Cairo University

Faculty of Engineering

Electronics and Electrical Communications Engineering Department

Third Year

Analog Communications

Term Project

MATLAB implementation of a superheterodyne receiver

Contents

Table of figures

1. ***The transmitter***

This part contains the following tasks

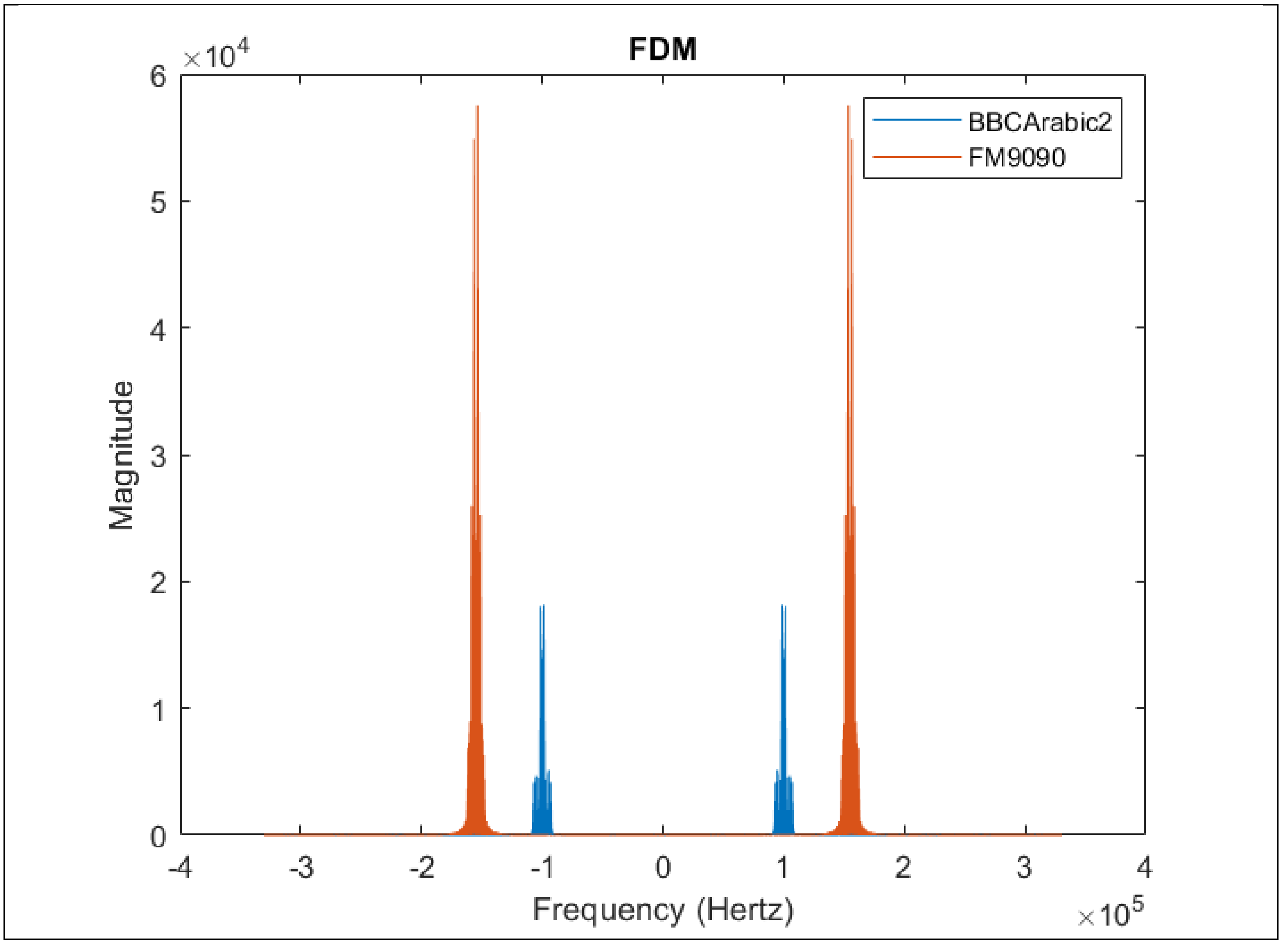
1. Reading monophonic audio signals into MATLAB.
2. Up-sampling the audio signals.
3. Modulating the audio signals (each on a separate carrier).
4. Addition of the modulated signals.

Discussion

Read audios for processing(adding 2 columns to 1), Then up-sample them to prevent information loss with interpolated factor of 15, Modulate each signal on separate carrier for transmission and finally, Add the modulated signals to form a composite signal for efficient communication system operation.

The figures

Figure 1: The spectrum of the output of the transmitter



2. ***The RF stage***

This part addresses the RF filter and the mixer following it.

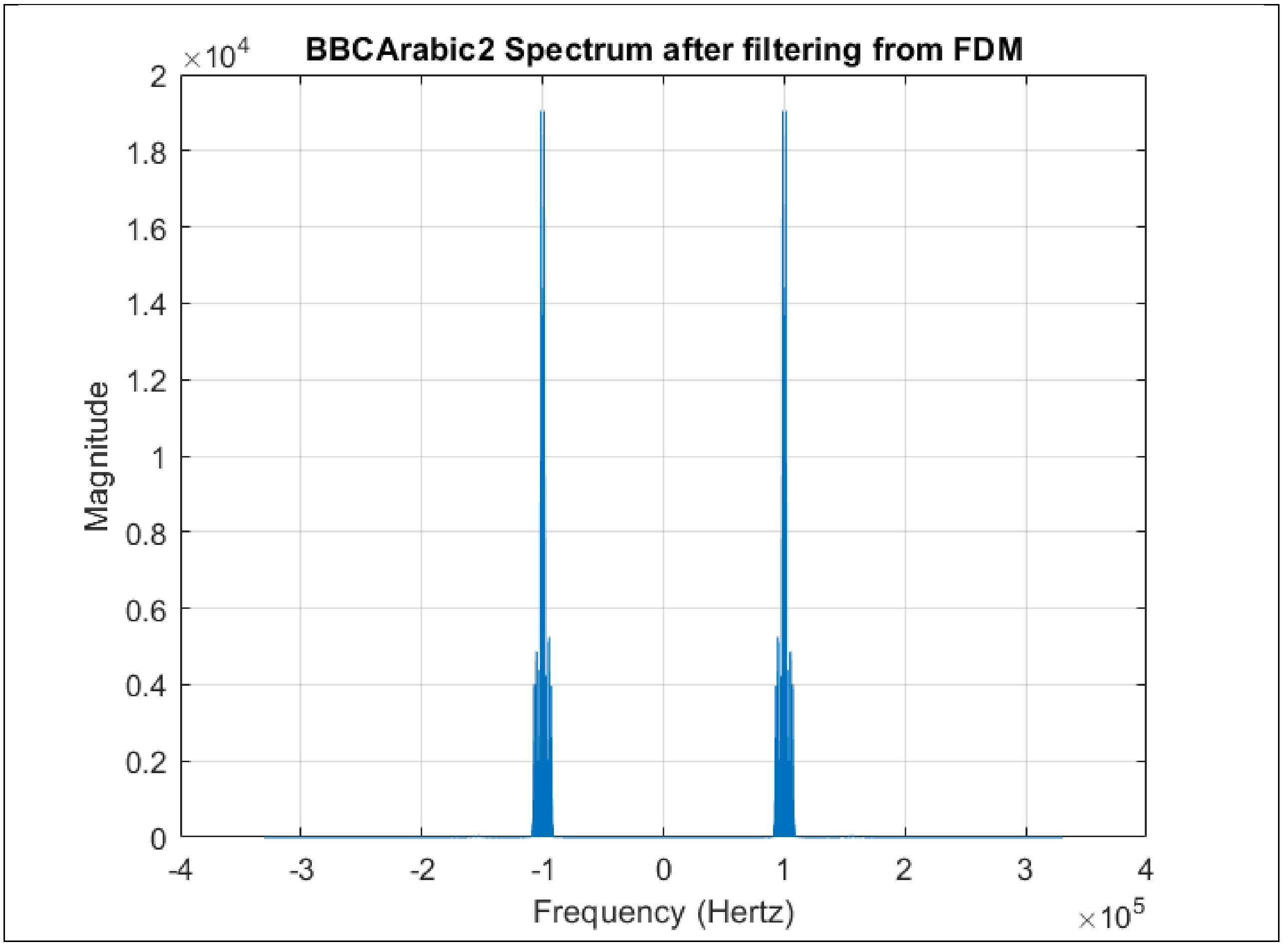
Discussion

The RF stage, comprising the RF filter and subsequent mixer, is important for refining signals in communication systems, The RF filter selectively permits desired frequency bands, enhancing signal quality by minimizing interference. The mixer then up-converts signals, optimizing their alignment with transmission requirements for efficient and extended-range communication.

The figures

Assume we want to demodulate the first signal (at ω_o).

Figure 2: the output of the RF filter (before the mixer)



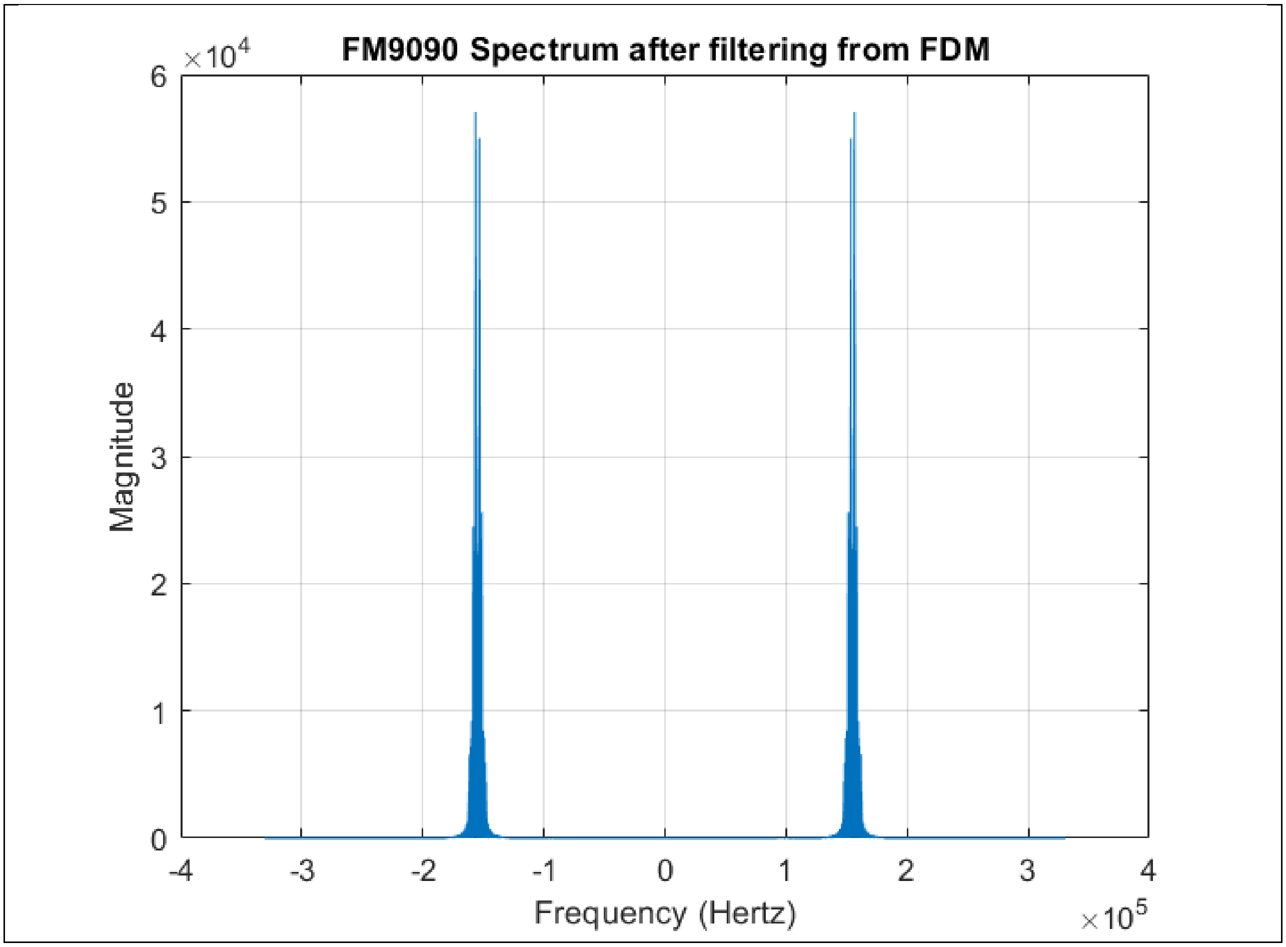
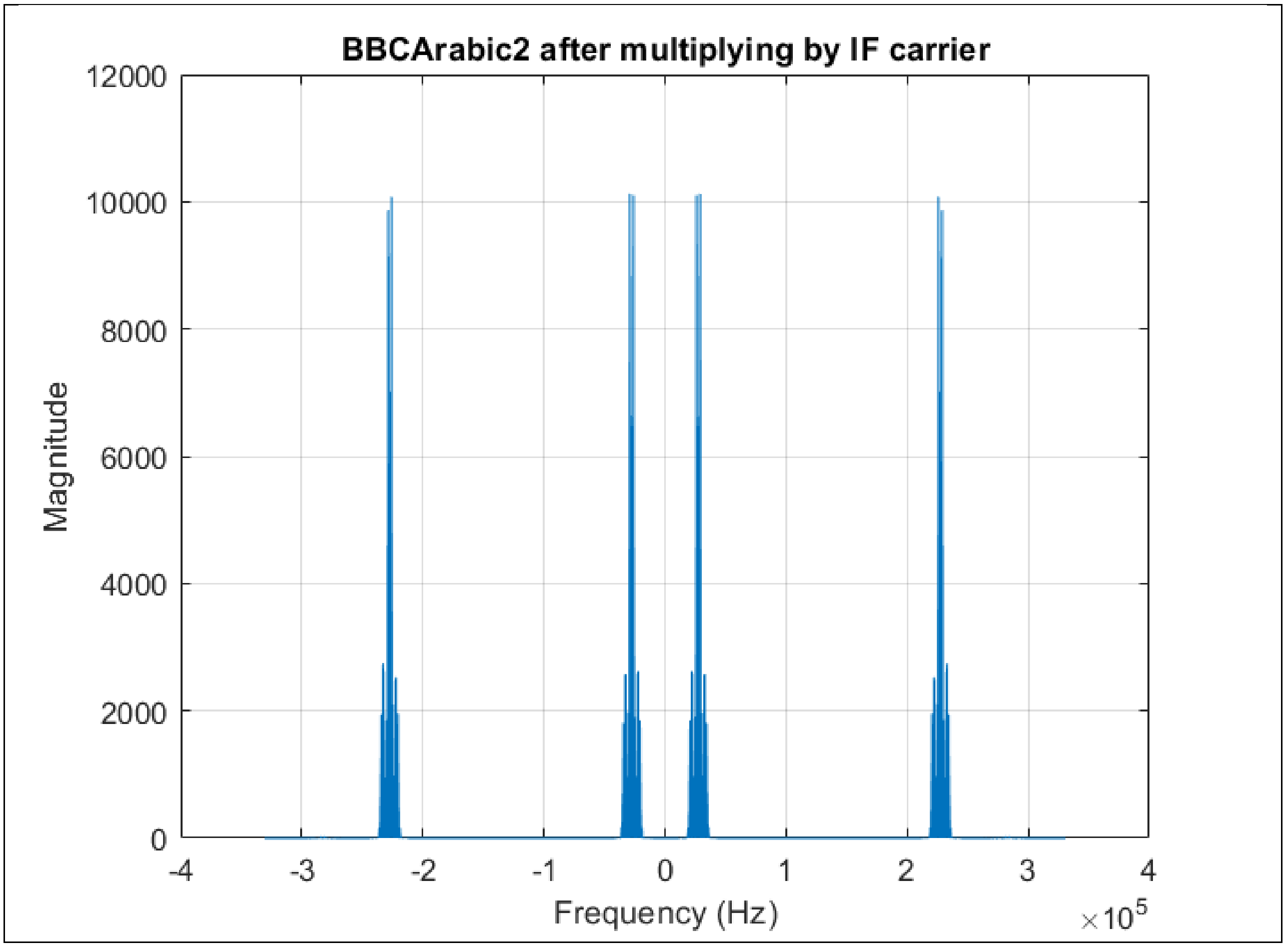
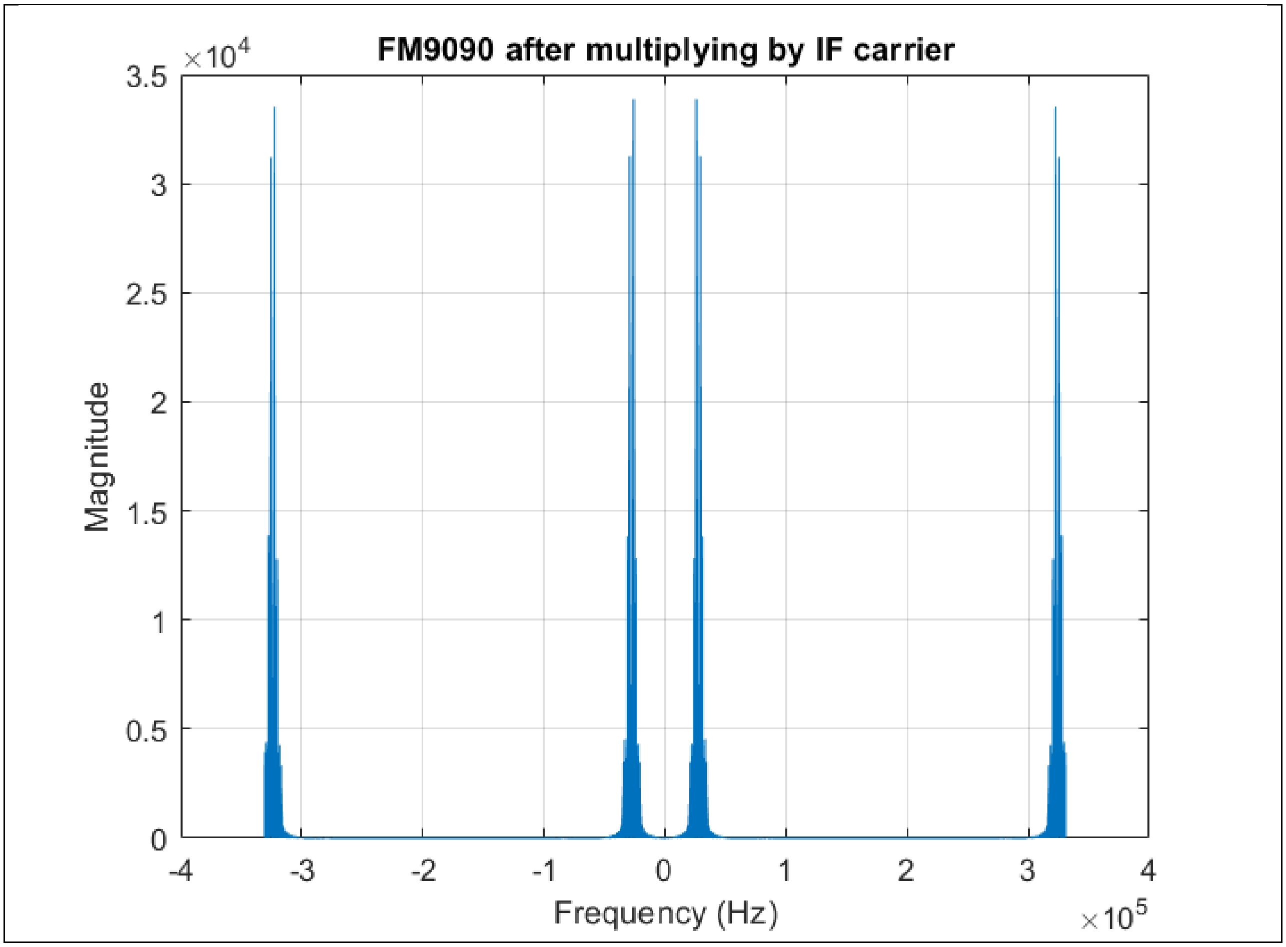


Figure 3: The output of the mixer


BBCArabic2 after multiplying by IF carrier
Magnitude
Frequency (Hz)
$\times 10^5$
0
2000
4000
6000
8000
10000
12000
-4
-3
-2
-1
0
1
2
3
4



3. *The IF stage*

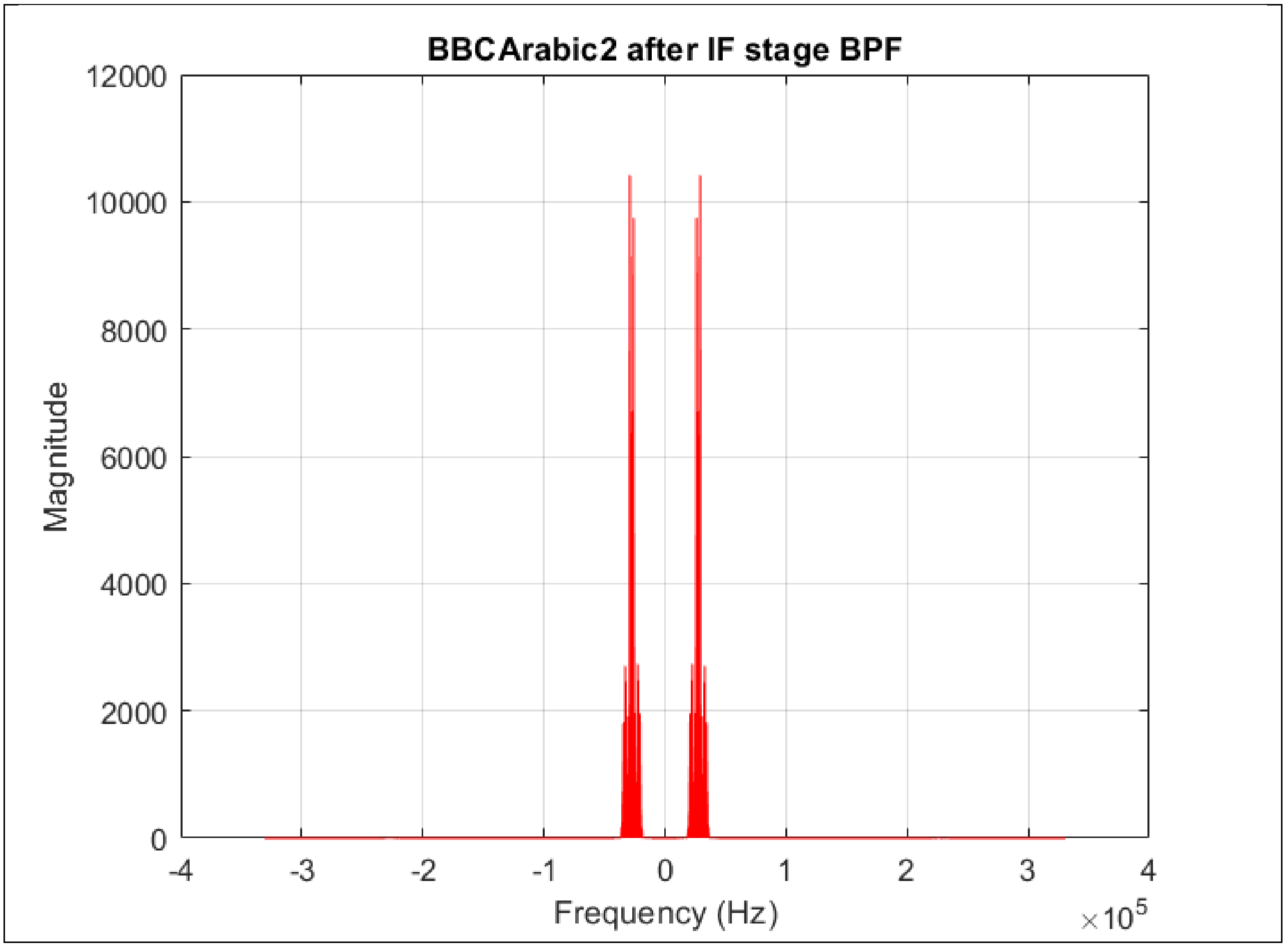
This part addresses the IF filter.

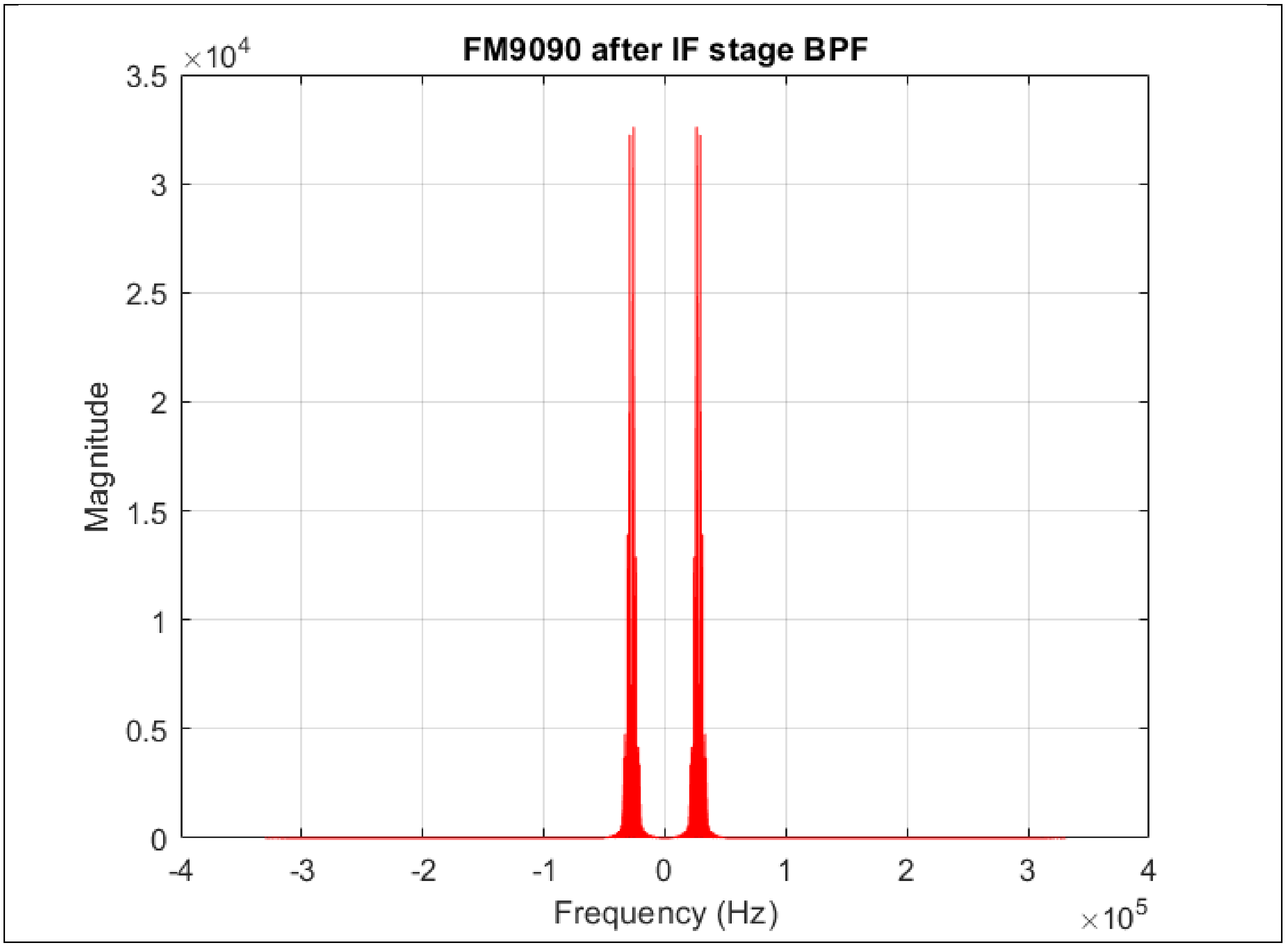
Discussion

The IF filter isolate and enhance specific frequency components, reducing interference and noise picked up during transmission. By focusing on an intermediate frequency range, the IF filter ensures the signals are well-prepared for subsequent demodulation. This stage is essential for maintaining signal clarity, contributing to the overall efficiency and reliability of the communication system.

The figures

Figure 4: Output of the IF filter


BBCArabic2 after IF stage BPF
Magnitude
12000
10000
8000
6000
4000
2000
0
-4
-3
-2
-1
0
1
2
3
4
Frequency (Hz)
×10^5



4. ***The baseband demodulator***

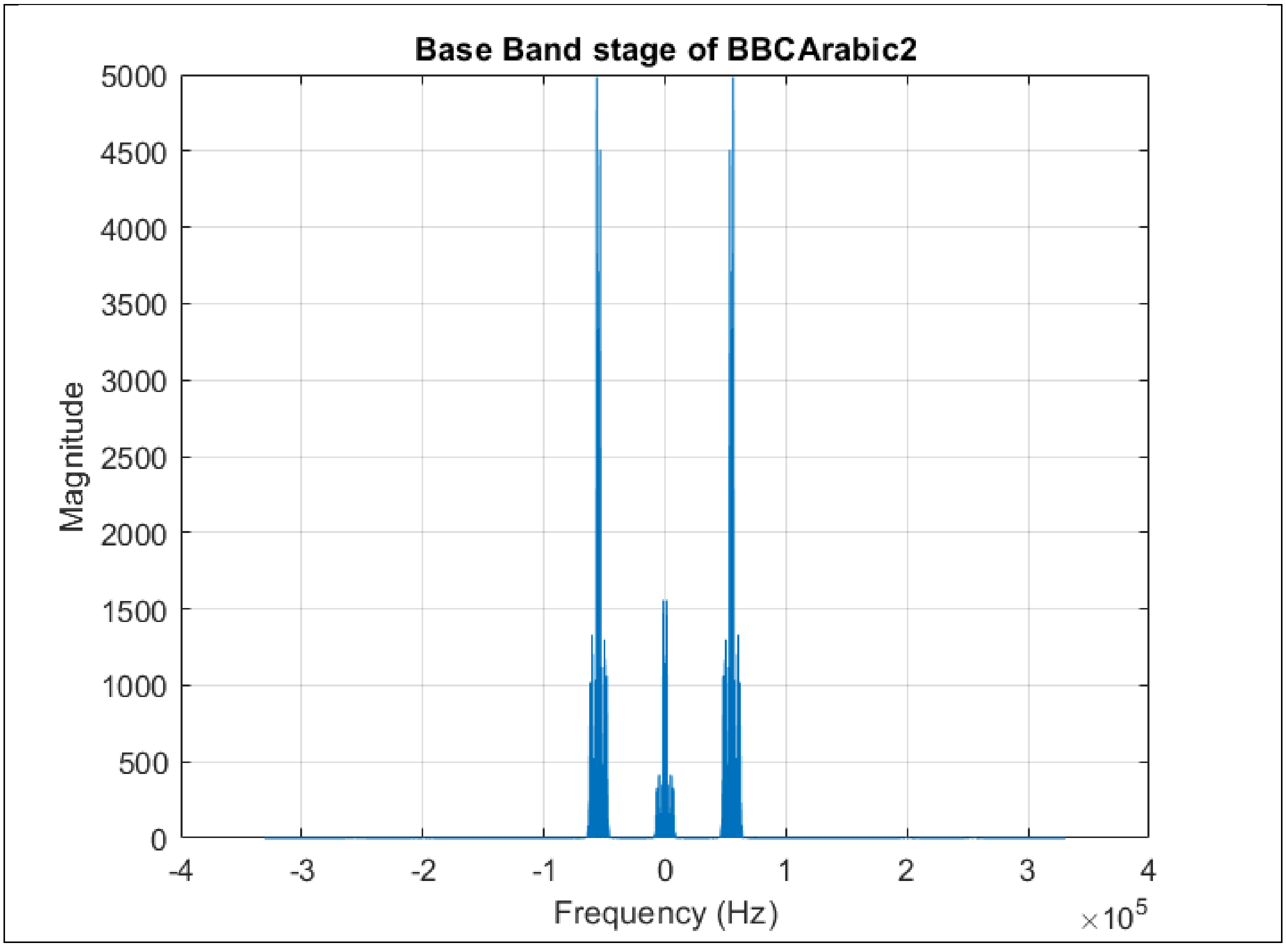
This part addresses the coherent detector used to demodulate the signal from the IF stage.

Discussion

Its key role involves extracting the original baseband signal, a critical step in reversing the earlier applied modulation. The coherent detector ensures accurate retrieval of phase and frequency information, enabling an accurate reproduction of the transmitted data. This stage's precision is important for accurate processing of sound or data, completing reception in communication system.

The figures

Figure 5: Output of the mixer (before the LPF)



Base Band stage of BBCArabic2
Magnitude
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
-4
-3
-2
-1
0
1
2
3
4
Frequency (Hz)
$\times 10^5$

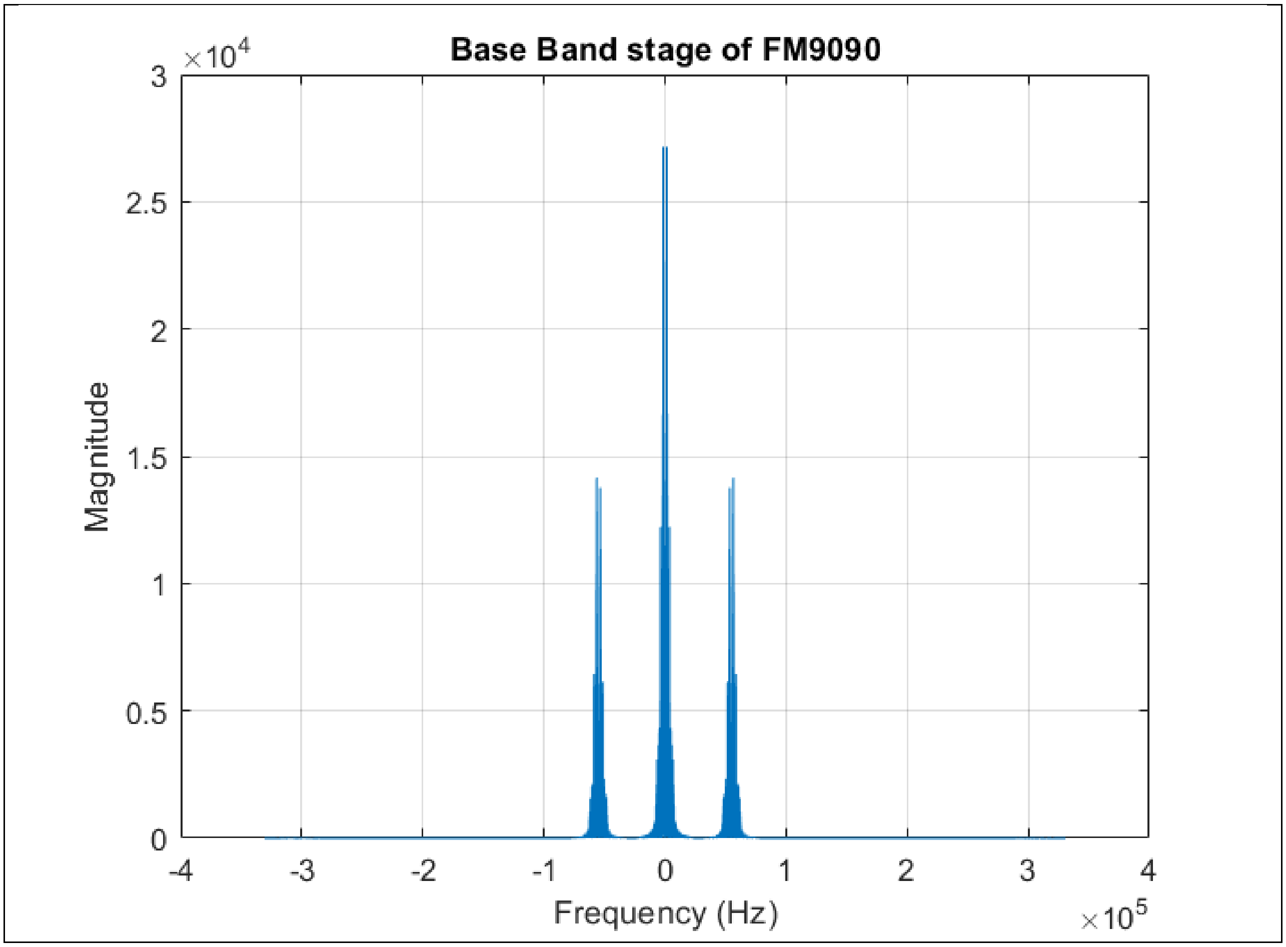
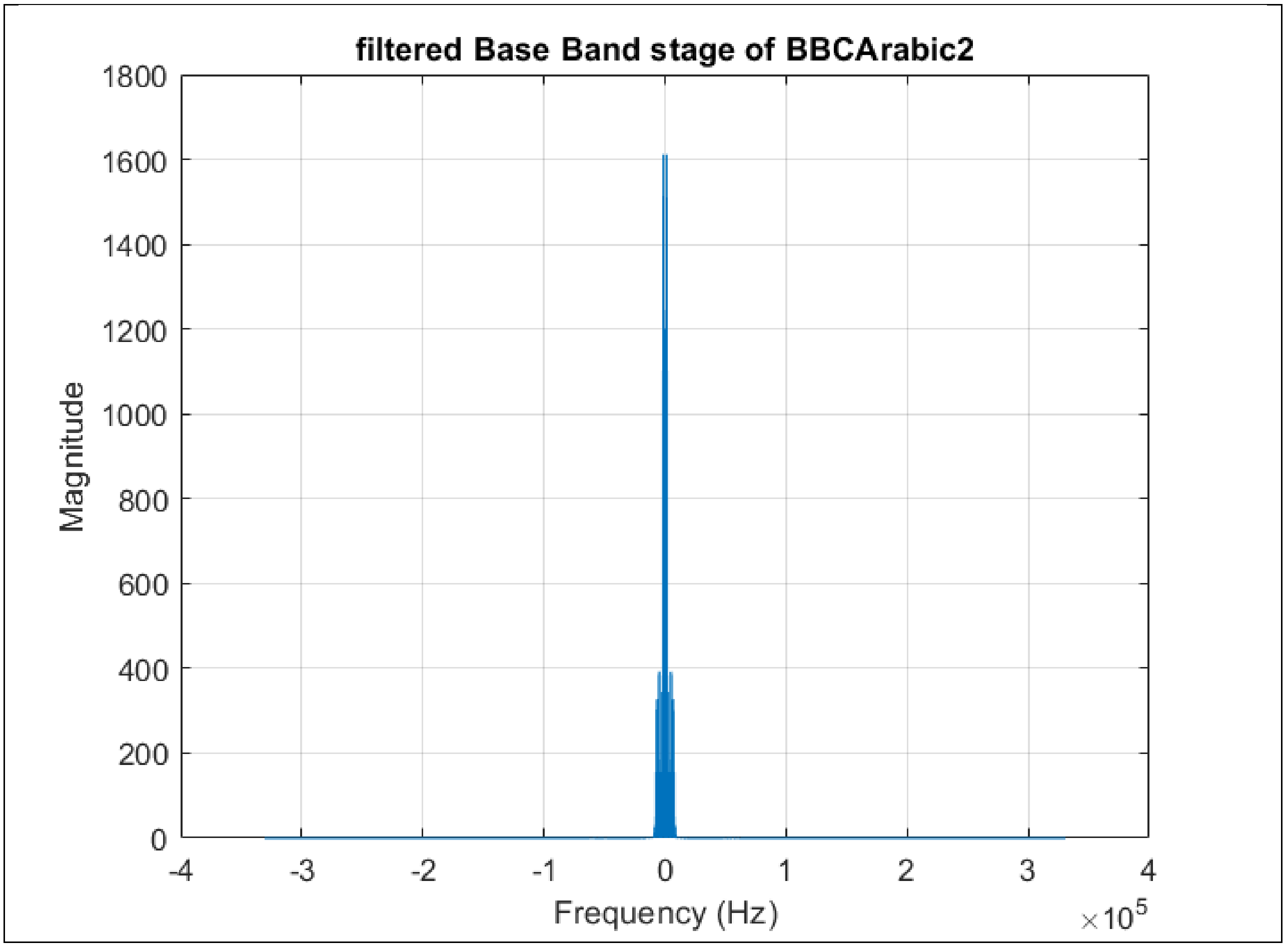
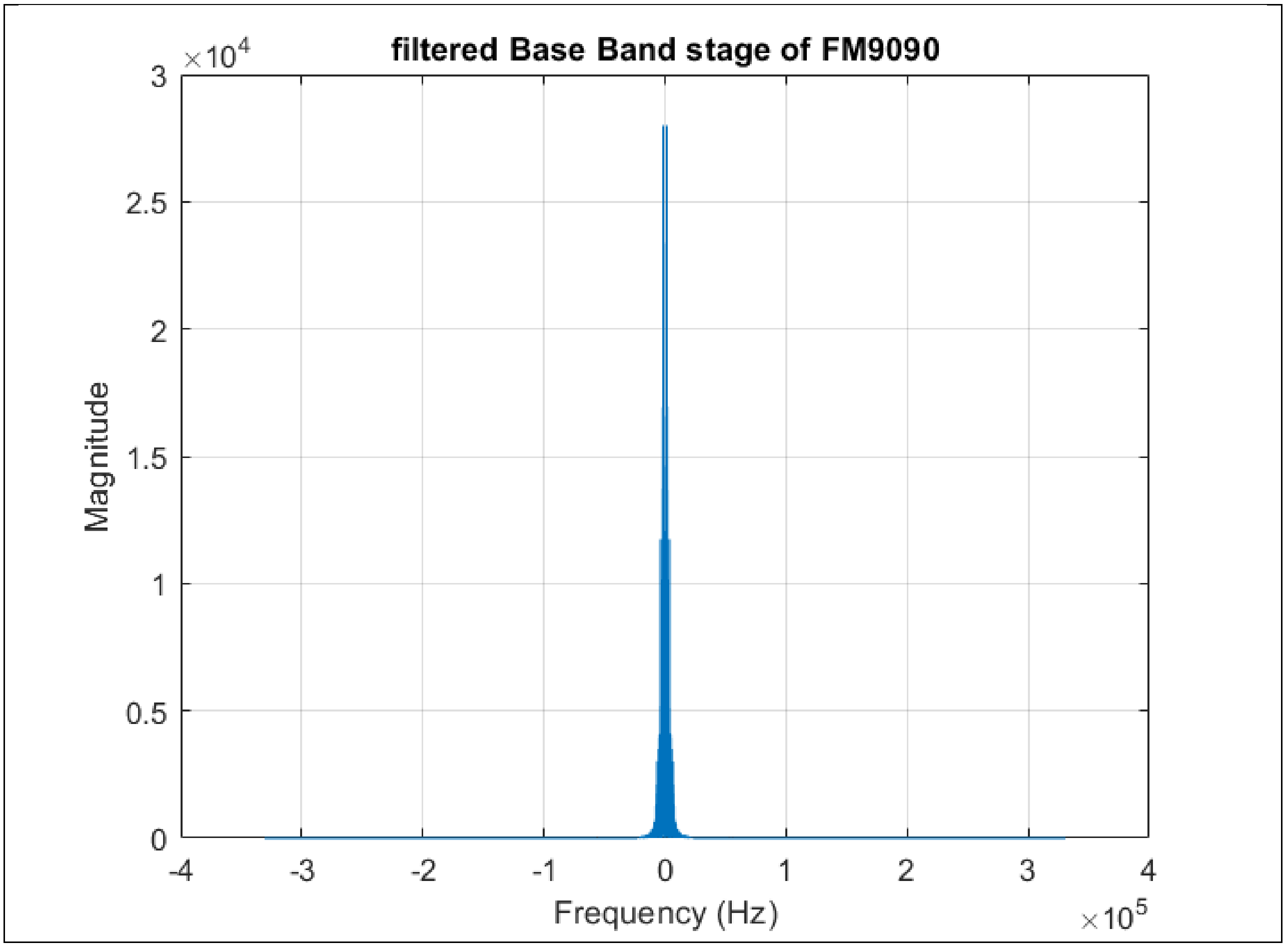


Figure 6: Output of the LPF



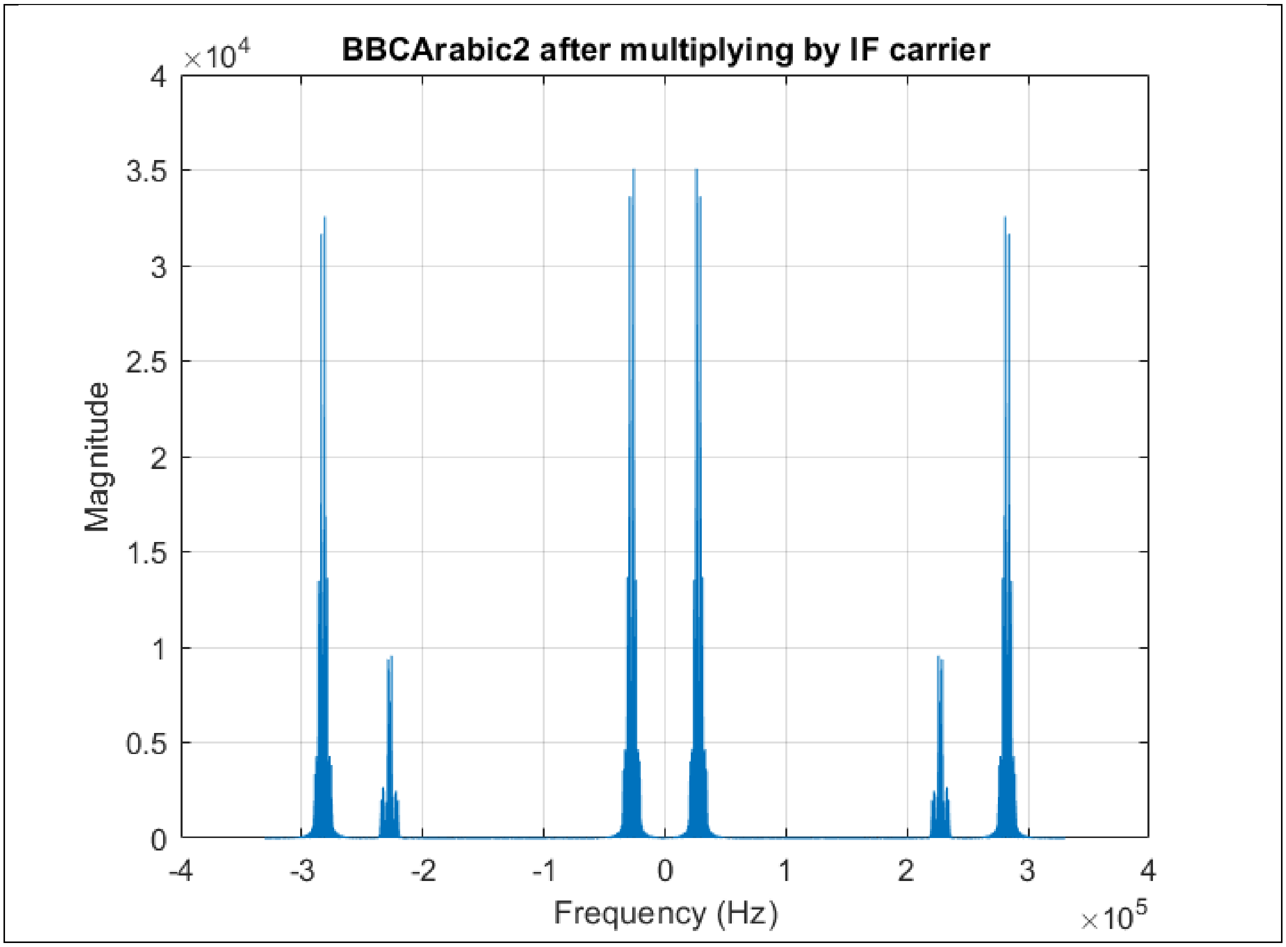
filtered Base Band stage of BBCArabic2
Magnitude
Frequency (Hz)
×10^5
0
200
400
600
800
1000
1200
1400
1600
1800
-4
-3
-2
-1
0
1
2
3
4



5. *Performance evaluation without the RF stage*

The figures

Figure 7: output of the RF mixer (no RF filter)



BBCArabic2 after multiplying by IF carrier
×10^4
4
3.5
3
2.5
2
1.5
1
0.5
0
Magnitude
-4
-3
-2
-1
0
1
2
3
4
Frequency (Hz)
×10^5

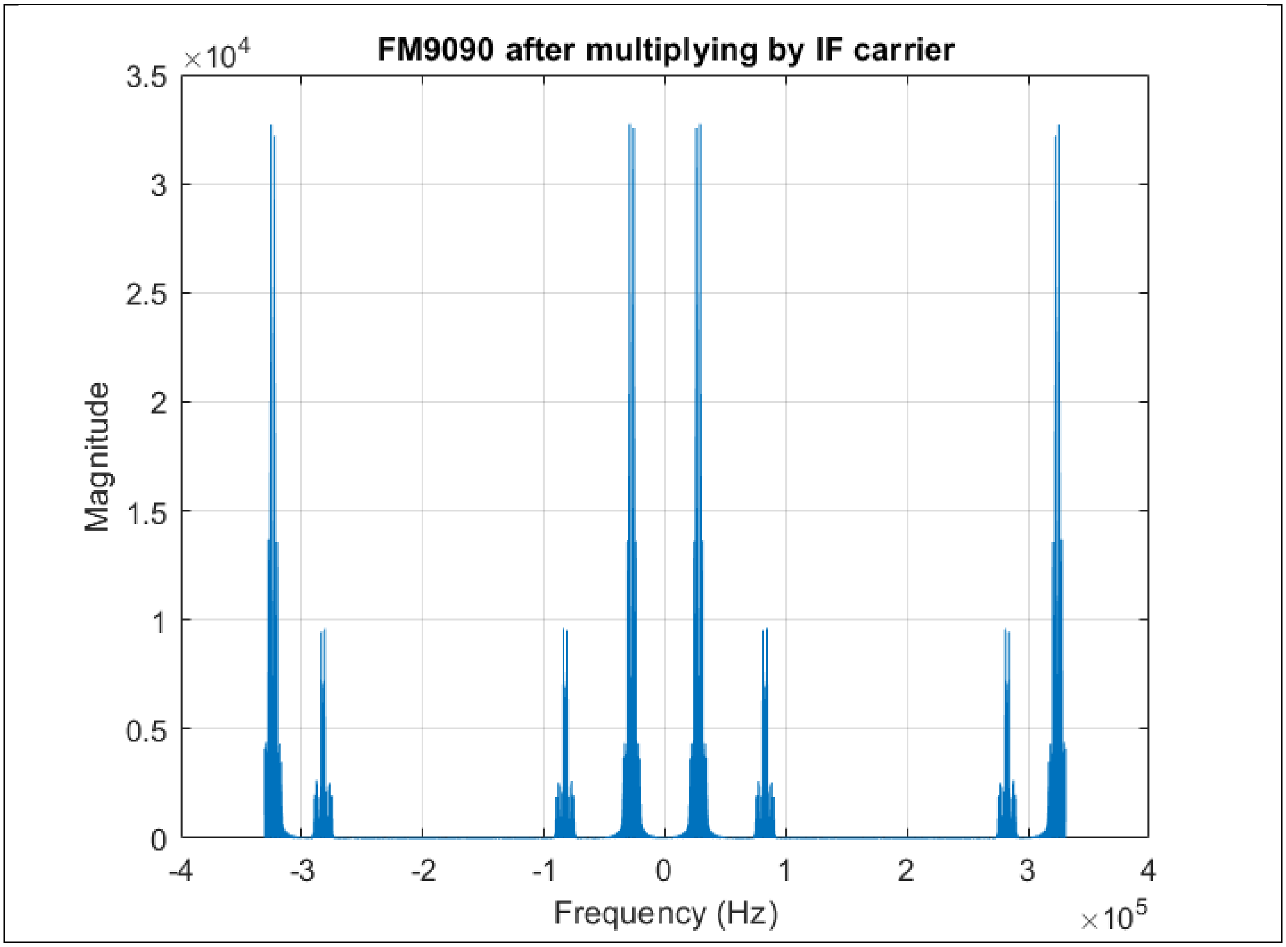
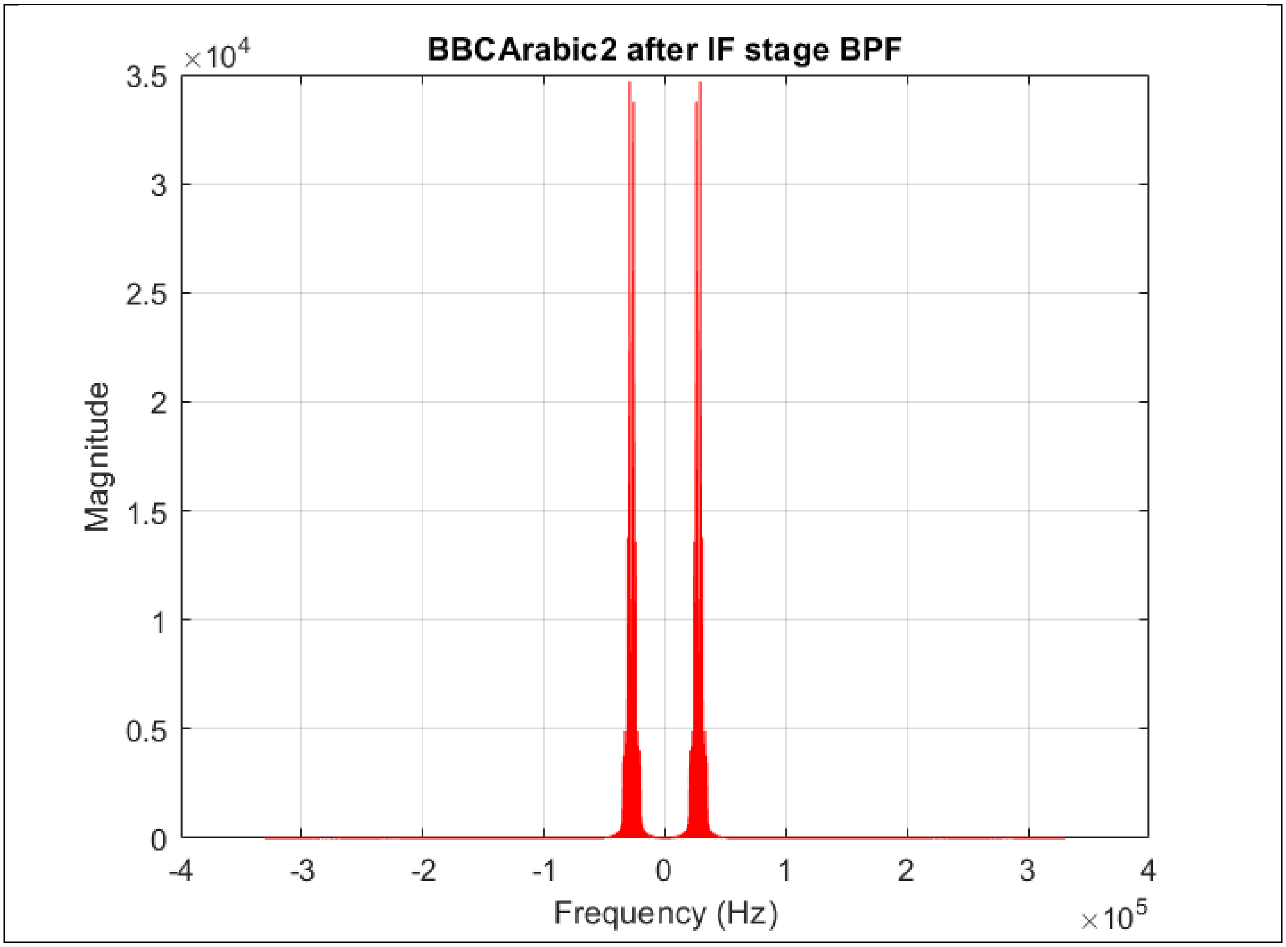


Figure 8: Output of the IF filter (no RF filter)


BBCArabic2 after IF stage BPF
Magnitude
Frequency (Hz)
$\times10^4$
$\times10^5$
3.5
3
2.5
2
1.5
1
0.5
0
-4
-3
-2
-1
0
1
2
3
4

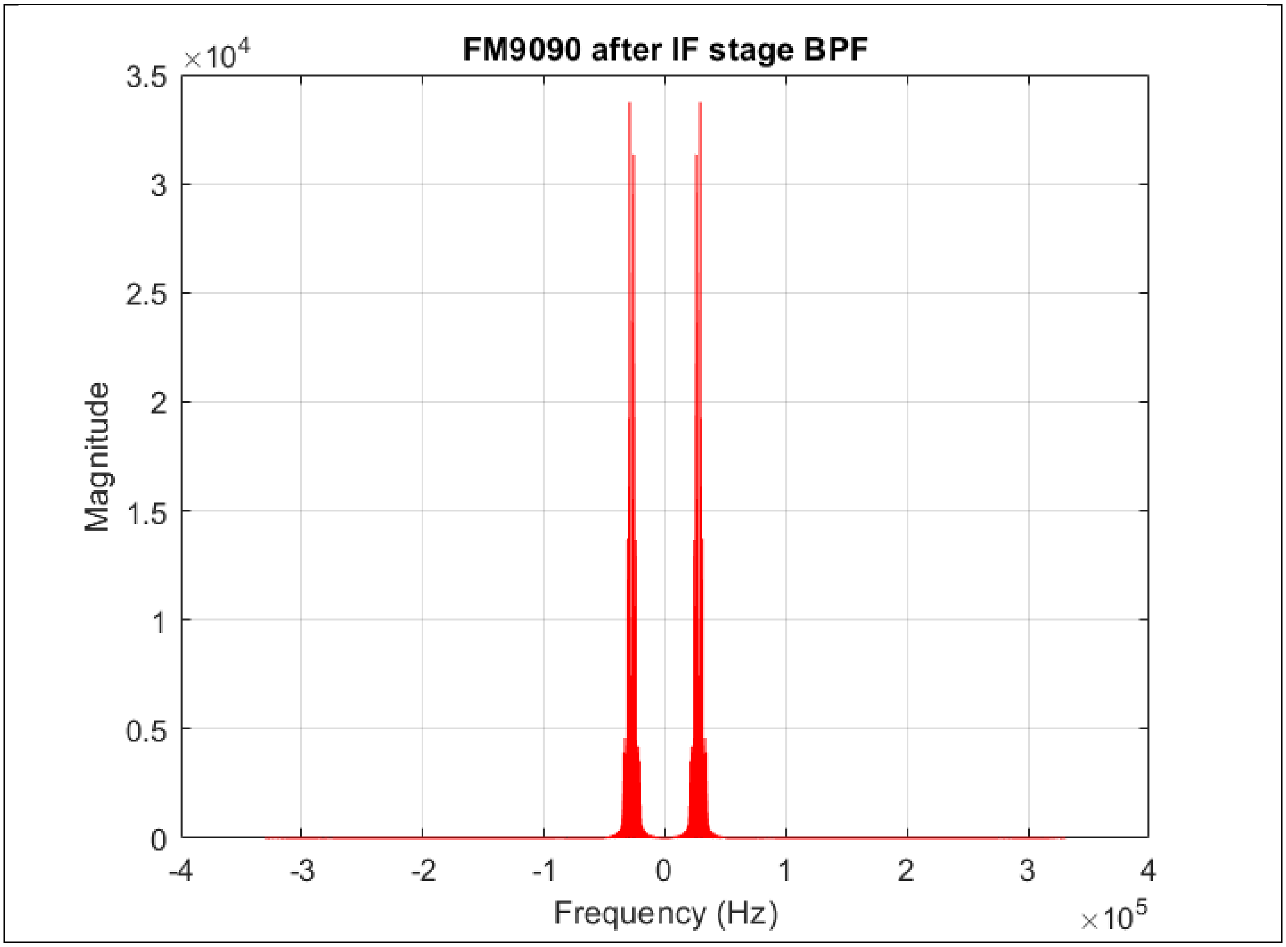
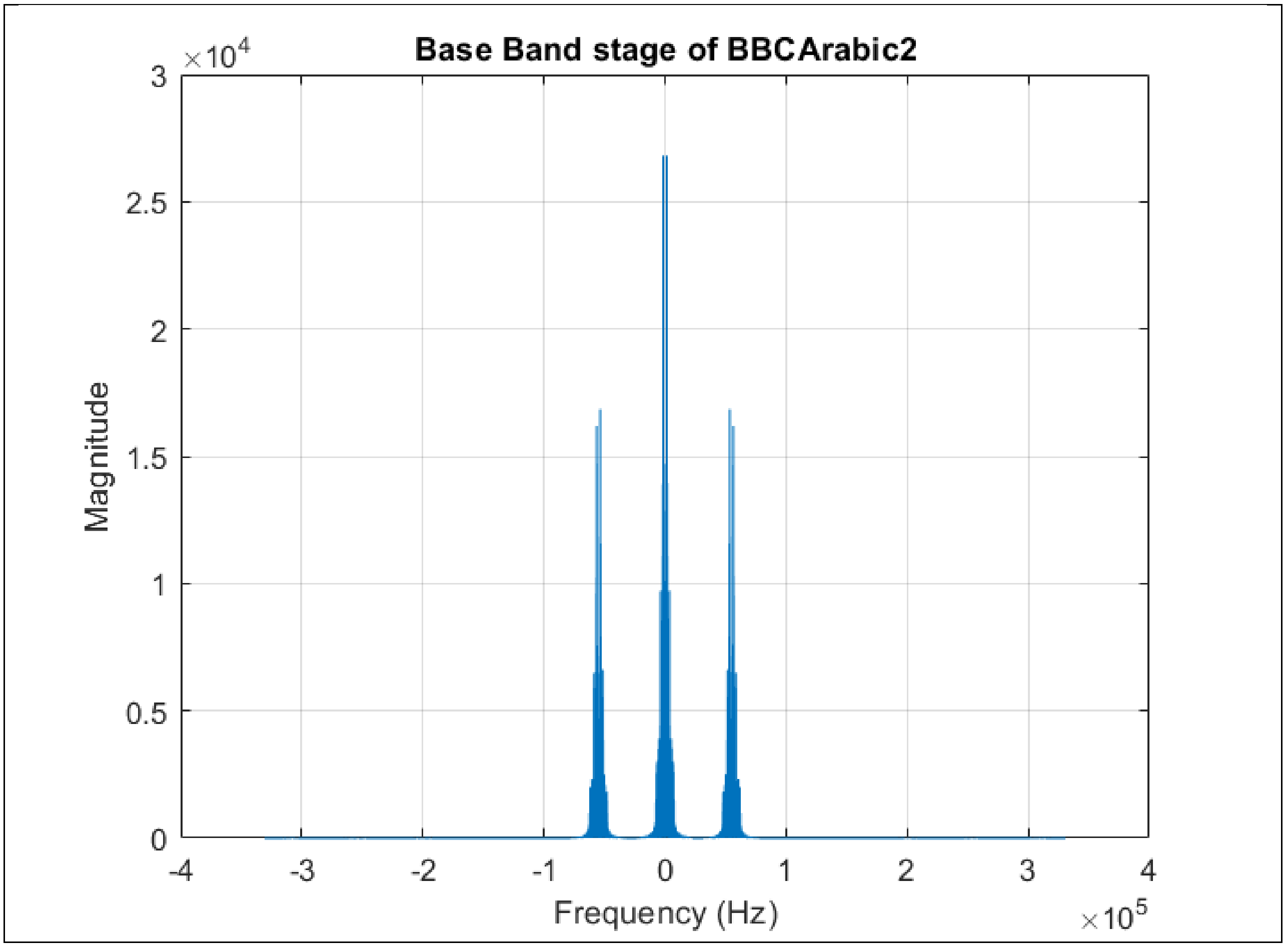


Figure 9: Output of the IF mixer before the LPF (no RF filter)



Base Band stage of BBCArabic2
×10^4
3
2.5
2
1.5
1
0.5
0
Magnitude
-4
-3
-2
-1
0
1
2
3
4
Frequency (Hz)
×10^5

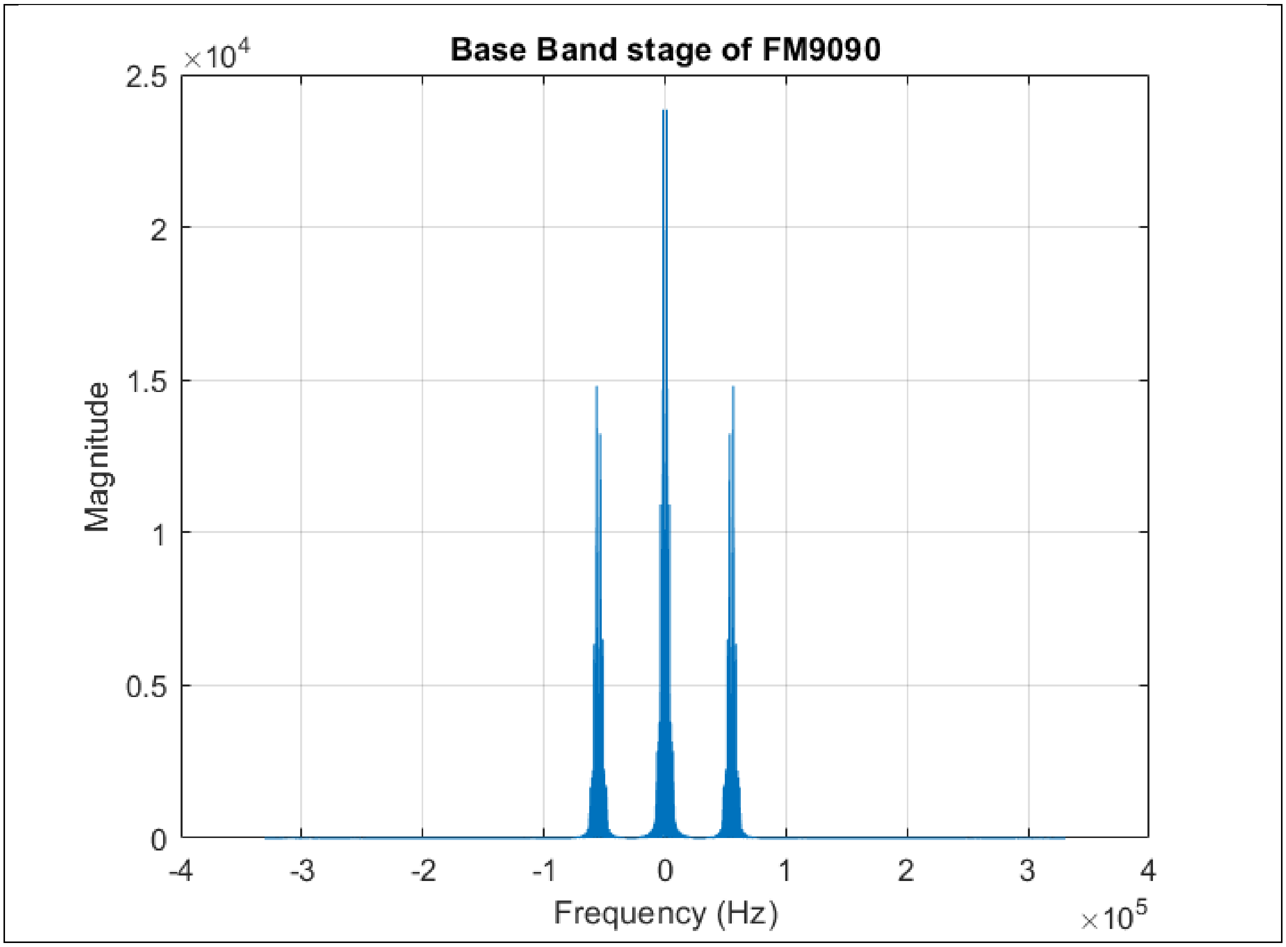
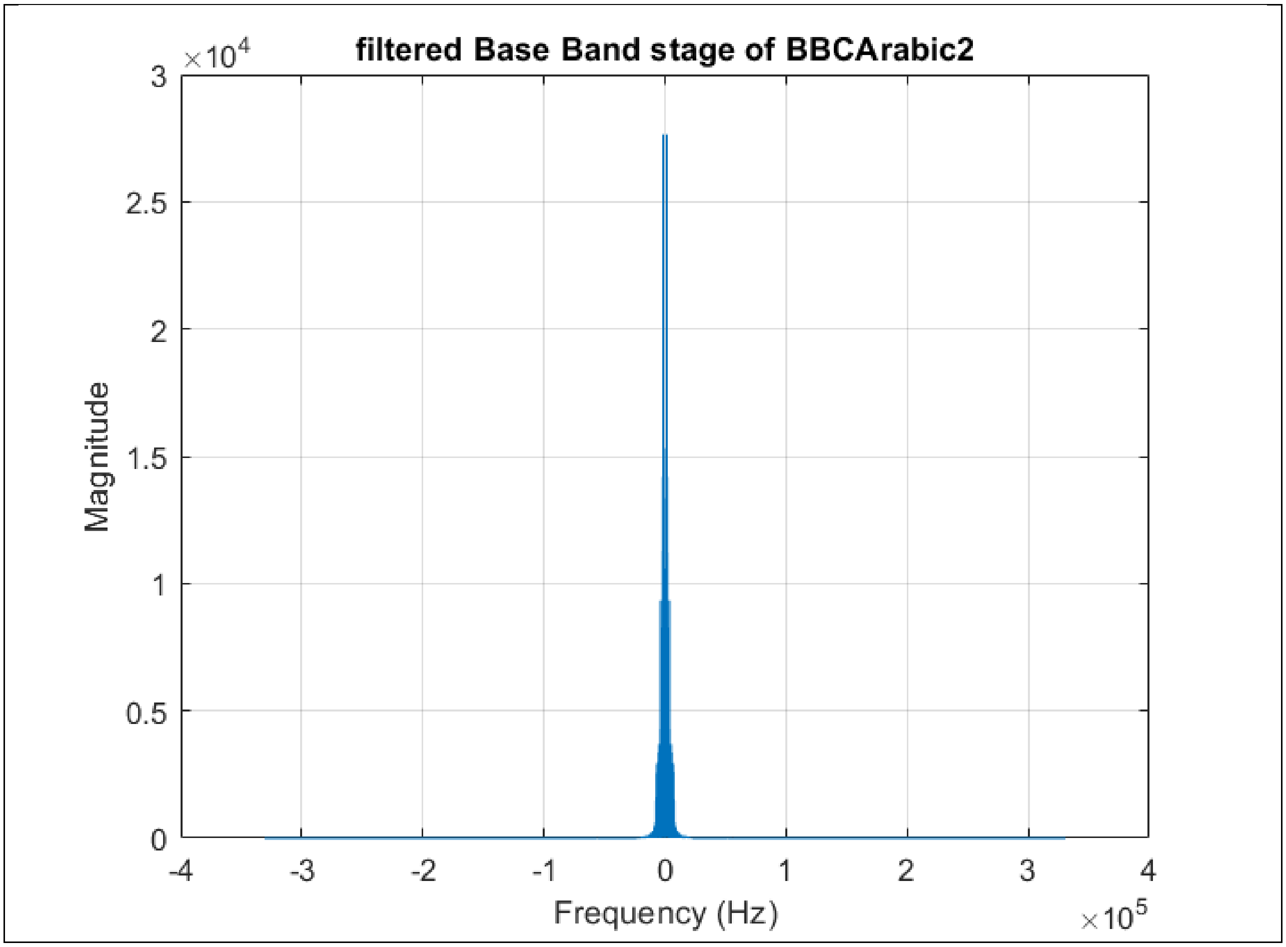
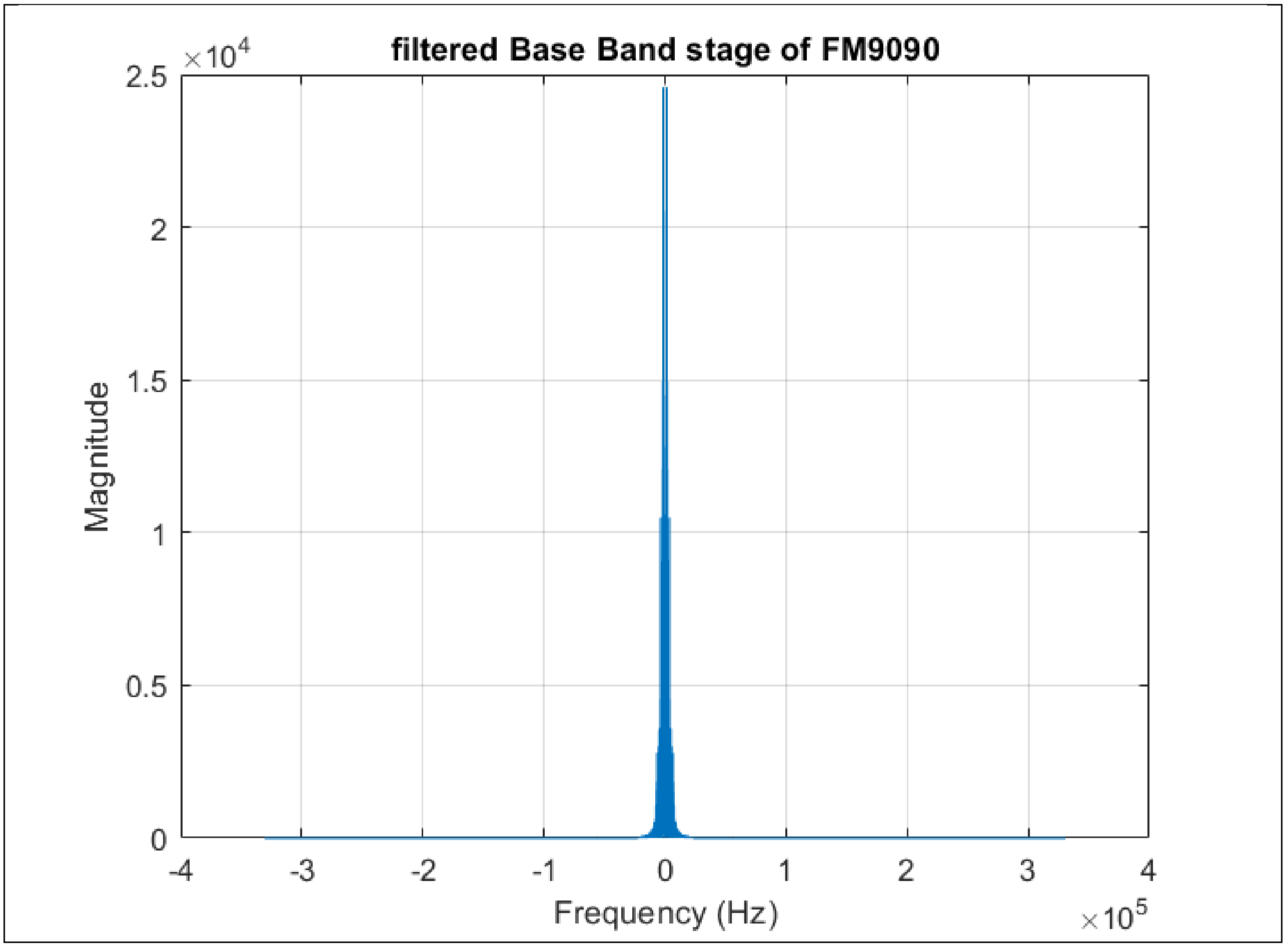


Figure 10: Ouptut of the LPF (no RF filter)



filtered Base Band stage of BBCArabic2
×10^4
3
2.5
2
1.5
1
0.5
0
Magnitude
-4
-3
-2
-1
0
1
2
3
4
Frequency (Hz)
×10^5



6. *Comment on the output sound*

In case of the presence of RF stage (bandpass filter), I can hear the each message of the two message separately clearly because the RF stage filtered each signal alone so nothing is wrong.

While when we remove the RF stage (bandpass filter) the two messages are present together so when seeing the result of the hand analysis we will find that when multiplying by the first oscillator with frequency equals to IF added to the carrier frequency of the first message we will see from the hand analysis that:

-The first message in the spectrum at frequencies: 27.5 KHz & 227.5 KHz
-The second message in the spectrum at frequencies: 27.5 KHz & 282.5 KHz

So we can see that the two messages come together at the IF frequency (27.5 KHz) and this is the frequency which we make BPF around it after multiplying by IF oscillator so we will find the two messages together when we try to

```
receive the first message , while the second message we will receive it alone
successfully when we try to receive it in this case because according to the
hand analysis (when multiplying by the second oscillator with frequency
equals to IF added to the carrier frequency of the second message)we will
find that:

-The first message in the spectrum at frequencies: 82.5 KHz & 282.5 KHz
-The second message in the spectrum at frequencies: 27.5 KHz & 227.5 KHz

So we can see that only the second message is at the IF frequency (27.5 KHz)
and this is the frequency which we make BPF around it after multiplying by IF
oscillator so we will successful receive the second message alone when we try
to in this case
```

What happens (in terms of spectrum and the sound quality) if the receiver oscillator has frequency offset by 0.1 KHz and 1 KHz

```
In terms of spectrum: the oscillator offset makes the message shifts form the
IF frequency by the offset value; So:

-at 0.1 KHz offset, the message is shifted in the spectrum with a small
amount which can be negligible shift thus we can consider the message in the
spectrum is still inside the IF BPF range so we can say that nothing changed.

-at 1 KHz offset, the message is shifted in the spectrum with an amount we
can consider thus we can say that a part of the frequencies of the message
gets out of the IF BPF range also after applying the BPF we find that the
magnitude of the message in the spectrum is decreased.

In terms of the sound quality(according to the filter design, if it is
designed for the message to be strictly confined within the filter frequency
range then the frequency offset would affect it even if it is small ,While if
the filter is designed to be as wide as possible in its frequency range such
that it doesn't accept any neighbor message in the spectrum then for a
frequency offset to make effect it is required not to be small.)

-at 0.1 KHz offset, The sound quality is nearly the same and we can hear its
content.

-at 1KHz offset, The sound quality decreased & the sound become less clear
than before.
```

7. ***The code***

```
clear all;
clc;
```

```
%carrier frequencies
Fc_BBCArabic2 = 100000;
Fc_FM9090 = 155000;

% Specify the path to .wav files (input messages)
file_path_BBCArabic2 = 'E:\3rd Year Electronics &
Electrical Communications\1st Term\labs\comm
project\Short_BBCArabic2.wav';
file_path_FM9090 = 'E:\3rd Year Electronics & Electrical
Communications\1st Term\labs\comm
project\Short_FM9090.wav';

% Use audioread to read the .wav files (input messages)
% note that the second argument that is read from
audioread function(sample rate) is the same for all the
five messages (I saw it in workspace while debugging) so
I worked on one of them
[BBCArabic2, Fm1] = audioread(file_path_BBCArabic2);
[FM9090, fs] = audioread(file_path_FM9090);
Fm2=fs;
% adding 2 channels to make single channel
BBCArabic2 = BBCArabic2(:,2) + BBCArabic2(:,1);
FM9090 = FM9090(:,2) + FM9090(:,1);

%increase the sampling freq by factor & using interp
function
 message_BBCArabic2 = interp(BBCArabic2,15);
 message_FM9090 = interp(FM9090,15);

 fs = fs * 15;

% get the max length of all messages
Messages_lengthes =
[length(message_BBCArabic2),length(message_FM9090)];%,len
gth(message_QuranPalestin),length(message_RussianVoice),l
ength(message_SkyNewsArabia)];
maxMessageLength = max(Messages_lengthes);

% make all messages have the same length
numZerosToAddIn_FM9090 = (maxMessageLength -
length(message_FM9090));
numZerosToAddIn_BBCArabic2 = (maxMessageLength -
length(message_BBCArabic2));
```

```
message_FM9090 = [message_FM9090;
zeros(numZerosToAddIn_FM9090,1)];
message_BBCArabic2 = [message_BBCArabic2;
zeros(numZerosToAddIn_BBCArabic2,1)];

% make fast fourier transform
BBCArabic2_spectrum =
fft(message_BBCArabic2,maxMessageLength);
FM9090_spectrum = fft(message_FM9090,maxMessageLength);

%plot the messages spectrum
f = fs * (0:(maxMessageLength/2))/maxMessageLength; %
Frequency axis
magnitude_spectrum_BBCArabic2 = 2/maxMessageLength *
abs(BBCArabic2_spectrum(1:maxMessageLength/2+1));
magnitude_spectrum_FM9090 = 2/maxMessageLength *
abs(FM9090_spectrum(1:maxMessageLength/2+1));

figure;
subplot(1,2,1);
plot(f, magnitude_spectrum_BBCArabic2);
title('BBCArabic2 Spectrum');
xlabel('Frequency (Hertz)');
ylabel('Magnitude');
grid on;
subplot(1,2,2);
plot(f, magnitude_spectrum_FM9090);
title('FM9090 Spectrum');
xlabel('Frequency (Hertz)');
ylabel('Magnitude');
grid on;

%AM_DSB_SC modulation using ammod function
dsb_sc_BBCArabic2  =
(ammod(message_BBCArabic2,Fc_BBCArabic2,fs));
dsb_sc_FM9090  = (ammod(message_FM9090,Fc_FM9090,fs));
TDM = dsb_sc_BBCArabic2 + dsb_sc_FM9090;

%plot the modulated signals spectrum
fft_of_modulated_BBCArabic2 = fft(dsb_sc_BBCArabic2);
fft_of_modulated_FM9090 = fft(dsb_sc_FM9090);

%Plotting FDM
f = linspace(-fs/2, fs/2,
```

```
length(fft_of_modulated_BBCArabic2)); % Frequency
vector(X-axis)
figure;
plot(f, fftshift(abs(fft_of_modulated_BBCArabic2)))
hold on;
plot(f, fftshift(abs(fft_of_modulated_FM9090)))
legend('BBCArabic2','FM9090')%,'QuranPalestin','RussianVo
ice','SkyNewsArabia')
title('FDM');
xlabel('Frequency (Hertz)');
ylabel('Magnitude');

%applying filters to get each message from the FDM
BBCArabic2_BandPassObject =
fdesign.bandpass('Fst1,Fp1,Fp2,Fst2,Ast1,Ap,Ast2', 82000,
85000, 115000, 118000, 60, 1, 60, fs);
BBCArabic2_BandPassFilter =
design(BBCArabic2_BandPassObject);
filtered_BBCArabic2 = filter(BBCArabic2_BandPassFilter,
TDM);
figure;
plot(f,fftshift(abs(fft((filtered_BBCArabic2)))));
title('BBCArabic2 Spectrum after filtering from FDM');
xlabel('Frequency (Hertz)');
ylabel('Magnitude');
grid on;

FM9090_BandPassObject =
fdesign.bandpass('Fst1,Fp1,Fp2,Fst2,Ast1,Ap,Ast2',
125000, 130000, 180000, 181500, 60, 1, 60, fs);
FM9090_BandPassFilter = design(FM9090_BandPassObject);
filtered_FM9090 = filter(FM9090_BandPassFilter, TDM);
figure;
plot(f,fftshift(abs(fft((filtered_FM9090)))));
title('FM9090 Spectrum after filtering from FDM');
xlabel('Frequency (Hertz)');
ylabel('Magnitude');
grid on;

%uncomment the two lines below and comment lines from 77
to 95 to remove IF
%stage without any problems

% filtered_FM9090 = TDM;
```

```
% filtered_BBCArabic2 = TDM;
%% IF stage
offset = 0;% change offset if you want
IF = 27500+offset; %Hz
IF_carrier_BBCArabic2 = Fc_BBCArabic2 + IF; % carrier
frequency1
IF_carrier_FM9090 = Fc_FM9090 + IF; % carrier frequency2

t_BBCArabic2 = (1:1:length(filtered_BBCArabic2))';
t_FM9090 = (1:1:length(filtered_FM9090))';

carrier_BBCArabic2_IF = cos(2 * pi *
IF_carrier_BBCArabic2 * t_BBCArabic2 * (1 / (fs)));
carrier_FM9090_IF2 = cos(2 * pi * IF_carrier_FM9090 *
t_FM9090 * (1 / (fs)));

carrier_BBCArabic2_IF(end + length(filtered_BBCArabic2) -
length(carrier_BBCArabic2_IF), 1) = 0;
carrier_FM9090_IF2(end + length(filtered_FM9090) -
length(carrier_FM9090_IF2), 1) = 0;

IF_received_BBCArabic2 = filtered_BBCArabic2 .*
carrier_BBCArabic2_IF;
IF_received_FM9090 = filtered_FM9090 .*
carrier_FM9090_IF2;

%prepearing for spectrum plotting
spectrum_IF_RECEIVED_BBCArabic2 =
fftshift(fft(IF_received_BBCArabic2));
spectrum_IF_RECEIVED_FM9090 =
fftshift(fft(IF_received_FM9090));

%adjusting X-axis for each then plotting each message
after IF oscillator
f_RECEIVED_BBCArabic2 = (-
length(spectrum_IF_RECEIVED_BBCArabic2) /
2:1:length(spectrum_IF_RECEIVED_BBCArabic2) / 2 - 1)';
f_RECEIVED_FM9090 = (-length(spectrum_IF_RECEIVED_FM9090)
/ 2:1:length(spectrum_IF_RECEIVED_FM9090) / 2 - 1)';

figure;
plot(f_RECEIVED_BBCArabic2 * fs /
length(spectrum_IF_RECEIVED_BBCArabic2),
abs(spectrum_IF_RECEIVED_BBCArabic2));
```

```
title("BBCArabic2 after multiplying by IF carrier");
xlabel("Frequency (Hz)");
ylabel("Magnitude");
grid on;

figure;
plot(f_RECEIVED_FM9090 * fs /
length(spectrum_IF_RECEIVED_FM9090),
abs(spectrum_IF_RECEIVED_FM9090));
title("FM9090 after multiplying by IF carrier");
xlabel("Frequency (Hz)");
ylabel("Magnitude");
grid on;
%% IF stage BPF

BBCArabic2_BandPassObject_IFStage =
fdesign.bandpass('Fst1,Fp1,Fp2,Fst2,Ast1,Ap,Ast2', 4000,
5000, 50000, 55000, 60, 1, 60, fs);
BBCArabic2_BandPassFilter_IfStage =
design(BBCArabic2_BandPassObject_IFStage,'equiripple');
filtered_BBCArabic2_IFStage =
filter(BBCArabic2_BandPassFilter_IfStage,
IF_received_BBCArabic2);
spectrum_filtered_BBCArabic2_IFStage =
fftshift(abs(fft(filtered_BBCArabic2_IFStage)));
f_RECEIVED_BBCArabic2_BPF = (-
length(spectrum_filtered_BBCArabic2_IFStage) /
2:1:length(spectrum_filtered_BBCArabic2_IFStage) / 2 -
1)';
figure;
plot(f_RECEIVED_BBCArabic2_BPF * fs /
length(spectrum_filtered_BBCArabic2_IFStage),abs(spectrum
_filtered_BBCArabic2_IFStage),'r');
title("BBCArabic2 after IF stage BPF");
xlabel("Frequency (Hz)");
ylabel("Magnitude");
grid on;

FM9090_BandPassObject_IFStage =
fdesign.bandpass('Fst1,Fp1,Fp2,Fst2,Ast1,Ap,Ast2', 4000,
5000, 50000, 55000, 60, 1, 60, fs);
FM9090_BandPassFilter_IfStage =
design(FM9090_BandPassObject_IFStage,'equiripple');
filtered_FM9090_IFStage =
```

```
filter(FM9090_BandPassFilter_IfStage,
IF_received_FM9090);
spectrum_filtered_FM9090_IFStage =
fftshift(abs(fft(filtered_FM9090_IFStage)));
f_RECEIVED_FM9090_BPF = (-
length(spectrum_filtered_FM9090_IFStage) /
2:1:length(spectrum_filtered_FM9090_IFStage) / 2 - 1)';
figure;
plot(f_RECEIVED_FM9090_BPF * fs /
length(spectrum_filtered_FM9090_IFStage),abs(spectrum_fil
tered_FM9090_IFStage),'r');
title("FM9090 after IF stage BPF");
xlabel("Frequency (Hz)");
ylabel("Magnitude");
grid on;
%% back to base band
T_BBCArabic2 =
(1:1:length(filtered_BBCArabic2_IFStage))';
T_FM9090 = (1:1:length(filtered_FM9090_IFStage))';

carrier_BBCArabic2_Base_Band = cos(2 * pi * IF *
T_BBCArabic2 * (1 / (fs)));
carrier_BBCArabic2_Base_Band(end +
length(filtered_BBCArabic2) -
length(carrier_BBCArabic2_Base_Band), 1) = 0;
Base_Band_received_BBCArabic2 =
filtered_BBCArabic2_IFStage .*
carrier_BBCArabic2_Base_Band;
spectrum_Base_Band_received_BBCArabic2 =
fftshift(fft(Base_Band_received_BBCArabic2));
f_BBCArabic2_BASE_BAND = (-
length(spectrum_Base_Band_received_BBCArabic2) /
2:1:length(spectrum_Base_Band_received_BBCArabic2) / 2 -
1)';

figure;
plot(f_BBCArabic2_BASE_BAND * fs /
length(spectrum_Base_Band_received_BBCArabic2),
abs(spectrum_Base_Band_received_BBCArabic2));
title("Base Band stage of BBCArabic2");
xlabel("Frequency (Hz)");
ylabel("Magnitude");
grid on;
```

```
BBCArabic2_lowPassObjectBaseBandFilter =
fdesign.lowpass('Fp,Fst,Ap,Ast',20000,30000,1,80,fs);
BBCArabic2_lowPassFilterBaseBandStage =
design(BBCArabic2_lowPassObjectBaseBandFilter,'equiripple
');
filtered_BBCArabic2_BaseBandStage =
filter(BBCArabic2_lowPassFilterBaseBandStage,
Base_Band_received_BBCArabic2);
spectrum_filtered_BBCArabic2_BaseBandStage =
fftshift(abs(fft(filtered_BBCArabic2_BaseBandStage)));

figure;
plot(f_BBCArabic2_BASE_BAND * fs /
length(spectrum_Base_Band_received_BBCArabic2),
abs(spectrum_filtered_BBCArabic2_BaseBandStage));
title("filtered Base Band stage of BBCArabic2");
xlabel("Frequency (Hz)");
ylabel("Magnitude");
grid on;
% Play the audio
Demodulated_BBCArabic2 =
downsample(filtered_BBCArabic2_BaseBandStage,15);
%uncomment sound function to play the sound or uncomment
audiowrite function
%to save audio as wav format
sound(Demodulated_BBCArabic2,Fm1);
audiowrite('Demodulated_BBCArabic2.wav',Demodulated_BBCAr
abic2,Fm1);

carrier_FM9090_Base_Band = cos(2 * pi * IF * T_FM9090 *
(1 / (fs)));
carrier_FM9090_Base_Band(end + length(filtered_FM9090) -
length(carrier_FM9090_Base_Band), 1) = 0;
Base_Band_received_FM9090 = filtered_FM9090_IFStage .*
carrier_FM9090_Base_Band;
spectrum_Base_Band_received_FM9090 =
fftshift(fft(Base_Band_received_FM9090));
f_FM9090_BASE_BAND = (-
length(spectrum_Base_Band_received_FM9090) /
2:1:length(spectrum_Base_Band_received_FM9090) / 2 - 1)';

figure;
plot(f_FM9090_BASE_BAND * fs /
length(spectrum_Base_Band_received_FM9090),
```

```
abs(spectrum_Base_Band_received_FM9090));
title("Base Band stage of FM9090");
xlabel("Frequency (Hz)");
ylabel("Magnitude");
grid on;

FM9090_lowPassObjectBaseBandFilter =
fdesign.lowpass('Fp,Fst,Ap,Ast',20000,30000,1,80,fs);
FM9090_lowPassFilterBaseBandStage =
design(FM9090_lowPassObjectBaseBandFilter,'equiripple');
filtered_FM9090_BaseBandStage =
filter(FM9090_lowPassFilterBaseBandStage,
Base_Band_received_FM9090);
spectrum_filtered_FM9090_BaseBandStage =
fftshift(abs(fft(filtered_FM9090_BaseBandStage)));

figure;
plot(f_FM9090_BASE_BAND * fs /
length(spectrum_Base_Band_received_FM9090),
abs(spectrum_filtered_FM9090_BaseBandStage));
title("filtered Base Band stage of FM9090");
xlabel("Frequency (Hz)");
ylabel("Magnitude");
grid on;
% Play the audio
Demodulated_FM9090 =
downsample(filtered_FM9090_BaseBandStage,15);
%uncoment sound function to play the sound or uncomment
audiowrite dunction
%to save audio as wav format
sound(Demodulated_FM9090,Fm2);
audiowrite('Demodulated_FM9090.wav',Demodulated_FM9090,Fm
2);
```

1) Multiplier:

- Gilbert-cell multiplier

Exploits linear relationship between collector current & transistor transconductance, initially designed with long-tailed pair of transistors & fixed non-linearity & temperature dependence issues using current-based operations & leveraging logarithmic properties. The resulting circuit, with a differential current input & unipolar current input, achieves linear transfer function ideal for multiplication tasks. When cross-coupled, circuit transforms to versatile four-quadrant multiplier, used in analog computation.

2) Switching modulator:

- Diode-ring mixer

Used for high-performance applications, relies on diodes for switching. Requires a high LO drive, ≈ 1 watt, for strong diode conduction and effective signal conversion. Challenges: poor impedance control & significant coupling between ports due to diode nonlinearity. Inherent conversion loss=3.92 dB, increase in practical models due to diode resistances & transformer losses. Users evaluate signal handling with a "Level" rating; for instance, a Level-17 mixer needs +17 dBm LO drive & costs ≈ $10 in small quantities.