```
clear, clc;

%The Cosine signal passes through a Turner Gain of 10dB, bandpass filter,
%Mixer of 3.5MHz, Lowpass filter BW=5 MHz, and Variable gain amplifier.
%For the Demodulator the signal then passes through an ADC 28.8MHz, splits
%into both a cos/sine mixer, lowpass filter, and decimation downsapling
%filter. The output is two signals (I & Q) which can be added to create a
%complex signal.

%Below I created a Cosine signal with a 4 GHz sampling rate and 100 MHz
%Frequency. I created a frequency vs domain graph. I use the 'plotter'
%function to call it.
Fs = 4.032e9;
dt = 1/Fs;
t =  (0:dt:0.0002-dt)';
Fm = 100e6;
Sig2 = cos(2*pi*(Fm-1e6)*t);
sig1 = cos(2*pi*(Fm+1e6)*t)+Sig2;
figure(1)
plotter(sig1, Fs, 'Cosine Signal')


%RAFAEL R820T2 TUNER SECTION

%Turner gain of 10dB (lna)
agc1 = comm.AGC('AveragingLength',10);
sigAGC = agc1(sig1);
plotter(sigAGC, Fs, 'Gain')

%RF Filter of 6MHZ(bandpass filter)
hRF = firpm(100,[0 Fm-50e6 Fm-3.5e6 Fm+3.5e6 Fm+50e6 (Fs/2)]*2/Fs, [0 0 1 1 0 0]);
figure(2)
freqz(hRF)
RF = filter(hRF,1,sigAGC);
figure(3)
plotter(RF, Fs, 'Bandpass Filter')

%Mixer of 3.5MHz
Fif =  3.57e6;
Flo = Fm - Fif;
mix = RF.*cos(2*pi*Flo*t);
figure(4)
plotter(mix, Fs, 'Mixer')

%IF Filter of BW=5MHz (lowpass filter)
hIF = firpm(1024,[0 Fif+3e6 Fif+10e6 (Fs/2)]*2/Fs, [1 1 0 0]);
figure(5)
freqz(hIF)
IF = filter(hIF,1,mix);
plotter(IF, Fs, 'Lowpass Filter')

%Variable Gain Amplifier
%vga1 = serdes.VGA('Gain', 10)
```

```
vga = IF * 10;
figure(5)
plotter(vga, Fs, 'VGA')


%RTL2832U DEMODULATOR

%Decimate by a factor of 275
D = 139;
f = vga(1:D:end);
Fs = Fs/D;

%ADC of 8-bit and 28.8MHz range
B = 8;
N = 2^(B-1);
out   = (round(f .* N)) ./ N;
error = f-out;
plotter(out, Fs, 'ADC')

%Cosine and Sine Mixers
n = (1:length(out)).';
cos1 = out.*cos(2*pi*(3.57e6/Fs)*n);
sin1 = out.*sin(2*pi*(3.57e6/Fs)*n);
plotter(cos1, Fs, 'Cosine Mixer')
plotter(sin1, Fs, 'Sine Mixer')

%Decimation/downsample filter for every 10
real = decimate(cos1, 10);
img = decimate(sin1, 10);
plotter(real, (Fs/10), 'Cosine Downsampling')
plotter(img, (Fs/10), 'Sine Downsampling')

%Combining both signals (I & Q) to get original signal
Final = real + 1j * img;
plotter(Final, (Fs/10), 'Final Complex Signal')
plot(real)
```

```
function plotter(input,sampleFreq,plotTitle)
%DOCUMENTATION
% Plots the input signal in the frequency domain.
% Gets the appropriate length for plotting.
% Then plots the graph with


fs = sampleFreq;
n = length(input);
f = (-n/2:n/2-1)*(fs/n);
Spectrum = 20*log10(abs(fft(input)));
Spectrum = fftshift(Spectrum);
plot(f,Spectrum);
xlabel('Frequency (Hz)');
ylabel('Amplitude (dB)');
title(plotTitle);
```

```
end
```