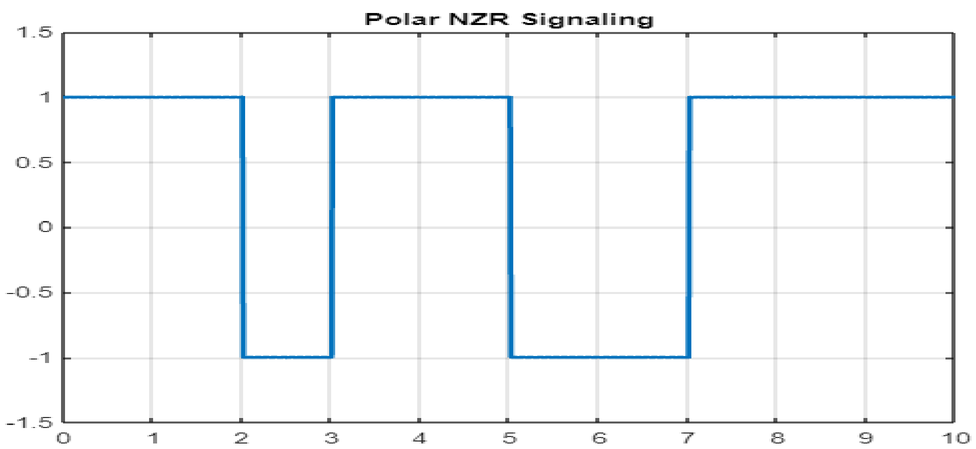
Source Code in MATLAB:

```
%Polar Non Return to Zero Line Coding
clc;
clear all;
close all;

N=10;                          % Number of bits
n=randi([0,1],1,N)             % Random Bit Generation
%Mapping Function
for m=1:N
 if n(m)==1
    nn(m)=1;
 else
    nn(m)=-1;                  % Assign -1 for Polar
 end
end
nn

%Signal Shaping
i=1;
t=0:0.01:length(n);            %Time Duration set up for a single binary bit
for j=1:length(t)
 if t(j)<=i
    y(j)=nn(i);                % Assign value from the mapping function
 else
    y(j)=nn(i);
    i=i+1;                     % Binaty input data index increament
 end
end
plot(t,y, 'linewidth',2);      % Linewidth 2 for clear visualization
axis([0,N,-1.5,1.5]);          % Axis set-up
grid on;
title("Polar NZR Signaling");
```

Output:

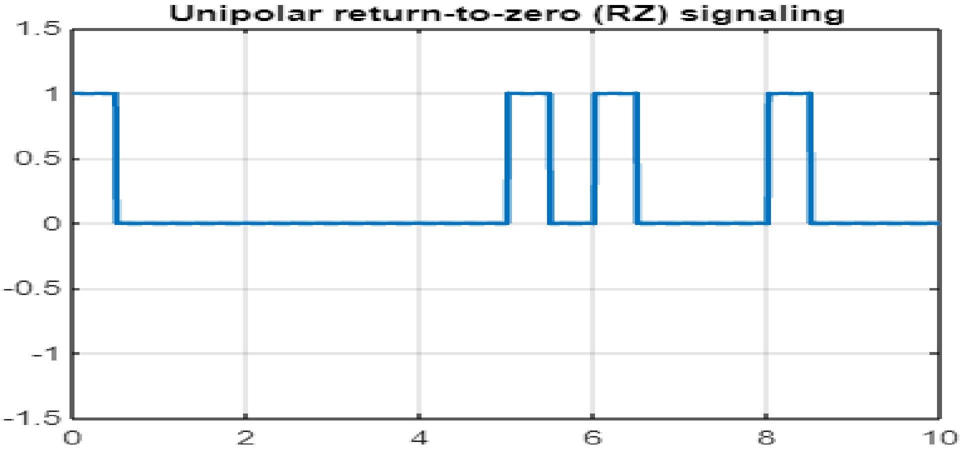


Source Code in MATLAB:

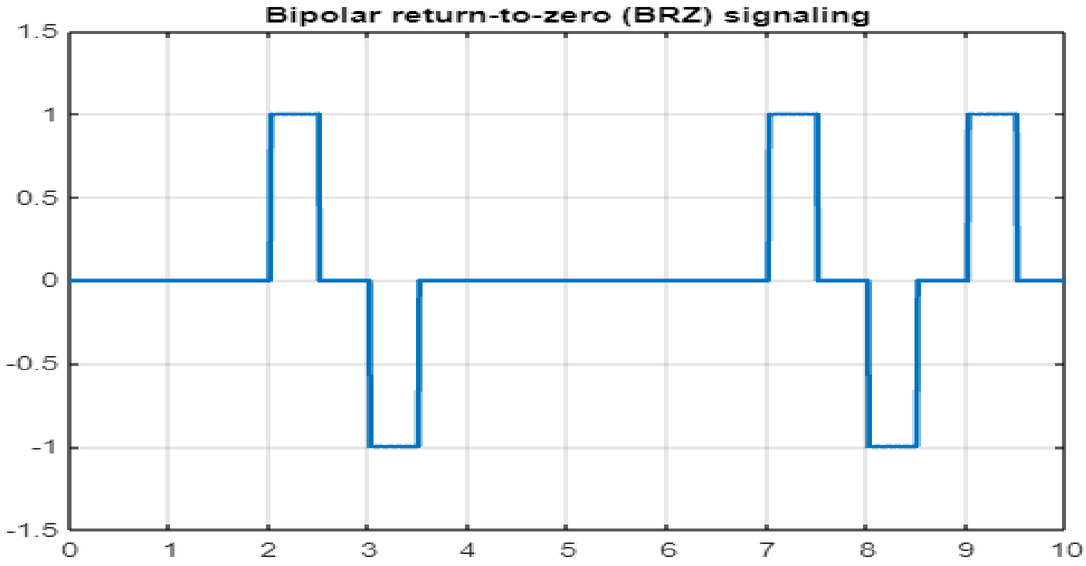
```
%Unipolar Return To Zero (RZ) line coding
clc;
clear all;
close all;
N=10;                          % Number of bits
n=randi([0,1],1,N)        % Random Bit Generation

%Unipolar RZ Pulse Shaping
i=1;
a=0;                           % Initial value for first half cycle
b=0.5;                         % Initial value for second half cycle
t=0:0.01:length(n);       %Time Duration set up for a single binary bit
for j=1:length(t)
 if t(j)>=a && t(j)<=b       % Condition for the first half cycle
    y(j)=n(i);                  % Assign first 50 values for 1st half cycle
 elseif t(j)>b && t(j)<=i   % Condition for second half cycle
    y(j)=0;                    %set all values 0 for 2nd half cycle
 else
     i=i+1;                   % Binaty input data index increament
     a=a+1;                   % Initial value for first half cycleIncrement
     b=b+1;                   % Initial value for second half cycle Increment
 end
end
plot(t,y,'lineWidth', 2);   % Linewidth 2 for clear visualization
axis([0,N,-1.5,1.5]);       % Axis set-up
grid on;
title('Unipolar return-to-zero (RZ) signaling');
```

Output:



Output:

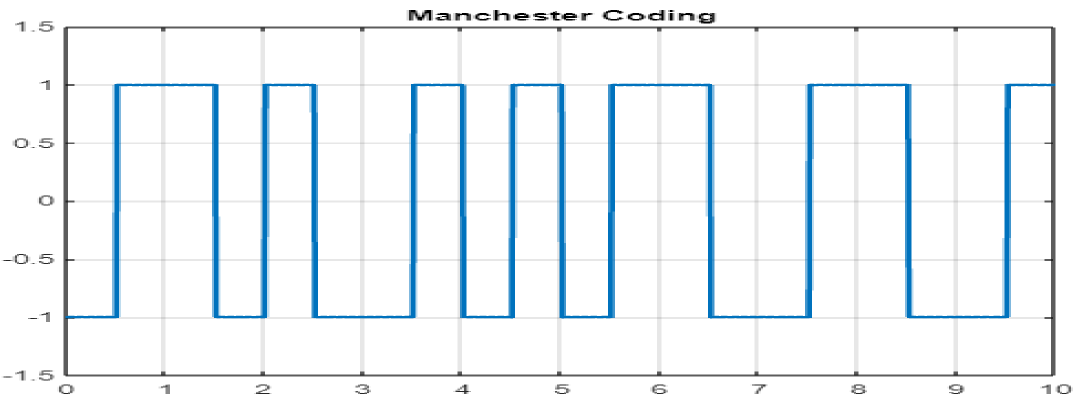


Bipolar return-to-zero (BRZ) signaling
1.5
1
0.5
0
-0.5
-1
-1.5
0
1
2
3
4
5
6
7
8
9
10

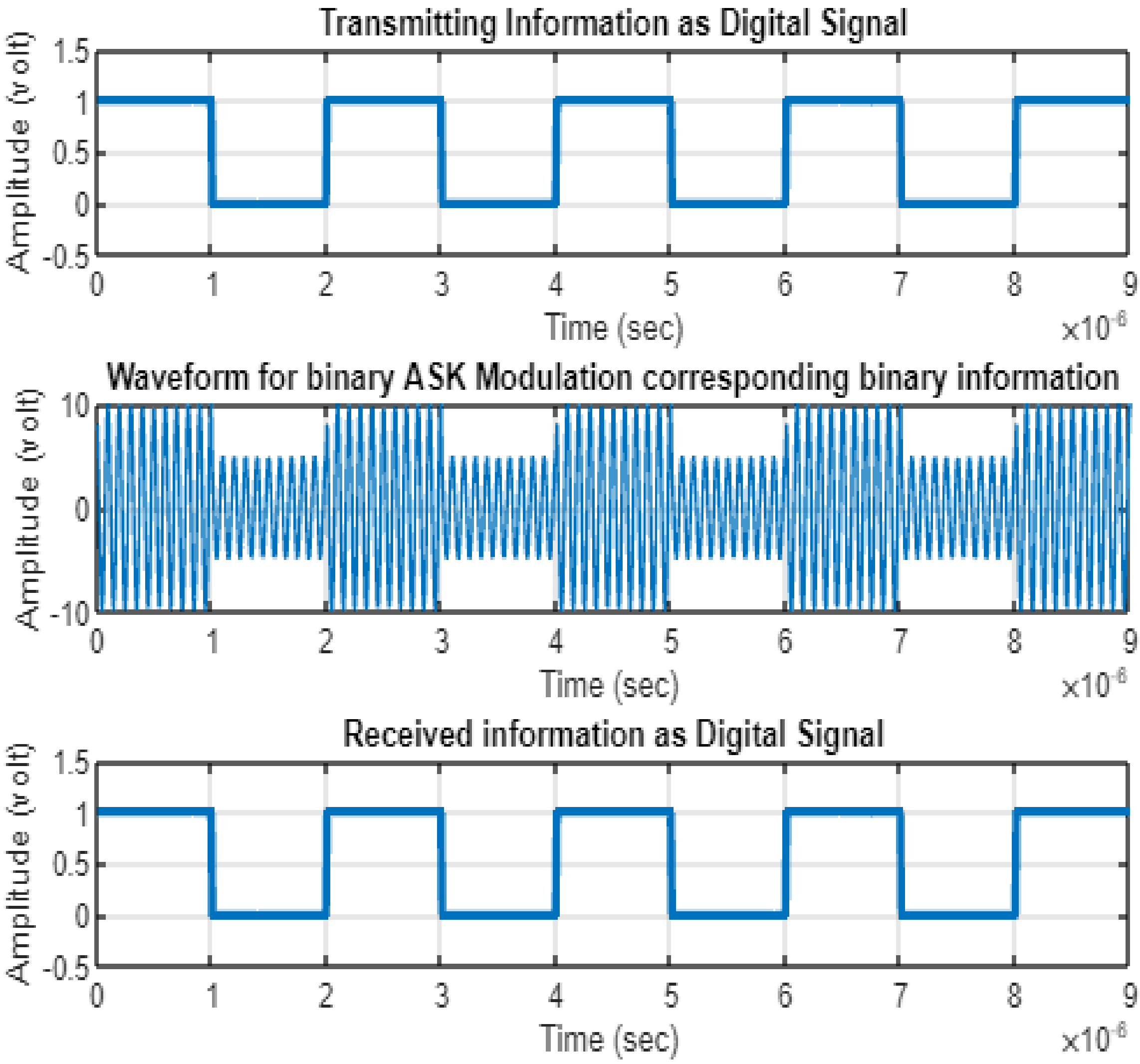
Source Code in MATLAB:

```
%Split Phase-Manchester Coding
clc;
clear all;
close all;
N=10;                          % Number of bits
n=randi([0,1],1,N)          % Random Bit Generation
%Binary to Manchester Conversion
nnn=[];
for m=1:N
   if n(m)==1
       nn=[1 -1];
   else
       nn=[-1 1];
   end
   nnn=[nnn nn];
   disp(nnn);
end
nnn;     %nnn = [-1     1     1    -1     1    -1    -1     1 1    -1    -1     1    -1     1     1    -1]
%Manchester Coding Pulse Shaping
i=1;
l=0.5;                              % Initial value for first half cycle
t=0:0.01:length(n);           % Time Duration set up for a single binary bit
for j=1:length(t)
     if t(j)<=l
        y(j)=nnn(i);               % Assign First 50 values for 1st half cycle
     else
         y(j)=nnn(i);             % Assign second 50 values for 1st half cycle
         i=i+1;                     % Binaty input data index increament
         l=l+0.5;                  % Initial value for each half cycle Increment
     end
end
plot(t,y,'lineWidth', 2);       % Linewidth 2 for clear visualization
axis([0,N,-1.5,1.5]);           % Axis set-up
grid on;
title('Manchester Coding');
```

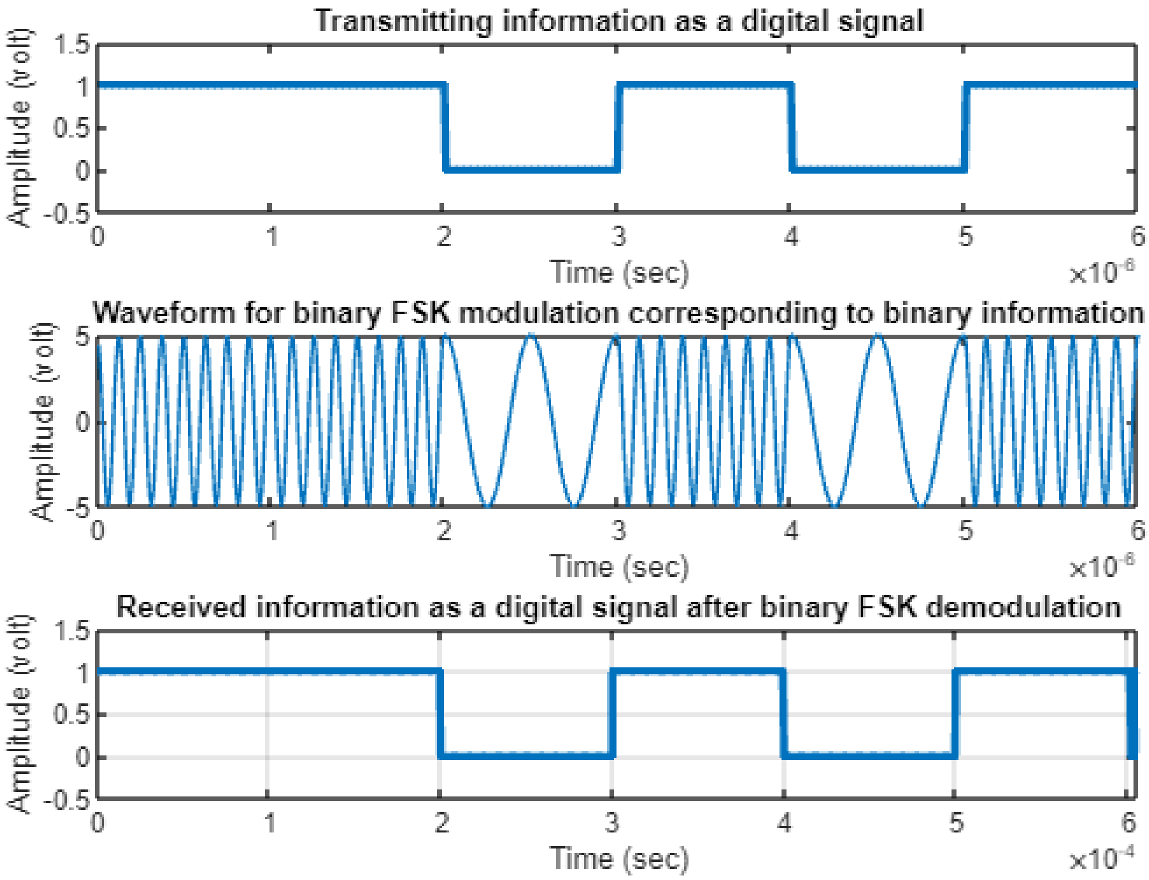
Output:



Output:

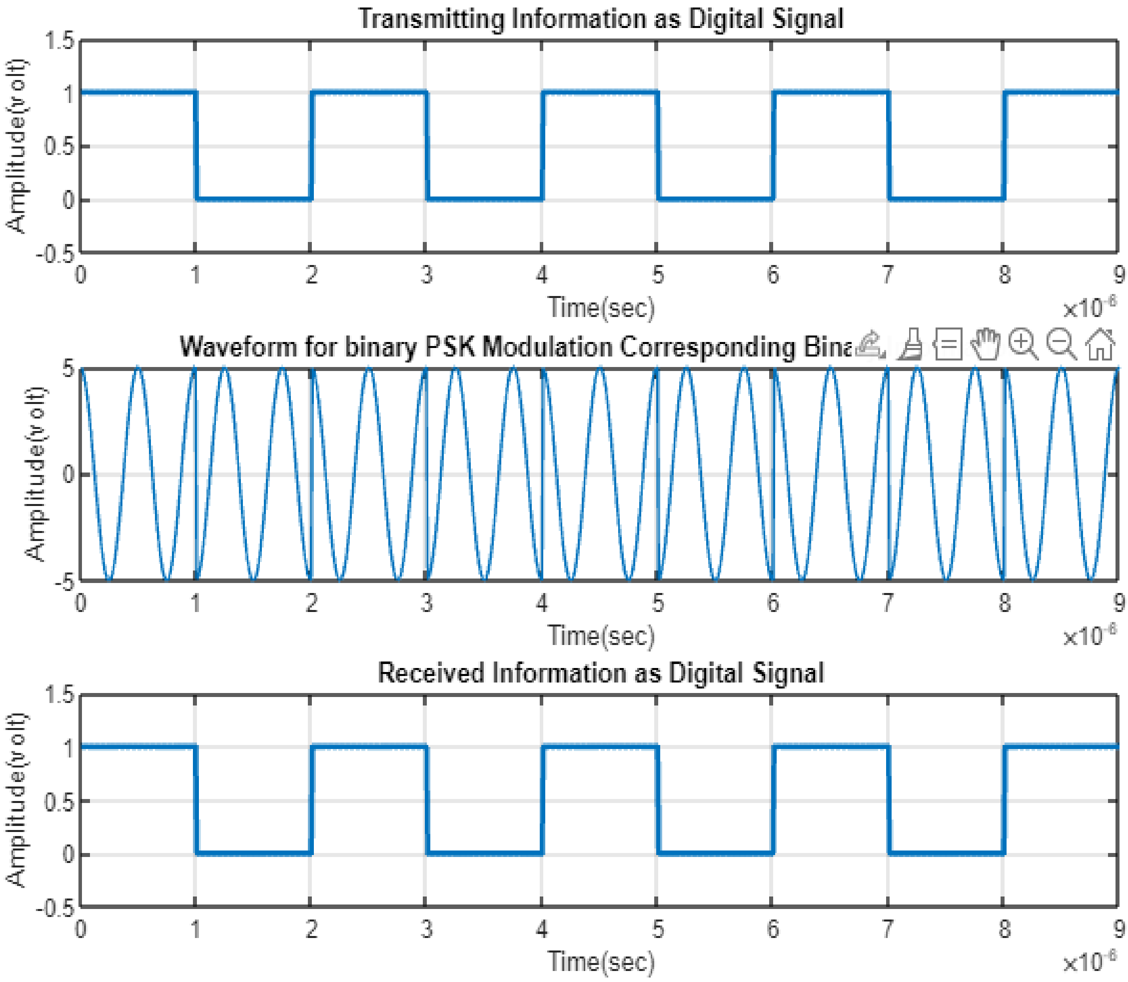

Transmitting Information as Digital Signal
Amplitude (volt)
1.5
1
0.5
0
-0.5
0
1
2
3
4
5
6
7
8
9
Time (sec)
×10-6
Waveform for binary ASK Modulation corresponding binary information
Amplitude (volt)
10
0
-10
0
1
2
3
4
5
6
7
8
9
Time (sec)
×10-6
Received information as Digital Signal
Amplitude (volt)
1.5
1
0.5
0
-0.5
0
1
2
3
4
5
6
7
8
9
Time (sec)
×10-6

Output:



Transmitting information as a digital signal
Amplitude (volt)
Time (sec)
×10^-6
Waveform for binary FSK modulation corresponding to binary information
Amplitude (volt)
Time (sec)
×10^-6
Received information as a digital signal after binary FSK demodulation
Amplitude (volt)
Time (sec)
×10^-4

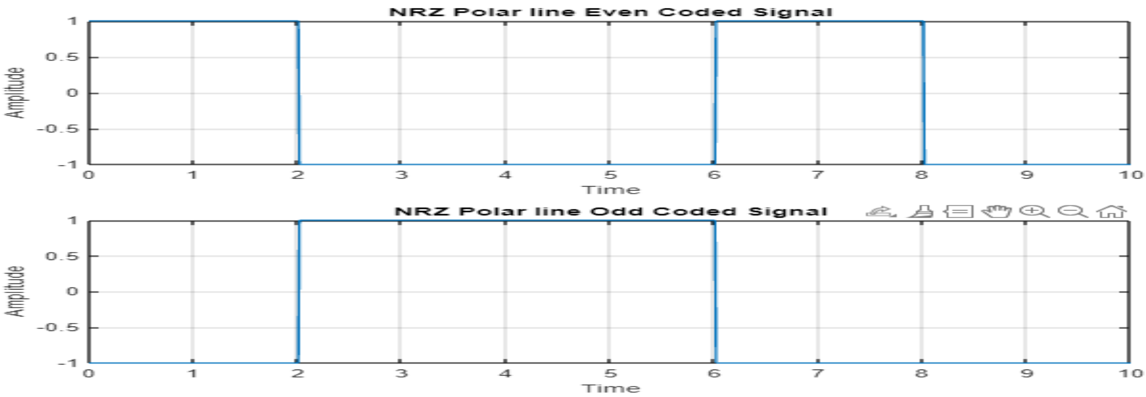
Output:

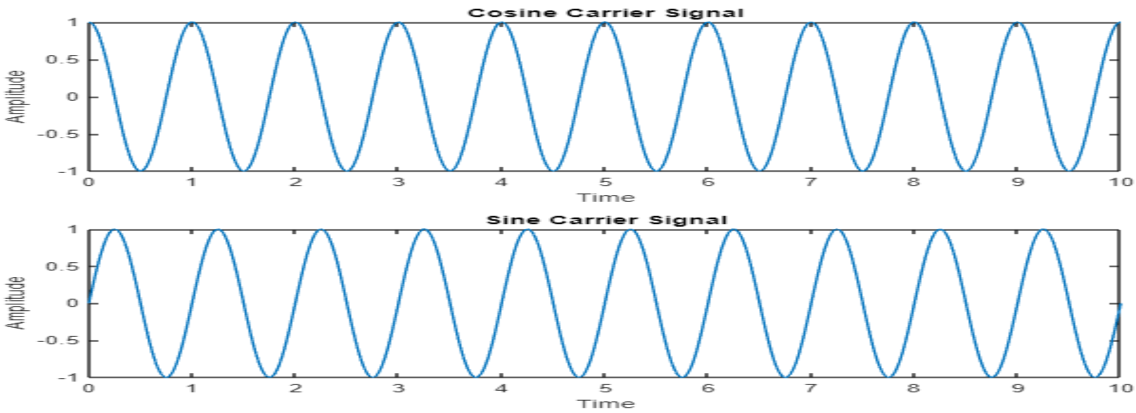


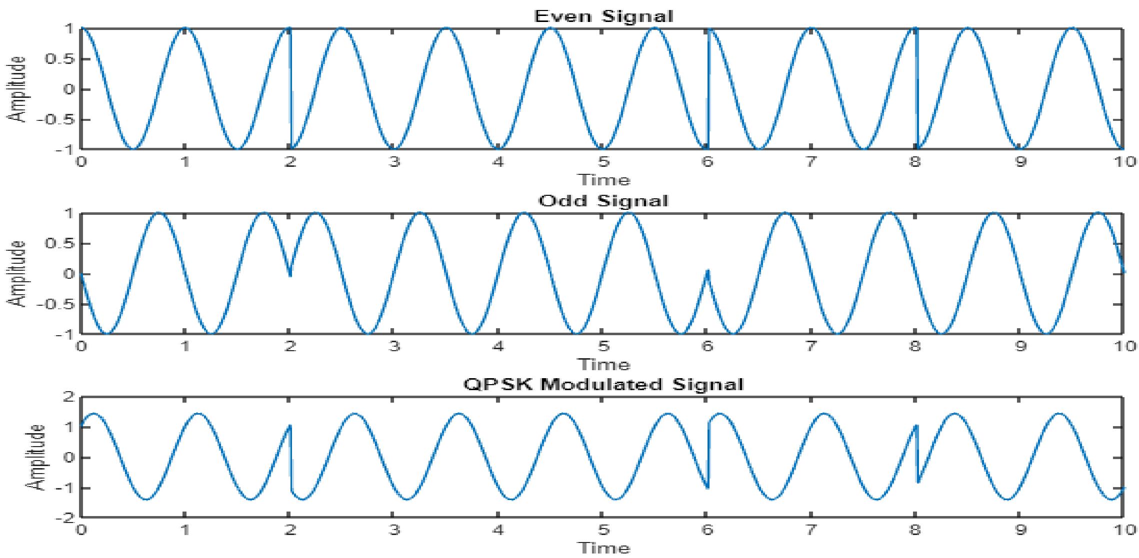
Transmitting Information as Digital Signal
Amplitude(volt)
1.5
1
0.5
0
-0.5
0 1 2 3 4 5 6 7 8 9
Time(sec)
×10^-6
Waveform for binary PSK Modulation Corresponding Bina
Amplitude(volt)
5
0
-5
0 1 2 3 4 5 6 7 8 9
Time(sec)
×10^-6
Received Information as Digital Signal
Amplitude(volt)
1.5
1
0.5
0
-0.5
0 1 2 3 4 5 6 7 8 9
Time(sec)
×10^-6

Output:

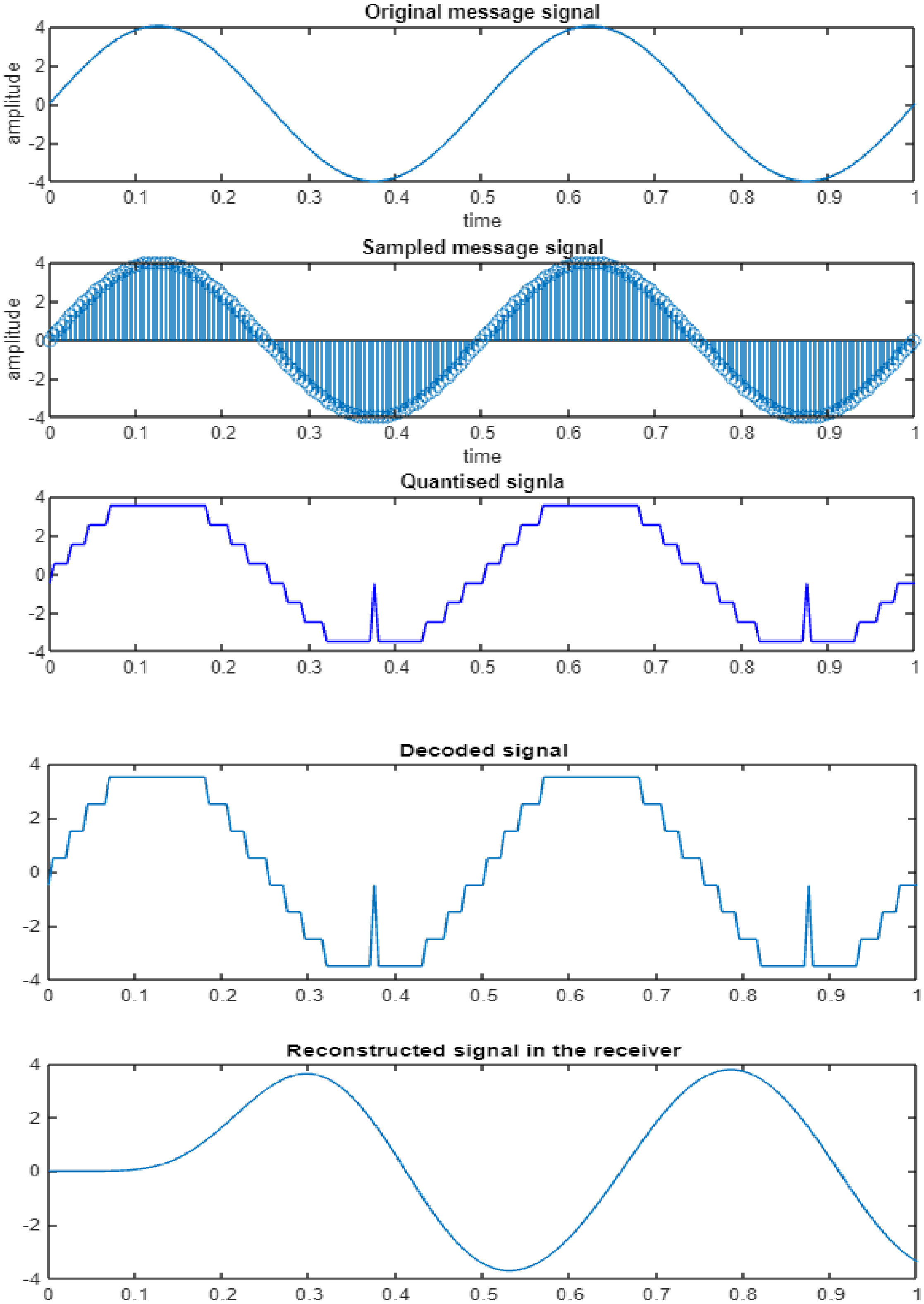
```
Binary Data x =     1     0     0     1     0     1     1     0     0     0
```







Output:


Original message signal
amplitude
time
Sampled message signal
amplitude
time
Quantised signla
Decoded signal
Reconstructed signal in the receiver

Source Code in MATLAB:

```
clc;
clear all;
close all;
fc=100;                              % Carrier Frequency
fm=fc/10;
fs=100*fc;                            % Sample Frequency
t=0:1/fs:4/fm;

% Message Signal
mt=cos(2*pi*fm*t);                    % Message signal Generating
subplot(4,1,1);
plot(t,mt);
grid on;
title('Message Signal');
xlabel('Time Period');
ylabel('Amplitude');

% Carrier Signal
ct=0.5*square(2*pi*fc*t)+0.5;        % Carrier signal Generating
subplot(4,1,2);
plot(t,ct);
grid on;
title('Carrier Signal');
xlabel('Time Period');
ylabel('Amplitude');

% Modulated Signal of double side band
st=mt.*ct;                           % Modulated signal Generating
subplot(4,1,3);
grid on;
plot(t,st);
title('Modulated Signal of Double Side Band');
xlabel('Time Period');
ylabel('Amplitude');

%single sided PAM
tt=[ ];                              % PAM signal Generating
for i=1:length(st)
    if st(i)==0
        tt=[tt,st(i)];
    else
        tt=[tt,st(i)+2];
    end
end
subplot(4,1,4);
plot(t,tt);
title('PAM Signal of Single Side Band');
xlabel('Time Period');
grid on;
ylabel('Amplitude');

%demdulated
figure(2)
```

```
d=st.*ct;
filter=fir1(200,fm/fs,'low');
original_t_signal=conv(filter,d);
t1=0:1/(length(original_t_signal)-1):1;
subplot(4,1,1);
plot(t1,original_t_signal);
title('Demodulated Signal');
xlabel('Time Period');
ylabel('Amplitude');
```

Output: