SIGNALS AND SYSTEMS

Prepared By :

HUZAIFA ZAFAR

Presented To :

MA'AM ALEENA

Project Report

Project Details

Design and Implementation of a 5-band Graphic Equalizer

Part I

The goal of this project is to design a 10-band graphic equalizer and then to implement it - first employing Simulink to check the design and then designing a GUI. Matlab can be employed to design the required filters and then Simulink can be used to implement the graphic equalizer in real time. Most commercial equalizers use either 1/3 octave or 2/3 octave bandpass filters but to keep this from becoming too large we will employ one octave bandpass filters.

Following are the design specifications for the equalizer:

1. Employing Matlab, design 5 different bandpass filters with center frequencies of 63 Hz, 250 Hz, 1000 Hz, 4000 Hz, and 16000 Hz. These center frequencies correspond to the ISO (International Standards Organization) standard for graphic equalizer center frequencies.

2. The bandwidth of each filter is the frequency difference $\Delta f = f2 - f1$, where f1 and f2 correspond to the frequencies where the gain is 3 dB less than the maximum gain at the center frequency. It also is necessary to choose f1 and f2 such that the center frequency, fc, is equal to the geometric mean of f1 and f2, i.e. fc = (f1f2) 1/2 . We also have to choose the bandwidth of each filter so that we get a flat frequency response when all filter gains are equal and added together.

3. You can use Butterworth filters; however you are free to choose the order of the filters. The Matlab help file for the Butterworth filter is the following: [B,A] = butter(N,Wn) designs an Nth order lowpass digital Butterworth filter and returns the filter coefficients in length N+1 vectors B (numerator) and A (denominator). The coefficients are listed in descending powers of z. The cutoff frequency Wn must be 0.0 < Wn < 1.0, with 1.0 corresponding to half the sample rate. If Wn is a two-element vector, Wn = [W1 W2], butter returns an order 2N bandpass filter with passband W1 < W < W2. [B,A] = butter(N,Wn,'high') designs a highpass filter. [B,A] = butter(N,Wn,'low') designs a lowpass filter. [B,A] = butter(N,Wn,'stop') is a bandstop filter if Wn = [W1 W2].

4. Write a Matlab m-file to compute the set of filter coefficients and plot the combination (sum) of all filter frequency responses. Note that you can use the 'freqz' command to easily find the frequency response of a filter defined by the filter coefficient arrays B and A. Your goal is to achieve as flat of a frequency response as you can when all the frequency response of all filters are added ± 1 dB is a good goal. Remember that the center frequency of each filter must be fixed to one of the five values given above and the upper and lower cutoff frequencies f2 & f1 must satisfy fc = (f1f2) 1/2 . Your goal is to find the Δf value for each filter that achieves a flat frequency response when all filters are combined with equal weights. [Hint: the filters should all be constant Q, where Q = fc / (f2 – f1), so once you find the right value for Q all filters should have the same Q.]

The simplified 3-band graphic equalizer is shown below, where the filters are in parallel and each one is followed by a gain (using the Matlab slider gain block). Your mixer will have 5 filters in parallel.

We would like to be able to adjust the gain of each band by ± 12 dB. Remember that 6dB corresponds to approximately a factor of 2x, so 12 dB is about 4x. So +12 dB is like multiplying by 4 and -12 dB is like multiplying by ¼.Use these values as the limits for the slider gain blocks.

One final note: The “From Multimedia File” and “To Audio Device” blocks can be found in the DSP Toolbox.

Part II

Design a GUI with the following provisions

1. A ‘load’ button that can load an audio file of your choice (Suggestion: choose a file with large frequency range e.g a symphony).

2. Display for the input and output signals.

3. Display for the input and output spectrum.

4. Adjustable gain sliders for the filters.

5. A ‘play’ button that can playback the output file.

Introduction

The project aims to design and implement a 5-band graphic equalizer in two parts. In Part I, the focus is on crafting five bandpass filters with specific center frequencies, adhering to ISO standards. Matlab and Simulink are employed for design validation and real-time implementation. The use of Butterworth filters with adjustable orders is recommended. The goal is to achieve a flat frequency response when combining all filters with equal weights, ensuring constant Q. Part II involves designing a user-friendly GUI with features such as a 'load' button for audio file selection, input/output signal displays, adjustable gain sliders, and a 'play' button for playback, facilitating intuitive audio customization.

Objectives

1. Designing of Filters on Matlab
2. Implementation of those filters using Simulink
3. Creating a GUI for our Audio Equaliser

Filter Designing using MATLAB

In this project conducted using MATLAB, we aimed to design a 5-band graphic equalizer. The process involved specifying center frequencies for each band and determining a Q factor. As we have to design a single octave filter so we will be using a Q factor of √2. Utilizing MATLAB's Butterworth filter design, octave bandpass filters were created for each center frequency. The higher frequency is just double the lower frequency in each case. The code then systematically plots the individual frequency responses of each band and their combined response, providing a visual representation of the equalizer's behavior. Importantly, this MATLAB implementation serves as a crucial step before transitioning to Simulink. Once the filters are designed and optimized in MATLAB, they are exported to Simulink for real-time implementation. The individual functions in the code handle the calculation of lower and upper cutoff frequencies, the design of Butterworth bandpass filters, and the plotting of frequency responses. This MATLAB-to-Simulink workflow ensures a seamless integration of the designed filters into a real-time graphic equalizer system.

Output:

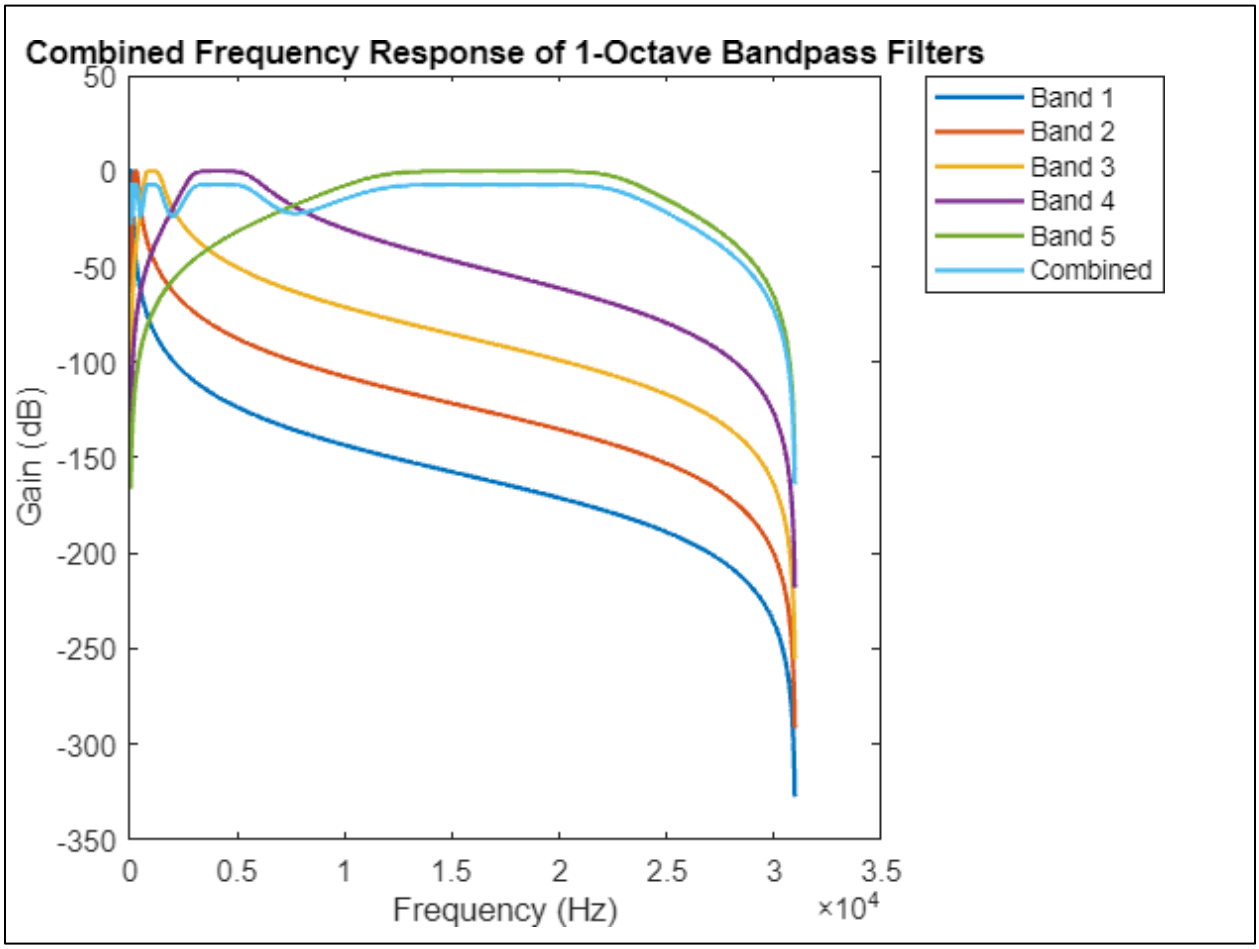


Figure 1: Combined Frequency Response of our 5 Band Filter

Values of Upper and Lower Frequencies Calculated and Printed by our Code:

```
Band 1 - Center Frequency: 63 Hz, Lower Frequency: 44.55 Hz, Upper Frequency:
89.10 Hz, Bandwidth: 44.55 Hz
Band 2 - Center Frequency: 250 Hz, Lower Frequency: 176.78 Hz, Upper
Frequency: 353.55 Hz, Bandwidth: 176.78 Hz
Band 3 - Center Frequency: 1000 Hz, Lower Frequency: 707.11 Hz, Upper
Frequency: 1414.21 Hz, Bandwidth: 707.11 Hz
Band 4 - Center Frequency: 4000 Hz, Lower Frequency: 2828.43 Hz, Upper
Frequency: 5656.85 Hz, Bandwidth: 2828.43 Hz
Band 5 - Center Frequency: 16000 Hz, Lower Frequency: 11313.71 Hz, Upper
Frequency: 22627.42 Hz, Bandwidth: 11313.71 Hz
```

Simulink Circuit and Filters

Simulink Schematic:

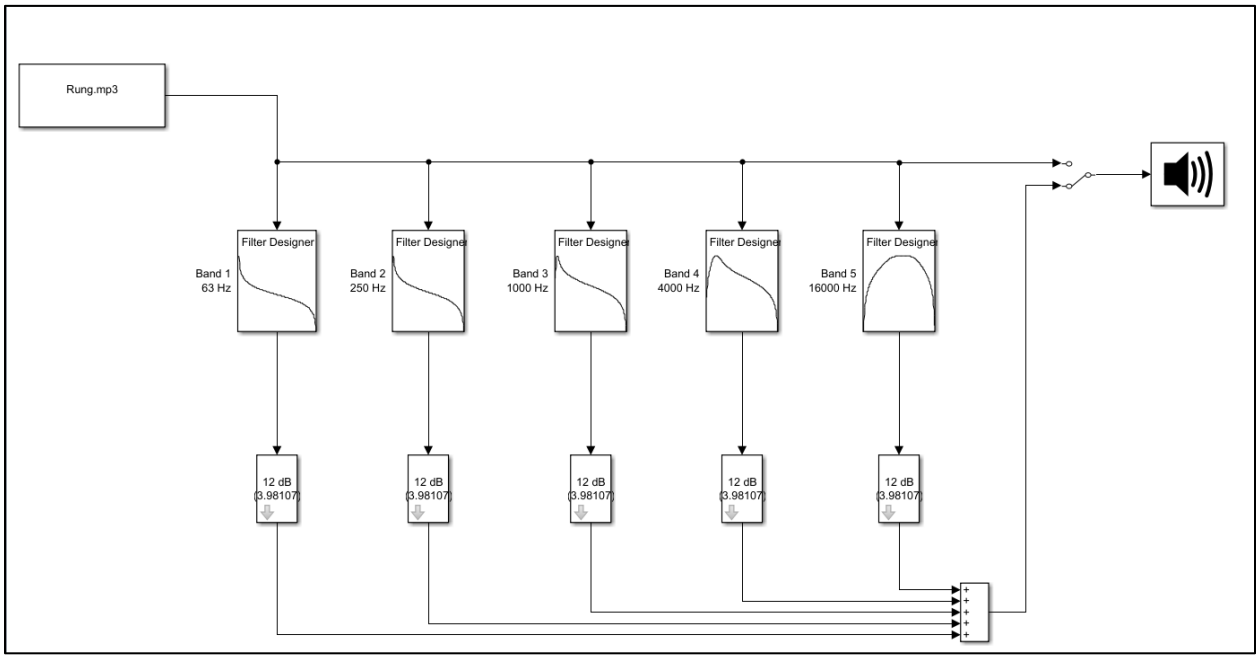
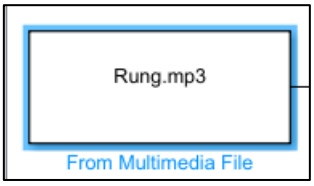


Figure 2: Simulink Schematic of our Audio Equaliser

We have used the following blocks in our Simulink Schematic:

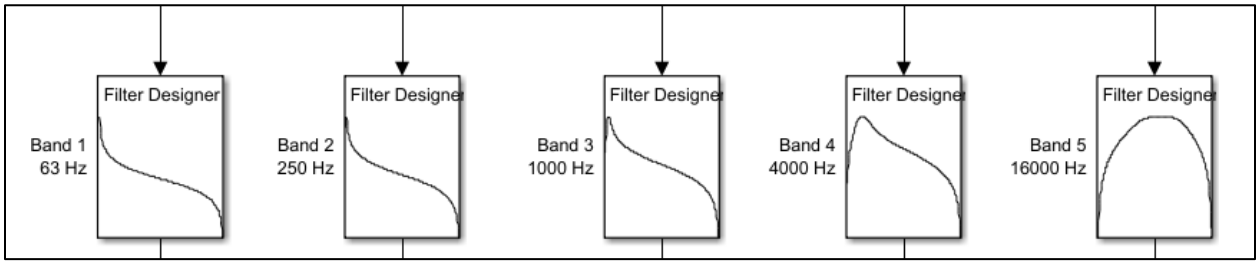
1. **From Multimedia File Block:**

The "From Multimedia File" block in Simulink is a vital tool for importing audio data into simulations. It serves as an input interface, facilitating the seamless integration of external audio files into Simulink models. This block is essential for testing and analyzing systems involving audio processing within the Simulink environment, enhancing the versatility of modeling and simulation tasks. It is a part of the DSP Toolbox



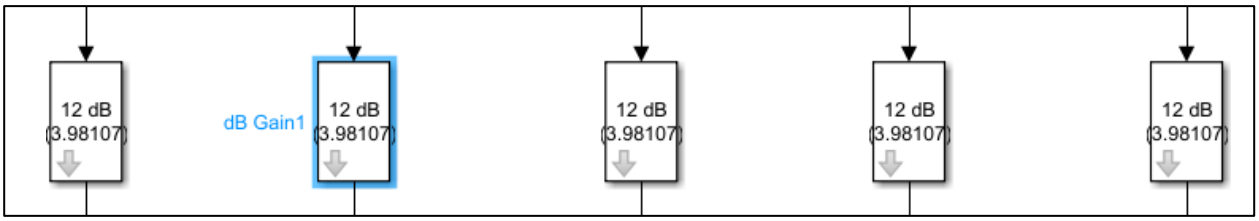
2. **Filter Designer Block:**

The "Filter Designer" block in Simulink is a user-friendly interface for creating and customizing digital filters. It enables real-time adjustments of filter specifications, including type, order, and cutoff frequencies.

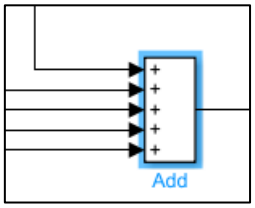


Supporting various filter designs, such as lowpass and highpass, this block is essential for efficient digital filter design within Simulink simulations.

3. **dB Gain Block:**

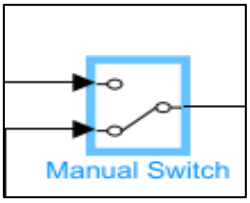


The "dB Gain" block in Simulink is a crucial component for adjusting signal gain in decibels. This block simplifies the process of scaling signal amplitudes, providing an efficient way to apply logarithmic gain adjustments within Simulink simulations.



4. **Add Block:**

The "Add" block in Simulink is a core element for mathematical operations in a model. It enables the summation of multiple input signals, providing a versatile tool for combining and manipulating signals within a simulation. The block enhances the modularity and flexibility of Simulink models, allowing users to perform additive operations and create complex signal processing or control system models efficiently.



5. **Manual Switch Block:**

The "Manual Switch" block in Simulink enables users to manually toggle between multiple input signals, controlling signal flow within a model. This block is valuable for scenarios requiring dynamic signal routing or conditional switching. By interacting with the "Manual Switch," users can efficiently simulate different scenarios and test the impact of various signal pathways in their models, enhancing adaptability and versatility in Simulink simulations.

6. **Audio Device Writer Block:**

The "Audio Device Writer" block in Simulink facilitates audio output in simulation models. It allows users to stream simulated audio signals directly to audio devices for real-time auditory feedback during simulations. By connecting this block to the desired audio source, users can integrate and test audio processing algorithms or control systems with ease. The "Audio Device Writer" enhances Simulink simulations by providing a practical way to evaluate the real-world impact of models through audible output.

GUI Interface Images

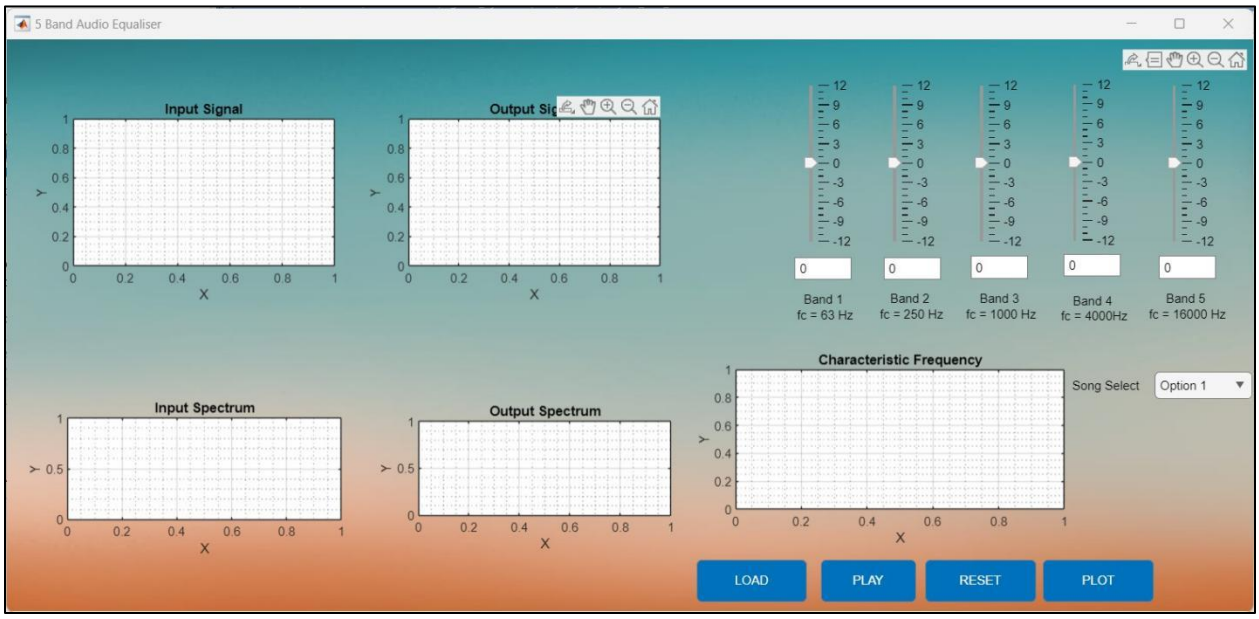


Figure 8: GUI Interface

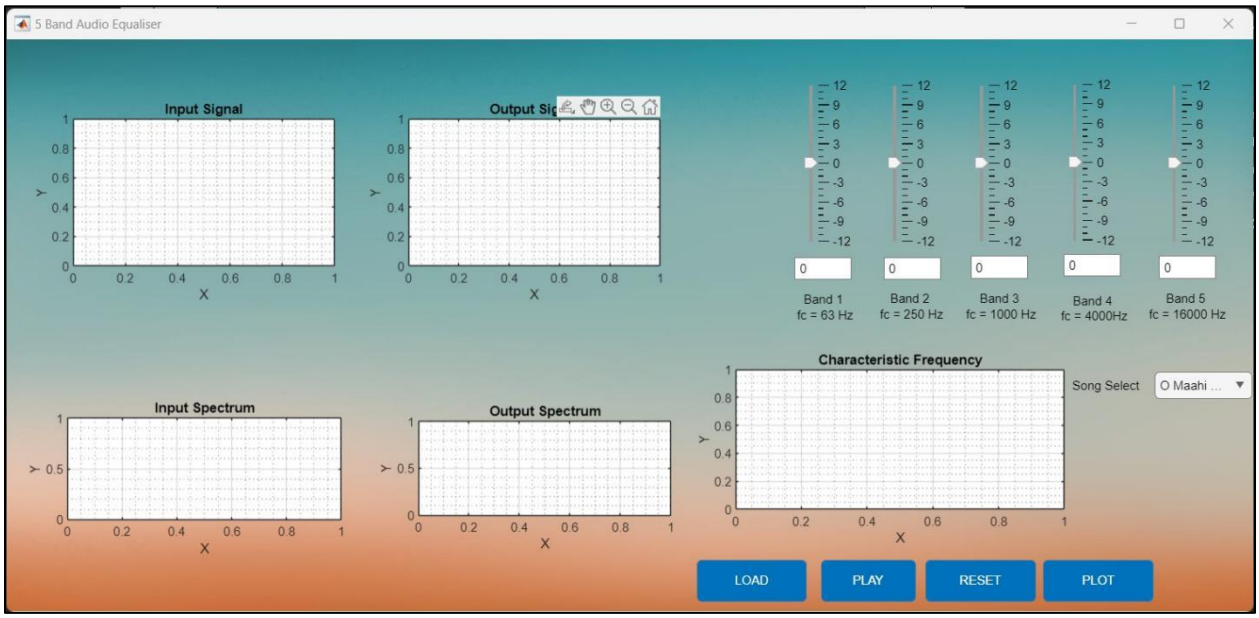


Figure 9: GUI after the Song has been loaded by pressing 'LOAD' button.

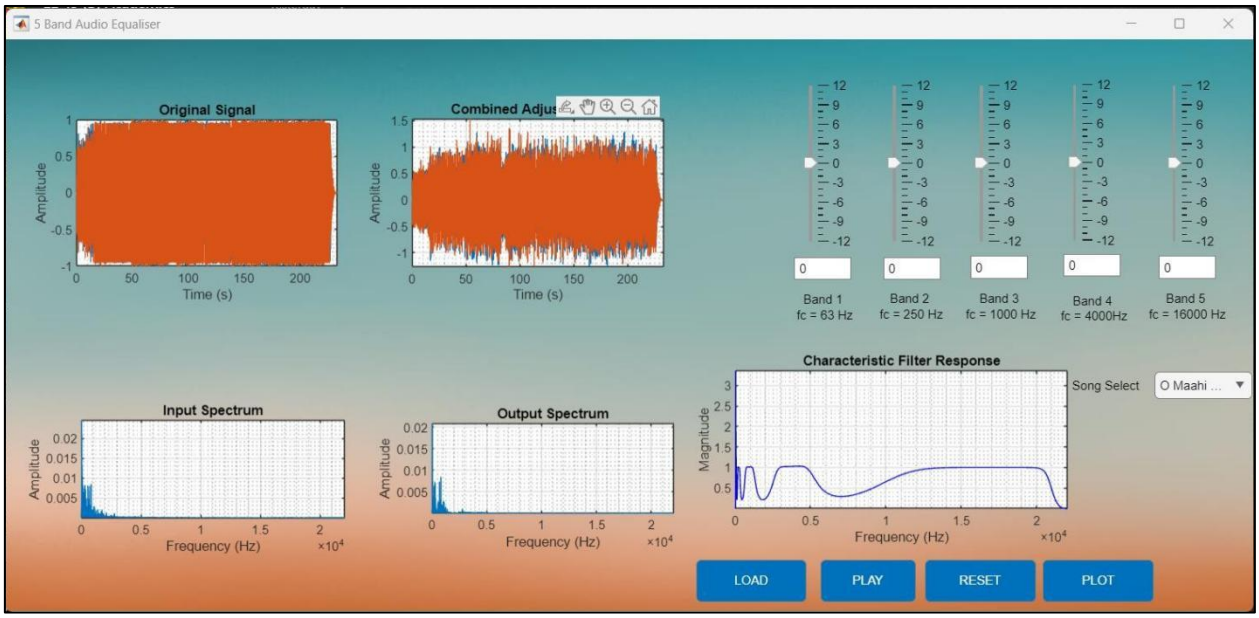


Figure 10: Signal Plotted without any gain

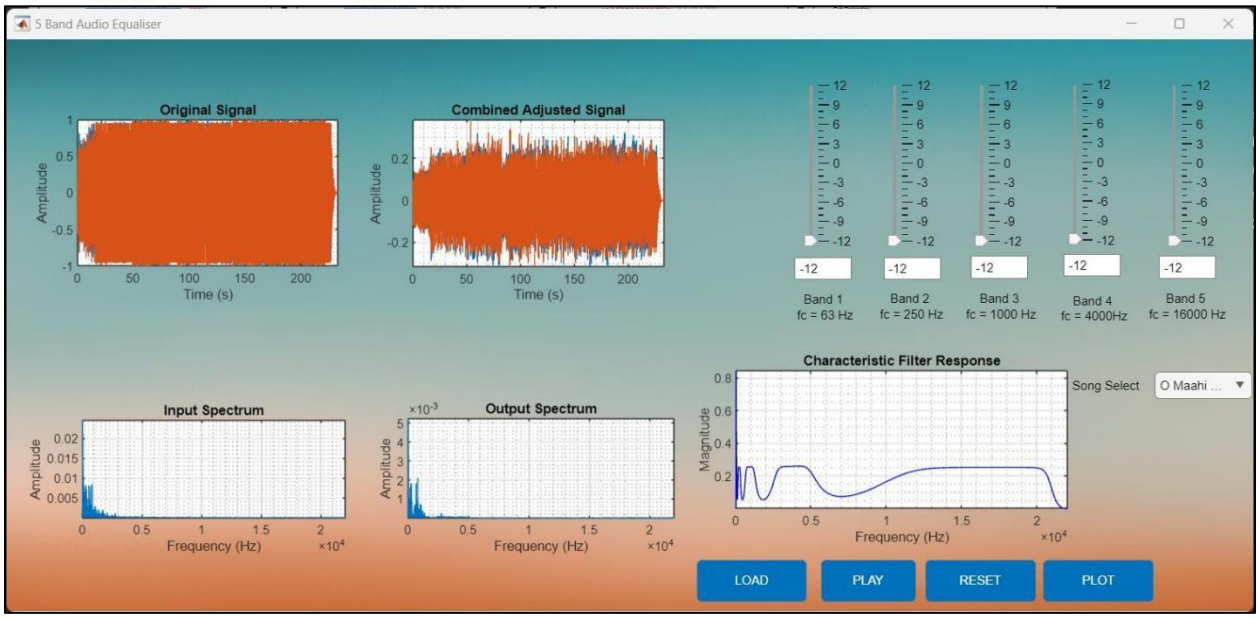


Figure 11: Signal plotted with full negative gain

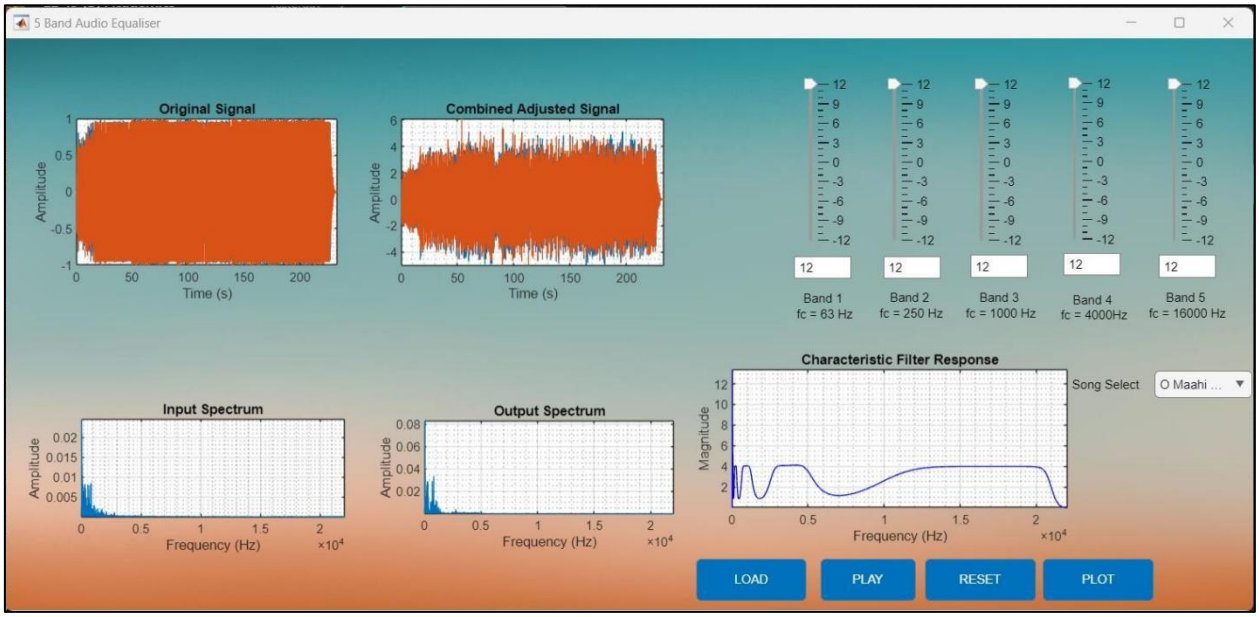


Figure 12: Signal plotted with full positive gain

Conclusion

In summary, this project merged theoretical concepts from our "Signals and Systems" course with practical applications. The first part focused on analyzing audio signals, providing insights into temporal and frequency characteristics. The second part introduced a user-friendly 5-band audio equalizer through a graphical interface, showcasing real-time signal manipulation.

This project bridged theory and practice, emphasizing the application of signal processing techniques like Fourier analysis and equalization. The graphical interface highlighted the importance of user-friendly tools in signal processing. The hands-on experience reinforced MATLAB programming and signal processing skills while deepening our understanding of audio quality's dependence on signal manipulation.

Overall, this project was a valuable exercise, enriching our grasp of signals and systems in the context of audio engineering.

References

1. https://www.diva-portal.org/smash/get/diva2:1334188/FULLTEXT02.pdf
2. https://www.mathworks.com/matlabcentral/fileexchange/23982-digital-audio-equalizer
3. https://www.youtube.com/watch?v=Z7urwX82Z8g
4. https://www.mathworks.com/help/audio/ug/equalization.html
5. https://www.mathworks.com/help/audio/ug/graphic-equalization.html
6. https://www.mathworks.com/help/audio/audio-processing-algorithm-design.html
7. https://github.com/splAcharya/AudioEqualizerMatlab_Simulink
8. https://github.com/MonicaSaid/audio-equalizer
9. https://github.com/mohamedmashaal/Digital-Audio-Equalizer/tree/master