****

***Ahsanullah University of Science and Technology***

***PROJECT REPORT***

Course No: ***EEE3218***

Course Name: ***Digital Signal Processing Laboratory***

* Name: ***Md. Idrak Efaz***
* ID: *190205121*
* Year: 3 , Semester: 2
* Section: B-2
* Department of *EEE*

***TASK 01***

***In question said that ‘****NonOverlapping.wav*’ contains the non-overlapping **modified** sounds of the musical instruments – ***guitar, piano, trumpet and violin*** (in this order)

|  |
| --- |
| ***For guitar***    ***Frequency:0-2000(about)*** |
| clc ;  clear all;  close all;  samples = [1,13\*44100];%range for  [es1,Fs1]=audioread('NonOverlapping.wav',samples);  N = length(es1); % sample lenth  slength = N/Fs1; % total time span of audio signal  t = linspace(0, N/Fs1, N);  n=0:length(es1)-1;  % time domain plot  figure(1)  plot(t, es1); % plots the audio  title('spectrum of 'NonOverlapping.wav');  xlabel('Time(s)');  ylabel('Amplitude');  %find the frequencies of the signal  es2=fft(es1);  N=length(es2);  figure(2)  f=(n\*Fs1)/N;  stem(f,es2);  xlabel('Frequency (Hz)');  title(' frequencies of NonOverlapping.wav'); |

|  |
| --- |
| ***For piano***  ***Frequency:2100-5000(about)*** |
| clc ;  clear all;  close all;  samples = [14\*44100,32\*44100];%range for  [es1,Fs1]=audioread('NonOverlapping.wav',samples);  N = length(es1); % sample lenth  slength = N/Fs1; % total time span of audio signal  t = linspace(0, N/Fs1, N);  n=0:length(es1)-1;  % time domain plot  figure(1)  plot(t, es1); % plots the audio  title('spectrum of NonOverlapping.wav');  xlabel('Time(s)');  ylabel('Amplitude');  %find the frequencies of the signal  es2=fft(es1);  N=length(es2);  figure(2)  f=(n\*Fs1)/N;  stem(f,es2);  xlabel('Frequency (Hz)');  title(' frequencies of NonOverlapping.wav'); |

|  |
| --- |
| ***For trumpet***    ***Frequency:5000-10000(about)*** |
| clc ;  clear all;  close all;  samples = [33\*44100,50\*44100];%range for  [es1,Fs1]=audioread('NonOverlapping.wav',samples);  N = length(es1); % sample lenth  slength = N/Fs1; % total time span of audio signal  t = linspace(0, N/Fs1, N);  n=0:length(es1)-1;  % time domain plot  figure(1)  plot(t, es1); % plots the audio  title('spectrum of NonOverlapping.wav');  xlabel('Time(s)');  ylabel('Amplitude');  %find the frequencies of the signal  es2=fft(es1);  N=length(es2);  figure(2)  f=(n\*Fs1)/N;  stem(f,es2);  xlabel('Frequency (Hz)');  title(' frequencies of NonOverlapping.wav'); |

|  |
| --- |
| ***For violin***    ***Frequency:16500-17000(about)*** |
| clc ;  clear all;  close all;  samples = [51\*44100,58\*44100];%range for  [es1,Fs1]=audioread('NonOverlapping.wav',samples);  N = length(es1); % sample lenth  slength = N/Fs1; % total time span of audio signal  t = linspace(0, N/Fs1, N);  n=0:length(es1)-1;  % time domain plot  figure(1)  plot(t, es1); % plots the audio  title('spectrum of NonOverlapping.wav'');  xlabel('Time(s)');  ylabel('Amplitude');  %find the frequencies of the signal  es2=fft(es1);  N=length(es2);  figure(2)  f=(n\*Fs1)/N;  stem(f,es2);  xlabel('Frequency (Hz)');  title(' frequencies of NonOverlapping.wav'); |

***TASK 02***

|  |
| --- |
| clc ;  clear all;  close all;  %samples = [51\*44100,58\*44100];%range for  [es1,Fs1]=audioread('final.wav');  N = length(es1); % sample lenth  slength = N/Fs1; % total time span of audio signal  t = linspace(0, N/Fs1, N);  n=0:length(es1)-1;  % time domain plot  figure(1)  plot(t, es1); % plots the audio  title('spectrum of final.wav');  xlabel('Time(s)');  ylabel('Amplitude');  %find the frequencies of the signal  es2=fft(es1);  N=length(es2);  figure(2)  f=(n\*Fs1)/N;  stem(f,es2);  xlabel('Frequency (Hz)');  title(' frequencies of final.wav'); |
| **Frequency Range 00hz to 17k** |

***TASK 03***

|  |
| --- |
| ***Filter no 1***  ***Low pass FIR filter***  ***Method: Window(Hamming)***  ***Order:1000***  ***Fc:2000;***  ***Filter no 2***  ***Band pass FIR filter***  ***Method: Window(Blackman)***  ***Order:2426***  ***Fc:2000-5000;***  ***Filter no 3***  ***Band pass FIR filter***  ***Method: Window(Blackman)***  ***Order:2426***  ***Fc:5000-10000;***  ***Filter no 4***  ***Band pass FIR filter***  ***Method: Window(blackman)***  ***Order:2426***  ***Fc:14000-16000;*** |

***TASK 04***

|  |
| --- |
| clc;  clear all;  close all;  [es1,Fs1]=audioread('final.wav');  **%% low pass FIR filter**  TW = 0.11;  PBE = 2000;  M1=1000;  corner = PBE+TW/2;  wn = 2\*corner/Fs1;  a1 = fir1(M1, wn, 'low', hamming(M1+1));  %% Filtering the Audio Data  filtered\_audio\_data\_1 = filter(a1,1,es1) ;  audiowrite('output1.wav',filtered\_audio\_data\_1 ,Fs1);  **%% FIR Bandpass filter 1**  Fs = Fs1;  TW = 100;  PBE1 = 2000;  PBE2 = 5000;  delf = TW/Fs;  M2 = round(5.5/delf);  corner1 = PBE1+TW/2;  corner2 = PBE2+TW/2;  wn1 = 2\*corner1/Fs;  wn2 = 2\*corner2/Fs;  wn=[wn1,wn2];  a2= fir1(M2, wn, 'bandpass', blackman(M2+1));  %% Filtering the Audio Data  filtered\_audio\_data\_2 = filter(a2,1,es1) ;  audiowrite('output2.wav',filtered\_audio\_data\_2,Fs1);  **%% FIR Band pass filter 2**  TW3 = 100;  PBE31 = 5000;  PBE32 = 10000;  delf3 = TW3/Fs;  M3 = round(5.5/delf3);  corner31 = PBE31+TW3/2;  corner32 = PBE32+TW3/2;  wn31 = 2\*corner31/Fs;  wn32 = 2\*corner32/Fs;  wn3=[wn31,wn32];  a3= fir1(M3, wn3, 'bandpass', blackman(M3+1));  %% Filtering the Audio Data  filtered\_audio\_data\_3 = filter(a3,1,es1) ;  audiowrite('output3.wav',filtered\_audio\_data\_3,Fs1);  **%% FIR Band pass filter 3**  TW4 = 100;  PBE41 = 14000;  PBE42 = 16000;  delf4 = TW3/Fs;  M4 = round(5.5/delf4);  corner41 = PBE41+TW4/2;  corner42 = PBE42+TW4/2;  wn41 = 2\*corner41/Fs;  wn42 = 2\*corner42/Fs;  wn4=[wn41,wn42];  a4= fir1(M4, wn4, 'bandpass', blackman(M4+1));  %% Filtering the Audio Data  filtered\_audio\_data\_4 = filter(a4,1,es1) ;  audiowrite('output4.wav',filtered\_audio\_data\_4,Fs1); |

***TASK05***

|  |
| --- |
| clc;  close all;  clear all;  [es1,Fs1]=audioread('output1.wav');  Ts=1/Fs1;  N = length(es1); % sample lenth  slength = N/Fs1; % total time span of audio signal  t = linspace(0, N/Fs1, N);  n=0:length(es1)-1;  %frequencies Plot  es2=fft(es1);  N=length(es2);  figure(2)  f=(n\*Fs1)/N;  stem(f,es2);  xlabel('Frequency (Hz)');  title(' frequencies of output1.wav'); |
| clc;  close all;  clear all;  [es1,Fs1]=audioread('output2.wav');  Ts=1/Fs1;  N = length(es1); % sample lenth  slength = N/Fs1; % total time span of audio signal  t = linspace(0, N/Fs1, N);  n=0:length(es1)-1;  %frequencies Plot  es2=fft(es1);  N=length(es2);  figure(2)  f=(n\*Fs1)/N;  stem(f,es2);  xlabel('Frequency (Hz)');  title(' frequencies of output2.wav'); |
| clc;  close all;  clear all;  [es1,Fs1]=audioread('output3.wav');  Ts=1/Fs1;  N = length(es1); % sample lenth  slength = N/Fs1; % total time span of audio signal  t = linspace(0, N/Fs1, N);  n=0:length(es1)-1;  %frequencies Plot  es2=fft(es1);  N=length(es2);  figure(2)  f=(n\*Fs1)/N;  stem(f,es2);  xlabel('Frequency (Hz)');  title(' frequencies of output3.wav'); |
| clc;  close all;  clear all;  [es1,Fs1]=audioread('output4.wav');  Ts=1/Fs1;  N = length(es1); % sample lenth  slength = N/Fs1; % total time span of audio signal  t = linspace(0, N/Fs1, N);  n=0:length(es1)-1;  %frequencies Plot  es2=fft(es1);  N=length(es2);  figure(2)  f=(n\*Fs1)/N;  stem(f,es2);  xlabel('Frequency (Hz)');  title(' frequencies of output4.wav'); |

***Pass the individual wav files separately using “Time-division multiplexing (TDM)”.***

***TASK06***

|  |
| --- |
| clc;  clear all;  close all;  **%% FDM multiplexer**  fc1=5121;  [m,fs1]=audioread('output1.wav');  [n,fs2]=audioread('output2.wav');  t1=[1:length(m)]';  ct1=cos(2\*pi\*fc1\*t1);  Xfdm1 = m.\*ct1;  Xfdm\_fft1=abs(fft(Xfdm1));  figure  stem(Xfdm\_fft1);  title('frequency domain Mulitplexed signal 1');  xlabel('Frequency');  ylabel('Amplitude');  fc2=6000; **%chosse another carrier**  t2=[1:length(n)]';  ct2=cos(2\*pi\*fc2\*t2);  Xfdm2 = n.\*ct2;  Xfdm\_fft2=abs(fft(Xfdm2));  figure  stem(Xfdm\_fft2);  title('frequency domain Mulitplexed signal 2');  xlabel('Frequency');  ylabel('Amplitude');  figure  stem(Xfdm\_fft1+Xfdm\_fft2);  title('frequency domain Mulitplexed sum of both signal ');  xlabel('Frequency'); |

|  |
| --- |
|  |

***TASK07***

**Convert this into an overlapping yet melodious one with proper synchronization of octaves.**

|  |
| --- |
| clc;  close all;  clear all;  [es1,Fs1]=audioread('output1.wav');  [es2,Fs2]=audioread('output2.wav');  [es3,Fs3]=audioread('output3.wav');  [es4,Fs4]=audioread('output4.wav');    final=es1+es2+es3+es4;  %% convertthis into an overlapping yet melodious ...  ... one with proper synchronization of octaves  audiowrite('mFinal.wav',final,Fs1);  sound(final,Fs1) |