Homework 2

Single Sideband Generation

Name: Jinhan Zhang

Class: ECE 251A Digital Signal Processing I

Date: 01/23/2017

Single Sideband Generation

• Objective

Implement a system which uses Hilbert Transforms to generate single sideband signal.

• Background

For amplitude modulation wave, the upper side band and lower side band have the similar information, and also the carrier wave doesn't have any useful information. So we can complete the process of information transmission by just transmitting one side band to improve the efficiency. In this assignment, we use Hilbert Transform to generate the single sideband.

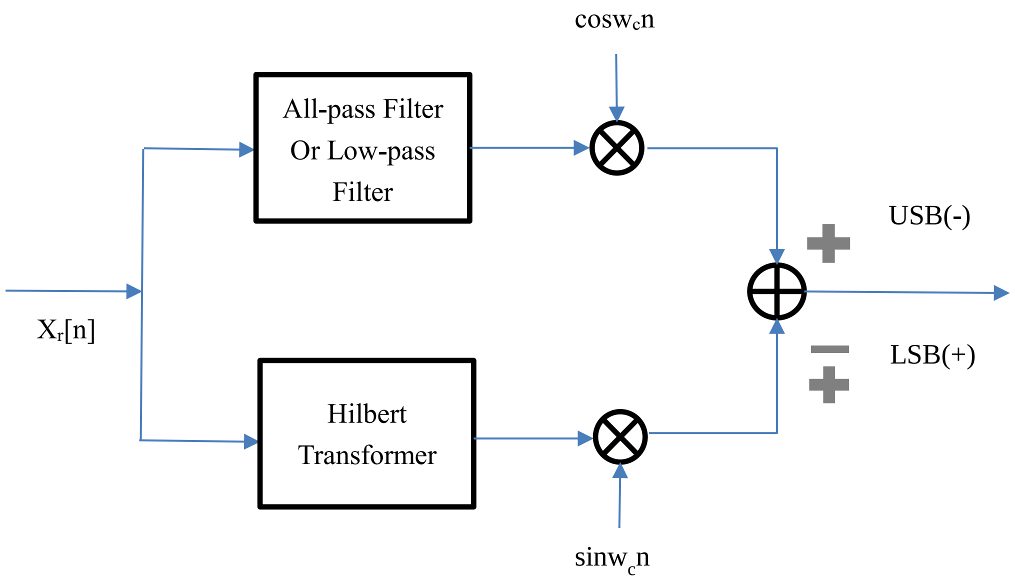
• Approach

Consider we have a 1024 points length signal and consist of three low frequency sinusoids (f1 = 0.05, f2 = 0.075, and f3 = 0.10 cycles/sample) separated in level by 10 dB (A1 > A2 > A3). In the lower path, we pass the signal through Hilbert Transform, and then modulate the signal by $\sin(\omega_c \mathrm{n})$. In the upper path, we firstly pass the signal through an all-pass (or low-pass filter which passes the three sinusoidal components of the original signal) in order to compensate the time delays caused by Hilbert Transform in the lower path. And then modulate the signal by $\cos(\omega_c \mathrm{n})$.

By upper path output summing/subtracting lower path output, we can get lower side band(LSB) or upper side band(USB) respectively.

- Results

Part A: The complete block diagram is shown as below.



Consider $x_r[n]$ as system input. In the lower path, we pass the signal through Hilbert Transformer to get the imaginary part, and followed by modulation through $\sin(\omega_c \mathrm{n})$. While in the upper path, we firstly need to compensate the time delays caused by Hilbert Transformer in the lower path. We do this by using an all-pass filter (or low-pass filter which passes the three sinusoidal components of the original signal). After this we can then modulate the filtered signal by $\cos(\omega_c \mathrm{n})$. Now we can get lower side band(LSB) by summing upper path output and lower path output, while get upper side band(USB) by subtracting lower path output from upper path output.

Part B:

In this part, we just create a 64-point FIR Hilbert transformer by firpm function, and will show its impulse response and transfer function plot in the next part.

Part C:

In Fig. 1 we show the impulse response of 64-point FIR Hilbert transformer created by firpm function. Fig. 2 shows the dB magnitude of 64-point FIR Hilbert transformer's transfer function while Fig.3 shows the phase characteristics. After blowing up the phase plot in the vicinity of f = 0 cycles/sample in Fig. 4, we can see that phase jumps from -95.87 to -92.63 radian in the range of [-0.01563 0.01563], which is the phase discontinuity. That is due to that $\mathrm{H}\left(e^{j\omega}\right)$ jumps from j to -j at 0 cycle/sample.

Part D:

Fig. 5 shows the dB magnitude of the original real part signal FFT, Fig. 6 shows the dB magnitude of the real part signal after time delay by all-pass filter. This step is needed because we should keep the time delays through the upper and lower paths identical. Fig. 7 shows the dB magnitude of the imaginary part obtained by Hilbert Transformer. Fig. 8 Shows the dB magnitude of the real part signal after modulated by $\cos(\omega_c \mathrm{n})$, while Fig. 9 shows that of the imaginary part signal after modulated by $\sin(\omega_c \mathrm{n})$. Fig. 10 and Fig. 11 shows the dB magnitude of lower side band and upper side band respectively. Although we can see a little flaw in the plot but the result is good in general. The dB magnitude of the complex signal composed by $\mathrm{x(n)} = x_r'(n) + jx_i(n)$ is shown in Fig. 12. After we multiply $\mathrm{x(n)}$ with $e^{+j2\pi f_c n}$, we see the plot shift 0.25 cycle/sample in the dB magnitude displayed by Fig. 13.

In additionally, the impulse response and transfer function of the low-pass filter introduced in upper path are shown in Fig. 14-16. We see it will pass the frequencies of all 3 sinusoids from Fig 15.

And a 256-point hamming window is used on FFTs above.

- Summary

In this assignment, we have generated single sideband by using the character of Hilbert Transform. We input the real part of complex sequence, and use Hilbert Transformer to get its imaginary part, and utilize the 2 parts to get the whole complex signal.

- Plots

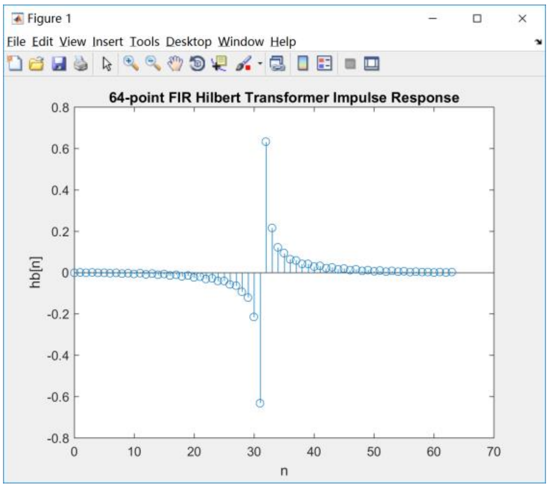


Fig. 1

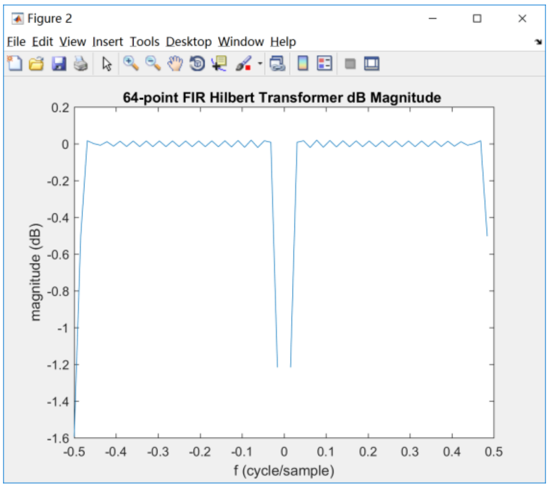


Fig. 2

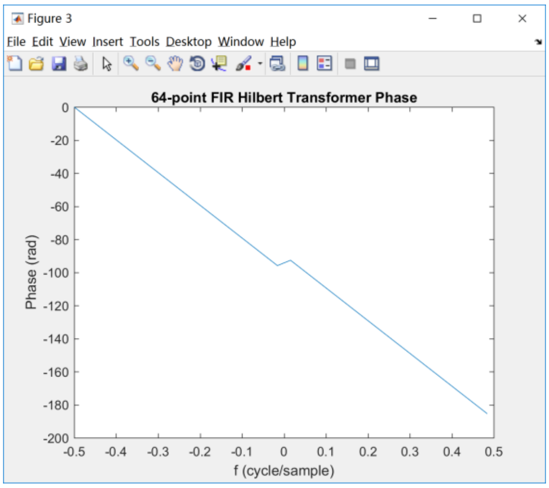


Fig. 3

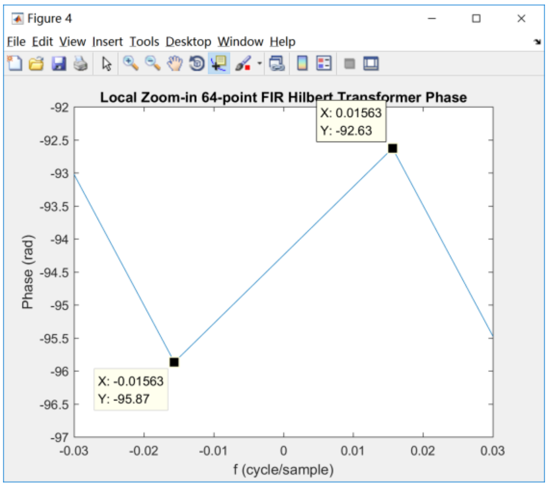


Fig. 4

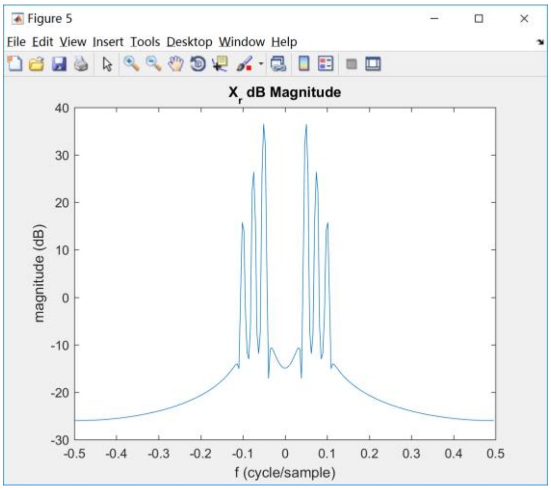


Fig. 5

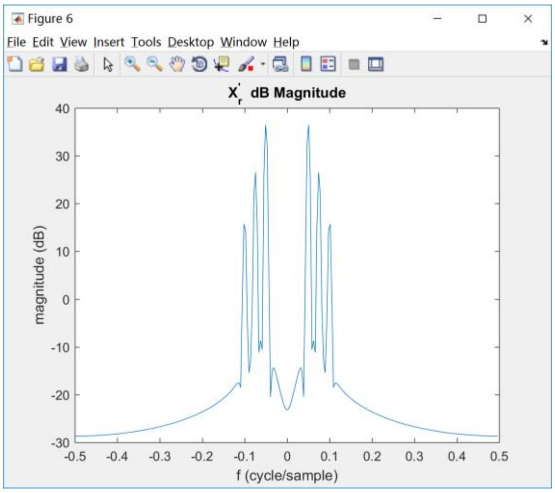


Fig. 6

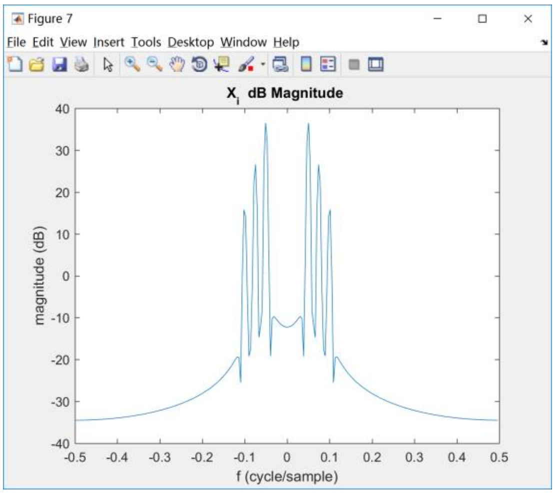


Fig. 7

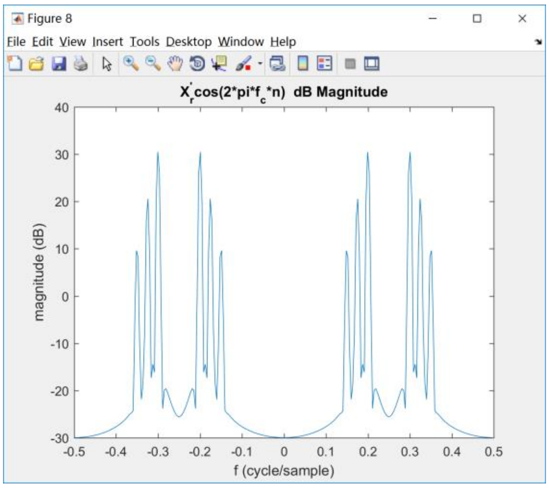


Fig. 8

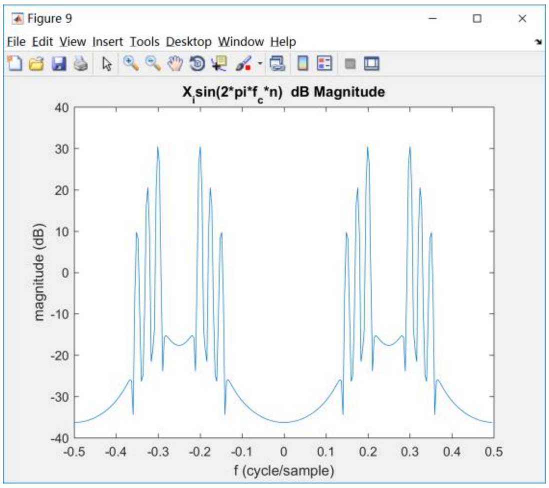


Fig. 9

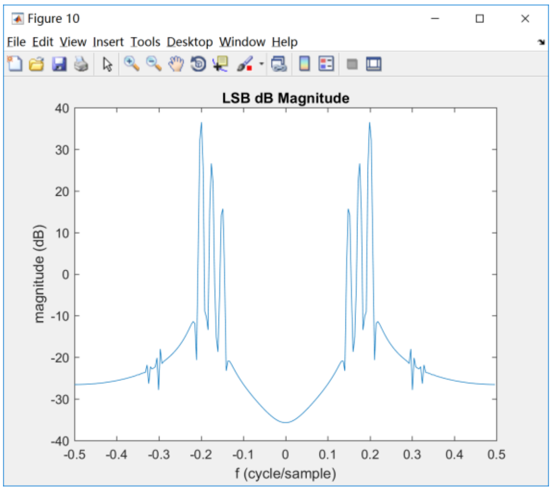


Fig. 10

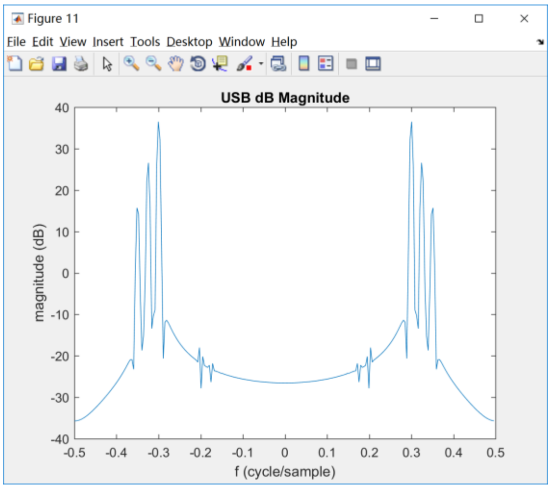


Fig. 11

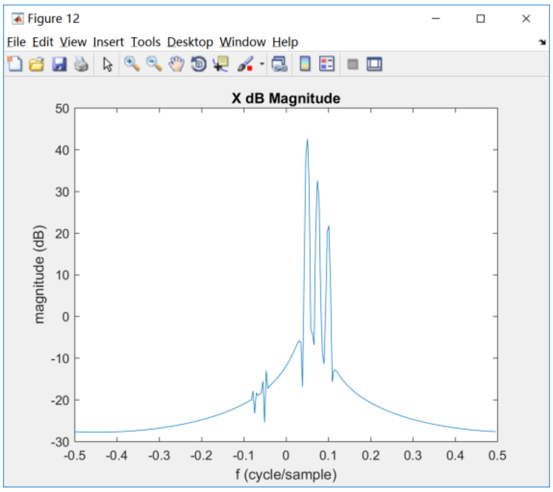


Fig. 12

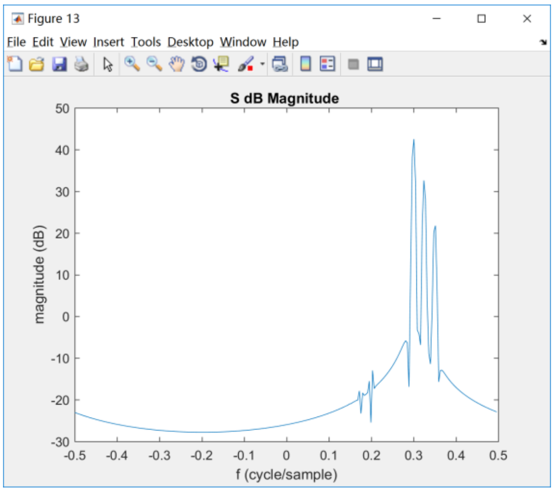


Fig. 13

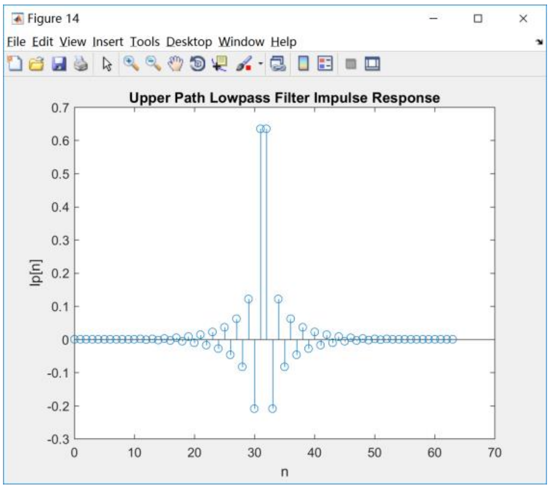


Fig. 14

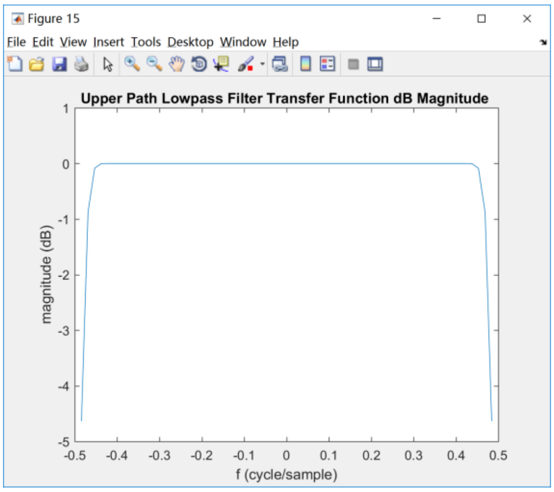


Fig. 15

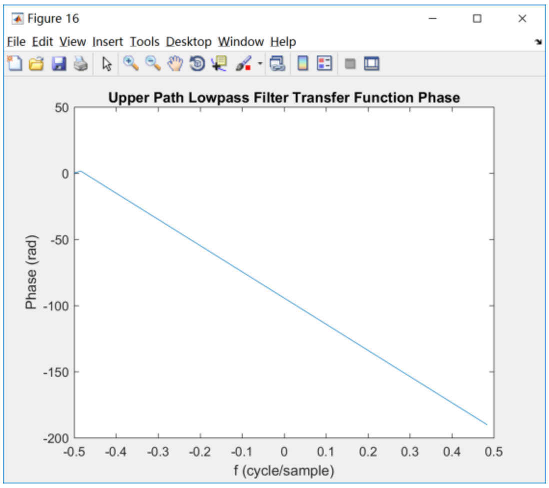


Fig. 16

- Appendix

Script:

```
clear;clc;close all;
%%  Prob B starts
NN=64;
F=[0.05,0.95];
A=[1,1];
hb=firpm(NN-1,F,A,'hilbert');
%% Prob C starts
figure(1);
stem([0:NN-1],hb);
title('64-point FIR Hilbert Transformer Impulse
Response');
```

```
xlabel('n');ylabel('hb[n]');

figure(2);
HB=fftshift(fft(hb,64));
plot([-0.5:1/64:0.5-1/64],20*log10(abs(HB)));
title('64-point FIR Hilbert Transformer dB Magnitude ');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');

figure(3);
plot([-0.5:1/64:0.5-1/64],unwrap(phase(HB)));
title('64-point FIR Hilbert Transformer Phase');
xlabel('f (cycle/sample)');ylabel('Phase (rad)');

figure(4);
plot([-0.5:1/64:0.5-1/64],unwrap(phase(HB)));
axis([-0.03 0.03 -97 -92]);
title('Local Zoom-in 64-point FIR Hilbert Transformer
Phase');
xlabel('f (cycle/sample)');ylabel('Phase (rad)');

%% Prob D Starts
f1=0.05;f2=0.075;f3=0.10;fc=0.25;
A_1=1;A_2=1/sqrt(10);A_3=1/10;   %20log10(A),so 10dB
equals 1/sqrt(10)
N=1024;
n=0:N-1;
window=hamming(256)';
x_r=A_1*cos(2*pi*f1*n)+A_2*cos(2*pi*f2*n)+A_3*cos(2*pi*f
3*n);

%%%xr(n)
figure(5);
xr_t=window.*x_r(100:355);
plot([-0.5:1/256:0.5-
1/256],20*log10(abs(fftshift(fft(xr_t)))));
title('X_{r} dB Magnitude');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');

%%%x'r(n)
figure(6);
delay_filter=firpm(NN-1,[0,0.85],[1,1]);
xr_d=conv(x_r,delay_filter);
xrd_t=window.*xr_d(100:355);
plot([-0.5:1/256:0.5-
1/256],20*log10(abs(fftshift(fft(xrd_t)))));
title('X^''_{r}  dB Magnitude');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');

%%%xi(n)
figure(7);
xi_d=conv(x_r,hb);
xid_t=window.*xi_d(100:355);
plot([-0.5:1/256:0.5-
1/256],20*log10(abs(fftshift(fft(xid_t)))));
title('X_{i}  dB Magnitude ');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');


%%%x'r(n)cos
figure(8);
xrd_cos=xr_d(1:1024).*cos(2*pi*fc*n);
xrdcos_t=window.*xrd_cos(100:355);
plot([-0.5:1/256:0.5-
1/256],20*log10(abs(fftshift(fft(xrdcos_t)))));
title('X^''_{r}cos(2*pi*f_{c}*n)  dB Magnitude');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');

%%%xi(n)sin
figure(9);
xid_sin=xi_d(1:1024).*sin(2*pi*fc*n);
xidsin_t=window.*xid_sin(100:355);
plot([-0.5:1/256:0.5-
1/256],20*log10(abs(fftshift(fft(xidsin_t)))));
title('X_{i}sin(2*pi*f_{c}*n)  dB Magnitude');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');

%%%LSB
figure(10);
LSB=xrd_cos+xid_sin;
LSB_t=window.*LSB(100:355);
plot([-0.5:1/256:0.5-
1/256],20*log10(abs(fftshift(fft(LSB_t)))));
title('LSB dB Magnitude');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');

%%%USB
figure(11);
USB=xrd_cos-xid_sin;
```

```
USB_t=window.*USB(100:355);
plot([-0.5:1/256:0.5-
1/256],20*log10(abs(fftshift(fft(USB_t)))));
title('USB dB Magnitude');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');

%%%complex x_comp
figure(12);
x_comp=complex(xr_d,xi_d);
xcomp_t=window.*x_comp(100:355);
plot([-0.5:1/256:0.5-
1/256],20*log10(abs(fftshift(fft(xcomp_t)))));
title('X dB Magnitude');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');

%%%s(n)
figure(13);
s=x_comp(1:1024).*exp(i*2*pi*fc*n);
s_t=window.*s(100:355);
plot([-0.5:1/256:0.5-
1/256],20*log10(abs(fftshift(fft(s_t)))));
title('S dB Magnitude ');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');

%% Lowpass Filter
figure(14);
stem([0:NN-1],delay_filter);
title('Upper Path Lowpass Filter Impulse Response');
xlabel('n');ylabel('lp[n]');

figure(15);
LP=fftshift(fft(delay_filter,64));
plot([-0.5:1/64:0.5-1/64],20*log10(abs(LP)));
title('Upper Path Lowpass Filter Transfer Function dB
Magnitude ');
xlabel('f (cycle/sample)');ylabel('magnitude (dB)');

figure(16);
plot([-0.5:1/64:0.5-1/64],unwrap(phase(LP)));
title('Upper Path Lowpass Filter Transfer Function
Phase');
xlabel('f (cycle/sample)');ylabel('Phase (rad)');
```