***Noakhali Science and Technology University***

Noakhali-3814

**Course Title** : Data Communication Lab

Course Code : ICE-2204

|  |
| --- |
| Submission Date : 15.12.2019 |

|  |  |
| --- | --- |
| Submitted to : | Submitted by : |
| Name: Tanvir Zaman Khan,  Lecturer  Apurba Adhikary,  Lecturer  **Depertment of** : Information & Communication Engineering  **Noakhali Science and Technology University** | Name : **Abdul Awal Nadim**  **Roll** : **ASH - 1811056M**  Year : 2, Term : 2  **Department Of :** Information & Communication Engineering  **Noakhali Science and Technology University** |

Lab Report

Write a program to generate

Amplitude Modulation

**Introduction:** **Amplitude modulation** (**AM**) is a [modulation](https://en.wikipedia.org/wiki/Modulation) technique used in electronic communication, most commonly for transmitting information via a [radio](https://en.wikipedia.org/wiki/Radio) [carrier wave](https://en.wikipedia.org/wiki/Carrier_wave). In amplitude modulation, the [amplitude](https://en.wikipedia.org/wiki/Amplitude) (signal strength) of the carrier wave is varied in proportion to that of the message signal being transmitted. The message signal is, for example, a function of the sound to be reproduced by a [loudspeaker](https://en.wikipedia.org/wiki/Loudspeaker), or the light intensity of pixels of a television screen. This technique contrasts with [frequency modulation](https://en.wikipedia.org/wiki/Frequency_modulation), in which the [frequency](https://en.wikipedia.org/wiki/Frequency) of the [carrier signal](https://en.wikipedia.org/wiki/Carrier_signal) is varied, and [phase modulation](https://en.wikipedia.org/wiki/Phase_modulation), in which its [phase](https://en.wikipedia.org/wiki/Phase_(waves)) is varied.

Am=input('enter message signal amplitude: ');

Ac=input('enter carrier signal amplitude: ');

fc=input('enter carrier frequency: ');

fm=input('enter message frequency: ');% fm<fc

m=input('enter modulation index: ');

t=input('enter time period: ');

%Making signal

t1=linspace(0,t,1000);

y1=sin(2\*pi\*fm\*t1); % message signal

y2=sin(2\*pi\*fc\*t1); % carrier signal

eq=(1+m.\*y1).\*(Ac.\*y2);

s % Showing Modulated Signal

subplot(313);

plot(t1,eq);

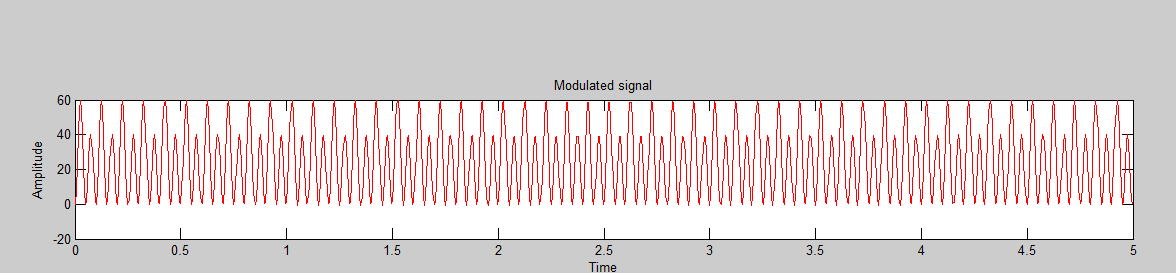
plot(t1,eq,'r');

xlabel('Time');

ylabel('Amplitude');

title('Modulated signal');

**Output**



Write a program to generate

Frequency Modulation

**Introduction:** In [telecommunications](https://en.wikipedia.org/wiki/Telecommunications) and [signal processing](https://en.wikipedia.org/wiki/Signal_processing), **frequency modulation** (**FM**) is the encoding of [information](https://en.wikipedia.org/wiki/Information) in a [carrier wave](https://en.wikipedia.org/wiki/Carrier_wave) by varying the [instantaneous frequency](https://en.wikipedia.org/wiki/Instantaneous_frequency) of the wave. [Digital data](https://en.wikipedia.org/wiki/Digital_data) can be encoded and transmitted via FM by shifting the carrier's frequency among a predefined set of frequencies representing digits  – for example one frequency can represent a [binary](https://en.wikipedia.org/wiki/Binary_number) 1 and a second can represent binary 0. This modulation technique is known as [frequency-shift keying](https://en.wikipedia.org/wiki/Frequency-shift_keying) (FSK). FSK is widely used in [modems](https://en.wikipedia.org/wiki/Modem) such as [fax modems](https://en.wikipedia.org/wiki/Fax_modem), and can also be used to send [Morse code](https://en.wikipedia.org/wiki/Morse_code).[[1]](https://en.wikipedia.org/wiki/Frequency_modulation#cite_note-1) [Radioteletype](https://en.wikipedia.org/wiki/Radioteletype" \o "Radioteletype) also uses FSK

m1 = input('Enter modulation index value');

fm = input('frequency of message signal');

fc = input('frequency of carrier signal');

t = 0 : 0.0001 : 0.1 ;

m = sin(2\*pi\*fm\*t);

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

y = sin(2\*pi\*fc\*t+(m1.\*sin(2\*pi\*fm\*t)));

subplot(3,1,1);

plot(t,m);

title('Message signal m');

xlabel('...time...');

ylabel('...amplitude...');

grid on;

subplot(3,1,2);

plot(t,c);

title('Carrier signal c');

xlabel('...time...');

ylabel('...amplitude...');

grid on;

subplot(3,1,3);

plot(t,y);

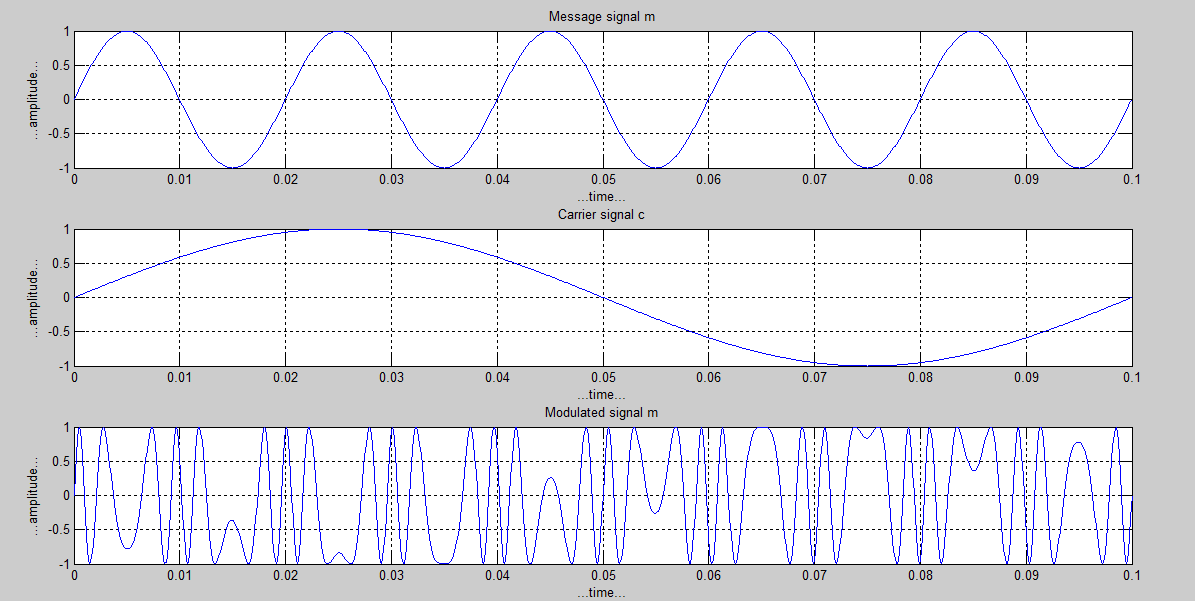
title('Modulated signal m');

xlabel('...time...');

ylabel('...amplitude...');

grid on;

**Output**



Write a program to generate

Phase Modulation

**Introduction:** **Phase modulation** (**PM**) is a [modulation](https://en.wikipedia.org/wiki/Modulation) pattern for conditioning communication signals for [transmission](https://en.wikipedia.org/wiki/Transmission_(telecommunications)). It encodes a message signal as variations in the [instantaneous phase](https://en.wikipedia.org/wiki/Instantaneous_phase) of a [carrier wave](https://en.wikipedia.org/wiki/Carrier_wave). Phase modulation is one of the two principal forms of [angle modulation](https://en.wikipedia.org/wiki/Angle_modulation), together with [frequency modulation](https://en.wikipedia.org/wiki/Frequency_modulation).

t = 0:0.01:3;

fc = 10;

m = sin(2\*pi\*t);

kp = pi/2; % phase deviation constant

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

modulated = cos(2\*pi\*fc\*t + kp\*m);

subplot (3,1,1);

plot (t,m);

axis([0 3 -1.5 1.5]);

xlabel('Time(seconds)');

ylabel('Amplitude(volt)');

title('Message');

grid on;

subplot (3,1,2);

plot (t,carrier);

axis([0 3 -1.5 1.5]);

xlabel('Time(seconds)');

ylabel('Amplitude(volt)');

title('Carrier Signal');

grid on;

subplot (3,1,3);

plot (t,modulated);

axis([0 3 -1.5 1.5]);

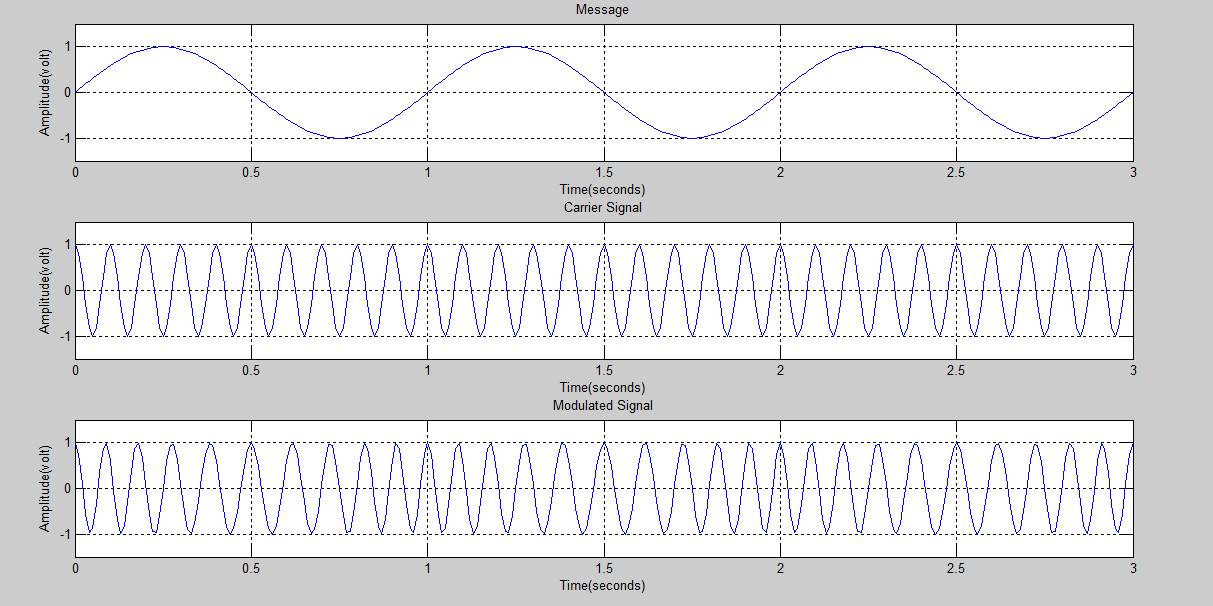
xlabel('Time(seconds)');

ylabel('Amplitude(volt)');

title('Modulated Signal');

grid on;

**Output**



Write a program to generate

Amplitude Shift Keying

**Introduction:** **Amplitude-shift keying** (**ASK**) is a form of [amplitude modulation](https://en.wikipedia.org/wiki/Amplitude_modulation) that represents [digital](https://en.wikipedia.org/wiki/Digital_data) [data](https://en.wikipedia.org/wiki/Data) as variations in the [amplitude](https://en.wikipedia.org/wiki/Amplitude) of a [carrier wave](https://en.wikipedia.org/wiki/Carrier_wave). In an ASK system, the binary symbol 1 is represented by transmitting a fixed-amplitude carrier wave and fixed frequency for a bit duration of T seconds. If the signal value is 1 then the carrier signal will be transmitted; otherwise, a signal value of 0 will be transmitted.

fm=input('enter the message frequency :');

fc=input('enter the carrier frequency :');

A=3;

t=0:0.001:1;

m=(A.\*square(2\*pi\*fm\*t)+A)/2; %1 works for positive value,-1 works for negative value

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

s=m.\*c;

subplot(3,1,1); %create 3 subplot by grid 1 and at position 1

plot(t,m);

title('Square Wave');

ylim([-1 4]);

xlabel('Time');

ylabel('Amplitude');

grid;

subplot(3,1,2);

plot(t,c);

title('Carrier Wave');

ylim([-5 5]);

xlabel('Time');

ylabel('Amplitude');

grid;

subplot(3,1,3);

plot(t,s);

title('Modulating Wave');

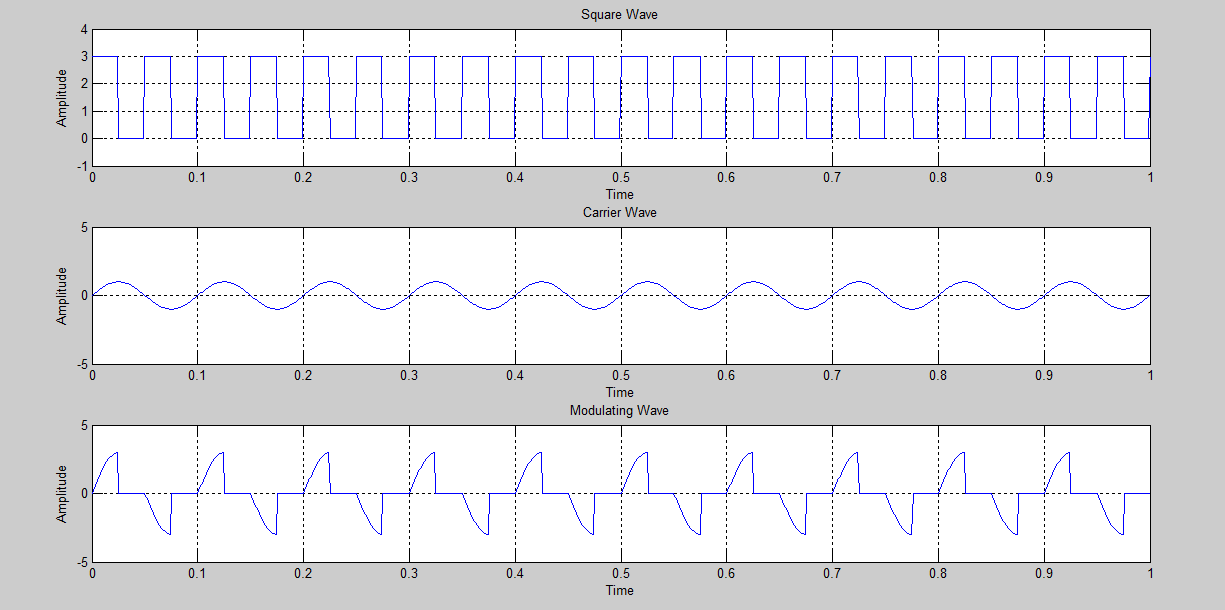
ylim([-5 5]);

xlabel('Time');

ylabel('Amplitude');

grid;

**Output**



Write a program to generate

Frequency Shift Keying

**Introduction:** **Frequency-shift keying** (**FSK**) is a [frequency modulation](https://en.wikipedia.org/wiki/Frequency_modulation) scheme in which digital information is transmitted through discrete frequency changes of a [carrier signal](https://en.wikipedia.org/wiki/Carrier_signal).[[1]](https://en.wikipedia.org/wiki/Frequency-shift_keying#cite_note-1) The technology is used for communication systems such as [telemetry](https://en.wikipedia.org/wiki/Telemetry), weather balloon [radiosondes](https://en.wikipedia.org/wiki/Radiosonde" \o "Radiosonde), [caller ID](https://en.wikipedia.org/wiki/Caller_ID), [garage door openers](https://en.wikipedia.org/wiki/Garage_door_opener), and low frequency radio transmission in the [VLF](https://en.wikipedia.org/wiki/Very_low_frequency) and [ELF](https://en.wikipedia.org/wiki/Extremely_low_frequency) bands.

fc1=input('Enter the freq of 1st Sine Wave carrier:');

fc2=input('Enter the freq of 2nd Sine Wave carrier:');

fp=input('Enter the freq of Periodic Binary pulse (Message):');

amp=input('Enter the amplitude (For Both Carrier & Binary Pulse Message):');

amp=amp/2;

t=0:0.001:1; % For setting the sampling interval

c1=amp.\*sin(2\*pi\*fc1\*t);% For Generating 1st Carrier Sine wave

c2=amp.\*sin(2\*pi\*fc2\*t);% For Generating 2nd Carrier Sine wave

subplot(4,1,1); %For Plotting The Carrier wave

plot(t,c1)

xlabel('Time')

ylabel('Amplitude')

title('Carrier 1 Wave')

subplot(4,1,2) %For Plotting The Carrier wave

plot(t,c2)

xlabel('Time')

ylabel('Amplitude')

title('Carrier 2 Wave')

m=amp.\*square(2\*pi\*fp\*t)+amp;%For Generating Square wave message

subplot(4,1,3) %For Plotting The Square Binary Pulse (Message)

plot(t,m)

xlabel('Time')

ylabel('Amplitude')

title('Binary Message Pulses')

for i=0:1000 %here we are generating the modulated wave

if m(i+1)==0

mm(i+1)=c2(i+1);

else

mm(i+1)=c1(i+1);

end

end

subplot(4,1,4) %For Plotting The Modulated wave

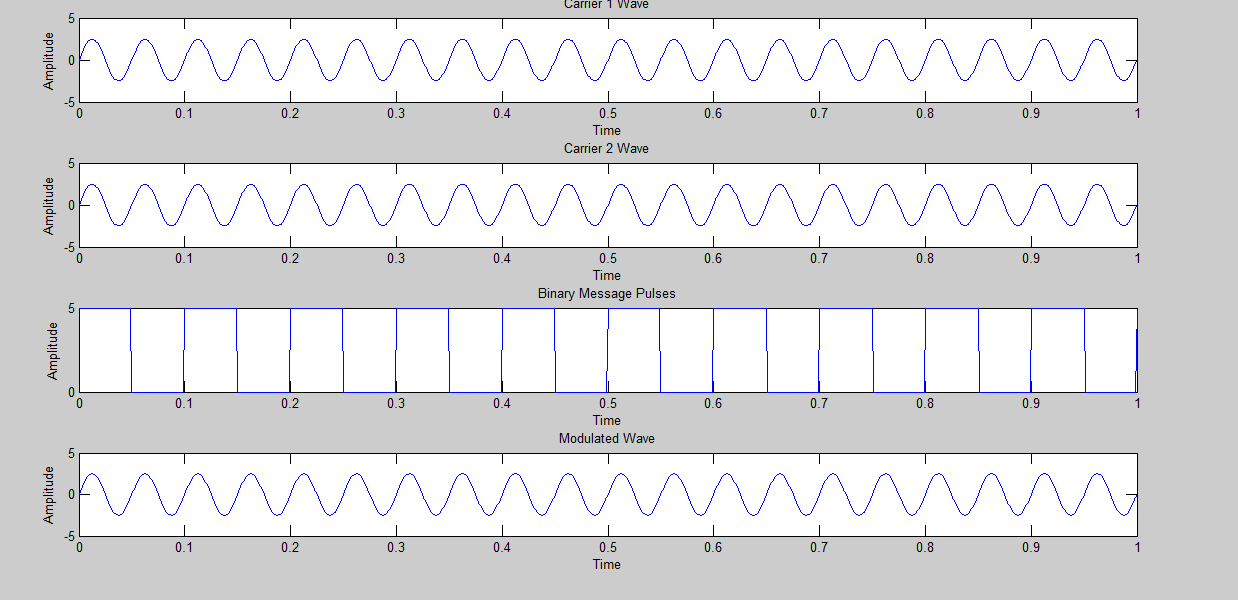
plot(t,mm)

xlabel('Time')

ylabel('Amplitude')

title('Modulated Wave')

**Output**



Write a program to generate

Phase Shift Keying

**Introduction:** **Phase-shift keying** (**PSK**) is a [digital modulation](https://en.wikipedia.org/wiki/Digital_modulation) process which conveys [data](https://en.wikipedia.org/wiki/Data#Uses_of_data_in_computing) by changing (modulating) the [phase](https://en.wikipedia.org/wiki/Phase_(waves)) of a constant [frequency](https://en.wikipedia.org/wiki/Frequency) reference [signal](https://en.wikipedia.org/wiki/Signal_(information_theory)) (the [carrier wave](https://en.wikipedia.org/wiki/Carrier_wave)). The modulation is accomplished by varying the [sine](https://en.wikipedia.org/wiki/Sine_wave) and [cosine](https://en.wikipedia.org/wiki/Cosine_wave) inputs at a precise time. It is widely used for [wireless LANs](https://en.wikipedia.org/wiki/Wireless_LAN), [RFID](https://en.wikipedia.org/wiki/RFID) and [Bluetooth](https://en.wikipedia.org/wiki/Bluetooth) communication.

fc=input('Enter frequency of Carrier Sine wave: ');

fm=input('Enter Message frequency : ');

amp=input('Enter Carrier & Message Amplitude(Assuming Both Equal):');

c=amp.\*sin(2\*pi\*fc\*t);% Generating Carrier Sine

subplot(3,1,1) %For Plotting The Carrier wave

plot(t,c)

xlabel('Time')

ylabel('Amplitude')

title('Carrier')

m=square(2\*pi\*fm\*t);% For Plotting Message signal

subplot(3,1,2)

plot(t,m)

xlabel('time')

ylabel('ampmplitude')

title('Message Signal')% Sine wave multiplied with square wave in order to generate PSK

x=c.\*m;

subplot(3,1,3) % For Plotting PSK (Phase Shift Keyed) signal

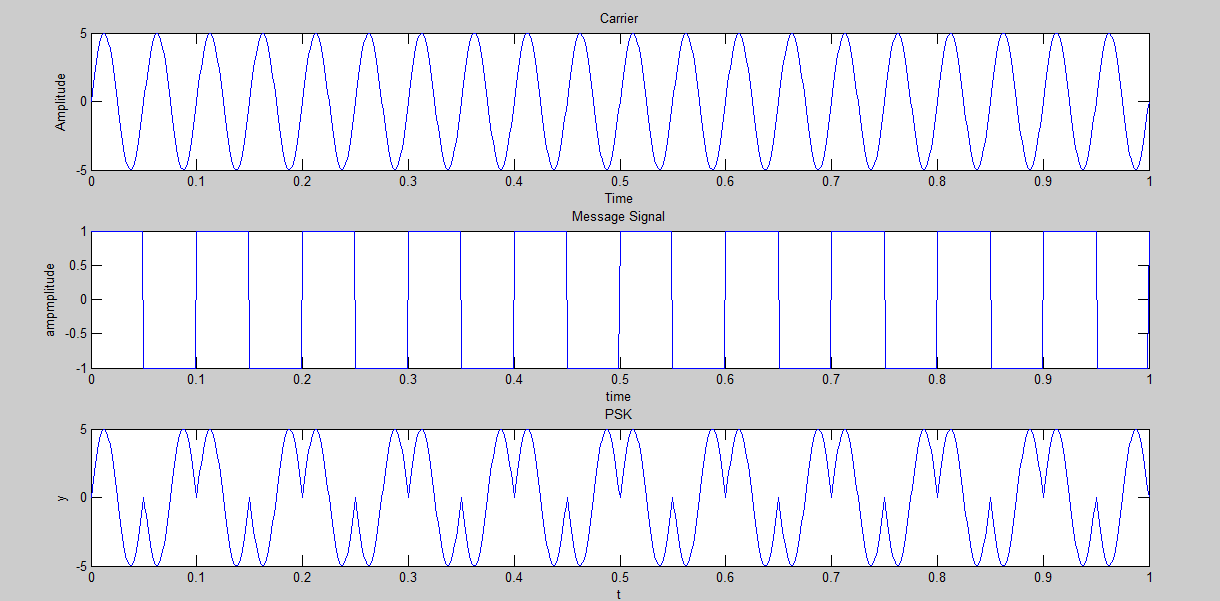
plot(t,x)

xlabel('t')

ylabel('y')

title('PSK')

**Output**



Write a program to generate Unipolar : Non-Return-to Zero(NRZ)

**Introduction:** In [telecommunication](https://en.wikipedia.org/wiki/Telecommunication), a **non-return-to-zero** (**NRZ**) [line code](https://en.wikipedia.org/wiki/Line_code) is a [binary](https://en.wikipedia.org/wiki/Binary_coding) code in which ones are represented by one [significant condition](https://en.wikipedia.org/wiki/Significant_condition), usually a positive voltage, while zeros are represented by some other significant condition, usually a negative voltage, with no other neutral or rest condition. The pulses in NRZ have more energy than a [return-to-zero](https://en.wikipedia.org/wiki/Return-to-zero) (RZ) code, which also has an additional rest state beside the conditions for ones and zeros.

n=[1,0,1,0,1,1,1,0]

%mapping

for i1=1:length(n)

if n(i1)==1;

nn(i1)=3;

else

nn(i1)=0;

end

end

i=1;

t=0:0.001:length(n);

for j=1:length(t)

if t(j)<=i

y(j)=nn(i);

else

y(j)=nn(i);

i=i+1;

end

end

%plotting

stairs(t,y);

grid;

%plot(t,y,'-r');

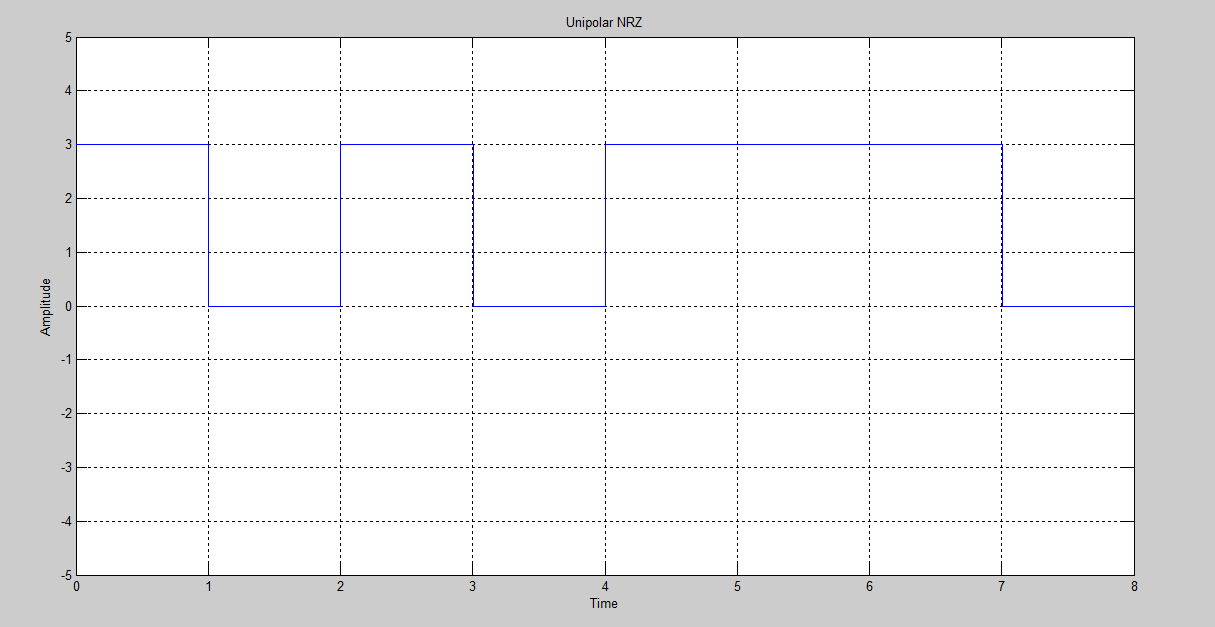
ylim([-5 5]);

xlabel('Time');

ylabel('Amplitude');

title('Unipolar NRZ');

**Output**



Write a program to generate polar: Non-Return-to Zero(NRZ-I)

**Introduction:** In [telecommunication](https://en.wikipedia.org/wiki/Telecommunication), a **non-return-to-zero** (**NRZ**) [line code](https://en.wikipedia.org/wiki/Line_code) is a [binary](https://en.wikipedia.org/wiki/Binary_coding) code in which ones are represented by one [significant condition](https://en.wikipedia.org/wiki/Significant_condition), usually a positive voltage, while zeros are represented by some other significant condition, usually a negative voltage, with no other neutral or rest condition. The pulses in NRZ have more energy than a [return-to-zero](https://en.wikipedia.org/wiki/Return-to-zero) (RZ) code, which also has an additional rest state beside the conditions for ones and zeros.

x=[ 0 1 0 0 1 1 0 0 0 1 1 ];

T=length(x);

n=200;

N=n\*T;

dt=T/N;

pulse=-1;

t=0:dt:T;

y=zeros(1,length(t));

for i=0:T-1;

if x(i+1)==1

if pulse==1

pulse=-1;

y((i\*n)+1 : (i+1)\*n)=pulse;

else

pulse=1;

y((i\*n)+1 : (i+1)\*n)=pulse;

end;

else

y((i\*n)+1 : (i+1)\*n)=pulse;

end;

end;

plot(t,y);

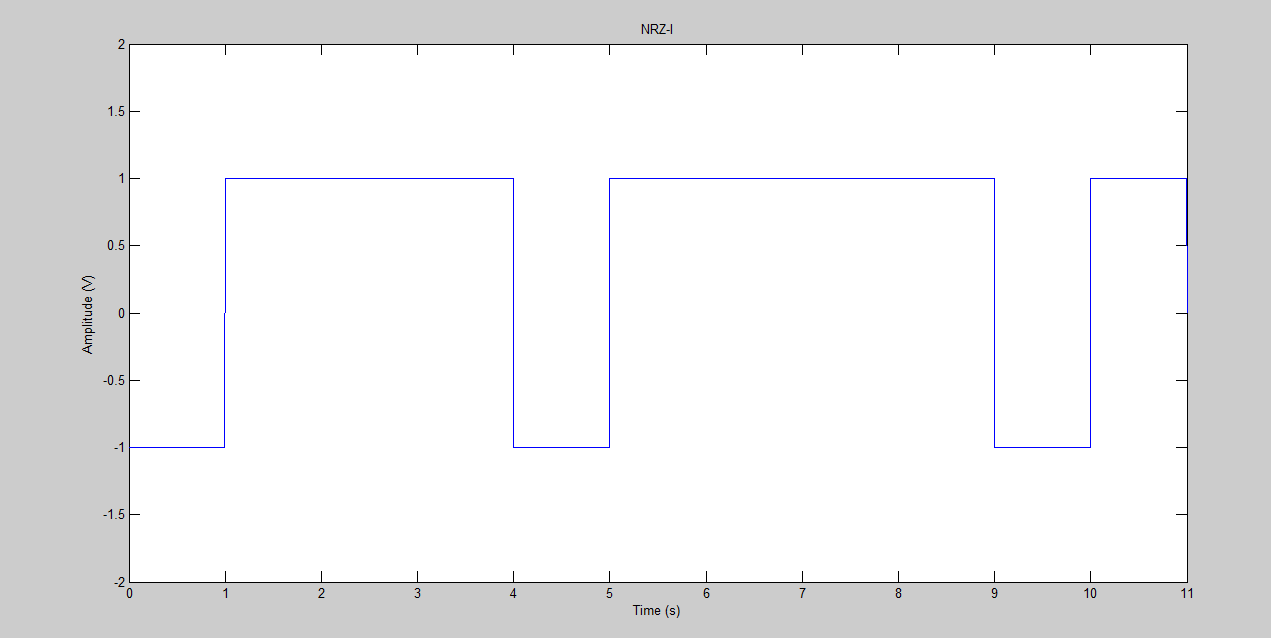
axis([0 11, -2 2]);

title('NRZ-I');

xlabel('Time (s)') ;

ylabel('Amplitude (V)');

**Output**



Write a program to generate polar: Non-Return-to Zero(NRZ-L)

**Introduction:** In [telecommunication](https://en.wikipedia.org/wiki/Telecommunication), a **non-return-to-zero** (**NRZ**) [line code](https://en.wikipedia.org/wiki/Line_code) is a [binary](https://en.wikipedia.org/wiki/Binary_coding) code in which ones are represented by one [significant condition](https://en.wikipedia.org/wiki/Significant_condition), usually a positive voltage, while zeros are represented by some other significant condition, usually a negative voltage, with no other neutral or rest condition. The pulses in NRZ have more energy than a [return-to-zero](https://en.wikipedia.org/wiki/Return-to-zero) (RZ) code, which also has an additional rest state beside the conditions for ones and zeros

x=[ 0 1 0 0 1 1 0 0 0 1 1 ];

T=length(x); n=10000;

N=n\*T;

dt=T/N;

t=0:dt:T;

y=zeros(1,length(t));

for i=0:T-1;

if x(i+1)==1

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

else

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

end;

end;

plot(t,y);

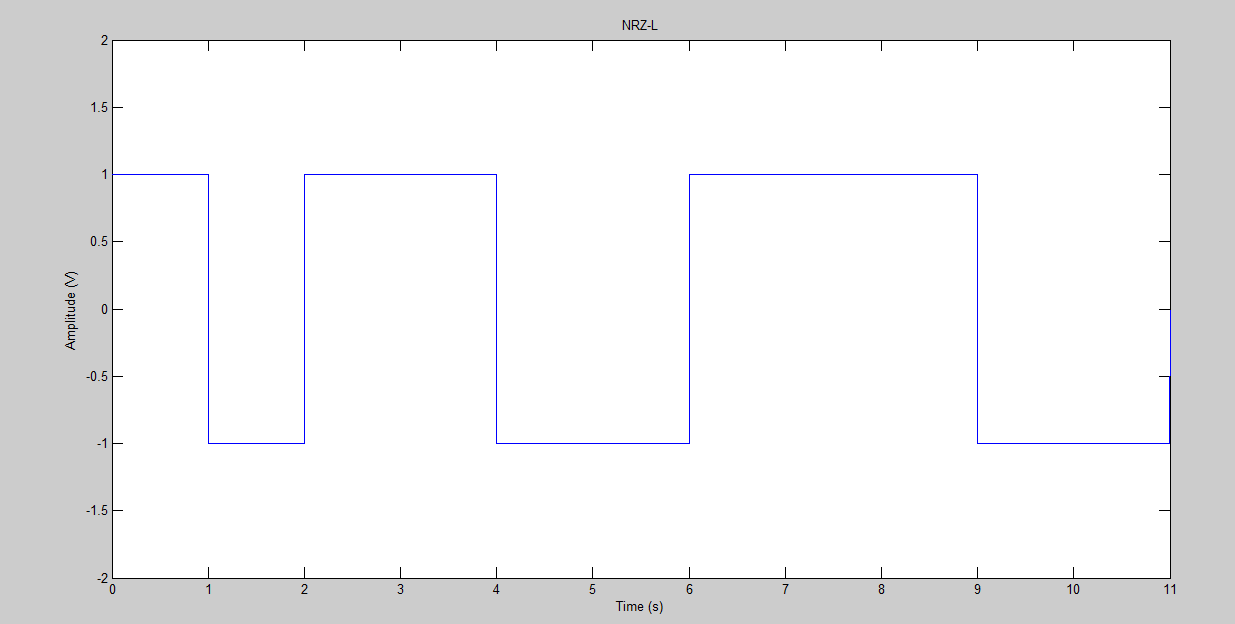
axis([0 11 , -2 2]);

title('NRZ-L');

xlabel('Time (s)')

ylabel('Amplitude (V)')

**Output**



Write a program to generate Polar : Return-to Zero(RZ)

**Introduction:** **Return-to-zero** (**RZ** or **RTZ**) describes a [line code](https://en.wikipedia.org/wiki/Line_code) used in [telecommunications](https://en.wikipedia.org/wiki/Telecommunication) [signals](https://en.wikipedia.org/wiki/Signal_(information_theory)) in which the signal drops (returns) to zero between each [pulse](https://en.wikipedia.org/wiki/Pulse_(signal_processing)). This takes place even if a number of consecutive 0s or 1s occur in the signal. The signal is [self-clocking](https://en.wikipedia.org/wiki/Self-clocking_signal). This means that a separate clock does not need to be sent alongside the signal, but suffers from using twice the bandwidth to achieve the same data-rate as compared to [non-return-to-zero](https://en.wikipedia.org/wiki/Non-return-to-zero) format.

N=4;

disp('enter the binary: ');

for a=1:N

n(a)=input(['' ]);

a=a+1;

end

for i=1:length(n)

if n(i)==1

n(i)=8;

else n(i)=-8;

end

end

y=0; i=1; a=0; b=0.5;

t=0:0.001:length(n);

for j=1:length(t)

if t(j)>=a && t(j)<=b

y(j) = n(i);

else if t(j)>=b && t(j)<=i

y(j)=0;

else i=i+1;

a=a+1;

b=b+1;

end

end

end

plot(t,y);

axis([0 length(n) -10 10]);

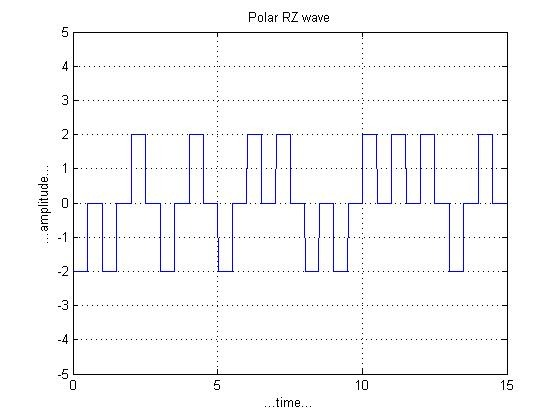
title('polar RZ wave');

xlabel('...time...');

ylabel('..amplitude..');

grid on;

**Output**



Write a program to generate Polar : Manchester

**Introduction:** In [telecommunication](https://en.wikipedia.org/wiki/Telecommunication) and [data storage](https://en.wikipedia.org/wiki/Computer_data_storage), **Manchester code** (also known as **phase encoding**, or **PE**) is a [line code](https://en.wikipedia.org/wiki/Line_code) in which the encoding of each data [bit](https://en.wikipedia.org/wiki/Bit) is either low then high, or high then low, for equal time. It is a [self-clocking signal](https://en.wikipedia.org/wiki/Self-clocking_signal) with no [DC component](https://en.wikipedia.org/wiki/DC_component). As a result, electrical connections using a Manchester code are easily [galvanically isolated](https://en.wikipedia.org/wiki/Galvanic_isolation" \o "Galvanic isolation).

N=4;

n=randi([0 1],1,N);

%Binary to bipolar conversion

nnn=[];

for m=1:N

if n(m)==0

nn=[-1 1];

else

nn=[1 -1];

end

nnn=[nnn nn];

end

%NRZ Pulse Shaping

i=1; l=0.5;

t=0:.001:length(n);

for j=1:length(t)

if t(j)<=1

y(j)=nnn(i);

else

y(j)=nnn(i);

i=i+1;

l=l+0.5;

end

end

plot(t,y);

axis([0 N -2 2]);

grid on;

**Output**

