Experiment Number	10
Date of Experiment	02/11/2020
Date of Submission	07/11/2020
Name of the student	Debagnik Kar
Roll Number	1804373
Section	ETC - 06

Aim of The Experiment :-

Open Ended 2
Application of Frequency division multiplexing using audio signal

Software Required:-

1. MATLAB r2018a
2. Audio files (Get it here)

Theory

Multiplexing is a process by which a number of signals can be transmitted through a single channel without interference.

There are mainly two type of multiplexing techniques for analog signals :

- Time division multiplexing
- Frequency division multiplexing.

In this project we will perform application of frequency division multiplexing using 3 audio clips.

Frequency-division multiplexing (FDM) is a technique by which the total bandwidth available in a communication medium is divided into a series of non-overlapping frequency bands, each of which is used to carry a separate signal.

FDM uses a carrier signal at a discrete frequency for each data stream and then combines many modulated signals. When FDM is used to allow multiple users to share a single physical communications medium (i.e. not broadcast through the air), the technology is called frequency-division multiple access (FDMA).

Here frequency division multiplexing is performed on 3 SSBSC modulated signals (audio clips) with different carrier frequencies. Each modulated signal is allocated with different frequency spectrum created at lets say fc1,fc2,and fc3 and therefore signal m1,m2 and m3 do not interfere with each other.

Code:-

<<<SimpleFDM.m Comment: Generates SSBSC signals from the audio files then transmit it through a noisy in frequency division multiplexed way, upon receiving noise is reduced, demultiplex and then SSBSC demodulated. Then played the message back>>>

```
%Experiment 10: Open ended 2:
%Application of FDM Multiplexing and Demultiplexing
%Collaboration of
%Debagnik Kar (1804373) and Sayani Ghoroi (1804406)
clear all
clc
close all
%PARAMETERS
bandwidth = 4000; % bandwidth for each frequency band in Hz
media_guard = 300; %
signal_to_noise_ratio = 20;
modudation_ssb = 1;% 1 for Single SSB modulation, 0 for AM
%The first signal will be placed in the third channel, the
second in the
%fourth and the third in the fifth.
frec_carrier1 = bandwidth*3;% Carrier frequency in Hz
frec_carrier2 = bandwidth*4;
frec_carrier3 = bandwidth*5;
Fs = frec_carrier3*2+5000;
cutoff_freq_passfilter = 2500;
% 1 show graphics, 0 don't
graphics = 1;
% 1 play sounds, 0 don't
sounds = 1;
%Define filters
[B,A] = butter(4,cutoff_freq_passfilter/(Fs/2));
low_pass = @(S) filter(B,A,S); %function handeling filter() in
low_pass variable to know more about it go to
[C1,D1] = butter(2,[bandwidth*2+media_guard bandwidth*3-
media_guard]/(Fs/2));
band_filter3 = @(S) filter(C1,D1,S);
[C2,D2] = butter(2,[bandwidth*3+media_guard bandwidth*4-
media_guard]/(Fs/2));
band_filter4 = @(S) filter(C2,D2,S);
[C3,D3] = butter(2,[bandwidth*4+media_guard bandwidth*5-
media_guard]/(Fs/2));
band_filter5 = @(S) filter(C3,D3,S);
%upload the files
[s1, g1] = audioread('1.wav');
len1 = length(s1);
[s2, g2]= audioread('2.wav');
len2 = length(s2);
[s3, g3]= audioread('3.wav'); % you got rick rolled lol in
2020
len3 = length(s3);
```

```
[beep, g4]= audioread('beep-8.wav');
playerbeep = audioplayer(beep,44100);
%are truncated to the length of the minor
min_len = min([len1 len2]);
t = linspace(0,5, min_len);
s1 = s1(1:min_len);
s2 = s2(1:min_len);
s3 = s3(1:min_len);
FLAG = input('STEP 1, the signals are reproduced as they
arrive');
%playing sounds
if (sounds > 0)
    player = audioplayer(s1,g1);
    playblocking(player);
    playblocking(playerbeep);
    player2 = audioplayer(s2,g2);
    playblocking(player2);
    playblocking(playerbeep);
    player3 = audioplayer(s3,g3);
    playblocking(player3);
end
FLAG = input('STEP 2, plot the spectra of the signals as they
arrive');
if (graphics > 0)
    figure
    esps1=abs(fft(s1));
    subplot(3,1,1),plot(esps1),grid on,zoom,title('Signal
Spectrum 1');xlabel("Frequency, Hz");ylabel("Amplitude, dB");
    esps2=abs(fft(s2));
    subplot(3,1,2),plot(esps2),grid on,zoom,title('Signal
Spectrum 2');xlabel("Frequency, Hz");ylabel("Amplitude, dB");
    esps3=abs(fft(s3));
    subplot(3,1,3),plot(esps3),grid on,zoom,title('Signal
Spectrum 3');xlabel("Frequency, Hz");ylabel("Amplitude, dB");
end;
FLAG = input('STEP 3, Signals are passed through a low pass
filter and plotted');
%they go through the low pass filter
s1 = low_pass(s1); %function handling by a variable
s2 = low_pass(s2);
s3 = low_pass(s3);
%Plot
if (graphics > 0)
    figure
    esps1=abs(fft(s1));
    subplot(3,1,1),plot(esps1),grid on,zoom,title('Signal
spectrum 1 filtered');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
    esps2=abs(fft(s2));
```

```
    subplot(3,1,2),plot(esps2),grid on,zoom,title('Signal
spectrum 2 filtered');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
    esps3=abs(fft(s3));
    subplot(3,1,3),plot(esps3),grid on,zoom,title('Signal
spectrum 3 filtered');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
end
FLAG = input('STEP 4, reproduce the signals after passing them
through the filter');
%Played again
if (sounds > 0)
    playerbeep = audioplayer(beep,44100);
    player = audioplayer(s1,g1);
    playblocking(player);
    playblocking(playerbeep);
    player2 = audioplayer(s2,g2);
    playblocking(player2);
    playblocking(playerbeep);
    player3 = audioplayer(s3,g3);
    playblocking(player3);
    playblocking(playerbeep);
end
FLAG = input('STEP 5, Signals are modulated to different
carriers');
%Modulate
if ( modudation_ssb  > 0)
    s1mod = ssbmod(s1,frec_carrier1,Fs);%modulates
    s2mod = ssbmod(s2,frec_carrier2,Fs);%modulates
    s3mod = ssbmod(s3,frec_carrier3,Fs);%modulates
else
    s1mod = ammod(s1,frec_carrier1,Fs);%modulates
    s2mod = ammod(s2,frec_carrier2,Fs);%modulates
    s3mod = ammod(s3,frec_carrier3,Fs);%modulates
end
%plotted
if (graphics > 0)
    figure
    esps1=abs(fft(s1mod));
    subplot(3,1,1),plot(esps1),grid on,zoom,title('Signal
spectrum1 modulated');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
    esps2=abs(fft(s2mod));
    subplot(3,1,2),plot(esps2),grid on,zoom,title('Signal
spectrum2 modulated');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
    esps3=abs(fft(s3mod));
    subplot(3,1,3),plot(esps3),grid on,zoom,title('Signal
spectrum2 modulated');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
end
```

```
FLAG = input('STEP 6, The modulated signals are filtered in
the determined band and summed');
fs1 = s1mod;
fs2 = s2mod;
fs3 = s3mod;
%added
x = fs1+fs2+fs3;
%plotted again
if (graphics > 0)
    figure
    esps1=abs(fft(fs1));
    subplot(4,1,1),plot(esps1),grid on,zoom,title('Signal
spectrum1 modulated and filtered');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
    esps2=abs(fft(fs2));
    subplot(4,1,2),plot(esps2),grid on,zoom,title('Signal
spectrum2 modulated and filtered');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
    esps3=abs(fft(fs3));
    subplot(4,1,3),plot(esps3),grid on,zoom,title('Signal
spectrum3 modulated and filtered');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
    espf=abs(fft(x));
    subplot(4,1,4),plot(espf),grid on,zoom,title('Summed
Spectrum');xlabel("Frequency, Hz");ylabel("Amplitude, dB");
end
FLAG = input('STEP 7, add some noise to the transmitted
signal');
if (graphics > 0)
    figure
    esps1=abs(fft(x));
    subplot(2,1,1),plot(esps1),grid on,zoom,title('Full signal
spectrum');xlabel("Frequency, Hz");ylabel("Amplitude, dB");
end
x = awgn(x, signal_to_noise_ratio );
if (graphics > 0)
    esps1=abs(fft(x));
    subplot(2,1,2),plot(esps1),grid on,zoom,title('Full signal
spectrum plus some noise'); xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
end
FLAG = input('STEP 8, upon arrival each band is filtered');
%signals are received and filtered
demuxs1 = band_filter3(x);
demuxs2 = band_filter4(x);
demuxs3 = band_filter5(x);
%Plotted again
if (graphics > 0)
    figure
    esps1=abs(fft(demuxs1));
```

```
    subplot(3,1,1),plot(esps1),grid on,zoom,title('Signal
spectrum1 filtered');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
    esps2=abs(fft(demuxs2));
    subplot(3,1,2),plot(esps2),grid on,zoom,title('Signal
spectrum2 filtered');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
    esps3=abs(fft(demuxs3));
    subplot(3,1,3),plot(esps3),grid on,zoom,title('Signal
spectrum3 filtered');xlabel("Frequency,
Hz");ylabel("Amplitude, dB");
end
FLAG = input('STEP 9, each recovered band is demodulated to
return the signal to the indicated frequency');
%Demodulate
if ( modudation_ssb  > 0)
    demods1 = ssbdemod(demuxs1, frec_carrier1,Fs);
    demods2 = ssbdemod(demuxs2, frec_carrier2,Fs);
    demods3 = ssbdemod(demuxs3, frec_carrier3,Fs);
else
    demods1 = amdemod(demuxs1, frec_carrier1,Fs);
    demods2 = amdemod(demuxs2, frec_carrier2,Fs);
    demods3 = amdemod(demuxs3, frec_carrier3,Fs);
end;
%Plotting
if (graphics > 0)
    figure
    esps1=abs(fft(demods1));
    subplot(3,1,1),plot(esps1),grid on,zoom,title('Demodulated
signal1 spectrum');xlabel("Frequency, Hz");ylabel("Amplitude,
dB");
    esps2=abs(fft(demods2));
    subplot(3,1,2),plot(esps2),grid on,zoom,title('Demodulated
signal2 spectrum');xlabel("Frequency, Hz");ylabel("Amplitude,
dB");
    esps3=abs(fft(demods3));
    subplot(3,1,3),plot(esps3),grid on,zoom,title('Demodulated
signal3 spectrum');xlabel("Frequency, Hz");ylabel("Amplitude,
dB");
end
FLAG = input('STEP 10, the recovered signal is passed through
a low pass filter');
%played
demods1 = low_pass(demods1);
demods2 = low_pass(demods2);
demods3 = low_pass(demods3);
if (graphics > 0)
    figure
    esps1=abs(fft(demods1));
```

```
    subplot(3,1,1),plot(esps1),grid on,zoom,title('Demodulated
signal1 spectrum');xlabel("Frequency, Hz");ylabel("Amplitude,
dB");
    esps2=abs(fft(demods2));
    subplot(3,1,2),plot(esps2),grid on,zoom,title('Demodulated
signal2 spectrum');xlabel("Frequency, Hz");ylabel("Amplitude,
dB");
    esps3=abs(fft(demods3));
    subplot(3,1,3),plot(esps3),grid on,zoom,title('Demodulated
signal3 spectrum');xlabel("Frequency, Hz");ylabel("Amplitude,
dB");
end
FLAG = input('STEP 11, Signal reproduce the signal after
transmission');
player4 = audioplayer(demods1,g2);
playblocking(player4);
playblocking(playerbeep);
player5 = audioplayer(demods2,g2);
playblocking(player5);
playblocking(playerbeep);
player6 = audioplayer(demods3,g3);
playblocking(player6);
```

Output/Graph:-

STEP 1, the signals are reproduced as they arrive:

**Three Audios plays*

STEP 2, plot the spectra of the signals as they arrive

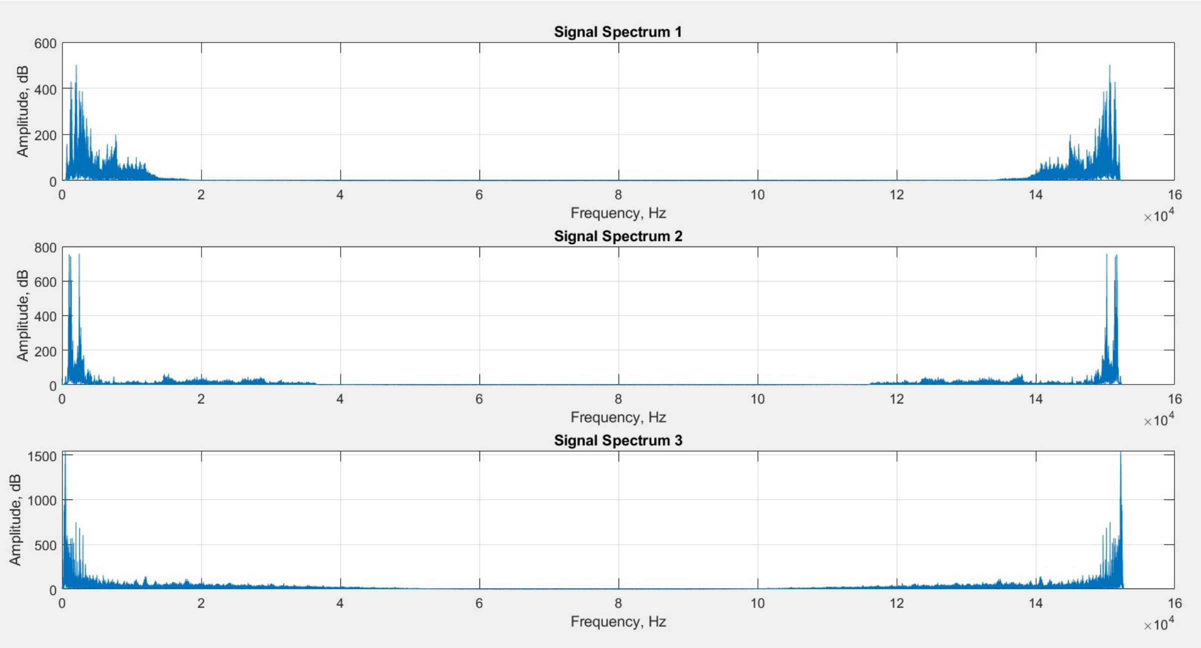

Fig 10.1: Step 2, spectral plot for the three signals

STEP 3, Signals are passed through a low pass filter and plotted

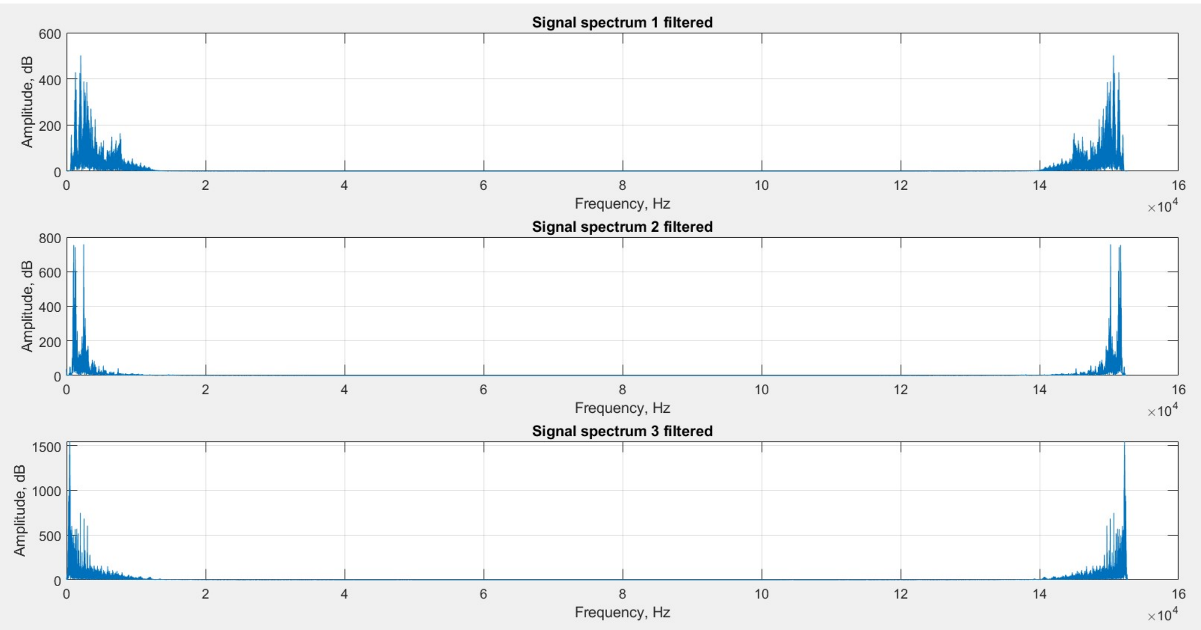

Fig 10.2: Step 3, Spectral plot for filtered signals

STEP 4, reproduce the signals after passing them through the filter

***Filtered audio signal plays*

STEP 5, Signals are modulated to different carriers

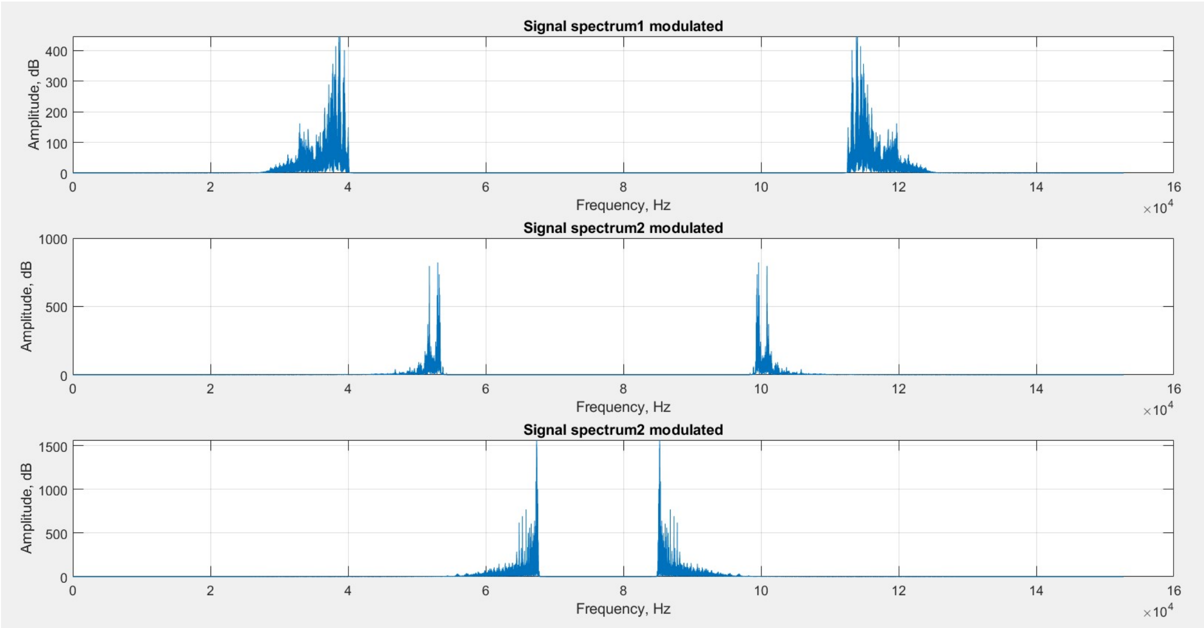

Fig 10.3: Step 5, SSBSC Modulated and filtered spectral plot

STEP 6, The modulated signals are filtered in the determined band and summed

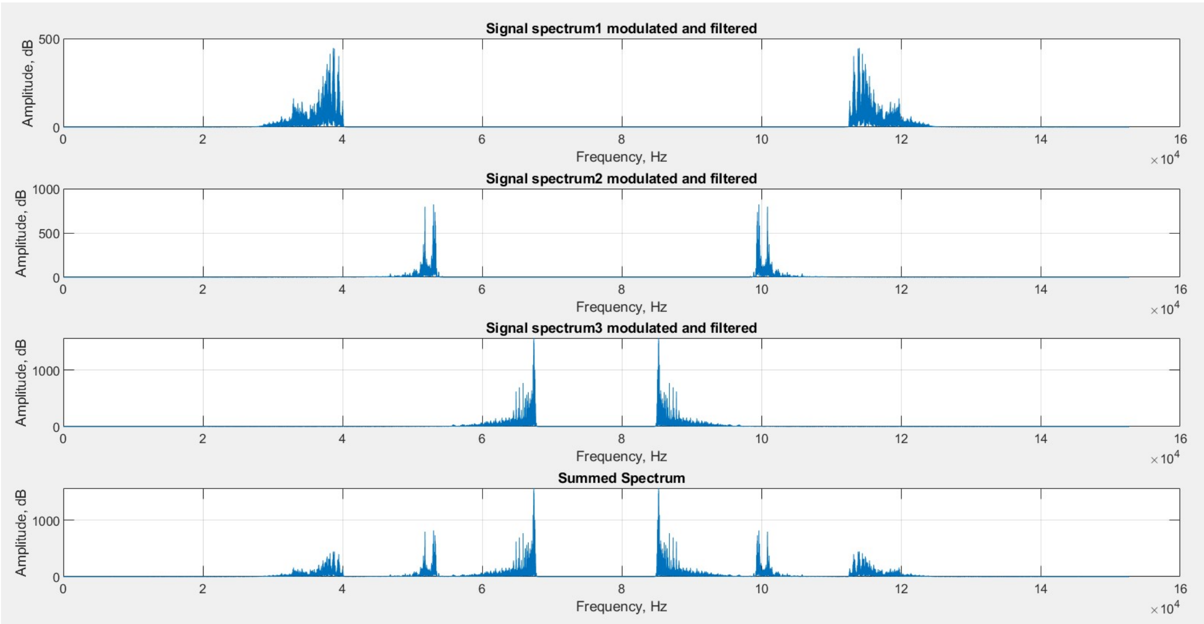

Fig 10.4: The three signals are frequency multiplexed

STEP 7, add some noise to the transmitted signal

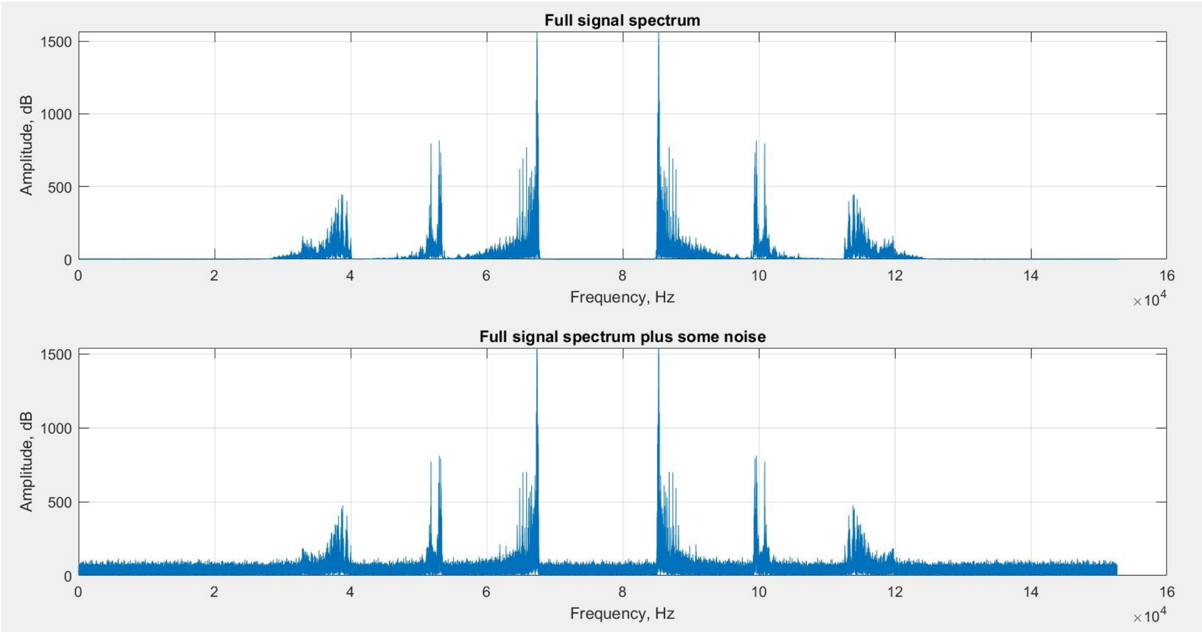

Fig 10.5: Multiplexed signal spectral plot with and without noise.

STEP 8, upon arrival each band is filtered

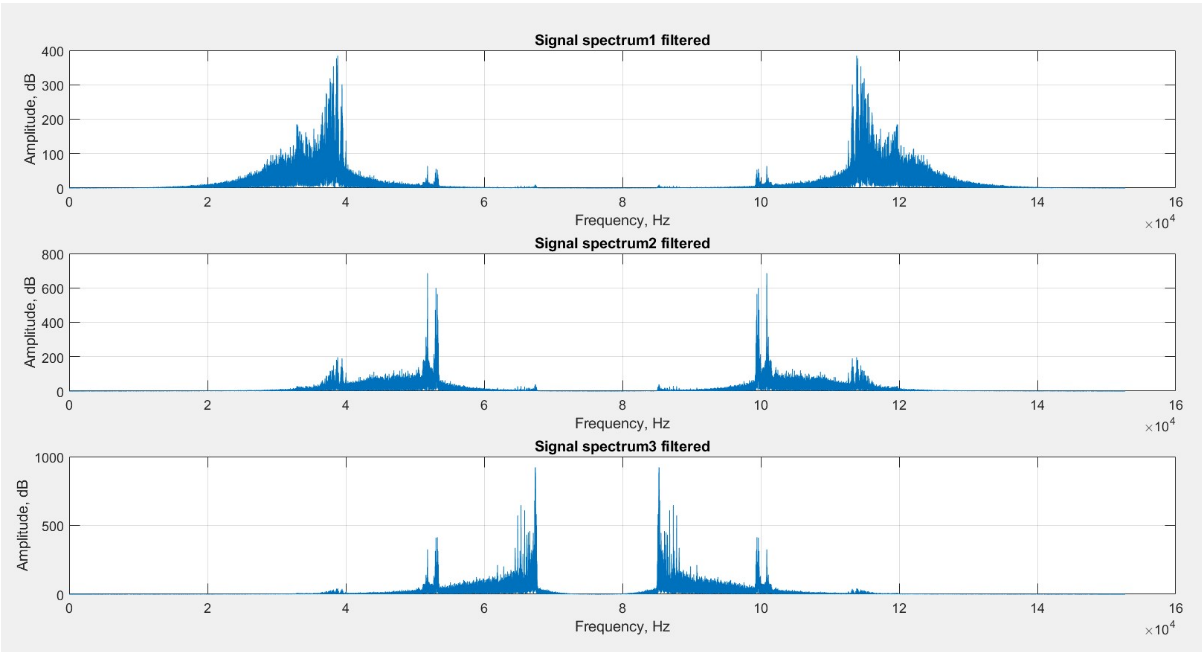

Fig 10.6: spectral plot of demultiplexed signals upon receiving.

STEP 9:, each recovered band is demodulated to return the signal to the indicated frequency

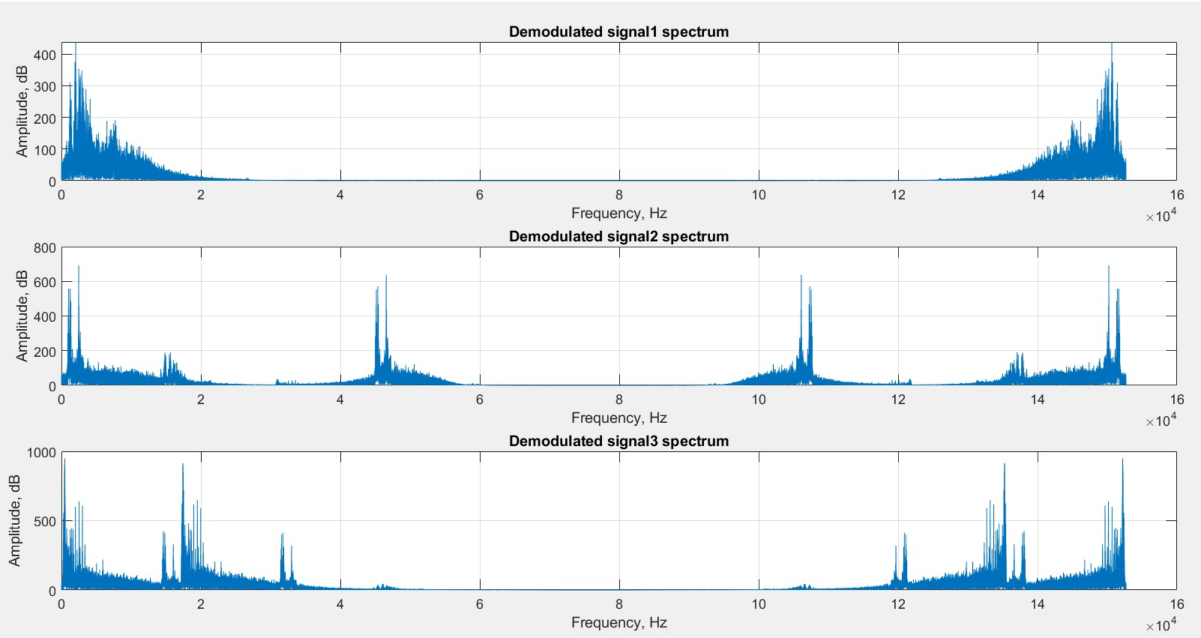

Fig 10.6: the signals are SSBSC demodulated

STEP 10, the recovered signal is passed through a low-pass filter

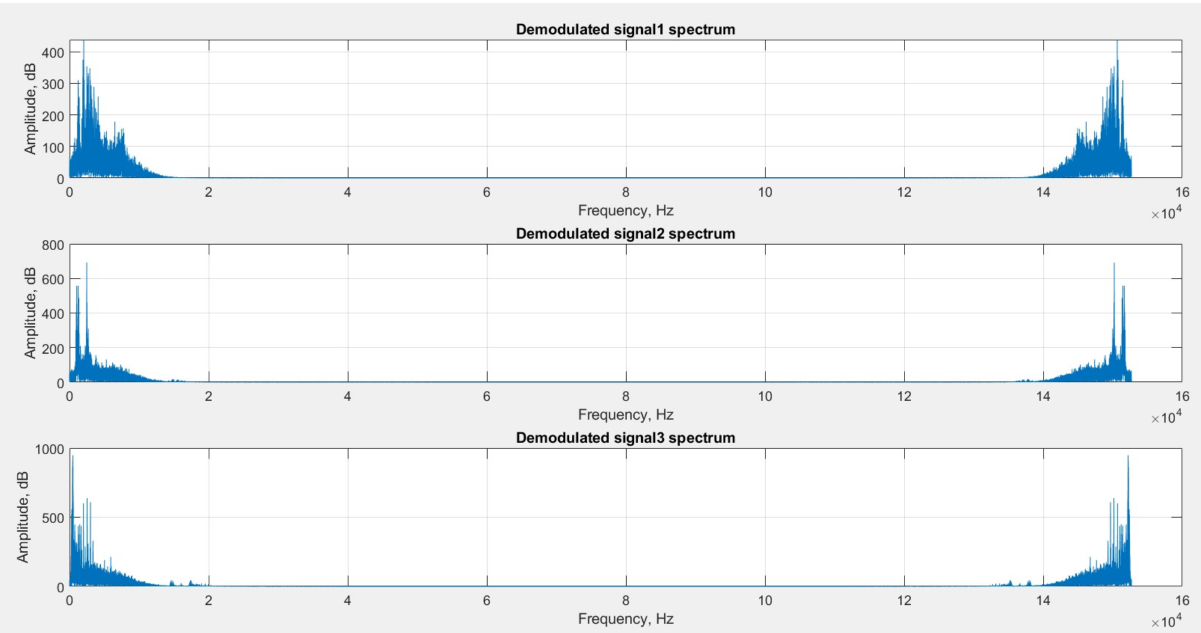

Fig 10.7: spectral plot of the recovered signals

STEP 11, Signal reproduce the signal after transmission

***The received and recovered signals are played*

Discussion or Inference of the experiment

From this open-ended project, we learned about the frequency division multiplexing and demultiplexing of signals during communication through a channel and we applied it generate 3 SSBSC modulated message signals (audio clips) with different carriers . we also were able to successfully multiplex ,demultiplex and demodulate the signal with the help of MATLAB to get back the message signals (audio clips) . Also, all the waveforms were simulated as expected in theory.

Conclusion:-

From this project we successfully performed Frequency division multiplexing using audio signal and its demultiplexing using MATLAB and simulated their waveforms.