|  |  |
| --- | --- |
|  | **2016-2017** |
|  | Project members: Thomas Suarez, Frank Yee, Ricky Zhang |

|  |
| --- |
| **[**Mosquito/Bug Zapper Report**]** |
| Senior Design Project Paper; Supervised by Prof. Westerfeld and Prof. Tang |

*This senior design project is supervised by Professor David Westerfeld. The project team consist of three members, two electrical engineers and one computer engineer. This project will utilize skills learned from our classes throughout our engineering programs, circuit design, digital signal processing, programming, etc. The project goal was to create a more effective product than current models in the market. The product was to be able to attract, detect, and eliminate mosquitoes. Current methods of eliminating mosquitoes utilize a bait (typically UV light). Once the target is within the trap, a high voltage electric shock will eliminate anything that makes contact. However, these methods are ineffective towards mosquito population control because the UV bait does not only attract mosquitoes, but other insects as well. Consequently, other insects are also eliminated instead. Our end-product will consist of more advanced methods that will increase chances eliminating our target.*

*We would like to thank Professor Westerfeld for advising us in our circuit analysis, and Professor Monica Bugallo in help with digital signal processing.*

# Table of Contents

* Section 1. Goals and Impacts
* Section 2. Background

1. Survey
2. Project Planning

* Section 3. System Design

1. Implementation problems
2. Final Implementation
3. Testing

* Section 4. Results and Discussions

1. Result Analysis
2. Mutli-Disciplinary Team Experience
3. Professional / Ethical Considerations
4. Impact of Project on Society or Contemporary Issues

* Section 5. Summary and Conclusion
* References
* Appendix

# Section 1. Goals and Impacts

The goal of this project is to demonstrate and design an effective way to control mosquito population through the means of a better mosquito trap. Current consumer grade appliances are not sophisticated enough target mosquitos only. Our specialized bug zapper will record and process the data recovered by a microphone. With recorded data it will then be processed and used to identify mosquitos by sex and species through frequencies.

A better designed mosquito trap will not only improve the overall efficiency of bug zappers, but is also capable of impacting personal health on a global scale. By eliminating female mosquitos in dense human populations, the disease and viruses spread through mosquito blood transmission can be better controlled or even possibly eliminated. Through this means of prevention, it can even save research and development time and money from creating the necessary vaccine/drugs used to combat mosquito borne diseases.

As an alternative use, the device can be configured to not kill but retrieve useful data. Such data includes mosquito population by area, female to male ratios, or even track the frequency of mosquito activity. With the Bluetooth incorporated it can communicate said data with person(s) nearby. The World Health Organization uses information like this for their Malaria Vector Control Strategy [1].

# Section 2. Background

## 2.1 Survey

Mosquito population is an extremely difficult thing to control because of how easily they can reproduce and how they can thrive in many environments [2]. Due to these increasing large populations of both humans and mosquitoes, deadly diseases are easily transmitted from one to the other in very short spans of time.

It can be debated whether keeping eradicating all mosquitoes is beneficial or may have detrimental consequences on the world. Some experts state that they are needed as food for other insects and animals, while others argue that there are almost no negative aspects to eradicating all mosquitoes [3]. Therefore, controlling the mosquito population must be done correctly and accurately.

To control them, many people use bug zappers to eliminate mosquitoes. However, these methods are ineffective because mosquitoes are not attracted to UV light, and other insects are harmed instead. Therefore, many products out in the market today do not eliminate the target correctly.

The reason for mosquito population control stems from an article and infographic by Bill Gates [4]. Malaria is one of the deadliest diseases that plague much of the world today, and is a life-threatening blood disease that are contracted by mosquitoes. Other mosquito borne diseases includes Zika and West Nile. With over 9000 cases and 264 human fatalities, due to West Nile virus alone were reported in the U.S. [5] Based on these numbers, it would be of interest to design a mosquito trap with these facts in consideration.

As mentioned earlier, our sophisticated trap utilizes sound and furthermore processes said sound into useful data. The detection of mosquitoes by flight sounds has a long history dating back to 1949 with Offenhauser and Kahn and has continued since. In 2007, a group of engineers created a field deployable acoustic insect flight detector [6]. The device was constructed only to recording insects with no further applications. The recorded data was then analyzed afterwards. The usage of noise cancelling, filtering, and recording aspects of that design is applicable to our design.

One more additional modern design that's under development, is a laser based bug zapper that uses a non-lethal lasers to detect then analyze an object's frequency. If the frequency meets the parameters then a lethal laser is activated and precisely uses the nonlethal laser to aim at the object. [7]

## 2.2 Project Planning

Since our project team consists of two electrical engineers and one computer engineer, work can be split amongst our members. Each member will demonstrate and brief the teammates how the respective design was implemented. The electrical engineers will focus on circuit design, while keeping design constraints in mind. The computer engineer is tasked with programming the computer and digital signal processing of input signals.

Knowledge for circuit design comes from various ESE courses, anything else lacking will come from research as well. Research includes microphone selection, the analog to voltage converter that follows the microphone, and the specifics of the components meant to zap the mosquitoes. For programming the Raspberry Pi, Python will be the main coding language, and the specifics of the language will be researched on to utilize its functions. Familiarity of the Raspberry Pi will be a must, therefore side projects with the Pi to learn basic functionality is to be expected for all teammates. The DSP will be done within the Raspberry Pi, so programming must be specifically tailored, and further research on this topic is required.

Professors have also been a very reliable resource in the beginning of our design. Professor Westerfeld, is a great source of advice for the general outline of the design concerning core components. Professor Monica Bugallo who teaches Communication Theory explains how signals can be processed and analyzed. First, we need to learn to recognize the signal of a mosquito. Then, we must also consider the different species of mosquitos and the different signals that female and male mosquitoes create.

# Section 3. System Design

## 3.1 Implementation Problems

Some initial problems occurred during the implementation of the final design. There was a problem in regards in choosing which hardware filter use. Initially, we planned to use a band-pass filter, but upon taking measurements and analyzing the data, the filter characteristics were rather questionable. This made us take a detour in investigating the type of parts used, and reconsider how much of the filtering can be done via software. In-regards to the code, during implementation phase, the CPU of the system would spike to 100%. This resulted in taking additional time to rebuild, and debug which package(s) used had caused such problem. Another problem was procuring a step up center tapped transformer which is not readily available for purchase from traditional electronic supply stores. A transformer from a bug zapper had to be salvaged. The op-amp from the original design could not be used since a negative voltage was needed so an inverter was used instead which improved the performance.

## 3.2 Final Design

The final design uses an omnidirectional electret condenser microphone from UCI Inc. Combined with a Maxim MAX4466 preamplifier to increase the voltage gain and providing high noise tolerance allowing for further processing as shown in figure 1. The signal is first processed with a first order highpass filter and to the ADS1015 analog to digital to converter (ADC).

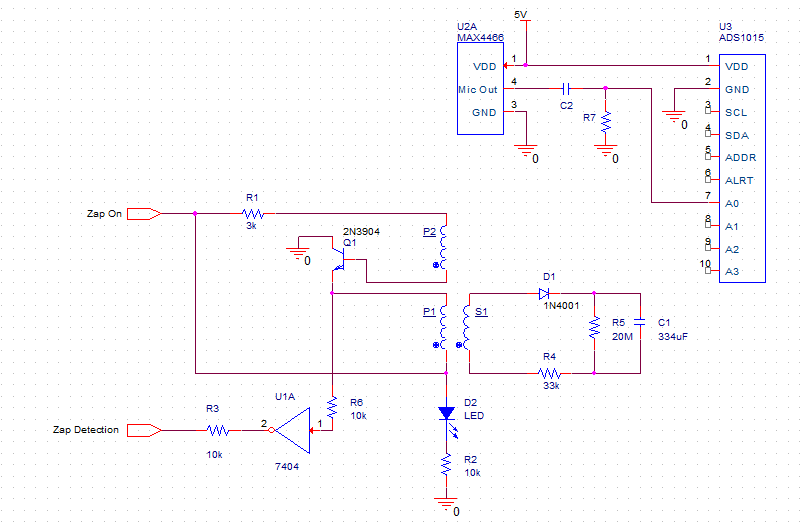


Figure 1: Final Schematic Design

The part of the device that will zap the mosquito will receive be power by one the I/O port when a mosquito is detected. As shown in figure 1 the I/O port will power the center tapped transformer with a step up ratio of 10:1. Unlike the original design which used a relay switch a transistor (Q1) provides the AC current needed for the transformer to provide a step up voltage. This is done by using flux from one of the primary transformer coils P1 to send a square wave to the base of the transistor. The secondary coil using a half rectifier provides a positive voltage wave to a 334uF capacitor. The capacitor stores the high voltage for two metal meshes that will zap the mosquito. One is attached to the positive end and the other to the negative end of the capacitor. The negative mesh is the outer layer with the positive as the inner layer spaced 2mm apart to prevent arcing. The The transistor also provides the zap detection from the drain by using a inverter with a current limiting resistor to a I/O port.

The pcb was designed using the Eagle Cad tool and created by OSH Park. It is 2 layered 50mm x 50mm pcb with a dielectric constant 4.6 at 1MHz. Since the board uses low frequency signals a lower quality pcb can be used while lowering the overall cost. Wide diameter traces were used at the capacitor as shown in figure 2 in case of high current. The microphone input and output are kept away from the transformer and transistor since there signals can potentially jump across traces if they are close enough.

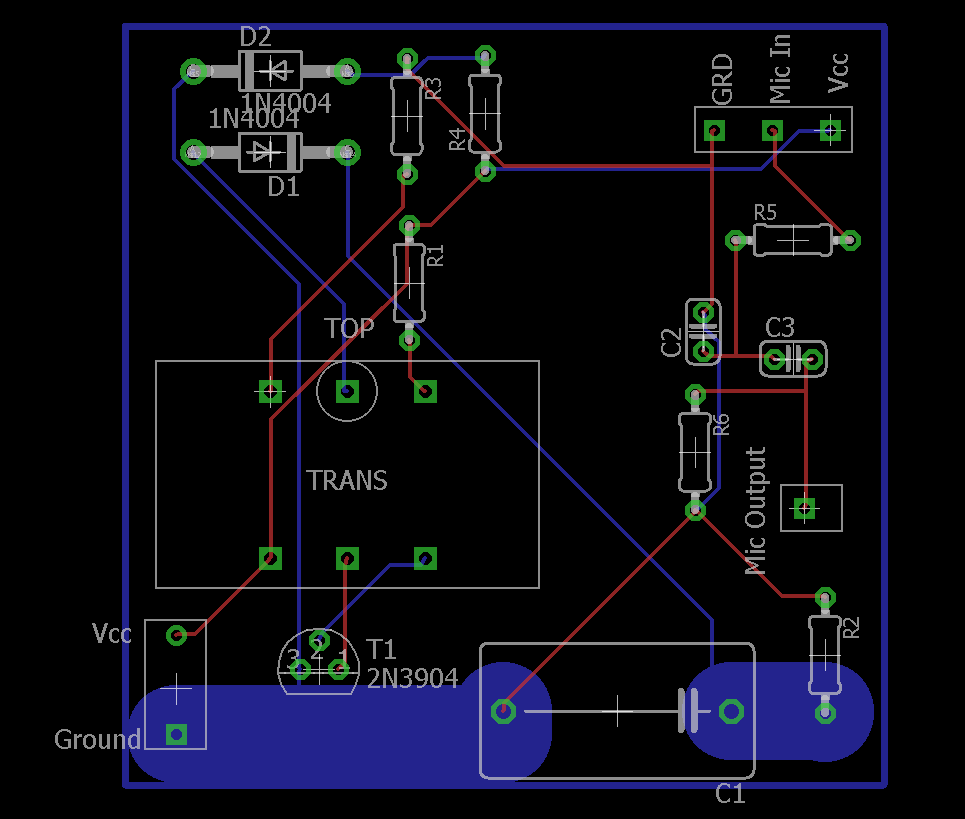


Figure 2: Final Board Layout

For the final implementation of the code for data and signal processing as well as any additional capabilities of our design. The main function of our code will be to sample the data from our microphone (MAX4466) at a user input rate along with a time interval, and with that information convert it to a set of digital data that we can perform a Fast Fourier Transform on. Doing so will allow us to search through our arrays for certain frequencies that we can identify as a mosquito and turn on the zapper. The zapper will be one of the outputs of the RPi, which means we need to configure a GPIO pin for it and also an input pin for feedback when we have eliminated a mosquito.

The first section of our code imports libraries used for the Adafruit ADS1015 ADC as well as time, which is necessary for our FFT, and numpy for our arrays. The next part of this section deals will pin definitions for I/O and setting them up correctly. Lastly, because the GAIN can be adjusted on the ADC, we set it to 1 and added user input variables for ease of access.

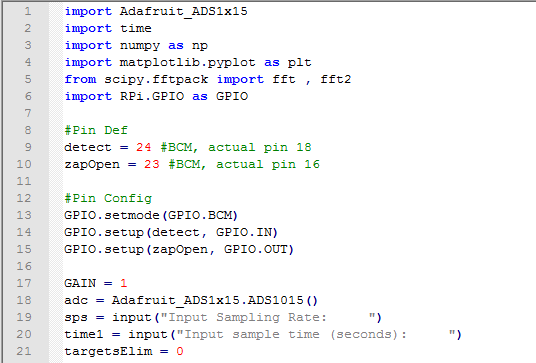


Figure 3 : Code Header

The next part of our code deals with logging in data into our numpy arrays to perform our Fast Fourier Transform. To do so, we will create a 2xN array, the first row being our digital data from the ADC, and our second row being our created time. The loop will create an array with a length we specified based on the length of our time interval as well as our sampling frequency. In the loop, we put in the result from the ADC with the corresponding time and then we increment our time variable.

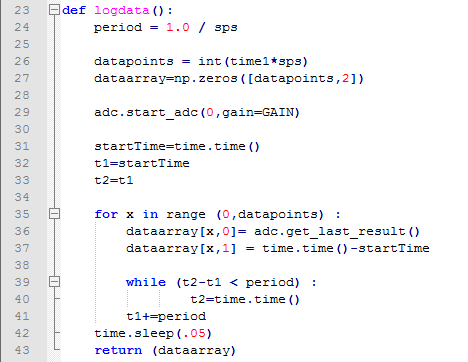


Figure 4: Function Logdata

This method will be used to output 3.3V to our designed circuit which will eliminate the mosquito. We set a time.sleep to .1 so there is a small delay before it actually zaps, as well as a delay while it zaps and discharges.

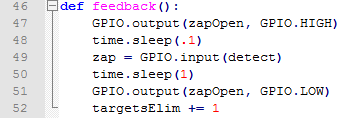


Figure 5: Function Feedback

For the last part of the code which is our main method. We put everything in a while loop which will continuously run. A conditional loop can be used instead to make sure that it doesn’t run forever. We then use logdata() to get retrieve our array and put it in dataSamples variable. We take that variable and perform a Fast Fourier Transform on it to get into the frequency domain. The consequent array will have phase as well as the signal in the frequency domain, so to find the magnitude of our signal, we perform abs() on it. We will also perform a fft.fftfreq on our signal in order to get our DFT sample frequencies. Once we have our arrays separated, we inspect whether the obtained data is within our frequency ranges. To do so we perform a simple maximum search algorithm to figure out what the maximum is in our array as well as the index. If the maximum is in our range, we will allow the zapper to be turned on. The rest of our program deals will plotting our signal-time and magnitude-frequency graphs for the user to easily see the data. Lastly we use time.sleep() to allow for some downtime in our loop as well as cleaning up our GPIO pins.

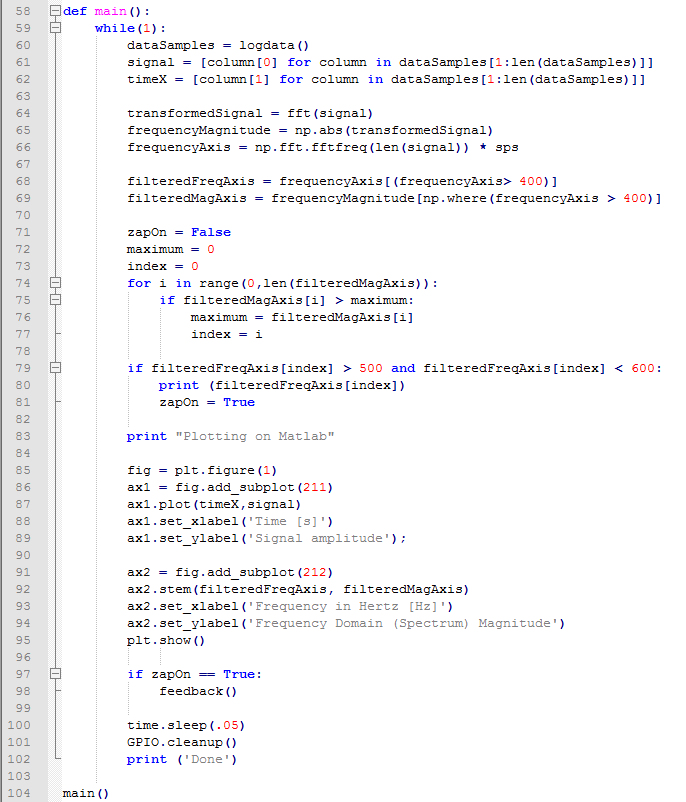


Figure 6: Main code, refer to appendix for complete code

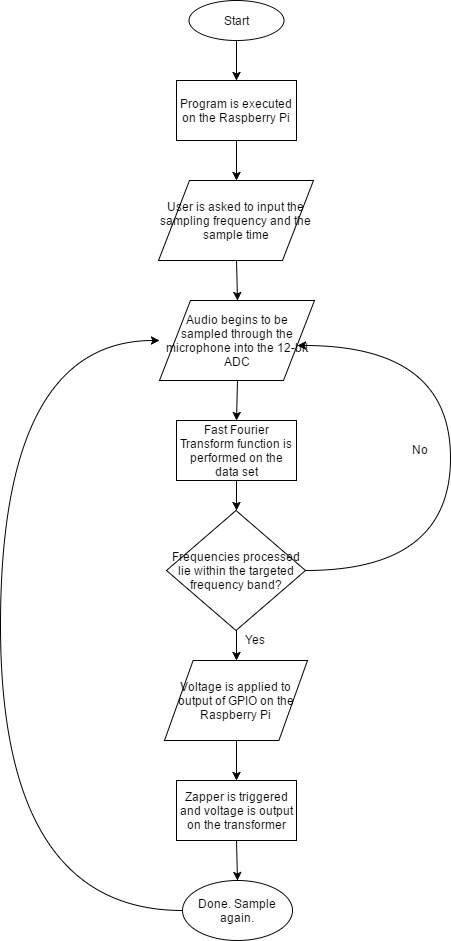


Figure 7 : Flowchart of how the code works

## 3.3 Testing

The testing was conducted in a series of phases testing each separate module before compiling to crate the whole circuitry. The 3 modules being: microphone, digital signal processing, and transformer(zap). Starting with the microphone module, the electret condenser microphone with Maxim MAX4466 preamplifier, 3.3V was used instead of 5V for VCC. 3.3V was used instead because it is considered the “quieter” supply (the General-purpose input/output (GPIO) are regulated on board), perfect for a sensitive sensor like the microphone. A multimeter was wired to the audio out to observe change in voltage based on sounds it detected. Hardware filters like first order passive/ second order active were considered after getting the base of DSP completed.

For the digital signal processing module, it the code was written in python 3 using the numpy[8] libraries (matlab for python). In part of considering what the package offered, and which functions to use to achieve accurate readings, signals were first hard-coded and manipulated using numpy functions. Collected and manipulated data was then visualized through a package called matplotlib[9].

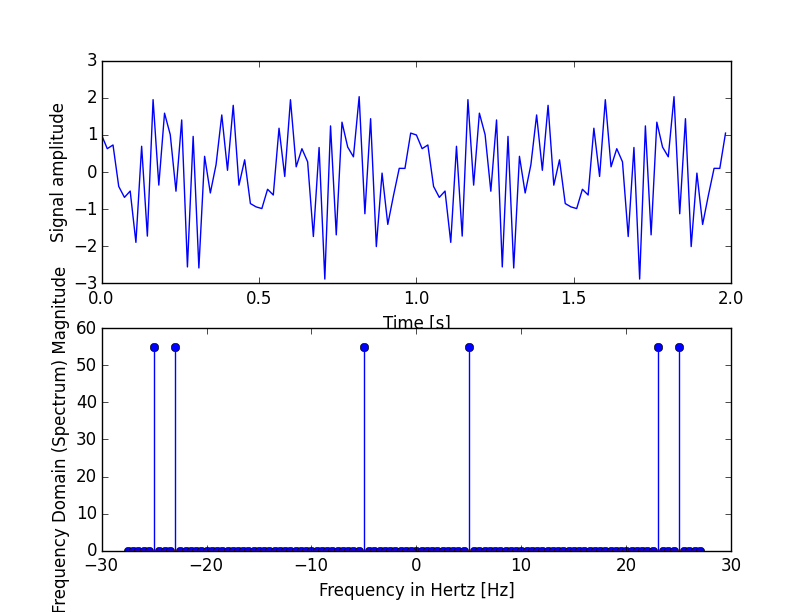


Figure 8: The top waveform is a hardcoded complex signal used to observe function behavior. Bottom graph indicates which frequencies were detected.

After comprehending which functions of the package are usable, we moved onto real signals. Next by storing the readings from the mic to ADC in an array, we then use the same numpy functions to compute the signal, and once again visualize the data collected via matplotlib. It is important to note along the way, each array was outputted and printout out to separate text files for detailed analysis.

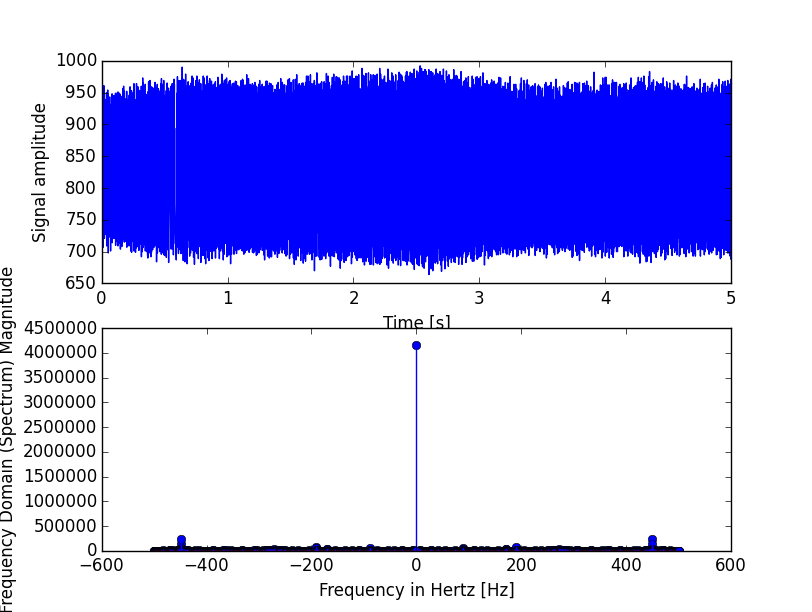


Figure 9: The top waveform is what’s recorded by the mic+ADC, a 450hz signal created by a tone generator. Bottom graph indicates which frequencies were detected(450hz is in-fact present).

At this point, the code functions as planned, however visually speaking it is difficult to make out what frequency is being detected, thus hardware filtering was thoroughly investigated. The hardware filtering is applied to the output signal the mic is sending to the ADC.

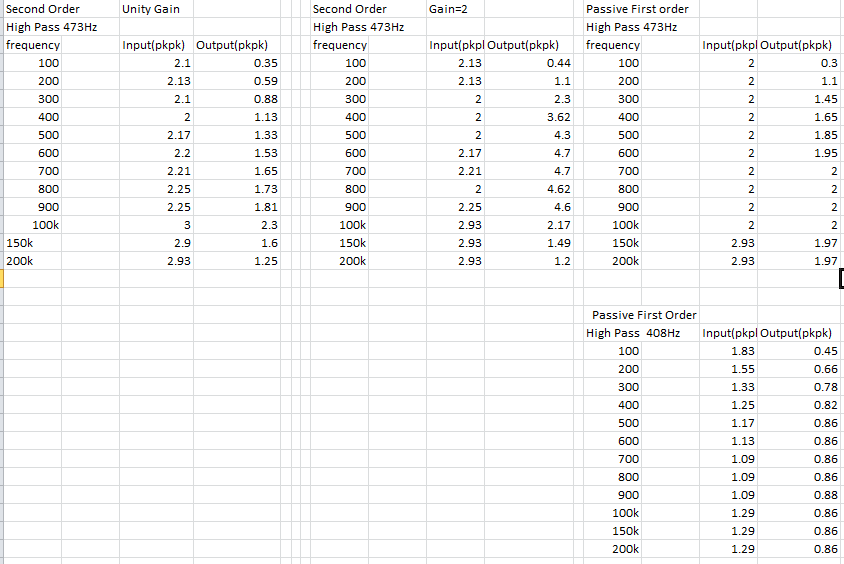


Figure 10: Snapshot of data taken on different hardware-filter implementations. Refer to appendix for complete table.

Looking that the data collected, we decided to go with the Passive First Order design, since that dataset made most sense to us. Later the electrolytic capacitors were swapped for ceramic capacitors and resistor loads were adjusted for a higher number. As a result the new filter behaved more ideally.

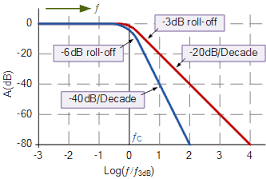


Figure 11: Image from electronics-tutorials.ws. Red illustrates 1st order, Blue illustrates 2nd order. Order correlates to the roll-off (steepness of decade), a steeper slope indicates how much more the voltage is attenuated once outside of the pass-range.

Once the hardware filter was implemented, the frequency detected was a lot easier to visualize and spot. For picture below, we used a first order highpass filter to significantly reduce the zero-beat frequency, and reduce noise below ~400Hz. Compare to figure .

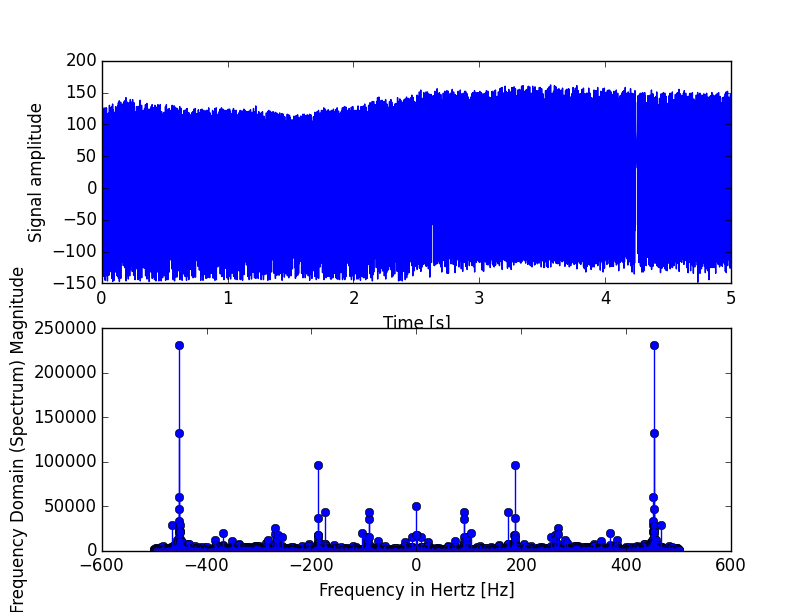


Figure 12: The top waveform is what’s recorded by the mic+ADC, a 450hz signal created by a tone generator. Bottom graph indicates which frequencies were detected(450hz is present, with the zero-beat frequency reduced significantly).

With majority of the code functionality completed, minor edits and tweaks were made to work with the rest of modules. Such as software filtering, where the detected frequency is outputted to the console, saving the user the trouble of having to refer the the graph each run. Giving the code the specific band to work with, it recognized when to turn on the the zap module.

The 3rd module tested was the zapping part. Using a scrapped 3-winding buck boost transformer, and Keysight’s DC power supply, voltage was carefully monitored to observe transformer behavior. Refer to appendix for data and observations made. In the completed circuitry of the 3rd module, a green LED is used to indicate when wires were live. Upon shorting floating wires, it would discharge the capacitor and create a spark to verify module functionality. For the raspberry pi to read such discharge it was critical to use a buffer with high independence to prevent damaging the brain via the delicate gpio pins.

# Section 4. Discussions

## 4.1 Results

Majority of the results were determined on module by module basis. Under the constraints of budget, the electret microphone we picked for the design is based on research published on insect tracking. As explained in the testing section 3.3, various filter implementations were considered. Furthermore, because the transformer model was not readily available off the shelf, we ended up scrapping one thus having no datasheet to work with. However, overall we are satisfied with the design’s ability to process and execute.

Because the the design is comprised of 3 distinct modules, there is definitely a margin of error regarding each part. In regards to the microphone module, we settled for a rather cheap electret mic, a more ideal microphone (such as Sanken contact mics, designed specifically for insects) could’ve been used for accurate reading, but was rather pricy. In respect to the filtering utilized, the pass-band was strictly determined by the flight tone of the female mosquito. Having no actual mosquitoes to work with, only the theoreticals supported by calculations were considered. A question like “how much does the roll-off attenuation factor matter for real signals recorded under low volumes and in an environment abundant of ambient noise” can only be determined once the design is completed and outdoor testing is conducted. As mentioned before, because the transformer was scrapped, there was no datasheet to refer to, all the data and expected behaviors are based on in-lab measurements. As students we could’ve overlooked something critical related to the transformer, that a datasheet would’ve warned us about.

## 4.2 Multi-Disciplinary Issue

As a team of 3 members who have experience in both Electrical Engineering and Computer Engineering, each member will have an important role in each part of our circuit design. Although each project member will be active in the development of each part in the circuit, the electrical engineers will design the audio detection, amplification, zapper, while the computer engineer will be focusing on programming the Raspberry Pi for the applications needed to fulfill our goals. Two members of our team have taken a class for signals processing using techniques such as Fourier Transforms. All members have also taken PCB design to make development of the circuit more compact and simple.

## 4.3 Professional/Ethical Issues

Our product is designed to affect the insect population, which will have a direct impact on the environment. If too large of the mosquito population is eliminated by our product, the food chain will be disrupted by a lack of prey for other animals. However, it has been debated by scientists whether it is detrimental at all to eradicate all mosquitoes because of the overwhelming negative consequences of their survival.

For professional considerations, our group wishes for our product to be optimized in both quality and cost. For now, our group has decided that a working effective product is the first step, and creating another circuit which can be cheaper, with less powerful components can be a later extension of the overall project goal.

## 4.4 Impact of Project on Society or Contemporary Issues

The goal of this project is to create a mosquito detecting and eliminating unit that can potentially save lives. Although, expertise in the engineering field may be needed to build and create the design, the final product can definitely be applied in other fields of biology using audio sensing. Audio detection can be used in many fields spanning from the music industry (audio engineering) to biology, to security and surveillance systems.

With the recent Zika virus outbreaks, underdeveloped places in Central and South America needs for mosquito control has increased along with fears of it spreading further. The World Health organization does try to track mosquito populations and a device like this that has the capabilities to detect and record amount of mosquitoes may be useful in the tracking of mosquito populations.

# Section 5. Summary and Conclusions

The goal in mind for this project is to create a working mosquito detector and zapper. We must design the product within the constraints set by Professor Westerfeld, and from our own considerations. For a quick overview of the circuit, audio is fed from the microphone through an amplifier into our Raspberry Pi. The Raspberry Pi will take the signal in the time domain and get frequencies for each sample and on a certain condition where the frequencies that correlate to a mosquito is met, the zapper will be set off.

The end product must work and be effective in eliminating the target with a low error percentages. Further improvements in our circuit can be to use a less expensive in terms of power consumption, finding a more effective mosquito bait, and a better design of the mesh trap. An additional feature to our current design would be to add Wi-Fi/Bluetooth capabilities that can communicate from the Raspberry Pi to a mobile device. Overall, our current design will meet our goals, but can be improved.

Our research has led us into various fields in engineering that we would normally not have been exposed to which will help in shaping us as engineers in the working field.

# References

1. World Health Organization, “Malaria vector control and personal protection” 7.1 On appropriate strategies and operational issues in eco-epidemiological settings, pp51-53
2. World Health Organization, “Malaria vector control and personal protection” 5. Effectiveness and challenges of implementing malaria vector control and personal protection strategies in the context of the various eco- epidemiological types, pp13-39
3. Janet Fang, “Ecology: A world without mosquitoes” Nature 466, web <[h](http://www.nature.com/news/2010/100721/full/466432a.html)ttp://www.nature.com/news/2010/100721/full/466432a.html>
4. Bill Gates , “The Deadliest Animal in the World”, web <https://www.gatesnotes.com/Health/Most-Lethal-Animal-Mosquito-Week>
5. Hayes, E. B., and D. J. Gubler. 2006. “West Nile virus: Epidemiology and clinical features of an emerging epidemic in the United States.” Annual Review of Medicine 57: pp181‐194
6. D. Raj Raman, Reid R. Gerhardt, John B. Wilkerson “Detecting Insect Flight Sounds in the Field: Implications for Acoustical Counting of Mosquitoes” <http://lib.dr.iastate.edu/cgi/viewcontent.cgi?article=1055&context=abe\_eng\_pubs>
7. Myhrvold, Nathan “Could this laser zap malaria?”, Talk 2010 California, TED <https://www.ted.com/talks/nathan\_myhrvold\_could\_this\_laser\_zap\_malaria>
8. Numpy <http://www.numpy.org/>
9. Matplotlib <https://matplotlib.org/>
10. 1st order vs 2nd order filter image from electronics-tutorials.ws. <http://www.electronics-tutorials.ws/filter/second-order-filters.html>

**Appendix**

**Source Code-**

import Adafruit\_ADS1x15

import time

import numpy as np

import matplotlib.pyplot as plt

from scipy.fftpack import fft , fft2

import RPi.GPIO as GPIO

#Pin Def

detect = 24 #BCM, actual pin 18

zapOpen = 23 #BCM, actual pin 16

#Pin Config

GPIO.setmode(GPIO.BCM)

GPIO.setup(detect, GPIO.IN)

GPIO.setup(zapOpen, GPIO.OUT)

GAIN = 1

adc = Adafruit\_ADS1x15.ADS1015()

sps = input("Input Sampling Rate: ")

time1 = input("Input sample time (seconds): ")

def logdata():

period = 1.0 / sps

datapoints = int(time1\*sps)

dataarray=np.zeros([datapoints,2])

adc.start\_adc(0,gain=GAIN)

#print "Press ENTER to start sampling"

#raw\_input()

startTime=time.time()

t1=startTime

t2=t1

for x in range (0,datapoints) :

dataarray[x,0]= adc.get\_last\_result()

dataarray[x,1] = time.time()-startTime

while (t2-t1 < period) :

t2=time.time()

t1+=period

return (dataarray)

time.sleep(.05)

def feedback():

GPIO.output(zapOpen, GPIO.HIGH)

time.sleep(.1)

zap = GPIO.input(detect)

time.sleep(20)

GPIO.output(zapOpen, GPIO.LOW)

## zapArray[i]=zap

## np.savetxt('zap.txt',zapArray, fmt='%.3f', delimiter = ',')

def main():

#while(1):

dataSamples = logdata()

signal = [column[0] for column in dataSamples[1:len(dataSamples)]]

timeX = [column[1] for column in dataSamples[1:len(dataSamples)]]

transformedSignal = fft(signal)

frequencyMagnitude = np.abs(transformedSignal)

frequencyAxis = np.fft.fftfreq(len(signal)) \* sps

filteredFreqAxis = frequencyAxis[(frequencyAxis> 400)]

filteredMagAxis = frequencyMagnitude[np.where(frequencyAxis > 400)]

zapOn = False

maximum = 0

index = 0

for i in range(0,len(filteredMagAxis)):

if filteredMagAxis[i] > maximum:

maximum = filteredMagAxis[i]

index = i

if filteredFreqAxis[index] > 500 and filteredFreqAxis[index] < 600:

print (filteredFreqAxis[index])

zapOn = True

print "Plotting on Matlab"

fig = plt.figure(1)

ax1 = fig.add\_subplot(211)

ax1.plot(timeX,signal)

ax1.set\_xlabel('Time [s]')

ax1.set\_ylabel('Signal amplitude');

ax2 = fig.add\_subplot(212)

ax2.stem(filteredFreqAxis, filteredMagAxis)

ax2.set\_xlabel('Frequency in Hertz [Hz]')

ax2.set\_ylabel('Frequency Domain (Spectrum) Magnitude')

plt.show()

if zapOn == True:

feedback()

time.sleep(.05)

GPIO.cleanup()

print ('Done')

main()

**Filter Data**

|  |  |  |  |
| --- | --- | --- | --- |
| Second Order | | Unity Gain |  |
| High Pass 473Hz | |  |  |
| frequency | | Input(pkpk) | Output(pkpk) |
| 100 |  | 2.1 | 0.35 |
| 200 |  | 2.13 | 0.59 |
| 300 |  | 2.1 | 0.88 |
| 400 |  | 2 | 1.13 |
| 500 |  | 2.17 | 1.33 |
| 600 |  | 2.2 | 1.53 |
| 700 |  | 2.21 | 1.65 |
| 800 |  | 2.25 | 1.73 |
| 900 |  | 2.25 | 1.81 |
| 100k |  | 3 | 2.3 |
| 150k |  | 2.9 | 1.6 |
| 200k |  | 2.93 | 1.25 |

|  |  |  |  |
| --- | --- | --- | --- |
| Second Order | | Gain=2 |  |
| High Pass 473Hz | |  |  |
| frequency | | Input(pkpk) | Output(pkpk) |
| 100 |  | 2.13 | 0.44 |
| 200 |  | 2.13 | 1.1 |
| 300 |  | 2 | 2.3 |
| 400 |  | 2 | 3.62 |
| 500 |  | 2 | 4.3 |
| 600 |  | 2.17 | 4.7 |
| 700 |  | 2.21 | 4.7 |
| 800 |  | 2 | 4.62 |
| 900 |  | 2.25 | 4.6 |
| 100k |  | 2.93 | 2.17 |
| 150k |  | 2.93 | 1.49 |
| 200k |  | 2.93 | 1.2 |

|  |  |  |  |
| --- | --- | --- | --- |
| Passive First order | |  |  |
| High Pass 473Hz | |  |  |
| frequency | | Input(pkpk) | Output(pkpk) |
| 100 |  | 2 | 0.3 |
| 200 |  | 2 | 1.1 |
| 300 |  | 2 | 1.45 |
| 400 |  | 2 | 1.65 |
| 500 |  | 2 | 1.85 |
| 600 |  | 2 | 1.95 |
| 700 |  | 2 | 2 |
| 800 |  | 2 | 2 |
| 900 |  | 2 | 2 |
| 100k |  | 2 | 2 |
| 150k |  | 2.93 | 1.97 |
| 200k |  | 2.93 | 1.97 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Passive First Order |  |  |
| High Pass 408Hz | | Input(pkpk) | Output(pkpk) |
| 100 |  | 1.83 | 0.45 |
| 200 |  | 1.55 | 0.66 |
| 300 |  | 1.33 | 0.78 |
| 400 |  | 1.25 | 0.82 |
| 500 |  | 1.17 | 0.86 |
| 600 |  | 1.13 | 0.86 |
| 700 |  | 1.09 | 0.86 |
| 800 |  | 1.09 | 0.86 |
| 900 |  | 1.09 | 0.88 |
| 100k |  | 1.29 | 0.86 |
| 150k |  | 1.29 | 0.86 |
| 200k |  | 1.29 | 0.86 |

**Transformer Data**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Transformer Input | | | | Transformer output max output | | | | | | | | | | | |
| 0.7 |  | | | 74 | | | | | | | |  | |  |  |
| 1 |  | | | 124 | | | | | | | |  | |  |  |
| 1.5 |  | | | 200 | | | | | | | |  | |  |  |
| 2 |  | | | 263 | | | | | | | |  | |  |  |
| |  | | --- | |  | |  | | |  | | | | | | | |  | |  |  |
|  |  | | |  | | | | | | | |  | |  |  |
|  |  | | |  | | | | | | | |  | |  |  |
|  |  | | |  | | | | | | | |  | |  |  |
|  |  | | |  | | | | | | | |  | |  |  |
|  |  | | |  | | | | | | | |  | |  |  |
|  |  | | |  | | | | | | | |  | |  |  |
|  |  | | |  | | | | | | | |  | |  |  |
|  |  | | |  | | | | | | | |  | |  |  |
|  |  | | |  | | | | | | | |  | |  |  |
|  |  | | |  | | | | | | | |  | |  |  |
| Transister Data | | |  | | |  | |  | |  | | |
|  |  | |  | | |  | |  | |  | | |
| transister base voltage | | | | | |  | |  | |  | | |
| |  | | --- | |  | |  | |  | | |  | |  | |  | | |
|  |  | |  | | |  | |  | |  | | |
|  |  | |  | | |  | |  | |  | | |
|  |  | |  | | |  | |  | |  | | |
|  |  | |  | | |  | |  | |  | | |
|  |  | |  | | |  | |  | |  | | |
|  |  | |  | | |  | |  | |  | | |
|  |  | |  | | |  | |  | |  | | |
|  |  | |  | | |  | |  | |  | | |
|  |  | |  | | |  | |  | |  | | |
| Emitter Voltage | |  | | |  | |  | |  | |
| |  | | --- | |  | |  |  | | |  | |  | |  | |
|  |  |  | | |  | |  | |  | |
|  |  |  | | |  | |  | |  | |
|  |  |  | | |  | |  | |  | |
|  |  |  | | |  | |  | |  | |
|  |  |  | | |  | |  | |  | |
|  |  |  | | |  | |  | |  | |
|  |  |  | | |  | |  | |  | |