Cairo University

Faculty of Engineering

Electronics & Communication Department

Communication Systems ELC3020

MATLAB Implementation of Super-Heterodyne Receiver

Name	**Shehab El-Din Tarek Foaad**
Sec	**2**
BN	**24**
ID	**9220392**

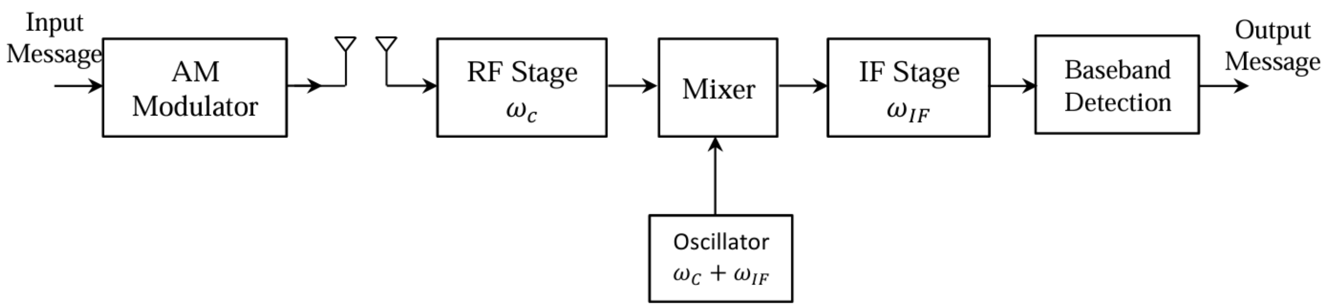
Under Supervision : Dr/ Ahmed Mehana

Table of Contents

Table of figures

i. Discussion :

➢ Block Diagram :



➔ AM Modulator (Tx):

- This stage is designed by these steps :
    1. Reading the Messages with its mean value to convert it to **one channel signal** instead of **stereo signal** & knowing their Sampling Frequency
    2. Apply on each message Fourier Transform to transform them into the Frequency Domain to know their Baseband Bandwidth
    3. Creating multiple of **Cosine Functions** to be **Carriers Signals for Modulation** , Each Carrier Signal is oscillating at a specific frequency from formula $\boldsymbol{\omega_n = 100 + n\Delta F}$, where $\boldsymbol{\Delta F = 50}$ KHz and **the index $\boldsymbol{n}$** is the signal index
    4. Adding All the Modulated messages in the **Time Domain Channel**
    5. Determining of the **Nyquist Sampling Frequency** to be equal to double the maximum frequency at the FD Spectrum **(Nyq Fs ≈ 2 * 322Khz ≈ 644Khz)**

- **The Spectrum of the Output of the AM Modulator Tx:**

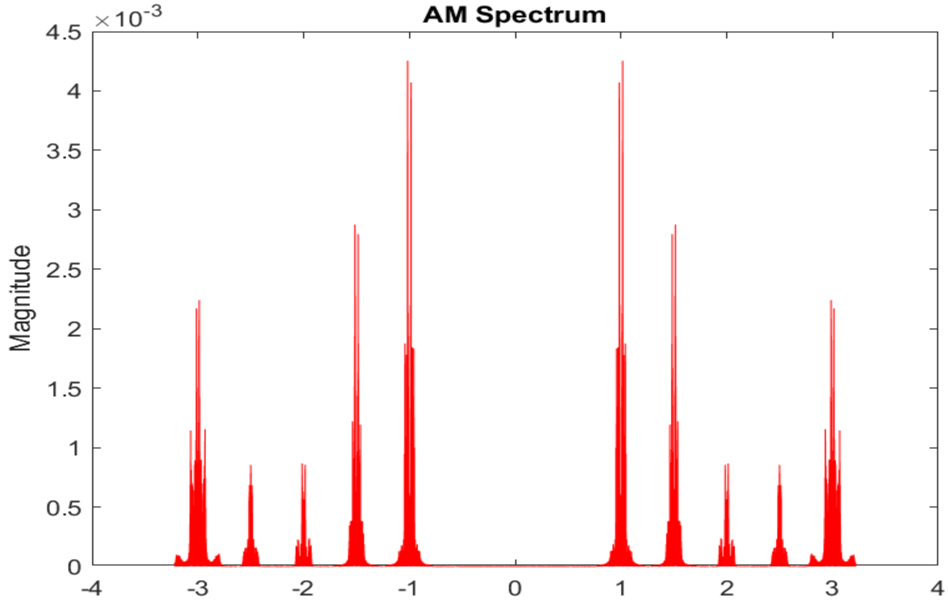


Figure 1

- **The Time Domain Channel of the Output of the AM Modulator Tx:**

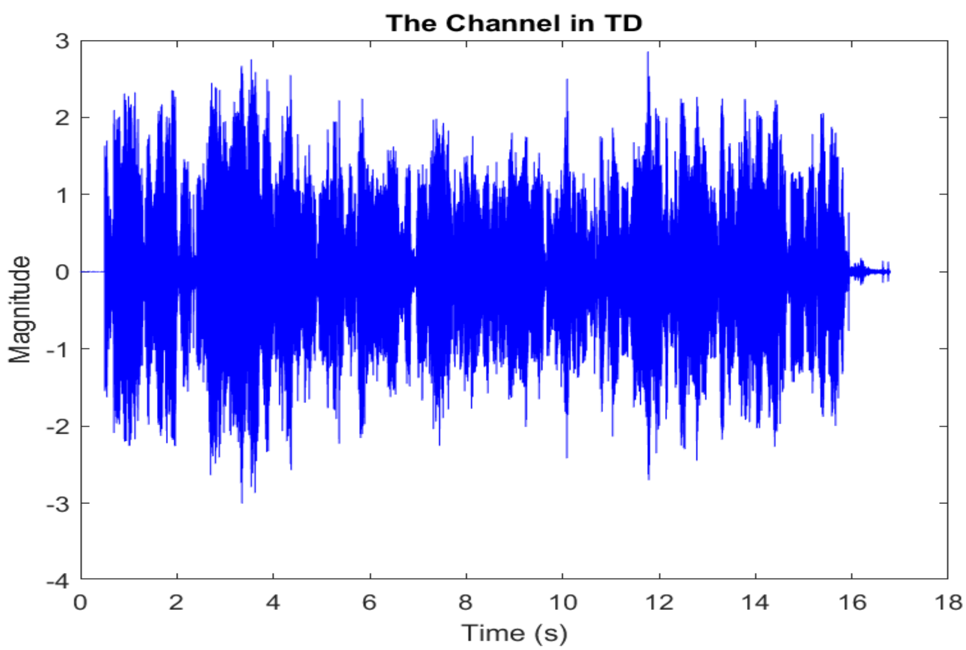


Figure 2

➔RF Stage :

- This stage is designed by these steps :

1. Designing **Tunable Band Pass Filter** for filtering the required receiving message from the channel
2. The RF BPF is tuned to be centered at the carrier frequencies with **Width ≈ 44Khz** which is the BW of the widest message in the transmitted messages to filter it regularly

- **The Spectrum of the Output of the RF BPF (Before The Mixer):**

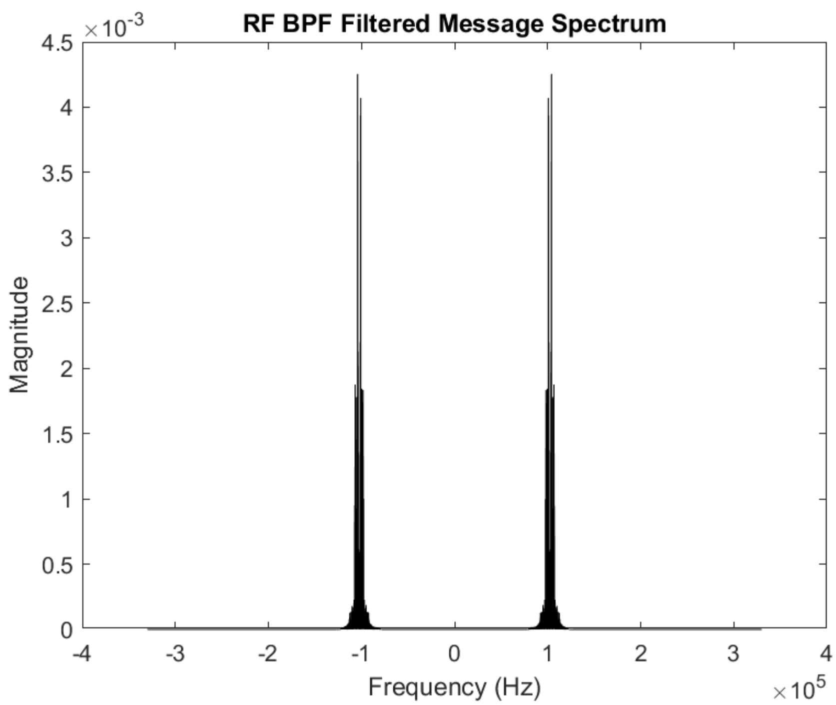


Figure 3

- **The Output of the RF BPF (Before The Mixer) in TD:**

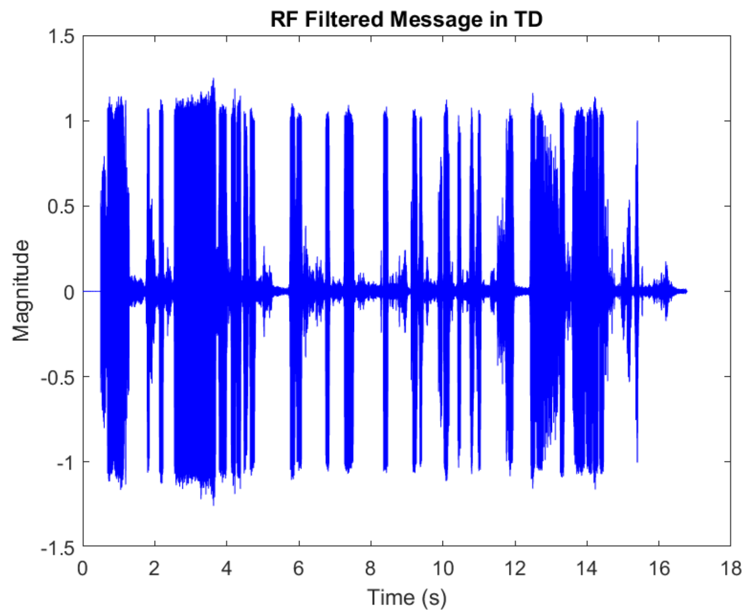


Figure 4

➔ Mixer with Oscillator (ω_c+ω_{IF}):

- This stage is designed by this step:
  1. **Demodulate** the RF Filtered Message by mixing it with **Oscillator oscillates @ (ω_c+ω_{IF})** where

     ($\omega_{IF} = 25\text{KHz}$) as a cosine function to upconvert the center frequency of the RF Message to ω_{IF} Instead of Baseband

- **The Spectrum of the Output of the RF BPF (After The Mixer):**

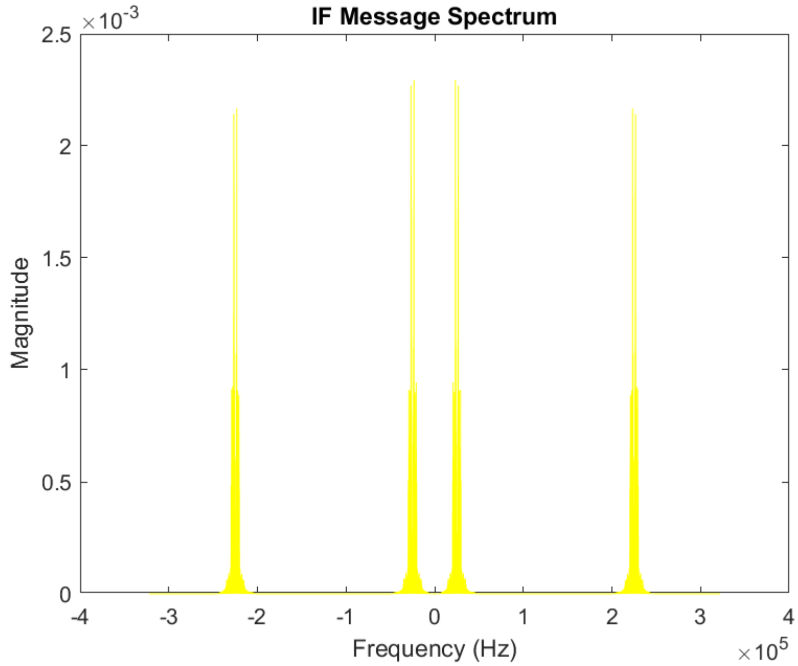


Figure 5

- **The Output of the RF BPF (After The Mixer) in TD:**

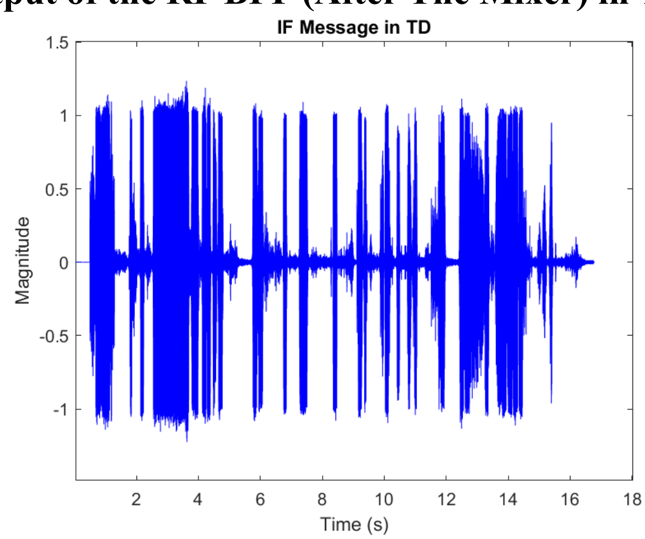


Figure 6

➔IF BPF:

- This stage is designed by these steps :

    1. Designing a **BPF centered at ($\omega_{IF} = 25$KHz)** & with **Width ≈ 44Khz** which is the BW of the widest message in the transmitted messages to filter it regularly without the **Image Signal** appeared in the spectrum
    2. **Demodulate** the message by mixing it with **oscillating frequency (ω_{IF})** as a cosine function to downconverte the message back to the **BaseBand**

- **The Spectrum of the Output of the IF BPF (Before BaseBand Mixer):**

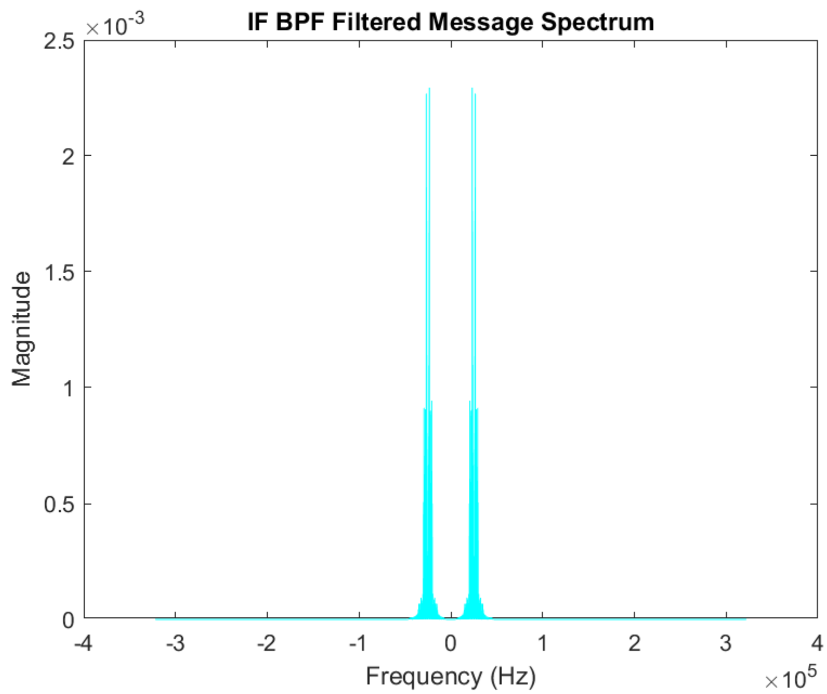


Figure 7

- **The Output of the IF BPF (Before BaseBand Mixer) in TD:**

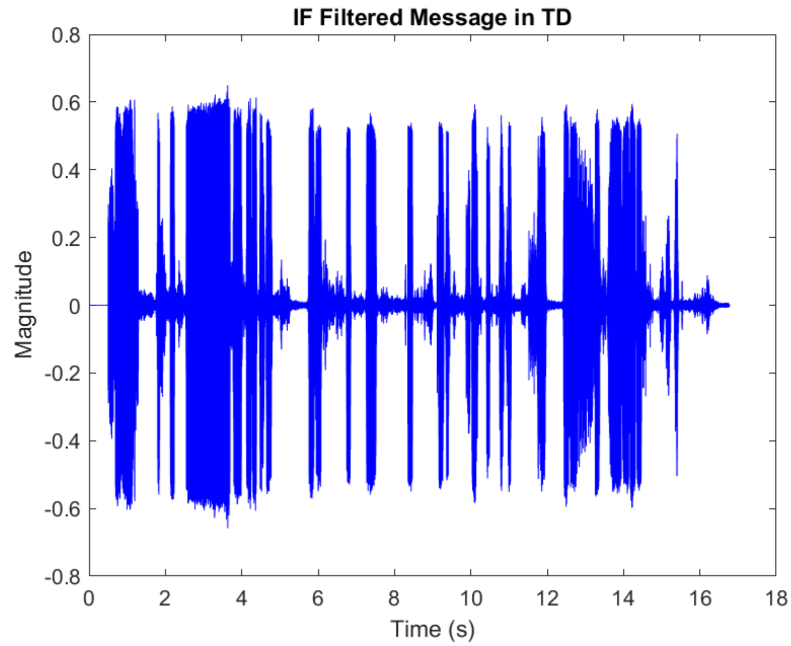


Figure 8

- **The Spectrum of the Output of the IF BPF (After BaseBand Mixer):**

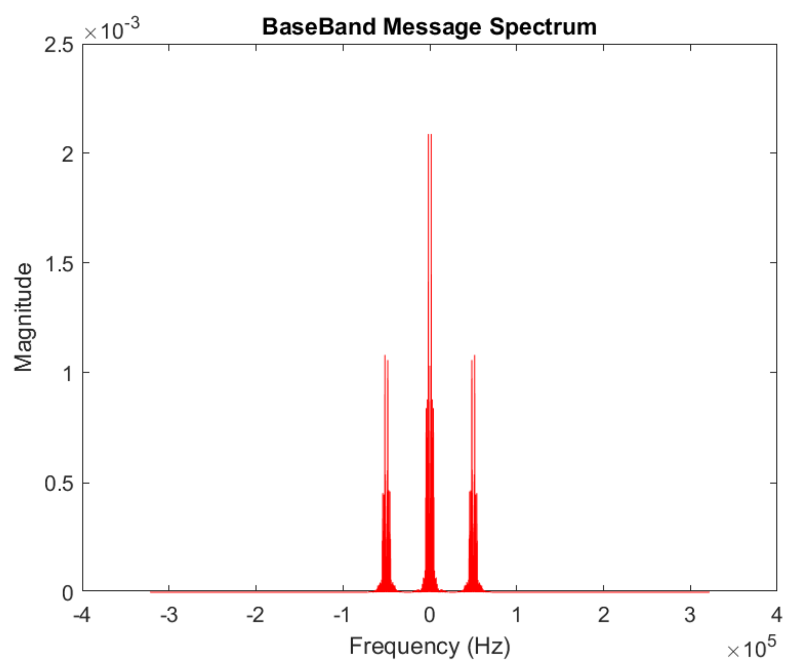


Figure 9

- **The Output of the IF BPF (After BaseBand Mixer) in TD:**

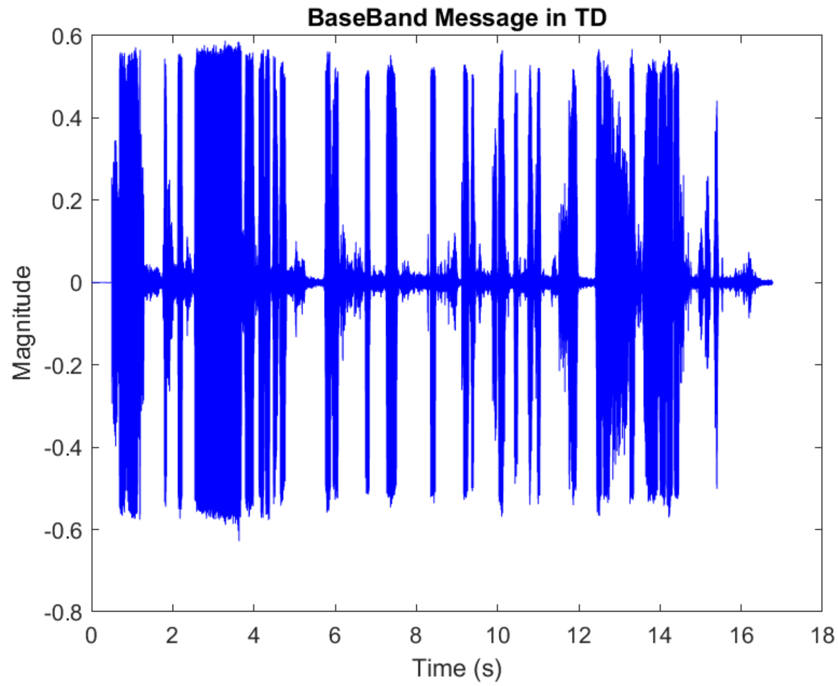


Figure 10

➔Baseband Detection:

- This stage is designed by these steps :
    1. Designing a **LPF with CuttOff Frequency = 50KHz** which is more than the BW of the widest message in the transmitted messages to filter it regularly
    2. **Multiply** the filtered message by **2** to regain the power of the message after demodulation & **Resampling** the filtered message with the original sampling frequency of the message which was **44.1KHz** to Receive & Read it regularly

- **The Spectrum of the Output of the BB Detection :**

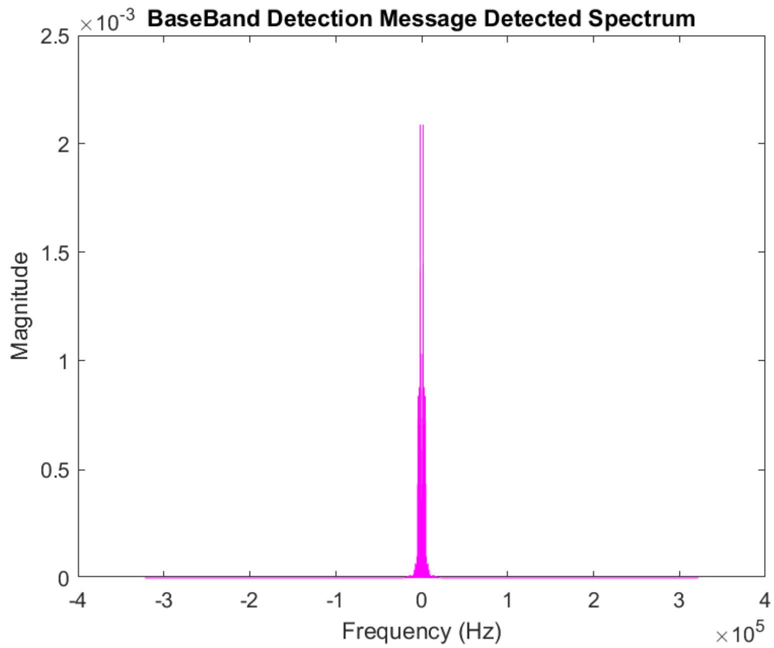


Figure 11

- **The Output of the BB Detection in TD :**

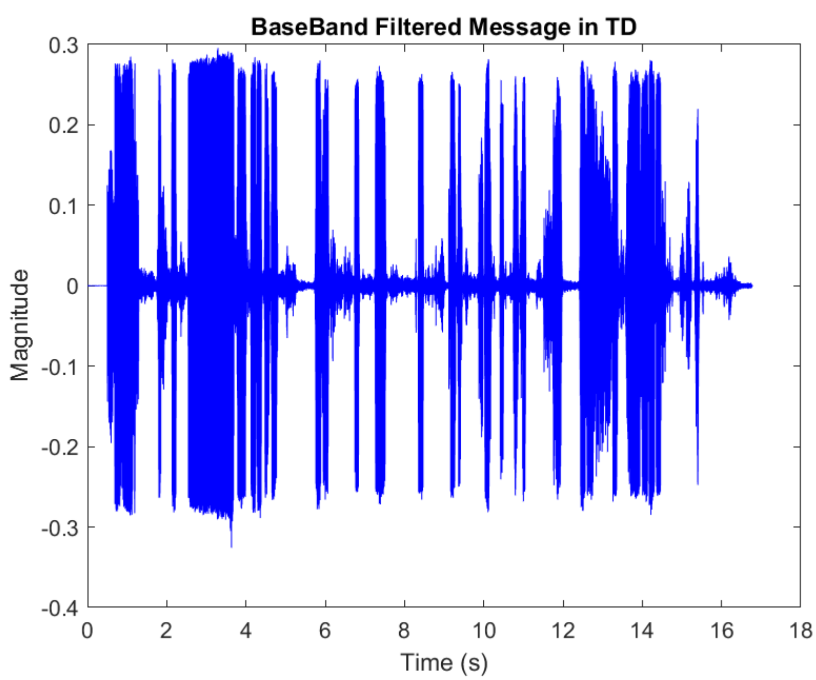


Figure 12

- **The Output Message After DownSampling to 44.1KHz in TD :**

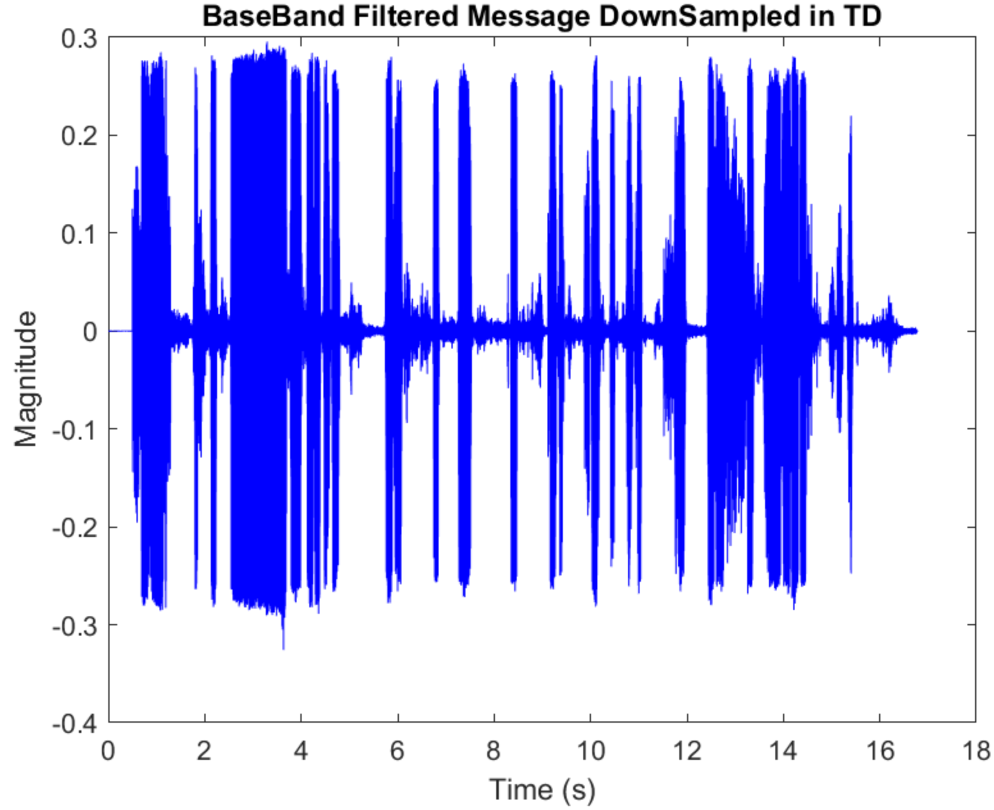


Figure 13

➔The Role of Filters :

1. **RF Stage :** To filter the required message out of the spectrum to prevent receiving the other messages in the channel .
2. **IF Stage :** To Demodulate the received message at ω_{IF} to prevent receiving the message at the BaseBand to protect the following block from the DC component of the signal to prevent saturating and prevent the probability of damaging due to the DC input power of the signal , And Filtering the Image signal appeared at ($\omega_o + 2 * \omega_{IF}$).

3. **BaseBand Detection Stage :** To Demodulate the received message to the BaseBand & Filter the Message at the BB by LPF with width more than the BB BW of the Message with gain = 2 to regain the message power , Then DownSampling the message to 44.1KHz to listen it regularly.

➔ Adding Noise To Message :

- After adding noise using AWGN Function the Message Sound is distorted by a noise along the audio message by adding noise power on the whole BW of message spectrum

➢ **Original Message vs Noisy Message (TD):**

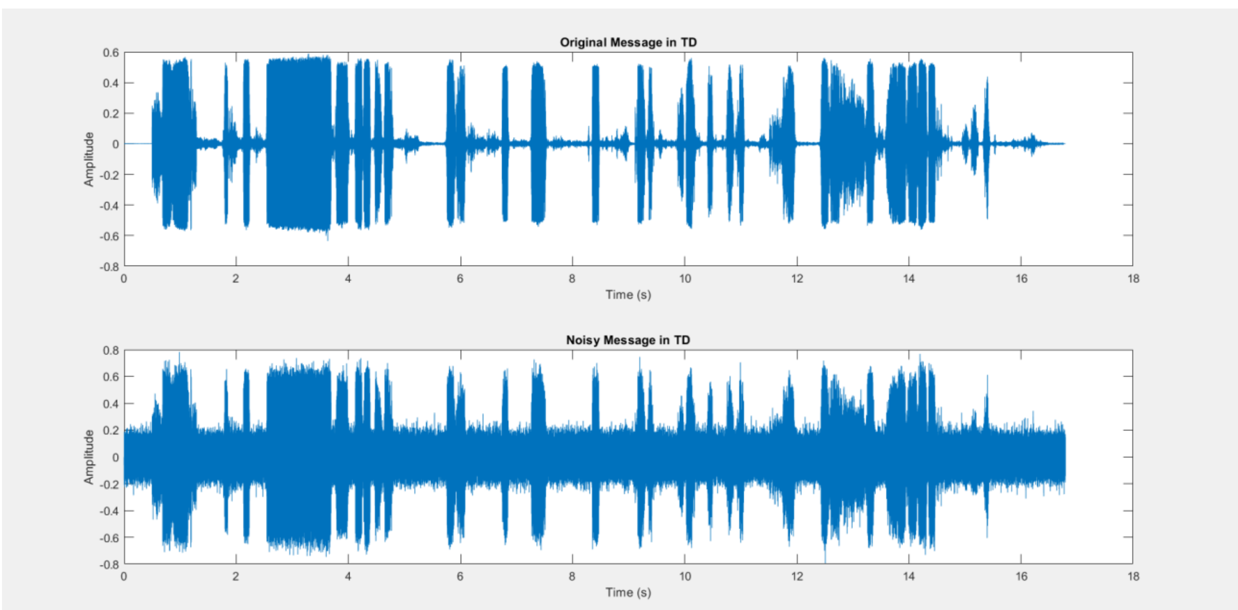


Figure 14

- **Original Message vs Noisy Message (FD) :**

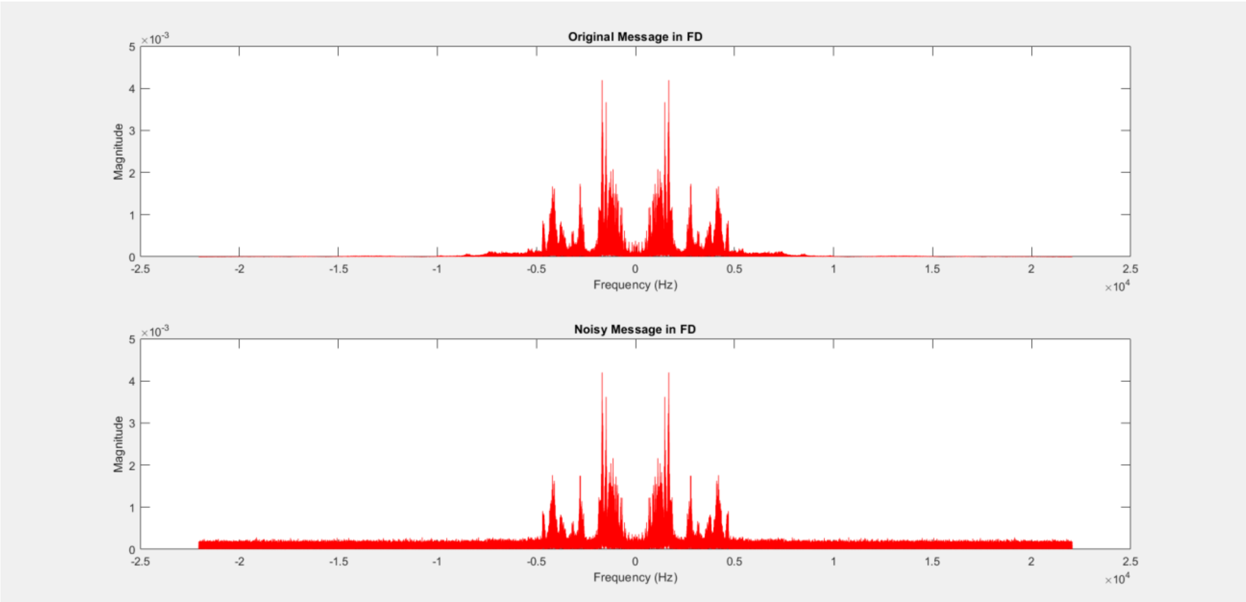


Figure 15

➔Removing the RF Stage:

- After removing the RF Stage ,The Message @100KHz is overlapping and interfering with the Message @150KHz & the Output Sound is playing both messages at the same time.

- All The Message in The Chennel Spectrum is shifted ±25KHz Around the Carrier Frequency.

- **The Output Spectrum After The RF Mixer without RF BPF (in FD) :**

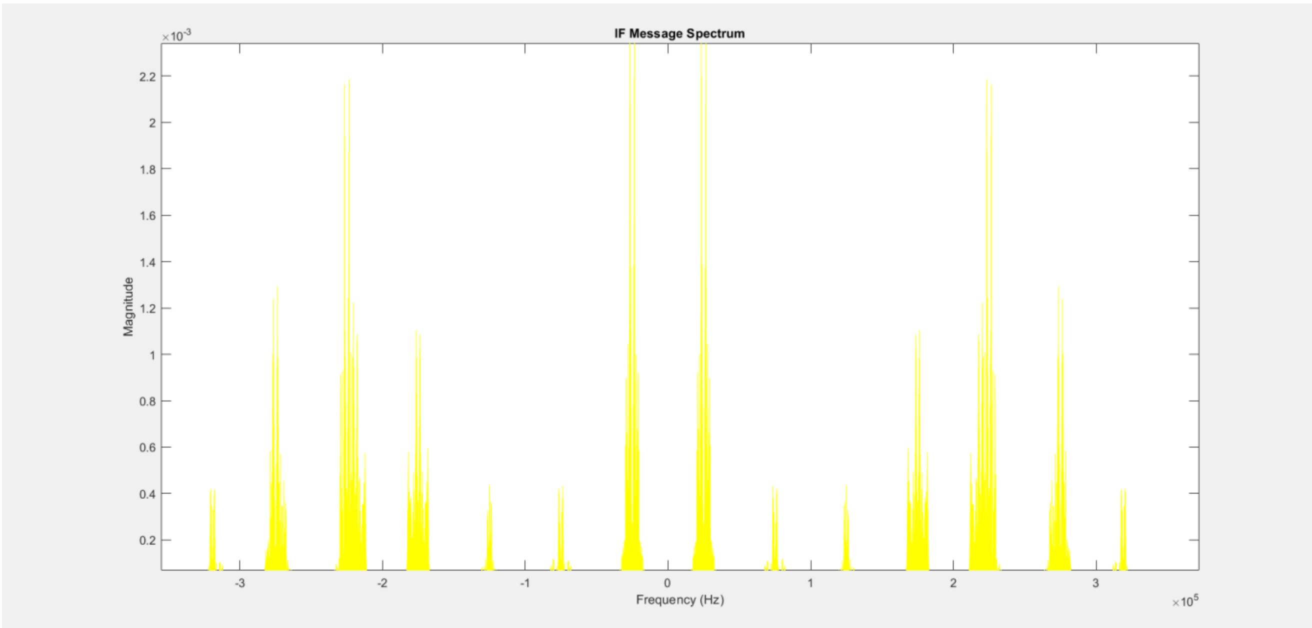


Figure 16

- **The Output Spectrum After The IF BPF without RF BPF (in FD) :**

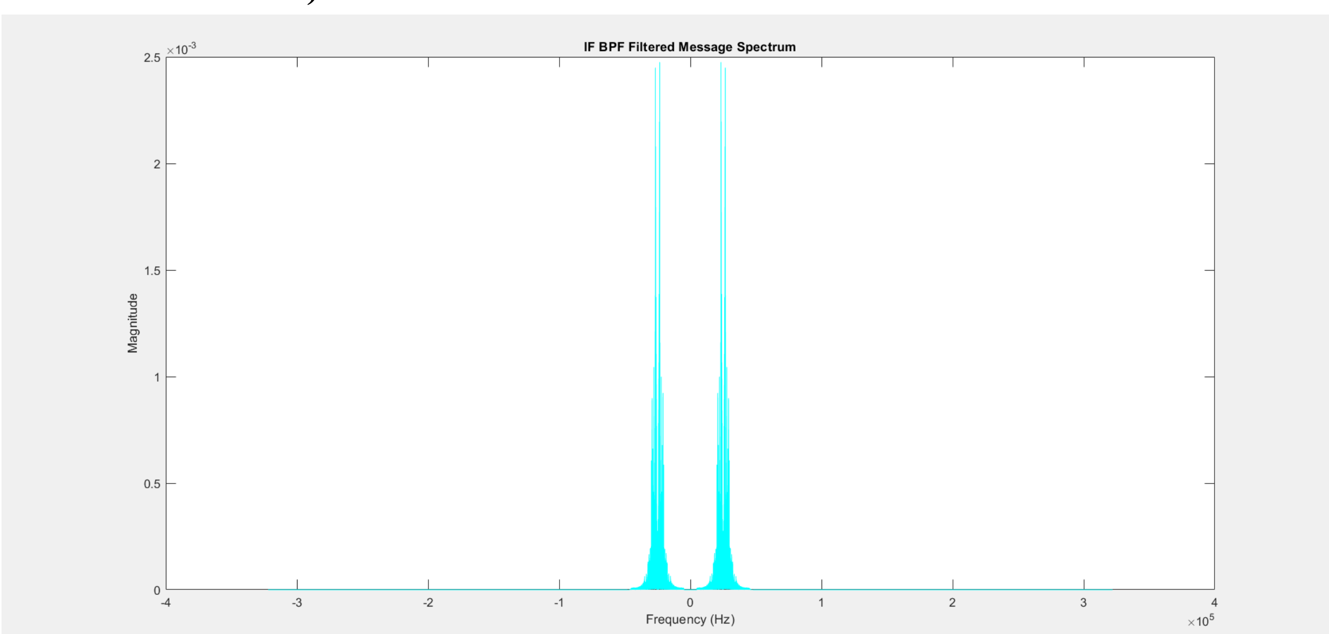


Figure 17

- **The Output Spectrum After The BB Mixer without RF BPF (in FD):**

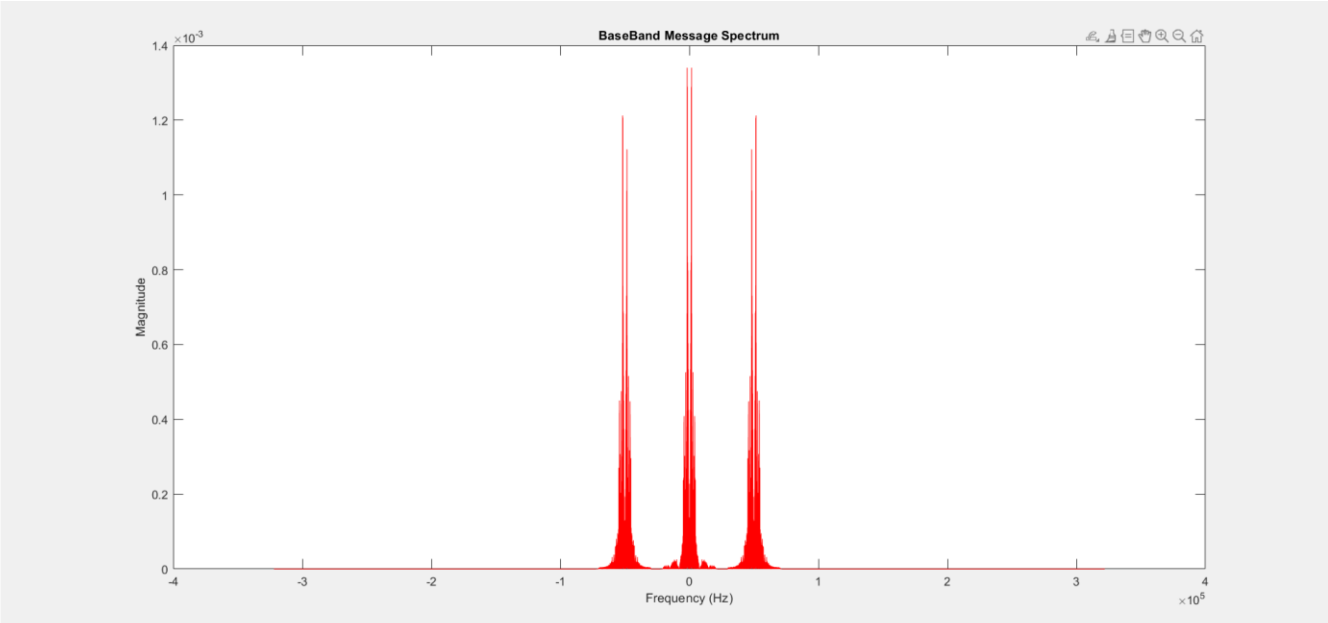


Figure 18

- **The Output Spectrum After The BB LPF without RF BPF (in FD) :**

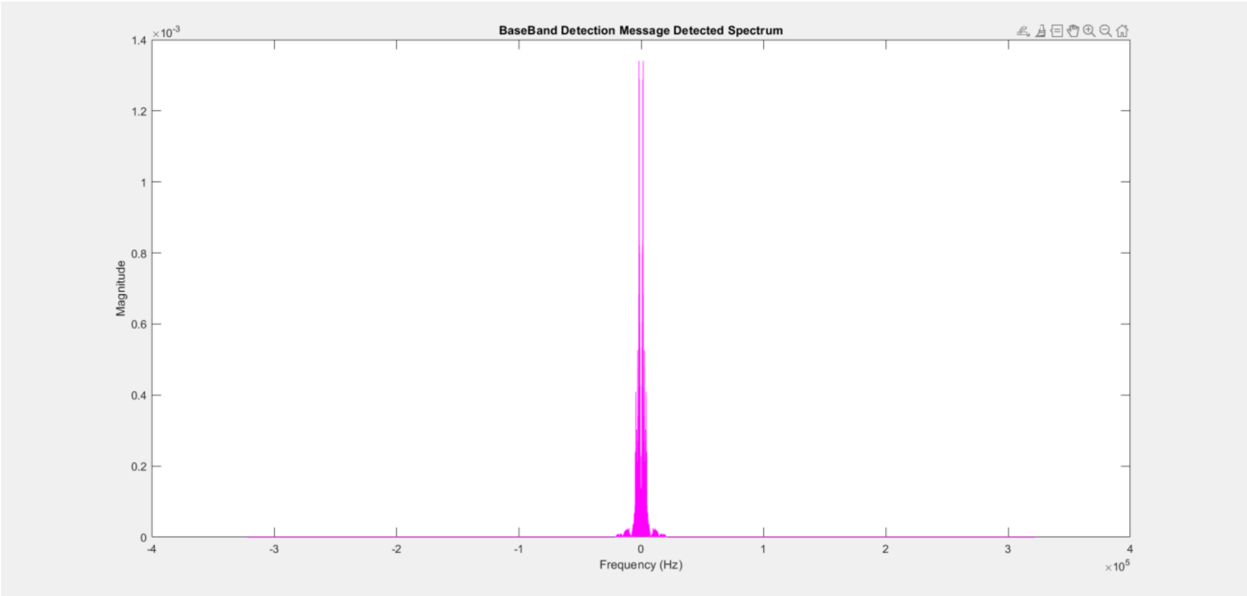


Figure 19

➔ 0.2KHz & 1.2KHz Frequency Shift in RF Mixer:

- After Frequency Shift of **0.2KHz & 1.2KHz** , The Sound is distorted by the interference and not clear and sounds like an Alien voice !!
- The Spectrum in Shifted slightly by **0.2KHz & 1.2KHz**:

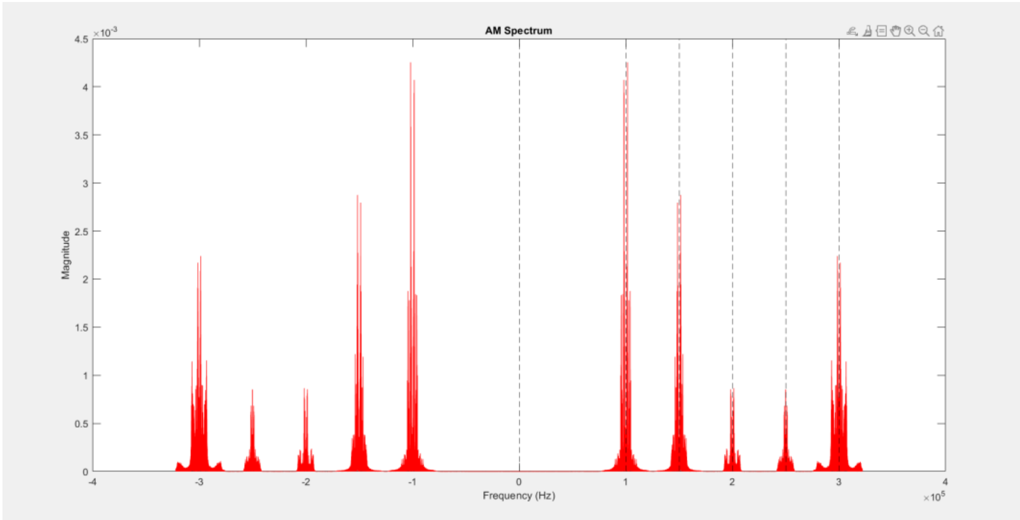


Figure 20 - 0.2KHz Frequency Shift

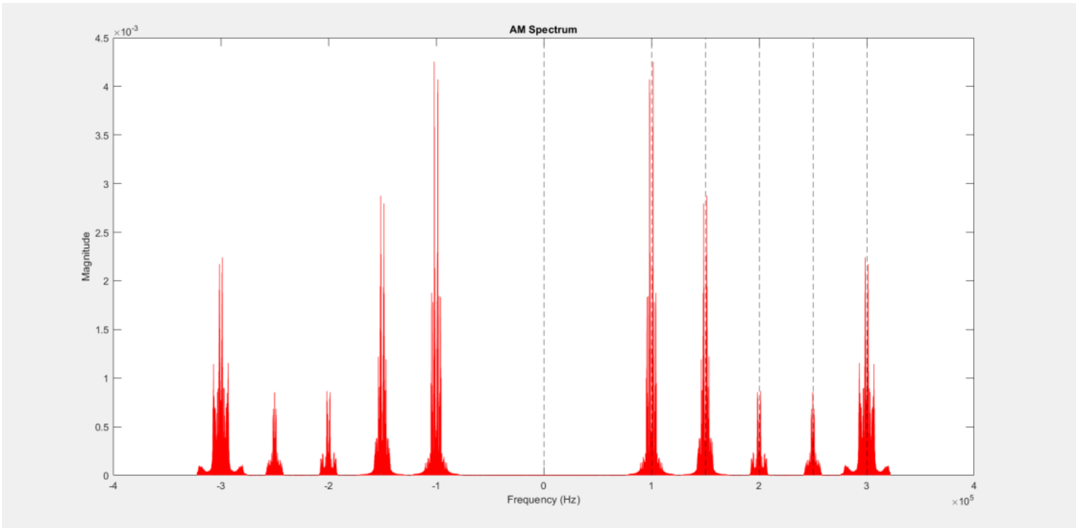


Figure 21 - 1.2KHz Frequency Shift

➔MATLAB Code:

```
%-----------------------------Reading Messages-----------------------------

%Reading Short_QuranPalestine.wav
[QuranPalestine,Fs_QPLS] = audioread('Short_QuranPalestine.wav');
%Converting stereo message to mono channel signal
QuranPalestine=mean(QuranPalestine,2);
%sound(Quraan_PLS,Fs_QPLS);


[FM9090,Fs_FM9090] = audioread('Short_FM9090.wav');
FM9090=mean(FM9090,2);
%sound(FM9090,Fs_FM9090);


[BBCArabic2,Fs_BBCArabic2] = audioread('Short_BBCArabic2.wav');
BBCArabic2=mean(BBCArabic2,2);
%sound(BBCArabic2,Fs_BBCArabic2);


[SkyNewsArabia,Fs_SkyNewsArabia] = audioread('Short_SkyNewsArabia.wav');
SkyNewsArabia=mean(SkyNewsArabia,2);
%sound(SkyNewsArabia,Fs_SkyNewsArabia);

[RussianVoice,Fs_RussianVoice] = audioread('Short_RussianVoice.wav');
RussianVoice=mean(RussianVoice,2);
%sound(RussianVoice,Fs_RussianVoice);

%----------------------------------------------------------------------------

%Defining The Sampling Frequency of the messages got from the audioread() function
Fs=44100;

%-----------------------------Padding with Zeroes----------------------------

 Max_Length = max([length(QuranPalestine), length(FM9090), length(BBCArabic2),
length(SkyNewsArabia), length(RussianVoice)]);
 QuranPalestine = [QuranPalestine; zeros(Max_Length - length(QuranPalestine), 1)];
 FM9090 = [FM9090; zeros(Max_Length - length(FM9090), 1)];
 BBCArabic2 = [BBCArabic2; zeros(Max_Length - length(BBCArabic2), 1)];
 SkyNewsArabia = [SkyNewsArabia; zeros(Max_Length - length(SkyNewsArabia), 1)];
 RussianVoice = [RussianVoice; zeros(Max_Length - length(RussianVoice), 1)];

%---------------------------------------------------------------------------
```

```
%------------------------------- QuranPalestine in TD---------------------------------

%Defining the time axis of the QuranPalestine Message
t_QPLS = (0:length(QuranPalestine)-1)/Fs_QPLS;

%-----------------------------------------------------------------------------------


%------------------------------- QuranPalestine in FD---------------------------------


%Fourier Transform to QuranPalestine
QPLS_FD = fft(QuranPalestine);
%Shift the DC component to the center of the Plot (The Zero Frequency Component)
QPLS_FD = fftshift(QPLS_FD);

%Normalizing the x-axis to be within [-Fs/2 , Fs/2] around origin
NS_FM9090=length(QuranPalestine);
QPLS_freq = (-NS_FM9090/2:NS_FM9090/2-1)* (Fs_QPLS/NS_FM9090);

%Normalizing the y-axis to indicate the power amplitude at each frequency sample
QPLS_Amp = abs(QPLS_FD/length(QPLS_FD));


%-----------------------------------------------------------------------------------


%------------------------------- FM9090 in TD----------------------------------------

t_FM9090 = (0:length(FM9090)-1)/Fs_FM9090;

%-----------------------------------------------------------------------------------


%------------------------------- FM9090 in FD----------------------------------------
FM9090_FD = fft(FM9090);
FM9090_FD = fftshift(FM9090_FD);
NS_FM9090=length(FM9090);
FM9090_freq = (-NS_FM9090/2:NS_FM9090/2-1)* (Fs_FM9090/NS_FM9090);
FM9090_Amp = abs(FM9090_FD/length(FM9090_FD));


%-----------------------------------------------------------------------------------



%------------------------------- BBCArabic2 in TD-------------------------------------
---

t_BBCArabic2 = (0:length(BBCArabic2)-1)/Fs_BBCArabic2;


%-----------------------------------------------------------------------------------
```

```
%------------------------------- BBCArabic2 in FD-------------------------------------
---
BBCArabic2_FD = fft(BBCArabic2);
BBCArabic2_FD = fftshift(BBCArabic2_FD);
NS_BBCArabic2=length(BBCArabic2);
BBCArabic2_freq = (-NS_BBCArabic2/2:NS_BBCArabic2/2-1)* (Fs_BBCArabic2/NS_BBCArabic2);
BBCArabic2_Amp = abs(BBCArabic2_FD/length(BBCArabic2_FD));


%-----------------------------------------------------------------------------------



%------------------------------- SkyNewsArabia in TD----------------------------------

t_SkyNewsArabia = (0:length(SkyNewsArabia)-1)/Fs_SkyNewsArabia;


%-----------------------------------------------------------------------------------


%------------------------------- SkyNewsArabia in FD---------------
SkyNewsArabia_FD = fft(SkyNewsArabia);
SkyNewsArabia_FD = fftshift(SkyNewsArabia_FD);
NS_SkyNewsArabia=length(SkyNewsArabia);
SkyNewsArabia_freq = (-NS_SkyNewsArabia/2:NS_SkyNewsArabia/2-1)*
(Fs_SkyNewsArabia/NS_SkyNewsArabia);
SkyNewsArabia_Amp = abs(SkyNewsArabia_FD/length(SkyNewsArabia_FD));


%--------------------------------------------------------------------------



%------------------------------- RussianVoice in TD----------------

t_RussianVoice = (0:length(RussianVoice)-1)/Fs_RussianVoice;

%--------------------------------------------------------------------------


%------------------------------- RussianVoice in FD----------------
RussianVoice_FD = fft(RussianVoice);
RussianVoice_FD = fftshift(RussianVoice_FD);
NS_RussianVoice=length(RussianVoice);
RussianVoice_freq = (-NS_RussianVoice/2:NS_RussianVoice/2-1)*
(Fs_RussianVoice/NS_RussianVoice);
RussianVoice_Amp = abs(RussianVoice_FD/length(RussianVoice_FD));

%--------------------------------------------------------------------------
```

```
%-----------------------------Plotting the Messages in Time Domain---------
figure;
subplot(3,2,1); plot(t_QPLS, QuranPalestine, 'b'); title('QuranPalestine (TD)');
xlabel('Time (s)');
ylabel('Magnitude');

subplot(3,2,2); plot(t_FM9090, FM9090, 'b'); title('FM9090 (TD)');
xlabel('Time (s)');
ylabel('Magnitude');

subplot(3,2,3); plot(t_BBCArabic2, BBCArabic2, 'b'); title('BBCArabic2 (TD)');
xlabel('Time (s)');
ylabel('Magnitude');

subplot(3,2,4); plot(t_SkyNewsArabia, SkyNewsArabia, 'b'); title('SkyNewsArabia
(TD)');
xlabel('Time (s)');
ylabel('Magnitude');


subplot(3,2,5); plot(t_RussianVoice, RussianVoice, 'b'); title('RussianVoice (TD)');
xlabel('Time (s)');
ylabel('Magnitude');
%--------------------------------------------------------------------------

%-----------------------------Plotting the Messages in Frequency Domain----
figure;
subplot(3,2,1); plot(QPLS_freq, QPLS_Amp, 'r'); title('QuranPalestine (FD)');
xlabel('Frequency (Hz)');
ylabel('Magnitude');

subplot(3,2,2); plot(FM9090_freq, FM9090_Amp, 'r'); title('FM9090 (FD)');
xlabel('Frequency (Hz)');
ylabel('Magnitude');


subplot(3,2,3); plot(BBCArabic2_freq, BBCArabic2_Amp, 'r'); title('BBCArabic2 (FD)');
xlabel('Frequency (Hz)');
ylabel('Magnitude');


subplot(3,2,4); plot(SkyNewsArabia_freq, SkyNewsArabia_Amp, 'r'); title('SkyNewsArabia
(FD)');
xlabel('Frequency (Hz)');
ylabel('Magnitude');


subplot(3,2,5); plot(RussianVoice_freq, RussianVoice_Amp, 'r'); title('RussianVoice
(FD)');
xlabel('Frequency (Hz)');
ylabel('Magnitude');

%--------------------------------------------------------------------------
```

```
%-------------------------Resampling----------------------------------
% T prevent Aliasing , Resampling the Messages to the Nyquist rate

Nyq_Fs=644e3; % more than 2 * 332KHz (Max Freq in the FD Spectrum)
QuranPalestine = resample(QuranPalestine, Nyq_Fs, Fs);
FM9090 = resample(FM9090, Nyq_Fs, Fs);
BBCArabic2 = resample(BBCArabic2, Nyq_Fs, Fs);
SkyNewsArabia = resample(SkyNewsArabia, Nyq_Fs, Fs);
RussianVoice = resample(RussianVoice, Nyq_Fs, Fs);

%--------------------------------------------------------------------------


%-----------------------------Padding with Zeroes After Resampling--------------------


 Max_Length = max([length(QuranPalestine), length(FM9090), length(BBCArabic2), 
length(SkyNewsArabia), length(RussianVoice)]);
 QuranPalestine = [QuranPalestine; zeros(Max_Length - length(QuranPalestine), 1)];
 FM9090 = [FM9090; zeros(Max_Length - length(FM9090), 1)];
 BBCArabic2 = [BBCArabic2; zeros(Max_Length - length(BBCArabic2), 1)];
 SkyNewsArabia = [SkyNewsArabia; zeros(Max_Length - length(SkyNewsArabia), 1)];
 RussianVoice = [RussianVoice; zeros(Max_Length - length(RussianVoice), 1)];


%--------------------------------------------------------------------------


%---------------------------AM Modulation---------------------------
Ts = 1/Nyq_Fs;
%Defining the time axis for the carriers to be multiple of Ts
T = (0:Max_Length-1)' * Ts;
% Carrier signals as a cos() functions
Carrier_QPLS = cos(2 * pi * 100e3 * T);
Carrier_FM9090 = cos(2 * pi * 150e3 * T);
Carrier_BBCArabic2 = cos(2 * pi * 200e3 * T);
Carrier_SkyNewsArabia = cos(2 * pi * 250e3 * T);
Carrier_RussianVoice = cos(2 * pi * 300e3 * T);

% Amplitude Modulating Messages
AM_QPLS = QuranPalestine .* Carrier_QPLS;
AM_FM9090 = FM9090 .* Carrier_FM9090;
AM_BBCArabic2 = BBCArabic2 .* Carrier_BBCArabic2;
AM_SkyNewsArabia = SkyNewsArabia .* Carrier_SkyNewsArabia;
AM_RussianVoice = RussianVoice .* Carrier_RussianVoice;

% Adding all the Modulation Messages in The TD Channel
TD_Channel = AM_QPLS + AM_FM9090 + AM_BBCArabic2 + AM_SkyNewsArabia + AM_RussianVoice;

% Plotting The Channel in TD
plotTimeDomainAudio(TD_Channel, Nyq_Fs, "The Channel in TD");
```

```matlab
% Plotting The FDM Spectrum
figure;
FDM_Spectrum = fft(TD_Channel);
FDM_Spectrum = fftshift(FDM_Spectrum);


NS_TD_Channel=length(TD_Channel);
FDM_Spectrum_freq = (-NS_TD_Channel/2:NS_TD_Channel/2-1)* (Nyq_Fs/NS_TD_Channel);
FDM_Spectrum_Amp = abs(FDM_Spectrum/length(FDM_Spectrum));

plot(FDM_Spectrum_freq, FDM_Spectrum_Amp, 'r');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
title('AM Spectrum');
%------------------------------------------------------------------------



%-------------------------RF BPF---------------------------------

Carrier_Frequencies =[100e3, 150e3, 200e3, 250e3, 300e3];

%Defining array of the FPass frequencies for each message to be ± 20KHz
%around the center frequency of the modulated message
RF_FPass_Frequencies = [Carrier_Frequencies(1)-20e3, Carrier_Frequencies(1)+20e3;
                         Carrier_Frequencies(2)-20e3, Carrier_Frequencies(2)+20e3;
                         Carrier_Frequencies(3)-20e3, Carrier_Frequencies(3)+20e3;
                         Carrier_Frequencies(4)-20e3, Carrier_Frequencies(4)+20e3;
                         Carrier_Frequencies(5)-30e3, Carrier_Frequencies(5)+20e3];

% Definging indeces for each message to choose its FPass frequencies easily
% from the array to pass them to the Filter Design function
QPLS_indx=1;
FM9090_indx=2;
BBCArabic2_indx=3;
SkyNewsArabia_indx=4;
RussianVoice_indx=5;

%%%%%%%%%%%%%%%%%%%%%%%%%%%
Chosen_Message=QPLS_indx; % To Choose which Message to Recieve
%%%%%%%%%%%%%%%%%%%%%%%%%%%

RF_BPF_Fpass1 = RF_FPass_Frequencies(Chosen_Message,1);
RF_BPF_Fpass2 = RF_FPass_Frequencies(Chosen_Message,2);

RF_BPF_Fs = 660e3;
RF_BPF= designfilt('bandpassiir', ...
    'PassbandFrequency1', RF_BPF_Fpass1, ...
    'PassbandFrequency2', RF_BPF_Fpass2, ...
    'StopbandFrequency1', RF_BPF_Fpass1 - 10e3, ...
    'StopbandFrequency2', RF_BPF_Fpass2 + 10e3, ...
    'SampleRate', RF_BPF_Fs);

RF_Filtered = filter(RF_BPF, TD_Channel); % The RF Filtered Message in TD
```

```
% Plotting RF Filtered Message
plotTimeDomainAudio(RF_Filtered, Nyq_Fs, "RF Filtered Message in TD");


%Plotting the RF Filtered Message in FD
figure;
RF_Filtered_Spectrum = fft(RF_Filtered);
RF_Filtered_Spectrum = fftshift(RF_Filtered_Spectrum);


NS_RF_Filtered=length(RF_Filtered);
RF_Filtered_freq = (-NS_RF_Filtered/2:NS_RF_Filtered/2-1)* (RF_BPF_Fs/NS_RF_Filtered);
RF_Filtered_Spectrum_Amp = abs(RF_Filtered_Spectrum/length(RF_Filtered_Spectrum));

plot(RF_Filtered_freq, RF_Filtered_Spectrum_Amp,'k');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
title('RF BPF Filtered Message Spectrum');

%-------------------------------------------------------------------------


%------------------------------IF Mixer-----------------------------------

W_IF=25e3;
IF_FS=644e3;

Carrier_Frequencies =[100e3, 150e3, 200e3, 250e3, 300e3];

LO_Osc = cos(2 * pi * (Carrier_Frequencies(Chosen_Message) + W_IF) * T);

IF_Message = RF_Filtered .* LO_Osc;


% Plotting The IF Message in TD
plotTimeDomainAudio(IF_Message, Nyq_Fs, "IF Message in TD");


%Plotting the IF Message in FD
figure;
IF_Spectrum = fft(IF_Message);
IF_Spectrum = fftshift(IF_Spectrum);


NS_IF_Message=length(IF_Message);
IF_Message_freq = (-NS_IF_Message/2:NS_IF_Message/2-1)* (IF_FS/NS_IF_Message);
IF_Message_Spectrum_Amp = abs(IF_Spectrum/length(IF_Spectrum));

plot(IF_Message_freq, IF_Message_Spectrum_Amp,'y');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
title('IF Message Spectrum');
%-------------------------------------------------------------------------
```

```matlab
%-----------------------------IF Filter---------------------------------


IF_BPF_Fpass1 = W_IF - 20e3;
IF_BPF_Fpass2 = W_IF + 20e3;
IF_BPF_Fs = 644e3;


IF_BPF= designfilt('bandpassiir', ...
    'PassbandFrequency1', IF_BPF_Fpass1, ...
    'PassbandFrequency2', IF_BPF_Fpass2, ...
    'StopbandFrequency1', IF_BPF_Fpass1 - 4e3, ...
    'StopbandFrequency2', IF_BPF_Fpass2 + 4e3, ...
    'SampleRate', IF_BPF_Fs);



IF_Filtered = filter(IF_BPF, IF_Message); % The IF Message in TD


% Plotting The IF Filtered Message in TD
plotTimeDomainAudio(IF_Filtered, Nyq_Fs, "IF Filtered Message in TD");



%Plotting the IF Filtered Message in FD
figure;
IF_Filtered_Spectrum = fft(IF_Filtered);
IF_Filtered_Spectrum = fftshift(IF_Filtered_Spectrum);


NS_IF_Filtered=length(IF_Filtered);
IF_Filtered_freq = (-NS_IF_Filtered/2:NS_IF_Filtered/2-1)* (IF_BPF_Fs/NS_IF_Filtered);
IF_Filtered_Spectrum_Amp = abs(IF_Filtered_Spectrum/length(IF_Filtered_Spectrum));

plot(IF_Filtered_freq, IF_Filtered_Spectrum_Amp,'c');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
title('IF BPF Filtered Message Spectrum');
%--------------------------------------------------------------------------
```

```matlab
%------------------------------  BaseBand Mixer ------------------------------
W_IF=25e3;
BB_FS=644e3;

% Demodulate the IF Message back to the BaseBand by shifting it in FD by W_IF
IF_Osc = cos(2 * pi * W_IF * T);
BB_Message = IF_Filtered .* IF_Osc;



% Plotting The BB Message in TD
plotTimeDomainAudio(BB_Message, Nyq_Fs, "BaseBand Message in TD");



%Plotting the BaseBand Message in FD
figure;
BB_Message_Spectrum = fft(BB_Message);
BB_Message_Spectrum = fftshift(BB_Message_Spectrum);


NS_BB_Message=length(BB_Message);
BB_Message_freq = (-NS_BB_Message/2:NS_BB_Message/2-1)* (BB_FS/NS_BB_Message);
BB_Message_Spectrum_Amp = abs(BB_Message_Spectrum/length(BB_Message_Spectrum));

plot(BB_Message_freq, BB_Message_Spectrum_Amp,'r');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
title('BaseBand Message Spectrum');
%-------------------------------------------------------------------------


%-------------------------------BaseBand Detection-------------------------

BB_LPF_Fs = 644e3;
%The Max BB BW in the Messages is about 24KHz (for RussianVoice message)
LPF_CutOff_Freq = 22e3;

BB_LPF = designfilt('lowpassiir', ...
    'PassbandFrequency', LPF_CutOff_Freq, ...
    'StopbandFrequency', LPF_CutOff_Freq + 5e3, ...
    'SampleRate', BB_LPF_Fs);

BB_Msg_Filtered = filter(BB_LPF, BB_Message);


% Plotting The BB Filtered Message in TD
plotTimeDomainAudio(BB_Msg_Filtered, Nyq_Fs, "BaseBand Filtered Message in TD");
```

```
%Plotting the BaseBand Filtered Message in FD
figure;
BB_Msg_Filtered_Spectrum = fft(BB_Msg_Filtered);
BB_Msg_Filtered_Spectrum = fftshift(BB_Msg_Filtered_Spectrum);


NS_BB_Msg_Filtered=length(BB_Msg_Filtered);
BB_Msg_Filtered_freq = (-NS_BB_Msg_Filtered/2:NS_BB_Msg_Filtered/2-1)*
(BB_LPF_Fs/NS_BB_Msg_Filtered);
BB_Msg_Filtered_Spectrum_Amp =
abs(BB_Msg_Filtered_Spectrum/length(BB_Msg_Filtered_Spectrum));

plot(BB_Msg_Filtered_freq, BB_Msg_Filtered_Spectrum_Amp,'m');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
title('BaseBand Detection Message Detected Spectrum');

%-------------------------------------------------------------------------



%-----------------------------Resampling BaseBand Message To 44.1KHz ------

%Resampling the Recieved Message back to 44.1KHz to listen it regularly
Nyq_Fs=644e3;
Fs=44100;
BB_Msg_DownSampled = 2 * resample(BB_Msg_Filtered, Fs, Nyq_Fs);


% Plotting The BB Filtered Message DownSampled in TD
plotTimeDomainAudio(BB_Msg_Filtered, Nyq_Fs, "BaseBand Filtered Message DownSampled in
TD");


%-------------------------------------------------------------------------
```

```
%--------------------------Adding Noise-----------------------------------

% Calculate Message Power to get exact signal power
Message_Power = mean(BB_Msg_DownSampled.^2);

% Defining desired Signal-to-Noise Ratio (SNR) in dB After Adding Noise
SNR_dB = 7;

% SNR in Linear Scale
SNR_Linear = 10^(SNR_dB/10);

% Add Noise to the Message by AWGN
Noisy_Message = awgn(BB_Msg_DownSampled, SNR_dB, 'measured');


% Plot the Original Message in TD
time = (0:length(BB_Msg_DownSampled)-1) / Fs;
figure;
subplot(2,1,1);
plot(time, BB_Msg_DownSampled);
title('Original Message in TD');
xlabel('Time (s)');
ylabel('Amplitude');

% Plot the Noisy Message in TD
subplot(2,1,2);
plot(time, Noisy_Message);
title('Noisy Message in TD');
xlabel('Time (s)');
ylabel('Amplitude');

%Plot the Original Message in FD
figure;
subplot(2,1,1);
Plot_FD_Spectrum(BB_Msg_DownSampled, Fs, 'Original Message in FD')

%Plot the Noisy Message in FD
subplot(2,1,2);
Plot_FD_Spectrum(Noisy_Message, Fs,  'Noisy Message in FD')

%Playing the Noisy Message
%sound(Noisy_Message, Fs);

%Saving the Noisy Message
%audiowrite('output QuranPalestine + Noise.wav',Noisy_Message, Fs);

%-------------------------------------------------------------------------
```

```
%------------------------------Reading Message Recieved--------------------
sound(BB_Msg_DownSampled, Fs);
%------------------------------------------------------------------------


%------------------------------Saving Message Recieved--------------------

%audiowrite('output QuranPalestine.wav',BB_Msg_Resampled_44KHz, Fs);
%------------------------------------------------------------------------




%----------------------------Time Domain Plotting-------------------------------
function plotTimeDomainAudio(Message, Nyq_Fs, plotTitle)
% Calculate time vector
t_TD = (0:length(Message)-1) / Nyq_Fs;

% Plot The Message in TD
figure;
plot(t_TD, Message, 'b');

% Set Plot titles and labels
title(plotTitle);
xlabel('Time (s)');
ylabel('Magnitude');
end
%------------------------------------------------------------------------



%----------------------------Frequency Domain Plotting-------------------------------

function Plot_FD_Spectrum(audio_signal, sampling_freq, plot_title)
    audio_spectrum = fft(audio_signal);
    audio_spectrum = fftshift(audio_spectrum);
    N_FD = length(audio_signal);
    freq_axis = (-N_FD/2:N_FD/2-1) * (sampling_freq/N_FD);
    audio_spectrum_amp = abs(audio_spectrum)/N_FD;
    plot(freq_axis, audio_spectrum_amp, 'r');
    xlabel('Frequency (Hz)');
    ylabel('Magnitude');
    title(plot_title);
end
%------------------------------------------------------------------------
```