EXPERIMENT 1
Amplitude Shift Keying

AIM:-

To plot the wave form for Binary Amplitude Shift Keying (BASK) signal using MATLAB for a stream of bits.

THEORY:-

Amplitude Shift Keying (ASK) is the digital modulation technique. In amplitude shift keying, the amplitude of the carrier signal is varied to create signal elements. Both frequency and phase remain constant while the amplitude changes. In ASK, the amplitude of the carrier assumes one of the two amplitudes dependent on the logic states of the input bit stream. This modulated signal can be expressed as:

$$x_c(t) = \begin{cases} 0 & \text{symbol "0"} \\ A\cos\omega_c t & \text{symbol "1"} \end{cases}$$

Amplitude shift keying (ASK) in the context of digital signal communications is a modulation process, which imparts to a sinusoid two or more discrete amplitude levels. These are related to the number of levels adopted by the digital message. For a binary message sequence there are two levels, one of which is typically zero. Thus the modulated waveform consists of bursts of a sinusoid. Figure 1 illustrates a binary ASK signal (lower), together with the binary sequence which initiated it (upper). Neither signal has been band limited.

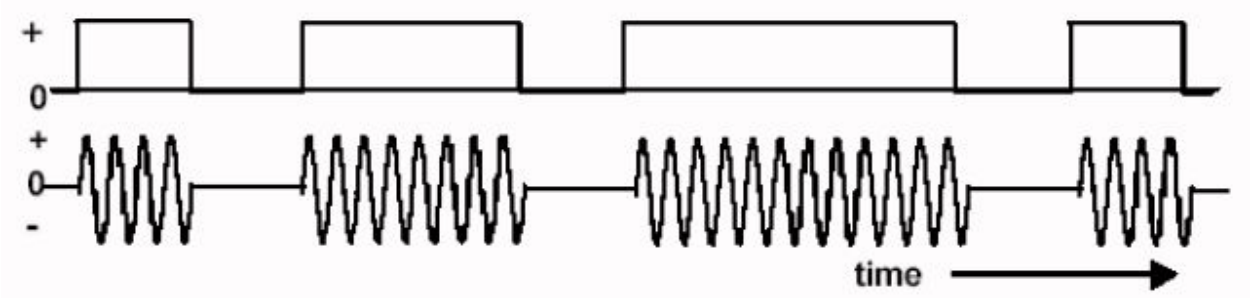


Fig: an ASK signal (below) and the message (above)

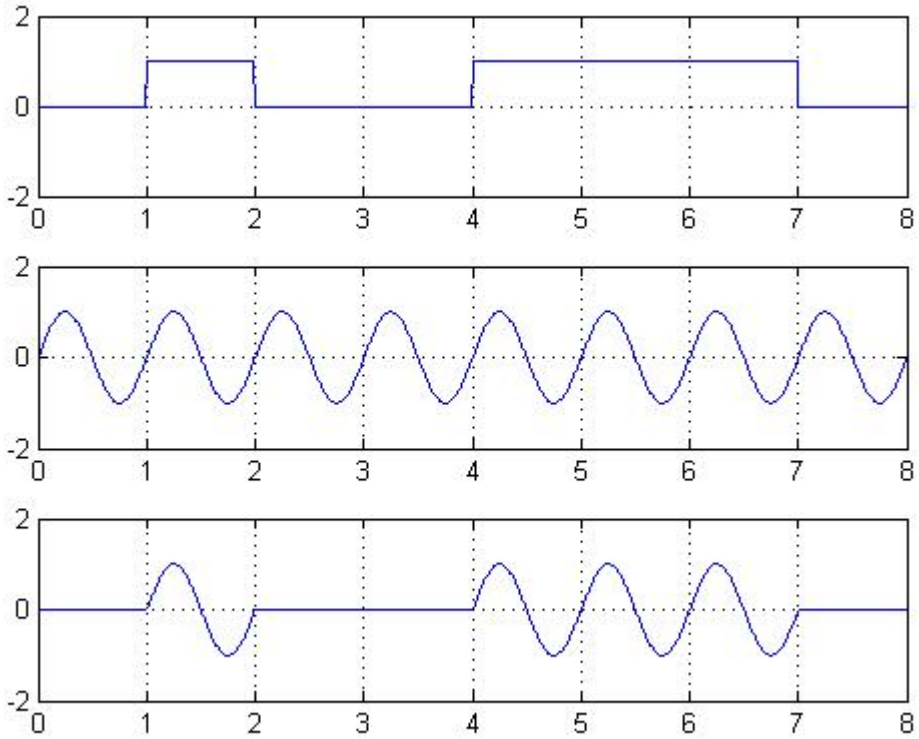
There are sharp discontinuities shown at the transition points. These result in the signal having an unnecessarily wide bandwidth. Band limiting is generally introduced before transmission, in which case these discontinuities would be 'rounded off'. The band limiting may be applied to the digital message, or the modulated signal itself. The data rate is often made a sub-multiple of the carrier frequency. This has been done in the waveform of Fig.

MATLAB PROGRAM:-

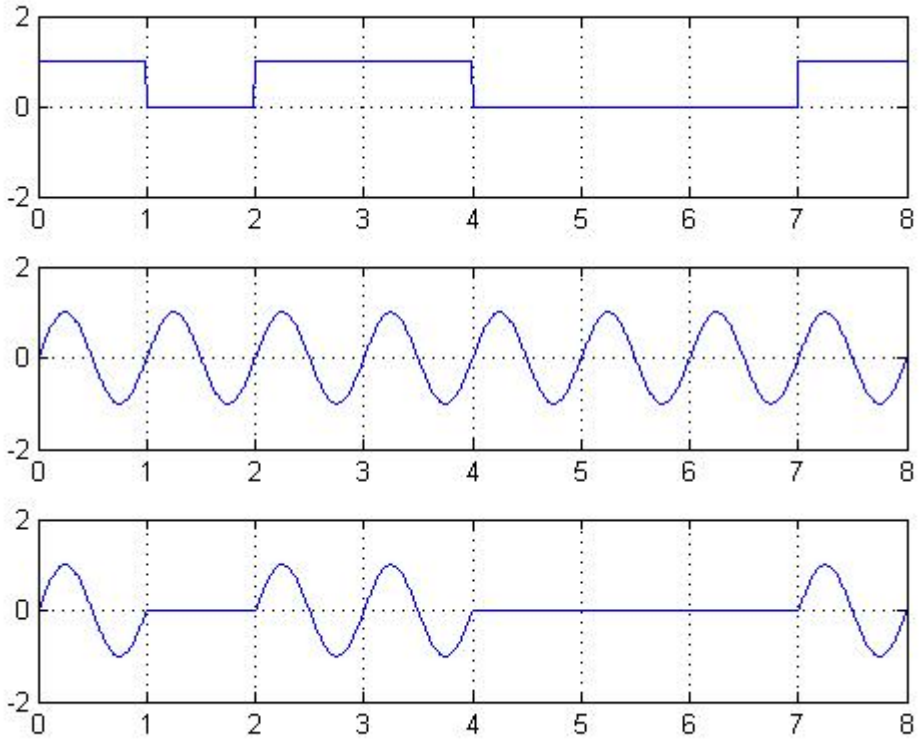
```
clear;
clc;
b = input('Enter the Bit stream \n ');   %b = [0 1 0 1 1 1 0];
n = length(b);
t = 0:.01:n;
x = 1:1:(n+1)*100;
for i = 1:n
   for j = i:.1:i+1
      bw(x(i*100:(i+1)*100)) = b(i);
   end
end
bw = bw(100:end);
sint = sin(2*pi*t);
st = bw.*sint;
subplot(3,1,1)
plot(t,bw)
grid on ; axis([0 n -2 +2])
subplot(3,1,2)
plot(t,sint)
grid on ; axis([0 n -2 +2])
subplot(3,1,3)
plot(t,st)
grid on ; axis([0 n -2 +2])
```

OBSERVATION:-

Output waveform for the bit stream [0 1 0 0 1 1 1 0]



Output waveform for the bit stream [1 0 1 1 0 0 0 1]



EXPERIMENT 2

Frequency Shift Keying

AIM:-

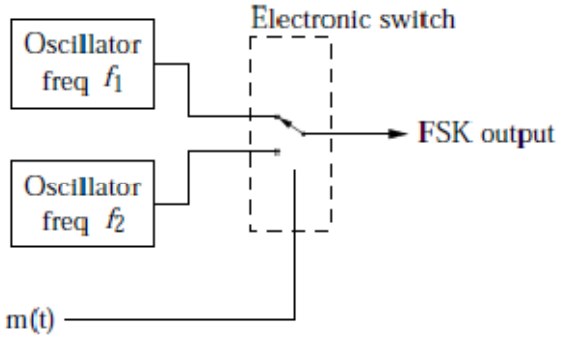
To plot the wave form for Binary Frequency Shift Keying (BFSK) signal using MATLAB for a stream of bits.

THEORY:-

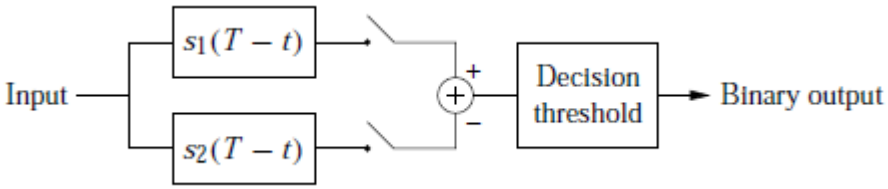
In frequency-shift keying, the signals transmitted for marks (binary ones) and spaces (binary zeros) are respectively.

$$s_1(t) = A\cos(\omega_1 t + \theta_c), \qquad 0 < t \leq T$$
$$s_2(t) = A\cos(\omega_2 t + \theta_c), \qquad 0 < t \leq T$$

This is called a **discontinuous phase** FSK system, because the phase of the signal is discontinuous at the switching times. A signal of this form can be generated by the following system.



If the bit intervals and the phases of the signals can be determined (usually by the use of a phase-lock loop), then the signal can be decoded by two separate matched filters:



The first filter is matched to the signal S1(t)and the second to S2(t) Under the assumption that the signals are mutually orthogonal, the output of one of the matched filters will be *E* and the other zero (where *E* is the energy of the signal). Decoding of the bandpass signal can therefore be achieved by subtracting the outputs of the two filters, and comparing the result to a threshold. If the signal S1(t) is present then the resulting output will be +E, and if S2(t) is present it will be –E. Since the noise variance at each filter output is *E*n/2, the noise in the difference signal will be doubled, namely □^{2}=En. Since the overall output variation is 2*E*, the probability of error is:

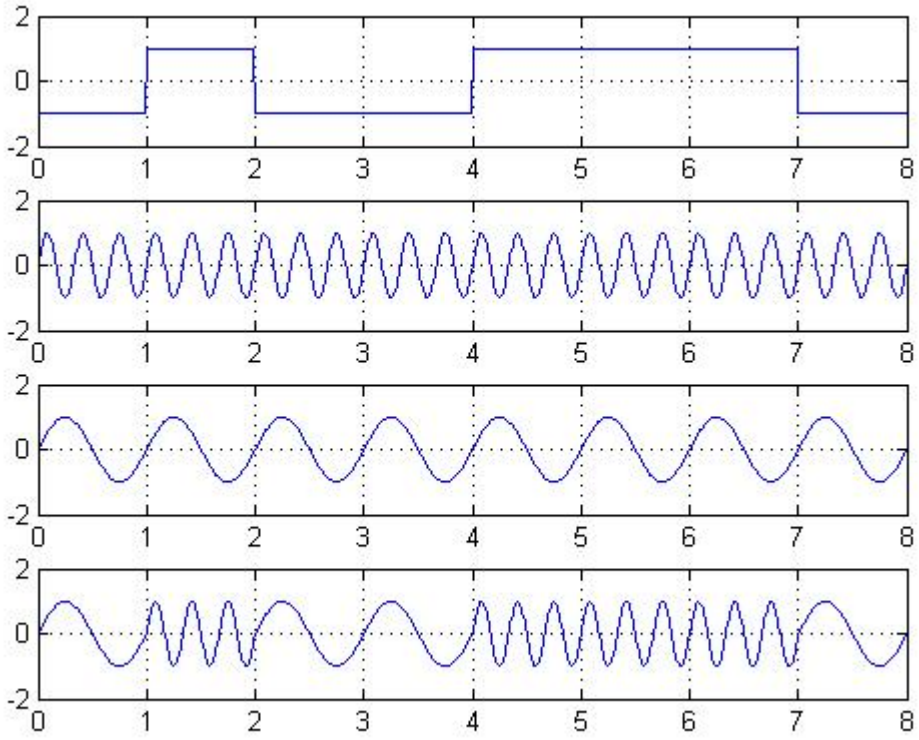
$$P_\epsilon = \text{erfc}\left(\frac{2E}{2\sqrt{E\eta}}\right) = \text{erfc}\sqrt{\frac{E}{\eta}}.$$

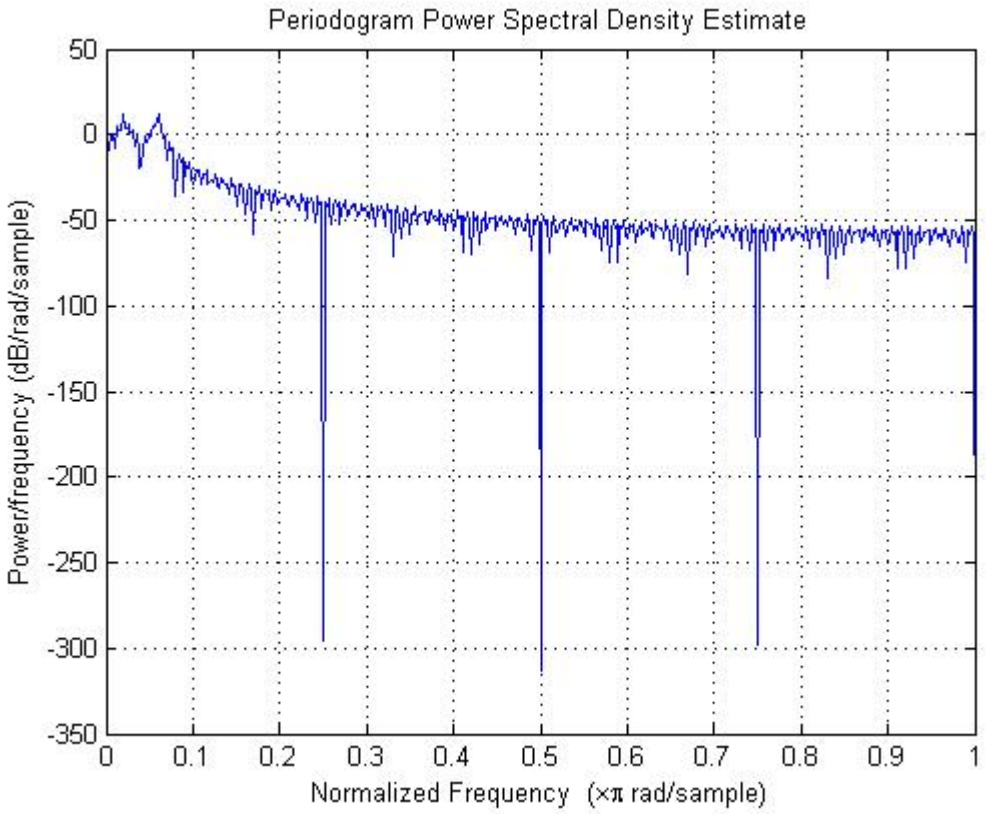
MATLAB PROGRAM:-

```
clear;
clc;
b = input('Enter the Bit stream \n ');   %b = [0 1 0 1 1 1 0];
n = length(b);
t = 0:.01:n;
x = 1:1:(n+1)*100;
for i = 1:n
   if (b(i) == 0)
      b_p(i) = -1;
   else
      b_p(i) = 1;
   end
   for j = i:.1:i+1
      bw(x(i*100:(i+1)*100)) = b_p(i);
   end
end
bw = bw(100:end);
wo = 2*(2*pi*t);
W  = 1*(2*pi*t);
sinHt = sin(wo+W);
sinLt = sin(wo-W);
st = sin(wo+(bw).*W);
subplot(4,1,1)
plot(t,bw)
grid on ; axis([0 n -2 +2])
subplot(4,1,2)
plot(t,sinHt)
grid on ; axis([0 n -2 +2])
subplot(4,1,3)
plot(t,sinLt)
grid on ; axis([0 n -2 +2])
subplot(4,1,4)
plot(t,st)
grid on ; axis([0 n -2 +2])
Fs=1;
figure  %pburg(st,10)
periodogram(st)
```

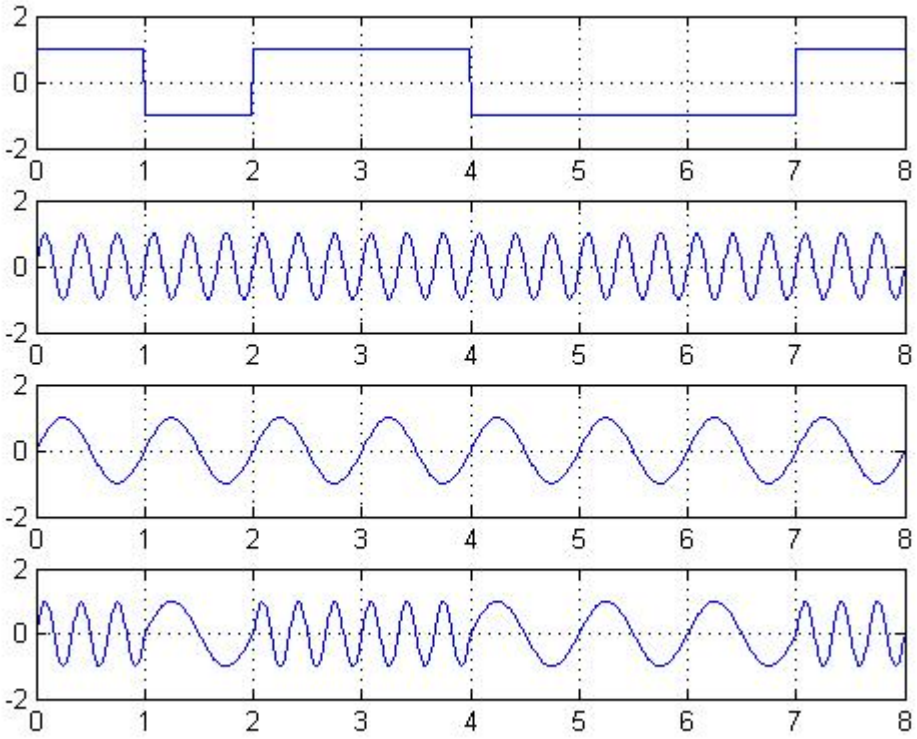
OBSERVATION:-

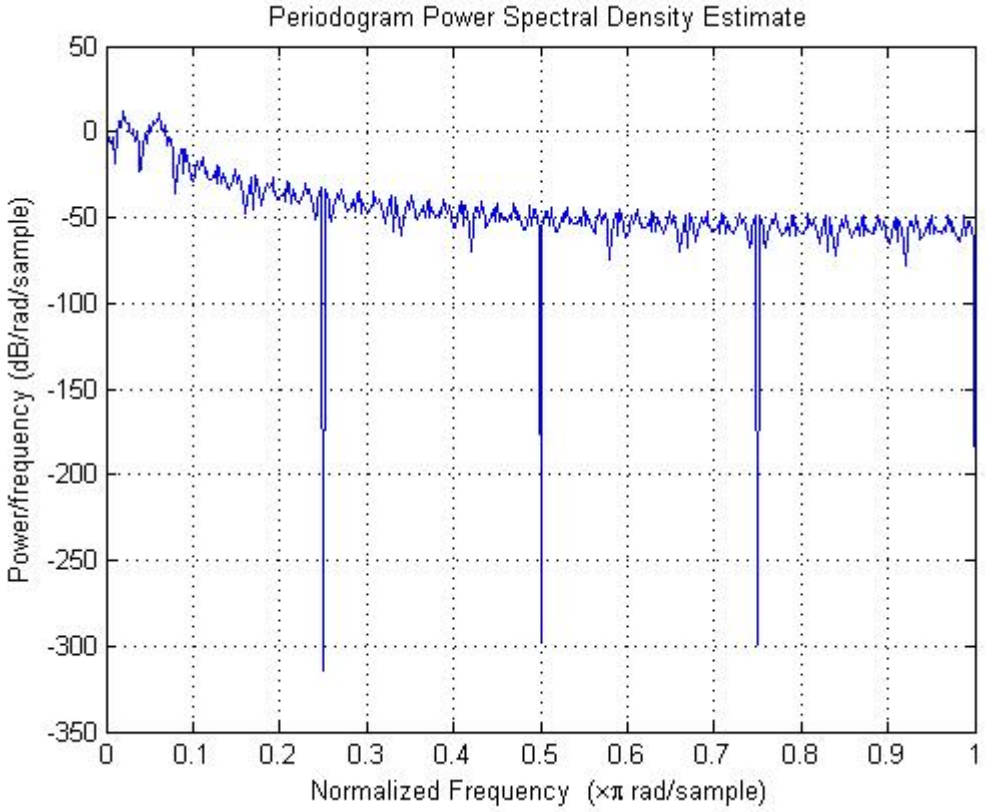
Output waveform for the bit stream [0 1 0 0 1 1 1 0]





Output waveform for the bit stream [1 0 1 1 0 0 0 1]





EXPERIMENT 3

Minimum Shift Keying

AIM:-

To plot the wave form for Minimum Shift Keying signal (MSK) using MATLAB for a stream of bits.

THEORY:-

In digital modulation, **minimum-shift keying (MSK)** is a type of continuous-phase frequency-shift keying Similar to OQPSK, MSK is encoded with bits alternating between quadrature components, with the Q component delayed by half the symbol period .

However, instead of square pulses as OQPSK uses, MSK encodes each bit as a half sinusoid. This results in a constant-modulus signal, which reduces problems caused by non-linear distortion. In addition to being viewed as related to OQPSK, MSK can also be viewed as a continuous phase frequency shift keyed (CPFSK) signal with a frequency separation of one-half the bit rate.

Mathematical representation

The resulting signal is represented by the formula

$$s(t) = a_I(t)\cos\left(\frac{\pi t}{2T}\right)\cos(2\pi f_c t) - a_Q(t)\sin\left(\frac{\pi t}{2T}\right)\sin(2\pi f_c t)$$

where $a_I(t)$ and $a_Q(t)$ encode the even and odd information respectively with a sequence of square pulses of duration $2T$. Using the trigonometric identity, this can be rewritten in a form where the phase and frequency modulation are more obvious,

$$s(t) = \cos\left[2\pi f_c t + b_k(t)\frac{\pi t}{2T} + \phi_k\right]$$

where $b_k(t)$ is +1 when $a_I(t) = a_Q(t)$ and -1 if they are of opposite signs, and φ_k is 0 if $a_I(t)$ is 1, and π otherwise. Therefore, the signal is modulated in frequency and phase, and the phase changes continuously and linearly.

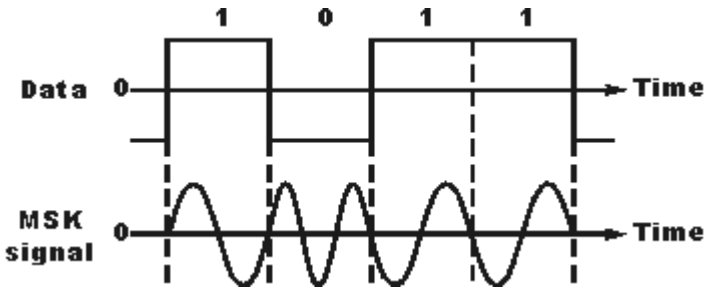
Reason for Minimum Shift Keying, MSK

It is found that binary data consisting of sharp transitions between "one" and "zero" states and vice versa potentially creates signals that have sidebands extending out a long way from the carrier, and this creates problems for many radio communications systems, as any sidebands outside the allowed bandwidth cause interference to adjacent channels and any radio communications links that may be using them.

Minimum Shift Keying, MSK basics

The problem can be overcome in part by filtering the signal, but is found that the transitions in the data become progressively less sharp as the level of filtering is increased and the bandwidth reduced. To overcome this problem GMSK is often used and this is based on Minimum Shift Keying, MSK modulation. The advantage of which is what is known as a continuous phase scheme. Here there are no phase discontinuities because the frequency changes occur at the carrier zero crossing points.

When looking at a plot of a signal using MSK modulation, it can be seen that the modulating data signal changes the frequency of the signal and there are no phase discontinuities. This arises as a result of the unique factor of MSK that the frequency difference between the logical one and logical zero states is always equal to half the data rate. This can be expressed in terms of the modulation index, and it is always equal to 0.5.



Signal using MSK modulation

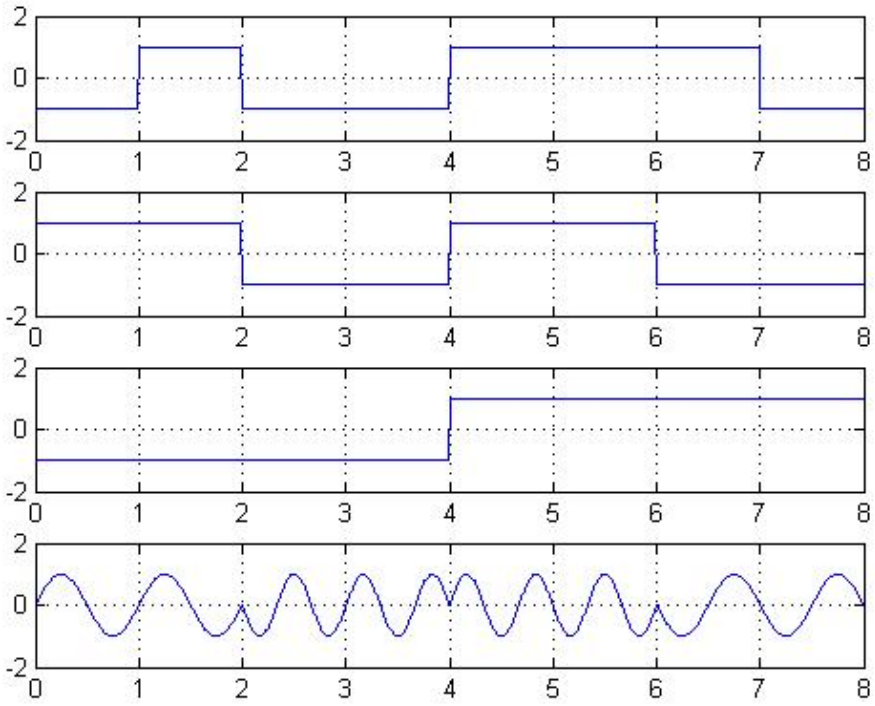
MATLAB PROGRAM:-

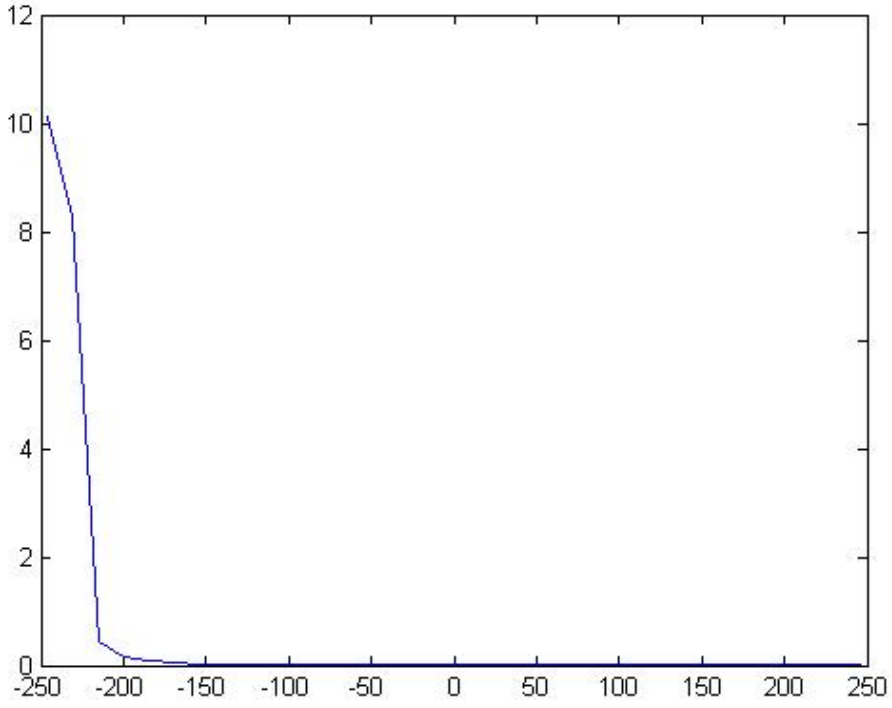
```
clear;
clc;
b = input('Enter the Bit stream \n '); %b = [0 1 0 1 1 1 0];
n = length(b);
t = 0:.01:n;
x = 1:1:(n+2)*100;
for i = 1:n
   if (b(i) == 0)
      b_p(i) = -1;
   else
      b_p(i) = 1;
   end
   for j = i:.1:i+1
      bw(x(i*100:(i+1)*100)) = b_p(i);
      if (mod(i,2) == 0)
         bow(x(i*100:(i+1)*100)) = b_p(i);
         bow(x((i+1)*100:(i+2)*100)) = b_p(i);
      else
         bew(x(i*100:(i+1)*100)) = b_p(i);
         bew(x((i+1)*100:(i+2)*100)) = b_p(i);
      end
      if (mod(n,2)~= 0)
         bow(x(n*100:(n+1)*100)) = -1;
         bow(x((n+1)*100:(n+2)*100)) = -1;
      end

   end
end
bw = bw(100:end);
bew = bew(100:(n+1)*100);
bow = bow(200:(n+2)*100);
wot = 2*pi*t*(5/4);
Wt = 2*pi*t/(4*1);
st = bow.*sin(wot+(bew.*bow).*Wt);
```

```
subplot(4,1,1)
plot(t,bw)
grid on ; axis([0 n -2 +2])
subplot(4,1,2)
plot(t,bow)
grid on ; axis([0 n -2 +2])
subplot(4,1,3)
plot(t,bew)
grid on ; axis([0 n -2 +2])
subplot(4,1,4)
plot(t,st)
grid on ; axis([0 n -2 +2])
Fs=5/4;                    %figure
                           %periodogram(st)
 S = fft(st,65);
PSS = S.* conj(S) / 65;
f = 1000*(-16:16)/65;
figure
plot(f,PSS(1:33))
```

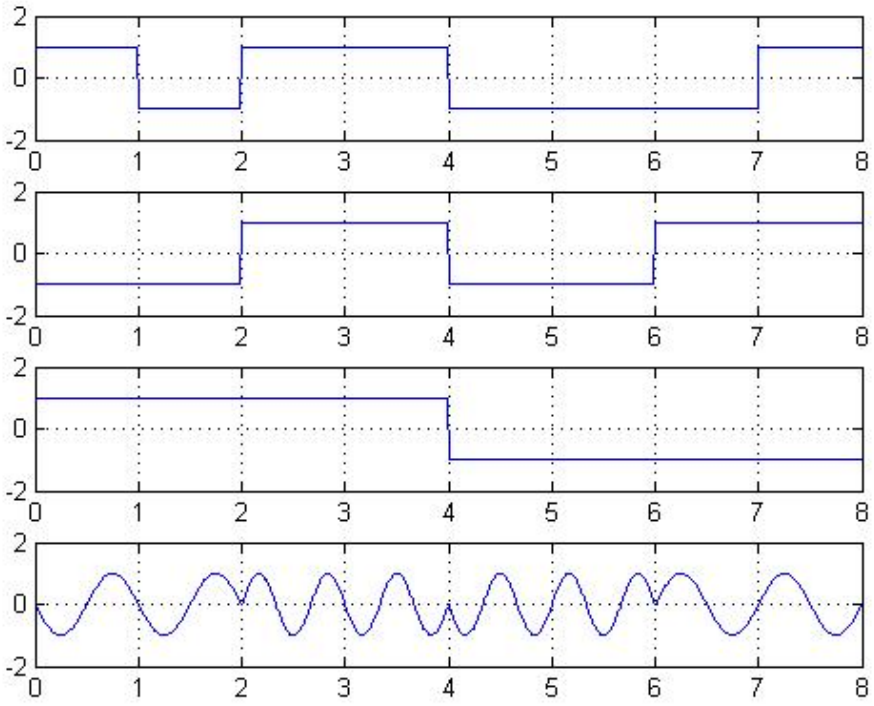
OBSERVATION

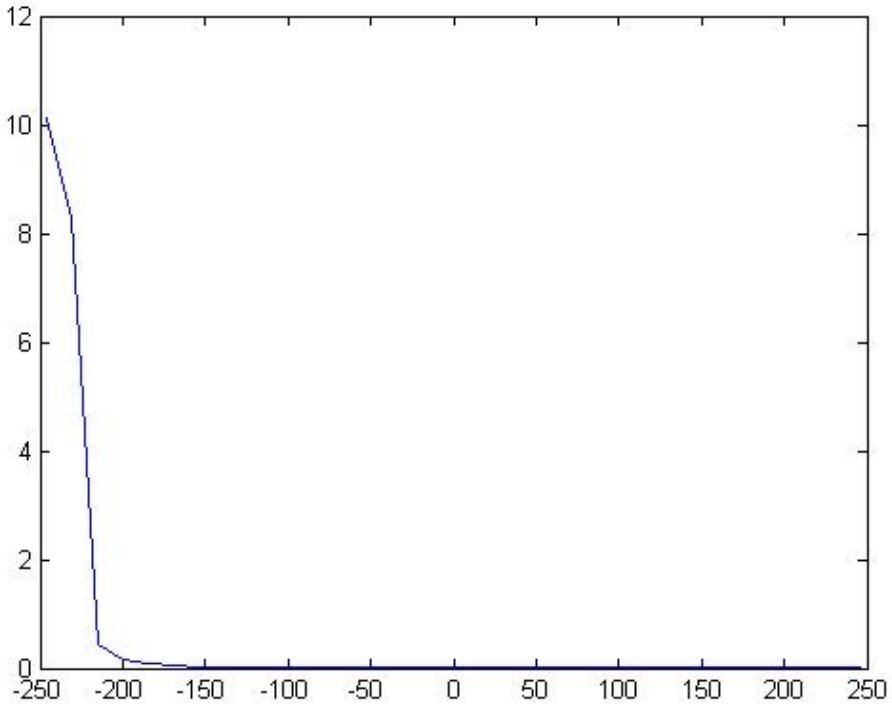
Output waveform for the bit stream [0 1 0 0 1 1 1 0]





Output waveform for the bit stream [1 0 1 1 0 0 0 1]





EXPERIMENT 4

Phase Shift Keying signal

AIM:-

To plot the wave form for Binary Phase Shift Keying signal (BPSK) using MATLAB for a stream of bits.

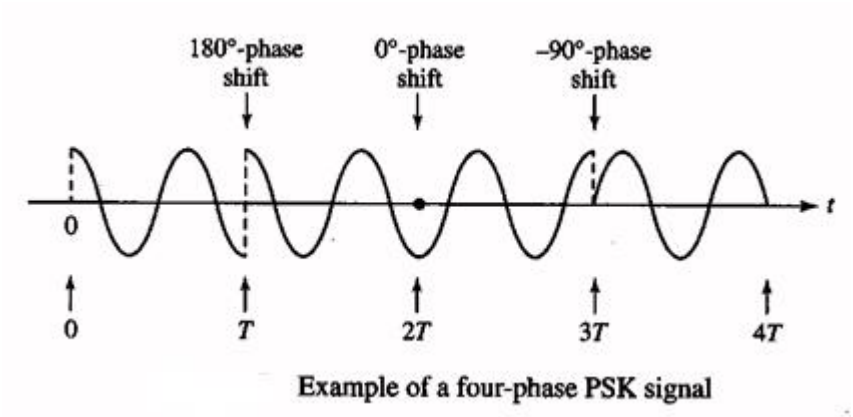
THEORY:-

In carrier-phase modulation, the information that is transmitted over a communication channel is impressed on the phase of the carrier. Science the range of the carrier phase is $0 \leq \theta \leq 2\Pi$,the carrier phases used to transmit digital information via digital-phase modulation are $\theta_m=2\Pi m/M$, for m=0,1,2…..,M-1.Thus for binary phase modulation(M=2),the two carrier phase are $\theta_0 =0$ and $\theta_1 = \Pi$ radian. For M-array phase modulation=$2^{k,}$ where k is the number of information bits per transmitted symbol.

The general representation of a set of M carrier-phase-modulated signal waveforms is

$u_m(t) = AgT(t) \cos(2\Pi fct+2\Pi m/M)$, m=0,1,……,M-1

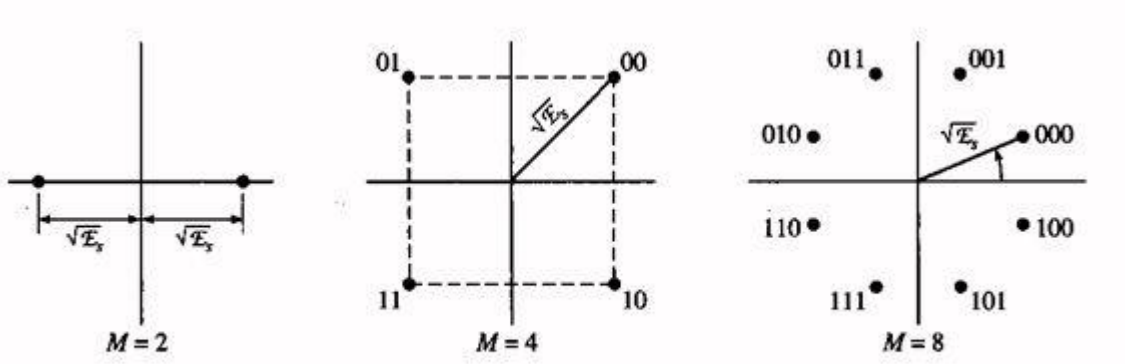
Where, gT(t) is the transmitting filter pulse shape, which determines the spectral characteristics of the transmitted signal, and A is the signal amplitude. This type of digital phase modulation is called phase-shift-keying.



Example of a four-phase PSK signal

Signal point constellations for M=2, 4 and 8 are illustrated in figure. We observe that binary phase modulation is identical to binary PAM (binary antipodal signals).The mapping or assignment, of k information bits into the M= 2^k possible phases may be done in a number of ways. The preferred assignment is to Gray in coding, in which adjacent phases differ by one binary digit, as illustrated

below in the figure. Consequently, only a single bit error occurs in the k-bit sequence with Gray encoding when noise causes the incorrect selection of an adjacent phase to the transmitted phase



PSK signal constellations

In figure shows that, block diagram of M=4 PSK system. The uniform random number generator fed to the 4-PSK mapper and also fed to the compare. The 4-PSK mapper split up into two phases. On the other hand, Gaussian RNG adds to the modulator. The two phases are fed to the detector. The output goes to the compare. The Uniform random number generator and detector also fed to the detector and finally fed to the bit-error counter and symbol-error counter.

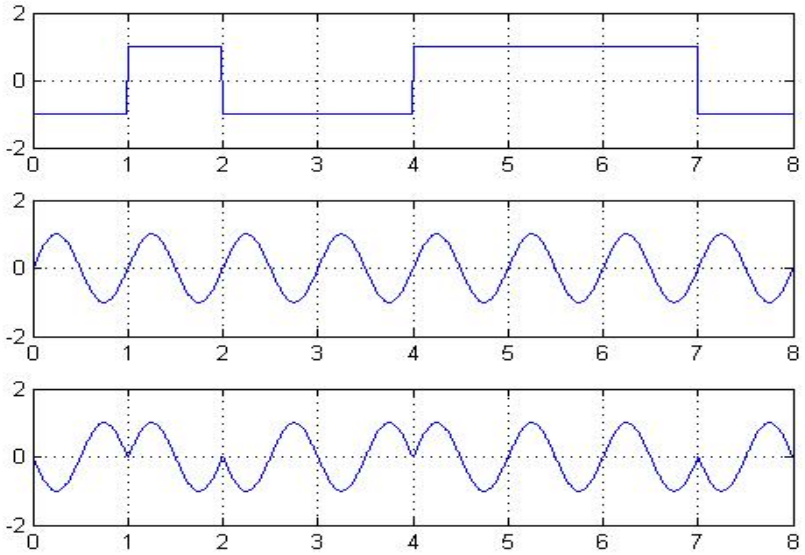
MATLAB PROGRAM:-

```
clear;
clc;
b = input('Enter the Bit stream \n ');          %b = [0 1 0 1 1 1 0];
n = length(b);
t = 0:.01:n;
x = 1:1:(n+1)*100;
for i = 1:n
   if (b(i) == 0)
      b_p(i) = -1;
   else
      b_p(i) = 1;
   end
   for j = i:.1:i+1
      bw(x(i*100:(i+1)*100)) = b_p(i);
   end
end
```

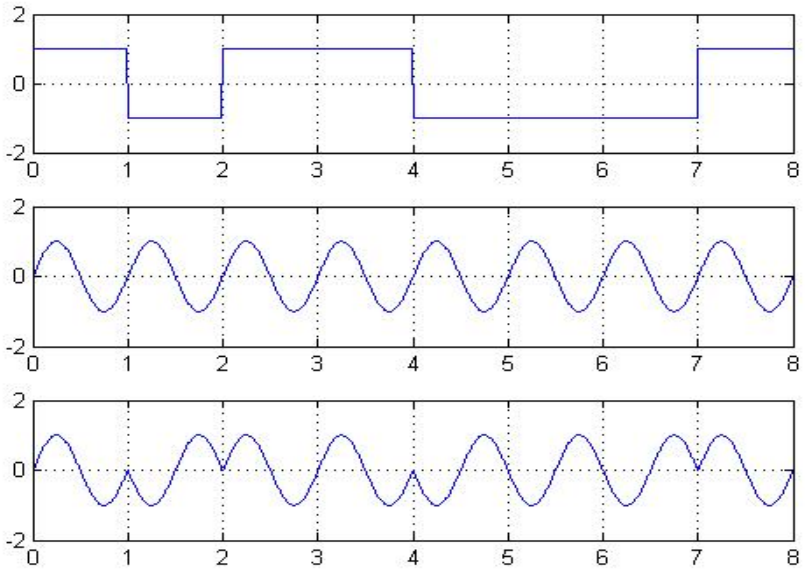
```
bw = bw(100:end);
sint = sin(2*pi*t);
st = bw.*sint;
subplot(3,1,1)
plot(t,bw)
grid on ; axis([0 n -2 +2])
subplot(3,1,2)
plot(t,sint)
grid on ; axis([0 n -2 +2])
subplot(3,1,3)
plot(t,st)
grid on ; axis([0 n -2 +2])
```

OBSERVATION:-

Output waveform for the bit stream [0 1 0 0 1 1 1 0]



Output waveform for the bit stream [1 0 1 1 0 0 0 1]



EXPERIMENT 5

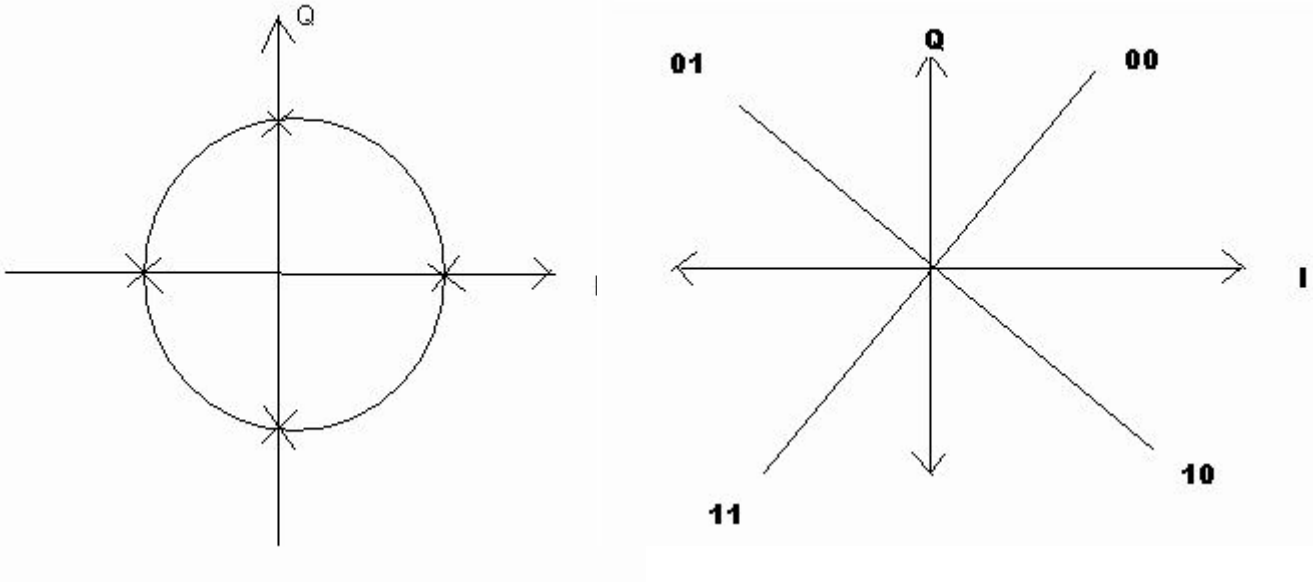
Quadrature Phase Shift Keying

AIM:-

To plot the wave form for Quadrature Phase Shift Keying (QPSK) signal using MATLAB for a stream of bits.

THEORY:-

Quadrature Phase Shift Keying (QPSK) is the digital modulation technique.Quadrature Phase Shift Keying (QPSK) is a form of Phase Shift Keying in which two bits are modulated at once, selecting one of four possible carrier phase shifts (0, Π/2, Π, and 3Π/2). QPSK perform by changing the phase of the In-phase (I) carrier from 0° to 180° and the Quadrature-phase (Q) carrier between 90° and 270°. This is used to indicate the four states of a 2-bit binary code. Each state of these carriers is referred to as a Symbol.



QPSK perform by changing the phase of the In-phase (I) carrier from 0° to 180° and the Quadrature-phase (Q) carrier between 90° and 270°. This is used to indicate the four states of a 2-bit binary code. Each state of these carriers is referred to as a Symbol. Quadrature Phase-shift Keying (QPSK) is a widely used method of transferring digital data by changing or modulating the phase of a carrier signal. In QPSK digital data is represented by 4 points around a circle which correspond to 4 phases of the carrier signal. These points are called symbols. Fig. shows this mapping.

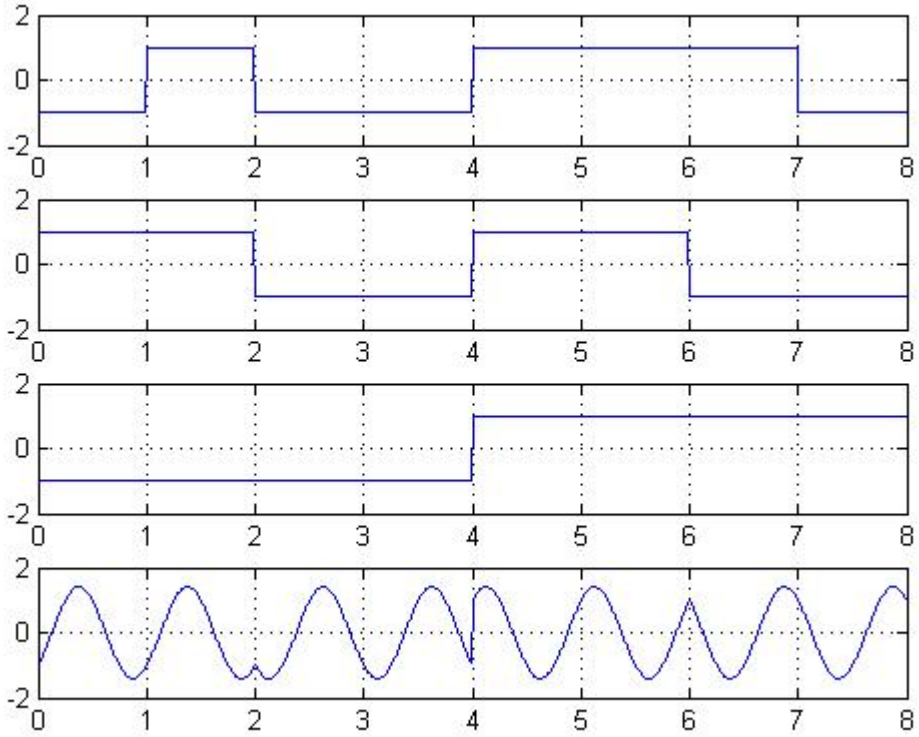
MATLAB PROGRAM:-

```
clear;
clc;
b = input('Enter the Bit stream \n ');              %b = [0 1 0 1 1 1 0];
n = length(b);
t = 0:.01:n;
x = 1:1:(n+2)*100;
for i = 1:n
   if (b(i) == 0)
      b_p(i) = -1;
   else
      b_p(i) = 1;
   end
   for j = i:.1:i+1
      bw(x(i*100:(i+1)*100)) = b_p(i);
      if (mod(i,2) == 0)
         bow(x(i*100:(i+1)*100)) = b_p(i);
         bow(x((i+1)*100:(i+2)*100)) = b_p(i);
      else
         bew(x(i*100:(i+1)*100)) = b_p(i);
         bew(x((i+1)*100:(i+2)*100)) = b_p(i);
      end
      if (mod(n,2)~= 0)
         bow(x(n*100:(n+1)*100)) = -1;
         bow(x((n+1)*100:(n+2)*100)) = -1;
      end
   end
end      %be = b_p(1:2:end);
         %bo = b_p(2:2:end);
bw = bw(100:end);
bew = bew(100:(n+1)*100);
bow = bow(200:(n+2)*100);
cost = cos(2*pi*t);
sint = sin(2*pi*t);
st = bew.*cost+bow.*sint;
```

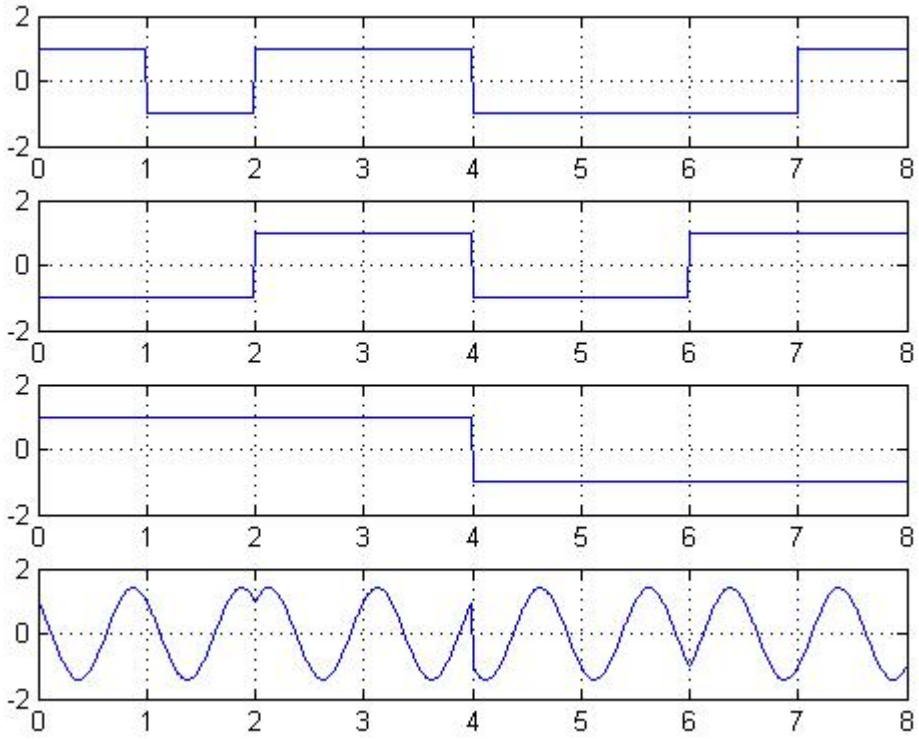
```
 subplot(4,1,1)
plot(t,bw)
grid on ; axis([0 n -2 +2])
subplot(4,1,2)
plot(t,bow)
grid on ; axis([0 n -2 +2])
subplot(4,1,3)
plot(t,bew)
grid on ; axis([0 n -2 +2])
subplot(4,1,4)
plot(t,st)
grid on ; axis([0 n -2 +2])
```

OBSERVATION:-

Output waveform for the bit stream [0 1 0 0 1 1 1 0]



Output waveform for the bit stream [1 0 1 1 0 0 0 1]



EXPERIMENT 6

8-QAM

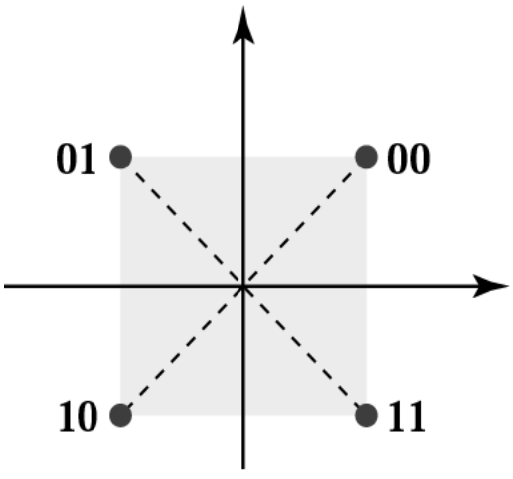
AIM:-

To plot the wave form for 8 quadrature amplitude modulated signal (QAM) using MATLAB for a stream of bits.

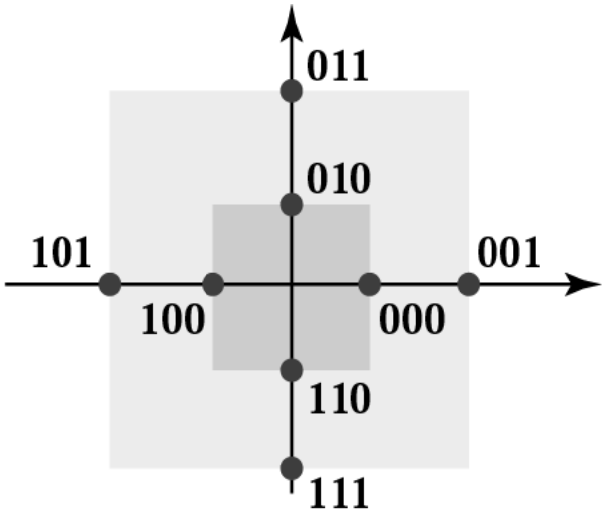
THEORY:-

Quadrature amplitude modulation (QAM) is both an analog and a digital modulation scheme. It conveys two analog message signals, or two digital bit streams, by changing (modulating) the amplitudes of two carrier waves, using the amplitude-shift keying (ASK) digital modulation scheme or amplitude modulation (AM) analog modulation scheme. The two carrier waves, usually sinusoids, are out of phase with each other by 90° and are thus called quadrature carriers or quadrature components — hence the name of the scheme. The modulated waves are summed, and the resulting waveform is a combination of both phase-shift keying (PSK) and amplitude-shift keying (ASK), or (in the analog case) of phase modulation (PM) and amplitude modulation. In the digital QAM case, a finite number of at least two phases and at least two amplitudes are used. PSK modulators are often designed using the QAM principle, but are not considered as QAM since the amplitude of the modulated carrier signal is constant. QAM is used extensively as a modulation scheme for digital telecommunication systems. Spectral efficiencies of 6 bits/s/Hz can be achieved with QAM.

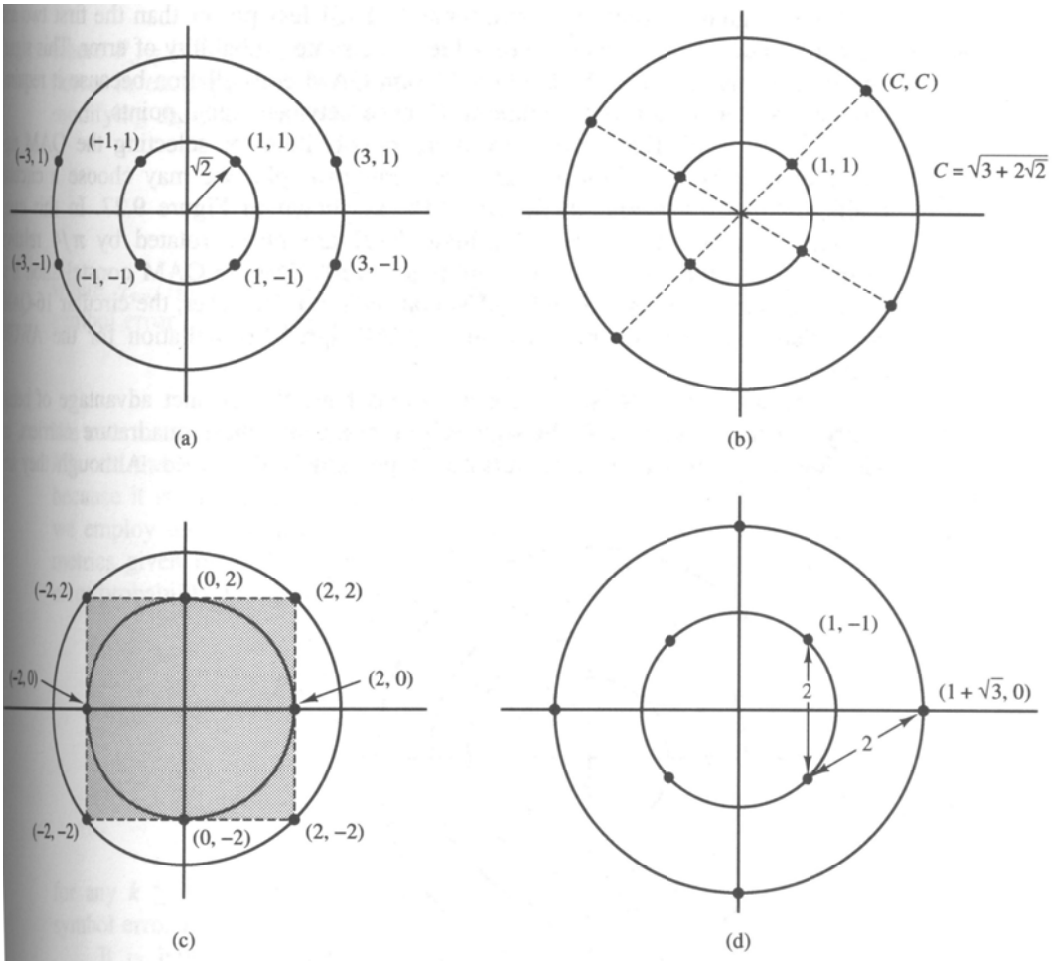
The 4-QAM and 8-QAM constellations



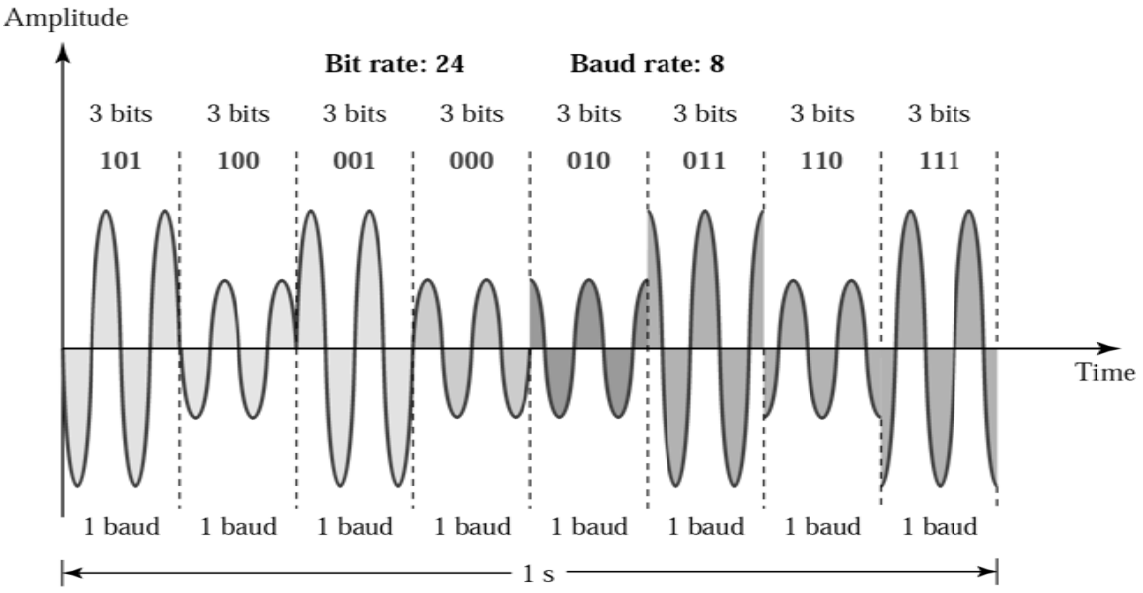
4-QAM
1 amplitude, 4 phases



8-QAM
2 amplitudes, 4 phases



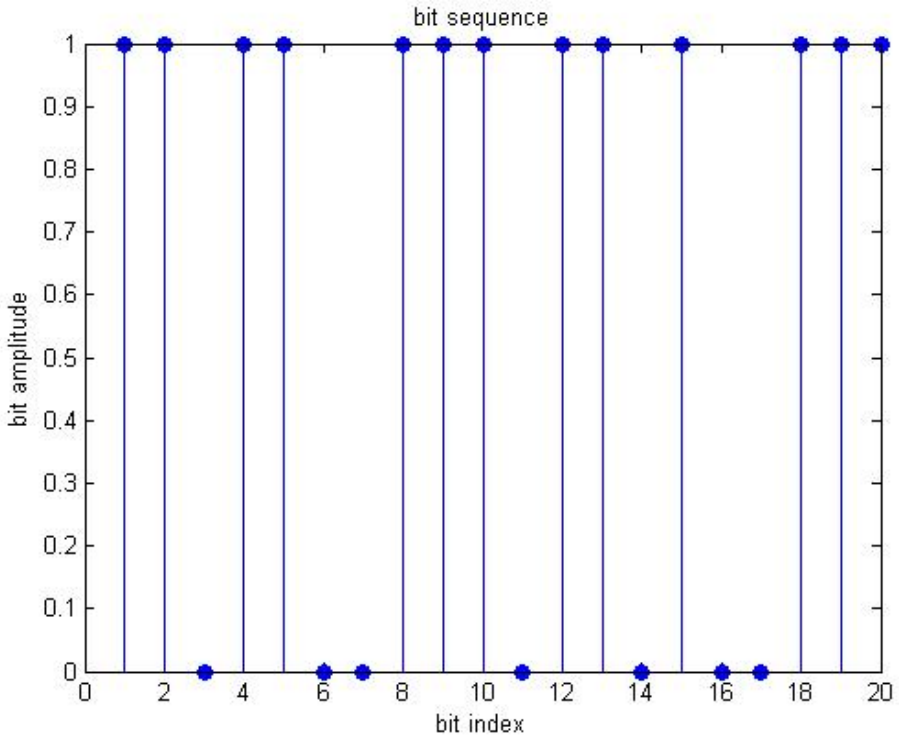
Time domain for an 8-QAM signal

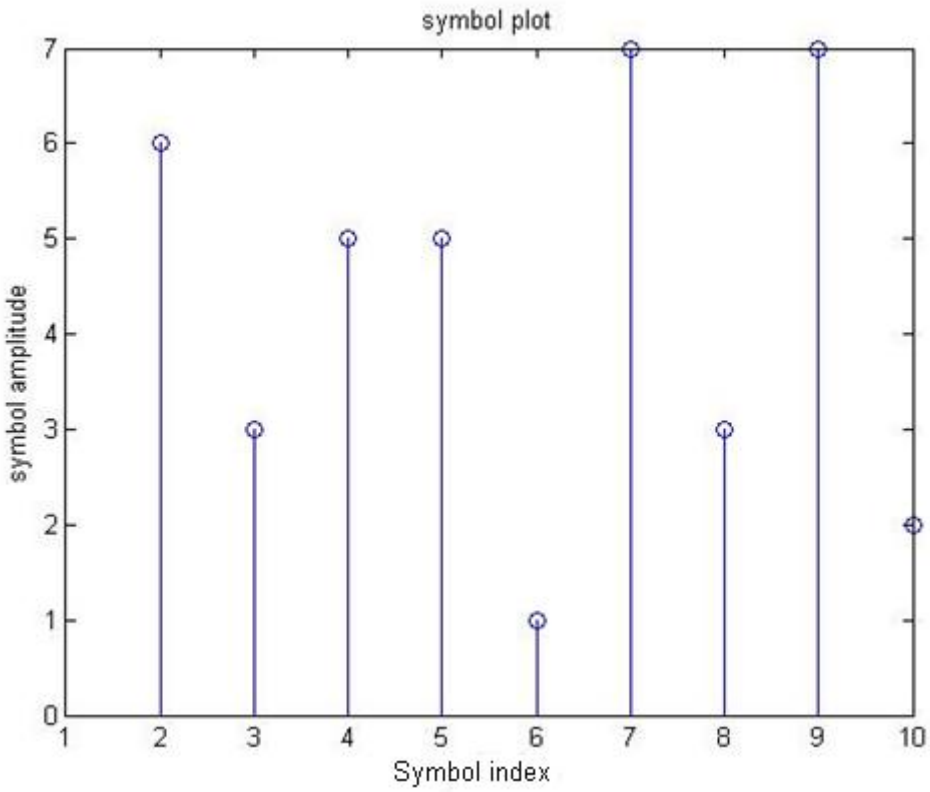


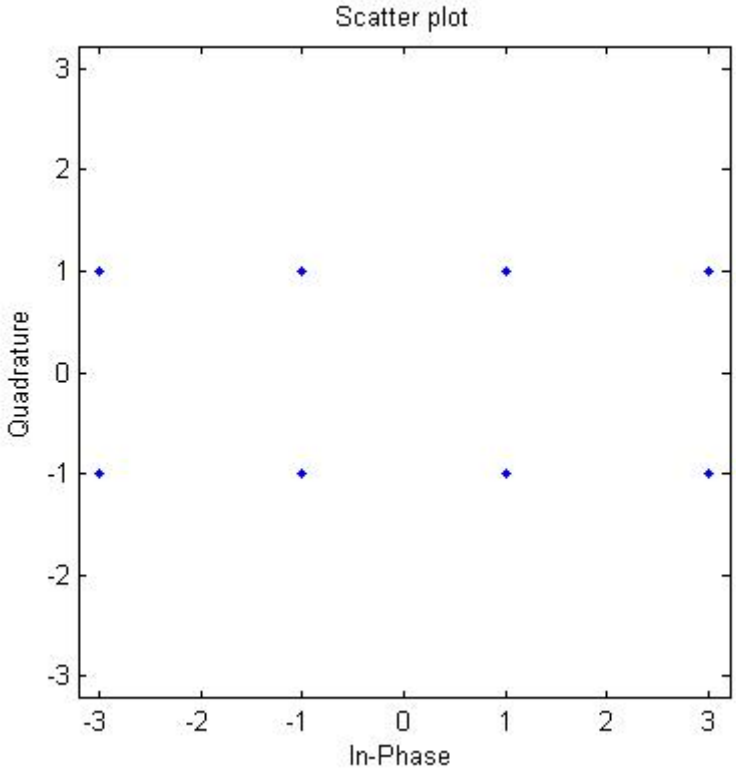
MATLAB PROGRAM:-

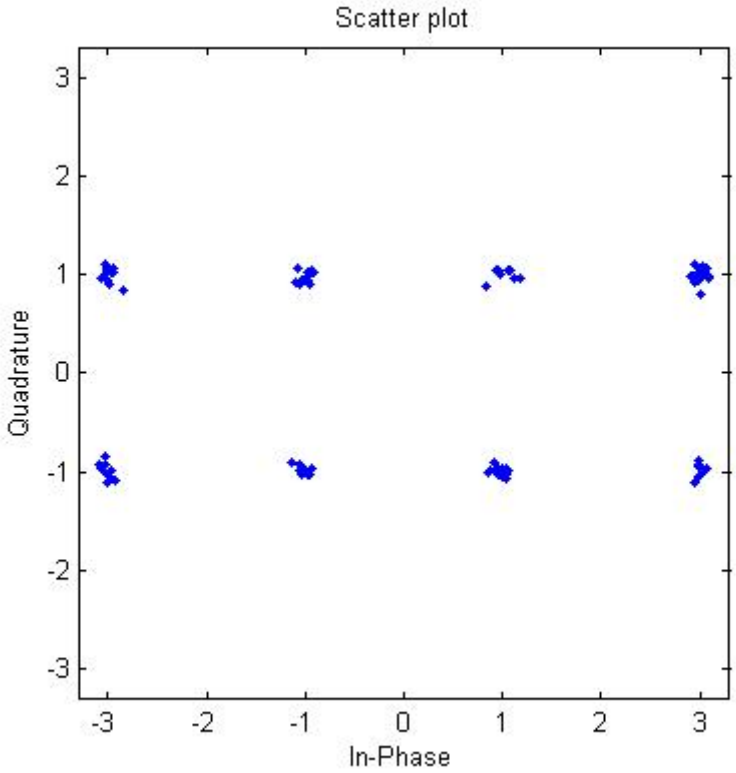
```
clc
close all
m=8
k=log2(m);
n=9e3;
nsamp=1;
x=randint(n,1);
stem(x(1:20),'filled');
title('bit sequence');
xlabel('bit index');
ylabel('bit amplitude');
xsym=bi2de(reshape(x,k,length(x)/k).','left-msb');
figure;
stem(xsym(1:10));
title('symbol plot');
xlabel('symbol index');
ylabel('symbol amplitude');
y=modulate(modem.qammod(m),xsym);
ytx=y;
ebno=10
snr=ebno+10*log(k)-10*log10(nsamp);
yn=awgn(ytx,snr);
yrx=yn;
scatterplot(y);
scatterplot(yrx,30);
```

OBSERVATION:-






Scatter plot
Quadrature
In-Phase
3
2
1
0
-1
-2
-3


Scatter plot
Quadrature
In-Phase
3
2
1
0
-1
-2
-3

EXPERIMENT 7

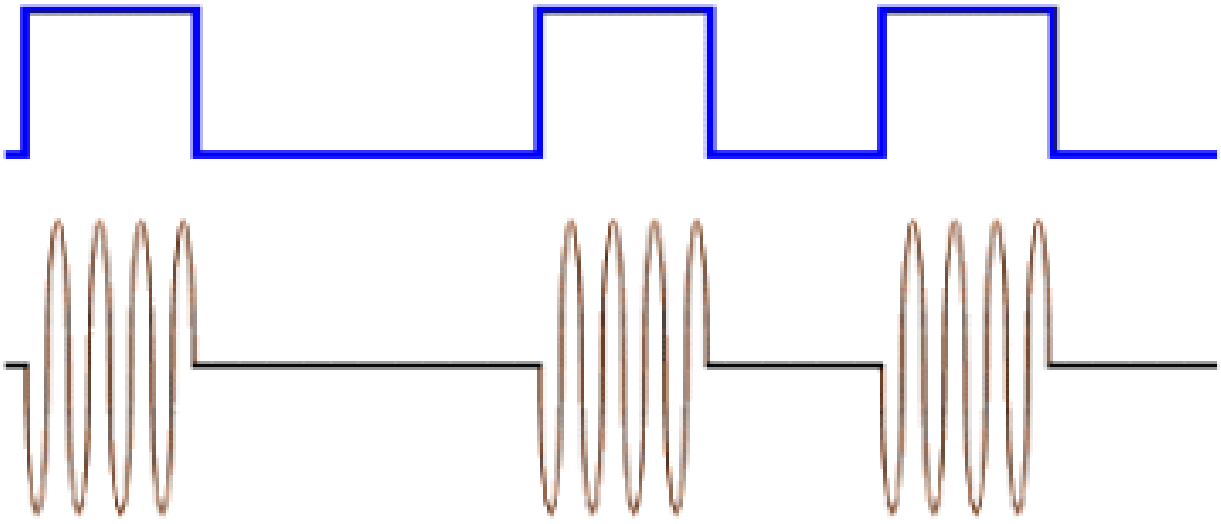
On Off Keying

AIM:-

To plot the wave forms for on-off keying using MATLAB.

THEORY:-

On-off keying (**OOK**) the simplest form of amplitude-shift keying (ASK) modulation that represents digital data as the presence or absence of a carrier wave. In its simplest form, the presence of a carrier for a specific duration represents a binary one, while its absence for the same duration represents a binary zero. Some more sophisticated schemes vary these durations to convey additional information. It is analogous to unipolar encoding line code. On-off keying is most commonly used to transmit Morse code over radio frequencies (referred to as CW (continuous wave) operation); although in principle any digital encoding scheme may be used. OOK has been used in the ISM bands to transfer data between computers, for example. OOK is more spectrally efficient than FSK, but more sensitive to noise. In addition to RF carrier waves, OOK is also used in optical communication systems (e.g. IrDA).



MATLAB PROGRAM:-

```
clear all
close all
tic

x = [1 0 0 0 0 0 0];
l = length(x);
sp = input('The Speed of Transmission for the Data');
d = input('The Number of data points you want to transmit');
lq = input('The number of times you want to transmit the data');
filat = input('Do you want to use a filter in this transmisssion, if yes press 1');
duty = input('What is the duty Ratio you want to use for the RZ pulse, must be less than 1');
i = 0;

while (j < (2^l - 1))

  y = xor(x(6),x(7));
  temp = [y,x];
  x = temp;
  j = length(x);

end

data(1:d) = zeros;
while(i<d)
dgen(i+1:i+j) = x;
i = length(dgen);
if (i>d)
   data(1:d) = dgen(1:d);

   clear dgen temp
t = (1/(32*sp))*(1:1:(32*d));
end
end
% Pulse Generation
% NRZ, RZ
nr(1:32) = ones;
rz(1:32) = zeros;
lo = nearest(duty*32);
rz(1:lo) = ones;

data2 = kron(data,nr);
data3 = kron(data,rz);
subplot(2,1,1),plot(t,data2);hold
grid minor

subplot(2,1,2),plot(t,data3);hold
grid minor

datt2(1:d) = zeros;
datt3(1:d) = zeros;
```

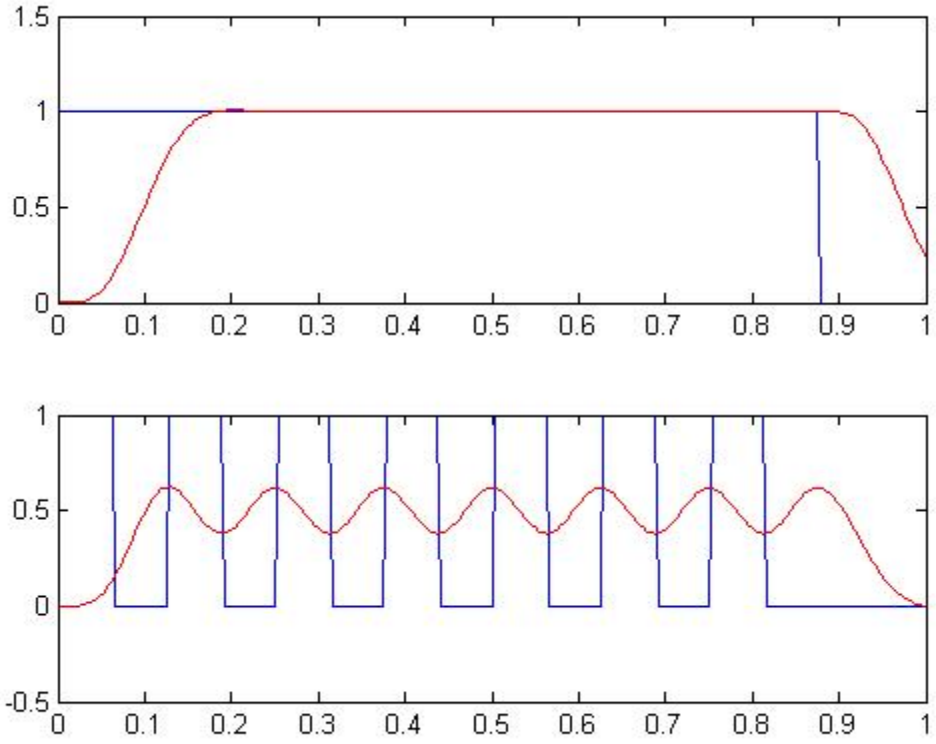
```
if (filat == 1)
[B,A] = besself(5,0.8*10^9*2*pi);
[NUMd,DENd] = bilinear(B,A, 32*10^9);
datat2 = filter(NUMd,DENd,data2);
datat3 = filter(NUMd,DENd,data3);
else
   datat2 = data2;
   datat3 = data3;
end
subplot(2,1,1),plot(t,datat2,'-r');
grid minor

subplot(2,1,2),plot(t,datat3,'-r');
grid minor

h = waitbar(0,'Error Computation');
BER1(1:lq,1:10) = zeros;
BER2(1:lq,1:10) = zeros;
for rep = 1:lq
for SNR = 1:10;
Pn2 =(sum(datat2.^2)/(length(datat2)))* 1*10^(-SNR/20);
Pn3 =(sum(datat3.^2)/(length(datat3)))* 1*10^(-SNR/20);

dat2 = datat2+(Pn2*randn(1,length(datat2)));
dat3 = datat3+(Pn3*randn(1,length(datat3)));

j = 0;
i =1;
lent = length(datat2);
while(j<=lent-1)

   if(dat2(j+16)<0.5)
     datt2(i) = 0;
   else
      datt2(i) =1;
   end

   if(dat3(j+nearest(lo/2))<0.5)
     datt3(i) = 0;
   else
      datt3(i) =1;
   end

   j = j+ 32;
   i = i+1;
end

BER1(rep,SNR) = mean(data~=datt2)/length(data);
BER2(rep,SNR) = mean(data~=datt3)/length(data);
waitbar((SNR+((rep-1)*10))/(10*lq),h);

end
```

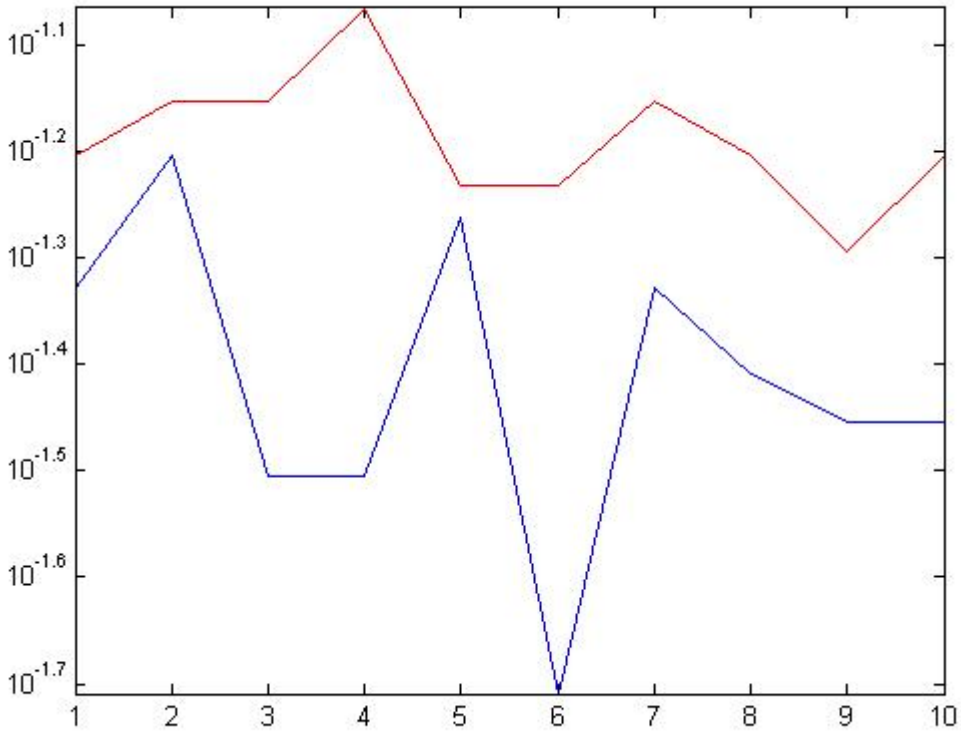
```
end
close(h)
BER1x = sum(BER1(1:lq,:))/lq;
BER2x = sum(BER2(1:lq,:))/lq;
figure(2),semilogy(BER1x);
hold
semilogy(BER2x,'-r');

toc
```

OBSERVATION:-

Input data:-

The Speed of Transmission for the Data 8
The Number of data points you want to transmit 8
The number of times you want to transmit the data 4
Do you want to use a filter in this transmission, if yes press 1 1
What is the duty Ratio you want to use for the RZ pulse, must be less than 1 0.5




$10^{-1.1}$
$10^{-1.2}$
$10^{-1.3}$
$10^{-1.4}$
$10^{-1.5}$
$10^{-1.6}$
$10^{-1.7}$
1
2
3
4
5
6
7
8
9
10

EXPERIMENT 8

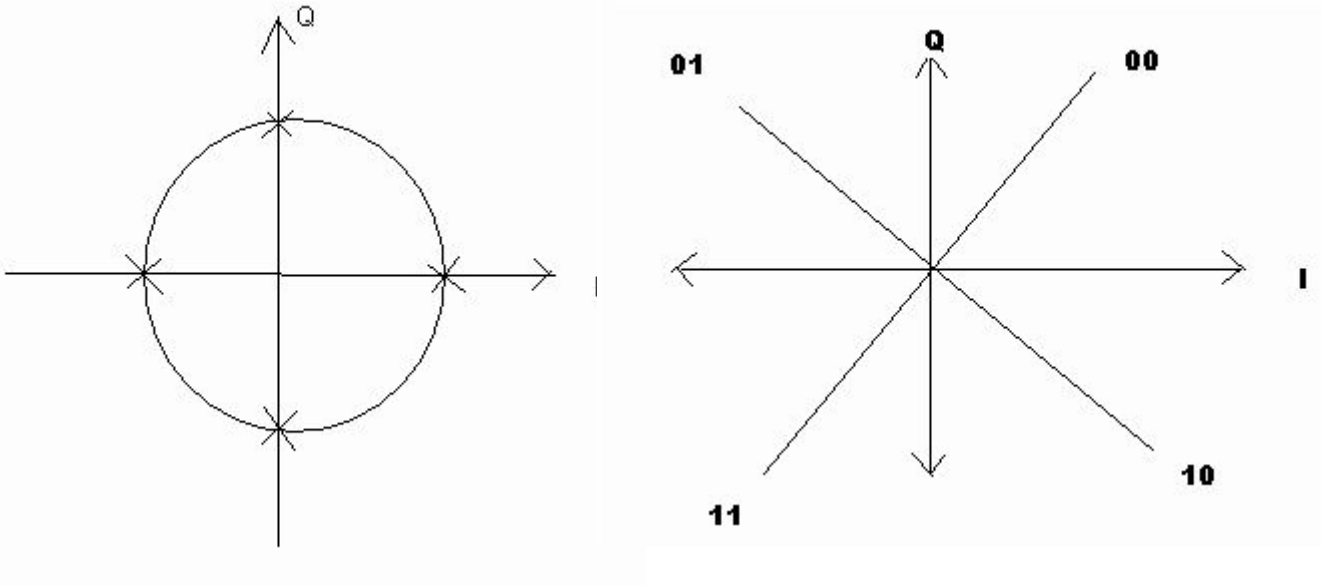
QPSK with rayleign fading & AWGN

AIM:-

To plot the wave forms for QPSK signal subjected to rayleigh AWGN using MATLAB.

THEORY:-

Quadrature Phase Shift Keying (QPSK) is the digital modulation technique.Quadrature Phase Shift Keying (QPSK) is a form of Phase Shift Keying in which two bits are modulated at once, selecting one of four possible carrier phase shifts (0, Π/2, Π, and 3Π/2). QPSK perform by changing the phase of the In-phase (I) carrier from 0° to 180° and the Quadrature-phase (Q) carrier between 90° and 270°. This is used to indicate the four states of a 2-bit binary code. Each state of these carriers is referred to as a Symbol.



Rayleigh fading is a statistical model for the effect of a propagation environment on a radio signal, such as that used by wireless devices.Rayleigh fading models assume that the magnitude of a signal that has passed through such a transmission medium (also called a communications channel) will vary randomly, or fade,according to a Rayleigh distribution — the radial component of the sum of two uncorrelated Gaussian random variables.

Additive white Gaussian noise (**AWGN**) is a channel model in which the only impairment to communication is a linear addition of wideband or white noise with a constant spectral density (expressed as watts per hertz of bandwidth) and a Gaussian distribution of amplitude.

MATLAB PROGRAM:-

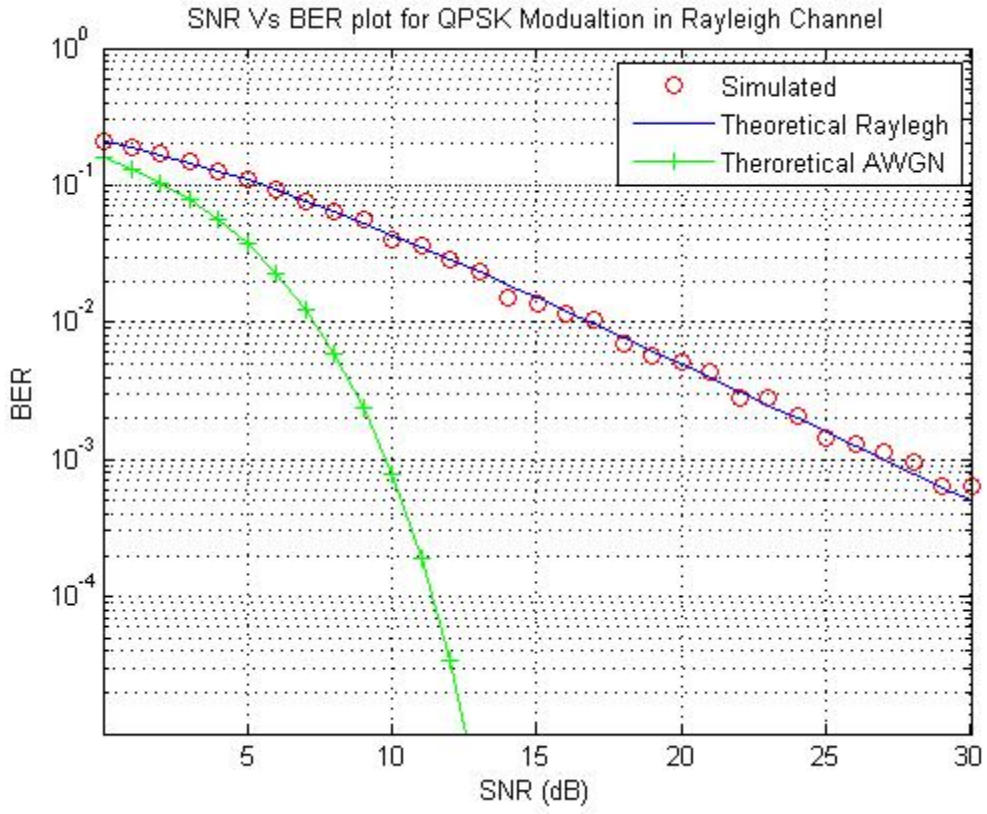
```
clear all;
close all;

format long;
bit_count = 10000;
Eb_No = -3: 1: 30;

SNR = Eb_No + 10*log10(2);
for aa = 1: 1: length(SNR)

   T_Errors = 0;
   T_bits = 0;

   while T_Errors < 100

      uncoded_bits  = round(rand(1,bit_count));

      B1 = uncoded_bits(1:2:end);
      B2 = uncoded_bits(2:2:end);

      qpsk_sig = ((B1==0).*(B2==0)*(exp(i*pi/4))+(B1==0).*(B2==1)...
         *(exp(3*i*pi/4))+(B1==1).*(B2==1)*(exp(5*i*pi/4))...
         +(B1==1).*(B2==0)*(exp(7*i*pi/4)));

            ray = sqrt(0.5*((randn(1,length(qpsk_sig))).^2+(randn(1,length(qpsk_sig))).^2));
      rx = qpsk_sig.*ray;

      N0 = 1/10^(SNR(aa)/10);

      rx = rx + sqrt(N0/2)*(randn(1,length(qpsk_sig))+i*randn(1,length(qpsk_sig)));

      rx = rx./ray;

      B4 = (real(rx)<0);
      B3 = (imag(rx)<0);

      uncoded_bits_rx = zeros(1,2*length(rx));
      uncoded_bits_rx(1:2:end) = B3;
      uncoded_bits_rx(2:2:end) = B4;

      diff = uncoded_bits - uncoded_bits_rx;
      T_Errors = T_Errors + sum(abs(diff));
      T_bits = T_bits + length(uncoded_bits);

   end
   figure; clf;
   plot(real(rx),imag(rx),'o');  % Scatter Plot
   title(['constellation of received symbols for SNR = ', num2str(SNR(aa))]);
   xlabel('Inphase Component'); ylabel('Quadrature Component');

   BER(aa) = T_Errors / T_bits;
```

```
    disp(sprintf('bit error probability = %f',BER(aa)));

end

figure(1);
semilogy(SNR,BER,'or');
hold on;
xlabel('SNR (dB)');
ylabel('BER');
title('SNR Vs BER plot for QPSK Modualtion in Rayleigh Channel');

figure(1);
EbN0Lin = 10.^(Eb_No/10);
theoryBerRay = 0.5.*(1-sqrt(EbN0Lin./(EbN0Lin+1)));
semilogy(SNR,theoryBerRay);
grid on;

figure(1);
theoryBerAWGN = 0.5*erfc(sqrt(10.^(Eb_No/10)));
semilogy(SNR,theoryBerAWGN,'g-+');
grid on;
legend('Simulated', 'Theoretical Raylegh', 'Theroretical AWGN');
axis([SNR(1,1) SNR(end-3) 0.00001 1]);
```

OBSERVATION:-

Output data:-

bit error probability = 0.211700
bit error probability = 0.197600
bit error probability = 0.169700
bit error probability = 0.151700
bit error probability = 0.119500
bit error probability = 0.107600
bit error probability = 0.091600
bit error probability = 0.077700
bit error probability = 0.063000
bit error probability = 0.055400
bit error probability = 0.044400
bit error probability = 0.036600
bit error probability = 0.030600
bit error probability = 0.023700
bit error probability = 0.020300
bit error probability = 0.017300
bit error probability = 0.011000
bit error probability = 0.009650
bit error probability = 0.007350
bit error probability = 0.006700
bit error probability = 0.005167
bit error probability = 0.004233
bit error probability = 0.003767
bit error probability = 0.002400
bit error probability = 0.001983
bit error probability = 0.001529
bit error probability = 0.001122
bit error probability = 0.001055
bit error probability = 0.000777
bit error probability = 0.000644
bit error probability = 0.000452
bit error probability = 0.000446
bit error probability = 0.000306
bit error probability = 0.000289


SNR Vs BER plot for QPSK Modualtion in Rayleigh Channel
Simulated
Theoretical Raylegh
Theroretical AWGN
BER
SNR (dB)
10^0
10^-1
10^-2
10^-3
10^-4
5
10
15
20
25
30

EXPERIMENT 9

QAM with AWGN fading

AIM:-

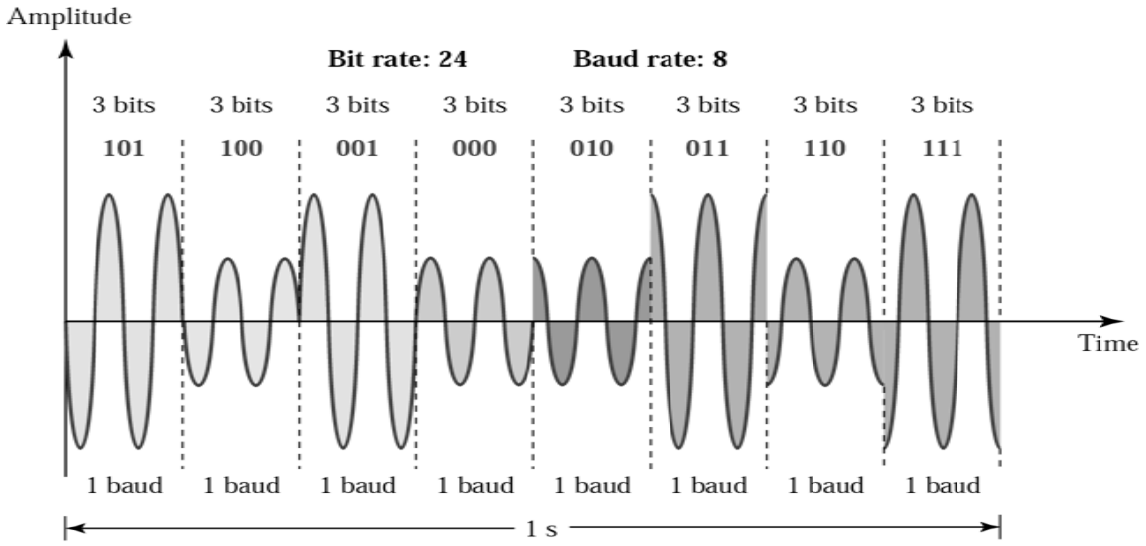
To plot the wave forms for QAM signal subjected to AWGN fading using MATLAB.

THEORY:-

Quadrature amplitude modulation (QAM) is both an analog and a digital modulation scheme. It conveys two analog message signals, or two digital bit streams, by changing (modulating) the amplitudes of two carrier waves, using the amplitude-shift keying (ASK) digital modulation scheme or amplitude modulation (AM) analog modulation scheme. The two carrier waves, usually sinusoids, are out of phase with each other by 90° and are thus called quadrature carriers or quadrature components — hence the name of the scheme. The modulated waves are summed, and the resulting waveform is a combination of both phase-shift keying (PSK) and amplitude-shift keying (ASK), or (in the analog case) of phase modulation (PM) and amplitude modulation. In the digital QAM case, a finite number of at least two phases and at least two amplitudes are used. PSK modulators are often designed using the QAM principle, but are not considered as QAM since the amplitude of the modulated carrier signal is constant. QAM is used extensively as a modulation scheme for digital telecommunication systems. Spectral efficiencies of 6 bits/s/Hz can be achieved with QAM.

Additive white Gaussian noise (**AWGN**) is a channel model in which the only impairment to communication is a linear addition of wideband or white noise with a constant spectral density (expressed as watts per hertz of bandwidth) and a Gaussian distribution of amplitude.

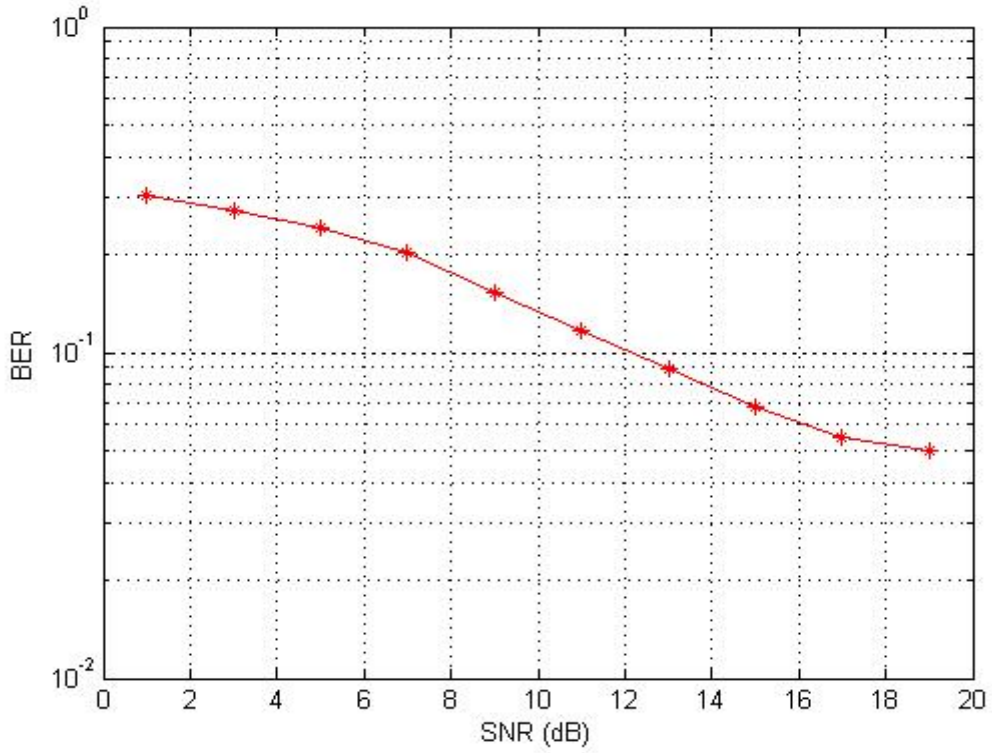
TIME DOMAIN FOR AN 8 QAM SIGNAL



MATLAB PROGRAM:-

```
M = 16;

SNR = 1:2:20;

bitRate = 10000;
ch = rayleighchan(1/bitRate, 10, 0 ,0);
ch.ResetBeforeFiltering = 0;

tx = randint(5000,1,M);
hmod=qammod(tx,M);
gmod = qamdemod(hmod,M);

fad = abs(filter(ch, ones(size(hmod))));
fadedSig = fad.*hmod;

BER = ones(size(SNR));
for n = 1:length(SNR)

   rxSig = awgn(fadedSig,SNR(n),'measured');

   rx = qamdemod(rxSig,M);
   [nErrors, BER(n)] = biterr(tx,rx);
end
semilogy(SNR,BER,'r-*');
xlabel('SNR (dB)');
ylabel('BER');
grid on
```

OBSERVATION:-



EXPERIMENT 10

16-QAM with BER

AIM:-

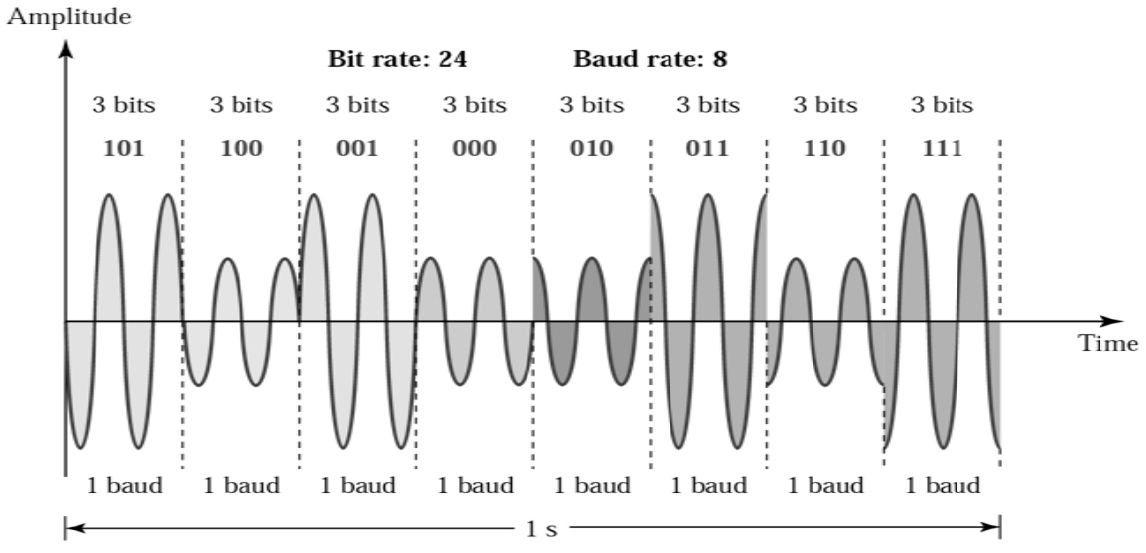
To plot the wave forms for 16-QAM signal with BER using MATLAB.

THEORY:-

Quadrature amplitude modulation (QAM) is both an analog and a digital modulation scheme. It conveys two analog message signals, or two digital bit streams, by changing (modulating) the amplitudes of two carrier waves, using the amplitude-shift keying (ASK) digital modulation scheme or amplitude modulation (AM) analog modulation scheme. The two carrier waves, usually sinusoids, are out of phase with each other by 90° and are thus called quadrature carriers or quadrature components — hence the name of the scheme. The modulated waves are summed, and the resulting waveform is a combination of both phase-shift keying (PSK) and amplitude-shift keying (ASK), or (in the analog case) of phase modulation (PM) and amplitude modulation. In the digital QAM case, a finite number of at least two phases and at least two amplitudes are used. PSK modulators are often designed using the QAM principle, but are not considered as QAM since the amplitude of the modulated carrier signal is constant. QAM is used extensively as a modulation scheme for digital telecommunication systems. Spectral efficiencies of 6 bits/s/Hz can be achieved with QAM.

The **bit error rate** or **bit error ratio** (**BER**) is the number of bit errors divided by the total number of transferred bits during a studied time interval. BER is a unit less performance measure, often expressed as a percentage.

TIME DOMAIN FOR AN 8 QAM SIGNAL



MATLAB PROGRAM:-

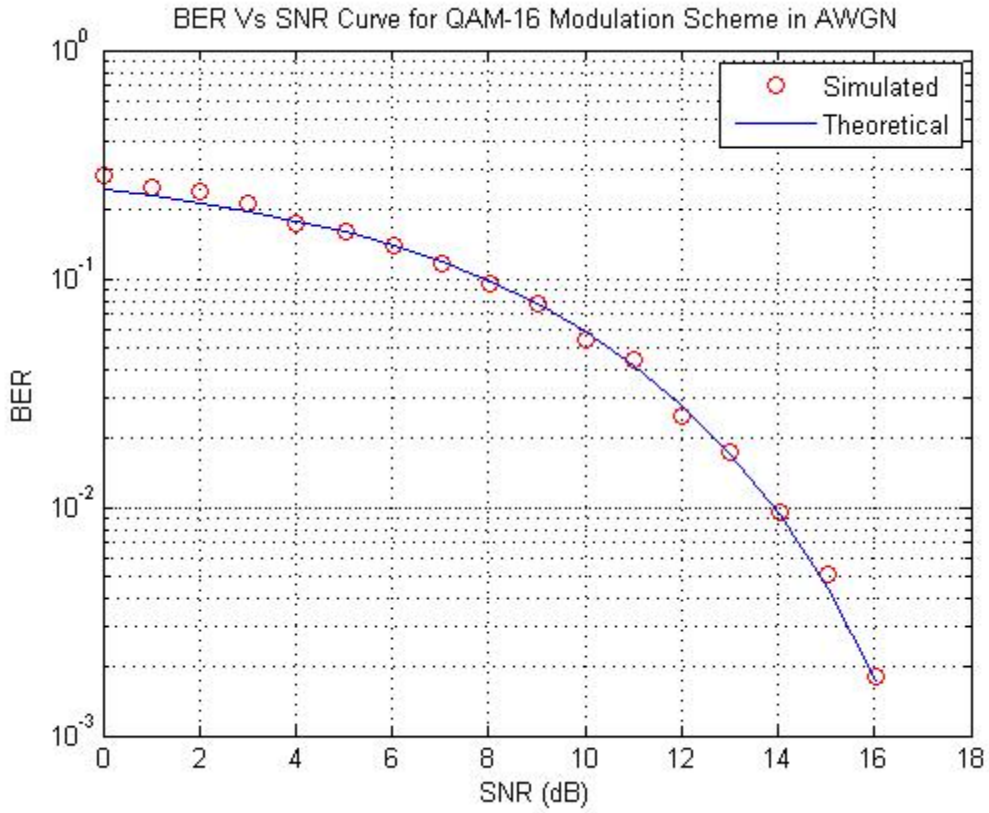
```
clear all;
close all;

format long;

bit_count = 4*1000;

Eb_No = -6: 1: 10;
SNR = Eb_No + 10*log10(4);

for aa = 1: 1: length(SNR)

   T_Errors = 0;
   T_bits = 0;
   while T_Errors < 100


      uncoded_bits  = round(rand(1,bit_count));

      B = reshape(uncoded_bits,4,length(uncoded_bits)/4);
      B1 = B(1,:);
      B2 = B(2,:);
      B3 = B(3,:);
      B4 = B(4,:);

      a = sqrt(1/10);
      tx = a*(-2*(B3-0.5).*(3-2*B4)-j*2*(B1-0.5).*(3-2*B2));
      N0 = 1/10^(SNR(aa)/10)
      rx = tx + sqrt(N0/2)*(randn(1,length(tx))+i*randn(1,length(tx)));

      a = 1/sqrt(10);

      B5 = imag(rx)<0;
      B6 = (imag(rx)<2*a) & (imag(rx)>-2*a);
      B7 = real(rx)<0;
      B8 = (real(rx)<2*a) & (real(rx)>-2*a);

      temp = [B5;B6;B7;B8];
      B_hat = reshape(temp,1,4*length(temp));

      diff =  uncoded_bits - B_hat ;
      T_Errors = T_Errors + sum(abs(diff));
      T_bits = T_bits + length(uncoded_bits);

   end
   BER(aa) = T_Errors / T_bits;
   disp(sprintf('bit error probability = %f',BER(aa)));

   figure;
   grid on;
   plot(rx,'x');
```

```
    xlabel('Inphase Component');
    ylabel('Quadrature Component');
    Title('Constellation of Transmitted Symbols');

end

figure(1);
semilogy(SNR,BER,'or');
hold on;
grid on
title('BER Vs SNR Curve for QAM-16 Modulation Scheme in AWGN');
xlabel('SNR (dB)'); ylabel('BER')

figure(1);
theoryBer = (1/4)*3/2*erfc(sqrt(4*0.1*(10.^(Eb_No/10))));
semilogy(SNR,theoryBer);
legend('Simulated','Theoretical');
```

OBSERVATION:-

Output data:-

```
bit error probability = 0.248750
bit error probability = 0.240000
bit error probability = 0.211250
bit error probability = 0.175750
bit error probability = 0.161250
bit error probability = 0.139250
bit error probability = 0.115750
bit error probability = 0.094250
bit error probability = 0.077000
bit error probability = 0.054500
bit error probability = 0.044250
bit error probability = 0.025000
bit error probability = 0.017375
bit error probability = 0.009500
bit error probability = 0.005100
bit error probability = 0.001821
```


BER Vs SNR Curve for QAM-16 Modulation Scheme in AWGN
Simulated
Theoretical
BER
SNR (dB)
10^0
10^-1
10^-2
10^-3
0
2
4
6
8
10
12
14
16
18

EXPERIMENT 11

Beam forming QAM

AIM:-

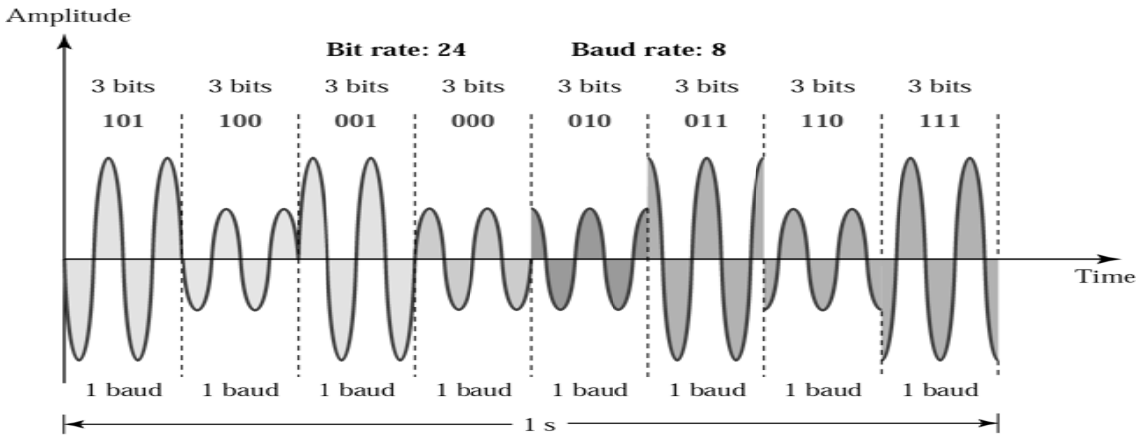
To plot the wave forms for beam forming QAM using MATLAB.

THEORY:-

Quadrature amplitude modulation (QAM) is both an analog and a digital modulation scheme. It conveys two analog message signals, or two digital bit streams, by changing (modulating) the amplitudes of two carrier waves, using the amplitude-shift keying (ASK) digital modulation scheme or amplitude modulation (AM) analog modulation scheme. The two carrier waves, usually sinusoids, are out of phase with each other by 90° and are thus called quadrature carriers or quadrature components — hence the name of the scheme. The modulated waves are summed, and the resulting waveform is a combination of both phase-shift keying (PSK) and amplitude-shift keying (ASK), or (in the analog case) of phase modulation (PM) and amplitude modulation. In the digital QAM case, a finite number of at least two phases and at least two amplitudes are used. PSK modulators are often designed using the QAM principle, but are not considered as QAM since the amplitude of the modulated carrier signal is constant. QAM is used extensively as a modulation scheme for digital telecommunication systems. Spectral efficiencies of 6 bits/s/Hz can be achieved with QAM.

Beamforming is a signal processing technique used in sensor arrays for directional signal transmission or reception. This is achieved by combining elements in the array in a way where signals at particular angles experience constructive interference and while others experience destructive interference. Beamforming can be used at both the transmitting and receiving ends in order to achieve spatial selectivity. The improvement compared with an omnidirectional reception/transmission is known as the receive/transmit gain (or loss).

TIME DOMAIN FOR AN 8 QAM SIGNAL

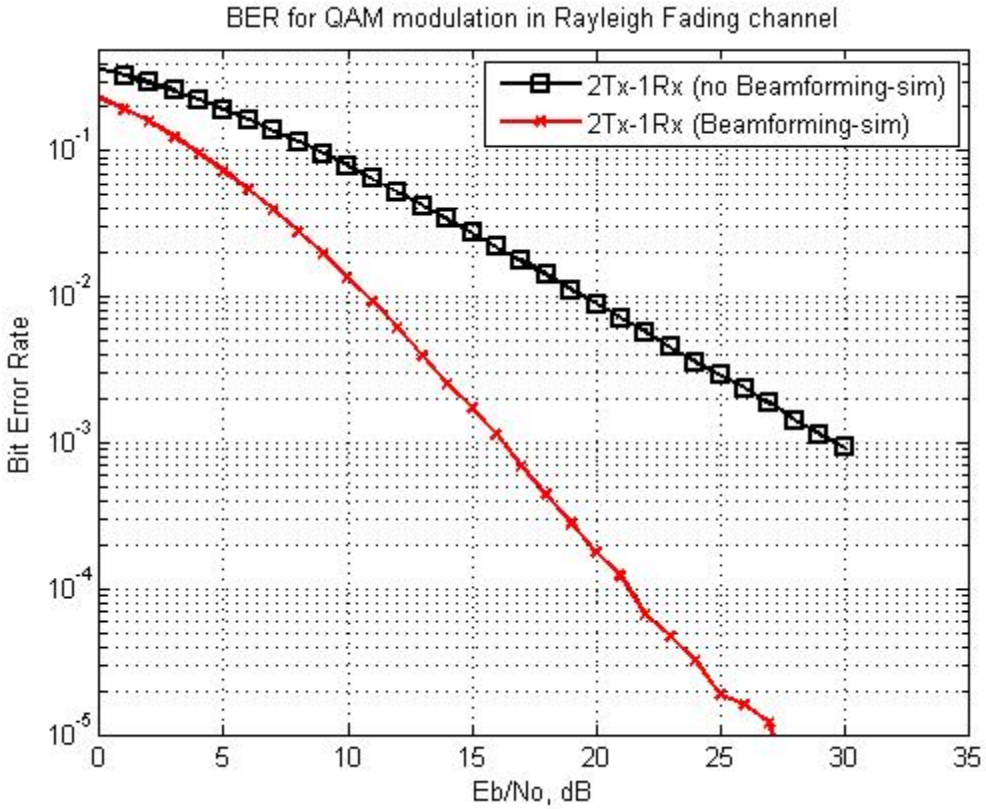


MATLAB PROGRAM:-

```
clear
N = 10^6

nTx = 2;

M=4;
k = sqrt(1/((2/3)*(M-1)));

m = [1:sqrt(M)/2];
alphaMqam = [-(2*m-1) 2*m-1];

Eb_N0_dB = [0:30];

ipHat1 = zeros(1,N);

ipHat2 = zeros(1,N);
for ii = 1:length(Eb_N0_dB)

    ip = randsrc(1,N,alphaMqam) + j*randsrc(1,N,alphaMqam);
    s = k*ip; % normalization of energy to 1
    n = 1/sqrt(2)*[randn(1,N) + j*randn(1,N)]; % white guassian noise, 0dB variance
    h = 1/sqrt(2)*[randn(nTx,N) + j*randn(nTx,N)]; % Rayleigh channel
    sr = (1/sqrt(nTx))*kron(ones(nTx,1),s);
    % Channel and noise Noise addition
   h_tx = h.*exp(-j*angle(h));
    y1 = sum(h.*sr,1) + 10^(-Eb_N0_dB(ii)/20)*n;
   y2 = sum(h_tx.*sr,1) + 10^(-Eb_N0_dB(ii)/20)*n;

y1Hat = y1./sum(h,1);
   y2Hat = y2./sum(h_tx,1);
    % demodulation
    y_re1 = real(y1Hat)/k; % real part
    y_im1 = imag(y1Hat)/k;
    y_re2 = real(y2Hat)/k; % real part
    y_im2 = imag(y2Hat)/k; % imaginary part


    ipHat_re1 = 2*floor(y_re1/2)+1;

    ipHat_re1(find(ipHat_re1>max(alphaMqam))) = max(alphaMqam);
    ipHat_re1(find(ipHat_re1<min(alphaMqam))) = min(alphaMqam);

     ipHat_re2 = 2*floor(y_re2/2)+1;

    ipHat_re2(find(ipHat_re2>max(alphaMqam))) = max(alphaMqam);
    ipHat_re2(find(ipHat_re2<min(alphaMqam))) = min(alphaMqam);


    ipHat_im1 = 2*floor(y_im1/2)+1;
    ipHat_im1(find(ipHat_im1>max(alphaMqam))) = max(alphaMqam);
    ipHat_im1(find(ipHat_im1<min(alphaMqam))) = min(alphaMqam);
```

```
    ipHat_im2 = 2*floor(y_im2/2)+1;
    ipHat_im2(find(ipHat_im2>max(alphaMqam))) = max(alphaMqam);
    ipHat_im2(find(ipHat_im2<min(alphaMqam))) = min(alphaMqam);


    ipHat1 = ipHat_re1 + j*ipHat_im1;
    nErr1(ii) = size(find([ip- ipHat1]),2); % counting the number of errors

    ipHat2 = ipHat_re2 + j*ipHat_im2;
    nErr2(ii) = size(find([ip- ipHat2]),2); % counting the number of errors

end

simBer1 = nErr1/N; % simulated ber (no beam forming)
simBer2 = nErr2/N; % simulated ber (with beam forming)

theoryBerAWGN = 0.5*erfc(sqrt(10.^(Eb_N0_dB/10))); % theoretical ber
EbN0Lin = 10.^(Eb_N0_dB/10);
theoryBer = 0.5.*(1-sqrt(EbN0Lin./(EbN0Lin+1)));
p = 1/2 - 1/2*(1+1./EbN0Lin).^(-1/2);
theoryBer_nRx2 = p.^2.*(1+2*(1-p));


close all
figure
semilogy(Eb_N0_dB,simBer1,'ks-','LineWidth',2);

hold on

semilogy(Eb_N0_dB,simBer2,'rx-','LineWidth',2);
axis([0 35 10^-5 0.5])
grid on
legend('2Tx-1Rx (no Beamforming-sim)','2Tx-1Rx (Beamforming-sim)');

xlabel('Eb/No, dB');
ylabel('Bit Error Rate');
title('BER for QAM modulation in Rayleigh Fading channel');
```

OBSERVATION:-



EXPERIMENT 12

16-QAM with BER & Rayleigh fading

AIM:-

To plot the wave forms 16-QAM signal subjected to BER and Rayleigh fading using MATLAB.

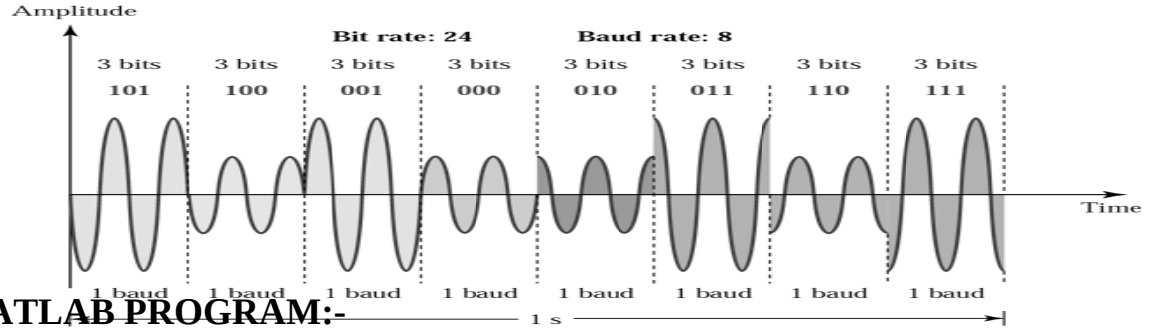
THEORY:-

Quadrature amplitude modulation (QAM) is both an analog and a digital modulation scheme. It conveys two analog message signals, or two digital bit streams, by changing (modulating) the amplitudes of two carrier waves, using the amplitude-shift keying (ASK) digital modulation scheme or amplitude modulation (AM) analog modulation scheme. The two carrier waves, usually sinusoids, are out of phase with each other by 90° and are thus called quadrature carriers or quadrature components — hence the name of the scheme. The modulated waves are summed, and the resulting waveform is a combination of both phase-shift keying (PSK) and amplitude-shift keying (ASK), or (in the analog case) of phase modulation (PM) and amplitude modulation. In the digital QAM case, a finite number of at least two phases and at least two amplitudes are used. PSK modulators are often designed using the QAM principle, but are not considered as QAM since the amplitude of the modulated carrier signal is constant. QAM is used extensively as a modulation scheme for digital telecommunication systems. Spectral efficiencies of 6 bits/s/Hz can be achieved with QAM.

The **bit error rate** or **bit error ratio** (**BER**) is the number of bit errors divided by the total number of transferred bits during a studied time interval. BER is a unit less performance measure, often expressed as a percentage.

Rayleigh fading is a statistical model for the effect of a propagation environment on a radio signal, such as that used by wireless devices.Rayleigh fading models assume that the magnitude of a signal that has passed through such a transmission medium (also called a communications channel) will vary randomly, or fade,according to a Rayleigh distribution — the radial component of the sum of two uncorrelated Gaussian random variables.

TIME DOMAIN FOR AN 8 QAM SIGNAL



MATLAB PROGRAM:-

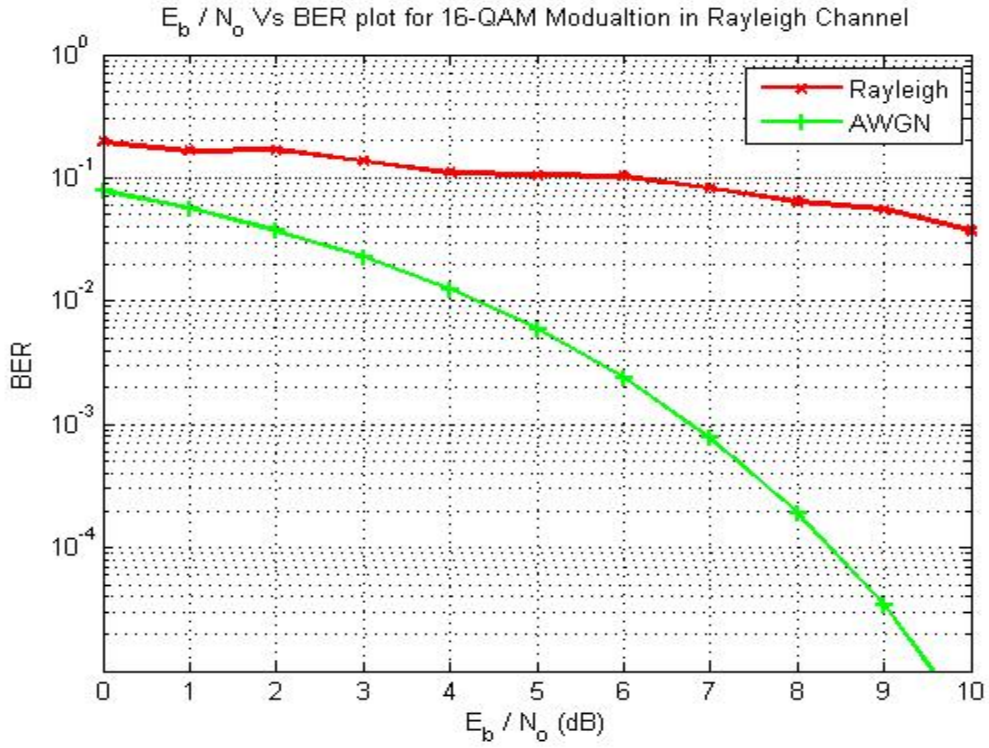
```
clear all;
close all;

format long;

bit_count = 1*1000;

Eb_No = 0: 1: 10;
SNR = Eb_No + 10*log10(4);

for aa = 1: 1: length(SNR)

   T_Errors = 0;
   T_bits = 0;

   while T_Errors < 100

      uncoded_bits  = round(rand(1,bit_count));

      B = reshape(uncoded_bits,4,length(uncoded_bits)/4);
      B1 = B(1,:);
      B2 = B(2,:);
      B3 = B(3,:);
      B4 = B(4,:);

      a = sqrt(1/10);

      tx = a*(-2*(B3-0.5).*(3-2*B4)-j*2*(B1-0.5).*(3-2*B2));

      ray = sqrt((1/2)*((randn(1,length(tx))).^2+(randn(1,length(tx))).^2));

      rx = tx.*ray;

      N0 = 1/10^(SNR(aa)/10);
      rx = rx + sqrt(N0/2)*(randn(1,length(tx))+1i*randn(1,length(tx)));

      rx = rx./ray;

      a = 1/sqrt(10);

      B5 = imag(rx)<0;
      B6 = (imag(rx)<2*a) & (imag(rx)>-2*a);
      B7 = real(rx)<0;
      B8 = (real(rx)<2*a) & (real(rx)>-2*a);

      temp = [B5;B6;B7;B8];
      B_hat = reshape(temp,1,4*length(temp));

      diff =  uncoded_bits - B_hat ;
      T_Errors = T_Errors + sum(abs(diff));
      T_bits = T_bits + length(uncoded_bits);
```

```
    end

    BER(aa) = T_Errors / T_bits;
    disp(sprintf('bit error probability = %f',BER(aa)));


    figure;
    grid on;
    plot(rx,'x');
    xlabel('Inphase Component');
    ylabel('Quadrature Component');
    Title(['Constellation of Transmitted Symbols for SNR =',num2str(SNR(aa))]);


end

figure(1);
semilogy(Eb_No,BER,'xr-','Linewidth',2);
hold on;
xlabel('E_b / N_o (dB)');
ylabel('BER');
title('E_b / N_o Vs BER plot for 16-QAM Modualtion in Rayleigh Channel');

figure(1);
theoryBerAWGN = 0.5.*erfc(sqrt((10.^(Eb_No/10))));
semilogy(Eb_No,theoryBerAWGN,'g-+','Linewidth',2);
grid on;
legend('Rayleigh', 'AWGN');
axis([Eb_No(1,1) Eb_No(end) 0.00001 1]);
```

OBSERVATION:-

Output data:-

```
bit error probability = 0.195000
bit error probability = 0.165000
bit error probability = 0.169000
bit error probability = 0.136000
bit error probability = 0.110000
bit error probability = 0.105000
bit error probability = 0.103000
bit error probability = 0.082500
bit error probability = 0.064000
bit error probability = 0.055500
bit error probability = 0.037333
```


Eb / No Vs BER plot for 16-QAM Modualtion in Rayleigh Channel
Rayleigh
AWGN
BER
Eb / No (dB)
10^0
10^-1
10^-2
10^-3
10^-4
0
1
2
3
4
5
6
7
8
9
10