Experiment III:

NETWORKTOPOLOGY

BUS TOPOLOGY

To study the performance of token bus protocol through simulation.

SOFTWARE REQUIRED:
1. Network Simulation tool (ns2)

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

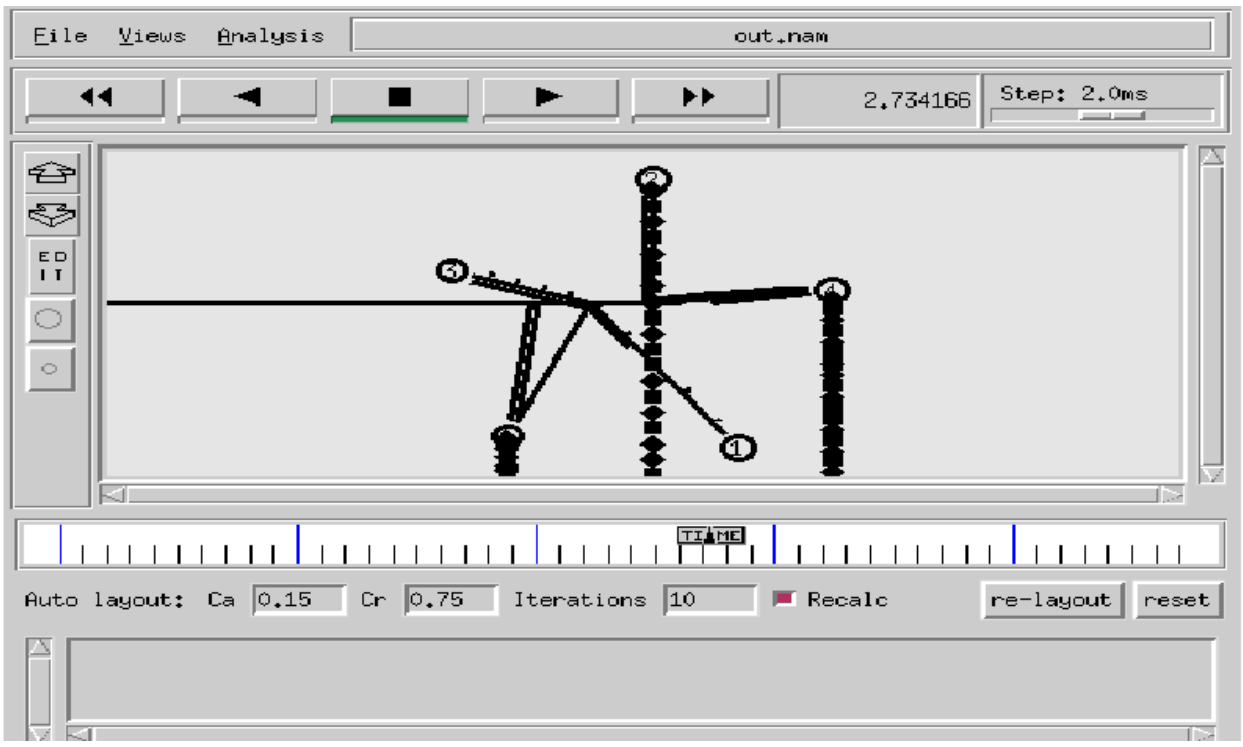
ALGORITHM:
1. Createasimulatorobject
2. Definedifferent colorsfordifferentdataflows
3. Opena namtracefileanddefinefinishprocedurethenclosethetracefile,andexecutenamontracefile.
4. Create fivenodes thatformsanetworknumbered from0to 4
5. Createduplexlinksbetween thenodesandaddOrientationtothe nodesfor settingaLANtopology
6. SetupTCPConnectionbetweenn(1) andn(3)
7. ApplyCBRTrafficoverTCP.
8. Scheduleeventsandruntheprogram.

PROGRAM:

```
#Create a simulator object
set ns [new Simulator]
#Open the nam trace file
set nf [open out.nam w]
$ns namtrace-all $nf
#Define a 'finish' procedure
proc finish {} {
global ns nf
$ns flush-trace
#Closethetracefile
close $nf
#Executenamonthetracefile
exec nam out.nam &
```

```
exit 0
}
#Create five nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
#Create Lanbetweenthe nodesset lan0 [$ns newLan "$n0 $n1 $n2 $n3 $n4" 0.5Mb 40ms
LLQueue/DropTailMAC/Csma/CdChannel]
#CreateaTCPagentand attach it to node n0
set tcp0 [new Agent/TCP]
$tcp0 set class_ 1
$ns attach-agent $n1 $tcp0
#Create a TCP Sink agent (a traffic sink) for TCP and attach it to node n3
set sink0 [new Agent/TCPSink]
$ns attach-agent $n3 $sink0
#Connectthetraffic sources withthetrafficsink
$ns connect $tcp0 $sink0
# Create a CBR traffic source and attach it to tcp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.01
$cbr0 attach-agent $tcp0
#ScheduleeventsfortheCBR agents
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
#Callthefinishprocedure after 5secondsofsimulationtime
$ns at 5.0 "finish"
#Runthesimulation
$ns run
```

OUTPUT:


File
Views
Analysis
out.nam
2.734166
Step: 2.0ms
ED
IT
TIME
Auto layout:
Ca
0.15
Cr
0.75
Iterations
10
Recalc
re-layout
reset

ExperimentIV:

RING TOPOLOGY

THEORY:
Token ring is a LAN protocol operating in the MAC layer. Token ring is standardized as per IEEE 802.5. Tokenring can operate at speeds of 4mbps and 16 mbps. The operation of token ring is as follows: When there is notraffic on the network a simple 3-byte token circulates the ring. If the token
is free (no reserved by a station ofhigher priority as explained later) then the station may seize the token and start sending the data frame. As theframe travels around the ring ach station examines the destination address and is either forwarded (if therecipient is another node) or copied. After copying4
bits of the last byte is changed. This packet then continuesaround the ring till it reaches the originating
station. After the frame makes a round trip the sender receives theframeandreleases anew token onto the ring.
ALGORITHM:
1. Createasimulatorobject
2. Definedifferent colorsfordifferentdataflows
3. Opena namtracefileanddefinefinishprocedurethenclosethetracefile,andexecutenamontracefile.
4. Create fivenodes thatformsanetworknumbered from0to 4
5. Create duplexlinksbetweenthenodestoformaRingTopology.
6. SetupTCPConnectionbetweenn(1) andn(3)
7. ApplyCBRTrafficoverTCP
8. Scheduleeventsandruntheprogram.
PROGRAM:

```
#Create a simulator object
set ns [new Simulator]
#Open the nam trace file
set nf [open out.nam w]
$ns namtrace-all $nf
#Define a 'finish' procedure
proc finish {} {
global ns nf
$ns flush-trace
#Close the trace file close $nf
#Execute nam on the trace file
exec nam out.nam &
exit 0
}
#Create five nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
#Create links between the nodes
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
```

```
$ns duplex-link $n2 $n3 1Mb 10ms DropTail
$ns duplex$ns duplex-link $n4 $n5 1Mb 10ms DropTail
$ns duplex-link $n5 $n0 1Mb 10ms DropTail
#Create a TCP agent and attach it to node n0
set tcp0 [new Agent/TCP]
$tcp0 set class_ 1
$ns attach-agent $n0 $tcp0
#Create a TCP Sink agent (a traffic sink) for TCP and attach it to node n3
set sink0 [new Agent/TCPSink]
$ns attach-agent $n4 $sink0
#Connect the traffic sources with the traffic sink
$ns connect $tcp0 $sink0
# Create a CBR traffic source and attach it to tcp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.01
$cbr0 attach-agent $tcp0
#Schedule events for the CBR agents
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
#Call the finish procedure after 10 seconds of simulation time
$ns at 5.0 "finish"
#Run the simulation
$ns run
```

Experiment No. 5 STAR TOPOLOGY

THEORY:

Star networks are one of the most common computer network topologies. In its simplest form, a star networkconsists of one central switch, hub or computer, which acts as a conduit to transmit messages.

This consists of acentral node, to which all other nodes are connected; this central node provides a common connection point forallnodes through ahub. In startopology, every node(computerworkstation orany otherperipheral)isconnected to a central node called a hub or switch. The switch is the server and the peripherals are the clients.Thus, the hub and leaf nodes, and the transmission lines between them, form a graph with the topology of a star.If the central node is passive, the originating node must be able to tolerate the reception of an echo of its owntransmission, delayed by the two-way transmission time (i.e. to and from the central node) plus any delaygenerated in the central node. An active star network has an active central node that usually has the means topreventecho-related problems.

The star topology reduces the damage caused by line failure by connecting all of the systems to a central node.When appliedto abus-basednetwork,thiscentralhubrebroadcastsalltransmissionsreceivedfrom anyperipheral node to all peripheral nodes on the network, sometimes including the originating node. All peripheralnodes may thus communicate with all others by transmitting to, and receiving from, the central node only. Thefailure of a transmission line linking any peripheral node to the central node will result in the isolation of thatperipheralnodefromallothers,buttherestof the systems willbeunaffected.

ALGORITHM:
1. Createasimulatorobject
2. Definedifferent colorsfordifferentdataflows
3. Opena namtracefileanddefinefinishprocedurethenclosethetracefile,andexecutenamontracefile.
4. Create sixnodesthatformsanetworknumberedfrom0to5
5. Create duplexlinksbetween thenodestoformaSTARTopology
6. SetupTCPConnectionbetweenn(1) andn(3)
7. ApplyCBRTrafficoverTCP
8. Scheduleeventsandrunthe program.

PROGRAM:

```
set ns [new Simulator]
set nf [open ex1.nam w]
#Open the nam trace file
set nf [open ex1.nam w]
$ns namtrace-all $nf
#Define a 'finish' procedure
proc finish {} {
global ns nf
$ns flush-trace
#Close the trace file
close $nf
#Executenam on the trace file
exec nam ex1.nam &exit 0
}
#Create six nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
#Change the shape of center node in a star topology
$n0 shape square
#Create links between the nodes
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n0 $n2 1Mb 10ms DropTail
$ns duplex-link $n0 $n3 1Mb 10ms DropTail
$ns duplex-link $n0 $n4 1Mb 10ms DropTail
$ns duplex-link $n0 $n5 1Mb 10ms DropTail
#Create a TCP agent and attach it to node n0
set tcp0 [new Agent/TCP]
$tcp0 set class_ 1
$ns attach-agent $n1 $tcp0
#Create a TCP Sink agent (a traffic sink) for TCP and attach it to node n3
set sink0 [new Agent/TCPSink]
$ns attach-agent $n3 $sink0
#Connect the traffic sources with the traffic sink
$ns connect $tcp0 $sink0
# Create a CBR traffic source and attach it to tcp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.01
```

```
$cbr0 attach-agent $tcp0
#Schedule events for the CBR agents
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
#Call the finish procedure after 5 seconds of simulation time
$ns at 1.0 "finish"
#Run the simulation
$ns run
```

Experiment 6:

IMPLEMENTATION OF DIFFERENT LANs USING SWITCH / HUB / ROUTER AS INTERCONNECTING DEVICE

THEORY:

Hub:

A Hub is just a connector that connects the wires coming from different sides. There is no signal processing or regeneration. It is an electronic device that operates only on physical layers of the OSI model.

It is also known as a repeater as it transmits signal to every port except the port from where signal is received. Also, hubs are not that intelligent in communication and processing information for 2nd and 3rd layer.

Switch:

Switch is a point to point communication device. It operates at the data link layer of OSI model. It uses switching table to find out the correct destination.

Basically, it is a kind of bridge that provides better connections. It is a kind of device that set up and stop the connections according to the requirements needed at that time. It comes up with many features such as flooding, filtering and frame transmission

Router:

Routers are the multiport devices and more sophisticated as compared to repeaters and bridges. It contains a routing table that enables it to make decision about the route i.e. to determine which of several possible paths between the source and destination is the best for a particular transmission.

PROGRAM:

```
set ns [new Simulator]
#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red
#Open the Trace files
set file1 [open out.tr w]
set winfile [open WinFile w]
$ns trace-all $file1
#Open the NAM trace file
set file2 [open out.nam w]
$ns namtrace-all $file2
#Define a 'finish' procedure
proc finish {} {
global ns file1 file2
$ns flush-trace
close $file1
close $file2
exec nam out.nam &
exit 0
```

```
#Create six nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
set n7 [$ns node]
set n8 [$ns node]
set n9 [$ns node]
$n9 label "Router"
$n1 color red
$n1 shape box
#Create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns simplex-link $n2 $n3 0.3Mb 100ms DropTail
$ns simplex-link $n3 $n2 0.3Mb 100ms DropTail
$ns duplex-link $n9 $n3 2Mb 10ms DropTail
$ns duplex-link $n9 $n6 2Mb 10ms DropTail
set lan [$ns newLan "$n3 $n4 $n5" 0.5Mb 40ms LL Queue/DropTail MAC/Csma/Cd
Channel]
set lan [$ns newLan "$n6 $n7 $n8" 0.5Mb 40ms LL Queue/DropTail MAC/Csma/Cd
Channel]
#Setup a TCP connection
set tcp [new Agent/TCP/Newreno]
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink/DelAck]
$ns attach-agent $n4 $sink
$ns connect $tcp $sink
$tcp set fid_ 1
$tcp set window_ 8000
$tcp set packetSize_ 552
set tcp3 [new Agent/TCP]
$ns attach-agent $n9 $tcp3
set sink4 [new Agent/TCPSink/DelAck]
$ns attach-agent $n4 $sink4
$ns connect $tcp3 $sink4
$tcp3 set fid_ 1
$tcp3 set window_ 8000
$tcp3 set packetSize_ 552
#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTPset ftp2 [new Application/FTP]
$ftp2 attach-agent $tcp3
$ftp2 set type_ FTP
#Setup a TCP connection
```

```
set tcp1 [new Agent/TCP/Newreno]
$ns attach-agent $n6 $tcp1
set sink1 [new Agent/TCPSink/DelAck]
$ns attach-agent $n8 $sink1
$ns connect $tcp1 $sink1
$tcp1 set fid_ 1
$tcp1 set window_ 8000
$tcp1 set packetSize_ 552
#Setup a FTP over TCP connection
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ftp1 set type_ FTP
#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n5 $null
$ns connect $udp $null
$udp set fid_ 2
#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 0.01mb
$cbr set random_ false
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 1.0 "$ftp1 start"
$ns at 1.0 "$ftp2 start"
$ns at 200.0 "$ftp1 stop"
$ns at 200.0 "$ftp2 stop"
$ns at 200.0 "$ftp stop"
$ns at 200.5 "$cbr stop"
# next procedure gets two arguments: the name of the
# tcp source node, will be called here "tcp",
# and the name of output file.
proc plotWindow {tcpSource file} {
global ns
set time 0.1
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
set wnd [$tcpSource set window_]puts $file "$now $cwnd"
$ns at [expr $now+$time] "plotWindow $tcpSource $file" }
$ns at 0.1 "plotWindow $tcp $winfile"
$ns at 5 "$ns trace-annotate \"packet drop\""
# PPP
$ns at 125.0 "finish"
$ns run
```