



Vande Sadgurum
Chandrasekharam



PATHWAY TO
SUCCESS



श्रीचन्द्रशेखरेन्द्रसरस्वतीविश्वमहाविद्यालयः

SRI CHANDRASEKHARENDRASARASWATHI VISWA MAHAVIDYALAYA

Deemed to be University u/s 3 of UGC Act 1956 | Accredited with "A" grade by NAAC

Enathur, Kanchipuram - 631 561. Tamilnadu, India | www.kanchiuniv.ac.in

Sponsored and run by Sri Kanchi Kamakoti Peetam Charitable Trust



STUDY MATERIAL
of
COMPUTER NETWORKS

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Prepared by: Dr.S.Selvakumar
Assistant Professor



Sri Chandrasekharendra Saraswathi Viswa Mahavidyalaya
Department of Electronics and Communication Engineering
Syllabus for Full Time BE, Regulations 2018
(Applicable for students admitted from 2018-19 onwards)

PCC22

COMPUTER NETWORKS

VI SEMESTER

PRE-REQUISITE:

Basic knowledge of Digital System Design, Signals & Systems and Digital Communication

L	T	P	C
3	0	0	3

OBJECTIVES:

The student should be made to -

- Understand the concepts of network architecture and transmission medium
- Perform and understand methods for error detection and correction of data
- Be exposed to various addressing schemes and routing protocols.
- Learn the flow control and congestion control algorithms
- Be familiar with real time applications of networks

UNIT I FUNDAMENTALS & SIGNAL TRANSMISSION (10 Hrs)

Fundamentals : Building a network – Requirements - Layering and protocols - OSI Model - Internet Architecture – Performance - Network Topology ; Physical Layer: Data and Signals - Digital Transmission - Analog Transmission - Multiplexing and Spread Spectrum - Transmission Media.

UNIT II MEDIA ACCESS & LOGICAL LINK CONTROL (9 Hrs)

Framing - Error Detection and Correction - Media access control - Ethernet (802.3) - Wireless LANs – 802.11 – Bluetooth - Switching and bridging - Flow control.

UNIT III ROUTING & ADDRESSING SCHEMES (9 Hrs)

Basic Internetworking (IP, CIDR, ARP, DHCP, ICMP) - Routing (RIP, OSPF, metrics) – Switch basics – Global Internet (Areas, BGP, IPv6), Multicast – addresses – multicast routing (DVMRP, PIM)

UNIT IV END TO END COMMUNICATION (9 Hrs)

Overview of Transport layer - UDP - Reliable byte stream (TCP) - Connection management – Flow control - Retransmission – Queueing Disciplines - TCP Congestion control - Congestion avoidance (DECbit, RED)

UNIT V APPLICATION LAYER PROTOCOLS (8 Hrs)

Electronic Mail (SMTP, POP3, IMAP, MIME) – HTTP – Web Services – DNS - SNMP - Multimedia applications

OUTCOMES:

Total: 45 Hrs

At the end of the course, the student should be able to

1. Choose the required functionality at each layer for given application
2. Detect and Correct the error in the frame
3. Apply the knowledge of addressing scheme and various routing protocols in data communication to select optimal path.
4. Trace the flow of information from one node to another node in the network
5. Develop real time applications of networks

TEXT BOOKS:

1. Larry L. Peterson, Bruce S. Davie, “Computer Networks: A Systems Approach”, Fifth Edition, Morgan Kaufmann Publishers, 2011.
2. Behrouz A. Forouzan, “Data Communications and Networking”, Fourth Edition, McGrawHill, 2011.

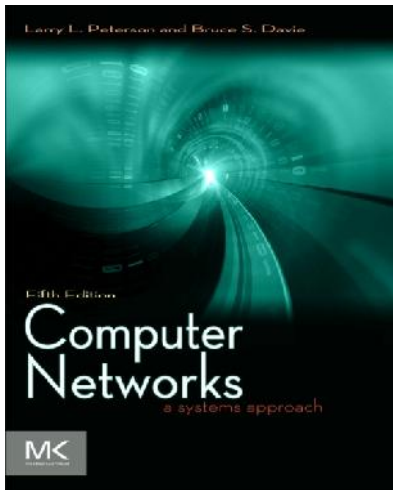
REFERENCES:

1. James F. Kurose, Keith W. Ross, “Computer Networking - A Top-Down Approach Featuring the Internet”, Fifth Edition, Pearson Education, 2009.
 2. Nader. F. Mir, “Computer and Communication Networks”, Pearson/Prentice Hall Publishers, 2010.
 3. Ying-Dar Lin, Ren-Hung Hwang, Fred Baker, “Computer Networks – An Open Source Approach, First Edition, McGraw Hill, 2011.
-

COMPUTER NETWORKS

UNIT I - FUNDAMENTALS & SIGNAL TRANSMISSION

Fundamentals : Building a network – Requirements - Layering and protocols - OSI Model – Internet Architecture – Performance - Network Topology ; Physical Layer: Data and Signals – Digital Transmission - Analog Transmission - Multiplexing and Spread Spectrum - Transmission Media.



Unit 1:

Fundamental : Building a Network

- **Requirements**
- **Layering and Protocols**
- **OSI models**
- **Internet Architecture**
- **Performance**

Chapter Goal

- Exploring the requirements that different applications and different communities place on the computer network
- Introducing the idea of network architecture
- Introducing some key elements in implementing Network Software
- Define key metrics that will be used to evaluate the performance of computer network

Topic 1

Fundamental : Building a Network

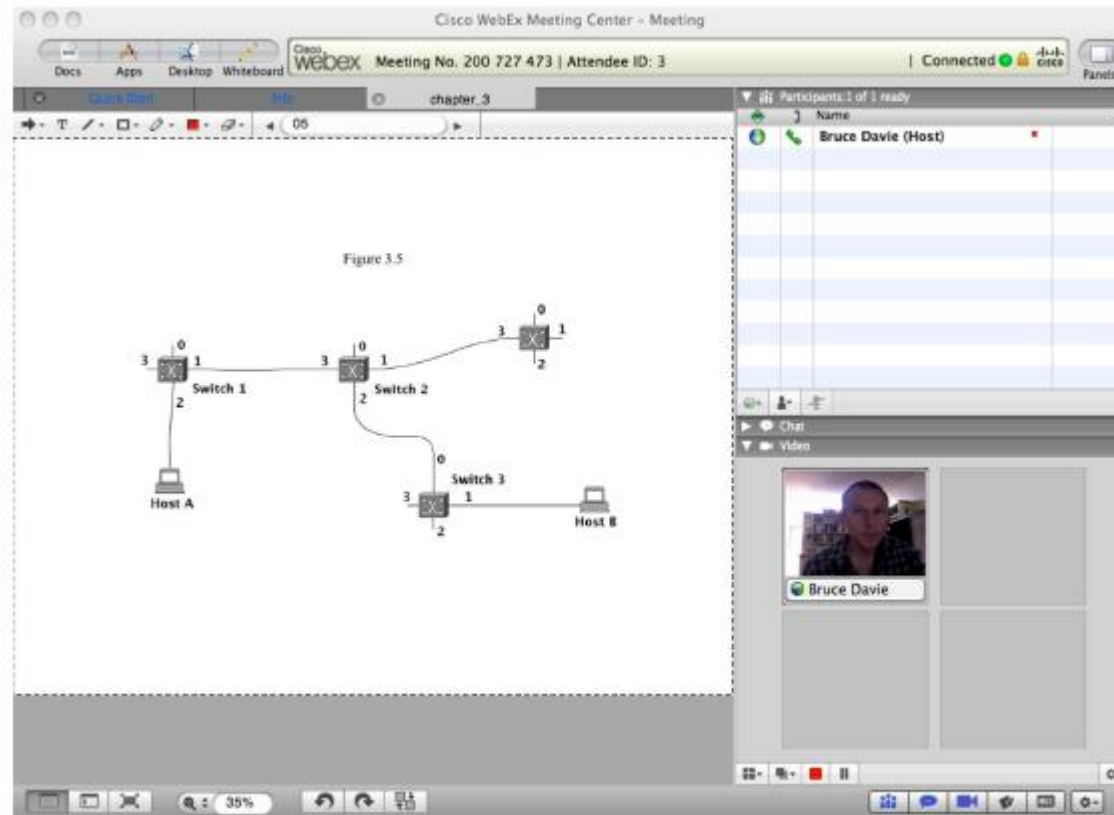
Problems

- How to build a scalable network that will support different applications?
- What is a computer network?
- How is a computer network different from other types of networks?
- What is a computer network architecture?

Applications

- Most people know about the Internet (a computer network) through applications
 - World Wide Web
 - Email
 - Online Social Network
 - Streaming Audio Video
 - File Sharing
 - Instant Messaging
 - ...

Example of an application



A multimedia application including video-conferencing

Application Protocol

- URL
 - Uniform resource locator
 - <http://www.cs.princeton.edu/~llp/index.html>
- HTTP
 - Hyper Text Transfer Protocol
- TCP
 - Transmission Control Protocol
- 17 messages for one URL request
 - 6 to find the IP (Internet Protocol) address
 - 3 for connection establishment of TCP
 - 4 for HTTP request and acknowledgement
 - Request: I got your request and I will send the data
 - Reply: Here is the data you requested; I got the data
 - 4 messages for tearing down TCP connection

Topic 2

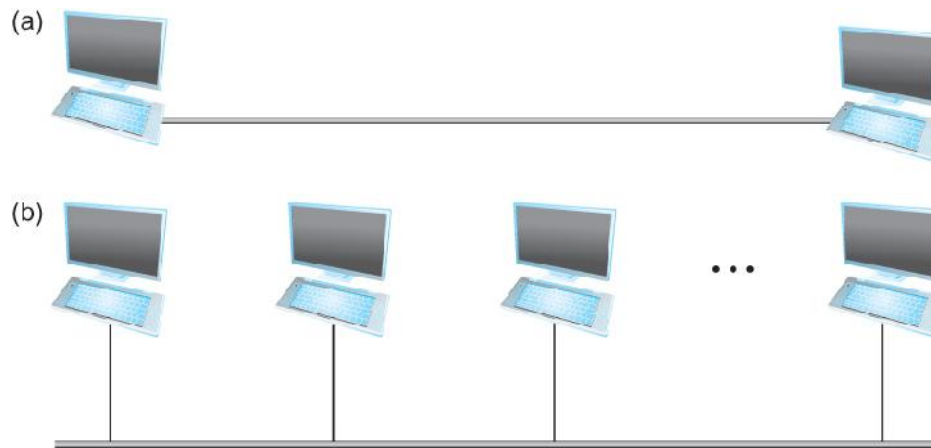
REQUIREMENTS

Requirements

- Application Programmer
 - List the services that his application needs: delay bounded delivery of data
- Network Designer
 - Design a cost-effective network with sharable resources
- Network Provider
 - List the characteristics of a system that is easy to manage

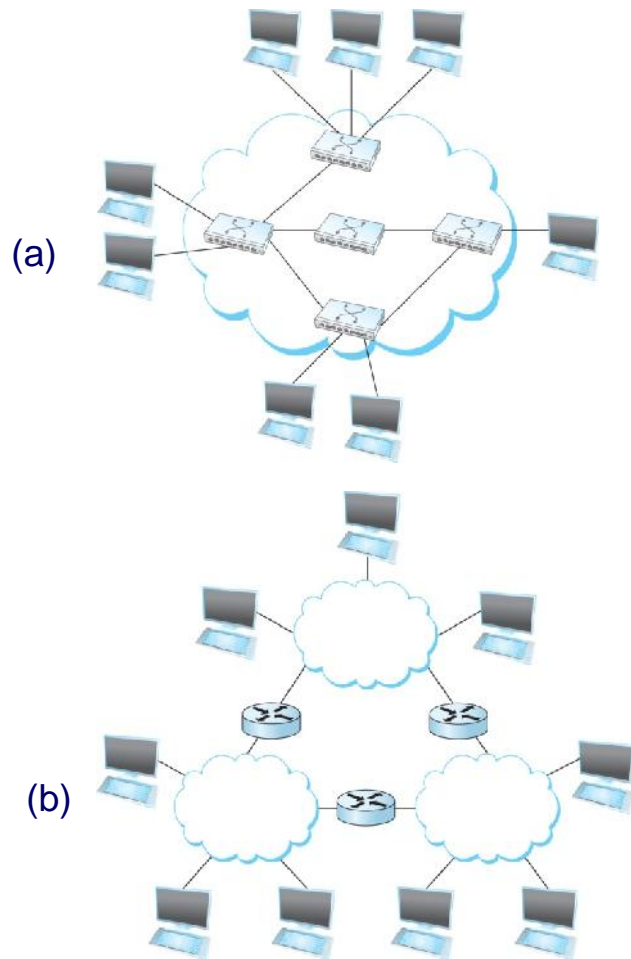
Connectivity

- Need to understand the following terminologies
 - Scale
 - Link
 - Nodes
 - Point-to-point
 - Multiple access
 - Switched Network
 - Circuit Switched
 - Packet Switched
 - Packet, message
 - Store-and-forward



- (a) Point-to-point
- (b) Multiple access

Connectivity

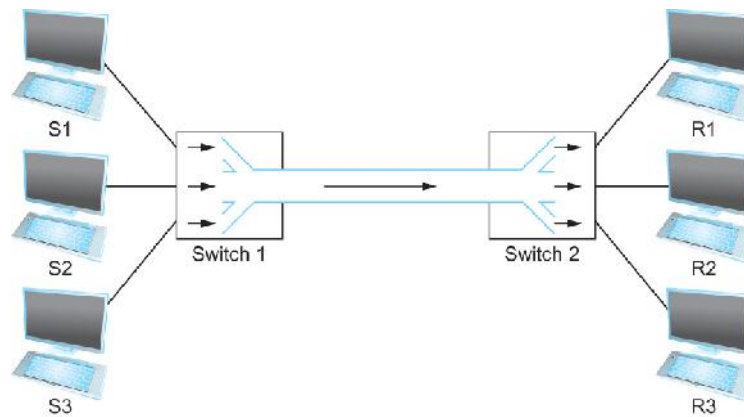


(a) A switched network

(b) Interconnection of networks

- Terminologies (contd.)
 - Cloud
 - Hosts
 - Switches
 - internetwork
 - Router/gateway
 - Host-to-host connectivity
 - Address
 - Routing
 - Unicast/broadcast/multicast

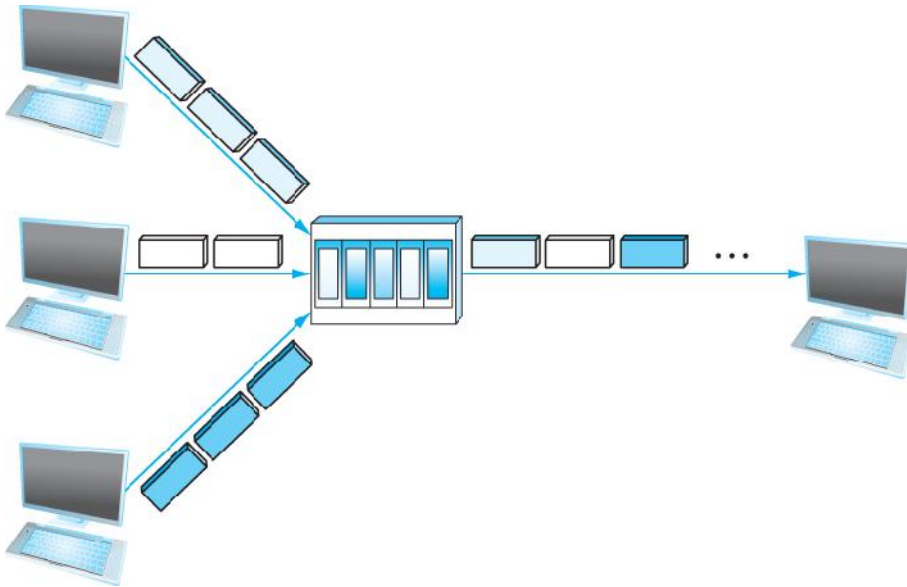
Cost-Effective Resource Sharing



Multiplexing multiple logical flows
over a single physical link

- Resource: links and nodes
- How to share a link?
 - Multiplexing
 - De-multiplexing
 - Synchronous Time-division Multiplexing
 - Time slots/data transmitted in predetermined slots

Cost-Effective Resource Sharing

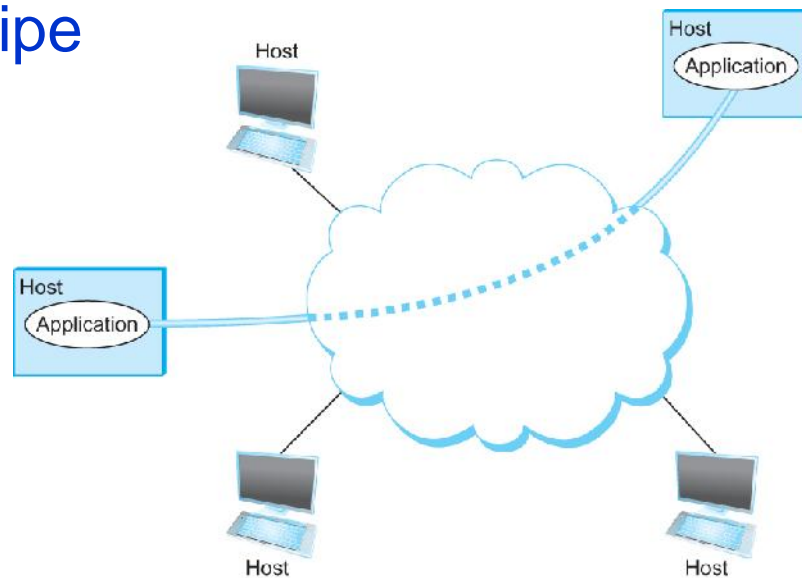


A switch multiplexing packets from multiple sources onto one shared link

- FDM: Frequency Division Multiplexing
- Statistical Multiplexing
 - Data is transmitted based on demand of each flow.
 - What is a flow?
 - Packets vs. Messages
 - FIFO, Round-Robin, Priorities (Quality-of-Service (QoS))
 - Congested?
- LAN, MAN, WAN
- SAN (System Area Networks)

Support for Common Services

- Logical Channels
 - Application-to-Application communication path or a pipe



Process communicating over an abstract channel

Common Communication Patterns

- Client/Server
- Two types of communication channel
 - Request/Reply Channels
 - Message Stream Channels

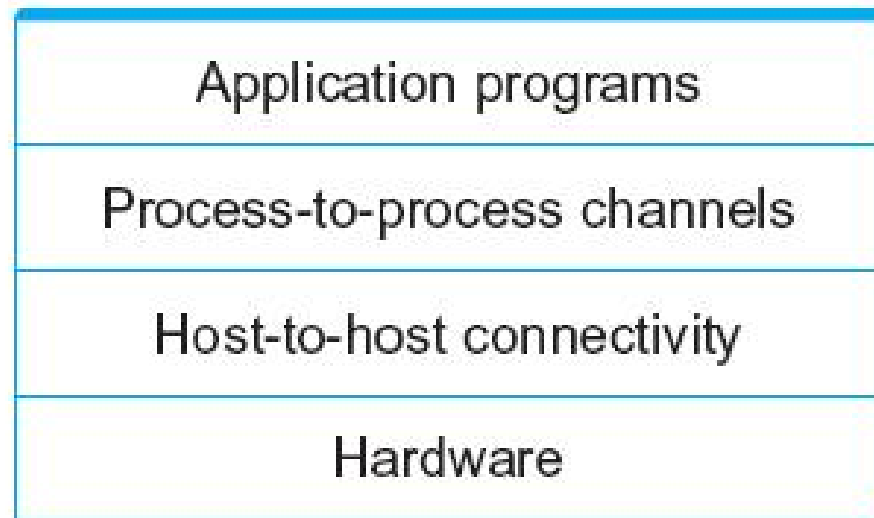
Reliability

- Network should hide the errors
- Bits are lost
 - Bit errors (1 to a 0, and vice versa)
 - Burst errors – several consecutive errors
- Packets are lost (Congestion)
- Links and Node failures
- Messages are delayed
- Messages are delivered out-of-order
- Third parties eavesdrop

Topic 3

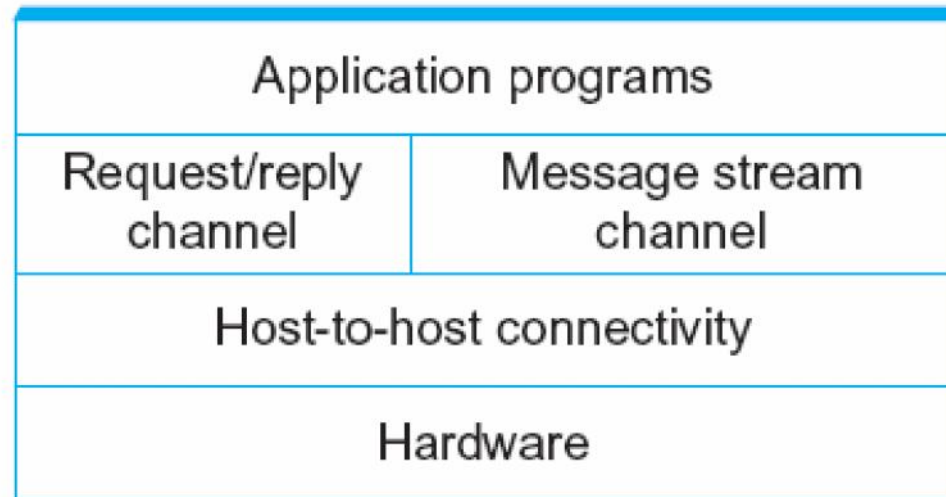
Layering and Protocols

Network Architecture



Example of a layered network system

Network Architecture

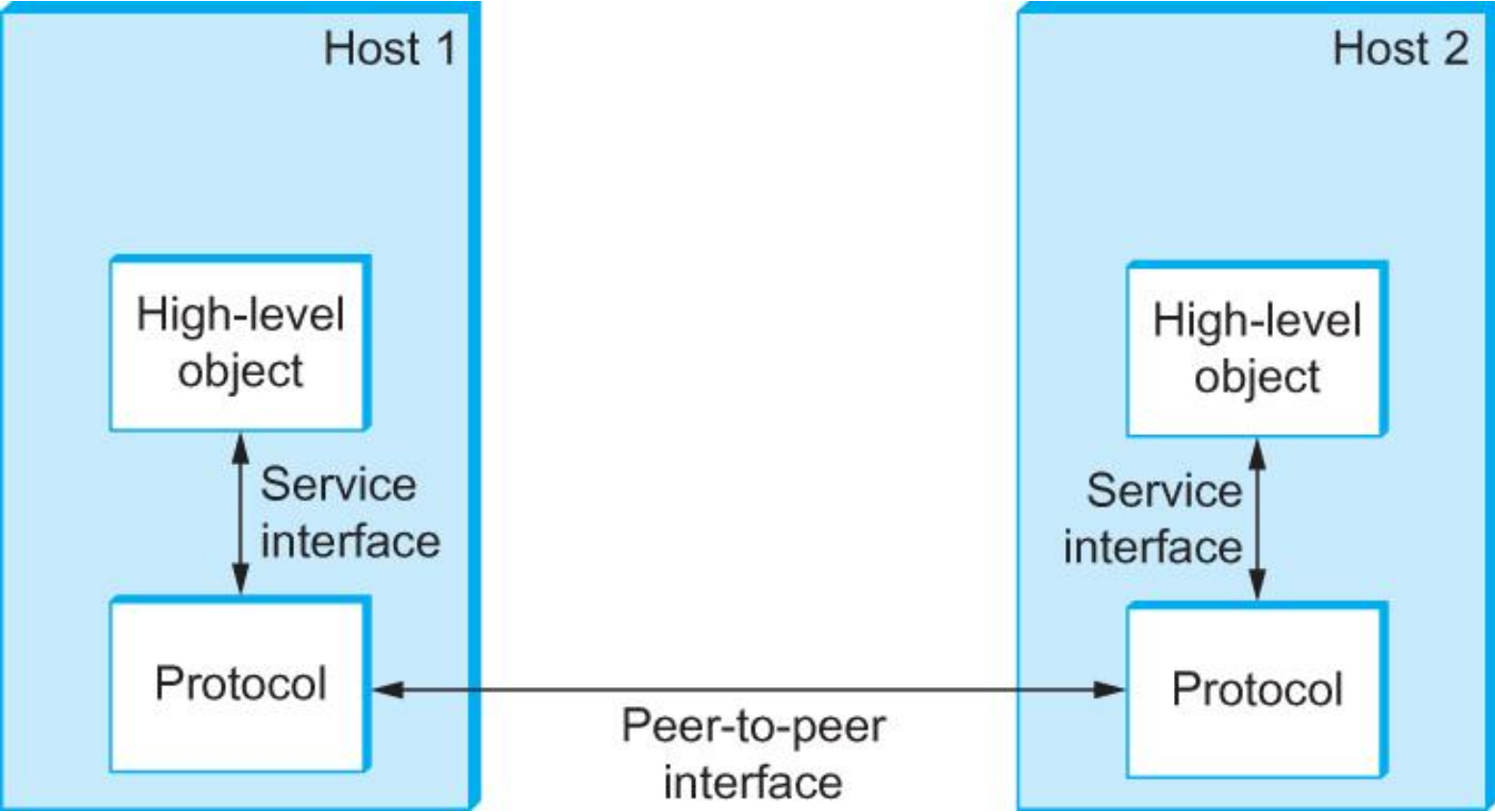


Layered system with alternative abstractions available at a given layer

Protocols

- Protocol defines the interfaces between the layers in the same system and with the layers of peer system
- Building blocks of a network architecture
- Each protocol object has two different interfaces
 - service interface: operations on this protocol
 - peer-to-peer interface: messages exchanged with peer
- Term “protocol” is overloaded
 - specification of peer-to-peer interface
 - module that implements this interface

Interfaces

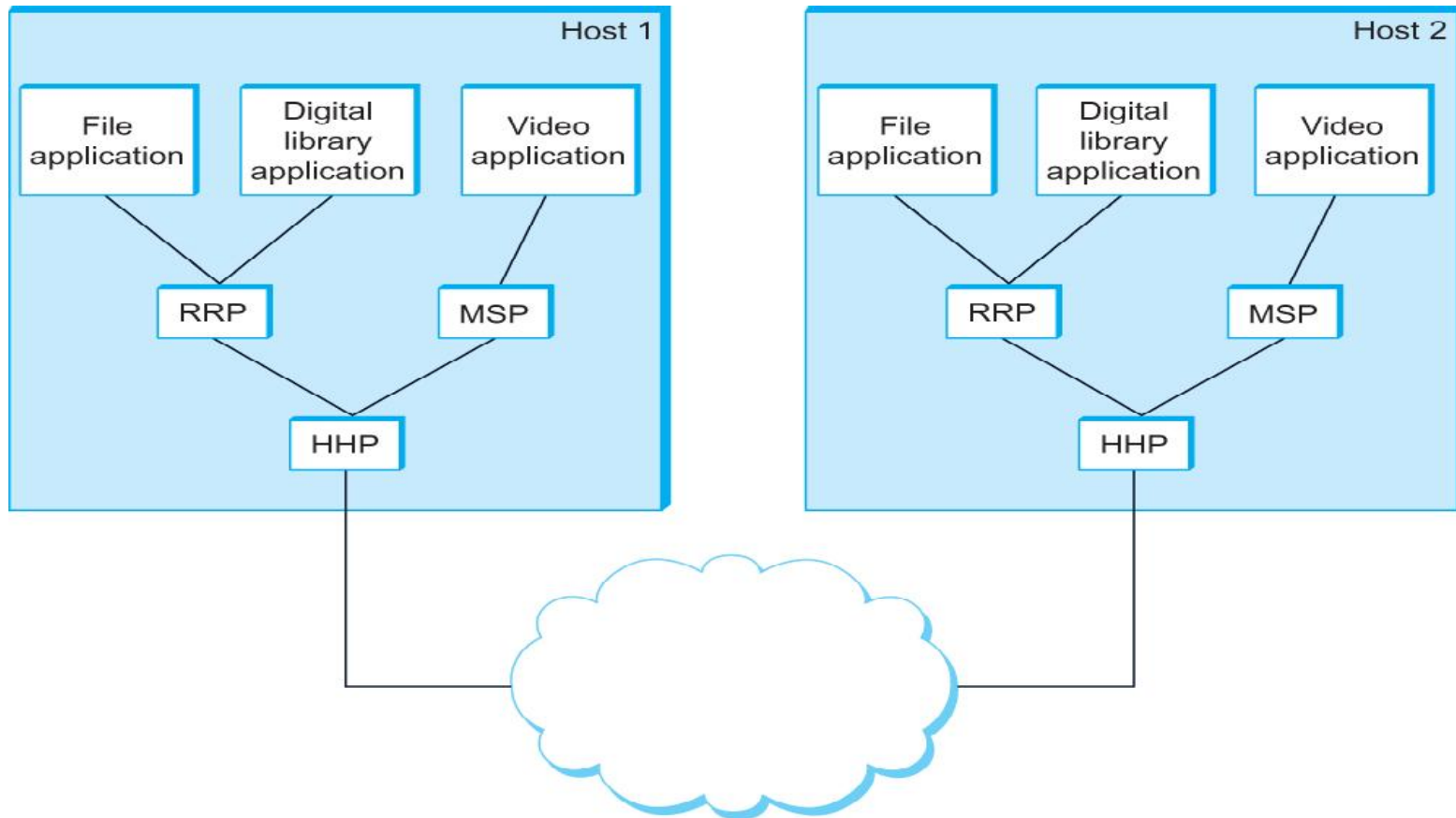


Service and Peer Interfaces

Protocols

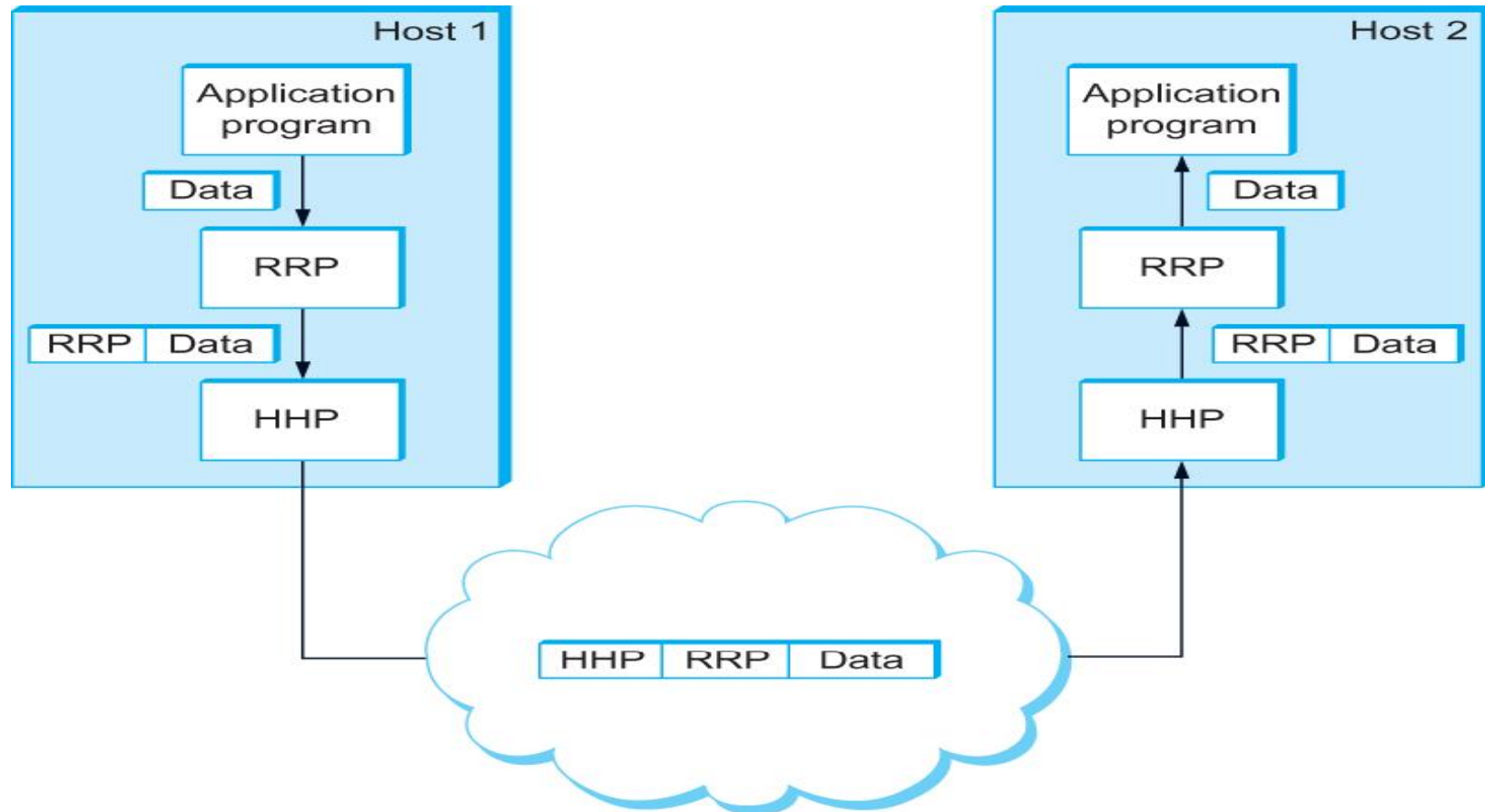
- Protocol Specification: prose, pseudo-code, state transition diagram
- Interoperable: when two or more protocols that implement the specification accurately
- IETF: Internet Engineering Task Force

Protocol Graph



Example of a protocol graph
nodes are the protocols and links the “depends-on” relation

Encapsulation

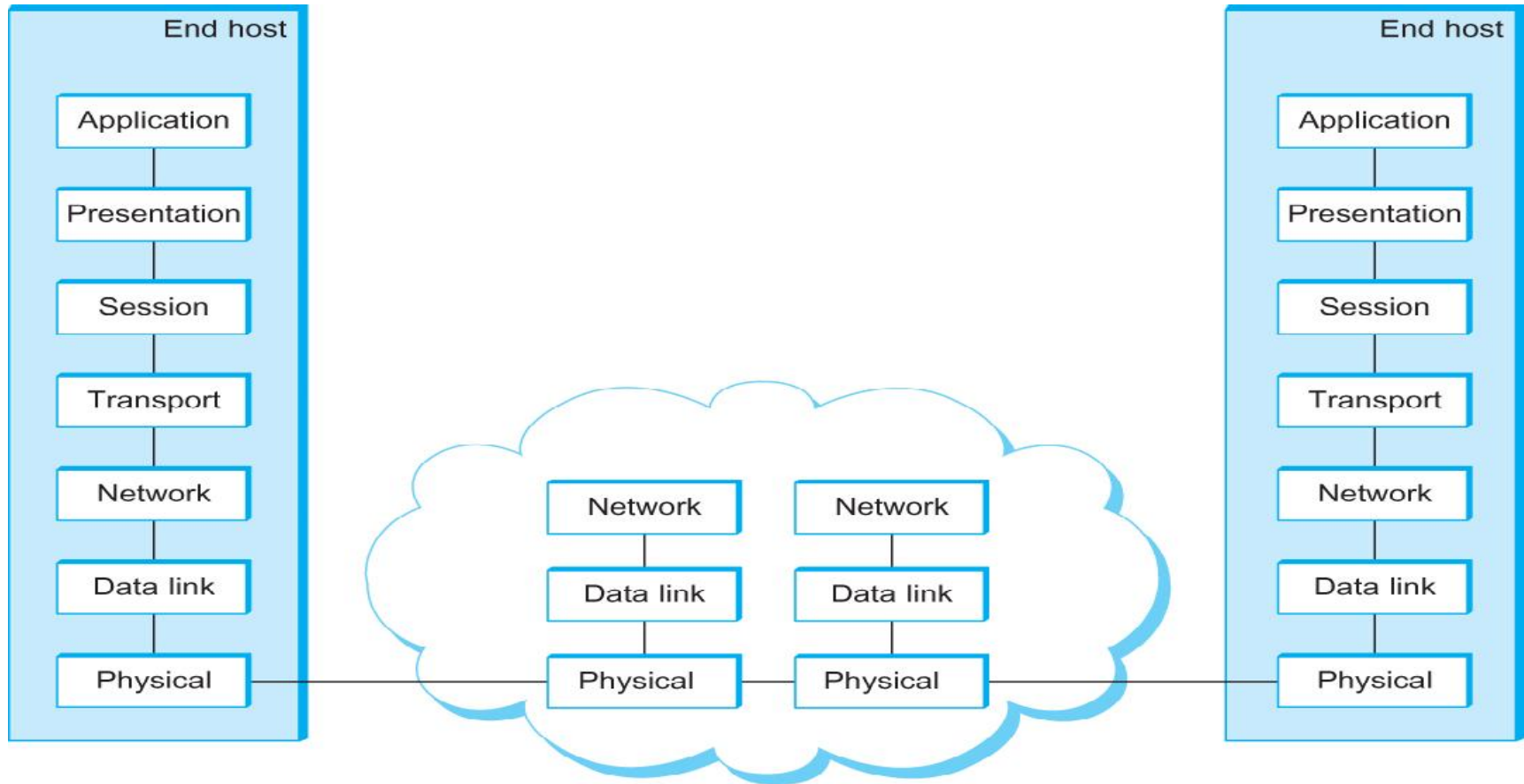


High-level messages are encapsulated inside of low-level messages

Topic 4

OSI Model

OSI Architecture



One or more nodes within the network

The OSI 7-layer Model

OSI – Open Systems Interconnection

Description of Layers

- Physical Layer
 - Handles the transmission of raw bits over a communication link
- Data Link Layer
 - Collects a stream of bits into a larger aggregate called a *frame*
 - Network adaptor along with device driver in OS implement the protocol in this layer
 - Frames are actually delivered to hosts
- Network Layer
 - Handles routing among nodes within a packet-switched network
 - Unit of data exchanged between nodes in this layer is called a *packet*

The lower three layers are implemented on all network nodes

Description of Layers

- Transport Layer
 - Implements a process-to-process channel
 - Unit of data exchanges in this layer is called a *message*
- Session Layer
 - Provides a name space that is used to tie together the potentially different transport streams that are part of a single application
- Presentation Layer
 - Concerned about the format of data exchanged between peers
- Application Layer
 - Standardize common type of exchanges

The transport layer and the higher layers typically run only on end-hosts and not on the intermediate switches and routers

Figure 3-1

OSI Model

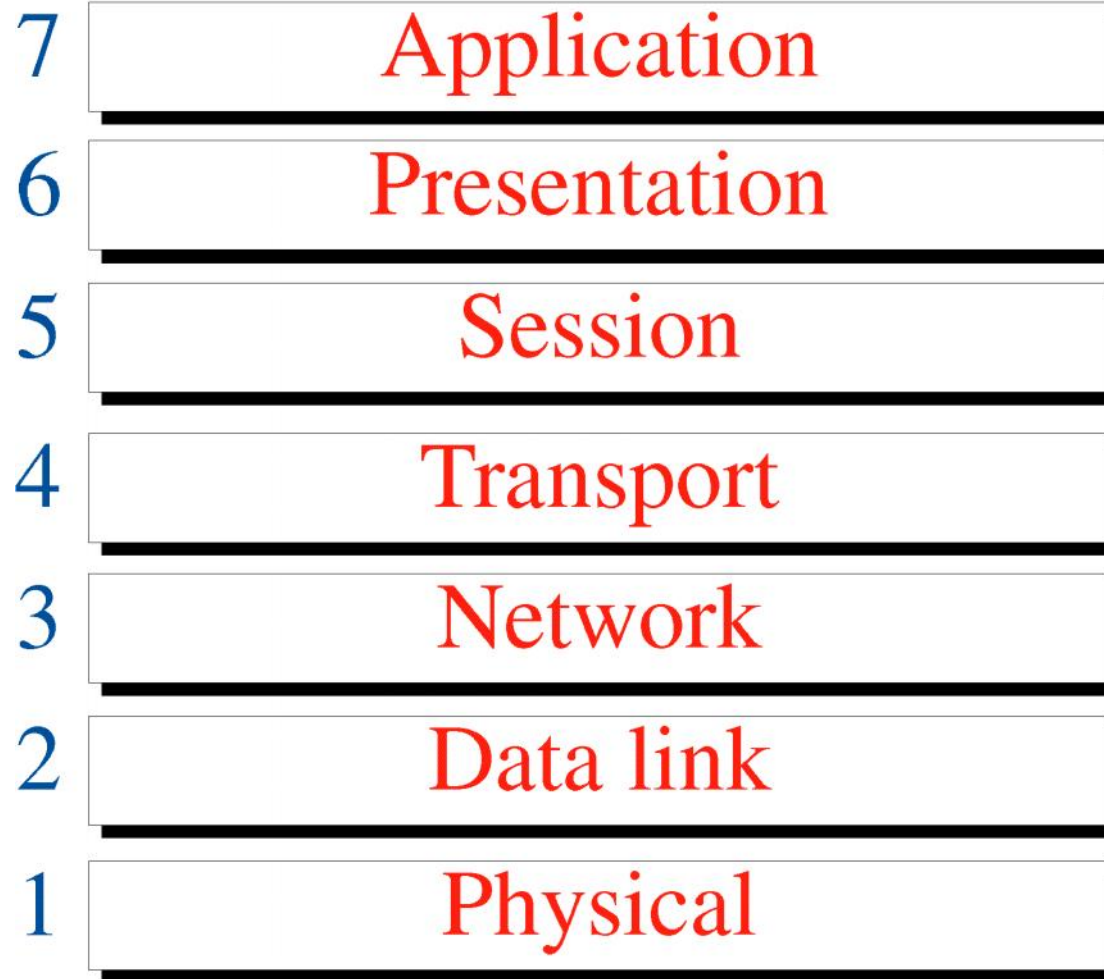


Figure 3-2

OSI Layers

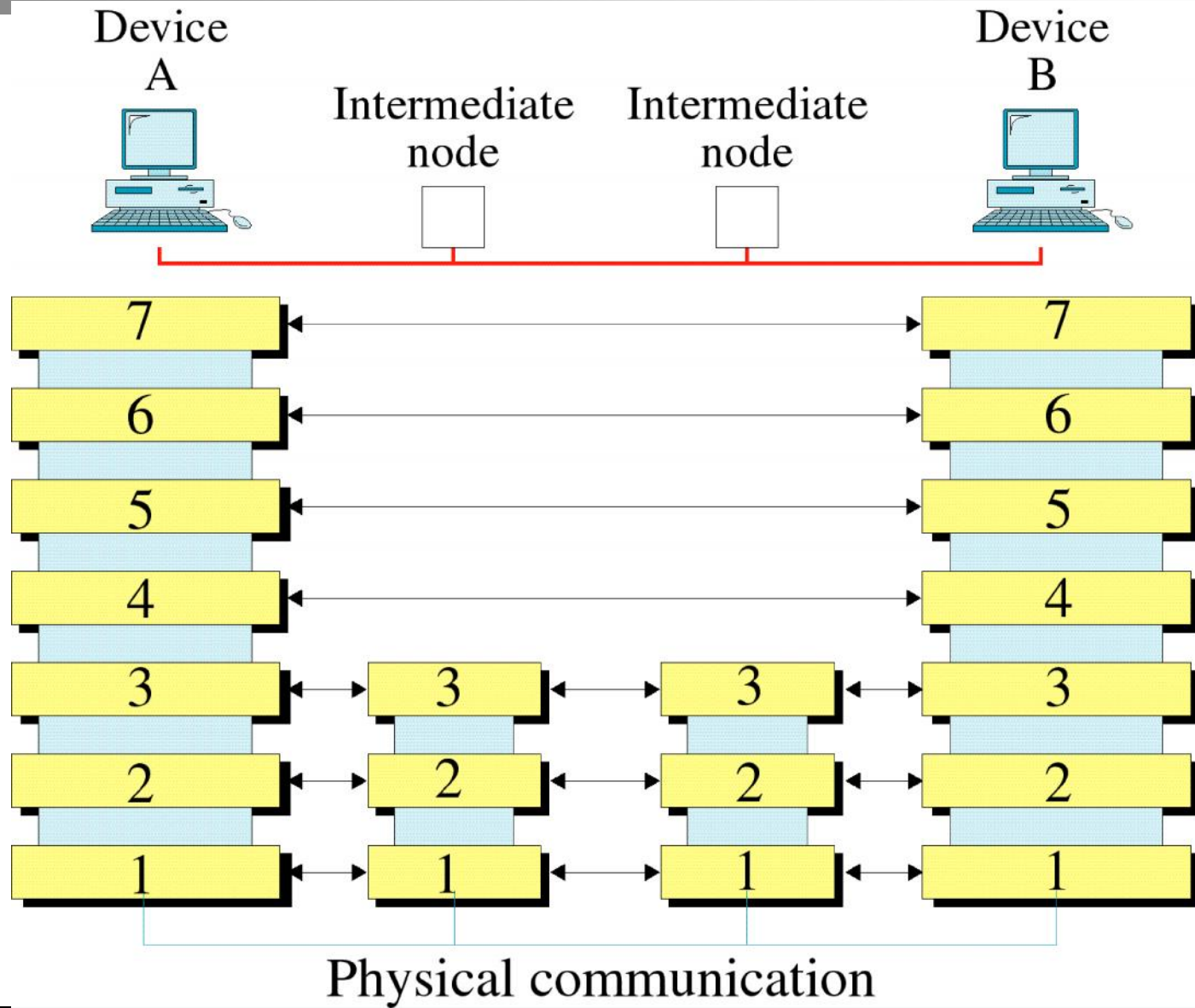


Figure 3-3

An Exchange Using the OSI Model

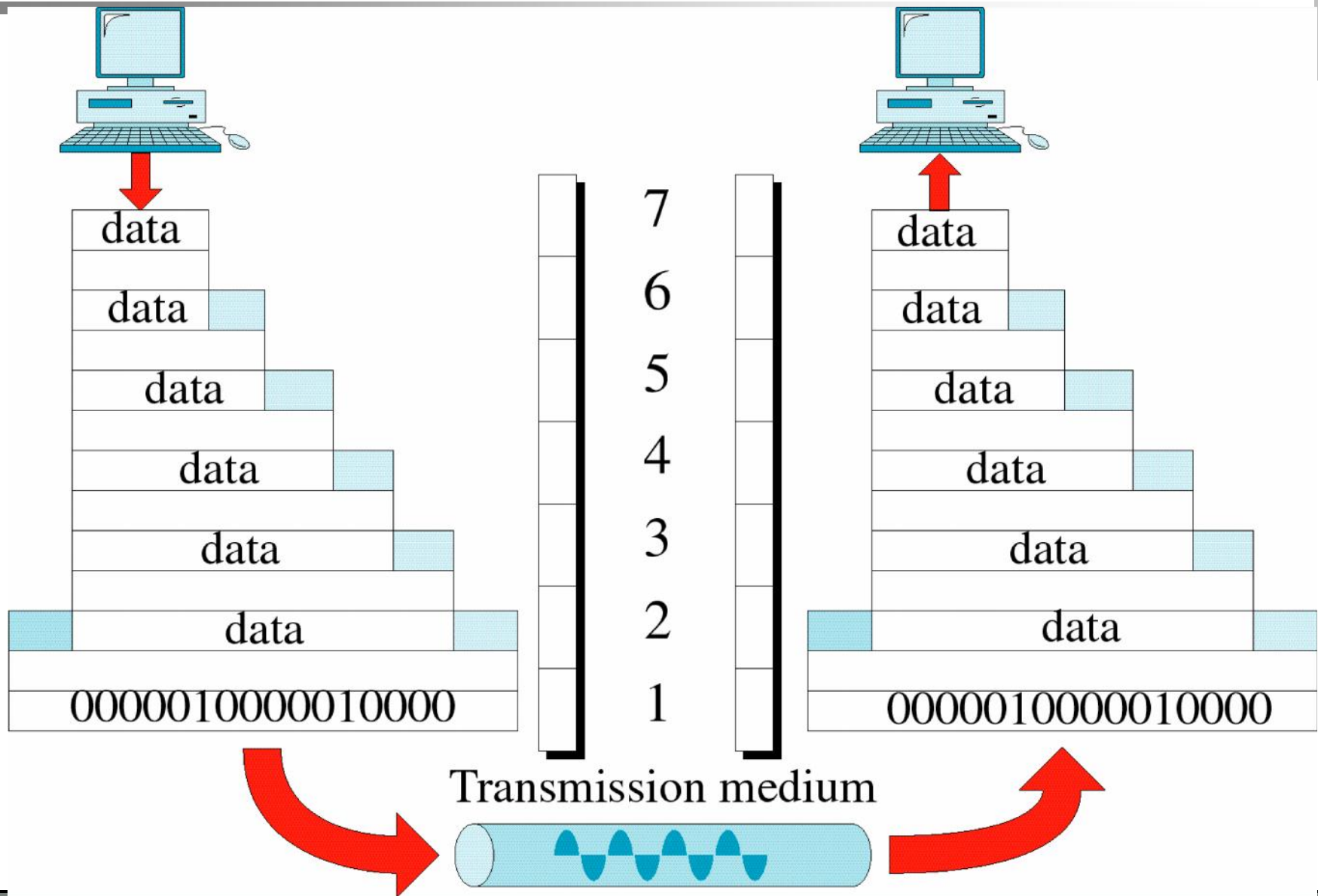


Figure 3-4

Physical Layer

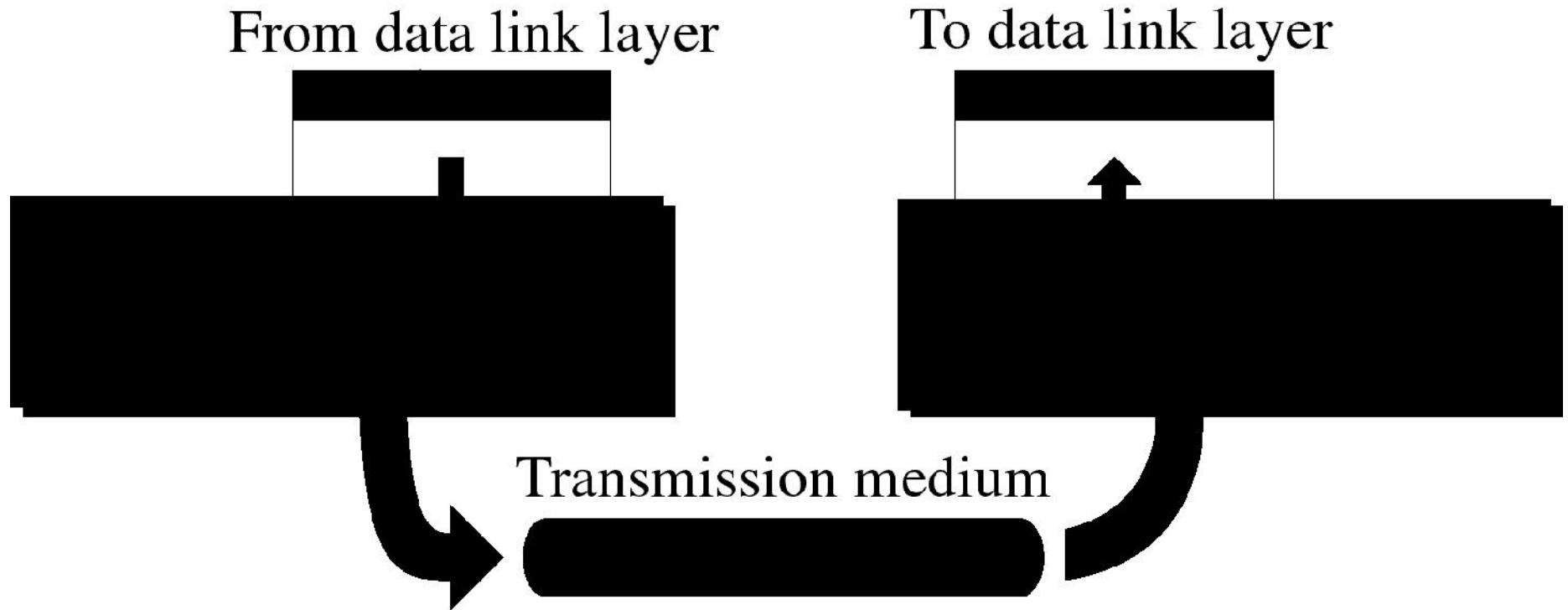


Figure 3-5

Data Link Layer

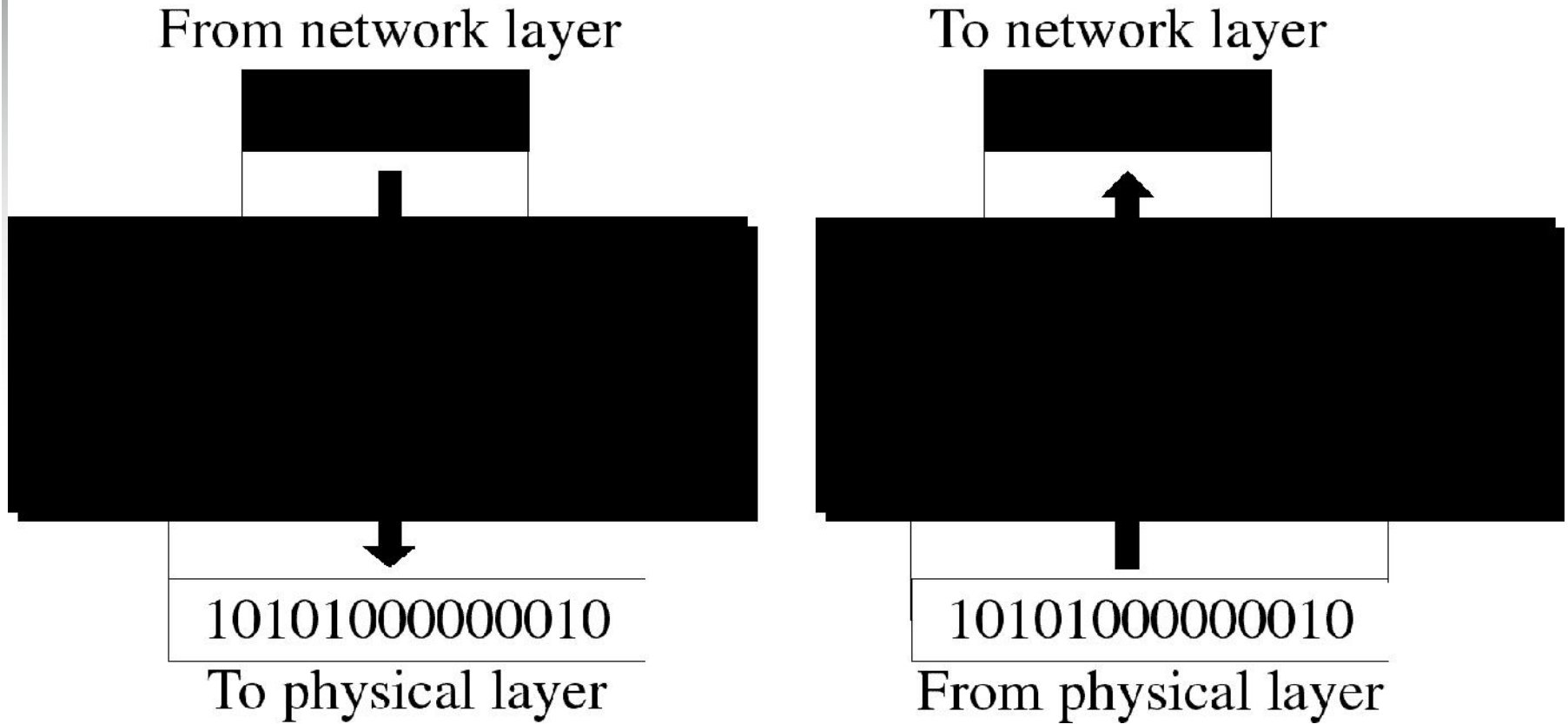


Figure 3-6

Data Link Layer Example

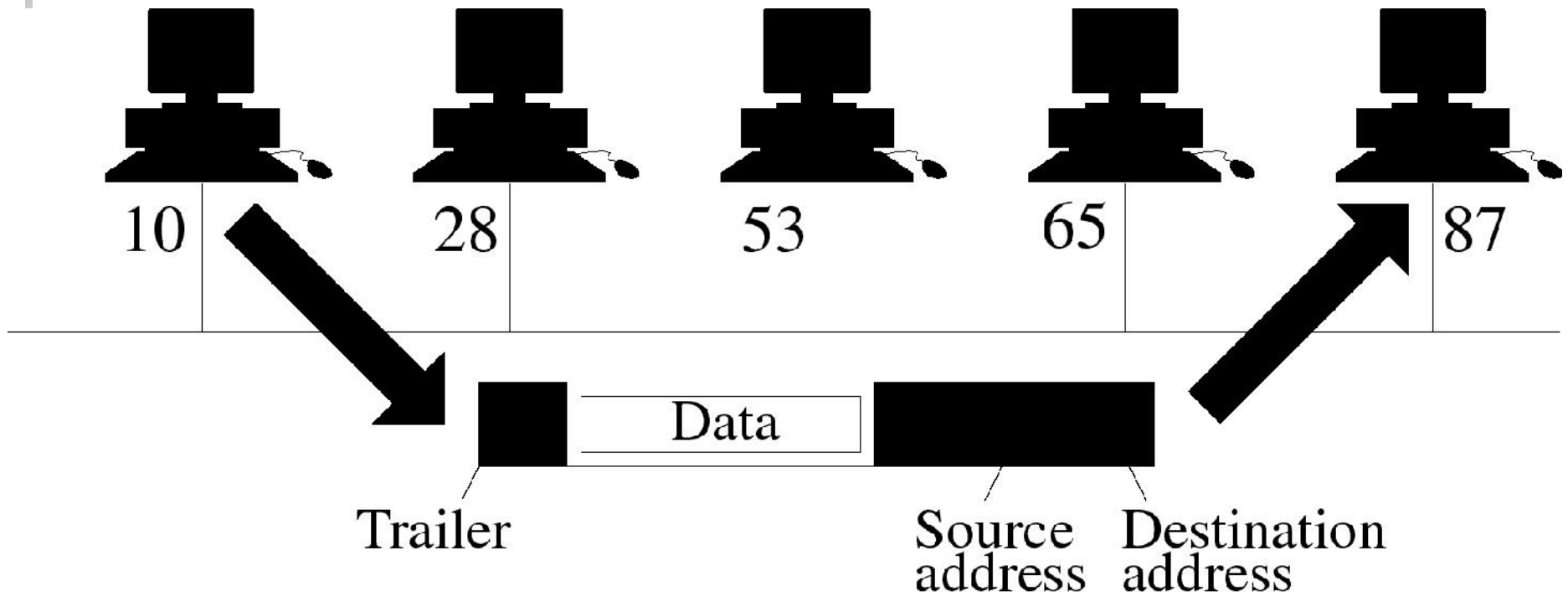


Figure 3-7

Network Layer

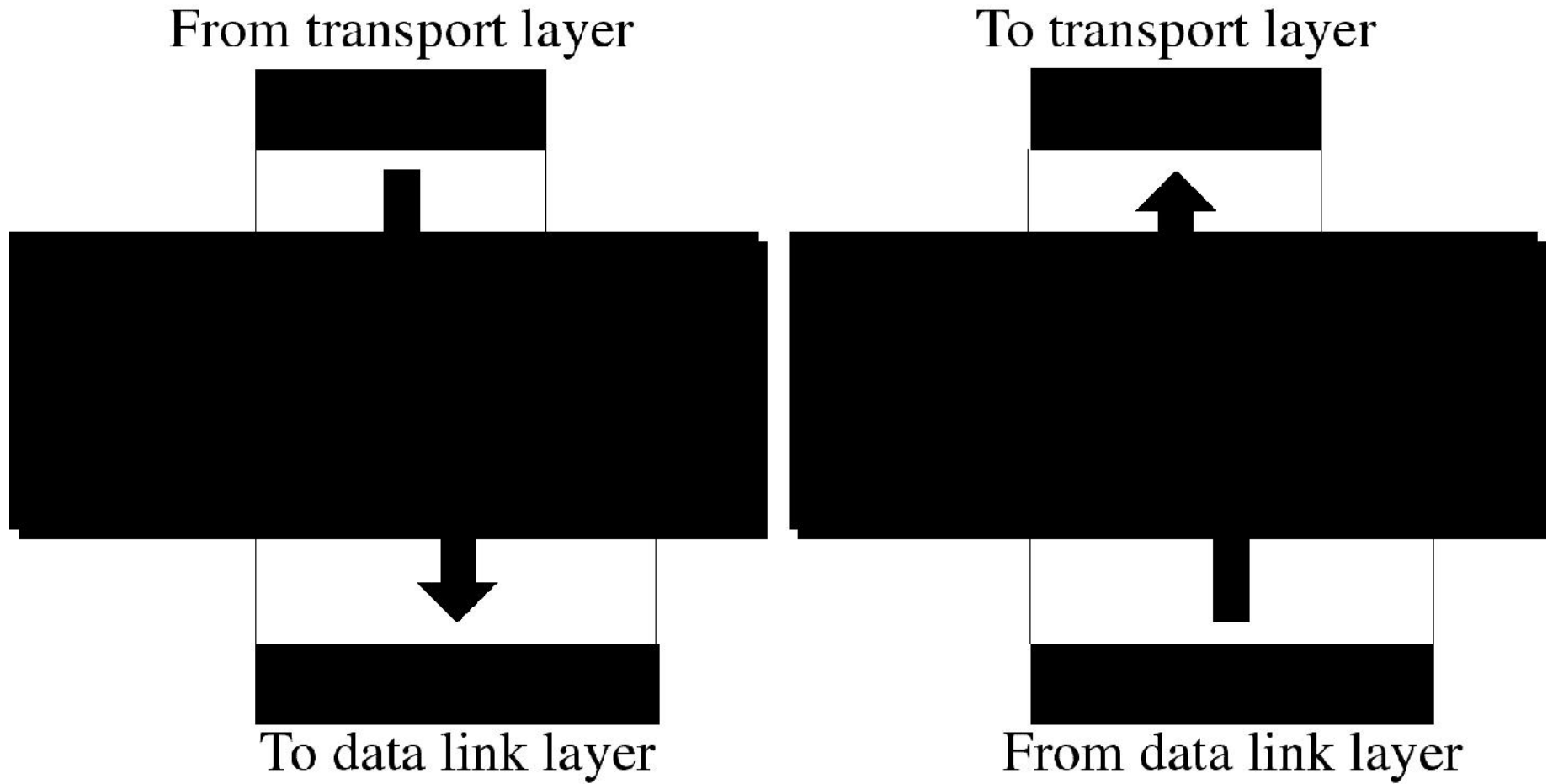
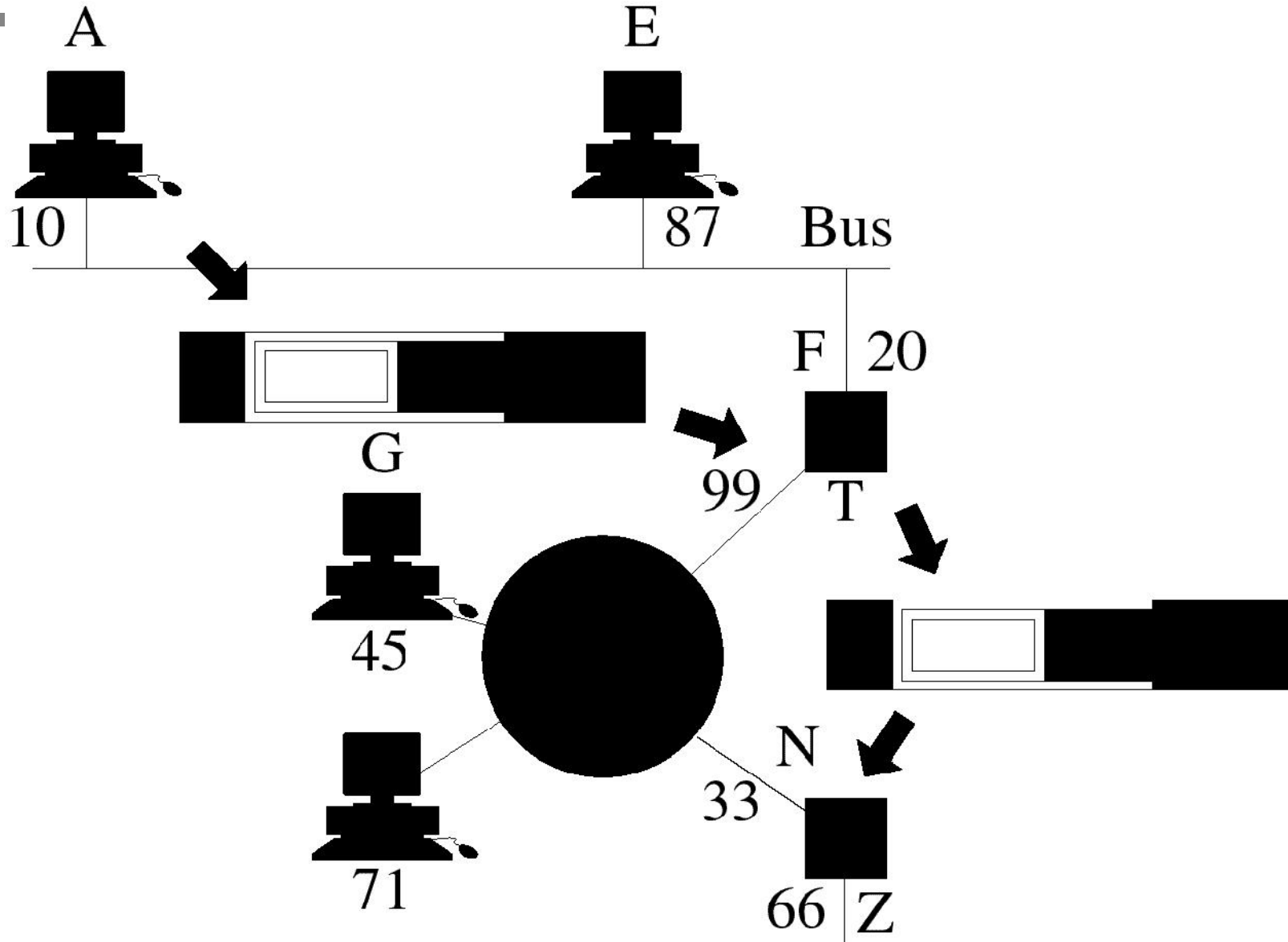


Figure 3-8

Network Layer Example



Network Layer Example

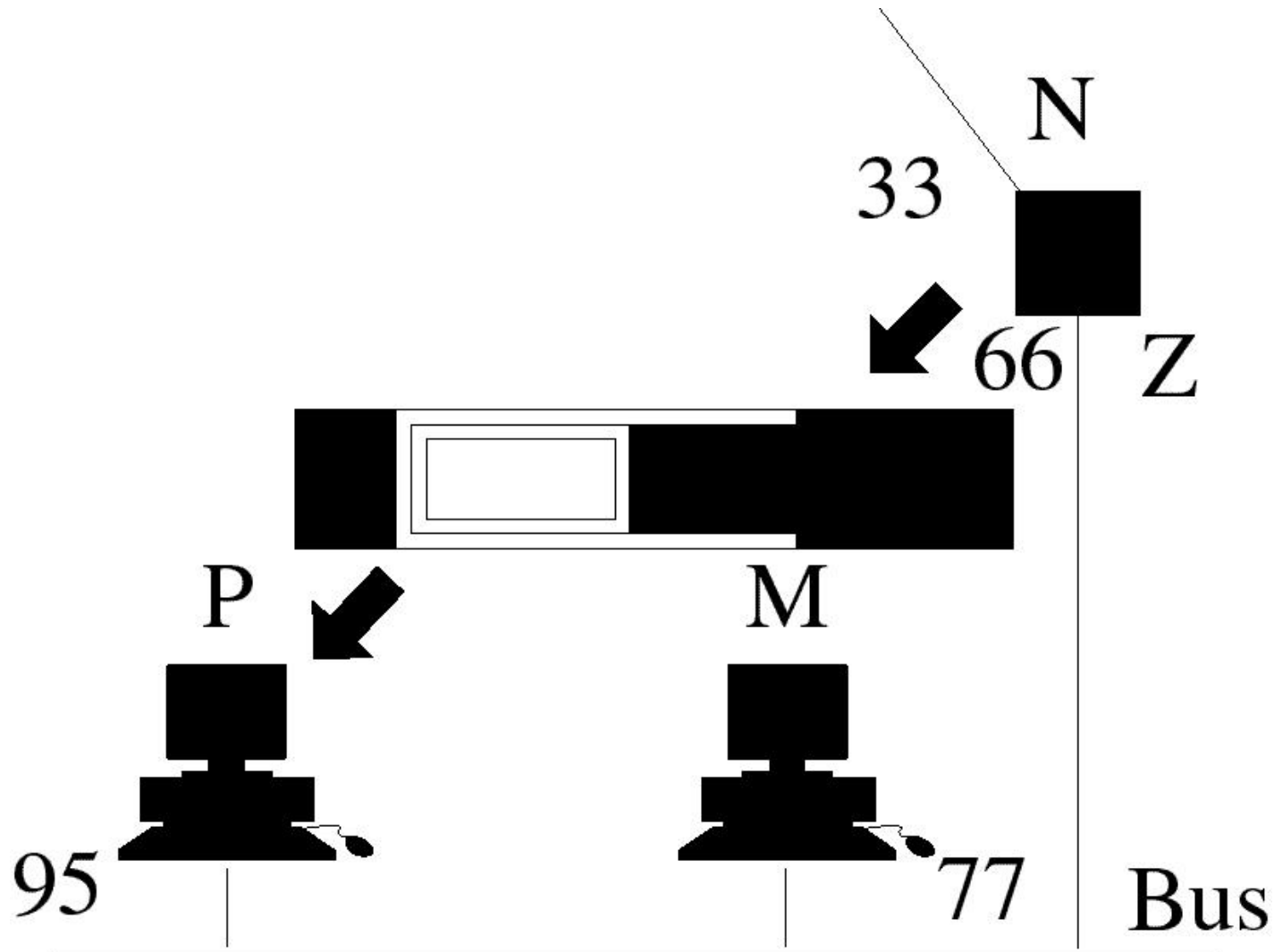


Figure 3-9

Transport Layer

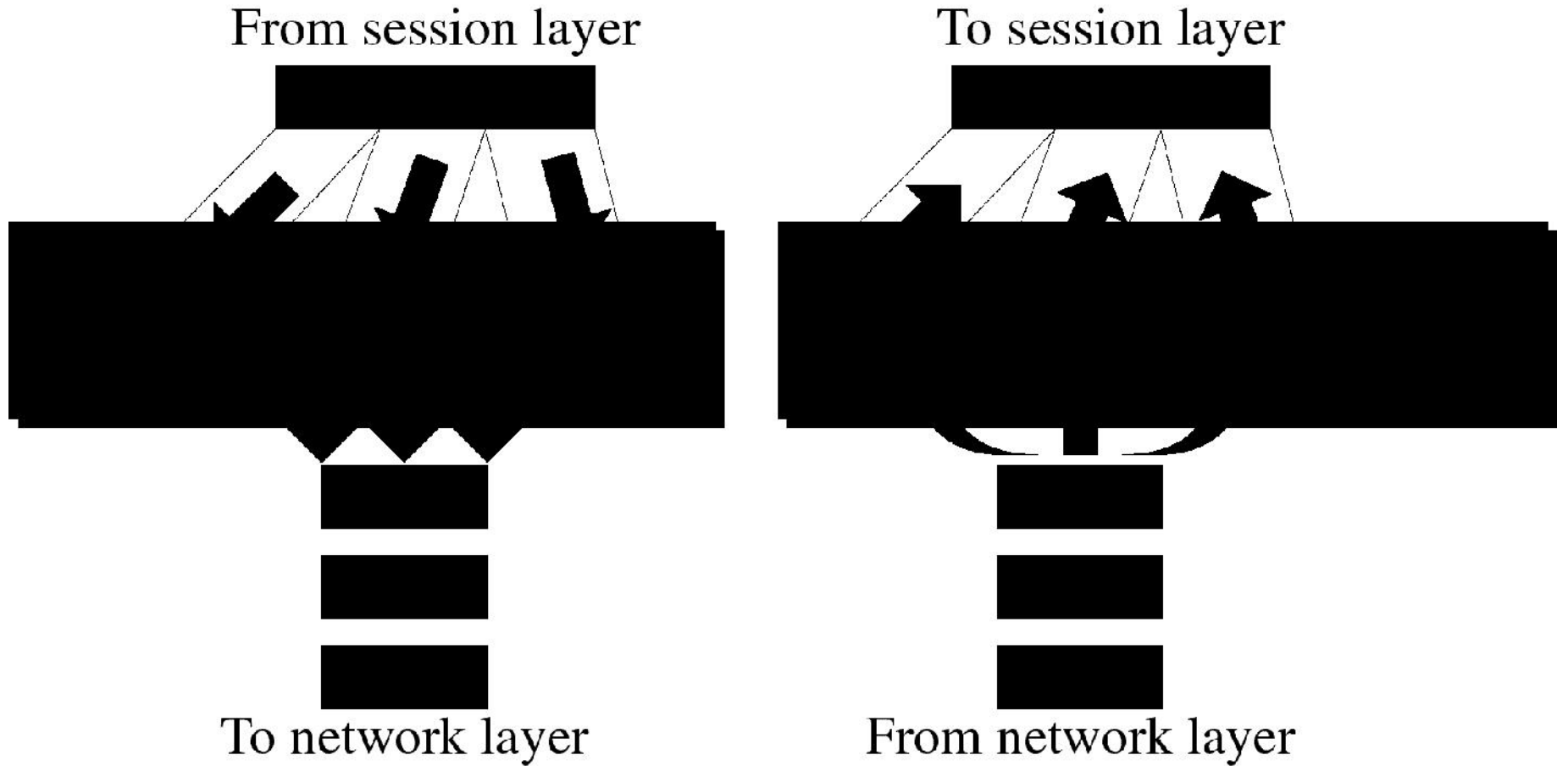
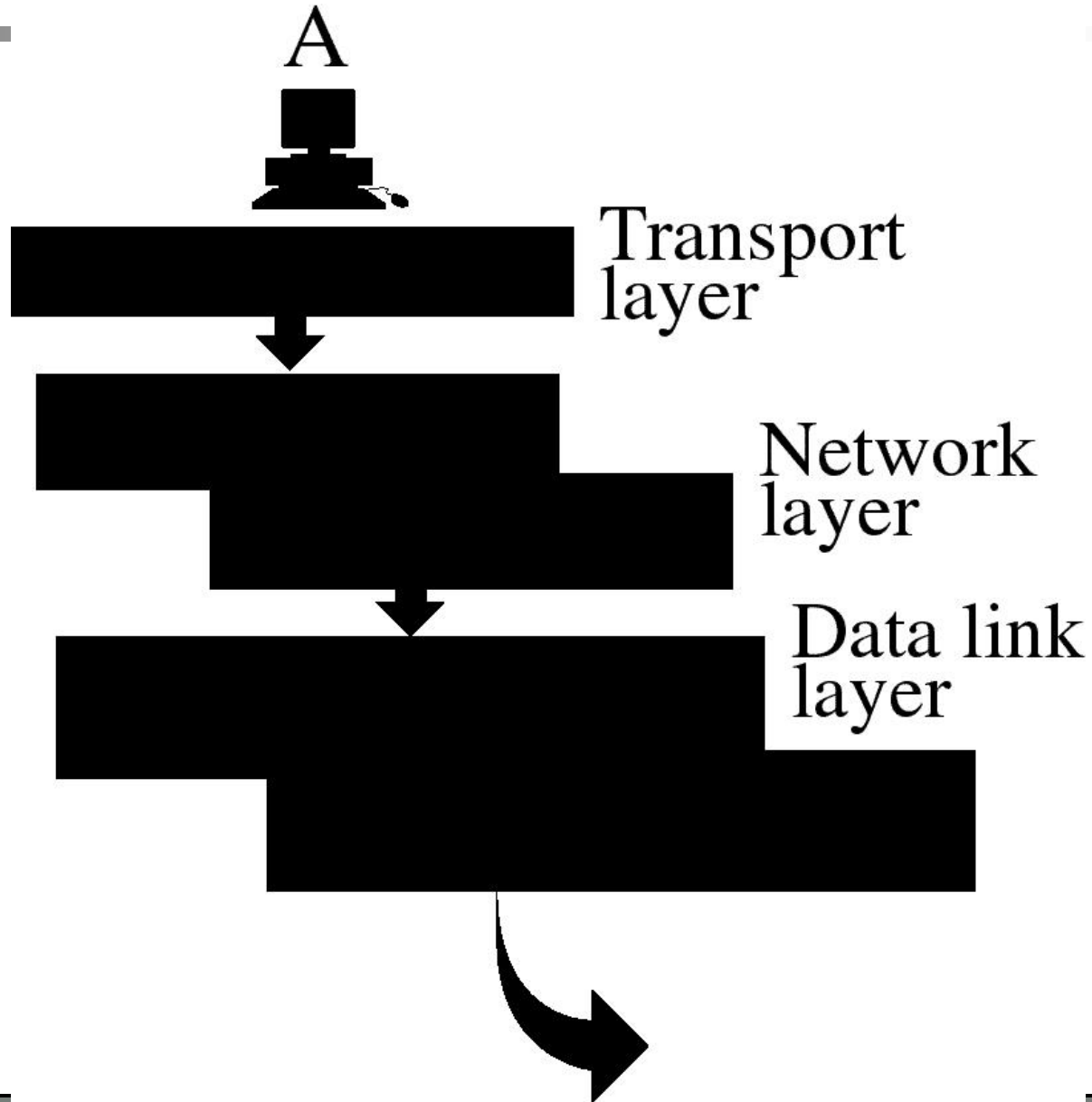


Figure 3-10

Transport Layer Example



Transport Layer Example

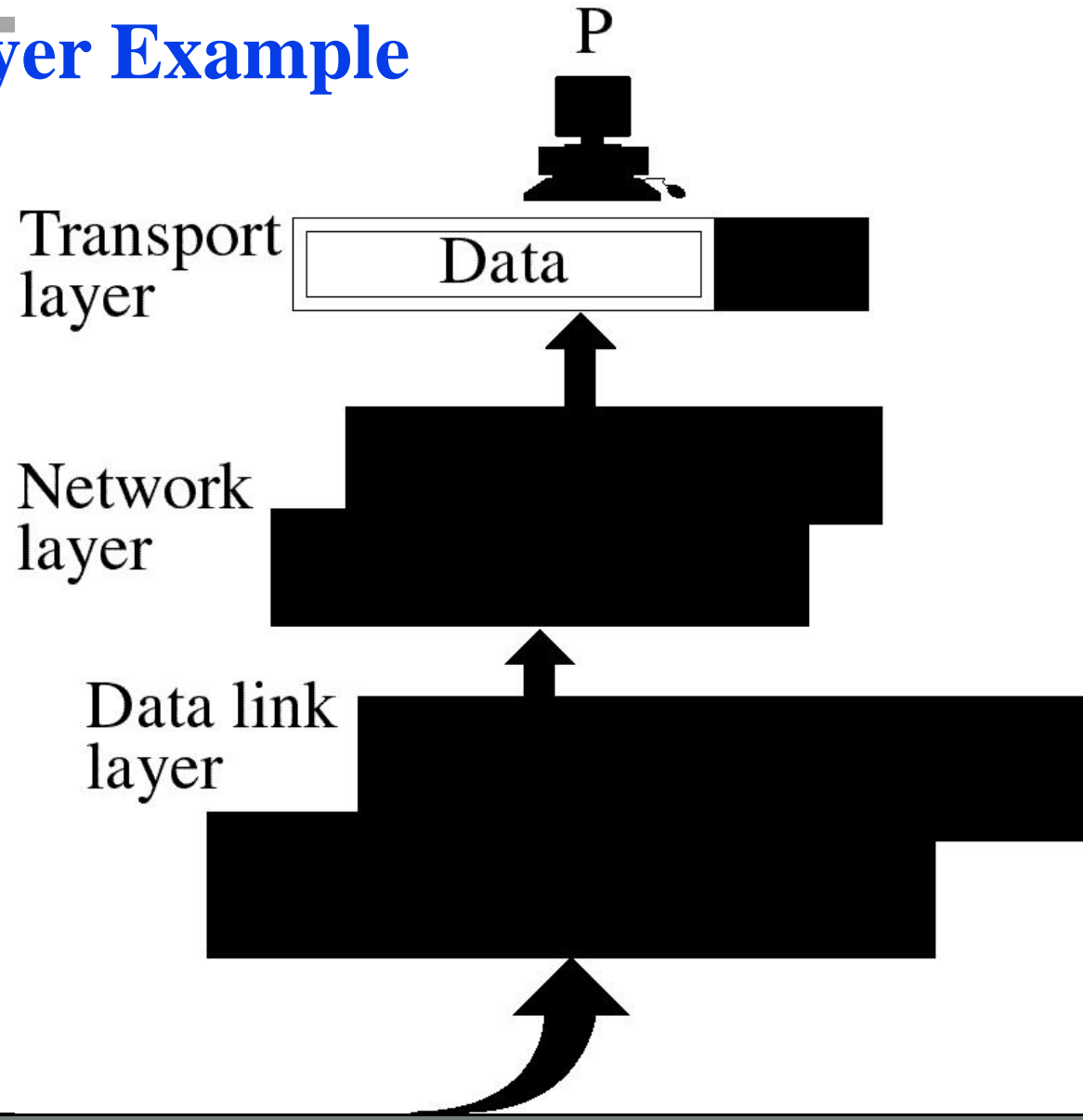


Figure 3-11

Session Layer

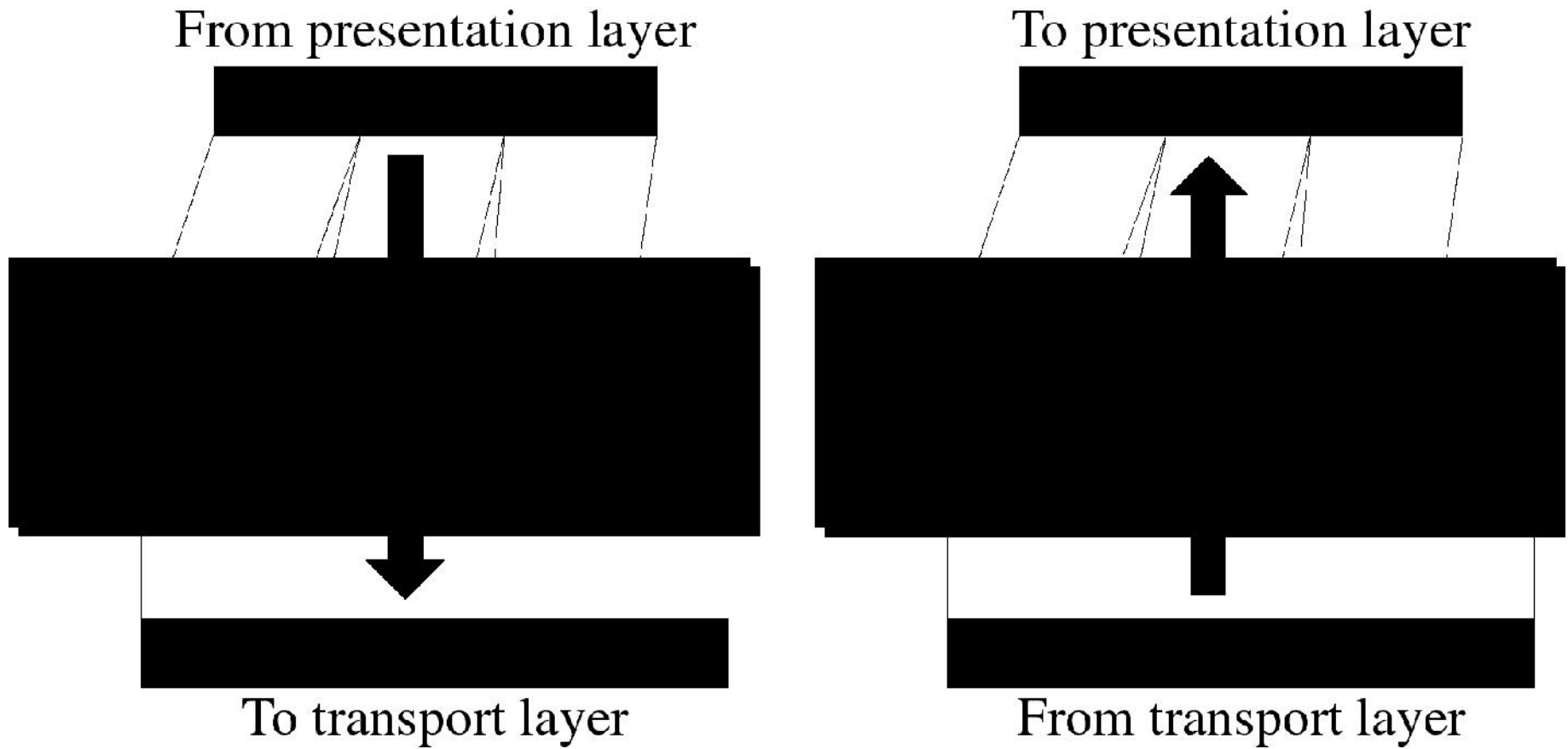


Figure 3-12

Presentation Layer

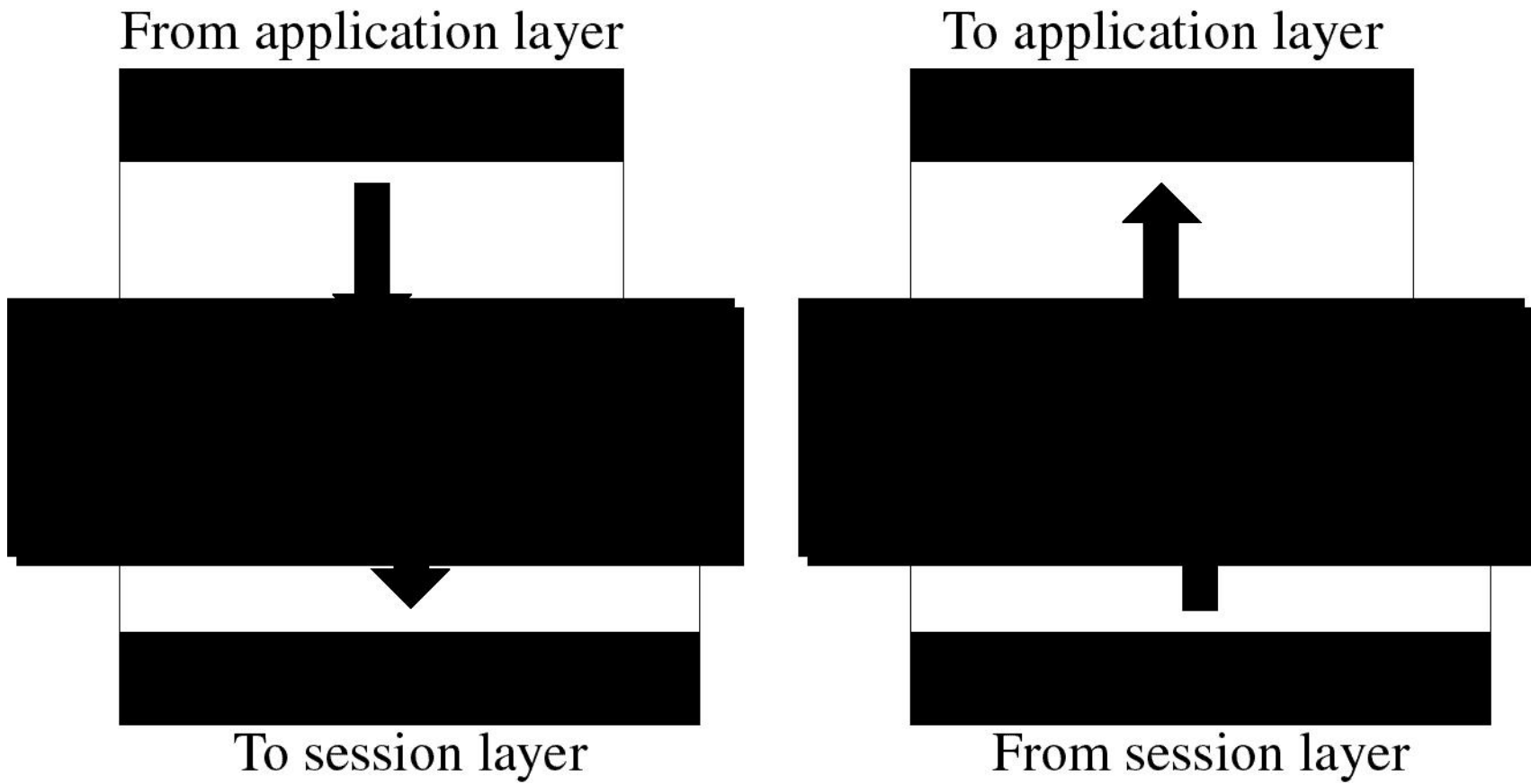
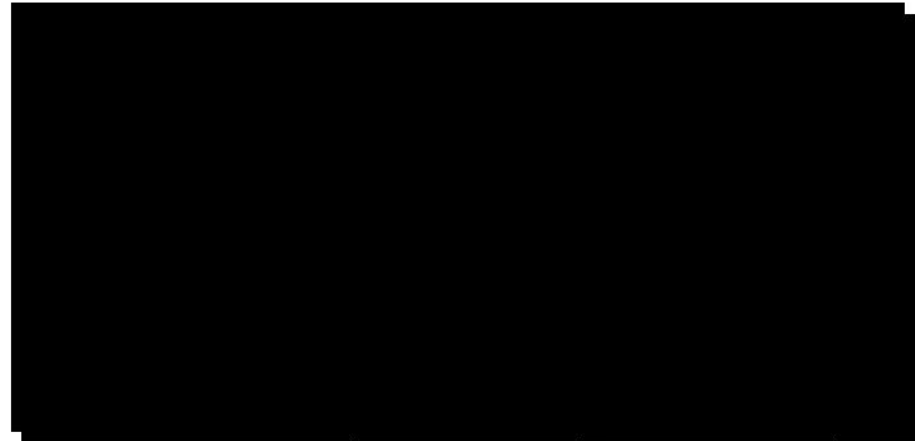


Figure 3-13

Application Layer

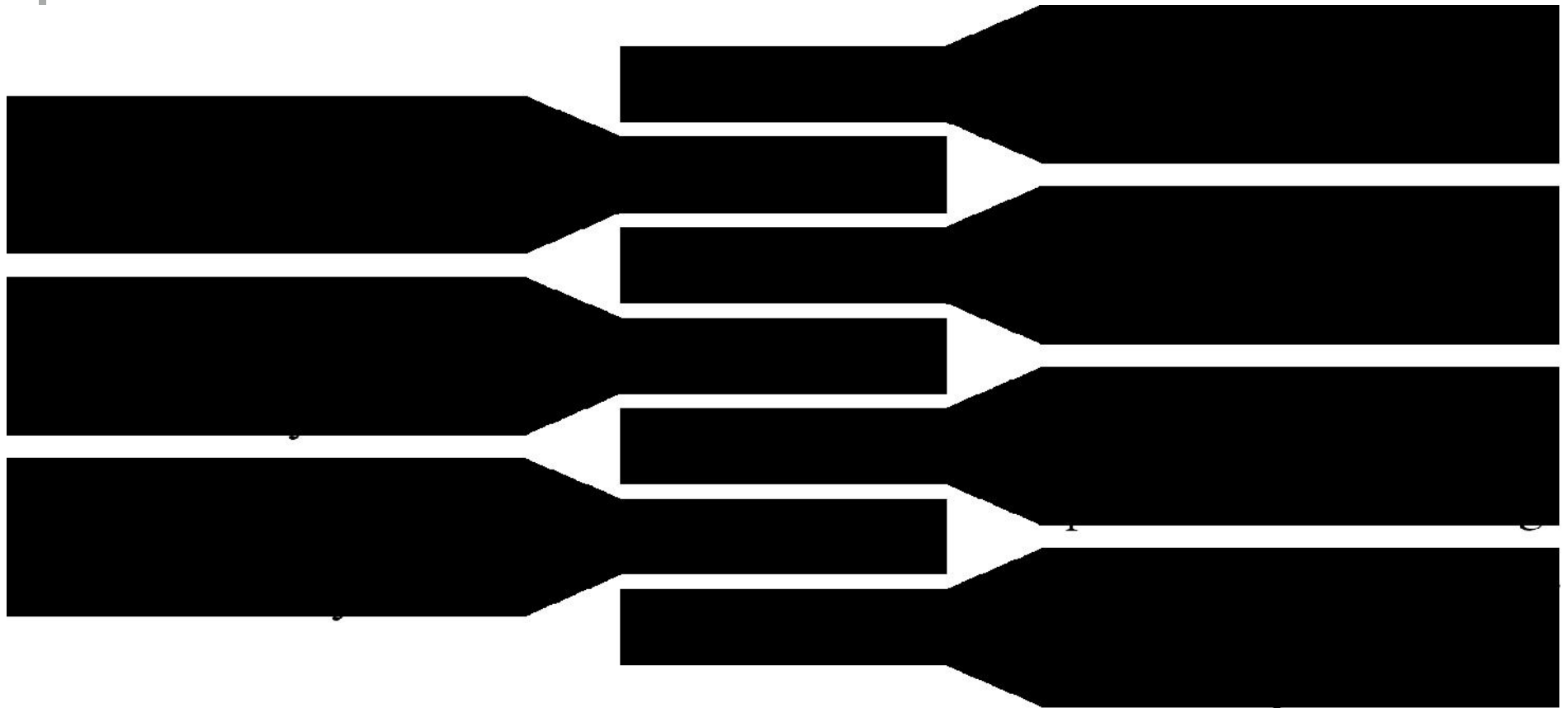


To presentation layer



From presentation layer

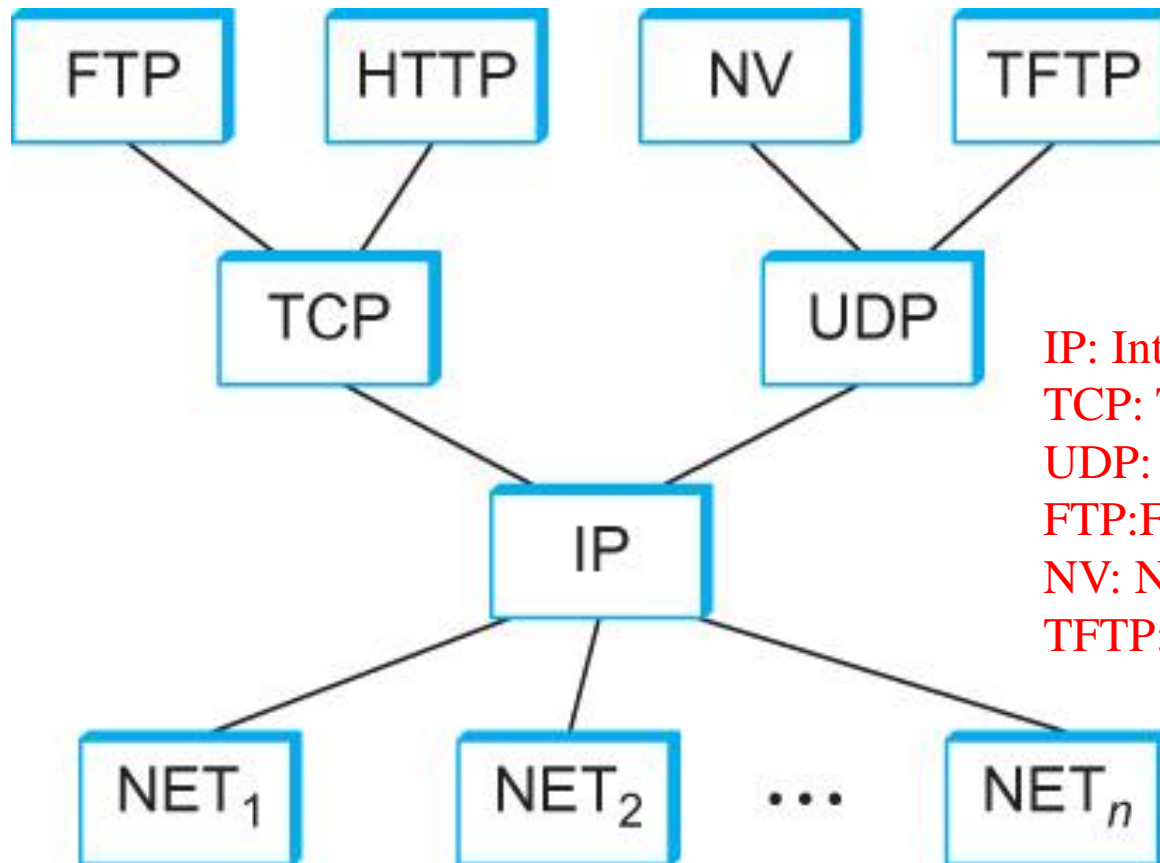
Summary of Layer Functions



Topic 5

Internet Architecture

Internet Architecture



IP: Internet Protocol
TCP: Transmission Control Protocol
UDP: User Datagram Protocol
FTP: File Transfer Protocol
NV: Network Virtualization
TFTP: Trivial File Transfer Protocol

Internet Protocol Graph

IP: The Internet Protocol is the principal communications protocol in the Internet protocol suite for relaying datagrams across network boundaries. Its routing function enables internetworking, and essentially establishes the Internet.

TCP: The Transmission Control Protocol is one of the main protocols of the Internet protocol suite. It originated in the initial network implementation in which it complemented the Internet Protocol. Therefore, the entire suite is commonly referred to as TCP/IP.

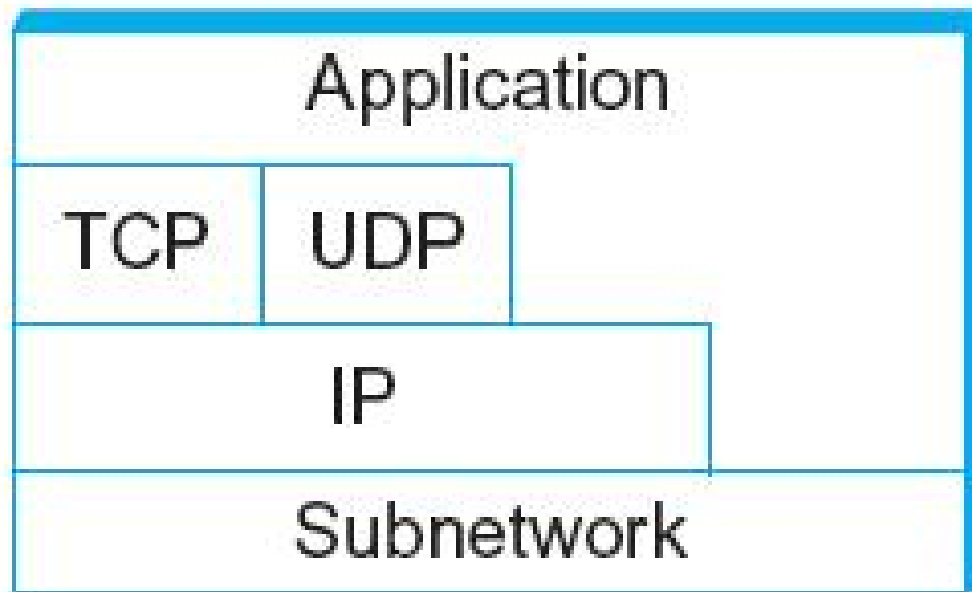
UDP: User Datagram Protocol (UDP) – a communications protocol that facilitates the exchange of messages between computing devices in a network. It's an alternative to the transmission control protocol (TCP).

FTP: The File Transfer Protocol is a standard network protocol used for the transfer of computer files between a client and server on a computer network. FTP is built on a client-server model architecture using separate control and data connections between the client and the server.

HTTP: The Hypertext Transfer Protocol is an application layer protocol for distributed, collaborative, hypermedia information systems.

NV: In computing, network virtualization or network virtualisation is the process of combining hardware and software network resources and network functionality into a single, software-based administrative entity, a virtual network

TFTP: Trivial File Transfer Protocol is a simple lockstep File Transfer Protocol which allows a client to get a file from or put a file onto a remote host. One of its primary uses is in the early stages of nodes booting from a local area network



Alternative view of the Internet architecture.

- The “Network” layer shown here is sometimes referred to as the “sub-network” or “link” layer.

Internet Architecture

- Defined by IETF
- Three main features
 - Does not imply strict layering. The application is free to bypass the defined transport layers and to directly use IP or other underlying networks
 - An hour-glass shape – wide at the top, narrow in the middle and wide at the bottom. IP serves as the focal point for the architecture
 - In order for a new protocol to be officially included in the architecture, there needs to be both a protocol specification and at least one (and preferably two) representative implementations of the specification

Application Programming Interface

Application Programming Interface

- Interface exported by the network
- Since most network protocols are implemented (those in the high protocol stack) in software and nearly all computer systems implement their network protocols as part of the operating system, when we refer to the interface “*exported by the network*”, we are generally referring to the interface that the OS provides to its networking subsystem
- The interface is called the network Application Programming Interface (API)

Application Programming Interface (Sockets)

- Socket Interface was originally provided by the Berkeley distribution of Unix
 - Now supported in virtually all operating systems
- Each protocol provides a certain set of *services*, and the API provides a syntax by which those services can be invoked in this particular OS

Socket

- What is a socket?
 - The point where a local application process attaches to the network
 - An interface between an application and the network
 - An application creates the socket
- The interface defines operations for
 - Creating a socket
 - Attaching a socket to the network
 - Sending and receiving messages through the socket
 - Closing the socket

Socket

- Socket Family
 - PF_INET denotes the Internet family
 - PF_UNIX denotes the Unix pipe facility
 - PF_PACKET denotes direct access to the network interface (i.e., it bypasses the TCP/IP protocol stack)
- Socket Type
 - SOCK_STREAM is used to denote a byte stream
 - SOCK_DGRAM is an alternative that denotes a message oriented service, such as that provided by UDP

Creating a Socket

```
int sockfd = socket(address_family, type, protocol);
```

- The socket number returned is the socket descriptor for the newly created socket

```
■ int sockfd = socket (PF_INET, SOCK_STREAM, 0);
```

```
■ int sockfd = socket (PF_INET, SOCK_DGRAM, 0);
```

The combination of PF_INET and SOCK_STREAM implies TCP

Client-Serve Model with TCP

Server

- Passive open
- Prepares to accept connection, does not actually establish a connection

Server invokes

```
int bind (int socket, struct sockaddr *address,  
          int addr_len)
```

```
int listen (int socket, int backlog)
```

```
int accept (int socket, struct sockaddr *address,  
            int *addr_len)
```

Client-Serve Model with TCP

Bind

- Binds the newly created socket to the specified address i.e. the network address of the local participant (the server)
- Address is a data structure which combines IP and port

Listen

- Defines how many connections can be pending on the specified socket

Client-Serve Model with TCP

Accept

- Carries out the passive open
- Blocking operation
 - Does not return until a remote participant has established a connection
 - When it does, it returns a new socket that corresponds to the new established connection and the address argument contains the remote participant's address

Client-Serve Model with TCP

Client

- Application performs active open
- It says who it wants to communicate with

Client invokes

```
int connect (int socket, struct sockaddr *address,  
            int addr_len)
```

Connect

- Does not return until TCP has successfully established a connection at which application is free to begin sending data
- Address contains remote machine's address

Client-Serve Model with TCP

In practice

- The client usually specifies only remote participant's address and let's the system fill in the local information
- Whereas a server usually listens for messages on a well-known port
- A client does not care which port it uses for itself, the OS simply selects an unused one

Client-Serve Model with TCP

Once a connection is established, the application process invokes two operation

```
int send (int socket, char *msg, int msg_len,  
          int flags)
```

```
int recv (int socket, char *buff, int buff_len,  
          int flags)
```

Example Application: Client

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#define SERVER_PORT 5432
#define MAX_LINE 256

int main(int argc, char * argv[])
{
    FILE *fp;
    struct hostent *hp;
    struct sockaddr_in sin;
    char *host;
    char buf[MAX_LINE];
    int s;
    int len;
    if (argc==2) {
        host = argv[1];
    }
    else {
        fprintf(stderr, "usage: simplex-talk host\n");
        exit(1);
    }
}
```

Example Application: Client

```
/* translate host name into peer's IP address */
hp = gethostbyname(host);
if (!hp) {
    fprintf(stderr, "simplex-talk: unknown host: %s\n", host);
    exit(1);
}
/* build address data structure */
bzero((char *)&sin, sizeof(sin));
sin.sin_family = AF_INET;
bcopy(hp->h_addr, (char *)&sin.sin_addr, hp->h_length);
sin.sin_port = htons(SERVER_PORT);
/* active open */
if ((s = socket(PF_INET, SOCK_STREAM, 0)) < 0) {
    perror("simplex-talk: socket");
    exit(1);
}
if (connect(s, (struct sockaddr *)&sin, sizeof(sin)) < 0) {
    perror("simplex-talk: connect");
    close(s);
    exit(1);
}
/* main loop: get and send lines of text */
while (fgets(buf, sizeof(buf), stdin)) {
    buf[MAX_LINE-1] = '\0';
    len = strlen(buf) + 1;
    send(s, buf, len, 0);
}
```

Example Application: Server

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#define SERVER_PORT 5432
#define MAX_PENDING 5
#define MAX_LINE 256

int main()
{
    struct sockaddr_in sin;
    char buf[MAX_LINE];
    int len;
    int s, new_s;
    /* build address data structure */
    bzero((char *)&sin, sizeof(sin));
    sin.sin_family = AF_INET;
    sin.sin_addr.s_addr = INADDR_ANY;
    sin.sin_port = htons(SERVER_PORT);

    /* setup passive open */
    if ((s = socket(PF_INET, SOCK_STREAM, 0)) < 0) {
        perror("simplex-talk: socket");
        exit(1);
    }
}
```

Example Application: Server

```
if ((bind(s, (struct sockaddr *)&sin, sizeof(sin))) < 0) {
    perror("simplex-talk: bind");
    exit(1);
}
listen(s, MAX_PENDING);
/* wait for connection, then receive and print text */
while(1) {
    if ((new_s = accept(s, (struct sockaddr *)&sin, &len)) < 0) {
        perror("simplex-talk: accept");
        exit(1);
    }
    while (len = recv(new_s, buf, sizeof(buf), 0))
        fputs(buf, stdout);
    close(new_s);
}
}
```

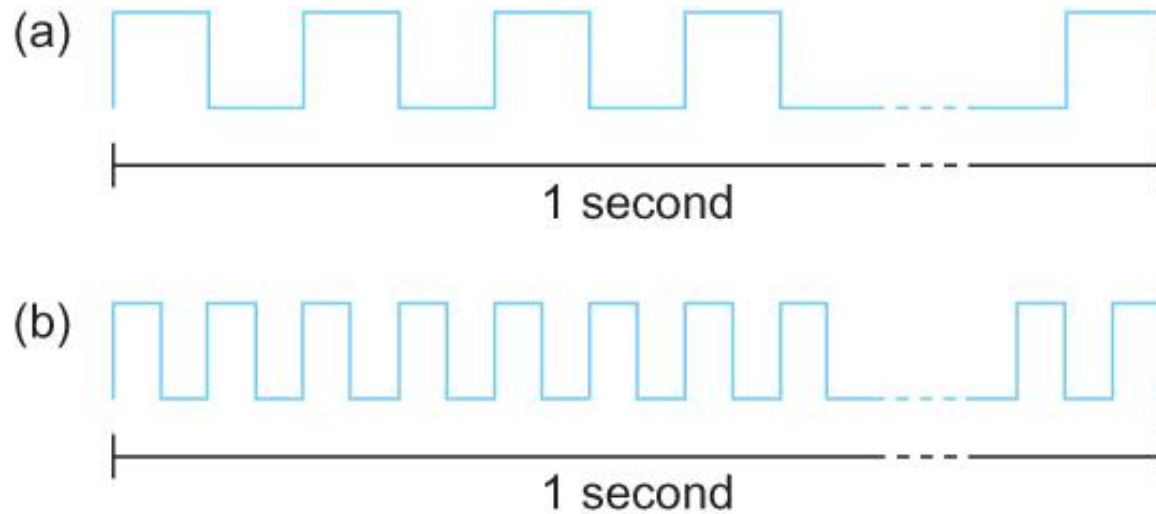
Topic 6

Performance

Performance

- Bandwidth
 - Width of the frequency band
 - Number of bits per second that can be transmitted over a communication link
- 1 Mbps: 1×10^6 bits/second = 1×2^{20} bits/sec
- 1×10^{-6} seconds to transmit each bit or imagine that a timeline, now each bit occupies 1 micro second space.
- On a 2 Mbps link the width is 0.5 micro second.
- Smaller the width more will be transmission per unit time.

Bandwidth



Bits transmitted at a particular bandwidth can be regarded as having some width:

- (a) bits transmitted at 1Mbps (each bit 1 μ s wide);
- (b) bits transmitted at 2Mbps (each bit 0.5 μ s wide).

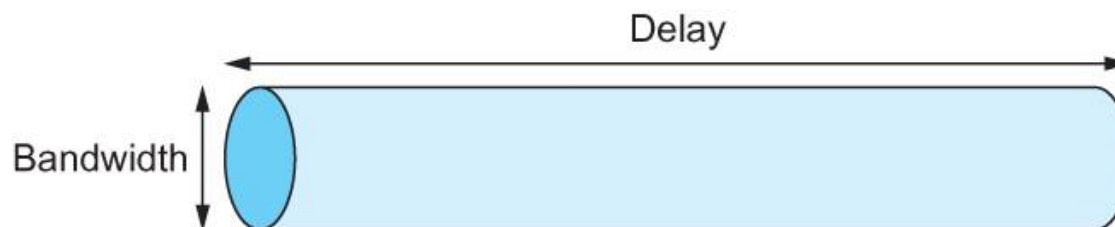
Performance

- Latency = Propagation + transmit + queue
- Propagation = distance/speed of light
- Transmit = size/bandwidth

- One bit transmission => propagation is important
- Large bytes transmission => bandwidth is important

Delay X Bandwidth

- We think the channel between a pair of processes as a hollow pipe
- Latency (delay) length of the pipe and bandwidth the width of the pipe
- Delay of 50 ms and bandwidth of 45 Mbps
 - ⇒ 50×10^{-3} seconds \times 45×10^6 bits/second
 - ⇒ 2.25×10^6 bits = 280 KB data.



Network as a pipe

Delay X Bandwidth

- Relative importance of bandwidth and latency depends on application
 - For large file transfer, bandwidth is critical
 - For small messages (HTTP, NFS, etc.), latency is critical
 - Variance in latency (jitter) can also affect some applications (e.g., audio/video conferencing)

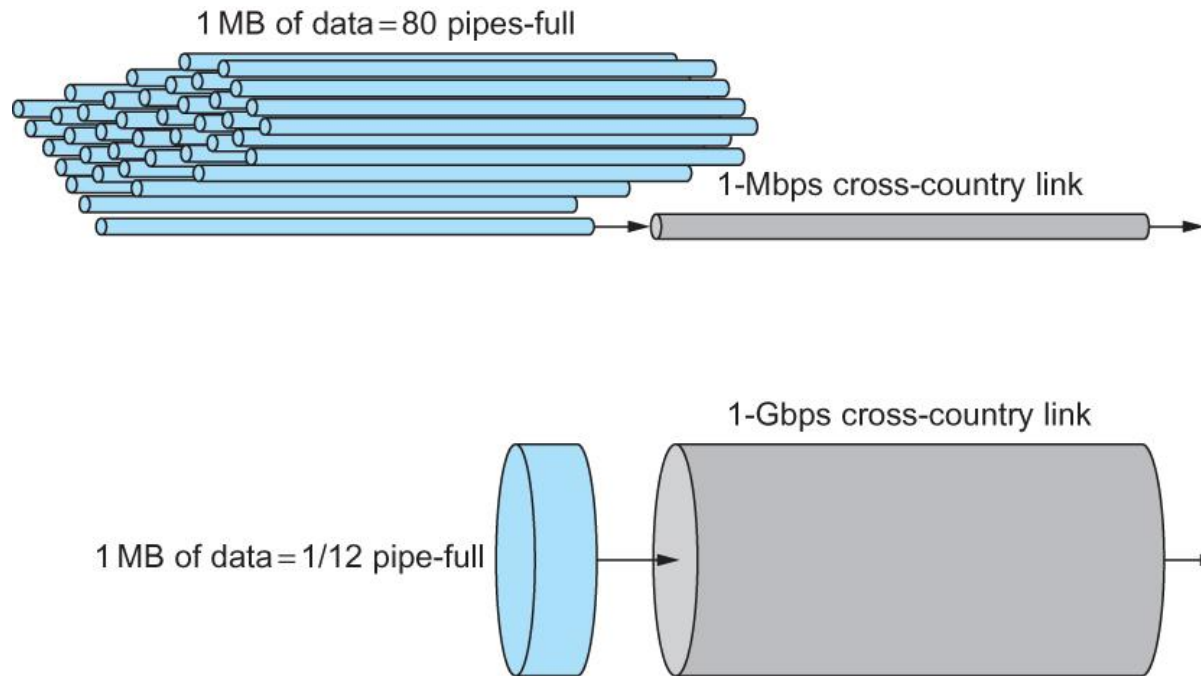
Delay X Bandwidth

- How many bits the sender must transmit before the first bit arrives at the receiver if the sender keeps the pipe full
- Takes another one-way latency to receive a response from the receiver
- If the sender does not fill the pipe—send a whole delay \times bandwidth product's worth of data before it stops to wait for a signal—the sender will not fully utilize the network

Delay X Bandwidth

- Infinite bandwidth
 - RTT dominates
 - $\text{Throughput} = \text{TransferSize} / \text{TransferTime}$
 - $\text{TransferTime} = \text{RTT} + 1/\text{Bandwidth} \times \text{TransferSize}$
- Its all relative
 - 1-MB file to 1-Gbps link looks like a 1-KB packet to 1-Mbps link

Relationship between bandwidth and latency



A 1-MB file would fill the 1-Mbps link 80 times,
but only fill the 1-Gbps link 1/12 of one time

Summary

- We have identified what we expect from a computer network
- We have defined a layered architecture for computer network that will serve as a blueprint for our design
- We have discussed the socket interface which will be used by applications for invoking the services of the network subsystem
- We have discussed two performance metrics using which we can analyze the performance of computer networks

COMPUTER NETWORKS – Unit 1

Topic 7

Network Topology

Figure 2-4

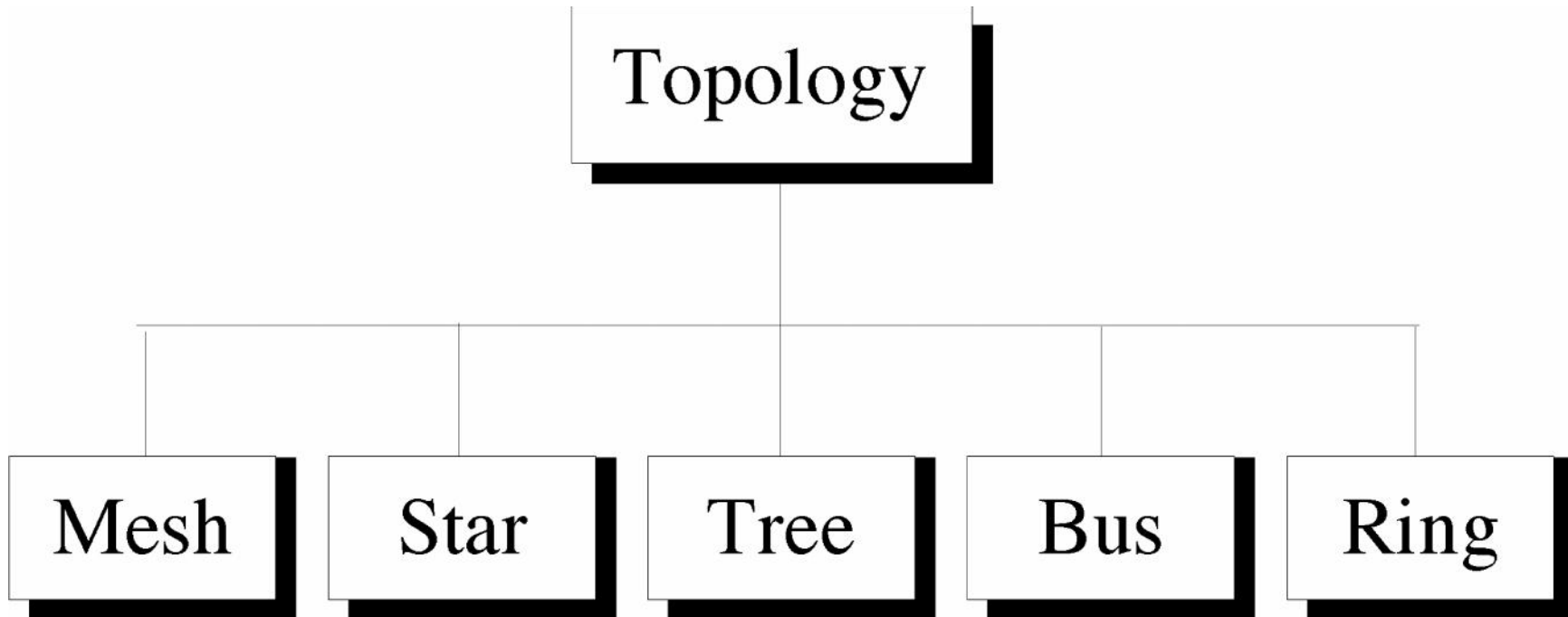


Figure 2-5

Mesh Topology

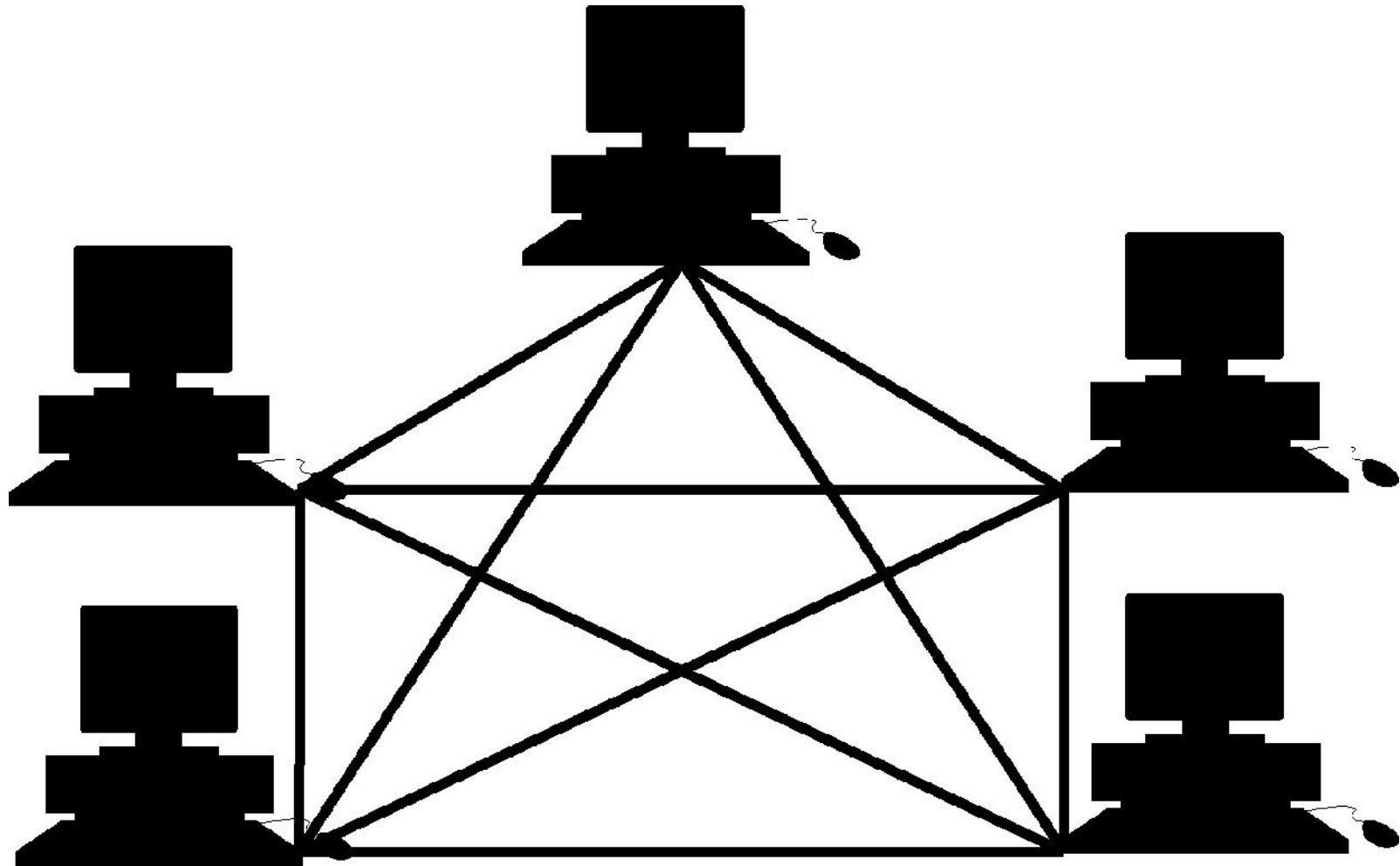


Figure 2-6

Star Topology

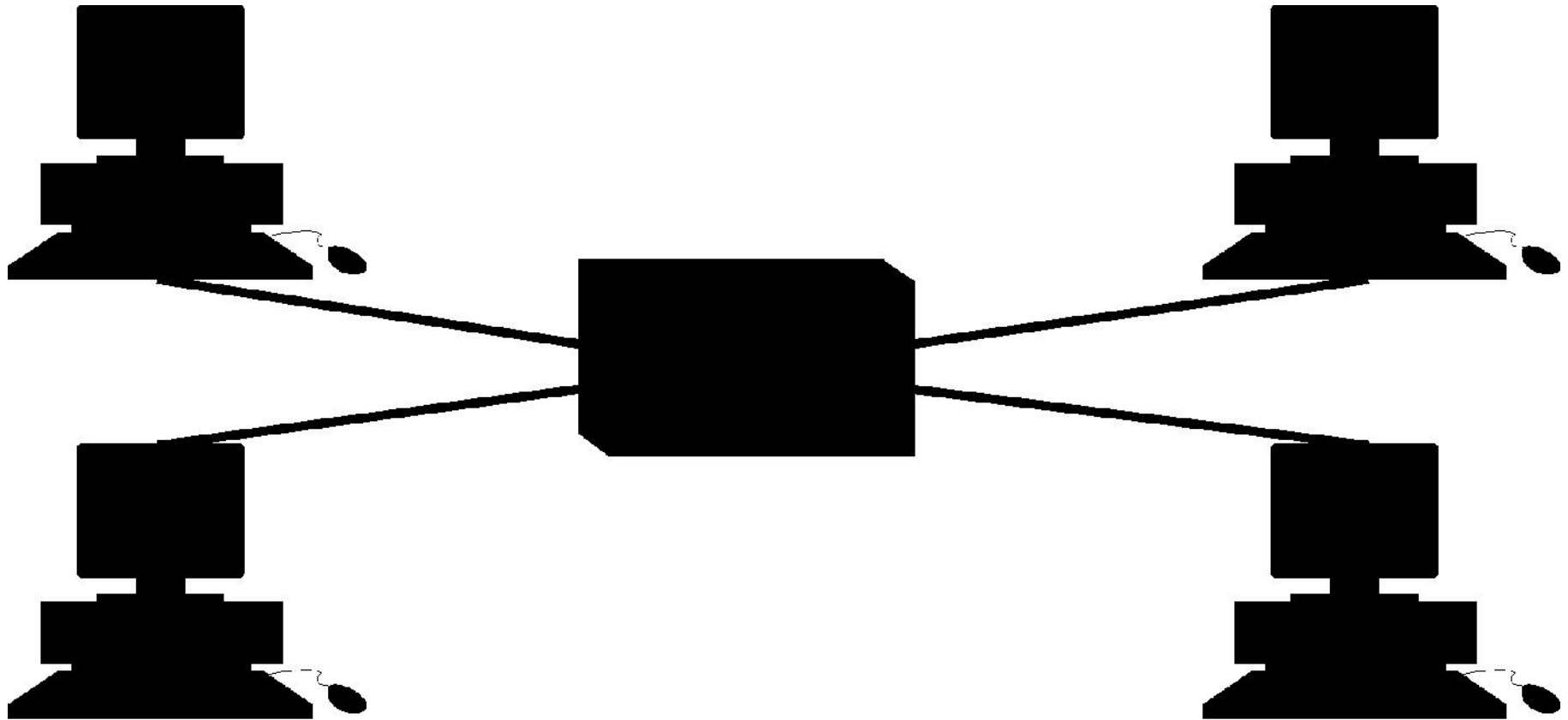


Figure 2-7

Tree Topology

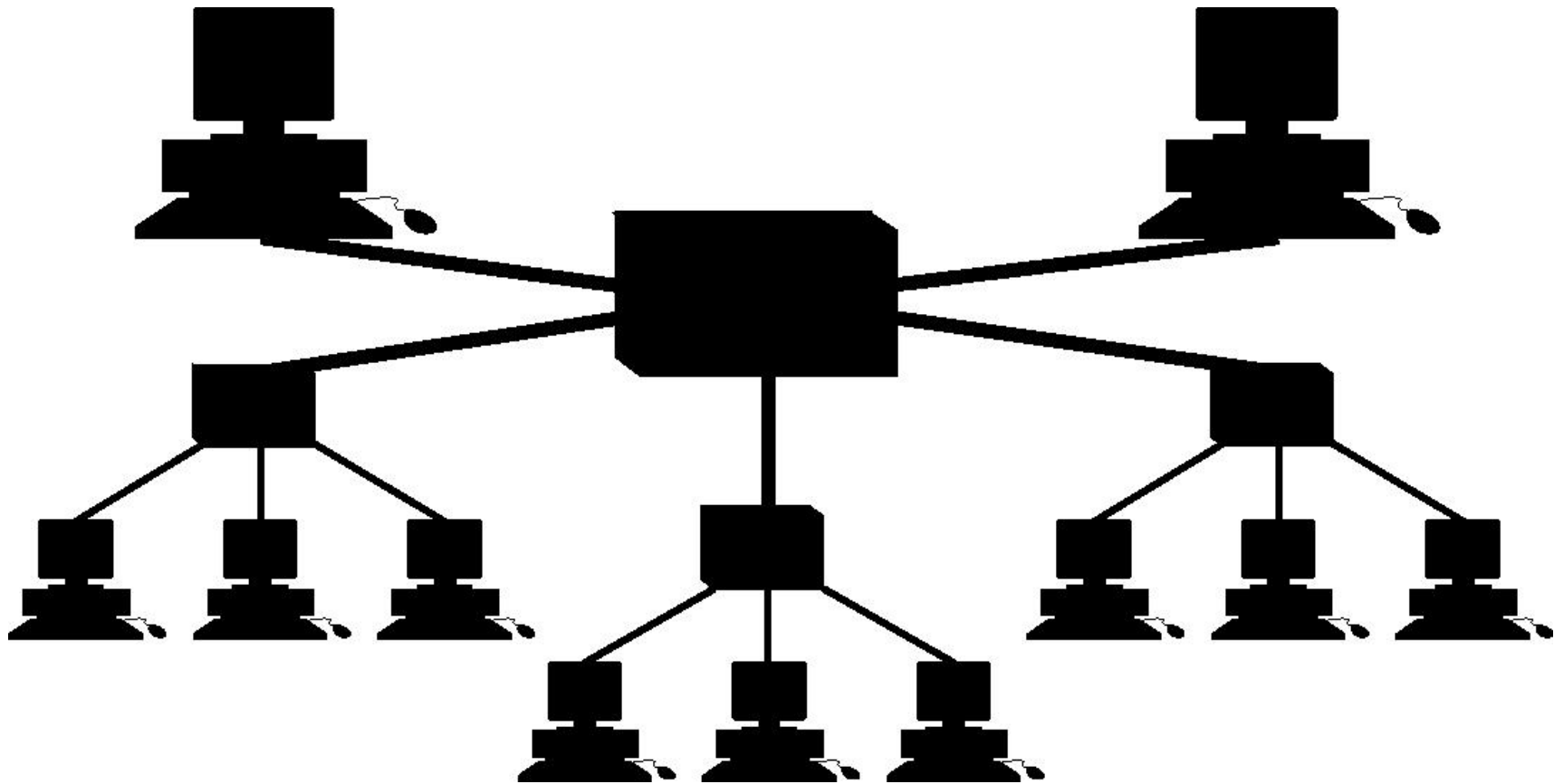


Figure 2-8

Bus Topology

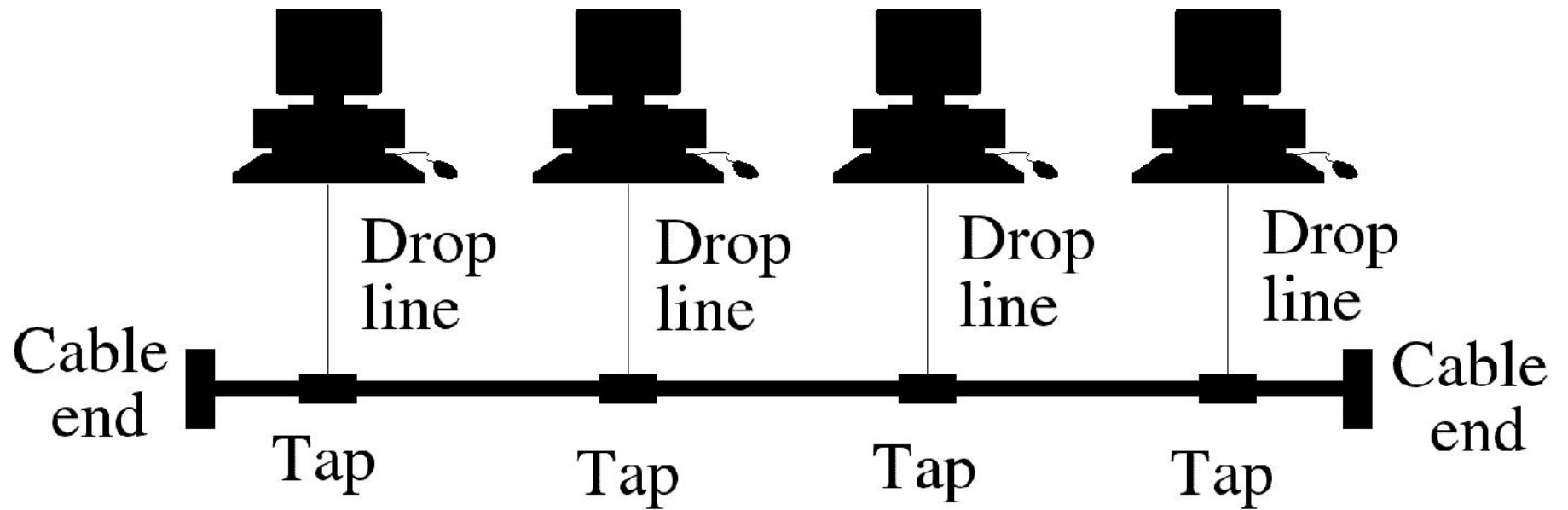


Figure 2-9

Ring Topology

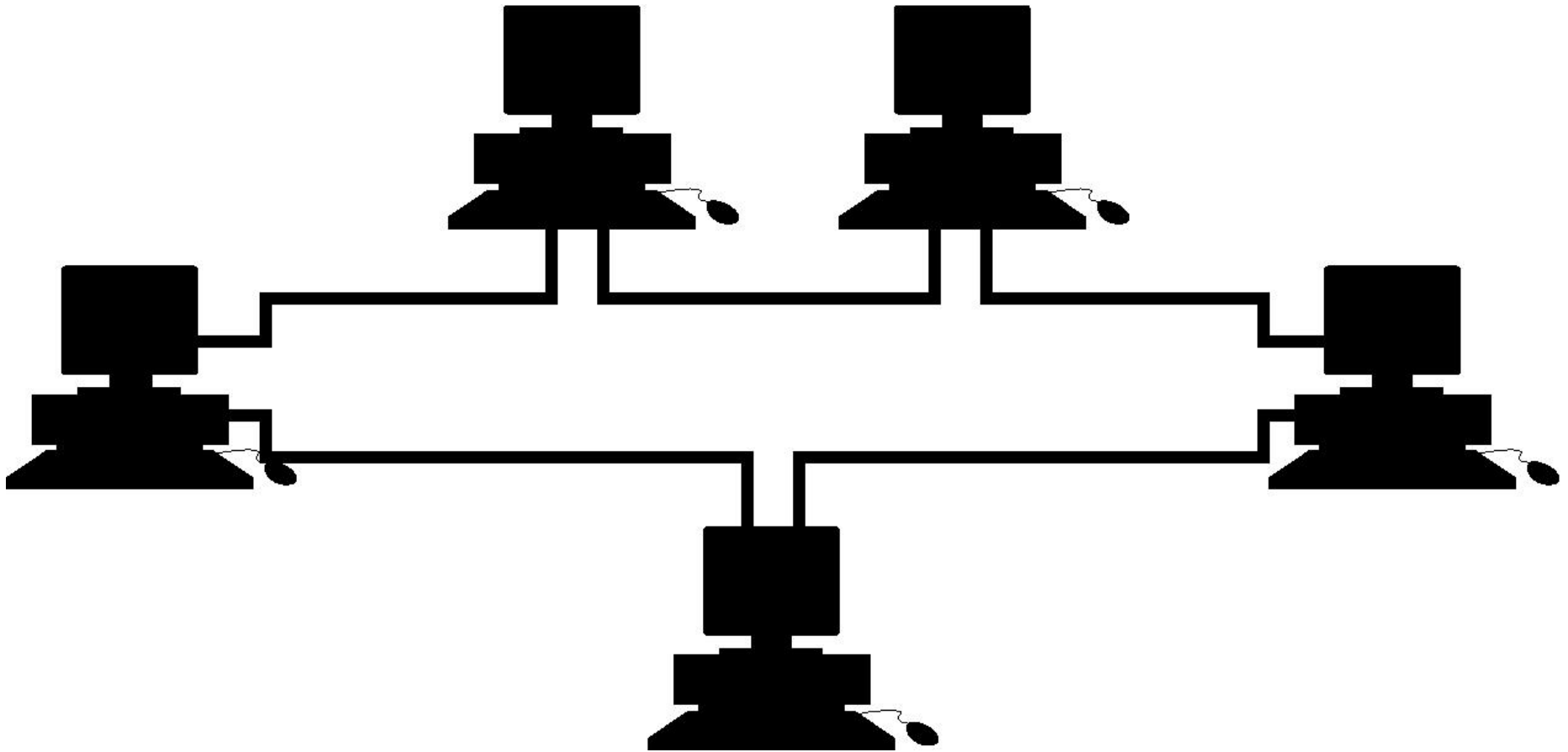
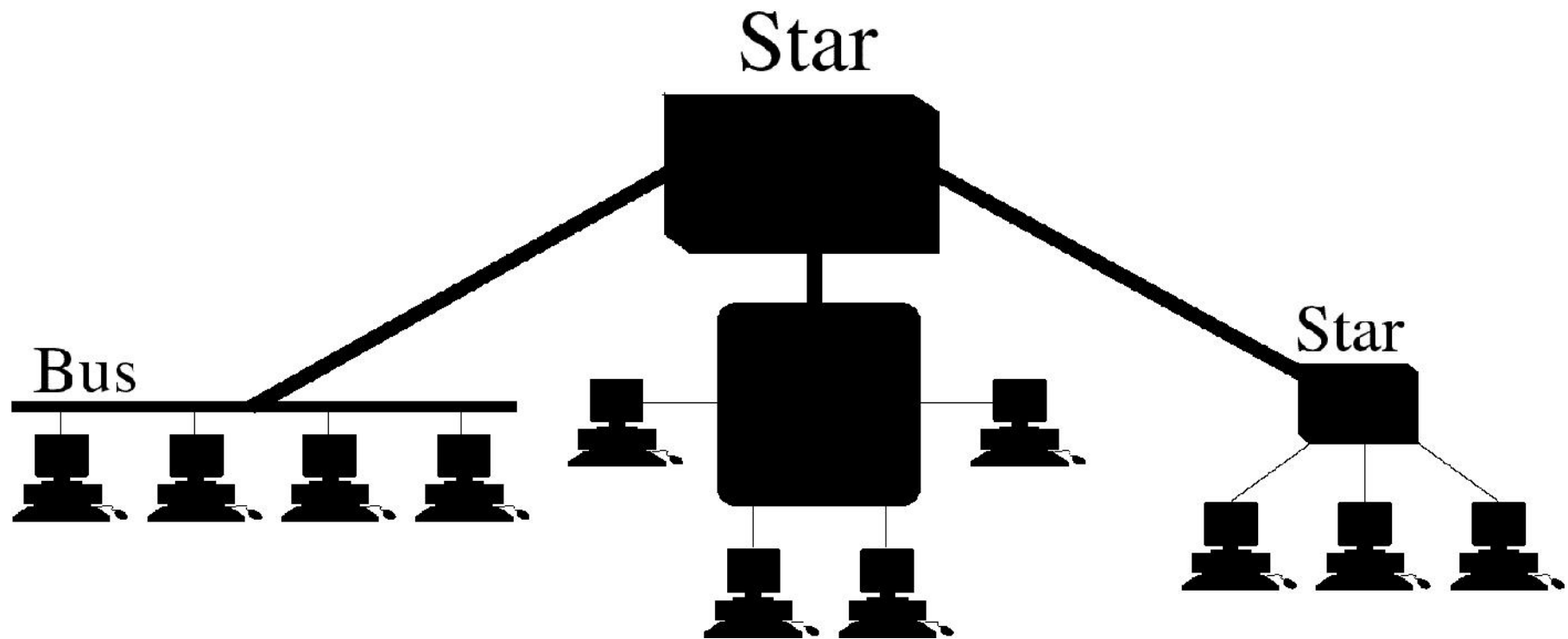


Figure 2-10

Hybrid Topology



COMPUTER NETWORKS – Unit 1

Topic 8

Physical layer: Data and Signals

- Analog and digital
- Periodic analog signals
- Digital signals
- Transmission impairment
- Data rate limits
- Performance

Data and Signals

- Data are entities that convey meaning (computer file, music on CD, results from a blood gas analysis machine)
- Signals are the electric or electromagnetic encoding of data (telephone conversation, web page download)
- Computer networks and data/voice communication systems transmit signals
- Data and signals can be analog or digital – this can be misleading – let's talk more about this

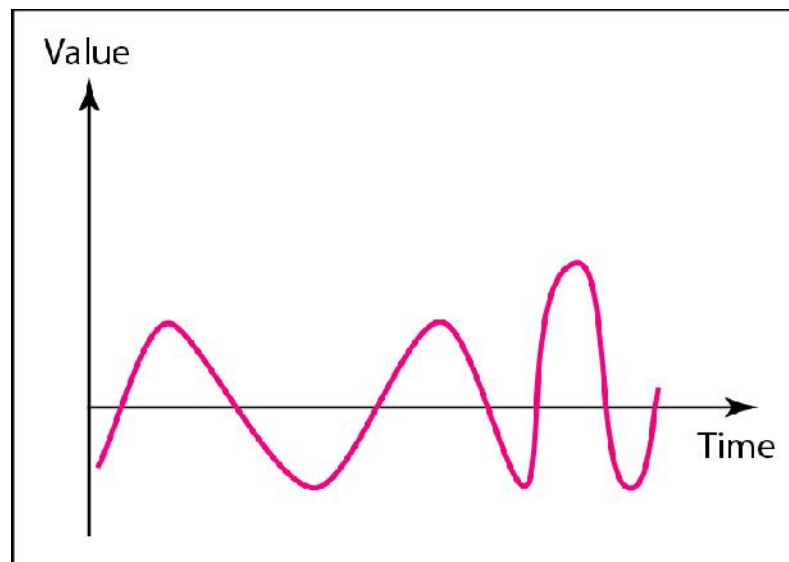
Analog vs. Digital Signals

- Signals can be interpreted as either analog or digital
- In reality, all signals are analog
- Analog signals are continuous, non-discrete
- Digital signals are non-continuous, discrete
- Digital signals lend themselves more nicely to noise reduction techniques

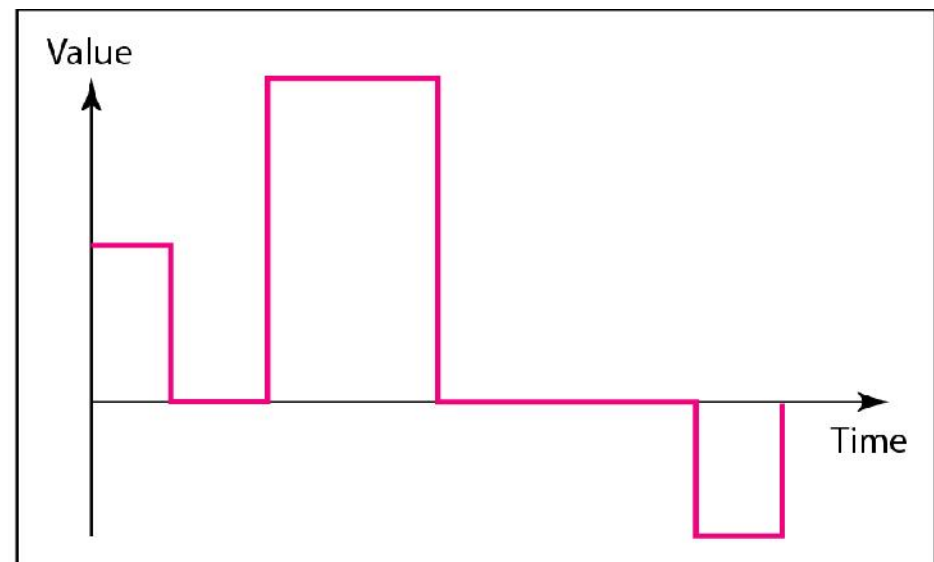
Analog vs. Digital Signals

- There are actually multiple “kinds” of digital signals
- Discrete square waveforms found in digital systems such as LANs (see an example on the next slide)
- Digital logic voltage levels (a binary 0 is 0 to 2 volts; a binary 1 is 4 to 6 volts)
- Analog signals which can only be interpreted in a finite number of ways (modulation techniques such as QAM-16)

Figure 3.1 *Comparison of analog and digital signals*



a. Analog signal



b. Digital signal

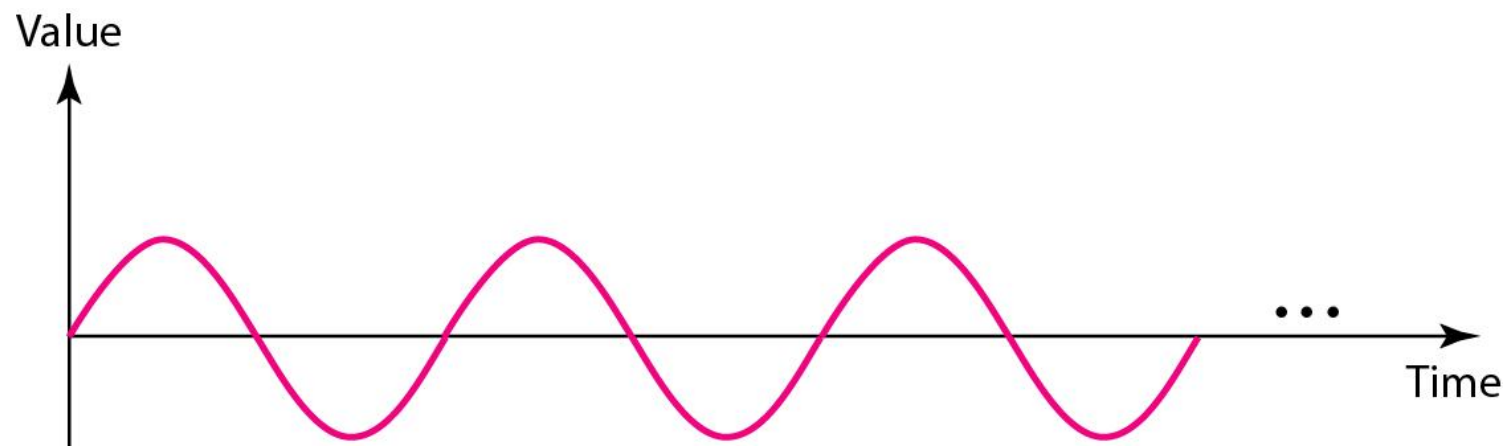
Time domain concepts

- Continuous signal
 - Infinite number of points at any given time
- Discrete signal
 - Finite number of points at any given time; maintains a constant level then changes to another constant level
- Periodic signal
 - Pattern repeated over time
- Aperiodic (non-periodic) signal
 - Pattern not repeated over time

Time domain concepts

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

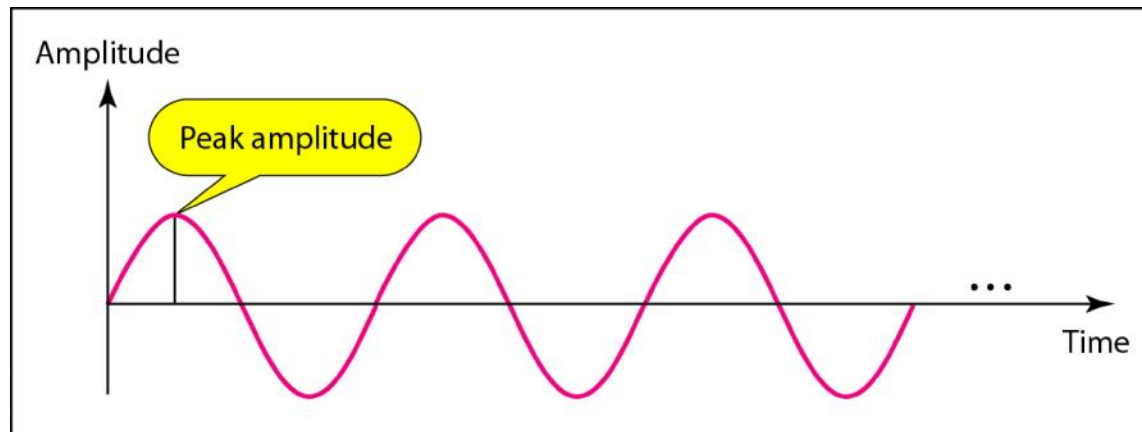
Figure 3.2 *A sine wave*



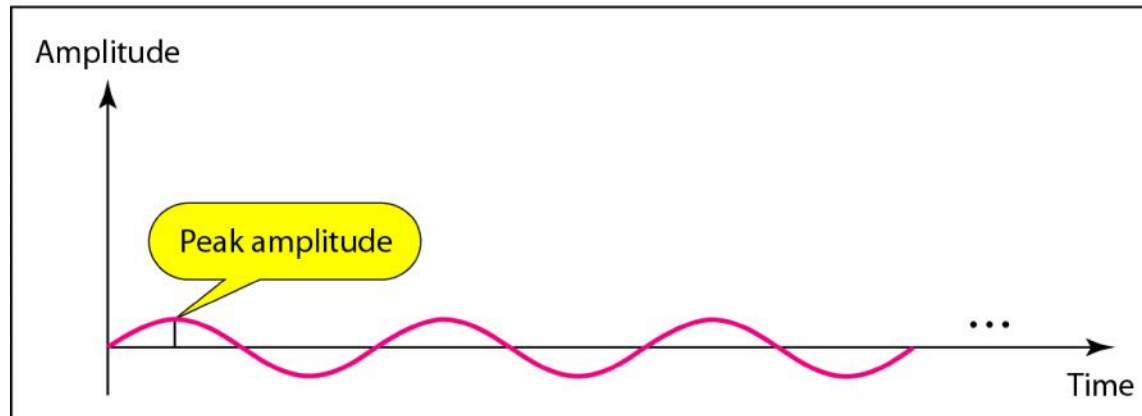
Signal Properties

- All signals are composed of three properties:
 - Amplitude
 - Frequency
 - Phase

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

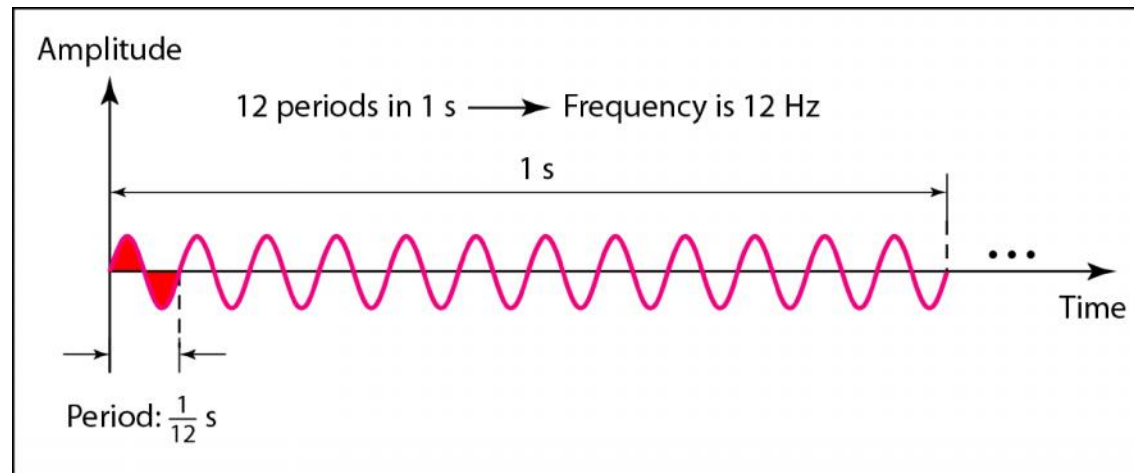


Note

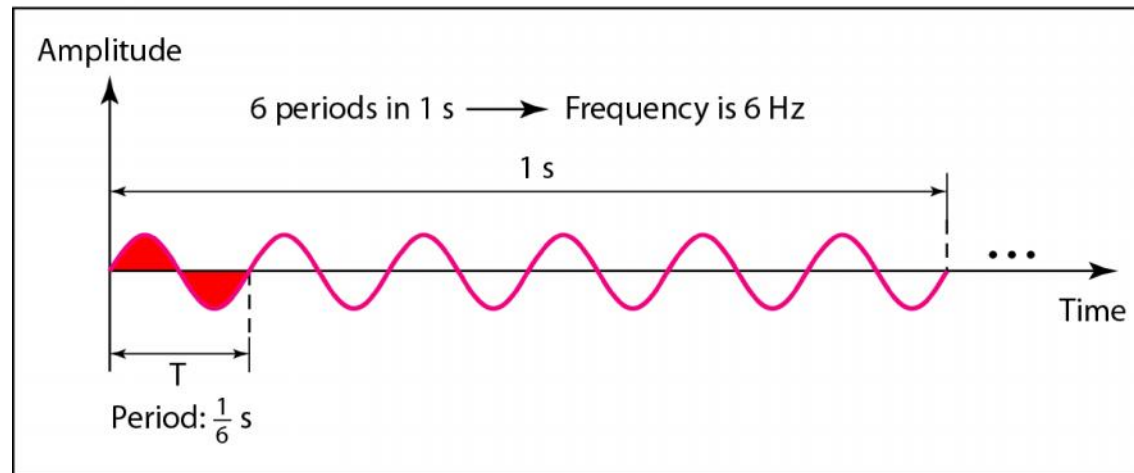
Frequency and period are the inverse of each other.

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

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

Table 3.1 *Units of period and frequency*

<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 (μ 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



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:*

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



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 rate of change with respect to time.

Change in a short span of time means high frequency.

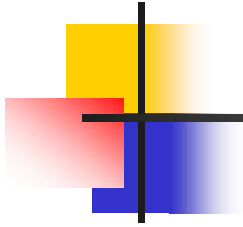
Change over a long span of time means low frequency.



Note

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

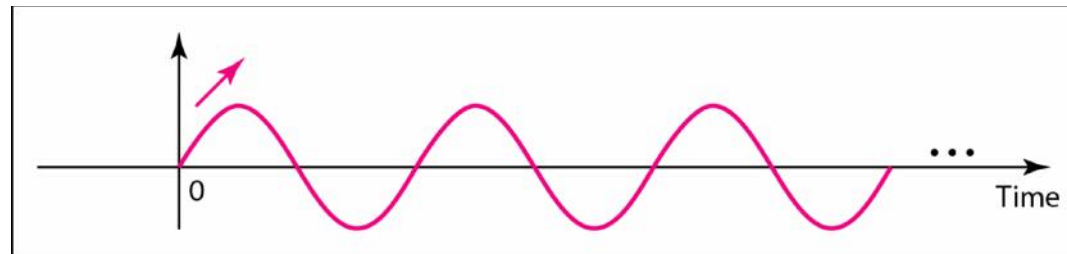
If a signal changes instantaneously, its frequency is infinite.



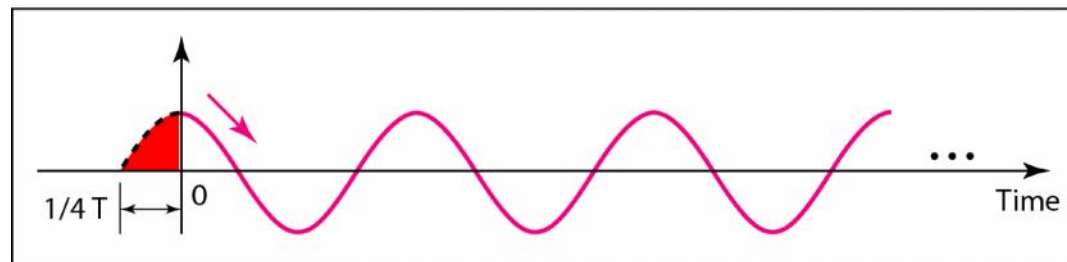
Note

Phase describes the position of the waveform relative to time 0.

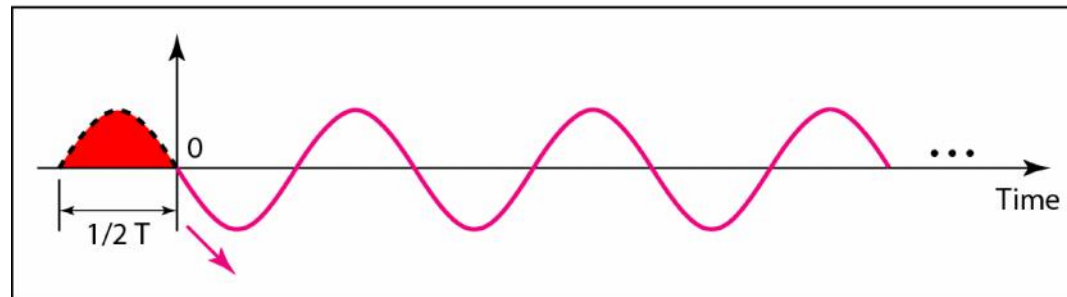
Figure 3.5 *Three sine waves with the same amplitude and frequency, but different phases*



a. 0 degrees



b. 90 degrees



c. 180 degrees



Example 3.6

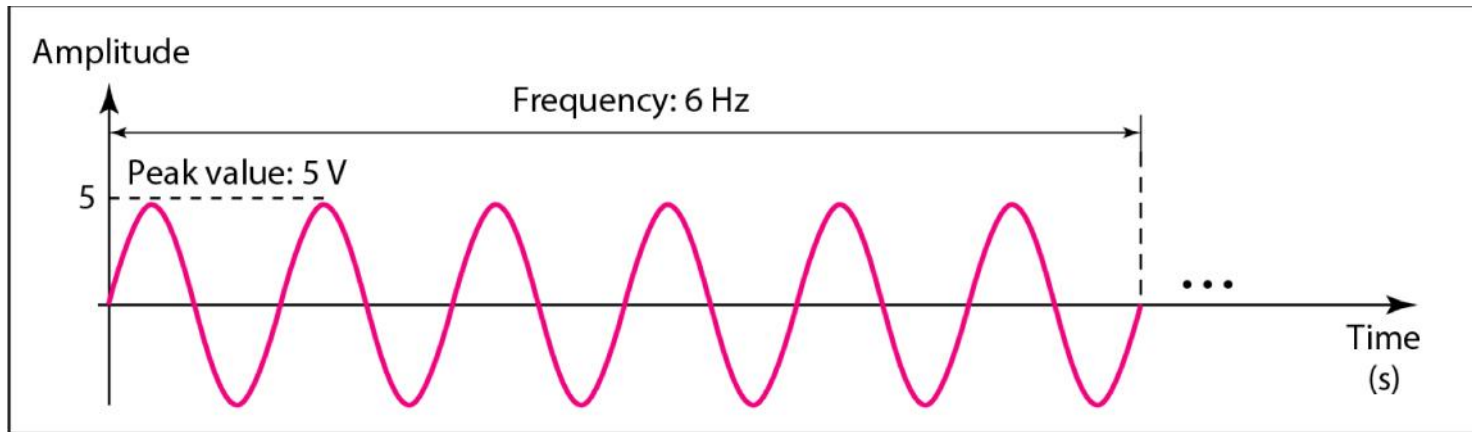
*A sine wave is offset 1/6 cycle with respect to time 0.
What is its phase in degrees and radians?*

Solution

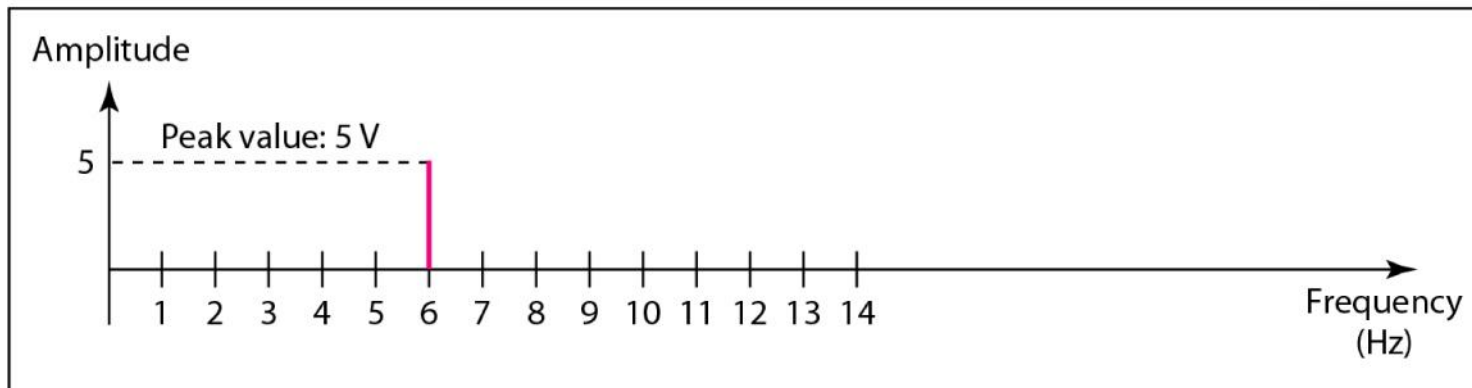
We know that 1 complete cycle is 360°. Therefore, 1/6 cycle is

$$\frac{1}{6} \times 360 = 60^\circ = 60 \times \frac{2\pi}{360} \text{ rad} = \frac{\pi}{3} \text{ rad} = 1.046 \text{ rad}$$

Figure 3.7 *The time-domain and frequency-domain plots of a sine wave*



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



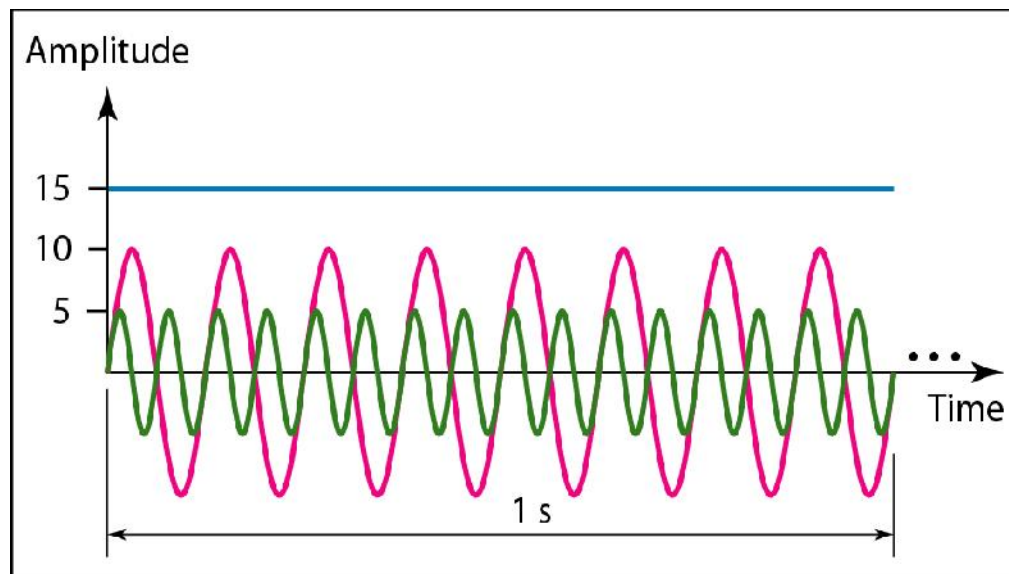
b. The same sine wave in the frequency domain (peak value: 5 V, frequency: 6 Hz)



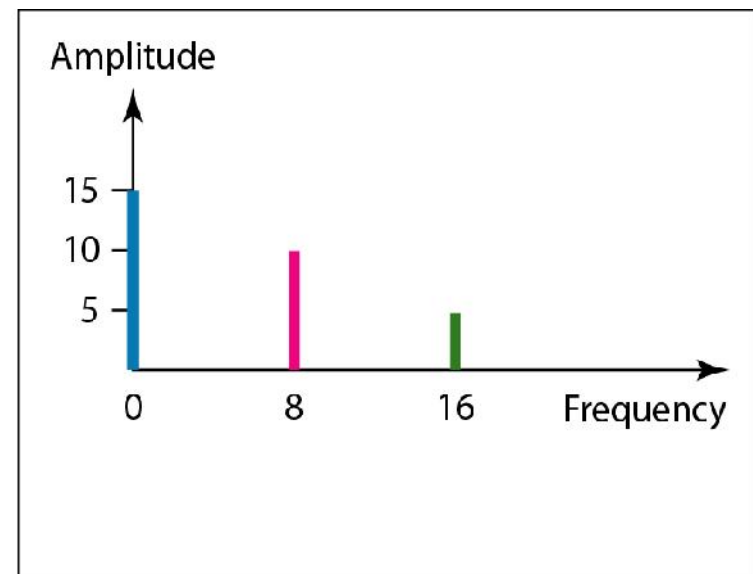
Example 3.7

The frequency domain is more compact and useful when we are dealing with more than one sine wave. For example, Figure 3.8 shows three sine waves, each with different amplitude and frequency. All can be represented by three spikes in the frequency domain.

Figure 3.8 *The time domain and frequency domain of three sine waves*

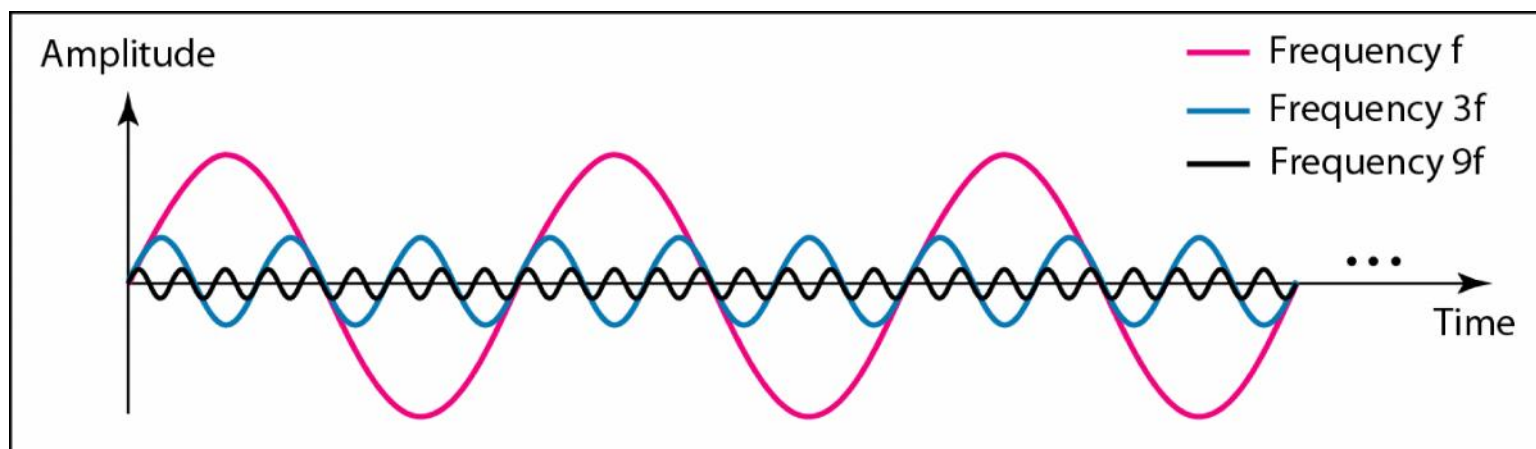


a. Time-domain representation of three sine waves with frequencies 0, 8, and 16

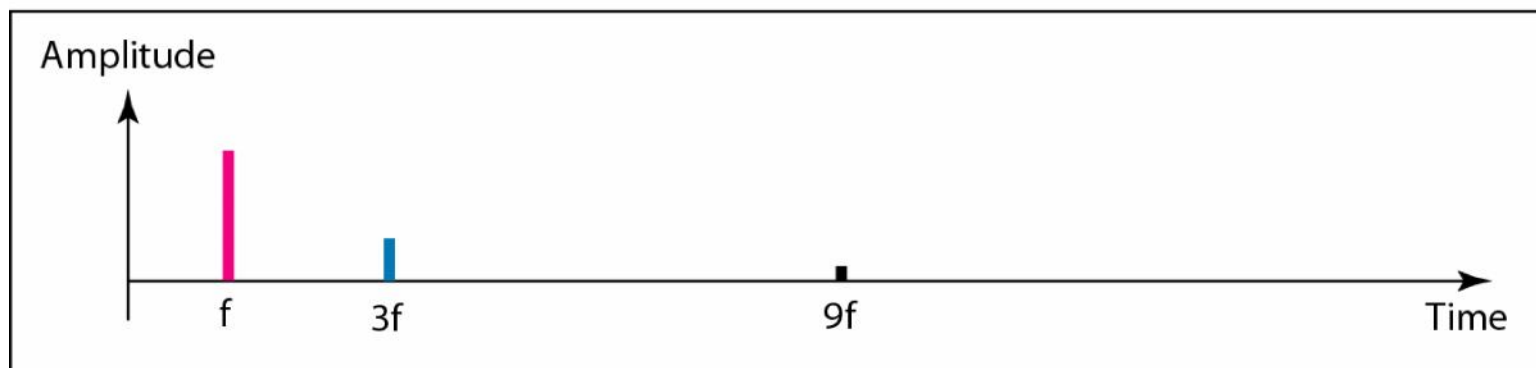


b. Frequency-domain representation of the same three signals

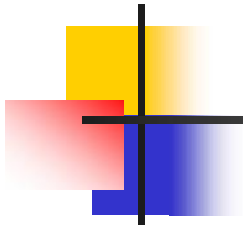
Figure 3.10 *Decomposition of a composite periodic signal in the time and frequency domains*



a. Time-domain decomposition of a composite signal



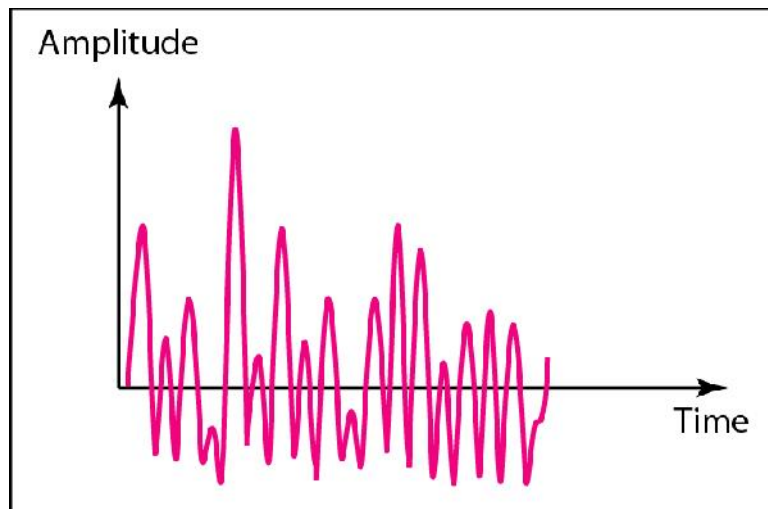
b. Frequency-domain decomposition of the composite signal



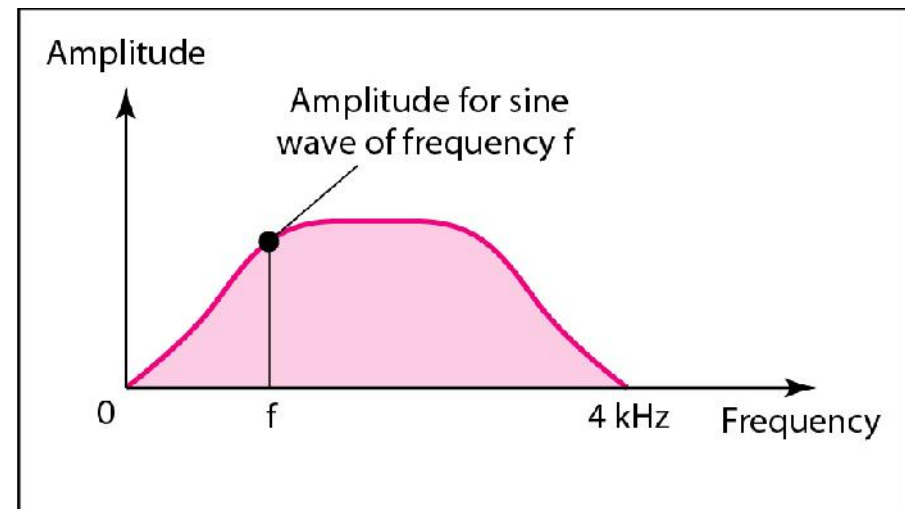
Note

A single-frequency sine wave is not useful in data communications; we need to send a composite signal, a signal made of many simple sine waves.

Figure 3.11 *The time and frequency domains of a nonperiodic signal, such as someone speaking into a microphone*



a. Time domain



b. Frequency domain

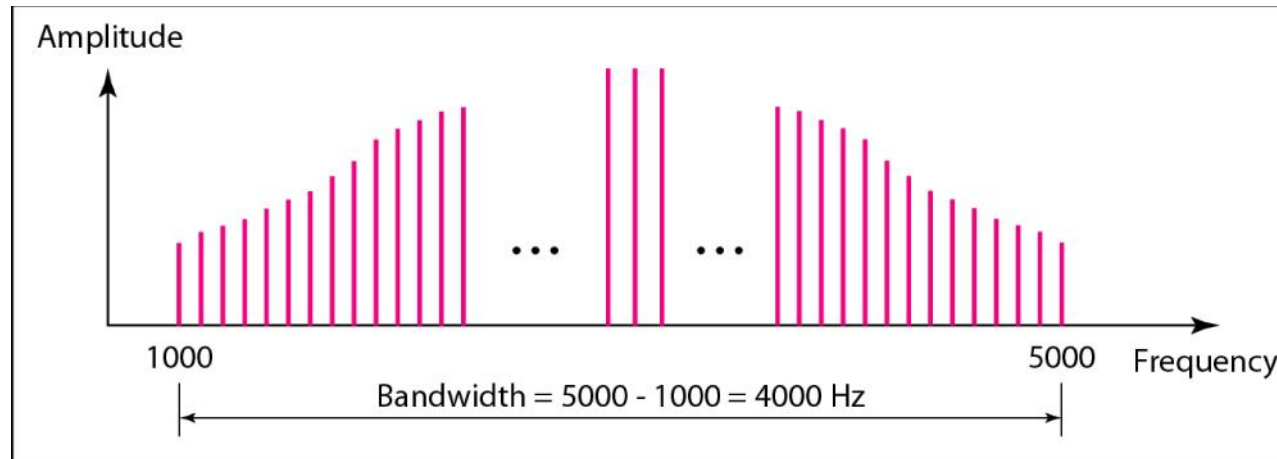


Note

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

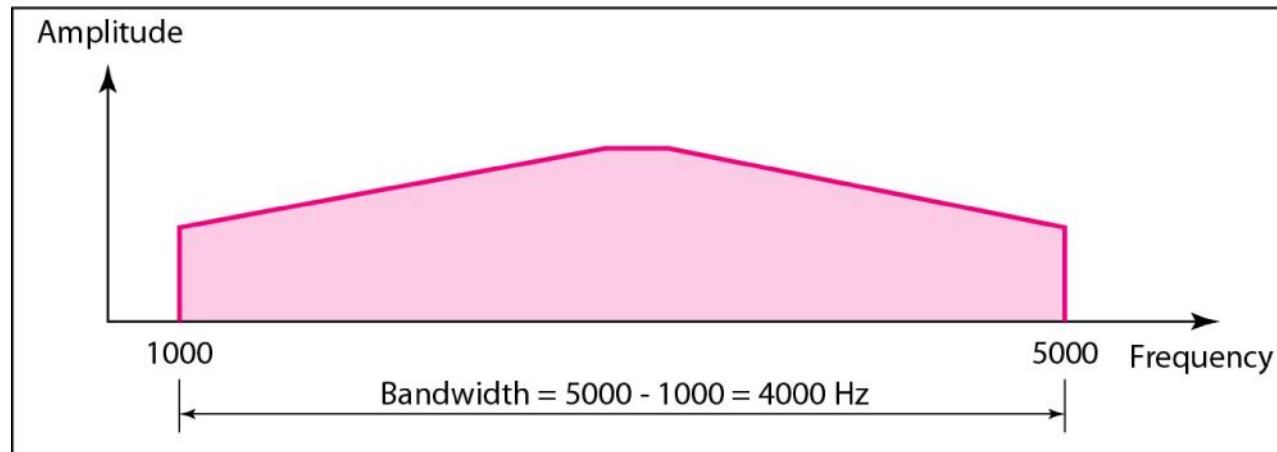
Figure 3.12 *The bandwidth of periodic and nonperiodic composite signals*

Note: each frequency is identifiable



a. Bandwidth of a periodic signal

Note: frequencies are all over the place



b. Bandwidth of a nonperiodic signal



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 (range of frequencies), assuming all components have a maximum amplitude of 10 V.

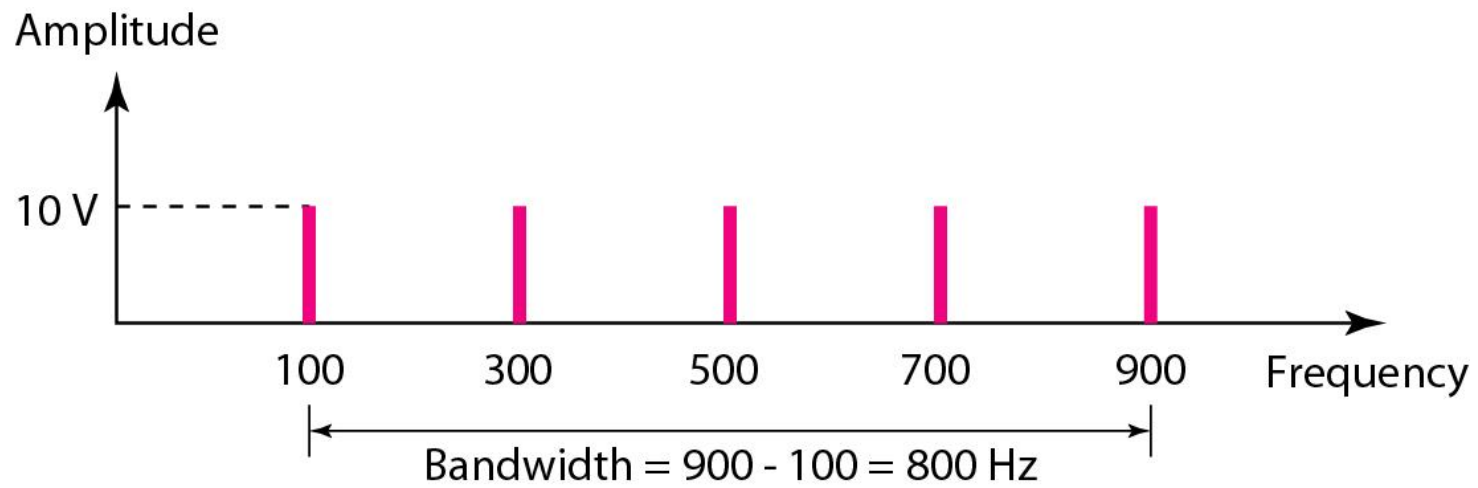
Solution

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*





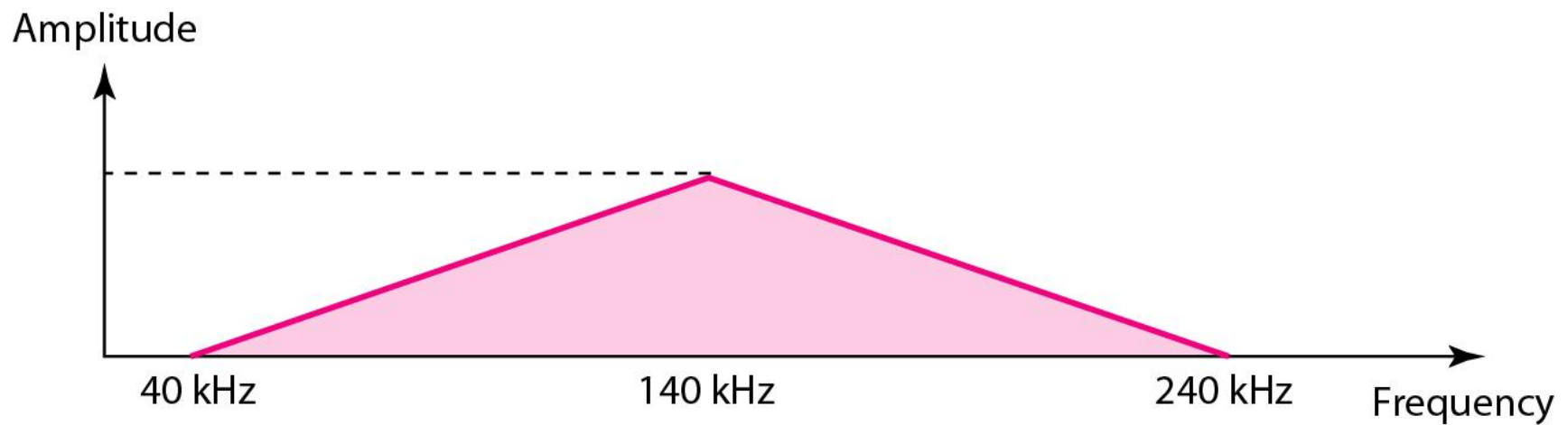
Example 3.12

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. Figure 3.15 shows the frequency domain and the bandwidth.

Figure 3.15 *The bandwidth for Example 3.12*



3-3 DIGITAL SIGNALS

*In addition to being represented by an analog signal, information can also be represented by a **digital signal**. For example, a 1 can be encoded as a positive voltage and a 0 as zero voltage. A digital signal can have more than two levels. In this case, we can send more than 1 bit for each level.*

Topics discussed in this section:

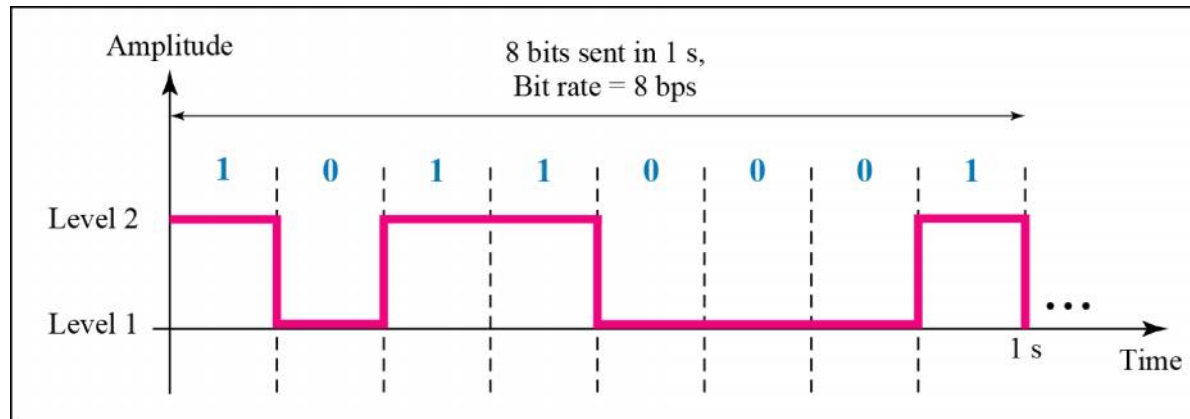
Bit Rate

Bit Length

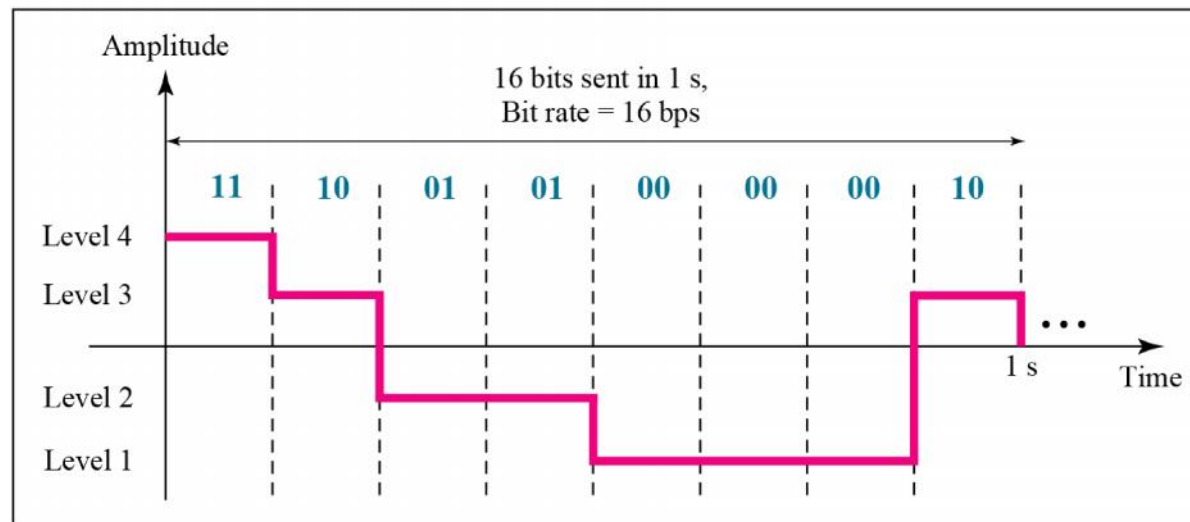
Digital Signal as a Composite Analog Signal

Application Layer

Figure 3.16 *Two digital signals: one with two signal levels and the other with four signal levels*



a. A digital signal with two levels



b. A digital signal with four levels



Example 3.16

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.



Example 3.17

What about a digital signal with 16 levels?

How many bits are needed per level?

What about 32 levels? 64 levels? 128 levels?

What about 9 levels??

$2^? = 9?$

3.17 bits

However, this answer is not realistic.



Example 3.18

Assume we need to download text documents at the rate of 100 pages per minute. 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

$$100 \times 24 \times 80 \times 8 = 1,636,000 \text{ bps} = 1.636 \text{ Mbps}$$



Example 3.19

A digitized voice channel, as we will see in Chapter 4, is made by digitizing a 4-kHz bandwidth analog voice signal. We need to sample the signal at twice the highest frequency (two samples per hertz). We assume that each sample requires 8 bits. What is the required bit rate?

Solution

The bit rate can be calculated as

$$2 \times 4000 \times 8 = 64,000 \text{ bps} = 64 \text{ kbps}$$



Example 3.20

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

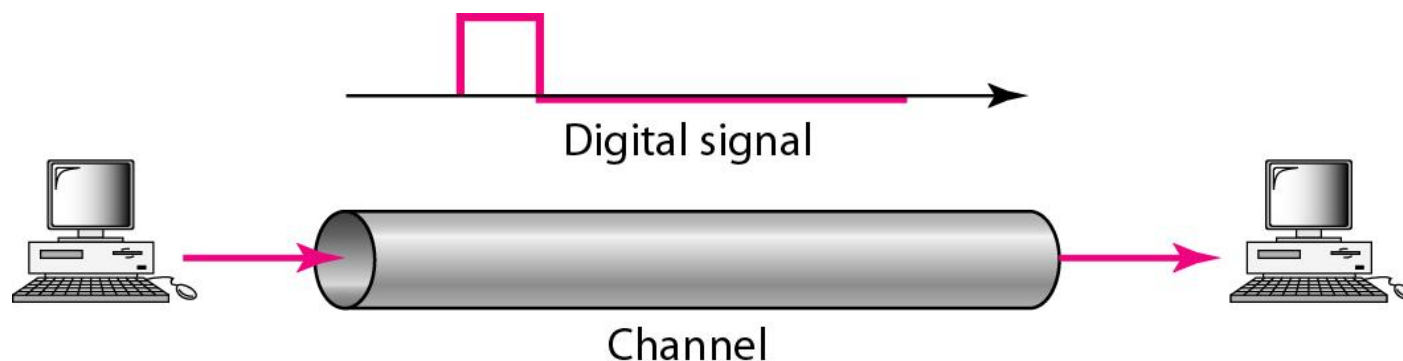
Solution

HDTV uses digital signals to broadcast high quality video signals. The HDTV screen is normally a ratio of 16 : 9. There are 1920 by 1080 pixels per screen, and the screen is renewed 30 times per second. Twenty-four 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.

Figure 3.18 *Baseband transmission*

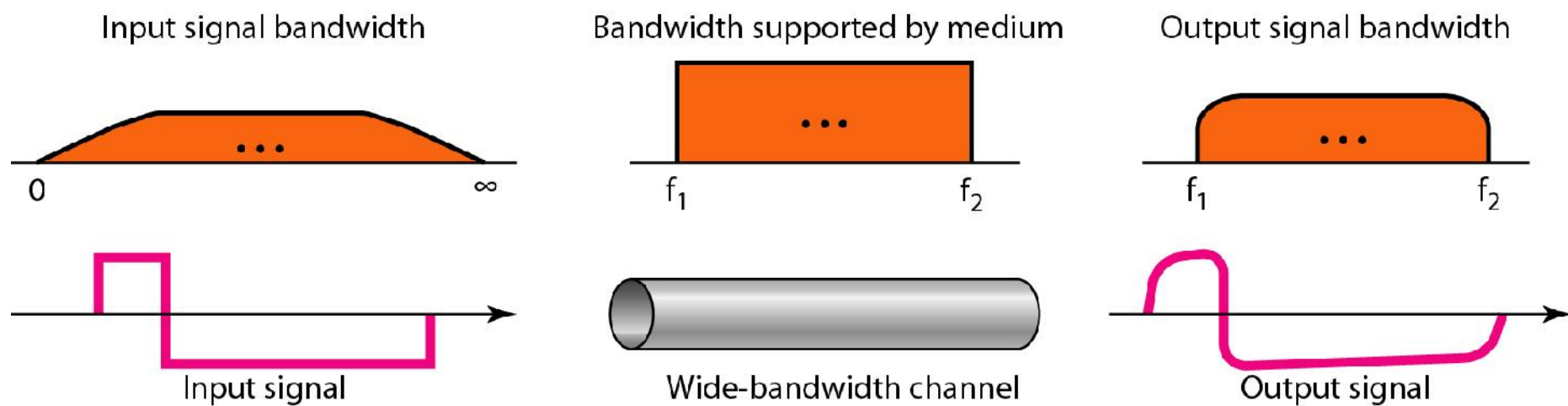


Digital transmission on a wire typically “consumes” the entire channel and is labeled baseband signaling.

A digital signal is a composite analog signal with infinite bandwidth. More on this in a few slides.

The LAN is a common example of baseband signaling.

Figure 3.20 *Baseband transmission using a dedicated medium*



This is ideal.

This is what we get.
Still readable however.

Fourier Analysis

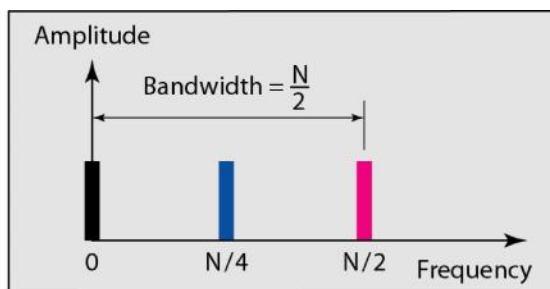
- Interestingly, digital signals are combinations of analog signals
- Fourier Analysis – any composite signal is a combination of simple sine waves with different frequencies and amplitudes
- These simple sine waves are also called harmonics

Fourier Analysis

- Square wave: $(4A/\pi)\sin(2\pi ft) + (4A/3\pi)\sin(2\pi 3ft) + (4A/5\pi)\sin(2\pi 5ft)\dots$
- Sawtooth: $(2A/\pi) \times [\sin(2\pi ft) - (1/2)\sin(2\pi 2ft) + (1/3)\sin(2\pi 3ft) - (1/4)\sin(2\pi 4ft) \dots]$
- www.falstad.com/fourier

Figure 3.21 *Rough approximation of a digital signal using only the first harmonic for worst case.*

$N = \text{Bit Rate}$



With only the first harmonic, you can represent a binary pattern with a frequency of $N/2$.

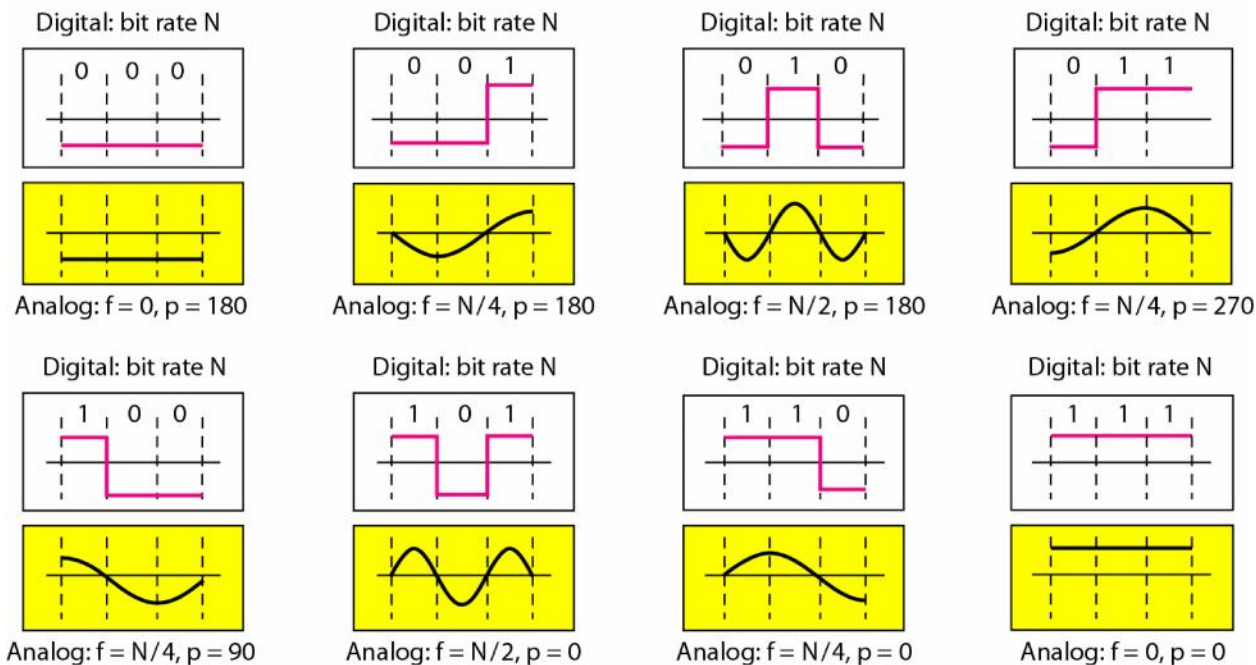
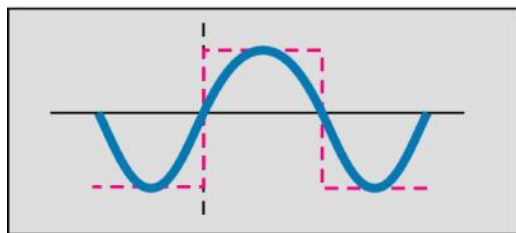
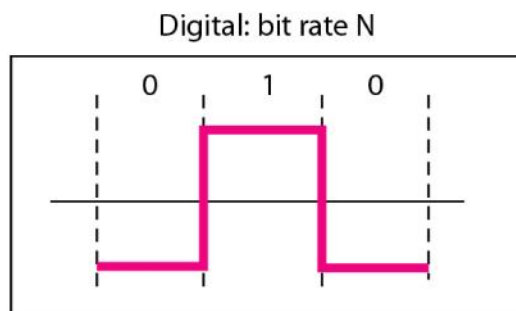
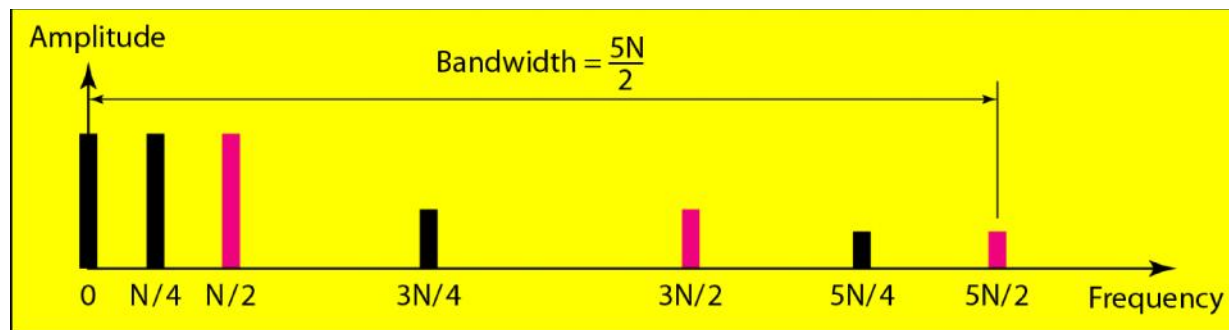
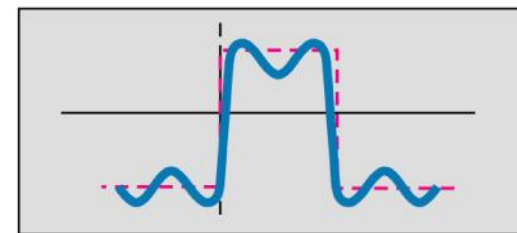
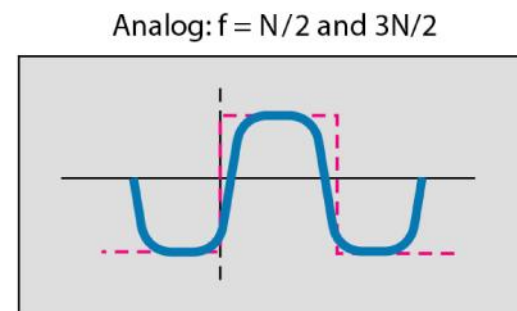


Figure 3.22 *Simulating a digital signal with first three harmonics. This gives us a better representation of the binary pattern.*



Analog: $f = N/2$



Analog: $f = N/2, 3N/2, \text{ and } 5N/2$



Note

In baseband transmission, the required bandwidth is proportional to the bit rate; if we need to send bits faster, we need more bandwidth.

Bandwidth (B) = 3N/2
where N= Bit Rate

Table 3.2 *Bandwidth requirements*

<i>Bit Rate</i>	<i>Harmonic 1</i>	<i>Harmonics 1, 3</i>	<i>Harmonics 1, 3, 5</i>
<i>n = 1 kbps</i>	<i>B = 500 Hz</i>	<i>B = 1.5 kHz</i>	<i>B = 2.5 kHz</i>
<i>n = 10 kbps</i>	<i>B = 5 kHz</i>	<i>B = 15 kHz</i>	<i>B = 25 kHz</i>
<i>n = 100 kbps</i>	<i>B = 50 kHz</i>	<i>B = 150 kHz</i>	<i>B = 250 kHz</i>

Bandwidth (B) = 5N/2
where N= Bit Rate



Example 3.22

What is the required bandwidth of a low-pass channel if we need to send 1 Mbps by using baseband transmission?

Solution

The answer depends on the accuracy desired.

a. The minimum bandwidth, is $B = \text{bit rate} / 2$, or 500 kHz.

b. A better solution is to use the first and the third harmonics with $B = 3 \times 500 \text{ kHz} = 1.5 \text{ MHz}$.

c. Still a better solution is to use the first, third, and fifth harmonics with $B = 5 \times 500 \text{ kHz} = 2.5 \text{ MHz}$.



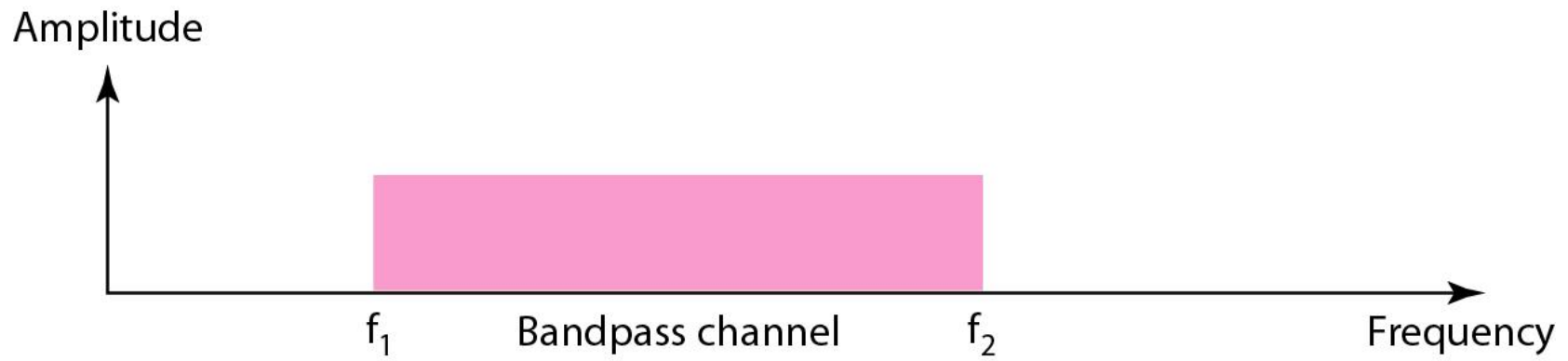
Example 3.22

We have a low-pass channel with bandwidth 100 kHz. What is the maximum bit rate of this channel?

Solution

The maximum bit rate can be achieved if we use the first harmonic. The bit rate is 2 times the available bandwidth, or 200 kbps.

Figure 3.23 *Bandwidth of a bandpass channel*

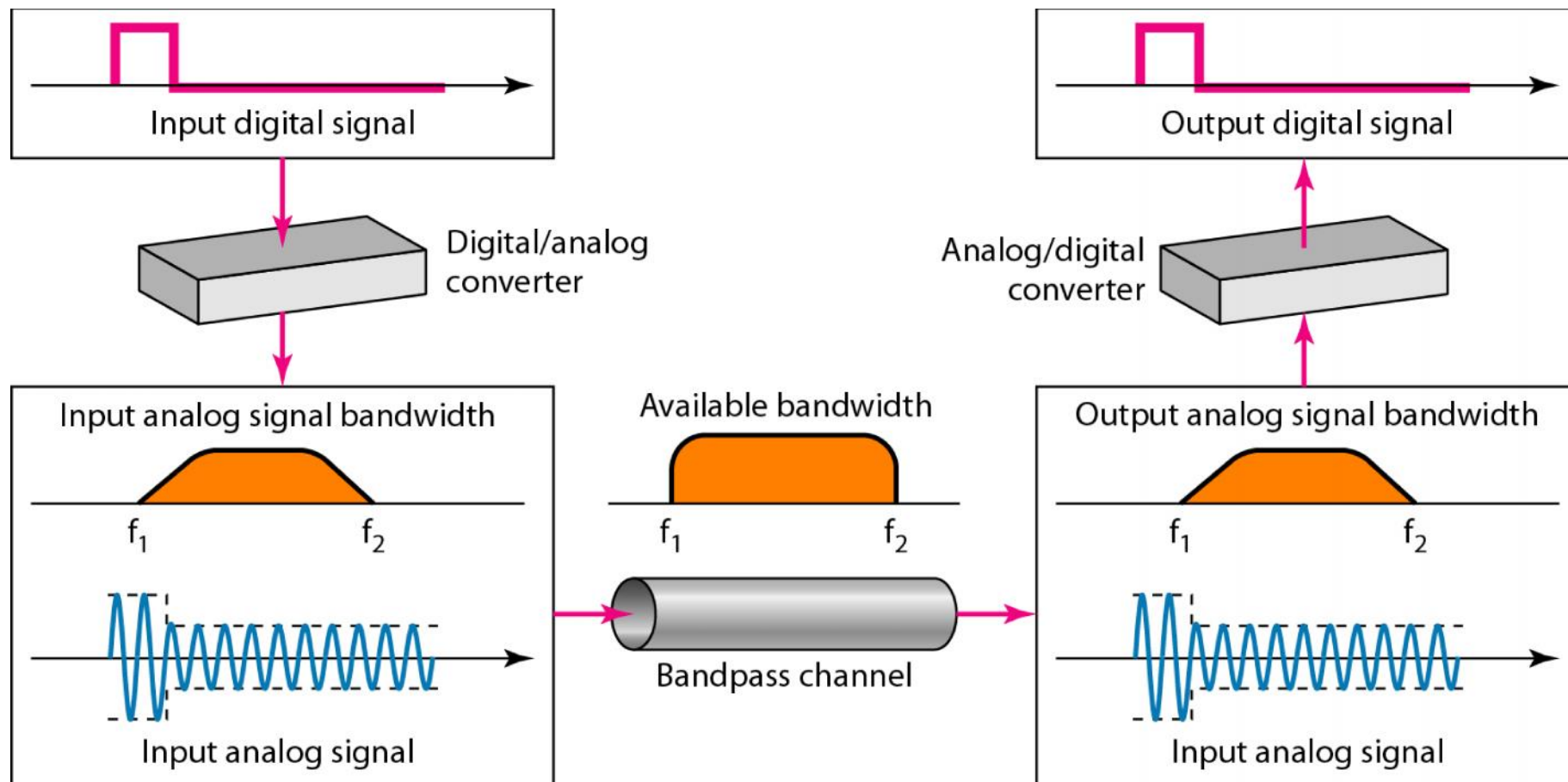




Note

If the available channel is a bandpass channel, we cannot send the digital signal directly to the channel; we need to convert the digital signal to an analog signal before transmission.

Figure 3.24 *Modulation of a digital signal for transmission on a bandpass channel*



3-4 TRANSMISSION IMPAIRMENT

*Signals travel through transmission media, which are not perfect. The imperfection causes signal impairment. This means that the signal at the beginning of the medium is not the same as the signal at the end of the medium. What is sent is not what is received. Three causes of impairment are **attenuation**, **distortion**, and **noise**.*

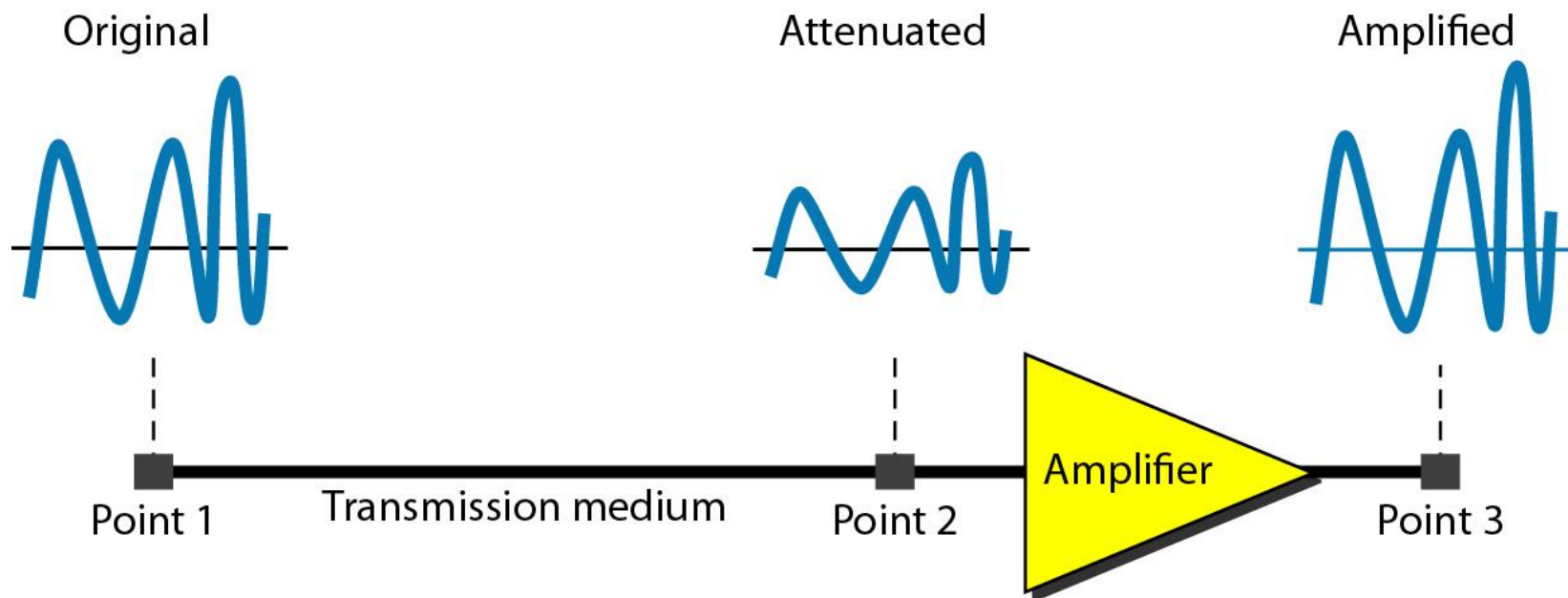
Topics discussed in this section:

Attenuation

Distortion

Noise

Figure 3.26 *Attenuation (the first impairment)*



Decibels

- ▶ Signal strength is measured in decibels (dB)
- ▶ dB is a relative measure of loss (or gain)
- ▶ $N_{dB} = 10 \times \log_{10} (P2 / P1)$
 - P2 = ending power level in watts
 - P1 = beginning power level in watts
- ▶ Example: P1 = 10 watts, P2 = 5 watts
- ▶ Even easier – remember ½ rule
- ▶ Losses and gains are additive

Signal to Noise Ratio (SNR or S/N)

- Signal to noise ratio shows the ratio of signal power to noise power
- Power often expressed in watts
- $S/N = \text{signal power}/\text{noise power}$
- Just a simple ratio

Signal to Noise Ratio_{dB} (SNR_{dB} or S/N_{dB})

- Signal to noise ratio_{dB} shows the ratio of signal power to noise power in decibels
- $S/N_{dB} = 10 \log_{10} (\text{signal power}/\text{noise power})$
- Example 1: Signal power = 1000 watts, noise power = 20 mw
- Example 2: Signal power = 100 w, noise power = 0.000002w

Signal to Noise Ratio_{dB}

- What if you know the S/N_{db} but want to know the signal power level?
- For example, you are given that $S/N_{\text{db}} = 85$. What is the signal power level?

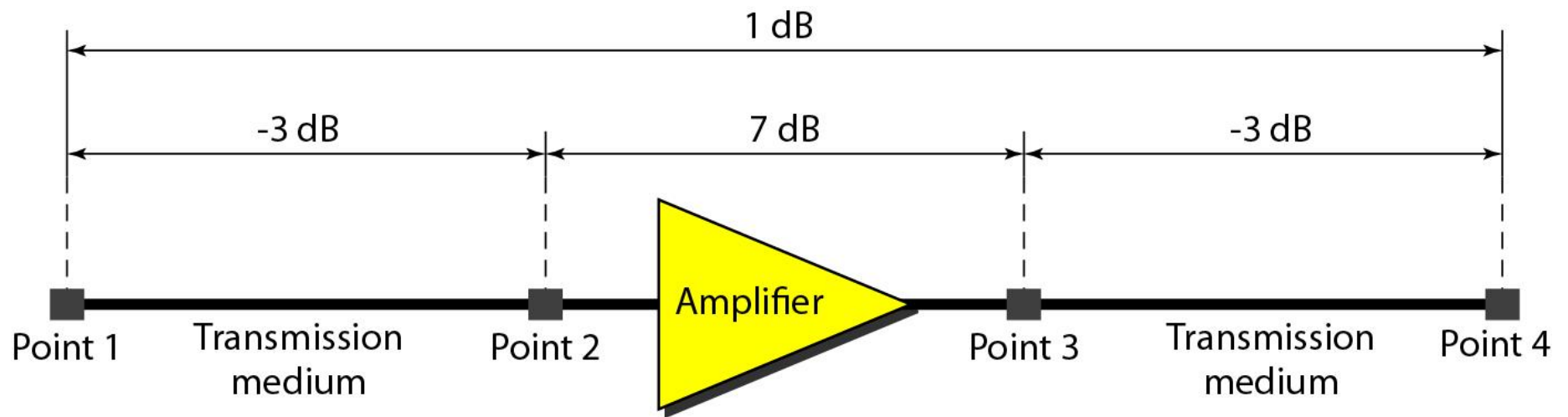


Example 3.28

One reason that engineers use the decibel to measure the changes in the strength of a signal is that decibel numbers can be added (or subtracted) when we are measuring several points (cascading) instead of just two. In Figure 3.27 a signal travels from point 1 to point 4. In this case, the decibel value can be calculated as

$$\text{dB} = -3 + 7 - 3 = +1$$

Figure 3.27 *Decibels for Example 3.28*





Example 3.29

Sometimes the decibel is used to measure signal power in milliwatts. In this case, it is referred to as \mathbf{dB}_m and is calculated as $\mathbf{dB}_m = 10 \log_{10} P_m$, where P_m is the power in milliwatts. Calculate the power of a signal with $\mathbf{dB}_m = -30$.

Solution

We can calculate the power in the signal as

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



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

$$\begin{aligned} \text{dB} &= 10 \log_{10} \frac{P_2}{P_1} = -1.5 \\ \frac{P_2}{P_1} &= 10^{-0.15} = 0.71 \\ P_2 &= 0.71P_1 = 0.7 \times 2 = 1.4 \text{ mW} \end{aligned}$$

Figure 3.28 *Distortion (the second impairment)*

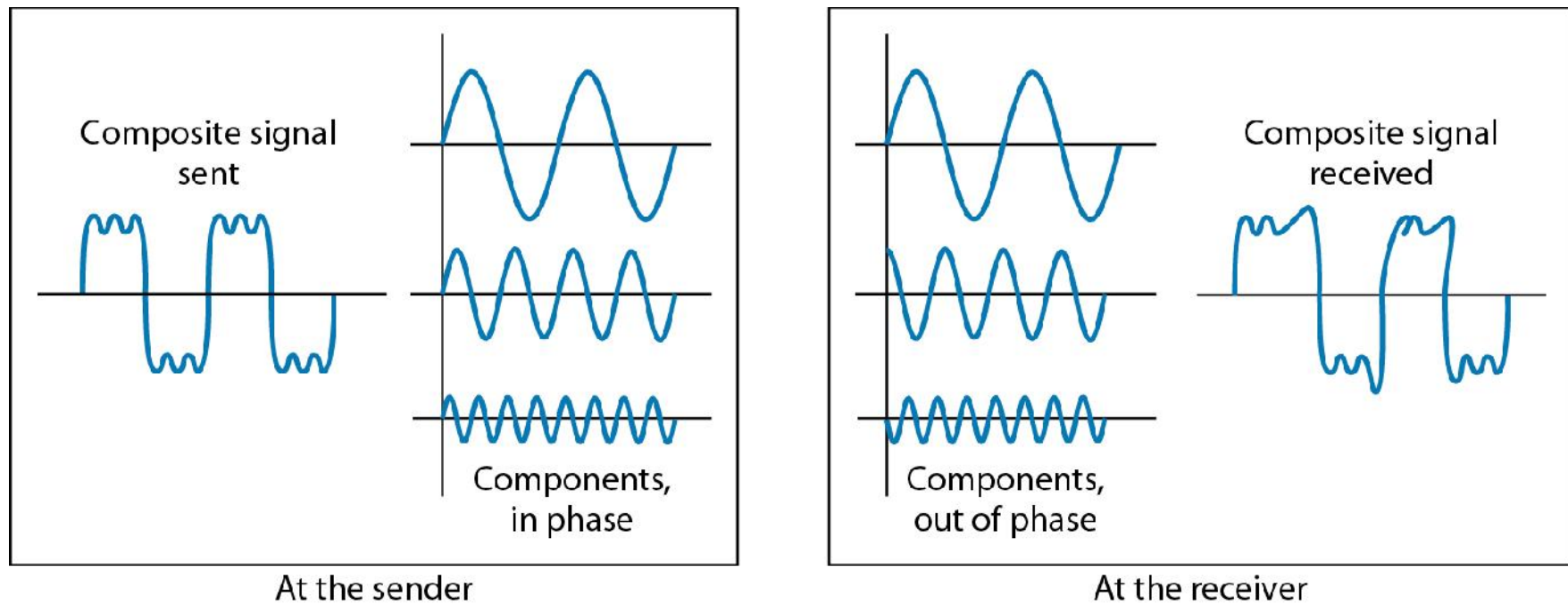
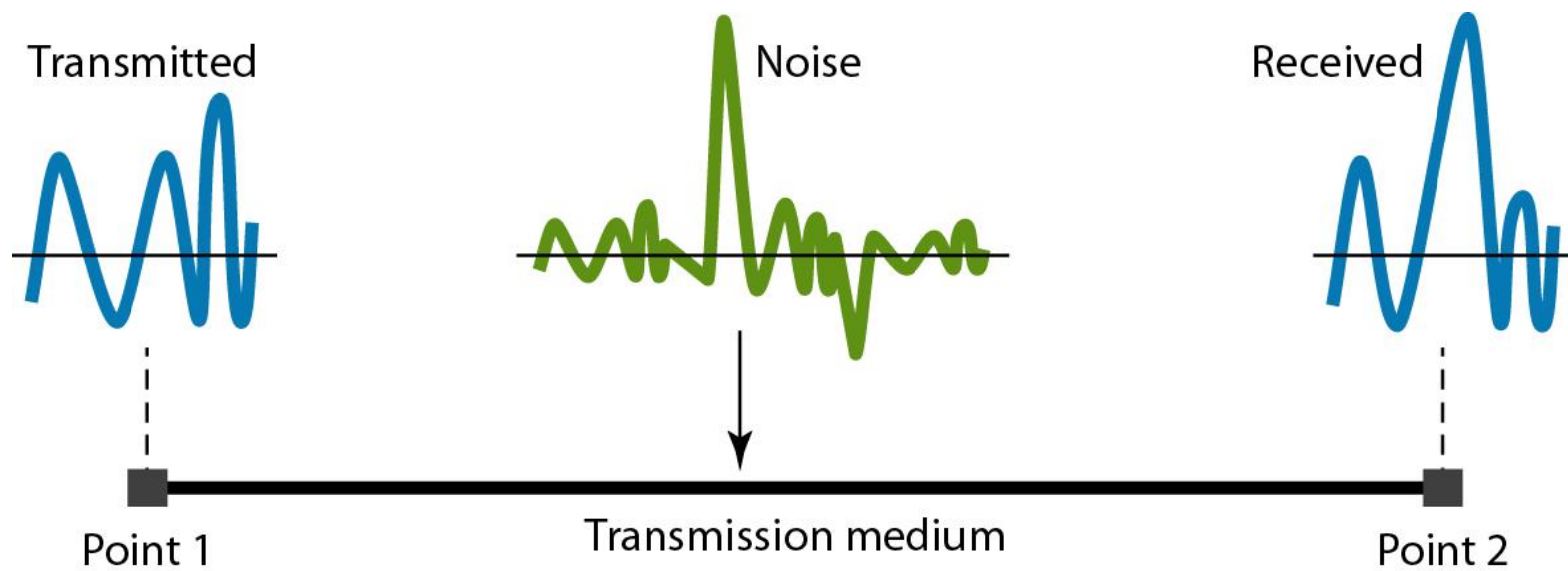


Figure 3.29 *Noise (the third impairment)*





Example 3.31

The power of a signal is 10 mW and the power of the noise is 1 μ W; what are the values of SNR and SNR_{dB} ?

Solution

The values of SNR and SNR_{dB} can be calculated as follows:

$$SNR = \frac{10,000 \mu W}{1 mW} = 10,000$$
$$SNR_{dB} = 10 \log_{10} 10,000 = 10 \log_{10} 10^4 = 40$$



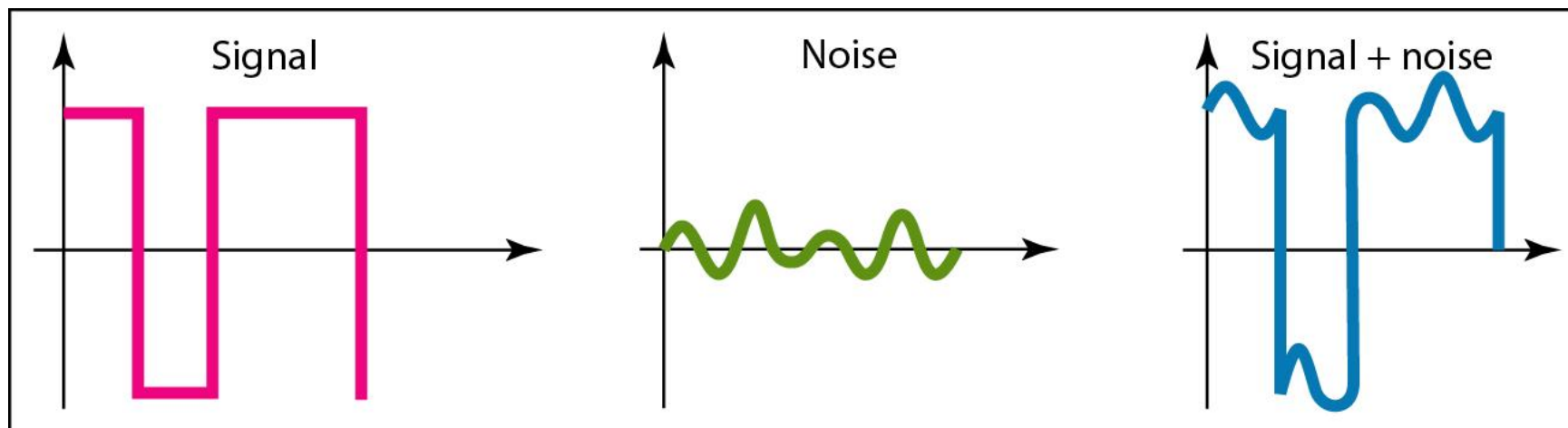
Example 3.32

The values of SNR and SNR_{dB} for a noiseless channel are

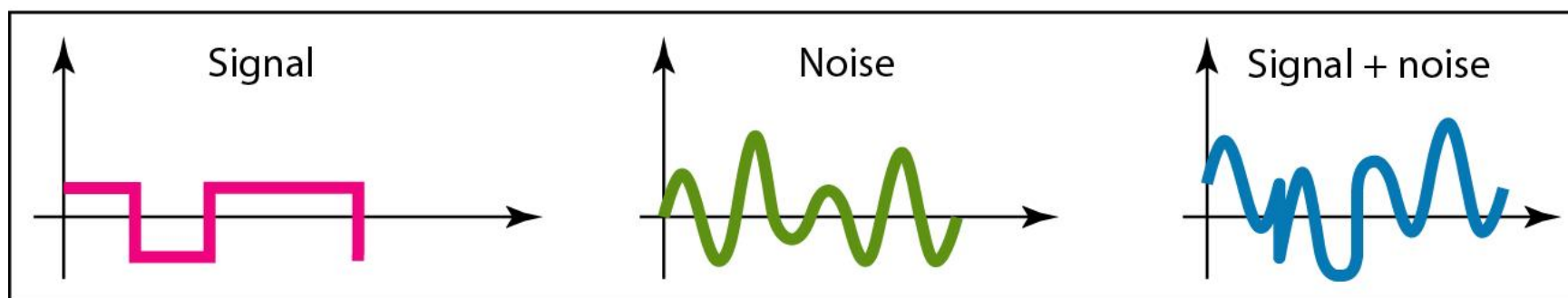
$$SNR = \frac{\text{signal power}}{0} = \infty$$
$$SNR_{dB} = 10 \log_{10} \infty = \infty$$

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

Figure 3.30 *Two cases of SNR: a high SNR and a low SNR*



a. Large SNR



b. Small SNR

3-5 DATA RATE LIMITS

A very important consideration in data communications is how fast we can send data, in bits per second, over a channel. Data rate depends on three factors:

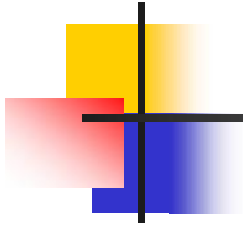
- 1. The bandwidth available*
- 2. The level of the signals we use*
- 3. The quality of the channel (the level of noise)*

Topics discussed in this section:

Noiseless Channel: Nyquist Bit Rate

Noisy Channel: Shannon Capacity

Using Both Limits



Note

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

Channel Capacity

- Data rate
 - In bits per second
 - Rate at which data can be communicated
- Bandwidth
 - In cycles per second of Hertz
 - Constrained by transmitter and medium
- Baud rate
 - Frequency with which the components change

BPS vs. Baud

- Data rate rarely the same as baud rate
- Examples:

Channel Capacity

- Nyquist

- Maximum data rate of a noiseless channel =
 $2 * F * \log_2(L)$ bps
- Where F = frequency
- L = the number of discrete levels
- Example: $F = 4000$ Hz, $L = 2$



Example 3.34

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

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



Example 3.35

Consider the same noiseless channel transmitting a signal with four signal levels (for each level, we send 2 bits). The maximum bit rate can be calculated as

$$\text{BitRate} = 2 \times 3000 \times \log_2 4 = 12,000 \text{ bps}$$



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:

$$\begin{aligned} 265,000 &= 2 \times 20,000 \times \log_2 L \\ \log_2 L &= 6.625 \quad L = 2^{6.625} = 98.7 \text{ levels} \end{aligned}$$

Since this result 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.

Channel Capacity

- Shannon (which includes noise)
 - Maximum data rate (in bps) = $B \times \log_2(1 + S/N)$
 - B = bandwidth, or frequency
 - S = signal power in watts
 - N = noise power in watts (S/N = SNR)
 - Example: H = 3400 Hz, S = 0.2 w, N = 0.0002 w
 - Max data rate = $3400 \times \log_2(1 + 1000)$
 - = 3400×9.97
 - = 33898 bps



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 $SNR_{dB} = 36$ and the channel bandwidth is 2 MHz. The theoretical channel capacity can be calculated as

$$SNR_{dB} = 10 \log_{10} SNR \quad \rightarrow \quad SNR = 10^{SNR_{dB}/10} \quad \rightarrow \quad SNR = 10^{3.6} = 3981$$
$$C = B \log_2 (1 + SNR) = 2 \times 10^6 \times \log_2 3982 = 24 \text{ Mbps}$$



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.

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



Note

The Shannon capacity gives us the upper limit; the Nyquist formula tells us how many signal levels we need.



Note

In networking, we use the term bandwidth in two contexts.

- The first, bandwidth in hertz, refers to the range of frequencies in a composite signal or the range of frequencies that a channel can pass.**
- The second, bandwidth in bits per second, refers to the speed of bit transmission in a channel or link.**



Example 3.42

As the bandwidth increases, so too can the data rate.

Stated another way, to transmit a high data rate, we need a wide bandwidth.

OR

We need to use a signaling technique that has a high number of signal levels.

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.

Review Questions

- What is a composite signal?
- How do you calculate a dB?
- How do you calculate signal to noise ratio?
- How do you read/create a frequency domain representation?
- What is a dc component?

Review Questions

- How do you use the Nyquist formula?
 - 4000 Hz, 8 signal levels, data rate?
 - 50,000 bps data rate, 4000 Hz, how many signal levels?
- How do you use the Shannon formula?
 - 8000 Hz, signal power = 20w, noise power = 0.002w, what is the data rate?
 - 5000 Hz, signal power = 50w, data rate = 20000bps, what is the possible noise power?

COMPUTER NETWORKS – Unit 1

Topic 9

DIGITAL TRANSMISSION

Transmission of Digital Data

Interfaces and Modems

- **Digital Data Transmission**
- **DTE-DCE Interface**
- **Other Interface Standards**
- **Modems**

Figure 6-1

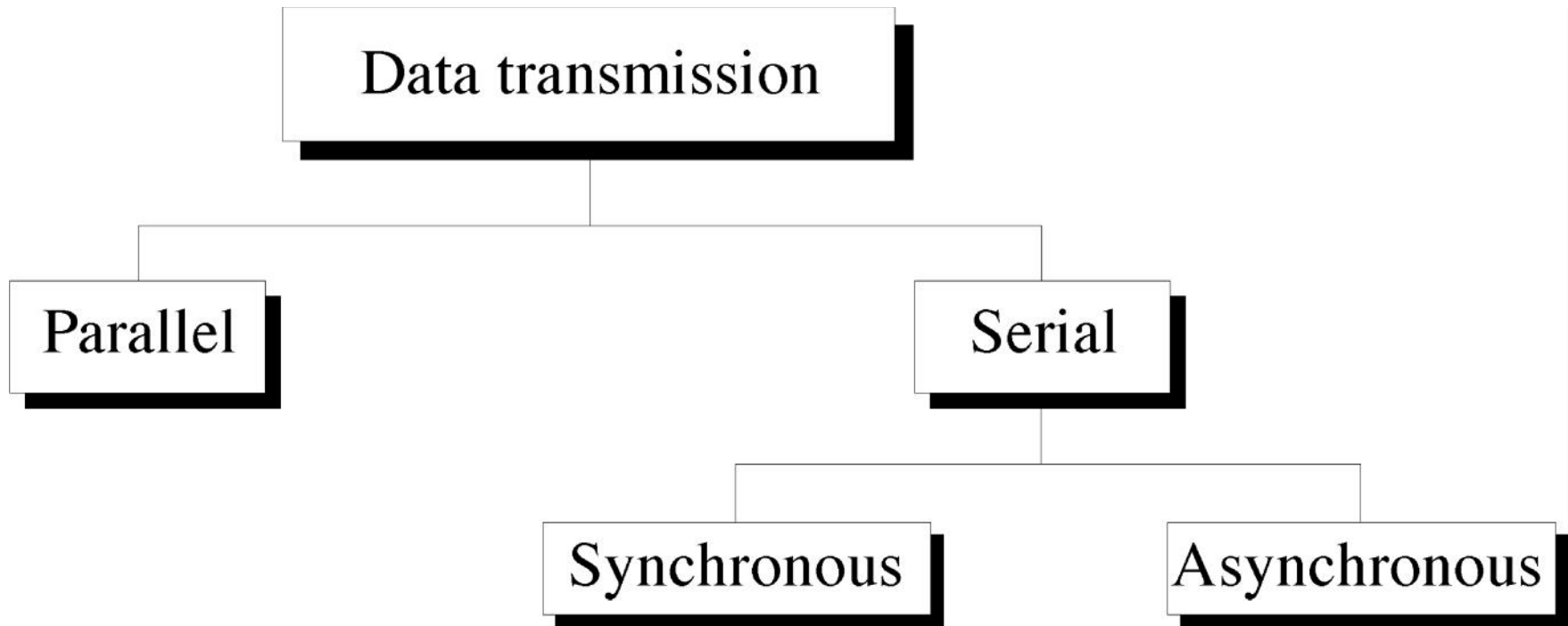


Figure 6-2

Parallel Transmission

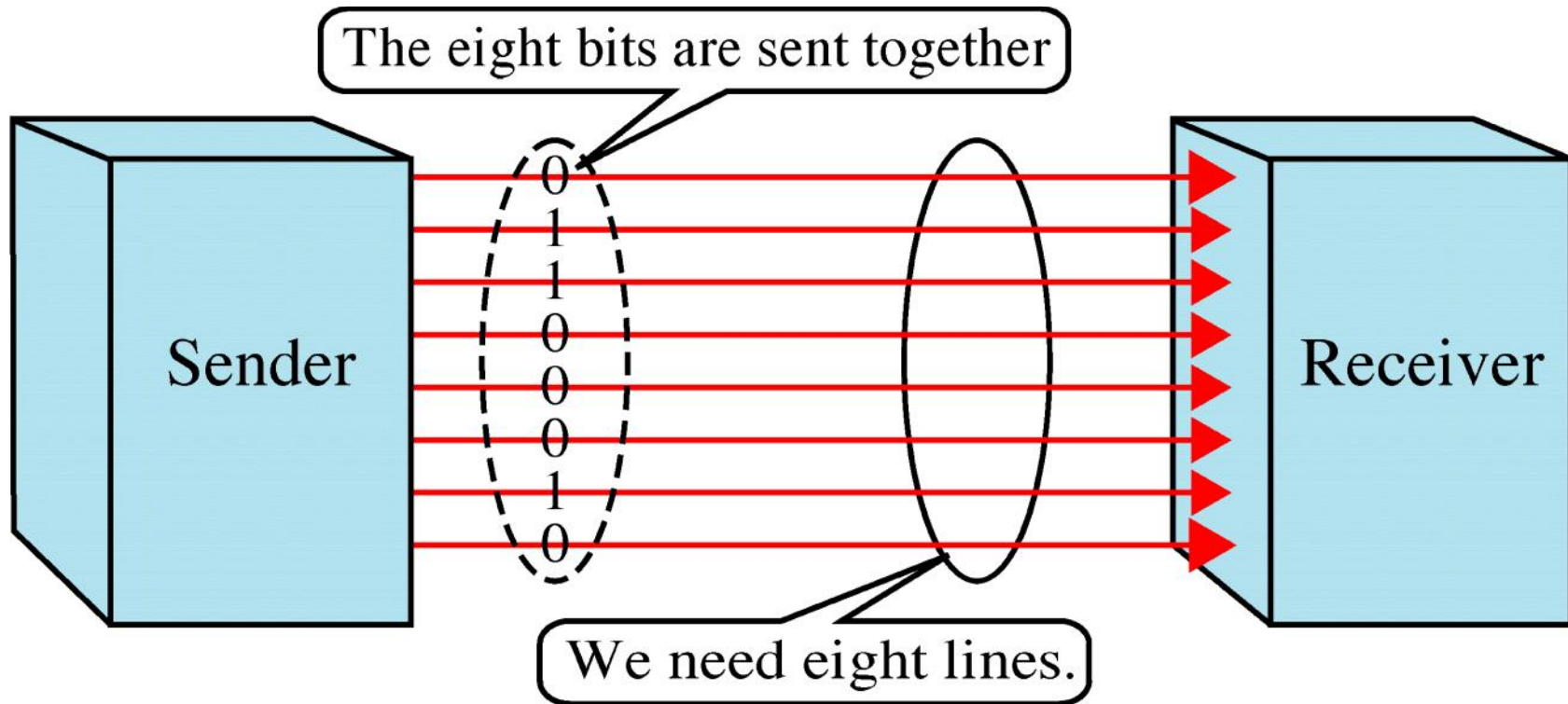


Figure 6-3

Serial Transmission

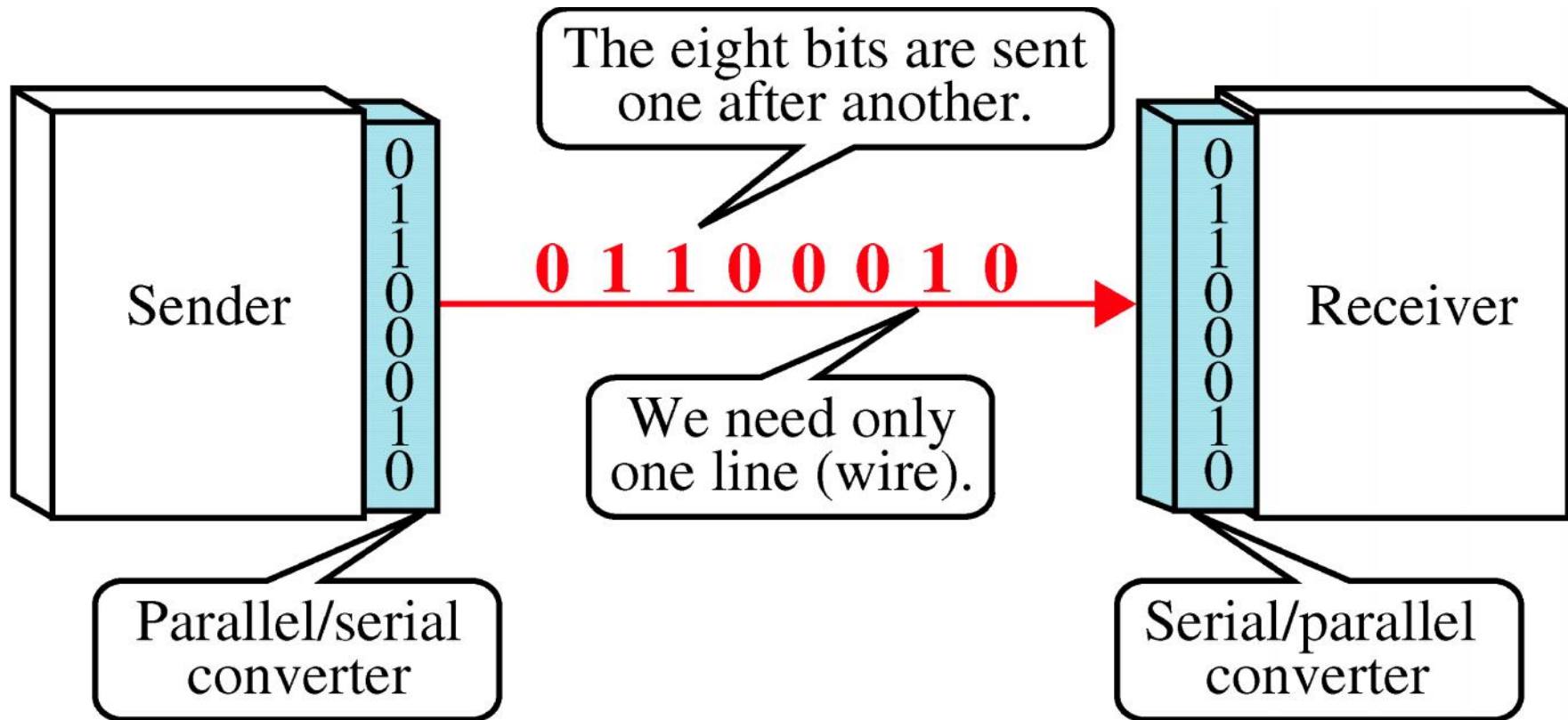


Figure 6-4

Asynchronous Transmission

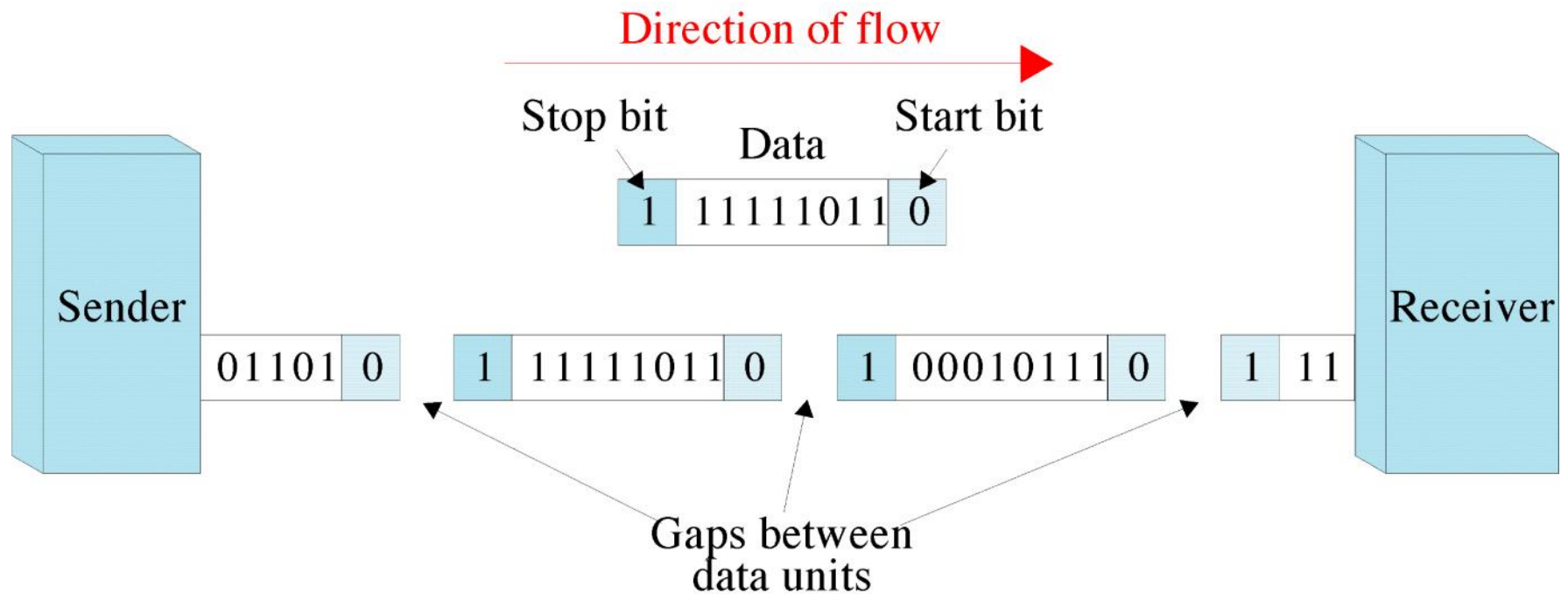


Figure 6-5

Synchronous Transmission

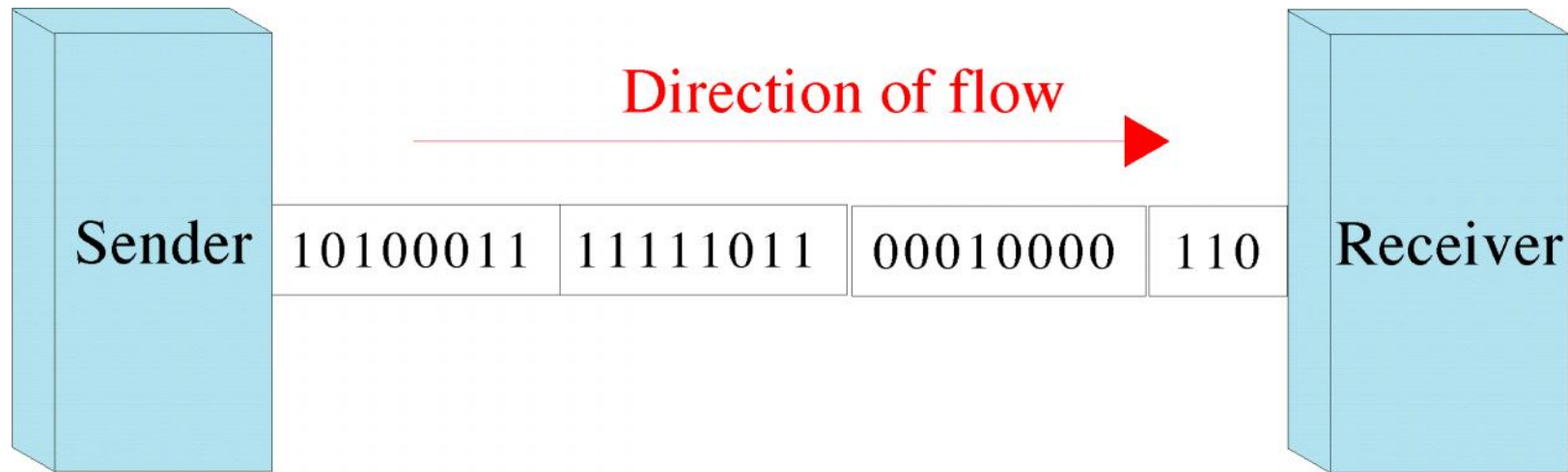


Figure 6-6

DTEs and DCEs

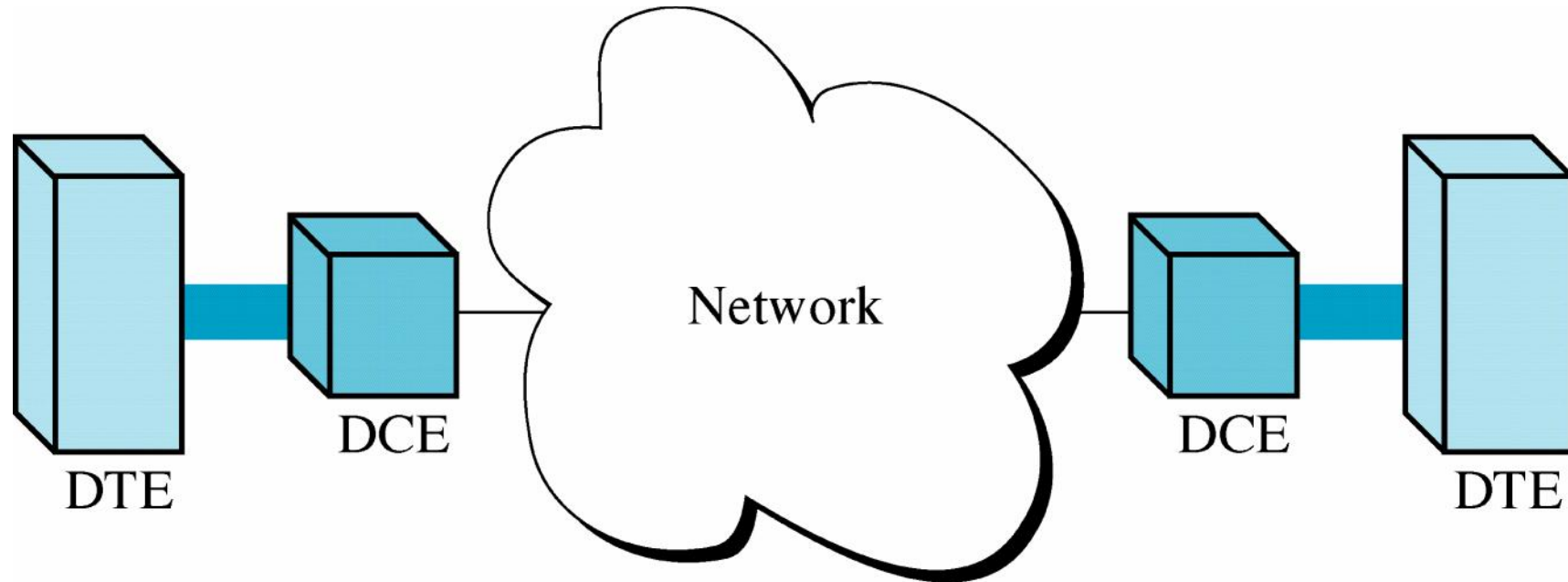


Figure 6-7

DTE-DCE interface

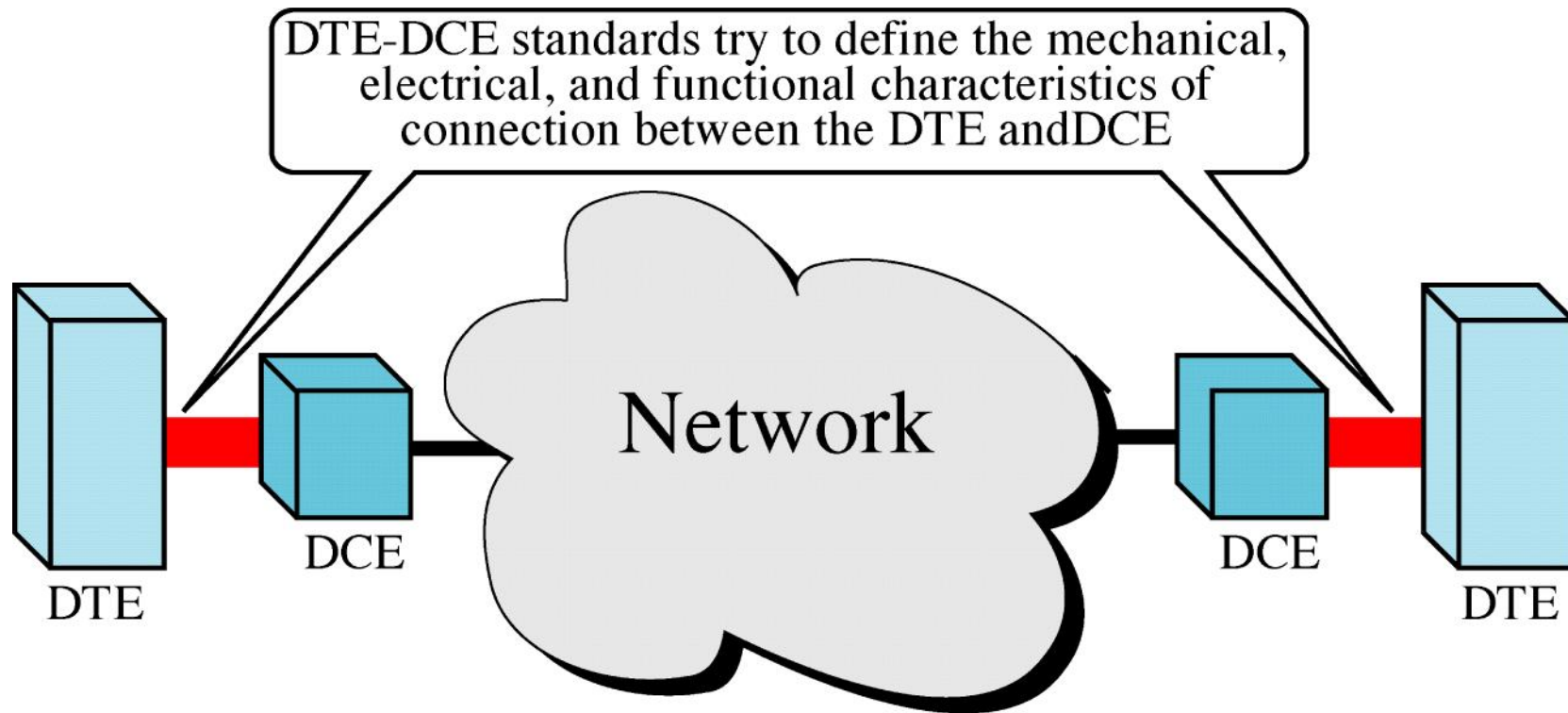


Figure 6-8

Sending Data

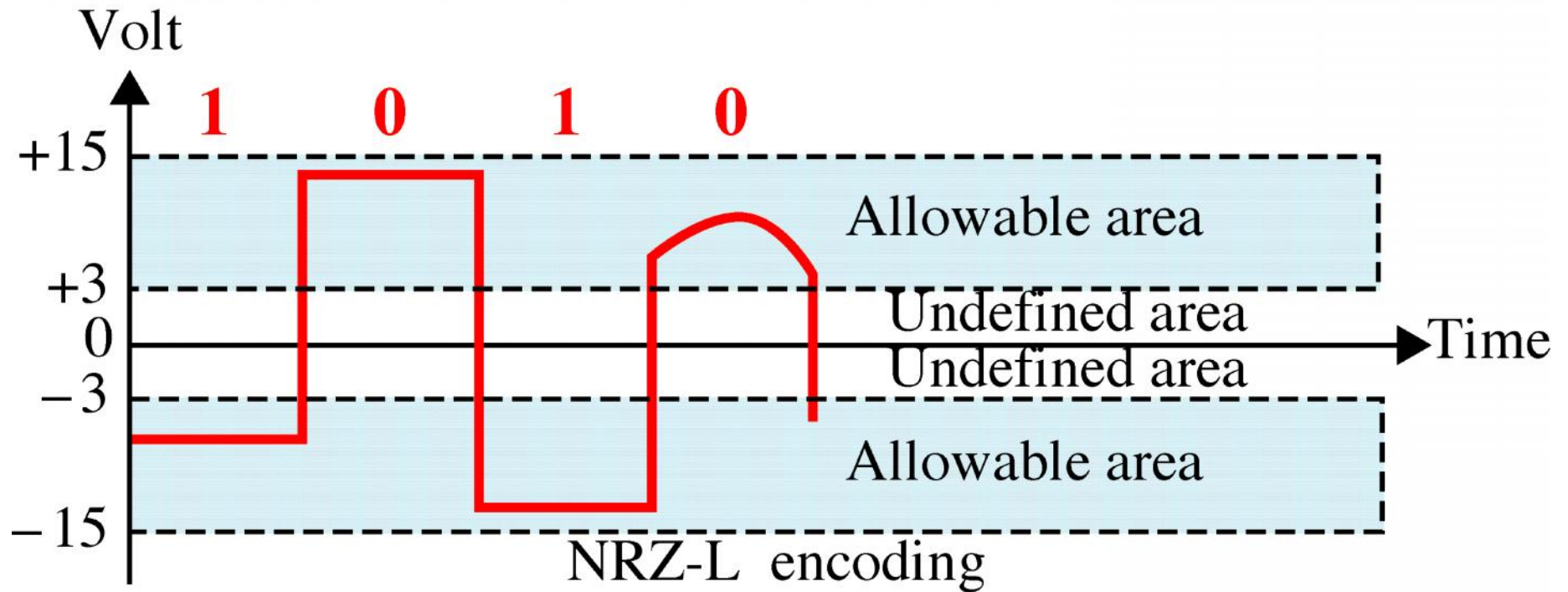


Figure 6-9

Control

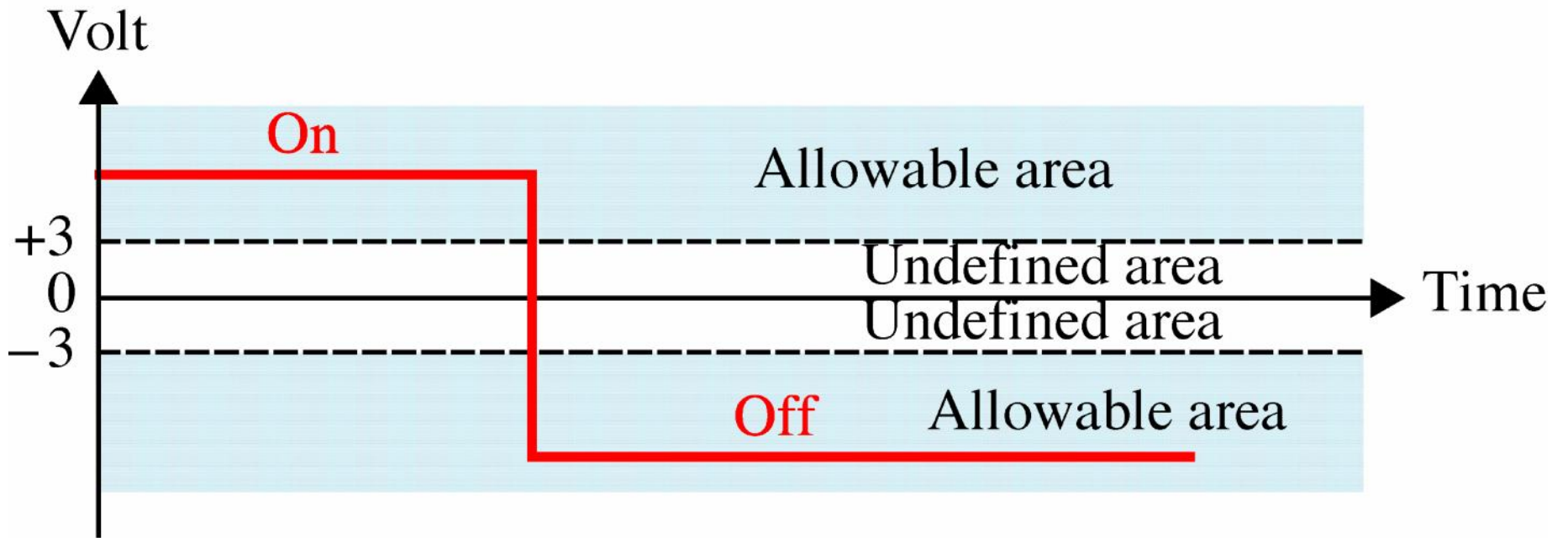


Figure 6-10

EIA-232

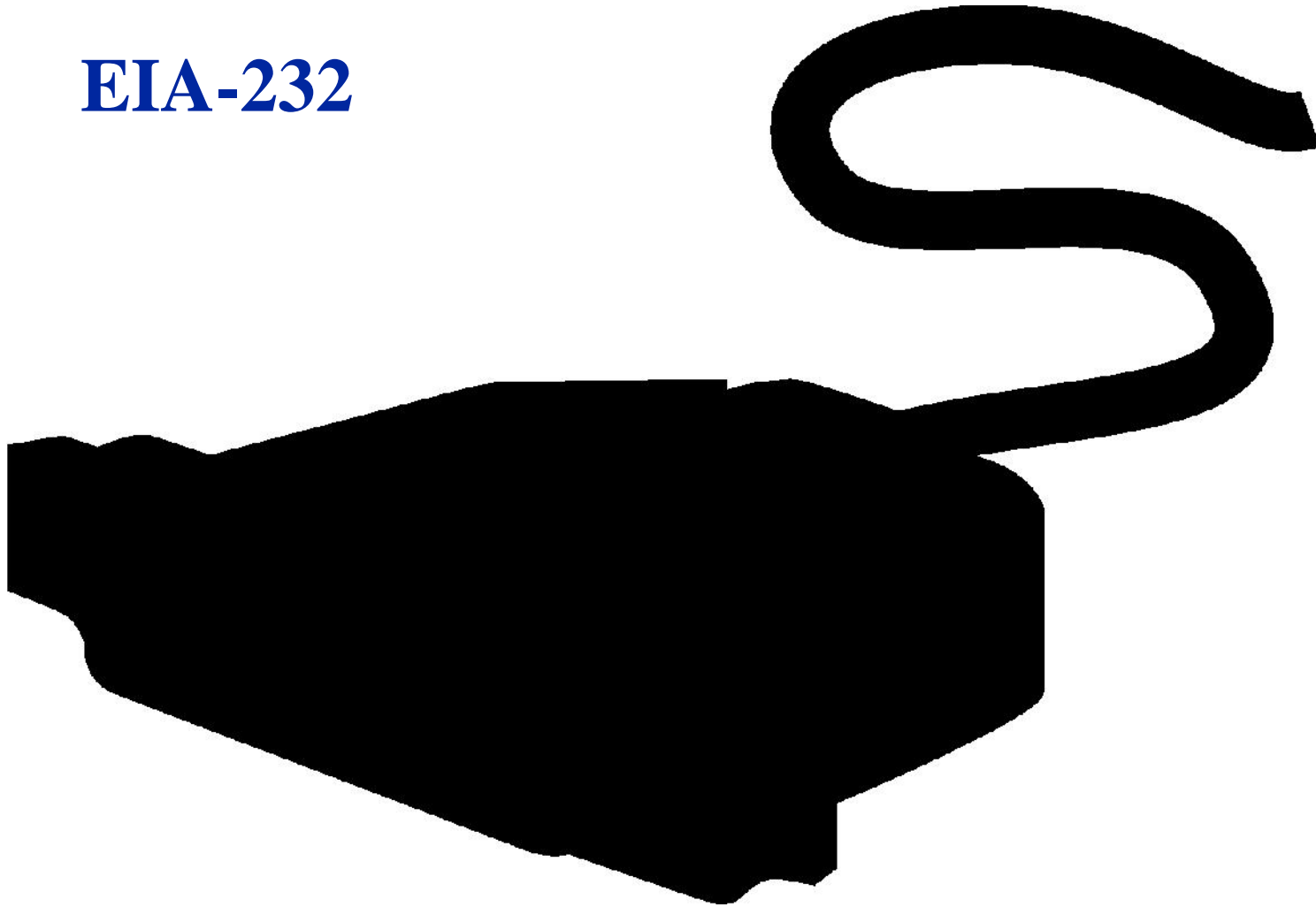


Figure 6-10-continued

Data Pins

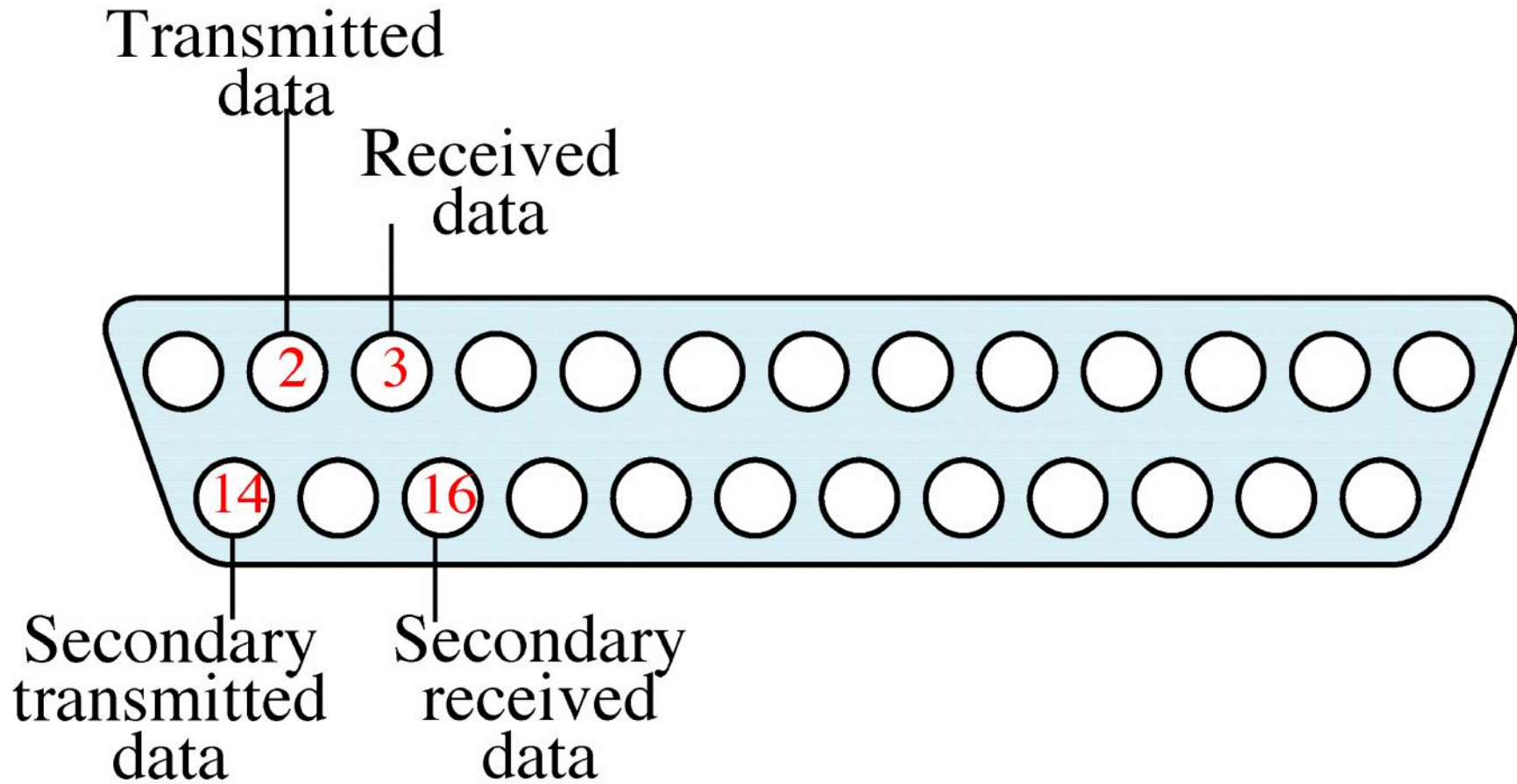


Figure 6-10-continued

Control Pins

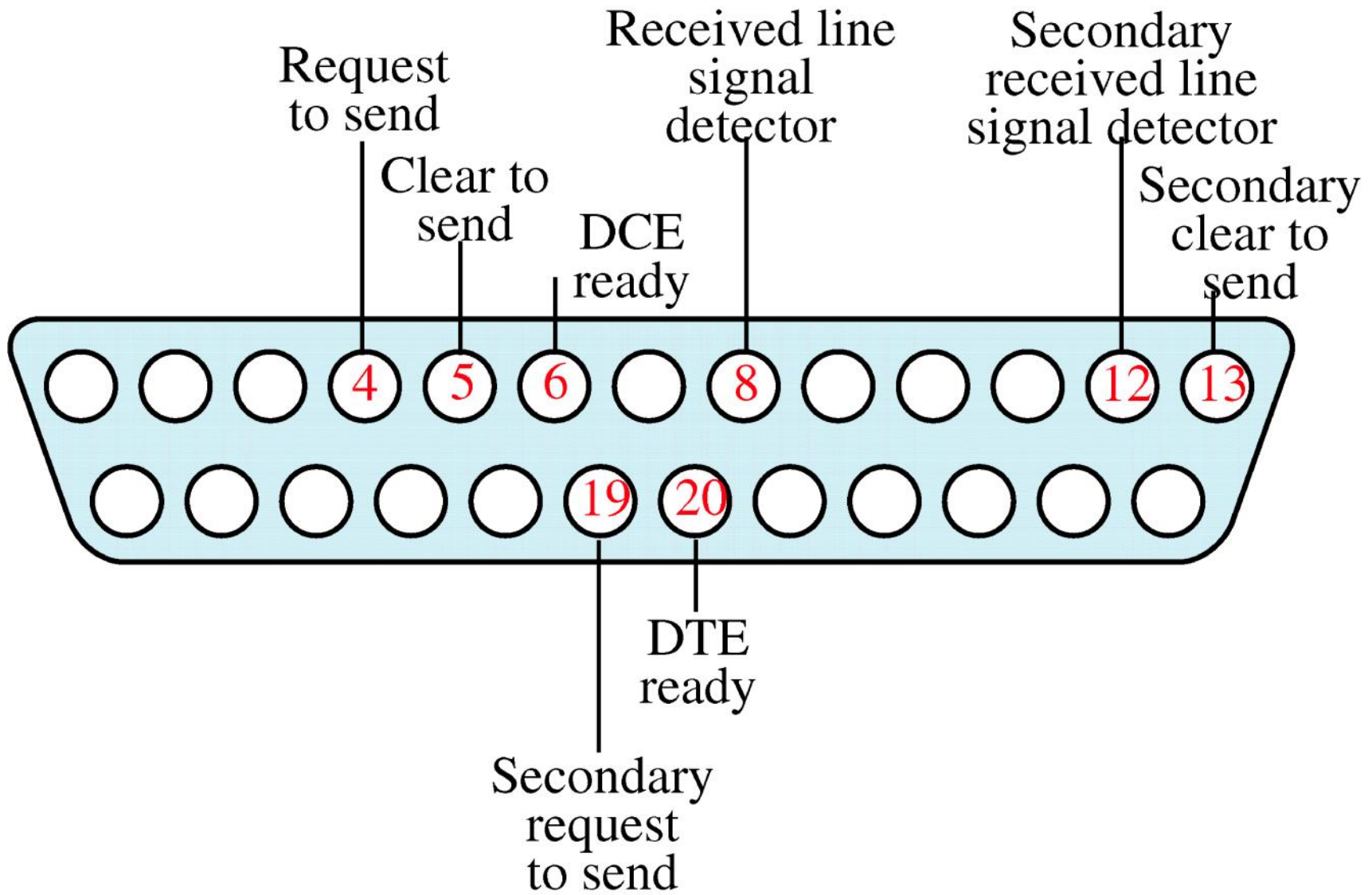


Figure 6-10-continued

Timing Pins

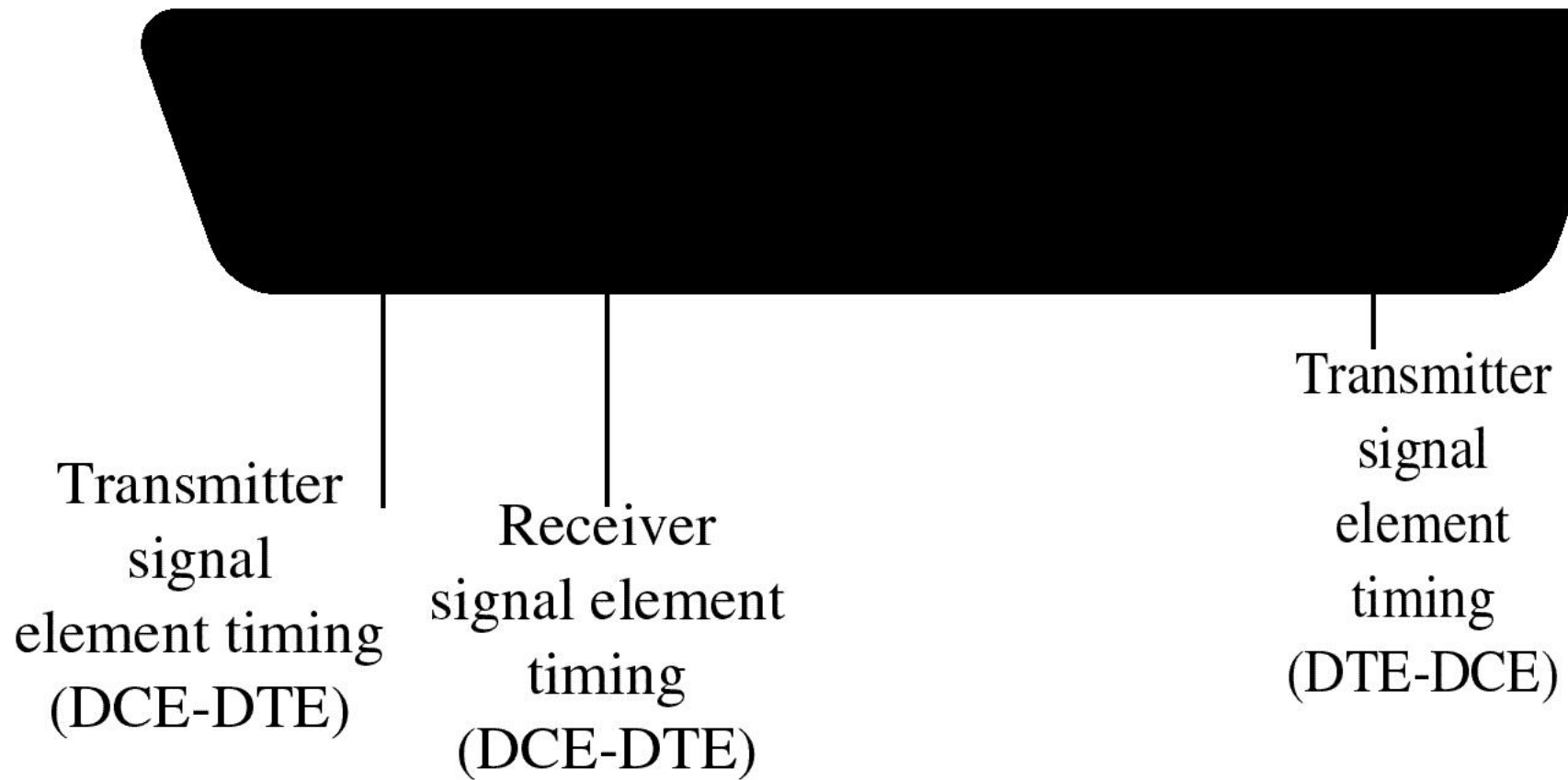


Figure 6-10-continued

Other Pins

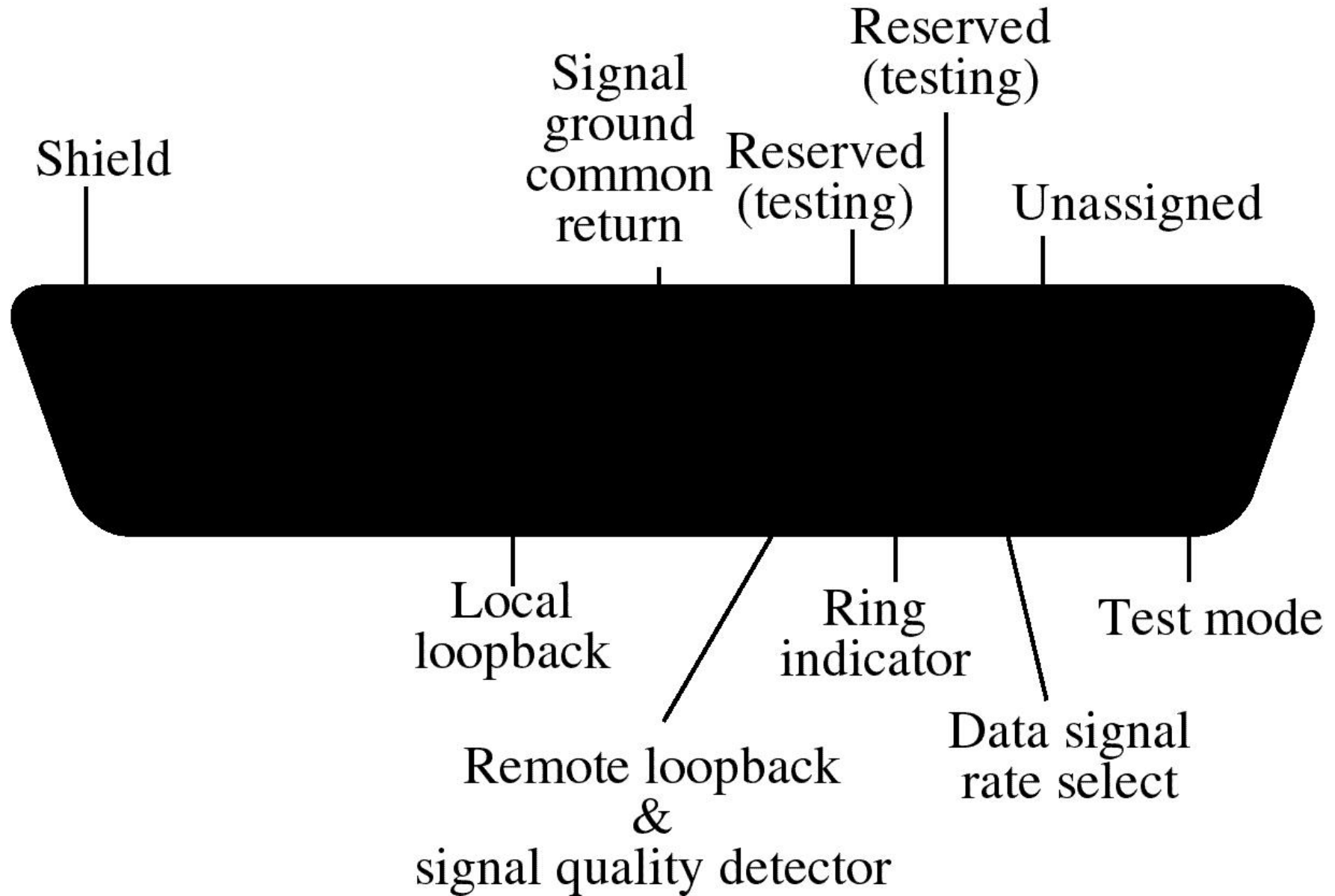


Figure 6-11

Synchronous Full-Duplex Transmission

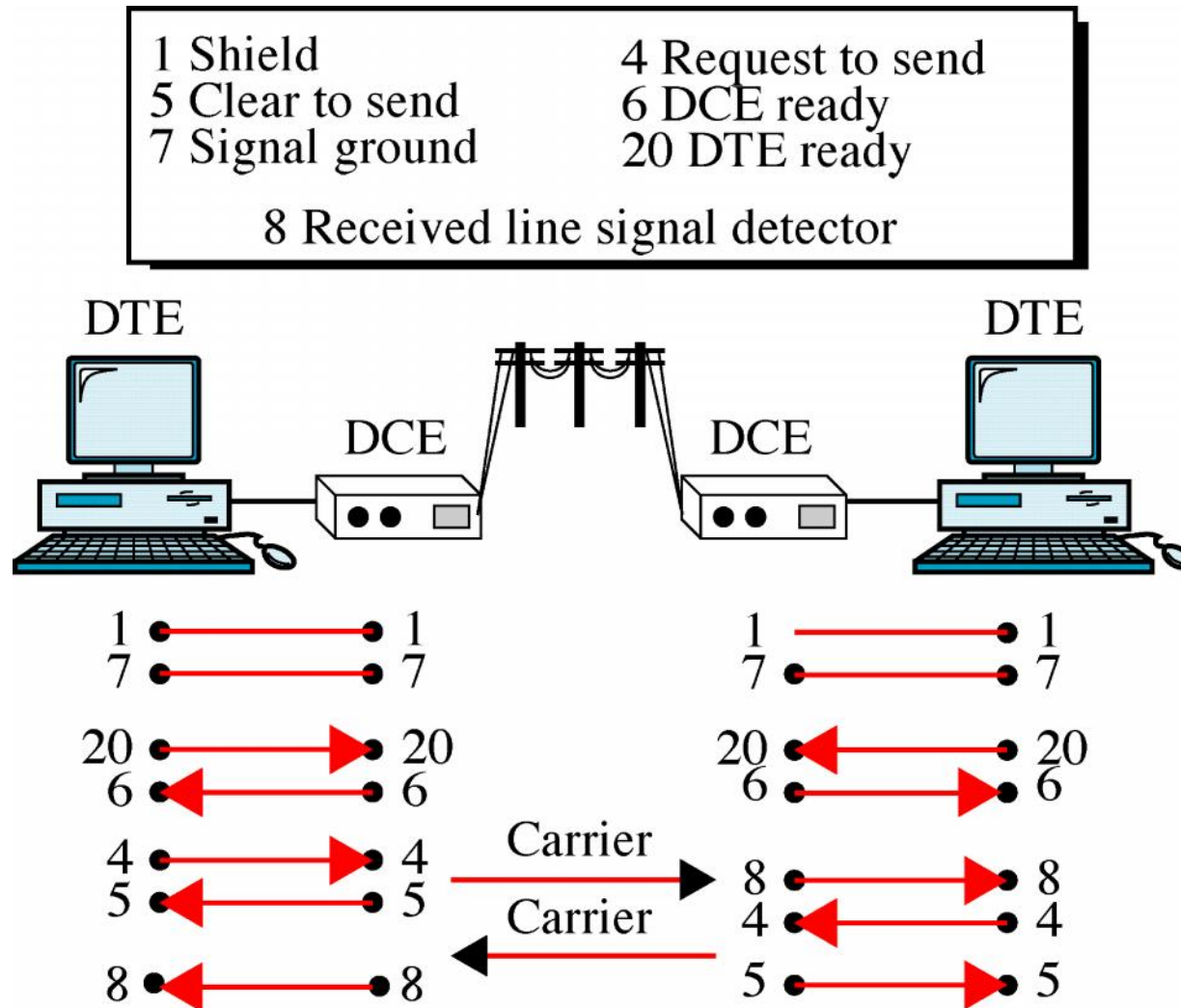


Figure 6-11-continued

Synchronous Full-Duplex Transmission

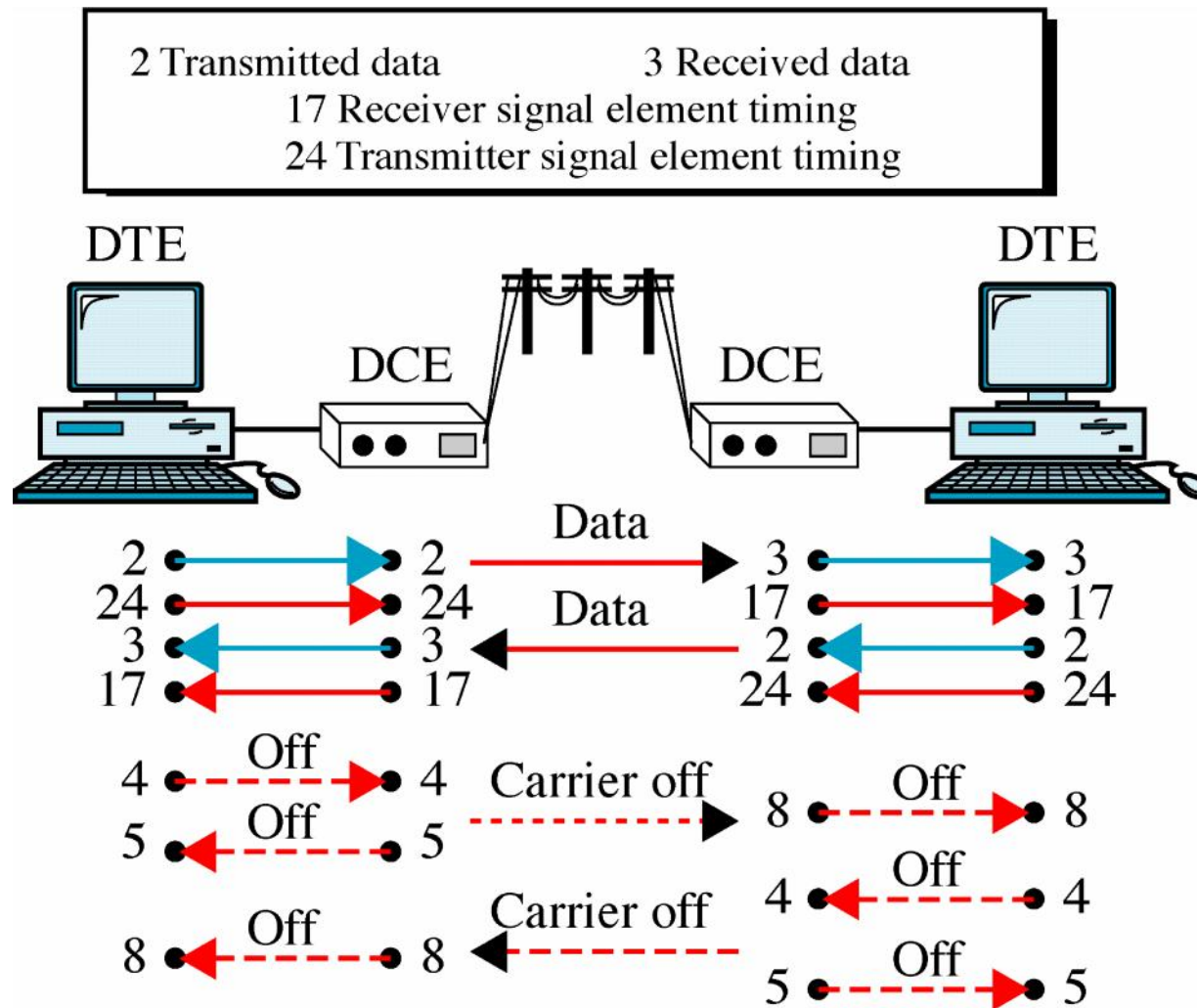


Figure 6-12

Pin Connection With and Without DCEs

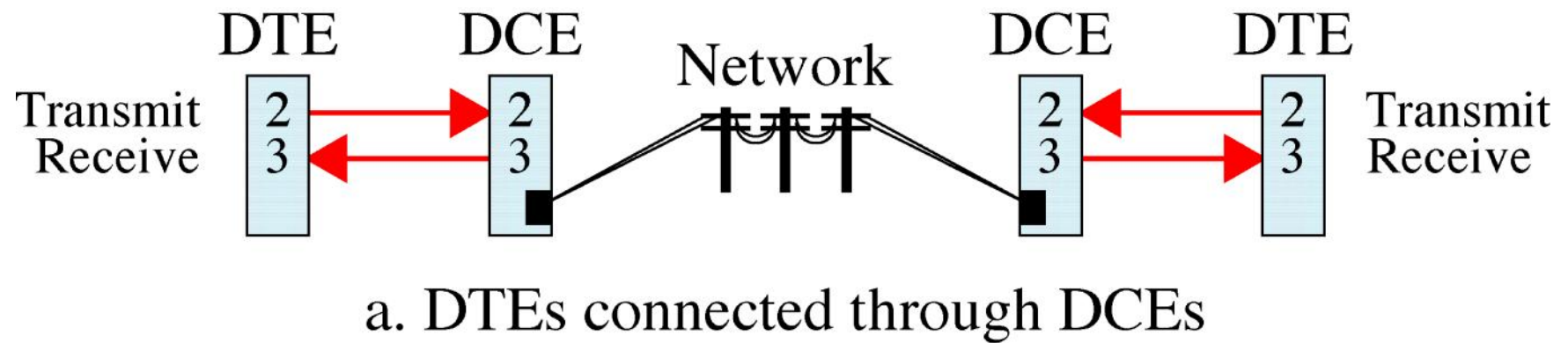
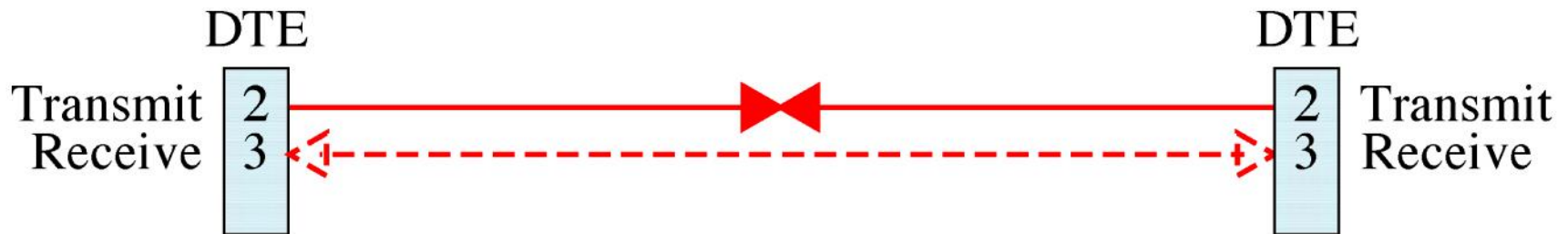


Figure 6-12-continued

Pin Connection With and Without DCEs



a. DTEs connected through DCEs



b. DTEs connected directly

Figure 6-13

Null Modem

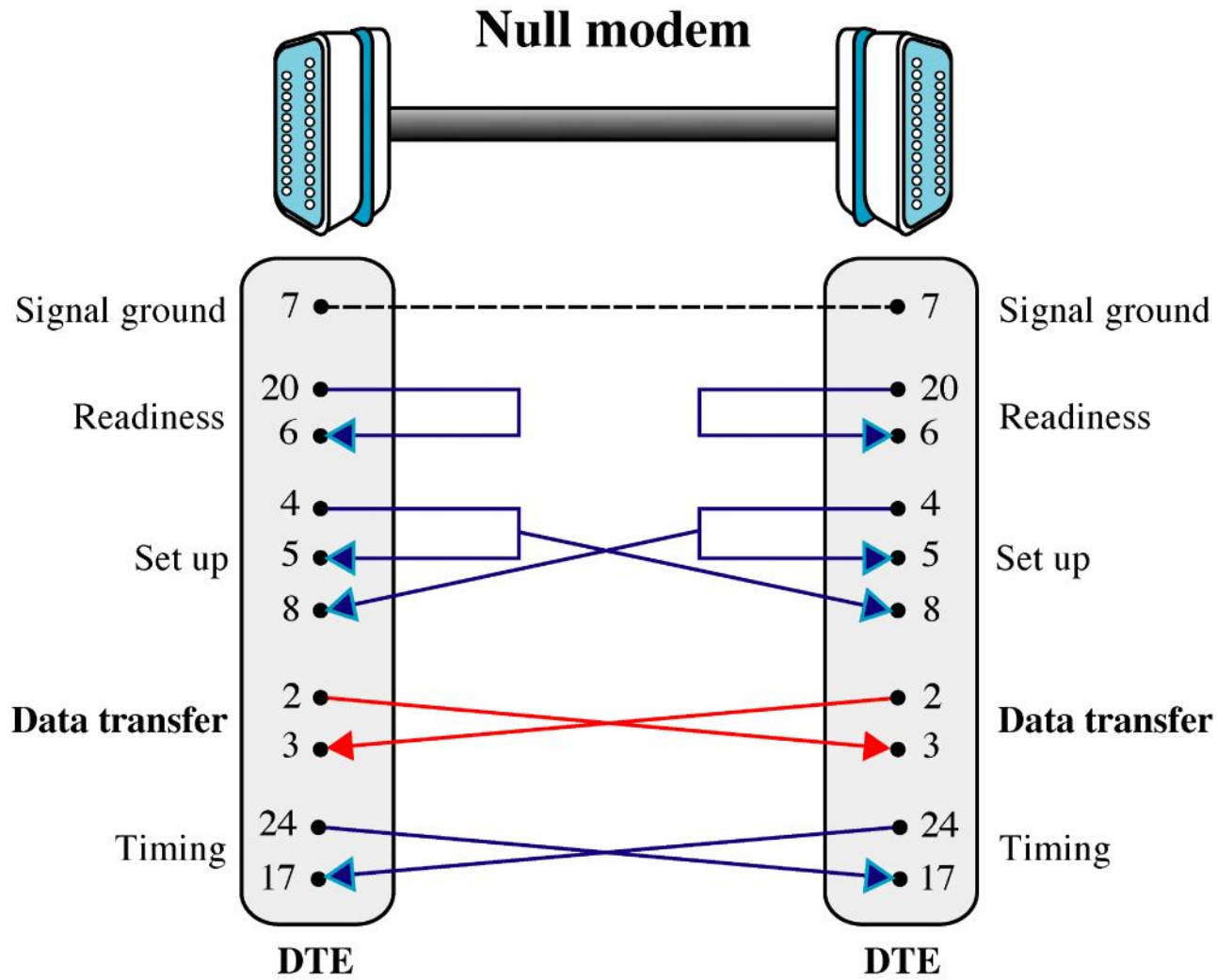
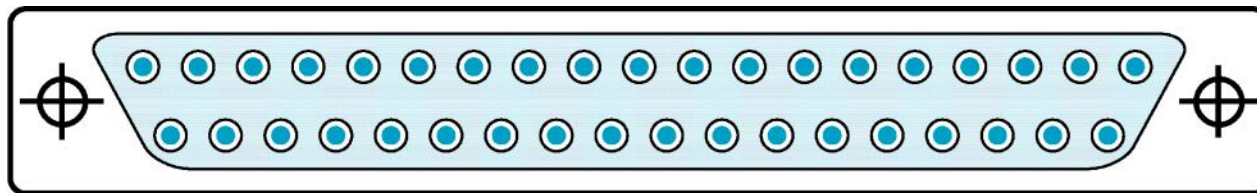
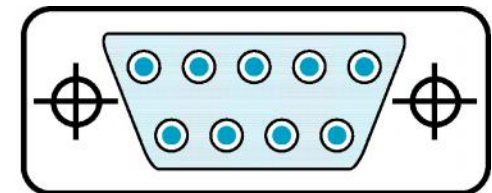


Figure 6-14

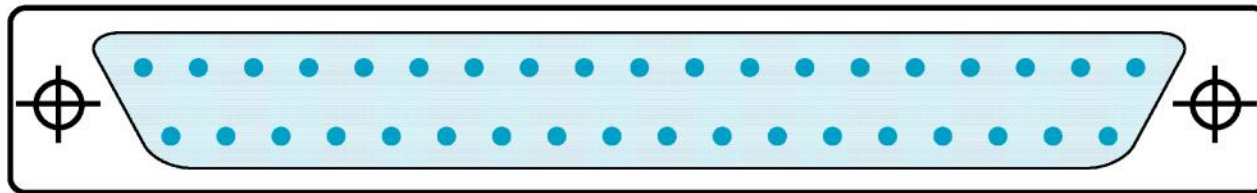
DB-37 and DB-9 Connectors



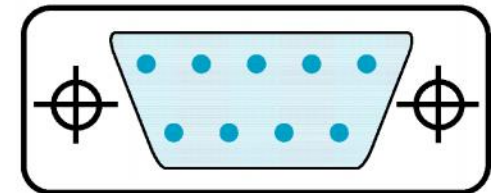
DB-37 receptacle



DB-9 receptacle



DB-37 plug



DB-9 plug

Figure 6-15

RS-423: Unbalanced Mode

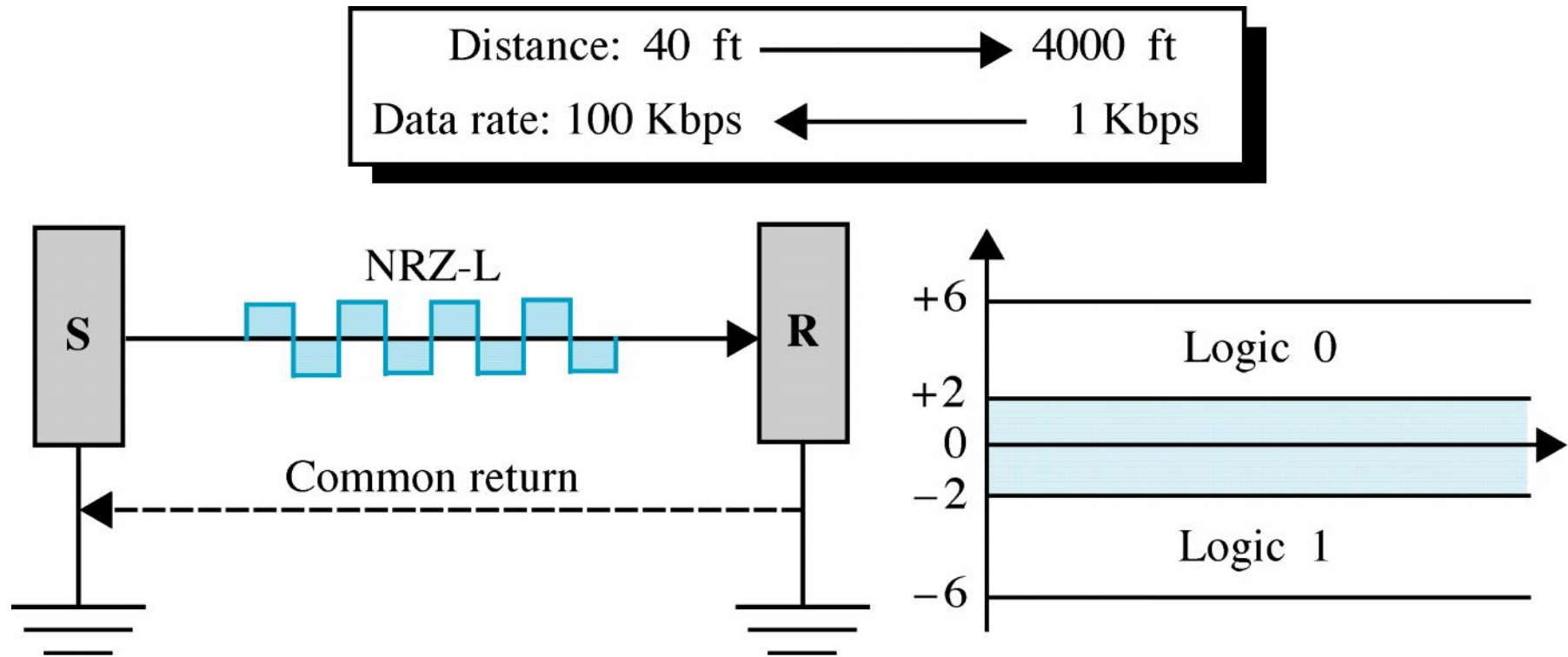


Figure 6-16

RS-422: Balanced Mode

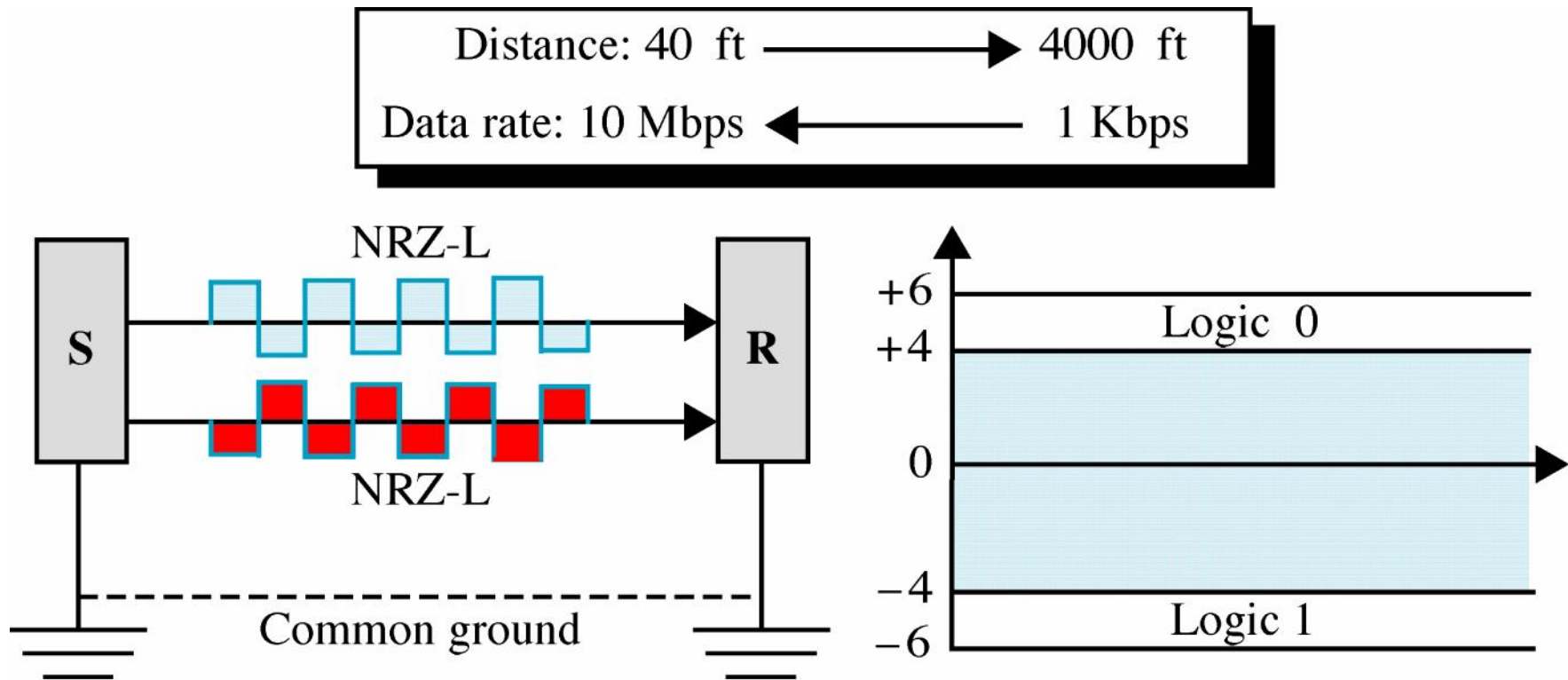


Figure 6-17

Canceling Noise

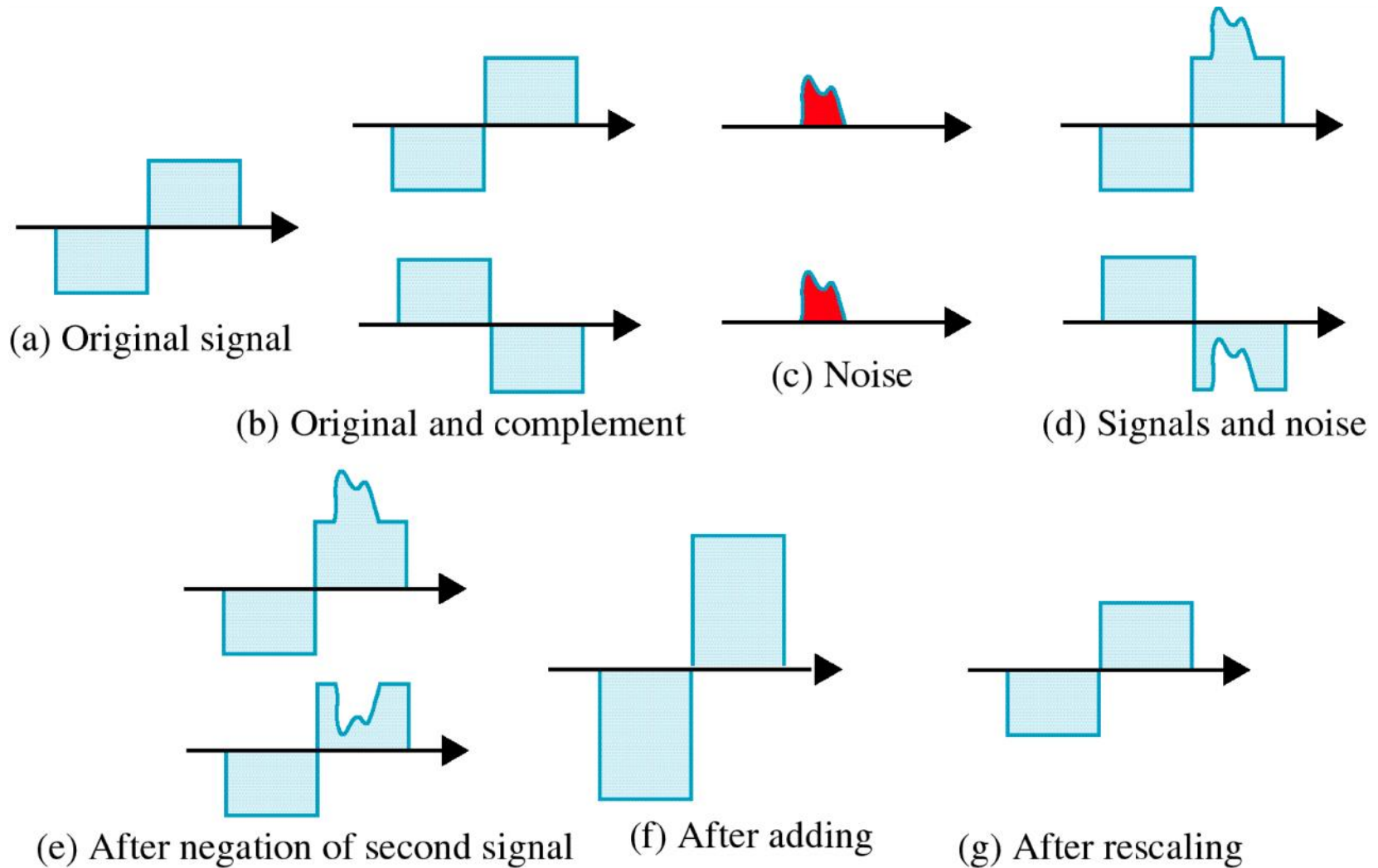
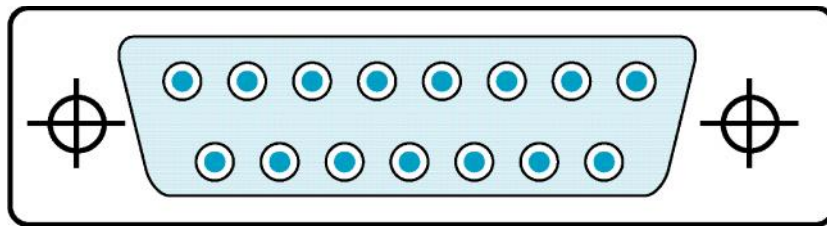
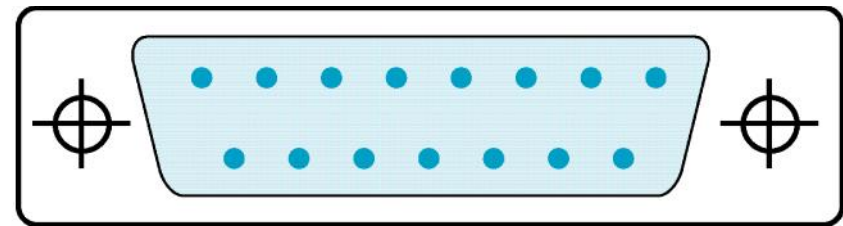


Figure 6-18

DB-15 Connector



DB-15 receptacle



DB-15 plug

Figure 6-19

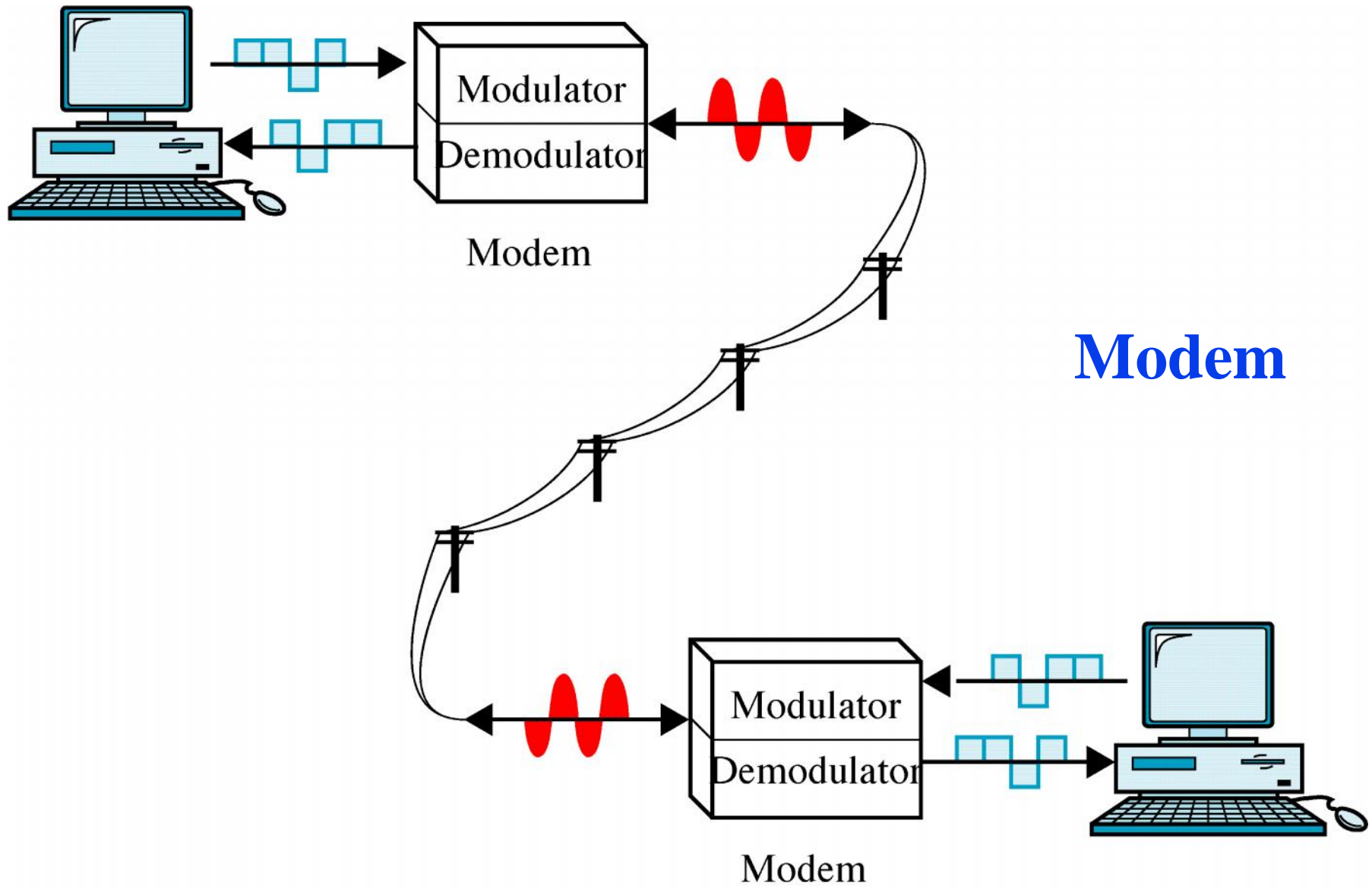


Figure 6-20

Bandwidth for Telephone Line

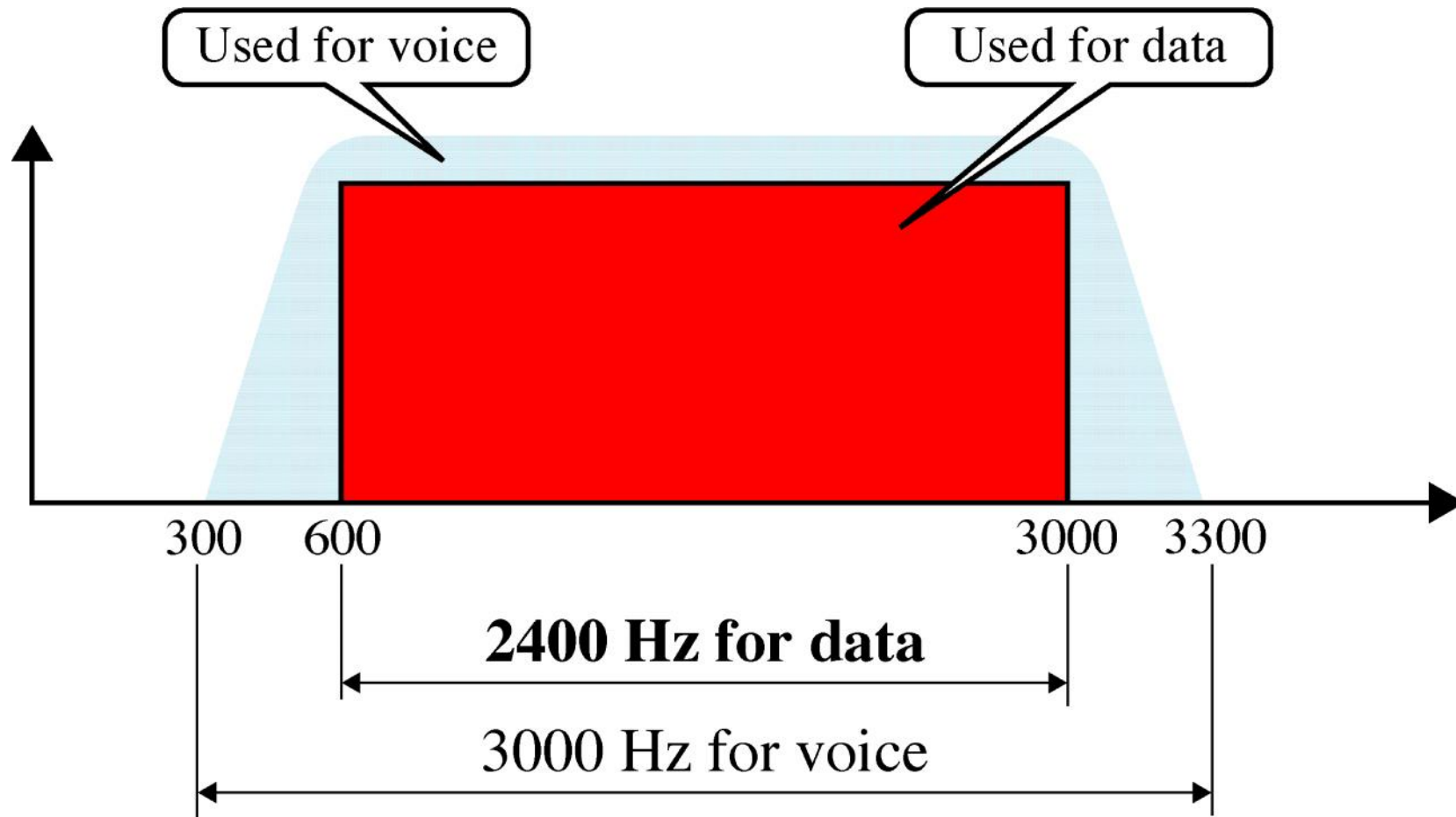


Figure 6-21

Baud Rate for Half-Duplex ASK

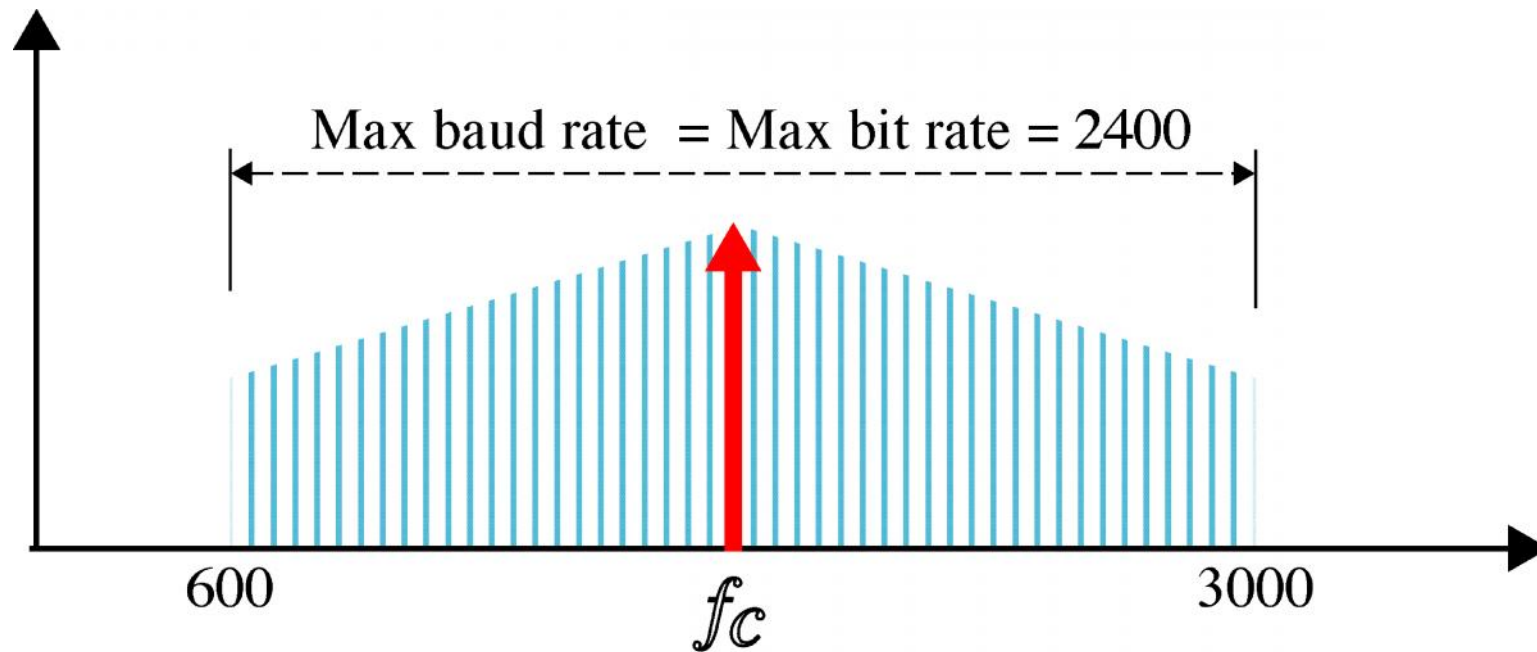


Figure 6-22

Baud Rate for Full-Duplex ASK

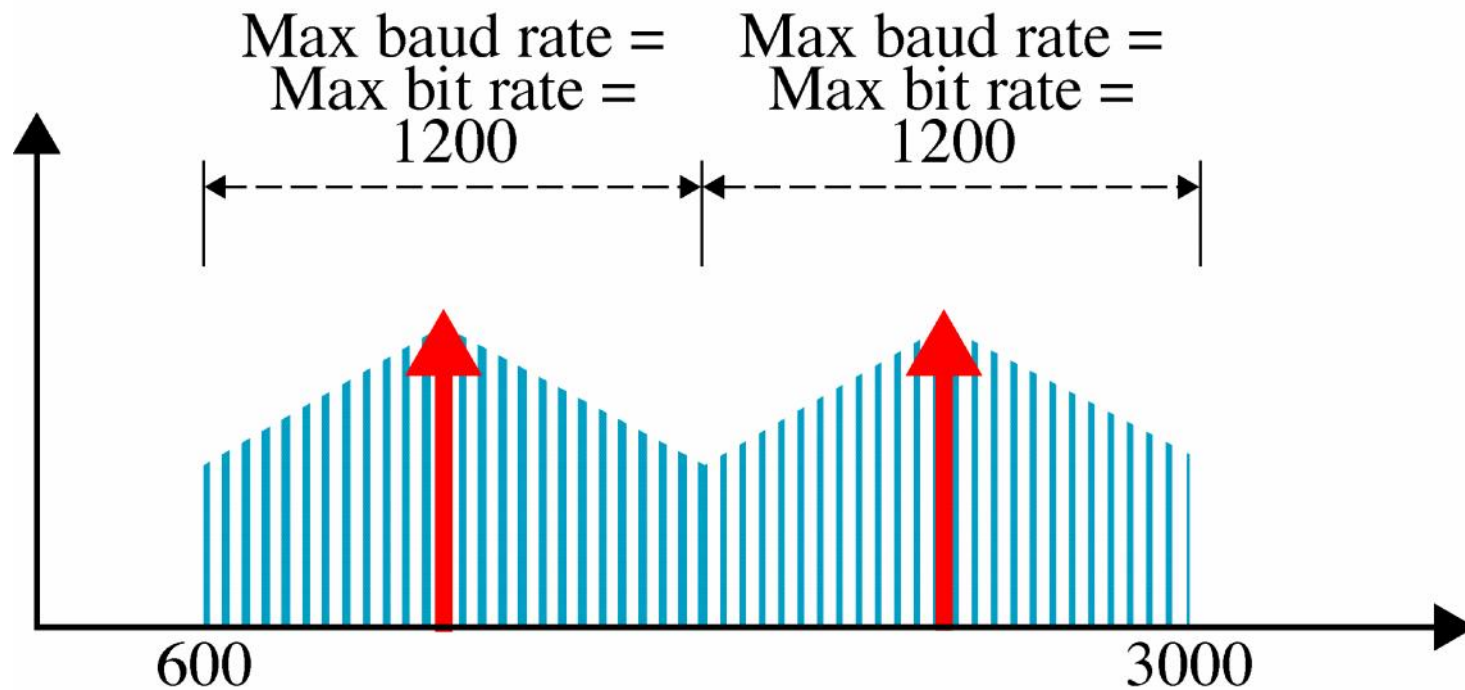


Figure 6-23

Baud Rate for Half-Duplex FSK

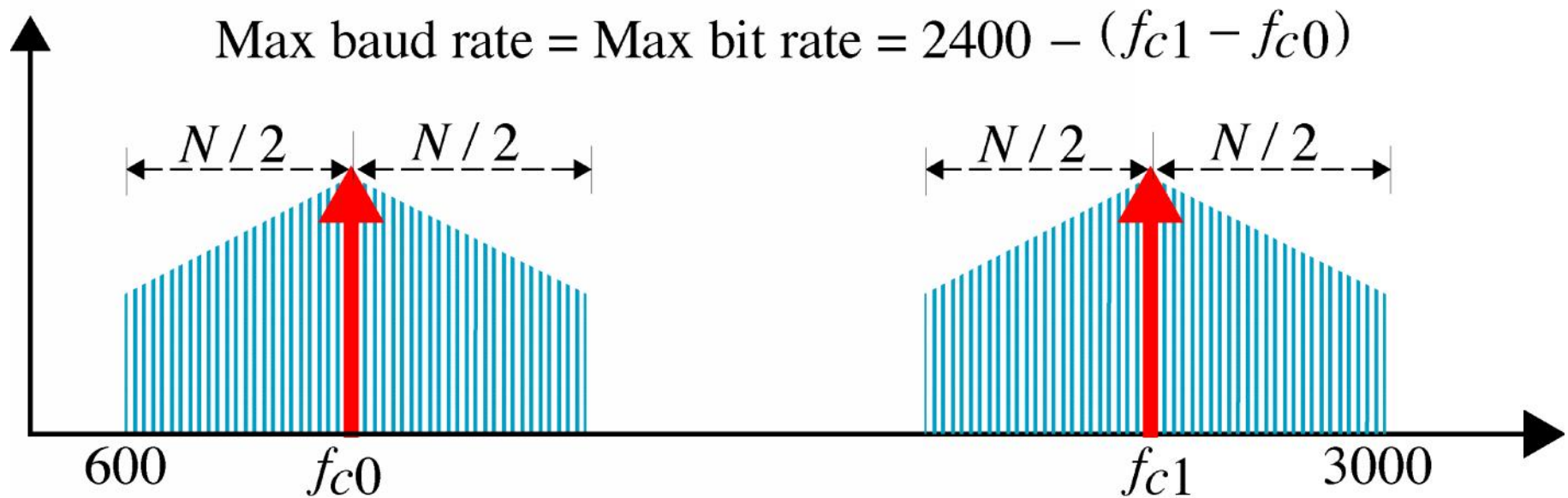


Figure 6-24

Baud Rate for Full-Duplex FSK

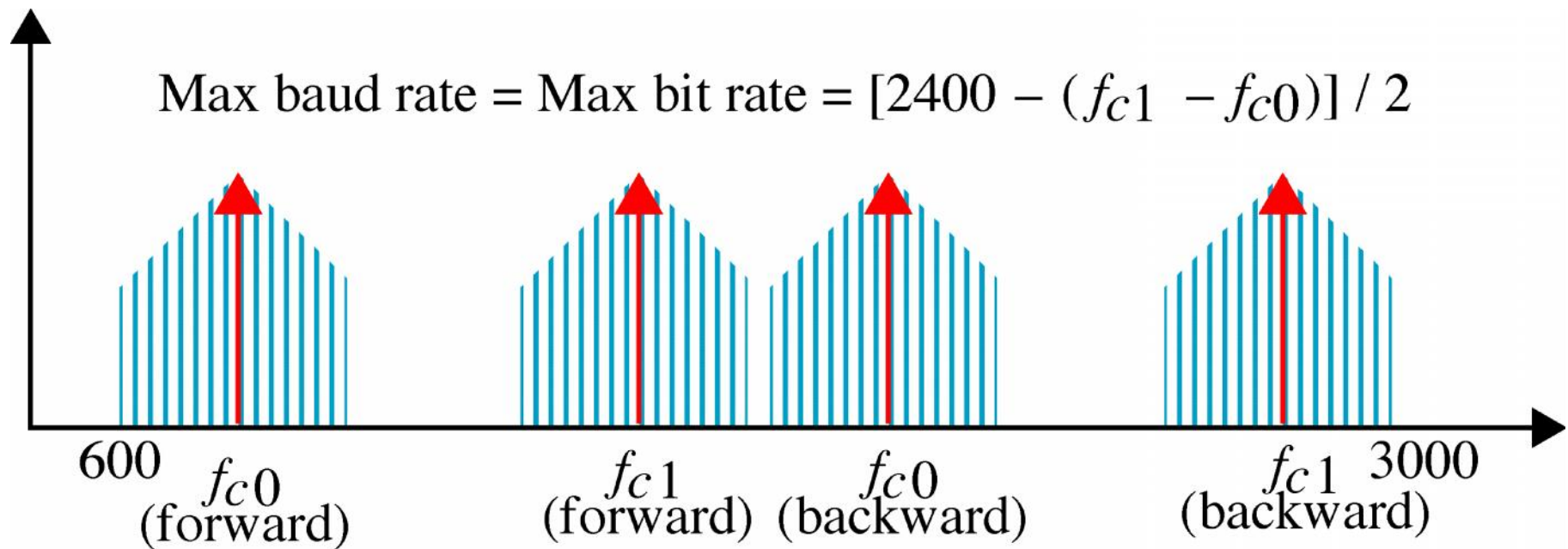


Figure 6-25

Bell Modems

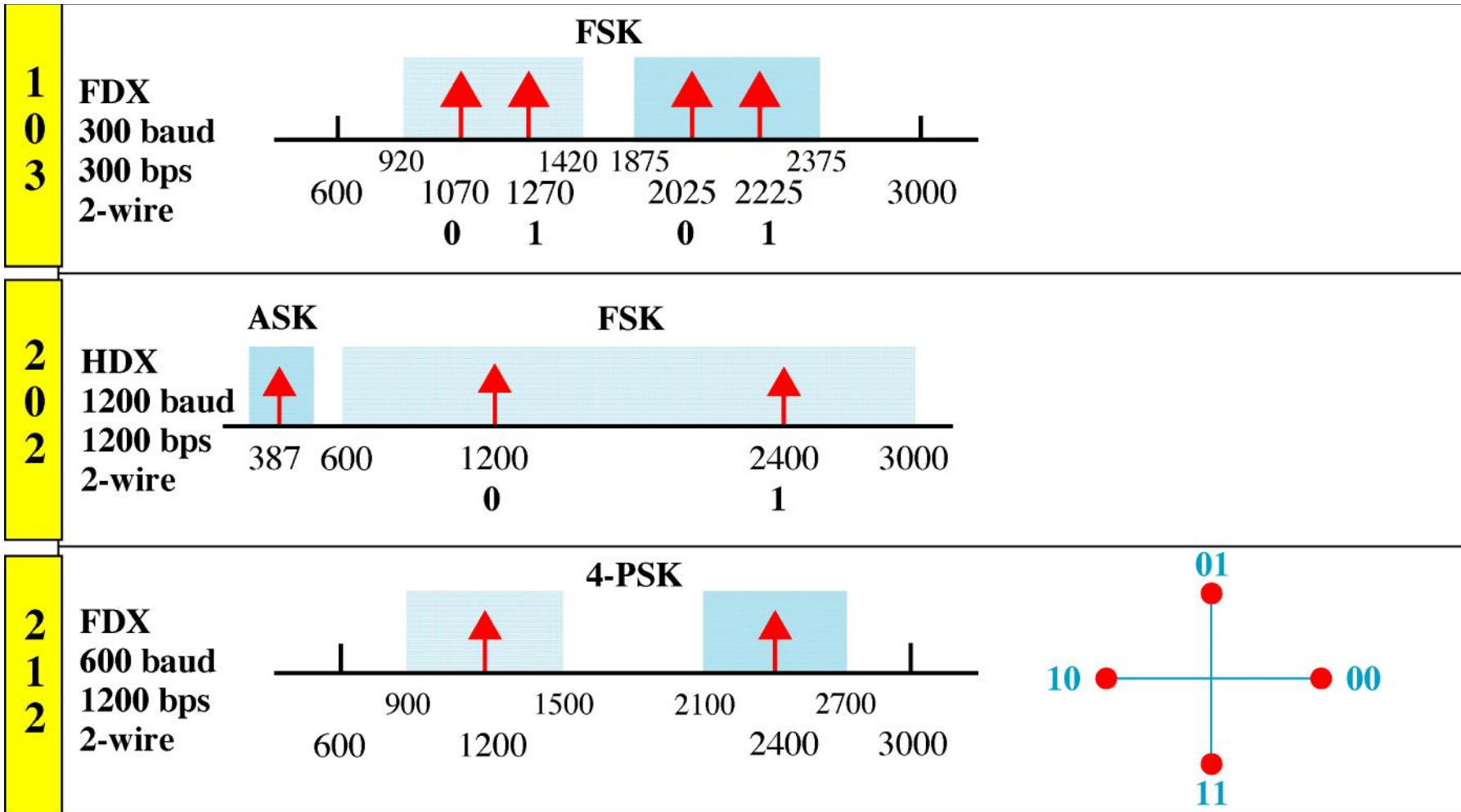


Figure 6-25-continued

Bell Modems

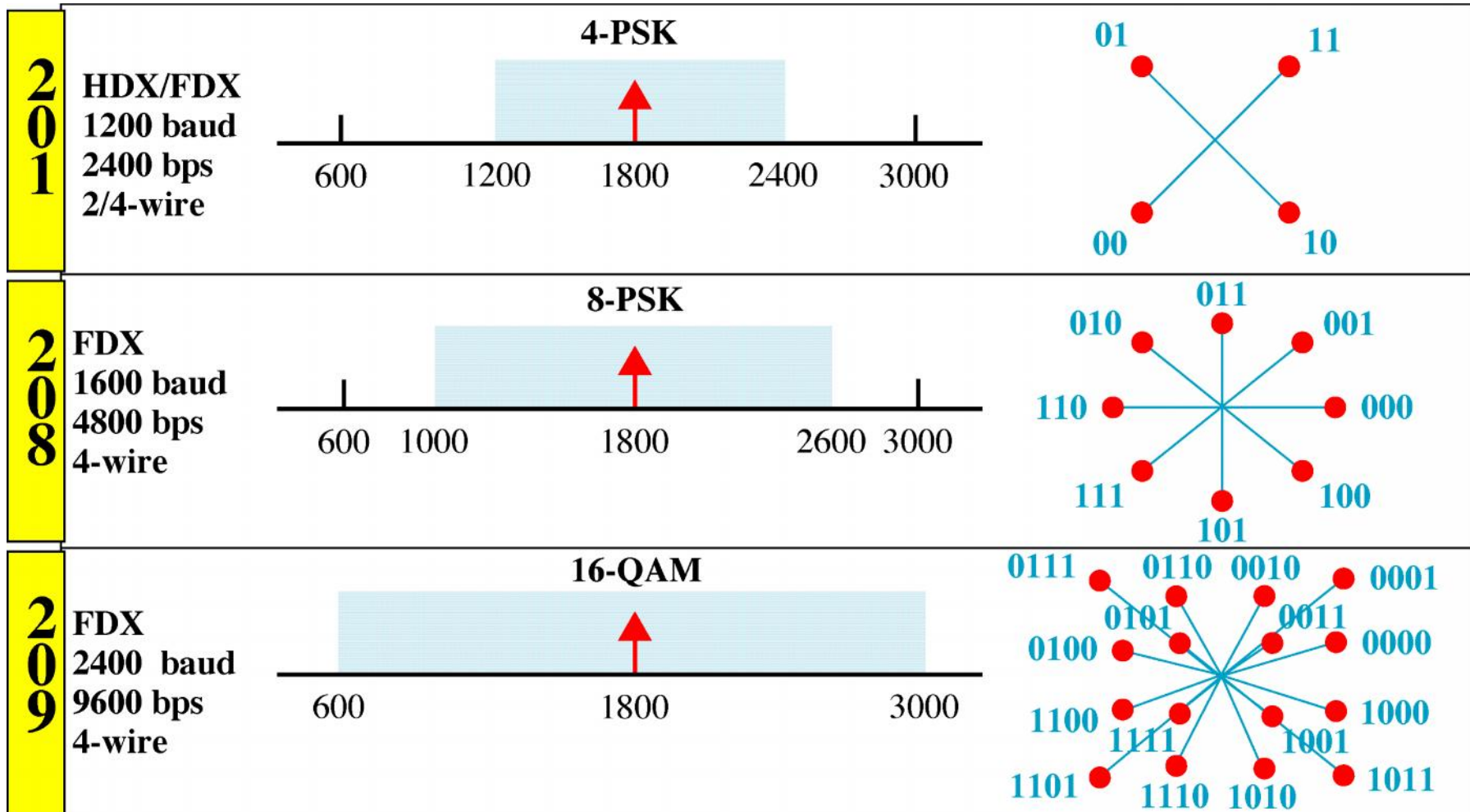


Figure 6-26

ITU Modems

Figure 6-26-continued

ITU Modems

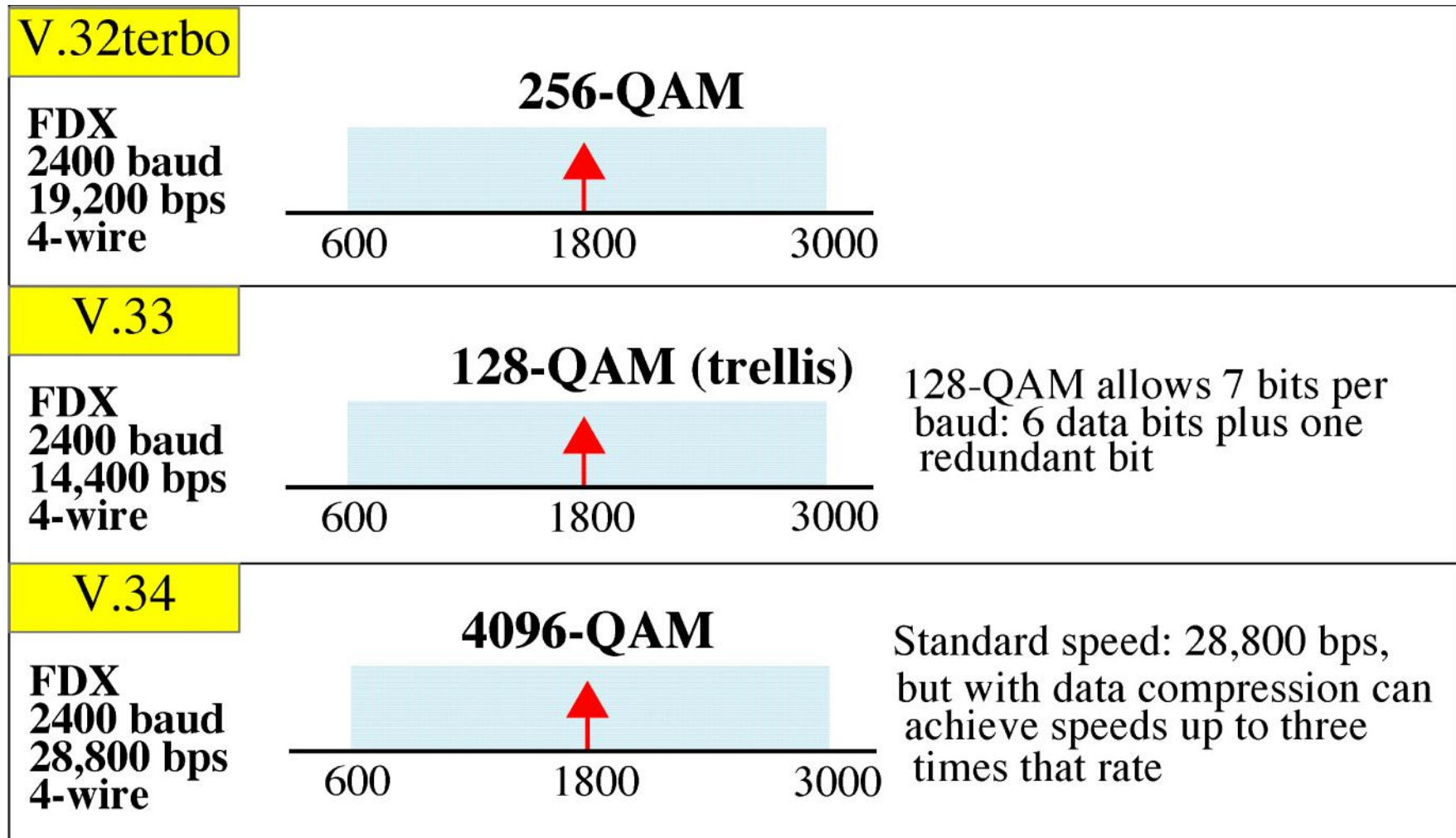


Figure 6-27

V.22bis 16-QAM Constellation

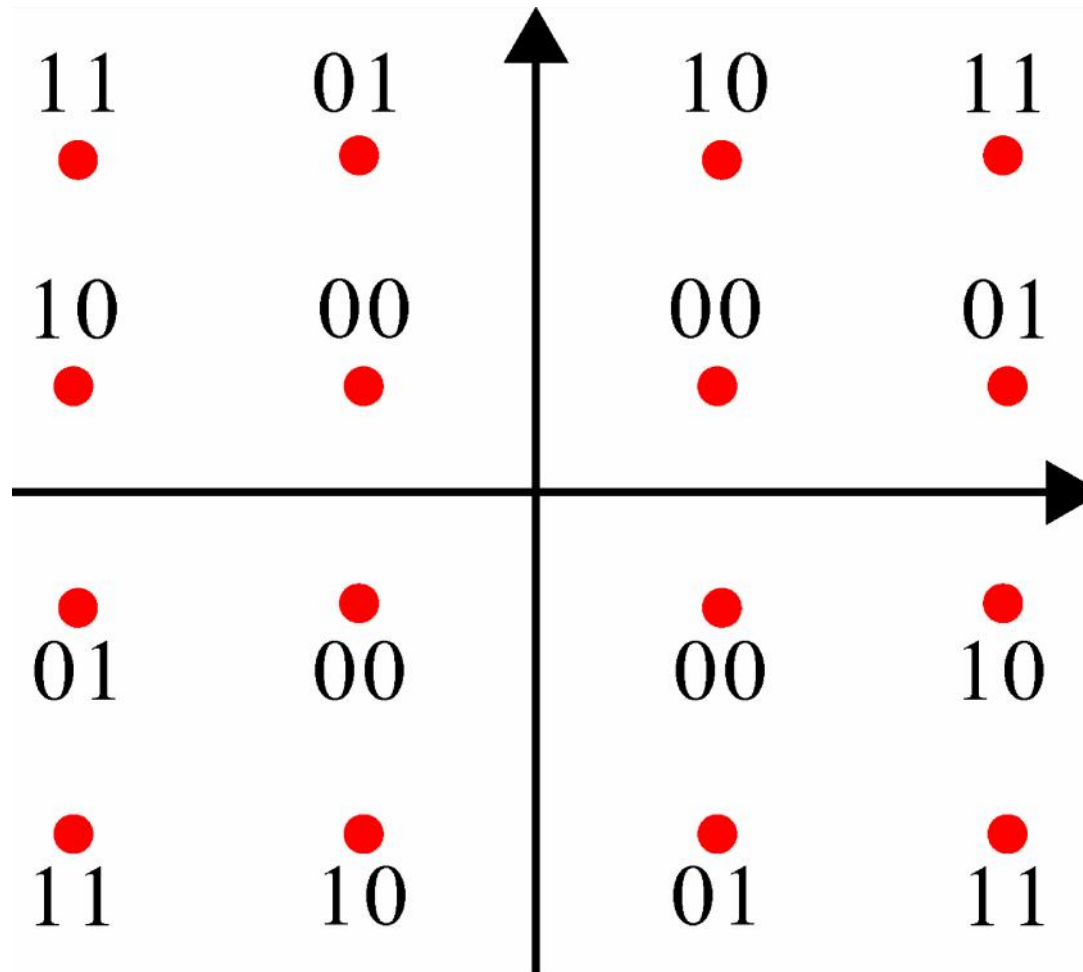


Figure 6-28

V.32 Constellation

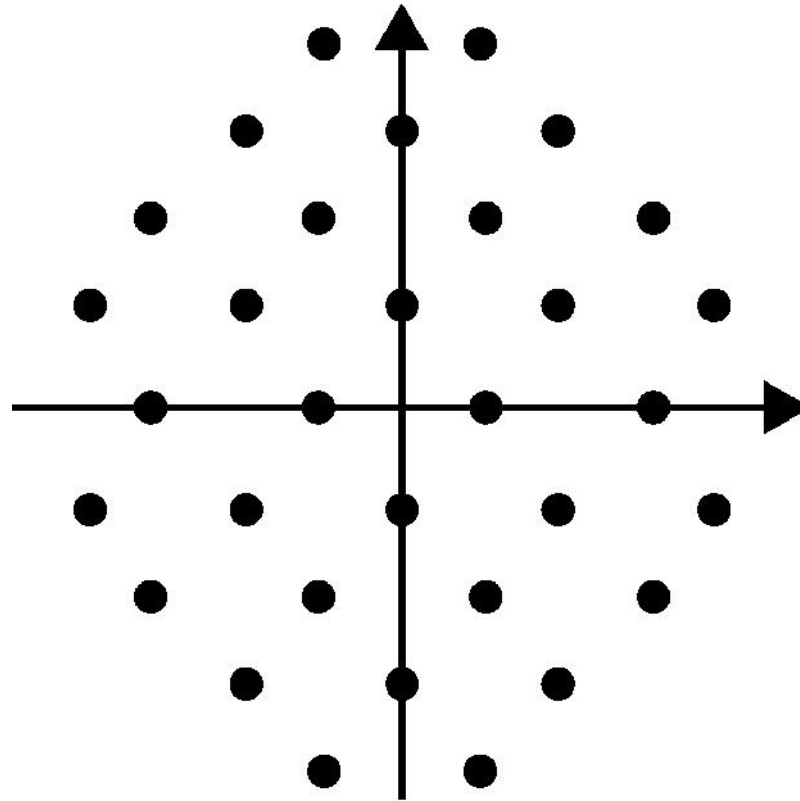
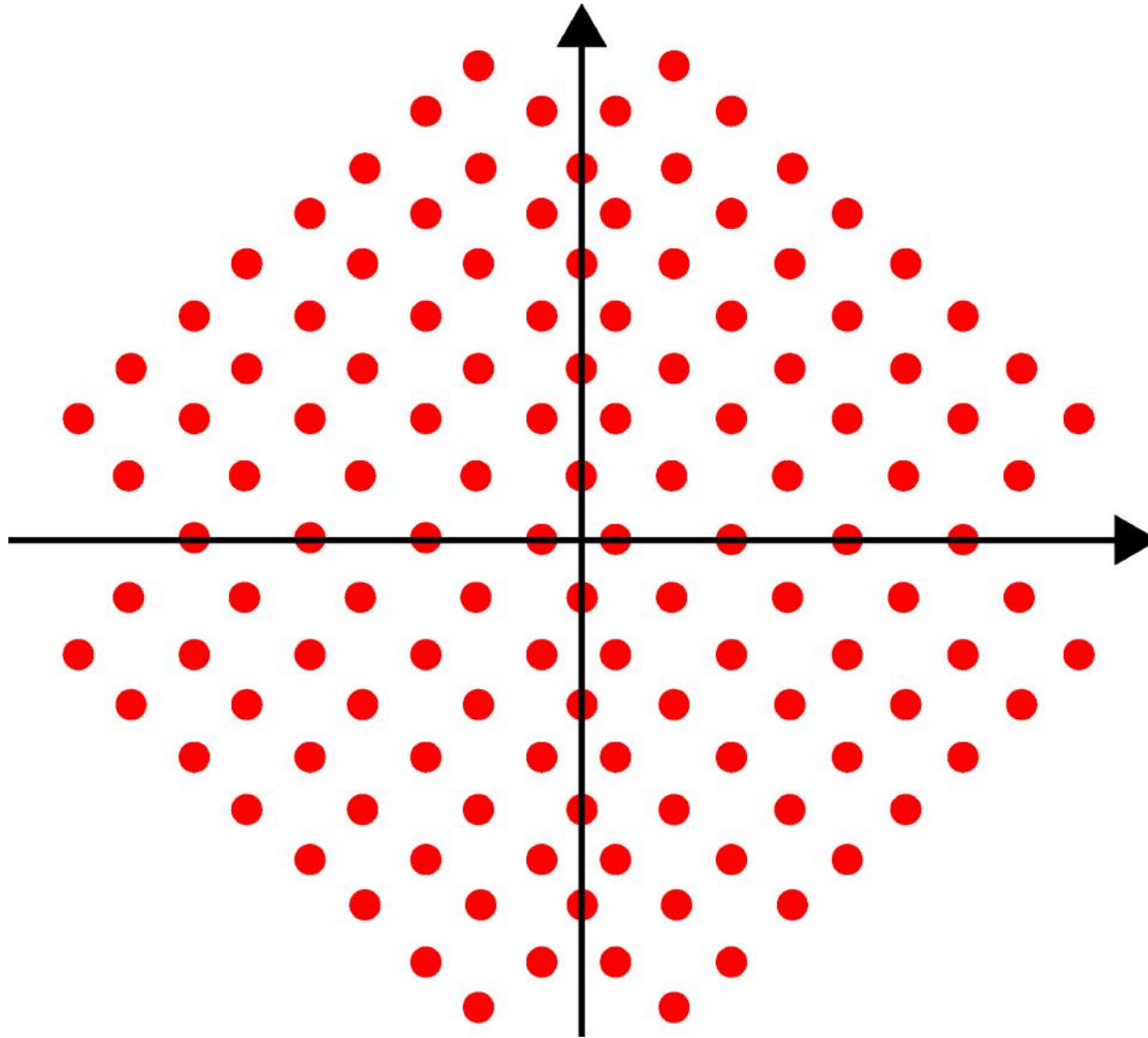


Figure 6-29

V.33 Constellation



COMPUTER NETWORKS – Unit 1

Topic 10

ANALOG TRANSMISSION

Analog Transmission

DIGITAL-TO-ANALOG CONVERSION

- Aspects of Digital-to-Analog Conversion
- Amplitude Shift Keying
- Frequency Shift Keying
- Phase Shift Keying
- Quadrature Amplitude Modulation

ANALOG-TO-ANALOG CONVERSION

- Amplitude Modulation
- Frequency Modulation
- Phase Modulation

Figure 5-22

Digital to Analog conversion



Figure 5-23

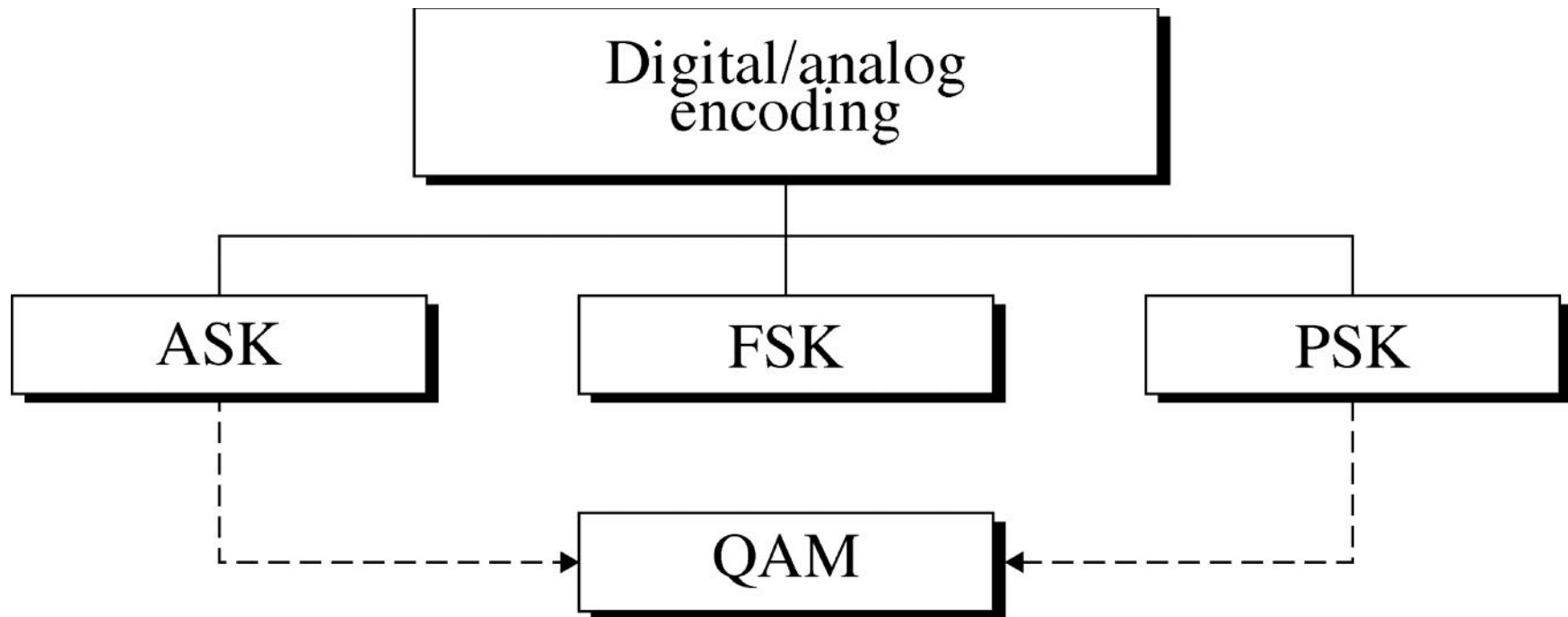


Figure 5-24

ASK

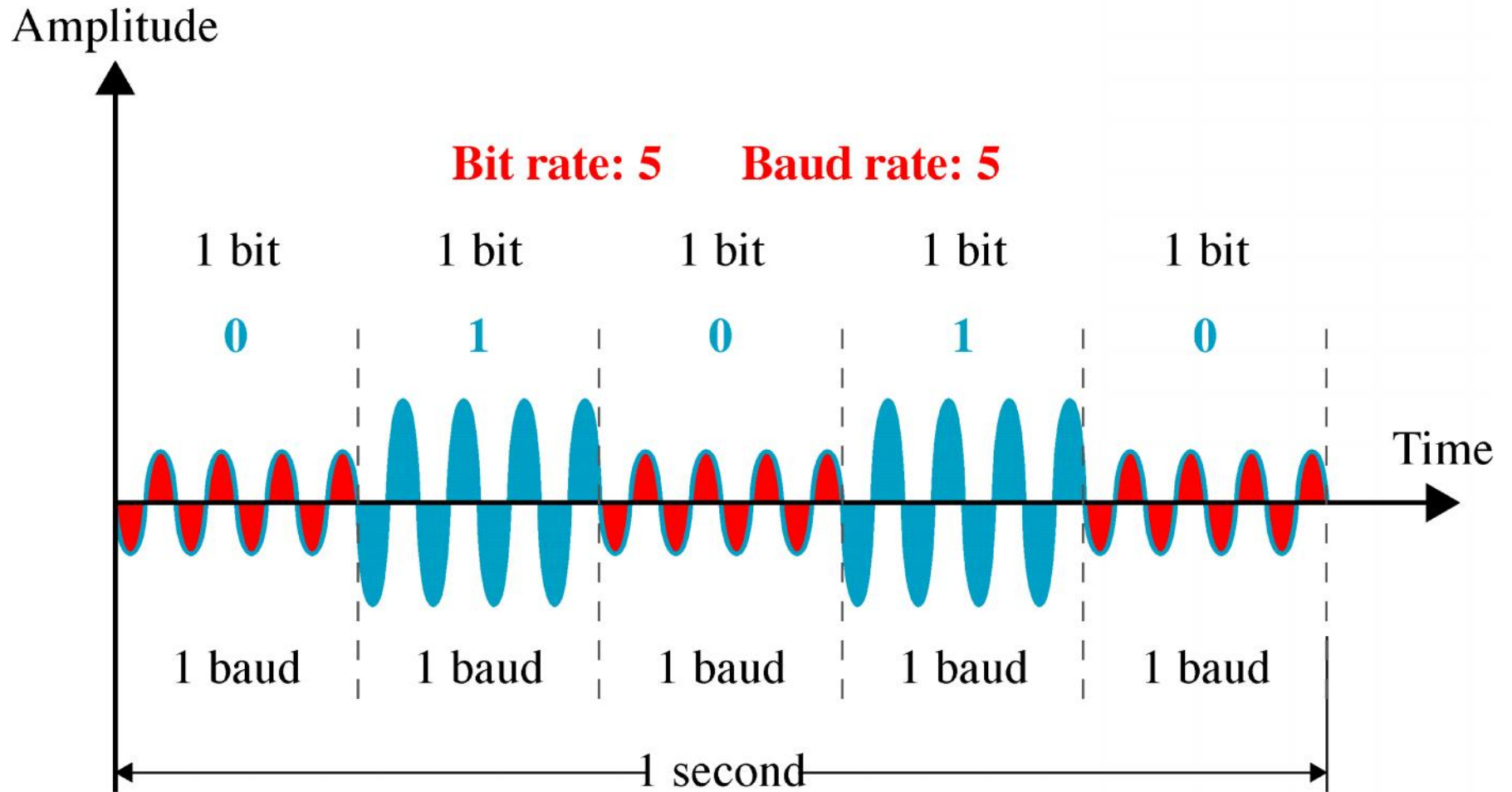


Figure 5-25

Bandwidth for ASK

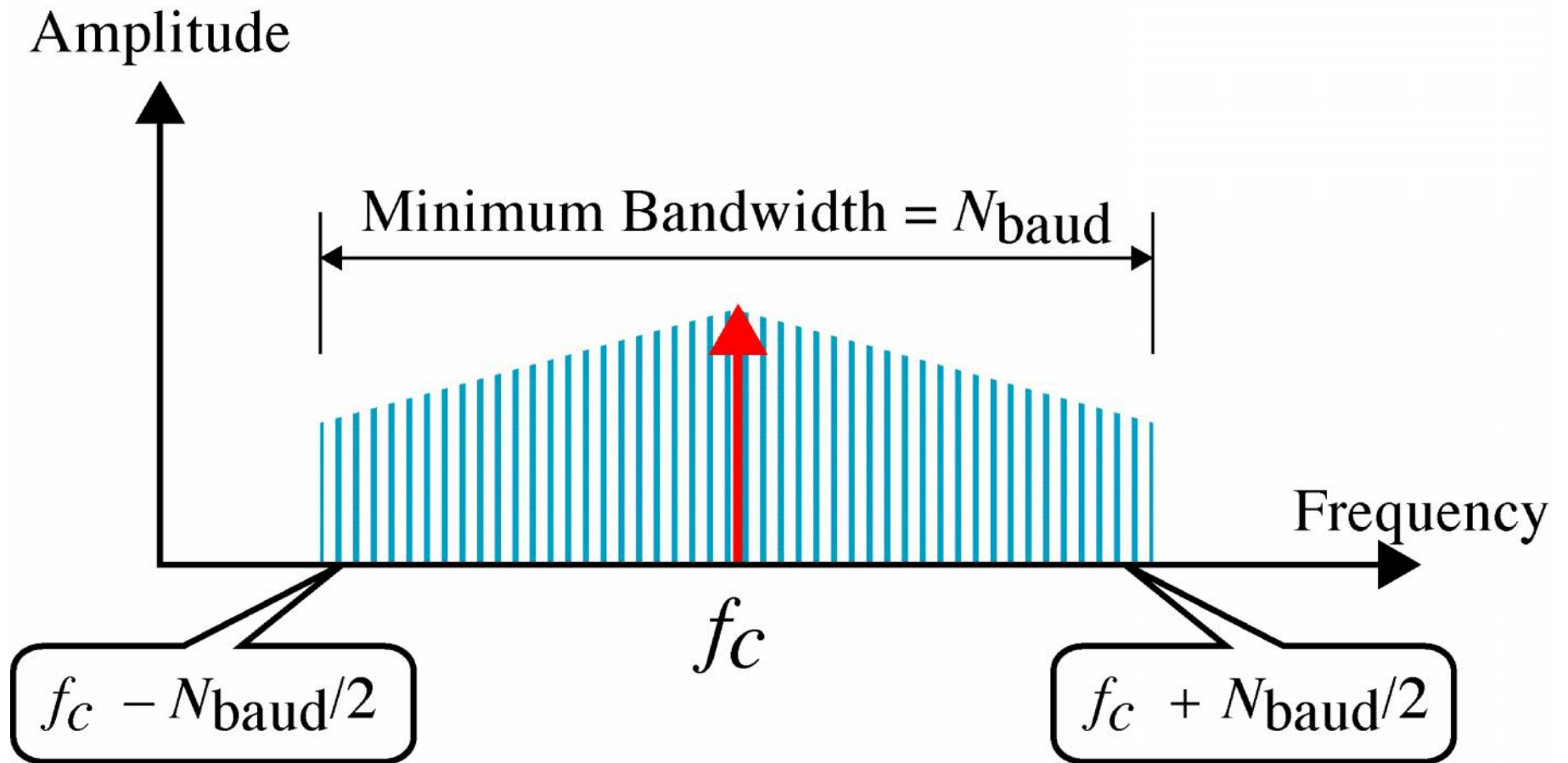


Figure 5-27

FSK

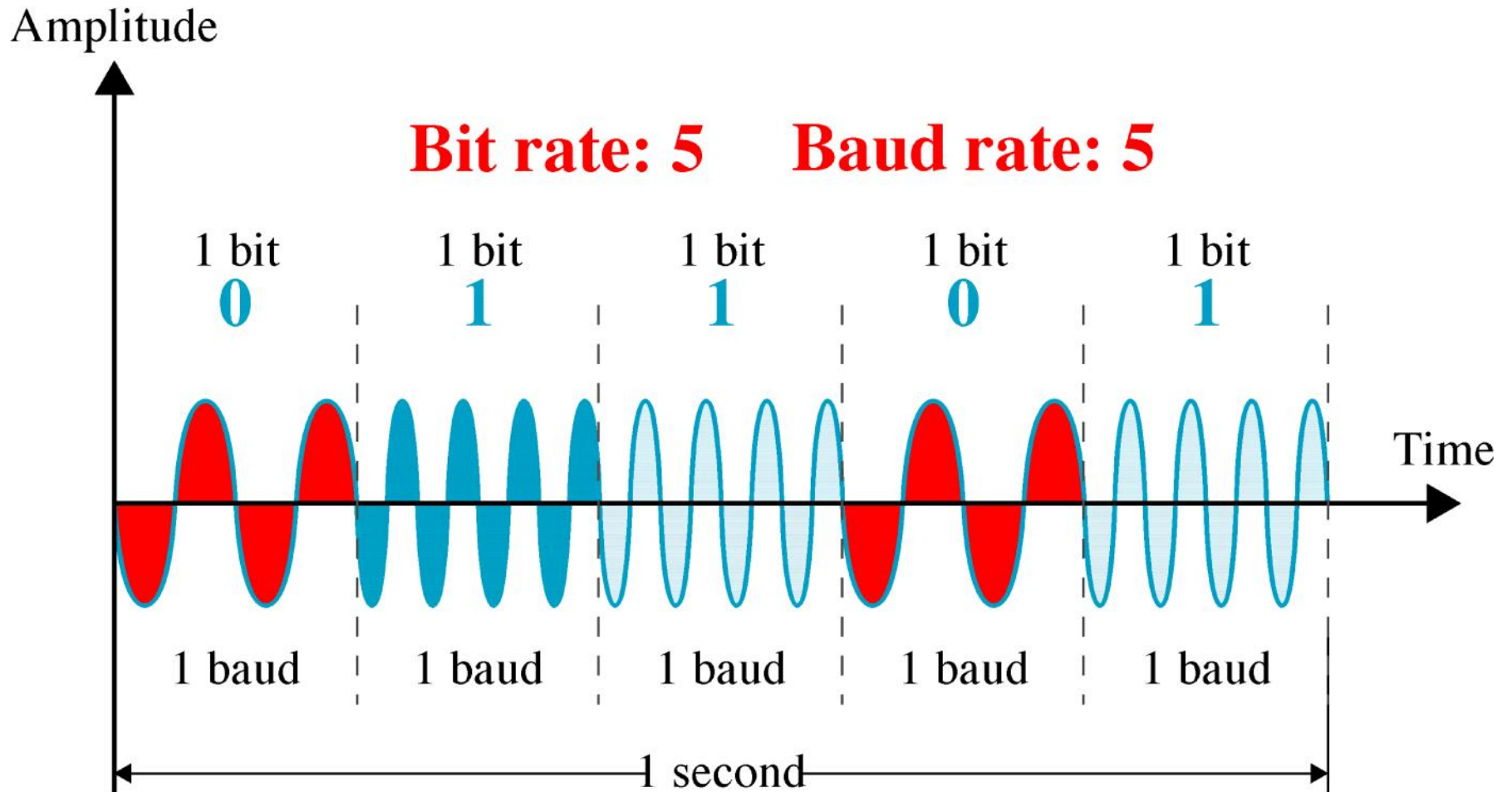


Figure 5-28

Bandwidth for FSK

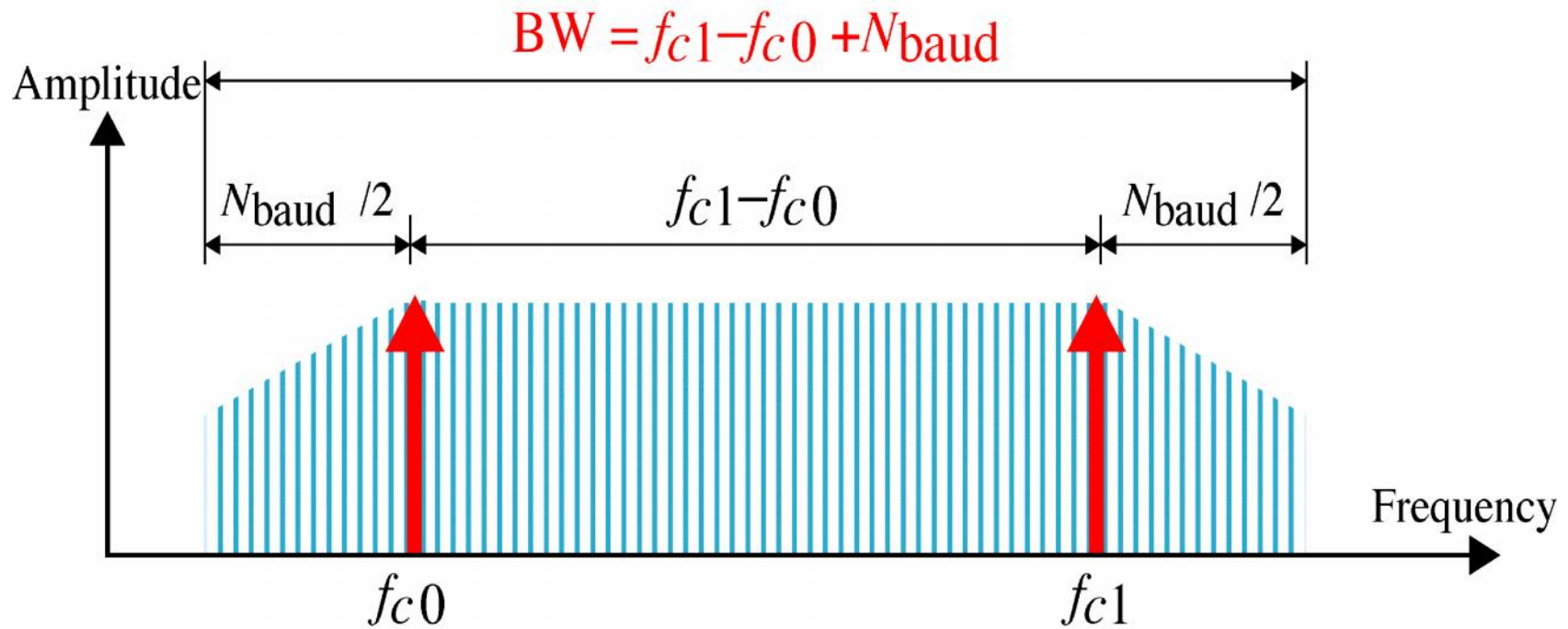


Figure 5-29

PSK

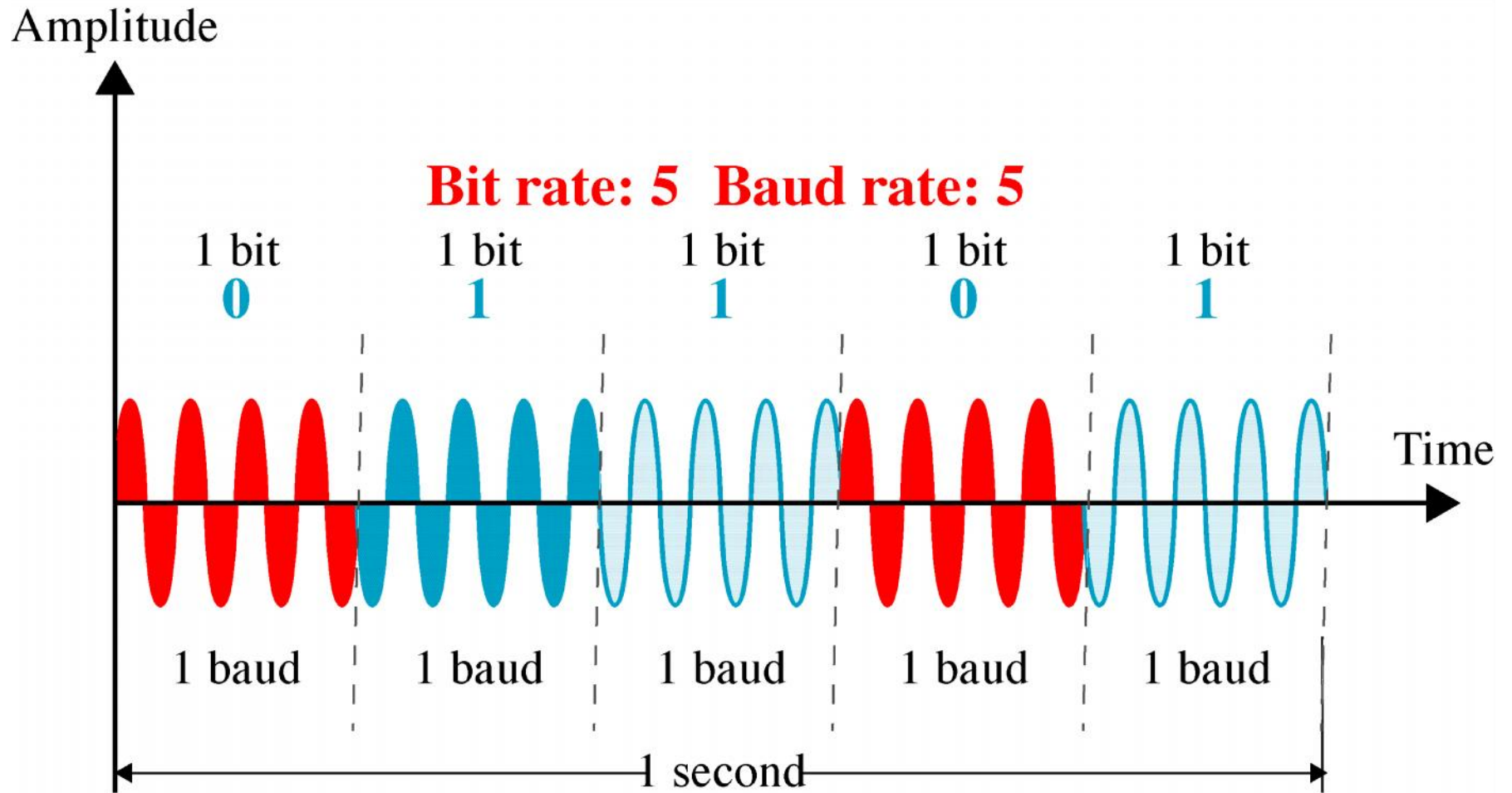
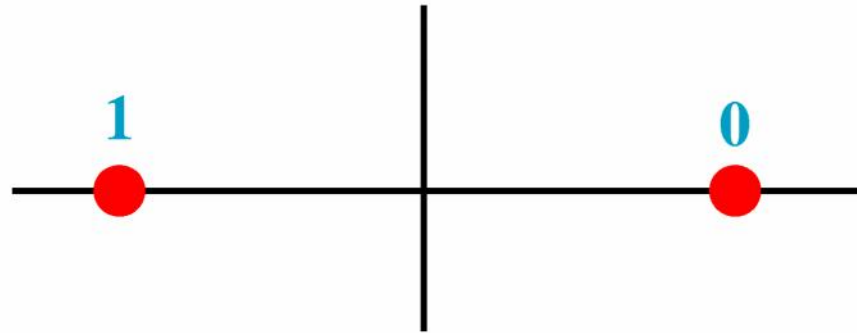


Figure 5-30

PSK Constellation

Bit	Phase
0	0
1	180

Bits



Constellation diagram

Figure 5-31

4-PSK

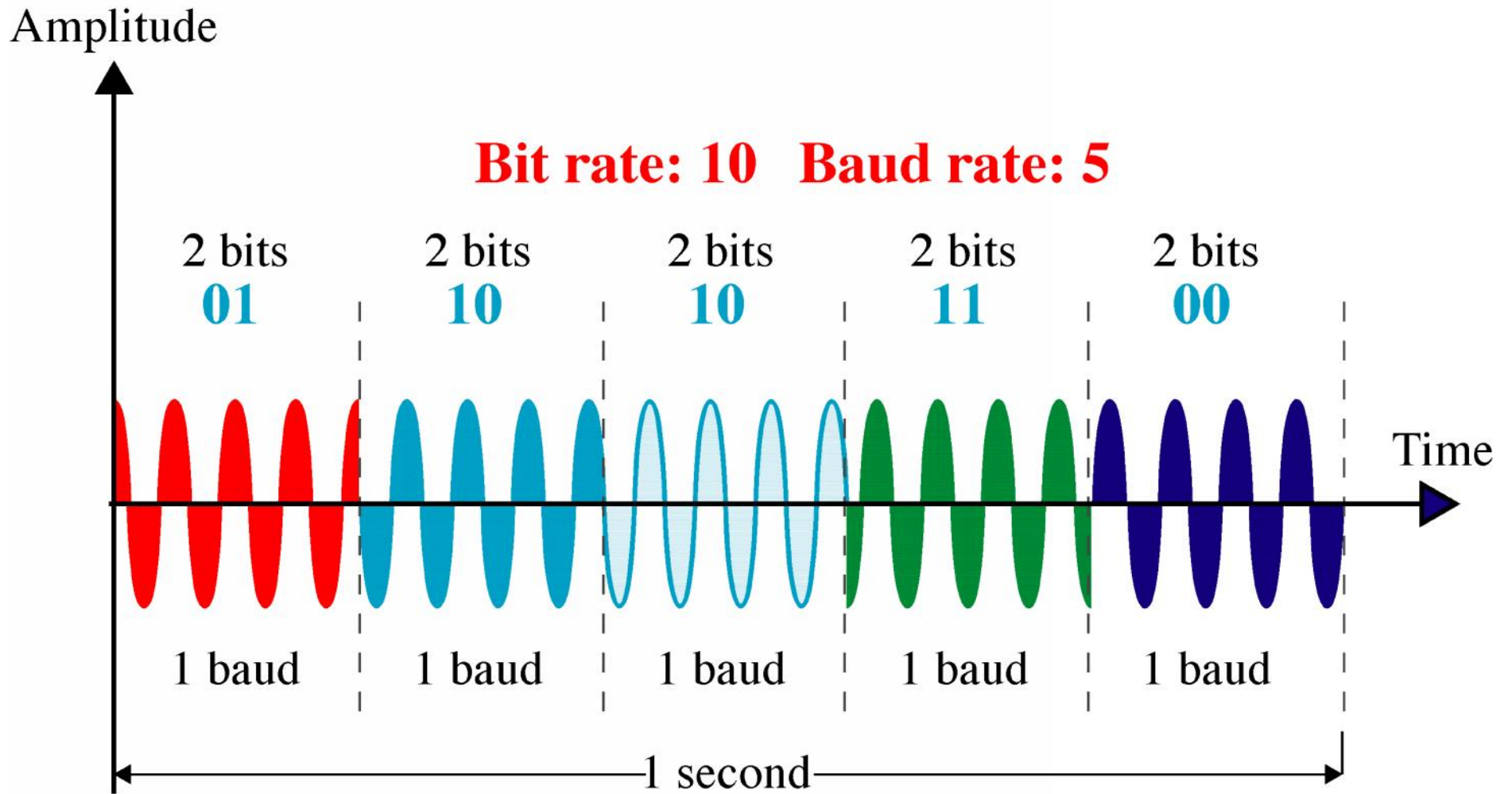
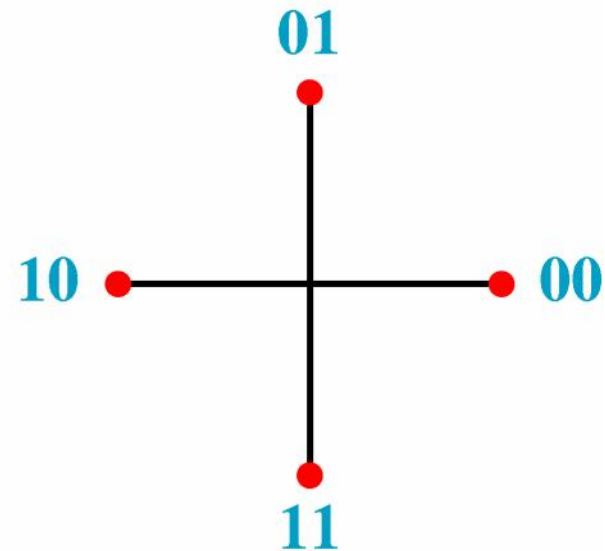


Figure 5-32

4-PSK Characteristics

Dibit	Phase
00	0
01	90
10	180
11	270

Dibit
(2 bits)



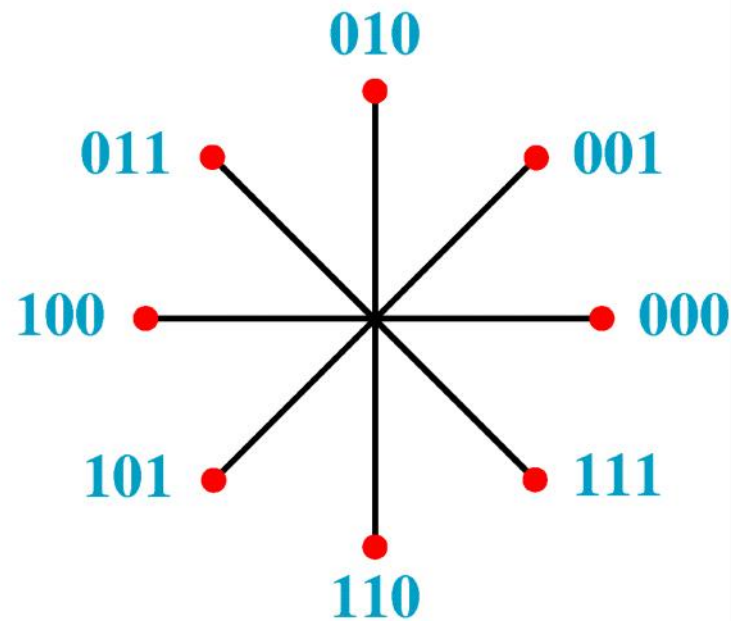
Constellation diagram

Figure 5-33

8-PSK Characteristics

Tribit	Phase
000	0
001	45
010	90
011	135
100	180
101	225
110	270
111	315

Tribits
(3 bits)



Constellation diagram

Figure 5-34

PSK Bandwidth

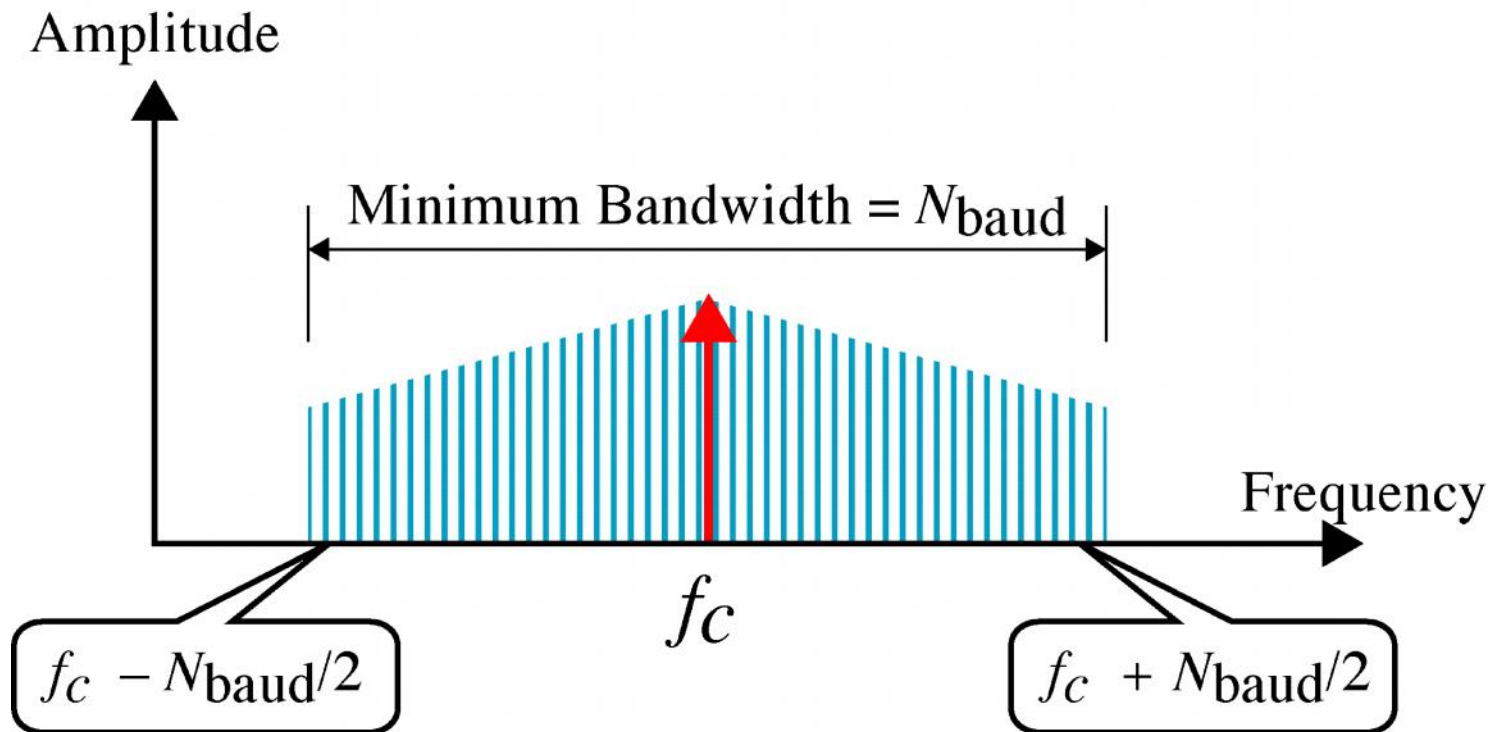
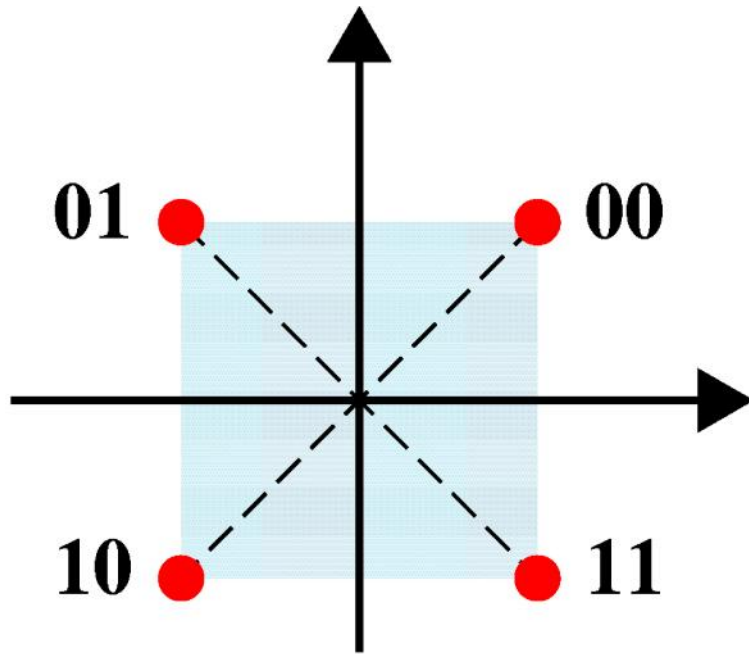


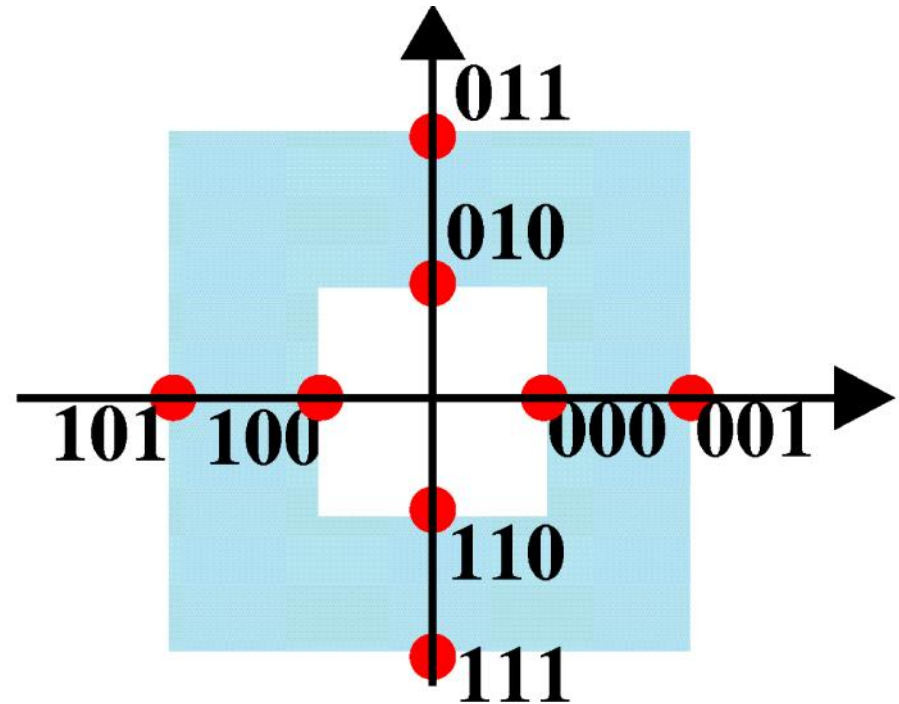
Figure 5-35

4-QAM and 8-QAM Constellations



4-QAM

1 amplitude, 4 phases



8-QAM

2 amplitudes, 4 phases

Figure 5-36

8-QAM Signal

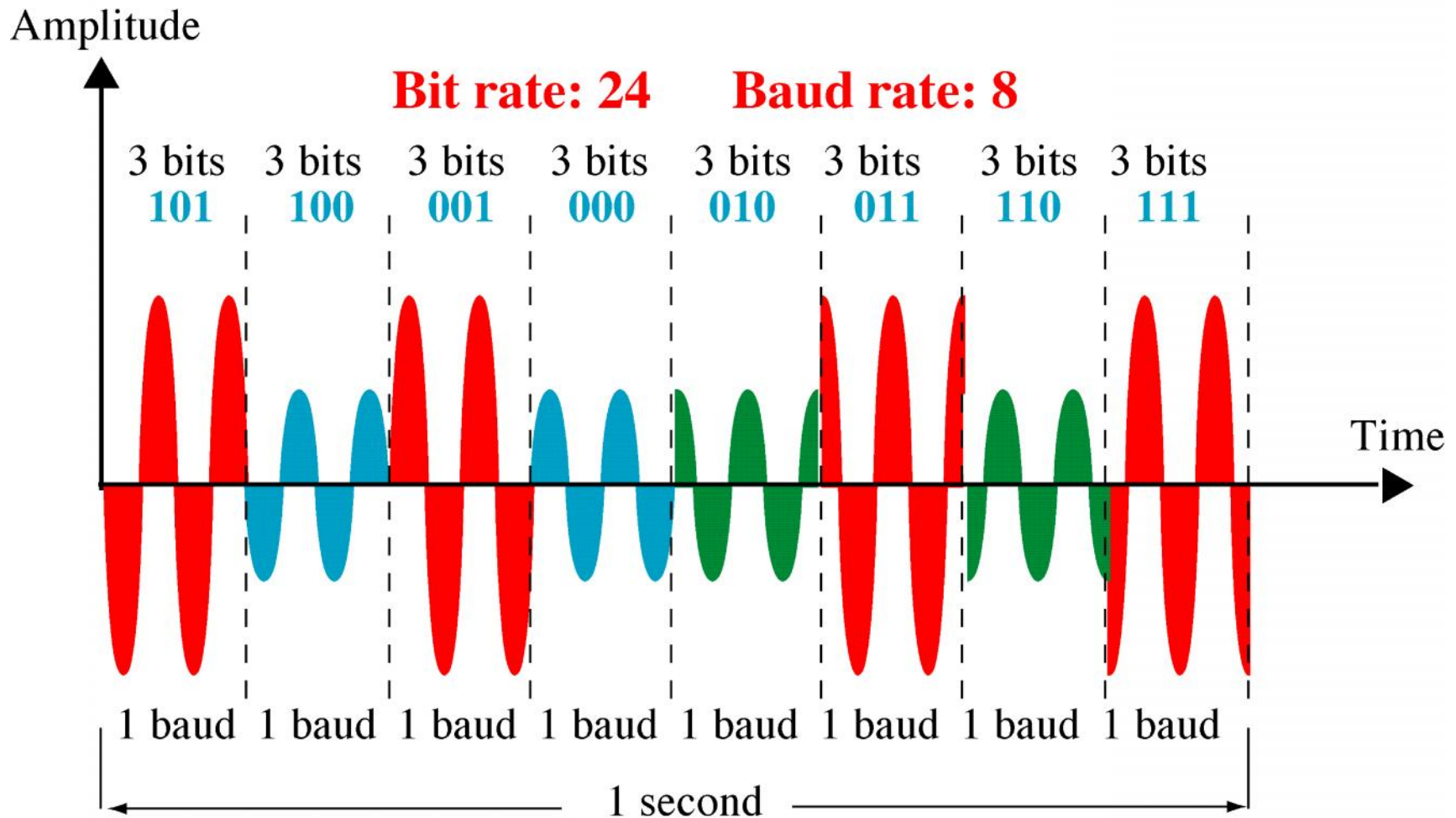
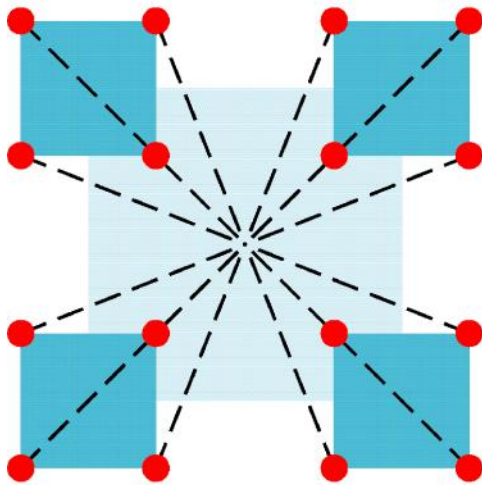


Figure 5-37

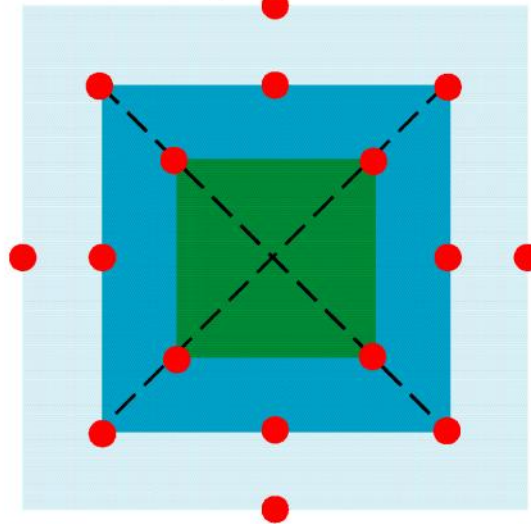
16-QAM Constellation

3 amplitudes,
12 phases



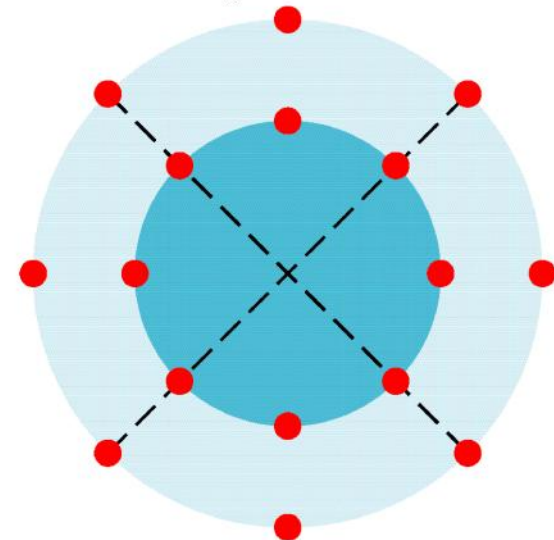
16-QAM

4 amplitudes,
8 phases



16-QAM

2 amplitudes,
8 phases



16-QAM

Figure 5-38

Bit Rate and Baud Rate

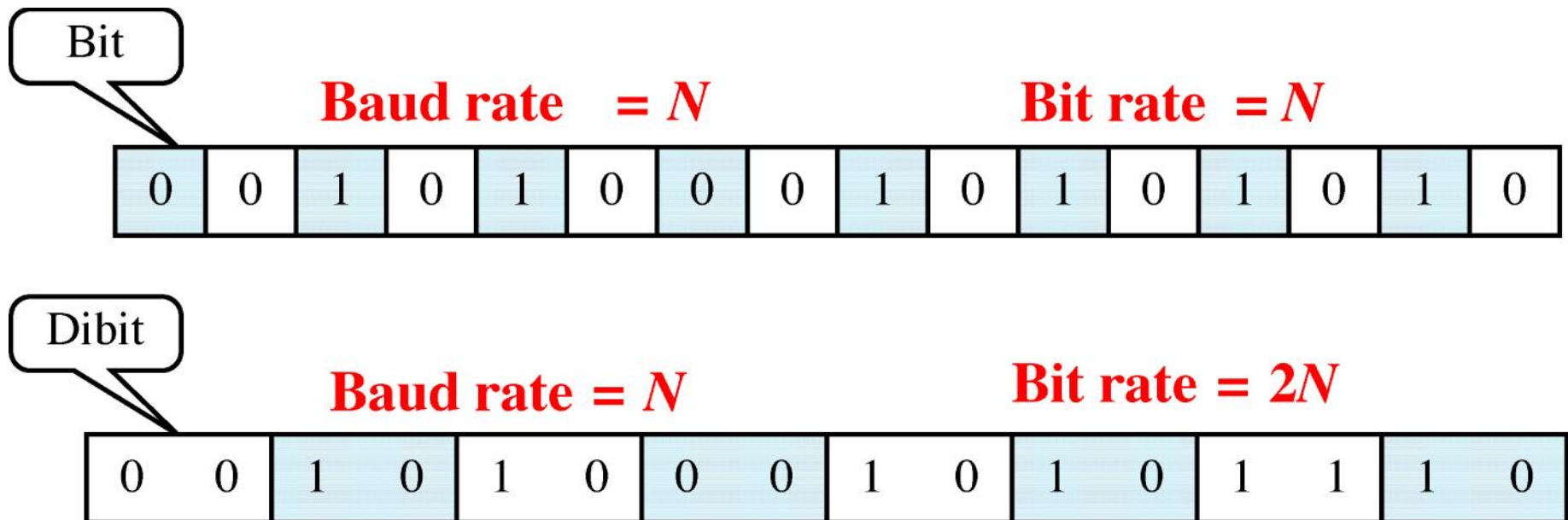


Figure 5-38-continued

Bit Rate and Baud Rate

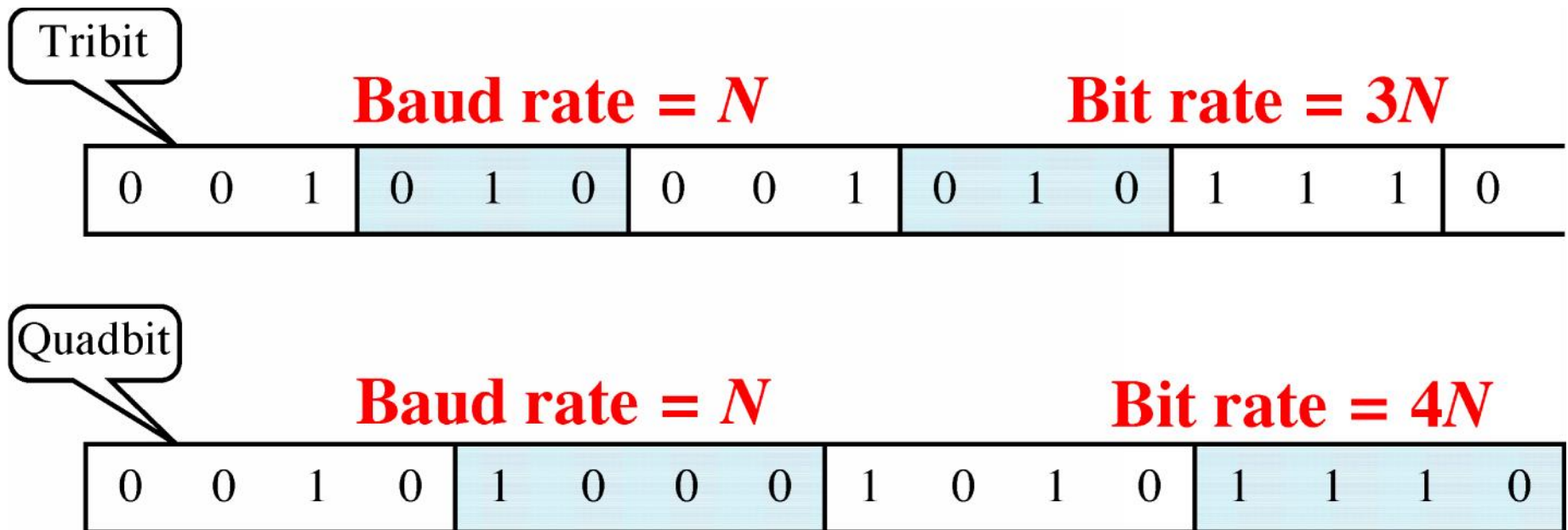


Figure 5-39

Analog to Analog Modulation



Figure 5-40

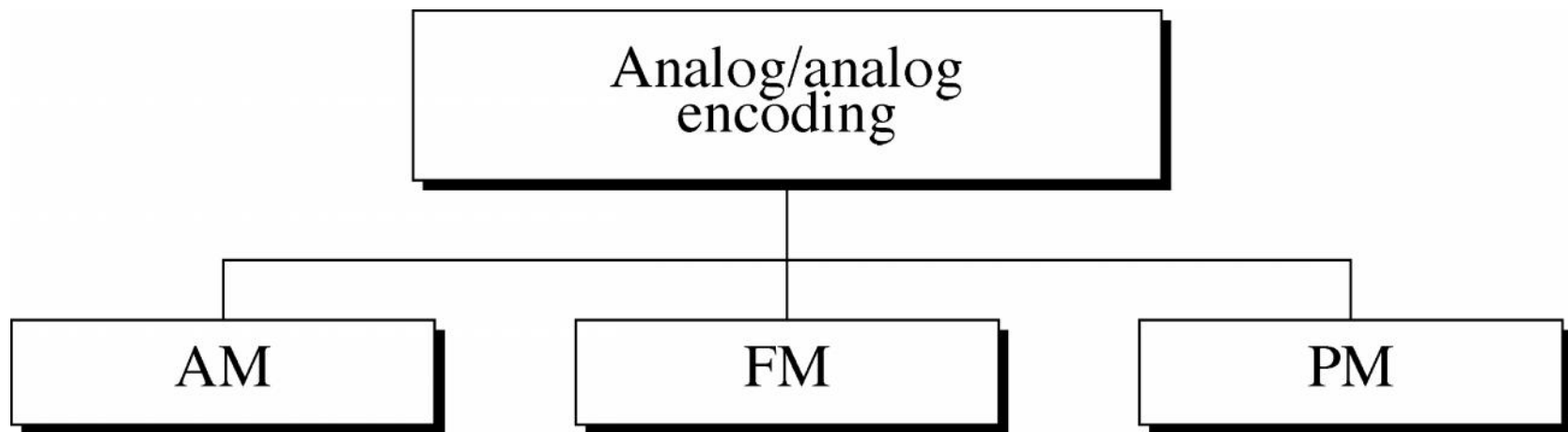


Figure 5-41

Amplitude Modulation

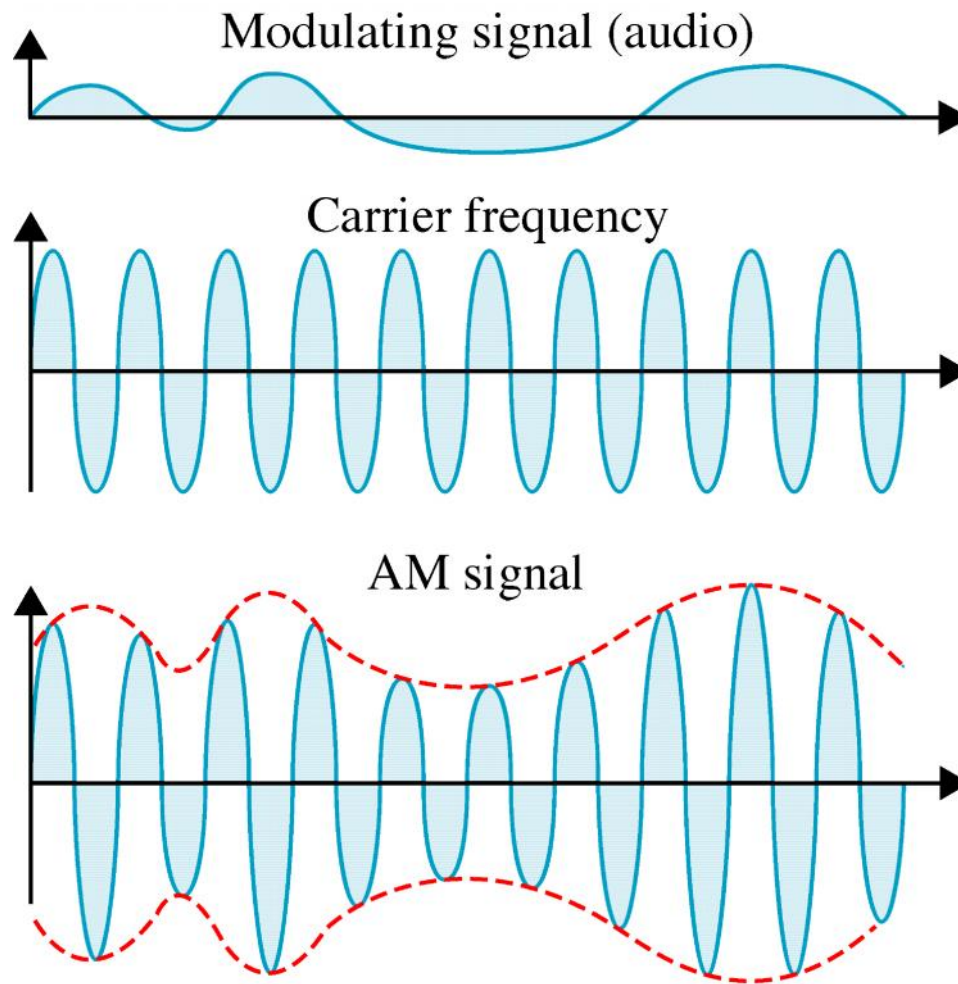


Figure 5-42

AM Bandwidth

BW_m = Bandwidth of the modulating signal (audio)

BW_t = Total bandwidth (radio)

f_c = Frequency of the carrier

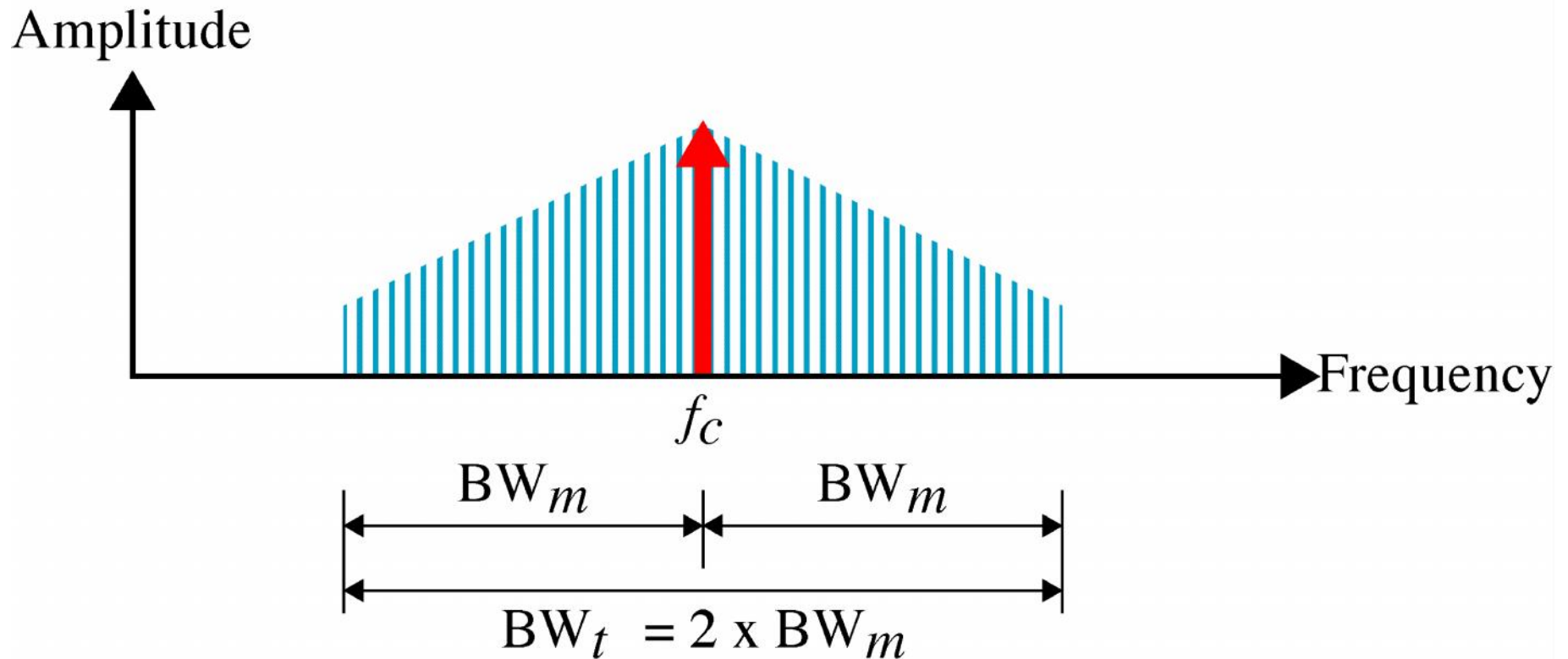


Figure 5-43

AM Band Allocation

f_c = Carrier frequency of the station

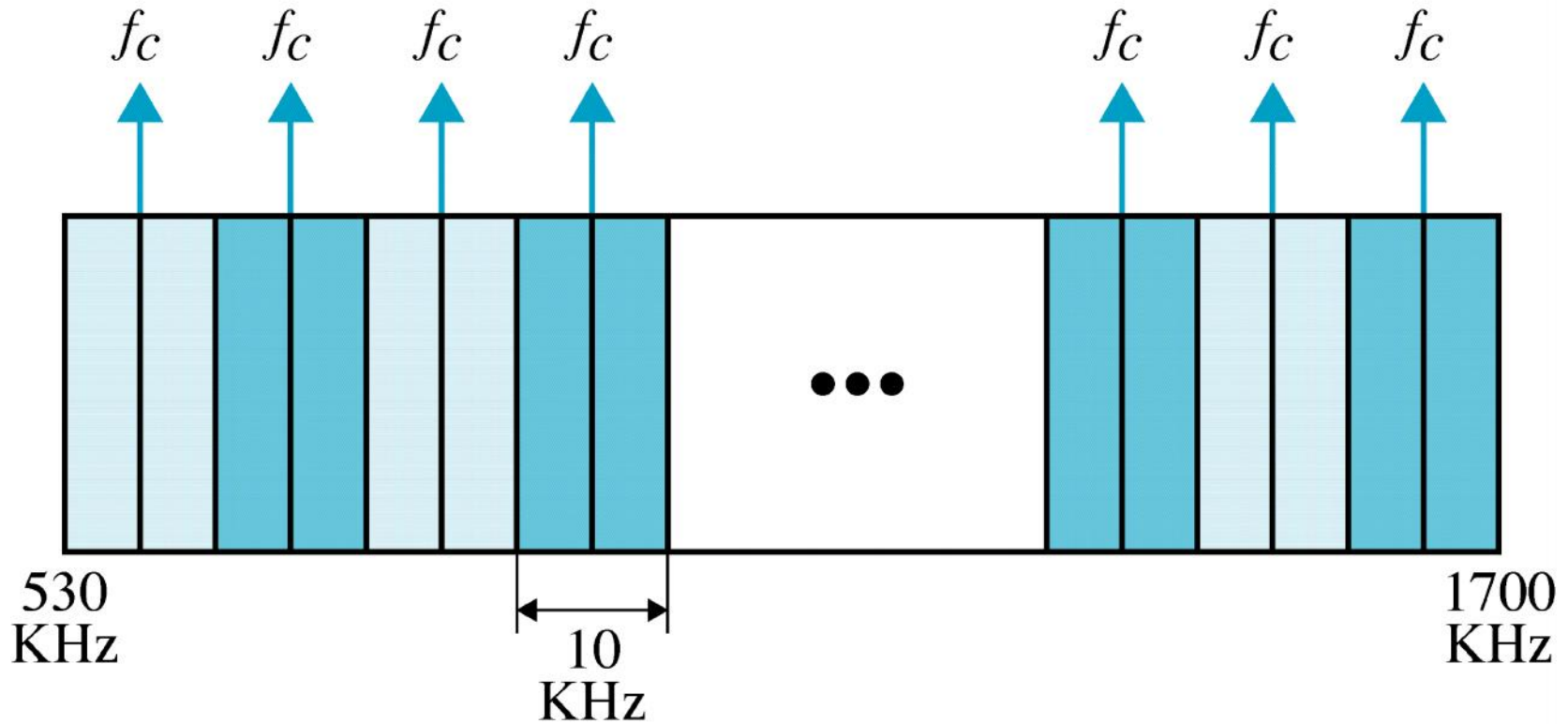


Figure 5-44

Frequency Modulation

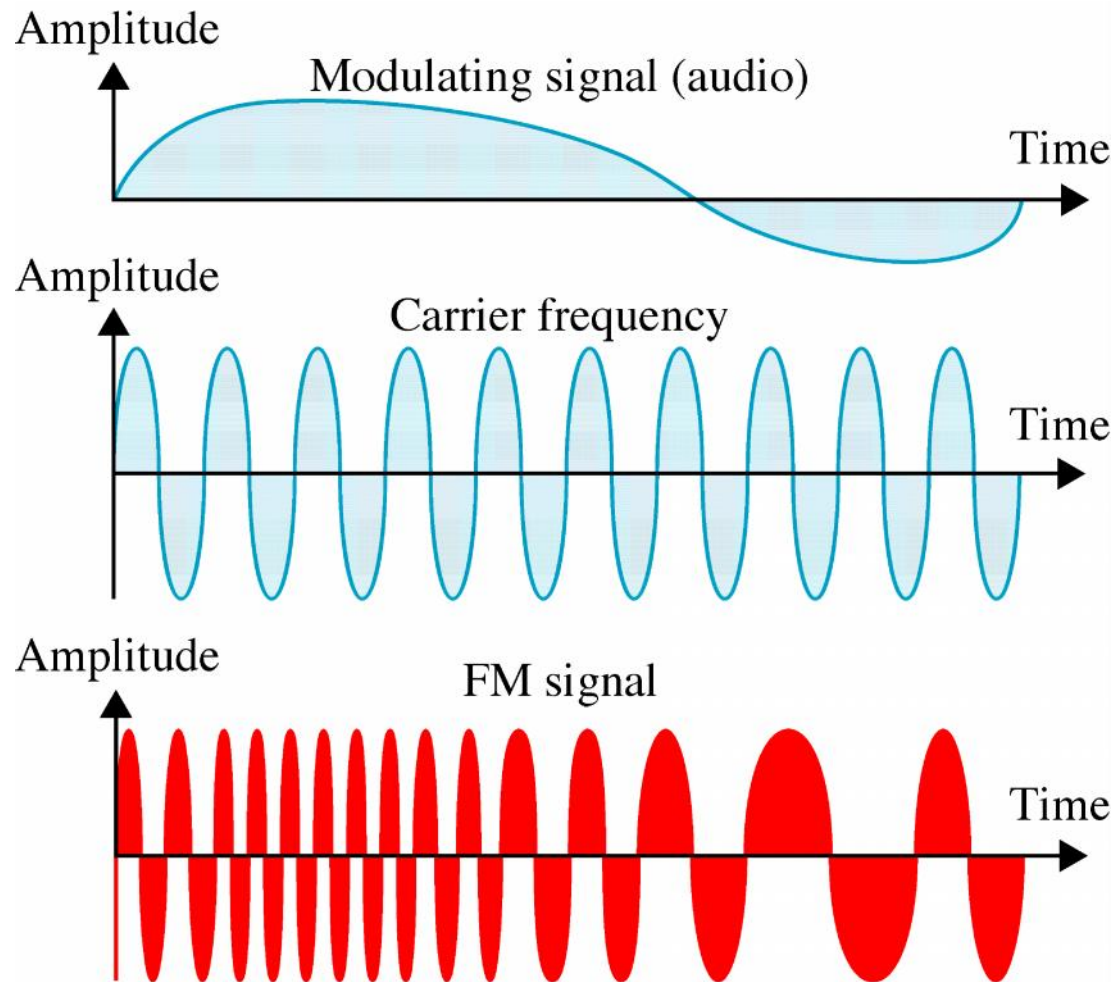


Figure 5-45

FM Bandwidth

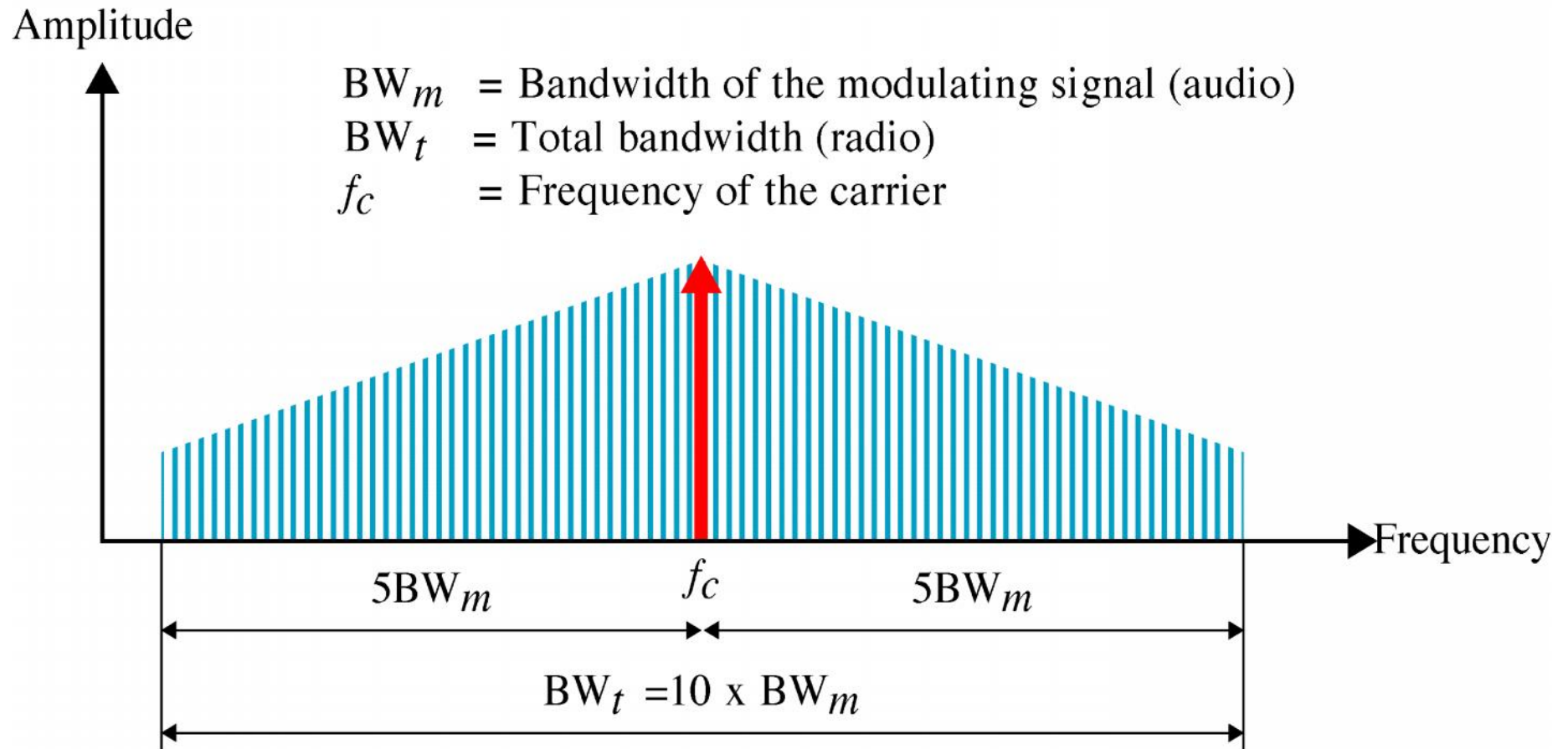
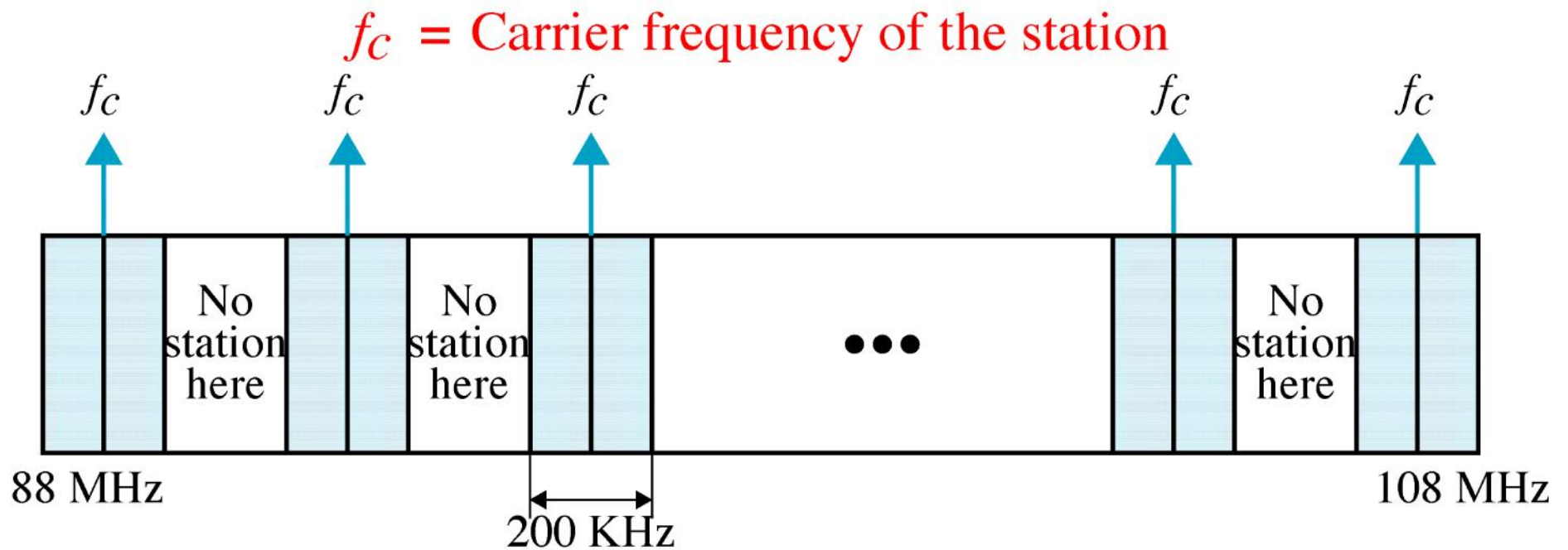


Figure 5-46

FM Band Allocation



COMPUTER NETWORKS – Unit 1

Topic 11

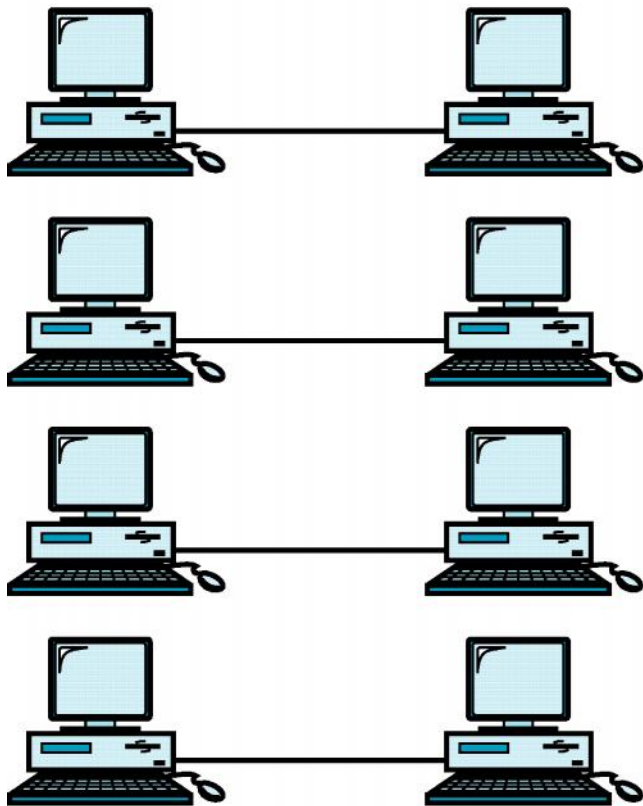
MULTIPLEXING

Multiplexing

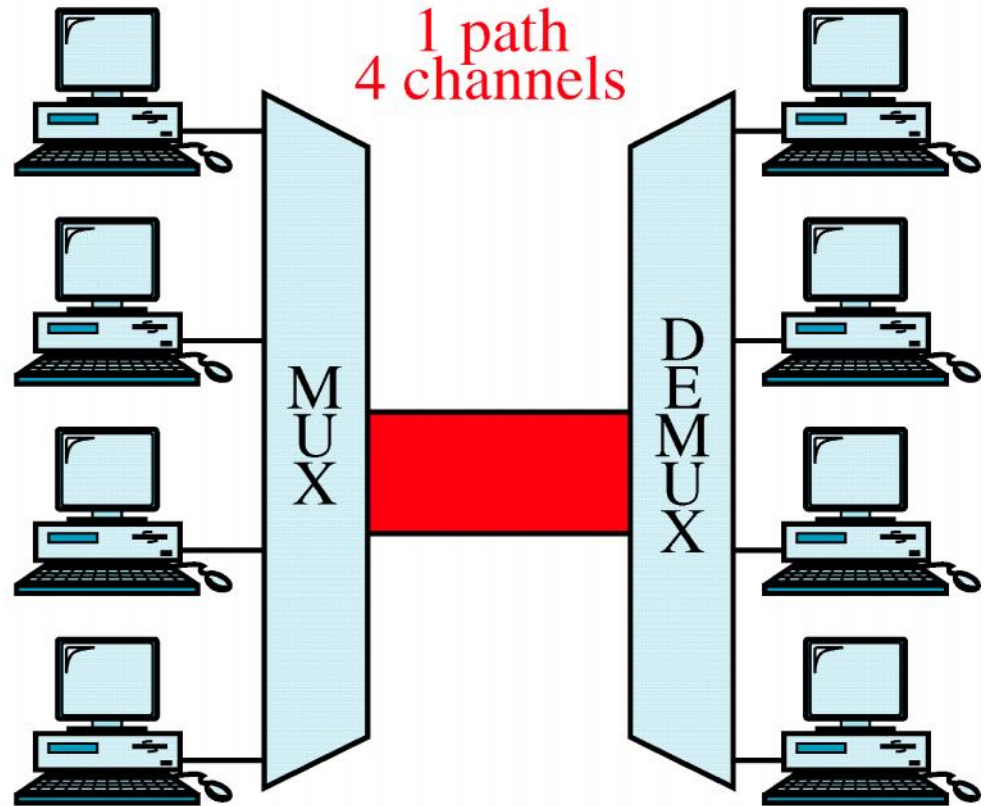
- **Many to one/one to many**
- **Types of multiplexing**
- **Telephone system**

Figure 8-1

Multiplexing vs. No Multiplexing



a. No multiplexing



b. Multiplexing

Figure 8-2

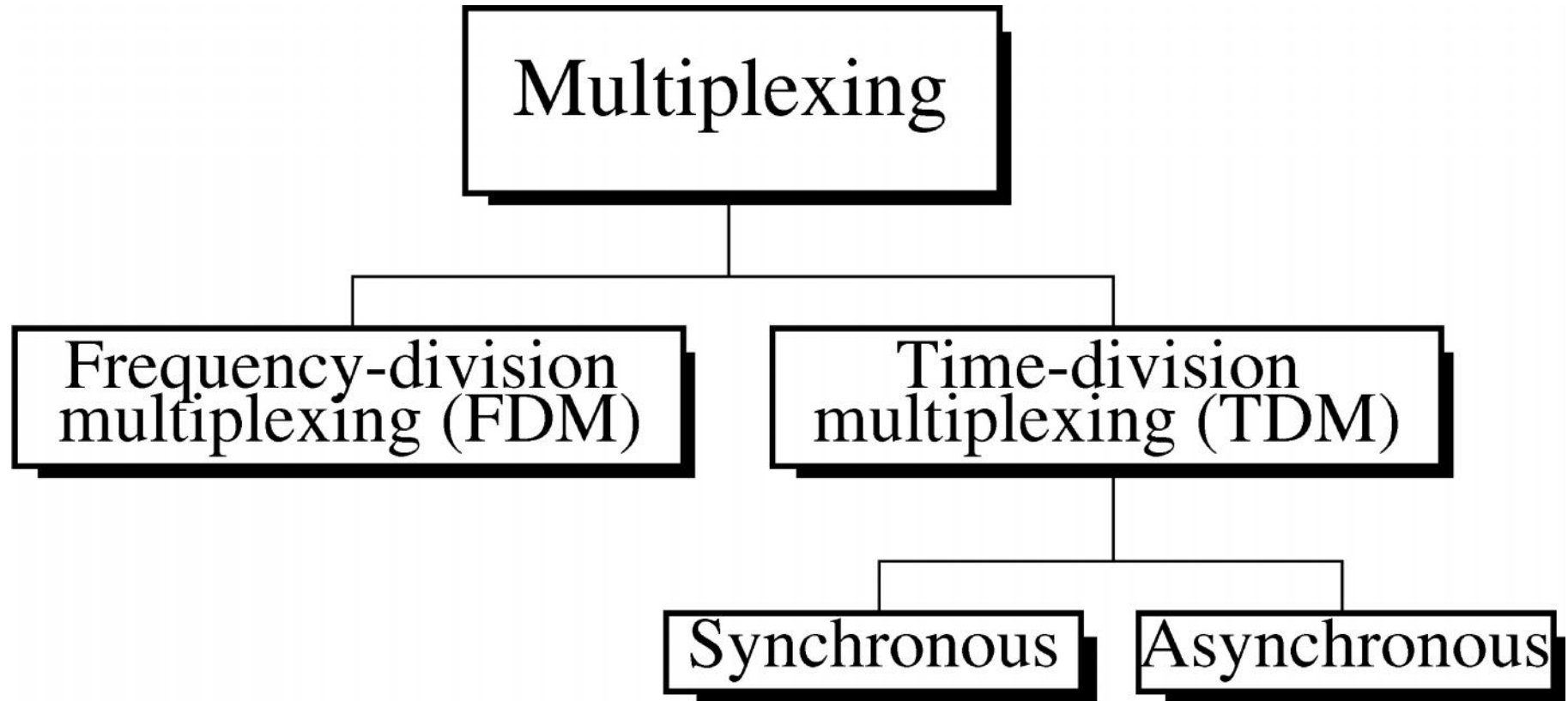


Figure 8-3

FDM

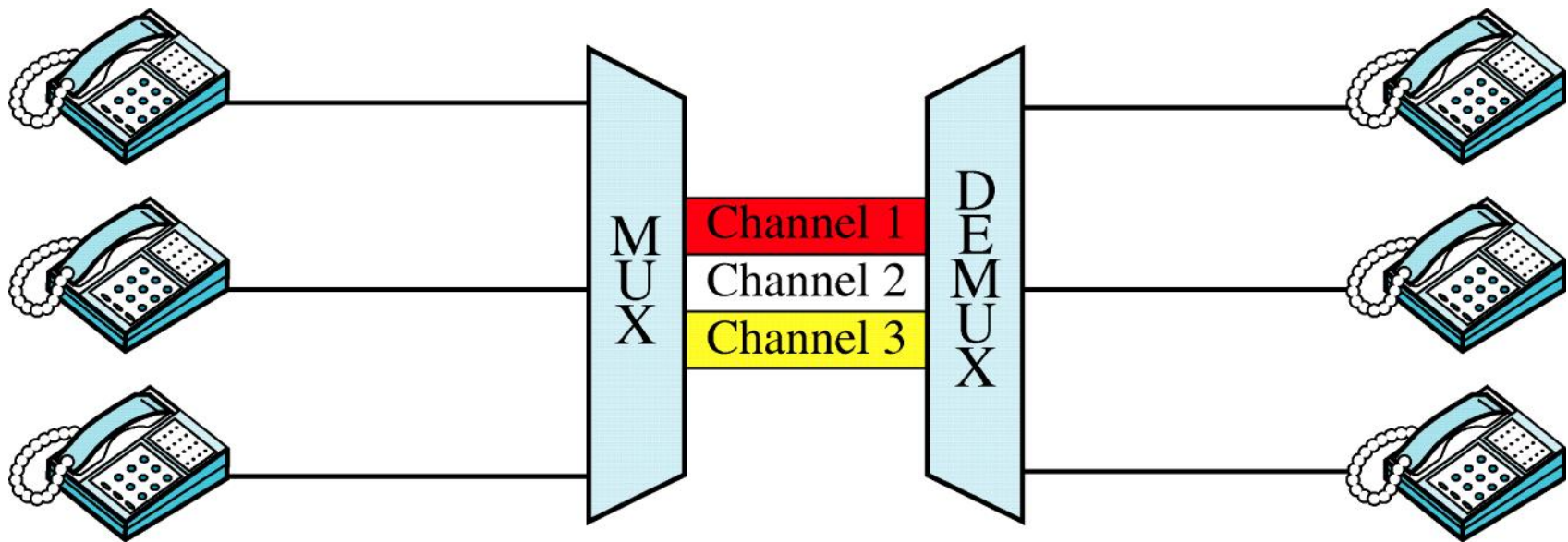


Figure 8-4

FDM, Time Domain

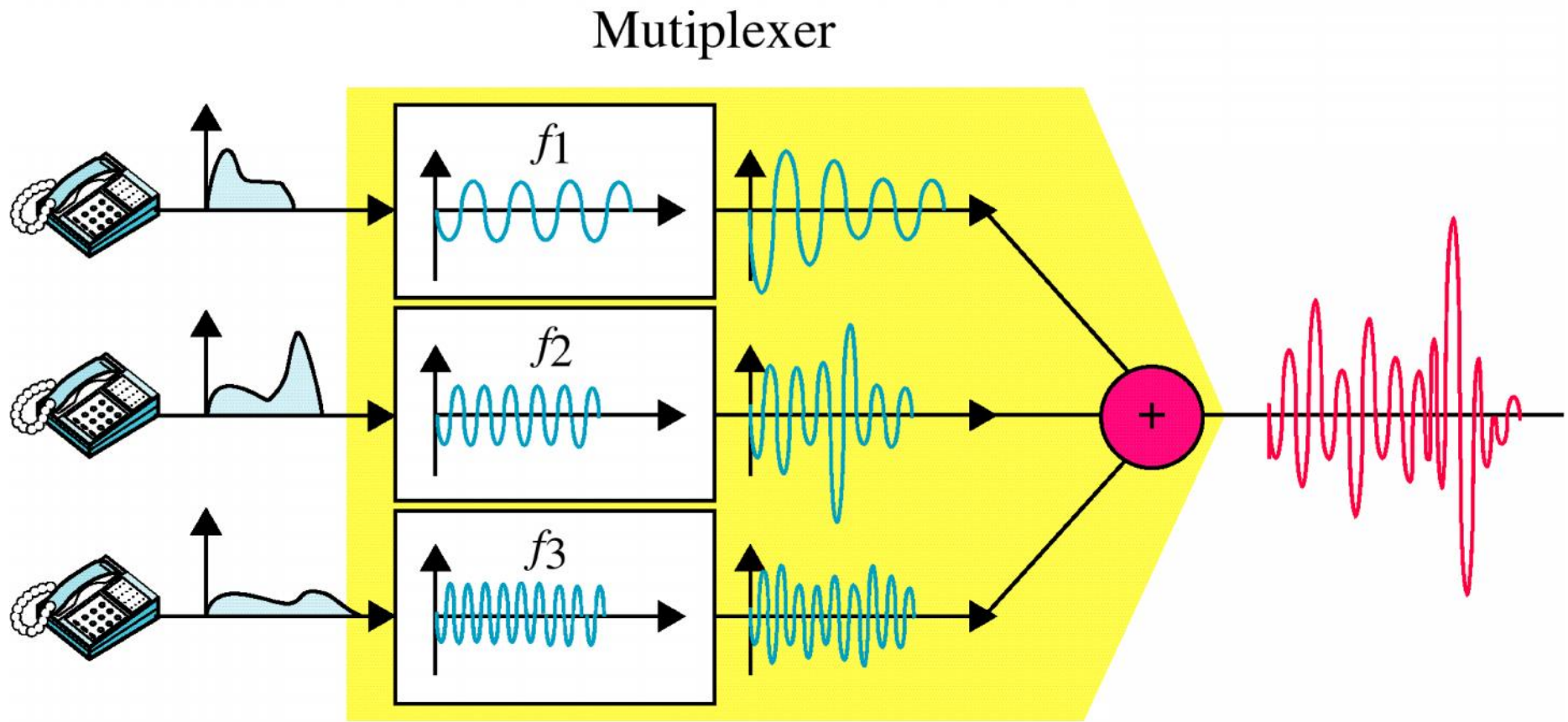


Figure 8-5

Multiplexing, Frequency Domain

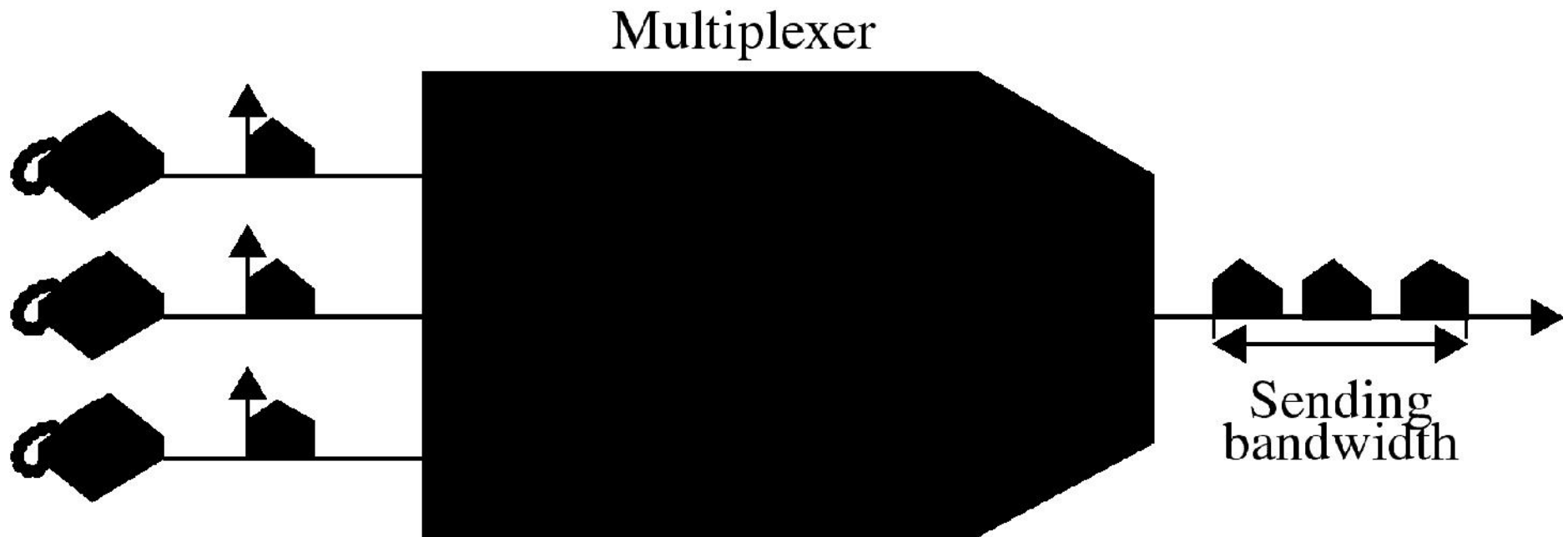


Figure 8-6

Demultiplexing, Time Domain

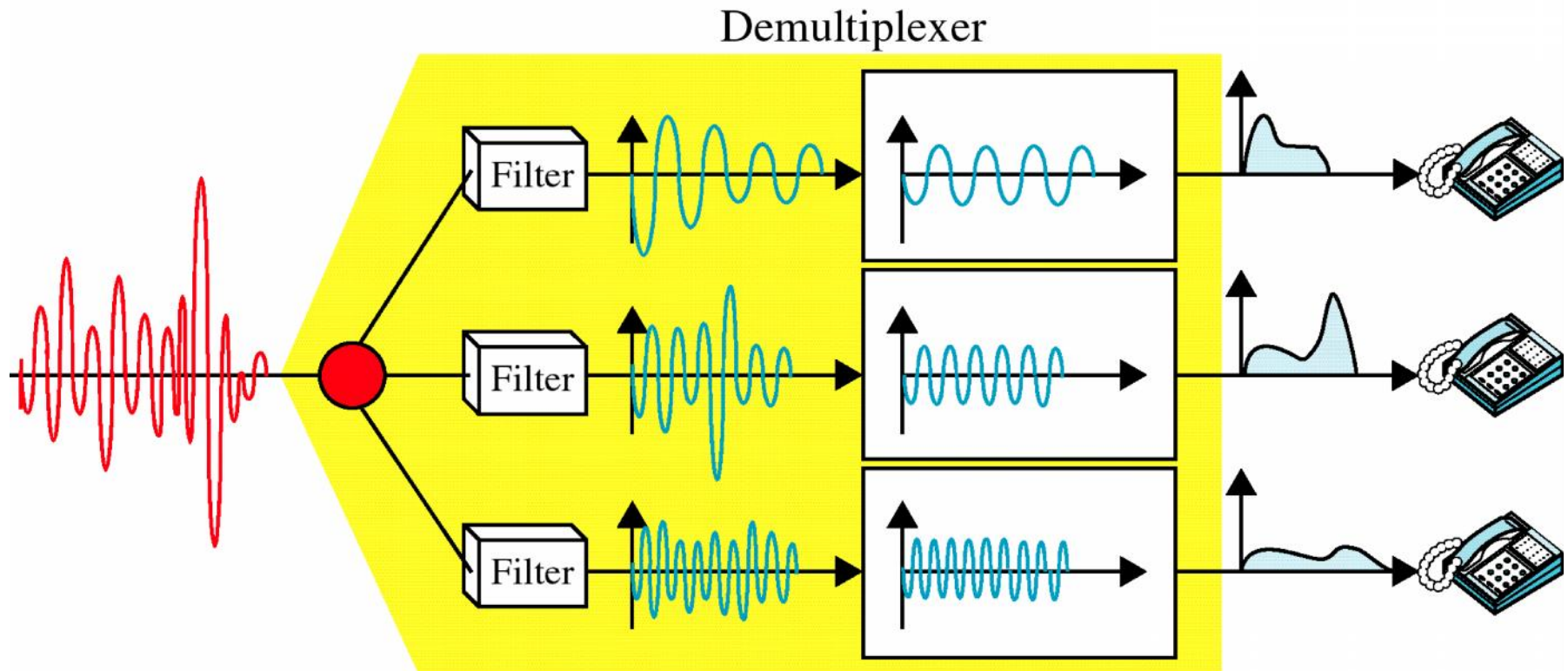


Figure 8-7

Demultiplexing, Frequency Domain

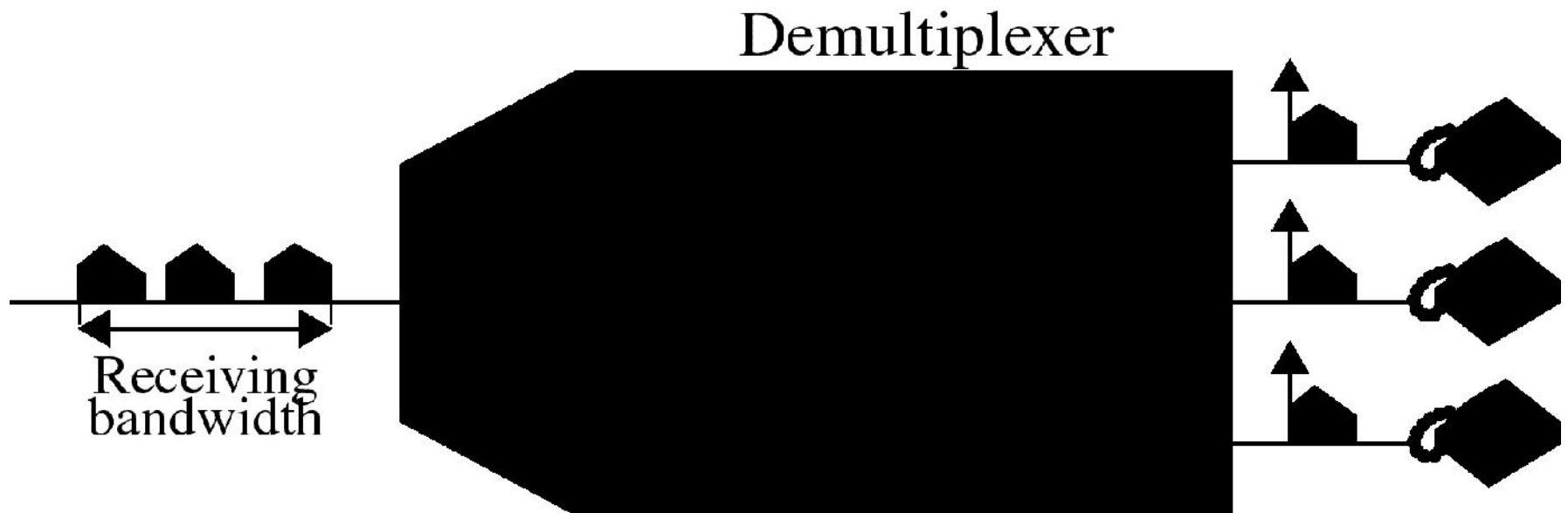


Figure 8-8

TDM

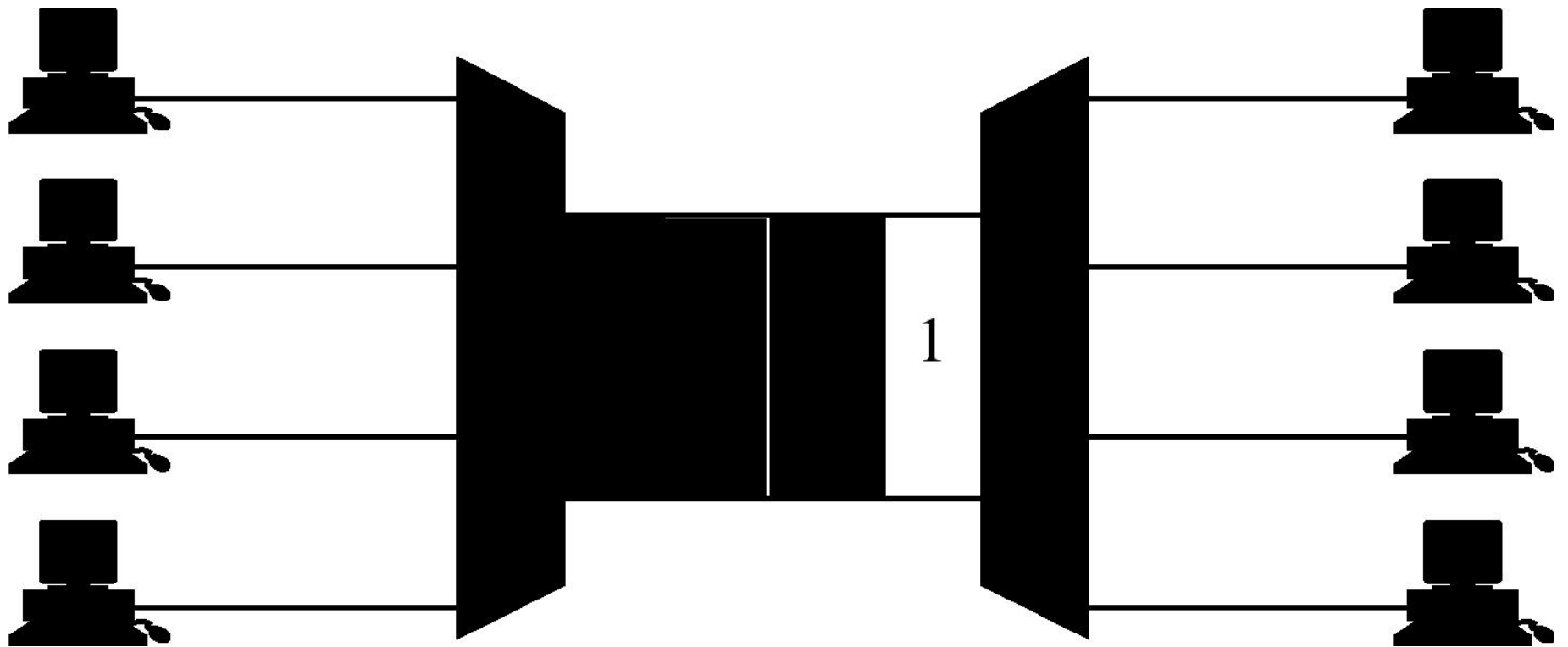


Figure 8-9

Synchronous TDM

5 Inputs

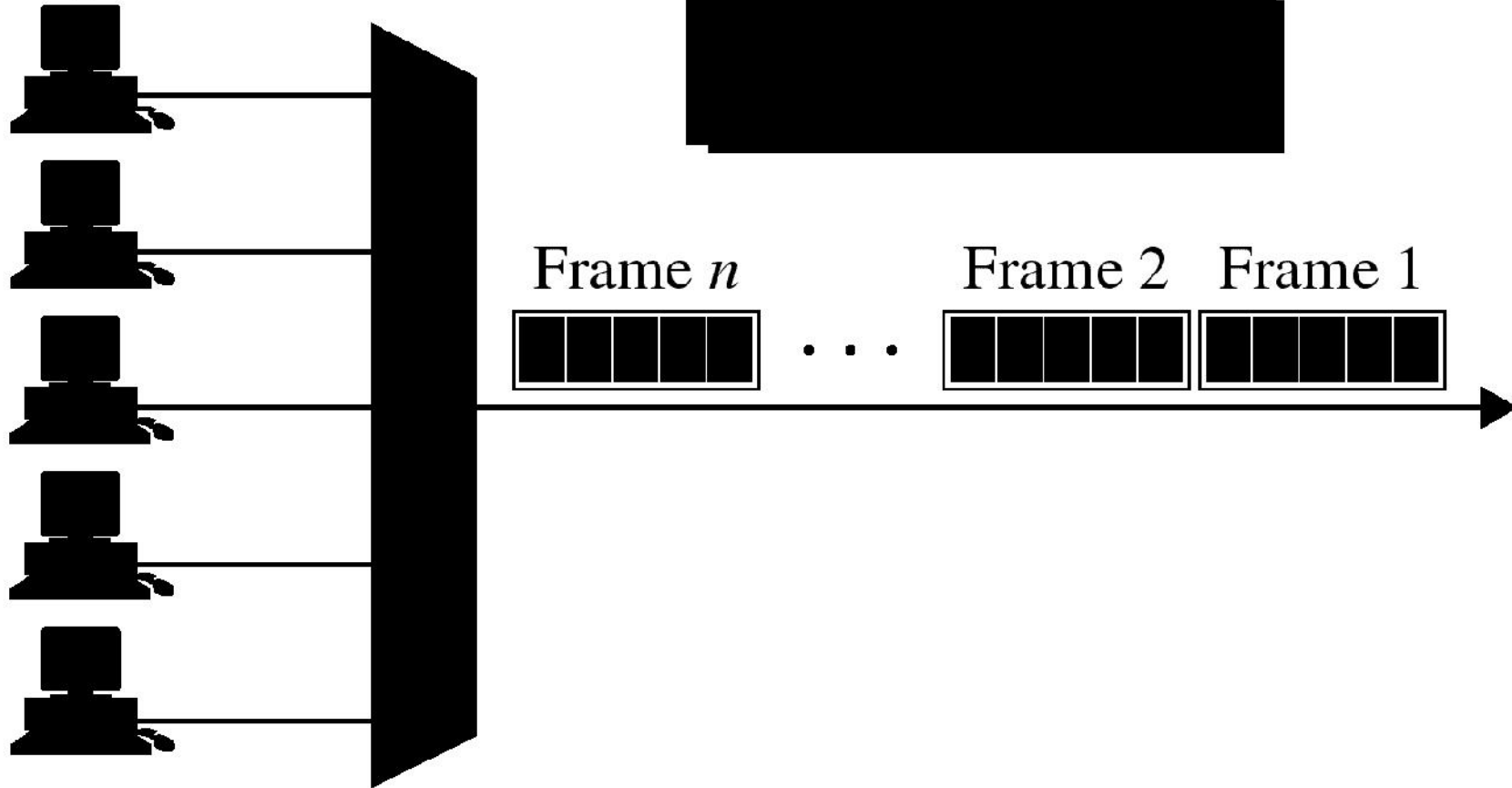


Figure 8-10

TDM, Multiplexing

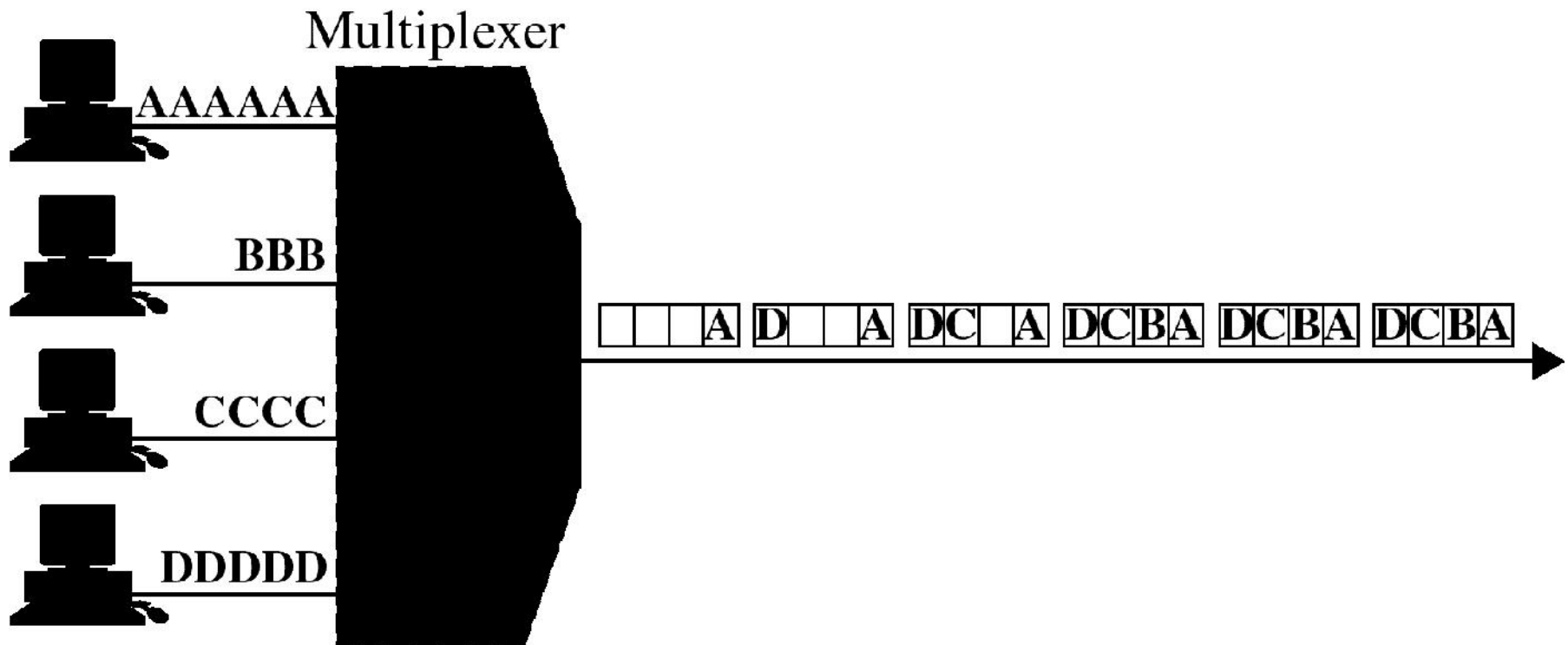


Figure 8-11

TDM, Demultiplexing

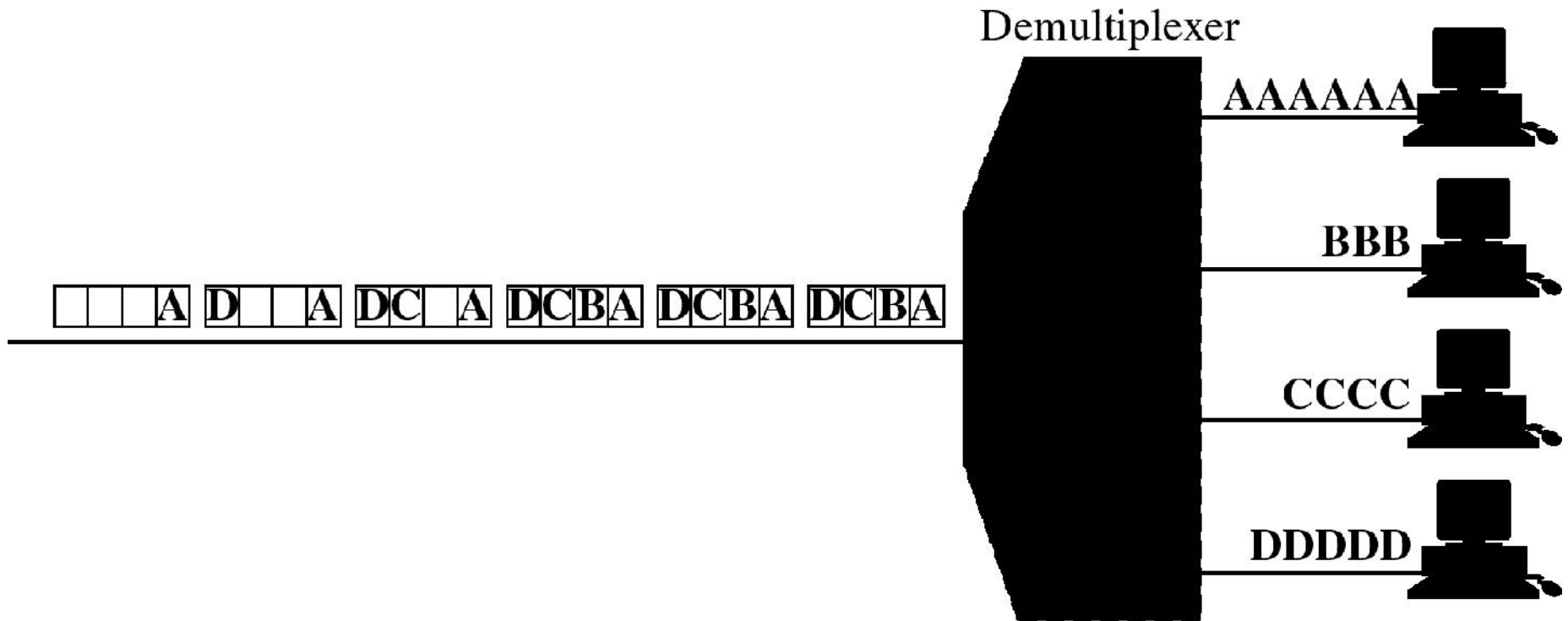


Figure 8-12

Framing Bits

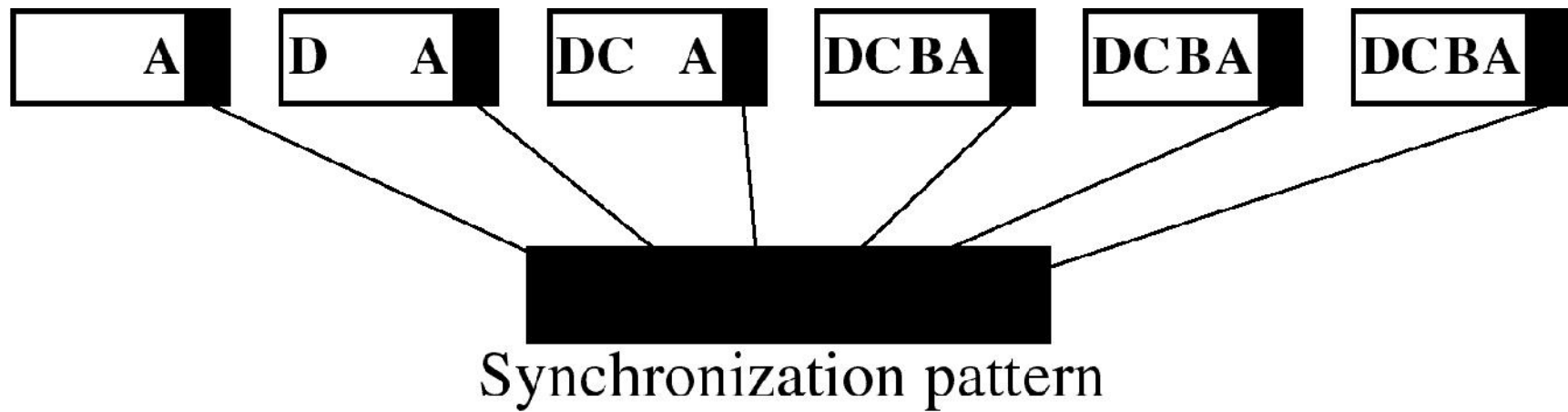


Figure 8-13

Data Rate

$$8250 \text{ bps} = 250 \text{ frames/second} \times 33 \text{ bits/frame}$$

or

$$8250 \text{ bps} = 4 \times 2000 \text{ bps} + 250 \text{ synchronization bps}$$

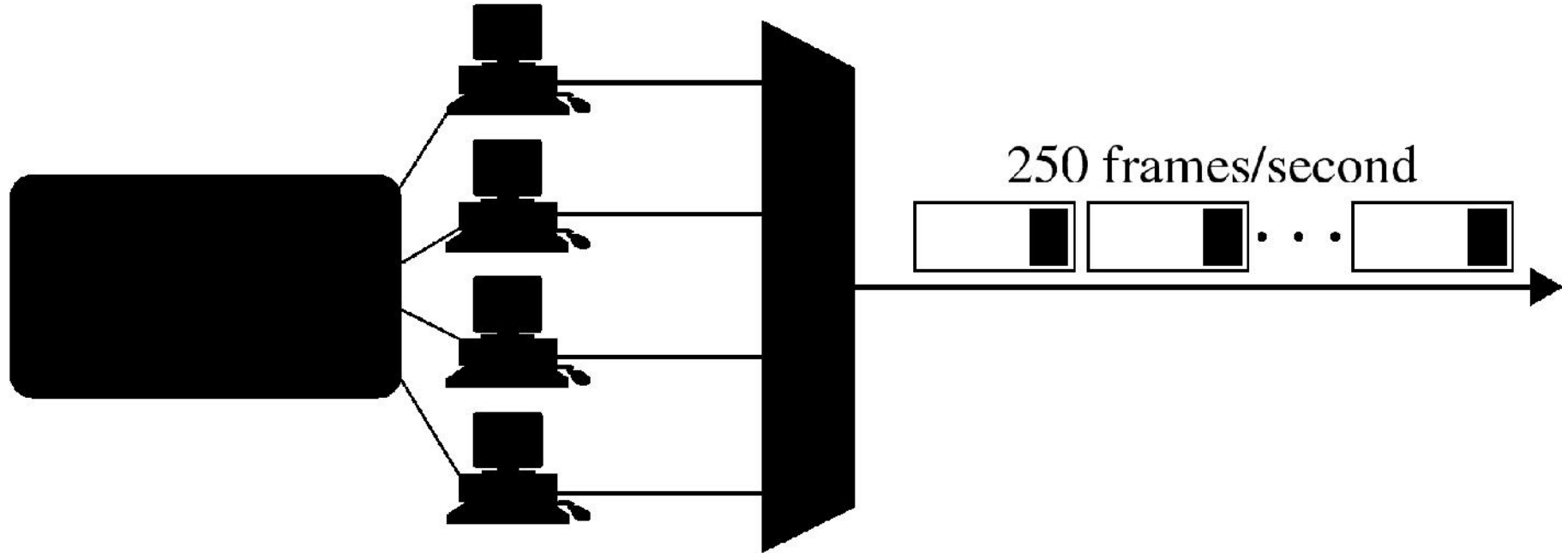


Figure 8-14

Asynchronous TDM

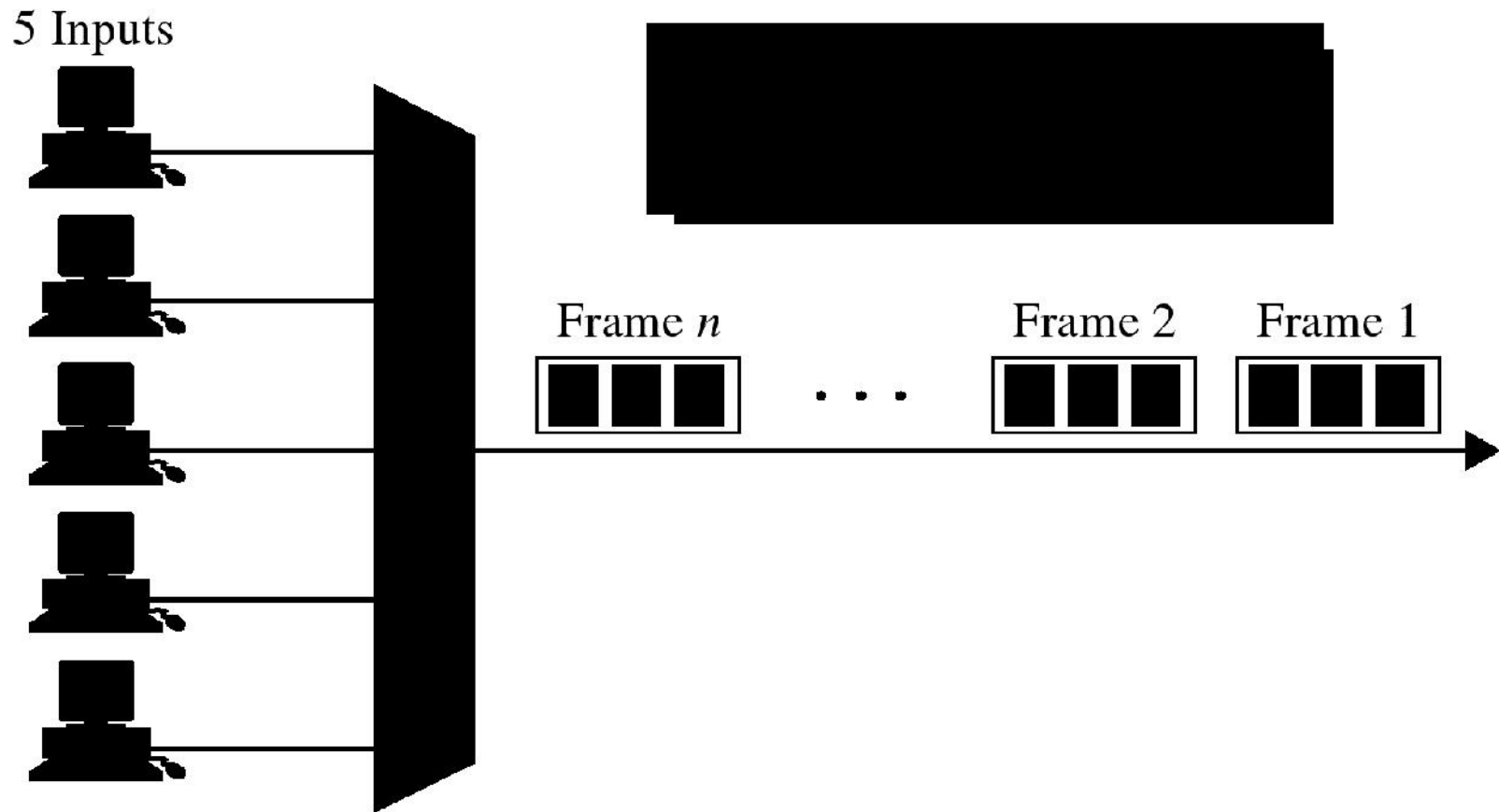
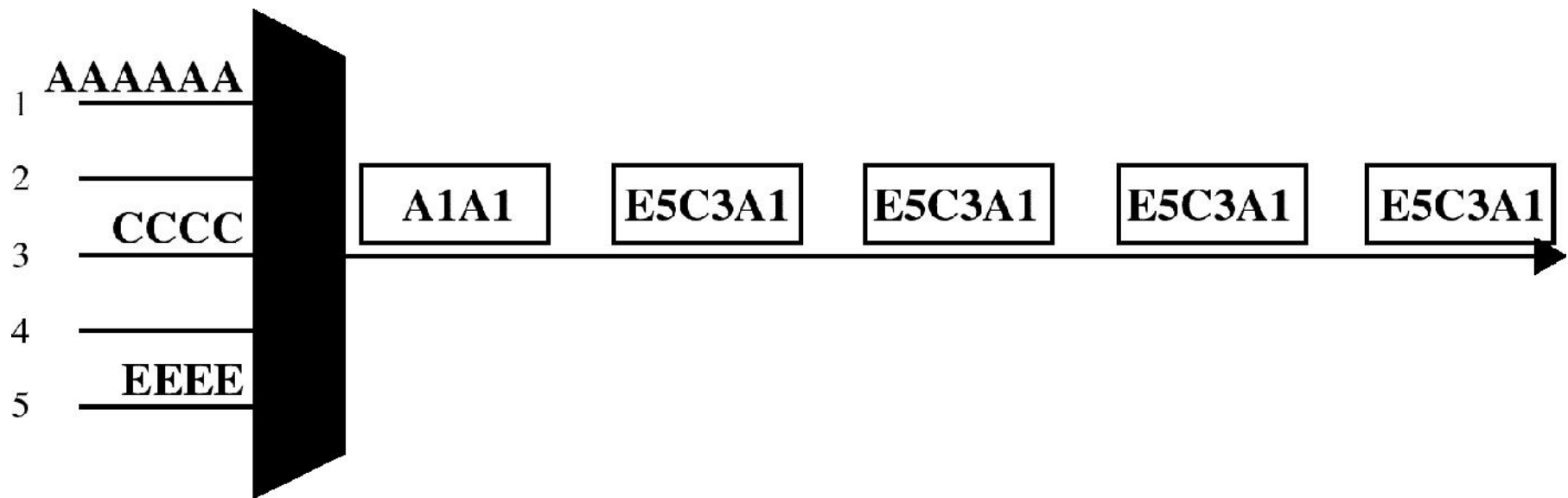


Figure 8-15

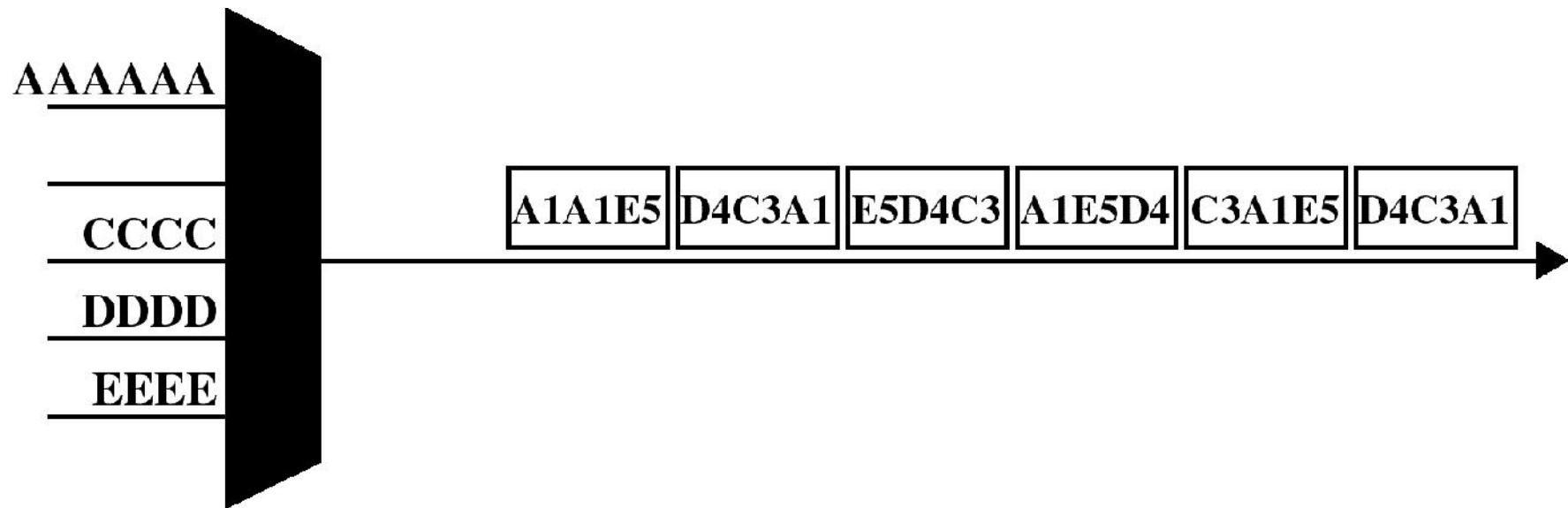
Frames and Addresses



a. Only three lines sending data

Figure 8-15-continued

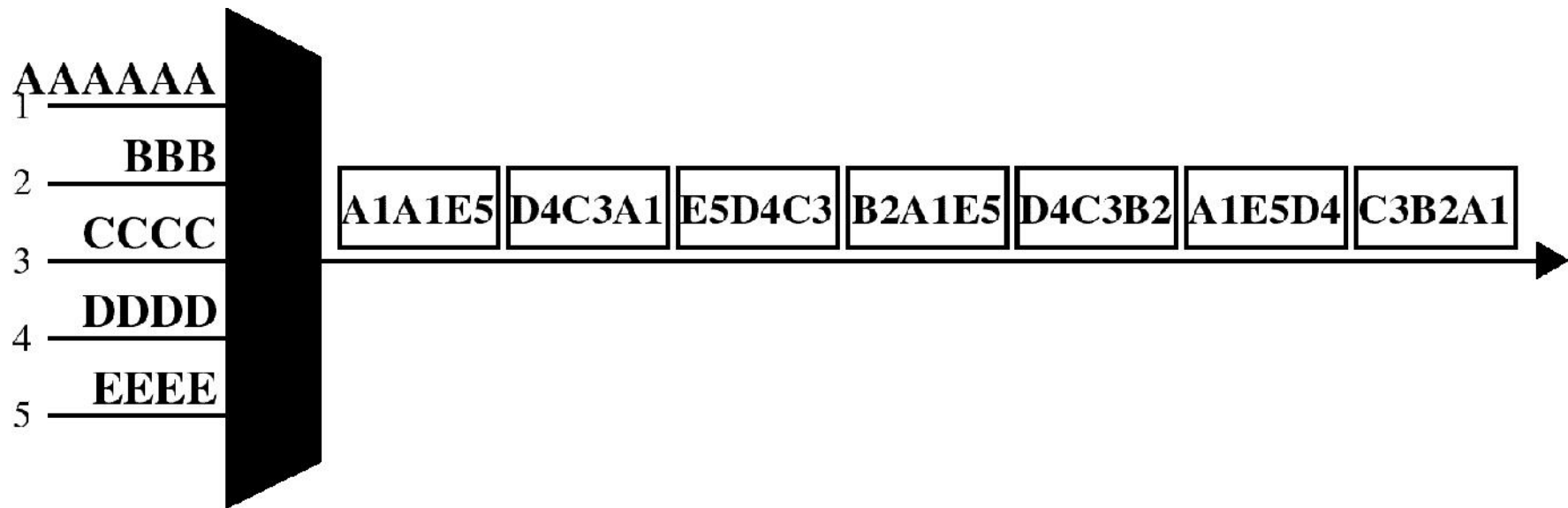
Frames and Addresses



d. Only four lines sending data

Figure 8-15-continued

Frames and Addresses



c. All five lines sending data

Figure 8-16

Multiplexing and Inverse Multiplexing

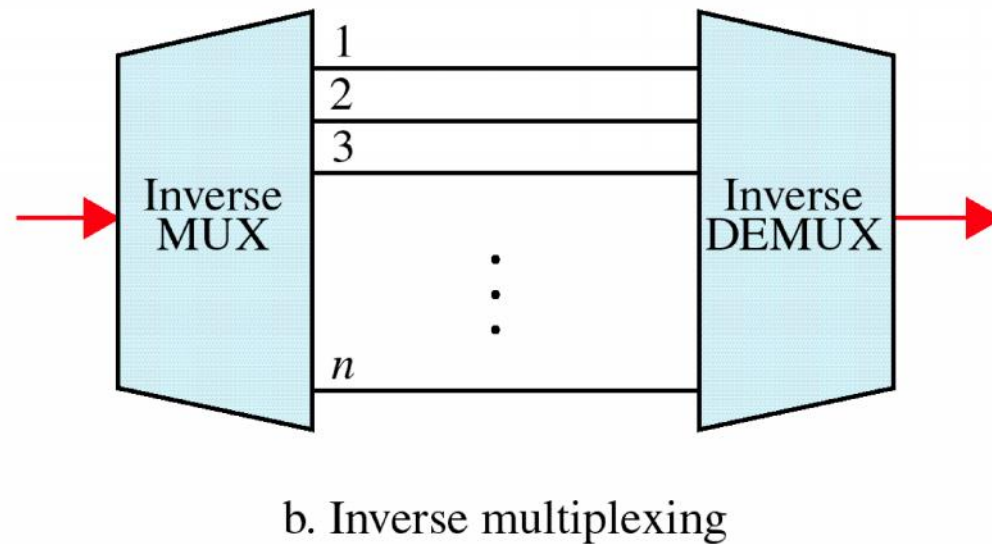
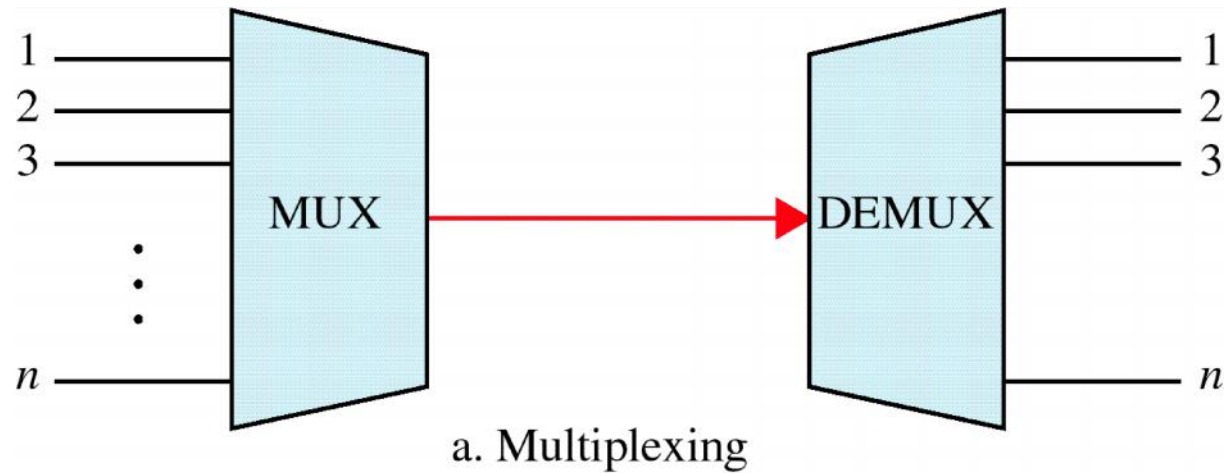


Figure 8-17

Telephone Network

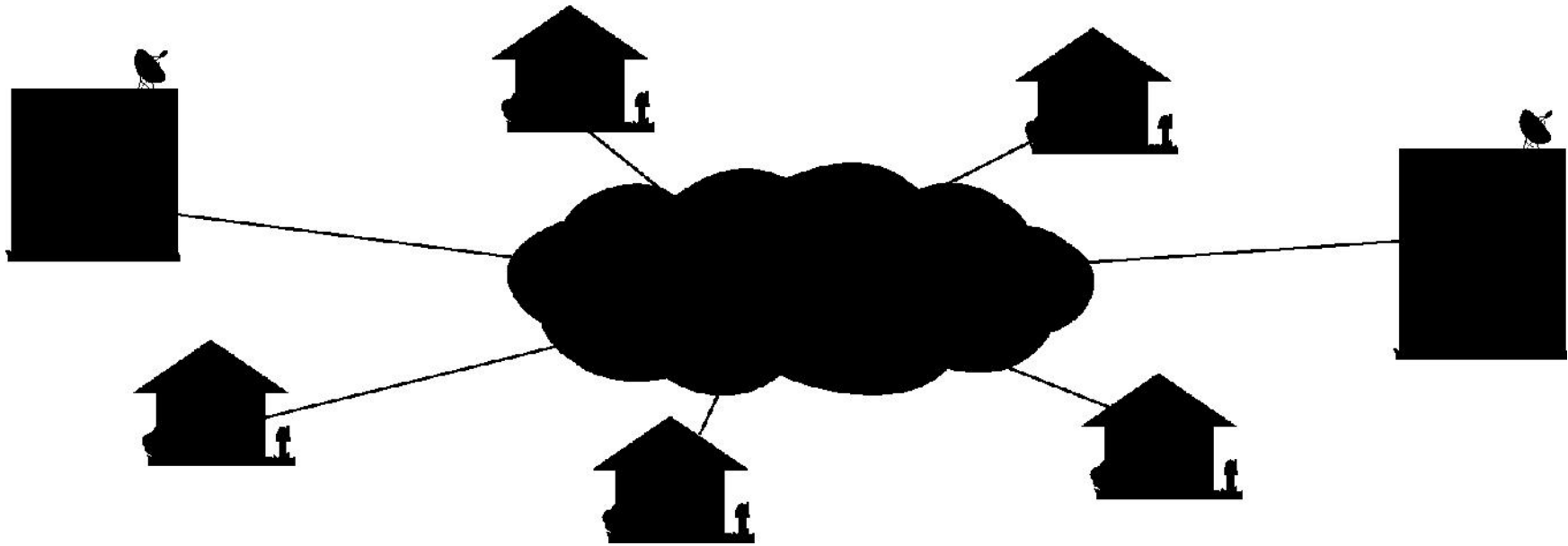


Figure 8-18



Figure 8-19

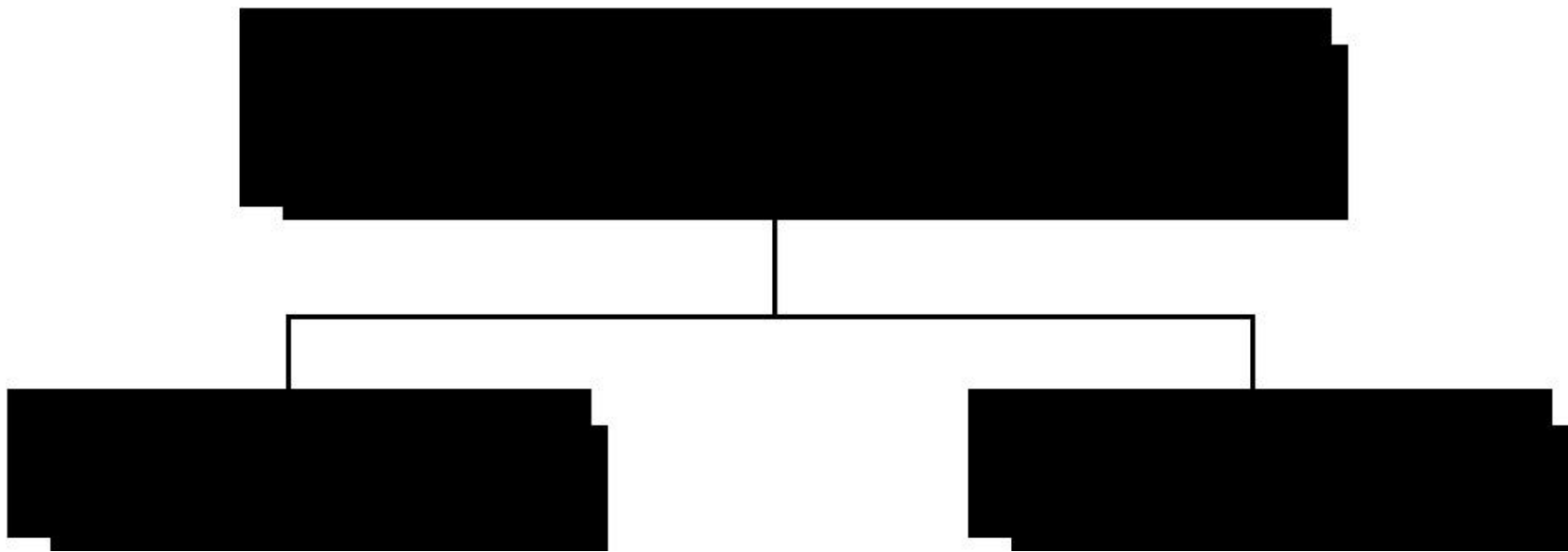


Figure 8-20

Analog Switched Service

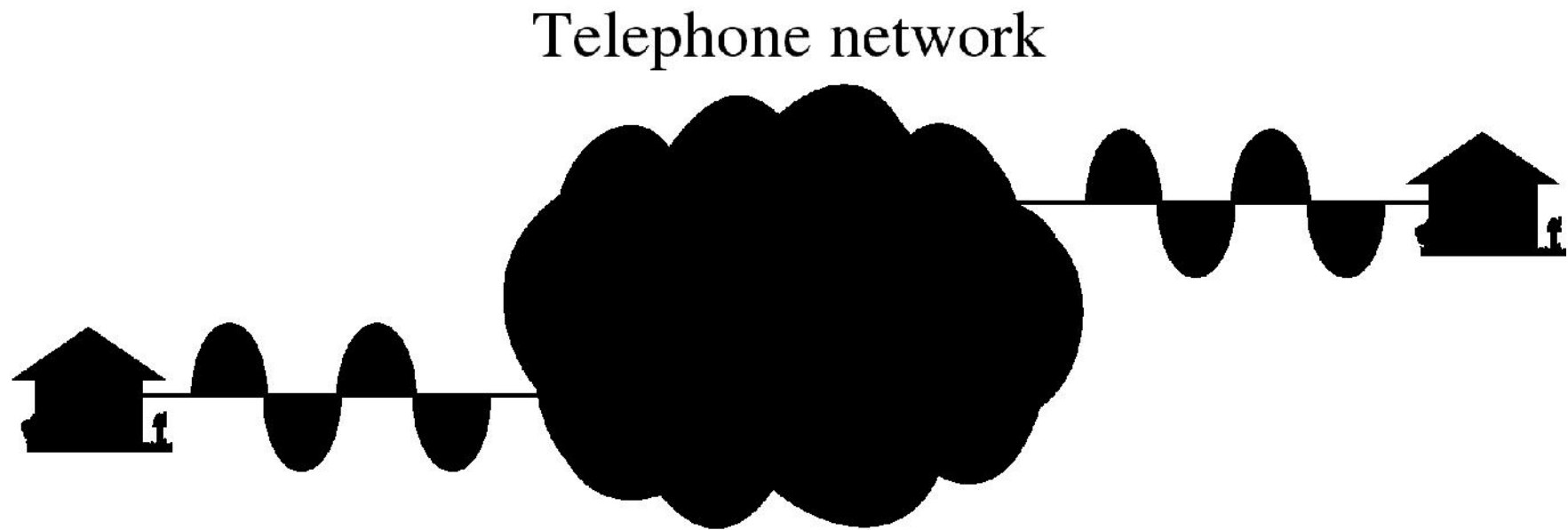


Figure 8-21

Analog Leased Service

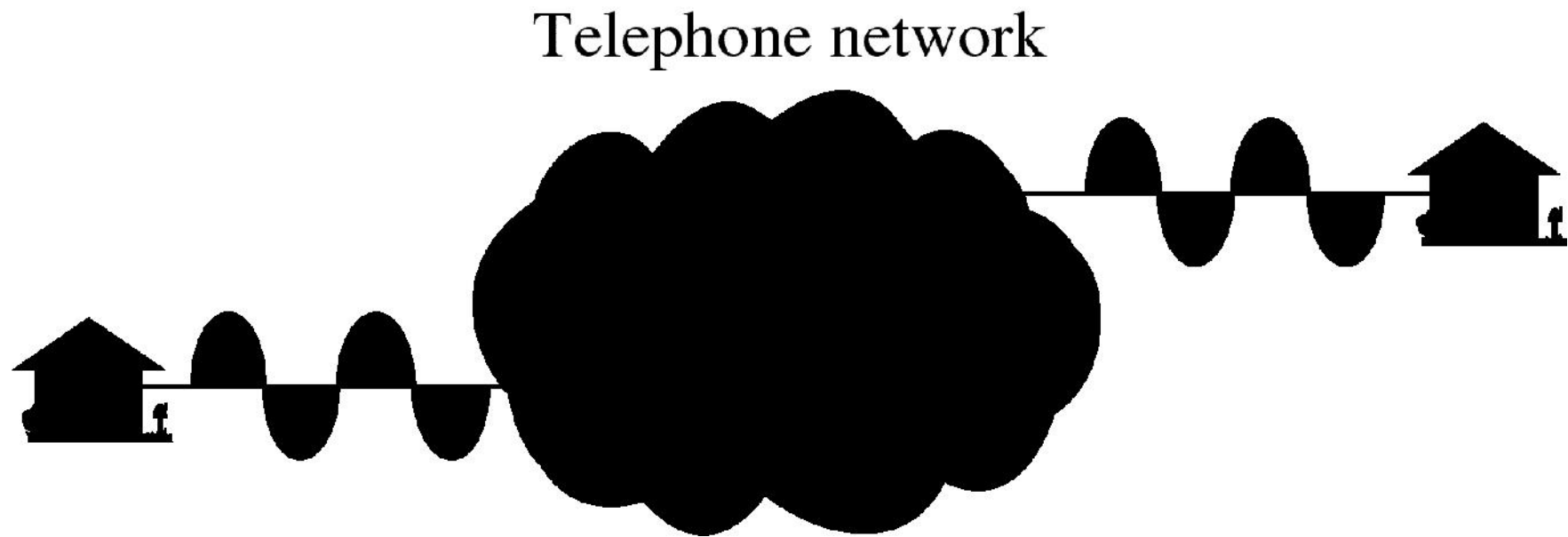


Figure 8-22

Analog Hierarchy

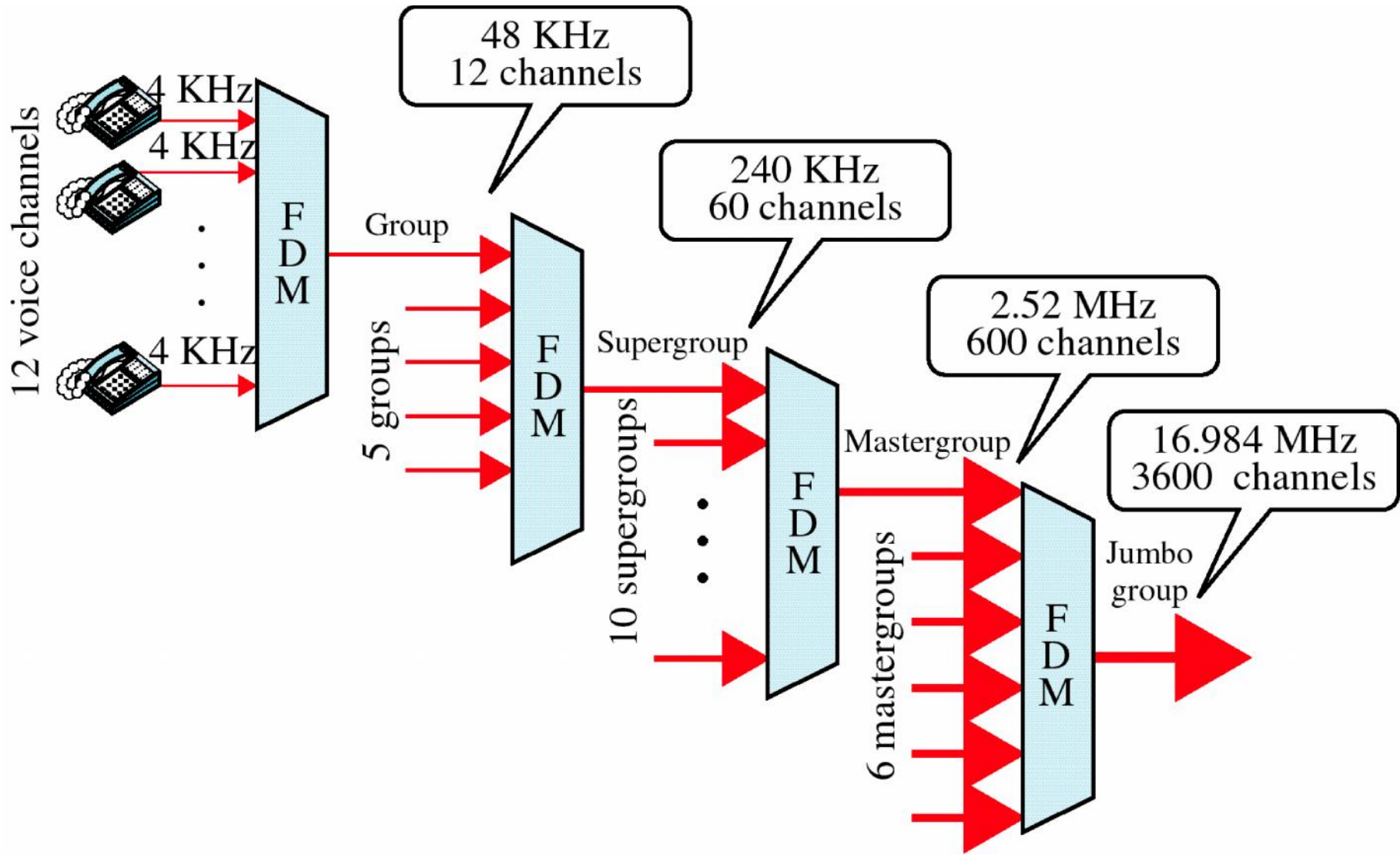


Figure 8-23

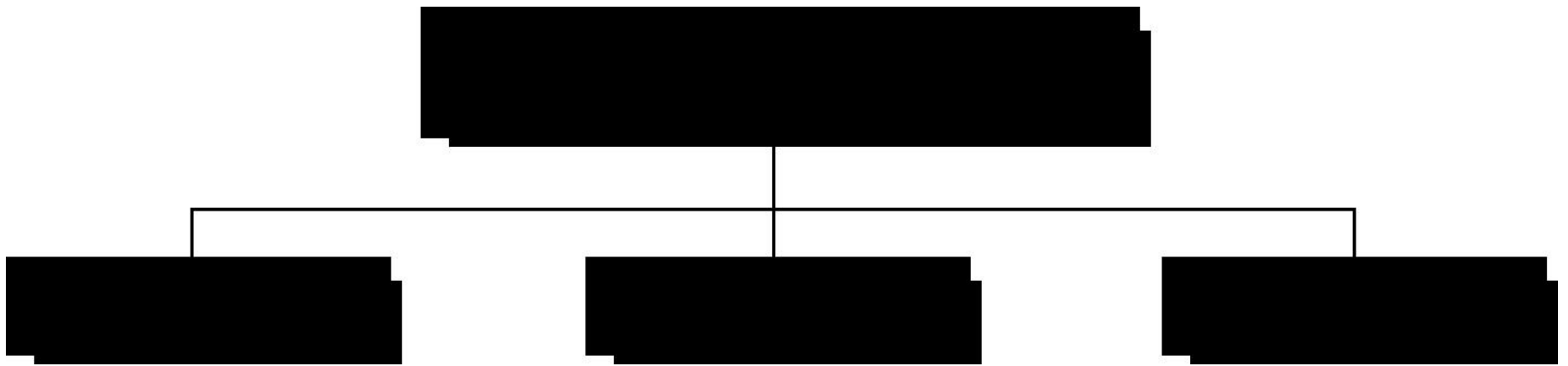


Figure 8-24

Switched/56 Service

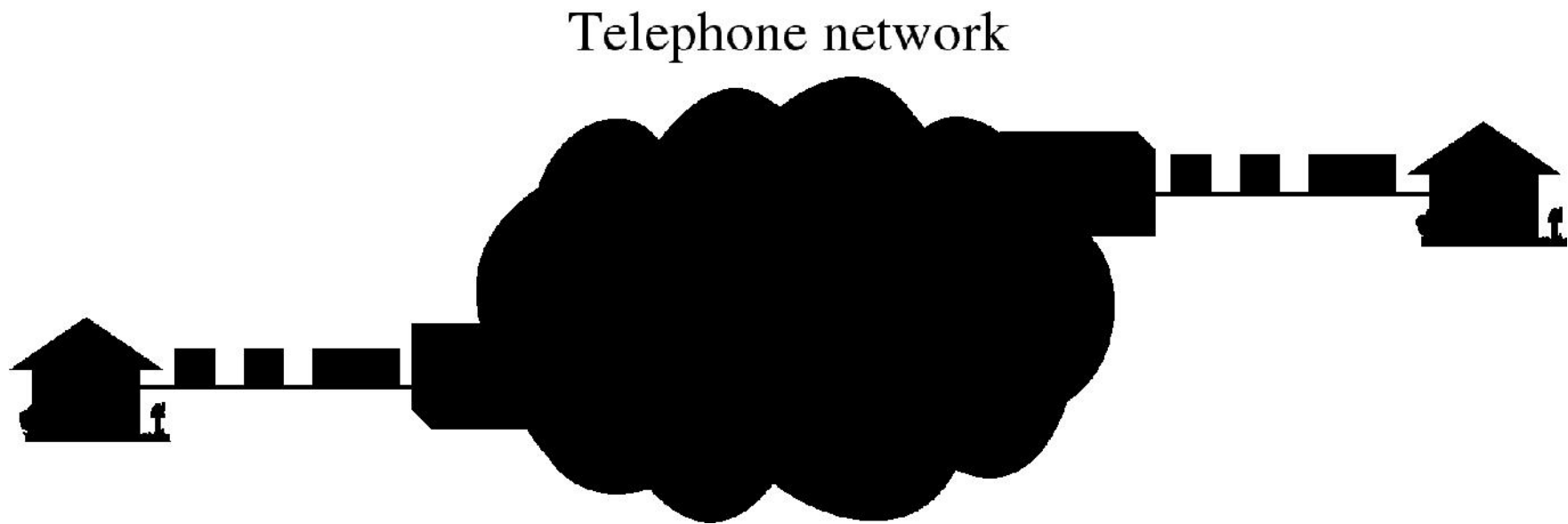


Figure 8-25

DDS

Telephone network



Figure 8-26

DS Hierarchy

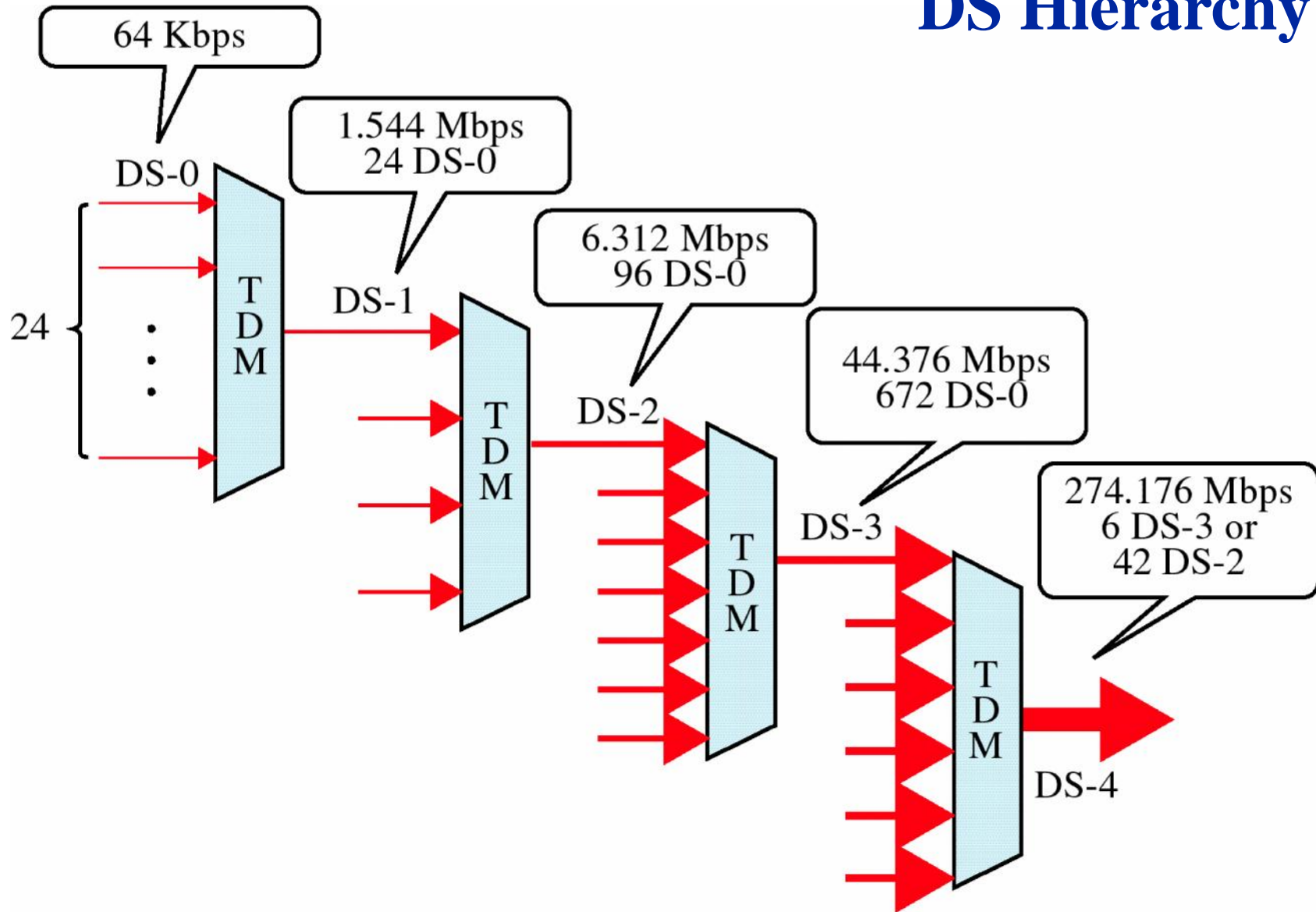


Figure 8-27

T-1 Line

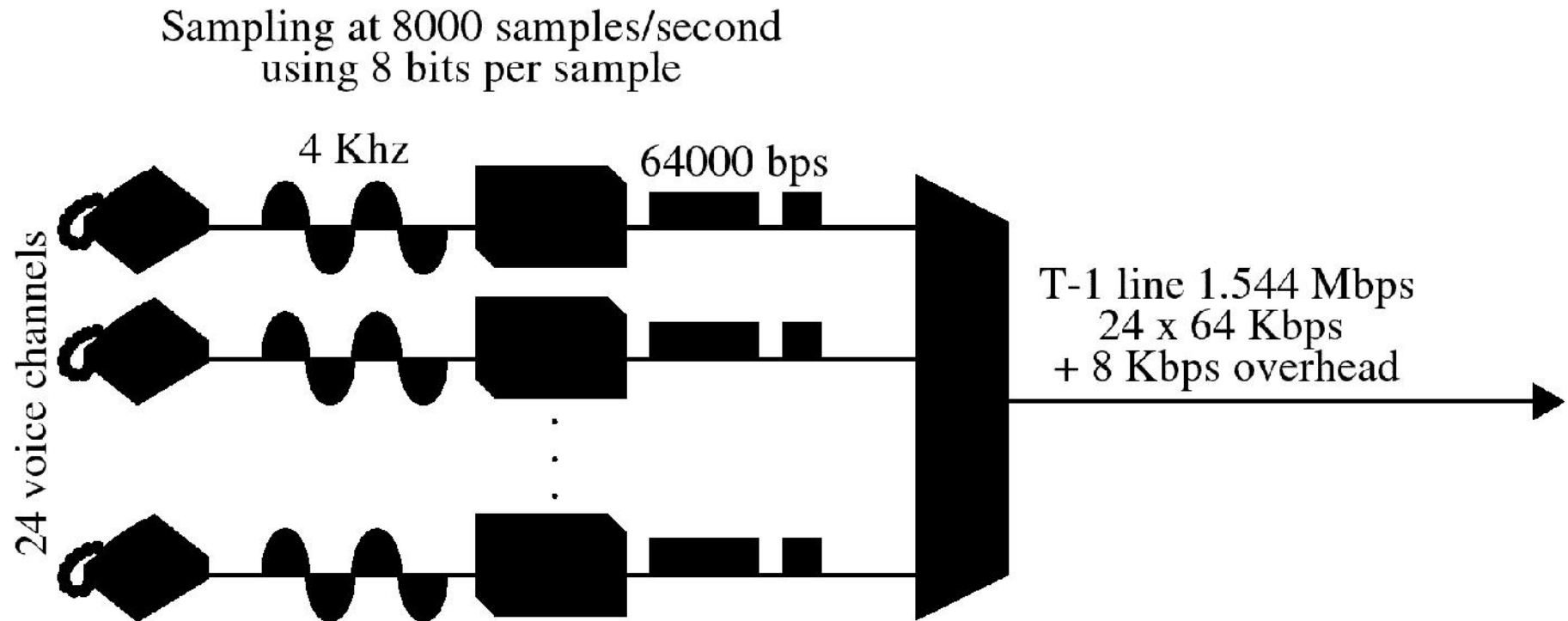


Figure 8-28

T-1 Frame

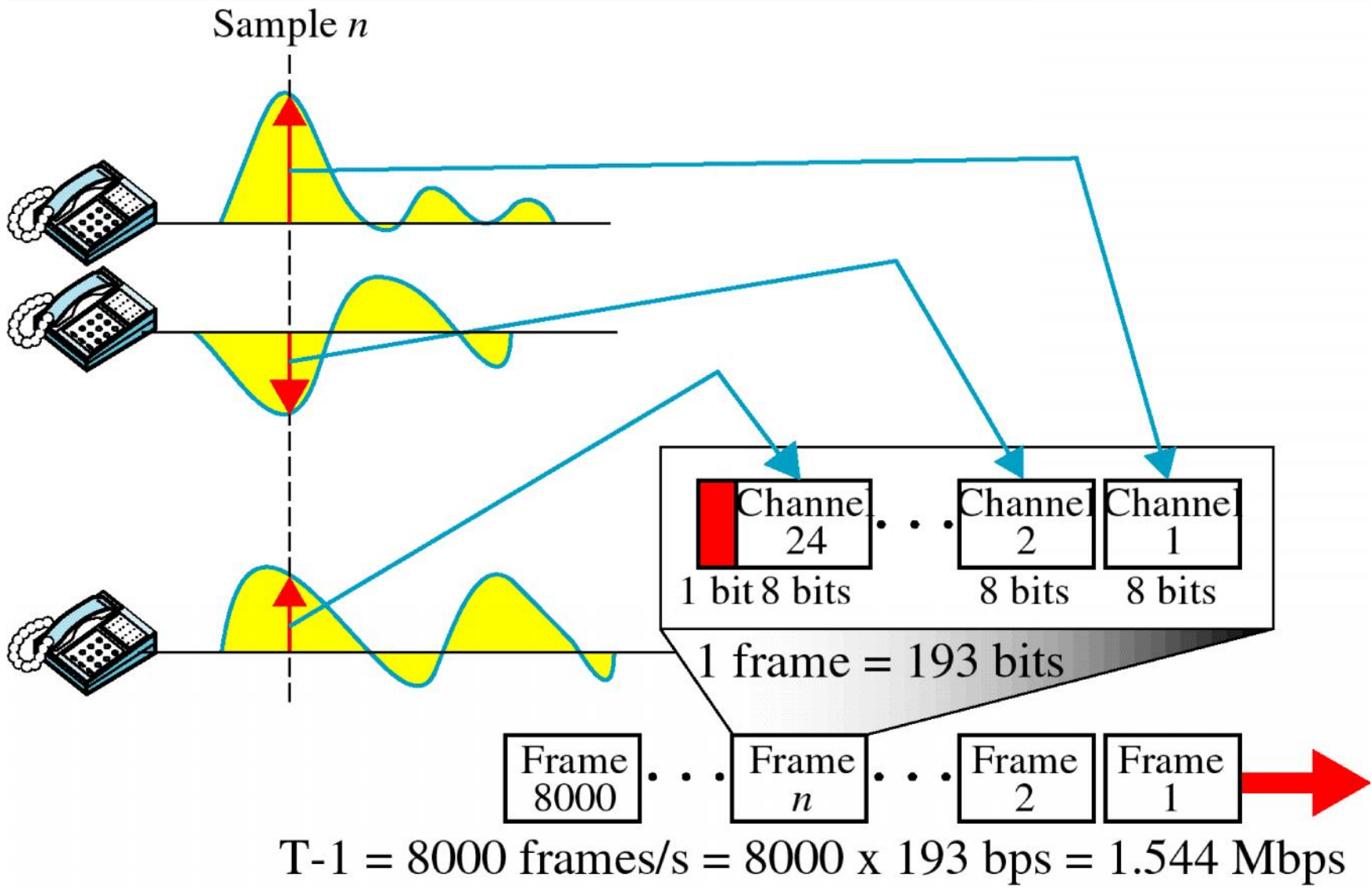
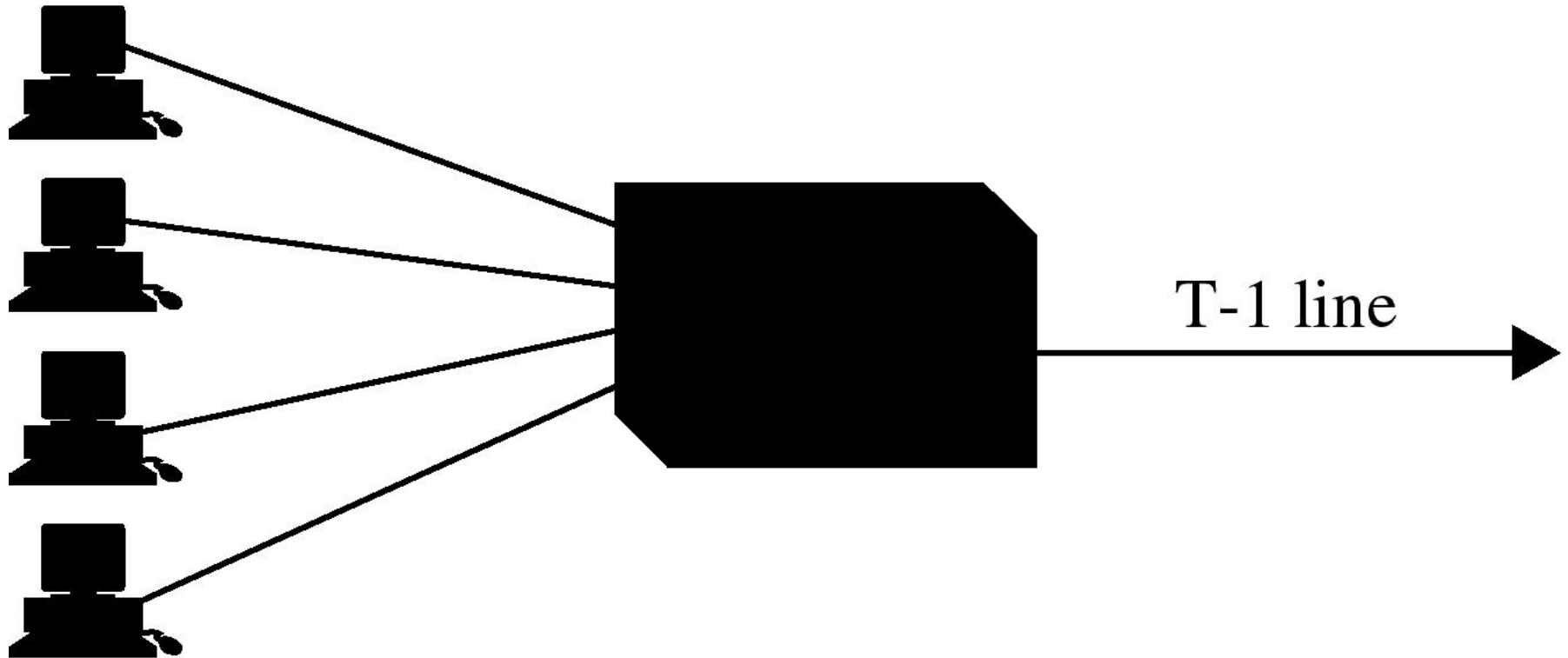


Figure 8-29

Fractional T-1 Line



COMPUTER NETWORKS – Unit 1

Topic 12

SPREADSPECTRUM



I. History of Spread Spectrum

II. Spread Spectrum System Model

III. Spread Spectrum Classification

IV. Spread Spectrum Techniques

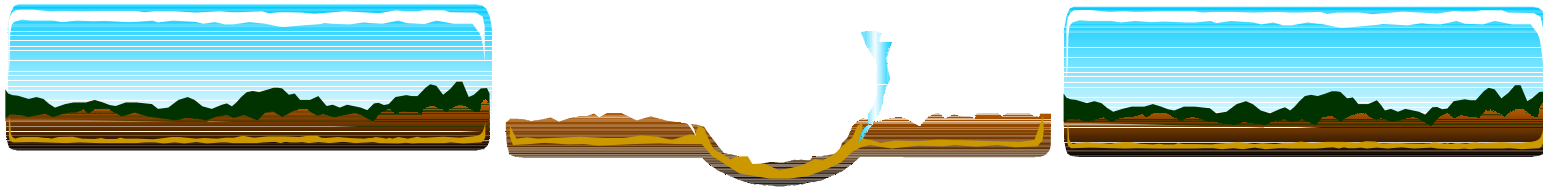
History Of Spread Spectrum

- Spread Spectrum was actually invented by 1940s Hollywood actress **Hedy Lamarr(1913-2000)**.
- An Austrian refugee, in 1940 at the age of 26, she devised together with music composer George Antheil a system to stop enemy detection and jamming of radio controlled torpedoes by **hopping around a set of frequencies in a random fashion.**
- She was granted a **patent in 1942** (US pat. 2292387) but considered it her contribution to the war effort and never profited.
- Techniques known since 1940s and used in military communication systems since 1950s.



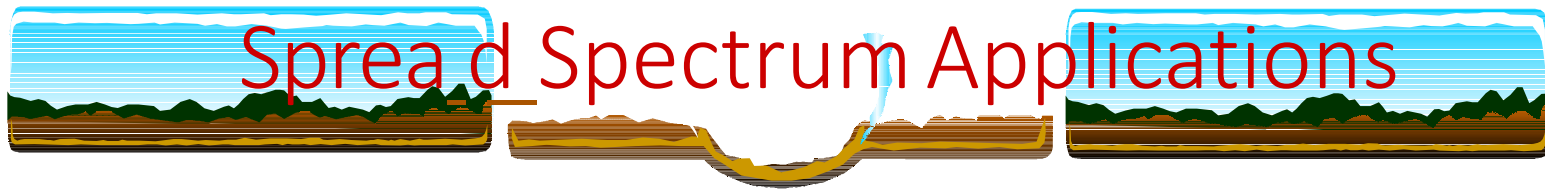
Introduction to Spread Spectrum

- ✚ **“Spread” radio signal over a wide frequency range**
- ✚ **Several magnitudes higher than minimum requirement**
- ✚ **Gained popularity by the needs of military communication**
- ✚ **Proved resistant against hostile jammers**
- ✚ **Ratio of information bandwidth and spreading bandwidth is identified as spreading gain or processing gain**
- ✚ **Processing gain does not combat white Noise**



☒ Offers the following applications:

- O able to deal with multi-path
- a multiple access due to different spreading sequences
- m spreading sequence design is very important for performance
- p low probability of interception | privacy
- p anti-jam capabilities



Spread Spectrum Applications

Interference

Prevents interference at specific frequencies

E.g. other radio users, electrical systems

Military

Prevents signal jamming

Scrambling of 'secret' messages

Wireless LAN security

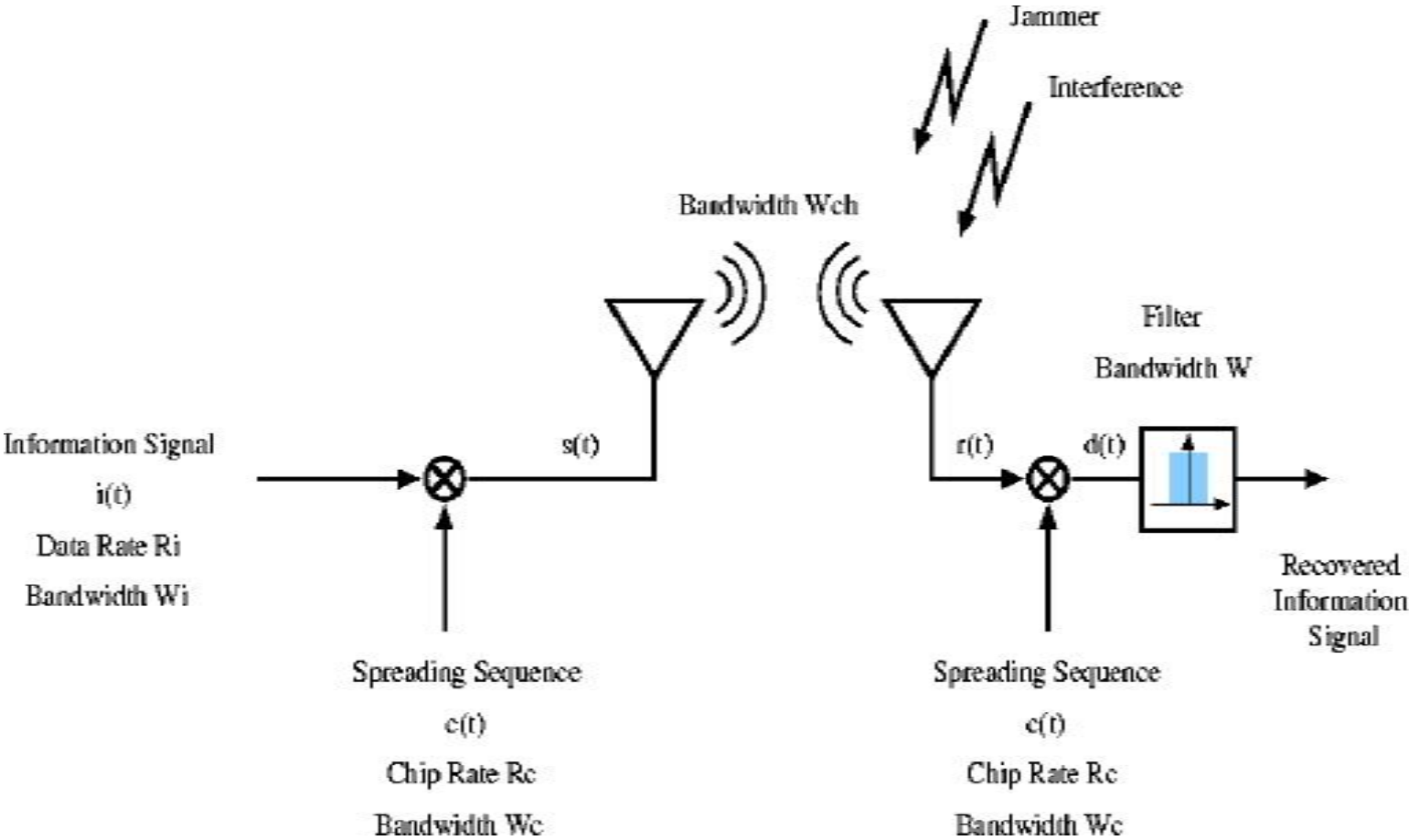
Prevents 'eavesdropping' of wireless links

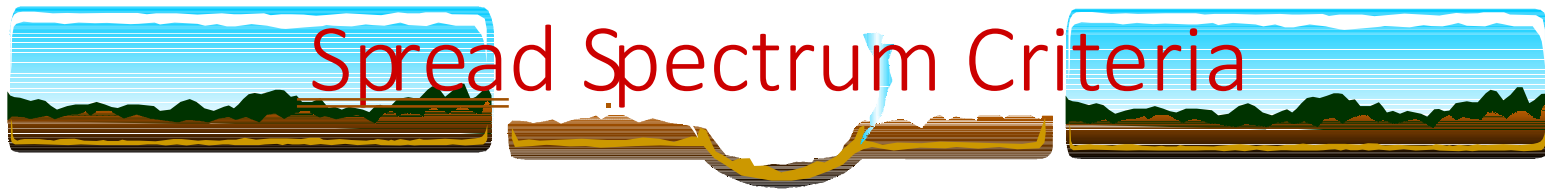
Prevents 'hacking' into wireless LANs

✂ **CDMA (Code Division Multiple Access)**

Multiple separate channels in same medium **using different spreading codes**

System Model: Spread Spectrum Transmission



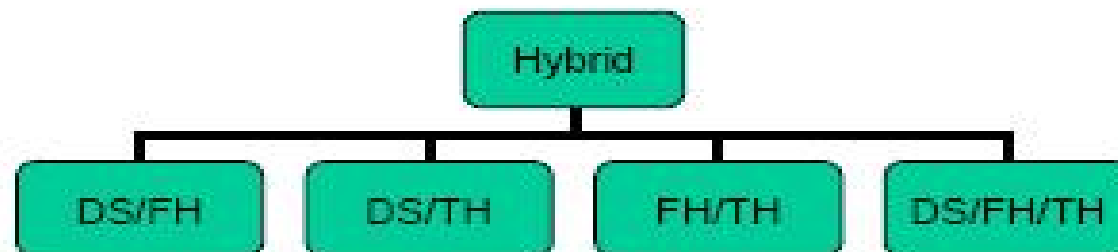
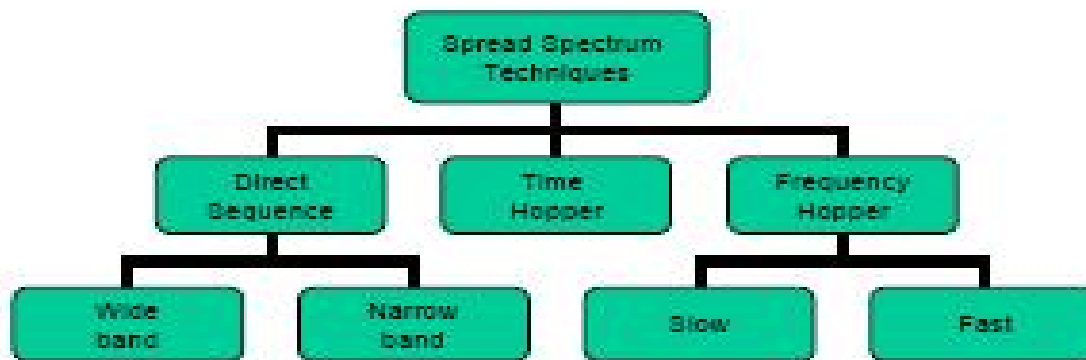


Spread Spectrum Criteria

A communication system is considered a spread spectrum system if it satisfies the following two criteria:

- ✚ Bandwidth of the spread spectrum signal has to be greater than the information bandwidth. (This is also true for frequency and pulse code modulation!)**
- ✚ The spreading sequence has to be independent from the information. Thus, no possibility to calculate the information if the sequence is known and vice versa.**

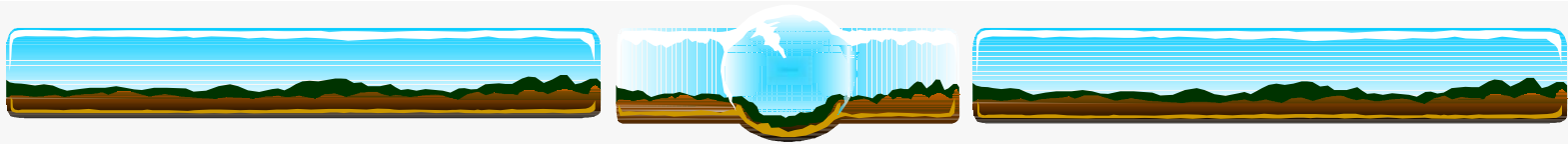
Spread Spectrum Classification



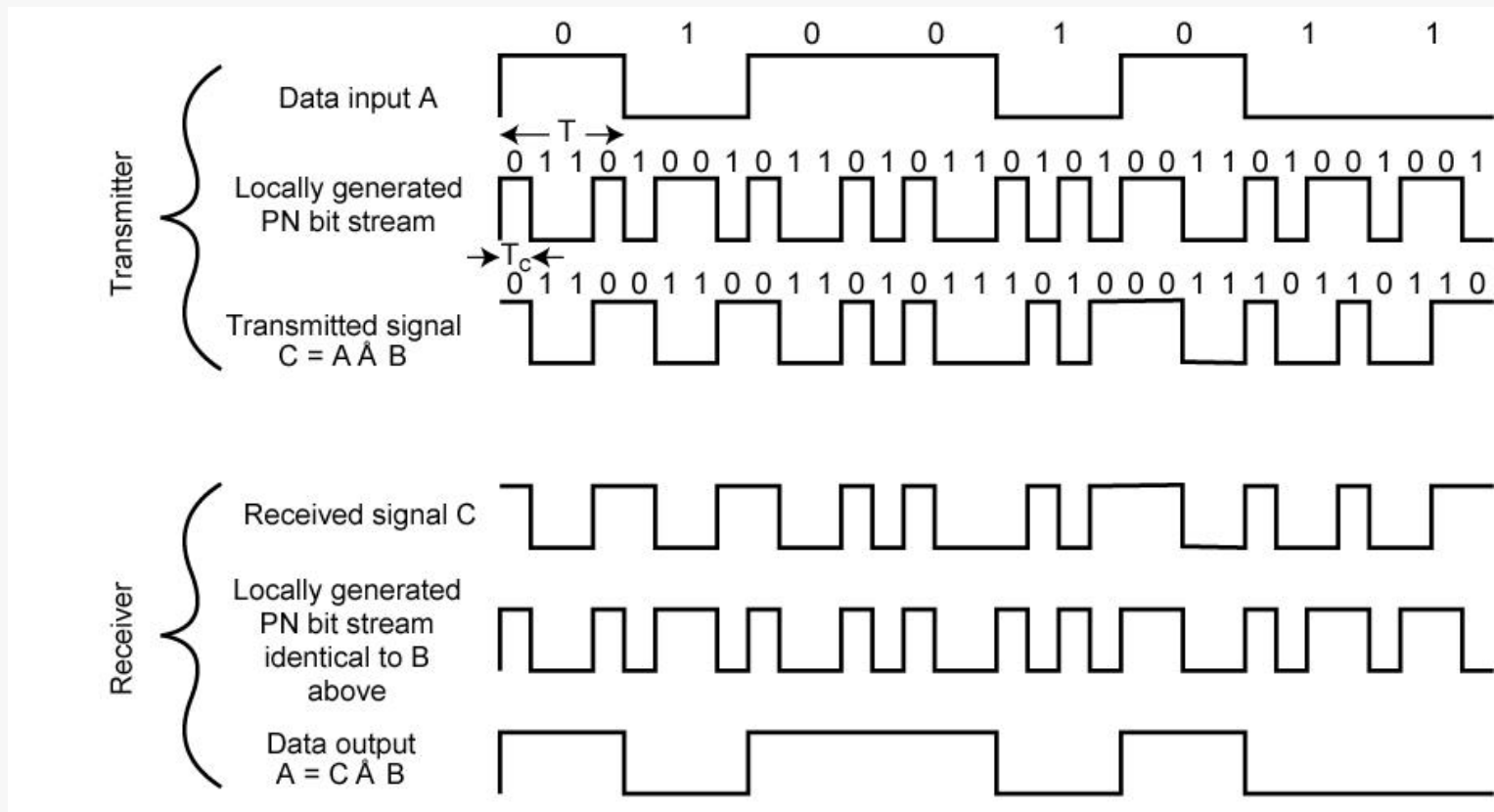
Direct Sequence Spread Spectrum



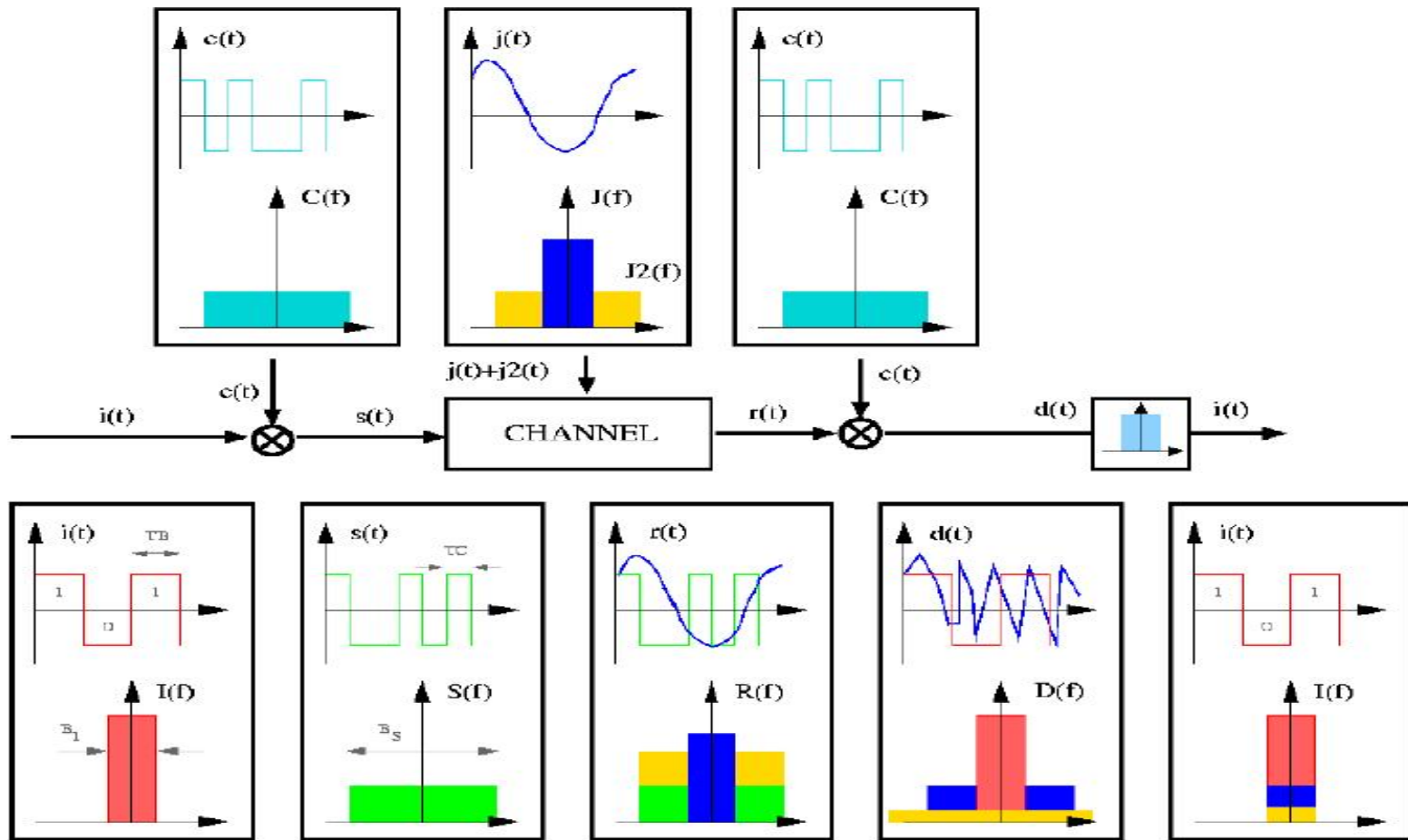
- ❏ **Information signal is directly modulated (multiplied) by a spreading sequences (see next slide)**
- ❏ **Spreading sequence consists of chips each with a duration of t_{chip}**
- ❏ **A set of chips represent a bit; the exact number of chips per bit equals the spreading gain**
- ❏ **Near far effect**
- ❏ **Require continuous bandwidth**

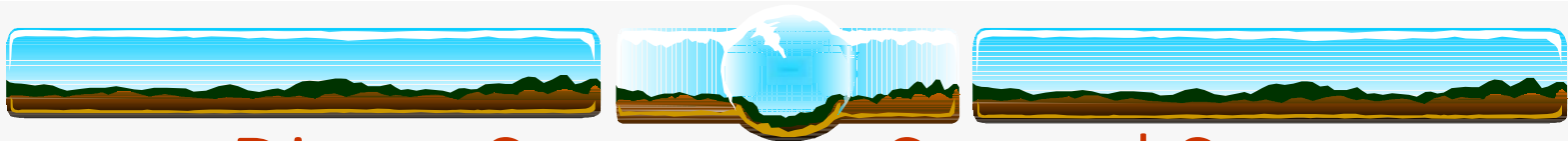


Direct Sequence Spread Spectrum Example

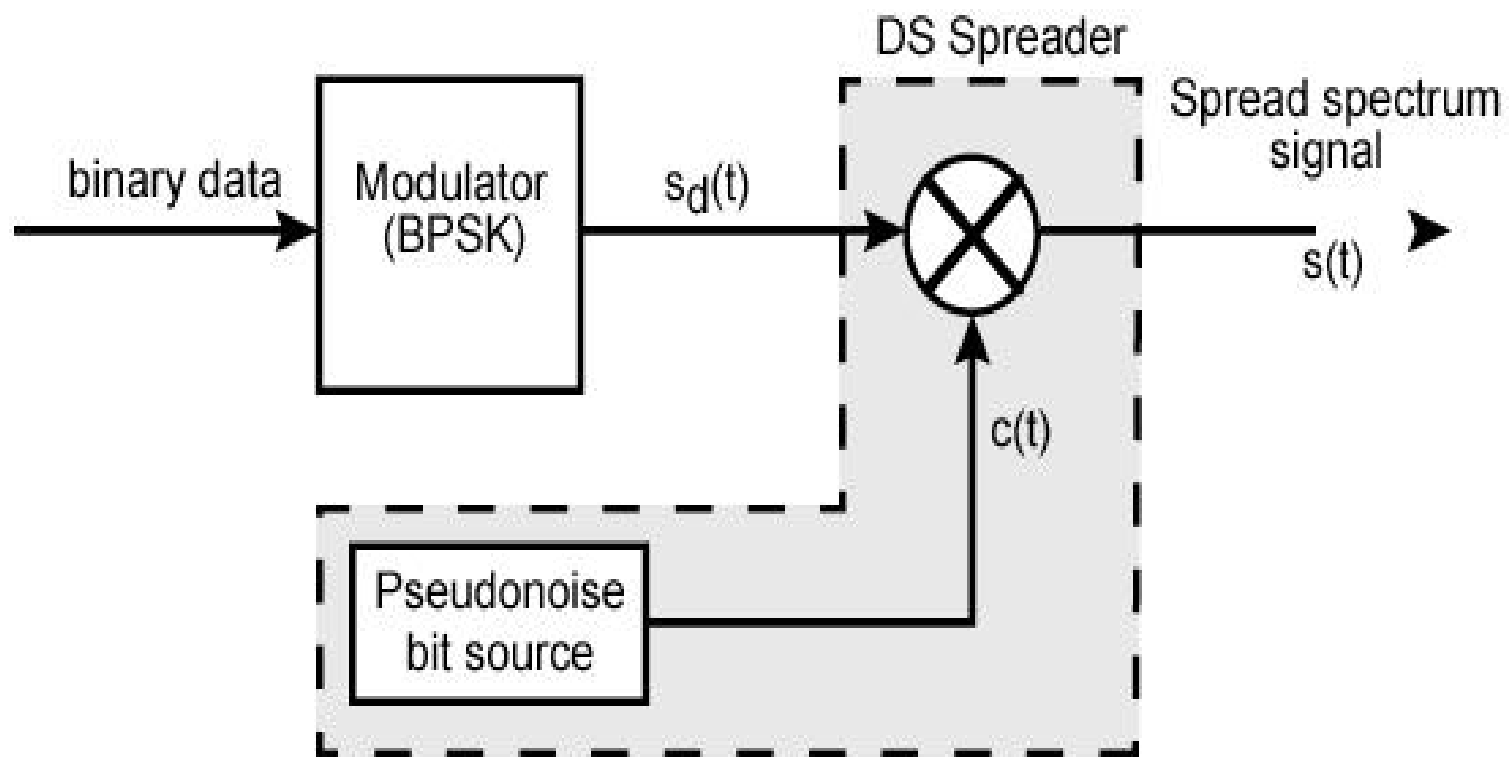


Direct Sequence Spread Spectrum: Transmission Technique



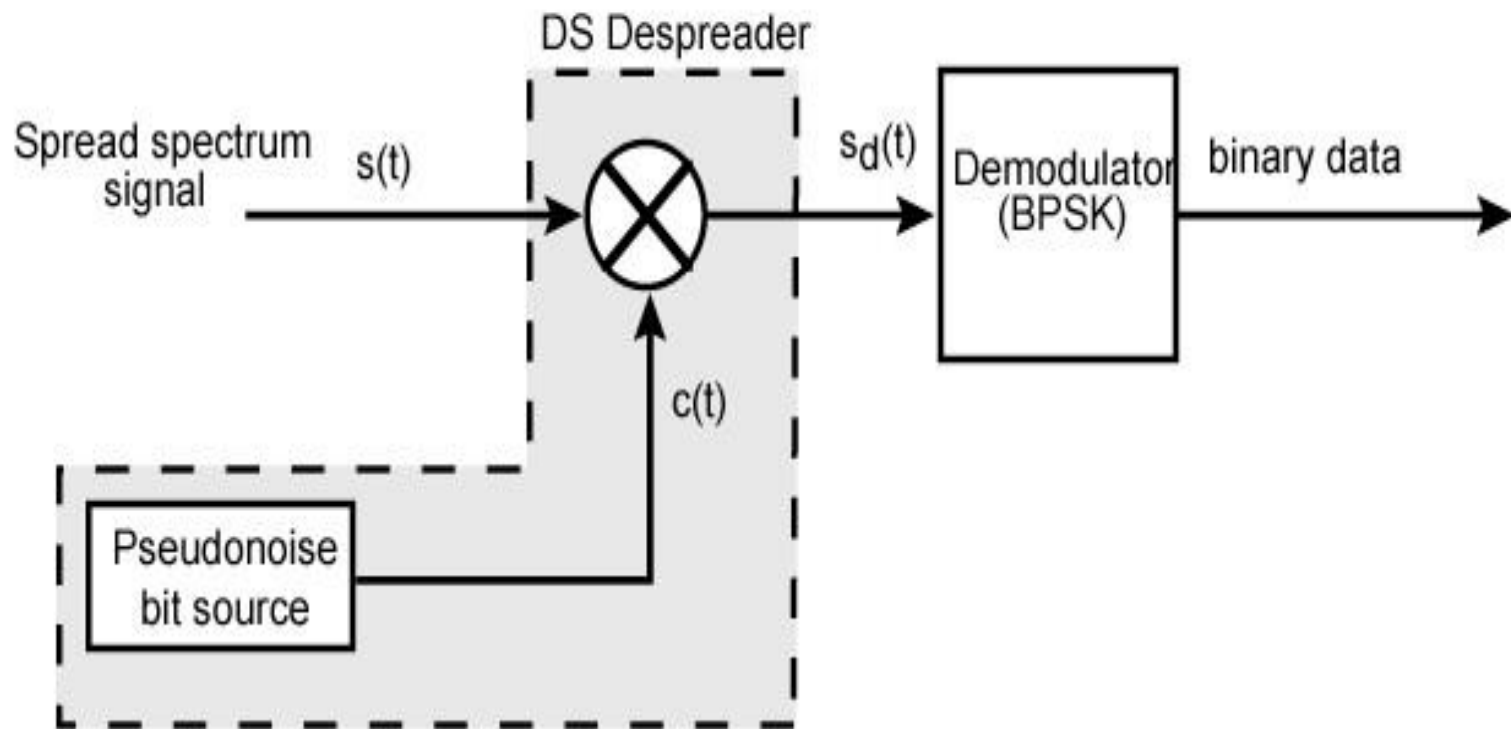


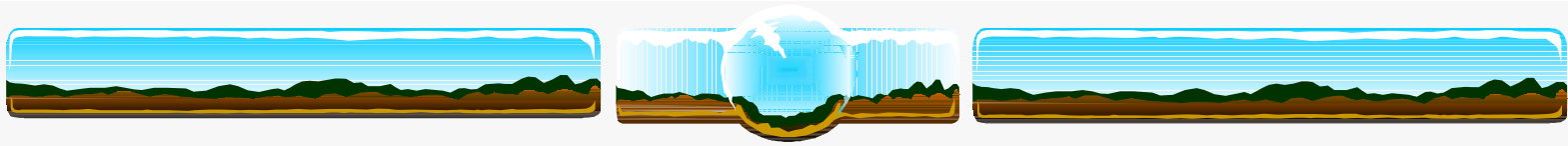
Direct Sequence Spread Spectrum Transmitter



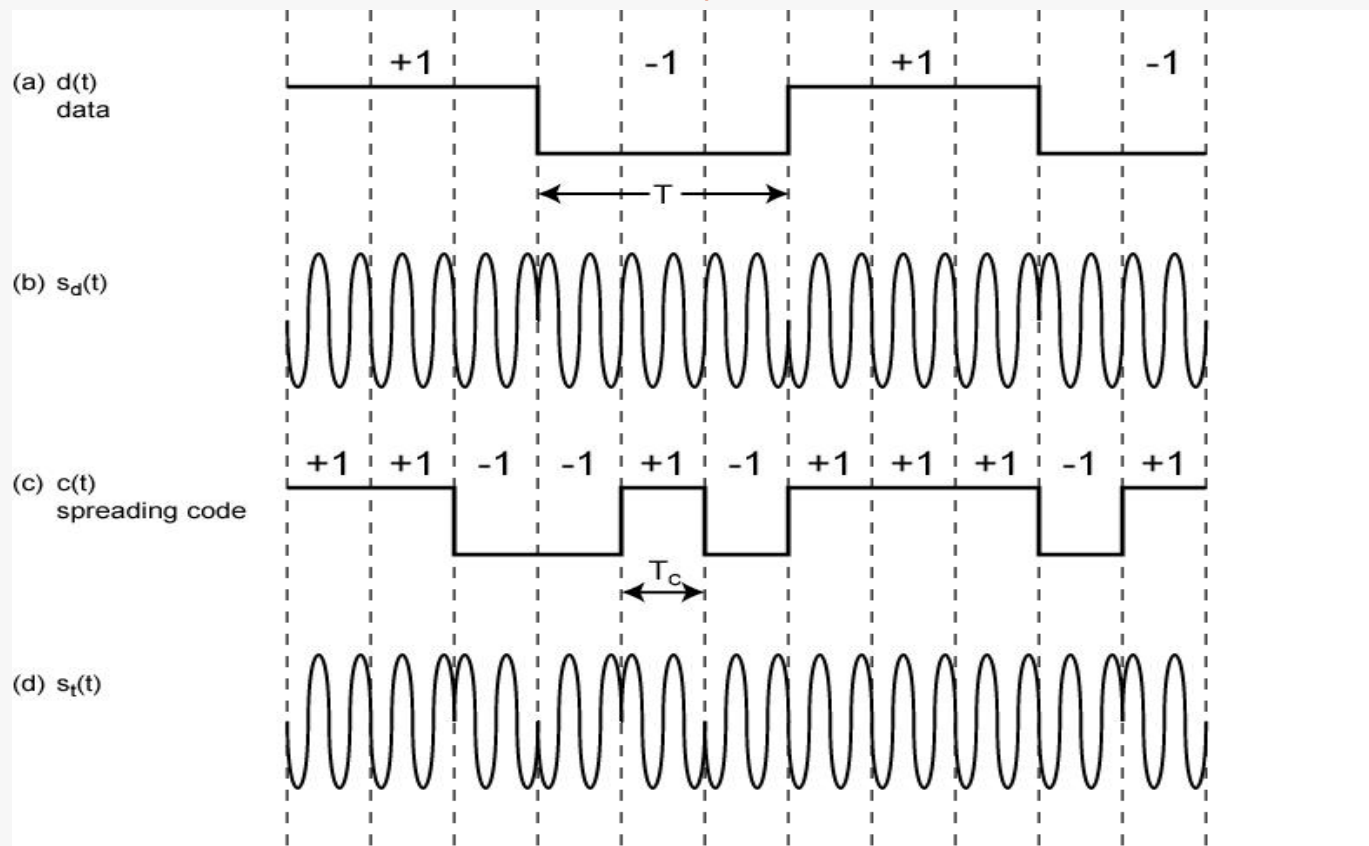


Direct Sequence Spread Spectrum Receiver





Direct Sequence Spread Spectrum Using BPSK Example

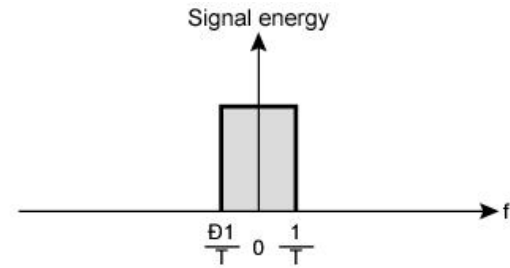




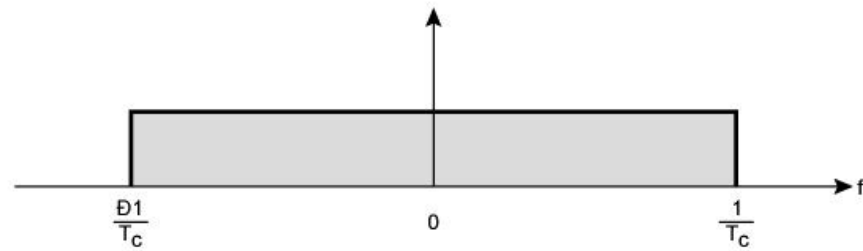
Approximate Spectrum of DSSS Signal

May 28-June 1, 2001

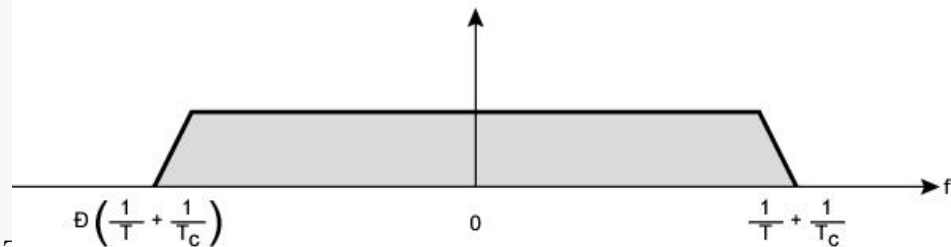
R. Z. 2



(a) Spectrum of data signal



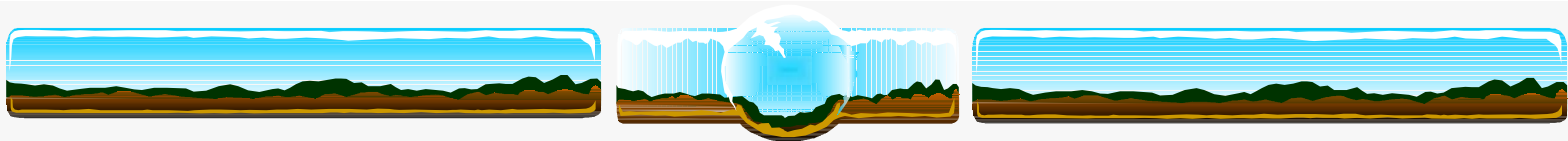
(b) Spectrum of pseudonoise signal



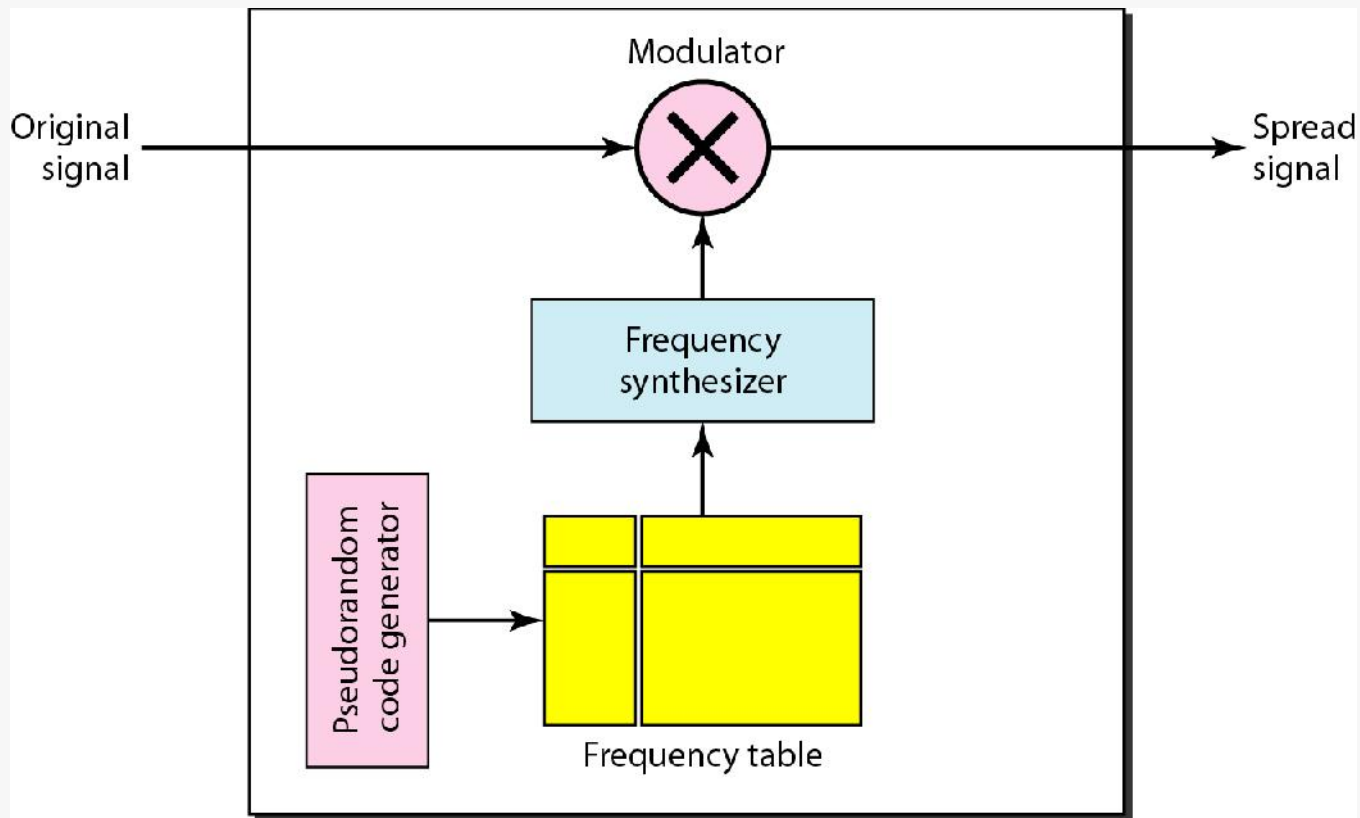
(c) Spectrum of combined signal

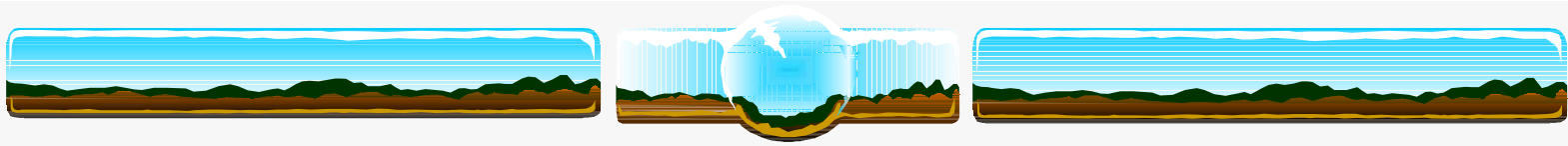


- # The information signal is transmitted on different frequencies
- # Time is divided in slots
- # Each slot the frequency is changed
- # The change of the frequency is referred to as slow if more than one bit is transmitted on one frequency, and as fast if one bit is transmitted over multiple frequencies
- # The frequencies are chosen based on the spreading sequences

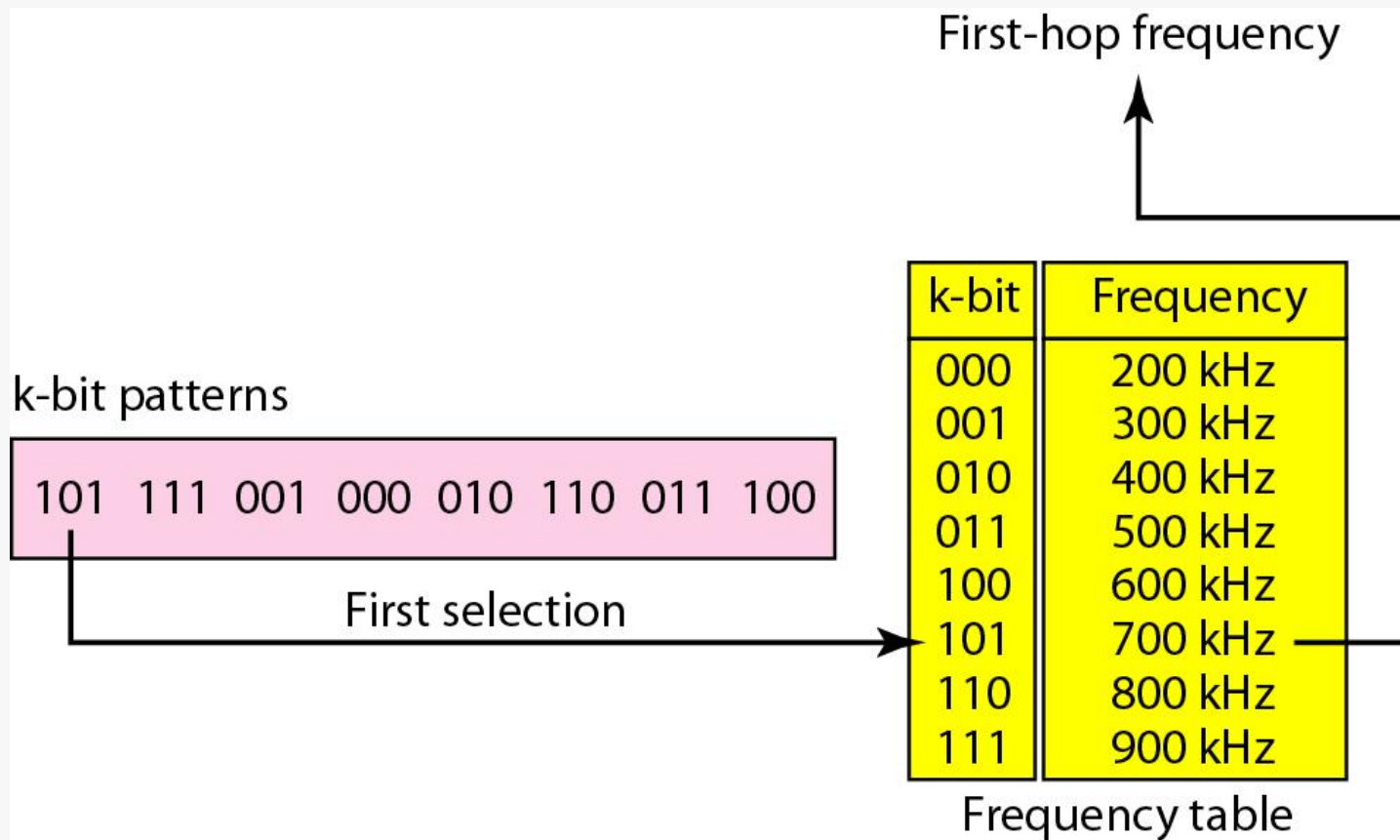


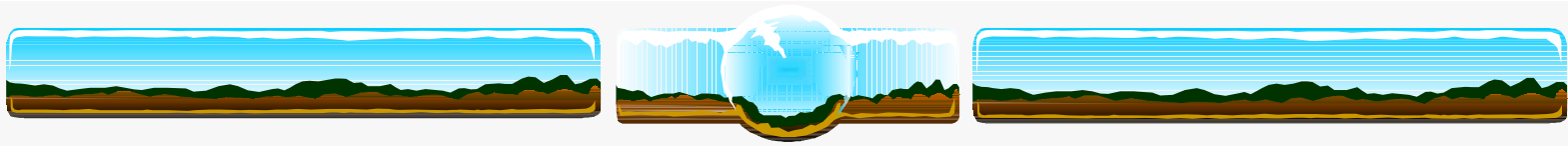
Frequency Hopping Spread Spectrum (FHSS)



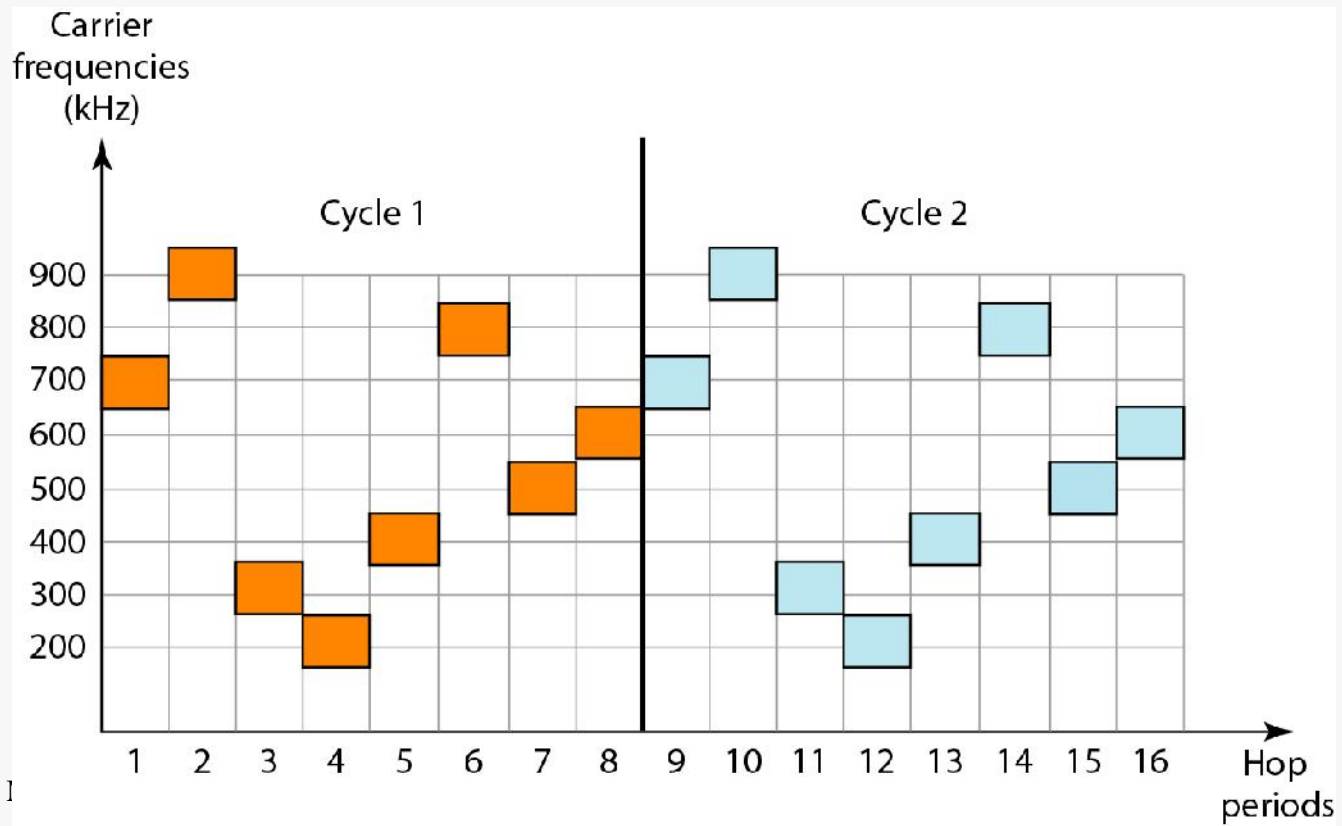


Frequency selection in FHSS

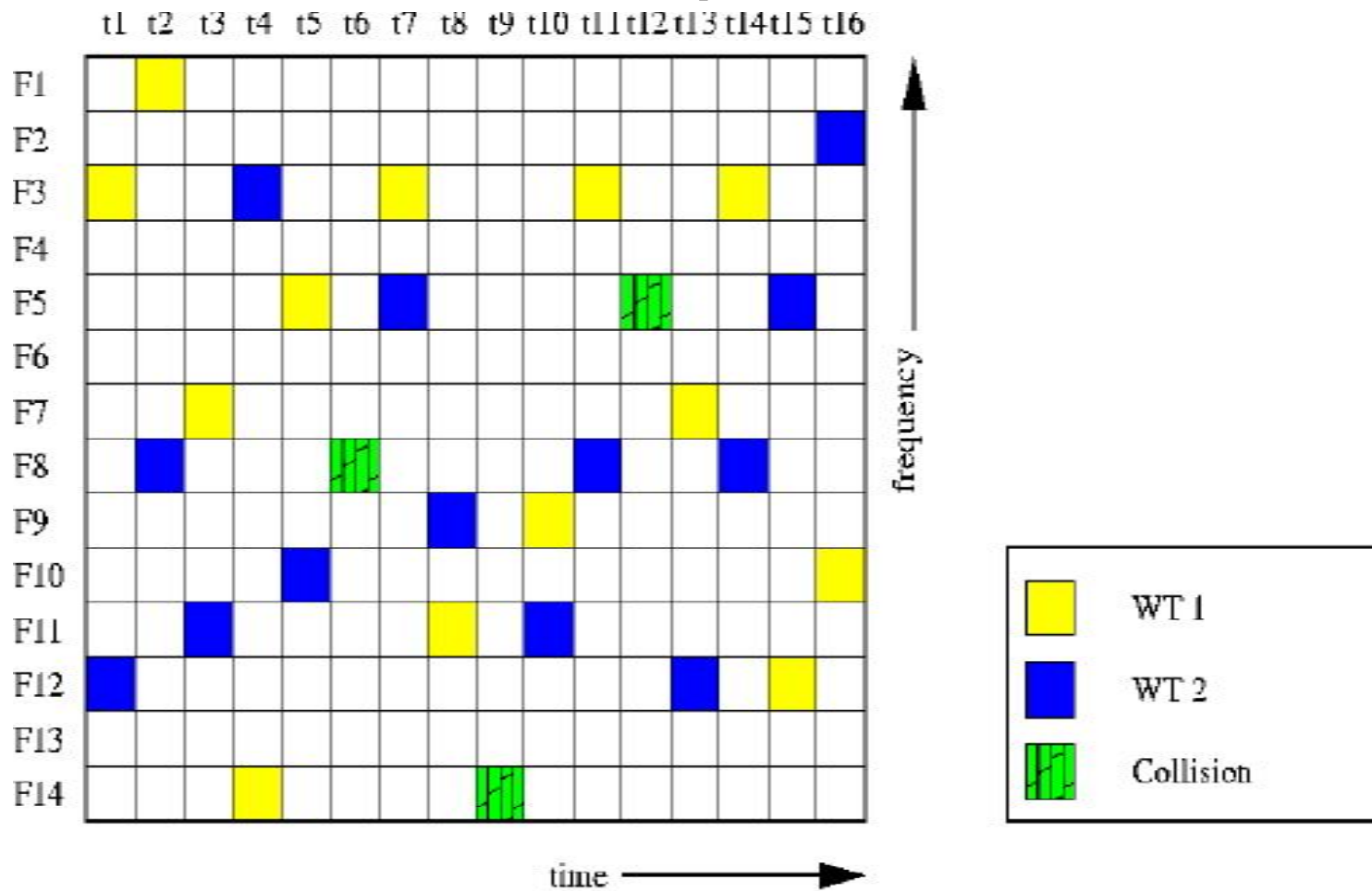


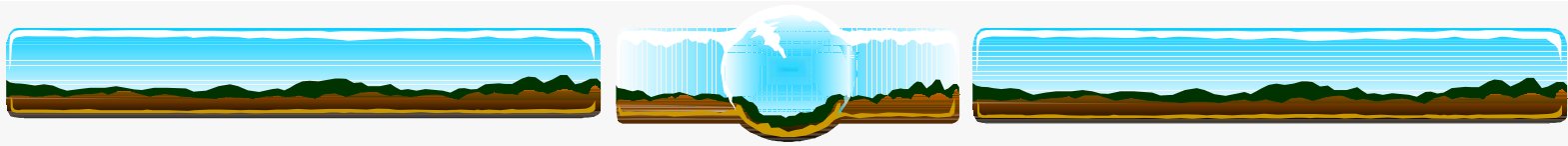


FHSS cycles

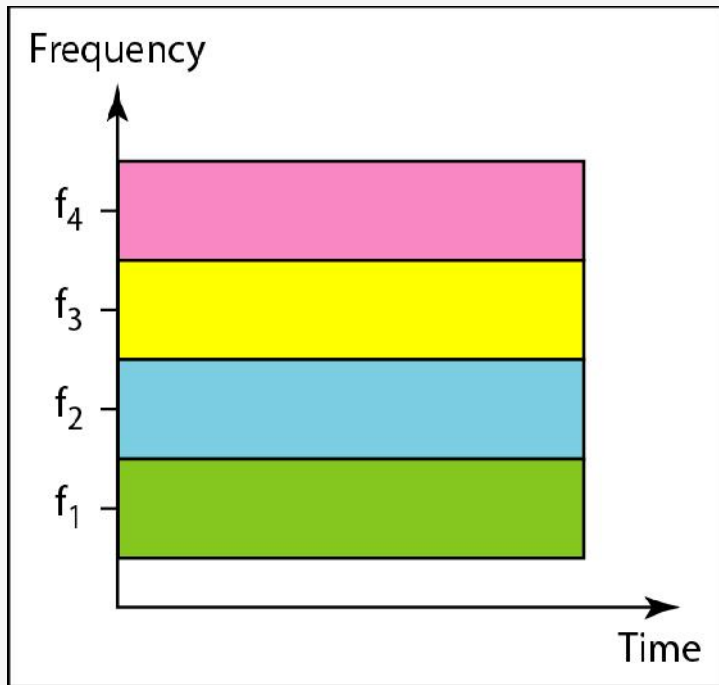


Frequency Hopping Spread Spectrum

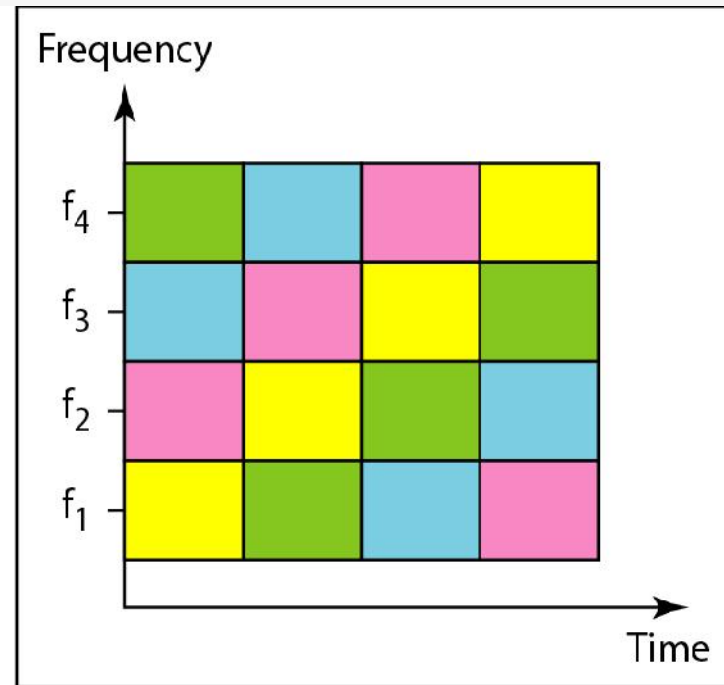




Bandwidth sharing



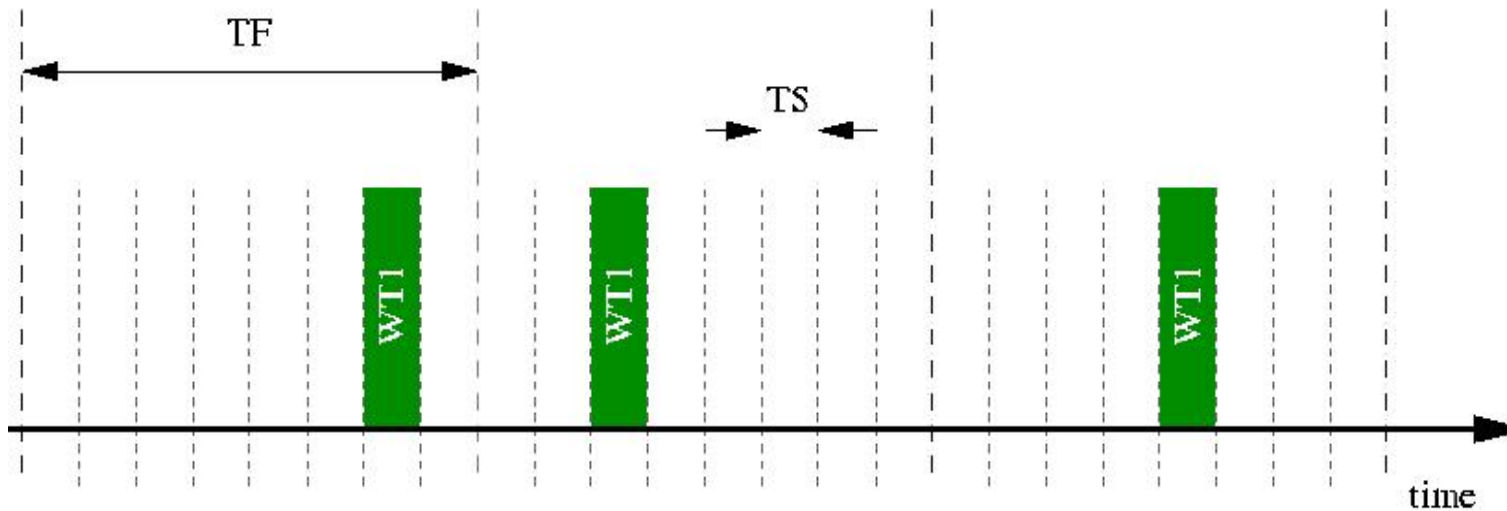
a. FDM



b. FHSS

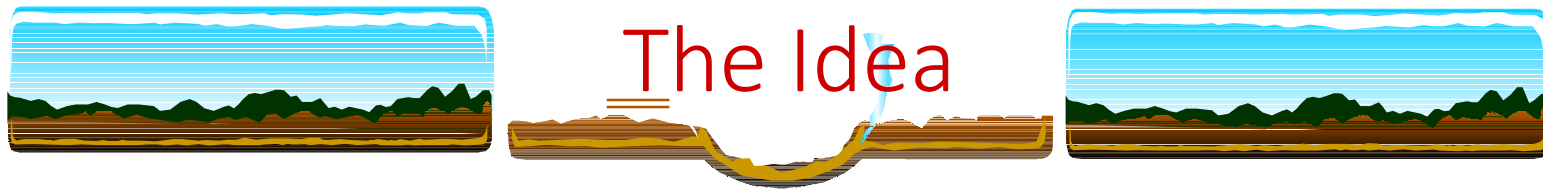


- ❑ Time divided into frames; each TF long
- ❑ Each frame is divided in slots
- ❑ Each wireless terminal send in exactly one of these slots per frame regarding the spreading sequence
- ❑ No near far effect



Comparison of different Spread Spectrum Techniques

SS Technique	advantage	disadvantage
Direct Sequence	<ul style="list-style-type: none"> Best behavior in multipath rejection no synchronization simple implementation difficult to detect 	<ul style="list-style-type: none"> near far effect coherent bandwidth
Frequency Hopper	<ul style="list-style-type: none"> no need for coherent bandwidth less affected by the near far effect 	<ul style="list-style-type: none"> complex hardware error correction needed
Time Hopper	<ul style="list-style-type: none"> high bandwidth efficiency less complex hardware less affected by the near far effect 	<ul style="list-style-type: none"> error correction needed



(a)



(b)

Fig. 1 – (a) A Regular-FDM single carrier – A whole bunch of water coming all in one stream. (b) Orthogonal-FDM – Same amount of water coming from a lot of small streams.



Fig. 2 – All cargo on one truck vs. splitting the shipment into more than one.

In MCM , we split the data in to different streams and transmit using separate Sub Carriers.

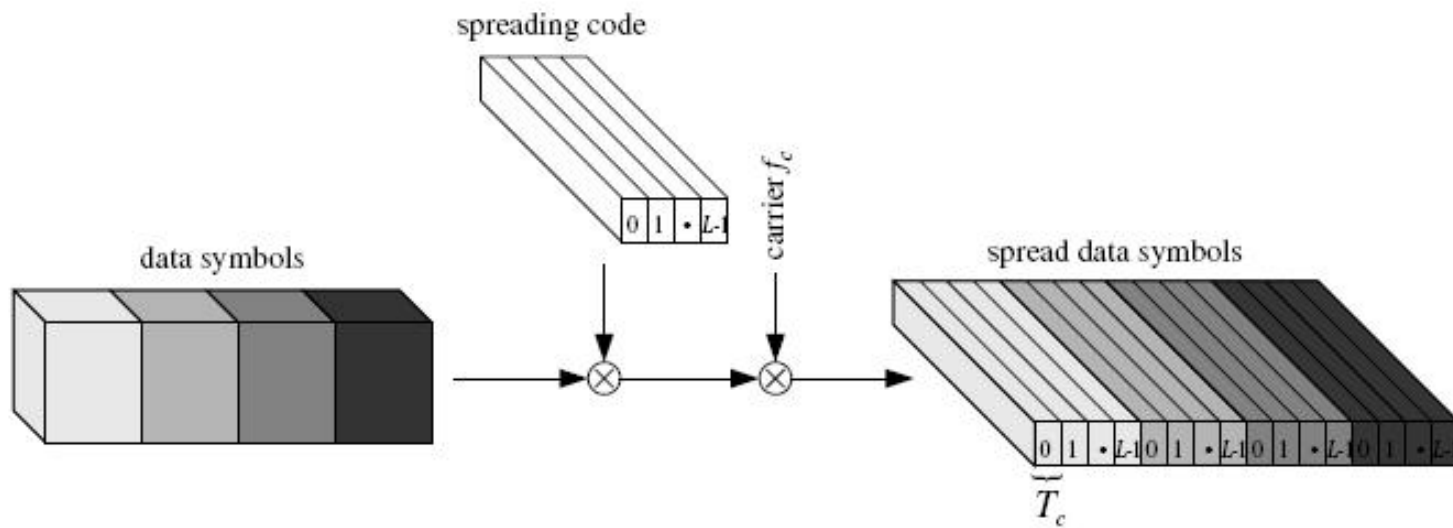
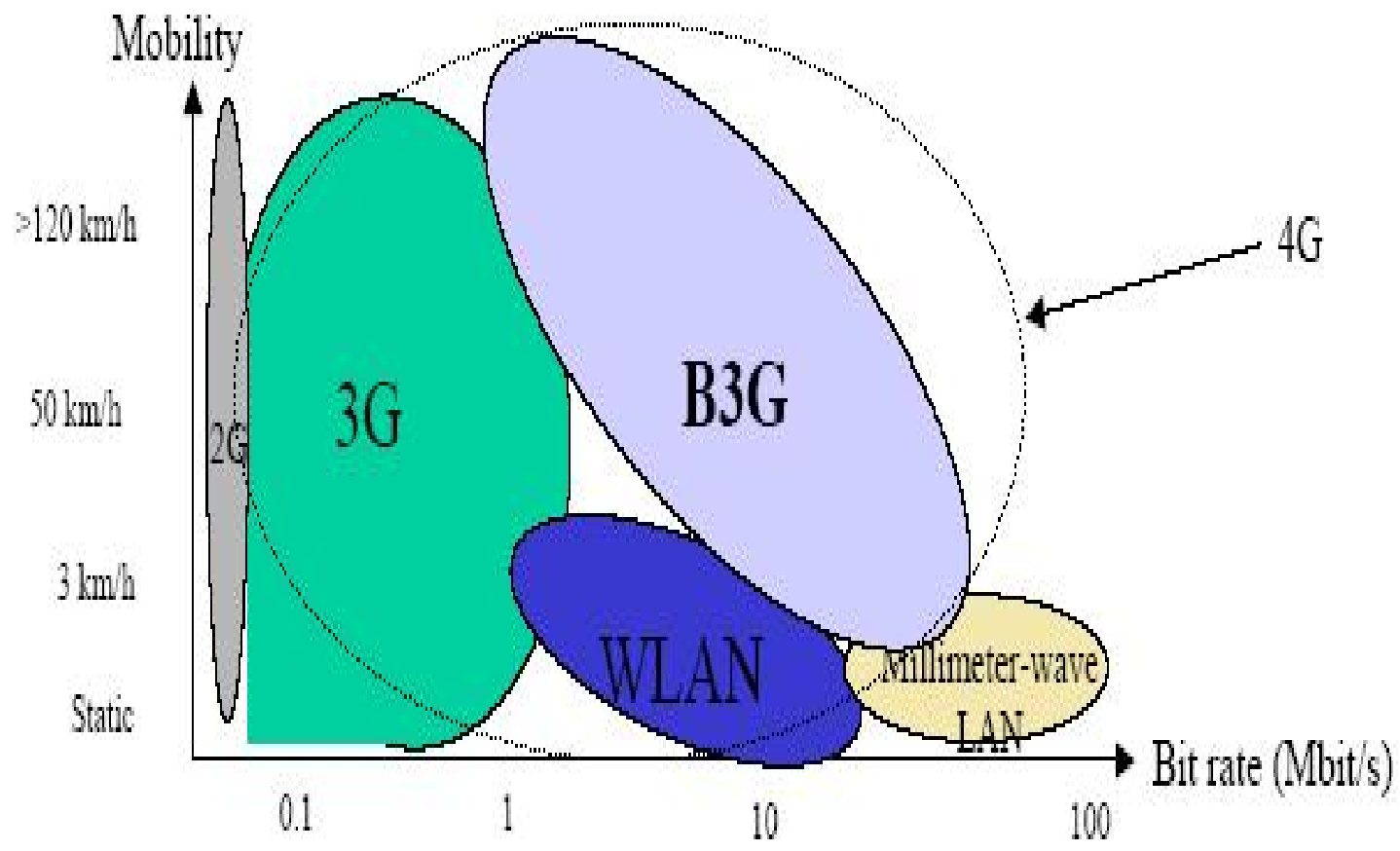


Figure: The Principle of DS-CDMA



Thank You



COMPUTER NETWORKS – Unit 1

Topic 13

TRANSMISSION MEDIA

Transmission Media

- **Guided Media**
- **Unguided Media**

Figure 7-1

Electromagnetic Spectrum

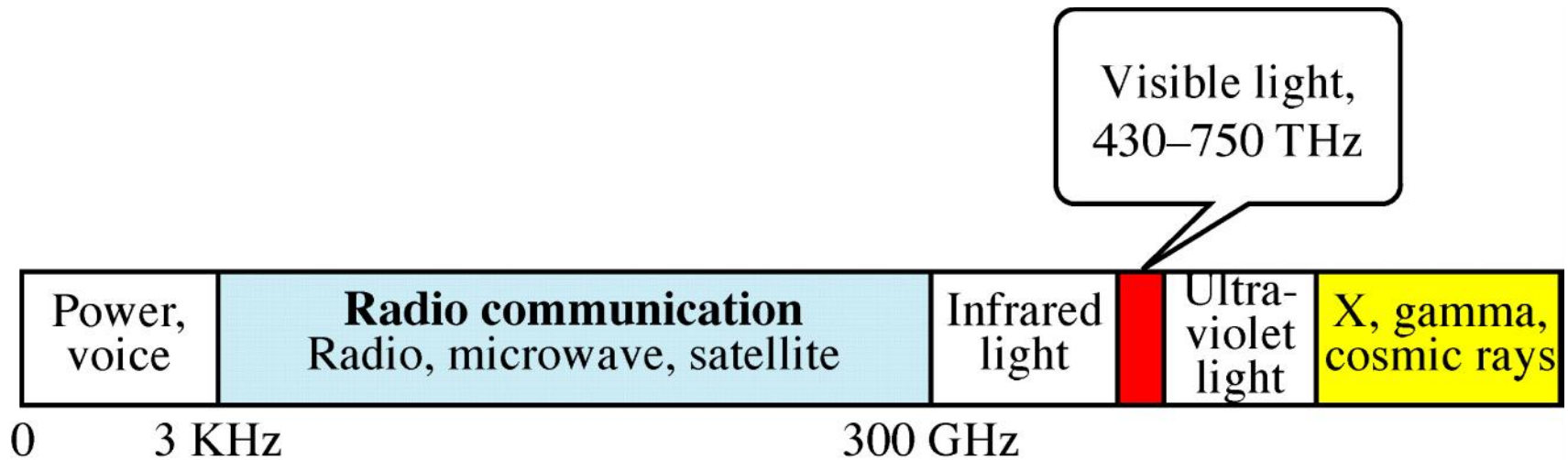


Figure 7-2

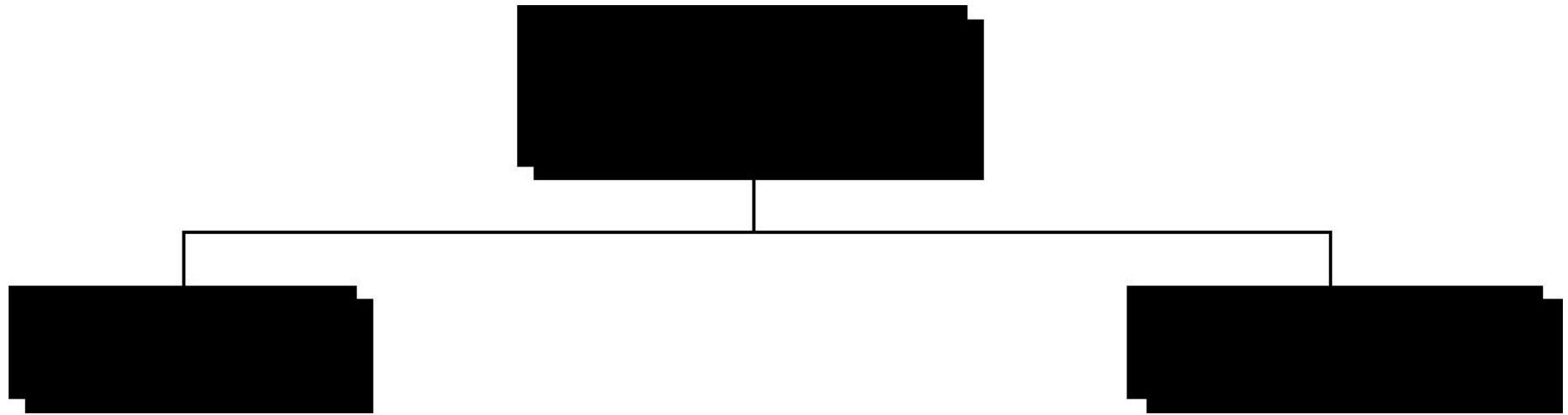


Figure 7-3

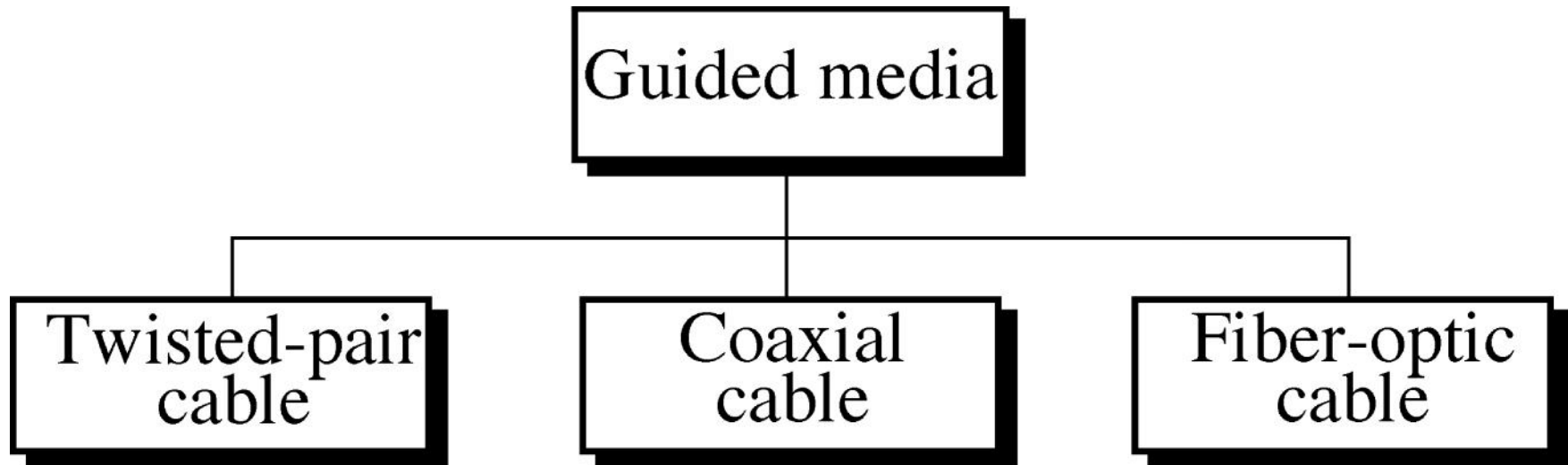


Figure 7-4 and 7-5

Twisted-Pair Cable

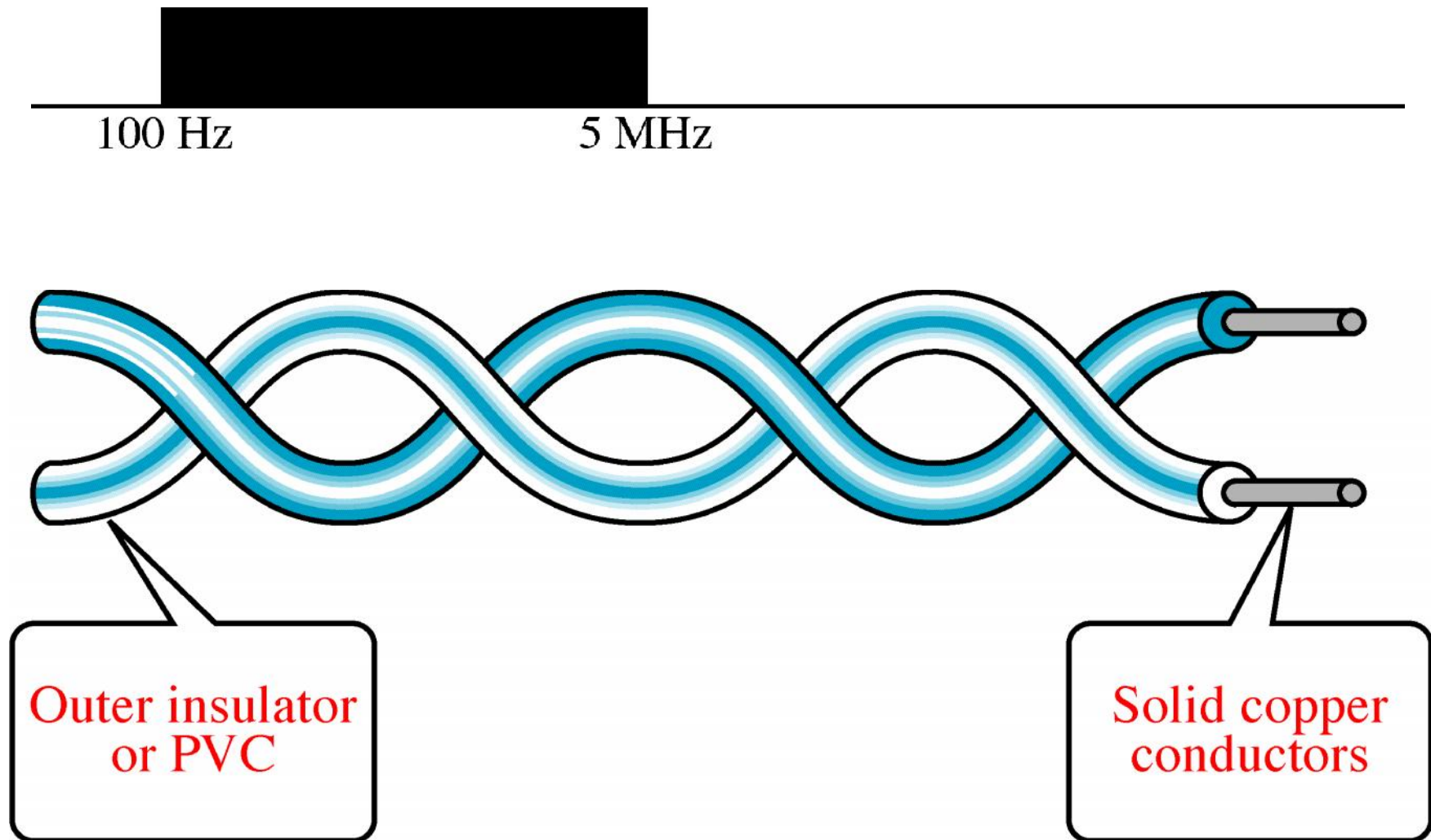


Figure 7-6

Effect of Noise on Parallel Lines

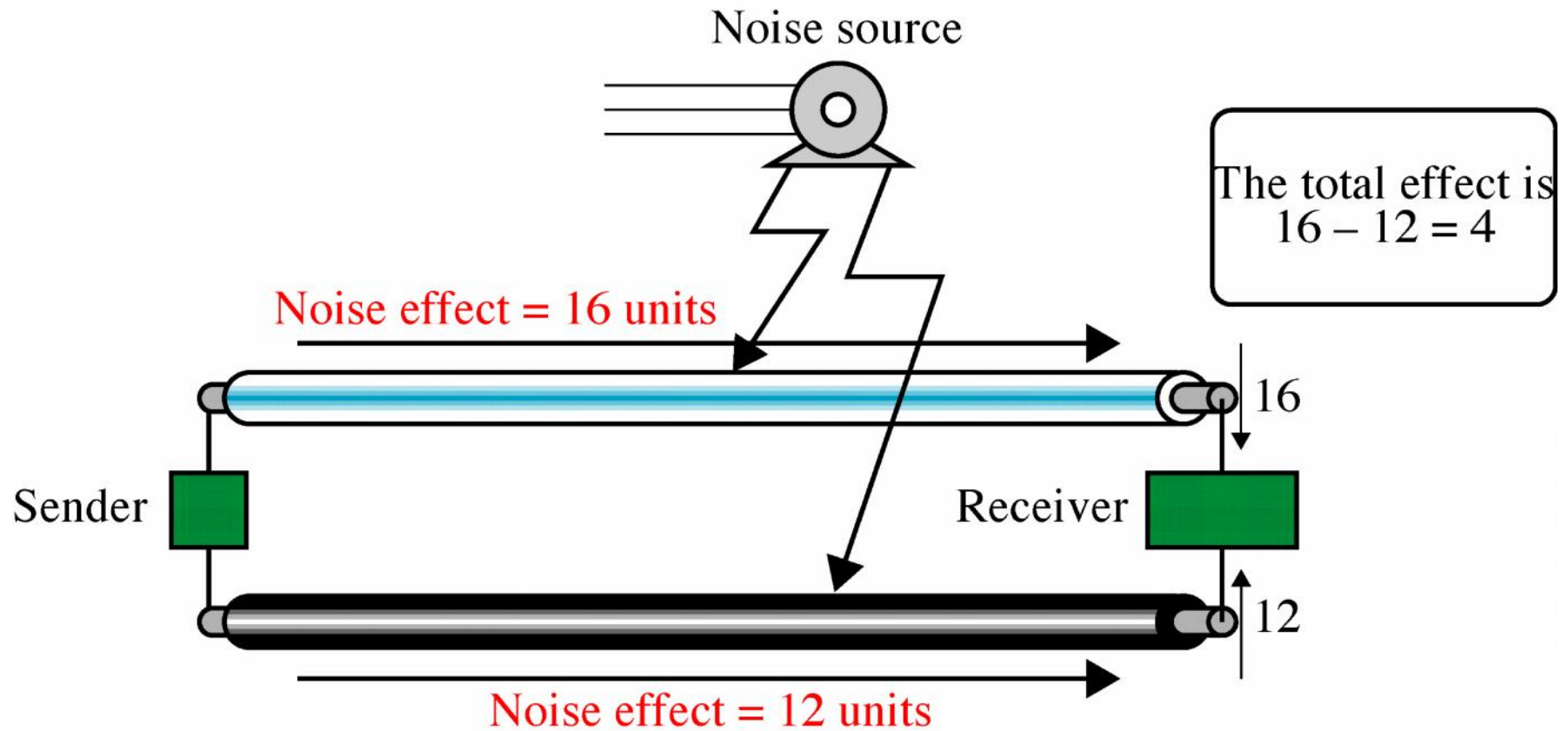


Figure 7-7

Noise on Twisted-Pair Lines

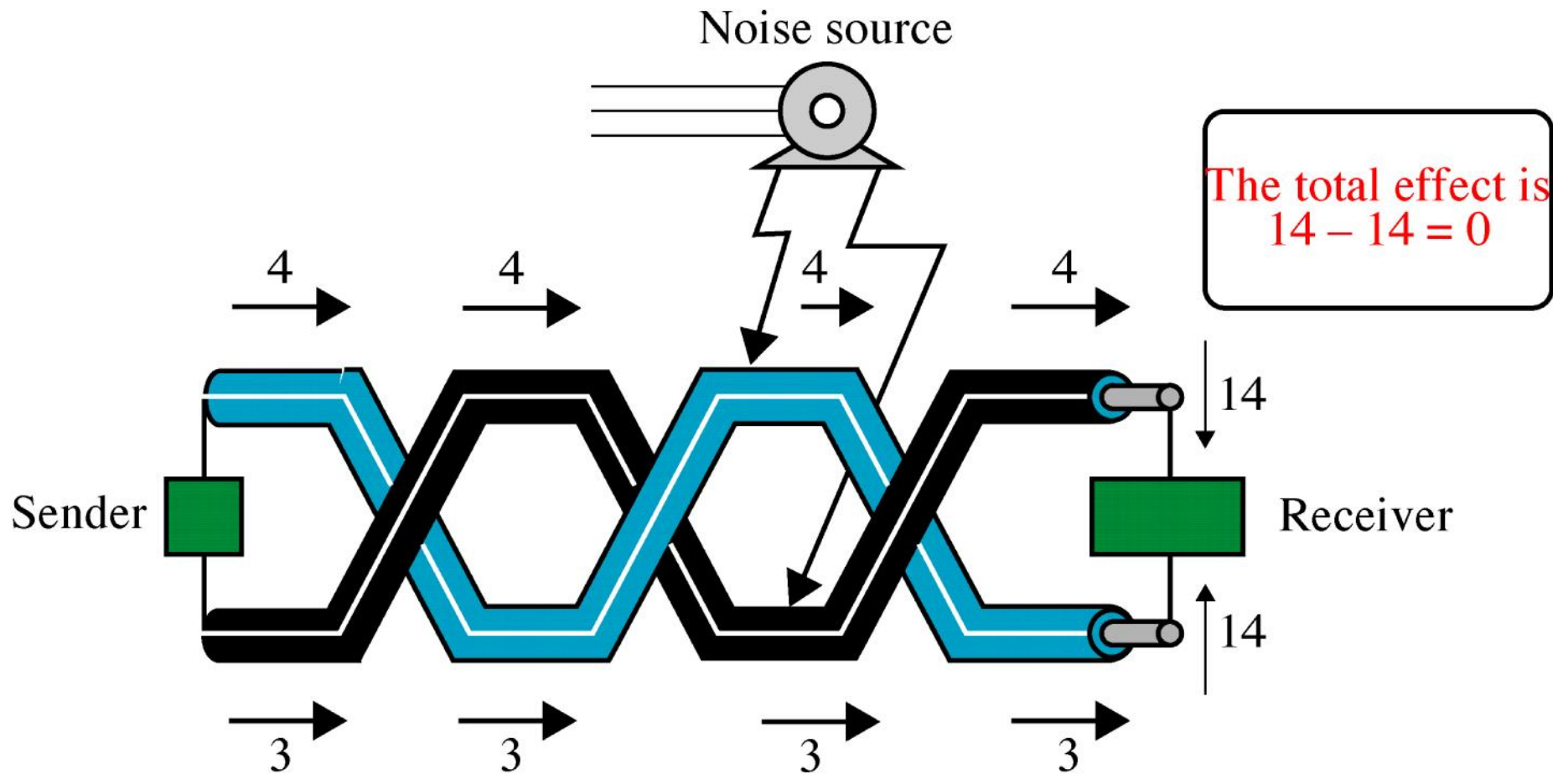


Figure 7-8

Unshielded Twisted-Pair Cable

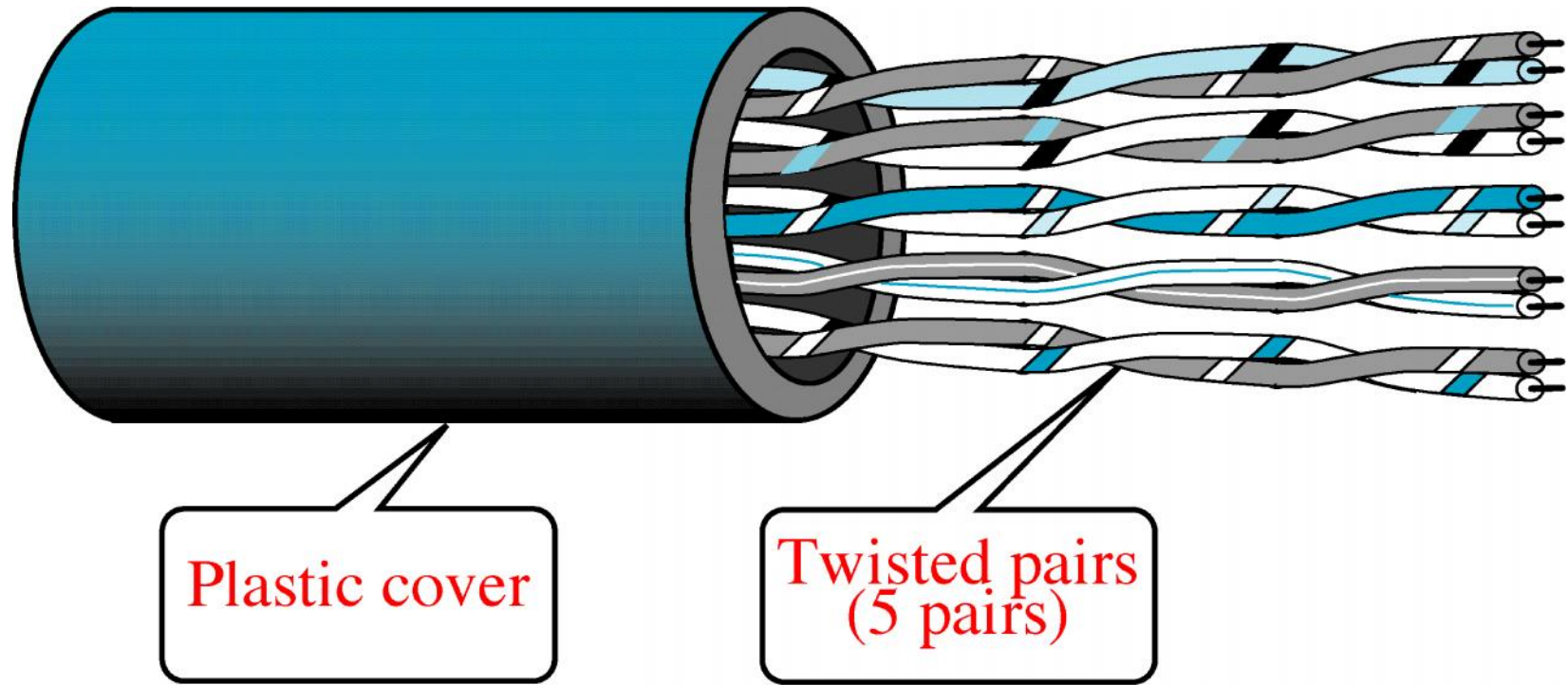
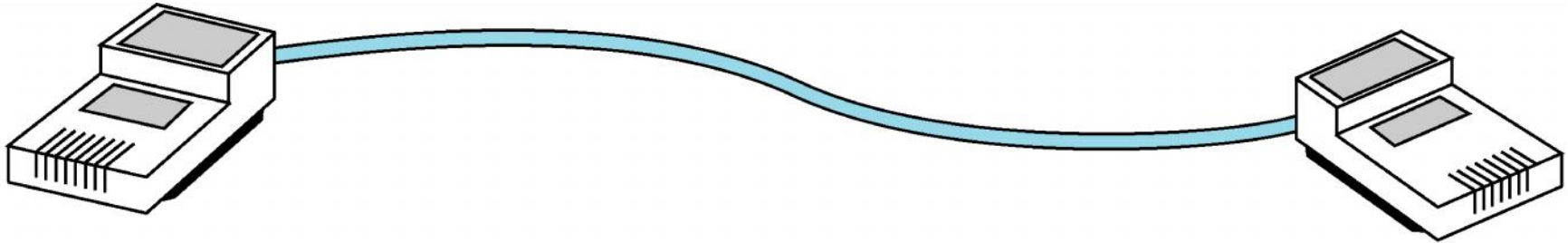
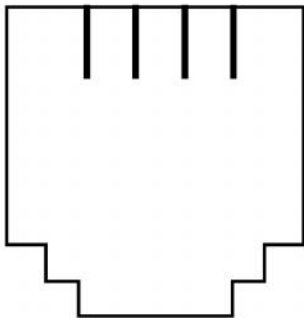


Figure 7-9

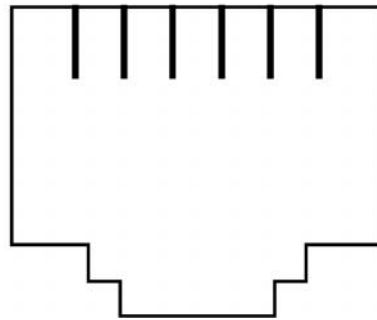
UTP Connectors



4-conductor



6-conductor



8-conductor

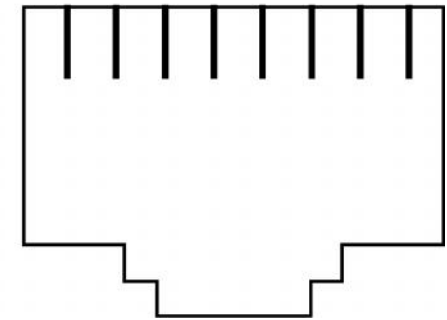


Figure 7-10

Shielded Twisted-Pair Cable

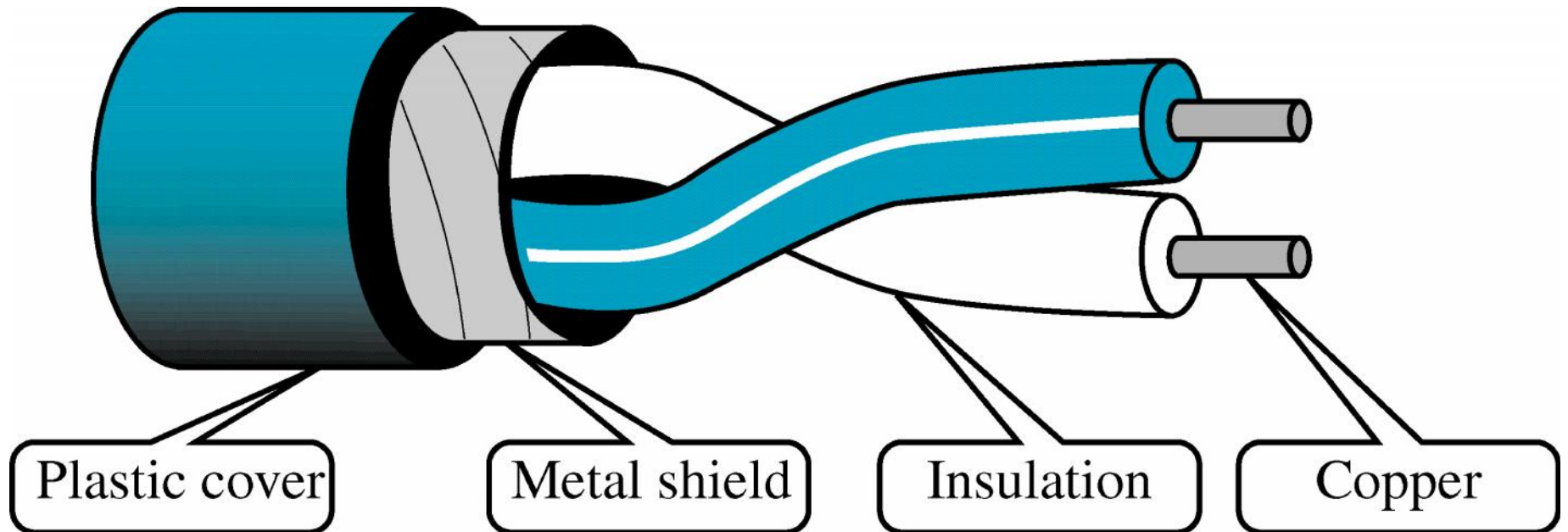


Figure 7-11 and 7-12

Coaxial Cable

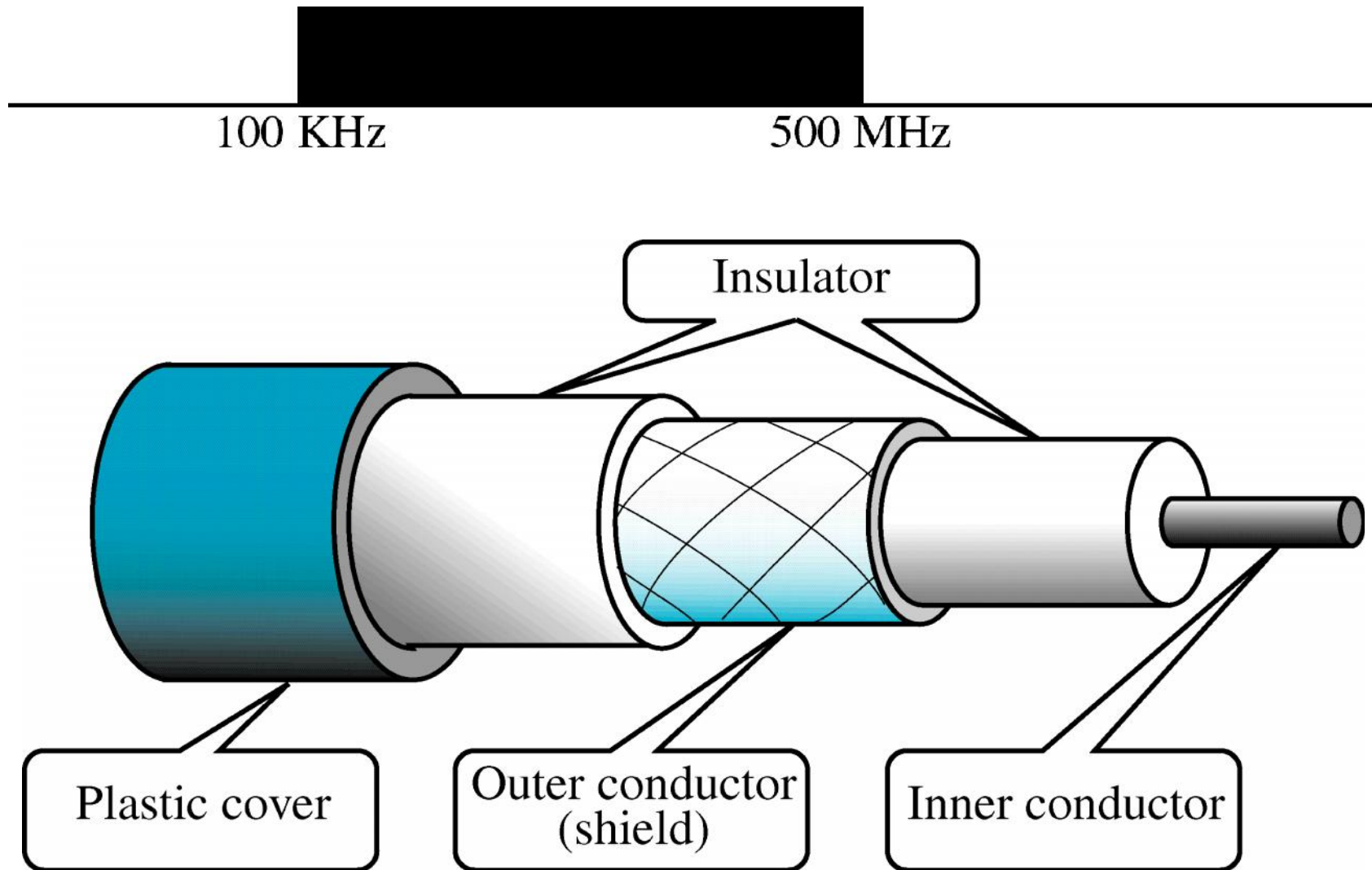
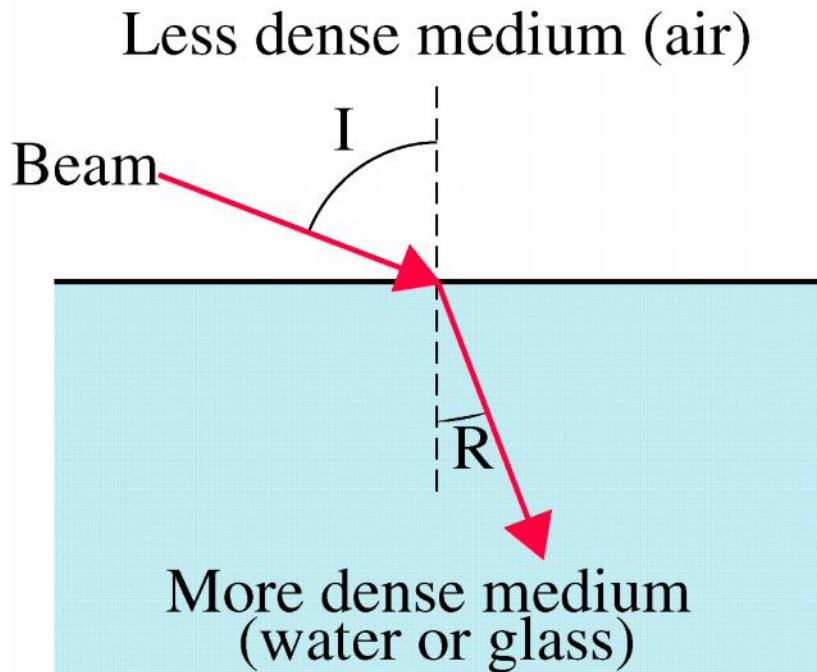
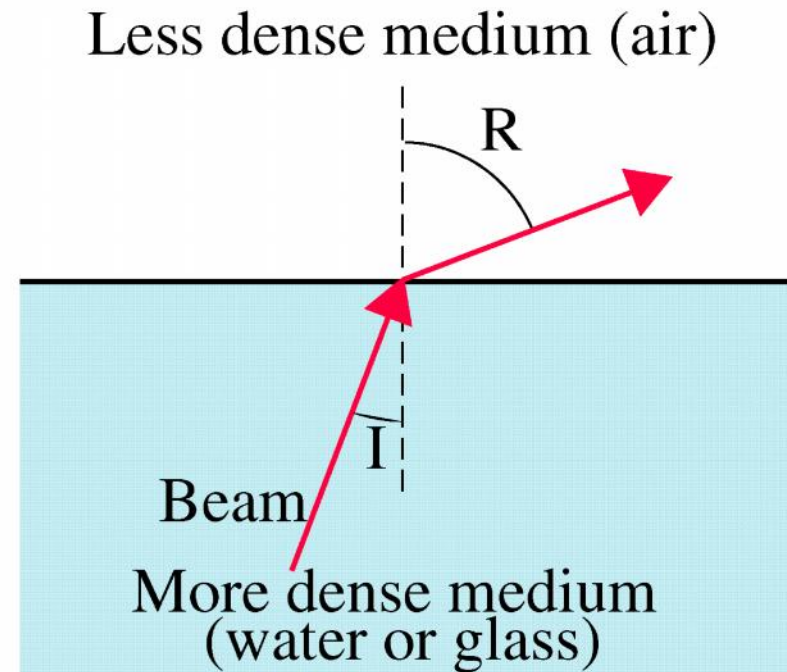


Figure 7-13

Refraction



a. From less dense to more dense medium



b. From more dense to less dense medium

Figure 7-14

Critical Angle

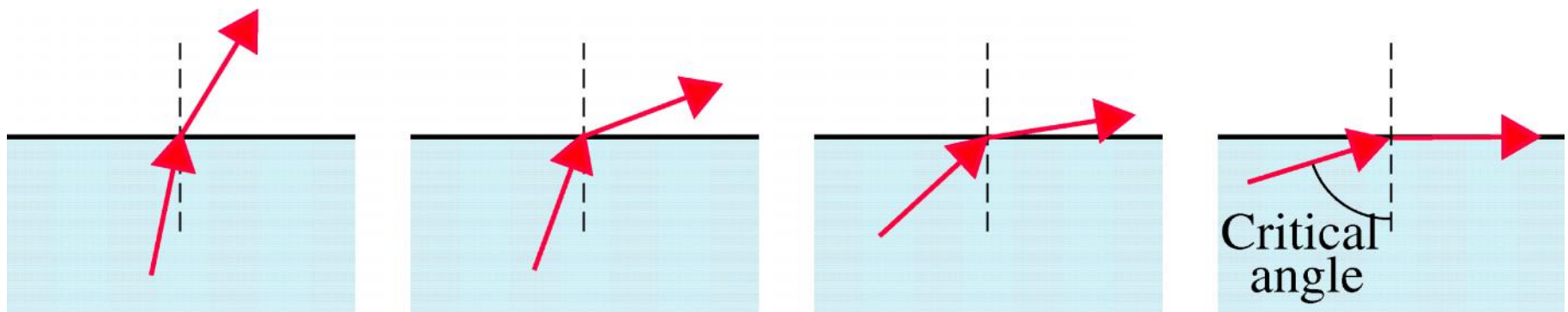


Figure 7-15

Reflection

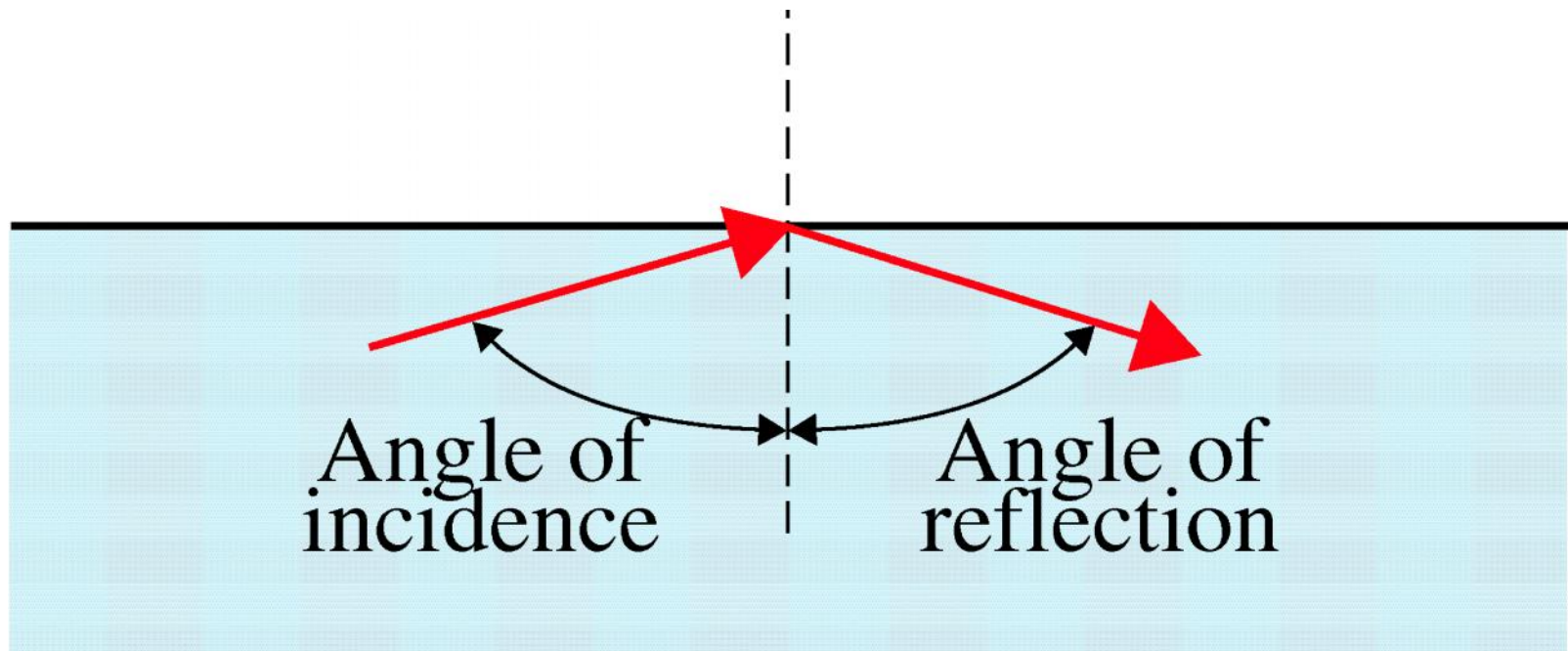


Figure 7-16

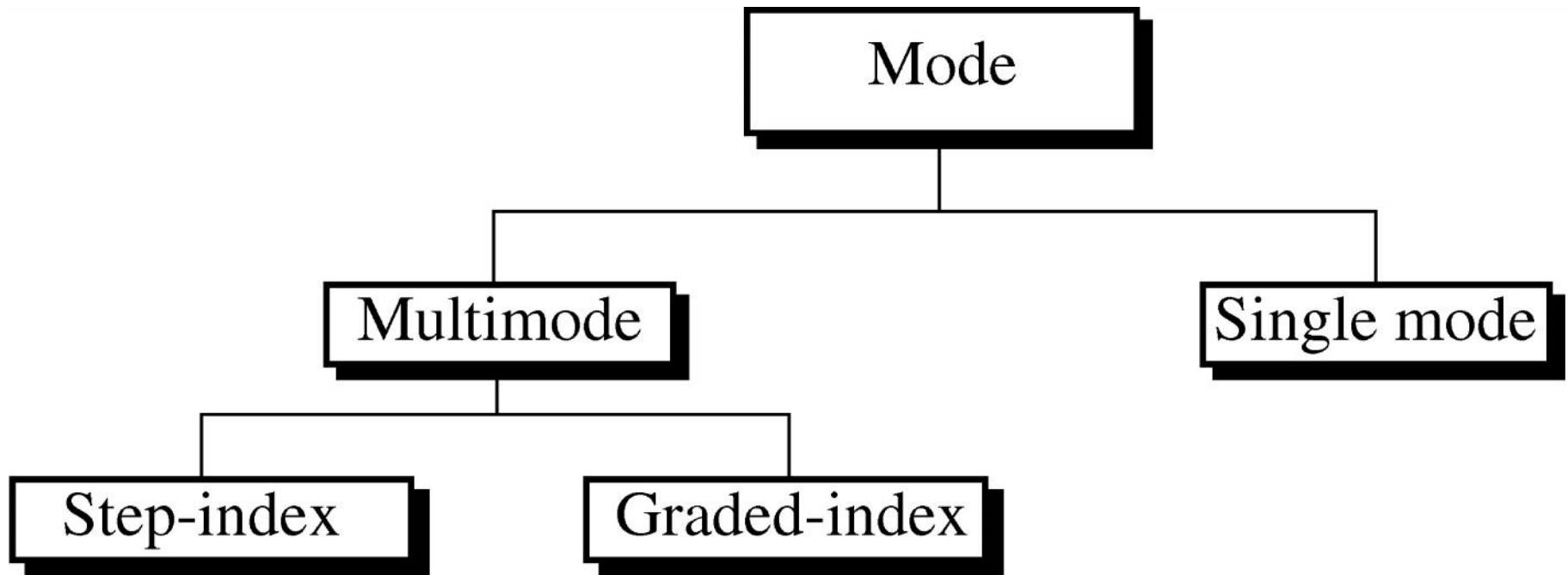


Figure 7-17

Multimode Step-Index

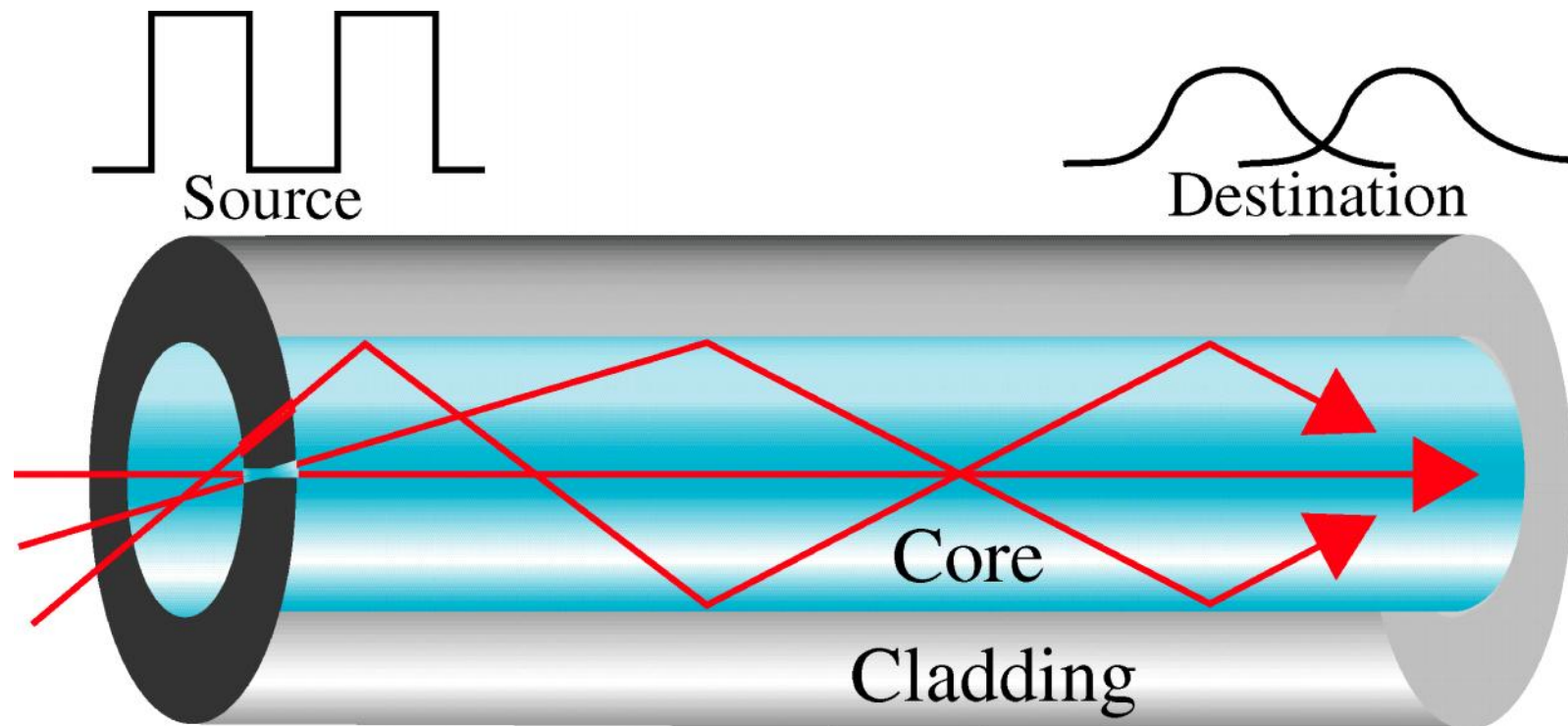


Figure 7-18

Multimode Graded-Index



Figure 7-19

Single Mode

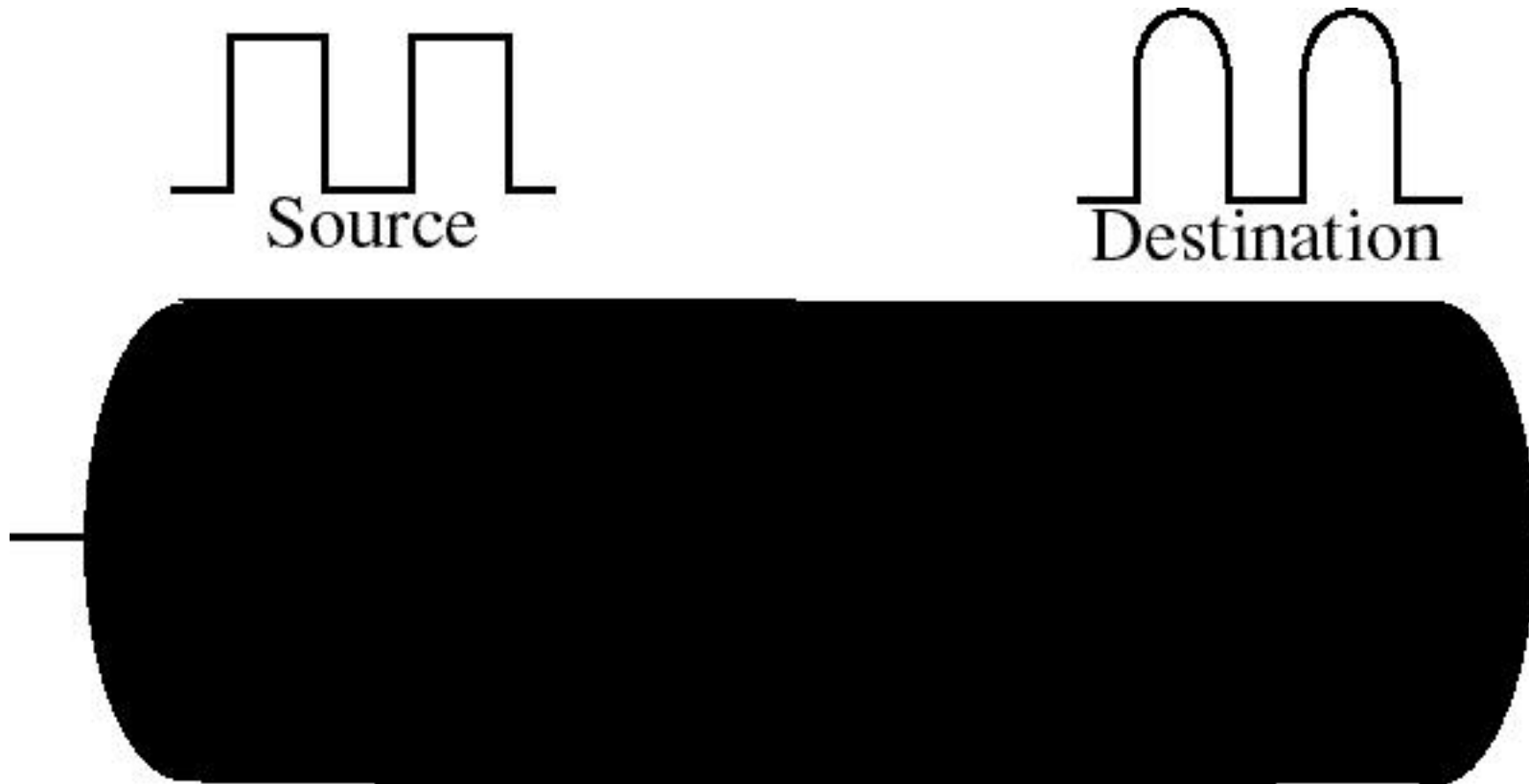


Figure 7-20

Fiber Construction

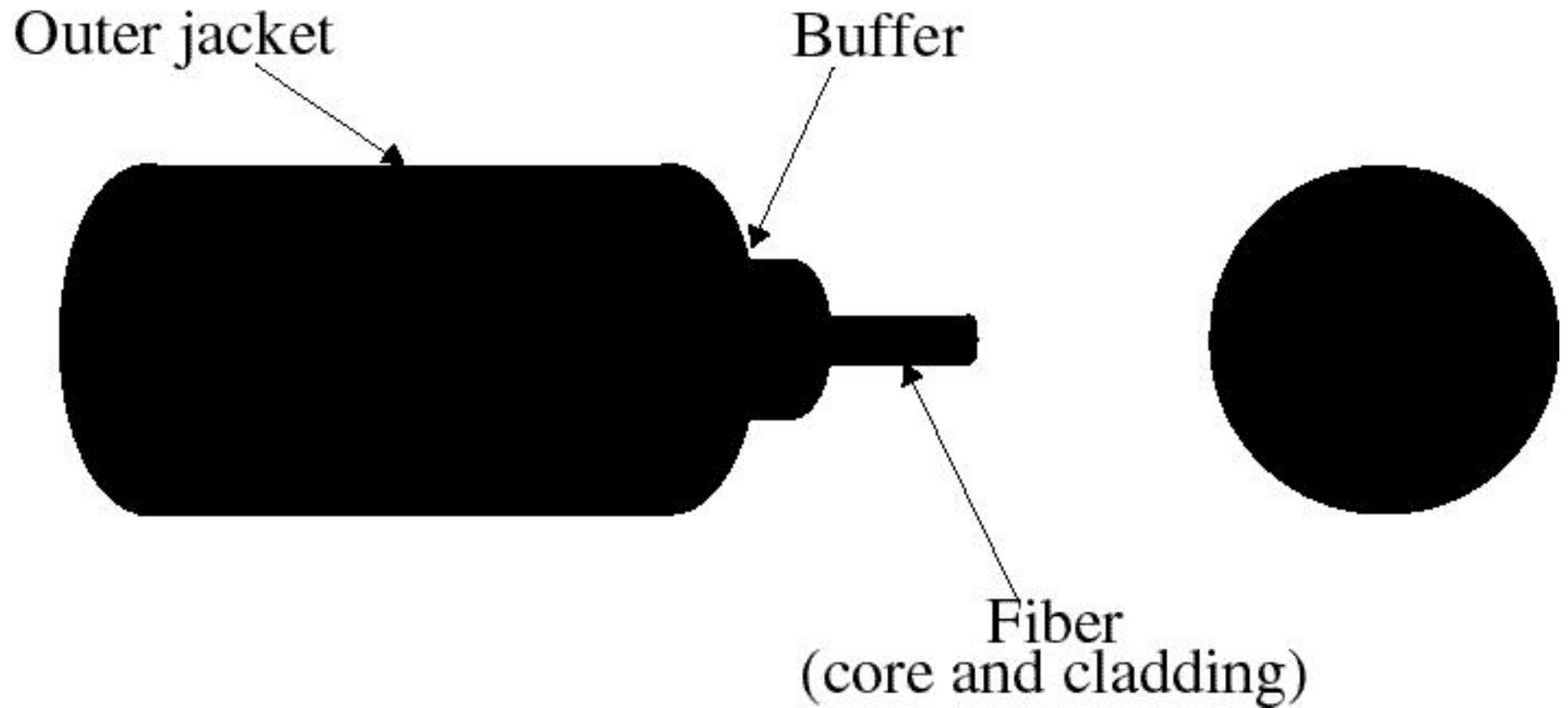


Figure 7-21

Radio Communication Band

VLF Very low frequency
LF Low frequency
MF Middle frequency
HF High frequency

VHF Very high frequency
UHF Ultra high frequency
SHF Super high frequency
EHF Extremely high frequency

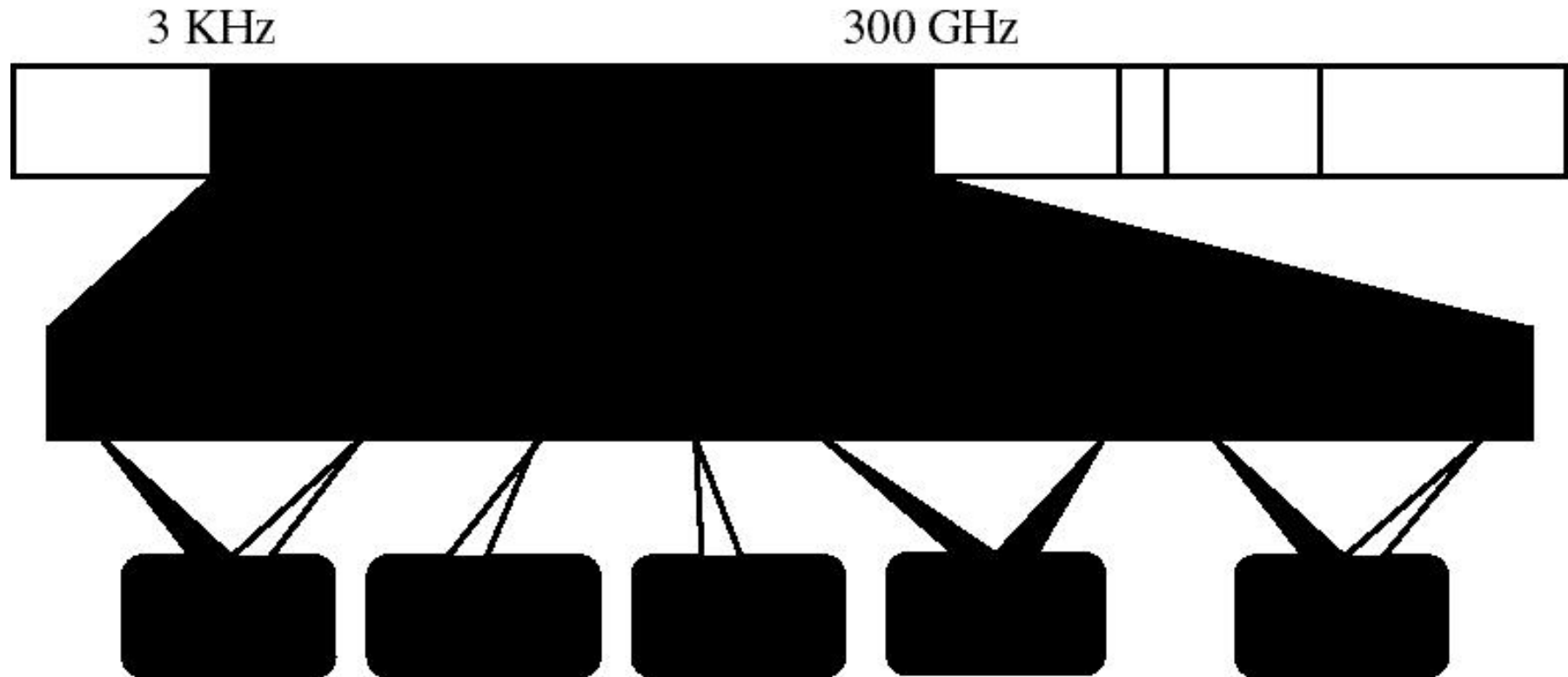


Figure 7-22

Propagation Types

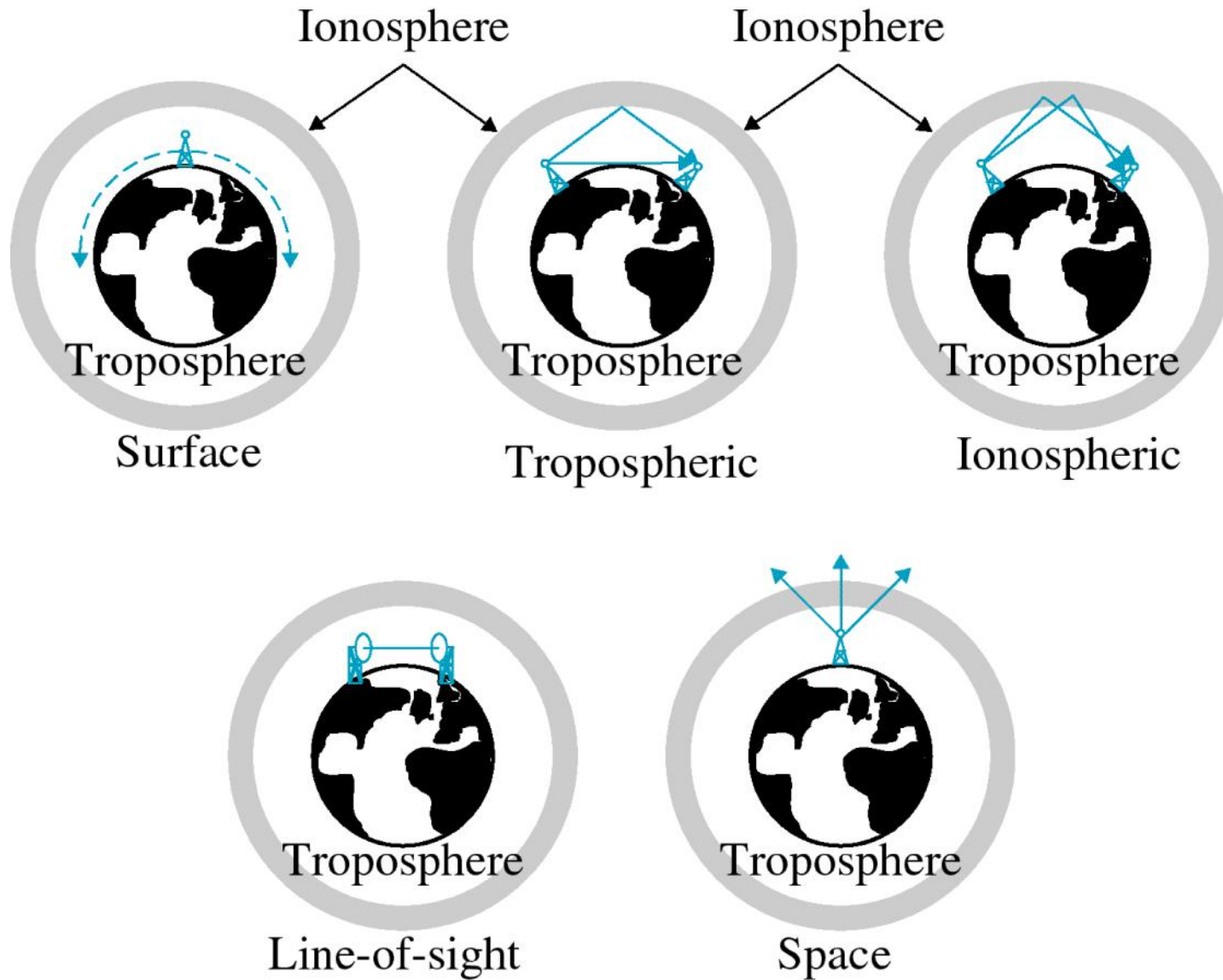


Figure 7-23, 24

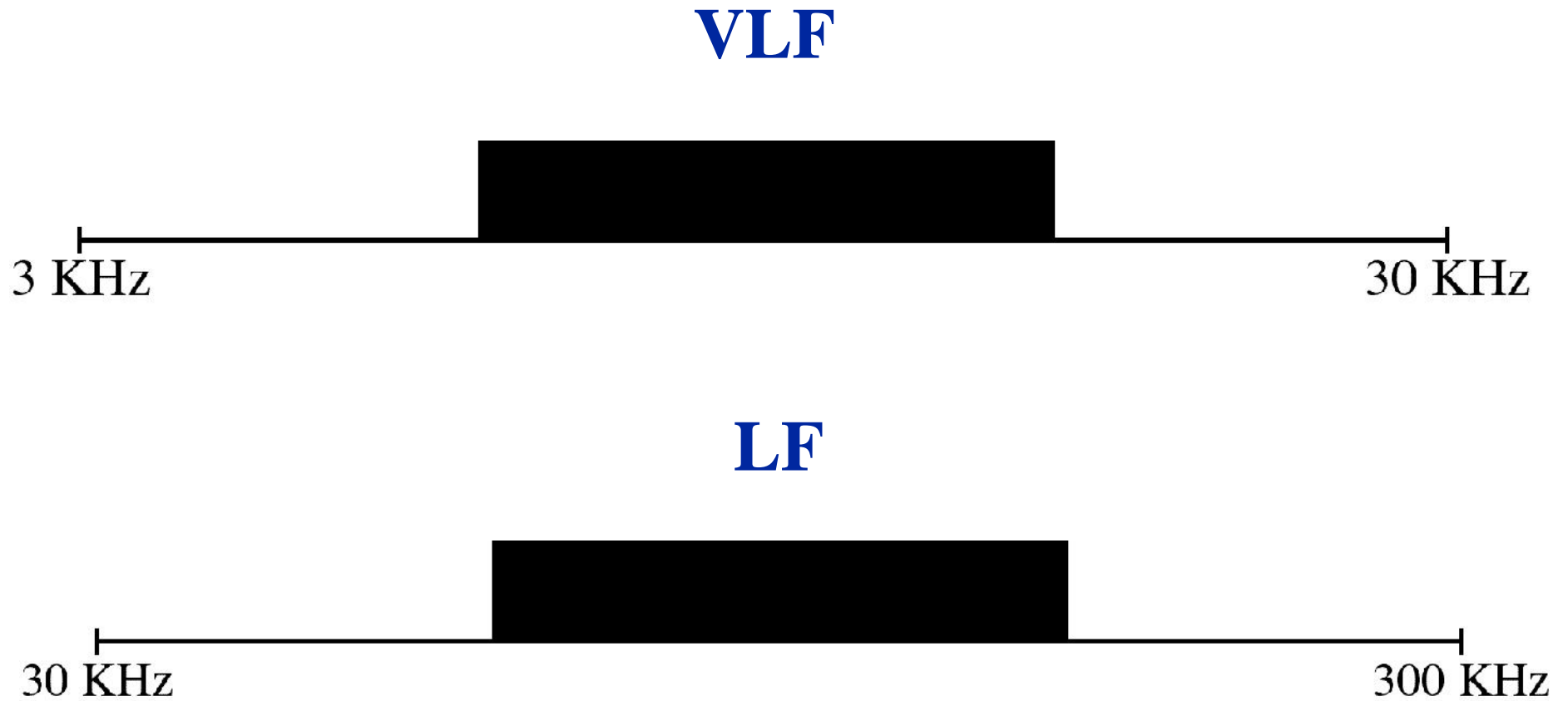


Figure 7-25, 26

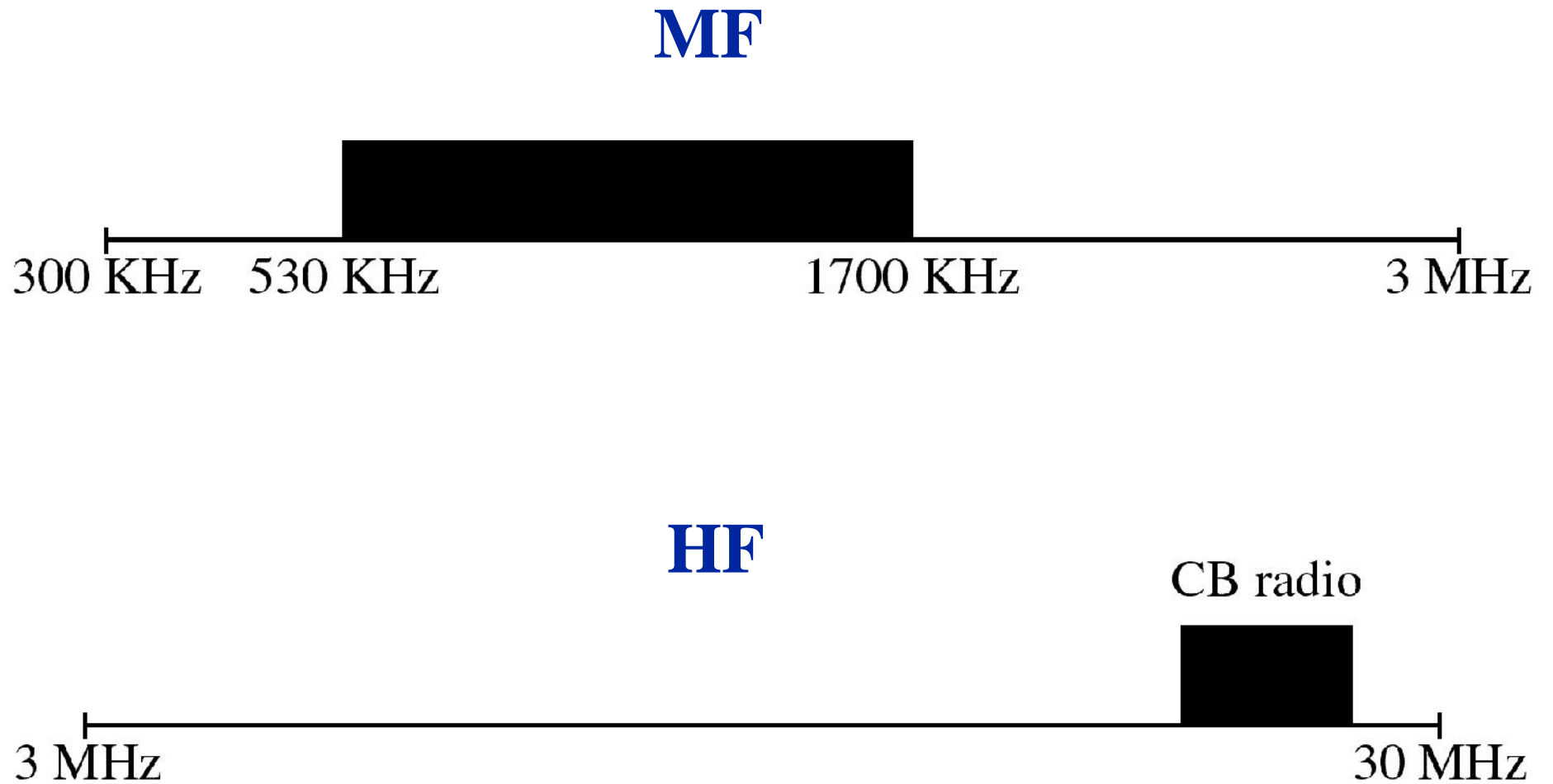
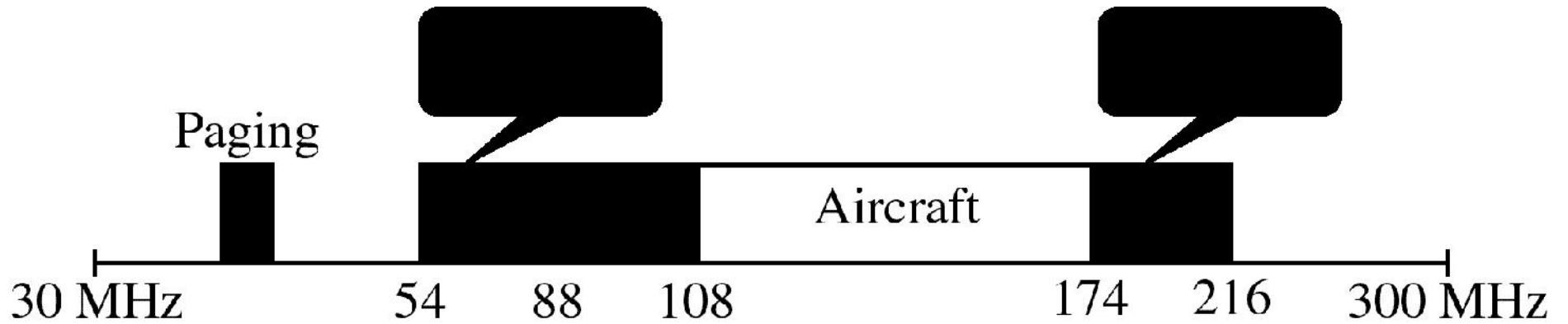


Figure 7-27, 28

VHF



UHF

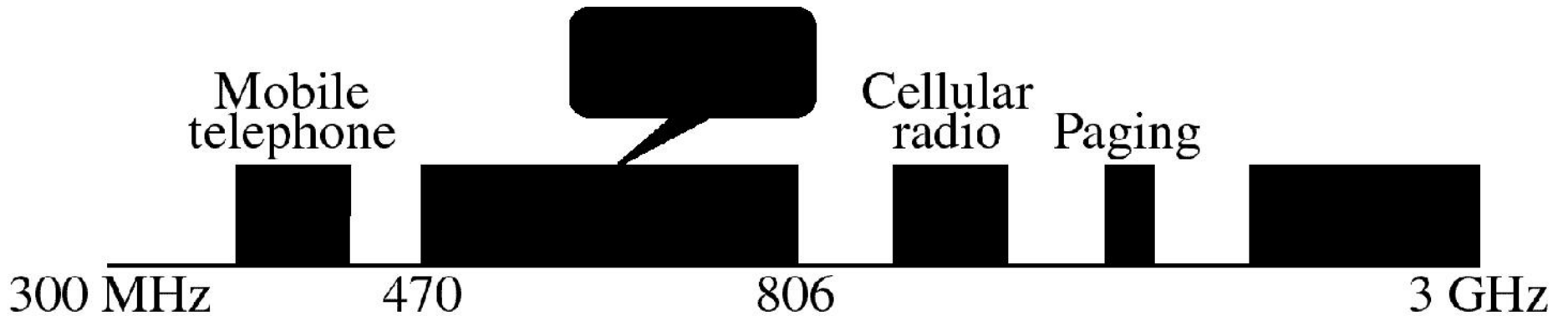


Figure 7-29, 30

SHF



EHF



Figure 7-31

Terrestrial Microwave

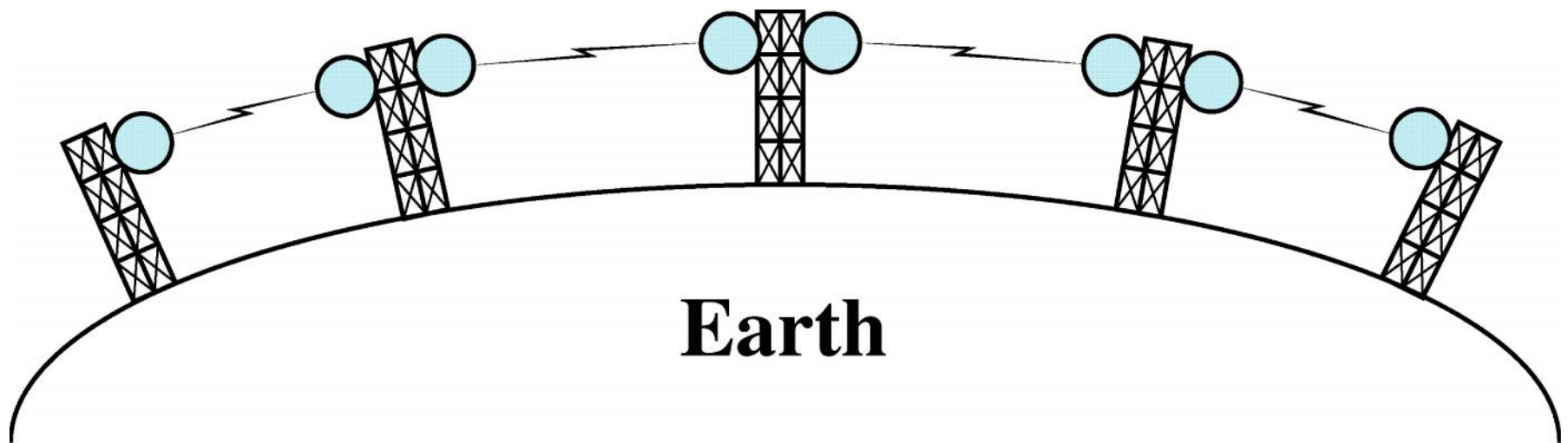


Figure 7-32

Parabolic Dish Antenna

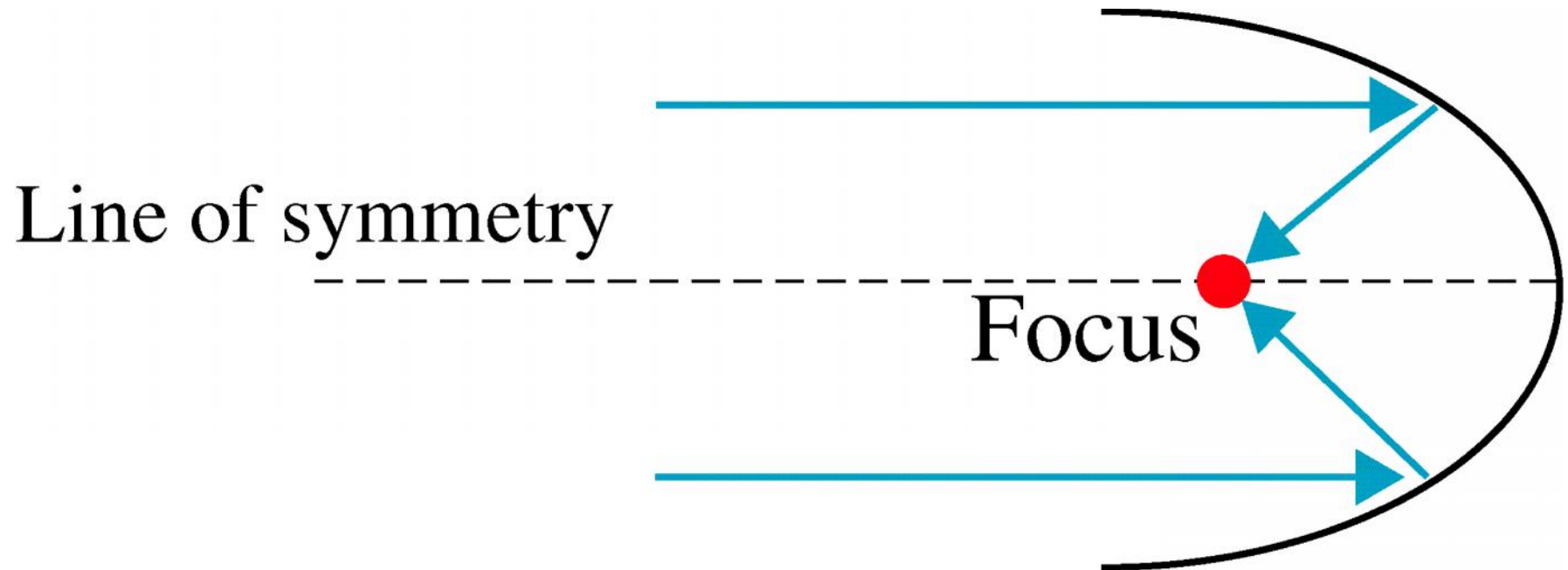


Figure 7-33

Horn Antenna

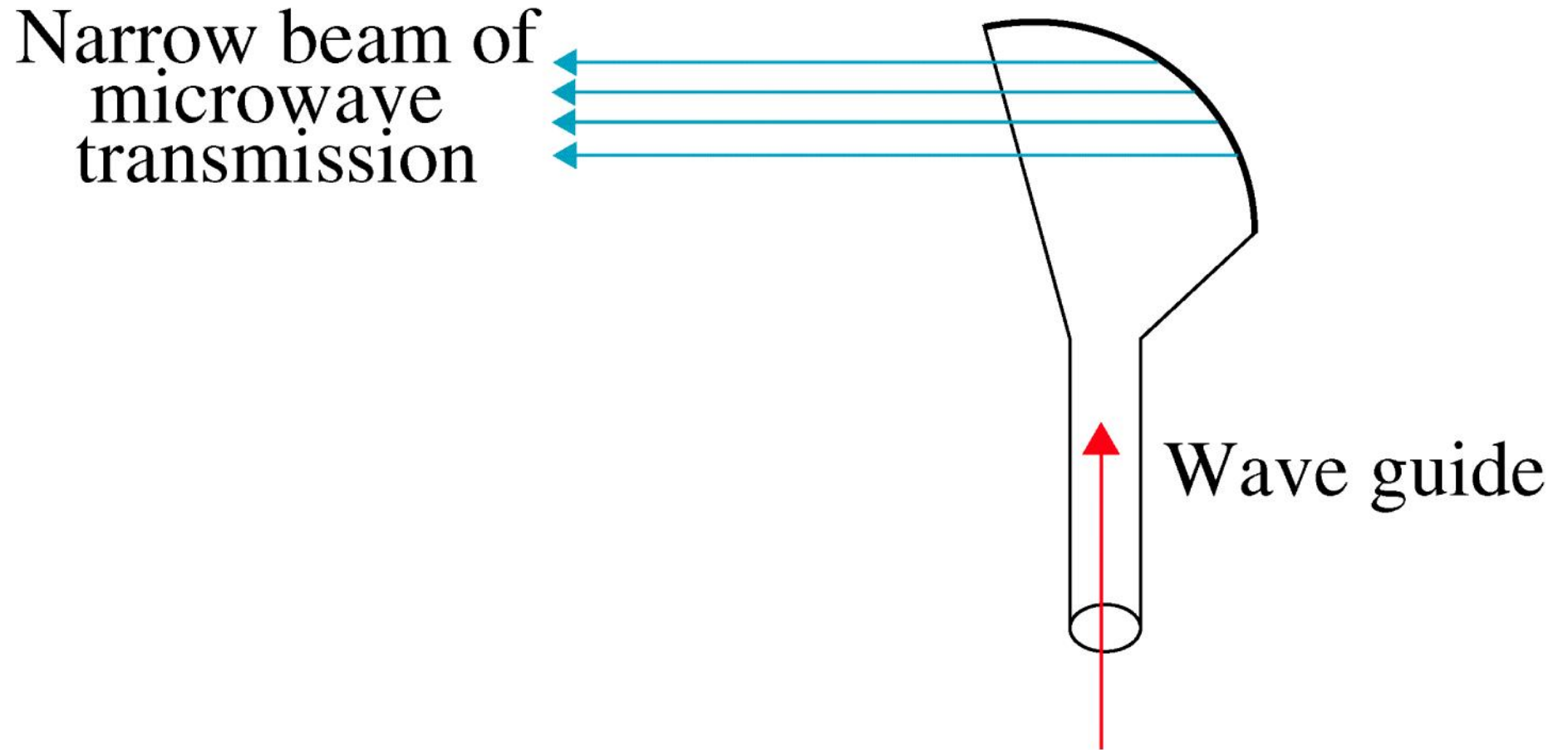


Figure 7-34

Satellite Communication

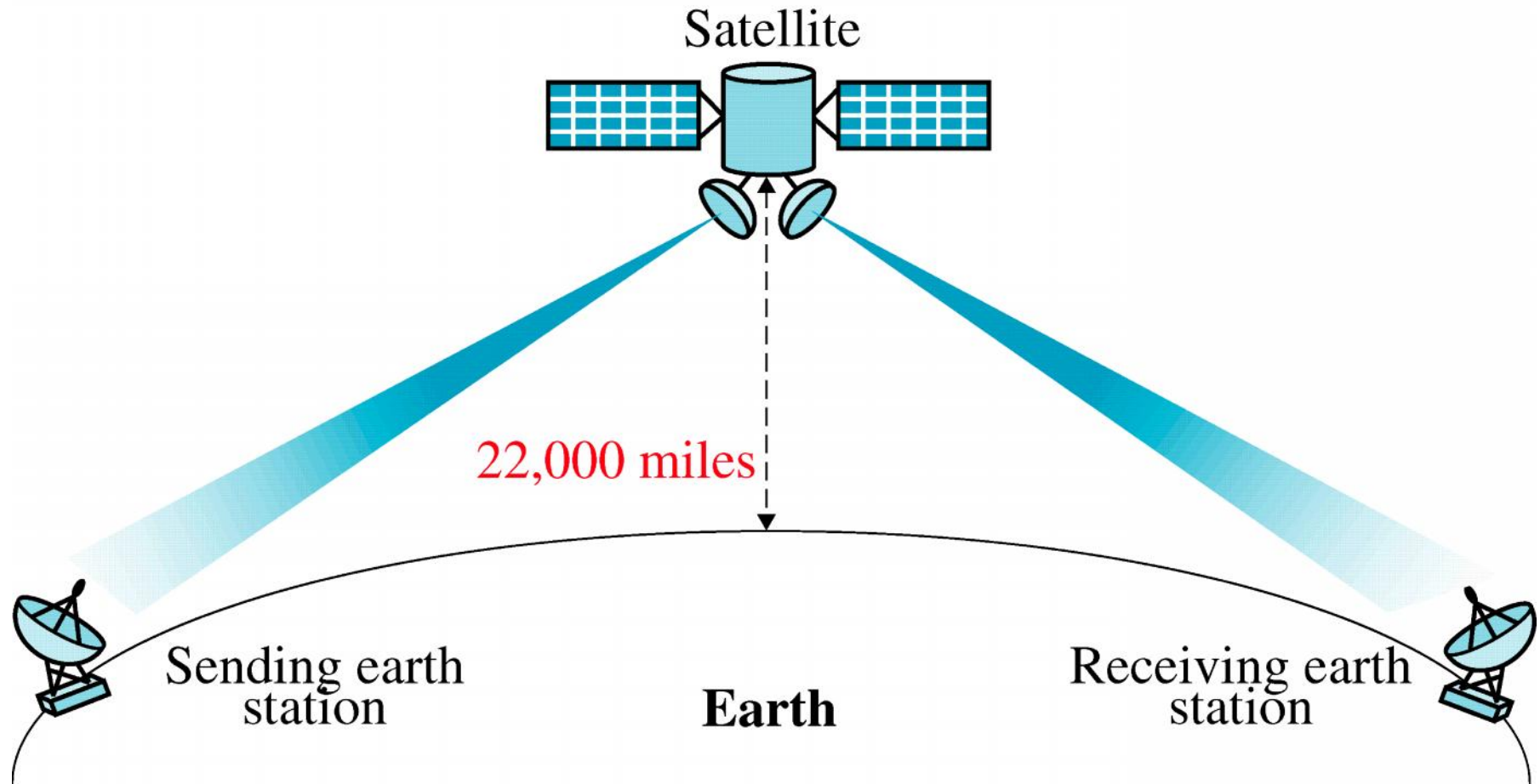


Figure 7-35

Geosynchronous Orbit

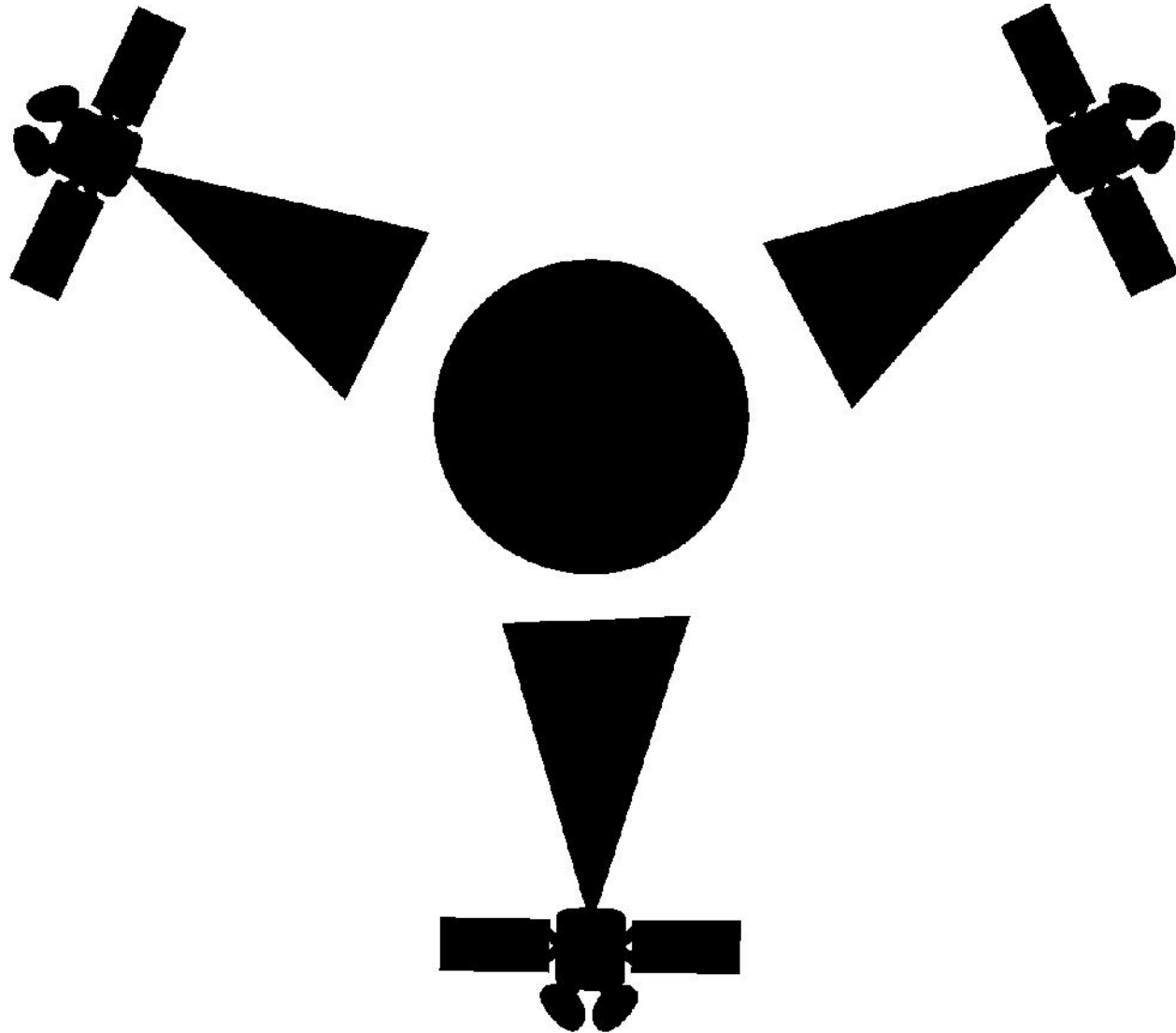


Figure 7-36

Cellular System

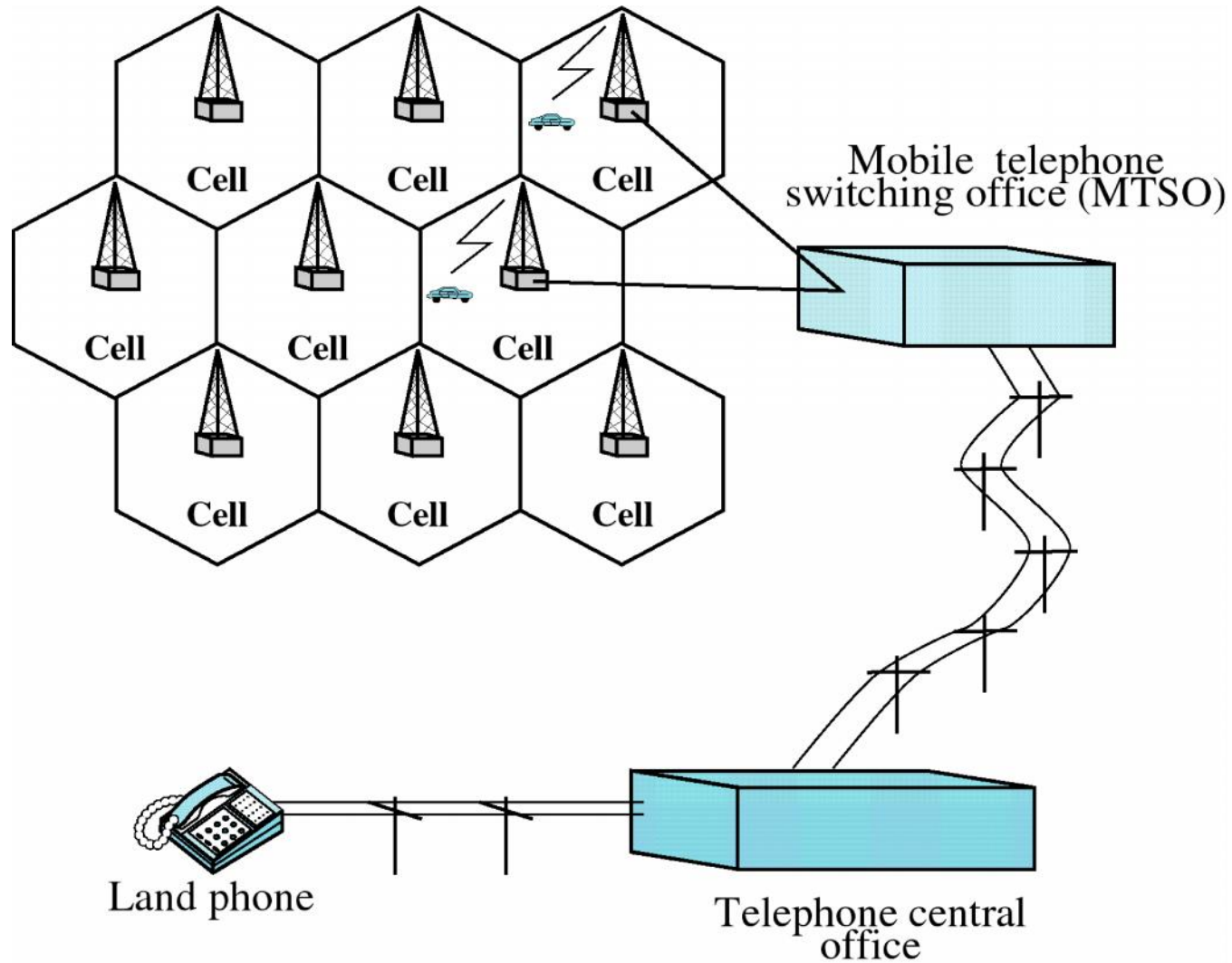
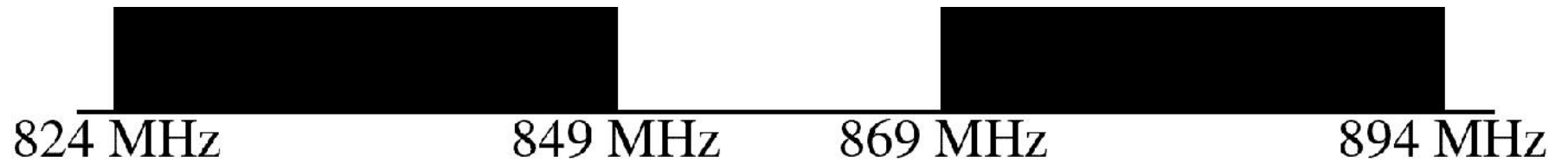
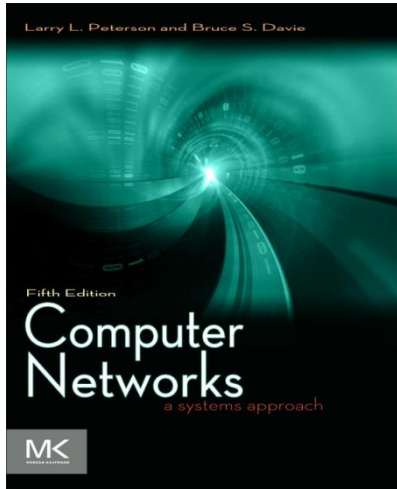


Figure 7-37

Cellular Bands





UNIT 2

MEDIA ACCESS & LOGICAL LINK CONTROL

Problems

- In Chapter 1 we saw networks consists of links interconnecting nodes. How to connect two nodes together?
- We also introduced the concept of “cloud” abstractions to represent a network without revealing its internal complexities. How to connect a host to a cloud?

Chapter Outline

- Perspectives on Connecting nodes
- Encoding
- Framing
- Error Detection
- Reliable Transmission
- Ethernet and Multiple Access Networks
- Wireless Networks
- Switching and Bridging

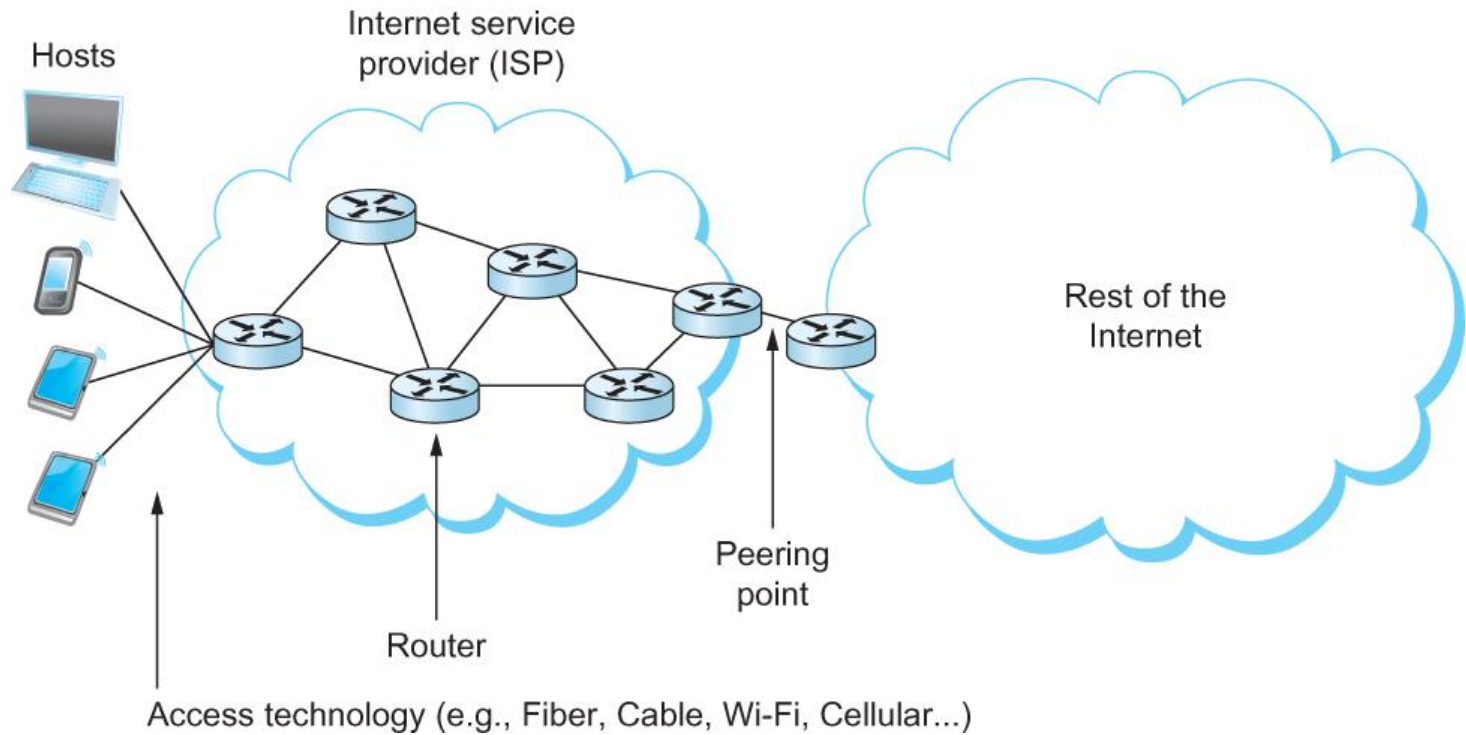
Chapter Goal

- Exploring different communication medium over which we can send data
- Understanding the issue of encoding bits onto transmission medium so that they can be understood by the receiving end
- Discussing the matter of delineating the sequence of bits transmitted over the link into complete messages that can be delivered to the end node
- Discussing different technique to detect transmission errors and take the appropriate action

Chapter Goal (contd.)

- Discussing the issue of making the links reliable in spite of transmission problems
- Introducing Media Access Control Problem
- Introducing Carrier Sense Multiple Access (CSMA) networks
- Introducing Wireless Networks with different available technologies and protocol

Perspectives on Connecting



An end-user's view of the Internet

Link Capacity and Shannon-Hartley Theorem

- Gives the upper bound to the capacity of a link in terms of bits per second (bps) as a function of signal-to-noise ratio of the link measured in decibels (dB).
- $C = B \log_2(1+S/N)$
 - Where $B = 3300 - 300 = 3000\text{Hz}$, S is the signal power, N the average noise.
 - The signal to noise ratio (S/N) is measured in decibels is related to $\text{dB} = 10 \times \log_{10}(S/N)$. If there is 30dB of noise then $S/N = 1000$.
 - Now $C = 3000 \times \log_2(1001) = 30\text{kbps}$.
 - How can we get 56kbps?

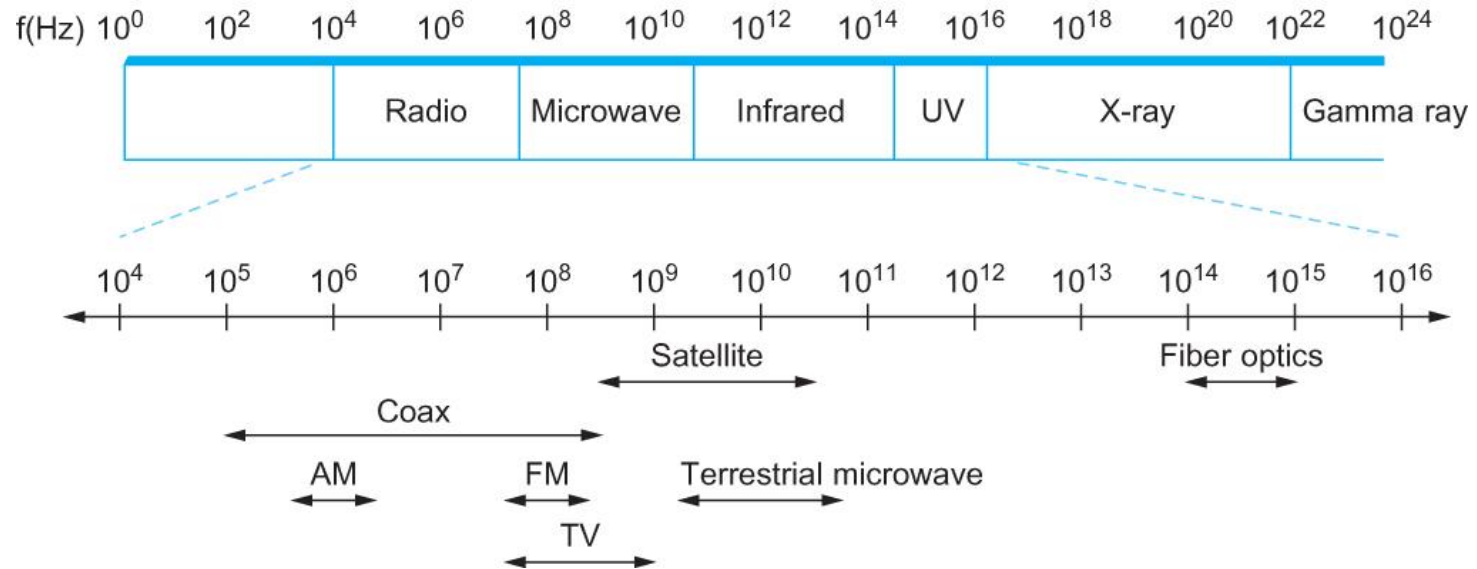
Links

- All practical links rely on some sort of electromagnetic radiation propagating through a medium or, in some cases, through free space
- One way to characterize links, then, is by the medium they use
 - Typically copper wire in some form (as in Digital Subscriber Line (DSL) and coaxial cable),
 - Optical fiber (as in both commercial fiber-to-the home services and many long-distance links in the Internet's backbone), or
 - Air/free space (for wireless links)

Links

- Another important link characteristic is the *frequency*
 - Measured in hertz, with which the electromagnetic waves oscillate
- Distance between the adjacent pair of maxima or minima of a wave measured in meters is called *wavelength*
 - Speed of light divided by frequency gives the wavelength.
 - Frequency on a copper cable range from 300Hz to 3300Hz; Wavelength for 300Hz wave through copper is speed of light on a copper / frequency
 - $2/3 \times 3 \times 10^8 / 300 = 667 \times 10^3$ meters.
- Placing binary data on a signal is called *encoding*.
- Modulation involves modifying the signals in terms of their frequency, amplitude, and phase.

Links



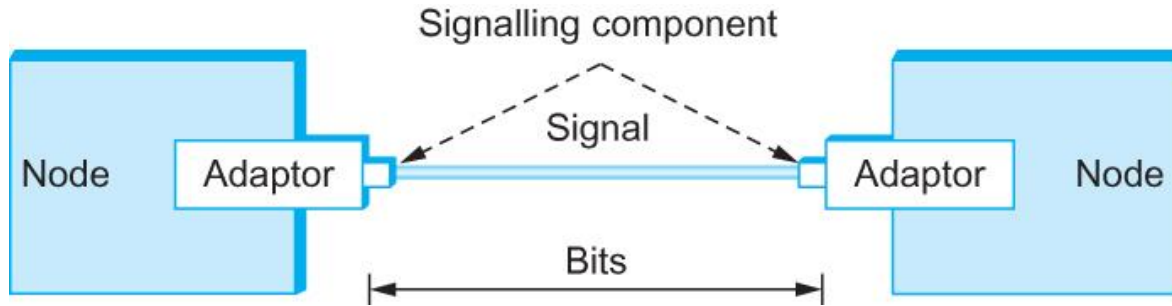
Electromagnetic spectrum

Links

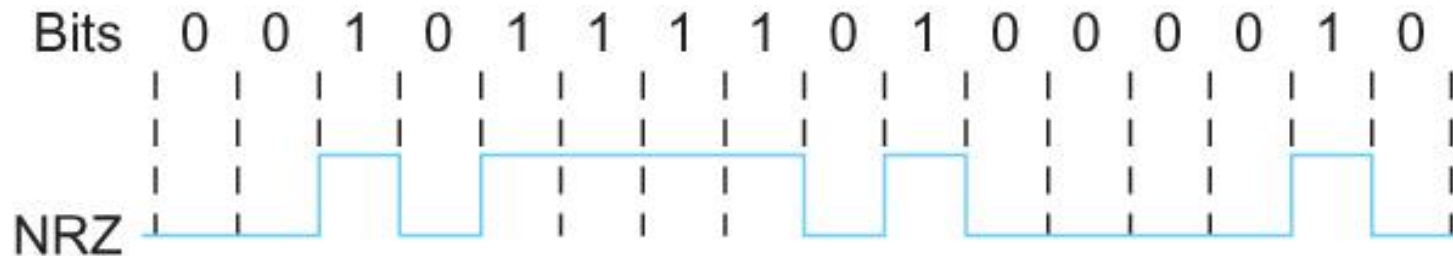
Service	Bandwidth (typical)
Dial-up	28–56 kbps
ISDN	64–128 kbps
DSL	128 kbps–100 Mbps
CATV (cable TV)	1–40 Mbps
FTTH (fibre to the home)	50 Mbps–1 Gbps

Common services available to connect your home

Encoding



Signals travel between signaling components; bits flow between adaptors



NRZ encoding of a bit stream

Encoding

- Problem with NRZ
 - Baseline wander
 - The receiver keeps an average of the signals it has seen so far
 - Uses the average to distinguish between low and high signal
 - When a signal is significantly low than the average, it is 0, else it is 1
 - Too many consecutive 0's and 1's cause this average to change, making it difficult to detect

Encoding

- Problem with NRZ
 - Clock recovery
 - Frequent transition from high to low or vice versa are necessary to enable clock recovery
 - Both the sending and decoding process is driven by a clock
 - Every clock cycle, the sender transmits a bit and the receiver recovers a bit
 - The sender and receiver have to be precisely synchronized

Encoding

- NRZI
 - Non Return to Zero Inverted
 - Sender makes a transition from the current signal to encode 1 and stay at the current signal to encode 0
 - Solves for consecutive 1's

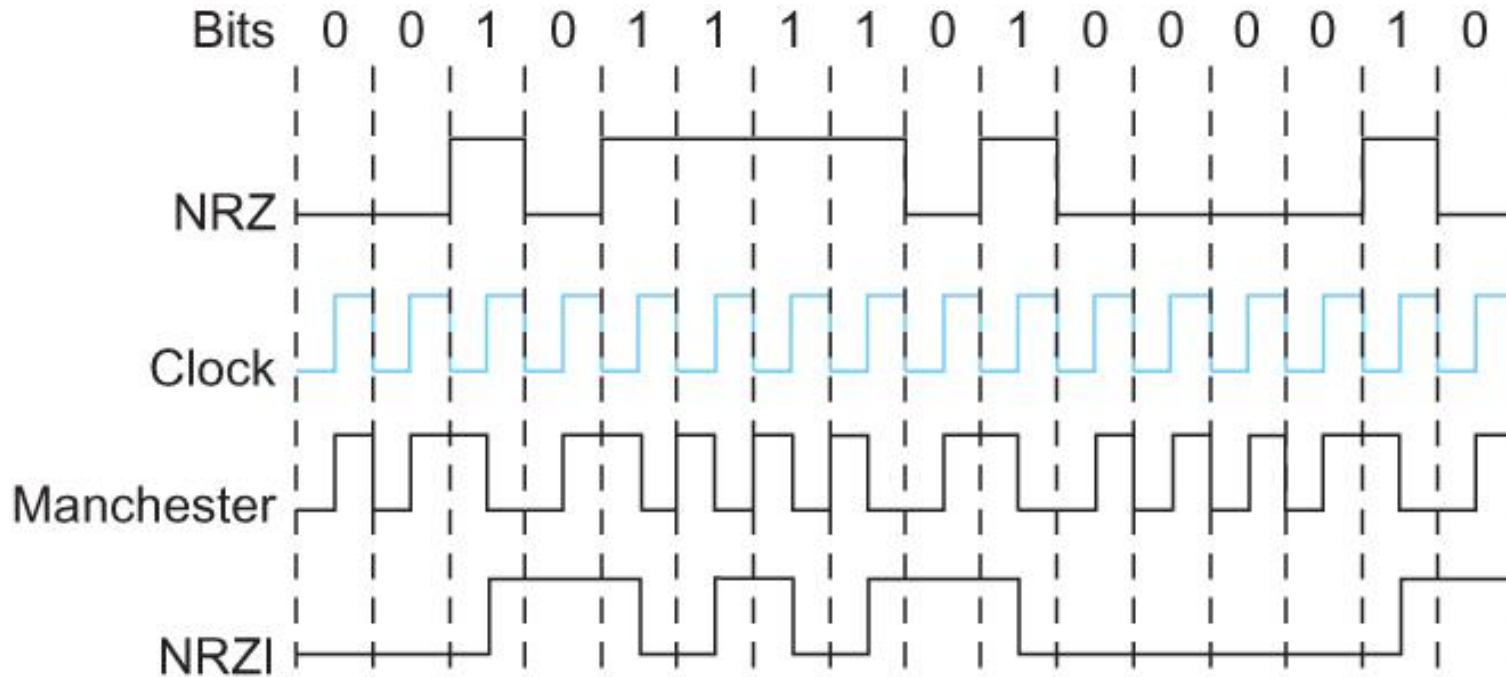
Encoding

- Manchester encoding
 - Merging the clock with signal by transmitting Ex-OR of the NRZ encoded data and the clock
 - Clock is an internal signal that alternates from low to high, a low/high pair is considered as one clock cycle
 - In Manchester encoding
 - 0: low → high transition
 - 1: high → low transition

Encoding

- Problem with Manchester encoding
 - Doubles the rate at which the signal transitions are made on the link
 - Which means the receiver has half of the time to detect each pulse of the signal
 - The rate at which the signal changes is called the link's baud rate
 - In Manchester the bit rate is half the baud rate

Encoding



Different encoding strategies

Encoding

■ 4B/5B encoding

- Insert extra bits into bit stream so as to break up the long sequence of 0's and 1's
- Every 4-bits of actual data are encoded in a 5-bit code that is transmitted to the receiver
- 5-bit codes are selected in such a way that each one has no more than one leading 0(zero) and no more than two trailing 0's.
- No pair of 5-bit codes results in more than three consecutive 0's

Encoding

■ 4B/5B encoding

0000 → 11110

0001 → 01001

0010 → 10100

..

..

1111 → 11101

16 left

11111 – when the line is idle

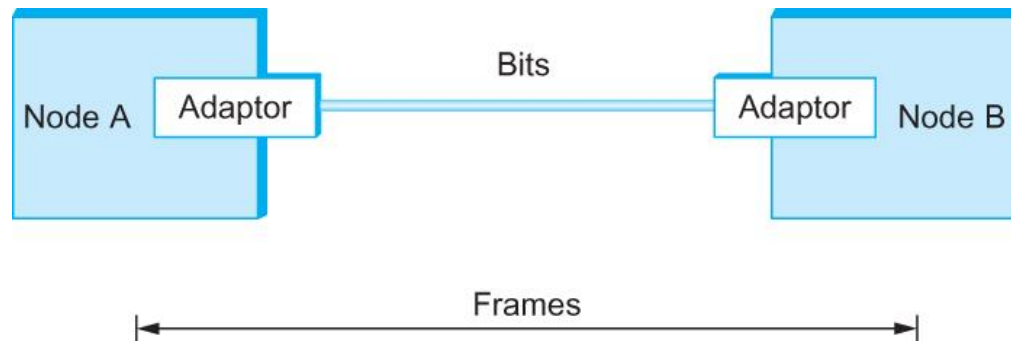
00000 – when the line is dead

00100 – to mean halt

13 left : 7 invalid, 6 for various
control signals

Framing

- We are focusing on packet-switched networks, which means that blocks of data (called *frames* at this level), not bit streams, are exchanged between nodes.
- It is the network adaptor that enables the nodes to exchange frames.



Bits flow between adaptors, frames between hosts

Framing

- When node A wishes to transmit a frame to node B, it tells its adaptor to transmit a frame from the node's memory. This results in a sequence of bits being sent over the link.
- The adaptor on node B then collects together the sequence of bits arriving on the link and deposits the corresponding frame in B's memory.
- Recognizing exactly what set of bits constitute a frame—that is, determining where the frame begins and ends—is the central challenge faced by the adaptor

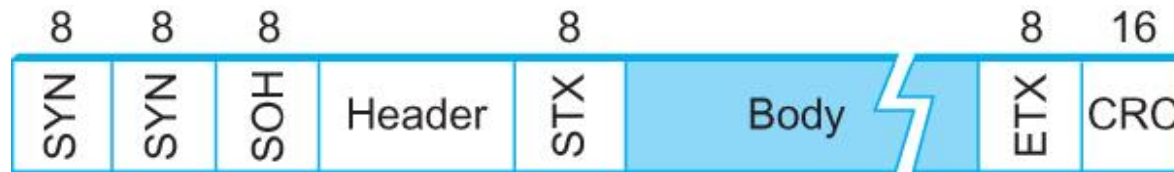
Framing

- Byte-oriented Protocols
 - To view each frame as a collection of bytes (characters) rather than bits
 - BISYNC (Binary Synchronous Communication) Protocol
 - Developed by IBM (late 1960)
 - DDCMP (Digital Data Communication Protocol)
 - Used in DECNet

Framing

- BISYNC – sentinel approach
 - Frames transmitted beginning with leftmost field
 - Beginning of a frame is denoted by sending a special SYN (synchronize) character
 - Data portion of the frame is contained between special sentinel character STX (start of text) and ETX (end of text)
 - SOH : Start of Header
 - DLE : Data Link Escape
 - CRC: Cyclic Redundancy Check

Framing

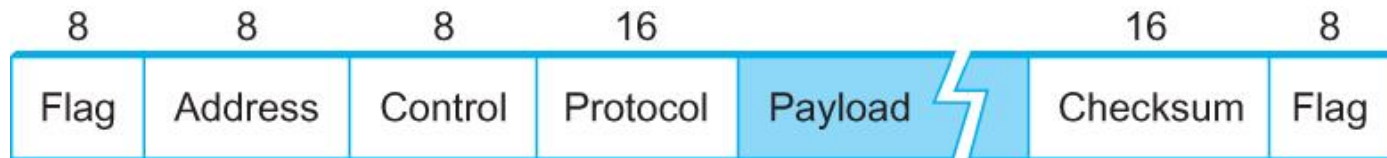


BISYNC Frame Format

Framing

- Recent PPP which is commonly run over Internet links uses sentinel approach
 - Special start of text character denoted as Flag
 - 0 1 1 1 1 1 0
 - Address, control : default numbers
 - Protocol for demux : IP / IPX
 - Payload : negotiated (1500 bytes)
 - Checksum : for error detection

Framing



PPP Frame Format

Framing

- Byte-counting approach
 - DDCMP
 - *count* : how many bytes are contained in the frame body
 - If *count* is corrupted
 - Framing error

Framing



DDCMP Frame Format

Framing

- Bit-oriented Protocol
 - HDLC : High Level Data Link Control
 - Beginning and Ending Sequences

0 1 1 1 1 1 0



HDLC Frame Format

Framing

- HDLC Protocol
 - On the sending side, any time five consecutive 1's have been transmitted from the body of the message (i.e. excluding when the sender is trying to send the distinguished 0111110 sequence)
 - The sender inserts 0 before transmitting the next bit

Framing

- HDLC Protocol

- On the receiving side

- 5 consecutive 1's

- Next bit 0 : Stuffed, so discard it

- 1 : Either End of the frame marker

- Or Error has been introduced in the bitstream

- Look at the next bit

- If 0 (01111110) → End of the frame marker

- If 1 (01111111) → Error, discard the whole frame

- The receiver needs to wait for next
01111110 before it can start
receiving again

Error Detection

- Bit errors are introduced into frames
 - Because of electrical interference and thermal noises
- Detecting Error
- Correction Error
- Two approaches when the recipient detects an error
 - Notify the sender that the message was corrupted, so the sender can send again.
 - If the error is rare, then the retransmitted message will be error-free
 - Using some error correct detection and correction algorithm, the receiver reconstructs the message

Error Detection

- Common technique for detecting transmission error
 - CRC (Cyclic Redundancy Check)
 - Used in HDLC, DDCMP, CSMA/CD, Token Ring
 - Other approaches
 - Two Dimensional Parity (BISYNC)
 - Checksum (IP)

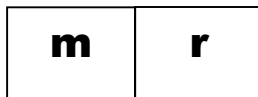
Error Detection

- Basic Idea of Error Detection
 - To add redundant information to a frame that can be used to determine if errors have been introduced
 - Imagine (Extreme Case)
 - Transmitting two complete copies of data
 - Identical → No error
 - Differ → Error
 - Poor Scheme ???
 - n bit message, n bit redundant information
 - Error can go undetected
 - In general, we can provide strong error detection technique
 - k redundant bits, n bits message, $k \ll n$
 - In Ethernet, a frame carrying up to 12,000 bits of data requires only 32-bit CRC

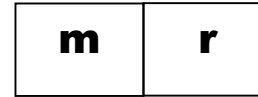
Error Detection

- Extra bits are redundant
 - They add no new information to the message
 - Derived from the original message using some algorithm
 - Both the sender and receiver know the algorithm

Sender



Receiver



Receiver computes r using m

If they match, no error

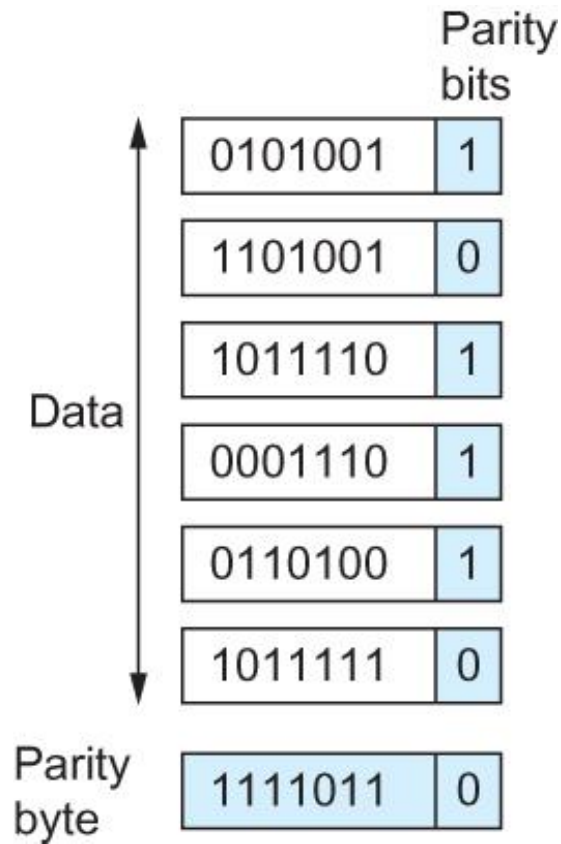
Two-dimensional parity

- Two-dimensional parity is exactly what the name suggests
- It is based on “simple” (one-dimensional) parity, which usually involves adding one extra bit to a 7-bit code to balance the number of 1s in the byte. For example,
 - Odd parity sets the eighth bit to 1 if needed to give an odd number of 1s in the byte, and
 - Even parity sets the eighth bit to 1 if needed to give an even number of 1s in the byte

Two-dimensional parity

- Two-dimensional parity does a similar calculation for each bit position across each of the bytes contained in the frame
- This results in an extra parity byte for the entire frame, in addition to a parity bit for each byte
- Two-dimensional parity catches all 1-, 2-, and 3-bit errors and most 4-bit errors

Two-dimensional parity



Two Dimensional Parity

Internet Checksum Algorithm

- Not used at the link level
- Add up all the words that are transmitted and then transmit the result of that sum
 - The result is called the checksum
- The receiver performs the same calculation on the received data and compares the result with the received checksum
- If any transmitted data, including the checksum itself, is corrupted, then the results will not match, so the receiver knows that an error occurred

Internet Checksum Algorithm

- Consider the data being checksummed as a sequence of 16-bit integers.
- Add them together using 16-bit ones complement arithmetic (explained next slide) and then take the ones complement of the result.
- That 16-bit number is the checksum

Internet Checksum Algorithm

- In ones complement arithmetic, a negative integer $-x$ is represented as the complement of x ;
 - Each bit of x is inverted.
- When adding numbers in ones complement arithmetic, a carryout from the most significant bit needs to be added to the result.

Internet Checksum Algorithm

- Consider, for example, the addition of -5 and -3 in ones complement arithmetic on 4-bit integers
 - $+5$ is 0101 , so -5 is 1010 ; $+3$ is 0011 , so -3 is 1100
- If we add 1010 and 1100 ignoring the carry, we get 0110
- In ones complement arithmetic, the fact that this operation caused a carry from the most significant bit causes us to increment the result, giving 0111 , which is the ones complement representation of -8 (obtained by inverting the bits in 1000), as we would expect

Cyclic Redundancy Check (CRC)

- Reduce the number of extra bits and maximize protection
- Given a bit string 110001 we can associate a polynomial on a single variable x for it.
 $1.x^5+1.x^4+0.x^3+0.x^2+0.x^1+1.x^0 = x^5+x^4+1$ and the degree is 5.
A k -bit frame has a maximum degree of $k-1$
- Let $M(x)$ be a message polynomial and $C(x)$ be a generator polynomial.

Cyclic Redundancy Check (CRC)

- Let $M(x)/C(x)$ leave a remainder of 0.
- When $M(x)$ is sent and $M'(x)$ is received we have $M'(x) = M(x) + E(x)$
- The receiver computes $M'(x)/C(x)$ and if the remainder is nonzero, then an error has occurred.
- The only thing the sender and the receiver should know is $C(x)$.

Cyclic Redundancy Check (CRC)

Polynomial Arithmetic Modulo 2

- Any polynomial $B(x)$ can be divided by a divisor polynomial $C(x)$ if $B(x)$ is of higher degree than $C(x)$.
- Any polynomial $B(x)$ can be divided once by a divisor polynomial $C(x)$ if $B(x)$ is of the same degree as $C(x)$.
- The remainder obtained when $B(x)$ is divided by $C(x)$ is obtained by subtracting $C(x)$ from $B(x)$.
- To subtract $C(x)$ from $B(x)$, we simply perform the exclusive-OR (XOR) operation on each pair of matching coefficients.

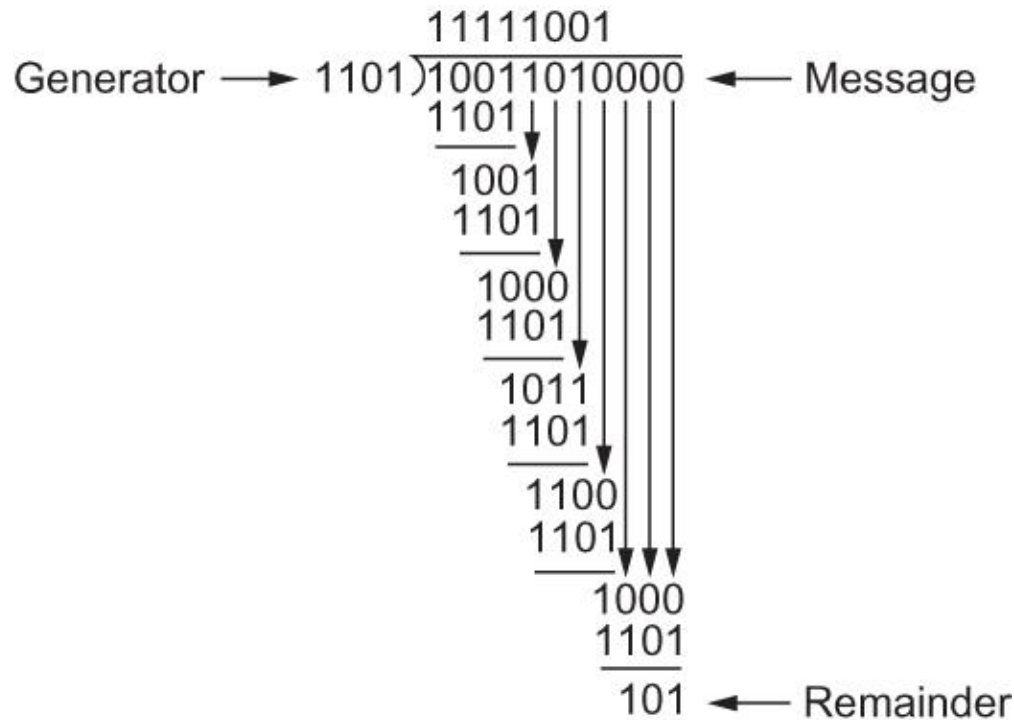
Cyclic Redundancy Check (CRC)

- Let $M(x)$ be a frame with m bits and let the generator polynomial have less than m bits say equal to r .
- Let r be the degree of $C(x)$. Append r zero bits to the low-order end of the frame, so it now contains $m+r$ bits and corresponds to the polynomial $x^rM(x)$.

Cyclic Redundancy Check (CRC)

- Divide the bit string corresponding to $x^r M(x)$ by the bit string corresponding to $C(x)$ using modulo 2 division.
- Subtract the remainder (which is always r or fewer bits) from the string corresponding to $x^r M(x)$ using modulo 2 subtraction (addition and subtraction are the same in modulo 2).
- The result is the checksummed frame to be transmitted. Call it polynomial $M'(x)$.

Cyclic Redundancy Check (CRC)



CRC Calculation using Polynomial Long Division

Cyclic Redundancy Check (CRC)

- Properties of Generator Polynomial
 - Let $P(x)$ represent what the sender sent and $P(x) + E(x)$ is the received string. A 1 in $E(x)$ represents that in the corresponding position in $P(x)$ the message the bit is flipped.
 - We know that $P(x)/C(x)$ leaves a remainder of 0, but if $E(x)/C(x)$ leaves a remainder of 0, then either $E(x) = 0$ or $C(x)$ is factor of $E(x)$.
 - When $C(x)$ is a factor of $E(x)$ we have problem; errors go unnoticed.
 - If there is a single bit error then $E(x) = x^i$, where i determines the bit in error. If $C(x)$ contains two or more terms it will never divide $E(x)$, so all single bit errors will be detected.

Cyclic Redundancy Check (CRC)

- Properties of Generator Polynomial
 - In general, it is possible to prove that the following types of errors can be detected by a $C(x)$ with the stated properties
 - All single-bit errors, as long as the x^k and x^0 terms have nonzero coefficients.
 - All double-bit errors, as long as $C(x)$ has a factor with at least three terms.
 - Any odd number of errors, as long as $C(x)$ contains the factor $(x+1)$.
 - Any “burst” error (i.e., sequence of consecutive error bits) for which the length of the burst is less than k bits. (Most burst errors of larger than k bits can also be detected.)

Cyclic Redundancy Check (CRC)

- Six generator polynomials that have become international standards are:
 - CRC-8 = x^8+x^2+x+1
 - CRC-10 = $x^{10}+x^9+x^5+x^4+x+1$
 - CRC-12 = $x^{12}+x^{11}+x^3+x^2+x+1$
 - CRC-16 = $x^{16}+x^{15}+x^2+1$
 - CRC-CCITT = $x^{16}+x^{12}+x^5+1$
 - CRC-32 =
 $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$

Reliable Transmission

- CRC is used to detect errors.
- Some error codes are strong enough to correct errors.
- The overhead is typically too high.
- Corrupt frames must be discarded.
- A link-level protocol that wants to deliver frames reliably must recover from these discarded frames.
- This is accomplished using a combination of two fundamental mechanisms
 - Acknowledgements and Timeouts

Reliable Transmission

- An *acknowledgement* (ACK for short) is a small control frame that a protocol sends back to its peer saying that it has received the earlier frame.
 - A control frame is a frame with header only (no data).
- The receipt of an *acknowledgement* indicates to the sender of the original frame that its frame was successfully delivered.

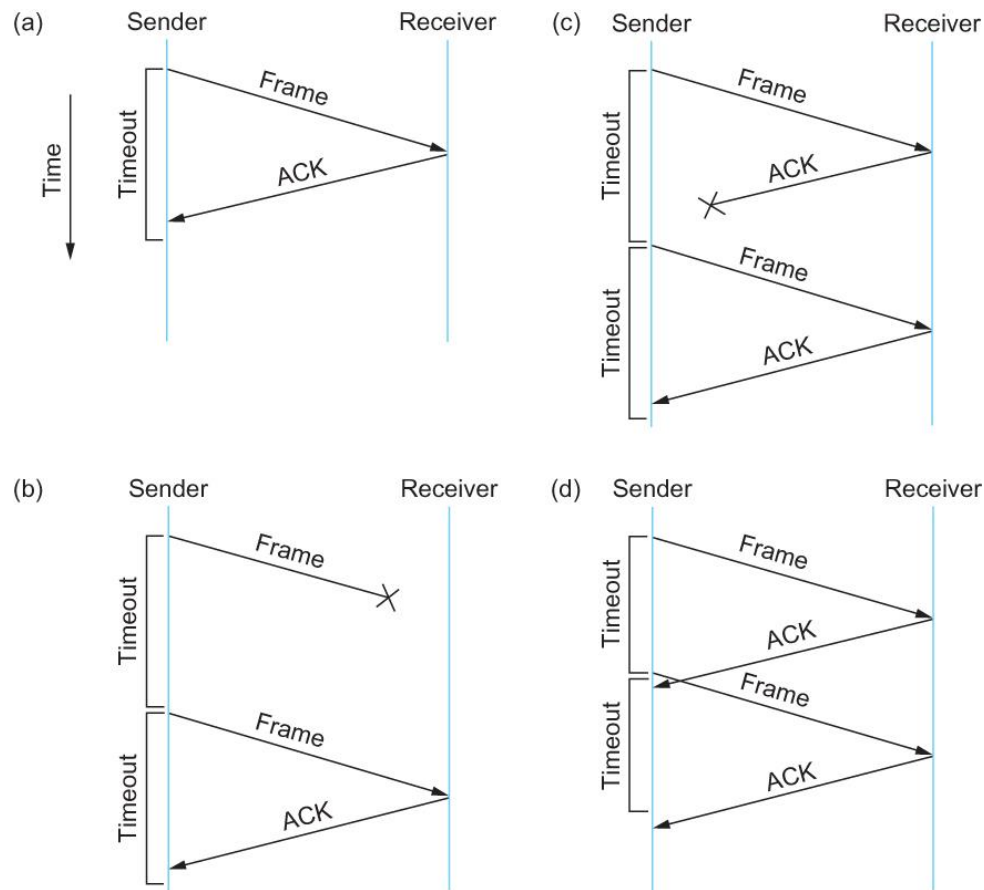
Reliable Transmission

- If the sender does not receive an *acknowledgment* after a reasonable amount of time, then it retransmits the original frame.
- The action of waiting a reasonable amount of time is called a *timeout*.
- The general strategy of using *acknowledgements* and *timeouts* to implement reliable delivery is sometimes called Automatic Repeat reQuest (ARQ).

Stop and Wait Protocol

- Idea of stop-and-wait protocol is straightforward
 - After transmitting one frame, the sender waits for an acknowledgement before transmitting the next frame.
 - If the acknowledgement does not arrive after a certain period of time, the sender times out and retransmits the original frame

Stop and Wait Protocol



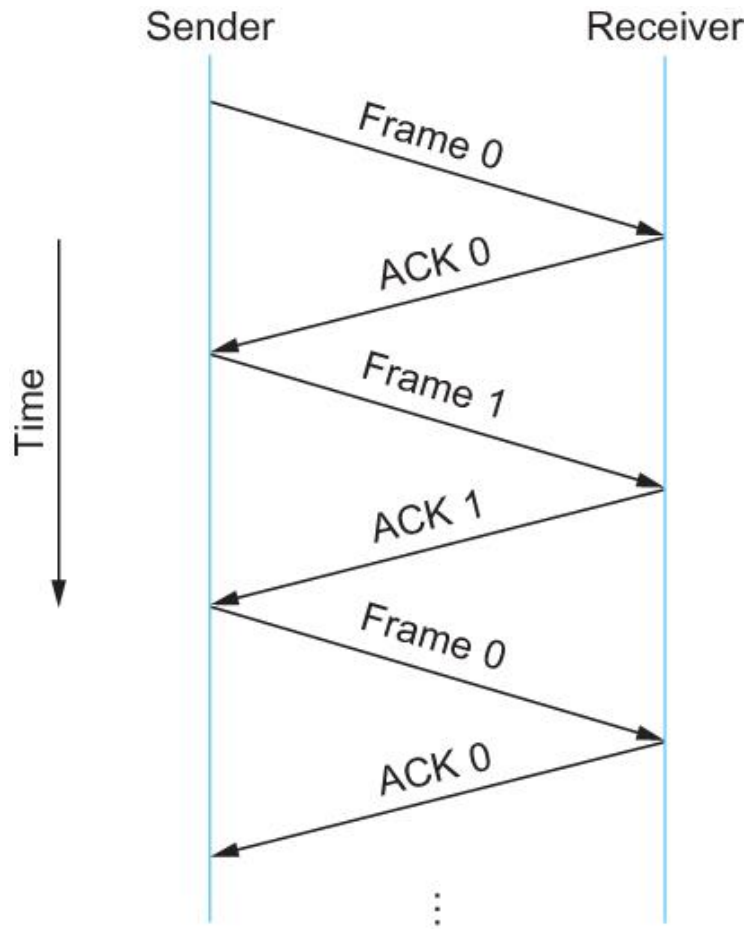
Timeline showing four different scenarios for the stop-and-wait algorithm.

(a) The ACK is received before the timer expires; (b) the original frame is lost; (c) the ACK is lost; (d) the timeout fires too soon

Stop and Wait Protocol

- If the acknowledgment is lost or delayed in arriving
 - The sender times out and retransmits the original frame, but the receiver will think that it is the next frame since it has correctly received and acknowledged the first frame
 - As a result, duplicate copies of frames will be delivered
- How to solve this
 - Use 1 bit sequence number (0 or 1)
 - When the sender retransmits frame 0, the receiver can determine that it is seeing a second copy of frame 0 rather than the first copy of frame 1 and therefore can ignore it (the receiver still acknowledges it, in case the first acknowledgement was lost)

Stop and Wait Protocol

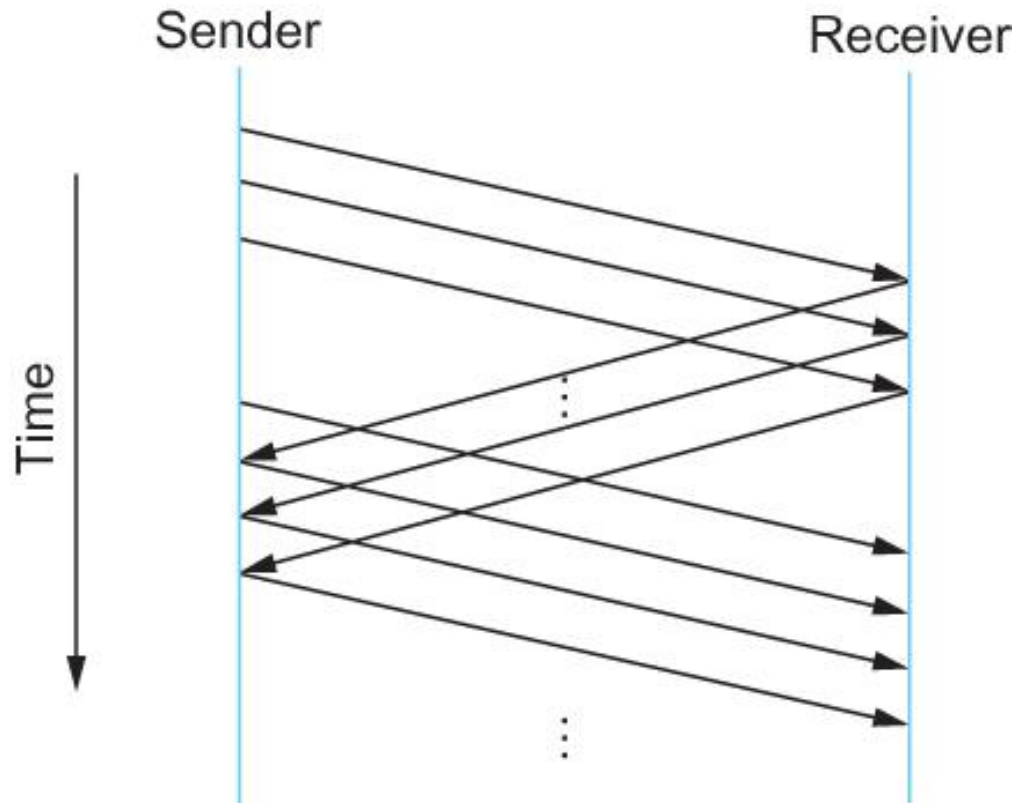


Timeline for stop-and-wait with 1-bit sequence number

Stop and Wait Protocol

- The sender has only one outstanding frame on the link at a time
 - This may be far below the link's capacity
- Consider a 1.5 Mbps link with a 45 ms RTT
 - The link has a delay \times bandwidth product of 67.5 Kb or approximately 8 KB
 - Since the sender can send only one frame per RTT and assuming a frame size of 1 KB
 - Maximum Sending rate
 - $\text{Bits per frame} \div \text{Time per frame} = 1024 \times 8 \div 0.045 = 182 \text{ Kbps}$
Or about one-eighth of the link's capacity
 - To use the link fully, then sender should transmit up to eight frames before having to wait for an acknowledgement

Sliding Window Protocol



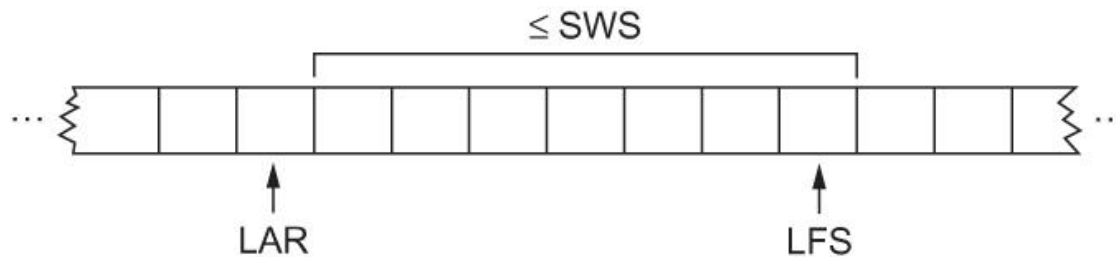
Timeline for Sliding Window Protocol

Sliding Window Protocol

- Sender assigns a sequence number denoted as SeqNum to each frame.
 - Assume it can grow infinitely large
- Sender maintains three variables
 - Sending Window Size (SWS)
 - Upper bound on the number of outstanding (unacknowledged) frames that the sender can transmit
 - Last Acknowledgement Received (LAR)
 - Sequence number of the last acknowledgement received
 - Last Frame Sent (LFS)
 - Sequence number of the last frame sent

Sliding Window Protocol

- Sender also maintains the following invariant
 $LFS - LAR \leq SWS$



Sliding Window on Sender

Sliding Window Protocol

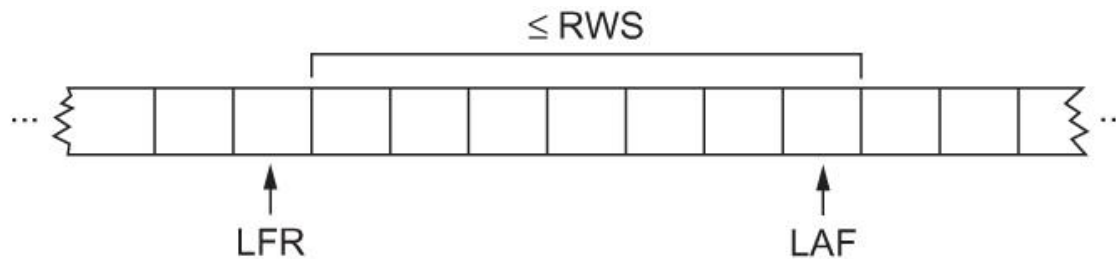
- When an acknowledgement arrives
 - the sender moves LAR to right, thereby allowing the sender to transmit another frame
- Also the sender associates a timer with each frame it transmits
 - It retransmits the frame if the timer expires before the ACK is received
- Note that the sender has to be willing to buffer up to SWS frames
 - WHY?

Sliding Window Protocol

- Receiver maintains three variables
 - Receiving Window Size (RWS)
 - Upper bound on the number of out-of-order frames that the receiver is willing to accept
 - Largest Acceptable Frame (LAF)
 - Sequence number of the largest acceptable frame
 - Last Frame Received (LFR)
 - Sequence number of the last frame received

Sliding Window Protocol

- Receiver also maintains the following invariant
 $LAF - LFR \leq RWS$



Sliding Window on Receiver

Sliding Window Protocol

- When a frame with sequence number SeqNum arrives, what does the receiver do?
 - If $\text{SeqNum} \leq \text{LFR}$ or $\text{SeqNum} > \text{LAF}$
 - Discard it (the frame is outside the receiver window)
 - If $\text{LFR} < \text{SeqNum} \leq \text{LAF}$
 - Accept it
 - Now the receiver needs to decide whether or not to send an ACK

Sliding Window Protocol

- Let SeqNumToAck
 - Denote the largest sequence number not yet acknowledged, such that all frames with sequence number less than or equal to SeqNumToAck have been received
- The receiver acknowledges the receipt of SeqNumToAck even if high-numbered packets have been received
 - This acknowledgement is said to be cumulative.
- The receiver then sets
 - $\text{LFR} = \text{SeqNumToAck}$ and adjusts
 - $\text{LAF} = \text{LFR} + \text{RWS}$

Sliding Window Protocol

For example, suppose $LFR = 5$ and $RWS = 4$
(i.e. the last ACK that the receiver sent was for seq. no. 5)
 $\Rightarrow LAF = 9$

If frames 7 and 8 arrive, they will be buffered because they are within the receiver window

But no ACK will be sent since frame 6 is yet to arrive

Frames 7 and 8 are out of order

Frame 6 arrives (it is late because it was lost first time and had to be retransmitted)

Now Receiver Acknowledges Frame 8
and bumps LFR to 8
and LAF to 12

Issues with Sliding Window Protocol

- When timeout occurs, the amount of data in transit decreases
 - Since the sender is unable to advance its window
- When the packet loss occurs, this scheme is no longer keeping the pipe full
 - The longer it takes to notice that a packet loss has occurred, the more severe the problem becomes
- How to improve this
 - Negative Acknowledgement (NAK)
 - Additional Acknowledgement
 - Selective Acknowledgement

Issues with Sliding Window Protocol

- Negative Acknowledgement (NAK)
 - Receiver sends NAK for frame 6 when frame 7 arrive (in the previous example)
 - However this is unnecessary since sender's timeout mechanism will be sufficient to catch the situation
- Additional Acknowledgement
 - Receiver sends additional ACK for frame 5 when frame 7 arrives
 - Sender uses duplicate ACK as a clue for frame loss
- Selective Acknowledgement
 - Receiver will acknowledge exactly those frames it has received, rather than the highest number frames
 - Receiver will acknowledge frames 7 and 8
 - Sender knows frame 6 is lost
 - Sender can keep the pipe full (additional complexity)

Issues with Sliding Window Protocol

How to select the window size

- SWS is easy to compute
 - Delay \times Bandwidth
- RWS can be anything
 - Two common setting
 - $RWS = 1$

No buffer at the receiver for frames that arrive out of order

- $RWS = SWS$

The receiver can buffer frames that the sender transmits

It does not make any sense to keep $RWS > SWS$

WHY?

Issues with Sliding Window Protocol

- Finite Sequence Number
 - Frame sequence number is specified in the header field
 - Finite size
 - 3 bit: eight possible sequence number: 0, 1, 2, 3, 4, 5, 6, 7
 - It is necessary to wrap around

Issues with Sliding Window Protocol

- How to distinguish between different incarnations of the same sequence number?
 - Number of possible sequence number must be larger than the number of outstanding frames allowed
 - Stop and Wait: One outstanding frame
 - 2 distinct sequence number (0 and 1)
 - Let `MaxSeqNum` be the number of available sequence numbers
 - $SWS + 1 \leq \text{MaxSeqNum}$
 - Is this sufficient?

Issues with Sliding Window Protocol

$SWS + 1 \leq \text{MaxSeqNum}$

- Is this sufficient?
 - Depends on RWS
 - If $RWS = 1$, then sufficient
 - If $RWS = SWS$, then not good enough
- For example, we have eight sequence numbers

0, 1, 2, 3, 4, 5, 6, 7

$RWS = SWS = 7$

Sender sends 0, 1, ..., 6

Receiver receives 0, 1, ..., 6

Receiver acknowledges 0, 1, ..., 6

ACK (0, 1, ..., 6) are lost

Sender retransmits 0, 1, ..., 6

Receiver is expecting 7, 0, ..., 5

Issues with Sliding Window Protocol

To avoid this,

If $RWS = SWS$

$SWS < (MaxSeqNum + 1)/2$

Issues with Sliding Window Protocol

- Serves three different roles
 - Reliable
 - Preserve the order
 - Each frame has a sequence number
 - The receiver makes sure that it does not pass a frame up to the next higher-level protocol until it has already passed up all frames with a smaller sequence number
 - Frame control
 - Receiver is able to throttle the sender
 - Keeps the sender from overrunning the receiver
 - From transmitting more data than the receiver is able to process

Ethernet

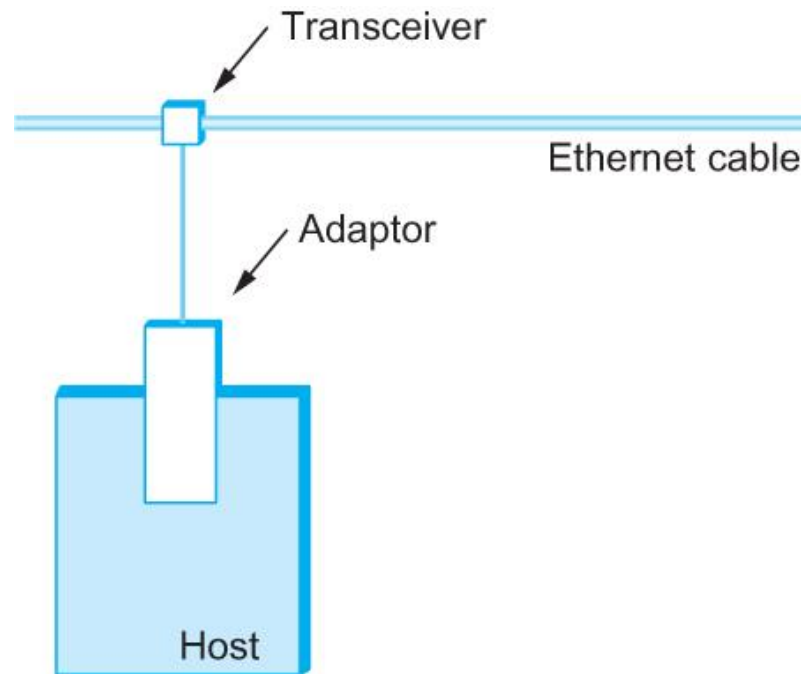
- Most successful local area networking technology of last 20 years.
- Developed in the mid-1970s by researchers at the Xerox Palo Alto Research Centers (PARC).
- Uses CSMA/CD technology
 - Carrier Sense Multiple Access with Collision Detection.
 - A set of nodes send and receive frames over a shared link.
 - Carrier sense means that all nodes can distinguish between an idle and a busy link.
 - Collision detection means that a node listens as it transmits and can therefore detect when a frame it is transmitting has collided with a frame transmitted by another node.

Ethernet

- Uses ALOHA (packet radio network) as the root protocol
 - Developed at the University of Hawaii to support communication across the Hawaiian Islands.
 - For ALOHA the medium was atmosphere, for Ethernet the medium is a coax cable.
- DEC and Intel joined Xerox to define a 10-Mbps Ethernet standard in 1978.
- This standard formed the basis for IEEE standard 802.3
- More recently 802.3 has been extended to include a 100-Mbps version called Fast Ethernet and a 1000-Mbps version called Gigabit Ethernet.

Ethernet

- An Ethernet segment is implemented on a coaxial cable of up to 500 m.
 - This cable is similar to the type used for cable TV except that it typically has an impedance of 50 ohms instead of cable TV's 75 ohms.
- Hosts connect to an Ethernet segment by tapping into it.
- A transceiver (a small device directly attached to the tap) detects when the line is idle and drives signal when the host is transmitting.
- The transceiver also receives incoming signal.
- The transceiver is connected to an Ethernet adaptor which is plugged into the host.
- The protocol is implemented on the adaptor.

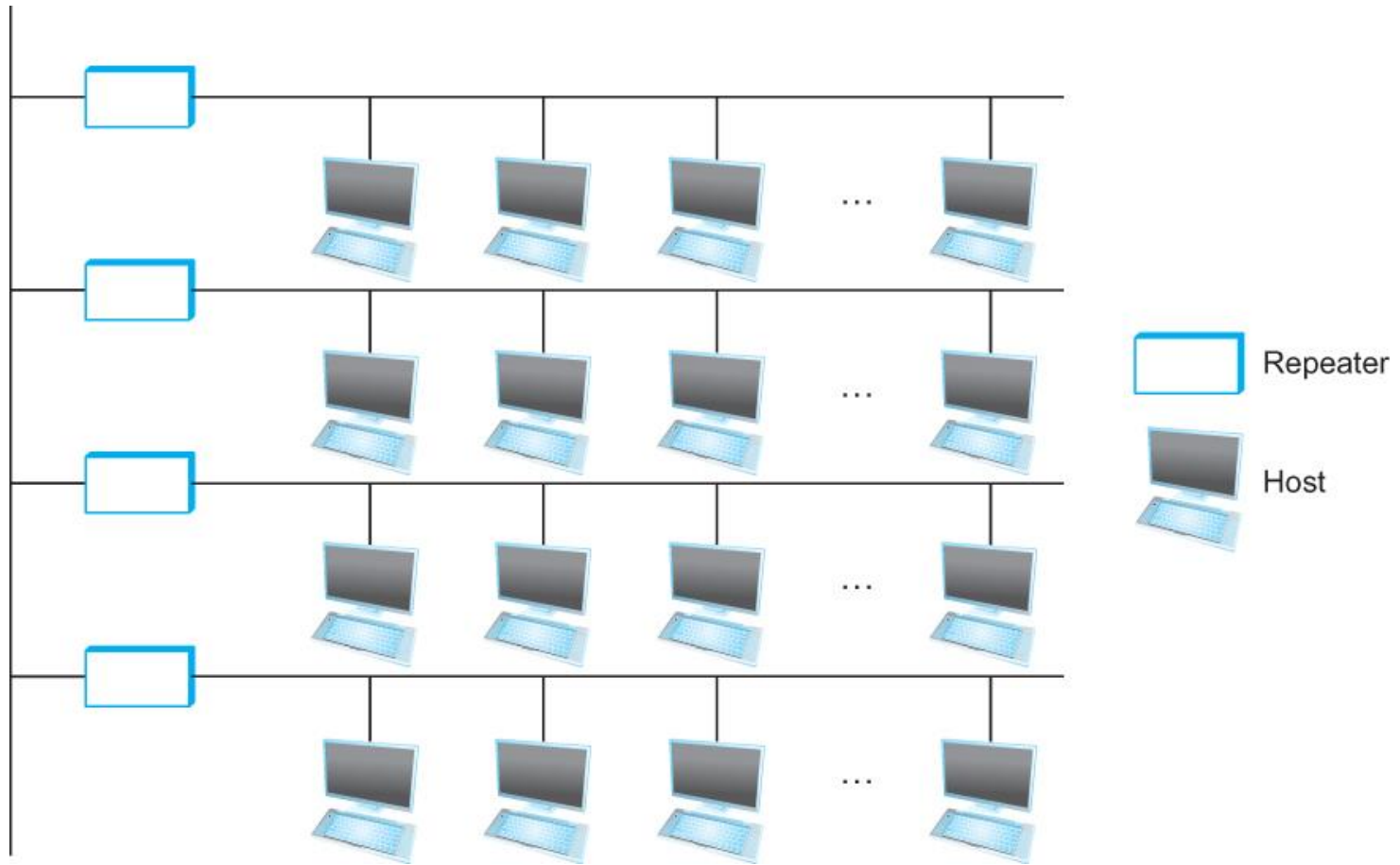


Ethernet transceiver and adaptor

Ethernet

- Multiple Ethernet segments can be joined together by *repeaters*.
- A *repeater* is a device that forwards digital signals.
- No more than four repeaters may be positioned between any pair of hosts.
 - An Ethernet has a total reach of only 2500 m.

Ethernet



Ethernet repeater

Ethernet

- Any signal placed on the Ethernet by a host is broadcast over the entire network
 - Signal is propagated in both directions.
 - Repeaters forward the signal on all outgoing segments.
 - Terminators attached to the end of each segment absorb the signal.
- Ethernet uses Manchester encoding scheme.

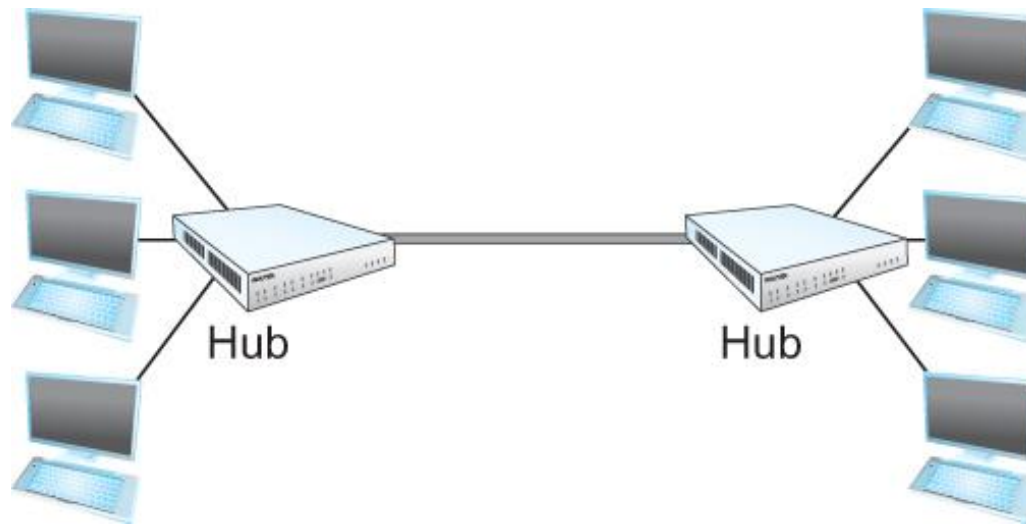
Ethernet

- New Technologies in Ethernet
 - Instead of using coax cable, an Ethernet can be constructed from a thinner cable known as 10Base2 (the original was 10Base5)
 - 10 means the network operates at 10 Mbps
 - Base means the cable is used in a baseband system
 - 2 means that a given segment can be no longer than 200 m

Ethernet

- New Technologies in Ethernet
 - Another cable technology is 10BaseT
 - T stands for twisted pair
 - Limited to 100 m in length
 - With 10BaseT, the common configuration is to have several point to point segments coming out of a multiway repeater, called *Hub*

Ethernet

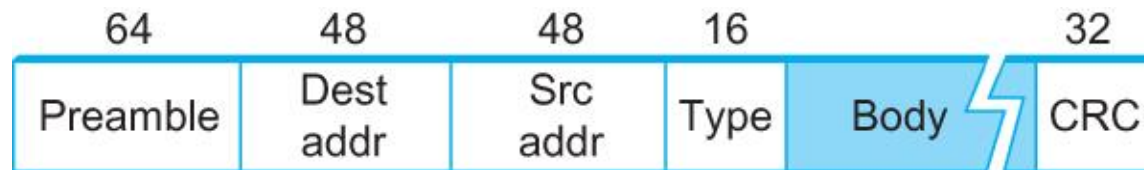


Ethernet Hub

Access Protocol for Ethernet

- The algorithm is commonly called Ethernet's Media Access Control (MAC).
 - It is implemented in Hardware on the network adaptor.
- Frame format
 - Preamble (64bit): allows the receiver to synchronize with the signal (sequence of alternating 0s and 1s).
 - Host and Destination Address (48bit each).
 - Packet type (16bit): acts as demux key to identify the higher level protocol.
 - Data (up to 1500 bytes)
 - Minimally a frame must contain at least 46 bytes of data.
 - Frame must be long enough to detect collision.
 - CRC (32bit)

Ethernet Frame



Ethernet Frame Format

Ethernet Addresses

- Each host on an Ethernet (in fact, every Ethernet host in the world) has a unique Ethernet Address.
- The address belongs to the adaptor, not the host.
 - It is usually burnt into ROM.
- Ethernet addresses are typically printed in a human readable format
 - As a sequence of six numbers separated by colons.
 - Each number corresponds to 1 byte of the 6 byte address and is given by a pair of hexadecimal digits, one for each of the 4-bit nibbles in the byte
 - Leading 0s are dropped.
 - For example, 8:0:2b:e4:b1:2 is
 - 00001000 00000000 00101011 11100100 10110001 00000010

Ethernet Addresses

- To ensure that every adaptor gets a unique address, each manufacturer of Ethernet devices is allocated a different prefix that must be prepended to the address on every adaptor they build
 - AMD has been assigned the 24bit prefix 8:0:20

Ethernet Addresses

- Each frame transmitted on an Ethernet is received by every adaptor connected to that Ethernet.
- Each adaptor recognizes those frames addressed to its address and passes only those frames on to the host.
- In addition, to *unicast* address, an Ethernet address consisting of all 1s is treated as a *broadcast* address.
 - All adaptors pass frames addressed to the *broadcast* address up to the host.
- Similarly, an address that has the first bit set to 1 but is not the *broadcast* address is called a *multicast* address.
 - A given host can program its adaptor to accept some set of *multicast* addresses.

Ethernet Addresses

- To summarize, an Ethernet adaptor receives all frames and accepts
 - Frames addressed to its own address
 - Frames addressed to the broadcast address
 - Frames addressed to a multicast address if it has been instructed

Ethernet Transmitter Algorithm

- When the adaptor has a frame to send and the line is idle, it transmits the frame immediately.
 - The upper bound of 1500 bytes in the message means that the adaptor can occupy the line for a fixed length of time.
- When the adaptor has a frame to send and the line is busy, it waits for the line to go idle and then transmits immediately.
- The Ethernet is said to be 1-persistent protocol because an adaptor with a frame to send transmits with probability 1 whenever a busy line goes idle.

Ethernet Transmitter Algorithm

- Since there is no centralized control it is possible for two (or more) adaptors to begin transmitting at the same time,
 - Either because both found the line to be idle,
 - Or, both had been waiting for a busy line to become idle.
- When this happens, the two (or more) frames are said to be *collide* on the network.

Ethernet Transmitter Algorithm

- Since Ethernet supports collision detection, each sender is able to determine that a collision is in progress.
- At the moment an adaptor detects that its frame is colliding with another, it first makes sure to transmit a 32-bit jamming sequence and then stops transmission.
 - Thus, a transmitter will minimally send 96 bits in the case of collision
 - 64-bit preamble + 32-bit jamming sequence

Ethernet Transmitter Algorithm

- One way that an adaptor will send only 96 bit (called a *runt frame*) is if the two hosts are close to each other.
- Had they been farther apart,
 - They would have had to transmit longer, and thus send more bits, before detecting the collision.

Ethernet Transmitter Algorithm

- The worst case scenario happens when the two hosts are at opposite ends of the Ethernet.
- To know for sure that the frame its just sent did not collide with another frame, the transmitter may need to send as many as 512 bits.
 - Every Ethernet frame must be at least 512 bits (64 bytes) long.
 - 14 bytes of header + 46 bytes of data + 4 bytes of CRC

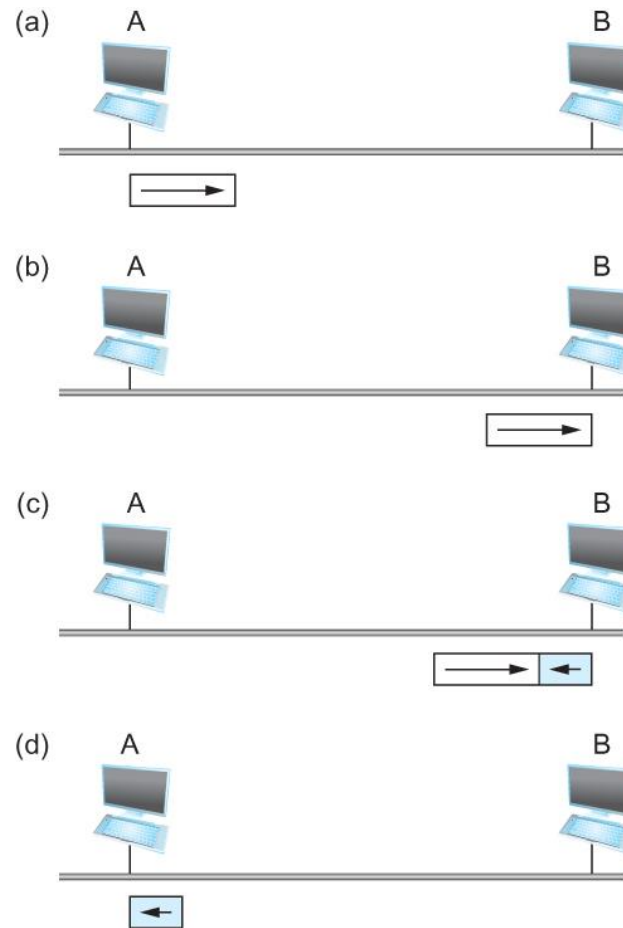
Ethernet Transmitter Algorithm

- Why 512 bits?
 - Why is its length limited to 2500 m?
- The farther apart two nodes are, the longer it takes for a frame sent by one to reach the other, and the network is vulnerable to collision during this time

Ethernet Transmitter Algorithm

- A begins transmitting a frame at time t
- d denotes the one link latency
- The first bit of A's frame arrives at B at time $t + d$
- Suppose an instant before host A's frame arrives, host B begins to transmit its own frame
- B's frame will immediately collide with A's frame and this collision will be detected by host B
- Host B will send the 32-bit jamming sequence
- Host A will not know that the collision occurred until B's frame reaches it, which will happen at $t + 2 * d$
- Host A must continue to transmit until this time in order to detect the collision
 - Host A must transmit for $2 * d$ to be sure that it detects all possible collisions

Ethernet Transmitter Algorithm



Worst-case scenario: (a) A sends a frame at time t ; (b) A's frame arrives at B at time $t + d$; (c) B begins transmitting at time $t + d$ and collides with A's frame; (d) B's runt (32-bit) frame arrives at A at time $t + 2d$.

Ethernet Transmitter Algorithm

- Consider that a maximally configured Ethernet is 2500 m long, and there may be up to four repeaters between any two hosts, the round trip delay has been determined to be $51.2 \mu\text{s}$
 - Which on 10 Mbps Ethernet corresponds to 512 bits
- The other way to look at this situation,
 - We need to limit the Ethernet's maximum latency to a fairly small value ($51.2 \mu\text{s}$) for the access algorithm to work
 - Hence the maximum length for the Ethernet is on the order of 2500 m.

Ethernet Transmitter Algorithm

- Once an adaptor has detected a collision, and stopped its transmission, it waits a certain amount of time and tries again.
- Each time the adaptor tries to transmit but fails, it doubles the amount of time it waits before trying again.
- This strategy of doubling the delay interval between each retransmission attempt is known as *Exponential Backoff*.

Ethernet Transmitter Algorithm

- The adaptor first delays either 0 or 51.2 μs , selected at random.
- If this effort fails, it then waits 0, 51.2, 102.4, 153.6 μs (selected randomly) before trying again;
 - This is $k * 51.2$ for $k = 0, 1, 2, 3$
- After the third collision, it waits $k * 51.2$ for $k = 0 \dots 2^3 - 1$ (again selected at random).
- In general, the algorithm randomly selects a k between 0 and $2^n - 1$ and waits for $k * 51.2 \mu\text{s}$, where n is the number of collisions experienced so far.

Experience with Ethernet

- Ethernets work best under lightly loaded conditions.
 - Under heavy loads, too much of the network's capacity is wasted by collisions.
- Most Ethernets are used in a conservative way.
 - Have fewer than 200 hosts connected to them which is far fewer than the maximum of 1024.
- Most Ethernets are far shorter than 2500m with a round-trip delay of closer to 5 μ s than 51.2 μ s.
- Ethernets are easy to administer and maintain.
 - There are no switches that can fail and no routing and configuration tables that have to be kept up-to-date.
 - It is easy to add a new host to the network.
 - It is inexpensive.
 - Cable is cheap, and only other cost is the network adaptor on each host.

Wireless Links

- Wireless links transmit electromagnetic signals
 - Radio, microwave, infrared
- Wireless links all share the same “wire” (so to speak)
 - The challenge is to share it efficiently without unduly interfering with each other
 - Most of this sharing is accomplished by dividing the “wire” along the dimensions of frequency and space
- Exclusive use of a particular frequency in a particular geographic area may be allocated to an individual entity such as a corporation

Wireless Links

- These allocations are determined by government agencies such as FCC (Federal Communications Commission) in USA
- Specific bands (frequency) ranges are allocated to certain uses.
 - Some bands are reserved for government use
 - Other bands are reserved for uses such as AM radio, FM radio, televisions, satellite communications, and cell phones
 - Specific frequencies within these bands are then allocated to individual organizations for use within certain geographical areas.
 - Finally, there are several frequency bands set aside for “license exempt” usage
 - Bands in which a license is not needed

Wireless Links

- Devices that use license-exempt frequencies are still subject to certain restrictions
 - The first is a limit on transmission power
 - This limits the range of signal, making it less likely to interfere with another signal
 - For example, a cordless phone might have a range of about 100 feet.

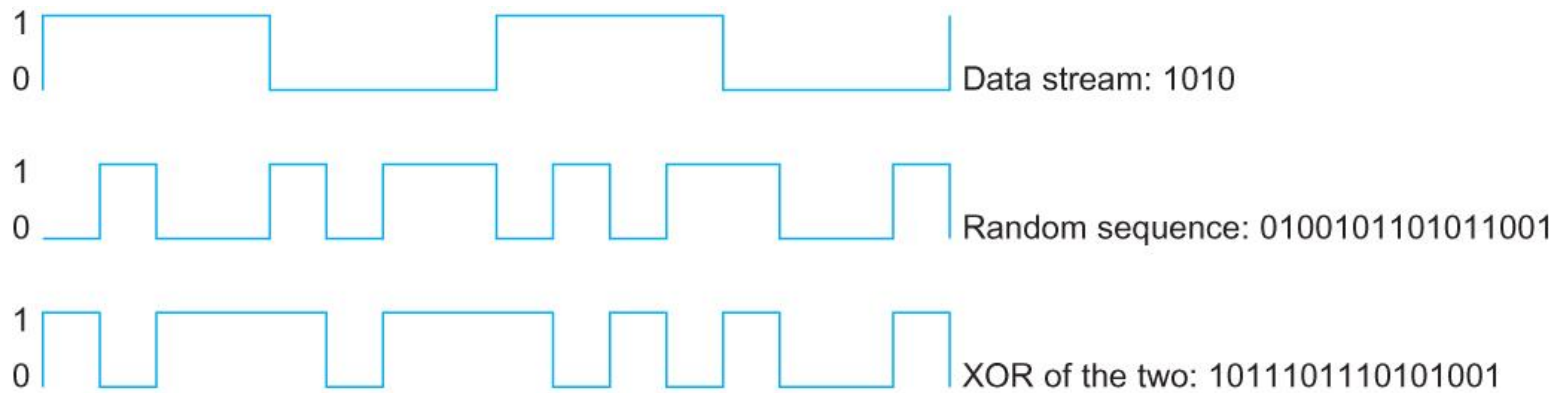
Wireless Links

- The second restriction requires the use of **Spread Spectrum** technique
 - Idea is to spread the signal over a wider frequency band
 - So as to minimize the impact of interference from other devices
 - Originally designed for military use
 - ***Frequency hopping***
 - Transmitting signal over a random sequence of frequencies
 - First transmitting at one frequency, then a second, then a third...
 - The sequence of frequencies is not truly random, instead computed algorithmically by a pseudorandom number generator
 - The receiver uses the same algorithm as the sender, initializes it with the same seed, and is
 - Able to hop frequencies in sync with the transmitter to correctly receive the frame

Wireless Links

- A second spread spectrum technique called ***Direct sequence***
 - Represents each bit in the frame by multiple bits in the transmitted signal.
 - For each bit the sender wants to transmit
 - It actually sends the exclusive OR of that bit and n random bits
 - The sequence of random bits is generated by a pseudorandom number generator known to both the sender and the receiver.
 - The transmitted values, known as an ***n -bit chipping code***, spread the signal across a frequency band that is n times wider

Wireless Links



Example 4-bit chipping sequence

Wireless Links

- Wireless technologies differ in a variety of dimensions
 - How much bandwidth they provide
 - How far apart the communication nodes can be

- Four prominent wireless technologies
 - Bluetooth
 - Wi-Fi (more formally known as 802.11)
 - WiMAX (802.16)
 - 3G cellular wireless

Wireless Links

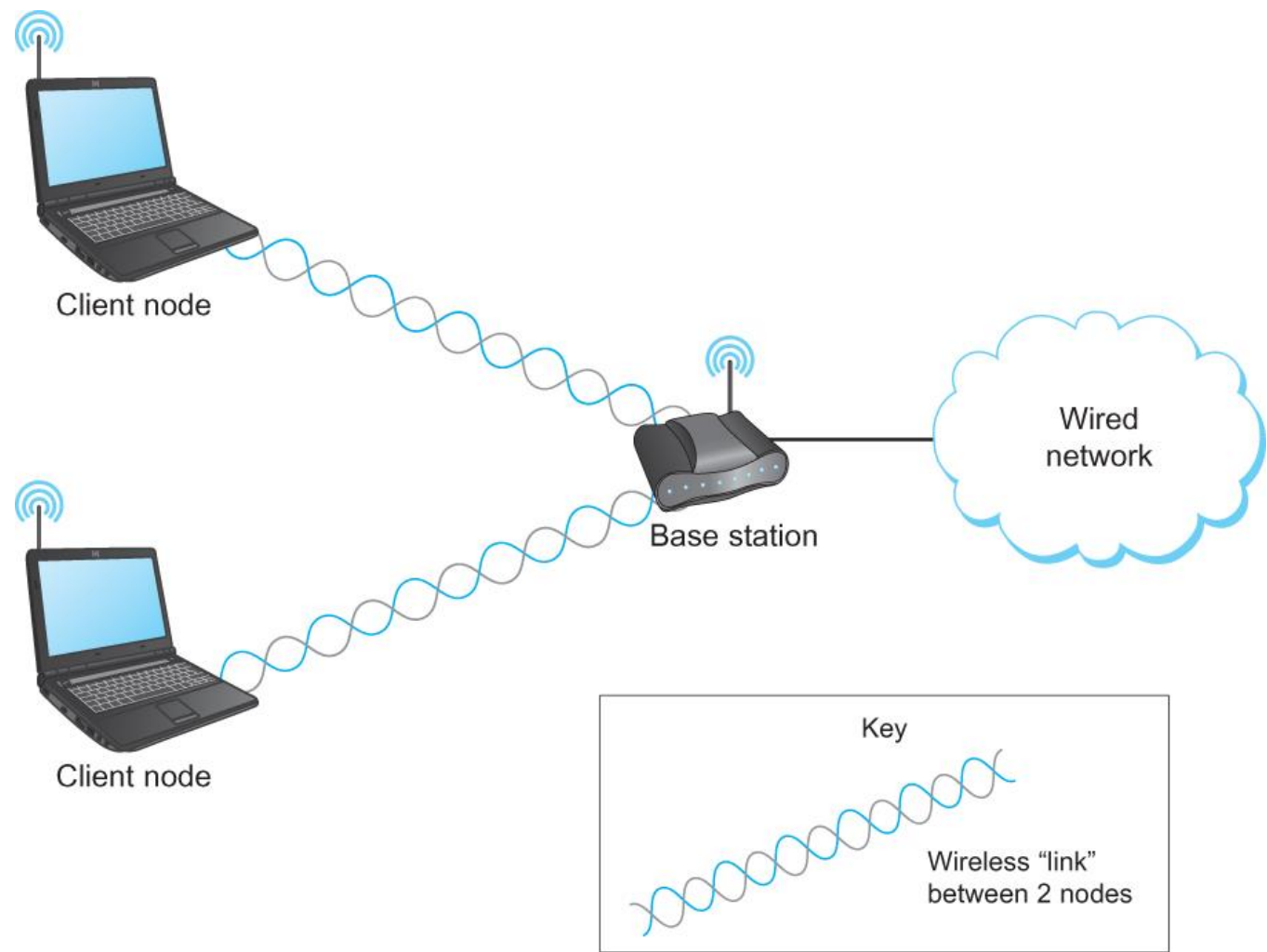
	Bluetooth (802.15.1)	Wi-Fi (802.11)	3G Cellular
Typical link length	10 m	100 m	Tens of kilometers
Typical data rate	2 Mbps (shared)	54 Mbps (shared)	Hundreds of kbps (per connection)
Typical use	Link a peripheral to a computer	Link a computer to a wired base	Link a mobile phone to a wired tower
Wired technology analogy	USB	Ethernet	DSL

Overview of leading wireless technologies

Wireless Links

- Mostly widely used wireless links today are usually asymmetric
 - Two end-points are usually different kinds of nodes
 - One end-point usually has no mobility, but has wired connection to the Internet (known as **base station**)
 - The node at the other end of the link is often mobile

Wireless Links



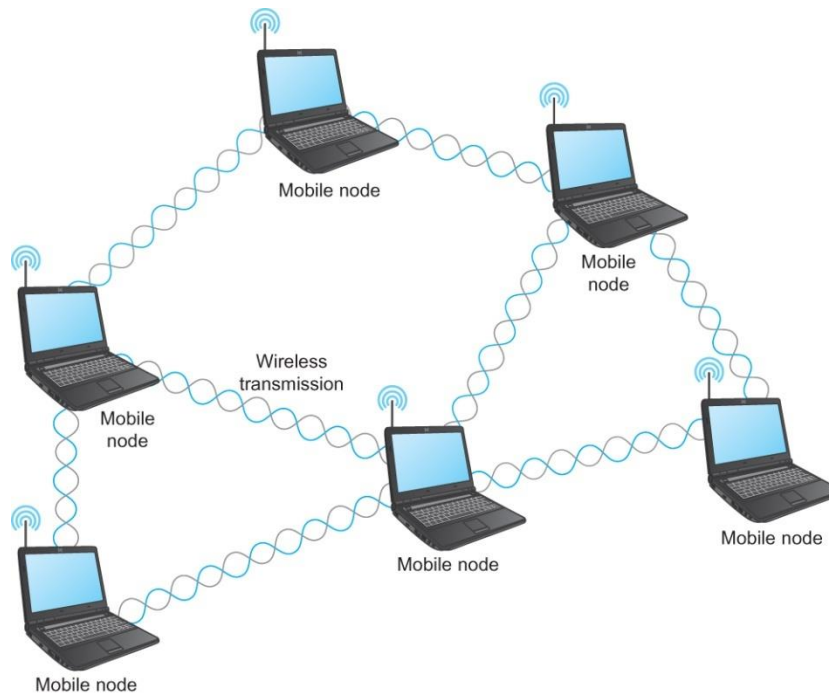
A wireless network using a base station

Wireless Links

- Wireless communication supports point-to-multipoint communication
- Communication between non-base (client) nodes is routed via the base station
- Three levels of mobility for clients
 - No mobility: the receiver must be in a fix location to receive a directional transmission from the base station (initial version of WiMAX)
 - Mobility is within the range of a base (Bluetooth)
 - Mobility between bases (Cell phones and Wi-Fi)

Wireless Links

- Mesh or Ad-hoc network
 - Nodes are peers
 - Messages may be forwarded via a chain of peer nodes



A wireless ad-hoc or mesh network

IEEE 802.11

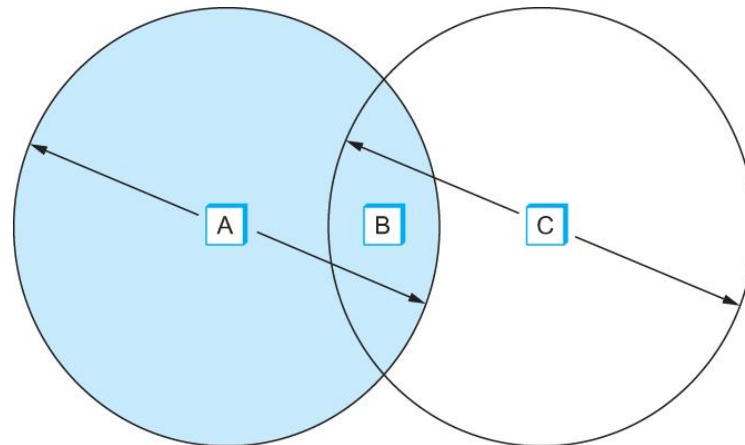
- Also known as Wi-Fi
- Like its Ethernet and token ring siblings, 802.11 is designed for use in a limited geographical area (homes, office buildings, campuses)
 - Primary challenge is to mediate access to a shared communication medium – in this case, signals propagating through space
- 802.11 supports additional features
 - power management and
 - security mechanisms

IEEE 802.11

- Original 802.11 standard defined two radio-based physical layer standard
 - One using the frequency hopping
 - Over 79 1-MHz-wide frequency bandwidths
 - Second using direct sequence
 - Using 11-bit chipping sequence
 - Both standards run in the 2.4-GHz and provide up to 2 Mbps
- Then physical layer standard 802.11b was added
 - Using a variant of direct sequence 802.11b provides up to 11 Mbps
 - Uses license-exempt 2.4-GHz band
- Then came 802.11a which delivers up to 54 Mbps using OFDM
 - 802.11a runs on license-exempt 5-GHz band
- Most recent standard is 802.11g which is backward compatible with 802.11b
 - Uses 2.4 GHz band, OFDM and delivers up to 54 Mbps

IEEE 802.11 – Collision Avoidance

- Consider the situation in the following figure where each of four nodes is able to send and receive signals that reach just the nodes to its immediate left and right
 - For example, B can exchange frames with A and C, but it cannot reach D
 - C can reach B and D but not A

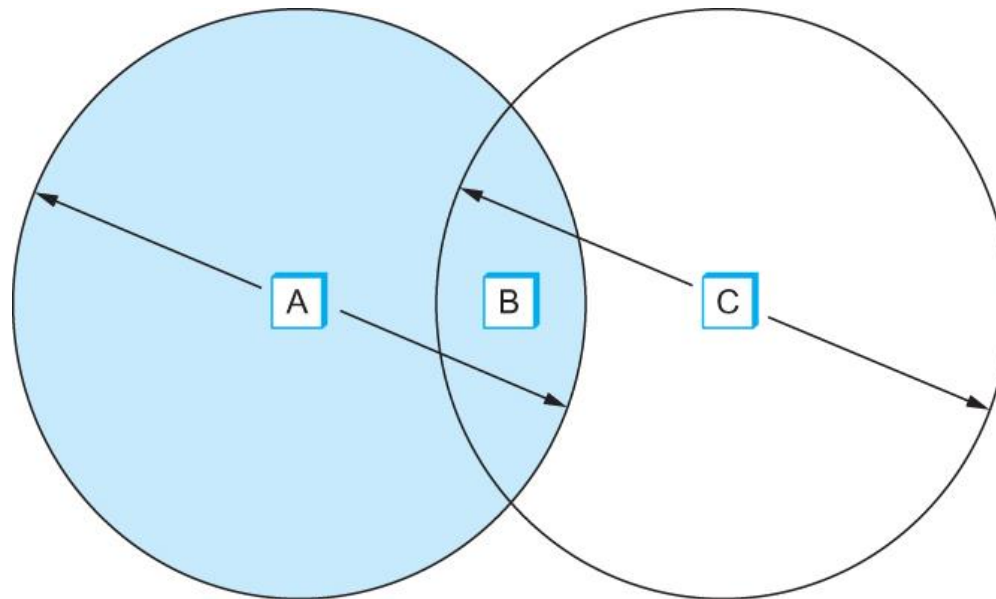


Example of a wireless network

IEEE 802.11 – Collision Avoidance

- Suppose both A and C want to communicate with B and so they each send it a frame.
 - A and C are unaware of each other since their signals do not carry that far
 - These two frames collide with each other at B
 - But unlike an Ethernet, neither A nor C is aware of this collision
 - A and C are said to *hidden nodes* with respect to each other

IEEE 802.11 – Collision Avoidance

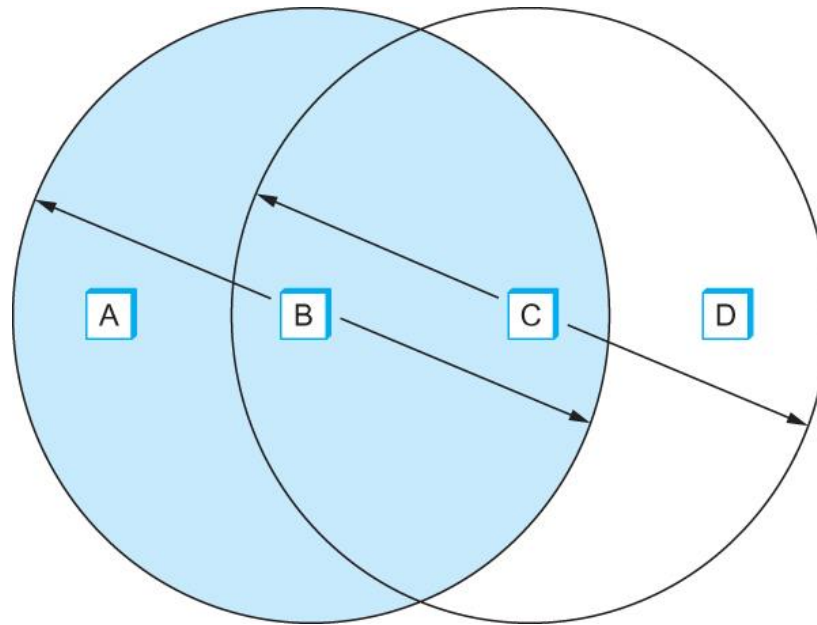


The “Hidden Node” Problem. Although A and C are hidden from each other, their signals can collide at B. (B’s reach is not shown.)

IEEE 802.11 – Collision Avoidance

- Another problem called *exposed node* problem occurs
 - Suppose B is sending to A. Node C is aware of this communication because it hears B's transmission.
 - It would be a mistake for C to conclude that it cannot transmit to anyone just because it can hear B's transmission.
 - Suppose C wants to transmit to node D.
 - This is not a problem since C's transmission to D will not interfere with A's ability to receive from B.

IEEE 802.11 – Collision Avoidance



Exposed Node Problem. Although B and C are exposed to each other's signals, there is no interference if B transmits to A while C transmits to D. (A and D's reaches are not shown.)

IEEE 802.11 – Collision Avoidance

- 802.11 addresses these two problems with an algorithm called Multiple Access with Collision Avoidance (MACA).
- Key Idea
 - Sender and receiver exchange control frames with each other before the sender actually transmits any data.
 - This exchange informs all nearby nodes that a transmission is about to begin
 - Sender transmits a *Request to Send* (RTS) frame to the receiver.
 - The RTS frame includes a field that indicates how long the sender wants to hold the medium
 - Length of the data frame to be transmitted
 - Receiver replies with a *Clear to Send* (CTS) frame
 - This frame echoes this length field back to the sender

IEEE 802.11 – Collision Avoidance

- Any node that sees the CTS frame knows that
 - it is close to the receiver, therefore
 - cannot transmit for the period of time it takes to send a frame of the specified length
- Any node that sees the RTS frame but not the CTS frame
 - is not close enough to the receiver to interfere with it, and
 - so is free to transmit

IEEE 802.11 – Collision Avoidance

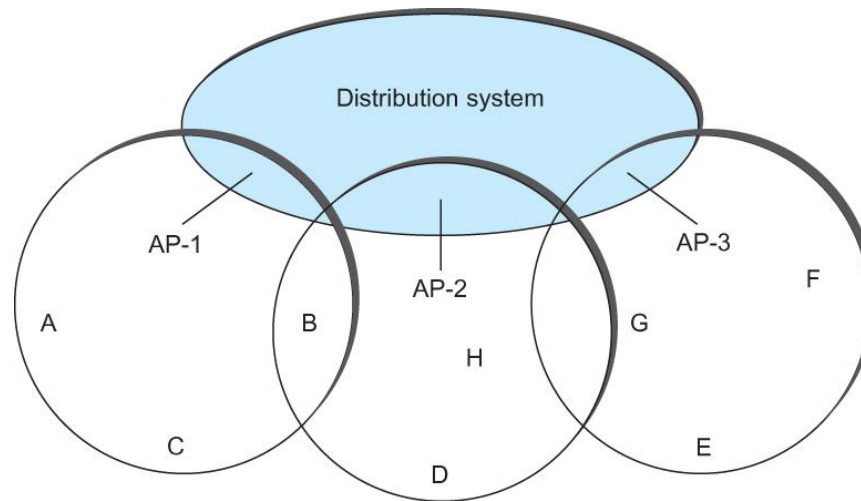
- Using ACK in MACA
 - Proposed in MACAW: MACA for Wireless LANs
- Receiver sends an ACK to the sender after successfully receiving a frame
- All nodes must wait for this ACK before trying to transmit
- If two or more nodes detect an idle link and try to transmit an RTS frame at the same time
 - Their RTS frame will collide with each other
- 802.11 does not support collision detection
 - So the senders realize the collision has happened when they do not receive the CTS frame after a period of time
 - In this case, they each wait a random amount of time before trying again.
 - The amount of time a given node delays is defined by the same *exponential backoff* algorithm used on the Ethernet.

IEEE 802.11 – Distribution System

- 802.11 is suitable for an ad-hoc configuration of nodes that may or may not be able to communicate with all other nodes.
- Nodes are free to move around
- The set of directly reachable nodes may change over time
- To deal with this mobility and partial connectivity,
 - 802.11 defines additional structures on a set of nodes
 - Instead of all nodes being created equal,
 - some nodes are allowed to roam
 - some are connected to a wired network infrastructure
 - they are called *Access Points (AP)* and they are connected to each other by a so-called *distribution system*

IEEE 802.11 – Distribution System

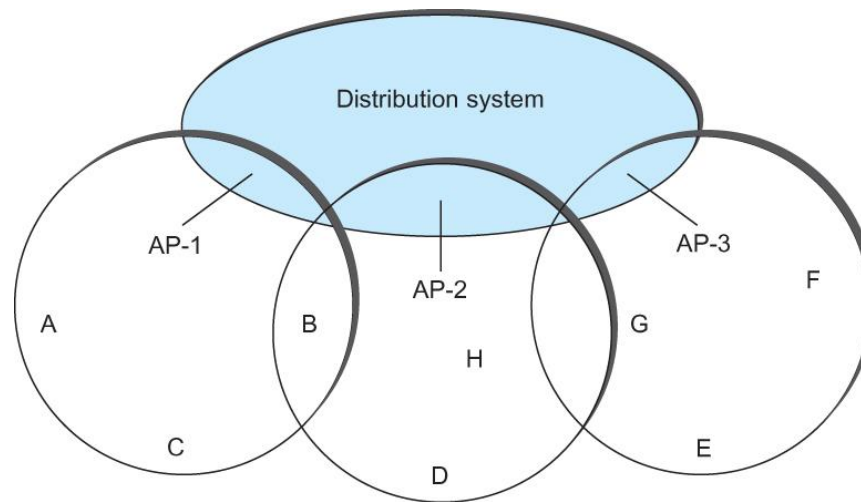
- Following figure illustrates a distribution system that connects three access points, each of which services the nodes in the same region
- Each of these regions is analogous to a cell in a cellular phone system with the APIs playing the same role as a base station
- The distribution network runs at layer 2 of the ISO architecture



Access points connected to a distribution network

IEEE 802.11 – Distribution System

- Although two nodes can communicate directly with each other if they are within reach of each other, the idea behind this configuration is
 - Each nodes associates itself with one access point
 - For node A to communicate with node E, A first sends a frame to its AP-1 which forwards the frame across the distribution system to AP-3, which finally transmits the frame to E



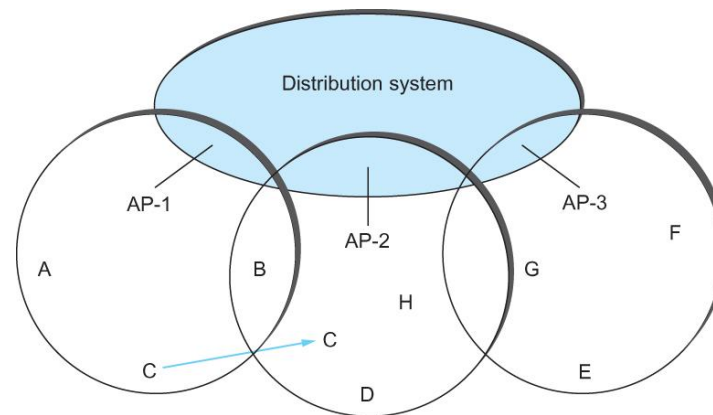
Access points connected to a distribution network

IEEE 802.11 – Distribution System

- How do the nodes select their access points
- How does it work when nodes move from one cell to another
- The technique for selecting an AP is called *scanning*
 - The node sends a *Probe* frame
 - All APs within reach reply with a *Probe Response* frame
 - The node selects one of the access points and sends that AP an *Association Request* frame
 - The AP replies with an *Association Response* frame
- A node engages this protocol whenever
 - it joins the network, as well as
 - when it becomes unhappy with its current AP
 - This might happen, for example, because the signal from its current AP has weakened due to the node moving away from it
 - Whenever a node acquires a new AP, the new AP notifies the old AP of the change via the distribution system

IEEE 802.11 – Distribution System

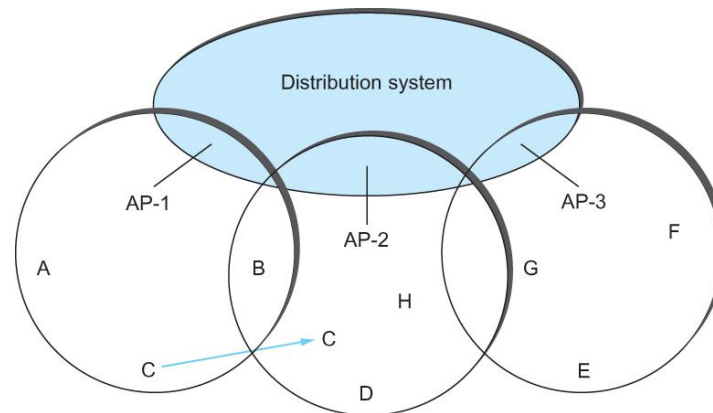
- Consider the situation shown in the following figure when node C moves from the cell serviced by AP-1 to the cell serviced by AP-2.
- As it moves, it sends *Probe* frames, which eventually result in *Probe Responses* from AP-2.
- At some point, C prefers AP-2 over AP-1, and so it associates itself with that access point.
 - This is called *active scanning* since the node is actively searching for an access point



Node Mobility

IEEE 802.11 – Distribution System

- APs also periodically send a *Beacon* frame that advertises the capabilities of the access point; these include the transmission rate supported by the AP
 - This is called *passive scanning*
 - A node can change to this AP based on the *Beacon* frame simply by sending it an *Association Request* frame back to the access point.



Node Mobility

IEEE 802.11 – Frame Format

- Source and Destinations addresses: each 48 bits
- Data: up to 2312 bytes
- CRC: 32 bit
- Control field: 16 bits
 - Contains three subfields (of interest)
 - 6 bit **Type** field: indicates whether the frame is an RTS or CTS frame or being used by the scanning algorithm
 - A pair of 1 bit fields : called **ToDS** and **FromDS**



Frame Format

IEEE 802.11 – Frame Format

- Frame contains four addresses
- How these addresses are interpreted depends on the settings of the **ToDS** and **FromDS** bits in the frame's Control field
- This is to account for the possibility that the frame had to be forwarded across the distribution system which would mean that,
 - the original sender is not necessarily the same as the most recent transmitting node
- Same is true for the destination address
- Simplest case
 - When one node is sending directly to another, both the DS bits are 0, Addr1 identifies the target node, and Addr2 identifies the source node

IEEE 802.11 – Frame Format

- Most complex case
 - Both DS bits are set to 1
 - Indicates that the message went from a wireless node onto the distribution system, and then from the distribution system to another wireless node
 - With both bits set,
 - Addr1 identifies the ultimate destination,
 - Addr2 identifies the immediate sender (the one that forwarded the frame from the distribution system to the ultimate destination)
 - Addr3 identifies the intermediate destination (the one that accepted the frame from a wireless node and forwarded across the distribution system)
 - Addr4 identifies the original source

- Addr1: E, Addr2: AP-3, Addr3: AP-1, Addr4: A

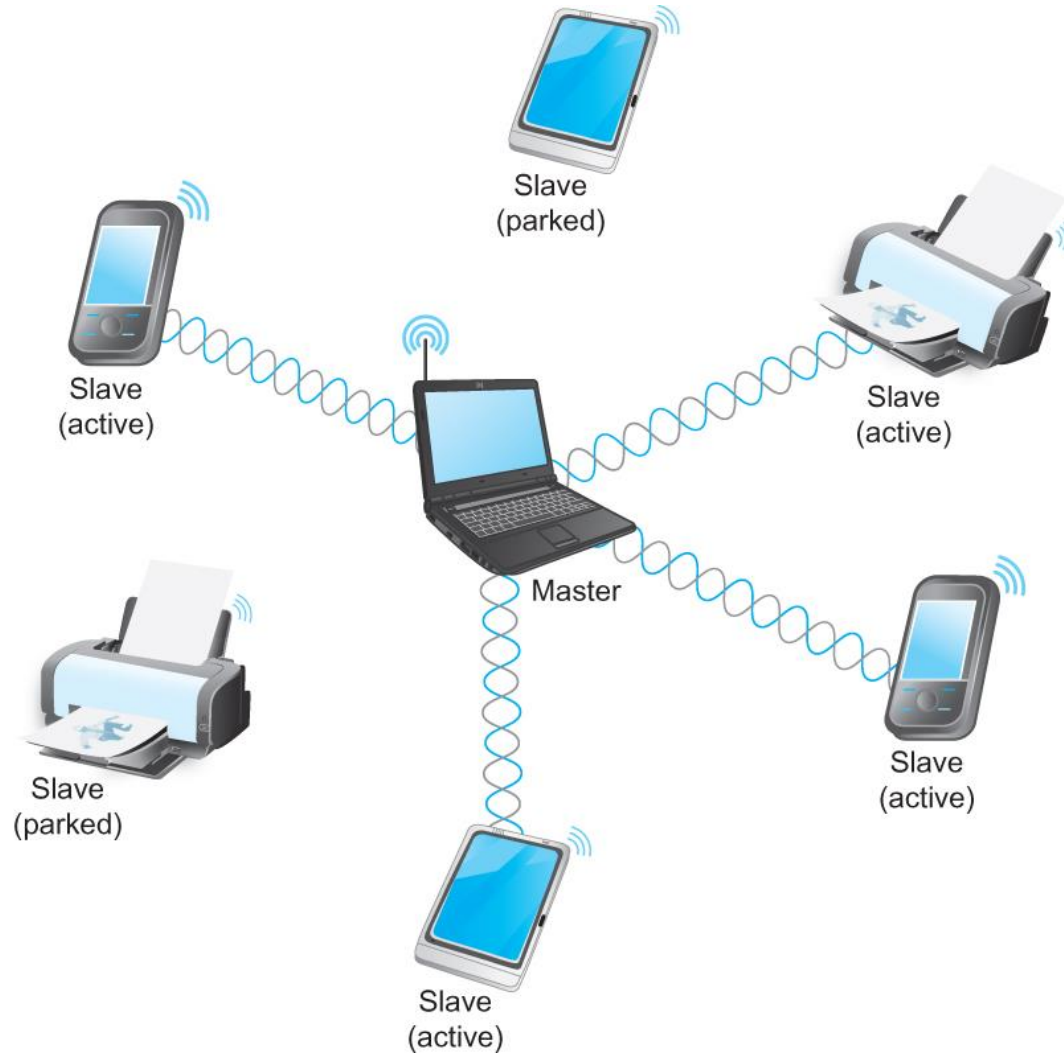
Bluetooth

- Used for very short range communication between mobile phones, PDAs, notebook computers and other personal or peripheral devices
- Operates in the license-exempt band at 2.45 GHz
- Has a range of only 10 m
- Communication devices typically belong to one individual or group
 - Sometimes categorized as Personal Area Network (PAN)
- Version 2.0 provides speeds up to 2.1 Mbps
- Power consumption is low

Bluetooth

- Bluetooth is specified by an industry consortium called the Bluetooth Special Interest Group
- It specifies an entire suite of protocols, going beyond the link layer to define application protocols, which it calls *profiles*, for a range of applications
 - There is a profile for synchronizing a PDA with personal computer
 - Another profile gives a mobile computer access to a wired LAN
- The basic Bluetooth network configuration is called a *piconet*
 - Consists of a master device and up to seven slave devices
 - Any communication is between the master and a slave
 - The slaves do not communicate directly with each other
 - A slave can be *parked*: set to an inactive, low-power state

Bluetooth



A Bluetooth Piconet

ZigBee

- ZigBee is a new technology that competes with Bluetooth
- Devised by the ZigBee alliance and standardized as IEEE 802.15.4
- It is designed for situations where the bandwidth requirements are low and power consumption must be very low to give very long battery life
- It is also intended to be simpler and cheaper than Bluetooth, making it financially feasible to incorporate in cheaper devices such as a wall switch that wirelessly communicates with a ceiling-mounted fan

Summary

- We introduced the many and varied type of links that are used to connect users to existing networks, and to construct large networks from scratch.
- We looked at the five key issues that must be addressed so that two or more nodes connected by some medium can exchange messages with each other
 - Encoding
 - Framing
 - Error Detecting
 - Reliability
 - Multiple Access Links
 - Ethernet
 - Wireless 802.11, Bluetooth

Switching and Forwarding

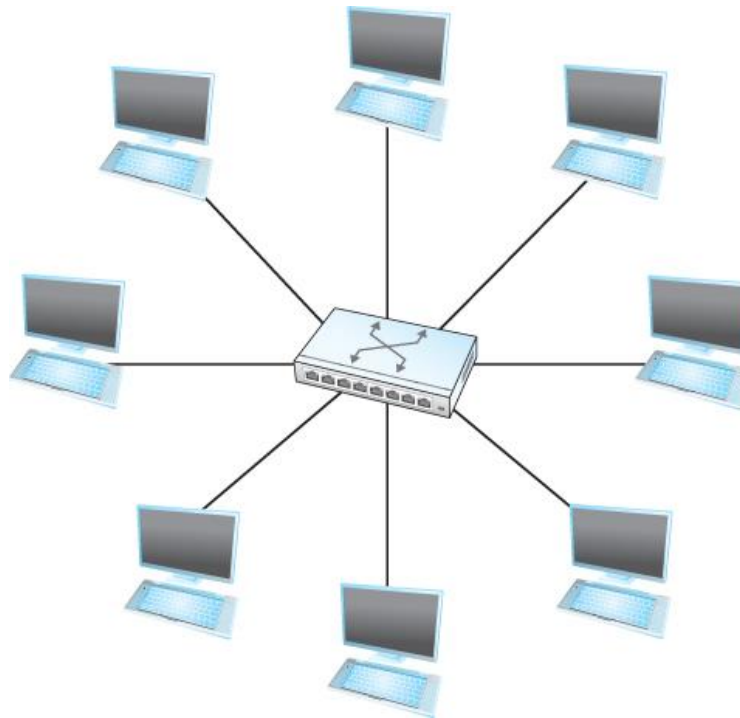
- Store-and-Forward Switches
- Bridges and Extended LANs
- Cell Switching
- Segmentation and Reassembly

Switching and Forwarding

- Switch
 - A mechanism that allows us to interconnect links to form a large network
 - A multi-input, multi-output device which transfers packets from an input to one or more outputs

Switching and Forwarding

Adds the star topology to the point-to-point link, bus (Ethernet), and ring (802.5 and FDDI) topologies



Switching and Forwarding

Properties of this star topology

- Even though a switch has a fixed number of inputs and outputs, which limits the number of hosts that can be connected to a single switch, large networks can be built by interconnecting a number of switches
- We can connect switches to each other and to hosts using point-to-point links, which typically means that we can build networks of large geographic scope
- Adding a new host to the network by connecting it to a switch does not necessarily mean that the hosts already connected will get worse performance from the network

Switching and Forwarding

The last claim cannot be made for the shared media network (discussed in Chapter 2)

- It is impossible for two hosts on the same Ethernet to transmit continuously at 10Mbps because they share the same transmission medium
- Every host on a switched network has its own link to the switch
 - So it may be entirely possible for many hosts to transmit at the full link speed (bandwidth) provided that the switch is designed with enough aggregate capacity

Switching and Forwarding

- A switch is connected to a set of links and for each of these links, runs the appropriate data link protocol to communicate with that node
- A switch's primary job is to receive incoming packets on one of its links and to transmit them on some other link
 - This function is referred as *switching and forwarding*
 - According to OSI architecture this is the main function of the network layer

Switching and Forwarding

- How does the switch decide which output port to place each packet on?
 - It looks at the header of the packet for an identifier that it uses to make the decision
 - Two common approaches
 - *Datagram or Connectionless approach*
 - *Virtual circuit or Connection-oriented approach*
 - A third approach *source routing* is less common

Switching and Forwarding

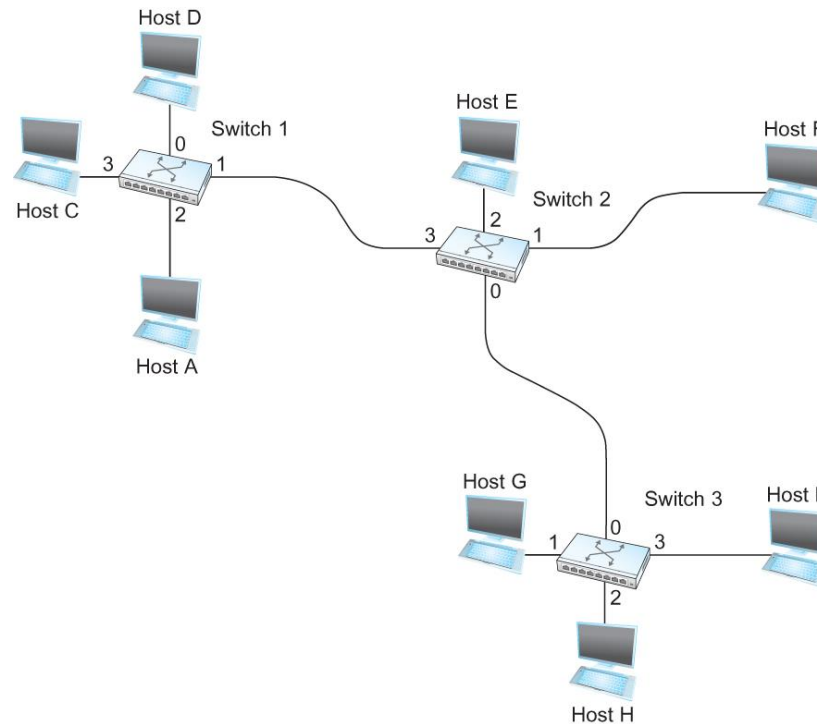
- Assumptions
 - Each host has a globally unique address
 - There is some way to identify the input and output ports of each switch
 - We can use numbers
 - We can use names

Switching and Forwarding

- Datagrams
 - Key Idea
 - Every packet contains enough information to enable any switch to decide how to get it to destination
 - Every packet contains the complete destination address

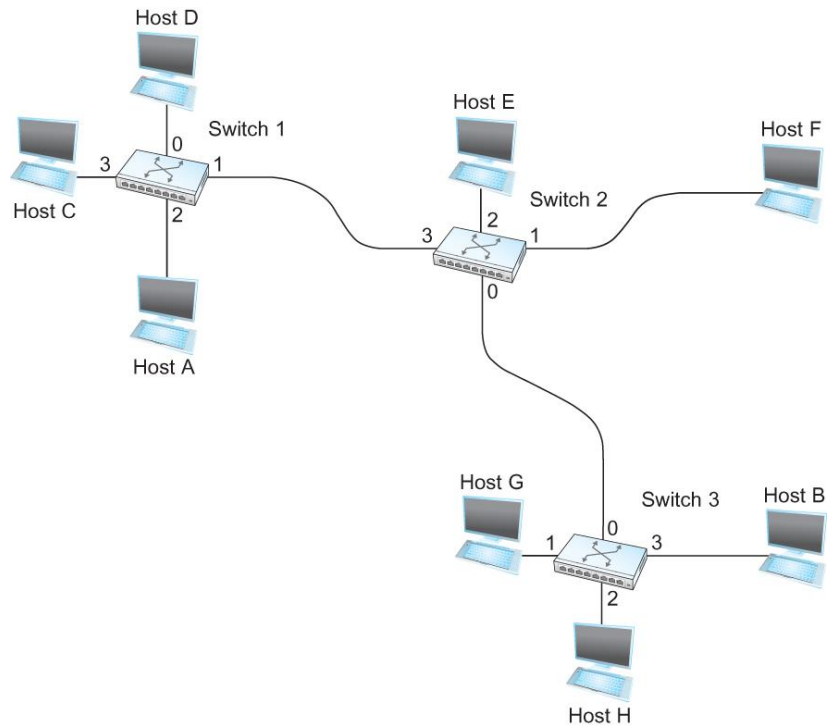
Switching and Forwarding

An example network



- To decide how to forward a packet, a switch consults a *forwarding table* (sometimes called a *routing table*)

Switching and Forwarding



Destination	Port

A	3
B	0
C	3
D	3
E	2
F	1
G	0
H	0

Forwarding Table for Switch 2

Switching and Forwarding

Characteristics of Connectionless (Datagram) Network

- A host can send a packet anywhere at any time, since any packet that turns up at the switch can be immediately forwarded (assuming a correctly populated forwarding table)
- When a host sends a packet, it has no way of knowing if the network is capable of delivering it or if the destination host is even up and running
- Each packet is forwarded independently of previous packets that might have been sent to the same destination.
 - Thus two successive packets from host A to host B may follow completely different paths
- A switch or link failure might not have any serious effect on communication if it is possible to find an alternate route around the failure and update the forwarding table accordingly

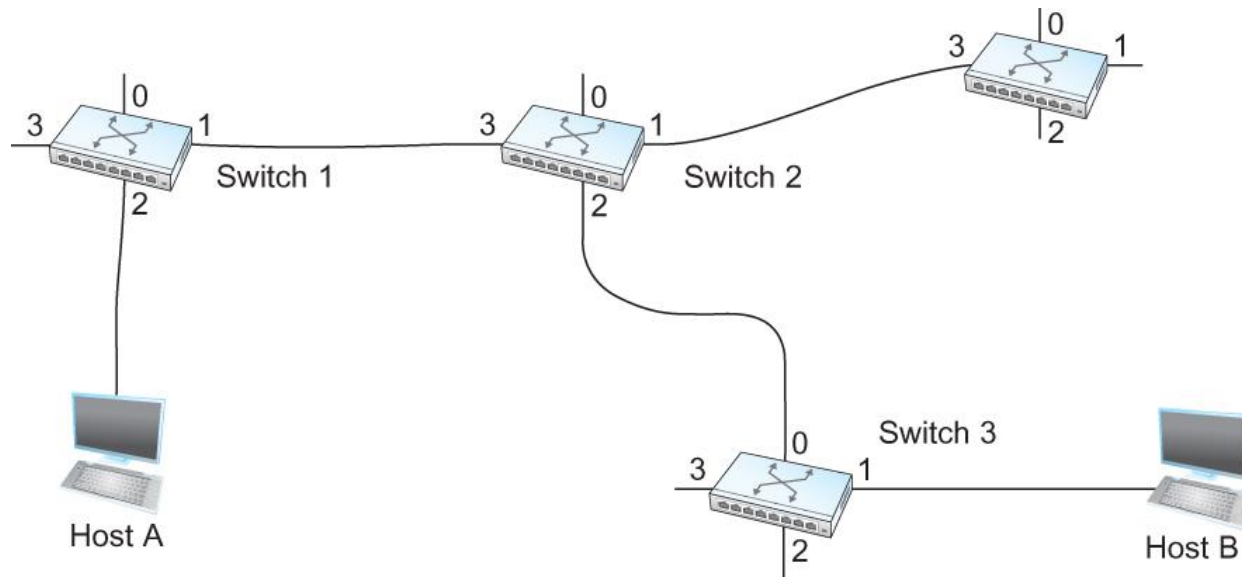
Switching and Forwarding

Virtual Circuit Switching

- Widely used technique for packet switching
- Uses the concept of *virtual circuit* (VC)
- Also called a connection-oriented model
- First set up a virtual connection from the source host to the destination host and then send the data

Switching and Forwarding

- Host A wants to send packets to host B



Switching and Forwarding

Two-stage process

- Connection setup
 - Data Transfer
-
- Connection setup
 - Establish “connection state” in each of the switches between the source and destination hosts
 - The connection state for a single connection consists of an entry in the “VC table” in each switch through which the connection passes

Switching and Forwarding

One entry in the VC table on a single switch contains

- A virtual circuit identifier (VCI) that uniquely identifies the connection at this switch and that will be carried inside the header of the packets that belong to this connection
 - An incoming interface on which packets for this VC arrive at the switch
 - An outgoing interface in which packets for this VC leave the switch
 - A potentially different VCI that will be used for outgoing packets
-
- The semantics for one such entry is
 - If a packet arrives on the designated incoming interface and that packet contains the designated VCI value in its header, then the packet should be sent out the specified outgoing interface with the specified outgoing VCI value first having been placed in its header

Switching and Forwarding

Note:

- The combination of the VCI of the packets as they are received at the switch and the interface on which they are received uniquely identifies the virtual connection
- There may be many virtual connections established in the switch at one time
- Incoming and outgoing VCI values are not generally the same
 - VCI is not a globally significant identifier for the connection; rather it has significance only on a given link
- Whenever a new connection is created, we need to assign a new VCI for that connection on each link that the connection will traverse
 - We also need to ensure that the chosen VCI on a given link is not currently in use on that link by some existing connection.

Switching and Forwarding

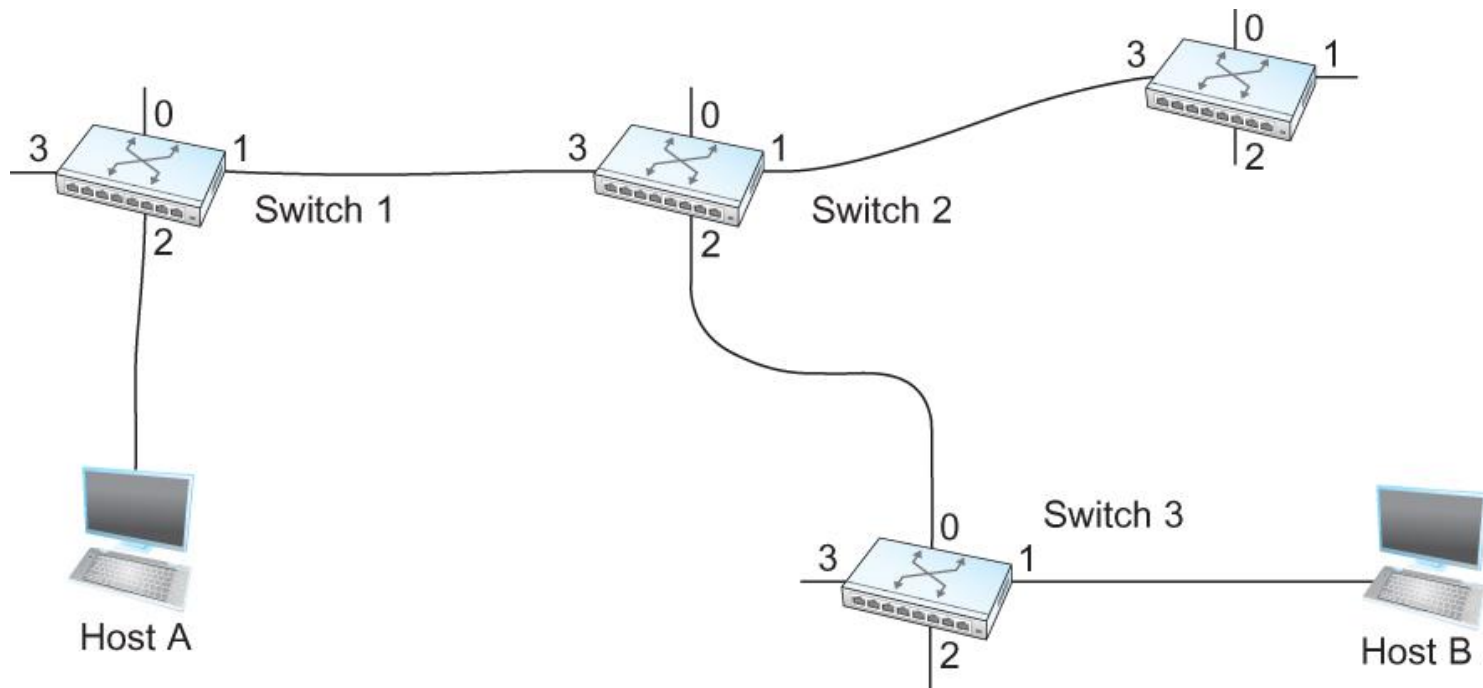
Two broad classes of approach to establishing connection state

- Network Administrator will configure the state
 - The virtual circuit is **permanent** (PVC)
 - The network administrator can delete this
 - Can be thought of as a long-lived or administratively configured VC
- A host can send messages into the network to cause the state to be established
 - This is referred as **signalling** and the resulting virtual circuit is said to be **switched** (SVC)
 - A host may set up and delete such a VC dynamically without the involvement of a network administrator

Switching and Forwarding

Let's assume that a network administrator wants to manually create a new virtual connection from host A to host B

- First the administrator identifies a path through the network from A to B



Switching and Forwarding

The administrator then picks a VCI value that is currently unused on each link for the connection

- For our example,
 - Suppose the VCI value 5 is chosen for the link from host A to switch 1
 - 11 is chosen for the link from switch 1 to switch 2
 - So the switch 1 will have an entry in the VC table

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
2	5	1	11

Switching and Forwarding

Similarly, suppose

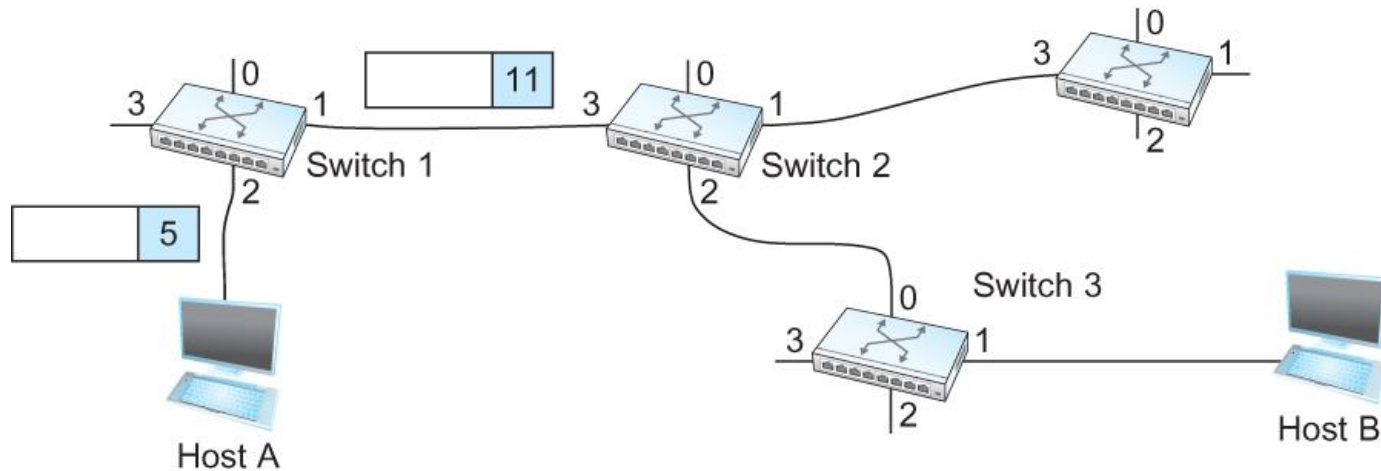
- VCI of 7 is chosen to identify this connection on the link from switch 2 to switch 3
- VCI of 4 is chosen for the link from switch 3 to host B
- Switches 2 and 3 are configured with the following VC table

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
3	11	2	7

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
0	7	1	4

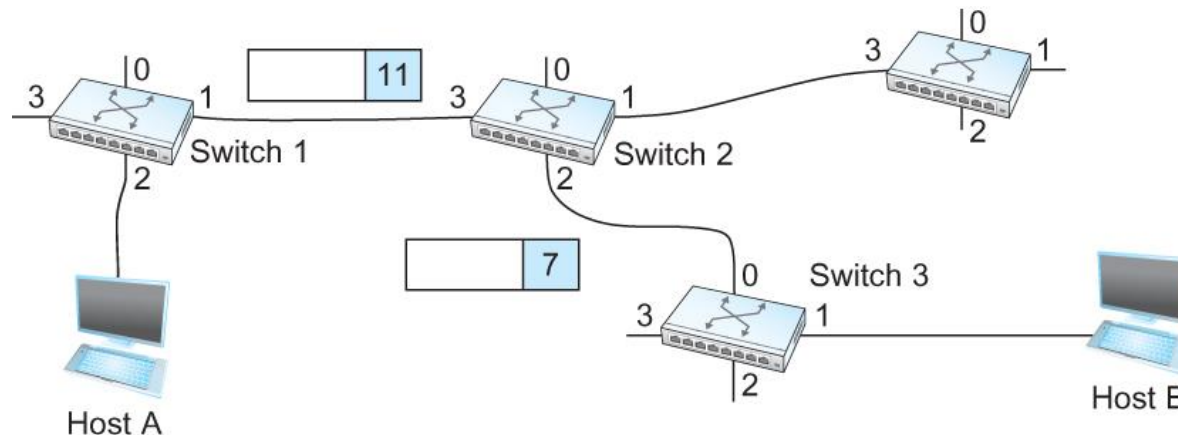
Switching and Forwarding

- For any packet that A wants to send to B, A puts the VCI value 5 in the header of the packet and sends it to switch 1
- Switch 1 receives any such packet on interface 2, and it uses the combination of the interface and the VCI in the packet header to find the appropriate VC table entry.
- The table entry on switch 1 tells the switch to forward the packet out of interface 1 and to put the VCI value 11 in the header



Switching and Forwarding

- Packet will arrive at switch 2 on interface 3 bearing VCI 11
- Switch 2 looks up interface 3 and VCI 11 in its VC table and sends the packet on to switch 3 after updating the VCI value appropriately
- This process continues until it arrives at host B with the VCI value of 4 in the packet
- To host B, this identifies the packet as having come from host A



Switching and Forwarding

- In real networks of reasonable size, the burden of configuring VC tables correctly in a large number of switches would quickly become excessive
 - Thus, some sort of signalling is almost always used, even when setting up “permanent” VCs
 - In case of PVCs, signalling is initiated by the network administrator
 - SVCs are usually set up using signalling by one of the hosts

Switching and Forwarding

- How does the signalling work
 - To start the signalling process, host A sends a setup message into the network (i.e. to switch 1)
 - The setup message contains (among other things) the complete destination address of B.
 - The setup message needs to get all the way to B to create the necessary connection state in every switch along the way
 - It is like sending a datagram to B where every switch knows which output to send the setup message so that it eventually reaches B
 - Assume that every switch knows the topology to figure out how to do that
 - When switch 1 receives the connection request, in addition to sending it on to switch 2, it creates a new entry in its VC table for this new connection
 - The entry is exactly the same shown in the previous table
 - Switch 1 picks the value 5 for this connection

Switching and Forwarding

- How does the signalling work (contd.)
 - When switch 2 receives the setup message, it performs the similar process and it picks the value 11 as the incoming VCI
 - Similarly switch 3 picks 7 as the value for its incoming VCI
 - Each switch can pick any number it likes, as long as that number is not currently in use for some other connection on that port of that switch
 - Finally the setup message arrives at host B.
 - Assuming that B is healthy and willing to accept a connection from host A, it allocates an incoming VCI value, in this case 4.
 - This VCI value can be used by B to identify all packets coming from A

Switching and Forwarding

- Now to complete the connection, everyone needs to be told what their downstream neighbor is using as the VCI for this connection
 - Host B sends an acknowledgement of the connection setup to switch 3 and includes in that message the VCI value that it chose (4)
 - Switch 3 completes the VC table entry for this connection and sends the acknowledgement on to switch 2 specifying the VCI of 7
 - Switch 2 completes the VC table entry for this connection and sends acknowledgement on to switch 1 specifying the VCI of 11
 - Finally switch 1 passes the acknowledgement on to host A telling it to use the VCI value of 5 for this connection

Switching and Forwarding

- When host A no longer wants to send data to host B, it tears down the connection by sending a teardown message to switch 1
- The switch 1 removes the relevant entry from its table and forwards the message on to the other switches in the path which similarly delete the appropriate table entries
- At this point, if host A were to send a packet with a VCI of 5 to switch 1, it would be dropped as if the connection had never existed

Switching and Forwarding

- Characteristics of VC
 - Since host A has to wait for the connection request to reach the far side of the network and return before it can send its first data packet, there is at least one RTT of delay before data is sent
 - While the connection request contains the full address for host B (which might be quite large, being a global identifier on the network), each data packet contains only a small identifier, which is only unique on one link.
 - Thus the per-packet overhead caused by the header is reduced relative to the datagram model
 - If a switch or a link in a connection fails, the connection is broken and a new one will need to be established.
 - Also the old one needs to be torn down to free up table storage space in the switches
 - The issue of how a switch decides which link to forward the connection request on has similarities with the function of a routing algorithm

Switching and Forwarding

- Good Properties of VC
 - By the time the host gets the go-ahead to send data, it knows quite a lot about the network-
 - For example, that there is really a route to the receiver and that the receiver is willing to receive data
 - It is also possible to allocate resources to the virtual circuit at the time it is established

Switching and Forwarding

- For example, an X.25 network – a packet-switched network that uses the connection-oriented model – employs the following three-part strategy
 - Buffers are allocated to each virtual circuit when the circuit is initialized
 - The sliding window protocol is run between each pair of nodes along the virtual circuit, and this protocol is augmented with the flow control to keep the sending node from overrunning the buffers allocated at the receiving node
 - The circuit is rejected by a given node if not enough buffers are available at that node when the connection request message is processed

Switching and Forwarding

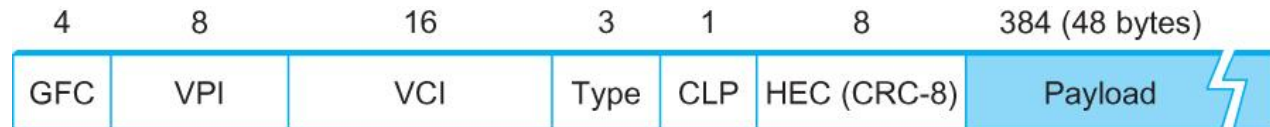
- Comparison with the Datagram Model
 - Datagram network has no connection establishment phase and each switch processes each packet independently
 - Each arriving packet competes with all other packets for buffer space
 - If there are no buffers, the incoming packet must be dropped
- In VC, we could imagine providing each circuit with a different quality of service (QoS)
 - The network gives the user some kind of performance related guarantee
 - Switches set aside the resources they need to meet this guarantee
 - For example, a percentage of each outgoing link's bandwidth
 - Delay tolerance on each switch
- Most popular examples of VC technologies are Frame Relay and ATM
 - One of the applications of Frame Relay is the construction of VPN

Switching and Forwarding

- ATM (Asynchronous Transfer Mode)
 - Connection-oriented packet-switched network
 - Packets are called cells
 - 5 byte header + 48 byte payload
 - Fixed length packets are easier to switch in hardware
 - Simpler to design
 - Enables parallelism

Switching and Forwarding

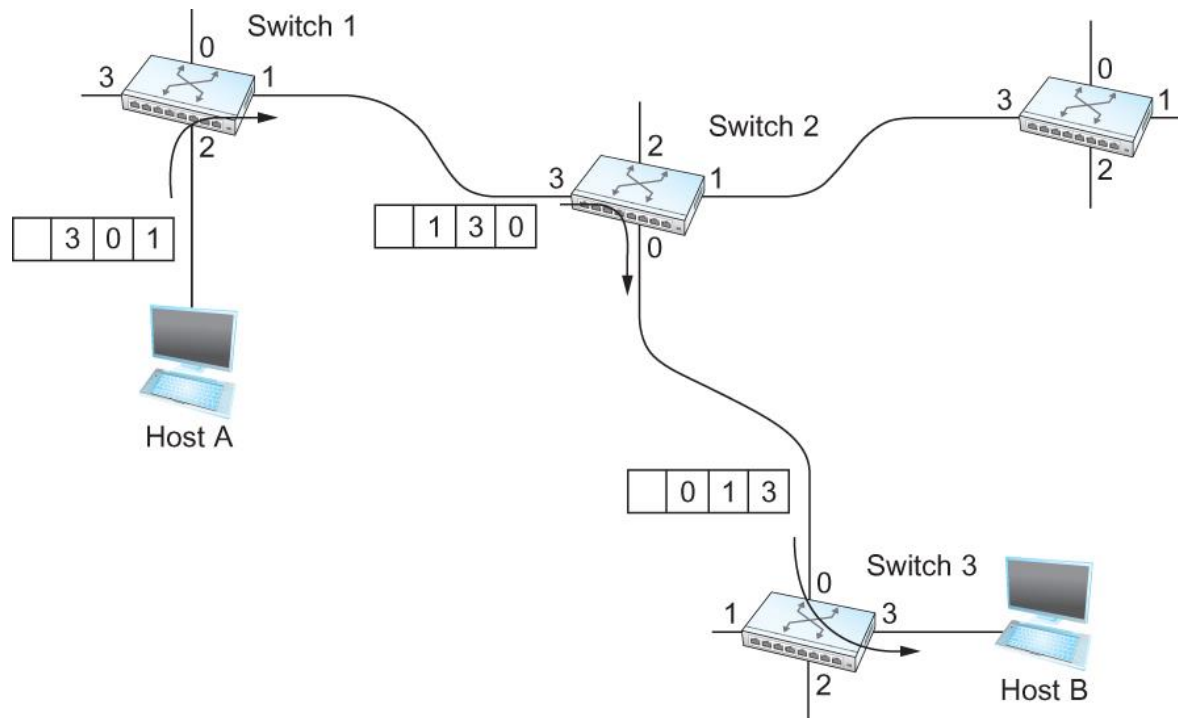
- ATM
 - User-Network Interface (UNI)
 - Host-to-switch format
 - GFC: Generic Flow Control
 - VCI: Virtual Circuit Identifier
 - Type: management, congestion control
 - CLP: Cell Loss Priority
 - HEC: Header Error Check (CRC-8)



- Network-Network Interface (NNI)
 - Switch-to-switch format
 - GFC becomes part of VPI field

Switching and Forwarding

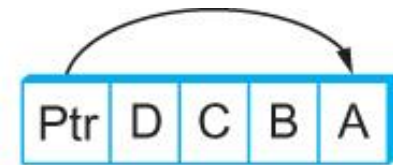
- Source Routing
 - All the information about network topology that is required to switch a packet across the network is provided by the source host



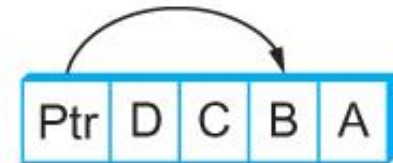
Switching and Forwarding

- Other approaches in Source Routing

Header entering
switch



Header leaving
switch



(a)

(b)

(c)

Bridges and LAN Switches

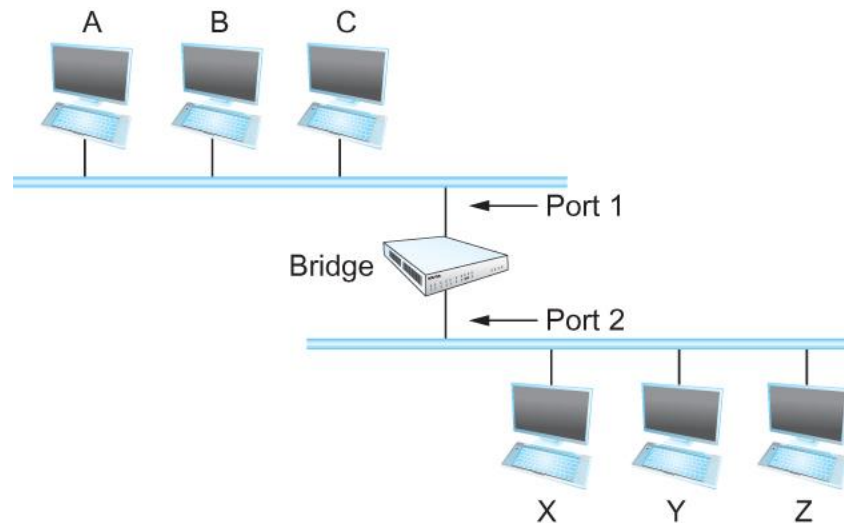
- Bridges and LAN Switches
 - Class of switches that is used to forward packets between shared-media LANs such as Ethernets
 - Known as LAN switches
 - Referred to as Bridges
 - Suppose you have a pair of Ethernets that you want to interconnect
 - One approach is put a repeater in between them
 - It might exceed the physical limitation of the Ethernet
 - No more than four repeaters between any pair of hosts
 - No more than a total of 2500 m in length is allowed
 - An alternative would be to put a node between the two Ethernets and have the node forward frames from one Ethernet to the other
 - This node is called a **Bridge**
 - A collection of LANs connected by one or more bridges is usually said to form an **Extended LAN**

Bridges and LAN Switches

- Simplest Strategy for Bridges
 - Accept LAN frames on their inputs and forward them out to all other outputs
 - Used by early bridges
- Learning Bridges
 - Observe that there is no need to forward all the frames that a bridge receives

Bridges and LAN Switches

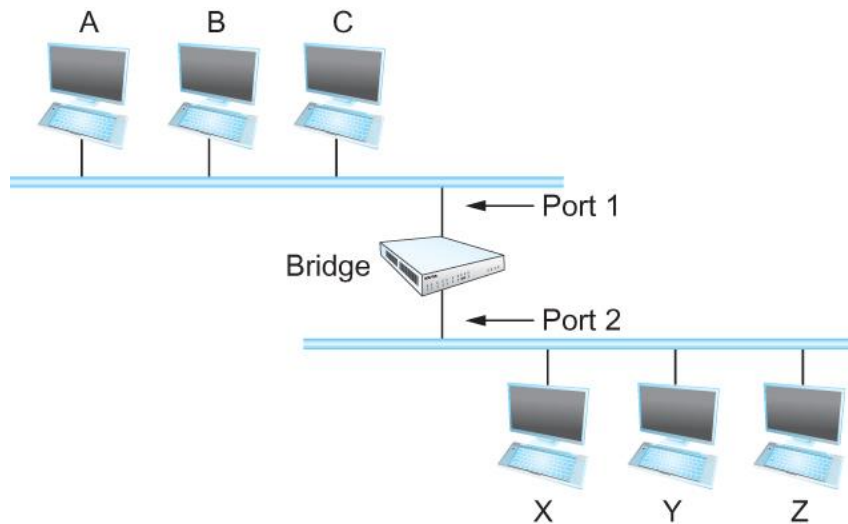
- Consider the following figure
 - When a frame from host A that is addressed to host B arrives on port 1, there is no need for the bridge to forward the frame out over port 2.



- How does a bridge come to learn on which port the various hosts reside?

Bridges and LAN Switches

- Solution
 - Download a table into the bridge



Host	Port

A	1
B	1
C	1
X	2
Y	2
Z	2

- Who does the download?
 - Human
 - Too much work for maintenance

Bridges and LAN Switches

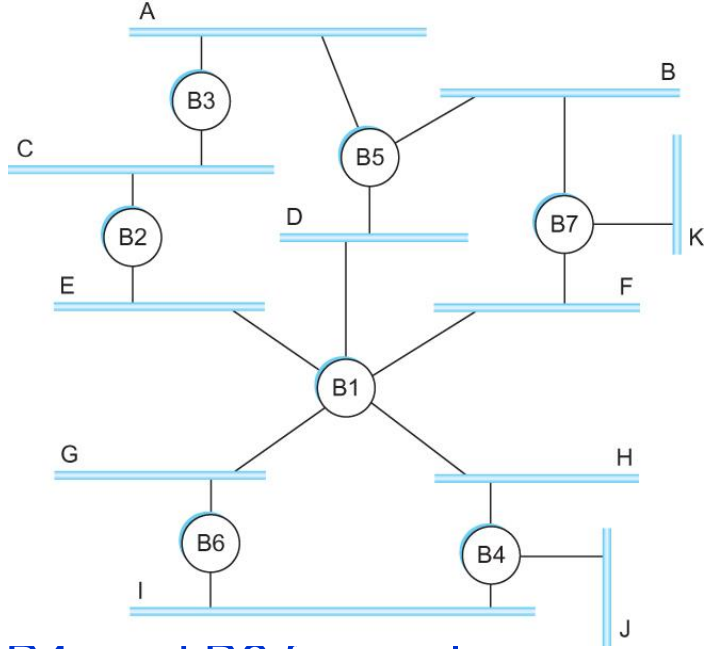
- Can the bridge learn this information by itself?
 - Yes

- How
 - Each bridge inspects the source address in all the frames it receives
 - Record the information at the bridge and build the table
 - When a bridge first boots, this table is empty
 - Entries are added over time
 - A timeout is associated with each entry
 - The bridge discards the entry after a specified period of time
 - To protect against the situation in which a host is moved from one network to another

- If the bridge receives a frame that is addressed to host not currently in the table
 - Forward the frame out on all other ports

Bridges and LAN Switches

- Strategy works fine if the extended LAN does not have a loop in it
- Why?
 - Frames potentially loop through the extended LAN forever



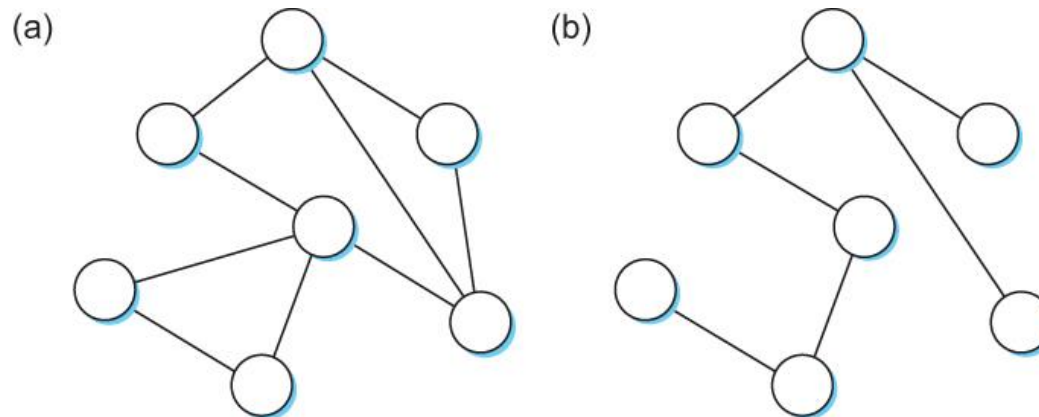
- Bridges B1, B4, and B6 form a loop

Bridges and LAN Switches

- How does an extended LAN come to have a loop in it?
 - Network is managed by more than one administrator
 - For example, it spans multiple departments in an organization
 - It is possible that no single person knows the entire configuration of the network
 - A bridge that closes a loop might be added without anyone knowing
 - Loops are built into the network to provide redundancy in case of failures
- Solution
 - Distributed Spanning Tree Algorithm

Spanning Tree Algorithm

- Think of the extended LAN as being represented by a graph that possibly has loops (cycles)
- A spanning tree is a sub-graph of this graph that covers all the vertices but contains no cycles
 - Spanning tree keeps all the vertices of the original graph but throws out some of the edges



Example of (a) a cyclic graph; (b) a corresponding spanning tree.

Spanning Tree Algorithm

- Developed by Radia Perlman at Digital
 - A protocol used by a set of bridges to agree upon a spanning tree for a particular extended LAN
 - IEEE 802.1 specification for LAN bridges is based on this algorithm
- Each bridge decides the ports over which it is and is not willing to forward frames
 - In a sense, it is by removing ports from the topology that the extended LAN is reduced to an acyclic tree
 - It is even possible that an entire bridge will not participate in forwarding frames

Spanning Tree Algorithm

- Algorithm is dynamic
 - The bridges are always prepared to reconfigure themselves into a new spanning tree if some bridges fail
- Main idea
 - Each bridge selects the ports over which they will forward the frames

Spanning Tree Algorithm

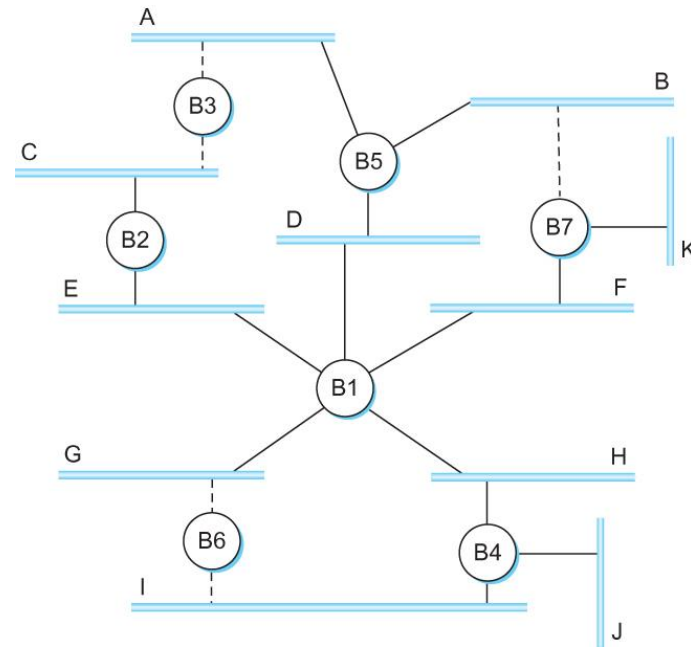
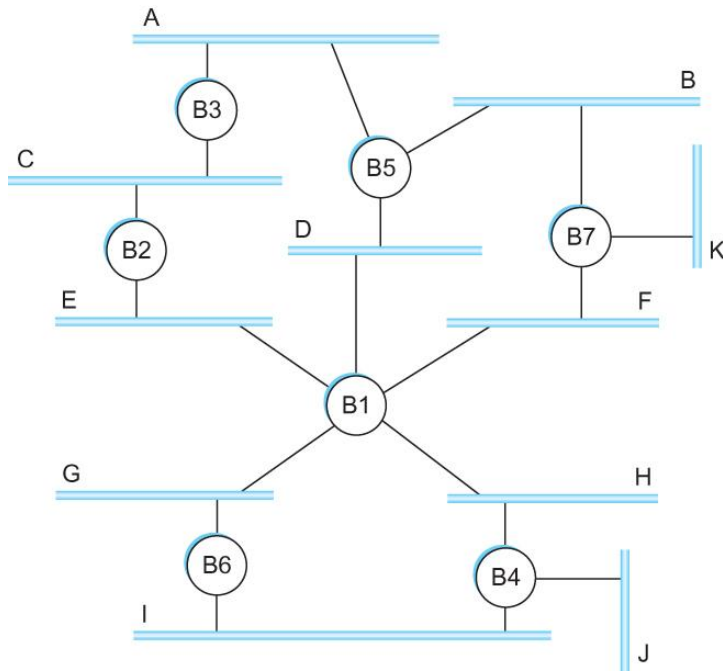
- Algorithm selects ports as follows:
 - Each bridge has a unique identifier
 - B1, B2, B3,...and so on.
 - Elect the bridge with the smallest id as the root of the spanning tree
 - The root bridge always forwards frames out over all of its ports
 - Each bridge computes the shortest path to the root and notes which of its ports is on this path
 - This port is selected as the bridge's preferred path to the root
 - Finally, all the bridges connected to a given LAN elect a single *designated bridge* that will be responsible for forwarding frames toward the root bridge

Spanning Tree Algorithm

- Each LAN's designated bridge is the one that is closest to the root
- If two or more bridges are equally close to the root,
 - Then select bridge with the smallest id
- Each bridge is connected to more than one LAN
 - So it participates in the election of a designated bridge for each LAN it is connected to.
 - Each bridge decides if it is the designated bridge relative to each of its ports
 - The bridge forwards frames over those ports for which it is the designated bridge

Spanning Tree Algorithm

- B1 is the root bridge
- B3 and B5 are connected to LAN A, but B5 is the designated bridge
- B5 and B7 are connected to LAN B, but B5 is the designated bridge



Spanning Tree Algorithm

- Initially each bridge thinks it is the root, so it sends a configuration message on each of its ports identifying itself as the root and giving a distance to the root of 0
- Upon receiving a configuration message over a particular port, the bridge checks to see if the new message is *better* than the current best configuration message recorded for that port
- The new configuration is better than the currently recorded information if
 - It identifies a root with a smaller id or
 - It identifies a root with an equal id but with a shorter distance or
 - The root id and distance are equal, but the sending bridge has a smaller id

Spanning Tree Algorithm

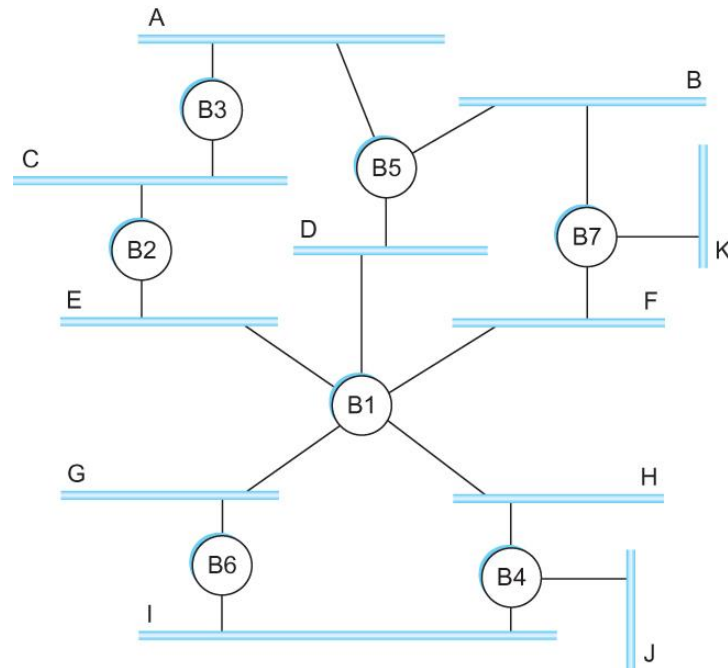
- If the new message is better than the currently recorded one,
 - The bridge discards the old information and saves the new information
 - It first adds 1 to the distance-to-root field
- When a bridge receives a configuration message indicating that it is not the root bridge (that is, a message from a bridge with smaller id)
 - The bridge stops generating configuration messages on its own
 - Only forwards configuration messages from other bridges after 1 adding to the distance field

Spanning Tree Algorithm

- When a bridge receives a configuration message that indicates it is not the designated bridge for that port
 - => a message from a bridge that is closer to the root or equally far from the root but with a smaller id
 - The bridge stops sending configuration messages over that port
- When the system stabilizes,
 - Only the root bridge is still generating configuration messages.
 - Other bridges are forwarding these messages only over ports for which they are the designated bridge

Spanning Tree Algorithm

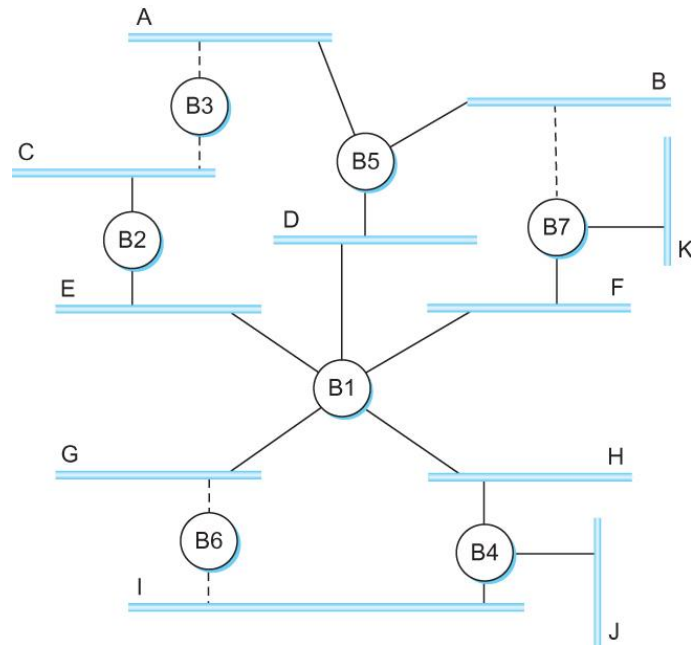
- Consider the situation when the power had just been restored to the building housing the following network



- All bridges would start off by claiming to be the root

Spanning Tree Algorithm

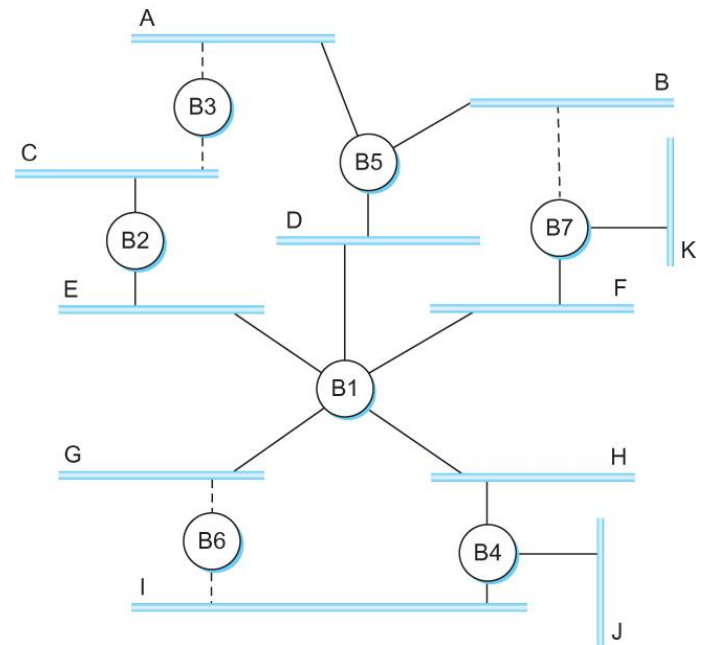
- Denote a configuration message from node X in which it claims to be distance d from the root node Y as (Y, d, X)



- Consider the activity at node B3

Spanning Tree Algorithm

- B3 receives (B2, 0, B2)
- Since $2 < 3$, B3 accepts B2 as root
- B3 adds 1 to the distance advertised by B2 and sends (B2, 1, B3) to B5
- Meanwhile B2 accepts B1 as root because it has the lower id and it sends (B1, 1, B2) toward B3
- B5 accepts B1 as root and sends (B1, 1, B5) to B3
- B3 accepts B1 as root and it notes that both B2 and B5 are closer to the root than it is.
 - Thus B3 stops forwarding messages on both its interfaces
 - This leaves B3 with both ports not selected



Spanning Tree Algorithm

- Even after the system has stabilized, the root bridge continues to send configuration messages periodically
 - Other bridges continue to forward these messages
- When a bridge fails, the downstream bridges will not receive the configuration messages
- After waiting a specified period of time, they will once again claim to be the root and the algorithm starts again
- Note
 - Although the algorithm is able to reconfigure the spanning tree whenever a bridge fails, it is not able to forward frames over alternative paths for the sake of routing around a congested bridge

Spanning Tree Algorithm

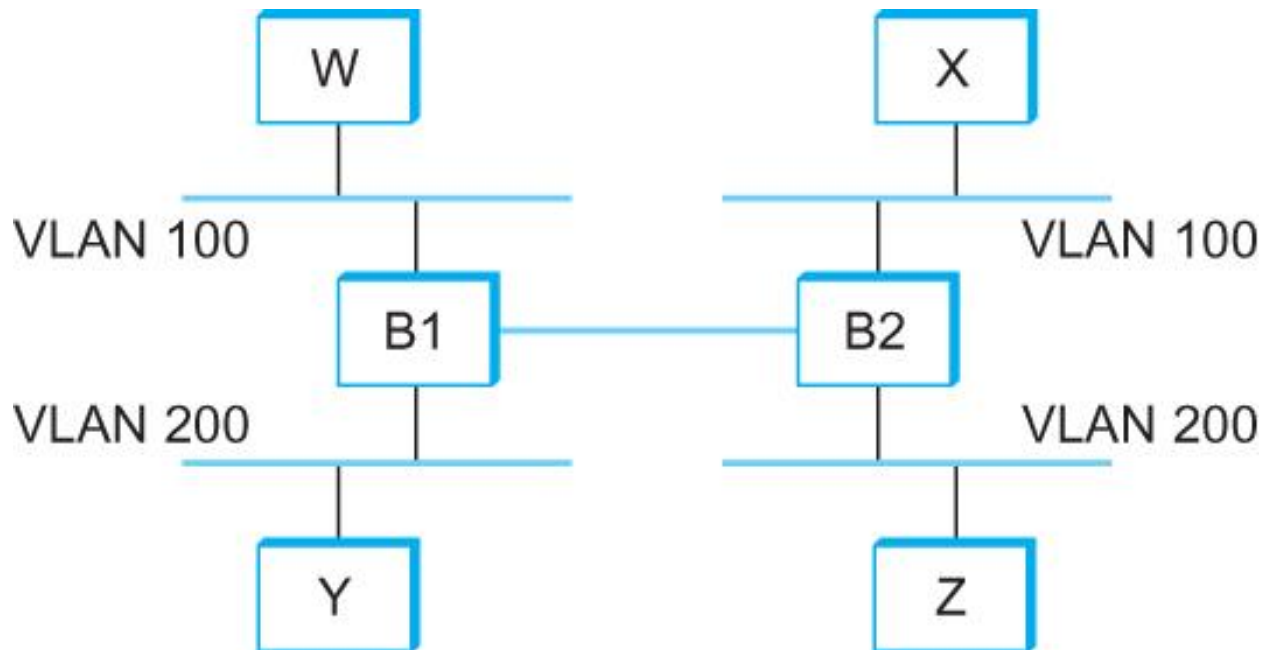
- Broadcast and Multicast
 - Forward all broadcast/multicast frames
 - Current practice
 - Learn when no group members downstream
 - Accomplished by having each member of group G send a frame to bridge multicast address with G in source field

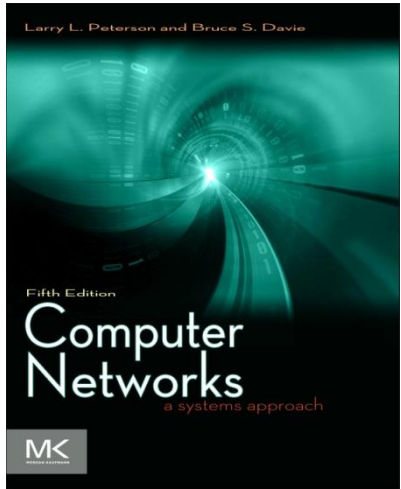
Spanning Tree Algorithm

- Limitation of Bridges
 - Do not scale
 - Spanning tree algorithm does not scale
 - Broadcast does not scale
 - Do not accommodate heterogeneity

Spanning Tree Algorithm

- Virtual LAN





UNIT 3

ROUTING AND ADDRESSING SCHEMES

Problems

- In Chapter 2 we saw how to connect one node to another, or to an existing network. How do we build networks of global scale?
- How do we interconnect different types of networks to build a large global network?

Chapter Outline

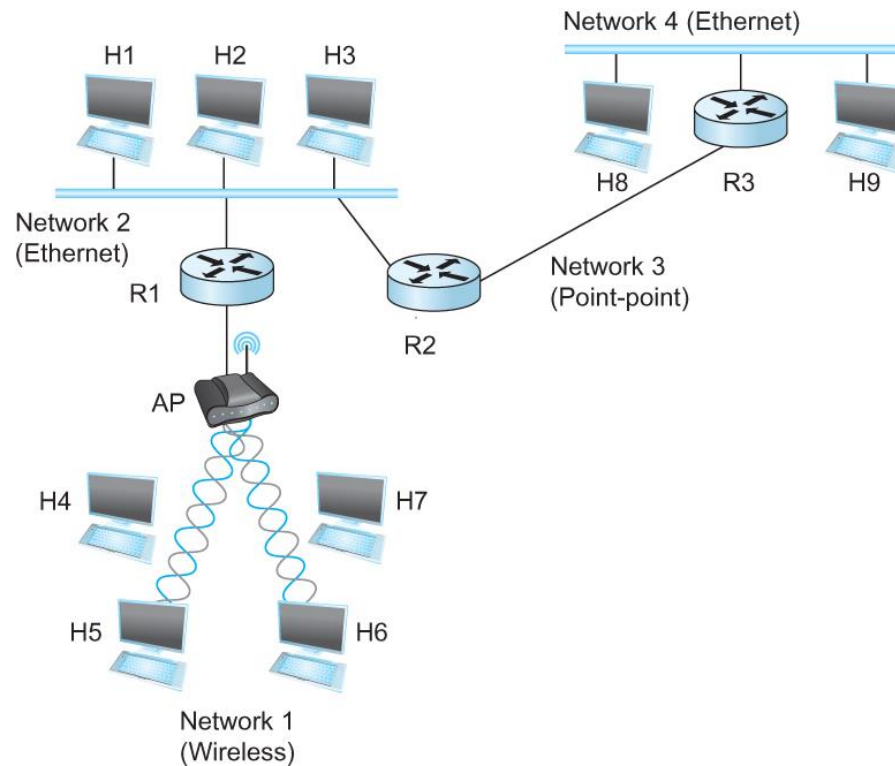
- Basic Internetworking (IP)
- Routing
- Global Internet
- Multicast

Chapter Goal

- Understanding the functions of switches, bridges and routers
- Discussing Internet Protocol (IP) for interconnecting networks
- Understanding the concept of routing

Internetworking

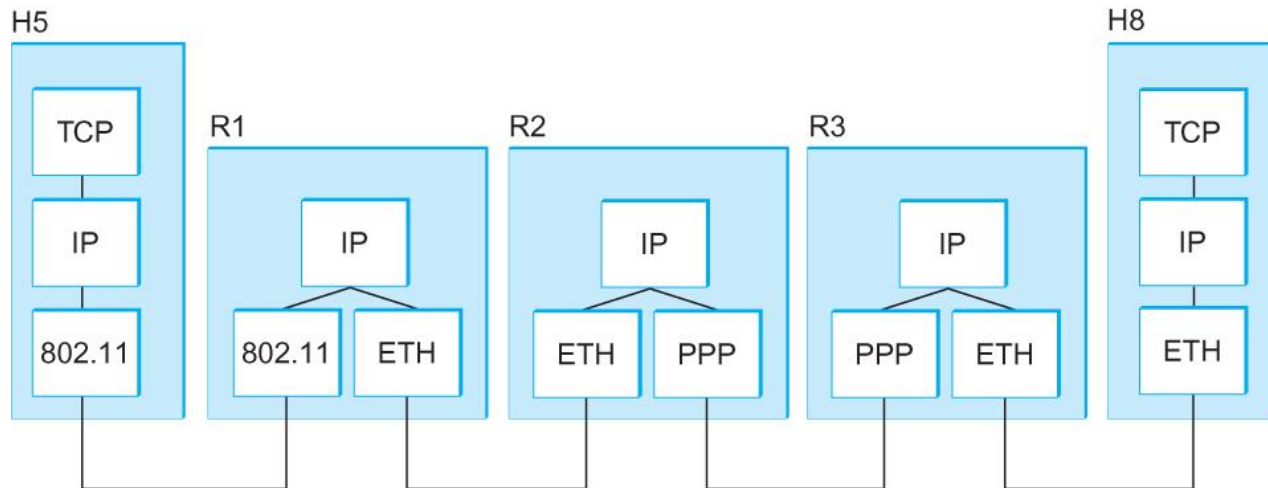
- What is internetwork
 - An arbitrary collection of networks interconnected to provide some sort of host-host to packet delivery service



A simple internetwork where H represents hosts and R represents routers

Internetworking

- What is IP
 - IP stands for Internet Protocol
 - Key tool used today to build scalable, heterogeneous internetworks
 - It runs on all the nodes in a collection of networks and defines the infrastructure that allows these nodes and networks to function as a single logical internetwork



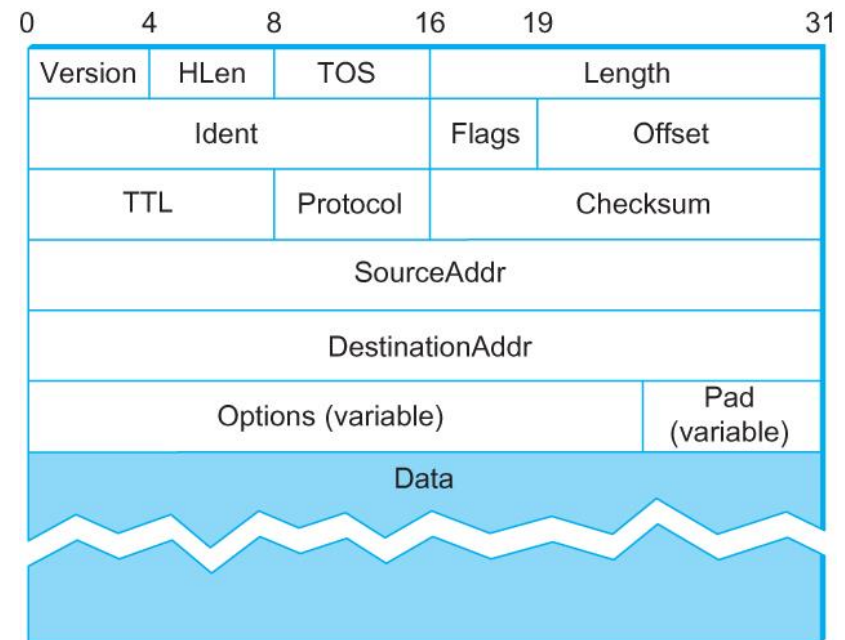
A simple internetwork showing the protocol layers

IP Service Model

- Packet Delivery Model
 - Connectionless model for data delivery
 - Best-effort delivery (unreliable service)
 - packets are lost
 - packets are delivered out of order
 - duplicate copies of a packet are delivered
 - packets can be delayed for a long time
- Global Addressing Scheme
 - Provides a way to identify all hosts in the network

Packet Format

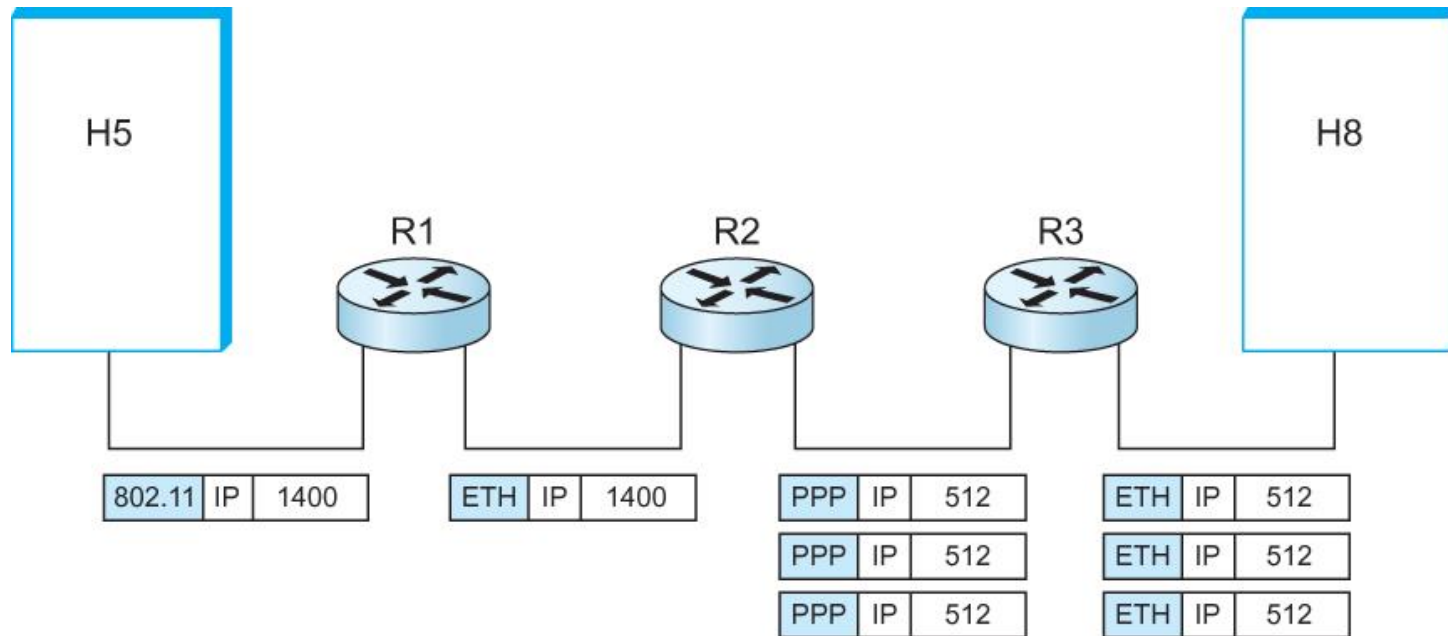
- Version (4): currently 4
- HLen (4): number of 32-bit words in header
- TOS (8): type of service (not widely used)
- Length (16): number of bytes in this datagram
- Ident (16): used by fragmentation
- Flags/Offset (16): used by fragmentation
- TTL (8): number of hops this datagram has traveled
- Protocol (8): demux key (TCP=6, UDP=17)
- Checksum (16): of the header only
- DestAddr & SrcAddr (32)



IP Fragmentation and Reassembly

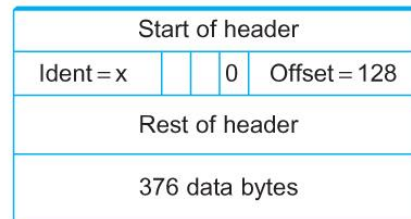
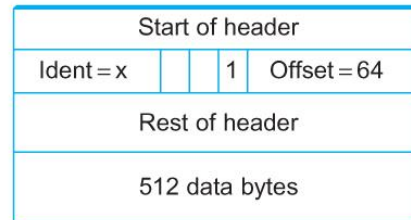
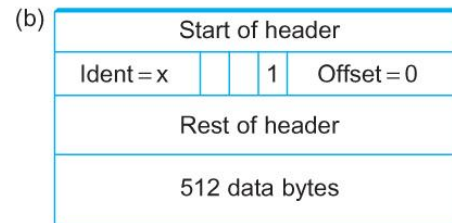
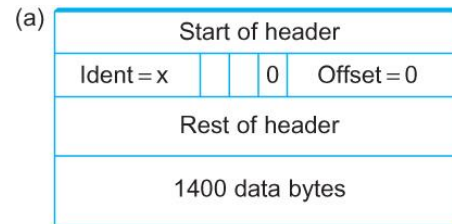
- Each network has some MTU (Maximum Transmission Unit)
 - Ethernet (1500 bytes), FDDI (4500 bytes)
- Strategy
 - Fragmentation occurs in a router when it receives a datagram that it wants to forward over a network which has (MTU < datagram)
 - Reassembly is done at the receiving host
 - All the fragments carry the same identifier in the *Ident* field
 - Fragments are self-contained datagrams
 - IP does not recover from missing fragments

IP Fragmentation and Reassembly



IP datagrams traversing the sequence of physical networks

IP Fragmentation and Reassembly



Header fields used in IP fragmentation. (a) Unfragmented packet; (b) fragmented packets.

Global Addresses

- Properties
 - globally unique
 - hierarchical: network + host
 - 4 Billion IP address, half are A type, $\frac{1}{4}$ is B type, and $\frac{1}{8}$ is C type

- Format



- Dot notation

- 10.3.2.4
- 128.96.33.81
- 192.12.69.77

IP Datagram Forwarding

- Strategy
 - every datagram contains destination's address
 - if directly connected to destination network, then forward to host
 - if not directly connected to destination network, then forward to some router
 - forwarding table maps network number into next hop
 - each host has a default router
 - each router maintains a forwarding table
- Example (router R2)

NetworkNum	NextHop
1	R1
2	Interface 1
3	Interface 0
4	R3

IP Datagram Forwarding

■ Algorithm

```
if (NetworkNum of destination = NetworkNum of one of my
    interfaces) then
    deliver packet to destination over that interface
else
    if (NetworkNum of destination is in my forwarding table)
    then
        deliver packet to NextHop router
    else
        deliver packet to default router
```

For a host with only one interface and only a default router in its forwarding table, this simplifies to

```
if (NetworkNum of destination = my NetworkNum) then
    deliver packet to destination directly
else
    deliver packet to default router
```


Subnetting

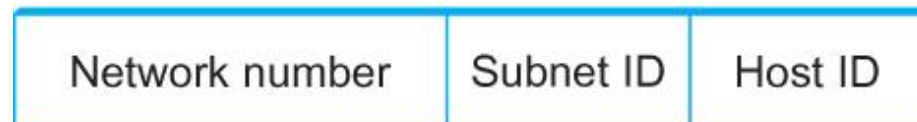
- Add another level to address/routing hierarchy: *subnet*
- *Subnet masks* define variable partition of host part of class A and B addresses
- Subnets visible only within site



Class B address

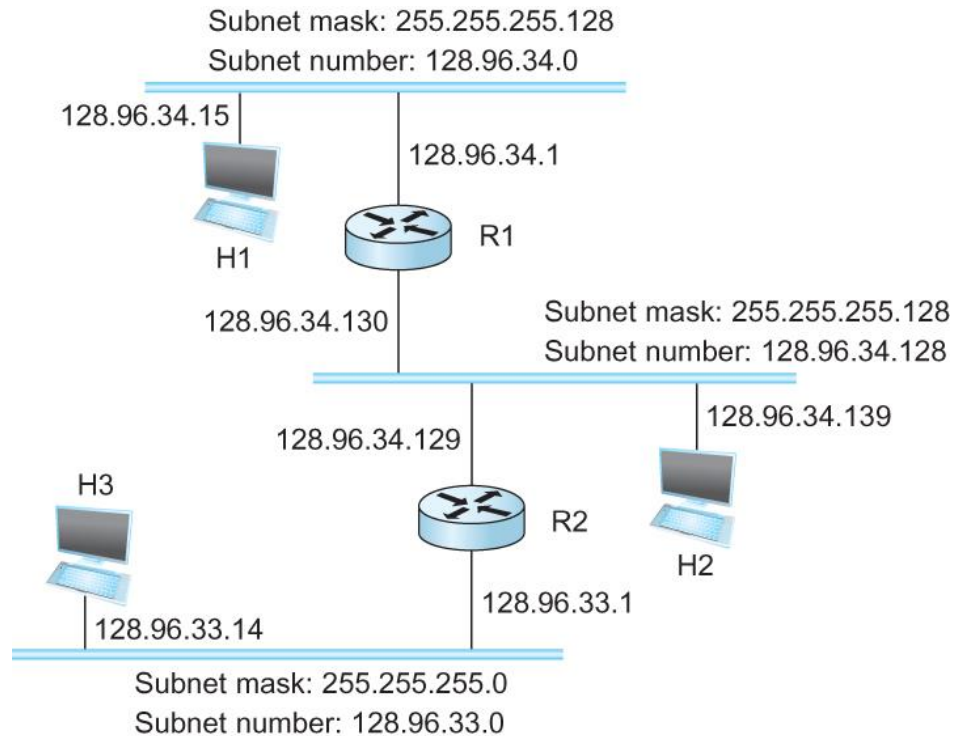


Subnet mask (255.255.255.0)



Subnetted address

Subnetting



■ Forwarding Table at Router R1

SubnetNumber	SubnetMask	NextHop
128.96.34.0	255.255.255.128	Interface 0
128.96.34.128	255.255.255.128	Interface 1
128.96.33.0	255.255.255.0	R2

Subnetting

Forwarding Algorithm

```
D = destination IP address
for each entry < SubnetNum, SubnetMask, NextHop >
  D1 = SubnetMask & D
  if D1 = SubnetNum
    if NextHop is an interface
      deliver datagram directly to destination
    else
      deliver datagram to NextHop (a router)
```

Subnetting

Notes

- Would use a default router if nothing matches
- Not necessary for all ones in subnet mask to be contiguous
- Can put multiple subnets on one physical network
- Subnets not visible from the rest of the Internet

Classless Addressing

- Classless Inter-Domain Routing
 - A technique that addresses two scaling concerns in the Internet
 - The growth of backbone routing table as more and more network numbers need to be stored in them
 - Potential exhaustion of the 32-bit address space
 - Address assignment efficiency
 - Arises because of the IP address structure with class A, B, and C addresses
 - Forces us to hand out network address space in fixed-size chunks of three very different sizes
 - A network with two hosts needs a class C address
 - Address assignment efficiency = $2/255 = 0.78$
 - A network with 256 hosts needs a class B address
 - Address assignment efficiency = $256/65535 = 0.39$

Classless Addressing

- Exhaustion of IP address space centers on exhaustion of the class B network numbers
- Solution
 - Say “NO” to any Autonomous System (AS) that requests a class B address unless they can show a need for something close to 64K addresses
 - Instead give them an appropriate number of class C addresses
 - For any AS with at least 256 hosts, we can guarantee an address space utilization of at least 50%
- What is the problem with this solution?

Classless Addressing

- Problem with this solution
 - Excessive storage requirement at the routers.
- If a single AS has, say 16 class C network numbers assigned to it,
 - Every Internet backbone router needs 16 entries in its routing tables for that AS
 - This is true, even if the path to every one of these networks is the same
- If we had assigned a class B address to the AS
 - The same routing information can be stored in one entry
 - Efficiency = $16 \times 255 / 65,536 = 6.2\%$

Classless Addressing

- CIDR tries to balance the desire to minimize the number of routes that a router needs to know against the need to hand out addresses efficiently.
- CIDR uses aggregate routes
 - Uses a single entry in the forwarding table to tell the router how to reach a lot of different networks
 - Breaks the rigid boundaries between address classes

Classless Addressing

- Consider an AS with 16 class C network numbers.
- Instead of handing out 16 addresses at random, hand out a block of contiguous class C addresses
- Suppose we assign the class C network numbers from 192.4.16 through 192.4.31
- Observe that top 20 bits of all the addresses in this range are the same (1 1000000 00000100 0001)
 - We have created a 20-bit network number (which is in between class B network number and class C number)
- Requires to hand out blocks of class C addresses that share a common prefix

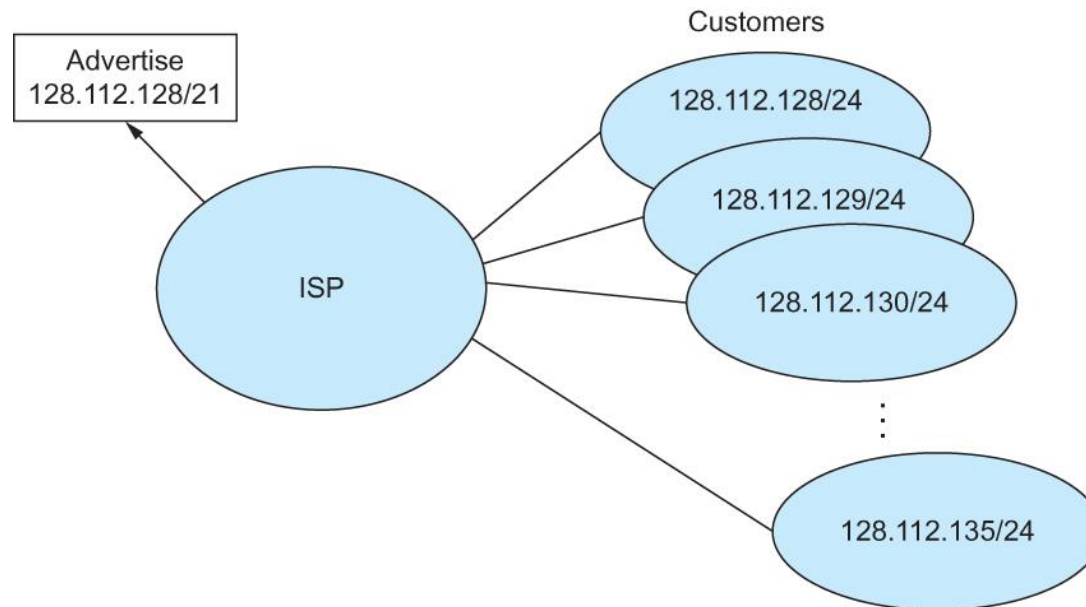
Classless Addressing

- Requires to hand out blocks of class C addresses that share a common prefix
- The convention is to place a /X after the prefix where X is the prefix length in bits
- For example, the 20-bit prefix for all the networks 192.4.16 through 192.4.31 is represented as 192.4.16/20
- By contrast, if we wanted to represent a single class C network number, which is 24 bits long, we would write it 192.4.16/24

Classless Addressing

- How do the routing protocols handle this classless addresses
 - It must understand that the network number may be of any length
- Represent network number with a single pair
<length, value>
- All routers must understand CIDR addressing

Classless Addressing



Route aggregation with CIDR

IP Forwarding Revisited

- IP forwarding mechanism assumes that it can find the network number in a packet and then look up that number in the forwarding table
- We need to change this assumption in case of CIDR
- CIDR means that prefixes may be of any length, from 2 to 32 bits

IP Forwarding Revisited

- It is also possible to have prefixes in the forwarding tables that overlap
 - Some addresses may match more than one prefix
- For example, we might find both 171.69 (a 16 bit prefix) and 171.69.10 (a 24 bit prefix) in the forwarding table of a single router
- A packet destined to 171.69.10.5 clearly matches both prefixes.
 - The rule is based on the principle of “longest match”
 - 171.69.10 in this case
- A packet destined to 171.69.20.5 would match 171.69 and not 171.69.10

Address Translation Protocol (ARP)

- Map IP addresses into physical addresses
 - destination host
 - next hop router
- Techniques
 - encode physical address in host part of IP address
 - table-based
- ARP (Address Resolution Protocol)
 - table of IP to physical address bindings
 - broadcast request if IP address not in table
 - target machine responds with its physical address
 - table entries are discarded if not refreshed

ARP Packet Format

0	8	16	31
Hardware type = 1		ProtocolType = 0x0800	
HLen = 48	PLen = 32	Operation	
SourceHardwareAddr (bytes 0–3)			
SourceHardwareAddr (bytes 4–5)		SourceProtocolAddr (bytes 0–1)	
SourceProtocolAddr (bytes 2–3)		TargetHardwareAddr (bytes 0–1)	
TargetHardwareAddr (bytes 2–5)			
TargetProtocolAddr (bytes 0–3)			

- HardwareType: type of physical network (e.g., Ethernet)
- ProtocolType: type of higher layer protocol (e.g., IP)
- HLEN & PLEN: length of physical and protocol addresses
- Operation: request or response
- Source/Target Physical/Protocol addresses

Host Configurations

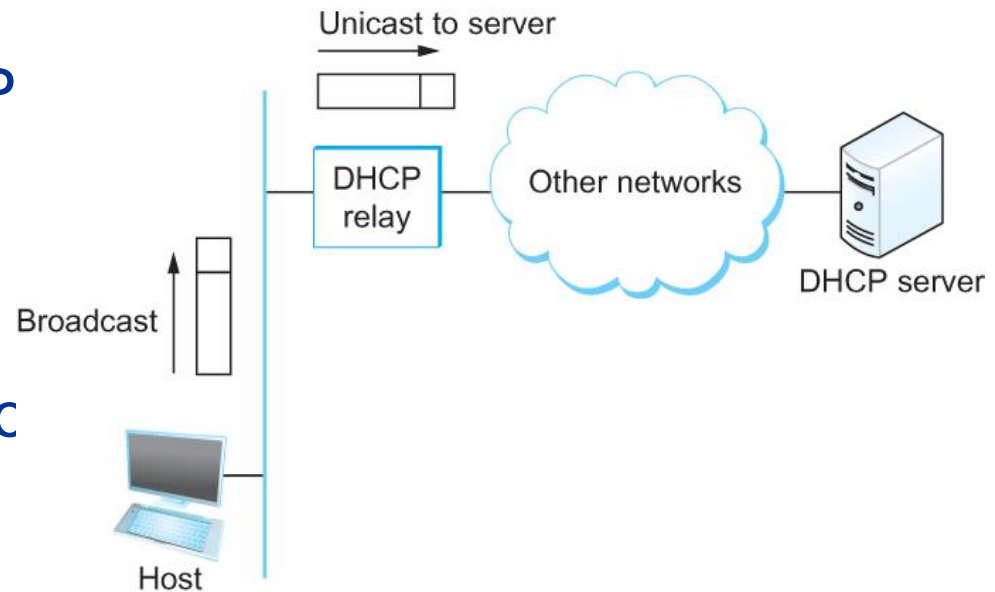
- Notes
 - Ethernet addresses are configured into network by manufacturer and they are unique
 - IP addresses must be unique on a given internetwork but also must reflect the structure of the internetwork
 - Most host Operating Systems provide a way to manually configure the IP information for the host
 - Drawbacks of manual configuration
 - A lot of work to configure all the hosts in a large network
 - Configuration process is error-prone
 - Automated Configuration Process is required

Dynamic Host Configuration Protocol (DHCP)

- DHCP server is responsible for providing configuration information to hosts
- There is at least one DHCP server for an administrative domain
- DHCP server maintains a pool of available addresses

DHCP

- Newly booted or attached host sends DHCPDISCOVER message to a special IP address (255.255.255.255)
- DHCP relay agent unicasts the message to DHCP server and waits for the response



Internet Control Message Protocol (ICMP)

- Defines a collection of error messages that are sent back to the source host whenever a router or host is unable to process an IP datagram successfully
 - Destination host unreachable due to link /node failure
 - Reassembly process failed
 - TTL had reached 0 (so datagrams don't cycle forever)
 - IP header checksum failed
- ICMP-Redirect
 - From router to a source host
 - With a better route information

Internet Control Message Protocol (ICMP)

- Defines a collection of error messages that are sent back to the source host whenever a router or host is unable to process an IP datagram successfully
 - Destination host unreachable due to link /node failure
 - Reassembly process failed
 - TTL had reached 0 (so datagrams don't cycle forever)
 - IP header checksum failed
- ICMP-Redirect
 - From router to a source host
 - With a better route information

Routing

Forwarding versus Routing

- Forwarding:
 - to select an output port based on destination address and routing table
- Routing:
 - process by which routing table is built

Routing

- Forwarding table VS Routing table
 - Forwarding table
 - Used when a packet is being forwarded and so must contain enough information to accomplish the forwarding function
 - A row in the forwarding table contains the mapping from a network number to an outgoing interface and some MAC information, such as Ethernet Address of the next hop
 - Routing table
 - Built by the routing algorithm as a precursor to build the forwarding table
 - Generally contains mapping from network numbers to next hops

Routing

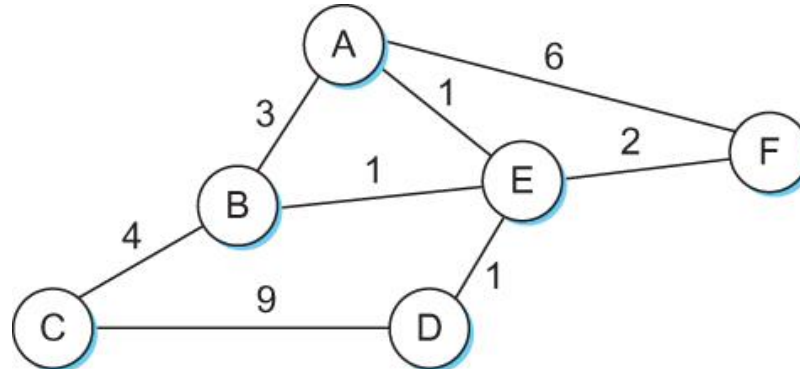
(a)		
Prefix/Length	Next Hop	
18/8	171.69.245.10	

(b)		
Prefix/Length	Interface	MAC Address
18/8	if0	8:0:2b:e4:b:1:2

Example rows from (a) routing and (b) forwarding tables

Routing

- Network as a Graph



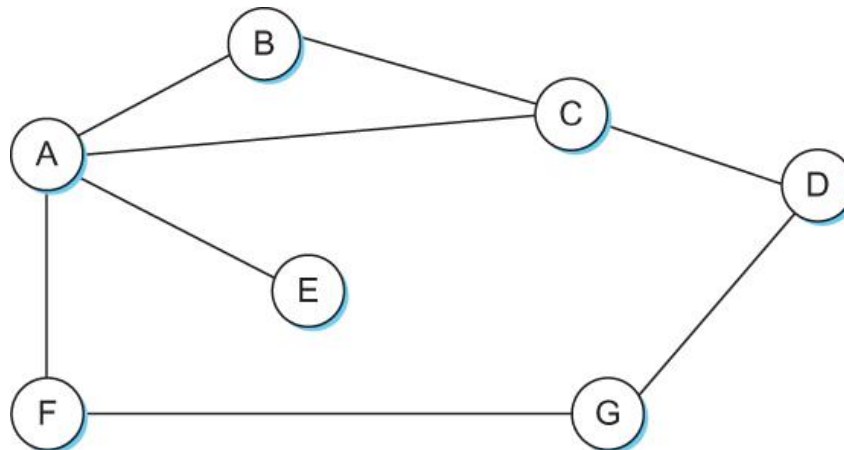
- The basic problem of routing is to find the lowest-cost path between any two nodes
 - Where the cost of a path equals the sum of the costs of all the edges that make up the path

Routing

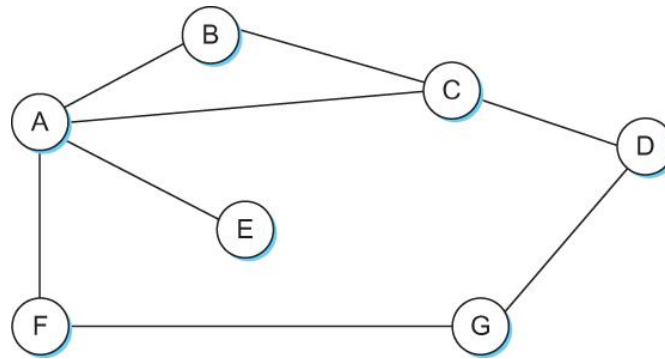
- For a simple network, we can calculate all shortest paths and load them into some nonvolatile storage on each node.
- Such a static approach has several shortcomings
 - It does not deal with node or link failures
 - It does not consider the addition of new nodes or links
 - It implies that edge costs cannot change
- What is the solution?
 - Need a distributed and dynamic protocol
 - Two main classes of protocols
 - Distance Vector
 - Link State

Distance Vector

- Each node constructs a one dimensional array (a vector) containing the “distances” (costs) to all other nodes and distributes that vector to its immediate neighbors
- Starting assumption is that each node knows the cost of the link to each of its directly connected neighbors



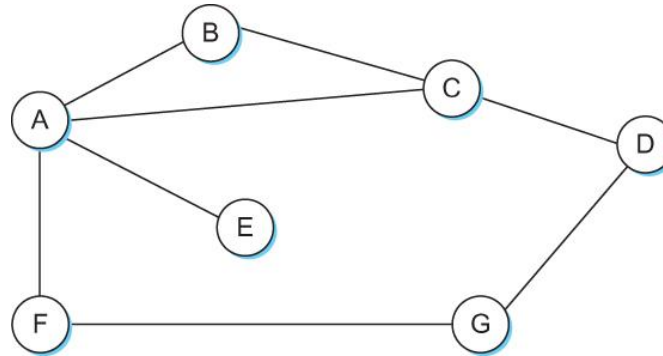
Distance Vector



Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

Initial distances stored at each node (global view)

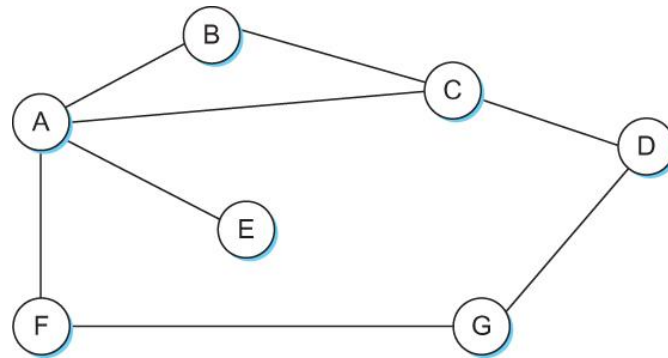
Distance Vector



Destination	Cost	NextHop
B	1	B
C	1	C
D	∞	—
E	1	E
F	1	F
G	∞	—

Initial routing table at node A

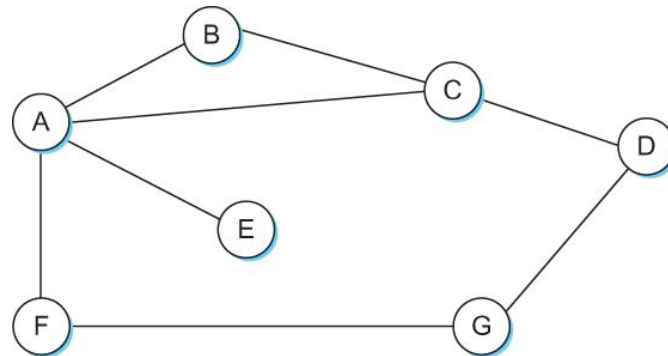
Distance Vector



Destination	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

Final routing table at node A

Distance Vector



Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

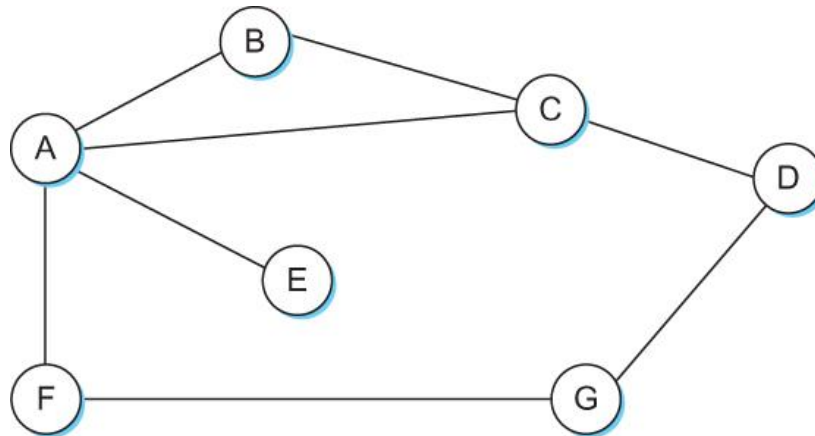
Final distances stored at each node (global view)

Distance Vector

- The distance vector routing algorithm is sometimes called as Bellman-Ford algorithm
- Every T seconds each router sends its table to its neighbor each each router then updates its table based on the new information
- Problems include fast response to good news and slow response to bad news. Also too many messages to update

Distance Vector

- When a node detects a link failure
 - F detects that link to G has failed
 - F sets distance to G to infinity and sends update to A
 - A sets distance to G to infinity since it uses F to reach G
 - A receives periodic update from C with 2-hop path to G
 - A sets distance to G to 3 and sends update to F
 - F decides it can reach G in 4 hops via A



Distance Vector

- Slightly different circumstances can prevent the network from stabilizing
 - Suppose the link from A to E goes down
 - In the next round of updates, A advertises a distance of infinity to E, but B and C advertise a distance of 2 to E
 - Depending on the exact timing of events, the following might happen
 - Node B, upon hearing that E can be reached in 2 hops from C, concludes that it can reach E in 3 hops and advertises this to A
 - Node A concludes that it can reach E in 4 hops and advertises this to C
 - Node C concludes that it can reach E in 5 hops; and so on.
 - This cycle stops only when the distances reach some number that is large enough to be considered infinite
 - **Count-to-infinity problem**

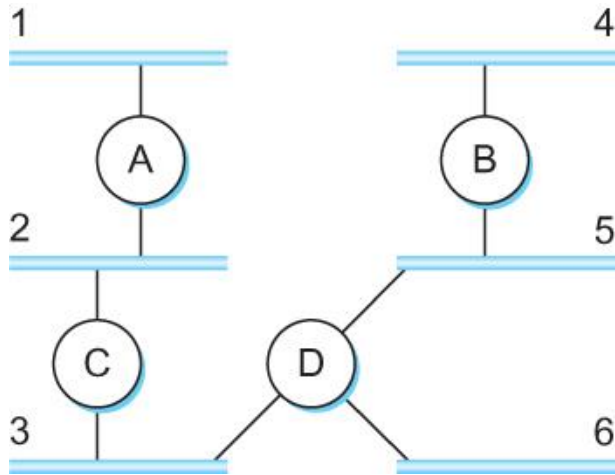
Count-to-infinity Problem

- Use some relatively small number as an approximation of infinity
- For example, the maximum number of hops to get across a certain network is never going to be more than 16
- One technique to improve the time to stabilize routing is called *split horizon*
 - When a node sends a routing update to its neighbors, it does not send those routes it learned from each neighbor back to that neighbor
 - For example, if B has the route (E, 2, A) in its table, then it knows it must have learned this route from A, and so whenever B sends a routing update to A, it does not include the route (E, 2) in that update

Count-to-infinity Problem

- In a stronger version of split horizon, called *split horizon with poison reverse*
 - B actually sends that back route to A, but it puts negative information in the route to ensure that A will not eventually use B to get to E
 - For example, B sends the route (E, ∞) to A

Routing Information Protocol (RIP)



Example Network running RIP

0	8	16	31
Command		Version	
Must be zero		Route Tags	
Family of net 1			
Address prefix of net 1			
Mask of net 1			
Distance to net 1			
Family of net 2		Route Tags	
Address prefix of net 2			
Mask of net 2			
Distance to net 2			

RIPv2 Packet Format

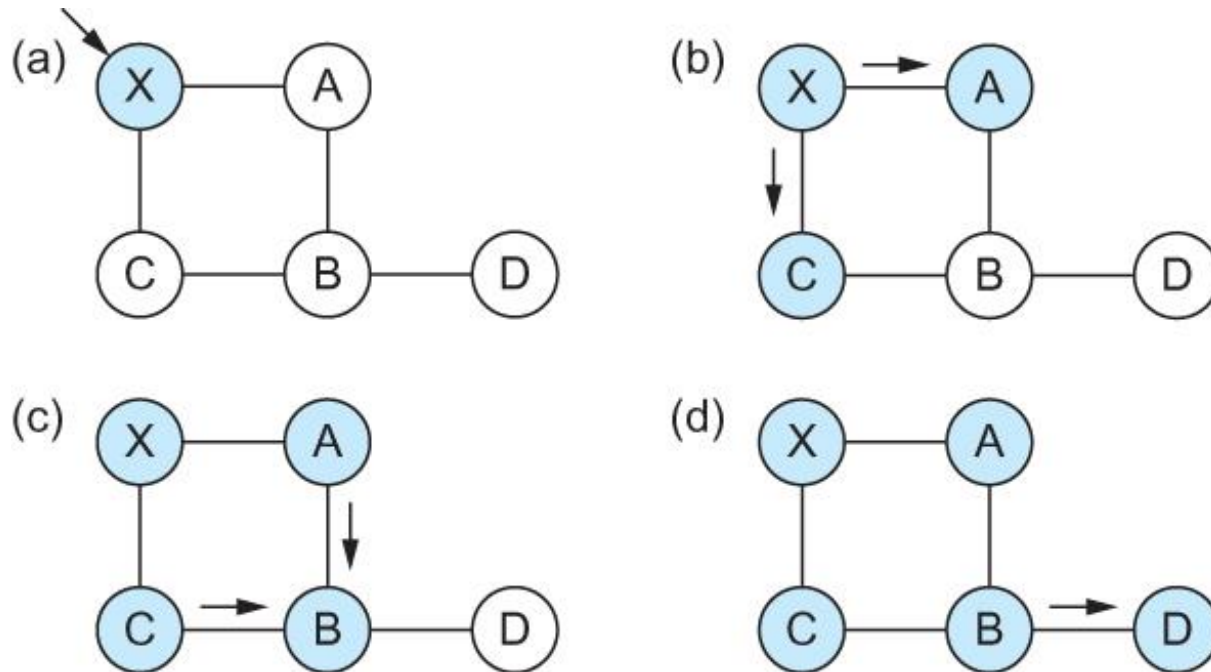
Link State Routing

Strategy: Send to all nodes (not just neighbors) information about directly connected links (not entire routing table).

- Link State Packet (LSP)
 - id of the node that created the LSP
 - cost of link to each directly connected neighbor
 - sequence number (SEQNO)
 - time-to-live (TTL) for this packet
- Reliable Flooding
 - store most recent LSP from each node
 - forward LSP to all nodes but one that sent it
 - generate new LSP periodically; increment SEQNO
 - start SEQNO at 0 when reboot
 - decrement TTL of each stored LSP; discard when TTL=0

Link State

Reliable Flooding



Flooding of link-state packets. (a) LSP arrives at node X; (b) X floods LSP to A and C; (c) A and C flood LSP to B (but not X); (d) flooding is complete

Shortest Path Routing

- Dijkstra's Algorithm - Assume non-negative link weights
 - N : set of nodes in the graph
 - $l((i, j))$: the non-negative cost associated with the edge between nodes $i, j \in N$ and $l(i, j) = \infty$ if no edge connects i and j
 - Let $s \in N$ be the starting node which executes the algorithm to find shortest paths to all other nodes in N
 - Two variables used by the algorithm
 - M : set of nodes incorporated so far by the algorithm
 - $C(n)$: the cost of the path from s to each node n
 - The algorithm

```
M = {s}
```

```
For each n in N - {s}
```

```
    C(n) = l(s, n)
```

```
while ( N ≠ M)
```

```
    M = M ∪ {w} such that C(w) is the minimum
                               for all w in (N-M)
```

```
    For each n in (N-M)
```

```
        C(n) = MIN (C(n), C(w) + l(w, n))
```

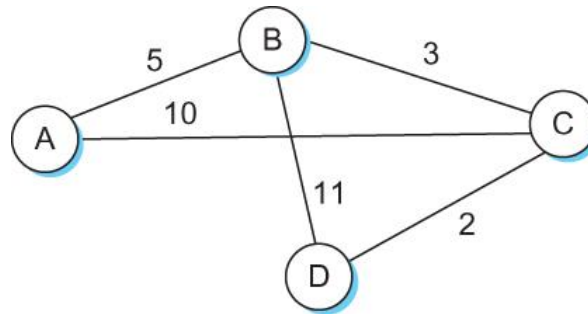

Shortest Path Routing

- In practice, each switch computes its routing table directly from the LSP's it has collected using a realization of Dijkstra's algorithm called the *forward search algorithm*
- Specifically each switch maintains two lists, known as **Tentative** and **Confirmed**
- Each of these lists contains a set of entries of the form (Destination, Cost, NextHop)

Shortest Path Routing

- The algorithm
 - Initialize the **Confirmed** list with an entry for myself; this entry has a cost of 0
 - For the node just added to the **Confirmed** list in the previous step, call it node **Next**, select its LSP
 - For each neighbor (Neighbor) of **Next**, calculate the cost (Cost) to reach this Neighbor as the sum of the cost from myself to Next and from Next to Neighbor
 - If Neighbor is currently on neither the **Confirmed** nor the **Tentative** list, then add (Neighbor, Cost, Nexthop) to the **Tentative** list, where Nexthop is the direction I go to reach Next
 - If Neighbor is currently on the **Tentative** list, and the Cost is less than the currently listed cost for the Neighbor, then replace the current entry with (Neighbor, Cost, Nexthop) where Nexthop is the direction I go to reach Next
 - If the **Tentative** list is empty, stop. Otherwise, pick the entry from the **Tentative** list with the lowest cost, move it to the **Confirmed** list, and return to Step 2.

Shortest Path Routing



Step	Confirmed	Tentative	Comments
1	(D,0,-)		Since D is the only new member of the confirmed list, look at its LSP.
2	(D,0,-)	(B,11,B) (C,2,C)	D's LSP says we can reach B through B at cost 11, which is better than anything else on either list, so put it on Tentative list; same for C.
3	(D,0,-) (C,2,C)	(B,11,B)	Put lowest-cost member of Tentative (C) onto Confirmed list. Next, examine LSP of newly confirmed member (C).
4	(D,0,-) (C,2,C)	(B,5,C) (A,12,C)	Cost to reach B through C is 5, so replace (B,11,B). C's LSP tells us that we can reach A at cost 12.
5	(D,0,-) (C,2,C) (B,5,C)	(A,12,C)	Move lowest-cost member of Tentative (B) to Confirmed, then look at its LSP.
6	(D,0,-) (C,2,C) (B,5,C)	(A,10,C)	Since we can reach A at cost 5 through B, replace the Tentative entry.
7	(D,0,-) (C,2,C) (B,5,C) (A,10,C)		Move lowest-cost member of Tentative (A) to Confirmed, and we are all done.

Open Shortest Path First (OSPF)

0	8	16	31
Version	Type	Message length	
SourceAddr			
AreaId			
Checksum		Authentication type	
Authentication			

OSPF Header Format

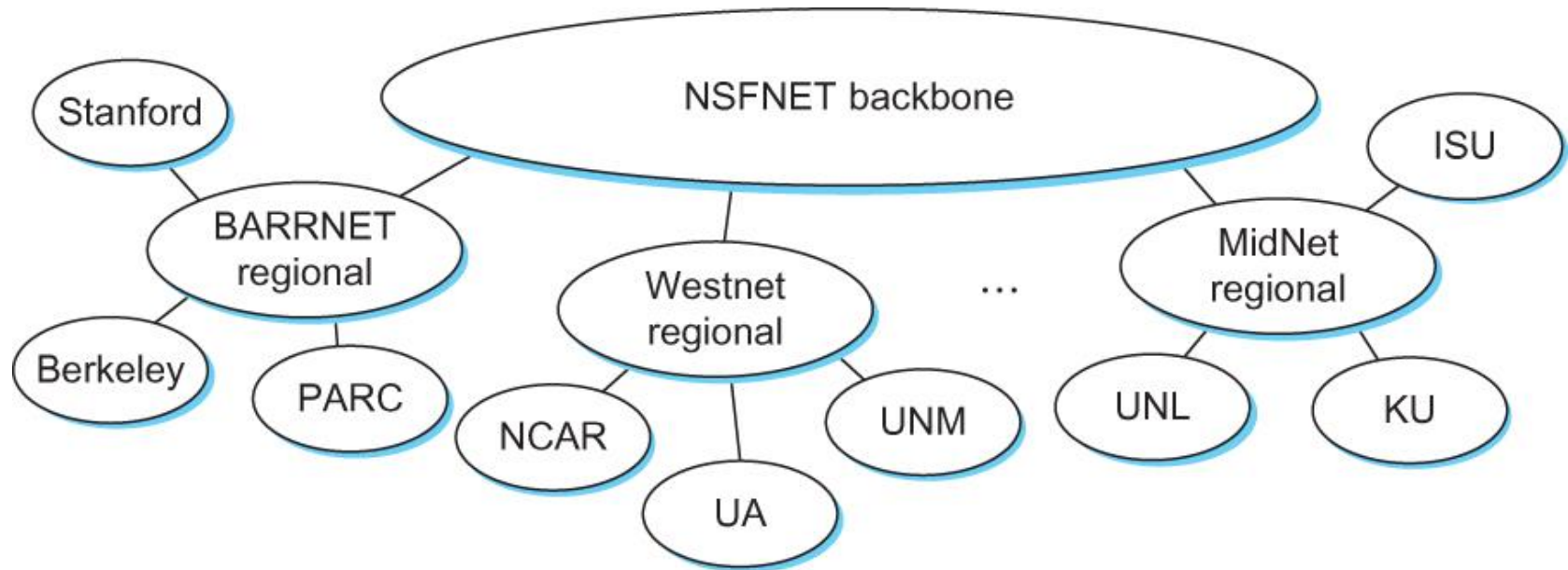
LS Age		Options	Type = 1
Link-state ID			
Advertising router			
LS sequence number			
LS checksum		Length	
0	Flags	0	Number of links
Link ID			
Link data			
Link type	Num_TOS	Metric	
Optional TOS information			
More links			

OSPF Link State Advertisement

Summary

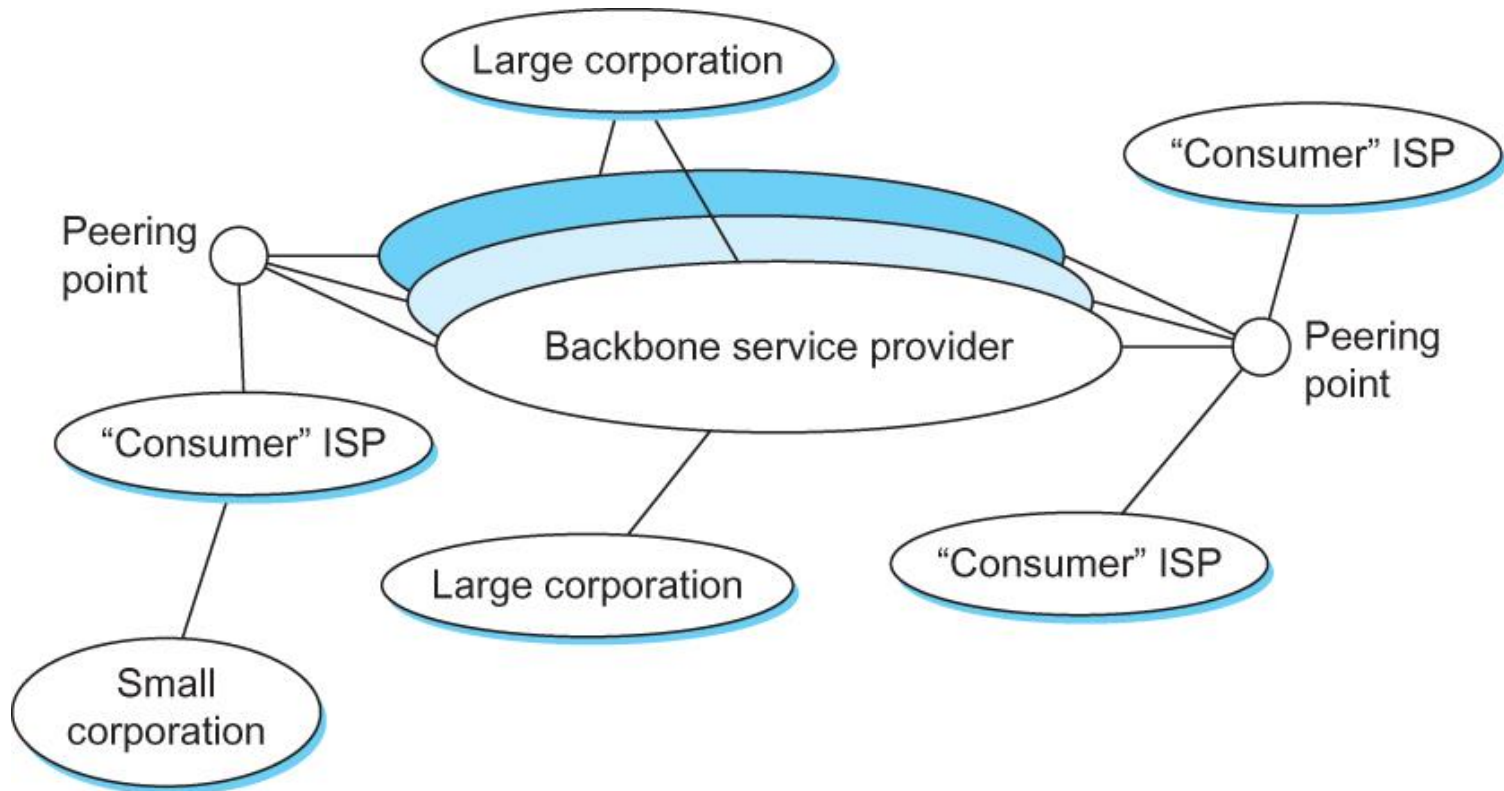
- We have looked at some of the issues involved in building scalable and heterogeneous networks by using switches and routers to interconnect links and networks.
- To deal with heterogeneous networks, we have discussed in details the service model of Internetworking Protocol (IP) which forms the basis of today's routers.
- We have discussed in details two major classes of routing algorithms
 - Distance Vector
 - Link State

The Global Internet



The tree structure of the Internet in 1990

The Global Internet

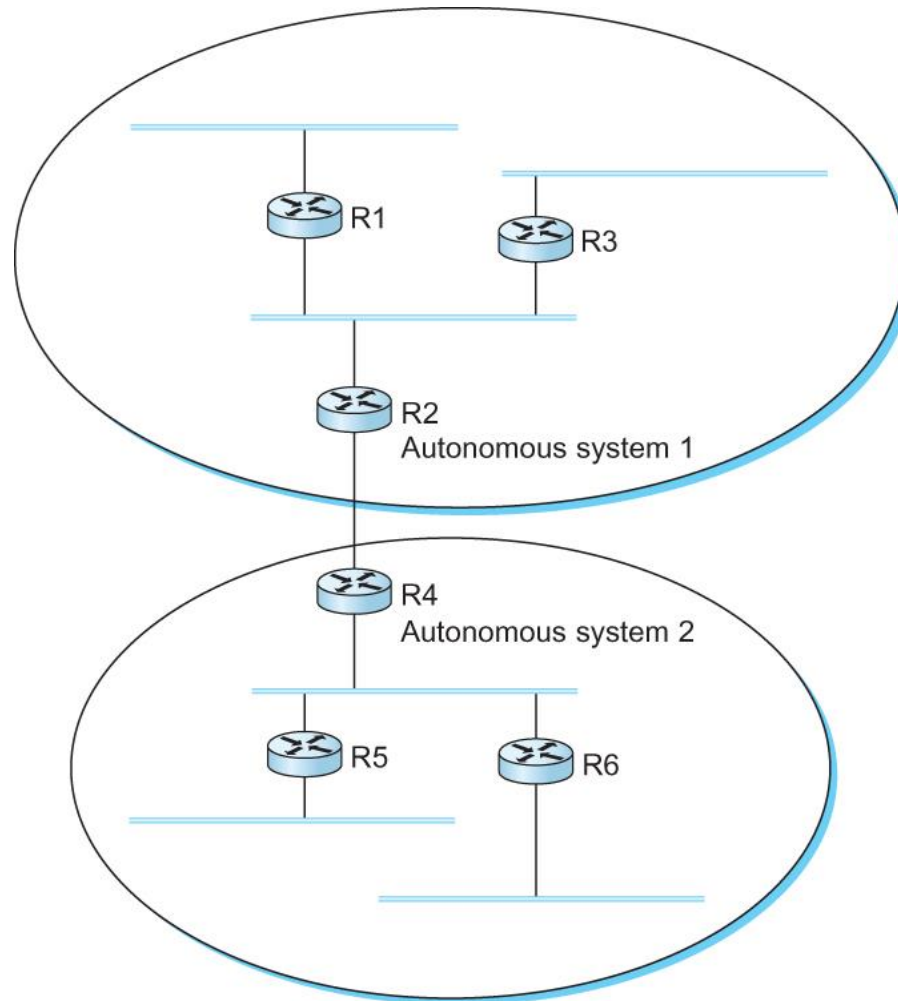


A simple multi-provider Internet

Interdomain Routing (BGP)

- Internet is organized as autonomous systems (AS) each of which is under the control of a single administrative entity
- Autonomous System (AS)
 - corresponds to an administrative domain
 - examples: University, company, backbone network
- A corporation's internal network might be a single AS, as may the network of a single Internet service provider

Interdomain Routing



A network with two autonomous system

Route Propagation

- Idea: Provide an additional way to hierarchically aggregate routing information in a large internet.
 - Improves scalability
- Divide the routing problem in two parts:
 - Routing within a single autonomous system
 - Routing between autonomous systems
- Another name for autonomous systems in the Internet is routing domains
 - Two-level route propagation hierarchy
 - Inter-domain routing protocol (Internet-wide standard)
 - Intra-domain routing protocol (each AS selects its own)

EGP and BGP

- Inter-domain Routing Protocols
 - Exterior Gateway Protocol (EGP)
 - Forced a tree-like topology onto the Internet
 - Did not allow for the topology to become general
 - Tree like structure: there is a single backbone and autonomous systems are connected only as parents and children and not as peers
 - Border Gateway Protocol (BGP)
 - Assumes that the Internet is an arbitrarily interconnected set of ASs.
 - Today's Internet consists of an interconnection of multiple backbone networks (they are usually called service provider networks, and they are operated by private companies rather than the government)
 - Sites are connected to each other in arbitrary ways

BGP

- Some large corporations connect directly to one or more of the backbone, while others connect to smaller, non-backbone service providers.
- Many service providers exist mainly to provide service to “consumers” (individuals with PCs in their homes), and these providers must connect to the backbone providers
- Often many providers arrange to interconnect with each other at a single “peering point”

BGP-4: Border Gateway Protocol

- Assumes the Internet is an arbitrarily interconnected set of AS's.
- Define *local traffic* as traffic that originates at or terminates on nodes within an AS, and *transit traffic* as traffic that passes through an AS.
- We can classify AS's into three types:
 - *Stub AS*: an AS that has only a single connection to one other AS; such an AS will only carry local traffic (*small corporation in the figure of the previous page*).
 - *Multihomed AS*: an AS that has connections to more than one other AS, but refuses to carry transit traffic (*large corporation at the top in the figure of the previous page*).
 - *Transit AS*: an AS that has connections to more than one other AS, and is designed to carry both transit and local traffic (*backbone providers in the figure of the previous page*).

BGP

- The goal of Inter-domain routing is to find any path to the intended destination that is loop free
 - We are concerned with reachability than optimality
 - Finding path anywhere close to optimal is considered to be a great achievement

- Why?

BGP

- Scalability: An Internet backbone router must be able to forward any packet destined anywhere in the Internet
 - Having a routing table that will provide a match for any valid IP address
- Autonomous nature of the domains
 - It is impossible to calculate meaningful path costs for a path that crosses multiple ASs
 - A cost of 1000 across one provider might imply a great path but it might mean an unacceptable bad one from another provid
- Issues of trust
 - Provider A might be unwilling to believe certain advertisements from provider B

BGP

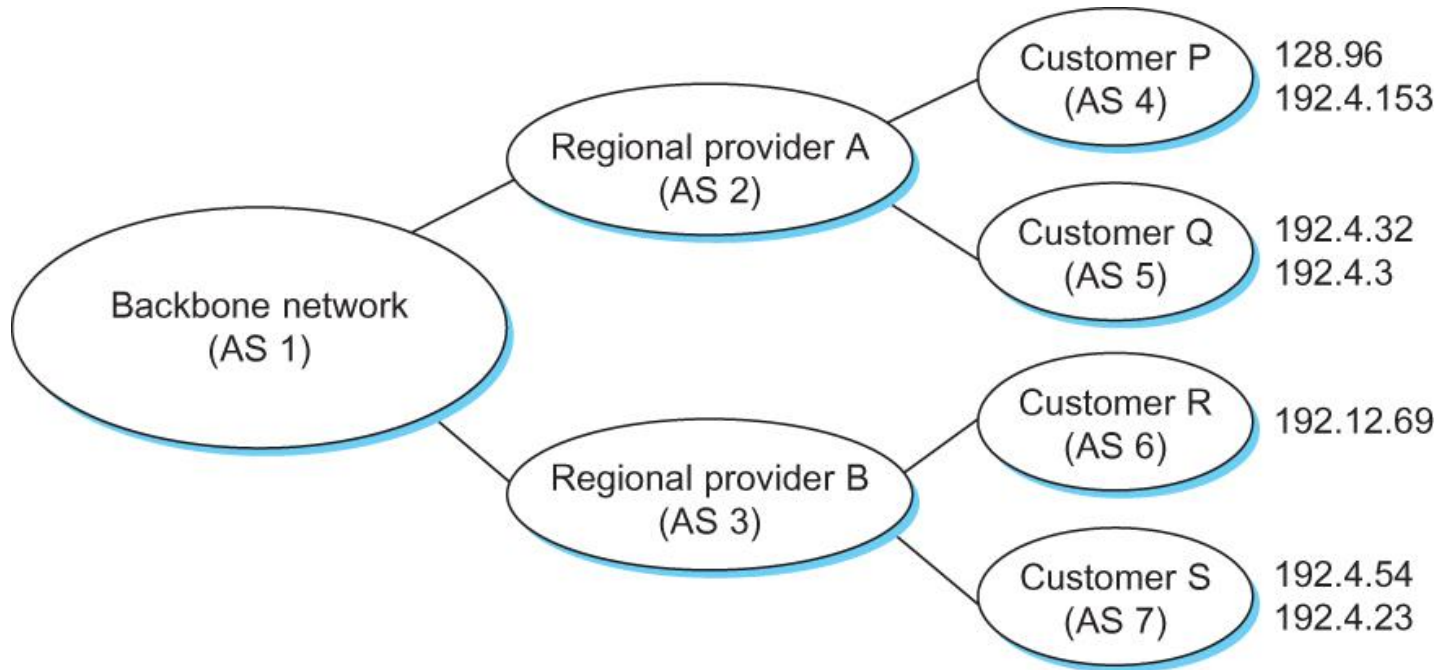
Each AS has:

- One BGP *speaker* that advertises:
 - local networks
 - other reachable networks (transit AS only)
 - gives *path* information
- In addition to the BGP speakers, the AS has one or more border “gateways” which need not be the same as the speakers
- The border gateways are the routers through which packets enter and leave the AS

BGP

- BGP does not belong to either of the two main classes of routing protocols (distance vectors and link-state protocols)
- BGP advertises *complete paths* as an enumerated lists of ASs to reach a particular network

BGP Example



Example of a network running BGP

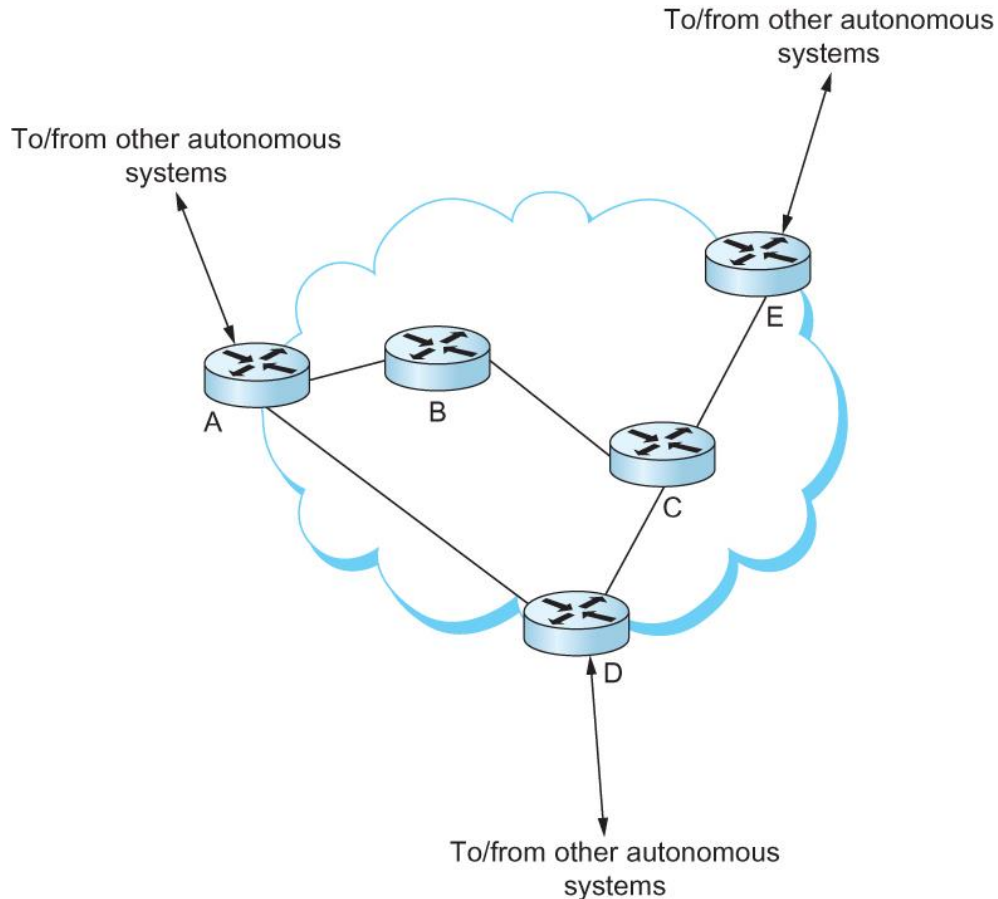
BGP Example

- Speaker for AS 2 advertises reachability to P and Q
 - Network 128.96, 192.4.153, 192.4.32, and 192.4.3, can be reached directly from AS 2.
- Speaker for backbone network then advertises
 - Networks 128.96, 192.4.153, 192.4.32, and 192.4.3 can be reached along the path <AS 1, AS 2>.
- Speaker can also cancel previously advertised paths

BGP Issues

- It should be apparent that the AS numbers carried in BGP need to be unique
- For example, AS 2 can only recognize itself in the AS path in the example if no other AS identifies itself in the same way
- AS numbers are 16-bit numbers assigned by a central authority

Integrating Interdomain and Intradomain Routing



All routers run iBGP and an intradomain routing protocol. Border routers (A, D, E) also run eBGP to other ASs

Integrating Interdomain and Intradomain Routing

Prefix	BGP Next Hop
18.0/16	E
12.5.5/24	A
128.34/16	D
128.69./16	A

BGP table for the AS

Router	IGP Path
A	A
C	C
D	C
E	C

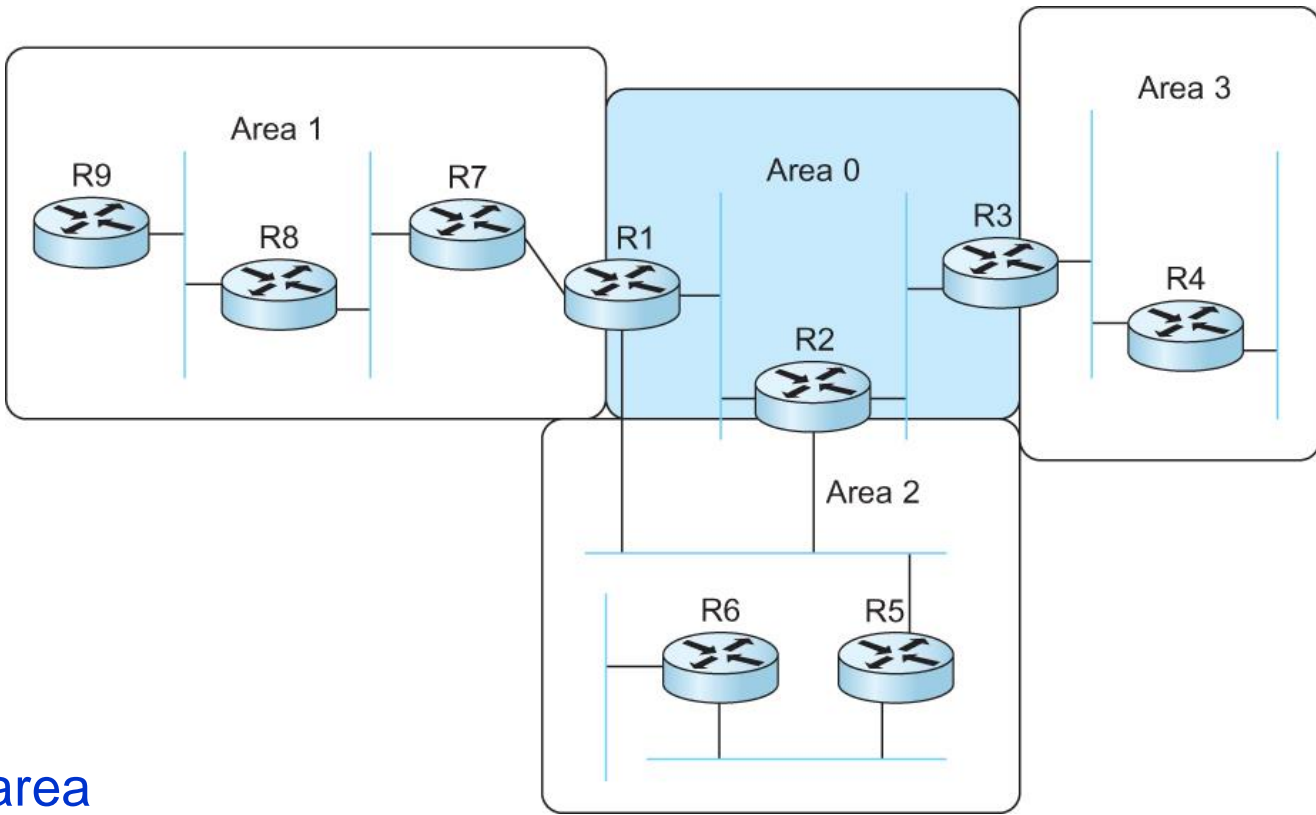
IGP table for router B

Prefix	IGP Path
18.0/16	C
12.5.5/24	A
128.34/16	C
128.69./16	A

Combined table for router B

BGP routing table, IGP routing table, and combined table at router B

Routing Areas



Backbone area

Area border router (ABR)

A domain divided into area

Next Generation IP (IPv6)

Major Features

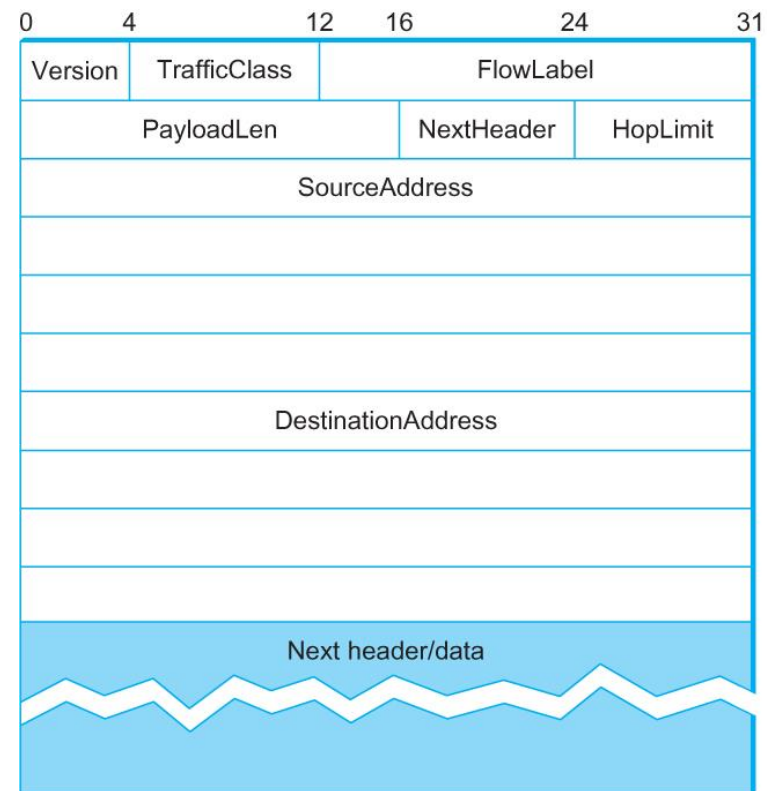
- 128-bit addresses
- Multicast
- Real-time service
- Authentication and security
- Auto-configuration
- End-to-end fragmentation
- Enhanced routing functionality, including support for mobile hosts

IPv6 Addresses

- Classless addressing/routing (similar to CIDR)
- Notation: x:x:x:x:x:x:x:x (x = 16-bit hex number)
 - contiguous 0s are compressed:
47CD:: - IPv6 compatible IPv4 address: ::128.42.1.87
- Address assignment
 - provider-based
 - geographic

IPv6 Header

- 40-byte “base” header
- Extension headers (fixed order, mostly fixed length)
 - fragmentation
 - source routing
 - authentication and security
 - other options



Internet Multicast

Overview

- IPv4
 - class D addresses
 - demonstrated with MBone
 - uses tunneling
- Integral part of IPv6
 - problem is making it scale

Overview

- One-to-many
 - Radio station broadcast
 - Transmitting news, stock-price
 - Software updates to multiple hosts
- Many-to-many
 - Multimedia teleconferencing
 - Online multi-player games
 - Distributed simulations

Overview

- Without support for multicast
 - A source needs to send a separate packet with the identical data to each member of the group
 - This redundancy consumes more bandwidth
 - Redundant traffic is not evenly distributed, concentrated near the sending host
 - Source needs to keep track of the IP address of each member in the group
 - Group may be dynamic
- To support many-to-many and one-to-many IP provides an IP-level multicast

Overview

- Basic IP multicast model is many-to-many based on multicast groups
 - Each group has its own IP multicast address
 - Hosts that are members of a group receive copies of any packets sent to that group's multicast address
 - A host can be in multiple groups
 - A host can join and leave groups

Overview

- Using IP multicast to send the identical packet to each member of the group
 - A host sends a single copy of the packet addressed to the group's multicast address
 - The sending host does not need to know the individual unicast IP address of each member
 - Sending host does not send multiple copies of the packet

Overview

- IP's original many-to-many multicast has been supplemented with support for a form of one-to-many multicast
- One-to-many multicast
 - Source specific multicast (SSM)
 - A receiving host specifies both a multicast group and a specific sending host
- Many-to-many model
 - Any source multicast (ASM)

Overview

- A host signals its desire to join or leave a multicast group by communicating with its local router using a special protocol
 - In IPv4, the protocol is Internet Group Management Protocol (IGMP)
 - In IPv6, the protocol is Multicast Listener Discovery (MLD)
- The router has the responsibility for making multicast behave correctly with regard to the host

Multicast Routing

- A router's unicast forwarding tables indicate for any IP address, which link to use to forward the unicast packet
- To support multicast, a router must additionally have multicast forwarding tables that indicate, based on multicast address, which links to use to forward the multicast packet
- Unicast forwarding tables collectively specify a set of paths
- Multicast forwarding tables collectively specify a set of trees
 - Multicast distribution trees

Multicast Routing

- To support source specific multicast, the multicast forwarding tables must indicate which links to use based on the combination of multicast address and the unicast IP address of the source
- Multicast routing is the process by which multicast distribution trees are determined

Distance-Vector Multicast

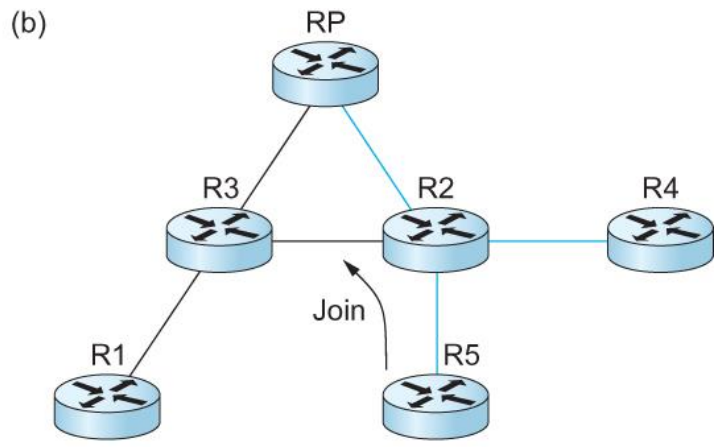
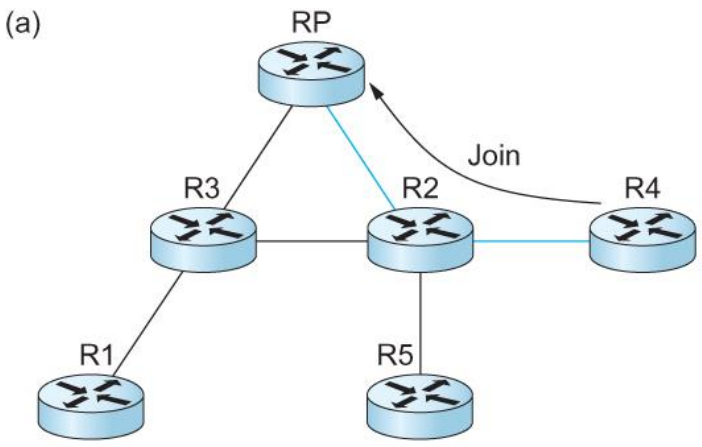
- Each router already knows that shortest path to source S goes through router N.
- When receive multicast packet from S, forward on all outgoing links (except the one on which the packet arrived), iff packet arrived from N.
- Eliminate duplicate broadcast packets by only letting
 - “parent” for LAN (relative to S) forward
 - shortest path to S (learn via distance vector)
 - smallest address to break ties

Distance-Vector Multicast

Reverse Path Broadcast (RPB)

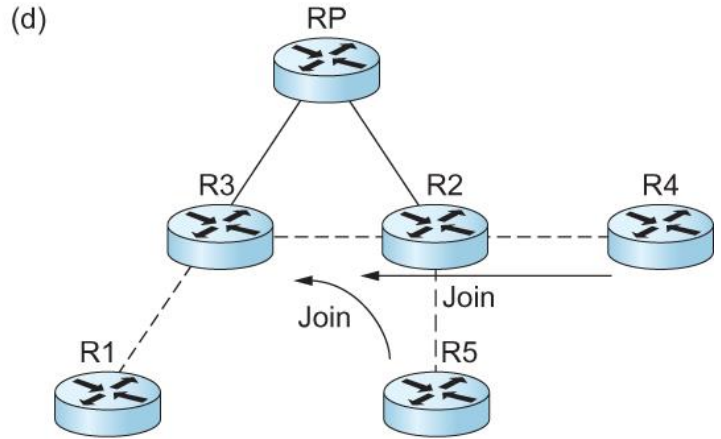
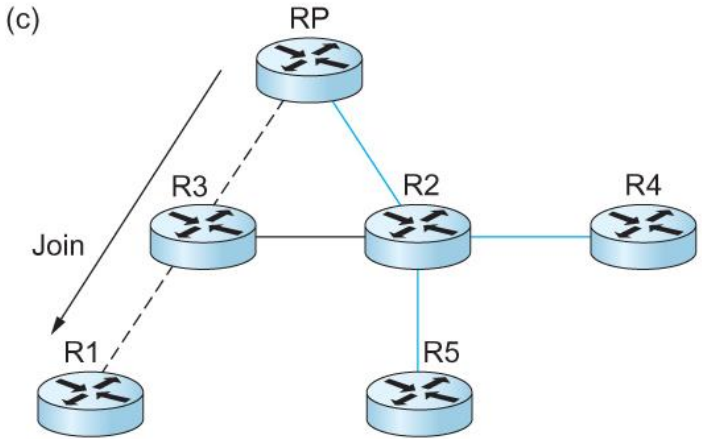
- Goal: Prune networks that have no hosts in group G
- Step 1: Determine if LAN is a *leaf* with no members in G
 - leaf if parent is only router on the LAN
 - determine if any hosts are members of G using IGMP
- Step 2: Propagate “no members of G here” information
 - augment **<Destination, Cost>** update sent to neighbors with set of groups for which this network is interested in receiving multicast packets.
 - only happens when multicast address becomes active.

Protocol Independent Multicast (PIM)



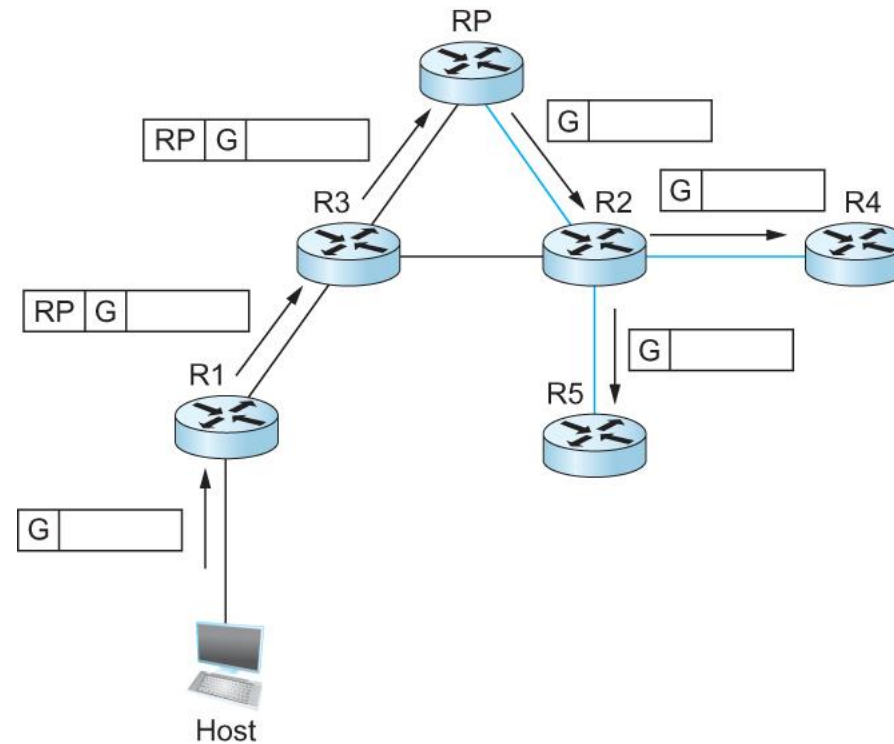
Shared Tree

Source specific tree



RP=Rendezvous point
 — Shared tree
 - - - Source-specific tree for source R1

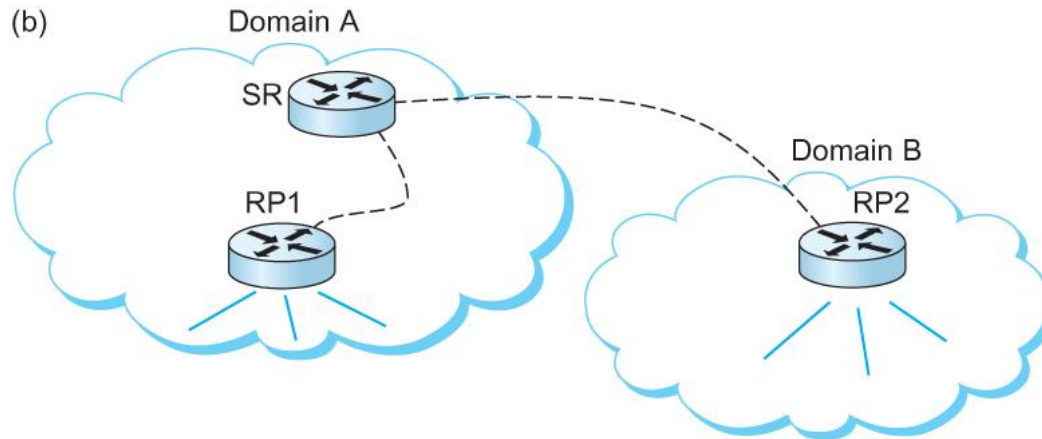
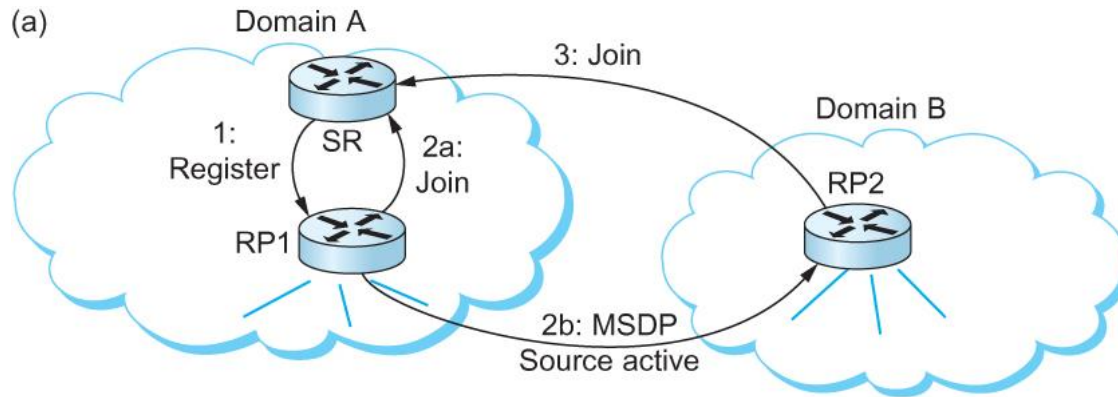
Protocol Independent Multicast (PIM)



Delivery of a packet along a shared tree. R1 tunnels the packet to the RP, which forwards it along the shared tree to R4 and R5.

Inter-domain Multicast

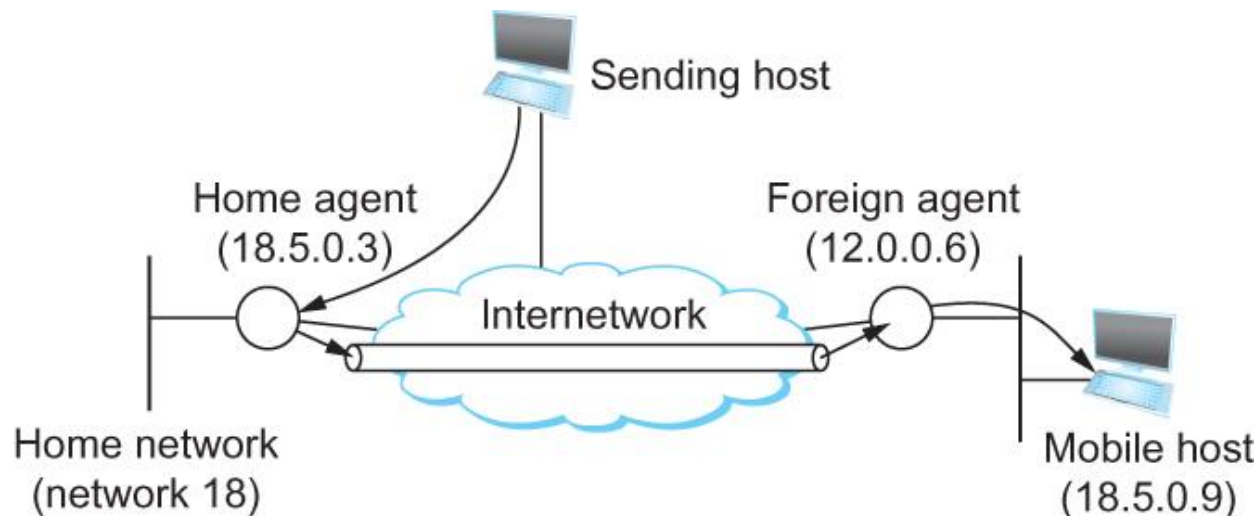
Multicast Source Discovery Protocol (MSDP)



— Shared tree
 - - - Source-specific tree for source SR

Routing for Mobile Hosts

- Mobile IP
 - *home agent*
 - Router located on the home network of the mobile hosts
 - *home address*
 - The permanent IP address of the mobile host.
 - Has a network number equal to that of the home network and thus of the home agent
 - *foreign agent*
 - Router located on a network to which the mobile node attaches itself when it is away from its home network



Routing for Mobile Hosts

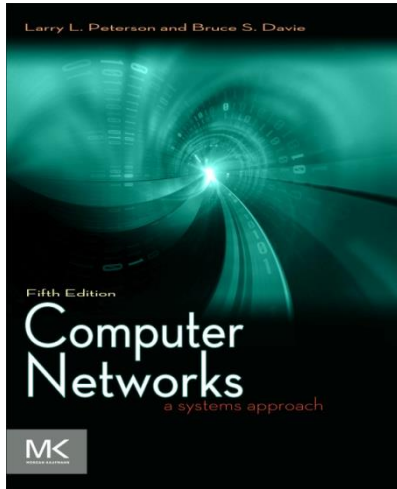
- Problem of delivering a packet to the mobile node
 - How does the home agent intercept a packet that is destined for the mobile node?
 - Proxy ARP
 - How does the home agent then deliver the packet to the foreign agent?
 - IP tunnel
 - Care-of-address
 - How does the foreign agent deliver the packet to the mobile node?

Routing for Mobile Hosts

- Route optimization in Mobile IP
 - The route from the sending node to mobile node can be significantly sub-optimal
 - One extreme example
 - The mobile node and the sending node are on the same network, but the home network for the mobile node is on the far side of the Internet
 - Triangle Routing Problem
 - Solution
 - Let the sending node know the care-of-address of the mobile node. The sending node can create its own tunnel to the foreign agent
 - Home agent sends binding update message
 - The sending node creates an entry in the binding cache
 - The binding cache may become out-of-date
 - The mobile node moved to a different network
 - Foreign agent sends a binding warning message

Summary

- We have looked at the issues of scalability in routing in the Internet
- We have discussed IPV6
- We have discussed Multicasting
- We have discussed Mobile IP



UNIT 4

End-to-End Communication

Problem

- How to turn this host-to-host packet delivery service into a process-to-process communication channel

Chapter Outline

- Simple Demultiplexer (UDP)
- Reliable Byte Stream (TCP)

Chapter Goal

- Understanding the demultiplexing service
- Discussing simple byte stream protocol

End-to-end Protocols

- Common properties that a transport protocol can be expected to provide
 - Guarantees message delivery
 - Delivers messages in the same order they were sent
 - Delivers at most one copy of each message
 - Supports arbitrarily large messages
 - Supports synchronization between the sender and the receiver
 - Allows the receiver to apply flow control to the sender
 - Supports multiple application processes on each host

End-to-end Protocols

- Typical limitations of the network on which transport protocol will operate
 - Drop messages
 - Reorder messages
 - Deliver duplicate copies of a given message
 - Limit messages to some finite size
 - Deliver messages after an arbitrarily long delay

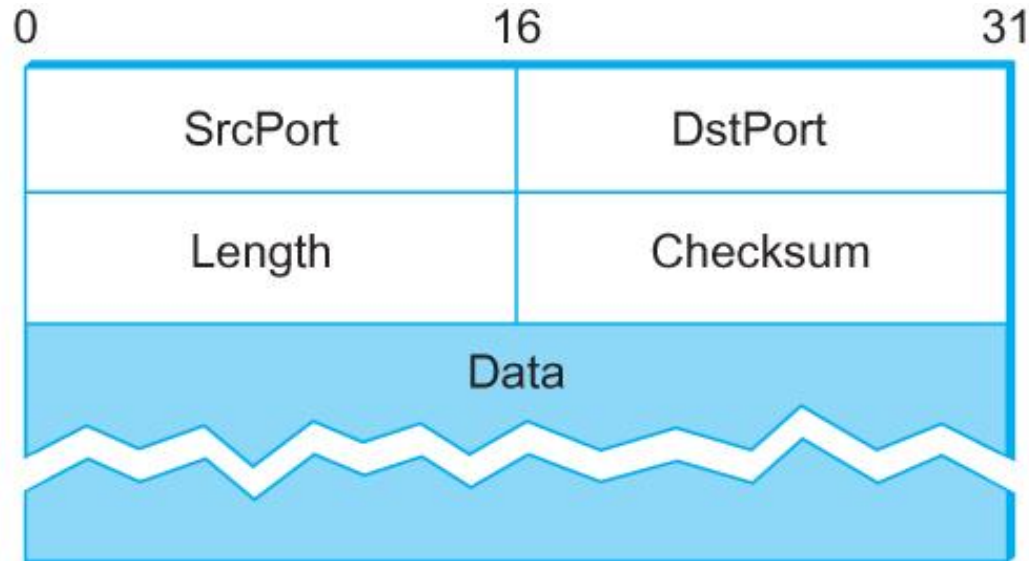
End-to-end Protocols

- Challenge for Transport Protocols
 - Develop algorithms that turn the less-than-desirable properties of the underlying network into the high level of service required by application programs

Simple Demultiplexer (UDP)

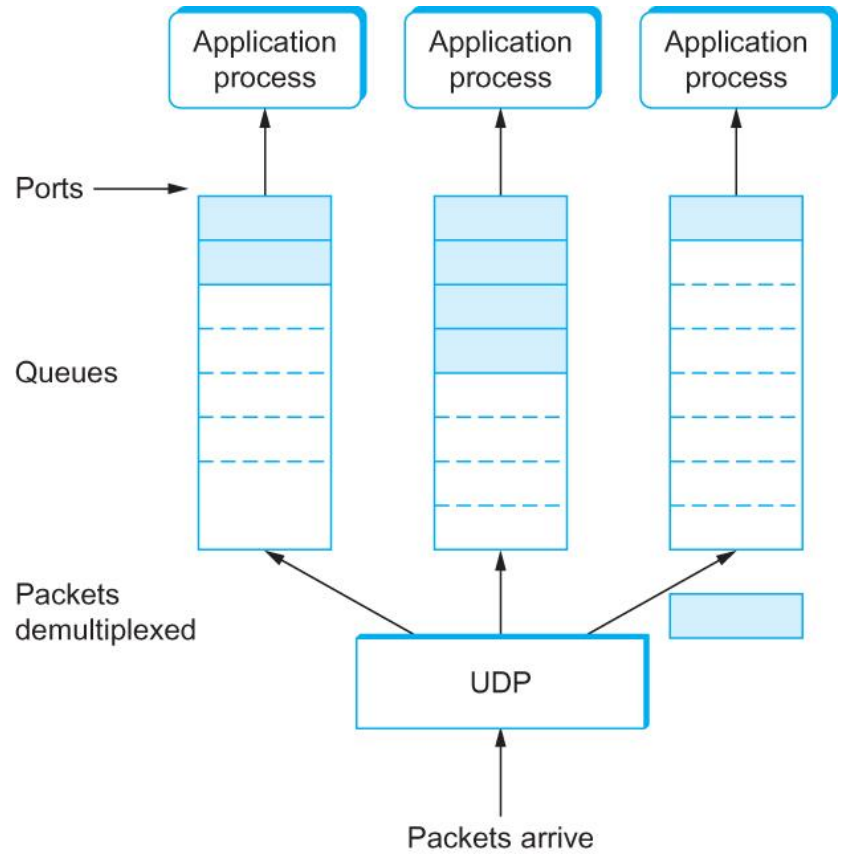
- Extends host-to-host delivery service of the underlying network into a process-to-process communication service
- Adds a level of demultiplexing which allows multiple application processes on each host to share the network

Simple Demultiplexer (UDP)



Format for UDP header (Note: length and checksum fields should be switched)

Simple Demultiplexer (UDP)



UDP Message Queue

Reliable Byte Stream (TCP)

- In contrast to UDP, Transmission Control Protocol (TCP) offers the following services
 - Reliable
 - Connection oriented
 - Byte-stream service

Flow control VS Congestion control

- Flow control involves preventing senders from overrunning the capacity of the receivers
- Congestion control involves preventing too much data from being injected into the network, thereby causing switches or links to become overloaded

End-to-end Issues

- At the heart of TCP is the sliding window algorithm (discussed in Chapter 2)
- As TCP runs over the Internet rather than a point-to-point link, the following issues need to be addressed by the sliding window algorithm
 - TCP supports logical connections between processes that are running on two different computers in the Internet
 - TCP connections are likely to have widely different RTT times
 - Packets may get reordered in the Internet

End-to-end Issues

- TCP needs a mechanism using which each side of a connection will learn what resources the other side is able to apply to the connection
- TCP needs a mechanism using which the sending side will learn the capacity of the network

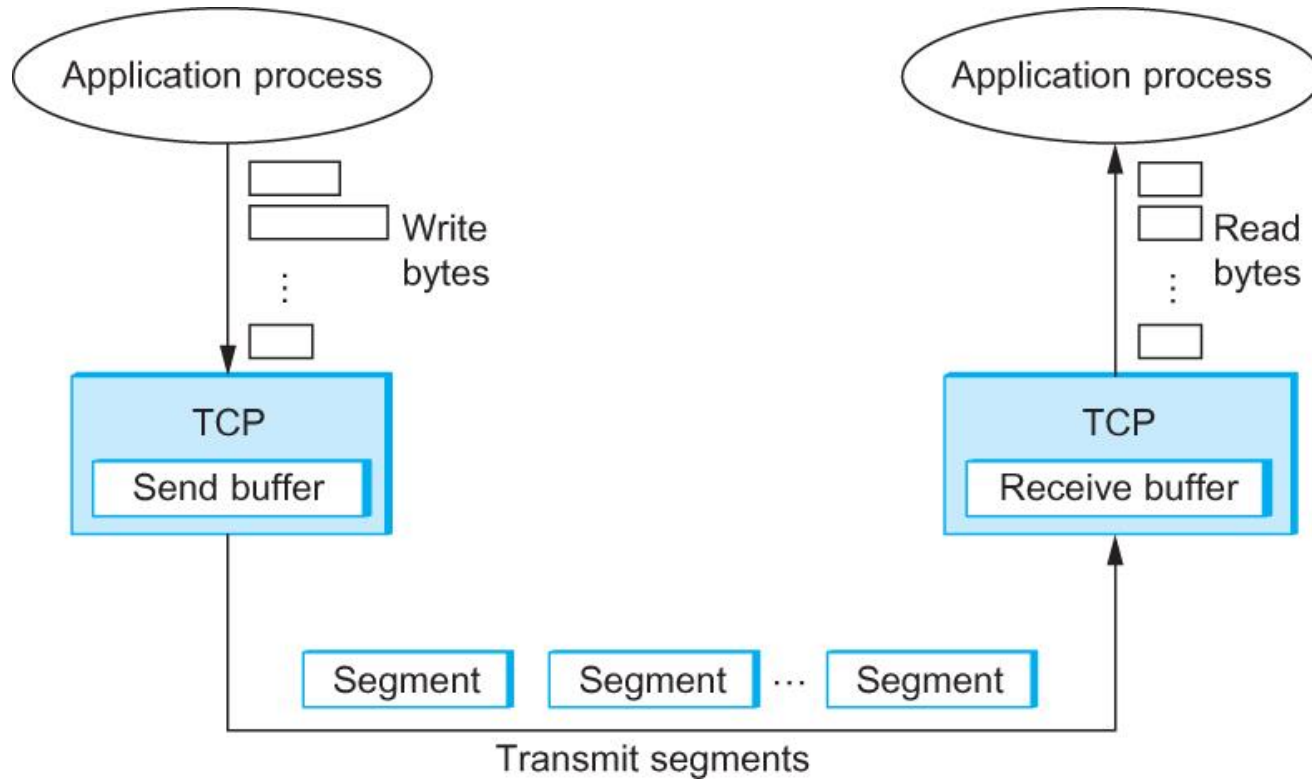
TCP Segment

- TCP is a byte-oriented protocol, which means that the sender writes bytes into a TCP connection and the receiver reads bytes out of the TCP connection.
- Although “byte stream” describes the service TCP offers to application processes, TCP does not, itself, transmit individual bytes over the Internet.

TCP Segment

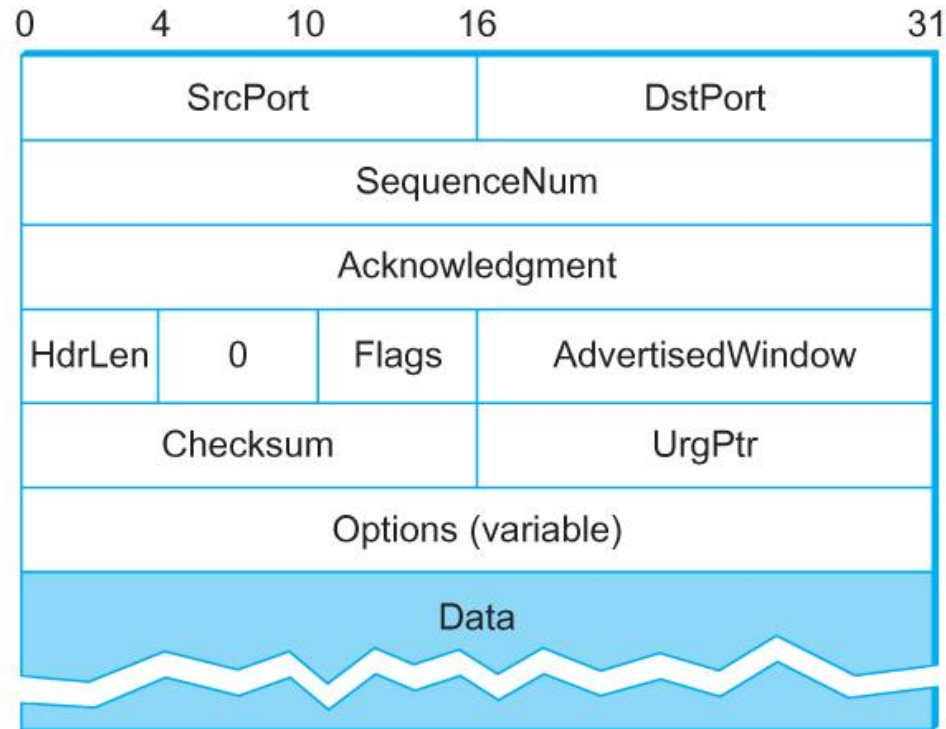
- TCP on the source host buffers enough bytes from the sending process to fill a reasonably sized packet and then sends this packet to its peer on the destination host.
- TCP on the destination host then empties the contents of the packet into a receive buffer, and the receiving process reads from this buffer at its leisure.
- The packets exchanged between TCP peers are called *segments*.

TCP Segment



How TCP manages a byte stream.

TCP Header



TCP Header Format

TCP Header

- The SrcPort and DstPort fields identify the source and destination ports, respectively.
- The Acknowledgment, SequenceNum, and AdvertisedWindow fields are all involved in TCP's sliding window algorithm.
- Because TCP is a byte-oriented protocol, each byte of data has a sequence number; the SequenceNum field contains the sequence number for the first byte of data carried in that segment.
- The Acknowledgment and AdvertisedWindow fields carry information about the flow of data going in the other direction.

TCP Header

- The 6-bit Flags field is used to relay control information between TCP peers.
- The possible flags include SYN, FIN, RESET, PUSH, URG, and ACK.
- The SYN and FIN flags are used when establishing and terminating a TCP connection, respectively.
- The ACK flag is set any time the Acknowledgment field is valid, implying that the receiver should pay attention to it.

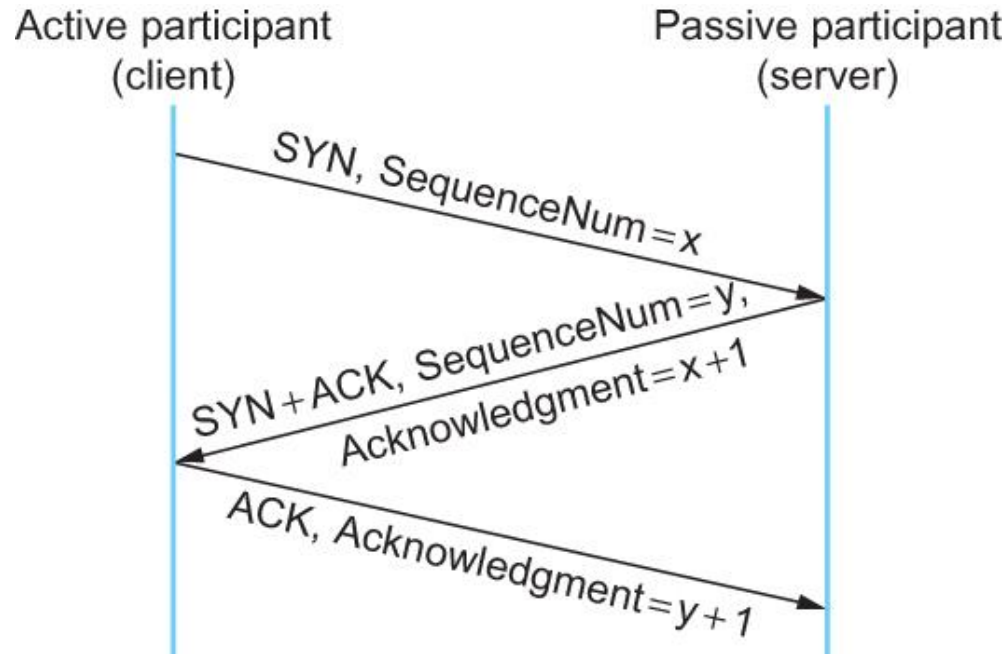
TCP Header

- The URG flag signifies that this segment contains urgent data. When this flag is set, the UrgPtr field indicates where the nonurgent data contained in this segment begins.
- The urgent data is contained at the front of the segment body, up to and including a value of UrgPtr bytes into the segment.
- The PUSH flag signifies that the sender invoked the push operation, which indicates to the receiving side of TCP that it should notify the receiving process of this fact.
- Finally, the RESET flag signifies that the receiver has become confused

TCP Header

- Finally, the RESET flag signifies that the receiver has become confused, it received a segment it did not expect to receive—and so wants to abort the connection.
- Finally, the Checksum field is used in exactly the same way as for UDP—it is computed over the TCP header, the TCP data, and the pseudoheader, which is made up of the source address, destination address, and length fields from the IP header.

Connection Establishment/Termination in TCP

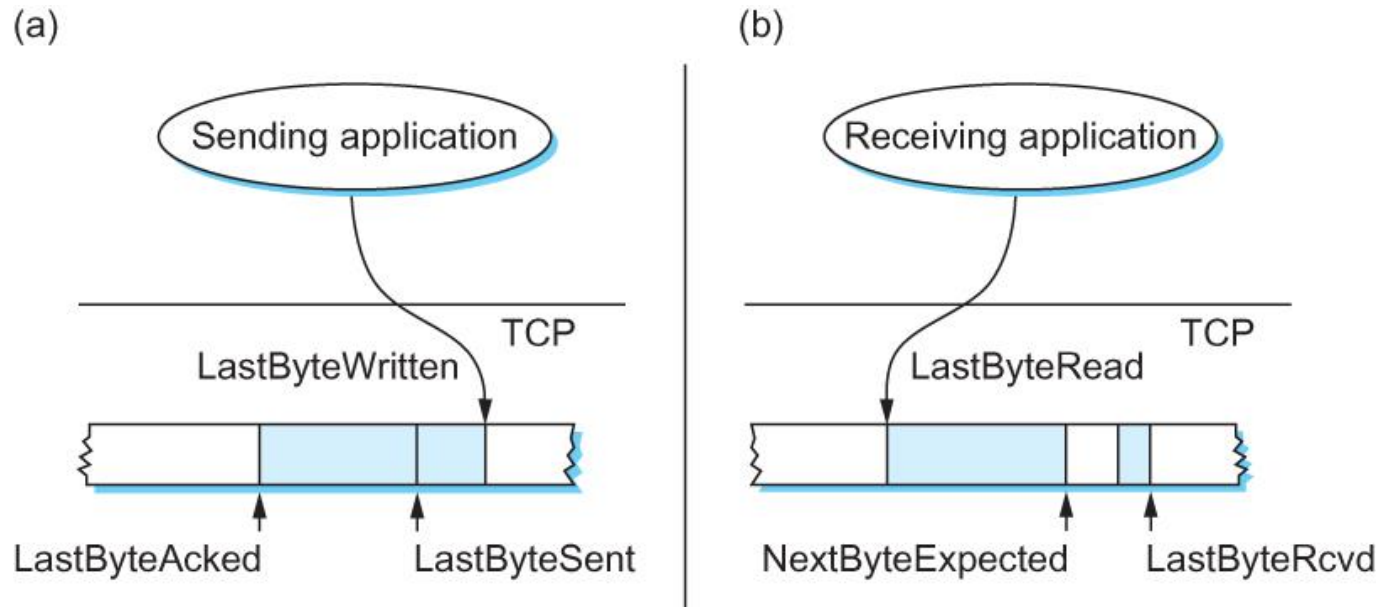


Timeline for three-way handshake algorithm

Sliding Window Revisited

- TCP's variant of the sliding window algorithm, which serves several purposes:
 - (1) it guarantees the reliable delivery of data,
 - (2) it ensures that data is delivered in order, and
 - (3) it enforces flow control between the sender and the receiver.

Sliding Window Revisited



Relationship between TCP send buffer (a) and receive buffer (b).

TCP Sliding Window

- Sending Side
 - $\text{LastByteAcked} \leq \text{LastByteSent}$
 - $\text{LastByteSent} \leq \text{LastByteWritten}$
- Receiving Side
 - $\text{LastByteRead} < \text{NextByteExpected}$
 - $\text{NextByteExpected} \leq \text{LastByteRcvd} + 1$

TCP Flow Control

- $\text{LastByteRcvd} - \text{LastByteRead} \leq \text{MaxRcvBuffer}$
- $\text{AdvertisedWindow} = \text{MaxRcvBuffer} - ((\text{NextByteExpected} - 1) - \text{LastByteRead})$
- $\text{LastByteSent} - \text{LastByteAked} \leq \text{AdvertisedWindow}$
- $\text{EffectiveWindow} = \text{AdvertisedWindow} - (\text{LastByteSent} - \text{LastByteAked})$
- $\text{LastByteWritten} - \text{LastByteAked} \leq \text{MaxSendBuffer}$
- If the sending process tries to write y bytes to TCP, but $(\text{LastByteWritten} - \text{LastByteAked}) + y > \text{MaxSendBuffer}$ then TCP blocks the sending process and does not allow it to generate more data.

Protecting against Wraparound

- SequenceNum: 32 bits long
- AdvertisedWindow: 16 bits long
 - TCP has satisfied the requirement of the sliding window algorithm that is the sequence number
 - space be twice as big as the window size
 - $2^{32} \gg 2 \times 2^{16}$

Protecting against Wraparound

- Relevance of the 32-bit sequence number space
 - The sequence number used on a given connection might wraparound
 - A byte with sequence number x could be sent at one time, and then at a later time a second byte with the same sequence number x could be sent
 - Packets cannot survive in the Internet for longer than the **MSL**
 - **MSL** is set to 120 sec
 - We need to make sure that the sequence number does not wrap around within a 120-second period of time
 - Depends on how fast data can be transmitted over the Internet

Protecting against Wraparound

Bandwidth	Time until Wraparound
T1 (1.5 Mbps)	6.4 hours
Ethernet (10 Mbps)	57 minutes
T3 (45 Mbps)	13 minutes
Fast Ethernet (100 Mbps)	6 minutes
OC-3 (155 Mbps)	4 minutes
OC-12 (622 Mbps)	55 seconds
OC-48 (2.5 Gbps)	14 seconds

Time until 32-bit sequence number space wraps around.

Keeping the Pipe Full

- 16-bit AdvertisedWindow field must be big enough to allow the sender to keep the pipe full
- Clearly the receiver is free not to open the window as large as the AdvertisedWindow field allows
- If the receiver has enough buffer space
 - The window needs to be opened far enough to allow a full delay \times bandwidth product's worth of data
 - Assuming an RTT of 100 ms

Keeping the Pipe Full

Bandwidth	Delay × Bandwidth Product
T1 (1.5 Mbps)	18 KB
Ethernet (10 Mbps)	122 KB
T3 (45 Mbps)	549 KB
Fast Ethernet (100 Mbps)	1.2 MB
OC-3 (155 Mbps)	1.8 MB
OC-12 (622 Mbps)	7.4 MB
OC-48 (2.5 Gbps)	29.6 MB

Required window size for 100-ms RTT.

Triggering Transmission

- How does TCP decide to transmit a segment?
 - TCP supports a byte stream abstraction
 - Application programs write bytes into streams
 - It is up to TCP to decide that it has enough bytes to send a segment

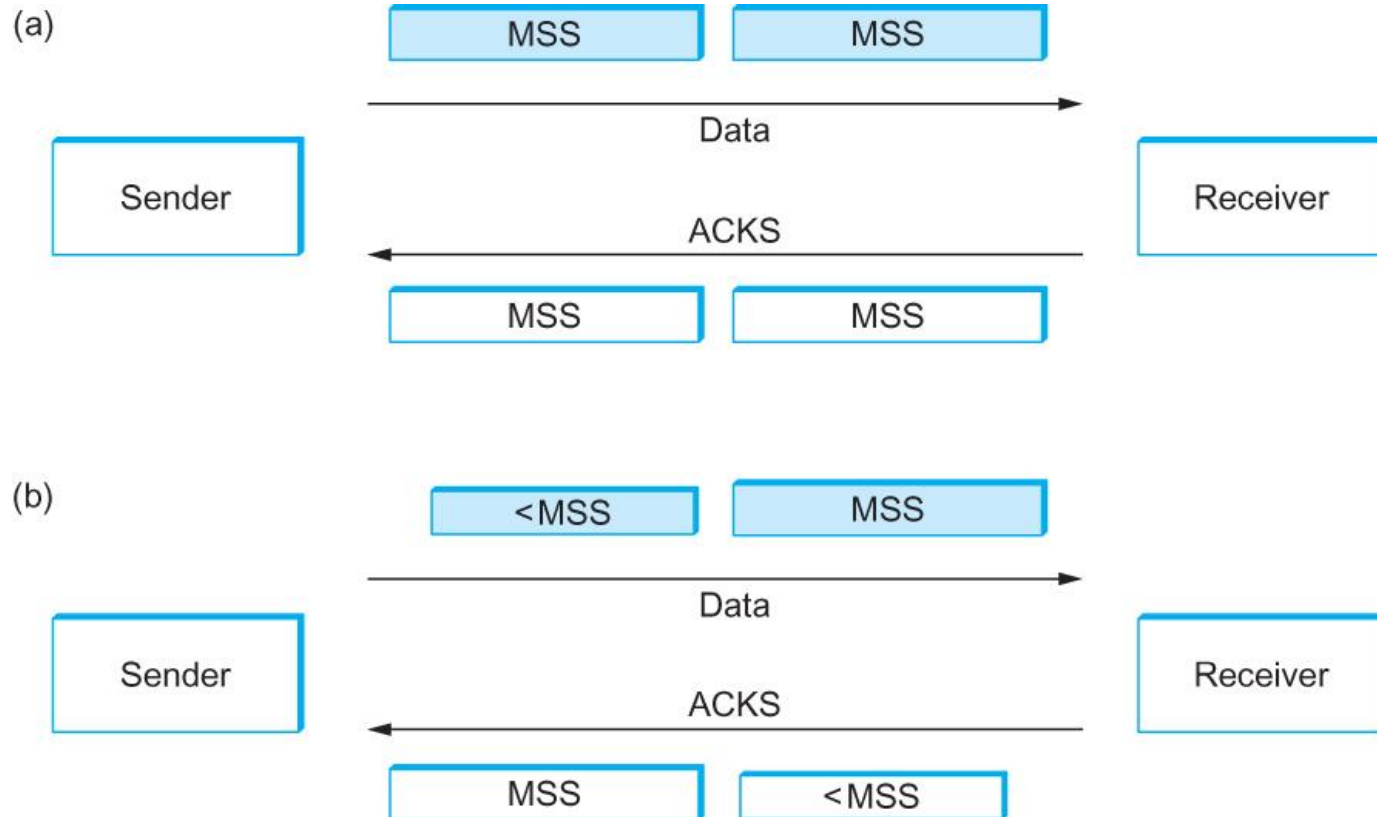
Triggering Transmission

- What factors governs this decision
 - Ignore flow control: window is wide open, as would be the case when the connection starts
 - TCP has three mechanisms to trigger the transmission of a segment
 - 1) TCP maintains a variable MSS and sends a segment as soon as it has collected MSS bytes from the sending process
 - MSS is usually set to the size of the largest segment TCP can send without causing local IP to fragment.
 - MSS: MTU of directly connected network – (TCP header + and IP header)
 - 2) Sending process has explicitly asked TCP to send it
 - TCP supports push operation
 - 3) When a timer fires
 - Resulting segment contains as many bytes as are currently buffered for transmission

Silly Window Syndrome

- If you think of a TCP stream as a conveyer belt with “full” containers (data segments) going in one direction and empty containers (ACKs) going in the reverse direction, then MSS-sized segments correspond to large containers and 1-byte segments correspond to very small containers.
- If the sender aggressively fills an empty container as soon as it arrives, then any small container introduced into the system remains in the system indefinitely.
- That is, it is immediately filled and emptied at each end, and never coalesced with adjacent containers to create larger containers.

Silly Window Syndrome



Silly Window Syndrome

Nagle's Algorithm

- If there is data to send but the window is open less than MSS, then we may want to wait some amount of time before sending the available data
- But how long?
- If we wait too long, then we hurt interactive applications like Telnet
- If we don't wait long enough, then we risk sending a bunch of tiny packets and falling into the *silly window* syndrome
 - The solution is to introduce a timer and to transmit when the timer expires

Nagle's Algorithm

- We could use a clock-based timer, for example one that fires every 100 ms
- Nagle introduced an elegant self-clocking solution
- Key Idea
 - As long as TCP has any data in flight, the sender will eventually receive an ACK
 - This ACK can be treated like a timer firing, triggering the transmission of more data

Nagle's Algorithm

When the application produces data to send

- if both the available data and the window \geq MSS
 - send a full segment
- else
 - if there is unACKed data in flight
 - buffer the new data until an ACK arrives
 - else
 - send all the new data now

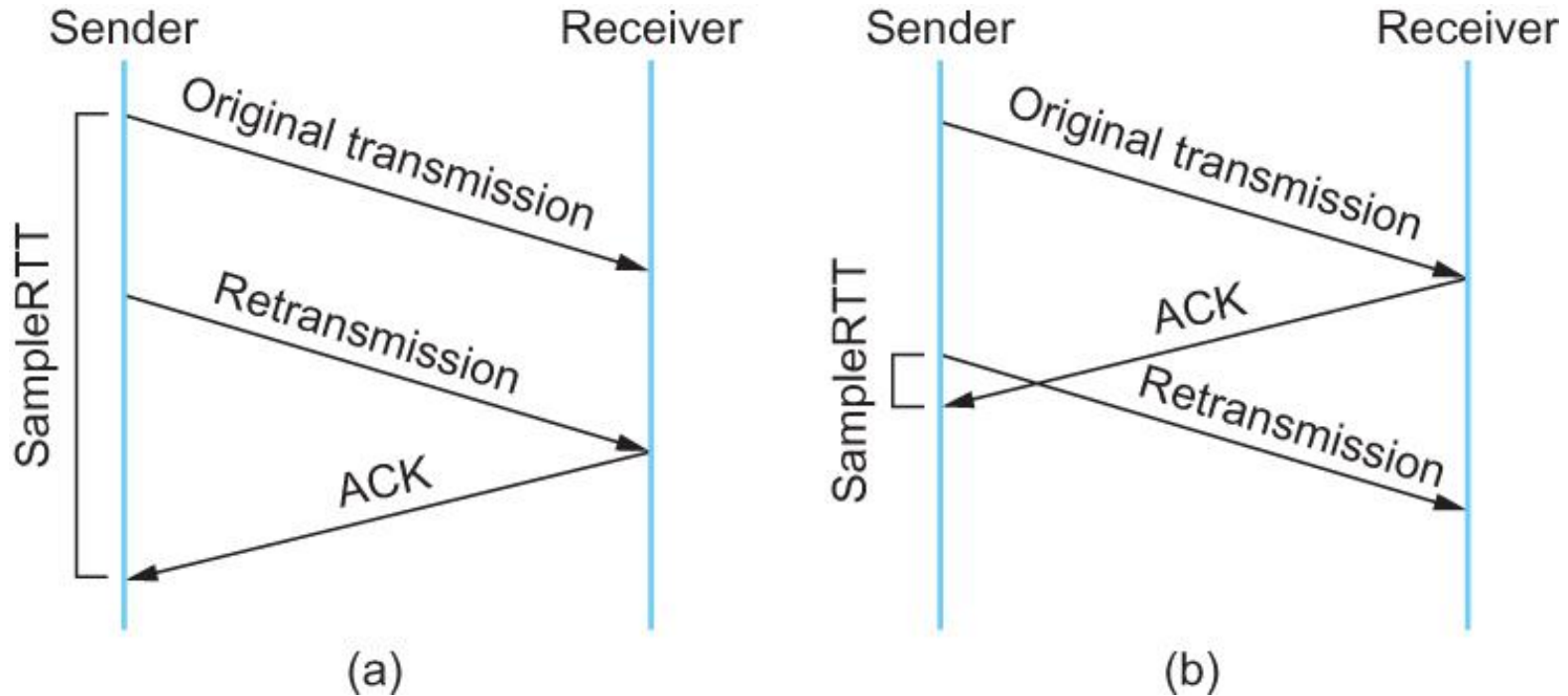
Adaptive Retransmission

- Original Algorithm
 - Measure `sampleRTT` for each segment/ ACK pair
 - Compute weighted average of RTT
 - $\text{EstRTT} = \alpha \times \text{EstRTT} + (1 - \alpha) \times \text{SampleRTT}$
 - α between 0.8 and 0.9
 - Set timeout based on `EstRTT`
 - $\text{TimeOut} = 2 \times \text{EstRTT}$

Original Algorithm

- Problem
 - ACK does not really acknowledge a transmission
 - It actually acknowledges the receipt of data
 - When a segment is retransmitted and then an ACK arrives at the sender
 - It is impossible to decide if this ACK should be associated with the first or the second transmission for calculating RTTs

Karn/Partridge Algorithm



Associating the ACK with (a) original transmission versus (b) retransmission

Karn/Partridge Algorithm

- Do not sample RTT when retransmitting
- Double timeout after each retransmission

Karn/Partridge Algorithm

- Karn-Partridge algorithm was an improvement over the original approach, but it does not eliminate congestion
- We need to understand how timeout is related to congestion
 - If you timeout too soon, you may unnecessarily retransmit a segment which adds load to the network

Karn/Partridge Algorithm

- Main problem with the original computation is that it does not take variance of Sample RTTs into consideration.
- If the variance among Sample RTTs is small
 - Then the Estimated RTT can be better trusted
 - There is no need to multiply this by 2 to compute the timeout

Karn/Partridge Algorithm

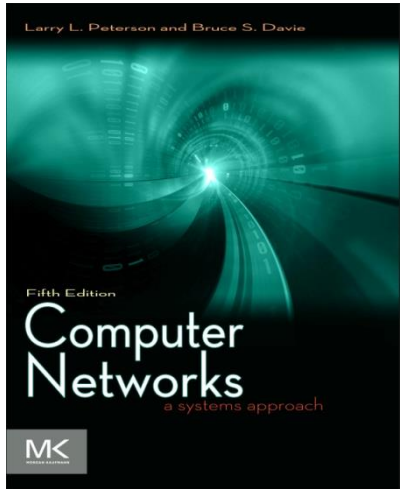
- On the other hand, a large variance in the samples suggest that timeout value should not be tightly coupled to the Estimated RTT
- Jacobson/Karels proposed a new scheme for TCP retransmission

Jacobson/Karels Algorithm

- $\text{Difference} = \text{SampleRTT} - \text{EstimatedRTT}$
- $\text{EstimatedRTT} = \text{EstimatedRTT} + (\alpha \times \text{Difference})$
- $\text{Deviation} = \text{Deviation} + (|\text{Difference}| - \text{Deviation}) \times \beta$
- $\text{TimeOut} = \mu \times \text{EstimatedRTT} + \gamma \times \text{Deviation}$
 - where based on experience, μ is typically set to 1 and γ is set to 4. Thus, when the variance is small, TimeOut is close to EstimatedRTT; a large variance causes the deviation term to dominate the calculation.

Summary

- We have discussed how to convert host-to-host packet delivery service to process-to-process communication channel.
- We have discussed UDP
- We have discussed TCP



UNIT 5

Applications Layer Protocols

Problem

- Applications need their own protocols.
- These applications are part network protocol (in the sense that they exchange messages with their peers on other machines) and part traditional application program (in the sense that they interact with the windowing system, the file system, and ultimately, the user).
- This chapter explores some of the most popular network applications available today.

Chapter Outline

- Traditional Applications
- Multimedia Applications
- Infrastructure Services
- Overlay Networks

Traditional Applications

- Two of the most popular—
 - The World Wide Web and
 - Email.
- Broadly speaking, both of these applications use the request/reply paradigm—users send requests to servers, which then respond accordingly.

Traditional Applications

- It is important to distinguish between application *programs and application protocols*.
- For example, *the HyperText Transport Protocol (HTTP)* is an application protocol that is used to retrieve Web pages from remote servers.
- There can be many different application programs—that is, Web clients like Internet Explorer, Chrome, Firefox, and Safari—that provide users with a different look and feel, but all of them use the same HTTP protocol to communicate with Web servers over the Internet.

Traditional Applications

- Two very widely-used, standardized application protocols:
 - SMTP: Simple Mail Transfer Protocol is used to exchange electronic mail.
 - HTTP: HyperText Transport Protocol is used to communicate between Web browsers and Web servers.

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Email is one of the oldest network applications
 - It is important
 - (1) to distinguish the user interface (i.e., your mail reader) from the underlying message transfer protocols (such as SMTP or IMAP), and
 - (2) to distinguish between this transfer protocol and a companion protocol (RFC 822 and MIME) that defines the format of the messages being exchanged

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Message Format
 - RFC 822 defines messages to have two parts: a *header and a body*. Both parts are represented in ASCII text.
 - Originally, the body was assumed to be simple text. This is still the case, although RFC 822 has been augmented by MIME to allow the message body to carry all sorts of data.
 - This data is still represented as ASCII text, but because it may be an encoded version of, say, a JPEG image, it's not necessarily readable by human users.
 - The message header is a series of <CRLF>-terminated lines. (<CRLF> stands for carriage-return+ line-feed, which are a pair of ASCII control characters often used to indicate the end of a line of text.)

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Message Format
 - The header is separated from the message body by a blank line. Each header line contains a type and value separated by a colon.
 - Many of these header lines are familiar to users since they are asked to fill them out when they compose an email message.
 - RFC 822 was extended in 1993 (and updated quite a few times since then) to allow email messages to carry many different types of data: audio, video, images, PDF documents, and so on.

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Message Format
 - MIME consists of three basic pieces.
 - The first piece is a collection of header lines that augment the original set defined by RFC 822.
 - These header lines describe, in various ways, the data being carried in the message body. They include MIME-Version: (the version of MIME being used), Content-Description: (a human-readable description of what's in the message, analogous to the Subject: line), Content-Type: (the type of data contained in the message), and Content-Transfer- Encoding (how the data in the message body is encoded).
 - The second piece is definitions for a set of content types (and subtypes). For example, MIME defines two different still image types, denoted image/gif and image/jpeg, each with the obvious meaning.
 - The third piece is a way to encode the various data types so they can be shipped in an ASCII email message.

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Message Transfer
 - For many years, the majority of email was moved from host to host using only SMTP.
 - While SMTP continues to play a central role, it is now just one email protocol of several,
 - IMAP and POP being two other important protocols for retrieving mail messages.

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Message Transfer
 - To place SMTP in the right context, we need to identify the key players.
 - First, users interact with a *mail reader when they compose, file, search, and read their email.*
 - *There are countless mail readers available, just like there are many Web browsers to choose from.*
 - *In the early days of the Internet, users typically logged into the machine on which their mailbox resided, and the mail reader they invoked was a local application program that extracted messages from the file system.*
 - *Today, of course, users remotely access their mailbox from their laptop or smartphone; they do not first log into the host that stores their mail (a mail server).*

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Message Transfer
 - To place SMTP in the right context, we need to identify the key players.
 - Second, there is a *mail daemon (or process) running on each host that holds a mailbox*.
 - You can think of this process, also called a *message transfer agent (MTA)*, as playing the role of a post office: users (or their mail readers) give the daemon messages they want to send to other users, the daemon uses SMTP running over TCP to transmit the message to a daemon running on another machine, and the daemon puts incoming messages into the user's mailbox (where that user's mail reader can later find it).
 - Since SMTP is a protocol that anyone could implement, in theory there could be many different implementations of the mail daemon.

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Message Transfer
 - While it is certainly possible that the MTA on a sender's machine establishes an SMTP/TCP connection to the MTA on the recipient's mail server, in many cases the mail traverses one or more *mail gateways on its route from the sender's host to the receiver's host*.
 - Like the end hosts, these gateways also run a message transfer agent process.
 - It's not an accident that these intermediate nodes are called "gateways" since their job is to store and forward email messages, much like an "IP gateway" (which we have referred to as a router) stores and forwards IP datagrams.

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Message Transfer (contd.)
 - The only difference is that a mail gateway typically buffers messages on disk and is willing to try retransmitting them to the next machine for several days, while an IP router buffers datagrams in memory and is only willing to retry transmitting them for a fraction of a second.

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Mail Reader
 - The final step is for the user to actually retrieve his or her messages from the mailbox, read them, reply to them, and possibly save a copy for future reference.
 - The user performs all these actions by interacting with a mail reader.
 - As pointed out earlier, this reader was originally just a program running on the same machine as the user's mailbox, in which case it could simply read and write the file that implements the mailbox.
 - This was the common case in the pre-laptop era.

Traditional Applications

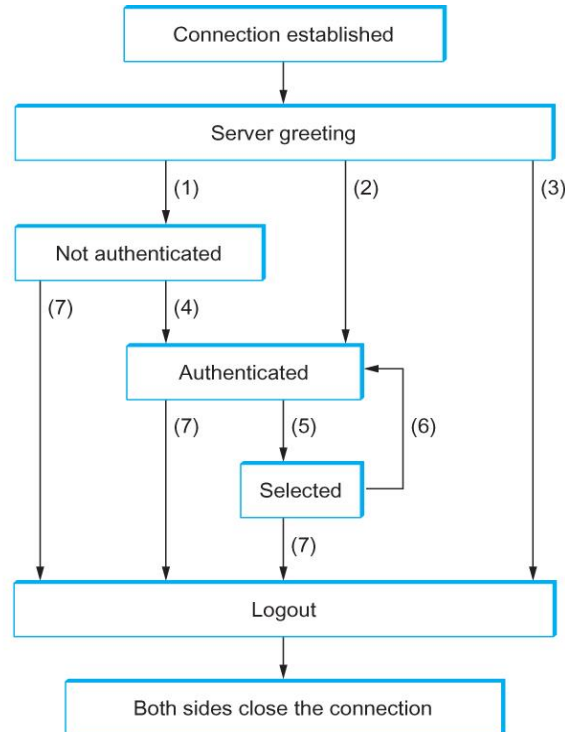
- Electronic Mail (SMTP, MIME, IMAP)
 - Mail Reader
 - Today, most often the user accesses his or her mailbox from a remote machine using yet another protocol, such as the Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP).
 - It is beyond the scope of this book to discuss the user interface aspects of the mail reader, but it is definitely within our scope to talk about the access protocol.

Traditional Applications

- Electronic Mail (SMTP, MIME, IMAP)
 - Mail Reader
 - IMAP is similar to SMTP in many ways.
 - It is a client/server protocol running over TCP, where the client (running on the user's desktop machine) issues commands in the form of <CRLF>-terminated ASCII text lines and the mail server (running on the machine that maintains the user's mailbox) responds in-kind.
 - The exchange begins with the client authenticating him or herself, and identifying the mailbox he or she wants to access.

Traditional Applications

■ Electronic Mail (SMTP, MIME, IMAP)



- (1) Connection without preauthentication (OK greeting)
- (2) Preauthenticated connection (PREAUTH greeting)
- (3) Rejected connection (BYE greeting)
- (4) Successful LOGIN or AUTHENTICATE command
- (5) Successful SELECT or EXAMINE command
- (6) CLOSE command, or failed SELECT or EXAMINE command
- (7) LOGOUT command, server shutdown, or connection closed

IMAP State Transition Diagram

Traditional Applications

- World Wide Web
 - The World Wide Web has been so successful and has made the Internet accessible to so many people that sometimes it seems to be synonymous with the Internet.
 - In fact, the design of the system that became the Web started around 1989, long after the Internet had become a widely deployed system.
 - The original goal of the Web was to find a way to organize and retrieve information, drawing on ideas about hypertext—interlinked documents—that had been around since at least the 1960s.

Traditional Applications

- World Wide Web
 - The core idea of hypertext is that one document can link to another document, and the protocol (HTTP) and document language (HTML) were designed to meet that goal.
 - One helpful way to think of the Web is as a set of cooperating clients and servers, all of whom speak the same language: HTTP.
 - Most people are exposed to the Web through a graphical client program, or Web browser, like Safari, Chrome, Firefox or Internet Explorer.

Traditional Applications

- World Wide Web
 - Clearly, if you want to organize information into a system of linked documents or objects, you need to be able to retrieve one document to get started.
 - Hence, any Web browser has a function that allows the user to obtain an object by “opening a URL.”
 - URLs (Uniform Resource Locators) are so familiar to most of us by now that it’s easy to forget that they haven’t been around forever.
 - They provide information that allows objects on the Web to be located, and they look like the following:
 - <http://www.cs.princeton.edu/index.html>

Traditional Applications

- World Wide Web
 - If you opened that particular URL, your Web browser would open a TCP connection to the Web server at a machine called `www.cs.princeton.edu` and immediately retrieve and display the file called `index.html`.
 - Most files on the Web contain images and text and many have other objects such as audio and video clips, pieces of code, etc.
 - They also frequently include URLs that point to other files that may be located on other machines, which is the core of the “hypertext” part of HTTP and HTML.

Traditional Applications

■ World Wide Web

- When you ask your browser to view a page, your browser (the client) fetches the page from the server using HTTP running over TCP.
- Like SMTP, HTTP is a text oriented protocol.
- At its core, HTTP is a request/response protocol, where every message has the general form

```
START_LINE <CRLF>
MESSAGE_HEADER <CRLF>
<CRLF>
MESSAGE_BODY <CRLF>
```
- where as before, <CRLF> stands for carriage-return-line-feed. The first line (START LINE)
- indicates whether this is a request message or a response message.

Traditional Applications

- World Wide Web
 - Request Messages
 - The first line of an HTTP request message specifies three things: the operation to be performed, the Web page the operation should be performed on, and the version of HTTP being used.
 - Although HTTP defines a wide assortment of possible request operations—including “write” operations that allow a Web page to be posted on a server—the two most common operations are GET (fetch the specified Web page) and HEAD (fetch status information about the specified Web page).

Traditional Applications

- World Wide Web
 - Request Messages

Operation	Description
OPTIONS	Request information about available options
GET	Retrieve document identified in URL
HEAD	Retrieve metainformation about document identified in URL
POST	Give information (e.g., annotation) to server
PUT	Store document under specified URL
DELETE	Delete specified URL
TRACE	Loopback request message
CONNECT	For use by proxies

HTTP request operations

Traditional Applications

- World Wide Web
 - Response Messages
 - Like request messages, response messages begin with a single START LINE.
 - In this case, the line specifies the version of HTTP being used, a three-digit code indicating whether or not the request was successful, and a text string giving the reason for the response.

Traditional Applications

- World Wide Web
 - Response Messages

Code	Type	Example Reasons
1xx	Informational	request received, continuing process
2xx	Success	action successfully received, understood, and accepted
3xx	Redirection	further action must be taken to complete the request
4xx	Client Error	request contains bad syntax or cannot be fulfilled
5xx	Server Error	server failed to fulfill an apparently valid request

Five types of HTTP result codes

Traditional Applications

- World Wide Web
 - Uniform Resource Identifiers
 - The URLs that HTTP uses as addresses are one type of *Uniform Resource Identifier (URI)*.
 - A URI is a character string that identifies a resource, where a resource can be anything that has identity, such as a document, an image, or a service.
 - The format of URIs allows various more-specialized kinds of resource identifiers to be incorporated into the URI space of identifiers.
 - The first part of a URI is a *scheme* that names a particular way of identifying a certain kind of resource, such as mailto for email addresses or file for file names.
 - The second part of a URI, separated from the first part by a colon, is the *scheme-specific part*.

Traditional Applications

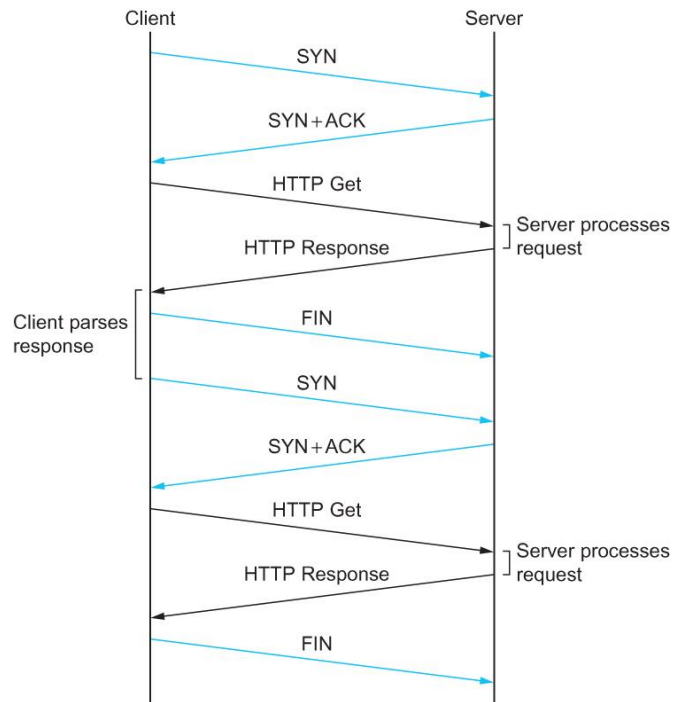
- World Wide Web
 - TCP Connections
 - The original version of HTTP (1.0) established a separate TCP connection for each data item retrieved from the server.
 - It's not too hard to see how this was a very inefficient mechanism: connection setup and teardown messages had to be exchanged between the client and server even if all the client wanted to do was verify that it had the most recent copy of a page.
 - Thus, retrieving a page that included some text and a dozen icons or other small graphics would result in 13 separate TCP connections being established and closed.

Traditional Applications

- World Wide Web
 - TCP Connections
 - To overcome this situation, HTTP version 1.1 introduced *persistent connections*— the client and server can exchange multiple request/response messages over the same TCP connection.
 - Persistent connections have many advantages.
 - First, they obviously eliminate the connection setup overhead, thereby reducing the load on the server, the load on the network caused by the additional TCP packets, and the delay perceived by the user.
 - Second, because a client can send multiple request messages down a single TCP connection, TCP's congestion window mechanism is able to operate more efficiently.
 - This is because it's not necessary to go through the slow start phase for each page.

Traditional Applications

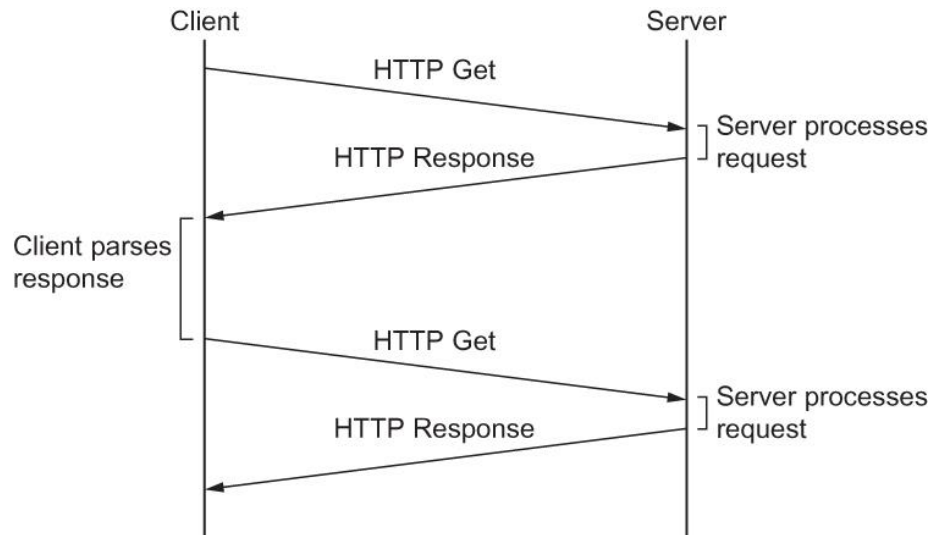
- World Wide Web
 - TCP Connections



HTTP 1.0 behavior

Traditional Applications

- World Wide Web
 - TCP Connections



HTTP 1.1 behavior with persistent connections

Traditional Applications

- World Wide Web
 - Caching
 - One of the most active areas of research (and entrepreneurship) in the Internet today is how to effectively cache Web pages.
 - Caching has many benefits. From the client's perspective, a page that can be retrieved from a nearby cache can be displayed much more quickly than if it has to be fetched from across the world.
 - From the server's perspective, having a cache intercept and satisfy a request reduces the load on the server.

Traditional Applications

- World Wide Web

- Caching

- Caching can be implemented in many different places. For example, a user's browser can cache recently accessed pages, and simply display the cached copy if the user visits the same page again.
 - As another example, a site can support a single site-wide cache.
 - This allows users to take advantage of pages previously downloaded by other users.
 - Closer to the middle of the Internet, ISPs can cache pages.
 - Note that in the second case, the users within the site most likely know what machine is caching pages on behalf of the site, and they configure their browsers to connect directly to the caching host. This node is sometimes called a *proxy*

Traditional Applications

- Web Services
 - Much of the motivation for enabling direct application-to-application communication comes from the business world.
 - Historically, interactions between enterprises—businesses or other organizations—have involved some manual steps such as filling out an order form or making a phone call to determine whether some product is in stock.
 - Even within a single enterprise it is common to have manual steps between software systems that cannot interact directly because they were developed independently.

Traditional Applications

- Web Services
 - Increasingly such manual interactions are being replaced with direct application-to application interaction.
 - An ordering application at enterprise A would send a message to an order fulfillment application at enterprise B, which would respond immediately indicating whether the order can be filled.
 - Perhaps, if the order cannot be filled by B, the application at A would immediately order from another supplier, or solicit bids from a collection of suppliers.

Traditional Applications

- Web Services
 - Two architectures have been advocated as solutions to this problem.
 - Both architectures are called *Web Services*, taking their name from the term for the individual applications that offer a remotely-accessible service to client applications to form network applications.
 - The terms used as informal shorthand to distinguish the two Web Services architectures are *SOAP* and *REST* (as in, “the SOAP vs. REST debate”).

Traditional Applications

- Web Services
 - The SOAP architecture's approach to the problem is to make it feasible, at least in theory, to generate protocols that are customized to each network application.
 - The key elements of the approach are a framework for protocol specification, software toolkits for automatically generating protocol implementations from the specifications, and modular partial specifications that can be reused across protocols.

Traditional Applications

- Web Services
 - The REST architecture's approach to the problem is to regard individual Web Services as World Wide Web resources—identified by URIs and accessed via HTTP.
 - Essentially, the REST architecture is just the Web architecture.
 - The Web architecture's strengths include stability and a demonstrated scalability (in the network-size sense).

Traditional Applications

- Custom Application Protocols (WSDL, SOAP)
 - The architecture informally referred to as SOAP is based on *Web Services Description Language (WSDL) and SOAP.4*
 - Both of these standards are issued by the World Wide Web Consortium (W3C).
 - This is the architecture that people usually mean when they use the term Web Services without any preceding qualifier.

Multimedia Applications

- Just like the traditional applications described earlier in this chapter, multimedia applications such as telephony and videoconferencing need their own protocols.
- We have already seen a number of protocols that multimedia applications use.
- The Real-Time Transport Protocol (RTP) provides many of the functions that are common to multimedia applications such as conveying timing information and identifying the coding schemes and media types of an application.

Multimedia Applications

- The Resource Reservation Protocol, RSVP can be used to request the allocation of resources in the network so that the desired quality of service (QoS) can be provided to an application.
- In addition to these protocols for multimedia transport and resource allocation, many multimedia applications also need a signalling or *session control protocol*.
- For example, suppose that we wanted to be able to make telephone calls across the internet (“voice over IP” or VOIP).

Multimedia Applications

- Session Control and Call Control (SDP, SIP, H.323)
 - To understand some of the issues of session control, consider the following problem.
 - Suppose you want to hold a videoconference at a certain time and make it available to a wide number of participants. Perhaps you have decided to encode the video stream using the MPEG-2 standard, to use the multicast IP address 224.1.1.1 for transmission of the data, and to send it using RTP over UDP port number 4000.
 - How would you make all that information available to the intended participants?
 - One way would be to put all that information in an email and send it out, but ideally there should be a standard format and protocol for disseminating this sort of information.

Multimedia Applications

- Session Control and Call Control (SDP, SIP, H.323)
 - The IETF has defined protocols for just this purpose. The protocols that have been defined include
 - SDP (Session Description Protocol)
 - SAP (Session Announcement Protocol)
 - SIP (Session Initiation Protocol)
 - SCCP (Simple Conference Control Protocol)

Multimedia Applications

- Session Description Protocol (SDP)
 - The Session Description Protocol (SDP) is a rather general protocol that can be used in a variety of situations and is typically used in conjunction with one or more other protocols (e.g., SIP).
 - It conveys the following information:
 - The name and purpose of the session
 - Start and end times for the session
 - The media types (e.g. audio, video) that comprise the session
 - Detailed information needed to receive the session (e.g. the multicast address to which data will be sent, the transport protocol to be used, the port numbers, the encoding scheme, etc.)

Multimedia Applications

- SIP
 - SIP is an application layer protocol that bears a certain resemblance to HTTP, being based on a similar request/response model.
 - However, it is designed with rather different sorts of applications in mind, and thus provides quite different capabilities than HTTP.

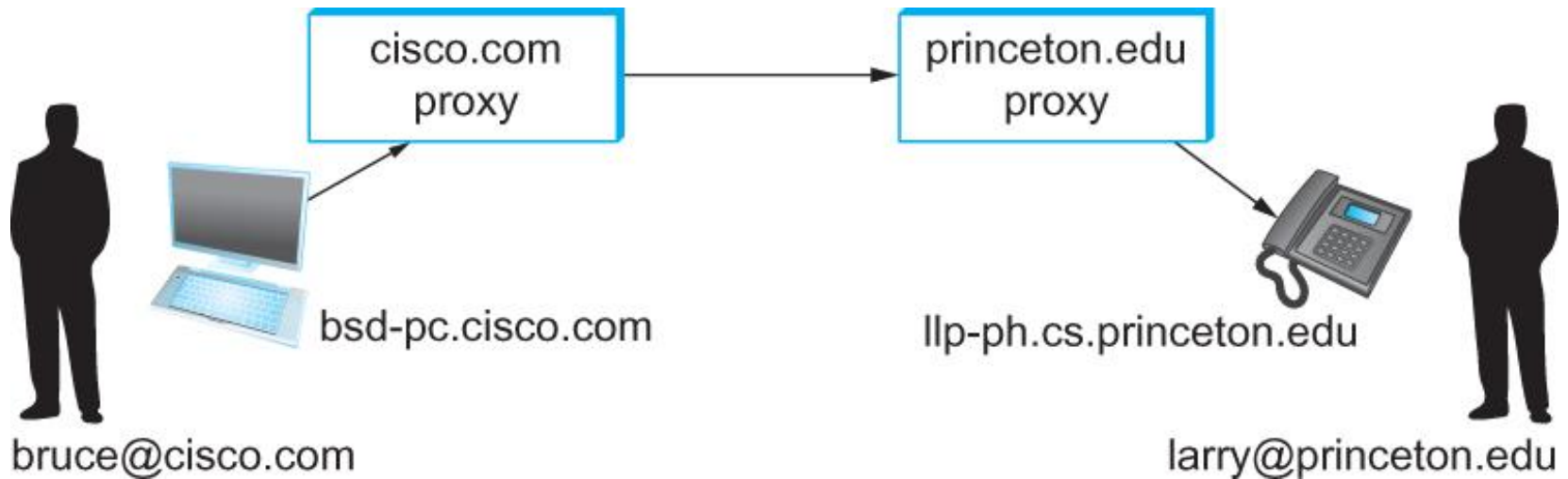
Multimedia Applications

■ SIP

- The capabilities provided by SIP can be grouped into five categories:
 - User location: determining the correct device with which to communicate to reach a particular user;
 - User availability: determining if the user is willing or able to take part in a particular communication session;
 - User capabilities: determining such items as the choice of media and coding scheme to use;
 - Session setup: establishing session parameters such as port numbers to be used by the communicating parties;
 - Session management: a range of functions including transferring sessions (e.g. to implement “call forwarding”) and modifying session parameters.

Multimedia Applications

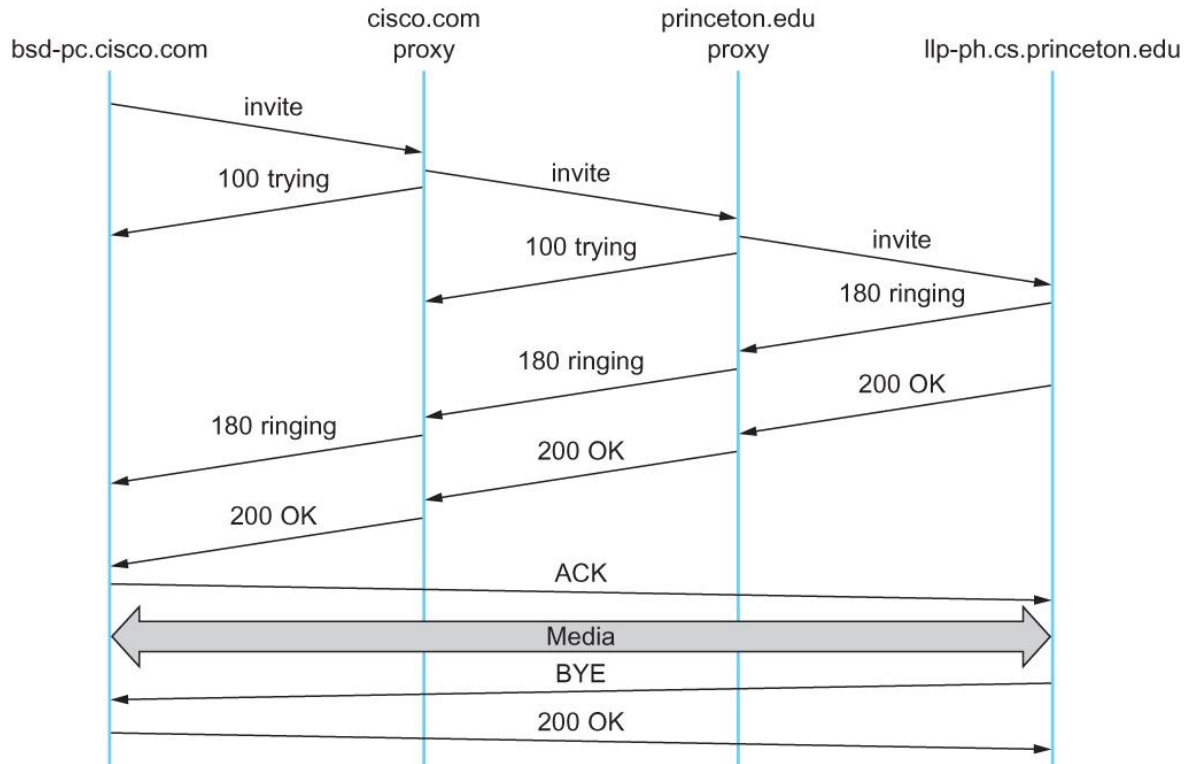
- SIP



Establishing communication through SIP proxies.

Multimedia Applications

- SIP



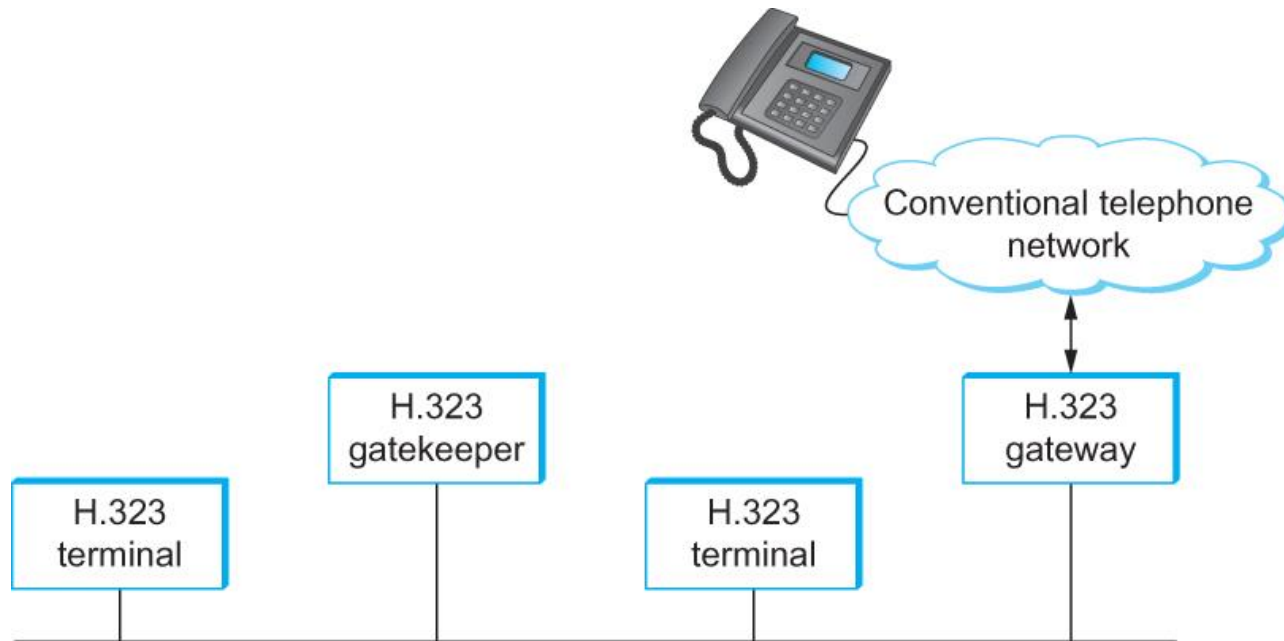
Message flow for a basic SIP session

Multimedia Applications

- H.323
 - The ITU has also been very active in the call control area, which is not surprising given its relevance to telephony, the traditional realm of that body.
 - Fortunately, there has been considerable coordination between the IETF and the ITU in this instance, so that the various protocols are somewhat interoperable.
 - The major ITU recommendation for multimedia communication over packet networks is known as H.323, which ties together many other recommendations, including H.225 for call control.
 - The full set of recommendations covered by H.323 runs to many hundreds of pages, and the protocol is known for its complexity

Multimedia Applications

- H.323



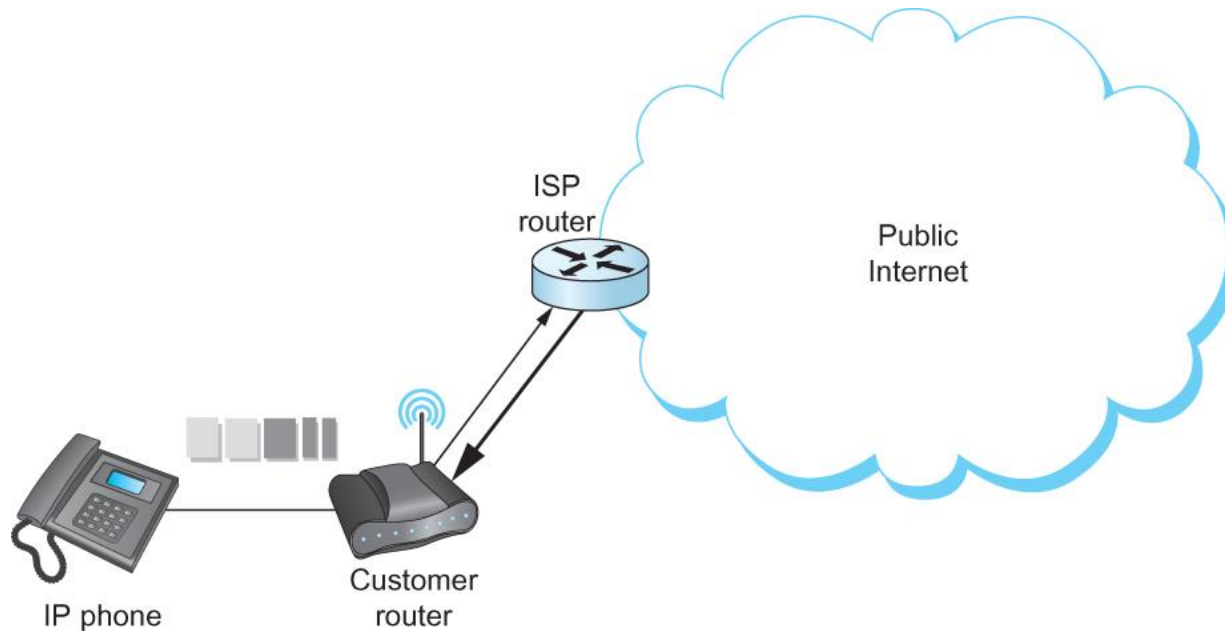
Devices in an H.323 network.

Multimedia Applications

- Resource Allocation for Multimedia Applications
 - As we have just seen, session control protocols like SIP and H.323 can be used to initiate and control communication in multimedia applications, while RTP provides transport level functions for the data streams of the applications.
 - A final piece of the puzzle in getting multimedia applications to work is making sure that suitable resources are allocated inside the network to ensure that the quality of service needs of the application are met.
 - Differentiated Services can be used to provide fairly basic and scalable resource allocation to applications.
 - A multimedia application can set the DSCP (differentiated services code point) in the IP header of the packets that it generates in an effort to ensure that both the media and control packets receive appropriate quality of service.

Multimedia Applications

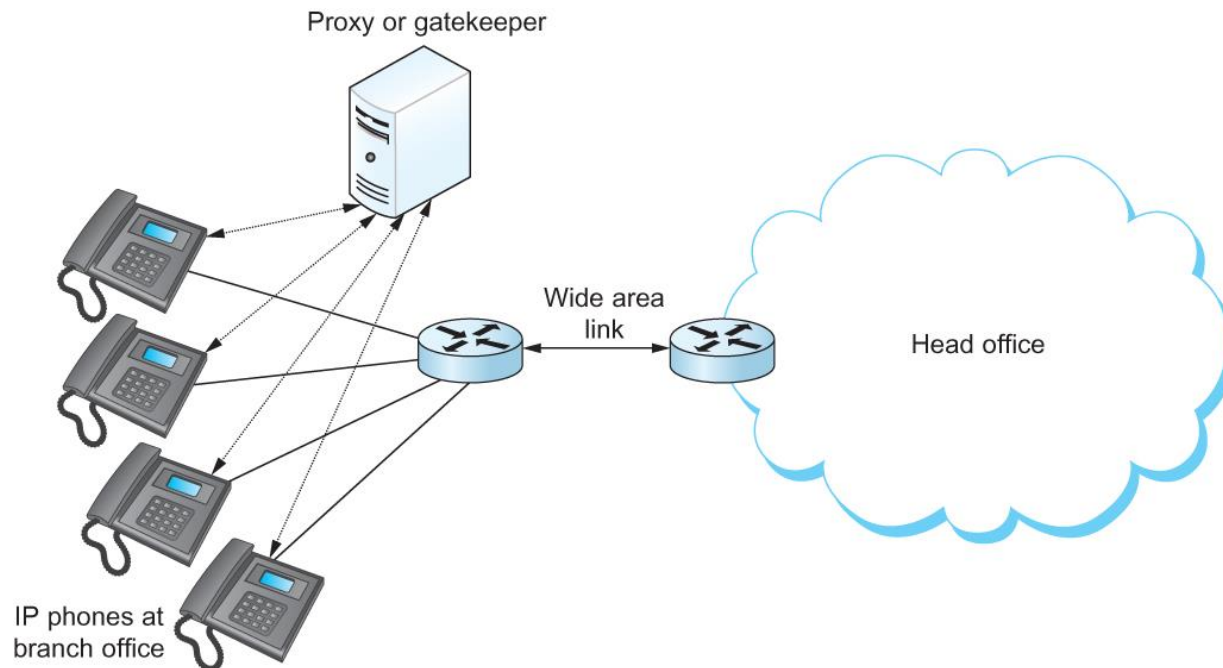
- Resource Allocation for Multimedia Applications



Differentiated Services applied to a VOIP application. DiffServ queueing is applied only on the upstream link from customer router to ISP.

Multimedia Applications

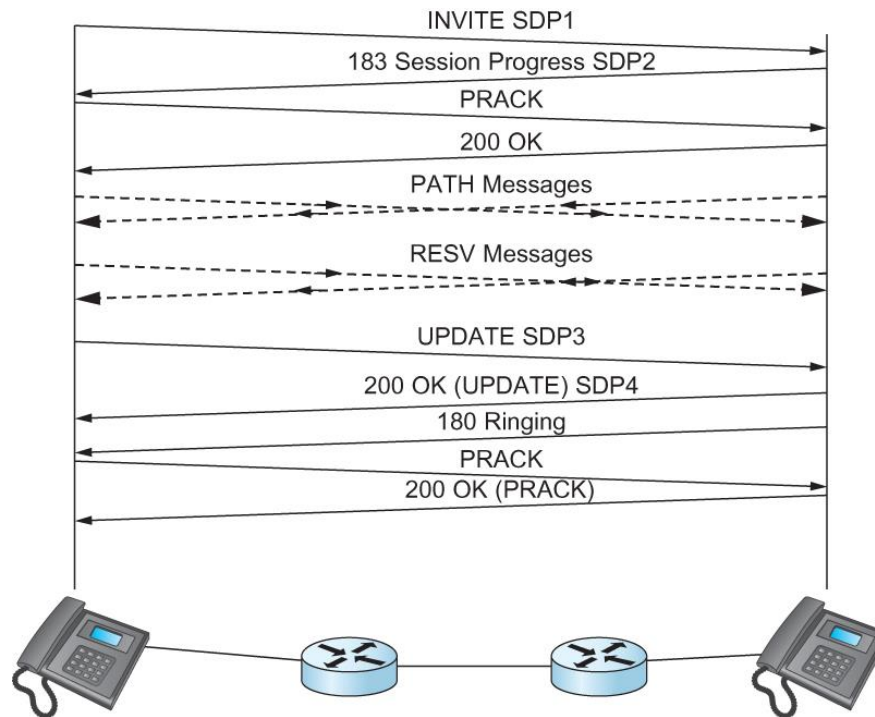
- Resource Allocation for Multimedia Applications



Admission control using session control protocol.

Multimedia Applications

- Resource Allocation for Multimedia Applications



Co-ordination of SIP signalling and resource reservation.

Infrastructure Services

- There are some protocols that are essential to the smooth running of the Internet, but that don't fit neatly into the strictly layered model.
- One of these is the Domain Name System (DNS)—not an application that users normally invoke explicitly, but rather a service that almost all other applications depend upon.
- This is because the name service is used to translate host names into host addresses; the existence of such an application allows the users of other applications to refer to remote hosts by name rather than by address.

Infrastructure Services

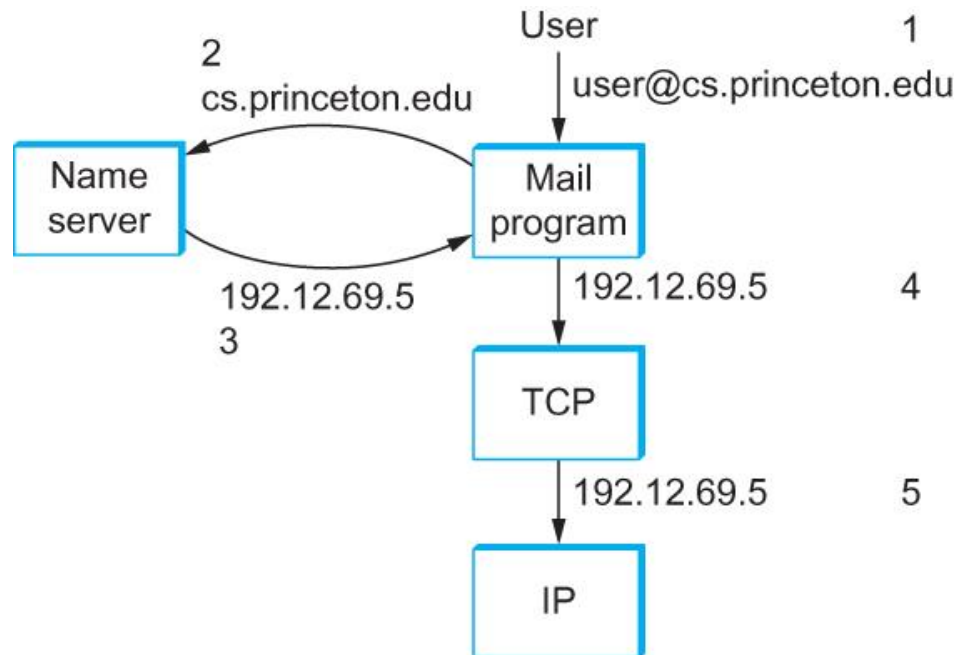
- Name Service (DNS)
 - In most of this book, we have been using addresses to identify hosts.
 - While perfectly suited for processing by routers, addresses are not exactly user-friendly.
 - It is for this reason that a unique *name is also typically assigned to each host in a network.*
 - Host names differ from host addresses in two important ways.
 - First, they are usually of variable length and mnemonic, thereby making them easier for humans to remember.
 - Second, names typically contain no information that helps the network locate (route packets toward) the host.

Infrastructure Services

- Name Service (DNS)
 - We first introduce some basic terminology.
 - First, a *name space* defines the set of possible names.
 - A name space can be either *flat* (names are not divisible into components), or it can be *hierarchical* (Unix file names are an obvious example).
 - Second, the *naming system* maintains a collection of *bindings of names to values*. The value can be anything we want the naming system to return when presented with a name; in many cases it is an address.
 - Finally, a *resolution mechanism* is a procedure that, when invoked with a name, returns the corresponding value. A *name server* is a specific *implementation of a resolution mechanism* that is available on a network and that can be queried by sending it a message.

Infrastructure Services

- Name Service (DNS)



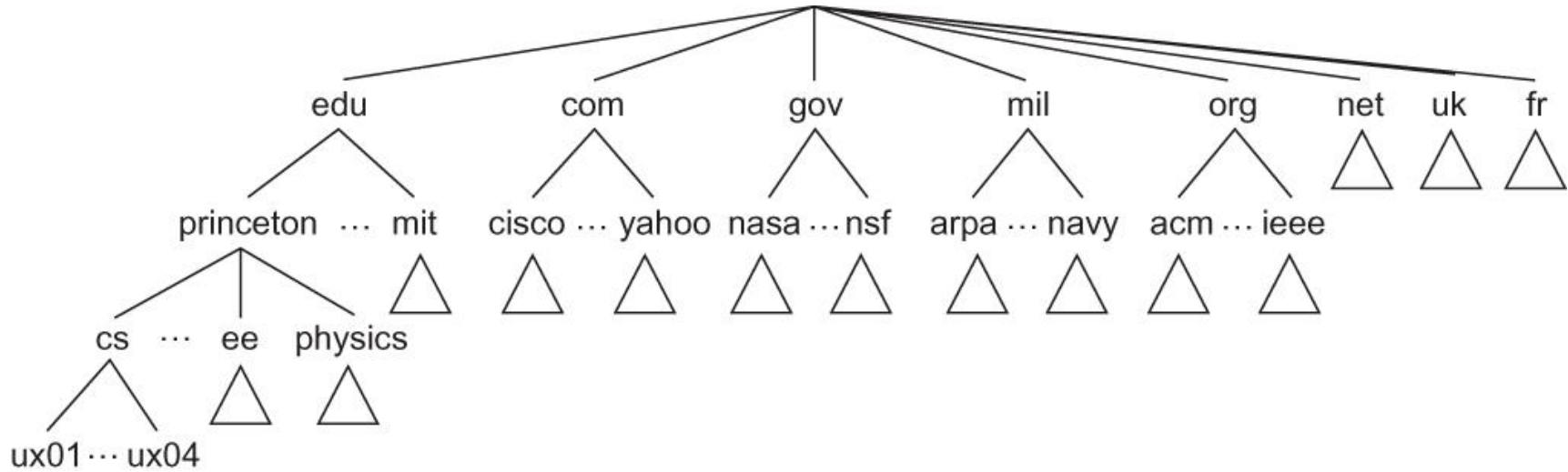
Names translated into addresses, where the numbers 1–5 show the sequence of steps in the process

Infrastructure Services

- Domain Hierarchy
 - DNS implements a hierarchical name space for Internet objects. Unlike Unix file names, which are processed from left to right with the naming components separated with slashes, DNS names are processed from right to left and use periods as the separator.
 - Like the Unix file hierarchy, the DNS hierarchy can be visualized as a tree, where each node in the tree corresponds to a domain, and the leaves in the tree correspond to the hosts being named.

Infrastructure Services

■ Domain Hierarchy



Example of a domain hierarchy

Infrastructure Services

■ Name Servers

- The complete domain name hierarchy exists only in the abstract.
- We now turn our attention to the question of how this hierarchy is actually implemented.
- The first step is to partition the hierarchy into subtrees called *zones*.
- Each zone can be thought of as corresponding to some administrative authority that is responsible for that portion of the hierarchy.
- For example, the top level of the hierarchy forms a zone that is managed by the Internet Corporation for Assigned Names and Numbers (ICANN).

Infrastructure Services

- Name Servers
 - Each name server implements the zone information as a collection of *resource records*.
 - In essence, a resource record is a name-to-value binding, or more specifically a 5-tuple that contains the following fields:
 - <Name, Value, Type, Class, TTL >

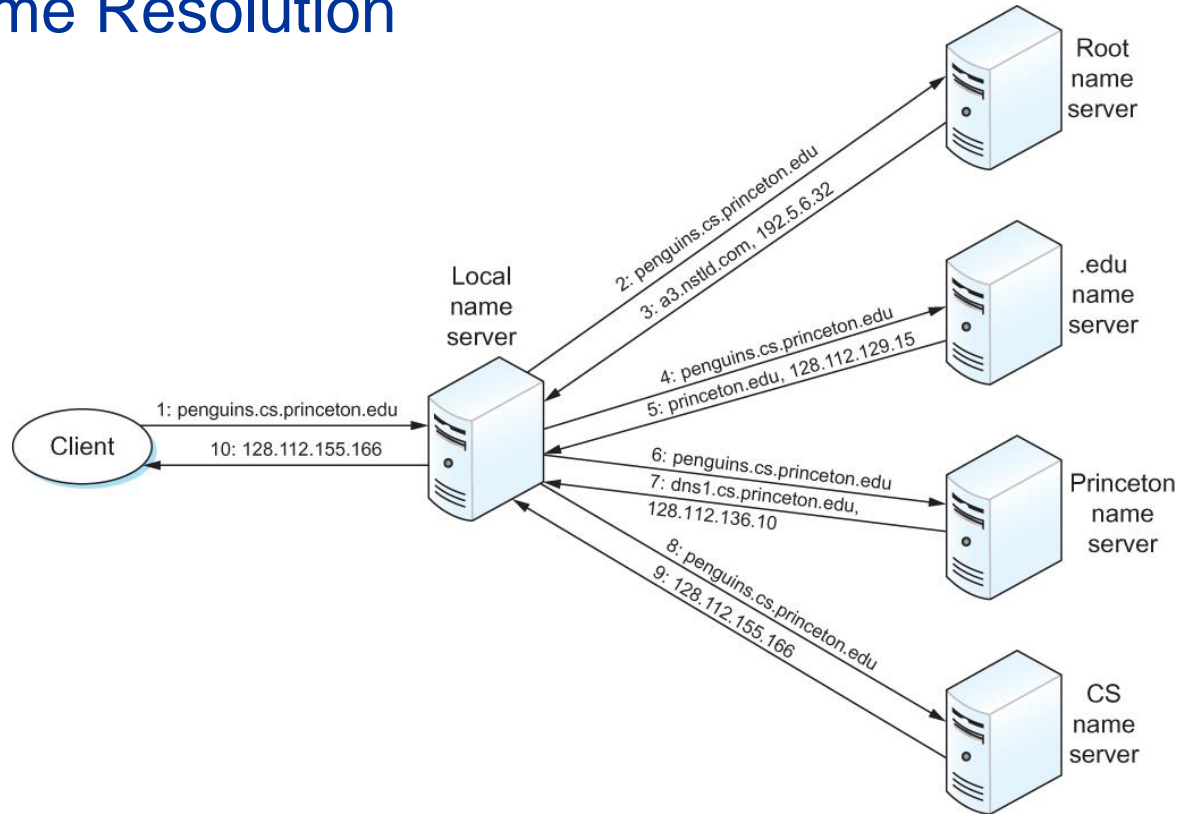
Infrastructure Services

■ Name Servers

- The Name and Value fields are exactly what you would expect, while the Type field specifies how the Value should be interpreted.
 - For example, Type = A indicates that the Value is an IP address. Thus, A records implement the name-to-address mapping we have been assuming. Other record types include
 - NS: The Value field gives the domain name for a host that is running a name server that knows how to resolve names within the specified domain.
 - CNAME: The Value field gives the canonical name for a particular host; it is used to define aliases.
 - MX: The Value field gives the domain name for a host that is running a mail server that accepts messages for the specified domain.

Infrastructure Services

■ Name Resolution



Name resolution in practice, where the numbers 1–10 show the sequence of steps in the process.

Infrastructure Services

- Network Management
 - A network is a complex system, both in terms of the number of nodes that are involved and in terms of the suite of protocols that can be running on any one node.
 - Even if you restrict yourself to worrying about the nodes within a single administrative domain, such as a campus, there might be dozens of routers and hundreds—or even thousands—of hosts to keep track of. If you think about all the state that is maintained and manipulated on any one of those nodes—for example, address translation tables, routing tables, TCP connection state, and so on—then it is easy to become depressed about the prospect of having to manage all of this information

Infrastructure Services

- Network Management
 - The most widely used protocol for this purpose is the Simple Network Management Protocol (SNMP).
 - SNMP is essentially a specialized request/reply protocol that supports two kinds of request messages: GET and SET.
 - The former is used to retrieve a piece of state from some node, and the latter is used to store a new piece of state in some node.
 - SNMP is used in the obvious way.
 - A system administrator interacts with a client program that displays information about the network.
 - This client program usually has a graphical interface. Whenever the administrator selects a certain piece of information that he or she wants to see, the client program uses SNMP to request that information from the node in question. (SNMP runs on top of UDP.)
 - An SNMP server running on that node receives the request, locates the appropriate piece of information, and returns it to the client program, which then displays it to the user.

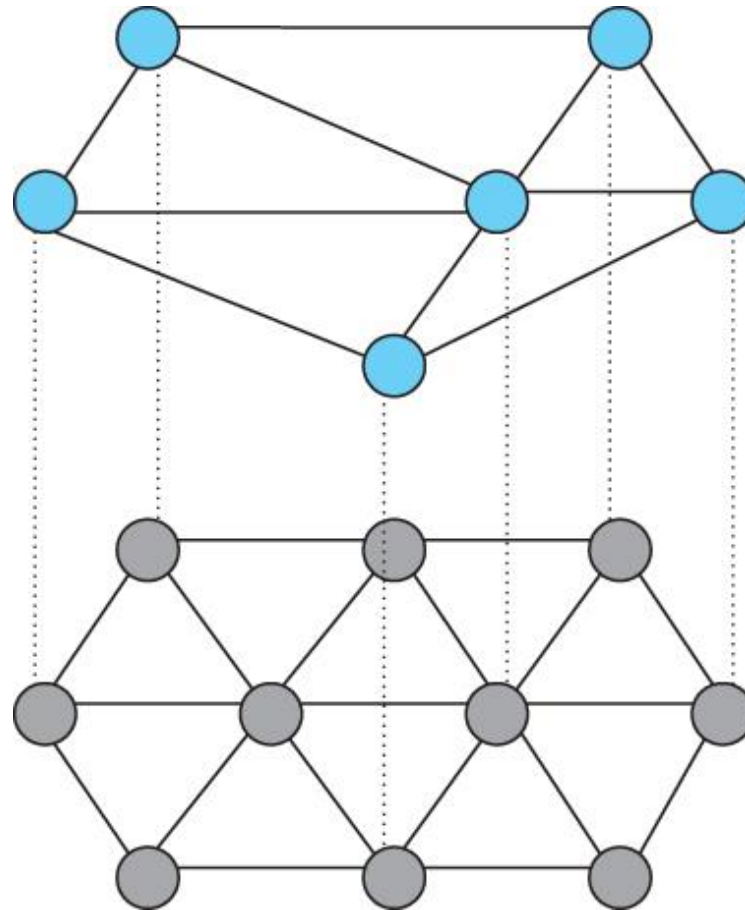
Overlay Network

- In the last few years, the distinction between *packet forwarding and application processing* has become less clear.
- New applications are being distributed across the Internet, and in many cases, these applications make their own forwarding decisions.
- These new hybrid applications can sometimes be implemented by extending traditional routers and switches to support a modest amount of application-specific processing.
- For example, so called *level-7 switches* sit in front of server clusters and forward HTTP requests to a specific server based on the requested URL.

Overlay Network

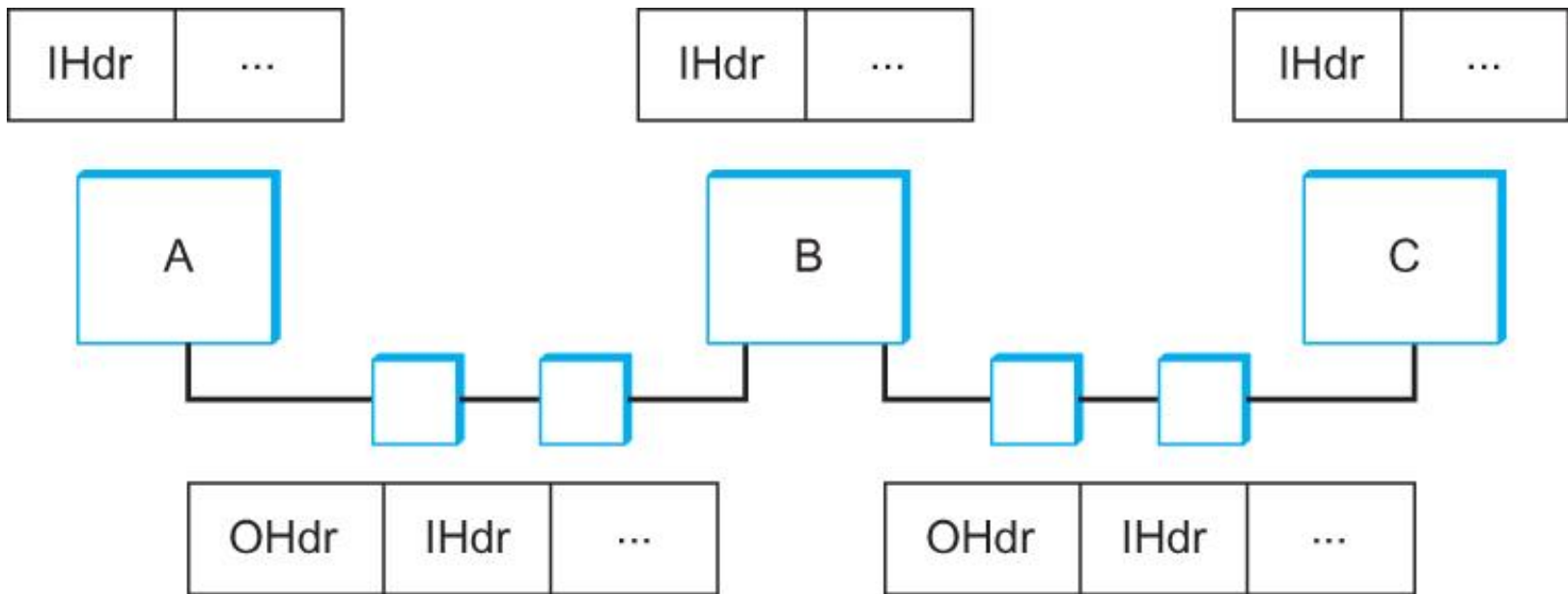
- However, *overlay networks are quickly emerging as the mechanism of choice for introducing new functionality into the Internet*
- You can think of an overlay as a logical network implemented on top of a some underlying network.
 - By this definition, the Internet started out as an overlay network on top of the links provided by the old telephone network
- Each node in the overlay also exists in the underlying network; it processes and forwards packets in an application-specific way.
- The links that connect the overlay nodes are implemented as tunnels through the underlying network.

Overlay Network



Overlay network layered on top of a physical network

Overlay Network



Overlay nodes tunnel through physical nodes

Overlay Network

- Routing Overlays
 - The simplest kind of overlay is one that exists purely to support an alternative routing strategy; no additional application-level processing is performed at the overlay nodes.
 - You can view a virtual private network as an example of a routing overlay.
 - In this particular case, the overlay is said to use “IP tunnels”, and the ability to utilize these VPNs is supported in many commercial routers.

Overlay Network

- Routing Overlays
 - Suppose, however, you wanted to use a routing algorithm that commercial router vendors were not willing to include in their products.
 - How would you go about doing it?
 - You could simply run your algorithm on a collection of end hosts, and tunnel through the Internet routers.
 - These hosts would behave like routers in the overlay network: as hosts they are probably connected to the Internet by only one physical link, but as a node in the overlay they would be connected to multiple neighbors via tunnels.

Overlay Network

- Routing Overlays
 - Experimental Versions of IP
 - Overlays are ideal for deploying experimental versions of IP that you hope will eventually take over the world.
 - For example, IP multicast started off as an extension to IP and even today is not enabled in many Internet routers.
 - The Mbone (multicast backbone) was an overlay network that implemented IP multicast on top of the unicast routing provided by the Internet.
 - A number of multimedia conference tools were developed for and deployed on the Mbone.
 - For example, IETF meetings—which are a week long and attract thousands of participants—were for many years broadcast over the MBone.

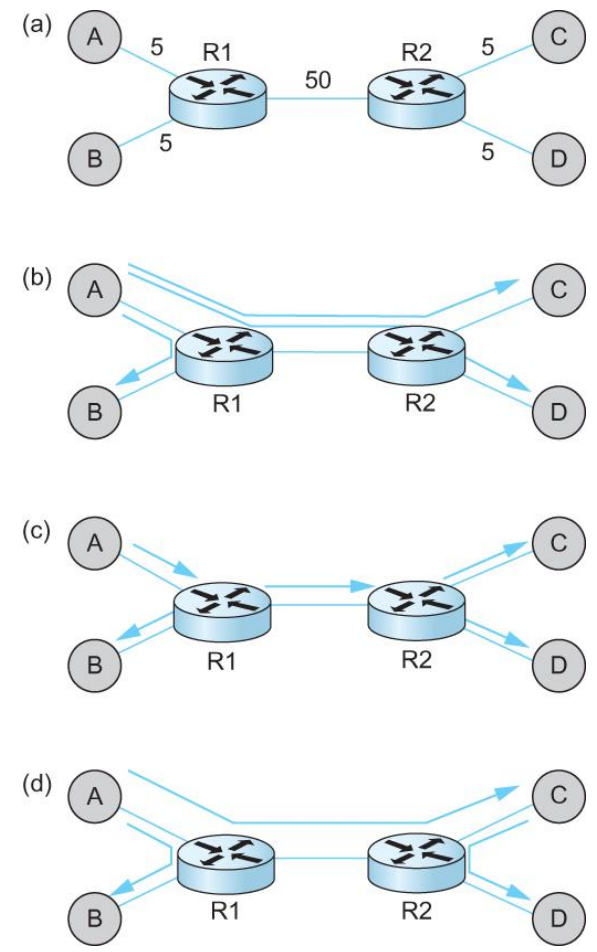
Overlay Network

- Routing Overlays
 - End System Multicast
 - Although IP multicast is popular with researchers and certain segments of the networking community, its deployment in the global internet has been limited at best.
 - In response, multicast-based applications like videoconferencing have recently turned to an alternative strategy, called *end system multicast*.
 - The idea of end system multicast is to accept that IP multicast will never become ubiquitous, and to instead let the end hosts that are participating in a particular multicast-based application implement their own multicast trees.

Overlay Network

- Routing Overlays
 - End System Multicast

(a) depicts an example physical topology, where R1 and R2 are routers connected by a low-bandwidth transcontinental link; A, B, C, and D are end hosts; and link delays are given as edge weights. Assuming A wants to send a multicast message to the other three hosts, (b) shows how naive unicast transmission would work. This is clearly undesirable because the same message must traverse the link A–R1 three times, and two copies of the message traverse R1–R2. (c) depicts the IP multicast tree constructed by DVMRP. Clearly, this approach eliminates the redundant messages. Without support from the routers, however, the best one can hope for with end system multicast is a tree similar to the one shown in (d). End system multicast defines an architecture for constructing this tree.

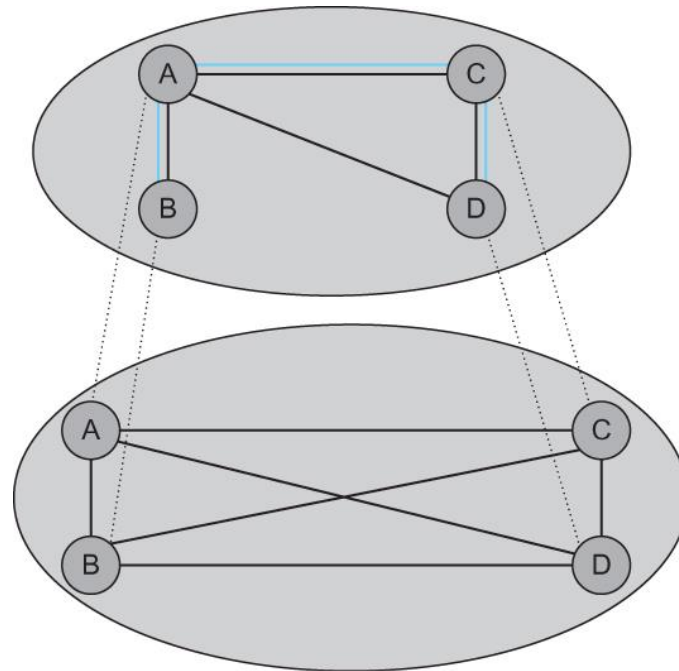


Overlay Network

- Routing Overlays
 - End System Multicast
 - The general approach is to support multiple levels of overlay networks, each of which extracts a subgraph from the overlay below it, until we have selected the subgraph that the application expects.
 - For end system multicast in particular, this happens in two stages: first we construct a simple *mesh overlay on top of the fully connected* Internet, and then we select a multicast tree within this mesh.

Overlay Network

- Routing Overlays
 - End System Multicast



Multicast tree embedded in an overlay mesh

Overlay Network

- Resilient Overlay Networks
 - Another function that can be performed by an overlay is to find alternative routes for traditional unicast applications.
 - Such overlays exploit the observation that the triangle inequality does not hold in the Internet

Overlay Network

- Peer-to-peer Networks
 - Music-sharing applications like Napster and KaZaA introduced the term “peer-to-peer” into the popular vernacular.
 - Attributes like *decentralized and self-organizing* are mentioned when discussing peer-to-peer networks, meaning that individual nodes organize themselves into a network without any centralized coordination

Overlay Network

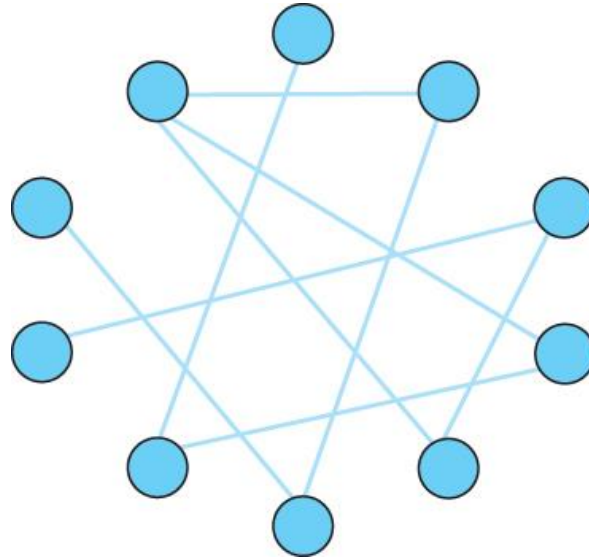
- Peer-to-peer Networks
 - What's interesting about peer-to-peer networks?
 - One answer is that both the process of locating an object of interest and the process of downloading that object onto your local machine happen without your having to contact a centralized authority, and at the same time, the system is able to scale to millions of nodes.
 - A peer-to-peer system that can accomplish these two tasks in a decentralized manner turns out to be an overlay network, where the nodes are those hosts that are willing to share objects of interest (e.g., music and other assorted files), and the links (tunnels) connecting these nodes represent the sequence of machines that you have to visit to track down the object you want.

Overlay Network

- Peer-to-peer Networks
 - Gnutella
 - Gnutella is an early peer-to-peer network that attempted to distinguish between exchanging music (which likely violates somebody's copyright) and the general sharing of files (which must be good since we've been taught to share since the age of two).
 - What's interesting about Gnutella is that it was one of the first such systems to not depend on a centralized registry of objects.
 - Instead Gnutella participants arrange themselves into an overlay network.

Overlay Network

- Peer-to-peer Networks
 - Gnutella



Example topology of a Gnutella peer-to-peer network

Overlay Network

- Peer-to-peer Networks
 - Structured Overlays
 - At the same time file sharing systems have been fighting to fill the void left by Napster, the research community has been exploring an alternative design for peer-to-peer networks.
 - We refer to these networks as *structured*, to contrast them with the essentially random (unstructured) way in which a Gnutella network evolves.
 - Unstructured overlays like Gnutella employ trivial overlay construction and maintenance algorithms, but the best they can offer is unreliable, random search.
 - In contrast, structured overlays are designed to conform to a particular graph structure that allows reliable and efficient object location, in return for additional complexity during overlay construction and maintenance..

Overlay Network

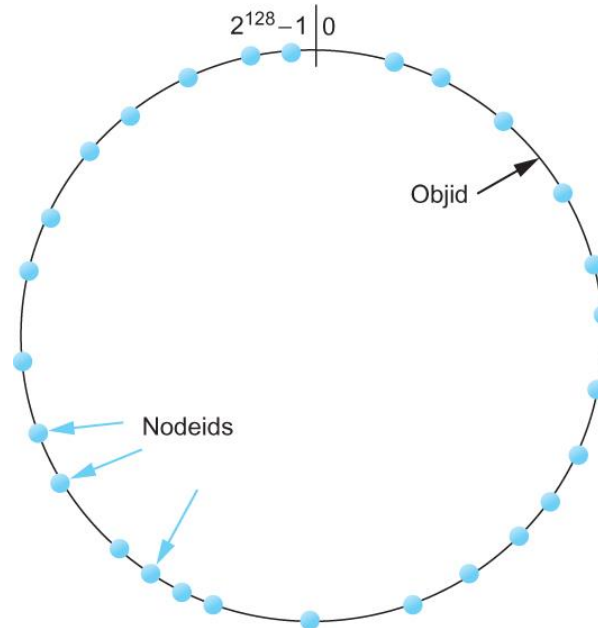
- Peer-to-peer Networks
 - Structured Overlays
 - If you think about what we are trying to do at a high level, there are two questions to consider:
 - (1) how do we map objects onto nodes, and
 - (2) how do we route a request to the node that is responsible for a given object.
 - We start with the first question, which has a simple statement: how do we map an object with name x into the address of some node n that is able to serve that object?
 - While traditional peer-to-peer networks have no control over which node hosts object x , if we could control how objects get distributed over the network, we might be able to do a better job of finding those objects at a later time.

Overlay Network

- Peer-to-peer Networks
 - Structured Overlays
 - A well-known technique for mapping names into address is to use a hash table, so that
 - $\text{hash}(x) \rightarrow n$
 - implies object x is first placed on node n , and at a later time, a client trying to locate x would only have to perform the hash of x to determine that it is on node n .
 - A hash-based approach has the nice property that it tends to spread the objects evenly across the set of nodes, but straightforward hashing algorithms suffer from a fatal flaw: how many possible values of n should we allow?
 - Naively, we could decide that there are, say, 101 possible hash values, and we use a modulo hash function; that is,
 - $\text{hash}(x)$
 - $\text{return } x \% 101.$

Overlay Network

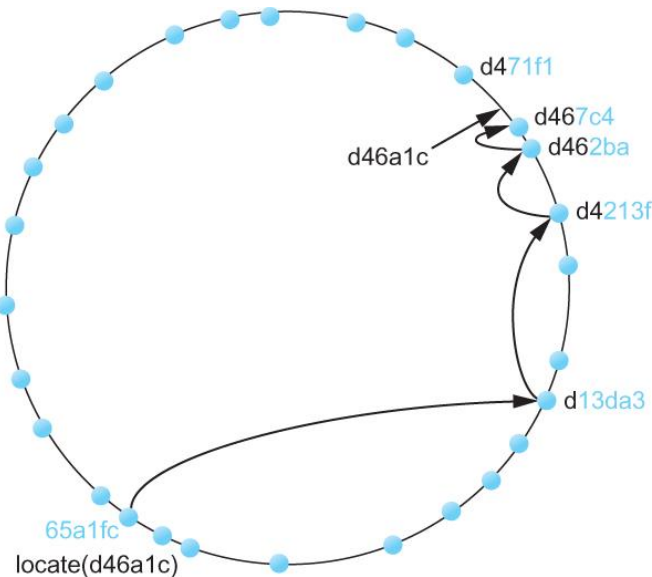
- Peer-to-peer Networks
 - Structured Overlays



Both nodes and objects map (hash) onto the id space, where objects are maintained at the nearest node in this space.

Overlay Network

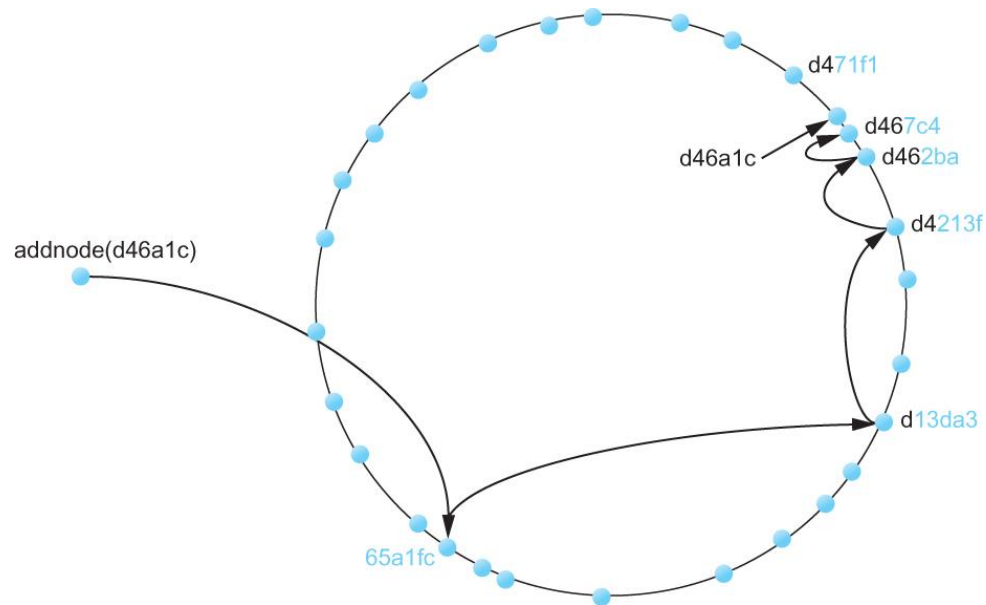
- Peer-to-peer Networks
 - Structured Overlays



Objects are located by routing through the peer-to-peer overlay network.

Overlay Network

- Peer-to-peer Networks
 - Structured Overlays



Adding a node to the network

Overlay Network

- Peer-to-peer Networks

- BitTorrent

- BitTorrent is a peer-to-peer file sharing protocol devised by Bram Cohen.
 - It is based on replicating the file, or rather, replicating segments of the file, which are called *pieces*.
 - Any particular piece can usually be downloaded from multiple peers, even if only one peer has the entire file.
 - The primary benefit of BitTorrent's replication is avoiding the bottleneck of having only one source for a file. This is particularly useful when you consider that any given computer has a limited speed at which it can serve files over its uplink to the Internet, often quite a low limit due to the asymmetric nature of most broadband networks.

Overlay Network

- Peer-to-peer Networks

- BitTorrent

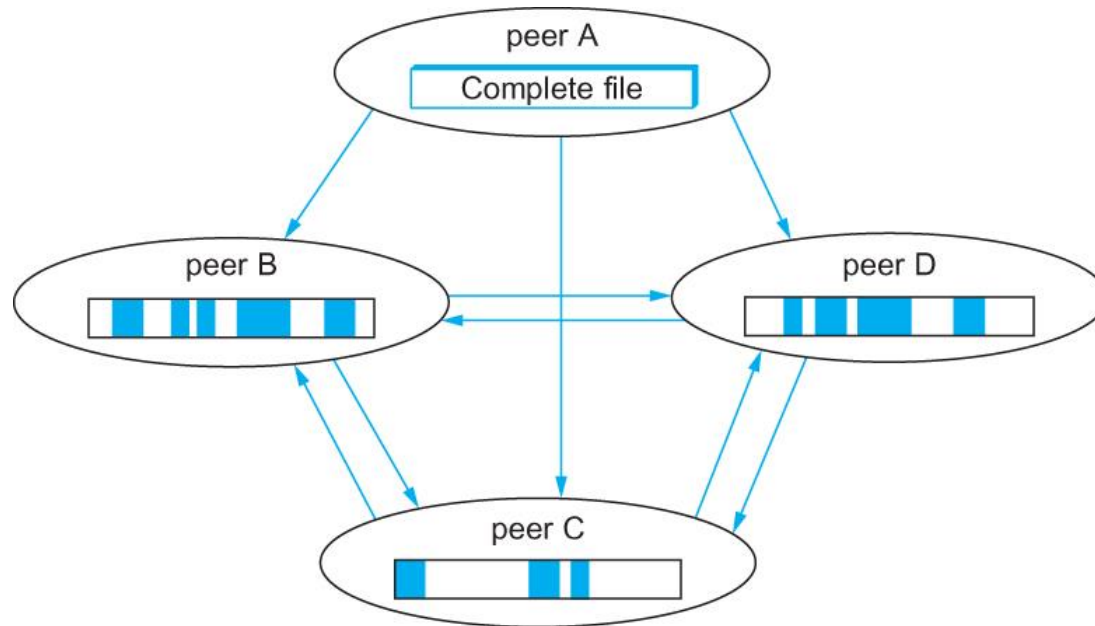
- The beauty of BitTorrent is that replication is a natural side-effect of the downloading process: as soon as a peer downloads a particular piece, it becomes another source for that piece.
 - The more peers downloading pieces of the file, the more piece replication occurs, distributing the load proportionately, and the more total bandwidth is available to share the file with others.
 - Pieces are downloaded in random order to avoid a situation where peers find themselves lacking the same set of pieces.

Overlay Network

- Peer-to-peer Networks
 - BitTorrent
 - Each file is shared via its own independent BitTorrent network, called a *swarm*. (A swarm could potentially share a set of files, but we describe the single file case for simplicity.)
 - The lifecycle of a typical swarm is as follows. The swarm starts as a singleton peer with a complete copy of the file.
 - A node that wants to download the file joins the swarm, becoming its second member, and begins downloading pieces of the file from the original peer.
 - In doing so, it becomes another source for the pieces it has downloaded, even if it has not yet downloaded the entire file.

Overlay Network

- Peer-to-peer Networks
 - BitTorrent



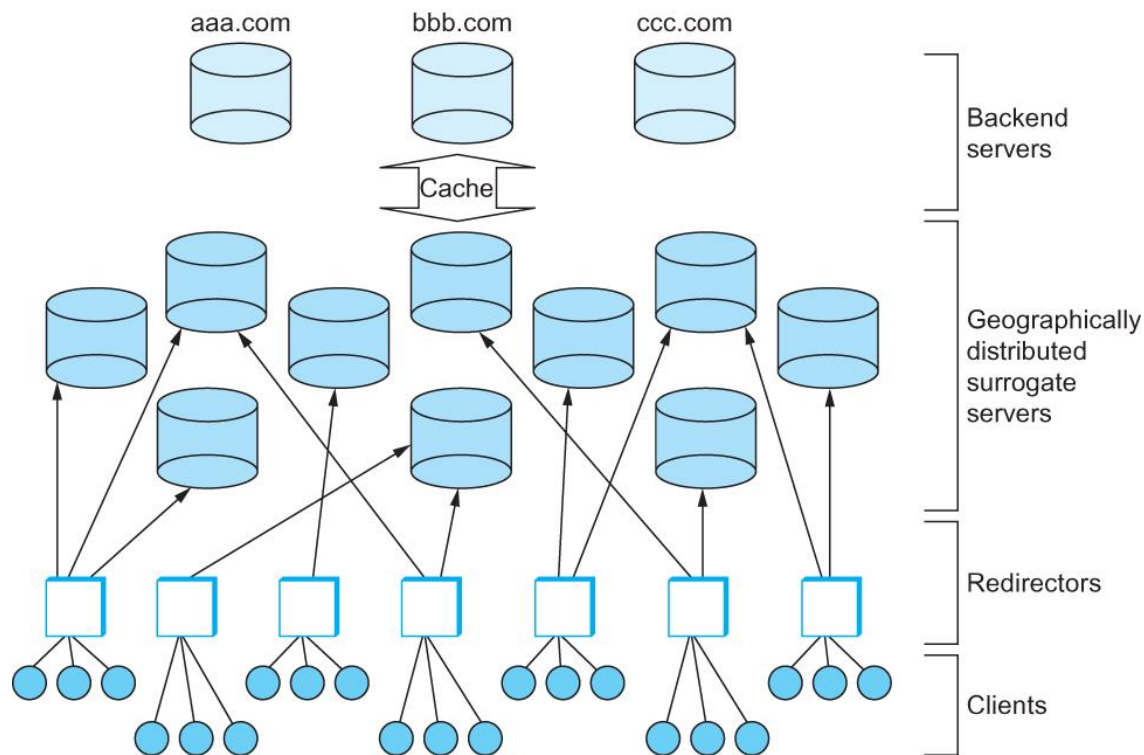
Peers in a BitTorrent swarm download from other peers that may not yet have the complete file

Overlay Network

- Content Distribution Network (CDN)
 - The idea of a CDN is to geographically distribute a collection of *server surrogates* that cache pages normally maintained in some set of *backend servers*
 - Akamai operates what is probably the best-known CDN.
 - Thus, rather than have millions of users wait forever to contact www.cnn.com when a big news story breaks—such a situation is known as a *flash crowd*—*it is possible to spread this load* across many servers.
 - Moreover, rather than having to traverse multiple ISPs to reach www.cnn.com, if these surrogate servers happen to be spread across all the backbone ISPs, then it should be possible to reach one without having to cross a peering point.

Overlay Network

■ Content Distribution Network (CDN)



Components in a Content Distribution Network (CDN).

Summary

- We have discussed some of the popular applications in the Internet
 - Electronic mail, World Wide Web
- We have discussed multimedia applications
- We have discussed infrastructure services
 - Domain Name Services (DNS)
- We have discussed overlay networks
 - Routing overlay, End-system multicast, Peer-to-peer networks
- We have discussed content distribution networks