**ECEN 192**

**Lab 8 – Microphone and Speakers**

**Overview:**

In this lab, you will learn the basics of microphones and speakers and how to read and write to audio files. You will also understand the basics of how we convert electric signals into physical sound waves and vice versa, and how we can store this information. We will talk about audio filters, and you will have the chance to explore the effects of several filter functions using your y-badge.

**Introduction:**

One of the most important aspects of any computer system is being able to interact with it. Audio is one of the most common forms of human-computer interaction. Think back to the last 24 hours. You likely used computer audio several times to watch a video, make a phone call, or record something.

**Speakers:**

Speakers work by converting electrical signals, called samples, into sound waves that we can hear. This is done through an electromagnet and a diaphragm. As the electromagnet pulls the diaphragm forwards and backwards, it creates pressure waves that are propagated through the air as sound.

<picture>

<img alt = "animated gif of a speaker, showing the inner mechanisms and function." src="{% link assets/y-board\_mic\_and\_speakers/speaker\_diagram.gif %}" width="200">

</picture>

**Microphones:**

A microphone works in the opposite fashion. There are many different types of microphones, but we’ll talk about the simplest. Your voice or any other sound wave pushes against a diaphragm and moves it back and forth. A coil of wire that’s connected to the diaphragm then moves up and down through a magnet which produces an electric field within the coil of wire which we can read as a voltage.

<picture>

<img alt = "animated gif of a microphone, showing the inner mechanisms and function." src="{% link assets/y-board\_mic\_and\_speakers/mic\_diagram.gif %}" width="200">

</picture>

One of the big questions with storing the data we get from a microphone is how accurate do we want it to be? Here is a sine wave that represents the pure tone A-440:

<picture>

<img alt = "a sine wave on a graph" src="{% link assets/y-board\_mic\_and\_speakers/A-440\_tone\_sine\_wave.png %}" width="200">

</picture>

There are an infinite number of points on this graph that I could measure. Now obviously, we can’t store an infinite number of points, but how many we store determines the quality of our audio. However, there reaches a point where the human ear can’t tell the difference, so we don’t go overboard. Generally, we take 44,100 samples a second. This is called the **sample rate**. Here is the same tone, but how we could sample it.

<picture>

<img alt = "a sine wave on a graph with several vertical bars representing sample points." src="{% link assets/y-board\_mic\_and\_speakers/sampled\_tone.png %}" width="200">

</picture>

**Lab Procedure:**

1. Record an audio file

First open the code for lab 8. Here you will find the basic setup to get your microphone and speaker up and running. First upload the sketch and try to record yourself by speaking while holding down button\_1. Be sure you have your SD card loaded in the y-badge or the board will not be able to save the file anywhere. For best results, be sure to speak into the mic which is located here.

<insert picture of final y-badge version>

Once recording is complete, the LEDs will return back to pulsing white and you may remove the SD card. Verify that you recording worked by plugging your SD card into the lab computer and finding the file “my\_recording.wav.”

1. Graph your audio file

Next, navigate to [this](https://planetcalc.com/8627/) website. Upload your audio file by pressing this button:

<picture>

<img alt = "a picture of an audio converter program." src="{% link assets/y-board\_mic\_and\_speakers/audio\_converter.png %}" width="200">

</picture>

Look at the graph it produces. This shows all of the samples that the y-badge took to record your voice. You can see that it takes many thousands of data points to produce even 10 seconds of audio. That is why audio files can be so large and why so much effort has been put into audio compression.

* What else can you learn from this graph?
* What was the sample rate the y-badge used?
* How many bits were in each sample?

1. Play your audio file

Now recording audio isn’t all that cool if you can’t play it. Put your SD card back into the board and open the lab 8 sketch again. Look through the code before you upload it and make sure you understand what is happening. You will have to insert the line:

Yboard.play\_sound\_file(FILE\_NAME);

somewhere in the code. Find out where it makes the most sense. Your Yboard should play the recorded sound file when you press button\_2. Make sure it only starts playing once the LEDs have changed color.

1. Implement high and low pass filters

Have you ever noticed the knobs on stereos that adjust the treble and bass? A very important part of audio processing is being able to apply filters to improve sound quality. We can use this to increase the bass in a song or remove the high-pitched ringing in a poor audio recording, or even add reverb to someone’s voice.

There is a large block of code (lines 89-104) that is commented out. Remove those comments and explore these filters. The code is almost complete. After you finish it, this block will apply one of these two filters:

Yboard.high\_pass\_filter(FILE\_NAME, OUTFILE\_NAME, cutoff\_frequency);

Yboard.high\_pass\_filter(FILE\_NAME, OUTFILE\_NAME, cutoff\_frequency);

A high pass filter (HPF) allows high frequencies to pass through, whereas a low pass filter (LPF) does the same for low frequencies. These functions take the saved audio file on your SD card, apply the filter, and save the new file under a different name (OUTFILE\_NAME). The parameter cutoff\_frequency refers to the frequency at which the filter will start to apply.

Decide where you should place the line of code for the filters. You will also have to add a line to play the new audio file. Try each of these filters one at a time. If done correctly, your board should apply the filter when switch\_2 is on and button\_2 is pressed and then play the filtered audio.

\*\*\*Before you leave, upload a sketch that implements only the high pass filter. You will need this for the further exploration portion.

**Further Exploration:**

Unless you were screaming, you probably didn’t produce very high frequencies in your recordings to see the effect of a high pass filter. Filters are very powerful, however. Try to see if you can find a source of high-pitched noise (a piano, whistle, high pitched singing etc.) and record yourself talking while the high-pitched noise is playing. Notice how when you apply your filter, the noise is almost completely removed.