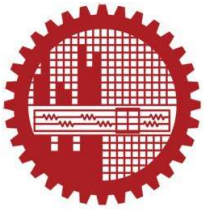
Bangladesh University of Engineering & Technology



Department of ELECTRICAL & ELECTRONIC ENGINEERING

Course: EEE-212

Course Title: Numerical Technique Laboratory

Project No: 07

Project Title: Time and Frequency domain analysis of Frequency division multiplexing (FDM) scheme using numerical method.

Name: Tusher Karmakar

Student ID: 1806174

Section: C2

Date of Submission: 30.07.21

Objective:

FDM or Frequency Division Multiplexing is a networking method of sharing the total available bandwidth of any communication channel by diving them into many non-overlapping bands of frequency. For the matlab project, I was assigned a project of Time and Frequency domain analysis of Frequency division multiplexing (FDM) scheme using numerical method. The purpose of this project is-

1. To able to take and process should be at least 3 or more Input signal (X1, X2o.....) of both the real time or recorded audio signal of reasonable duration.
2. To able to show the retrieved (demodulated signal) and must be able to play it in computer audio.
3. To Show the output after each block both in time domain and frequency domain.
4. To be able to Make a suitable GUI for this project.

Algorithm:

Algorithm for frequency division multiplexing in App Designer matlab:

Step 1: At first, we need take input number of audio signals. We can take N number of signals here where N =3

Step 2: Then we need to take the time duration for the signals in second.

Step 3: We need to take input of sample frequency , number of channel Number of channels--2 options--1 (mono) or 2 (stereo), number of bits 8,16,24. Also need to take three carrier frequency.

Step 4: Here, we need to choose the mp3/wav file or need to start recording the audio.

Step 5: We need to take N number of carrier frequency for N input signals.

Step 6: Here, the task is to modulate. We need to multiply the input signal with the carrier signal.

Step 7: Now, we have to sum all the modulated signals and pass it in a channel.

Step 8: Here, the task is to pass the modulated signal in the bandpass filters. For this we have used the function of filter Bandpass

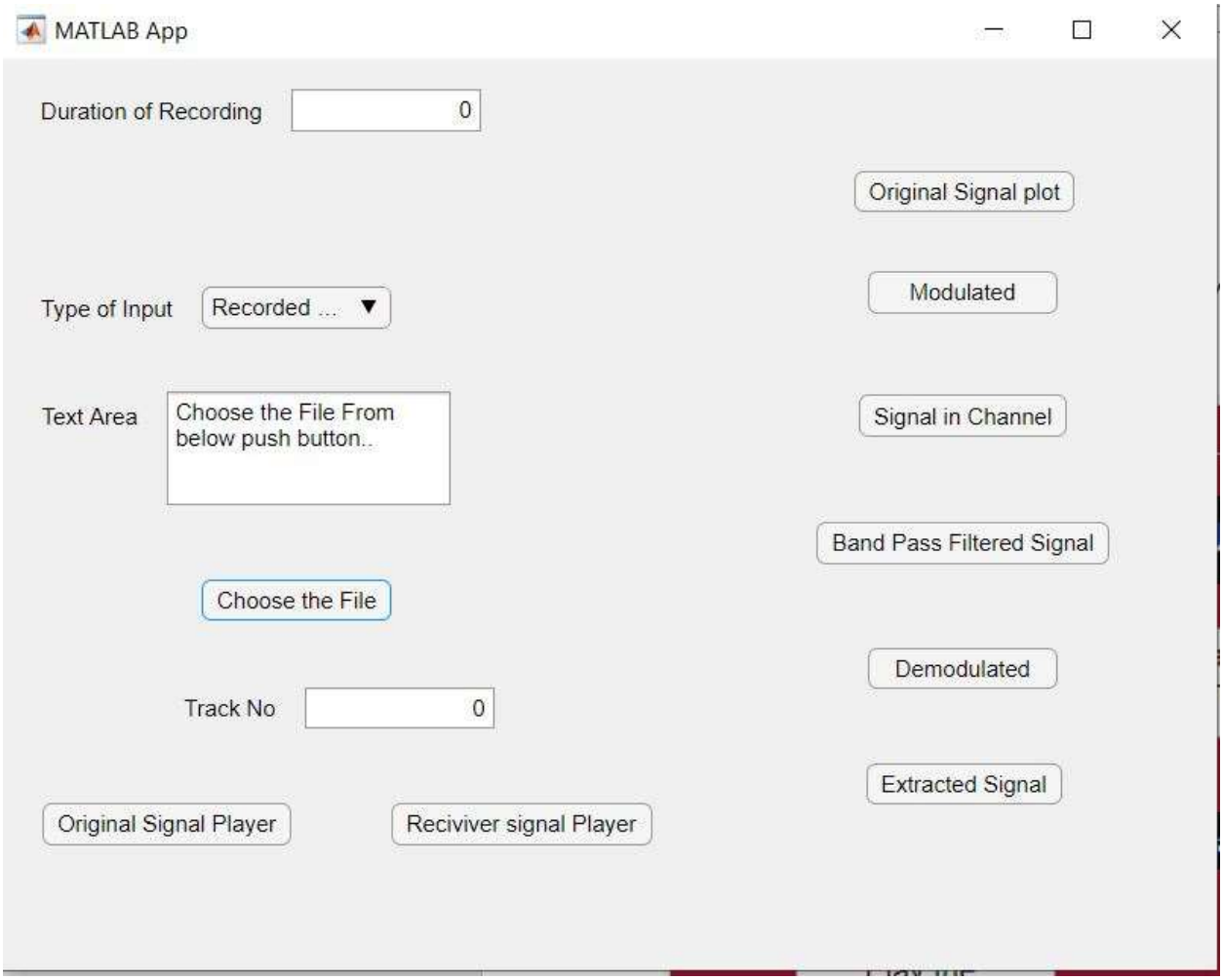
Step 9: Here, we need to demodulate all the bandpass filtered signals. To demodulate we have to multiply the filtered signals with its carrier signals.

Step 10: Then to retrieve the signal we need to pass the demodulated signal through the lowpass filter and thus we can recover all the signals.

Step 11: Here we need to convert the signals from time domain to frequency domain and need to plot both time and frequency domain graphs of every block. To convert in frequency domain we have used fft and fftshift functions.

Step 12: The last part is to play the original and the retrieved signals. To play them we have used sound function. To play the audios we need to run the code and tap the buttons for playing the audios in the app designer of matlab.

Design View of Code:



Code View:

```
classdef Project1806174 < matlab.apps.AppBase


% Properties that correspond to app components
properties (Access = public)
UIFigure matlab.ui.Figure
TypeofInputDropDownLabel matlab.ui.control.Label
TypeofInputDropDown matlab.ui.control.DropDown
ChoosetheFileButton matlab.ui.control.Button
```

```
TextAreaLabel matlab.ui.control.Label
TextArea matlab.ui.control.TextArea
OriginalSignalplotButton matlab.ui.control.Button
ModulatedButton matlab.ui.control.Button
SignalinChannelButton matlab.ui.control.Button
BandPassFilteredSignalButton matlab.ui.control.Button
DemodulatedButton matlab.ui.control.Button
ExtractedSignalButton matlab.ui.control.Button
OriginalSignalPlayerButton matlab.ui.control.Button
ReciviersignalPlayerButton matlab.ui.control.Button
TrackNoEditFieldLabel matlab.ui.control.Label
TrackNoEditField matlab.ui.control.NumericEditField
DurationofRecordingEditFieldLabel matlab.ui.control.Label
DurationofRecordingEditField matlab.ui.control.NumericEditField
end


properties (Access = private)
Nor=3;
Fs=44100;
fc1;
fc2;
fc3;
x1=[];
x2=[];
x3=[];
cs1=[];
cs2=[];
cs3=[];
chasig=[];
N;
N1;
N2;
N3;
ymod1=[];
ymod2=[];
ymod3=[];
z1=[];
z2=[];
z3=[];
z1b=[];
z2b=[];
z3b=[];
rec1=[];
rec2=[];
rec3=[];
t1=[];
t2=[];
t3=[];
% Description
```

```
end


% Callbacks that handle component events
methods (Access = private)


% Button pushed function: ChoosetheFileButton
function ChoosetheFileButtonPushed(app, event)


rt=app.DurationofRecordingEditField.Value;
app.TextArea.Value='Choose the File From below push button..';
if (strcmp(app.TypeofinputDropDown.Value,'Recorded Signal')==1)
%Reciving First Input
[filename, pathname,]=uigetfile('*.*','Select the input');
samples=[1,rt*app.Fs];
[app.x1,app.Fs]=audioread(num2str(filename),samples);
%Reciving Second Input
[filename, pathname,Index]=uigetfile('*.*','Select the input');
samples=[1,rt*app.Fs];
                [app.x2,app.Fs]=audioread(num2str(filename),samples);
                %Reciving Third Input
                [filename, pathname,Index]=uigetfile('*.*','Select the input');
                samples=[1,rt*app.Fs];
                [app.x3,app.Fs]=audioread(num2str(filename),samples);
                app.TextArea.Value='Done..'
                %Frequency of Carrier signal
                app.fc1=1.1*10^(6);
                app.fc2=app.fc1+3*app.fc1;
                app.fc3=app.fc2+3*app.fc1;
                %First carrier signal
                app.N1=length(app.x1);
                app.t1=ones(app.N1,1);
                for i=1:app.N1
                  app.t1(i,1)=(i-1)*(1/app.Fs);
                end
                app.cs1=cos(2*pi*app.fc1*app.t1);
                app.ymod1=(app.x1).*app.cs1;
                %Second carrier signal
                app.N2=length(app.x2);
                app.t2=ones(app.N2,1);
                for i=1:app.N2
                  app.t2(i,1)=(i-1)*(1/app.Fs);
                end
                app.cs2=cos(2*pi*app.fc2*app.t2);
                app.ymod2=(app.x2).*app.cs2;
                %Third carrier signal
                app.N3=length(app.x3);
                app.t3=ones(app.N3,1);
```

```
    for i=1:app.N3
      app.t3(i,1)=(i-1)*(1/app.Fs);
    end
    app.cs3=cos(2*pi*app.fc3*app.t3);
    app.ymod3=(app.x3.*app.cs3);
    app.chasig=(app.x1.*app.cs1)+(app.x2.*app.cs2)+(app.x3.*app.cs3);
    %For Bandpass
    %For the first One
    app.z1=bandpass(app.chasig,[50 9000],app.Fs);
    %For Second one
    app.z2=bandpass(app.chasig,[6000 14700],app.Fs);
    %For Third one
    app.z3=bandpass(app.chasig,[13000 21600],app.Fs);
    %Now mixing
    app.z1b=app.z1.*app.cs1;
    app.z2b=app.z2.*app.cs2;
    app.z3b=app.z3.*app.cs3;
    %reciving First signal
    app.rec1=lowpass(app.z1b,2500,app.Fs);
    %reciving Second Signal
    app.rec2=lowpass(app.z2b,2500,app.Fs);
    %reciving Third Signal
    app.rec3=lowpass(app.z3b,2500,app.Fs);

else
    ch=1;
    datatype='uint8';
    nbits=16;
    %Reciving First Input
    recorder1=audiorecorder(app.Fs,nbits,ch);
    disp('start spreaking1....');
    recordblocking(recorder1,rt);
    disp('End of Recording1...');
    app.x1=getaudiodata(recorder1,datatype);
    %Reciving Second Input
    recorder2=audiorecorder(app.Fs,nbits,ch);
    disp('start spreaking2....');
    recordblocking(recorder2,rt);
    disp('End of Recording2...');
    app.x2=getaudiodata(recorder2,datatype);
    %Reciving Third Input
    recorder3=audiorecorder(app.Fs,nbits,ch);
    disp('start spreaking3....');
    recordblocking(recorder3,rt);
    disp('End of Recording3...');
    app.x3=getaudiodata(recorder3,datatype);
    %Frequency of Carrier signal
    app.fc1=1.1*10^(6);
    app.fc2=app.fc1+3*app.fc1;
    app.fc3=app.fc2+3*app.fc1;
```

```
                %First carrier signal
                app.N1=length(app.x1);
                app.t1=ones(app.N1,1);
                for i=1:app.N1
                  app.t1(i,1)=(i-1)*(1/app.Fs);
                end
                app.cs1=cos(2*pi*app.fc1*app.t1);
                app.ymod1=(app.x1).*app.cs1;
                %Second carrier signal
                app.N2=length(app.x2);
                app.t2=ones(app.N2,1);
                for i=1:app.N2
                  app.t2(i,1)=(i-1)*(1/app.Fs);
                end
                app.cs2=cos(2*pi*app.fc2*app.t2);
                app.ymod2=(app.x2).*app.cs2;
                %Third carrier signal
                app.N3=length(app.x3);
                app.t3=ones(app.N3,1);
                for i=1:app.N3
                  app.t3(i,1)=(i-1)*(1/app.Fs);
                end
app.cs3=cos(2*pi*app.fc3*app.t3);
app.ymod3=(app.x3.*app.cs3);
app.chasig=(app.x1.*app.cs1)+(app.x2.*app.cs2)+(app.x3.*app.cs3);
%Band pass
%For the first One
app.z1=bandpass(app.chasig,[50 9000],app.Fs);
%For Second one
app.z2=bandpass(app.chasig,[6000 14700],app.Fs);
%For Third one
app.z3=bandpass(app.chasig,[13000 21600],app.Fs);
%Now mixing
app.z1b=app.z1.*app.cs1;
app.z2b=app.z2.*app.cs2;
app.z3b=app.z3.*app.cs3;
%reciving First signal
app.rec1=lowpass(app.z1b,2500,app.Fs);
%reciving Second Signal
app.rec2=lowpass(app.z2b,2500,app.Fs);
%reciving Third Signal
app.rec3=lowpass(app.z3b,2500,app.Fs);
end
end


% Callback function
function NoofInputSignalEditFieldValueChanged(app, event)
value = app.NoofInputSignalEditField.Value;
end
```

```
% Value changed function: TypeofInputDropDown
function TypeofInputDropDownValueChanged(app, event)
value = app.TypeofInputDropDown.Value;
end


% Button pushed function: OriginalSignalplotButton
function OriginalSignalplotButtonPushed(app, event)
figure(1);
subplot(2,1,1),plot(app.t1,app.x1);
xlabel('time');
ylabel('Amplitude');
title('Spectrum of First signal in Time domain');
X1=fftshift(fft(app.x1,app.N1));
f=app.Fs*[-app.N1/2:app.N1/2-1]/app.N1;
subplot(2,1,2),plot(f,abs(X1));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of First signal in frequency domain');
xlabel('Frequency');
ylabel('Amplitude');
figure(2);
subplot(2,1,1),plot(app.t2,app.x2);
xlabel('time');
ylabel('Amplitude');
title('Spectrum of Second signal in Time domain');
X2=fftshift(fft(app.x2,app.N2));
f=app.Fs*[-app.N2/2:app.N2/2-1]/app.N2;
subplot(2,1,2),plot(f,abs(X2));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Second signal in Frequency domain');
xlabel('Frequency');
ylabel('Amplitude');
figure(3);
subplot(2,1,1),plot(app.t3,app.x3);
xlabel('time');
ylabel('Amplitude');
title('Spectrum of Third signal in Time domain');
X3=fftshift(fft(app.x3,app.N3));
f=app.Fs*[-app.N3/2:app.N3/2-1]/app.N3;
subplot(2,1,2),plot(f,abs(X3));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Third signal in Frequency domain');
xlabel('Frequency');
ylabel('Amplitude');
end
```

```
% Button pushed function: ModulatedButton
function ModulatedButtonPushed(app, event)
figure(1);
subplot(2,1,1),plot(t1,ymod1);
xlabel('time');
ylabel('Amplitude');
title('Spectrum of Modulated First signal in Time domain');
X1m=fftshift(fft(ymod1,N1));
f=Fs*[-N1/2:N1/2-1]/N1;
subplot(2,1,2),plot(f,abs(X1m));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Modulated First signal in frequency domain');
xlabel('Frequency');
ylabel('Amplitude');
figure(1);
subplot(2,1,1),plot(t1,ymod2);
xlabel('time');
ylabel('Amplitude');
title('Spectrum of Modulated Second signal in Time domain');
X2m=fftshift(fft(ymod1,N2));
f=Fs*[-N2/2:N2/2-1]/N2;
subplot(2,1,2),plot(f,abs(X2m));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Modulated Second signal in frequency domain');
xlabel('Frequency');
ylabel('Amplitude');
figure(3);
subplot(2,1,1),plot(t3,ymod3);
xlabel('time');
ylabel('Amplitude');
title('Spectrum of Modulated Third signal in Time domain');
X3m=fftshift(fft(app.x3,N3));
f=Fs*[-N3/2:N3/2-1]/N3;
subplot(2,1,2),plot(f,abs(X3m));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Modulated Third signal in Frequency domain');
xlabel('Frequency');
ylabel('Amplitude');
end


% Button pushed function: SignalinChannelButton
function SignalinChannelButtonPushed(app, event)
```

```
subplot(2,1,1),plot(t1,chasig);
xlabel('time');
ylabel('Amplitude');
title('Spectrum of Modulated Composite signal in Time domain');
Xmain=fftshift(fft(chasig,N));
f=Fs*[-N/2:N/2-1]/N;
subplot(2,1,2),plot(f,abs(Xmain));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Modulated Composite signal in frequency domain');
xlabel('Frequency');
ylabel('Amplitude');
end


% Button pushed function: BandPassFilteredSignalButton
function BandPassFilteredSignalButtonPushed(app, event)
%Bandpass Design
%For the first One
figure(1)
z1=bandpass(chasig,[50 9000],fs1);
subplot(2,1,1),plot(t1,z1);
xlabel('Time');
ylabel('Amplitude');
title('Spectrum of First Bandpass filtered signal in time domain');
Na1=length(z1);
X1=fftshift(fft(z1,Na1));
f=Fs*[-Na1/2:Na1/2-1]/Na1;
subplot(2,1,2),plot(f,abs(X1));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of First Bandpass filtered signal');
%For second one
figure(2)
z2=bandpass(chasig,[6000 14700],fs1);
subplot(2,1,1),plot(t1,z2);
xlabel('Time');
ylabel('Amplitude');
title('Spectrum of Second Bandpass filtered signal in time domain');
Na2=length(z2);
X2=fftshift(fft(z2,Na2));
f=Fs*[-Na2/2:Na2/2-1]/Na2;
subplot(2,1,2),plot(f,abs(X2));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Second Bandpass filtered signal');
% For Third one
figure(3)
z3=bandpass(chasig,[13000 21600],fs1);
subplot(2,1,1),plot(t1,z3);
```

```
xlabel('Time');
ylabel('Amplitude');
title('Spectrum of Third Bandpass filtered signal in time domain');
Na3=length(z3);
X3=fftshift(fft(z2,Na1));
f=Fs*[-Na3/2:Na3/2-1]/Na3;
subplot(2,1,2),plot(f,abs(X3));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Third Bandpass filtered signal');
end


% Button pushed function: DemodulatedButton
function DemodulatedButtonPushed(app, event)
Figure(1)
Na4=length(z1b);
subplot(2,1,1),plot(app.t1,app.z1b);
xlabel('Time');
ylabel('Amplitude');
title('Spectrum of First Bandpass filtered signal in time domain');
Na41=length(app.z1b);
X1=fftshift(fft(app.z1b,Na41));
f=Fs*[-Na41/2:Na41/2-1]/Na41;
subplot(2,1,2),plot(f,abs(X1));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of First Bandpass filtered signal in frequency domain');
%Second one
Figure(2)
Na5=length(z2b);
subplot(2,1,1),plot(t1,z2b);
xlabel('Time');
ylabel('Amplitude');
title('Spectrum of Second Bandpass filtered signal in time domain');
Na51=length(z2b);
X1=fftshift(fft(z2b,Na51));
f=Fs*[-Na51/2:Na51/2-1]/Na51;
subplot(2,1,2),plot(f,abs(X1));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Second Bandpass filtered signal in frequency domain');
%Third one
Figure(2)
Na6=length(z3b);
subplot(2,1,1),plot(t1,z3b);
xlabel('Time');
ylabel('Amplitude');
title('Spectrum of Third Bandpass filtered signal in time domain');
Na61=length(z3b);
```

```
X1=fftshift(fft(z2b,Na61));
f=Fs*[-Na61/2:Na61/2-1]/Na61;
subplot(2,1,2),plot(f,abs(X1));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Third Bandpass filtered signal in frequency domain');
end


% Button pushed function: ExtractedSignalButton
function ExtractedSignalButtonPushed(app, event)
rec1=lowpass(z1b,2500,Fs);
Figure(1)
Na42=length(rec1);
subplot(2,1,1),plot(t1,rec1);
xlabel('Time');
ylabel('Amplitude');
title('Spectrum of first Output signal in time domain');
Na42=length(rec1);
X1=fftshift(fft(rec1,Na42));
f=Fs*[-Na42/2:Na42/2-1]/Na42;
subplot(2,1,2),plot(f,abs(X1));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of First output signal in frequency domain');
%For second one
rec2=lowpass(z2b,2500,Fs);
Figure(2)
Na43=length(rec2);
subplot(2,1,1),plot(t1,rec2);
xlabel('Time');
ylabel('Amplitude');
title('Spectrum of Second Output signal in time domain');
Na43=length(rec2);
X1=fftshift(fft(rec2,Na43));
f=Fs*[-Na43/2:Na43/2-1]/Na43;
subplot(2,1,2),plot(f,abs(X1));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Second output signal in frequency domain');
%For third one
rec3=lowpass(z3b,2500,Fs);
Figure(3)
Na44=length(rec3);
subplot(2,1,1),plot(t3,rec3);
xlabel('Time');
ylabel('Amplitude');
title('Spectrum of Third Output signal in time domain');
Na44=length(rec3);
X1=fftshift(fft(rec3,Na44));
```

```
f=Fs*[-Na44/2:Na44/2-1]/Na44;
subplot(2,1,2),plot(f,abs(X1));
xlabel('Frequency');
ylabel('DFT Value');
title('Spectrum of Third output signal in frequency domain');
end


% Value changed function: TrackNoEditField
function TrackNoEditFieldValueChanged(app, event)
value = app.TrackNoEditField.Value;
end


% Button pushed function: OriginalSignalPlayerButton
function OriginalSignalPlayerButtonPushed(app, event)
i=app.TrackNoEditField.Value;
app.TextArea.Value=sprintf('Playing Original Signal No.: %d',i);
if i==1
player1= audioplayer(app.x1,Fs);
play(player1);
end
if i==2
player2= audioplayer(app.x2,Fs);
play(player2);
end
if i==3
player3= audioplayer(app.x3,Fs);
play(player3);
end
end


% Button pushed function: ReciviersignalPlayerButton
function ReciviersignalPlayerButtonPushed(app, event)
i=app.TrackNoEditField.Value;
app.TextArea.Value=sprintf('Playing Original Signal No.: %d',i);
if i==1
player1= audioplayer(app.rec1,Fs);
play(player1);
end
if i==2
player2= audioplayer(app.rec2,Fs);
play(player2);
end
if i==3
player3= audioplayer(app.rec3,Fs);
play(player3);
end
end
```

```
end


% Component initialization
methods (Access = private)


% Create UIFigure and components
function createComponents(app)


% Create UIFigure and hide until all components are created
app.UIFigure = uifigure('Visible', 'off');
app.UIFigure.Position = [100 100 640 480];
app.UIFigure.Name = 'MATLAB App';


% Create TypeofInputDropDownLabel
app.TypeofInputDropDownLabel = uilabel(app.UIFigure);
app.TypeofInputDropDownLabel.HorizontalAlignment = 'right';
app.TypeofInputDropDownLabel.Position = [16 339 75 22];
app.TypeofInputDropDownLabel.Text = 'Type of Input';


% Create TypeofInputDropDown
app.TypeofInputDropDown = uidropdown(app.UIFigure);
app.TypeofInputDropDown.Items = {'Recorded Signal', 'Live signal'};
app.TypeofInputDropDown.ValueChangedFcn = createCallbackFcn(app,
@TypeofInputDropDownValueChanged, true);
app.TypeofInputDropDown.Position = [106 339 100 22];
app.TypeofInputDropDown.Value = 'Recorded Signal';


% Create ChoosetheFileButton
app.ChoosetheFileButton = uibutton(app.UIFigure, 'push');
app.ChoosetheFileButton.ButtonPushedFcn = createCallbackFcn(app,
@ChoosetheFileButtonPushed, true);
app.ChoosetheFileButton.Position = [106 185 100 22];
app.ChoosetheFileButton.Text = 'Choose the File';


% Create TextAreaLabel
app.TextAreaLabel = uilabel(app.UIFigure);
app.TextAreaLabel.HorizontalAlignment = 'right';
app.TextAreaLabel.Position = [16 282 56 22];
app.TextAreaLabel.Text = 'Text Area';


% Create TextArea
app.TextArea = uitextarea(app.UIFigure);
```

```
app.TextArea.Position = [87 246 150 60];


% Create OriginalSignalplotButton
app.OriginalSignalplotButton = uibutton(app.UIFigure, 'push');
app.OriginalSignalplotButton.ButtonPushedFcn = createCallbackFcn(app,
@OriginalSignalplotButtonPushed, true);
app.OriginalSignalplotButton.Position = [450 400 116 22];
app.OriginalSignalplotButton.Text = 'Original Signal plot';


% Create ModulatedButton
app.ModulatedButton = uibutton(app.UIFigure, 'push');
app.ModulatedButton.ButtonPushedFcn = createCallbackFcn(app,
@ModulatedButtonPushed, true);
app.ModulatedButton.Position = [457 347 100 22];
app.ModulatedButton.Text = 'Modulated';


% Create SignalinChannelButton
app.SignalinChannelButton = uibutton(app.UIFigure, 'push');
app.SignalinChannelButton.ButtonPushedFcn = createCallbackFcn(app,
@SignalinChannelButtonPushed, true);
app.SignalinChannelButton.Position = [452 282 110 22];
app.SignalinChannelButton.Text = 'Signal in Channel';


% Create BandPassFilteredSignalButton
app.BandPassFilteredSignalButton = uibutton(app.UIFigure, 'push');
app.BandPassFilteredSignalButton.ButtonPushedFcn = createCallbackFcn(app,
@BandPassFilteredSignalButtonPushed, true);
app.BandPassFilteredSignalButton.Position = [430 215 154 22];
app.BandPassFilteredSignalButton.Text = 'Band Pass Filtered Signal';


% Create DemodulatedButton
app.DemodulatedButton = uibutton(app.UIFigure, 'push');
app.DemodulatedButton.ButtonPushedFcn = createCallbackFcn(app,
@DemodulatedButtonPushed, true);
app.DemodulatedButton.Position = [457 149 100 22];
app.DemodulatedButton.Text = 'Demodulated';


% Create ExtractedSignalButton
app.ExtractedSignalButton = uibutton(app.UIFigure, 'push');
app.ExtractedSignalButton.ButtonPushedFcn = createCallbackFcn(app,
@ExtractedSignalButtonPushed, true);
app.ExtractedSignalButton.Position = [456 88 103 22];
app.ExtractedSignalButton.Text = 'Extracted Signal';
```

```
% Create OriginalSignalPlayerButton
app.OriginalSignalPlayerButton = uibutton(app.UIFigure, 'push');
app.OriginalSignalPlayerButton.ButtonPushedFcn = createCallbackFcn(app,
@OriginalSignalPlayerButtonPushed, true);
app.OriginalSignalPlayerButton.Position = [22 67 131 22];
app.OriginalSignalPlayerButton.Text = 'Original Signal Player';


% Create ReciviersignalPlayerButton
app.ReciviersignalPlayerButton = uibutton(app.UIFigure, 'push');
app.ReciviersignalPlayerButton.ButtonPushedFcn = createCallbackFcn(app,
@ReciviersignalPlayerButtonPushed, true);
app.ReciviersignalPlayerButton.Position = [206 67 137 22];
app.ReciviersignalPlayerButton.Text = 'Reciviver signal Player';


% Create TrackNoEditFieldLabel
app.TrackNoEditFieldLabel = uilabel(app.UIFigure);
app.TrackNoEditFieldLabel.HorizontalAlignment = 'right';
app.TrackNoEditFieldLabel.Position = [91 128 54 22];
app.TrackNoEditFieldLabel.Text = 'Track No';


% Create TrackNoEditField
app.TrackNoEditField = uieditfield(app.UIFigure, 'numeric');
app.TrackNoEditField.ValueChangedFcn = createCallbackFcn(app,
@TrackNoEditFieldValueChanged, true);
app.TrackNoEditField.Position = [160 128 100 22];


% Create DurationofRecordingEditFieldLabel
app.DurationofRecordingEditFieldLabel = uilabel(app.UIFigure);
app.DurationofRecordingEditFieldLabel.HorizontalAlignment = 'right';
app.DurationofRecordingEditFieldLabel.Position = [16 443 122 22];
app.DurationofRecordingEditFieldLabel.Text = 'Duration of Recording';


% Create DurationofRecordingEditField
app.DurationofRecordingEditField = uieditfield(app.UIFigure, 'numeric');
app.DurationofRecordingEditField.Position = [153 443 100 22];


% Show the figure after all components are created
app.UIFigure.Visible = 'on';
end
end


% App creation and deletion
methods (Access = public)
```

```
% Construct app
function app = Project1806174


% Create UIFigure and components
createComponents(app)


% Register the app with App Designer
registerApp(app, app.UIFigure)


if nargout == 0
clear app
end
end


% Code that executes before app deletion
function delete(app)


% Delete UIFigure when app is deleted
delete(app.UIFigure)
end
end
end
```

Output:

Here, By taking the number of inputs=3

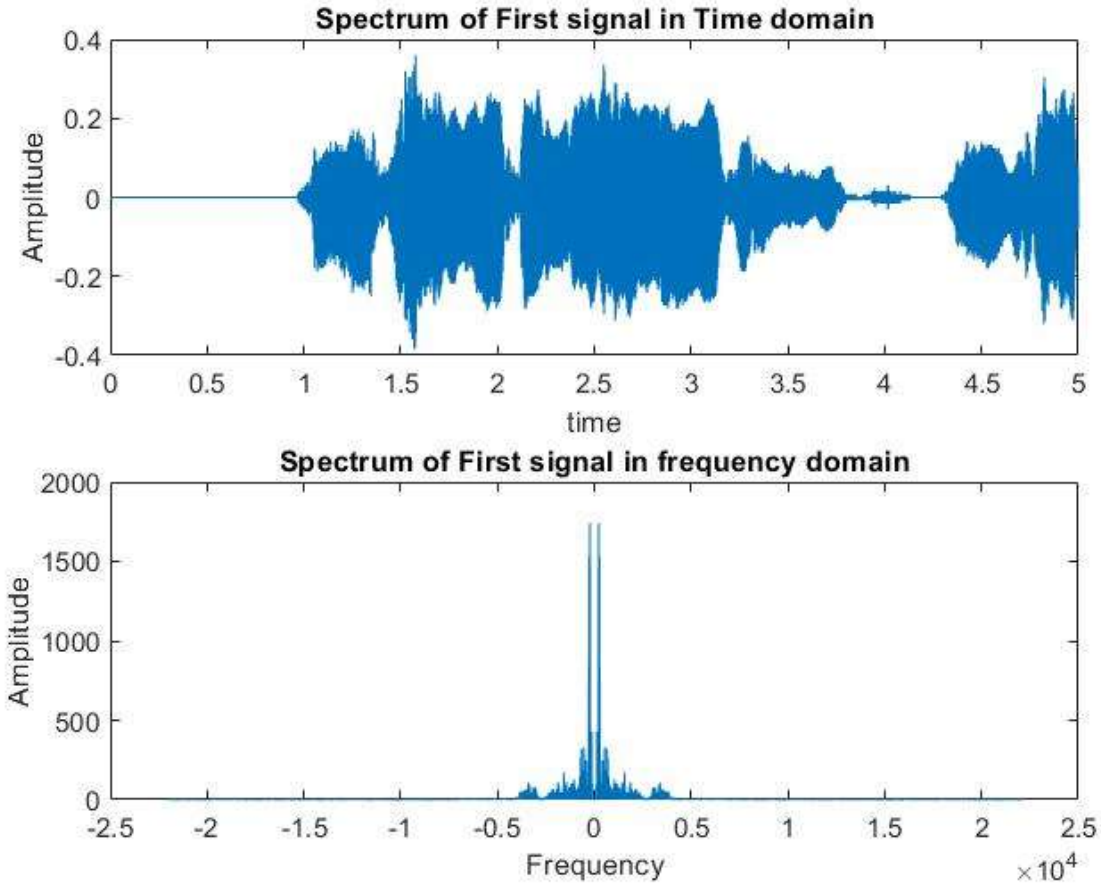
Type of Recording= ‘Real-Time Recording’

And the duration of recording= 3s

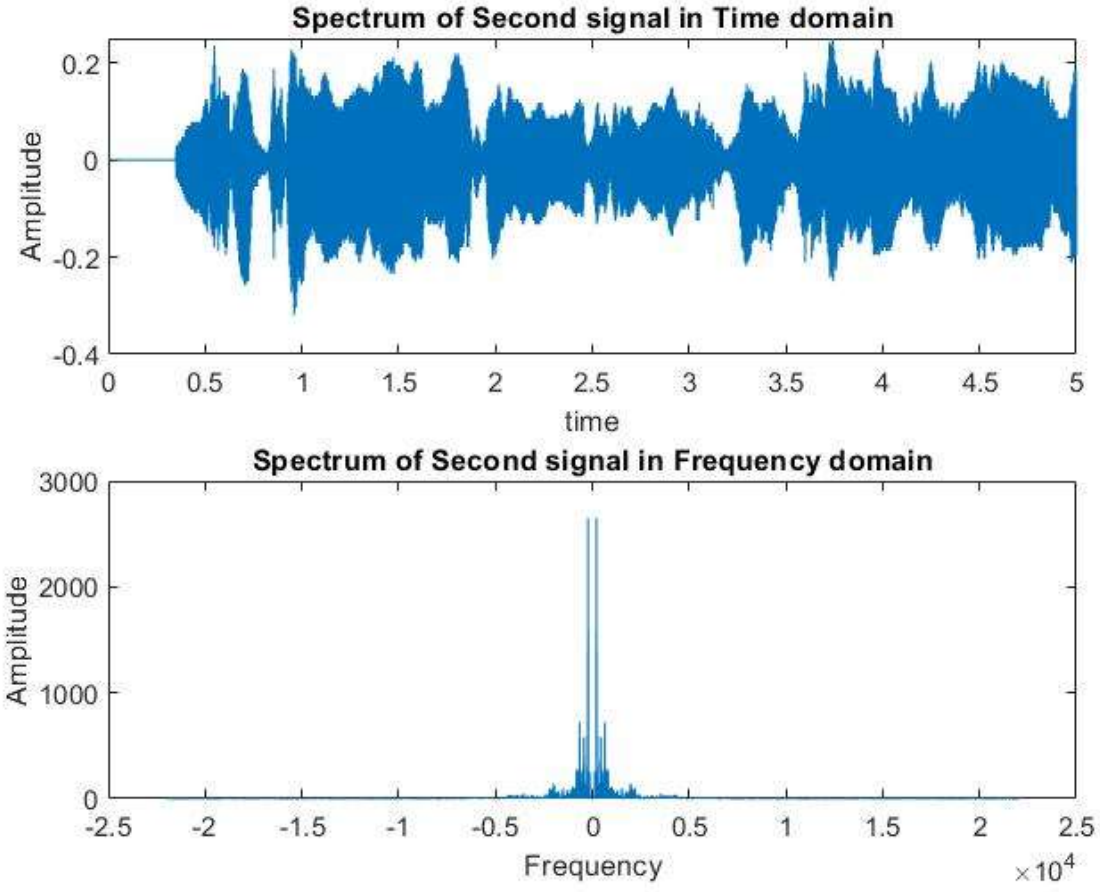
We get the following outputs:

Plots of Original Signal in time and frequency domain:

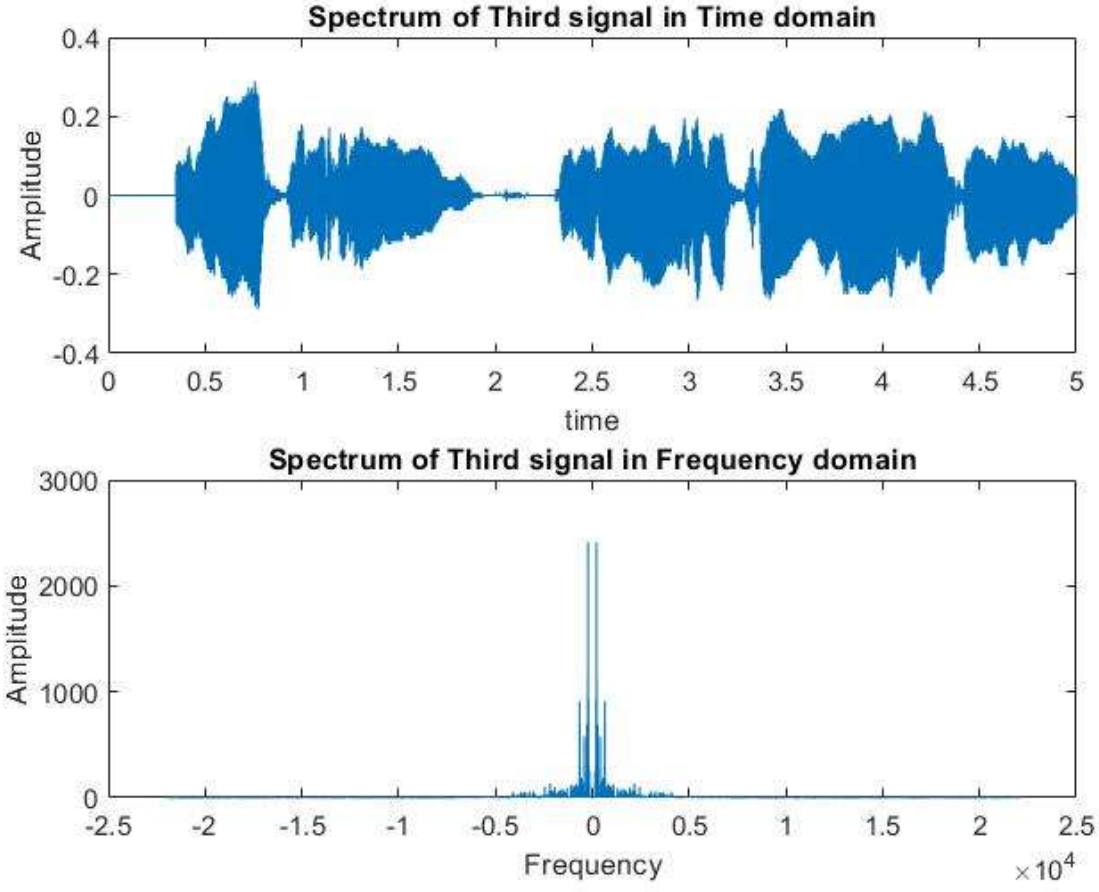
1st Signal:



2nd Signal:

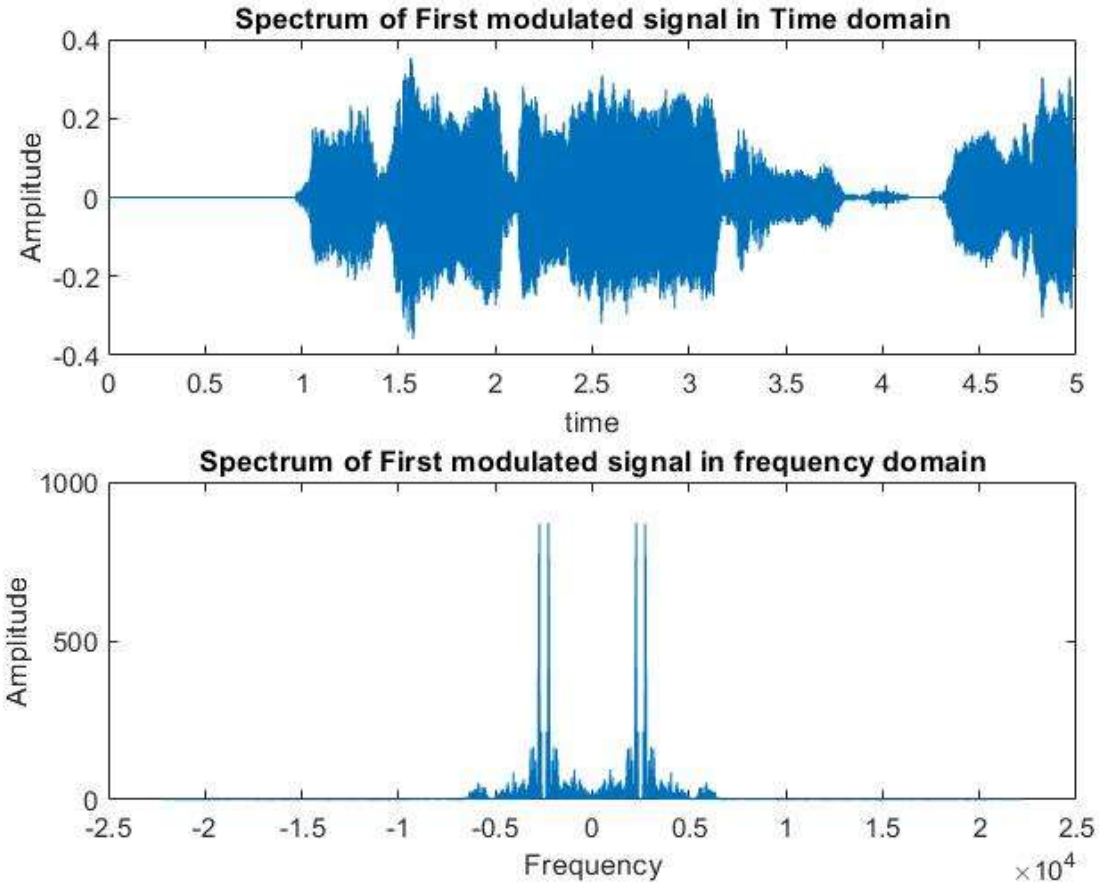

Spectrum of Second signal in Time domain
Amplitude
time
Spectrum of Second signal in Frequency domain
Amplitude
Frequency
×10^4

3rd Signal:

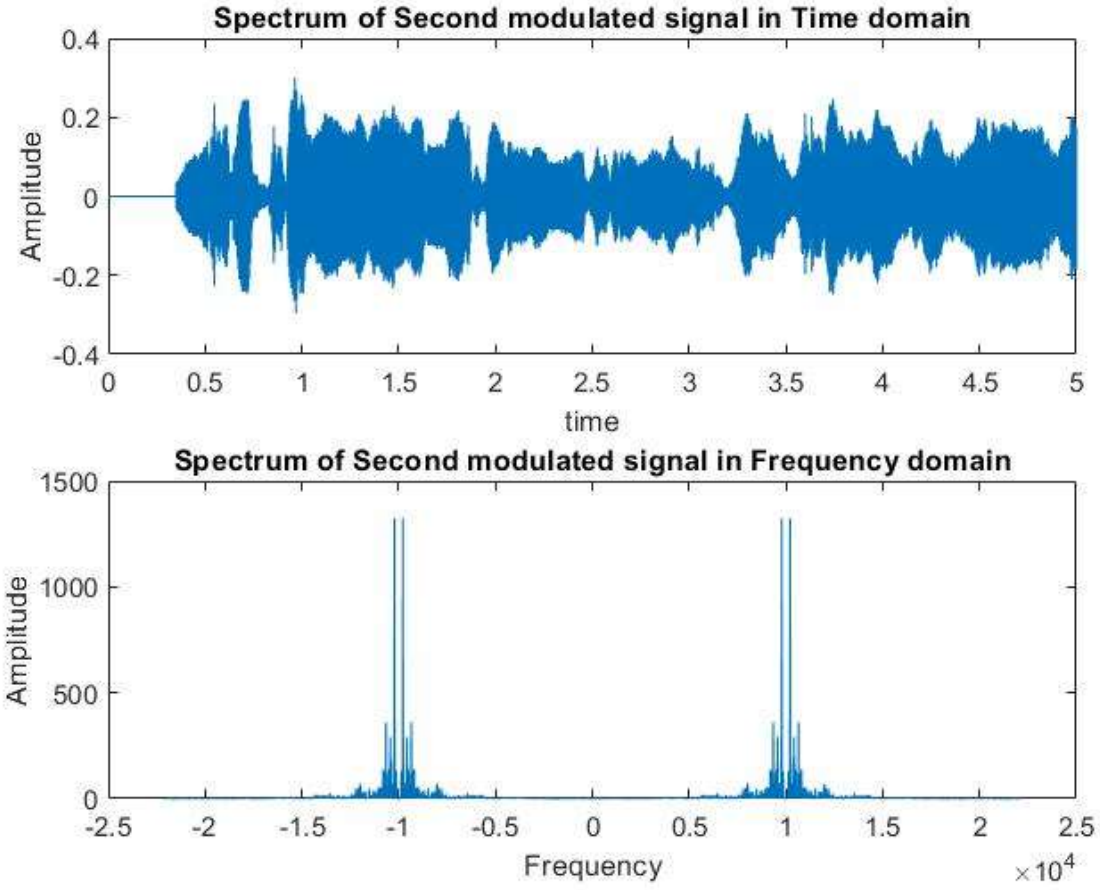

Spectrum of Third signal in Time domain
Amplitude
0.4
0.2
0
-0.2
-0.4
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
time
Spectrum of Third signal in Frequency domain
Amplitude
3000
2000
1000
0
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Frequency
×10^4

Plot of modulated signals in time and frequency domain:

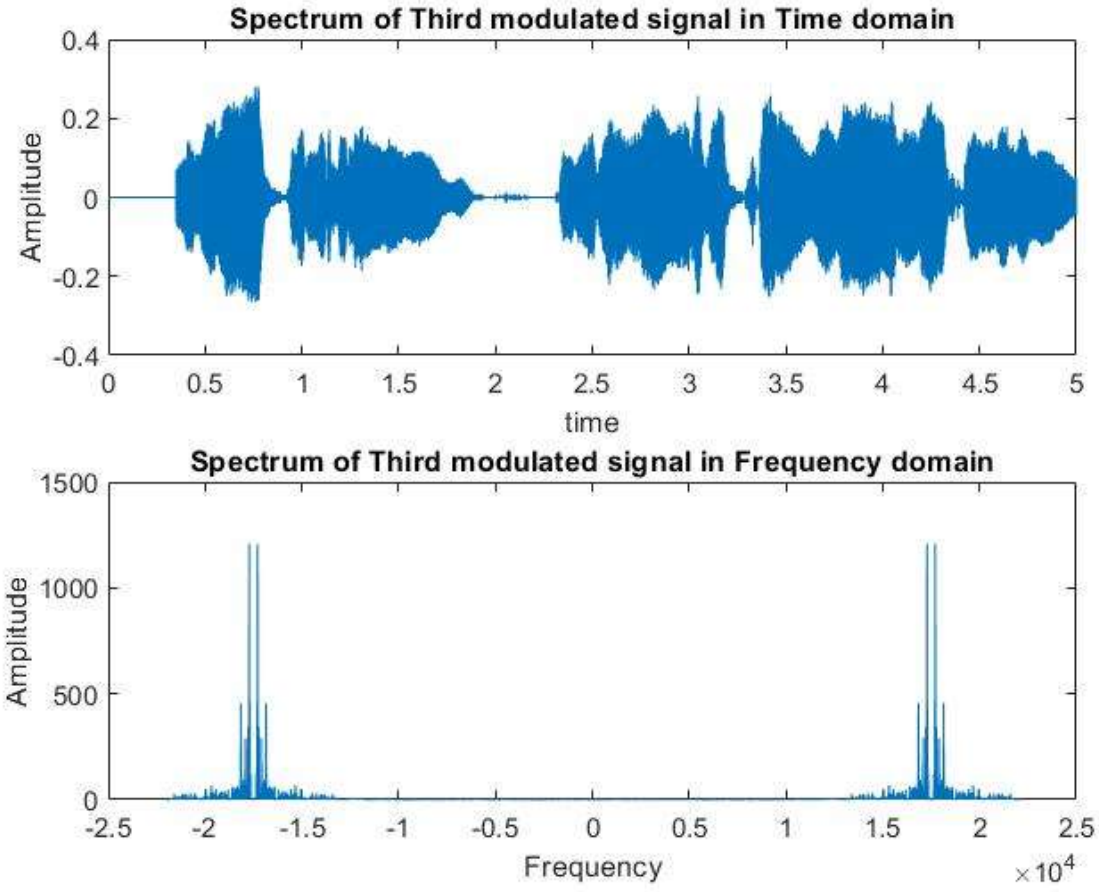
1st Signal:



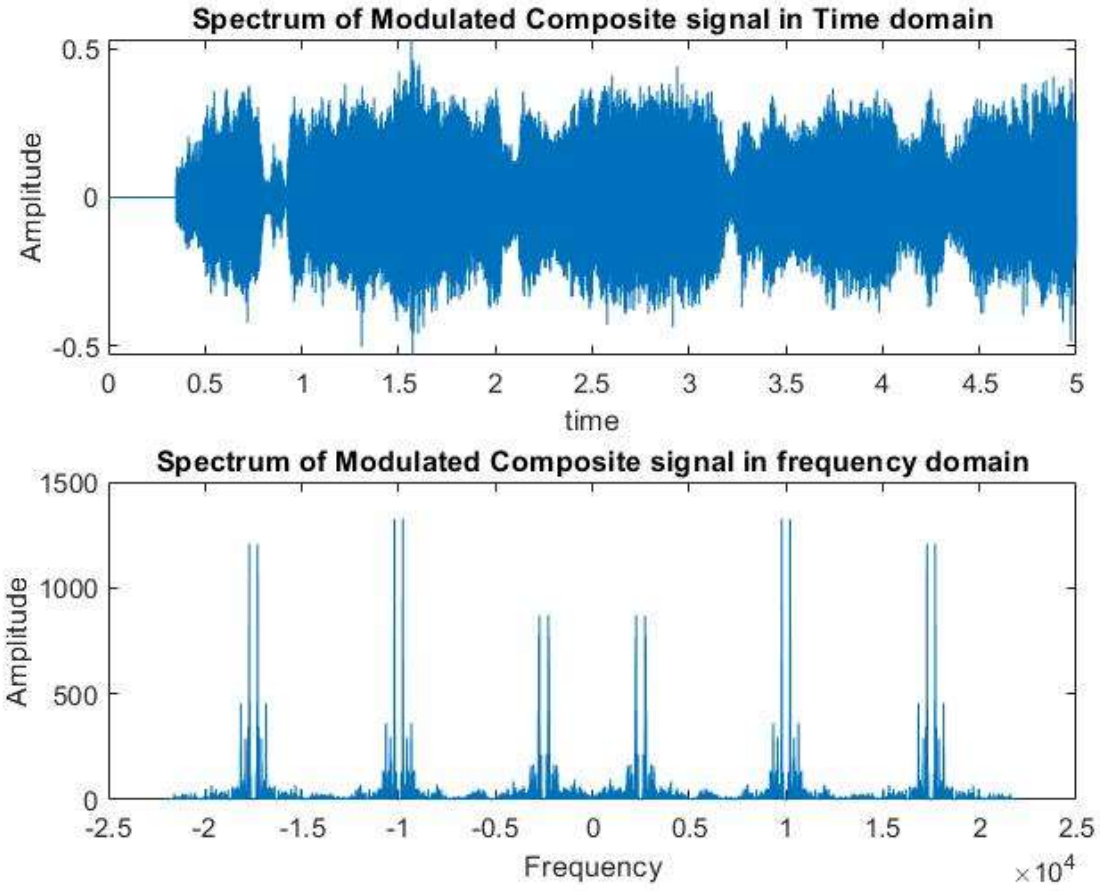
2nd Signal:


Spectrum of Second modulated signal in Time domain
Amplitude
time
Spectrum of Second modulated signal in Frequency domain
Amplitude
Frequency
×10^4

3rd Signal:

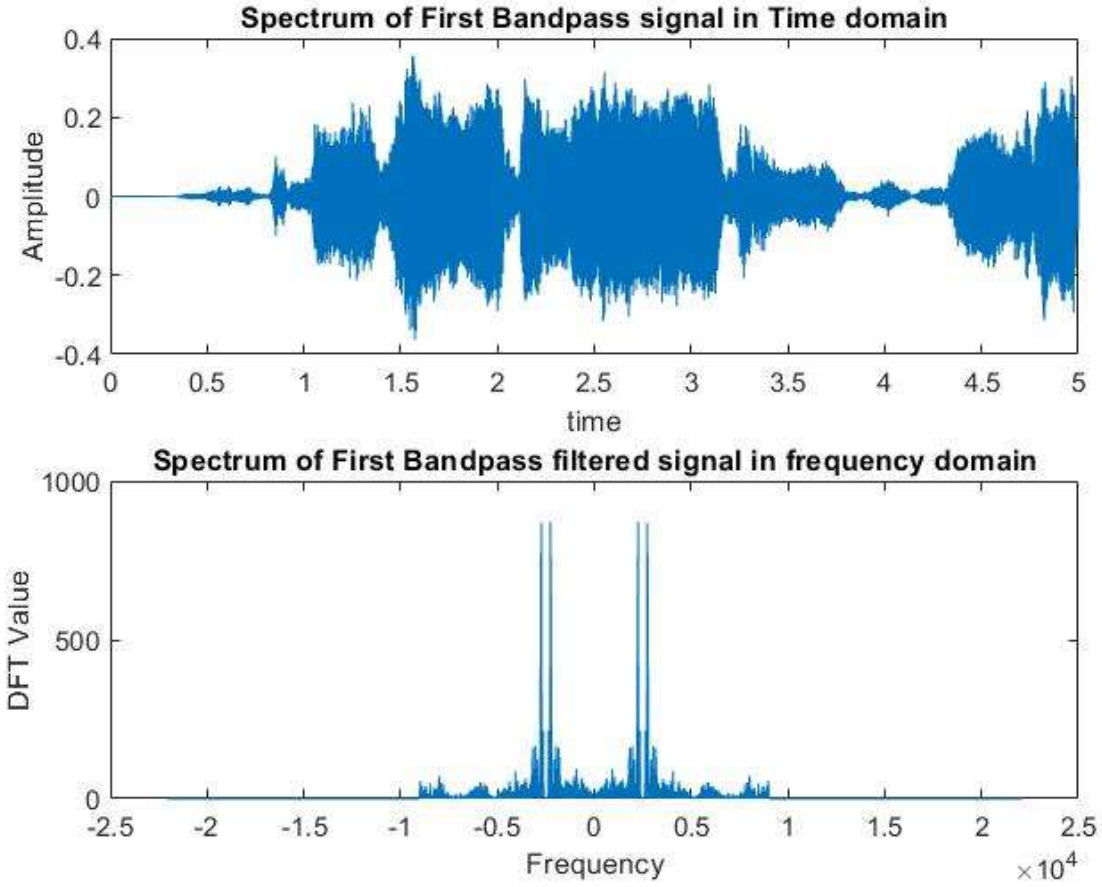

Spectrum of Third modulated signal in Time domain
Amplitude
time
Spectrum of Third modulated signal in Frequency domain
Amplitude
Frequency
×10^4

Plot of signals in Channel:

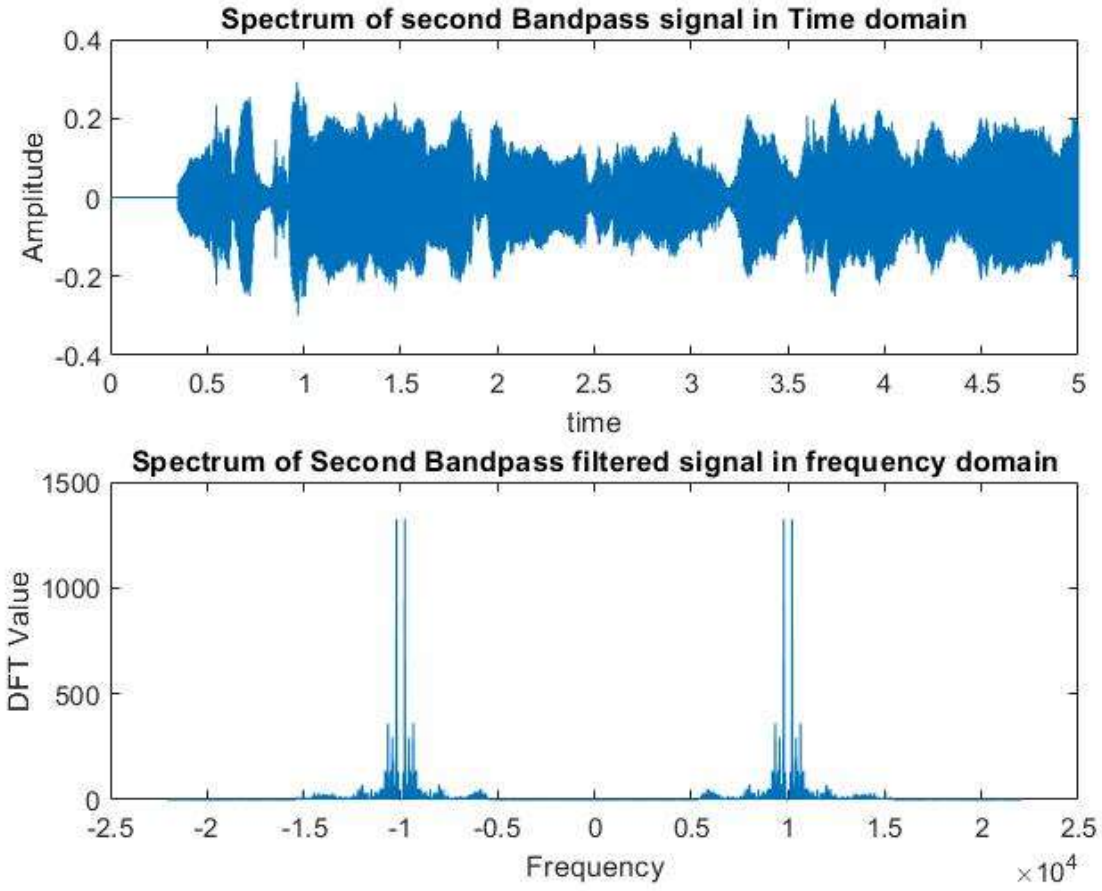


Plot of bandpass filtered signals in time and frequency domain:

1st Figure :



2nd Figure :


Spectrum of second Bandpass signal in Time domain
Amplitude
time
Spectrum of Second Bandpass filtered signal in frequency domain
DFT Value
Frequency
×10^4

3rd Figure:

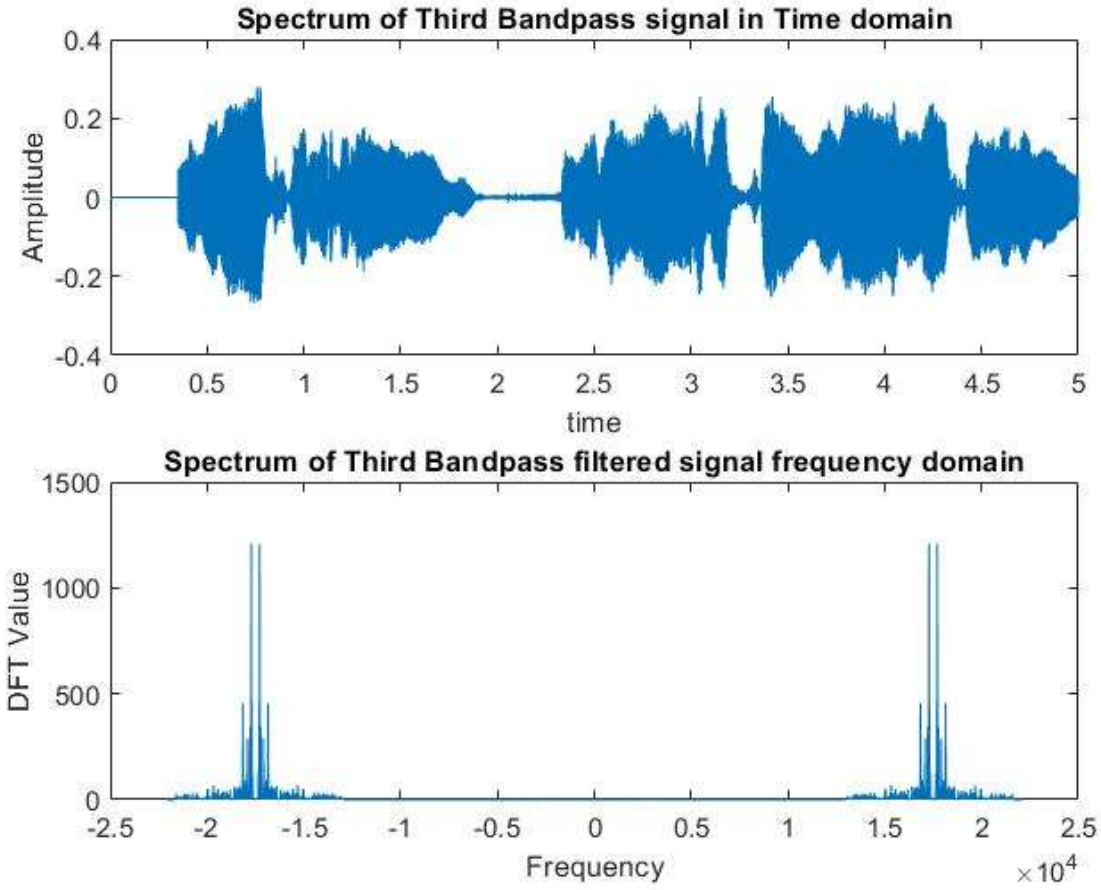
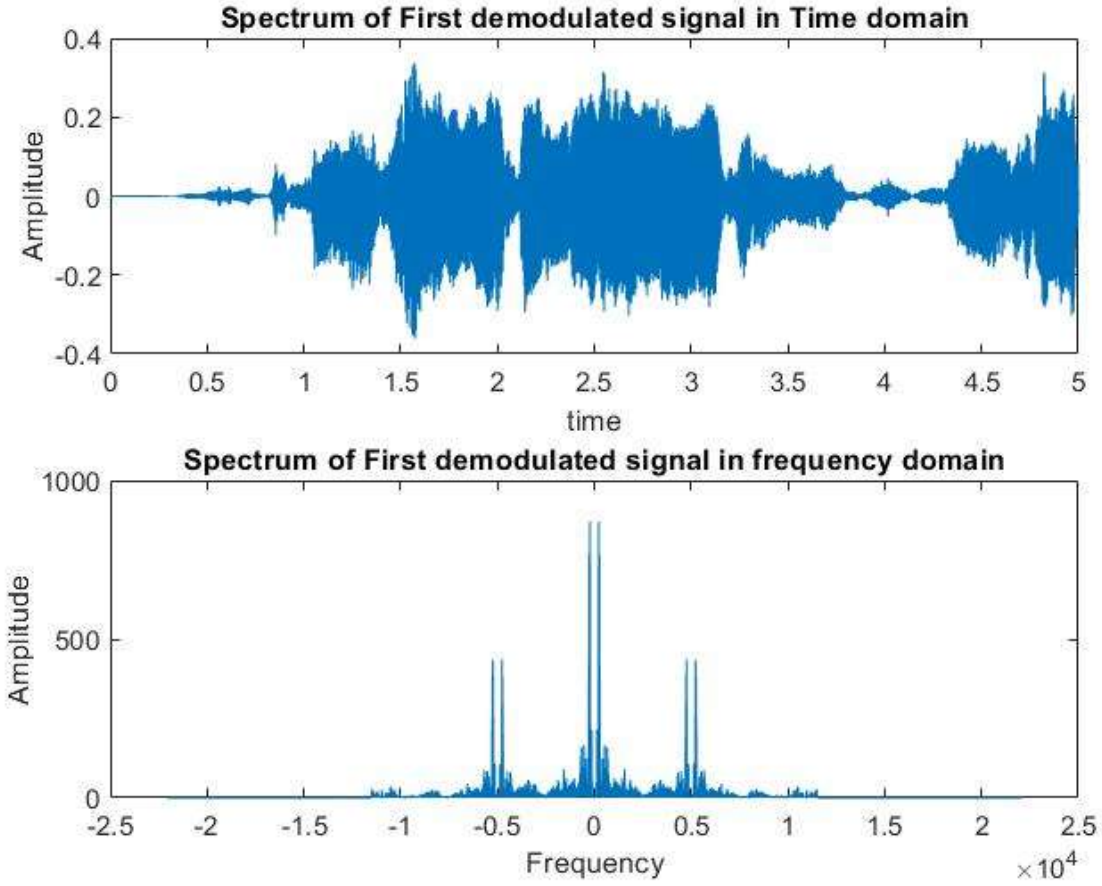


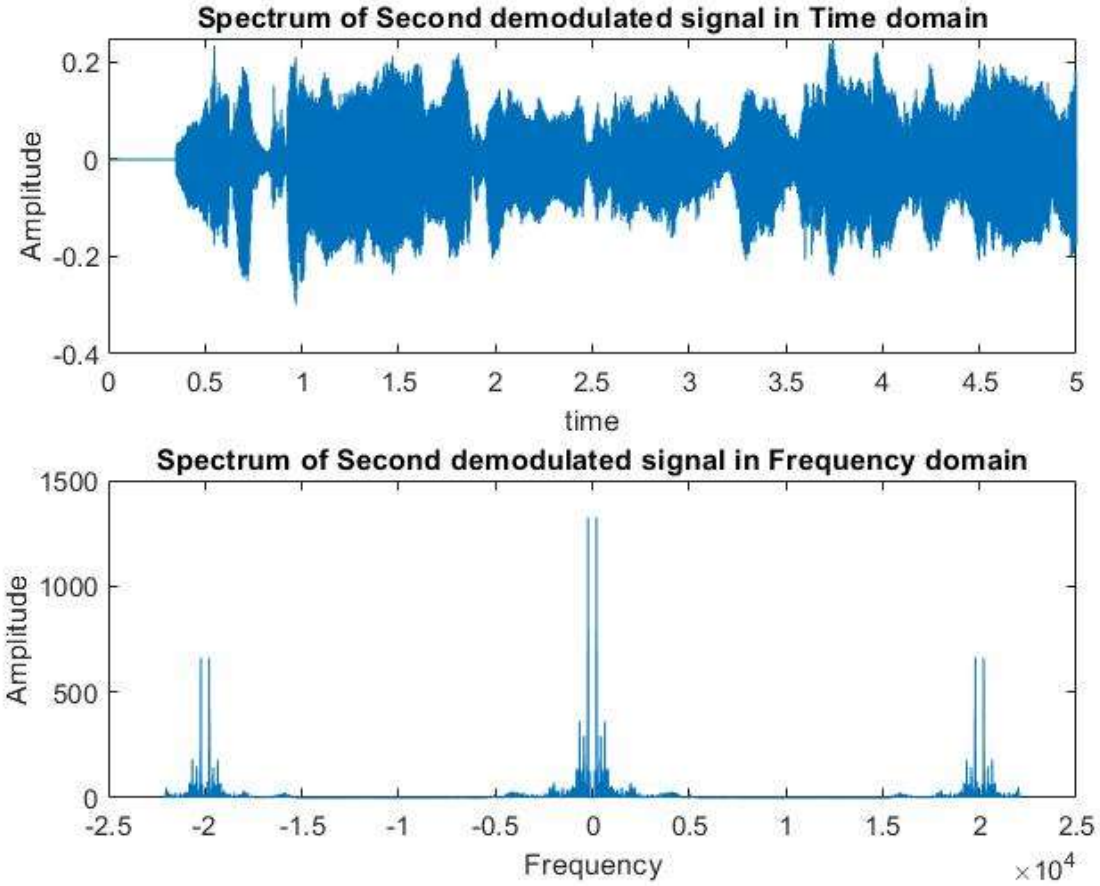
Figure in frequency domain:

Plot of demodulated signals in time and frequency domain:

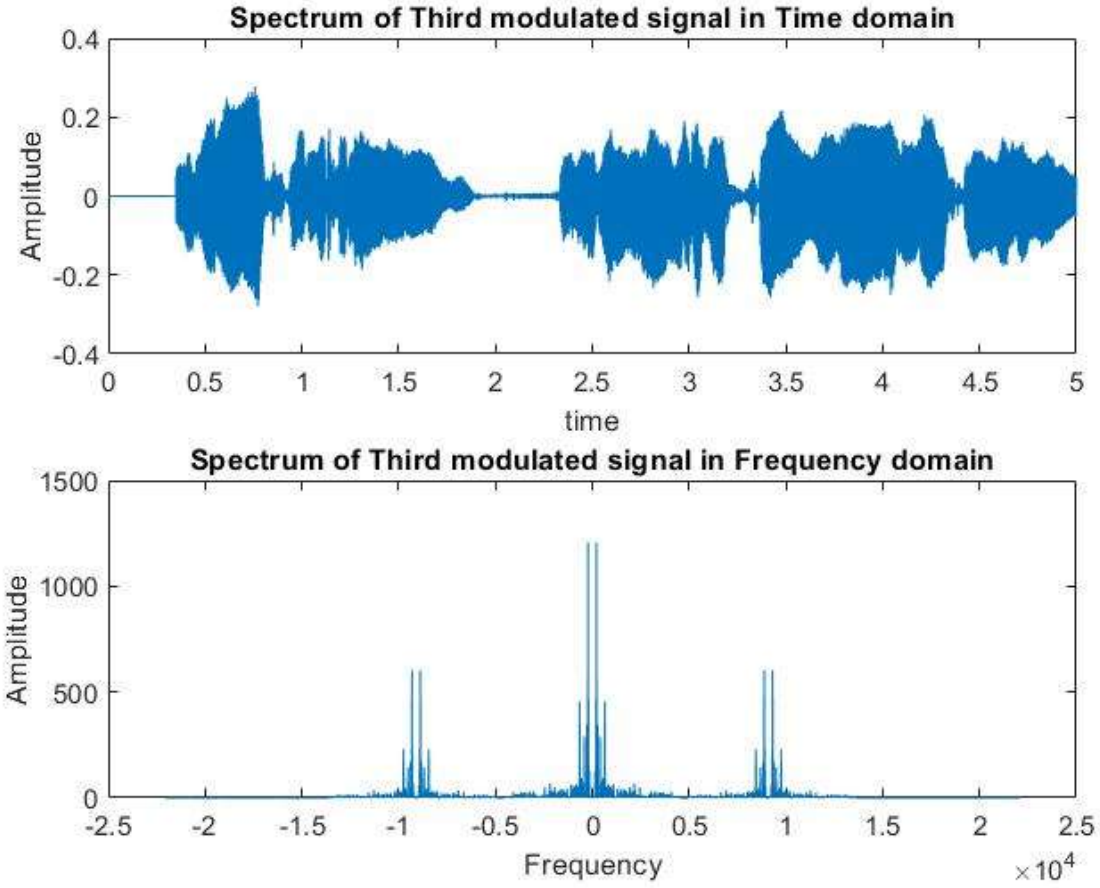
1st Figure :


Spectrum of First demodulated signal in Time domain
Amplitude
time
Spectrum of First demodulated signal in frequency domain
Amplitude
Frequency
×10^4

2nd Figure :

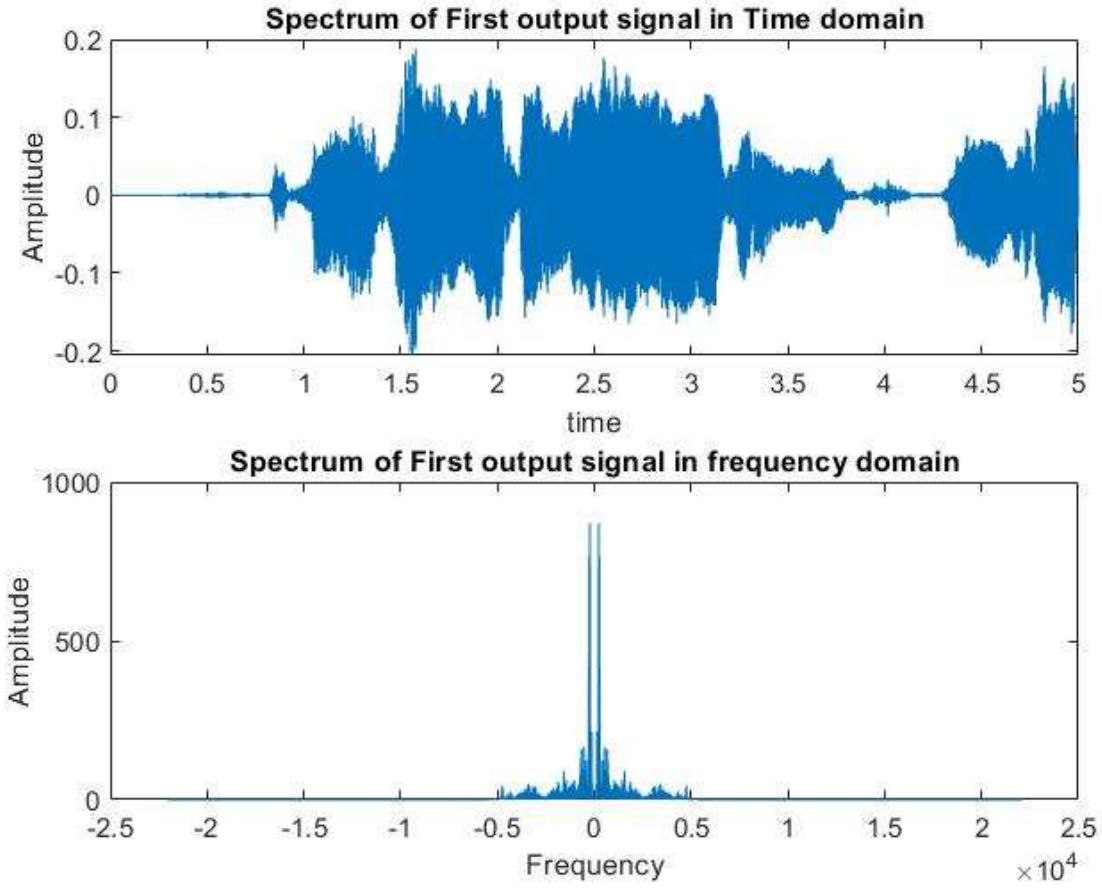

Spectrum of Second demodulated signal in Time domain
Amplitude
time
Spectrum of Second demodulated signal in Frequency domain
Amplitude
Frequency
×10^4

3rd figure:



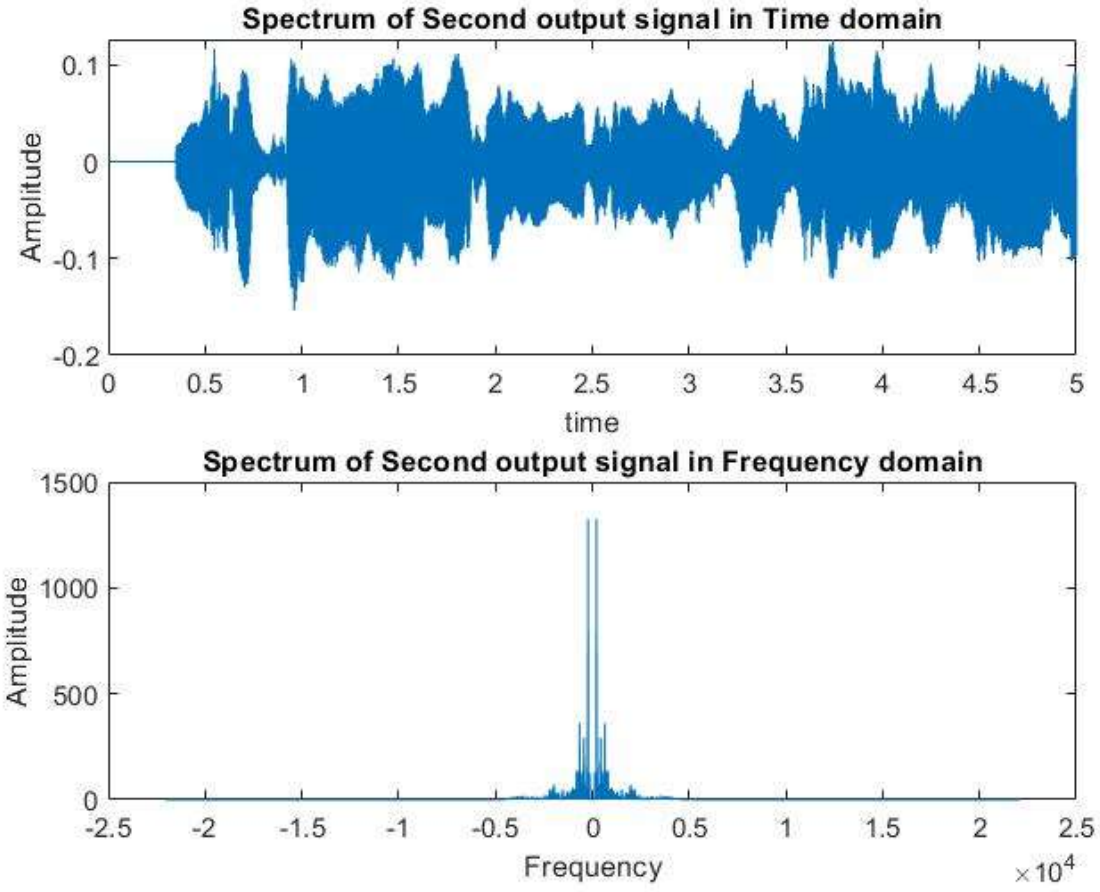
Plot of retrieved signals in time and frequency domain:

1st Figure

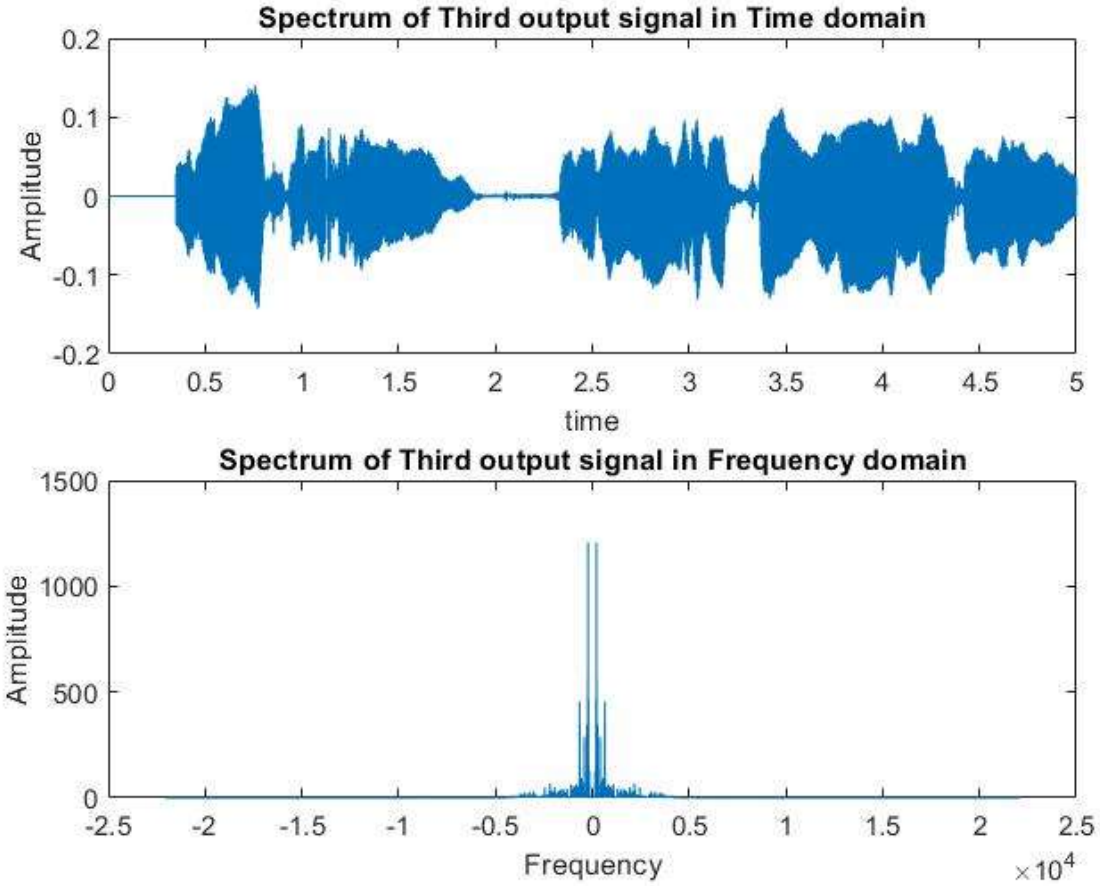

Spectrum of First output signal in Time domain
Amplitude
time
Spectrum of First output signal in frequency domain
Amplitude
Frequency
×10^4

:

2nd Figure:



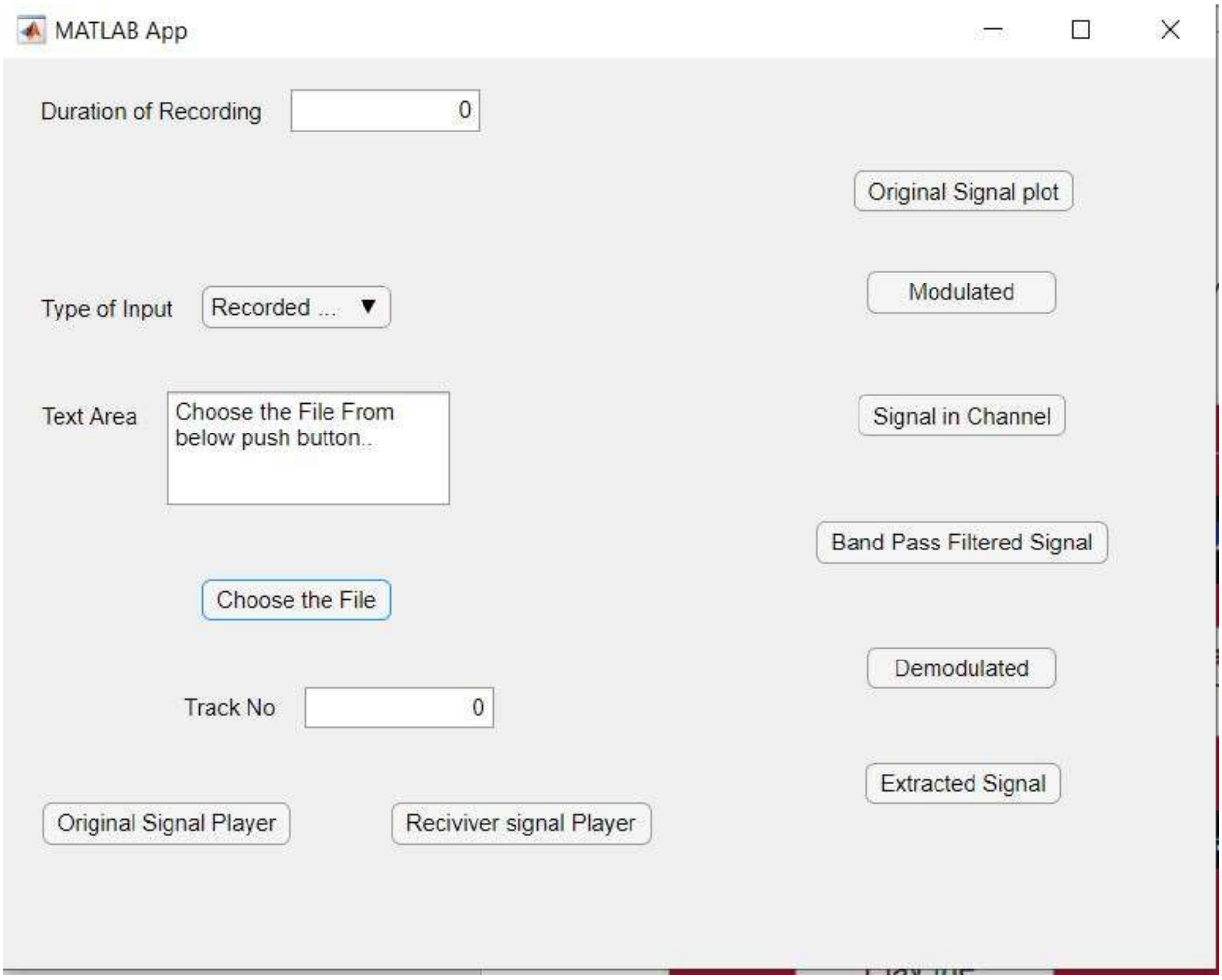
3rd Figure



These are the output plots from every blocks of execution.

Discussion:

This project is related to frequency division multiplexing. To do so, we have take inputs of three audio file here. Above I have showed the output plots for real time recording . This can also be done by recorded audio mp3/ wav file by choosing ‘Recorded mp3/WAV File’ from the app window.



We can play the original and retrieved audios by clicking the buttons of play and selecting the number of recording’s we want to play.

In the code, for recording we have taken the sample frequency 441000 Hz. We have also taken the frequencies of carrier signals and modulated

them. Then we passed the modulated signal by adding them together in a channel. Then we filtered them by bandpass filters and demodulated them by multiplying filtered signals with carrier signal. And finally, we retrieved the signals by passing demodulated signals in the lowpass filters.

During this project I have also face some difficulties -

1) The first difficulty was in choosing carrier frequency so that the dimention of inpuy signal's matrix matches with the carrier signal dimention's.

2) The next challange I faced in designing bandpass filter and lowpass filter. It was difficult to find out the cutoff frequency for each audio signals.

3)In order to implement the task, we needed to use some variables in callback function of more than one pushbutton. But usually we can't use same variable in different callback functions.

But I have overcome all these difficulties and completed the project.

Future Prospect of Improvement:

There is possibilities of improvement of this project in some angles like-

1) These project can take inputs of 3. So there is a possibility to improve these program more in future so that the system can take more signals as input if there is enough time to explore.

2) Then in the retrieved audio the amplitude of audio signal is not the same. So some amplification of the signal can be done there to have the same sound in the output.

3) Again we face problem for recording of very large duration. Like in the output the recording was for 5 seconds. It can be improved so that the system can take audios of large duration.