Experiment 4

Student Name: Divyanshu Pandey **UID: 22BCS10270**
Branch: CSE **Section/Group: DC-701 A**
Semester: 5th **Date of Performance:07-08-2024**
Subject Name: Computer Networks **Subject Code:22CSH-312**

1. Aim: Configure and Understand the working of network devices Hub, Switch, Routers

2. Objective: Students will understand working of network devices

3. Software Requirement:

- Processor – Any suitable Processor e.g. Celeron
- Main Memory - 128 MB RAM
- Hard Disk – minimum 20 GB IDE Hard Disk
- Removable Drives–1.44 MB Floppy Disk Drive –52X IDE CD-ROM Drive
- PS/2 HCL Keyboard and Mouse

4. Apparatus Used: Packet Tracer.

5. Theory:

Network devices are hardware components that facilitate communication and data transfer between different devices within a computer network. They play crucial roles in the construction, maintenance, and management of networks, ensuring that data can move efficiently and securely from one point to another.

- **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 signals to every port except the port from where the 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 the 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 an 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:** Mesh topology is a network setup where each device is interconnected with every other device in the network. This can be either a full mesh, where every device is connected to every other device, or a partial mesh, where some devices are connected to others, but not all.

6. Procedure:

Step 1: Setting Up the Network ✞

Open Cisco Packet Tracer.

Step 2: Configuring a Hub Network

- From the device toolbar, drag a hub onto the workspace. ✞ You can find the hub under "Network Devices" > "Hubs." ✞ Drag and drop three or more PCs onto the workspace.
- You can find the PCs under "End Devices."
- Use the "Copper Straight-Through" cable to connect each PC to the hub.
- Click on each PC, select the "FastEthernet" port, and then connect to an available port on the hub.
- Click on each PC, go to the "Desktop" tab, and open the "IP Configuration" tool.
- Assign each PC an IP address in the same subnet (e.g., 192.168.1.2, 192.168.1.3, etc.).
- Use the "Command Prompt" on one of the PCs to ping the other PCs in the network.
- Observe how the hub broadcasts the data to all connected devices.

Step 3: Configuring a Switch Network

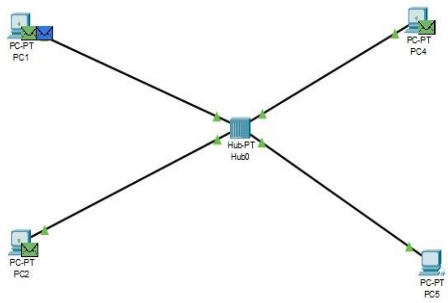
- Drag a switch onto the workspace from "Network Devices" > "Switches." ✞ Add the same number of PCs as in the hub setup.
- Use the "Copper Straight-Through" cable to connect each PC to the switch.
- Use the same IP addresses as in the hub network.
- Use the "Command Prompt" on one of the PCs to ping the other PCs in the network.
- Observe how the switch only forwards the data to the intended recipient, reducing unnecessary traffic.

Step 4: Configuring a Router Network

- Drag a router onto the workspace from "Network Devices" > "Routers."
- Add two switches to the workspace and connect them to different interfaces on the router.
- Add PCs to each switch, creating two distinct subnets.
- Use the "Copper Straight-Through" cable to connect each PC to the corresponding switch.
- Click on the router, go to the "CLI" tab, and configure the IP addresses for each interface connected to the switches.
- Assign IP addresses to PCs in each subnet (e.g., 192.168.1.2/24 for subnet 1, 192.168.2.2/24 for subnet 2).
- Use the "Command Prompt" on one of the PCs in subnet 1 to ping a PC in subnet 2.
- Observe how the router routes the traffic between the two subnets.

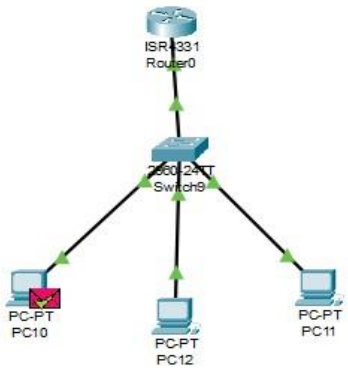
7. Output:

i) Configuring a Hub Network



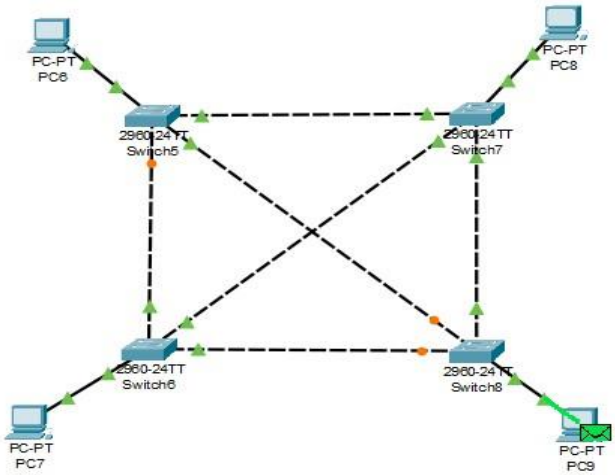
Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC1	PC5	ICMP		0.000	N	0	(edit)	

ii) Configuring a Router Network



Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC10	PC11	ICMP		0.000	N	0	(edit)	

iii) Configuring a Switch Network



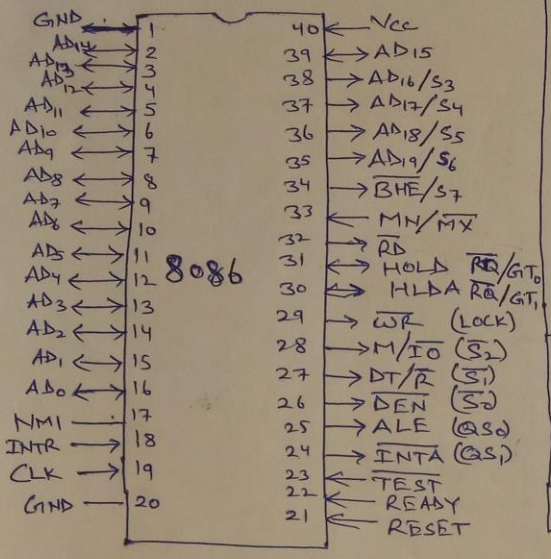
Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC6	PC9	ICMP		0.000	N	0	(edit)	

8. Learning Outcomes:

Understand different network topologies (Star, Bus, Mesh) and their practical applications.

- Gain hands-on experience with network configuration using Cisco Packet Tracer.
- Comprehend the roles and functions of network devices like hubs, switches, and routers.
- Analyze how data is transmitted across various network topologies and devices.
- Develop network design skills for optimizing performance based on specific requirements.
- Enhance problem-solving and troubleshooting skills for network connectivity issues.
- Prepare for advanced networking concepts like VLANs, subnetting, and network security.
- Apply theoretical networking knowledge to real-world scenarios in a simulated environment.
- Understand the difference between broadcasting, switching, and routing in network communication.

8086 Pin Diagram



AD_0–AD_{15} (Bidirectional)

Address/Data Bus

⇒ Low order address Bus; these are multiplexed with data.

⇒ When AD lines are used to transmit memory Address, the symbol A is used instead of AD, for example A_0–A_{15}

⇒ When data are transmitted over AD lines, the symbol D is used in place of AD, for example D_0–D_7, D_8–D_{15} or D_0–D_{15}.

A_{16}/S_3, A_{17}/S_4, A_{18}/S_5, A_{19}/S_6

High Order Address Bus. These are multiplexed with status signals

⇒ **$\overline{BHE}/S_7$ (Bus High Enable)**: It is used to enable data onto the most significant half of data Bus, D_8–D_{15}. 8-bit device connected to upper half of the data bus use $\overline{BHE}$ (Active low) signal. It is multiplexed with status signal S_7.

⇒ **MN/$\overline{MX}$ (MINIMUM/MAXIMUM)**: This PIN signal indicates, what mode the processor is to operate in.

⇒ **$\overline{TEST}$**: $\overline{TEST}$ I/P is tested by 'WAIT' instruction. 8086 will enter a wait state after execution of the WAIT instruction. and will resume execution only when the $\overline{TEST}$ is made low by an active hardware. This is used to synchronize an external activity to the processor internal operation

⇒ **READY**: This is acknowledgement from the slow device or memory that they have completed the data transfer. The signal made available by the devices is synchronized by the 8284A clock generator to provide ready input to the 8086.

⇒ **RESET**: Causes the processor to immediately terminate its present activity. The signal must be active HIGH for at least four clock cycles.

⇒ **CLK**: The clock I/P provides the basic timing for processor operation & BUS control activity. Its an asymmetric square wave with 33% duty cycle.

⇒ **INTR Interrupt Request**: This is a triggered I/p. This is sampled during the last clock cycles of each instruction to determine the availability of the request. If any interrupt request is pending, the processor enters the interrupt acknowledge cycle.

⇒ 8086 microprocessor can work in two modes of operation: (Minimum Mode & Maximum Mode)

→ In minimum mode of operation the microprocessor do not associate with any co-processors and can not be used for Multiprocessor system.

→ In the maximum mode, the 8086 can work in multi-processor or co-processor configuration.

→ Minimum or Maximum mode operation decided by MN/$\overline{MX}$.

Minimum mode signals

For minimum mode operation, the MN/$\overline{MX}$ is tied to V_{cc}. 8086 itself generates all the bus control signals.

DT/$\overline{R}$ (Data Transmit/Receive): This is output signal from the processor to control the direction of data flow through the data transceivers.

DEN (Data Enable): This is o/p signal from the processor used as output enable for the transceivers.

ALE (Addre Latch Enable): used to demultiplex the address and data lines using external latches.

M/$\overline{IO}$: used to differentiate memory access and I/O access. For memory reference instructions, it is high. For IN & OUT instructions it is low.

$\overline{WR}$ (write Control signal): This is low whenever processor writes data to memory or I/O port

$\overline{INTA}$ (Interrupt Acknowledge): when the interrupt request is accepted by the processor, the output is low on this line.

HOLD: This is Output signal to the processor from the Bus masters as a request to grant the control of the bus. It is usually used by the DMA Controller to get the control of the bus.

HLDA (Hold Acknowledge): Acknowledge signal by the processor to the bus master requesting the control of the bus through HOLD. The acknowledge is asserted high, when the processor accepts hold.

Maximum Mode Signals:

During Maximum mode operation, the MN/$\overline{MX}$ is grounded.

⇒ $\overline{S_0}, \overline{S_1}, \overline{S_2}$ Status signals: used by the 8086 bus controller to generate bus timing and control signals. These are decoded as shown.

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	Machine Cycle
0	0	0	Interrupt acknowledge
0	0	1	Read I/O port
0	1	0	Write I/O port
0	1	1	Halt
1	0	0	Code access
1	0	1	Read memory
1	1	0	Write Memory
1	1	1	Passive/Inactive

⇒ $\overline{QS_0}, \overline{QS_1}$ (Queue status): The processor provides the status of Queue in these lines. The Queue status can be used by external device to track the internal status of the Queue in 8086. The output on QS_0 and QS_1 can be interpreted as shown in the table.

QS_1	QS_0	Queue operation
0	0	No operation
0	1	First byte of an Opcode from Queue
1	0	Empty the Queue
1	1	Subsequent byte from Queue

⇒ $\overline{RQ}/\overline{GT_0}, \overline{RQ}/\overline{GT_1}$ (Bus Request/Bus Grant): These requests are used by other local Bus masters to force the processor to release the local bus at the end of the processors current bus cycle.

→ These PINs are bidirectional.

→ The requests on $\overline{GT_0}$ will have higher prority than $\overline{GT_1}$.

⇒ $\overline{LOCK}$: ⇒ An o/p signal activated by the LOCK prefix Inst.

→ Remains active until the completion of the Instruction prefixed by LOCK.

→ The 8086 output low on the $\overline{LOCK}$ PIN while executing an instruction prefixed by Lock to prevent other Bus masters from gaining control of the system bus.

→ When it is low, all the interrupts are masked & no HOLD request is granted. In a multiprocessor system all other processors are informed by the signal that they should not ask the CPU for relinquishing the bus control.

Register Organization of 8086

Powerful Set of Register

- General purpose Register
- Special purpose Register

⇒ All of them are 16-bit.

⇒ The General purpose register can be used as either 8-bit or 16-bit registers. They are used for holding data, variables and or for other purpose like counters or for storing offset address for some particular addressing modes.

⇒ The special purpose registers are used as segment registers, pointers, index registers or as offset registers for particular addressing modes.

(1) General Data Registers :

AX Register: known as accumulator register that stores operands for arithmatic operation like divided, rotate.

BX Register: This register is mainly used as a base register. It holds the starting base location of a memory region within a data segment.

CX Register: It is defined as a Counter. It is primarily used in loop instruction to store loop counter.

DX Register: Dx register is used to contain I/o port address for I/o instructions.

	8-bits	8-bits
AX	AH	AL
BX	BH	BL
CX	CH	CL
DX	DH	DL

General data registers

(2) Segment Registers :

Additional registers called segment registers generate memory address when combined with other offset address. It contains the starting address or segment address (16-bit) of memory location in a particular segment.

CS	CS Register
SS	SS Register
DS	DS Register
ES	ES Register

segment Registers

Code Segment Register: The CS Register is used for addressing a memory location in the code segment of memory where executable program is stored.

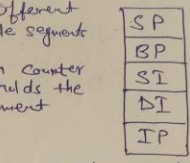
Stack Segment Register: The SS Register used for addressing stack segment of memory which used to store stack data.

Data Segment Register: The data segment register is used to address data segment of memory in which data is resided.

Extra Segment Register: The ES register is used to address Extra segment which is mostly used for string operations

③ **Pointers & Index Registers :**

The segments of memory are accessed by different pointers, no single pointer can access multiple segment at the same time.



Pointers & Index Registers

(i) The instruction pointer IP (similar to program counter in 8085) is used to fetch the code which holds the offset address and the starting of the segment address is held by the CS register.

(ii) The DI/Bx/SI register holds the offset address of the Data segment along with the DS register.

(iii) Similarly, the SP/BP has offset address of stack segment & starting address is held by SS register.

(iv) The DI holds the offset address of Extra segment and starting address is held by ES register.

④ **Flag Register** : 8086 Flag register contents indicate the results of computations in the ALU. It also contains some flag bits to control the CPU operations.

⇒ While addressing any location in memory Bank, the physical address is calculated from two parts

(i) Segment address.
(ii) offset address.