Experiment No: 1

Experiment Name: Design and implement Amplitude Shift Keying (ASK) modulation technique.

Objectives:

i) To understand the operation theory of Amplitude Shift Keying (ASK) modulation.

ii) To understand the signal waveform of the ASK modulation.

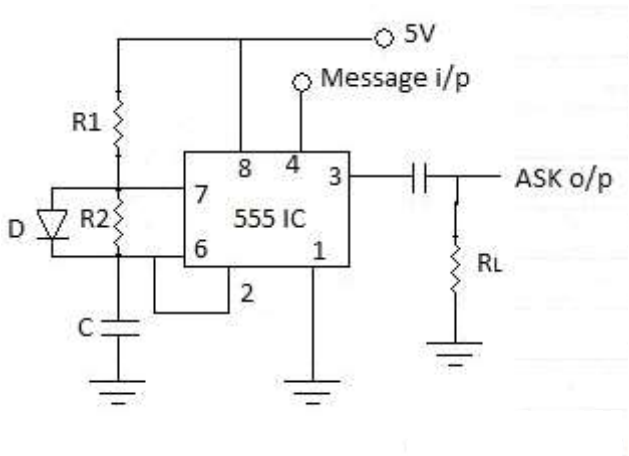
Description:

Modulation: The most important and interesting concept in communication is Modulation. It has different types. Modulation is defined as improving the signal characteristics amplitude, frequency, or phase concerning the carrier signal. If the input signal is analog, then such modulation is called analog modulation. And if the input signal is in the form of digital, such modulation is called Digital modulation.

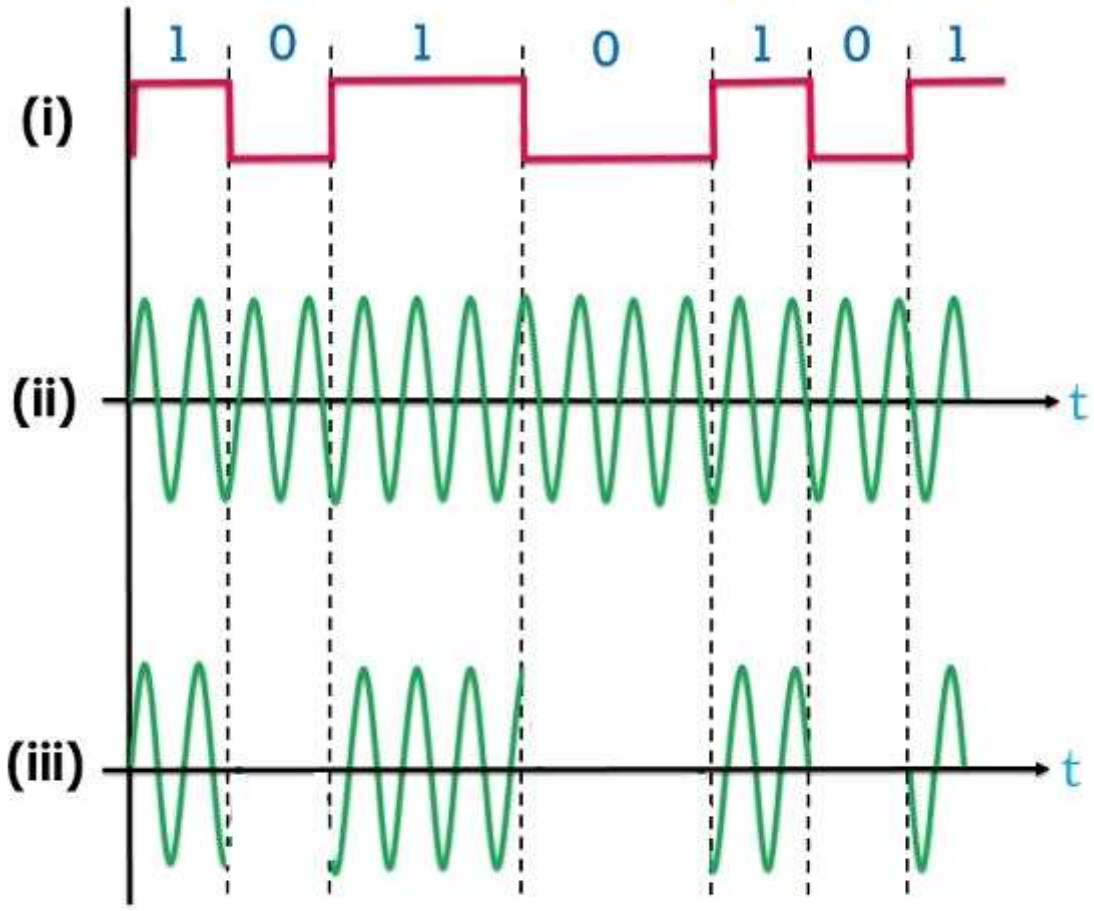
ASK: This type of modulation comes under Digital Modulation schemes. Here, the word keying has some importance, i.e. Keying is indicating the transmission of digital signal over the channel. By the amplitude shift keying theory, we can understand the process of the ASK technique. In ASK, it requires two input signals, First input is a binary sequence signal and the second input is a carrier signal. Here the most important point we need to always consider is the second input which is the carrier signal has a more amplitude/voltage range than the input binary sequence signal.

Figure:

Block diagram:



Output waveform:



(i) = Digital bit sequence
(ii) = Carrier wave
(iii) = ASK modulated wave

Result:

The carrier generator, sends a continuous high-frequency carrier. The binary sequence from the message signal makes the unipolar input to be either High or Low. The high signal closes the switch, allowing a carrier wave. Hence, the output will be the carrier signal at high input.

Conclusion:

Thus we can conclude that by using the ASK technique, digital data can be transmitted by varying only the amplitude factor of the carrier wave.

Experiment No: 2

Experiment Name: Design and implement Frequency Shift Keying (FSK) modulation technique.

Objectives:

• To describe the FSK modulation and demodulation (Frequency Shift Keying).

• To carry out an FSK connection.

• To examine the noise effect on the connection.

Materials:

• Power unit PSU.

• Module holder base.

• Individual Control Unit SIS1.

• Experiment module MCM31.

• Oscilloscope.

Description:

Frequency Shift Keying (FSK):

Frequency shift keying is defined as a signaling technique in which the amplitude of the carrier signal is keyed or switched based on the incoming data or signal.

In digital data communication, shifting a carrier frequency between two preset frequencies

transmitted binary code. This type of transmission is called the frequency shift keying (FSK) technique. A555 timer in astable mode can be used to generate an FSK signal. The standard digital data input frequency is 150Hz.when input is HIGH, the transistor Q is off and the 555 timer works in the normal astable mode of operation.

Procedure:

1. The circuit connections are made as shown in the figure.

2. The free running frequency of the astable multivibratoris measured using CRO.

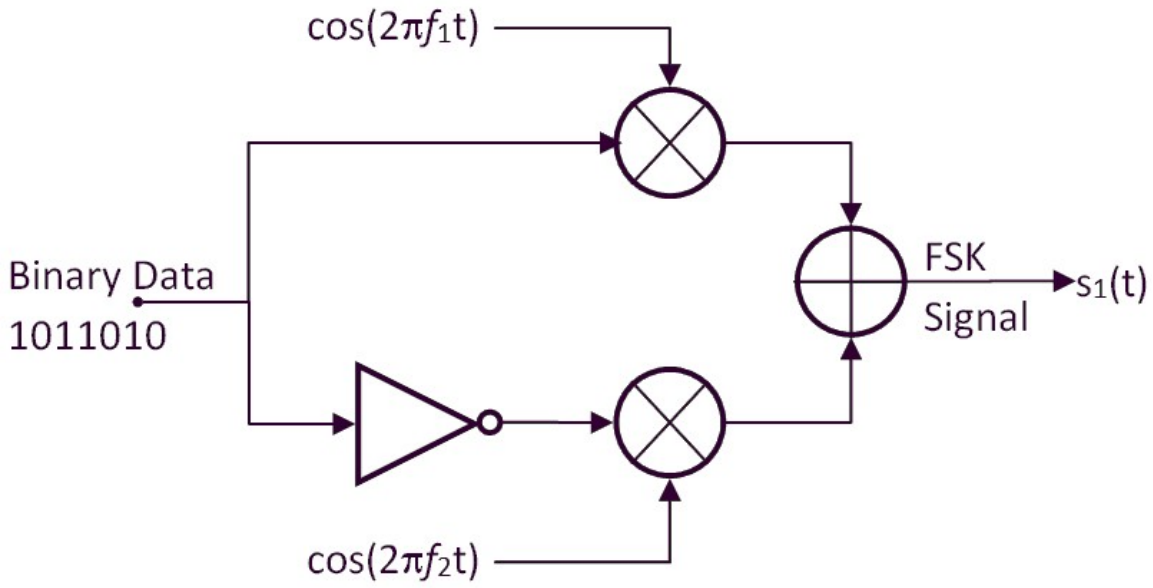
3. The input square wave (digital data) is given from the AFO.

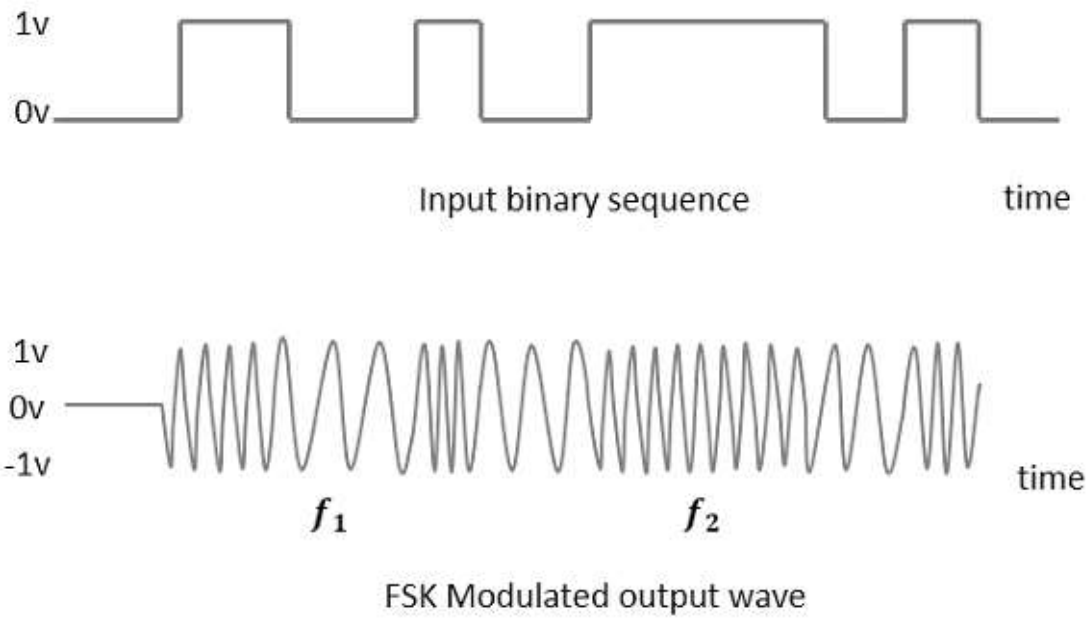
4. The FSK waveform is noted from the CRO and plotted.

Generation of FSK:

The FSK signal can be generated by applying the incoming binary data to a frequency modulator and to other input a sinusoidal carrier wave of amplitude AC and frequency fC is applied. As the binary data changes from one level to another (but non-zero being pear) the output changes its frequencies is the corresponding manner.

Figure:





FSK Modulated output wave

Result:

Hence generated frequency modulated and demodulated signals.

Conclusion:

By the FSK modulation technique we can achieve error-free communication in a few digital applications. But this FSK has a finite data rate and consumes more bandwidth can be overcome by the QAM, which is known as quadrature amplitude modulation.

Experiment No: 4

Experiment Name: Design and develop 4 to 1 Multiplexer.

Objectives:

1. To determine the 4-to-1 multiplexer.

2. To learn the working principle of the multiplexers.

3. To use a multiplexer.

Description:

The multiplexer is one of the basic building blocks of any digital design system. A digital multiplexer or data selector is a logic circuit that accepts several inputs and selects one of them at any given time to pass on the output. 4 to 1 multiplexer that accepts 4 inputs and produces one output. Selects input provides which will pass to the output. The 74LS153 is a very high-speed dual 4-input multiplexer with common select input and individual enable input for each selection. It can select two bits of data from the source.

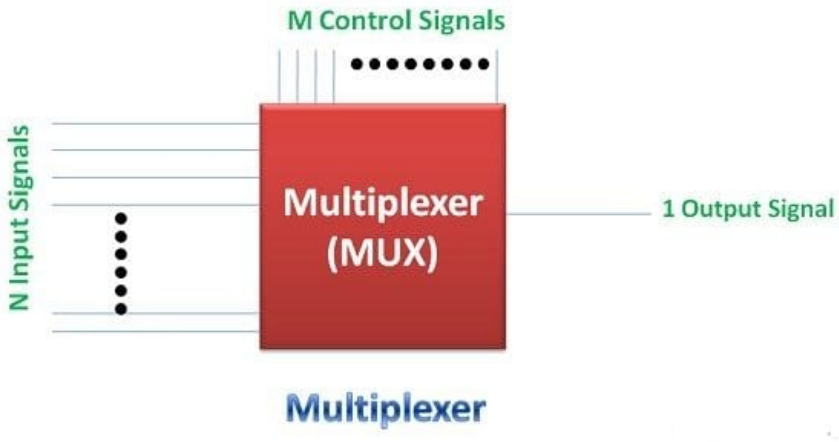
Equipment and apparatus:

1. IC74LS153

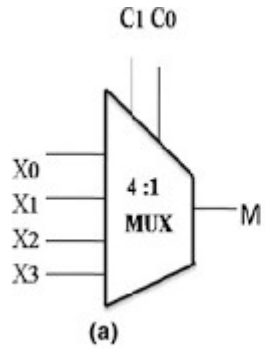
2. Project Board

3. Connecting wire

Figure:



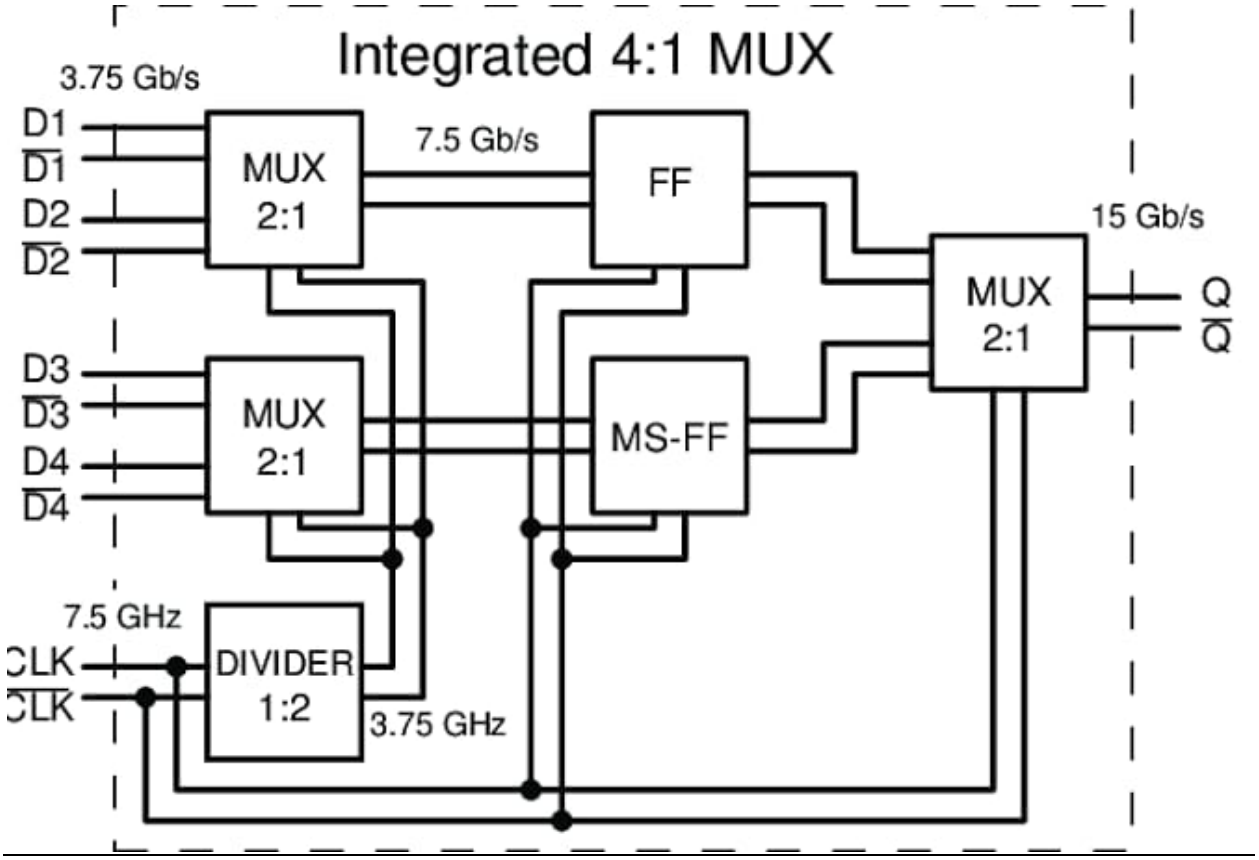
Pin Diagram:



C1	C0	M
0	0	X0
0	1	X1
1	0	X2
1	1	X3

(b)

Circuit diagram:



Procedure:

1. Multiplexing is the process in which multiple data streams, come from different 2. sources are combined and transmitted over a single Data channel or Data Stream.

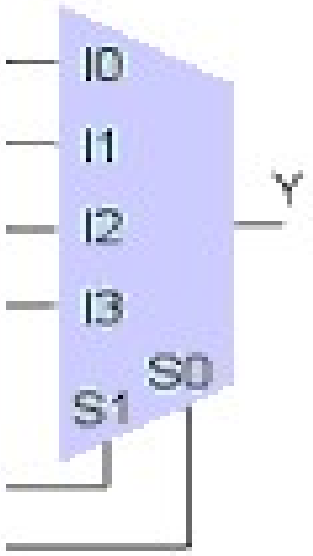
2. Give biasing to the input and do not need a connection.

3. Give various combinations of input.

Result:

S1	S0	Y
0	0	I0
0	1	I1
1	0	I2
1	1	I3

S1 \ S0	0	1
0	I0	I1
1	I2	I3



Conclusion:

A multiplexer, or MUX, is a circuit that selects a single output from multiple inputs. It has multiple uses. The main use of a MUX is to select between input values, such as input values to the ALU in a CPU. But it can also be used to implement logic where the select lines are the inputs to a function, and the outputs to the function are hardwired to input values for the MUX.

Experiment No: 06

Experiment Name: Implement Hamming code generation and error detection.

Objectives:

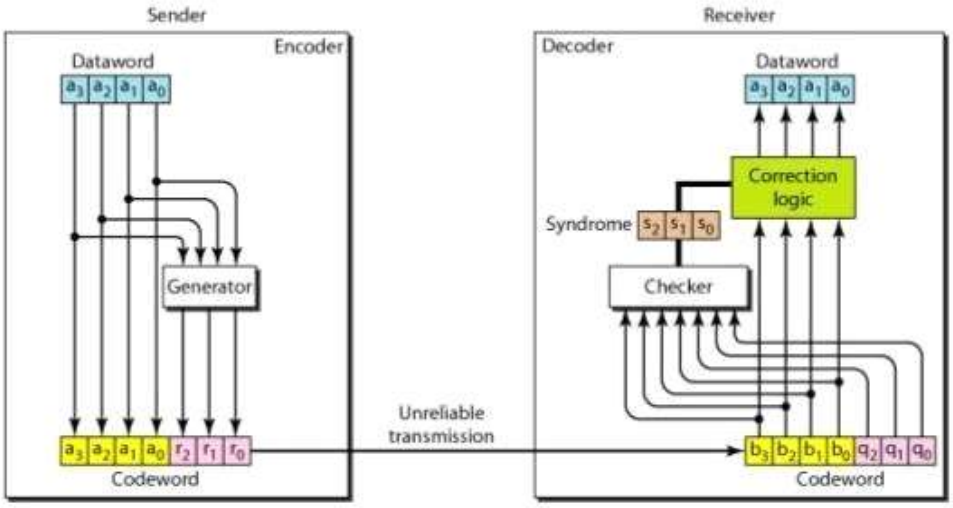
i) The main objective is detecting single-bit error burst errors and correcting them using a suitable methodology.

ii) The purpose of this project is to overcome the unreliable transmission of data during the transmission from the encoder to the decoder.

Description:

The theory of error detecting and correcting codes is that branch of engineering and mathematics that deals with the reliable transmission and storage of data. Information media are not 100% reliable in practice, in the sense that noise (any form of interference) frequently causes data to be distorted. To deal with this undesirable but inevitable situation, self-correcting codes are constructed by combining data bits with several redundant check-bits, which together are referred to as code words. By adding a single bit (as in Parity Checking) along with a given number of data bits, half of the possible code words become valid and half invalid. This is referred to as a code that has Hamming Distance 2 (two single-bit flips are required to get from one valid code to another). This particular approach, using Hamming Distance, is the minimum requirement for self-error correction and is commonly referred to as a Hamming Code.

Figure:



Hamming code implementation in Cpp:

```
#include <iostream>
#include <string>
#include <algorithm>
using namespace std;
class hamming{
	public:
		string data;
		int m, r = 0; redundant bits
		char * msg;
	hamming(string data){
			this->data = data;
			reverse(data.begin(),data.end());
			m = data.size();
			int power = 1;
			while(power < (m + r + 1)){
				r++;
			power*=2;
			}
			msg = new char[m+r+1];
			intcurr= 0;
			for(inti = 1 ; i<= m+r ; i++){
				if(i& (i-1)){
					msg[i] = data[curr++];
				}
				elsemsg[i] = 'n';
			}
			setRedundantBits();
		}
		void show msg(){
			cout<< "the data packet to be sent is: ";
			for(inti = m+r ; i>= 1 ; i--){
				cout<<msg[i] << " ";
			}
			cout<<endl;
		}
		voidsetRedundantBits(){
			int bit = 0;
			for(inti = 1 ; i<= m+r ; i*=2){
				int count = 0;

				for(int j = i+1 ; j<=m+r ; j++){
					redundant bit or not using bit manipulation
					if(j & (1 << bit)){
						if(msg[j] == '1') count++;
}
				}
				if(count & 1) msg[i] = '1';
				elsemsg[i] = '0';
				bit++;
			}
			showmsg();
		}
		void receiver(){
```

```
                string and = "";
                int bit = 0;
        for(inti = 1 ; i<= m+r ; i*=2){
                        int count = 0;
                        for(int j = i+1 ; j<=m+r ; j++){
                                if(j & (1 << bit)){
                                        if(msg[j] == '1') count++;
                                }
                        }

                if(count & 1){
                                if(msg[i] == '1') ans.push_back('0');
                                elseans.push_back('1');
                        }
                        else{
                                if(msg[i]=='0') ans.push_back('0');
                                elseans.push_back('1');
                        }
                        bit++;
                }
        if(ans.find('1') != string::npos){
                        int power = 1;
                        intwrongbit = 0;
                        for(inti = 0 ; i<ans.size() ; i++)
                                if(ans[i]=='1') wrongbit+=power;
                                power*=2;
                        }
    cout<< "bit number " <<wrongbit<< " is wrong and having error " <<endl;
                }
                else{
                        cout<< "correct data packet received " <<endl;
                }
        }
};
int main(){
        string data = "1011001";
        hamming h(data);

        //h.msg[i] == '0' ? h.msg[i] = '1' : h.msg[i] = '0';

        h.receiver();
        return 0;
}
```

Result:

The data packet to be sent is: 1 0 1 0 1 0 0 1 1 1 0
a correct data packet received.

Conclusion:

We have discussed a lot about the Hamming codes and now we can conclude that these codes are capable of correcting single-bit errors on messages of any length. Although this code can detect two-bit errors, we are unable to get the error locations.

Experiment No: 09

Experiment Name: Design and implement star and tree topology.

Objectives:

i) To design network topologies for networking systems.

ii) Implement the proper topologies for the proper network.

iii) To get knowledge about topologies in various fields.

Description:

Star Topology: Star topology is a network topology in which each network component is physically connected to a central node such as a router, hub, or switch. The central hub acts like a server in a star topology, and the connecting nodes act like clients. When the central node receives a packet from a connecting node, it can pass it on to other network nodes. A star topology is also known as a star network.

Star networks require a point-to-point connection between the central node and connecting devices. The central node can provide signal reconditioning and amplification services to improve communication between the devices on the network. Star topologies are often used in home networks. The benefits of a star network topology include the following:

Limits the impact of a single point of failure. In star networks, each connecting node is isolated from other connecting nodes. If one connecting node goes down, it will not impact the performance of other connecting nodes in the network.

Facilitates adding or removing individual components to and from a network. Star networks are usually kept small because network performance can suffer when too many devices compete for access to the central node.

Tree Topology: A tree topology is a type of network topology that includes at least three specific levels in a topology hierarchy. Tree topologies are valued for their scalability and accessibility for troubleshooting.

There are numerous ways to explain the structure of tree topology, as follows:

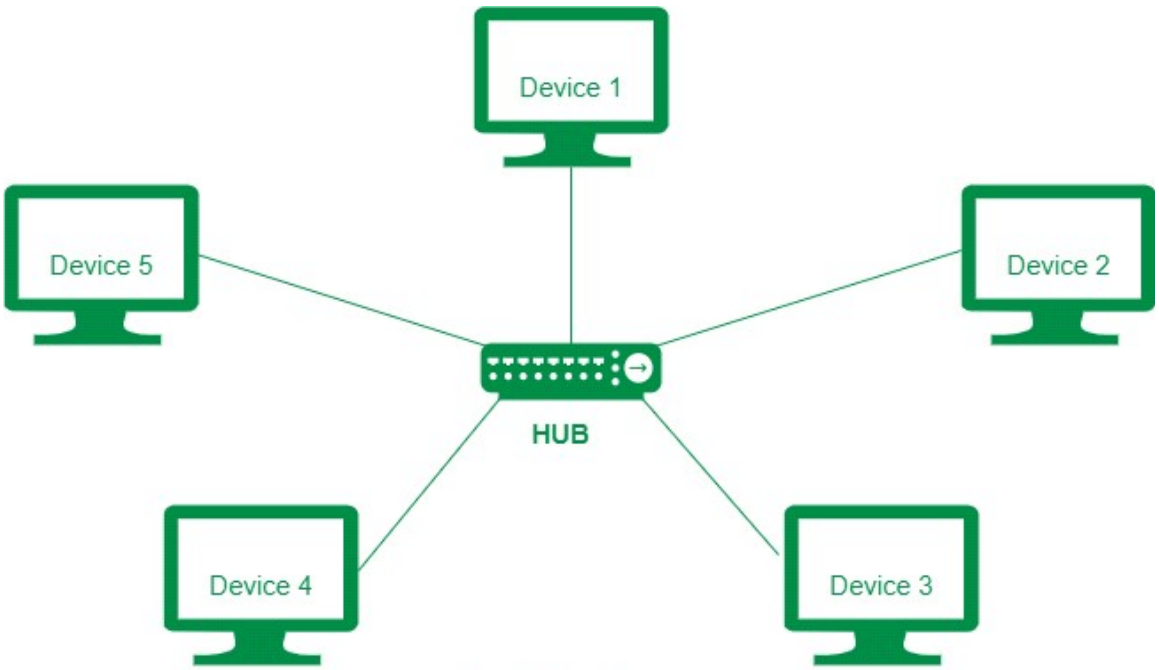
- A tree topology includes multiple star topologies, which involve a variety of single nodes connected to a central node. Multiple stars involve either a series or tertiary nodes attached to two or more secondary nodes, which are attached to the tree's primary trunk node.
- Experts may define a tree topology as a combination of star and bus topologies, where multiple elements are connected through a single lateral connection.
- Each node in a hierarchy level has point-to-point links with each adjacent node on its below level. All secondary nodes have point-to-point attachments to the tertiary nodes in their jurisdiction, and the primary node has a point-to-point connection to each secondary node.

When viewed visually, these systems appear similar to a tree structure.

A drawback of a tree topology is that an entire system can be crippled by any damage or mal function of the primary node. This is why managers of tree topologies often have a "protect the tree" approach, where the primary node receives special attention or safeguards.

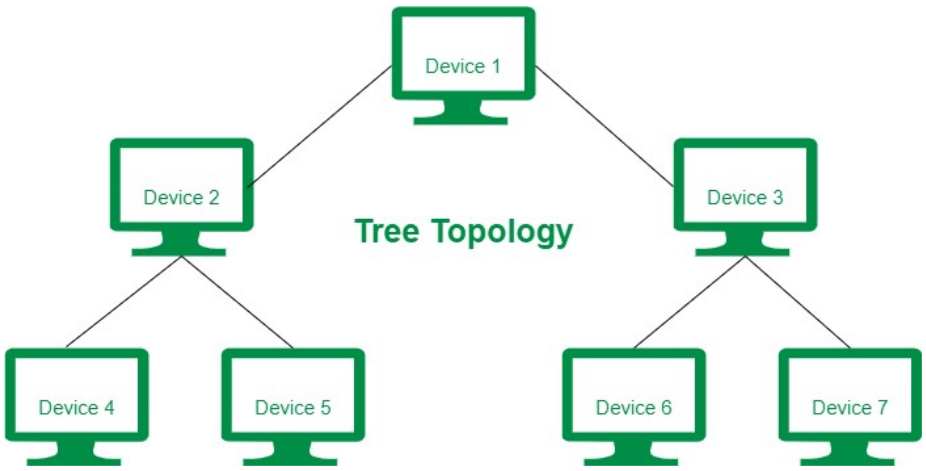
Figure:

Star Topology:



Star Topology

Tree Topology:



Result:

Command: "ping ip_address_of _any_device"

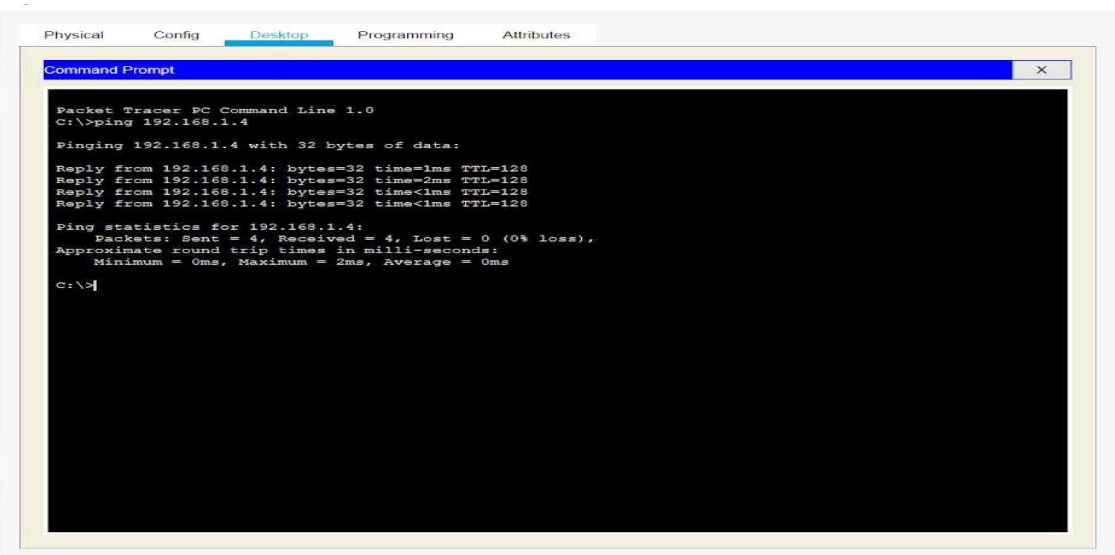
Example:

ping 192.168.1.4

Note: If the connections are correct

then you will receive the response.

Output:



Conclusion:

Star Topology:

A Star Network Topology is best suited for smaller networks and works efficiently when there is a limited number of nodes. One has to ensure that the hub or the central node is always working and extra security features should be added to the hub because it s the heart of the network.

Tree Topology:

Tree network topology is considered to be the simplest topology in all the topologies which is having only one route between any two nodes on the network. The pattern of connection resembles a tree in which all branches spring from one root hence (Tree Topology).