INDEX

Name : Shuvom Dhar

Sem Class : 5 Sec. : 1 Roll No. : 2211200001043

School / College : Sister Nivedita University

Subject : Computer Network Lab

PARTICULARS OF THE EXPERIMENTS PERFORMED

Date	Serial No.	Experiment No.	Subject/Experiment	Signature	Remarks
11/8/24	1	1	Sending packets between directly linked two devices		
17/8/24	2	2.1	Implementation of Basic Topologies using Packet Tracer - Star Topology	03/09/2024	
17/8/24	3	2.2	Bus Topology		
24/8/24	4	2.3	Mesh Topology		
24/8/24	5	3.1	Ring Topology		
25/8/24	6	3.2	Configuration of a DHCP Server		
7/9/24	7	4	Configuration of DNS and Web Server	24/09/2024	
1/10/24	8	5	Implementation of Router to connect different types of	22/10/2024	

PARTICULARS OF THE EXPERIMENTS PERFORMED

Date	Serial No.	Experiment No.	Subject/Experiment	Signature	Remarks
			Network		
25/10/2024	9	6	Parity Bit Method Error Checking	29/10/2024	
1/11/2024	10	7	CheckSum Method for Error Detection	06/11/2024	
5/11/2024	11	8	Cyclic Redundancy Check		

DATE	PAGE NO.
11/8/2024	1

EXPT. NO.
1

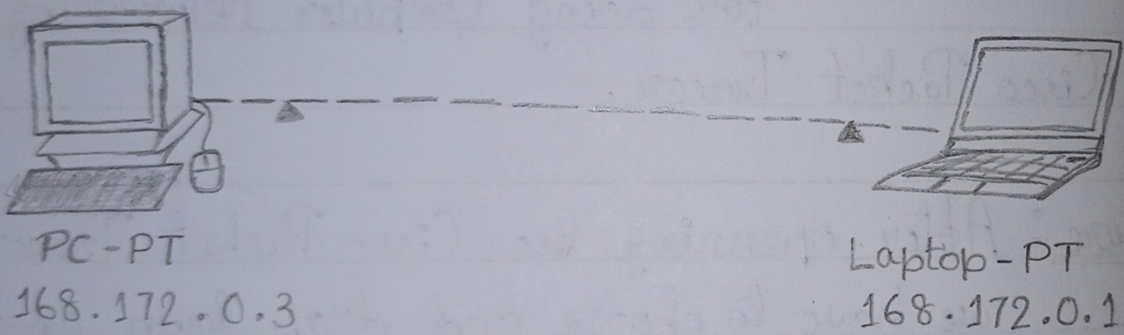
Sending packets between directly linked two devices :—

Objective: Our main goal is to connect two devices directly with a wire so that each of the devices can send packets with each other.

Required Software: Here we are going to use a software for doing Computer Networking lab with named Cisco Packet Tracer.

Procedure: After openning the Cisco Packet Tracer simulator, we have to choose and drop down any two devices. Then we have to make connection between them. After that we have to provide an IP Address to each of the devices. Now they are ready to send packets. But before that we need to finally assign the IP Addresses to each of the devices according to the following steps :

Double tap on each devices ⟶ Desktop ⟶ IP Configuration ⟶ Write your given IP Address to the respective devices ⟶ the subnet mask for the corresponding devices would automatically be set



Connecting two nodes directly

Then we need to check whether our design is properly working or not. So for that we need to double tap on any one of the device then select 'Desktop' then go to 'Command Prompt'. In the 'Command Prompt' we need to write one command —

ping "IP-Address" output heading

At the place of "IP Address" you need to write receiver's IP Address.

After writting the "ping" command if you get some message like —

"Reply from "IP-Address": bytes = 32 time = 1ms TTL = 120"

then you will get know that you packets have already been sent.

By writing a command "ipconfig "IP-Address", we will get know all the details of that device.

Conclusion: By doing this we will get know how to set IP-Address of the device, how to send packets to the another device, how to check for the packets are successfully reached and how to see the details of the devices. These will help me for doing other lab projects in future.

13/08/2024

Output: The output will be —

C:\>ping 168.172.0.1
Pinging 168.172.0.1 with 32 bytes of data:

Reply from 168.172.0.1: bytes=32 time<1ms TTL=128
Reply from 168.172.0.1: bytes=32 time<1ms TTL=128
Reply from 168.172.0.1: bytes=32 time<1ms TTL=128
Reply from 168.172.0.1: bytes=32 time<1ms TTL=128

Ping statistics for 168.172.0.1:
    Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum=0ms, Maximum=0ms, Average=0ms

DATE	PAGE NO.	EXPT. NO.
17/8/2024	1	2-1

Implementation of Basic Topologies using Packet Tracer

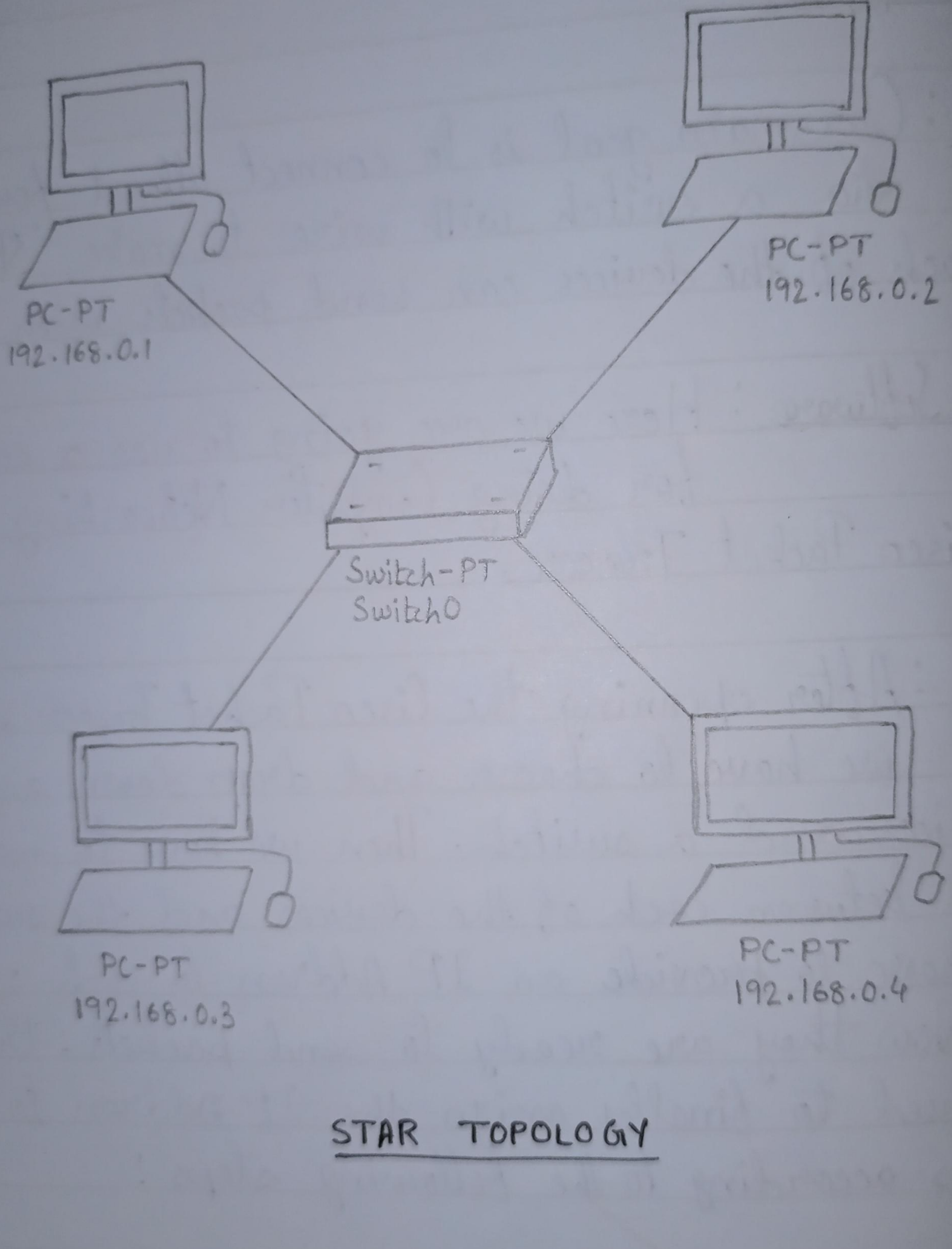
Star Topology:

Objective: Our main goal is to connect atleast four devices via a switch with wires to make 'Star Topology' so that each of the devices can send packets with each other.

Required Software: Here we are going to use a software for doing Computer Networking lab with named Cisco Packet Tracer.

Procedure: After openning the Cisco Packet Tracer simulator, we have to choose and drop down any two or four devices and a switch. Then we have to make connection between each of the devices and the switch. After that we have to provide an IP Address to each of the devices. Now they are ready to send packets. But before that we need to finally assign the IP Address to each of the devices according to the following steps:

Double tap on each devices → Desktop → IP Configuration → Write your given IP Address to the respective devices → the subnet mask for the corresponding devices would automatically be set



STAR TOPOLOGY

Then we need to check whether our design is properly working or not. So for that we need to check the output.

Output: The output will be –

C:\> ping 192.168.0.2

Pinging 192.168.0.2: bytes=32 time<1ms TTL=128

Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.2:
    Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum=0ms, Maximum=0ms, Average=0ms

C:\> ping 192.168.0.3

Pinging 192.168.0.3 with 32 bytes of data:

Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128

Reply from 192.168.0.3: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.3:
Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum=0ms, Maximum=0ms, Average=0ms

C:\>ping 192.168.0.4

Pinging 192.168.0.4 with 32 bytes of data:

Reply from 192.168.0.4: byte=32 time<1ms TTL=128
Reply from 192.168.0.4: byte=32 time<1ms TTL=128
Reply from 192.168.0.4: byte=32 time<1ms TTL=128
Reply from 192.168.0.4: byte=32 time<1ms TTL=128

Ping statistics for 192.168.0.4:
Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum=0ms, Maximum=0ms, Average=0ms

Conclusion: By doing this we will get know how 'Star Topology' is made and how it helps to send packets to different devices. This will help me for doing other lab projects in future.

DATE	PAGE NO.	EXPT. NO.
17/8/2024	4	2.2

Bus Topology :

Objective : Our main goal is to make 'Bus Topology' by using atleast four/three devices, switches and wires.

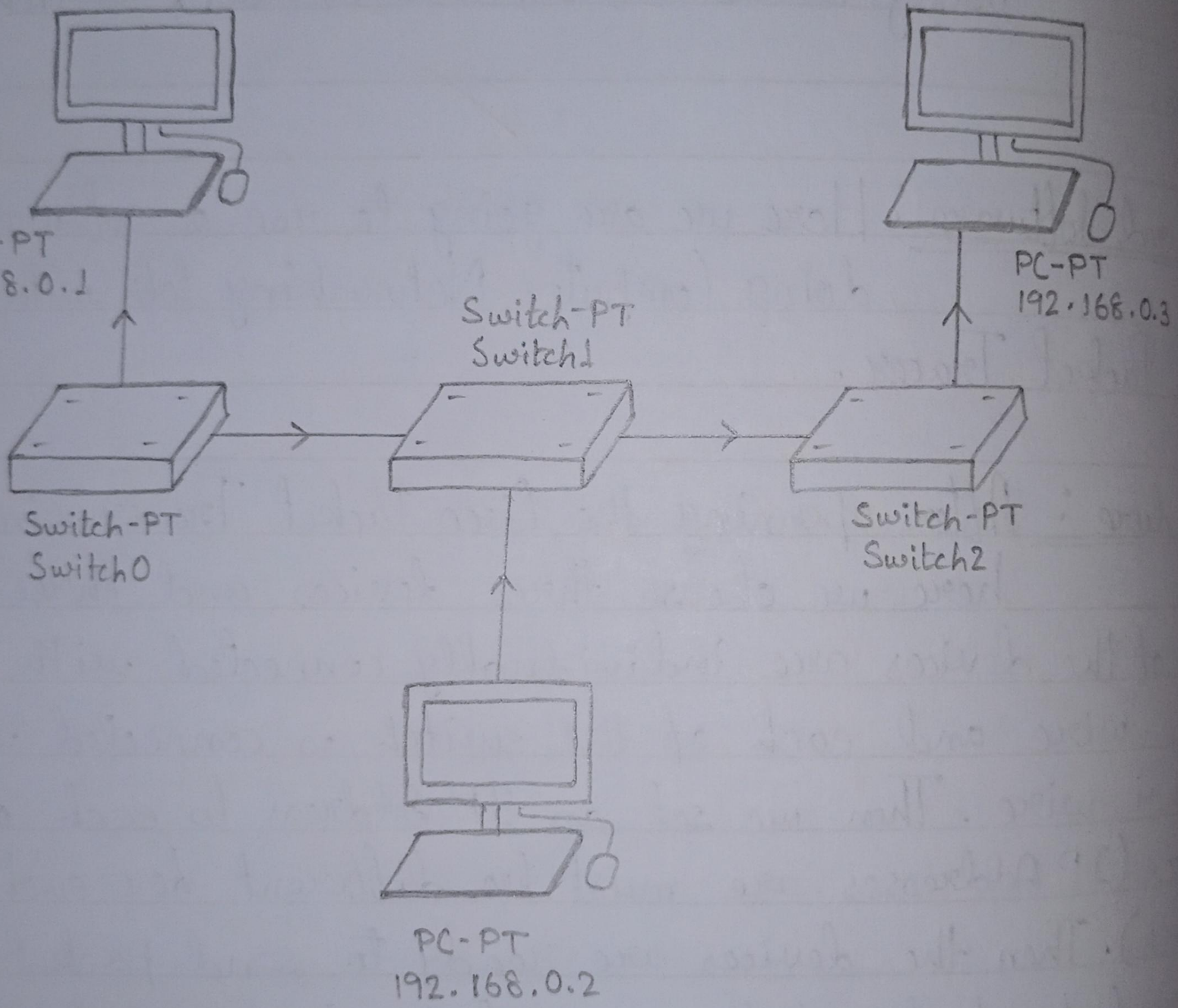
Required Software : Here we are going to use a software for doing Computer Networking lab with named Cisco Packet Tracer.

Procedure : After openning the Cisco Packet Tracer simulator, here we choose three devices and three switches. Each of the devices are individually connected with a switch by a wire and each of the switch is connected with another wire. Then we set an IP Address to each of the devices (IP Addresses are must be different for each of the devices). Then the devices are ready to send packets. Then we need to check whether our design is properly working or not. So for that we need to check the output.

Output : The output will be —

C:\> ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:



BUS TOPOLOGY

Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time=9ms TTL=128

Ping statistics for 192.168.0.2:
Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum=0ms, Maximum=9ms, Average=2ms

C:\> ping 192.168.0.3

Pinging 192.168.0.3 with 32 bytes of data:

Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.3:
Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum=0ms, Maximum=0ms, Average=0ms

DATE	PAGE NO.	EXPT. NO.
17/8/2024	6	2.2

<u>Conclusion</u> : By doing this we will get know how 'Bus Topology' is made and how it helps to send packets to different devices. This will help me for doing other lab projects in future.

DATE	PAGE NO.	EXPT. NO.
17/8/2024	7	2.3

Mesh Topology

Objective: Our main goal is to make 'Mesh Topology' by using atleast four devices, switches and wires.

Required Software: Here we are going to use a software for doing Computer Networking lab with named Cisco Packet Tracer.

Procedure: After openning the Cisco Packet Tracer simulator, here we choose four devices and four switches. Each of the devices are individually connected with a switch by a wire and each of the switch is connected with another wire. Then we set an IP Address to each of the devices (IP Addresses are must be different for each of the devices). Then the devices are ready to send packets. Then we need to check whether our design is properly working or not. So for that we need to check the output.

Output: The output will be —

C:\>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.2:
    Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum=0ms, Maximum=0ms, Average=0ms

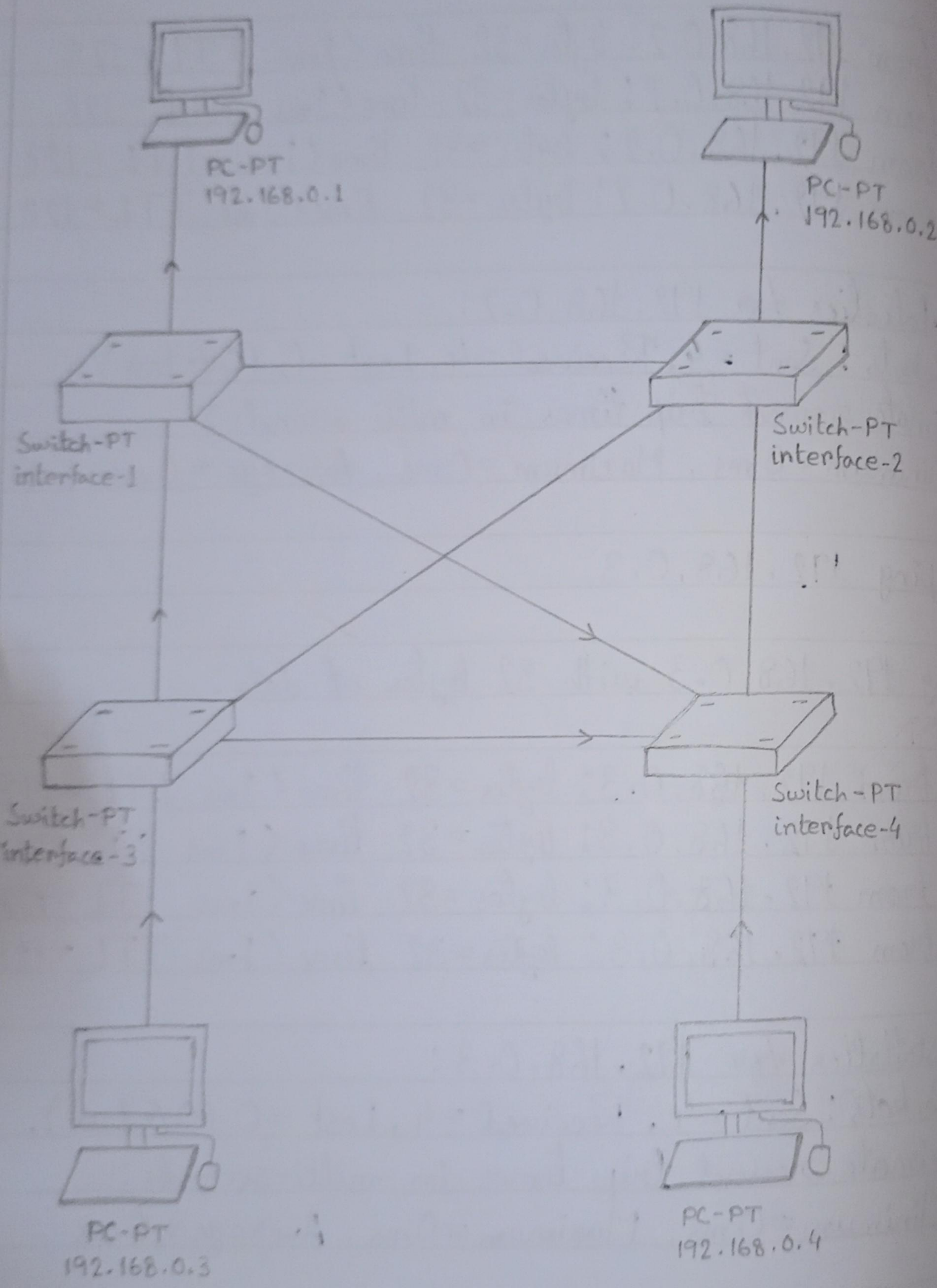
C:\>ping 192.168.0.3

Pinging 192.168.0.3 with 32 bytes of data:

Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.3:
    Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum=0ms, Maximum=0ms, Average=0ms

C:\>ping 192.168.0.4



MESH TOPOLOGY

DATE	PAGE NO.	EXPT. NO.
17/8/2024	9	2.3

Pinging 192.168.0.4 with 32 bytes of data:

Reply from 192.168.0.4: bytes=32 time<1ms TTL=128
Reply from 192.168.0.4: bytes=32 time<1ms TTL=128
Reply from 192.168.0.4: bytes=32 time<1ms TTL=128
Reply from 192.168.0.4: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.4:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms

Conclusion: By doing this we will get know how 'Mesh Topology' is made and how it helps to send packets to different devices. This will help me for doing other lab projects in future.

20/08/2024

DATE	PAGE NO.	EXPT. NO.
24/8/2024	1	3.1

Ring Topology

Objective: Our main goal is to make 'Ring Topology' by using atleast four devices, four switches and some wires.

Required Software: Here we are going to use a software for doing Computer Networking lab with named Cisco Packet Tracer.

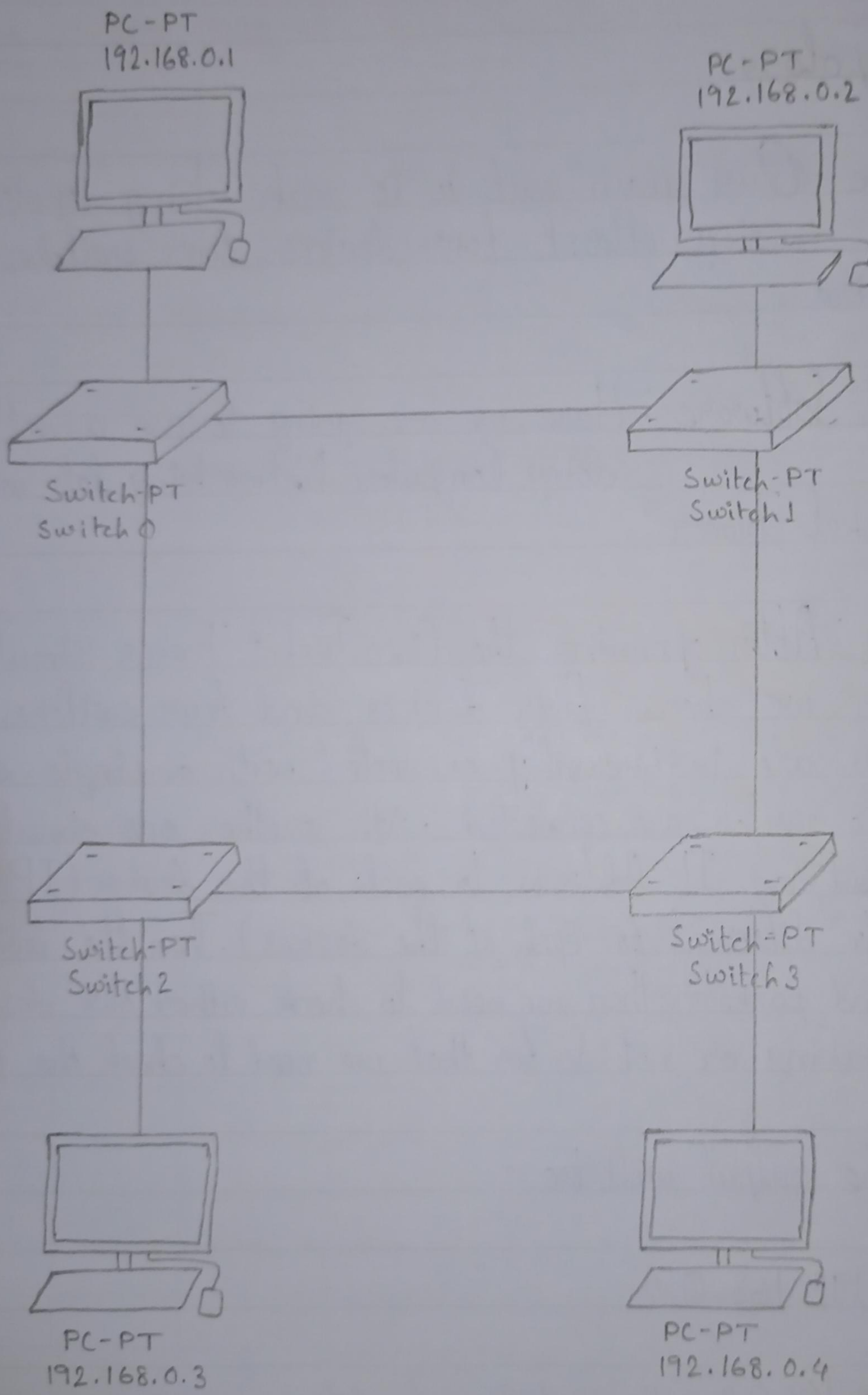
Procedure: After openning the Cisco Packet Tracer simulator, here we choose four devices and four switches. Each of the devices are individually connected with a single switch and each of the switch are connected with another one simultaneously. Then we set an IP Address to each of the devices (IP Addresses are must be different for each of the devices). Then the devices are ready to send packets. Then we need to check either our design is properly working or not. So for that we need to check the output.

Output: The output will be—

C:\>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time<1ms TTL=128



RING TOPOLOGY

DATE	PAGE NO.	EXPT. NO.
24/8/2024	2	3.1

Reply from 192.168.0.2: bytes=32 time=3ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.2:
Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum=0ms, Maximum=3ms, Average=0ms

C:\>ping 192.168.0.3

Pinging 192.168.0.3 with 32 bytes of data:

Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128
Reply from 192.168.0.3: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.3:
Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum=0ms, Maximum=0ms, Average=0ms

C:\>ping 192.168.0.4

Pinging 192.168.0.4 with 32 bytes of data:

Reply from 192.168.0.4: bytes=32 time=18ms TTL=128
Reply from 192.168.0.4: bytes=32 time<1ms TTL=128
Reply from 192.168.0.4: bytes=32 time<1ms TTL=128
Reply from 192.168.0.4: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.4:
Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum=0ms, Maximum=0ms, Average=0ms

<u>Conclusion</u>: By doing this we will get know how 'Ring Topology' is made and how it helps to send packets to different devices. This will help me for doing other lab projects in future.

DATE 25/8/2024 PAGE NO. 4

EXPT. NO. 3.2

Configuration of a DHCP Server

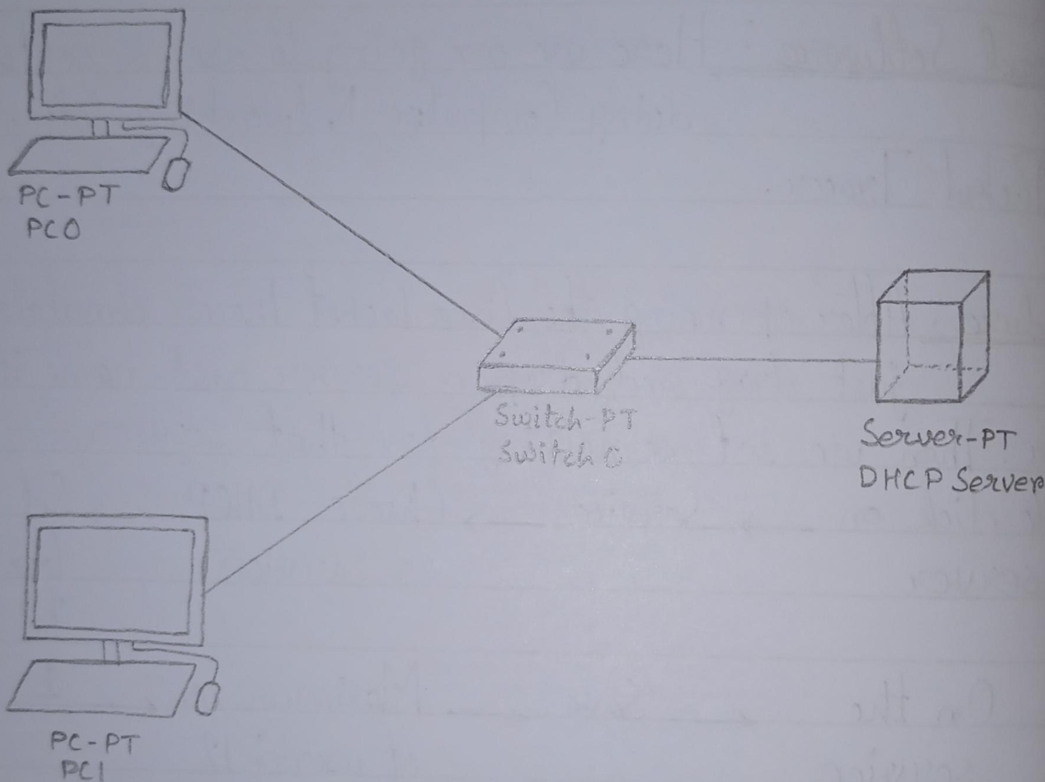
Objective: Our main goal is to make the configuration of a DHCP Server. Here we are going to use mainly one DHCP Server, one switch and two devices.

Required Software: Here we are going to use a software for doing Computer Networking lab with named Cisco Packet Tracer.

Procedure: After openning the Cisco Packet Tracer simulator, we just drag and drop a server and name it as DHCP Server. Then we set something for that server ⟶

Double click on the server ⟶ Services ⟶ Choose DHCP as Services ⟶ Set the last box of Start IP Address as 1 ⟶ Maximum no. of users: 12 ⟶ Save ⟶ On the service

Then drag and drop one switch and two devices. Create a connection between them. The single connection will be made between the server and switch and each of the switch is connected with the switch. Before making any changes for the DHCP Server and do save it, we just need to assign an IP Address for it. After doing these all we can ensure that the IP Addresses of the devices will automatically be assigned. Then the devices are ready to send packets. Then we need to check



Configuration of a DHCP Server

either our design is properly working or not. So for that we need to check the output.

Output: The output will be—

C:\>ping 192.168.0.1

Pinging 192.168.0.1 with 32 bytes of data:

Reply from 192.168.0.1: bytes=32 time<1ms TTL=128
Reply from 192.168.0.1: bytes=32 time<1ms TTL=128
Reply from 192.168.0.1: bytes=32 time<1ms TTL=128
Reply from 192.168.0.1: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.1:
Packets: Sent=4, Received=4, Lost=0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum=0ms, Maximum=0ms, Average=0ms

C:\>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128
Reply from 192.168.0.2: bytes=32 time<1ms TTL=128

Reply from 192.168.0.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms

Conclusion: By doing this we will get know how to configure a DHCP Server and how it helps to send packets to different devices. This will help me for doing lab projects in future.

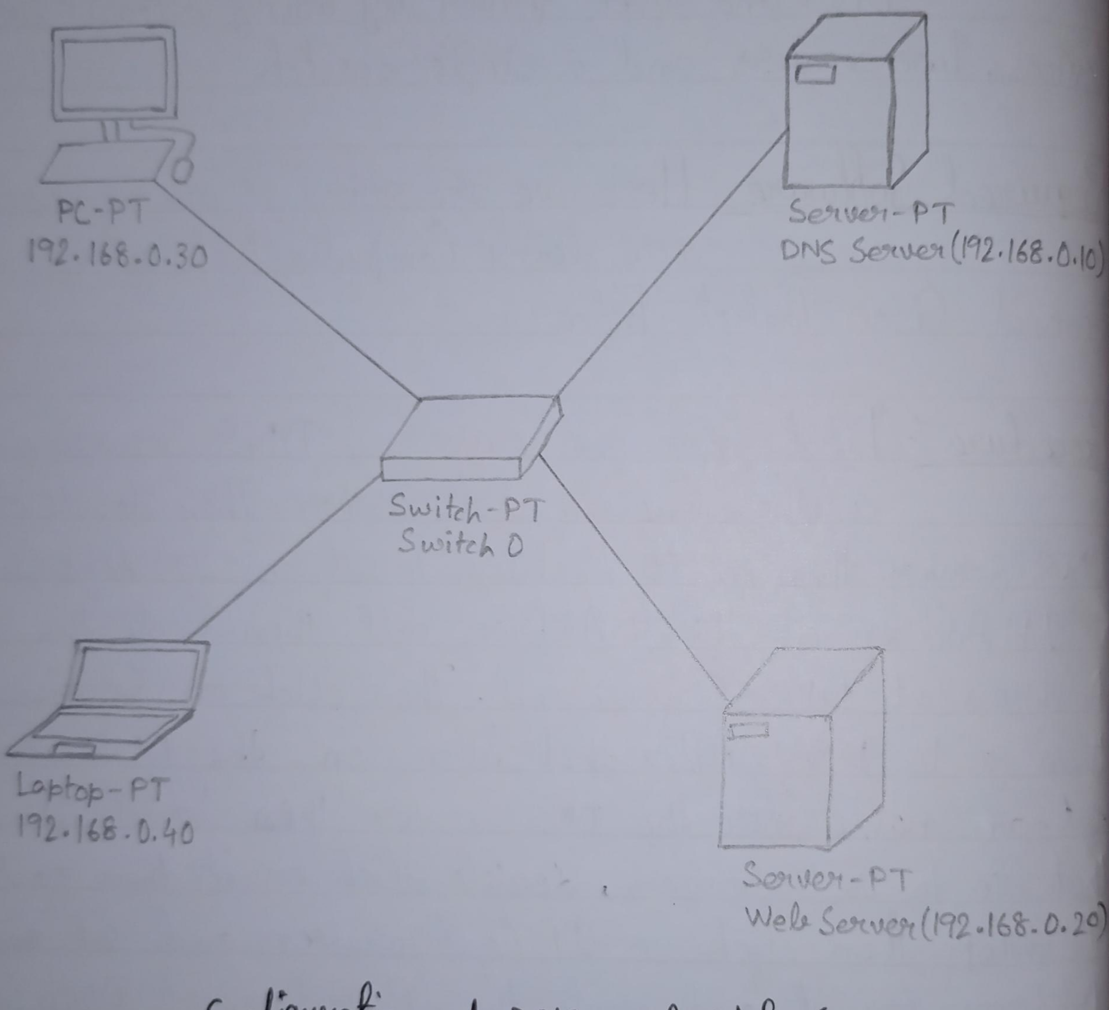
DATE	PAGE NO.	EXPT. NO.
7/9/2024	1	4

Configuration of DNS and Web Server

Objective: Our main goal is to make the 'Configuration of DNS and Web Server' by using atleast two devices, two servers and a single switch.

Required Software: Here we are going to use a software for doing Computer Networking lab with named Cisco Packet Tracer.

Procedure: First name one server as DNS Server and another one as Web Server. Then double click on DNS Server then go to Desktop to set an IP Address. Assign an IP Address at IPv4 Address cell then paste the same IP Address at DNS Server cell. Then click on Services and then go to DNS. After that turn 'on' the DNS Service and set a name for the DNS Server then 'add' it. After that go to Web Server, double click on it then go to 'Desktop' then click on IP Configuration and set an IP Address for it and paste the IP Address of DNS Server here. Then at the devices do the same process, set the different IP Address for the different devices and paste the same IP Address of DNS Server there in that actual place. Then the devices and servers are ready. To test it, double click on any devices then go to Web Browser and enter the name of the DNS Server or the IP Address of



Configuration of DNS and Web Server

the DNS Server at the place of URL. Then click on 'Go'.

Output: The output will be –

Cisco Packet Tracer

Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.

Quick Links:
A small page
Copyrights
Image page
Image

Conclusion: By doing this we will get to know how to 'Configuration of DNS and Web Server'. This will help me for doing other lab projects in future.

Implementation of Router to connect different types of Networks:

Objective: Our main goal is to make connections between the router and different types of networks by using four devices, two switches and one router.

Required Software: Here we are going to use a software for doing Computer Networking lab with named Cisco Packet Tracer.

Procedure: After openning the software, take four devices, two switches and one router. For two devices, they are connected with a single switch and that swich is connected with the router. The same is happening for rest of the devices.

(i) Router 0 → Config → Fast Ethernet 0/0 → IPv4 Address: 192.168.0.10 → On

IPv4 of PCs are 192.168.0.1 and 192.168.0.2 and default gateway is 192.168.0.10.

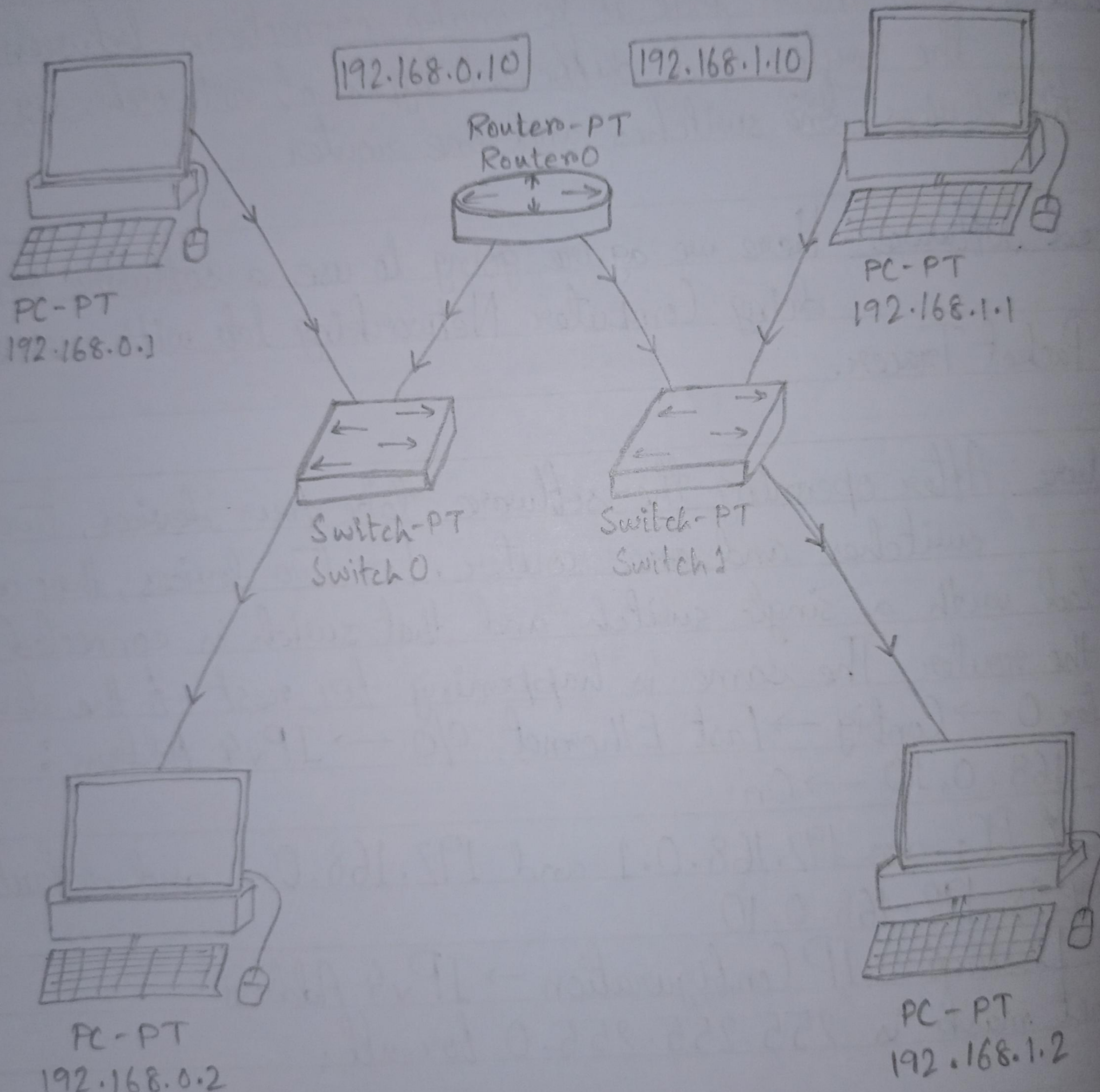
PC → Desktop → IP Configuration → IPv4 Address

Subnet mask is 255.255.255.0 for all.

(ii) Router 0 → Config → Fast Ethernet 1/0 → IPv4 Address: 192.168.1.10 → On

PC → Desktop → IP Configuration → IPv4 Address

192.168.1.1 and 192.168.1.2 for two different PCs.


192.168.0.10
192.168.1.10
Routero-PT
Router0
PC-PT
192.168.0.1
PC-PT
192.168.1.1
Switch-PT
Switch0
Switch-PT
Switch1
PC-PT
192.168.0.2
PC-PT
192.168.1.2

Subnet Mask is 255.255.255.0 for all.

Output : Command prompt of PC(192.168.1.2) —
PC → Desktop → Command Prompt

(i) ping 192.168.0.1
Pinging 192.168.0.1 with 32 bytes of data :
Request timed out.
Reply from 192.168.0.1 : bytes = 32 time < 1ms TTL = 127
Reply from 192.168.0.1 : bytes = 32 time < 1ms TTL = 127
Reply from 192.168.0.1 : bytes = 32 time < 1ms TTL = 127

Ping statistics for 192.168.0.1 :
Packets : Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds :
Minimum = 0ms, Maximum = 0ms, Average = 0ms

(ii) ping 192.168.0.2
Pinging 192.168.0.2 with 32 bytes of data :
Request timed out.
Reply from 192.168.0.2 : bytes = 32 time < 1ms TTL = 127
Reply from 192.168.0.2 : bytes = 32 time < 1ms TTL = 127
Reply from 192.168.0.2 : bytes = 32 time < 1ms TTL = 127

Ping statistics for 192.168.0.2 :
Packets : Sent = 4, Received = 3, Lost = 1 (25% lost),
Approximate round trip times in milli-seconds :

Minimum = 0ms, Maximum = 0ms, Average = 0ms.

(iii) ping 192.168.1.1
Pinging 192.168.1.1 with 32 bytes of data:
Reply from 192.168.1.1: bytes = 32 time < 1ms TTL = 128
Reply from 192.168.1.1: bytes = 32 time < 1ms TTL = 128
Reply from 192.168.1.1: bytes = 32 time < 1ms TTL = 128
Reply from 192.168.1.1: bytes = 32 time < 1ms TTL = 128

Ping statistics for 192.168.1.1:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0 ms, Maximum = 0ms, Average = 0ms.

Conclusion: After completing this assignment, we will get to know about how to connect two different networks, holding different gateways with the help of router and configuring it.

'Parity Bit Method' to encode data at sender side and decoding the data at receiver end for error detection

Sender side :

```
# include <stdio.h>
# include <string.h>
int main () {
    printf ("Encoding the data......\n");
    char binary[100];
    int i, count = 0;
    char parity_choice;
    printf ("Enter a binary number: ");
    scanf ("%s", binary);
    printf ("Choose parity - even(e) or odd(o): ");
    scanf ("%c", &parity_choice);
    for (int i=0; i<strlen(binary); i++) {
        if (binary[i]=='1') {
            count ++;
        }
    }
    if (parity_choice == 'e') {
        if (count %2 ==0) {
            strcat (binary, "0");
        }
```

```
            else {
                strcat(binary, "1");
            }
        }
        else if (parity_choice == 'o') {
            if (count % 2 == 0) {
                strcat(binary, "1");
            }
            else {
                strcat(binary, "0");
            }
        }
        else {
            printf("Invalid parity choice! \n");
            return 1;
        }
        printf("Parity encoded data: %s \n", binary);
```

Receiver side:

```
        printf("Decoding the data.....\n");
        for (int i = 0; i < strlen(binary); i++) {
            if (binary[i] == '1') {
                count++;
            }
        }
```

Expt. No. 6


```
if(parity_choice == 'e'){
    if(count %2 ==0){
        binary[strlen(binary)-1] = '\0';
    }
    else{
        binary[strlen(binary)-1] = '\0';
    }
}
else if(parity_choice == 'o'){
    if(count %2 == 0){
        binary[strlen(binary)-1] = '\0';
    }
    else{
        binary[strlen(binary)-1] = '\0';
    }
}
else{
    printf("Invalid parity choice!\n");
    return 1;
}
printf("Parity decoded data : %s\n", binary);
return 0;
}
```

Teacher's Signature ....

Expt. No. 6

Output :

Encoding the data.....
Enter a binary number : 10011
Choose parity - even (e) or odd (o) : e
Parity encoded data : 100111
Decoding the data.....
Parity decoded data : 10011

DATE 1/11/2024 PAGE NO. 1 EXPT. NO. 7

Checksum Method to encode data at sender side and decoding the data at receiver end for error detection

Sender Side:

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    int a[10], b[10], sum[10], chk[10], carry=0, n, i, p,
    chks[20];
    printf("....Checksum program for two segments only....\n\r");
    printf("\nEnter the number of bits of each segment:");
    scanf("%d", &n);
    printf("\nAt the sender's end....");
    printf("\n\r..............\n\r");
    printf("\n\rEnter the first segment(%d bits one after
    another):\n", n);
    for(i=n-1; i>=0; i--){
        scanf("%d", &a[i]);
    }
    printf("\n\r Enter the second segment:\n");
    for(i=n-1; i>=0; i--){
        scanf("%d", &b[i]);
    }
    for(i=0; i<n; i++){
        sum[i]=(a[i]+b[i]+carry)%2;
```

LOKENATH®

Page No. 2
Date. 1/11/2024

```
        carry = (a[i] + b[i] + carry) / 2;
}
printf("\n sum.....");
for(i=n-1; i>=0; i--){
        printf("%d", sum[i]);
}
if (carry==1){
    for (i=0; i<n; i++){
            p = (sum[i] + carry) % 2;
            carry = (sum[i] + carry) / 2;
            sum[i] = p;
    }
}
printf("\n Wrappedsum.....");
for(i=n-1; i>=0; i--){
        chk[i] = sum[i];
        printf("%d", sum[i]);
}
printf("\n Checksum.....");
for (i=n-1; i>=0; i--){
    if (chk[i]==1){
            chks[i] = 0;
    }
    else {
            chks[i] = 1;
```

Teacher's Signature ....

```
        printf("%d", chks[i]);
    }
  }
  return 0;
}
```

Input and Output:

```
........Checksum program for two segments only..........
Enter the number of bits of each segment : 4
At the sender's end....
-----------------------------------------------
Enter the first segment (4 bits one after another):
1 1 0 1
Enter the second segment:
1 0 1 1
sum.....1000
Wrappedsum.....1001
Checksum......0110
```

Receiver Side:

```
#include<stdio.h>
#include <stdlib.h>
int main () {
    int a[10], b[10], c[20], temp, sum1[10], chkr[20], carry,
    i, n, p;
    printf ("Enter the number of bits in each segments:");
    scanf ("%d", &n);
    printf("\n\r At the receiver's end...");
    carry=0;
    printf ("\n\r Enter the received bits one after another:\n
                                                          \r");
    for (i=n-1; i>=0; i--) {
        scanf ("%d", &a[i]);
    }
    printf ("\n\r");
    for(i=n-1; i>=0; i--){
        scanf ("%d", &b[i]);
    }
    printf("\n\r Next bits will be identified as
    checksum \n\r");
    for(i=n-1; i>=0; i--){
        scanf("%d", &c[i]);
    }
```

```
printf("\n\r.....Calculating Receiver's Checksum.....
                                              \n\r");
for(i=0; i<n; i++){
    sum1[i]=(a[i]+b[i]+c[i]+carry)%2;
    carry=(a[i]+b[i]+c[i]+carry)/2;
}
printf("\n sum.....");
for(i=n-1; i>=0; i--){
    printf("%d", sum1[i]);
    if(carry==1){
        for(i=0; i<n; i++){
            p=(sum1[i]+carry)%2;
            carry=(sum1[i]+carry)/2;
            sum1[i]=p;
        }
    }
}
printf("\n Wrapped sum.....");
for(i=n-1; i>=0; i--){
    chkr[i]=sum1[i];
    printf("%d", sum1[i]);
}
printf("\n Checksum.....");
temp=0;
```

```
for (i=n-1; i>=0; i--) {
    if (chkr[i] == 1) {
        chkr[i]=0;
    }
    else {
        chkr[i] = 1;
        temp = temp + chkr[i];
        printf("%d", chkr[i]);
    }
}
if(temp != 0) {
    printf("\n\r Checksum FAILED, Received data is
    Corrupted.");
}
else {
    printf("\n\r Checksum PASSED.");
}
return 0;
}
```

Input and Output :

Enter the number of bits in each segment : 4
At the RECEIVER's end....
Enter the received bits one after another :
1 1 0 1
1 0 1 1
Next bits will be identified as checksum
0 1 1 0
.........Calculating Receiver's Checksum.......
sum.... 1110
Wrapped sum..... 1111
Checksum..... 0000
Checksum PASSED.

Cyclic Redundancy Check

```
#include<stdio.h>
int main() {
    int m[20], p[20], d[10], i, j, k, len, rem[10], divlen;
    int m1[20], rem1[10];
    printf("\nEnter the length of divisor: ");
    scanf("%d", &divlen);
    printf("\nEnter the divisor : ");
    for(i=0; i<divlen; i++) {
        scanf("%d", &d[i]);
    }
    printf("\nEnter the length of data: ");
    scanf("%d", &len);
    printf("\nEnter the data: ");
    for(i=0; i<len; i++) {
        scanf("%d", &m[i]);
        p[i]=m[i];
    }
    for(i=len; i<len+(divlen-1); i++) {
        m[i]=0;
    }
    printf("\n The append value is: ");
    for(i=0; i<len+(divlen-1); i++) {
        printf("%d", m[i]);
```

```
    }
//*****X-OR operation*****//
for(i=0; i<len; i++) {
    if(m[i]==1) {
        for(j=0; j<divlen; j++) {
            if(m[i+j]==d[j]) {
                rem[j]=0;
            }
            else {
                rem[j]=1;
            }
            for(k=0; k<divlen; k++) {
                m[i+k]=rem[k];
            }
        }
    }
    else {
        for(k=0; k<divlen; k++) {
            rem[k]=m[i+k];
        }
    }
}
printf("\n");
printf("\n CRC = ");
for(i=1; i<divlen; i++) {
```

```
        printf("%d", rem[i]);
}
printf("\n The complete data + CRC received is: ");
for(i=0; i<len+(divlen-1); i++) {
    scanf("%d", &m1[i]);
}
for (i=0; i<len; i++) {
    if(m1[i]==1) {
        for(j=0; j<divlen; j++) {
            if(m1[i+j] == d[j]) {
                rem1[j]=0;
            }
            else {
                rem1[j]=1;
            }
            for(k=0; k<divlen; k++) {
                m1[i+k] = rem1[k];
            }
        }
    }
    else {
        for(k=0; k<divlen; k++) {
            rem1[k] = rem1[i+k];
        }
    }
```

```
}
printf("\n");
printf("\n CRC = ");
j=0;
for(i=1; i<divlen; i++) {
    printf("%d", rem1[i]);
    j=j+rem1[i];
}
printf("\n");
printf("\n CRC = ");
j=0;
for(i=1; i<divlen; i++) {
    printf("%d", rem1[i]);
    j=j+rem1[i];
}
if(j==0) {
    printf("\n There is no error. \n");
}
else {
    printf("\n There is error \n");
}
return 0;
}
```

Input and Output

Enter the length of divisor : 4
Enter the divisor : 1 1 0 1
Enter the length of data : 8
Enter the data : 1 0 1 0 0 0 1 0
The append value is : 10100010000
CRC = 010
The complete data + CRC received is :
1 0 1 0 0 0 1 0 0 1 0
CRC = 440
CRC = 440
There is error