EXP -1

```
close all
clear all
clc
fs=8000;
fm=20;
fc=500;
Am=1;
Ac=1;
t=(0:0.1*fs)/fs;
m=Am*cos(2*pi*fm*t);
c=Ac*cos(2*pi*fc*t);
ka=0.5;
u=ka*Am;
s1=Ac*(1+u*cos(2*pi*fm*t)).*cos(2*pi*fc*t);
subplot(4,3,1:3);
plot(t,m);
title('Modulating signal(fm=20Hz)');
subplot(4,3,4:6);
plot(t,c);
title('Carrier signal(fc=500Hz)');
subplot(4,3,7);
plot(t,s1);
title('Under Modulated signal(ka.Am=0.5)');
ka1=1;
u1=ka1*Am;
s2=Ac*(1+u1*cos(2*pi*fm*t)).*cos(2*pi*fc*t);
subplot(4,3,8);
plot(t,s2);
title('Exact Modulated signal(ka.Am=1)');
ka2=2;
u2=ka2*Am;
s3=Ac*(1+u2*cos(2*pi*fm*t)).*cos(2*pi*fc*t);
subplot(4,3,9);
plot(t,s3);
title('Over Modulated signal(ka2.Am=2)');
r1 = s1.*c;
r2 = s2.*c;
r3 = s3.*c;
[b, a] = butter(1,0.01);
mr1 = filter(b,a,r1);
mr2 = filter(b,a,r2);
mr3 = filter(b,a,r3);
subplot(4,3,10);
plot(t,mr1);
title('Demodulated signal(u)')
subplot(4,3,11);
plot(t,mr2);
title('Demodulated signal(E)')
subplot(4,3,12);
plot(t,mr3);
title('Demodulated signal(O)')
```

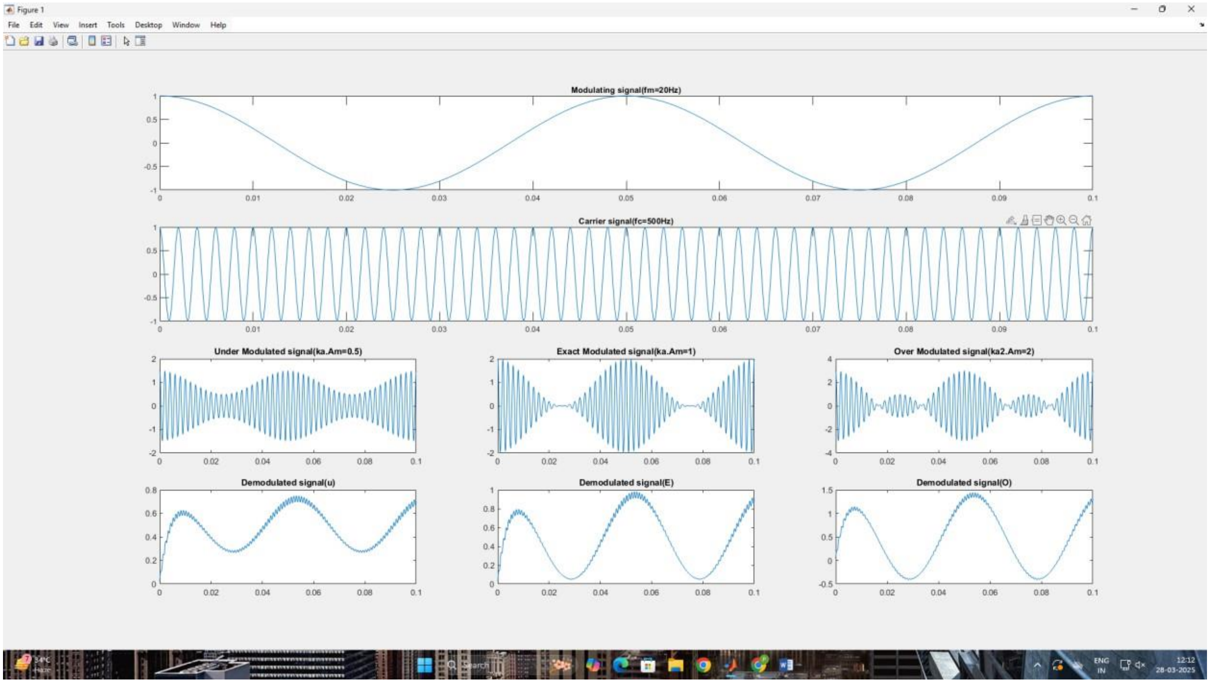


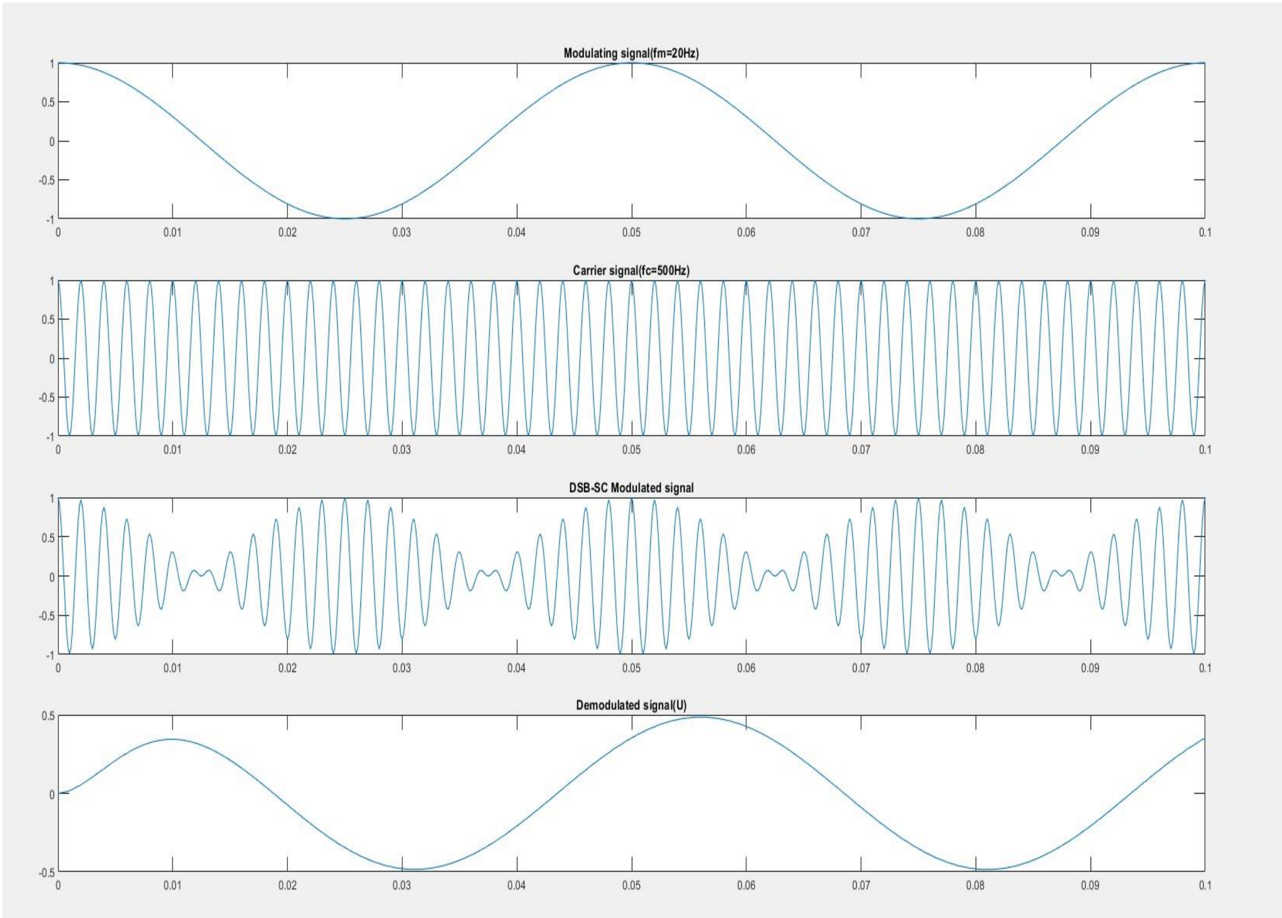
Figure 1
File Edit View Insert Tools Desktop Window Help
Modulating signal(fm=20Hz)
Carrier signal(fc=500Hz)
Under Modulated signal(ka.Am=0.5)
Exact Modulated signal(ka.Am=1)
Over Modulated signal(ka2.Am=2)
Demodulated signal(u)
Demodulated signal(E)
Demodulated signal(O)

Exp-2 DSB-SC

Code:

```
close all
clear all
clc
fs=8000;
fm=20;
fc=500;
Am=1;
Ac=1;
t=(0:0.1*fs)/fs;
m=Am*cos(2*pi*fm*t);
c=Ac*cos(2*pi*fc*t);
s1=m.*c;
subplot(4,3,1:3);
plot(t,m);
title('Modulating signal(fm=20Hz)');
subplot(4,3,4:6);
plot(t,c);
title('Carrier signal(fc=500Hz)');
subplot(4,3,7:9);
plot(t,s1);
title('DSB-SC Modulated signal');
r1=s1.*c;
[b,a]=butter(2,0.01);
mr1=filter(b,a,r1);
subplot(4,3,10:12);
plot(t,mr1);
title('Demodulated signal(U)')
```

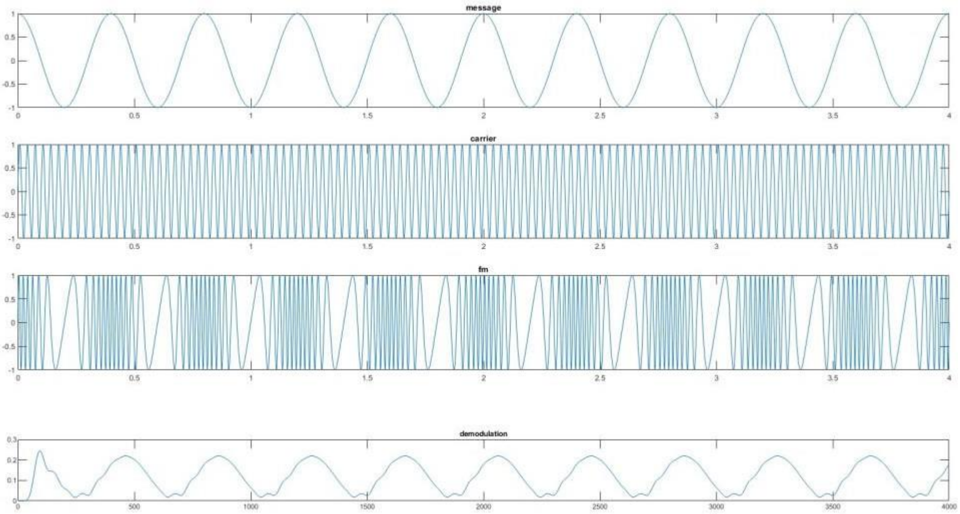
O/P


Modulating signal(fm=20Hz)
Carrier signal(fc=500Hz)
DSB-SC Modulated signal
Demodulated signal(U)

Exp -3 FREQUENCY MODULATION CODE:

```
clc;
clear all;
close all;
t=[0:0.001:4];
f1=2.5;
m=cos(2*pi*f1*t);
subplot(4,2,[1,2]);
plot(t,m);
title('message');
f2=30;
c=sin(2*pi*f2*t);
subplot(4,2,[3,4]);
plot(t,c);
title('carrier');
mf=10;
s=sin((2*pi*f2*t)+(mf*sin(2*pi*f1*t)));
subplot(4,2,[5,6]);
plot(t,s);
title('fm');
syms t1;
x=diff(s);
y=abs(x);
[b,a]=butter(10,0.033);
s1=filter(b,a,y);
subplot(6,2,[11,12]);
plot(s1);
title('demodulation');
```

OUTPUT:

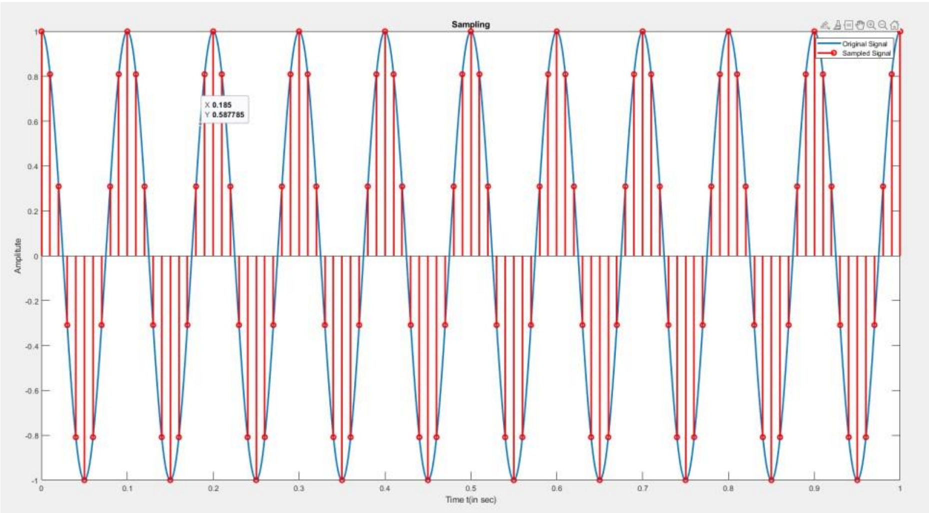


Exp-4

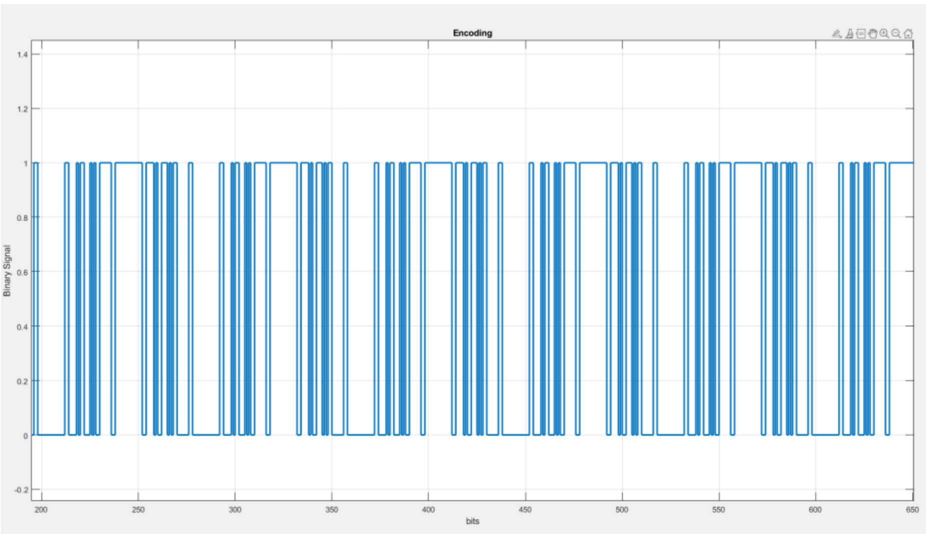
```
clc;

clear all;

close all;

A=1;

fm=10;

fs=100;

n= 8

t=0:1/(100*fm):1;

x=A*cos(2*pi*fm*t);

%---Sampling-----

ts=0:1/fs:1;

xs=A*cos(2*pi*fm*ts);

%xs Sampled signal


%--Quantization---

x1=xs+A;

x1=x1/(2*A);

L=(-1+2^n); % Levels

x1=L*x1;

xq=round(x1);

r=xq/L;

r=2*A*r;

r=r-A;

%r quantized signal


%----Encoding---

y=[];

for i=1:length(xq)
```

```
    d=dec2bin(xq(i),n);
    y=[y double(d)-48];
end
%Calculations
MSE=sum((xs-r).^2)/length(x);
Bitrate=n*fs;
Stepsize=2*A/L;
QNoise=((Stepsize)^2)/12;
figure(1)
plot(t,x,'linewidth',2)
title('Sampling')
ylabel('Amplitute')
xlabel('Time t(in sec)')
hold on
stem(ts,xs,'r','linewidth',2)
hold off
legend('Original Signal','Sampled Signal');
figure(2)
stem(ts,x1,'linewidth',2)
title('Quantization')
ylabel('Levels L')
hold on
stem(ts,xq,'r','linewidth',2)
plot(ts,xq,'--r')
plot(t,(x+A)*L/(2*A),'--b')
grid
hold off
legend('Sampled Signal','Quantized Signal');
```

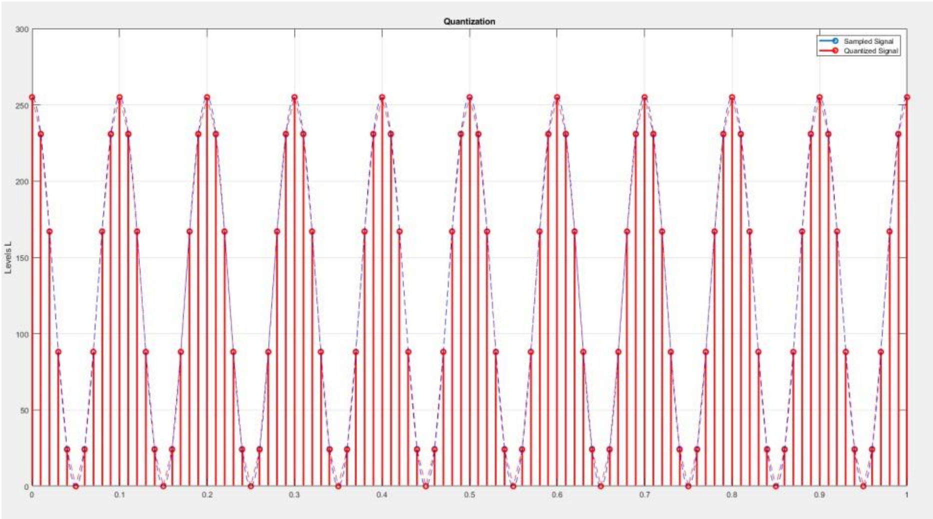
```
figure(3)

stairs([y y(length(y))],'linewidth',2)

title('Encoding')

ylabel('Binary Signal')

xlabel('bits')

axis([0 length(y) -1 2])

grid
```



Sampling



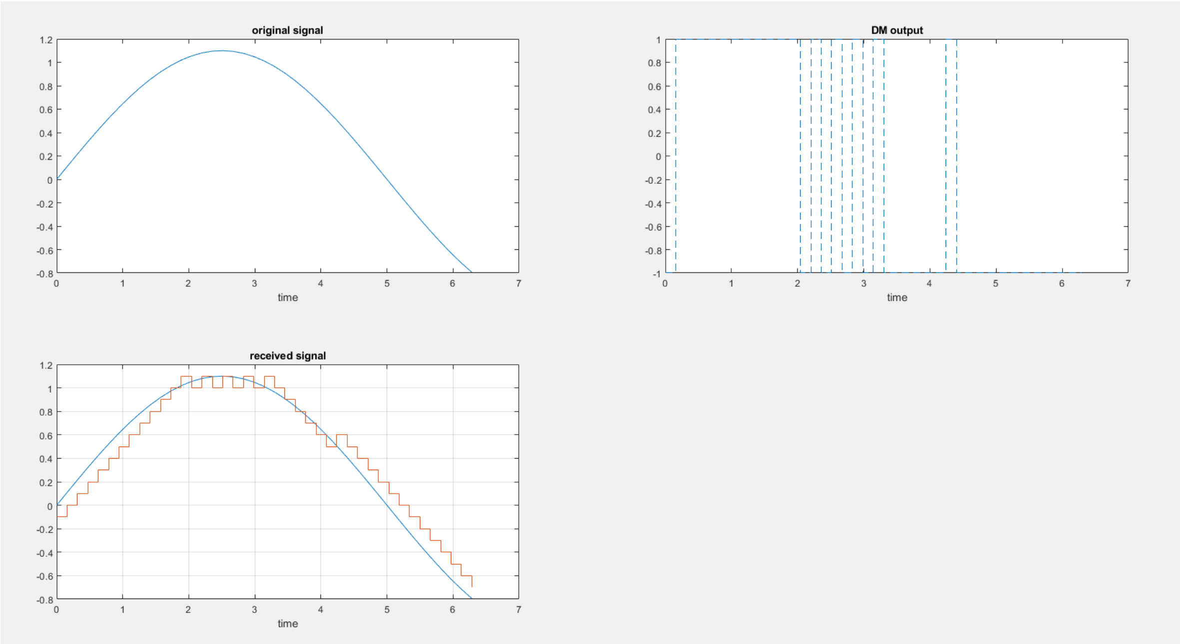
Encoding



Quantization

Exp-5 Delta Modulation

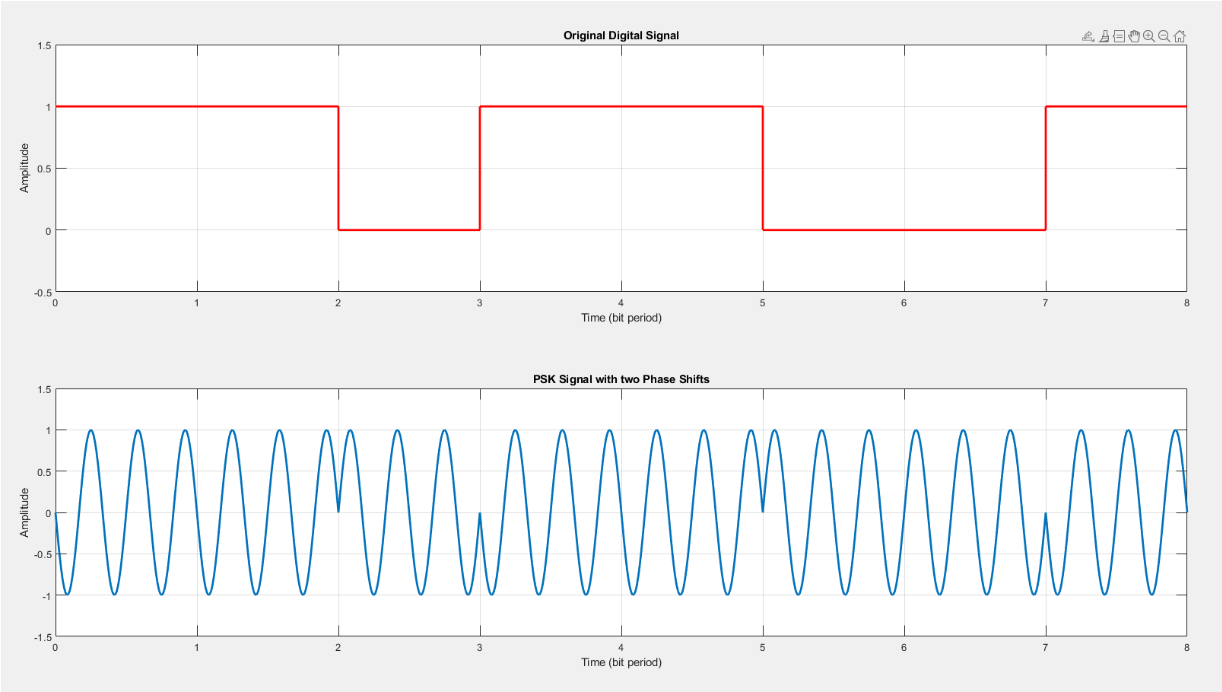
```
clc;

clear all;

close all;

%delta modulation = 1-bit differential pulse code modulation (DPCM)

predictor = [0 1]; % y(k)=x(k-1)

%partition = [-1:.1:.9];codebook = [-1:.1:1];

step=0.1; %SFs>=2pifA

partition = [0];codebook = [-1*step step]; %DM quantizer


t = [0:pi/20:2*pi];

x = 1.1*sin(2*pi*0.1*t); % Original signal, a sine wave

%t = [0:0.1:2*pi];x = 4*sin(t);

%x=exp(-1/3*t);

%x = sawtooth(3*t); % Original signal


% Quantize x(t) using DPCM.

encodedx = dpcmenco(x,codebook,partition,predictor);


% Try to recover x from the modulated signal.

decodedx = dpcmdeco(encodedx,codebook,predictor);

distor = sum((x-decodedx).^2)/length(x) % Mean square error


% plots

figure,subplot(2,2,1);plot(t,x);xlabel('time');title('original signal');

subplot(2,2,2);stairs(t,10*codebook(encodedx+1),'--');xlabel('time');title('DM output');

subplot(2,2,3);plot(t,x);hold;stairs(t,decodedx);grid;xlabel('time');title('received signal');
```


original signal
time
DM output
time
received signal
time

Exp -6 PSKM

```
% MATLAB Script for a Binary PSK with two Phases

format long;

% Clear all variables and close all figures

clear all;

close all;

% The number of bits to send - Frame Length

N = 8;

% Generate a random bit stream

bit_stream = round(rand(1,N));

% Enter the two Phase shifts - in Radians

% Phase for 0 bit

P1 = 0;

% Phase for 1 bit

P2 = pi;

% Frequency of Modulating Signal

f = 3;

% Sampling rate - This will define the resoultion

fs = 100;

% Time for one bit

t = 0: 1/fs : 1;

% This time variable is just for plot

time = [];

PSK_signal = [];

Digital_signal = [];

for ii = 1: 1: length(bit_stream)


  % The FSK Signal

  PSK_signal = [PSK_signal (bit_stream(ii)==0)*sin(2*pi*f*t + P1)+...
```

```
     (bit_stream(ii)==1)*sin(2*pi*f*t + P2)];


   % The Original Digital Signal
   Digital_signal = [Digital_signal (bit_stream(ii)==0)*...
     zeros(1,length(t)) + (bit_stream(ii)==1)*ones(1,length(t))];


  time = [time t];
   t =  t + 1;
  end
% Plot the PSK Signal
subplot(2,1,2);
plot(time,PSK_signal,'LineWidth',2);
xlabel('Time (bit period)');
ylabel('Amplitude');
title('PSK Signal with two Phase Shifts');
axis([0 time(end) -1.5 1.5]);
grid  on;
% Plot the Original Digital Signal
subplot(2,1,1);
plot(time,Digital_signal,'r','LineWidth',2);
xlabel('Time (bit period)');
ylabel('Amplitude');
title('Original Digital Signal');
axis([0 time(end) -0.5 1.5]);
grid on;
```

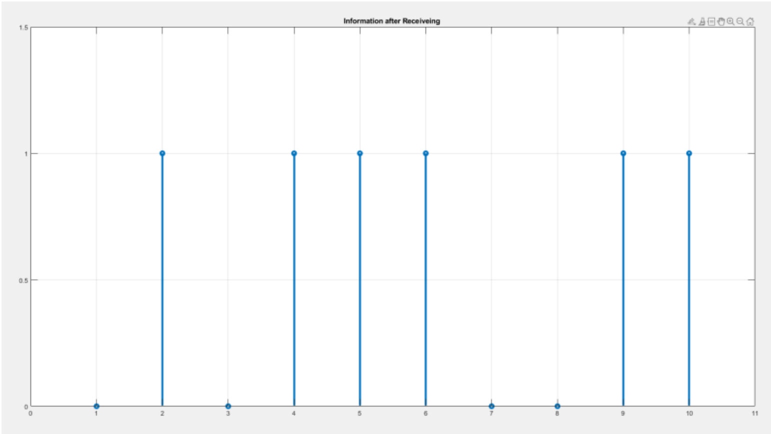

Original Digital Signal
Amplitude
Time (bit period)
PSK Signal with two Phase Shifts
Amplitude
Time (bit period)

Exp-7 QPSK

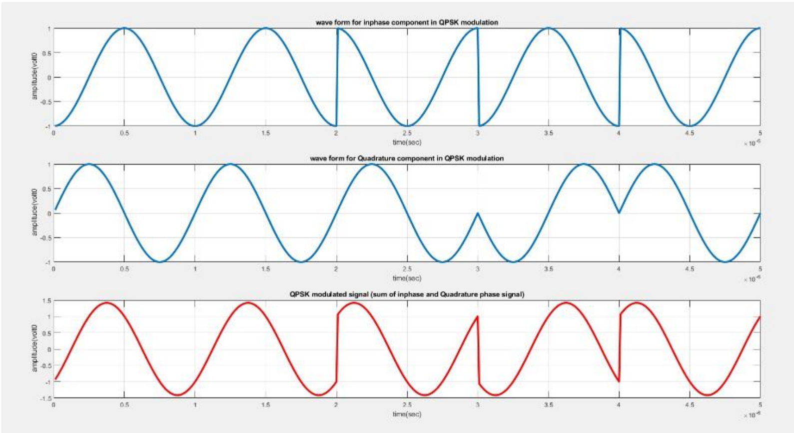
```
%XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

%XXXX QPSK Modulation and Demodulation without consideration of noise XXXXX

%XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

clc;

clear all;

close all;

data=[0  1 0 1 1 1 0 0 1 1]; % information

%Number_of_bit=1024;

%data=randint(Number_of_bit,1);

figure(1)

stem(data, 'linewidth',3), grid on;

title('  Information before Transmiting ');

axis([ 0 11 0 1.5]);

data_NZR=2*data-1; % Data Represented at NZR form for QPSK modulation

s_p_data=reshape(data_NZR,2,length(data)/2);  % S/P convertion of data

br=10.^6; %Let us transmission bit rate  1000000

f=br; % minimum carrier frequency

T=1/br; % bit duration

t=T/99:T/99:T; % Time vector for one bit information

% XXXXXXXXXXXXXXXXXXXXXXX QPSK modulatio  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

y=[];

y_in=[];

y_qd=[];

for(i=1:length(data)/2)

    y1=s_p_data(1,i)*cos(2*pi*f*t); % inphase component

    y2=s_p_data(2,i)*sin(2*pi*f*t) ;% Quadrature component

    y_in=[y_in y1]; % inphase signal vector

    y_qd=[y_qd y2]; %quadrature signal vector
```

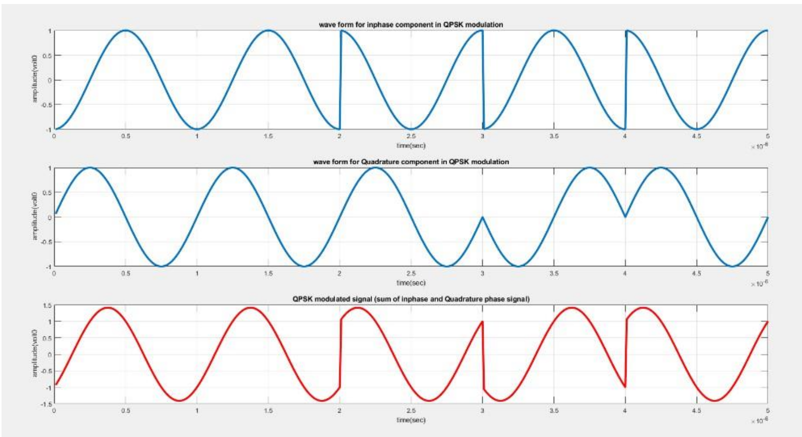
```
    y=[y y1+y2]; % modulated signal vector
end
Tx_sig=y; % transmitting signal after modulation
tt=T/99:T/99:(T*length(data))/2;
figure(2)
subplot(3,1,1);
plot(tt,y_in,'linewidth',3), grid on;
title(' wave form for inphase component in QPSK modulation ');
xlabel('time(sec)');
ylabel(' amplitude(volt0');
subplot(3,1,2);
plot(tt,y_qd,'linewidth',3), grid on;
title(' wave form for Quadrature component in QPSK modulation ');
xlabel('time(sec)');
ylabel(' amplitude(volt0');
subplot(3,1,3);
plot(tt,Tx_sig,'r','linewidth',3), grid on;
title('QPSK modulated signal (sum of inphase and Quadrature phase signal)');
xlabel('time(sec)');
ylabel(' amplitude(volt0');
% XXXXXXXXXXXXXXXXXXXXXXXXXXXXX QPSK demodulation XXXXXXXXXXXXXXXXXXXXXXXXXXX
Rx_data=[];
Rx_sig=Tx_sig; % Received signal
for(i=1:1:length(data)/2)
    %%XXXXXX inphase coherent dector XXXXXXX
    Z_in=Rx_sig((i-1)*length(t)+1:i*length(t)).*cos(2*pi*f*t);
    % above line indicat multiplication of received & inphase carred signal
```

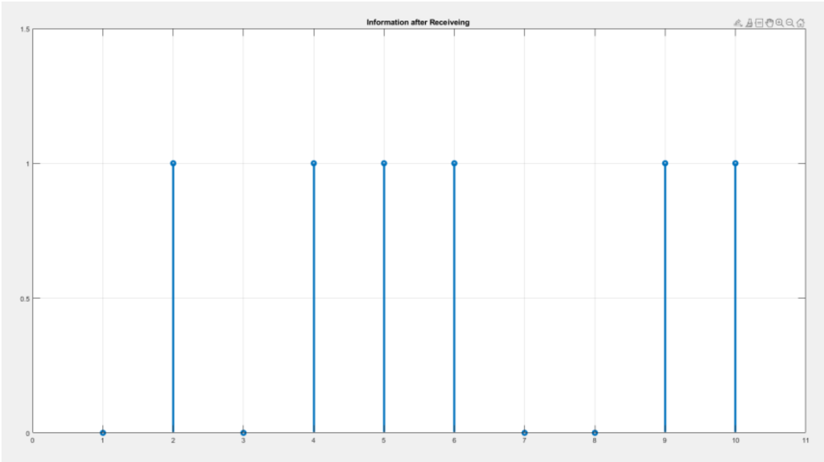
```
   Z_in_intg=(trapz(t,Z_in))*(2/T);% integration using trapizodial rull
   if(Z_in_intg>0) % Decession Maker
      Rx_in_data=1;
   else
     Rx_in_data=0;
   end

   %%XXXXXX Quadrature coherent dector XXXXXX
   Z_qd=Rx_sig((i-1)*length(t)+1:i*length(t)).*sin(2*pi*f*t);
   %above line indicat multiplication ofreceived & Quadphase carred signal

   Z_qd_intg=(trapz(t,Z_qd))*(2/T);%integration using trapizodial rull
      if (Z_qd_intg>0)% Decession Maker
      Rx_qd_data=1;
     else
     Rx_qd_data=0;
     end

      Rx_data=[Rx_data  Rx_in_data  Rx_qd_data]; % Received Data vector
end
figure(3)
stem(Rx_data,'linewidth',3)
title('Information after Receiveing ');
axis([ 0 11 0 1.5]), grid on;
% XXXXXXXXXXXXXXXXXXXXXXXXX    end of program     XXXXXXXXXXXXXXXXXXXXXXXXXX
```



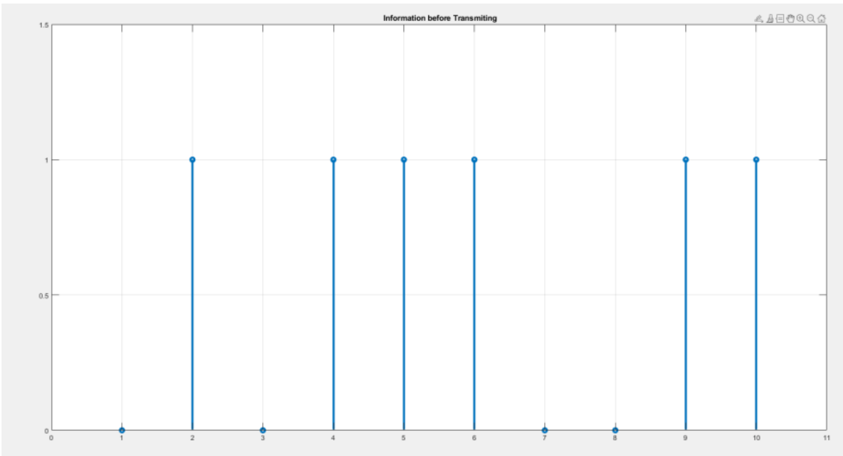
Information of Receiving







Information after Receiving



Information Before transmitting