DIGITAL COMMUNICATIONS LAB

DIGITAL PROJECT

Rewan khaled Abdulkariem	**19015677**
Rana Medhat Hussien Osman	**19015670**
Jailan Nashaat Abd Elfatah	**19015559**

Part 1:

```
% Simulation parameters
clear all;  clc;
NumBits = 1e5;     % Number of bits to transmit
SNRrange = 0:2:30; % Range of SNR values to test
NumSamples = 20;   % Number of samples to represent each bit waveform

prompt = sprintf('Enter the number of samples : ', NumSamples);
%(default is %d)
user = input(prompt);

if ~isempty(user)
    NewNumofSamples = user;
else
    NewNumofSamples = NumSamples;
end
SamplingInstant = NewNumofSamples; % Sampling instant for matched filter
filter = ones(NewNumofSamples,1);

% Generate random binary data vector
Bits = randi([0 1], NumBits, 1);

% Initialization
MfBer = zeros(1,length(SNRrange));
corrBer = zeros(1,length(SNRrange));
detectorBer =  zeros(1,length(SNRrange));

for n = 1:length(SNRrange)
    % Represent each bit with proper waveform
    dataTransmitted = reshape(repmat(Bits,1,NewNumofSamples)',NumBits*NewNumofSamples,1);

    % Add noise
    SNR = SNRrange(n);
    dataRecieved = awgn(dataTransmitted,SNR,'measured');

    %% Matched filter
    MF_Out = conv(dataRecieved,filter);
    samples = MF_Out(SamplingInstant:NewNumofSamples:end);

    % Perform thresholding
    threshold = mean(dataRecieved);

    MFdetectedBit = samples >= threshold;
```

```
    % Calculate BER
    bitErrors = xor(MFdetectedBit,Bits);
    MfBer(n) = sum(bitErrors)/NumBits;

    %% correlator
    % reshape the received signal into NumBits rows and 20 columns
    recieved_corr = reshape(dataRecieved,NewNumofSamples,NumBits)';
    corrData = recieved_corr * filter;
    CorrdetectedBit = corrData >= threshold;
    CorrDifference  = xor(CorrdetectedBit,Bits);    %xor between Received signal after comparator with bits
    corrBer(n) =  sum(CorrDifference)/NumBits;
    %% simple detector
     samples = dataRecieved(SamplingInstant:NewNumofSamples:end);
     detectedBit = samples >= threshold;
     bitErrors = xor(detectedBit,Bits);
     detectorBer(n) = sum(bitErrors)/NumBits;
end
figure
% Plot BER vs SNR
tansmisttedPower = (1/NumBits) * sum(Bits.^2);
display(tansmisttedPower);
semilogy(SNRrange,MfBer,'linewidth', 2,'color','b','marker','o');
hold on;
semilogy(SNRrange,corrBer,'linewidth', 2,'color','r');
semilogy(SNRrange,detectorBer,'linewidth', 2,'color','k');
xlim([0 30])
xlabel('SNR (dB)');
ylabel('Bit Error Rate');
title('BER vs SNR Correlator and Matched Filter');
legend('Matched filter','Correlator')
grid on;

hold off;
```

3-

transmitted signal power=0.4988 Watt.

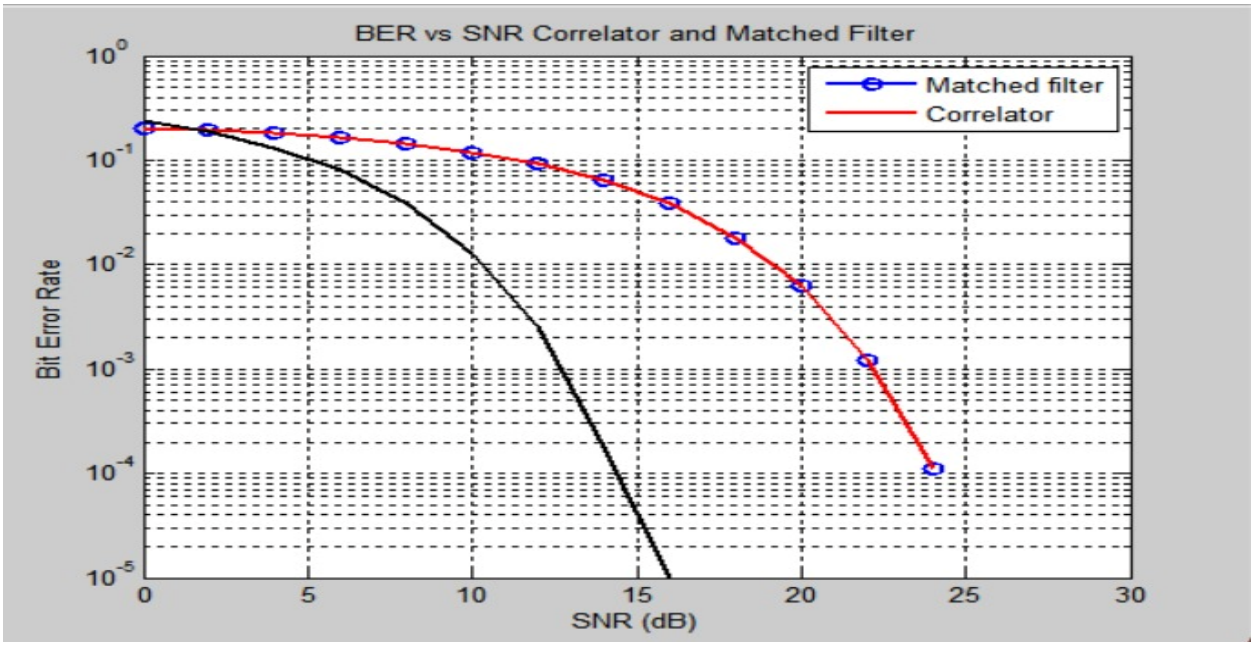
```
Enter the number of samples : 20

tansmisttedPower =

    0.4988
```

4-

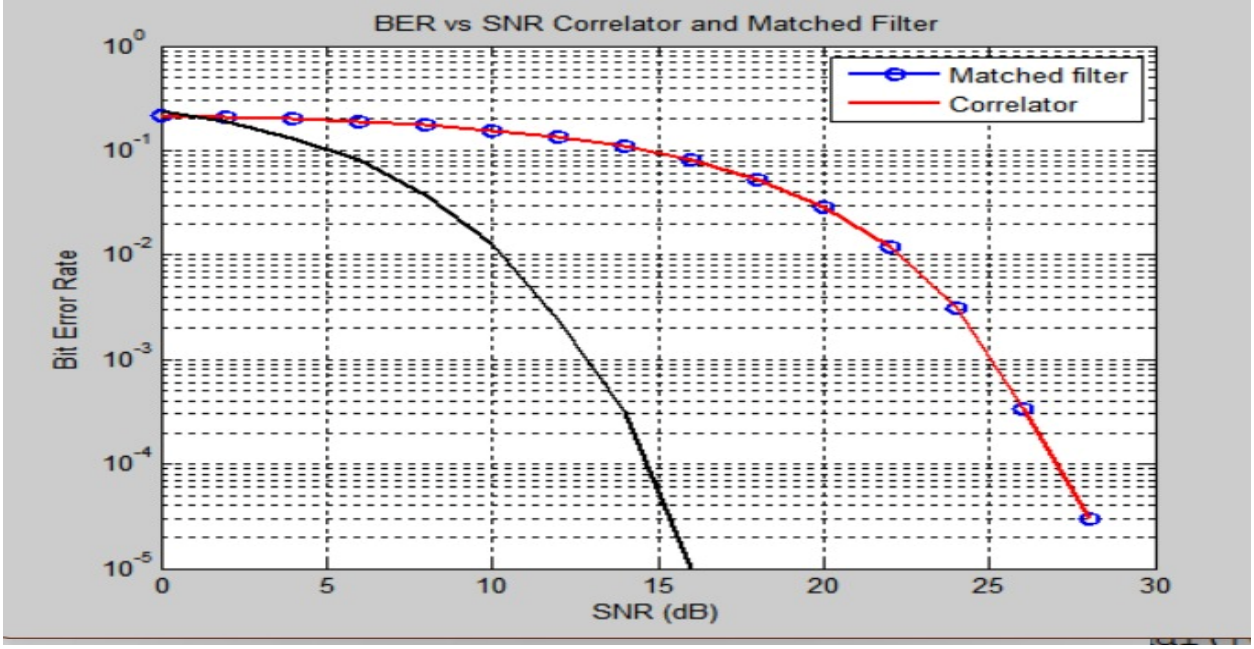
the value of SNR at which the system is nearly without error is 28 db

Graphs:

N= 10



N=20



N=100

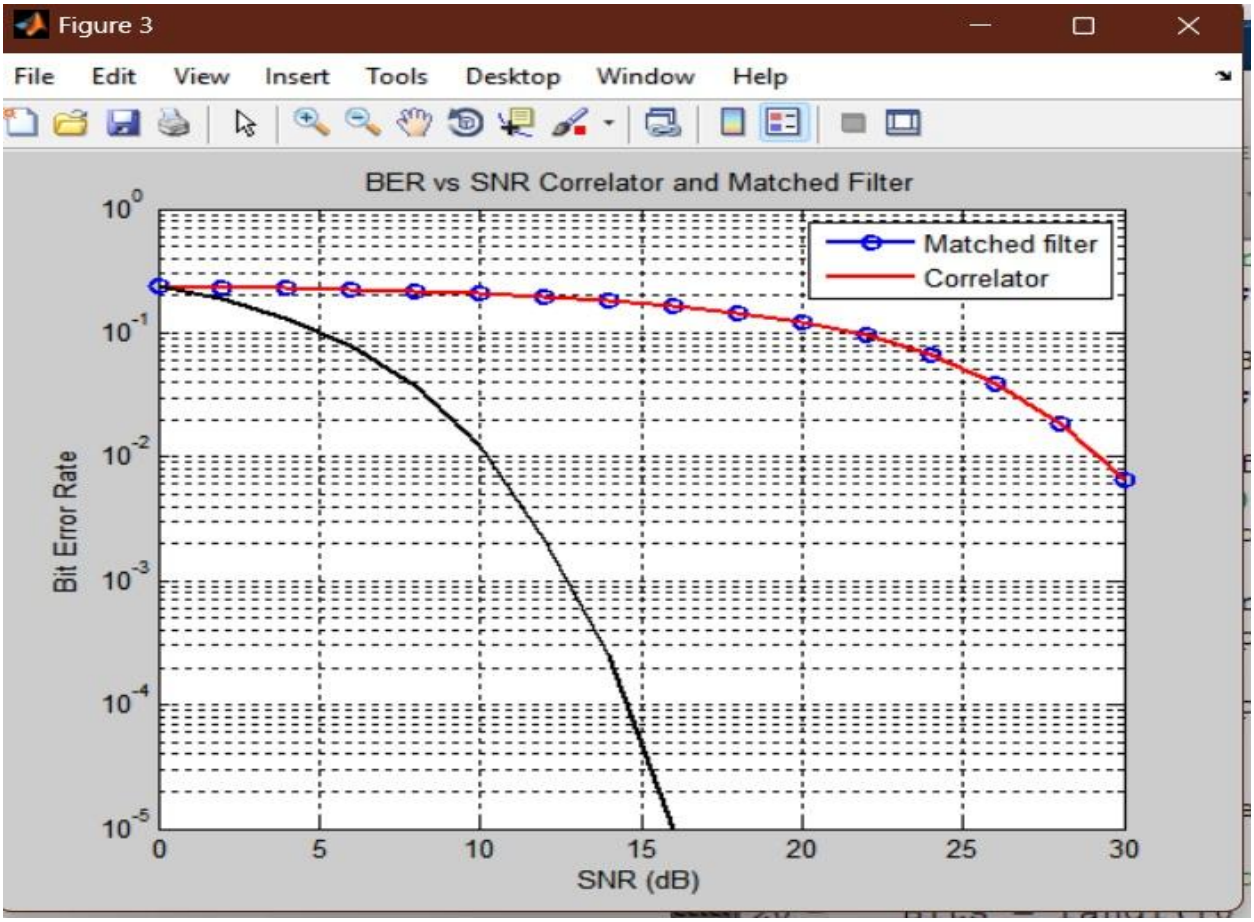


Figure 3
File Edit View Insert Tools Desktop Window Help
BER vs SNR Correlator and Matched Filter
Matched filter
Correlator
Bit Error Rate
SNR (dB)
10^0
10^-1
10^-2
10^-3
10^-4
10^-5
0
5
10
15
20
25
30

Part 2

Contents

- Generate some bits randomly
- Modulate using NRZ-L line code
- Modulate using NRZ-I line code
- Modulate using RZ line code
- Modulate using Alternative mark inversion (AMI)
- Modulate using Manchester line code
- Modulate using Multi-level transmission 3


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

Generate some bits randomly

```
NumberOfBits = 10 ;
RandomBits = randi([0 1],1,NumberOfBits);
%RandomBits = [1 0 1 0 0 1 1 0 1 1]; %  test case
Tb = 1;
f = 0: 0.01: 5;
Ts = 1;

% plot orignal squence
figure(1)
stairs(0:NumberOfBits,[RandomBits 0],'linewidth', 2);
title('Orignal Squence');
ylim([-.2 1.2]);

% Intialize arrays for output waves
NRZ_out = zeros(1,10);
NRZI_out = zeros(1,10);
RZ_out = zeros(1,20);
AMI_out = zeros(1,10);
Manchester_out = zeros(1,10);
MLT3_out = zeros(1,10);

% Modulate the signal using the specified line code
```

Modulate using NRZ-L line code

```
% expected squeunce = [1 -1 1 -1 -1 1 1 -1 1 1]
signal = ones(1,NumberOfBits+1);
signal(RandomBits==0) = -1;
NRZ_out = signal;

figure(2);
subplot(3,2,1)
stairs(0:NumberOfBits ,NRZ_out ,'linewidth', 2);
title('NRZ-L');
ylim([-2 2]);

% Calculate the power spectrum density
NRZ_psd = Tb*(sinc(f*Tb)).^2;

figure(3);
subplot(3,2,1);
plot(NRZ_psd,'linewidth', 2);
title('Non-return to zero PSD');
```

Modulate using NRZ-I line code

```
% expected squeunce = [1  1 -1 -1 -1 1 -1 -1 1 -1]
%OneFlag is a flag used to indicate the last "One" state (positive/negative)
OneFlag = 1; %Initial value from +vp
signal = zeros(1,NumberOfBits+1);
signal(1) = OneFlag;
for index=2:length(RandomBits)
  if RandomBits(index)==1
     OneFlag = -1* OneFlag; %Invert the "One" state
     signal(index) = OneFlag;
 elseif RandomBits(index)== 0
    signal(index) = OneFlag ;
 end
end
NRZI_out = signal;
```

```
figure(2);
subplot(3,2,2)
stairs(0:NumberOfBits, NRZI_out ,'linewidth', 2);
title('NRZ-Inverted');
ylim([-2 2]);

% Calculate the power spectrum density
NRZI_psd = Tb*(sinc(f*Tb)).^2;

figure(3);
subplot(3,2,2);
plot(NRZI_psd ,'linewidth', 2);
title('Non-return to zero inverted');
```

Modulate using RZ line code

```
% expected squeunce = [1 -1 1 -1 -1 1 1 -1 1 1] each 50% of cycle
signal = zeros(1,2*NumberOfBits+1);

for i = 1 :2: 2*NumberOfBits
    if RandomBits((i+1)/2) == 1
        signal(i) = 1;
        signal(i+1) = 0;

    else
        signal(i) = -1;
        signal(i+1) = 0;
    end
end
RZ_out = signal;
figure(2);
subplot(3,2,3)
stairs(0:0.5:NumberOfBits, RZ_out ,'linewidth', 2);
title('RZ');
ylim([-2 2]);

% Calculate the power spectrum density
RZ_psd=Tb/4*(sinc(f*Tb/2)).^2;
```

```
figure(3);
subplot(3,2,3);
plot(RZ_psd,'linewidth', 2);
title('Return to zero');
```

Modulate using Alternative mark inversion (AMI)

expected squeunce = [1 0 -1 0 0 1 -1 0 1 -1]

```
OneFlag = 1; % Initial value from +vp
signal = ones(1,NumberOfBits+1);

for index=1:length(RandomBits)
  if RandomBits(index)== 1
     signal(index) = OneFlag;
     OneFlag = -1*OneFlag;          % Invert the "One" state
 elseif RandomBits(index)== 0
    signal(index) = 0 ;
 end
end
AMI_out = signal;
figure(2);
subplot(3,2,4)
stairs(0:NumberOfBits, AMI_out ,'linewidth', 2);
title('AMI');
ylim([-2 2]);

% Calculate the power spectrum density
AMI_psd=Tb/4*(sinc(pi*f*Tb/2)).^2.*(sin(pi*f*Tb)).^2;

figure(3);
subplot(3,2,4);
plot(AMI_psd,'linewidth', 2);
title('Alternate mark inversion');
```

Modulate using Manchester line code

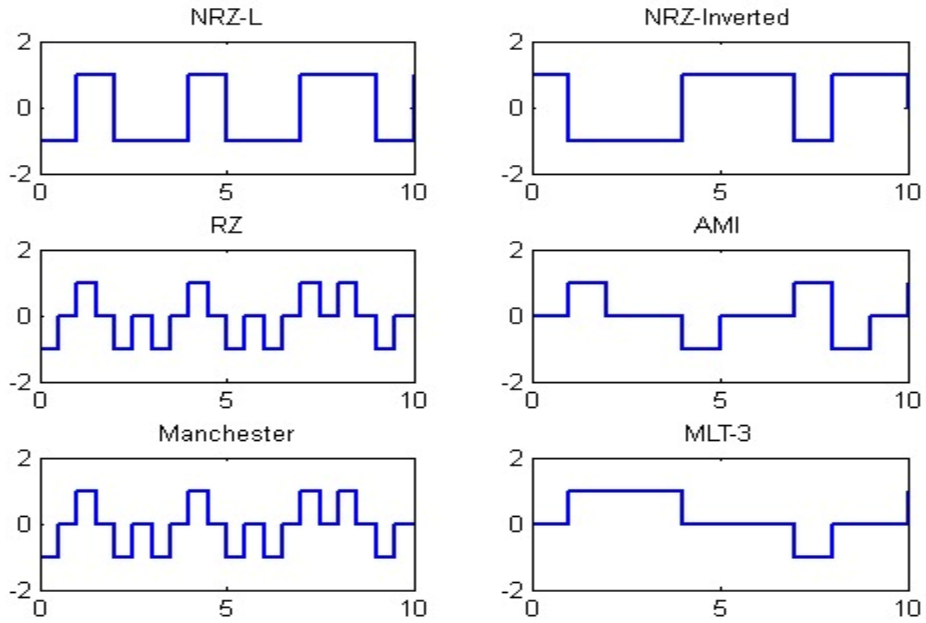
RandomBits = [1 0 1 0 0 1 1 0 1 1]; expected squeunce = [(1 -1) (-1 1) (1 -1) (-1 1) (-1 1)(1 -1)(1 -1)(-1 1)(1 -1)(1 -1)]

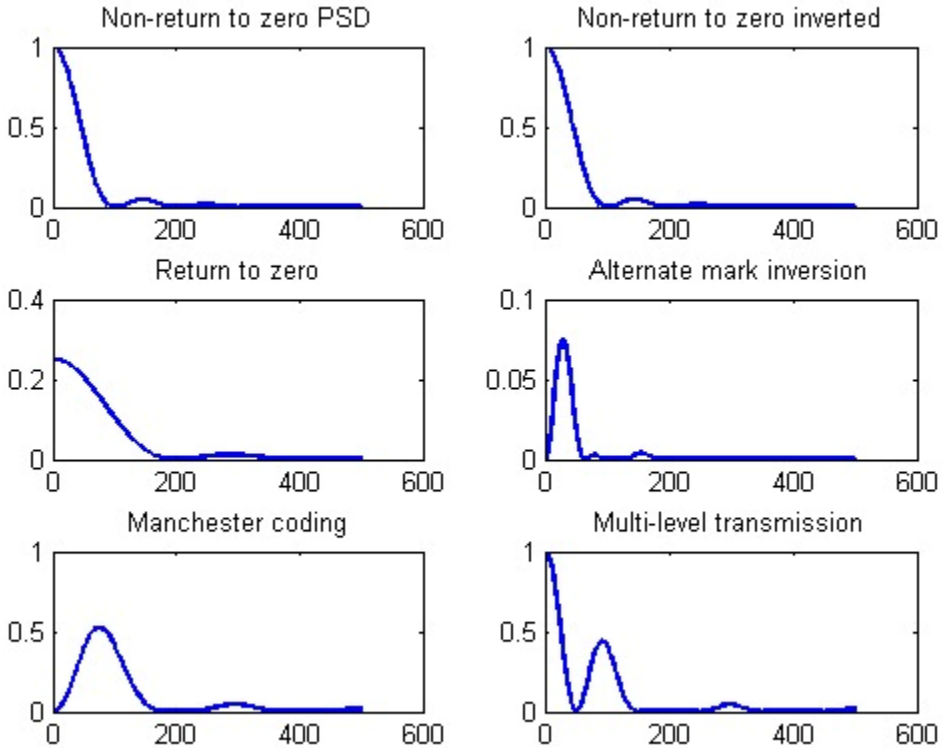
```
signal = zeros(1,2*NumberOfBits+1);

for i = 1 :2: 2*NumberOfBits
    if RandomBits((i+1)/2) == 1
        signal(i) = 1;
        signal(i+1) = -1;

    else
        signal(i) = -1;
        signal(i+1) = 1;
    end
end
Manchester_out = signal;
figure(2);

subplot(3,2,5)
stairs(0:0.5:NumberOfBits, RZ_out ,'linewidth', 2);
title('Manchester');

ylim([-2 2]);


% Calculate the power spectrum density
Manchester_psd=Tb*(sinc(f*Tb/2)).^2.*(sin(pi*f*Tb/2)).^2;

figure(3);
subplot(3,2,5);
plot(Manchester_psd,'linewidth', 2);
title('Manchester coding');
```

Modulate using Multi-level transmission 3

```
%RandomBits =           [1 0 1 0 0  1 1 0 1 1];
% expected squeunce =   [1 1 0 0 0 -1 0 0 1 0]
signal = ones(1,NumberOfBits+1);
Level = [1 0 -1 0];
i = 1;
for index=1:length(RandomBits)
 if RandomBits(index)==1
     signal(index)= Level(i);
      if (i < 4)
        i = i+1;
      else
        i = 1;
      end
 elseif RandomBits(index)==0
     if index == 1
         signal(index)= 0;
     else
        signal(index)= signal(index - 1);
     end

 end
end
MLT3_out = signal;
figure(2);

subplot(3,2,6)
stairs(0:NumberOfBits, MLT3_out ,'linewidth', 2);
title('MLT-3');
ylim([-2 2]);

% Calculate the power spectrum density
MLT_psd=Tb*(sinc(f*Tb/2)).^2.*(cos(pi*f*Tb)).^2 ;

figure(3);
subplot(3,2,6);
handle2 = plot(MLT_psd );
set(handle2,'LineWidth',2)
title('Multi-level transmission');
```

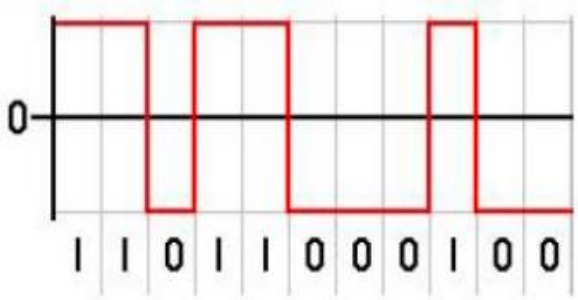
Graphs:


NRZ-L
NRZ-Inverted
RZ
AMI
Manchester
MLT-3


Non-return to zero PSD
Non-return to zero inverted
Return to zero
Alternate mark inversion
Manchester coding
Multi-level transmission

2- Manchester line code has the highest bandwidth. This is because Manchester encoding consumes up to twice the bandwidth of the original signal.

Polar NRZ:



Advantages:

1-Simple.

2-There is no low-frequency component present.

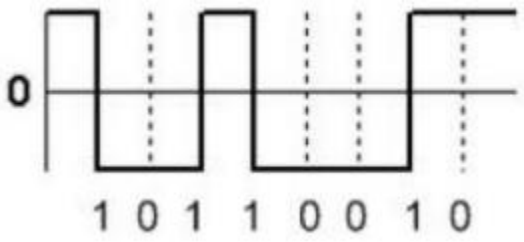
3- Requires relatively low bandwidth

Disadvantages:

1-No error correction

2-Long sequences may cause loss of synchronization due to the absence of transitions.

Polar Non Return to Zero Inverted(NRZI):



Advantages:

1- NRZI ensures that after a 0 bit appears, the voltage will immediately switch to a 1 bit voltage level.

2- These voltage changes allow the sending and receiving clocks to synchronize.

3-Non-Return-to-Zero Inverted A method for transmitting and recording data so that it keeps the sending and receiving clocks synchronized. This is especially helpful in situations where bit stuffing is employed — the practice of adding bits to a data stream so it conforms with communications protocols.

Disadvantages:

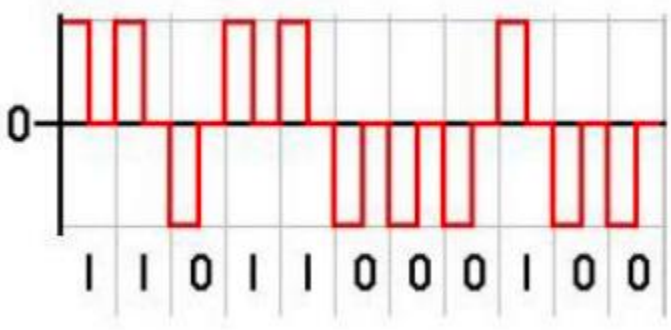
1- Long strings of values does not have a level transition resulting in difficulty to make sure that the clock stays in synchronization.

2-For this reason, many NRZI encoding systems make use of bit stuffing or run-length limited coding to limit the maximum number of transition-less bit times.

3-The lack of regular signal transitions makes clock recovery from the signal transitions difficult.

4- Long sequence with infrequent changes in voltage causes the DC value to drift. Since that average voltage is used to discriminate between +V and -V, error may be introduced.

Polar RZ:



Advantages:

1-Relatively simple to implement.

2- No low frequency component are present.

3-Uses less power than NRZ.

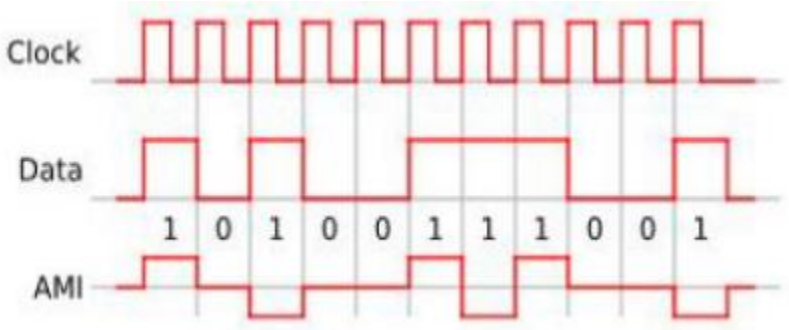
4- Maintains synchronization due to transitions.

Disadvantages:

1-Requires twice the bandwidth of NRZ.

2-No error correction.

AMI (Alternate Mark Inversion):



Advantages:

1-Simple

2- No low frequency component are present.

3- Occupies low bandwidth than unipolar and polar NRZ schemes.

4-It is suitable for transmission over AC coupled lines as signal drooping does not occur here.

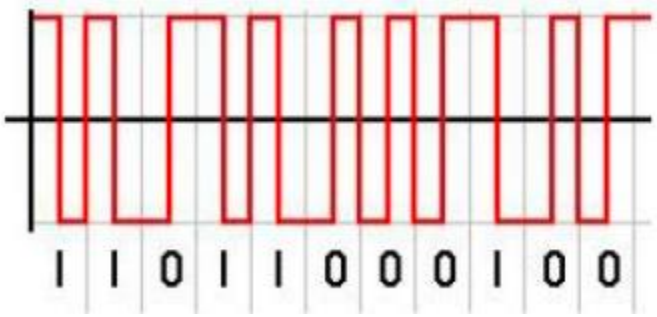
5- A single error detection capability is present.

Disadvantages:

1- No clock is present.

2- Long strings of data cause loss of synchronization**.**

Manchester code:



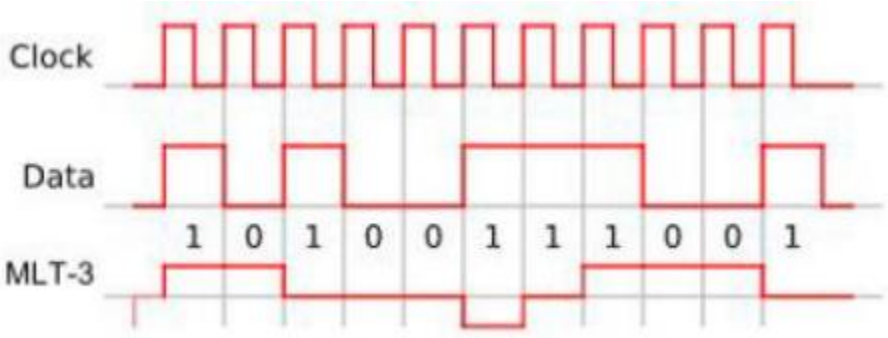
Advantages:

- The DC component of the signal carries no information. This makes it possible that standards that usually do not carry power can transmit this information.

Disadvantages:

- It needs more bandwidth than other encodings such as NRZ.

MLT-3 encoding:



Advantages:

1-It has signal rate which is (1/4)th of the bit rate.

2-Due to its signal shape, it reduces required bandwidth.

Disadvantages:

1-It does not support self-synchronization for long string of zeros ('0').

2-It is more complex than NRZ-I due to use of three levels and complex transition rules.

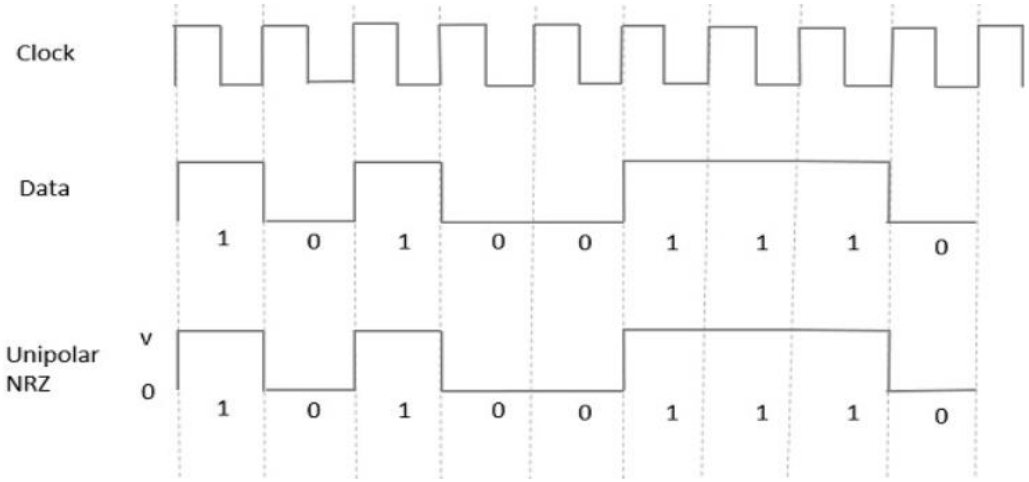
5-

1-Unipolar Non-Return to Zero NRZ:

In this type of unipolar signaling, a High in data is represented by a positive pulse called as Mark, which has a duration T0 equal to the symbol bit duration. A Low in data input has no pulse.

The following figure clearly depicts this.

The



advantages of Unipolar NRZ are :

1-It is simple.
2-A lesser bandwidth is required.

The disadvantages of Unipolar NRZ are:

1-No error correction done.

2-Presence of low frequency components may cause the signal droop.

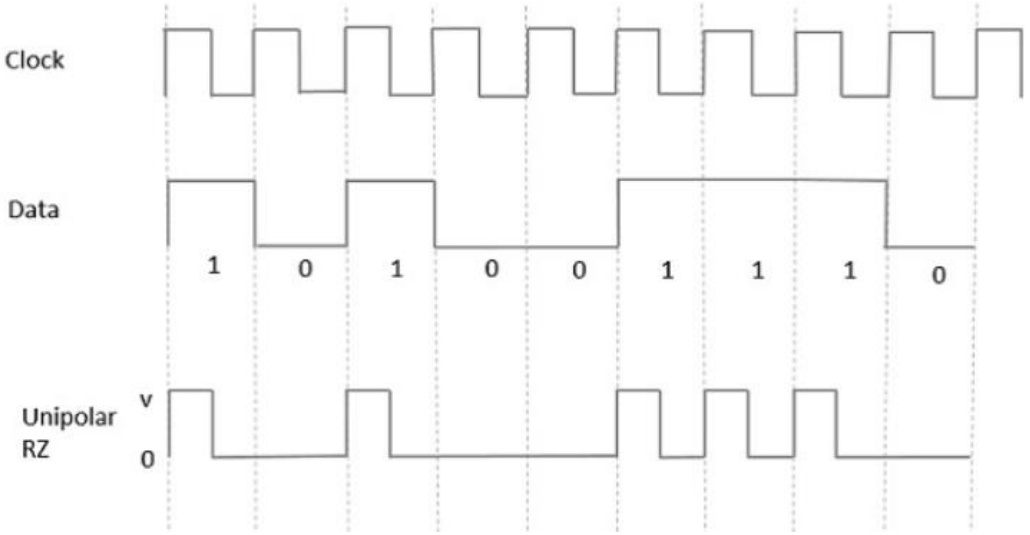
3-No clock is present.

3-Loss of synchronization is likely to occur (especially for long strings of 1s and 0s).

2-Unipolar Return to Zero RZ:

In this type of unipolar signaling, a High in data, though represented by a Mark pulse, its duration T0 is less than the symbol bit duration. Half of the bit duration remains high but it immediately returns to zero and shows the absence of pulse during the remaining half of the bit duration.

It is clearly understood with the help of the following figure.



The advantages of Unipolar RZ are:

1-It is simple.
2-The spectral line present at the symbol rate can be used as a clock.

The disadvantages of Unipolar RZ are:

1-No error correction.
2-Occupies twice the bandwidth as unipolar NRZ.
3-The signal droop is caused at the places where signal is non-zero at 0 Hz.