2019

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH

DUBLIN

TECHNOLOGICAL UNIVERSITY DUBLIN

# AUDIO EFFECTS SYSTEM

DSP LAB PROJECT BASED ON AUDIO EFFECTS SYSTEM

D18124645 TALHA TALLAT

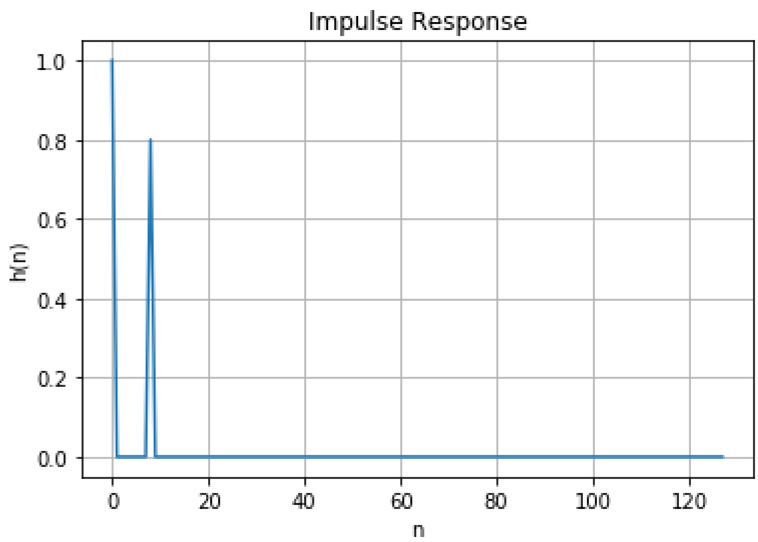
TECHNOLOGICAL UNIVERSITY DUBLIN | [Company address]

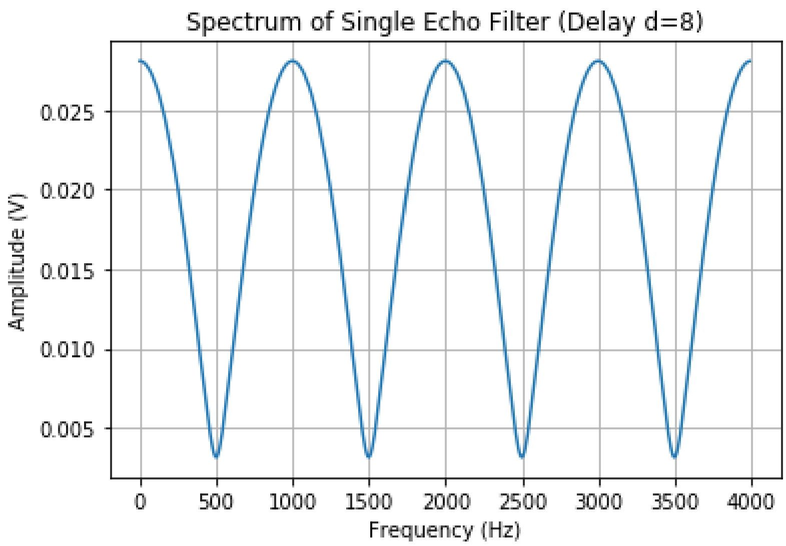
AUDIO EFFECTS SYSTEM

(1) Enter the code below into Python:

```
import numpy as np
import scipy.fftpack
from scipy.signal import lfilter
import matplotlib.pyplot as plt
from scipy.io import wavfile
# ---------------------creating an input signal---------------------
plt.close("all")
fs=8000
b=np.array([1,0,0,0,0,0,0,0,0.8])
a,N=1,128
ip=np.zeros(N);
ip[0]=1; #apply impulse to input
op=lfilter(b,a,ip)
plt.plot(op)
plt.grid()
plt.figure(1)
plt.title('Impulse Response')
plt.xlabel('n')
plt.ylabel('h(n)')
plt.show()

NFFT=1024 # No. of values in FFT
M = 2*np.abs(scipy.fftpack.fft(op,NFFT))/N
M = M[0:int(NFFT/2)] #slicing operation to avoid mirroring
freq = np.arange(0,NFFT/2) #frequency vector
freq = freq*fs/NFFT
plt.figure(2)
plt.plot(freq,M)
plt.title('Spectrum of Single Echo Filter (Delay d=8)')
plt.xlabel('Frequency (Hz)')
plt.ylabel('Amplitude (V)')
plt.grid()
```

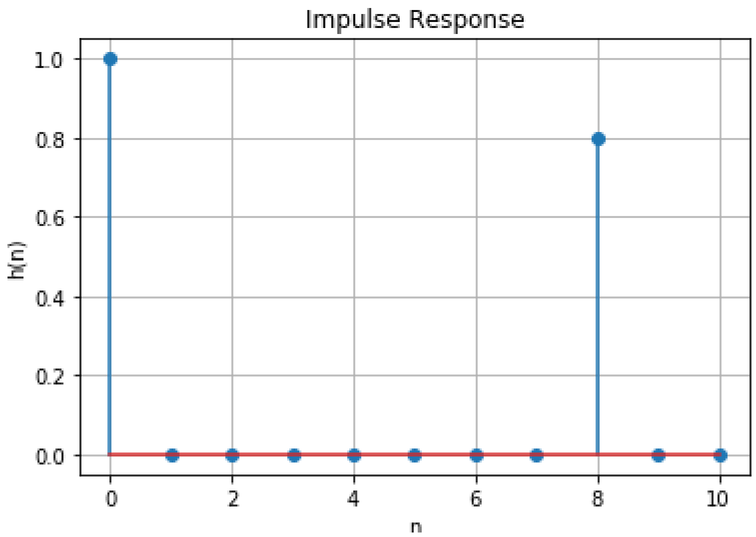
```
plt.show()
```




(2) Verify that the above impulse response and frequency response are obtained. Use the plt.stem function to plot the impulse response for 11 samples of the output.

```
plt.stem(op[0:11])
plt.grid()
plt.figure(1)
plt.title('Impulse Response')
plt.xlabel('n')
plt.ylabel('h(n)')
plt.show()
```




(2) Verify that the above impulse response and frequency response are obtained. Now modify the code to give a delay of 16 samples and verify the following results:

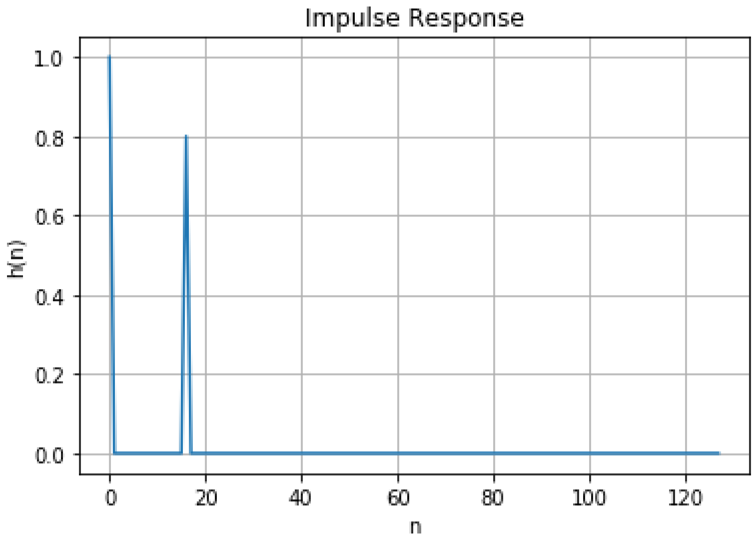
```
b=np.array([1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0.8])
a,N=1,128
ip=np.zeros(N);
ip[0]=1; #apply impulse to input
op=lfilter(b,a,ip)
plt.plot(op)
plt.grid()
plt.title('Impulse Response')
plt.xlabel('n')
plt.ylabel('h(n)')
plt.show()

NFFT=1024 # No. of values in FFT
M = 2*np.abs(scipy.fftpack.fft(op,NFFT))/N
M = M[0:int(NFFT/2)] #slicing operation to avoid mirroring
freq = np.arange(0,NFFT/2) #frequency vector
freq = freq*fs/NFFT
plt.plot(freq,M)
plt.title('Spectrum of Single Echo Filter (Delay d=8)')
plt.xlabel('Frequency (Hz)')
plt.ylabel('Amplitude (V)')
plt.grid()
plt.show()
```

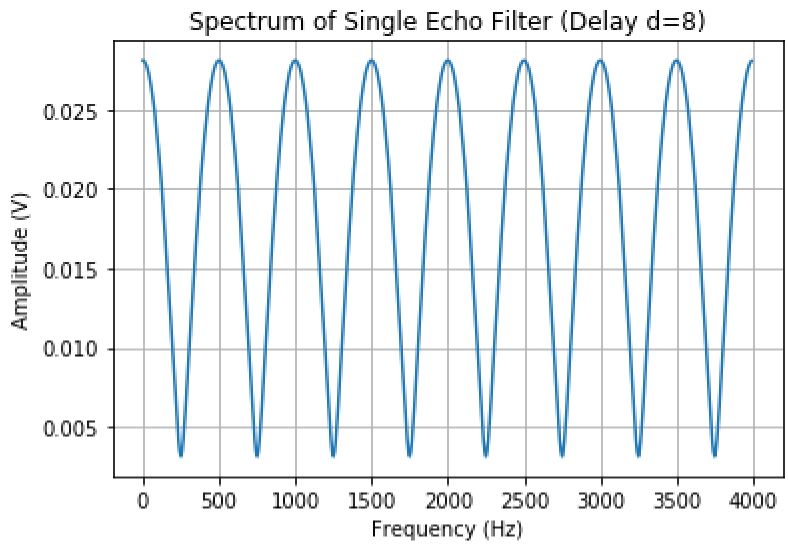
23 samples

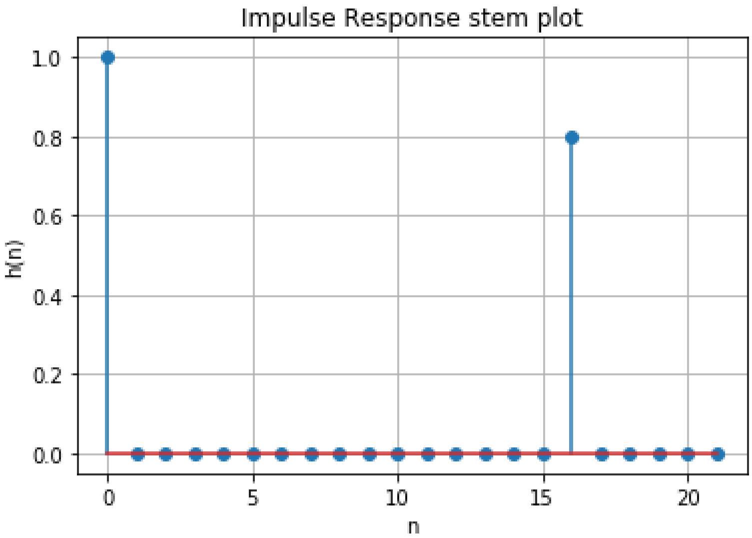
```
plt.stem(op[0:22])
plt.grid()
plt.title('Impulse Response stem plot')
```

```
plt.xlabel('n')
plt.ylabel('h(n)')
plt.show()
```

(3) Now modify the code to give a delay of 16 samples and verify the following results:






16 samples

(5) Show by a derivation that the peaks in the frequency response occur at multiples of 2π/d where d represents the number of samples that the signal has been delayed. Give a brief discussion of the results so far. Why this filter is called a Comb Filter?

$$H(\theta) = 1 + be^{-j\theta d} = 1 + b\{\cos(\theta d) - j\sin(\theta d)\}$$

$$|H(\theta)|^2 = \{1 + b\cos(\theta d)\}^2 + \{b\sin(\theta d)\}^2 = 1 + 2b\cos(\theta d) + b^2\{\cos^2(\theta d) + \sin^2(\theta d)\}$$

$$|H(\theta)|^2 = 1 + 2b\cos(\theta d) + b^2$$

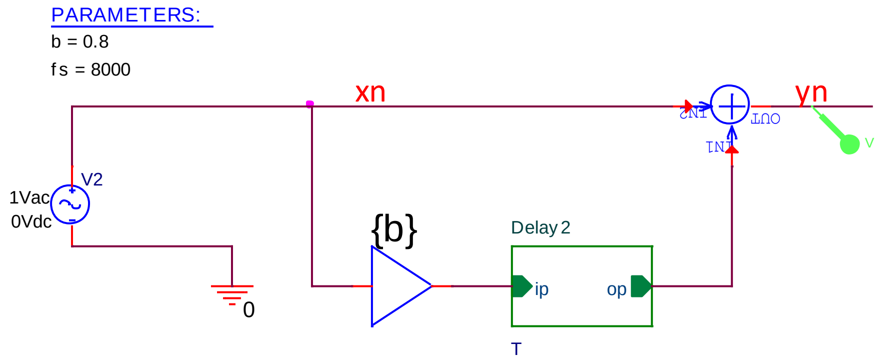
$$|H(\theta)|^2_{\max} = 1 + 2b + b^2 = (1+b)^2 \Rightarrow |H(\theta)|_{\max} = (1+b)$$

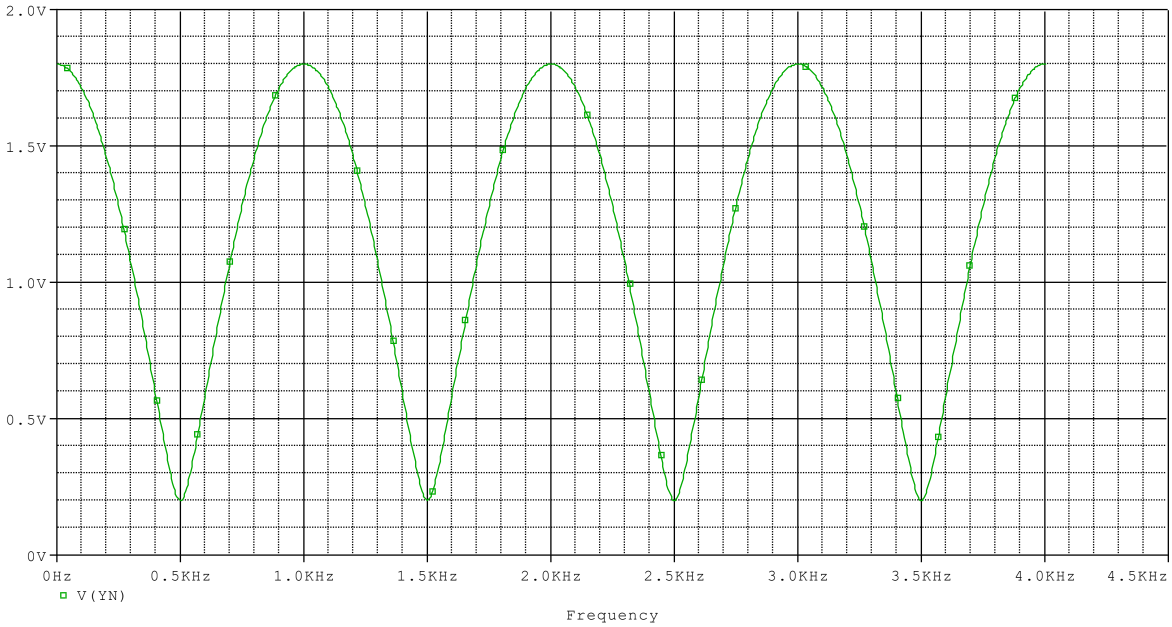
$$|H(\theta)|^2_{\min} = 1 - 2b + b^2 = (1-b)^2 \Rightarrow |H(\theta)|_{\min} = (1-b)$$

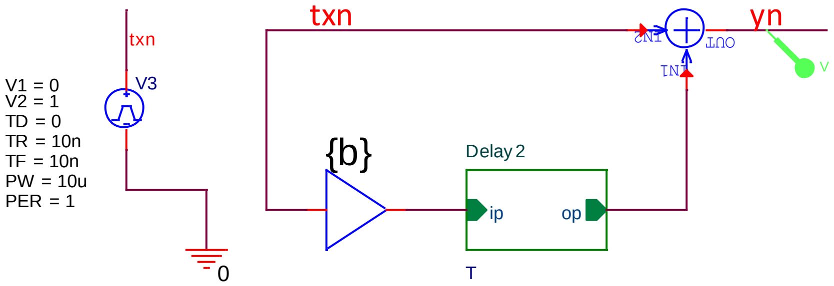
$$|H(\theta)|_{\max} \ @ \ \cos(\theta d) = 1 \ \Rightarrow \ \theta d = 2n\pi \Rightarrow \theta = \frac{2n\pi}{d}$$

$$If\ d = 8, \theta = \frac{2n\pi}{8} = \frac{n\pi}{4} = n \times 1\ kHz\ \ for\ f_s = 8\ kHz$$

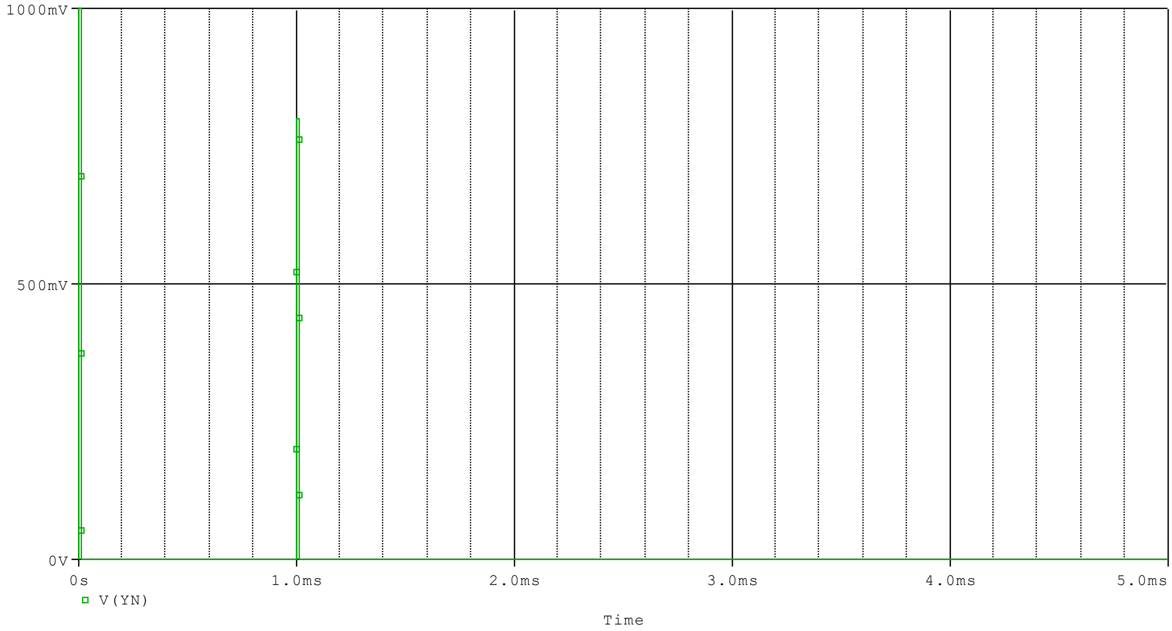
(4) Now investigate the implementation of both above filters in Pspice. Compare the results obtained with the Python system.


PARAMETERS:
b = 0.8
fs = 8000
xn
yn
V2
1Vac
0Vdc
0
{b}
Delay 2
ip
op
T
V

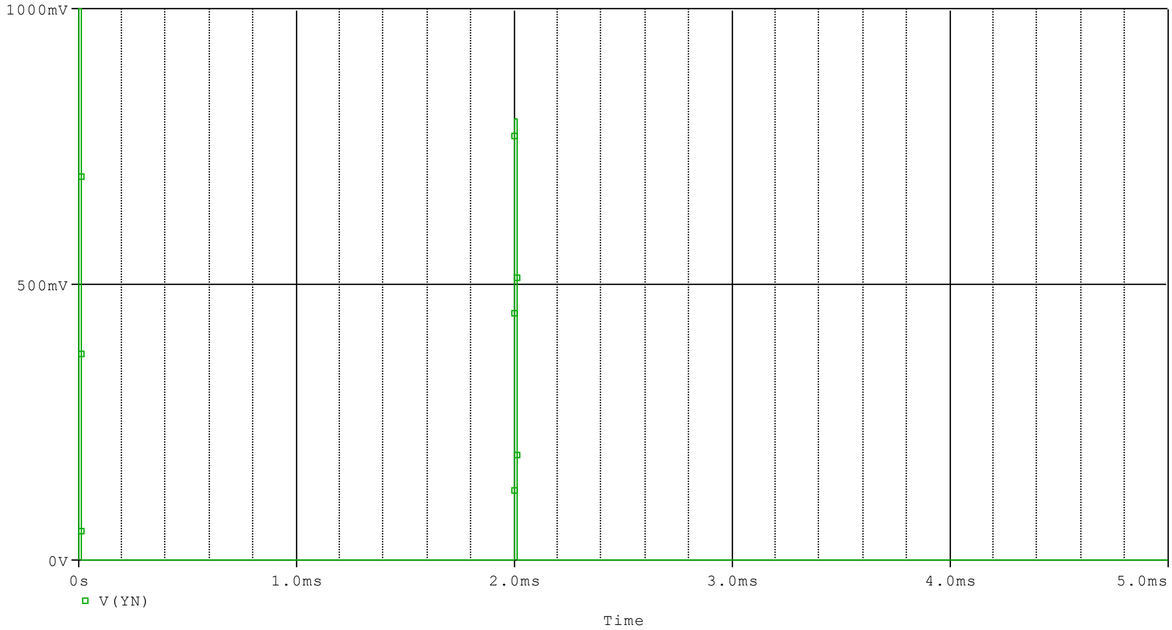

2.0V
1.5V
1.0V
0.5V
0V
0Hz
0.5KHz
1.0KHz
1.5KHz
2.0KHz
2.5KHz
3.0KHz
3.5KHz
4.0KHz
4.5KHz
V(YN)
Frequency


txn
V3
V1 = 0
V2 = 1
TD = 0
TR = 10n
TF = 10n
PW = 10u
PER = 1
0
txn
yn
{b}
Delay 2
ip
op
T
V

TD =8/fs

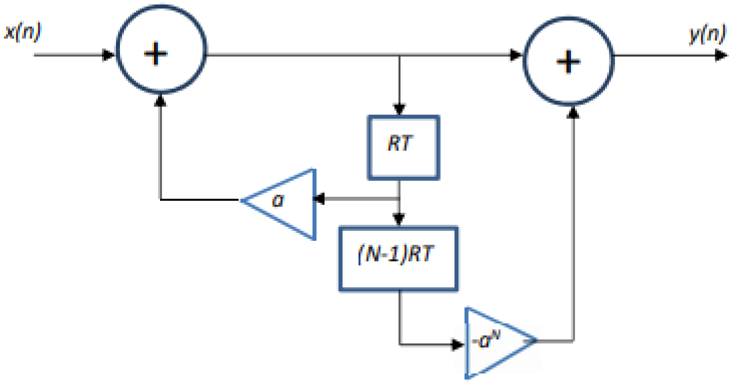



TD =16/fs



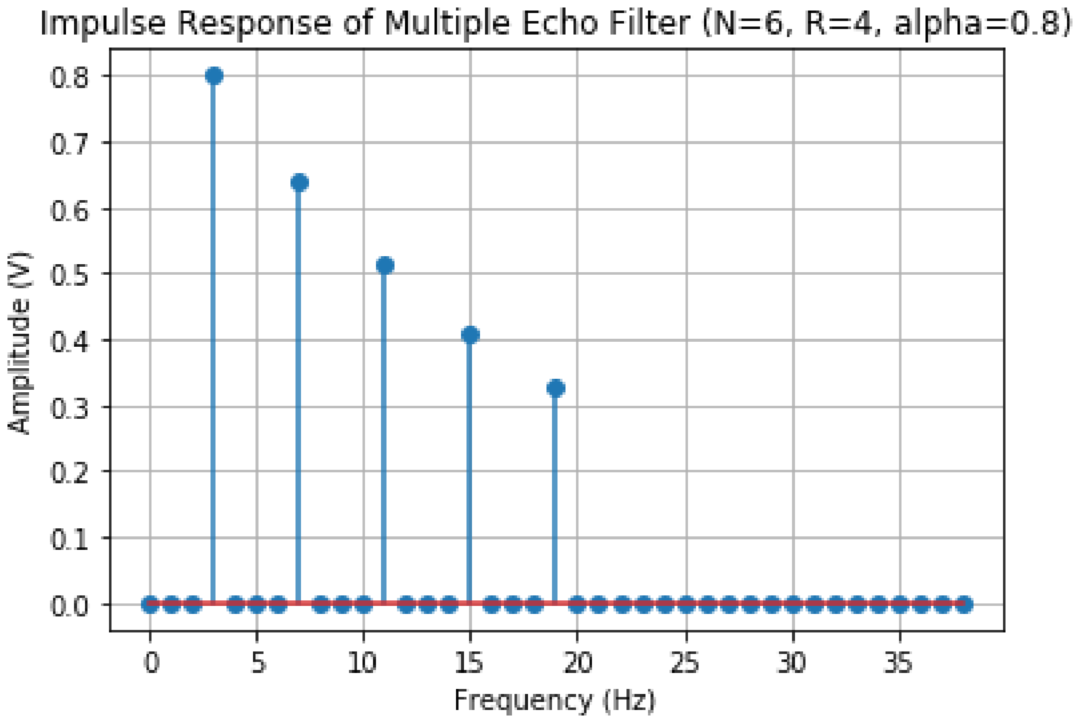
**DSP Lab Project (Audio Effects System) Part (II)**

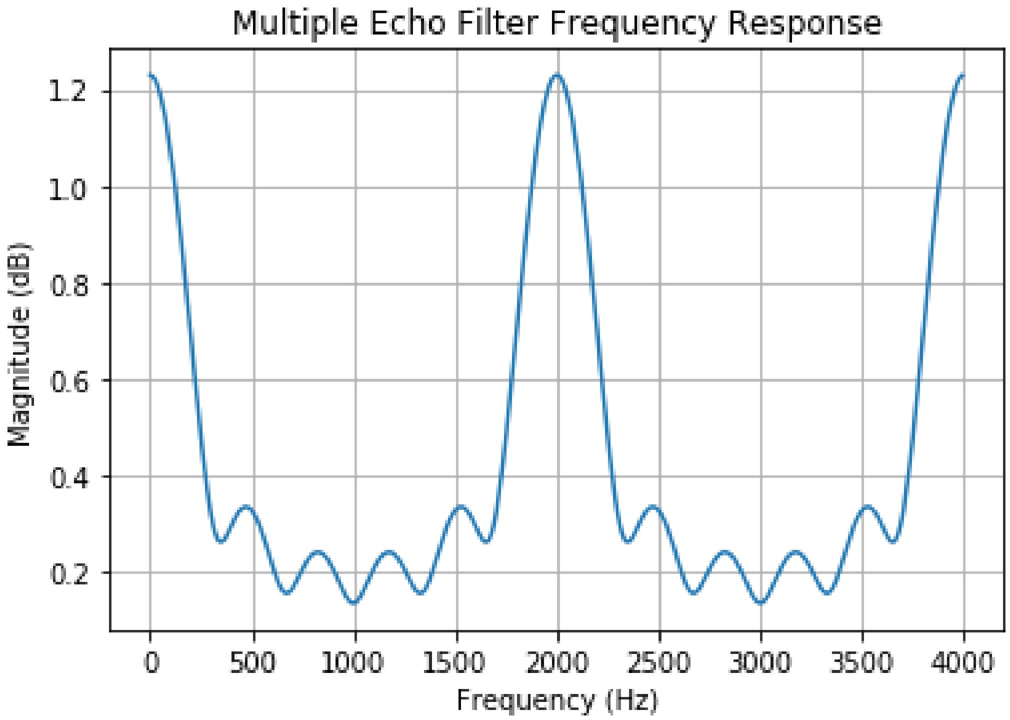
(1) Implement the multiple echo filter shown below in Python:




$$H(z) = \frac{1-\alpha^N z^{-NR}}{1-\alpha z^{-R}}$$

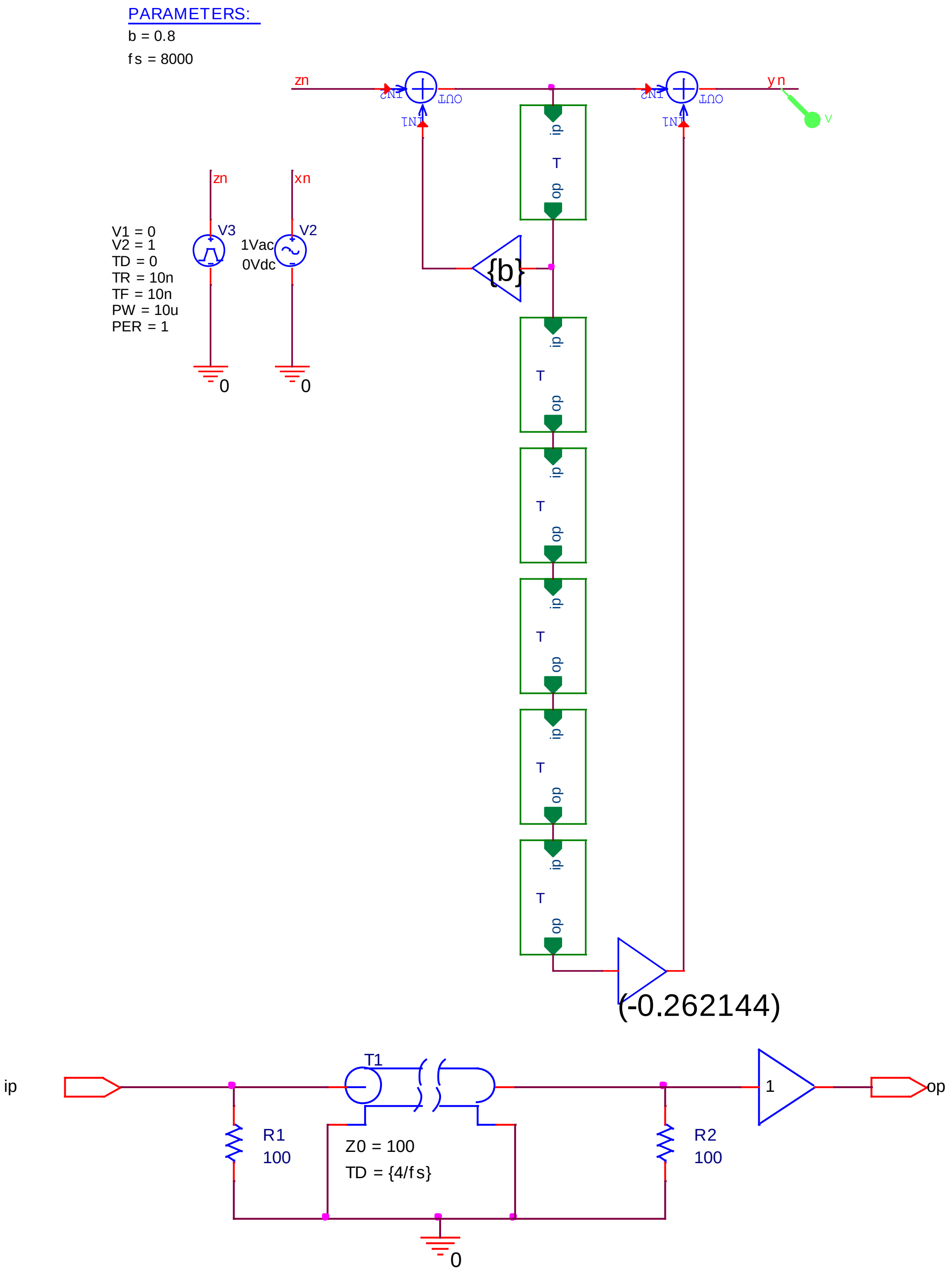
```
N=6; R=4; alpha=0.8;
b=np.zeros(N*R+1);
b[N*R]=-alpha**N; b[0]=1;
a=np.zeros(R+1);
a[R]=-alpha; a[0]=1;
ip=np.zeros(1024);
ip[0]=1;
op=lfilter(b,a,ip);
plt.stem(op[1:40])
plt.title('Impulse Response of Multiple Echo Filter (N=6, R=4, alpha=0.8)')
plt.xlabel('Frequency (Hz)')
plt.ylabel('Amplitude (V)')
plt.grid()
plt.show()
NFFT=1024 # No. of values in FFT
M = 2*np.abs(scipy.fftpack.fft(op,NFFT))/N
M = M[0:int(NFFT/2)] #slicing operation to avoid mirroring
freq = np.arange(0,NFFT/2) #frequency vector
freq = freq*fs/NFFT
plt.plot(freq,M)
plt.title ('Multiple Echo Filter Frequency Response');
plt.xlabel ('Frequency (Hz)');
plt.ylabel ('Magnitude (dB)');
plt.grid()
plt.show()
```

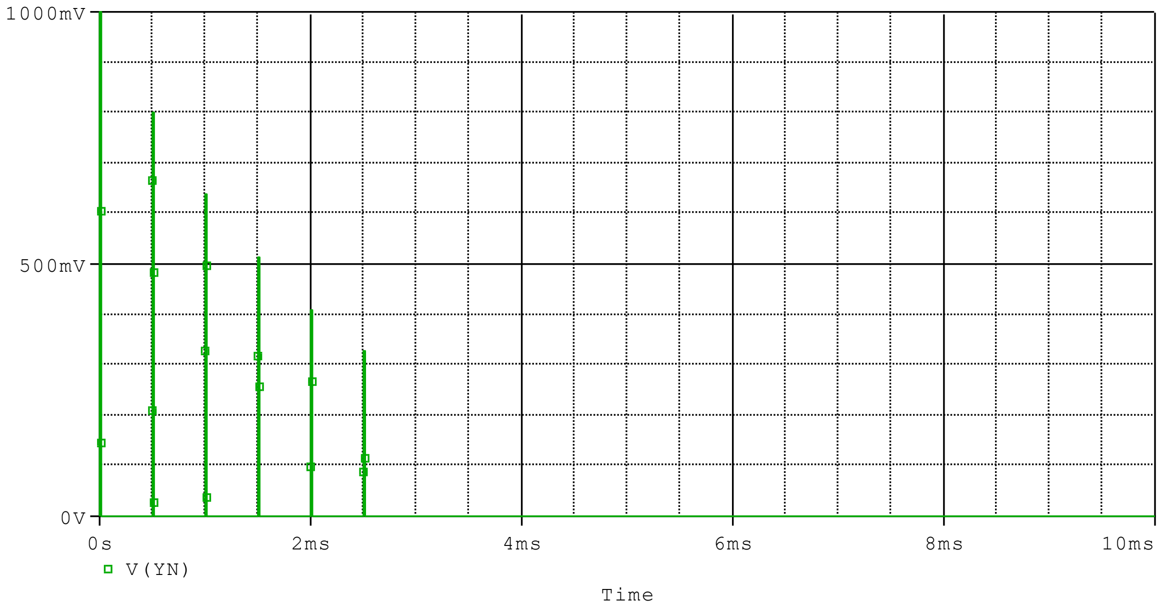


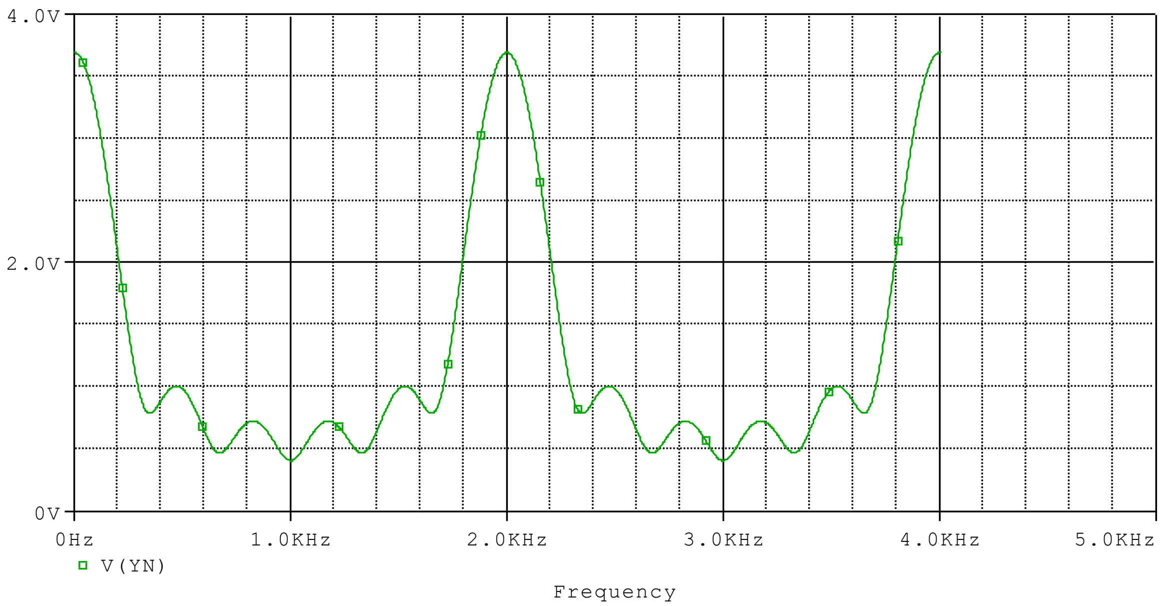

Provide a brief discussion of your results obtained in part 1?

(2) Now implement the multiple echo filter in Pspice.





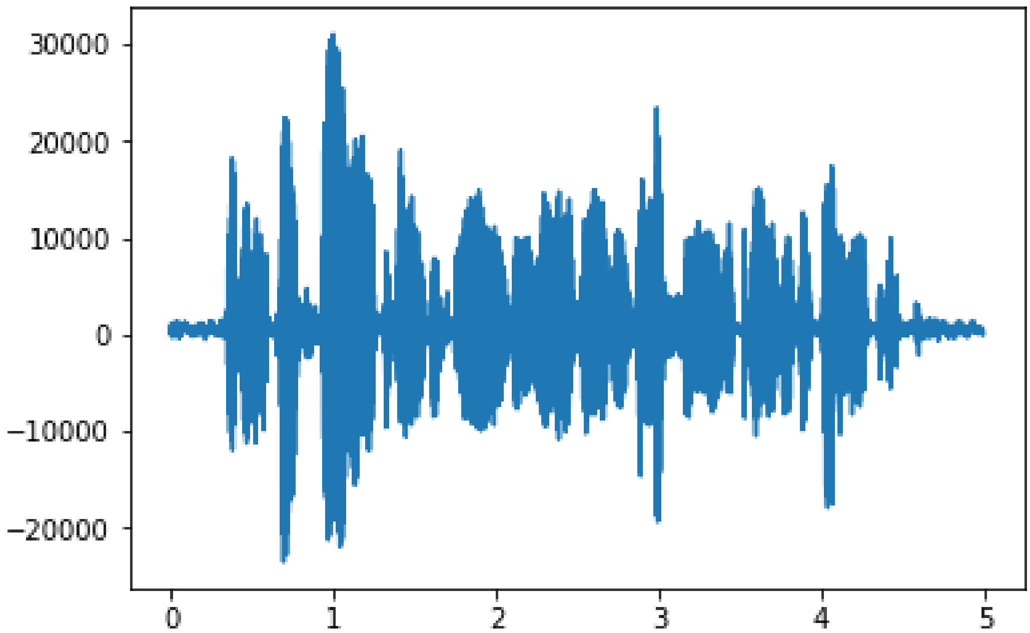

Time-domain runtime is set to 10ms and maximum step size is set to 1us.

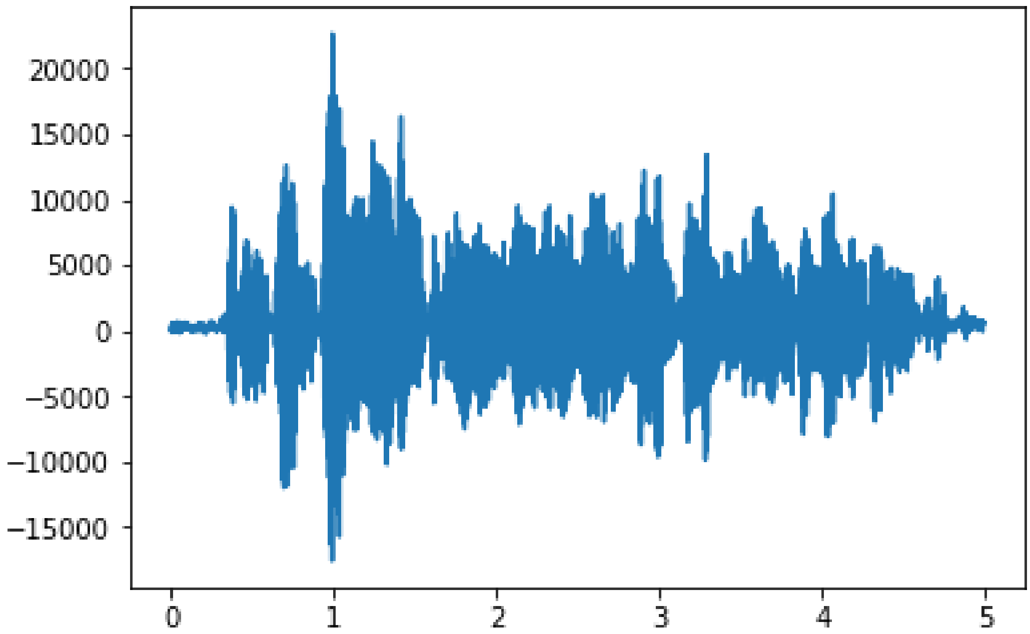



Linear AC sweep frequency starts at 1 Hz and ends at 4000 Hz.

(3) Working with real audio inputs.
It is now required to go back to the 1st section and apply a real audio signal to the input and observe the results. You can read in a real signal using:

```
fs, audio = wavfile.read('c:\Temp\speech_dft.wav')
audio=audio[::4] # Dedimates by a factor of 4
fs=int(fs/4)
NN=len(audio)*1.0
t=np.arange(0,NN)/fs
plt.plot(t,audio)
plt.show()
b=np.zeros(1601)
b[0]=1
b[1600]=0.8
a=1
audioOP=op=lfilter(b,a,audio)
audioOP=audioOP/2
for i in range (len(audioOP)):
 audioOP[i]=int(audioOP[i])
plt.plot(t,audioOP)
plt.show()
wavfile.write('c:\Temp\save_speech_dft.wav',fs,audio)
wavfile.write('c:\Temp\echo_speech_dft.wav',fs,audioOP)
```




20000
15000
10000
5000
0
−5000
−10000
−15000
0
1
2
3
4
5