| **Name of Student:** Ajay Karthikesan | | | |
| --- | --- | --- | --- |
| **Roll Number:** 57 | | **Assignment Number:** 4 | |
| **Aim of Assignment:**  Write a program to simulate bus topology | | | |
| **DOP:** 24.4.23 | | **DOS:** 29.4.23 | |
| **CO Mapped:**  CO2 | **PO Mapped:**  PO1, PO2, PO3, PO5, PO7, PSO1 | **Faculty Signature:** | **Marks:** |

## 

## Practical No. 4

**Aim:** Write a program to simulate bus topology

**Theory:**

* A bus network is a network topology in which nodes are directly connected to a common half-duplex link called a bus.
* A host on a bus network is called a station. In a bus network, every station will receive all network traffic, and the traffic generated by each station has equal transmission priority. A bus network forms a single network segment and collision domain. In order for nodes to share the bus, they use a medium access control technology such as carrier-sense multiple access (CSMA) or a bus master.
* CsmaHelper
  + build a set of CsmaNetDevice objects
  + Normally we eschew multiple inheritance, however, the classes PcapUserHelperForDevice and AsciiTraceUserHelperForDevice are treated as "mixins". A mixin is a self-contained class that encapsulates a general attribute or a set of functionality that may be of interest to many other classes.
* MobilityHelper
  + Helper class used to assign positions and mobility models to nodes.
  + MobilityHelper::Install is the most important method here.
* ApplicationInterface
  + Interface to network animator
  + Provides functions that facilitate communications with an external or internal network animator.

**Code:**

#include "ns3/animation-interface.h"

#include "ns3/applications-module.h"

#include "ns3/core-module.h"

#include "ns3/csma-module.h"

#include "ns3/internet-module.h"

#include "ns3/ipv4-global-routing-helper.h"

#include "ns3/mobility-module.h"

#include "ns3/network-module.h"

#include "ns3/point-to-point-module.h"

#include <bits/stdint-intn.h>

// Default Network Topology

//

// 10.1.1.0

// n0 -------------- n1 n2 n3 n4

// point-to-point | | | |

// ================

// LAN 10.1.2.0

using namespace ns3;

NS\_LOG\_COMPONENT\_DEFINE("Bus");

int main(int argc, char \*argv[]) {

bool verbose = true;

uint32\_t nCsma = 3;

CommandLine cmd(\_\_FILE\_\_);

cmd.AddValue("nCsma", "Number of \"extra\" CSMA nodes/devices", nCsma);

cmd.AddValue("verbose", "Tell echo applications to log if true", verbose);

cmd.Parse(argc, argv);

if (verbose) {

LogComponentEnable("UdpEchoClientApplication", LOG\_LEVEL\_INFO);

LogComponentEnable("UdpEchoServerApplication", LOG\_LEVEL\_INFO);

}

nCsma = nCsma == 0 ? 1 : nCsma;

NodeContainer p2pNodes;

p2pNodes.Create(2);

NodeContainer csmaNodes;

csmaNodes.Add(p2pNodes.Get(1));

csmaNodes.Create(nCsma);

PointToPointHelper pointToPoint;

pointToPoint.SetDeviceAttribute("DataRate", StringValue("5Mbps"));

pointToPoint.SetChannelAttribute("Delay", StringValue("2ms"));

NetDeviceContainer p2pDevices;

p2pDevices = pointToPoint.Install(p2pNodes);

CsmaHelper csma;

csma.SetChannelAttribute("DataRate", StringValue("100Mbps"));

csma.SetChannelAttribute("Delay", TimeValue(NanoSeconds(6560)));

NetDeviceContainer csmaDevices;

csmaDevices = csma.Install(csmaNodes);

InternetStackHelper stack;

stack.Install(p2pNodes.Get(0));

stack.Install(csmaNodes);

Ipv4AddressHelper address;

address.SetBase("10.1.1.0", "255.255.255.0");

Ipv4InterfaceContainer p2pInterfaces;

p2pInterfaces = address.Assign(p2pDevices);

address.SetBase("10.1.2.0", "255.255.255.0");

Ipv4InterfaceContainer csmaInterfaces;

csmaInterfaces = address.Assign(csmaDevices);

UdpEchoServerHelper echoServer(9);

ApplicationContainer serverApps = echoServer.Install(csmaNodes.Get(nCsma));

serverApps.Start(Seconds(1.0));

serverApps.Stop(Seconds(10.0));

UdpEchoClientHelper echoClient(csmaInterfaces.GetAddress(nCsma), 57);

echoClient.SetAttribute("MaxPackets", UintegerValue(10));

echoClient.SetAttribute("Interval", TimeValue(Seconds(2.0)));

echoClient.SetAttribute("PacketSize", UintegerValue(1024));

ApplicationContainer clientApps = echoClient.Install(p2pNodes.Get(0));

clientApps.Start(Seconds(2.0));

clientApps.Stop(Seconds(10.0));

Ipv4GlobalRoutingHelper::PopulateRoutingTables();

pointToPoint.EnablePcapAll("second");

csma.EnablePcap("second", csmaDevices.Get(1), true);

// generate trace file

MobilityHelper mobility;

mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");

mobility.SetPositionAllocator(

"ns3::GridPositionAllocator", "MinX", DoubleValue(0.0), "MinY",

DoubleValue(0.0), "DeltaX", DoubleValue(5.0), "DeltaY", DoubleValue(5.0),

"GridWidth", UintegerValue(3), "LayoutType", StringValue("RowFirst"));

mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");

mobility.Install(p2pNodes);

mobility.Install(csmaNodes);

AnimationInterface{"test.xml"};

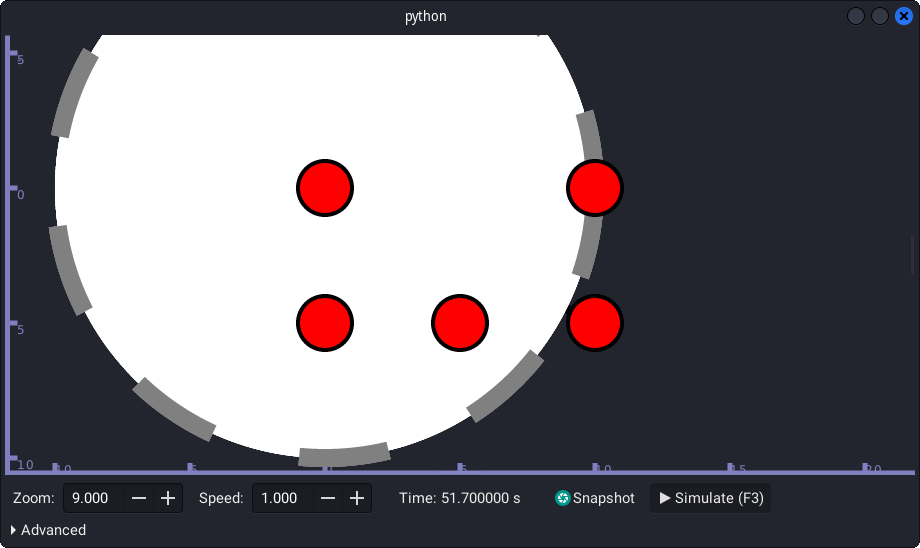
Simulator::Run();

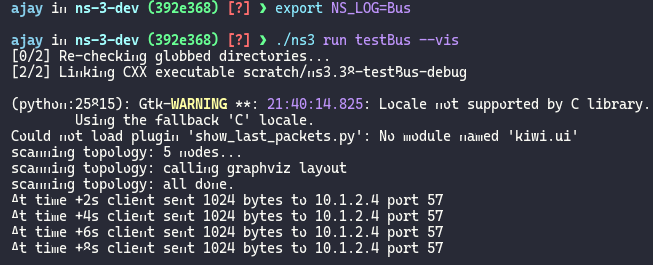
Simulator::Destroy();

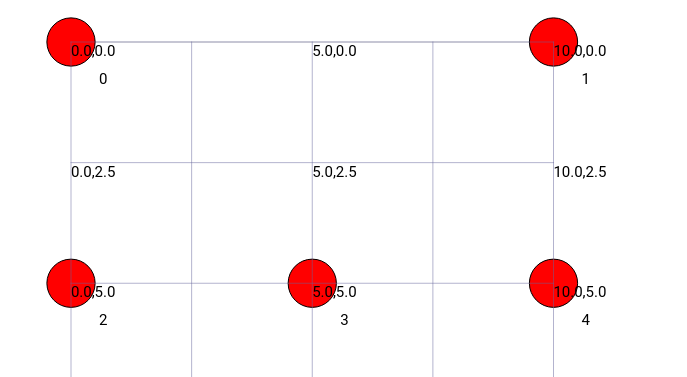
return 0;

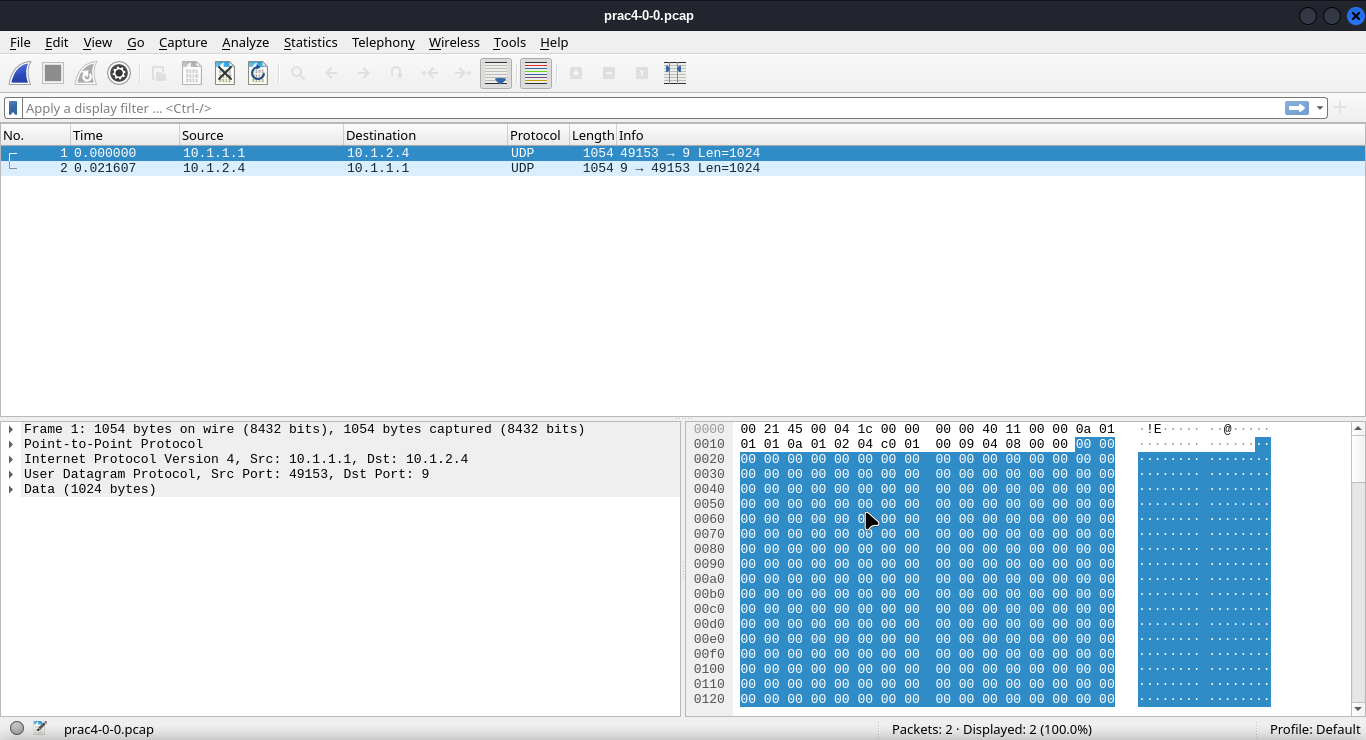
}

**Output:**









**Conclusion:**

I learnt how to write a program to simulate bus topology.