Exp 1: BPSK Modulation

```
clc
clear all;
close all;
N=10;
x_inp=round(rand(1,N));
Tb=0.0001;
x_bit=[];
nb=100;
for n=1:1:N
   if (x_inp(n)==1)
      x_bitt=ones(1,nb);
   elseif(x_inp(n)==0)
      x_bitt=zeros(1,nb);
   end
   x_bit=[x_bit x_bitt];
end
t1=Tb/nb:Tb/nb:nb*N*(Tb/nb);
f1=figure(1);
set(f1,'color',[1 1 1]);
subplot(3,1,1);
plot(t1,x_bit,'LineWidth',2);
axis([0 Tb*N -0.5 1.5]);
ylabel("Aplitude(Volt)");
xlabel("Time(Sec)");
title("Input signal as digital signal");

Ac=5;
mc=4;
```

```
fc=mc*(1/Tb);
fi1=0;
fi2=pi;
t2=Tb/nb:Tb/nb:Tb;
t2L=length(t2);
x_mod=[];
for i=1:1:N
    if (x_inp(i)==1)
        x_mod0=Ac*cos(2*pi*fc*t2+fi1);
    else
        x_mod0=Ac*cos(2*pi*fc*t2+fi2);
    end
    x_mod=[x_mod x_mod0];
end
t3=Tb/nb:Tb/nb:Tb*N;
subplot(3,1,2);
plot(t3,x_mod);
xlabel('Time(sec)');
ylabel('Amplitude(volt)');
title('Signal of BPSK modulation ');
x=x_mod;
h=1;
w=0;
y=h.*x+w;
y_dem=[];
for n=t2L:t2L:length(y)
    t=Tb/nb:Tb/nb:Tb;
    c=cos(2*pi*fc*t);
    y_dem0=c.*y((n-(t2L-1)):n);
```

```
   t4=Tb/nb:Tb/nb:Tb;
   z=trapz(t4,y_dem0);
   A_dem=round((2*z/Tb));
   if(A_dem>Ac/2)
      A=1;
   else
      A=0;
   end
   y_dem=[y_dem A];
end
x_out=y_dem;
xx_bit=[];
for n=1:length(x_out)
   if (x_out(n)==1)
      xx_bitt=ones(1,nb);
   elseif (x_out(n)==0)
      xx_bitt=zeros(1,nb);
   end
   xx_bit=[xx_bit xx_bitt];
end
t4=Tb/nb:Tb/nb:nb*length(x_out)*(Tb/nb);
subplot(3,1,3)
plot(t4,xx_bit,'LineWidth',2);grid on;
axis([ 0 Tb*length(x_out) -0.5 1.5]);
ylabel('Amplitude(volt)');
xlabel(' Time(sec)');
title('Output signal as digital signal');
```

Exp 2: DPSK Modulation

```
clc;
clear all;
close all;
N=10;
bk=round(rand(1,N));
br=10^6;
f=br;
T=1/br;
grid on;
subplot(4,1,1);
stem(bk,'LineWidth',1.5);
title('Information bits to be transimitted');
axis([0 11 0 1.5]);
dk=1;
coded= [dk];
for i=1:length(bk)
   temp=~xor(dk,bk(i));
   coded=[coded temp];
   dk=temp;
end
subplot (4,1,2);
stem(coded,'linewidth',1.5);
grid on;
title('Differentially encoded signal');
axis([0 11 0 1.5]);
coded_PNRZ=2*coded-1;
mod_sig=[];
t=T/99:T/99:T;
```

```
for i=1:length(coded)
    temp=coded_PNRZ(i)*sqrt(2/T)*cos(2*pi*f*t);
    mod_sig=[mod_sig temp];
end
subplot(4,1,3);
 tt=T/99:T/99:(T*length(coded));
plot(tt,mod_sig,'linewidth',1.5);
title('DPSK Modulated Signal');
grid on;
rec_sig=mod_sig;
rec_data=[];
for i=1:length(coded)-1
    y_in=rec_sig((i-1)*length(t)+1:i*length(t)).*rec_sig((i)*length(t)+1:(i+1)*length(t));
    y_in_intg=trapz(t,y_in);
    if(y_in_intg>0)
        temp=1;
    else
        temp=0;
    end
    rec_data=[rec_data temp];
end
subplot(4,1,4);
stem(rec_data,'linewidth',3);
title('Received information bits');
axis([0 11 0 1.5]);
```

Exp 3: QPSK Modulation

```
clc;
clear all;
close all;
b=[1 0 1 0 1];
N=8;
b = round(rand(1,N));
subplot(4,1,1);
stem(b, 'filled')
xlabel('Bit index')
ylabel ('Transmitted bits')
NRZ_out=[];
Vp=1;
% Encode input bitstream as Bipolar NRZ waveform
for index=1:size(b,2)
   if b(index)==1
      NRZ_out=[NRZ_out ones(1,200)*Vp];
   elseif b(index)==0
      NRZ_out=[NRZ_out zeros(1,200)*(-Vp)];
   end
end
subplot(4,1,2);
plot(NRZ_out)
%BFSK modulation
t=0.005:0.005:8;
f1=3;
f2=5;
A=5; %Carrier amplitude
mod_sig=[];
```

```
for i=1:1:length(NRZ_out)
   if (NRZ_out(i)==1)
      y=A*cos(2*pi*f1*t(i));
   else
      y=A*cos(2*pi*f2*t(i));
   end
mod_sig =[mod_sig y];
end
%Plot the modulated signal
subplot (4,1,3);
plot(t,mod_sig)
xlabel('Time in seconds')
ylabel('Modulated Signal')
%Demodulation
demod_branch_1=mod_sig.*(cos(2*pi*f1*t));
demod_branch_2=mod_sig.*(cos(2*pi*f2*t));
%Integration (LPF operation)
y_1=[];
for i=1:200:size(demod_branch_1,2)
    y_1=[y_1 trapz(t(i:i+199),demod_branch_1(i:i+199))];
end
y_2=[];
for i=1:200:size(demod_branch_2,2)
   y_2=[y_2 trapz(t(i:i+199),demod_branch_2(i:i+199))];
end
rec_sig=y_1>y_2;
subplot (4,1,4);
stem(rec_sig, 'filled', 'r' )
xlabel(' Bit index ' )
```

```
ylabel(' Received bits ' )
```

Exp 4: BFSK Modulation

```
clc;
clear all;
close all;
Tb=1;
t=0:(Tb/100):Tb;
fc=1;
c1=sqrt(2/Tb)*cos(2*pi*fc*t);
c2=sqrt(2/Tb)*sin(2*pi*fc*t);
subplot(321);
plot(t,c1);
title('Carrier Signal-1');
xlabel('t--------->');
ylabel('c1(t)');
grid on;
subplot(322);
plot(t,c2);
title('Carrier Signal-2');
xlabel('t--------->');
ylabel('c2(t)');
grid on;
N=16;
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_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_sig(i,:)=c2.*m_s;
qpsk=odd_sig+even_sig;
subplot(323);
stem(m)
title('Binary Data Bits of Message Signal');
xlabel('n--------->');
ylabel('b(n)');
grid on;
subplot(324);
plot(t,qpsk(i,:));
title('QPSK Modulated Signal');
xlabel('t--------->');
ylabel('s(t)');
grid on;
```

```
  hold on;
  t1=t1+(Tb+0.01);
  t2=t2+(Tb+0.01);
end
hold off;
t1=0;
t2=Tb;
for i=1:N-1
  t=[t1:(Tb/100):t2];
  x1=sum(c1.*qpsk(i,:));
  x2=sum(c2.*qpsk(i,:));
  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+0.01);
  t2=t2+(Tb+0.01);
end
subplot(325);
stem(demod)
```

```
title('QPSK Demodulated Signal');

xlabel('n--------->');

ylabel('b(n)');

grid on;
```

Exp 5: Pulse Width Modulation

```
clc;

clear all;

close all;

F2=input('Message Frequency(fm)=');

F1=input('Carrier Sine frequency(fs)=');

A=5;

t=0:0.001:1;

c=A.*sin(2*pi*F1*t);

subplot(311);

plot(t,c);

xlabel('Time');

ylabel('Amplitude');

title('Carrier Sine Wave');

grid on;

m=0.75*A.*square(2*pi*F2*t);

subplot(312);

plot(t,m);

xlabel('Time');

ylabel('Amplitude');

title('Message Signal');

grid on;

n=length(c);

for i=1:n
```

```
    if(m(i)>=c(i))
        pwm(i)=1;
    else
        pwm(i)=0;
    end
end
subplot(313);
plot(t,pwm);
xlabel('Time');
ylabel('Amplitude');
title('Plot of PWM');
axis([0 1 0 2]);
grid on;
```

Exp 6:

```
P=5;
t=[0:0.1:1*pi];
sig=4*sin(t);
Vh=max(sig);
V1=min(sig);
N=3;M=2^N;
S=(Vh-V1)/M;
partition=[V1+S:S:Vh-S];
codebook=[V1+S/2:S:Vh-S/2];
[index, quantized_sig,distor]=quantiz(sig,partition,codebook);
codedsig=de2bi(index,'left-msb');
codedsig=codedsig';
txbits=codedsig(:);
errvec=randsrc(length(txbits),1,[0 1;(1-P/100) P/100]);
```

```
rxbits=rem(txbits+errvec,2);
rxbits=reshape(rxbits,N,length(sig));
rxbits=rxbits';
index1=bi2de(rxbits,'left-msb');
reconstructedsig=codebook(index1+1);
figure,
subplot(221);
stem(t,sig);
xlabel('Time');
title("Original Signal");
subplot(222);
stem(t,quantized_sig);
xlabel('Time');
title("Quantized Signal");
tt=[0:N*length(t)-1];
subplot(223);
stairs(tt,txbits);
xlabel('Time');
title("PCM waveform");
subplot(224);
stem(t,reconstructedsig);
xlabel('Time');
title("Recieved  Signal");
```

Exp 7: QAM Modulation

```
clc;
clear all;
close all;
M=8;
```

```
N=12;
msg=round(rand(N,1));
disp("Binary input at transmitter");
disp(msg);
Tb=0.000001;
x=msg;
bits=[];
for n=1:1:length(x)
   if x(n)==1
      sig=ones(1,100);
   elseif x(n)==0
      sig=zeros(1,100);
   end
   bits=[bits sig];
end
t1=Tb/100:Tb/100:100*length(x)*(Tb/100);
subplot(311);
plot(t1,bits,'LineWidth',2.5);
grid on;
axis([0 Tb*length(x) -0.5 1.5]);
xlabel('Time (Sec) ');
ylabel('Amplitude (Volts) ')
title('Digital input signal');
msg_reshape=reshape(x,log2(M),N/log2(M))';
disp('Information is reshaped to convert into sybol form');
disp(msg_reshape);
fprintf('\n\n');
size(msg_reshape);
for (j=1:1:N/log2(M))
```

```
        for (i=1:1:log2(M))
            a(j,i)=num2str(msg_reshape(j,i));
        end
    end
    as=bin2dec(a);
    ast=as';
    subplot(312);
    stem(ast,'LineWidth',2.0);
    title('Serial symbol for 8-QAM Modulation');
    xlabel('n(discrete time)');
    ylabel('magnitude');
    disp('Symbol form of information for 8-QAM');
    disp(ast);
    fprintf('\n\n');
    p=qammod(ast,M);
    scatterplot(p);
    grid on;
    title('8-QAM constellation diagram');
    RR=real(p);
    II=imag(p);
    sp=Tb*2;
    sr=1/sp;
    f=sr*2;
    t=sp/100:sp/100:sp;
    ss=length(t);
    m=[];
    for (k=1:1:length(RR))
        yr=RR(k)*cos(2*pi*f*t);
        yim=II(k)*sin(2*pi*f*t);
        y=yr+yim;
        m=[m y];
    end
    tt=sp/100:sp/100:sp*length(RR);
    figure(1);
    subplot(313);
    plot(tt,m);
    xlabel('Time (Sec) ');
    ylabel('Amplitude (Volts) ');
    title('8-QAM Modulated signal');
```

Exp 8: Multiple Input Multiple Output

```
clc;
clear all;
close all;
x=[2 3];
Date_input_bit(1,1)=x(1,1);
Data_input_bit(1,2)=x(1,2);
z=qammod(Data_input_bit,4);
h=[1.3 -0.4; 6 0.11];
e=[0.1 0.1; 0.1 0.1];
out=zeros(10,1);
for i=1;
    out(i,1)=z(i);
    out(i+1,1)=z(i+1);
    out(i,1)=-conj(z(i+1));
    out(i+1,2)=conj(z(i));
end
s1=out(i,1);
```

```
s2=out(i+1,1);

for i=1;

   r(1,1)=(h(1,1)*s1)+(h(1,2)*s2)+e(1,1);

   r(1,2)=((-h(1,1))*conj(s2))+(h(1,2)*conj(s1))+e(1,2);

   r(2,1)=(h(2,1)*s1)+(h(2,2)*s2)+e(2,1);

   r(2,2)=((-h(2,1))*conj(s2))+(h(2,2)*conj(s1))+e(2,2);

end

t(1,1)=((conj(h(1,1))*r(1,1)));

t(1,2)=h(1,2)*(conj(r(1,2)));

t(2,1)=((conj(h(2,1)))*r(2,1));

t(2,2)=((h(2,2)*(conj(r(2,2)))));

c(1,1)=((conj(h(1,2)))*r(1,2));

c(1,2)=h(1,1)*(conj(r(1,2)));

c(2,1)=((conj(h(2,2)))*r(2,1));

c(2,2)=((h(2,1)*(conj(r(2,2)))));

s1_e=t(1,1)+t(1,2)+t(2,1)+t(2,2);

s2_e=c(1,1)-c(1,2)+c(2,1)-c(2,2);

final_output_bits(1,1)=qamdemod(s1_e,4)

final_output_bits(1,2)=qamdemod(s2_e,4)
```

Exp 9 &10 : GSM Kit

1) ATD=? Used to check whether a command is supported or not by the MODEM
2) AT+CBC? Used to get mobile phone or MODEM settings for an operation.
3) AT+CSCA=”+1234567890”,120 Used to modify phone or MODEM settings for an operation.
4) AT+CMSS=1,”+1234567890”,120 Used to carry out an operation(The read commands are not available to get value of last parameter assigned in execution commands because parameters of execution commands are not stored.

Commands	**Description**
SIM Detection	
AT	Test command
AT+CPIN?	Request the PIN registration status of the sim
AT+CREG?	Request registration Status of the SIM
AT + CMEE=1	Error log view
AT+COPS=? :	View the different operations available
AT+COPS?	View the service operator
Phone Control Commands	
AT	Test command
AT+CGMI	Request Manufacturer Identification
AT+CGMM	Request Model Identification
AT+CGMR	Request Revision Identification
AT+CGSN	Request Product Serial No Identification
AT+CSQ	Signal Quality
AT+CPAS	Phone active status
Call Control	
ATA	Answer Command
ATD	Dial Command
ATH	Hang Up Command
ATL	Monitor speaker loudness
ATM	Monitor speaker mode
ATO	Go-on-line
ATP	Set pulse dial as default
ATT	Set tone dial as default
ATT+CRC	Cellular result codes

Commands	**Description**
Call Making and receiving commands	
AT	Test command
ATD+91NUMBER;	Dial a number
ATA	Answer a call
ATA+CLIP=1	Show the callers number
ATA+CLIP=0	Hide the callers number
ATH	Hang a call
AT+CRC	Celllular Result Codes
Sending and receiving Message	
AT	Test Command
AT+CMGF=1 { The text mode of SMS is easier to operate but it allows limited features of SMS .The PDU(Protection data unit) allows more access to SMS services. The headers and body of SMS are accessed in hex format in PDU mode so it allows availing more features.}	Enter the SMS Mode 0: for PDU mode 1: for TEXT Mode
AT+CMGS="+91NUMBER" >Type a message and prtess Ctrl+Z (Send a message)	Specify the number to which message has to be sent
AT+CMGW="Phone number" >Message to be stored and press Crtl+z	As you type AT+CMGW and phone number , > sign appears on next line where one can type the message. Multiple line message can betyped in this case. This is why the message is terminated by providing a " CTRL+Z" combination. As CTRL=Z is pressed, the following info is displayed on the screen: +CMGW:Number on which message has been stored
AT+CMGD	Delete the message
AT+CMGR	Read the message
AT+CMGL	List the message

All Message related AT Commands	
AT+CSMS	Select message service
AT+CPMS	Preferred messaage storage
AT+CMGF	Message format
AT+CSCA	Service center address
AT+CSMP	Set text mode parameters
AT+CSDH	Show text mode parameters
AT+CSCB	Select cell broadband message
AT+CSAS	Save settings
AT+CRES	Restore settings
AT+CMGL	List message
AT+CMGR	Read message
AT+CMGS	Send message
AT+CMGD	Delete message