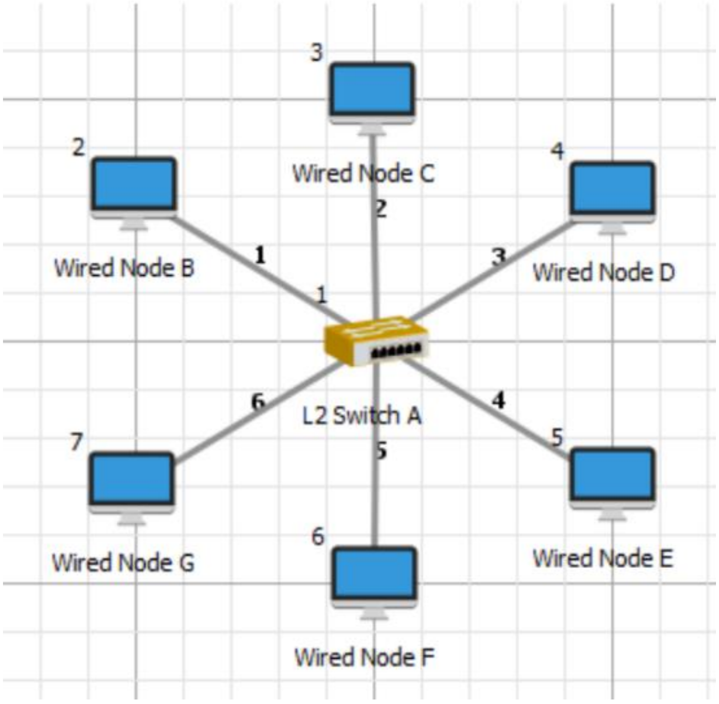
# INDEX

**LAYOUT:**

**STAR:**


3
Wired Node C
2
2
Wired Node B
1
1
4
Wired Node D
3
L2 Switch A
6
7
Wired Node G
5
4
5
Wired Node E
6
Wired Node F

| EX:NO:1<br><br>09.09.2021 | STAR, BUS AND RING TOPOLOGIES |
|---|---|

## AIM:

To study star, bus and ring topologies and analyse their throughputs and delays.

## SOFTWARE REQUIRED:

NETSIM

## A. STAR:

## THEORY:

In local area networks with a star topology, each network host is connected to a central switch with a point-to-point connection. In Star topology every node (computer workstation or any other peripheral) is connected to central node called switch. The switch is the server and the peripherals are the clients. The network does not necessarily have to resemble a star to be classified as a star network, but all of the nodes on the network must be connected to one central device. All traffic that traverses the network passes through the central switch.The star topology is considered the easiest topology to design and implement. An advantage of the star topology is the simplicity of adding additional nodes.

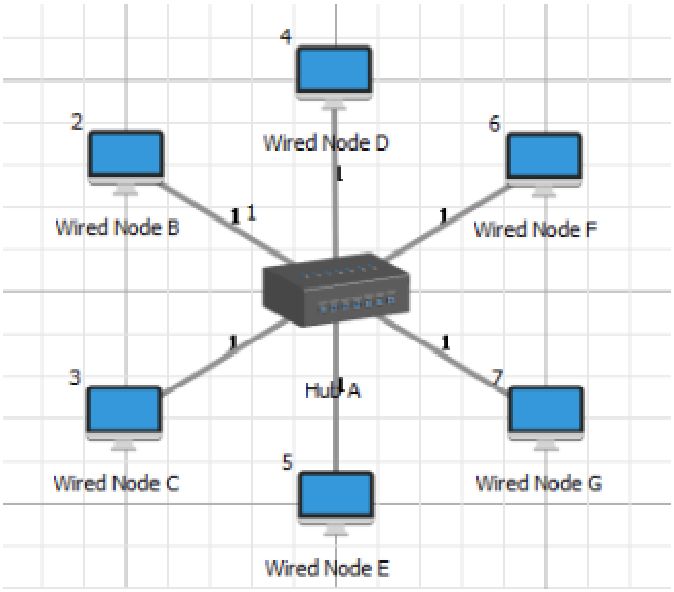
## PROCEDURE:

1. Select New → Internetworks.
2. Click and drop required number of nodes and a switch.
3. Disable TCP in properties of all the wired nodes
4. **Wired Link Properties:**

| **Wired Link Properties** | **All Links** |
|---|---|
| **Uplink Speed (Mbps)** | 100 |
| **Downlink Speed(Mbps)** | 100 |
| **Uplink BER** | No Error |
| **Downlink BER** | No Error |

**LAYOUT:**

**BUS:**



4
Wired Node D
2
Wired Node B
6
Wired Node F
3
Wired Node C
Hub A
7
Wired Node G
5
Wired Node E

5. **Application Properties:**

| Application Properties | 2(Wired Node B) | 3(Wired Node C) | 4(Wired Node D) | 5(Wired Node E) | 6(Wired Node F) | 7(Wired Node F) |
|---|---|---|---|---|---|---|
| **Destination** | 3(Wired Node C) | 4(Wired Node D) | 5(Wired Node E) | 6(Wired Node F) | 7(Wired Node B) | 2(Wired Node B) |
| **Application Type** | Custom | Custom | Custom | Custom | Custom | Custom |
| **Packet Size** | | | | | | |
| **Distribution** | Constant | Constant | Constant | Constant | Constant | Constant |
| **Packet Size (Bytes)** | 10000 | 10000 | 10000 | 10000 | 10000 | 10000 |
| **Packet Inter Arrival Time** | | | | | | |
| **Distribution** | Constant | Constant | Constant | Constant | Constant | Constant |
| **Packet Inter Arrival Time (μs)** | 1000 | 1000 | 1000 | 1000 | 1000 | 1000 |

6. Run the simulation for 10s.

## B. TOKEN BUS:

## THEORY:

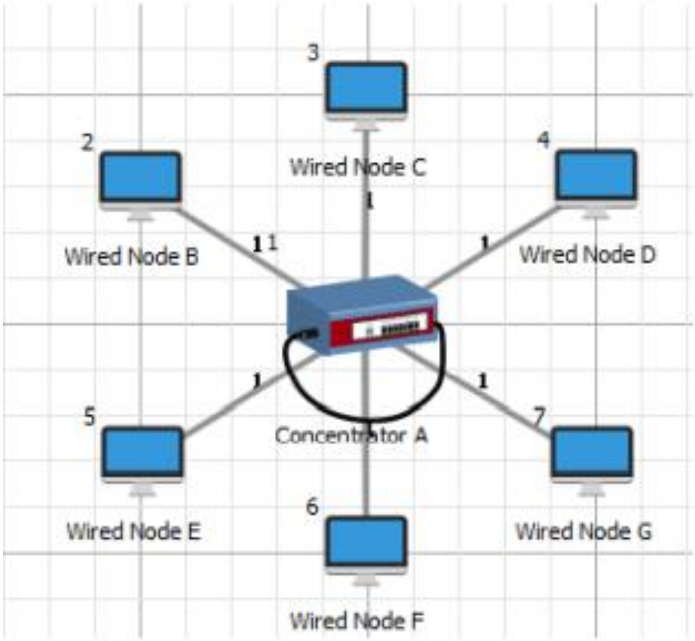
Token bus is a LAN protocol operating in the MAC layer. Token bus is standardized as per IEEE 802.4. Token bus can operate at speeds of 5Mbps, 10 Mbps and 20 Mbps. The operation of token bus is as follows. Unlike token ring in token bus the ring topology is virtually created and maintained by the protocol. A node can receive data even if it is not part of the virtual ring, a node joins the virtual ring only if it has data to transmit. In token bus data is transmitted to the destination node only where as other control frames is hop to hop. After each data transmission there is a solicit_successsor control frame transmitted which reduces the performance of the protocol.

## PROCEDURE:

1. Select New → Legacy Networks → Token Bus.
2. Click and drop required number of nodes and a hub.
3. Set the same Application properties as Star topology.
4. In Link properties, set Bit error rate as 0.
5. Run the simulation for 10s.

**LAYOUT:**

**RING:**


3
2
4
Wired Node C
Wired Node B
Wired Node D
Concentrator A
5
7
6
Wired Node E
Wired Node G
Wired Node F

## B. TOKEN RING:

## THEORY:

Token ring is a LAN protocol operating in the MAC layer. Token ring is standardized as per IEEE 802.5. Token ring can operate at speeds of 4mbps and 16 mbps. The operation of token ring is as follows. When there is no traffic on the network a simple 3-byte token circulates the ring. If the token is free (no reserved by a station of higher priority as explained later) then the station may seize the token and start sending the data frame. As the frame travels around the ring a station examines the destination address and is either forwarded (if the recipient is another node) or copied. After copying 4 bits of the last byte is changed. This packet then continues around the ring till it reaches the originating station. After the frame makes a round trip the sender receives the frame and releases a new token onto the ring.

## PROCEDURE:

1. Select New → Legacy Networks → Token Ring.
2. Click and drop required number of nodes and a concentrator.
3. Set the same Application properties as Star topology.

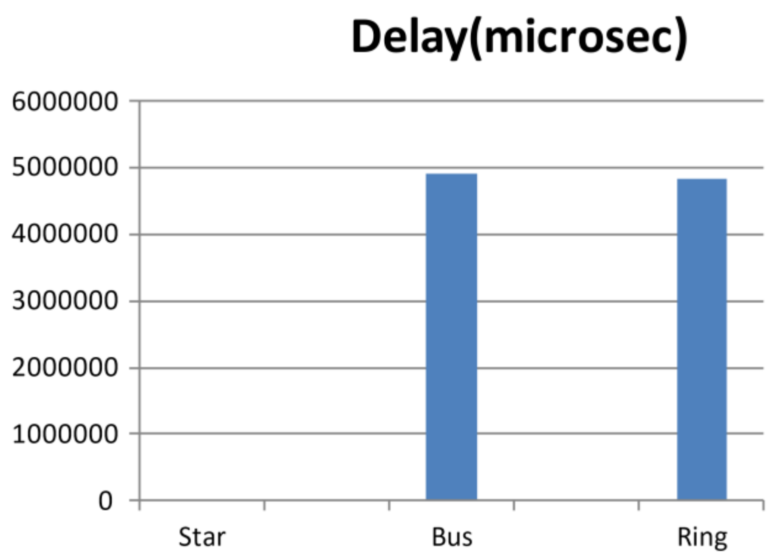
| Concentrator Properties | Values to be Selected |
|---|---|
| **Data Rate(Mbps)** | 16 |
| **Error Rate (bit error rate)** | No error |

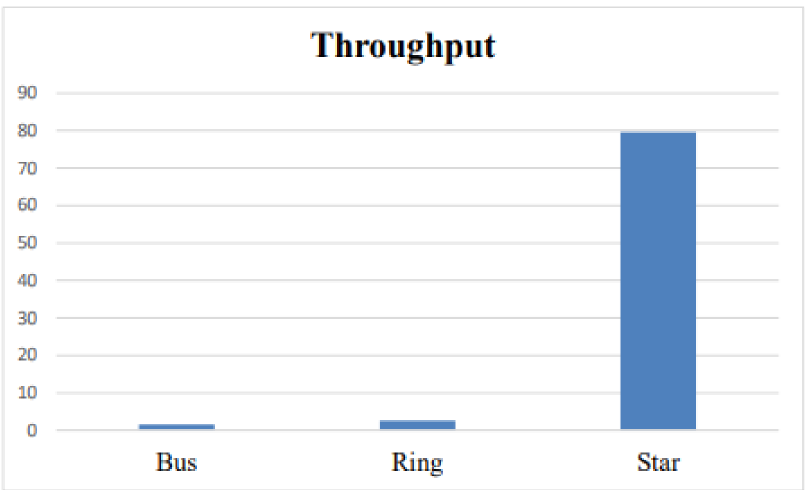
4. Run the simulation for 10s.

## TABULATION:

| NO.OF NODES | TOPOLOGY | THROUGHPUT(Mbps) | DELAY(μs) |
|---|---|---|---|
| 6 | STAR | 79.992 | 619.805714 |
| 6 | BUS | 1.611533 | 4900512.37 |
| 6 | RING | 2.578355 | 483878.057 |

## OUTPUT:



## INFERENCE:

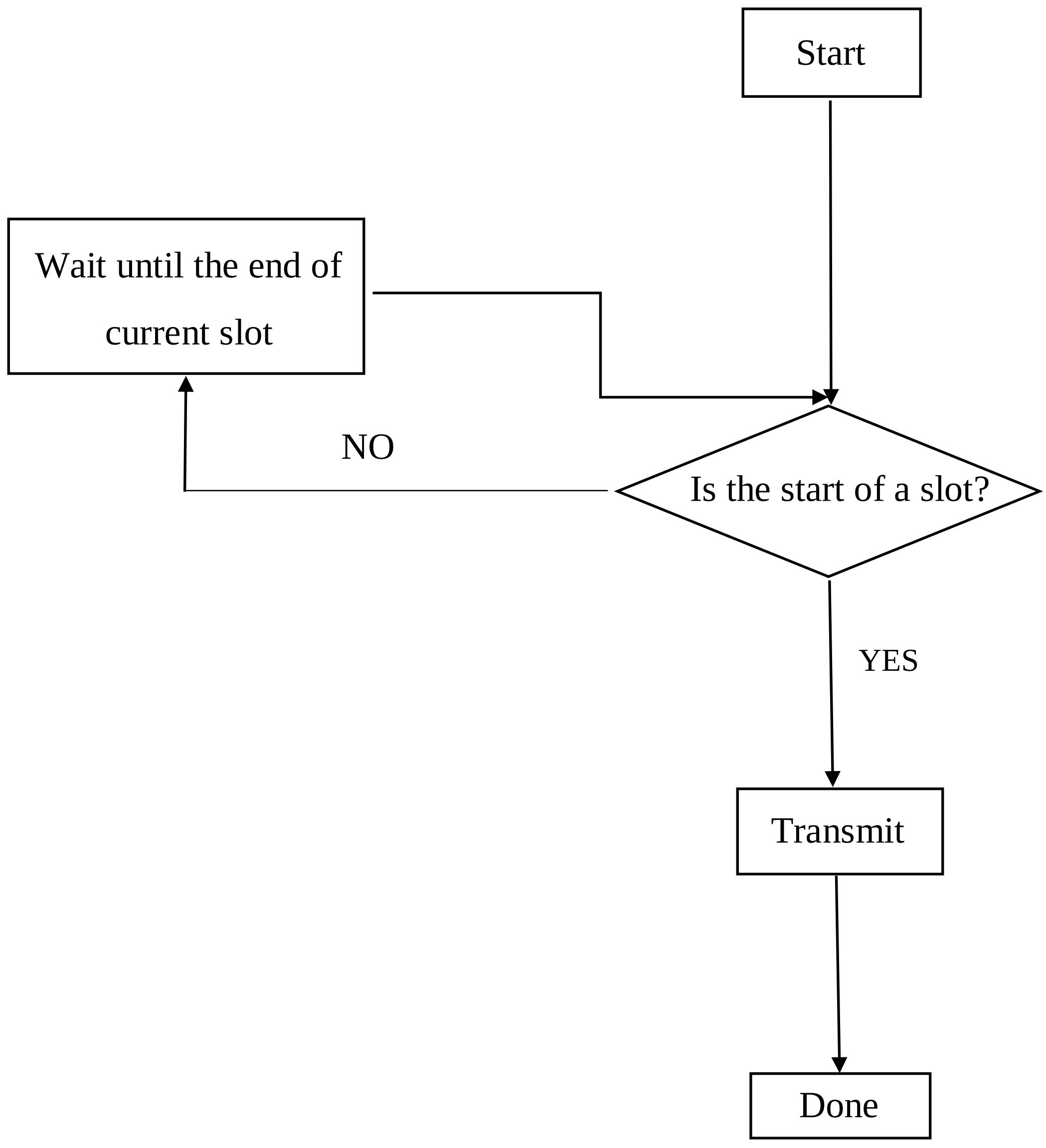
Due to more overheads and complicated ring maintenance procedure token bus always less than token rings. Star topology throughput is higher than other topology (Bus and Ring) because Star has higher data rate.

## RESULT:

Thus star, bus and ring topologies are studied and their throughputs and delays are analysed.

## SLOTTED ALOHA:

### FLOW CHART:



| EX:NO:2<br><br>13.09.2021 | MEDIUM ACCESS CONTROL (MAC) PROTOCOLS |
|---|---|

## AIM:

To study and analyse the following MAC protocols:

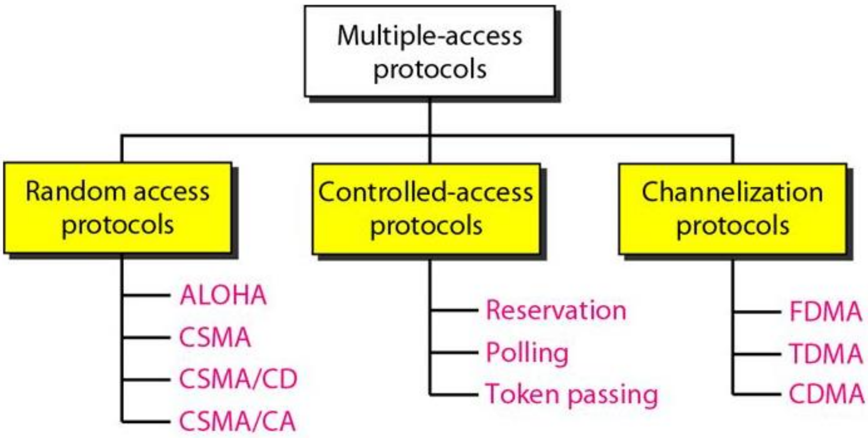
- Slotted ALOHA
- CSMA/CD
- CSMA/CA

## SOFTWARE REQUIRED:

NETSIM

## MAC PROTOCOLS:

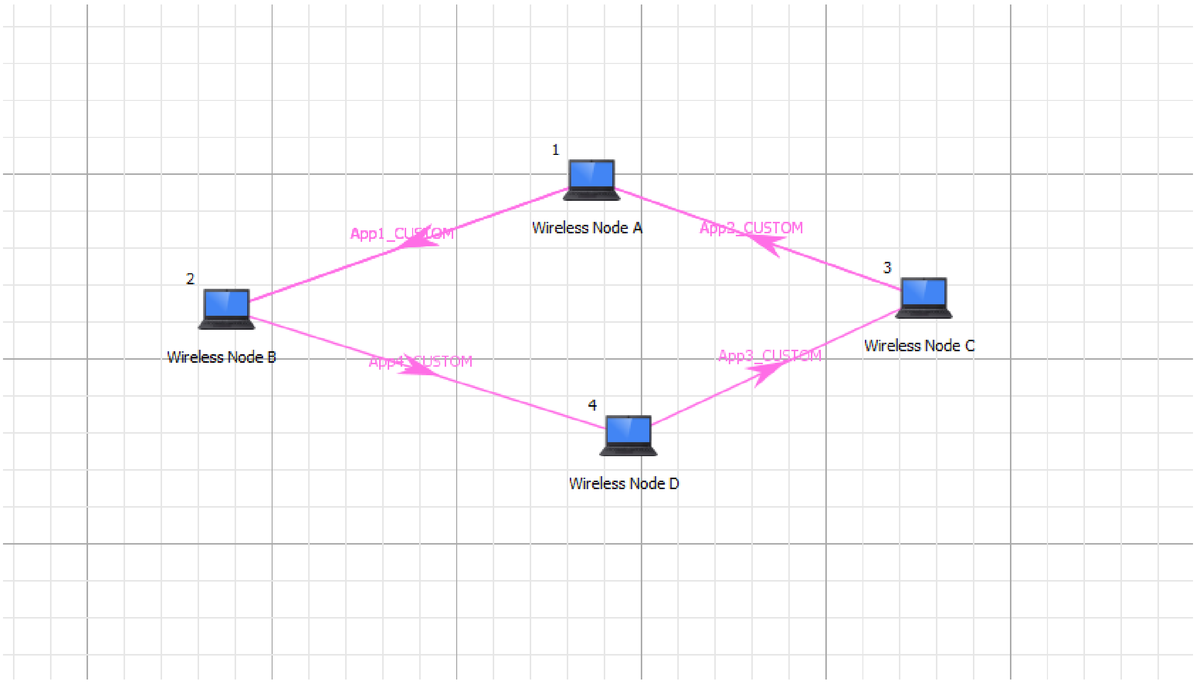
The medium access control (MAC) is a sublayer of the data link layer (DLL) in the seven-layer of the open system interconnections (OSI) reference model for data transmission. MAC is responsible for flow control and multiplexing for transmission medium. It controls the transmission of data packets via remotely shared channels. It sends data over the network interface card.




# RANDOM ACCESS PROTOCOL:

In random access or contention methods, no station is superior to another station and none is assigned the control over another. No station permits, or does not permit, another station to send. At each instance, a station that has data to send uses a procedure defined by the protocol make a decision on whether or not to send.

## LAYOUT:



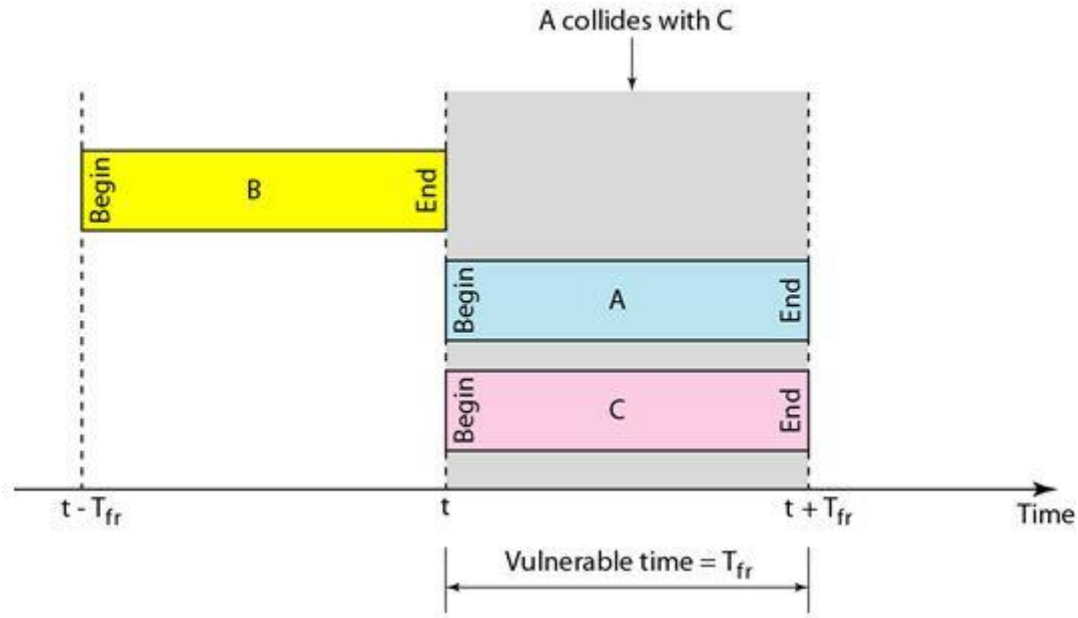

## TABULATION:

| NO OF NODES | THROUGHPUT (Mbps) | NO OF PACKETS TRANSMITTED | DELAY (µs) |
|---|---|---|---|
| 1 | 0.273008 | 809 | 8163.74 |
| 2 | 0.366735 | 919 | 9897.076 |
| 3 | 0.264314 | 703 | 9585.91 |
| 4 | 0.318776 | 772 | 11208.03 |
| 5 | 0.342304 | 773 | 10391.3 |
| 6 | 0.284067 | 719 | 9249.156 |
| 7 | 0.359071 | 746 | 12153.87 |

## SLOTTED ALOHA:

### THEORY:

This allows station to transmit data only at specific time slots. So, station cannot transmit whenever it wants. In this, the vulnerable time is reduced by half.




The throughput of the slotted ALOHA is $S = Ge^{-G}$ which is maximum when G = 1 (37%).

## PROCEDURE:

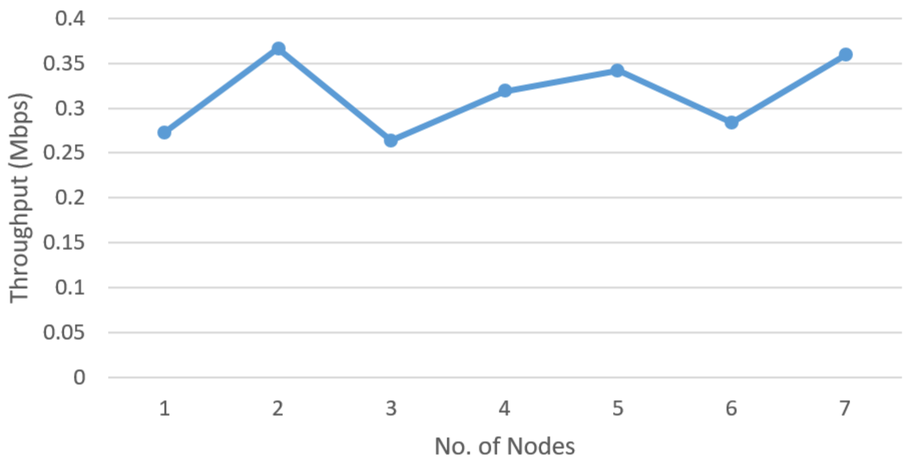
1. Select New → Legacy Networks → Slotted ALOHA.
2. Click and drop required number of nodes.
3. Right click on the grid environment and select the channel characteristics as no path loss.
4. Click and drop the appropriate icon and set the following properties as,

| | |
|---|---|
| Application method: | Unicast |
| Application type: | Custom |
| Source ID: | 1 |
| Destination ID: | 2 |
| Packet Size: | Exponential, 1472 bytes |
| Inter Arrival time: | Exponential, 20000 µs |

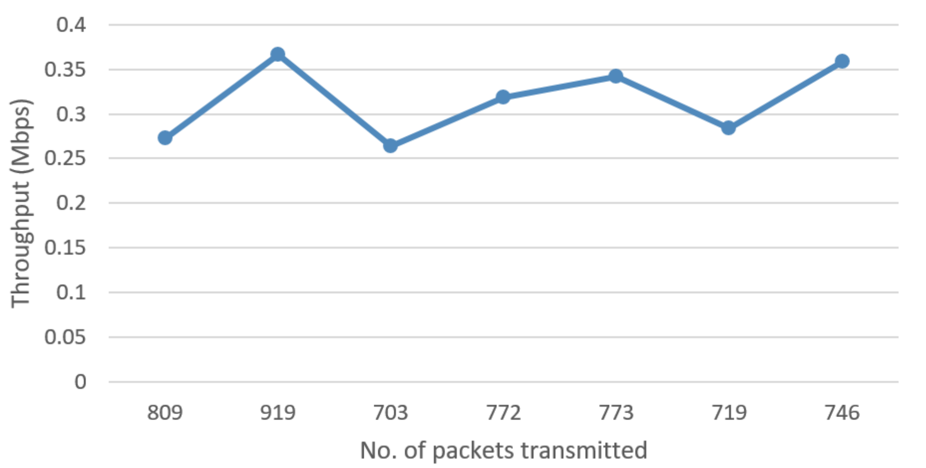
5. Click on 'Run Simulation' for 10 seconds.
6. Obtain the values of throughput and total number of packets transmitted.
7. Repeat the steps for higher number of traffic generating nodes.
8. Compare the efficiency, throughput and total number of packets transmitted.

## OUTPUT GRAPH: SLOTTED ALOHA:

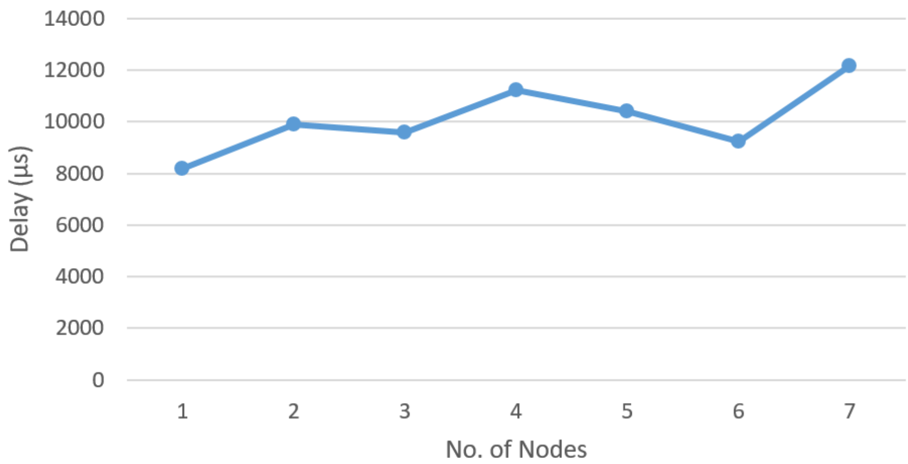
### No. Of Nodes Vs Throughput:




### No. Of Packets Tranmitted Vs Throughput:




### No. Of Nodes Vs Delay:



# CSMA/CD:

## THEORY:

Carrier Sense Multiple Access (CSMA) with collision detection (CD) is a Medium Access Control Method, used notably in Ethernet Technology. CSMA/CD is used to improve the performance of pure CSMA by terminating transition as soon as a collision is detected. Thus, shortening the time required before a retry can be attempted.

## COLLISION DETECTION:

When more than one signal is sent in the cable, the power level exceeds a certain limit which indicates that collision is occurred.

## MECHANISM:

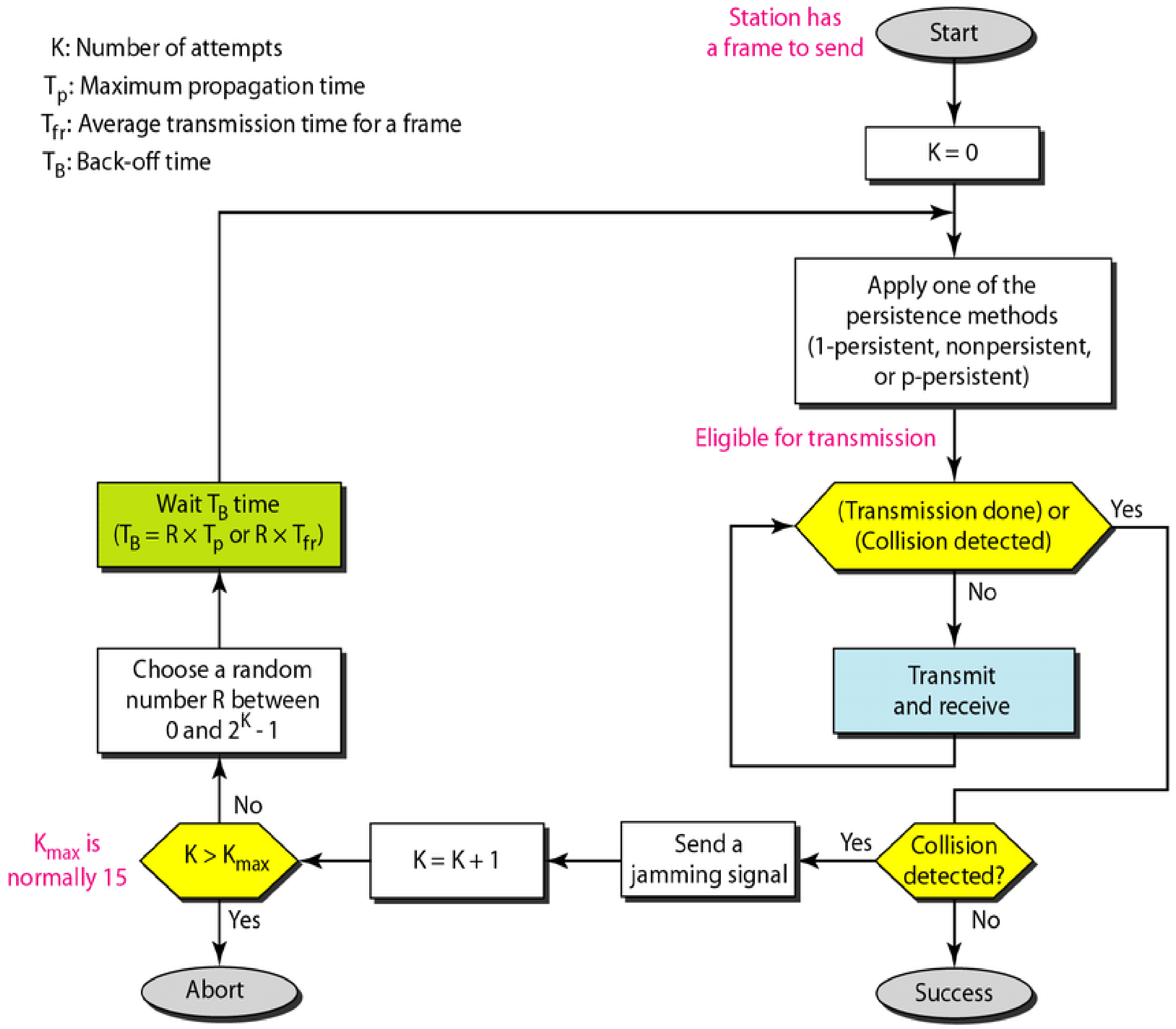
1. Check if the sender is ready for transmitting data packets
2. Sender has to keep on checking if the transmission link/medium is idle. If it senses that carrier is free and there are no collisions, it sends the data. Otherwise, it retains from sending data.
3. Transmit the data and check for collisions. If collisions is detected, the transmission is stopped and after some transfer data and repeats above process.
4. If no collision was detected, the counter is reset.

## PROCEDURE:

1. Select NEW → LEGACY NETWORKS → CSMA/CD.
2. Place required number of nodes and one hub are to be placed.
3. Vary persistence from ½,1/3, … . , in all wired nodes to generate other experiments.
4. Create broadcast application for all wired nodes and set the following properties.

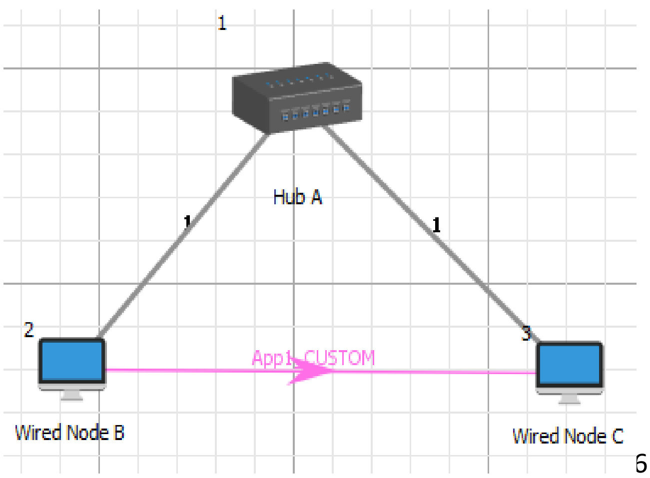
   APPLICATION METHOD: BROADCAST
   APPLICATION TYPE: CUSTOM
   PACKET SIZE: CONSTANT 1472
   INTER ARRIVAL TIME: EXPONENTIAL 1000
5. Run simulator for 10 Seconds.
6. Obtain the throughput for various persistent values.

# FLOW CHART: CSMA/CD:

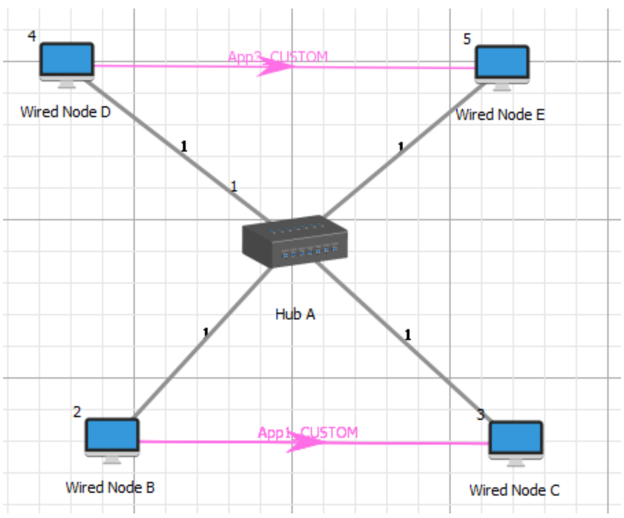



# LAYOUT: CSMA/CD:

## One node transmission:




## Two node transmission:



**TABULATION:**

- **One node transmission**

| BER | Packets Errored | Packets Generated |
|---|---|---|
| 0 | 0 | 3999 |
| 10E-09 | 0 | 3999 |
| 10E-08 | 1 | 3999 |
| 10E-07 | 6 | 3999 |
| 10E-06 | 46 | 3999 |

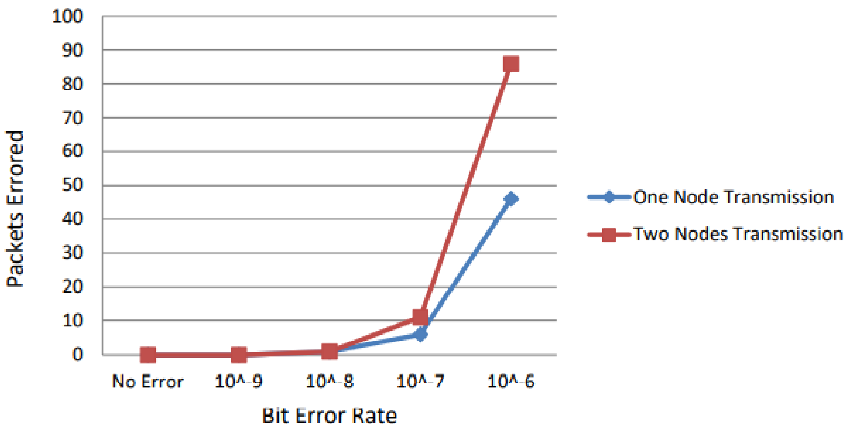
- **Two nodes transmission**

| BER | Packets Errored | Packets Generated |
|---|---|---|
| 0 | 0 | 7998 |
| 10E-09 | 0 | 7998 |
| 10E-08 | 1 | 7998 |
| 10E-07 | 11 | 7998 |
| 10E-06 | 74 | 7998 |

- **Throughput and Delay**

| **No. of nodes** | **Throughput in Mbps** | **Delay in µs** |
|---|---|---|
| 2 | 4.655 | 1220.8 |
| 4 | 2.985 | 8225515 |

## OUTPUT GRAPH: CSMA/CD:



**CSMA/CA:**

**THEORY:**

MANET is a self-configuring network of mobile nodes connected by wireless links to form an arbitrary topology without the use of existing infrastructure. The nodes are free to move randomly. Thus, the network wireless topology may be unpredictable and may change rapidly.

The node density also has an impact on the routing performance with very sparsely populated network. The number of possible connections between any two nodes are very less and hence the performance is poor. It is expected that if the node density is increased the throughput of the network shall increase but beyond a certain level if density is increased the performance degrades.

**PERFORMANCE METRICS:**

The different parameters used to analyze the performance are explained as follows:

THROUGHPUT:

It is the rate of successfully transmitting data packets is unit time in network during the simulation.

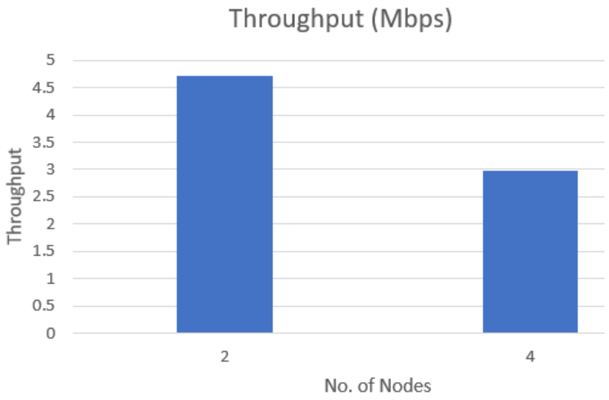
AVERAGE DELAY:

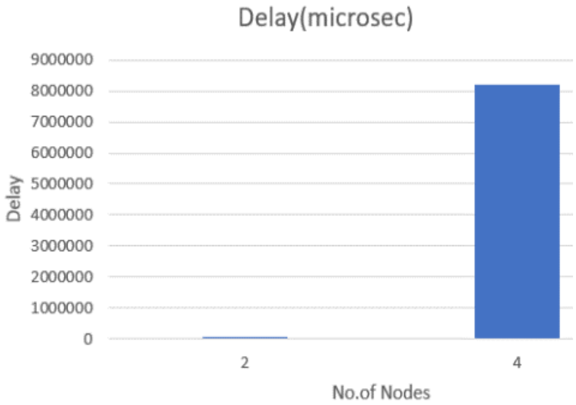
It is defined as the average time taken by the data packets to propagate from source to destination across a MANET. This includes all possible delays caused by routing discovery latency, queuing at the interface queue, and retransmission delays at the MAC, propagation and transfer times.

**PROCEDURE:**

1. In NETSIM, select → NEW ADVANCED WIRELESS → NETWORK MANET.
2. Create / Design the network with 2 wireless nodes.
3. Configure the network.
4. Click and drop 2 wireless nodes onto the simulation environment.
5. Position of node 1 as (50,100) and node 2 as (100, 150) (i.e) (x- coordinate, y-coordinate).
6. Disable TCP in all wireless nodes.
   Wireless link properties:
   PATH LOSS MODEL: LOG-DISTANCE
   PATH LOSS EXPONENT (n): 2
   MODEL TRAFFIC IN THE NETWORK.

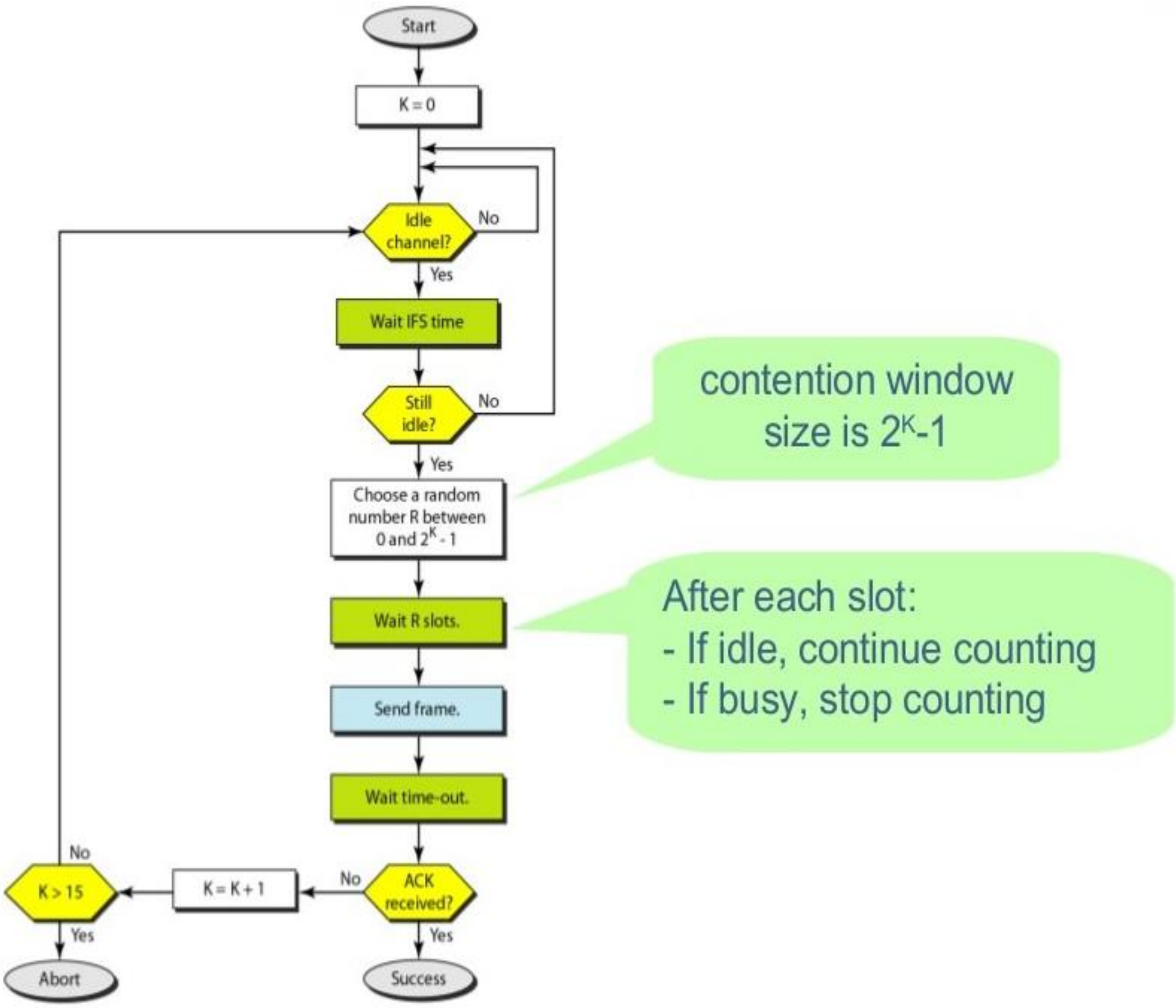
## OUTPUT GRAPH: CSMA/CD:




## CSMA/CA:

## FLOW CHART:



## APPLICATION PROPERTIES:

| APPLICATION PROPERTIES | |
|---|---|
| Application Type | Custom |
| Source ID | 1 |
| Destination ID | 2 |
| PACKET SIZE | |
| Distribution | Constant |
| Value (bytes) | 1460 |
| Inter Arrival Time | |
| Distribution | Constant |
| Value ((microseconds)) | 60,000 |

7. Set the simulation time for 10 seconds and run the experiment.
8. Repeat the same process by adding more numbers of nodes and compare the throughput and delay among different nodes.

## OUTPUT:

After simulation, Metrics window is obtained, we can analyze various parameters like

• Go to Application metrics and compute the sum of all column wise values of throughput.

Throughput = Sum of throughputs obtained from all Applications

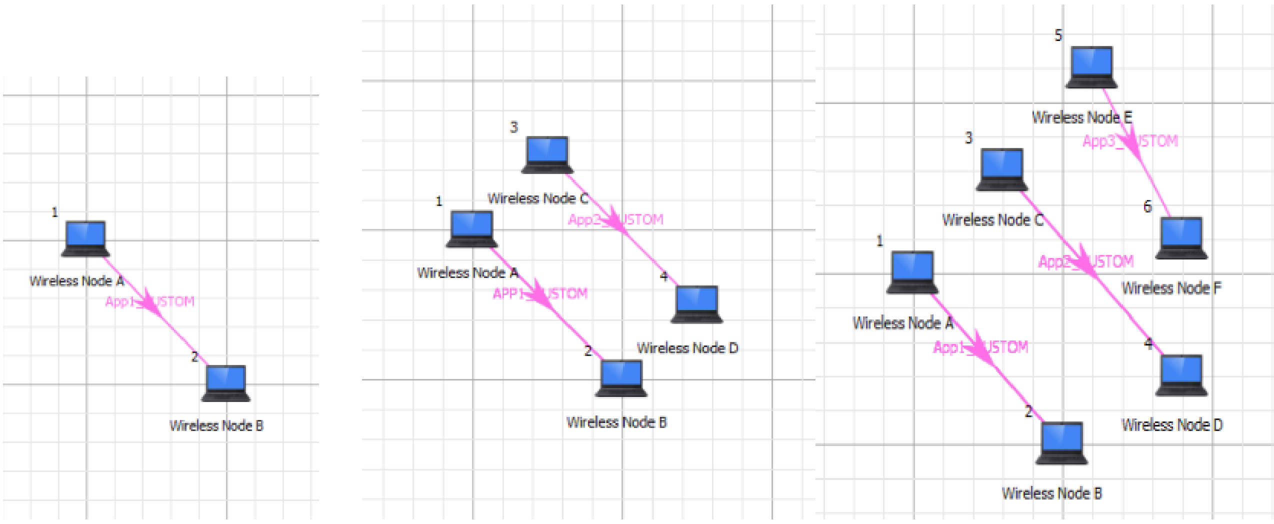
• Go to Application metrics and compute the Average delay by taking average of all row wise values of Delay

Delay = Average delay obtained from all Applications

NOTE – To create Graph in Excel , follow the steps

1. Copy the data in an Excel sheet.
2. Select the data. Go to Insert →Scatter (under Charts) → Scatter with Straight lines and Markers.

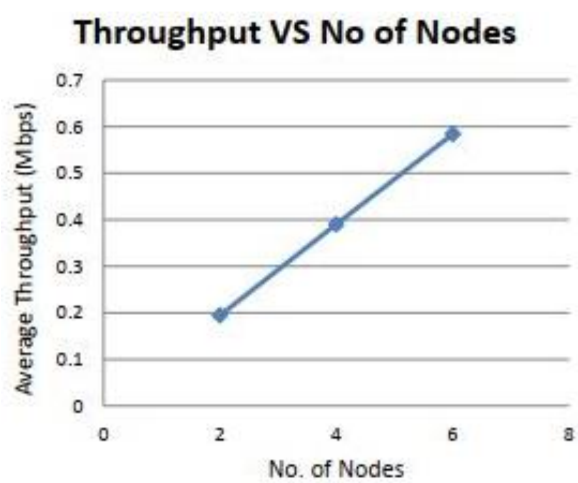
## LAYOUT: CSMA/CA:

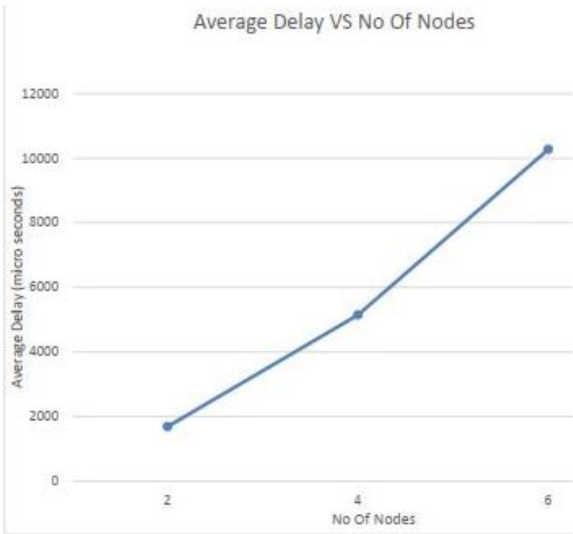



## TABULATION:

| No. of nodes | Throughput in Mbps | Delay in μs |
|---|---|---|
| 2 | 0.194 | 1691.53 |
| 4 | 0.389 | 5148.46 |
| 6 | 0.584 | 10275.27 |

## OUTPUT GRAPH:



**INFERENCE:**

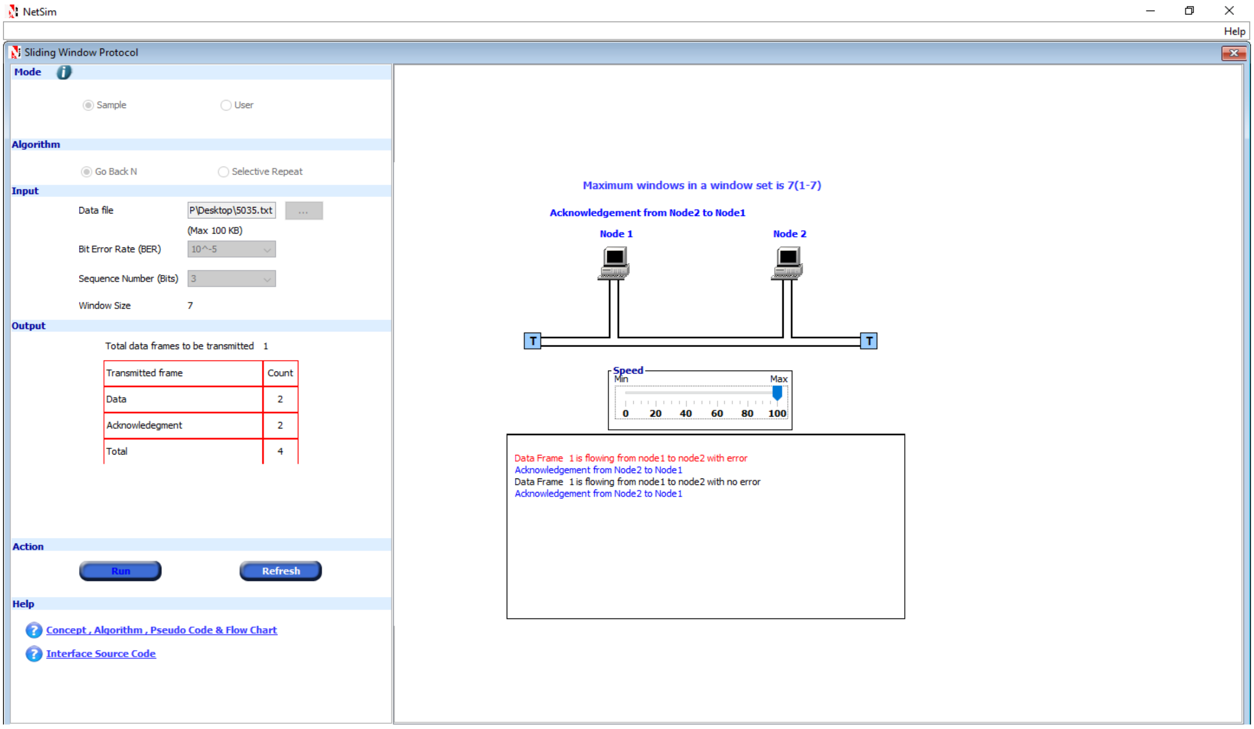
It is inferred that though we have to wait to send packets in slotted ALOHA, the throughput of slotted ALOHA is high. In CSMA/CD and CSMA/CA, as persistence decreases, throughput increases. As the number of nodes increases (2 →4 → 6) the throughput of the network increases because the channel is able to handle additional network traffic. However, if the number of nodes is increased further, the throughput may decrease as the network traffic is too high, and this leads to collisions. As the number of nodes increases, the delay also increases as it takes more time for a packet to reach its destination.

**RESULT:**

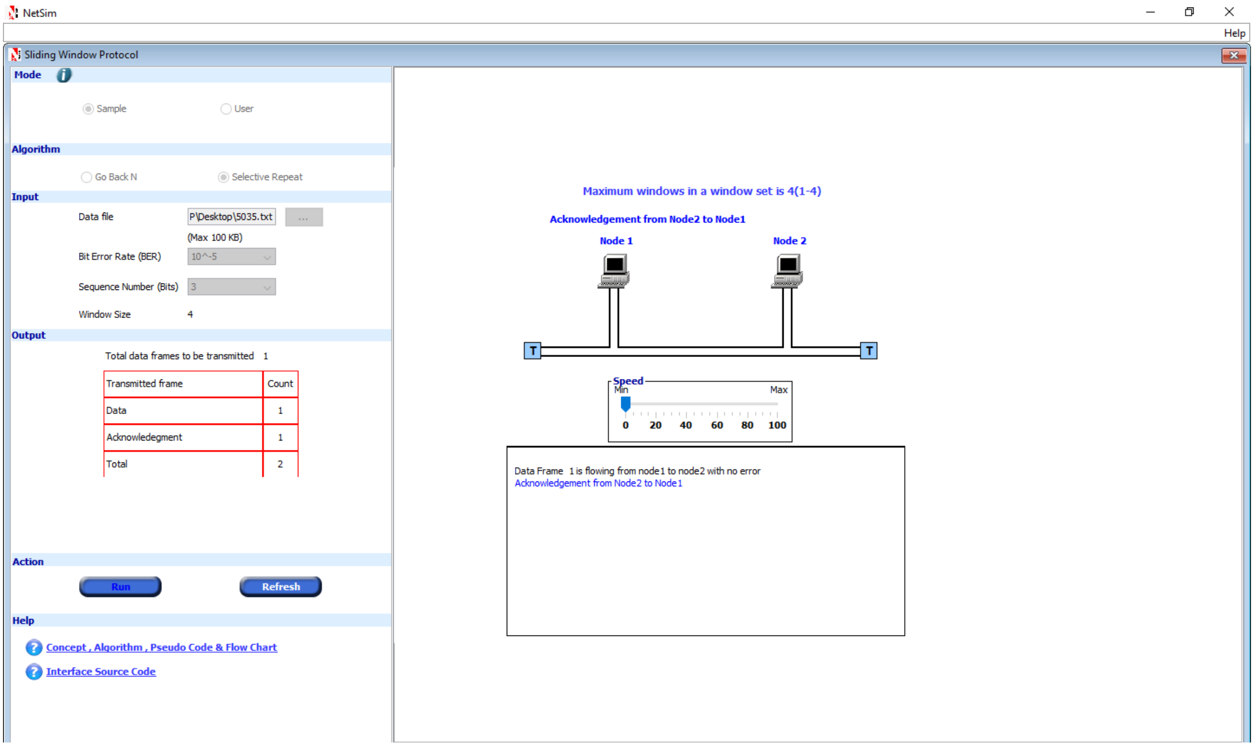
Thus, Slotted ALOHA, CSMA/CD and CSMA/CA protocols of MAC layer are analyzed using NETSIM and throughput for all protocols are calculated and computed.

# LAYOUT

## GO BACK ‘N’ PROTOCOL:




## SELECTIVE REPEAT PROTOCOL:



| EX:NO:3<br>15.09.2021 | **Analysis Of LLC Protocols Using NETSIM** |
|---|---|

## AIM:

To implement

- Go Back 'N' Protocol
- Selective Repeat Protocol

and to determine the throughput and delay for each.

## SOFTWARE REQUIRED:

- NETSIM Software

## THEORY:

### 1) <u>GO BACK 'N' PROTOCOL:</u>

Go Back 'N' is a connection-oriented transmission. The sender transmits the frames continuously. Each frame in the buffer has a sequence number starting from 1 and increasing up to the window size. The sender has a window i.e., a buffer to store the frames. This buffer size is the number of frames to be transmitted continuously. The size of the window depends on the protocol designer.

**<u>ALGORITHM:</u>**

- The source code transmits the frames continuously.
- Each frame in the buffer has a sequence number starting from 1 and increasing up to the window size.
- The source code has a window i.e., a buffer to store the frames. This buffer size is the number of frames to be transmitted continuously.
- The size of the window depends on the protocol designer.
- For the first frame, the receiving node forms a positive acknowledgement if the frame is received without any error.
- If subsequent frames are received without error (up to window size) cumulative positive acknowledgement is formed.
- If the subsequent frame is received with error, the cumulative acknowledgement error-free frames are transmitted. If, In the same window two frames or more frames are received with error, the second and the subsequent error frames are neglected. Similarly, even the frames received without error after the receipt of a frame with error are neglected.
- The source code re-transmits all frames of window from the first error frame.

## TABULATION:

### GO BACK 'N' (Sequence number: 3 bits)

| BER | TOTAL DATA FRAMES TO BE TRANSMITTED | DATA | ACKNOWLEDGEMENT | TOTAL |
|---|---|---|---|---|
| 10^-5 | 1 | 2 | 2 | 4 |
| 10^-6 | 1 | 2 | 2 | 4 |
| 10^-7 | 1 | 1 | 1 | 2 |
| 10^-8 | 1 | 1 | 1 | 2 |
| 10^-9 | 1 | 1 | 1 | 2 |
| NO ERROR | 1 | 1 | 1 | 2 |

### SELECTIVE REPEAT (Sequence number: 3 bits)

| BER | TOTAL DATA FRAMES TO BE TRANSMITTED | DATA | ACKNOWLEDGEMENT | TOTAL |
|---|---|---|---|---|
| 10^-5 | 1 | 1 | 1 | 2 |
| 10^-6 | 1 | 1 | 1 | 2 |
| 10^-7 | 1 | 1 | 1 | 2 |
| 10^-8 | 1 | 1 | 1 | 2 |
| 10^-9 | 1 | 1 | 1 | 2 |
| NO ERROR | 1 | 1 | 1 | 2 |

- If the frames are errorless in the next transmission and if the acknowledgement is error-free, the window slides by the number of error-free frames being transmitted.
- If the acknowledgement is transmitted with error, all the frames of window at source are re-transmitted and window doesn't slide.
- This concept of replacing the transmission from the first error frame in the window is called as **Go Back-N** transmission flow control protocol.

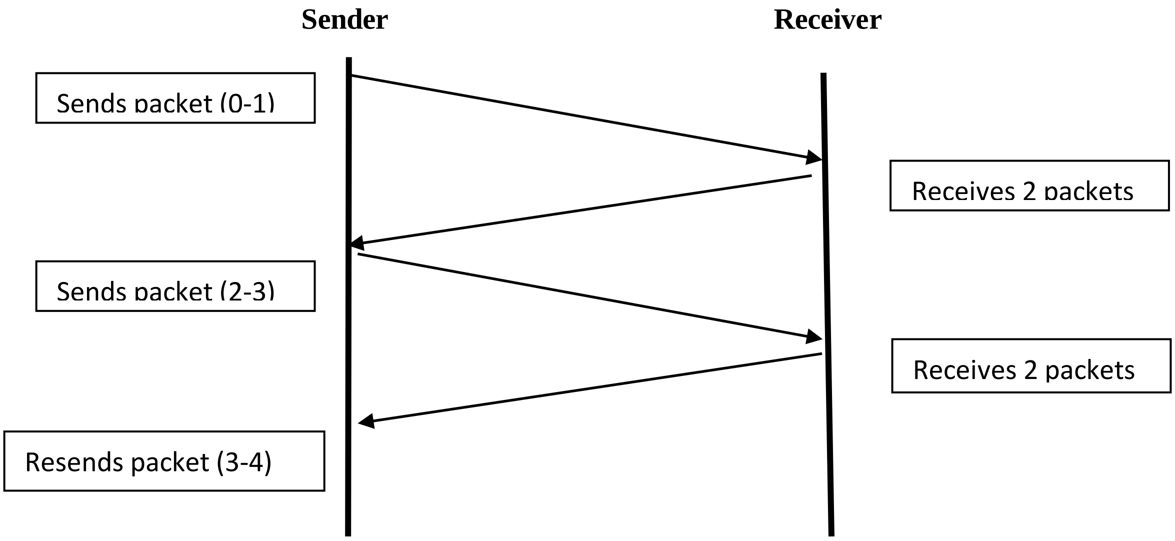
## 2) <u>SELECTIVE REPEAT PROTOCOL:</u>

It is similar to Go Back 'N' Protocol, but the sender send frame only after the reception of ACK signal. It may be used as a protocol for delivery and ACK for message units for delivery of subdivided message. It is used as a protocol for delivery of message sender continuous to send frames specifies by windows size even after becoming frameless. Once the sender has sent all the frame in its windows, it resends the frame number given by ACK and continuous where it left off.

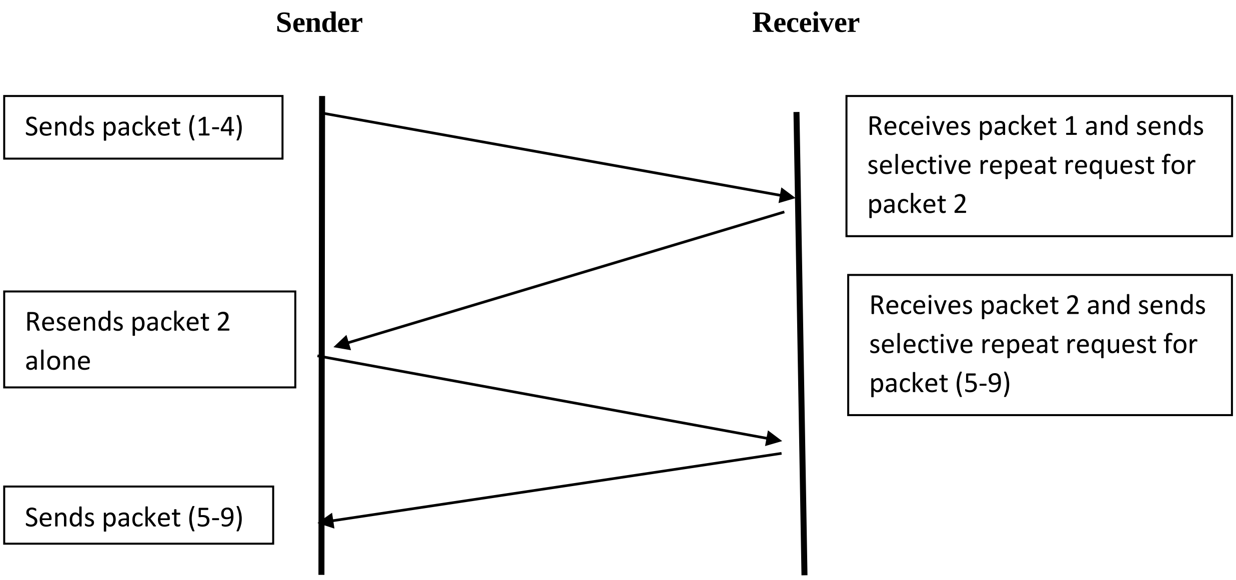
**<u>ALGORITHM:</u>**

- The source node transmits the frames continuously.
- Each frame in the buffer has a sequence number starting from 1 and increasing up to the window size.
- The source node has a window i.e., buffer to store the frames. This buffer size is the number of frames to be transmitted continuously.
- The receiver has a buffer to store the received frames. The size of the buffer depends upon the window size defined by the protocol designer.
- The size of the window depends according to the protocol designer.
- The source node transmits frames continuously till the window size is exhausted. If any of the frames are received with error only those frames are requested for retransmission (with a negative acknowledgement).
- If all the frames are received without error, a cumulative positive acknowledgement is sent.
- If there is an error in frame 3, an acknowledgement for the frame 2 is sent and then only frame 3 is retransmitted. Now the window slides to get the next frames to the window.
- If acknowledgement is transmitted with error, all the frames of window are retransmitted. Else ordinary window sliding takes place. (*In implementation part, Acknowledgement error is not considered).
- If all the frames transmitted are errorless the next transmission is carried out for the new window.
- This concept of repeating the transmission for the error frames only is called **Selective Repeat** transmission flow control protocol.

**Go Back-N Protocol:**




**Selective Repeat Protocol:**



## PROCEDURE:

1.Open NETSIM.

2.Click Programming → Transmission Flow Control.

3.Select Sample.

4.Select **Go Back-N** transmission.

5.Enter input data and BER.

6.Click **Link** to execute the program.

7.Repeat steps 1 to 3.

8.Select **Selective Repeat** protocol.

9.Enter input data and BER.

10.Click **Link** to execute the program.

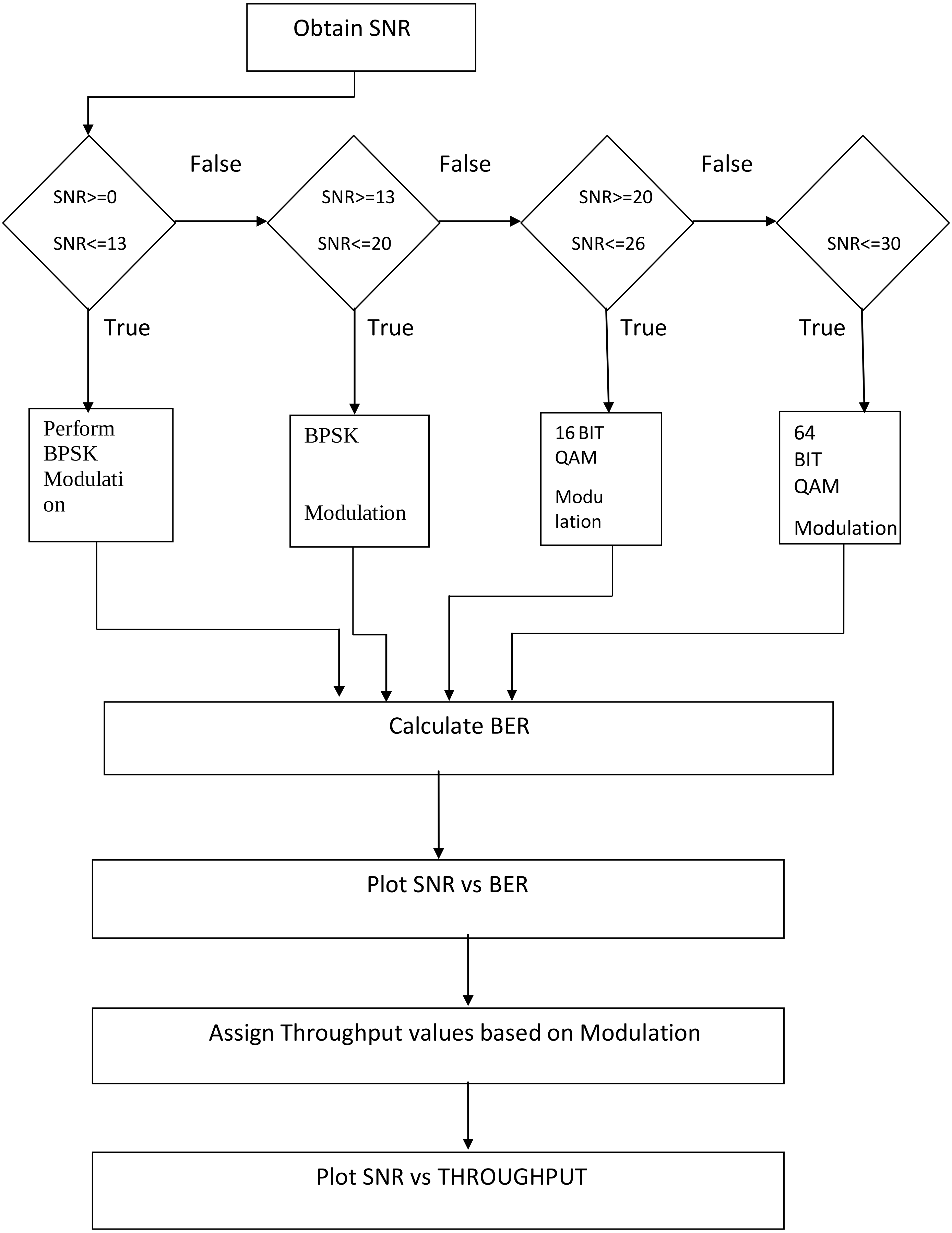
## INFERENCE:

It is found that Selective Repeat Protocol is the most optimum out of these two LLC protocols. But the complexity equipment's are high. So, a tradeoff exists the total number of transmissions taken place and complexity.

## RESULT:

Thus, the LLC protocols such as "Go Back 'N'" and "Selective Repeat" were studied using NETSIM and their performance were analyzed.

**FLOWCHART:**


Obtain SNR
SNR>=0
SNR<=13
False
SNR>=13
SNR<=20
False
SNR>=20
SNR<=26
False
SNR<=30
True
True
True
True
Perform BPSK Modulation
BPSK Modulation
16 BIT QAM Modulation
64 BIT QAM Modulation
Calculate BER
Plot SNR vs BER
Assign Throughput values based on Modulation
Plot SNR vs THROUGHPUT

| EX:NO:4<br>18.09.2021 | ADAPTIVE MODULATION AND CODING |
|---|---|

## AIM:

To perform Adaptive modulation and coding and observe its characteristics.

## SOFTWARE REQUIRED:

MATLAB R2017a

## THEORY:

Adaptive Modulation is a technique which allows a radio to change its speed (modulation rate) as conditions in the radio network change. Interference from outside sources, such as changes in the environment (temperature, tree foliage, moving objects)all effect radio coverage. Adaptive modulation enables robust and spectrally efficient transmission over time varying channels. The basic premise is to estimate the channel at the receiverand feed this channel back to the transmitter so that the transmission can be adapted relative to channel characteristics. Thus the system can be designed efficiently for the worst case channel characteristics.

## ALGORITHM:

1) Range of SNR for each modulation technique is calculated such that it provides areliable transmission ( minimum BER of 10^-7)
2) Random signal is generated , modulated and demodulated based on SNR and BERis calculated
3) The Throughput is assigned based on the modulation and demodulation technique

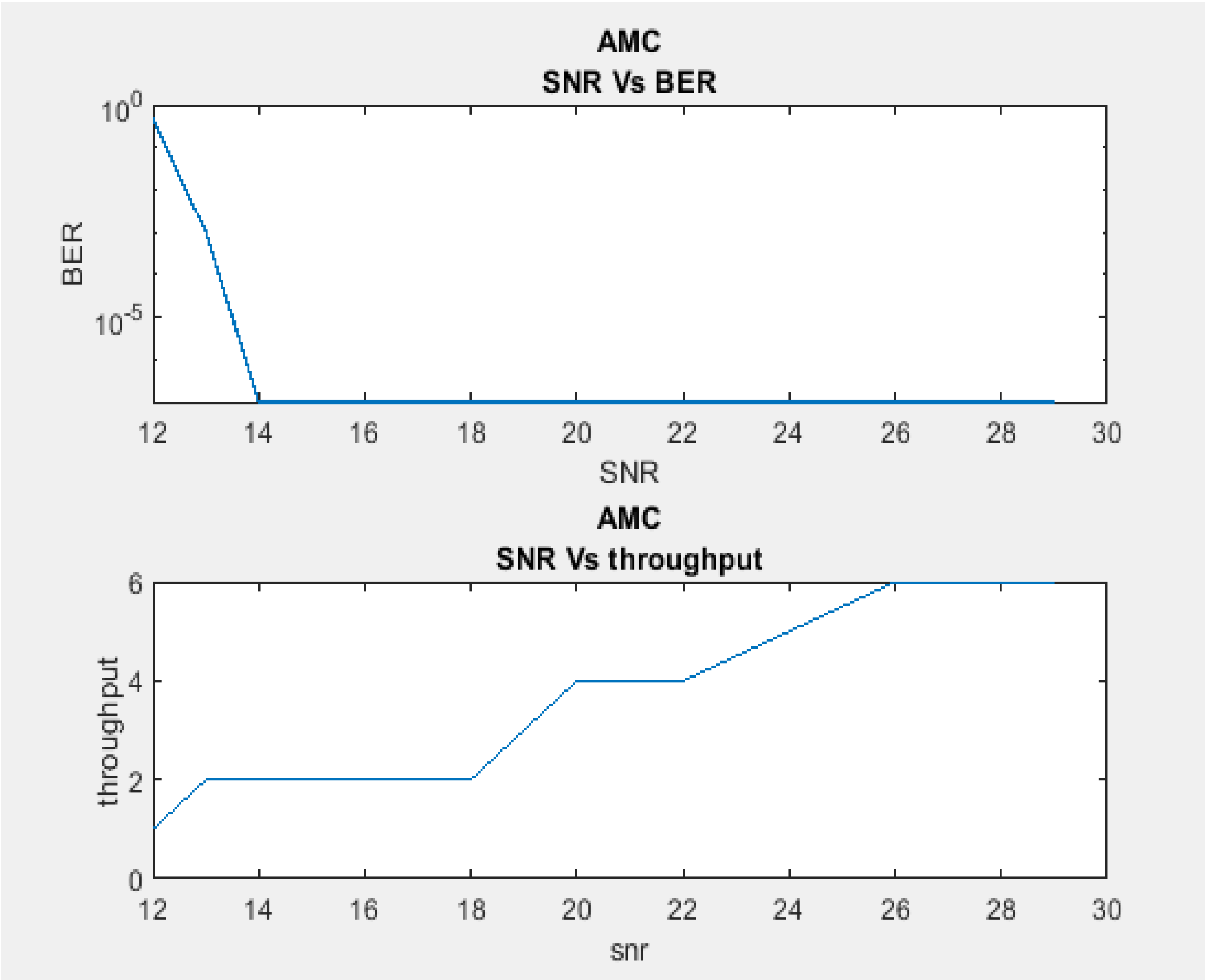
## CODE:

```
clc;
close all;
snr= randi([10,30],1,15);
snr=sort(snr);
err=[];
for i=1:length(snr)
```

```
 if(snr(i)>=0 && snr(i)<13)
a=randi([0,1],1,1000);
b=pskmod(a,2); end
if(snr(i)>=13 && snr(i)<20)
a=randi([0,3],1,1000);
b=pskmod(a,4);
 end
if(snr(i)>=20 && snr(i)<26)
a=randi([0,15],1,1000);
b=qammod(a,16);
 end
if(snr(i)>=26 && snr(i)<=30)
a=randi([0,63],1,1000);
b=qammod(a,64);
end
 N0=1/10^(snr(i)/10);
g=awgn(b,snr(i));
n=sqrt(N0/2)*(randn(1,length(a))+ 1i*randn(1,length(a)));
f= g+n;
if(snr(i)>=0 && snr(i)<13)
d=pskdemod(f,2);
end
if(snr(i)>=13 && snr(i)<20)
d=pskdemod(f,4);
end
if(snr(i)>=20 && snr(i)<26)
d=qamdemod(f,16);
end
if(snr(i)>=26 && snr(i)<=30)
d=qamdemod(f,64);
end [n,r]=biterr(a,d);
 if(r==0)
```

```
r=1e-7;
end
err=[err r];
end
subplot(2,1,1);
 semilogy(snr,err,'linewidth',1);
xlabel('SNR');
ylabel('BER');
title({'AMC';'SNR Vs BER'});
thruput=[];
for i=1:length(snr)
if(snr(i)>=10 && snr(i)<13)
 thruput=[thruput 1];
end
if(snr(i)>=13 && snr(i)<20)
thruput=[thruput 2];
end
if(snr(i)>=20 && snr(i)<=25)
thruput=[thruput 4];
end
if(snr(i)>=26 && snr(i)<=30)
thruput=[thruput 6];
end
end
subplot(2,1,2);
plot(snr ,thruput);
xlabel('snr');
ylabel('throughput');
title({'AMC';'SNR Vs throughput'});
```

## OUTPUT:



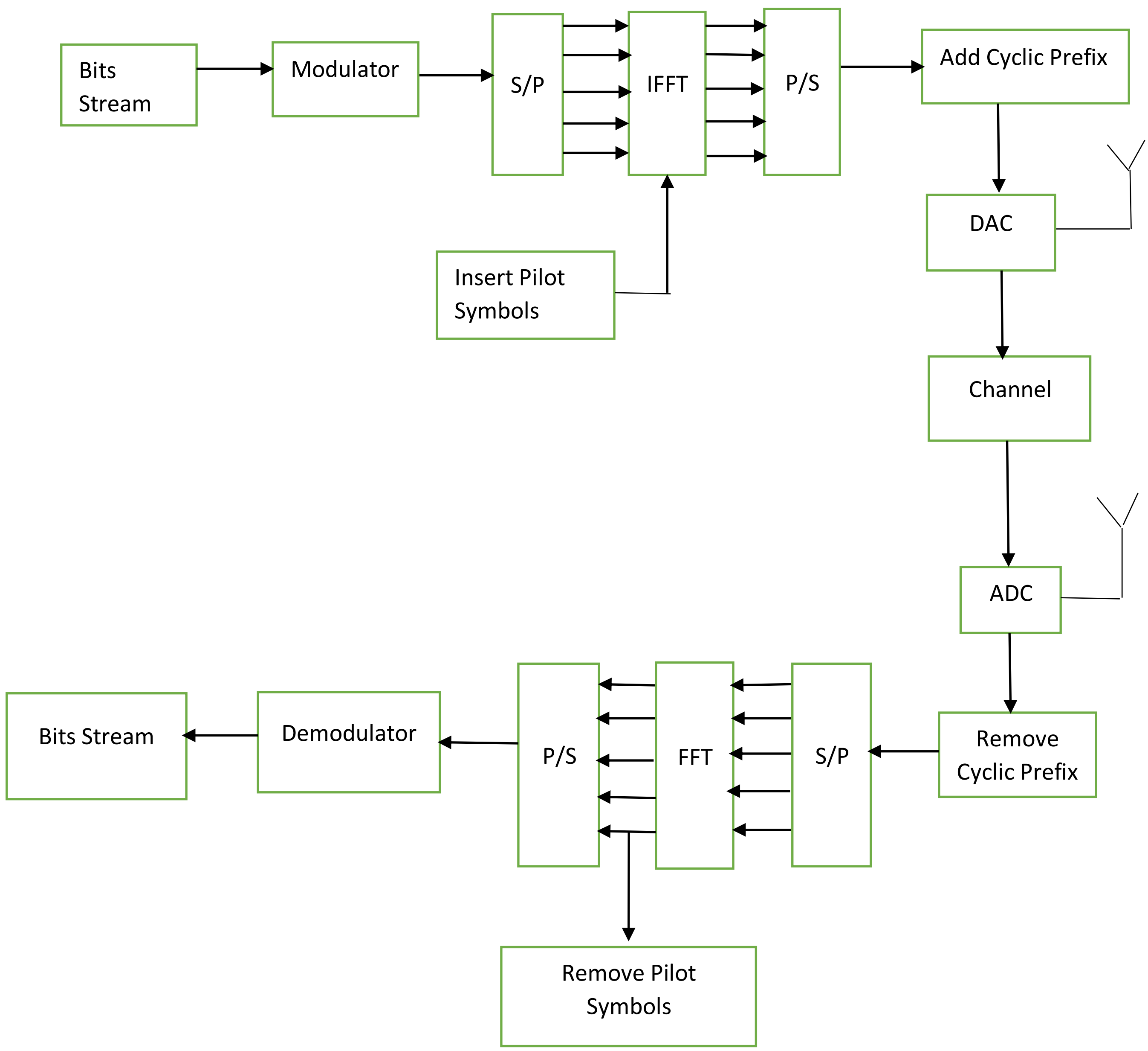
## INFERENCE:

- When the SNR is low, BER is high and hence Throughput is low.
- When the SNR is high, BER is low and hence Throughput is high.

## RESULT:

Thus Adaptive Modulation and coding has been implemented and analyzed using MATLAB.

# **Block Diagram:**



| EX:NO:5<br><br>21.09.2021 | IMPLEMENTATION OF MULTICARRIER MODULATION (OFDM) |
|---|---|

## Aim:

To study the characteristics of orthogonal frequency division multiplexing using MATLAB and to plot the BER vs SNR graph.

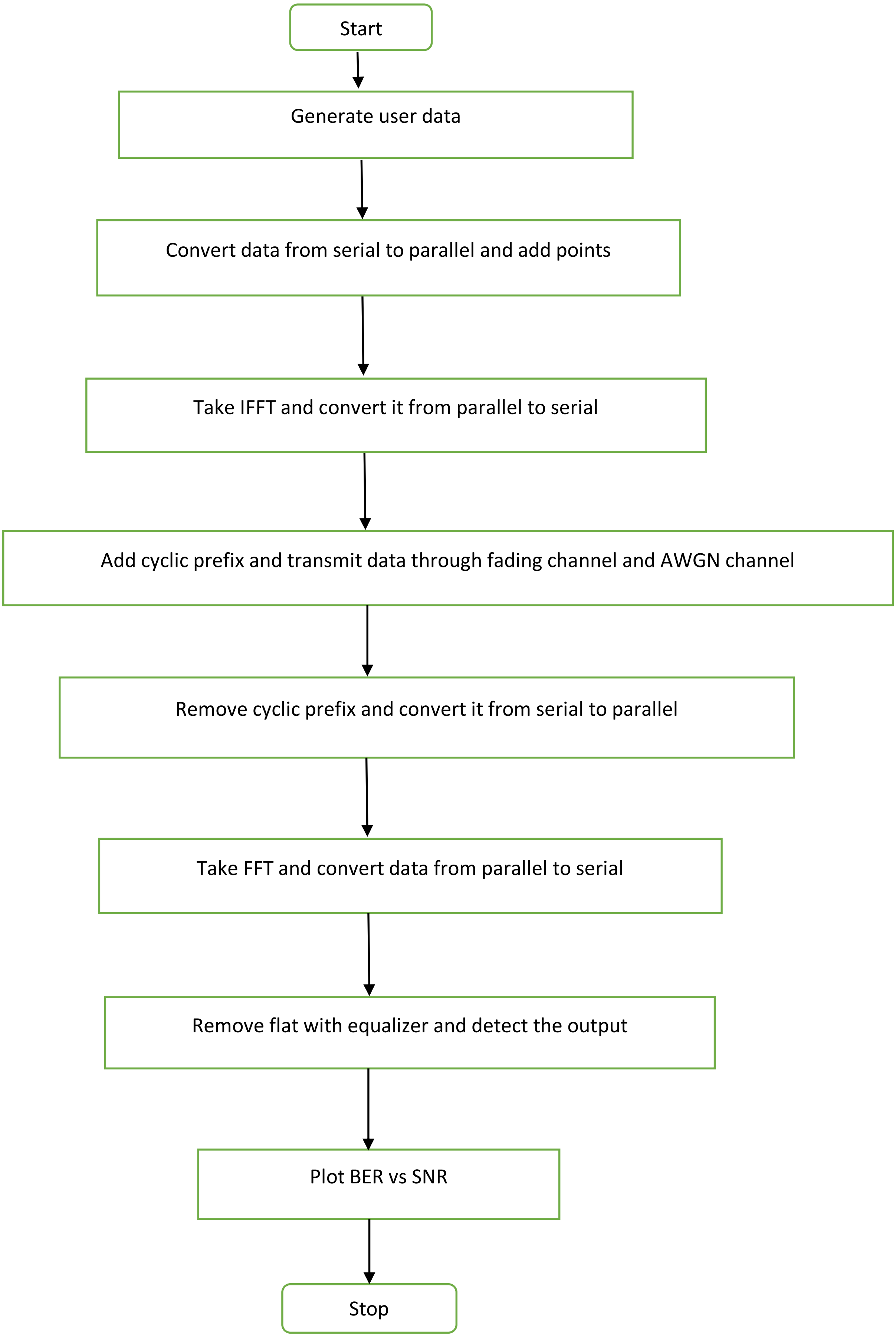
## Software Required:

MATLAB R2017a

## Algorithm:

- Generate user data.
- Convert data from serial to parallel and add points.
- Take IFFT and convert it from parallel to serial.
- Add cyclic prefix and transmit data through fading channel and AWGN channel.
- Remove cyclic prefix and convert it from serial to parallel.
- Take FFT and convert data from parallel to serial.
- Remove flat with equalizer and detect the output.
- Plot BER vs SNR.

## Flow chart:


Start
Generate user data
Convert data from serial to parallel and add points
Take IFFT and convert it from parallel to serial
Add cyclic prefix and transmit data through fading channel and AWGN channel
Remove cyclic prefix and convert it from serial to parallel
Take FFT and convert data from parallel to serial
Remove flat with equalizer and detect the output
Plot BER vs SNR
Stop

## Theory:

Orthogonal FDM (OFDM) spread spectrum technique distributes data over a large number of carriers that are spaced apart at precise frequencies. This spacing provides the orthogonality in this technique which prevents the demodulators from seeing frequencies other than their own. The benefits of OFDM are high spectral efficiency, resiliency to RF interference, and lower multi-path distortion. OFDM is currently the basis of the physical layer of the major wireless systems, such as WiMAX ou IEEE 802.11n, but it can be also found in cable technology systems such as DSL. There are several variants of OFDM such as VOFDM (vector) or COFDM (coded).

## Code:

```
clc;
close all;
x=randi([0 1],1,4096);
snr=0:30;
biterror=[];
for i =1:4
    y=pskmod(x,2^i);
    if(i==3)
        y=qammod(x,2^(i+1));
    end
    if(i==4)
        y=qammod(x,2^(i+2));
    end
    p=reshape(y,64,64);
    q=ifft(p,64);
    s=reshape(q,1,4096);
    be=[];
    for j=0:1:30
        h=1/sqrt(rand(1,1)+i*sqrt(rand(1,1)));
        r=h*s;
        n=awgn(r,j,'measured');
        m=inv(h)*n;
        p11=reshape(m,64,64);
        q11=fft(p11,64);
        s11=reshape(q11,1,4096);
```

## **OUTPUT:**

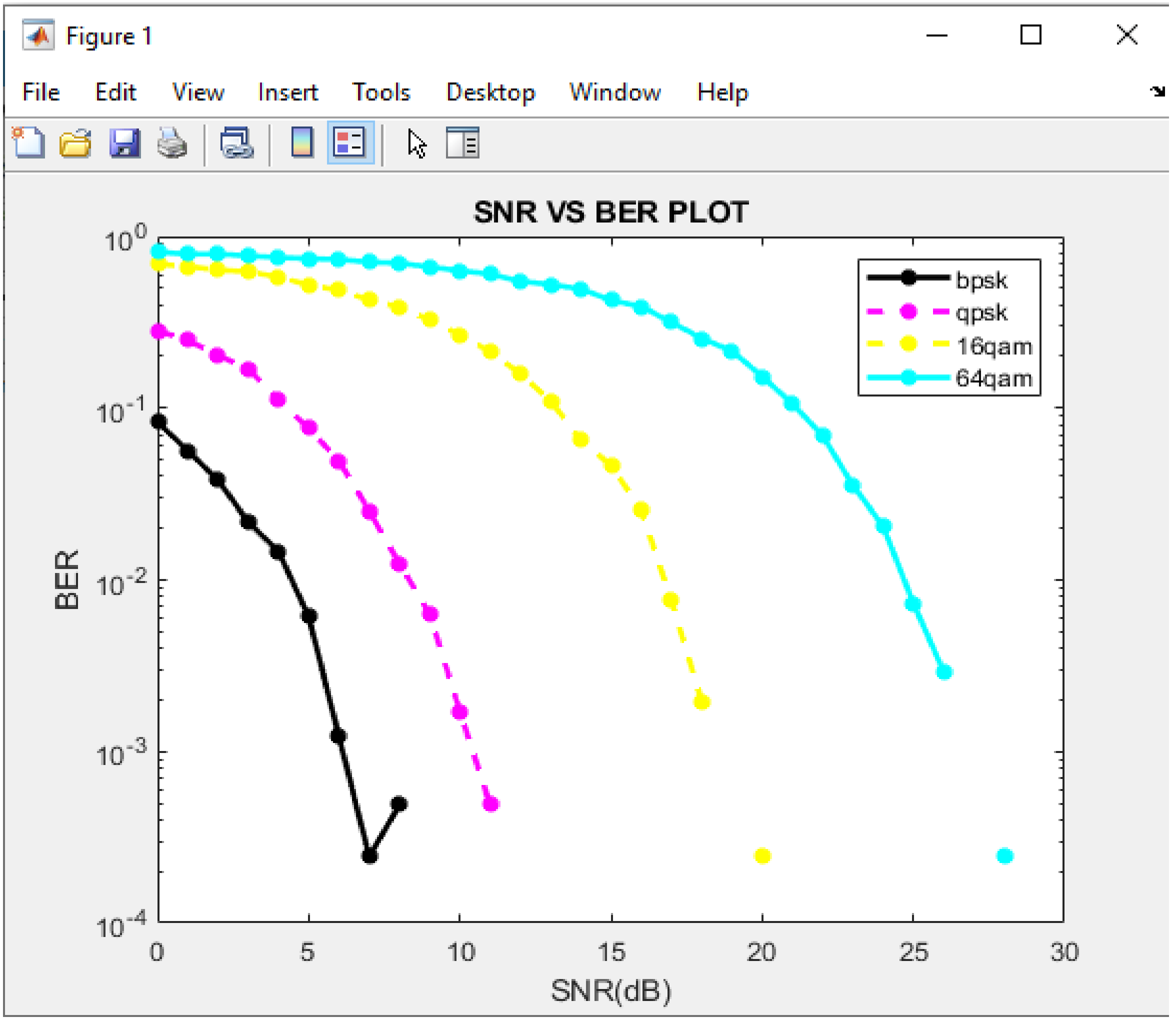


Figure 1
File Edit View Insert Tools Desktop Window Help
SNR VS BER PLOT
BER
SNR(dB)
bpsk
qpsk
16qam
64qam
$10^0$
$10^{-1}$
$10^{-2}$
$10^{-3}$
$10^{-4}$
0
5
10
15
20
25
30

```
        y11=pskdemod(s11,2^i);
        if(i==3)
            y11=qamdemod(s11,2^(i+1));
        end
        if(i==4)
            y11=qamdemod(s11,2^(i+2));
        end
        [num1,e1]=symerr(y11,x);
        be=[be e1];
    end
    biterror(i,:)=be;
end
semilogy(snr, biterror(1,:),'*-k','linewidth',2);hold on;
semilogy(snr, biterror(2,:),'*--m','linewidth',2);hold on;
semilogy(snr, biterror(3,:),'*--y','linewidth',2);hold on;
semilogy(snr, biterror(4,:),'*-c','linewidth',2);hold on;
xlabel('SNR(dB)');
ylabel('BER');
title('SNR VS BER PLOT');
legend('bpsk','qpsk','16qam','64qam');
```

## Inference:

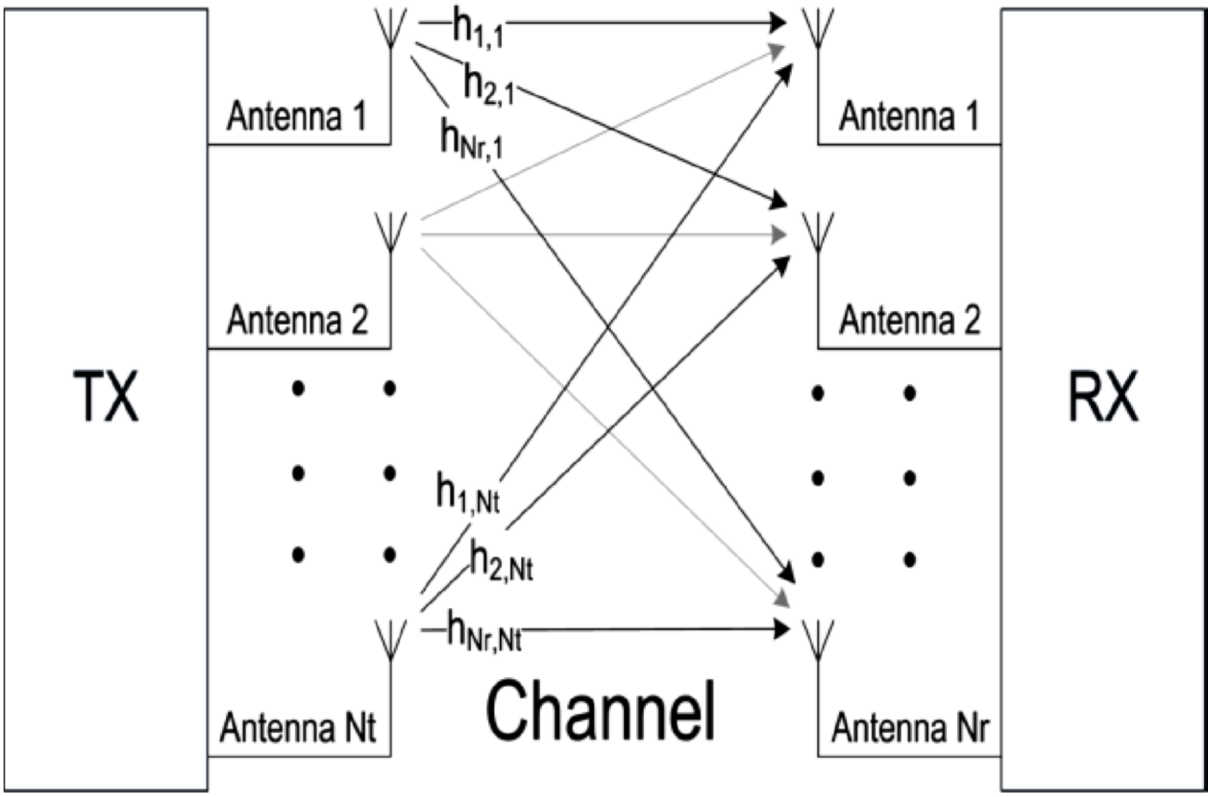
The main advantage of OFDM over single-carrier schemes is its ability to cope with severe channel conditions (for example, attenuation at high frequencies in long copper wire, narrowband interference and frequency – selective fading due to multipath) without complex equalization filters.

## Result:

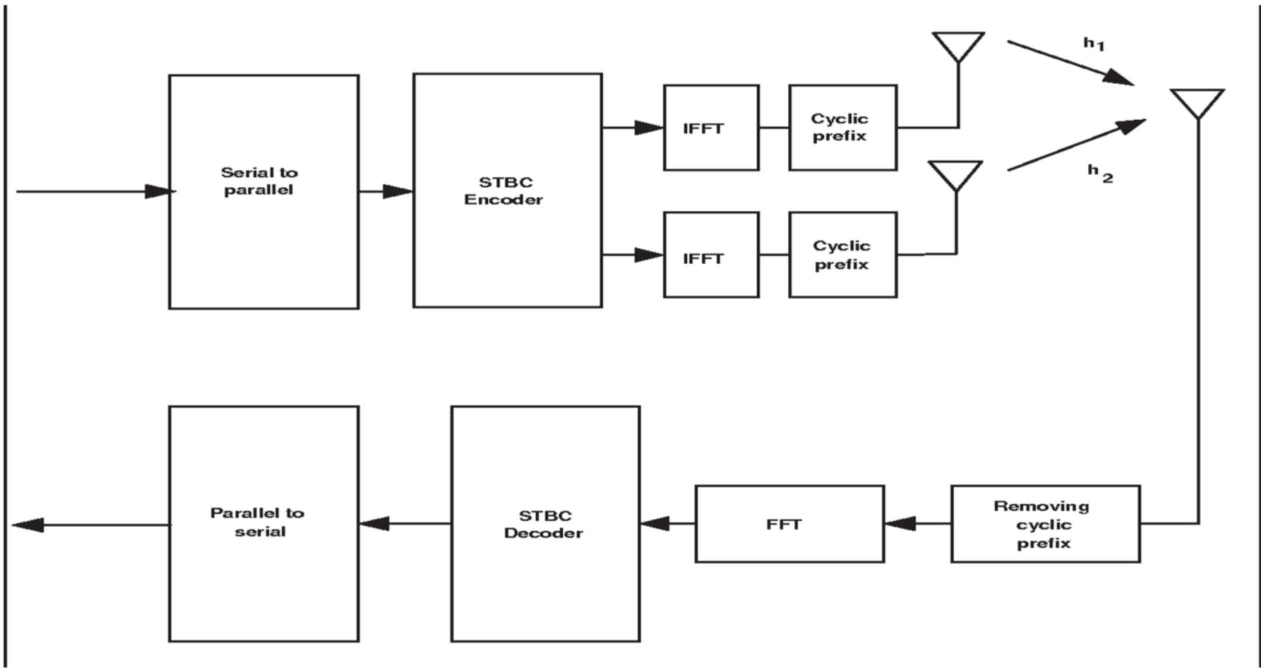
The characteristics of OFDM is studied using MATLAB and BER vs SNR is plotted.

## BLOCK DIAGRAM:

**MIMO:**




**STBC:**



| EX:NO:6<br><br>22.09.2021 | # SPACE TIME BLOCK CODES-ALAMOUTI |
|---|---|

AIM:

To study Space Time Block Codes, using MATLAB.

## SOFTWARE USED:

MATLAB software.

## THEORY:

## MIMO TECHNOLOGY:

MIMO stands for Multiple-Input Multiple-Output, is a wireless technology that uses multiple transmitters and receivers to transfer more data at the same time. All wireless products with 802.11n support MIMO. The technology helps allow 802.11n to reach higher speeds than products without 802.11n.

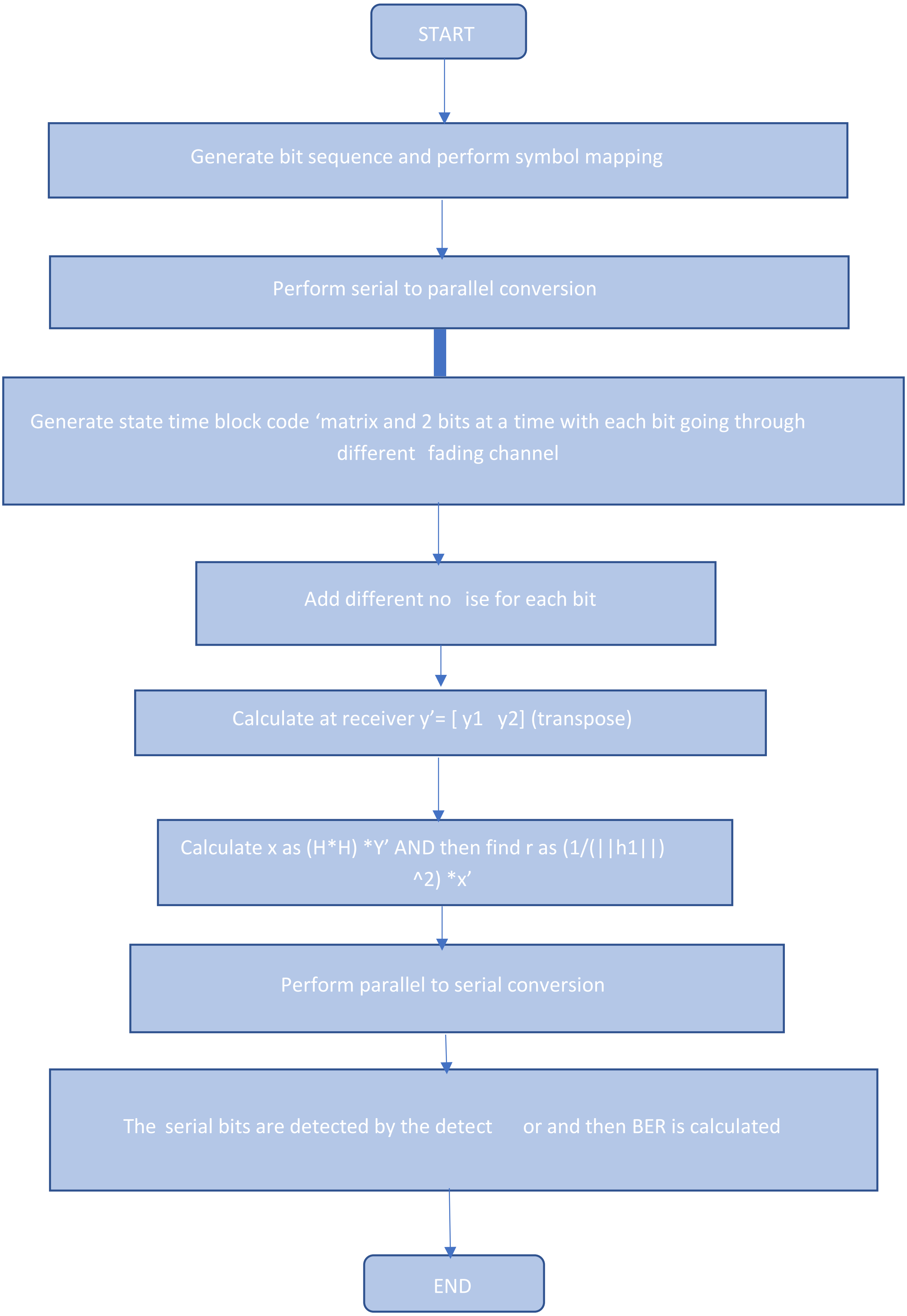
MIMO technology uses a natural radio-wave phenomenon called multipath. With multipath, transmitted information bounces off walls, ceilings and other objects, reaching the receiving antenna multiple times at different angles and slightly different times. With multipath, MIMO technology uses multiple, smart transmitters and receivers with an added spatial dimension, increasing performance and range. It increases receiver signal-capturing power by enabling antennas to combine data streams arriving from different paths at different times.

It introduces signalling degrees of freedom that were absent in SISO system. This is referred to as the spatial degree of freedom. The spatial degrees of freedom can either be exploited for "Diversity" or "Multiplexing" or a combination of the two. When antennas out number spatial streams, the antennas can add receiver diversity and increase range. In simple terms, diversity means redundancy. Diversity can also be achieved using multiple transmit antennas by using Space Time Coding (STC) techniques.

## SPACE-TIME BLOCK CODING:

Space-time block codes (STBC) are a generalized version of Alamouti scheme. These schemes have the same key features. Therefore, these codes are orthogonal and can achieve full transmit diversity specified by the number of transmit antennas. In another word, spacetime block codes are a complex version of Alamouti's space-time code in, where the encoding and decoding schemes are the same as there in the Alamouti space-time code in both the transmitter and receiver sides.

**FLOW CHART:**


START
Generate bit sequence and perform symbol mapping
Perform serial to parallel conversion
Generate state time block code 'matrix and 2 bits at a time with each bit going through different fading channel
Add different no ise for each bit
Calculate at receiver y'= [ y1 y2] (transpose)
Calculate x as (H*H) *Y' AND then find r as (1/(||h1||)^2) *x'
Perform parallel to serial conversion
The serial bits are detected by the detect or and then BER is calculated
END

The data are constructed as a matrix which has its rows equal to the number of the transmit antennas and its columns equal to the number of the time slots required to transmit the data. At the receiver side, when signals are received, they are first combined and then sent to the maximum likelihood detector where the decision rules are applied.

Space-time block codes were designed to achieve the maximum diversity order for the given number of transmit and receive antennas subject to the constraint of having a simple linear decoding algorithm. This has made space-time block codes a very poplar scheme and most widely used. Space-time block codes and indeed many other space-time techniques including STTCs are designed for coherent detection where channel estimation is necessary. There is a substantial literature addressing the channel estimation issue for multiple-input multiple-output (MIMO) systems, ranging from standard training-based techniques that rely on pilot symbols in the data stream to blind which does not require pilot sequences and semi blind estimation where observations corresponding to data and pilot are used jointly. Other authors have considered non-coherent detection schemes based on differential encoding which do not require channel state information (CSI).

Although these methods avoid the need for channel estimation, they often suffer from problems such as error propagation. Training-based methods seem to give very good results on the performance of channel estimation at the receiver. Pure training-based schemes can be considered as an advantage when an accurate and reliable MIMO channel needs to be obtained. However, this could also be a disadvantage when bandwidth efficiency is required. This is because pure training-based schemes reduce the bandwidth efficiency considerably due to the use of a long training sequence which is necessarily needed in order to obtain a reliable MIMO channel estimate. Because of the computation complexity of blind and semiblind methods, many wireless communication systems still use pilot sequences to estimate the channel parameters at the receiver side.

**ENCODING**:

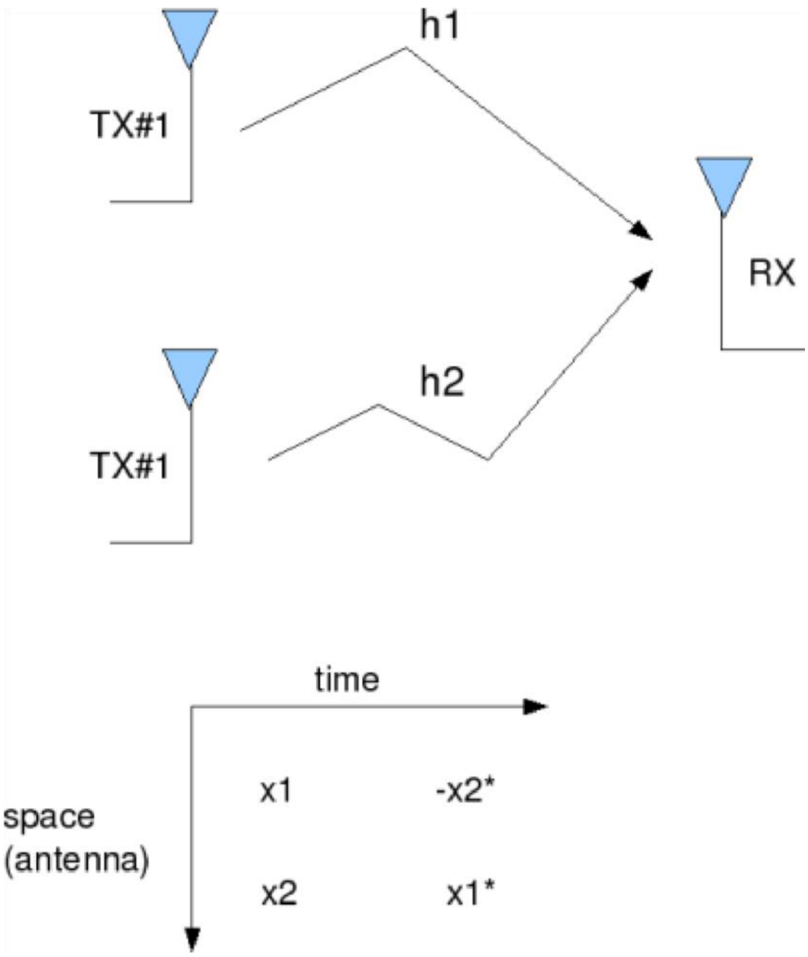
It was designed for a two-transmit antenna system and has the coding matrix:

$$C_2 = \begin{bmatrix} c_1 & c_2 \\ -c_2^* & c_1^* \end{bmatrix}$$

Where * denotes complex conjugate.

It is readily apparent that this is a rate-1 code. It takes two time slots to transmit two symbols. Using the optimal decoding scheme discussed below, the Bit Error Rate (BER) of this STBC is equivalent to 2 n R – branch maximal ratio combining (MRC). This is a result of the perfect orthogonality between the symbols after receive processing – there are two copies of each symbol transmitted and n R copies received.

## DERIVATION:




At time t1, transmit x1 from antenna 1 and x2 from antenna 2,

y1= h1x1+h2x2+n1

At time t2, transmit -x2* from antenna 1 and x1* from antenna 2,   y2=-h11x2*+h21x1*+n2

where $n_1$, $n_2$ denote AWGN noise and $y_1$, $y_2$ denote the received vectors.

These equations can be written in matrix equation form as

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} h_1 & h_2 \\ h_2^* & -h_1^* \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}$$

$H = \begin{bmatrix} h_1 & h_2 \\ h_2^* & -h_1^* \end{bmatrix}$ is an 2X2 Channel matrix where a* represents the Complex Conjugate of a.

Receiver r=H^H*y

## DECODING:

One particularly attractive feature of orthogonal STBCs is that maximum likelihood decoding can be achieved at the receiver with only linear processing. In order to consider a decoding method, a model of the wireless communications system is needed.

**ALGORITHM**:

1. Generate random binary sequence of +1s and -1s.
2. Group them into pairs of two symbols.
3. Code it as per the Alamouti Space Time Code, multiply the symbols with the channels and then add AWGN.
4. Equalize the received symbols.
5. Perform hard decision decoding and count the bit errors.
6. Repeat for multiple values and plot the simulation.

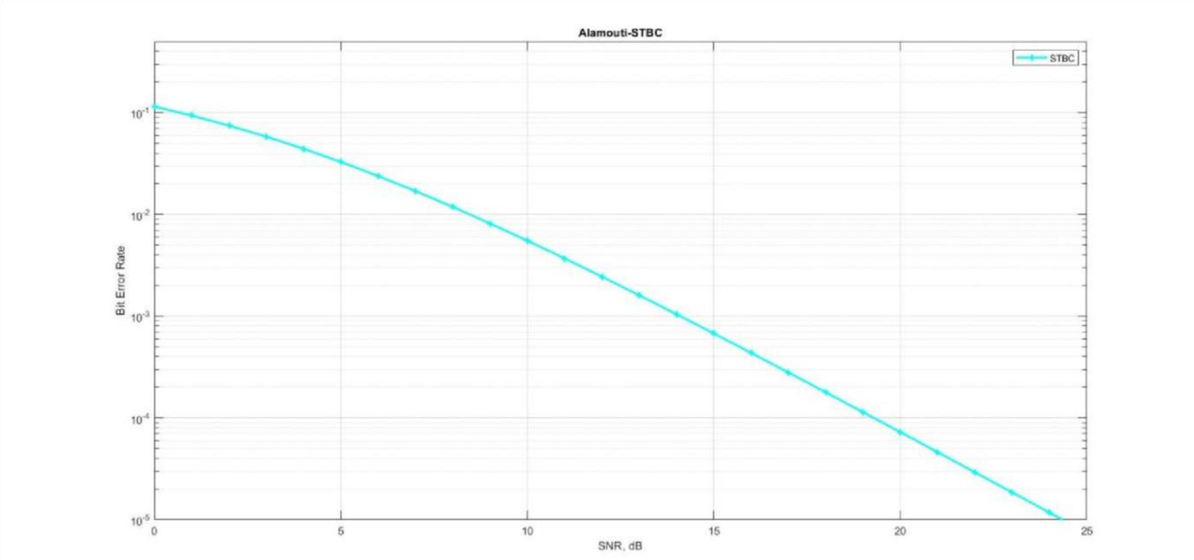
## CODE:

```
clc

clear

all

close

all

N = 10^6; % number of bits or symbols

SNR = [0:25];  for ii = 1:length(SNR)

% Transmitter  ip = rand(1,N)>0.5;  s =

2*ip-1;

% Alamouti STBC

sCode = zeros(2,N);

sCode(:,1:2:end) = (1/sqrt(2))*reshape(s,2,N/2);

sCode(:,2:2:end) = (1/sqrt(2))*(kron(ones(1,N/2),[-1;1]).*flipud(reshape(conj(s),2,N/2)));
h = 1/sqrt(2)*[randn(1,N) + j*randn(1,N)];  hMod

= kron(reshape(h,2,N/2),ones(1,2));  n =

1/sqrt(2)*[randn(1,N) + j*randn(1,N)];  %

Channel and Noise addition  y =

sum(hMod.*sCode,1) + 10^(-SNR(ii)/20)*n;

% Receiver

yMod = kron(reshape(y,2,N/2),ones(1,2));

yMod(2,:) = conj(yMod(2,:));  % forming
```

$$\begin{bmatrix} R1 \\ R2 \end{bmatrix} = \begin{bmatrix} h1^* & h21 \\ h21^* & -h1 \end{bmatrix} \begin{bmatrix} h11 & h21 \\ h21^* & -h11^* \end{bmatrix} \begin{bmatrix} x1 \\ x2 \end{bmatrix} + \begin{bmatrix} h11^* & h21 \\ h21^* & h11 \end{bmatrix} \begin{bmatrix} h1 \\ h2^* \end{bmatrix}$$

$$\begin{aligned} \widehat{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}} &= (H^H H)^{-1} H^H \begin{bmatrix} y_1 \\ y_2^* \end{bmatrix} \\ &= (H^H H)^{-1} H^H \left( H \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2^* \end{bmatrix} \right) \\ &= \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + (H^H H)^{-1} H^H \begin{bmatrix} n_1 \\ n_2^* \end{bmatrix} \end{aligned}$$

## OUTPUT:



```
the equalization matrix  hEq = zeros(2,N);
hEq(:,[1:2:end]) = reshape(h,2,N/2);

hEq(:,[2:2:end]) = kron(ones(1,N/2),[1;-1]).*flipud(reshape(h,2,N/2));
hEq(1,:) = conj(hEq(1,:));  hEqPower = sum(hEq.*conj(hEq),1);  yHat =
sum(hEq.*yMod,1)./hEqPower;  yHat(2:2:end) = conj(yHat(2:2:end));
% receiver - hard decision decoding  ipHat = real(yHat)>0;  % counting
the errors  nErr(ii) = size(find([ip- ipHat]),2);  end

simBer = nErr/N; % simulated ber
SNRLin = 10.^(SNR/10);

pAlamouti = 1/2 - 1/2*(1+2./SNRLin).^(-1/2);
berAlamouti = pAlamouti.^2.(1+2(1-pAlamouti));
close all  figure

semilogy(SNR,berAlamouti,'c+-','LineWidth',2);
hold on  axis([0 25 10^-5 0.5])
grid on
legend('STBC');
xlabel('SNR, dB');
ylabel('Bit Error
Rate');
title('Alamouti-
STBC');
```

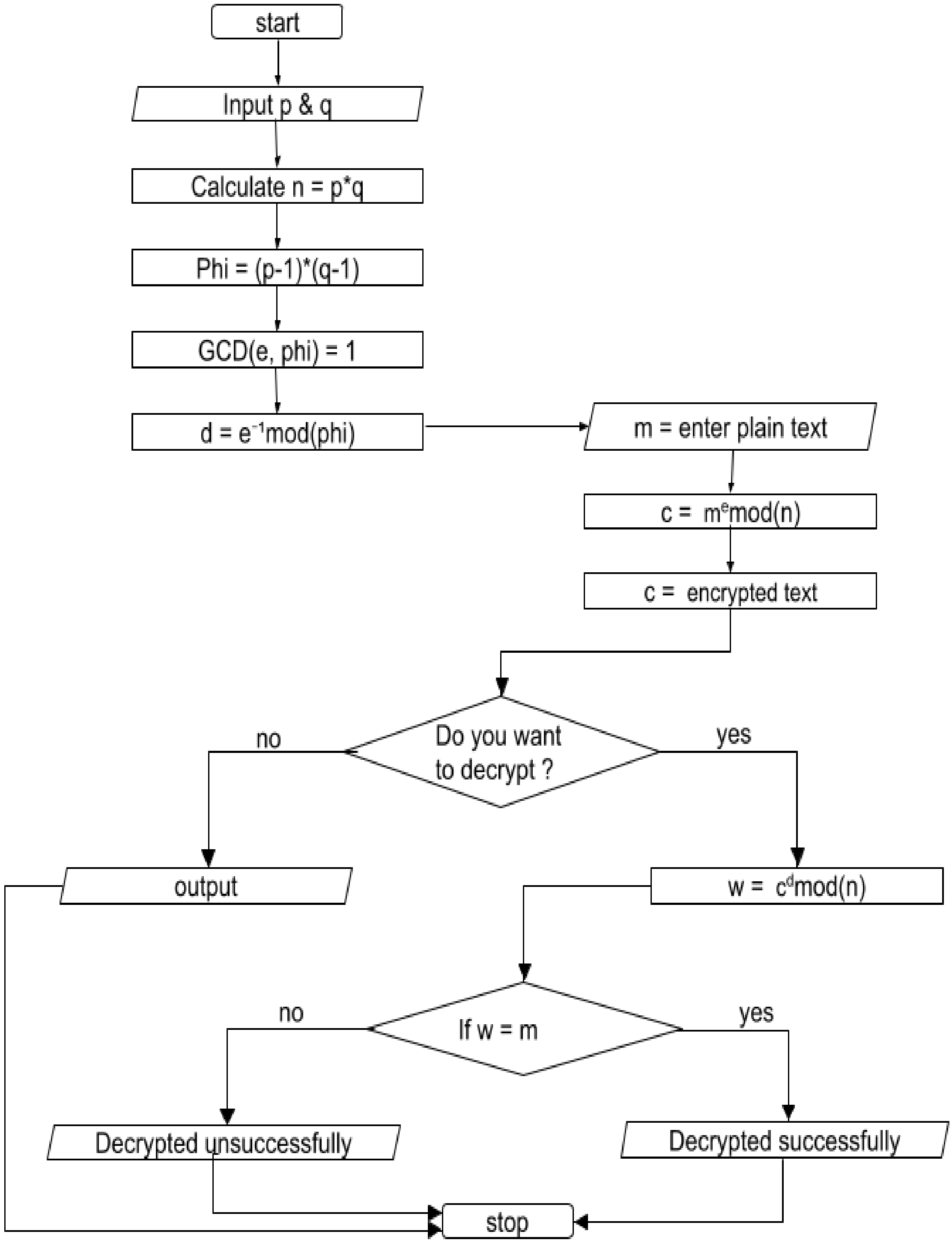
## INFERENCE:

1. Channel response of transmitter antenna is independent of each other.
2. The fact that $H^H$. Y is a diagonal matrix ensured that there is no cross talk between x1, x2.
3. With Alamouti STBC, we are transmitting from two antennas. Hence the total power transmitted is higher than any other schemes.
4. The signal to the noise ratio is getting reduced while using STBC coding.

## RESULT:

Thus, the Space Time Block Code using Alamouti coding scheme is studied and the Bit Error Rate is analysed.

**Flow chart:**


start
Input p & q
Calculate n = p*q
Phi = (p-1)*(q-1)
GCD(e, phi) = 1
d = e⁻¹mod(phi)
m = enter plain text
c = m^e mod(n)
c = encrypted text
Do you want to decrypt ?
no
yes
output
w = c^d mod(n)
If w = m
no
yes
Decrypted unsuccessfully
Decrypted successfully
stop

| EX:NO:7<br><br>23.09.2021 | **Network security protocols** |
|---|---|

## Aim:

To write a code using MATLAB to encrypt and decrypt a message using the RSA algorithm.

## Software required:

MATLAB R2019a

## Theory:

The RSA algorithm is an asymmetric cryptography algorithm where it works on two different keys i.e. public key and private key. As the name describes, the public key is given to everyone and the private key is kept private.

An example of asymmetric cryptography:

1. A client (for example a browser) sends its public key to the server and requests for some data.
2. The server encrypts the data using the client's public key and sends the encrypted data.
3. Client receives this data and decrypts it.

Since this is asymmetric, nobody except brower can decrypt the data even if a third party has the public key of the browser.

The idea of RSA is based on the fact that it is difficult to factorize a large integer. The public key consists of two numbers where one number is a multiplication of two large prime numbers. The private key is also derived from the same two prime numbers. So if somebody can factorize the large number, the private key is compromised. Therefore encryption strength totally lies on the key size and if we double or triple the key size, the strength of encryption increases exponentially. RSA keys can be typically 1024 or 2048 bits long, but experts believe that 1024 bit keys could be broken in the near future. But still now it seems to be an infeasible task.

## Algorithm:

- Choose two random large prime numbers, p and q
- Compute the product n = p*q
- Randomly choose the encryption key, e such that e and (p-1)*(q-1) are relatively prime
- Use the extended Euclidean algorithm to compute the decryption key, d such that

$e*d \equiv 1 \bmod ( (p-1)*(q-1) )$   i.e. $d = e^{-1} \bmod ( (p-1)*(q-1) )$

d and n are also relatively prime

- Encryption
  - Divide message into numerical block smaller than n
  - For each block $c = m^e \bmod (n)$
- Decryption
  - For each cipher text block $m = c^d \bmod (n)$

**Program:**

```
Clc;
clear all;
disp('RSA algorithm');
p=input('Enter the prime no. for p: ');
q=input('Enter the prime no. for q: ');
n=p*q;
fprintf('\nn=%d',n);
phi=(p-1)*(q-1);
fprintf('\nphi(%d) is %d',n,phi);
val=0;
cd=0;
while(cd~=1||val==0)
n1=randi(n,1,1);
e=randi([2 n1],1,1);
val=isprime(e);
cd=gcd(e,phi);
end
val1=0;
d=0;
while(val1~=1);
d=d+1;
val1=mod(d*e,phi);
end
```

```
fprintf('\nd=%d',d);
fprintf('\nPublic key is (%d,%d)',e,n);
fprintf('\nPrivate key is (%d,%d)',d,n);
m=input('\nEnter the message: ','s');
m1=m-0;
disp('ASCII equivalent of message ');
disp(m1);
over=length(m1);
 o=1;
while(o<=over);
        m=m1(o);
        diff=0;
        if(m>n);
        diff=m-n+1;
        end
        m=m-diff;
qm=dec2bin(e);
len=length(qm);
c=1;
xz=1;
while(xz<=len)
        if(qm(xz)=='1')
    c=mod(mod((c^2),n)*m,n);
        elseif(qm(xz)=='0')
        c=(mod(c^2,n));
        end
        xz=xz+1;
end
c1(o)=c;
qm1=dec2bin(d);
```

## Output :

RSA algorithm

Enter the prime no. for p: 5

Enter the prime no. for q: 7

n=35

phi(35) is 24

d=5

Public key is (5,35)

Private key is (5,35)

Enter the message: hello

ASCII equivalent of message

104 101 108 108 111

The encrypted message is

34 34 34 34 34

The decrypted message in ASCII is

104 101 108 108 111

The decrypted message is: hello

```
len1=length(qm1);
nm=1;
xy=1;
while(xy<=len1)
        if(qm1(xy)=='1')
        nm=mod(mod((nm^2),n)*c,n);
        elseif(qm1(xy)=='0')
        nm=(mod(nm^2,n));
        end
        xy=xy+1;
end
nm=nm+diff;
nm1(o)=char(nm);
o=o+1;

end
o=1;
fprintf('\nThe encrypted message is \n');
while(o<=over)
  fprintf('\t%d',c1(o));
  o=o+1;
end
o=1;
fprintf('\nThe decrypted message in ASCII is \n');
while(o<=over)
fprintf('\t%d',nm1(o));
o=o+1;
end
fprintf('\nThe decrypted message is: ');
disp(nm1);
```

```
fprintf('\n');
```

## Inference:

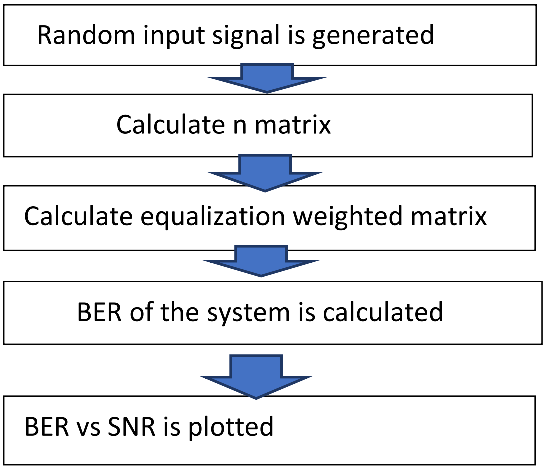
RSA is faster at encrypting and verifying, which enables public key encryption and is widely used to secure sensitive data, particularly when it is being sent over an insecure network such as the internet.

## Result:

Hence the message is encrypted and decrypted using the RSA algorithm which is implemented using MATLAB.

# FLOW CHART:

# ZERO FORCING:



| EX:NO:8 25.09.2021 | PERFORMANCE OF EQUALIZER |
|---|---|

## AIM:

To implement equalizers using ZF &LMS algorithm.

## SOFTWARE USED:

MATLAB R2020a

## THEORY:

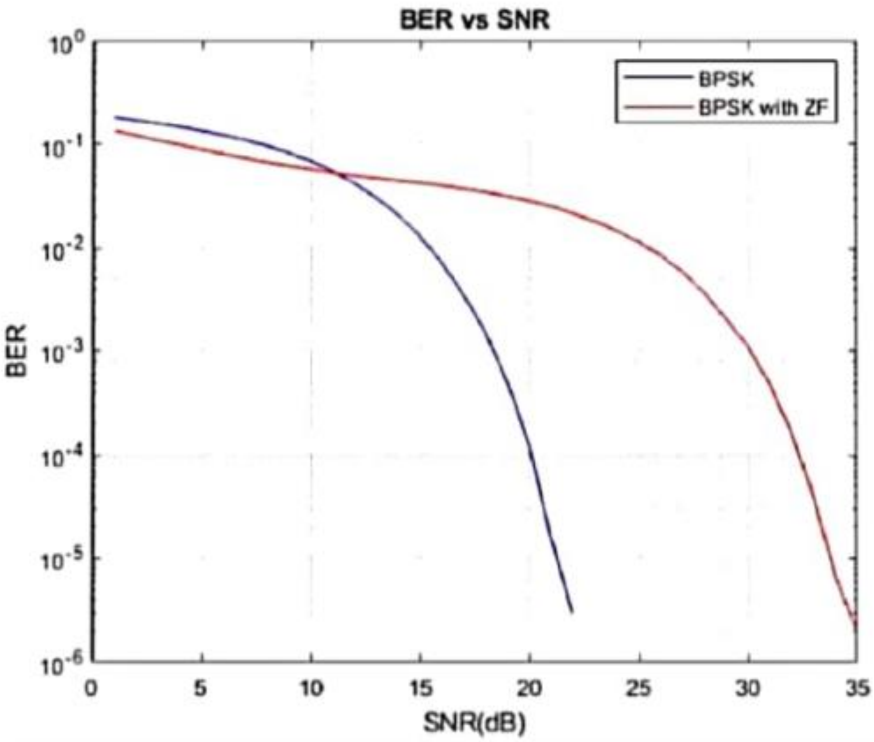
1. **ZERO FORCING:**

   It's a linear equalizer which uses universe impulse response to compensate channel effect. An overall impulse response is equal to one for defected symbol and zero for all other received symbols. The zero-forcing equalizer nullified the circumference but does not consider effect of noise. The noise may be increased in this process.

## MATLAB CODE:

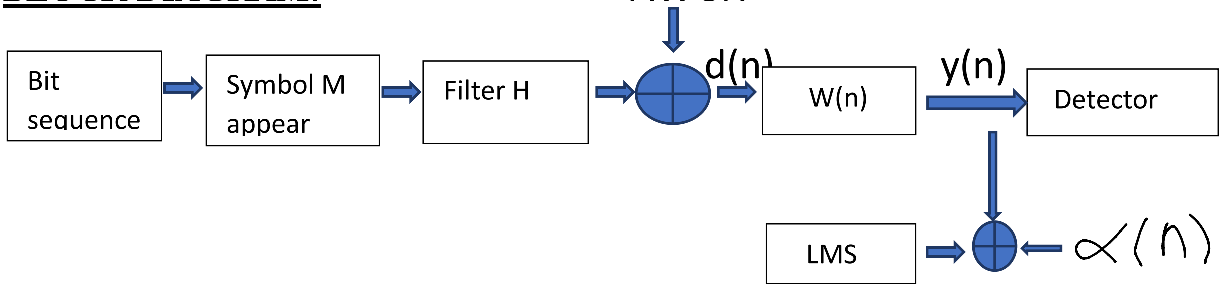
```
clc;
clear all;
close all;
n_bits = 1024*1000;
x_in = randi([0,1],n_bits,1);
x_mod=pskmod(x_in,2);
L=2;
h1=1;
h2=0.7;
r=3;
x=x_mod';
x_1=circshift(x,1);
x_1(1)=0;
x1=circshift(x,-1);
x1(end)=0;
x2=circshift(x,-2);
x2(end-1:end)=0;
```

## OUTPUT:


BER vs SNR
BPSK
BPSK with ZF
BER
SNR(dB)
$10^0$
$10^{-1}$
$10^{-2}$
$10^{-3}$
$10^{-4}$
$10^{-5}$
$10^{-6}$
0
5
10
15
20
25
30
35

```
X=[x2 x1 x x_1];
H=zeros(r,r+L-1);
for i=1:r
   H(i,i:i+L-1)=[h1 h2];
end
c=(H*H.')\(H*[0;0;1;0]);
ber1=[];
ber2=[];
for snr=1:40
   y=awgn([h1 h2]*[x;x_1],snr,'measured');
   noise = y-([h1 h2]*[x x_1]);
   noise1= circshift(noise,-1);
   noise1(end)=0;
   noise2= circshift(noise,-2);
   noise1(end-1,end)=0;
   Y=(H*[x2;x1;x;x_1])+[noise2 noise1 noise];
   X_prime = c.'*Y;
   x_demod = pskmod(y.',2);
   X_demod= pskdemod(X_prime.',2);
   [~,error1]=biterr(x_in,x_demod);
   [~,error2]=biterr(x_in,X_demod);
   ber1=[ber1,error1];
   ber2=[ber2,error2];
end
semilogy(1:40,ber1,'-b',1:40,ber2,'-r');
grid on;
legend({'BPSK Without ZF','BPSK with ZF'});
title('SNR vs BER');
xlabel('SNR');
ylabel('BER');
```

## BLOCK DIAGRAM:




## LEAST MEAN SQUARE:

Generate bit sequence and perform symbol mapping and get d(n)

↓

Pass bit sequence through filter with coefficients b=e() and add awgn noise with SNR=10 and get x(n)

↓

For adaptive filter design, initialize filter coefficients to zero and design value to filter order and u

↓

Y(n)=Wn^T*x(n)

↓

e(n)=d(n)-y(n)

↓

Update filter coefficient

↓

Perform steps recursively for entire length of bit sequence

↓

Plot the graph for iteration and mean square filter coefficient

## PROCEDURE:

1. Random input signal is generated.
2. Calculate H matrix.
3. Calculate equalization weighted matrix.
4. BER of the system is calculated.
5. BER vs SNR is plotted.

## INFERENCE:

The zero forcing equalizer nullifies signal interference but performance of CF equalizer degraded of noise enhancement.

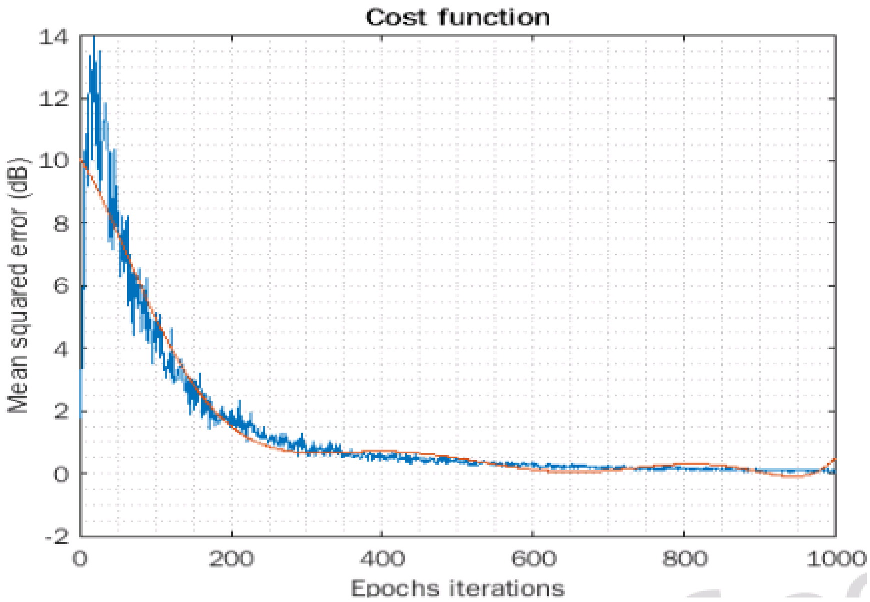
## 2. LEAST MEAN SQUARE:

LMS algorithm is a type of filter used in machine learning. It uses a technique called " Method of steepest descent" and continuously estimates result by updating filter weights. Through the principles of algorithm convergence, The LMS algorithm provides particular learning curves useful in machine learning theory and implementation. Many of these ideas are part of dedicated work on refining machine learning models, matching i/p s to o/p s, making training and test processors more effective , and generally pursuing "convergence" where the iterative learning process resolves into a final result instead of getting off track.

## MATLAB CODE:

```
clc;
clear all;
close all;
h = [0.9 0.3 0.5 -0.1];
SNRr = 30;
runs = 100;
eta = 5e-3;
order=12;
for run = 1 : runs
U = zeros(1,order);
W = randn(1,order);
N = 1000;
Bits = 2;
data = randi([0 1],1,N);
```

## OUTPUT:



Cost function
Mean squared error (dB)
14
12
10
8
6
4
2
0
-2
0
200
400
600
800
1000
Epochs iterations

```
d = real(pskmod(data,Bits));
r = filter(h,1,d);
x = awgn(r, SNRr);
for n = 1 : N
U(1,2:end) = U(1,1:end-1);
U(1,1) = x(n);
y = (W)*U';
e = d(n) - y;
W = W +
eta * e * U ;
J(run,n) = e * e';
end
end
MJ = mean(J,1);
figure
plot((MJ))
trendMJ = polyval(polyfit((0:N),[0 (MJ)],7),(1:N));
hold on
plot(trendMJ)
grid minor
xlabel('Epochs iterations');
ylabel('Mean squared error (dB)');
title('Cost function');
```

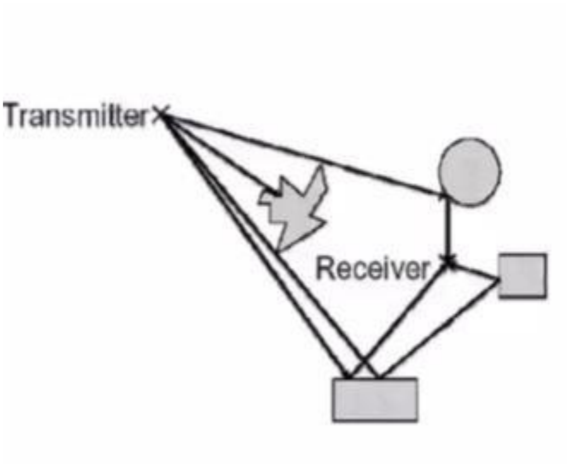
**INFERENCE:**

Thus in Least square algorithm when iteration increases the error decreases and therefore the mean square error(MSE) decreases.

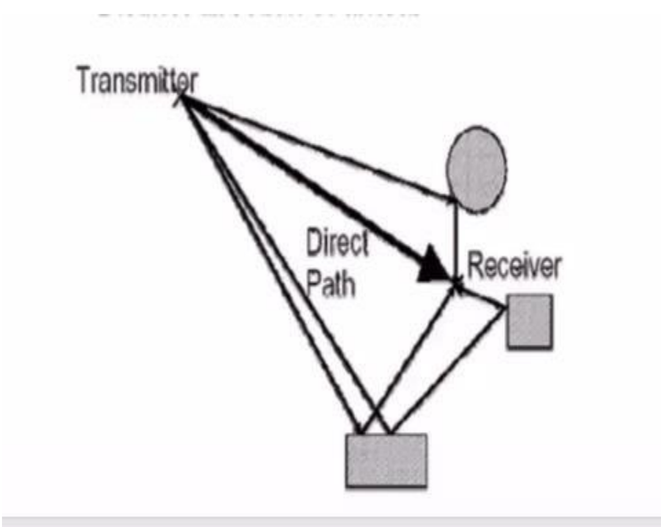
**RESULT:**

Thus channel Equalization using least mean square algorithm is carried out successfully and spectrum is spotted. MATLAB code for communication method with zero forcing equalizer was written and performance is analysed.

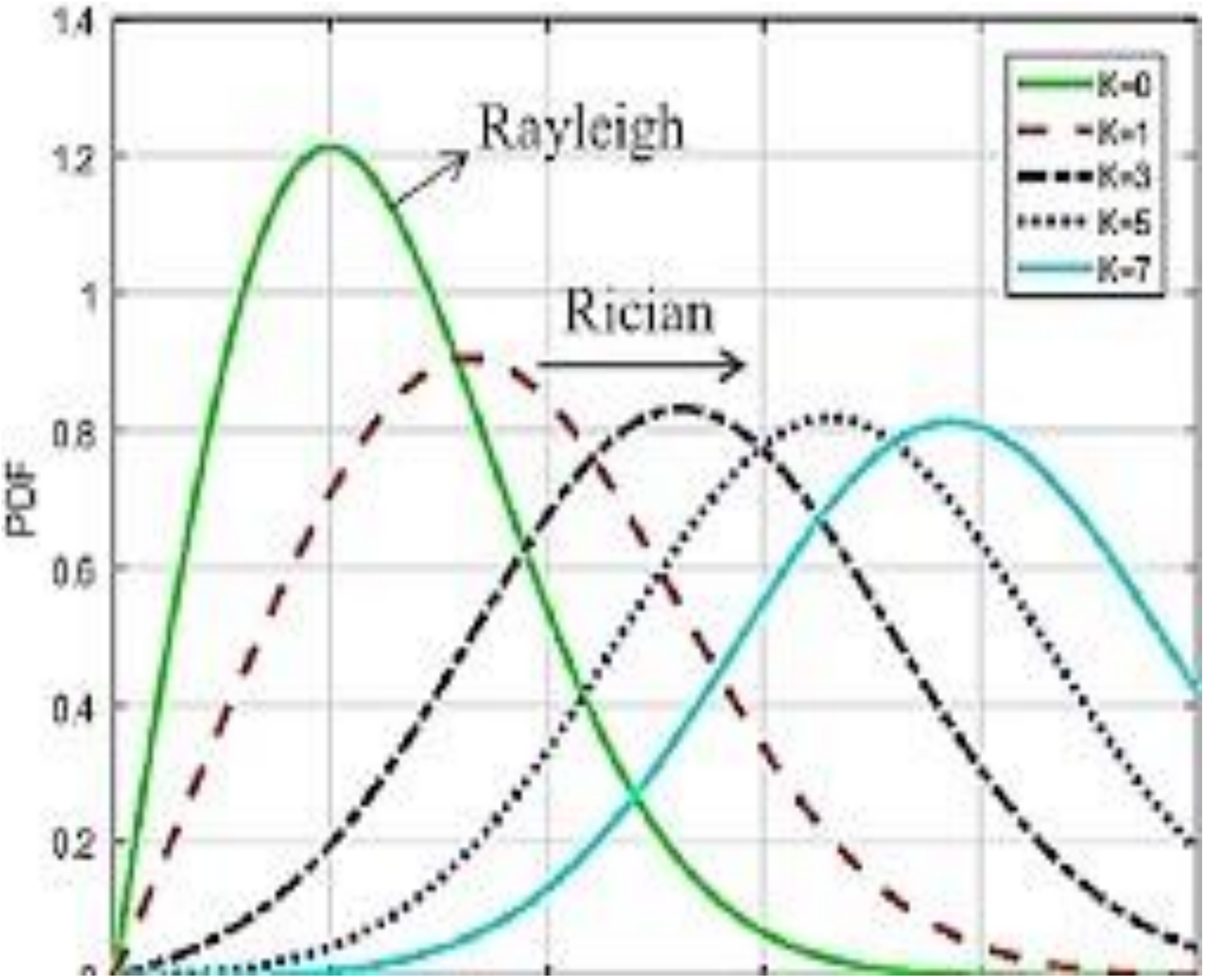
**RAYLEIGH DISTRIBUTION:**




**RICIAN DISTRIBUTION:**




**POWER DENSITY FUNCTION:**



| EX:NO:9<br>27.09.2021 | **CHARACTERISTICS OF WIRELESS FADING CHANNEL** |
|---|---|

# RAYLEIGH AND RICIAN FADING CHANNELS

**AIM:**

To analyze the BER performance of Rayleigh and Rician fading channels.

**SOFTWARE REQUIRED:**

MATLAB R2019b

**THEORY:**

**RAYLEIGH DISTRIBUTION:**

- Describes the received signal envelope distribution for channels where all the components are non-LOS.
- NO LOS (Line Of Sight) COMPONENT.
- The Rayleigh distribution is commonly used to describe the statistical time varying nature of the received envelope of a flat fading signal, or the envelope of an individual multipath component
- The envelope of sum of two quadrature gaussian noise signals obey a Rayleigh distribution.

$$p(r) = \begin{cases} \frac{r}{\sigma^2}\exp\left(-\frac{r^2}{2\sigma^2}\right) & 0 \le r \le \infty \\ 0 & r < 0 \end{cases}$$

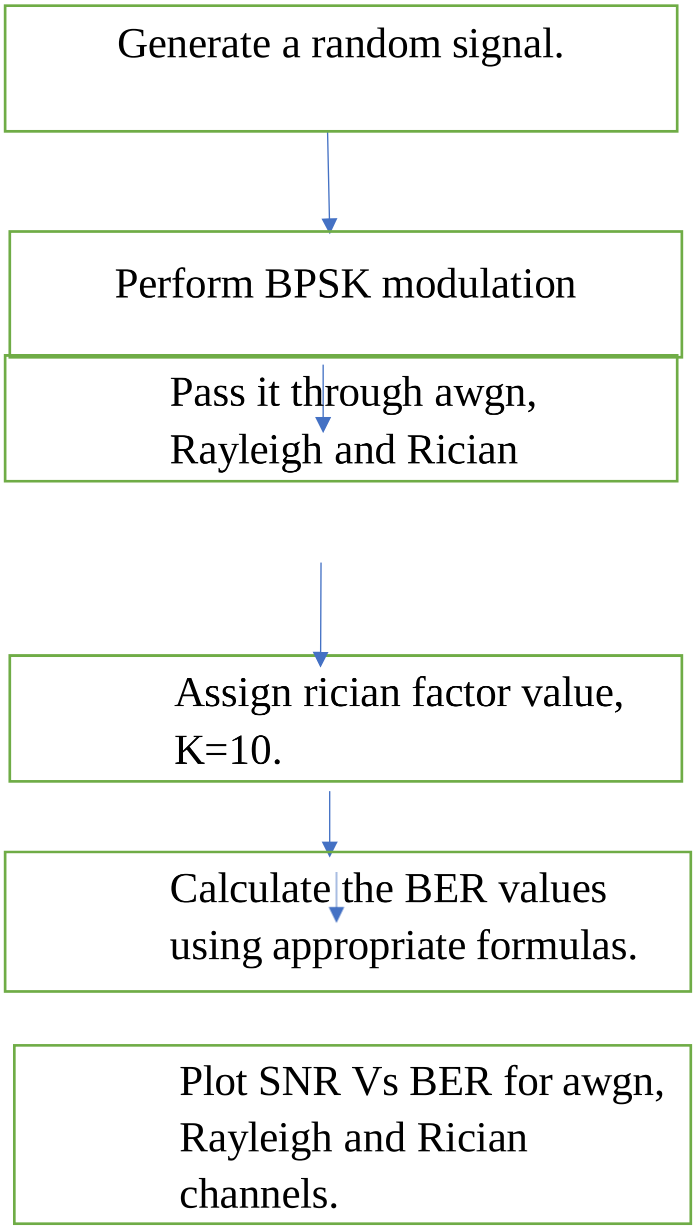
- **σ** is the rms value of the received voltage before envelope detection, and **$\sigma^2$** is the time-average power of the received signal before envelope detection.

**RICIAN DISTRIBUTION:**

- Describes the received signal envelope distribution for channels where one of the multipath components is LOS component.

- When there is a dominant stationary signal component present, the small-scale fading envelope distribution is Rician. The effect of a dominant signal arriving with many weaker multipath signals gives rise to the Rician distribution.

**FLOWCHART:**


Generate a random signal.
Perform BPSK modulation
Pass it through awgn, Rayleigh and Rician
Assign rician factor value, K=10.
Calculate the BER values using appropriate formulas.
Plot SNR Vs BER for awgn, Rayleigh and Rician channels.

- The Rician distribution degenerates to a Rayleigh distribution when the dominant component fades away.

$$p(r) = \begin{cases} \frac{r}{\sigma^2} \exp[-\frac{(r^2 + A^2)}{2\sigma^2}] I_0(\frac{Ar}{\sigma^2}) & 0 \le r \le \infty, \quad A \ge 0 \\ 0 & r < 0 \end{cases}$$

- The Rician distribution is often described in terms of a parameter K which is defined as the ratio between the deterministic signal power and the variance of the multipath.

$$K = \frac{A^2}{2\sigma^2}$$

- K is known as the Rician factor.

**PROCEDURE:**

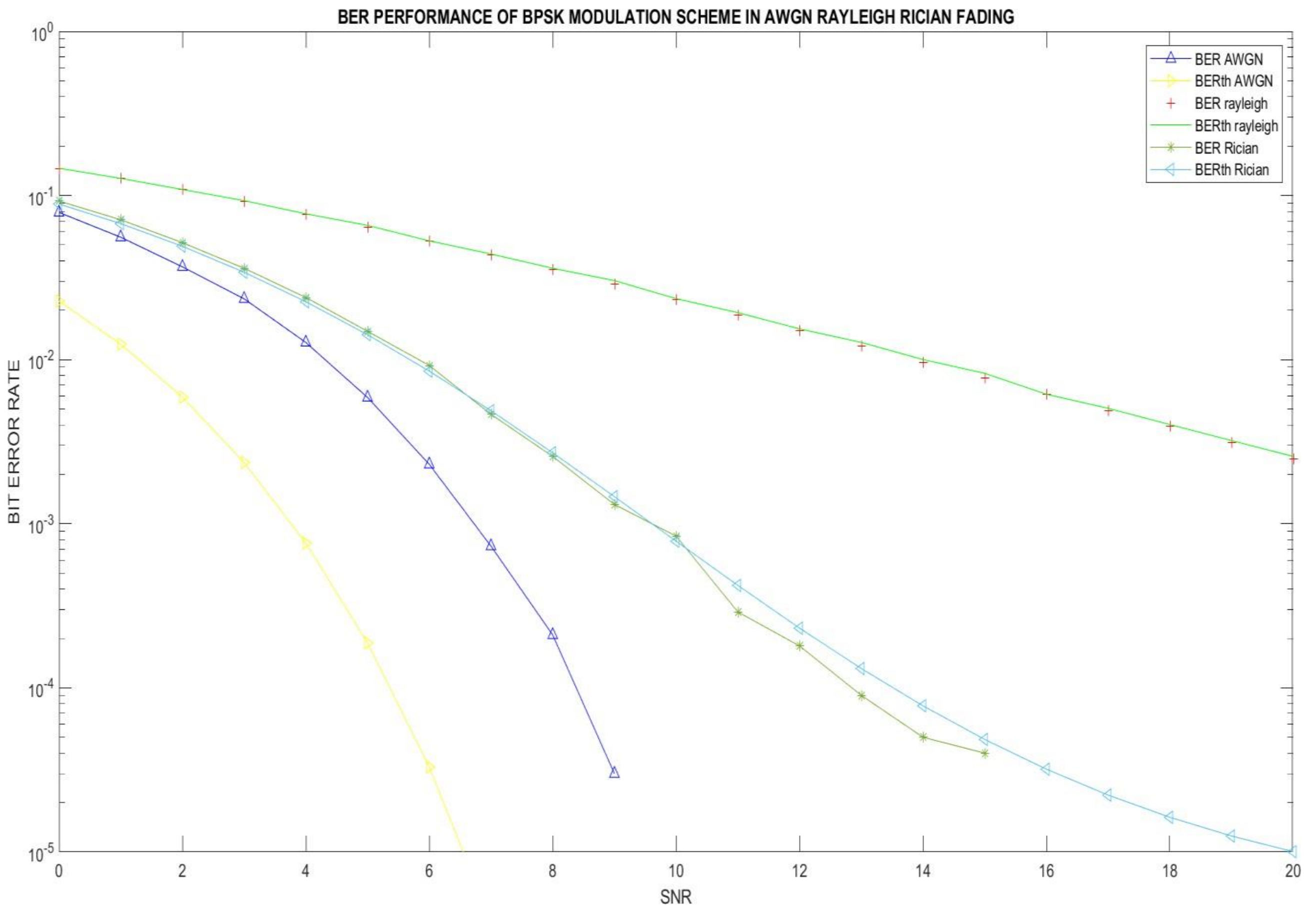
- Generate a random signal.
- Perform BPSK modulation.
- Pass it through awgn, Rayleigh and Rician channels.
- Calculate the BER values using appropriate formulas.
- Plot SNR Vs BER [ theoretical and practical] for awgn, Rayleigh and Rician channels.

**CODE:**

```
clc
clear all;
close all;
n=100000;
i=randi([0,1],1,n);
i1=2*i-1;
a=randn(1,n);
b=randn(1,n);
rc=1/sqrt(2)*(sqrt(a.^2+b.^2));
for l=0:1:20
snr=10^((l/10));
sdev=sqrt(0.5/snr);
N=random('norm',0,sdev,[1,n]);
yrc=rc.*i1+N;
yawgn=i1+N;
Yb=(yawgn>=0);
YR=(yrc>=0);
ErrorR=sum((xor(YR,i)));
ErrorA=sum((xor(Yb,i)));
ber_A(l+1)=ErrorA/n;
```

```
ber_R(l+1)=ErrorR/n;
berthR(l+1)=0.5*(1-sqrt(snr/(snr+1)));
p=((1-2*ber_R(l+1))^2)/(4*(ber_R(l+1)-(ber_R(l+1)^2)));
outage(l+1)=1-exp(-3.16/p);
outageT(l+1)=1-exp(-3.16/snr);

berthA(l+1)=0.5*erfc(sqrt(2*snr));
end

figure

q=0:1:20 ;
semilogy(q,ber_A(q+1),'-^b');
hold on
semilogy(q,berthA(q+1),'->y');
hold on
semilogy(q,berthR(q+1),'+r');
hold on
semilogy(q,ber_R(q+1),'g-');
axis([0 20 10^-5 1]);

xlabel('SNR')
ylabel('BIT ERROR RATE')

xlabel('SNR')
ylabel('OUTAGE PROBABILITY')

k1=10;
mean=sqrt(k1/(k1+1));
sigma=sqrt(1/(2*(k1+1)));
Nr2=randn(1,length(i1))*sigma+mean;
Ni2=randn(1,length(i1))*sigma;

No3=sqrt(Nr2.^2+Ni2.^2);
for k=0:1:20
    snrl=10^(k/10);
    Np=1/snrl;
    sd=sqrt(Np/2);
    No=random('Normal',0,sd,1,length(i1));
    t1=i1.*No3+No;
    z1=t1./No3;
 op1=(z1>0);
    Berr(k+1)=sum(xor(op1,i))/n;
    BerTr(k+1)=.5*erfc(sqrt(k1*snrl/(k1+snrl)));
    end;

    k=0:1:20;
    semilogy(k,Berr(k+1),'-*');
    hold on;
    semilogy(k,BerTr(k+1),'-<');
```

# OUTPUT:


BER PERFORMANCE OF BPSK MODULATION SCHEME IN AWGN RAYLEIGH RICIAN FADING
BIT ERROR RATE
SNR
BER AWGN
BERth AWGN
BER rayleigh
BERth rayleigh
BER Rician
BERth Rician

```
title('BER PERFORMANCE OF BPSK MODULATION SCHEME IN AWGN
RAYLEIGH RICIAN FADING')
xlabel('SNR')
ylabel('BIT ERROR RATE')
hleg=legend('BER AWGN','BERth AWGN','BER rayleigh','BERth
rayleigh','BER Rician','BERth Rician')
```

**INFERENCE:**

If the channel is assumed to be ideal, less SNR is sufficient to achieve less BER, but if the channel is assumed to be practical, having Rayleigh or Rician distribution, a higher SNR is required to achieve same BER as that of the ideal channel.

**RESULT:**

Thus, the SNR Vs BER characteristics of Rayleigh and Rician fading channels are plotted and analyzed using MATLAB software.

**FLOWCHART:**

Generate two tone signal

↓

Find N-Point DFT

↓

**1.Slow Flat**

- Generate complex gaussian random variable h(t).
- Y(t)=h(t)*x(t).
- Find the FFT & plot the spectrum.

↓

**2.Fast Flat**

- Generate complex gaussian random variable h(t).
- Pass h(t) through LPF.
- Y(t)=h(t)*x(t).
- Find the FFT & plot the spectrum.

↓

**3.Slow Frequency Selective**

- Generate two complex gaussian random variable α1, α2.
- Y1(t)= α1.x(t), Y2(t)= α2.x(t-τ).
- Y(t)=Y1(t)*Y2(t).
- Find the FFT & plot the spectrum.

↓

**4.Fast Frequency Selective**

- Generate a complex Gaussian RV α1,α2 equal to length of x(t).
- Pass α1,α2 through LPF with fc=fd.
- Y1(t)=α1(t)*x(t)
- Y2(t)=α2(t)*x(t- τ)
- Y(t)=Y1(τ)*Y2(t).
- Find FFT & Plot Spectrum.

| EX:NO:10<br><br>29.09.2021 | # TYPES OF FADING CHANNEL AND ITS CHARACTERISTICS |
|---|---|

**AIM:**

To plot the spectra of a two-tone signal due to slow flat fading, slow and frequency selective fading, fast flat fading and flat frequency selective fading.

**SOFTWARE REQUIRED:**

MATLAB 2019a

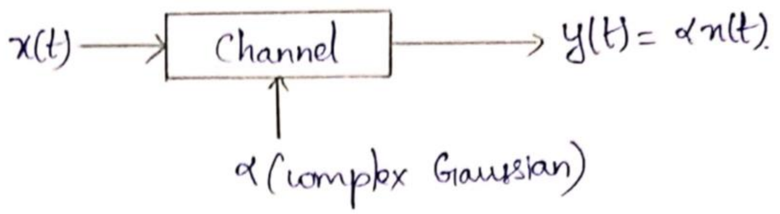
**THEORY:**

In wireless communications, **fading** is variation of the attenuation of a signal with various variables. These variables include time, geographical position, and radio frequency. Fading is often modelled as a random process. A **fading channel** is a communication channel that experiences fading. In wireless systems, fading may either be due to multipath propagation, referred to as multipath-induced fading, weather (particularly rain), or shadowing from obstacles affecting the wave propagation, sometimes referred to as **shadow fading**.

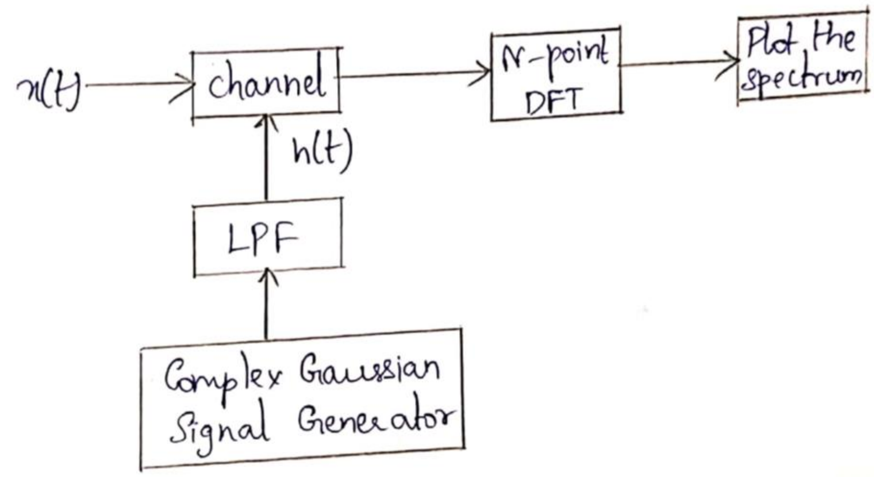
- **Slow fading** arises when the coherence time of the channel is large relative to the delay requirement of the application.[2] In this regime, the amplitude and phase change imposed by the channel can be considered roughly constant over the period of use.
- **Fast fading** occurs when the coherence time of the channel is small relative to the delay requirement of the application. In this case, the amplitude and phase change imposed by the channel varies considerably over the period of use.
- In **flat fading**, the coherence bandwidth of the channel is larger than the bandwidth of the signal. Therefore, all frequency components of the signal will experience the same magnitude of fading.
- In **frequency-selective fading**, the coherence bandwidth of the channel is smaller than the bandwidth of the signal. Different frequency components of the signal therefore experiences uncorrelated fading.

Block diagram

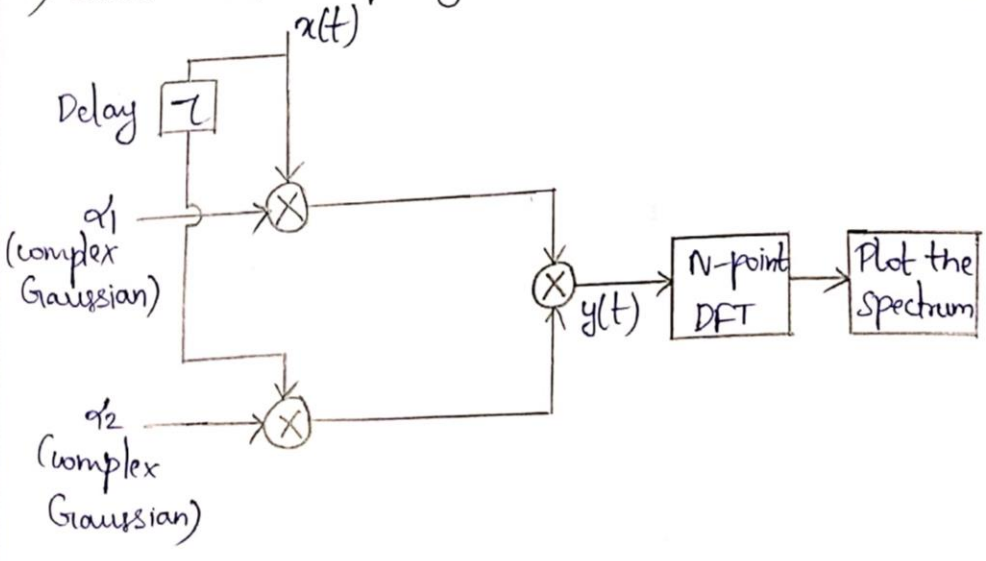
1) Slow and Flat fading:




2) Fast and flat Fading:




3) Slow and Frequency selective Fading:



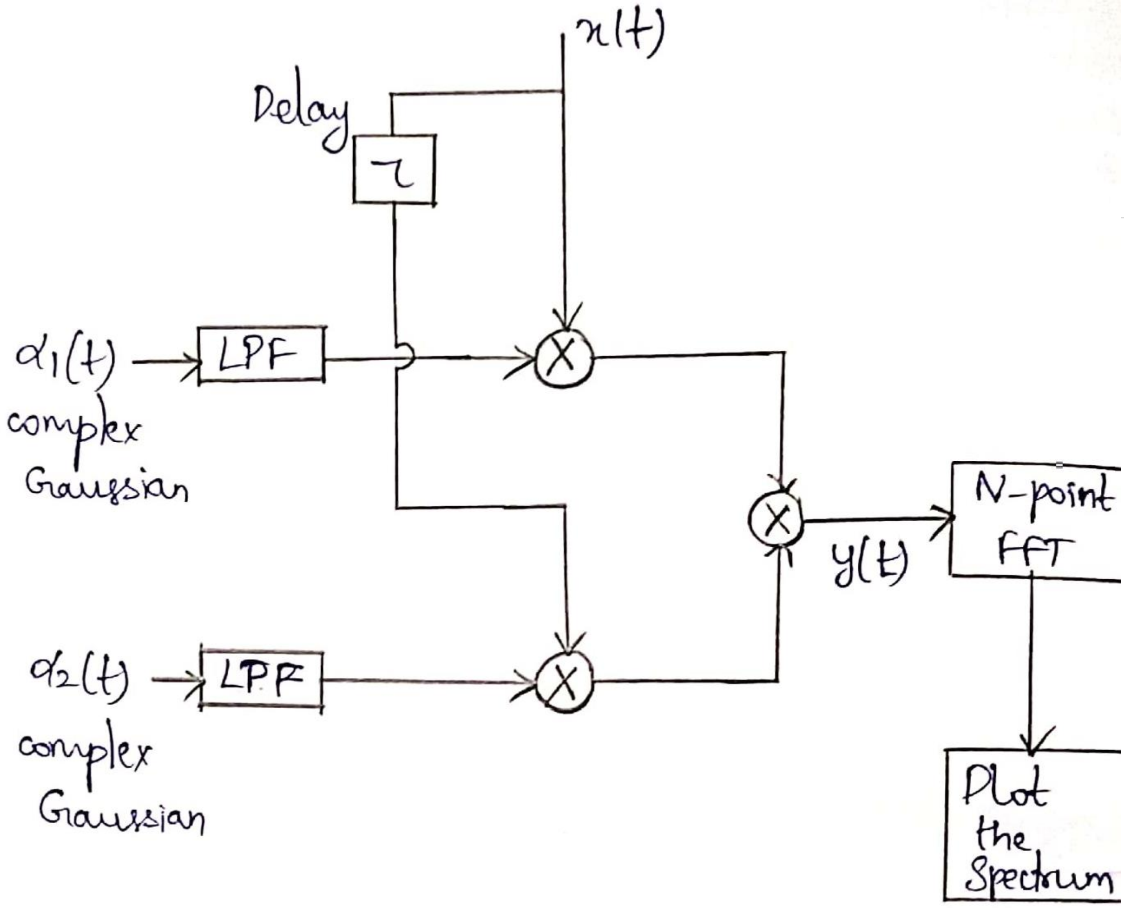
**ALGORITHM:**

- Generate a two tone signal.
- Find the N-point DFT of the signal.
- For the slow & fast fading generate the random complex gaussian variable and multiply with the signal.
- Find the FFT of the signal and plot.
- For the Fast & Flat fading generate a gaussian random variable and pass it through LPF & multiply with x(t).
- Find y1(t)+y2(t) and plot.

**CODE:**

```
clc;
clear all;
close all;
fs=1000;                    %Sampling frequency
f1=100;
f2=400;
t= 0:1/fs:1-1/fs;          %Defining time interval using
fs
x1=cos(2*pi*f1*t)+cos(2*pi*f2*t);
figure()
subplot(2,1,1)
plot(t,x1);
xlabel("Time --->");
ylabel("Amplitude --->");
title("Generation of dual tone signal");
x1_f=fft(x1);
f = fs/2*linspace(-1,1,fs); %calculate the frequency
axis, which is defined by the sampling rate
subplot(2,1,2)
plot(f,abs(x1_f));
xlabel("Frequency --->");
ylabel("Amplitude --->");
title("Spectrum of the dual tone signal");
%generation of delayed signal
n=77;
for i=1:length(x1)
if i<=n
    x2(i)=0;
end
if (i>n)
        x2(i)=x1(i-n);
```

4) Fast and Frequency selective Fading:



```
end
end
figure()
subplot(2,1,1)
plot(t,x2);
xlabel("Time --->");ylabel("Amplitude ---
>");title("Generation of dual tone delayed signal");
x2_f=fft(x2);
f = fs/2*linspace(-1,1,fs); %calculate the frequency
axis, which is defined by the sampling rate
subplot(2,1,2)
plot(f,abs(x2_f));
xlabel("Frequency --->");ylabel("Amplitude ---
>");title("Spectrum of the dual tone delayed signal");

%%slow flat fading
h1=randn+1i*(randn);
sff=h1.*x1;
sff_f=fft(sff);
figure()
subplot(2,1,1)
plot(f,abs(x1_f));
xlabel("Frequency --->");ylabel("Amplitude ---
>");title("Spectrum of input");
subplot(2,1,2)
plot(f,abs(sff_f));
xlabel("Frequency --->");ylabel("Amplitude ---
>");title("Spectrum of slow flat fading");

%%slow and frequency selective
h2=randn+1i*randn;
sfs=(h1.*x1)+(h2.*x2);
sfs_f=fft(sfs);
figure()
subplot(2,1,1)
plot(f,abs(x1_f));
xlabel("Frequency --->");ylabel("Amplitude ---
>");title("Spectrum of input");
subplot(2,1,2)
plot(f,abs(sfs_f));
xlabel("Frequency --->");ylabel("Amplitude ---
>");title("Spectrum of slow frequency selective fading");
% fast and flat fading
fc=100;
n1=(randi([0 1],1,length(x1))+1i*randi([0
1],1,length(x1)));
figure()
```

# OUTPUT:

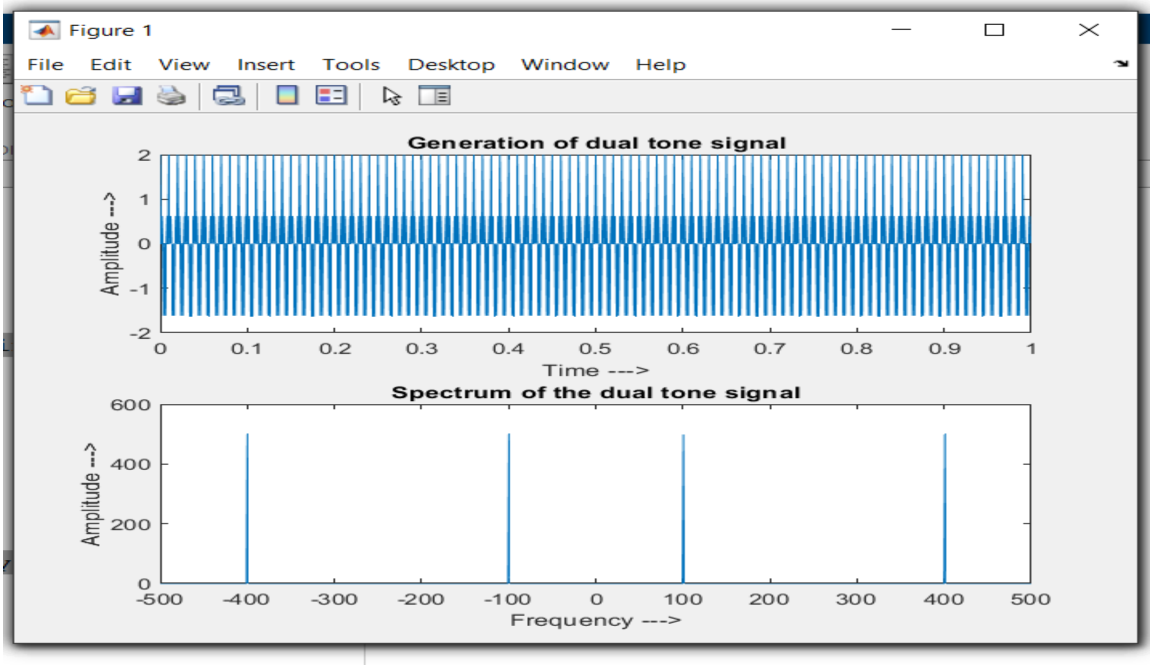


Figure 1
File Edit View Insert Tools Desktop Window Help
Generation of dual tone signal
Amplitude --->
Time --->
Spectrum of the dual tone signal
Amplitude --->
Frequency --->

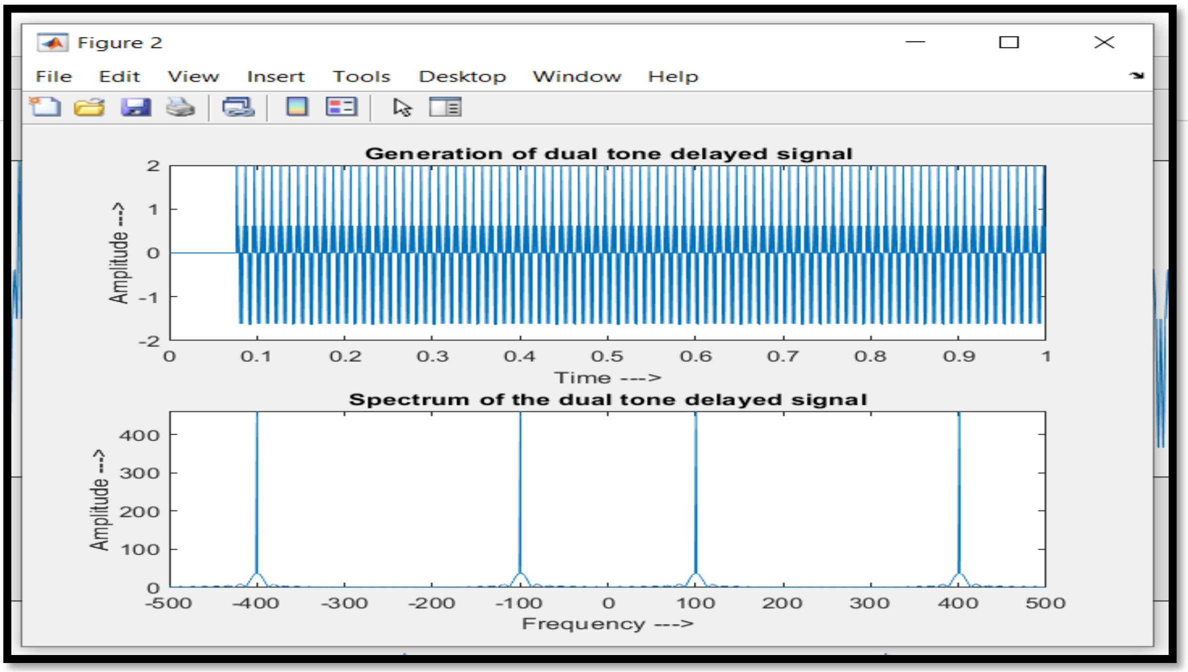
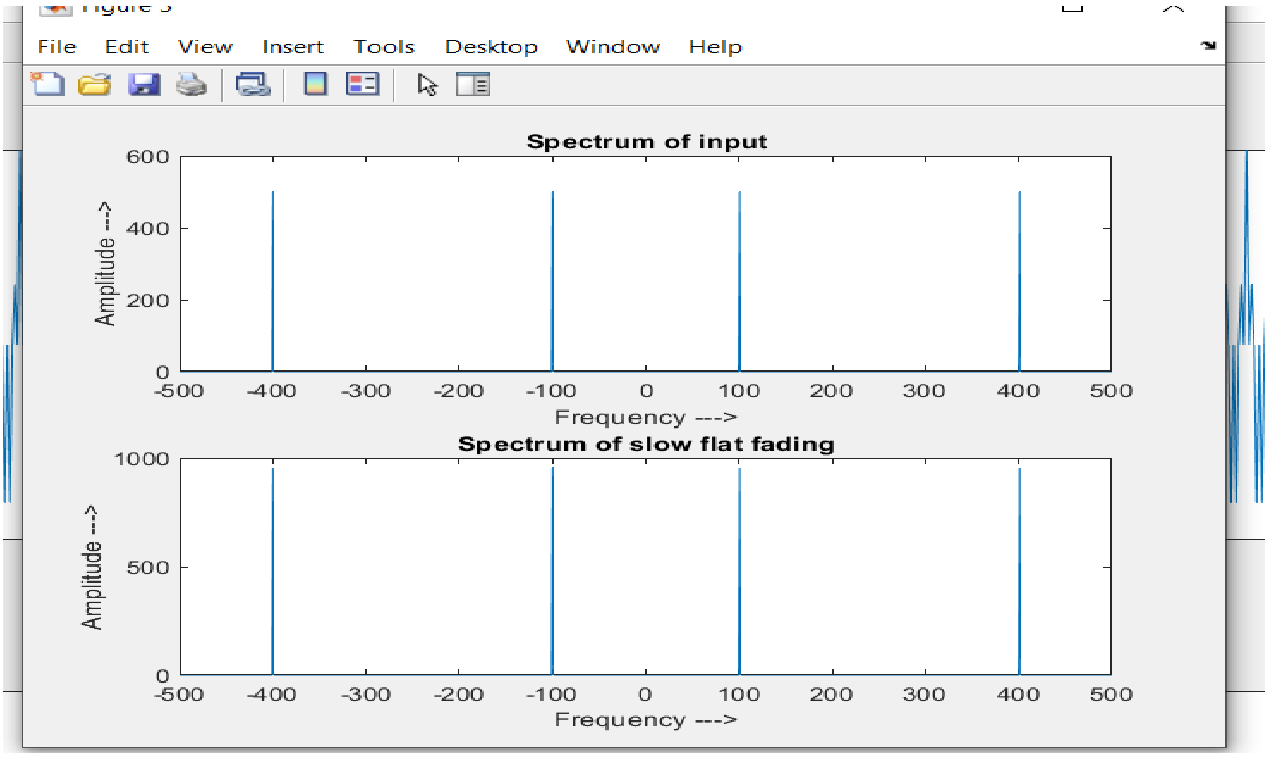



Figure 2
File Edit View Insert Tools Desktop Window Help
Generation of dual tone delayed signal
Amplitude --->
Time --->
Spectrum of the dual tone delayed signal
Amplitude --->
Frequency --->

```
subplot(2,1,1)

plot(t,n1)
xlabel("time--->");ylabel("Amplitude ---
>");title("Gaussian noise");
[b,a] = butter(6,fc/(fs/2)); % Butterworth filter of
order 6
fx1 = filter(b,a,n1); % Will be the filtered signal
ff1=x1.*fx1;
ff_f=fft(ff1);
subplot(2,1,2)
plot(f,abs(ff_f));
xlabel("Frequency --->");ylabel("Amplitude ---
>");title("Spectrum of fast flat fading");

%%fast frequency selective fading
n2=(randi([0 1],1,length(x1))+1i*randi([0
1],1,length(x1)));
[b,a] = butter(6,fc/(fs/2)); % Butterworth filter of
order 6
fx2 = filter(b,a,n2); % Will be the filtered signal
ff2=x2.*fx2;
ffs=ff1+ff2;
ffs_f=fft(ffs);
figure()
plot(abs(ffs_f));
xlabel("Frequency --->");ylabel("Amplitude ---
>");title("Spectrum of fast frequency selective fading");
```



File Edit View Insert Tools Desktop Window Help
Spectrum of input
Amplitude --->
Frequency --->
Spectrum of slow flat fading
Amplitude --->
Frequency --->

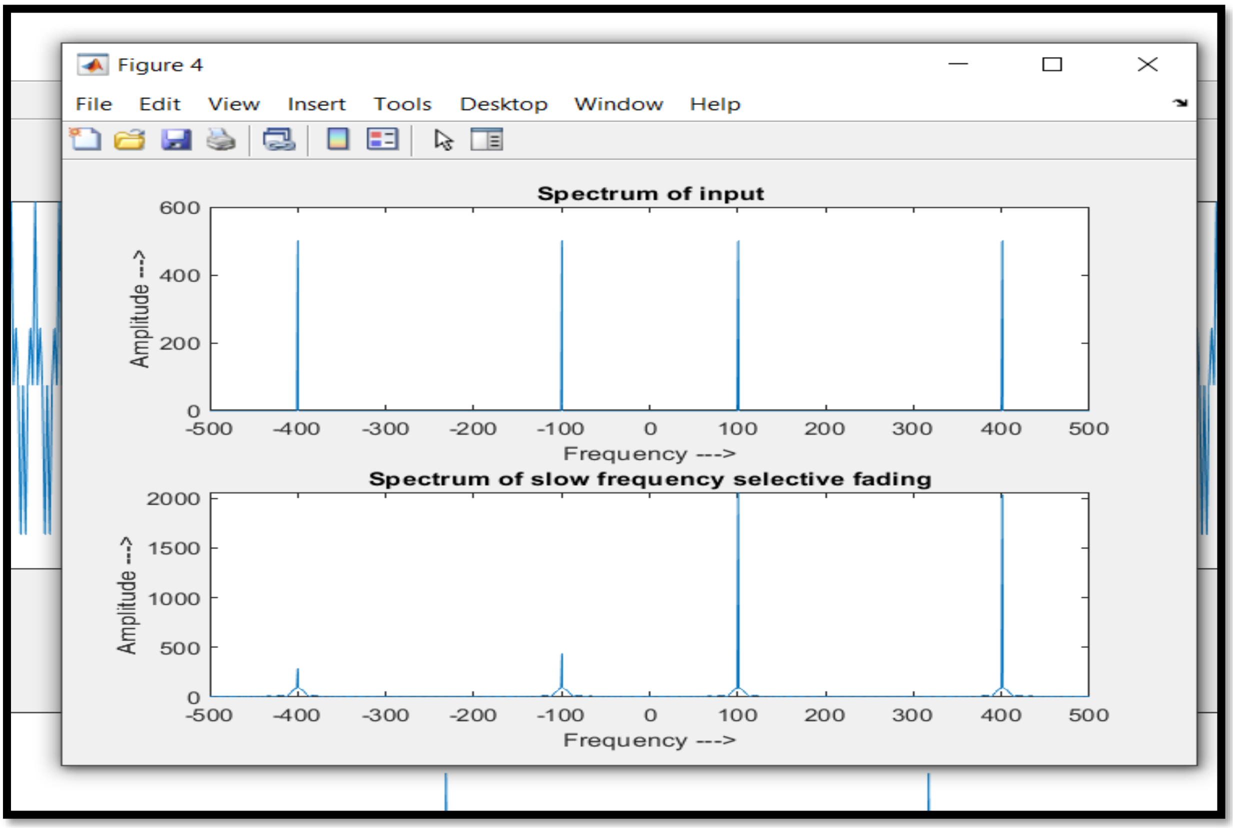



Figure 4
File Edit View Insert Tools Desktop Window Help
Spectrum of input
Amplitude --->
Frequency --->
Spectrum of slow frequency selective fading
Amplitude --->
Frequency --->

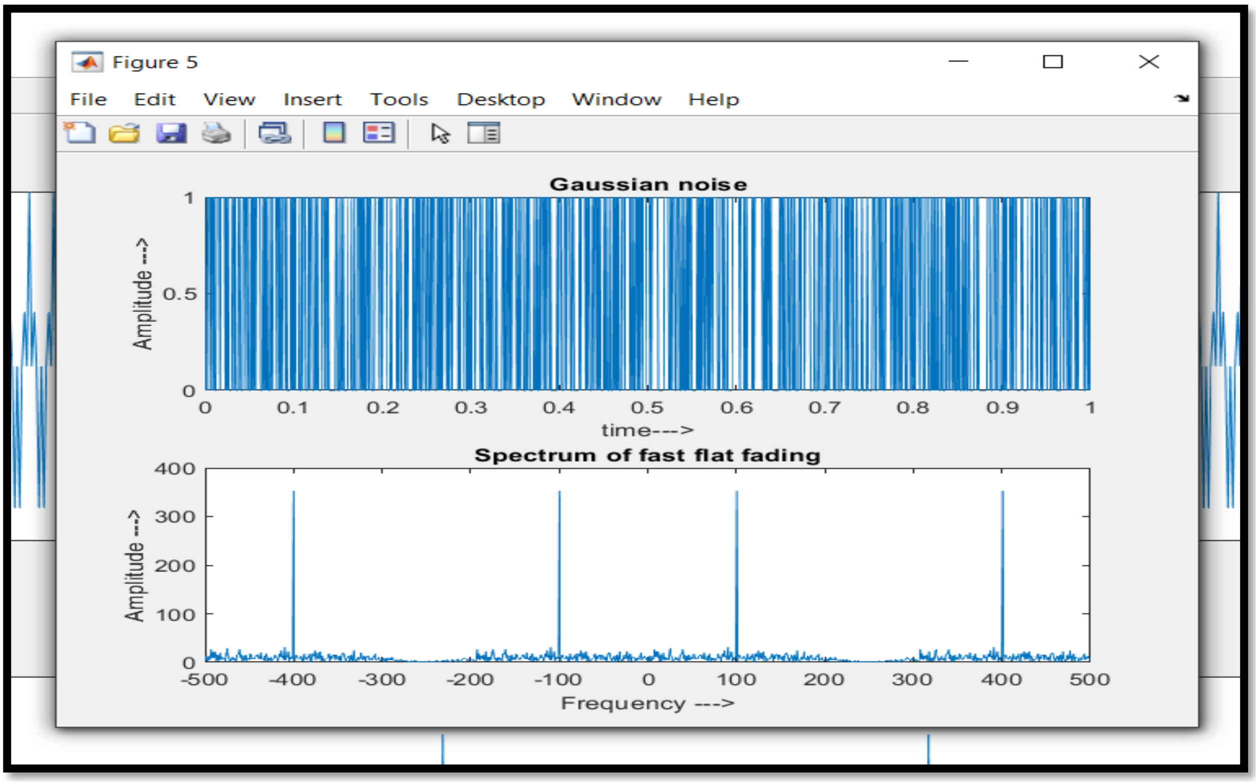


Figure 5
File Edit View Insert Tools Desktop Window Help
Gaussian noise
Amplitude --->
time--->
Spectrum of fast flat fading
Amplitude --->
Frequency --->

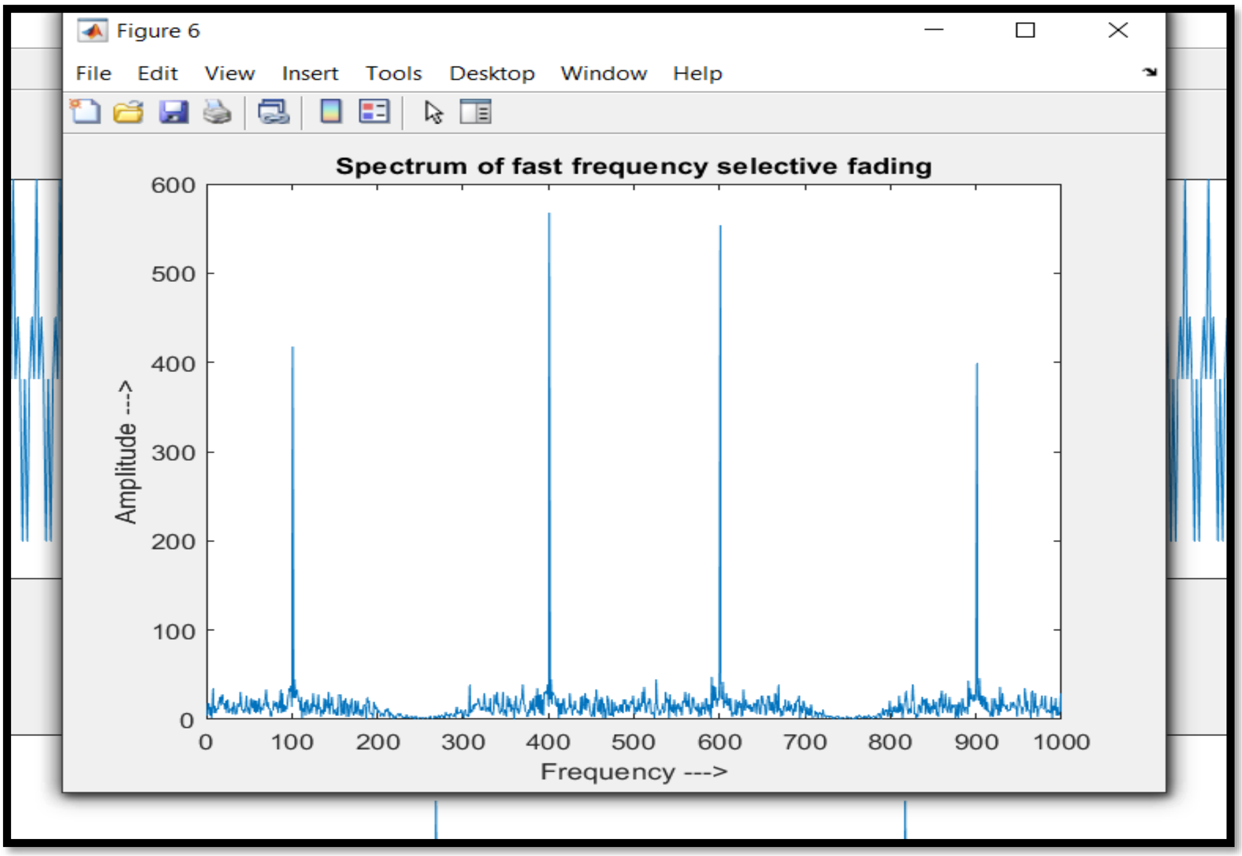



Figure 6
File Edit View Insert Tools Desktop Window Help
Spectrum of fast frequency selective fading
Amplitude --->
Frequency --->

**INFERENCE:**

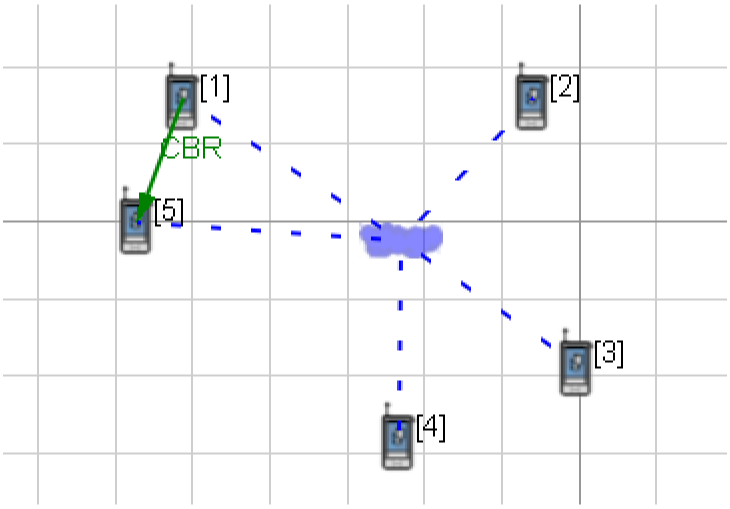
- The amplitude remains the same in slow flat fading.
- The amplitude varies for slow and frequency selective fading.
- The amplitude remains the same but the signal is spread in fast and flat fading.
- The amplitude changes and the signal is spread over the frequencies in fast and frequency selective fading.

## RESULT:

Thus, the four types of fading were simulated and graphs are plotted using MATLAB software.

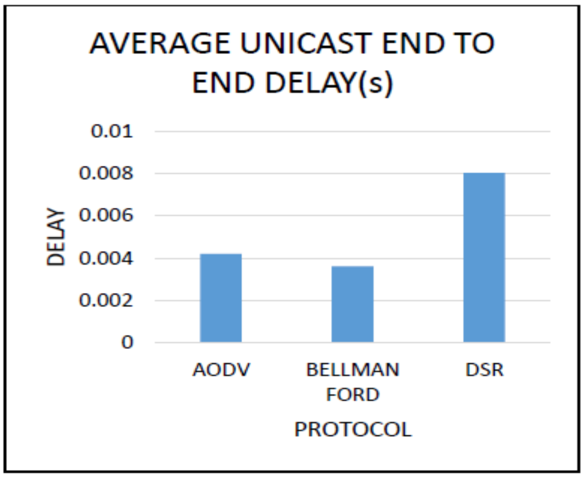
**OUTPUTS:**

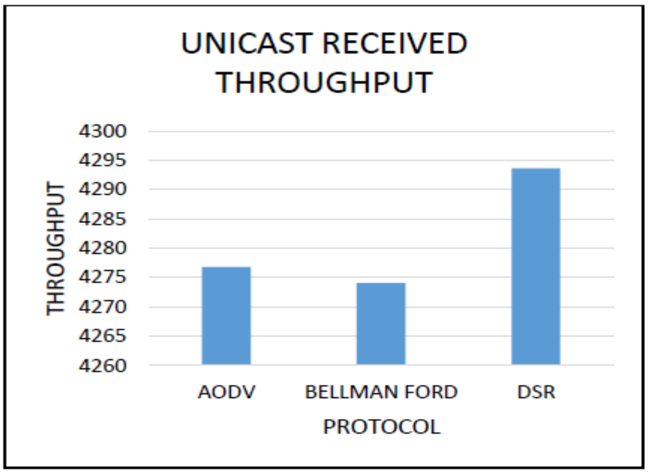
**LAYOUT:** WITHOUT MOBILITY




**TABULATION:**

| PROTOCOL | AVERAGE UNICAST END TO END DELAY(s) | UNICAST RECEIVED THROUGHPUT |
|---|---|---|
| AODV | 0.0041981 | 4276.77 |
| BELLMAN FORD | 0.00359559 | 4274.05 |
| DSR | 0.00801132 | 4293.64 |



| EX:NO:11<br><br>30.09.2021 | STUDY OF WIRELESS PROTOCOLS USING QUALNET |
|---|---|

**AIM:**

To study and implement wireless routing protocols using qualnet

- AODV-Ad-hoc on demand distance vector
- DSR-Dynamic Static Routing
- Bellman ford

**SOFTWARE REQUIRED:**

- QUALNET simulator

**THEORY:**

**AODV:AD-HOC ON-DEMAND DISTANCE VECTOR**

AODV is a routing protocol for ad-hoc mobile networks with large numbers of mobile nodes.The protocols algorithm creates routes between nodes only when the routes are requested by source nodes giving the network the flexibility to allow nodes to enter and leave the network .Routes remain active only as long as data packets are travelling along the paths from the source to the destination .When the source stops sending packets the path will time out and close.

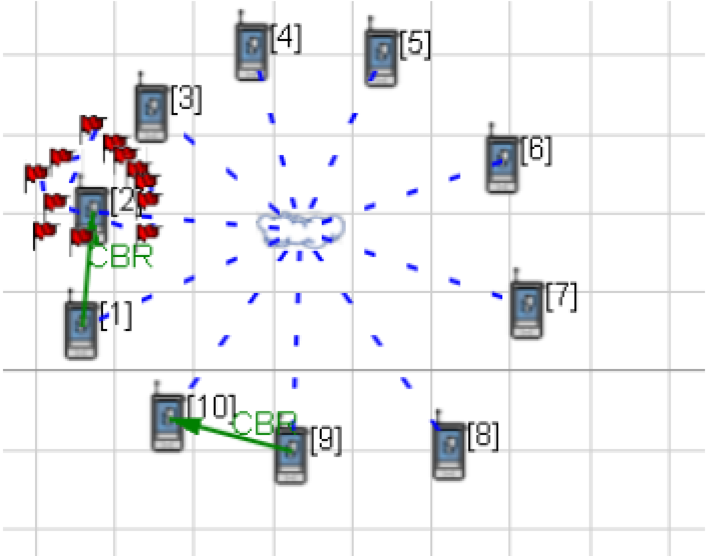
**DSR:**

Dynamic source routing is a routing protocol for wireless mesh networks. In DSR each source determines the route to be used in transmitting its packets to selected destination. There are two main components called route discovery and route maintenance. Route discovery determines the optimum path for transmission between a given source and destination. Route maintenance ensures that the transmission path remains optimum and loop free as network condition changes, even if this requires changing the route during a transmission.

**BELLMAN FORD:**

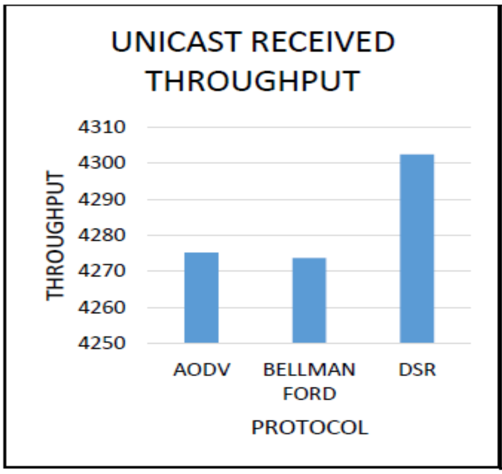
It is used to find the shortest path from one node to all other nodes in a weighed graph. The first step is to initialize the vertices. The algorithm is initially set from the starting vertex to all vertices till infinity. After the initializing step the algorithm starts circulating shortest distance from starting vertex to all other vertices. It is an example of dynamic programming. It follows bottom-up approach.

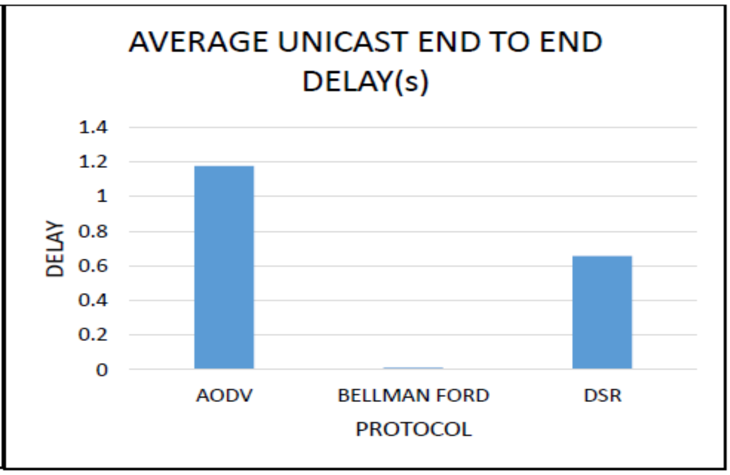
**LAYOUT:** WITH MOBILITY




**TABULATION:**

| **PROTOCOL** | **AVERAGE UNICAST END TO END DELAY(s)** | **UNICAST RECEIVED THROUGHPUT** |
|---|---|---|
| AODV | 1.176 | 4275.19 |
| BELLMAN FORD | 0.012 | 4273.67 |
| DSR | 0.656 | 4302.4 |



**PROCEDURE:**

- Open qualnet and set the suitable terrain.
- Place the required nodes and define properties for the nodes and using a subnet.
- In the properties , set the routing protocol to DSR.
- Establish CBR between two nodes.
- Save and run the simulation and there in the analyser window observe the required parameters such as throughput, average end to end delay and average jitter.
- Repeat the steps by varying the node density, (i.e) increase or decrease the number of nodes and tabulate the values.
- The same steps are to be followed for the other routing protocols such as AODV and Bellman Ford .
- The observed metrics are tabulated and plotted for different routing protocols.

**INFERENCE**

From the graph, the throughput and delay increase with increase in number of nodes. It is the highest in DSR compared to AODV and Bellman Ford. With mobility as the speed increases, DSR has maximum ed to end delay with AODV has stable delay despite the mobility

**RESULT:**

The performance evaluation of routing protocols Bellman Ford, AODV and DSR are studied by varying node density using Qualnet.