Analog and Digital Communication Theory Project

Simulation of QPSK and BFSK with their respective BER simulations and bandwidth calculations

Programme : B.Tech - ICT (4th semester)

Course : Analog and Digital Communication

SEAS, Ahmedabad University

[Team Members]

Enrollment No.	Name
AU1841057	Mayankkumar Tank
AU1841073	Meet Akbari
AU1841076	Rahul Chocha
AU1841109	Jeet Karia

[Table of Contents]

[Synopsys]

- **QPSK:** Also known quaternary PSK (Phase Shift Keying) quadriphase PSK, 4-PSK, or 4-QAM.

- It is the modulation technique which has varying phases and all other physical quantities are constant which includes Amplitude, frequency.

- **BFSK:** It is the simplest form of FSK (Frequency Shift Keying).

- BFSK uses a pair of discrete frequencies to transmit binary (0s and 1s) information. With this scheme, the "1" is called the mark frequency and the "0" is called the space frequency.

- **BER:** Bit error probability is the probability that information received at the receiver end is incorrect (bitwise).

- **Bandwidth:** Required Bandwidth for particular System.

[QPSK Description]

- Quadrature Phase Shift Keying (**QPSK**) is a form of Phase Shift Keying in which two bits are modulated at once. QPSK allows the signal to carry twice as much information with the same frequency of the carrier and same bandwidth. In QPSK carrier 1(In phase component - I signal) modulates the odd number of bits and carrier 2(Quadrature component - Q signal) modulates the even number of bits from the data sequence. In QPSK, Number of Points on the constellation is four. There are two possibilities. One choice is 0, pi/2, pi, 3pi/2. The second and most preferable choice is pi/4, 3pi/4, 5pi/4, 7pi/4.

- The formula of M'ary Phase shift keying given as,

$$s(t) = \sqrt{E} cos\theta \phi_1(t) - \sqrt{E} sin\theta \phi_2(t)$$

$$s(t) = x\phi_1(t) + y\phi_2(t)$$

- So,

$$x = \sqrt{E}\cos\theta$$

$$y = -\sqrt{E}\sin\theta$$

- Orthogonal signals are,

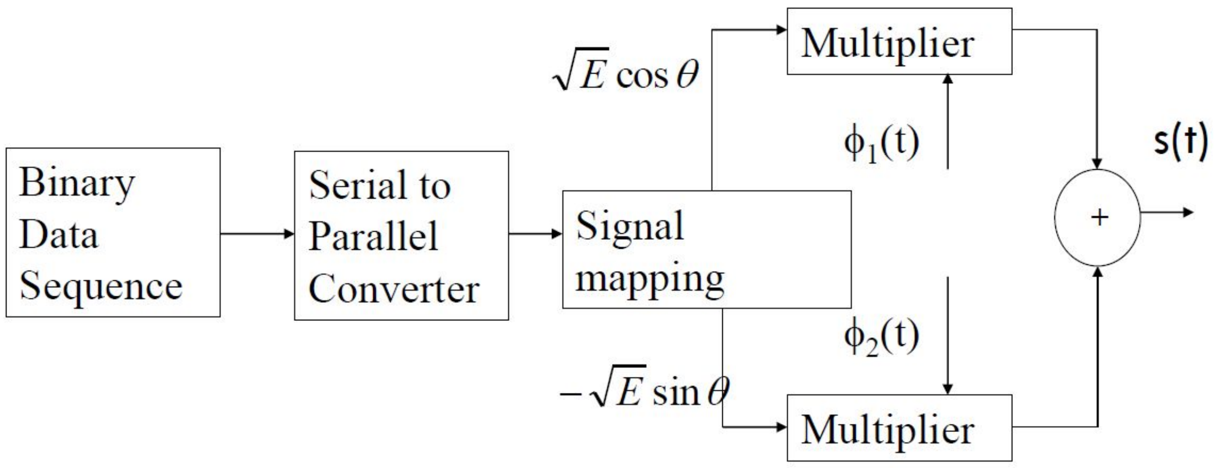
$$\phi_1(t) = \sqrt{\frac{2}{T}} cos(\omega_c t)$$

$$\phi_2(t) = \sqrt{\frac{2}{T}} sin(\omega_c t)$$

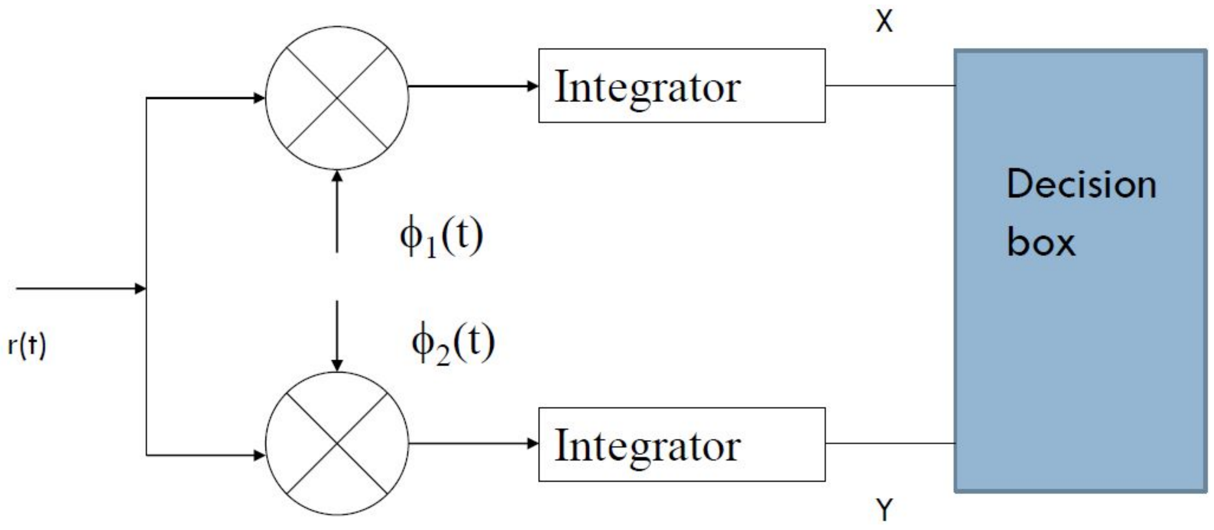
- **Constellation points:**

Input debits	Phase of QPSK signal	Co-ordinates of message signals	
		S1	S2
10	$\pi/4$	$\sqrt{E/2}$	$-\sqrt{E/2}$
00	$3\pi/4$	$-\sqrt{E/2}$	$-\sqrt{E/2}$
01	$5\pi/4$	$-\sqrt{E/2}$	$+\sqrt{E/2}$
11	$7\pi/4$	$+\sqrt{E/2}$	$+\sqrt{E/2}$

- **QPSK Transmitter:**



- **QPSK Receiver:**



- **Decision Box:**

X	Y	Data
$\sqrt{\frac{E}{2}}$	$-\sqrt{\frac{E}{2}}$	10
$-\sqrt{\frac{E}{2}}$	$-\sqrt{\frac{E}{2}}$	00
$-\sqrt{\frac{E}{2}}$	$\sqrt{\frac{E}{2}}$	01
$\sqrt{\frac{E}{2}}$	$\sqrt{\frac{E}{2}}$	11

[Algorithm]

Initialization commands

QPSK modulation

1. Generate quadrature carriers.
2. Start FOR loop
3. Generate binary data, message signal(bipolar form)
4. Multiply carrier 1 with odd bits of message signal and carrier 2 with even bits of message signal
5. Perform addition of odd and even modulated signals to get the QPSK modulated signal
6. Plot QPSK modulated signal.
7. End FOR loop.
8. Plot the binary data and carriers.

QPSK demodulation

1. Start FOR loop
2. Perform integration of QPSK modulated signal with quadrature carriers to get two decision variables x and y.
3. Make a decision on x and y to get demodulated binary data.
   If x>0and y>0, choose '11'. If x>0and y<0, choose '10'. If x<0and y>0, choose '01. If x<0and y<0, choose '00'.
4. End FOR loop
5. Plot demodulated data

QPSK Bandwidth

1. Input Roll-off factor.
2. input bit time or bit rate.

3. Apply formula $B = \frac{(1+r)R}{k}$

   Where R = bit rate

   k = 2 in QPSK

QPSK Bit error probability

1. 4Input amplitude
2. Calculate bit energy

$$E = \frac{A^2 T}{2}$$

3. Apply bit error probability formula

$$P = Q\left(\sqrt{\frac{2E}{N_0}}\right)$$

4. This is about the Theoretical Calculations.
5. For Practical Purpose, Compare the modulation bits and demodulated bits. Count the difference bits. Output probability is equal to difference bits/total bits.

[Matlab Simulation code of QPSK]

```
clc;
clear all;
close all;
```

% QPSK Modulation

```
%GENERATE QUADRATURE CARRIER SIGNAL
```

```
amplitude = 1; %from gui (input)
N0 = 1; %input
Tb=1; %bit time from input gui
t=0:(Tb/100):Tb;
fc=1;  %frequency of carrier can taken from gui

c1=amplitude*sqrt(2/Tb)*cos(2*pi*fc*t);
c2=amplitude*sqrt(2/Tb)*sin(2*pi*fc*t);

%bandwidth calculations
r = 0;  %input roll off factor
bandw = (1+r)/(Tb*2);
disp('Bandwidth : ');
disp(bandw);  %should be displayed on gui

%bit error probability
Energy = amplitude*amplitude*Tb/2;
BEP = qfunc(sqrt(Energy*2/N0));
disp('Bit Error Probability : ');
disp(BEP); %should be displayed on gui


%generate message signal
N=16;  %Number of bits you want to send input from gui
m=rand(1,N);
t1=0;t2=Tb;
for i=1:2:(N-1)
 t=[t1:(Tb/100):t2];
 if m(i)>0.5
 m(i)=1;
 m_s=ones(1,length(t));
 else
 m(i)=0;
 m_s=-1*ones(1,length(t));
 end
 %odd bits modulated signal
 odd_sig(i,:)=c1.*m_s;
 if m(i+1)>0.5
 m(i+1)=1;
 m_s=ones(1,length(t));
 else
 m(i+1)=0;
 m_s=-1*ones(1,length(t));
 end
 %even bits modulated signal
 even_sig(i,:)=c2.*m_s;
 %qpsk signal
 qpsk=odd_sig+even_sig;
 %Plot the QPSK modulated signal
 subplot(3,2,4);plot(t,qpsk(i,:));
 title('QPSK signal');xlabel('t---->');ylabel('s(t)');grid on; hold
on;
 t1=t1+(Tb+.01); t2=t2+(Tb+.01);
end
hold off
%Plot the binary data bits and carrier signal
subplot(3,2,1);stem(m);
title('binary data bits');xlabel('n---->');ylabel('b(n)');grid on;
subplot(3,2,2);plot(t,c1);
title('carrier signal-1');xlabel('t---->');ylabel('c1(t)');grid on;
subplot(3,2,3);plot(t,c2);
title('carrier signal-2');xlabel('t---->');ylabel('c2(t)');grid on;
```

% QPSK Demodulation

```
t1=0;
t2=Tb;
for i=1:N-1
t=[t1:(Tb/100):t2];
%correlator
x1=sum(c1.*qpsk(i,:));
x2=sum(c2.*qpsk(i,:));
%decision device
if (x1>0&&x2>0)
```

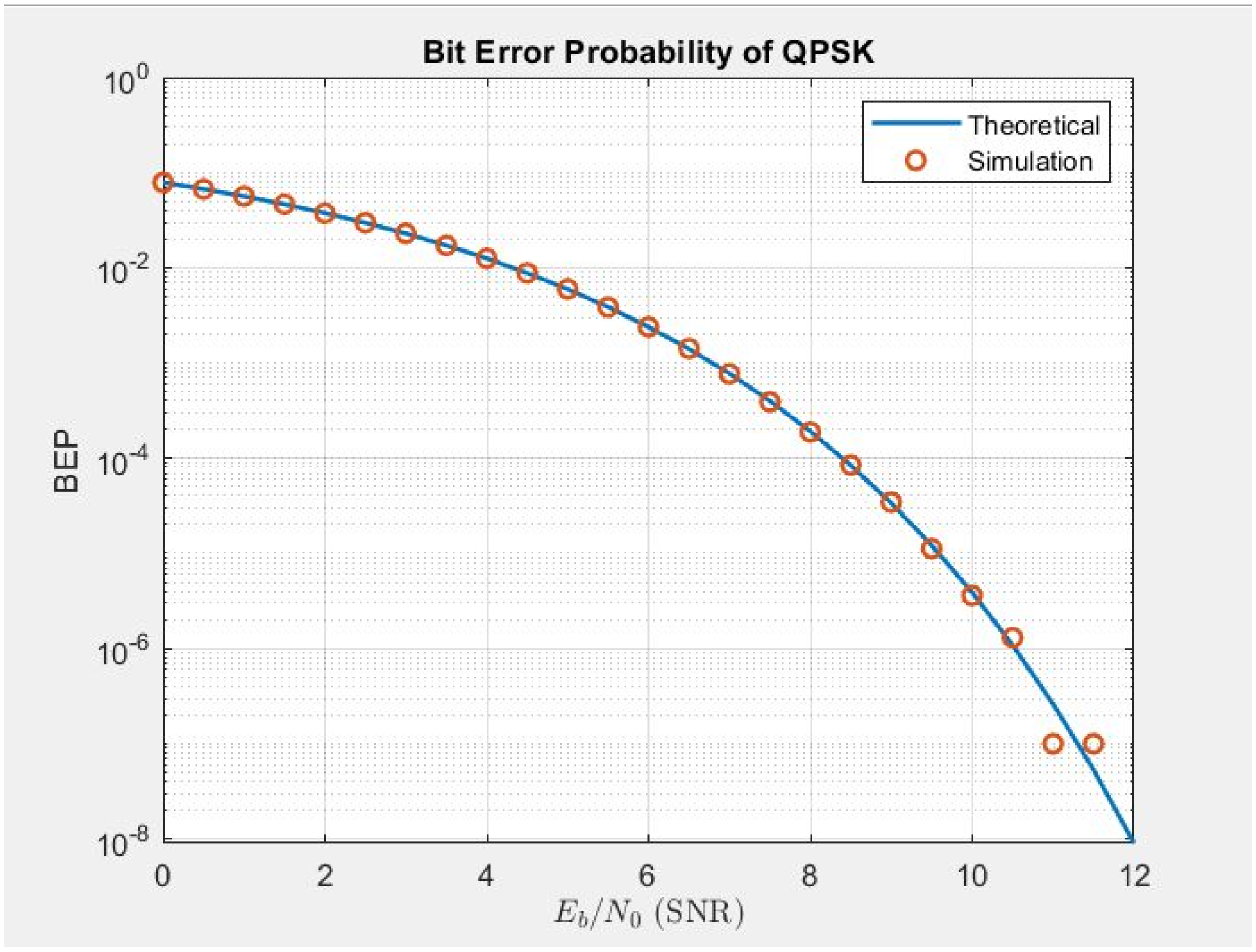
```
demod(i)=1;
demod(i+1)=1;
elseif (x1>0 && x2<0)
demod(i)=1;
demod(i+1)=0;
elseif (x1<0 && x2<0)
demod(i)=0;
demod(i+1)=0;
elseif (x1<0 && x2>0)
demod(i)=0;
demod(i+1)=1;
end
t1=t1+(Tb+.01); t2=t2+(Tb+.01);
end
cnt = 0;
for i=1:N
   if(m(i) ~= demod(i))
        cnt = cnt+1;
   end
end
disp('Bit error Probability : ');
disp(cnt/(N));
subplot(3,2,5);stem(demod);
title('QPSK demodulated bits');xlabel('n---->');ylabel('b(n)');grid
on;
```

% Simulation of Bit Error Probability(BER) of QPSK System :

```
clc;
clear all;
close all;
% This program simulate the bit error probability of QPSK and
compare the results
% with the theoretical value
```

```
N = 1e7; %10^7 samples taken for simulation
E = 0:.5:12; %these are the values of Eb/N0 estimated
c = [-1 1]; %source c of prescribed values
n_error = zeros(1,length(E)); %assigning 0 to all values of n_error
for i=1:length(E)
   m = randsrc(1,N,c); %generating 10^7 samples from the source
code
   n = 1/sqrt(2)*10^(-E(i)/20) * randn(1,N);
   x = m+n;

   for j=1:N
       if (x(j)>0)
           x(j)=1;
       else
           x(j)=-1;
       end
   end
   counter = 0;
   for k=1:N
       if (x(k)~=m(k)) %checking bit errors
           counter=counter+1; %counting errors
       end
   end
   n_error(i) = counter; %storing errors for each value of E
end

snr = 10.^(E./10);
P = qfunc(sqrt(2*snr)); %theoretical calculation of BEP
semilogy(E,P,'LineWidth',1.5); %Plotting the theoretical BEP
BER = n_error/N; %dividing no. of bits in error by total N samples
hold on
semilogy(E,BER,'O','LineWidth',1.5); %Plotting the SIMULATED BEP
xlabel('$E_b/N_0$ (SNR)','Interpreter','LaTeX');
ylabel('BEP');
title ('Bit Error Probability for QPSK Demodulation');
legend('Theoretical','Simulation');
grid on
```

- OUTPUT of BER of QPSK Simulation :



[BFSK Description]

- Frequency-shift keying (FSK) is a frequency modulation scheme in which digital information is transmitted through discrete frequency changes of a carrier wave. The simplest FSK is binary FSK (**BFSK**). **BFSK** uses a pair of discrete frequencies to transmit binary (0s and 1s) information. With this scheme, the "1" is called the mark frequency and the "0" is called the space frequency.

- **BFSK signal expressions :**

$$s_1(t) = A\cos\omega_1 t$$
$$s_2(t) = A\cos\omega_2 t$$

$$s_1(t) = \sqrt{\frac{2E_b}{T_b}}\cos\omega_1 t$$

$$\phi_1(t) = \sqrt{\frac{2}{T_b}}\cos\omega_1 t$$

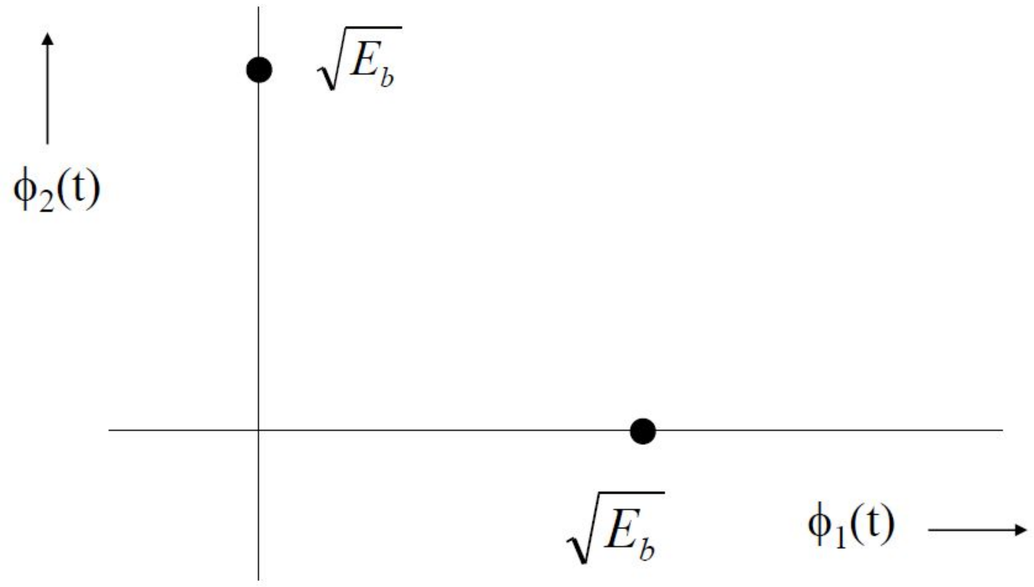
$$s_1(t) = \sqrt{E_b}\,\phi_1(t)$$

$$s_2(t) = \sqrt{\frac{2E_b}{T_b}}\cos\omega_2 t$$

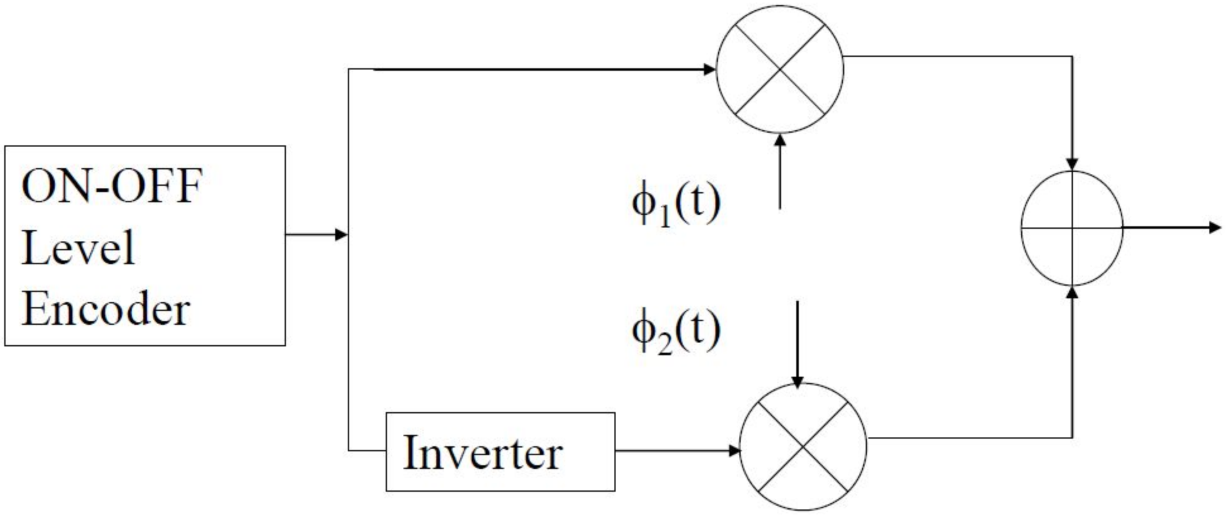
$$s_2(t) = \sqrt{E_b}\,\phi_2(t)$$

- f1 and f2 are chosen such that s1(t) and s2(t) are orthogonal.

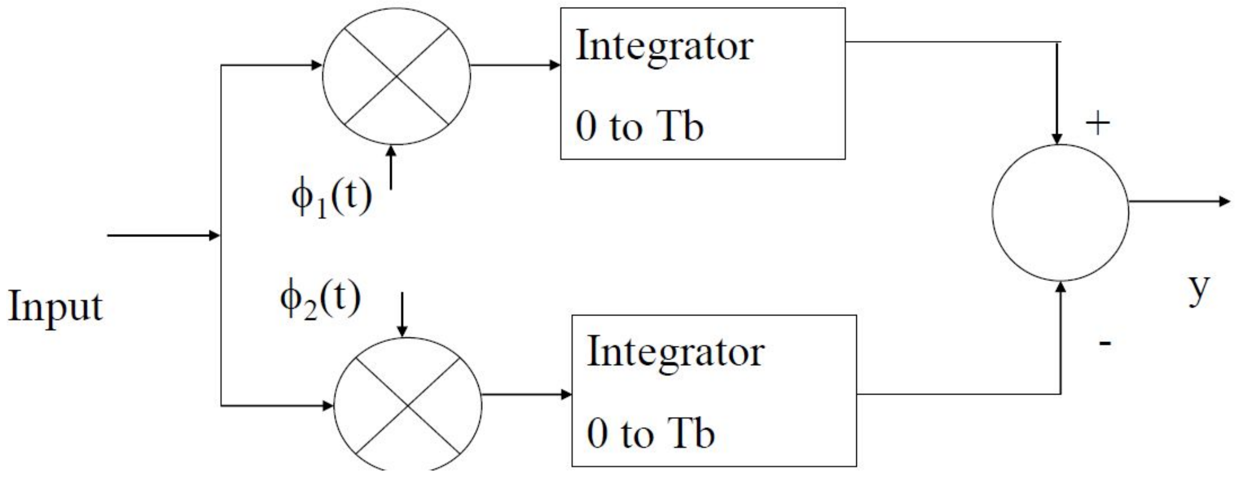
- **BFSK Constellation diagram:**



- **FSK Transmitter :**



- **FSK Receiver :**



- The upper part of the receiver gives the value of x1 and the lower part gives the value of x2. If x1-x2 is greater than 0 then the received bit is 1. Else received a bit is 0.

[Algorithm]

Initialization commands

BFSK modulation

1. Generate two carrier signals.
2. Start FOR loop
3. Generate binary data, message signal and inverted message signal
4. Multiply carrier 1 with message signal and carrier 2 with inverted message signal

5. Perform addition to get the FSK modulated signal
6. Plot message signal and FSK modulated signal.
7. End FOR loop.
8. Plot the binary data and carriers.

BFSK demodulation

1. Start FOR loop
2. Perform correlation of FSK modulated signal with carrier 1 and carrier 2 to get two decision variables x1 and x2.
3. Make a decision on x = x1-x2 to get demodulated binary data. If x>0, choose '1' else choose '0'.
4. Plot the demodulated binary data.

BFSK Bandwidth Calculation with non coherent detection

1. Input roll-off factor
2. Input bit rate
3. Apply formula $B = Rs(2+r)$
4. Output bandwidth

BFSK Bandwidth Calculation with coherent detection

1. Input roll-off factor
2. Input bit rate
3. Apply formula $B = Rs(1.5+r)$
4. Output bandwidth

BFSK BEP Calculation with non coherent detection

1. Input amplitude
2. Calculate bit energy

$$E = \frac{A^2T}{2}$$

3. Apply bit error probability formula

$$P_B = \frac{1}{2}e^{\left(-\frac{E_b}{2N_0}\right)}$$

4. This is about the Theoretical Calculations.
5. For Practical Purpose, Compare the modulation bits and demodulated bits. Count the difference bits. Output probability is equal to difference bits/total bits.

BFSK BEP Calculation with coherent detection

1. Input amplitude
2. Calculate bit energy

$$E = \frac{A^2T}{2}$$

3. Apply bit error probability formula

$$P_B = Q\left(\sqrt{\frac{E_b}{N_0}}\right)$$

4. This is about the Theoretical Calculations.
5. For Practical Purpose, Compare the modulation bits and demodulated bits. Count the difference bits. Output probability is equal to difference bits/total bits.

[Matlab Simulation code of BFSK]

% BFSK Modulation

```
% FSK Modulation
% NOTE : we always give fc1 < fc2 so that we can make bit 1 for
higher
%frequency and bit 0 for lower frequency
clc;
clear all;
close all;

%GENERATE CARRIER SIGNAL
Tb=1;  %INPUT FROM GUI   Bit Time
A = 2; %INPUT FROM GUI   Amplitude
fc1=2; %INPUT FROM GUI   Frequency for bit 0
fc2=5; %INPUT FROM GUI   Frequency for bit 1
N0 = 0.1; %Noise Power Spectral Density...INPUT FROM GUI
r = 0; %roll-off factor...INPUT FROM GUI
Eb = (A.^2)*Tb/2; %Calculation of bit Energy
t=0:(Tb/100):Tb;
c1=sqrt((Eb*2)/Tb)*sin(2*pi*fc1*t); %Generated carrier signal 1
c2=sqrt((Eb*2)/Tb)*sin(2*pi*fc2*t); %Generated carrier signal 2

% generate message signal
N=8; % INPUT FROM GUI (No. of bits you want to send)
m=rand(1,N); % Generating total N numbers randomly between 0 and 1
t1=0;
t2=Tb;
for i=1:N
    t = t1:(Tb/100):t2;
    if m(i)>0.5
        m(i)=1;
        m_s=ones(1,length(t));
        inverted_m_s=zeros(1,length(t));
    else
```

```
        m(i)=0;
        m_s=zeros(1,length(t));
        inverted_m_s=ones(1,length(t));
    end
    message(i,:) = m_s; %meaning one row, many columns at a time
    %Multiplier
    fsk_sig1(i,:) = c2.*m_s;
    fsk_sig2(i,:) = c1.*inverted_m_s;
    fsk = fsk_sig1 + fsk_sig2;
    %plotting the message signal and the modulated signal
    subplot(3,2,2);
    axis([0 N -2 2]);
    plot(t,message(i,:),'r'); %plotting the message which will be
sent
    title('Message signal');
    xlabel('t','Interpreter','LaTeX');
    ylabel('m(t)','Interpreter','LaTeX');
    grid on;
    hold on;
    subplot(3,2,5);
    plot(t,fsk(i,:)); %plotting the final bfsk signal
    title('BFSK signal');
    xlabel('t','Interpreter','LaTeX');
    ylabel('s(t)','Interpreter','LaTeX');
    grid on;
    hold on;
    t1=t1+(Tb+.01); %incrementing the time pointer t1 by 1 for next
bit
    t2=t2+(Tb+.01); %incrementing the time pointer t2 by 1 for next
bit
end
hold off

%Bandwidth for coherent signalling
bw_coherent = (fc2 - fc1)*(1.5 + r);
disp('BW of coherent signalling : ');
disp(bw_coherent);
hold on;
```

```
%Bandwidth for non-coherent signalling
bw_non_coherent = (fc2 - fc1)*(2 + r);
disp('BW of non-coherent signalling : ');
disp(bw_non_coherent);
hold off;

%Bit Error Probability for coherent orthogonal signalling
Pb = qfunc(sqrt(Eb./N0));
disp('Bit Error Probability(coherent orthogonal signalling) : ');
disp(Pb);
hold on;
%Bit Error Probability for non-coherent orthogonal signalling
P_b_non_c = (1/2).*exp(-Eb./(2.*N0));
disp('Bit Error Probability(non-coherent orthogonal signalling) :
');
disp(P_b_non_c);
hold off;

%Plotting binary data bits and carrier signal
subplot(3,2,1);
stem(m); %plotting binary data which we want to send through BFSK
title('Binary data of message');
xlabel('n','Interpreter','LaTeX');
ylabel('b(n)','Interpreter','LaTeX');
grid on;
subplot(3,2,3);
plot(t,c1); %plotting carrier signal 1
title('Carrier signal 1');
xlabel('t','Interpreter','LaTeX');
ylabel('$c_1(t)$','Interpreter','LaTeX');
grid on;
subplot(3,2,4);
plot(t,c2); %plotting carrier signal 1
title('Carrier signal 2');
xlabel('t','Interpreter','LaTeX');
ylabel('$c_2(t)$','Interpreter','LaTeX');
grid on;
```

% BFSK Demodulation

```
% BFSK Demodulation - Here we are doing coherent detection of
orthogonal signalling
t1 = 0;
t2 = Tb;
for i=1:N
    t = t1:(Tb/100):t2;
    %correlator
    x1 = sum(c1.*fsk_sig1(i,:));
    x2 = sum(c2.*fsk_sig2(i,:));
    x = x1 - x2;
    %decision device
    if x>0
        demod(i)=1;
    else
        demod(i)=0;
    end
    t1 = t1 + (Tb+.01);
    t2 = t2 + (Tb+.01);
end
%Plotting the demodulated data bits
subplot(3,2,6);
stem(demod);
title('Demodulated data');
xlabel('n','Interpreter','LaTeX');
ylabel('b(n)','Interpreter','LaTeX');
grid on;
```

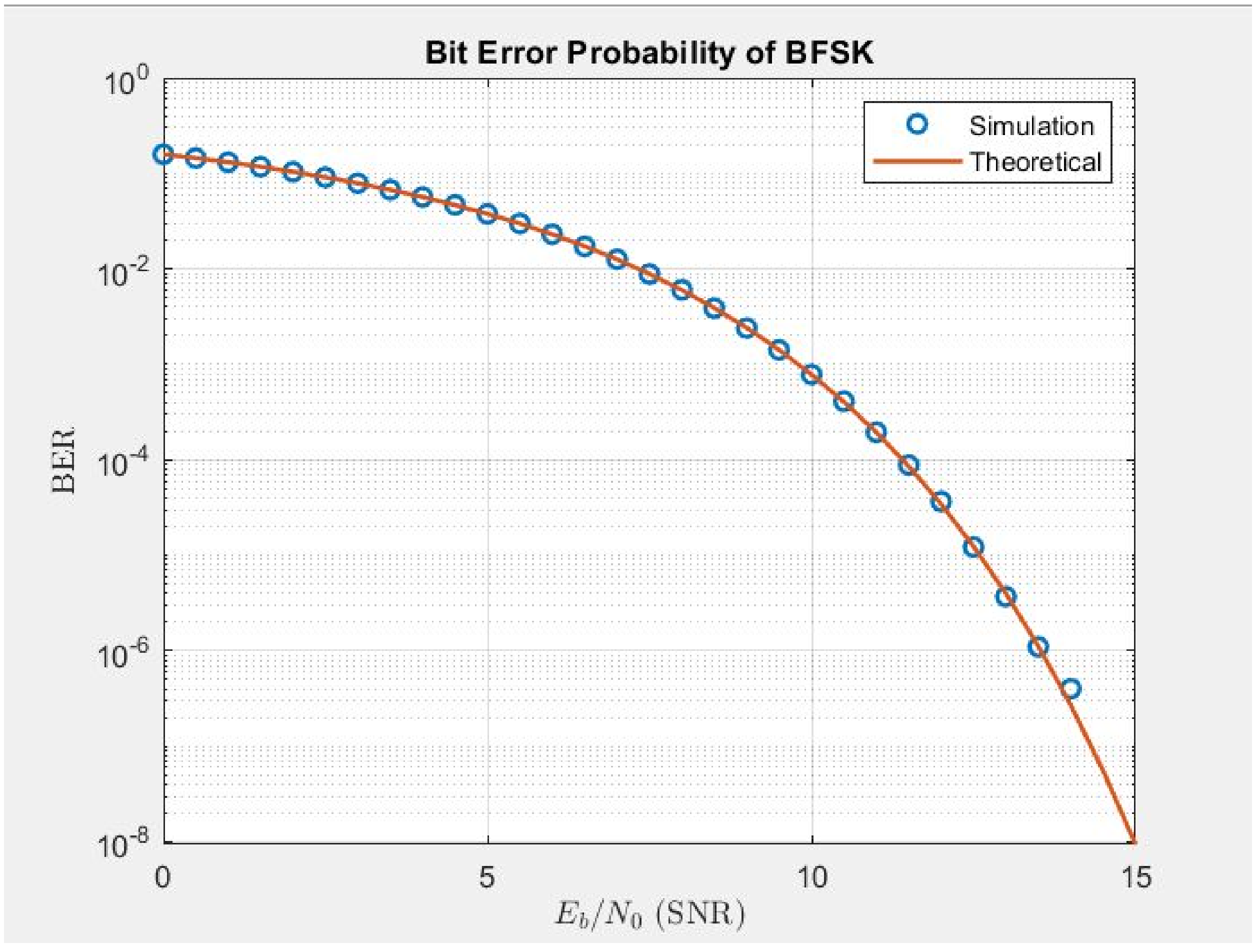
% Simulation of Bit Error Probability(BER) of BFSK System

```
%Bit Error Probability of BFSK using AWGN
%Input Bits Generations
```

```
N = 1e7; %using 10^7 samples for simulation
m = randi([0 1],1,N); %Generated 10^7 random samples between 0
and 1
%BFSK Mapping
for i=1:N
    if m(i)==0
        x(i) = sqrt(-1);
    else
        x(i) = 1;
    end
end

%AWGN Channel
BER_simulation = []; %declaring empty array to store simulation
result
BER_theoretical = []; %declaring empty array to store theoretical
equation result

for EbN0_in_dB = 0:0.5:15
    EbN0 = 10^(EbN0_in_dB/10);
    n = (1/sqrt(2)).*[randn(1,N) + sqrt(-1)*randn(1,N)]; %Complex
Gaussian Noise
    sigma = sqrt(1/EbN0); %Noise standard deviation
    r = x + sigma.*n; %Received Signal

    %Detection of received signal
    m_cap = (real(r)>imag(r)); %if real(r)>imag(r) then consider
bit as 1 else 0
    %BER Calculation
    BER_sim1 = sum(m~=m_cap)/N; %ratio of total bits in error to
total N transmitted bits
    BER_simulation = [BER_simulation BER_sim1];
    BER_th_Qfunction = qfunc(sqrt(EbN0)); %calculation of
theoretical BER using Q-function
    BER_theoretical = [BER_theoretical BER_th_Qfunction];
%appending BER_th_Qfunction to BER_theoretical array
    end
```

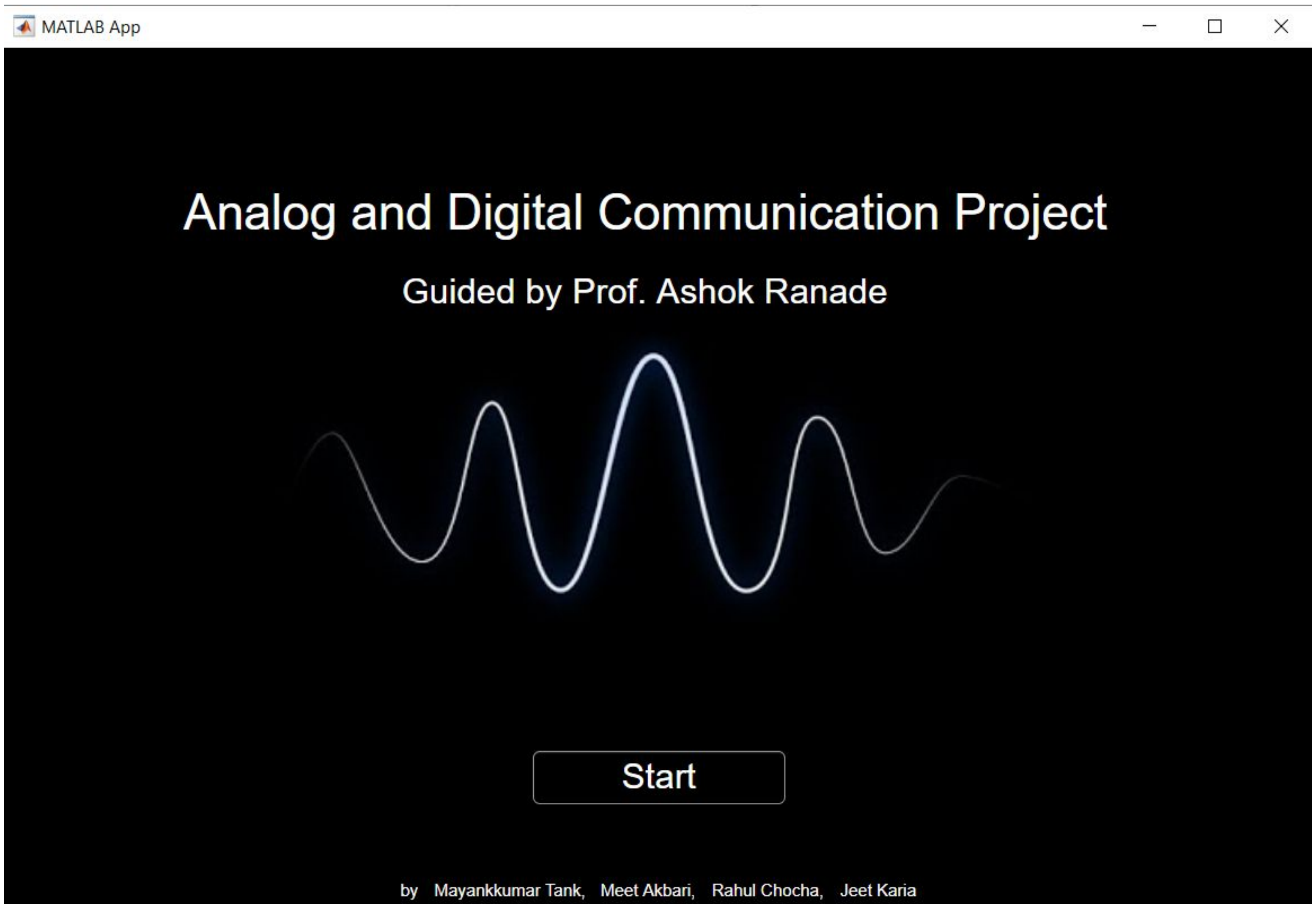
```
%Plotting the simulation result
EbN0_in_dB = 0:0.5:15; %values of SNR in db on for x-axis
semilogy(EbN0_in_dB,BER_simulation,'O','LineWidth',1.5);
hold on
semilogy(EbN0_in_dB,BER_theoretical,'Linewidth',1.5);
legend('Simulation','Theoretical');
grid on
xlabel('$E_b/N_0$ (SNR)','Interpreter','LaTeX');
ylabel('BER','Interpreter','LaTeX');
title('Bit Error Probability of BFSK');
```

- OUTPUT of BER of BFSK Simulation :



[Screenshots of GUI]

<u>Screen 1</u>



Analog and Digital Communication Project

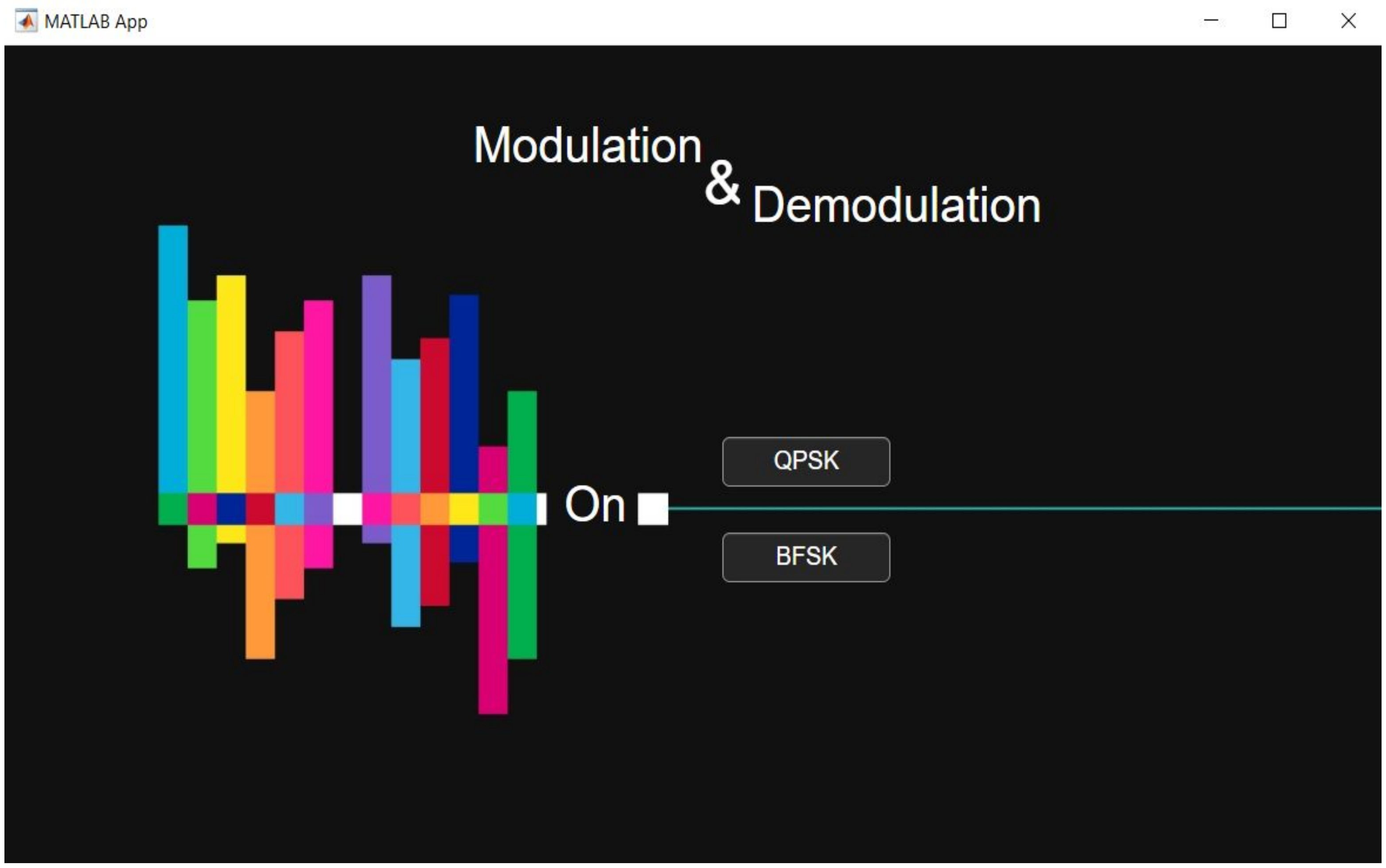
Guided by Prof. Ashok Ranade

by Mayankkumar Tank, Meet Akbari, Rahul Chocha, Jeet Karia

- **Description:**

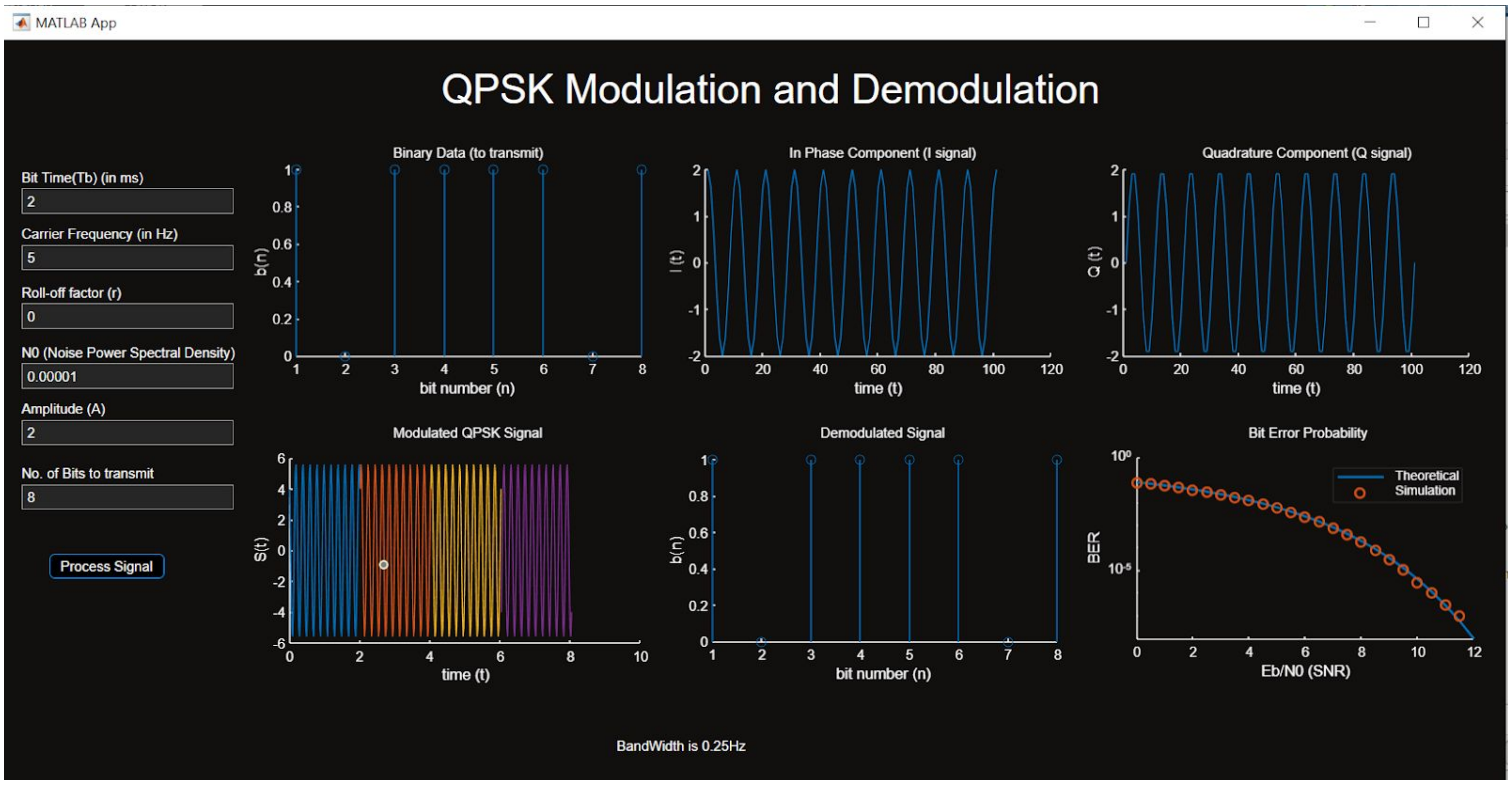
- This is the cover page screen of the project. On this screen, you find the start button. By clicking on it you will jump to **screen 2**.

Screen 2



- **Description:**
- This is the menu page screen of the project. It contains two buttons **QPSK** and **BFSK**. What type of modulation you want to do on the digital data. By clicking on the QPSK button you will be redirected to **screen 3**(as shown below) or by clicking on the BFSK button you will be redirected to **screen 4**(as shown below).

Screen 3


MATLAB App

QPSK Modulation and Demodulation

Bit Time(Tb) (in ms): 2

Carrier Frequency (in Hz): 5

Roll-off factor (r): 0

N0 (Noise Power Spectral Density): 0.00001

Amplitude (A): 2

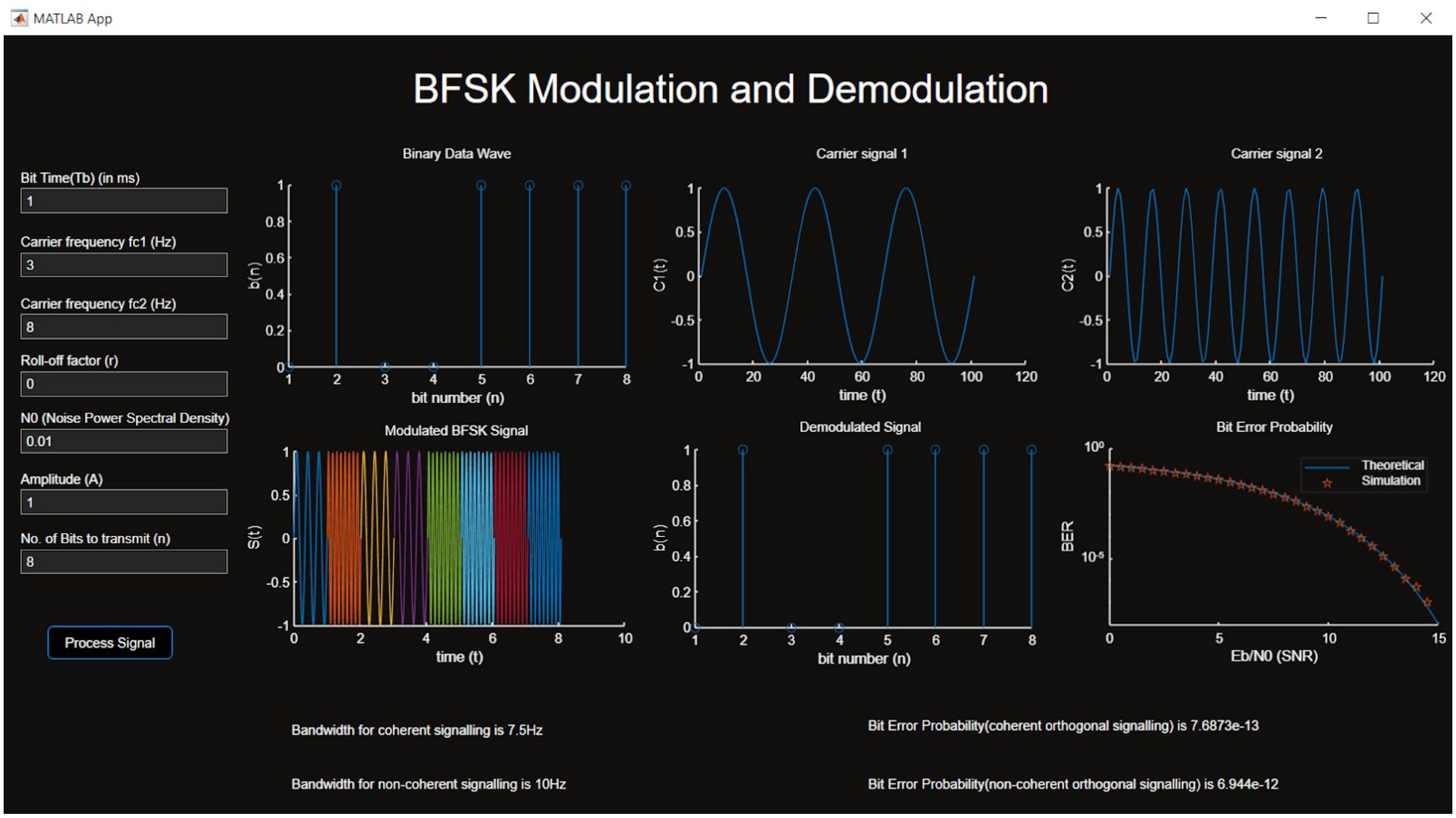
No. of Bits to transmit: 8

BandWidth is 0.25Hz

- **Description:**

- On this screen you are able to run the **QPSK system**. On the left side of the GUI screen you have to input all the parameters needed for modulation. After that you have to **click** on the Process Signal button. So you can see all the graphs generated by **simulation code**(as shown above). Here for the modulation of binary data, the binary data is generated **randomly** from the number of bits entered by the user.

Screen 4



Bit Time(Tb) (in ms): 1

Carrier frequency fc1 (Hz): 3

Carrier frequency fc2 (Hz): 8

Roll-off factor (r): 0

N0 (Noise Power Spectral Density): 0.01

Amplitude (A): 1

No. of Bits to transmit (n): 8

Bandwidth for coherent signalling is 7.5Hz

Bandwidth for non-coherent signalling is 10Hz

Bit Error Probability(coherent orthogonal signalling) is 7.6873e-13

Bit Error Probability(non-coherent orthogonal signalling) is 6.944e-12

- **Description:**

- On this screen you are able to run the **BFSK system**. On the left side of the GUI screen you have to input all the parameters needed for modulation. After that you have to **click** on the Process Signal button. So you can see all the graphs generated by **simulation code**(as shown above) Here for the modulation of binary data, the binary data is generated **randomly** from the number of bits entered by the user.

[Team mates]

Thank you!