AUDIO EQUALIZER USING PASS FILTERS

PROJECT DESCRIPTION:

We have built an audio equalizer using a low-pass filter, a high pass filter and a band pass filter. the audio equalizer can be connected to mp3 player, phone, or laptop any device having a standard audio jack and play filtered music through an amplifier and speaker.

OBJECTIVE:

The objective/aim of the project is to build an audio equalizer using pass filters through which we can modify audio characteristics such as bass, treble and other modifications.

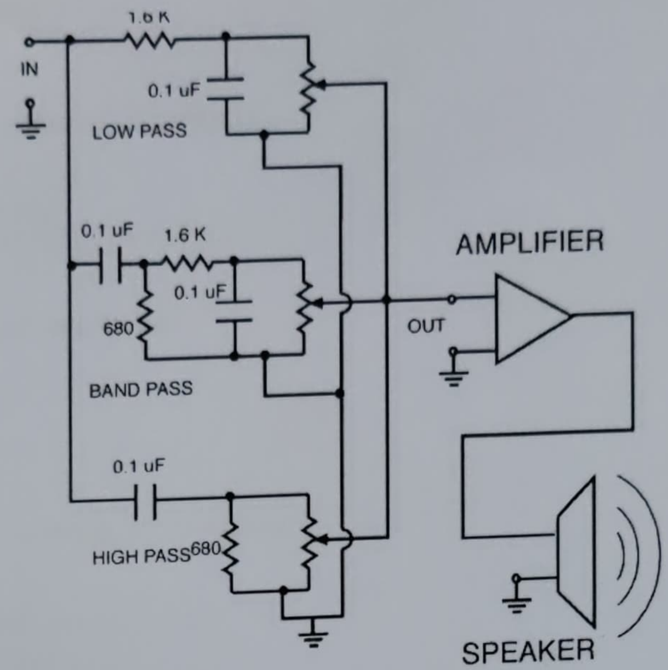
INTRODUCTION:

- **LOW PASS FILTER:** In low pass filter transmission (fraction of signal passed) a function of frequency is plotted. This is called the "transfer function" of a circuit. Low frequencies are completely passed (fraction=1), but high frequencies are not passed. In the middle is a transition range.
  The transition may be slow or abrupt, depending on the filter design. Ideally the cutoff would be very sharp, but the sharper the curve the more electrical compon ents are required to implement it.

- **HIGH PASS FILTER:** The transfer function of a high-pass filter. We can infer that low frequencies are in the stop band, high frequencies are in the pass band. and there is still a cutoff frequency.
  The first part of the signal has zero slope, and the output of the filter is zero. Wh en the slope increases suddenly, the output likewise goes to a high positive value When the signal degrades slowly, that is where t has a small, negative slope, the filter output goes to a small negative value.

- **BAND PASS FILTER:** Finally, a bandpass filter's transfer function we can infer that it is a combination of this can be of a low pass filter and a high pass filter.

CIRCUIT DIAGRAM:



CODE:

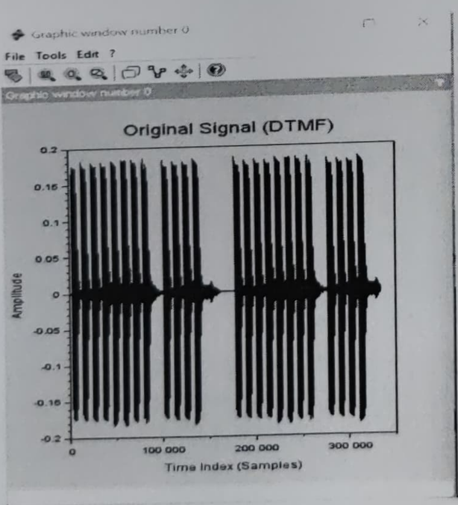
```
// DSP Miniproject
// Audio Equalizer
// 210, 211, 216
clc;
close;
clear;
function [xm1, fr1]=low(fl, fh)
delta1=0.1;
delta2=0.1;
fs=8000;
A=-20*log10(min(delta1:delta2));
w1=2*3.14*fl/fs;
w2=2*3.14*fh/fs;
temp=1+((A-8)/(2.285*((2*3.14*fh/fs)-(2*3.14*fl/fs))));
N=ceil((temp-1)/2);
n=-N:N;
h=(((w2+w1)/2)*(sinc(((w2+w1)/2)*n)))/(3.14);
[xm1,fr1]=frmag(h,8000);
endfunction
function [xmh, frh]=highpass(fl, fh)
```

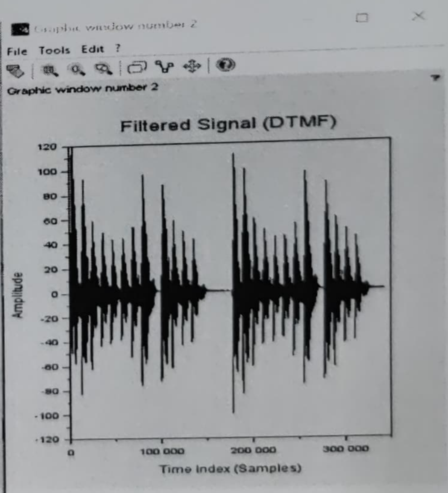
```
delta1=0.1;
delta2=0.1;
fs=8000;
A=-20*log10(min(delta1:delta2));
w1=2*3.14*fl/fs;

w2=2*3.14*fh/fs;
temp=1+((A-8)/(2.285*((2*3.14*fh/fs)-(2*3.14*fl/fs))));
N=ceil((temp-1)/2);
n=-N:N;
del=[zeros(1:N) 1 zeros(N+1:2*N)];
h=del-(((w2+w1)/2)*(sinc(((w2+w1)/2)*n)))/(3.14);
[xmh,frh]=frmag(h,8000);
endfunction
function [xmb, frb]=bandpass(fl1, fh1, fl2, fh2)
delta1=0.1;
delta2=0.1;
fs=8000;
A=-20*log10(min(delta1:delta2));
w12=2*3.14*fl2/fs;
w22=2*3.14*fh2/fs;
temp2=1+((A-8)/(2.285*((2*3.14*fh2/fs)-(2*3.14*fl2/fs))));
N=ceil((temp2-1)/2);
n=-N:N;
h2=(((w22+w12)/2)*(sinc(((w22+w12)/2)*n)))/(3.14);
delta1=0.1;
delta2=0.1;
fs=8000;
A=-20*log10(min(delta1:delta2));
w11=2*3.14*fl1/fs;
w21=2*3.14*fh1/fs;
temp1=1+((A-8)/(2.285*((2*3.14*fh1/fs)-(2*3.14*fl1/fs))));
N=ceil((temp1-1)/2);
n=-N:N;
h1=(((w21+w11)/2)*(sinc(((w21+w11)/2)*n)))/(3.14);
h=h2-h1;
[xmb,frb]=frmag(h,8000);
endfunction
/////////////////////Main Program/////////////
[y,fs]=wavread("C:\Users\Abhinav\OneDrive\Documents\scilab\audio.wav"); // User can read any wav file
y = matrix(y,1,-1)
//////////// LOW PASS ///////////////////////
fl=60; // Lower Cut-Off Frequency
fh=250; // Higher Cut-Off Frequency
[xm1,fr1]=low(fl,fh); // Function for Low Pass Filter
///////////////// HIGH PASS //////////////////
fl=2048; // Lower Cut-Off Frequency
fh=16384; // Higher Cut-Off Frequency
[xmh,frh]=highpass(fl,fh); // Function for high Pass Filter
//////////////////// BAND PASS //////////////////
```

```
fl1=250; // Lower Cut-Off Frequency1
fh1=350; // Higher Cut-Off Frequency1
fl2=1000; // Lower Cut-Off Frequency2
fh2=1100; // Higher Cut-Off Frequency2
[xmb,frb]=bandpass(fl1,fh1,fl2,fh2);
///////////////// Function for Band Pass Filter////////////////////
gain_L=2; // Gain for Low Pass Frequency
gain_B=5; // Gain for band Pass Frequency
gain_H=2; // Gain for High Pass Frequency
sig_L=conv(y*gain_L,xm1);
sig_B=conv(y*gain_B,xmb);

sig_H=conv(y*gain_H,xmh);
sig_T=sig_L+sig_H+sig_B;
figure;
plot(y);
title('Original Signal (DTMF)','color','red','fontsize', 4);
xlabel("Time Index (Samples)", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
figure;
plot(sig_T);
title('Filtered Signal (DTMF)','color','red','fontsize', 4);
xlabel("Time Index (Samples)", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
playsnd(sig_T,fs,1);
playsnd(y,fs,1);
```

OUTPUT:

ORIGINAL SIGNAL (DTMF) **VS** **FILTERED SIGNAL (DTMF)**





CONCLUSION:
Thus, we have successfully created an AUDIO EQUALIZER using PASS FILTERS in SCILAB environment.

RESULT:
Thus, program was executed successfully in SCILAB environment.