E232 Final Project: Amplifier + Crossover Filter Network

Matthew Raghunandan   Angel Ordonez-Retamar
Kai Jiang

May 5, 2022

We Pledge That We Have Abided By The Stevens Honor System

Contents

1 Introduction

Introduction: The goal of this lab report is to analyze and explain our process in amplifying a specific sound file with specific frequencies and playing that sound through a speaker. The sound file was chosen arbitrarily, so we went for the same number file as our lab station (10). All of the sound files have 3 frequencies which were identified using matlab code. To filter the sound file, we used an online website to design two filters, one for low pass which only allowed the lowest frequency to pass, one for high pass which only allowed the highest frequency to pass. We also needed a buffer at the ends of both amplifiers to reduce loading error when connected to the speaker. To finish off the design, we need to be sure that the ampitude of the sound when it got to the speaker was sufficient enough to hear. This meant designing an ampifier at the beggining of the circuit the amplify the signal coming from the computer 10x. Though this sounds simple, there were several steps to ensure that the project was possible, namely ensuring the design worked by checking with simscape and prior lab experience.

2 Materials and Methods

2.1 Step 1: Retrieval of frequencies

Step 1: As mentioned before we first began by using Matlab code (Figure 1) to identify the 3 frequencies. This was allowed to run and gave us a chart (Figure 2) that indicated which three frequencies had the greatest amplitude. The retrieved frequencies are as follows:

High Frequency: 2489 Hz

Mid Frequency: 830 Hz

Low Frequency: 261 Hz

```
[y,Fs] = audioread('C:\Users\mattr\MATLAB\Projects\E232_HW5\sound5.m4a');
x=y(:,1);
% sound(x,44100);
amplitude_spectrum=abs(fft(x));
f=linspace(0,44100,length(x));
stop = 20000/2
  plot(f(1:stop),amplitude_spectrum(1:stop))
```

Figure 1: Matlab Code Depicting The Retrieval of Frequency Values from the Sound File

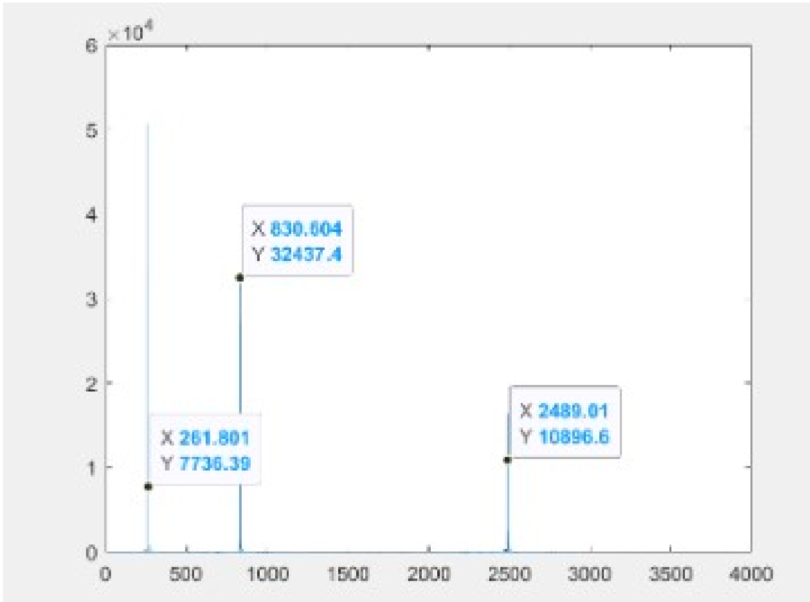

Figure 2: Amplitude Spectra of the Frequencies in the Sound File

2.2 Step 2: Design Of Filters

Step 2: After identifying the frequencies we knew what the stop bands would be from the lowest and highest frequency and so we began designing the filters. The decision on what cut off frequencies to use were made under this logic:

High Pass Filter: This filter needed to pass the high frequency (2489 Hz) while blocking the low (261 Hz) and mid (830 Hz) frequencies. As such, the HP Filter was given a passband of 951 Hz and a stopband of 835 Hz. The attenuation is -12dB.

Low Pass Filter: This filter need to pass the low frequency (261 Hz) while blocking the mid and high frequencies (830 Hz, 2489 Hz). In the end, it was given a passband of 500 Hz and a stopband of 800Hz. The attenuation is -12dB.

We used an 3rd party website that we used allowed us to input frequencies as the passband and the stopband. This made our job much easier as the website would give us the specific capacitor and resistor values and the build.

However, we had to adjust the values and types of filter in order to get resistors and capacitor values that are available to us during the lab. One thing we noticed during this is that the closer the order is to 2nd order, the easier the filter is to build so we prioritized that. The following images are the lowpass and highpass filter designs we ended up using:

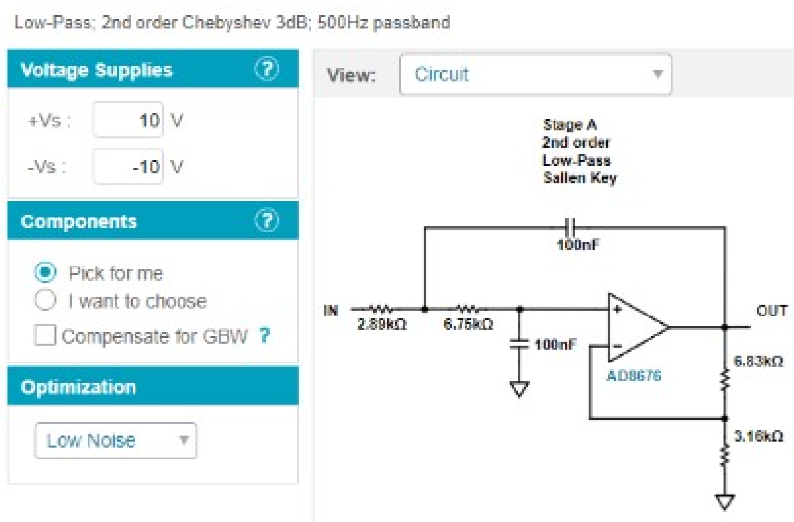

Figure 3: Website Output for the Low Pass Filter

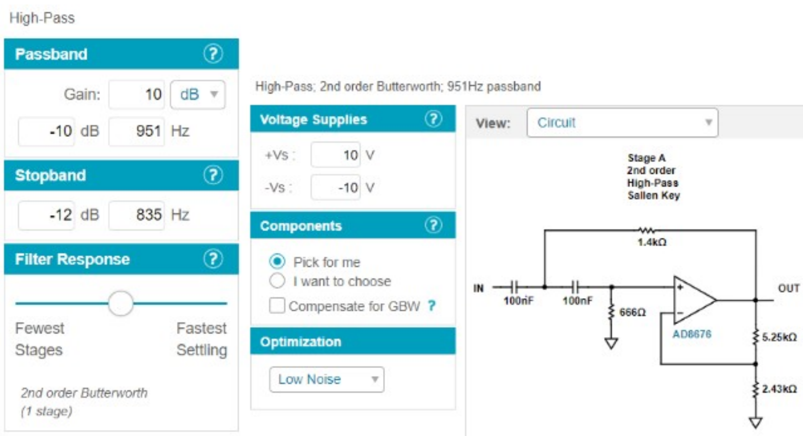

Figure 4: Website Output for the High Pass Filter

2.3 Step 3: Simscape

Step 3: After completing this, an ideal model was created in simscape where we tested and recorded the ideal sound results. As mentioned before, in order to get a sound loud enough to be heard, we first needed an amplifier at the start of the circuit which would need to be powered. This amplifier will have its output connected to both the filters. The outputs of the filters will be connected to a buffer where they will be connected to the speaker to play noise. The following image shows the simscape model:

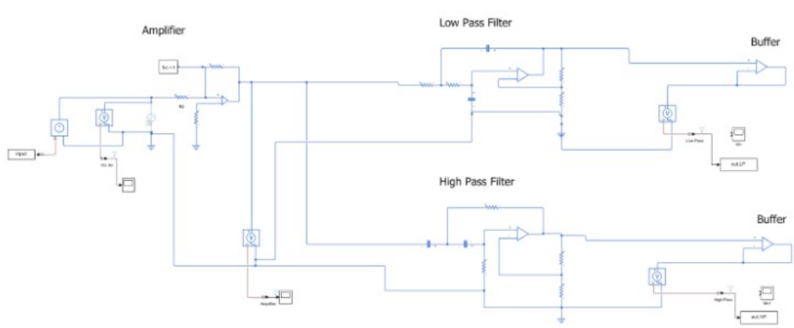

Figure 5: Website Output for the High Pass Filter

2.4 Step 4: Physical Circuit

Step 4: After we had the simscape outline done, we began building the actual circuit. It consisted of the following components:

2.4.1 Amplifier

Amplifier: The amplifier was used to amplify the signal outputted from the computer to ensure it was loud enough to be heard.

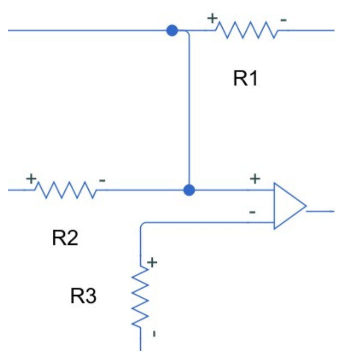

Figure 6: Design Of The Amplifier

Amplifier Components

1. $R_1 = 333\,\Omega$
2. $R_2 = 10000\,\Omega$
3. $R_3 = 5000\,\Omega$
4. LM741 OP-Amp

For this to be possible it needed a gain greater than 10; which for our amplifier we calculated as follows:

$$G = \frac{R_2}{R_1} = \frac{10000\,\Omega}{333\,\Omega} = 30 \tag{1}$$

2.4.2 High Pass Filter:

High Pass Filter: The high pass filter was built on the line as the amplifier and we used values of resistors that were close to the calculated values. The 1.4KOhm resistor was two resistors in series (1K Ohm and 402 Ohm). 5.25K Ohm was 5.1K Ohm, 2.43K Ohm was 2.7K Ohm, and the 666 Ohm was instead 510 Ohm. When designing the circuit on the website, we made sure that the capacitor values were exactly 100 microfarad so we can use the given 100 Microfarad instead of placing more complication on the circuit. Building the circuit, we tried to utilize jumper cables but also tried to make it simple so there wouldn't be confusion troubleshooting. The output of the amplifier was connected to the input of the high pass filter like described in the simscape mode. The end of this high pass filter was connected to a simple buffer design. After having this built we tested the sound through the speaker only using high pass inputs.

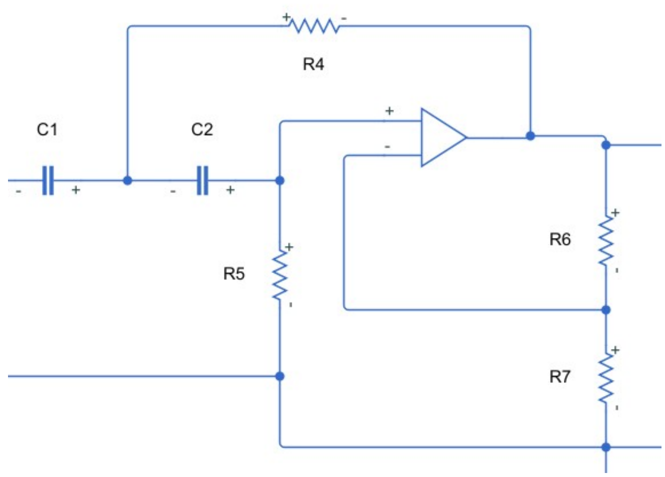

Figure 7: Design Of The High Pass Filter

High Pass Components

1. $R_4 = 1.4E3\,\Omega$
2. $R_5 = 667\,\Omega$
3. $R_6 = 5.25E3\,\Omega$

4. $R_7 = 2.43E3\,\Omega$
5. $C_7 = 100E - 9\,F$
6. $C_7 = 100E - 9\,F$
7. LM741 OP-Amp

2.4.3 Low Pass Filter:

Low Pass Filter:

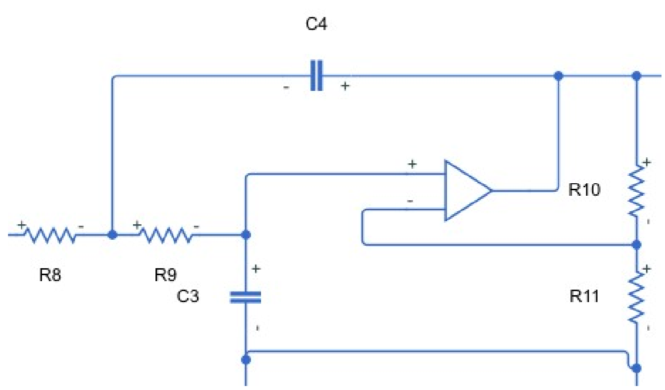

Figure 8: Design Of The Low Pass Filter

Low Pass Components

1. $R_4 = 1.4E3\,\Omega$
2. $R_5 = 667\,\Omega$
3. $R_6 = 5.25E3\,\Omega$
4. $R_7 = 2.43E3\,\Omega$
5. $C_7 = 100E - 9\,F$
6. $C_7 = 100E - 9\,F$
7. LM741 OP-Amp

2.4.4 Buffer And Aux Cord:

Buff And Aux Cord: Buffers were connected to the low pass filter and the high pass filter to remove the loading error. Additionally, in order to get the speaker to work we first needed an aux cord and a way to connect them from the computer to the input of the breadboard. To do this, we put together two banana plugs to get two ends with clips and we made two of them. One end of the aux cable was connected to a laptop playing the audio and the other end was connected to two clips, ground at the bottom and positive at the top. This would finish off the circuit.

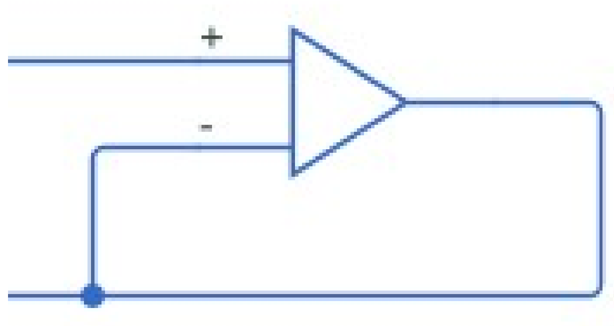

Figure 9: Design Of The Buffer

2.4.5 Complete Materials List:

Currently our materials list only list the ideal values that we would have liked to used for our filter. Our final materials list is listed below:

Material	**Total Quantity**
Op-amp	5
Capacitor (100 uf)	4
Resistor 1K Ohm	5
Resistor 402 Ohm	2
Resistor 5.1 K Ohm	3
Resistor 2.7 K Ohm	3
Resistor 510 Ohm	1
Resistor 10000 Ohm	1
Resistor 333 Ohm	1

Figure 10: Final Materials List

2.4.6 Physical Circuit:

Our final circuit is depicted below. High pass on far left, low pass on middle. Buffers on the bottom and amplifier on far top left.

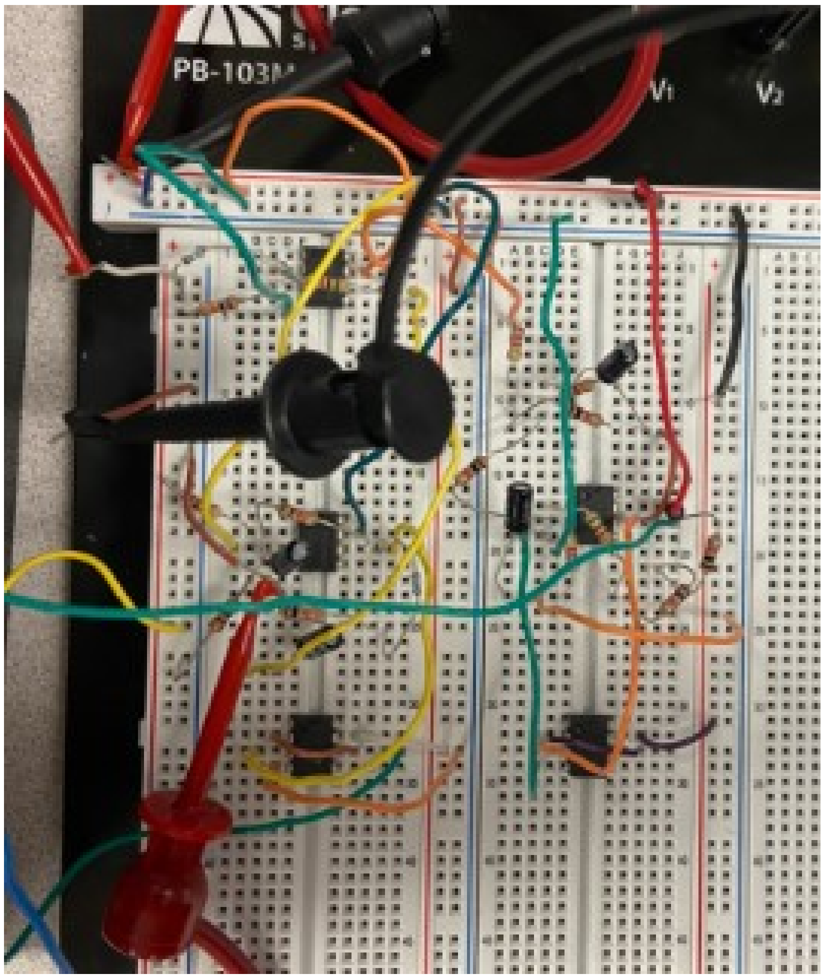

Figure 11: Final Circuit

3 Results and Discussion

Results and Discussion: Using the circuit setup we were able to test the frequencies values. Notably the low frequency of 261 Hz and the high frequency of 2490 Hz.

Initially, when we tested just the high pass portion of the circuit after we completed it, it made very little noise. We first thought it was a problem with the circuit build or op-amps but after troubleshooting we found out that was not the case. However, we eventually figured out that since our amplifier is inverting, we needed to switch up the inputs to the filters by sampling and rearranging the jumper wires. This fixed the low noise issue. When the lowpass was completed, there was also the issue of low noise when first tested. This one however was a circuit building mistake rather than connection error.

When we had all the troubleshooting done and saw our results, they were acceptable. We listened to the high pass and low pass by ourselves and then both at once. We also called over our TA who listened and gave us his approval. However, in our physical circuit, we experienced some error with our high pass filter. Compared to the Matlab simulation, our physical filter came out sounding like a much lower frequency. This also in turn caused our circuit as a whole to sound a bit different than the Matlab simulation. The low pass filter, however, was spot on with the Matlab simulation. (All of these sounds are listed in the appendix or submitted alongside the report).

Upon analyzing our project as a whole, and what could have caused this discrepancy with our high pass filter, we concluded that while there was error it was likely within the margin of acceptable error. The slight difference in sound could have been due to messy circuitry (bad wires, bad amplifiers, or general in-person errors that are not present in Matlab) or because our amplifier did not amplify the signal enough to detect that high frequency. If the project were to be analog-to-digital, the minimum sampling frequency would be twice the highest cutoff. This would mean $951Hz * 2 = 1902Hz$.

Our review of this project is mostly positive. We enjoyed the concept of it much more than what we had been doing in this course up until now or in circuit one. The fact that we built a filter and amplifier that we could then play whatever sound we wanted through it just to phear how it changed the original sounds was really fun. It also did a great job of making us use all of the different skills that we had learned in the course: amplifiers, filters, Matlab, and, of course, writing lab reports. We also enjoyed the presentation aspect, even though we were unable to present in person.

A few areas that could be improved for future classes would be: cleaning out faulty amps/wires, and more resources to use such as electrical tape. One big problem that our group faced not only in this final project but throughout the semester was amplifiers being burnt out or broken, as well as wires that weren't working very well/a general lack of jumper cables. If this area could be fixed it would just make the labs a bit less tedious. Finally, our last suggestion is some electrical tape or a better way to connect the auxiliary cable to the circuit. As you can see in our videos, we were holding the wires to the cable at all times, and this was not a very stable way to keep it connected to the circuit. On top of that, it was just very uncomfortable. Some electrical tape to hold the wires in place would have freed up some hands.

4 Appendices

```
[y,Fs] = audioread('C:\Users\mattr\MATLAB\Projects\E232_HW5\sound5.m4a');
x=y(:,1);
% sound(x,44100);
amplitude_spectrum=abs(fft(x));
f=linspace(0,44100,length(x));
stop = 20000/2
  plot(f(1:stop),amplitude_spectrum(1:stop))
```

Figure 12: Matlab Code for Plotting Amplitude Spectra

```
[Audio, Fs] = audioread('Audio 10.wav'); % Reads the audio file
Input=zeros(length(Audio),2); % Creates a 2 column array with length of Audio, filled entirely with zeros
Input(:,2)=Audio(:,2); % Sets the right column of Input equal to the right column of Audio

for n=2:length(Input) % Creates timestamps for each reading
Input(n,1)=Input(n-1,1)+1/Fs;
end



sim('CircuitModel.slx'); % Runs the Simscape file (SHOULD BE IN THE SAME FOLDER AS YOUR AUDIO FILE)
out = ans; % Stores the outputs of the Simscape file in "out"
sound(Input,Fs); % Plays the original audio file
```

```
sound(out.LP.Data,Fs); % Plays the output of the woofer (after low pass filter)
audiowrite("LP.wav",out.LP.Data,Fs)
```

```
sound(out.HP.Data,Fs); % Plays the output of the woofer (after high pass filter)
audiowrite("HP.wav",out.HP.Data,Fs)
```

```
sound(out.LP.Data+out.HP.Data,Fs); % Plays the output together?
audiowrite("Both.wav",out.LP.Data+out.HP.Data,Fs)
```

Figure 13: Matlab Code for Simulating Simscape Model and Playing Sound

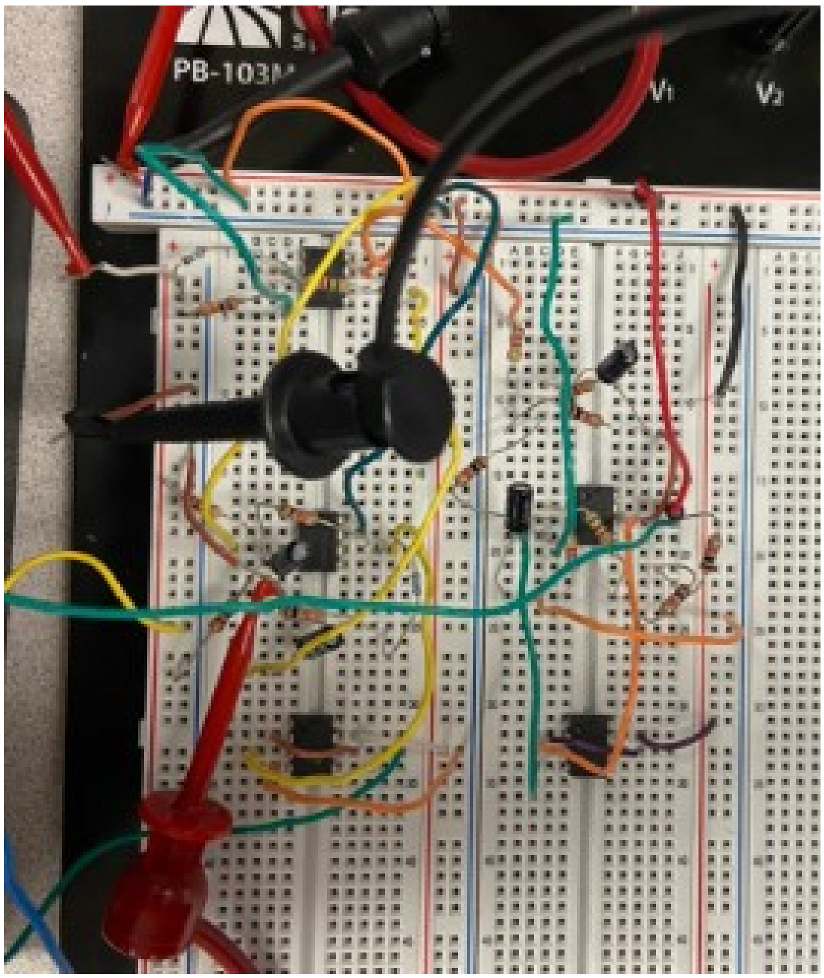

Figure 14: Final Circuit

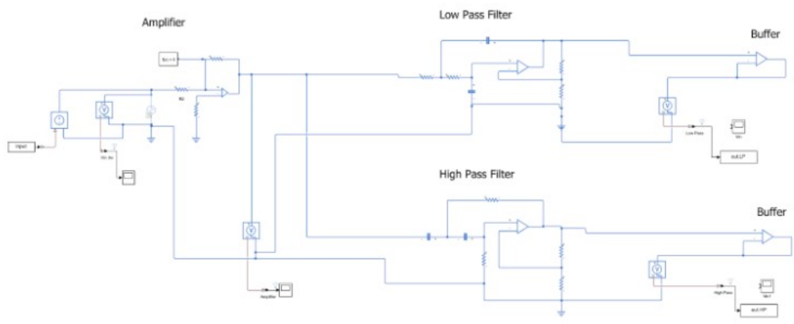

Figure 15: Simscape Model

Low Pass Video: https://www.youtube.com/shorts/puThcnoXgqc
High Pass Video: https://www.youtube.com/shorts/prP8dKTg-p4
Both Video: https://www.youtube.com/shorts/etaBgdwPG2I