

# Data Communications

Prakash Poudyal

Department of Computer Science and Engineering

Kathmandu University

# Data Communication

- Are the exchange of the data between two devices through some form of transform medium such as a wire cable.
- Depend upon four fundamental characteristics:
  - **Delivery** : Deliver data to the correct destination.
  - **Accuracy** : The system must deliver the data accurately
  - **Timeliness** : time is matter
  - **Jitter** : Jitter refers to the variation in the packet arrival time.
    - Eg : let us assume that video packets are sent every 30ms
    - Others are sent 30-ms delay and other 40ms delay than quality of video decreases

# Data Communication Systems Components

- Message :
  - Message is the information (text), numbers, pictures, audio and video
- Sender
  - The sender is the device that sends the data message . Eg computer, workstation, camera
- Receiver
  - The receiver is the device that receives the message eg computer, television
- Transmission medium
  - Physical path : eg twisted-pair wire, coaxial cable, fiber-optic cable
- Protocol
  - A protocol is a set of rules that governs data communication. It represents an agreement between the communication devices. Without protocol, **two devices may be connected but not communicating.**  
Eg. Person speaking French cannot understood by a person who speaks Japanese only

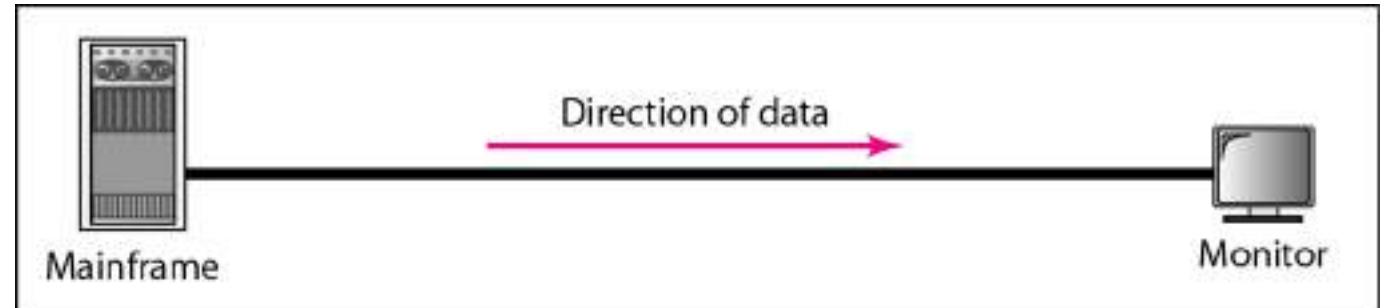
# Data Representation

- Text
  - Represented through a bit pattern, a sequence of bits (0s or 1s).
  - Different sets of bit patterns have been designed to represent texts symbols.
  - Today, the coding system is called Unicode, which uses 32 bits to represent a symbol or characters.
  - American Standard Code for Information Interchange (ASCII)
    - 127 characters in Unicode.
- Numbers
  - Numbers are also representation by bits patterns.
- Images
  - Represented by bit patterns
  - Image is composed of a matrix of pixels, where each pixel is a small dot.
  - Black pixel is 00, dark gray pixel 01, light gray pixel by 10 and white pixel by 11
- Audio
  - Recording or broadcasting or sound or music
- Video
  - Recording or broadcasting of pictures or movie.

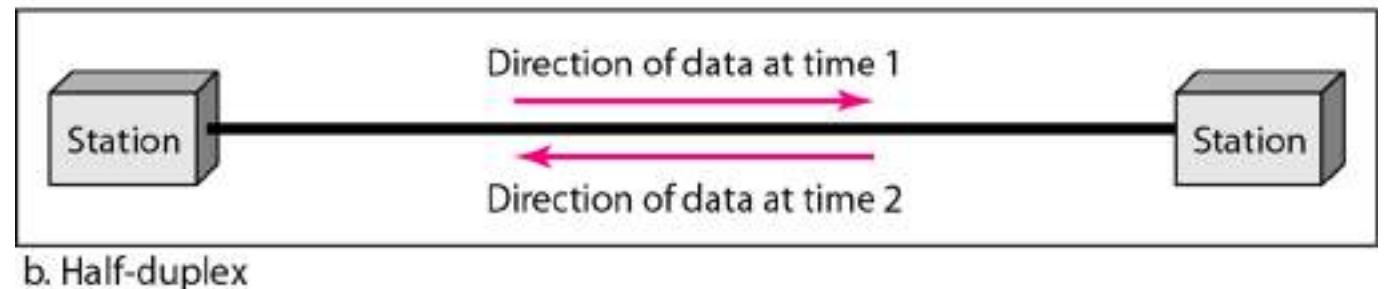
# Data Flow

- **Simplex**

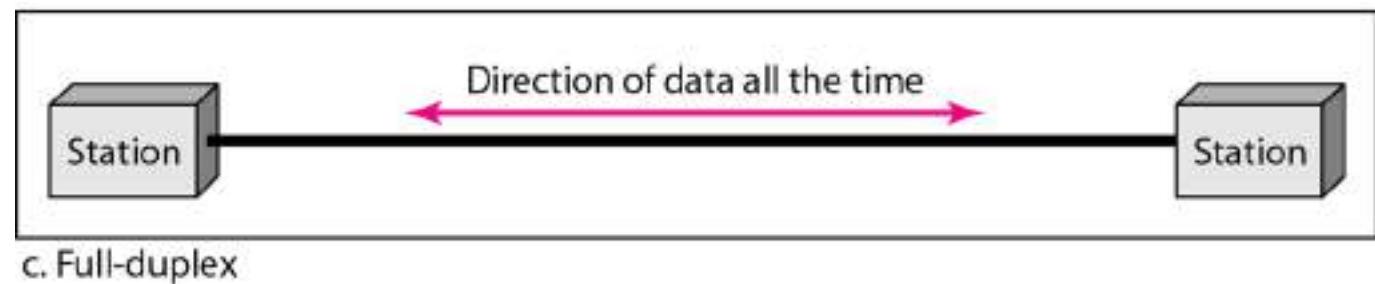
- Simplex mode, the communication is unidirectional , as a one-way street.
- Only one of the two devices on a link can transmit; other can only receive
- Example : keywords and traditional monitors



- Half-Duplex
  - Each station can both transmit and receive, but not at the same time.
  - Example : crossing road by the car
  - Example : Walkie-talkies



- Full-Duplex
  - Called also duplex
  - Both stations can transmit and receive simultaneously
  - Example : telephone network



# Network

- A network is the interconnection of a set of devices capable of communication.
- A device can be a host(e.g computer , desktop, laptop, cellular phone)
- A network must be able to meet certain number of criteria.
- The most important of these are :
  - Performance :
    - Measured in many ways, including transit time and response time.
  - Reliability
    - Is measured in frequency of failure, the time it takes a link to recover from a failure
  - Security
    - Network security issues include protecting data from unauthorized access
    - Protecting data from damage and development

# Physical Structures

- Types of connection

A network is two or more devices connected through the links.

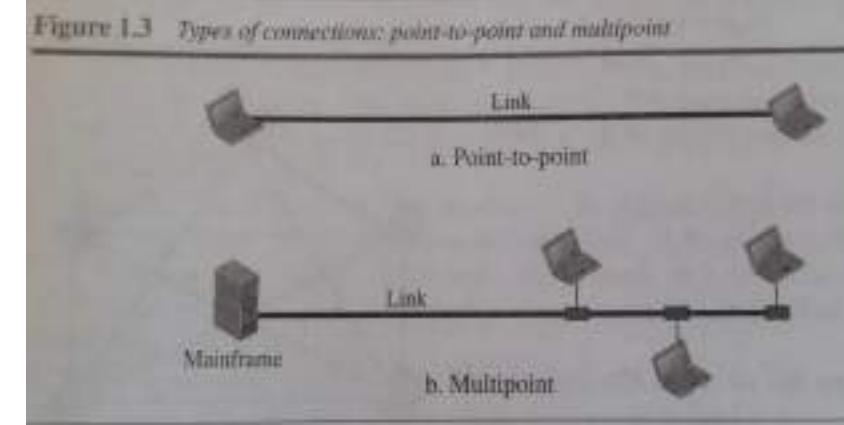
- **Point to Point**

- Provides a dedicated link between two devices
- The entire capacity of the link is reserved for transmission between these two devices.
- Wire or cable

Example : while changing the channel from remote to TV, satellite link

- **MultiPoint**

- Is one in which more than one specific devices share the single link



# LAN Topologies(Physical)

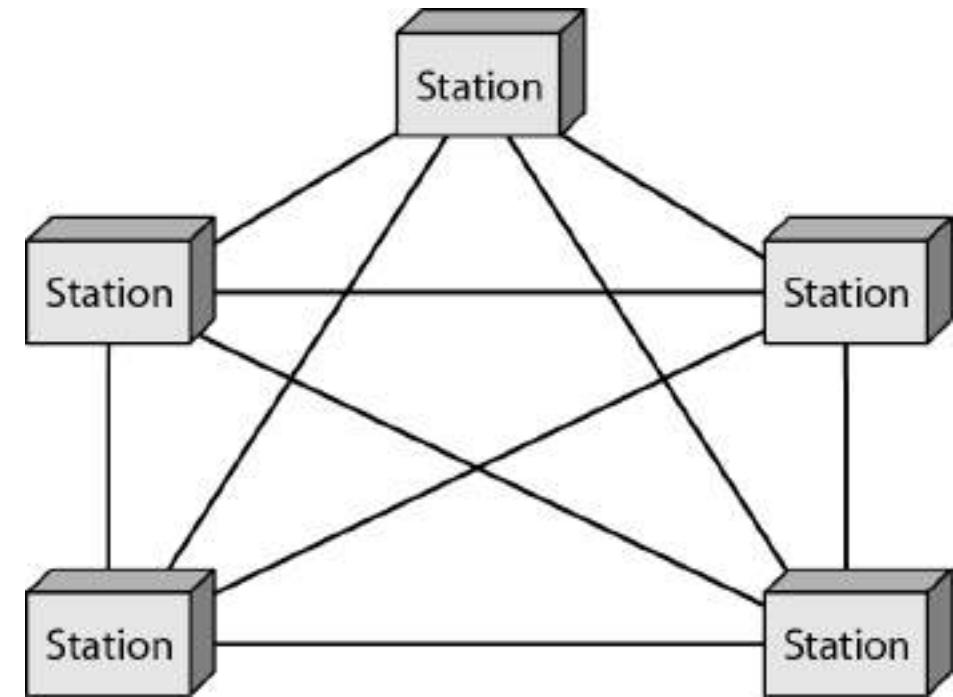
- Physical

Describes the geometric arrangement of components that make up the LAN

- 1) Mesh Topology
- 2) Bus Topology
- 3) Star Topology
- 4) Ring Topology

- Mesh Topology

- Every device has dedicated point to point link to every other
- Dedicated means link carries traffic only between the two devices it connects.
- Find the physical links in a fully connected mesh networks with  $n$  nodes
  - Node 1 must be connected to  $n-1$  nodes
  - Node 2 must be connected to  $n-1$  nodes
  - Node  $n$  must be connected to  $n-1$  nodes
- We need  $n(n-1)$  physical links.
- If both directions duplex mode than  $n(n-1)/2$



# Advantage

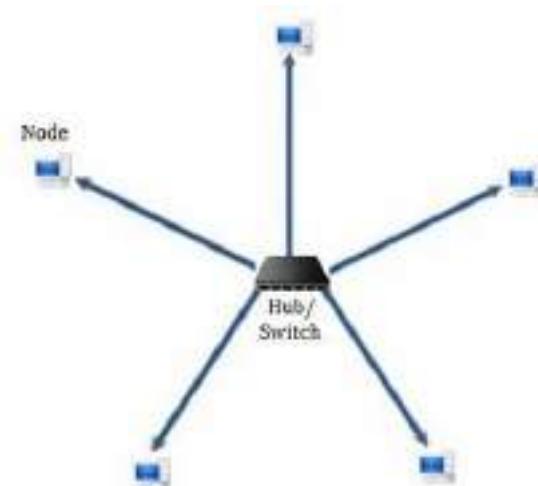
- Dedicated link guarantees that each connection can carry its own data load.
- Helps to eliminating the traffic problems
- If one link becomes unusable, it does not incapacitate the entire system
- Advantage of having privacy or security
- Helps to find fault identification and fault isolation easy.
- Traffic can be routed to avoid links

# Disadvantage

- The main disadvantage of a mesh are related to the amount of cabling and number of I/O ports
- Every device need to connected
- Sheer bulk of wiring can be greater than the available space can accommodate
- Hardware required to connect I/O
- Therefore mesh topology is usually implemented in a limited fashion.

# Star topology

- Have connections to networked devices that “radiate” out from a common point
- Each networked device in star topology can access the media independently
- Have become the dominant topology type in contemporary LANs
- Stars have made buses and rings obsolete in LAN topologies



# Advantages of star topology

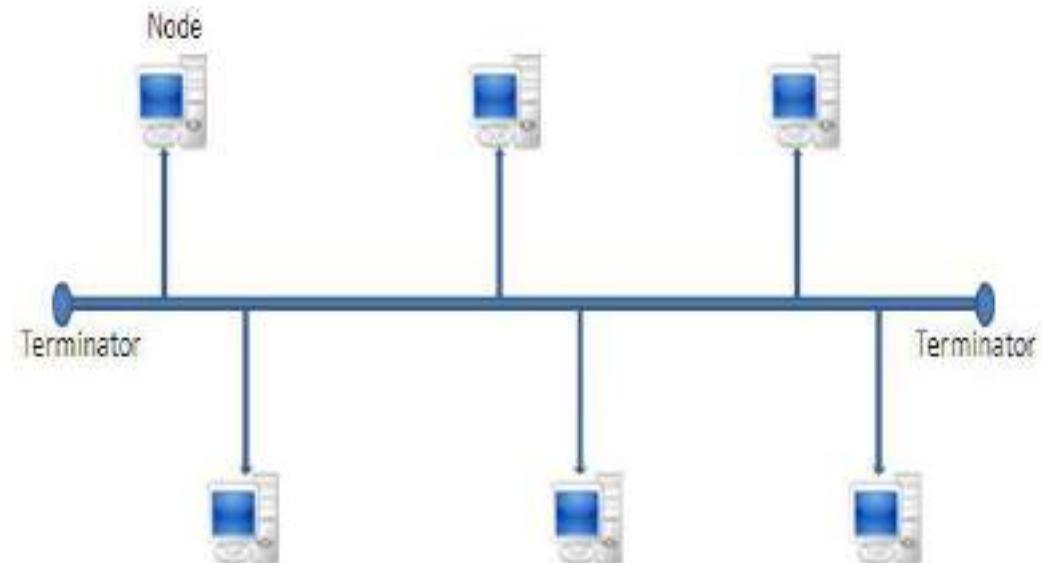
- 1) Compared to Bus topology it gives far much better performance
- 2) Easy to connect new nodes or devices
- 3) Centralized management. It helps in monitoring the network
- 4) Failure of one node or link doesn't affect the rest of network

# Disadvantages of star topology

- 1) If central device fails whole network goes down
- 2) The use of hub, a router or a switch as central device increases the overall cost of the network
- 3) Performance and as well number of nodes which can be added in such topology is depended on capacity of central device

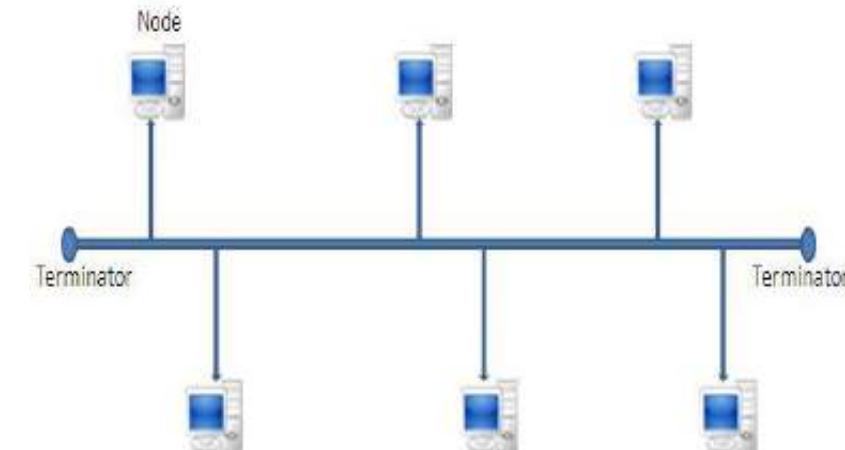
# Bus topology

- All networked nodes are interconnected, peer to peer, using a single, open-ended cable
- Both ends of the bus must be terminated with a terminating resistor to prevent signal bounce



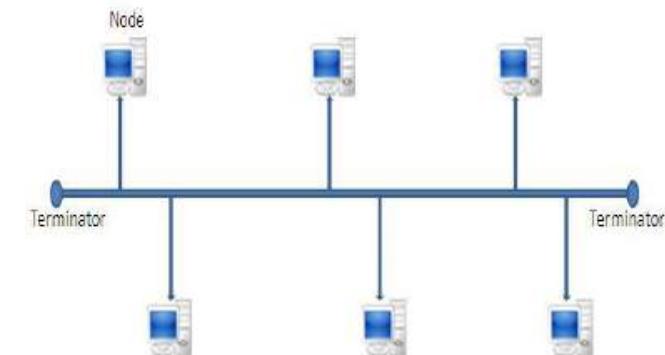
# Advantages of Bus topology

- 1) Easy to implement and extend
- 2) Well suited for temporary networks that must be set up in a hurry
- 3) Typically the least cheapest topology to implement
- 4) Failure of one station does not affect others



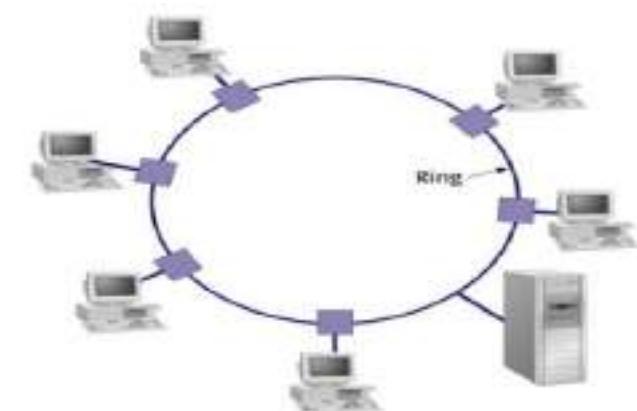
# Disadvantages of Bus topology

- 1) Difficult to administer/troubleshoot
- 2) Limited cable length and number of stations
- 3) A cable break can disable the entire network; no redundancy
- 4) Maintenance costs may be higher in the long run
- 5) Performance degrades as additional computers are added



# Ring topology

- started out as a simple peer-to-peer LAN topology
- Data was transmitted unidirectionally around the ring
- Sending and receiving of data takes place by the help of TOKEN
- Token contains a piece of information which along with data is sent by the source computer
- This token then passes to next node, which checks if the signal is intended to it
  - If yes, it receives it and passes the empty to into the network
  - otherwise passes token along with the data to next node



# Advantages of Ring topology

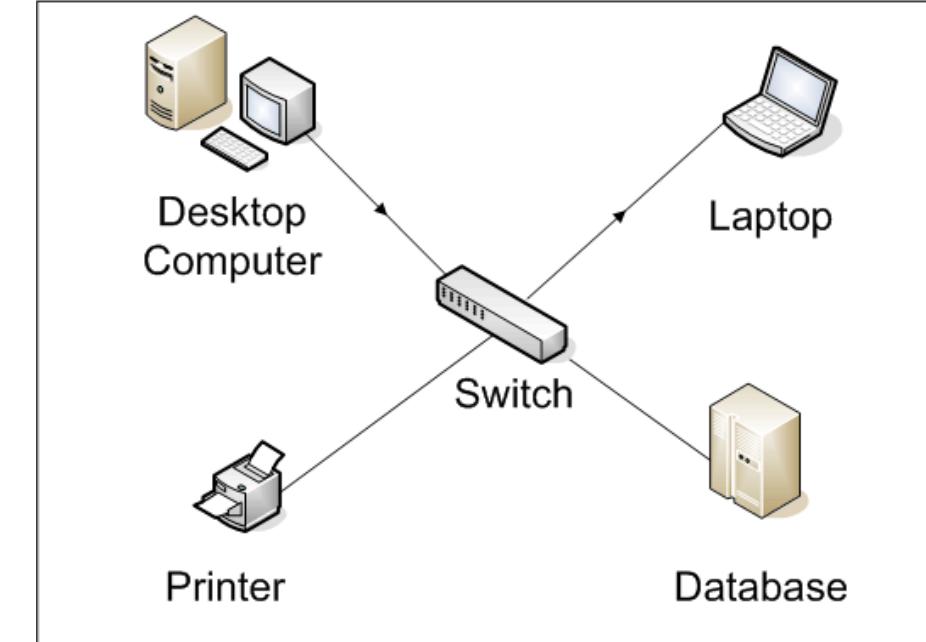
- 1) This type of network topology is very organized
- 2) No need for network server to control the connectivity between workstations
- 3) Additional components do not affect the performance of network
- 4) Each computer has equal access to resources

# Disadvantages of Ring topology

- 1) Each packet of data must pass through all the computers between source and destination, slower than star topology
- 2) If one workstation or port goes down, the entire network gets affected
- 3) Network is highly dependent on the wire which connects different components

# Network types :

- Local Area Network (LAN)
  - Less than 1 km in range
  - 10 mbps – 1 Gbps – data flow capacity
  - Within Room Building Campus



LAN or Local Area Network connects network devices in such a way **that personal computer** and workstations can share data, tools and programs.

The group of computers and devices are connected together by a switch, or stack of switches, using a private addressing scheme as defined by the TCP/IP protocol.

Private addresses are unique in relation to other computers on the local network. Routers are found at the boundary of a LAN, connecting them to the larger WAN

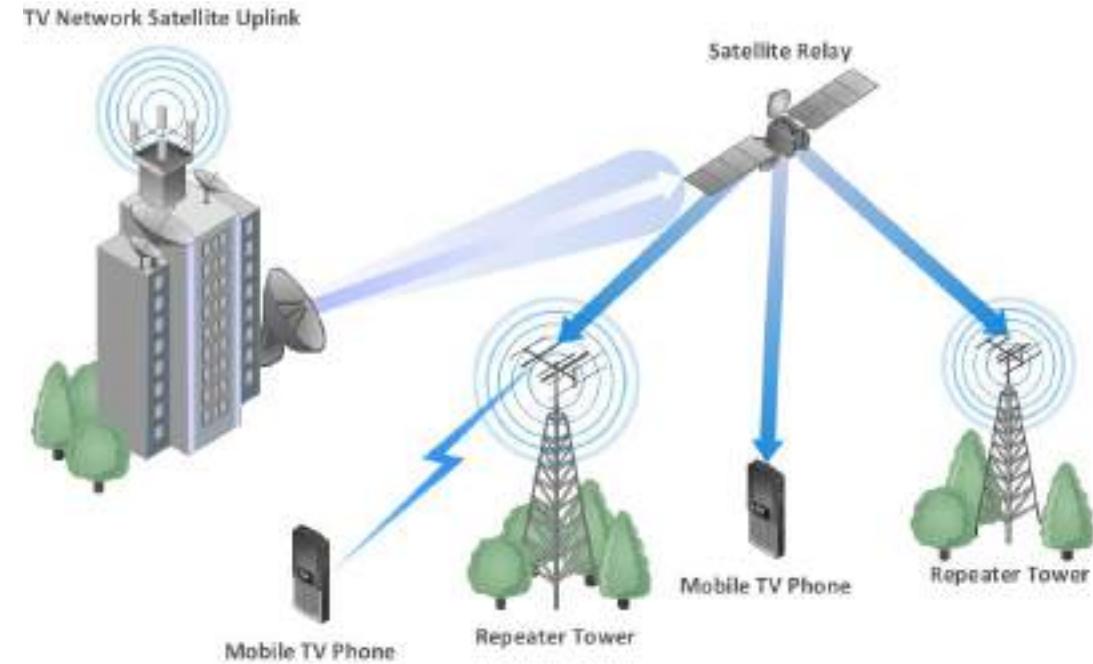
- Metropolitan Area Networks (MAN)
  - Size between a LAN and WAN (Within city area)
  - 1-10 kms in range
  - Data flow capacity is less than LAN
  - Combination of Multiple LAN's
  - High Speed Wireless Internet access has been standardized
  - Example Cable TV network

MAN or Metropolitan area Network covers a larger area than that of a LAN and smaller area as compared to WAN.

It connects two or more computers that are apart but resides in the same or different cities. It covers a large geographical area and may serve as an ISP (Internet Service Provider).

# WAN

- 100 Km -1000 km
- Data flow less than MAN
- Combination of multiple LAN's or MAN's
- Hosts are connected by a communication subnet.
- Subnets consist of many routers
- Images, audio and video information over the large geographic areas



WAN or Wide Area Network is a computer network that extends over a large geographical area, although it might be confined within the bounds of a state or country.

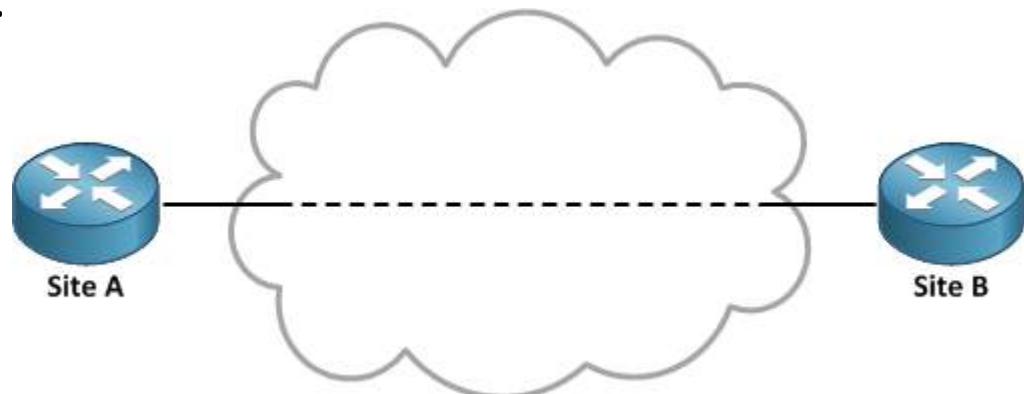
A WAN could be a connection of LAN connecting to other LAN's via telephone lines and radio waves and may be limited to an enterprise (a corporation or an organization) or accessible to the public.

The technology is high speed and relatively expensive.

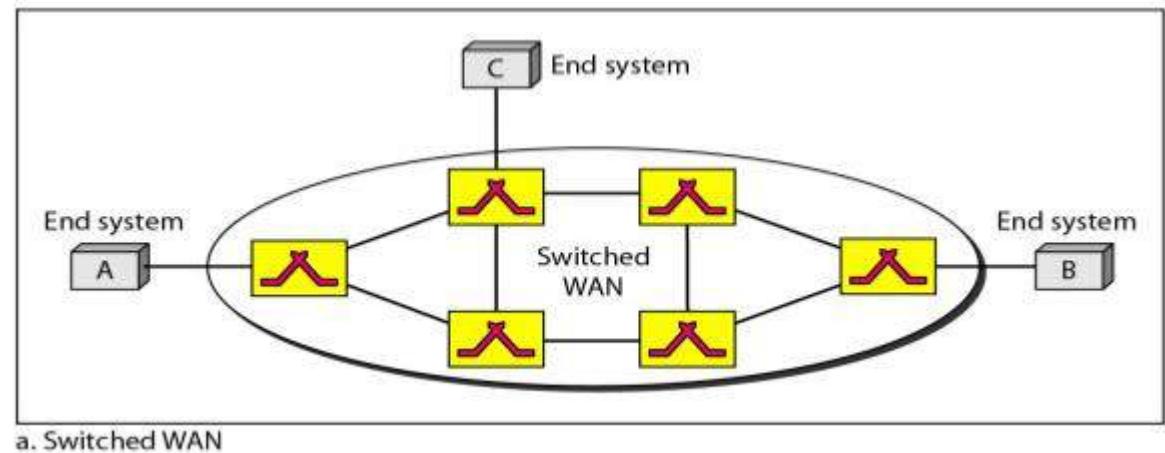
- Two example of WANs today

- Point to point WAN

- A point –to point WAN is a network that connects two communicating devices through the mission media (Cable, air)
- Network connecting in different network.



- Switched WAN
  - A switched WAN is a network with more than two ends
  - Used as a backbone of global communications todays
  - Switched WAN is a combination of several point-to-point WANs that are connected by switches.

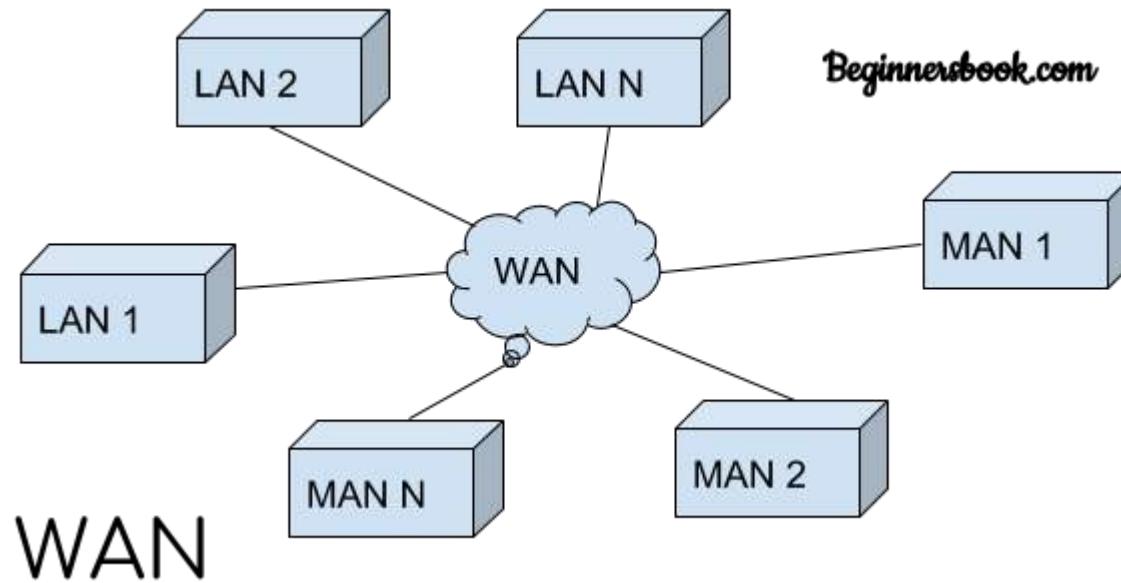


# Wireless Network

- System Interconnection : Interconnecting the components of a compute using short range radio. Wireless mouse , keyword
  - Wireless LAN's
  - Wireless WAN's

# Interconnection of Networks : Internet Work

- It is very rare to see a LAN, MAN or a LAN in isolation, they are connected to one another.
- When two or more networks are connected then we say it internet work or Internet.
- The internet is a structured , organized system.



# Protocols and Standards

- Protocols :
  - In computer networks, communication occurs between entities in different systems.
  - An entity is anything capable of sending or receiving information.
  - Two entities cannot simply send bit streams to each other and expect to be understand
    - for communication to occur the entities must agree on a protocol which is a set of rules that govern data communication.

- Key elements of protocols are:
  - **Syntax**: It refers to the structure or format of data, meaning the order in which they are presented.
  - **Semantics** : The word semantics refers to the meaning of each section of bits. The pattern of bits must be correct.
  - **Timing** : It refers to the two characteristics :
    - When data should be sent and how fast they can be sent.

# Standards

- Standard provide guidelines to manufactures, venders, government agencies and other service providers to ensure the kind of interconnectivity necessary is today's marketplace and international communications.
- Some organizations that set standards are:
  - International Organization for Standardization (ISO)
  - International Telecommunication Union - Telecommunication Standard Sector (ITU-T)
  - American National Standards Institute (ANSI)
  - Institute of Electrical and Electronics Engineer (IEEE)

# Network Models

# Protocol Layering

- First Scenario

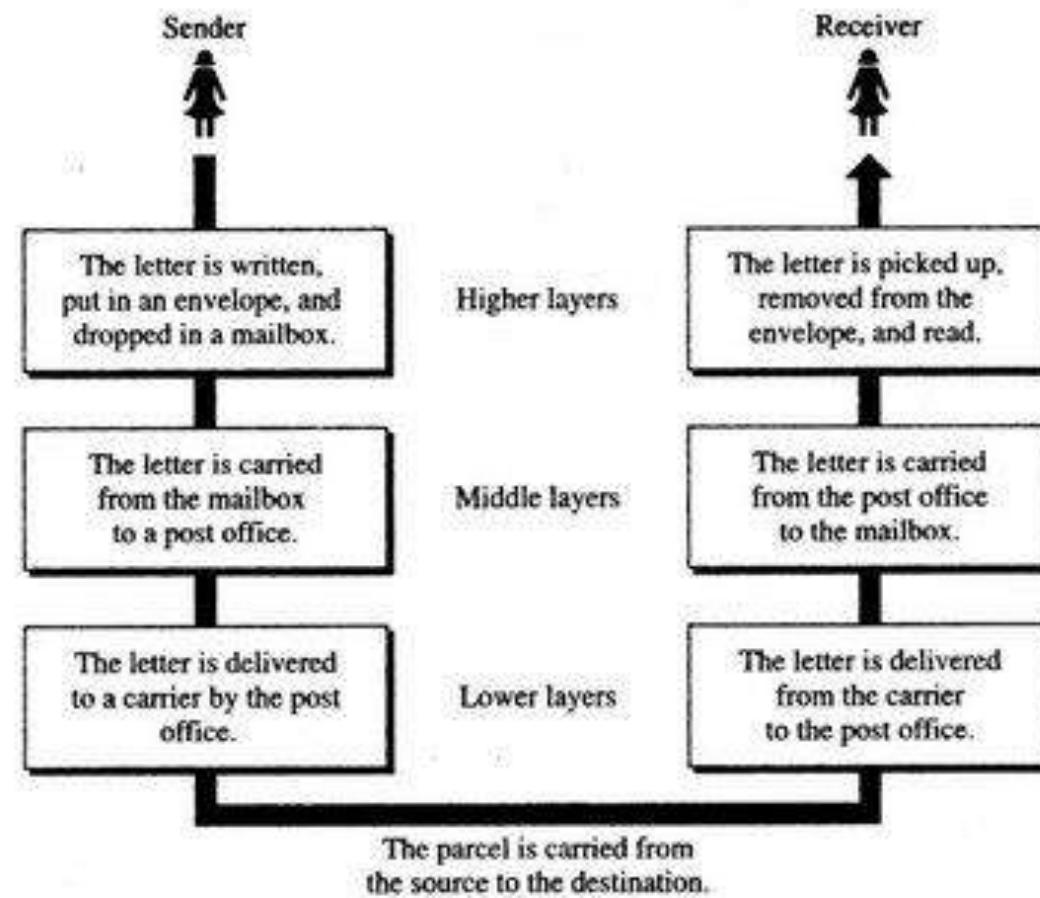
## Single-layer Protocol



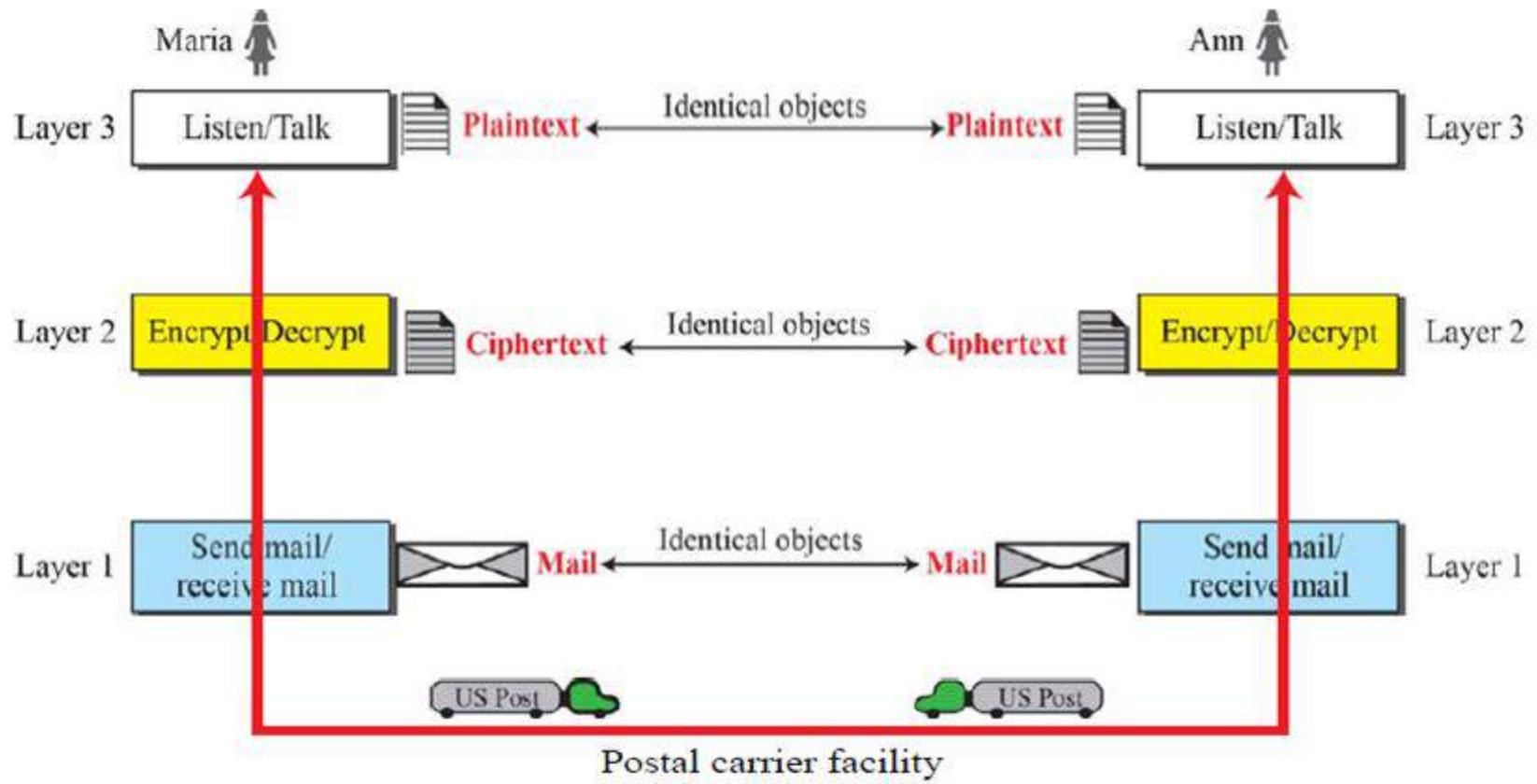
# Layered Tasks

- Layered task, play vital role to accomplished the task successfully.

- Lets take an example,
- Each layer at the site uses the services of the layer immediately below.



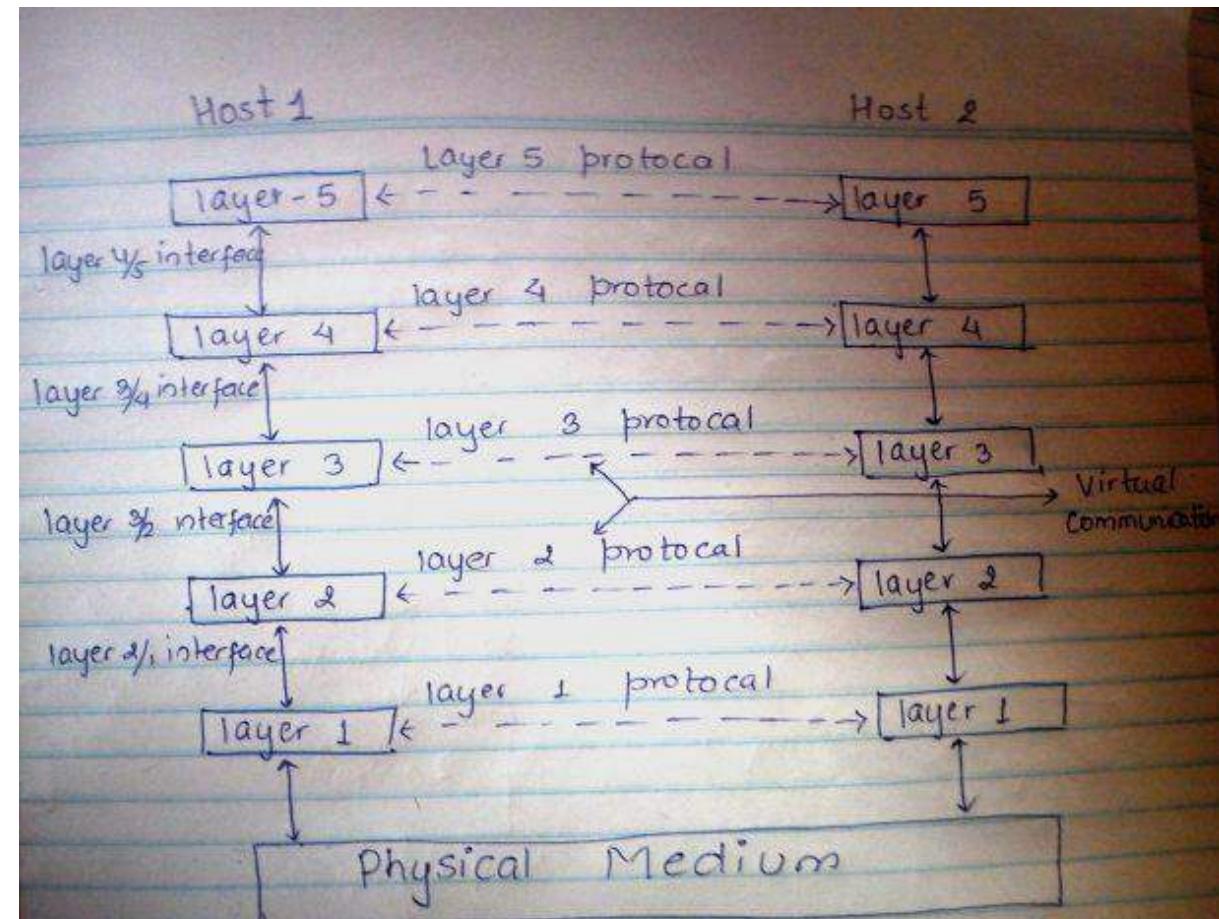
# A Three-layer protocol



- A network is a **combination of hardware and software** that **sends data** from one location to another.
- Hardware consists of **physical equipments** that **carries signals from one point of the network** to another.
- Software consists of **instruction sets** that make possible **the services that is expected** from networks

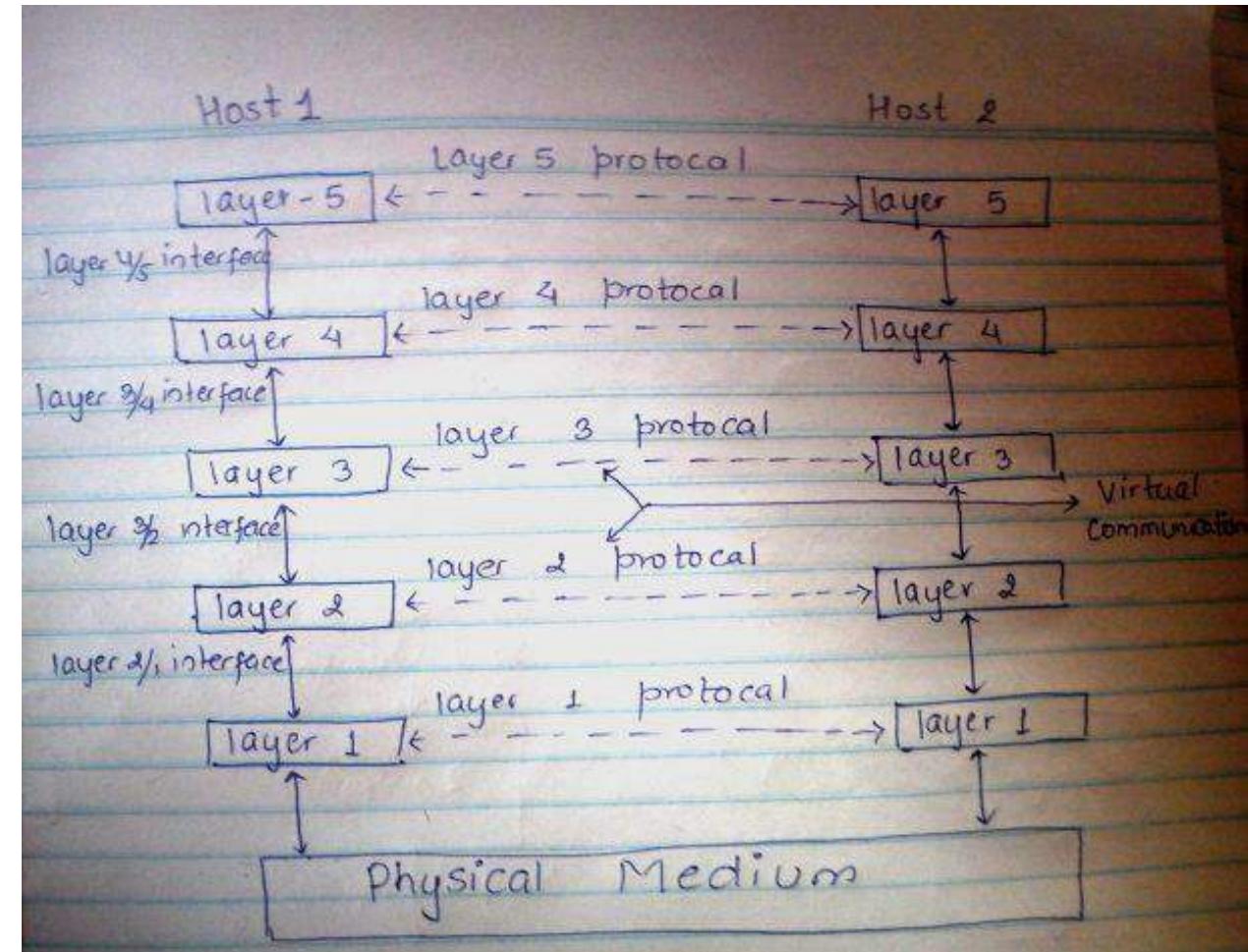
# Protocol Hierarchies :

- To reduce their design complexity, most networks are organized as a stack of layers or levels, each one built upon the one below it.
- No data are directly transferred from the layer “n” or one machine to layer “n” of other machine.
- Instead, each layer passes data and control information to the layer immediately below it, until the lowest layer is reached.



# Protocol Hierarchies :

- **Interface**
- Defines which primitive operations and services the lower layer makes available to the upper level.
- A set of layers and protocols is called **network architecture**.



# Design Issues for the layers

- **Addressing :**

- Every layer needs a mechanism for identifying senders and receivers.
- Since a **network has many computers**, it needs some sort of addressing for the accurate delivery of the message.

- **Error Control:**

- There must **be some kind of error control mechanism**.
- Receivers must have some way of telling the sender which message have been correctly received and which have not .

- **Flow Control :**

- There must be some kind of mechanism for flow control in order to guarantee that sent message has been received.
- There might be a case like the **sender in fast** and the **receiver in slow**.

## **Multiplexing:**

- Using **multiple channel in a single link in order to maximum utilization** of available bandwidth of the medium.

## **Routing:**

- When there are multiple paths between sender and receiver, then a route must be chosen for the fast and efficient transmission.

# **Connection Oriented and Connection less Services:**

- **Connection Oriented:**
  - It is modeled after **telephone system**
  - Service user first establishes a connection, uses the connection and then releases the connection.
  - It acts like a tube
- **Connectionless :**
  - It is modeled after **postal system**.
  - Each message carries the full destination address and each one is routed through the system independent of all the others.

# Reference Model

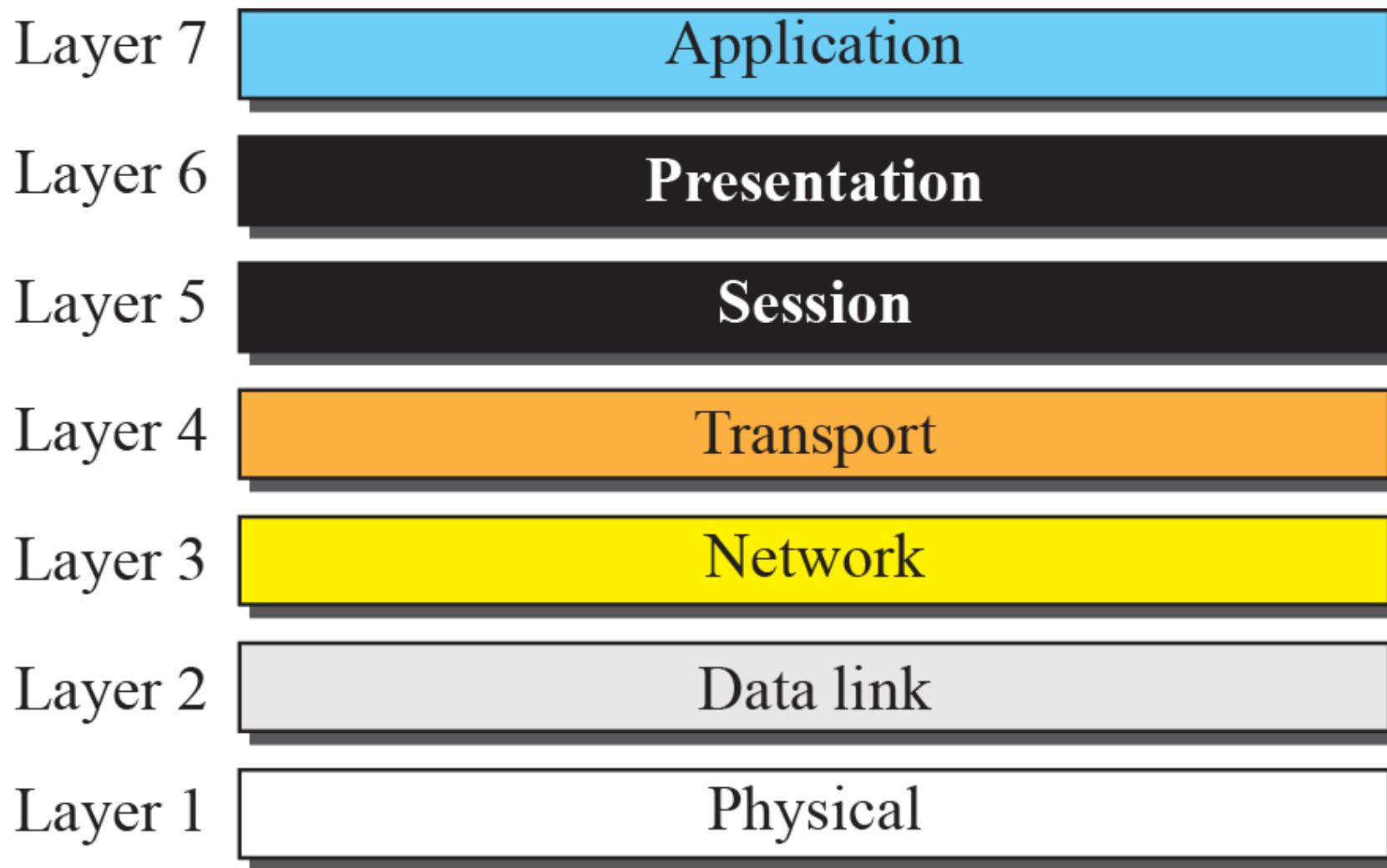
- It is a standard reference model for communication between two end users in a network.
- It deals with connecting open system ie, system that are open for communication with other system.

# Open Systems Interconnection (OSI)

## The OSI Model

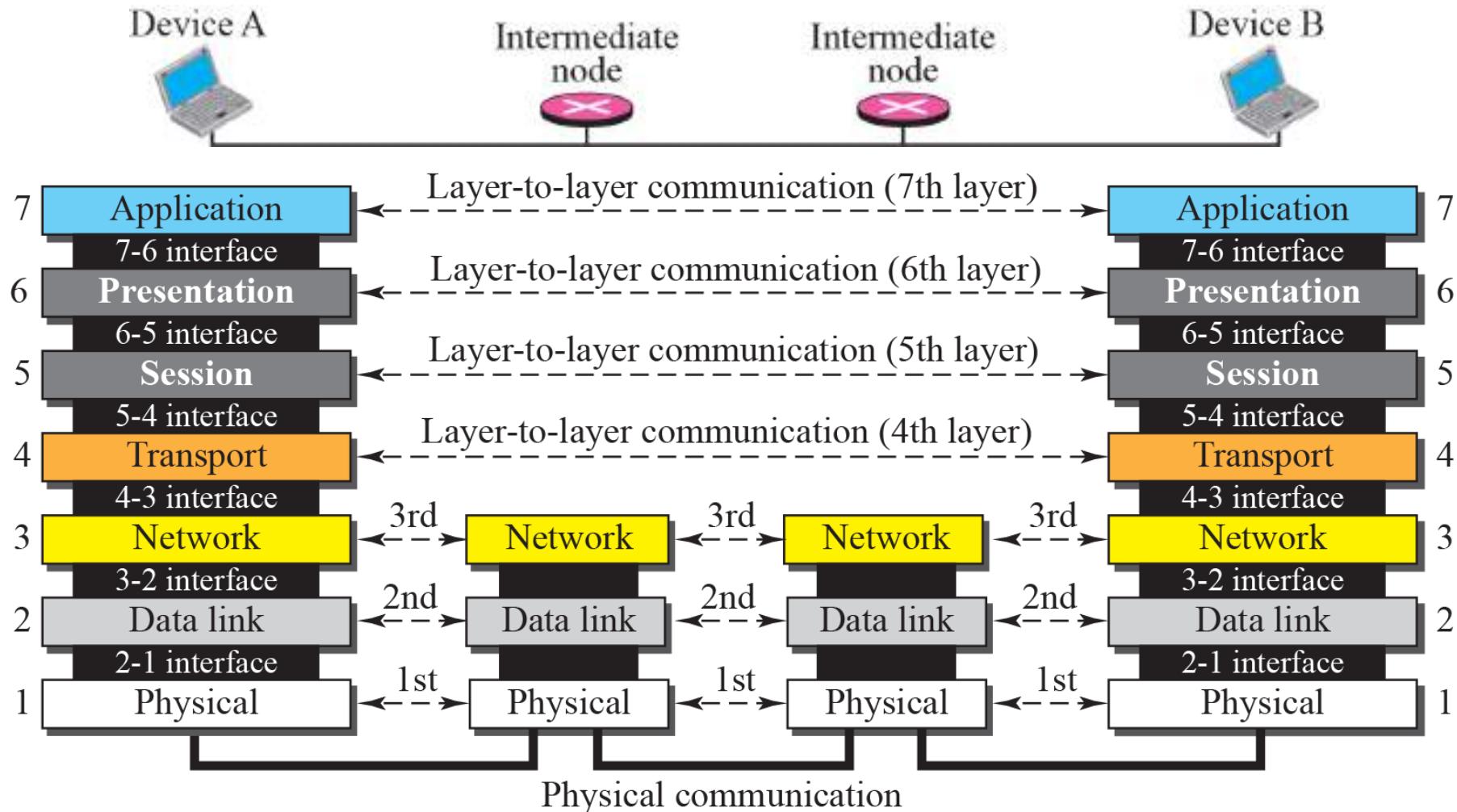
- The purpose of the OSI model to facilitate communication between different systems without requiring changes to the logic of the underlying hardware and software.
- The OSI model is not a protocol; it is a model for understanding and design of network.
- The OSI model is a layered framework for the design of network systems that allows communication between all types of computer systems.
- It consists of **seven separate** but related layers, each has responsibility for the process of moving information across a network.

# The OSI Reference Model



# Protocol Reference Model of OSI

Layer abstraction and the path of the message



# Layered Architecture

- **Peer to Peer Processes**

- Between machines, layer “x” on one machine communicates with layer “x” on another machine. This communication is governed by an agreed-upon series of rules and conventions called protocols.
- The processes on each machine that communicate at a given layer are called **peer-to-peer** processes.

- **Interface**

- Defines which primitive operations and services the lower layer makes available to the upper level.
- A set of layers and protocols is called **Network Architecture**

## Organization of Layers

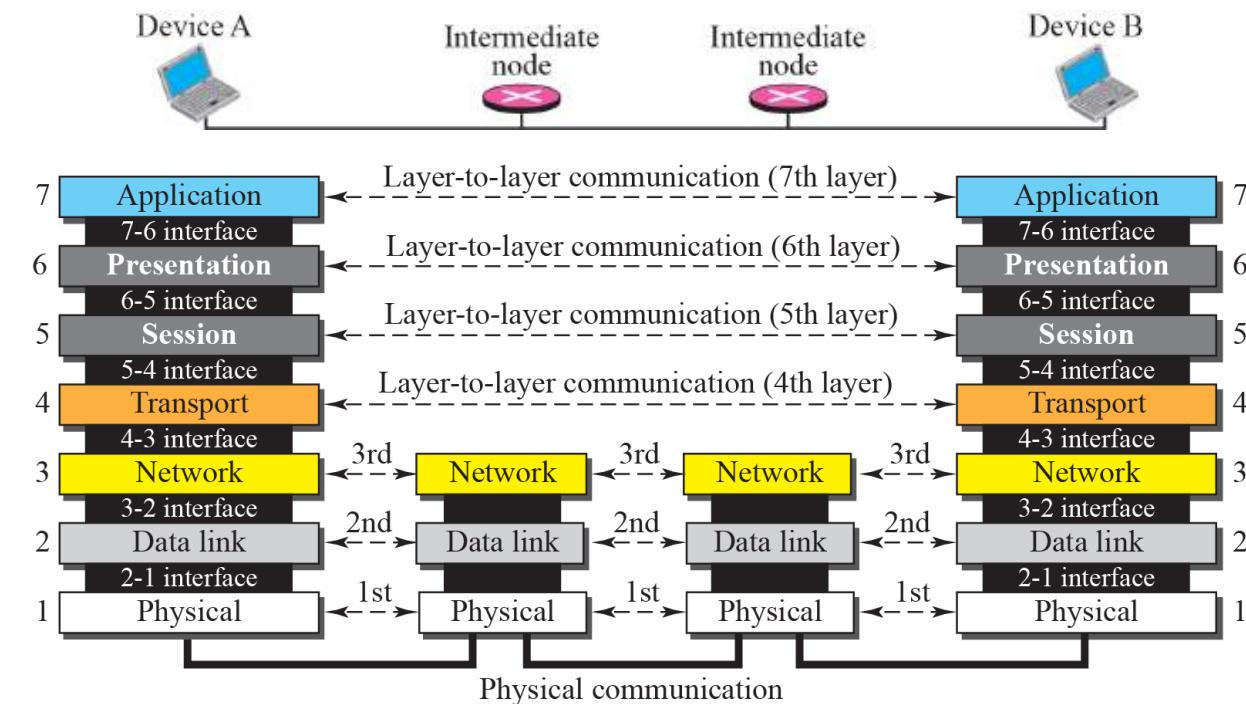
The seven layers can be thought of as belonging to three subgroups.

Layer 1,2,3 are the network support layers ;

deal with physical aspects of moving data from one device to another

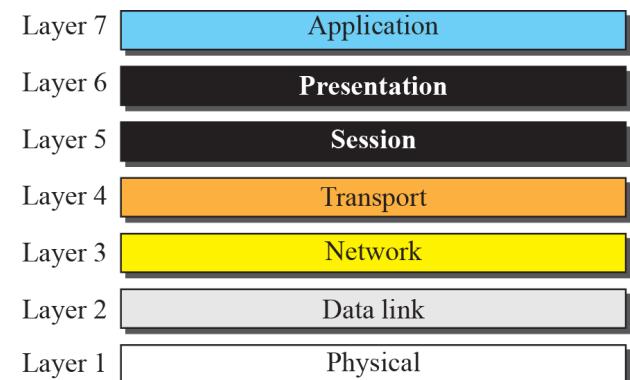
(such as electrical specifications, physical connections, physical addressing and transport timing and reliability)

- Layers 5,6,7 are the user support layers: Allows interoperability among unrelated software systems
- Layer 4 links these two subgroups and ensure that what the lower layers have transmitted is in a form that the upper layers can use.



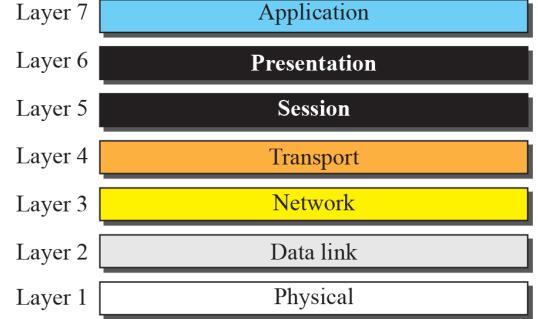
# Application Layer

- The application layer enables the user, whether human or software, to access the network.
- It Provides user interfaces and support for services such as electronic mail, remote file access and transfer.
- Applications on that layer (E-mail clients, web browsers, Chats, etc.) – top-stack applications (As people are on the top of the stack)



# Presentation Layer :

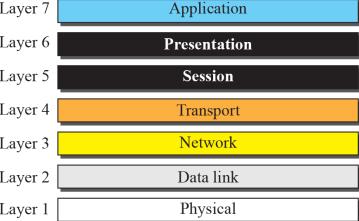
- Presentation layer is also called the **Translation layer**.
- The data from the application layer is extracted here and manipulated as per the required format to transmit over the network.



The functions of the presentation layer are :

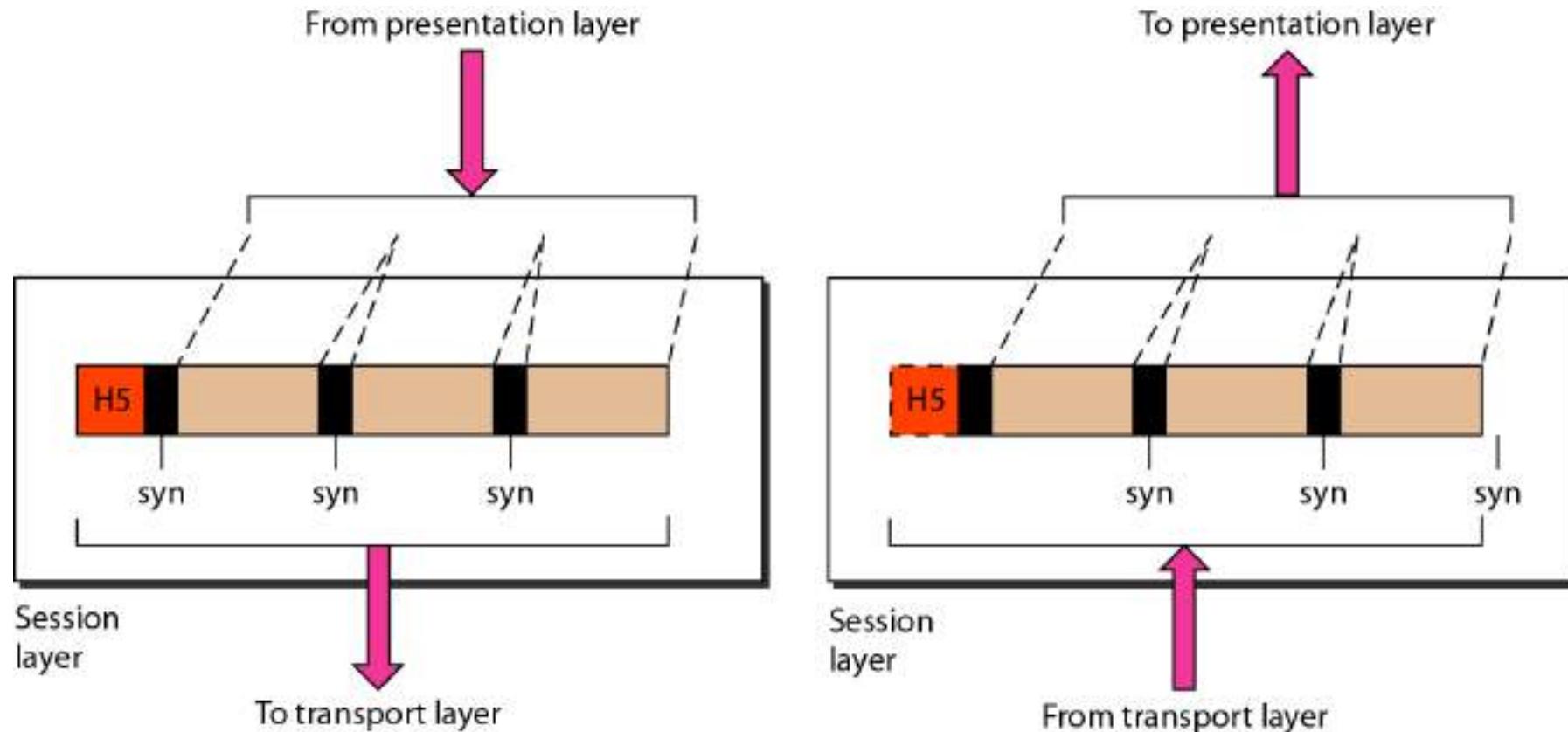
- **Translation** : For example, ASCII to EBCDIC (Extended Binary Coded Decimal Interchange Code).
- **Encryption/ Decryption** : Data encryption translates the data into another form or code. The encrypted data is known as the cipher text and the decrypted data is known as plain text. A key value is used for encrypting as well as decrypting data.
- **Compression**: Reduces the number of bits that need to be transmitted on the network.

# Session Layer



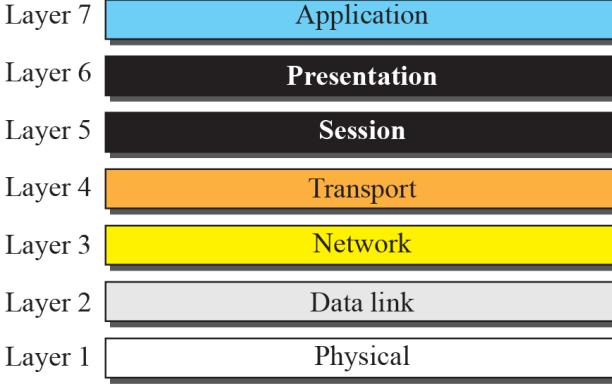
- This layer is responsible for establishment of connection, maintenance of sessions, authentication and also ensures security.
- The functions of the session layer are :
- Session establishment, maintenance and termination: The layer allows the two processes to establish, use and terminate a connection.
  - Synchronization : This layer allows a process to add checkpoints which are considered as synchronization points into the data. These synchronization point help to identify the error so that the data is re-synchronized properly, and ends of the messages are not cut prematurely and data loss is avoided.
  - Dialog Controller : The session layer allows two systems to start communication with each other in half-duplex or full-duplex.
- All the below 3 layers(including Session Layer) are integrated as a single layer in the TCP/IP model as “Application Layer”.

**Figure 2.12 Session layer**



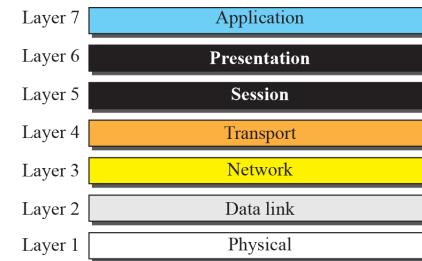
# Session Layer (Cont..)

- Lets suppose,
  - a system sends 1000 page of file.
  - It is advisable to insert checkpoints after every 100 pages to ensure that each
  - 100 pages unit is received and acknowledges independently.
  - In this case if a crash happens during the transmission of page after 535.
  - The only pages that need to be resent after system recovery are pages 501 to be resend.



# Transport Layer:

- Transport layer provides services to session layer and takes services from network layer.
- The data in the transport layer is referred to as **Segments**.
- At sender's side:
  - Transport layer receives the formatted data from the upper layers, performs Segmentation and also implements Flow & Error control to ensure proper data transmission.
  - It also adds Source and Destination port number in its header and forwards the segmented data to the Network Layer.
    - Note: The sender need to know the port number associated with the receiver's application.
  - Generally, this destination port number is configured, either by default or manually. For example, when a web application makes a request to a web server, it typically uses port number 80, because this is the default port assigned to web applications. Many applications have default port assigned.
- At receiver's side:
  - Transport Layer reads the port number from its header and forwards the Data which it has received to the respective application. It also performs sequencing and reassembling of the segmented data.



- The functions of the transport layer are :

### **Segmentation and Reassembly:**

- This layer accepts the message from the (session) layer , breaks the message into smaller units .
- Each of the segment produced has a header associated with it.
- The transport layer at the destination station reassembles the message.

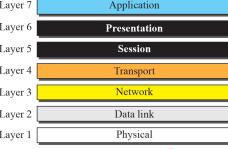
### **Service Point Addressing:**

- In order to deliver the message to correct process, transport layer header includes a type of address called service point address or port address.
- Thus by specifying this address, transport layer makes sure that the message is delivered to the correct process.

- The services provided by the transport layer :
  - Connection Oriented Service: It is a three-phase process which include
    - Connection Establishment
    - Data Transfer
    - Termination / disconnection
  - In this type of transmission, the receiving device sends an acknowledgement, back to the source after a packet or group of packet is received. This type of transmission is reliable and secure.
  - Connection less service:
    - It is a one-phase process and includes Data Transfer.
    - In this type of transmission, the receiver does not acknowledge receipt of a packet.
    - This approach allows for much faster communication between devices.
    - Connection-oriented service is more reliable than connectionless Service.

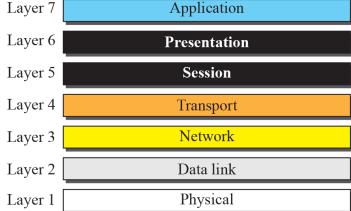
Data in the Transport Layer is called as **Segments**.

Transport layer is operated by the Operating System. It is a part of the OS and communicates with the Application Layer by making system calls.



# Network Layer:

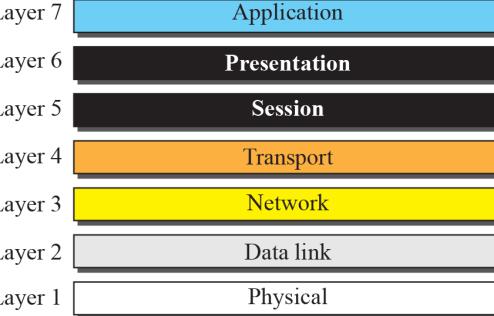
- The network layer is responsible for creating a **connection between the source computer and the destination** computer.
- It also takes care of **packet routing** i.e. selection of the shortest path to transmit the packet, from the number of routes available.
- The sender & receiver's IP address are placed in the header by the network layer.
- The functions of the Network layer are :
  - **Routing:** The network layer protocols determine which route is suitable from source to destination. This function of network layer is known as routing.
  - **Logical Addressing:** In order to identify each device on internetwork uniquely, network layer defines an addressing scheme. The sender & receiver's IP address are placed in the header by network layer. Such an address distinguishes each device uniquely and universally.
- Segment in Network layer is referred as **datagram**. Network layer is implemented by networking devices such as routers.



# Data Link Layer

- The main function of this layer is to make sure data transfer is **error-free** from one node to another, over the physical layer.
- When a packet arrives in a network, it is the responsibility of DLL to transmit it to the Host using its MAC address.
- The packet received from Network layer is further divided into **frames** depending on the frame size of NIC(Network Interface Card). DLL also encapsulates Sender and Receiver's MAC address in the header.
- The Receiver's MAC address is obtained by placing an ARP(Address Resolution Protocol) request onto the wire asking "Who has that IP address?" and the destination host will reply with its MAC address.

# Data Link Layer



- The functions of the data Link layer are :
- **Framing:** Framing is a function of the data link layer. It provides a way for a sender to transmit a set of bits that are meaningful to the receiver. This can be accomplished by attaching special bit patterns to the beginning and end of the frame.
- **Physical addressing:** After creating frames, Data link layer adds physical addresses (MAC address) of sender and/or receiver in the header of each frame.
- **Error control:** Data link layer provides the mechanism of error control in which it detects and retransmits damaged or lost frames.
- **Flow Control:** The data rate must be constant on both sides else the data may get corrupted thus , flow control coordinates that amount of data that can be sent before receiving acknowledgement.
- **Access control:** When a single communication channel is shared by multiple devices, MAC sub-layer of data link layer helps to determine which device has control over the channel at a given time.

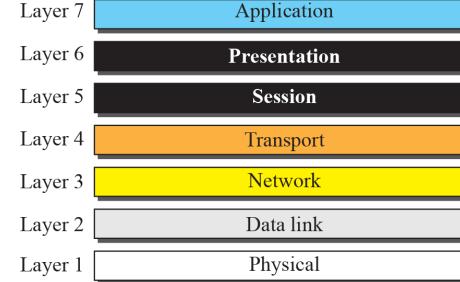
Packet in Data Link layer is referred as **Frame**.

Data Link layer is handled by the NIC (Network Interface Card) and device drivers of host machines.

Eg. Switch & Bridge are Data Link Layer devices.

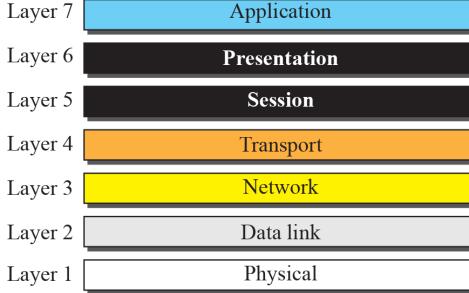
# Physical Layer:

- The lowest layer of the OSI reference model is the physical layer.
- It is responsible for the **actual physical connection between the devices**. The physical layer contains information in the form of bits.
- It is responsible for transmitting individual bits from one node to the next.
- When receiving data, this layer will get the signal received and convert it into 0s and 1s and send them to the Data Link layer, which will put the frame back together.



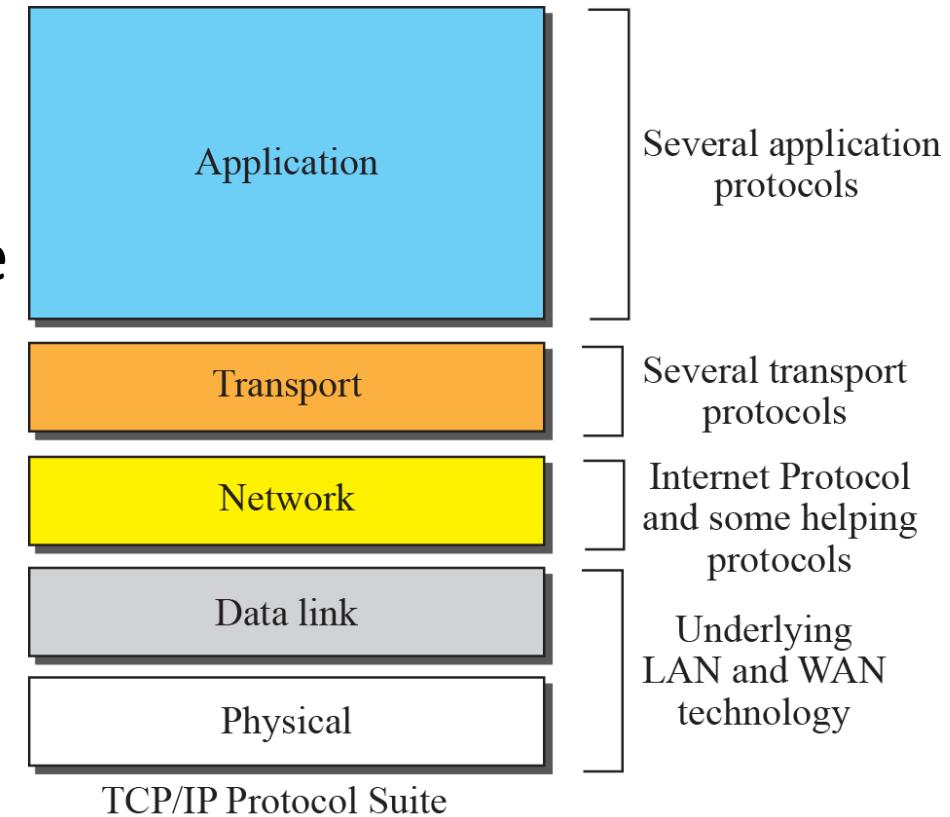
# Physical Layer:

- The functions of the physical layer are :
  - **Bit synchronization:** The physical layer provides the synchronization of the bits by providing a clock. This clock controls both sender and receiver thus providing synchronization at bit level.
  - **Bit rate control:** The Physical layer also defines the transmission rate i.e. the number of bits sent per second.
  - **Transmission mode:** Physical layer also defines the way in which the data flows between the two connected devices. The various transmission modes possible are: Simplex, half-duplex and full-duplex.
- Hub, Repeater, Modem, Cables are Physical Layer devices.

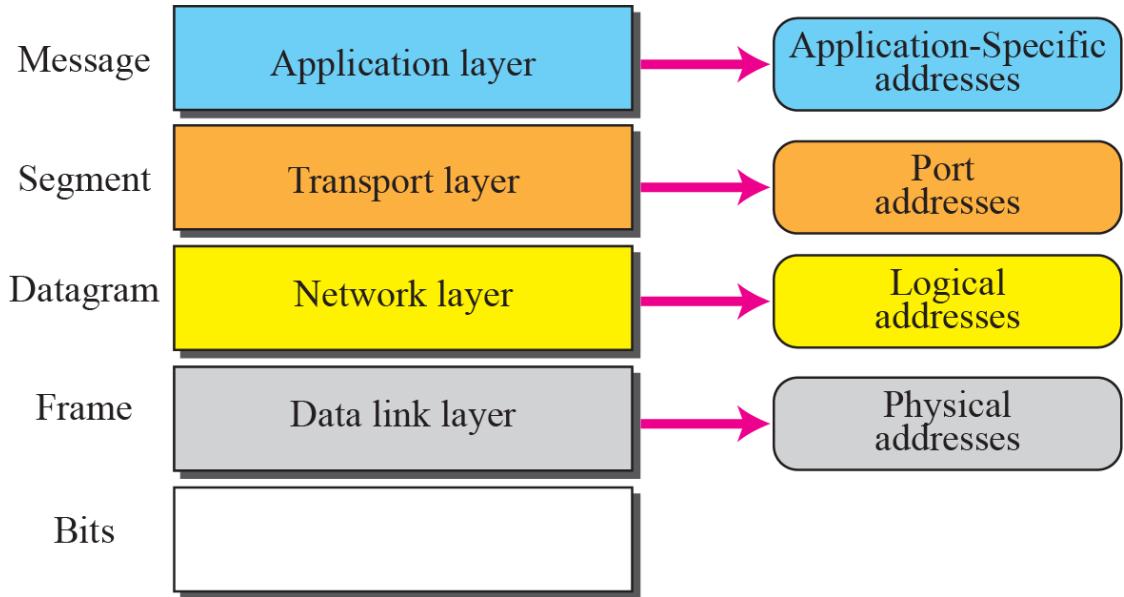


# TCP/IP Protocol Suite

- This protocol was developed prior to OSI layer
- It has five layers
- TCP/IP is a protocol suite used in the Internet today.
- Missing session and presentation



**Figure 2.15 Addresses in the TCP/IP protocol suite**



Application layer -> someorg.com or email address

Transport layer → port numbers

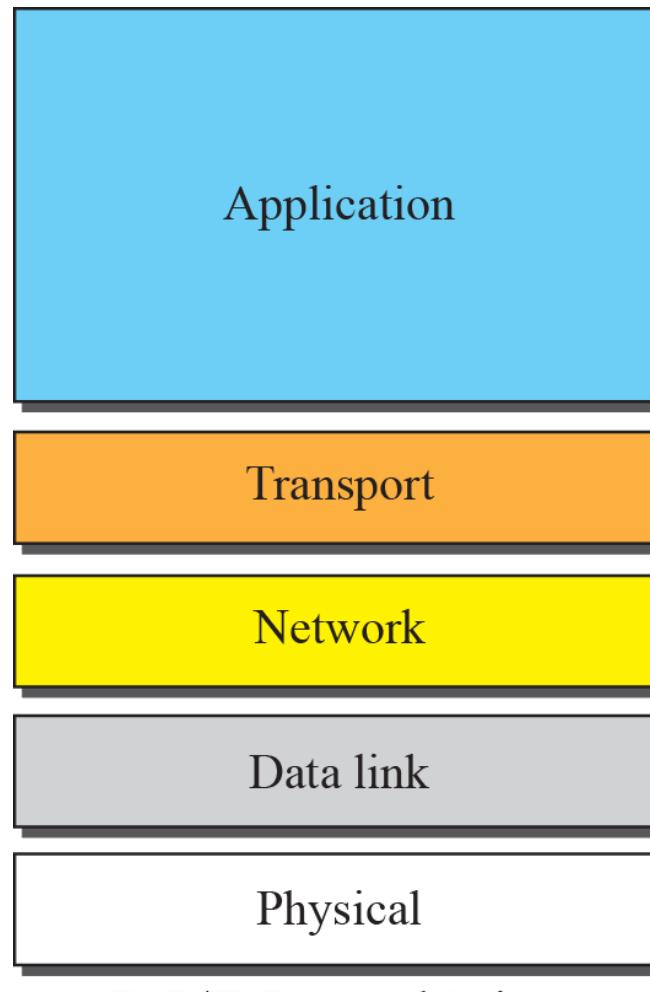
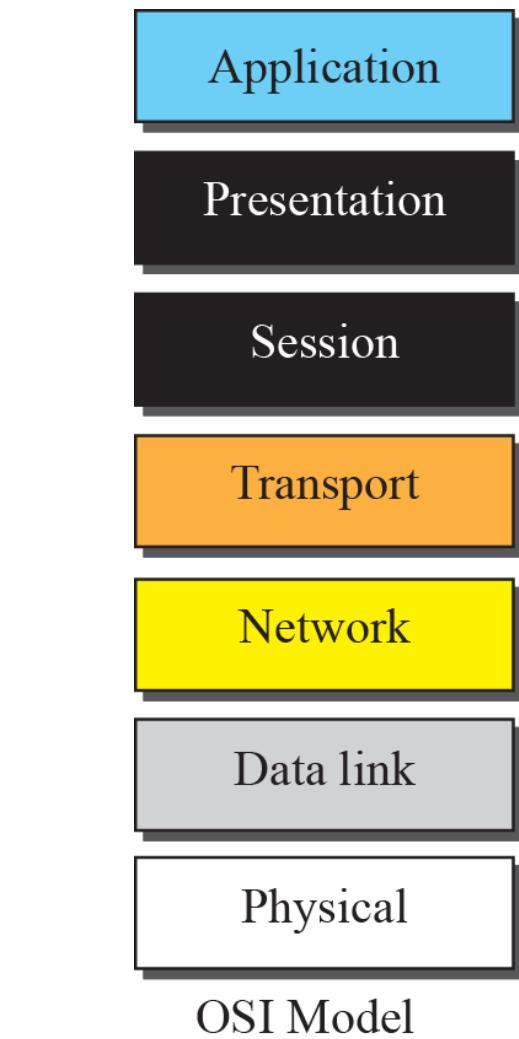
port numbers are local addresses

Network-layer address → addresses are global, with the whole Internet as the scope

Data-link layer address → MAC addresses

are locally address, each of which defines a specific host or router in a network (LAN or WAN)

## *OSI model vs TCP/IP*



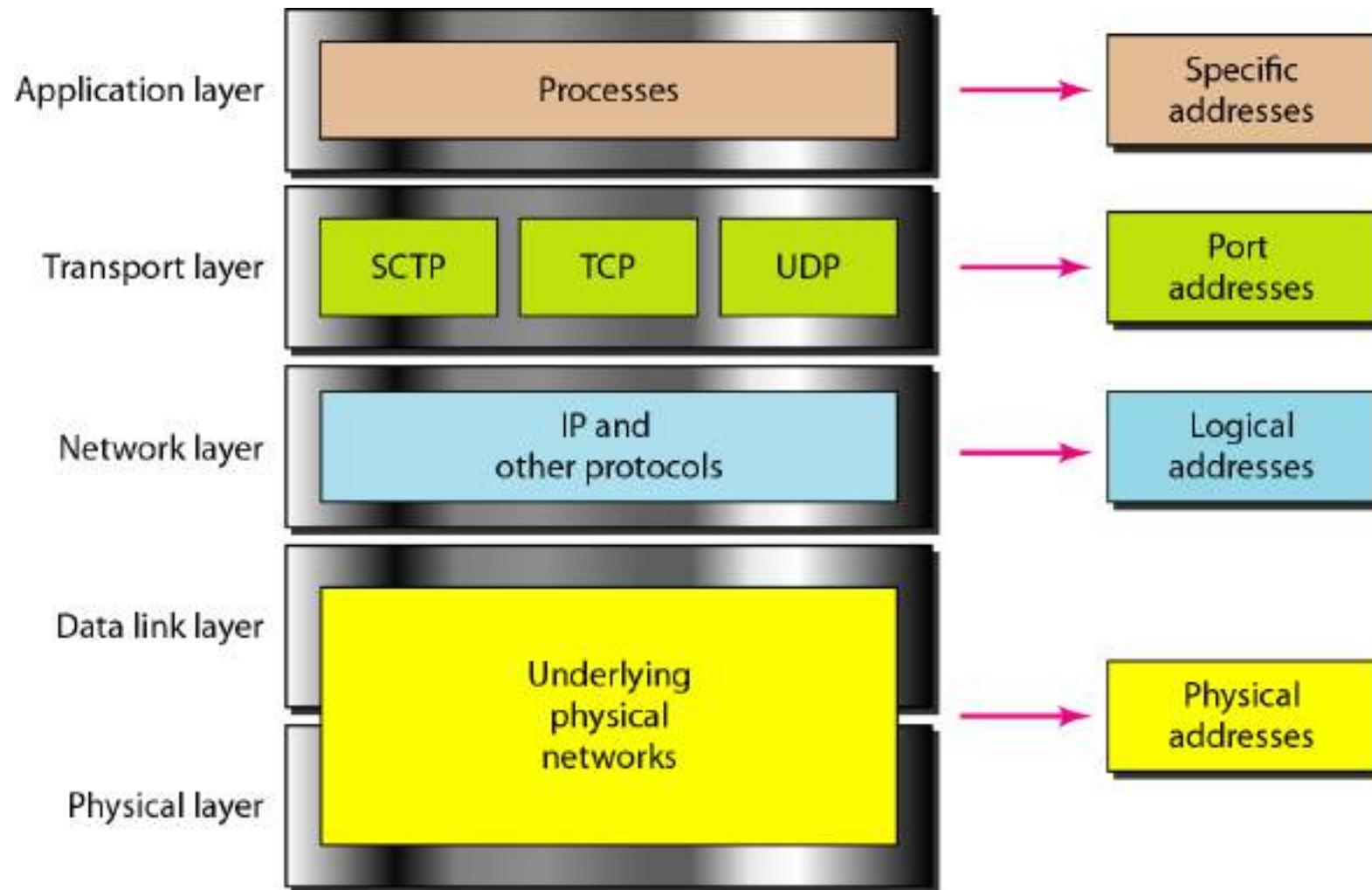
Several application protocols

Several transport protocols

Internet Protocol and some helping protocols

Underlying LAN and WAN technology

**Figure 2.18** Relationship of layers and addresses in TCP/IP

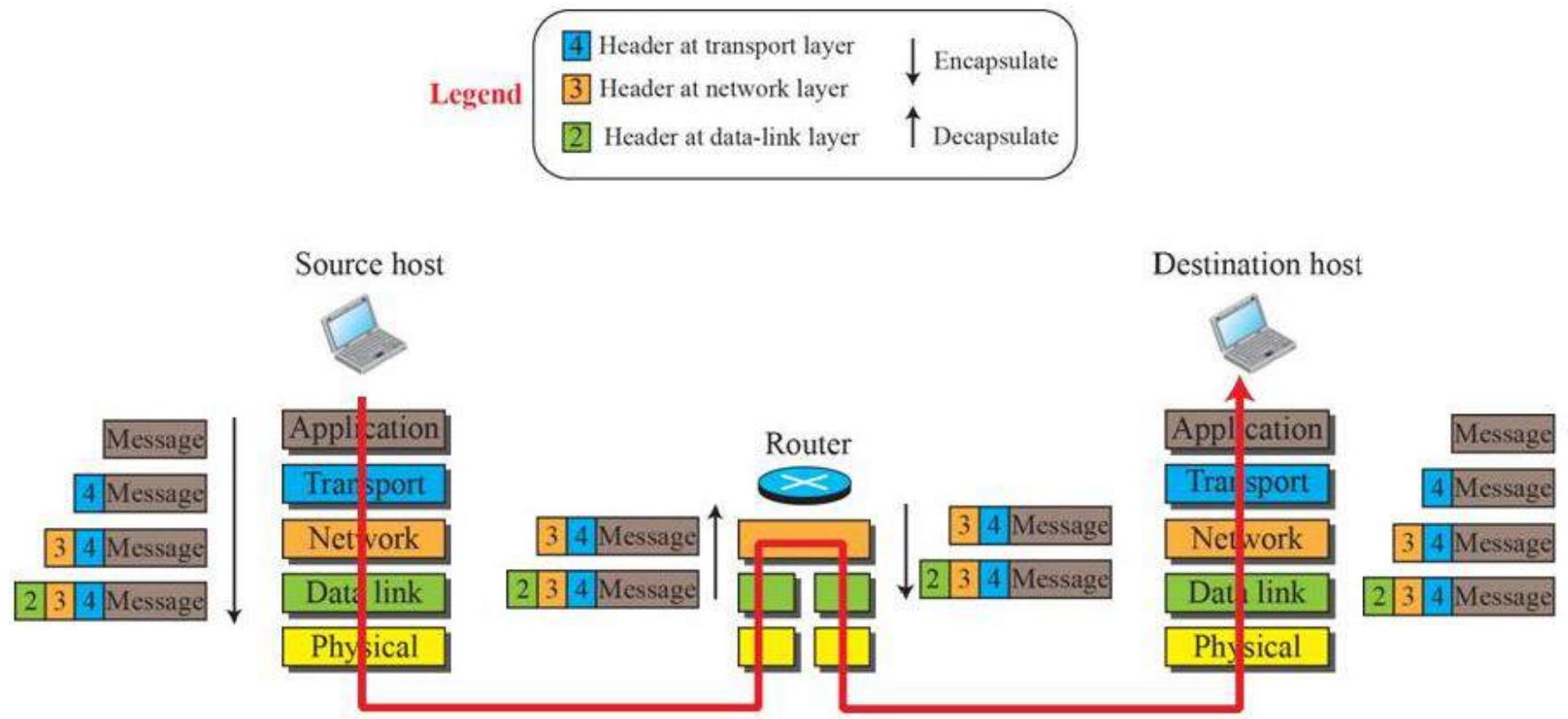


# *OSI model vs TCP/IP*

- The OSI model appeared after the TCP/IP protocol suite.
- Most experts thought that the TCP/IP protocol would be fully replaced by the OSI model.
- This didn't happen
- There were three reasons (agreed by experts)
  - OSI was completed when TCP/IP was fully in place and a lot of time and money has been spent which results changing would cost a lot of money
  - Some layers were not fully defined such as presentation and session layer.
  - When OSI was implemented by an organization in a different application, it didn't show a high enough level of performance to entice the Internet authority to switch from the TCP/IP protocol suite to the OSI model

# Encapsulation and Decapsulation

**Figure 2.8: Encapsulation/Decapsulation**



# Encapsulation at the Source Host

At the application layer : message (does not contain any header or trailer)

Transport layer : takes message as the payload

Adds a transport layer header to the payload which contains the identifiers of the source and destination application programs

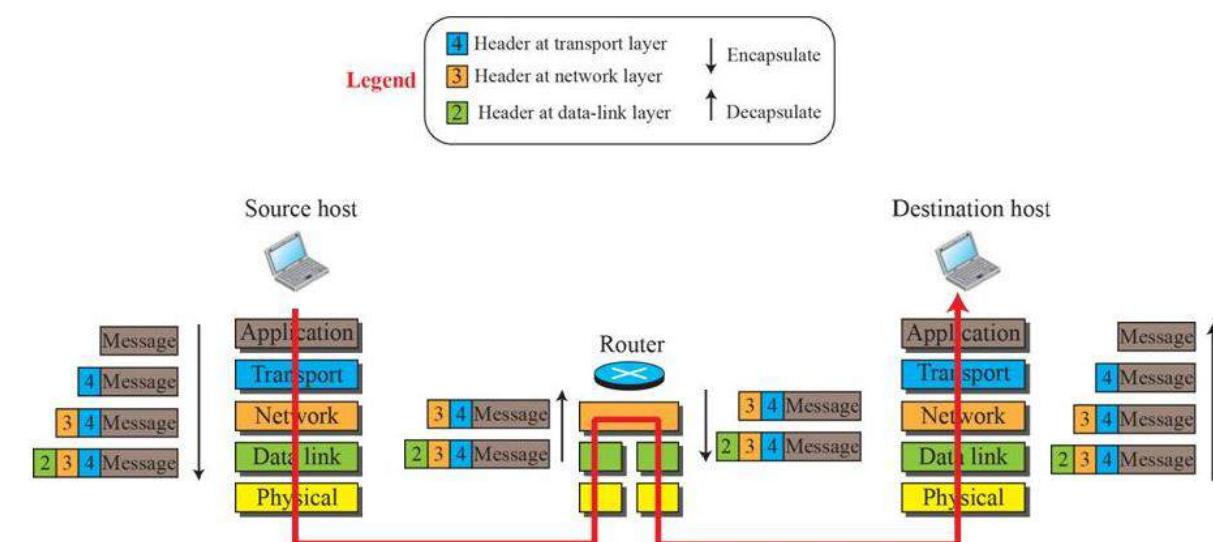
Transport-layer packet is called segment in (TCP) and the user datagram (in UDP)

The transport layer then passes the packet to the network layer.

- Network Layer : takes the transport – layer packet as data or payload.

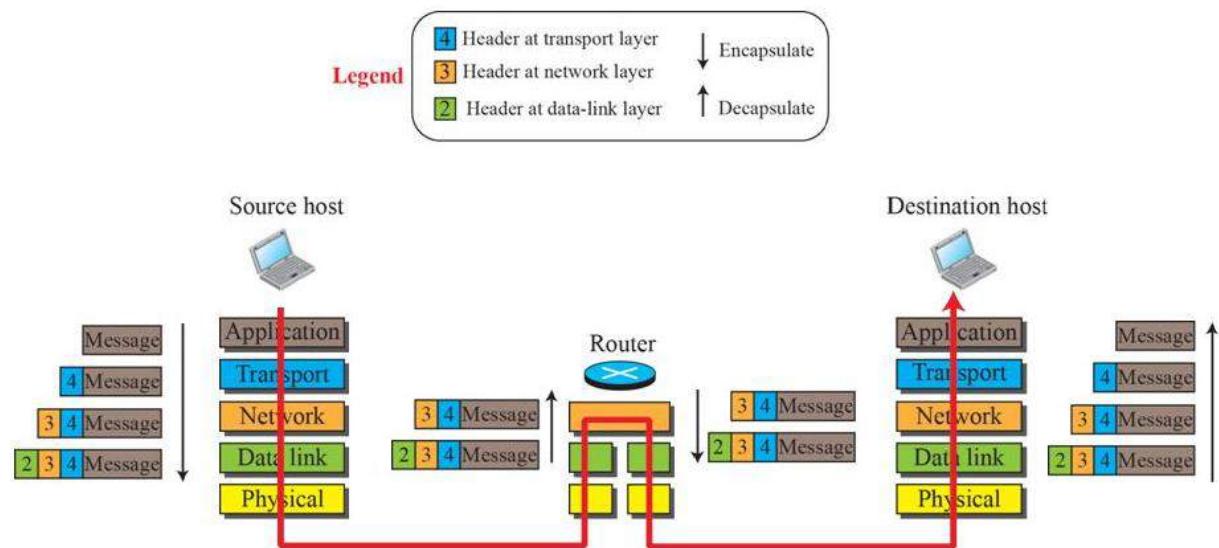
- Adds its own header to the payload.
- The header contains the addresses of the source and destination hosts and some more information used for error checking for the header.
- The result is the network-layer packet called as datagram.
- The network layer packets transfer to the data link layer.

**Figure 2.8: Encapsulation / Decapsulation**



- Data-link layer takes the network layer packet as data and its own header which contains link layer address of the host or next hop (router).
- The link layer packet is called as **frame**.
- The frame is passed to the physical layer for the transmission.

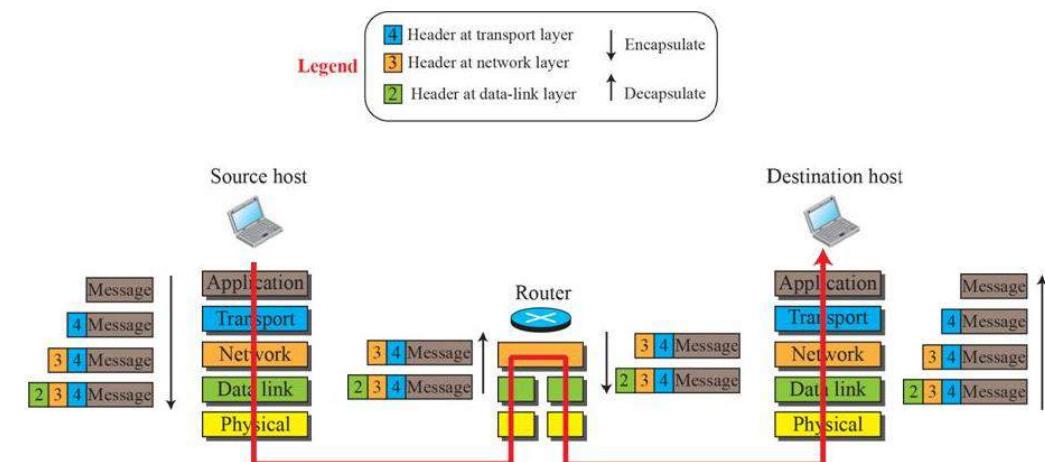
**Figure 2.8: Encapsulation / Decapsulation**



# Decapsulation and Encapsulation at the Router

- Router has encapsulation as well as decapsulation.
- After sets of bits are delivered to the data-link layer , this layer decapsulates the datagram from the frame and passes it to the network layer.
- Network layer just inspect the source and destination address in the datagram header and consults its table to forward to the respective host.
- The datagram should not be changed by the network layer in the router unless there is a need to fragment the datagram if it is too big.

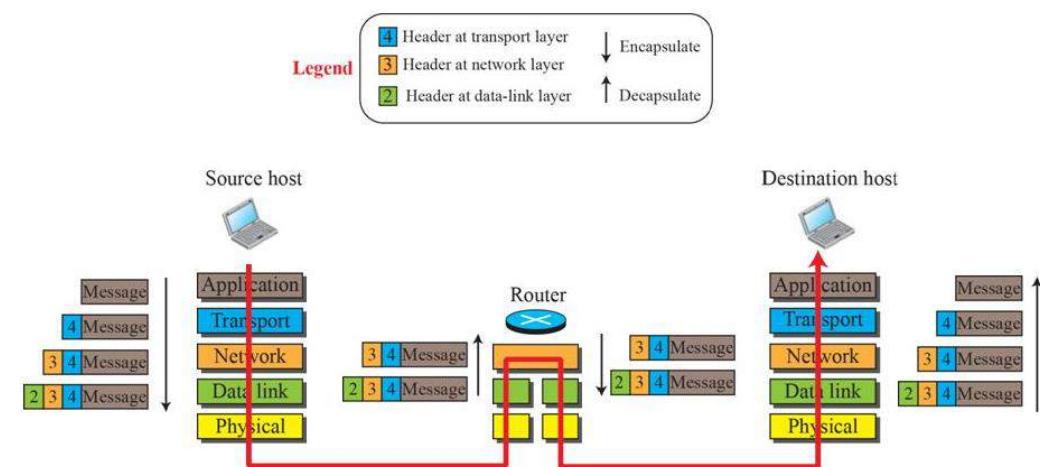
**Figure 2.8: Encapsulation / Decapsulation**



# Decapsulation at the Destination Host

- At the destination host, each layer only decapsulates the packet received, removes the payload, and delivers the payload to the next-higher layer protocol until the message reaches the application layer.
- It is necessary to say that decapsulation in the host error checking.

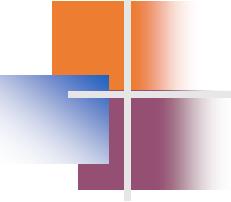
**Figure 2.8: Encapsulation / Decapsulation**



# Physical Layer

Prakash Poudyal

Department of Computer Science and Engineering  
Kathmandu University



---

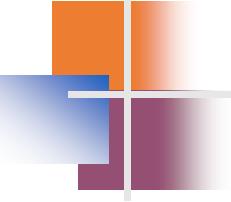
To be transmitted, data must be transformed to electromagnetic signals.

---

## 3-1 ANALOG AND DIGITAL

Data can be **analog** or **digital**.

The term **analog data** refers to information that is continuous;  
**digital data** refers to information that has discrete states.



---

Analog signals can have an infinite number of values in a range;

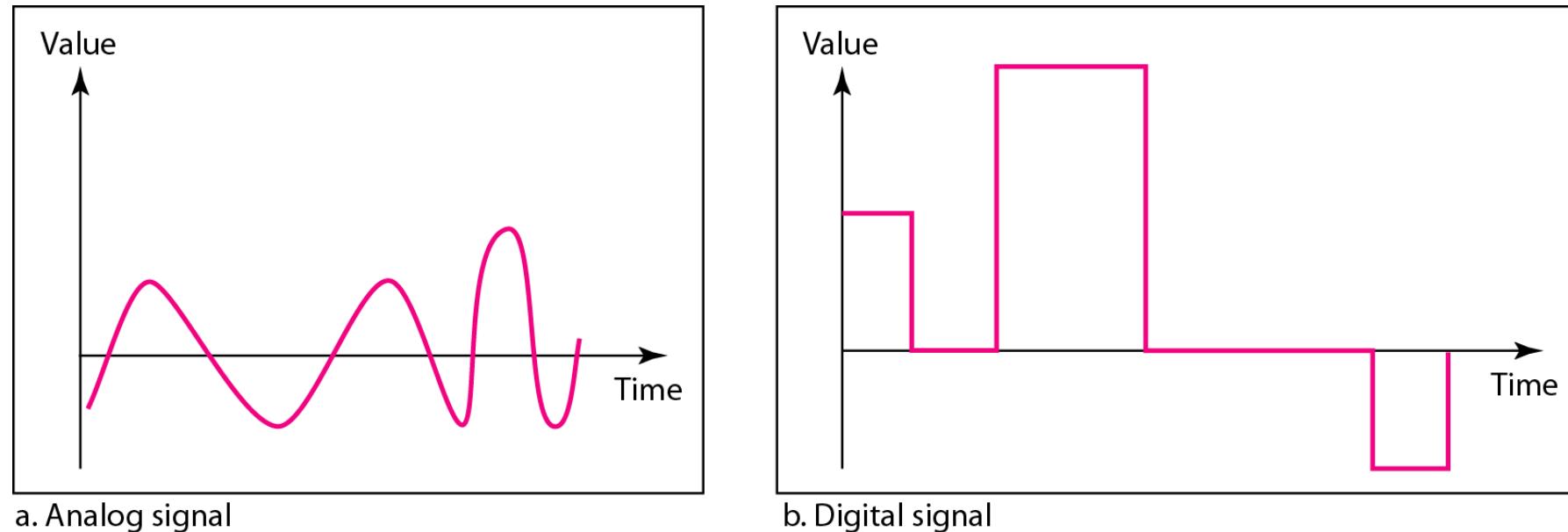
digital signals can have only a limited number of values.

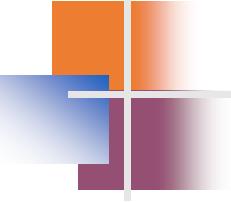
---

---

**Figure 3.1** Comparison of analog and digital signals

---





---

In data communications, we commonly use periodic analog signals and nonperiodic digital signals.

---

## 3-2 PERIODIC ANALOG SIGNALS

Periodic analog signals can be classified as **simple** or **composite**.

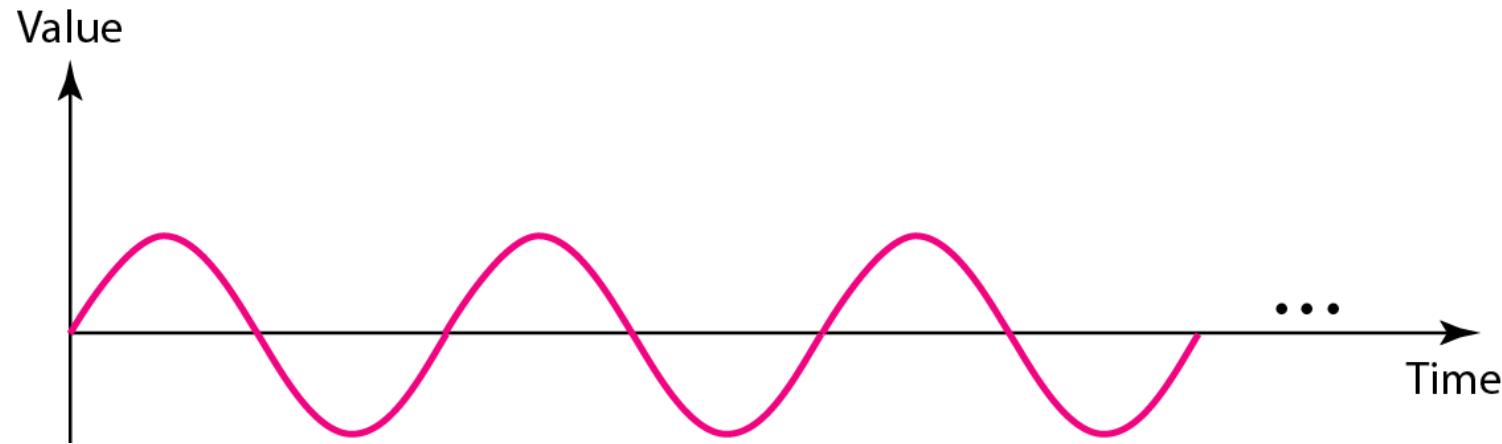
A simple periodic analog signal, a **sine wave**, cannot be decomposed into simpler signals.

A composite periodic analog signal is composed of multiple sine waves.

---

**Figure 3.2** A sine wave

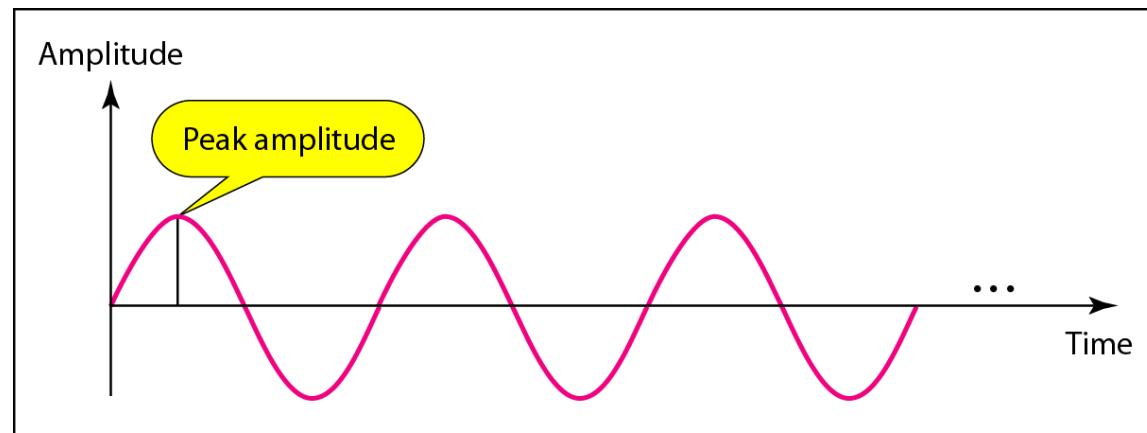
---



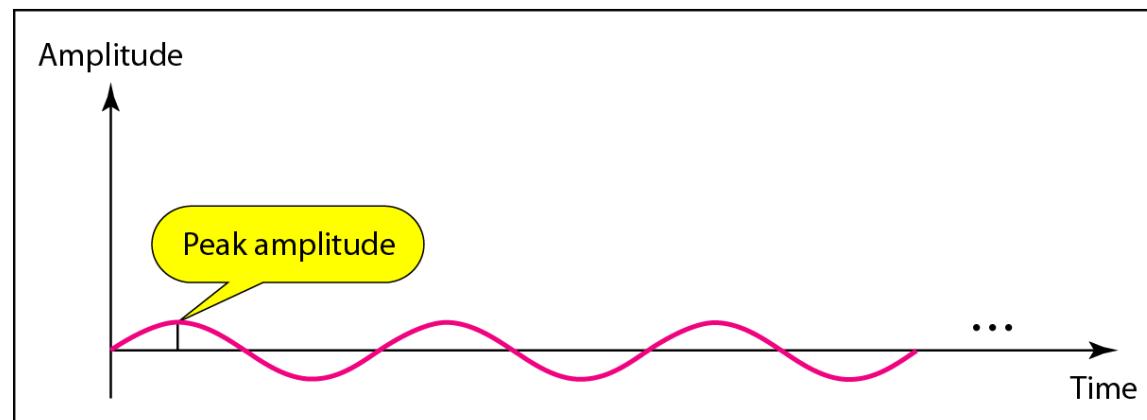
---

A signal which repeats after each certain time interval is known as a periodic signal.

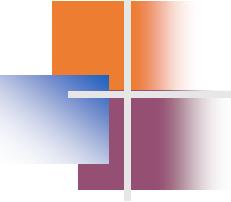
**Figure 3.3** Two signals with the same phase and frequency, but different amplitudes



a. A signal with high peak amplitude



b. A signal with low peak amplitude



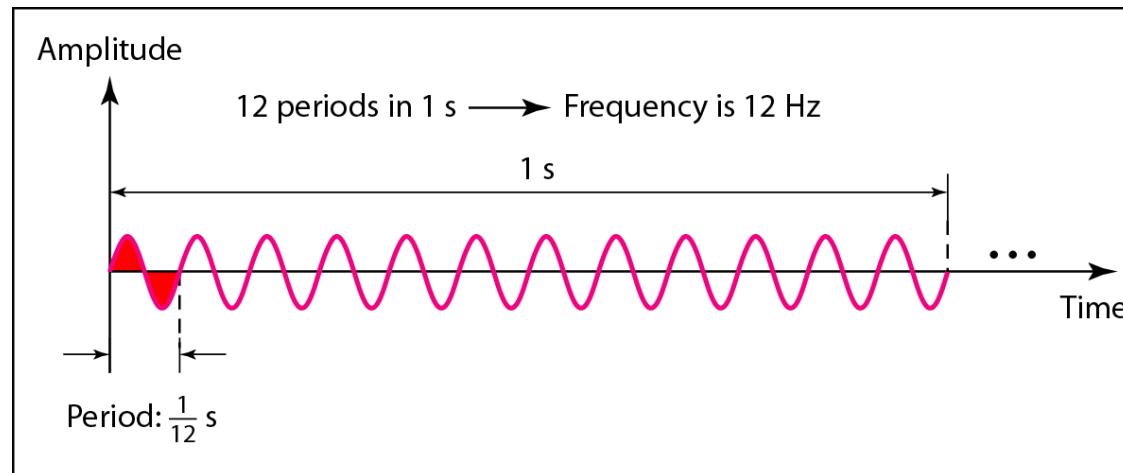
---

Frequency and period are the inverse of each other.

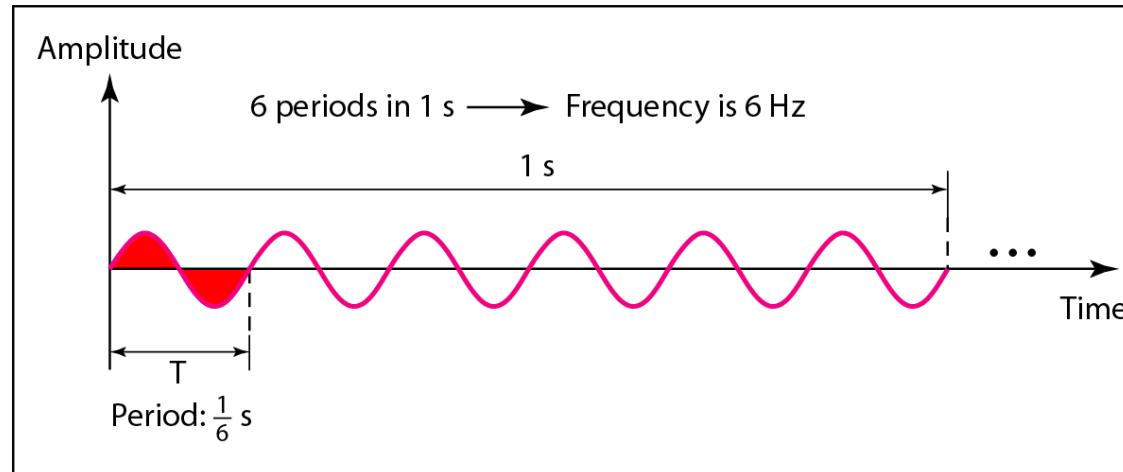
---

$$f = \frac{1}{T} \quad \text{and} \quad T = \frac{1}{f}$$

**Figure 3.4** Two signals with the same amplitude and phase, but different frequencies



a. A signal with a frequency of 12 Hz



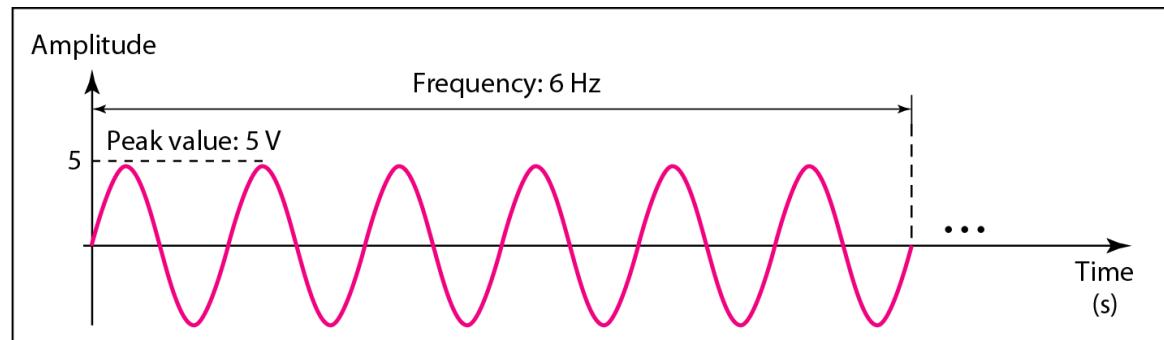
b. A signal with a frequency of 6 Hz

# Analog and Digital Data

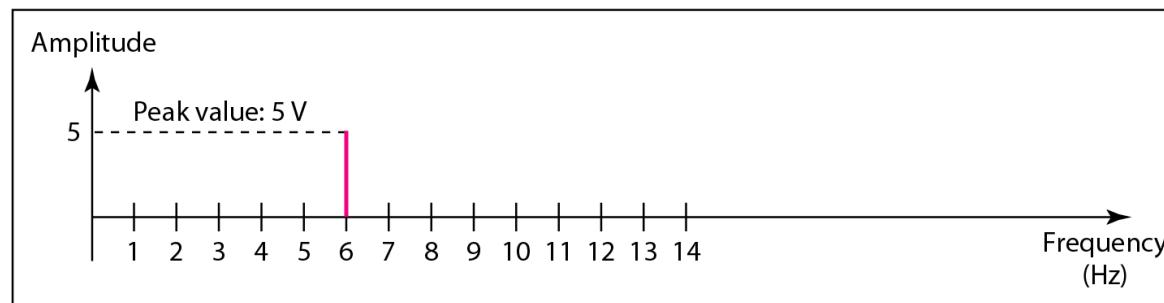
- **Data can be analog or digital.**
  - The term analog data refers that is continuous;
  - Digital data refers to information that has discrete states.
- For a example, an analog clock and has hour, minute, and seconds
- Digital clock that reports the hours and the minutes will change suddenly from 8:05 to 8:06
- Analog data, such as sounds made by a human voice, take on continuous values. When someone speaks, an analog wave is created in the air.

## Time-domain and frequency-domain plots of a sine wave

A complete sine wave in the time domain can be represented by one single spike in the frequency domain.



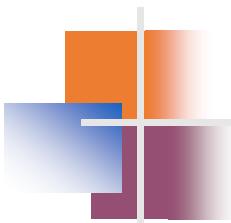
a. A sine wave in the time domain (peak value: 5 V, frequency: 6 Hz)



b. The same sine wave in the frequency domain (peak value: 5 V, frequency: 6 Hz)

<i>Unit</i>	<i>Equivalent</i>	<i>Unit</i>	<i>Equivalent</i>
Seconds (s)	1 s	Hertz (Hz)	1 Hz
Milliseconds (ms)	$10^{-3}$ s	Kilohertz (kHz)	$10^3$ Hz
Microseconds ( $\mu$ s)	$10^{-6}$ s	Megahertz (MHz)	$10^6$ Hz
Nanoseconds (ns)	$10^{-9}$ s	Gigahertz (GHz)	$10^9$ Hz
Picoseconds (ps)	$10^{-12}$ s	Terahertz (THz)	$10^{12}$ Hz

### Units of period and frequency



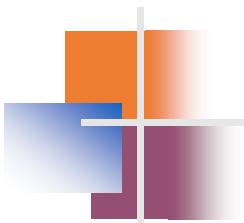
## Example 3.3

The power we use at home has a frequency of **60 Hz**. The period of this sine wave can be determined as follows:

$$T = \frac{1}{f} = \frac{1}{60} = 0.0166 \text{ s} = 0.0166 \times 10^3 \text{ ms} = 16.6 \text{ ms}$$

This means that the period of the power for our lights at home is 0.0116s or 16.6 ms.

Our eyes are not sensitive enough to distinguish these rapid changes in amplitude.



## Example 3.4

Express a period of 100 ms in microseconds.

### Solution

From Table 3.1 we find the equivalents of 1 ms (1 ms is  $10^{-3}$  s) and 1 s (1 s is  $10^6$   $\mu$ s). We make the following substitutions::

$$100 \text{ ms} = 100 \times 10^{-3} \text{ s} = 100 \times 10^{-3} \times 10^6 \mu\text{s} = 10^2 \times 10^{-3} \times 10^6 \mu\text{s} = 10^5 \mu\text{s}$$

### **Example 3.5**

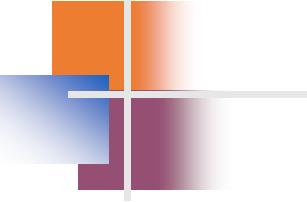
The period of a signal is 100 ms. What is its frequency in kilohertz?

### **Solution**

First we change 100 ms to seconds, and then we calculate the frequency from the period ( $1 \text{ Hz} = 10^{-3} \text{ kHz}$ ).

$$100 \text{ ms} = 100 \times 10^{-3} \text{ s} = 10^{-1} \text{ s}$$

$$f = \frac{1}{T} = \frac{1}{10^{-1}} \text{ Hz} = 10 \text{ Hz} = 10 \times 10^{-3} \text{ kHz} = 10^{-2} \text{ kHz}$$



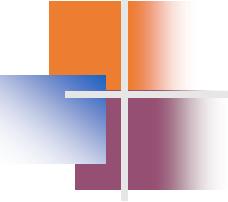
## Note

---

Frequency is the number of cycles per seconds.

Change in a short span of time means high frequency.

Change over a long span of time means low frequency.



If a signal does not change at all, its frequency is zero.

If a signal changes instantaneously, its frequency is infinite.

Composite Signal is the result of superposition of large number of pure sine waves having different frequencies

And different amplitude.

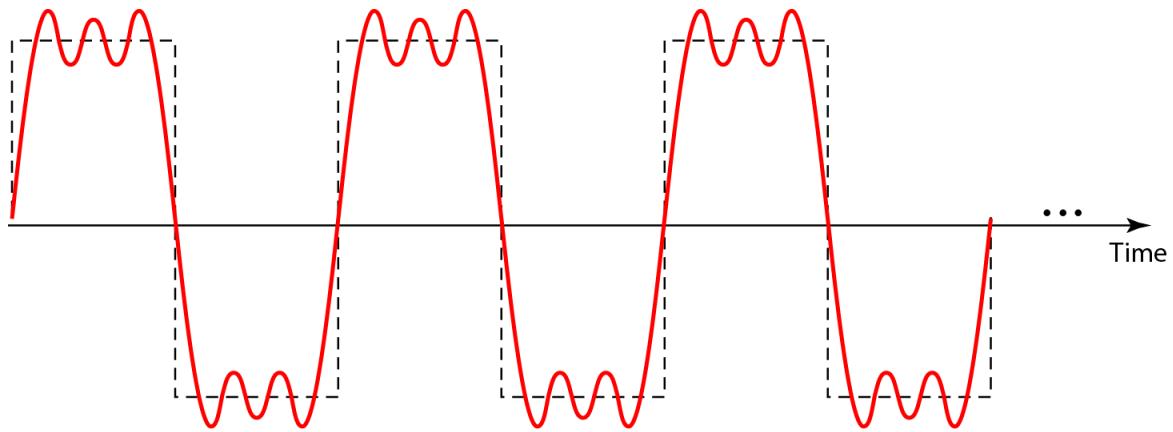
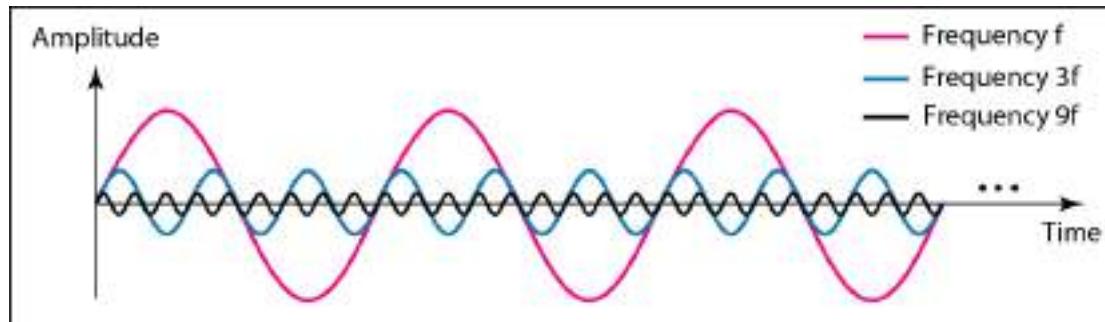
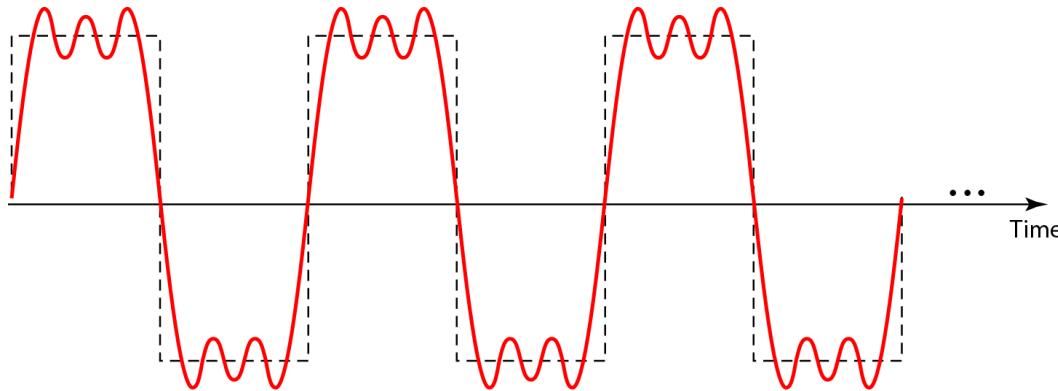


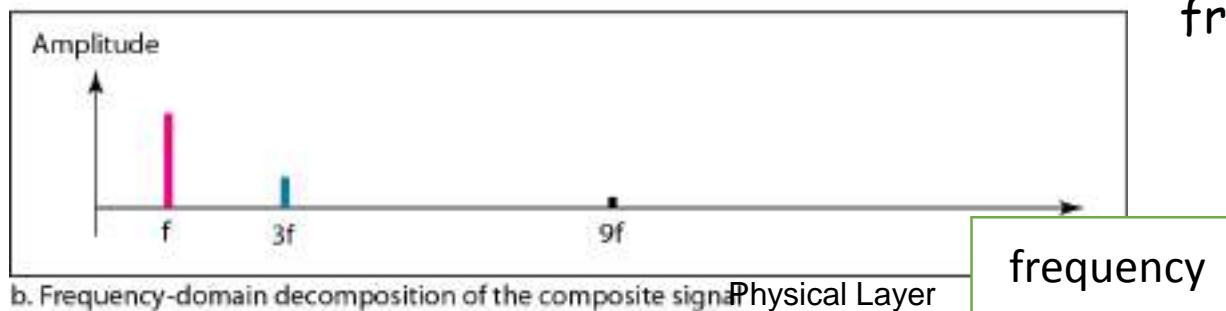
Figure 3.9 A composite periodic signal

# A composite periodic signal



a. Time-domain decomposition of a composite signal

Decomposition of the composite periodic signal in the time and frequency domains

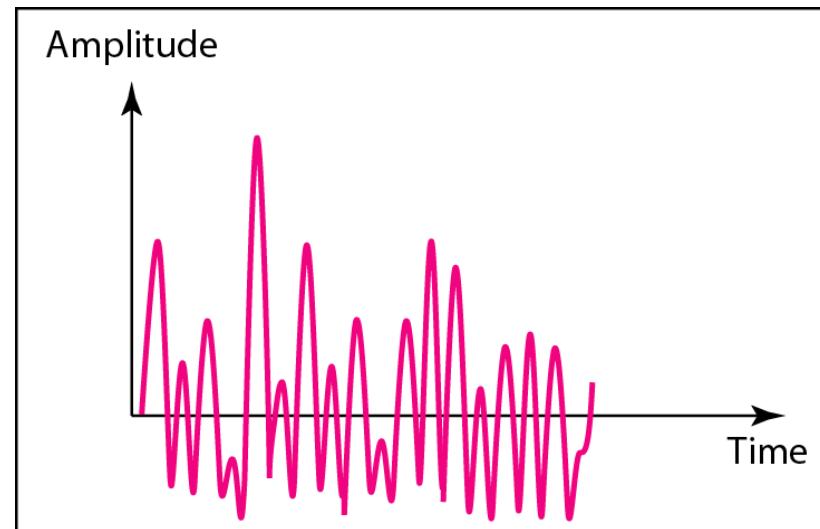


b. Frequency-domain decomposition of the composite signal

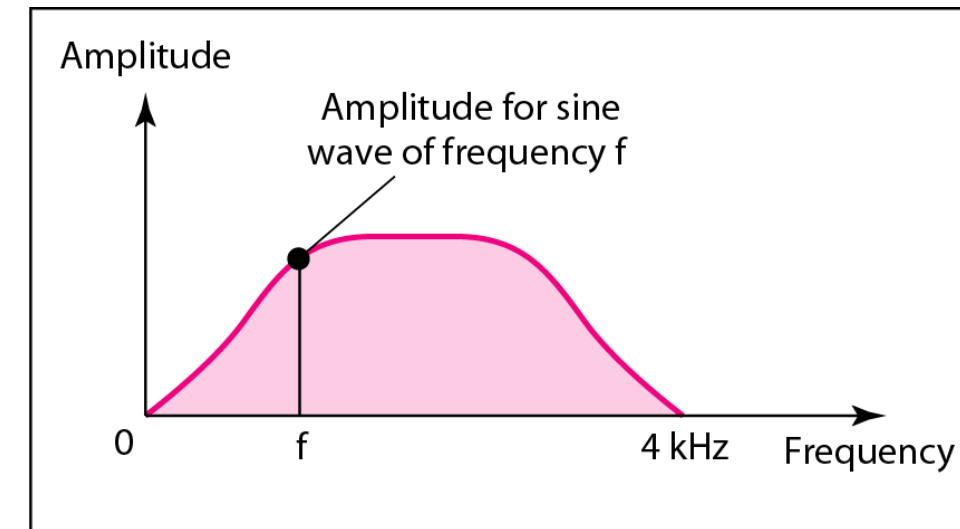
Physical Layer

## Time and frequency domains of a nonperiodic signal

- ❑ A non-periodic signal changes without showing fixed cycle that repeats any time.
- ❑ A nonperiodic composite signal
  - It can be a signal created by a **microphone** or a **telephone** set when a word is pronounced.

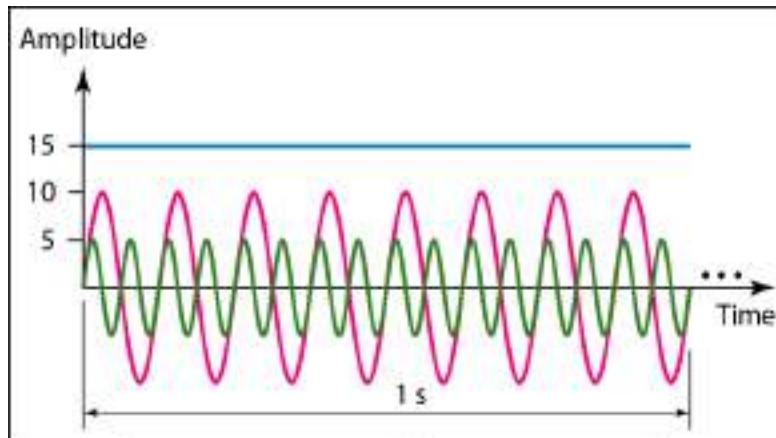


a. Time domain

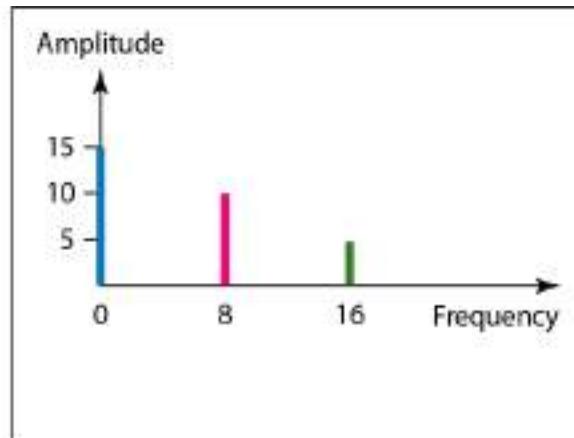


b. Frequency domain

# Frequency Domain



a. Time-domain representation of three sine waves with frequencies 0, 8, and 16



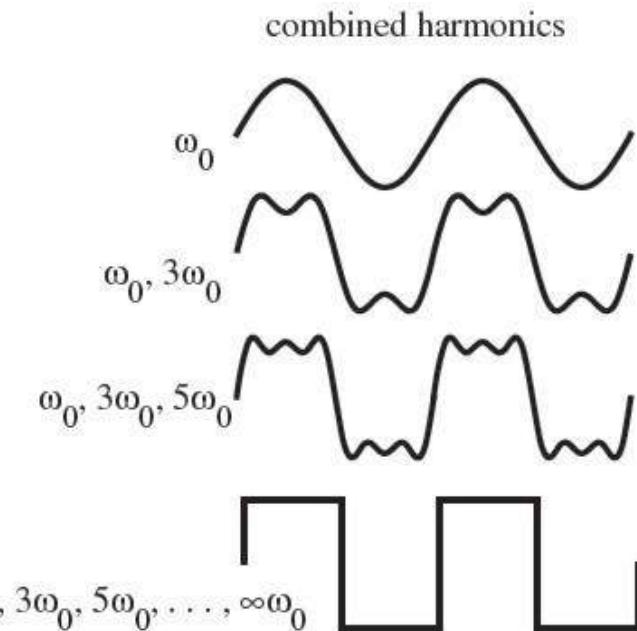
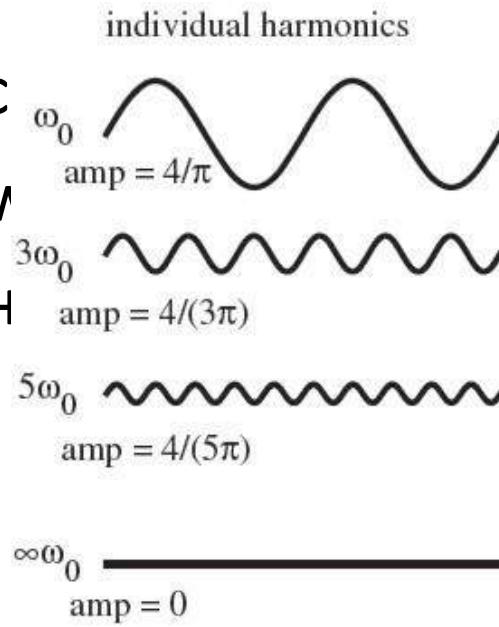
b. Frequency-domain representation of the same three signals

- The frequency domain is more compact and useful when we are dealing with more than one sine wave.
- A single-frequency sine wave is not useful in data communication
  - We need to send a **composite signal**, a signal made of many simple sine waves.

# Fourier analysis

Superposition position of infinite number of pure sinusoidal waves of different frequencies and pure sine waves or cosine number of different amplitude results the formation of the signal which is periodic.

- $f(x,t) = a_0 + a_1 \cos \omega t + a_2 \cos 2\omega t + a_3 \cos 3\omega t + \dots + a_n \cos n\omega t + b_1 \sin \omega t + b_2 \sin 2\omega t + b_3 \sin 3\omega t + \dots + b_n \sin n\omega t + \dots$



# *Fourier analysis*

---

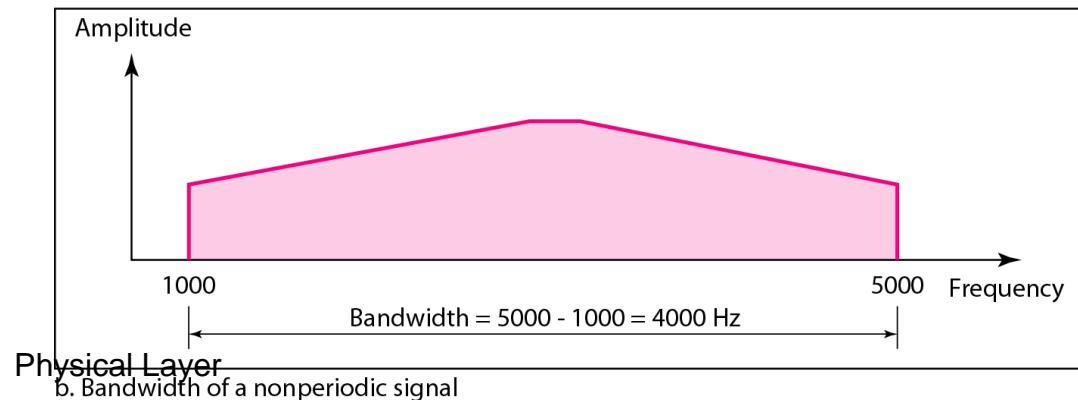
According to Fourier analysis,  
any composite signal is a combination of simple sine  
waves with different frequencies, amplitudes, and phases.

- If the composite signal is **periodic**, the decomposition gives a series of signals with discrete frequencies;
- If the composite signal is **nonperiodic**, the decomposition gives a combination of sine waves with continuous frequencies.

# Bandwidth

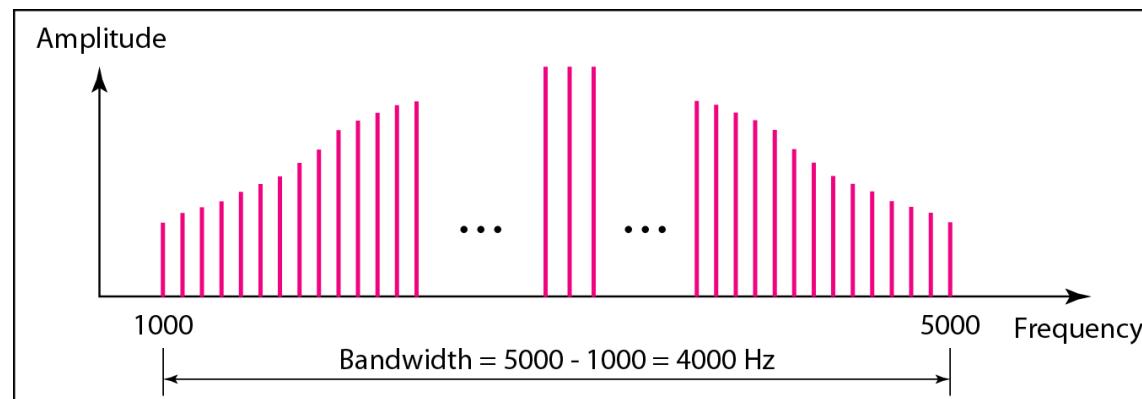
The bandwidth of a **composite signal** is the **difference between the highest and the lowest** frequencies contained in that signal.

When we decompose non-periodical signal into simple harmonic waves (sine-waves) the frequencies are found to be **1000Hz**, **1000.1Hz**, **1000.2Hz.....5000Hz** in continues range.



# *Bandwidth*

When we decompose periodic signal into simple harmonic waves (sine-waves) the frequencies are found to be **1000Hz, 1001Hz, 1002Hz.....5000Hz** in discrete.



a. Bandwidth of a periodic signal

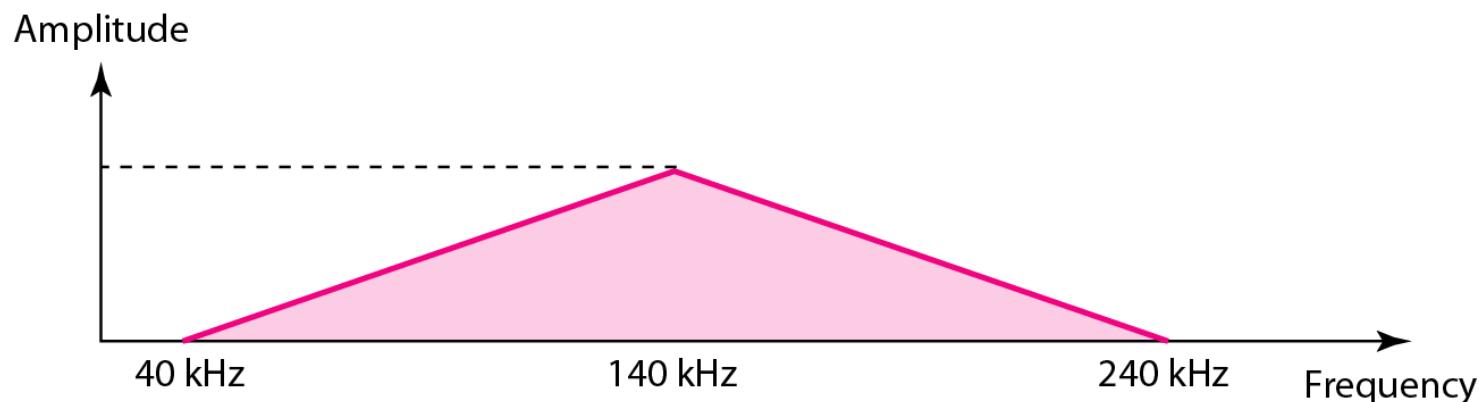
# Example

---

A nonperiodic composite signal has a bandwidth of 200 kHz, with a middle frequency of 140 kHz and peak amplitude of 20 V. The two extreme frequencies have an amplitude of 0. Draw the frequency domain of the signal.

## Solution

The lowest frequency must be at 40 kHz and the highest at 240 kHz.



## Example 3.10

If a periodic signal is decomposed into five sine waves with frequencies of 100, 300, 500, 700, and 900 Hz, what is its bandwidth? Draw the spectrum, assuming all components have a amplitude of 10 V.

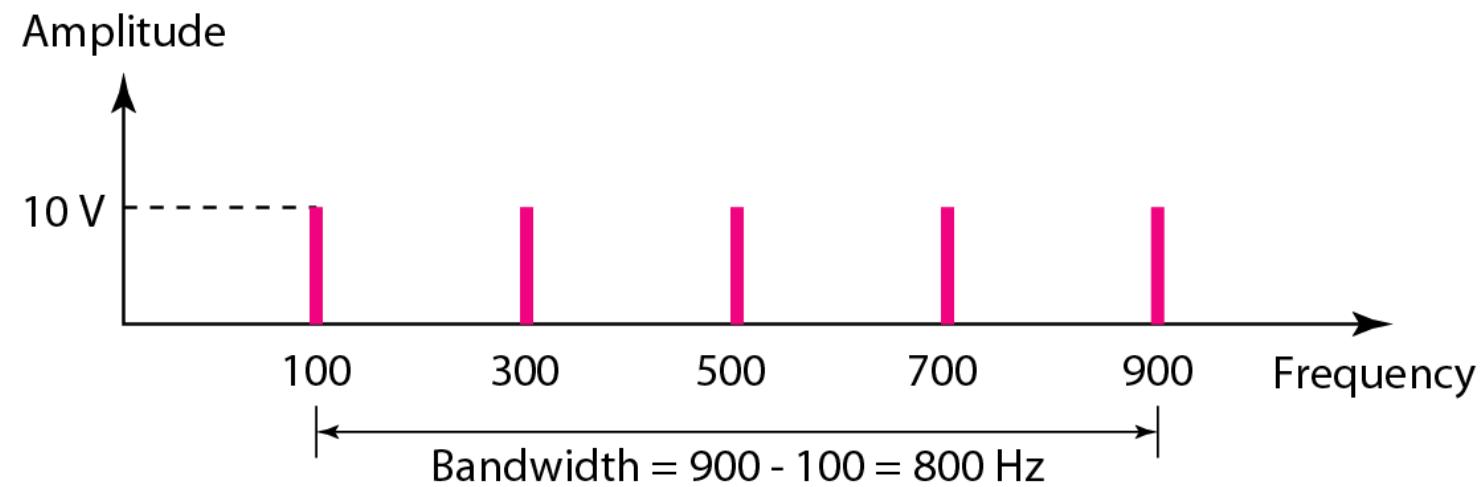
### Solution

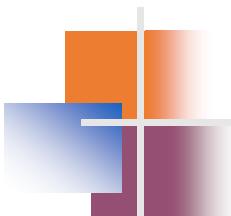
Let  $f_h$  be the highest frequency,  $f_l$  the lowest frequency, and  $B$  the bandwidth. Then

$$B = f_h - f_l = 900 - 100 = 800 \text{ Hz}$$

The spectrum has only five spikes, at 100, 300, 500, 700, and 900 Hz (see Figure 3.13).

**Figure 3.13** The bandwidth for Example 3.10





## Example 3.11

A periodic signal has a bandwidth of 20 Hz. The highest frequency is 60 Hz. What is the lowest frequency? Draw the spectrum if the signal contains all frequencies of the same amplitude.

### Solution

Let  $f_h$  be the highest frequency,  $f_l$  the lowest frequency, and  $B$  the bandwidth. Then

$$B = f_h - f_l \Rightarrow 20 = 60 - f_l \Rightarrow f_l = 60 - 20 = 40 \text{ Hz}$$

The spectrum contains all integer frequencies. We show this by a series of spikes.

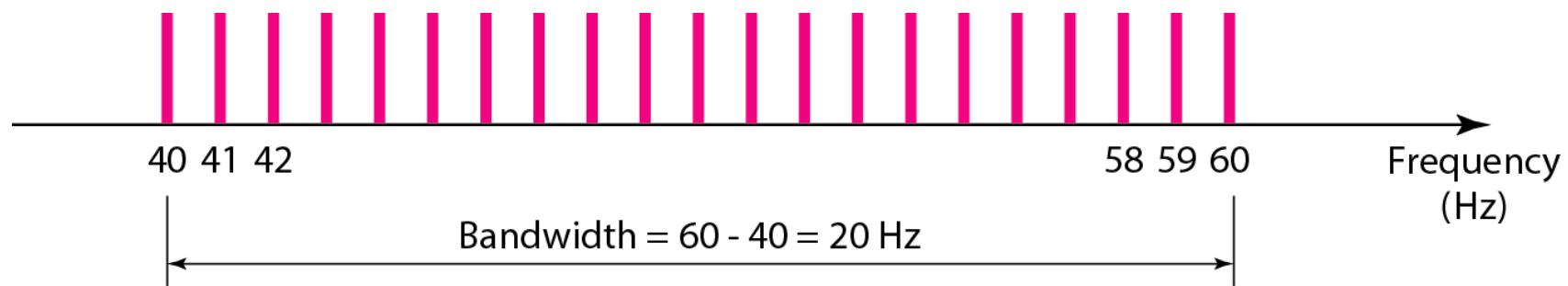
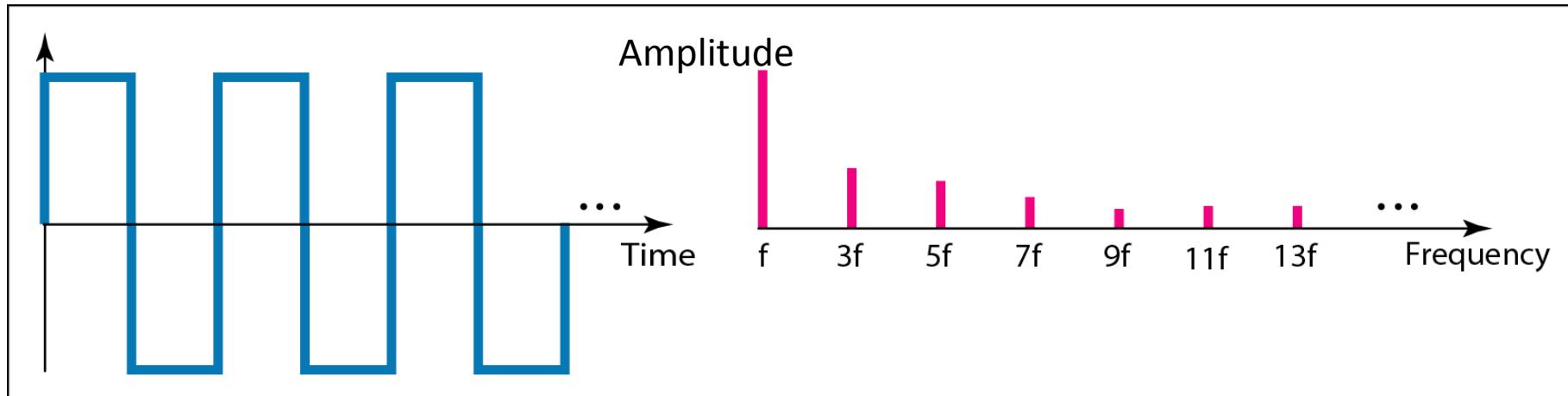
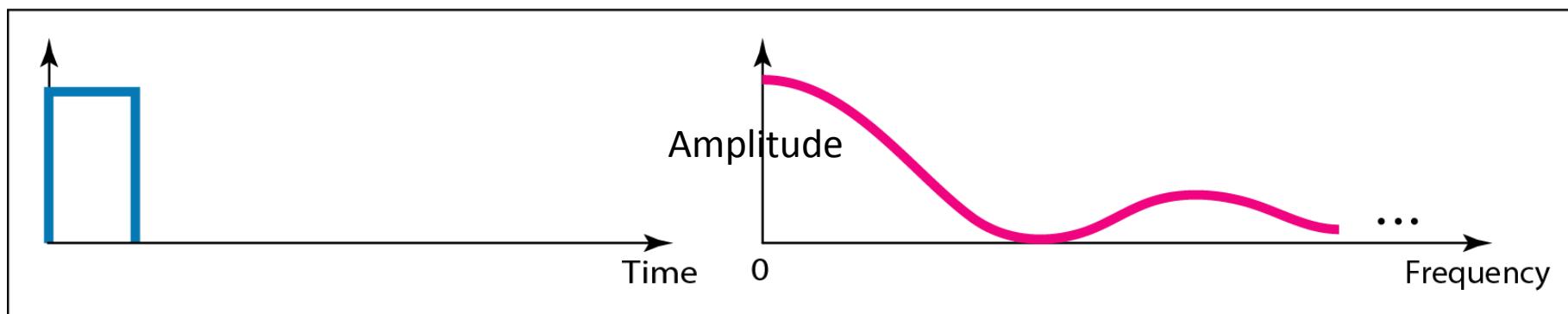


Figure 3.14 The bandwidth for Example 3.11

**Figure 3.17** The time and frequency domains of periodic and nonperiodic digital signals



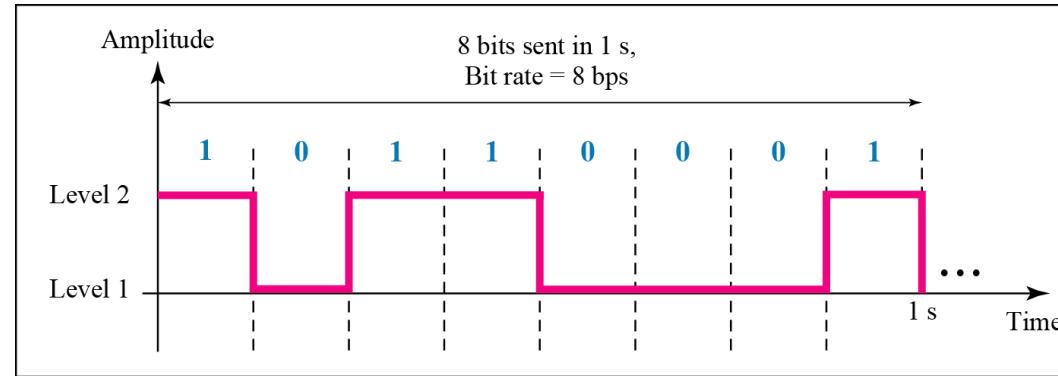
a. Time and frequency domains of periodic digital signal



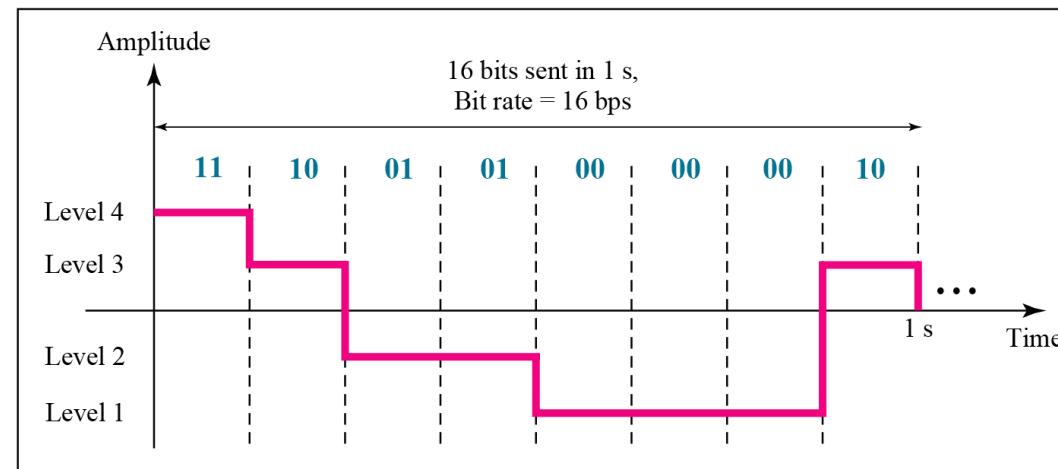
b. Time and frequency domains of nonperiodic digital signal

# Digital Signal

- represented by a digital signal, in the form of bits where
  - 1 represent positive voltage and
  - 0 as zero voltage.
- A digital signal can have more than **two levels**.
- That is we can send more than 1 bit for each level.



a. A digital signal with two levels



b. A digital signal with four levels

We can send 1 bit per level and another down before we can send 2 bits per level.

if a signal has levels, each level needs

$$\log_2 L \text{ bits}$$

- **Problem:** A digital signal has eight levels. How many bits are needed per level ? We calculate the number of bits from the formula.

$$\text{Number of bits per level} = \log_2 8 = 3$$

*Each signal level is represented by 3 bits*

- A digital signal has nine levels. How many bits are needed per level ?  
We calculate the number of bits by using the formula.

Each signal level is represented by 3.17 bits. However, this answer is not realistic.

The number of bits sent per level needs to be an integer as well as a power of 2.

## Bit Rate:

- Most **digital signals are non periodic**, and thus period and frequency are not appropriate characteristics.
- Another term – **bit rate** (instead of frequency) is used to describe digital signals.
- The **bit rate** is the number of bits sent in 1s and is expressed in **bits per second (bps)**.

- **Problem 1 :** Assume we need to download text documents at the rate of 100 pages per second. What is the required bit rate of the channel ?

### Solution

A page is an average of 24 lines with 80 characters in each line. If we assume that one character requires 8 bits, the bit rate is

$$100 * 24 * 80 * 8 = 1,536,000 \text{ bps} = 1.536 \text{ Mbps}$$

What is the bit rate of high definition TV (HDTV) ?

### Solution

- The HDTV screen is normally a ratio of 16 : 9.
- There are 1920 by 1080 pixels per screen,
- the screen is renewed 30 times per second.
- 24 bits represents one color pixel.

$$1920 \times 1080 \times 30 \times 24 = 1,492,992,000 \text{ or } 1.5 \text{ Gbps}$$

The TV stations reduce this rate to 20 to 40 Mbps through compression.

# Bit length

- Wavelength for an analog signal: the distance one cycle occupies on the transmission medium. (*Just for understanding*)
- In case of Digital signal, Bit length is the distance one bit occupies on the transmission medium.

Bit length = propagation speed \* bit duration

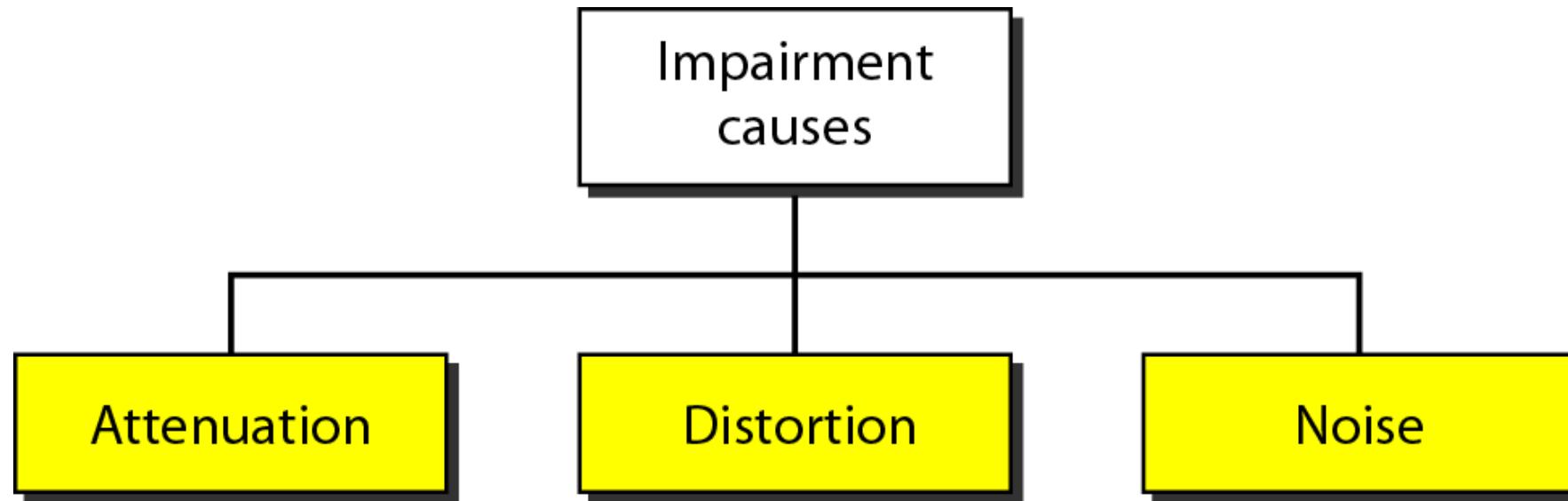
# Transmission impairment

- Signals travel through transmission media which actually is not perfect.
- The imperfection causes signal impairment.
- This means that the signal at the **beginning of the medium** is not that same as the signal at the **end of the medium**.
- There causes of impairment are
  - Attenuation
  - Distortion
  - Noise.

---

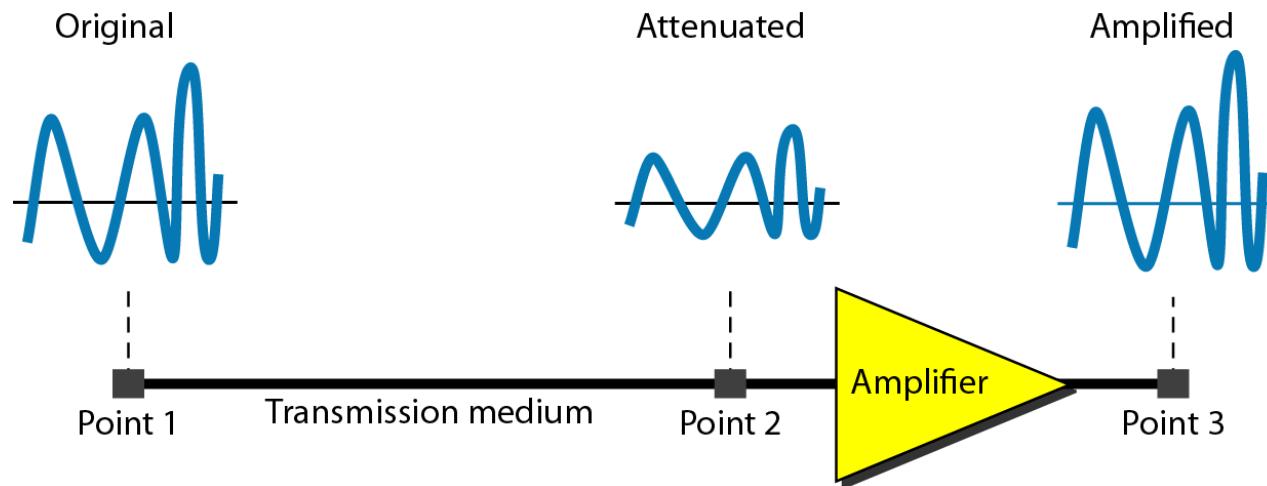
**Figure 3.25** Causes of impairment

---



# Transmission impairment

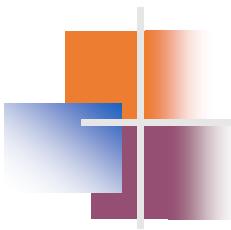
- **Attenuation :**
  - loss of energy.
  - When a signal (simple or composite) travels through a medium, it losses some of its energy in overcoming the resistance of the medium.
  - Electric signals warm, if not hot.
- To compensate this loss amplifiers are used.



- **Decibel :**
  - To show the signal has lost or gained strength, the unit of the decibel.
  - The decibel (dB) measures the **relative strengths** of two signals or one signal at the different points.
  - The decibel is negative if a signal is attenuated and positive if the a signal is amplified.

$$dB = 10 \log_{10} \left( \frac{P_2}{P_1} \right)$$

Where  $P_1$  and  $P_2$  are the powers of a signal at point 1 & 2 respect.



## Example 3.26

→ Suppose a signal travels through a transmission medium and its power is reduced to one-half.

This means that  $P_2$  is  $(1/2)P_1$ . In this case, the **attenuation** (loss of power) can be calculated as

$$\text{dB} = 10 \log_{10} \frac{P_2}{P_1}$$

$$= 10 \log_{10} \frac{0.5P_1}{P_1}$$

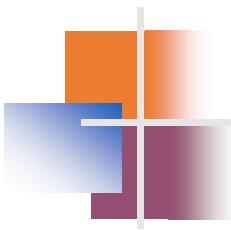
$$= 10 \log_{10} 0.5$$

$$= 10(-0.3)$$

dB

$$= -3 \text{ dB}$$

A loss of 3 dB ( $-3 \text{ dB}$ ) is equivalent to losing one-half the power.



## Example 3.27

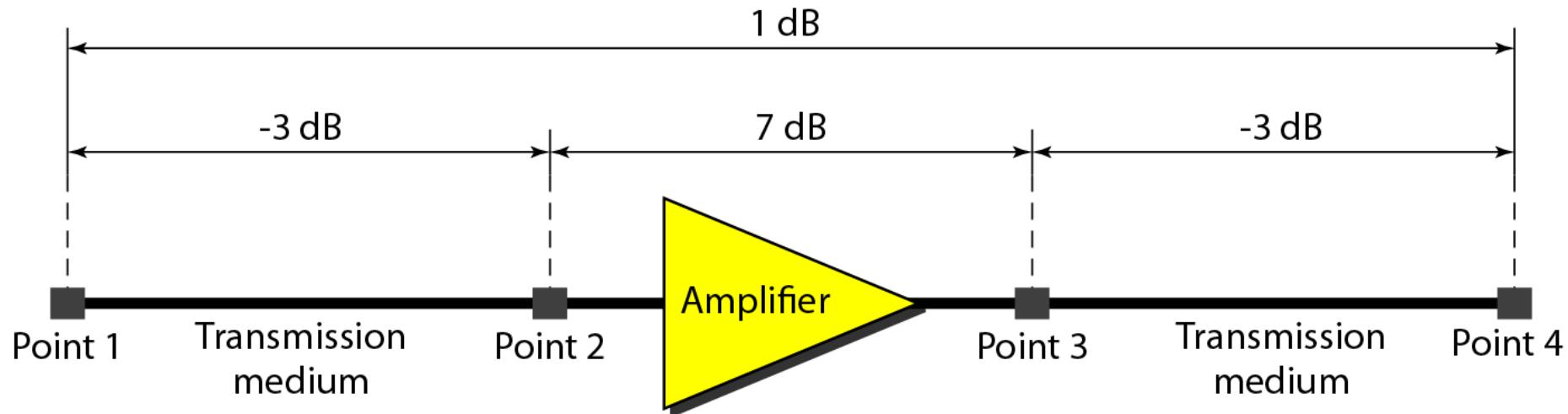
A signal travels through an amplifier, and its power is increased 10 times. This means that  $P_2 = 10P_1$ . In this case, the amplification (gain of power) can be calculated as

$$10 \log_{10} \frac{P_2}{P_1} = 10 \log_{10} \frac{10P_1}{P_1}$$

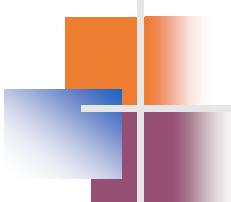
$$= 10 \log_{10} 10 = 10(1) = 10 \text{ dB}$$

## Example 3.28

Decibel is used to measure the changes in the strength of a signal  
decibel numbers can be added (or subtracted)



$$dB = -3 + 7 - 3 = +1$$



## Example 3.29

Sometimes the decibel is used to measure signal power in milliwatts.

In this case, it is referred to as  $\text{dB}_m$  and is calculated as

$$\text{dB}_m = 10 \log_{10} P_m$$

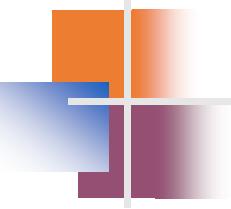
where  $P_m$  is the power in milliwatts.

Calculate the power of a signal with  $\text{dB}_m = -30$ .

**Solution**

We can calculate the power in the signal as

$$\begin{aligned}\text{dB}_m &= 10 \log_{10} P_m = -30 \\ \log_{10} P_m &= -3 \quad P_m = 10^{-3} \text{ mW}\end{aligned}$$



## Example 3.30

The loss in a cable is usually defined in decibels per kilometer (dB/km).

If the signal at the beginning of a cable with  $-0.3$  dB/km has a power of 2 mW, what is the power of the signal at 5 km?

### Solution

The loss in the cable in decibels is  $5 \times (-0.3) = -1.5$  dB. We can calculate the power as

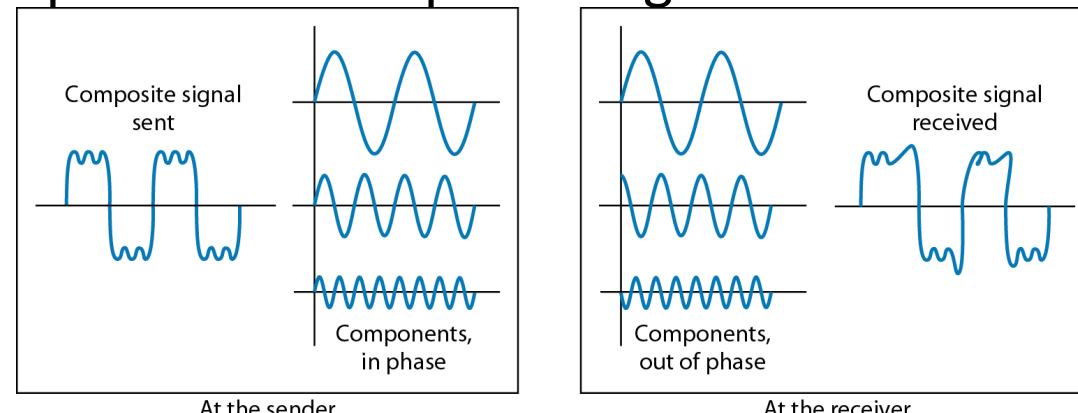
$$\text{dB} = 10 \log_{10} \frac{P_2}{P_1} = -1.5$$

$$\frac{P_2}{P_1} = 10^{-0.15} = 0.71$$

$$P_2 = 0.71P_1 = 0.7 \times 2 = 1.4 \text{ mW}$$

# Distortion :

- Distortion means that the **signal changes its form or shape**.
- Distortion can occur in a composite signal.
- Since each signal component has its own **propagation speed**, through a medium and therefore its own **delay in arriving** at the final destination thus result in distortion.
- In other words components at the receiver have **phases different** from what they had at the sender. The shape of the composite signal is therefore not the same.



# Noise

- **Noise** : Several types of noise, such as thermal noise, indeed noise, crosstalk and impulse noise may corrupt the signal.
  - **Thermal noise** → Caused by **random motion of electrons** in a wire which creates extra signal not originally sent by the transmitter.
  - **Induced noise** → Caused by sources such as motors and applications. These devices act as a sending antenna, and the transmission medium acts as the receiving antenna.

- **Cross Talk** → Its an effect of one wire on the other. One wire acts as a sending antenna and the other as the receiving antenna. Example of *Twisted pair cable* .

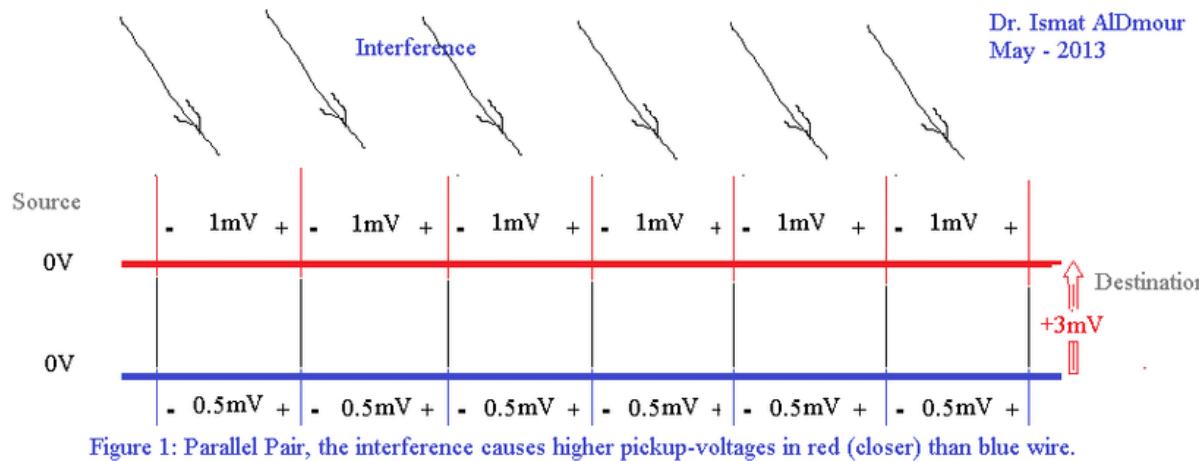


Figure 1: Parallel Pair, the interference causes higher pickup-voltages in red (closer) than blue wire.

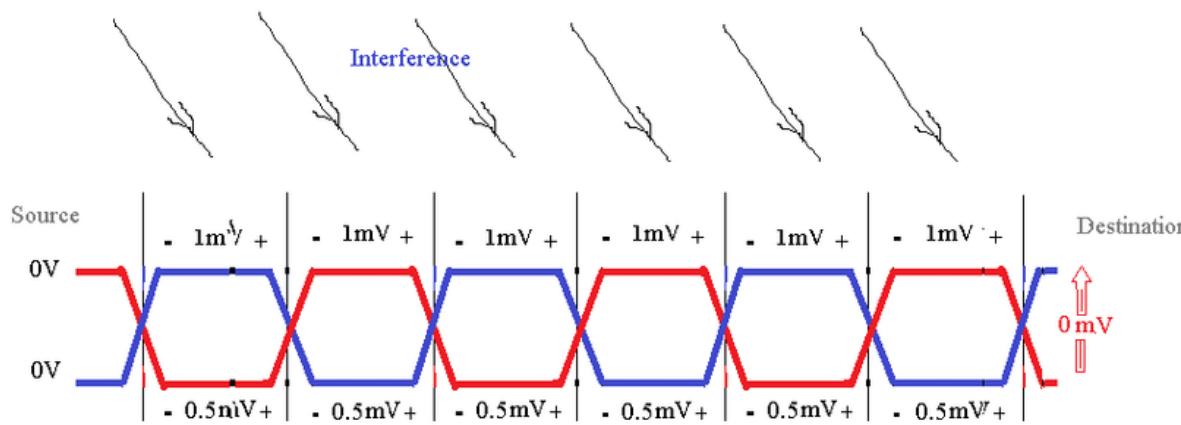
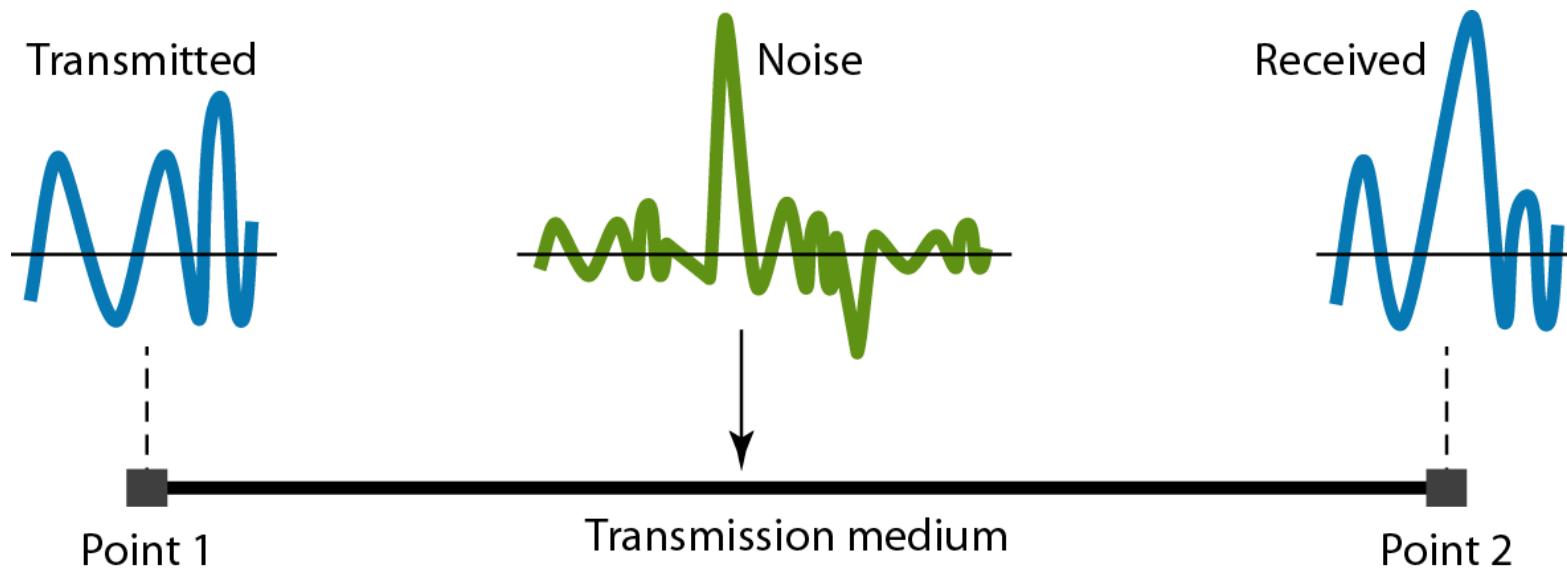


Figure 2: Twisted Pair, the interference alternatively produces high and low pickup-voltages in both wires.

- **Impulse noise** → is the spike (a signal with high energy in a very short time) shows the effect of noise on a signal such as lightening.

Figure 3.29 Noise

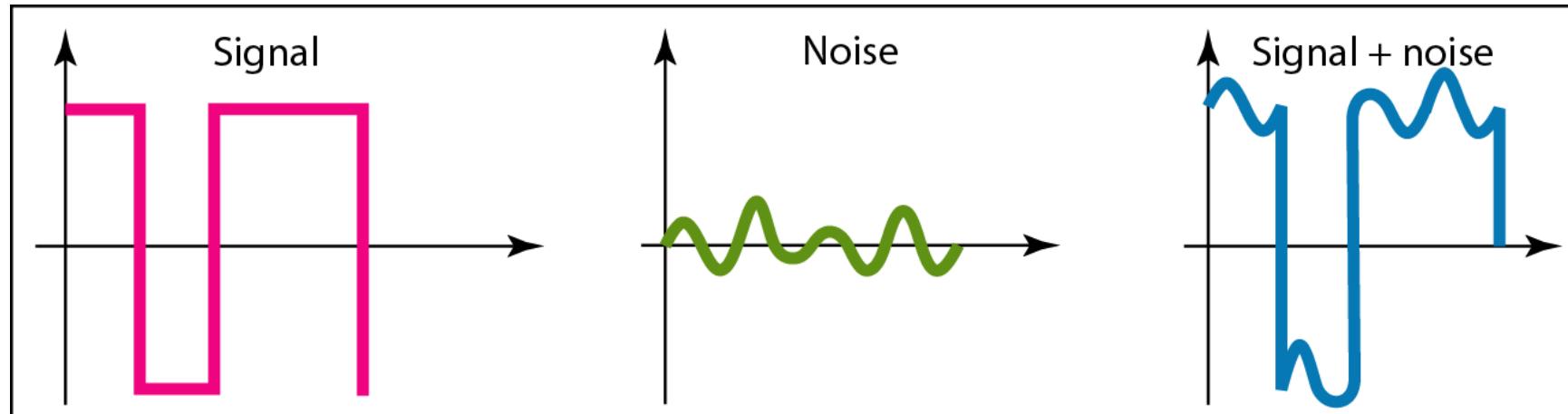


# Signal to Noise Ratio (SNR)

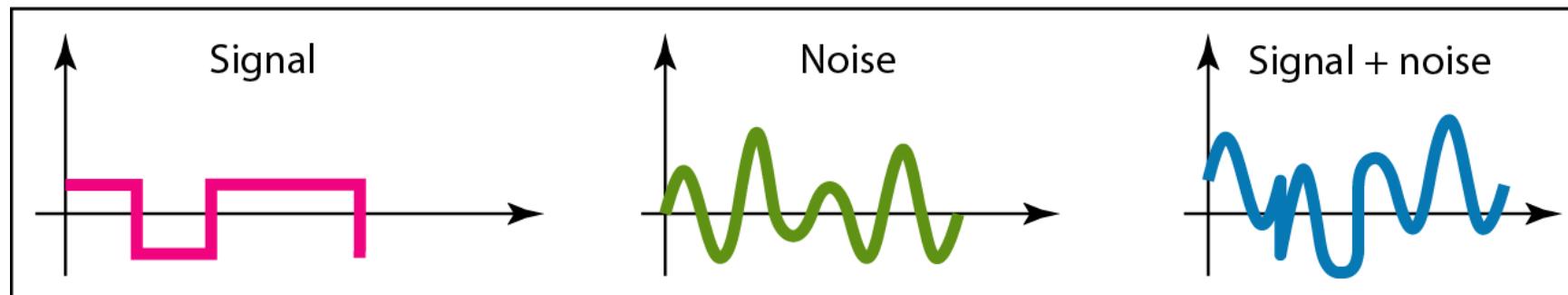
- SNR is actually the **ratio of what is wanted (signal) to what is not wanted (noise).**
- A high SNR means the signal is less corrupted by noise; a low SNR means the signal is more corrupted by noise.
- The signal to noise ratio is defined as
  - High SNR → Signal is **less corrupted**
  - Low SNR → Signal is more corrupted by noise
  - Because SNR is the ratio of two powers, it is often described in decibel units
- $\text{SNR}_{\text{db}} = 10 \log_{10} \text{SNR}$ .

$$\text{SNR} = \frac{\text{average signal power}}{\text{average noise power}}$$

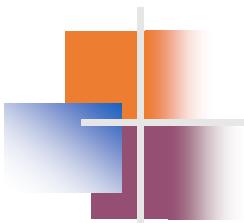
Figure 3.31 Two cases of SNR: a high SNR and a low SNR



a. Large SNR



b. Small SNR



## Example 3.31

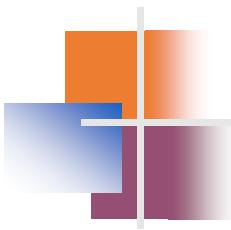
The power of a signal is 10 mW and the power of the noise is 1 μW; what are the values of SNR and  $\text{SNR}_{\text{dB}}$ ?

Solution

The values of SNR and  $\text{SNR}_{\text{dB}}$  can be calculated as follows:

$$\text{SNR} = \frac{10,000 \mu\text{W}}{1 \mu\text{W}} = 10,000$$

$$\text{SNR}_{\text{dB}} = 10 \log_{10} 10,000 = 10 \log_{10} 10^4 = 40$$



## Example 3.32

The values of SNR and SNR<sub>dB</sub> for a noiseless channel are

$$\text{SNR} = \frac{\text{signal power}}{0} = \infty$$
$$\text{SNR}_{\text{dB}} = 10 \log_{10} \infty = \infty$$

We can never achieve this ratio in real life; it is an ideal.

# Data Rate Limits

- Concern is all about **how fast we can send data, in bits per second, over a channel.**
- Data rate depends on three factors.
  - **Bandwidth available**
  - **Level of signals we use**
  - **The quality of the channel (the level of noise)**
- Two theoretical formulas were developed to calculate the data rate:
  - Nyquist for a noiseless channel,
  - Shannon for a noisy channel.

# Noiseless Channel : Nyquist Bit Rate

- For **noiseless channel**, the Nyquist Bit Rate formula defines the theoretical maximum bit rate.

$$\text{Bit Rate} = 2 \times \text{Bandwidth} \times \log_2 L$$

where

- Bit Rate express in bps
- Bandwidth express in Hz
- L is the number of signal level used to represent data

- the idea is **theoretically correct, practically there is a limit.**

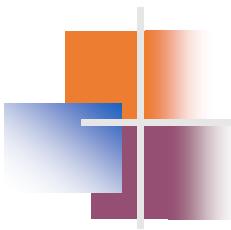


## Example 3.34

Consider the same noiseless channel with a bandwidth of 3000 Hz transmitting a signal with two signal levels. The maximum bit rate can be calculated as

$$\text{BitRate} = 2 \times 3000 \times \log_2 2 = 6000 \text{ bps}$$

**Bit Rate = 2 x Bandwidth x Log<sub>2</sub>L**



## Example 3.36

We need to send 265 kbps over a noiseless channel with a bandwidth of 20 kHz. How many signal levels do we need?

### Solution

We can use the Nyquist formula as shown:

$$265,000 = 2 * 20,000 * \log_2 L$$

$$\log_2 L = 6.625$$

$$L = 2^{6.625}$$

$$= 98.7 \text{ levels}$$

Since, this results is not a power of 2, we need to either increase the number of levels or reduce the bit rate. If we have 128 levels, the bit rate is 280 kbps. If we have 64 levels, the bit rate is 240 kbps

# Noisy Channel : Shannon Capacity

- In reality, **we cannot have a noiseless channel**; the channel is always noisy.
- Shannon Capacity determines the **theoretical highest rate for a noisy channel**.

$$\text{Capacity} = \text{bandwidth} * \log_2 (1 + \text{SNR})$$

where

bandwidth is bandwidth of channel ,

SNR is the signal-to-noise ratio, and

capacity is the capacity of the channel in bits per second.

- Note that in the Shannon formula there is no indication of the signal level, which means that no matter how many levels we have.



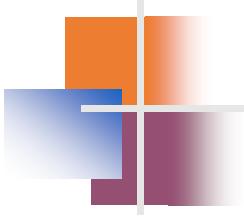
## Example 3.37

Consider an extremely noisy channel in which  
**the value of the signal-to-noise ratio is almost zero.**  
In other words, the noise is so strong that the signal is faint.

For this channel the capacity C is calculated as

$$C = B \log_2 (1 + \text{SNR}) = B \log_2 (1 + 0) = B \log_2 1 = B \times 0 = 0$$

This means that the capacity of this channel is zero regardless of the bandwidth. **In other words, we cannot receive any data through this channel.**



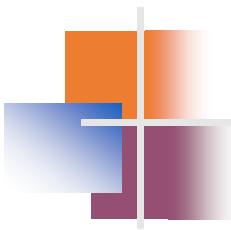
## Example 3.38

We can calculate the theoretical highest bit rate of a regular telephone line. A telephone line normally has a bandwidth of 3000. The signal-to-noise ratio is usually 3162. For this channel the capacity is calculated as

$$\begin{aligned}C &= B \log_2 (1 + \text{SNR}) = 3000 \log_2 (1 + 3162) = 3000 \log_2 3163 \\&= 3000 \times 11.62 = 34,860 \text{ bps}\end{aligned}$$

This means that the highest bit rate for a telephone line is 34.860 kbps.

If we want to send data faster than this, we can either increase the bandwidth of the line or improve the signal-to-noise ratio.



## Example 3.39

The signal-to-noise ratio is often given in decibels.

Assume that

$$\text{SNR}_{\text{dB}} = 36 \text{ and}$$

the channel bandwidth is 2 MHz. The

theoretical channel capacity can be calculated as

$$\text{SNR}_{\text{dB}} = 10 \log_{10} \text{SNR} \rightarrow \text{SNR} = 10^{\text{SNR}_{\text{dB}}/10} \rightarrow \text{SNR} = 10^{3.6} = 3981$$

$$C = B \log_2 (1 + \text{SNR}) = 2 \times 10^6 \times \log_2 3982 = 24 \text{ Mbps}$$

# Performance

- Bandwidth
- **Bandwidth in Hertz**
  - Bandwidth in hertz is the **range of frequencies contained in a composite signal** or the range of frequencies a channel can pass.
- **Bandwidth in Bits per Seconds**
  - Bandwidth can also refer to the **number of bits per second** that a channel, link, or even a network can transmit.
  - Eg Fast Ethernet network is a maximum of 100 Mbps.
  - This means that this network can send 100 Mbps

# Throughput

- It is the measure of how fast we can actually send data through a network.
- Although **bandwidth in bits per second** and **throughput seems the same**, they are different.
- For example, assume a link having
  - **B bps bandwidth**, but can only sent **T bps through this link**
  - **T is always less than B**.
- Hence, the **bandwidth is a potential measurement of the link** whereas **throughput is an actual measurement of how fast we can send data**.

- For example,
  - bandwidth of 1 Mbps,
  - but the devices connected to the end of the link may handle only 200 kbps.
  - This means that we cannot send more than 200 kbps through the link



## Example 3.44

A network with bandwidth of 10 Mbps can pass only an average of 12,000 frames per minute with each frame carrying an average of 10,000 bits. What is the throughput of this network?

### Solution

We can calculate the throughput as

$$\text{Throughput} = \frac{12,000 \times 10,000}{60} = 2 \text{ Mbps}$$

The throughput is almost one-fifth of the bandwidth in this case.

# Latency (delay)

- The latency or delay defines **how long it takes for an entire message to completely arrive at the destination** from the time the first bit is sent out from the source.
- complete time that a **sending and receiving of a message** takes place from a source to a destination.

**Latency** = Propagation of time + transmission time + queuing time + processing delay

- Latency is made of four components:
  - Propagation time,
  - transmission time
  - Queuing time
  - Processing delay

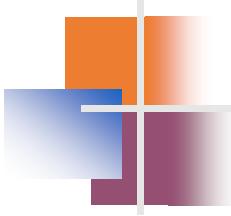
- **Propagation Time** : It measures the time required for **a bit** to travel from the source to the destination:

**Propagation time = distance / Propagation Speed**

The propagation speed of electromagnetic signals depends on the medium and on the frequency of the signal.

For example, in a vacuum, light is propagated with a speed of  $3 * 10^8$  m/s.

It is lower in air; it is much lower in cable.



## Example 3.45

What is the propagation time if the distance between the two points is 12,000 km? Assume the propagation speed to be  $2.4 \times 10^8$  m/s in cable.

### Solution

We can calculate the propagation time as

$$\text{Propagation time} = \frac{12,000 \times 1000}{2.4 \times 10^8} = 50 \text{ ms}$$

The example shows that a bit can go over the Atlantic Ocean in only 50 ms if there is a direct cable between the source and the destination.

- **Transmission Time :**
  - Transmission time is the amount of time from the beginning until the end of a message transmission.
  - In the case of a digital message, it is the time from the first bit until the last bit of a message has left the transmitting node.
  - $\text{Transmission time} = \text{Message size} / \text{Bandwidth}$

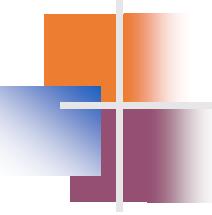


## Example 3.46

What are the propagation time and the transmission time for a 2.5-kbyte message (an e-mail) if the bandwidth of the network is 1 Gbps? Assume that the distance between the sender and the receiver is 12,000 km and that light travels at  $2.4 \times 10^8$  m/s.

### Solution

We can calculate the propagation and transmission time as shown on the next slide:



## Example 3.46 (continued)

$$\text{Propagation time} = \frac{12,000 \times 1000}{2.4 \times 10^8} = 50 \text{ ms}$$

$$\text{Transmission time} = \frac{2500 \times 8}{10^9} = 0.020 \text{ ms}$$

Note that in this case, because the message is short and the bandwidth is high, the dominant factor is the propagation time, not the transmission time. The transmission time can be ignored.

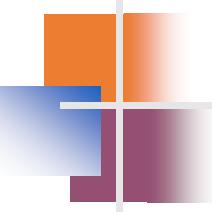


## Example 3.47

What are the propagation time and the transmission time for a 5-Mbyte message (an image) if the bandwidth of the network is 1 Mbps? Assume that the distance between the sender and the receiver is 12,000 km and that light travels at  $2.4 \times 10^8$  m/s.

### Solution

We can calculate the propagation and transmission times as shown on the next slide.



## Example 3.47 (continued)

$$\text{Propagation time} = \frac{12,000 \times 1000}{2.4 \times 10^8} = 50 \text{ ms}$$

$$\text{Transmission time} = \frac{5,000,000 \times 8}{10^6} = 40 \text{ s}$$

Note that in this case, because the message is very long and the bandwidth is not very high, the dominant factor is the transmission time, not the propagation time. The propagation time can be ignored.

- **Queuing Time :**
  - It is **the time needed for each intermediate** or end device to hold the message before it can be processed.
  - It is not a fixed factor, it changes with the load imposed in the network.
  - When there is **heavy traffic on the network, the queuing time increases.**
  - An intermediate device, **such as a router, queues the arrived messages and processes them one by one.** If there are many messages, each message will have to wait.

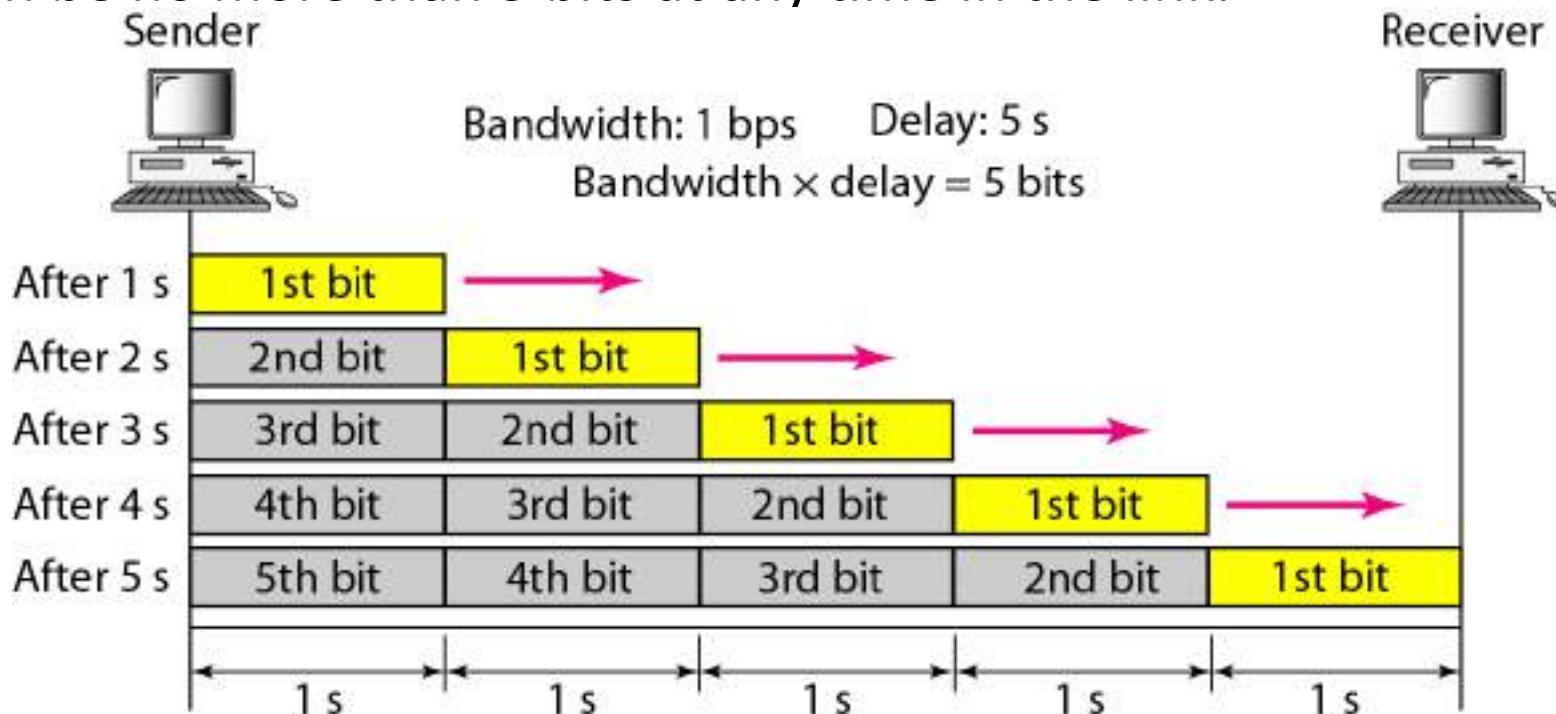
*(give the example of mikretek bandwidth server that ISMS has)*

# Processing Delay

- In a network based on packet switching, processing delay is the time it takes routers to process the packet header. Processing delay is a key component in network delay.
- During processing of a packet, routers may check for bit-level errors in the packet that occurred during transmission as well as determining where the packet's next destination is.
- Processing delays in high-speed routers are typically on the order of microseconds or less.
- After this nodal processing, the router directs the packet to the queue where further delay can happen.

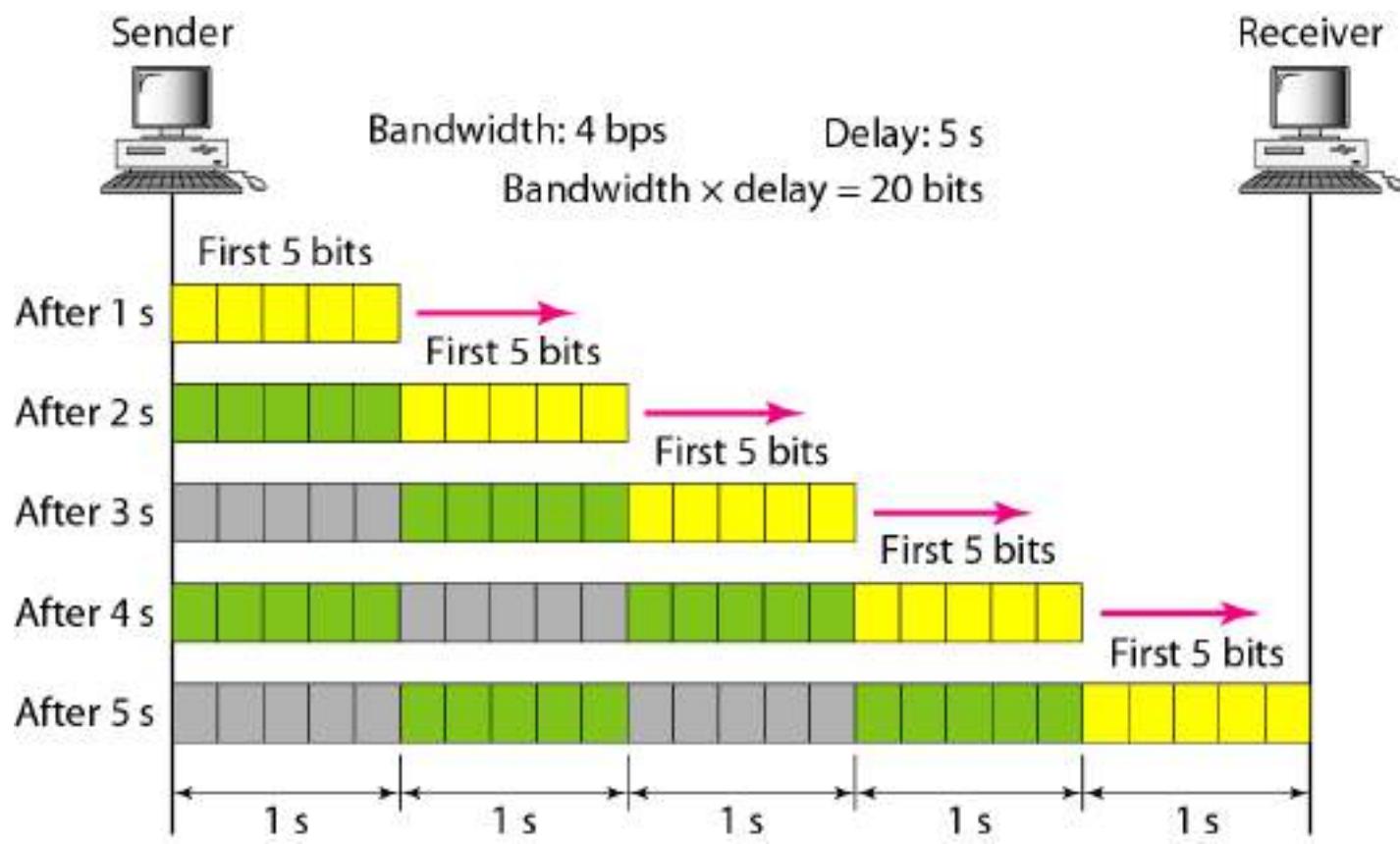
# Bandwidth-Delay Product

- We can say that this product  $1 * 5$  is the maximum number of bits that can fill in the link.
- There can be no more than 5 bits at any time in the link.



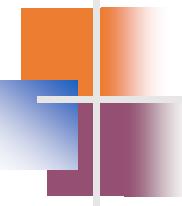
# Bandwidth-Delay Product

- There can be maximum  $5 * 5 = 25$  bits on the line.
- At each second, there are 5 bits on the line;



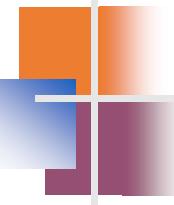
# Jitter

- Another performance issue that is related to delay is **jitter**.
- Jitter is a problem if different packets of data encounter different delays and the application using the data at the receiver site is time-sensitive (audio and video data, for example).
- If the delay for the first packet is 20 ms, for the second is 45 ms, and for the third is 40 ms, then the real-time application that uses the packets endures **jitter**.



## Example 3.42

The bandwidth of a subscriber line is 4 kHz for voice or data. The bandwidth of this line for data transmission can be up to 56,000 bps using a sophisticated modem to change the digital signal to analog.



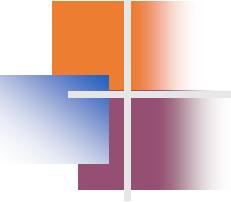
## Example 3.43

If the telephone company improves the quality of the line and increases the bandwidth to 8 kHz, we can send 112,000 bps by using the same technology as mentioned in Example 3.42.



## Example 3.48

We can think about the link between two points as a pipe. The cross section of the pipe represents the bandwidth, and the length of the pipe represents the delay. We can say the volume of the pipe defines the bandwidth-delay product, as shown in Figure 3.33.



### Note

---

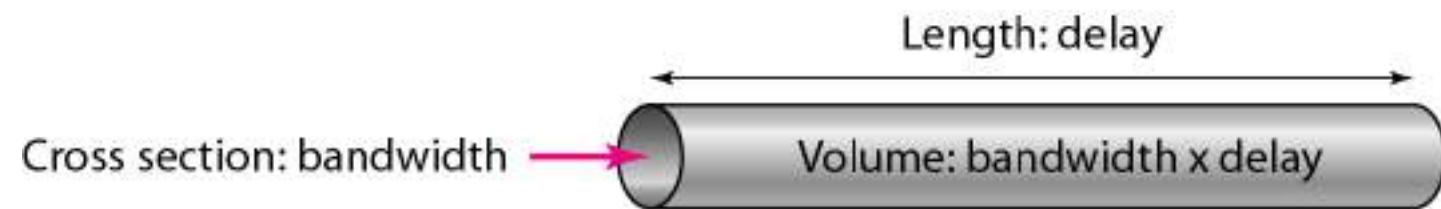
The bandwidth-delay product defines the number of bits that can fill the link.

---

---

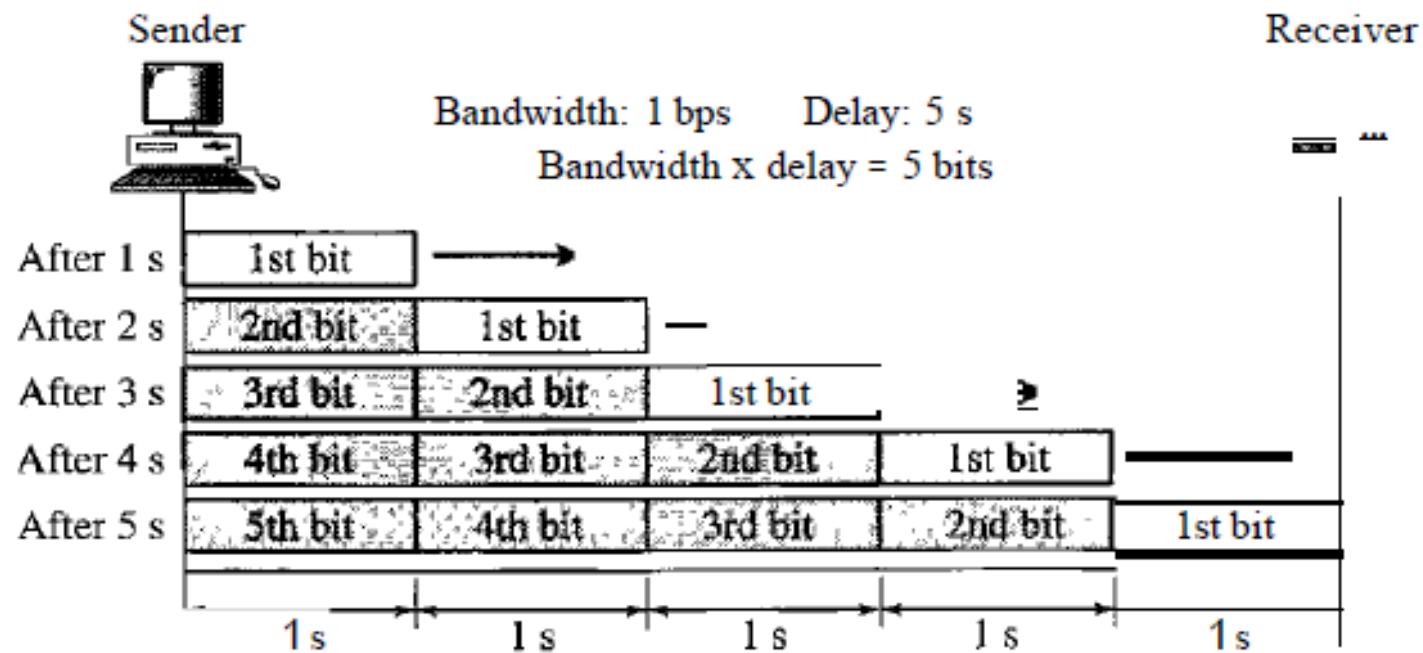
**Figure 3.33** Concept of bandwidth-delay product

---



# Bandwidth- Delay Product

- It is the number of bits that can fill the link.



A bandwidth of 1 bps

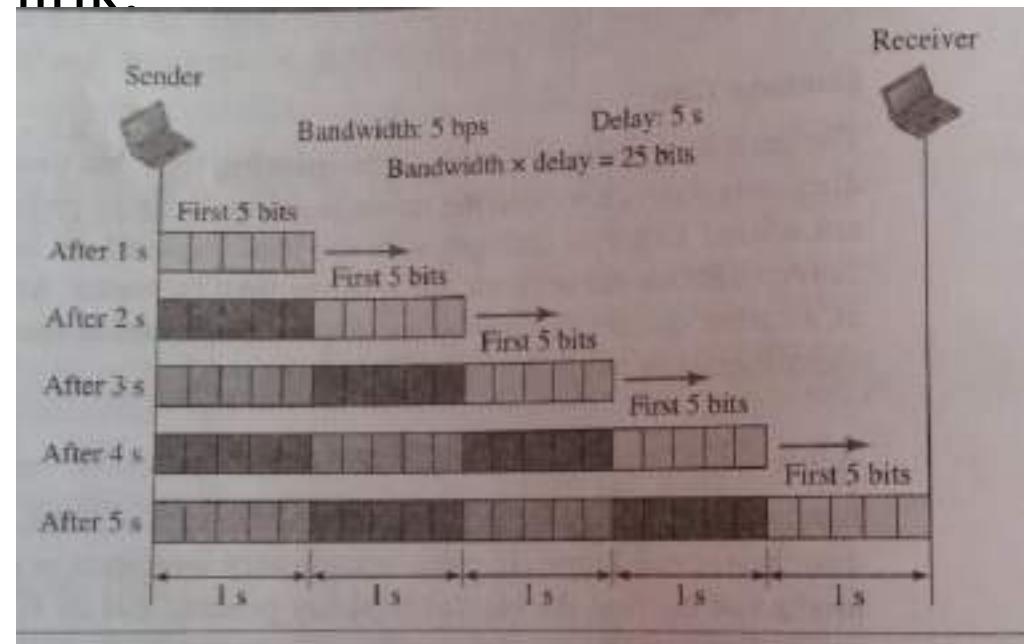
There can be no more than 5 bits at any time on the link

# Bandwidth- Delay Product

- It is the number of bits that can fill the link.

A bandwidth of 5 bps

There can be no more than  $5 * 5 = 25$  bits at a any time on the link  
Duration of each bits: 0.20s

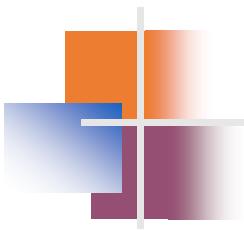


- The above two cases show that the product of bandwidth and delay is the number of bits that can fill the link
- To use the maximum capability of the link, we need to make a size 2 times the product.
- The amount  $2 * \text{bandwidth} * \text{delay}$  is the number of bits that can be in the transition at any time.

The bandwidth delay product defines the number of bits that can fill the link.

- When we increase the number of **signal levels**, we impose a burden on the receiver.
  - If the number of levels in a signal is just 2, the receiver can easily distinguish between a 0 and 1.
  - If the level of signal is 64, the **receiver must face complexity** to distinguish between 64 different levels.
  - In other words, increasing the levels of a signal reduces the reliability of the system.

Increasing the levels of a signal may reduce the reliability of the system



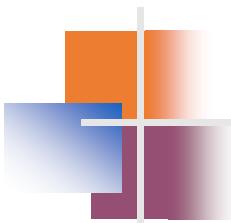
## Example 3.40

For practical purposes, when the SNR is very high, we can assume that  $\text{SNR} + 1$  is almost the same as  $\text{SNR}$ . In these cases, the theoretical channel capacity can be simplified to

$$C = B \times \frac{\text{SNR}_{\text{dB}}}{3}$$

For example, we can calculate the theoretical capacity of the previous example as

$$C = 2 \text{ MHz} \times \frac{36}{3} = 24 \text{ Mbps}$$



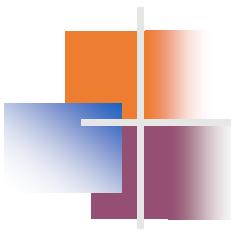
## Example 3.41

We have a channel with a 1-MHz bandwidth. The SNR for this channel is 63. What are the appropriate bit rate and signal level?

### Solution

First, we use the Shannon formula to find the upper limit.

$$C = B \log_2 (1 + \text{SNR}) = 10^6 \log_2 (1 + 63) = 10^6 \log_2 64 = 6 \text{ Mbps}$$



## Example 3.41 (continued)

The Shannon formula gives us 6 Mbps, the upper limit.

For better performance we choose something lower, 4 Mbps, for example.

Then we use the Nyquist formula to find the number of signal levels.

$$4 \text{ Mbps} = 2 \times 1 \text{ MHz} \times \log_2 L \quad \rightarrow \quad L = 4$$

- **Problem 1 :** Assume we need to download text documents at the rate of 100 pages per second. What is the required bit rate of the channel ?

### Solution

A page is an average of 24 lines with 80 characters in each line. If we assume that one character requires 8 bits, the bit rate is

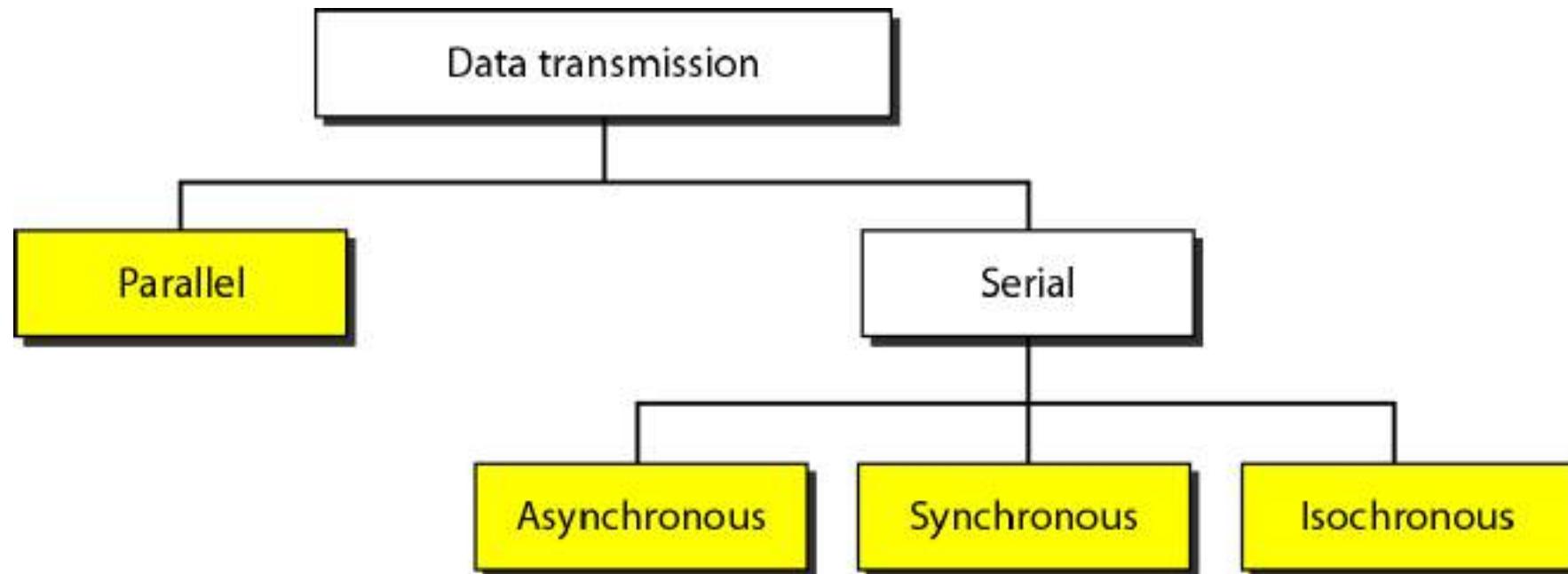
$$100 * 24 * 80 * 8 = 1,536,000 \text{ bps} = 1.536 \text{ Mbps}$$

# Transmission Mode

## *Data transmission and modes*

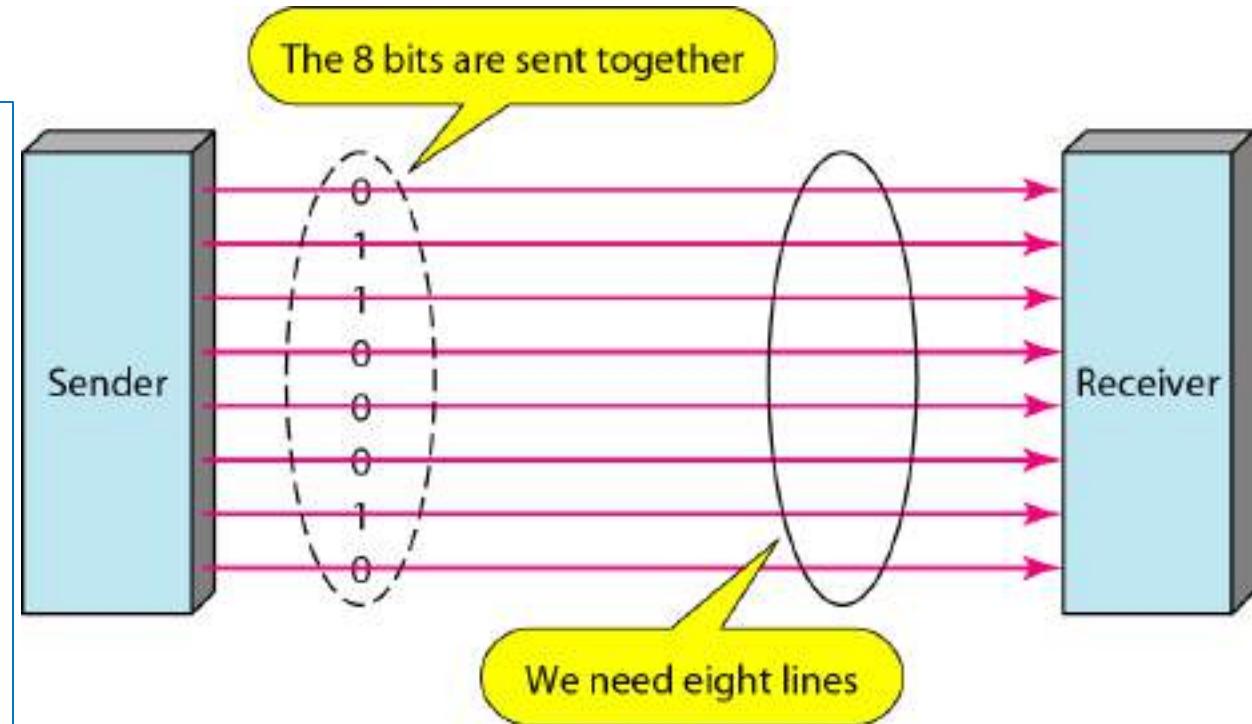
---

Its all about transmission of data from one device to another.  
How we send data in the single bit or in group ?



## *Parallel transmission*

- **Parallel Transmission** : data in the form of the bits are organized into groups of  $n$  bits each and send in  $n$  wires at a time.
- There is need for each bit to have its own wire, and all  $n$  bits of one group can be transmitted with each clock tick from one device to another.



**Advantage**  
**-speed.**

**Disadvantage are**

**Costly**

Parallel transmission requires n communication lines (wires in the example) just to transmit the data stream.

**Conversion device is need**

It is very hard to make the possibilities for long distance but for short distance it could be possible.

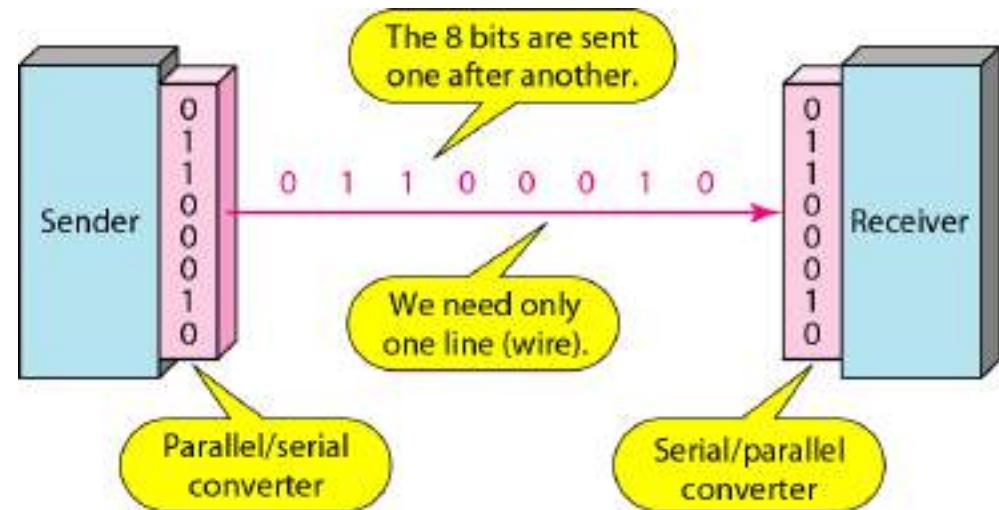
## *Serial transmission*

---

**Serial Transmission:** In this mode one bit follows the other that one channel is enough for the transmission.

The advantage of serial over communication within devices is parallel,

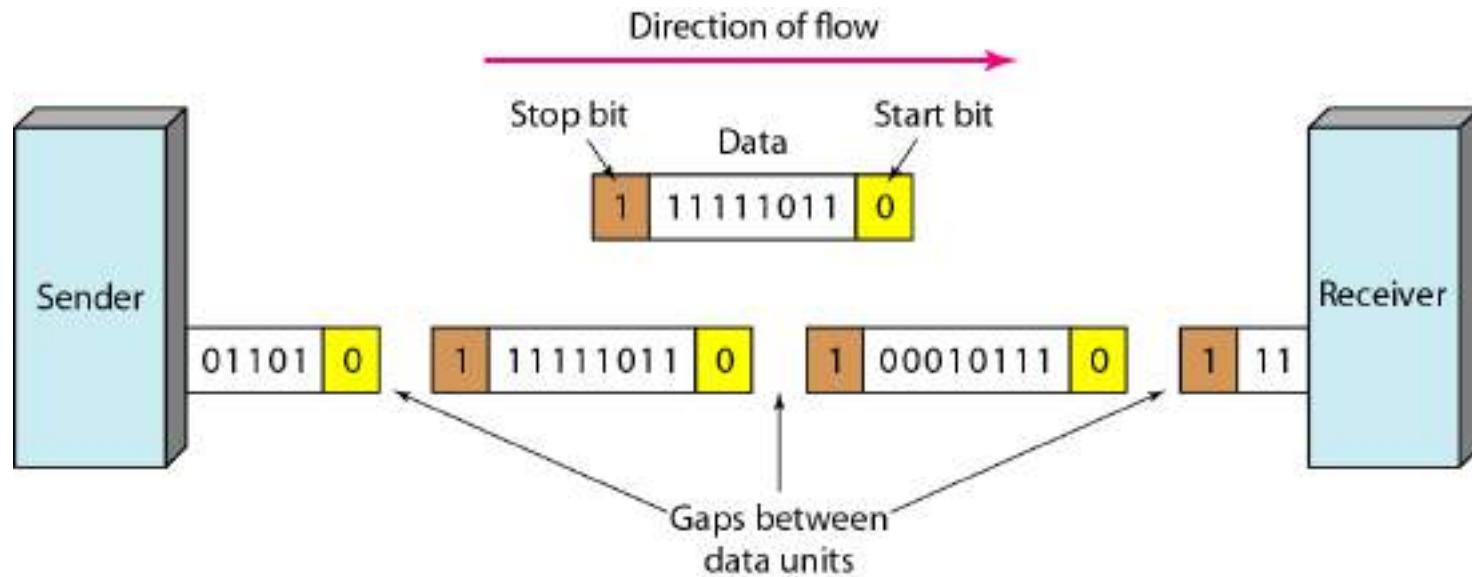
One communication channel, serial transmission reduces the cost of transmission over parallel by roughly a factor of  $n$ .



- Serial transmission occurs in one of the three ways
  - Asynchronous
  - Synchronous
  - Isochronous

## Asynchronous transmission

In this transmission, timing of a **signal is unimportant**  
**pattern for transforming of bit stream is important.**  
transmission can occur at any time  
Pattern are based on grouping the bit stream into bytes.



*It is “asynchronous at the byte level,” bits are still synchronized; their durations are the same.*

- The size of byte will increase due to the addition of **start** and **stop** bits.
- By this method at least **10 bits of which 8 bits** is information and 2 bits or more are signals to the receiver.
- In addition, the transmission of each byte may then be followed by a **gap** of varying duration.
- This mechanism is called asynchronous because, at the byte level, the sender and receiver do not have to be synchronized.
- Slower the transmission but is cheaper
- Connection of a **keyboard to a computer** is a natural application for asynchronous transmission.

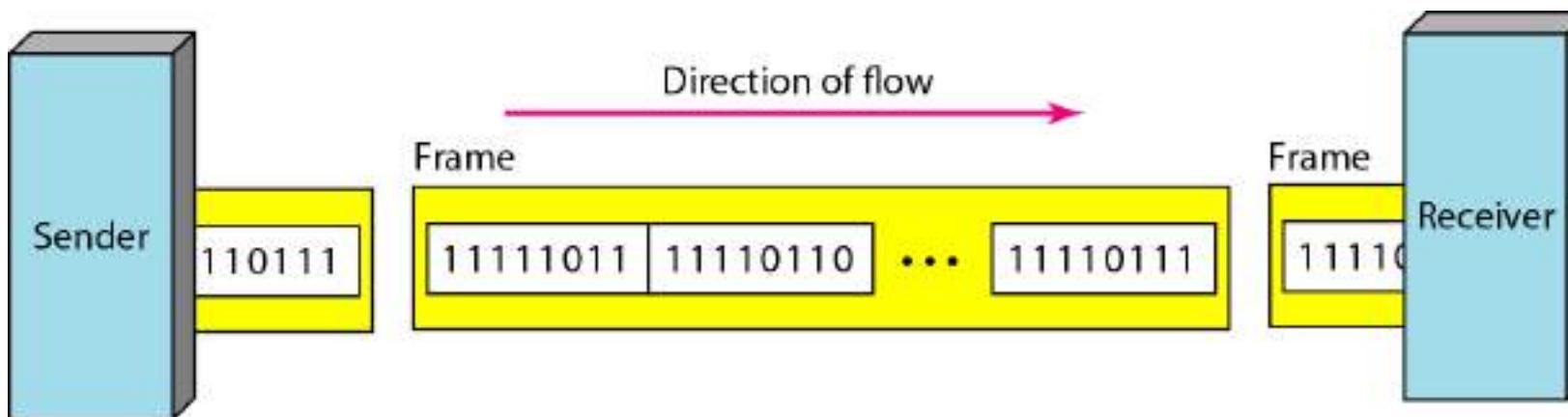
# Synchronous Transmission

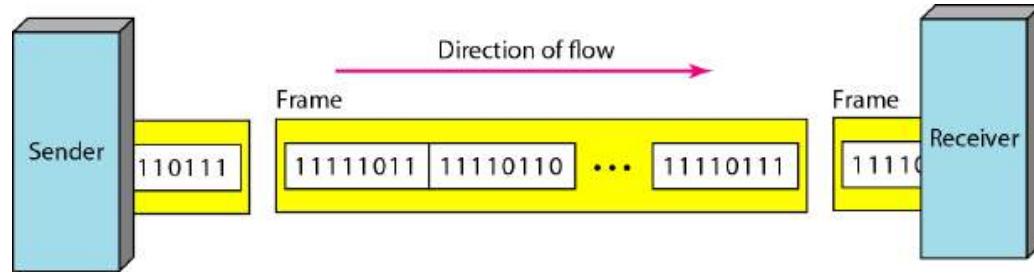
- In this mode, **the bit stream is combined** into longer frames which contain multiple bytes.
- **Bits are send one after another without start or stop bits or gaps.**
- It is the responsibility of the receiver to group the bits into bytes for decoding purposes.
- Since there is **no start and stop bits** or any gaps, **timing plays an important role** versus to the group the bits.
- Hence the accuracy depends completely on timing.
- If the sender wishes to send data in separate bursts, the gaps between bursts must be filled with a special sequences of 0s and 1s that means idle

# Synchronous transmission

We send bits one after another without start or stop bits or gaps.

It is the responsibility of the receiver to group the bits.



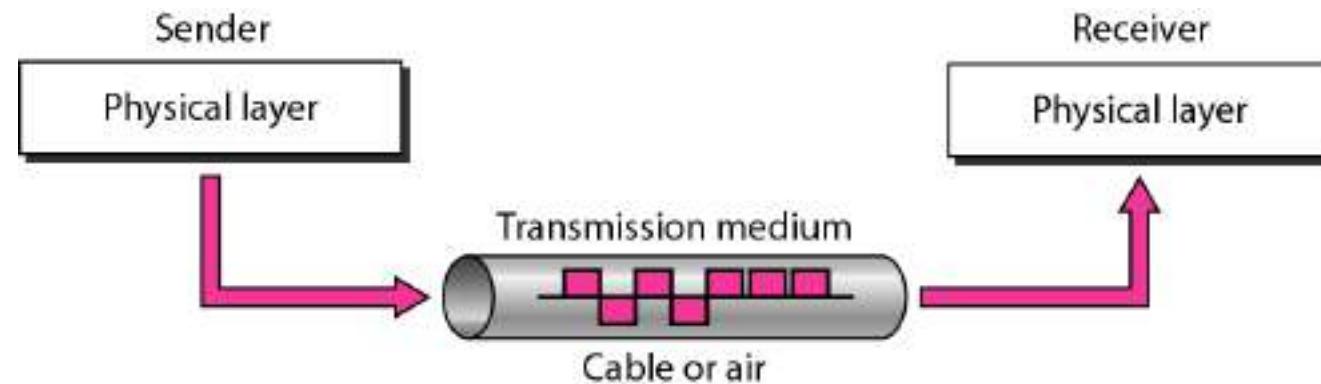


- Synchronous transmission is **faster** than asynchronous transmission as it doesn't need to create a gap and also no extra bits.
- It is more useful for high-speed applications such as the transmission of data from one computer to another.
- Although there is no gap between characters in synchronous serial transmission, there may be uneven gaps between frames.

# Isochronous Transmission

- In real-time audio and video, in which **uneven delays between frames are not acceptable**, synchronous transmission fails.
- Eg TV broadcasting where broadcast occurs at **30 images/sec** to which each TV set must be able to viewed at same rate.
- If **each images is sent by using one or more frames**, there should be no delays between frames.
- For this type of application, **synchronization between characters is not enough**; the entire stream of bits must be synchronized.
- **The isochronous transmission guarantees the fixed data rate arrival of data frames.**

# Transmission medium and physical layer

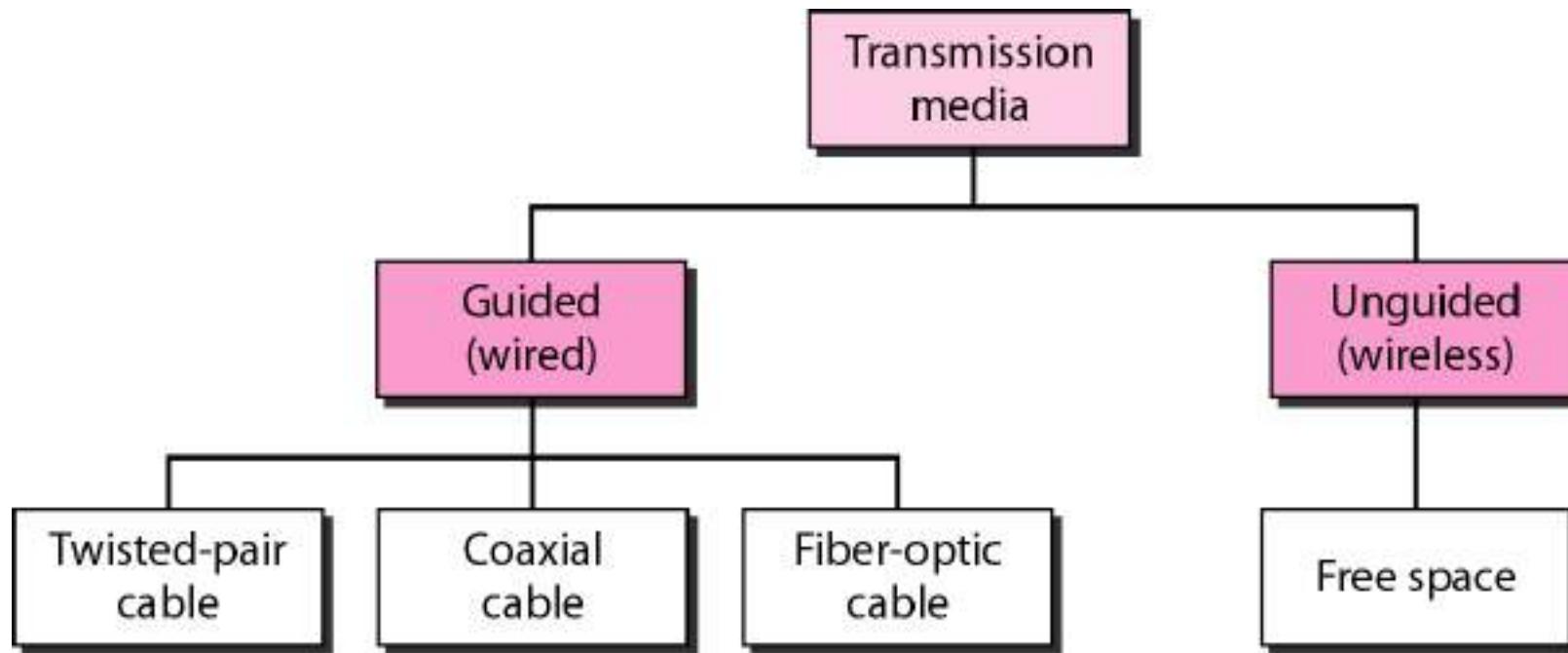


A transmission medium can be broadly defined as anything that can carry information from a source to a destination.

Signals are transmitted from one device to another in the form of electromagnetic energy.

In communication, transmission medium can be divided into following categories.

# Classes of Transmission media



# GUIDED MEDIA

*Guided media, which are those that provide a conduit from one device to another, include.*

*The transmission of signal is carried out through a physical medium i.e..*

*Cables in which the transmission capacity is determined either by **data rate** or by the **bandwidth** and it depends critically on the distance.*

Twisted-Pair Cable

Coaxial Cable

Fiber-Optic Cable

## Figure 7.3 *Twisted-pair cable*

---

A twisted pair consists of two conductors (normally copper), each with its own plastic insulation, twisted together.

One of the wires is used to carry signals to the receiver and the other is used only as a ground reference.

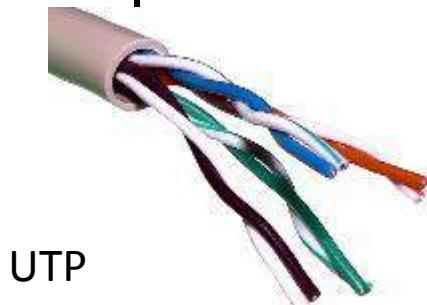
**The receiver uses the difference between the two.**



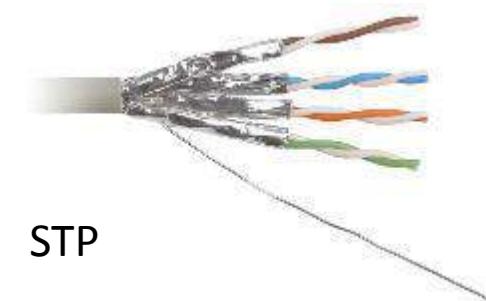
- Lets suppose in one twisted, one wire is closer to the noise source and other is farther; in the next twist, the reverse is true,
- Twisting makes it probable that both wires are equally affected by external influences.
- This means that difference between the two, receives no unwanted signals. The unwanted signals are mostly canceled out.
- It is least expensive and is a commonly used medium.
- Consists of two insulated copper wires arranged a regular spiral pattern.

# Unshielded Versus Shielded Twisted-Pair Cable

- The most common twisted-pair cable used in communication is referred to as
  - unshielded twisted-pair (**UTP**). It is cheaper than STP.
- **IBM** Produced **STP cable** which consists of a metal foil in between outer cover and insulator.
- It is expensive and heavy however it improves the quality of cable of preventing the penetration of noise.



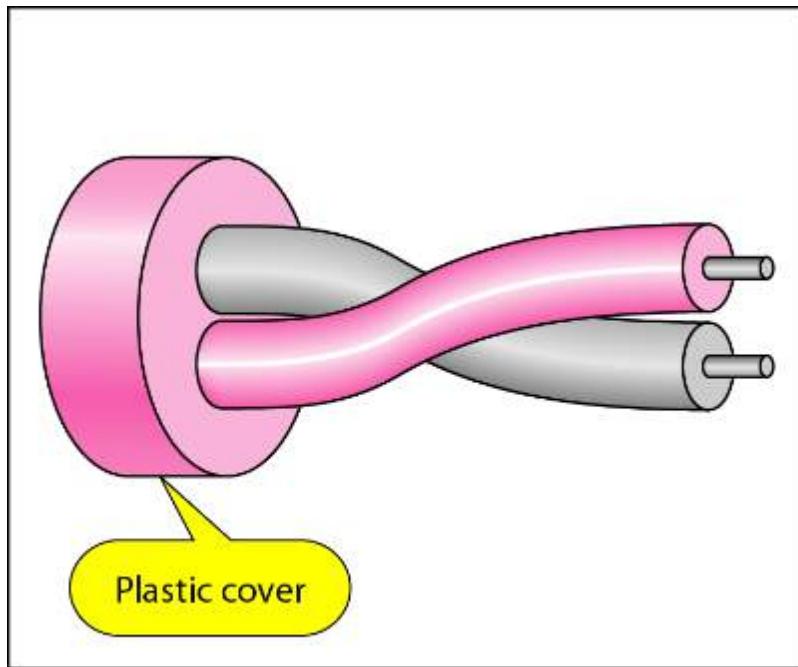
UTP



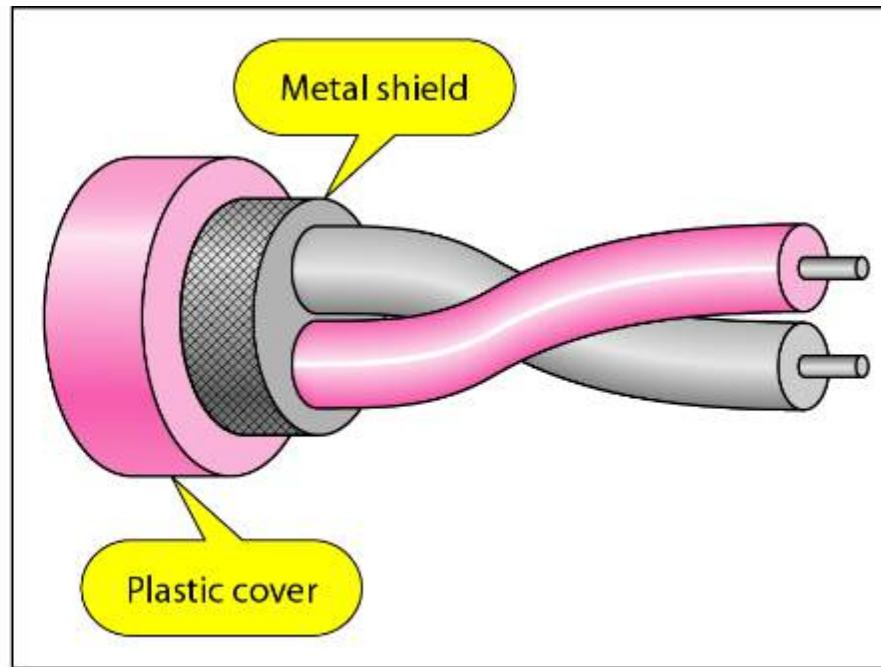
STP

*Figure 7.4 UTP and STP cables*

---



a. UTP

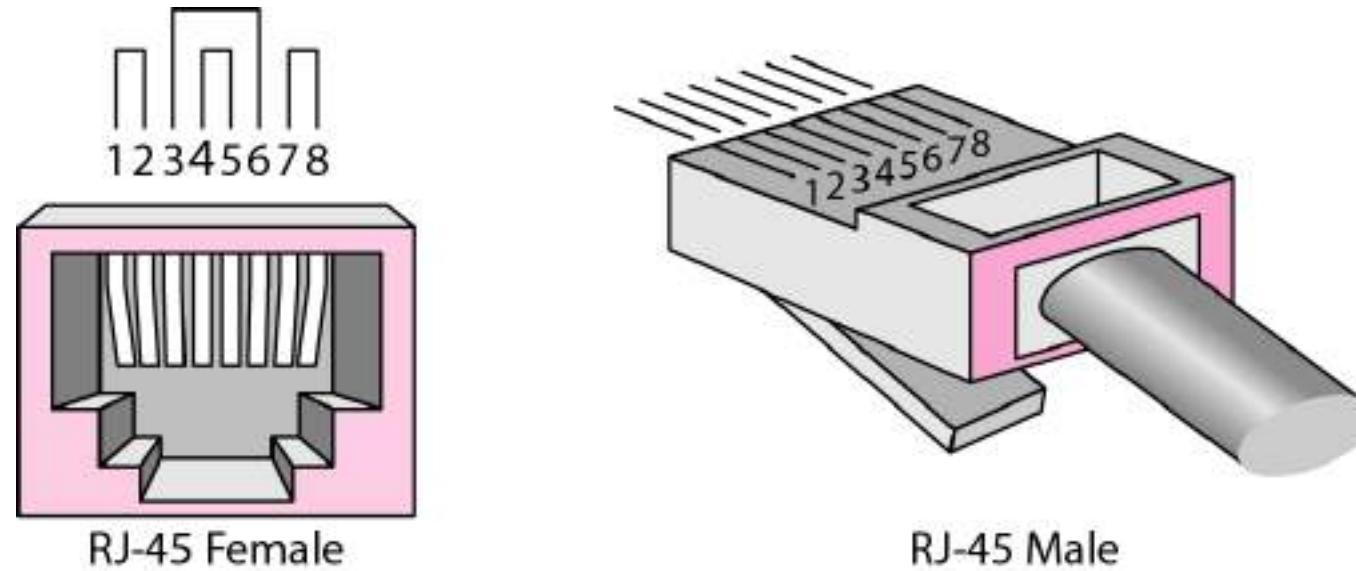


b. STP

**Table 7.1** *Categories of unshielded twisted-pair cables*

<i>Category</i>	<i>Specification</i>	<i>Data Rate (Mbps)</i>	<i>Use</i>
1	Unshielded twisted-pair used in telephone	< 0.1	Telephone
2	Unshielded twisted-pair originally used in T-lines	2	T-1 lines
3	Improved CAT 2 used in LANs	10	LANs
4	Improved CAT 3 used in Token Ring networks	20	LANs
5	Cable wire is normally 24 AWG with a jacket and outside sheath	100	LANs
5E	An extension to category 5 that includes extra features to minimize the crosstalk and electromagnetic interference	125	LANs
6	A new category with matched components coming from the same manufacturer. The cable must be tested at a 200-Mbps data rate.	200	LANs
7	Sometimes called SSTP (shielded screen twisted-pair). Each pair is individually wrapped in a helical metallic foil followed by a metallic foil shield in addition to the outside sheath. The shield decreases the effect of crosstalk and increases the data rate.	600	LANs

Figure 7.5 UTP connector



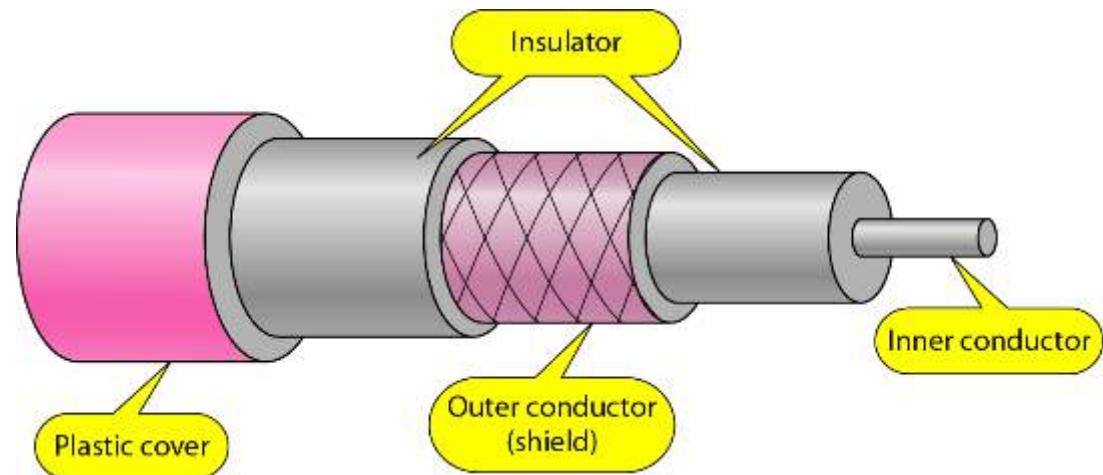
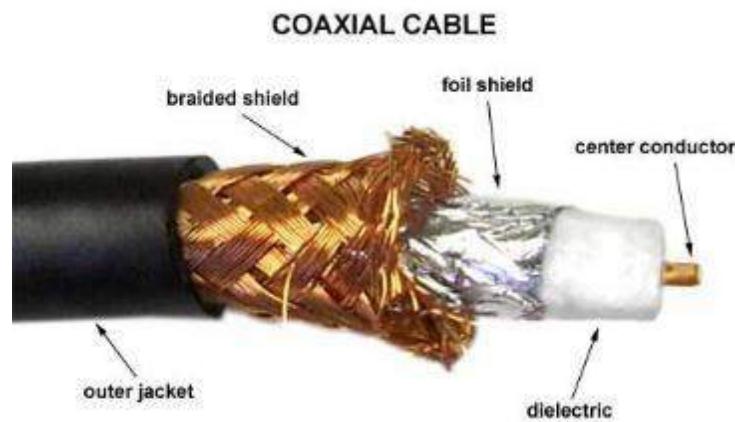
## RJ- Registered jack

Figure 7.7 *Coaxial cable*

---

## Coaxial cable:

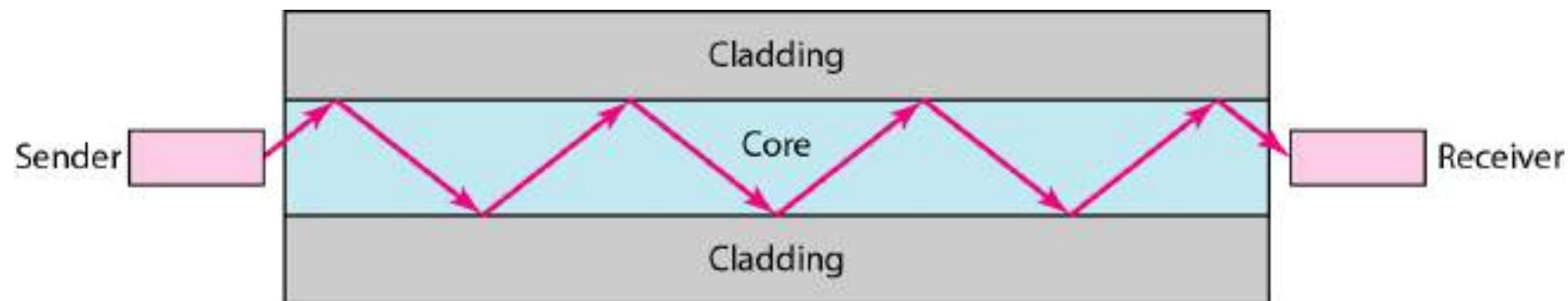
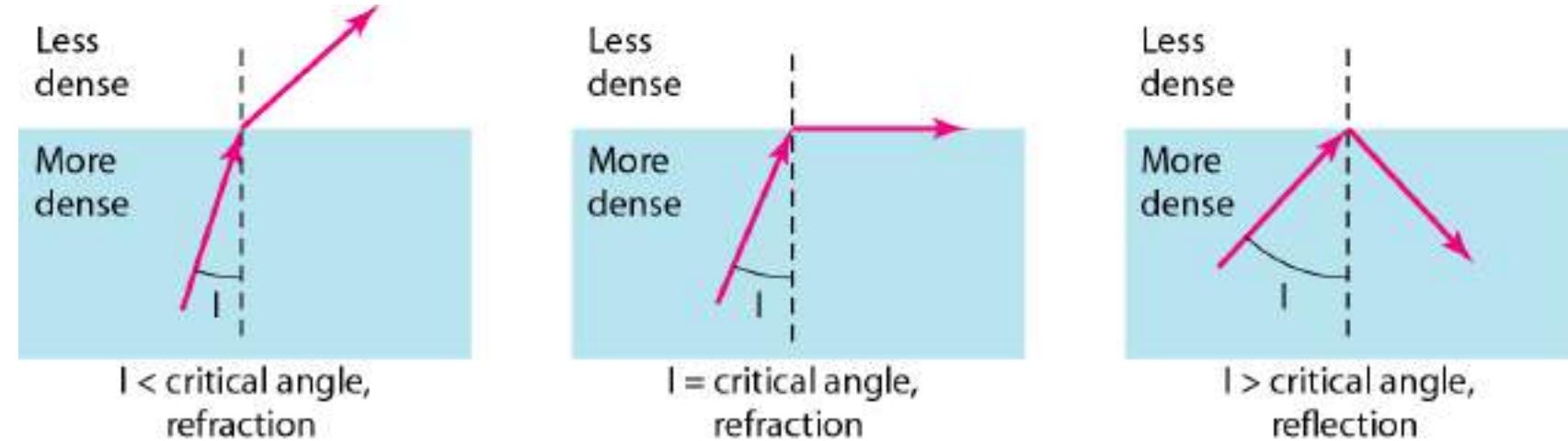
Coaxial cable carries signals of higher frequency ranges than those in twisted pair cable, in part because the two media are constructed quite differently.



- Link cables joined in TV
- **Better than twisted pair**
- Consists of two conductor (Innes and outer) which are separated by insulators.
- It is designed to carry higher signal frequency than twisted pair
- Can be used over a long distances and supports more stations on a shared line than twisted pair.
- Eg. TV cable long distance telephone transmission etc.

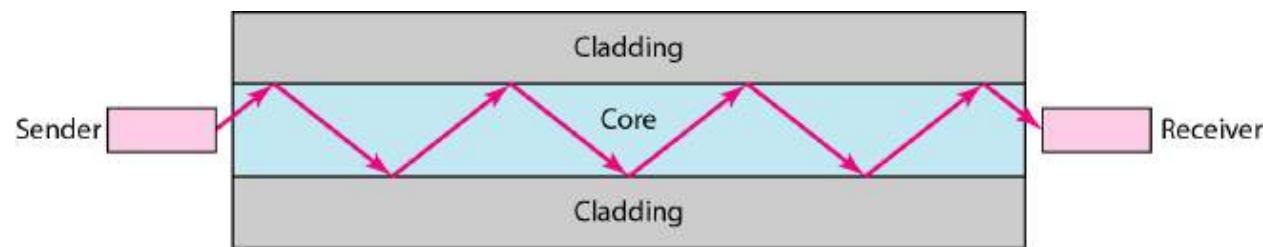
Figure 7.10 Fiber optics: *Bending of light ray*

---



# Fiber optic cable

- A fiber-optic cable is made of **glass or plastic** and transmits signals in the form of light.
- Optical fibers use reflection to **guide light through a channel**. A glass or plastic core is surrounded by a **cladding** of less dense glass or plastic.
- The difference in density of the two material must be such that a beam of light moving through the core is reflected **off the cladding** instead of being refracted into it.
- No data loss, faster transmission
- Works on the principle of total internal reflection.



# Advantage of Optical Fiber

- Higher Bandwidth
- Less signal attenuation
- Electromagnetic noise cannot affect fiber optic cables
- Resistance to corrosive materials
- Weight of cable is low compared to copper cable
- Greater immunity to tapping than copper cable *since copper cables create antenna effects that can easily be tapped.*

## Disadvantage

Installation and maintenance  
Unidirectional light propagation.

## 7-2 UNGUIDED MEDIA: WIRELESS

*Unguided media transport electromagnetic waves without using a physical conductor.*

*This type of communication is often referred to as wireless communication.*

Unguided signals can travel from source to destination in several ways:

Ground Propagation,

Sky Propagation

Line of Sight Propagation

# Ground Propagation

- In **ground propagation**, radio waves travel through the lowest portion of the atmosphere, hugging the earth.
- These low frequency signals emanate in all directions from the transmitting antenna and follow the curvature of the planet.
- Distance depends on the amount of power in the signal.
- The greater the power, the greater the distance.



# Sky propagation

- In the **sky propagation**, the **higher the frequency** radio waves radiate upward into the ionosphere
- (the layer of atmosphere where particles exist as ions) where they are reflected back to earth.



# Line of sight propagation

- In **line of sight propagation**, very high frequency signals are transmitted in straight lines directly from antenna to antenna.
- Antennas must be directional, facing each other, and either tall enough or close enough together not to be affected by the curvature of the earth.



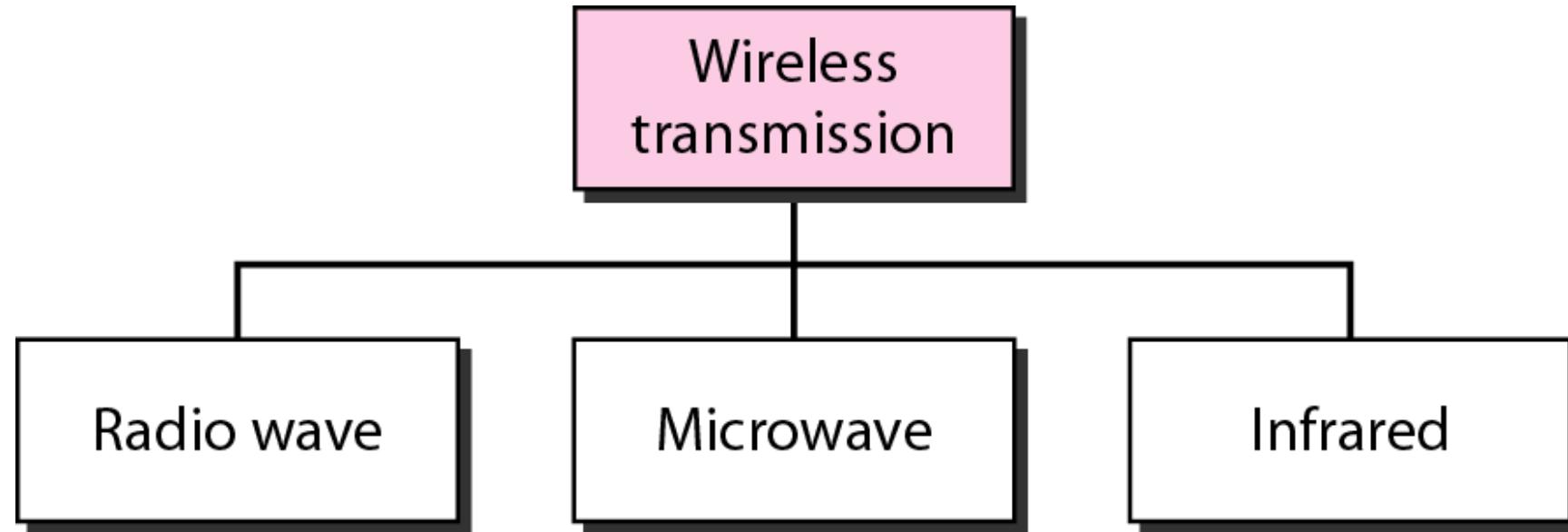
Line-of-sight propagation  
(above 30 MHz)

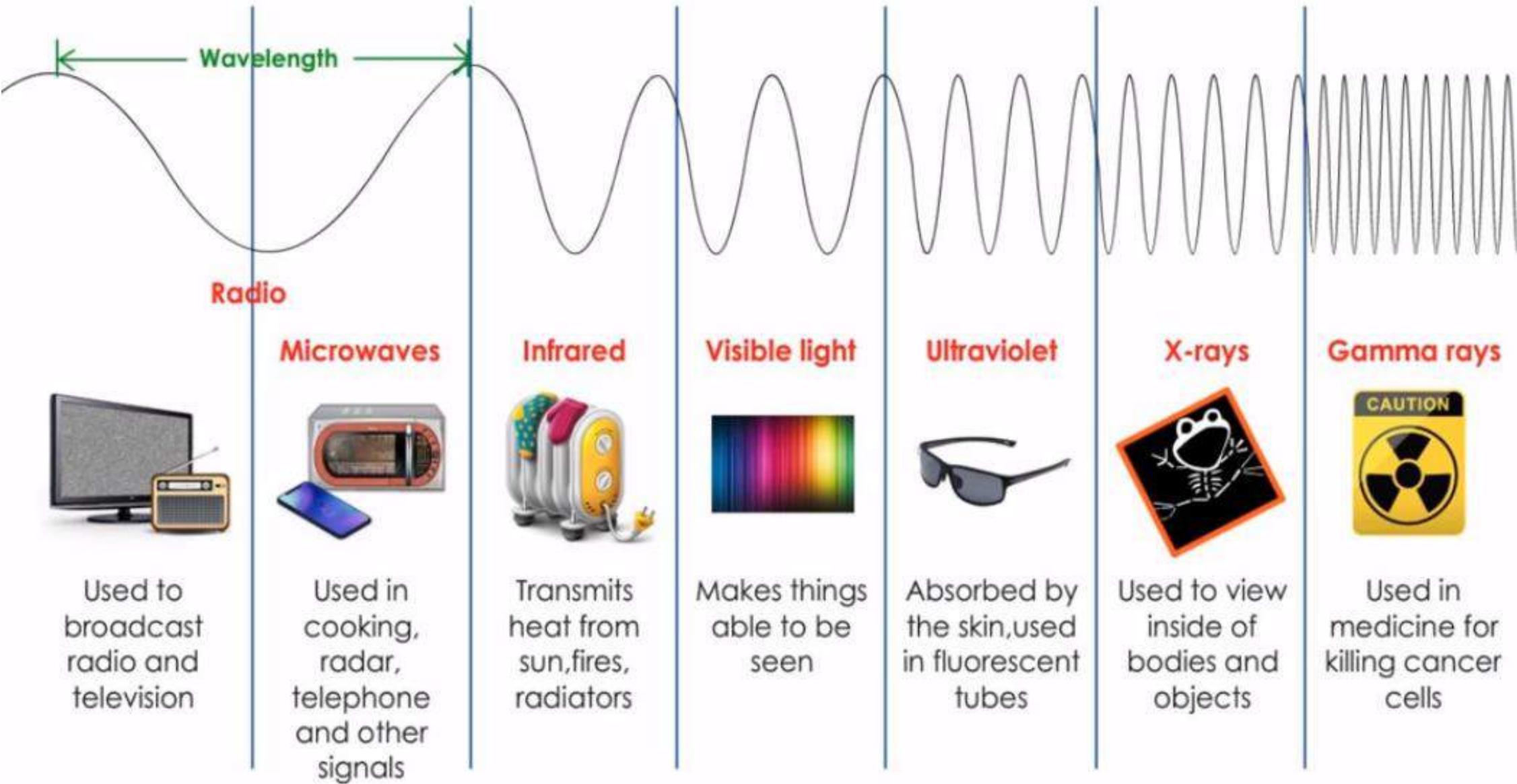
*Table 7.4 Bands*

<i>Band</i>	<i>Range</i>	<i>Propagation</i>	<i>Application</i>
VLF (very low frequency)	3-30 kHz	Ground	Long-range radio navigation
LF (low frequency)	30-300 kHz	Ground	Radio beacons and navigational locators
MF (middle frequency)	300 kHz-3 MHz	Sky	AM radio
HF (high frequency)	3-30 MHz	Sky	Citizens band (CB), ship/aircraft communication
VHF (very high frequency)	30-300 MHz	Sky and line-of-sight	VHF TV, FM radio
UHF (ultrahigh frequency)	300 MHz-3 GHz	Line-of-sight	UHF TV, cellular phones, paging, satellite
SHF (superhigh frequency)	3-30 GHz	Line-of-sight	Satellite communication
EHF (extremely high frequency)	30-300 GHz	Line-of-sight	Radar, satellite

Figure 7.19 *Wireless transmission waves*

---

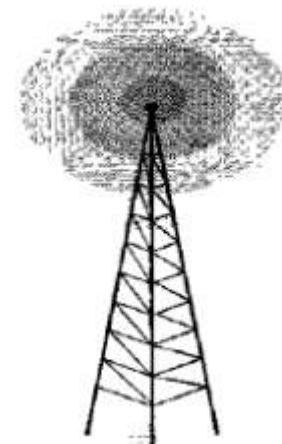




On the spectrum, frequency varies greatly.

# Radio Waves

- Electromagnetic waves ranging in frequency between  $3\text{kHz}$  and  $1\text{ gHz}$
- Most of Radiowave are **Omni-directional**.
- Sender antenna can send waves that can be received by any receiving antenna.
- Radio waves are used for multicast communication, such as radio, TV etc



Radio waves are used for multicast communications, such as radio and television, and paging systems. They can penetrate through walls.

Highly regulated. Use omni directional antennas



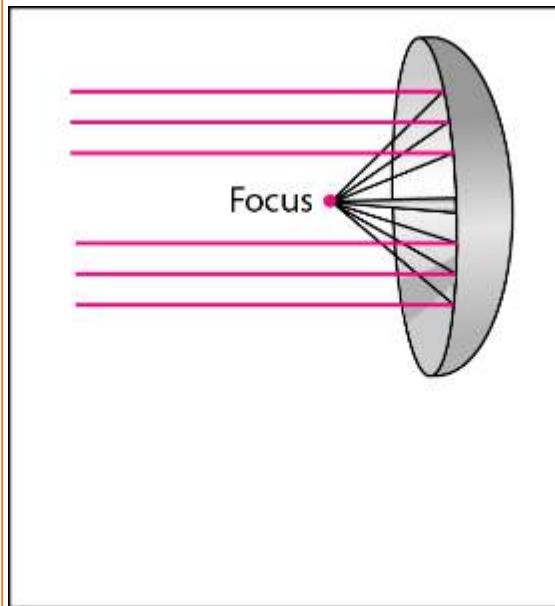
Figure 7.20 *Omnidirectional antenna*

**Figure 7.17** *Electromagnetic spectrum for wireless communication*

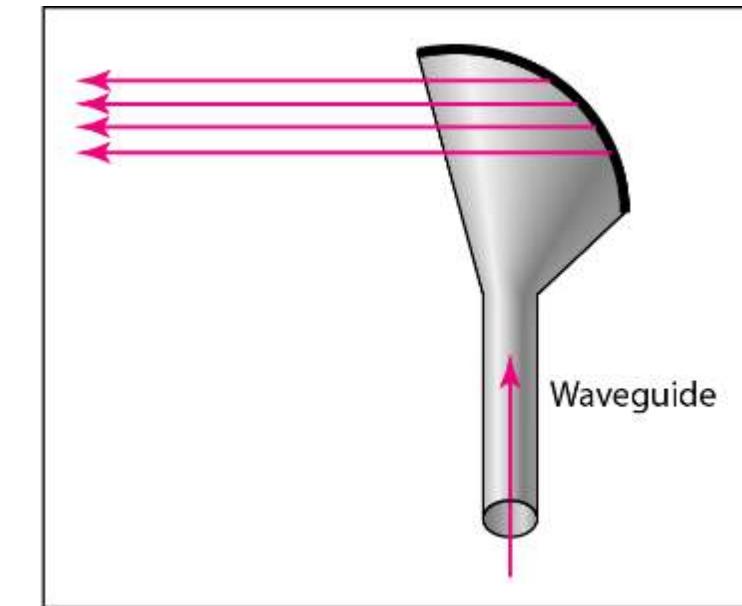


# Microwaves

- Microwaves are used for unicast communication such as **cellular telephones**, **satellite networks**, and **wireless LANs**.
- Higher frequency ranges cannot penetrate walls.
- Use directional antennas - point to point line of sight communications.



a. Dish antenna



b. Horn antenna

Figure 7.21 Unidirectional antennas

# MicroWaves

- Electromagnetic waves having frequencies between 1 and 300 GHz.
- They are **unidirectional waves**.
- Sending & receiving antenna need to be aligned (line of sight).
- **Very high frequency microwaves** cannot penetrate walls which is a disadvantage if the receiver is inside the building.
- Microwaves due to its unidirectional properties are very useful when unicasting (one to one) communication is needed.
- Used in cellular phones, satellites.

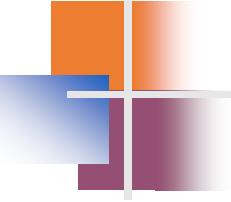


# Infrared

Infrared signals can be used for short-range communication in a closed area using line-of-sight propagation.

- The frequency range of infrared ranges from 300 GHz to 400 GHz.
- Used for short range communication.
- Interference occurs if operated on exposure to sun's ray which also contain infrared wave.
  - Eg Remote Control.

# Data Link Layer

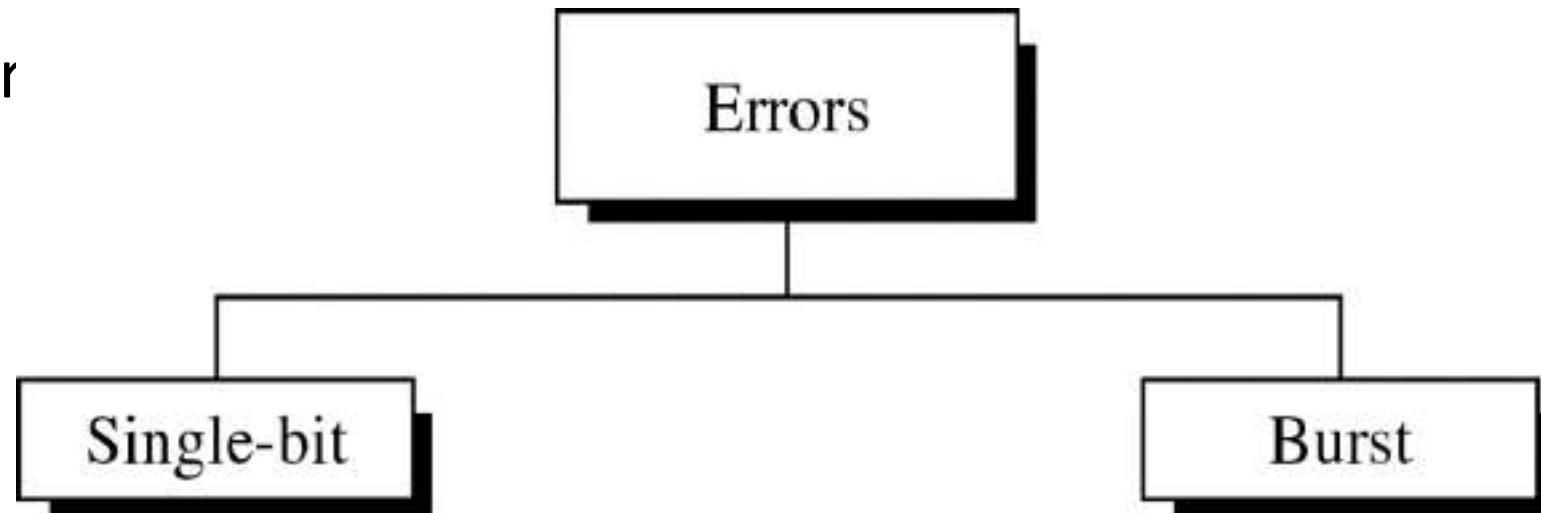


Data can be corrupted  
during transmission.

Some applications require that  
errors be detected and corrected.

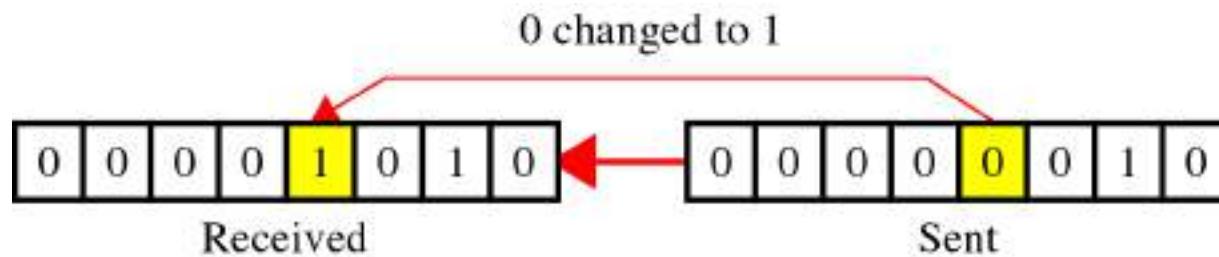
# Error Detection and Correction

- Network must be able to transfer data from one devices to another with acceptable accuracy.
- The chances of data being corrupt cannot be ignored.
- So there must be a mechanism for detecting such errors and correct them.
- Types of Error



## Single bit error:

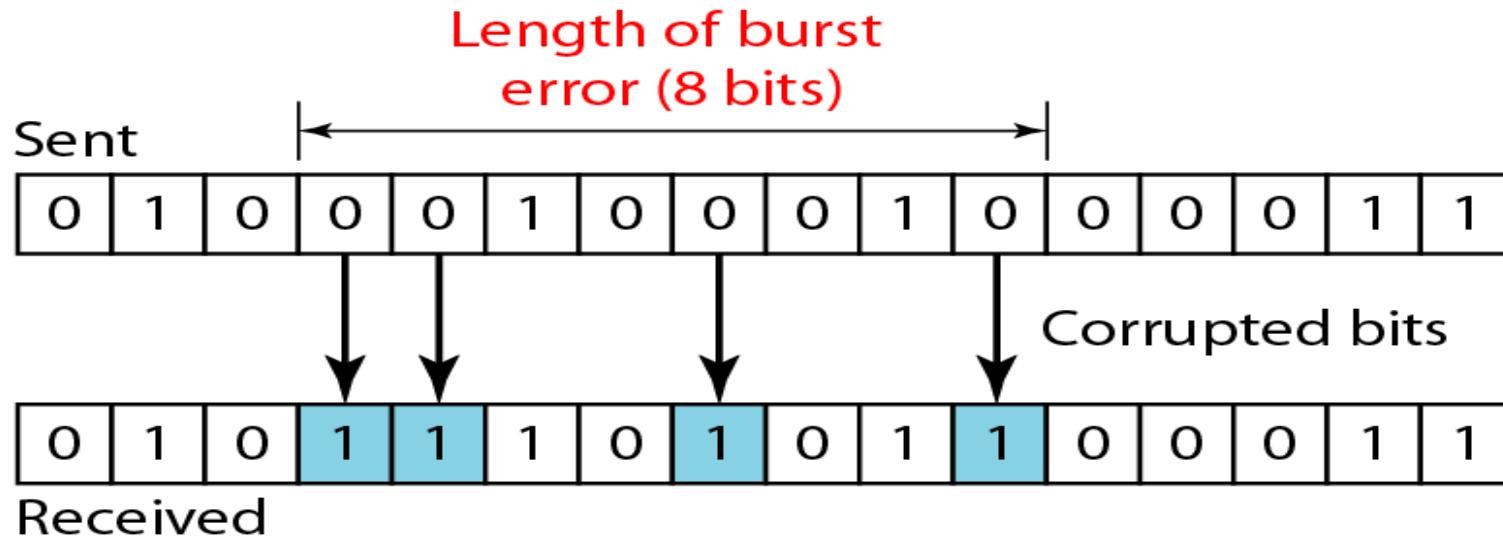
- Only 1 bit of a given data unit (byte/packet) is changed from 1 to 0 or 0 to 1.



This type of error is very rare to occur.

- For eg. If a data is sent at 1 mbps, then each one bit last only for 1/1,000,000 second (1 (u)micro second).
- So for single bit error to occur, the noise must have (duration of only 1 micro second which is very rare.)

# Burst Error



- ❖ A burst error means that 2 or more bits in the data unit have changed.
- ❖ The length of the burst is measured from the 1<sup>st</sup> corrupted bit to the last corrupted bit.
- ❖ Some bits in between may not have been corrupted.

# Redundancy

- The central concept in detecting or correcting errors is Redundancy.
- Instead of repeating the entire data stream, a shorter group of bits may be appended to the end of each unit.
- → This technique is called **Redundancy**
- These extra bits are discarded as soon as the accuracy of the transmission has been determined.

---

**To detect or correct errors, we need to send extra (redundant) bits with data.**

---

# Detection Vs Correction

- In detection
  - we only checked if any errors have occurred.
  - The answer is **yes** or **no** and we are not concerned on the number of errors.
- In correction,
  - we need to know the exact number of bits that are corrupted, and their location in the message.
  - The number of errors and the size of the message are important factors.
    - 8 bit data, correcting one (single) error bit = 8 possibility
    - 8 bit data, correcting 2 error bits = 28 possibilities
    - 1000 bit data, correcting 10 error bits = ..... Possibilities.

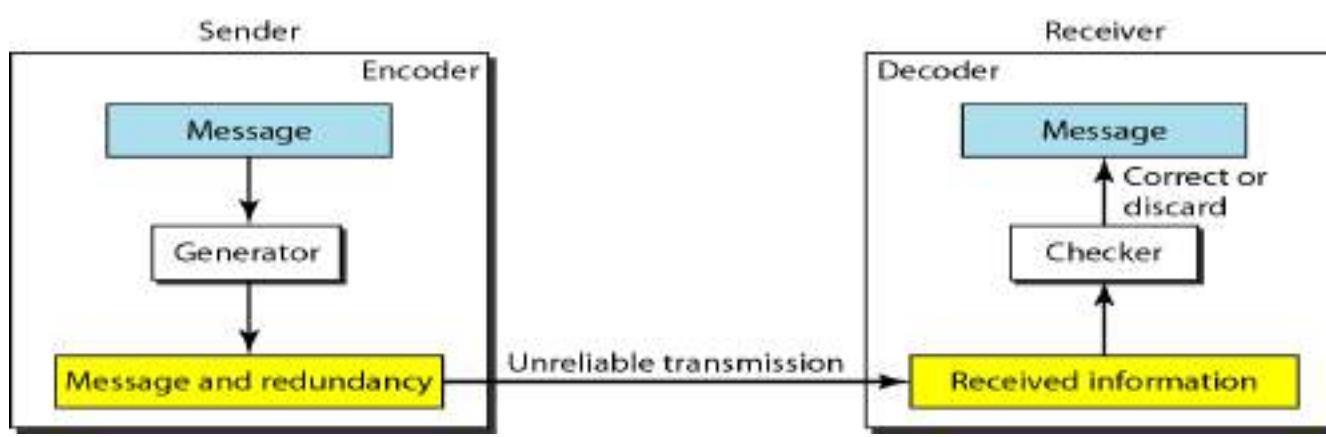
# Error Correction Method

- There are two main methods of error correction.
  - **Forward Error Correction**
    - Process in which the receiver tries to **guess** the message by Redundant bit.
  - **Retransmission:**
    - Receiver asks for **re-transmission** of message if it detects error. (usually not all errors are detected)

# Coding

The sender adds redundant bits

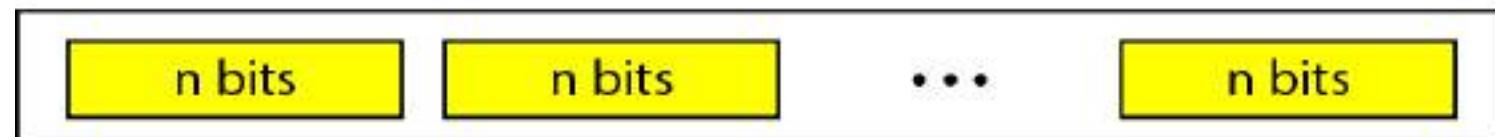
- The receiver checks the relationships between the two sets of bits to detect or correct the errors.
- Coding schemes are divided into two broad categories:
  - **Block coding**
  - **Convolution coding**



Dataword	Codeword
00	000
01	011
10	101
11	110

# Block Coding

- Message is divided into blocks, each of **k** bits, called **datawords**.
- We add **r** redundant bits to each block to make the length  **$n = k + r$** .
- The resulting n-bit blocks are called **codewords**.



*Datawords and codewords in block coding*

# Block Coding

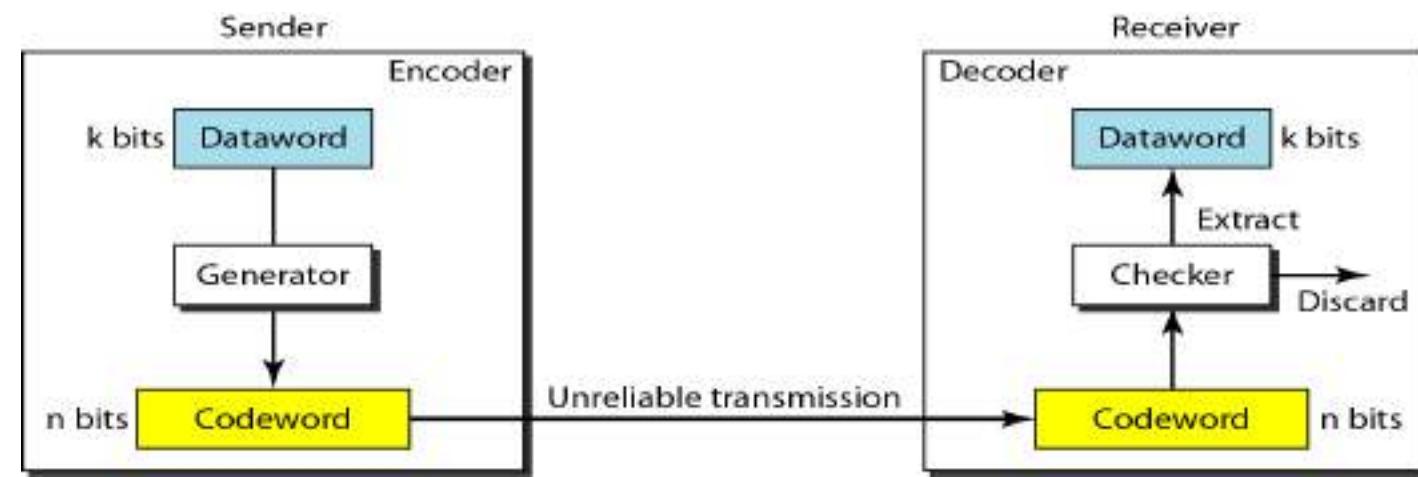
- With  $k$  bits, we can create a combination of  $2^k$  datawords.
- Since  $n$  bits, we can create a combination of  $2^n$  codewords.
- Since  $n > k$ . The possibility codeword is greater than possible data words.
- This means we have  $2^n - 2^k$  codewords extra.
  
- Eg if  $k=4$  and  $n=5$ ,
- we have  $2^4 = 16$  dataword and  $2^5 = 32$  codewords.
- Hence 16 out of 32 codewords are used for message transfer and rest are unused.

- This means that we have  $2^n - 2^k$  codewords that are not used.
  - We call these codewords invalid or illegal.
- 
- If the receiver receives an invalid codeword, this indicates that the data was corrupted during transmission.

# Error Detection in Block coding

- Two conditions satisfy the error detection.
    - The receiver has a list of valid codewords.
    - The original codewords has changed to an invalid one.
- Eg. Let  $k = 2$  and  $n = 3$

Dataword	Codeword
00	000
01	011
10	101
11	110



- Let the sender encodes dataword 01 as 011 and send it receiver.
- Following cases may arise.
  - Receiver receives 011, it is valid codeword. Dataword 01 is extracted by receiver. **No error.**
  - Receiver receives **111**. Code word is corrupted. This is not a valid codeword and it is discarded.
  - Receiver receives **000**. Codeword is corrupted but this is valid codeword. Receiver incorrectly extracts dataword 00. **Here two corrupted bits have made the error undetectable.**

An error-detecting code can detect only the types of errors for which it is designed; other types of errors may remain undetected.

Dataword	Codeword
00	000
01	011
10	101
11	110

# Hamming Distance

- It is one of the central concepts in coding for error control.
- The Hamming distance between two words (of the same size) is the number of differences between the corresponding bits.
- Why do you think Hamming distance is important for error detection?
- The reason is that the Hamming distance between the received codeword and the sent codeword is the number of bits that are corrupted during transmission.

*Let us find the Hamming distance between two pairs of words.*

*1. The Hamming distance  $d(000, 011)$  is 2 because*

$000 \oplus 011$  is 011 (two 1s)

*2. The Hamming distance  $d(10101, 11110)$  is 3 because*

$10101 \oplus 11110$  is 01011 (three 1s)

# Hamming Distance

*Let us find the Hamming distance between two pairs of words.*

*1. The Hamming distance  $d(000, 011)$  is 2 because*

000  $\oplus$  011 is 011 (two 1s)

*2. The Hamming distance  $d(10101, 11110)$  is 3 because*

10101  $\oplus$  11110 is 01011 (three 1s)

## BLOCK CODING - Minimum Hamming Distance

The minimum Hamming distance is the smallest Hamming distance all possible pairs in a set of words.

$d_{min}$  used to define the minimum Hamming distance in a coding scheme.

### Solution

We first find all Hamming distances.

$$d(000, 011) = 2$$

$$d(000, 101) = 2$$

$$d(000, 110) = 2$$

$$d(011, 101) = 2$$

$$d(011, 110) = 2$$

$$d(101, 110) = 2$$

The  $d_{min}$  in this case is 2.

Table 10.1 A code for  
error detection  
(Example 10.2)

Datawords	Codewords
00	000
01	011
10	101
11	110

# BLOCK CODING - Minimum Hamming Distance

Find the minimum Hamming distance of the coding scheme in Table 10.2.

## *Solution*

*We first find all the Hamming distances.*

$$\begin{array}{lll} d(00000, 01011) = 3 & d(00000, 10101) = 3 & d(00000, 11110) = 4 \\ d(01011, 10101) = 4 & d(01011, 11110) = 3 & d(10101, 11110) = 3 \end{array}$$

*The  $d_{min}$  in this case is 3.*

**Table 10.2 A code  
for error correction  
(Example 10.3)**

<i>Dataword</i>	<i>Codeword</i>
00	00000
01	01011
10	10101
11	11110

- **Three Parameters:**
  - For any coding scheme we need three parameters.
    - Code word size  $n$
    - Dataword size  $k$
    - Minimum Hamming distance  $d_{\min}$
  - A **coding scheme  $C$**  is written is  $C(n, k)$  with a separate expression for  $d_{\min}$
  - For example  $C(3,2)$  with  $d_{\min}=2$  and  $C(5,2)$  with  $d_{\min}=3$ .

## Relationship in between Hamming distance and errors occurring

- Hamming distance between the sent and received codewords is the **number of bits affected by the error.**
- For example,
  - send codeword 00000
  - received codeword 01101, 3 bits are in error and the Hamming distance is
  - $d(00000,01101) = 3$

# Minimum Hamming Distance for Error Detection

- If  $s$  errors occur during transmission,
  - the Hamming distance between the sent codeword and received codeword is  $s$ .
- If it is necessary to detect upto  $s$  errors,
  - the minimum hamming distance between the valid codes must be  $s+1$ , so that the received codeword does not match a valid codeword.

**To guarantee the detection of up to  $s$  errors in all cases, the minimum Hamming distance in a block code must be  $d_{\min}=s + 1$**

- The minimum Hamming distance for our first code scheme from the table is

$$d_{\min} = s+1$$

or  $2=s+1$

$S=1$

So, this code guarantees detection of

Dataword	Codeword
00	000
01	011
10	101
11	110

# BLOCK CODING - Minimum Distance for Error Correction

---

To guarantee **correction** of up to  $t$  errors in all cases, the minimum Hamming distance in a block code must be  $d_{\min} = 2t + 1$ .

---

# BLOCK CODING - Minimum Distance for Error Correction

## Example 10.9

A code scheme has a Hamming distance  $d_{\min} = 4$ . What is the error detection and correction capability of this scheme?

### Solution

- This code guarantees the detection of up to **three** errors ( $s = 3$ ), but it can correct up to **one** error.
- Error correction codes need to have an **odd minimum distance** (3, 5, 7, ...).

# Linear Block Codes

- This is the widespread used coding scheme. A linear block code is a code in which the exclusive OR of two valid codeword creates another valid codeword.
- Eg.

$$000 \oplus 011 = 011$$

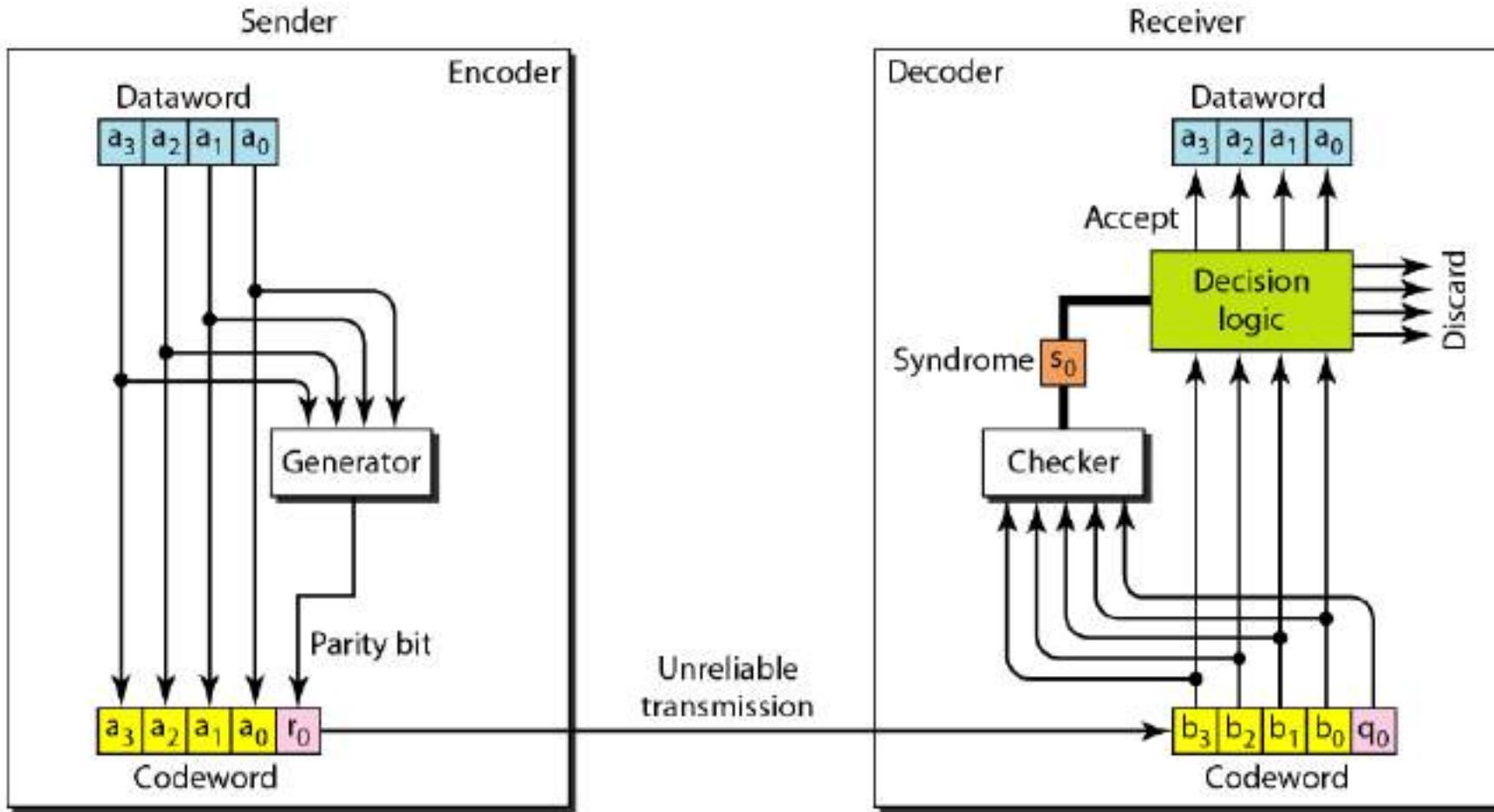
$$011 \oplus 101 = 110$$

$$101 \oplus 110 = 011$$

# Simple parity-Check Code

- In Simple parity-check code, a k-bit dataword is changed to an n-bit codeword where  $n=k+1$ .
- The extra bit, called the parity bit, is selected to make the total number of 1s in the codeword even.
- Although some implementations specify an odd number of 1s, we discuss the even case.
- This code is a single-bit error-detecting code· it cannot correct any error.

A simple parity-check code is a  
single-bit error-detecting  
code in which  
 $n = k + 1$  with  $d_{\min} = 2$ .



- At Generator,  $r_0 = a_3 + a_2 + a_1 + a_0 \text{ (modulo 2)}$ 
  - Where if number of 1s is even, the result is zero
  - if number of 1s is odd, the result is one

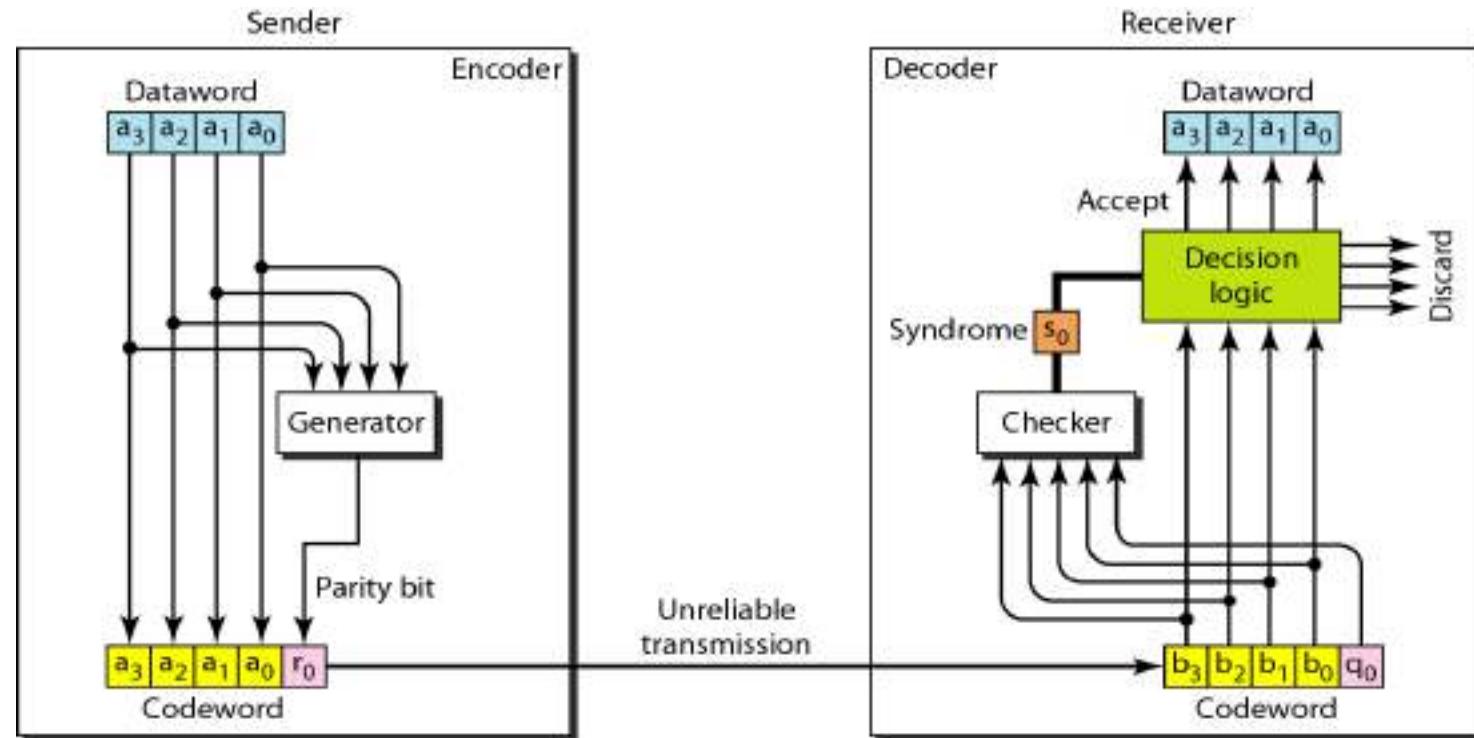
## Minimum Distance for Linear Block Codes

*Simple parity-check code C(5, 4)*

Datawords	Codewords	Datawords	Codewords
0000	00000	1000	10001
0001	00011	1001	10010
0010	00101	1010	10100
0011	00110	1011	10111
0100	01001	1100	11000
0101	01010	1101	11011
0110	01100	1110	11101
0111	01111	1111	11110

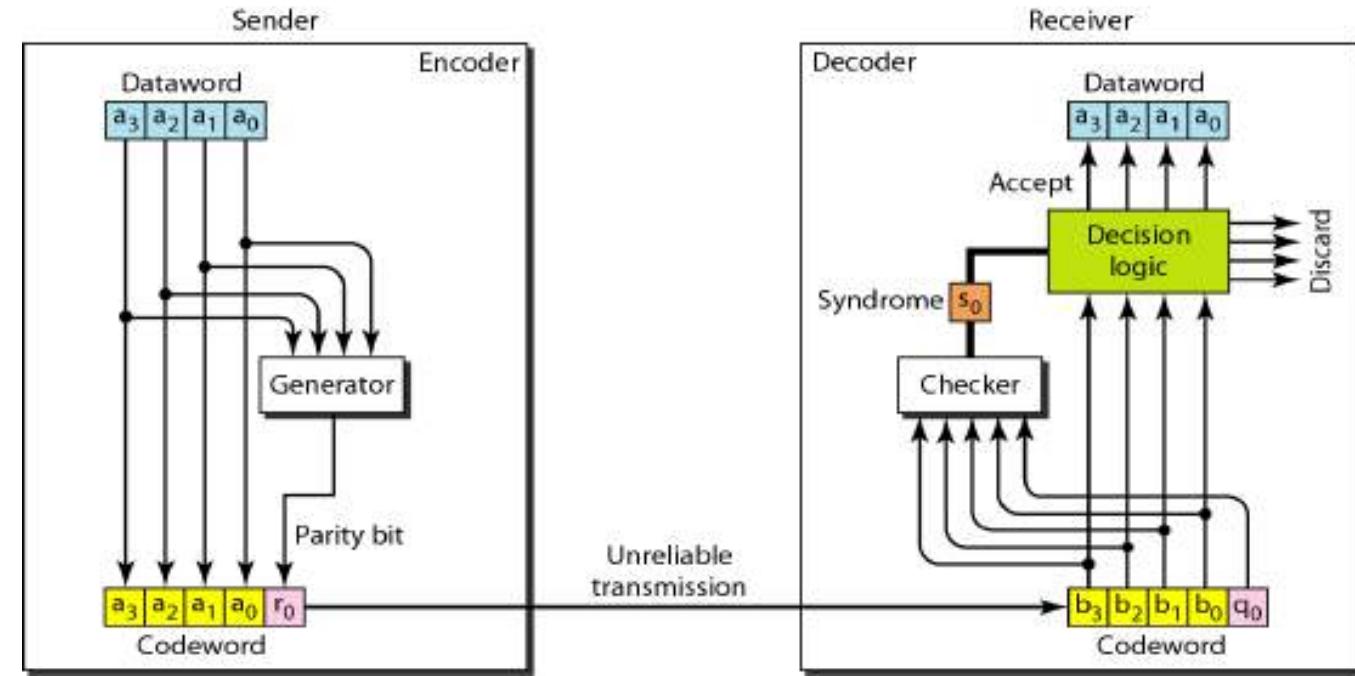
At Generator,  $r_o = a_3 + a_2 + a_1 + a_0 \text{ (modulo 2)}$

Where if number of 1s is even, the result is zero  
if number of 1s is odd, the result is one



- The checker at the receiver does the same thing as the generator in the sender with one exception: The addition is done over all 5 bits.
- The result, which is called the **syndrome**, is just 1 bit.
- The syndrome is **0** when the number of **1s** in the received codeword is even; otherwise, it is **1**.

- The syndrome is passed to the decision logic analyzer.
- If the syndrome is 0, **there is no error in the received codeword**; codeword is accepted as the dataword;
- if the syndrome is 1, the data portion of the received codeword is discarded.
- **The dataword is not created.**



# *Simple parity-check (Cont..)*

Assume the sender sends the dataword **1011**.

The codeword created from this dataword is **10111**, which is sent to the receiver. We examine five cases:

1. No error occurs; the received codeword is **10111**. The syndrome is 0. The dataword **1011** is created.
2. One single-bit error changes  $a_1$ . The received codeword is **10011**. The syndrome is 1. No dataword is created.
3. One single-bit error changes  $r_0$ . The received codeword is **10110**. The syndrome is 1. No dataword is created.

# *Simple parity-check (Cont...)*

4. An error changes  $r_0$  and a second error changes  $a_3$ .

The received codeword is 00110. The syndrome is 0.

The dataword 0011 is created at the receiver.

Note that here the dataword is **wrongly created** due to the syndrome value.

**The simply parity-check decoder cannot detect an even numbers of errors.**

5. Three bits— $a_3$ ,  $a_2$ , and  $a_1$ —are changed by errors.

The received codeword is 01011. The syndrome is 1.

The dataword is not created. This shows that the simple parity check, guaranteed to detect one single error, can also find any odd number of errors.

This shows that the simple parity check, guaranteed to detect **one** single error, can also find any **odd number** of errors.

# Hamming codes

- Hamming codes were originally designed with  $d_{\min}=3$ , which means that can detect up to **two errors or correct one** single error.

A codeword consists of **n** bits of which **k** are data bits and **r** are check bits.

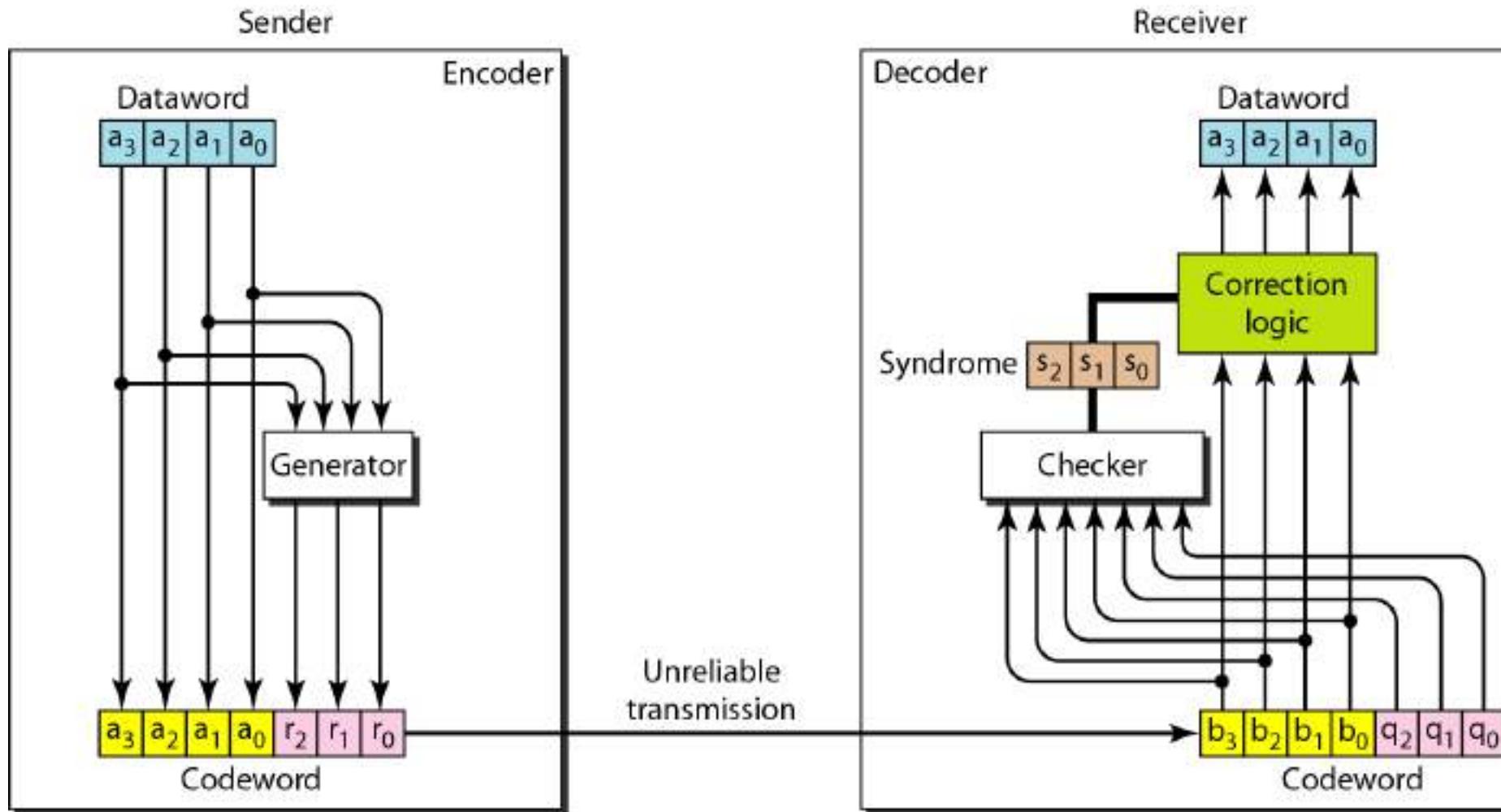
Let  $m = r$ , then we have:

$$n=2^m-1 \quad \text{and}$$

$$k=n-m$$

- For eg. If  $m=3$  then  $n=7$  and  $k=4$ ,
  - Hence it is a Hamming code  $C(7,4)$  with  $d_{\min}=3$

**Figure 10.12** *The structure of the encoder and decoder for a Hamming code*



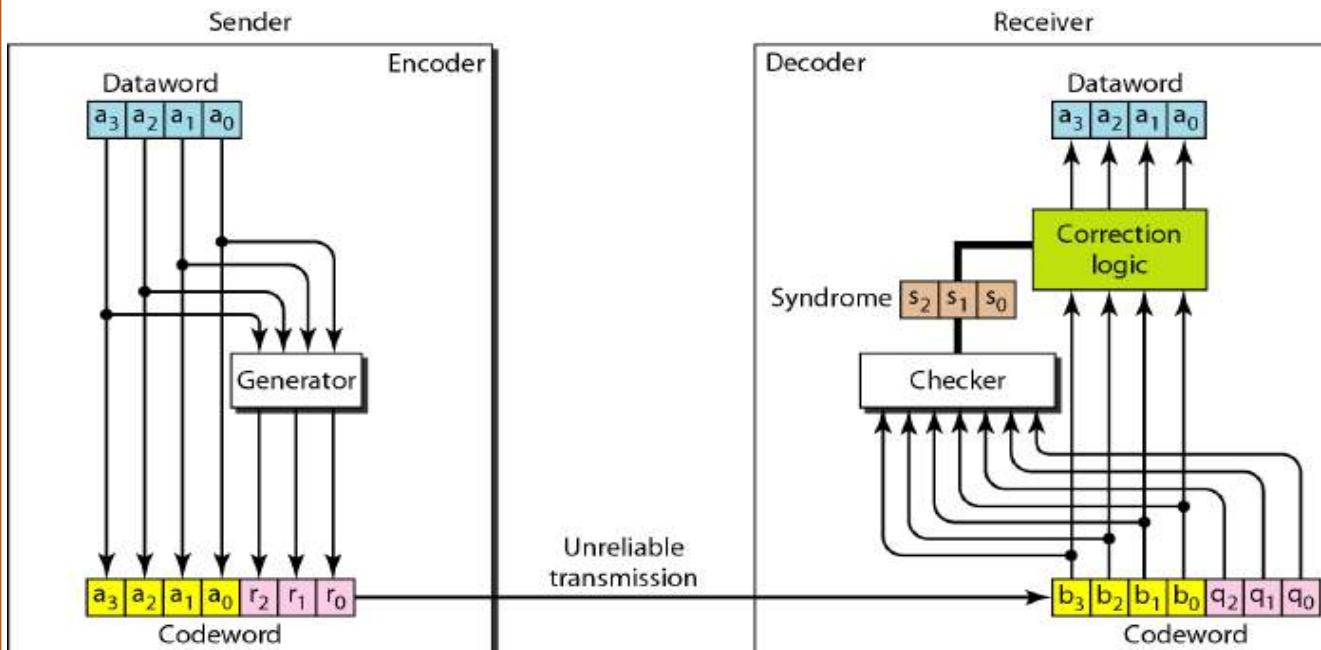
A copy of a 4-bit dataword is fed into the generator that creates three parity checks  $r_0, r_1, r_2$ , as shown below:

$$r_0 = a_2 + a_1 + a_0 \text{ (modulo -2)}$$

$$r_1 = a_3 + a_2 + a_1 \text{ (modulo -2)}$$

$$r_2 = a_1 + a_0 + a_3 \text{ (modulo -2)}$$

Each of parity check bits handles **3 out of 4 bits** of **dataword**. The total number 1s in each 4-bit combination (3 dataword bits & 1 parity bit) must be even.



$$r_0 = a_2 + a_1 + a_0 \quad (\text{modulo } -2)$$

$$r_1 = a_3 + a_2 + a_1 \quad (\text{modulo } -2)$$

$$r_2 = a_1 + a_0 + a_3 \quad (\text{modulo } -2)$$

### ***Hamming code C(7, 4)***

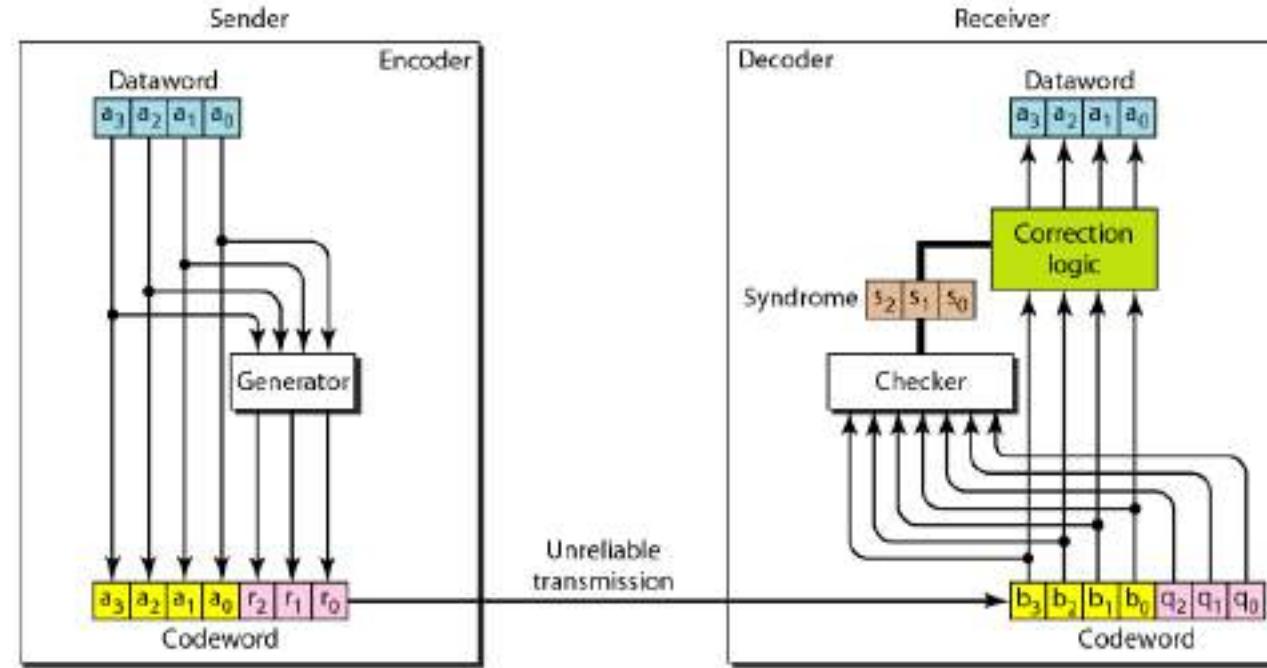
<i>Datawords</i>	<i>Codewords</i>	<i>Datawords</i>	<i>Codewords</i>
0000	0000000	1000	1000110
0001	0001101	1001	1001011
0010	0010111	1010	1010001
0011	0011010	1011	1011100
0100	0100011	1100	1100101
0101	0101110	1101	1101000
0110	0110100	1110	1110010
0111	0111001	1111	1111111

- At Checker

$$S_0 = b_2 + b_1 + b_0 + q_0 \pmod{2}$$

$$S_1 = b_3 + b_2 + b_1 + q_1 \pmod{2}$$

$$S_2 = b_1 + b_0 + b_3 + q_2 \pmod{2}$$



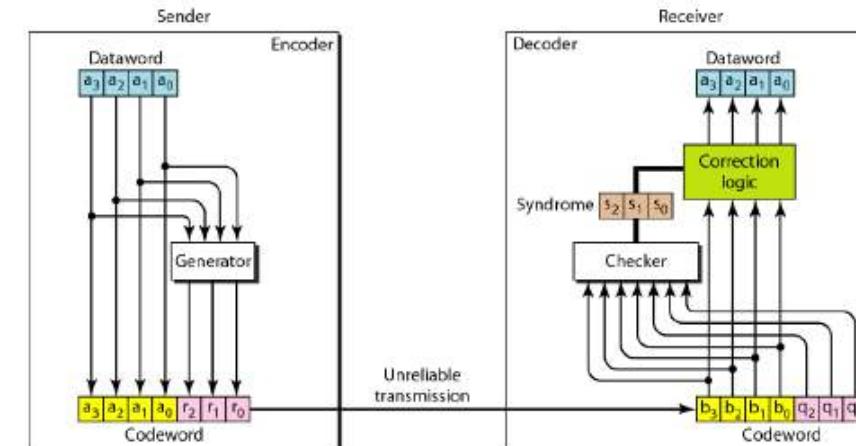
- The 3 bit syndrome creates 8 different bit patterns (000 to 111) that can represent 8 different conditions.

Syndrome	000	001	010	011	100	101	110	111
Error	None	$q_0$	$q_1$	$b_2$	$q_2$	$b_0$	$b_3$	$b_1$

Fig: logical decision made by the correction logic Analyzer

- In Syndrome 000, 001, 010 & 100 is not concerned with generator since there is error or an error in the parity bit.
- In Syndrome 011, 101, 110, 111 one of the bits must be flipped (from 0 to 1 or from 1 to 0) to find correct dataword.

$$\begin{aligned} S_0 &= b_2 + b_1 + b_0 + q_0 \pmod{2} \\ S_1 &= b_3 + b_2 + b_1 + q_1 \pmod{2} \\ S_2 &= b_1 + b_0 + b_3 + q_2 \pmod{2} \end{aligned}$$

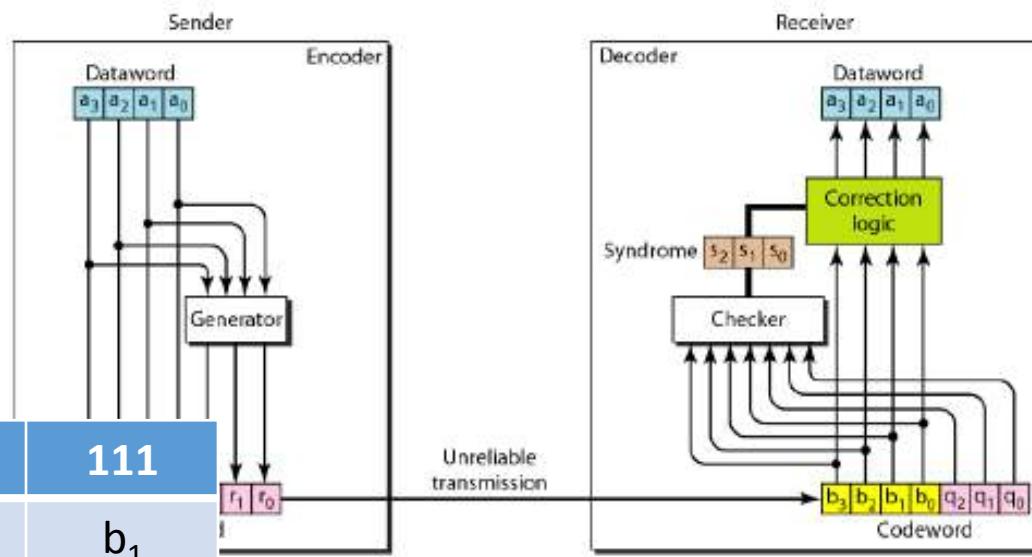


## Cases:

1. Dataword **0100** becomes **0100011** codeword **0100011** is received, syndrome **000**. no error

2. Dataword **0111** becomes **0111001**. Codeword **0011001** is received. Syndrome is **011**. Hence the error is  $b_2$ . After flipping  $b_2$  (changing 0 to 1), the final dataword **0111** is obtained.

Dataword	Codeword
0000	0000000
0001	0001101
0010	0010111
0011	0011010
<b>0100</b>	<b>0100011</b>
0101	0101110
0110	0110100
<b>0111</b>	<b>0111001</b>
1000	1000110



Syndrome	000	001	010	011	100	101	110	111	$r_1, r_0$
Error	None	$q_0$	$q_1$	$b_2$	$q_2$	$b_0$	$b_3$	$b_1$	

3. The dataword 1101 is sent as codeword **1101000**. Receiver receives **0001000** (2 errors). The syndrome is 101 i.e  $b_0$  is in error.

After flipping  $b_0$  we get 0000 (wrong dataword).

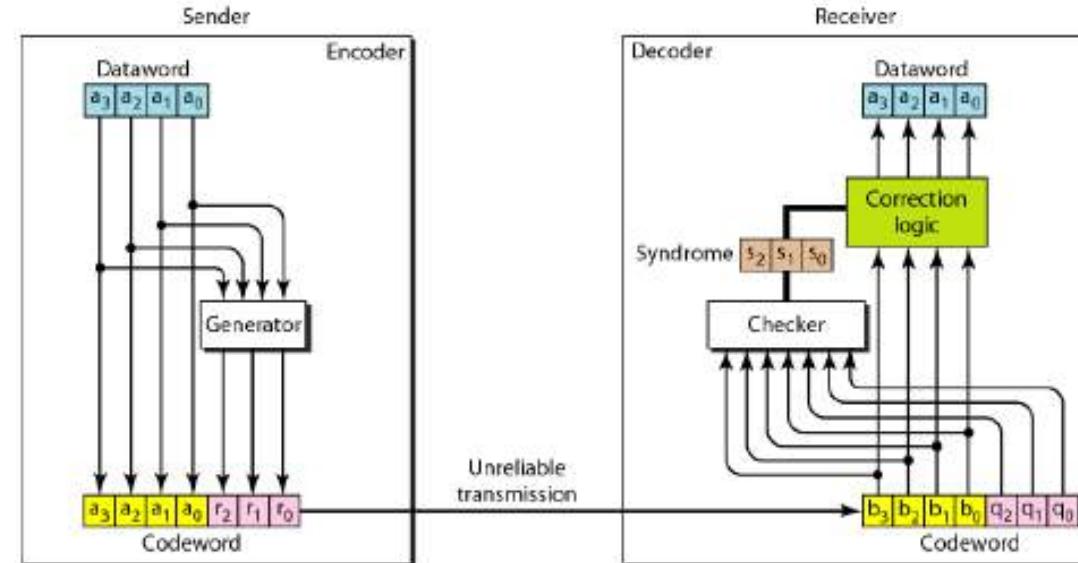
**Hence two error cannot be corrected but can be detected. Single error is detected and corrected.**

Dataword	Codeword
1101	1101000
1110	1110010
1111	1111111

$$S_0 = b_2 + b_1 + b_0 + q_0 \pmod{2}$$

$$S_1 = b_3 + b_2 + b_1 + q_1 \pmod{2}$$

$$S_2 = b_1 + b_0 + b_3 + q_2 \pmod{2}$$



Syndrome	000	001	010	011	100	101	110	111
Error	None	$q_0$	$q_1$	$b_2$	$q_2$	$b_0$	$b_3$	$b_1$

# Cyclic Codes

- Cyclic codes are special linear block codes with one extra property.
- In a cyclic code, if a codeword is cyclically shifted(rotated), the result is another codeword.
- For example, if

1 0 1 1 0 0 0

is a codeword and we cyclically left-shift, then

0 1 1 0 0 0 1

is also a codeword.

- In this case, if we call the bits in the first word  $a_0$  to  $a_6$ , and the bits in the second word  $b_0$  to  $b_6$ , we can shift the bits by using the following:

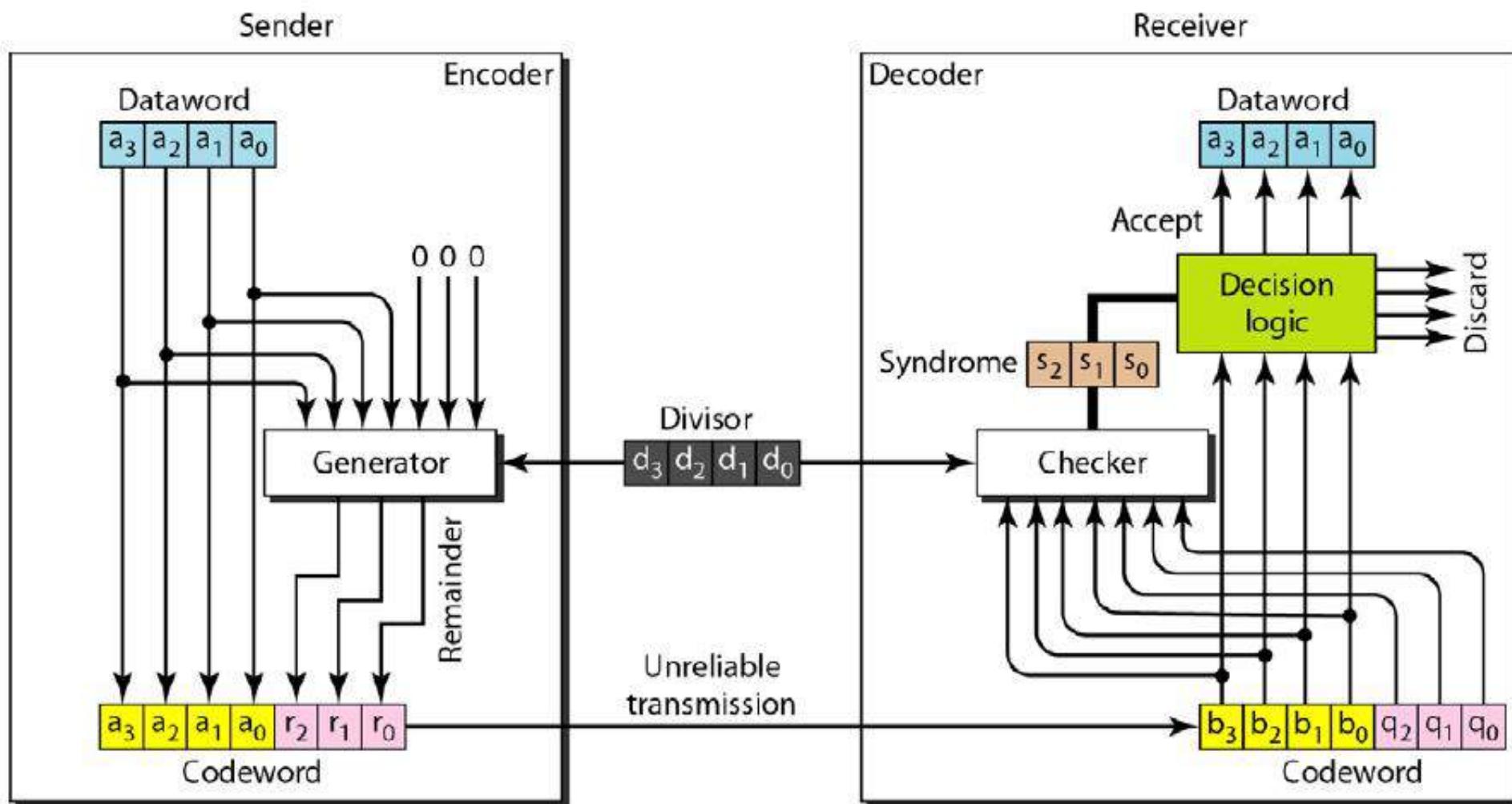
$$b_1=a_0, b_2=a_1, b_3=a_2, b_4=a_3, b_5=a_4, b_6=a_5, b_0=a_6$$

- A category of cyclic code is Cyclic Redundancy Check (CRC)
- Used in LANs & WAN's

# Cyclic Redundancy Check

**Table 10.6** A CRC code with  $C(7, 4)$

<i>Dataword</i>	<i>Codeword</i>	<i>Dataword</i>	<i>Codeword</i>
0000	0000000	1000	1000101
0001	0001011	1001	1001110
0010	0010110	1010	1010011
0011	0011101	1011	1011000
0100	0100111	1100	1100010
0101	0101100	1101	1101001
0110	0110001	1110	1110100
0111	0111010	1111	1111111



The generator uses  $n-k+1$  bit divisor which is predefined and agreed

The remainder (3 bit) is appended to the dataword to make the final codeword.

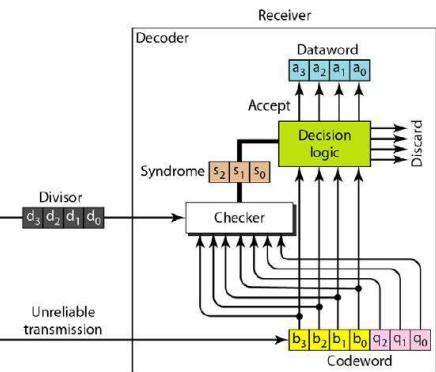
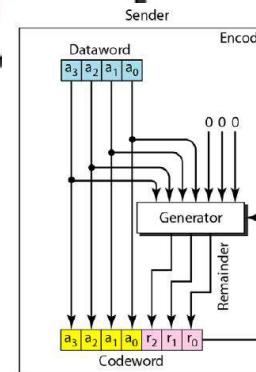
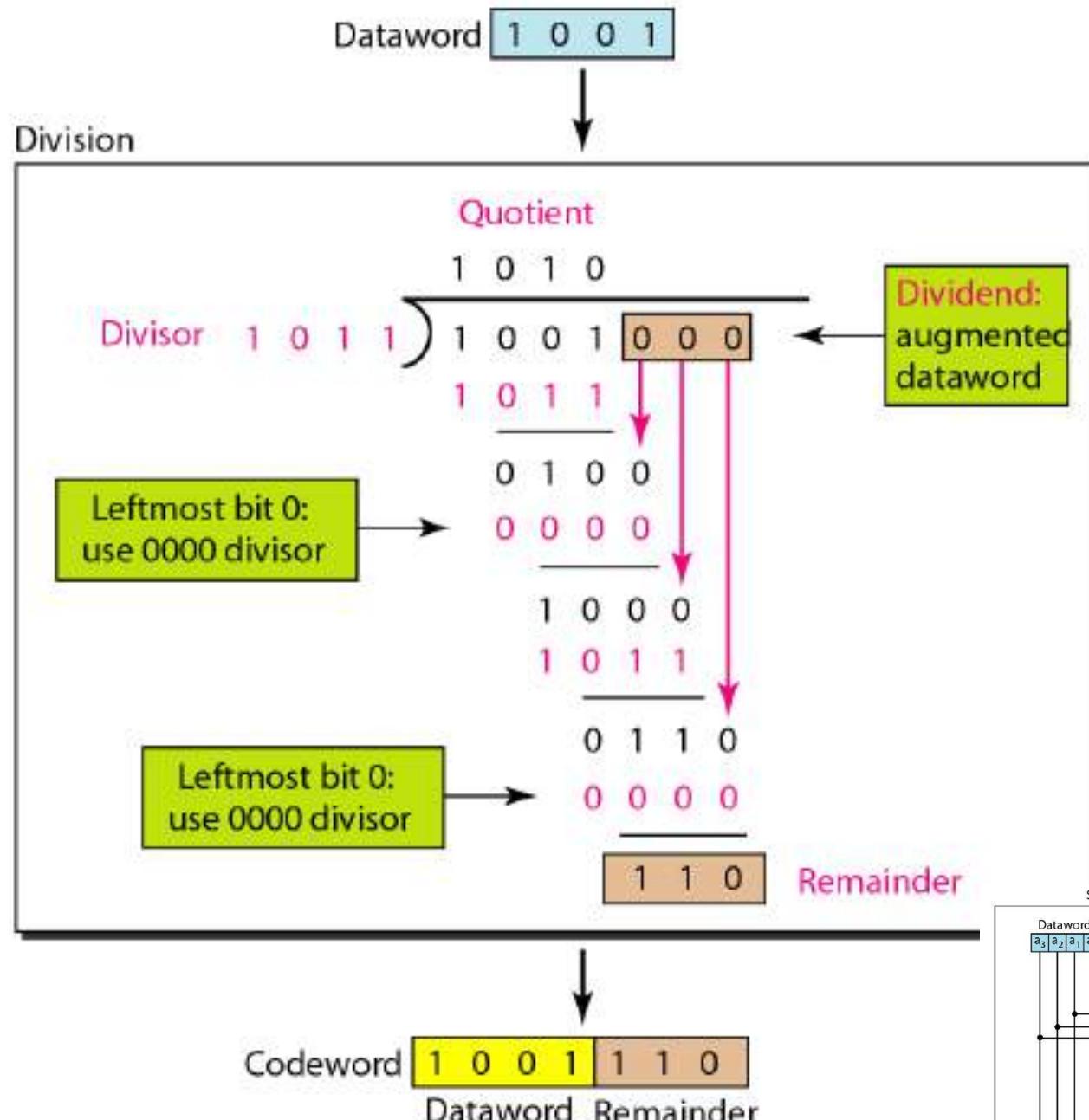
# At Encoder

- In encoder, the dataword has **k** bits (4 here)
- The codeword has **n** bits (7 here)
- The size of the dataword is augmented by adding **n – k** (3 here)
- The generator uses a **divisor of size  $n-k+1$**  (4 here)
- The quotient of the division is discarded.
- The remainder ( $r_2r_1r_0$ ) is **appended** to the dataword to create the codeword.

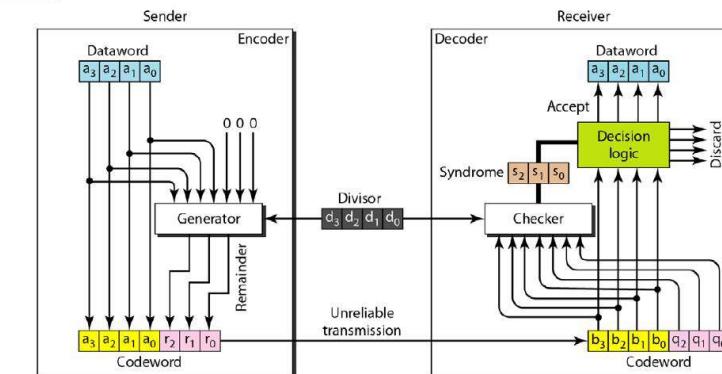
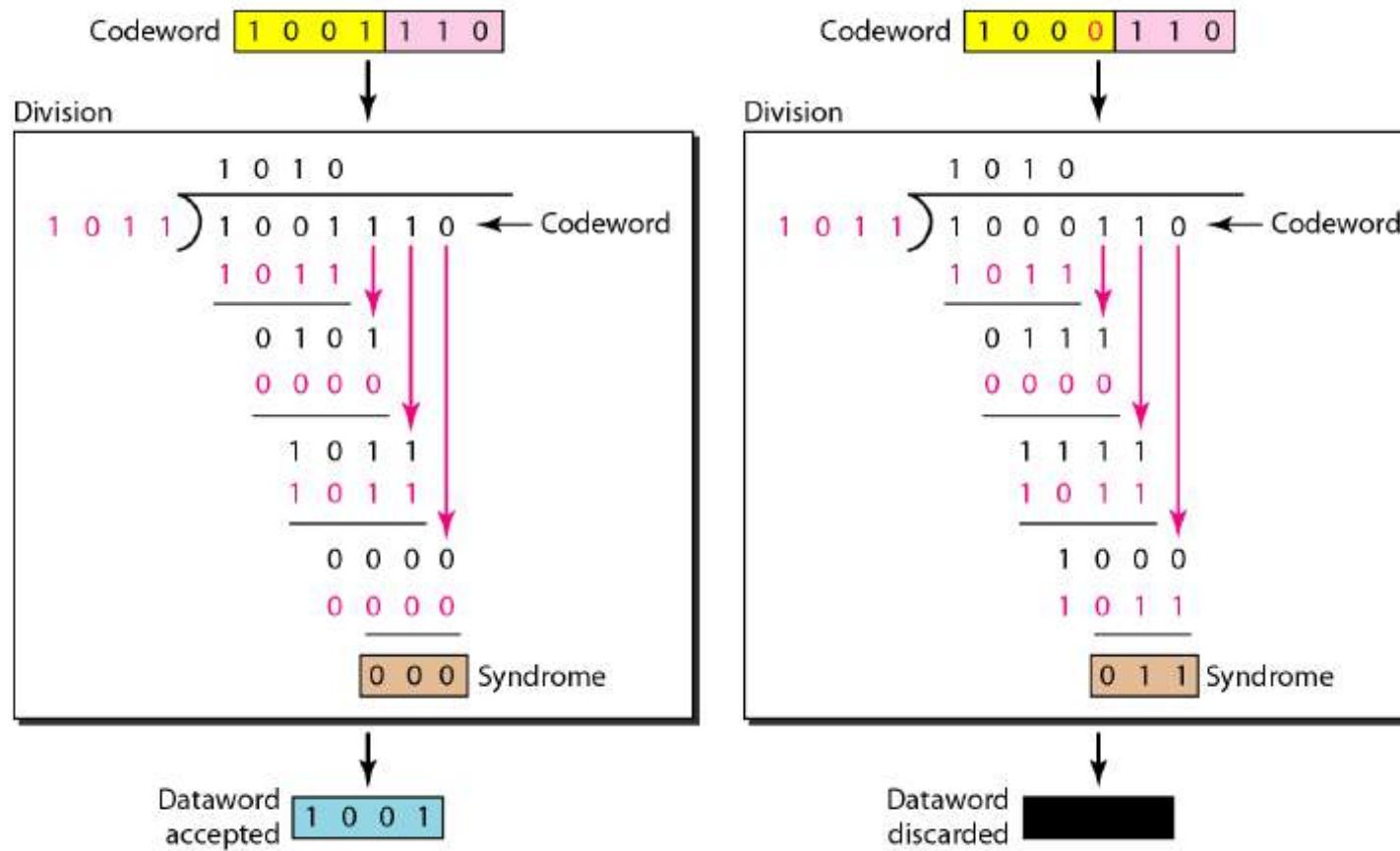
- Eg. Dataword =1001 ( $k=4$ ), & if codeword ( $n$ ) = 7 bits.
- Augment dataword by  $(n-k)$  zero bits i.e  $7-4=3$  zero bits.
  - i.e 1001000
- Let the divisor is 1101 (Predefined and agreed)  $(n-k+1)$  digits
- The remainder  $(r_2 r_1 r_0)$  is appended to the dataword to create the codeword

## At decoder

- Decoder does the same division as encoder.
- Eg. Dataword =1001 ( $k=4$ ), & if codeword ( $n$ ) = 7 bits.
- Augment D.W by  $(n-k)$  zero bits i.e  $7-4=3$  zero bits.
  - i.e 1001000
- Let the divisor is 1101 (Predefined and agreed)  $(n-k+1)$
- If remainder is all 0's there is no error, else there is error.



**Figure 10.16** Division in the CRC decoder for two cases

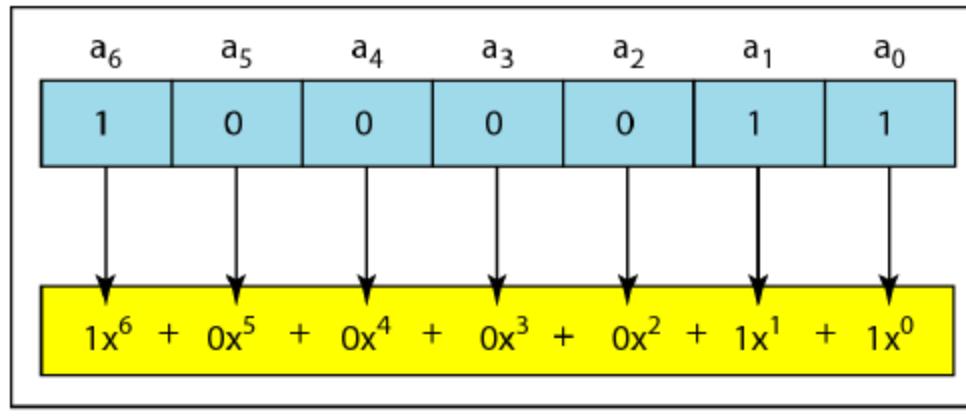


# Polynomials

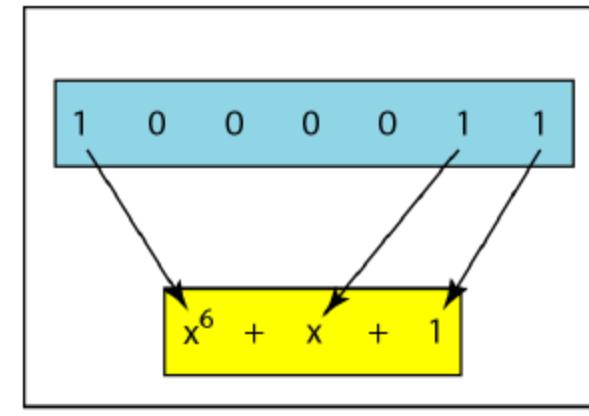
A better way to understand cyclic codes and how they can be analyzed is to represent them as polynomials.

A pattern of 0s and 1s can be represented as a polynomial with coefficients of 0s and 1.

**Figure 10.21** *A polynomial to represent a binary word*



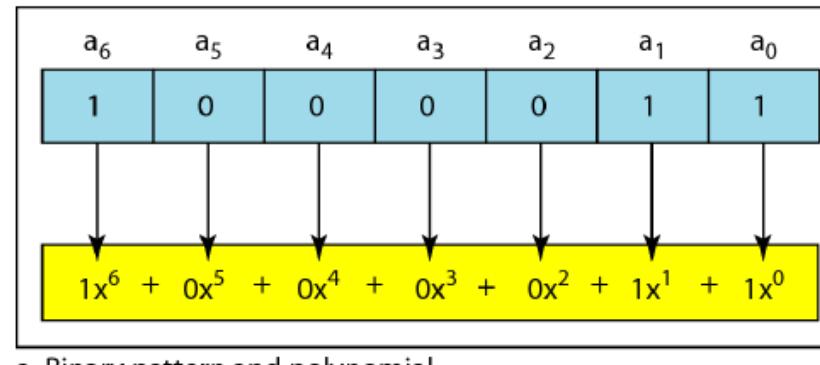
a. Binary pattern and polynomial



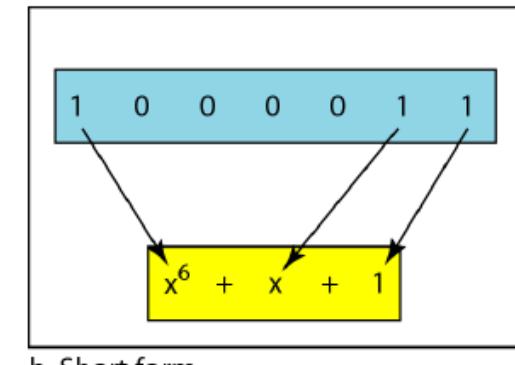
b. Short form

- **Degree of Polynomial** : It is the highest power
  - Eg  $x^6+x+1 \Rightarrow 6$
- Note: The degree of a polynomial is less than that the number of bits in the pattern. The bit pattern in this case has 7 bits.
  - So general D.P  $\Rightarrow n-1$  bit if n is number of bits.

**Figure 10.21 A polynomial to represent a binary word**



a. Binary pattern and polynomial



b. Short form

## Addition & Subtraction :

Addition or subtraction is done by combining terms and deleting pairs of identical terms.

$$\text{Eg. } (x^5+x^4+x^2) + (x^6+x^4+x^2) \Rightarrow x^6 + x^5$$

$x^4$  and  $x^2$  are deleted

- **Multiplying or dividing terms:**
  - Just add the powers of multiplication.
    - Eg.  $X^3 \times X^4 = X^7$
  - Subtract the powers for division.
    - Eg.  $X^5/X^4 = X$

# Multiplying two polynomials

- It is conceptually same as we did it for the encoder/decoder, pairs equal terms are deleted.

- $$\bullet (x^5+x^3+x^2+x) (x^2+x+1)$$

*(let the student do rest of the problem)*

- $$\bullet x^7+x^6+x^3+x^2+x$$

## Shifting

- A binary pattern is often shifted a number of bits to the right or left.
- Shifting to the left means adding 0's extra bit at right side.
- Shifting to the Right means deleting some right most bits.

Sifting left 3 bits:

$$10011 \Rightarrow 10011000$$

$$X^4 + X + 1 \Rightarrow X^7 + X^4 + X^3$$

Sifting right 3 bits :

- $10011 \Rightarrow 10$

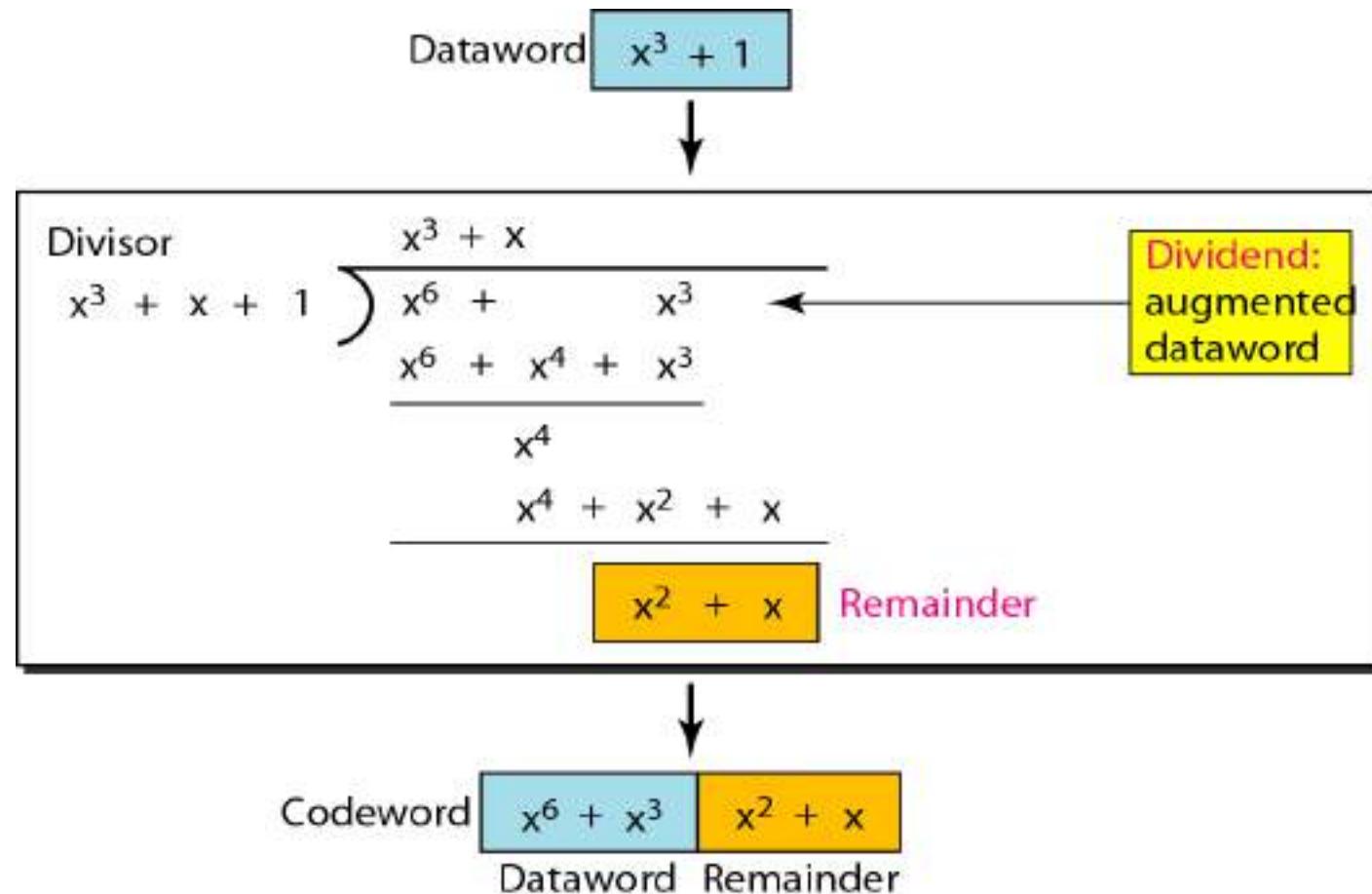
- $X^4 + X + 1 \Rightarrow X$

# Cyclic Code Encoder Using Polynomials

- Cyclic Code Encoder Using Polynomial
  - Dataword 1001 =>  $x^3+1$
  - Divisor 1011 =>  $x^3+x+1$
- To find the augmented dataword, we have left-shifted the dataword 3 bits (multiplying by  $x^3$ )
- Finally we augmented data is =>  $x^6+x^3$

# Cyclic Redundancy Check

Figure 10.22 CRC division using polynomials



# Equivalence of Polynomial and Binary

	Polynomial	Binary Vector
Generator, $g(x)$	$x^3+x+1$	1011
Data, $f(x)$	$x^3+x^2$	1100
$x^n f(x)$	$x^6+x^5$ $\frac{x^3+x^2+x}{x^3+x+1}$	1100000 1110
Division $x^n f(x) \div g(x)$	$  \begin{array}{r}  x^3+x+1 ) \overline{x^6+x^5} \\  \quad \quad \quad x^6+x^4+x^3 \\  \hline  \quad \quad \quad x^5+x^4+x^3 \\  \quad \quad \quad x^5+x^3+x^2 \\  \hline  \quad \quad \quad x^4+x^2 \\  \quad \quad \quad x^4+x^2+x \\  \hline  \quad \quad \quad x  \end{array}  $	$  \begin{array}{r}  1011 ) \overline{1100000} \\  \quad \quad \quad 1011 \\  \hline  \quad \quad \quad 111000 \\  \quad \quad \quad 1011 \\  \hline  \quad \quad \quad 10100 \\  \quad \quad \quad 1011 \\  \hline  \quad \quad \quad 10  \end{array}  $
Codeword, $x^n f(x) - r(x)$	$x^6+x^5+x$	1100010

- In a polynomial representation, the divisor is normally referred to as the generator polynomial  $g(x)$ .

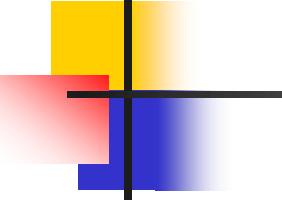
---

**The divisor in a cyclic code is normally called the generator polynomial or simply the generator.**

---

# Cyclic Code Analysis

- The divisor is normally called the generator polynomial or generator
- Analysis:
  - Let  $f(x)$  is a polynomial with binary coefficient
  - $d(x)$ : Dataword
  - $c(x)$ : Codeword
  - $g(x)$ : Generator
  - $s(x)$ : Syndrome
  - $e(x)$ : Error



## **Note**

---

**In a cyclic code,**

**If  $s(x) \neq 0$ , one or more bits is corrupted.**

**If  $s(x) = 0$ , either**

- a. No bit is corrupted. or**
  - b. Some bits are corrupted, but the decoder failed to detect them.**
-

- To find the **criteria** that must be imposed on the generator  $g(x)$  to detect the type of error that need to be detected.
- Relationship among the **sent codeword**, **error**, **received codeword**, and the **generator**.

$$\text{Received Codeword} = c(x) + e(x)$$

- The receiver divides the received codeword by  $g(x)$  to get the syndrome. We can write this

$$\frac{\text{Received Codeword}}{g(x)} = \frac{c(x)}{g(x)} + \frac{e(x)}{g(x)}$$

$$\frac{\text{Received Codeword}}{g(x)} = \frac{c(x)}{g(x)} + \frac{e(x)}{g(x)}$$

The **first term** at the right hand side of the equality has a remainder of zero (according to the definition of codeword).

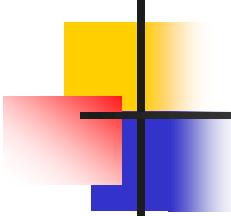
So the syndrome is actually the remainder of the **second term** on the right-hand side.

If the term does not have remainder (syndrome=0), either

1.  $e(x) = 0$  or
2.  $e(x)$  is divisible by  $g(x)$ .

We don't need to worry about first case, but

The second term is important. Those errors that are not divisible by  $g(x)$  are not caught.



---

**In a cyclic code, those  $e(x)$  errors that are divisible by  $g(x)$  are not caught.**

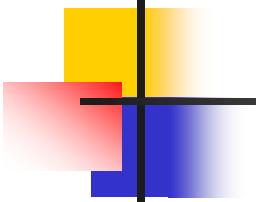
---

# Single bit error.

- Question is :

What should be the structure of  $g(x)$  to guarantee the detection of single-bit error?

- Here, A single bit error  $e(x) = x^i$  where  $i$ =position of error bit.
- then  $x^i/g(x) \Rightarrow$  will have a remainder.
- If  $g(x)$  have at least two terms and the coefficient of  $x^0$  is 1 then  $e(x)$  cannot be divided by  $g(x)$  i.e there will be some remainder.

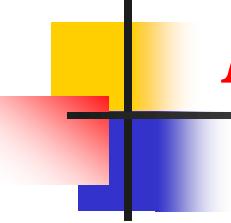


### **Note**

---

**If the generator has more than one term  
and the coefficient of  $x^0$  is 1,  
all single errors can be caught.**

---



## *Example 10.8*

*Which of the following  $g(x)$  values guarantees that a single-bit error is caught? For each case, what is the error that cannot be caught?*

- a.*  $x + 1$
- b.*  $x^3$
- c.* 1

### *Solution*

- a.* No  $x^i$  can be divisible by  $x + 1$ . Any single-bit error can be caught.
- b.* If  $i$  is equal to or greater than 3,  $x^i$  is divisible by  $g(x)$ . All single-bit errors in positions 1 to 3 are caught.
- c.* All values of  $i$  make  $x^i$  divisible by  $g(x)$ . No single-bit error can be caught. This  $g(x)$  is useless.

**Table 10.7** *Standard polynomials*

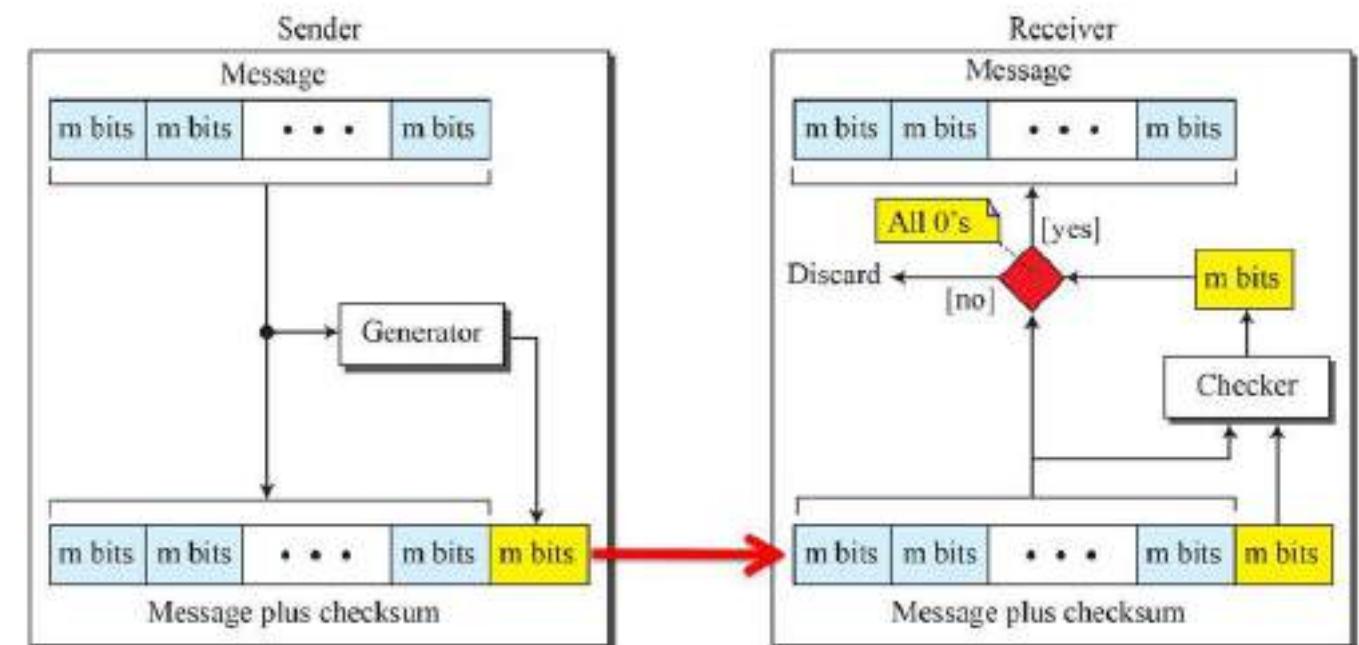
Name	Polynomial	Application
CRC-8	$x^8 + x^2 + x + 1$	ATM header
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$	LANs

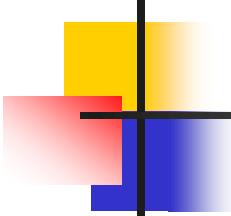
# Checksum

- **Checksum is an error detecting** technique that can be applied to a message of any length.
- In the Internet, the checksum technique is mostly used at the **network and transport layer rather than the data-link layer**.

- At the source, the message is first divided into **m-bits** units.
- The generator then creates an **extra m-bit** unit called the checksum, which is sent with the message.
- At the destination, the checker creates a new checksum from the combination of the message and sent checksum.
- If the checksum is all 0s, the message is accepted; otherwise the message is discarded.

*Figure 10.15: Checksum*





## *Example 10.18*

*Suppose our data is a list of five 4-bit numbers that we want to send to a destination.*

In addition to sending these numbers, we send the sum of the numbers.

For example, if the set of numbers is (7, 11, 12, 0, 6), we send (7, 11, 12, 0, 6, 36), where 36 is the sum of the original numbers. The receiver adds the five numbers and compares the result with the sum.

*If the two are the same, the receiver assumes no error, accepts the five numbers, and discards the sum. Otherwise, there is an error somewhere and the data are not accepted.*

- **For Simplicity,**
- Example 10.19:
  - We can make the job of the receiver easier if we send the negative (complement) of the sum, called the **checksum**.
  - In this case, we send (**7,11,12,0,6, -36**). The receiver can add all the numbers received(including the checksum).
  - If the result is 0, assumes no error, otherwise, there is an error.

# One's Complement

- The previous example has one major drawback. All our data can be written as a **4-bit word (they are less than 15)** except the checksum.
- We have solution to use **one's complement** arithmetic.

- In this one's complement arithmetic, we can represent **unsigned numbers** between **0** and  **$2^m-1$**  using only  **$m$**  bits.
- If the number has more than  **$m$**  bits, the extra leftmost bits need to be added to the  **$m$**  rightmost bits (wrapping).
- In one's complement arithmetic, we have two 0s; one positive and one negative, which are complements of each other.

How can we represent the number 36 in one's complement arithmetic using only four bits ?

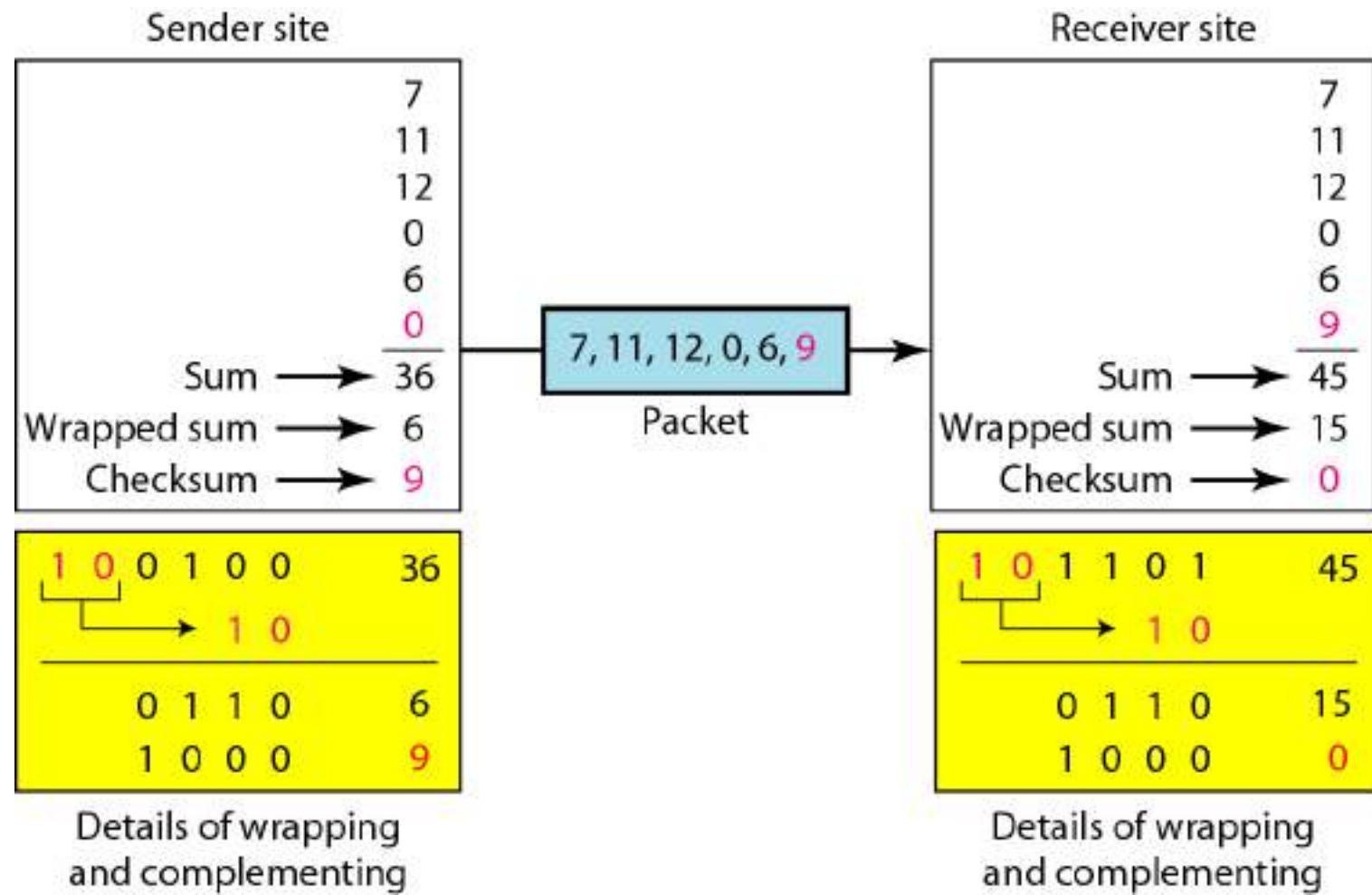
- **Solution**

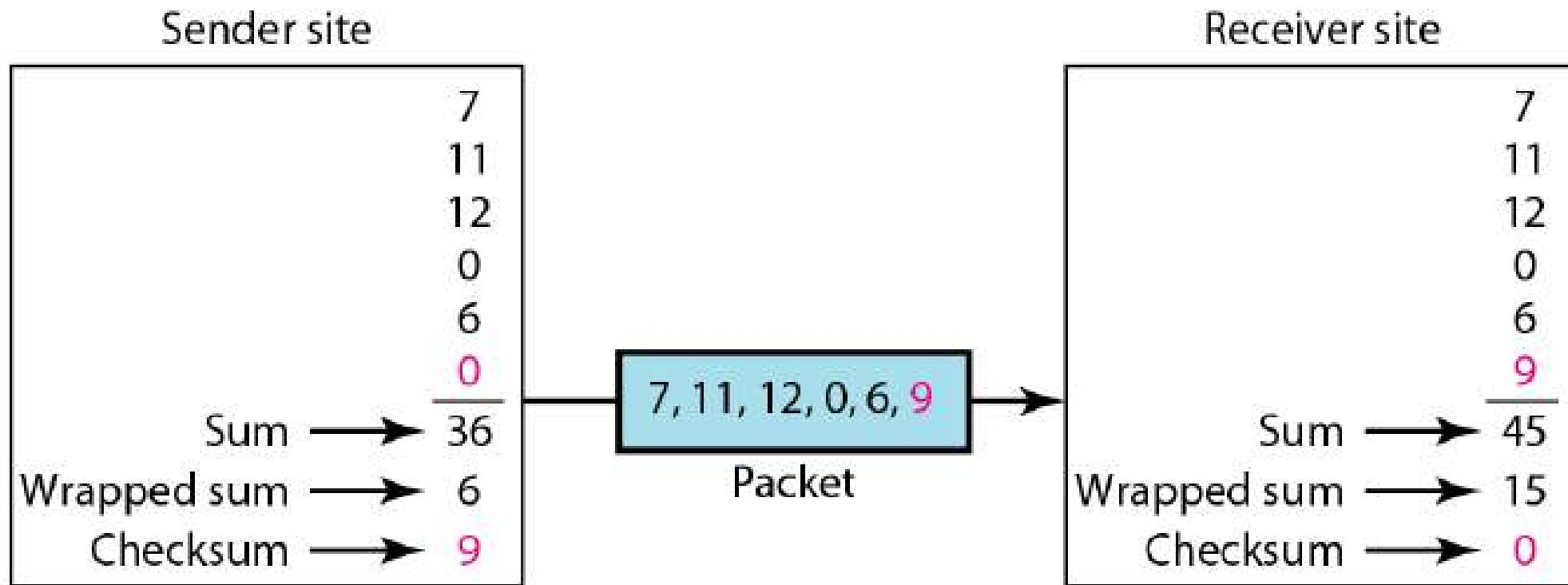
- The number 36 in binary is 100100 (**it needs six bits**).
- We can wrap the leftmost bit and add it to the four rightmost bits.
- We have  $(10)_2 + (0100)_2 = (0110)_2 = (6)_{10}$

How can we represent the number 6 in one's complement arithmetic using only four bits ?

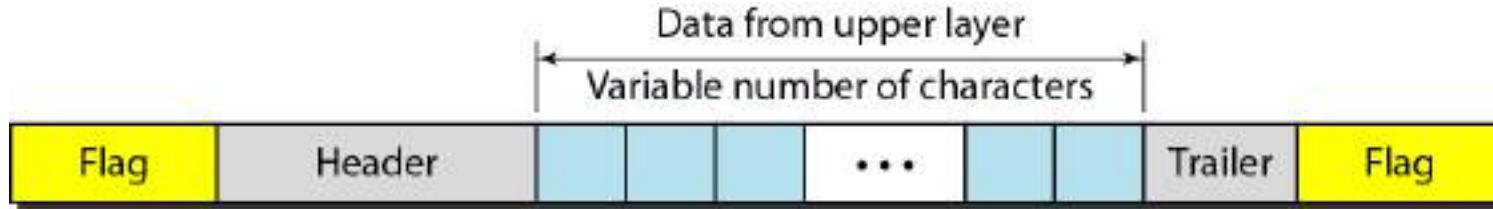
- Solution
  - In one's complement arithmetic, the negative or complement of a number is found by inverting all bits.
  - Positive 6 is 0110;
  - One's complement of 6 is 1001;
  - If we consider only unsigned number, this is 9.
  - In other words, the complement of 6 is 9.

Figure 10.24 Example 10.22





# Data Link Control



- **Framing:**

- The data link layer translates the physical layers raw bit stream into discrete units(message) called frames.
- **Flag** : Signals that **start or end** of the frame
- **Header**: Address of the sender and receiver
- **Data**: Actual message to be sent
- **Trailer**: Error correction and error detection redundant bits.

- There are two types of Framing
  - Fixed sized Framing
  - Variable sized Framing
- In fixed size framing, the boundaries of a frame is not required as the size of frame defines its size. An example of this type of framing is the ATM wide-area network, which uses frames of fixed size called cells.
- In variable sized frame, we need to define the start and end of the text.
  - Character Oriented (Byte Stuffing)
  - Bit Oriented (Bit stuffing)

# Character-oriented Framing

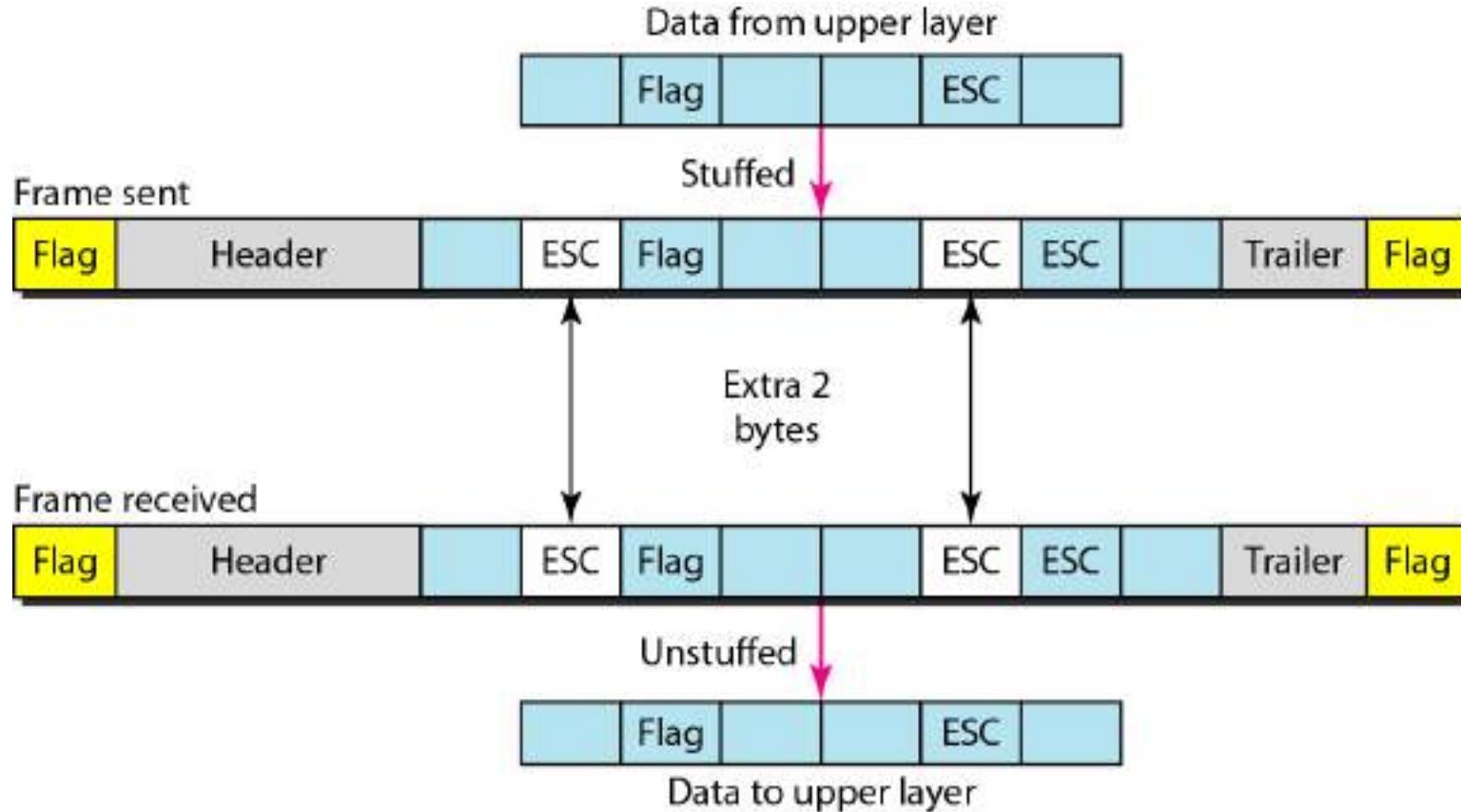
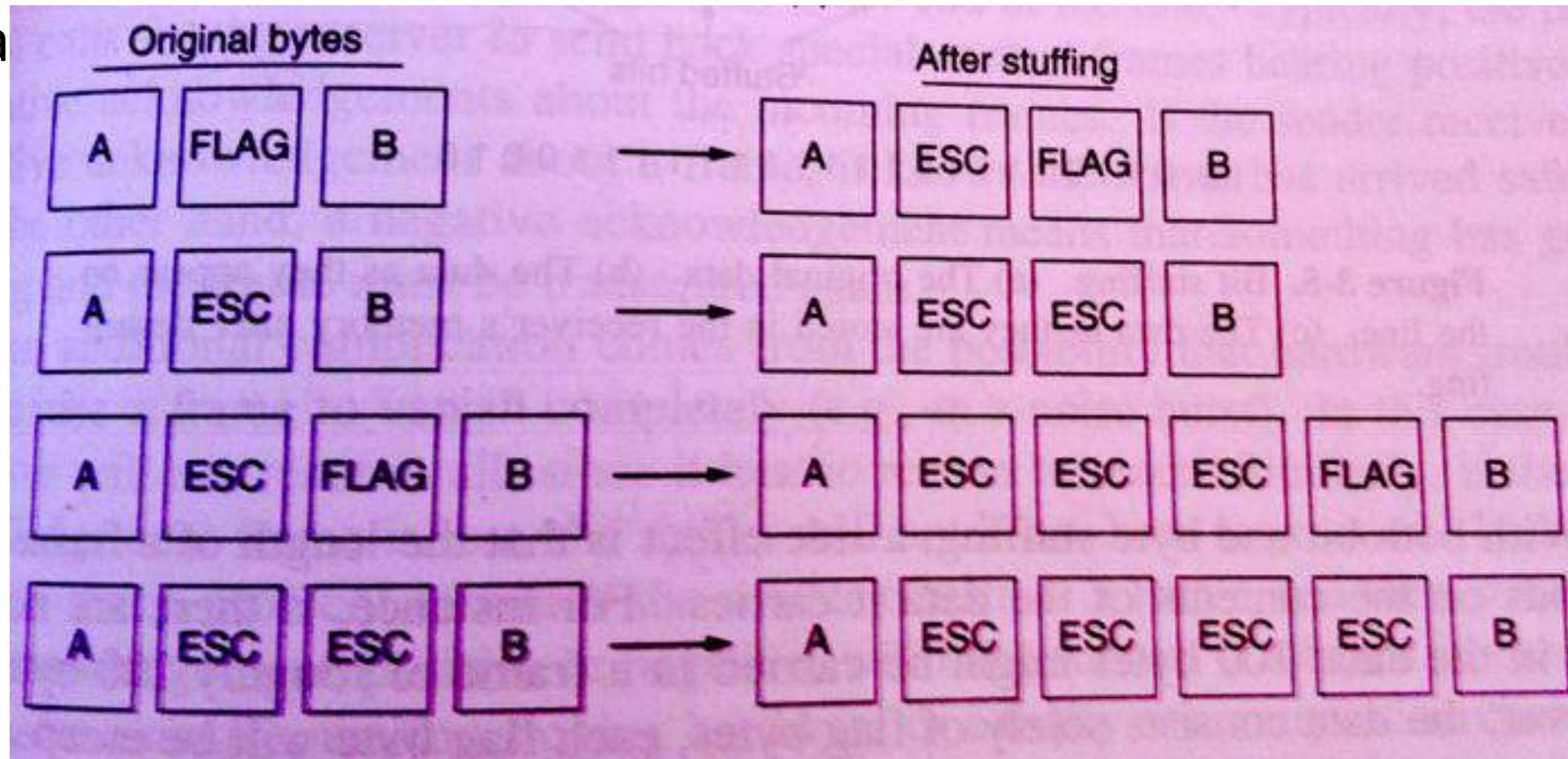


Figure 11.2 *Byte stuffing and unstuffing*

- What happens if we receive



**Bit stuffing** is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern 0111110 for a flag.

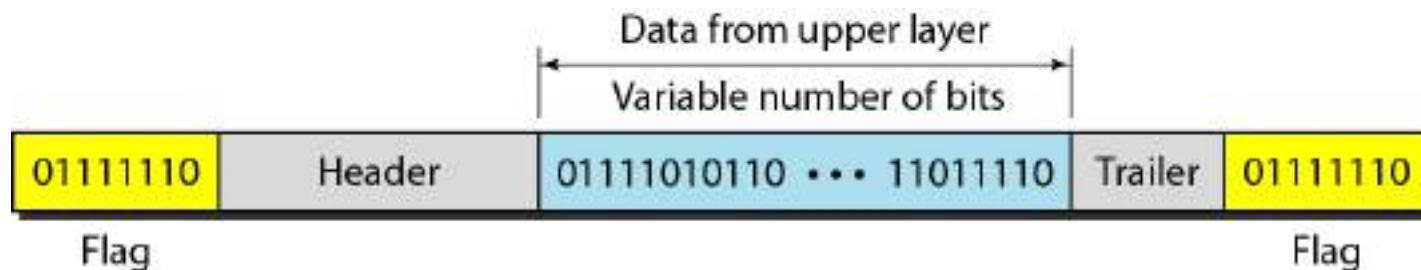
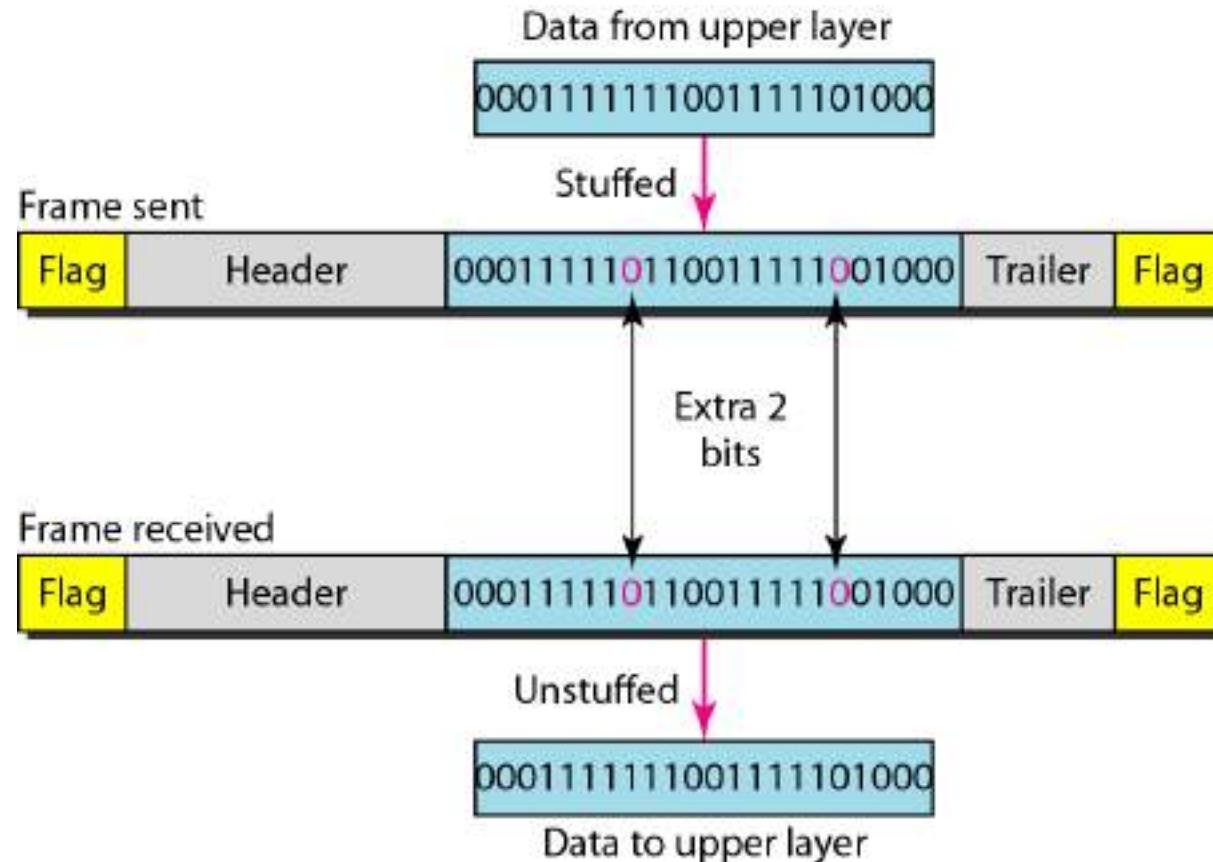


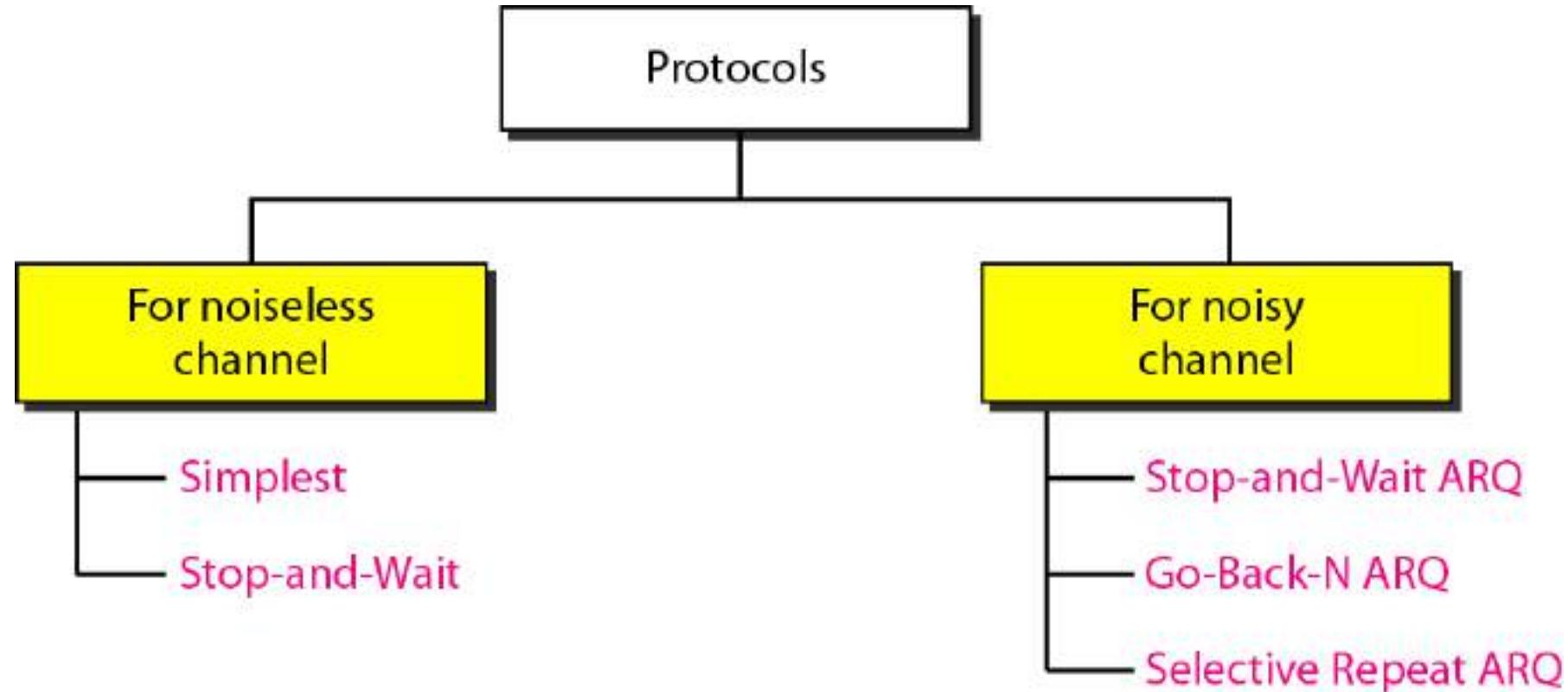
Figure 11.3 A frame in a bit-oriented protocol

Figure 11.4 Bit stuffing and unstuffing



# Flow and Error Control

**Figure 11.5** Taxonomy of protocols discussed in this chapter



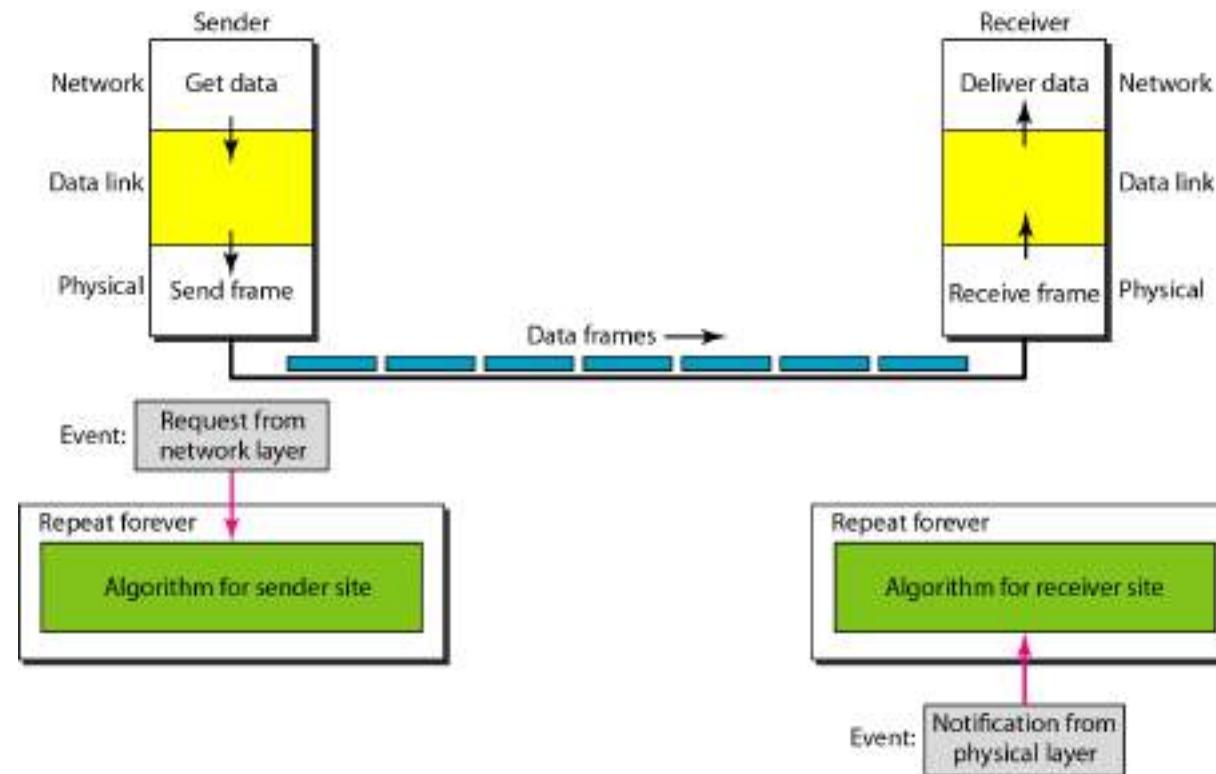
## 11-4 NOISELESS CHANNELS

*Let us first assume we have an ideal channel in which no frames are lost, duplicated, or corrupted. We introduce two protocols for this type of channel.*

*Topics discussed in this section:*

Simplest Protocol

Stop-and-Wait Protocol



- **Simplest Protocol**
  - No error or flow control
  - Unidirectional
  - Receiver can handle any frame immediately
  - Time is negligible
  - Never over bandwidth

## Algorithm 11.1 *Sender-site algorithm for the simplest protocol*

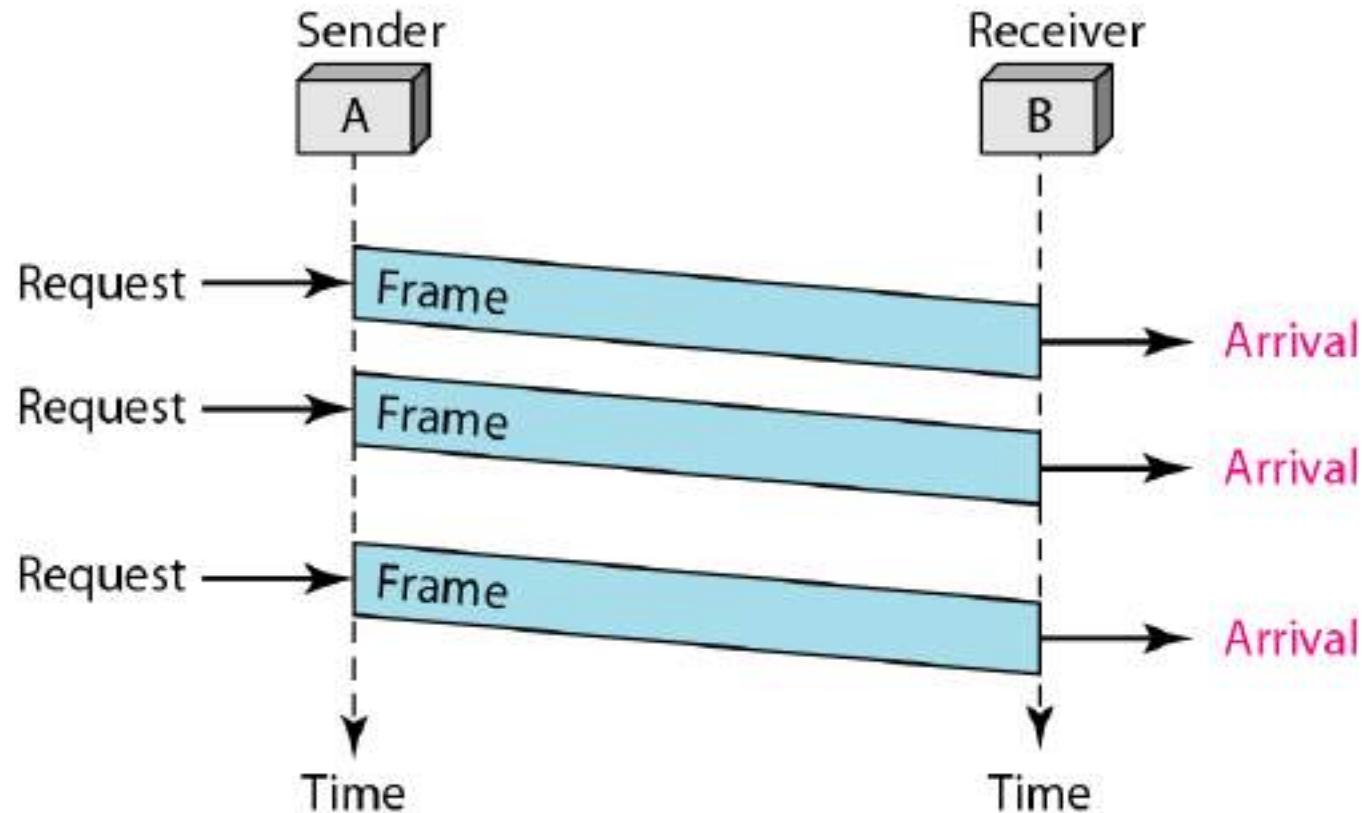
```
Algorithm 11.1 Sender-site algorithm for the simplest protocol
1 while(true)                                // Repeat forever
2 {
3     WaitForEvent();                         // Sleep until an event occurs
4     if(Event(RequestToSend))               // There is a packet to send
5     {
6         GetData();
7         MakeFrame();
8         SendFrame();                      // Send the frame
9     }
10 }
```

The algorithm has an infinite loop, which means line 3 to 9 are repeated forever.

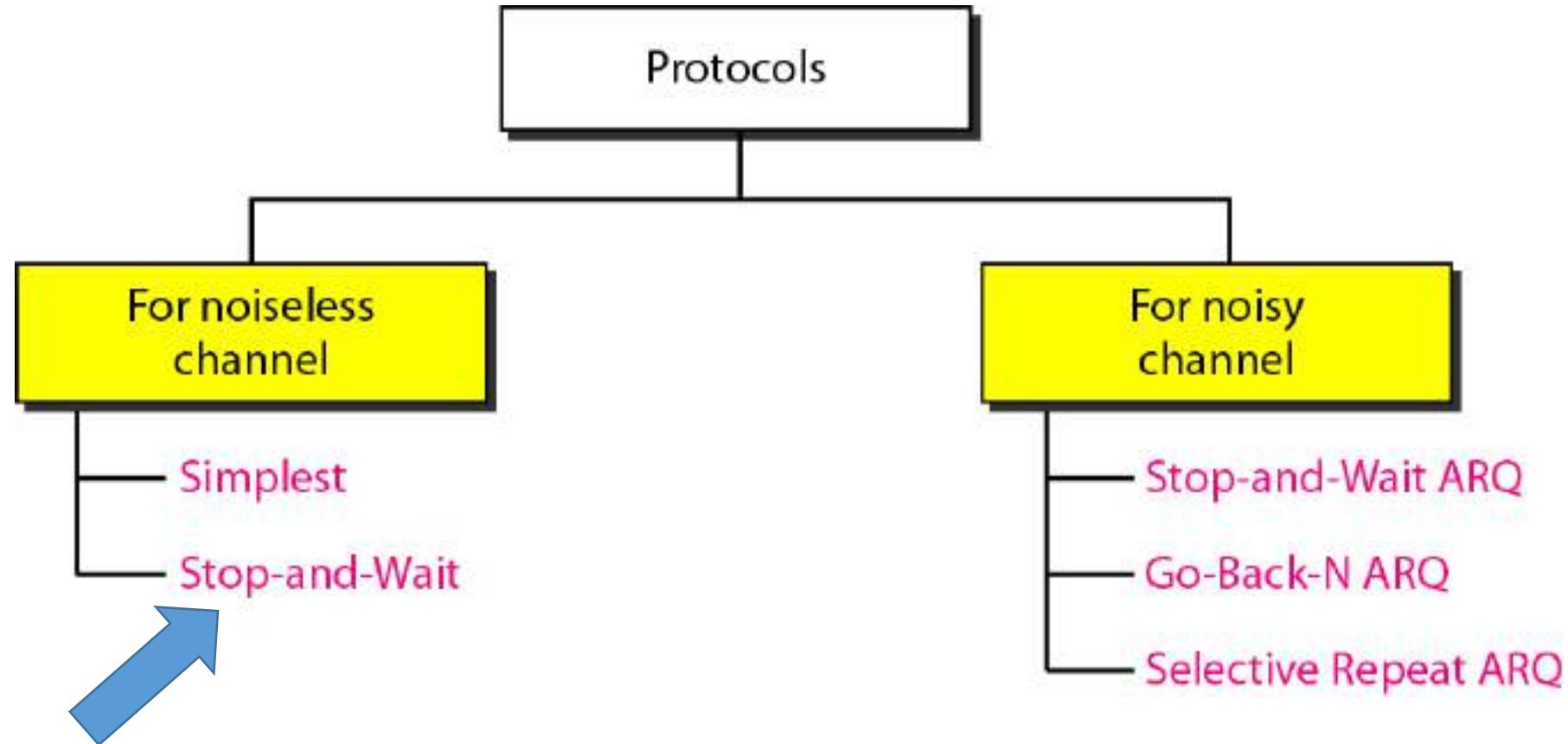
## Algorithm 11.2 *Receiver-site algorithm for the simplest protocol*

```
1 while(true)                                // Repeat forever
2 {
3     WaitForEvent();                         // Sleep until an event occurs
4     if(Event(ArrivalNotification)) //Data frame arrived
5     {
6         ReceiveFrame();
7         ExtractData();
8         DeliverData();                  //Deliver data to network layer
9     }
10 }
```

- It is a unidirectional protocol in which data frames are traveling in only one direction from the sender to receiver.



**Figure 11.5** Taxonomy of protocols discussed in this chapter



# Stop and wait Protocol

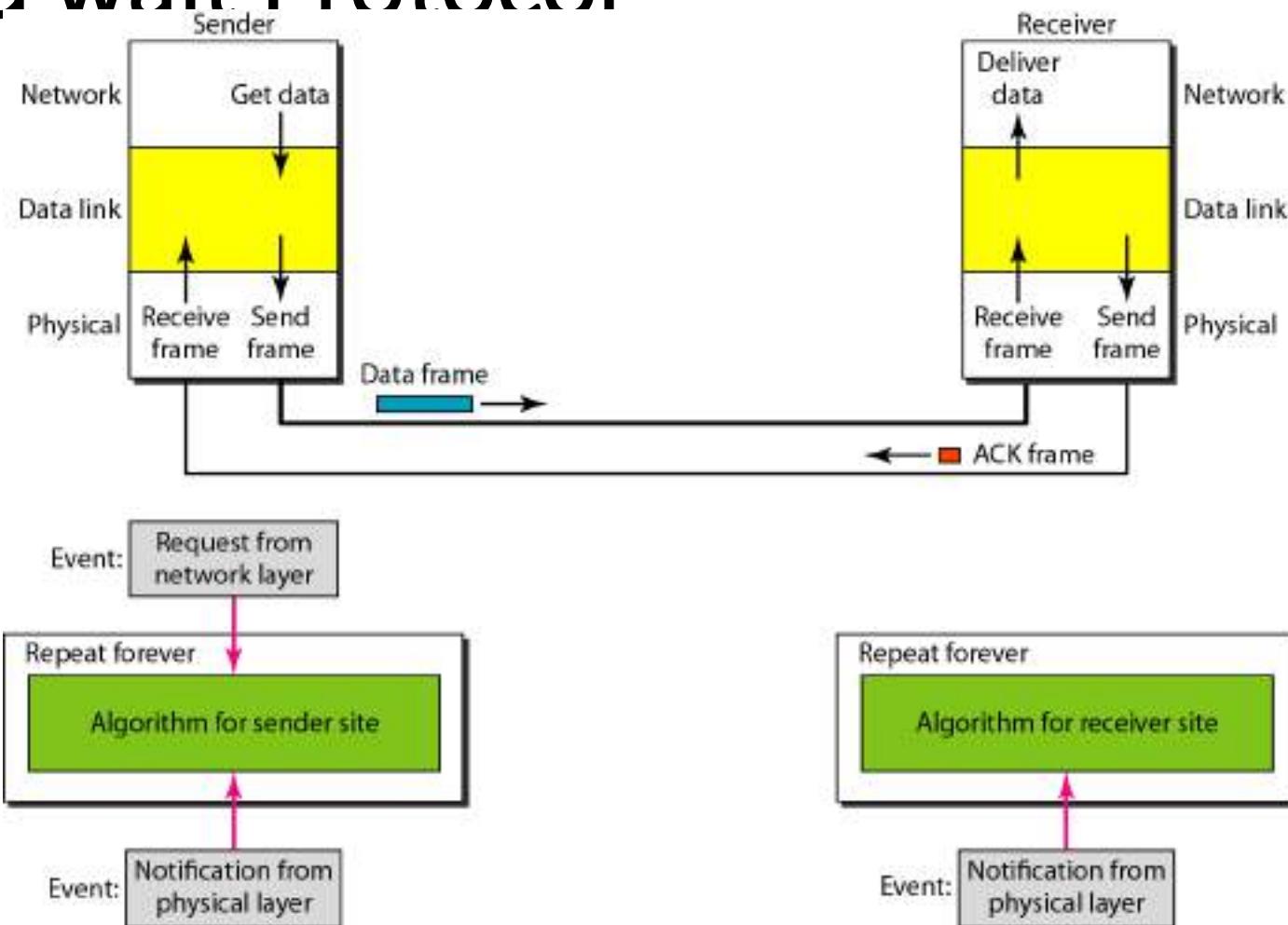
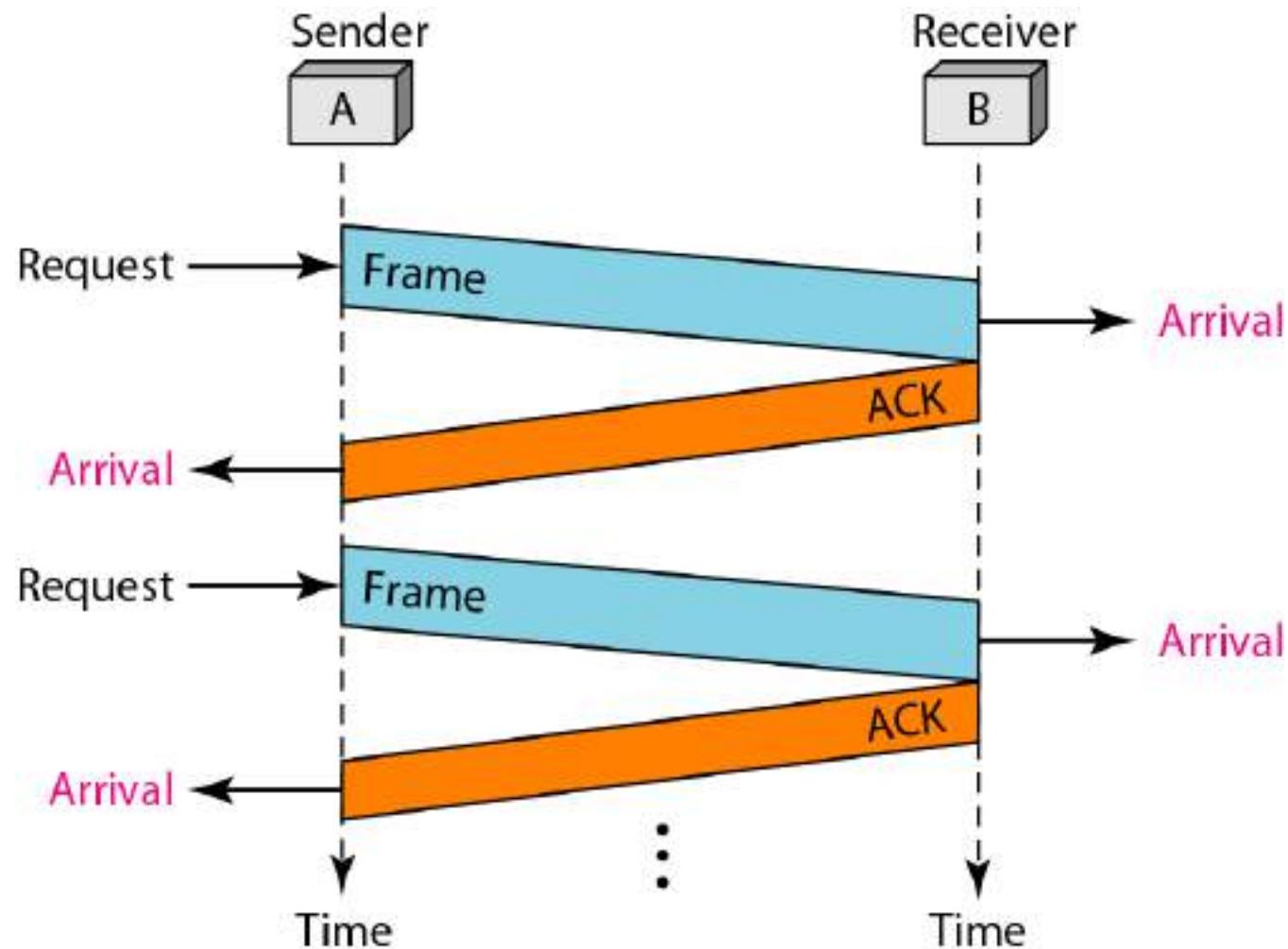


Figure 11.8 Design of Stop-and-Wait Protocol

Figure 11.9 Flow diagram for Example 11.2



### Algorithm 11.3 Sender-site algorithm for Stop-and-Wait Protocol

```
1 while(true)                                //Repeat forever
2 canSend = true                            //Allow the first frame to go
3 {
4     WaitForEvent();                      // Sleep until an event occurs
5     if(Event(RequestToSend) AND canSend)
6     {
7         GetData();
8         MakeFrame();
9         SendFrame();                     //Send the data frame
10        canSend = false;                //Cannot send until ACK arrives
11    }
12    WaitForEvent();                      // Sleep until an event occurs
13    if(Event(ArrivalNotification) // An ACK has arrived
14    {
15        ReceiveFrame();                //Receive the ACK frame
16        canSend = true;
17    }
18 }
```

#### Algorithm 11.4 *Receiver-site algorithm for Stop-and-Wait Protocol*

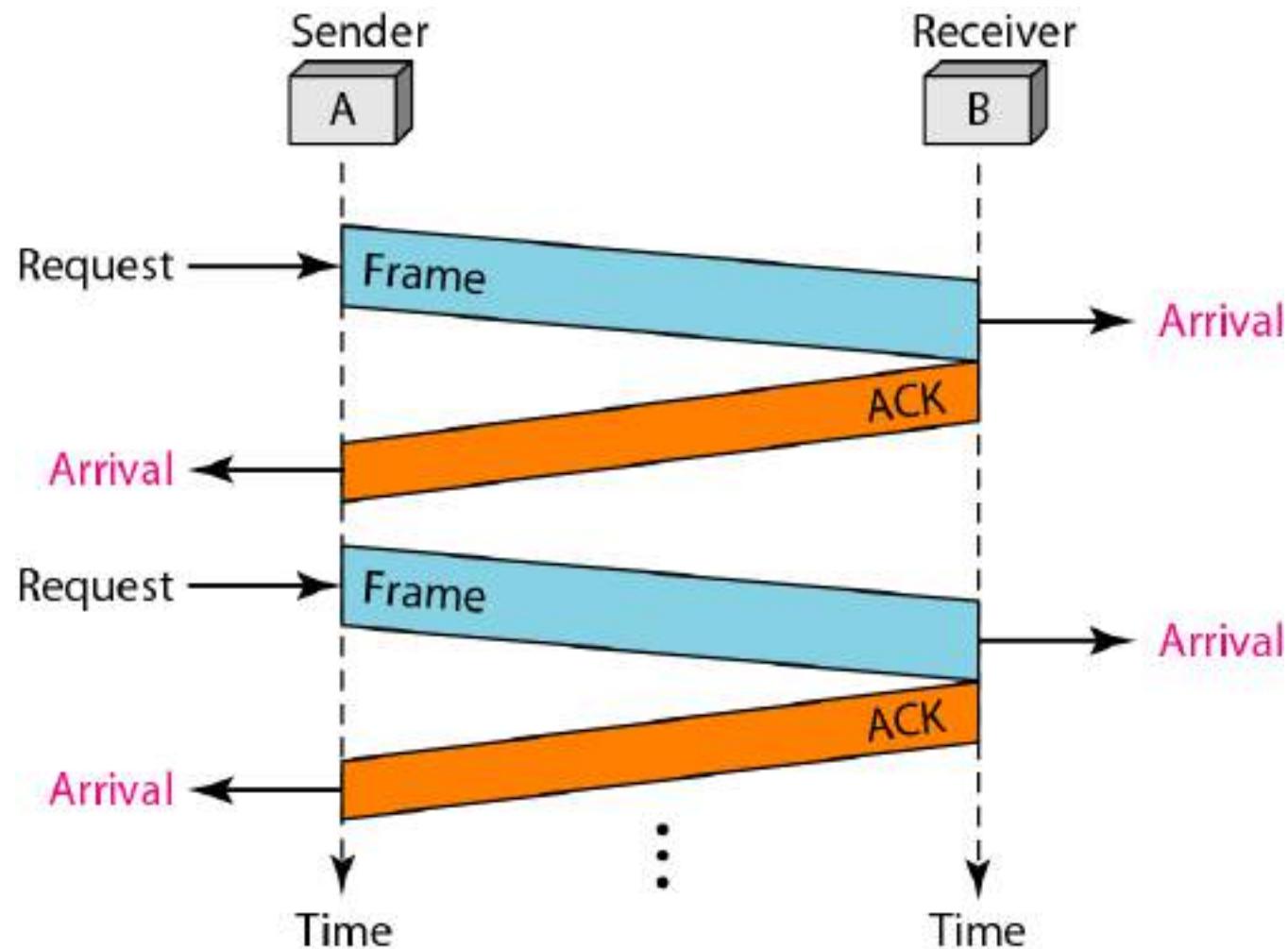
```
1 while(true)                                //Repeat forever
2 {
3     WaitForEvent();                         // Sleep until an event occurs
4     if(Event(ArrivalNotification)) //Data frame arrives
5     {
6         ReceiveFrame();
7         ExtractData();
8         Deliver(data);                  //Deliver data to network layer
9         SendFrame();                  //Send an ACK frame
10    }
11 }
```

*Figure 11.9 shows an example of communication using this protocol.*

*It is still very simple. The sender sends one frame and waits for feedback from the receiver.*

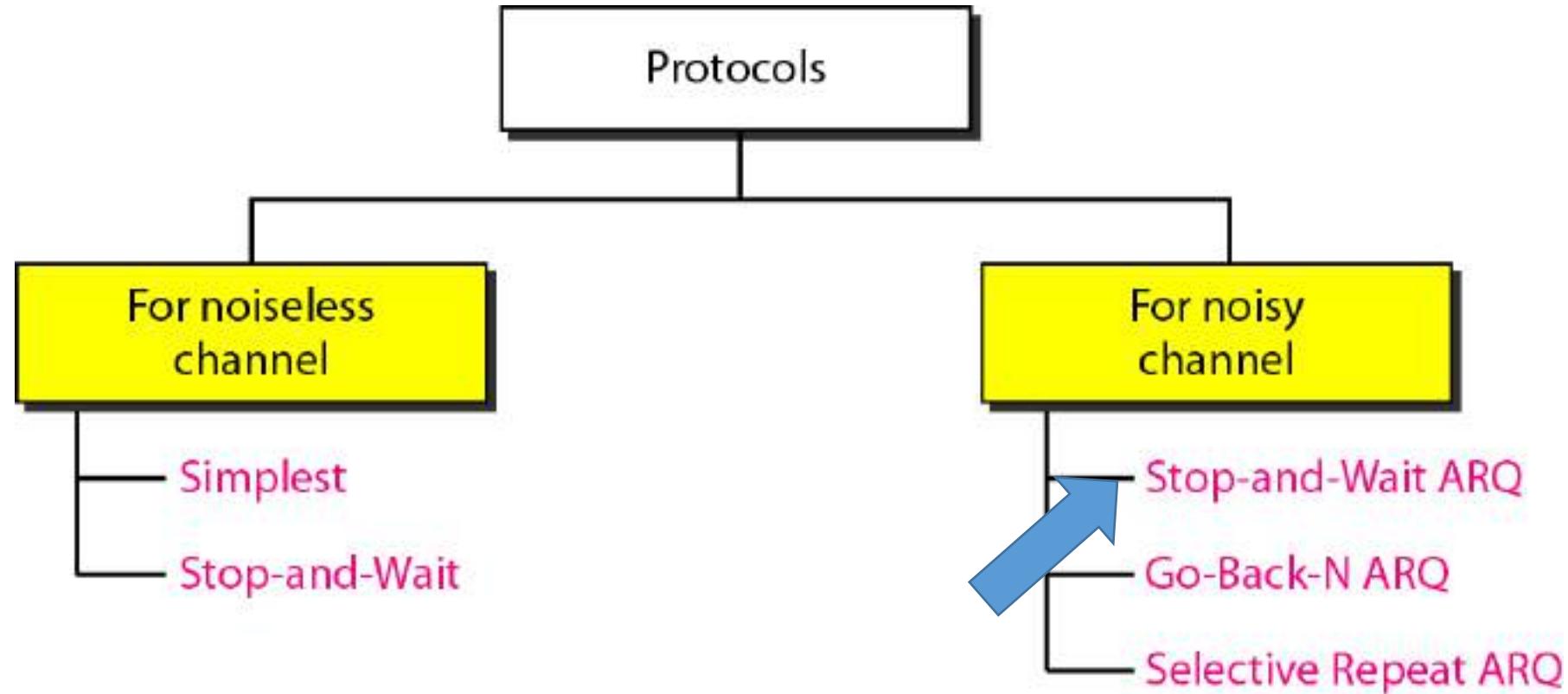
*When the ACK arrives, the sender sends the next frame.*

Figure 11.9 Flow diagram for Example 11.2



- **In Summary**
  - Considers flow control
  - Sender sends one frame
  - Stops until it receives confirmation from the receiver (ACK)
  - After receiving confirmation, sends the next frame

**Figure 11.5** Taxonomy of protocols discussed in this chapter



*Although the Stop-and-Wait Protocol gives us an idea of how to add flow control to its predecessor, noiseless channels are nonexistent. We discuss three protocols in this section that use error control.*

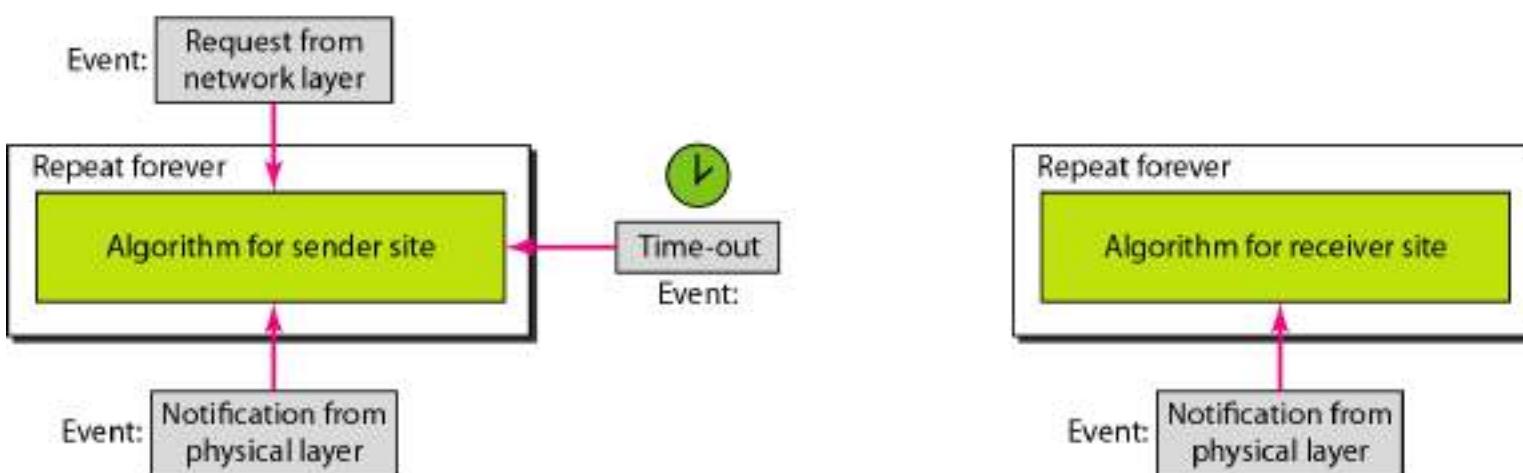
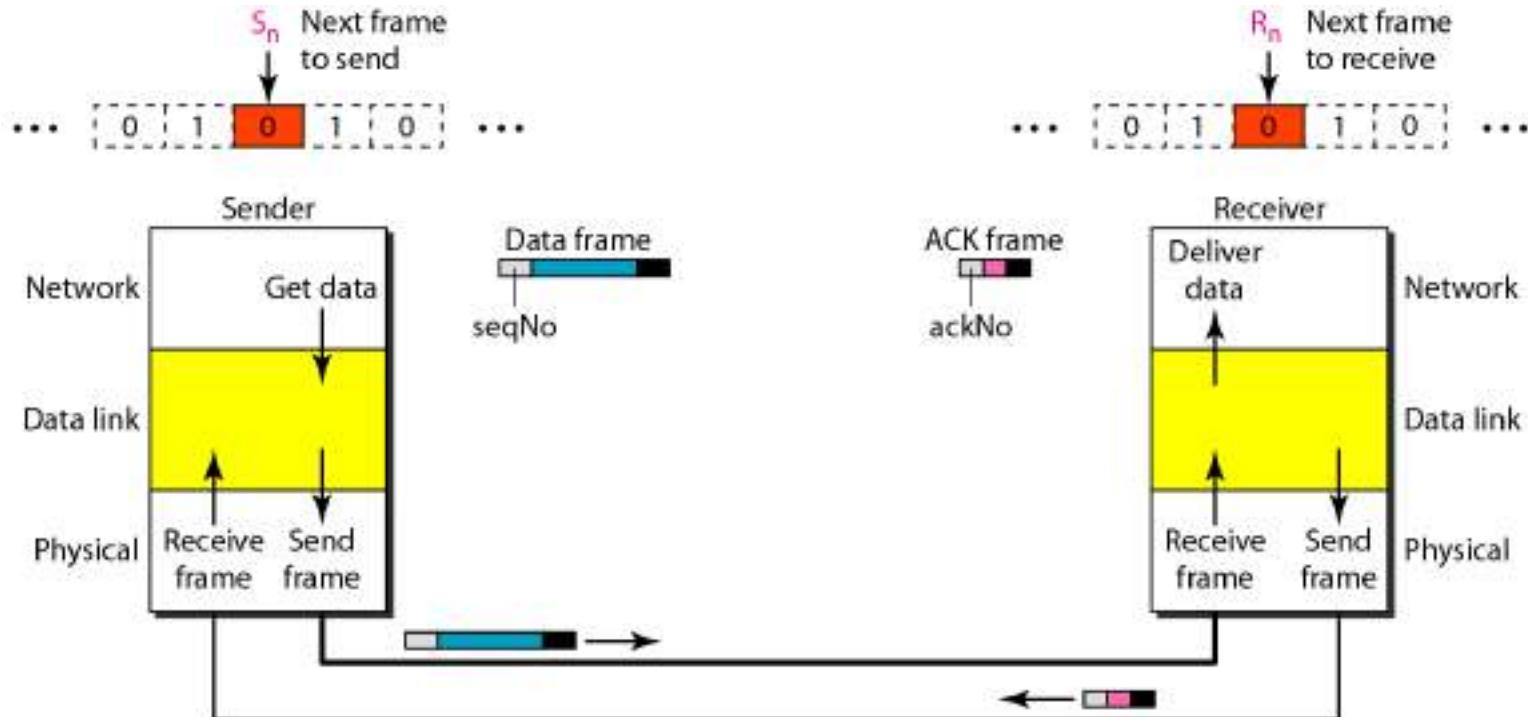
### Topics discussed in this section:

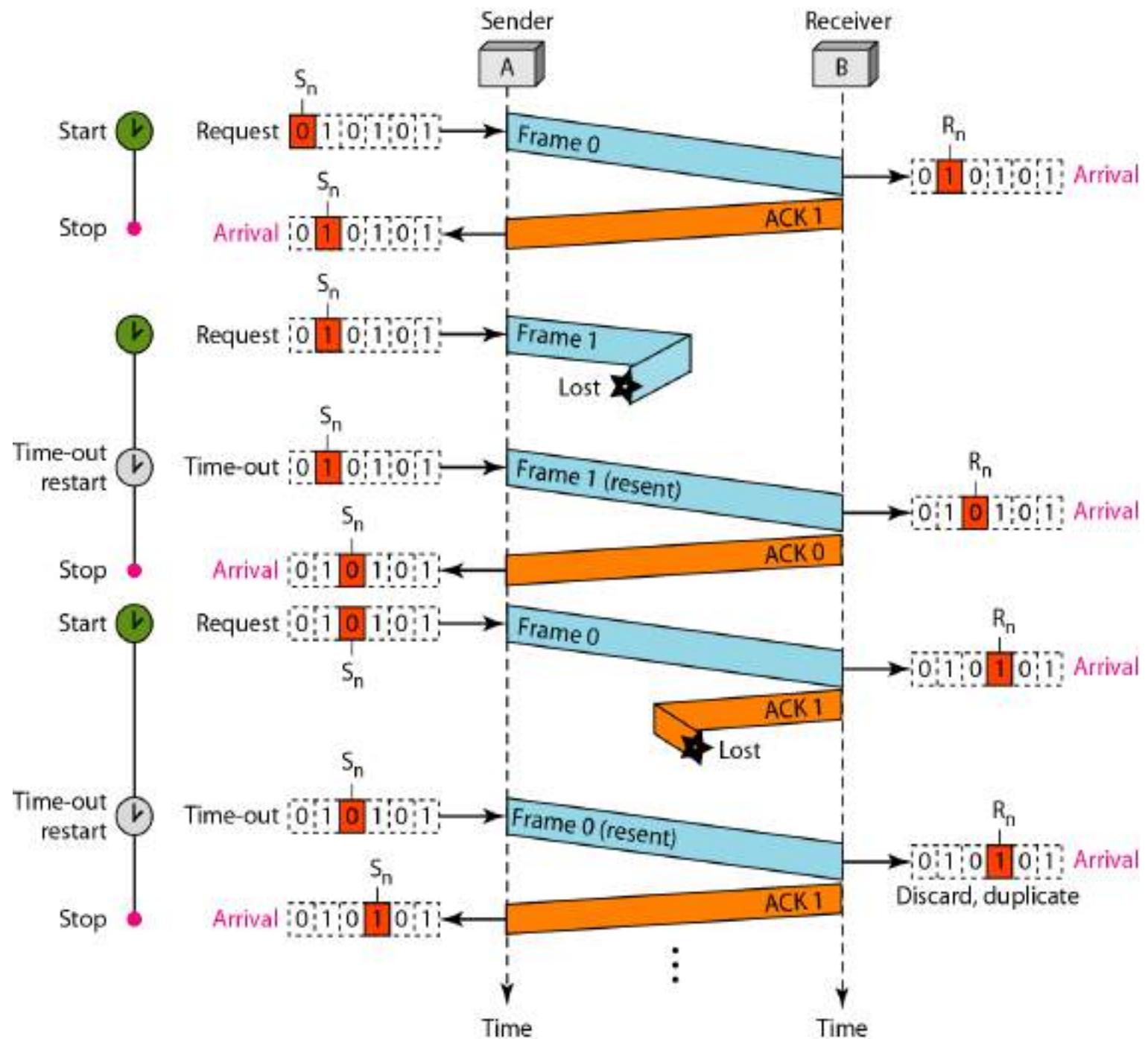
Stop-and-Wait Automatic Repeat Request

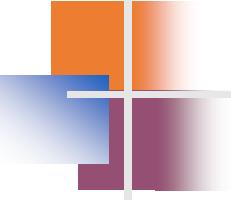
Go-Back-N Automatic Repeat Request

Selective Repeat Automatic Repeat Request

# Stop and wait Automatic Repeat Request (ARQ)





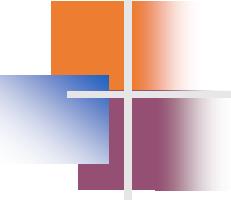


## *Note*

---

Error correction in Stop-and-Wait ARQ is done by keeping a copy of the sent frame and retransmitting of the frame when the timer expires.

---



## *Note*

---

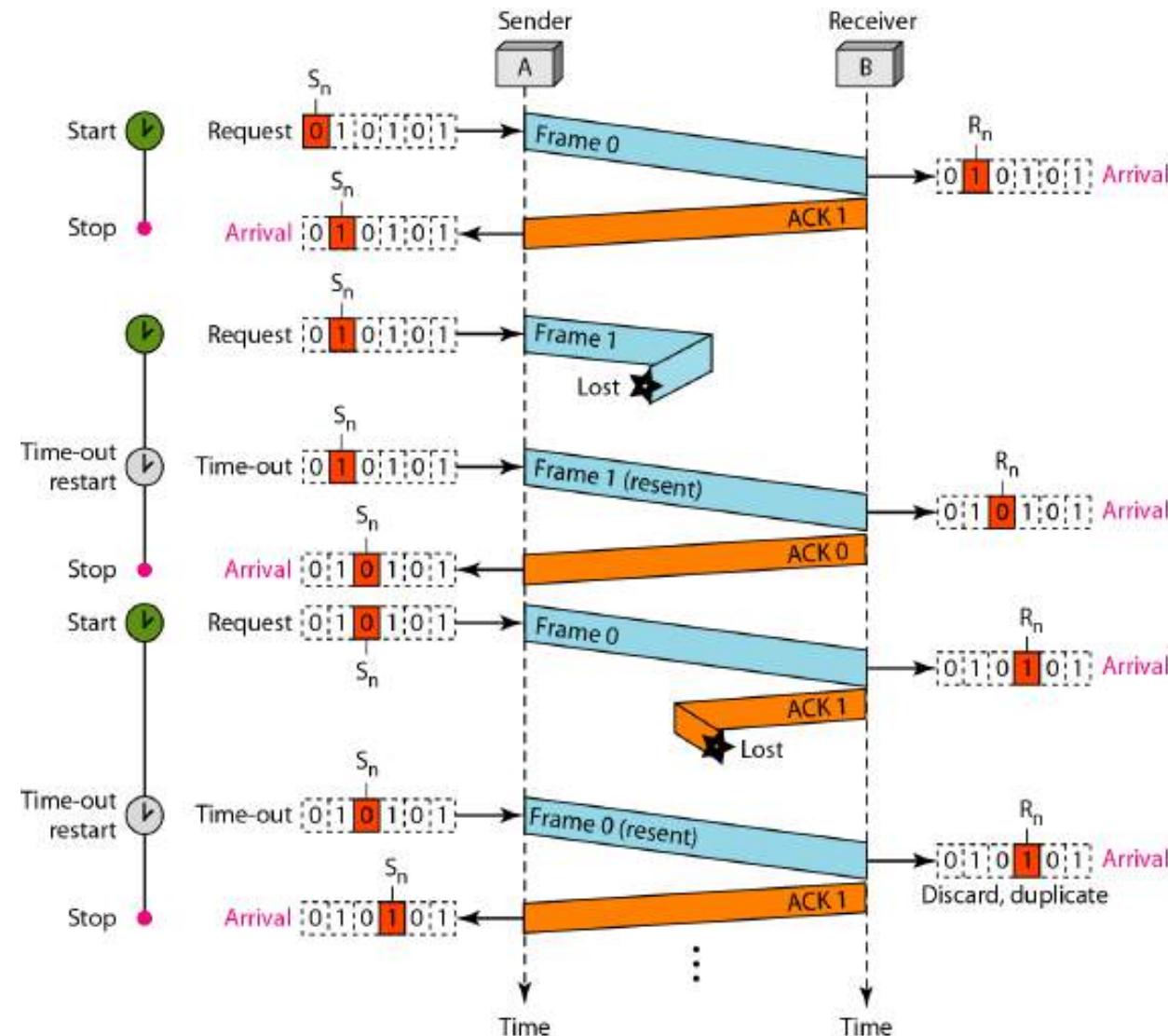
In Stop-and-Wait ARQ, we use sequence numbers to number the frames.

The sequence numbers are based on modulo-2 arithmetic.

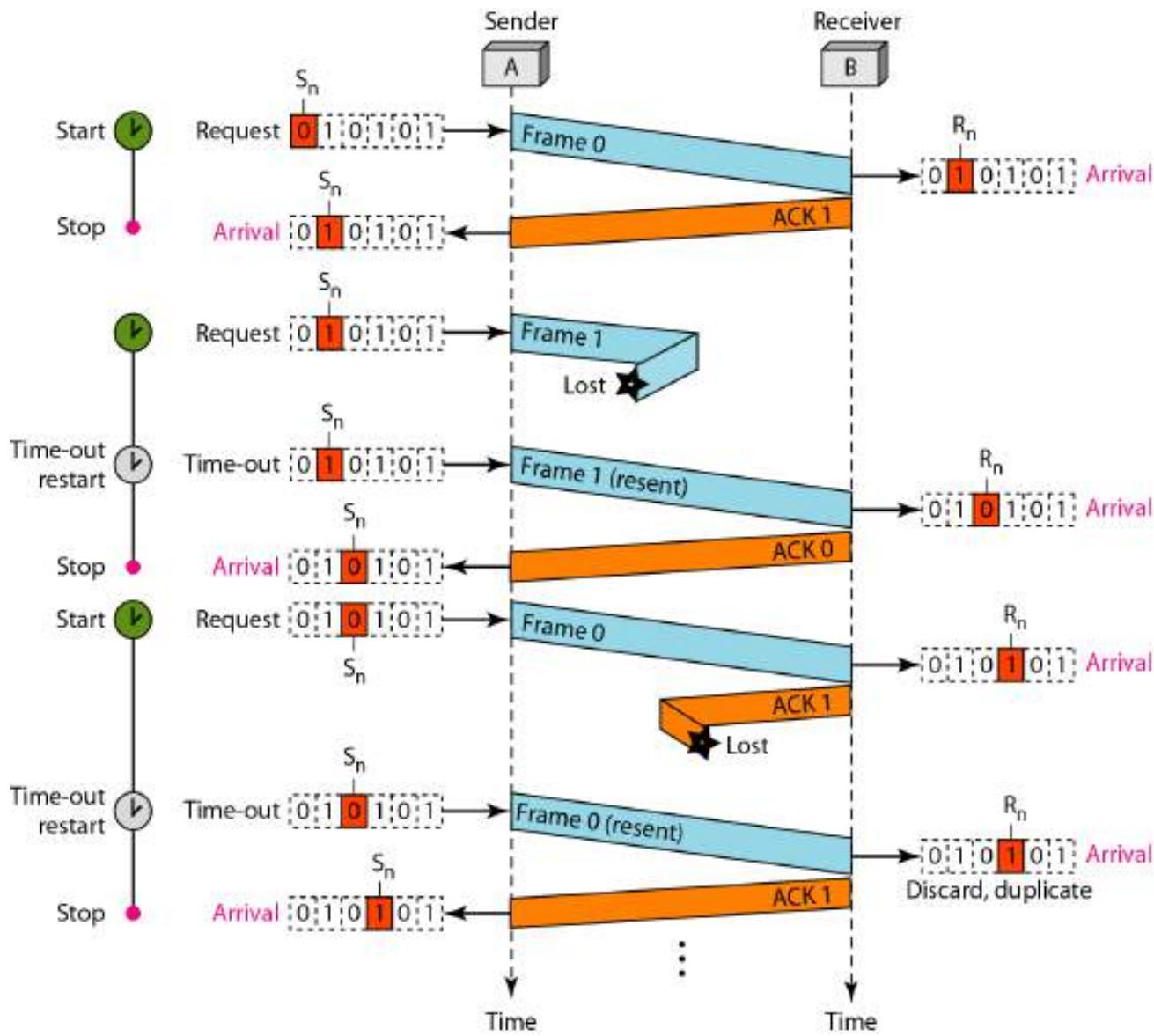
---

# Sequence number

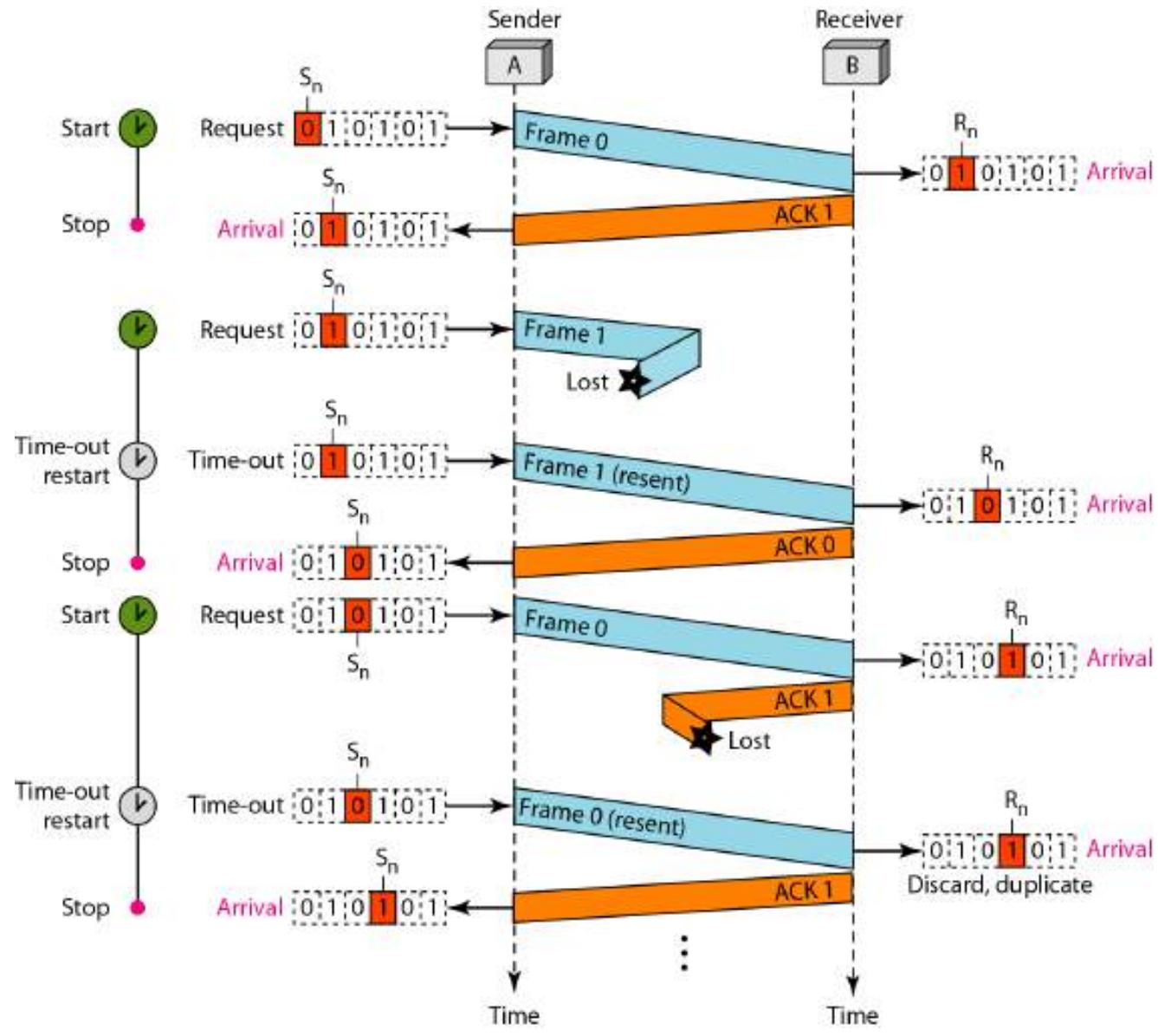
- Let us assume that  $x$  as a sequence number; another frame sequence number is  $x+1$ .
- There is no need for  $x+2$ .
- OK lets consider
  - The frame arrives
  - the receiver sends an acknowledgement.
  - The acknowledgement arrives at the sender site; causing the sender to send the next frame numbered  $x+1$ .



- So there is no need of  $x+2$ .
- So we can let  $x=0$  and  $x+1=1$ .
- This means sequence is 0, 1, 0, 1, 0 and so on.
  
- In Stop-and Wait ARQ, we use sequence numbers to number the frames. The sequence numbers are based on modulo-2 arithmetic.



- **Acknowledgement Numbers:**
  - If frame 0 has arrived safe and sound, the receiver sends an ACK frame with acknowledgement 1 (frame 1 is expected next.)
  - If frame 1 is arrived safe and sound, the receiver sends an ACK frame with acknowledgement 0 (meaning frame 0 is expected)



- Status of the Sender
  - **Ready state :**
    - If **packet** waits from the network layer, the sender creates the frame, saves the copy of the frame, start the timer and sends the frame. Sender moves to blocking states
  - **Blocking State :** in **three** events
    - If a time-out occurs, the sender resends the saved copy of the frame and restarts the timer.
    - If a corrupted **ACK** arrives, it is discarded.
    - If an error-free ACK arrives, the sender stops the timer and discards the saved copy of the frame.
  - **Receiver :** Two events may occur
    - If an error-free frame arrives, the message in the frame is delivered to the network layer and an ACK is sent.
    - If a corrupted frame arrives, the frame is discarded.

### Algorithm 11.5 Sender-site algorithm for Stop-and-Wait ARQ

```
1 Sn = 0;                                // Frame 0 should be sent first
2 canSend = true;                           // Allow the first request to go
3 while(true)                               // Repeat forever
4 {
5   WaitForEvent();                         // Sleep until an event occurs
6   if(Event(RequestToSend) AND canSend)
7   {
8     GetData();
9     MakeFrame(Sn);                   //The seqNo is Sn
10    StoreFrame(Sn);                  //Keep copy
11    SendFrame(Sn);
12    StartTimer();
13    Sn = Sn + 1;
14    canSend = false;
15  }
16  WaitForEvent();                         // Sleep
```

(continued)

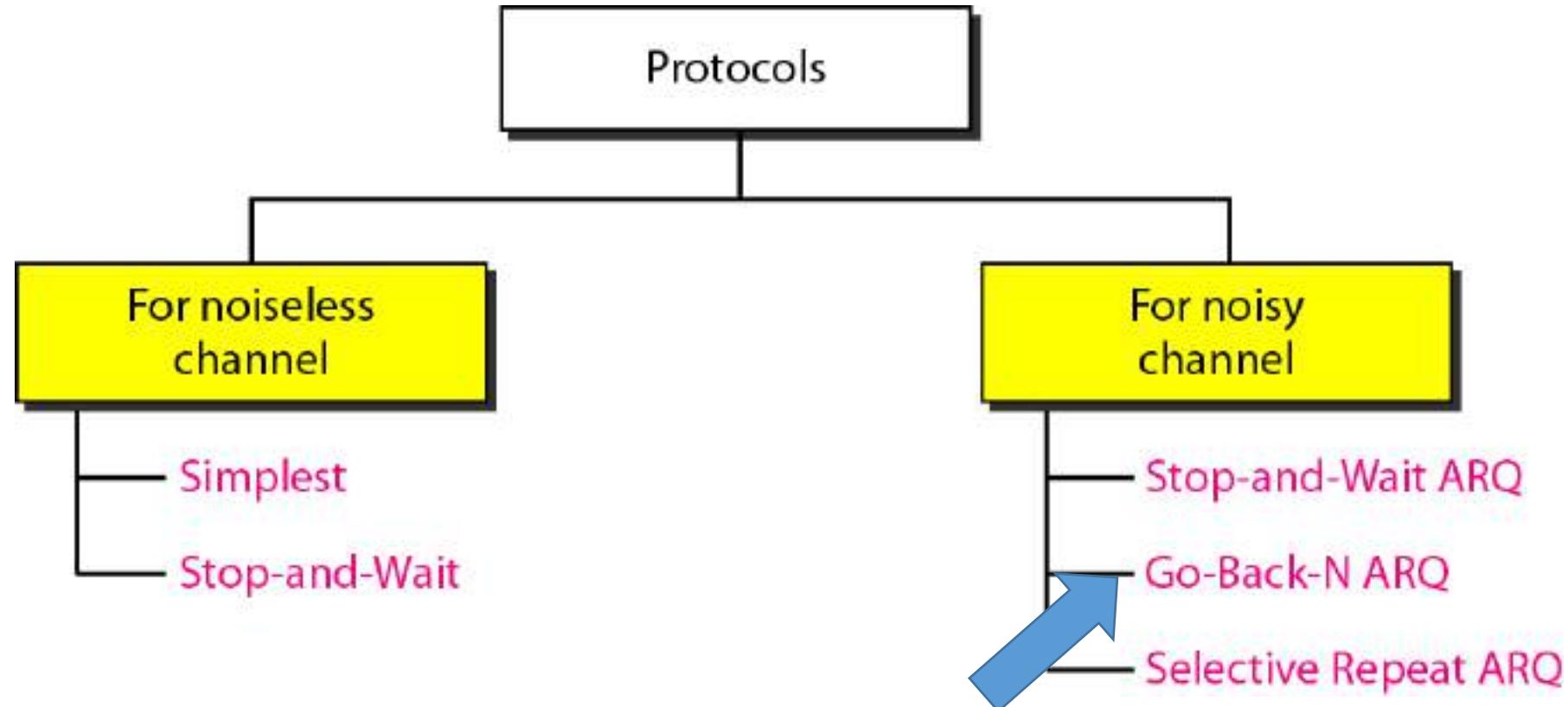
## Algorithm 11.5 Sender-site algorithm for Stop-and-Wait ARQ

```
17  if(Event(ArrivalNotification)          // An ACK has arrived
18  {
19      ReceiveFrame(ackNo);           //Receive the ACK frame
20      if(not corrupted AND ackNo == Sn) //Valid ACK
21      {
22          StopTimer();
23          PurgeFrame(Sn-1);          //Copy is not needed
24          canSend = true;
25      }
26  }
27
28  if(Event(TimeOut)                  // The timer expired
29  {
30      StartTimer();
31      ResendFrame(Sn-1);          //Resend a copy check
32  }
33 }
```

## Algorithm 11.6 Receiver-site algorithm for Stop-and-Wait ARQ Protocol

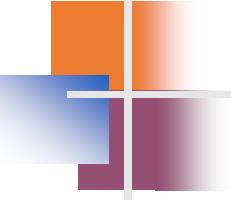
```
1 Rn = 0;                                // Frame 0 expected to arrive first
2 while(true)
3 {
4     WaitForEvent();                      // Sleep until an event occurs
5     if(Event(ArrivalNotification))    //Data frame arrives
6     {
7         ReceiveFrame();
8         if(corrupted(frame));
9             sleep();
10        if(seqNo == Rn)           //Valid data frame
11        {
12            ExtractData();
13            DeliverData();          //Deliver data
14            Rn = Rn + 1;
15        }
16        SendFrame(Rn);          //Send an ACK
17    }
18 }
```

**Figure 11.5** Taxonomy of protocols discussed in this chapter



# Go back-N Automatic Repeat Request

- To improve the efficiency of transmission (filling the pipe), the **multiple frames** must be in transition while waiting for acknowledgement.
- In other words, we need to let more than one frame be outstanding to keep the **channel busy** while the sender is waiting for acknowledgement.
- In **Go back-N Automatic Repeat Request** protocol we can send several frames before receiving acknowledgement.



## *Note*

---

**In the Go-Back-N Protocol, the sequence numbers are modulo  $2^m$ , where m is the size of the sequence number field in bits.**

# Sequence Number

- Frames from a sending station are numbered sequentially.
- We need to include the sequence number of each frame in the **header**.
- If the header of the frame allows  $m$  bits from the sequence number, the sequence numbers range from **0 to  $2^m-1$** .
  - Eg. If  $m=4$ 
    - Sequence number are from 0 to 15
    - After 0 to 15, numbers are repeated.
  - **In the Go-Back-N Protocol, the sequence numbers are modulo  $2^m$ , where  $m$  is the size of the sequence number field in bits.**

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, ...

In other words, the sequence numbers are modulo- $2^m$ .

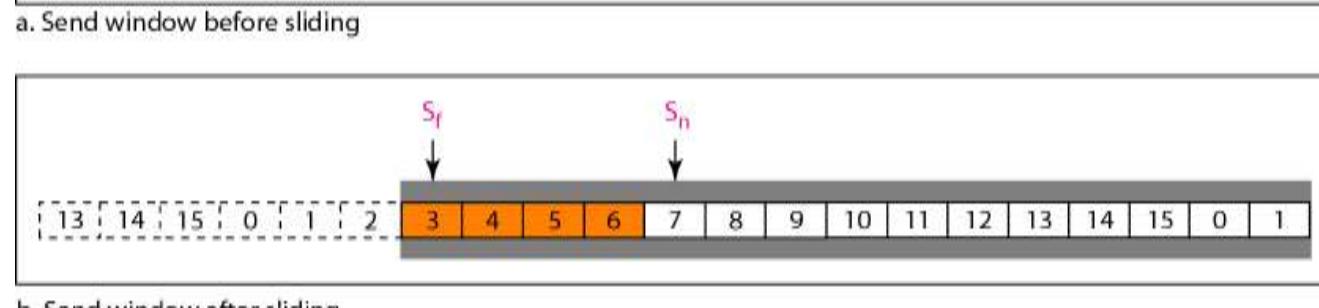
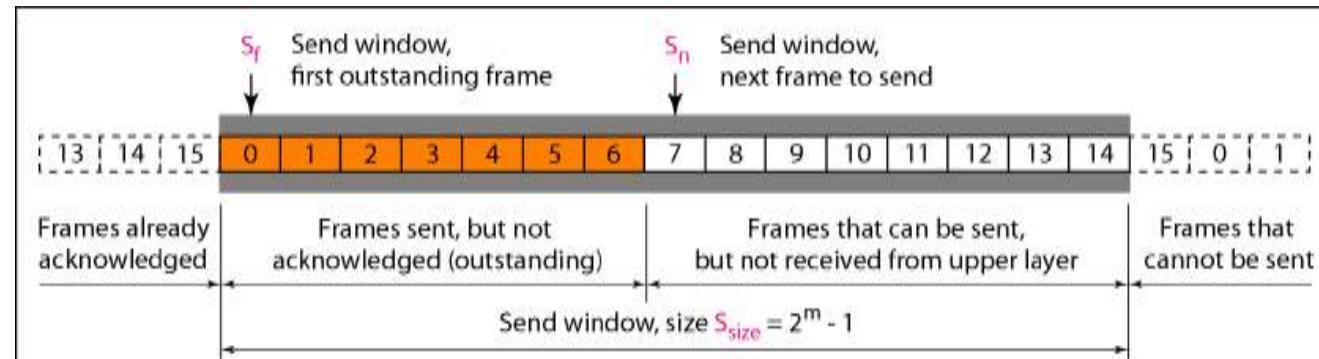
---

In the Go-Back-N Protocol, the sequence numbers are modulo  $2^m$ ,  
where  $m$  is the size of the sequence number field in bits.

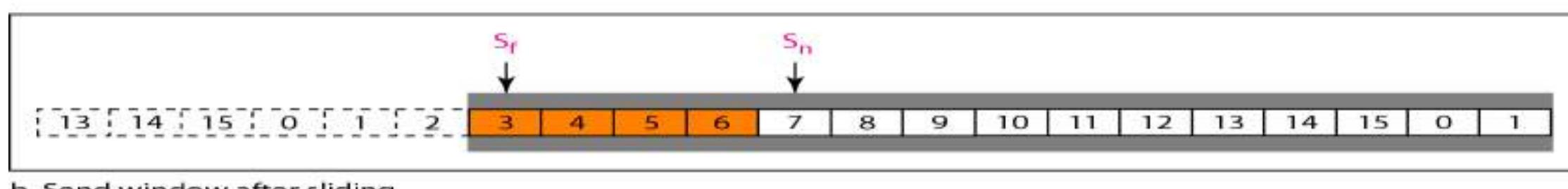
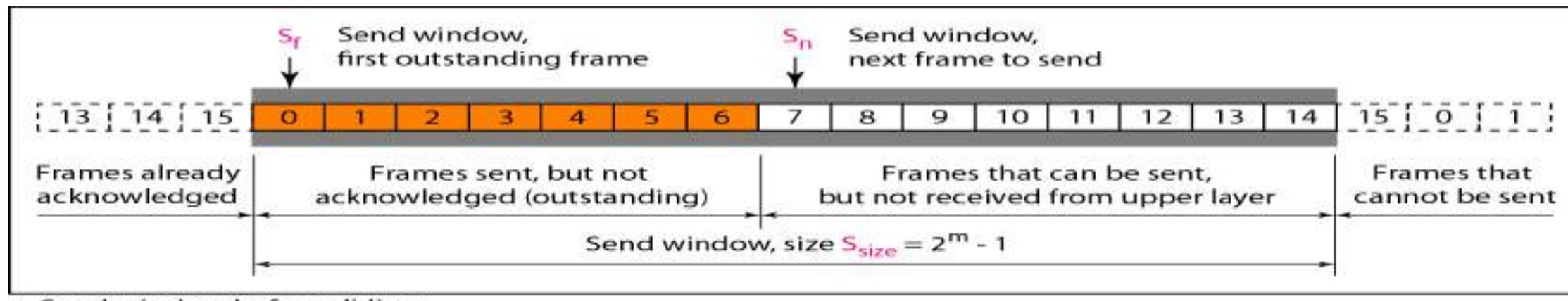
---

# Send Window for Go-Back N ARQ

- It consists of the range of sequence numbers.
- The range which is concerned with sender is called **send sliding window**.
- The range which is concerned with receiver is called **receive sliding window**.
- Maximum size of window is  $2^m - 1$
- In each window position some of these sequence numbers define the frames that have been sent, others define that can be sent.



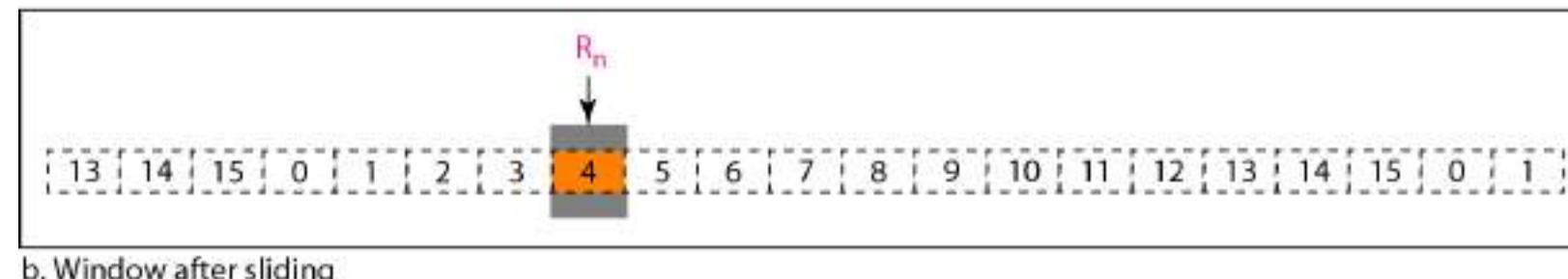
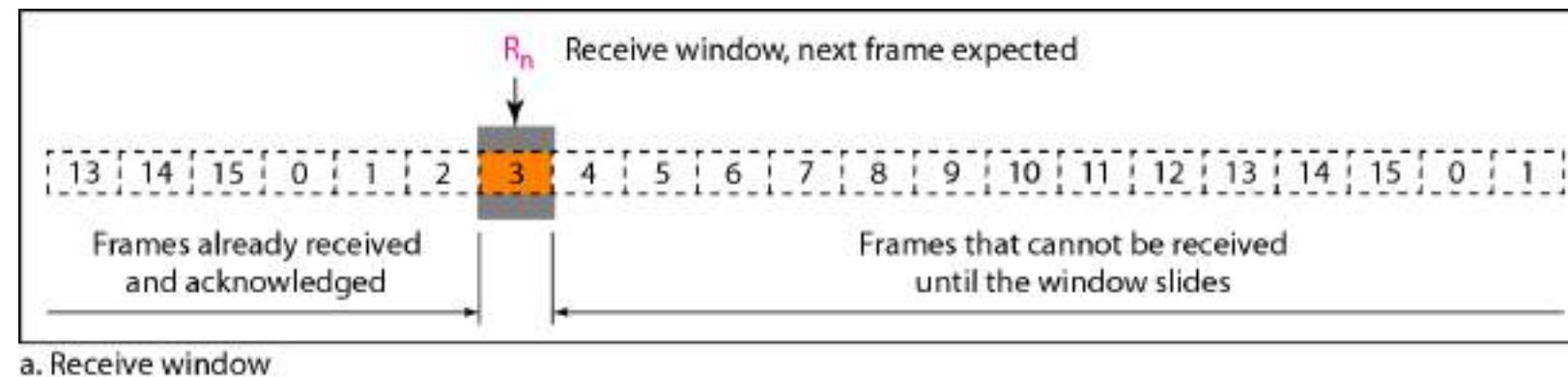
- The window at any time divides the possible sequence numbers into four regions.
- **The first region:**
- **Second region :**
- **Third region :**
- **Fourth region:**



- **Receiver:**

- The receiver window makes sure that the correct data frames are reviewed and correct acknowledgement are sent.
- The size of the receive window is always 1.
- The receiver is always looking for the arrival of a specific frame.
- Any frame arriving out of order is discarded and needs to be resent.

**The receive window is an abstract concept defining an imaginary box of size 1 with one single variable  $R_n$ . The window slides when a correct frame has arrived; sliding occurs one slot at time**



- **Acknowledgment:**
  - The receiver sends a positive acknowledgment if a frame has arrived safe and sound and in order.
  - If a frame is damaged or a is received out of order, the receiver is silent and will **discard all subsequent frames** until it receives the one it is expecting.
  - Again, the sender to go back and **resend all frames**, beginning with the one with the expired timer.
  - It can send one **cumulative acknowledgment** for several frames.

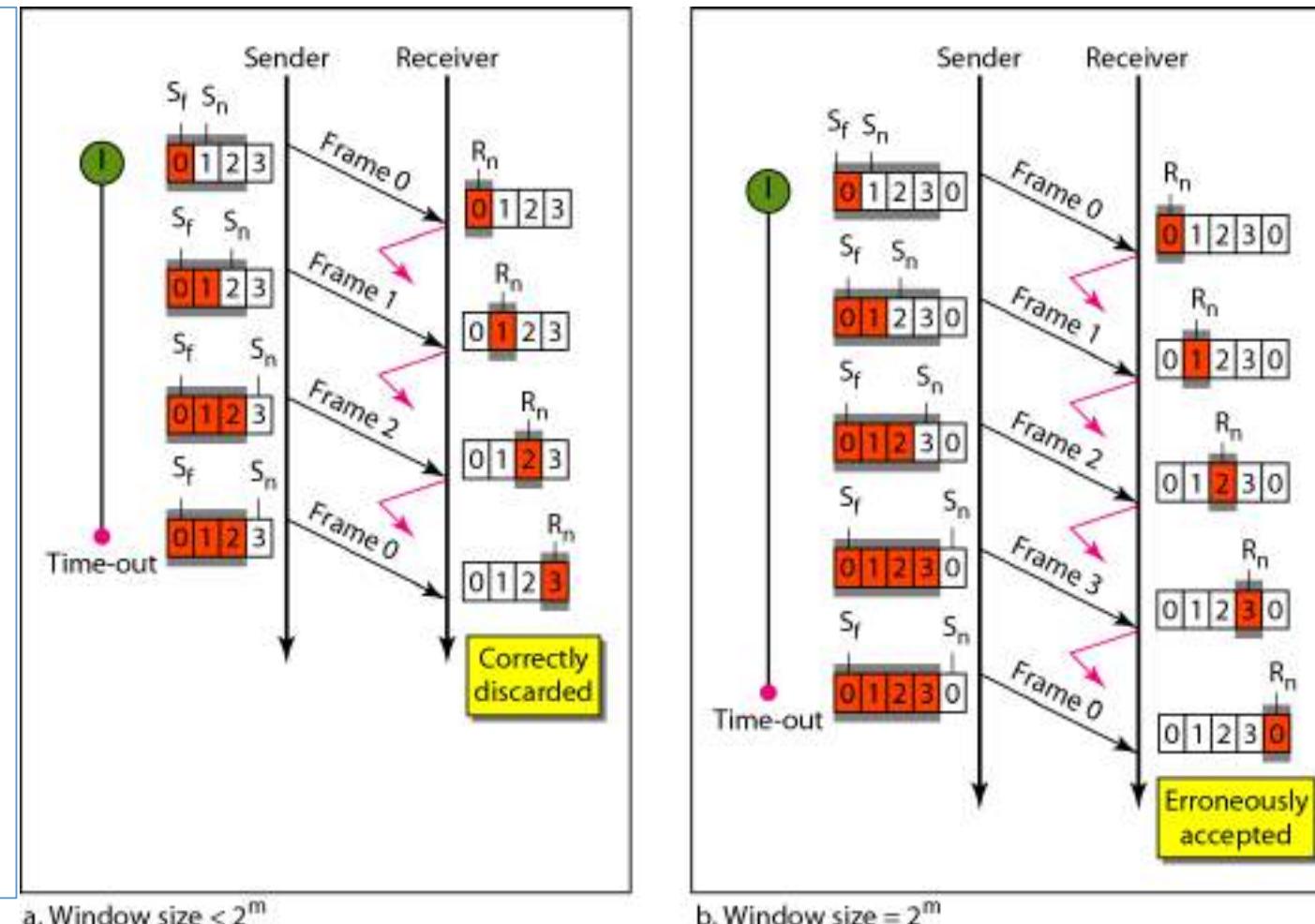
# *Resending a Frame*

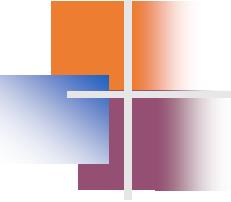
- When the timer expires, the sender resends all outstanding frames.
- For example, suppose the sender has already sent frame 6, but the timer for frame 3 expires.
- This means that frame 3 has not been acknowledged; the sender goes back and sends frames 3, 4, 5, and 6 again.
- That is why the protocol is called *Go-Back-N ARQ*.

Figure 11.15 Window size for Go-Back-N ARQ

## Send Window Size

- We can now show why the size of the send window must be less than  $2^m$ . As an example,
- we choose  $m = 2$ , which means the size of the window can be  $2^m - 1$ , or 3.
- Figure 11.15 compares a window size of 3 against a window size of 4.





## *Note*

---

**In Go-Back-N ARQ, the size of the send window must be less than  $2^m$ ;  
the size of the receiver window is always 1.**

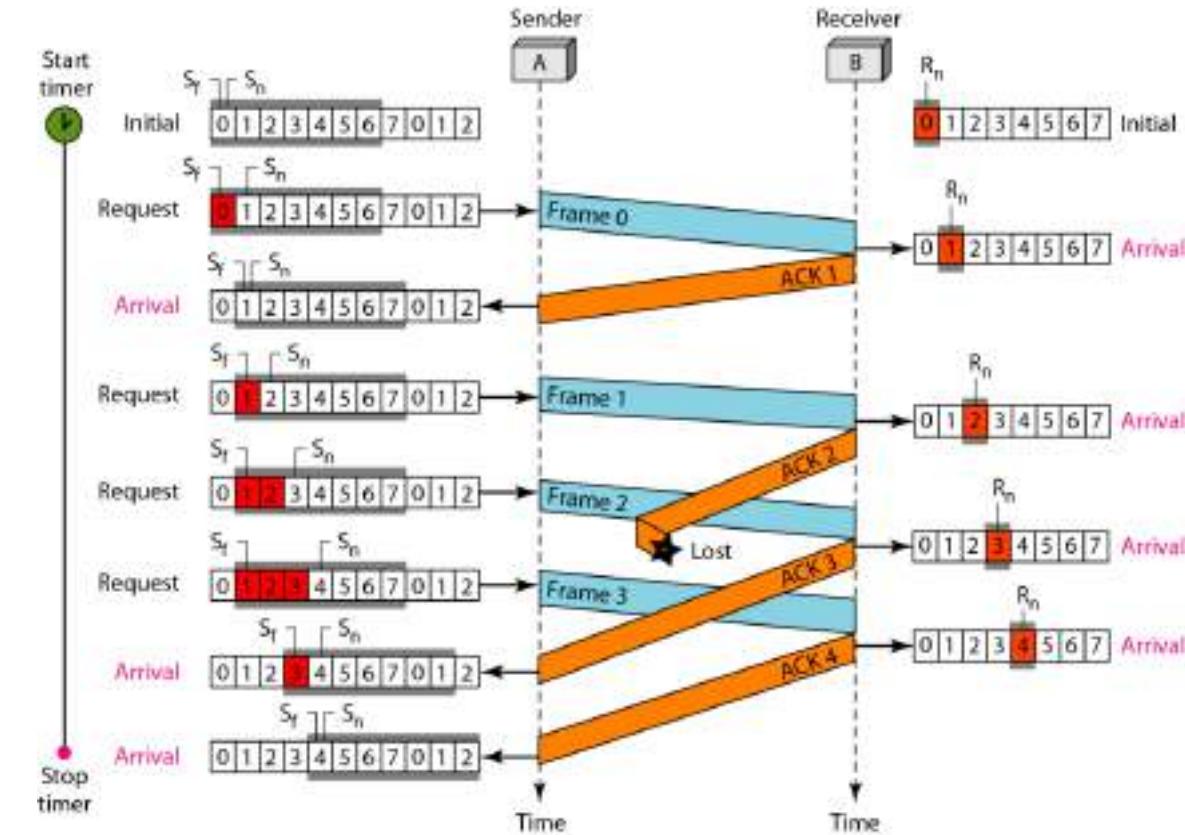
---

**Figure 11.16** Flow diagram for Example 11.6 Go-Back-N.

This is an example of a case where the forward channel is reliable, but the reverse is not.

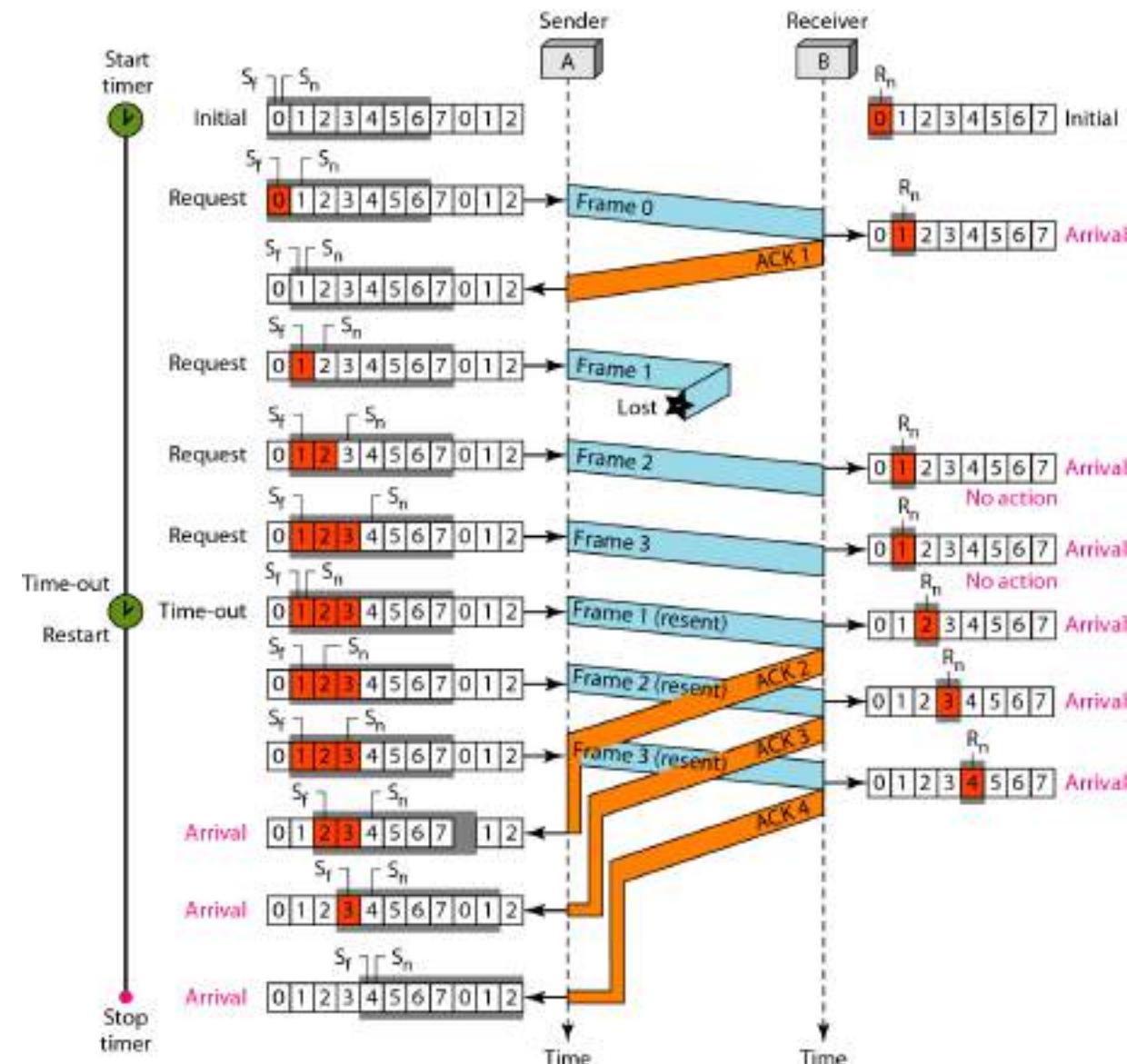
No data frames are lost, but some ACKs are delayed and one is lost.

The example also shows how **cumulative acknowledgments** can help if acknowledgments are delayed or lost.



There is no time-out event here because all outstanding frames are acknowledged before the timer expires. Note that although ACK 2 is lost, ACK 3 serves as both ACK 2 and ACK 3.

- Figure 11.17 Figure 11.17 shows what happens when a frame is lost.



## Algorithm 11.7 Go-Back-N sender algorithm

```
1 Sw = 2m - 1;  
2 Sf = 0;  
3 Sn = 0;  
4  
5 while (true) //Repeat forever  
6 {  
7     WaitForEvent();  
8     if(Event(RequestToSend)) //A packet to send  
9     {  
10         if(Sn-Sf >= Sw) //If window is full  
11             Sleep();  
12         GetData();  
13         MakeFrame(Sn);  
14         StoreFrame(Sn);  
15         SendFrame(Sn);  
16         Sn = Sn + 1;  
17         if(timer not running)  
18             StartTimer();  
19     }  
20 }
```

(continued)

## Algorithm 11.7 Go-Back-N sender algorithm

```

21  if(Event(ArrivalNotification)) //ACK arrives
22  {
23      Receive(ACK);
24      if(corrupted(ACK))
25          Sleep();
26      if((ackNo>Sf)&&(ackNo<=Sn)) //If a valid ACK
27      While(Sf <= ackNo)
28      {
29          PurgeFrame(Sf);
30          Sf = Sf + 1;
31      }
32      StopTimer();
33  }
34
35  if(Event(TimeOut)) //The timer expires
36  {
37      StartTimer();
38      Temp = Sf;
39      while(Temp < Sn);
40      {
41          SendFrame(Sf);
42          Sf = Sf + 1;
43      }
44  }
45 }
```

## Algorithm 11.8 Go-Back-N receiver algorithm

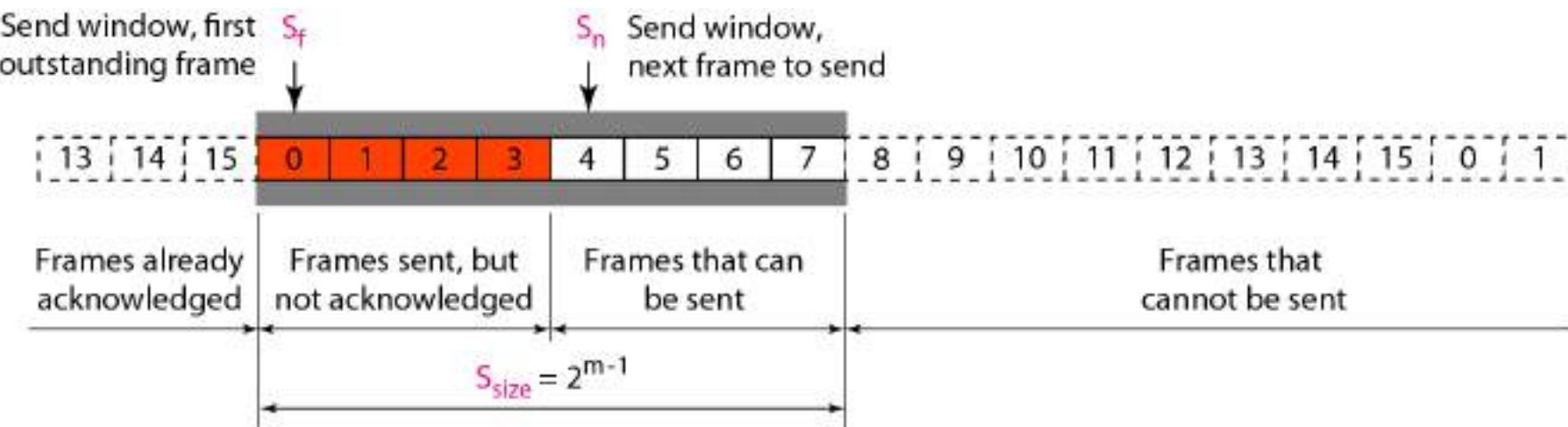
```
1 Rn = 0;  
2  
3 while (true) //Repeat forever  
4 {  
5     WaitForEvent();  
6  
7     if(Event(ArrivalNotification)) /Data frame arrives  
8     {  
9         Receive(Frame);  
10        if(corrupted(Frame))  
11            Sleep();  
12        if(seqNo == Rn) //If expected frame  
13        {  
14            DeliverData(); //Deliver data  
15            Rn = Rn + 1; //Slide window  
16            SendACK(Rn);  
17        }  
18    }  
19 }
```

# Selective Repeat Automatic Repeat Request

- For Noisy links, there is a mechanism that does not resend **N** frames when just one frame is damaged; only the damaged frame are resent.
- This mechanism is called **Selective Repeat ARQ**.

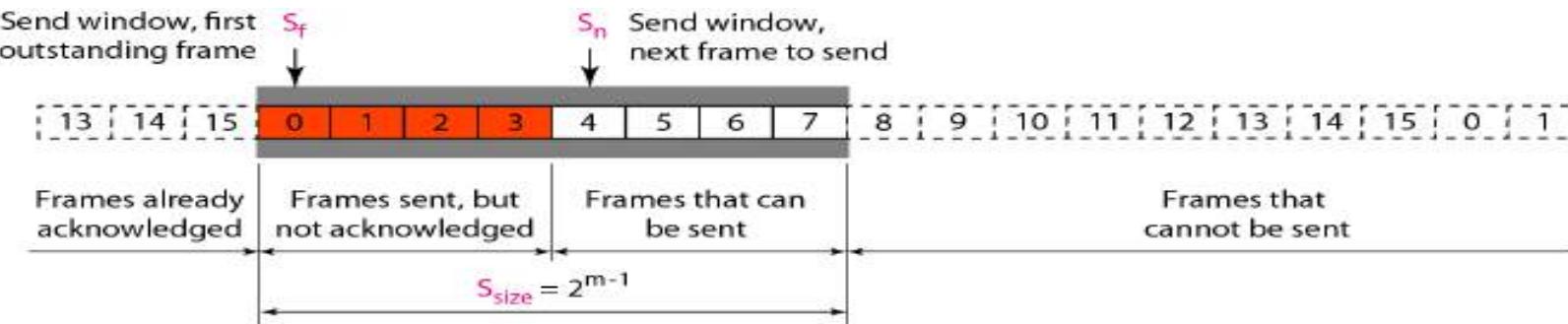
# Windows

- The Selective Repeat Protocol uses two windows :
  - send window
  - receive window
- Send Window size is  $2^{m-1}$  (smaller than of Go-Back-N)
- Receive window is the same size as the send window.

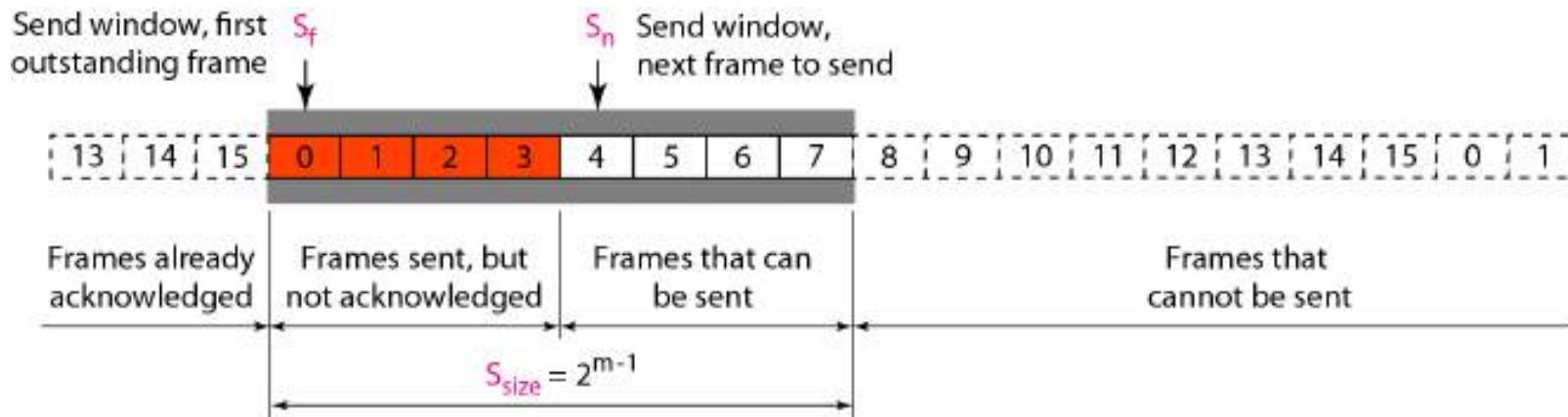


### Send window for Selective Repeat ARQ

- The send window maximum size can be  $2^{m-1}$ .
- For example,  $m=4$ , the sequence numbers go from 0 to 15, but the size of the window is just 8.
- The smaller window size means less efficiency in filling the pipe.



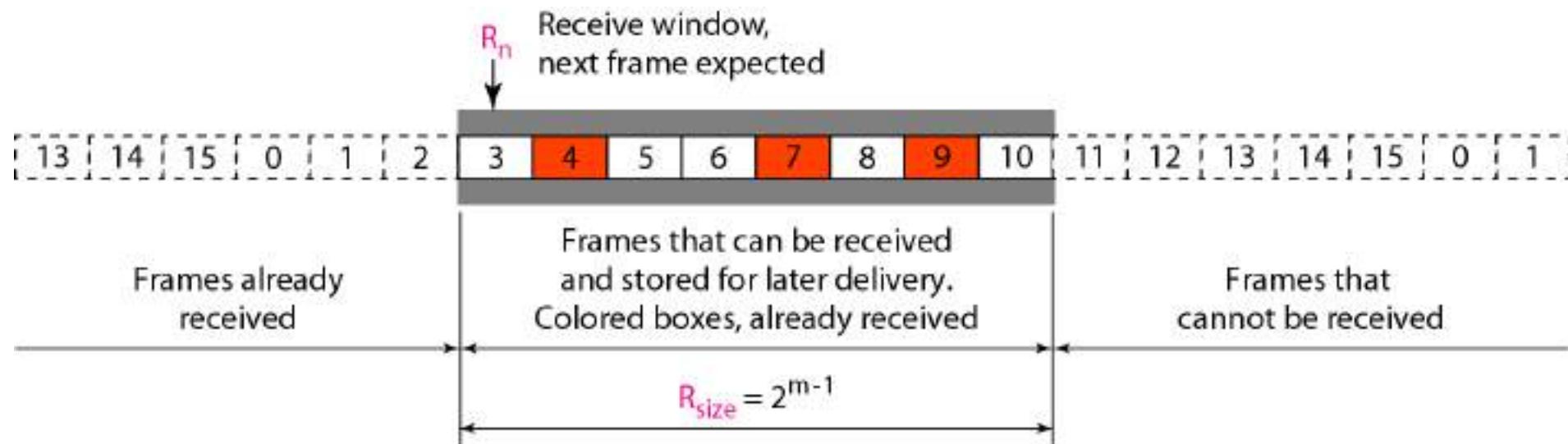
**Send window for Selective Repeat ARQ**



**Receive window for Selective Repeat ARQ**

- The receive window of Selective Repeat is totally different than the one in Go-Back-N
- First the size of the receive window is **the same as the size of the send window ( $2^{m-1}$ )**

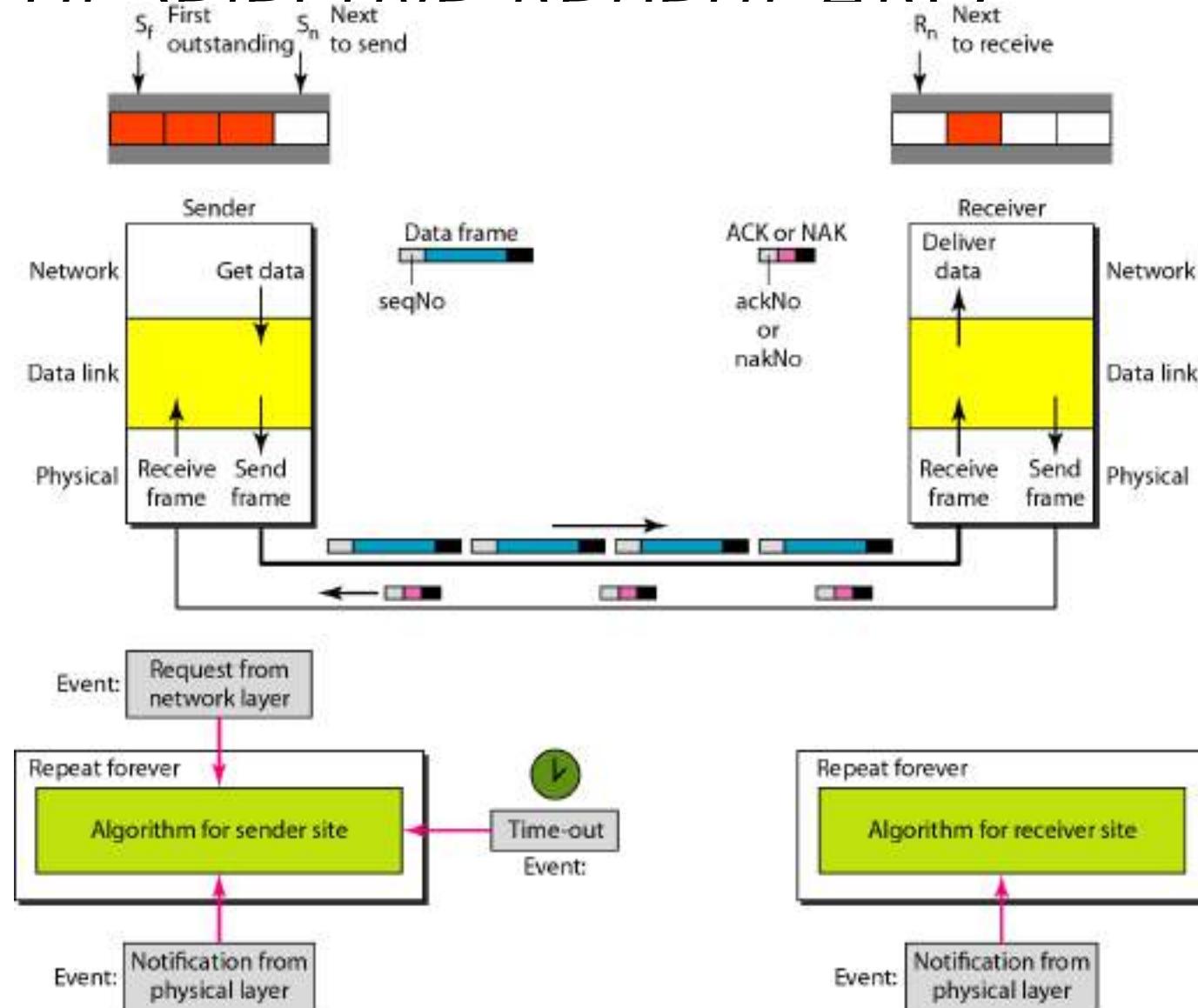
Figure 11.19 Receive window for Selective Repeat ARQ



The Selective Repeat Protocol allows as many frames **as the size of the receive window to arrive out of order** and be kept until there is a set of in-order frames to be delivered to the network layer.

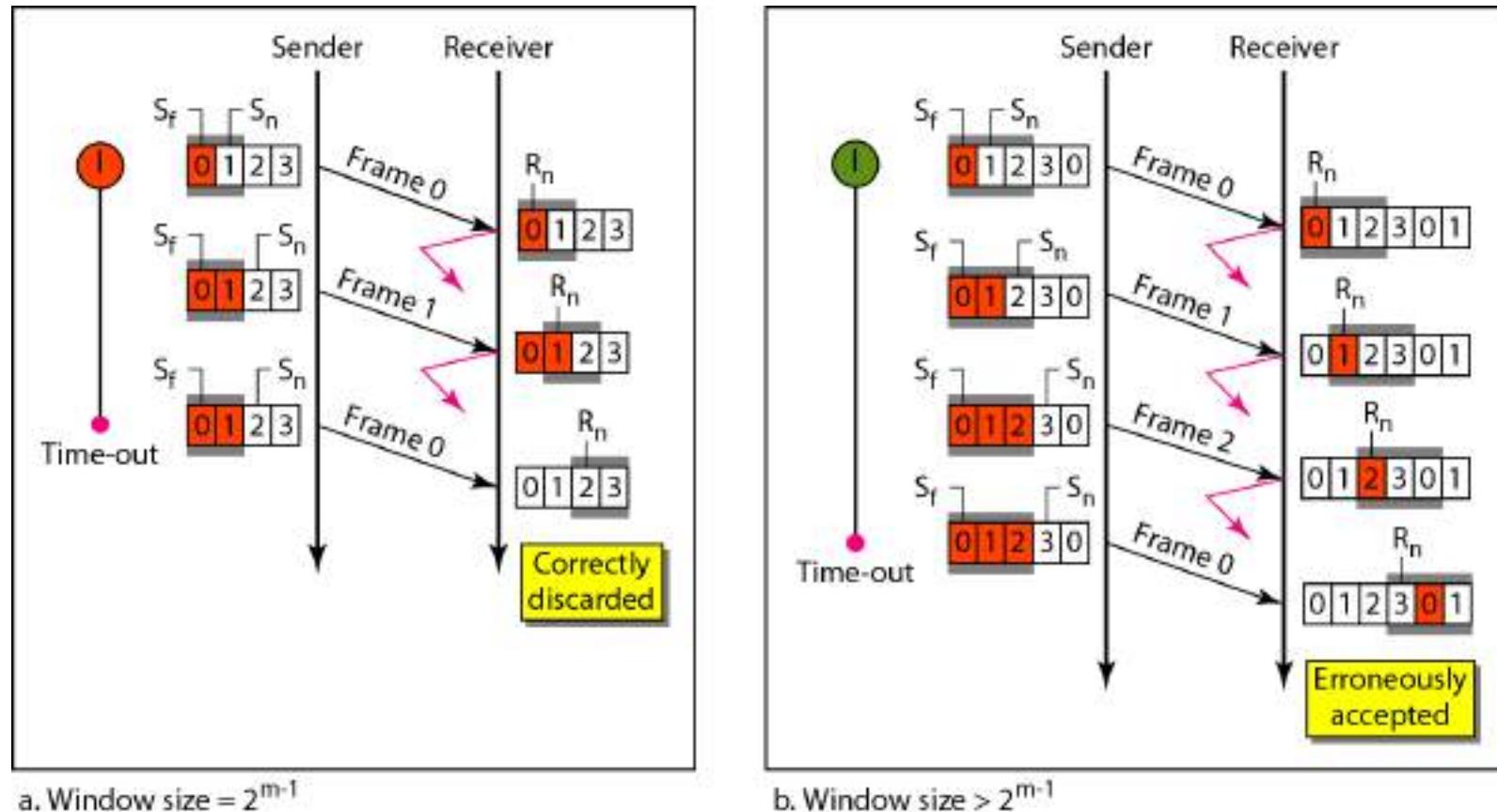
Because the size of the send window and receive window are the same, all the frames in the send frame **can arrive out order** and be stored until they can be delivered.

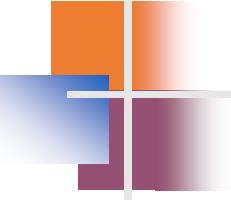
# Design of Selective Repeat ARQ



**Figure 11.21 Selective Repeat ARQ, window size**

- We can now show why the size of the sender and receiver windows must be at most one half of  $2^m$ .
- For an example, we choose  $m = 2$ , which means the size of the window is  $2m/2$ , or 2.
- Figure 11.21 compares a window size of 2 with a window size of 3.





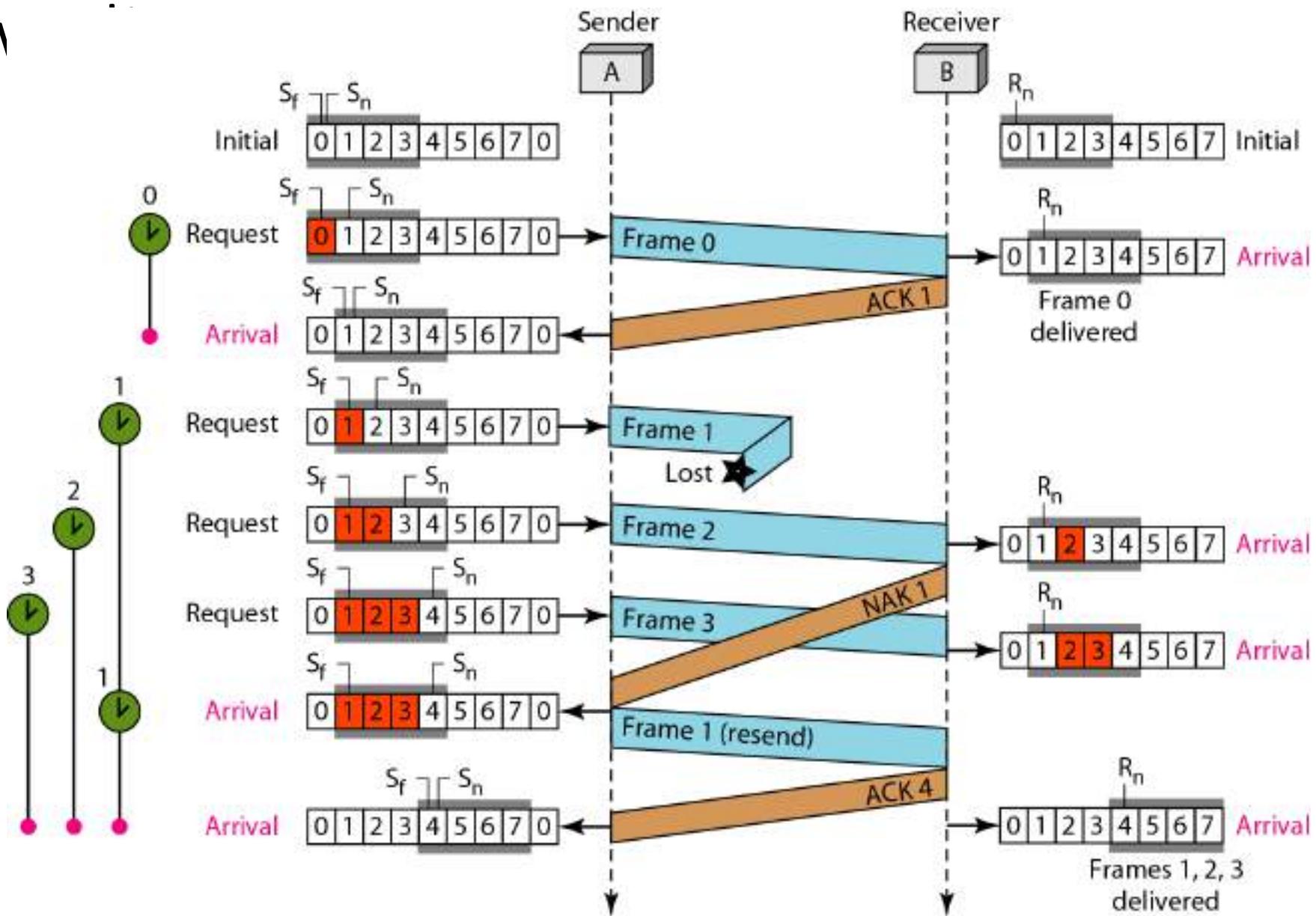
## *Note*

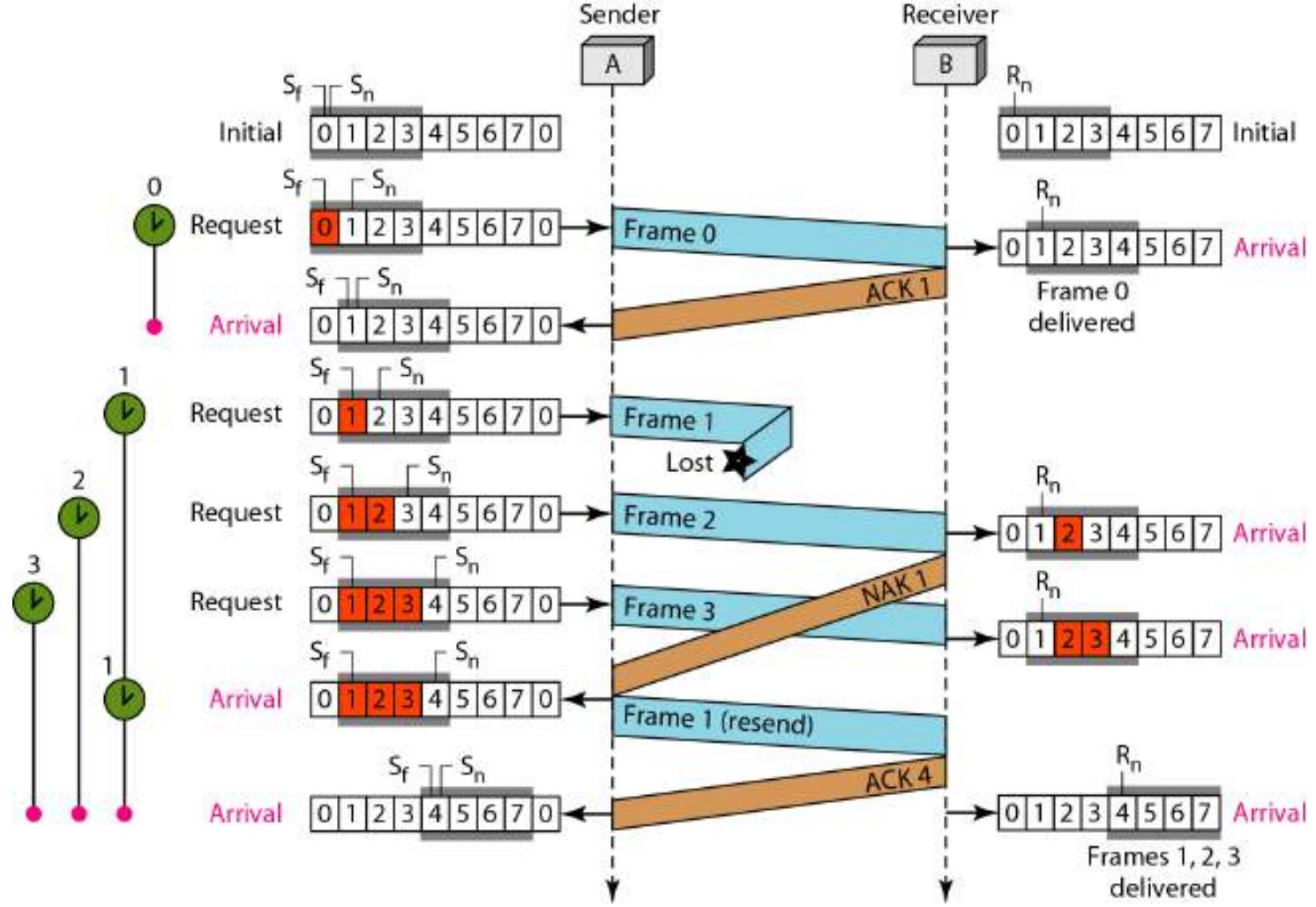
---

In Selective Repeat ARQ, the size of the sender and receiver window must be at most one-half of  $2^m$ .

---

# Flow





- The next point is about the ACKs. Notice that only two ACKs are sent here. The first one acknowledges only the **first frame**; the second one acknowledges **three frames**.
- In Selective Repeat, **ACKs are sent when data are delivered to the network layer**. If the data belonging to n frames are delivered in one shot, only one ACK is sent for all of them

## Algorithm 11.9 Sender-site Selective Repeat algorithm

```
1 Sw = 2m-1 ;
2 Sf = 0;
3 Sn = 0;
4
5 while (true)                                //Repeat forever
6 {
7     WaitForEvent();
8     if(Event(RequestToSend))                //There is a packet to send
9     {
10        if(Sn-Sf >= Sw)              //If window is full
11            Sleep();
12        GetData();
13        MakeFrame(Sn);
14        StoreFrame(Sn);
15        SendFrame(Sn);
16        Sn = Sn + 1;
17        StartTimer(Sn);
18    }
19 }
```

(continued)

Algorithm 11.9 *Sender-site Selective Repeat algorithm*

```
20  if(Event(ArrivalNotification)) //ACK arrives
21  {
22      Receive(frame);           //Receive ACK or NAK
23      if(corrupted(frame))
24          Sleep();
25      if (FrameType == NAK)
26          if (nakNo between Sf and Sn)
27          {
28              resend(nakNo);
29              StartTimer(nakNo);
30          }
31      if (FrameType == ACK)
32          if (ackNo between Sf and Sn)
33          {
34              while(sf < ackNo)
35              {
36                  Purge(sf);
37                  StopTimer(sf);
38                  Sf = Sf + 1;
39              }
40          }
41 }
```

(continued)

**Algorithm 11.9** *Sender-site Selective Repeat algorithm*

```
42  
43     if(Event(TimeOut(t)))          //The timer expires  
44     {  
45         StartTimer(t);  
46         SendFrame(t);  
47     }  
48 }
```

# Piggybacking

A technique called piggybacking **is used to improve the efficiency of the bidirectional protocols.**

- When a frame is carrying data from A to B, it can also carry control information about arrived (or lost) frames from B; when a frame is carrying data from B to A, it can also carry control information about the arrived (or lost) frames from A.

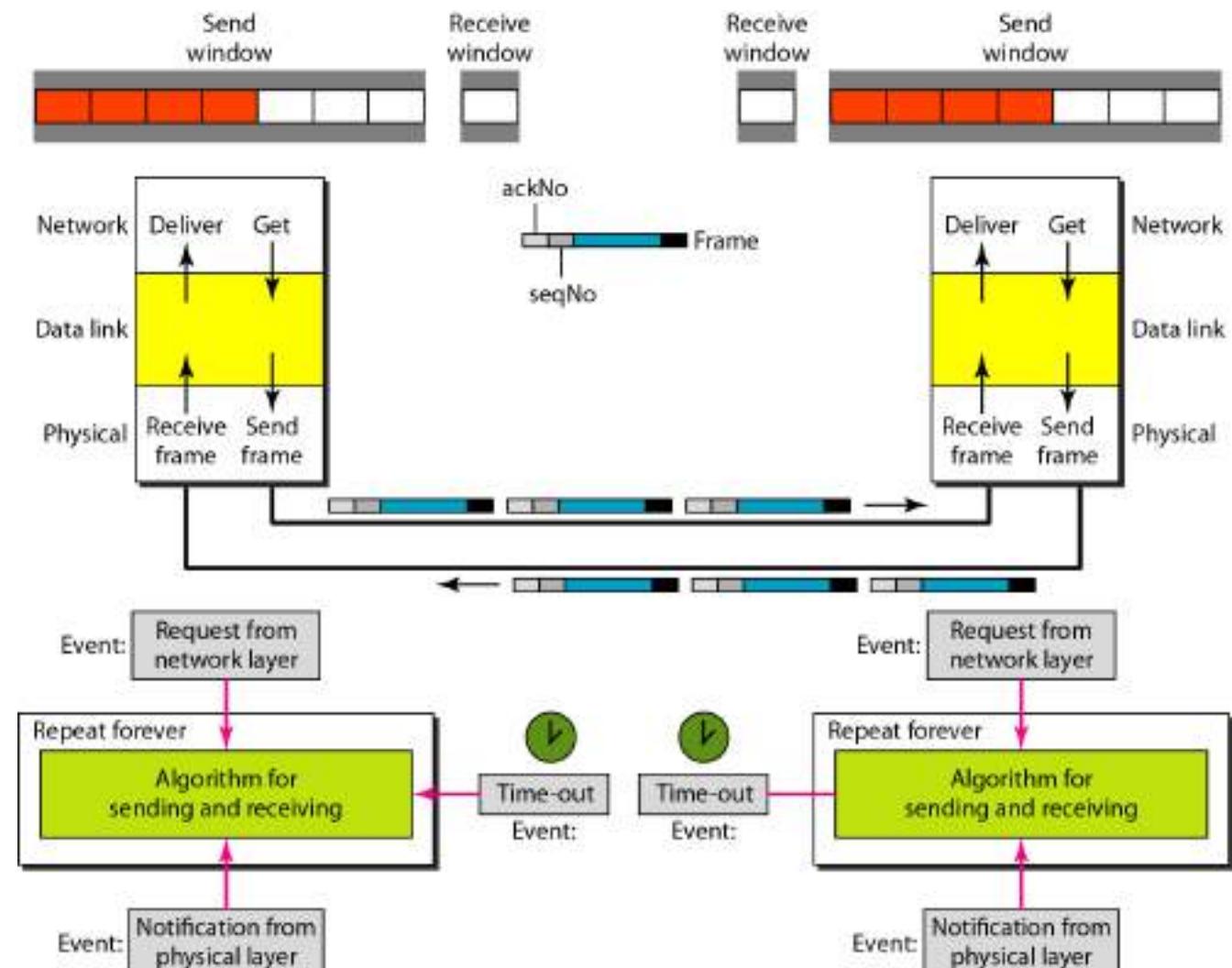
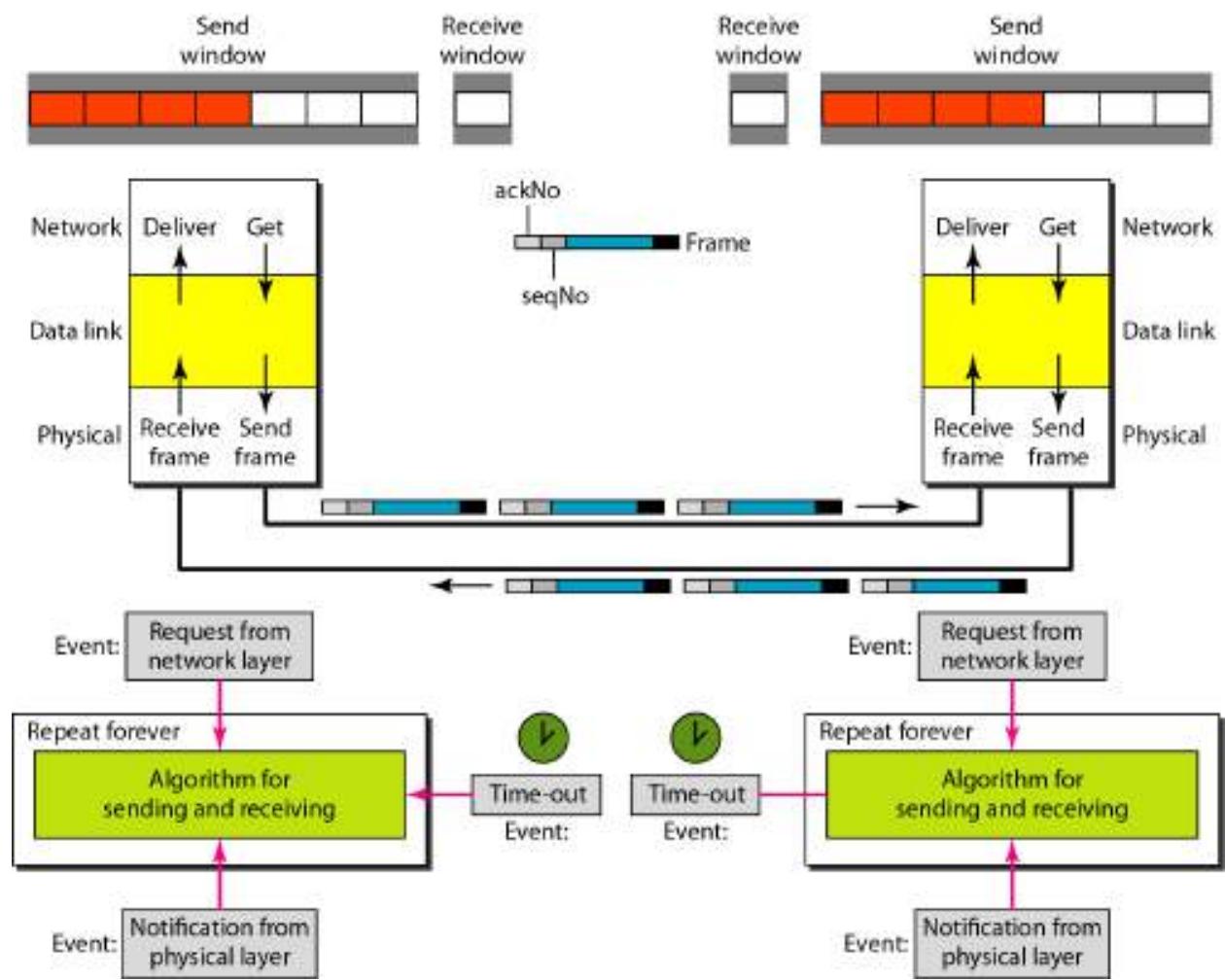


Figure 11.24 Design of piggybacking in Go-Back-N ARQ

- In real life, data frames are normally flowing in both directions: **from node A to node B** and from **node B to node A**.
- This means that the control information also needs to flow in both directions.
  - An important point about piggybacking is that both sites must use the same algorithm.
  - This algorithm is complicated because it needs to combine two arrival events into one.

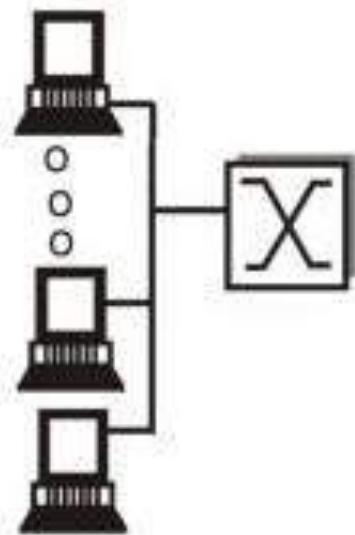


# **Media Access Control (MAC)**

# Multiple Access

- Broadcast link used in LAN consists of multiple sending and receiving nodes connected to or use a single shared link

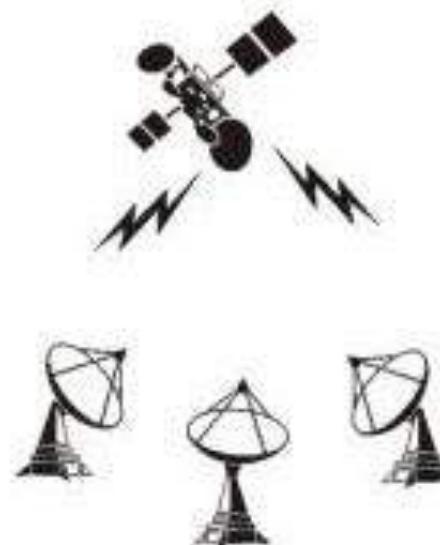
## Broadcast links Examples



shared wire  
(e.g. Ethernet)



shared wireless  
(e.g. Wavelan)



satellite

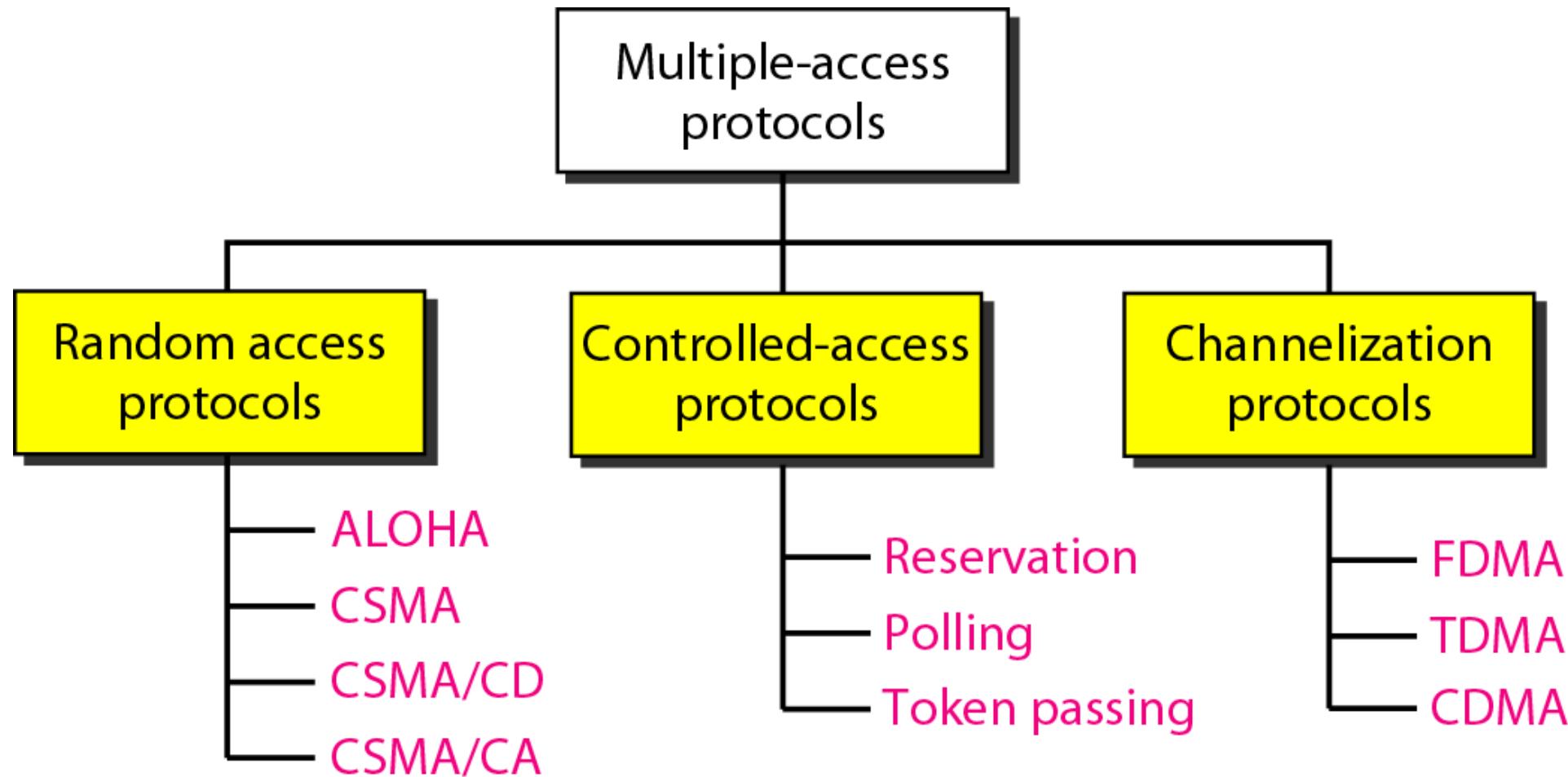


cocktail party

# Multiple Access

- **Problem:** When two or more **nodes transmit at the same time, their frames will collide** and the link bandwidth is **wasted** during collision
  - How to coordinate the access of multiple sending/receiving nodes to the shared link???
- **Solution:** We need a **protocol** to coordinate the transmission of the active nodes
- These protocols are called **Medium or Multiple Access Control (MAC) Protocols** belong to a **sublayer** of the data link layer called **MAC** (Medium Access Control)
- What is expected from Multiple Access Protocols:
  - Main task is to **minimize collisions** in order to **utilize the bandwidth** by:
    - Determining **when** a station can use the link (medium)
    - **what** a station should do when the link is **busy**
    - **what** the station should do when it is involved in **collision**

**Figure 12.2** *Taxonomy of multiple-access protocols discussed in this chapter*



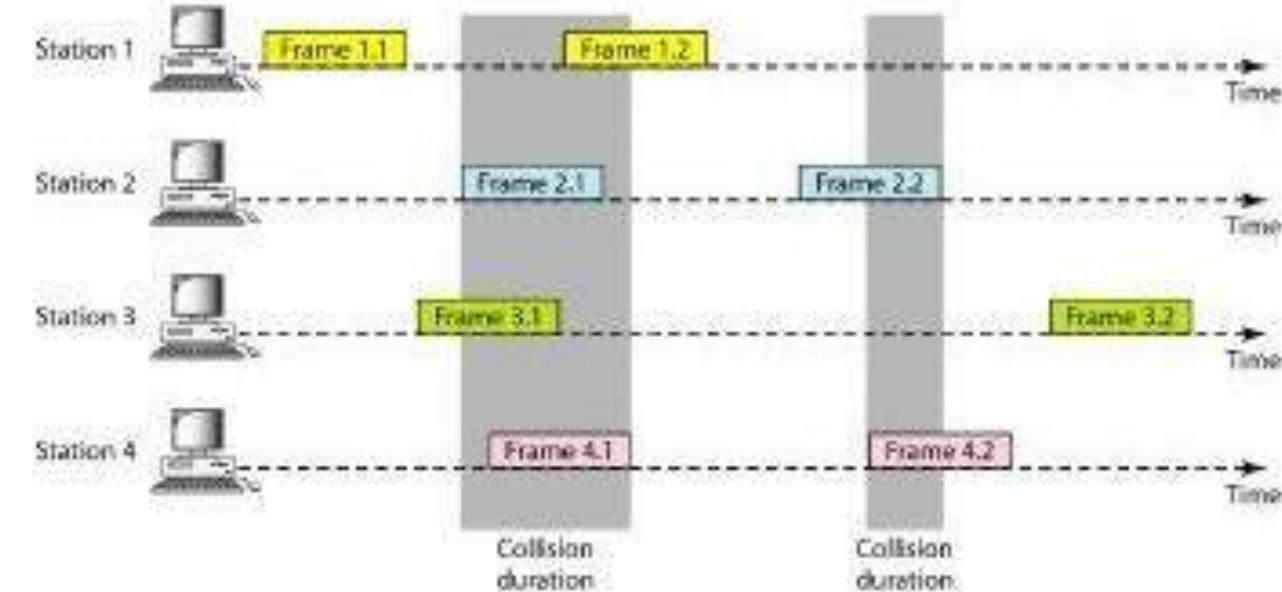
# Random Access Protocols

- **Random Access (or contention) Protocols:**
  - No station is superior to another station and none is assigned the control over another.
  - In a random-access method, each station has the right to the medium without being controlled by any other station.
  - However, if more than one station tries to send, there is an access conflict- ***collision***.
  - A station with a frame to be transmitted **can use the link directly based** on a procedure defined by the protocol to make a decision on whether or not to send.

# ALOHA Protocols

- ALOHA, the earliest random access method, was developed at the University of Hawaii in early 1970.
- It was designed for a **radio (wireless) LAN**, but it can be used on any shared medium.

Figure 12.3 *Frames in a pure ALOHA network*



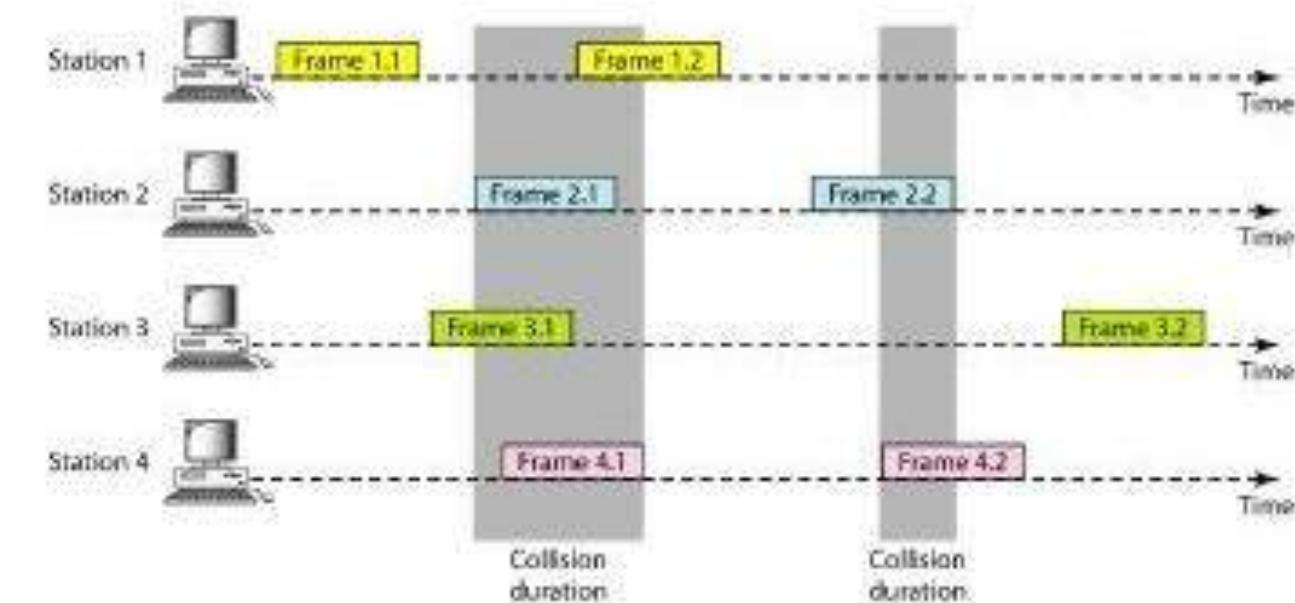
# ALOHA Protocols

- Pure ALOHA Protocol

## Description

- All frames from any station are of fixed length (**L bits**)
- Stations transmit at equal **transmission time** (*all stations produce frames with equal frame lengths*).
- A station that has data **can transmit at any time**

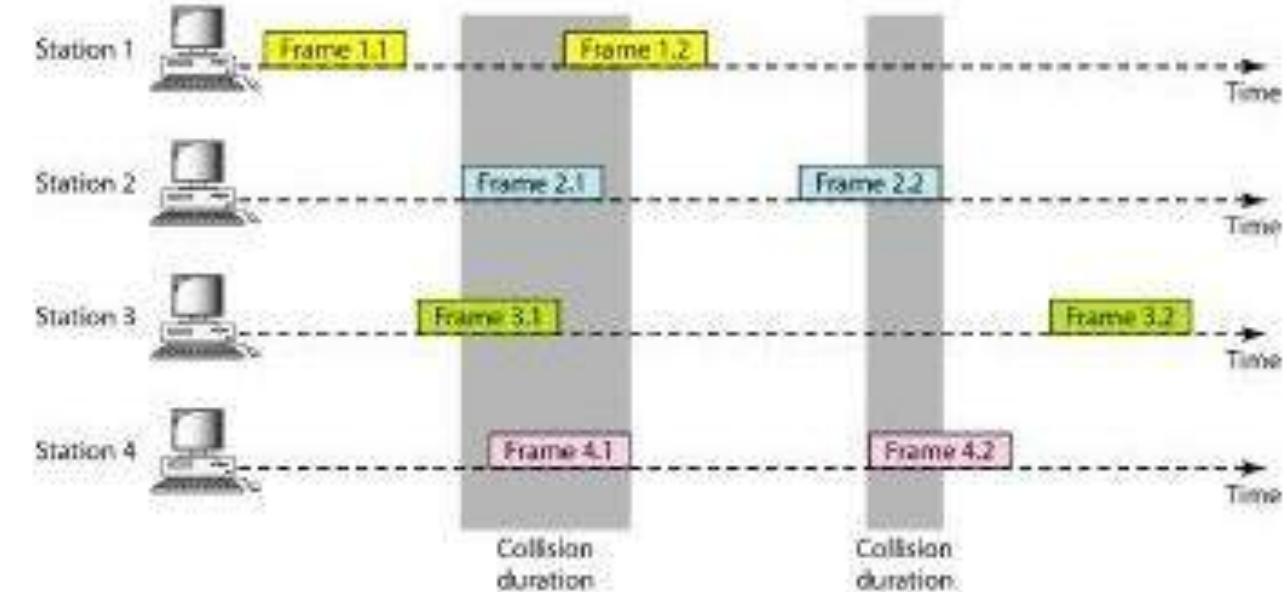
Figure 12.3 *Frames in a pure ALOHA network*



# ALOHA Protocols

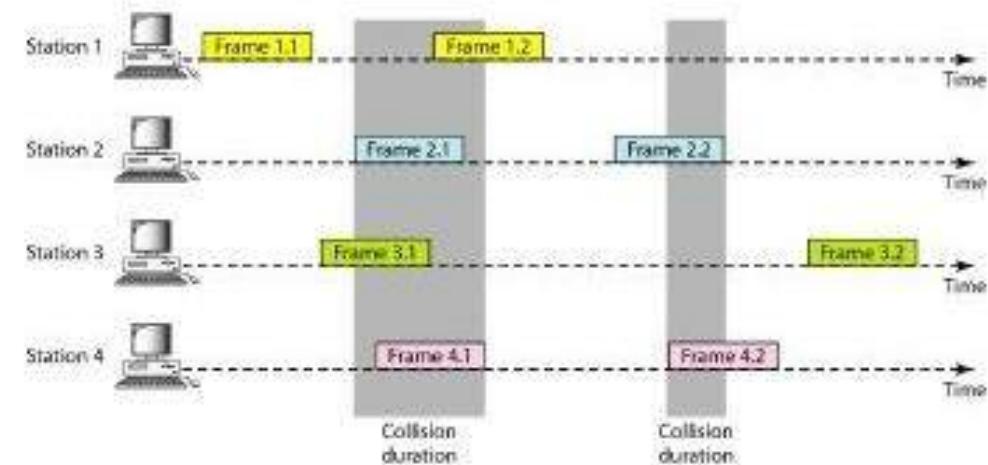
- Four stations that need to share the channel.
- Each station sends two frames
- Only two frames survive
- One bit of a frame coexists on the channel with one bit from another frame, there is a collision and both will be destroyed.
- Need to resend the frames that have been destroyed.

Figure 12.3 *Frames in a pure ALOHA network*



- After transmitting a frame, the sender waits for an acknowledgment for an amount of time (time out) equal to the **maximum round-trip propagation delay** =  $2 * t_{\text{prop}}$
- If no ACK was received, sender assumes that the frame or ACK has been destroyed and resends that frame after it waits for a *random* amount of time.

Figure 12.3 Frames in a pure ALOHA network



- A collision **involves two or more** stations.
- If all these stations try to resend their frames after the time-out, the frames will collide again.
- Each station waits **a random amount of time** before resending its frame. **The randomness will help avoid more collisions**. The time is called as **backoff time  $T_B$** .

- Pure ALOHA has a **second method** to prevent congesting the channel with retransmitted frames.
- After a maximum number of retransmission attempts  $K_{\max}$ , a station must give up and try later.
- For each retransmission a multiplier  $R = 0$  to  $2^K - 1$  (Randomly chosen)
- **Is randomly chosen and multiplied by  $T_p$  (maximum propagation time) or  $T_{fr}$  (the average time required to send out a frame) to find  $T_B$ .**

$$(T_B = R \times T_p \text{ or } R \times T_{fr})$$

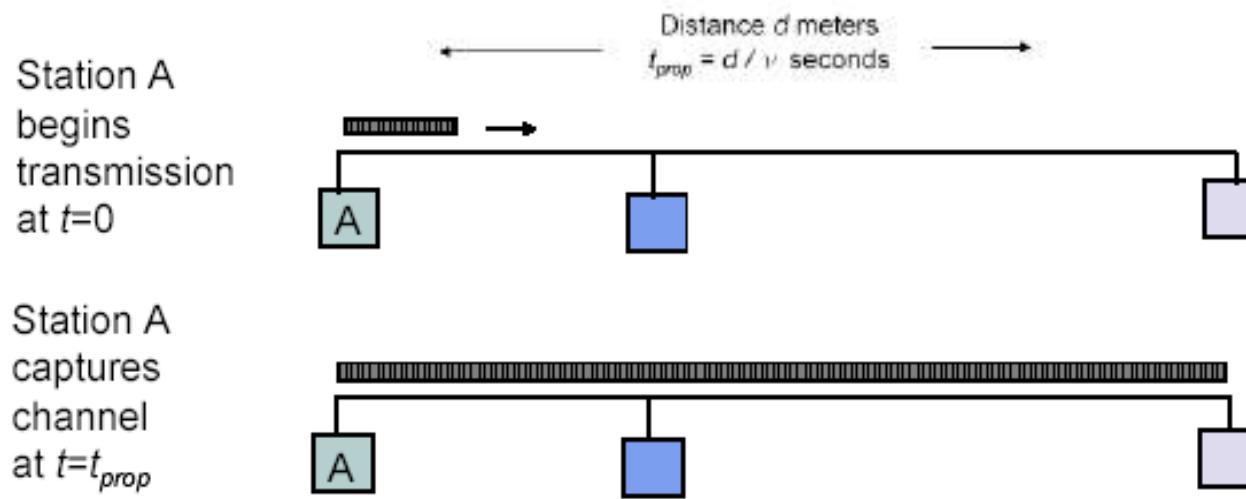
- The **backoff time  $T_B$**  is a random value that normally depends on  **$K$  (the number of attempted unsuccessful transmissions)**.

*K : the number of attempted unsuccessful transmissions*

# Maximum Propagation Delay ( $t_{\text{prop}}$ )

time it takes for a bit of a frame to travel between the two most widely separated stations.

- The time-out period is equal to the maximum possible round-trip propagation delay,
  - which is twice the amount of time required to send a frame between the two most widely separated stations ( $2 * T_p$ )



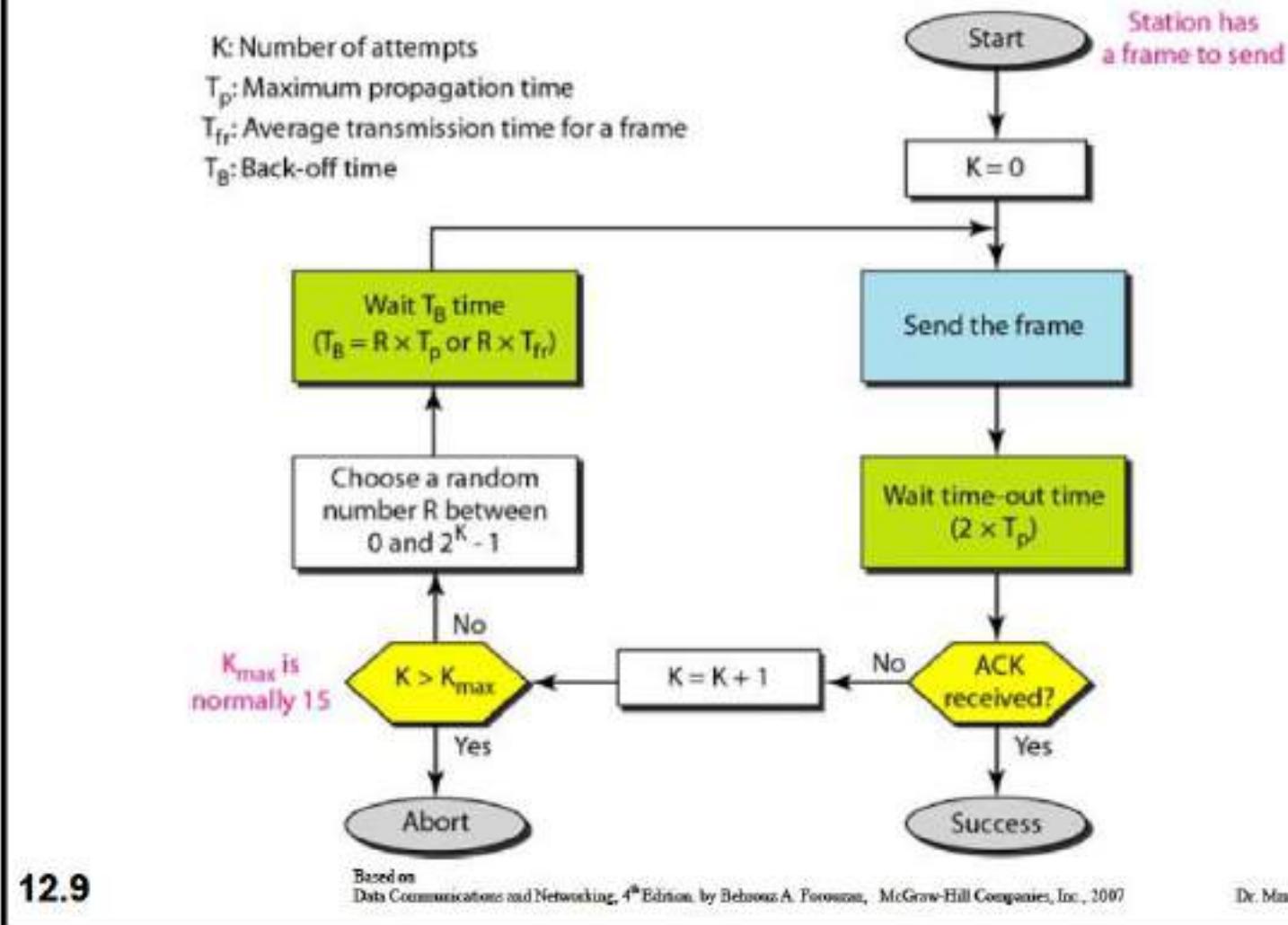
The farthest station

Station B receives the first bit  
of the frame at time  $t = t_{\text{prop}}$

**Figure 13.4** Procedure for ALOHA protocol

The range of the random numbers increases after each collision. The value of  $K_{\max}$  is usually chosen as 15.

**Figure 12.4** Procedure for pure ALOHA protocol





## **Example 12.1**

*The stations on a wireless ALOHA network are a maximum of 600 km apart. If we assume that signals propagate at  $3 \times 10^8$  m/s, we find*

$$T_p = (600 \times 10^5) / (3 \times 10^8) = 2 \text{ ms.}$$

*Now we can find the value of  $T_B$  for different values of  $K$ .*

- a. *For  $K = 1$ , the range is  $\{0, 1\}$ . The station needs to generate a random number with a value of 0 or 1. This means that  $T_B$  is either 0 ms ( $0 \times 2$ ) or 2 ms ( $1 \times 2$ ), based on the outcome of the random variable.*



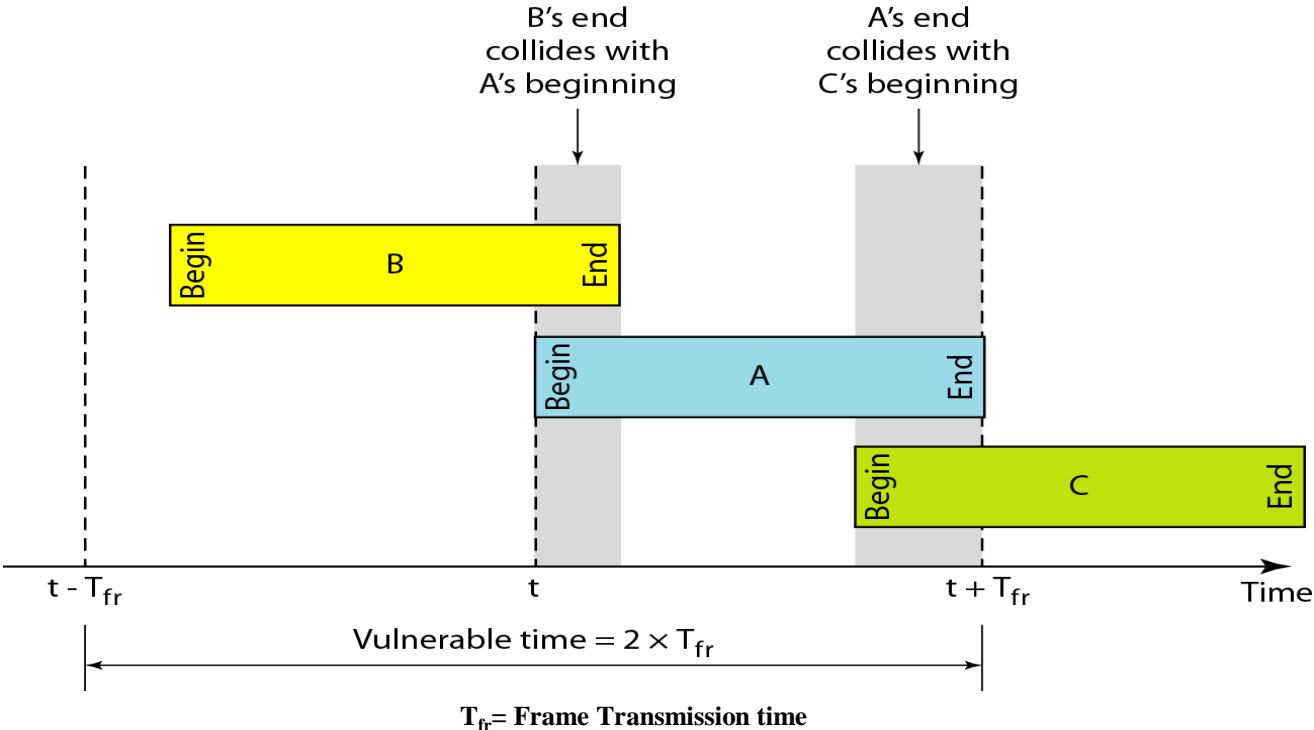
## *Example 12.1 (continued)*

- b.** For  $K = 2$ , the range is  $\{0, 1, 2, 3\}$ . This means that  $T_B$  can be 0, 2, 4, or 6 ms, based on the outcome of the random variable.
- c.** For  $K = 3$ , the range is  $\{0, 1, 2, 3, 4, 5, 6, 7\}$ . This means that  $T_B$  can be 0, 2, 4, . . . , 14 ms, based on the outcome of the random variable.

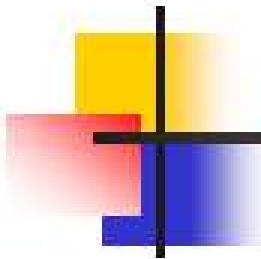
# Vulnerable time

**Vulnerable time** is the **time** during which no transmission should be done to avoid any collision.

- The length of time in which there is a **possibility of collision**.
- The stations send **fixed-length frames** with each frame taking  $T_{fr}$  seconds to send.
- Station A starts to send a frame at time  $t$ .
- Station B has started to send its frame after  $t-T_{fr}$ . This leads to **a collision between the frames from station A and station B**.



- Station C starts to send a frame before time  $t+T_{fr}$ . Here is also a collision between frames from station A and station C.
- We see that the **vulnerable time** during which a collision may occur in pure ALOHA is 2 times the frame transmission time.
- **Pure ALOHA vulnerable time =  $2 \times T_{fr}$**



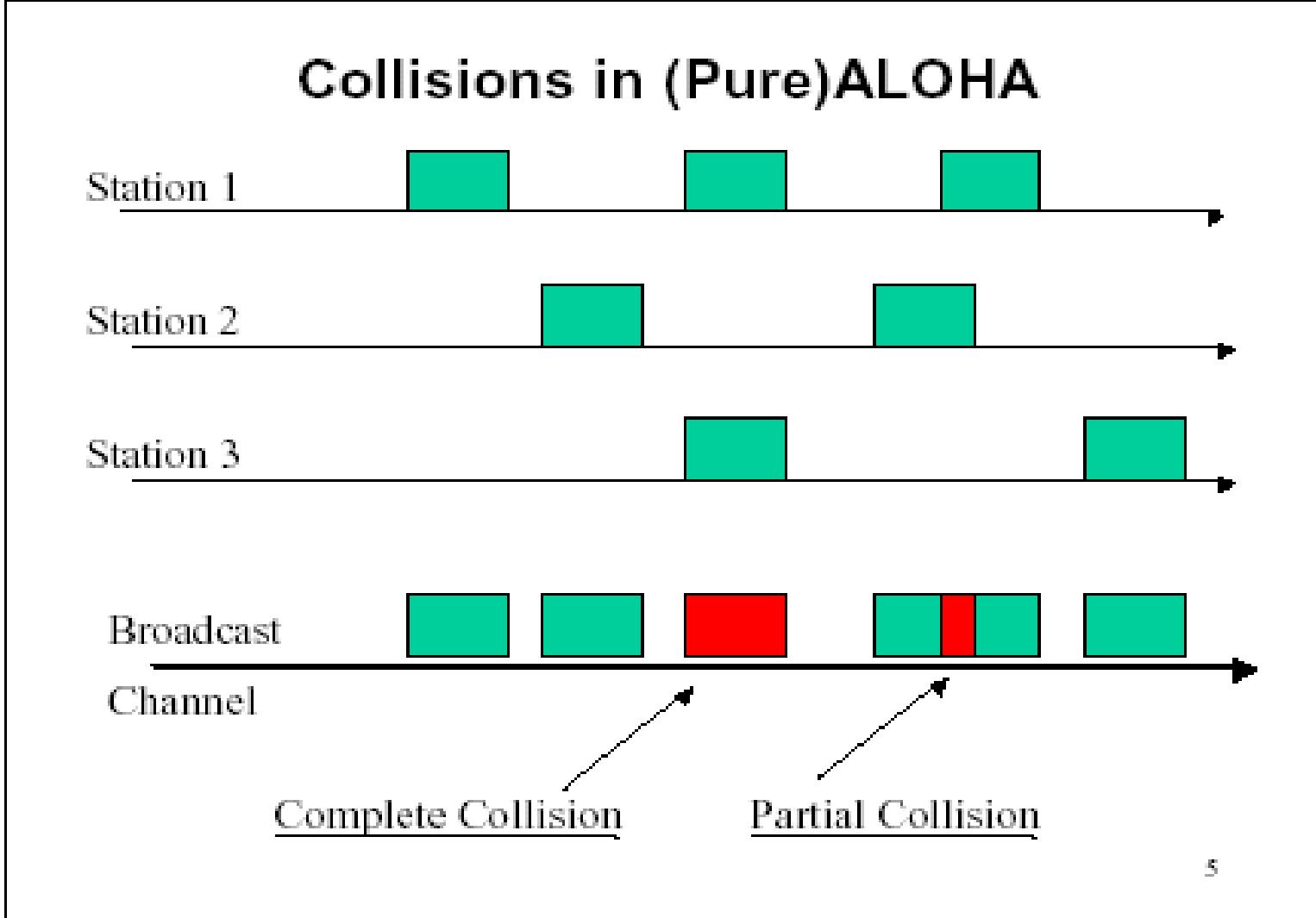
## Example 12.2

*A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the requirement to make this frame collision-free?*

### Solution

*Average frame transmission time  $T_{fr}$  is 200 bits/200 kbps or 1 ms. The vulnerable time is  $2 \times 1 \text{ ms} = 2 \text{ ms}$ . This means no station should send later than 1 ms before this station starts transmission and no station should start sending during the one 1-ms period that this station is sending.*

# Pure ALOHA



In pure ALOHA, frames are transmitted at completely arbitrary times.

# Throughput

- Let us call **G** the average number of frames generated by the system during one frame transmission time.

- Then average number of **successfully transmitted frames (S)** for pure ALOHA is

$$S = G \times e^{-2G}$$

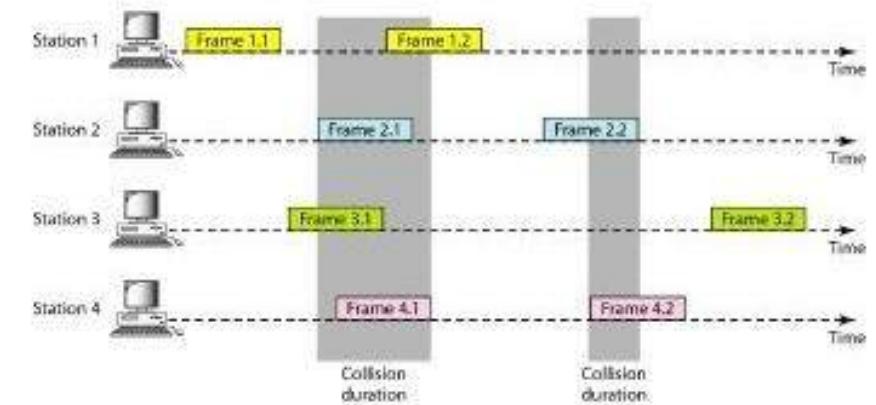
- The maximum throughput  $S_{\max}$  is 0.184 for  $G=1/2$

- In other words, if one-half a frame is generated during one frame transmission time (one frame during two frame transmission times), than 18.4 percent of these frames reach their destination successfully.

- If station generates only **one frame** in this **vulnerable time** (add no other stations generates a frame during this time), the frame will reach its destination successfully.

- **Channel utilization or efficiency or Throughput** is the **percentage** of the transmitted frames that arrive **successfully** (without collisions) **or** the **percentage of the channel bandwidth** that will be used for transmitting frames without collisions
- Pure ALOHA Maximum channel utilization is **18%** (i.e, if the system produces **F frames/s**, then  **$0.18 * F$**  frames will arrive **successfully on average without the need of retransmission**).

Figure 12.3 Frames in a pure ALOHA network



# Throughput

- Let us call  $G$  the average number of frames generated by the system during one frame transmission time.

---

**The throughput (  $S$ ) for pure ALOHA is**  
 $S = G \times e^{-2G}$  .

**The maximum throughput**

$S_{\max} = 0.184$  when  $G= (1/2)$ .

---

$G$  = Average number of frames generated by the system  
(all stations) during one frame transmission time



## **Example 12.3**

*A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the throughput if the system (all stations together) produces*

- a. 1000 frames per second*
- b. 500 frames per second*
- c. 250 frames per second.*

### **Solution**

*The frame transmission time is 200/200 kbps or 1 ms.*

- a. If the system creates 1000 frames per second, this is 1 frame per millisecond. The load is 1. In this case  $S = G \times e^{-2G}$  or  $S = 0.135$  (13.5 percent). This means that the throughput is  $1000 \times 0.135 = 135$  frames. Only 135 frames out of 1000 will probably survive.*

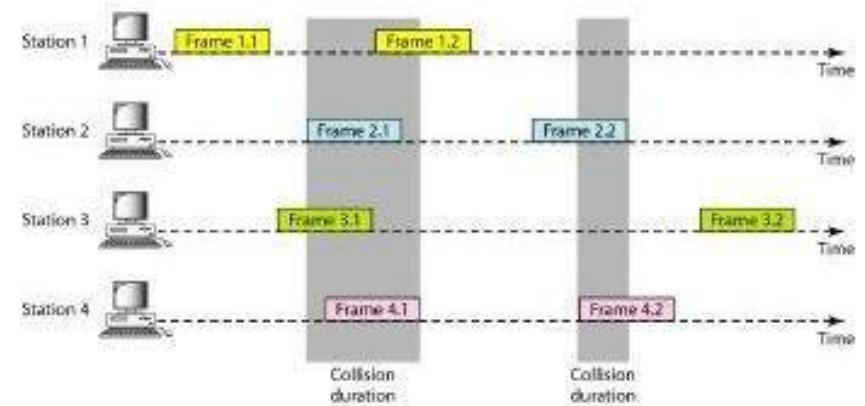


## **Example 12.3 (continued)**

- b.** If the system creates 500 frames per second, this is  $(1/2)$  frame per millisecond. The load is  $(1/2)$ . In this case  $S = G \times e^{-2G}$  or  $S = 0.184$  (18.4 percent). This means that the throughput is  $500 \times 0.184 = 92$  and that only 92 frames out of 500 will probably survive. Note that this is the maximum throughput case, percentagewise.
- c.** If the system creates 250 frames per second, this is  $(1/4)$  frame per millisecond. The load is  $(1/4)$ . In this case  $S = G \times e^{-2G}$  or  $S = 0.152$  (15.2 percent). This means that the throughput is  $250 \times 0.152 = 38$ . Only 38 frames out of 250 will probably survive.

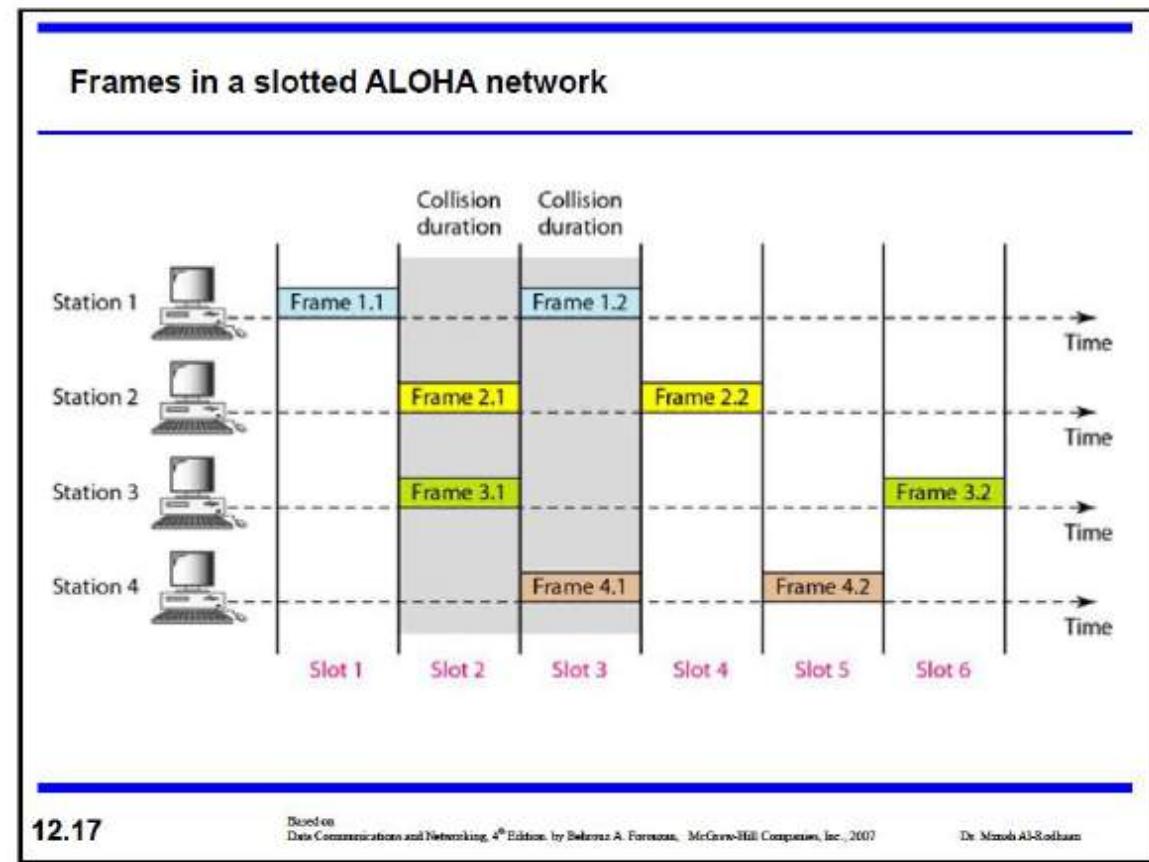
- **Pure ALOHA vulnerable time =  $2 \times T_{fr}$ .**
- This is so because there is no rule that defines when the station can send.
- A station may send soon after another station has started or just before another station has finished.
- **Slotted ALOHA** was invented to improve the **efficiency of pure ALOHA**.

Figure 12.3 Frames in a pure ALOHA network



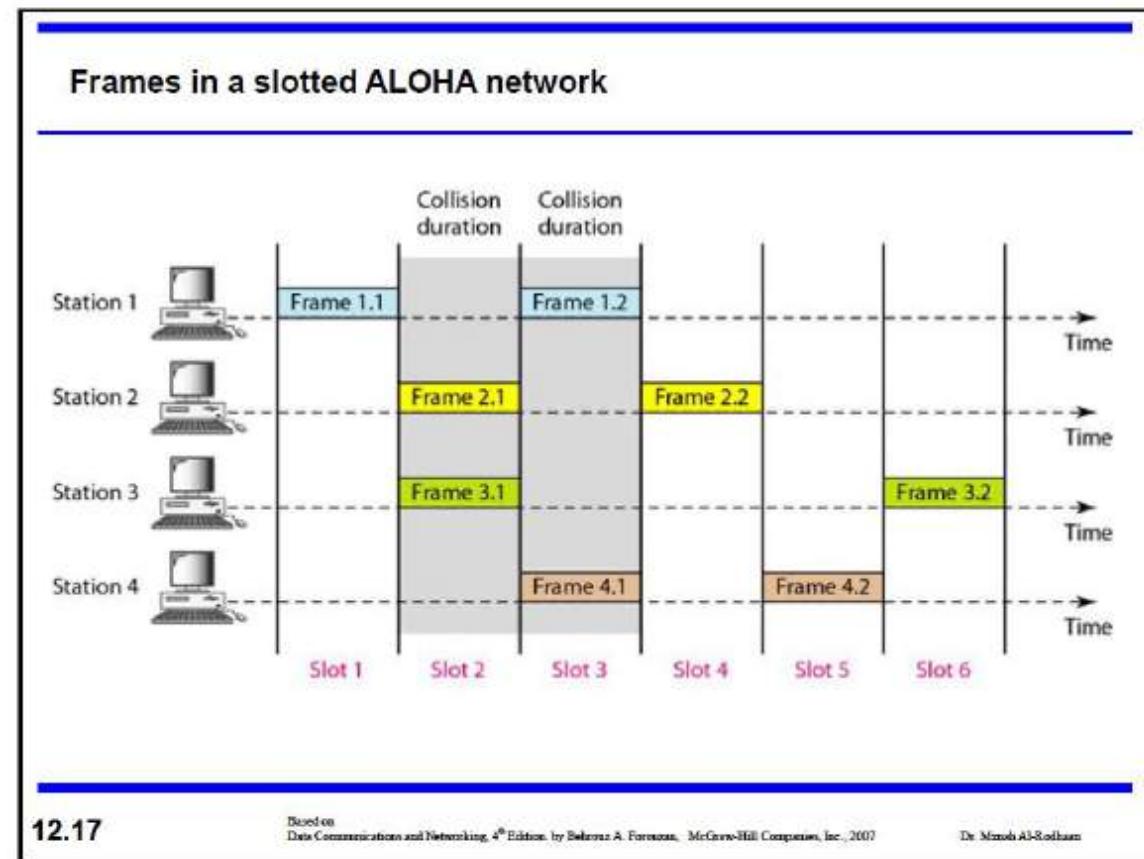
# Random Access – Slotted ALOHA

- Time is divided into slots equal to a **frame transmission time ( $T_{fr}$ )**
- A station can transmit at the beginning of a slot only.
- If a station misses the beginning of a slot, it has to wait until the beginning of the next time slot.
- This means that the station which started at the beginning of this slot has already finished sending its frame.



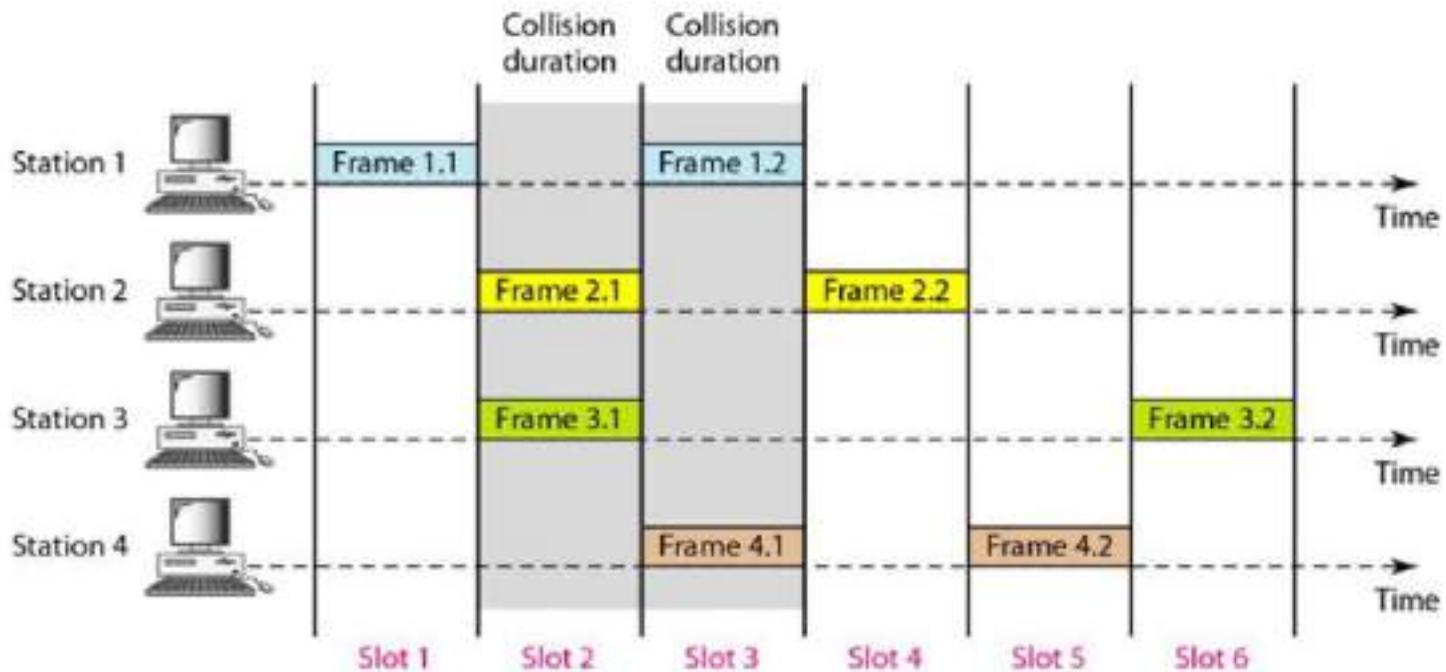
# Random Access – Slotted ALOHA

- Still there is a possibility of collision if two stations try to send at the beginning of the same time slot.
- However, the vulnerable time is now reduced to one-half, equal to ?

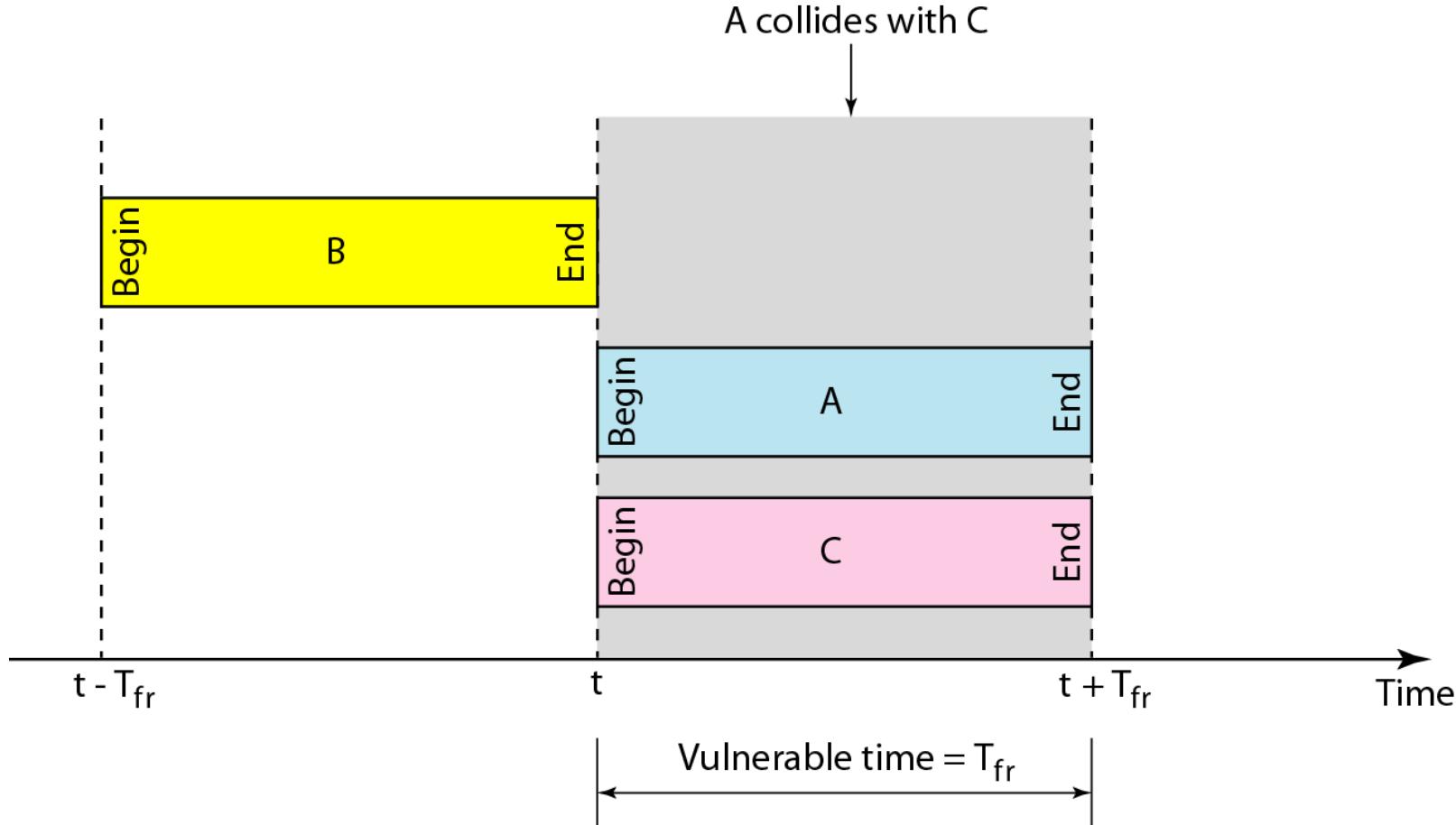


# Random Access – Slotted ALOHA

Frames in a slotted ALOHA network



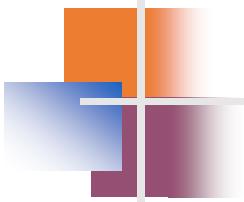
## *In danger time for slotted ALOHA protocol*



**Slotted ALOHA vulnerable time =  $T_{fr}$**

# Throughput

- Let us call **G** the average number of frames generated by the system during one frame transmission time.
- Then average number of **successfully transmitted frames (S)** for slotted ALOHA is
$$S = G \times e^{-G}$$
- The maximum throughput  $S_{\max}$  is 0.368 for G=1
- In other words, if one frame is generated during one frame transmission time ,than 36.8 percent of these frames reach their destination successfully.
- We expect G=1 to produce maximum throughput because the vulnerable time is equal to the frame transmission time.



If a station generates only one frame in this vulnerable time (and no other station generates a frame during this time), the frame will reach its destination successfully.

---

**The throughput for slotted ALOHA is**  
 **$S = G \times e^{-G}$ .**

**The maximum throughput**  
 **$S_{max} = 0.368$  when  $G = 1$ .**

---



## *Example 12.4*

*A slotted ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the throughput if the system (all stations together) produces*

- a. 1000 frames per second*
- b. 500 frames per second*
- c. 250 frames per second.*

### *Solution*

*The frame transmission time is 200/200 kbps or 1 ms.*

- a. If the system creates 1000 frames per second, this is 1 frame per millisecond. The load is 1. In this case  $S = G \times e^{-G}$  or  $S = 0.368$  (36.8 percent). This means that the throughput is  $1000 \times 0.0368 = 368$  frames. Only 386 frames out of 1000 will probably survive.*



## *Example 12.4 (continued)*

- b. If the system creates 500 frames per second, this is (1/2) frame per millisecond. The load is (1/2). In this case  $S = G \times e^{-G}$  or  $S = 0.303$  (30.3 percent). This means that the throughput is  $500 \times 0.0303 = 151$ . Only 151 frames out of 500 will probably survive.*
- c. If the system creates 250 frames per second, this is (1/4) frame per millisecond. The load is (1/4). In this case  $S = G \times e^{-G}$  or  $S = 0.195$  (19.5 percent). This means that the throughput is  $250 \times 0.195 = 49$ . Only 49 frames out of 250 will probably survive.*

# Carrier Sense Multiple Access (CSMA)

- To minimize the chance of collision and therefore, increase the performance, the CSMA method was developed.
- The chance of collision can be reduced if a station senses the medium before trying to use it.
- CSMA is based on the principle “sense before transmit” or “listen before talk”
- CSMA can reduce the possibility of collision, but it cannot eliminate it.

# Carrier Sense Multiple Access (CSMA)

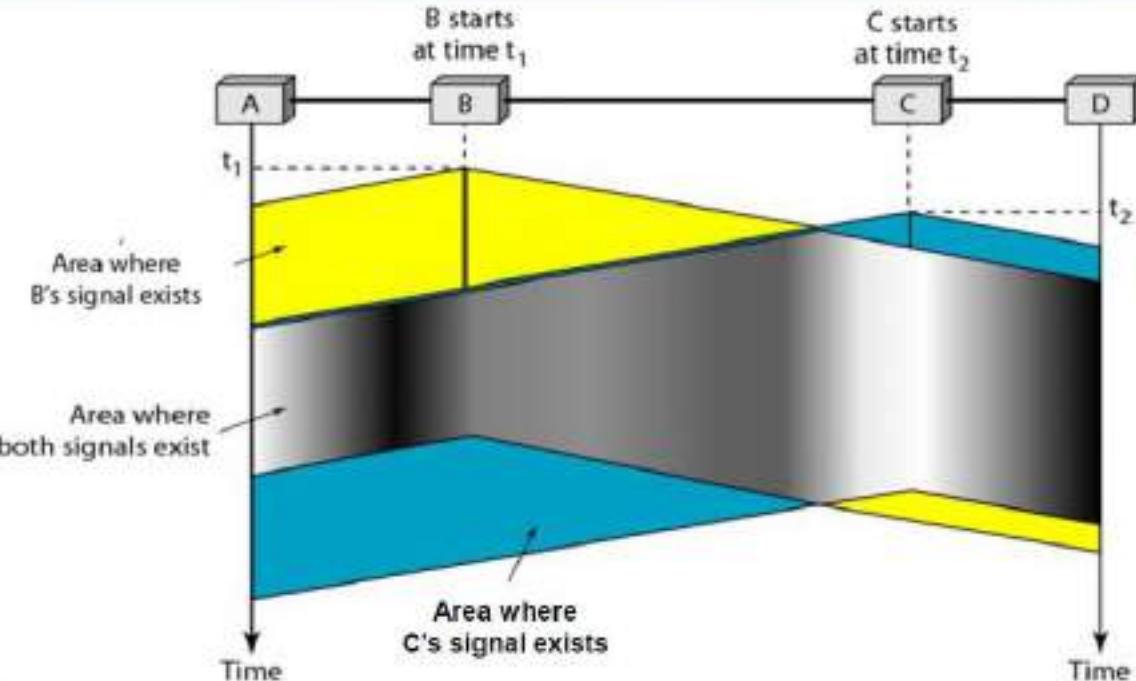
- The possibility of collision still exists because of **propagation delay**;
- When a station sends frame, **it still takes time (although very short) for the first bit to reach every station** and for every station to sense it.
- A station may sense the medium and find it idle, only because the first bit sent by another station has not yet been received.

# Carrier Sense Multiple Access (CSMA)

- At time  $t_1$ , station B senses the medium and finds it idle, so it sends a frame.
- At time  $t_2$  ( $t_2 > t_1$ ), station C senses the medium and finds it idle because, at this time, the first bits from station B have not reached station C.
- Station C also sends a frame.
- The two signals collide and both frames are destroyed.

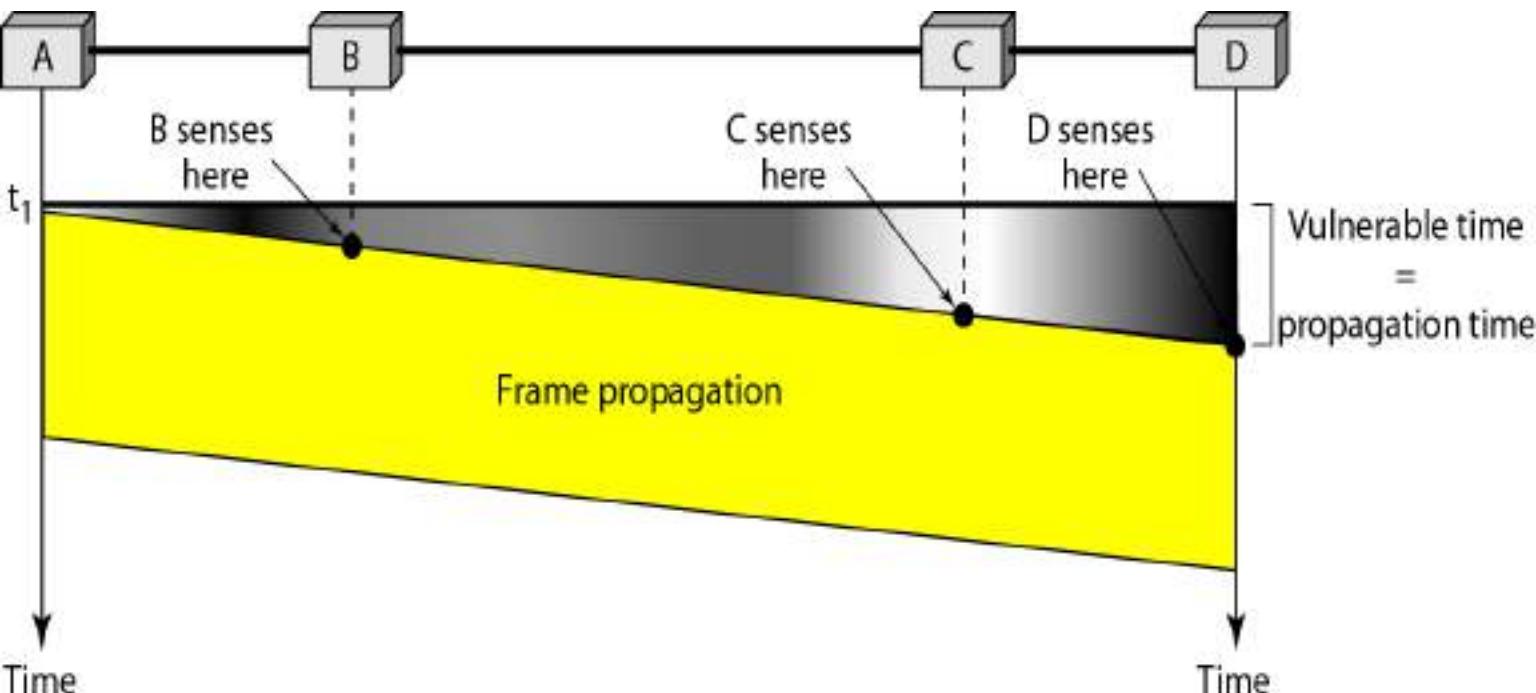
**Figure 12.8** Space/time model of the collision in CSMA

If everyone is sensing the medium, why collisions still occur?



## Random Access – Carrier Sense Multiple Access (CSMA)

- Vulnerable time for CSMA is the maximum propagation time
- The longer the propagation delay, the worse the performance of the protocol because of the above case.



- The figure shows the worst case.
- The left station, A, sends a frame at a time  $t_1$ , which reaches the rightmost station, D, at time  $t_1 + T_p$ .

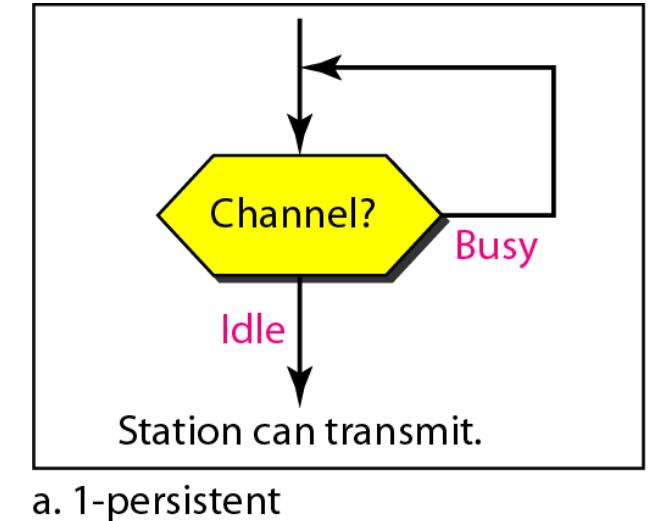
# Types of CSMA Protocols

## Persistence Method

- What a station should do when the medium is **idle**?
- What a station should do when the medium is **busy**?
  1. 1-Persistent CSMA
  2. Non-Persistent CSMA
  3. p-Persistent CSMA

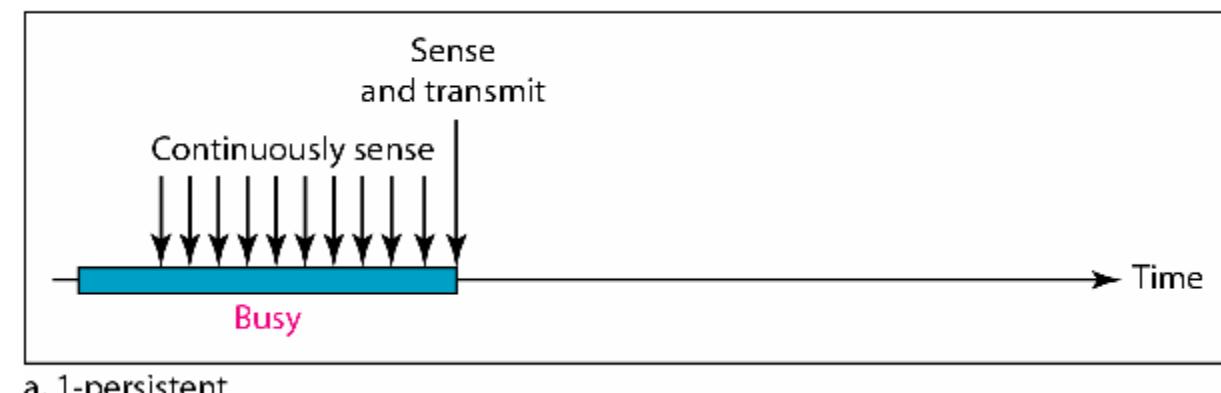
# 1-persistent CSMA

- To avoid **idle channel time**, 1-persistent protocol used
- Station wishing to transmit listens to the medium:
  - If medium idle, **transmit** immediately;
  - If medium busy, **continuously listen** until medium becomes idle; then transmit immediately with probability 1
- Performance
  - 1-persistent stations are **selfish**
  - If two or more stations becomes ready at the same time, **collision guaranteed**



a. 1-persistent

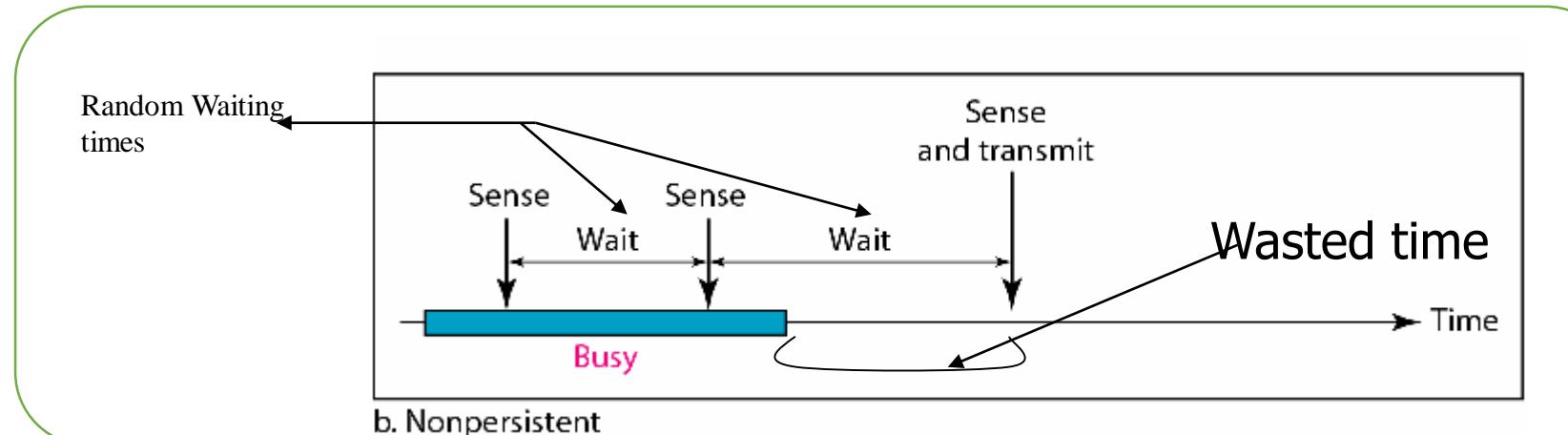
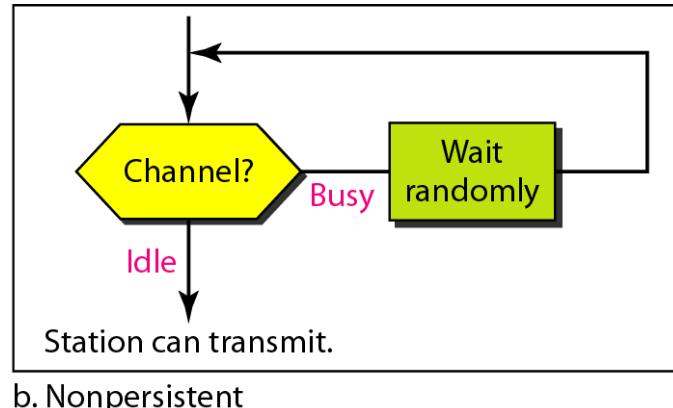
This method has the **highest chance of collision** because two or more stations may find the idle and send their frames immediately.



a. 1-persistent

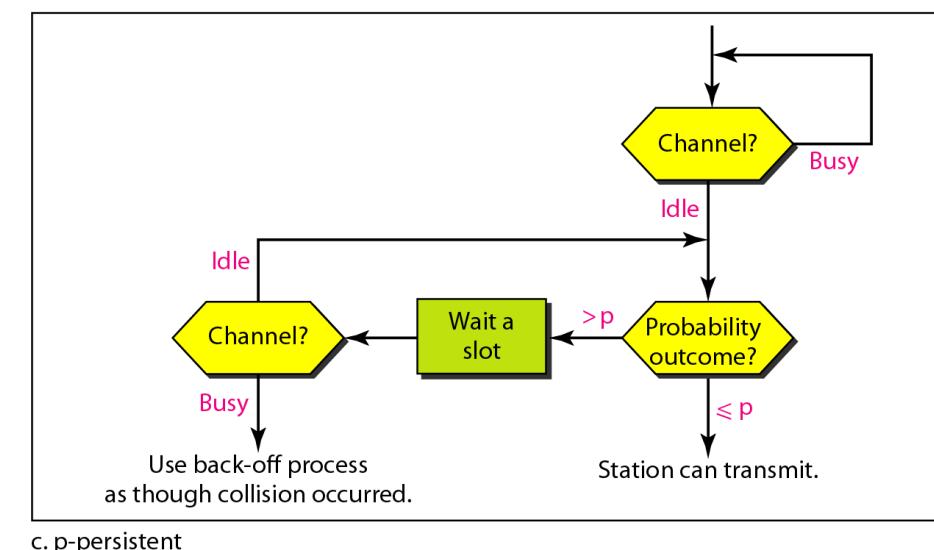
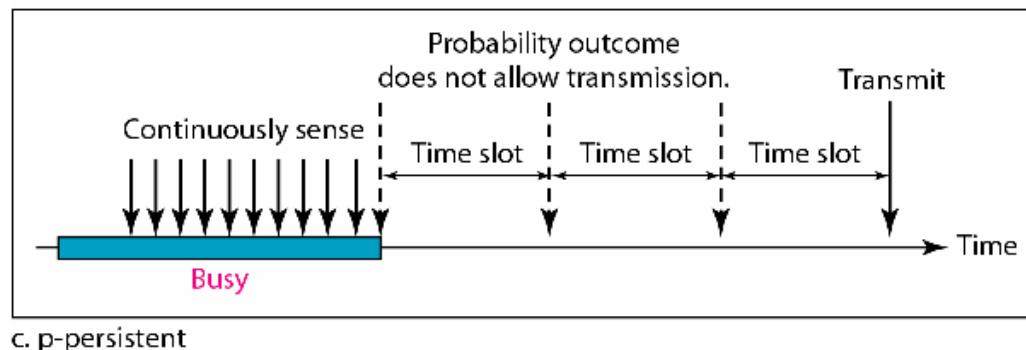
# Nonpersistent CSMA

- A station with frames to be sent, should sense the medium
  1. If medium is idle, **transmit**; otherwise, go to 2
  2. If medium is busy, (**backoff**) wait a *random amount of time* and repeat 1
- Performance:
  - **Random delays reduces probability of collisions** because two stations with data to be transmitted will wait for different amount of times.
  - **Bandwidth is wasted** if waiting time (backoff) is large because medium will remain idle following end of transmission even if one or more stations have frames to send



# P-persistent CSMA

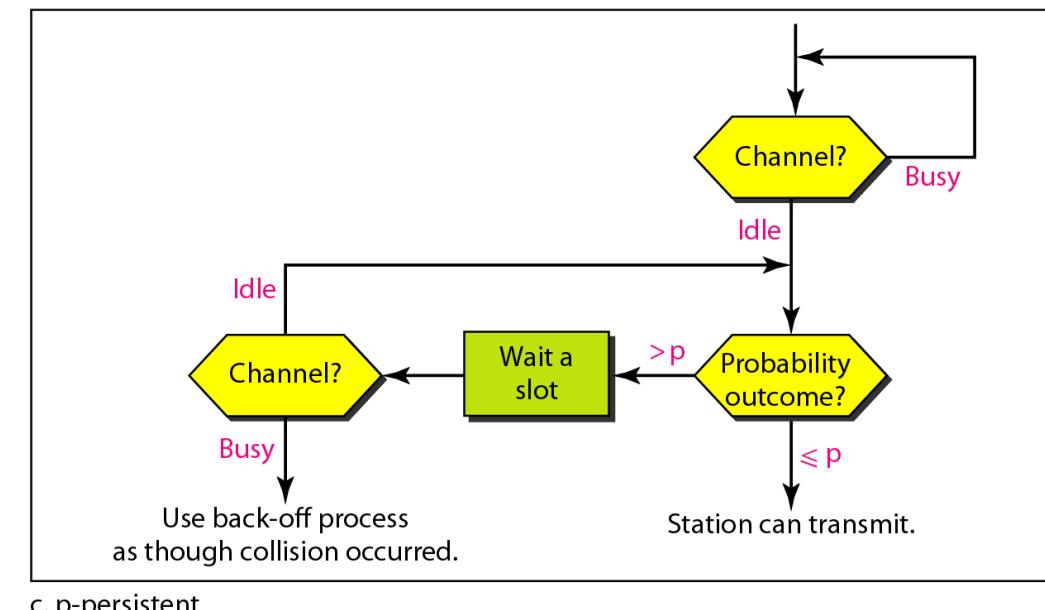
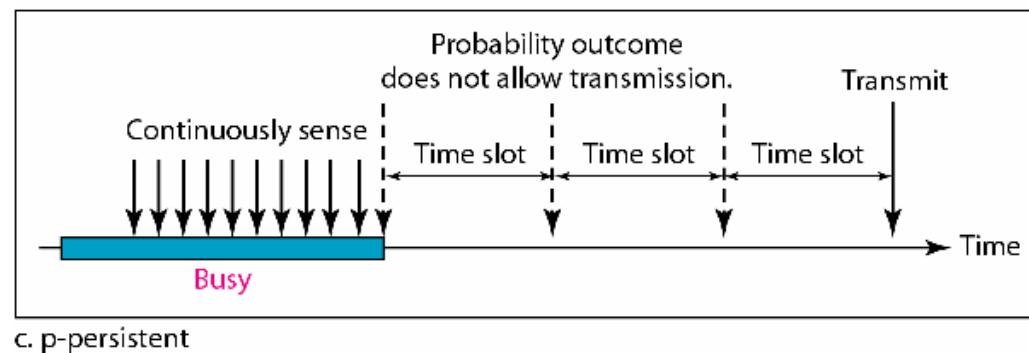
- Time is divided to slots where each Time unit (slot) typically equals **maximum propagation delay**
- Station wishing to transmit listens to the medium if medium idle,
  1. With probability  $p$ , the station sends its frame
  2. With probability  $q=1-p$ , the station waits for the beginning of the next time slot and checks the line again.
    - If the line is idle, it goes to step 1
    - If the line is busy, it acts as though a collision has occurred and uses the back-off procedure.



# P-persistent CSMA

## 1. Performance

- Reduces the possibility of collisions like **nonpersistent**
- Reduces channel idle time like **1-persistent**



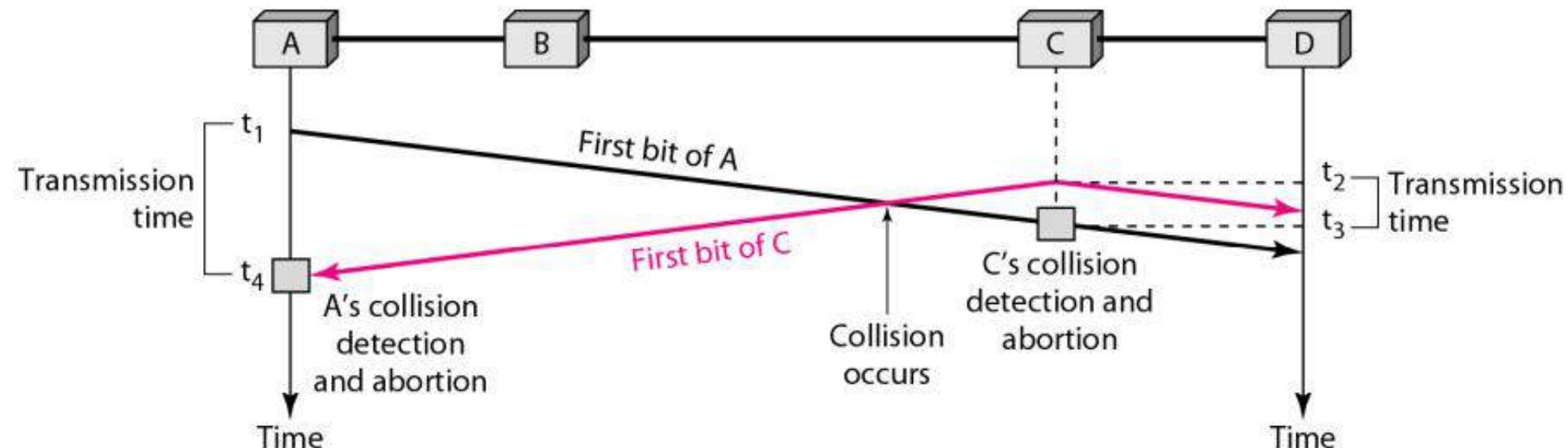
# CSMA/CD (Carrier Sense Multiple Access Collision Detection)

- *CSMA (all previous methods) has an inefficiency:*
  - If a collision has occurred, the channel is **unstable** until colliding packets have been fully transmitted
- *CSMA/CD (Carrier Sense Multiple Access with Collision Detection) overcomes this as follows:*
  - While transmitting, the sender is **listening** to **medium** for collisions.
  - Sender **stops transmission** if collision has occurred **reducing channel wastage**.

CSMA/CD is Widely used for **bus topology LANs** (IEEE 802.3, **Ethernet**).

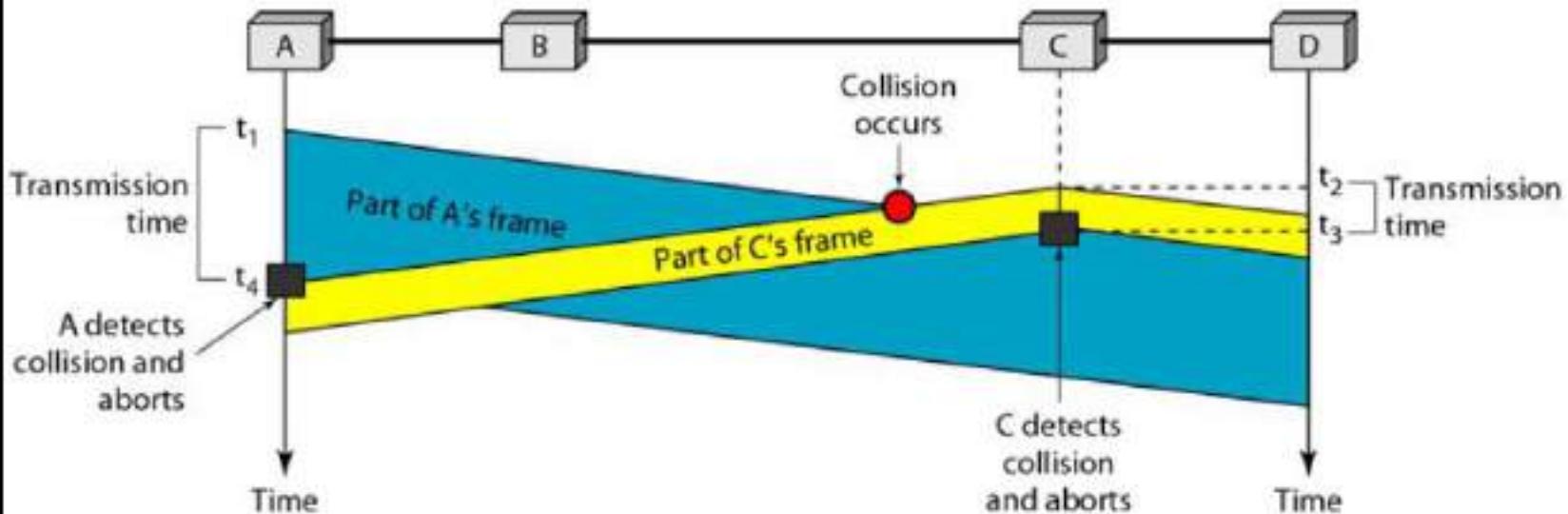
# CSMA/CD (Carrier Sense Multiple Access Collision Detection)

- At the first bits transmitted by the two stations involved in the collision.
- Although each station continues to send bits in the frame until it detects the collision,



- Looking at the figure , we see that A transmits for the duration  $t_4-t_1$ ; C transmits for the duration  $t_3-t_2$ .

**Figure 12.13 Collision and abortion in CSMA/CD**

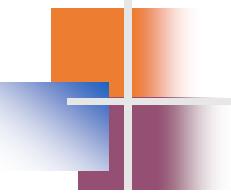


# Minimum Frame Size

- ❑ For CSMA/CD to work, **we need a restriction on the frame size.** Before sending the last bit of the frame, the sending must detect a collision, if any, and abort the transmission.
- ❑ This is so because the station, once the entire frame is sent, does not keep a copy of the frame and does not monitor the line for collision detection.
- ❑ Therefore, the frame transmission time  $T_{fr}$  must be at least two times the maximum propagation time  $T_p$

# Minimum Frame Size

- ❑ The frame transmission time  $T_{fr}$  must be at least two times the maximum propagation time  $T_p$ .
- ❑ To understand the reason, let us think about the worst-case scenario.
- ❑ If the two stations involved in a collision are **the maximum distance collision** are **the maximum distance apart**, the signal from the first takes time  $T_p$  to reach the second, and the effect of the collision takes another time  $T_p$  to reach the first. So the requirement is that the first station must still be transmitting after  $2T_p$ .



## Example 12.5

A network using CSMA/CD has a bandwidth of 10 Mbps. If the maximum propagation time (including the delays in the devices and ignoring the time needed to send a jamming signal, as we see later) is 25.6  $\mu$ s, what is the minimum size of the frame?

### Solution

The frame transmission time is  $T_{fr} = 2 \times T_p = 51.2 \mu s$ . This means, in the worst case, a station needs to transmit for a period of 51.2  $\mu$ s to detect the collision. The minimum size of the frame is  $10 \text{ Mbps} \times 51.2 \mu s = 512$  bits or 64 bytes. This is actually the minimum size of the frame for Standard Ethernet.

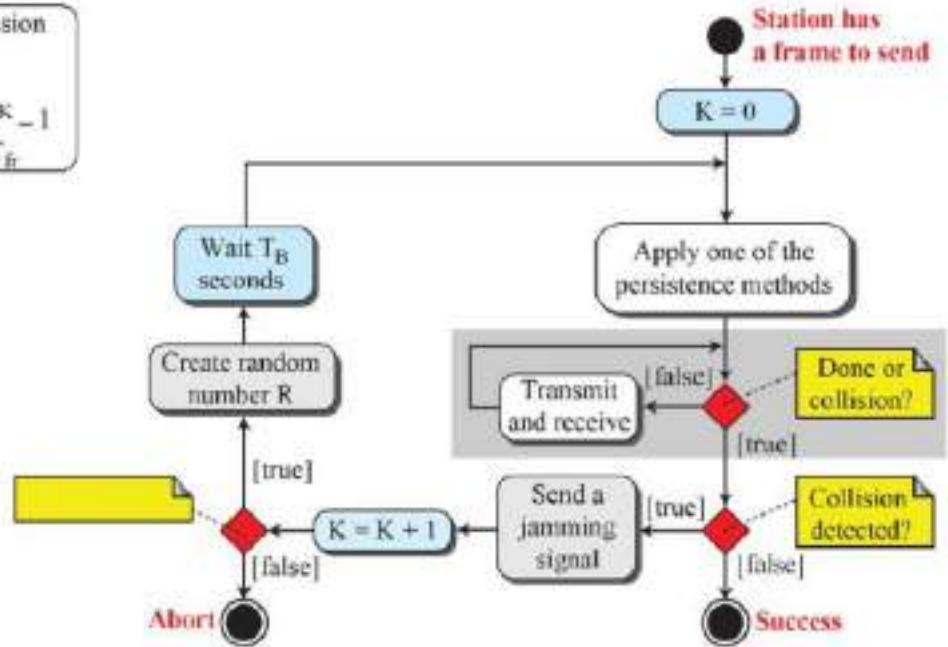
# Procedure

- Now let us look at the flow diagram for CSMA/CD. It is similar to the one of the ALOHA protocol, but there are differences.
- The first difference** is the addition of persistence process.
- We need to sense the channel before we start sending the frame by using one of the persistence processes (1-persistent, nonpersistent, or p-persistent)

## • *Procedure*

### Legend

$T_{ft}$ : Frame average transmission time  
 $K$ : Number of attempts  
 $R$ : (random number): 0 to  $2^K - 1$   
 $T_B$ : (Back-off time) =  $R \times T_{ft}$



**Fig. 12.13** Flow diagram for the CSMA/CD

# Procedure

- The second difference is the frame transmission.
- In ALOHA, we first transmit the entire frame and then wait for an acknowledgement.
- In CSMA/CD, transmission and collision detection are continuous processes.
- We don't send the entire frame and then look for a collision.
- We use a loop to show that transmission is a continuous process
- The third difference** is the sending of a short jamming signal to make sure that all other stations because aware of the collision.

## Procedure

### Legend

$T_{ft}$ : Frame average transmission time  
 $K$ : Number of attempts  
 $R$ : (random number): 0 to  $2^K - 1$   
 $T_B$ : (Back-off time) =  $R \times T_{ft}$

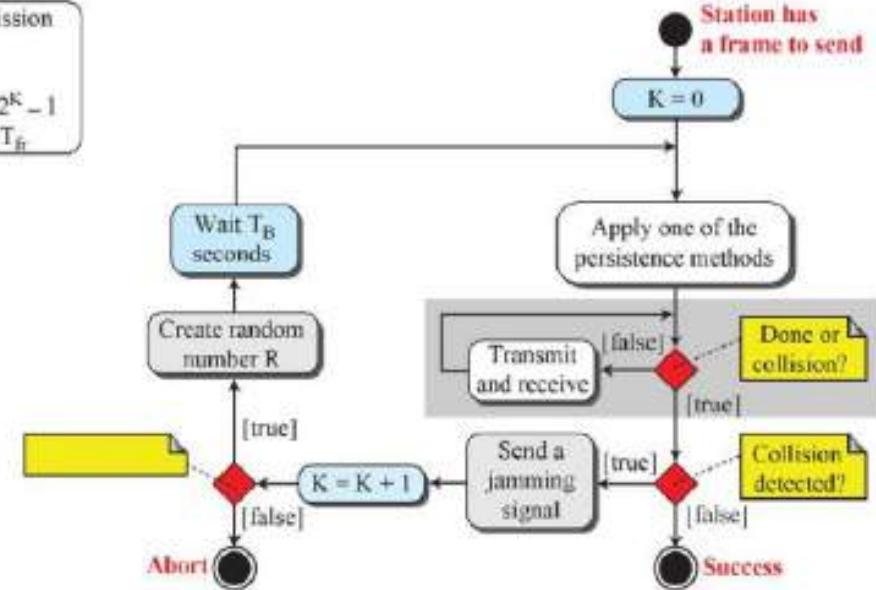


Fig. 12.13 Flow diagram for the CSMA/CD

# *CSMA/CD Protocol*

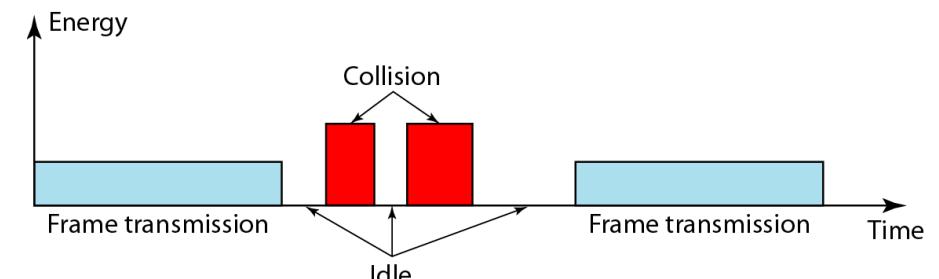
- Use one of the CSMA persistence algorithm  
**(non-persistent, 1-persistent, p-persistent)** for transmission
- If a collision is detected by a station during its transmission then it should do the following:
  - **Abort transmission** and
  - **Transmit a *jam signal* (48 bit)** to notify other stations of collision so that they will **discard the transmitted frame** also to make sure that the collision signal will stay until detected by the furthest station
  - After sending the ***jam signal***, **backoff (wait) for a *random* amount of time**, then
  - Transmit the frame again

# CSMA/CD

- Restrictions of CSMA / CD:
  - Packet **transmission time** should be **at least** as long as the time needed to detect a collision ( $2 * \text{maximum propagation delay} + \text{jam sequence transmission time}$ )
  - Otherwise, CSMA/CD does not have an advantage over CSMA

# Energy Level

- Level of energy in a channel can have three values :
  - Zero
  - Normal
  - Abnormal
- At zero level the channel is idle
- At the normal level, a station has successfully captured the channel and is sending its frame.
- At the abnormal level, there is a collision and the level of the energy is twice the normal level.
- A station that has a frame to send or is sending a frame needs to monitor the energy level to determine if the channel is idle, busy or in collision mode.



# Throughput

- Better than ALOHA.
- Max. throughput is achieved at different value of G and is based on the persistence method.
- For the 1-persistent method, the maximum throughput is around **50 percent** when G=1.
- For the nonpersistent method, the maximum throughput can go up to **90 percent** when G is between 3 and 8.

# CSMA/CA :

- Carrier Sense multiple access with collision avoidance (CSMA/CA) was invented for wireless networks.
- In CSMA/CA: three strategy are applied:
  - Interframe Space (IFS)
  - Contention Window
  - Acknowledgement

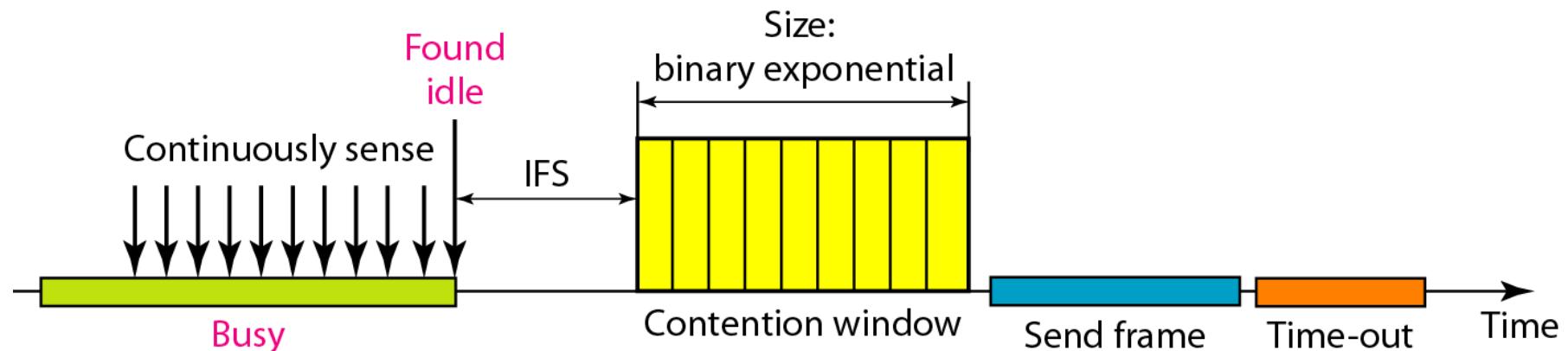
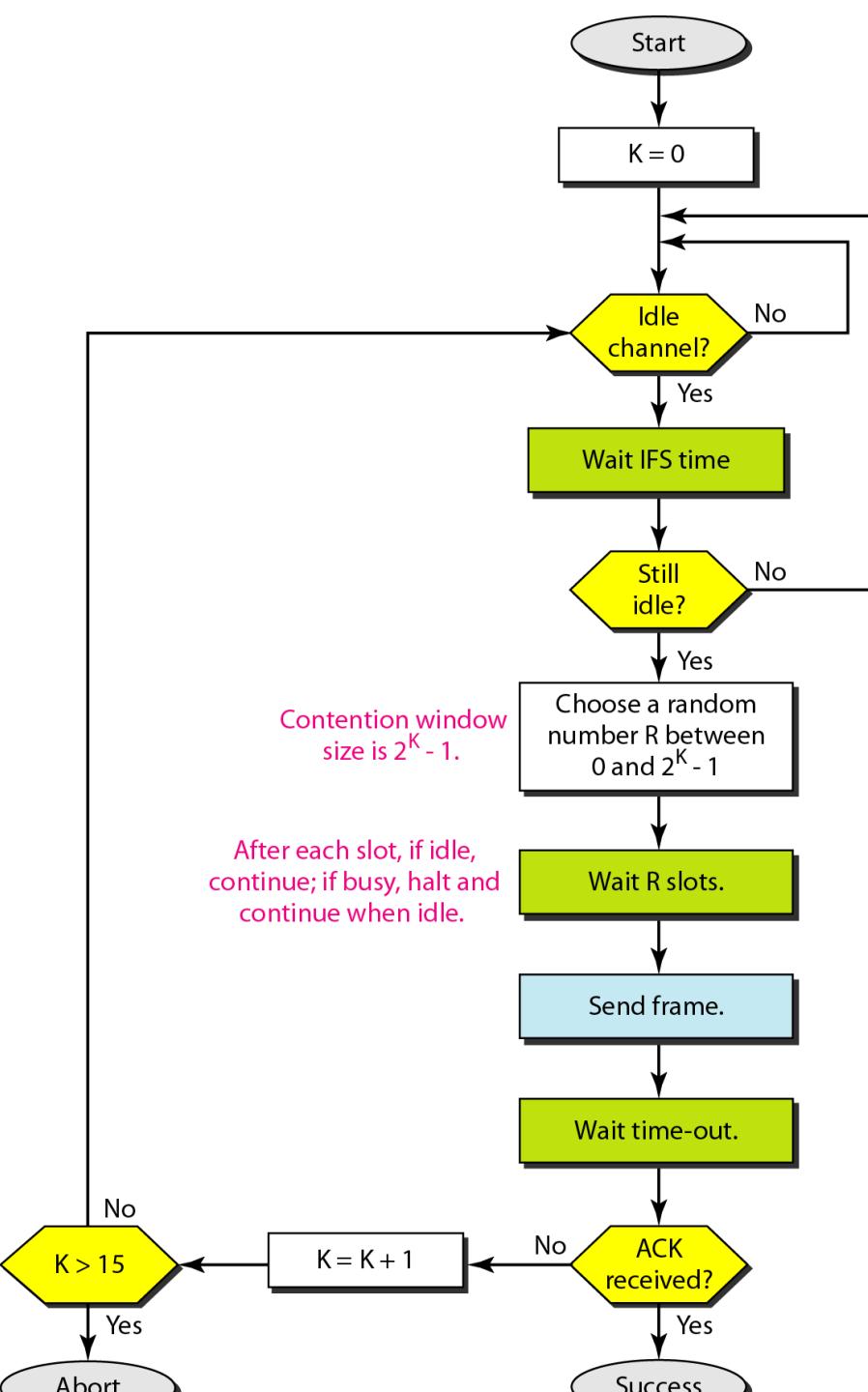


Figure 12.17 Flow diagram for CSMA/CA



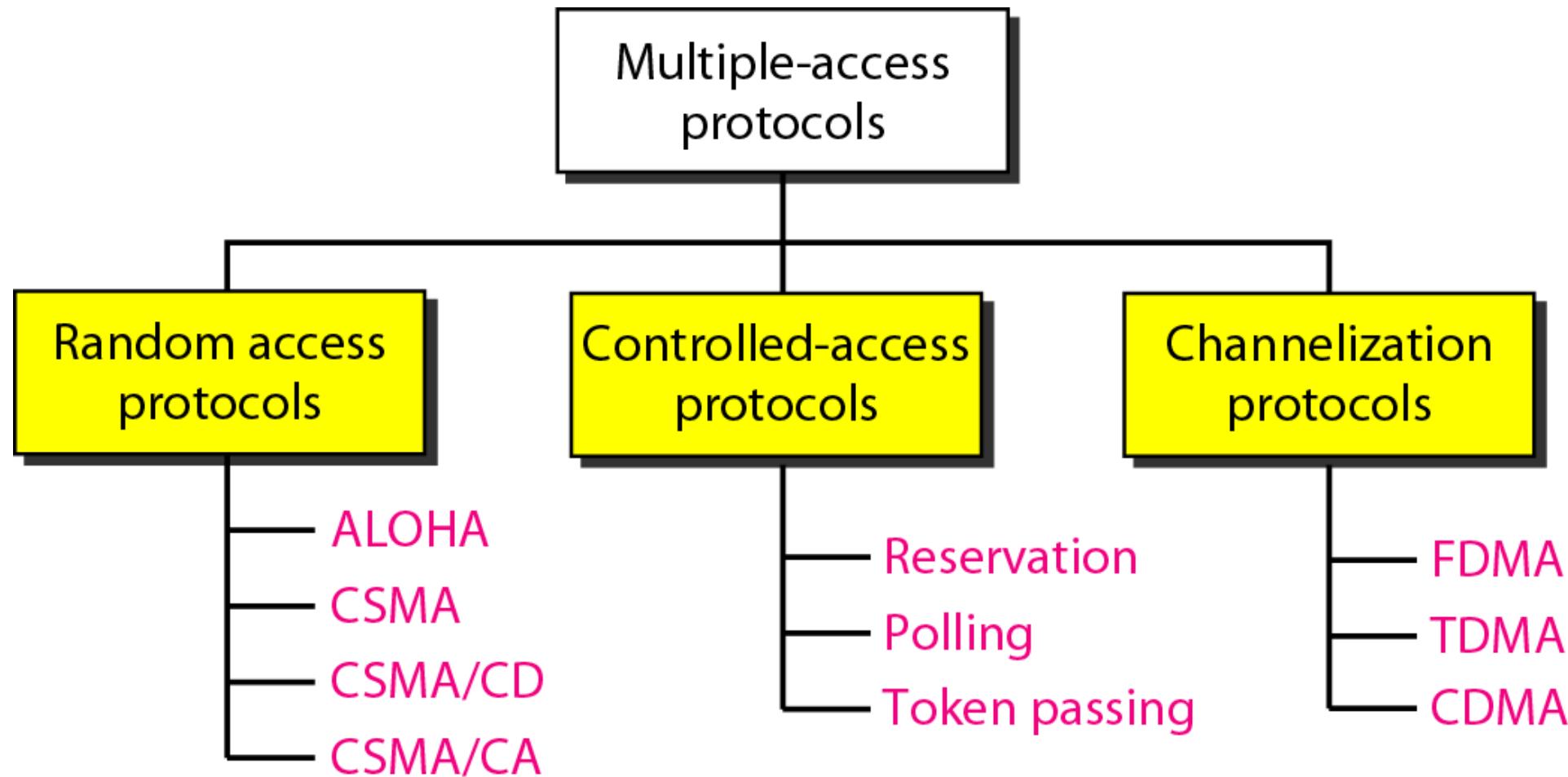
Channel idle? Don't transmit yet!  
Wait IFS time.

Still idle after IFS? Don't transmit yet!  
Now in Contention Window.  
Choose random number and wait that many slots.

Did you wait  $R$  slots and all slots were available? Go ahead, transmit.  
Now, wait time-out for a response.

(How big is a slot? 50 µs in 802.11 FH  
and  
20 µs in 802.11 DS)

**Figure 12.2** *Taxonomy of multiple-access protocols discussed in this chapter*



## 13.2 Controlled Access or Scheduling

- the stations **consult one another** to find which station has the right to send.
- A station cannot send unless it has been authorized by other stations.
- **Three methods** for controlled access:
  - Reservation
  - Polling
  - Token Passing

## 1-Reservation access method

- Stations needs to make a reservation before sending data.

- **Time is divided** into intervals.

- Reservation frame is sent.

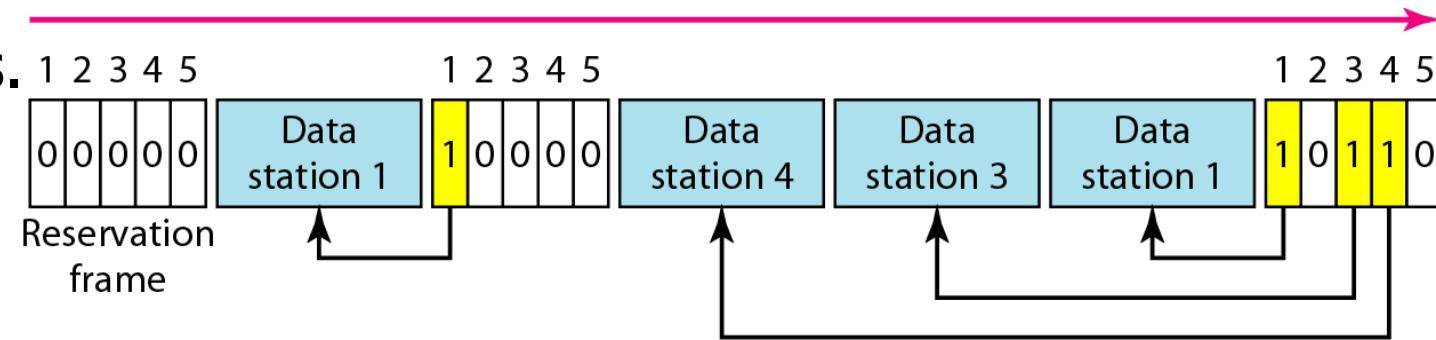
- It changes the value 0 to 1.

- If there are N stations in the system, there are exactly N reservation minislots in the reservation frame.

- Each minislots belongs to a station.

- When a station needs to send a data frame, it makes a **reservation** in its own minslot.

- The stations that made reservations can send their data frames after the reservation frame.

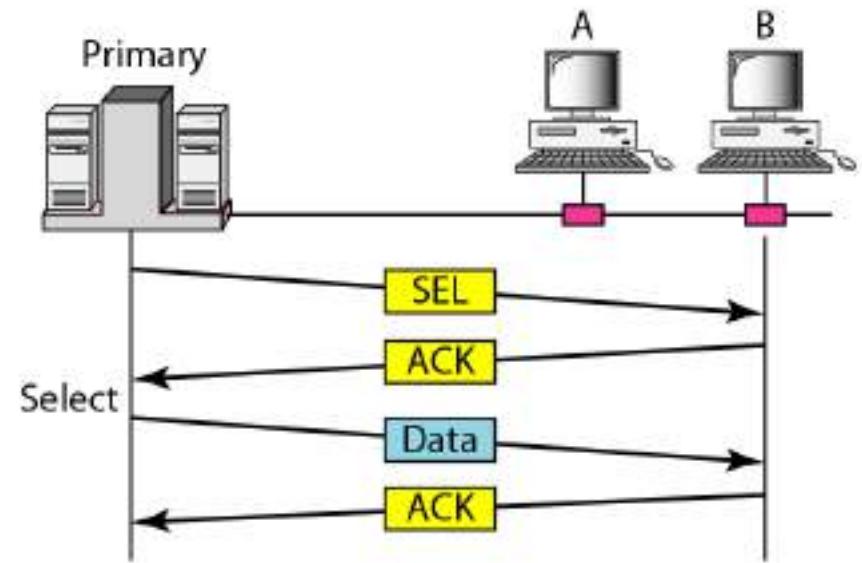


## 2- Polling

- Polling works with topologies in which one device is designated as a **primary station** and the other devices are **secondary stations**.
- All **data exchanges** must be made through the primary device even when the ultimate destination is a secondary device.
- The **primary devices controls the link**; the secondary devices follow its instruction.
- It is **upto the primary device to determine which device is allowed to use the channel at a given time**.

**Figure 12.19** Select and poll functions in polling access method

- The **select function** is used whenever the primary device has something to **send**.
- Primary controls the link
- Primary must alert the secondary to the upcoming transmission and wait for an acknowledgment of the secondary's ready status.
- Before sending data, the primary creates and transmits a select (**SEL**) frame, one field of which includes the address of the intended secondary.



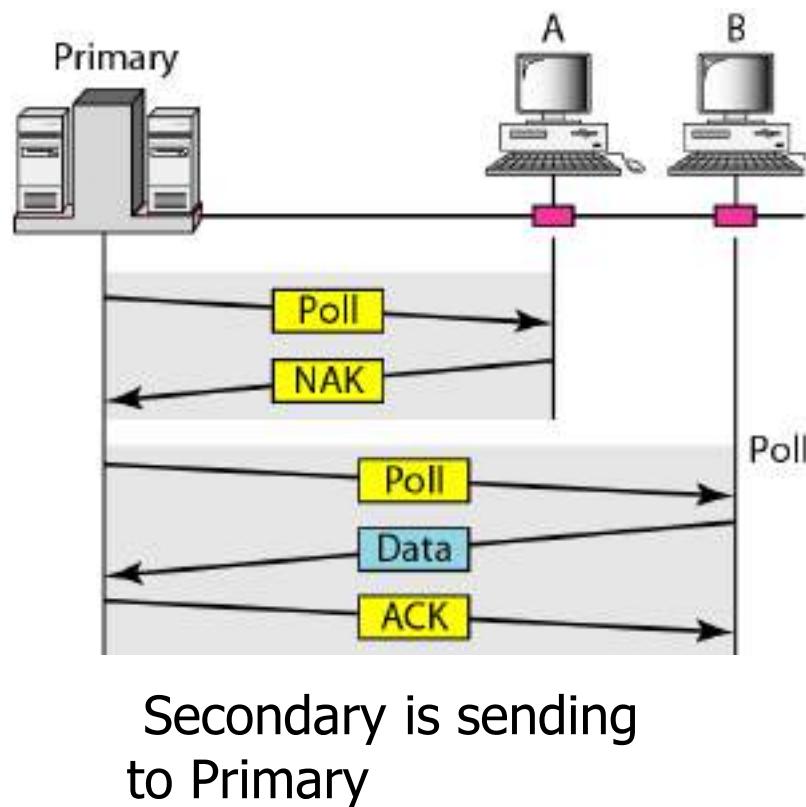
Primary is sending to Secondary

---

**Figure 12.19** *Select and poll functions in polling access method*

---

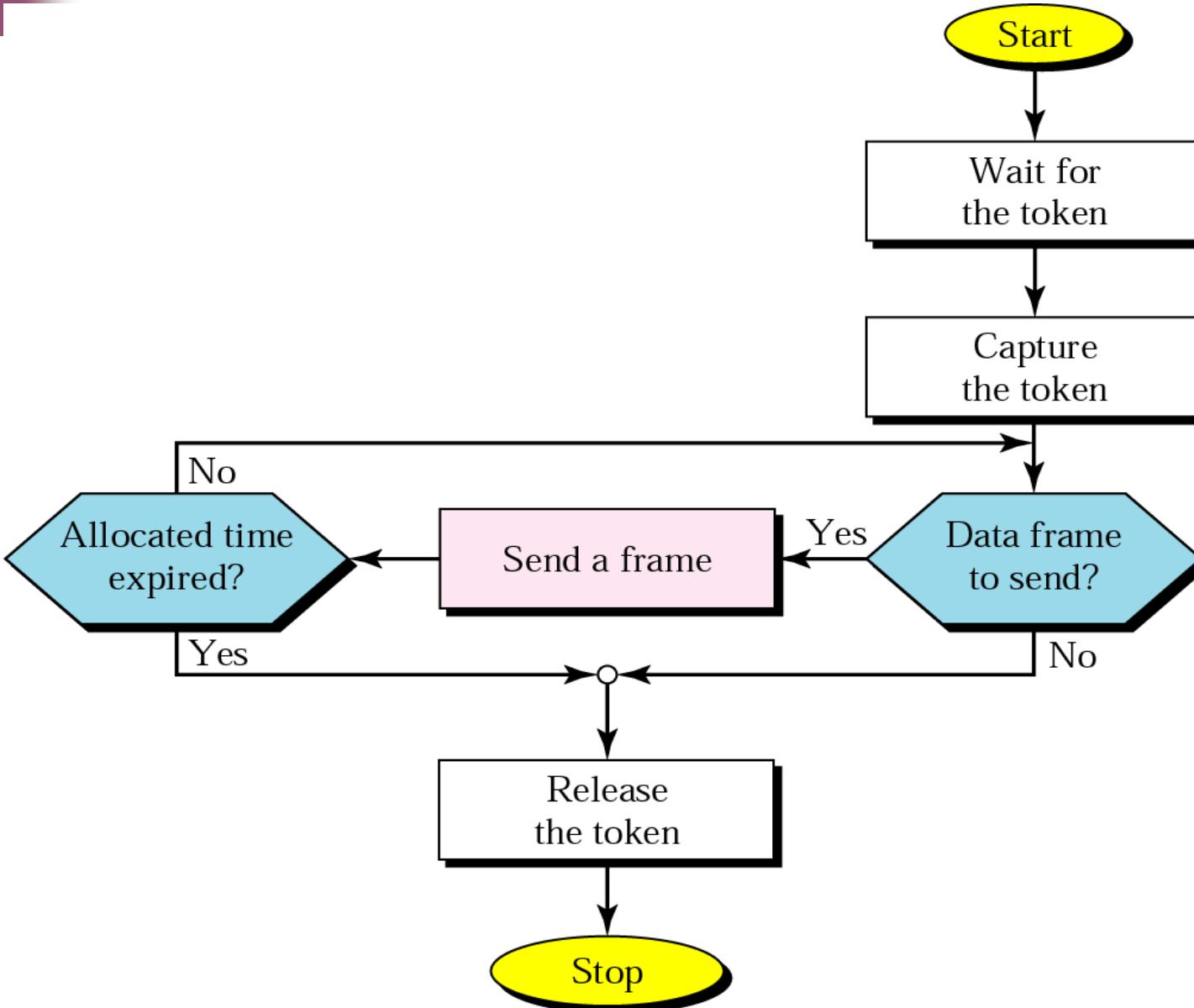
- The poll function is used by the primary device to solicit (ask for) transmissions from the secondary devices.
- When primary is ready to receive data, it must ask (**poll**) each device in turn if it has anything to send.
- When the first secondary is approached, it responds either with a NAK frame if it has nothing to send or with data.



# Token Passing

- In the token-passing method, the stations in a network are organized in a **logical ring**.
- For each station, there is a **predecessor** and a **successor**.
- The **current station is the one** that is accessing the channel now.
- The right to this access has been **passed from the predecessor** to the current station.
- The right will **be passed to the successor** when the current station has no more data to send.
- A special **packet called a token circulates** through the ring.

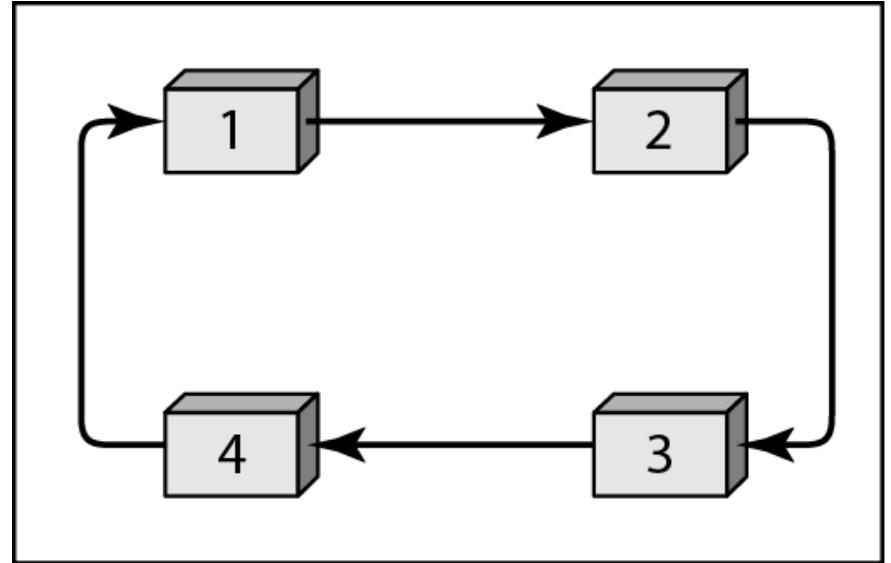
**Figure 13.13** Token-passing procedure



- The station cannot send data until it receives the token again in the next round.
- Logical Ring
- Dual ring
- Bus ring
- Star Ring

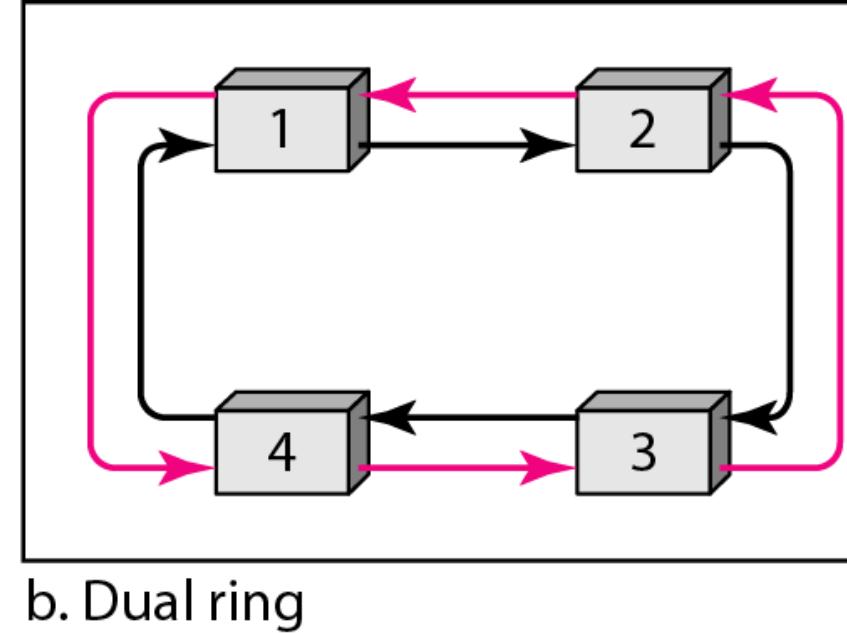
# Logical ring

- In the logical ring topology, when a station sends the token to its successor,
- **Token cannot be seen by other stations.**
- The successor is the next one in line.
- **Means that the token does not have** to have the address of the next successor.
- The problem with the topology is that **if one of the links fail; the whole system is fails**



# Dual ring

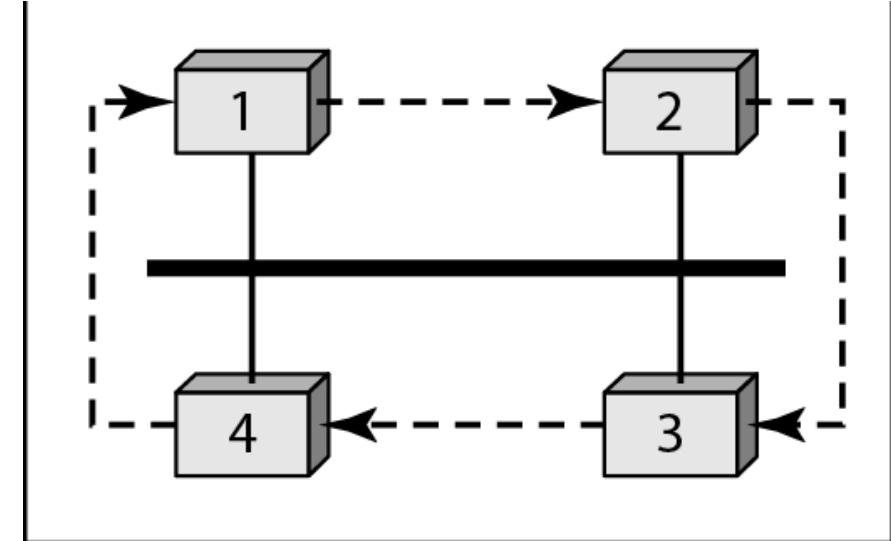
- Uses second (auxiliary ring) which operates in the reverse direction compared with the main ring.
- The high-speed Token Ring networks called
  - FDDI (Fiber Distributed Data Interface)
  - CDDI (Copper Distributed Data Interface) use this topology.



b. Dual ring

# Bus ring

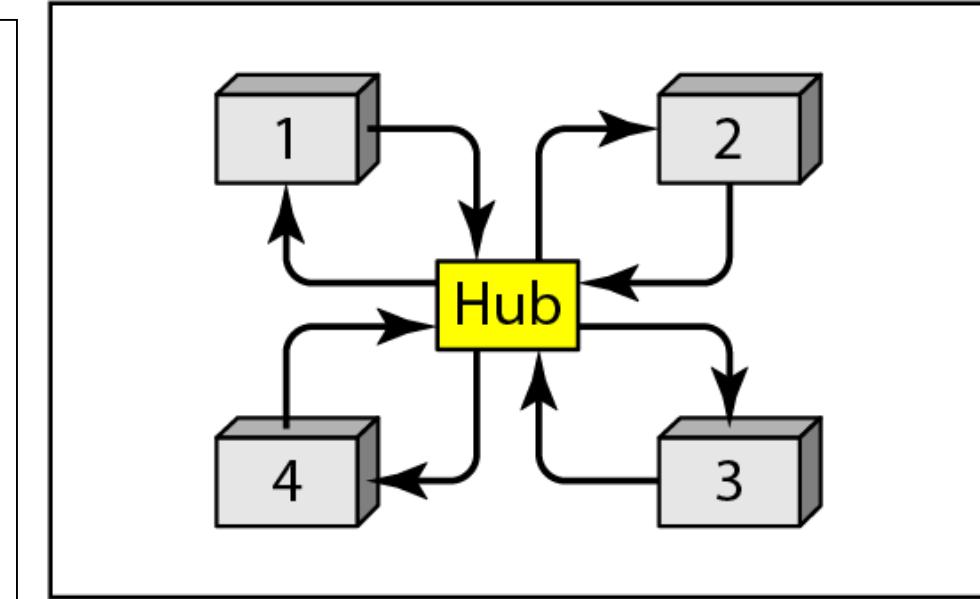
- Also called a token bus
- The stations are connected to a **single cable called a bus**.
- They, however make a logical ring, because each station knows the **address of its successor** (and also predecessor for token management purposes.)
- When a **station has finished sending its data**, it releases the token and **inserts the address of its successor in the token**.
- Only the station with the address matching the destination address of the token gets the token to access the shared media.



c. Bus ring

# Star Ring

- The physical topology is a star.
- There is a hub, however that acts as the connector.
- The wiring inside the hub makes the ring; that stations are connected to this ring through the two wire connections.
- Less chance of network failure.
- Also adding and removing stations from the ring is easier.



d. Star ring

## 12-3 CHANNELIZATION

*Channelization is a multiple-access method in which the available bandwidth of a link is shared in time, frequency, or through code, between different stations. In this section, we discuss three channelization protocols.*

*Topics discussed in this section:*

Frequency-Division Multiple Access (FDMA)

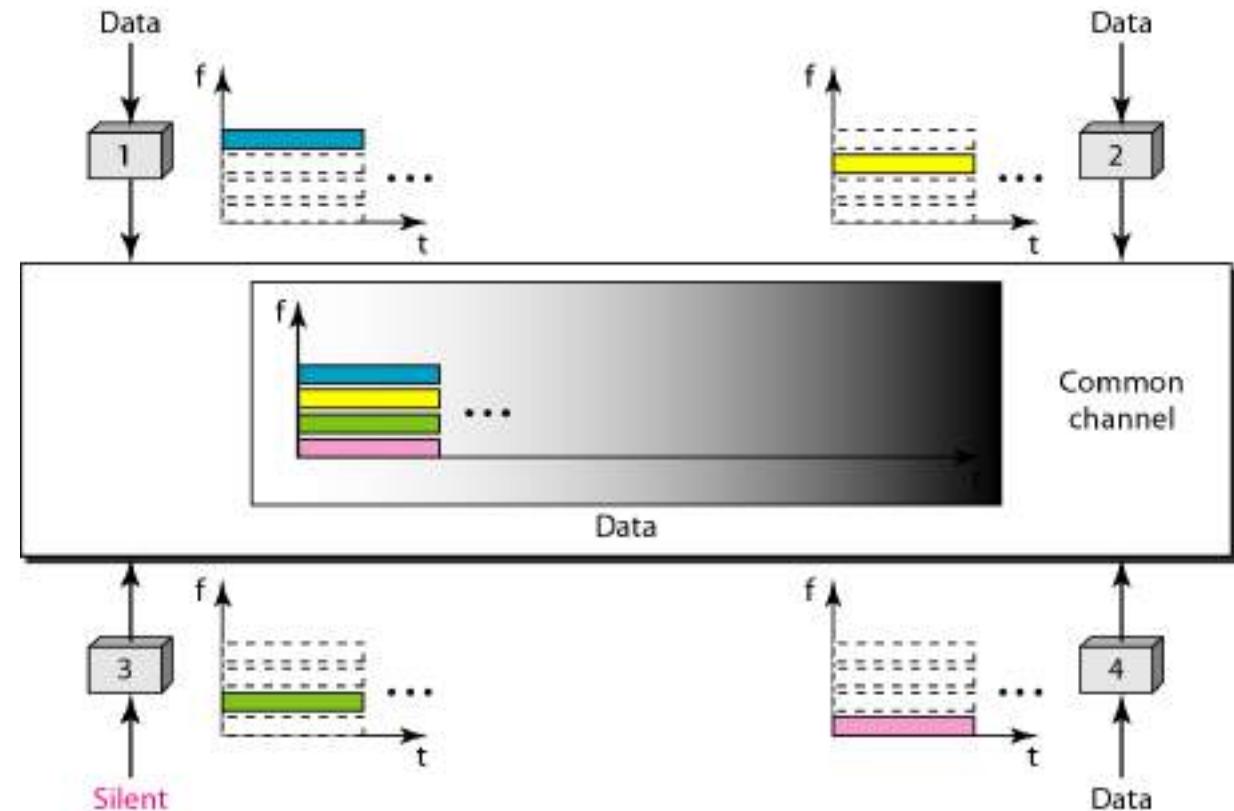
Time-Division Multiple Access (TDMA)

Code-Division Multiple Access (CDMA)

**Figure 12.21 Frequency-division multiple access (FDMA)**

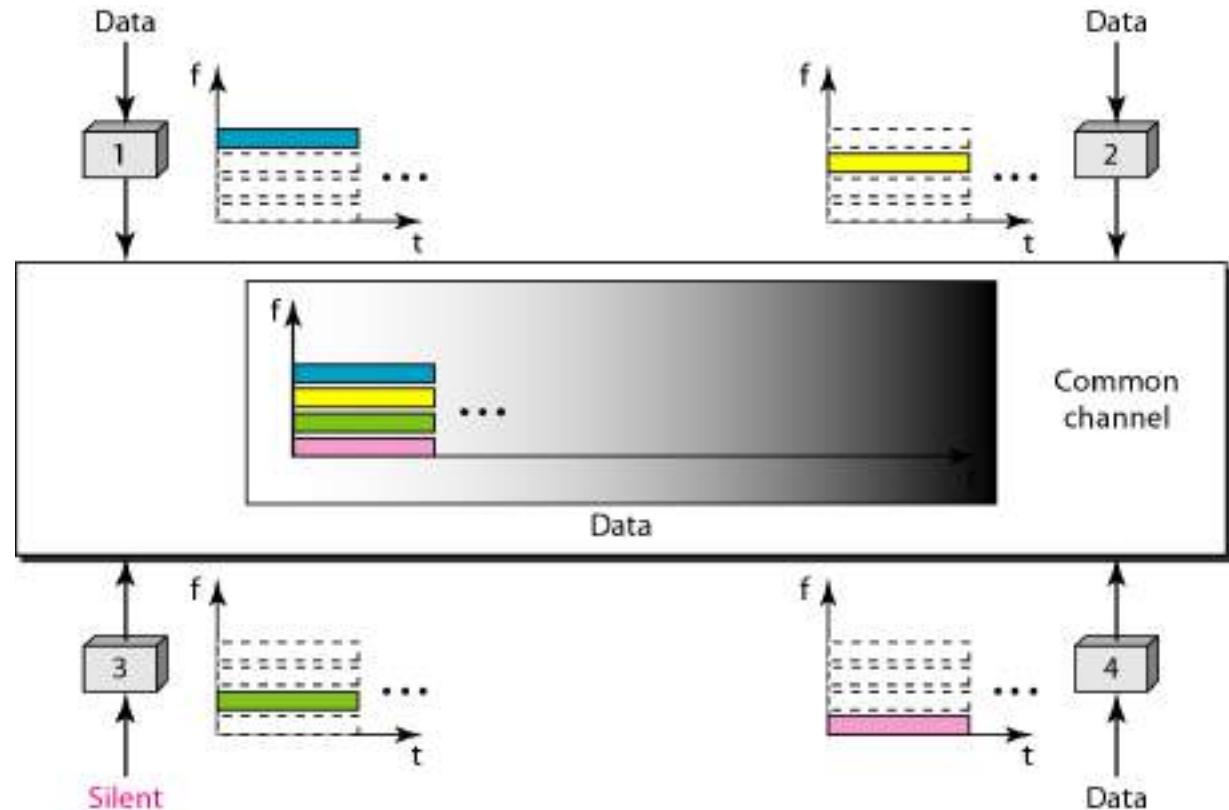
**FDMA: Frequency Division Multiple Access:**

- Transmission medium is divided into  $M$  separate frequency bands
- Each station is **allocated a band to send its data**. i.e each band is reserved for a specific station, and it belongs to the station all the time.
- **Each station also uses a bandpass filter** to confine the transmitter frequencies. To prevent station interferences, the allocated bands are separated from one another by **small guard bands**.



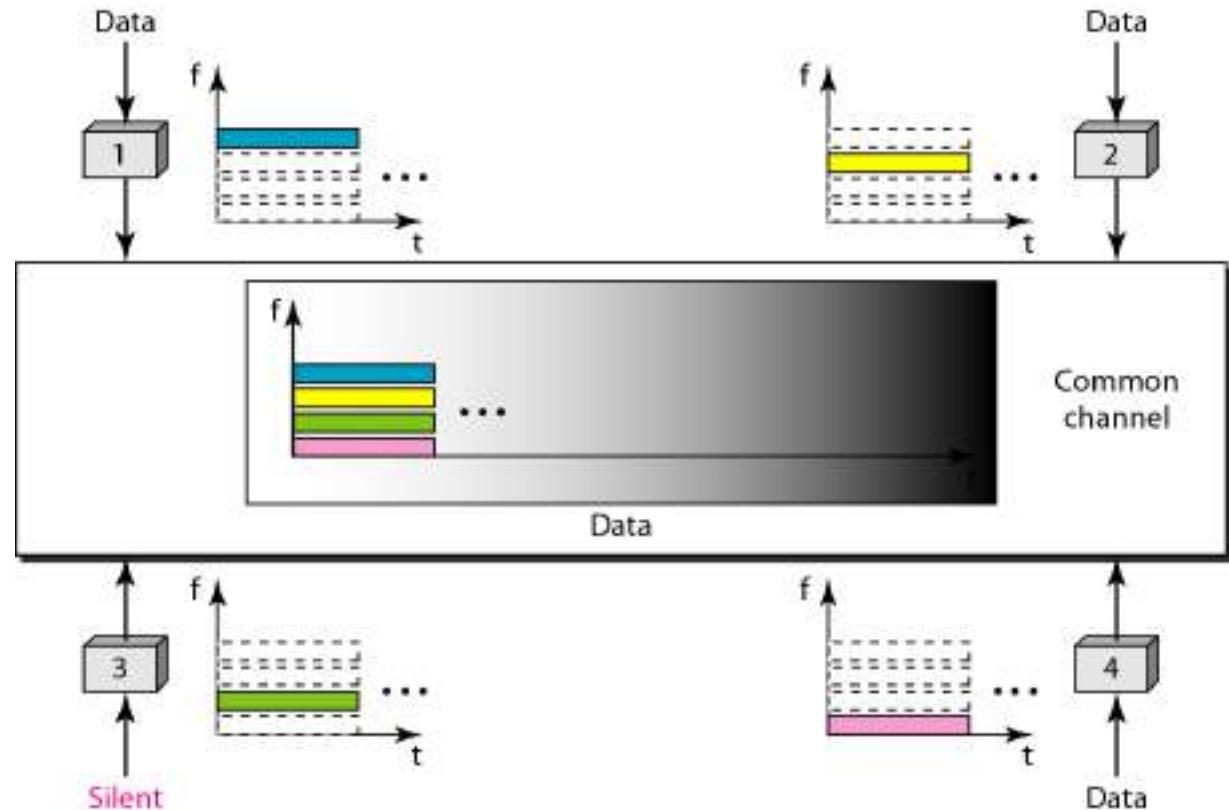
**Figure 12.21 Frequency-division multiple access (FDMA)**

- FDMA, is an access method in the data link layer.
- The data link layer in each **station tells its physical layer to make a bandpass signal from the data passed to it**.
- The signal must be created in the allocated band.



**Figure 12.21 Frequency-division multiple access (FDMA)**

- Although FDMA and **frequency-division multiplexing (FDM)** conceptually seem similar, there are differences between them.
- **FDM is a physical layer technique** that combines the loads from **low bandwidth channels** and transmits them by using **a high-bandwidth channel**.



FDMA allows multiple users to send data through a single communication channel, such as a coaxial cable or microwave beam, by dividing the bandwidth of the channel into separate non-overlapping frequency sub-channels and allocating each sub-channel to a separate user.

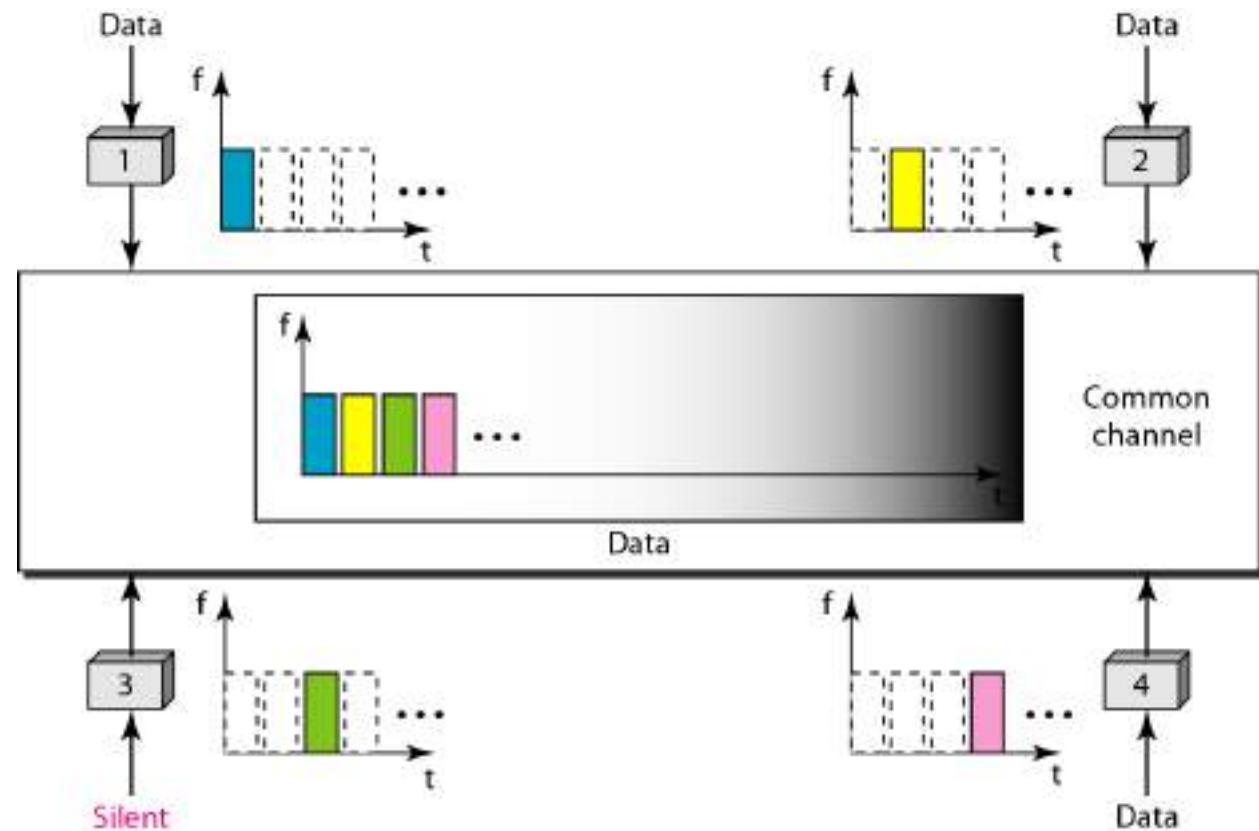
Users can send data through a subchannel by modulating it on a carrier wave at the subchannel's frequency. It is used in satellite communication systems and telephone trunklines.

**Figure 12.22 Time-division multiple access (TDMA)**

## TDMA: Time Division Multiple Access

The entire bandwidth capacity is a **single channel** with its capacity shared **in time** between **M** stations

In TDMA, the bandwidth is just one channel that is timeshared between different stations.



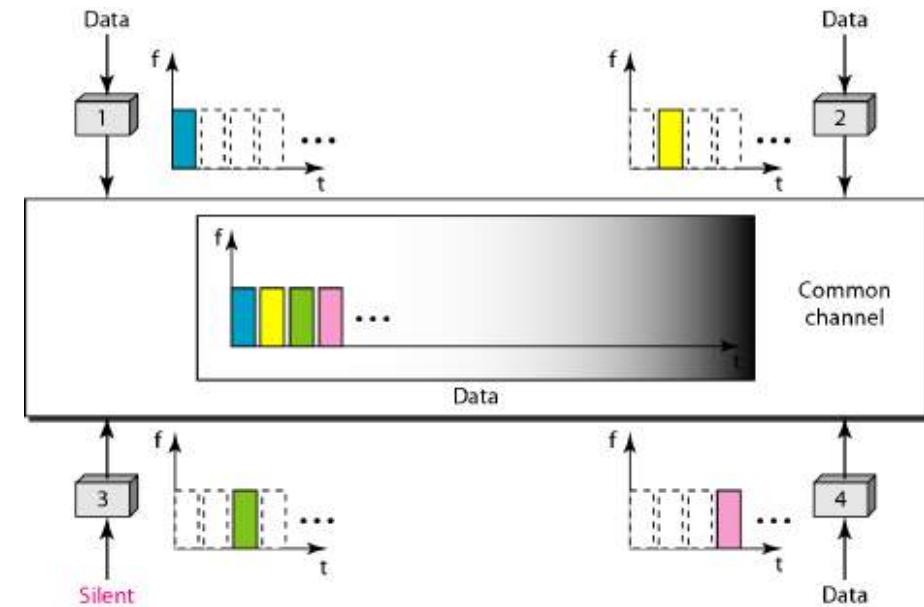
**Figure 12.22 Time-division multiple access (TDMA)**

The main problem with **TDMA** lies in achieving **synchronization** between the different stations. Each station needs to know the beginning of its slot and location of its lot.

This may be difficult **because of propagation delays introduced in the system** if the stations are spread over a large area.

To compensate for the delays, we can insert **guard times**.

Synchronization is normally accomplished by having some synchronization bits at the beginning of each slot.



---

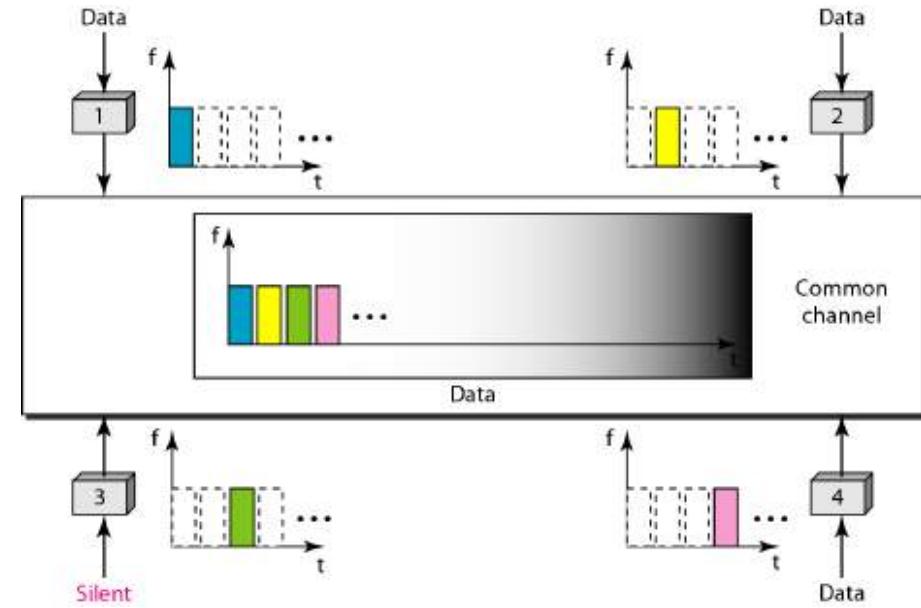
**Figure 12.22 Time-division multiple access (TDMA)**

---

We need to emphasize that although TDMA and **time-division multiplexing (TDM)** conceptually seems the same, there are differences between them.

In TDM, as a physical layer technique that combines the data from **slower channels** and transmits them by using a **faster channel**.

TDMA, on the other hand, is an access method in the data-link layer. **The data-link layer in each station tells its physical layer** to use the allocated time slot.



## **TDMA: Time Division Multiple Access**

TDMA is used in the digital 2G cellular systems such as Global System for Mobile Communications (GSM), IS-136, Personal Digital Cellular (PDC) and iDEN, and in the Digital Enhanced Cordless Telecommunications (DECT) standard for portable phones.

TDMA was first used in satellite communication systems by Western Union in its Westar 3 communications satellite in 1979.

## 12-3 CHANNELIZATION - CDMA

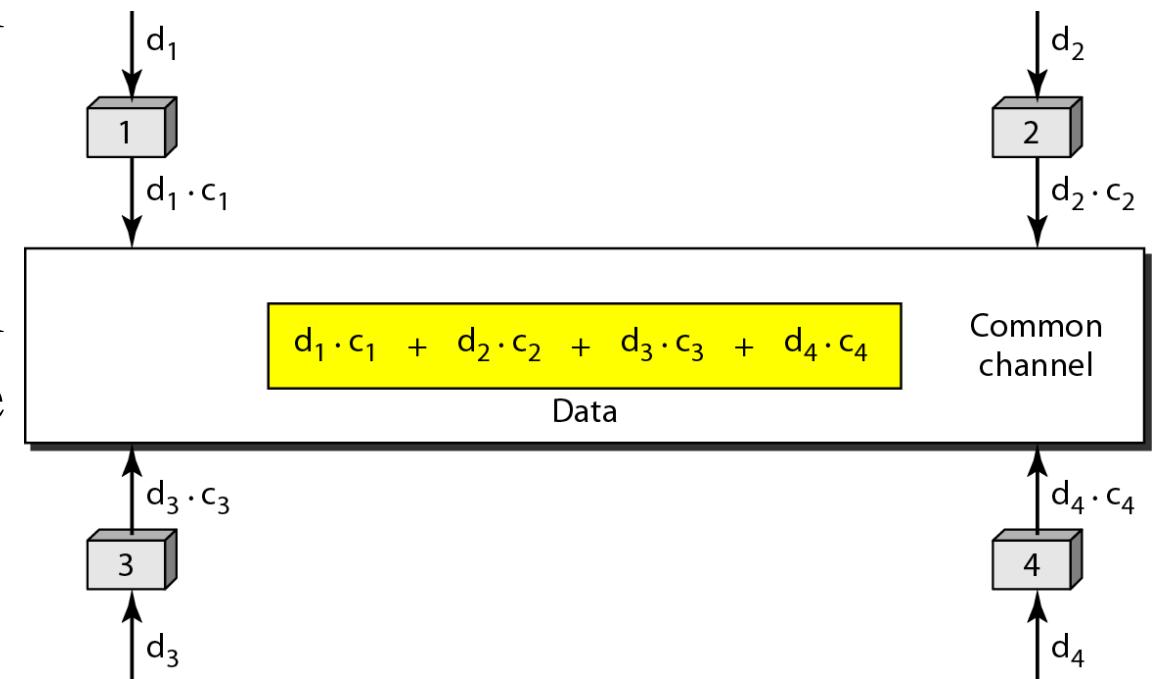
### ■ CDMA: Code Division Multiple Access

- In CDMA, one channel carries all transmissions **simultaneously**

- Each station **codes its data** signal by a specific codes before transmission.

- The stations receivers use these codes to recover the data for the desired station

Language : Chinese speaker and Nepali speaker



# Idea

- Let us assume we have four stations 1,2,3, 4 connected to the same channel.
- The data from the station 1 are  $d_1$ , from station 2 are  $d_2$  and so on.
- Two properties

## Code-Division Multiple Access (CDMA)

$$\mathbf{C}_x^* \mathbf{C}_y = 0$$

$$\mathbf{C}_x^* \mathbf{C}_x = N$$

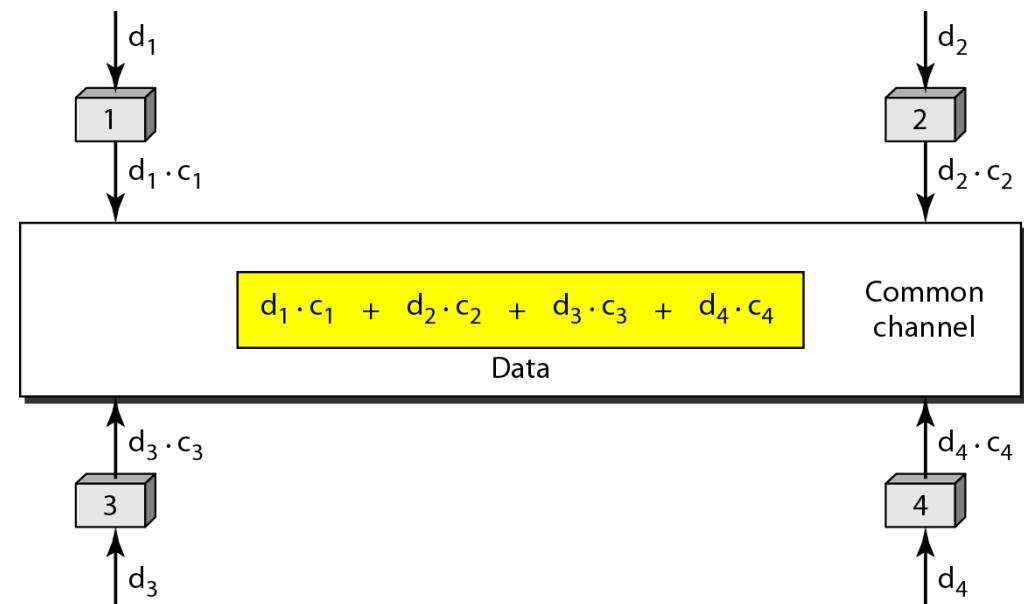
- That means  $c_1 \cdot c_1 = 4$  but  $c_2 \cdot c_1$ ,  $c_3 \cdot c_1$  and  $c_4 \cdot c_1$  are all 0

# Idea

- Suppose stations 1 and 2 are talking to each other. **Station 2 wants** to hear what **station 1 is saying**. It multiplies the data on the channel by  $c_1$ , the code of station 1.
- The obtained result is divided by the number of station (i.e 4) to get the data from station 1.

$$\text{■ } Data = \frac{(d_1 \cdot c_1 + d_2 \cdot c_2 + d_3 \cdot c_3 + d_4 \cdot c_4) \cdot c_1}{4}$$

$$\begin{aligned} &= \frac{d_1 \cdot c_1 \cdot c_1 + d_2 \cdot c_2 \cdot c_1 + d_3 \cdot c_3 \cdot c_1 + d_4 \cdot c_4 \cdot c_1}{4} \\ &= \frac{4 d_1 + 0 + 0 + 0}{4} \\ &= d_1 \end{aligned}$$



# Chips

CDMA is based on coding theory. Each station is assigned a code, which is a sequence of numbers called **chips**,

## Example: Chip sequences

$c_1$	$c_2$	$c_3$	$c_4$
[+1 +1 +1 +1]	[+1 -1 +1 -1]	[+1 +1 -1 -1]	[+1 -1 -1 +1]

These sequences number are not selected randomly, they were carefully selected. They are called **orthogonal sequences**.

**They are called orthogonal sequences and having following properties**

1. Each sequence is made of  **$N$  elements**, where  **$N$  is the number of stations.**
2. If we multiply a sequence by a number, every element in the sequence is multiplied by that element. This is called multiplication of a sequence by a scalar.

For example, 2.  $[+1 +1-1-1] = [+2+2-2-2]$

3. If we multiply two equal sequences, element by element, and add the results, we get  $N$ , where  $N$  is the number of elements in the each sequence. **This is called the inner product of two equal sequences.**

For example,  $[+1 +1-1 -1] \cdot [+1 +1 -1 -1] = 1 + 1 + 1 + 1 = 4$

4. If we multiply two different sequences, element by element, and add the results, we get 0.

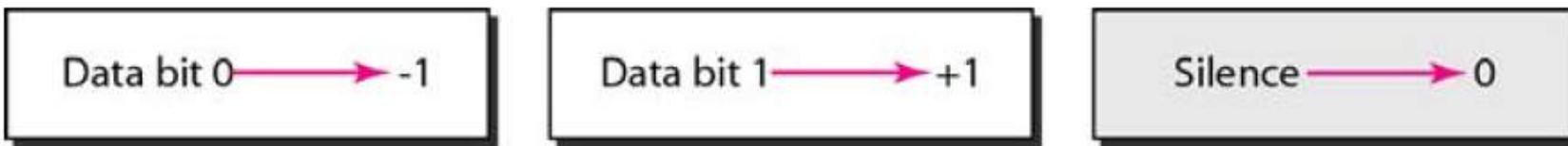
This is called inner product of two different sequences. For example,

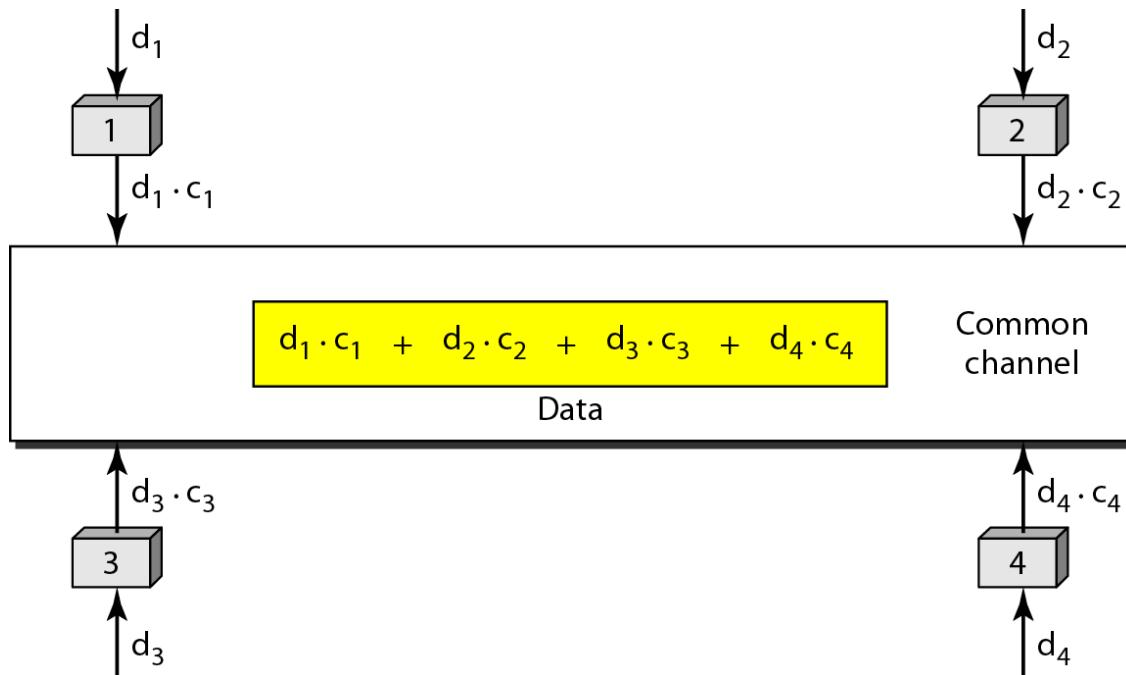
$$[+1 +1 -1 -1] \bullet [+1 +1 +1 +1] = 1 + 1 - 1 - 1 = 0$$

5. Adding two sequences means adding the corresponding elements. The result is another sequence. For example,

$$[+1+1-1-1]+ [+1+1+1+1]=[+2+2\ 0\ 0]$$

**Figure 12.25** Data representation in CDMA





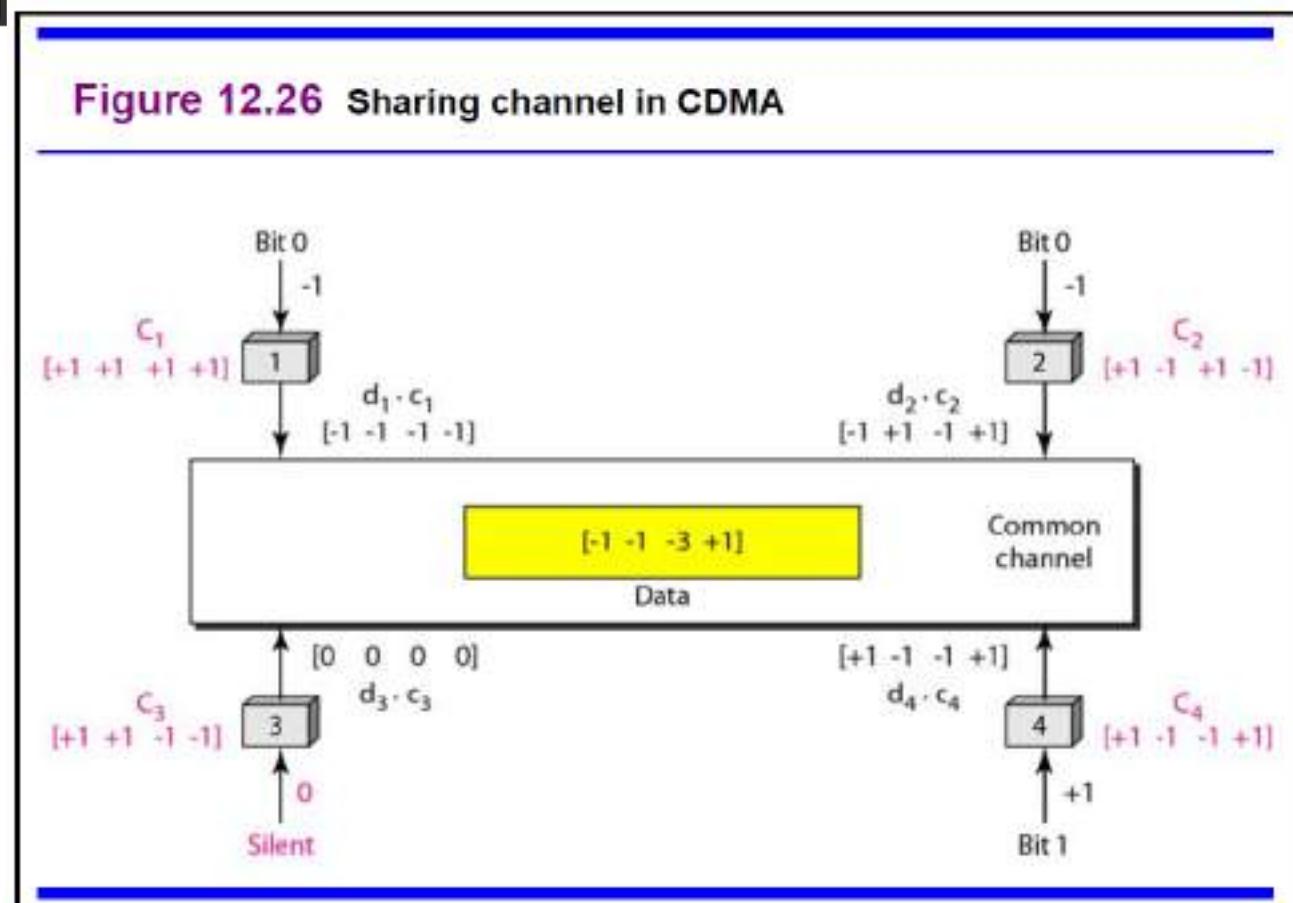
Let us assume,

Station 1 is sending a 0 bit  $\rightarrow -1$

Station 2 is sending a 0 bit  $\rightarrow -1$

Station 4 is sending a 1 bit  $\rightarrow +1$

Station 3 is silent  $\rightarrow 0$



Let us assume,

Station 3, which is said to be silent is listening to station 2.

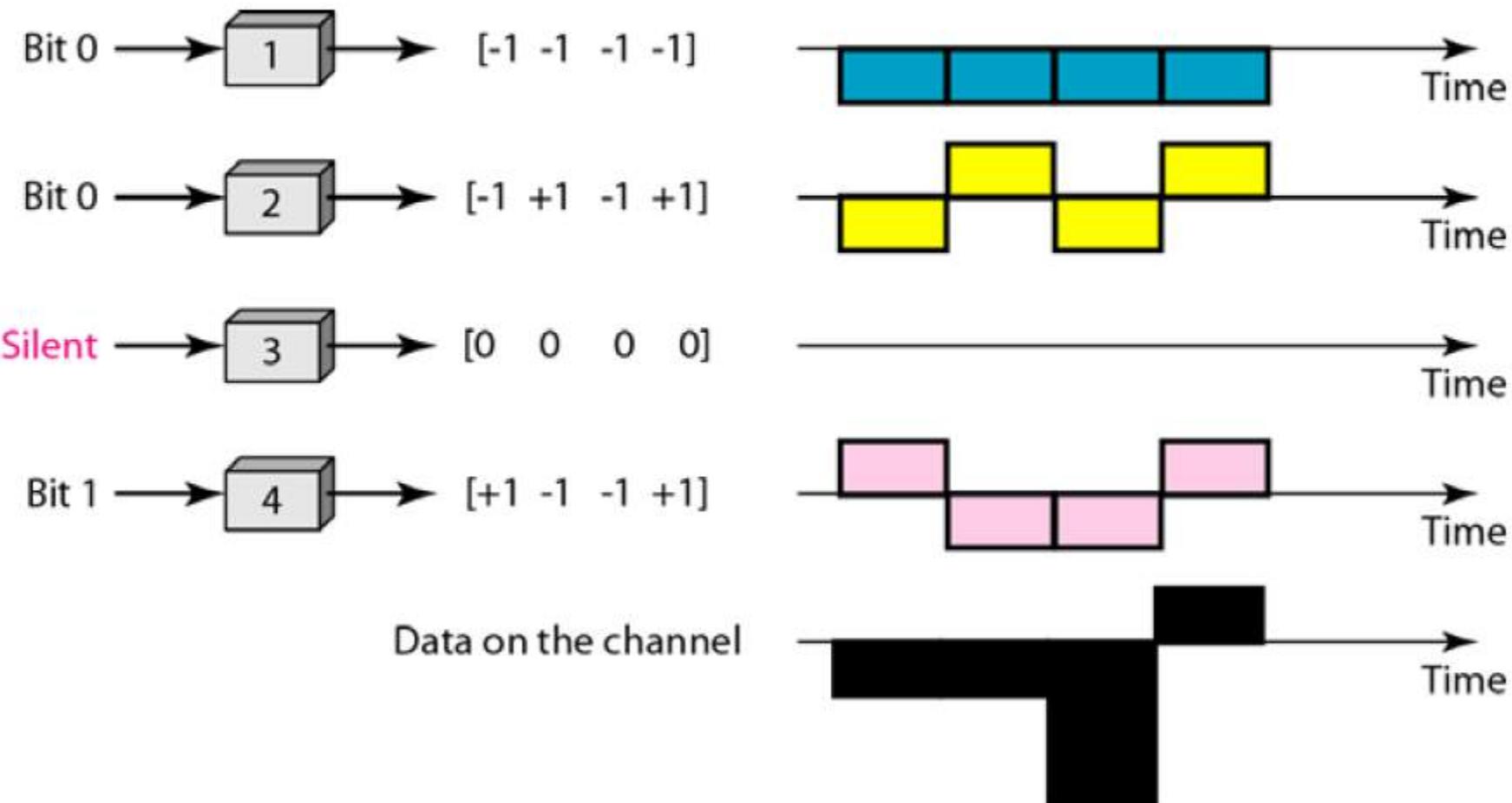
Station 3 multiplies the total data on the channel by the code for station 2 which is  $[+1 -1 +1 -1]$

To get

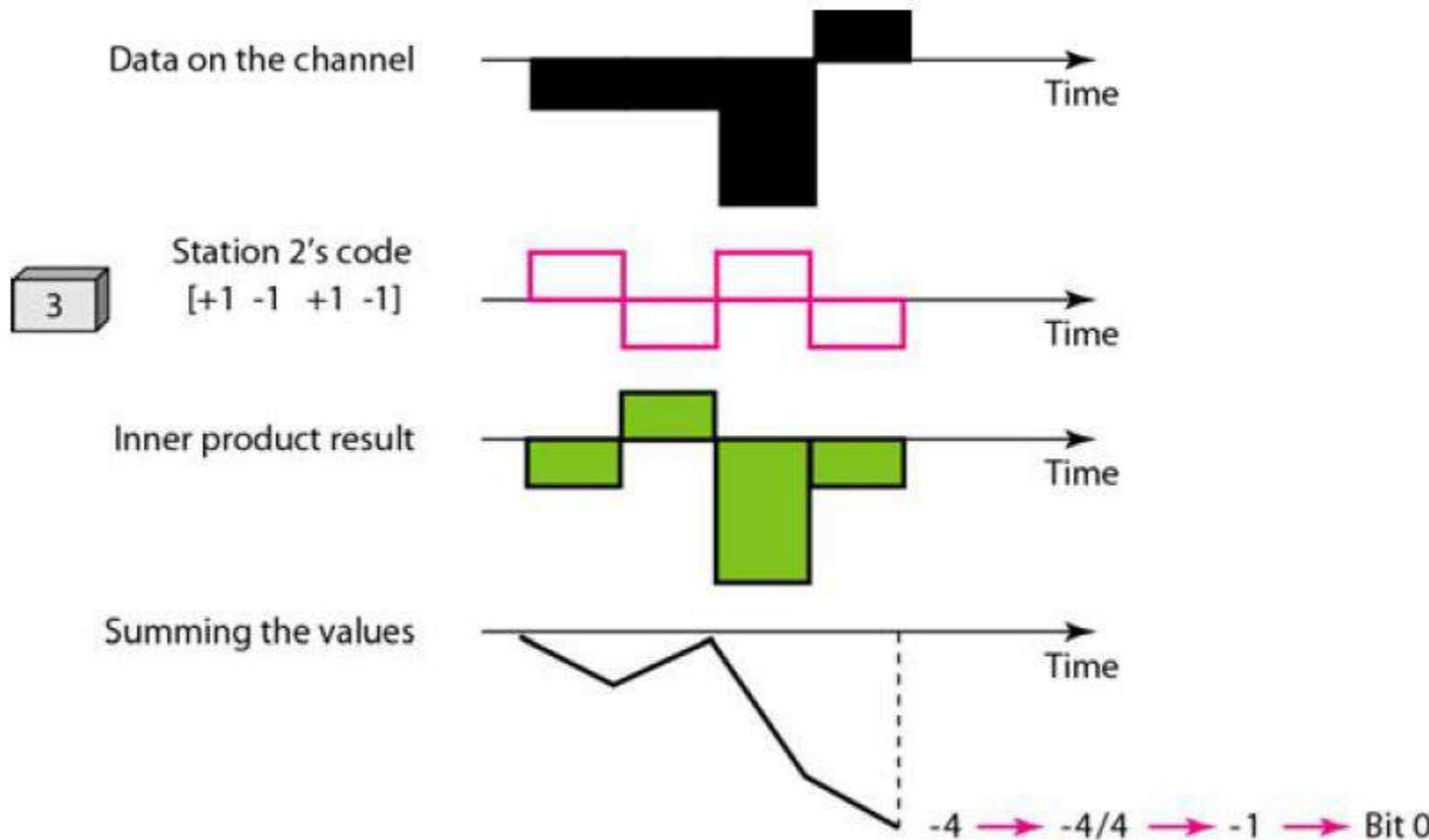
$$[-1 -1 -3 +1] \cdot [+1 -1 +1 -1] = -4/4$$

$$= -1 \rightarrow \text{bit 0}$$

**Figure 12.27** Digital signal created by four stations in CDMA



**Figure 12.28 Decoding of the composite signal for one in CDMA**



# Sequence Generation

- To generate chip sequences, we use a **Walsh table** which is a **two-dimensional table** with an equal number of rows and columns.
- According to Walsh, if we know the table of  $N$  sequences  $W_N$ , we can create the table for  $2N$  sequences  $W_{2N}$
- After  $W_2$  is generated,  $W_4$  can be made of four  $W_2$ s, with the last one the complement of  $W_2$ .
- Similarly,  $W_8$  is composed of four  $W_4$ s, and so on.

Figure 12.29 General rule and examples of creating Walsh tables

a. Two basic rules

$$W_1 = [ +1 ]$$
$$W_{2N} = \begin{bmatrix} W_N & W_N \\ W_N & \overline{W_N} \end{bmatrix}$$

b. Generation of  $W_1$ ,  $W_2$ , and  $W_4$

$$W_1 = [ +1 ]$$
$$W_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix}$$
$$W_4 = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}$$

**Figure 12.29** General rule and examples of creating Walsh tables

$$W_1 = \begin{bmatrix} +1 \end{bmatrix}$$

$$W_{2N} = \begin{bmatrix} W_N & W_N \\ W_N & \overline{W}_N \end{bmatrix}$$

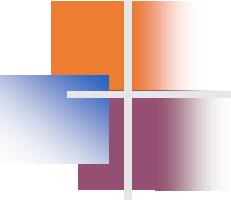
a. Two basic rules

$$W_1 = \begin{bmatrix} +1 \end{bmatrix}$$

$$W_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix}$$

$$W_4 = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}$$

b. Generation of  $W_1$ ,  $W_2$ , and  $W_4$

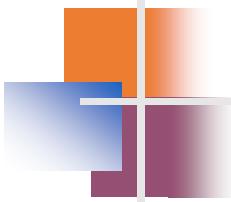


Note

---

The number of sequences in a Walsh table  
needs to be  $N = 2^m$ .

---



## Example 12.6

Find the chips for a network with

- a. Two stations
- b. Four stations

### Solution

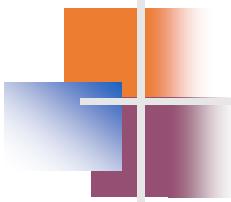
We can use the rows of  $W_2$  and  $W_4$  in Figure 12.29:

a. For a two-station network, we have

$$[+1 \ 1] \text{ and } [+1 \ -1].$$

b. For a four-station network we have

$$[+1 \ +1 \ +1 \ +1], \ [+1 \ -1 \ +1 \ -1], \\ [+1 \ +1 \ -1 \ -1], \text{ and } [+1 \ -1 \ -1 \ +1].$$

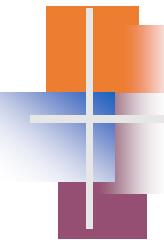


## Example 12.7

**What is the number of sequences if we have 90 stations in our network?**

### Solution

The number of sequences needs to be  $2^m$ . We need to choose  $m = 7$  and  $N = 2^7$  or 128. We can then use 90 of the sequences as the chips.



## Example 12.8

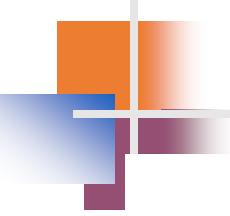
**Prove that a receiving station can get the data sent by a specific sender if it multiplies the entire data on the channel by the sender's chip code and then divides it by the number of stations.**

### Solution

Let us prove this for the first station, using our previous four-station example. We can say that the data on the channel

$$D = (d_1 \cdot c_1 + d_2 \cdot c_2 + d_3 \cdot c_3 + d_4 \cdot c_4).$$

The receiver which wants to get the data sent by station 1 multiplies these data by  $c_1$ .



## Example 12.8 (continued)

$$\begin{aligned}D \cdot c_1 &= (d_1 \cdot c_1 + d_2 \cdot c_2 + d_3 \cdot c_3 + d_4 \cdot c_4) \cdot c_1 \\&= d_1 \cdot c_1 \cdot c_1 + d_2 \cdot c_2 \cdot c_1 + d_3 \cdot c_3 \cdot c_1 + d_4 \cdot c_4 \cdot c_1 \\&= d_1 \times N + d_2 \times 0 + d_3 \times 0 + d_4 \times 0 \\&= d_1 \times N\end{aligned}$$

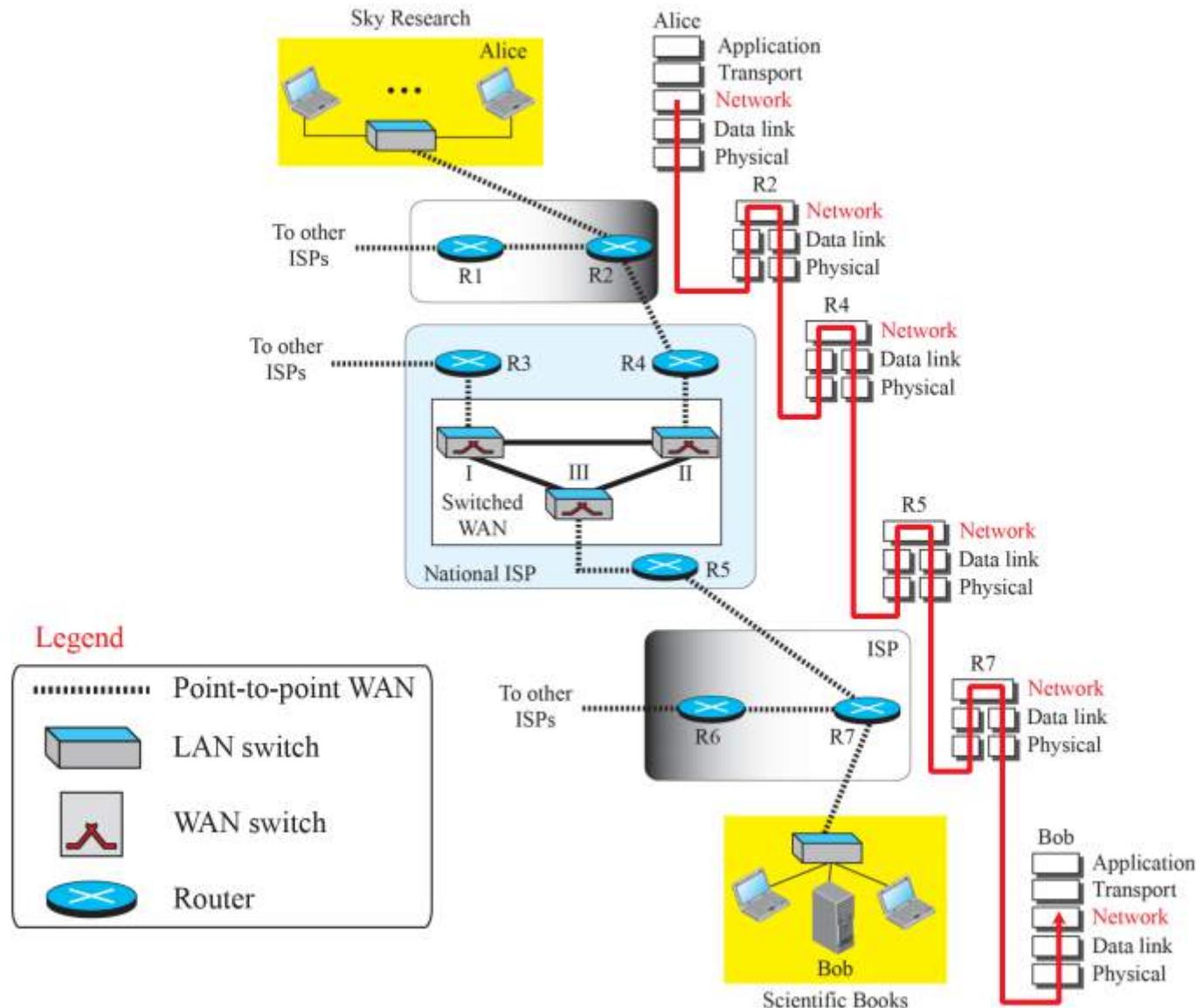
When we divide the result by N, we get d<sub>1</sub>.

# Network Layer

# Introduction to Network Layer

- Concerned with getting packets from the source to the destination.
- Lowest layer that deals with end to end transmission.
- To achieve this goal, network must know about the topology of the communication subnet. (i.e. set of all routers) and choose appropriate path for it.

## ***Communication at the network layer***



# Packetizing

- **Encapsulating** the payload (data received from upper layer) in a network-layer packet at the source and
- **Decapsulating** the payload from the network-layer packet at the destination.
- The source host receives the payload from an upper-layer protocol, adds a header that contains the source and destination address

- The destination host receives the network-layer packet from its data-link layer, **decapsulates** the packet, and delivers the payload to the corresponding upper-layer protocol.
- If the packet is fragmented at the source or at routers, along the path, the network layer is responsible for waiting until all fragments arrive, reassembling them, and delivering them to the upper-layer protocol.

# Routing and Forwarding

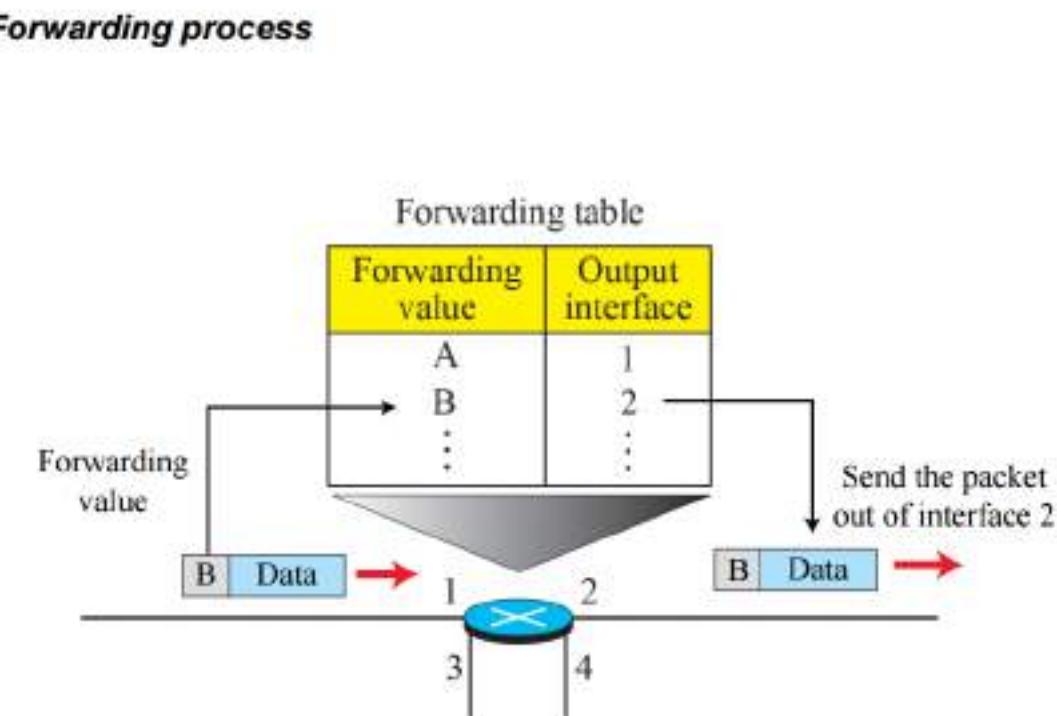
- Other duties of the network layer are

- Routing

- The network layer is responsible for finding the **best one** among these possible routes.
- The network layer needs to have some specific strategies for defining the best route.
- Routing protocol

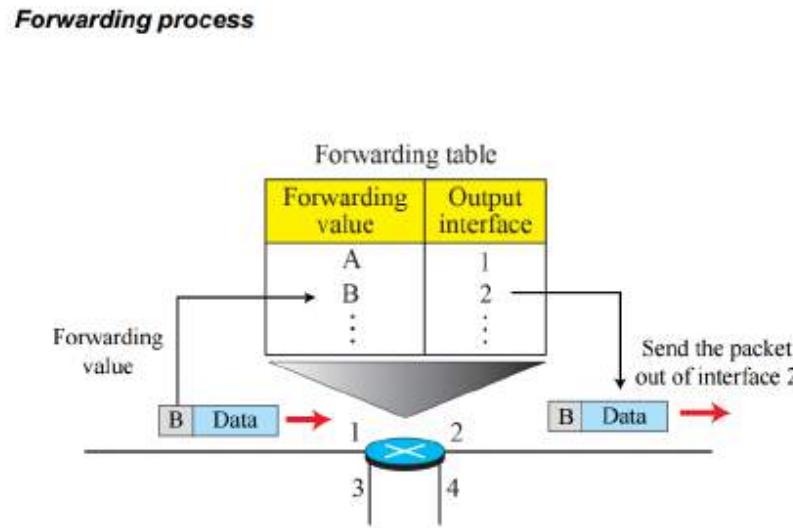
Network uses virtual circuits internally, routing decisions are made only when a new virtual circuit is being set up.

Therefore, data packets just follow the already established route. The latter case is sometimes called **session routing because route remains in force for an entire session.**



# Routing and Forwarding

- Other duties of the network layer are
  - Router has having two process inside it.
    - The network layer is responsible for routing the packet from its source to the destination.
    - Connected LAN and WANs and routers that connect them.
    - Network layer needs to have some specific strategies for defining the best route.
    - Routing Protocols
  - Forwarding
    - Forwarding is simply defined as the action applied by each router when a packet arrives at one of its interfaces.
    - When a router receives a packet from one of its attached networks, it needs to forward the packet to another attached network (in unicasting routing) or to some attached networks (in multicasting routing)
    - To make this decision, the router uses a piece of information in the packet header, which can be the destination address to find the corresponding output interface number in the forwarding table (routing table).



# Other Services

- Error Control
  - Can be implemented in the network layer but designers left this issue.
  - The main reason is packet in the network layer may be fragmented at each router, which makes the error checking at this layer inefficient.
- However, added a checksum field to the datagram to control any corruption in the header, but not in whole datagram.
- This checksum may prevent any changes or corruptions in the header of the datagram.

- Flow Control
  - The datagrams are sent by the sender when they are ready, without any attention to the readiness of the receiver.
  - Doesn't directly provide any flow control. There are three reasons
    - First, there is no error control in this layer, the job of the network layer at the receiver is so simple that it may rarely be overwhelmed.
    - Second, the upper layers that use the service of the network layer can implement buffers to receive data from the network layer as they are ready and do not have to consume the data as fast as it received.
    - Third, flow control is provided for most of the upper-layer protocols that the use of the service of the network layer, so another level of flow control makes the network layer more complicated and the whole system less efficient.

- Congestion Control
  - Is a situation in which too many datagrams are present in an area of the Internet.
  - Congestion may occur if the number of datagrams sent by source computers is beyond the capacity of the network or routers.
  - In this situation, some router may drop some datagrams.
  - in this situation more worst case as error control mechanism is in upper layers, the sender may send duplicates of the lost packets.
  - If the congestion continues , sometimes a situation may reach a point where the system collapse and no datagrams are delivered.
- Quality of Service
  - New applications such as multimedia communication arrives, QoS of the communication has become more and more important.
- Security
  - Initially security was not made concern as small number of users at universities for research activities;
  - Today is big issues

# PACKET SWITCHING

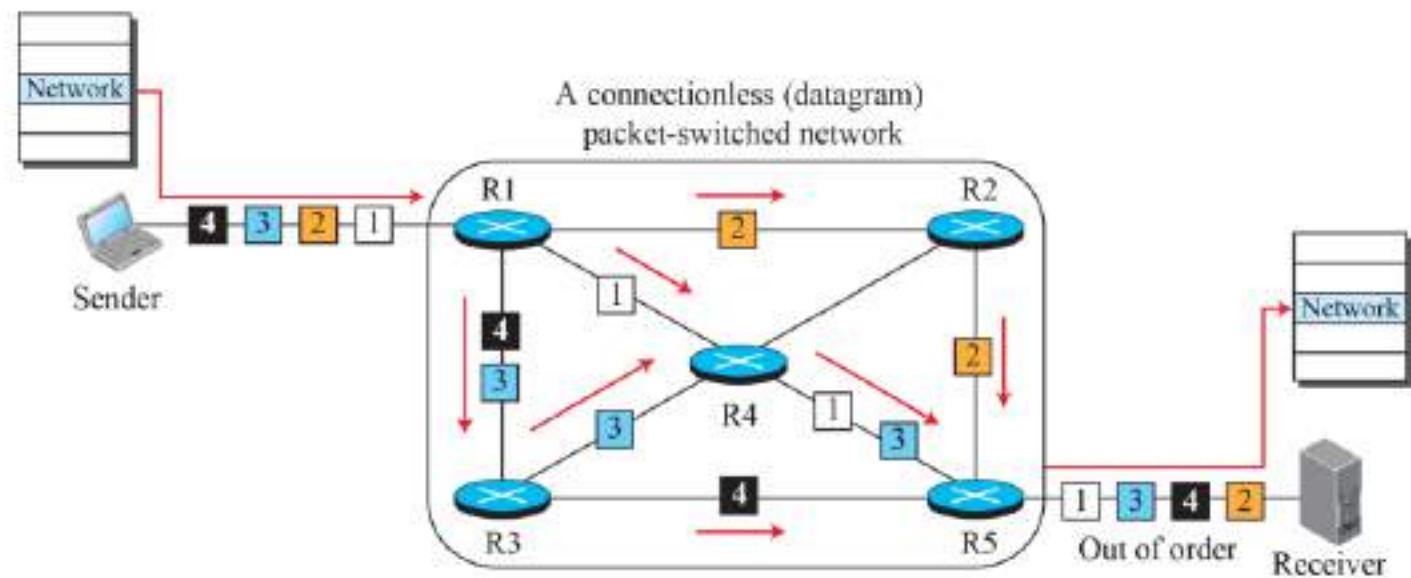
- A router is a switch that creates a connection between an input port and an output port (or a set of output ports).
- Two broad categories :
  - circuit switching and
  - packet switching
- A **message** from the upper layer is divided into manageable packets and each packet is sent through the network.
- A packet-switched network can use two different approaches to route the packets:
  - The datagram approach : Connectionless Service
  - Virtual circuit approach :Connection-Oriented Service

# Datagram Approach : Connectionless Service

- The network layer was designed to provide a connectionless service in which the network-layer protocol treats **each packet independently**, with each packet having no relationship to any other packet.
- The idea was that the network layer is only responsible for delivery of packets from the source to the destination.
- In this approach, the packets in a message may or may not travel the same path to their destination.

### *A connectionless packet-switched network*

- Each packet is routed based on the information contained in its header: source and destination addresses.



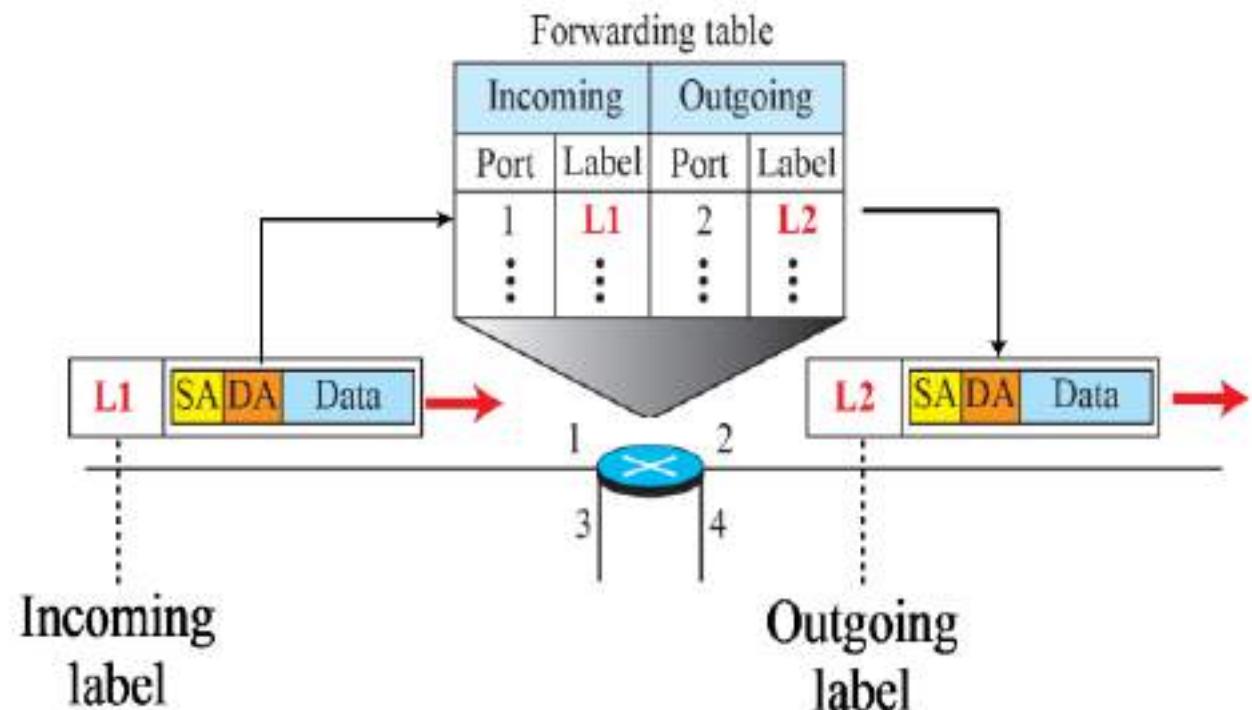
# Virtual-Circuit Approach : Connection-Oriented Service

- There is a relationship between all packets belonging to a message.
- Before all datagrams in a message can be sent, a virtual connection should be set up to define the path for the datagrams.
- After connection setup, the datagrams can all follow the same path.
- In this type of service, not only must the packet contain the **source and destination addresses**, it must also contain a **flow label**, a virtual circuit identifier that defines the virtual path the packet should follow.

---

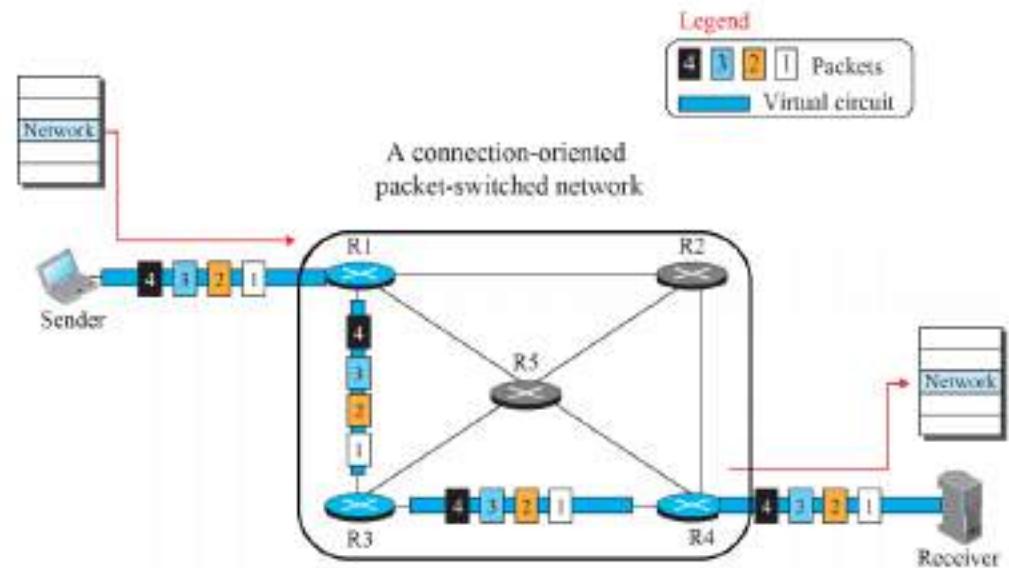
*Forwarding process in a router when used in a virtual circuit network*

- Each packet is forwarded based on the label in the packet.
- Forwarding decision is based on the value of the **label or virtual circuit identifier**.



- To create a connection-oriented service, a three phase process is used:
  - Setup
  - Data transfer
  - teardown

*A virtual-circuit packet-switched network*



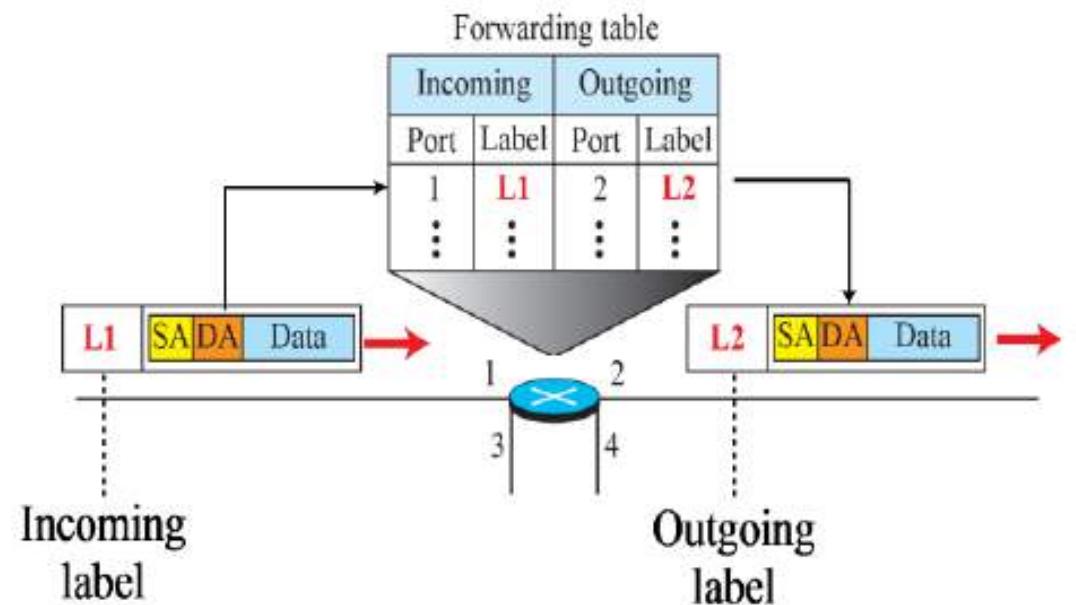
# Setup Phase

In the setup phase, a router creates an entry for a virtual circuit.

- Source A needs to create a virtual circuit to destination B.
- **Two auxiliary packets** need to be exchanged between the sender and the receiver: the **request packet** and the **acknowledgement packet**.

---

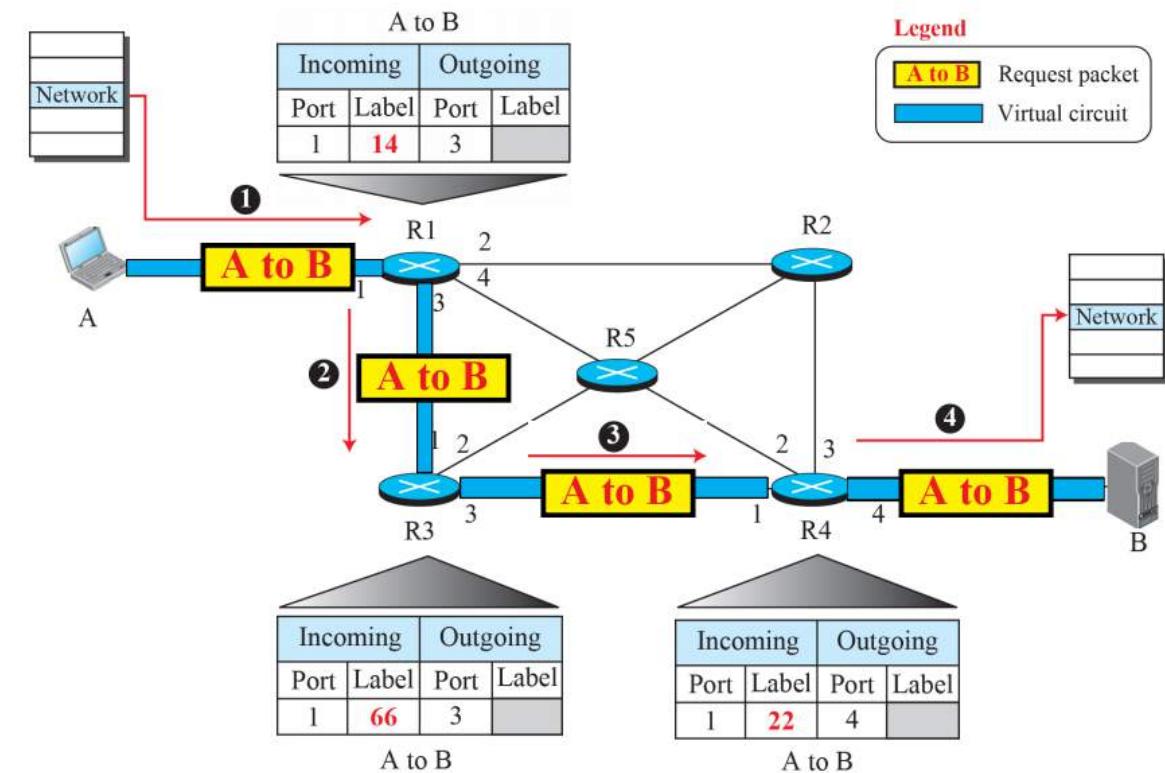
*Forwarding process in a router when used in a virtual circuit network*



# Request Packet

- A request packet is sent from the source to the destination.
- This auxiliary packet carries the source and destination address.

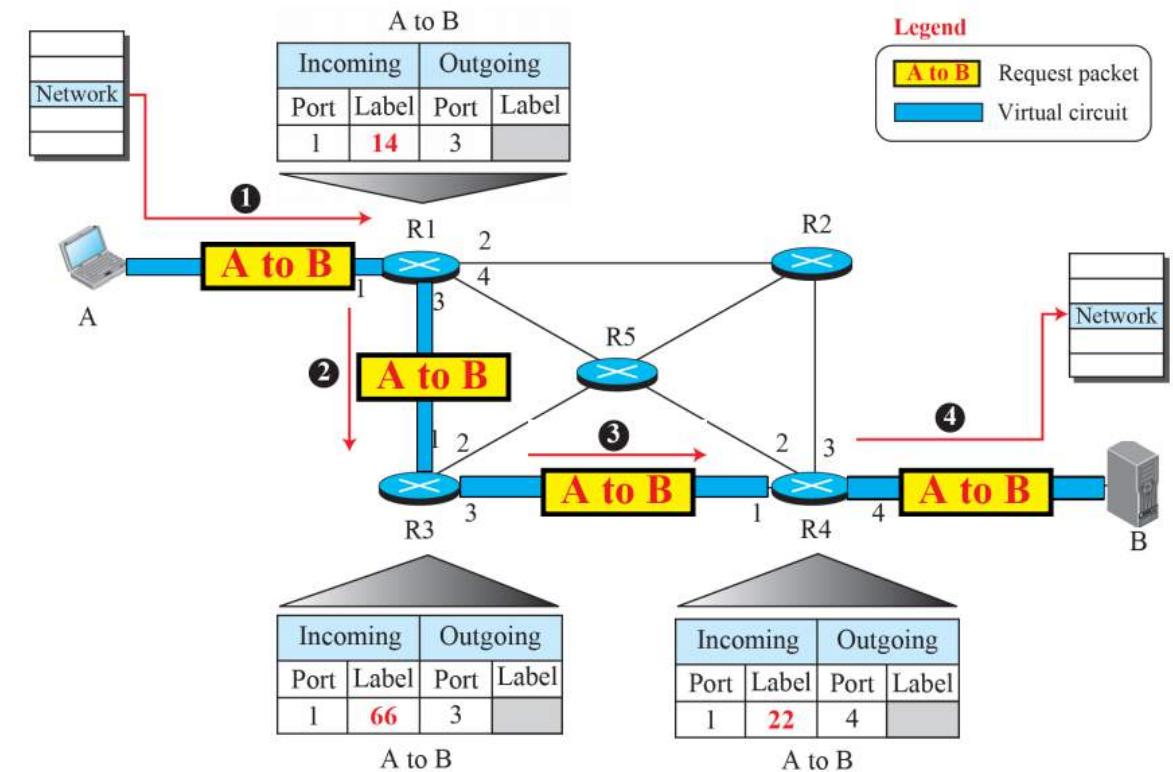
*Sending request packet in a virtual-circuit network*



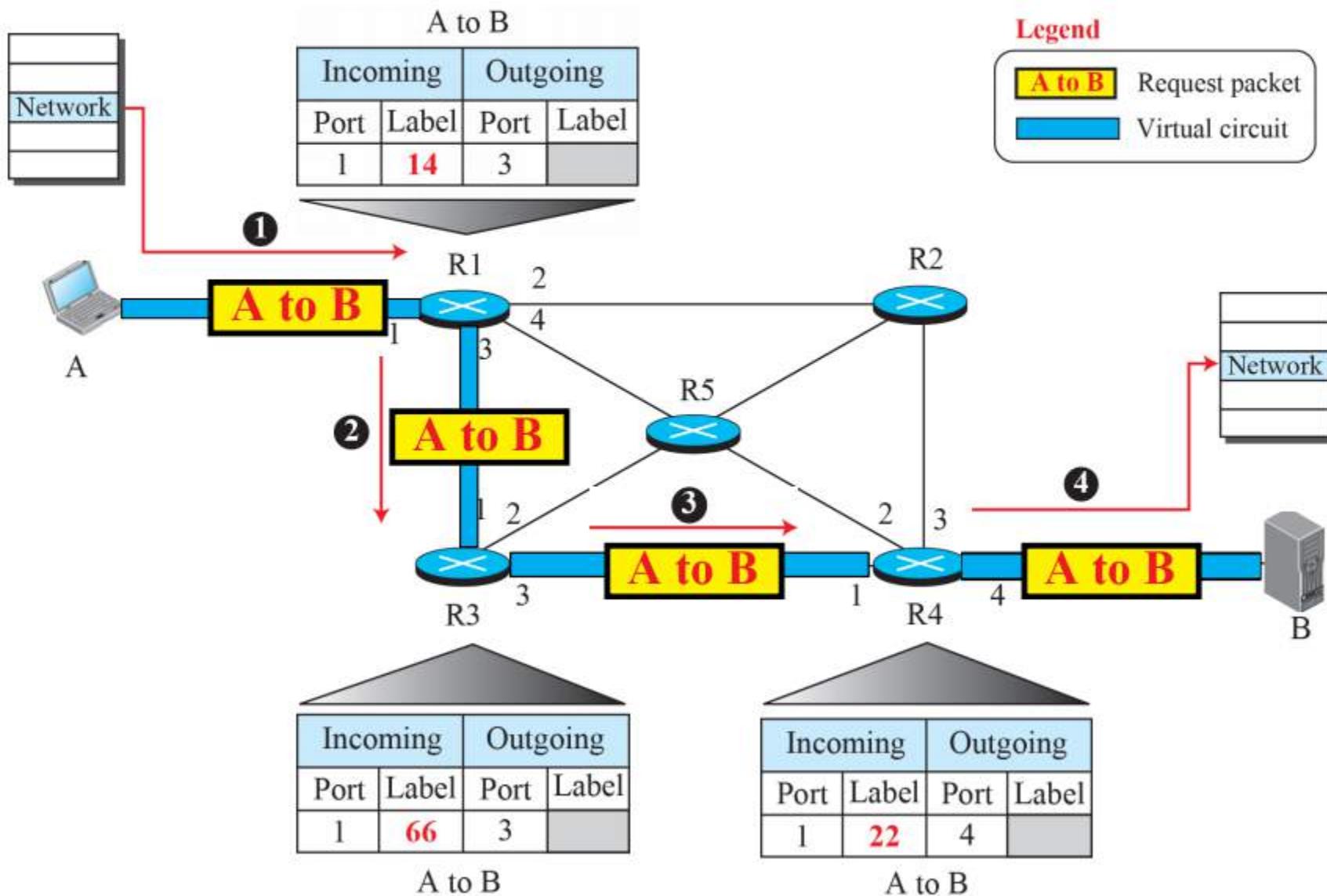
# Request Packet

- Source A sends a request packet to router R1
- Router R1 receives the request packet.

*Sending request packet in a virtual-circuit network*



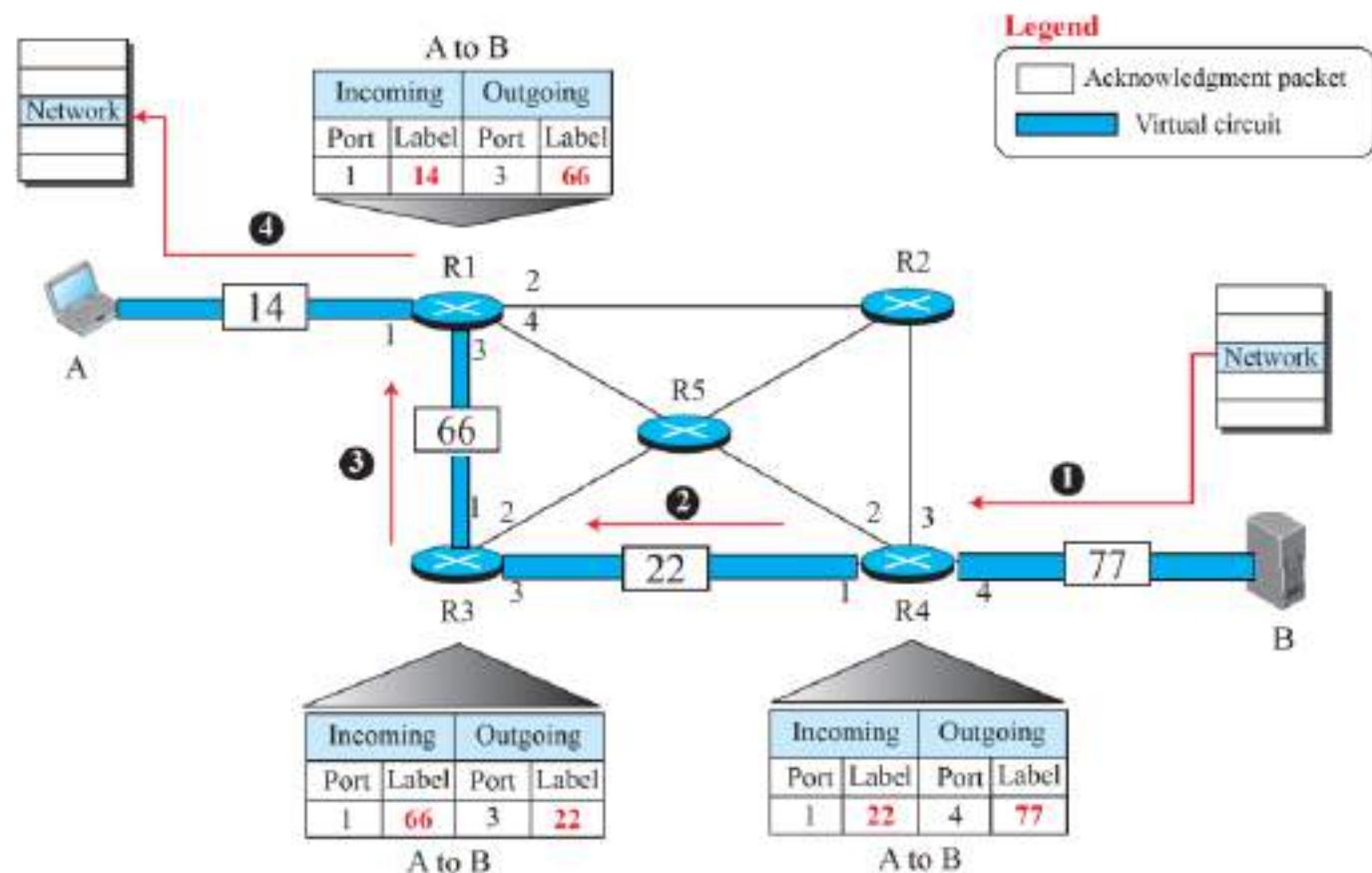
## Sending request packet in a virtual-circuit network



# Acknowledgment Packet

- The destination sends an acknowledgment to router R4.
- The packet also carries label 77, chosen by the destination as the incoming label for packets from A.

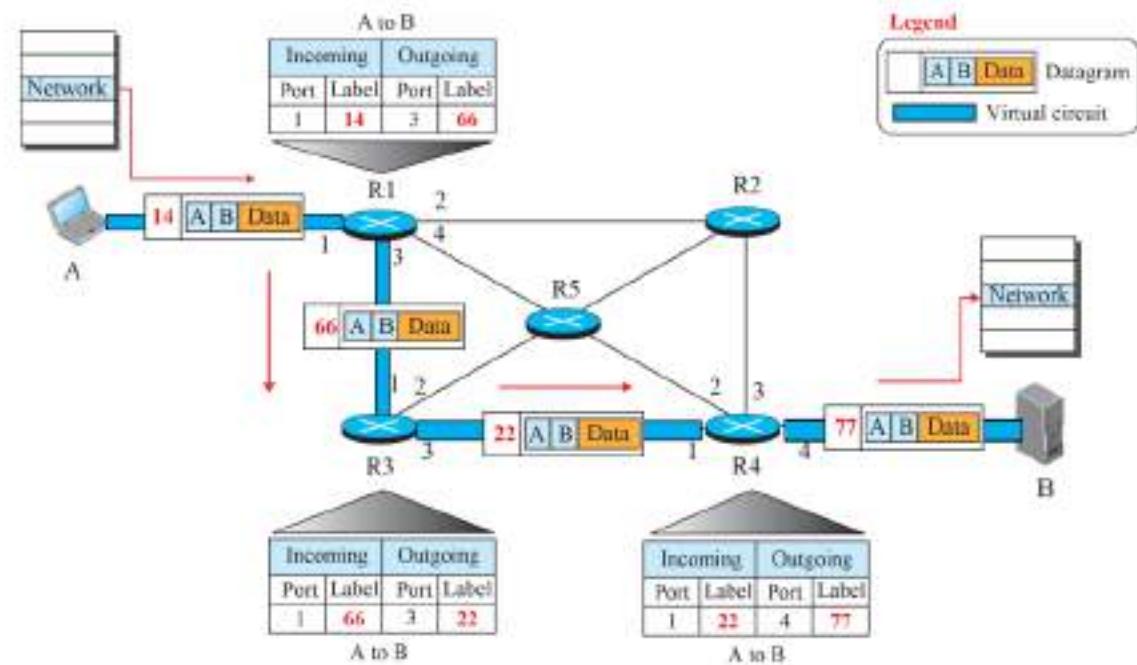
*Sending acknowledgments in a virtual-circuit network*



# Data-Transfer Phase

- After all routers have created their forwarding table for a specific virtual circuit, then the network-layer packets belonging to one message can be sent one after another.
- The source computer uses the label 14, which it has received from router R1 in the setup phase.
- All the packets in the message follow the same sequences of labels, and the packets arrive in order at the destination.

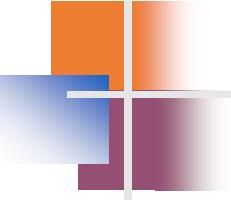
*Flow of one packet in an established virtual circuit*



# Teardown Phase

- In the teardown phase, source A, after sending all packets to B, sends a special packet called a teardown packet.
- Destination B responds with a confirmation packet. All routers delete the corresponding entries from their tables.

# **Network Layer: Logical Addressing**

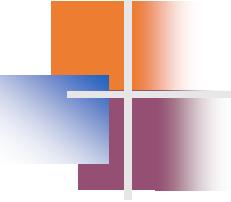


### *Note*

---

**An IPv4 address is 32 bits long.**

---



## *Note*

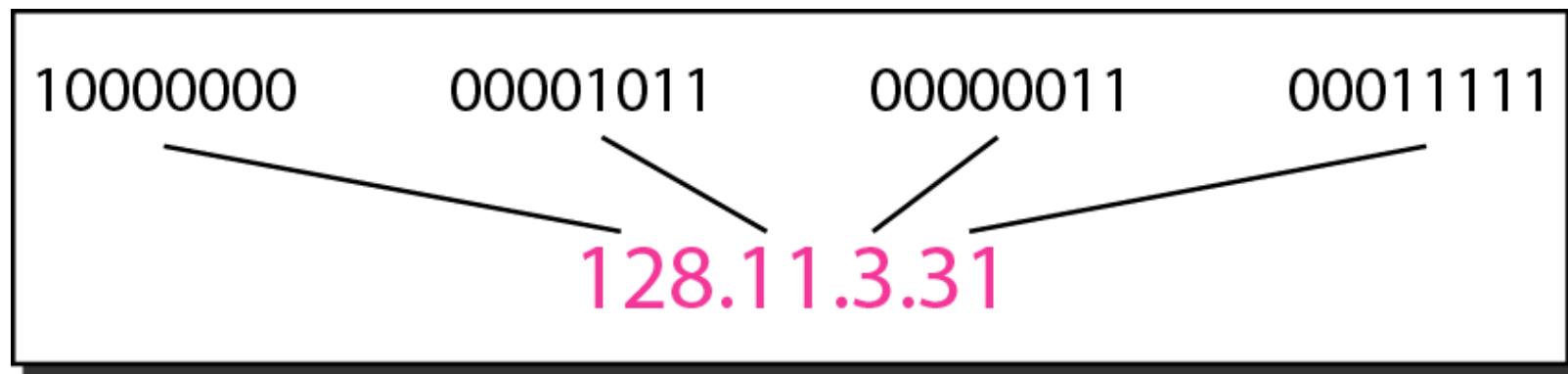
---

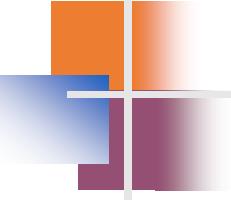
**The address space of IPv4 is  
 $2^{32}$  or 4,294,967,296.**

---

**Figure 19.1 Dotted-decimal notation and binary notation for an IPv4 address**

---





## Example 19.1

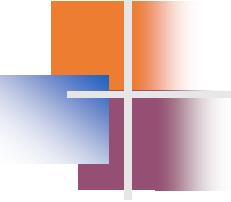
*Change the following IPv4 addresses from binary notation to dotted-decimal notation.*

- a. 10000001 00001011 00001011 11101111
- b. 11000001 10000011 00011011 11111111

### Solution

*each group of 8 bits with its equivalent decimal number and add dots for separation.*

- a. 129.11.11.239
- b. 193.131.27.255



## **Note**

---

**In classful addressing, the address space is divided into five classes:  
A, B, C, D, and E.**

---

**Figure 19.2** Finding the classes in binary and dotted-decimal notation

	First byte	Second byte	Third byte	Fourth byte
Class A	0			
Class B	10			
Class C	110			
Class D	1110			
Class E	1111			

a. Binary notation

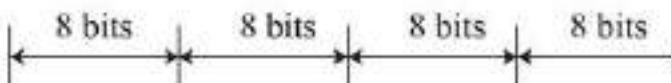
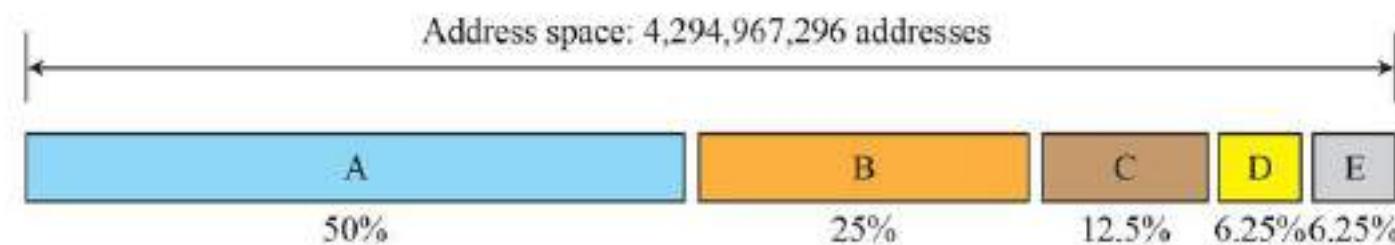
	First byte	Second byte	Third byte	Fourth byte
Class A	0–127			
Class B	128–191			
Class C	192–223			
Class D	224–239			
Class E	240–255			

b. Dotted-decimal notation

Class	Number of Blocks	Block Size	Application
A	128	16,777,216	Unicast
B	16,384	65,536	Unicast
C	2,097,152	256	Unicast
D	1	268,435,456	Multicast
E	1	268,435,456	Reserved

<i>Class</i>	<i>Number of Blocks</i>	<i>Block Size</i>	<i>Application</i>
A	128	16,777,216	Unicast
B	16,384	65,536	Unicast
C	2,097,152	256	Unicast
D	1	268,435,456	Multicast
E	1	268,435,456	Reserved

**Figure 18.18:** Occupation of the address space in classful addressing



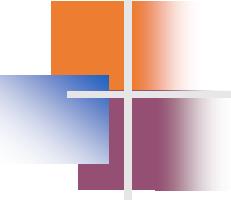
Class A	0 Prefix	Suffix
Class B	10 Prefix	Suffix
Class C	110 Prefix	Suffix
Class D	1110 Multicast addresses	
Class E	1111 Reserved for future use	

<b>Class</b>	<b>Prefixes</b>	<b>First byte</b>
A	$n = 8$ bits	0 to 127
B	$n = 16$ bits	128 to 191
C	$n = 24$ bits	192 to 223
D	Not applicable	224 to 239
E	Not applicable	240 to 255

## Classless interdomain Routing

**Table 19.2** *Default masks for classful addressing*

Class	Binary	Dotted-Decimal	CIDR
A	11111111 00000000 00000000 00000000	255.0.0.0	/8
B	11111111 11111111 00000000 00000000	255.255.0.0	/16
C	11111111 11111111 11111111 00000000	255.255.255.0	/24



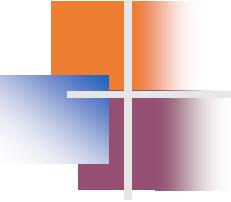
## *Example 19.4*

*Find the class of each address.*

- a. 00000001 00001011 00001011 11101111
- b. 11000001 10000011 00011011 11111111
- c. 14.23.120.8
- d. 252.5.15.111

*Solution*

- a. *The first bit is 0. This is a class A address.*
- b. *The first 2 bits are 1; the third bit is 0. This is a class C address.*
- c. *The first byte is 14; the class is A.*
- d. *The first byte is 252; the class is E.*

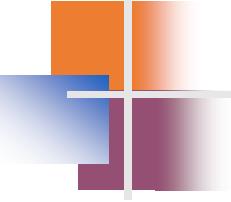


## *Note*

---

**Classful addressing, which is almost obsolete, is replaced with classless addressing.**

---



## **Note**

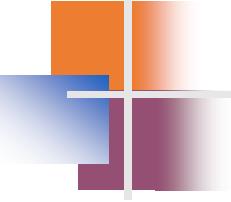
---

**In IPv4 addressing, a block of addresses can be defined as**

**x.y.z.t /n**

**in which x.y.z.t defines one of the addresses and the /n defines the mask.**

---

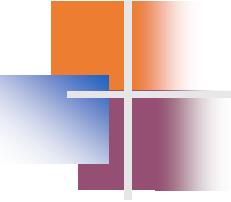


## *Note*

---

**The first address in the block can be found by setting the rightmost  $32 - n$  bits to 0s.**

---



## Example 19.6

*A block of addresses is granted to a small organization. We know that one of the addresses is 205.16.37.39/28. What is the first address in the block?*

### *Solution*

*The binary representation of the given address is*

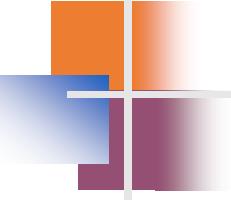
*11001101 00010000 00100101 00100111*

*If we set 32–28 rightmost bits to 0, we get*

*11001101 00010000 00100101 00100000*

*or*

*205.16.37.32.*

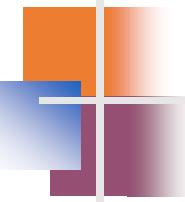


## **Note**

---

**The last address in the block can be found by setting the rightmost  $32 - n$  bits to 1s.**

---



## Example 19.7

*Find the last address for the block in Example 19.6.*

### Solution

*The binary representation of the given address is*

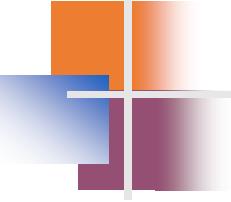
*11001101 00010000 00100101 00100111*

*If we set 32 – 28 rightmost bits to 1, we get*

*11001101 00010000 00100101 00101111*

*or*

*205.16.37.47*

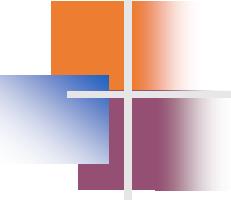


## **Note**

---

**The number of addresses in the block  
can be found by using the formula  
 $2^{32-n}$ .**

---

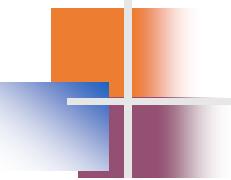


## Example 19.8

*Find the number of addresses in 205.16.37.39/28*

### Solution

*The value of n is 28, which means that number of addresses is  $2^{32-28}$  or 16.*



## Example 19.9

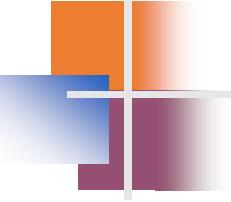
*Another way to find the first address, the last address, and the number of addresses is to represent the mask as a 32-bit binary (or 8-digit hexadecimal) number. This is particularly useful when we are writing a program to find these pieces of information. In Example 19.5 the /28 can be represented as*

**11111111 11111111 11111111 11110000**

*(twenty-eight 1s and four 0s).*

*Find*

- a. *The first address*
- b. *The last address*
- c. *The number of addresses.*

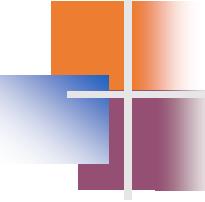


## Example 19.9 (continued)

### Solution

- a. The first address can be found by ANDing the given addresses with the mask. ANDing here is done bit by bit. The result of ANDing 2 bits is 1 if both bits are 1s; the result is 0 otherwise.

Address:	11001101 00010000 00100101 00100111
Mask:	<b>11111111 11111111 11111111 11110000</b>
First address:	11001101 00010000 00100101 00100000

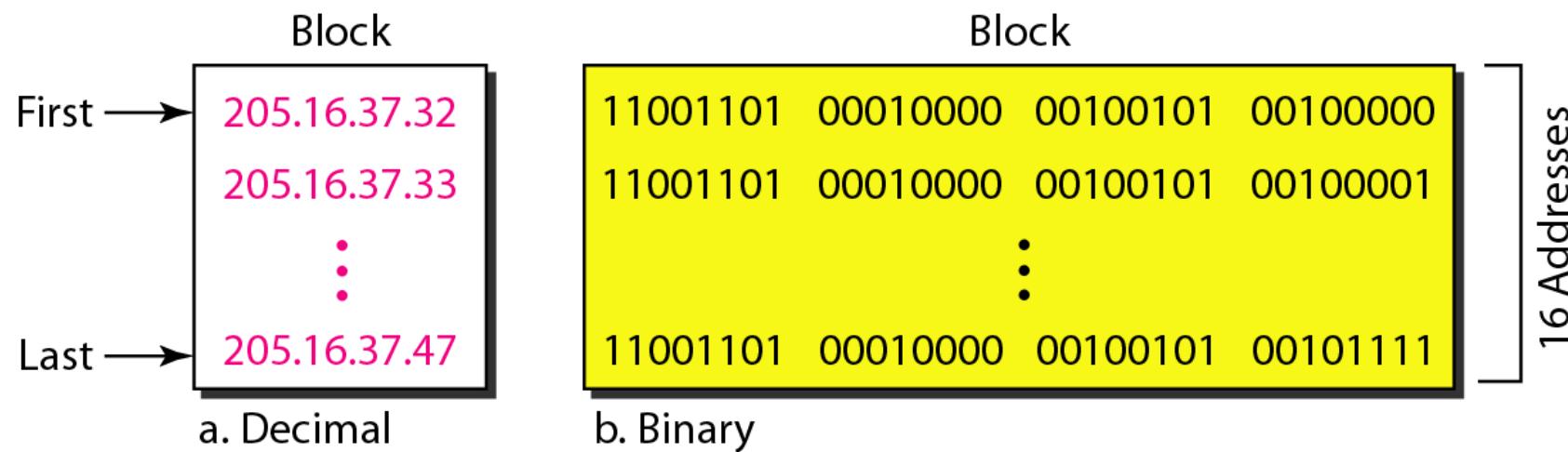


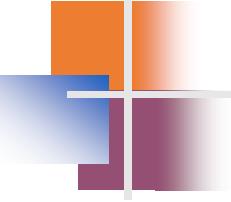
## *Example 19.9 (continued)*

*b. The last address can be found by ORing the given addresses with the complement of the mask. ORing here is done bit by bit. The result of ORing 2 bits is 0 if both bits are 0s; the result is 1 otherwise. The complement of a number is found by changing each 1 to 0 and each 0 to 1.*

Address:	11001101	00010000	00100101	00100111
Mask complement:	00000000	00000000	00000000	00001111
Last address:	11001101	00010000	00100101	00101111

**Figure 19.4** A network configuration for the block 205.16.37.32/28





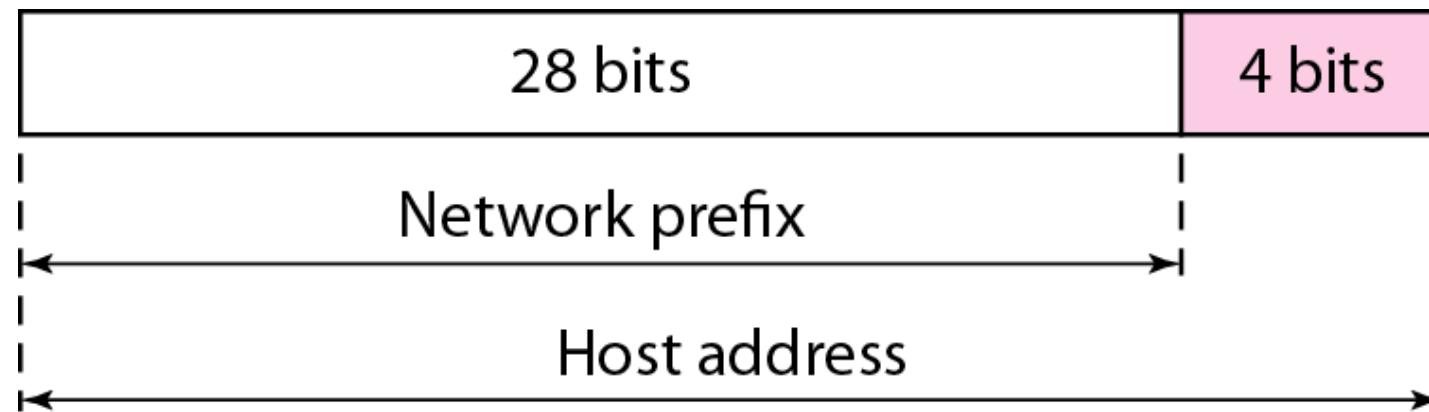
## **Note**

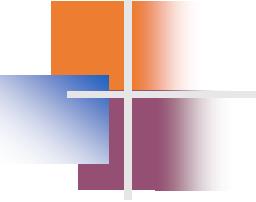
---

**The first address in a block is normally not assigned to any device; it is used as the network address that represents the organization to the rest of the world.**

---

**Figure 19.6** A frame in a character-oriented protocol

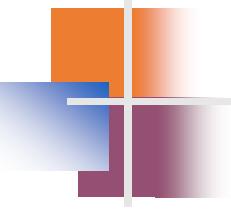




## **Note**

---

**Each address in the block can be considered as a two-level hierarchical structure:**  
**the leftmost  $n$  bits (prefix) define the network;**  
**the rightmost  $32 - n$  bits define the host.**



## *Block Allocation*

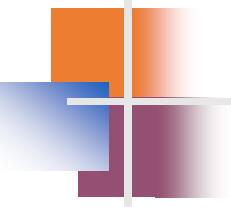
The next issue is classless addressing is block allocation.  
How are the blocks allocated ?

The ultimate responsibility of block allocation is given to a global authority called the Internet Corporation for Assigned Names and Numbers (ICANN)

The number of requested address, N, needs to be of 2. The reason is that

$$N=2^{32-n} \quad \text{or} \quad n=32-\log_2 N.$$

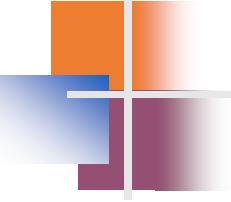
If N is not a power of 2, we cannot have an integer value of n.



## *Example 18.5*

- An organization is granted a block of addresses with the beginning address 14.24.74.0/24.
- The organization needs to have 3 subblocks of addresses
  - one subblocks of 10 addresses,
  - one subblock of 60 addresses, and
  - one subblock of 120 addresses.

Design the subblocks.



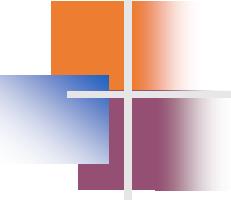
## *Example 18.5*

There are  $2^{32-24} = 256$  addresses in this block.

- The first address is 14.24.74.0/24.
- the last address is 14.24.74.255/24.

To satisfy the third requirement,  
we assign addresses to subblocks,

starting with the largest and ending with the smallest one.



## *Example 18.5*

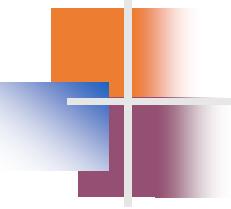
- a. *The number of addresses in the largest subblock, which requires 120 addresses, is not a power of 2.*

*We allocate 128 addresses.*

*The subnet mask for this subnet can be found as*

$$n_1 = 32 - \log_2 128 = 25.$$

*The first address in this block is 14.24.74.0/25;  
the last address is 14.24.74.127/25*



## *Example 18.5*

b. The number of addresses

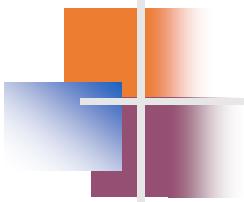
in the second largest subblock,  
which requires 60 addresses, is not a power of 2.

We allocate 64 addresses.

The subnet mask for this subnet can be found as

$$n_2 = 32 - \log_2 64 = 26.$$

The first address in this block is 14.24.74.128/26;  
the last address is 14.24.74.191/26



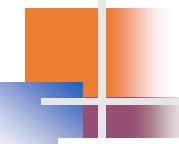
## Example 18.5

c. The number of addresses in the smallest subblock, which requires 10 addresses, is not a power of 2.

We allocate 16 addresses. The subnet mask for this subnet can be found as

$$n_3 = 32 - \log_2 16 = 28.$$

The first address in this block is 14.24.74.192/28; the last address is 14.24.74.207/28



# *Special Address*

## **This-host Address**

- 0.0.0.0/32
  - It is used whenever a host needs to send an IP datagram

## **Limited-broadcast Address**

- 255.255.255.255/32
  - Whenever a router or a host needs to send a datagram to all devices in a network.

## **Loopback Address**

- 127.0.0.0/8

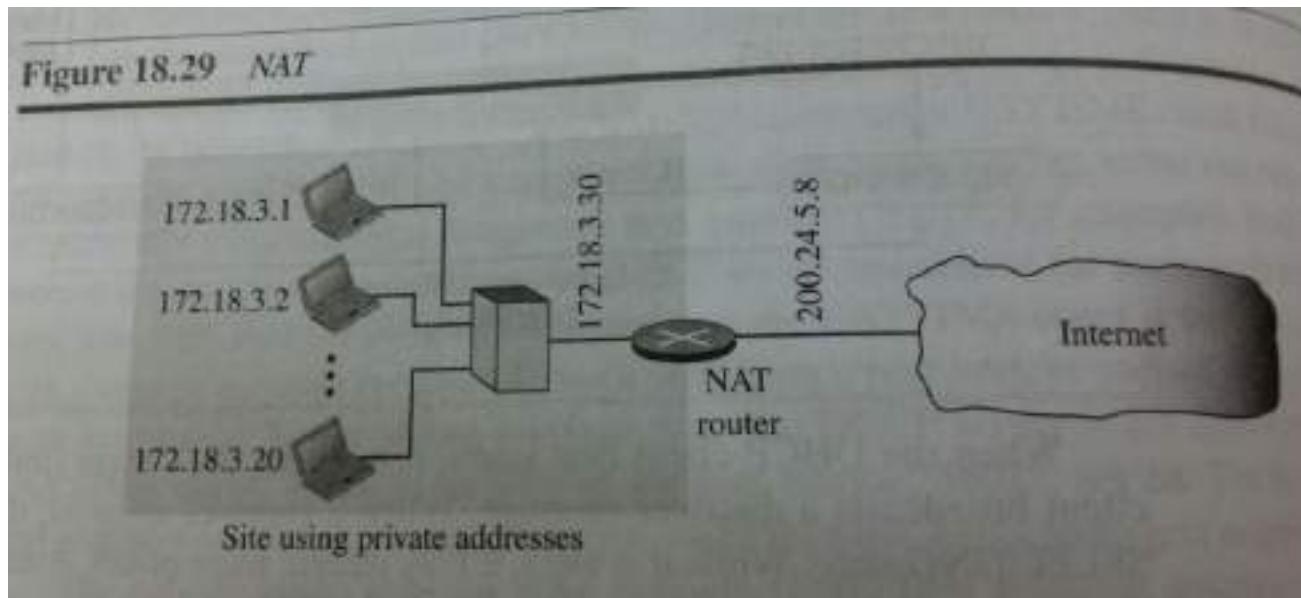
## **Private Address**

- 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 and 169.254.0.0/16

## **Multicast Addresses**

224.0.0.0/4

## *Network Address Translation (NAT)*



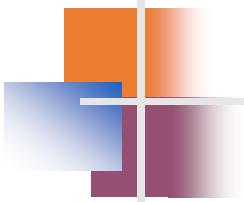
*The technology allows a site to use a set of private addresses for internal communication and a set of global Internet addresses (at least one) for communication with the rest of the world.*

## 19-2 IPv6 ADDRESSES

*Despite all short-term solutions, address depletion is still a long-term problem for the Internet. This and other problems in the IP protocol itself have been the motivation for IPv6.*

[\*Topics discussed in this section:\*](#)

**Structure**



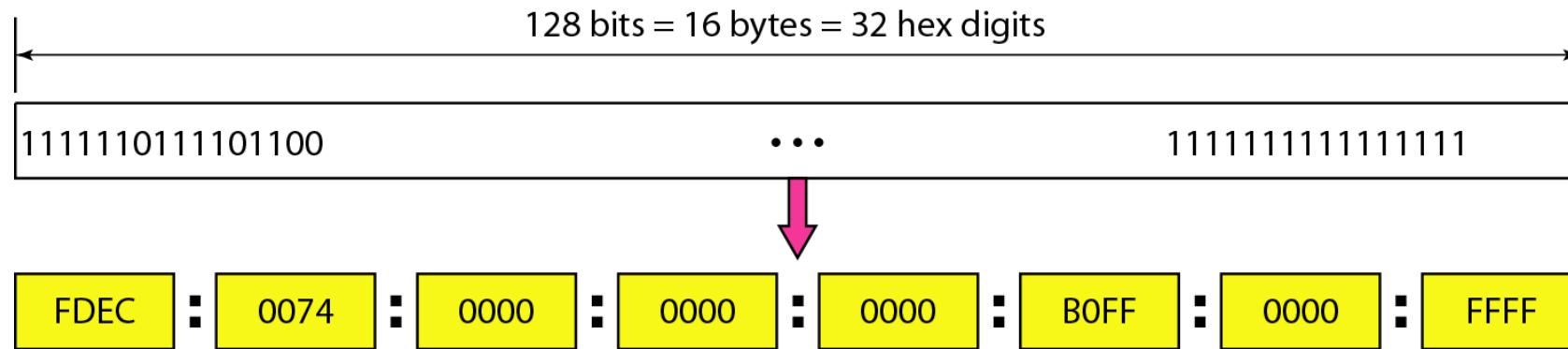
*Note*

---

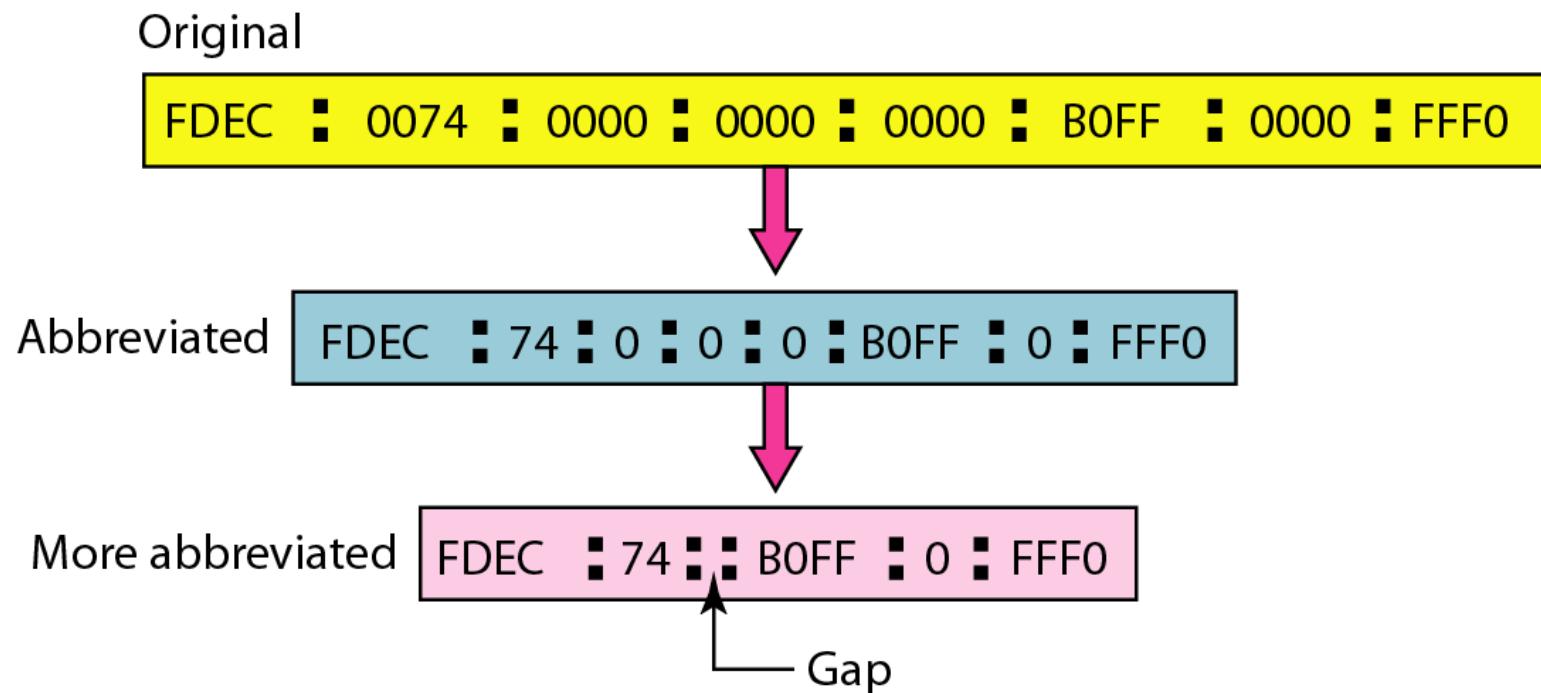
**An IPv6 address is 128 bits long.**

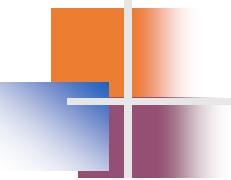
---

**Figure 19.14** IPv6 address in binary and hexadecimal colon notation



**Figure 19.15 Abbreviated IPv6 addresses**





## Example 19.11

*Expand the address 0:15::1:12:1213 to its original.*

### **Solution**

*We first need to align the left side of the double colon to the left of the original pattern and the right side of the double colon to the right of the original pattern to find how many 0s we need to replace the double colon.*

XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX
0: 15: : 1: 12:1213

*This means that the original address is.*

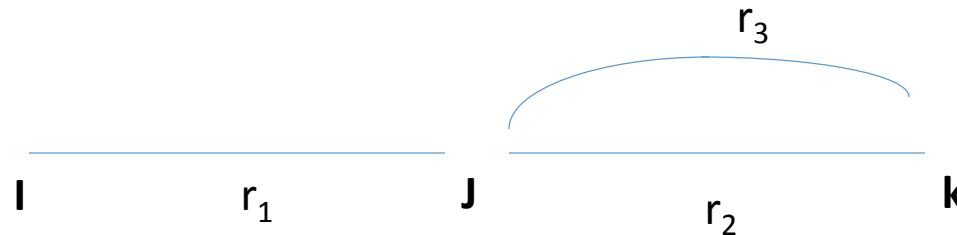
0000:0015:0000:0000:0000:0001:0012:1213
---

# Routing Algorithms

- Routing algorithms are the part of network layer software that are responsible for deciding which output line should an incoming packets be transmitted on.
- Certain properties are desirable in a routing algorithm : correctness, simplicity, robustness, stability, fairness and efficiency.
  - **Non-Adaptive**
    - Predefined do not consider network traffic.
    - This procedure is sometimes called static routing.
    - static routing is mostly useful for situations in which the routing choice is clear
      - *Give example of Video conference of KU*
  - **Adaptive →**
    - Real time optimal, changes regularly dynamic

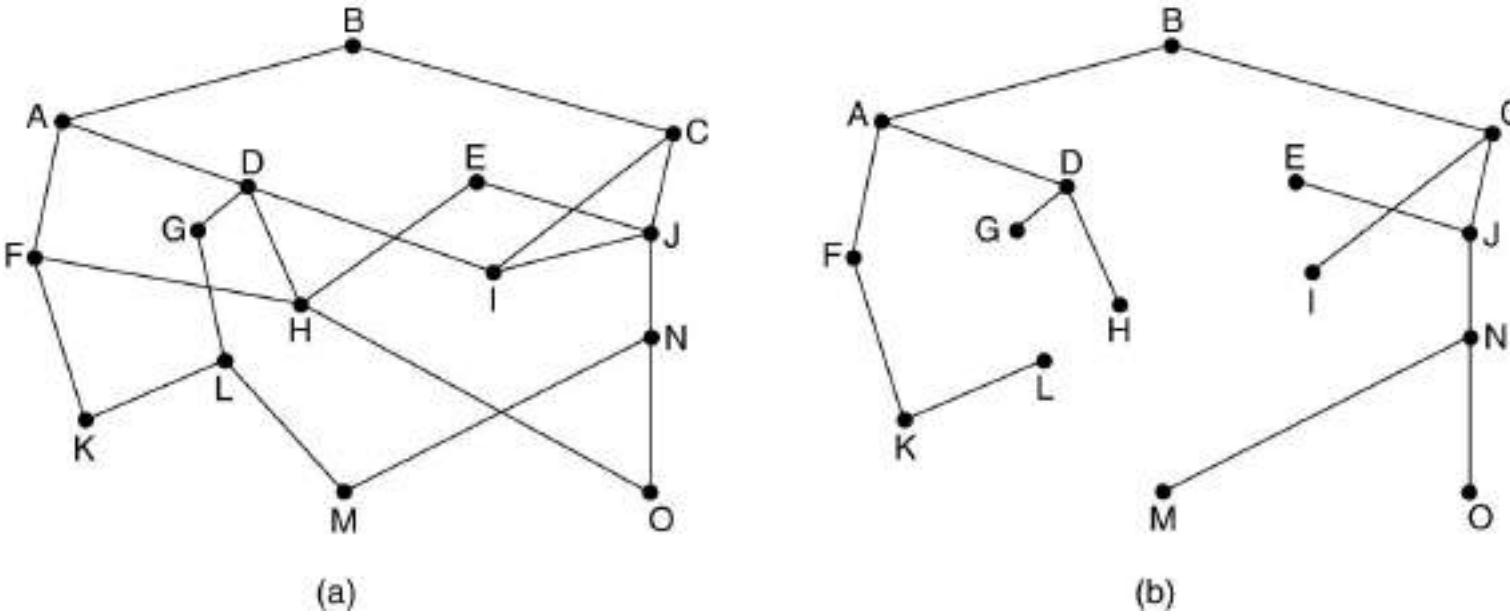
# Optimality Principle

- It states that “*if a router J is on the optimal path from router I to router K, then the optimal path from J to K also falls on the same route*”



- Let  $I$  to  $J$  be  $r_1$  and rest be  $r_2$
- It states that if router  $J$  is on the optimal path from router  $I$  to router  $K$  then the optimal path from  $J$  to  $K$  also falls along the same route.
- To see this, call the part of the route from  $I$  to  $J$   $r_1$  and the rest of the route  $r_2$ .
- If a route better than  $r_2$  (say  $r_3$ ) existed from  $J$  to  $K$ , it could be concatenated with  $r_1$  to improve the route from  $I$  to  $K$ , contradicting our statement that  $r_1r_2$  is optimal.

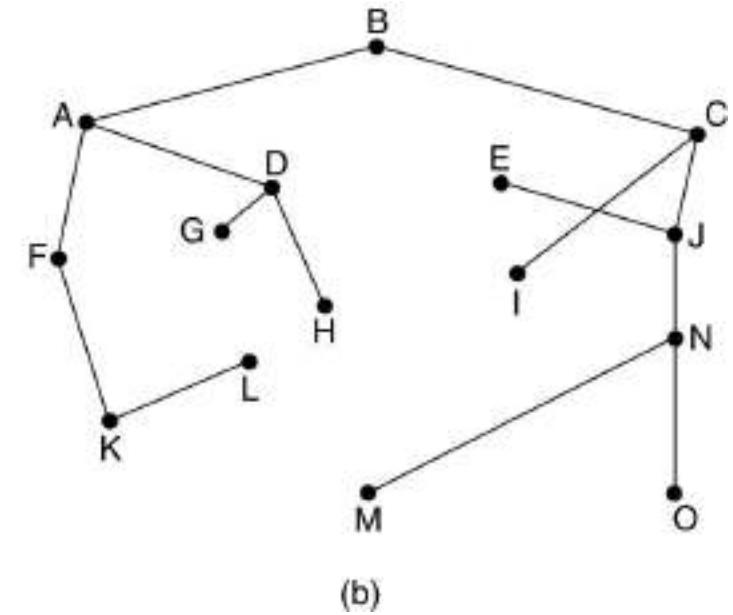
# The Optimality Principle



As a direct consequence of the optimality principle, we can see that the set of **optimal routes** from all sources to a given destination form a tree rooted at the destination. Such a tree is called **a sink tree**.

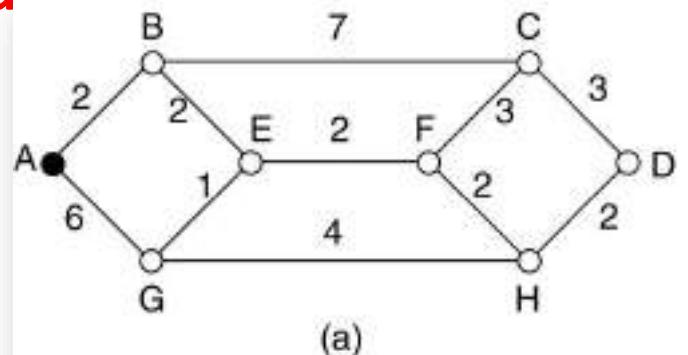
**(a)** A subnet. **(b)** A sink tree for router B.

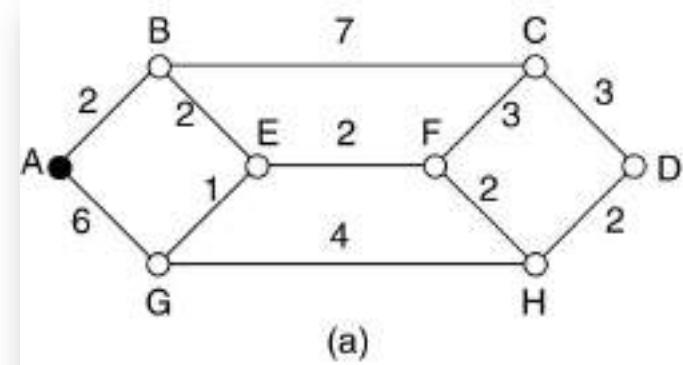
- a sink tree is not necessarily unique; other trees with the same path lengths may exist.
- If we allow all of the possible paths to be chosen, the tree becomes a more general structure called a DAG (Directed Acyclic Graph). DAGs have no loops.
- The goal of all routing algorithm is to discover and use the sink trees for all routers.

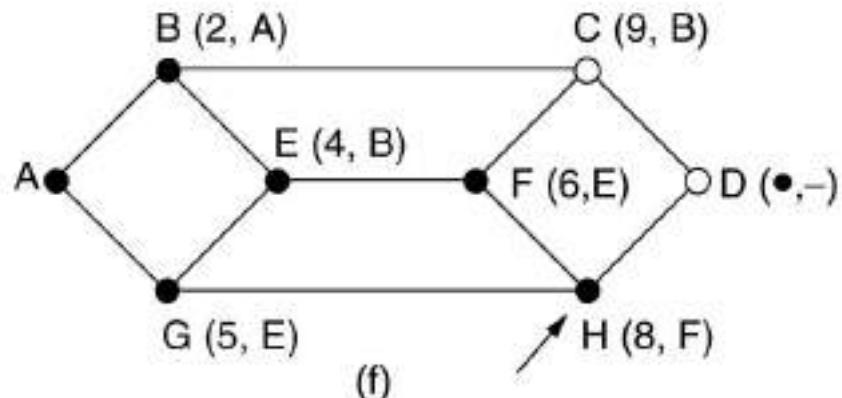
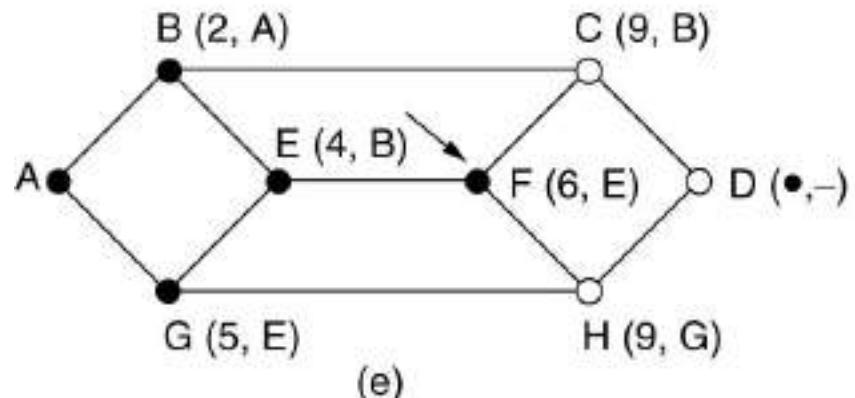
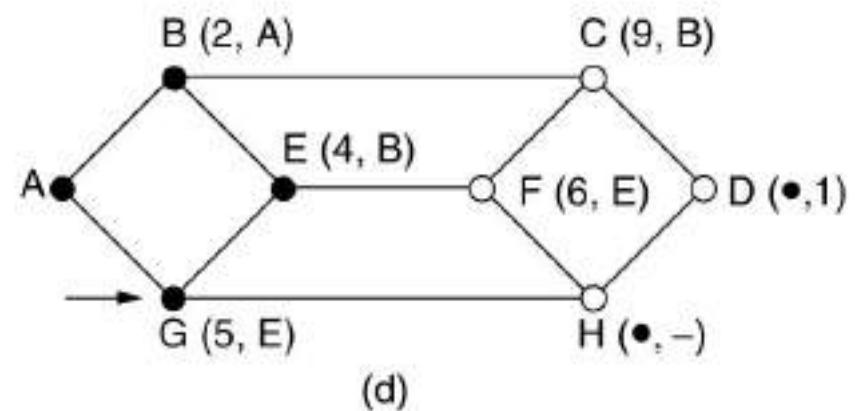
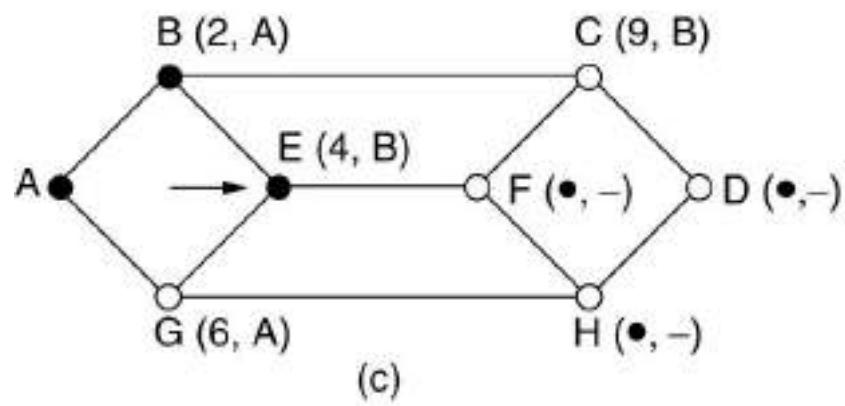
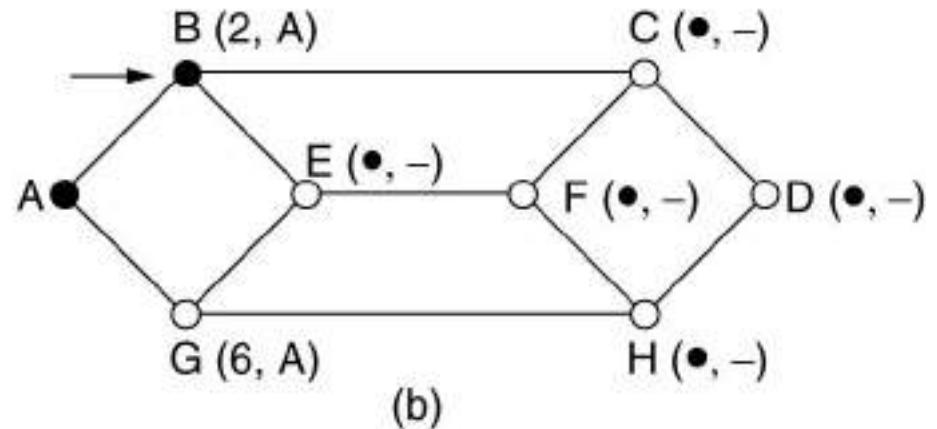
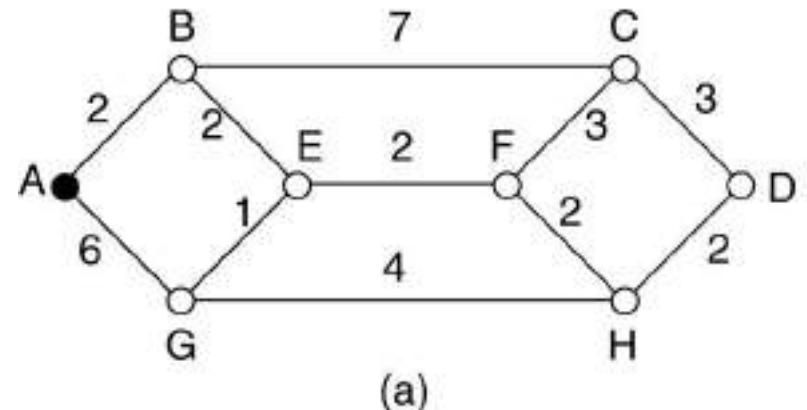


# Shortest Path Routing

- Every path has its measurement.
- For every node(router) there is a path connecting other nodes.
- There might be multiple paths from a node to the other node.
- While traversing, calculate the shortest distance and proceed.
- We use **Dijkstra's algorithm** to find the shortest path.



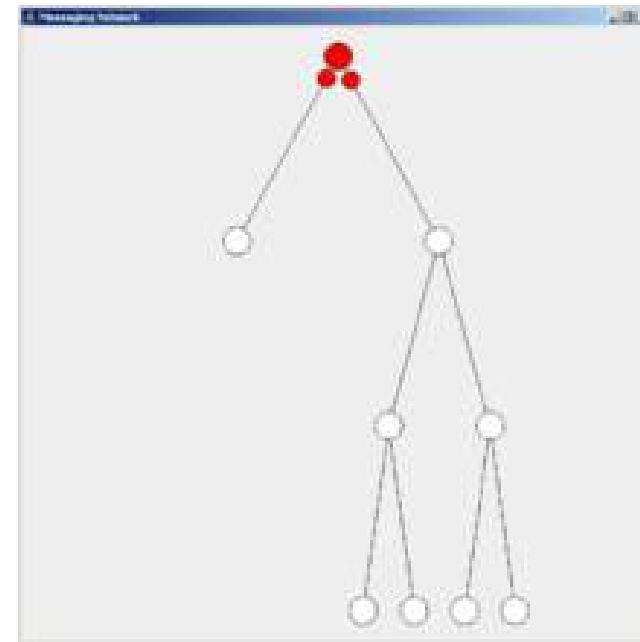




# Flooding

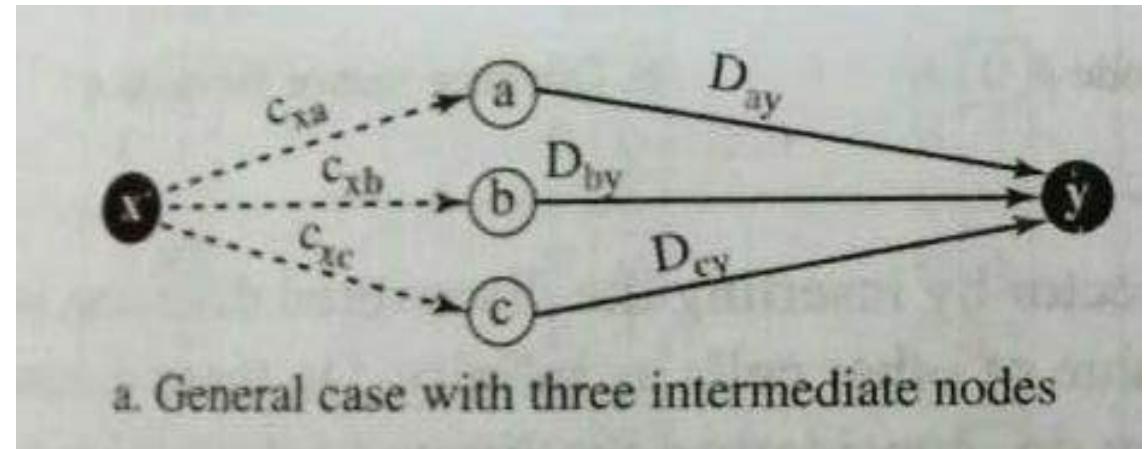
- When a routing algorithm is implemented, each router must make decisions, based on local knowledge, not the complete picture of the network.
  - i.e Each router must have the detail of adjacent router.
- A simple local technique is flooding, in which incoming packet is sent out on every outgoing line except the one it arrived on.
- Many duplicate packets will be received.
- Its solution is the implementation of hop counter.
- Hop counter is contained in the header of the packet which is decremented at each hop and when it reaches zero it is discarded.

- If the sender does not know how long the path is, it can initialize the counter to the worst case, namely, the full diameter of the network.



# Distance Vector Routing

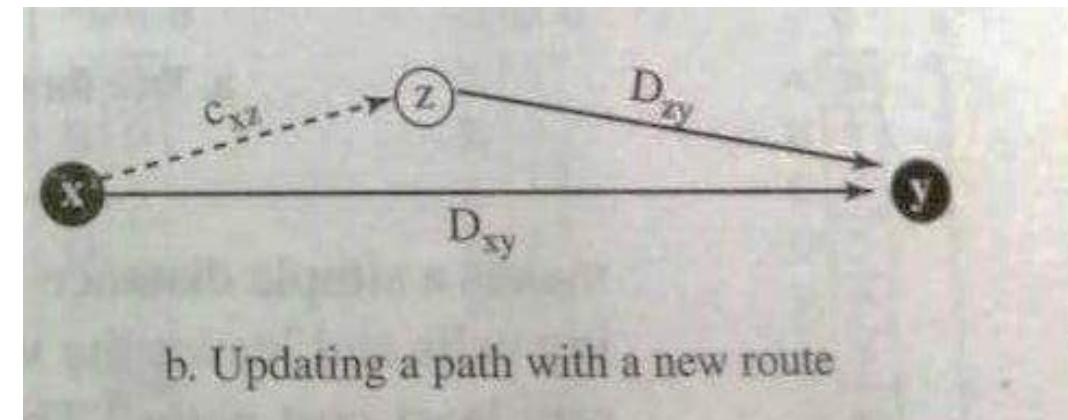
- It is sometime called distributed **Bellman-Ford routing algorithm**.
- Equation is used to find the least cost (shortest distance) between a source node, x and a destination node y, through some intermediary nodes (a,b,c)



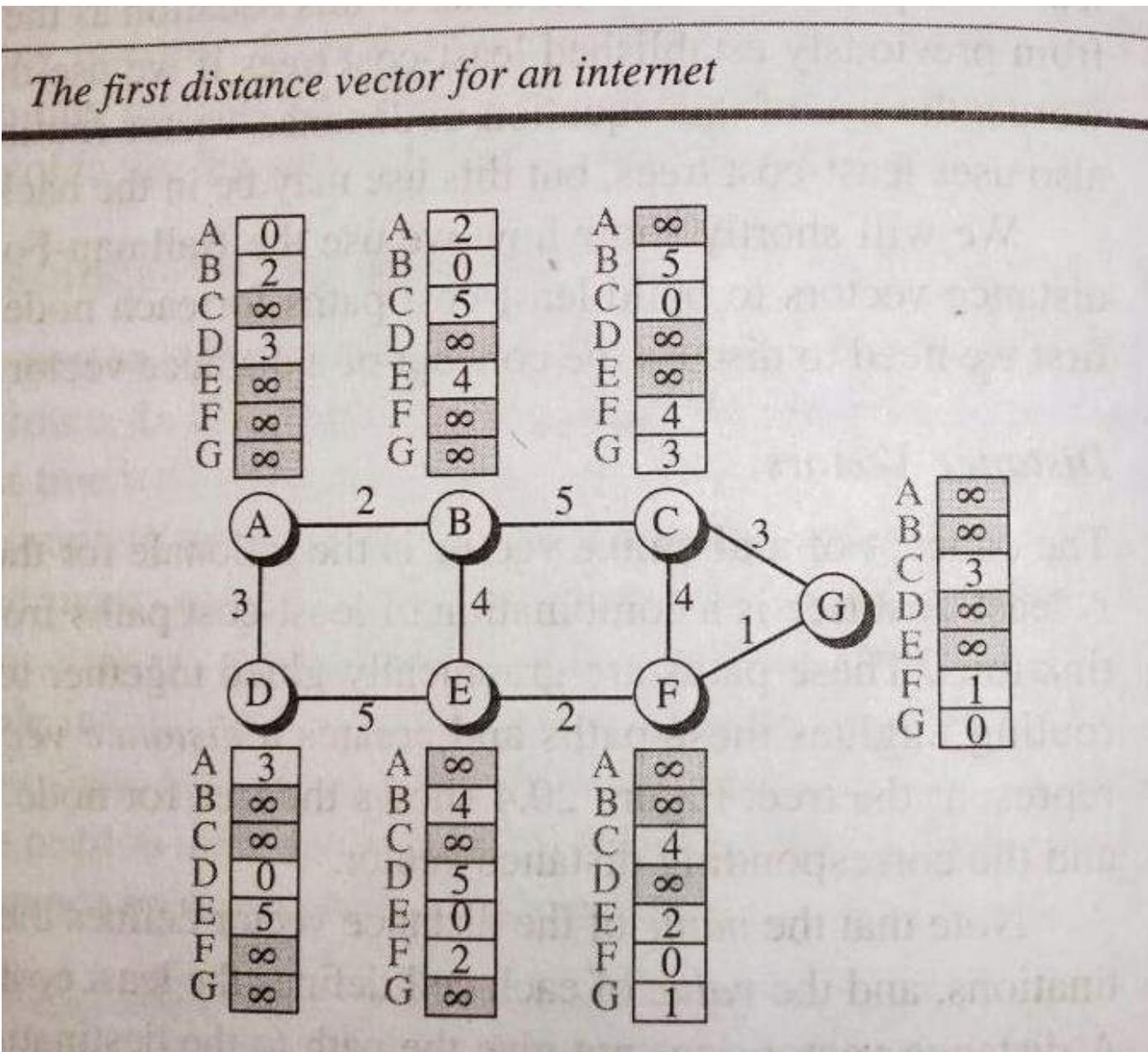
$$D_{xy} = \min \{ (c_{xa} + D_{ay}), (c_{xb} + D_{by}), (c_{xc} + D_{cy}), \dots \}$$

- Update an existing least cost with a least cost through an intermediary node, such as z, if the later is shorter.
- The equation is

$$D_{xy} = \min \{D_{xy}, (C_{xz} + D_{zy})\}$$



- It is adaptive or dynamic router
- A distance vector routing algorithm operates by having each router **maintain a table** (i.e a vector) giving the best known distance.
- These tables are updated by **exchanging information with the neighbors**.
- Every router knows the best link to reach each destination.
- Each router maintains a table called “Vector”
- Send **ECHO** packet to neighbors and receive response time ASAP and record it.



## Exchanging vectors

eventually stabilizes the system and allows all nodes to find the ultimate least cost between themselves and any other node.

Regular updates between routers communicate topology changes.

Figure 20.6 Updating distance vectors

New B	Old B	A
A 2	A 2	A 0
B 0	B 0	B 2
C 5	C 5	C $\infty$
D 5	D $\infty$	D 3
E 4	E 4	E $\infty$
F $\infty$	F $\infty$	F $\infty$
G $\infty$	G $\infty$	G $\infty$

$$B[ ] = \min (B[ ], 2 + A[ ])$$

a. First event: B receives a copy of A's vector.

New B	Old B	E
A 2	A 2	A 8
B 0	B 0	B 4
C 5	C 5	C $\infty$
D 5	D 5	D 5
E 4	E 4	E 0
F 6	F $\infty$	F 2
G $\infty$	G $\infty$	G $\infty$

$$B[ ] = \min (B[ ], 4 + E[ ])$$

b. Second event: B receives a copy of E's vector.

Note:

X[ ]: the whole vector

Tables are updated by exchanging information with neighbors {at specific Interval}

Need to remember that after updating a node, it immediately send its **updated vector to all neighbors**. Even if its neighbors have received the previous vector, the updated one may help more

Figure 20.6 Updating distance vectors

New B	Old B	A
A 2	A 2	A 0
B 0	B 0	B 2
C 5	C 5	C $\infty$
D 5	D $\infty$	D 3
E 4	E 4	E $\infty$
F $\infty$	F $\infty$	F $\infty$
G $\infty$	G $\infty$	G $\infty$

$$B[ ] = \min(B[ ], 2 + A[ ])$$

a. First event: B receives a copy of A's vector.

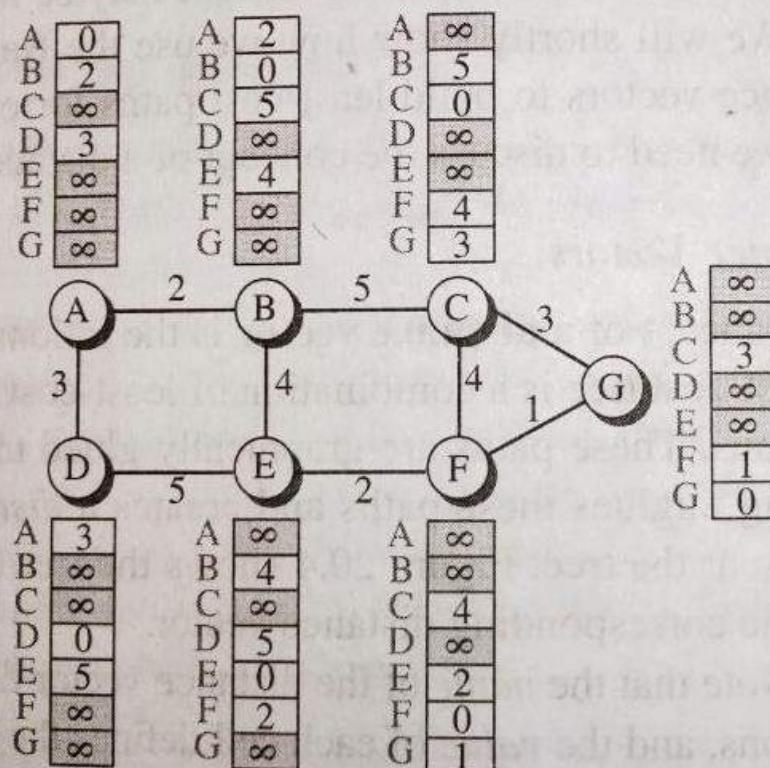
New B	Old B	E
A 2	A 2	A $\infty$
B 0	B 0	B 4
C 5	C 5	C $\infty$
D 5	D 5	D 5
E 4	E 4	E 0
F 6	F $\infty$	F 2
G $\infty$	G $\infty$	G $\infty$

$$B[ ] = \min(B[ ], 4 + E[ ])$$

b. Second event: B receives a copy of E's vector.

Note:  
X[ ]: the whole vector

The first distance vector for an internet



---

**In distance vector routing, each node shares its routing table with its immediate neighbors periodically and when there is a change.**

---

- Problem
- Count to infinity problem (looping issue)



- Distance Vector Routing algorithm often took too long to converge after the network topology changed  
(due to the count to infinity problem)

# Link-State Routing Algorithms

- Distance vector algorithm base nonspecific information about distant networks and no knowledge of distant routers.
- Whereas link-state routing algorithm maintains full knowledge of distant routers and how they interconnect.
- Links with **lower costs are preferred to links with higher costs**; if the cost of link is infinity, it means that link does not exist or have been broken.

- First Phase : Reliable flooding
  - Initial State : Each node knows the cost to its neighbors
  - Final State: Each node knows the entire graph (network topology)
- Second phase : route Calculation
  - Each node uses Dijkstra's algorithm on the graph and calculate routes to all nodes.

- Each router learns its own directly connected networks.
- Each router is responsible for meeting its neighbors on directly connected networks.
  - link state routers do this by exchanging **Hello packets** with other link-state routers on directly connected networks.
- Each router builds a **Link-State Packet (LSP)** containing the state of each directly connected link.
  - Identity of node
  - Cost of link
- Each **router floods the LSP to all neighbors**,
- Each router uses the database to construct a complete map of the topology and computes the best path to each destination network.

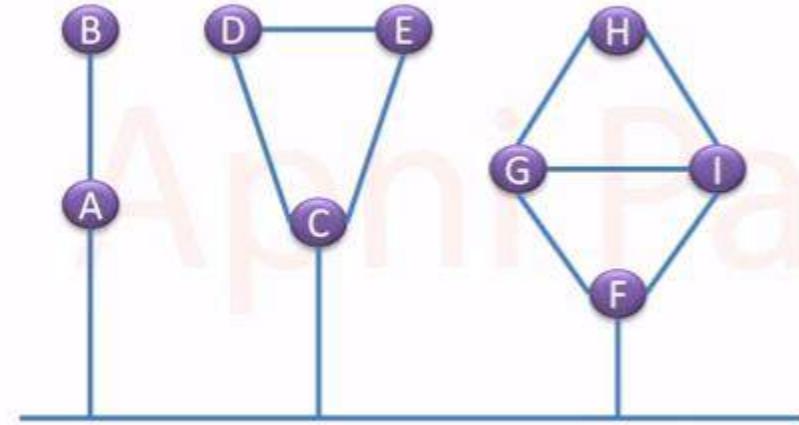
The idea behind link state routing is fairly simple and can be stated as five parts.

Each router must do the following things to make it work:

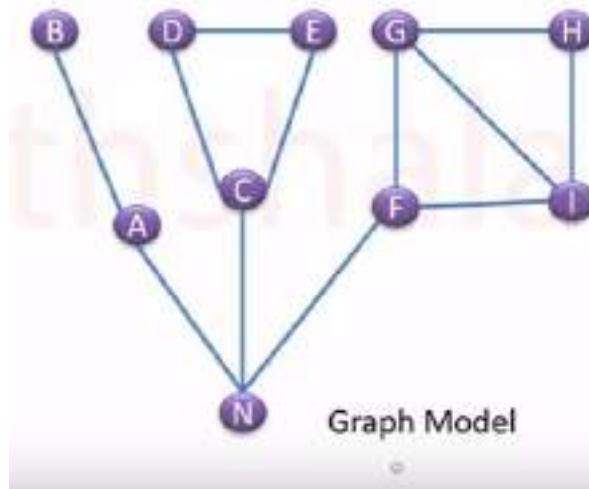
- Discover its neighbors and learn their network addresses.
- Set the distance or cost metric to each of its neighbors
- Construct a packet telling all it has just learned.
- Send this packet to and receive packets from all other routers.
- Compute the shortest path to every other router.

# Learning about the Neighbors

- Learn who its neighbors
  - Sends **HELLO** packet
  - Send back a reply giving its name
- Names must be globally unique



Broadcast LAN



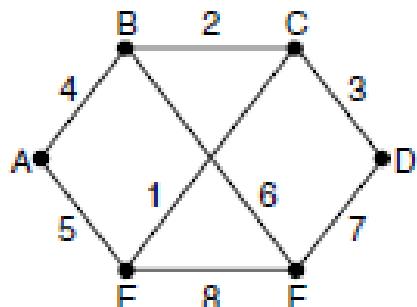
Graph Model

# Setting Link Costs

- The cost to reach neighbors
- A common choice is to make the cost inversely proportional to the bandwidth of the link.
  - Eg. 1. Gbps and 100Mbps Ethernet
- The most direct way to measure this delay is to send over the line a special ECHO packet that the other side back immediately.
- Round-trip time and dividing it by two

# Building Link Packets

- Building the link state packets is easy.
- The hard part is determining when to build them
  - One possibility is to build them periodically,
    - i.e **regular intervals**
  - To build them when some significant even occurs
    - **a line of neighbor going down or**
    - **coming back up again or**
    - changing its properties appreciably



(a)

Link	State	Packets
A	C	E
Seq.	Seq.	Seq.
Age	Age	Age
B 4	B 2	B 6
A 4	D 3	C 5
C 2	C 3	D 7
F 6	E 1	E 8

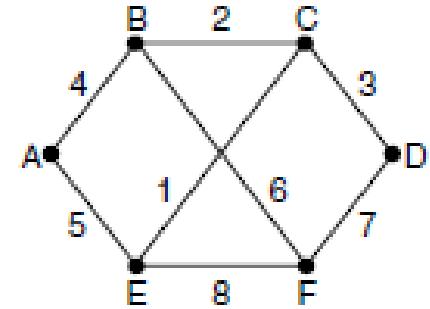
(b)

**Figure 5-12.** (a) A network. (b) The link state packets for this network.

**Sequence Packet** - Each packet contains a sequence no, this keep tracks how many times this packet has been sent.

**Age Packet** - shows the life of a packet in Network, after passing by one Hop it is decremented by one.

# Distributing the link state packets



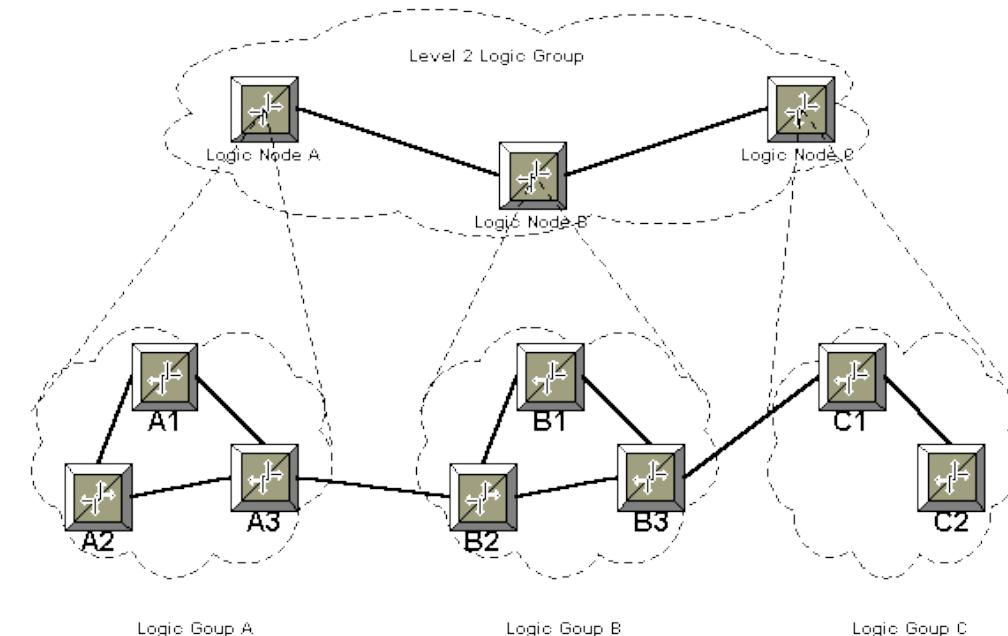
- **Flooding** is used to distribute the link state packets to all routers.
- To keep the flood in check,
  - each packet contains a sequence number
  - And incremented for each new packet sent.
- Routers keep track of all the (source router , sequence) pairs they see.
- When packets comes in, it is checked against the list packets already seen.
  - If it is new, it is forwarded on all lines.
  - If it is duplicate, it is discarded.
- If a packet with a sequence number lower than the highest one seen so far ever arrives, it is rejected as router receives more recent data.

# Computing the New Route

- Once a router has **accumulated a full set of link state packets**, it can construct the entire network graph because every link is represented.
- Dijkstra's algorithm can be run locally to construct the shortest paths to all possible destinations.

# Hierarchical Routing

- As networks grow in size, the router routing tables grow proportionally.
- Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them and more bandwidth is needed to send status reports about them.
- When hierarchical routing is used, the routers are divided into what we call **regions**.



- Example of how US packets comes to Nepal

```
C:\Windows\System32>TRACERT google.com

Tracing route to google.com [173.194.38.135]
over a maximum of 30 hops:

      1      1 ms      1 ms    <1 ms  192.168.14.1
      2      1 ms      1 ms    <1 ms  116.90.239.1
      3      2 ms      2 ms     1 ms  116.90.230.241
      4      2 ms      2 ms    12 ms  116.90.227.22
      5     46 ms     45 ms    45 ms  if-0-1-7.core1.MLU-Mumbai.as6453.net [209.58.105
.89]
      6    104 ms    103 ms   103 ms  if-1-2-1-0.tcore1.MLU-Mumbai.as6453.net [209.58.
105.226]
      7    104 ms    114 ms   106 ms  if-7-2.tcore1.CXR-Chennai.as6453.net [180.87.36.
34]
      8    132 ms    142 ms   155 ms  if-5-2.tcore1.SUW-Singapore.as6453.net [180.87.1
2.53]
      9    131 ms    103 ms   103 ms  72.14.220.134
     10   106 ms    119 ms   103 ms  66.249.95.124
     11   130 ms    103 ms   103 ms  72.14.233.79
     12   102 ms    103 ms   104 ms  sin04s01-in-f7.1e100.net [173.194.38.135]

Trace complete.
```

# Congestion Control Algorithms

- Too many packets present in the network causes packet delay and loss that degrades performance. This situation is called **congestion**.
- Informally: “too many sources sending too much data too fast for network to handle”.
- **Congestion Control** refers to the mechanisms and techniques to control the congestion and keep the load below the capacity.
- In data communications, a state in which the data traffic approaches or exceeds the channel’s capacity, resulting in a severe performance **degradation**, and possibility the loss of packets.
- **Flow Control** is concerned with whether receiver can accept the traffic from the sender **an issue between two ends**.

# Reasons

- Flood of traffic designed for the **same output line (queue fills and packets drop)**
  - Memory doesn't necessarily solve this problem.
- **Slow processors, or inefficient routing software.**
- Mismatch between parts of the system (several fast lines and one slow)

# General Principles of Congestion Control

- Solutions to the problem of Congestion is divided into two groups
  - Open-loop congestion Control (prevention)
  - Closed-loop congestion control (removal)

# Open-loop congestion Control

- Attempt to solve the problem by **good design** to make sure it does not occur in the first place.
- Tools for doing open-loop control include deciding when to **accept new traffic**, **deciding when to discard packets and which ones**, and making scheduling decisions at various points in the network.
- Some congestion prevention policies are :
  - **Retransmission Policy** : The **retransmission policy** and **retransmission times** must be **designed to optimize efficiency and prevent congestion**.
  - **Window Policy** : Selective repeat window is better than Go Back N window for congestion control.
  - **Acknowledgment Policy** : The acknowledgment policy imposed by the receiver may also affect congestion. If the receiver **does not acknowledge every packet it receives**, it may slow down the sender and help to prevent congestion.
  - **Discarding policy** : A good discarding policy by the router may prevent congestion and at the same time may not harm the integrity of the transmission.

- **Closed loop:** Closed loop solutions are based on the concept of a feedback loop.
- Feedback loops have three parts when applied to congestion control.
  - monitor the system to detect when congestion occurs
  - pass this information to places where actions can be taken.
  - adjust system operation to correct the problem.

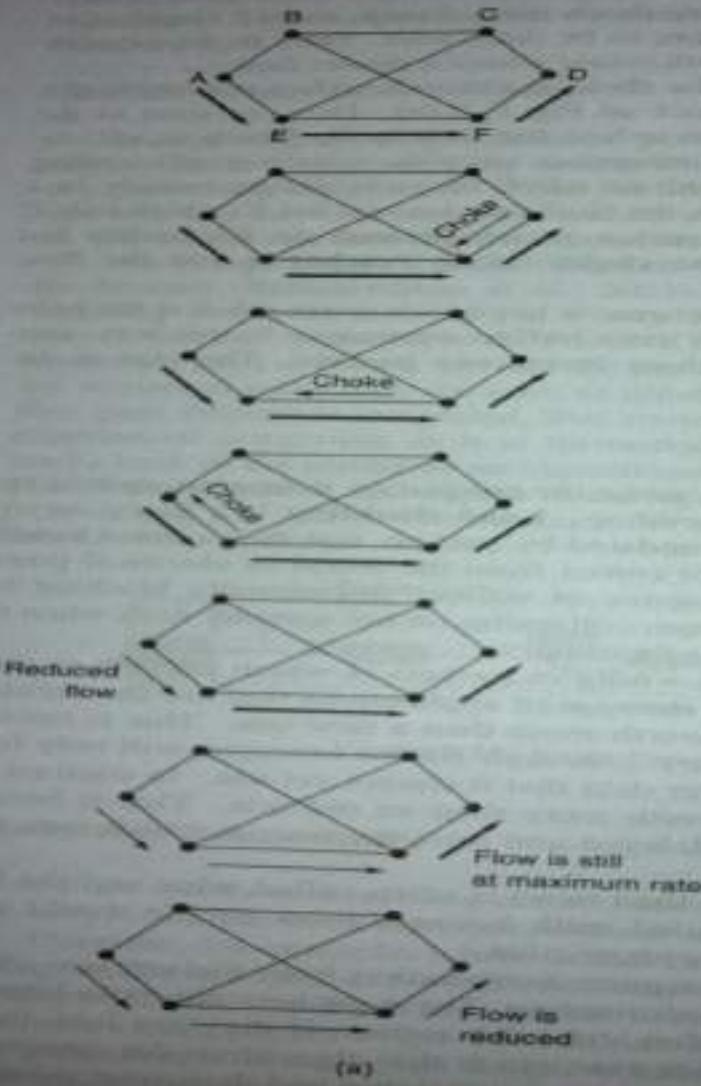
# Congestion Control in Virtual Circuit Subnet

- One technique that is widely used to keep congestion is **admission control.**
  - Once congestion has been signaled, **no more virtual circuits** are set up until the problem has gone away. (transport layer connections fails)
  - This is better than the alternative, as letting more people in when the network is busy just makes matters worse.
- An alternative approach is to allow new virtual circuits but carefully route all new virtual circuits around problem areas.
- Another strategy relating to virtual circuits is to settle an agreement between the host and subnet when a virtual circuit is set up.
  - Agreement normally specifies the **volume and shape of the traffic, quality of service required**

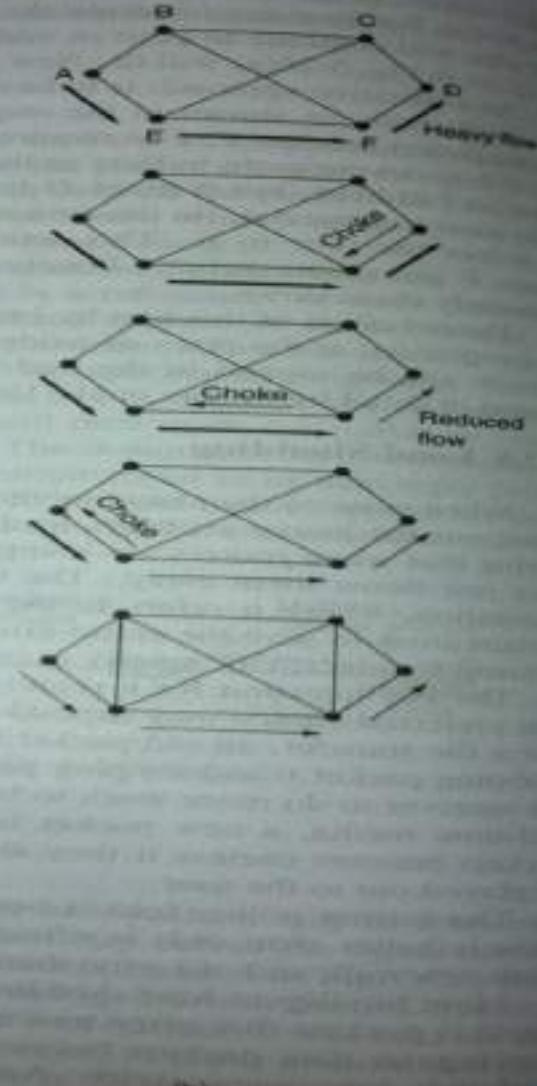
# Congestion Control in Datagram Subnet

- **Choke Packets**
  - The most direct way to notify a sender of congestion is to tell it directly.
  - It is a control packet generated at a congested node and transmitted back to the source node to reduce traffic X percent .
  - We can say 50% . That packet is called choke packet.
- **Problem**
  - The time to reach from destination to source might take long to reduce the flow immediately.

- At high speeds or **over long distances**, sending a choke packet to the source hosts does not work well because the reaction is so slow.
- **An alternative approach** is to have the choke packet take effect at every hop.
- **Solution: Hop by Hop choke Packet**
  - Here Choke packet works at each hop (router) it passes until it reaches destination, conveying order to reduce the data flow.



(a) A choke packet that affects only the source.



(b) A choke packet that affects each hop it passes through.

- Here, as soon as the choke packet reaches F, F is required to reduce the flow to D.
- Doing so will require F to devote more buffers to the flow, since the source is still sending away at full blast, but D immediate relief.

**Comparing the two methods it is seen that **choke packet** method doesn't give a relief in the process of congestion control, but **Hop by Hop choke method provides immediate relief at every hop it takes.****

# Load Shedding

- The term comes from the world of electric power generation.
- When demand of power is higher than supply.
- Load shedding is applied.
- When routers are being inundated(flood) by packets that they cannot handle then just **throw them away**.
- The preferred choice may depend on the type of applications that use the network.
  - For **a file transfer**, an **old packet** is worth more than a new one.
  - For **real-time media**, a **new packet** is worth more than an old one.
- An example is packets that carry routing information. These packets are more important than regular data packets because they establish routes; if they are lost, the network may lose connectivity.

# Wine Policy

- For file transfer, an old packet is worth more than one because dropping packet 6 and keeping packets 7 through 10 will cause a gap at the receiver that may force packets 6 through 10 to be retransmitted.

# Milk Policy

- For multimedia, a new packet is more important than an old one. Thus “fresher is better”.
- Therefore a new packet is more intelligent discard policy can be decided depending on the applications.
- To implement an intelligent discard policy, applications must mark their packets in priority classes to indicates how important they are.

# RED (Random Early Detection)

- When source does not receive acknowledgement, then the source automatically reduces the number of packets.
- To determine when to start discarding, routers maintain a running **average of their queue lengths**.
- When the **average queue length on some link exceeds** a threshold, the link is said to be congested and a small fraction of the packets are dropped at random.
- Please go to the page number 421 of Computer network by Tanenbaum

# Jitter

- The variation (i.e Standard Deviation) in the packet arrival time is called the jitter.
- Variation in arrival time for consecutive packets will lead to **high jitter result to uneven quality** especially in sound and video.
- When a packet arrives at a router, the router checks to **see how much the packet is behind or ahead of its schedule**. This information is stored in the packet and update at each hop.
- If the packet is ahead of schedule than the router will try to send it out as quickly as possible. This will helps in keeping the average delay per packet constant and will avoid time jitter.

- In some applications, such as **video on demand**, buffering at the **receiver** and then **fetching data for display** instead of from the network in real time **can eliminate jitter**.
- However, **Internet telephony** and **video-conferencing** the delay inherent in buffering is not acceptable.

# Quality of Service

- Till now we tried to designed to reduce congestion and to improve network performance.
- There are **some applications and customers** that demand stronger performance guarantees from the network.
  - A stream of packets from a source to Destination is called **flow**.
  - In **connection-oriented network** → all packets on the flow follows same route.
  - In **connectionless network** → They travel in different route.
  - There are four parameter that characterize the QoS reliability, delay, jitter, bandwidth.

- **Reliability :**
  - Reliability is a characteristic that a flow needs.
  - Lack of reliability means losing a packet or acknowledgment.
  - Reliability is important in electronic mail , file transfer than audio or video
- **Delay :**
  - Source to destination delay is another flow characteristic.
    - Audio conference, video conference, remote login (delay must be minimum)
    - File transfer or email (concern about delay is less important)
- **Jitter**
  - Jitter is the variation in delay for packets belong to the same flow.
  - Real-time audio and video cannot tolerate high jitter.
- **Bandwidth:**
  - Different applications require different bandwidths. In video conferencing, we need millions of bits per second to refresh a color screen while total number of bits in an email may not reach even million.

# Technique of Achieving Good QoS.

- **Overprovisioning:**

Here we increase the **router capacity**, **buffer space** and **bandwidth** compared to the flow statistics.

Problem: It is expensive.

- **Buffering:**

- Flows can be buffered on the receiving side before being delivered.
- Buffering doesn't affect reliability or bandwidth but increases delay but it smooth out the jitter.
- For audio and video on demand, jitter is the main problem, so this technique helps a lot..

# Traffic Shaping

- Traffic Shaping is a mechanism to **control the amount of rate of the traffic sent to the network.**
- Traffic in data networks is **bursty**.
  - It typically arrives at non uniform rate as the traffic rate varies (video conference), users interact with applications.
- **Bursts of traffic** are more **difficult** to handle than **constant-rate traffic** because they can fill buffers and cause packets to be lost.
  - So, Traffic shaping is a technique for regulating the average rate and burstiness of a flow of data that enters the network.
- The goal is to allow applications to transmit a wide variety of traffic that suits their needs.

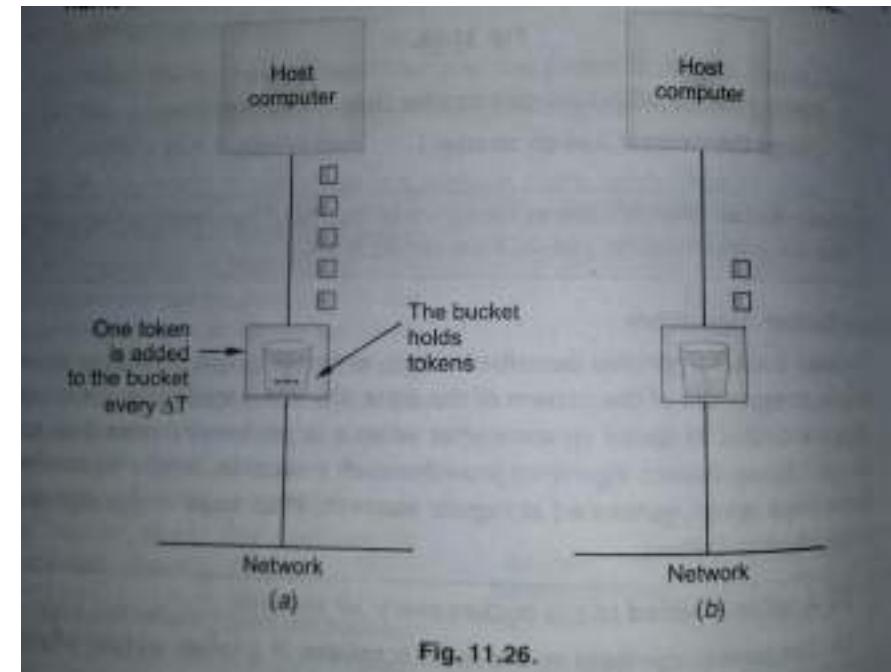
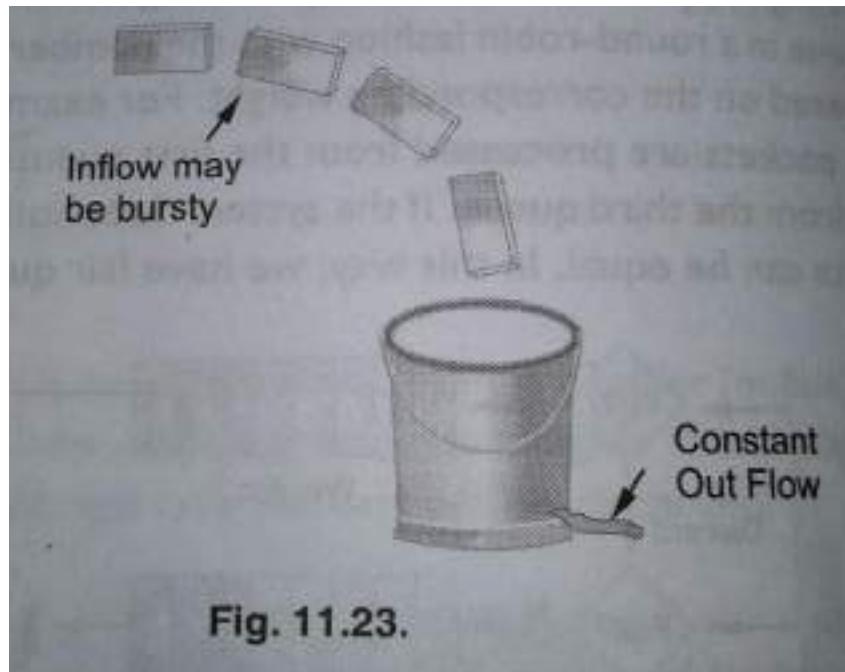
- Monitoring a traffic flow is called traffic policing.
- If packet stream violates its descriptor give penalty.
- Penalty will be
  - Drop packets that violate the descriptor
  - Give low priority to them
- Two traffic shaping techniques are as follows:
  - Leaky Bucket Algorithm
  - Token Bucket Algorithm

# Leaky Bucket Algorithm

- It is the **algorithm used to control congestion** in network traffic.
- Let's consider a bucket with a **small hole** at the bottom;
- If a bucket has a small hole at the bottom, that water leaks from the bucket at a constant rate, as long as there is water.
- Here input rate can vary but output rate is constant.

# Leaky Bucket Disadvantage

- Leaky bucket is very restrictive
- Bucket becomes empty if host is not sending for a while.
- If host has **burst data**, it allows only an average rate.



# The Token Bucket Algorithm

- The tap is running at **rate R** and the bucket has a **capacity of B**.
- Take out water or tokens from the bucket (rather than putting).
- If bucket is empty, we must wait until more tokens arrive before we can send the packet.
- This algorithm is token bucket algorithm

# The Token Bucket Algorithm

- The leaky bucket algorithm described above, **enforces a rigid pattern** at the output stream, irrespective of the pattern of the input.
- For many applications, it is better to allow the output to **speed up somewhat** when **large bursts arrive**, so a more flexible algorithm is needed, preferably one that never loses data.
- Token Bucket algorithm provides such a solution.
- **limit the long-term rate of a flow** but **allow short term bursts** up to a maximum regulated length to pass through unaltered and without suffering any artificial delays.

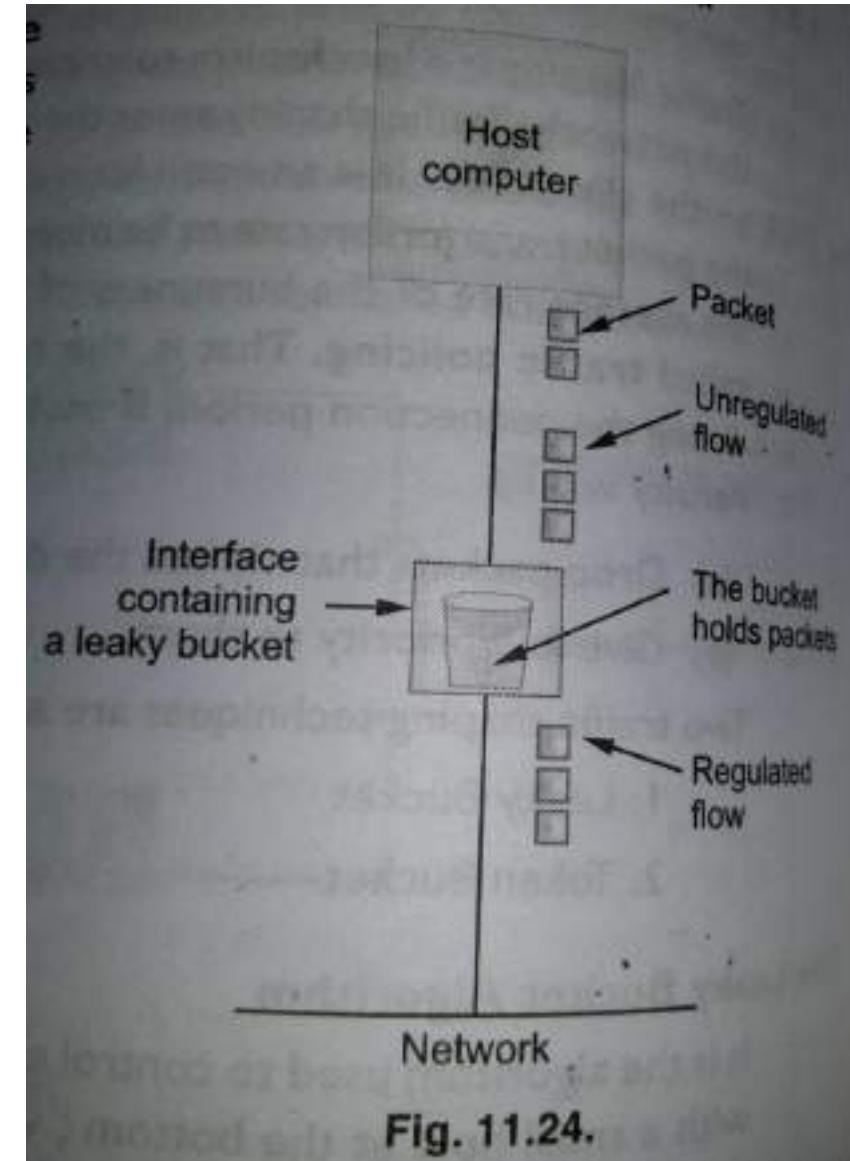


Fig. 11.24.

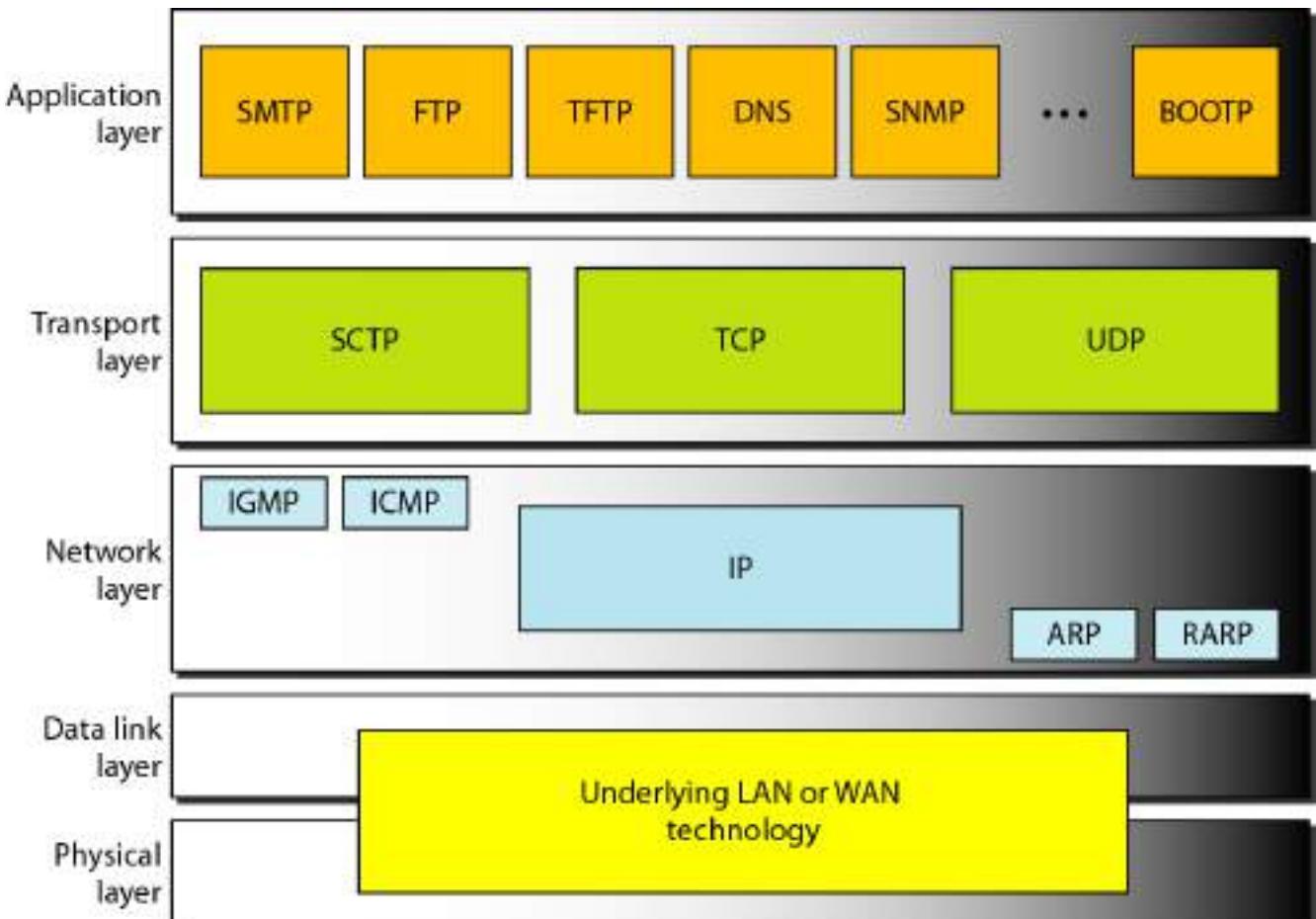
- Lets consider that
  - A computer can produce data at up to 1000 Mbps and that the first link of the network also runs at this speed.
  - This pattern is bursty,
  - The average rate over one second is 200 Mbps, even through the host sends a burst of 16,000 KB at the top speed of 1000 Mbps.

- Routers can accept data at the **top speed only for short intervals** until their buffer fills up.
- The buffer size is 9600 KB, smaller than the traffic burst.
- The implication is that if traffic is sent in this pattern, some of it will be dropped in the network.

- Main steps of the algorithm can be described as follows:
  - A token is added to the bucket every ...T seconds
  - The bucket can hold at the most *b* tokens. If a token arrives when the bucket is full it is discarded.
  - When a packet of n bytes arrives, n tokens are removed from the bucket, and the packet is sent to the network.
  - If fewer than n tokens are available , no tokens are removed from the bucket, and the packet is considered to be **non-conformant**.

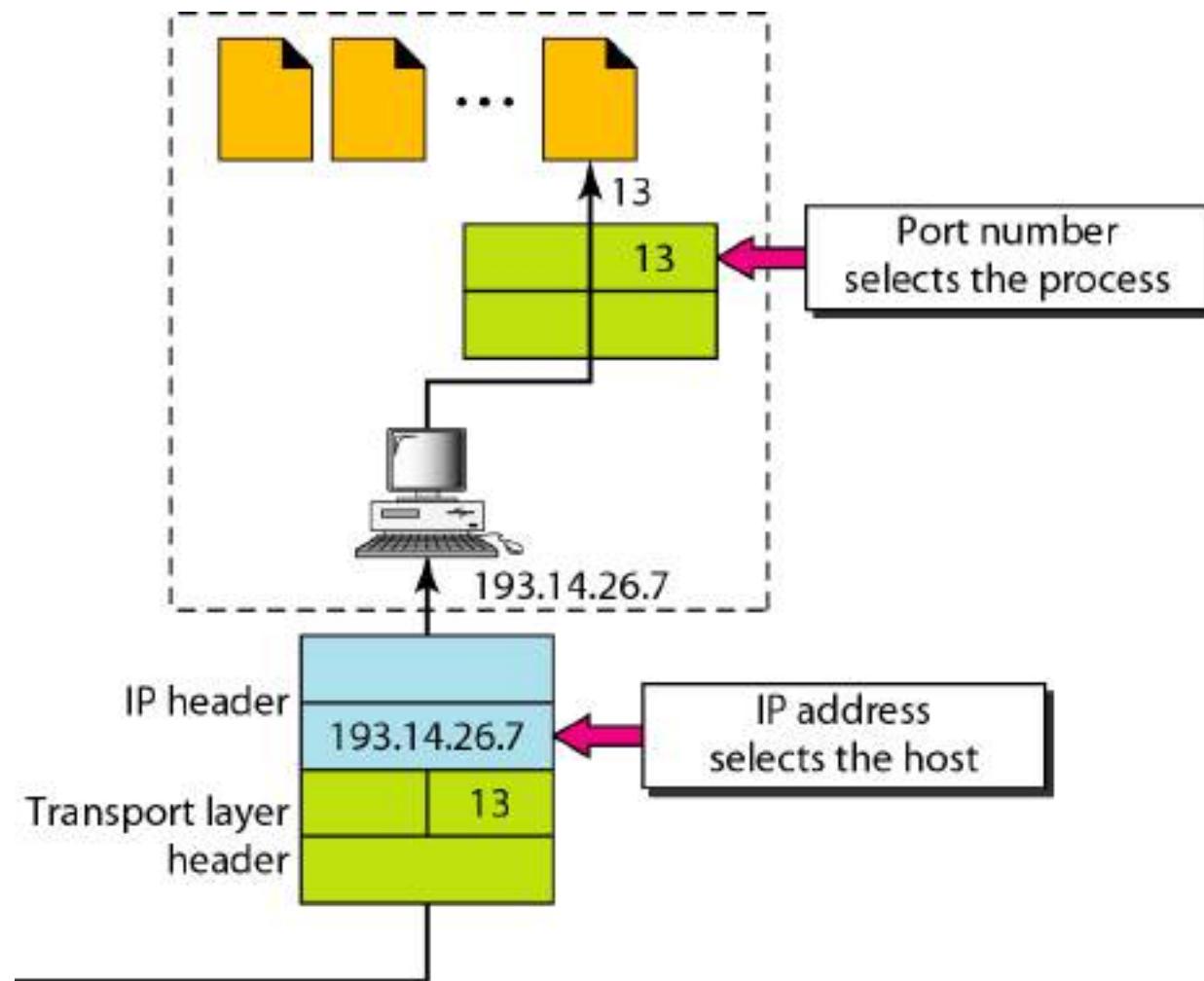
# Transport Layer

- The transport layer in the TCP/IP suite is located between the **application layer** and the **network layer**.
- The transport layer is responsible for **process-to-process delivery** of the entire message.
  - Process-to-process delivery:** logical communication between processes on different hosts
- The transport layer acts as a **liaison between a client program and a server program**.



- The most common form of Process to Process delivery is **Client/Server Model**.
- A process on the **local host called a client**, needs service from a process usually on the **remote host called the server**.
- Similar to Network Layer that uses IP address, at Transport Layer we need Port number which the transport layer gets from Network Layer.
- Port number ranges from 0 to 65,535.

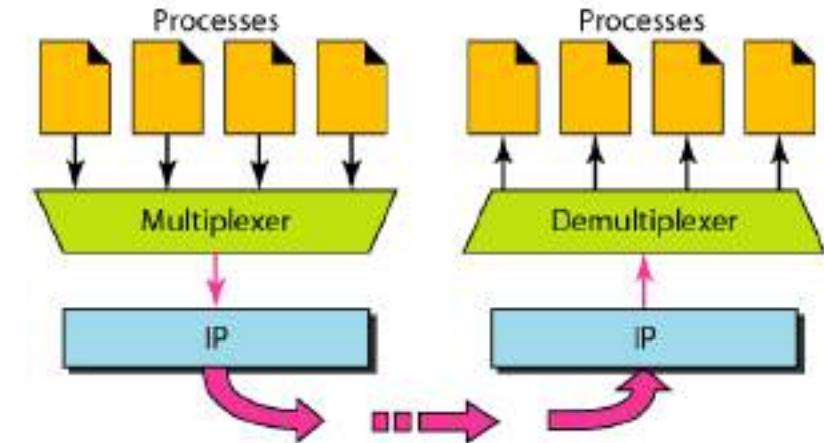
**Figure 23.3** IP addresses versus port numbers



# Multiplexing and Demultiplexing

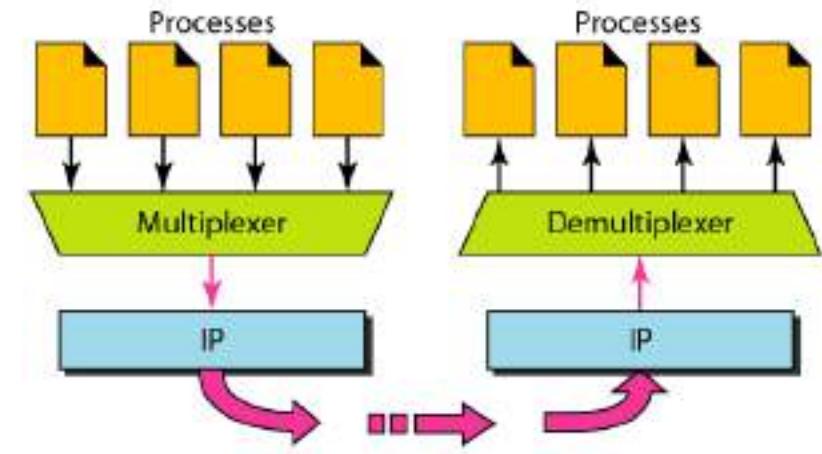
## Multiplexing :

- At the sender site, there may be several processes that need to send packets.
- However, there is only one transport layer protocol at any time.
- This is **many-to-one relationship** and requires multiplexing.
- The protocol accepts messages from different processes, differentiated by their assigned port numbers.
- After adding the header, the transport layer passes the packet to the network layer.



## Demultiplexing

- At the receiver site, the relationship is **one to many** and requires demultiplexing.
- The transport layer receives datagrams from the network layer.
- After error checking and dropping of the header, the transport layer delivers each message to the appropriate process based on the port.
- One to Many relationship



# Connectionless Vs Connection-Oriented Service

- **Connectionless Service**

- In a connectionless service, the packets are sent from one party to another with no need for connection establishment or connection release.
- The packets are not numbered; they may be delayed or lost or may arrive out of sequence.
- There is no acknowledgement either.
- User datagram protocol (UDP) is connectionless

- **Connection-Oriented Service**
  - In a connection-oriented Service, a connection is first established between the sender and the receiver.
  - Data are transfer.
  - At the end, the connection is released.
  - For example, TCP and SCTP are connection-oriented protocols.

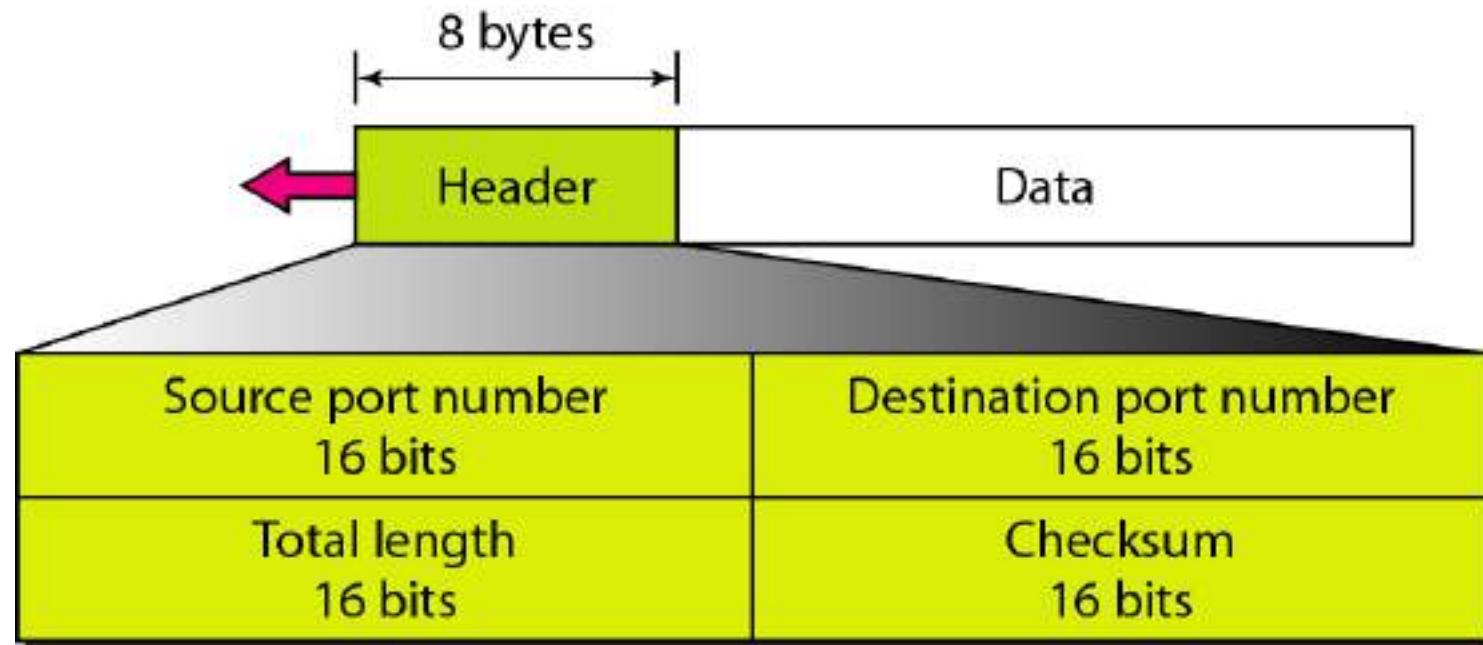
# Protocols

- User Datagram Protocol (UDP)
- Transmission Control Protocol (TCP)
- Stream Control Transmission (SCTP)

# UDP : User Datagram Protocol

- The User Datagram Protocol (UDP) is called a **connectionless, unreliable transport protocol.**
- It performs very **limited error checking.**
- UDP is a **very simple protocol using a minimum of overhead.**
- If a process wants to send message and does not care much about reliability, it can use UDP.
- Sending a **small message by using UDP takes much less interaction** between the sender and receiver than using TCP.

**Figure 23.9 User datagram format**



# UDP Services

- **Process –to–Process Communication :**

- UDP provides process-to process **communication using socket address**: a combination of IP address and port numbers :

- **Socket Address :**

- The combination of **IP address and port number** is called a **socket address**.

- **Client Socket Address**

- Defines the client process uniquely

- **Server Socket Address**

- Defines the server process uniquely

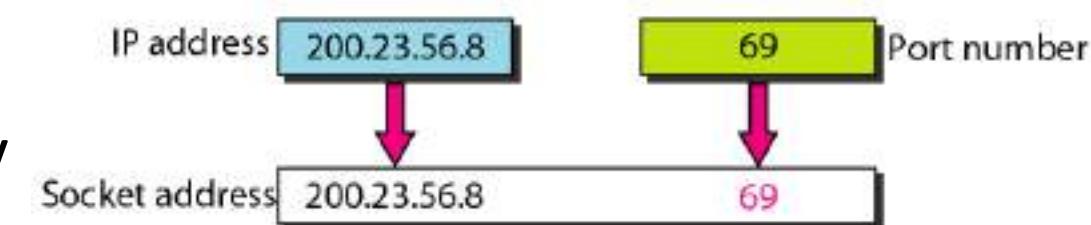
- **Connectionless Services:**

- UDP provides a connectionless services.

- User datagram sent by UDP is an independent datagram.

- There is no relationship

- Userdatagram are not numbered.



- **Flow Control:**

- UDP is very simple protocol. **There is no flow control.**
- The lack of flow control means that the process using UDP should provide for this service; if needed

- **Error Control**

- No error control mechanism in UDP except for the checksum.
- The lack of error Control means that the process using UDP should provide for this services, if needed.

- **Checksum**

- **Congestion Control**

- It doesn't provide congestion control.
- UDP assumes that the packets sent are small and sporadic and cannot create congestion in the network.

## • Encapsulation and Decapsulation

- To send a message from one process to another, the UDP protocol encapsulates and decapsulates messages.

## • Queuing

- In UDP, queues are associated with ports
- At the client site, when a process starts, it requests a port number from the operating system.
- Try c:\netstat -an*

C:\Windows\system32\cmd.exe			
TCP	192.168.14.109:49891	58.27.22.8:443	TIME_WAIT
TCP	192.168.14.109:49892	173.194.36.94:443	ESTABLISHED
TCP	192.168.14.109:49895	58.27.22.26:443	ESTABLISHED
TCP	192.168.14.109:49896	202.52.233.170:443	ESTABLISHED
TCP	192.168.14.109:49897	58.27.22.26:443	TIME_WAIT
TCP	192.168.14.109:49898	202.51.65.23:443	ESTABLISHED
TCP	192.168.14.109:49899	88.221.14.66:443	ESTABLISHED
TCP	192.168.14.109:49900	88.221.14.66:443	TIME_WAIT
TCP	192.168.14.109:49901	58.27.22.65:443	ESTABLISHED
TCP	192.168.14.109:49902	88.221.14.67:443	ESTABLISHED
TCP	192.168.14.109:49903	58.27.22.65:443	TIME_WAIT
TCP	192.168.14.109:49904	88.221.14.67:443	TIME_WAIT
TCP	192.168.14.109:49914	58.26.1.80:443	ESTABLISHED
TCP	192.168.14.109:49915	58.26.1.80:443	ESTABLISHED
TCP	192.168.14.109:49920	58.27.22.66:443	TIME_WAIT
TCP	192.168.14.109:49921	58.27.22.91:443	TIME_WAIT
TCP	192.168.14.109:49922	202.51.65.29:80	ESTABLISHED
TCP	[::]:135	[::]:0	LISTENING
TCP	[::]:445	[::]:0	LISTENING
TCP	[::]:49152	[::]:0	LISTENING
TCP	[::]:49153	[::]:0	LISTENING
TCP	[::]:49154	[::]:0	LISTENING
TCP	[::]:49157	[::]:0	LISTENING
TCP	[::]:49158	[::]:0	LISTENING
TCP	[::]:49156	[::]:0	LISTENING
UDP	0.0.0.0:1900	**:	
UDP	0.0.0.0:5355	**:	
UDP	0.0.0.0:50000	**:	
UDP	0.0.0.0:53247	**:	
UDP	0.0.0.0:57039	**:	
UDP	0.0.0.0:57044	**:	
UDP	0.0.0.0:62577	**:	
UDP	127.0.0.1:1900	**:	
UDP	127.0.0.1:50966	**:	
UDP	127.0.0.1:53246	**:	
UDP	127.0.0.1:53254	**:	
UDP	127.0.0.1:53255	**:	
UDP	127.0.0.1:57041	**:	
UDP	127.0.0.1:57042	**:	
UDP	127.0.0.1:57043	**:	
UDP	192.168.14.109:137	**:	
UDP	192.168.14.109:138	**:	
UDP	192.168.14.109:1900	**:	

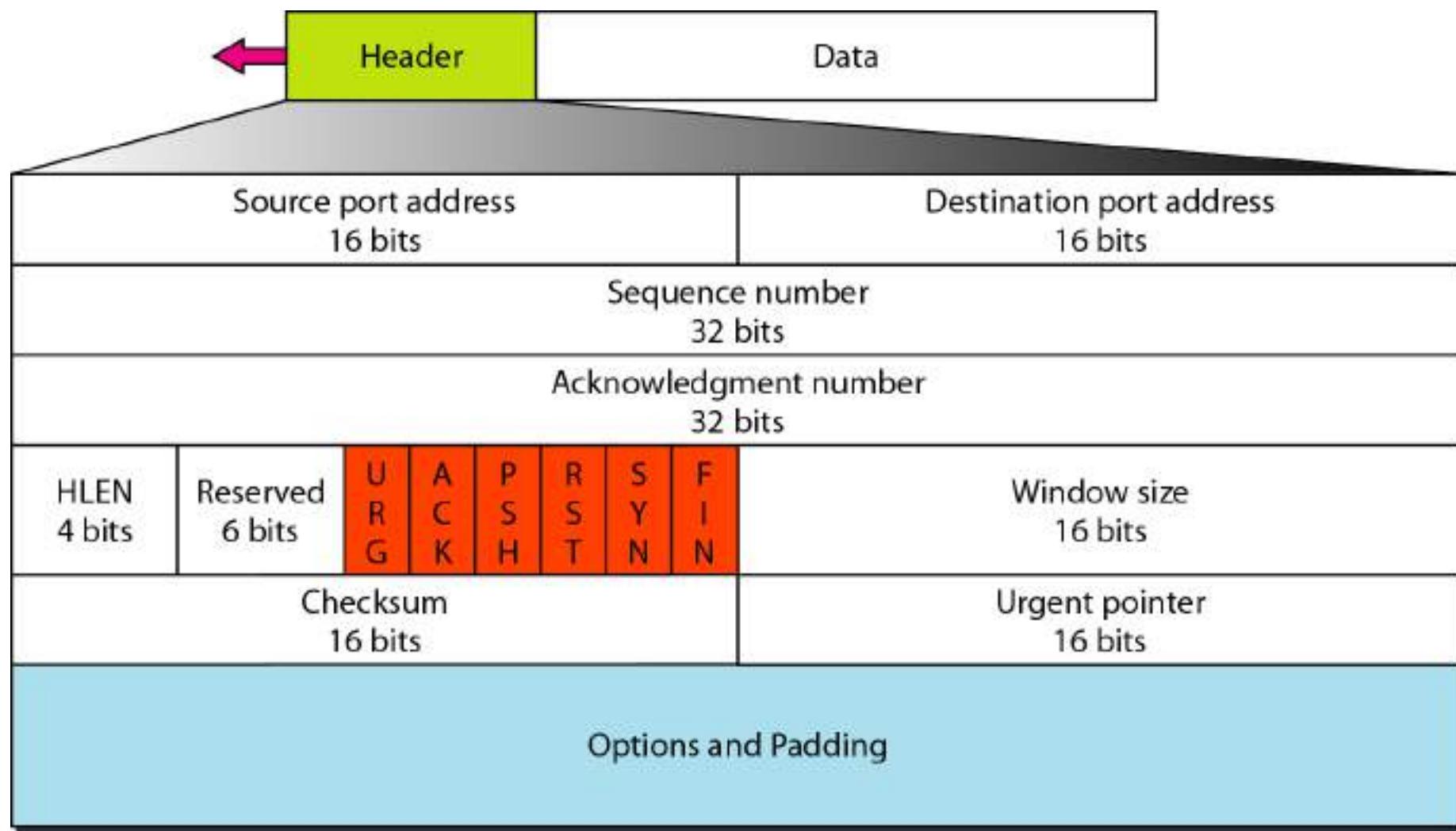
# UDP applications

- **Connectionless Services**
  - A client application needs to send a **short request** to a server and to receive a short response.
  - The overhead to establish and **close a connection** may be significant in this case.
  - In **connection oriented service**, to achieve the goal, at least 9 packets are exchanged between the client and the server;
  - In **connectionless service only 2 packets are exchanged**.
  - The connectionless service provides less delay; the connection-oriented service creates more delay;
- Example 24.3
  - DNS uses the services of UDP because client needs to send a short request to a server.
  - The request and response can each fit in one user datagram.

# TCP: Transmission Control Protocol

- It is a **connection-oriented protocol; reliable protocol.**
- Provides
  - connection establishment,
  - data transfer,
  - connection teardown phase.
- Uses checksum, retransmission of lost or corrupted packets. Cumulative and selective acknowledgment and timers.
- TCP uses flow and error control mechanisms at the transport level.

Figure 23.16 TCP segment format



# A TCP Connection

- TCP transmits data in **full-duplex mode**.
- Two parties get initialized and gets approved for connection and transfer for data.
- To establish the connection; the host perform **three way handshake**.
- In TCP, connection-oriented transmission requires three phases:
  - connection establishment,
  - data transfer and
  - connection termination.

- Connection Establishment through

## **Three-Way Handshaking**

- When a connection to a remote station is needed, an application will request TCP to place an OPEN call : passive and active
  - An **active Open** is made when a connection attempt to a remote network station is needed.
  - A **passive Open** is a call to allow connections to be accepted from a **remote station**.

Try c:\netstat -an

```
C:\Windows\system32\cmd.exe
TCP 192.168.14.109:49891 58.27.22.8:443 TIME_WAIT
TCP 192.168.14.109:49892 173.194.36.94:443 ESTABLISHED
TCP 192.168.14.109:49895 58.27.22.26:443 ESTABLISHED
TCP 192.168.14.109:49896 202.52.233.170:443 ESTABLISHED
TCP 192.168.14.109:49897 58.27.22.26:443 TIME_WAIT
TCP 192.168.14.109:49898 202.51.65.23:443 ESTABLISHED
TCP 192.168.14.109:49899 88.221.14.66:443 ESTABLISHED
TCP 192.168.14.109:49900 88.221.14.66:443 TIME_WAIT
TCP 192.168.14.109:49901 58.27.22.65:443 ESTABLISHED
TCP 192.168.14.109:49902 88.221.14.67:443 ESTABLISHED
TCP 192.168.14.109:49903 58.27.22.65:443 TIME_WAIT
TCP 192.168.14.109:49904 88.221.14.67:443 TIME_WAIT
TCP 192.168.14.109:49914 58.26.1.80:443 ESTABLISHED
TCP 192.168.14.109:49915 58.26.1.80:443 ESTABLISHED
TCP 192.168.14.109:49920 58.27.22.66:443 TIME_WAIT
TCP 192.168.14.109:49921 58.27.22.91:443 TIME_WAIT
TCP 192.168.14.109:49922 202.51.65.29:80 ESTABLISHED
TCP [::]:135 [::]:0 LISTENING
TCP [::]:445 [::]:0 LISTENING
TCP [::]:49152 [::]:0 LISTENING
TCP [::]:49153 [::]:0 LISTENING
TCP [::]:49154 [::]:0 LISTENING
TCP [::]:49157 [::]:0 LISTENING
TCP [::]:49158 [::]:0 LISTENING
TCP [::]:49156 [::]:0 LISTENING
UDP 0.0.0.0:1900 *:*
UDP 0.0.0.0:5355 *:*
UDP 0.0.0.0:50000 *:*
UDP 0.0.0.0:53247 *:*
UDP 0.0.0.0:57039 *:*
UDP 0.0.0.0:57044 *:*
UDP 0.0.0.0:62577 *:*
UDP 127.0.0.1:1900 *:*
UDP 127.0.0.1:50966 *:*
UDP 127.0.0.1:53246 *:*
UDP 127.0.0.1:53254 *:*
UDP 127.0.0.1:53255 *:*
UDP 127.0.0.1:57041 *:*
UDP 127.0.0.1:57042 *:*
UDP 127.0.0.1:57043 *:*
UDP 192.168.14.109:137 *:*
UDP 192.168.14.109:138 *:*
UDP 192.168.14.109:1900 *:*
```

Figure 23.18 Connection establishment using three-way handshaking

- The client sends the first segment, a **SYN segment** in which only the SYN flag is set.
- This segment is for synchronization of sequence numbers.
- The client chooses a **random number as the first sequence number** and sends this number to the server.
- The sequence number is called the **initial sequence number(ISN)**

A SYN segment cannot carry data, but it consumes one sequence number.

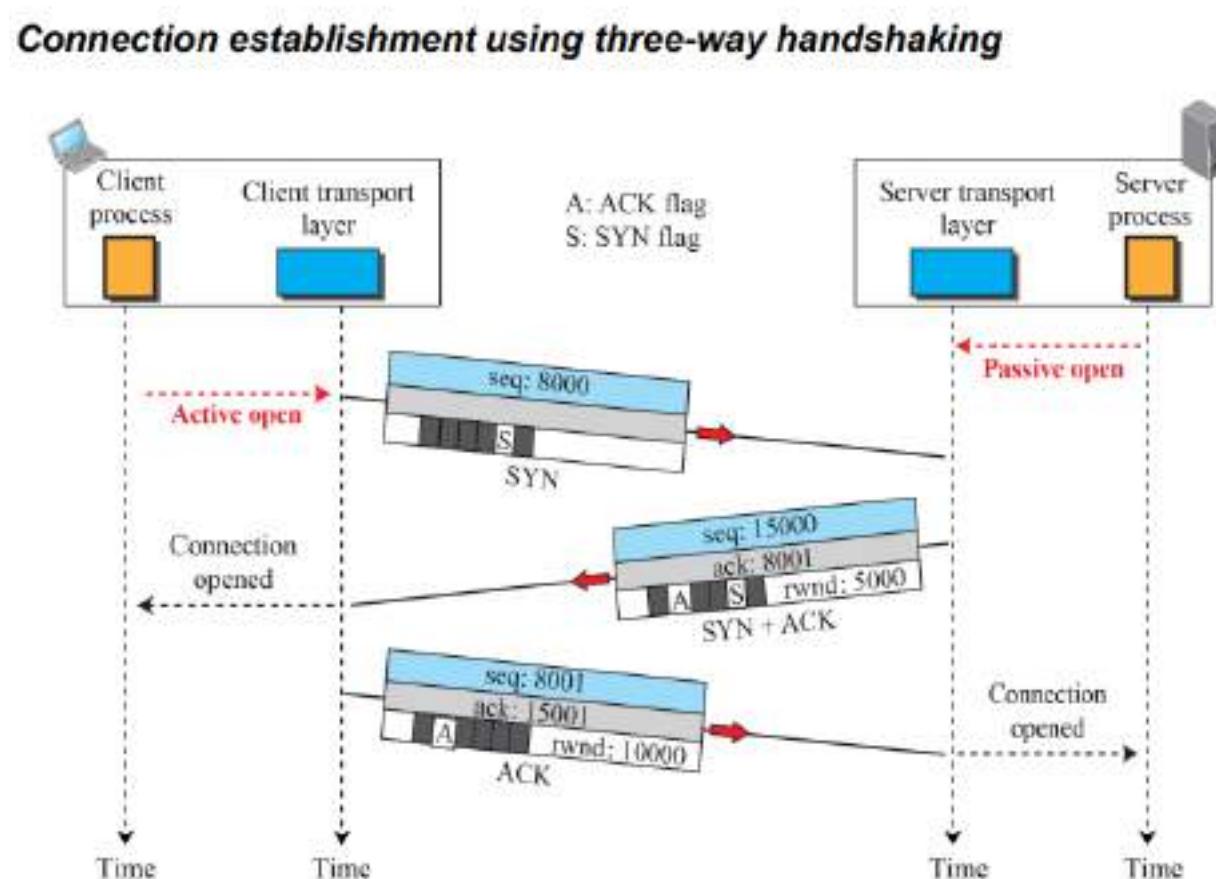


Figure 23.18 Connection establishment using three-way handshaking

- The server sends the second segment, a **SYN + ACK** segment with two flag bits set as: **SYN and ACK**.
- The server uses the **this segment to initialize a sequence number** for numbering the bytes sent from the server to the client.
- The server also acknowledges the receipt of the SYN segment from the client by setting the ACK flag.

A SYN + ACK segment cannot carry data, but does consume one sequence number.

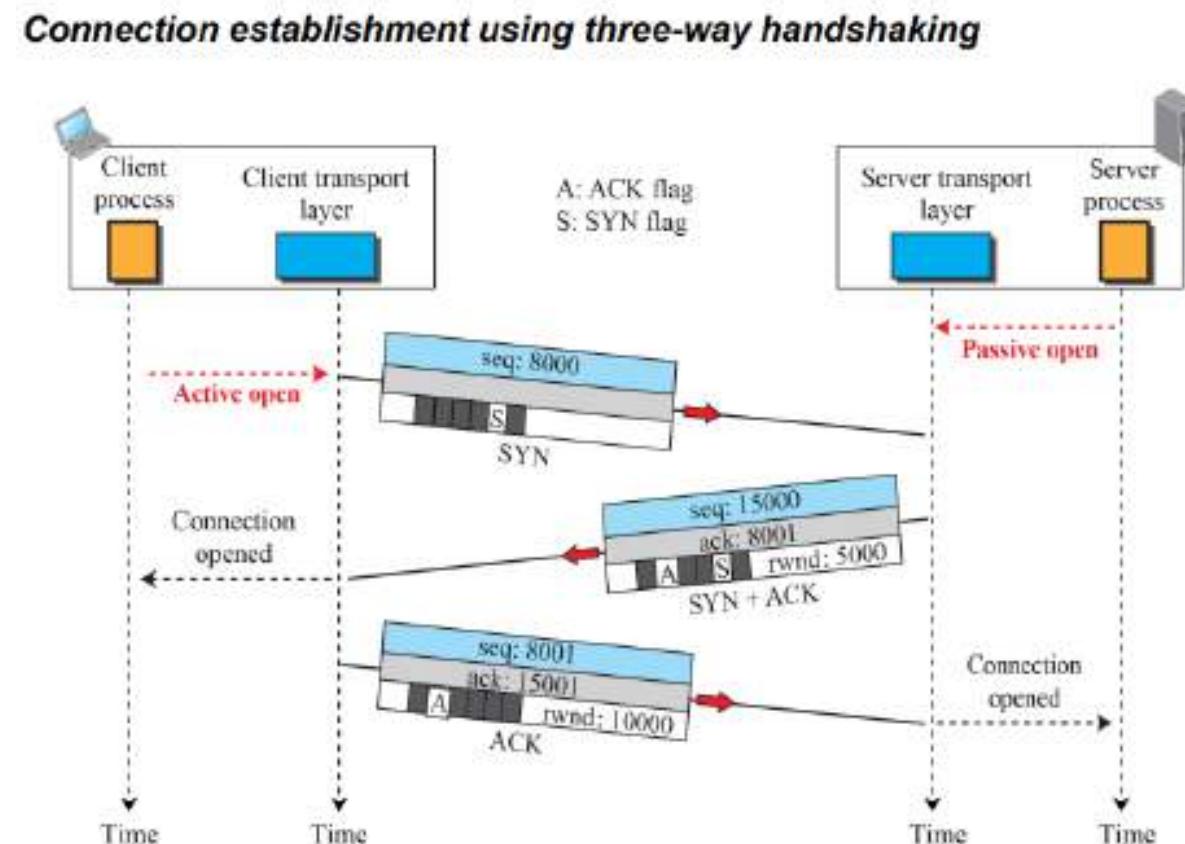
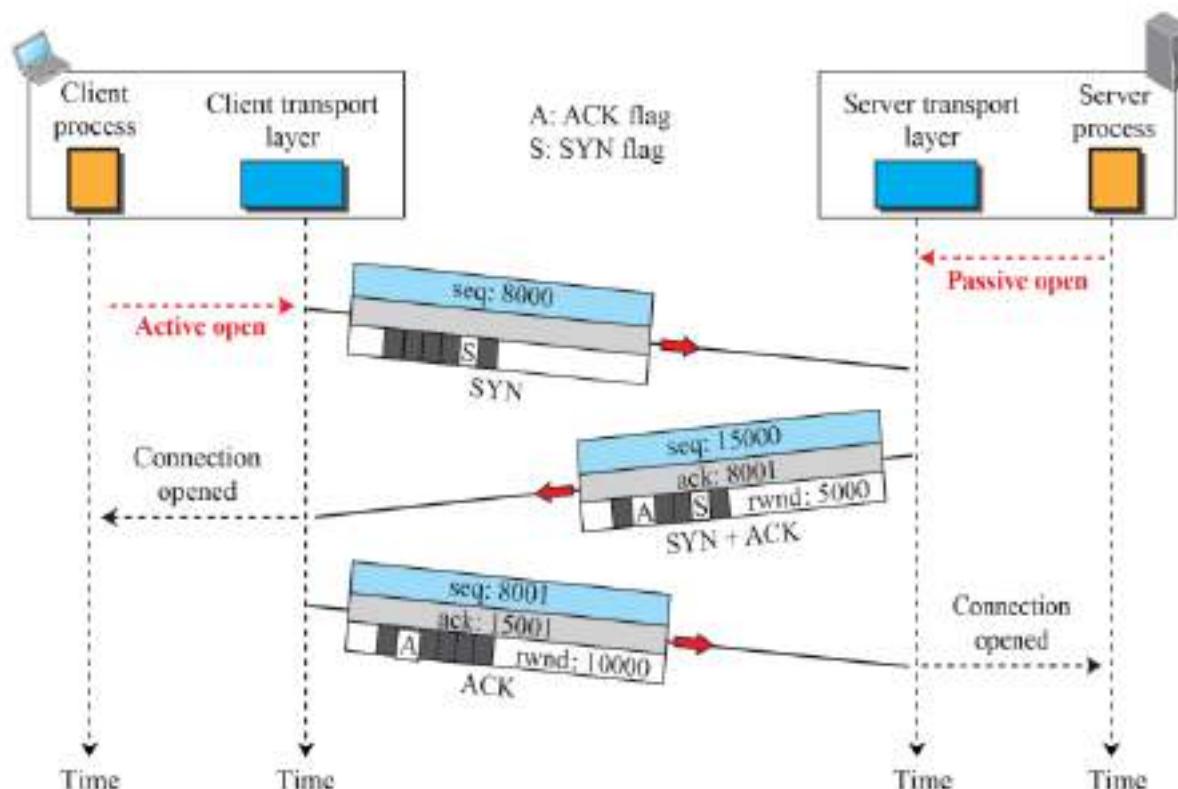


Figure 23.18 Connection establishment using three-way handshaking

The client sends the third segment. This is just an ACK segment.

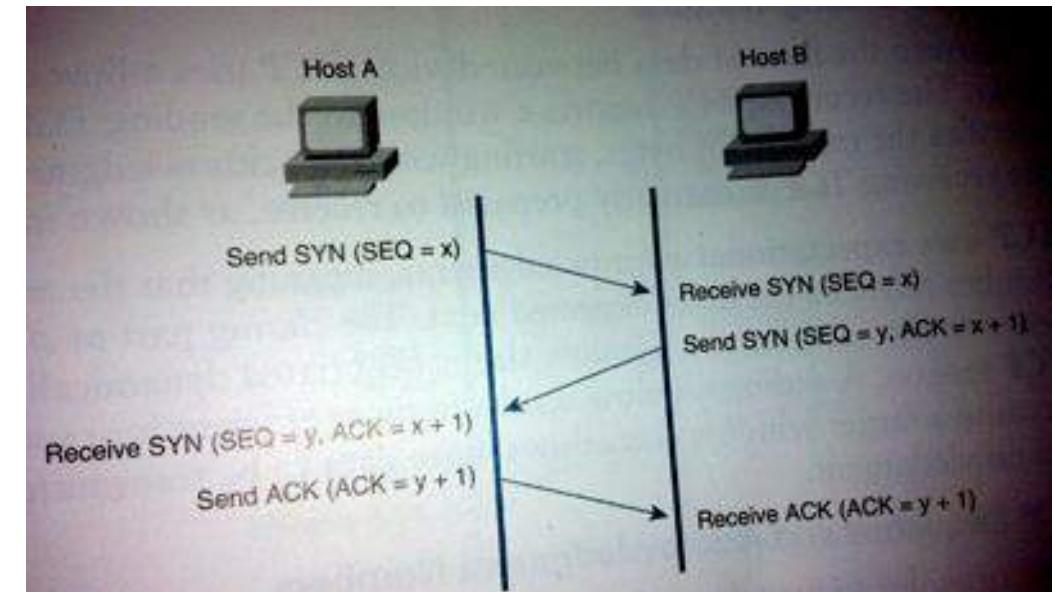
An ACK segment, if carrying no data, consumes no sequence number.

Connection establishment using three-way handshaking



## Connection Establish (Cont...)

- Host A chooses a sequence number  $x$ , and sends a connection request, TPDU (Transport Protocol Data Unit) to Host B
- Host B replies with an ACK=  $x+1$  and announces its own sequence number by  $y$ .
- Host A again sends acknowledgement ACK= $y+1$
- The sequence is like two people talking .
- The first person wants to talk to the second, so she says, “**I would like to talk with you**” (SYN).
- The second person responds, “**Good, I want to talk with you**” (SYN, ACK).



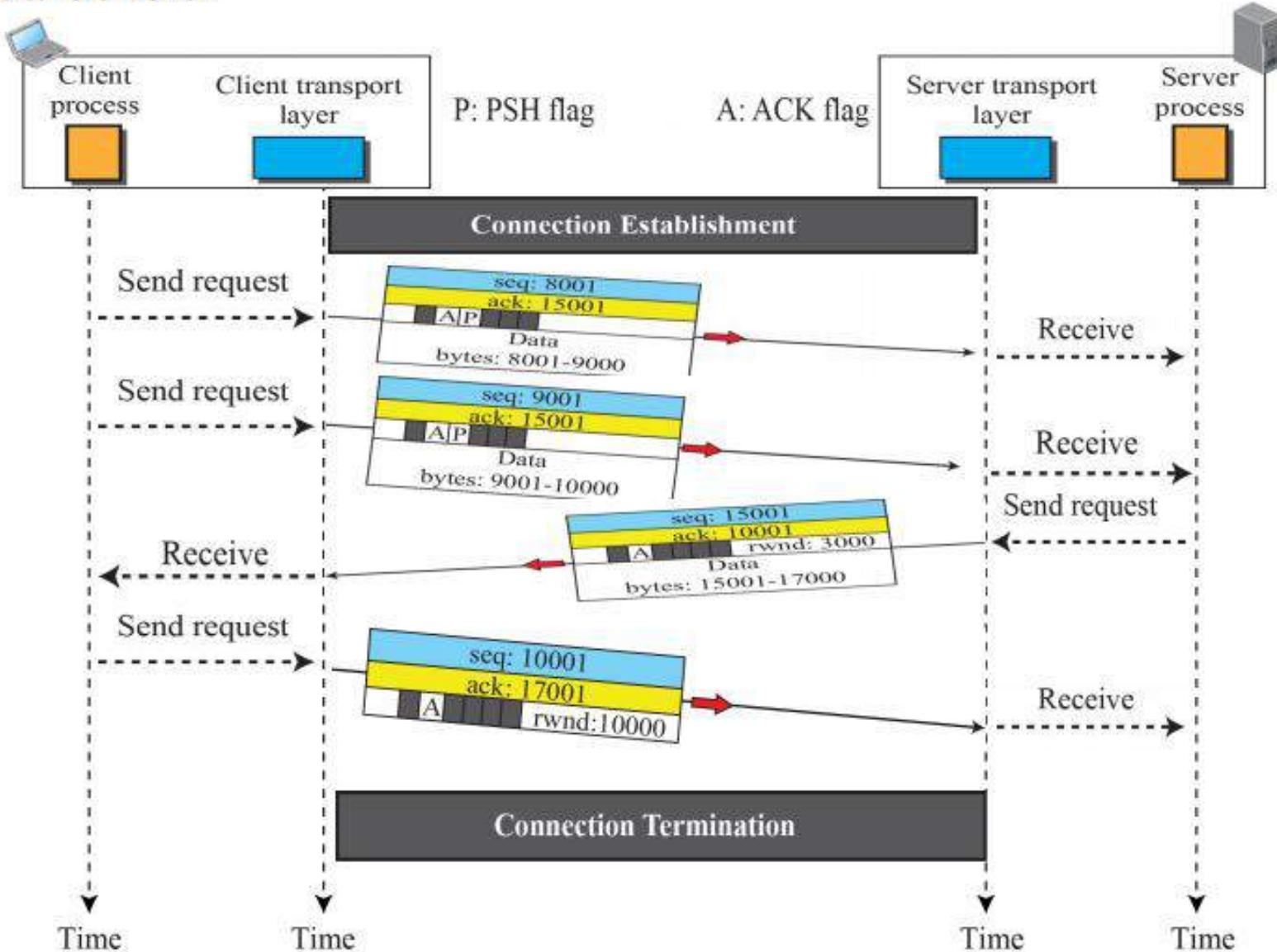
# SYN Flooding Attack

- Serious security problems called SYN flooding attack.
- One more **malicious attackers** send a large number of SYN segments to a server pretending that each of them is coming from a different client by faking the source IP addresses in the datagrams.
- The server assumes that the clients are issuing an active open, allocates the necessary resources, as a creating **transfer control block (TCB) tables** and **setting timers**.
- TCP **sever send then sends the SYN+ACK** segments to the fake clients, which are lost
- If, during this short period of time, the number of SYN segments is large, **the server eventually runs out resources** and may be unable to accept connection requests from valid clients.
- This SYN flooding attack belongs to a group of security attacks known as a **denial of service attack**.
  - Solution is limit of connection during specified of time
  - Identifying valid ip through a cookie

# Data Transfer

- After connection is established, **bidirectional transfer** can take place.
- The client and server can send data and acknowledgments in both directions.

## Data transfer



# Connection Termination

- When a Host A has **no more data to send in a stream**. Client starts a session called Active close.
- After receiving a close command from the client process, sends the first segment , A **FIN** segment in which the **FIN** flag is set.
- FIN segment can **include the last chunk of data** sent by the client or it can be just a control segment.

The FIN segment consumes one sequence number if it does not carry data.

*Connection termination using three-way handshaking*

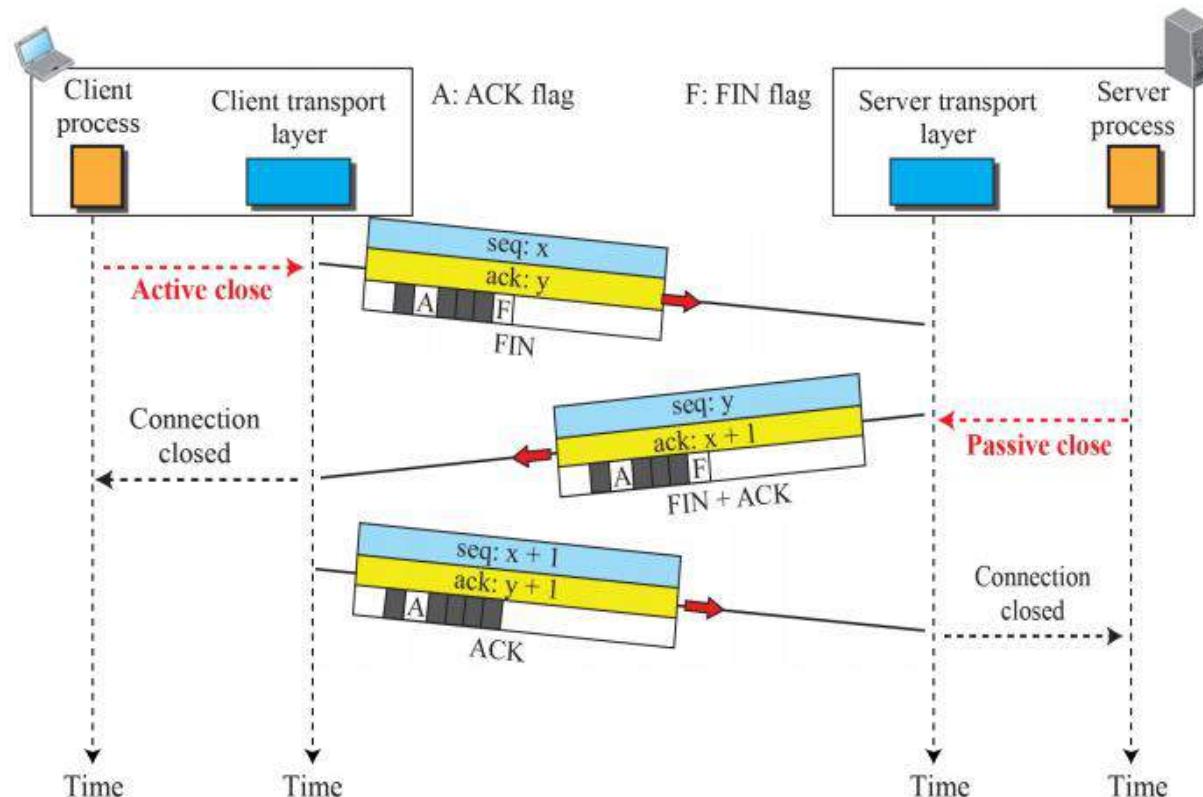


Figure 23.18 Connection termination using three-way handshaking

- The server sends segment, a **FIN + ACK** segment, to conform the receipt of the FIN segment from the client and at the same time to announce the closing of the connection in the other direction.
- This segment can also contain the last chunk of data from the server.
- If it doesn't carry data, it consumes only **one sequence** number because it needs to be acknowledgment.

The FIN + ACK segment consumes one sequence number if it does not carry data.

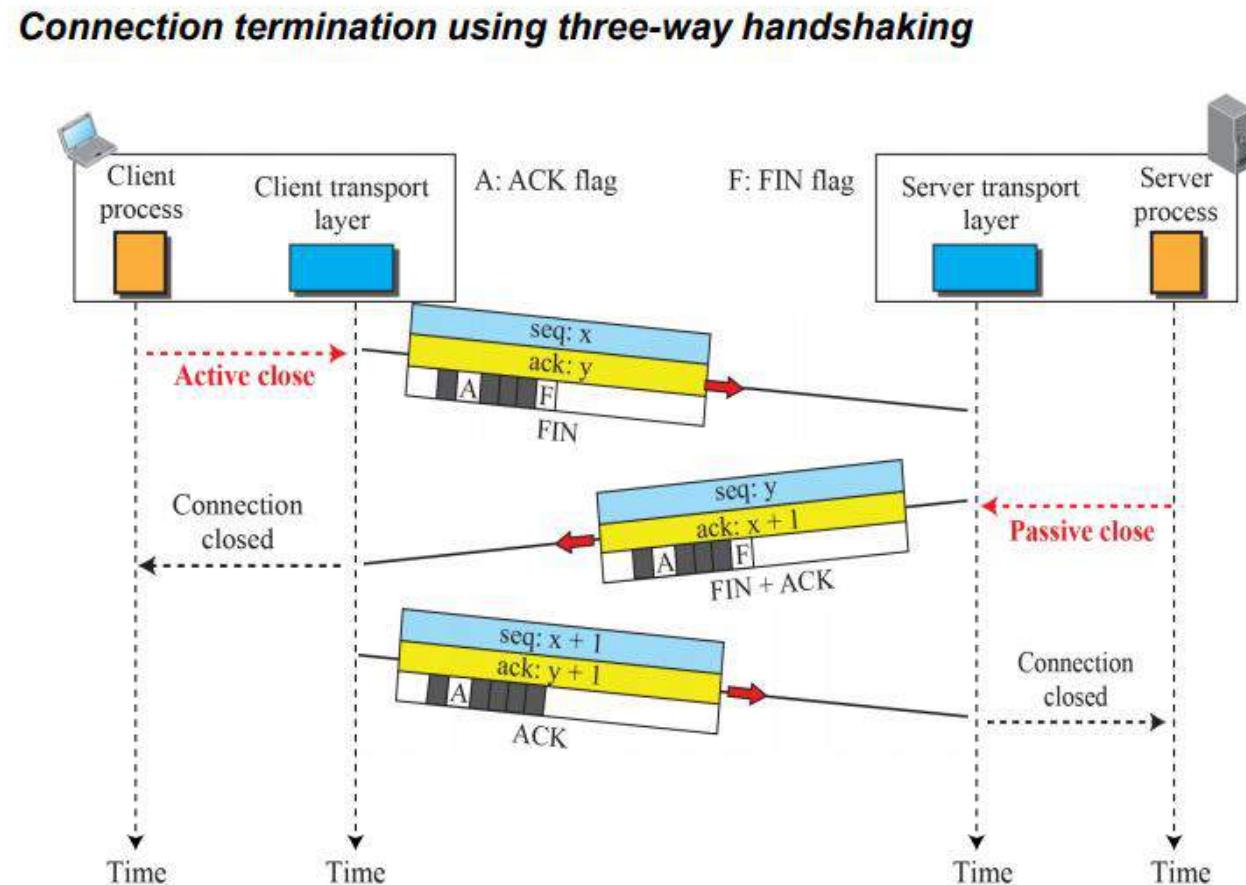
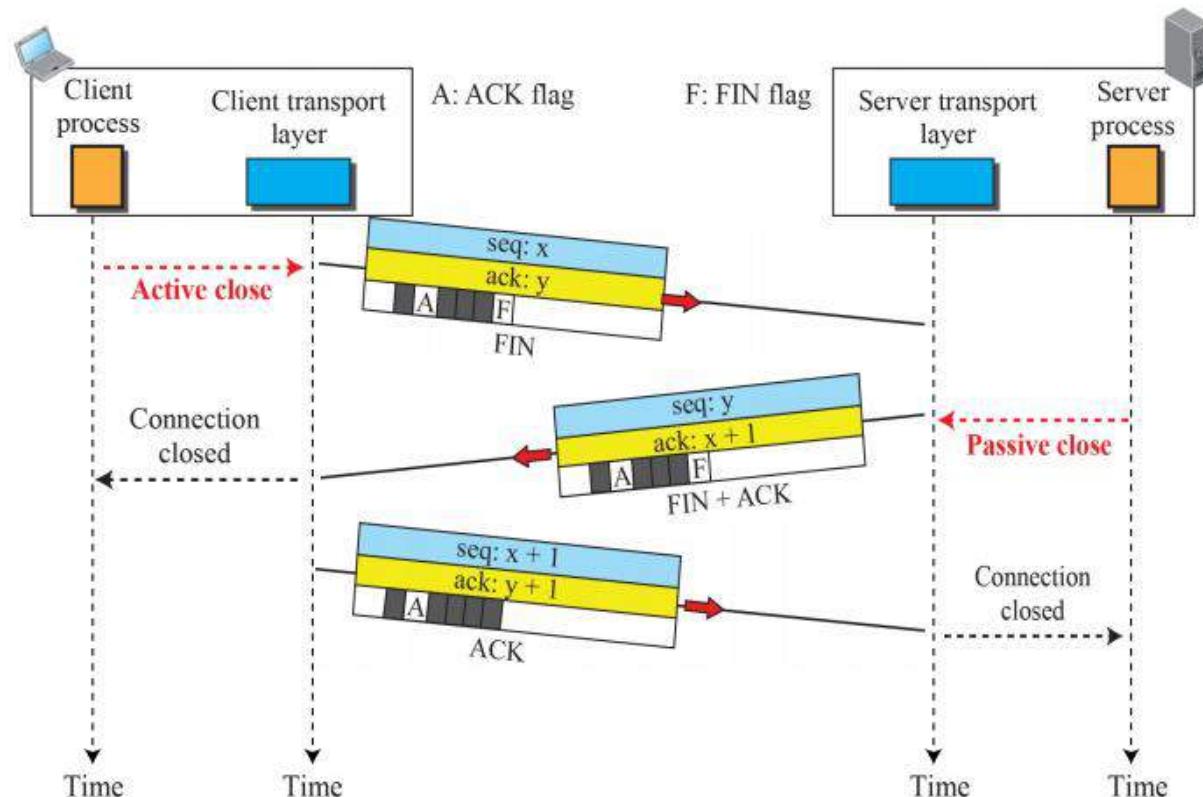


Figure 23.18 Connection termination using three-way handshaking

- The client TCP sends the last segment, an ACK segment, to confirm the receipt of the FIN segment from the TCP server.
- The segment contains the acknowledgment number.
- This segment cannot carry data and consumes **no sequence numbers**.

An ACK segment, if carrying no data, consumes no sequence number.

Connection termination using three-way handshaking



---

## Connection Reset

---

- TCP at one end may deny a connection request, may abort an existing connection, or may terminate an idle connection.
- All of these are done with the **RST flag**.

## Some well-known ports used with UDP and TCP

<i>Port</i>	<i>Protocol</i>	<i>UDP</i>	<i>TCP</i>	<i>Description</i>
7	Echo	✓		Echoes back a received datagram
9	Discard	✓		Discards any datagram that is received
11	Users	✓	✓	Active users
13	Daytime	✓	✓	Returns the date and the time
17	Quote	✓	✓	Returns a quote of the day
19	Chargen	✓	✓	Returns a string of characters
20, 21	FTP		✓	File Transfer Protocol
23	TELNET		✓	Terminal Network
25	SMTP		✓	Simple Mail Transfer Protocol
53	DNS	✓	✓	Domain Name Service
67	DHCP	✓	✓	Dynamic Host Configuration Protocol
69	TFTP	✓		Trivial File Transfer Protocol
80	HTTP		✓	Hypertext Transfer Protocol
111	RPC	✓	✓	Remote Procedure Call
123	NTP	✓	✓	Network Time Protocol
161, 162	SNMP		✓	Simple Network Management Protocol

# Application Layer

slides are modified from J. Kurose & K. Ross

# Chapter Application layer

- Principles of network applications
- Web and HTTP
- FTP
- Electronic Mail
  - ❖ SMTP, POP3, IMAP
- DNS
- TELNET
- SSH

# Processes communicating

**Process:**

program running within a host

**Client process:**

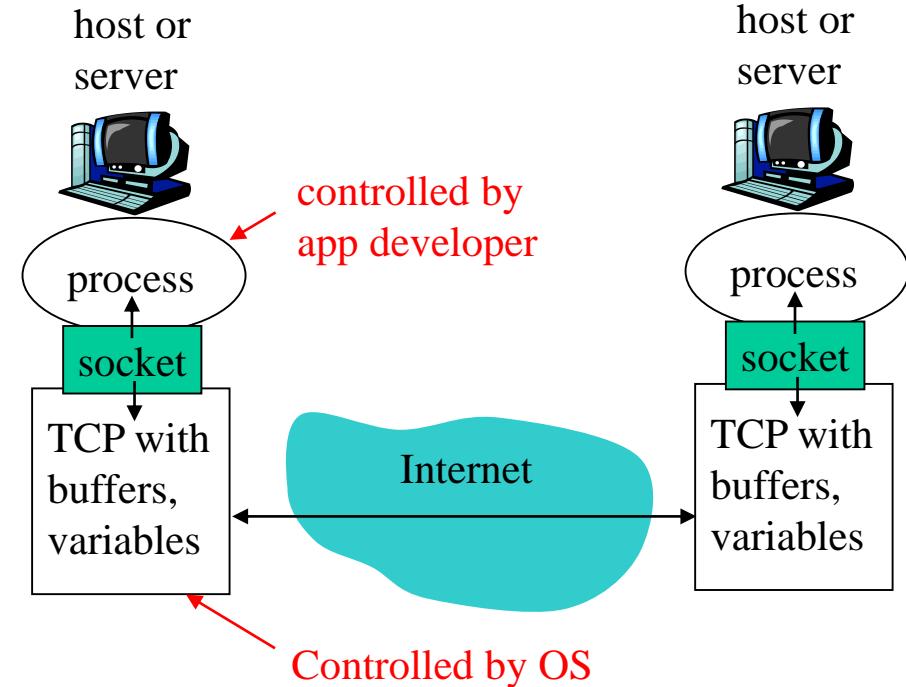
initiates communication

**Server process:**

waits to be contacted

process sends/receives messages to/from its **socket**

**identifier** includes both **IP address** and **port numbers** associated with process on host.



# App-layer protocol defines

- Types of messages exchanged,
  - ❖ e.g., request, response
- Message syntax:
  - ❖ what fields in messages & how fields are delineated(defined precisely)
- Message semantics
  - ❖ meaning of information in fields
- Rules for when and how processes send & respond to messages

## Public-domain protocols:

- ◆ defined in RFCs
- ◆ allows for interoperability
- ◆ e.g., HTTP, SMTP

(RFC: Request for Comments; Standard developed by Internet Engineering Task Force & Internet Society Publication for protocol)

## Proprietary protocols:

- ◆ e.g., Skype

# Internet transport protocols services

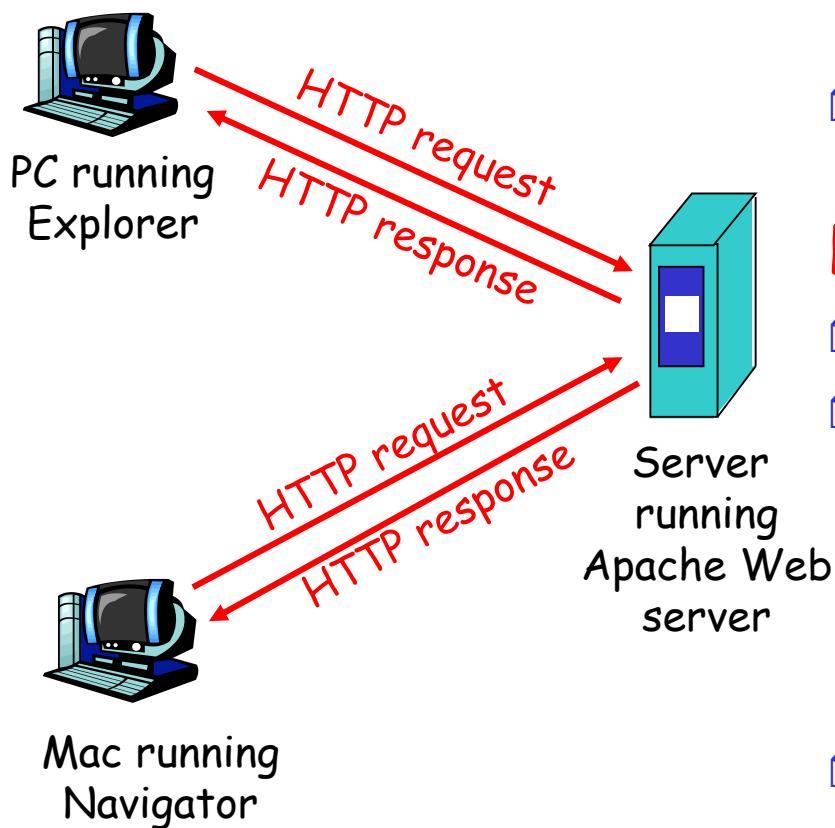
## TCP service:

- connection-oriented*: setup required between client and server processes
- reliable transport* between sending and receiving process
- flow control*: sender won't overwhelm receiver
- congestion control*: throttle sender when network overloaded
- does not provide*: timing, minimum throughput guarantees, security

## UDP service:

- unreliable data transfer between sending and receiving process
- does not provide: connection setup, reliability, flow control, congestion control, timing, throughput guarantee, or security

# HTTP overview



- Web page consists of **base HTML-file** which includes several referenced **objects**
- Each object is addressable by a **URL**

## **HTTP: hypertext transfer protocol**

- Web's application layer protocol
- client/server model
  - ❖ **client:** browser that requests, receives, "displays" Web objects
  - ❖ **server:** Web server sends objects in response to requests
- uses TCP
- **is "stateless"**

Stateless: no record of previous interactions and each interaction request has to be handled based entirely on information that comes with it.

# HTTP connections

## Nonpersistent HTTP

- At most one object is sent over a TCP connection.

## Persistent HTTP

- Multiple objects can be sent over single TCP connection between client and server.

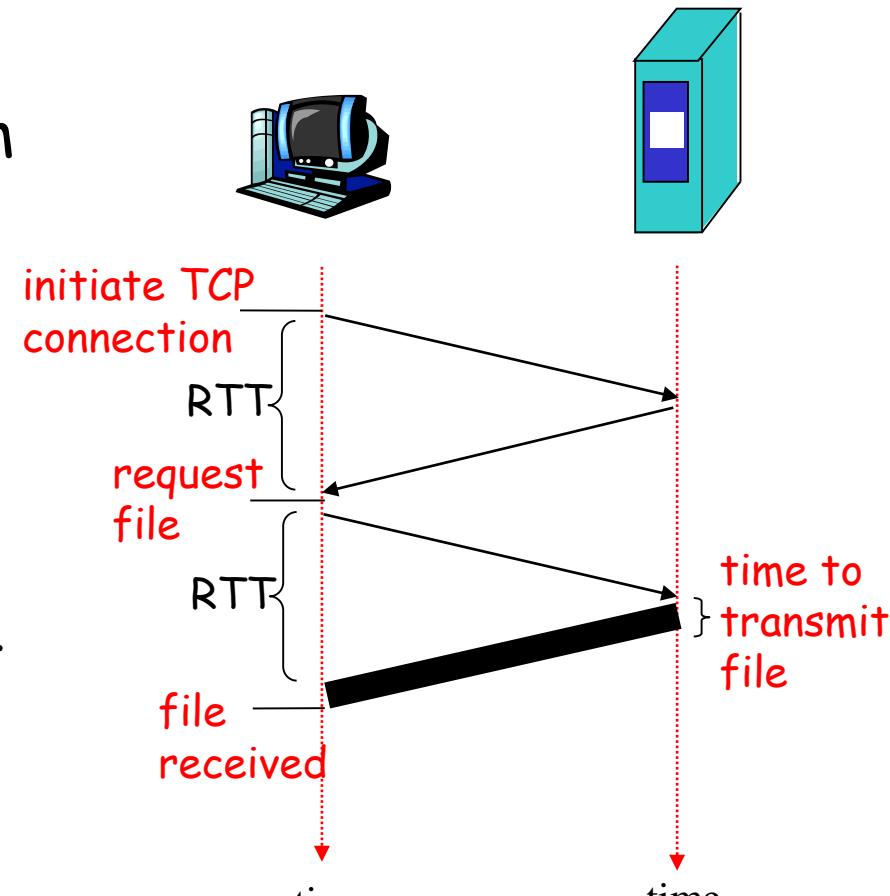
# Non-Persistent HTTP: Response time

**Definition of RTT:** time for a small packet to travel from client to server and back.

## Response time:

- one RTT to initiate TCP connection
- one RTT for HTTP request and first few bytes of HTTP response to return
- file transmission time

$$\text{total} = 2\text{RTT} + \text{transmit time}$$



RTT= Round trip delay Time

# Persistent HTTP

## Nonpersistent HTTP issues:

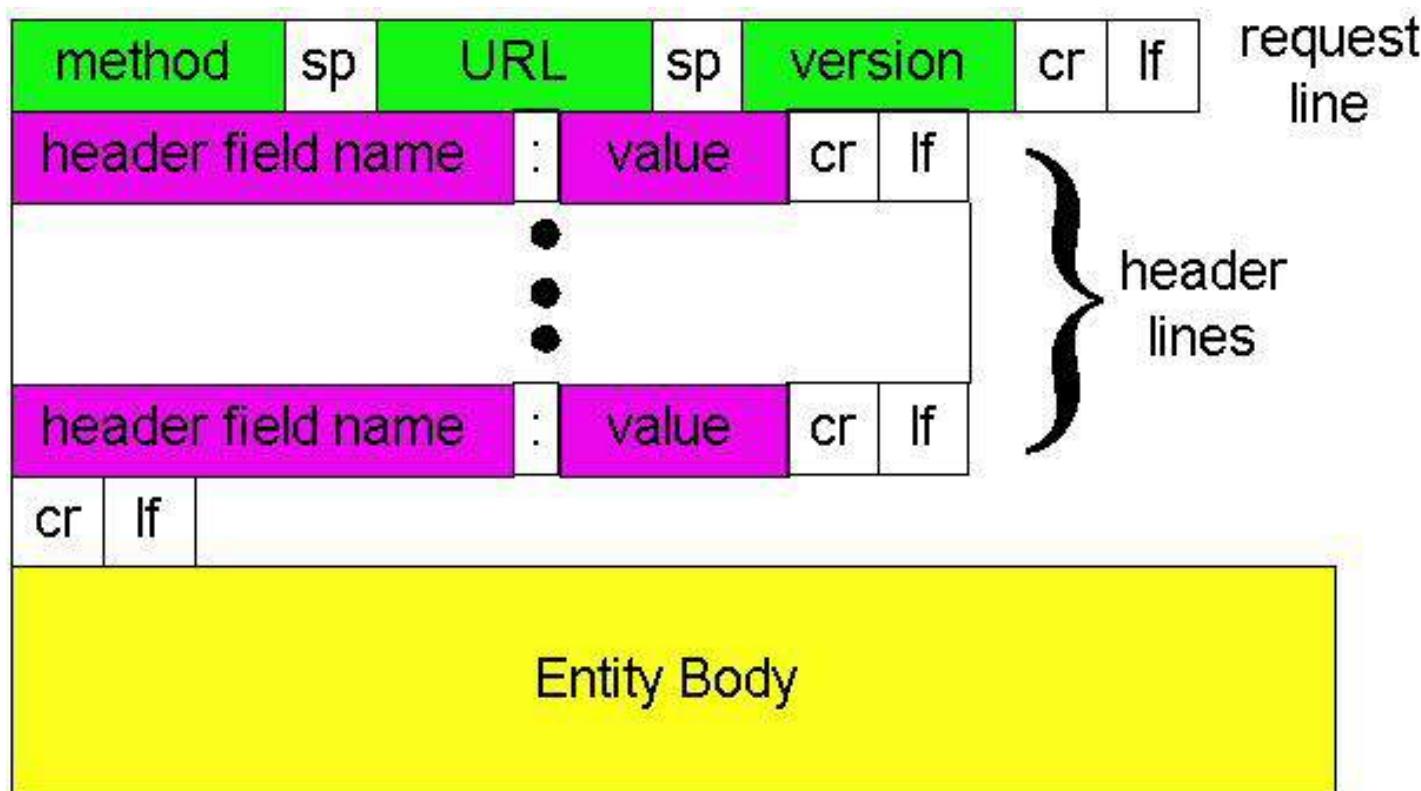
- requires 2 RTTs per object
- OS overhead for each TCP connection
- browsers often open parallel TCP connections to fetch referenced objects

## Persistent HTTP

- server leaves connection open after sending response
- subsequent HTTP messages between same client/server sent over open connection
- client sends requests as soon as it encounters a referenced object
- as little as one RTT for all the referenced objects

# HTTP messages

- two types of HTTP messages: *request, response*
- **HTTP request message:**
  - ❖ ASCII (human-readable format)



# Method types

## HTTP/1.0

- GET
  - ❖ request an object from server
- POST
  - ❖ upload information using forms
- HEAD
  - ❖ asks server to leave requested object out of response

## HTTP/1.1

- GET, POST, HEAD
- PUT
  - ❖ uploads file in entity body to path specified in URL field
- DELETE
  - ❖ deletes file specified in the URL field

# Cookies: Keeping state

## What cookies can bring:

- authorization
- shopping carts
- recommendations
- user session state (Web e-mail)

Cookies and privacy: aside

- cookies permit sites to learn a lot about you
- you may supply name and e-mail to sites

## How to keep "state":

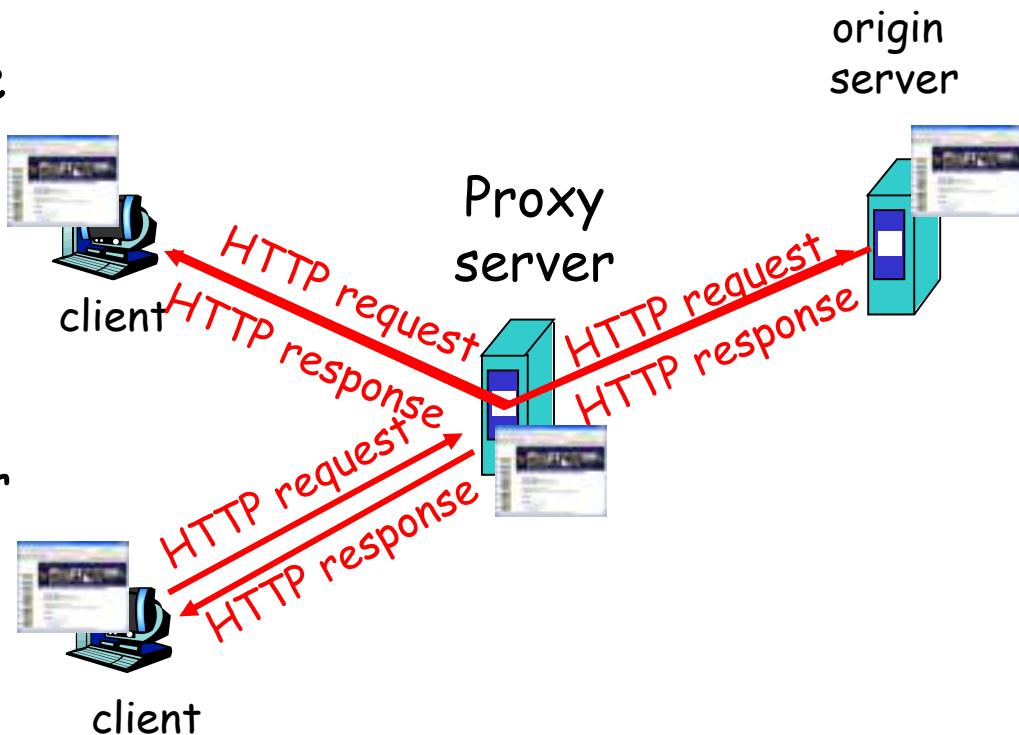
- protocol endpoints: maintain state at sender/receiver over multiple transactions
- cookies: http messages carry state

# Web caches (proxy server)

**Goal:** satisfy client request without involving origin server

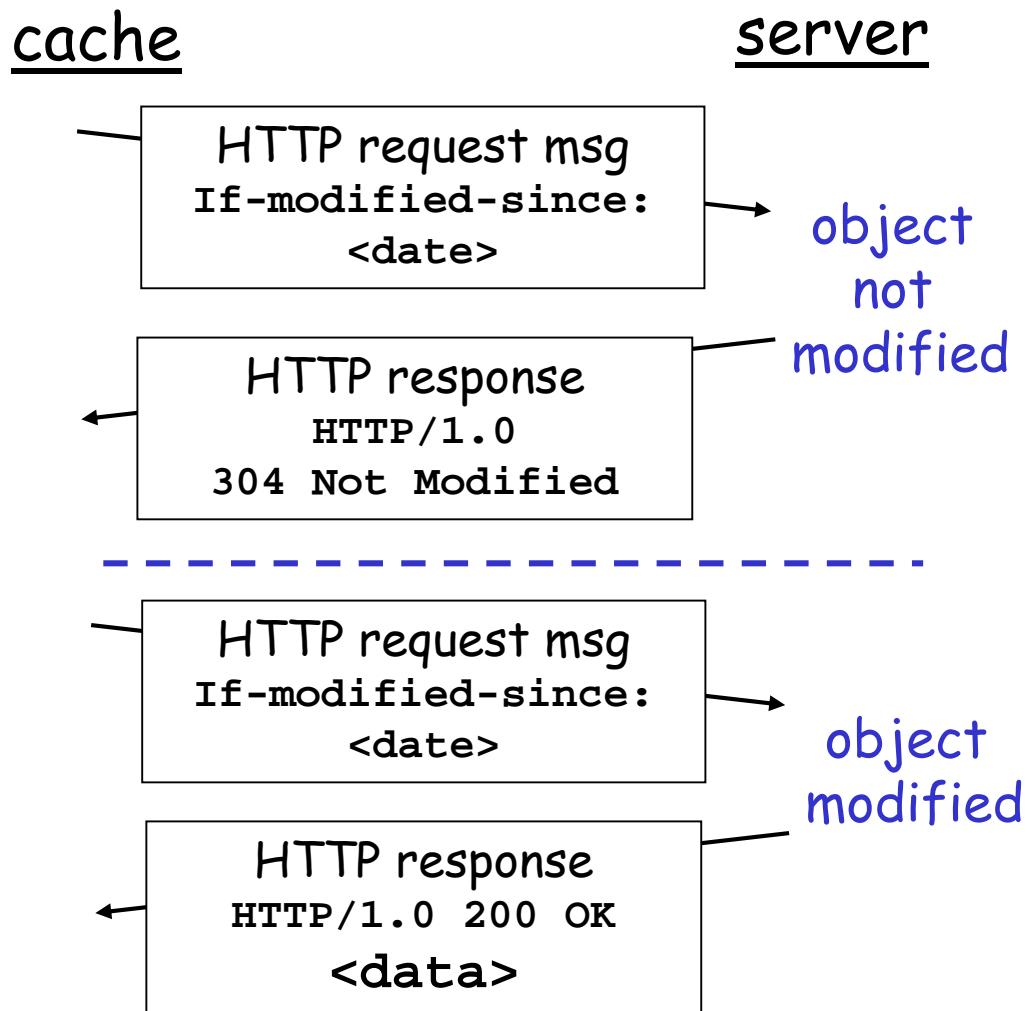
- user sets browser:  
Web accesses via cache
- browser sends all HTTP  
requests to cache

- Why Web caching?
  - ❖ reduce response time for  
client request
  - ❖ reduce traffic on an  
institution's access link.
  - ❖ enables "poor" content  
providers to effectively  
deliver content

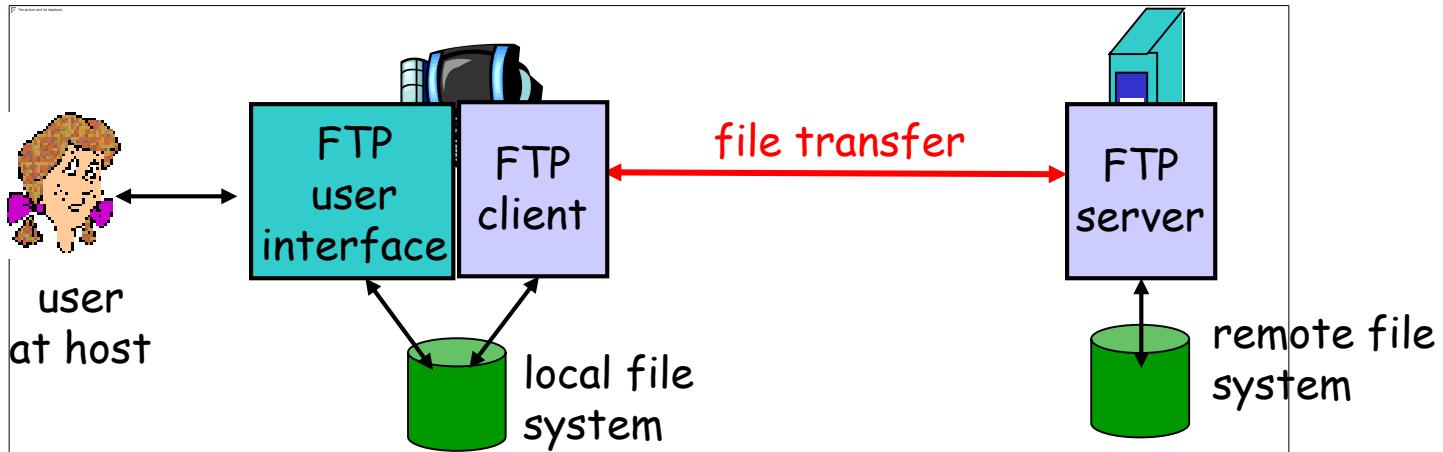


# Conditional GET

- **Goal:** don't send object if cache has up-to-date cached version
- **cache:** specify date of cached copy in HTTP request  
**If-modified-since: <date>**
- **server:** response contains no object if cached copy is up-to-date:  
**HTTP/1.0 304 Not Modified**



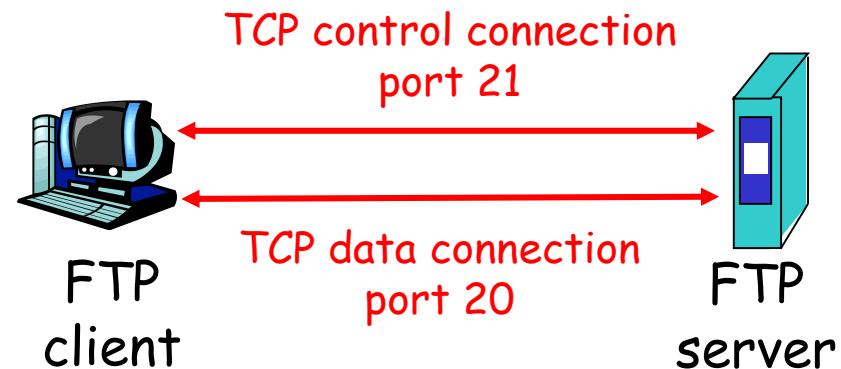
# FTP: the file transfer protocol



- transfer file to/from remote host
- client/server model
  - ❖ *client*: side that initiates transfer (either to/from remote)
  - ❖ *server*: remote host
- ftp: RFC 959
- ftp server: port 21

# FTP: separate control, data connections

- FTP client contacts FTP server at port 21
- client authorized over control connection
- client browses remote directory by sending commands over control connection.
- when server receives file transfer command, server opens 2<sup>nd</sup> TCP connection (for file) to client
- after transferring one file, server closes data connection.
- server opens another TCP data connection to transfer another file.
- control connection: "**out of band**"
- FTP server maintains "state": current directory, earlier authentication



# FTP commands, responses

## Sample commands:

- ❑ sent as ASCII text over control channel
- ❑ **USER *username***
- ❑ **PASS *password***
- ❑ **LIST** return list of file in current directory
- ❑ **RETR *filename*** retrieves (gets) file
- ❑ **STOR *filename*** stores (puts) file onto remote host

## Sample return codes

- ❑ status code and phrase (as in HTTP)
- ❑ **331 Username OK, password required**
- ❑ **125 data connection already open; transfer starting**
- ❑ **425 Can't open data connection**
- ❑ **452 Error writing file**

# FTP issues

- Multiple connections are used
  - ❖ for each directory listing and file transmission
- No integrity check at receiver
- Messages are sent in clear text
  - ❖ including **Passwords** and file contents
  - ❖ can be sniffed by eavesdroppers
- Solution
  - ❖ Secure FTP (SSH FTP)
    - allows a range of operations on remote files
  - ❖ FTPS ( FTP over Secure Sockets Layer (SSL) )
  - ❖ Transport Layer Security (TLS) encryption

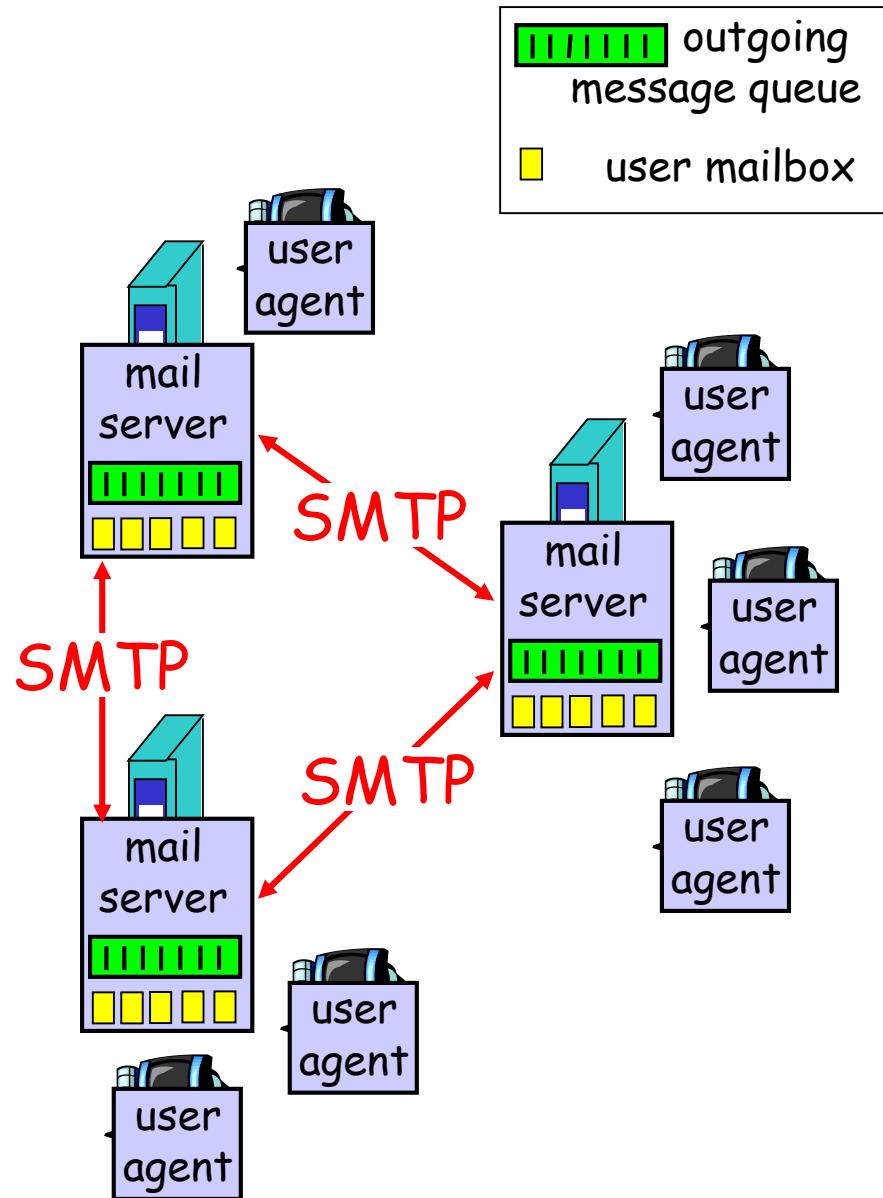
# Electronic Mail

Three major components:

- user agents
- mail servers
- simple mail transfer protocol: SMTP

## User Agent(UA)

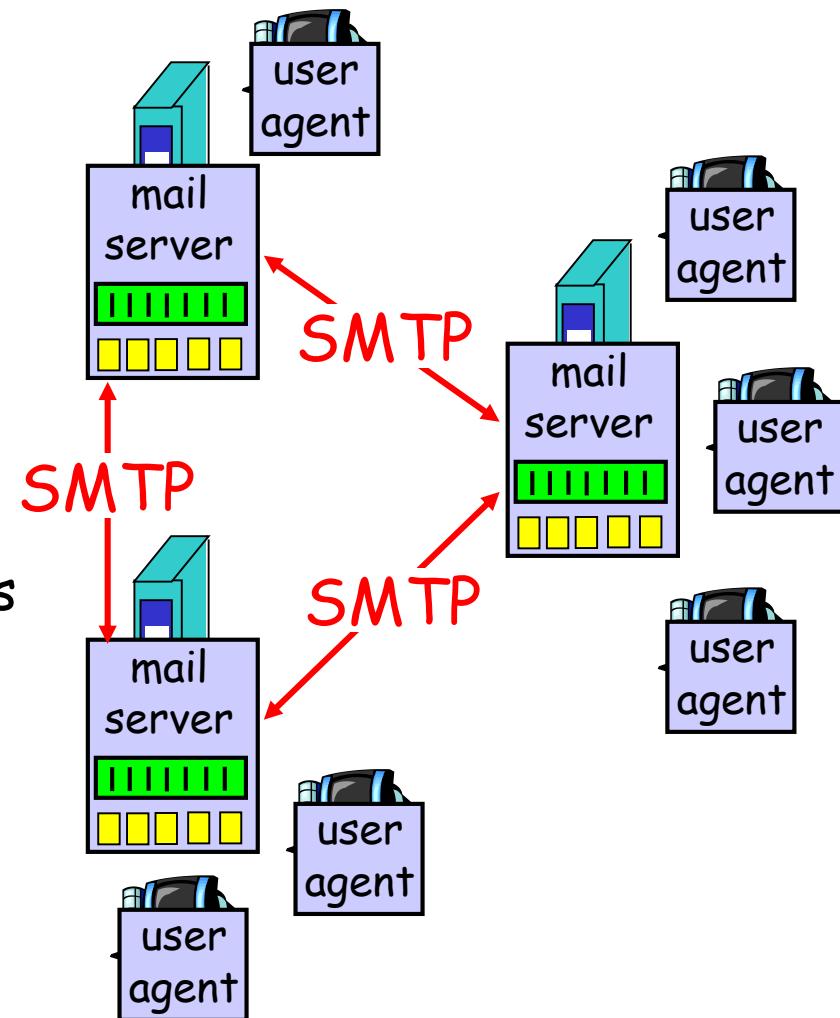
- a.k.a. "mail reader"
- composing, editing, reading mail messages
- e.g., Eudora, Outlook, elm, Mozilla Thunderbird
- outgoing, incoming messages stored on server



# Electronic Mail: mail servers

## Mail Servers

- **mailbox** contains incoming messages for user
- **message queue** of outgoing (to be sent) mail messages
- **SMTP protocol** between mail servers to send email messages
  - ❖ client: sending mail server
  - ❖ "server": receiving mail server

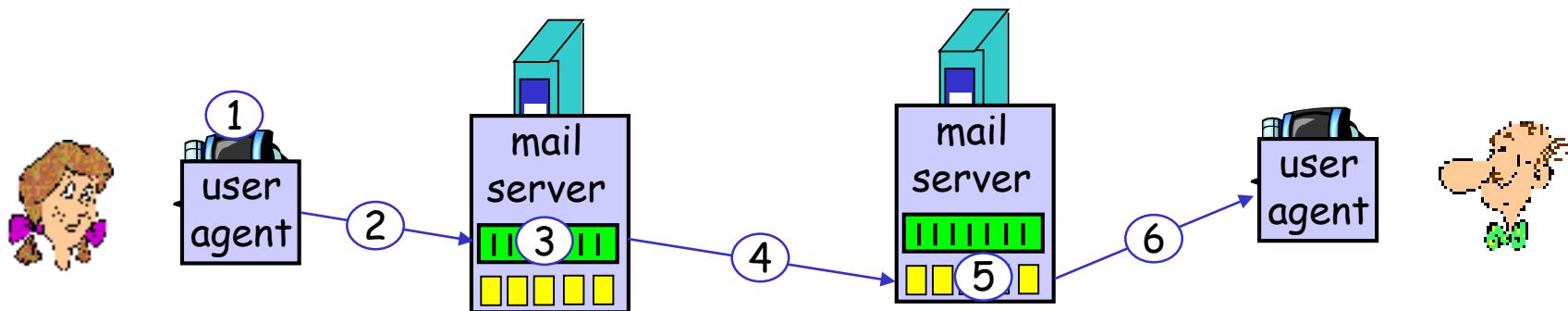


# Electronic Mail: SMTP [RFC 2821]

- uses TCP to reliably transfer email message from client to server (port 25)
- direct transfer: sending server to receiving server
- three phases of transfer
  - ❖ handshaking (greeting)
  - ❖ transfer of messages
  - ❖ closure
- command/response interaction
  - ❖ commands: ASCII text
  - ❖ response: status code and phrase
- messages must be in 7-bit ASCII

# Scenario: Alice sends message to Bob

- 1) Alice uses UA to compose message and "to" bob@someschool.edu
- 2) Alice's UA sends message to her mail server; message placed in message queue
- 3) Client side of SMTP opens TCP connection with Bob's mail server
- 4) SMTP client sends Alice's message over the TCP connection
- 5) Bob's mail server places the message in Bob's mailbox
- 6) Bob invokes his user agent to read message



# SMTP: final words

- SMTP uses persistent connections
- SMTP requires message (header & body) to be in 7-bit ASCII
- SMTP server uses CRLF . CRLF to determine end of message

## Comparison with HTTP:

- HTTP: pull( initial request for data originates from the client, and then is responded to by the server).
- SMTP: push(the request for a given transaction is initiated by the central server).
- both have ASCII command/response interaction, status codes
- HTTP: each object encapsulated in its own response msg
- SMTP: multiple objects sent in multipart msg

# Mail message format

SMTP: protocol for exchanging email msgs

RFC 822: standard for text message format:

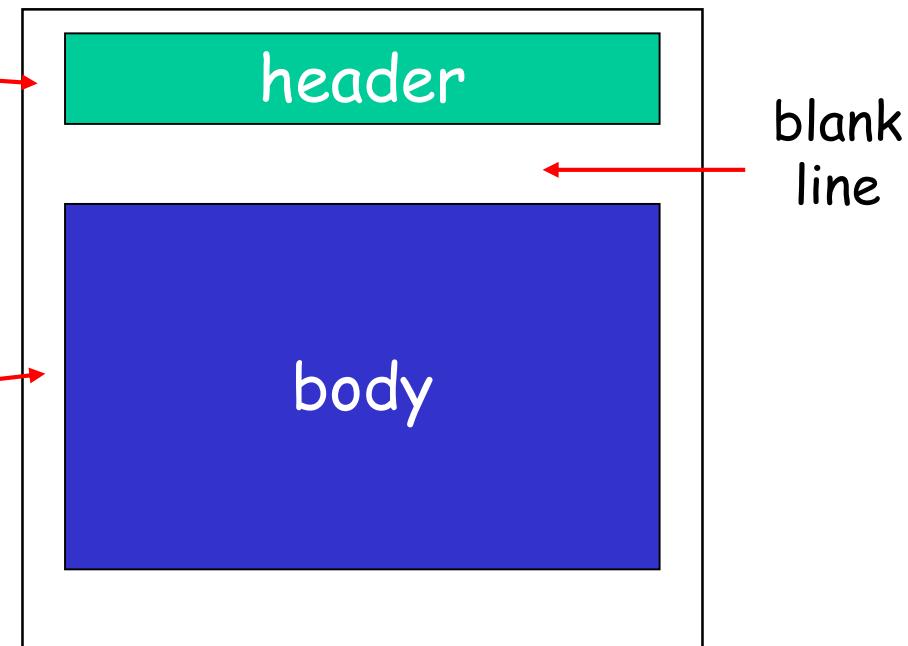
- header lines, e.g.,

- ❖ To:
- ❖ From:
- ❖ Subject:

*different from SMTP commands!*

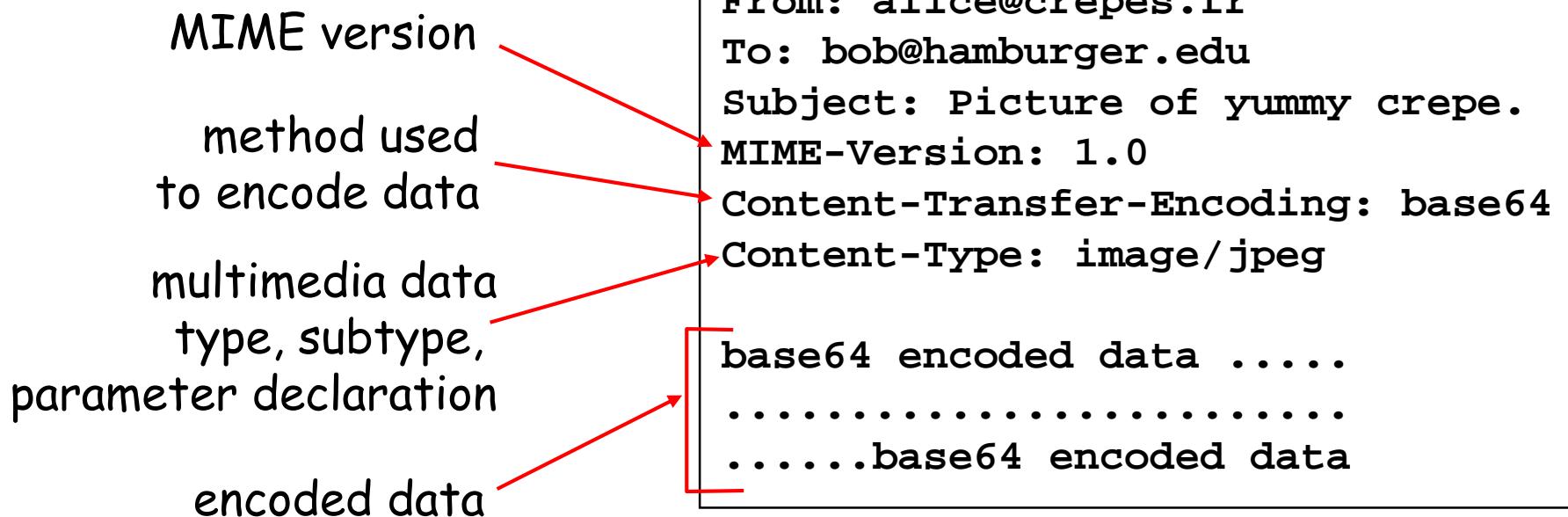
- body

- ❖ the "message",  
*ASCII characters only*

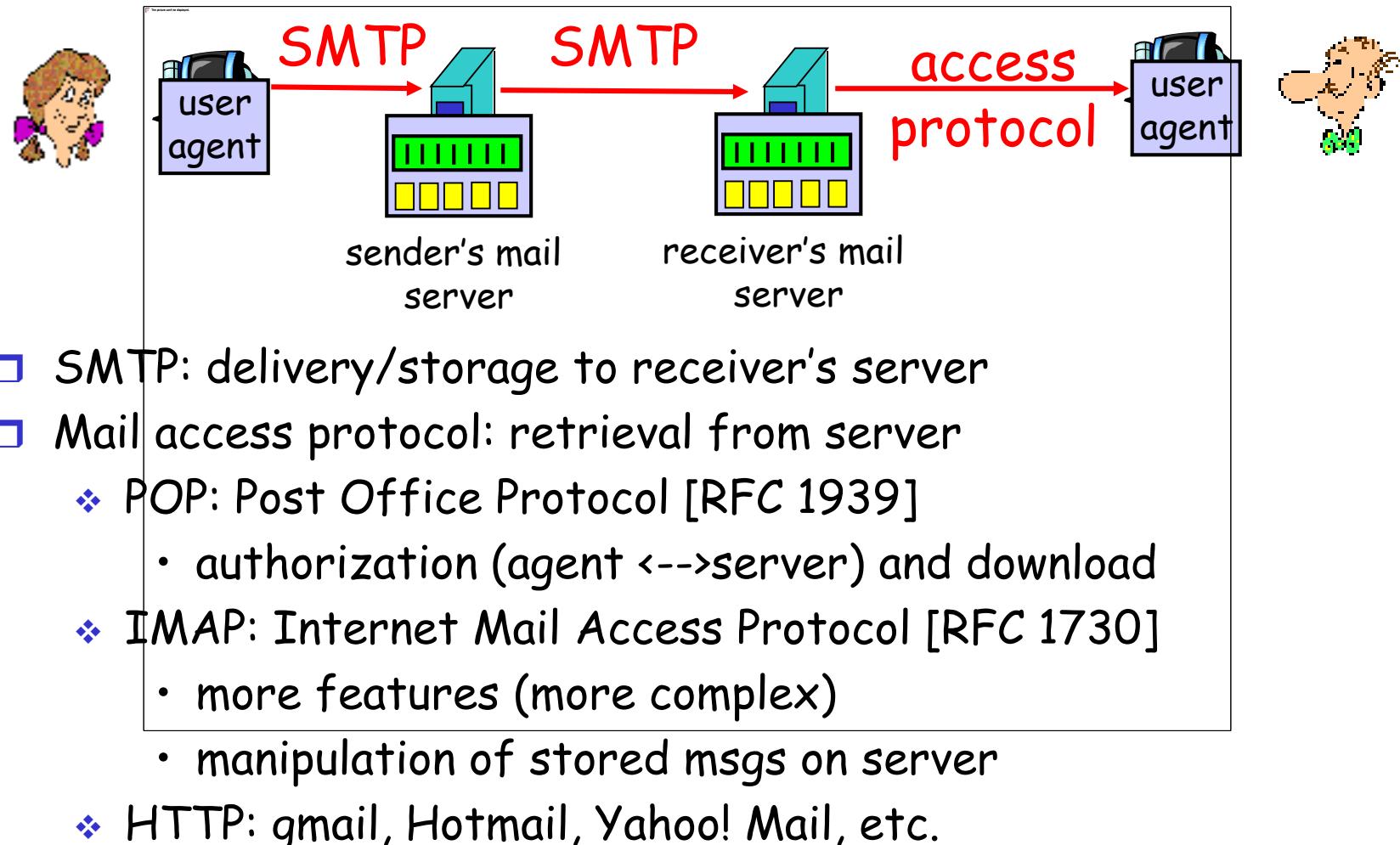


# Message format: multimedia extensions

- ❑ MIME: multimedia mail extension, RFC 2045, 2056
- ❑ additional lines in msg header declare MIME content type



# Mail access protocols



# DNS: Domain Name System

People: many identifiers:

- ❖ SSN, name, passport #

Internet hosts, routers:

- ❖ IP address (32 bit) - used for addressing datagrams
- ❖ "name", e.g., www.yahoo.com - used by humans

Domain Name System:

- *distributed database* implemented in hierarchy of many *name servers*
- *application-layer protocol* host, routers, name servers to communicate to *resolve* names (address/name translation)
  - ❖ note: core Internet function, implemented as application-layer protocol
  - ❖ complexity at network's "edge"

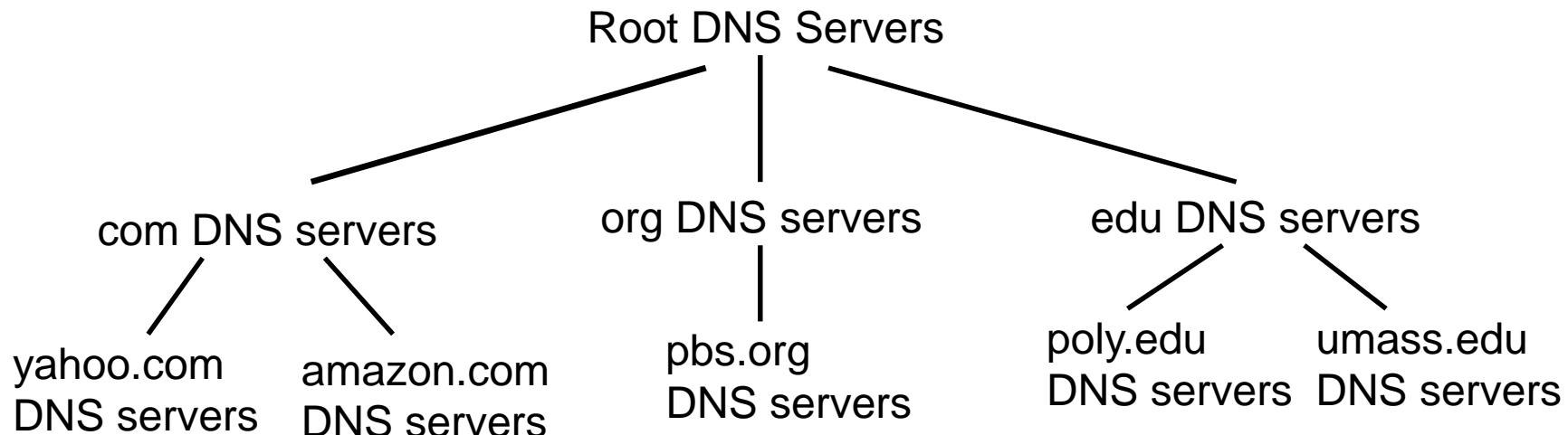
# DNS services

- hostname to IP address translation
- host aliasing
- mail server aliasing
- load distribution
  - ❖ replicated Web servers: set of IP addresses for one canonical name

## Why not centralize DNS?

- single point of failure
- traffic volume
- distant centralized database doesn't scale!
- maintenance

# Distributed, Hierarchical Database



Client wants IP for www.amazon.com; 1<sup>st</sup> approx:

- client queries a **root server** to find **com DNS server**
- client queries **com DNS server** to get **amazon.com DNS server**
- client queries **amazon.com DNS server** to get **IP address** for **www.amazon.com**

# TELNET protocol

- TELNET (TELecommunication NETwork) was developed in 1969 beginning with RFC 15 and standardized as IETF STD 8, one of the first Internet standards.
- TELNET clients have been available on most Unix systems for many years and are available for virtually all platforms. Most network equipment and OSs with a TCP/IP stack support some kind of TELNET service server for their remote configuration (including ones based on MS Windows NT and later).
- Because of security issues with TELNET, its use has waned as it is replaced by the use of SSH for remote access.

# TELNET protocol

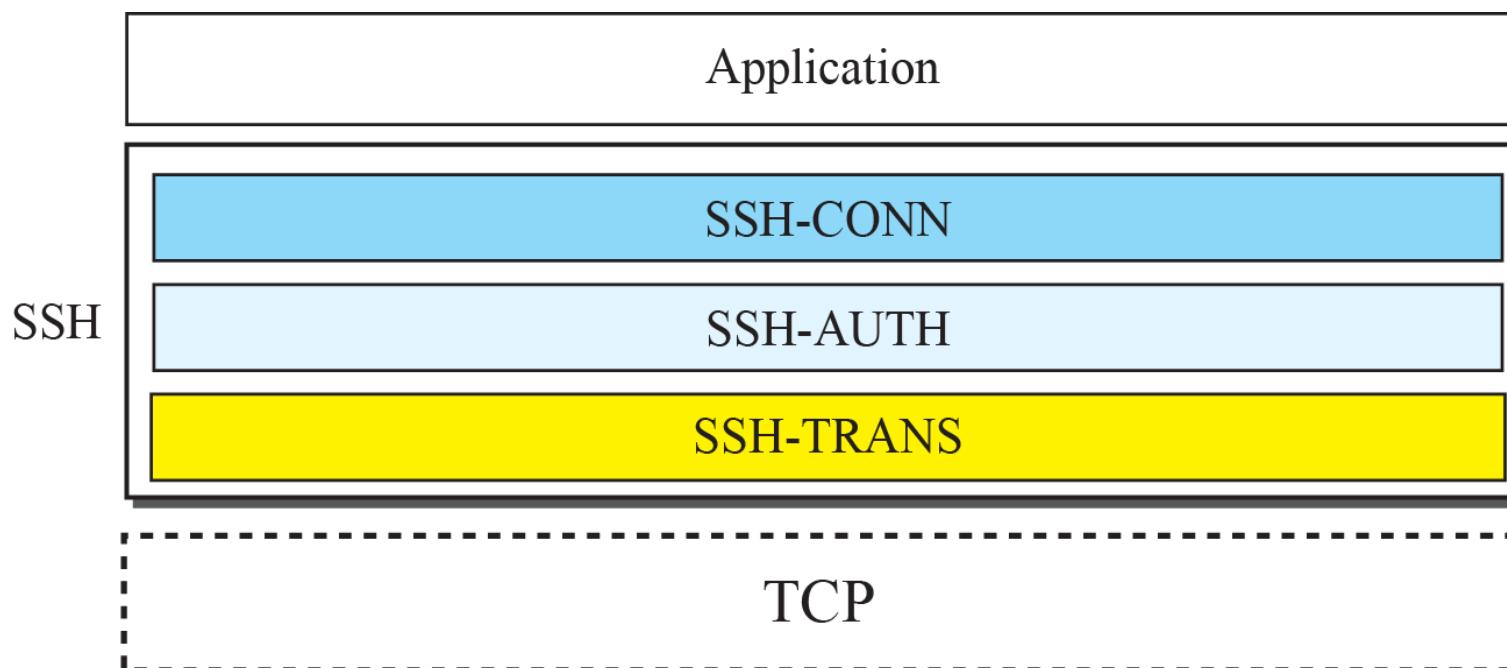
- Most often, a user will be telneting to a Unix-like server system or a simple network device such as a switch. Once the connection is established, he would then log in with his account information and execute operating system commands remotely on that computer, such as ls or cd etc.
- For testing and debugging purposes: On many systems, the client may also be used to make interactive **raw-TCP sessions**, even when that option is not available. The sessions are equivalent to raw TCP as long as byte 255 never appears in the data.
- TELNET works on the well known TCP port 23.

## **26-5 SECURE SHELL (SSH)**

Although Secure Shell (SSH) is a secure application program that can be used today for several purposes such as remote logging and file transfer, it was originally designed to replace TELNET. There are two versions of SSH. The first version, SSH-1, is now deprecated because of security flaws in it. In this section, we discuss only SSH-2.

## 26.5.1 Components

SSH is an application-layer protocol with three components, as shown in Figure.



## 26.5.2 Applications

SSH is a general-purpose protocol that provides a secure connection between a client and server.

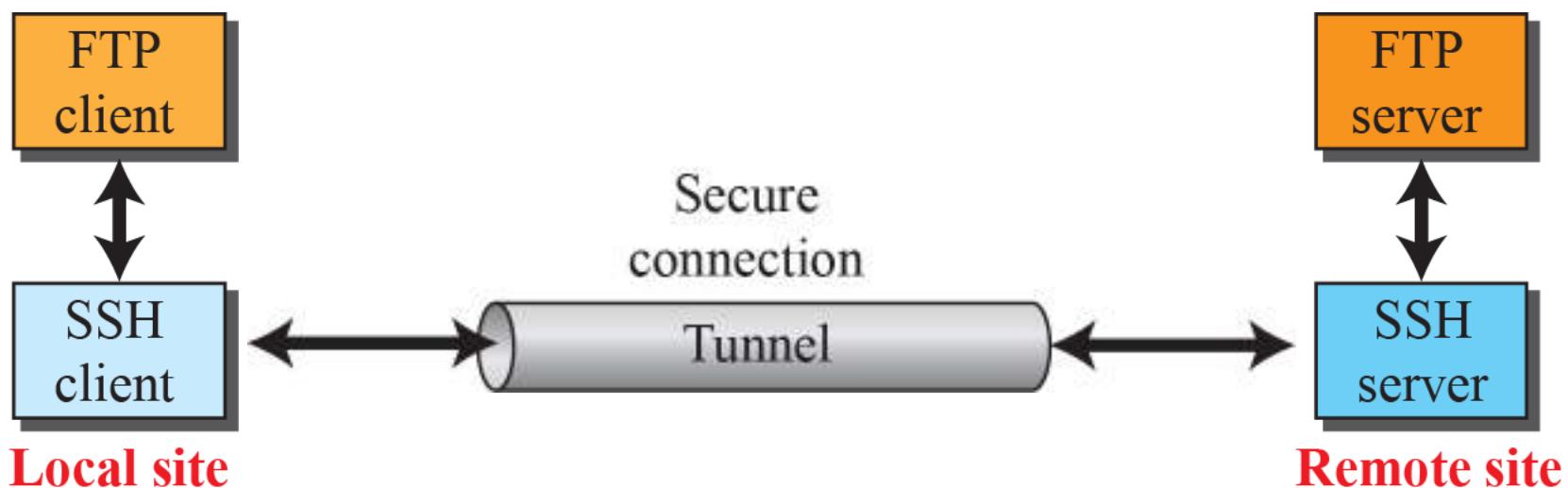
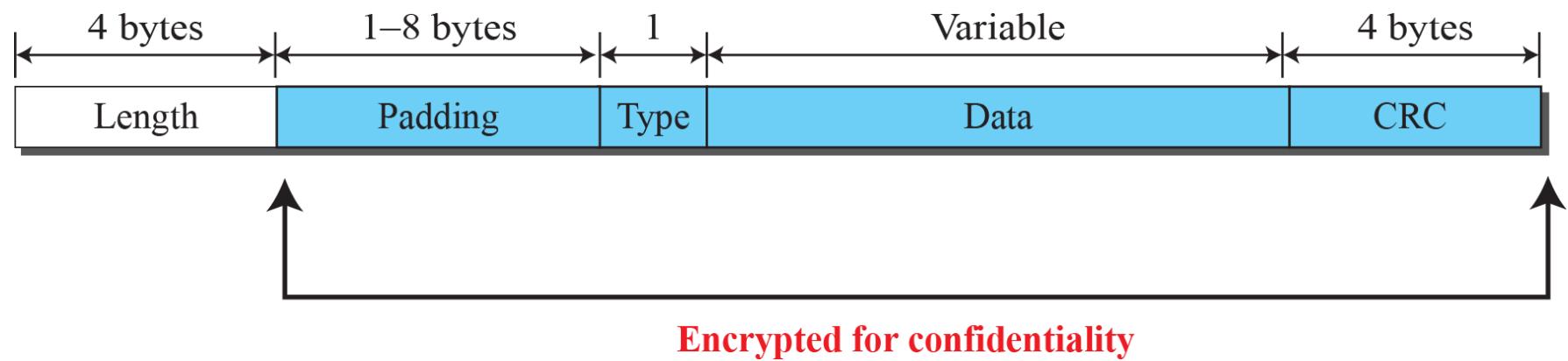


Figure 26.27: SSH Packet Format



<b>Port Number</b>	<b>Protocol</b>	<b>Application</b>
20	TCP	FTP data
21	TCP	FTP control
22	TCP	SSH
23	TCP	Telnet
25	TCP	SMTP
53	UDP, TCP	DNS
67, 68	UDP	DHCP
69	UDP	TFTP
80	TCP	HTTP (WWW)
110	TCP	POP3
161	UDP	SNMP
443	TCP	HTTPS (SSL)
16,384–32,767	UDP	RTP-based voice (VoIP) and video

# Summary

- Application
- Web and HTTP
- File Transfer Protocol
- Electronic Mail
  - ❖ SMTP
  - ❖ POP3
  - ❖ IMAP
- Domain Name Service
- TELNET
- SSH