ECE 445
Spring 2013

Music Response Light Show

Design Review
Andy Groesch
TA: Lydia Majure
February 25, 2013

Table of Contents

1.0 Introduction

1.1 Project Summary

The intent of this project is to make a lighting system that responds to music that is heard through the air. The microphone will detect the sonic signal and convert it into an analog signal. Then, the signal will be filtered based on the low, middle, and high frequency components. These signals will then be passed through a beat detection algorithm that will then pass data to a series of LED Driver Shields. These shields will control four strands of LEDs each, for a total of twelve LED arrangements.

1.2 Objectives

1.2.1 Goals:

- Split audio signal into three frequency bands
- Uses Beat Detection algorithm to identify beats in audio
- LED strands strobe based on output signal from

1.2.2 Functions

- Separates frequencies based on cutoff values determined from common instrument ranges
- LEDS strobe operate based on pressure fluctuations
- Different combinations of LEDs turn on based beat detection ouputs

1.2.3 Benefits

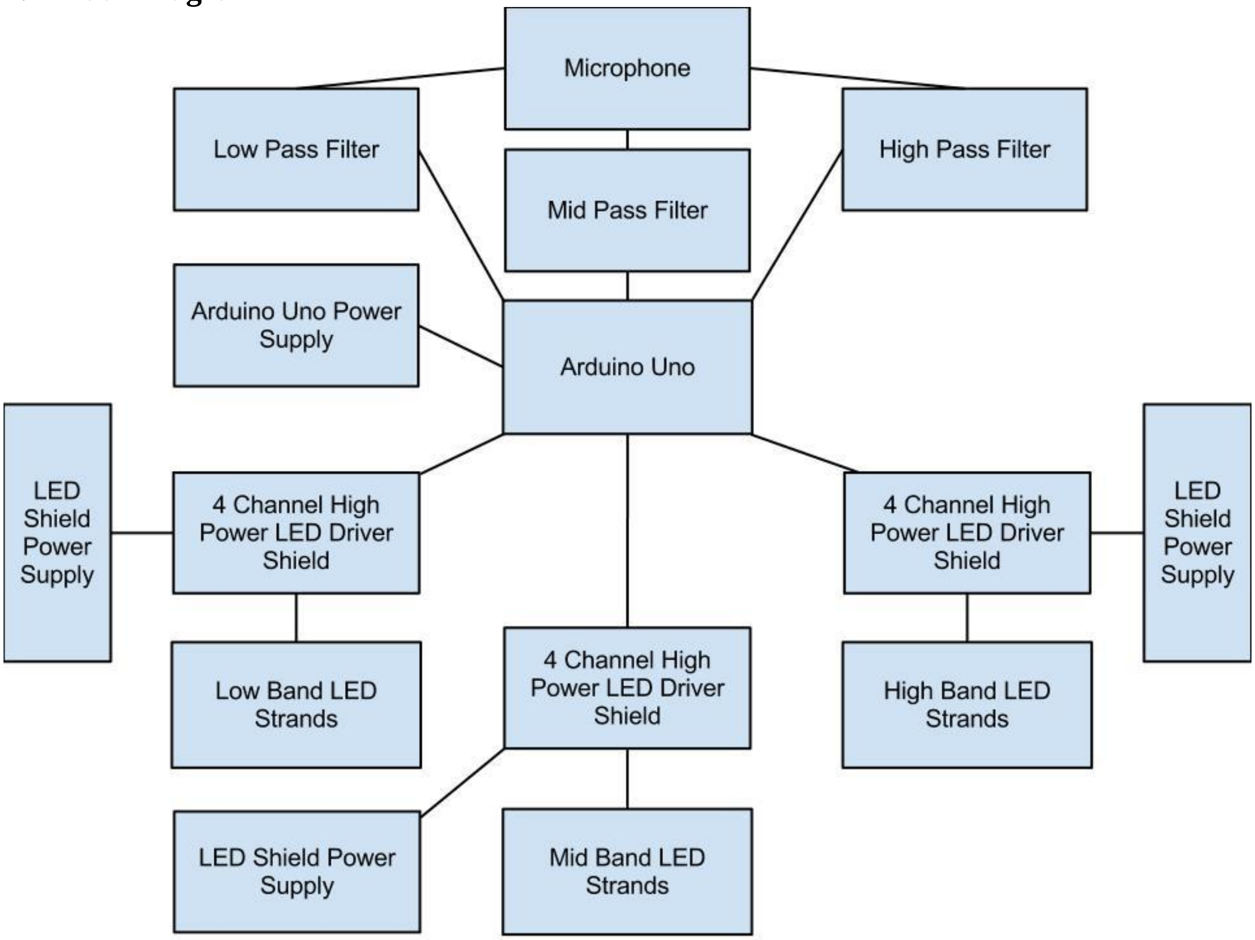
- Provides visual component to music entertainment
- Helps to understand concepts of harmonic frequency and electronic music
- Easily replicable

1.2.4 Features

- Autonomous light control; doesn’t require user input
- LEDs strobe to the beat of each of the frequency bands
- Automatically enables combinations of lights based on different frequency spectra

2.0 Design

2.1 Block Diagram



2.2 Block Description Summary

The microphone detects an audio signal from the air and converts it into an analog one. Then, a DC offset is added such that the signal is entirely positive. This signal is then sent through three filters (low pass, band pass, and high pass) such that these frequency bands are extracted. Then, these signals are sent into the through a beat detection algorithm. The beat detection algorithm then outputs a digital signal that is sent to the 4 Channel LED Driver that indicates when the LEDs should oscillate between On and Off states.

2.2.1 Microphone

This is used to detect an audio signal and convert it into an analog sinusoid representing the pressure fluctuations. The ADMP504 microphone is used in this design. It possesses maximum output voltage amplitude of .25 V; the microcontroller accepts voltage inputs ranging from 0 to 5 volts, and as such the output voltage of the microphone needs to be amplified with a gain of 10. The V_{DD} range is from 1.6 V to 3.3 V and is powered by the Arduino Uno.

2.2.2 Low Pass Active Filter

This circuit only allows low frequencies to pass as it attenuates the larger ones. This allows some of the lights to be mapped only to the lower frequencies. This typically includes the bass band and instruments like the kick drum, toms, bass, and tuba. The cutoff point of 160 Hz was chosen to ideally exclude the singer's voice.

2.2.3 High Pass Active Filter

This circuit only allows high frequencies to pass as it attenuates the smaller ones. This allows some of the lights to be mapped only to these higher frequencies. The cutoff frequency was chosen to be 1000 Hz

2.2.4 Band Pass Filter

This circuit allows frequencies between a desired range of cutoff frequencies to pass. For this design, the cutoff frequencies of 60 Hz and 1000 Hz were chosen.

2.2.5 Arduino Uno Microcontroller

The microcontroller is powered by the Arduino Uno Power Supply. It accepts an analog audio input from the microphone circuit, and detects voltage input ranges from 0 to 5 Volts. The microcontroller first separates the signal into three different frequency ranges by the way of a low pass, band pass, and high pass filter. These three signals are then analyzed by the beat detection algorithm on the data. The output data is then dispatched to the 4 Channel High Power LED Driver.

2.2.6 Arduino Uno Power Supply

The power supply delivers 9V of DC voltage with a 650 mA current to the Arduino Microcontroller.

2.2.7 4 Channel High Power LED Driver Shield

This shield accepts any voltage range from 8 to 27 Volts. A 24 Volt power supply was chosen o power the Driver Shield. This value was chosen because the amount of LEDs is related to the applied voltage; thus, as the applied voltage increases, more LEDs can be powered by the Driver Shield. Each shield will have four LED strands connected to its output terminals.

2.2.8 24V Power Supply

This device was chosen to power the 4 Channel High Power LED Driver Shield. It is capable of a maximum output current of 1.1 A corresponding to a maximum power output of 26.4 W.

2.2.9 LED Strands

Four strands of LEDs will be affixed to each of the Driver Shields, totaling 12 strands total. These LED strands will be mounted on a 42” tomato cage, with the low band LED string on the lowest rung, the mid band LED string on the middle rung, and the band LED string on the highest rung.

2.2.10 Beat Detection Algorithm

From Frederic Patin, “Beat Detection Algorithms”

The algorithm detects beats by first computing the average sound energy, comparing it to the instantaneous sound energy, and indicating a detected beat once the instantaneous energy exceeds the average energy.

The electret microphone can detect frequencies up to 2000 Hz; therefore, the minimum sampling rate (by the Nyquist criterion) is equal to (2*2,000) = 4kHz

First, the algorithm performs the Fast Fourier Transform on the input signal ‘a(t)’ to get A[f]. To obtain the energy, these values are squared and stored in a buffer named B containing 256 data entry positions. Then, the buffer is divided into 16 sub-bands of 16 values. The instantaneous energy is calculated by summating the 16 different energy values that were sampled. Thus, the value of the instantaneous energy in each subband ‘I’ will be calculated by the following formula:

$$Es[i] = 16/256*Sum(B[k]) \text{ from } k = i*16 \text{ to } k = (i+1)*16$$

Following this is an ‘energy history buffer’ named Ei, which is a data storage element holding the previous sixteen values for instantaneous energy in each subband ‘i’. The ‘average energy’ is calculated by the following formula:

$$<E>=1/16*Sum(Ei[k]) \text{ from } k=0 \text{ to } k =15$$

Then, the sound energy history buffer Ei is shifted one index to the right to make room for new data; the oldest data is pushed out of the buffer. Then, a new energy value Es[i] is entered into the first spot of the buffer at place Ei[0].

Then, the model computes the Variance of the Energies in order to compare the instantaneous energy value to the average energy value:

$$V(Ei)=1/16*Sum(Ei[k] - <Ei>)^2 \text{ for } k = 0 \text{ to } k = 15$$

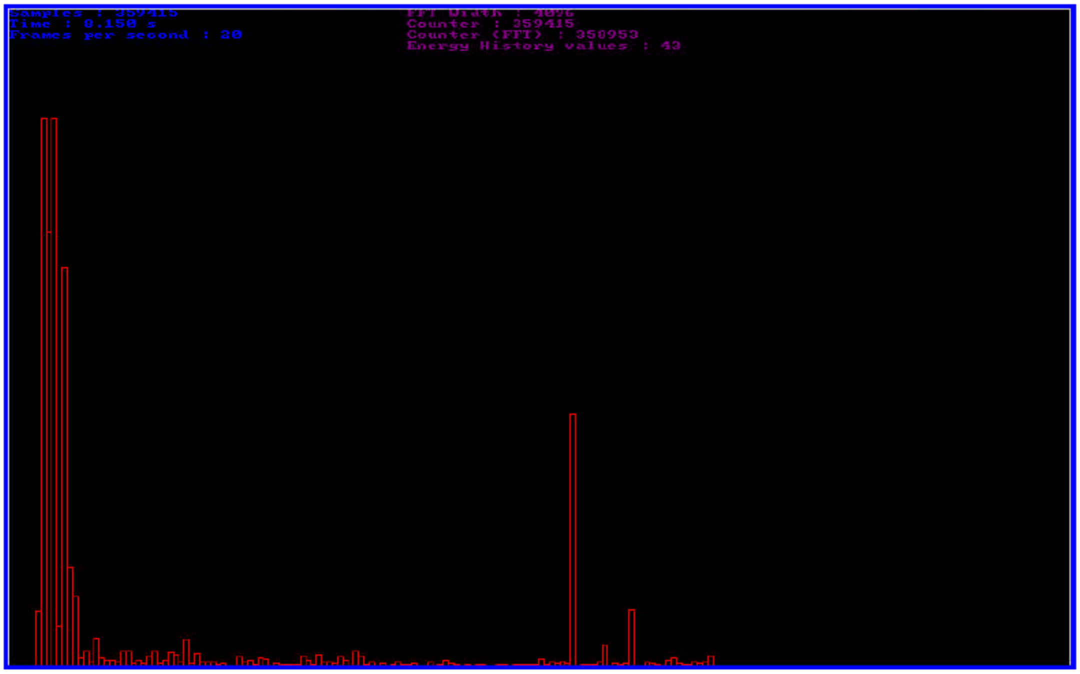
Then, compute the ‘C’ constant for comparison with the instantaneous energy value
C = (-.0025714*V)+1.5142857

This new energy value at Ei[0] is compared to the average energy from the past 16 iterations times the constant C to determine if a beat was detected.

If E_i [0]> C*<Ei>
then a beat is detected

If E_i [0]< C*<Ei>
then a beat is not detected

Histogram of Instantaneous Energy



From: Bass Detection Algorithms, page 13

Sound Energy Value Calculations

Samples : 641655
Time : 14.550 s
Frames per second : 20
FFT Width : 4096
Counter : 641655
Counter (FFT) : 640988
Energy History values : 43

Number of subbands : 128
Beat Detect Test Rate : 43

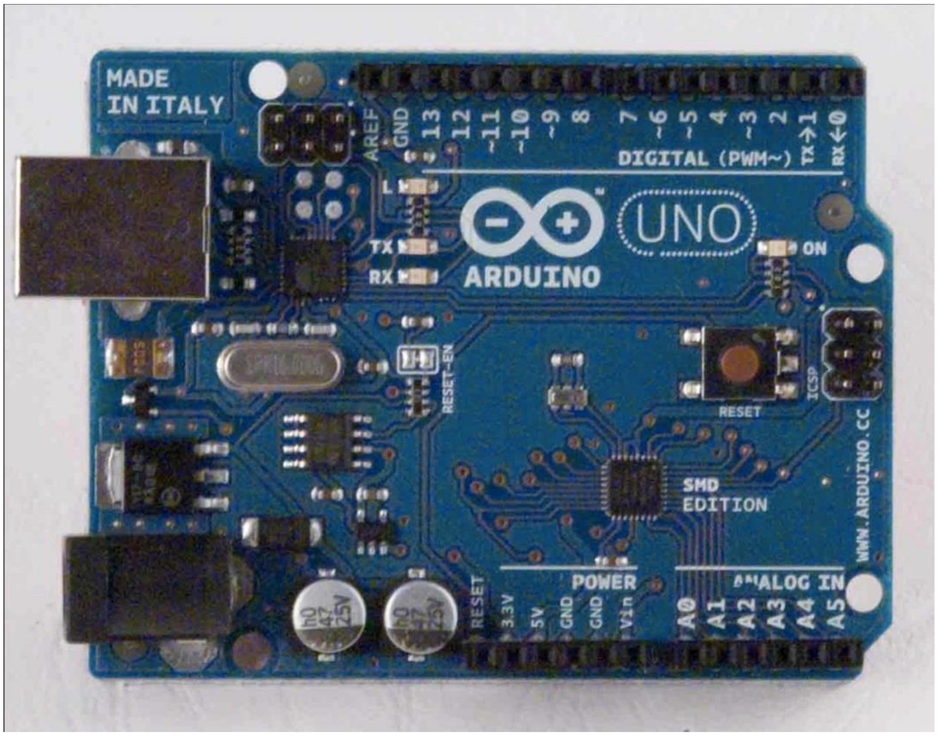
Sound Energy Values :

Instant E/<E> :	6.152	2.949	2.917	5.525	1.004	0.192	1.986	45.404	3.692	3.471
BandWidth :	BP = 2	BP = 2	BP = 3	BP = 3	BP = 3	BP = 3	BP = 3	BP = 3	BP = 3	BP = 3
Subband Number :	#0	#1	#2	#3	#4	#5	#6	#7	#8	#9
Instant E/<E> :	14.341	17.026	2.579	0.819	13.780	4.369	2.957	11.481	0.919	4.626
BandWidth :	BP = 3	BP = 3	BP = 3	BP = 3	BP = 4	BP = 4	BP = 4	BP = 4	BP = 4	BP = 4
Subband Number :	#10	#11	#12	#13	#14	#15	#16	#17	#18	#19
Instant E/<E> :	5.711	27.509	2.633	6.017	5.367	2.958	6.000	2.661	25.444	23.574
BandWidth :	BP = 4	BP = 4	BP = 4	BP = 4	BP = 4	BP = 4	BP = 4	BP = 5	BP = 5	BP = 5
Subband Number :	#20	#21	#22	#23	#24	#25	#26	#27	#28	#29
Instant E/<E> :	12.233	7.656	4.565	5.338	10.090	25.377	4.174	13.660	13.293	9.651
BandWidth :	BP = 5	BP = 5	BP = 5	BP = 5	BP = 5	BP = 5	BP = 5	BP = 5	BP = 5	BP = 5
Subband Number :	#30	#31	#32	#33	#34	#35	#36	#37	#38	#39
Instant E/<E> :	6.237	7.682	10.244	11.973	7.783	4.156	7.688	5.486	12.488	24.175
BandWidth :	BP = 6	BP = 6	BP = 6	BP = 6	BP = 6	BP = 6	BP = 6	BP = 6	BP = 6	BP = 6
Subband Number :	#40	#41	#42	#43	#44	#45	#46	#47	#48	#49
Instant E/<E> :	9.163	9.671	14.882	11.293	6.828	16.133	11.559	5.419	9.674	21.412
BandWidth :	BP = 6	BP = 6	BP = 7	BP = 7	BP = 7	BP = 7	BP = 7	BP = 7	BP = 7	BP = 7
Subband Number :	#50	#51	#52	#53	#54	#55	#56	#57	#58	#59
Instant E/<E> :	29.762	14.833	8.444	6.239	13.659	9.269	18.571	10.588	14.267	12.864
BandWidth :	BP = 7	BP = 7	BP = 7	BP = 7	BP = 7	BP = 8	BP = 8	BP = 8	BP = 8	BP = 8
Subband Number :	#60	#61	#62	#63	#64	#65	#66	#67	#68	#69
Instant E/<E> :	4.947	14.182	11.133	8.350	58.133	31.467	49.297	16.438	15.739	21.579
BandWidth :	BP = 8	BP = 8	BP = 8	BP = 8	BP = 8	BP = 8	BP = 8	BP = 8	BP = 9	BP = 9
Subband Number :	#70	#71	#72	#73	#74	#75	#76	#77	#78	#79
Instant E/<E> :	19.176	11.636	18.667	13.842	12.423	5.040	23.512	19.786	11.429	10.370
BandWidth :	BP = 9	BP = 9	BP = 9	BP = 9	BP = 9	BP = 9	BP = 9	BP = 9	BP = 9	BP = 9
Subband Number :	#80	#81	#82	#83	#84	#85	#86	#87	#88	#89
Instant E/<E> :	6.581	13.065	9.833	10.692	11.160	8.517	7.519	2.756	8.563	13.759
BandWidth :	BP = 10	BP = 10	BP = 10	BP = 10	BP = 10	BP = 10	BP = 10	BP = 10	BP = 10	BP = 10
Subband Number :	#90	#91	#92	#93	#94	#95	#96	#97	#98	#99
Instant E/<E> :	7.967	1.256	3.054	9.604	8.265	17.051	8.893	8.206	8.333	9.038
BandWidth :	BP = 10	BP = 10	BP = 10	BP = 11	BP = 11	BP = 11	BP = 11	BP = 11	BP = 11	BP = 11
Subband Number :	#100	#101	#102	#103	#104	#105	#106	#107	#108	#109
Instant E/<E> :	16.778	5.382	11.822	9.280	17.529	11.043	20.143	8.468	18.476	12.611
BandWidth :	BP = 11	BP = 11	BP = 11	BP = 11	BP = 11	BP = 11	BP = 12	BP = 12	BP = 12	BP = 12
Subband Number :	#110	#111	#112	#113	#114	#115	#116	#117	#118	#119
Instant E/<E> :	5.935	20.400	3.692	9.727	17.851	1.360	3.723	2.243		
BandWidth :	BP = 12	BP = 12	BP = 12	BP = 12	BP = 12	BP = 12	BP = 12	BP = 12		
Subband Number :	#120	#121	#122	#123	#124	#125	#126	#127		

From: Beat Detection Algorithms, page 14

2.3 Schematics, Calculations, and Simulations

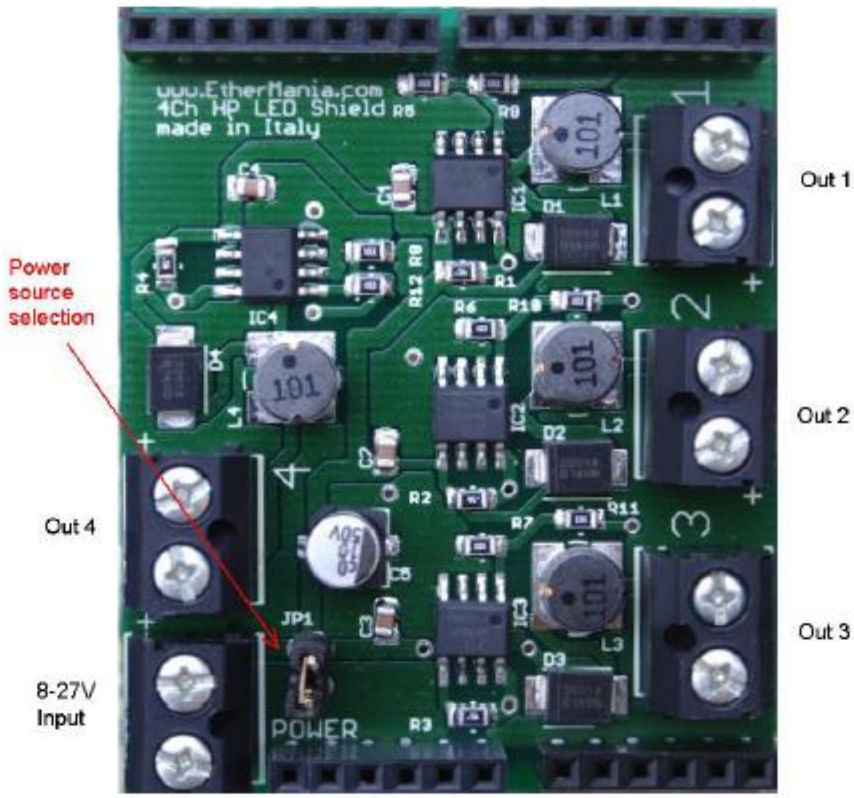
Arduino Uno



From http://arduino.cc/en/Main/arduinoBoardUno

Input Name	Function	Output Name	Connection
A0	Signal from LPF	D3	Output to Driver Shield 1
A1	Signal from BPF	D5	Output to Driver Shield 2
A2	Signal from HPF	D6	Output to Driver Shield 3
+5V	Power for Microphone		

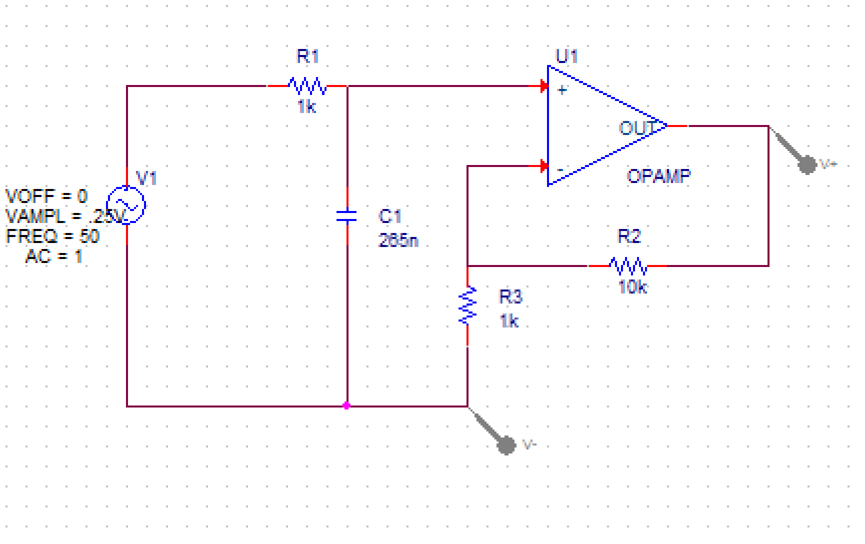
HP LED Driver Shield



http://www.ethermania.com/shop/index.php?main_page=product_info&products_id=1176

Powered by 24 V Power Supply
Data Input: from Arduino Uno PWM Digital Output
Out1: LED Strand 1
Out2: LED Strand 2
Out3: LED Strand 3
Out4: LED Strand 4

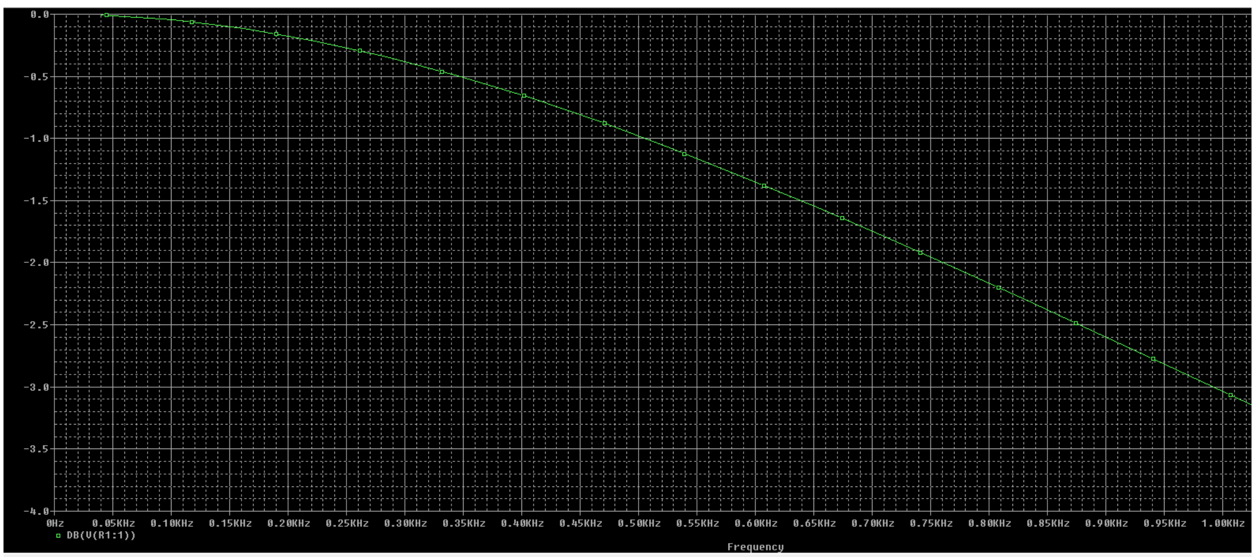
Low Pass Circuit Schematic



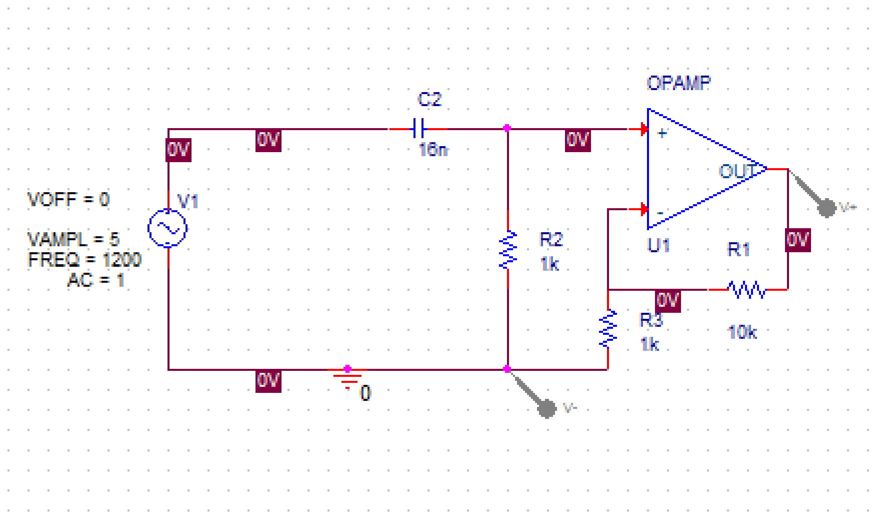
Design Specifications

$f_{low} = 1/(2*pi*R1*C_{low})$
$C_{low} = 1/(2*pi*R1*f_{low})$
$C_{low} = 1/(2*pi*1kOhm*60\ Hz)$
$C_{low} = 265\ nF$

Simulation



High Pass Filter Circuit Schematic



Design Specifications

$f_{high} = 1/(2*pi*R1* C_{high})$

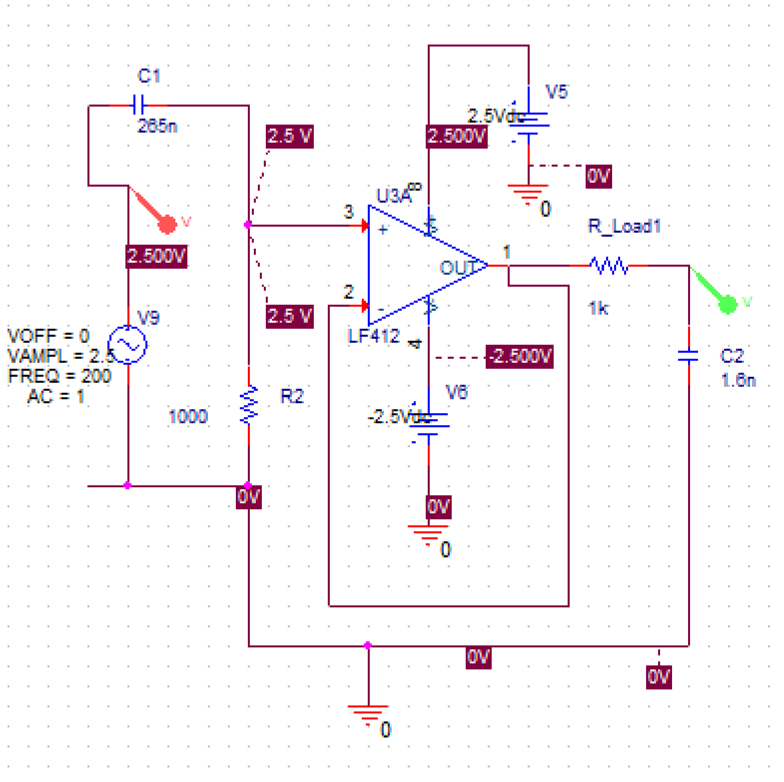
$C_{high} = 1/(2*pi*R1*f_{high})$

$C_{high} = 1/(2*pi*1kOhm*1000\ Hz)$

$C_{high} = 16\ nF$

Simulation

Band Pass Filter Circuit Schematic



Design Specifications

$f_{low} = 1/(2*pi*R1*C_{low})$

$C_{low} = 1/(2*pi*R1*f_{low})$

$C_{low} = 1/(2*pi*1kOhm*60\ Hz)$

$C_{low} = 265\ nF$

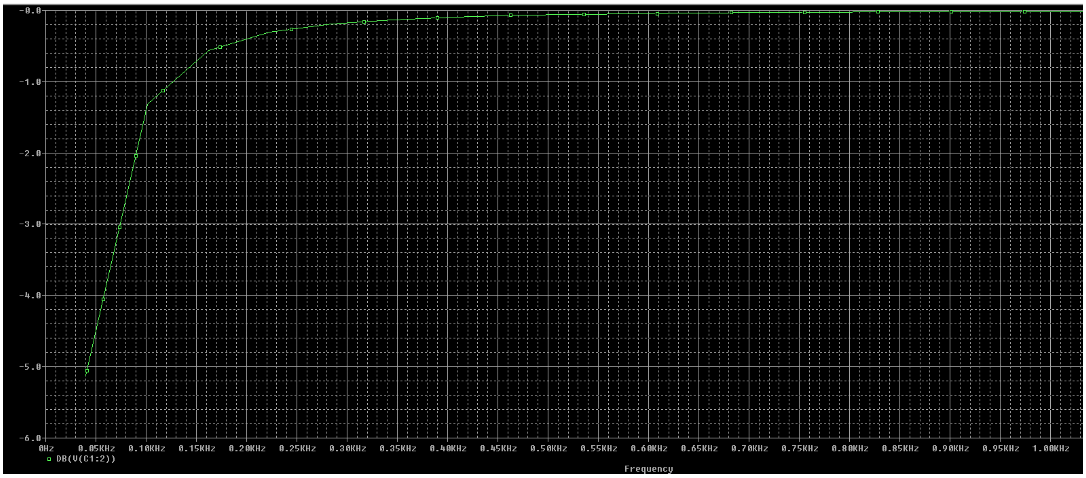
$f_{high} = 1/(2*pi*R1*C_{high})$

$C_{high} = 1/(2*pi*R1*f_{high})$

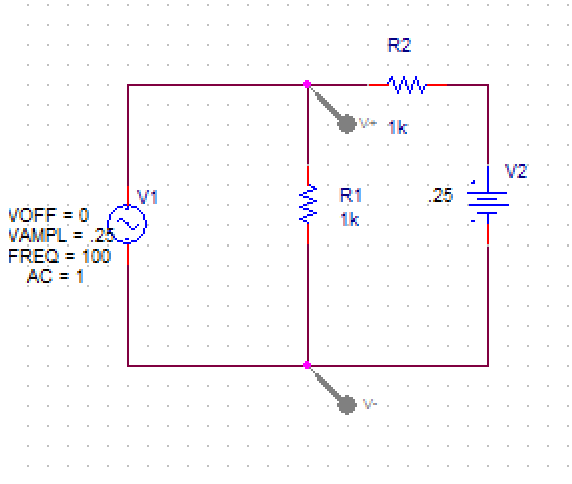
$C_{high} = 1/(2*pi*1kOhm*1000\ Hz)$

$C_{high} = 16\ nF$

Simulation

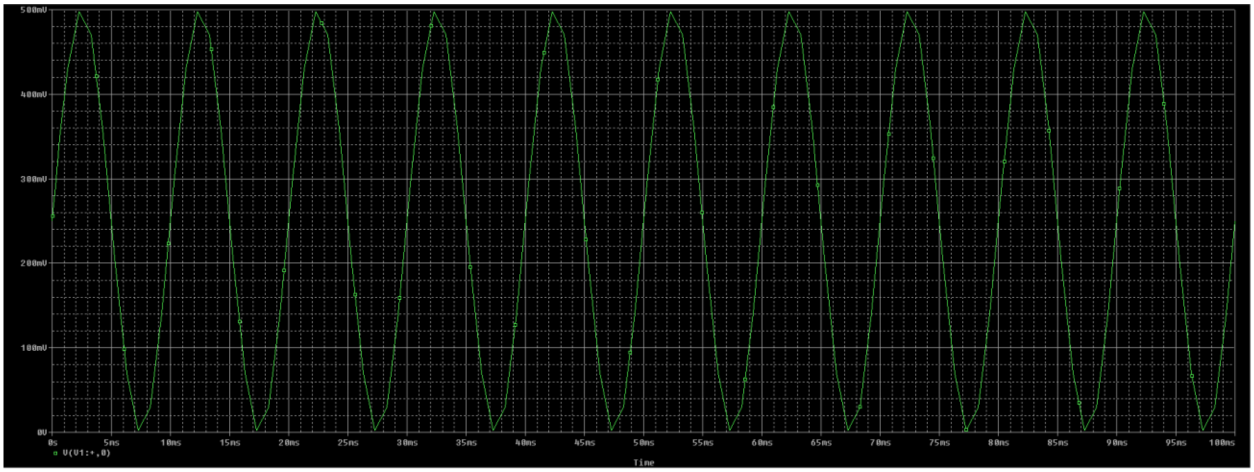

-0.0
-1.0
-2.0
-3.0
-4.0
-5.0
-6.0
0Hz
0.05KHz
0.10KHz
0.15KHz
0.20KHz
0.25KHz
0.30KHz
0.35KHz
0.40KHz
0.45KHz
0.50KHz
0.55KHz
0.60KHz
0.65KHz
0.70KHz
0.75KHz
0.80KHz
0.85KHz
0.90KHz
0.95KHz
1.00KHz
DB(V(C1:2))
Frequency

Microphone Circuit



Design Specifications
DC Offset = V2*(R1)/(R2+R1)
Desired Offset = .25 V = .5V(R1)/(R2+R1)
R1 = 1kOhm
R2 = 1kOhm

Simulation



3.0 Requirements and Verification

3.1.1 Microphone

Requirement	Verification
Microphone Powered and Grounded 1) Microphone powered with V_{DD} range from 1.6 V to 3.3V +/- 20%\ 2) Microphone grounded to 0 Volts	1) Play audio signal possessing frequency of 1000 Hz. Using a volt-meter, find the voltage difference at the microphone output and display on the oscilloscope. 2) Using a volt-meter, measure the voltage at the ground pin to ensure that it equals 0 V +/- .25 V
Microphone detects audio signal 1) Vout from microphone has range of -.25 V to .25V +/- 50%	1) Play audio signal possessing frequency of 1000 Hz. Using a volt-meter, find the voltage difference at the microphone output and display on the oscilloscope.
Maximum output voltage amplitude level maintained by Gain factor of 1+/- 40% 1) Measure input voltage range range [-.25 V, .25 V] +/-20% 2) Measure output voltage range range [-.25 V, .25 V] +/-20% 3) Compute the average input voltage [-.3 V, .3 V] +/-20% 4) Compute the average output voltage [-.3 V, .3 V] +/-20% 5) Gain = 1 +/- 40%	1) Apply 1000 Hz signal of amplitude = .25 V with function generator to Vin terminal. Using a volt-meter, measure input voltage from filter. Display on oscilloscope. 2) Using a volt-meter, measure output voltage from filter. Display on oscilloscope. 3) (Vin Max+Vin Min)/2 4) (Vout Max+Vout Min)/2 5) Divide the average output voltage by the average input voltage
DC Offset shifts signal 1) Offset Circuit Powered 2) Offset Circuit Grounded 3) Vout from Offset Circuit between [0, .5] V +/-.2V	1) Probe the V++ pin with voltmeter to ensure that it receives 5 V +/- 20% 2) Probe the V-- pin with voltmeter to ensure that it receives -5 V +/- 20% 3) Input 1000 Hz signal with voltage amplitude = .25 V. Measure the output with a voltmeter and display output on oscilloscope to ensure signal range is from [0, .5] V +/- .20 V

3.1.2 Low Pass Filter

This device is intended to pass frequencies below 60 Hz

Requirement	Verification
Gain = 10 1) Measure Vout range [-2.5 V, 2.5 V] +/-20% 2) Compute Vout average [-2.5 V, 2.5 V] +/-20% 3) Measure Vin range [-.25V, ,25 V] +/-20% 4) Compute Vin average [-.25 V, .25 V] +/-20% 5) Gain = 10 +/-50%	Apply 40 Hz signal of amplitude = 2.5 V with function generator to Vin terminal. 1) Using a volt-meter, measure output voltage range from filter. Display on oscilloscope 2) (Vout Max+Vout Min)/2 3) Using a volt-meter, measure input voltage range from filter. Display on oscilloscope 4) (Vin Max+Vin Min)/2 5) Divide average Vout by average Vin.
Frequency below 60 Hz passed Vout = Vin+/- 20%	Apply 40 Hz signal of amplitude = 2.5 V with function generator to Vin terminal. Using a volt-meter, measure output voltage range from filter. Display on oscilloscope
Frequency above 60 Hz attenuated Vout = 0 +/-.5 V	Apply 100 Hz signal of amplitude = 2.5 V with function generator to Vin terminal. Using a volt-meter, measure output voltage from filter. Display on oscilloscope

3.1.3 High Pass Filter

This device passes the frequencies above 1000 Hz

Requirement	Verification
Gain = 1 1) Measure Vout range [-2.5 V, 2.5 V] +/-20% 2) Compute Vout average [-2.5 V, 2.5 V] +/-20% 3) Measure Vin range [-.25V, .25 V] +/-20% 4) Compute Vin average [-.25 V, .25 V] +/-20% 5) Gain = 10 +/-50%	Apply 1000 Hz signal of amplitude = 2.5 V with function generator to Vin terminal. 1) Using a volt-meter, measure output voltage range from filter. Display on oscilloscope 2) (Vout Max+Vout Min)/2 3) Using a volt-meter, measure input voltage range from filter. Display on oscilloscope 4) (Vin Max+Vin Min)/2 5) Divide average Vout by average Vin.
Frequency below 1000 Hz attenuated Vout = 0 +/- .5 V	Apply 40 Hz signal of amplitude = 2.5 V with function generator to Vin terminal. Using a volt-meter, measure output voltage from filter. Display on oscilloscope
Frequency above 1000 Hz passed Vout = Vin+/- 20%	Apply 1200 Hz signal of amplitude = 2.5 V with function generator to Vin terminal. Using a volt-meter, measure output voltage from filter. Display on oscilloscope

3.1.4 Band Pass Filter

Circuit passes frequencies between 60 Hz and 1000 Hz

Requirement	Verification
Gain = 10 1) Measure Vout range [-2.5 V, 2.5 V] +/-20% 2) Compute Vout average [-2.5 V, 2.5 V] +/-20% 3) Measure Vin range [-.25V, .25 V] +/-20% 4) Compute Vin average [-.25 V, .25 V] +/-20% 5) Gain = 10 +/-50%	Apply 500 Hz signal of amplitude = 2.5 V with function generator to Vin terminal. 1) Using a volt-meter, measure output voltage range from filter. Display on oscilloscope 2) (Vout Max+Vout Min)/2 3) Using a volt-meter, measure input voltage range from filter. Display on oscilloscope 4) (Vin Max+Vin Min)/2 5) Divide average Vout by average Vin.
Frequency below 60 Hz attenuated Vout = 0 +/- .5 V	Apply 40 Hz signal of amplitude = .25 V with function generator to Vin terminal. Using a volt-meter, measure output voltage from filter. Display on oscilloscope
Frequency above 1000 Hz attenuated Vout = 0 +/- .5 V	Apply 1200 Hz signal of amplitude = .25 V with function generator to Vin terminal. Using a volt-meter, measure output voltage from filter. Display on oscilloscope
Frequencies between 60 Hz & 1000 Hz passed Vout = Vin+/- 20%	Apply 500 Hz signal of amplitude = .25 V with function generator to Vin terminal. Using a volt-meter, measure output voltage from filter. Display on oscilloscope

3.1.5 Arduino Microcontroller

Requirement	**Verification**
Device is powered 1) Voltage Supplied to Arduino Uno is between [7,12] Volts	1) Probe supply terminal of Arduino Uno with a volt-meter. Display signal on oscilloscope.
Device accepts signal from analog input 1) Analog input detects range of [0, 5] V	1) Apply 500 Hz signal of amplitude = 2.5 V and DC offset 2.5 V with function generator to analog input 1. Graph input data in Arduino programming module
Device outputs to digital PWM pins 1) Ensure that pins D3,D5, D6 output dispatch signal	1) Apply 500 Hz signal of amplitude = 2.5 V and DC offset 2.5 V with function generator to analog input 1. Measure output on digital pins D3, D5, & D6 using oscilloscope.

3.1.6 Arduino Power Supply

Requirement	Verification
Device supplies proper voltage to the Arduino Microcontroller 1) Voltage Supplied between [7, 12] Volts	1) Probe Arduino Uno's voltage input port with a volt-meter. Display signal on oscilloscope.
Supplied Current is Acceptable 1) Applied current remains between 0 A and 1.0A	1) Place a resistor in series between the Power Supply and the Arduino Uno. Use Ammeter to measure current

3.1.7 Four Channel High Power LED Driver Shield

Requirement	**Verification**
Driver receives correct voltage 1) Voltage Supplied between [7, 27] Volts	**1)** Probe output of battery voltage port with a volt-meter. Display signal on oscilloscope.
Driver receives signal from Arduino Uno 1) Output signal from Arduino Data Pins between [0, 40] mA +/- 20%	**1)** Probe input of Driver Shield with a ammeter. Display signal on oscilloscope.
Driver powers LED strands 1) With input signal, LEDs become distinguishably on	1) Input square wave function with 50% Duty Cycle. Observe visible illumination

3.1.8 24V Power Supply

Requirement	Verification
Device supplies proper voltage to the LED Driver Shield 1) Voltage Supplied between [7, 27] Volts	1) Probe LED Driver Shield's voltage input port with a volt-meter. Display signal on oscilloscope.
Supplied Current is Acceptable 1) Applied current remains between 0 A and 1.5A	1) Place a resistor in series between the Power Supply and the Arduino Uno. Use Ammeter to measure current

3.1.9 LED Strand

Requirement	Verification
Voltage over LEDs fluctuates with time 1) LEDs receive voltage difference between [0,3.0] +/- 20% V 2) Calculate Vout Average [0,3.0] +/- 20% V 3) LEDs turn on/off	Apply 2 Hz square wave signal of amplitude = 2.5 V with function generator to Vin terminal. 1) Using a volt-meter, measure input voltage range over LEDs. Display on oscilloscope 2) (Vout Max+Vout Min)/2 3) Using a volt-meter, measure output voltage range from filter. Display on oscilloscope
LEDs receive proper current 1) LEDs receive current between [250 mA, 450 mA] +/- 20%	Apply 2 Hz square wave signal of amplitude = 2.5 V with function generator to LED strip input terminal. 1) Using an ammeter, measure input

	current range from filter. Display on oscilloscope

3.1.10 Beat Detecting Algorithm

Calculates Fast Fourier Transform	Apply data series of 16 data points into Fast Fourier Transform component of Algorithm. Compare algorithm output with expected Fourier Transforms of data inputs. Find the ratio of each ‘Algorithm Output’ to ‘Calculated Fast Fourier Transform s’
Calculates the sum of a series	Apply data series of 16 data points into Sum component of Algorithm. Calculate expected value for the sum. Find ratio of ‘Algorithm Output’ to ‘Calculated Sum’. Ratio = 1+/- 20%
Variance between Instantaneous Energy and Average Energy Calculated	Enter values for Instantaneous Energy and Average Energy. Compare output value with calculated value. Find ratio of ‘Algorithm Output’ to ‘Calculated Variance’. Ratio = 1 +/- 20%
Comparison Value Calculated	Enter value for Variance. Compare output value with calculated value. Find ratio of ‘Algorithm Output’ to ‘Calculated C”. Ratio = 1 +/- 20%
Comparison between Instantaneous Energy and Average Energy Calculated 1) Beat Detected 2) No Beat Detected	1) Enter values for Instantaneous Energy, Average Energy, and ‘C’ that indicate a beat is detected. Ensure that algorithm indicates that a beat has been detected with ‘1’ output. 2) Enter values for Instantaneous Energy, Average Energy, and ‘C’ that indicate a beat is not detected. Ensure that algorithm indicates that a beat has been detected with ‘0’ output.

3.2 Tolerance Analysis

The beat detection algorithm should correctly detect beats such that the lights correctly follow the auditory beat signature. If the beat detection algorithm detects too few beats, then the lights will seem slow and uneven compared to the audio signal and thus will provide an unacceptable light show. If the beat detection algorithm detects too many beats (as in it detects non), the LEDs will respond too fast and possibly keep the remain in a superposition of On/Off states; this is also unacceptable performance. An analysis of calculated beat frequency versus expected beat frequency will be conducted by dividing the former by the latter. The target ratio is 1.00, with 25% error permissible on each end providing an acceptable error range of [.75, 1.25].

4.0 Cost & Schedule

Cost Breakdown

Labor

40 ($/hr)* 200 (hrs) * 2.5 = 20,000

Parts

Part Type	Part Number	Cost	Quantity	Cost*Quantity
Arduino Uno	Uno R3	31.99	1	31.99
LED Driver Shield	HPLED1SH	36.43	3	109.29
Arduino Power Supply	TOL-00298	5.95	1	5.95
Power Supply	RS-25-24	14.95	3	44.85
Resistor	1 kOhm Resistor	.10	6	.60
	10 kOhm Resistor	.10	2	.60
Capacitor	.015 uF Cap	.39	2	.78
	.27 uF Cap	.89	2	1.78
Op Amp	LF412CD	.44	4	1.76
Microphone	ADMP504	1.95	1	1.95
LEDs	SMD5050	.80	12	9.60

Final Cost Breakdown	(in US Dollars)
Anticipated Parts Cost	209.15
Estimated Labor Cost	20,000
Total	20,209.15

4.2 Schedule

Date	Tasks to be Completed
2/11/2013	Research specific part numbers for components
2/18/2013	Test power supply with LEDs Buy parts
2/25/2013	Combine Modular circuit diagrams
3/4/2013	Begin implementing beat detection algorithm
3/11/2013	Configure LED Driver Shield
3/18/2013	Construct Microphone & Power Supply
3/25/2013	Connect and test LED Driver Shield with LEDs

4/1/2013	Fix design flaws, buy new replacement parts
4/8/2013	Assemble each module and connect them together
4/15/2013	Test Integrated system
4/22/2013	Fix Bugs related to system
4/29/2013	Test Final Project

5.0 Safety

This product must be safe to operate as it can cause bodily harm if not utilized correctly. The voltages (less than 24.0 V) and currents (less than 1.4 A) used are relatively small; thus, power dissipation that could cause harm within the circuit is not of large concern. In order to maintain this level of safety, the gains on the microphone amplifier and the filters must remain on the same order of magnitude. Resistors can vary wildly in magnitude; the ECE parts shop has resistors of at least 22 Mega Ohms in resistance magnitude. An increase in magnitude on the order of one million for the amplifiers would upset this balance and could cause injury.

6.0 Ethics

The IEEE Code of Ethics details guidelines upon which all members are supposed to act on. From the listed guidelines, this project makes pertinent the need to accept full responsibility for the claims that the project states it will perform. Furthermore, the data must support the design reasoning. Furthermore, circuit designs should be shared such that they can be tested, verified, and improved upon by the scientific community in the name of progress and the betterment of mankind. Any errors found must be corrected such that the intended outcome is realized. I fully understand my limitations and the extent of my knowledge when it comes to constructing circuits.

7.0 References

24 V Power Supply http://www.mouser.com/ProductDetail/Mean-Well/RS-25-24/?qs=pqZ7J9Gt/mpeEI49gnAroA==

4 Channel High Power LED Driver Shield:
http://www.ethermania.com/shop/download/HPLedShield.pdf

ADMP504 Microphone http://www.analog.com/en/audiovideo-products/mems-microphones/admp504/products/product.html

Arduino Uno http://arduino.cc/en/Main/arduinoBoardUno

Arduino Power Supply https://www.sparkfun.com/products/298

Goto, Matasaka. *An Audio-based Real-time Beat Tracking System for Music With* *or Without Drum-sounds*

National Semiconductor. *Op Amp Circuit Collection*. *Texas Instruments*. May 2004. Web. 11 Feb. 2013. http://www.ti.com/lit/an/snla140a/snla140a.pdf

National Semiconductor. LM102/LM302 Voltage Followers. November 1994. Web. 13 Feb. 2013.
http://read.pudn.com/downloads107/ebook/440848/%E4%BB%AA%E8%A1%A8%E6%94%BE%E5%A4%A7%E5%99%A8%E7%94%B5%E8%B7%AF/LM102_IcpdfCom.pdf

Parchizadeh, Hassan. "Pulse Width Modulation." (n.d.): n. pag. Web. 15 Feb. 2013.

Patin, Frederic: *Beat Detection Algorithms.*

Tsai, Sung-Feng, & Gu, Hung-Yan: *A Discrete-cepstrum Based Spectrum-envelope Estimation Scheme and Its Example Application of Voice Transformation*