**Pcm**

clc

clear all

close all

%orginal signal

fm=input("enterthe value of frquency fm");

n=input("enter the bo of bits for pcm");

A=2;

t=0:0.01:1;

x=A\*cos(2\*pi\*fm\*t);

%sampling

fa=2\*fm;

ts=0:0.01:1;

xs=A\*cos(2\*pi\*fm\*t);

%quantization

x1=xs+A

x1=x1/(2\*A);

L=(-1+2^n);

x1=L\*x1;

xq=round(x1);

%encoding

y=[];

for i=1:length(xq)

d=dec2bin(xq(i),n);

y=[y double(d)-48];

end

figure(1)

plot(t,x,'Linewidth',2);

title('sampling','fontsize',14);

ylabel('Amplitude rightarrow');

xlabel('time t Rightarrow');

hold on

stem(ts,xs,'r','linewidth',14);

hold off

legend('original signal','sampled signal');

figure(2)

stem(ts,x1-(L/2),'LineWidth',2);

title('quantization','fontsize',14);

ylabel('level L rightarrow');

xlabel('sample Rightarrow');

hold on

stem(ts,xq-(L/2),'r','Linewidth',2);

plot(ts,xq-(L/2),'-k');

hold off

legend('sampled signal ','Quantized signal');

figure(3)

stairs([y y(length(y))],'linewidth',2);

title('encoded waveform','fontsize',14);

ylabel('binary signal');

xlabel('bits')

grid on

**ask**

clc;

clear all;

close all;

Tb=1;

fc=10;

t=0:Tb/100:1;

c=cos(2\*pi\*fc\*t);

m=[1,0,1,0,1,0];

t1=0;

t2=Tb;

for i=1:6

t=[t1:.01:t2]

if m(i)==0

m\_s=zeros(1,length(t));

else

m\_s=ones(1,length(t));

end

msg(i,:)=m\_s;

ask(i,:)=c.\*m\_s;

t1=t1+(Tb+.01);

t2=t2+(Tb+.01);

subplot(5,1,2);

plot(t,msg(i,:),'Linewidth',3);

title('Message-22BEC1456','fontsize',12,'fontweight','bold');

xlabel('t--->','fontsize',12,'fontweight','bold');

ylabel('m(t)','fontsize',12,'fontweight','bold');

hold on

grid on

subplot(5,1,4);

plot(t,ask(i,:),'Linewidth',3);

title('ASK-22BEC1456','fontsize',12,'fontweight','bold');

xlabel('t--->','fontsize',12,'fontweight','bold');

ylabel('s(t)','fontsize',12,'fontweight','bold');

hold on

grid on

end

hold off

subplot(5,1,3);

plot(t,c,'Linewidth',3);

title('Carrier-22BEC1456','fontsize',12,'fontweight','bold');

xlabel('t--->','fontsize',12,'fontweight','bold');

ylabel('c(t)','fontsize',12,'fontweight','bold');

grid on;

subplot(5,1,1);

stem(m,'Linewidth',3);

title('binary data bits-22BEC1456','fontsize',12,'fontweight','bold');

xlabel('n--->','fontsize',12,'fontweight','bold');

ylabel('b(n)','fontsize',12,'fontweight','bold');

grid on;

t1=0;t2=Tb

for i=1:6

t=[t1:Tb/100:t2]

x=sum(c.\*ask(i,:));

if x>0

demod(i)=1;

else

demod(i)=0;

end

t1=t1+(Tb+.01);

t2=t2+(Tb+.01);

end

subplot(5,1,5);

stem(demod,'Linewidth',3);

title('ASK demodulated signal-22BEC1456','fontsize',12,'fontweight','bold');

xlabel('n--->');

ylabel('b(n)');

grid on

**FSK**

clc;

clear all;

close all;

%GENERATE CARRIER SIGNAL

Tb=1; fc1=2;fc2=5;

t=0:(Tb/100):Tb;

c1=sqrt(2/Tb)\*sin(2\*pi\*fc1\*t);

c2=sqrt(2/Tb)\*sin(2\*pi\*fc2\*t);

%generate message signal

N=8;

m=rand(1,N);

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));

invm\_s=zeros(1,length(t));

else

m(i)=0;

m\_s=zeros(1,length(t));

invm\_s=ones(1,length(t));

end

message(i,:)=m\_s;

%Multiplier

fsk\_sig1(i,:)=c1.\*m\_s;

fsk\_sig2(i,:)=c2.\*invm\_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');

title('message signal');xlabel('t---->');ylabel('m(t)');grid on;hold on;

subplot(3,2,5);plot(t,fsk(i,:));

title('FSK signal');xlabel('t---->');ylabel('s(t)');grid on;hold on;

t1=t1+(Tb+.01); t2=t2+(Tb+.01);

end

hold off

%Plotting binary data bits and carrier signal

subplot(3,2,1);stem(m);

title('binary data');xlabel('n---->'); ylabel('b(n)');grid on;

subplot(3,2,3);plot(t,c1);

title('carrier signal-1');xlabel('t---->');ylabel('c1(t)');grid on;

subplot(3,2,4);plot(t,c2);

title('carrier signal-2');xlabel('t---->');ylabel('c2(t)');grid on;

13

% FSK Demodulation

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---->');ylabel('b(n)'); grid on;

**PSK**

clc;

clear all;

close all;

Tb=1;

t=0:Tb/100:Tb;

fc=2;

c=sqrt(2/Tb)\*sin(2\*pi\*fc\*t);

N=8;

m=rand(1,N);

t1=0;t2=Tb

for i=1:N

t=[t1:.01: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

message(i,:)=m\_s;

bpsk\_sig(i,:)=c.\*m\_s;

subplot(5,1,2);axis([0 N -2 2]);plot(t,message(i,:),'r','Linewidth',2);

set(gca,'fontsize',12,'fontweight','bold');

title('message signal(POLAR form)-22BEC1444','fontsize',12,'fontweight','bold');

xlabel('t--->');ylabel('m(t)');

grid on;

hold on;

subplot(5,1,4);plot(t,bpsk\_sig(i,:),'Linewidth',2);

title('BPSK signal-22BEC1444','fontsize',12,'fontweight','bold');

xlabel('t--->','fontsize',12,'fontweight','bold');

ylabel('s(t)','fontsize',12,'fontweight','bold');

grid on;

hold on;

t1=t1+1.01; t2=t2+1.01;

end

hold off

subplot(5,1,1);stem(m,'Linewidth',2);

set(gca,'fontsize',12,'fontweight','bold');

title('binary data bits-22BEC1444','fontsize',12,'fontweight','bold');

xlabel('n--->','fontsize',12,'fontweight','bold');

ylabel('b(n)','fontsize',12,'fontweight','bold');

grid on;

subplot(5,1,3);plot(t,c,'Linewidth',2);

set(gca,'fontsize',12,'fontweight','bold');

title('carrier signal-22BEC1444','fontsize',12,'fontweight','bold');

xlabel('t--->','fontsize',12,'fontweight','bold');

ylabel('c(t)','fontsize',12,'fontweight','bold');

grid on;

t1=0;t2=Tb

for i=1:N

t=[t1:.01:t2]

x=sum(c.\*bpsk\_sig(i,:));

if x>0

demod(i)=1;

else

demod(i)=0;

end

t1=t1+1.01;

t2=t2+1.01;

end

subplot(5,1,5);stem(demod,'Linewidth',2);

set(gca,'fontsize',12,'fontweight','bold');

title('demodulated data-22BEC1444','fontsize',12,'fontweight','bold');

xlabel('n--->','fontsize',12,'fontweight','bold');

ylabel('b(n)','fontsize',12,'fontweight','bold');

grid on

**DUOBINARY**

clc;

close all;

clear all;

b = [1 0 1 0 1 0 1 0];

n = length(b);

a = zeros(1, n);

a(1) = 1;

% binary seq ak

for i = 2:n+1

a(i) = xor(a(i-1), b(i-1));

end

precoding=zeros(1,n+1);

for i=1:n+1

if a(i)==0

precoding(i)=-1;

end

if a(i)==1

precoding(i)=1;

end

end

duobinary =zeros(1,n);

for i=1:n

duobinary(i) =precoding(i)+precoding(i+1);

end

seq\_out=zeros(1,n);

for i=1:n

if duobinary(i)==0

seq\_out(i) = 1;

end

if duobinary(i)==2||duobinary(i)==-2

seq\_out(i) = 0;

end

end

%disp('binary seq bk');

%disp(b);

%disp('binary seq ak');

%disp(a);

%disp('precoding');

%disp(precoding);

%disp('duobinary');

%disp(duobinary);

%disp('seq output');

%disp(seq\_out);

figure;

subplot(5, 1, 1);

stem(b, 'LineWidth', 2);

title('Binary Sequence b');

xlabel('Sample Index');

ylabel('Value');

ylim([0, 1]);

grid on;

subplot(5, 1, 2);

stairs(a(1:n), 'LineWidth', 2);

title('Precoded Sequence ak');

xlabel('Sample Index');

ylabel('Value');

ylim([0, 1]);

grid on;

subplot(5, 1, 3);

stairs(precoding, 'LineWidth', 2);

title('Precoded Values');

xlabel('Sample Index');

ylabel('Value');

ylim([-1, 1]);

grid on;

subplot(5, 1, 4);

stairs(duobinary, 'LineWidth', 2);

title('duobinary Values');

xlabel('Sample Index');

ylabel('Value');

ylim([-2, 2]);

grid on;

subplot(5, 1, 5);

stairs(seq\_out, 'LineWidth', 2);

title('Output Sequence');

xlabel('Sample Index');

ylabel('Value');

ylim([0, 1]);

grid on;

sgtitle('Duobinary Encoding and Output Sequences');

**PN SEQUENCE**

clc;

clear all;

close all;

% Initial binary input

binaryin = [1 0 0 0];

seq = zeros(1, 2^(length(binaryin)) - 1);

for i = 1:length(seq)

% Generate XOR of specific bits (for 4-bit LFSR, using feedback from 1st and 4th bits)

y = xor(binaryin(1), binaryin(4));

% Store the output of the last bit in the sequence

seq(i) = binaryin(4);

% Shift bits to the right

for j = length(binaryin):-1:2

binaryin(j) = binaryin(j-1);

end

% Update the first bit with XOR result

binaryin(1) = y;

end

% Plotting the PN sequence

stem(seq, 'filled');

title('4-bit PN Sequence');

xlabel('Sequence Index');

ylabel('Output Bit');

**PSD**

clc;

clear all;

close all;

% Parameters

bitrate = 1;

bits = [1 1 1 0 0 1 1 0 1];

Tb = 1 / bitrate;

V = 1;

f = 0:bitrate / 100:2 \* bitrate;

% PSD Calculations

PSD\_unipolar\_nrz = (V^2 \* Tb / 2) \* sinc(f \* Tb).^2 + (V^2 \* Tb / 4) \* sinc(f \* Tb).^2;

PSD\_polar\_nrz = (V^2 \* Tb) \* sinc(f \* Tb).^2;

PSD\_bipolar\_nrz = (V^2 \* Tb) \* sinc(f \* Tb).^2 .\* (sin(pi \* f \* Tb).^2);

PSD\_manchester = (V^2 / Tb) \* (sinc(f \* Tb / 2).^2) .\* (sin(pi \* f \* Tb / 2).^2);

% Plotting

figure;

plot(f, PSD\_unipolar\_nrz, 'r', 'LineWidth', 2);

hold on;

plot(f, PSD\_polar\_nrz, 'g', 'LineWidth', 2);

plot(f, PSD\_bipolar\_nrz, 'b', 'LineWidth', 2);

plot(f, PSD\_manchester, 'k', 'LineWidth', 2);

hold off;

% Plot Labels

title('Power Spectral Density of Various Line Codes');

xlabel('Frequency (Hz)');

ylabel('Power Spectral Density (V^2)');

legend('Unipolar NRZ', 'Polar NRZ', 'Bipolar NRZ', 'Manchester');

grid on;

**TDM**

clc;

close all;

clear all;

% Signal generation

x = 0:0.1:4\*pi;

sig1 = 8 \* sin(x);

l = length(sig1);

sig2 = 8 \* sawtooth(x, 0.5);

sig3 = 8 \* square(x);

% Display of All Three Signals

figure;

subplot(3, 2, 1);

plot(sig1);

title('Sinusoidal Signal');

ylabel('Amplitude');

xlabel('Time');

subplot(3, 2, 2);

plot(sig2);

title('Triangular Signal');

ylabel('Amplitude');

xlabel('Time');

subplot(3, 2, 3);

plot(sig3);

title('Square Signal');

ylabel('Amplitude');

xlabel('Time');

% Display of All Three Sampled Signals

subplot(3, 2, 4);

stem(sig1, 'r');

title('Sampled Sinusoidal Signal');

ylabel('Amplitude');

xlabel('Time');

subplot(3, 2, 5);

stem(sig2, 'b');

title('Sampled Triangular Signal');

ylabel('Amplitude');

xlabel('Time');

subplot(3, 2, 6);

stem(sig3, 'g');

title('Sampled Square Signal');

ylabel('Amplitude');

xlabel('Time');

% Combine all three signals into a matrix

sig = zeros(3, l);

sig(1, :) = sig1;

sig(2, :) = sig2;

sig(3, :) = sig3;

% TDM of all three quantized signals

tdmsig = reshape(sig, 1, 3 \* l);

% Display of TDM Signal

figure;

hold on;

stem(1:3:3\*l, sig1, 'r', 'filled');

stem(2:3:3\*l, sig2, 'b', 'filled');

stem(3:3:3\*l, sig3, 'g', 'filled');

hold off;

title('TDM Signal');

ylabel('Amplitude');

xlabel('Time');

grid on;

**%bipolar(PSD)**

clc;

clear all;

bitrate=1; %Rb-bitrate

bits=[1 0 0 1 0 1 1 1 0 1 0];

T= length(bits)/bitrate; %Symbol duration

n=200; %bit duration

dt=1/n;

TB=1/bitrate; %1/bitrate value

a=1;

f=0:bitrate/100:2\*bitrate;

x1=f\*TB; % power spectra frequency

t=0:dt:T; %t takes the valu 0 to T with innerspace equal to dt

x=zeros(1,length(t)); %initialisation of x

count=0;

%line coding :BI-polar

for(i=0:length(bits)-1)

if bits(i+1)==1

if mod(count,2)==0

x((i\*n)+1:(i+1)\*n)=1;

else

x((i\*n)+1:(i+1)\*n)=-1;

end

count=count+1;

else

x((i\*n)+1:(i+1)\*n)=0;

end

end

subplot(2,1,1)

plot(x);

xlabel('Descrete time');

ylabel('Amplitude');

title('BI-POLAR-LINE CODING')

axis([0 2500 -2 2])

sx=(a^2)\*TB\*(sinc(x1)).\*(sinc(x1)).\*(sin(pi\*x1)).\*(sin(pi\*x1)) %power spectra BIPOLAR formila

subplot(2,1,2)

plot(sx);

xlabel('Frequency');

ylabel('Power');

title(' BI-POLAR-POWER SPECTRUM-LINE CODING')

**%unipolar(PSD)**

clc;

clear all;

bitrate=1; %Rb-bitrate

bits=[1 0 0 1 0 1 1 1 0 1 0];

T= length(bits)/bitrate; %Symbol duration

n=200; %bit duration

dt=1/n;

TB=1/bitrate; %1/bitrate value

a=1;

f=0:bitrate/100:2\*bitrate;

x1=f\*TB; % power spectra frequency

t=0:dt:T; %t takes the valu 0 to T with innerspace equal to dt

x=zeros(1,length(t)); %initialisation of x

%line coding :Uni-polar

for(i=0:length(bits)-1)

if bits(i+1)==1

x((i\*n)+1:(i+1)\*n)=1;

else

x((i\*n)+1:(i+1)\*n)=0;

end

end

subplot(2,1,1)

plot(x);

xlabel('Descrete time');

ylabel('Amplitude');

title('NRZ UNI POLAR-LINE CODING')

axis([0 2500 0 2])

sx=((a^2)/4)\*TB\*(sinc(x1).^2)+((a^2)/4)\*dirac(f); %power spectra UNIPOLAR formila

subplot(2,1,2)

plot(sx);

xlabel('Frequency');

ylabel('Power');

title('NRZ UNI POLAR-POWER SPECTRUM-LINE CODING')

**%polar(PSD)**

clc;

clear all;

bitrate=1; %Rb-bitrate

bits=[1 0 0 1 0 1 1 1 0 1 0];

T= length(bits)/bitrate; %Symbol duration

n=200; %bit duration

dt=1/n;

TB=1/bitrate; %1/bitrate value

a=1;

f=0:bitrate/100:2\*bitrate;

x1=f\*TB; % power spectra frequency

t=0:dt:T; %t takes the valu 0 to T with innerspace equal to dt

x=zeros(1,length(t)); %initialisation of x

%line coding -polar

for(i=0:length(bits)-1)

if bits(i+1)==1

x((i\*n)+1:(i+1)\*n)=1;

else

x((i\*n)+1:(i+1)\*n)=-1;

end

end

subplot(2,1,1)

plot(x);

xlabel('Descrete time');

ylabel('Amplitude');

title('NRZ POLAR-LINE CODING')

axis([0 2500 0 2]);

sx=(a^2)\*TB\*(sinc(x1).^2); %power spectra POLAR formila

subplot(2,1,2)

plot(sx);

xlabel('Frequency');

ylabel('Power');

title('NRZ POLAR-POWER SPECTRUM-LINE CODING')

**MANCHESTER(PSD)**

clc;

clear all;

bitrate=1; %Rb-bitrate

bits=[1 0 0 1 0 1 1 1 0 1 0];

T= length(bits)/bitrate; %Symbol duration

n=200; %bit duration

dt=1/n;

TB=1/bitrate; %1/bitrate value

a=1;

f=0:bitrate/100:2\*bitrate;

x1=f\*TB; % power spectra frequency

t=0:dt:T; %t takes the valu 0 to T with innerspace equal to dt

x=zeros(1,length(t)); %initialisation of x

%line coding :MANCHESTER

for(i=0:length(bits)-1)

if bits(i+1)==1

x((i\*n)+1:(i+0.5)\*n)=1;

x(((i+0.5)\*n)+1:(i+1)\*n)=-1;

else

x((i\*n)+1:(i+0.5)\*n)=-1;

x(((i+0.5)\*n)+1:(i+1)\*n)=1;

end

end

subplot(2,1,1)

plot(x);

xlabel('Descrete time');

ylabel('Amplitude');

title('MANCHESTER-LINE CODING')

axis([0 2500 -2 2])

sx=(a^2)\*TB\*(sinc(x1/2)).\*(sinc(x1/2)).\*(sin(pi\*x1/2)).\*(sin(pi\*x1/2)); %power spectra MANCHESTER formila

subplot(2,1,2)

plot(sx);

xlabel('Frequency');

ylabel('Power');

title('MANCHESTER-POWER SPECTRUM-LINE CODING')