```
% 1. Write a Matlab  program to evaluate performance of a   1/2-rated convolutionally encoded DS CDMA s
% This program has been developed to study BER performance of a   1/2-rated convolutionally encoded D
% for a single user under AWGN channel
clear all;
close all;
msg=round(rand(1,1000));                %%msg is a random bit sequence of length 1000.
%1/2 rated convolutional Encoder        %%Creates a rate-1/2 convolutional encoder with memory order 3.
trellis=poly2trellis(3,[6 7]);          %%defines the trellis structure with generator polynomials 6 (binary
user=convenc(msg,trellis);      %%encodes the 1000 bits into a longer sequence (since rate is 1/2, output
% Convolutionally encoded data(0,1) are  mapping into +1/1
%% To convert the binary sequences to bipolar NRZ format
length_user=length(user);               %% After convolutional encoding, we have a binary sequence {0,1}
for i=1:length_user                     %%This loop changes all 0s to -1, effectively mapping {0,1} → {-1,+
if user(i)==0
user(i)=-1;                             %%After this, user contains +1 or -1.
end
end
fc=5000; %%carrier frequency, %KHz
eb=.5;    %% energy per bit for BPSK
bitrate=1000;% 1KHz
tb=1/bitrate; %% time per bit of message sequence .each bit is 1 ms (tb = 1/1000 = 0.001 s).
chiprate=10000;       %%each chip is tc = 1/10000 = 0.0001 s. The ratio of chip rate to bit rate is 10:1, meaning
tc=1/chiprate;
%%% CDMA transmitter for a single user
t=tc:tc:tb*length_user;
%%plotting base band signal for user
basebandsig=[];
for i=1:length_user
for j=tc:tc:tb
if user(i)==1
basebandsig=[basebandsig 1];
else
basebandsig=[basebandsig -1];
end
end
end
figure(1)
stairs(t(1:800),basebandsig(1:800))          %%Plots the first 800 samples to show a segment of the baseband
xlabel('Time(sec)')
ylabel('Binary value')
set(gca,'ytick',[ -1  1 ])
title('A segment of original binary sequence for a single user')
%%%% BPSK Modulation
bpskmod=[];                                 %%For each bit (of duration tb), the code multiplies the carrier cos(2
for i=1:length_user
for j=tc:tc:tb
bpskmod=[bpskmod sqrt(2*eb)*user(i)*cos(2*pi*fc*j)];
end
end
%length(bpskmod)
```

```
number=length(t); %Total number of time segments
spectrum=abs(fft(bpskmod));
sampling_frequency=2*fc;
sampling_interval=(1.0/sampling_frequency);
nyquest_frequency=1.0/(2.0*sampling_interval);
for i=1:number
frequency(i)=(1.0/(number*sampling_interval)).*i;
end
figure(2)
plot(frequency,spectrum)
title('Frequency Domain analysis of BPSK modulated signal for a single user')
xlabel('Frequency (Hz)')
ylabel('Magnitude')
grid on
%% PN generator for  a single user
%% let initial seed for a single user  is 1000
seed=[1 -1 1 -1];  %convert it into bipolar NRZ format
spreadspectrum=[];
pn=[];
for i=1:length_user
for j=1:10 %chip rate is 10 times the bit rate
pn=[pn seed(4)];
if seed (4)==seed(3) temp=-1;
else temp=1;
end
seed(4)=seed(3);        %%seed is simply the starting register state used to generate the PN sequence for spreadi
seed(3)=seed(2);
seed(2)=seed(1);
seed(1)=temp;
end
end
% each bit has 100 samples. and each pn chip has 10 samples. there r
% 10 chip per bit there fore size of pn samples and original bit is same
pnupsampled=[];
len_pn=length(pn);
for i=1:len_pn
for j=10*tc:10*tc:tb
if pn(i)==1
pnupsampled=[pnupsampled 1];
else
pnupsampled=[pnupsampled -1];
end
end
end
length_pnupsampled=length(pnupsampled);
sigtx=bpskmod.*pnupsampled;
figure(3)
plot(t(1:200), sigtx(1:200))
title('A segment of Transmitted  DS CDMA signal')
xlabel('Time(sec)')
```

```
ylabel('Amplitude')
grid on
%%%%%%%%%%%%%AWGN CHANNEL%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%snr_in_dBs=1;
snr_in_dBs=0:1.0:10;
for m=1:length(snr_in_dBs)
ber(m)=0.0;
composite_signal=awgn(sigtx,snr_in_dBs(m),'measured');  %% SNR of % dbs
%%%%%%%%%%%%%DEMODULATION FOR USER 1%%%%%%%%%%%%%%%%%%%%%
rx=composite_signal.*pnupsampled;
%%%% BPSK demodulation for a single user
demodcar=[];
for i=1:length_user
for j=tc:tc:tb
demodcar=[demodcar sqrt(2*eb)*cos(2*pi*fc*j)];
end
end
bpskdemod=rx.*demodcar;
len_dmod=length(bpskdemod);
sum=zeros(1,len_dmod/10);
for i=1:len_dmod/10
for j=(i-1)*10+1:i*10
sum(i)=sum(i)+bpskdemod(j);
end
end
sum;
rxbits=[];
for i=1:length_user
if sum(i)>0
rxbits=[rxbits 1];
else
rxbits=[rxbits 0];
end
end
tblen = 3; delay = tblen; % Traceback length
decoded = vitdec(rxbits,trellis,tblen,'cont','hard');
[number,rat] = biterr(decoded(delay+1:end),msg(1:end-delay));
ber(m)=rat;
end % for m
figure(4)
plot(snr_in_dBs,ber);
xlabel('Signal to noise ratio(dB)');
ylabel('BER');
legend('BER simulation for a single user');
title(' Coded BER simulation under AWGN chaanel ')
grid on
```

% 2.Write a Matlab program to evaluate performance of a   1/2-rated convolutionally encoded DS CDMA s

```
% This program has been developed to study BER performance of a   1/2-rated convolutionally encoded D
% for a single user under AWGN and Rayleigh fading channel
clear all;
close all;
msg=round(rand(1,1000));
%1/2 rated convolutional Encoder
trellis=poly2trellis(3,[6 7]);
user=convenc(msg,trellis);
% Convolutionally encoded data(0,1) are  mapping into +1/1
%% To convert the binary sequences to bipolar NRZ format
length_user=length(user);
for i=1:length_user
if user(i)==0
user(i)=-1;
end
end
fc=5000; %%carrier frequency, %KHz
eb=.5;     %% energy per bit
bitrate=1000;% 1KHz
tb=1/bitrate; %% time per bit of message sequence
chiprate=10000;
tc=1/chiprate;
%%% CDMA transmitter for a single user
t=tc:tc:tb*length_user;
%%plotting base band signal for user
basebandsig=[];
for i=1:length_user
for j=tc:tc:tb
if user(i)==1
basebandsig=[basebandsig 1];
else
basebandsig=[basebandsig -1];
end
end
end
figure(1)
stairs(t(1:800),basebandsig(1:800))
xlabel('Time(sec)')
ylabel('Binary value')
set(gca,'ytick',[ -1  1 ])
title('A segment of original binary sequence for a single user')
%%%% BPSK Modulation
bpskmod=[];
for i=1:length_user
for j=tc:tc:tb
bpskmod=[bpskmod sqrt(2*eb)*user(i)*cos(2*pi*fc*j)];
end
end
%length(bpskmod)
number=length(t); %Total number of time segments
```

```
spectrum=abs(fft(bpskmod));
sampling_frequency=2*fc;
sampling_interval=(1.0/sampling_frequency);
nyquest_frequency=1.0/(2.0*sampling_interval);
for i=1:number
frequency(i)=(1.0/(number*sampling_interval)).*i;
end
figure(2)
plot(frequency,spectrum)
title('Frequency Domain analysis of BPSK modulated signal for a single user')
xlabel('Frequency (Hz)')
ylabel('Magnitude')
grid on
%% PN generator for  a single user
%% let initial seed for a single user  is 1000
seed=[1 -1 1 -1];  %convert it into bipolar NRZ format
spreadspectrum=[];
pn=[];
for i=1:length_user
for j=1:10 %chip rate is 10 times the bit rate
pn=[pn seed(4)];
if seed (4)==seed(3) temp=-1;
else temp=1;
end
seed(4)=seed(3);
seed(3)=seed(2);
seed(2)=seed(1);
seed(1)=temp;
end
end
% each bit has 100 samples. and each pn chip has 10 samples. there r
% 10 chip per bit there fore size of pn samples and original bit is same
pnupsampled=[];
len_pn=length(pn);
for i=1:len_pn
for j=10*tc:10*tc:tb
if pn(i)==1
pnupsampled=[pnupsampled 1];
else
pnupsampled=[pnupsampled -1];
end
end
end
length_pnupsampled=length(pnupsampled);
sigtx=bpskmod.*pnupsampled;
figure(3)
plot(t(1:200), sigtx(1:200))
title('A segment of Transmitted  DS CDMA signal')
xlabel('Time(sec)')
ylabel('Amplitude')
```

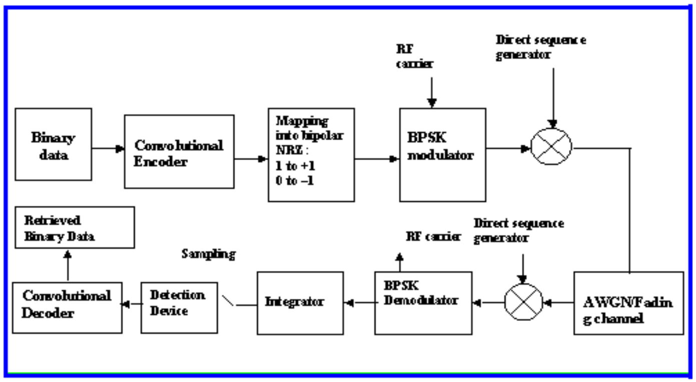
```
grid on
%%%%%%%%%%%%%Adding fadig channel effect%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
chan=rayleighchan(1/chiprate,100);
chan.ResetBeforeFiltering=0;
fad=abs(filter(chan,ones(size(sigtx))));
fadedsig=fad.*sigtx;
snr_in_dBs=0:1.0:10;
for m=1:length(snr_in_dBs)
ber(m)=0.0;
composite_signal=awgn(fadedsig,snr_in_dBs(m),'measured');  %% SNR of % dbs
%%%%%%%%%%%%%DEMODULATION FOR USER 1%%%%%%%%%%%%%%%%%%%%%
rx=composite_signal.*pnupsampled;
%%%% BPSK demodulation for a single user
demodcar=[];
for i=1:length_user
for j=tc:tc:tb
demodcar=[demodcar sqrt(2*eb)*cos(2*pi*fc*j)];
end
end
bpskdemod=rx.*demodcar;
len_dmod=length(bpskdemod);
sum=zeros(1,len_dmod/10);
for i=1:len_dmod/10
for j=(i-1)*10+1:i*10
sum(i)=sum(i)+bpskdemod(j);
end
end
sum;
rxbits=[];
for i=1:length_user
if sum(i)>0
rxbits=[rxbits 1];
else
rxbits=[rxbits 0];
end
end
tblen = 3; delay = tblen; % Traceback length
decoded = vitdec(rxbits,trellis,tblen,'cont','hard');
[number,rat] = biterr(decoded(delay+1:end),msg(1:end-delay));
ber(m)=rat;
end % for m
figure(4)
plot(snr_in_dBs,ber);
xlabel('Signal to noise ratio(dB)');
ylabel('BER');
legend('BER simulation for a single user');
title(' Coded BER simulation under AWGN and Rayleigh fading channel ')
grid on
```

```
% 3.Write a Matlab  program to evaluate performance of a   1/2-rated convolutionally encoded DS CDMA s
% This program has been developed to study BER performance of a   1/2-rated convolutionally encoded D
% for a single user under AWGN and Rician fading channel
clear all;
close all;
msg=round(rand(1,1000));
%1/2 rated convolutional Encoder
trellis=poly2trellis(3,[6 7]);
user=convenc(msg,trellis);
% Convolutionally encoded data(0,1) are  mapping into +1/1
%% To convert the binary sequences to bipolar NRZ format
length_user=length(user);
for i=1:length_user
if user(i)==0
user(i)=-1;
end
end
fc=5000; %%carrier frequency, %KHz
eb=.5;    %% energy per bit
bitrate=1000;% 1KHz
tb=1/bitrate; %% time per bit of message sequence
chiprate=10000;
tc=1/chiprate;
%%% CDMA transmitter for a single user
t=tc:tc:tb*length_user;
%%plotting base band signal for user
basebandsig=[];
for i=1:length_user
for j=tc:tc:tb
if user(i)==1
basebandsig=[basebandsig 1];
else
basebandsig=[basebandsig -1];
end
end
end
figure(1)
stairs(t(1:800),basebandsig(1:800))
xlabel('Time(sec)')
ylabel('Binary value')
set(gca,'ytick',[ -1  1 ])
title('A segment of original binary sequence for a single user')
%%%% BPSK Modulation
bpskmod=[];
for i=1:length_user
for j=tc:tc:tb
bpskmod=[bpskmod sqrt(2*eb)*user(i)*cos(2*pi*fc*j)];
end
```

```
end
%length(bpskmod)
number=length(t); %Total number of time segments
spectrum=abs(fft(bpskmod));
sampling_frequency=2*fc;
sampling_interval=(1.0/sampling_frequency);
nyquest_frequency=1.0/(2.0*sampling_interval);
for i=1:number
frequency(i)=(1.0/(number*sampling_interval)).*i;
end
figure(2)
plot(frequency,spectrum)
title('Frequency Domain analysis of BPSK modulated signal for a single user')
xlabel('Frequency (Hz)')
ylabel('Magnitude')
grid on
%% PN generator for  a single user
%% let initial seed for a single user  is 1000
seed=[1 -1 1 -1];  %convert it into bipolar NRZ format
spreadspectrum=[];
pn=[];
for i=1:length_user
for j=1:10 %chip rate is 10 times the bit rate
pn=[pn seed(4)];
if seed (4)==seed(3) temp=-1;
else temp=1;
end
seed(4)=seed(3);
seed(3)=seed(2);
seed(2)=seed(1);
seed(1)=temp;
end
end
% each bit has 100 samples. and each pn chip has 10 samples. there r
% 10 chip per bit there fore size of pn samples and original bit is same
pnupsampled=[];
len_pn=length(pn);
for i=1:len_pn
for j=10*tc:10*tc:tb
if pn(i)==1
pnupsampled=[pnupsampled 1];
else
pnupsampled=[pnupsampled -1];
end
end
end
length_pnupsampled=length(pnupsampled);
sigtx=bpskmod.*pnupsampled;
figure(3)
plot(t(1:200), sigtx(1:200))
```

```
title('A segment of Transmitted  DS CDMA signal')
xlabel('Time(sec)')
ylabel('Amplitude')
grid on
%%%%%%%%%%%%%Adding fadig channel effect%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
chan=ricianchan(1/chiprate,100,15);
chan.ResetBeforeFiltering=0;
fad=abs(filter(chan,ones(size(sigtx))));
fadedsig=fad.*sigtx;
snr_in_dBs=0:1.0:10;
for m=1:length(snr_in_dBs)
ber(m)=0.0;
composite_signal=awgn(fadedsig,snr_in_dBs(m),'measured');  %% SNR of % dbs
%%%%%%%%%%%%%DEMODULATION FOR USER 1%%%%%%%%%%%%%%%%%%%%%
rx=composite_signal.*pnupsampled;
%%%% BPSK demodulation for a single user
demodcar=[];
for i=1:length_user
for j=tc:tc:tb
demodcar=[demodcar sqrt(2*eb)*cos(2*pi*fc*j)];
end
end
bpskdemod=rx.*demodcar;
len_dmod=length(bpskdemod);
sum=zeros(1,len_dmod/10);
for i=1:len_dmod/10
for j=(i-1)*10+1:i*10
sum(i)=sum(i)+bpskdemod(j);
end
end
sum;
rxbits=[];
for i=1:length_user
if sum(i)>0
rxbits=[rxbits 1];
else
rxbits=[rxbits 0];
end
end
tblen = 3; delay = tblen; % Traceback length
decoded = vitdec(rxbits,trellis,tblen,'cont','hard');
[number,rat] = biterr(decoded(delay+1:end),msg(1:end-delay));
ber(m)=rat;
end % for m
figure(4)
plot(snr_in_dBs,ber);
xlabel('Signal to noise ratio(dB)');
ylabel('BER');
legend('BER simulation for a single user');
title(' Coded BER simulation under AWGN and Rician fading channel ')
```

```
grid on
```



Block diagram of a convolutionally encoded DS CDMA system

```
%4. Write a  Matlab program to study the performance of a  differentially encoded OQPSK based wireless
clear all;
close all;
xbit=[1 0 1 1 0 1 0 0 0 1 1 0];
% Initial reference bit is assumed to be 1
% Binary bit strream is in 0 and 1 : 12 bits
% NOT of Exclusive OR operation
difencod(1)=~(1-xbit(1));
for i=2:length(xbit)
difencod(i)=~(difencod(i-1)-xbit(i));
end
% Differential Encoded binary b
xbit(1)=1-~(difencod(1));
for i=2:length(xbit)
xbit(i)=difencod(i-1)-~(difencod(i));
if(xbit(i)==-1)
xbit(i)=1;
end
end
%Inphase unipolar bit stream
%from differentially encoded ba
for i=1:2:(length(difencod)-1)
inp(i)=difencod(i);
inp(i+1)=inp(i);
end
%Quadrature unipolar bit strea
%from differentially encoded ba
for i=2:2:(length(difencod))
qp(i)=difencod(i);
qp(i-1)=qp(i);
```

```
end
%Inphase bipolar NRZ bit strea
for i=1:(length(inp))
if(inp(i)== 1)
it(i)=1;
elseif(inp(i)==0)
it(i)=-1;
end
end
%Quadrature bipolar NRZ bit st
for i=1:(length(qp))
if(qp(i)== 1)
qt(i)=1;
elseif(qp(i)==0)
qt(i)=-1;
end
end
% Raised Cosine Filter used
filtorder = 40; % Filter order
nsamp=4;
delay = filtorder/(nsamp*2);
rolloff = 0.5; % Rolloff factor of filter
rrcfilter = rcosine(1,nsamp,'fir/normal',rolloff,delay);
% Plot impulse response.
figure(1);
impz(rrcfilter,1);
grid on
%title(' Impulse response of Raised Cosine Filter');
%% Transmitted Signal
% Upsample and apply  raised cosine filter.
itx = rcosflt(it,1,nsamp,'filter',rrcfilter);
Drate=64000;%Bit rate
T=1/Drate;
Ts=T/nsamp;
time=0:Ts:(length(itx)-1)*Ts;
figure(2);
plot(time,itx)
%title(' Low pass filtered InPhase Component');
xlabel( 'Time(sec)');
ylabel( 'Amplitude(volt)');
grid on
tme=Ts:Ts:(length(itx)-1)*Ts+Ts;
qtx = rcosflt(qt,1,nsamp,'filter',rrcfilter);
figure(3);
plot(tme,qtx)
title(' Low pass filtered Quadrature Component');
xlabel( 'Time(sec)');
ylabel( 'Amplitude(volt)');
grid on
fc=900*100000;% 900MHz Carrier frequency chosen
```

```
dd=2*pi*fc*time';
ddd=2*pi*fc*tme';
% One bit or 1/2 of symbol dela
delay(1:nsamp)=0.0;
delay((nsamp+1):length(qtx))=qtx(1:(length(qtx)-nsamp));
half=filtorder/2;
mt=(cos(dd)).*itx+(sin(ddd)).*delay';
figure(4);
plot(time,mt)
xlabel( 'Time(sec)');
ylabel( 'Amplitude(volt)');
title(' Differentially encoded OQPSK modulated signal');
grid on
snr=10;
%Signal-to-noise ratio per samp
madd=awgn(mt,snr);
figure(5);
plot(time,madd)
grid on
xlabel( 'Time(sec)');
ylabel( 'Amplitude(volt)');
%title(' Differentially encoded OQPSK modulated signal with added white noise');
cscomp=mt.*(cos(dd));
sincomp=mt.*(sin(ddd));
plot(time,cscomp)
grid on
xlabel( 'Time(sec)');
ylabel( 'Amplitude(volt)');
lpfin = rcosflt(cscomp,1,nsamp,'filter',rrcfilter);
lpfqu = rcosflt(sincomp,1,nsamp,'filter',rrcfilter);
tmx=0:Ts:(length(lpfin)-1)*Ts;
tmy=Ts:Ts:(length(lpfqu)-1)*Ts+Ts;
figure(5);
plot(tmx,lpfin)
grid on
xlabel( 'Time(sec)');
ylabel( 'Amplitude');
figure(6);
plot(tmy,lpfqu)
grid on
xlabel( 'Time(sec)');
ylabel( 'Amplitude(volt)');
% Initial checking for I and Q ch
itxx=itx(half:nsamp:length(xbit)*nsamp+half-1);
for i=1:1:length(itxx)
if(itxx(i)> 0)
chk1(i)=1;
elseif(itxx(i)< 0)
chk1(i)=-1;
end
```

```
end
ityy=qtx(half:nsamp:length(xbit)*nsamp+half-1);
for i=1:1:length(ityy)
if(ityy(i)> 0)
chk2(i)=1;
elseif(ityy(i)< 0)
chk2(i)=-1;
end
end
disp('I channel bit stream check
distortion = sum((it-chk1).^2)/length(chk1); % Mean square error
distortion
disp('Q channel bit stream chec

distortion = sum((qt-chk2).^2)/length(chk2); % Mean square error
distortion
% Differentially decoded bit stream from I and Q channels
for i=1:2:(length(xbit)-1)
dfd(i)=chk1(i);
end
for i=2:2:(length(xbit))
dfd(i)=chk2(i);
end
for i=1:(length(xbit))
if(dfd(i)== 1)
dfdecod(i)=1;
elseif(dfd(i)==-1)
dfdecod(i)=0;
end
end
detected(1)=1-~(dfdecod(1));
for i=2:length(xbit)
detected(i)=dfdecod(i-1)-(~dfdecod(i));
if(detected(i)==-1)
detected(i)=1;
end
end
disp('Distortion between transm
distortion = sum((xbit-detected).^2)/length(detected); % Mean square error
distortion
tmx=0:(1/64000):(1/64000).*(length(xbit)-1)
figure(7);
subplot(211)
stairs(tmx,xbit)
set(gca,'ytick',[ 0  1 ])
grid on
xlabel( 'Time(sec)');
ylabel( 'Binary value');
title(' Transmitted bit stream ');
subplot(212)
```

```
stairs(tmx,detected)
xlabel( 'Time(sec)');
set(gca,'ytick',[ 0  1 ])
ylabel( 'Binary value');
title(' Received bit stream ');
grid on
```

Expt 5:Develop a matlab source to simulate an Interleaved FEC encoded wireless communication system with implementation of BPSK digital modulation technique
Show atleast three waveforms generated at different sections of the simulated system

```
clear all;
close all;
% Test with synthetically generated sinusoidal wave
f=1000;% Frequency of the audio signal
Fs =4000; % Sampling rate is 4000 samples per second.
t = [1/Fs:1/Fs:1];% total time for simulation=0.05 second.
% Number of samples=4000
Am=1.0;
signal = Am*sin(2*pi*1000*t); % Original signal
figure(1);
plot(t(1:200),signal(1:200))
set(gca,'ytick',[ -1.0   0 1.0 ])
title('A segment of synthetically generated sinusiodal wavform')
grid on
xlabel( 'time(sec)');
ylabel( 'Amplitude(volt)');
maximumvalue=max(signal);
minimumvalue=min(signal);
interval=(maximumvalue-minimumvalue)/255; % interval:
partition = [minimumvalue:interval:maximumvalue]; % -1:0.0078:1
codebook = [(minimumvalue-interval):interval:maximumvalue]; % -1.0078:0.0078:1
[index,quants,distor] = quantiz(signal,partition,codebook);
% Convertion of deci  into binary from least  to most significant
indxtrn=index';
for i=1:4000
matrix(i,1:1:8)=bitget(uint8(indxtrn(i)),1:1:8);
end,
% matrix is of  4000 rows X 8 columns
% matrixtps is a matrix of 8 rows X4000 columns
matrixtps=matrix';
% Baseband is produced, it has 32000 bits
baseband=reshape(matrixtps,4000*8,1);
Tb=1/32000;
% bit rate 32 kbps
time=[0:Tb:1];
figure(2);
stairs(time(1:500),baseband(1:500))
```

```
title(' A segment of baseband signal')
xlabel('Time(sec)')
ylabel('Binary value')
set(gca,'ytick',[0  1 ])
axis([0,time(500),0,1])
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Serial the data for the next step.
input_to_Convolutional_encoder = baseband'; % 1 X  32000
%Now, the binary converted data is sent to th Convolutional encoder.
t=poly2trellis(7, [171 133]);
%Channel coding
code = convenc(input_to_Convolutional_encoder,t); % 1 x 64000
%Interleaving
st2 = 4831;
data_interleave = randintrlv(code,st2); % Interleave, 1 row x 64000 columns
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Binary phase shift keying modulation
M=2;
k=log2(M);
% bit to symbol mapping
symbol=bi2de(reshape(data_interleave,k,length(data_interleave)/k).','left-msb');
symbol=double(symbol);
Binary_phase_shift_keying_modulated_data = pskmod(symbol,M);
%demodulation of  Binary phase shift keying data
Binary_phase_shift_keying_demodulated_data = pskdemod(Binary_phase_shift_keying_modulated_data
[number,ratio]= symerr(symbol,Binary_phase_shift_keying_demodulated_data) % symbol error
%symbol to  bit mapping
%1-bit symbol to Binary bit mapping
Retrieved_bit = de2bi(Binary_phase_shift_keying_demodulated_data,'left-msb');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Deinterleaving
errors = zeros(size(Retrieved_bit));
inter_err = bitxor(Retrieved_bit,errors); % Include burst error.
data_deinterleave=randdeintrlv(inter_err,st2);
%Convolutional Decoding
tblen=3;
decodx= vitdec(data_deinterleave,t,tblen,'cont','hard'); %
N3=length(decodx);
NN=N3/8;
decod2(1:(N3-3))=decodx(tblen+1:end);
decod2(N3)=decodx(1);
decod2=decod2' ; % 32000 X 1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
baseband=double(baseband);
[number,ratio]= biterr(decod2,baseband);
convert=reshape(decod2,8,4000); % First reshaping and then transposing
matrixtps=double(matrixtps);
[number,ratio]= biterr(convert,matrixtps);
convert=convert' ; % 4000 rows X 8 columns
%binary to decimally converted value
```

```
intconv=bi2de(convert); % converted into interger values(0-255) of 4000 samples
% intconv is 4000 rows X 1 column
[number,ratio]= biterr(intconv,index');
sample_value=minimumvalue +intconv.*interval;
figure(3)
subplot(2,1,1)
plot(time(1:100),signal(1:100));
set(gca,'ytick',[ -1.0   0 1.0 ])
axis([0,time(100),-1,1])
title('Graph for a segment of recoded Audio signal')
xlabel('Time(sec)')
ylabel('Amplitude')
grid on
subplot(2,1,2)
plot(time(1:100),sample_value(1:100));
axis([0,time(100),-1,1])
set(gca,'ytick',[ -1.0   0 1.0 ])
title('Graph for a segment of retrieved Audio signal')
xlabel('Time(sec)')
ylabel('Amplitude')
grid on
```

Expt 6:Develop a matlab source to simulate an Interleaved FEC encoded wireless communication communication system with implementation of QPSK digital modulation technique Show atleast three waveforms generated at different sections of the simulated system

```
clear all;
close all;
% Test with synthetically generated sinusoidal wave
f=1000;% Frequency of the audio signal
Fs =4000; % Sampling rate is 4000 samples per second.
t = [1/Fs:1/Fs:1];% total time for simulation=0.05 second.
% Number of samples=4000
Am=1.0;
signal = Am*sin(2*pi*1000*t); % Original signal
figure(1);
plot(t(1:200),signal(1:200))
set(gca,'ytick',[ -1.0   0 1.0 ])
title('A segment of synthetically generated sinusiodal wavform')
grid on
xlabel( 'time(sec)');
ylabel( 'Amplitude(volt)');
maximumvalue=max(signal);
minimumvalue=min(signal);
interval=(maximumvalue-minimumvalue)/255; % interval:
partition = [minimumvalue:interval:maximumvalue]; % -1:0.0078:1
codebook = [(minimumvalue-interval):interval:maximumvalue]; % -1.0078:0.0078:1
```

```
[index,quants,distor] = quantiz(signal,partition,codebook);
% Convertion of deci  into binary from least  to most significant
indxtrn=index';
for i=1:4000
matrix(i,1:1:8)=bitget(uint8(indxtrn(i)),1:1:8);
end,
% matrix is of  4000 rows X 8 columns
% matrixtps is a matrix of 8 rows X4000 columns
matrixtps=matrix';
% Baseband is produced, it has 32000 bits
baseband=reshape(matrixtps,4000*8,1);
Tb=1/32000;
% bit rate 32 kbps
time=[0:Tb:1];
figure(2);
stairs(time(1:500),baseband(1:500))
title(' A segment of baseband signal')
xlabel('Time(sec)')
ylabel('Binary value')
set(gca,'ytick',[0  1 ])
axis([0,time(500),0,1])
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Serial the data for the next step.
input_to_Convolutional_encoder = baseband'; % 1 X  32000
%Now, the binary converted data is sent to th Convolutional encoder.
t=poly2trellis(7, [171 133]);
%Channel coding
code = convenc(input_to_Convolutional_encoder,t); % 1 x 64000
%Interleaving
st2 = 4831;
data_interleave = randintrlv(code,st2); % Interleave, 1 row x 64000 columns
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Quadrature phase shift keying modulation
M=4;
k=log2(M);
baseband=double(baseband);
% bit to symbol mapping
symbol=bi2de(reshape(data_interleave,k,length(data_interleave)/k).','left-msb');
Quadrature_phase_shift_keying_modulated_data = pskmod(symbol,M);
% demodulation of Quadrature phase shift keying data
Quadrature_phase_shift_keying_demodulated_data = pskdemod(Quadrature_phase_shift_keying_modula
[number,ratio]= symerr(symbol,Quadrature_phase_shift_keying_demodulated_data) % symbol error
% symbol to  bit mapping
% 2-bit symbol to Binary bit mapping
Retrieved_bit = de2bi(Quadrature_phase_shift_keying_demodulated_data,'left-msb');
Retrieved_bit=Retrieved_bit';
Retrieved_bit=reshape(Retrieved_bit, 64000,1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Deinterleaving
errors = zeros(size(Retrieved_bit));
```

```
inter_err = bitxor(Retrieved_bit,errors); % Include burst error.
data_deinterleave=randdeintrlv(inter_err,st2);
%Convolutional Decoding
tblen=3;
decodx= vitdec(data_deinterleave,t,tblen,'cont','hard'); %
N3=length(decodx);
NN=N3/8;
decod2(1:(N3-3))=decodx(tblen+1:end);
decod2(N3)=decodx(1);
decod2=decod2' ; % 32000 X 1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
baseband=double(baseband);
[number,ratio]= biterr(decod2,baseband)
convert=reshape(decod2,8,4000); % First reshaping and then transposing
matrixtps=double(matrixtps);
[number,ratio]= biterr(convert,matrixtps)
convert=convert' ; % 4000 rows X 8 columns
% binary to decimally converted value
intconv=bi2de(convert); % converted into interger values(0-255) of 4000 samples
% intconv is 4000 rows X 1 column
[number,ratio]= biterr(intconv,index');
sample_value=minimumvalue +intconv.*interval;
figure(3)
subplot(2,1,1)
plot(time(1:100),signal(1:100));
set(gca,'ytick',[ -1.0   0 1.0 ])
axis([0,time(100),-1,1])
title('Graph for a segment of recoded Audio signal')
xlabel('Time(sec)')
ylabel('Amplitude')
grid on
subplot(2,1,2)
plot(time(1:100),sample_value(1:100));
axis([0,time(100),-1,1])
set(gca,'ytick',[ -1.0   0 1.0 ])
title('Graph for a segment of retrieved Audio signal')
xlabel('Time(sec)')
ylabel('Amplitude')
grid on


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Expt 7:Develop a matlab source to simulate an Interleaved FEC encoded wireless communication communication system with implementation of 4-QAM digital modulation technique
Show atleast three waveforms generated at different sections of the simulated system

```
clear all;
close all;
% Test with synthetically generated sinusoidal wave
f=1000;% Frequency of the audio signal
```

```
Fs =4000; % Sampling rate is 4000 samples per second.
t = [1/Fs:1/Fs:1];% total time for simulation=0.05 second.
% Number of samples=4000
Am=1.0;
signal = Am*sin(2*pi*1000*t); % Original signal
figure(1);
plot(t(1:200),signal(1:200))
set(gca,'ytick',[ -1.0   0 1.0 ])
title('A segment of synthetically generated sinusiodal wavform')
grid on
xlabel( 'time(sec)');
ylabel( 'Amplitude(volt)');
maximumvalue=max(signal);
minimumvalue=min(signal);
interval=(maximumvalue-minimumvalue)/255; % interval:
partition = [minimumvalue:interval:maximumvalue]; % -1:0.0078:1
codebook = [(minimumvalue-interval):interval:maximumvalue]; % -1.0078:0.0078:1
[index,quants,distor] = quantiz(signal,partition,codebook);
% Convertion of deci  into binary from least  to most significant
indxtrn=index';
for i=1:4000
matrix(i,1:1:8)=bitget(uint8(indxtrn(i)),1:1:8);
end,
% matrix is of  4000 rows X 8 columns
% matrixtps is a matrix of 8 rows X4000 columns
matrixtps=matrix';
% Baseband is produced, it has 32000 bits
baseband=reshape(matrixtps,4000*8,1);
Tb=1/32000;
% bit rate 32 kbps
time=[0:Tb:1];
figure(2);
stairs(time(1:500),baseband(1:500))
title(' A segment of baseband signal')
xlabel('Time(sec)')
ylabel('Binary value')
set(gca,'ytick',[0  1 ])
axis([0,time(500),0,1])
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Serial the data for the next step.
input_to_Convolutional_encoder = baseband'; % 1 X  32000
%Now, the binary converted data is sent to th Convolutional encoder.
t=poly2trellis(7, [171 133]);
%Channel coding
code = convenc(input_to_Convolutional_encoder,t); % 1 x 64000
%Interleaving
st2 = 4831;
data_interleave = randintrlv(code,st2); % Interleave, 1 row x 64000 columns
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Quadrature amplitude modulation
```

```
M=4;
k=log2(M);
baseband=double(baseband);
% bit to symbol mapping
symbol=bi2de(reshape(data_interleave,k,length(data_interleave)/k).','left-msb');
Quadrature_amplitude_modulated_data = qammod(symbol,M);
% demodulation of Quadrature amplitude data
Quadrature_amplitude_demodulated_data = qamdemod(Quadrature_amplitude_modulated_data,M);
[number,ratio]= symerr(symbol,Quadrature_amplitude_demodulated_data) % symbol error
% symbol to  bit mapping
% 2-bit symbol to Binary bit mapping
Retrieved_bit = de2bi(Quadrature_amplitude_demodulated_data,'left-msb');
Retrieved_bit=Retrieved_bit';
Retrieved_bit=reshape(Retrieved_bit, 64000,1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Deinterleaving
errors = zeros(size(Retrieved_bit));
inter_err = bitxor(Retrieved_bit,errors); % Include burst error.
data_deinterleave=randdeintrlv(inter_err,st2);
%Convolutional Decoding
tblen=3;
decodx= vitdec(data_deinterleave,t,tblen,'cont','hard'); %
N3=length(decodx);
NN=N3/8;
decod2(1:(N3-3))=decodx(tblen+1:end);
decod2(N3)=decodx(1);
decod2=decod2' ; % 32000 X 1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
baseband=double(baseband);
[number,ratio]= biterr(decod2,baseband)
convert=reshape(decod2,8,4000); % First reshaping and then transposing
matrixtps=double(matrixtps);
[number,ratio]= biterr(convert,matrixtps)
convert=convert' ; % 4000 rows X 8 columns
% binary to decimally converted value
intconv=bi2de(convert); % converted into interger values(0-255) of 4000 samples
% intconv is 4000 rows X 1 column
[number,ratio]= biterr(intconv,index');
sample_value=minimumvalue +intconv.*interval;
figure(3)
subplot(2,1,1)
plot(time(1:100),signal(1:100));
set(gca,'ytick',[ -1.0   0 1.0 ])
axis([0,time(100),-1,1])
title('Graph for a segment of recoded Audio signal')
xlabel('Time(sec)')
ylabel('Amplitude')
grid on
subplot(2,1,2)
plot(time(1:100),sample_value(1:100));
```

```
axis([0,time(100),-1,1])
set(gca,'ytick',[ -1.0   0 1.0 ])
title('Graph for a segment of retrieved Audio signal')
xlabel('Time(sec)')
ylabel('Amplitude')
grid on
```

%%

Expt 8:Develop a matlab source to simulate an Interleaved FEC encoded wireless communication communication system with implementation of 16-QAM digital modulation technique
Show atleast three waveforms generated at different sections of the simulated system

```
clear all;
close all;
% Test with synthetically generated sinusoidal wave
f=1000;% Frequency of the audio signal
Fs =4000; % Sampling rate is 4000 samples per second.
t = [1/Fs:1/Fs:1];% total time for simulation=0.05 second.
% Number of samples=4000
Am=1.0;
signal = Am*sin(2*pi*1000*t); % Original signal
figure(1);
plot(t(1:200),signal(1:200))
set(gca,'ytick',[ -1.0   0 1.0 ])
title('A segment of synthetically generated sinusiodal wavform')
grid on
xlabel( 'time(sec)');
ylabel( 'Amplitude(volt)');
maximumvalue=max(signal);
minimumvalue=min(signal);
interval=(maximumvalue-minimumvalue)/255; % interval:
partition = [minimumvalue:interval:maximumvalue]; % -1:0.0078:1
codebook = [(minimumvalue-interval):interval:maximumvalue]; % -1.0078:0.0078:1
[index,quants,distor] = quantiz(signal,partition,codebook);
% Convertion of deci  into binary from least  to most significant
indxtrn=index';
for i=1:4000
matrix(i,1:1:8)=bitget(uint8(indxtrn(i)),1:1:8);
end,
% matrix is of  4000 rows X 8 columns
% matrixtps is a matrix of 8 rows X4000 columns
```

```
matrixtps=matrix';
% Baseband is produced, it has 32000 bits
baseband=reshape(matrixtps,4000*8,1);
Tb=1/32000;
% bit rate 32 kbps
time=[0:Tb:1];
figure(2);
stairs(time(1:500),baseband(1:500))
title(' A segment of baseband signal')
xlabel('Time(sec)')
ylabel('Binary value')
set(gca,'ytick',[0  1 ])
axis([0,time(500),0,1])
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Serial the data for the next step.
input_to_Convolutional_encoder = baseband'; % 1 X  32000
%Now, the binary converted data is sent to th Convolutional encoder.
t=poly2trellis(7, [171 133]);
%Channel coding
code = convenc(input_to_Convolutional_encoder,t); % 1 x 64000
%Interleaving
st2 = 4831;
data_interleave = randintrlv(code,st2); % Interleave, 1 row x 64000 columns
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Quadrature amplitude modulation
M=16;
k=log2(M);
baseband=double(baseband);
% bit to symbol mapping
symbol=bi2de(reshape(data_interleave,k,length(data_interleave)/k).','left-msb');
Quadrature_amplitude_modulated_data = qammod(symbol,M);
% demodulation of Quadrature amplitude data
Quadrature_amplitude_demodulated_data = qamdemod(Quadrature_amplitude_modulated_data,M);
[number,ratio]= symerr(symbol,Quadrature_amplitude_demodulated_data) % symbol error
% symbol to  bit mapping
% 2-bit symbol to Binary bit mapping
Retrieved_bit = de2bi(Quadrature_amplitude_demodulated_data,'left-msb');
Retrieved_bit=Retrieved_bit';
Retrieved_bit=reshape(Retrieved_bit, 64000,1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Deinterleaving
errors = zeros(size(Retrieved_bit));
inter_err = bitxor(Retrieved_bit,errors); % Include burst error.
data_deinterleave=randdeintrlv(inter_err,st2);
%Convolutional Decoding
tblen=3;
decodx= vitdec(data_deinterleave,t,tblen,'cont','hard'); %
N3=length(decodx);
NN=N3/8;
decod2(1:(N3-3))=decodx(tblen+1:end);
```

```
decod2(N3)=decodx(1);
decod2=decod2' ; % 32000 X 1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
baseband=double(baseband);
[number,ratio]= biterr(decod2,baseband)
convert=reshape(decod2,8,4000); % First reshaping and then transposing
matrixtps=double(matrixtps);
[number,ratio]= biterr(convert,matrixtps)
convert=convert' ; % 4000 rows X 8 columns
% binary to decimally converted value
intconv=bi2de(convert); % converted into interger values(0-255) of 4000 samples
% intconv is 4000 rows X 1 column
[number,ratio]= biterr(intconv,index');
sample_value=minimumvalue +intconv.*interval;
figure(3)
subplot(2,1,1)
plot(time(1:100),signal(1:100));
set(gca,'ytick',[ -1.0   0 1.0 ])
axis([0,time(100),-1,1])
title('Graph for a segment of recoded Audio signal')
xlabel('Time(sec)')
ylabel('Amplitude')
grid on
subplot(2,1,2)
plot(time(1:100),sample_value(1:100));
axis([0,time(100),-1,1])
set(gca,'ytick',[ -1.0   0 1.0 ])
title('Graph for a segment of retrieved Audio signal')
xlabel('Time(sec)')
ylabel('Amplitude')
grid on
```

system in AWGN channel
DS CDMA system

110) and 7 (binary 111).
length is about 2000 bits).

1}

10 chips per bit.

signal.

2*pi*fc*j) by sqrt(2*eb)*user(i).

ing.

ystem in AWGN and Rayleigh fading channel

)S CDMA system

%

ystem in AWGN and Rician fading channel
DS CDMA system

%

communication systemt

,M);

```
ated_data,M);
```

%%%

%%%