REPORT:

Name: Bhagyashree Khairnar
Student ID: 1000917278

Project: **Low Noise Cancelation**

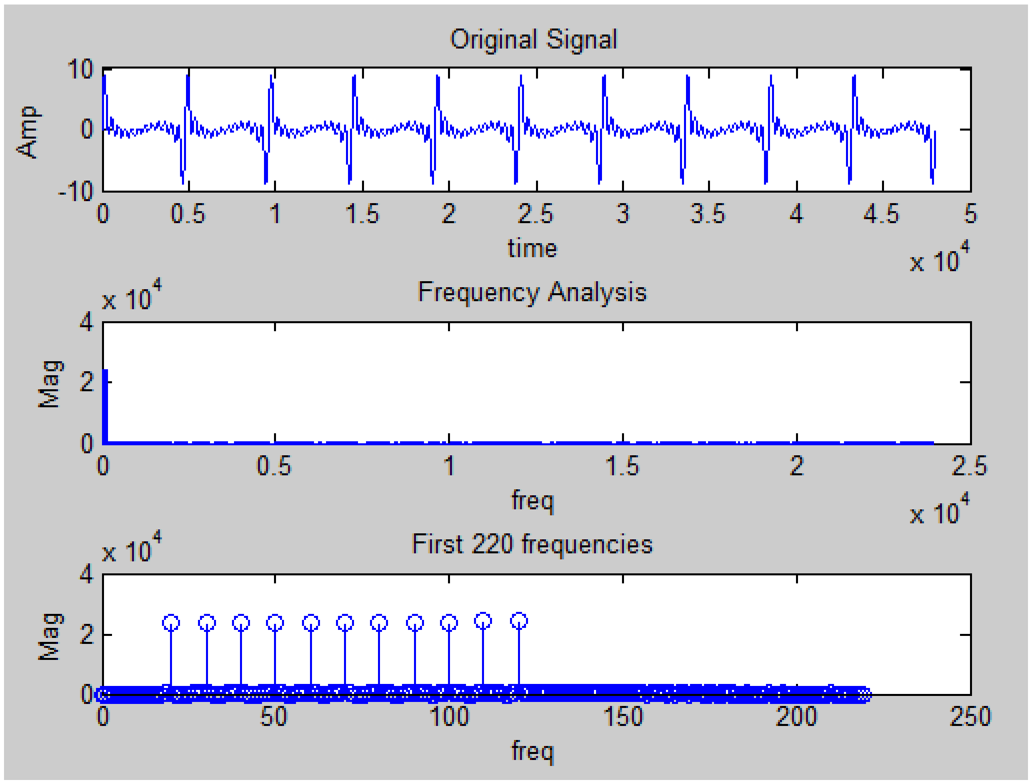
- Determine the low frequency that resonates objects in the ambience and filter them (using narrow notch filter)

What I have accomplished in this Project?

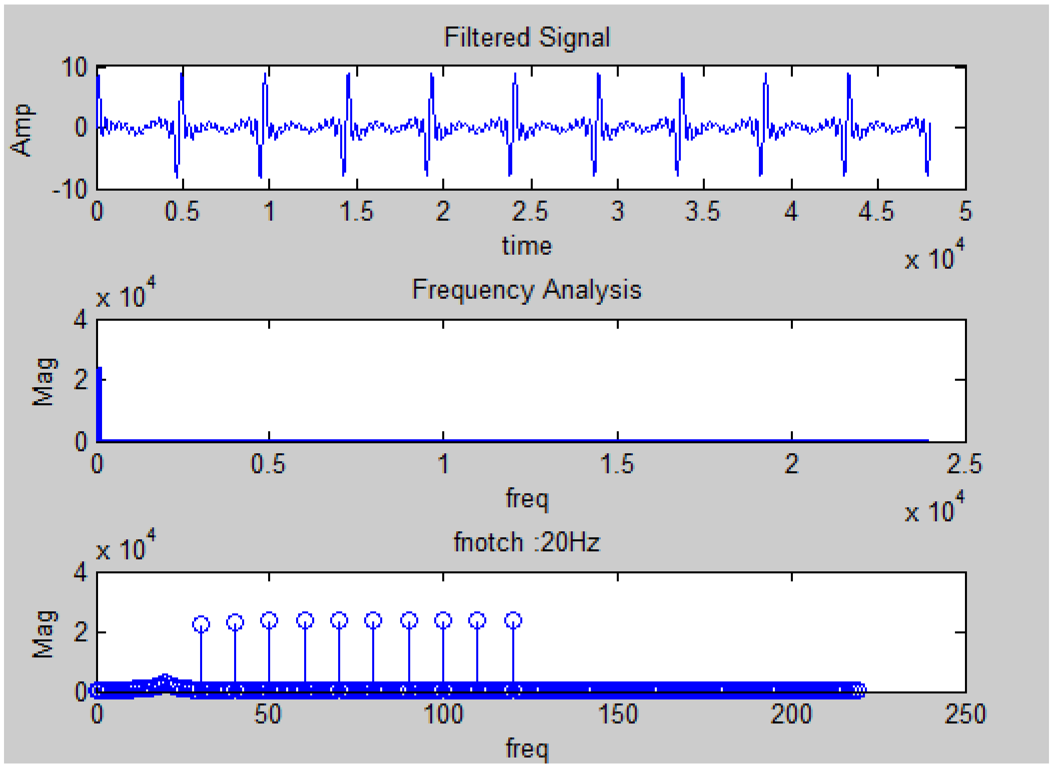
- Selected a range of frequencies in audible range particularly bass frequencies(used in real world applications)
- The frequencies were in the range 20-120Hz (inclusive of 20 and 120Hz)

  TRAINING MODE:
- Used Phase accumulator to generate each frequency tone (Please refer to file:Line_Table_256_LUT.xlsx)
- The LOOK UP TABLE contained 16 bit signed hex values and the sample values were found by dividing the unit circle into 256 points
- The phase for each frequency was implemented using the fixed point arithmetic logic.
- The principle used for determining the frequency that generates the highest noise was based on the fact that total energy on reception of the played back will be highest for the frequency that generates highest vibrations or noise in the objects near by
- An Aluminum foil model was used to perform the experiment
- Experiment & algorithm flow:
  1. Two arrays are used for storing the frequencies (freq[11]) and for storing the cycles per second per frequency (cycles[11])
  2. The entire experiment begins by generating the 20Hz tone (the code fetches the corresponding phase value and the number of cycles for which the respective frequency tone should be generated)
  3. The tone is played simultaneously, and as the microphone records the voltage values, the energy is calculated immediately and stored in an array (energy[11])
  4. 4 LEDS present on the board indicate the frequency being played
  5. The entire experiment ends with 120Hz played and its energy calculations.
  6. Once all the frequencies are played and their energy is calculated, sort() (Bubble Sort) function is used to determine the frequency that recorded highest energy.
  7. The frequencies in freq[] are also sorted and now, freq[0] contains the frequency to be filtered
  8. The LEDs indicate the frequency to be filtered

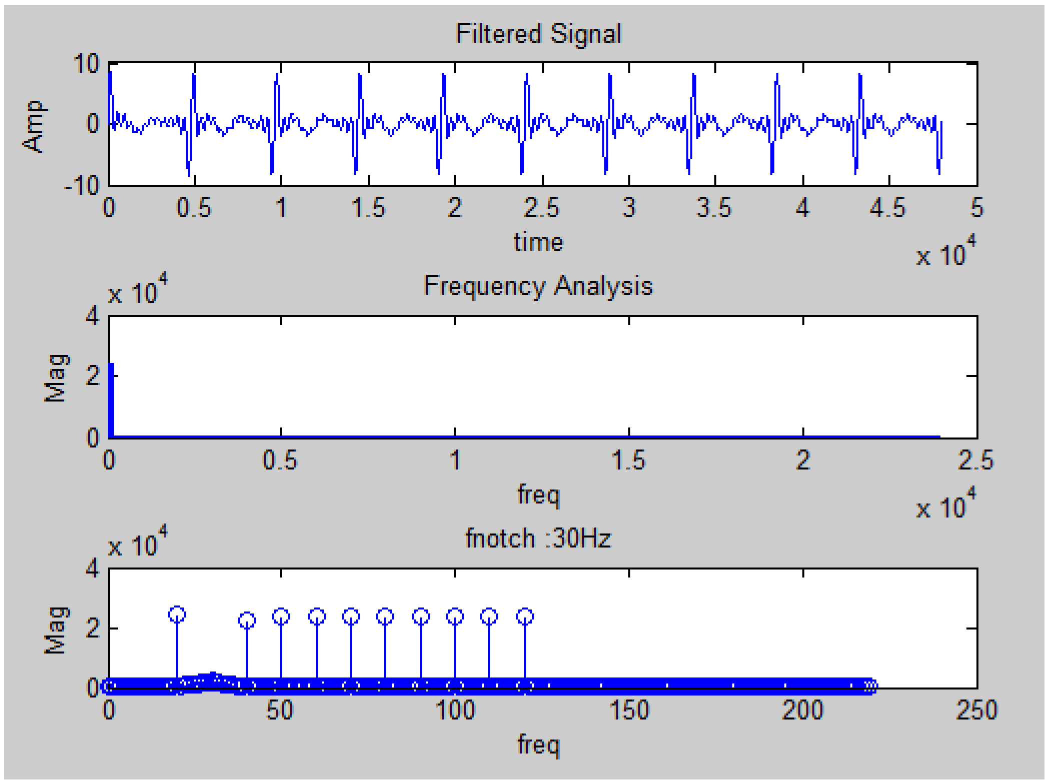
  PLAYBACK MODE:
- As per the requirement 11 narrow notch filters for each bass frequency were designed using the FDAtool in MATLAB.
- These filters are second order Direct Form I filters and the coefficients are double precision floating point numbers. (Please refer to file:Bass_Notch_Filter.xlsx)
- Following are frequency analysis figures obtained by using the equation
- $y[n] = a0 * x[n] + a1 * x[n-1] + a2 * x[n-2] – b1 * y[n-1] - b2 * y[n-2]$
- The input signal is a basic signal obtained by adding sine waves of each frequency sampled at fs = 48000Hz (please run the MATLAB code file:EE5391_BassConvolution.m)


Original Signal
Amp
time
x 10^4
Frequency Analysis
Mag
freq
First 220 frequencies

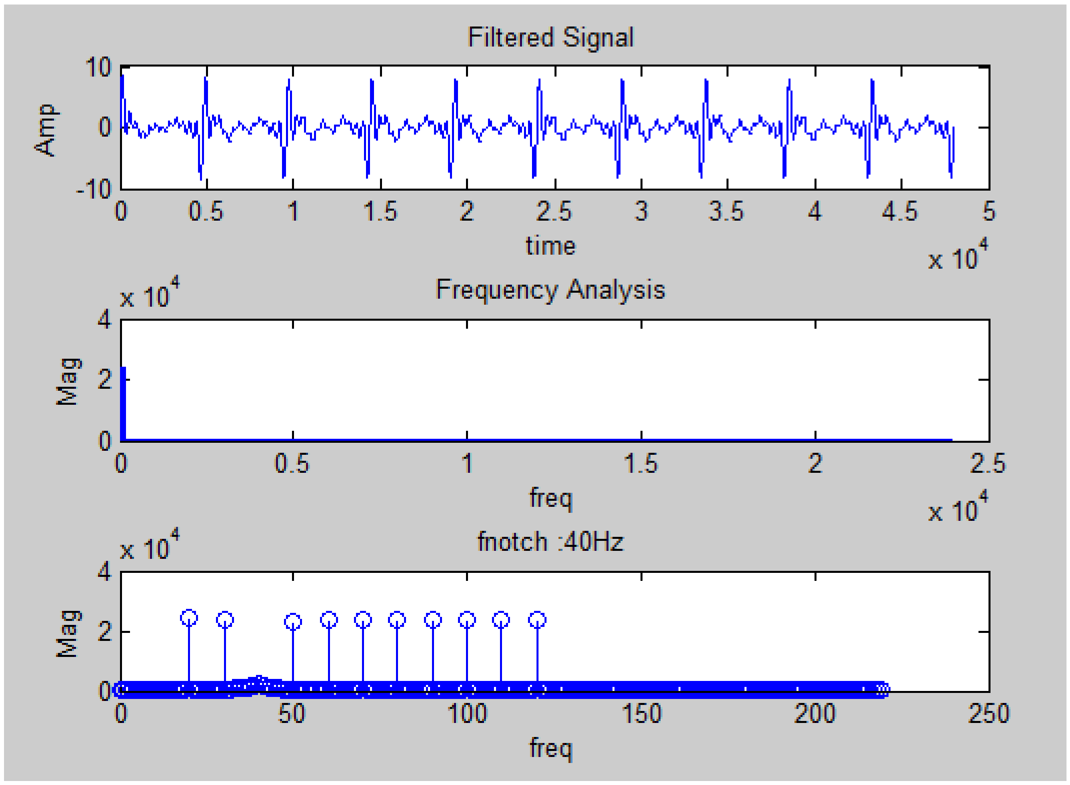
Fnotch = 20Hz


Filtered Signal
Amp
time
x 10^4
Frequency Analysis
Mag
freq
fnotch :20Hz

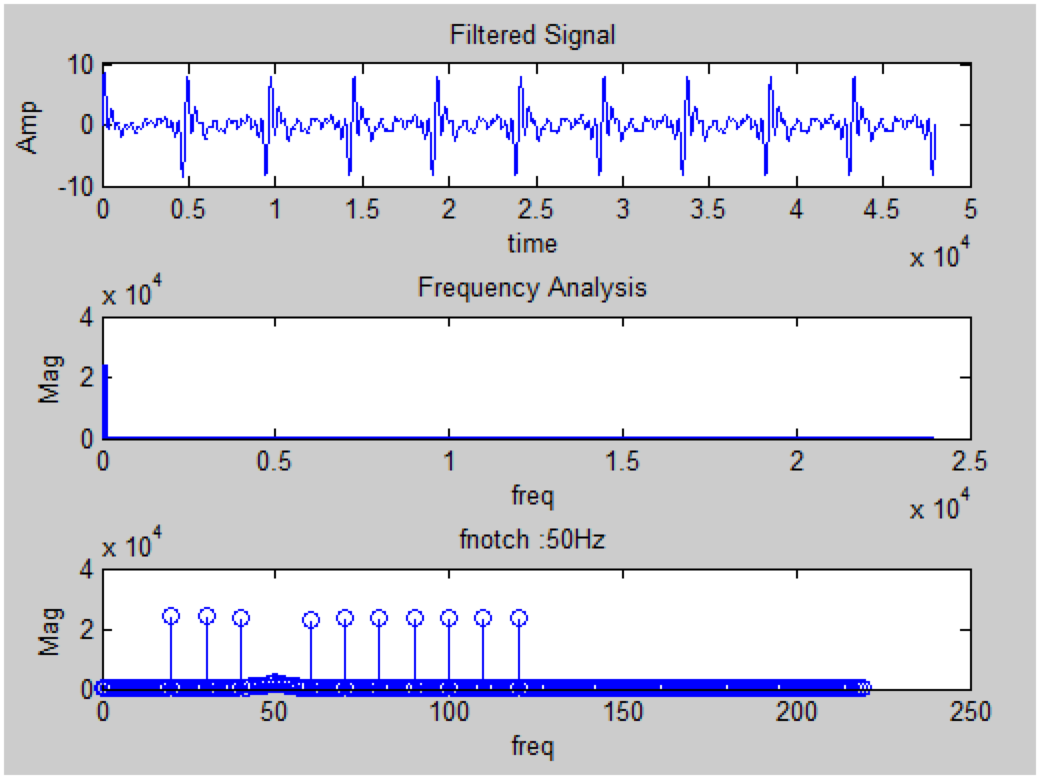
Fnotch = 30Hz



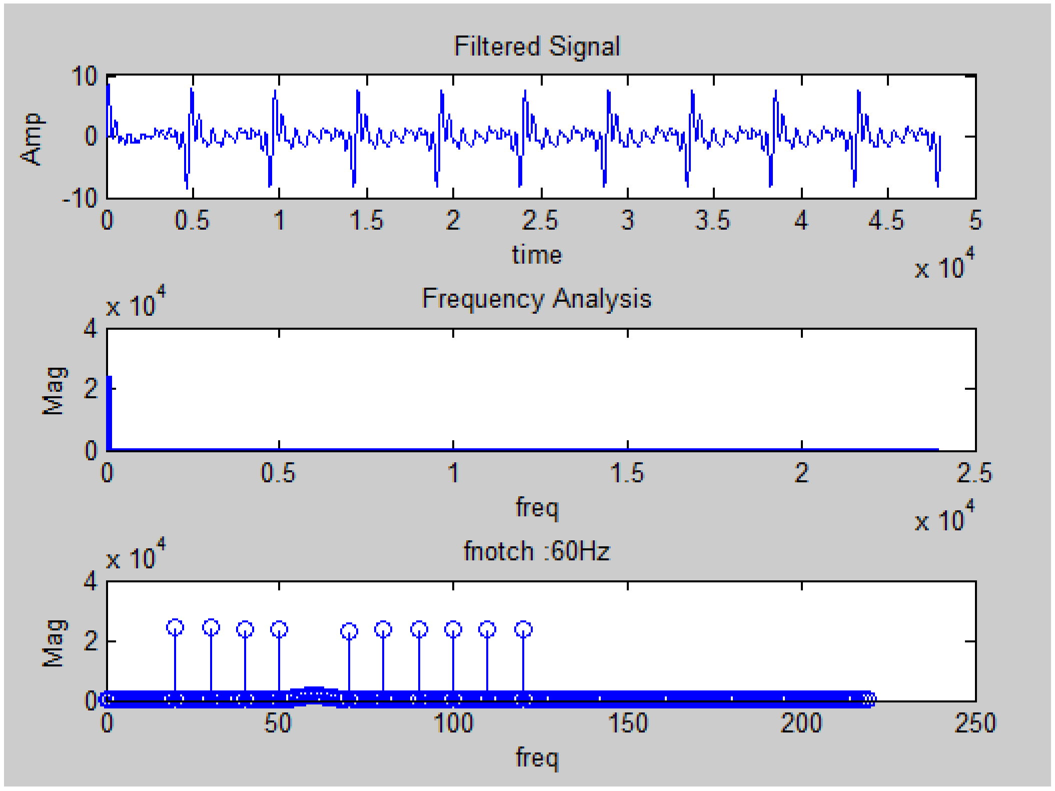
Fnotch = 40Hz



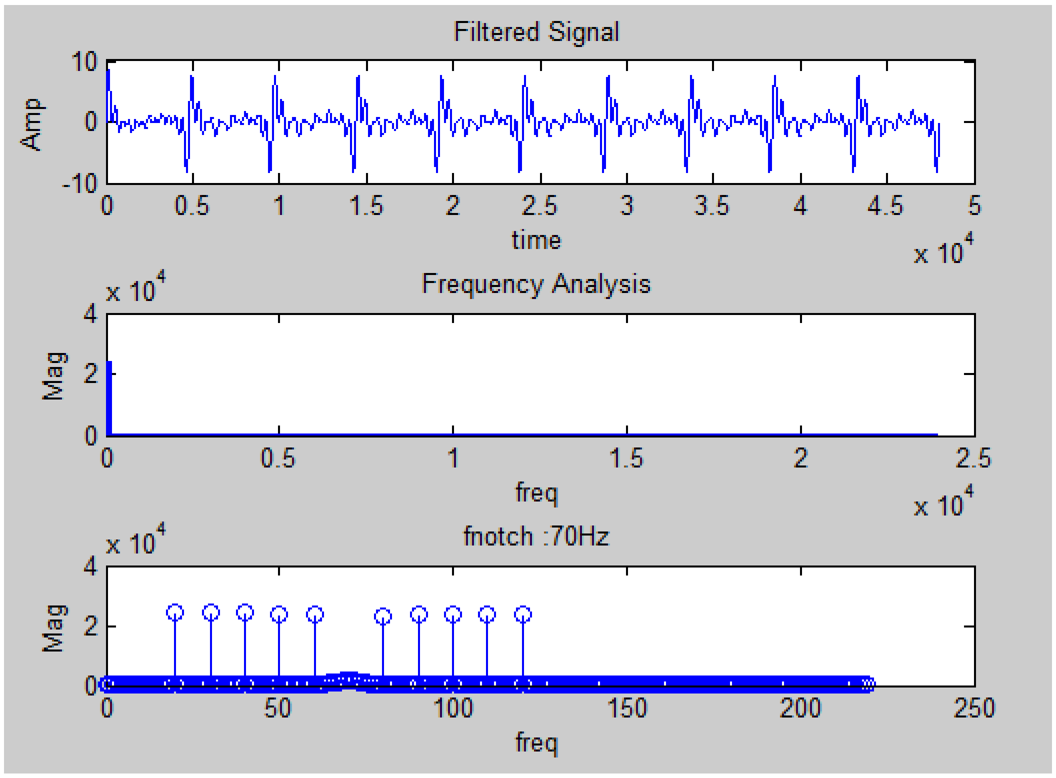
Fnotch = 50Hz



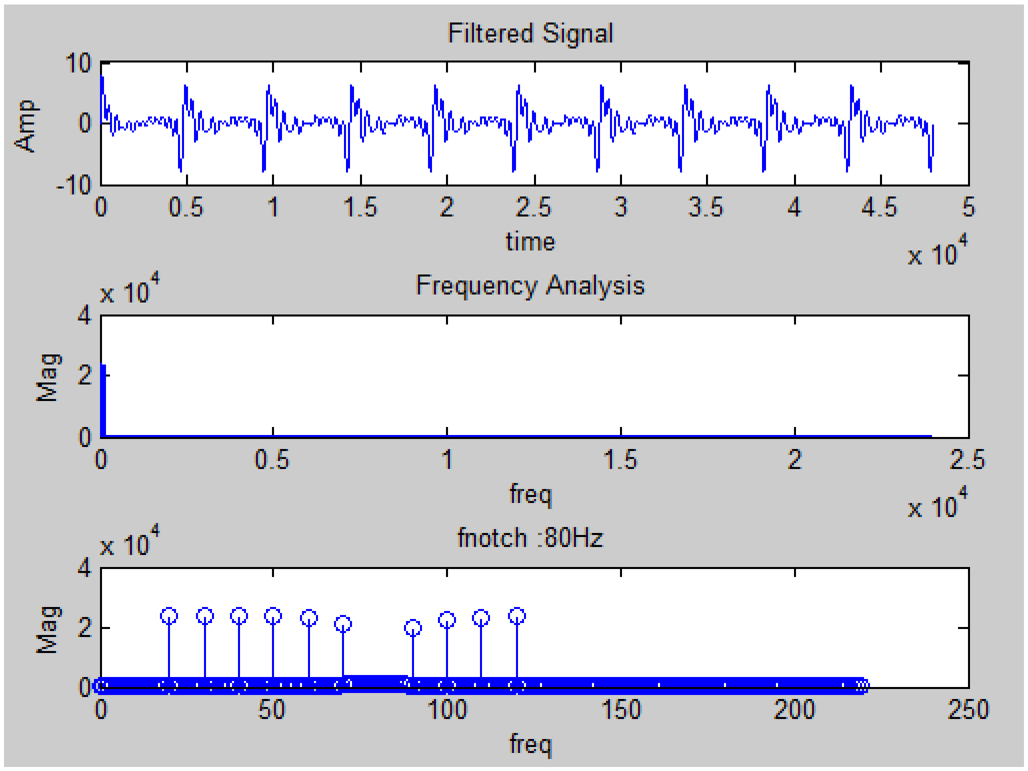
Fnotch = 60Hz



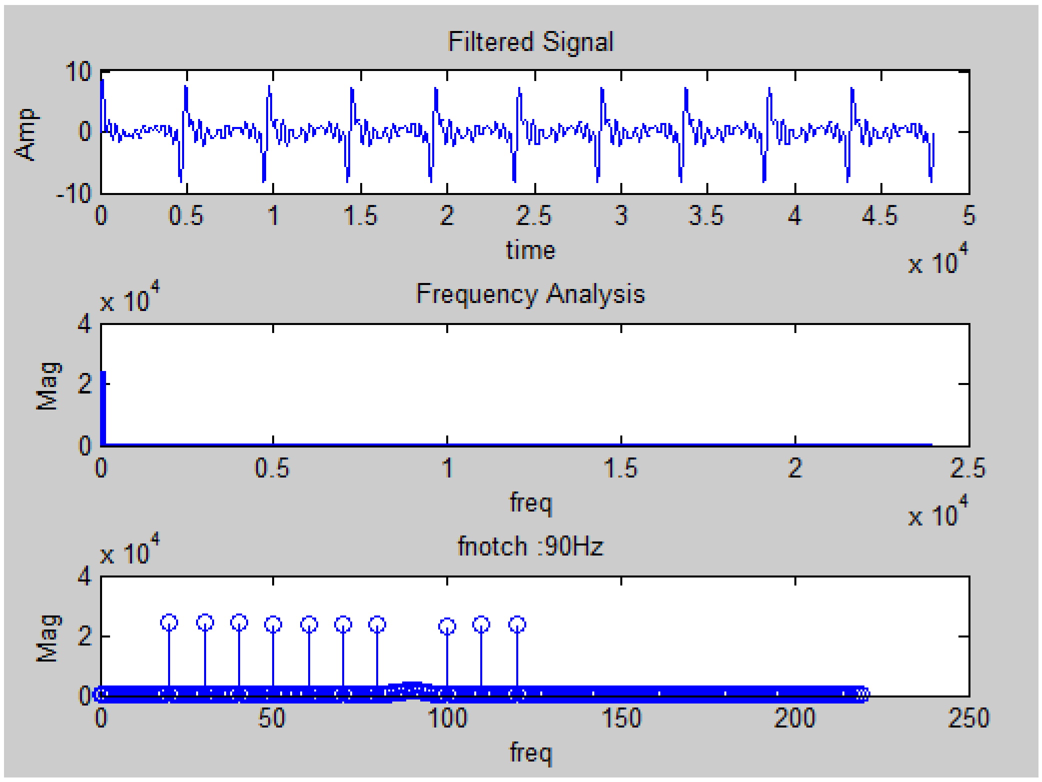
Fnotch = 70Hz



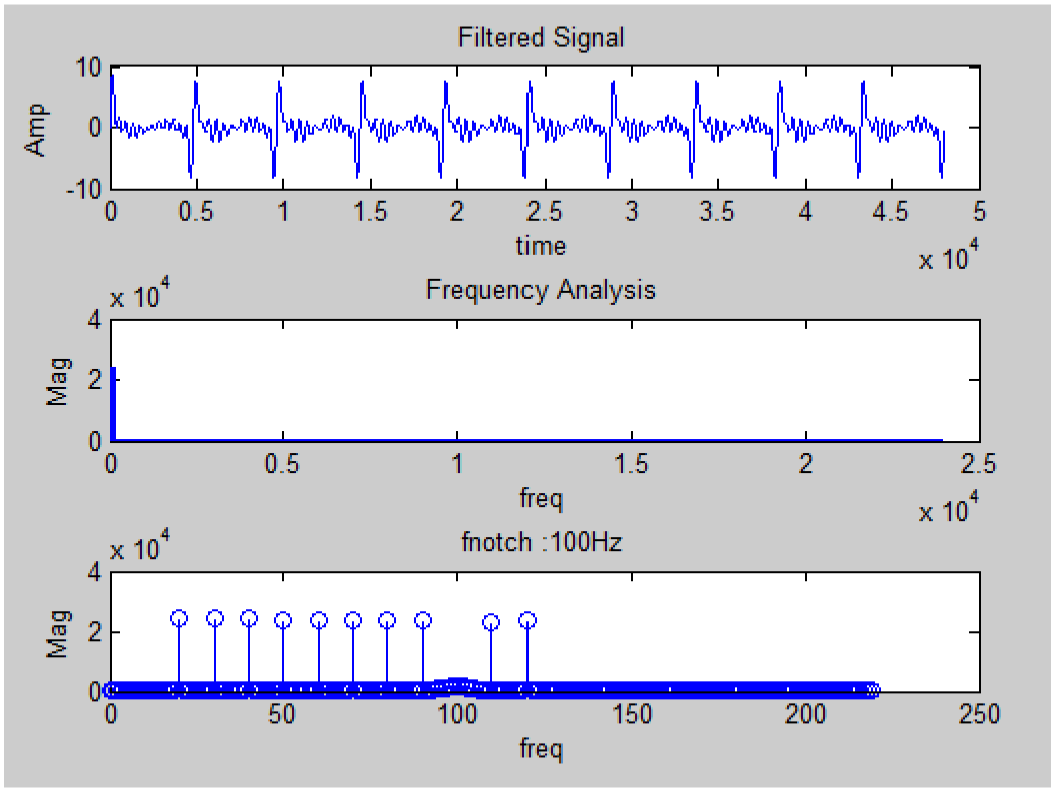
Fnotch = 80Hz



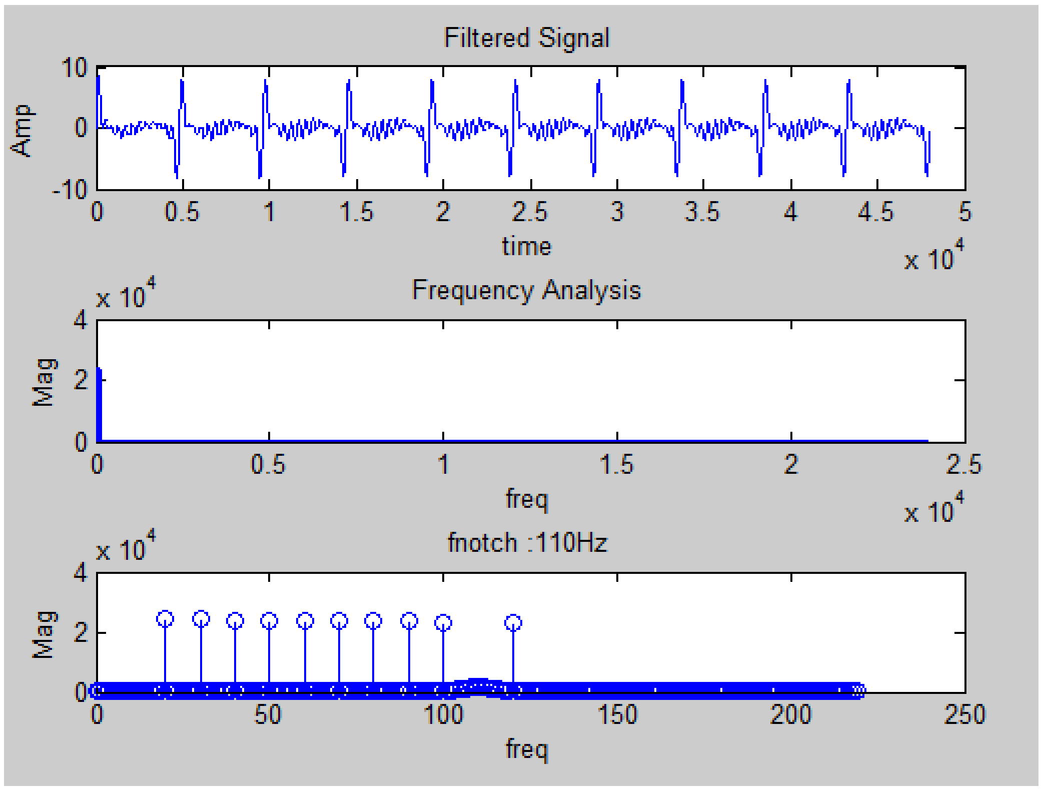
Fnotch = 90Hz



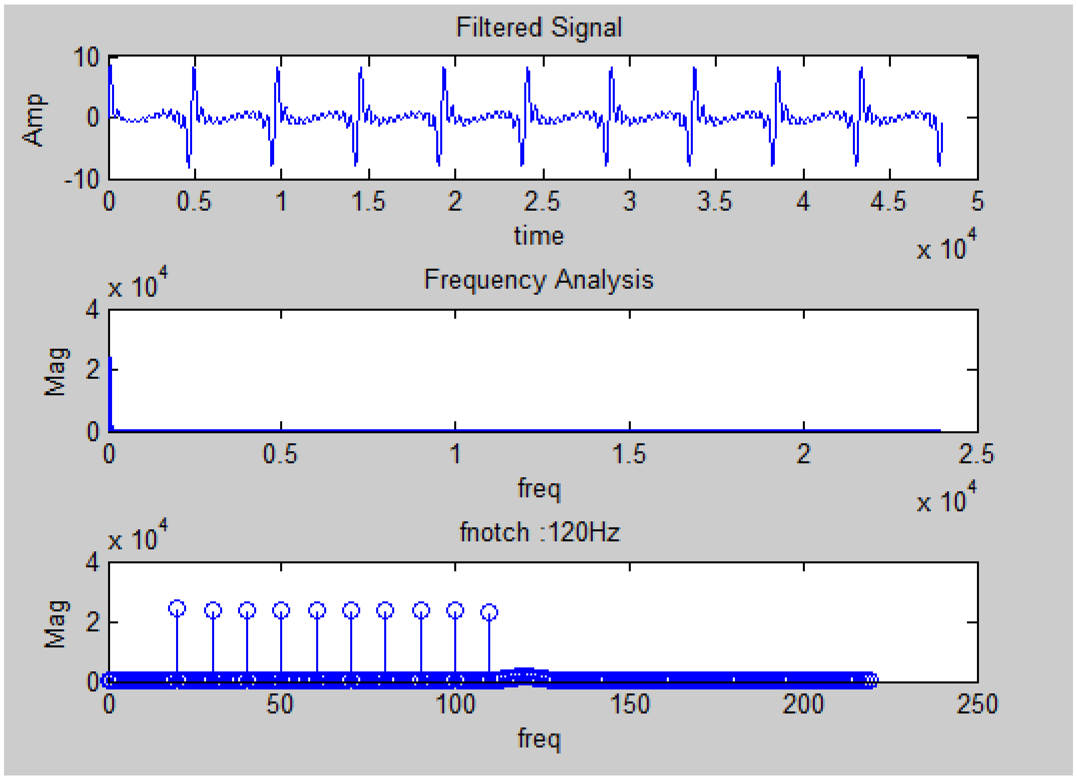
Fnotch = 100Hz



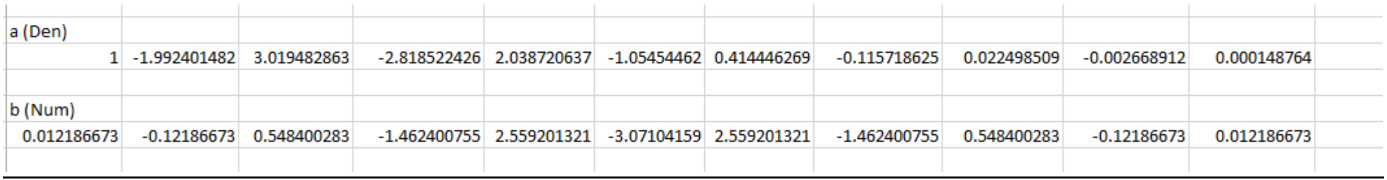
Fnotch = 110Hz



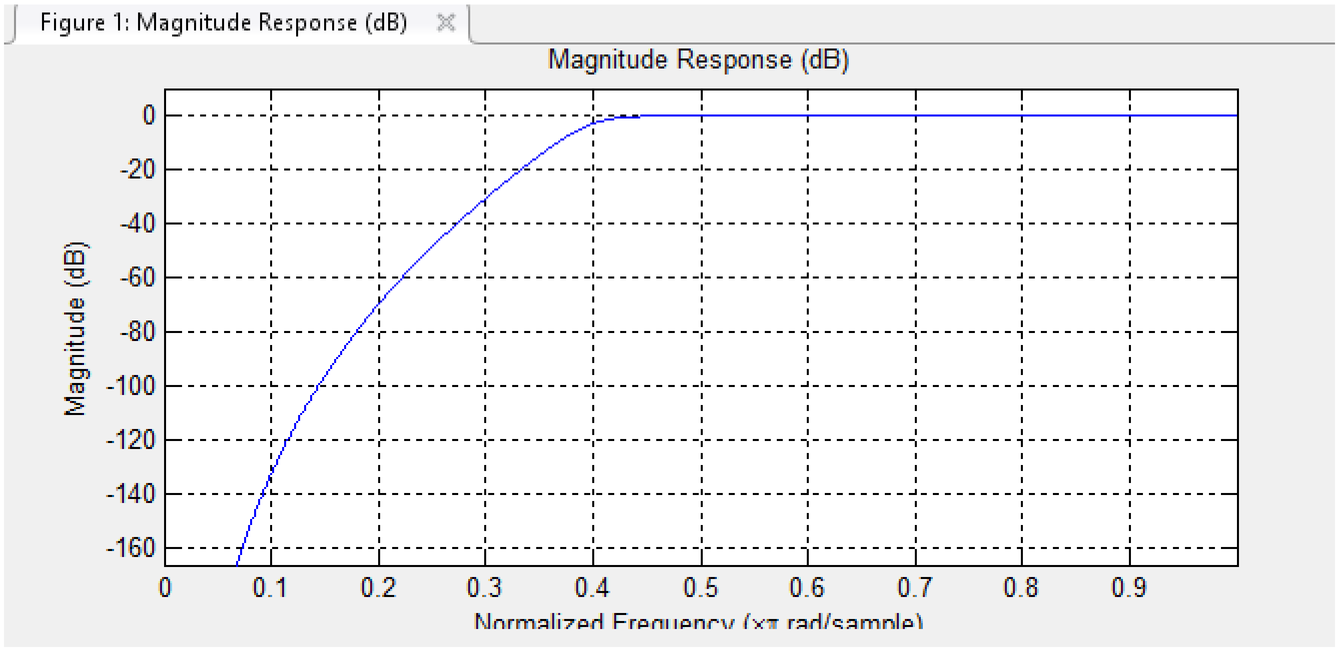
Fnotch = 120Hz



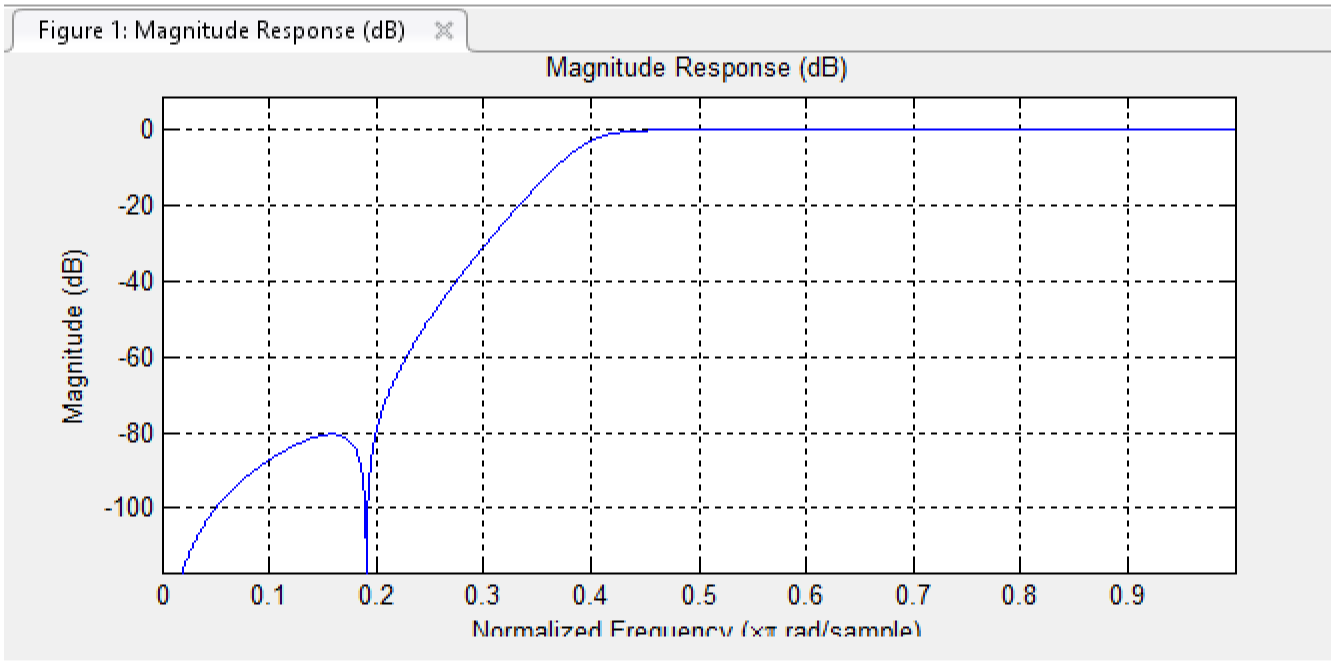
- Playback Mode Double Precision Implementation Limitation
    1. As the coefficients were of the type double, scaling and rounding them would yield erroneous results when tried implementing in C
    2. Several attempts were made to make it work and ultimately, found out that rounding the results to 7 points and scaling it by 2^24 was the modification (or manipulation) limit
    3. However, even this did not work
    4. The major problem was division of a negative number in C (The results would be unexpected, probably because of overflow).
    5. Hence, the code contains a check of whether the result to be divided is negative or not and if yeas, it is multiplied by -1, then the required division is performed and then, again the final result is multiplied by -1.
    6. This however was possible only because the sign of the divisor was known.
- To demonstrate that I have completely understood the concept, I used the function butter() and designed a filter in MATLAB (10th order)



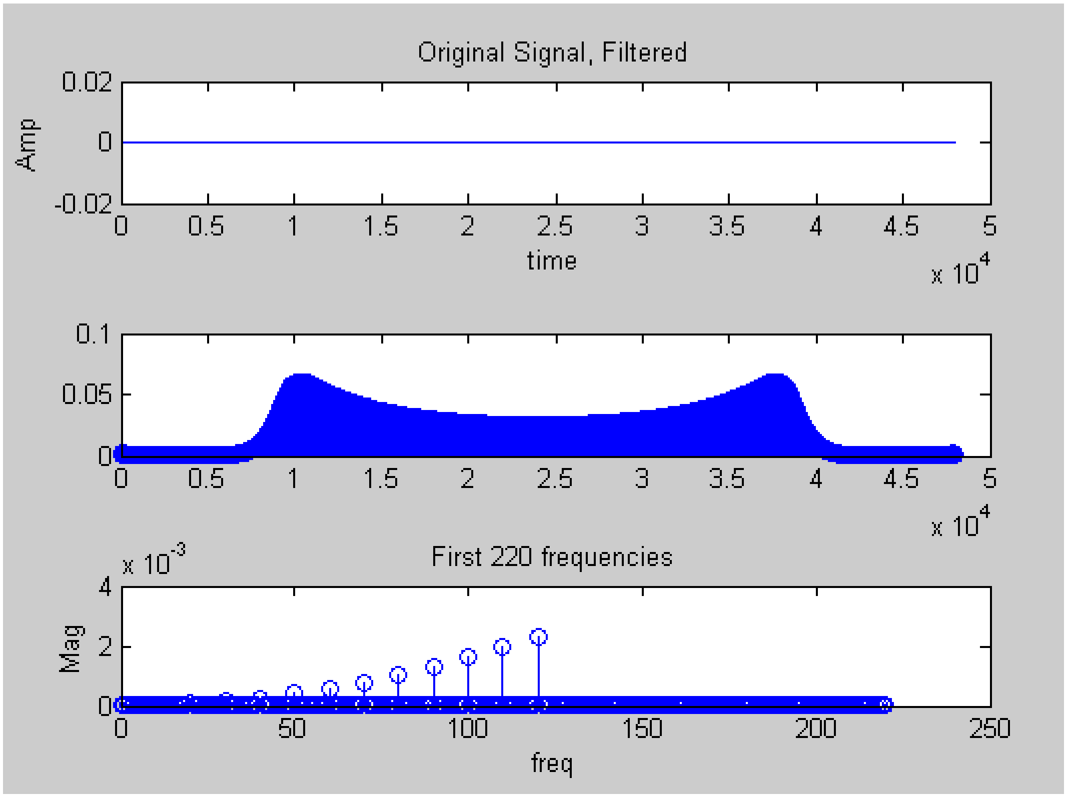
a (Den)										
1	-1.992401482	3.019482863	-2.818522426	2.038720637	-1.05454462	0.414446269	-0.115718625	0.022498509	-0.002668912	0.000148764
b (Num)										
0.012186673	-0.12186673	0.548400283	-1.462400755	2.559201321	-3.07104159	2.559201321	-1.462400755	0.548400283	-0.12186673	0.012186673



- However, those values were also of the type double and so on rounding them, the following results were obtained
- `a = [ 1.0000    -1.9924     3.0195    -2.8185     2.0387    -1.0545     0.4144    -0.1157     0.0225    -0.0027     0.0001];`
- `b = [0.0122    -0.1219     0.5484    -1.4624     2.5592    -3.0710     2.5592    -1.4624     0.5484    -0.1219     0.0122];`



- The following figure shows that the original signal when filtered using the above scaled values, yields acceptable results and hence, the same scaled values were used for filtering (C implementation)
- Please refer to file: Butter_filter.xlsx, EE5391_ButterHighPass.m



- <u>Playback Mode and Algorithm Flow:</u>
    1. Line in is enabled and the input signal is sampled at fs = 48kHZ
    2. This signal is continuously filtered
    3. Depending on the switch 1 position the signal is played as it is if switch is pressed, if the switch is not pressed the filtered signal is played(can be heard via speakers) .

Source Code:
Please find the source code in the file attached (EE5391_SourceCode.txt).

Concepts Learned:

- Fixed Point Arithmetic
- Audio file generation (Bass, Beats, Pitch, Amplification)
- Direct form I and Direct Form II filters
- Multi-Channel buffered serial communication
- Worked for the first time on a codec
- Implemented all the basic concepts learned in DSP class on the board

Concepts Cleared:

- Significance of Poles and zeros
- Frequency Analysis (DFT)
- FIR and IIR filters
- Notch Filters Analysis
- Resonance

Helped strengthen the following skills:

- Firmware development
- Embedded C

Additional Softwares used:

- MATLAB (also, FDAtool)
- Audacity
- Format Factory

Books referred to:

- Understanding Digital Signal Processing, by Richard G Lyons, 2nd Edition

Conclusion:
This project has helped me understand all the basic concepts of DSP, well, as well as working on the DSP processor has helped gain an experience that will help me in the future ‘firmware development for embedded systems’ projects.