my UART LCD to work, I decided to just use my TTL instead. The unfortunate part about using my TTL however, is that it limits the number of analog ports to the board drastically. Due to this reason, I was unable to use the external temperature sensors I wired up. Instead I just used the temperature sensor on the board as a general barn temperature. I could have taken my potentiometer off and soldered in another temperature sensor but I felt that would be a waste of time being that it would just be another analog in and the code would be practically identical to the one already on the board other than changing the ports.

4.0 Difficulties

As mentioned previously, this project of course did not come without difficulties. From the very beginning with deciding whether or not I was even buying the appropriate parts or not, until the very end with coding it all up, there were hurdles. It was nice this project was given to us very early on, the only unfortunate thing was I had no clue how to code or how the certain interfaces required worked. Eventually, we learned them all but then came the hurdle of building the circuit. Unfortunately, I have not taken an electrical class relating to the same types of electrical components I was using in my design in quite some time. I managed to get help with a few of my questions, referred to my lab book for certain things, and went along my way building my design. Once built and beginning to program, I of course came to a few errors in the code that slowed me down trying to solve. However, for the most part it was not that bad. One issue includes having the wrong functions for my ADC inputs. Luckily, I was able to get help with anything needed.

The real issue came about when I initialized a port, PC13, as an output. Everything up until that point was working but then certain outputs no longer worked and my STM chip got really hot. I immediately unplugged everything thinking I most likely fried my chip somehow. The chip cooled down fairly fast so I decided to plug it back into my laptop to see if it would even recognize it anymore. My laptop instantly recognized it and all the contents on the device. The unfortunate part is I was unable to load anything onto the device. The chip got fairly hot again as well. Therefore, I unplugged the device and went to look in the reference manual for any advice. After talking with the lab instructor it was because my buzzer was asking for too much current for my device to give. The buzzer needed at most 30mA of current whereas our chip could produce at most 25mA. I managed to get a new head unit and was told to use a transistor to control my buzzer. This was the last hardware issue I faced.

5.0 Components

The final components and parts used in the end to design my project were as follows:

2 – BC547 NPN Transistors

1 – 10Amp PC Relay

1 – LM35 Temperature Sensor

2 – MCP6002 Op Amp IC's

1 – 47k Ohm Resistor

3 – 10k Ohm Resistors

1 – 12VDC Halogen Lamp

2 – Photo Resistors

1 – Sub-miniature Toggle Switch

1 – Audible Buzzer

1 – TTL LCD

A lot – Jumper Wire/Other Wire

1 – 9VDC Transformer

1 – 3-12VDC Transformer

1 – Power Regulator