# Laboratory #2 Report:

*ECE:4880 - Principles of Electrical and Computer Engineering Design*

*Team 9: Alex Arand, Brandon Cano, Ian Kuk, Rogelio Valle*

# Design Documentation

## Overview of Functionality

The goal of this lab was to design an “electric eye” that would work similar to a garage door sensor. This consists of a system with a transmitter that projects a beam of light and a receiver which was not required to be built. There was another main goal in this lab which was dealing with so-called “noisy” sensor data. As a part of the final test the beam of light would be interrupted with light from a lamp.

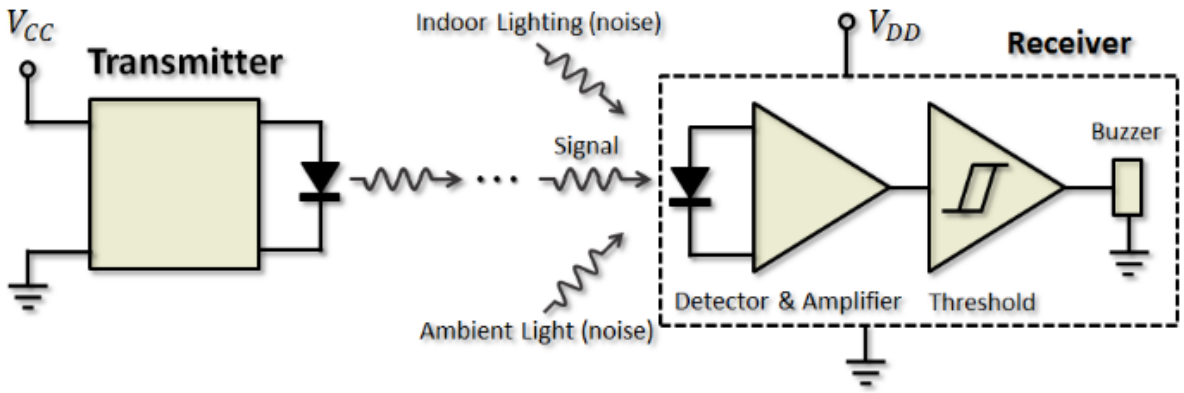



*Fig. 1: High Level diagram of the system*

The transmitter is given and we are assigned to construct a receiver. There was also a mock receiver given to further help us. The transmitter projects a beam of light that the receiver detects using a photodiode. When the beam is interrupted in any way, there should be an SMS message sent detailing when the message was interrupted.

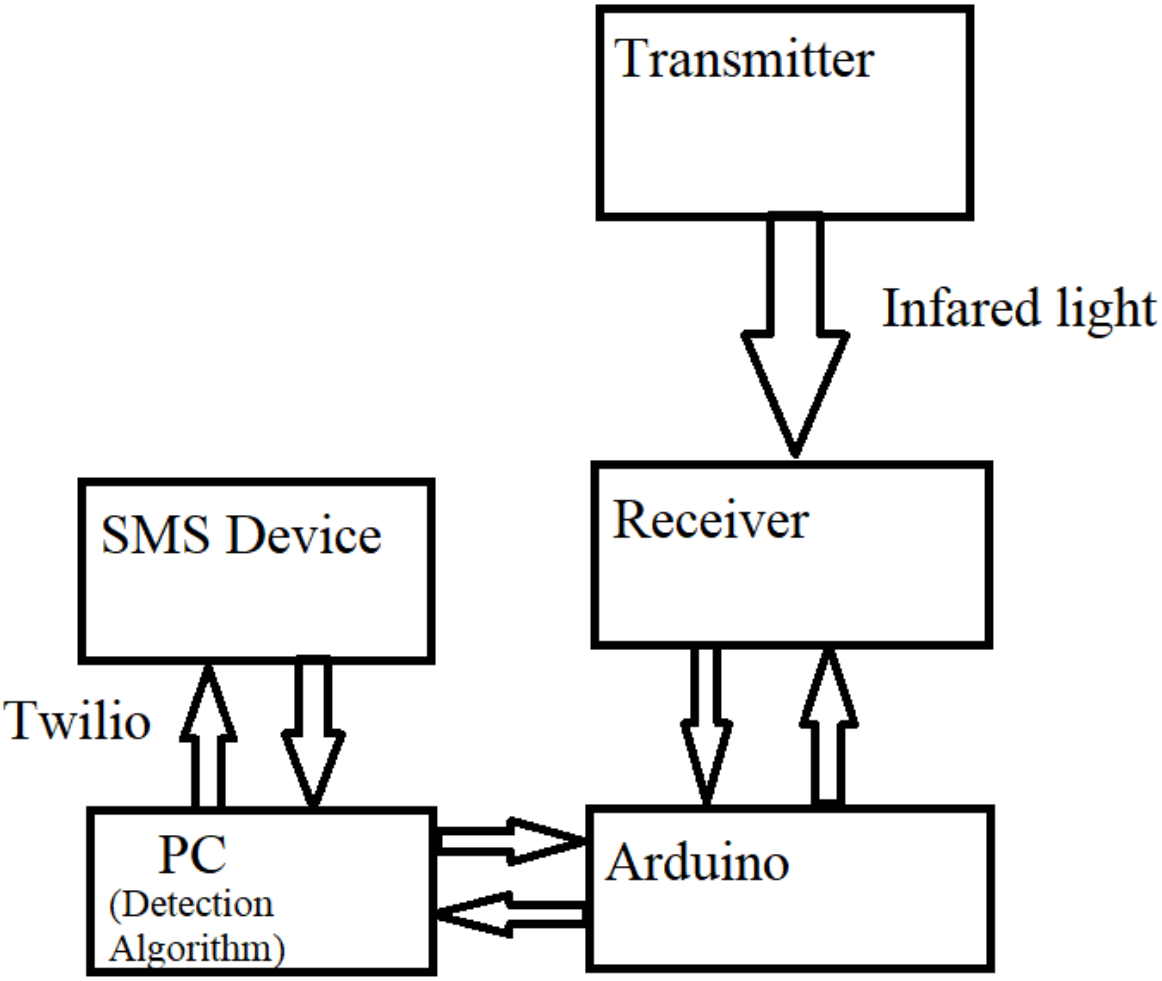



*Fig. 2: High level view of our system*

As shown in the figure our project was divided into a hardware side and a software side. This lab was lopsided towards the hardware side. A majority of our time working was done working on the hardware with the software being done fairly quickly.

## Hardware Documentation:

Our initial hardware design was based on the given receiver used as an example. Our design uses two buffers, a high pass filter, and an amplifier. The circuit starts by having a photodiode connected to the initial op-amp buffer. The buffer's output is sent to a high pass filter. The highpass filter has a stopband of 250Hz and a pass band of 300Hz. The design, and the values of the components of the filter were given by an online resource called Filter Wizard. The filter used two op-amps for its implementation. The output of the filter was then sent to an amplifier with a 100 times gain to make the signal strength even greater. The amplifier was then connected to a final op-amp buffer so there would be less interference when the arduino reads in the signal. The Arduino was connected to the output of the final buffer, and an LED, which would be powered by the arduino power source and the code would determine if it should be on or off. Finally we implement a simple voltage divider so the top rail of the bread board would be equal to -5 volts and the bottom rail would equal +5 volts. This was down because the op-amps perform better when connected to -5 and +5 instead of +5 and ground. However these decisions were made with lots of trial and error and many redesigns while attempting to debug why the circuit would not filter something correctly or why the circuit gets lost from an op-amp.

As shown in figure 3, this was the initial idea for having the amplifier come first and then a band pass filter that would go straight to the microcontroller. This design ended up failing because it had no buffers to help with the signal interference thus the signal would get lost before it got to the output. It also had the amplifier before the filter which made the signal very unreadable. On top of all that, the band pass filter also did not work.

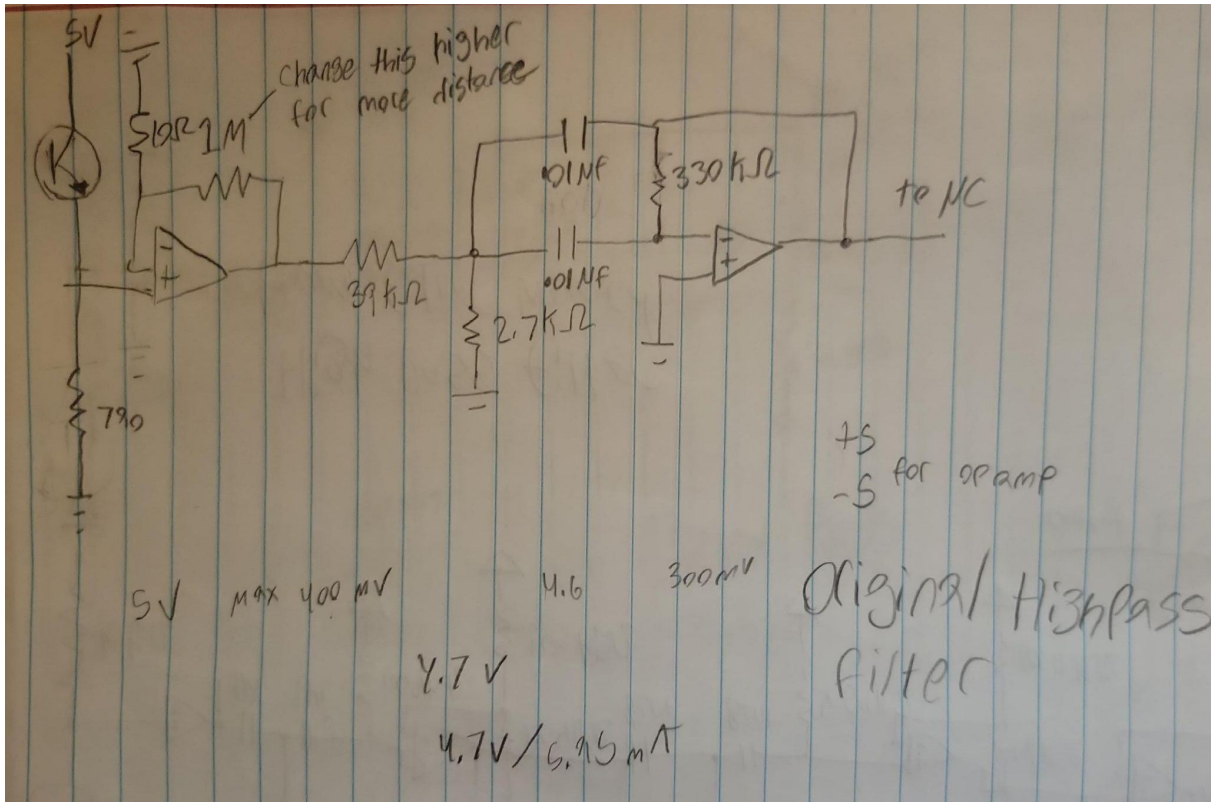



*Fig. 3: First circuit schematic*

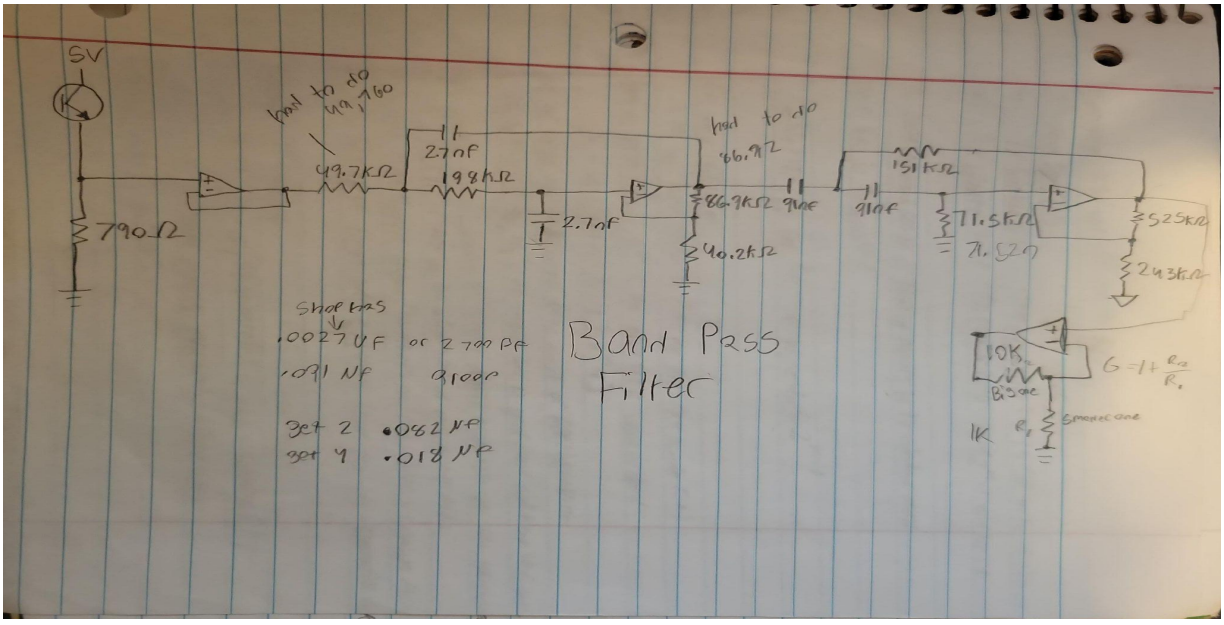



*Fig. 4: Second design of the receiver*

Our second implementation (figure 4) used the band pass filter that was given from the website Filter Wizard. In this drawing there was a proper buffer at the beginning and had the

amplifier at the end. This one worked, but not very well. It could work at two feet, but during the highpass portion of the filter it would kill the amplitude of the signal meaning the output voltage was just far too low for it to be usable since no amount of amplification was able to help us after a certain point. Then between this not working under the lightbulb, and the signal strength being reduced we decided to scrap this design.

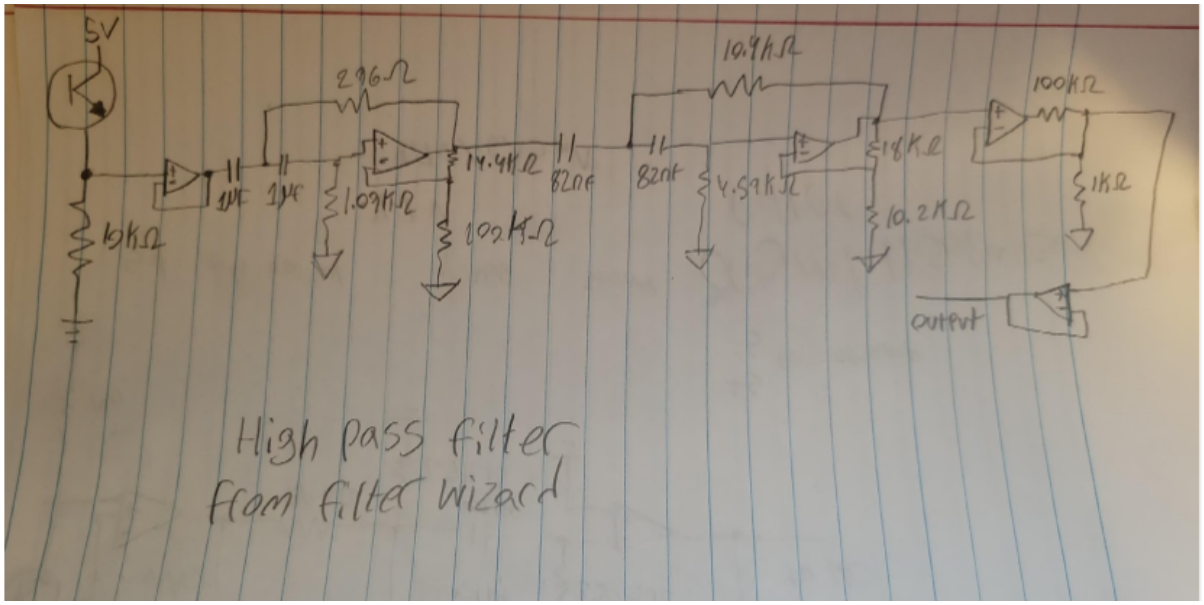



*Fig. 5: Final design of the receiver*

This was the final design of the circuit (figure 5). We decided to scrap the band pass idea and only do a high pass filter. This design consisted of a buffer at the beginning and end of the circuit, and a filter that worked in the light that also did not kill the strength of the signal. This design worked well beyond 3 feet but still struggled with the light until we switch our diode out for one that was able to block the outside light from being a distraction so we did not have to filter it out ourselves, but this new diode also lowered our range a bit, but that was able to be solved with a high amplification as seen in figure 5.

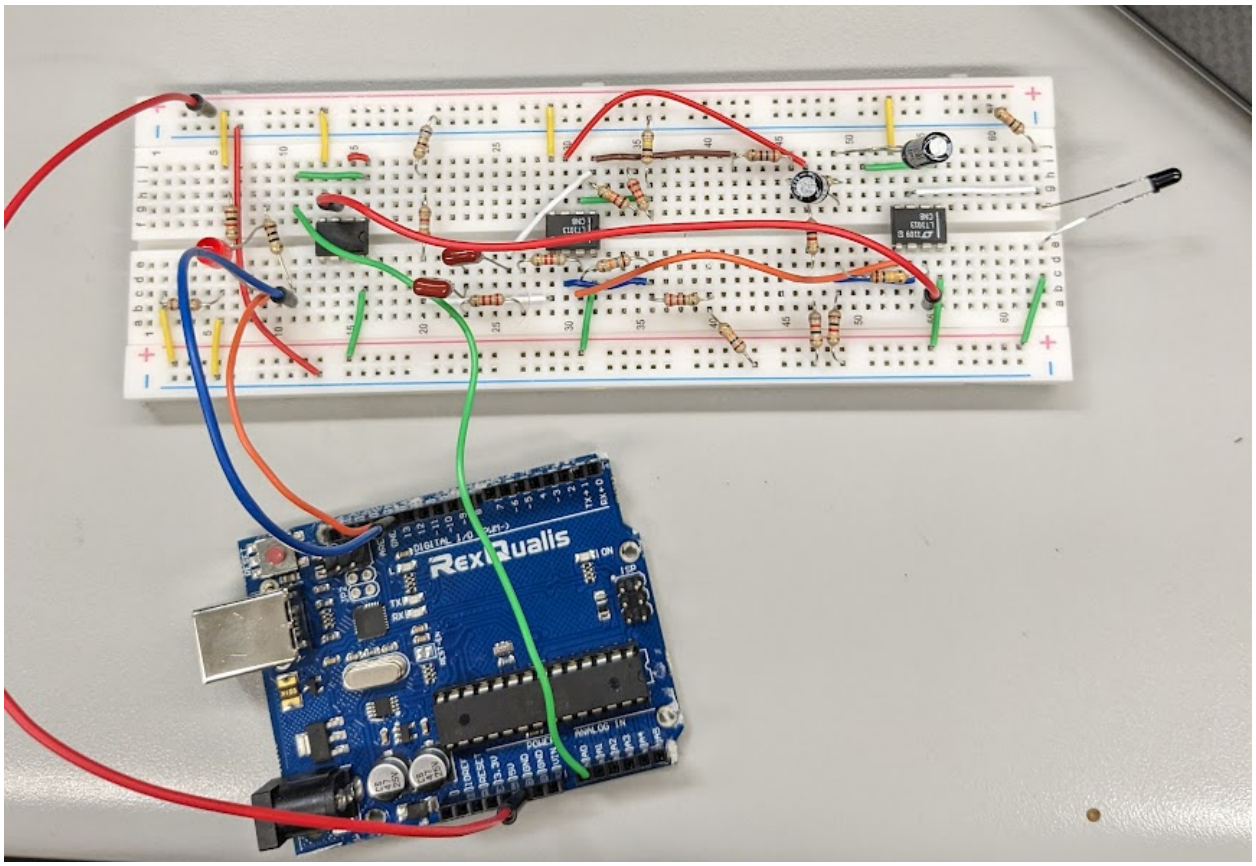


*Fig. 6: Receiver that we built*

| **Part** | **Quantity** |
|---|---|
| Photodiode | 1 |
| Resistors | 20 |
| Capacitors | 4 |
| LED | 1 |
| LT1013 OP Amp | 3 |
| Arduino | 1 |

*Table 1: Parts list for circuit*

## Software Documentation:

The two parts of our software were the detection of the signal and sending SMS messages. For the signal detection, we decided to use Arduino code in the Arduino IDE since we were familiar with it from the previous lab and could easily setup the connections from the Arduino itself. Here is a table of all the libraries and software components we used in this lab

| Name | Description |
|---|---|
| Arduino IDE | - Compiles and uploaded code to the ATmega328P<br>- Easily search for, download, and add libraries to the code |
| Intellij IDE | - Was our IDE for developing and running the Java code<br>- Easy access to add libraries from Maven |
| Twilio | - Java library used for interfacing with the API<br>- Sends text messages from the Java interface |
| jSerialComm | - Java serial communication library<br>- Allows us to read in values from the serial monitor |

*Table 2: Software and libraries used*

For reading in the output from the receiver, at first we tried reading in the value from the analog pin, A0, with the `analogRead()` function, but quickly noticed that we could not get a good easy reading to determine if the signal was lost or not. To solve this issue we did some debugging with the receivers to see what values we were reading in and if we could find a detectable pattern to make the distinction. From our findings if the signal is connected, then the readings ranged from 0-1023, and the speed of this was affected by the distance apart the transmitter and receiver were. However, once the signal is cut then the readings go to the 960 - 1023 range, meaning we now had a pattern we could check for. For this detection we decided to sample a set of readings and see if any drop below 950, if any do, then the signal is detected and if none do then the signal is lost. To do this we had a function that would add a new reading to an array with a the

sample size set to 8, if the array is full then we check all 8 values at once to see if we have a signal or not

---

```
if (count == sample) {
  ...
  for (int i=0; i<sample; i++) {
    if (values[i] < 950) {
      interference = 0;
    }
  }
} else {
  values[count] = value;
  ...
}
```

---

This set of code would also change our interference variable to say that the signal is lost which we then use to turn on or off the LED and to write a string value of either “0” or “1” to the serial monitor to be read in by our running java program.

---

```
if (interference == 1) {
  // signal lost
  digitalWrite(ledPin, HIGH);
  Serial.println("1");
} else {
  // signal detected
  digitalWrite(ledPin, LOW);
  Serial.println("0");
}
```

---

In order to get a text message we decided to utilize the Twilio library in Java as we could not find an option for Arduino code. In order for us to read in values from the serial monitor we also had to utilize the `jSerialComm` library.  This library allowed us to set up the port and baud rate

information and for us to create our own serial event function. This made it so that we could trim the data into the strings of “0” and “1” that the arduino is sending out constantly. From this we have some simple logic that detects what value is being read in and if a text should be sent or not to prevent the program from spamming the user.

---

```
if (receivedData.equals("1") && !textSent) {
  textSent = true;
  ...
  SendText.sendATextToPhone(textMessage);
} else if (receivedData.equals("0")) {
  textSent = false;
}
```

---

Essentially, once the code detects the signal is lost then it sends a text message but also triggers another flag that prevents it from sending another text message until the signal is detected again.

# Design Process and Experimentation

To begin this lab we all scraped together the lab kits we had left over from embedded systems. Each of these lab kits had all the tools we needed to begin working on the lab. For our microcontroller we used an Arduino ATmega328P since it was what we had from embedded systems. We could have done this project using C or Assembly language as in embedded, but we chose to use the Arduino language since it had more accessible libraries.

Since we have previously used Twilio to send SMS messages, we decided to use it again, but this also meant we had to find a way to transmit data from the Arduino to our Java code. Since we could do either a wired or wireless connection for getting the data, we ended up doing a

wired serial connection since the jSerialPort library was readily available with minimal setup to read in the data being sent to the serial monitor.

As for the values for all the resistors and capacitors and other items on the circuit, Filter Wizard was able to generate the values we needed for all of our designs which helped make it easy to know what we needed so we could just focus on building it instead. There were times that it gave some odd values which is made apparent in figure 4, so to fix that issue we could modify some of the values for ones that we could obtain which fixed our issue of an over crowded circuit with all the resistors and capacitors in series/parallel.

# Test Report

| Test Number | Lab Requirement | Pass Criteria | Test Result | Observed Results | Pass/ Fail | Date |
|---|---|---|---|---|---|---|
| **1** | **The receiver detects a signal from the transmitter** | **The signal from the transmitter is detected by the transmitter** | **The signal successfully showed on the oscilloscope** | **There is a signal in the correct shape on the oscilloscope.** | **Pass** | **10/12/2023** |
| **2** | **The receiver can read a signal from 3 feet** | **The signal is still readable from the transmitter at a distance of 3 feet** | **Our receiver successfully reaches a length of 3 feet** | **Length fades away from 3.5-4 ft in length** | **Pass** | **10/17/2023** |
| **3** | **Under low-light conditions, the receiver should not have a measurably worse rate of false positives or false negatives** | **There are not false positives in low light conditions** | **Our receiver does not have any false positives in low light conditions** | **There are no false positives when the light is right next to the receiver** | **Pass** | **10/17/2023** |
| **4** | **The receiver shall successfully detect an object within 10 seconds of the IR beam being interrupted** | **There is a text send when there is an interruption** | **SMS successfully sends a text alert** | **SMS sent to the phone number in the repository, in this case Brandon's** | **Pass** | **10/20/2023** |

| | | | | | | |
|---|---|---|---|---|---|---|
| **5** | **The receiver should work reliably in the presence of a 100 W incandescent light bulb** | **When a bright light is in the presence of the receiver there is no major performance degradation** | **Our receiver works reliably when in the presence of the 100W light bulb** | **There is a minor but negligible performance degrees when the light is right next to the photodiode** | **Pass** | **10/23/3023** |
| **6** | **The Receiver should not have an abnormally large size or weight or other physical characteristics that may prevent it from being a reasonable replacement** | **The receiver is a reasonably small size and does not have any overbearing components** | **Our receiver is fit neatly on one breadboard and does not have any cables that block connection to the computer** | **After experimentation we managed to fit it all on one. We also tried the long breadboard with the square one.** | **Pass** | **10/25/2023** |
| **7** | **Protect against the possibility of cables attached to the receiver causing physical movements that affect the alignment of the photodiode** | **The receiver should have a clean enough design to where a mess of cables will not interrupt functionality** | **Our receiver has clean circuitry and connection to the laptop is simple** | **We initially had odd capacitor values but we used Filter Wizard to get cleaner values** | **Pass** | **10/25/2023** |
| **8** | **The SMS message should include a time-stamp in the format "Critical Safety Event at HH:MM on Month/Day/2023"** | **The formatting for the SMS message matches that of the requirement** | **Our SMS message matches the required formatting** | **N/A** | **Pass** | **10/20/2023** |
| **9** | **The computer connected should be able to read values coming in from the receiver** | **The receiver should be able to send values for the computer to read** | **Our computer successfully reads the values sent from the receiver** | **Found using the correct pin, analog pins and digital pins give different values** | **Pass** | **10/17/23** |
| **10** | **The computer will use software to tell when there is an interruption** | **Our software should have some method to tell whether or not there is an interruption detected** | **Our software detection method detects when there is an interruption** | **When there is a loss the signal being red there will be a message send to the SMS device** | **Pass** | **10/20/2023** |

# Project Retrospective

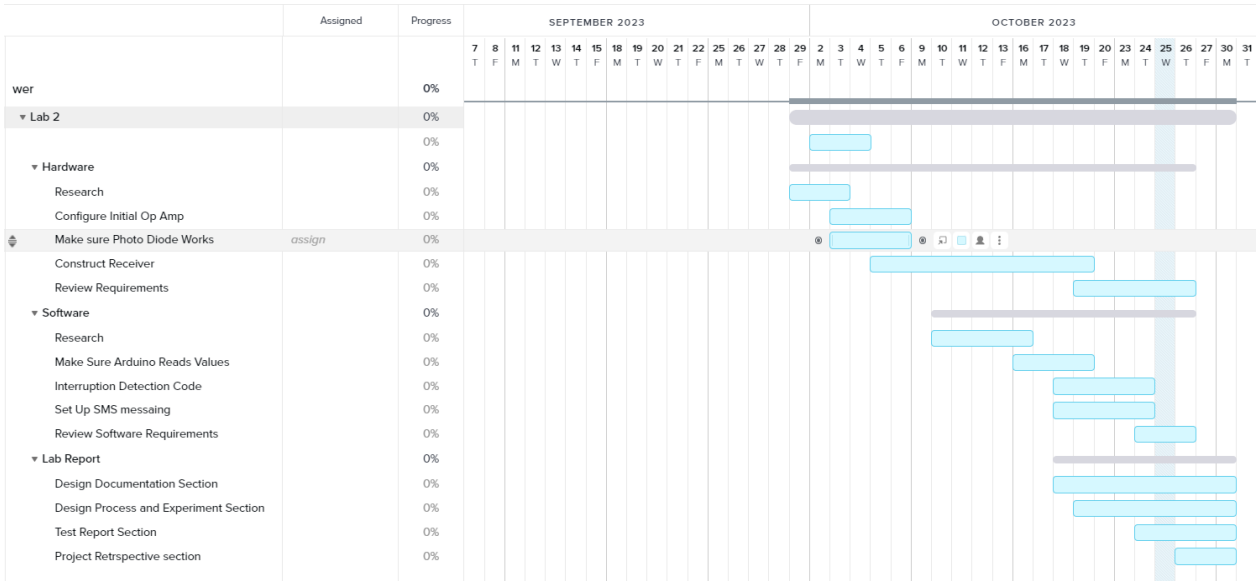
**Alex (CSE):** Got the initial setup of the code going and researched the necessary libraries for the Java code.

**Brandon (CSE):** Responsible for connecting all the three sections together, Java code, Arduino code, and circuit. Also helped with the final build of the circuit.

**Ian (CSE):** Researched necessary components. Responsible for almost all of the implementation of hardware requirements. Debugged the circuit in great lengths and rebuilt different designs in order for the receiver to work.

**Rogelio (CSE):** Responsible documentation and time management. Input and ideas for both hardware and software sections of the lab.

## Gantt Chart:



## Development Ideology and work distribution

All of the group members contributed to the success of this project. One of the biggest issues we had earlier on is a lack of combined knowledge on circuits since all of us are CSE majors. Ian was the member the most hardware inclined so he took an early and sustained command over that section. Alex and Brandon focused more on the software portions of this lab. Rogelio took the role of doing documentation as well as providing input on any needed changes. With all of us together we managed to have a working product ready to check off.

## Project Management Process

When considering the development ideologies to be used for the lab we decided to use the Agile method. This method benefited us greatly because once we had the base design done we could experiment with different parts until something worked. With this time management process we were ahead of the curve when it came to having our hardware and software done.

## Timeline

When we began the project we decided to get an immediate start on the hardware. This is because all four of us are CSE majors and we predicted having some troubles with hardware. We based our initial design on the receiver that was given. Later on we found out we could use Filter Wizard to enhance our design to read the necessary 3ft of length.

When it came to the software side of things we put that off until we had a working hardware product. All of us are more software oriented so we knew we could recycle our SMS

code from Lab 1 and solve any software issues much faster. We were also able to develop a method to be able to tell when the signal was interrupted to send a SMS message.

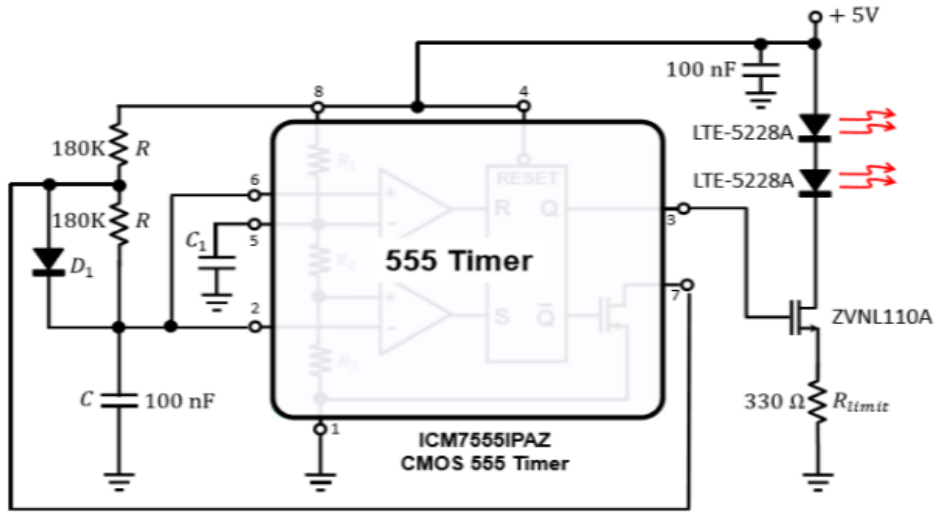
## Critical Assessment

Now that we have completed two out of three labs we feel even more confident going into the final lab. Our group is made entirely of CSE majors so we struggled initially with this hardware focused lab. We overcame all challenges that arose over the course of this lab and we were able to properly communicate with each other. Initially we had some troubles getting the length requirements. This is because FilterWizard gives us unorthodox capacitor values. To fix this issue we calculated cleaner capacitor numbers for the system. With the newer values this also fixed our length issues. Beyond this there were not many major challenges present throughout the lab. Our team was most confident when it came to the software side of things. All of us have an inclination towards software and pitched ideas on how to detect a loss in signal. The only drawback to this is how little software was actually used in this hardware based lab. We consider this lab to be a success since we were never on a time crunch to get things done.

# Appendix & References

Filter Wizard:
https://tools.analog.com/en/filterwizard/

Transmitter Schematic:




$C_1$ = 10 nF
$D_1$ = Small Signal Schottky

ATmega328P Datasheet: