Goal

The goal of my project was to create a circuit that will allow LEDs to react to certain frequencies bands that are common in music without the use of an arduino. This was achieved through building a circuit with 5 different stages: sound detection, band-pass filters, Schmitt Triggers, signal inversion, and reactive LEDs. I also wanted to make this very modular so that it can possibly be used with an arduino in order to make an audio visualizer with LEDs.

The Entire Circuit

As stated earlier, I split up the entire project into 5 different stages (or circuits) in order to make it modular and also easy to troubleshoot throughout the entire building process. Each stage will have separate subsections in which I will go into greater detail about the design and also show a circuit diagram for them. The way that this works is that I first get an audio signal from a sound sensor, that audio signal is then sent through a bus line. From the bus line, I have 4 different band-pass filters connected, which correspond to the 4 main frequencies that are commonly used in music: 60-250Hz (bass), 250-500Hz (lower midrange), 500Hz-2kHz (midrange), and 2-4kHz(upper midrange). Once the signal passes through the filters, they go through a Schmitt Trigger to be converted into a logic signal, which makes it safe to use with the LED. The Schmitt Trigger inverts the signal though so it must go through an inverter, which I made using the 74HC00 NAND gate.

The Sound Detector

This circuit is pretty simple and only has the sound detector connected to $+5V$ and GND. The audio output was sent to a bus line on the proto board so that it can be used by different band-pass filters. The only issue that this circuit had was that the sound detector was not sensitive enough to pick up music, and the output voltage for the audio signal was very low ($0.4V$). This was fixed by removing the resistor labeled $R3$ on the board and placing a $1M\Omega$ resistor on the $R7$ label. This increased the gain of the board, which in turn made it really sensitive, and also significantly increased the output voltage to about $2.02V$.

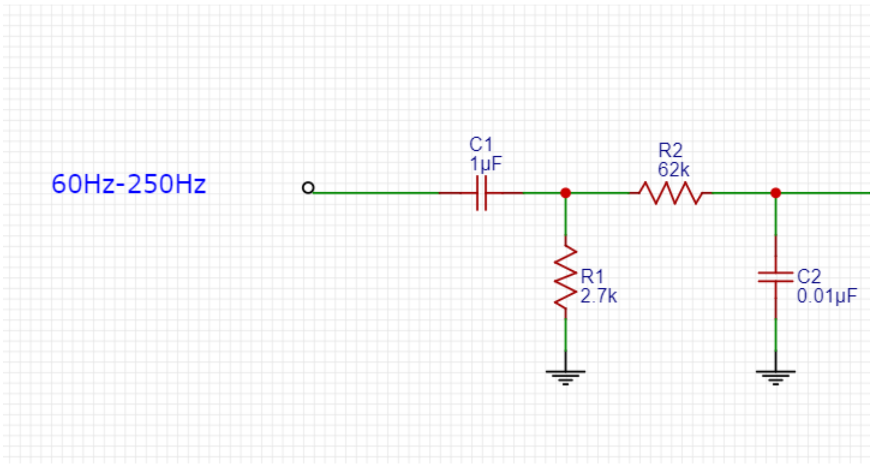
The Band-Pass Filters

The is made up of 4 different Band-Pass Filters. As stated earlier, this stage filters the audio signal into 4 different frequency bands found within the audio spectrum found in music. The original design for this stage comprised of using a $1M\Omega$ resistor and then determining which capacitors to use by using the equation:

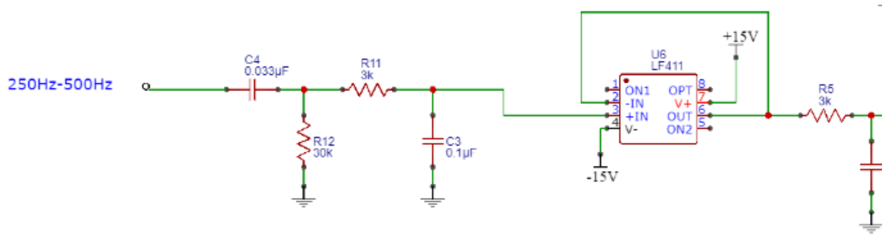
$$C = \frac{1}{2\pi R f}$$

Keeping the resistors the same made building the circuit easy, but it did lead to a lot of problems. The first problem was that the signal had a large voltage drop after going through the band-pass filter. I fixed this by using higher capacitance within the filters and finding the appropriate resistance by using the equation

$$R = \frac{1}{2\pi C f}$$



This is the final design for one of the band-pass filters. There was an issue with the $250-500Hz$ band-pass filter which had it be sensitive to frequencies way past the $500Hz$ range. I fixed this issue by cascading the low pass filter by using a follower circuit. I used the follower circuit in order to use the same resistor and capacitor which made the design easier to build.



These filters were designed with the output voltage of the sound detector in mind. On average, the output voltage would be about $5.2V$ peak to peak which is roughly about $2.6V$. Each band-pass filter was designed to have at least $0.5V$ as its output voltage near the lower and upper extremes of each filters.

The following are tables showing the values of V_{out} for each filter which I used to determine an appropriate threshold voltage for the Schmitt Triggers.

60Hz-250Hz				
Frequency	Vin	Vout	Vout/Vin	Vout Pk-Pk
40	2.08	1.16	0.557692308	2.32
50	2.08	1.32	0.634615385	2.64
60	2.08	1.44	0.692307692	2.88
70	2.08	1.48	0.711538462	2.96
80	2.08	1.56	0.75	3.12
90	2.08	1.56	0.75	3.12
100	2.08	1.6	0.769230769	3.2
110	2.08	1.6	0.769230769	3.2
120	2.08	1.6	0.769230769	3.2
130	2.08	1.6	0.769230769	3.2
140	2.08	1.6	0.769230769	3.2
150	2.08	1.6	0.769230769	3.2
160	2.08	1.56	0.75	3.12
170	2.08	1.56	0.75	3.12
180	2.08	1.56	0.75	3.12
190	2.08	1.52	0.730769231	3.04
200	2.08	1.52	0.730769231	3.04
330	2.08	1.24	0.596153846	2.48
400	2.08	1.08	0.519230769	2.16
500	2.08	0.92	0.442307692	1.84
600	2.08	0.84	0.403846154	1.68

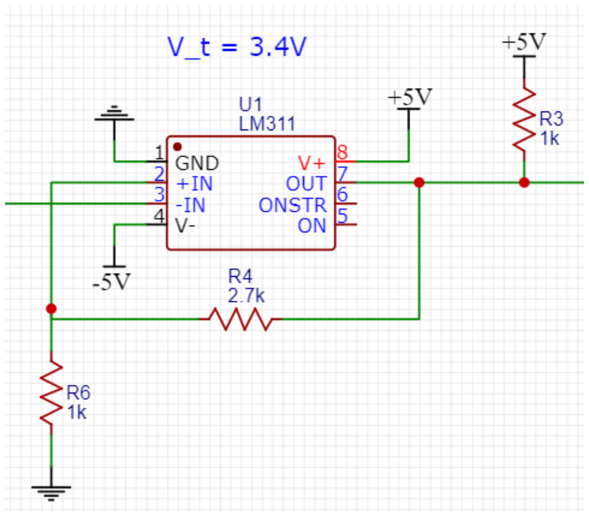
250Hz-500Hz				
Frequency	Vin	Vout	Vout/Vin	Vout Pk-Pk
50	2.04	0.4	0.196078431	0.8
100	2.04	0.76	0.37254902	1.52
200	2.04	1.12	0.549019608	2.24
300	2.04	1.2	0.588235294	2.4
400	2.04	1.2	0.588235294	2.4
500	2.04	1.16	0.568627451	2.32
600	2.04	1.12	0.549019608	2.24
700	2.04	1.04	0.509803922	2.08
800	2.04	1	0.490196078	2

500Hz-2kHz				
Frequency	Vin	Vout	Vout/Vin	Vout Pk-Pk
200	2.04	0.52	0.254901961	1.04
300	2.04	0.64	0.31372549	1.28
400	2.04	0.72	0.352941176	1.44
500	2.04	0.76	0.37254902	1.52
600	2.04	0.76	0.37254902	1.52
700	2.04	0.8	0.392156863	1.6
800	2.04	0.76	0.37254902	1.52
900	2.04	0.8	0.392156863	1.6
1000	2.04	0.76	0.37254902	1.52
1100	2.04	0.8	0.392156863	1.6
1200	2.04	0.8	0.392156863	1.6
1300	2.04	0.76	0.37254902	1.52
1400	2.04	0.76	0.37254902	1.52
1500	2.04	0.76	0.37254902	1.52
1600	2.04	0.76	0.37254902	1.52
1700	2.04	0.76	0.37254902	1.52
1800	2.04	0.76	0.37254902	1.52
1900	2.04	0.76	0.37254902	1.52
2000	2.04	0.76	0.37254902	1.52

2kHz-4kHz				
Frequency	Vin	Vout	Vout/Vin	Vout Pk-Pk
2000	2.12	1.36	0.641509434	2.72
3000	2.12	1.44	0.679245283	2.88
4000	2.12	1.36	0.641509434	2.72
5000	2.12	1.28	0.603773585	2.56
6000	2.12	1.2	0.566037736	2.4
7000	2.12	1.08	0.509433962	2.16

The Schmitt Triggers

I used the Scmitt Triggers in order to turn the audio signal into a logic signal. I also used this in order to filter out any noise that isn't music since the sound detector also reacts to background noise. Since each band-pass filter had different output voltages, they each had to have their own design for the Scmitt Trigger voltage. This means that each Schmitt Trigger has a different threshold voltage which will be shown on the full circuit diagram.



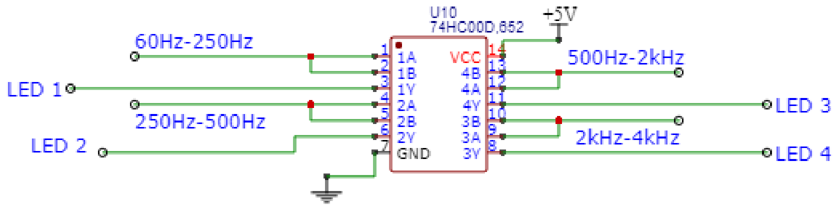
The design is pretty much identical from the one I showed on the proposal, the only difference is that $R2$ is different in order to compensate for the lower output voltage of the sound detector. I used the following equation in order to determine the value for $R2$ to get an appropriate threshold voltage.

$$R_2 = (\frac{V_+}{V_{th}} - 1)R_1$$

Here, V_+ is $5V$ but this can be changed if you want a higher output voltage. I set R_1 to be $1k\Omega$ for each trigger just to make the calculations easier.

The Inverter and LEDs

This isn't a really interesting circuit, I only used a 74HC00 NAND gate to create an inverter. The reasoning behind using the 74HC00 is because this is a CMOS IC, which means that the output voltage will either be 0 volts or 5 volts.



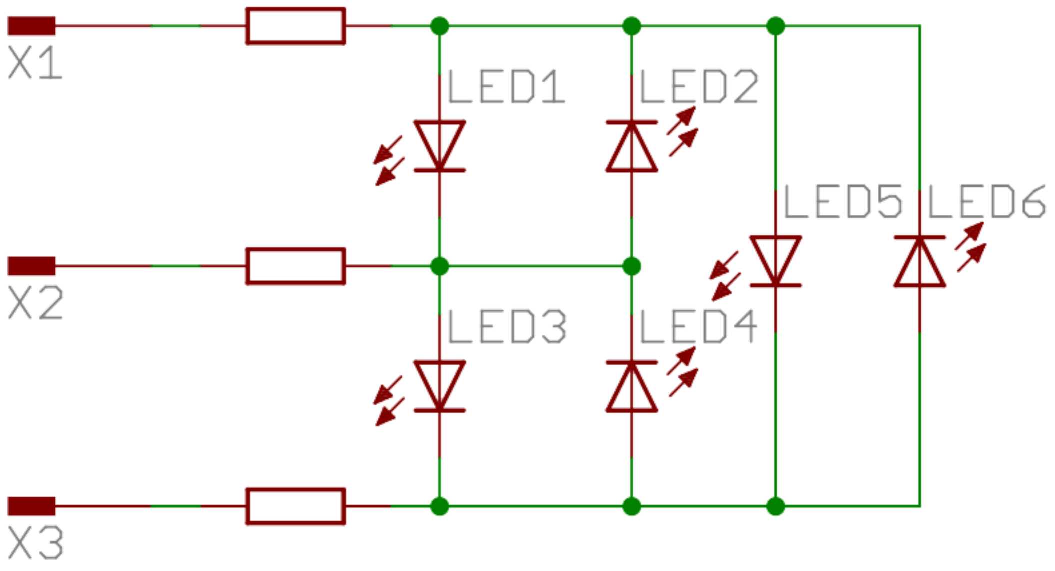
The LEDs are also just connected in series with a 330Ω resistor which isn't really all that exciting.

The Full Circuit and Final Remarks

The circuit works fine with the sound detector but some of the band-pass filters are still sensitive to frequencies that are slightly above its upper range. This can be fixed by either using more cascading filters or by changing the threshold voltage in the Schmitt Triggers. The circuit can also technically be used with an arduino in order to make a better audio visualizer with LEDs. I did some research and found that you can use a technique called Charlieplexing which allows the arduino to control multiple LEDs with a couple of pins. In fact, the amount of LEDs you can control is given by the equation:

$$LEDs = n^2 - n$$

In this equation, n is the number of pins used, meaning that 20 LEDs can be controlled with 5 pins and so on... The only issue with this is that the coding is highly complex since the LEDs would act as a logic gate. I have uploaded a test of the code on my course file. The following circuit diagram is for 6 LEDs and I found this on https://www.keychainino.com/how-to-drive-many-leds-with-only-few-pins-charlieplexing/.



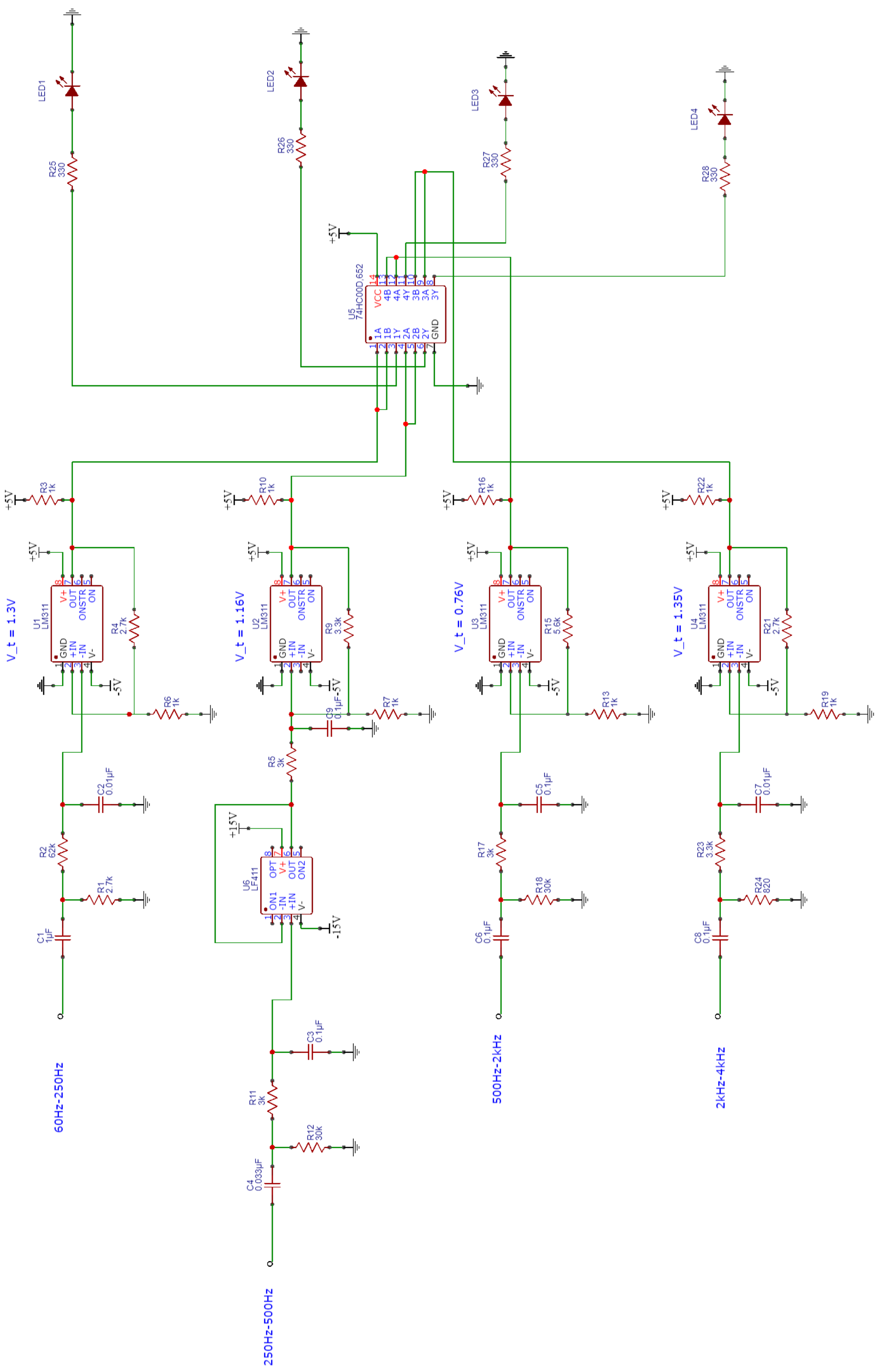
While I tested the circuit using 3 pins and manged to get the circuit to actually work with the arduino. During that, I was trying to find out what the logic is, but I only managed to get a truth table:

A	**B**	**C**	**LED#**
1	0	INPUT	1
0	1	INPUT	2
INPUT	0	1	3
INPUT	1	0	4
1	INPUT	0	5
0	INPUT	1	6

In here, INPUT is just the pin not doing anything, setting it to OUTPUT and LOW would actually result in different LEDs lighting up which isn't really useful. If anything, this kinda reminds me of a flip-flop but this circuit does not really show exactly the same behavior. Another thing to note is that I used A, B, C to represent the pins and the diagram that I got from online uses $X1, X2, X3$ to represent the pins.

It would have definitely been cool to use this, but I could not find a way to simplify the code to make the visualizer faster to respond to the music.

Apart from that, the circuit works very well and passed the Mariah Carey test (checking to see if it responds to All I want For Christmas). I put the full circuit in the next page, I did not include the sound detector as part of the circuit since it isn't really doing much, just know that the circles are meant to symbolize the audio signal coming from the sound detector.


LED1
LED2
LED3
LED4
R25 330
R26 330
R27 330
R28 330
U5 74HC00D,652
+5V
V_t = 1.3V
V_t = 1.16V
V_t = 0.76V
V_t = 1.35V
U1 LM311
U2 LM311
U3 LM311
U4 LM311
R3 1k
R10 1k
R16 1k
R22 1k
R4 2.7k
R9 3.3k
R15 5.6k
R21 2.7k
R6 1k
R7 1k
R13 1k
R19 1k
-5V
C9 0.1µF
R5 3k
+15V
-15V
U6 LF411
C2 0.01µF
R2 62k
R1 2.7k
C1 1µF
C5 0.1µF
R17 3k
R18 30k
C6 0.1µF
C7 0.01µF
R23 3.3k
R24 820
C8 0.1µF
C3 0.1µF
R11 3k
R12 30k
C4 0.033µF
60Hz-250Hz
250Hz-500Hz
500Hz-2kHz
2kHz-4kHz