CT303 Lab Exercise-1

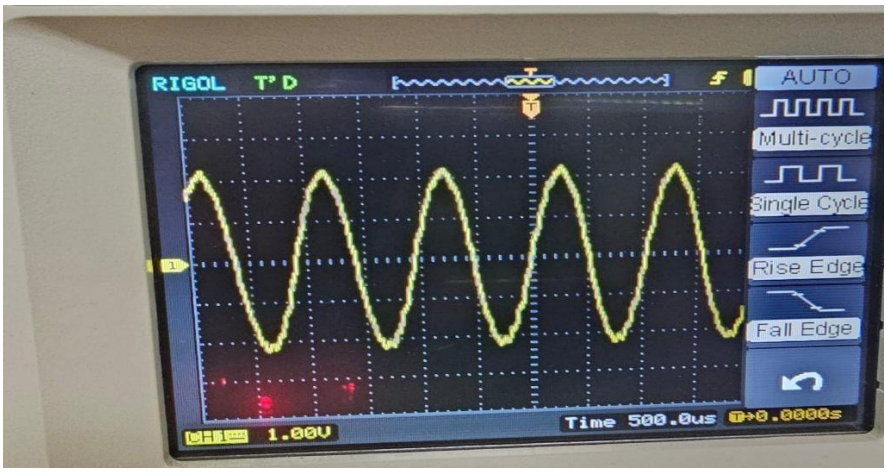
Group Members:-

Manan Ghonia	202301240
Archan Maru	202301217
Rishabh Jalu	202301265
Bhadarka Kanu	202301257

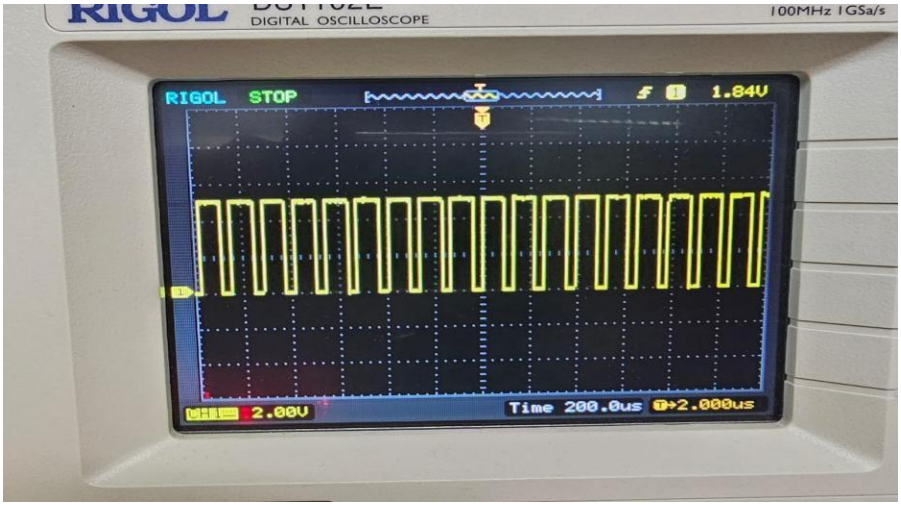
Hardware Experiment:

Analog Signal Sampling and Reconstruction

Input Signals:



Clock Signals:



1) Natural Sampling

Circuit Diagram

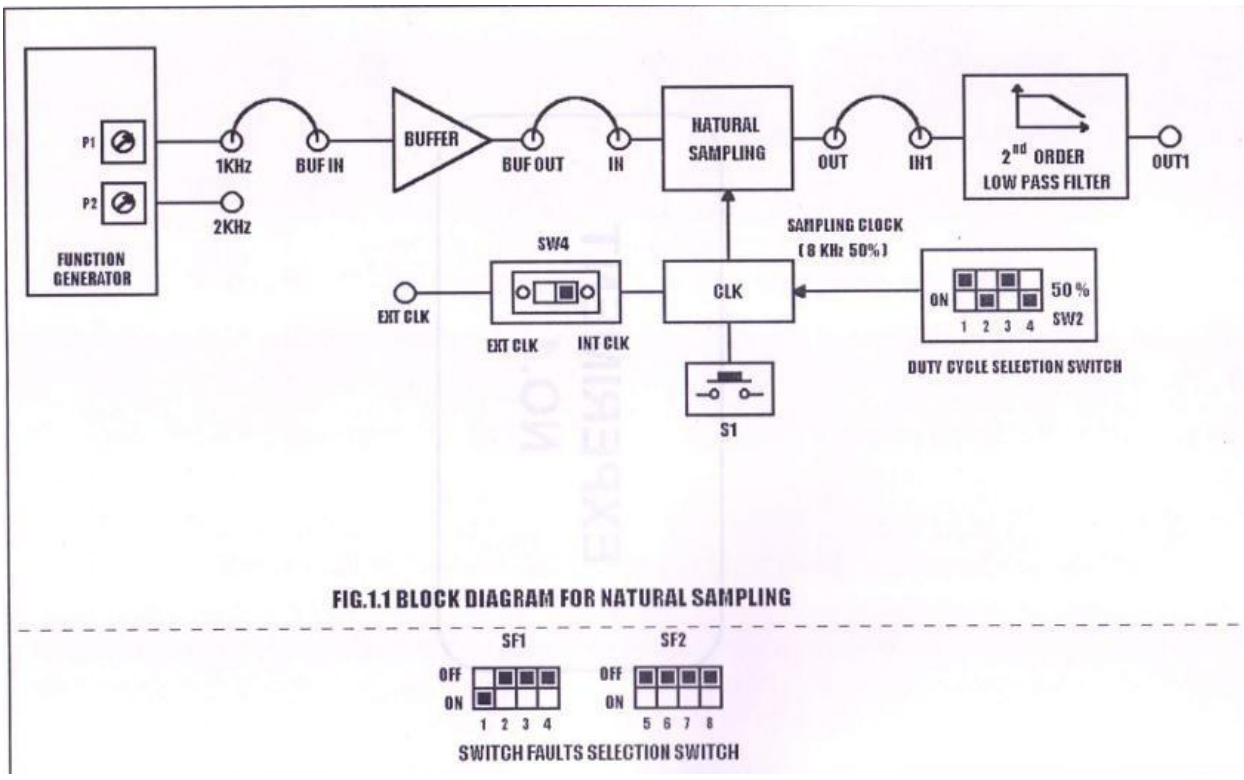
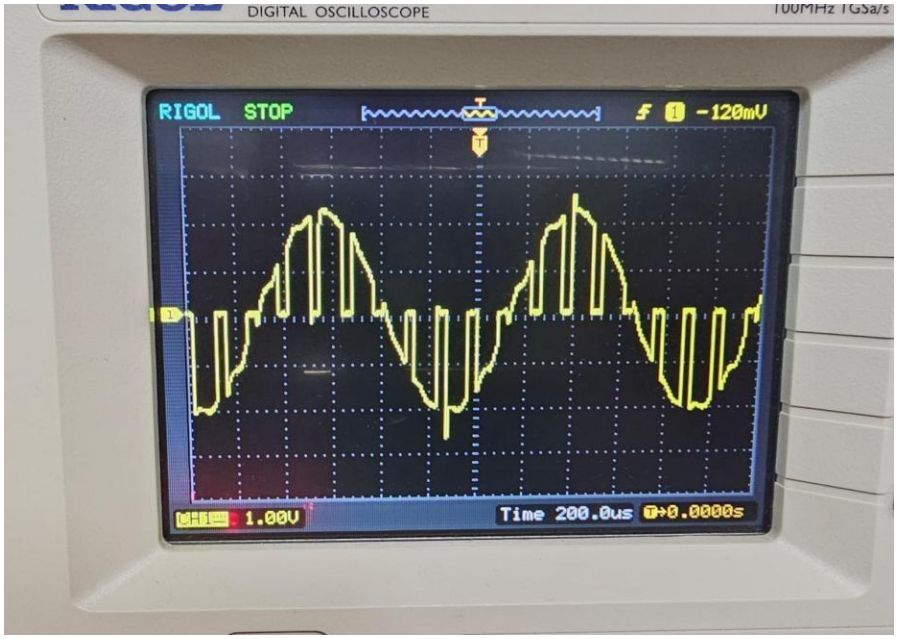


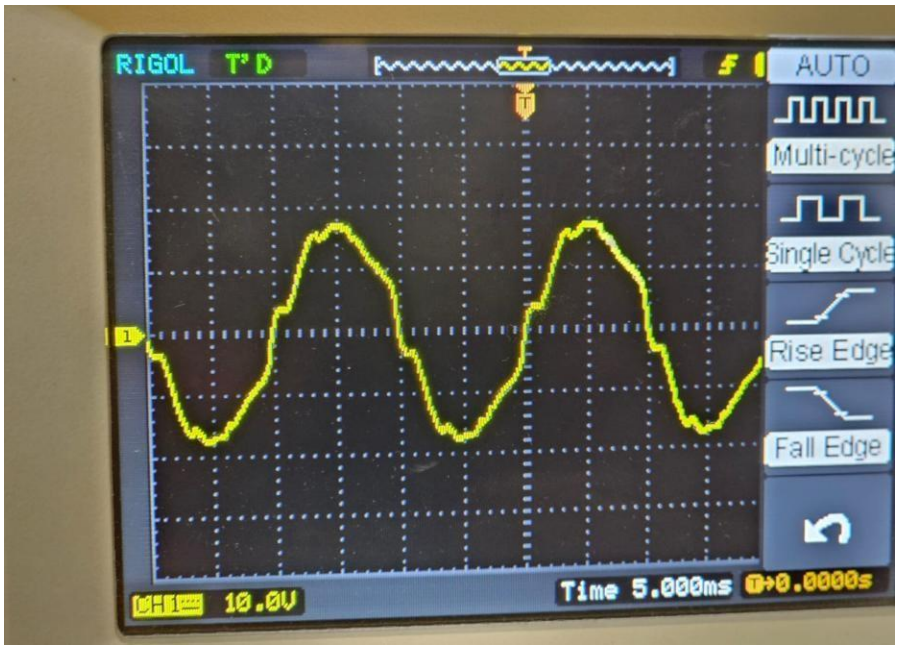
FIG.1.1 BLOCK DIAGRAM FOR NATURAL SAMPLING

Outputs:

1) Natural Sampling Output



2) Filter Output



1) Sample&Hold Sampling

Circuit Diagram

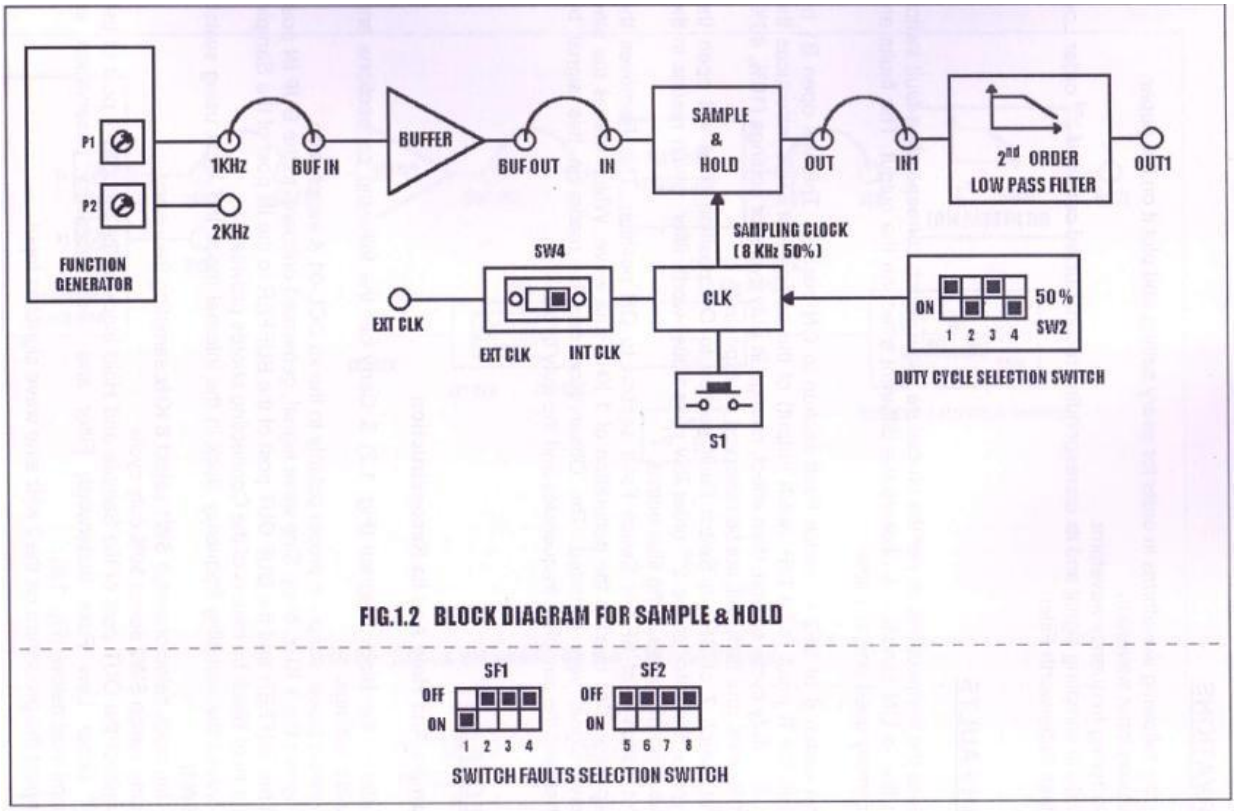
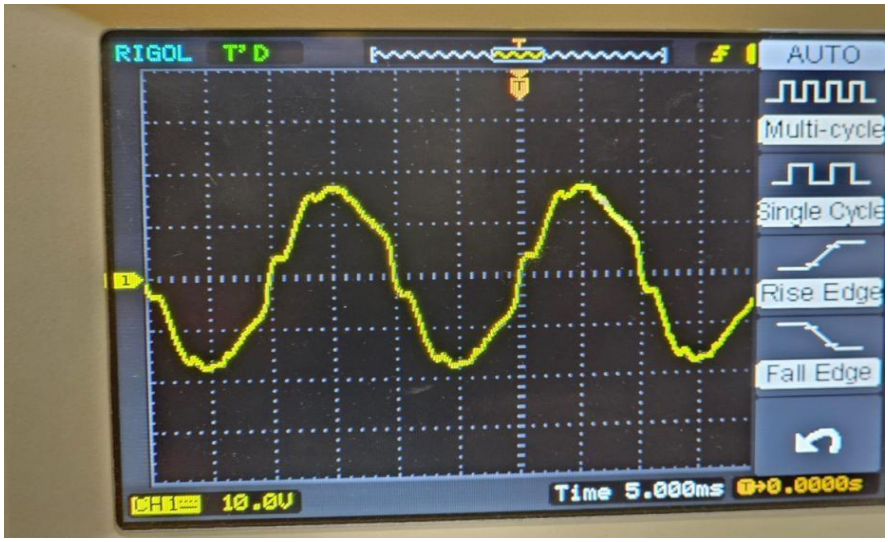


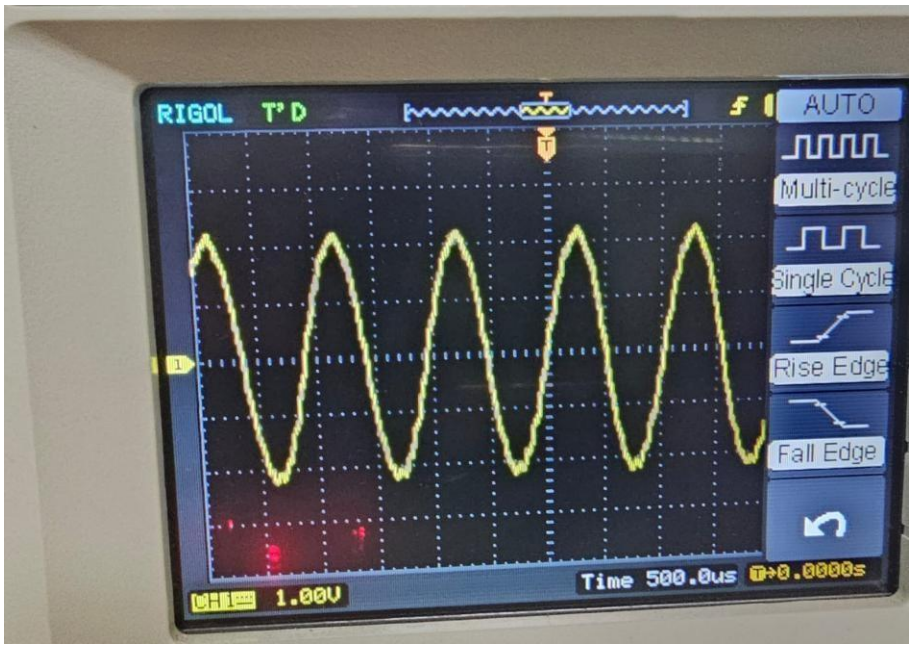
FIG.1.2 BLOCK DIAGRAM FOR SAMPLE & HOLD

Outputs:

1) SAMPLE_HOLD Output



2) Filter Output



3) Flat-Top Sampling

Circuit Diagram:

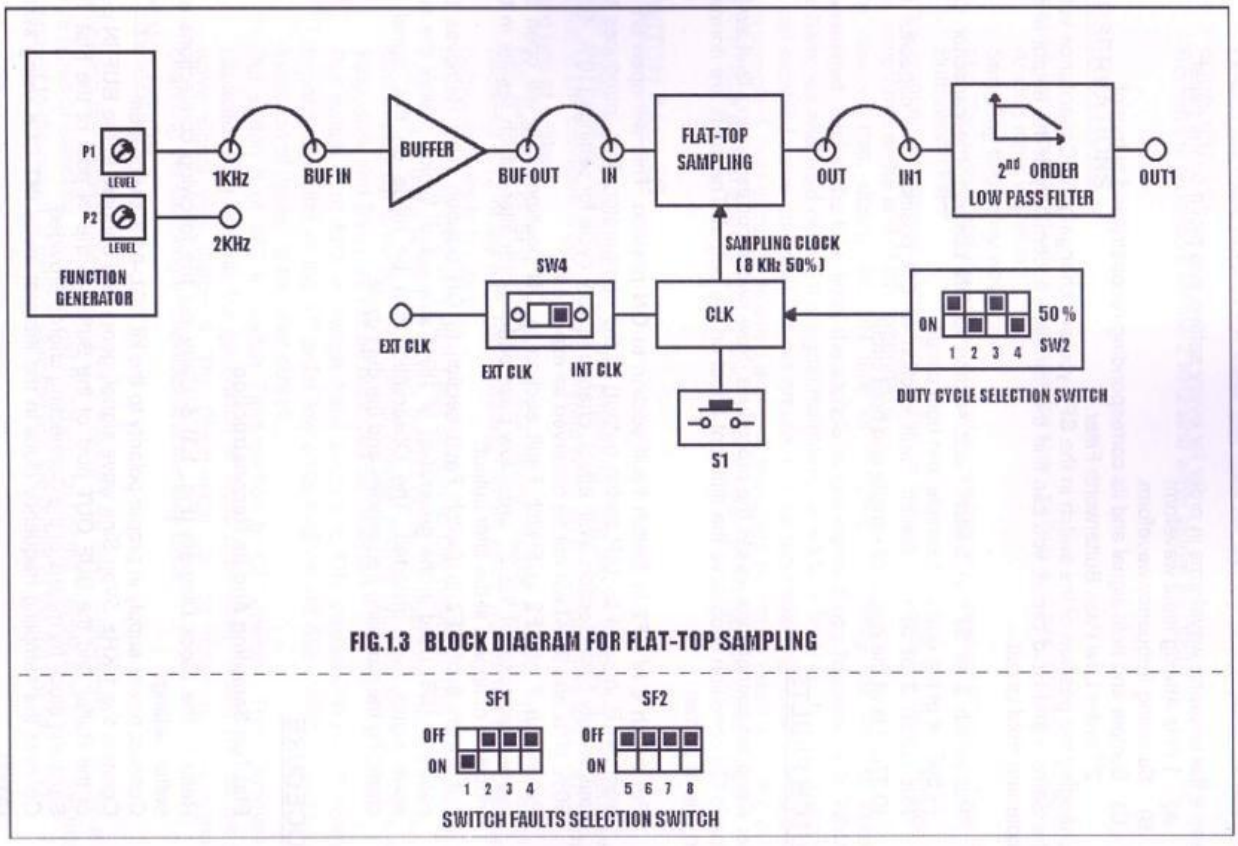
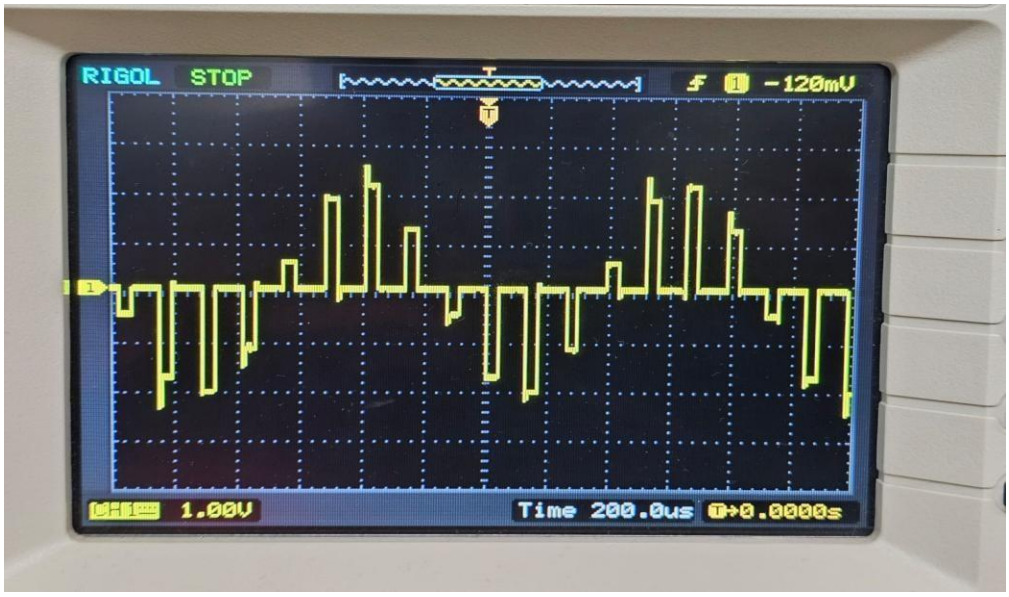


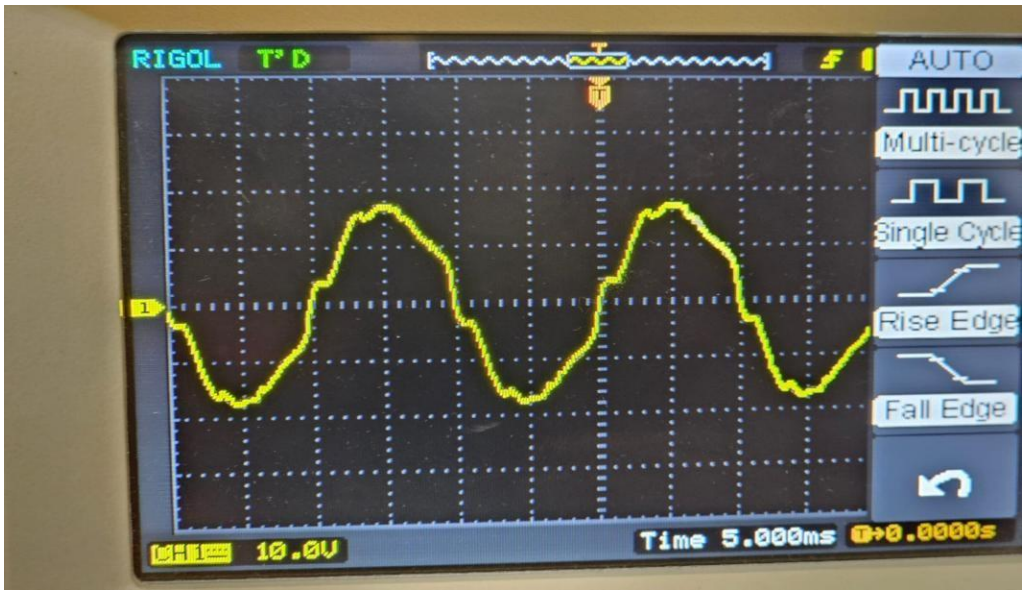
FIG.1.3 BLOCK DIAGRAM FOR FLAT-TOP SAMPLING

Outputs:

1) Flat-Top Sampling Output



2) Filter Output



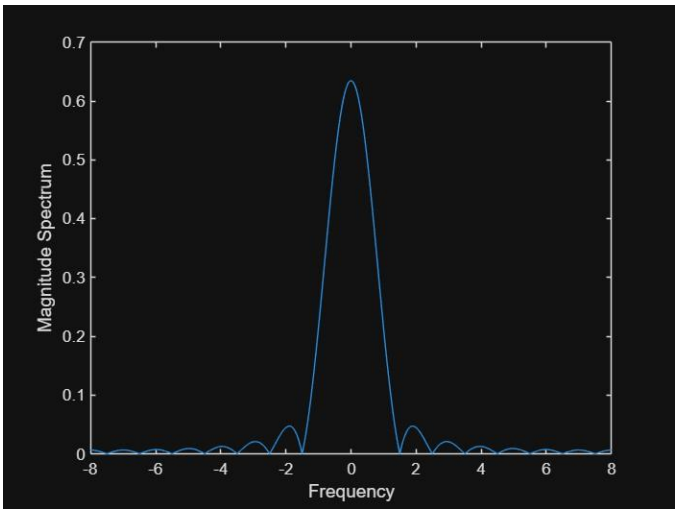
MATLAB CODE-

```
%code 2.5.1 from the book.


ts=1/16; %sampling interval
time_interval = 0:ts:1; %sampling time instants
%%time domain signal evaluated at sampling instants
signal_timedomain = sin(pi*time_interval); %sinusoidal pulse in our example
fs_desired = 1/160; %desired frequency granularity
Nmin = ceil(1/(fs_desired*ts)); %minimum length DFT for desired frequency
granularity
%for efficient computation, choose FFT size to be power of 2
Nfft = 2^(nextpow2(Nmin)) %FFT size = the next power of 2 at least as big as
Nmin
```

```
Nfft = 4096
```

```
%Alternatively, one could also use DFT size equal to the minimum length
%Nfft=Nmin;
%note: fft function in Matlab is just the DFT when Nfft is not a power of 2
%freq domain signal computed using DFT
%fft function of size Nfft automatically zeropads as needed
signal_freqdomain = ts*fft(signal_timedomain,Nfft);
%fftshift function shifts DC to center of spectrum
signal_freqdomain_centered = fftshift(signal_freqdomain);
fs=1/(Nfft*ts); %actual frequency resolution attained
%set of frequencies for which Fourier transform has been computed using DFT
freqs = ((1:Nfft)-1-Nfft/2)*fs;
%plot the magnitude spectrum
plot(freqs,abs(signal_freqdomain_centered));
xlabel('Frequency');
ylabel('Magnitude Spectrum');
```




```
%problem 4

function [X,f,df] = contFT(x,tstart,dt,df_desired)
    %Use Matlab DFT for approximate computation of continuous time Fourier
    %transform
    %INPUTS
    %x = vector of time domain samples, assumed uniformly spaced
    %tstart= time at which first sample is taken
    %dt = spacing between samples
    %df_desired = desired frequency resolution
    %OUTPUTS
    %X=vector of samples of Fourier transform
    %f=corresponding vector of frequencies at which samples are obtained
    %df=freq resolution attained (redundant--already available from
    %difference of consecutive entries of f)
    %%%%%%%%
    %minimum FFT size determined by desired freq res or length of x
    Nmin=max(ceil(1/(df_desired*dt)),length(x));
    %choose FFT size to be the next power of 2
    Nfft = 2^(nextpow2(Nmin));
    %compute Fourier transform, centering around DC
    X=dt*fftshift(fft(x,Nfft));
    %achieved frequency resolution
    df=1/(Nfft*dt);
    %range of frequencies covered
    f = ((0:Nfft-1)-Nfft/2)*df; %same as f=-1/(2*dt):df:1/(2*dt) - df
    %phase shift associated with start time
    X=X.*exp(-1j*2*pi*f*tstart);
end
```

```
Fs = 16;
dt = 1/Fs;
t = -8:dt:8;
tstart = t(1);

s = 3 * sinc(2*t - 3);

df_desired = 0.001;

[X, f, df] = contFT(s, tstart, dt, df_desired);

figure;
subplot(2,1,1);
plot(f, abs(X));
xlabel('Frequency (MHz)');
ylabel('|X(f)|');
title('Magnitude Spectrum');
grid on;

subplot(2,1,2);
plot(f, angle(X));
xlabel('Frequency (MHz)');
ylabel('Phase (radians)');
title('Phase of Fourier Transform');
grid on;
```