FUN2 Final Project Report

Andrew Vithoulkas, Joey Cohen, and Daniel Sarria

***Abstract*—This project has a multiple purposes, each of which play a role in cementing key understandings of course material and engineering as a whole. The first is the real-life implementation of topics taught in class, including summing amplifiers, 2nd order high and low pass filters, peak detectors, and MOSFETS; Most of which use the underlying component of an Operational Amplifier (Op-Amp) in the negative feedback configuration. Developing an understanding of negative (and sometimes positive) feedback in this context allows us as students to be able to calculate component values and use our knowledge to create a circuit like this. The project also allowed us to work in groups to accomplish goals and troubleshoot parts. At almost every part in the circuit was some sort of problem, so our only option was to work together as a design team to debug and fix the problem.**

The main goal of the circuit is to light up an LED with a corresponding frequency when a song is played. This happens using a combination of the previously mentioned components. High frequencies are blocked by the low pass filter but allowed through the output of the high pass filter, allowing for the green LED to turn on for high frequencies. The same happens for low frequencies on the other side of the circuit, where the red LED is on for a low frequency input. Our peak detectors work with our MOSFET to ensure that the LEDs are operating at a frequency that aligns with the vision of the naked eye to ensure that the LED can be seen as on, but obvious voltage/frequency changes can be noted. The song we chose for this project is "All We Know" by the Chainsmokers, which has plenty of high and low frequencies to be portrayed by the LEDs.

I. Background Information and Rationale

In this design project, we aimed to create a system that visually represents the frequency components of a chosen song, "All We Know" by the Chainsmokers. The song uses both low and high frequencies, with a strong bass line and drumbeat, as well as high-pitched vocals to create a diverse mix. The song's bass line and drumbeat provide a strong foundation for low frequencies and the high-pitched vocals add prominent high frequencies within the song. The cutoff frequency of the low pass filter is 400 Hz and the cutoff frequency of the high pass filter is 800 Hz. These values were chosen because we wanted to have equal amounts of high and low frequencies in order to have both of the LEDs light up roughly equivalently throughout the song. As a result, we divided the frequency spectrum of the song to have equal areas of frequency.

The system's overall design is organized into several blocks: a summing amplifier, Sallen-Key amplifiers, including low pass and high pass variants, peak detectors, and LED drivers.

This is the final project report for ECE 2660 Fundamentals 2, Spring 2023 at the University of Virginia

II. Design and Analysis

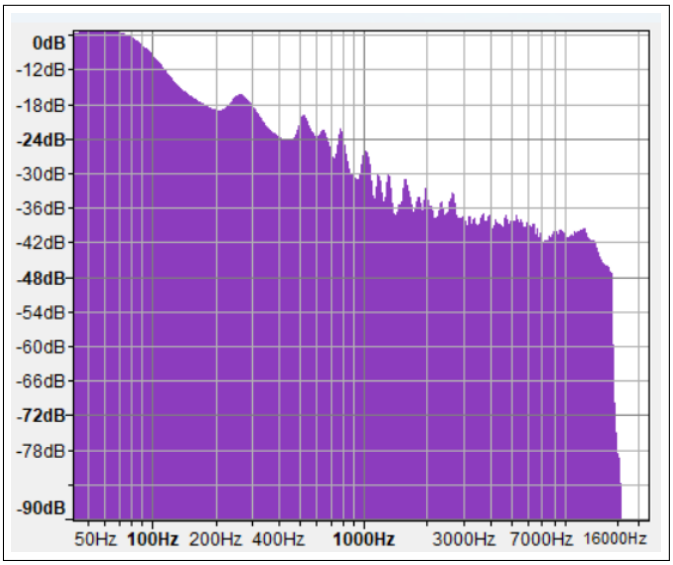


Fig. 1: Frequency Spectrum of 'All We Know' by the Chainsmokers

Our high-level strategy involved dividing the frequency spectrum into low and high-frequency bands, using cutoff frequencies of 400 Hz for the low pass filter and 800 Hz for the high pass filter. This choice ensured that both low and high frequencies were equally represented, allowing two sets of LEDs to light up consistently throughout the song. The overall system design is divided into multiple blocks. The first block is the summing amplifier that takes in our 1V input voltage. The summing amplifier will have a gain of 3 because a 1V input voltage results in a 3V output which is within the range of our 4.5V power supply.

The second block consists of the Sallen-Key amplifiers. The Sallen-key amplifiers are made up of the low pass Sallen-key amplifier and the high pass Sallen-key amplifier. The low pass Sallen-key amplifier will have a gain of 1 because any increase in gain results in railing of the op-amps, which is what we want to avoid. The corner frequency of the low pass Sallen-key filter will be 400 Hz as we noted previously. The high pass Sallen-key amplifier will also have a gain of 1 for the same reason as the low pass Sallen-key amplifier. The corner frequency of the high pass Sallen-key amplifier will be 800 Hz as we noted previously for our song.

The third block consists of the peak detectors. The peak detectors will have a frequency range between 30-60 Hz in order to avoid flickering of the LEDS.

The last block consists of the LED drivers. The LED drivers consist of the bias resistors and the source resistors. The bias resistors are set to have a gate voltage below the voltage threshold. The source resistors are set to limit the current through the LEDs.

A. Summing Amplifier

In order to implement the summing amplifier, we designed it to obtain a corner frequency near 20 Hz with resistor values that were at least $10k\Omega$. Additionally, all input resistor values and capacitor values had to be obtained from our lab kits. We also decided to produce a linear gain of 3 in the pass band for the summing amplifier in order to avoid railing the op amp given our 4.5V power supply. The summing amplifier schematic is shown below:

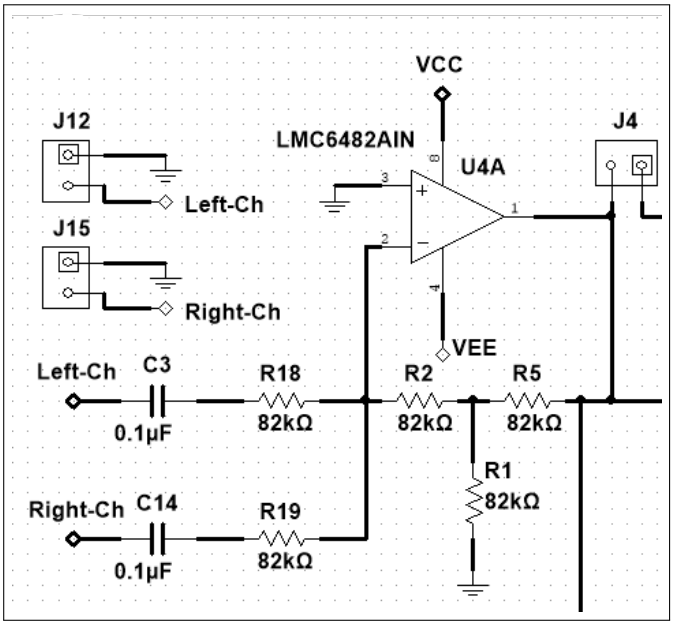


Fig. 2: Summing Amplifier Schematic on Multisim with Chosen Resistor and Capacitor Values

In order to determine C_3, C_14, R_{18}, and R_{19}, the following equations were used:

$$f_{RC,cutoff} = \frac{1}{2\pi RC} \tag{1}$$

Using a capacitor value from the lab kit: $0.1\mu F$

$$R = \frac{1}{40\pi(0.1 \cdot 10^{-}6)} \approx 79.57k\Omega \tag{2}$$

Closest lab kit resistor: $82k\Omega$

$$\frac{1}{2\pi(82 \cdot 10^{3})(0.1 \cdot 10^{-}6)} = 19.409Hz \approx 20Hz \tag{3}$$

Therefore, the following component values are as follows:

$$R_{18} = 82k\Omega$$

$$R_{19} = 82k\Omega$$

$$C_3 = 0.1\mu F$$

$$C_{14} = 0.1\mu F$$

To determine the T-network resistors R_2, R_5, and R_1, the following equations were utilized:

$$R_f = R_2 + R_5 + \frac{R_5 R_2}{R1} \tag{4}$$

$$V_{out} = -(V_L + V_R)\frac{R_f}{R_{in}} \tag{5}$$

Then, a Java program that implemented these equations was created to calculate every possible combination of resistors that would meet the requirement of a linear gain of 3. The Java code is shown in the appendix Fig. 30.

In the code, an input voltage of 2V was used to calculate all the possible combinations of resistor values that met the criteria of a linear gain of 3. However, it is important to note that a 2V input would result in a 6V peak to peak output which would result in the summing amplifier to peak. Therefore, our actual input voltage would be 1V instead, which is a 3V peak to peak output.

The resulting code produced many combinations, but we ultimately chose a combination that resulted in a tolerance value of 0, meaning it had an exact output voltage of 6V. The resulting resistor values are shown below:

$$R_1 = R_2 = R_5 = 82k\Omega$$

B. High Pass Filter (HPF)

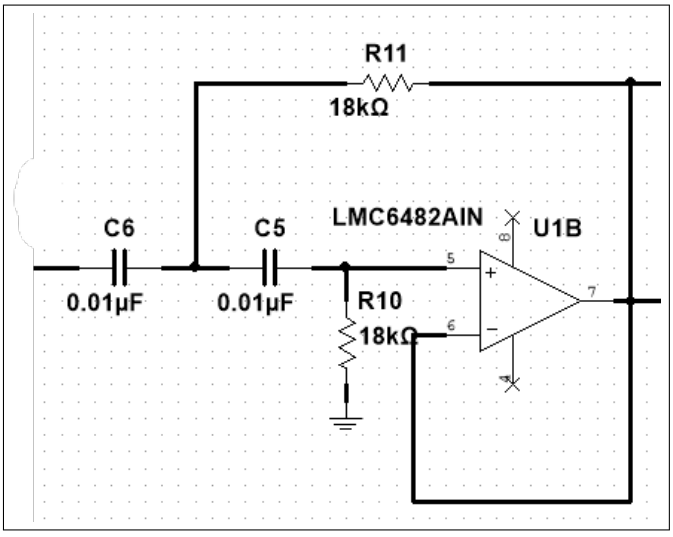


Fig. 3: Sallen-Key High Pass Amplifier Schematic on Multisim with Chosen Resistor and Capacitor Values

When designing the High Pass Filter, it was noted from Fig. 1 that a frequency of 800Hz was desired in order to equally split the area of the spectrum graph between high and low frequencies. The purpose of this is so that the song will trigger the high frequency LED as well as the low

frequency LED so that they are operating for roughly equal time periods throughout the song. The Q value for this filter was required to be between 0.5 and 0.707 so that the filter is critically damped to ensure around a -6dB gain at the corner frequency and a -40dB/dec drop in magnitude.

In order to make the computation easier while working with the limitations of the lab kit, the Q value was set to 0.5. It was accepted that this was a trade-off between performance, analytical ability, and availability in the lab kit. All resistor values were chosen to be over 10k Ohms to ensure no current issues would arise in the circuit. The transfer function for a standard 2nd order high pass filter is as follows:

$$H(s) = \frac{Ks^2}{s^2 + s\frac{w_0}{Q} + w_0^2} \tag{6}$$

The -6dB gain at the corner frequency can be confirmed from our Q value of 0.5:

$$20log0.5 = -6.02dB \tag{7}$$

To find the R10, R11, C5, and C6 values, Java code was written in Fig. 31 that would iterate through all values of the lab kit to achieve the following:

$$Q = \frac{\sqrt{R_{10}R_{11}C_6C_2}}{R_{11}C_5 + R_{11}C_6 + R_{10}C_5(1-K)} \tag{8}$$

Where the gain K, is set equal to 1 because in a normal 2nd order high pass filter, K would be dependent on two other resistors; in this case, one of which is treated as an open circuit, the other of which is treated as a short circuit. The calculated resistor and capacitor values that satisfy a Q value of 0.5 is:

$$R_{10} = 18k\Omega$$

$$R_{11} = 18k\Omega$$

$$C_5 = 0.1\mu F$$

$$C_6 = 0.1\mu F$$

The corner frequency for this circuit can also be calculated from our resistor values:

$$w_0 = \frac{1}{\sqrt{R_1R_2C_1C_2}} \tag{9}$$

Leading to a corner frequency for the high pass system in this circuit of:

$$w_0 = 884.2Hz$$

Another design that could be used would be one that would result in a closer corner frequency to our initially desired 800Hz. However, as previously stated, this approach was not used in order to comply with our analysis of the circuit as well as the limitations of our lab kit. The trade-off for this approach is a non-exact corner frequency. This could lead to less time that the high frequency LED is on during the song.

C. Low Pass Filter (LPF)

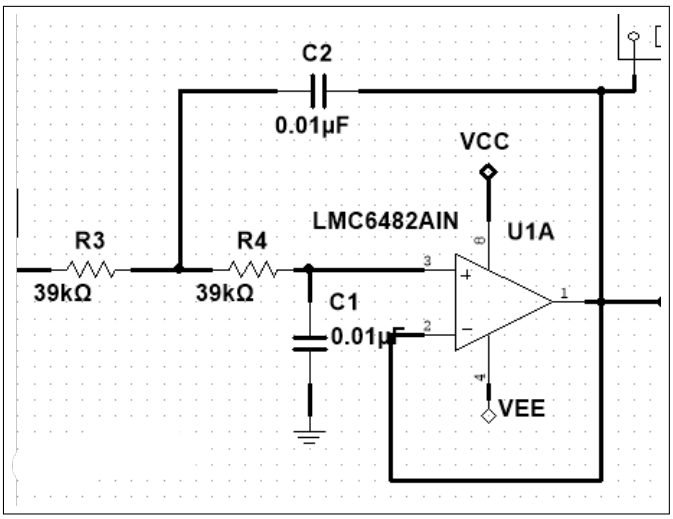


Fig. 4: Sallen-Key Low Pass Amplifier Schematic on Multisim with Chosen Resistor and Capacitor Values

For the Sallen-Key low-pass filter, our corner frequency is set to 400Hz because it divides the frequencies of the song roughly equally Fig. 1, allowing the LED to light up the same amount as the higher frequencies throughout the song. The K value is set to 1 so the output of the summming amp stays the same and doesn't increase, hitting the rail. This results in $K = 1$ due to the equation, $K = 1 + \frac{R_f2}{R_f1}$ where $\frac{R_f2}{R_f1}$ approaches zero.

The quality factor (Q) was selected as 0.5 to make the analysis easier, which lies within the acceptable range of 0.5 to 0.707 for Sallen-Key filters. This range is set to ensure the filter is critically damped with a -6dB gain at the corner frequency and a -40dB/dec drop in magnitude. We then set the equation (10) to 0.5 and solved for various R_3, R_4, C_1, and C_2 values that gives the correct Q value using the python script, (Appendix Fig. 13):

$$Q = \frac{\sqrt{R_3 \cdot R_4 \cdot C_1 \cdot C_2}}{R_3 \cdot C_1 + R_4 \cdot C_1 + R_3 \cdot C_2(1-K)} \tag{10}$$

Capacitor values C_1 and C_2 were both chosen as 0.01uF, based on the available components in the lab kit. This choice was arbitrary since the desired Q, K, and cutoff frequency (f_c) values can still be achieved by adjusting R1 and R2. Once the capacitor values were chosen, R_3 and R_4 were selected to be 39kΩ and 39kΩ because they were in our lab kit and were above 10kΩ. It's important to note that the python script produced a bunch of values that work with our Q and K values and the C_1, C_2, R_3, and R_4 values were one of these options that happened to also work with our lab kit materials. Next we solved for ω_o (11) from our now chosen values.

$$\omega_o = \frac{1}{\sqrt{R_3R_4C_1C_2}} \tag{11}$$

We have all the information we need to verify our choice using the corner frequency equation (12). The analysis produced a corner frequency of 408.25 Hz which is within a 5% error of the desired 400Hz.

$$f_c = \frac{1}{2\pi\sqrt{R_3 R_4 C_1 C_2}} \tag{12}$$

The desired gain of -6dB at the corner frequency can be verified using our Q value (13):

$$20\log(0.5) = -6.02dB \tag{13}$$

The response of the system can be checked using the low-pass transfer function (14) below using all of the values we have solved for above:

$$H(s) = \frac{K\omega_o^2}{s^2 + s\frac{\omega_o}{Q} + \omega_o^2} \tag{14}$$

To check the Sallen-Key low-pass circuit's performance, Multisim was used to execute an AC Sweep analysis. The typical low-pass Sallen-Key filter was built and tested for this sweep. The results of the AC Sweep test are shown in the figure above (Figure 2). To analyze the transfer function, we put the red cursor on the flat portion (pass band) of the asymptotic curve and then placed a blue cursor at the specific desired frequency of 400Hz (low-pass corner frequency) to check if the decibel (dB) drop (dy) corresponds to the target value of -6.02 dB. In this particular case, the outcome was close to the precise value, being -6.01 which is within 5% error. We believe this error came from not being able to place the yellow cursor exactly at 400Hz. As a Low Pass Filter, this outcome verifies the designed circuit and demonstrates the expected graphical representation.

D. Peak Detector

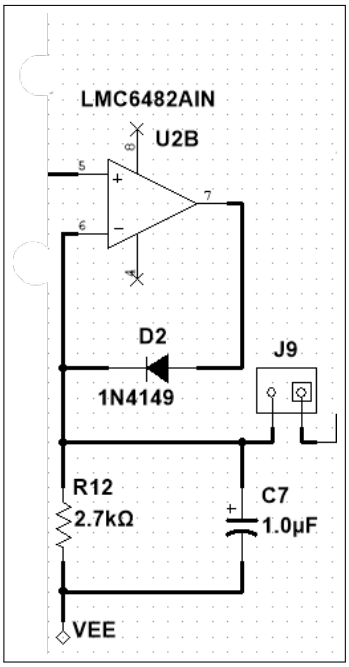


Fig. 5: Multisim Schematic of the High Pass Peak Detector

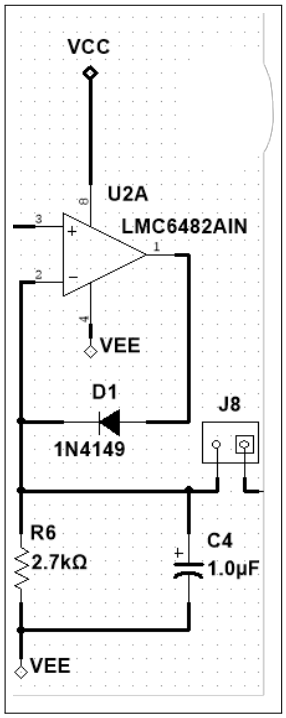


Fig. 6: Multisim Schematic of the Low Pass Peak Detector

In order to design the peak detector, it was necessary to realize that the time constant was dependent on R_{12} and C_7 in Fig. 5 and similarly on R_6 and C_4 in Fig. 6. Therefore the following equation was used:

$$\tau = RC \tag{15}$$

Additionally, based off non-scientific estimates, the human eye can perceive changes from 30-60 Hz, which gives us a rough estimate for calculating the resistor and capacitor values for the peak detector. We aimed to obtain a frequency in the upper range of the range of frequencies while staying within the boundaries in order to prevent too much flickering of the LEDs. We chose an initial capacitor value of $1.0\mu F$ due to the limited capacitors in our lab kit and the size constraints on the PCB. We then went through multiple resistors using the equation below to calculate our desired frequency till we reached a resistor value of $2.7k\Omega$ to get us a frequency of 58.94 Hz.

$$\frac{1}{2\pi RC} \tag{16}$$

$$\frac{1}{2\pi(2.7 \cdot 10^3)(1.0 \cdot 10^{-6})} = 58.94Hz$$

E. LED Driver

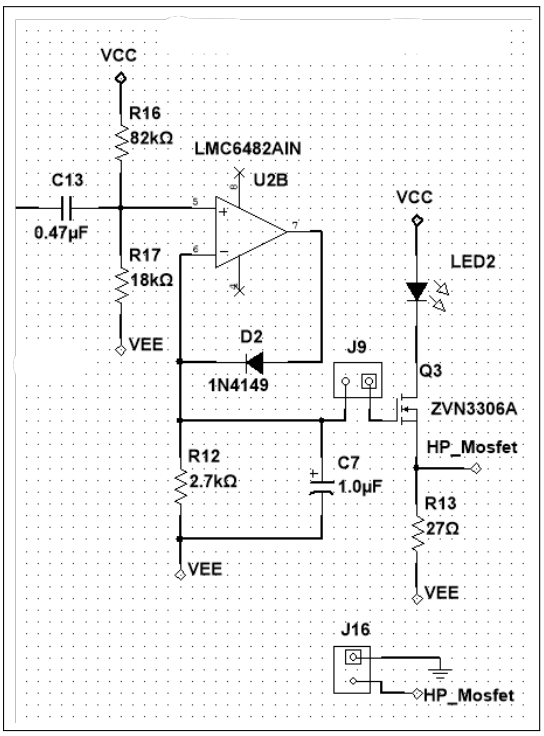


Fig. 7: Multisim Schematic of the High Pass LED Driver

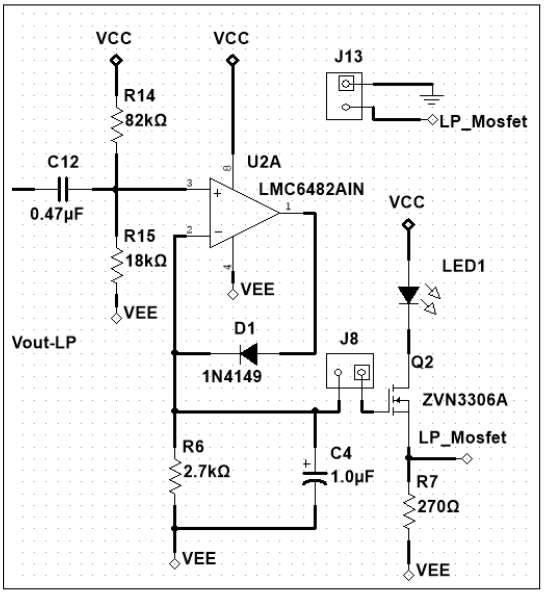


Fig. 8: Multisim Schematic of the Low Pass LED Driver

In order to design the LED driver, it was necessary to measure the K_N and the V_T values of both LP MOSFET and HP MOSFET. The following results of such values are below:

LP MOSFET	Value
K_N	0.040133
V_T	2.05

HP MOSFET	Value
K_N	0.05238
V_T	2.08

Because the V_T values of both MOSFETS are similar to each other, it was safe to assume that the biasing resistors would be identical to each other. Additionally, the BFCs, C_{13} and C_{12}, on both MOSFETS would be set to the highest ceramic capacitor in our lab kits, which is $0.47\mu F$. In steady-state, the peak detector will pass a DC signal with no change, so R_{16} and R_{17} (as well as R_{14} and R_{15}) set the gate voltage V_G when no external signal is applied. As a result, we selected our bias resistors to generate a gate voltage that was less than our threshold voltage in order for the LED to remain off when no audio signal is applied and turns on when there is signal present. The Java code in Fig. 33 was used to produce the following values of the bias resistors with the resulting gate voltage:

$$R_{14} = R_{16} = 82k\Omega$$

$$R_{15} = R_{17} = 18k\Omega$$

$$V_G = 1.62V$$

Lastly, in order to calculate the source resistor for each LED, it was necessary to refer to the green [1] and red LED [2] data sheets to find the absolute maximum current. Specifically, the red LED had a maximum current of 7 mA and the green LED had a maximum current of 30 mA.

Then, we measured the maximum output of the peak detector using a 3V input wave through the low pass Sallen-Key filter, which resulted in -0.68946V. The maximum voltage was then calculated as follows:

$$Maximum\ voltage = -0.68946 - (-4.5) = 3.81054V \tag{17}$$

In terms of the red LED, we decided to set the I_{LED} to be 5 mA to stay below the maximum current, but produce enough current for the red LED to be bright enough when the audio signal comes through. The following equations were then used to calculate the source resistor R_7 for the red LED:

$$I_{DS} = \frac{K_N}{2}(V_{GS} - V_T)^2 \tag{18}$$

$$I_{DS} = \frac{K_N}{2}(V_G - V_S - V_T)^2 \tag{19}$$

$$I_{DS} = \frac{K_N}{2}(V_G - I_{DS} \cdot R_s - V_T)^2 \tag{20}$$

Plugging in the corresponding values:

$$0.005 = \frac{0.040133}{2}(3.81054 - 0.005R_7 - 2.05)^2$$

$$R_7 = 270\Omega$$

In terms of the green LED, we decided to set the I_{LED} to be 25 mA to stay below the maximum current, but produce enough current for the green LED to be bright enough when the audio signal comes through. The same equations for the red LED were then used to calculate the source resistor R_{13} for the green LED:

$$0.025 = \frac{0.05238}{2}(3.81054 - 0.025R_{13} - 2.08)^2$$

$$R_{13} = 27\Omega$$

III. Simulations

A. Summing Amplifier

A Multisim simulation was conducted with all the calculated resistor and capacitor values, and a 1V input voltage, meaning each channel would receive 0.5V as its own individual input voltage. The resulting frequency response of the left and right channels are displayed below:

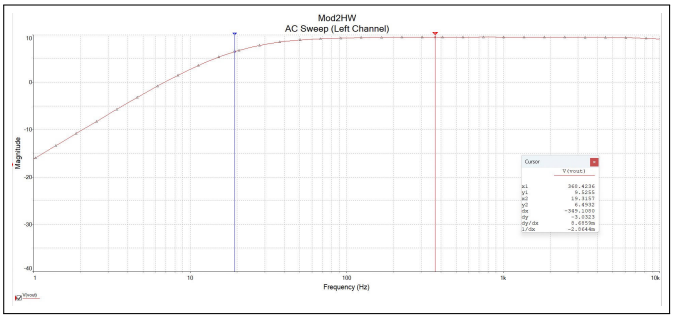


Fig. 9: Left Channel of the Summing Amplifier with a Gain of 3, a Corner Frequency of 19.31 Hz, and a -3 dB Drop

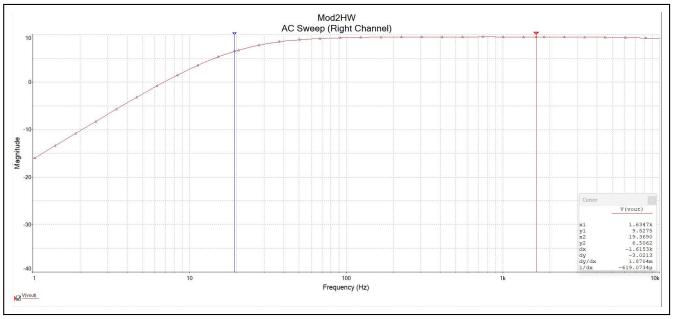


Fig. 10: Right Channel of the Summing Amplifier with a Gain of 3, a Corner Frequency of 19.37 Hz, and a -3 dB Drop

B. High Pass Filter

A Multisim simulation was conducted with all calculated resistor and capacitor values, along with a 1V input voltage at a 1kHz frequency. The figure below shows the attenuation decreasing as frequency increases up to the corner frequency of 888 Hz at the correct -6dB from the pass-band, aligning very closely (within 5%) with analytically calculated values for a high pass filter.

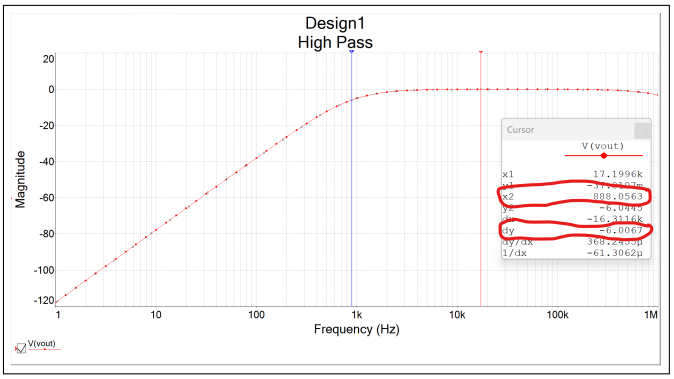


Fig. 11: High Pass Filter Simulation showing a corner frequency of 888Hz at -6dB from pass-band

C. Low Pass Filter

A Multisim simulation was conducted with all calculated resistor and capacitor values, along with a 1V input voltage at a 1kHz frequency. The figure below shows the attenuation increases as frequency increases after the corner frequency of 408 Hz at the correct -6dB from the pass-band, aligning very closely (within 5%) with analytically calculated values for a high pass filter.

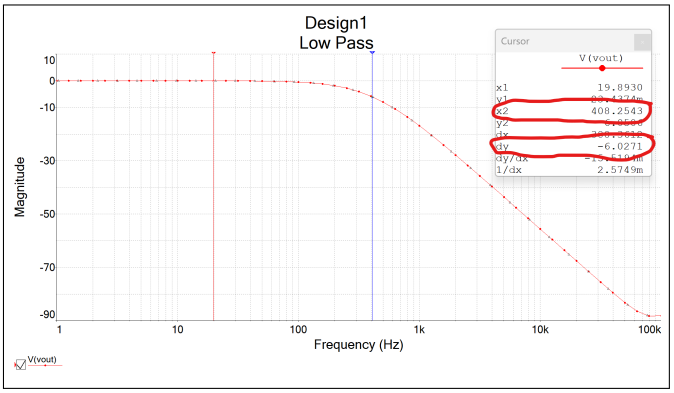


Fig. 12: Low Pass Filter Simulation showing a corner frequency of 408Hz at -6dB from pass-band

D. Peak Detector

A Multisim simulation was conducted with all the calculated resistors and capacitor values, along with a 3V input voltage at a 200 Hz frequency into the low pass Sallen-Key filter. The figure below shows the output of the low pass Sallen-Key filter, which is the input into the peak detector. The figure shows that the maximum voltage of the peak detector is approximately 3.79V which is within the range of our analytical calculation of the maximum voltage (equation 17).

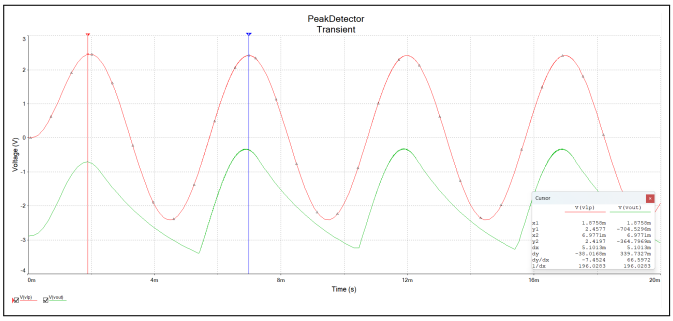


Fig. 13: Low Pass Peak Detector with a maximum voltage of $-0.704 + 4.5 = 3.79V$

E. LED Driver

A Multisim simulation was conducted with the calculated source resistor, along with a 3V input voltage at a 200 Hz frequency into the low pass Sallen-Key filter. The figure below shows the output of the drain source current. The figure shows that the maximum current of the LED driver is approximately 7.3 mA which is just above the maximum current of the red LED according to the data sheet, and 2.3 mA off from our analytical result of 5 mA.

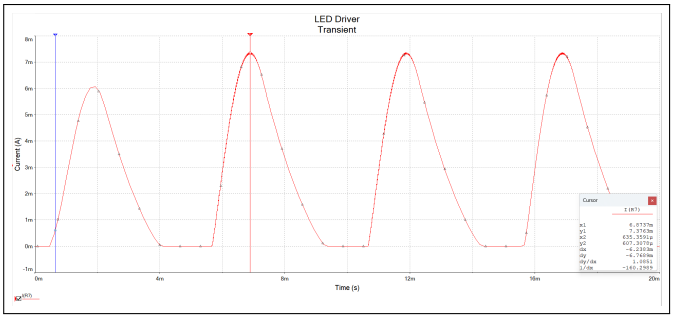


Fig. 14: Low Pass LED Driver with a Maximum Current of 7.3 mA

A Multisim simulation was conducted with the calculated source resistor, along with a 3V input voltage at a 1 kHz frequency into the high pass Sallen-Key filter. The figure below shows the output of the drain source current. The figure shows that the maximum current of the LED driver is approximately 28.6 mA which is below the maximum current of the green LED according to the data sheet. This result is 3.6 mA off from our analytical result of 25 mA.

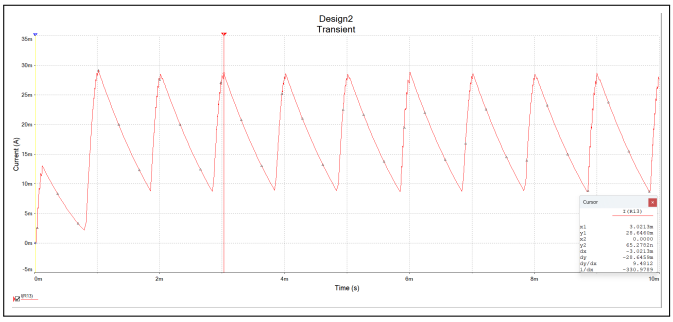


Fig. 15: High Pass LED Driver with a Maximum Current of 28.6 mA

F. System Simulations

First, a Multisim simulation was created with a 0.5V amplitude AC voltage source operating at 100Hz was applied to each left and right channel input of the circuit. This frequency was chosen in simulation so that it was within the pass-band for the low pass filter (before 408Hz), allowing for minimal to 0 attenuation of the signal from the low pass filter. The summing amplifier should take each 0.5V source, combine them to make 1V, then produce a gain of 3V, making the final output for the summing amplifier to be 3V. This output at a frequency within the pass-band of the low pass filter should pass straight through, allowing the signal to be handled by the peak detector on the low frequency side, thus lighting up the diode at this low frequency. The below simulation figure shows exactly this. The blue line is showing the simulated voltage measured at the output of the low pass filter, which outputs the 3V without attenuation as coming from the summing amplifier, indicating that the frequency is within the pass-band of the low pass filter. The red line measures the voltage at the gate of the MOSFET after the peak detector. This proves that our peak detector is working correctly, as this signal is only rising in voltage when the output from the low pass filter is. When the output is decreasing to a negative voltage, the time constant (previously calculated from resistor and capacitor values) holds the voltage up until the next rising segment. This is seen from the unnatural decrease in voltage in between peaks. Our peak detector is also seen to be calculated correctly, as it is operating at our previously calculated 58.94Hz. The peaks of the MOSFET gate voltage allow current to flow between the drain and the source, which in turn flows current through the LED. The green line represents the voltage at the source of the MOSFET, which proves that when this peaks (note 4.5V should be added to this to get the voltage in relation to ground, as the resistor goes from the source to VEE, or -4.5V). This combined voltage needs to be above the threshold voltage for the MOSFET to allow current to flow, which it is. The below filter proves that our LED turns on fully for low frequencies within the pass-band of 408Hz. Any higher frequency would begin to attenuate and the voltage at the source would be lower, meaning the current through the diode would decrease, meaning a less bright LED.

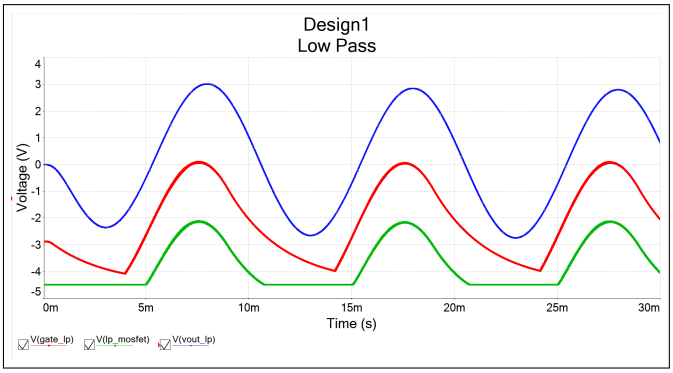


Fig. 16: Low Pass Filter Characteristics at Low Frequency

The below figure shows the output of the high pass filter as well as the voltage at the source of the MOSFET when the same low frequency as before is applied. This proves that the implemented system is working correctly, as the high pass filter should not let any low frequency signals through. Here, our 3V that is passed through the low pass filter has a very high attenuation when going through the high pass filter. This in turn does not create a voltage high enough to cross the threshold for the MOSFET after the peak detector, which means that current does not flow from the drain to the source, thus the LED will stay off. This can be seen in the figure at the red line, which represents gate voltage at the MOSFET. This line when properly combined does not cross the voltage threshold for the MOSFET, which in turn does not allow current to flow across the LED from the drain to the source. Overall, this figure proves that at low frequencies, the LED on the high frequency side of the circuit will not turn on.

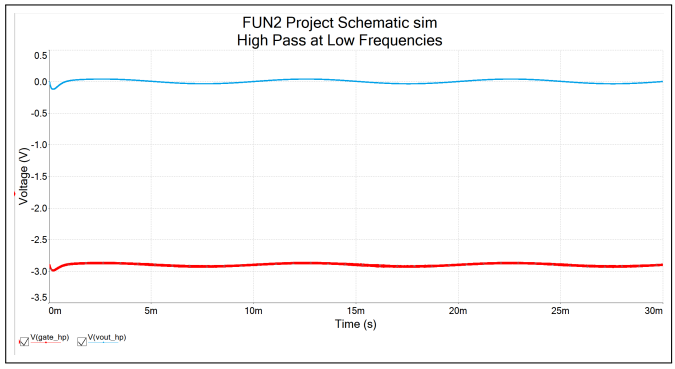


Fig. 17: High Pass Filter Characteristics at Low Frequency

Next, a 0.5V amplitude AC source operating at 2kHz was applied to each left and right channel input of the circuit. This frequency was chosen in simulation so that it was within the pass-band for the high pass filter (after 888Hz), allowing for minimal to 0 attenuation of the signal from the high pass filter. The summing amplifier should take each 0.5V source, combine them to make 1V, then produce a gain of 3V, making the final output to be 3V. This output at a frequency within the pass-band should be passed straight through the high pass filter, allowing the signal to be handled by the peak detector, thus lighting up the diode at this high frequency. The below simulation figure shows exactly this. The blue line is showing the simulated voltage measured at the output of the high pass filter, which is outputting teh 3V without attenuation as coming from the summing amplifier, indicating that the frequency is within the pass-band of the high pass filter. The red line measures the voltage at the gate of the MOSFET after the peak detector. This proves that our peak detector is working correctly, as the signal is only rising in voltage when the output of the high pass filter is surpassing it. When the output decreases, the time constant (previously calculated from the resistor and capacitor values) holds the voltage up, as this voltage decreases with the capacitor discharging (the capacitor is now holding up the voltage at the gate). This is seen from the unnatural decrease in voltage in between peaks. Our peak detector is also found to be working correctly, as this operating frequency is again 58.94Hz. The purpose of this is so that the naked eye will not be able to see the LED flickering when on. The peaks of the MOSFET gate voltage allow current to flow between the gate and the source, and at this High Frequency, the LED will essentially always be on (note 4.5 should be added to the green line to get the voltage at the source in relation to ground, as the resistor goes from the source to VEE, or -4.5V). This combined voltage needs to be above the threshold for the MOSFET to allow current to flow, which it is at this frequency. The below figure proves that our LED turns on fully for high frequencies after the corner frequency of 888Hz. Any lower frequencies would behin to attenuate and the voltage at the source would be lower, meaning the current through the diode would decrease and eventually turn off.

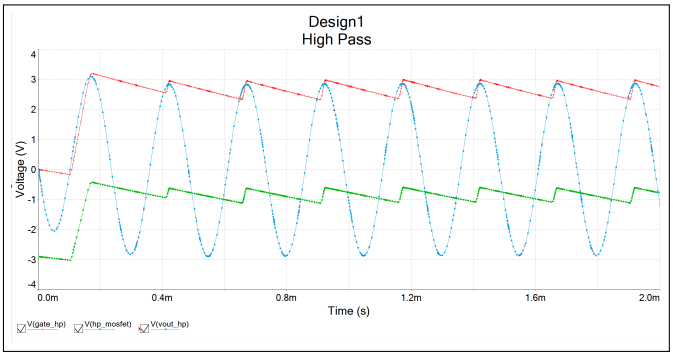


Fig. 18: High Pass Filter Characteristics at High Frequency

The below figure shows the output of the low pass filter as well as the voltages at the source and gate of the MOSFET after the peak detector on the low pass side of the circuit with the same 2kHz frequency applied to the circuit. This figure proves that the low pass filter has high attenuation when a high frequency input is applied. It can be seen that the output of the low pass filter does not look like the 3V amplitude that was input to it. Also, the voltage at the gate of the MOSFET is not high enough to allow current to pass through from the drain to the source, meaning the LED is not on. This can be seen at the green line, which is the voltage at the source of the MOSFET. This shows -4.5V, meaning there is no voltage across resistor R7, which leads to the conclusion that there is no current flowing through the diode. This figure overall proves that at high frequencies, the LED on the low frequency side of the circuit will not turn on.

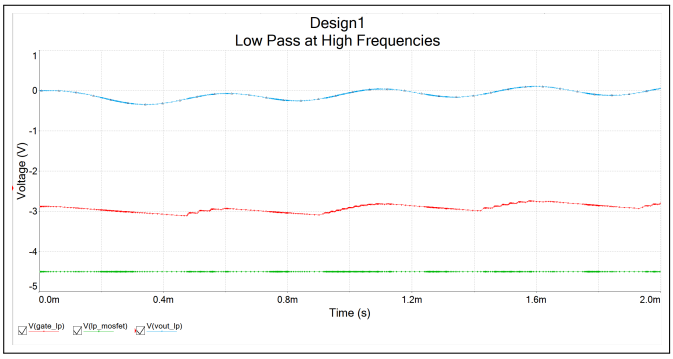


Fig. 19: Low Pass Filter Characteristics at High Frequency

IV. Experimental Results

A. Summing Amplifier Experimental Results

The network analyzer on Waveforms was conducted on both the left and right channels separately with a 0.5V source on each channel, which sums to a 1V input voltage, to plot the frequency response shown below from 2 Hz to 50 kHz:

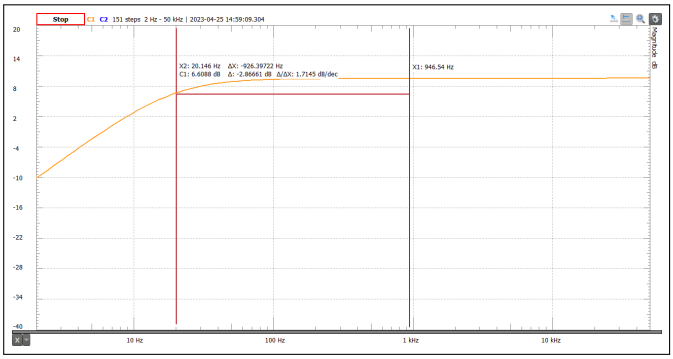


Fig. 20: Experimental Right Channel of the Summing Amplifier with a Gain of 3, a Corner Frequency of 20.1 Hz, and a -2.86 dB Drop

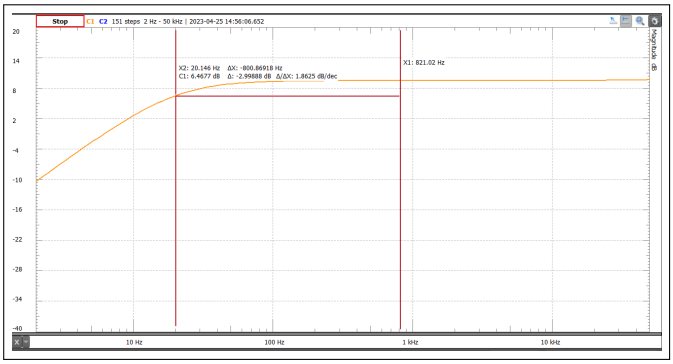


Fig. 21: Experimental Left Channel of the Summing Amplifier with a Gain of 3, a Corner Frequency of 20.1 Hz, and a -3 dB Drop

B. High Pass Filter Experimental Results

The network analyzer on Waveforms was conducted on the low pass Sallen-key filer with a 1V input voltage to plot the frequency response shown below from 2 Hz to 50 kHz:

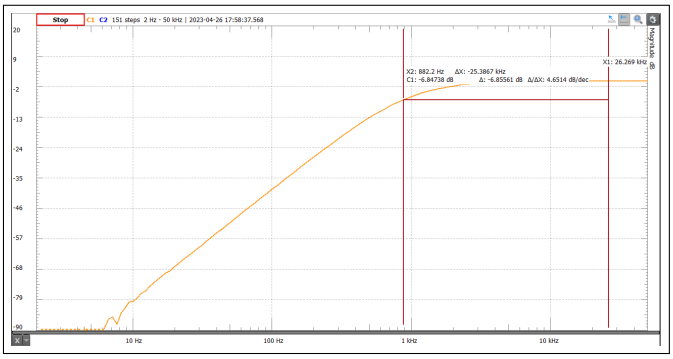


Fig. 22: Experimental Frequency Response of the High Pass Sallen-Key Filter with a corner frequency of 882 Hz and a -6.84 dB Drop

C. Low Pass Filter Experimental Results

The network analyzer on Waveforms was conducted on the low pass Sallen-key filter with a 1V input voltage to plot the frequency response shown below from 2 Hz to 50 kHz:

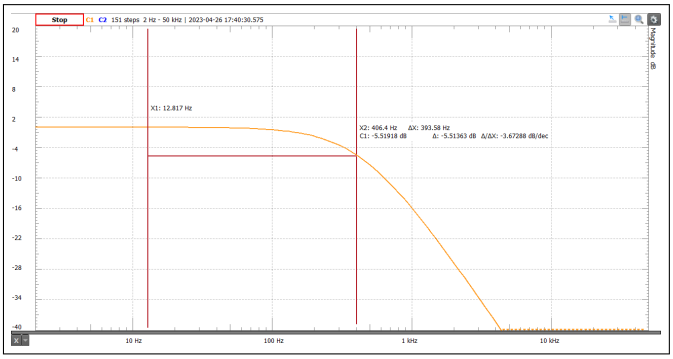


Fig. 23: Experimental Frequency Response of the Low Pass Sallen-Key Filter with a corner frequency of 406 Hz and a -5.51 dB Drop

D. Peak Detector Experimental Results

The oscilloscope on Waveforms was conducted on the low pass peak detector with a 9V battery and our input audio signal. The input voltage (blue) and the output of the peak detector (orange) was measured below.

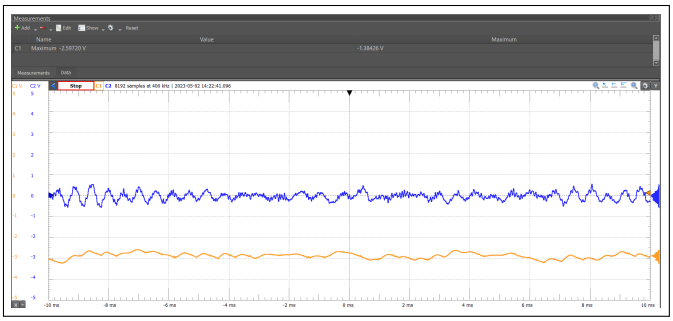


Fig. 24: Experimental Time Domain of the Low Pass Peak Detector with a maximum voltage of $-1.38 + 4.5 = 3.12V$

The oscilloscope on Waveforms was conducted on the high pass peak detector with a 9V battery and our input

audio signal. The input voltage (blue) and the output of the peak detector (orange) was measured below.

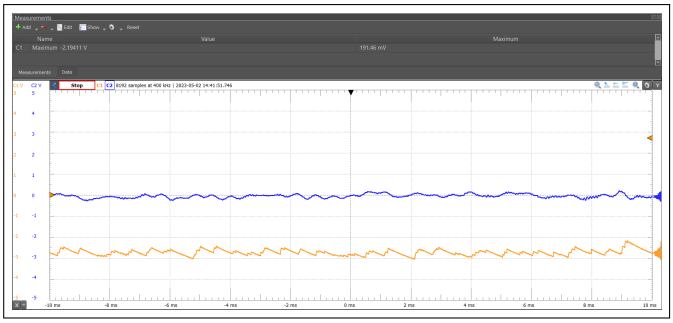


Fig. 25: Experimental Time Domain of the High Pass Peak Detector with a maximum voltage of $0.191 + 4.5 = 4.7V$

E. LED Driver Experimental Results

The oscilloscope on Waveforms was conducted on the low pass LED Driver with a 9V Duracell battery and our input audio signal. The output voltage (orange) measures the source voltage of the MOSFET. The maximum voltage throughout the song was taken and the voltage drop across the resistor was calculated. Using Ohm's law, the drain source current was calculated below.

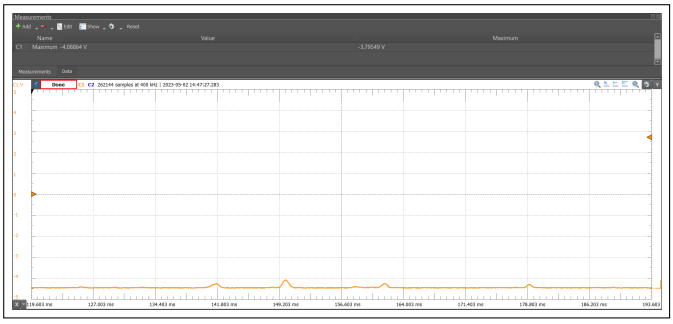


Fig. 26: Experimental Time Domain of the Low Pass LED Driver with a maximum voltage of -4.16V;
$-3.79 + 4.5 = 0.71V$; $I_{DS} = \frac{0.71}{270} = 2.6mA$

The oscilloscope on Waveforms was conducted on the high pass LED Driver with a 9V Duracell battery and our input audio signal. The output voltage (orange) measures the source voltage of the MOSFET. The maximum voltage throughout the song was taken and the voltage drop across the resistor was calculated. Using Ohm's law, the drain source current was calculated below.

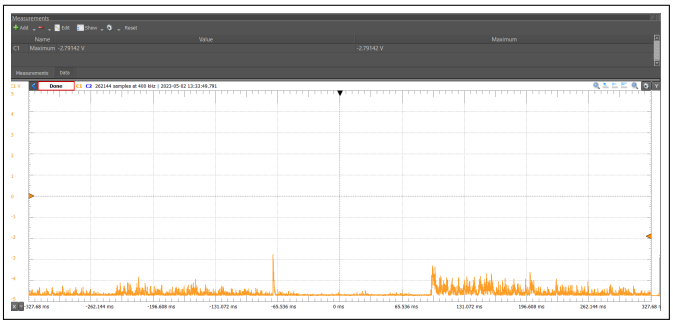


Fig. 27: Experimental Time Domain of the High Pass LED Driver with a maximum voltage of -2.79V;
$-2.79 + 4.5 = 1.71V$; $I_{DS} = \frac{1.71}{27} = 63mA$

F. System Level Experimental Results

Both figures below had a 9V battery with the audio signal as the input. Both inputs (blue) and outputs (orange) of the low pass and high pass sections of the system were measured and shown below. It can be observed that the peak detector is flattening out the peaks of both systems

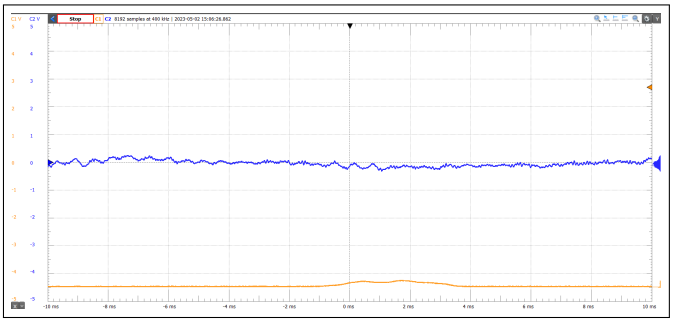


Fig. 28: Experimental Time Domain of the Low Pass Section of the System

G. System Level Experimental Results

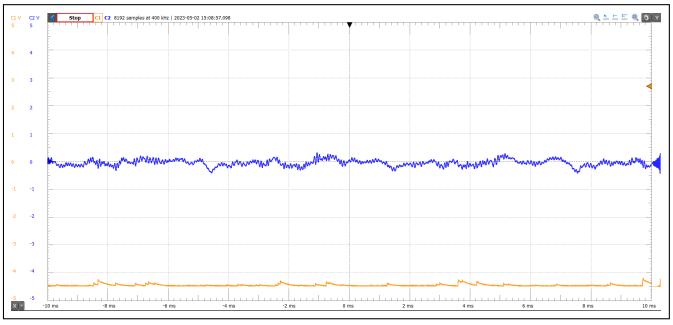


Fig. 29: Experimental Time Domain of the High Pass Section of the System

V. Challenges and Workarounds

Challenge 1: Our green LED stopped responding to the audio signal Solution: We tested the MOSFET and LED separately and together on the breadboard to find out the solution only to find out our MOSFET stopped functioning. We replaced the MOSFET.

Challenge 2: Gate voltage value was initially set near the calculated threshold voltage of the MOSFET, resulting in our LEDs to be dim rather than completely off when inputting solely a DC source. Solution: We recalculated the bias resistors multiple times and found a gate voltage much less than the calculated threshold voltage that allowed the LEDs to turn completely off when only a DC source was the input without any AC signal.

Challenge 3: Soldering the MOSFET resulted in a MOSFET to not work properly. Solution: We cleaned up the solder around the area and soldered the leads of the MOSFET to the LED.

VI. Conclusions and Thoughts for Future Classes

The process of making the project come to life was very enriching. As a team, we learned to solve a large scale problem together by breaking them up into smaller problems that were more feasible to handle. While there were many ups and downs we faced as a group when designing the project, our perseverance and dedication to make it come to life allowed us to simultaneously meet our objectives and complete the project. An emphasis on debugging and the understanding and analysis of each block was critical in the creation of this project, which is the main takeaway for all of us.

For future courses, we believe it would be important to allow for more project time in class throughout the semester in order to avoid last minute project work. Some advice to future students in approaching this project is to choose a song that doesn't have too much going on it in. Many mainstream songs have too much going on that makes it difficult to discern between high and low frequencies, such as too much low frequency bass or too much high frequency vocals. Additionally, we advise that students understand what each block of the system is doing and to always analytically and numerically test each component before soldering on the PCB.

VII. Collaboration Statement

Daniel Sarria - Analytical: Summing Amplifier, Peak Detector, LED Driver; Simulations: Summing Amplifier, Peak Detector (Choose component values), LED Driver (Choose component values); Experimental: Summing Amplifier, Peak Detector, LED Driver.

Joey Cohen - Background Information; Analytical: Low Pass Filter; Simulations: Low Pass Filter, Peak Detector (Build circuit on breadboard), LED Driver (Build circuit on breadboard); Experimental: Low Pass Filter; Challenges

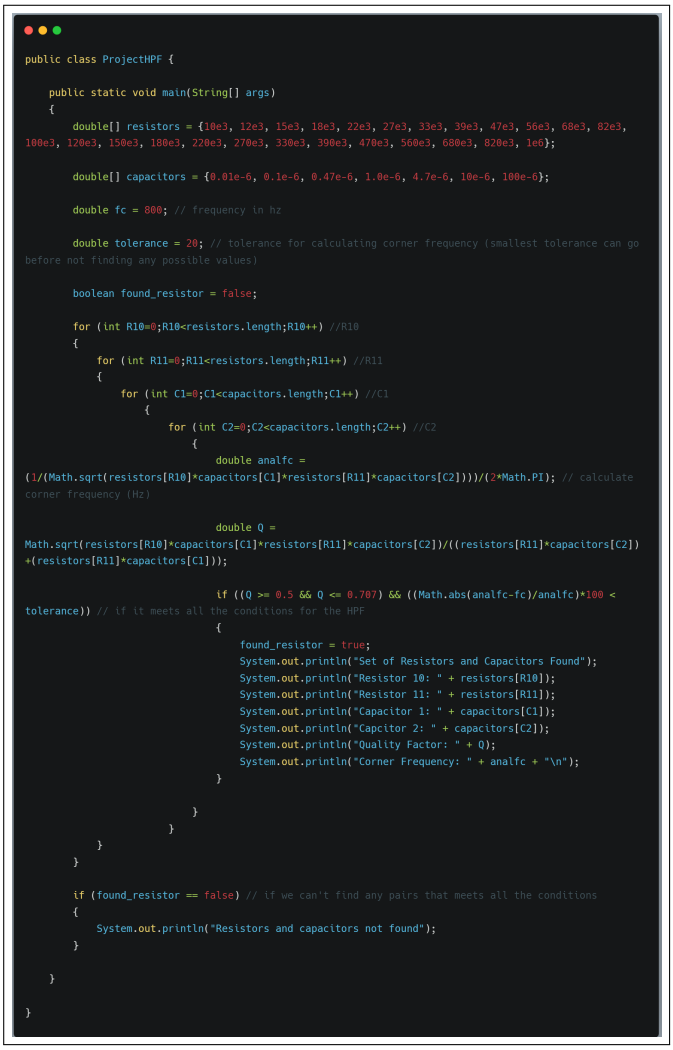
Andrew Vithoulkas - Abstract; Analytical: High Pass Filter; Simulations: High Pass Filter, Peak Detector (Build circuit on breadboard and choose component values), LED Driver (Build circuit on breadboard and choose component values); Experimental: High Pass Filter; Java Code;

Appendix

```java
public class ResistorDesign{

    public static void main(String[] args)
    {
        double[] resistors = {10, 12, 15, 18, 22, 27, 33, 39, 47, 56, 68, 82, 100, 120, 150, 180, 220,
270, 330, 390, 470, 560, 680, 820, 1e3, 1.2e3, 1.5e3, 1.8e3, 2.2e3, 2.7e3, 3.3e3, 3.9e3, 4.7e3, 5.6e3,
6.8e3, 8.2e3, 10e3, 12e3, 15e3, 18e3, 22e3, 27e3, 33e3, 39e3, 47e3, 56e3, 68e3, 82e3, 100e3, 120e3,
150e3, 180e3, 220e3, 270e3, 330e3, 390e3, 470e3, 560e3, 680e3, 820e3, 1e6}; // lab kit resistors

        double output_voltage = 6;

        double input_voltage = 2;

        double input_resistance = 82e3;

        double tolerance = 5e-3;

        boolean found_resistor = false;

        for (int R1=0;R1<resistors.length;R1++) //R1
        {
            for (int R2=0; R2<resistors.length;R2++) //R2
            {
                for (int R5=0; R5<resistors.length;R5++) //R5
                {
                    double feedback_resistor = resistors[R2]+resistors[R5]+
((resistors[R5]*resistors[R2])/resistors[R1]);

                    double output_voltage2 = 1*input_voltage*(feedback_resistor/input_resistance);

                    if (Math.abs(output_voltage2-output_voltage)/output_voltage2 < tolerance)
                    {
                        found_resistor = true;
                        System.out.println("Set of Resistors Found");
                        System.out.println("Resistor 1: " + resistors[R1]);
                        System.out.println("Resistor 2: " + resistors[R2]);
                        System.out.println("Resistor 5: " + resistors[R5]);
                        System.out.println("Output Voltage: " + Math.abs(output_voltage2));
                        System.out.println("Tolerance: " + Math.abs(output_voltage2-
output_voltage)/output_voltage2+"<"+tolerance +"\n");
                    }
                }
            }
        }
        if (found_resistor == false)
        {
            System.out.println("Resistors not found");
        }

    }
}
```

Fig. 30: Java Program to Calculate T-Network Resistors

The code above utilizes a double nested for loop to iterate through every possible combination of resistors in our lab kit. It then calculates the value of the feedback resistor and the output voltage. If the output voltage met the tolerance value conditions, it would output the feedback resistor value and the output voltage it calculated. The tolerance was set to be as close as possible to the output voltage.



```java
public class ProjectHPF {

    public static void main(String[] args)
    {
        double[] resistors = {10e3, 12e3, 15e3, 18e3, 22e3, 27e3, 33e3, 39e3, 47e3, 56e3, 68e3, 82e3,
100e3, 120e3, 150e3, 180e3, 220e3, 270e3, 330e3, 390e3, 470e3, 560e3, 680e3, 820e3, 1e6};

        double[] capacitors = {0.01e-6, 0.1e-6, 0.47e-6, 1.0e-6, 4.7e-6, 10e-6, 100e-6};

        double fc = 800; // frequency in hz

        double tolerance = 20; // tolerance for calculating corner frequency (smallest tolerance can go
before not finding any possible values)

        boolean found_resistor = false;

        for (int R10=0;R10<resistors.length;R10++) //R10
        {
            for (int R11=0;R11<resistors.length;R11++) //R11
            {
                for (int C1=0;C1<capacitors.length;C1++) //C1
                    {
                        for (int C2=0;C2<capacitors.length;C2++) //C2
                            {
                                double analfc =
(1/(Math.sqrt(resistors[R10]*capacitors[C1]*resistors[R11]*capacitors[C2])))/(2*Math.PI); // calculate
corner frequency (Hz)

                                double Q =
Math.sqrt(resistors[R10]*capacitors[C1]*resistors[R11]*capacitors[C2])/((resistors[R11]*capacitors[C2])
+(resistors[R11]*capacitors[C1]));

                                if ((Q >= 0.5 && Q <= 0.707) && ((Math.abs(analfc-fc)/analfc)*100 <
tolerance)) // if it meets all the conditions for the HPF
                                {
                                    found_resistor = true;
                                    System.out.println("Set of Resistors and Capacitors Found");
                                    System.out.println("Resistor 10: " + resistors[R10]);
                                    System.out.println("Resistor 11: " + resistors[R11]);
                                    System.out.println("Capacitor 1: " + capacitors[C1]);
                                    System.out.println("Capcitor 2: " + capacitors[C2]);
                                    System.out.println("Quality Factor: " + Q);
                                    System.out.println("Corner Frequency: " + analfc + "\n");
                                }

                            }
                    }
            }
        }

        if (found_resistor == false) // if we can't find any pairs that meets all the conditions
        {
            System.out.println("Resistors and capacitors not found");
        }

    }

}
```

Fig. 31: Java Program to Calculate Resistor and Capacitor Values for Sallen-Key High Pass Filter

```java
public class ProjectLPF {

    public static void main(String[] args)
    {
        double[] resistors = {10e3, 12e3, 15e3, 18e3, 22e3, 27e3, 33e3, 39e3, 47e3, 56e3, 68e3, 82e3,
100e3, 120e3, 150e3, 180e3, 220e3, 270e3, 330e3, 390e3, 470e3, 560e3, 680e3, 820e3, 1e6};

        double[] capacitors = {0.01e-6, 0.1e-6, 0.47e-6, 1.0e-6, 4.7e-6, 10e-6, 100e-6};

        double fc = 400; // frequency in hz

        double tolerance = 5; // tolerance for calculating corner frequency (smallest tolerance can go
before not finding any possible values)

        boolean found_resistor = false;

        for (int R3=0;R3<resistors.length;R3++) //R3
        {
            for (int R4=0;R4<resistors.length;R4++) //R4
            {
                for (int C1=0;C1<capacitors.length;C1++) //C1
                    {
                        for (int C2=0;C2<capacitors.length;C2++) //C2
                        {
                            double analfc =
(1/(Math.sqrt(resistors[R3]*capacitors[C1]*resistors[R4]*capacitors[C2])))/(2*Math.PI); // calculate
corner frequency

                            double Q =
Math.sqrt(resistors[R3]*capacitors[C1]*resistors[R4]*capacitors[C2])/((resistors[R3]*capacitors[C1])+
(resistors[R4]*capacitors[C1]));

                            if ((Q >= 0.5 && Q <= 0.707) && ((Math.abs(analfc-fc)/analfc)*100 <
tolerance)) // if it meets all the conditions for the LPF
                                {
                                    found_resistor = true;
                                    System.out.println("Set of Resistors and Capacitors Found");
                                    System.out.println("Resistor 3: " + resistors[R3]);
                                    System.out.println("Resistor 4: " + resistors[R4]);
                                    System.out.println("Capacitor 1: " + capacitors[C1]);
                                    System.out.println("Capacitor 2: " + capacitors[C2]);
                                    System.out.println("Quality Factor: " + Q);
                                    System.out.println("Corner Frequency: " + analfc + "\n");
                                }

                            }
                        }


            }
        }
        if (found_resistor == false) // if we can't find any pairs that meets all the conditions
        {
            System.out.println("Resistors and capacitors not found");
        }

    }

}
```

Fig. 32: Java Program to Calculate Resistor and Capacitor Values for Sallen-Key Low Pass Filter

The code above utilizes a double nested for loop to iterate through every possible combination of resistors and capacitors in our lab kit. It then calculates the value of the feedback resistor and the output voltage. If the corner frequency meets the tolerance value conditions and meets the conditions of the Q values, it would output the set of resistor values and capacitor values it calculated those results. The tolerance was set to be as close as possible to meet the requirements of the corner frequency.

The code above utilizes a double nested for loop to iterate through every possible combination of resistors and capacitors in our lab kit. It then calculates the value of the feedback resistor and the output voltage. If the corner frequency meets the tolerance value conditions and meets the conditions of the Q values, it would output the set of resistor values and capacitor values it calculated those results. The tolerance was set to be as close as possible to meet the requirements of the corner frequency.

```java
public class BiasResistors
{
    public static void main(String[] args)
    {
        double[] resistors = {10, 12, 15, 18, 22, 27, 33, 39, 47, 56, 68, 82, 100, 120, 150, 180, 220,
270, 330, 390, 470, 560, 680, 820, 1e3, 1.2e3, 1.5e3, 1.8e3, 2.2e3, 2.7e3, 3.3e3, 3.9e3, 4.7e3, 5.6e3,
6.8e3, 8.2e3, 10e3, 12e3, 15e3, 18e3, 22e3, 27e3, 33e3, 39e3, 47e3, 56e3, 68e3, 82e3, 100e3, 120e3,
150e3, 180e3, 220e3, 270e3, 330e3, 390e3, 470e3, 560e3, 680e3, 820e3, 1e6};

        double Vg = 2.05; // aka vt

        double Vcc = 9;

        double tolerance = 50; // subject to change

        boolean found_resistor = false;

        for (int R14 = 0; R14 < resistors.length; R14++) // r14
        {
            for (int R15 = 0; R15 < resistors.length; R15++) // r15
            {
                double analvg = (resistors[R15]*Vcc)/(resistors[R14]+resistors[R15]);

                if ((Math.abs(analvg-Vg)/analvg)*100 < tolerance && analvg < 1.7)
                {
                    found_resistor = true;
                    System.out.println("Pair of Resistors Found");
                    System.out.println("Resistor 14: " + resistors[R14]);
                    System.out.println("Resistor 15: " + resistors[R15]);
                    System.out.println("Gate Voltage: " + analvg + "V" + "\n");
                }
            }
        }
        if (found_resistor == false)
        {
            System.out.println("No Pairs of Resistors Found");
        }
    }
}
```

Fig. 33: Java Program to Calculate Bias Resistor Values for LED Drivers

The code above utilizes a nested for loop to iterate through every possible combination of resistors in our lab kit. It then calculates the gate voltage using a voltage divider. If the gate voltage meets the tolerance value conditions, it would output the set of resistor values that calculated those results. The tolerance was set to be particularly high in order to find varying values of gate voltages.

REFERENCES

[1] *Tlhr440., tlho440., tlhy440., tlhg440.* TLHG4400-MS12, Vishay, 2022. [Online]. Available: https://www.digikey.com/en/products/detail/vishay-semiconductor-opto-division/TLHG4400-MS12/6594865?s=N4IgTCBcDaIC4BsAWBzALGgDJgtAWwGcBGMAYzhwDsATEAXQF8g.

[2] *Hlmp-4700, hlmp-4719, hlmp-4740, hlmp-1700, hlmp-1719, hlmp-1790*, HLMP-1700, BROADCOM, 2021. [Online]. Available: https://www.digikey.com/en/products/detail/broadcom-limited/HLMP-1700-B0002/1234784.