

Chapter 1

Introduction

1-1 DATA COMMUNICATIONS

*The term **telecommunication** means communication at a distance. The word **data** refers to information presented in whatever form is agreed upon by the parties creating and using the data. **Data communications** are the exchange of data between two devices via some form of transmission medium such as a wire cable.*

Topics discussed in this section:

Components

Data Representation

Data Flow

Figure 1.1 Five components of data communication

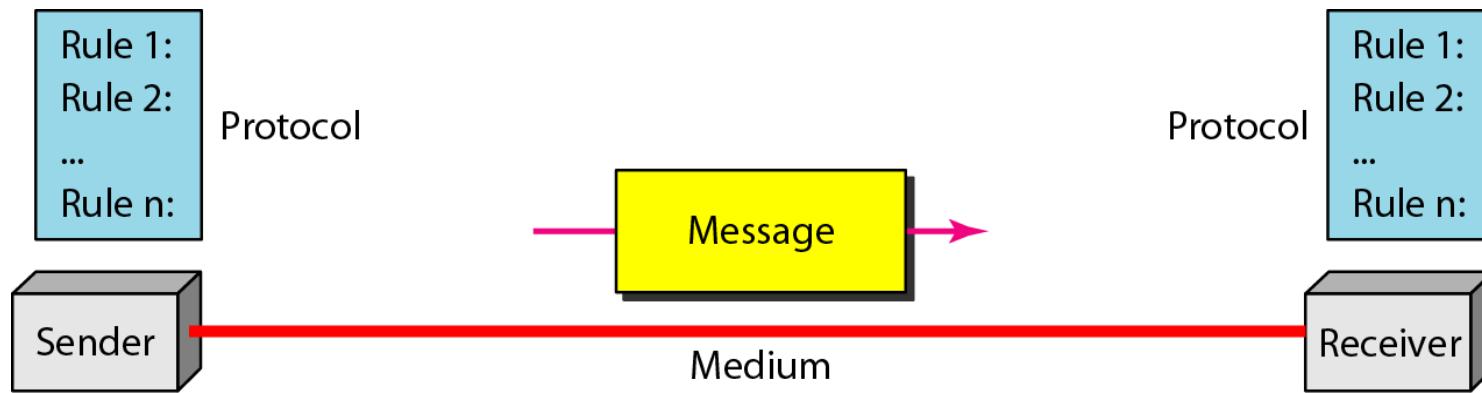
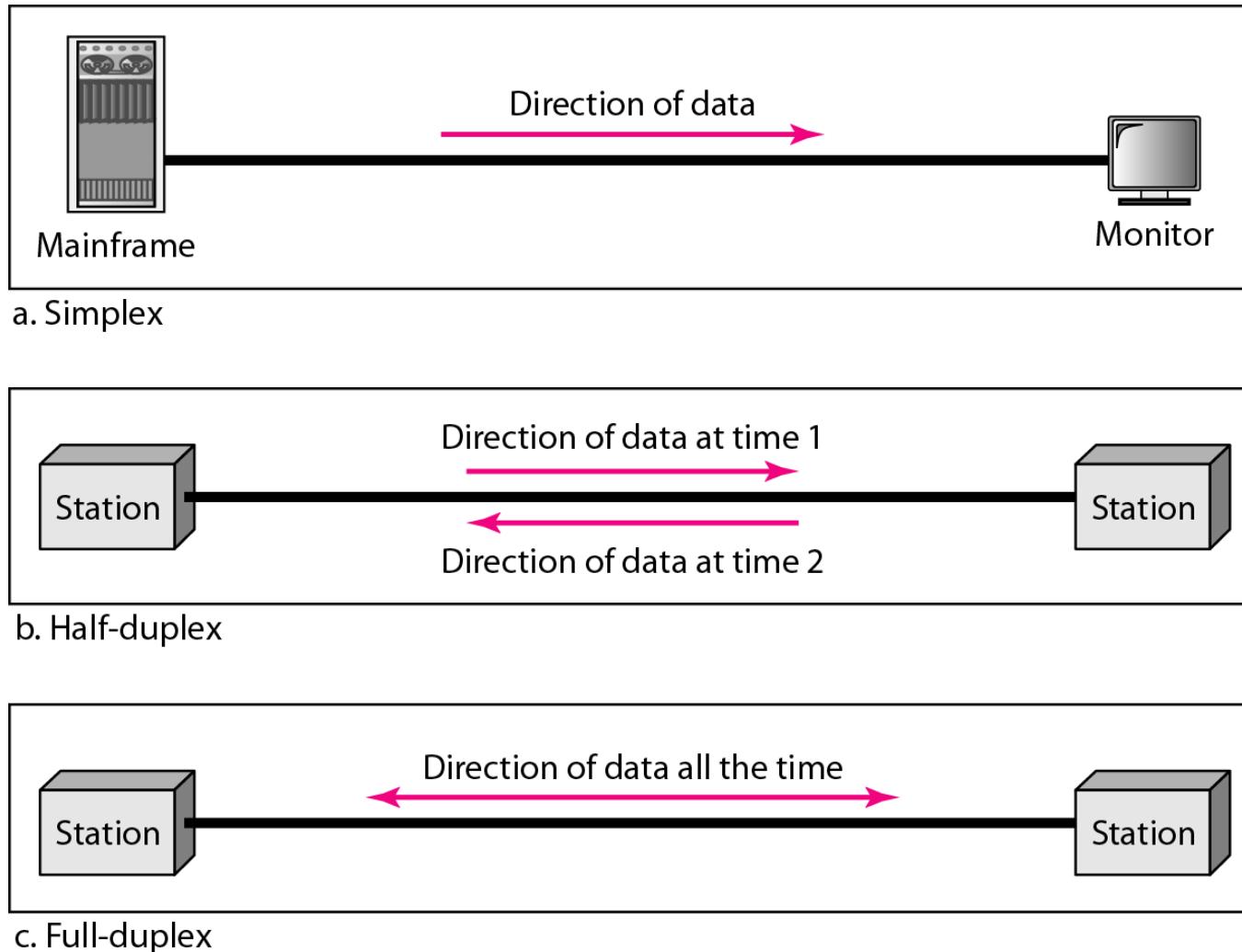


Figure 1.2 Data flow (simplex, half-duplex, and full-duplex)



1-2 NETWORKS

A **network** is a set of devices (often referred to as **nodes**) connected by communication **links**. A node can be a computer, printer, or any other device capable of sending and/or receiving data generated by other nodes on the network.

Topics discussed in this section:

Distributed Processing

Network Criteria

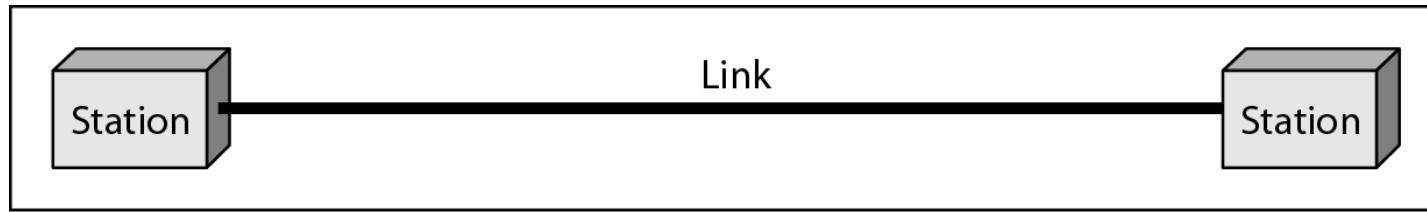
Physical Structures

Network Models

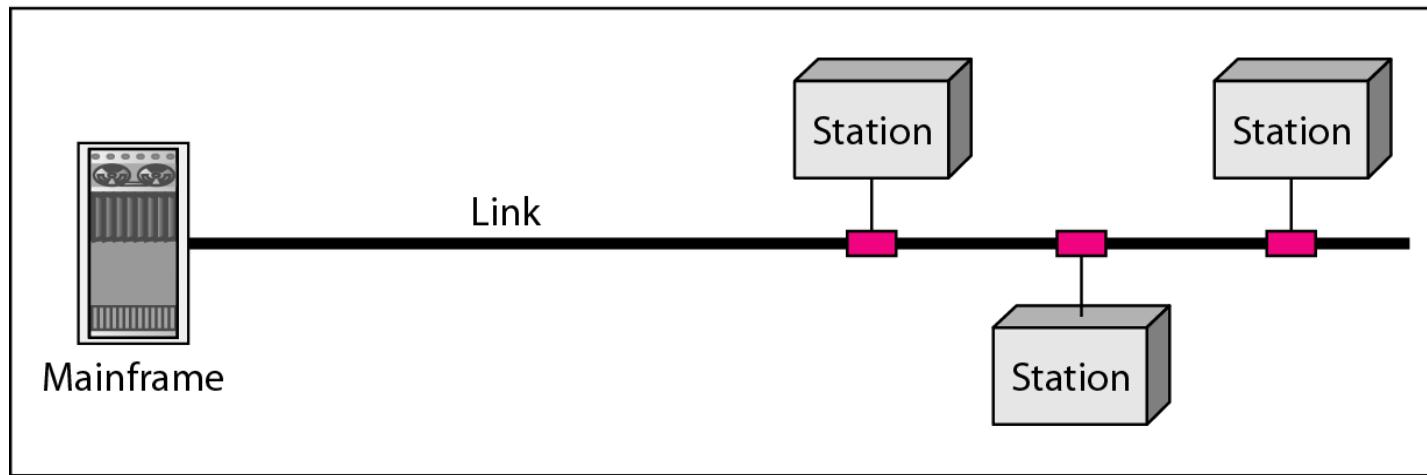
Categories of Networks

Interconnection of Networks: Internetwork

Figure 1.3 *Types of connections: point-to-point and multipoint*



a. Point-to-point



b. Multipoint

Figure 1.4 *Categories of topology*

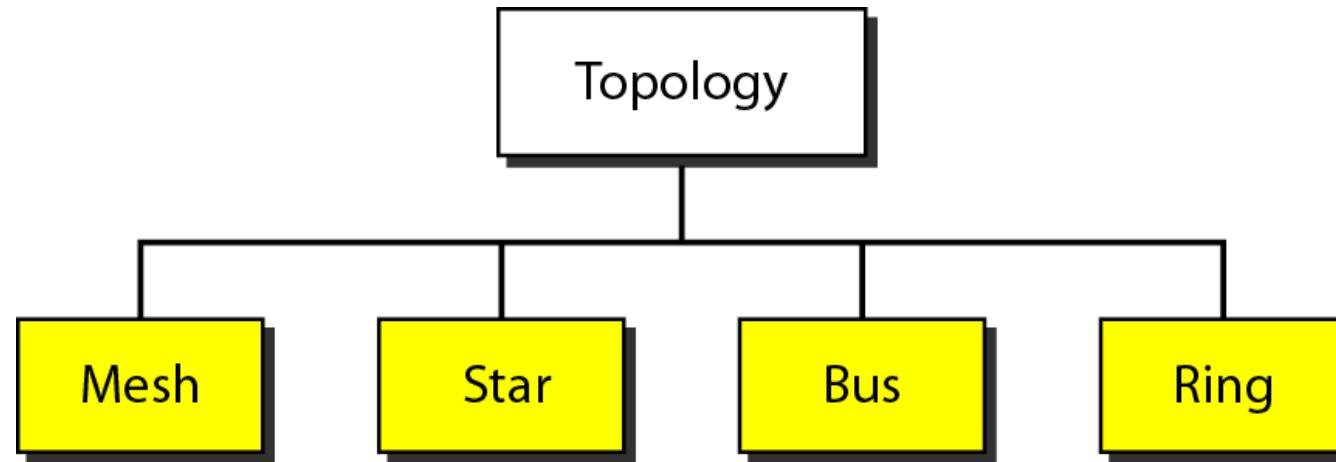


Figure 1.5 A *fully connected mesh topology (five devices)*

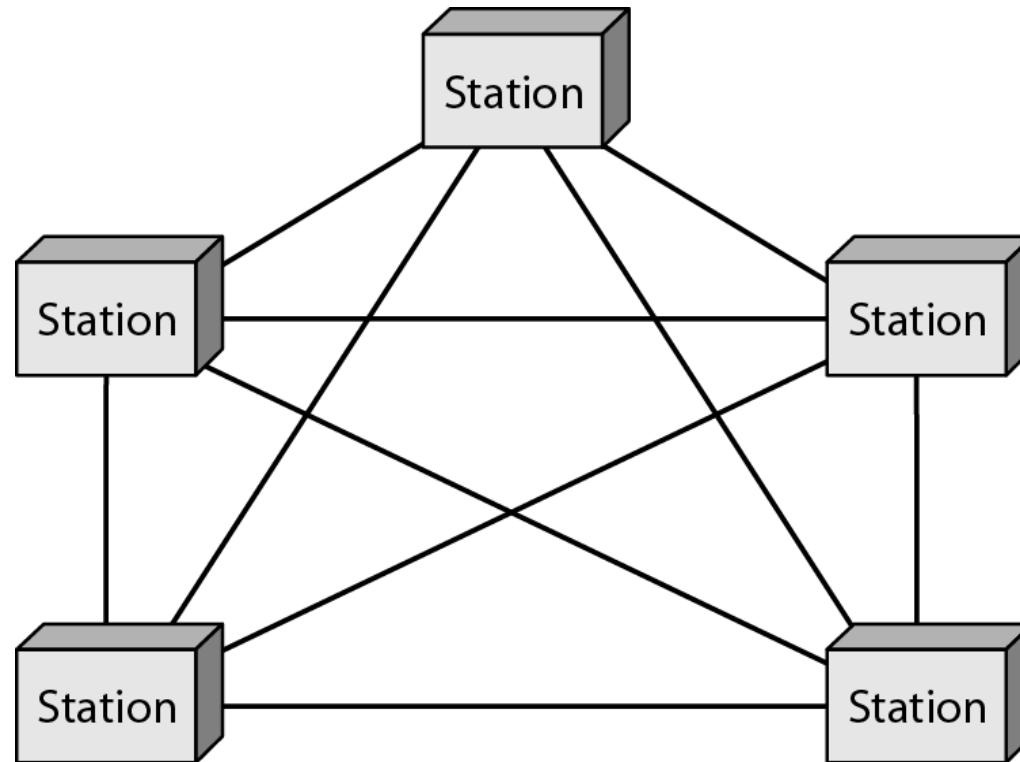


Figure 1.6 A star topology connecting four stations

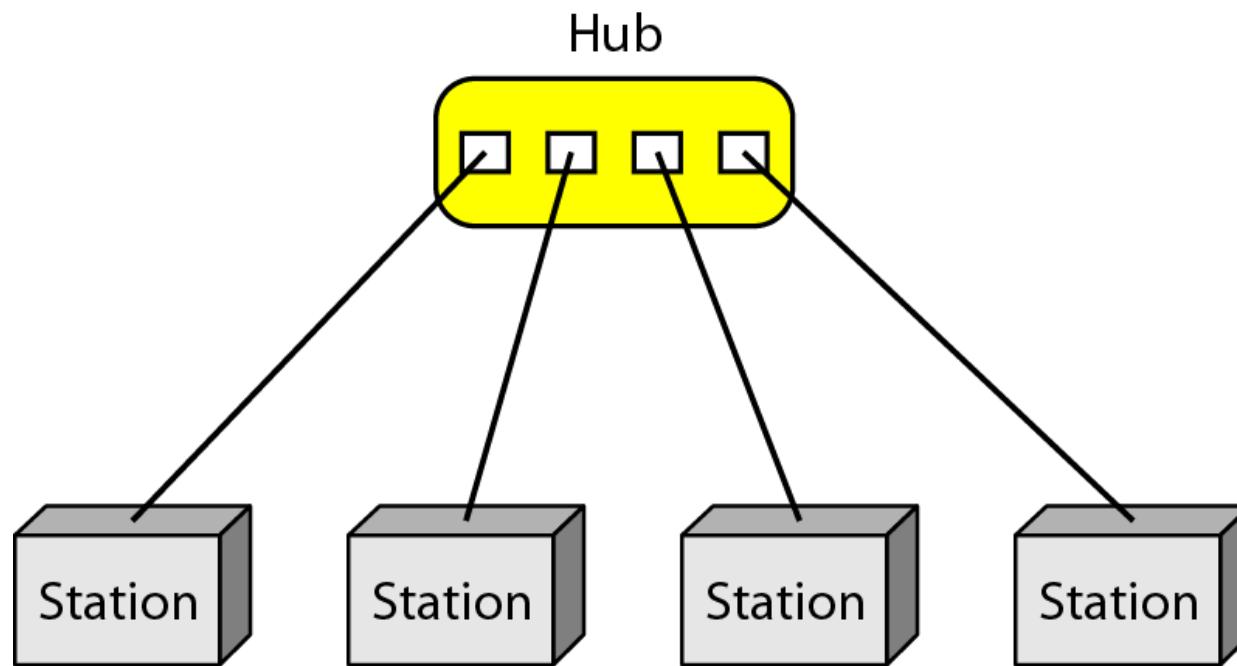


Figure 1.7 A bus topology connecting three stations

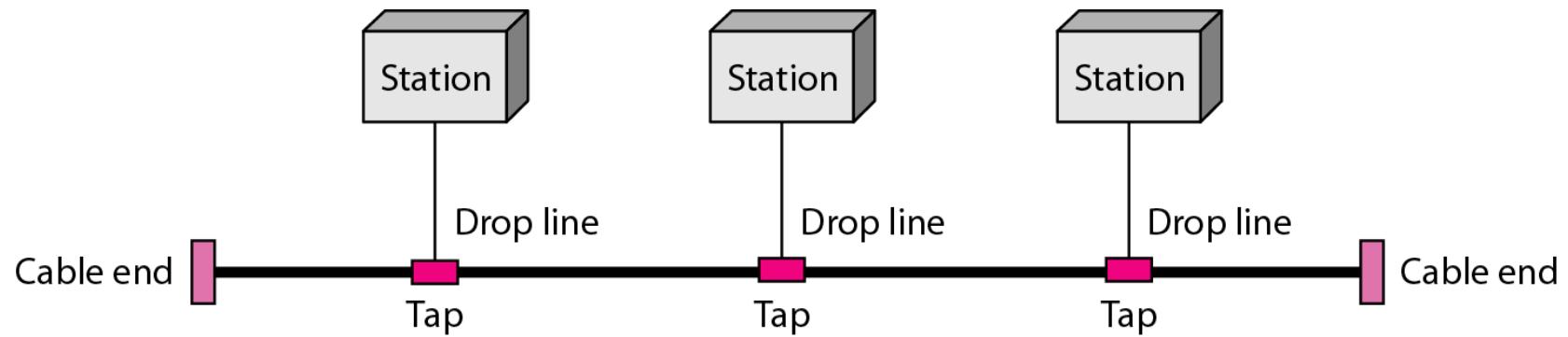


Figure 1.8 A *ring topology connecting six stations*

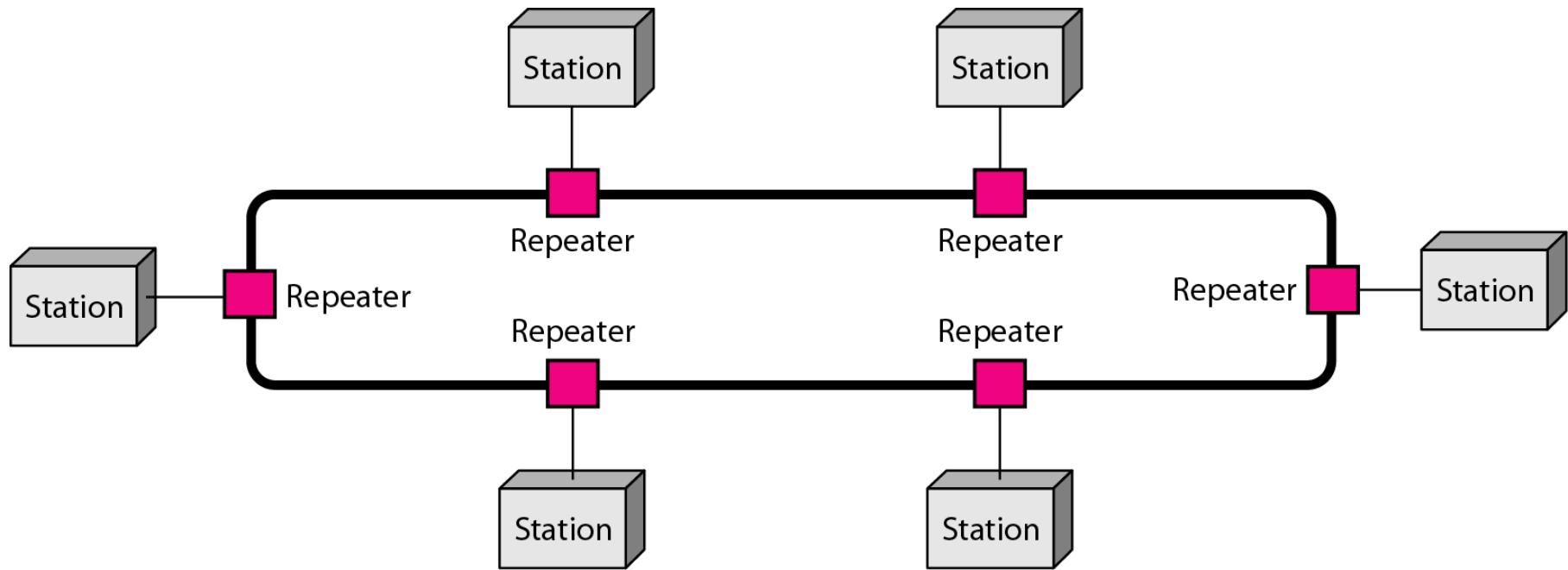


Figure 1.9 A hybrid topology: a star backbone with three bus networks

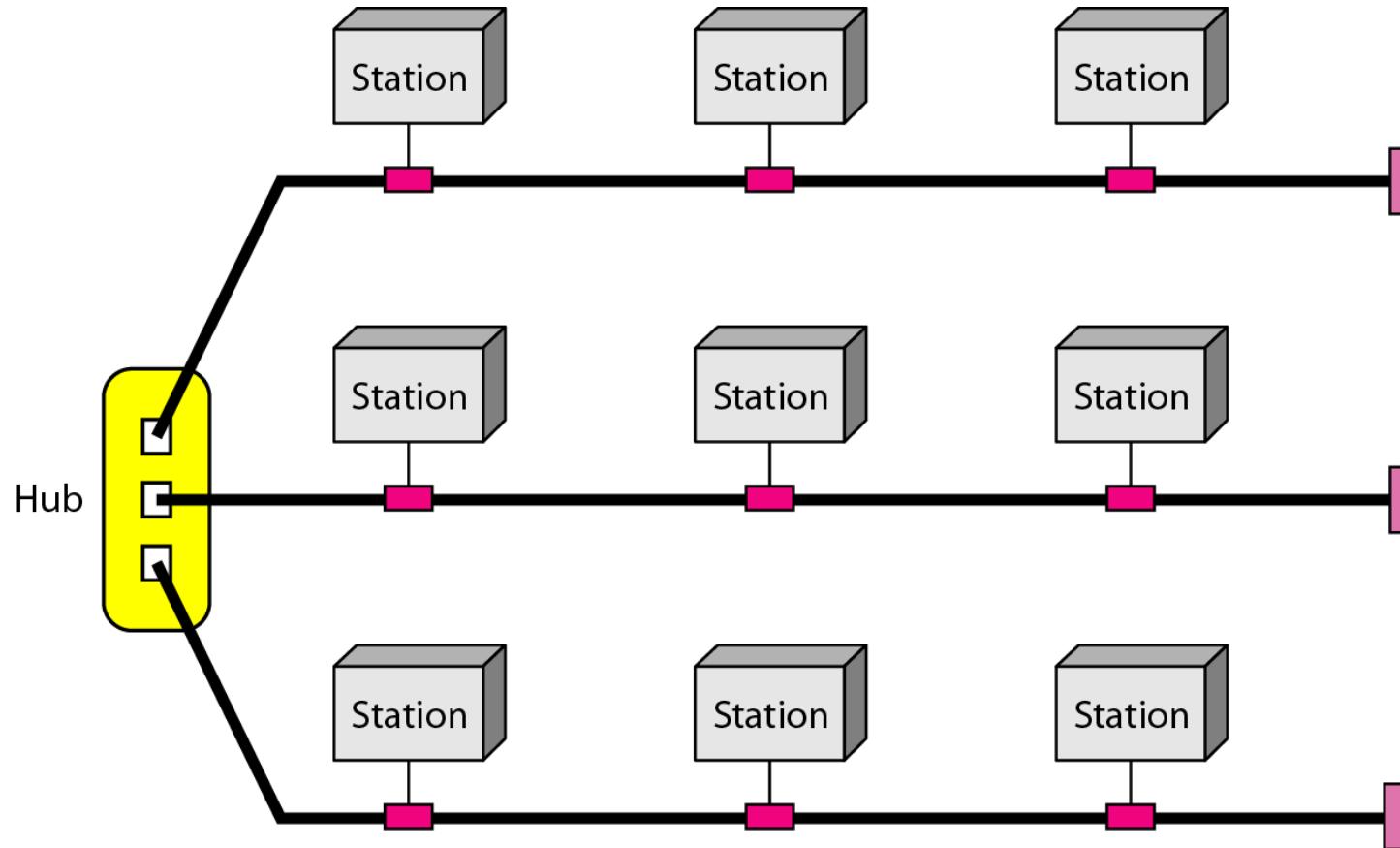


Figure 1.10 An isolated LAN connecting 12 computers to a hub in a closet

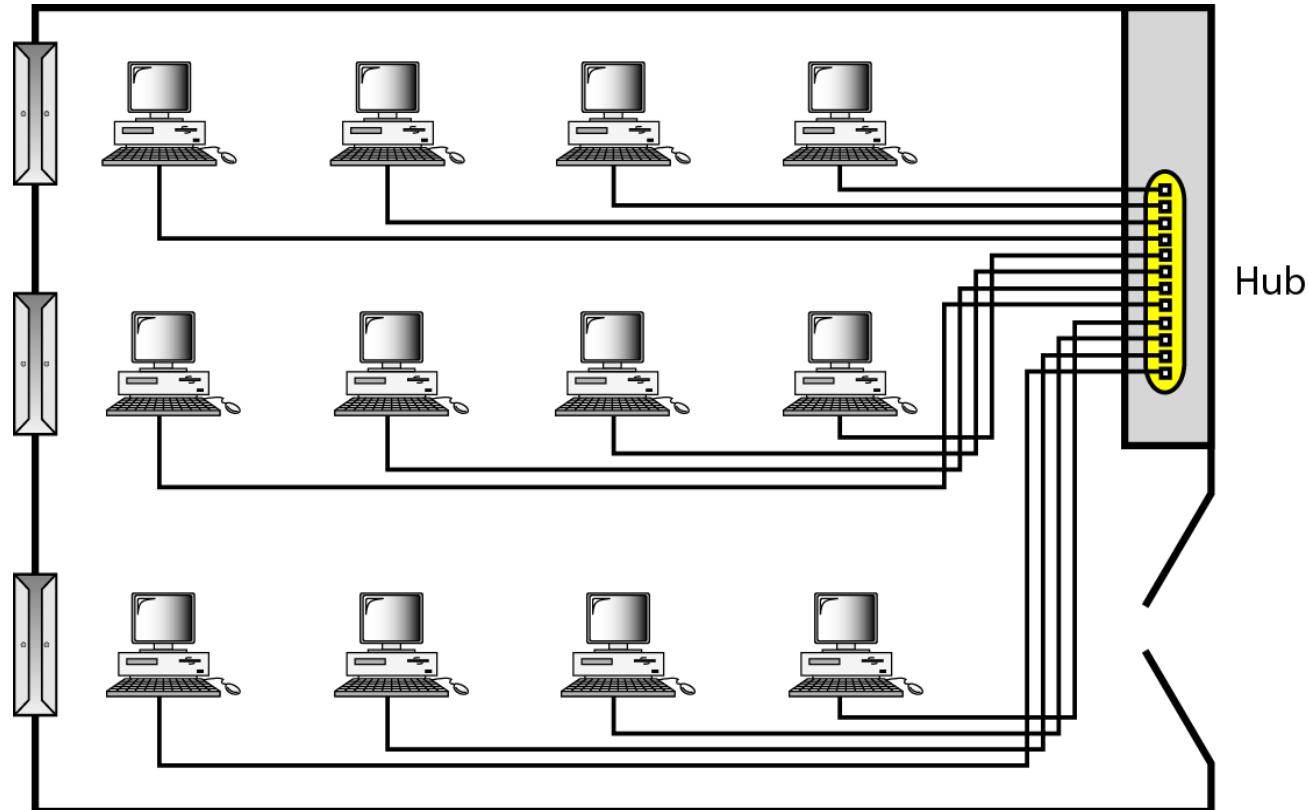
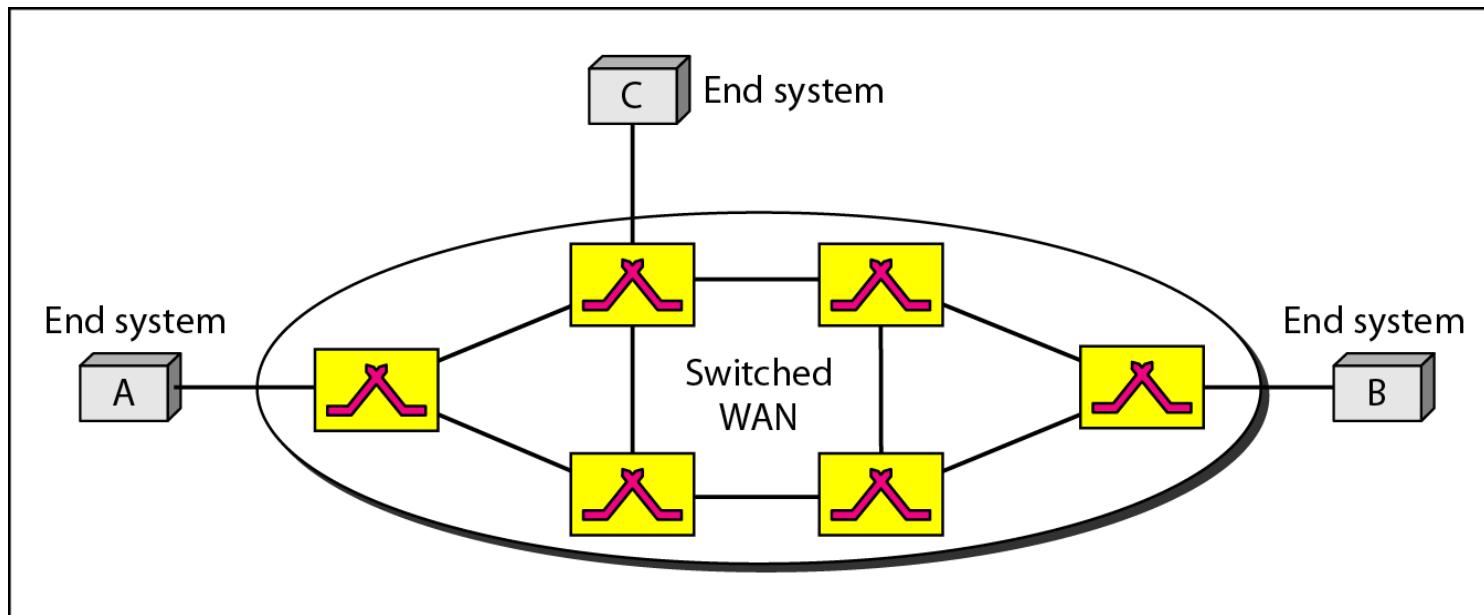
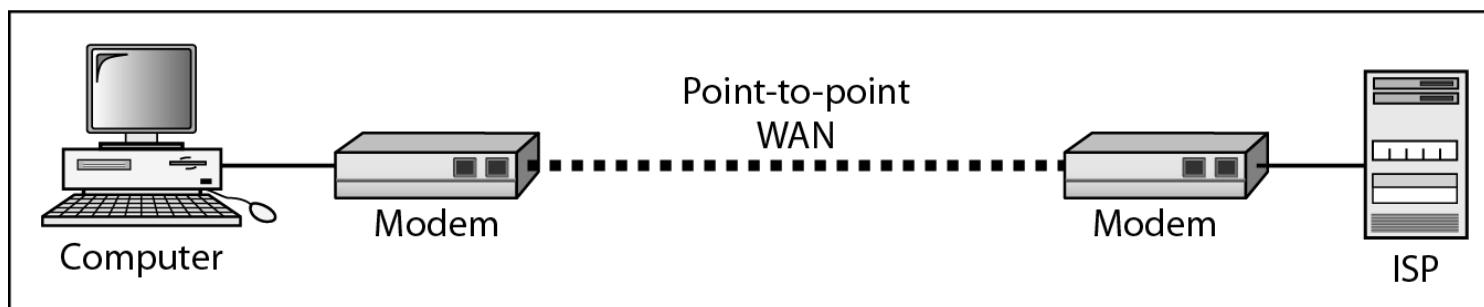


Figure 1.11 WANs: a switched WAN and a point-to-point WAN

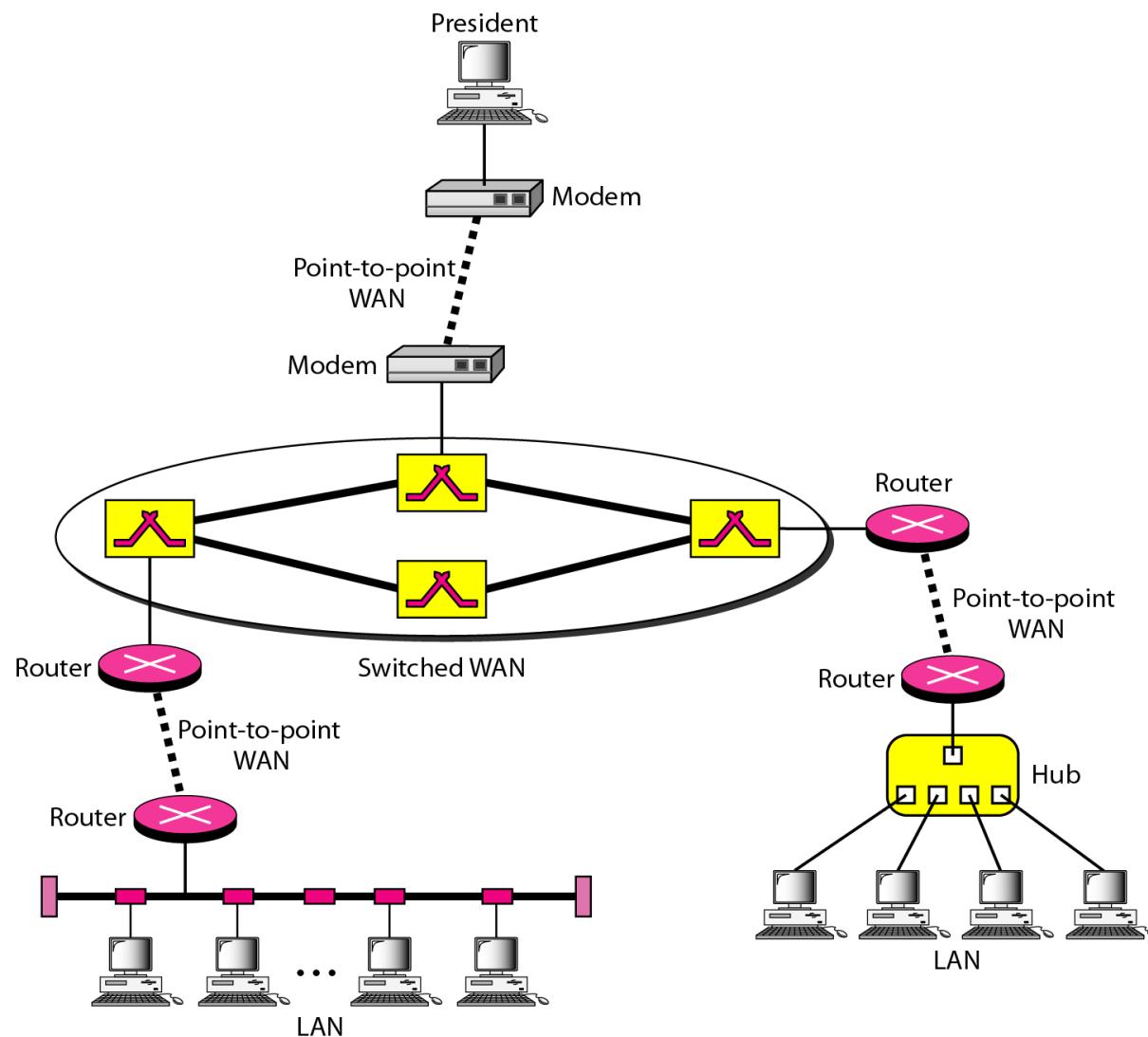


a. Switched WAN



b. Point-to-point WAN

Figure 1.12 A heterogeneous network made of four WANs and two LANs



1-3 THE INTERNET

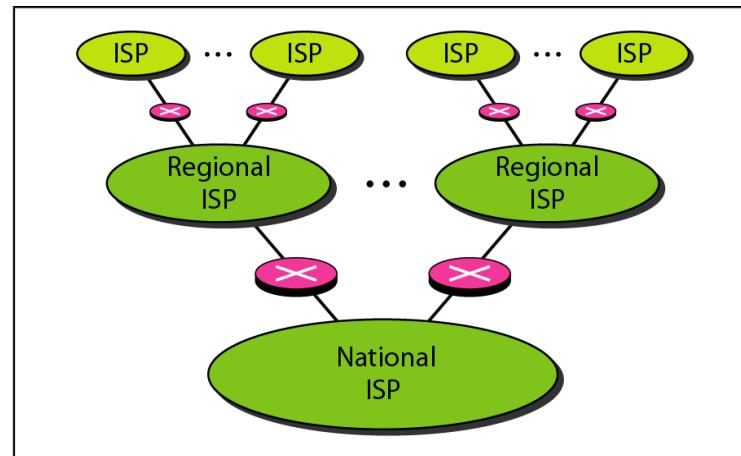
The Internet has revolutionized many aspects of our daily lives. It has affected the way we do business as well as the way we spend our leisure time. The Internet is a communication system that has brought a wealth of information to our fingertips and organized it for our use.

Topics discussed in this section:

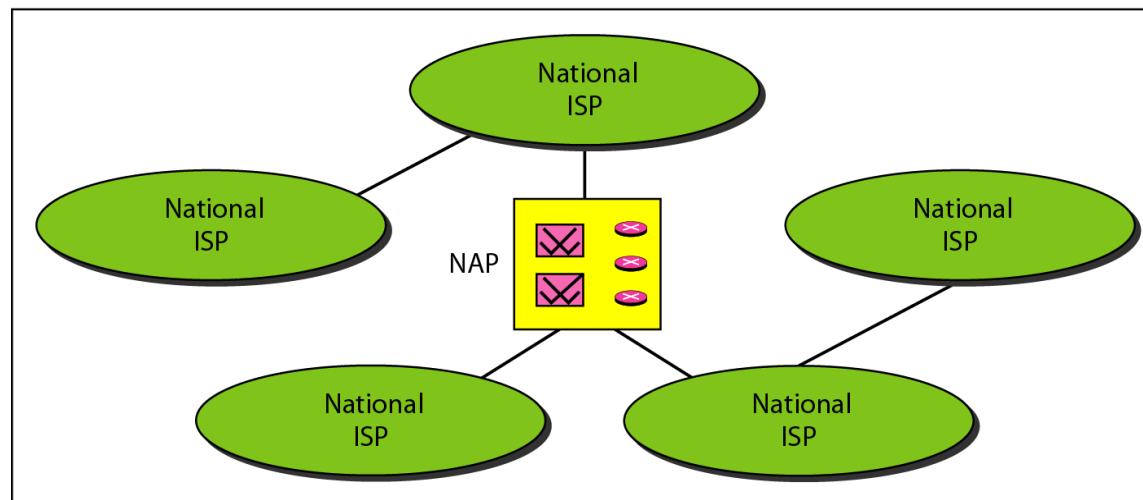
A Brief History

The Internet Today (ISPs)

Figure 1.13 *Hierarchical organization of the Internet*



a. Structure of a national ISP



b. Interconnection of national ISPs

1-4 PROTOCOLS AND STANDARDS

*In this section, we define two widely used terms: **protocols** and **standards**. First, we define protocol, which is synonymous with rule. Then we discuss standards, which are agreed-upon rules.*

Topics discussed in this section:

Protocols

Standards

Standards Organizations

Internet Standards

Chapter 2

Network Models

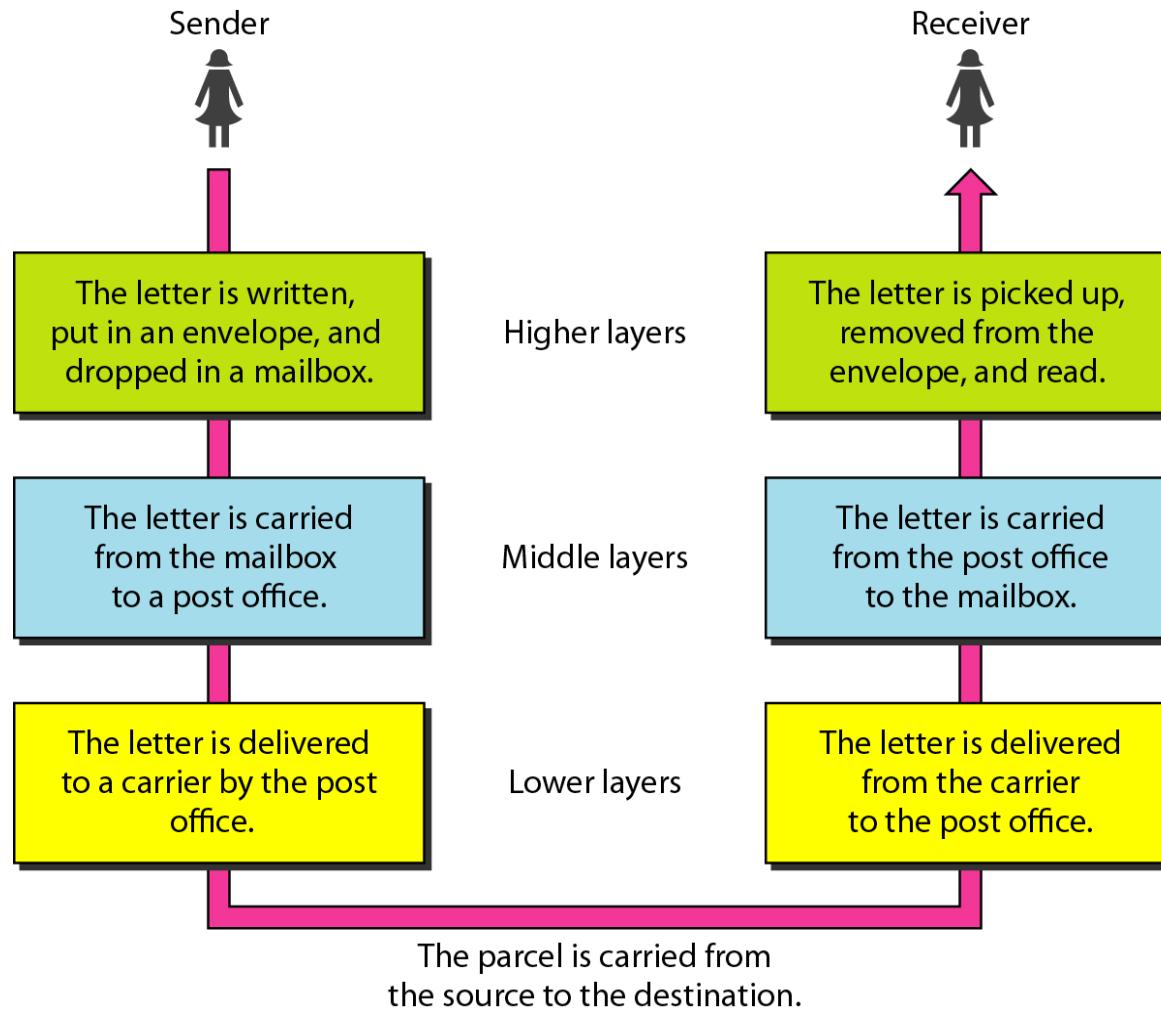
2-1 LAYERED TASKS

*We use the concept of **layers** in our daily life. As an example, let us consider two friends who communicate through postal mail. The process of sending a letter to a friend would be complex if there were no services available from the post office.*

Topics discussed in this section:

Sender, Receiver, and Carrier
Hierarchy

Figure 2.1 Tasks involved in sending a letter



2-2 THE OSI MODEL

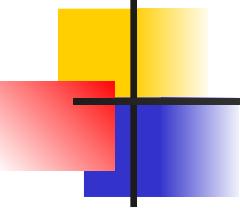
*Established in 1947, the International Standards Organization (**ISO**) is a multinational body dedicated to worldwide agreement on international standards. An ISO standard that covers all aspects of network communications is the Open Systems Interconnection (**OSI**) model. It was first introduced in the late 1970s.*

Topics discussed in this section:

Layered Architecture

Peer-to-Peer Processes

Encapsulation



Note

**ISO is the organization.
OSI is the model.**

Figure 2.2 *Seven layers of the OSI model*

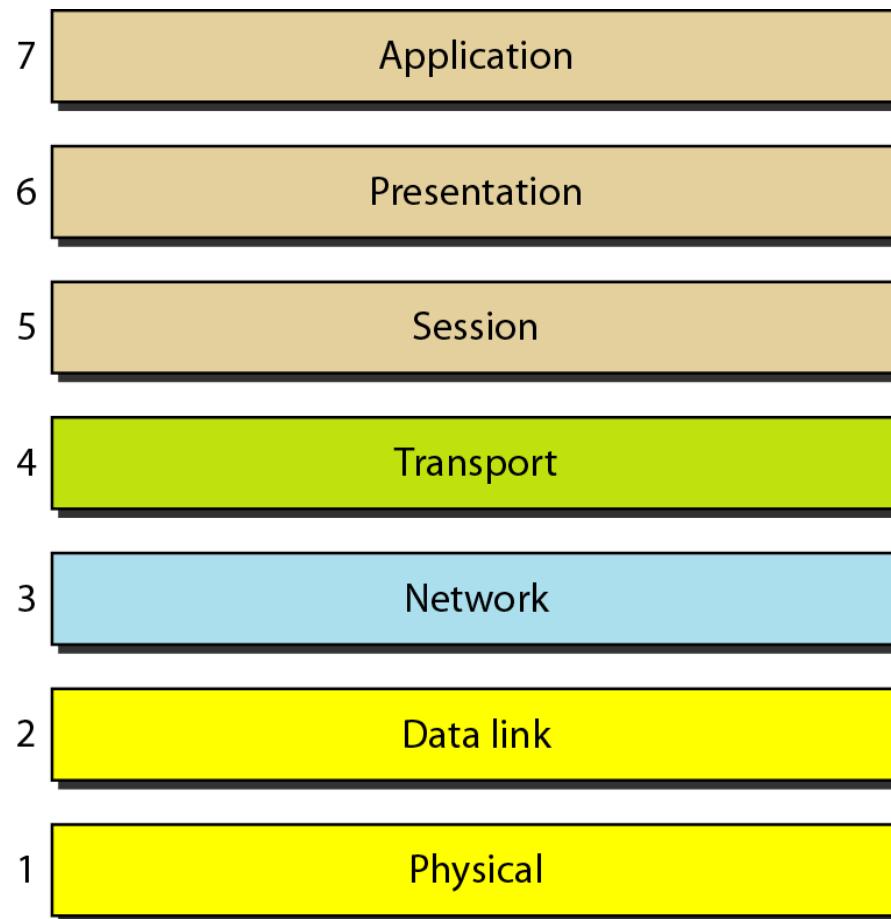


Figure 2.3 *The interaction between layers in the OSI model*

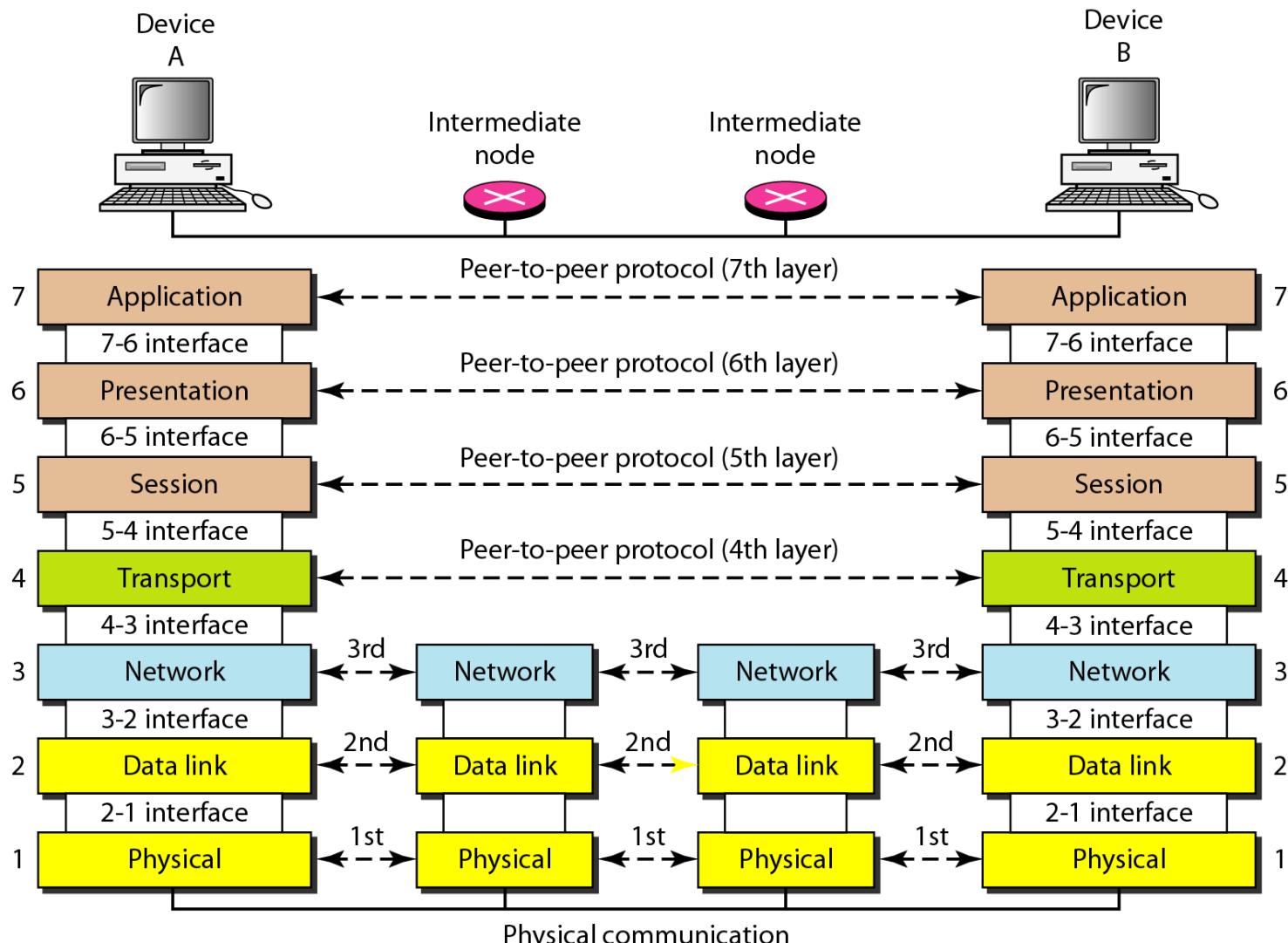
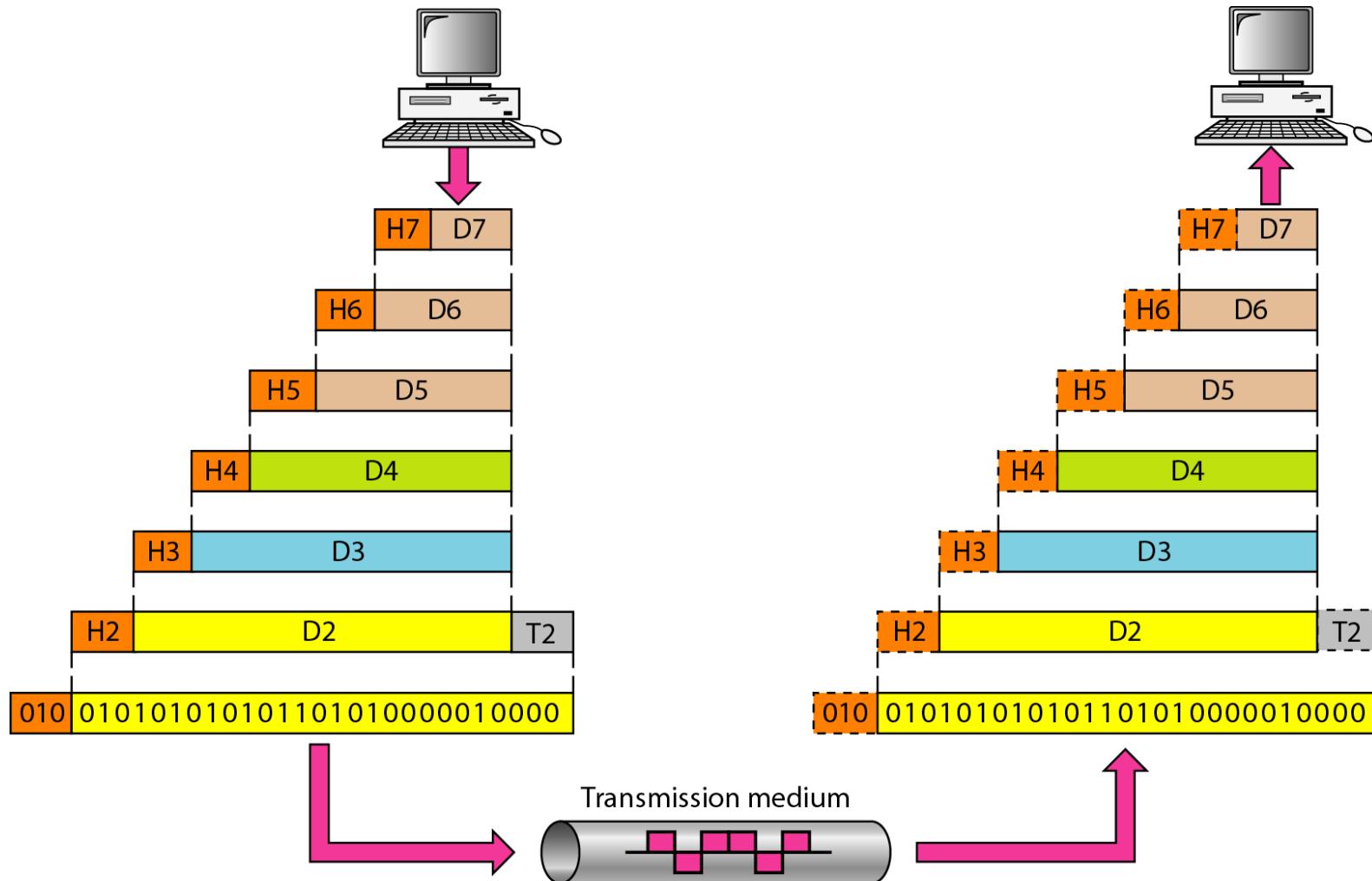


Figure 2.4 An exchange using the OSI model



2-3 LAYERS IN THE OSI MODEL

In this section we briefly describe the functions of each layer in the OSI model.

Topics discussed in this section:

Physical Layer

Data Link Layer

Network Layer

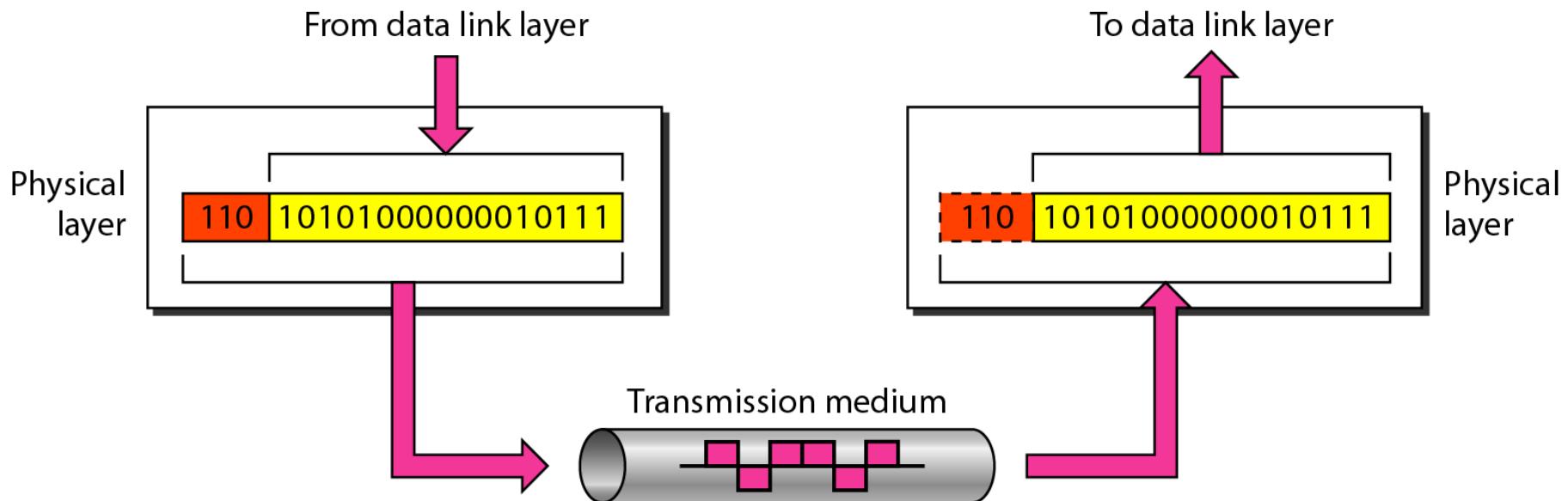
Transport Layer

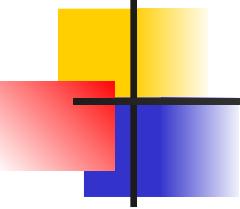
Session Layer

Presentation Layer

Application Layer

Figure 2.5 Physical layer

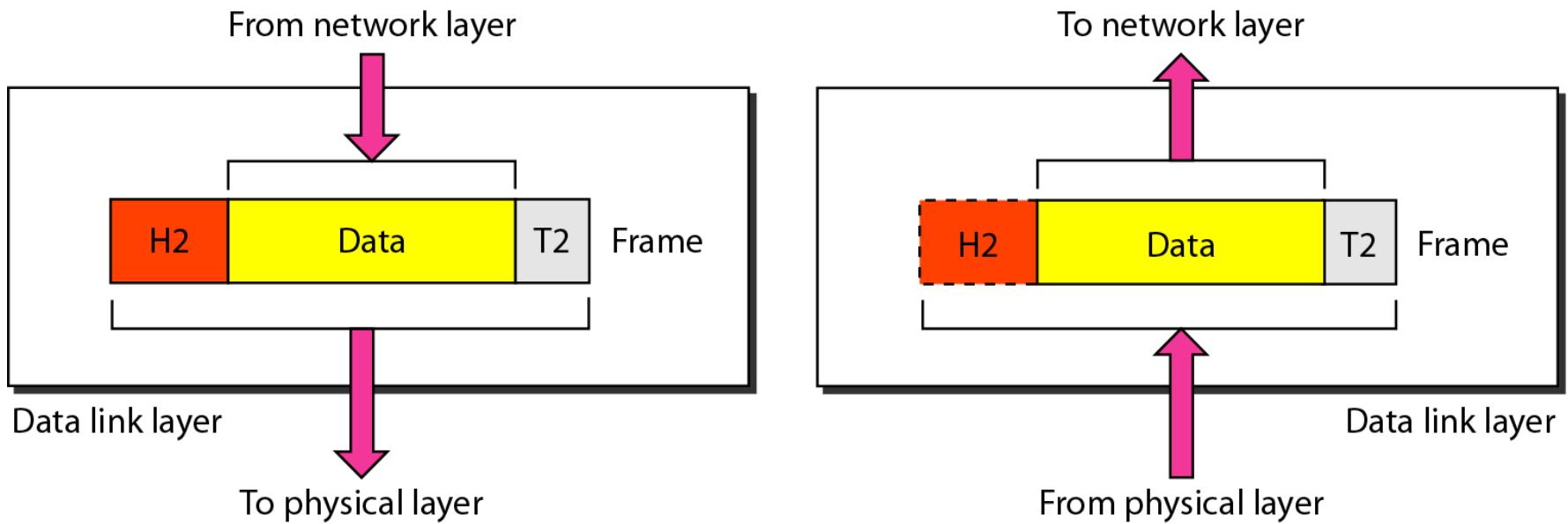


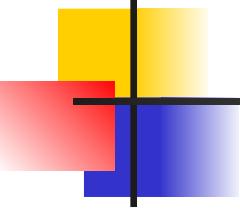


Note

The physical layer is responsible for movements of individual bits from one hop (node) to the next.

Figure 2.6 Data link layer





Note

The data link layer is responsible for moving frames from one hop (node) to the next.

Figure 2.7 Hop-to-hop delivery

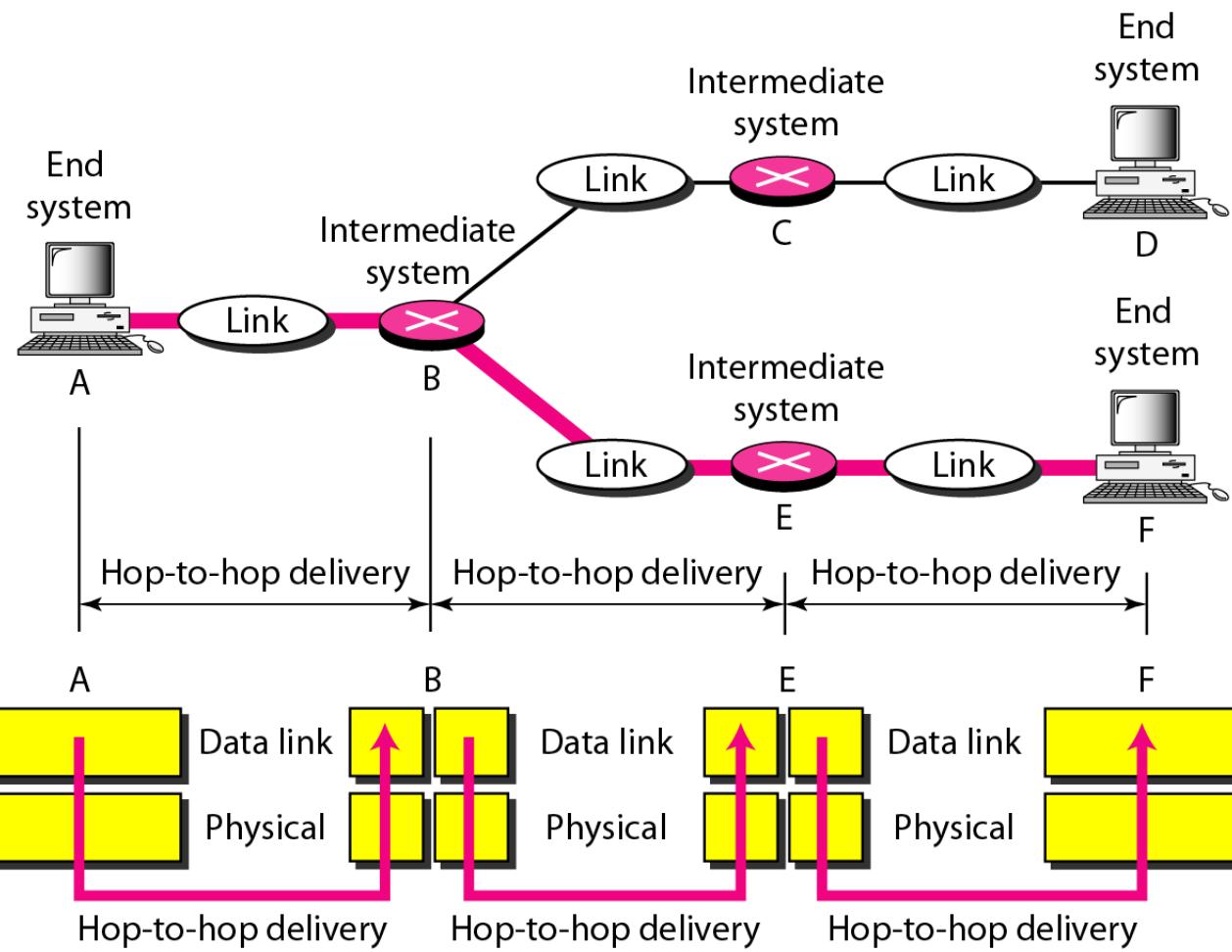
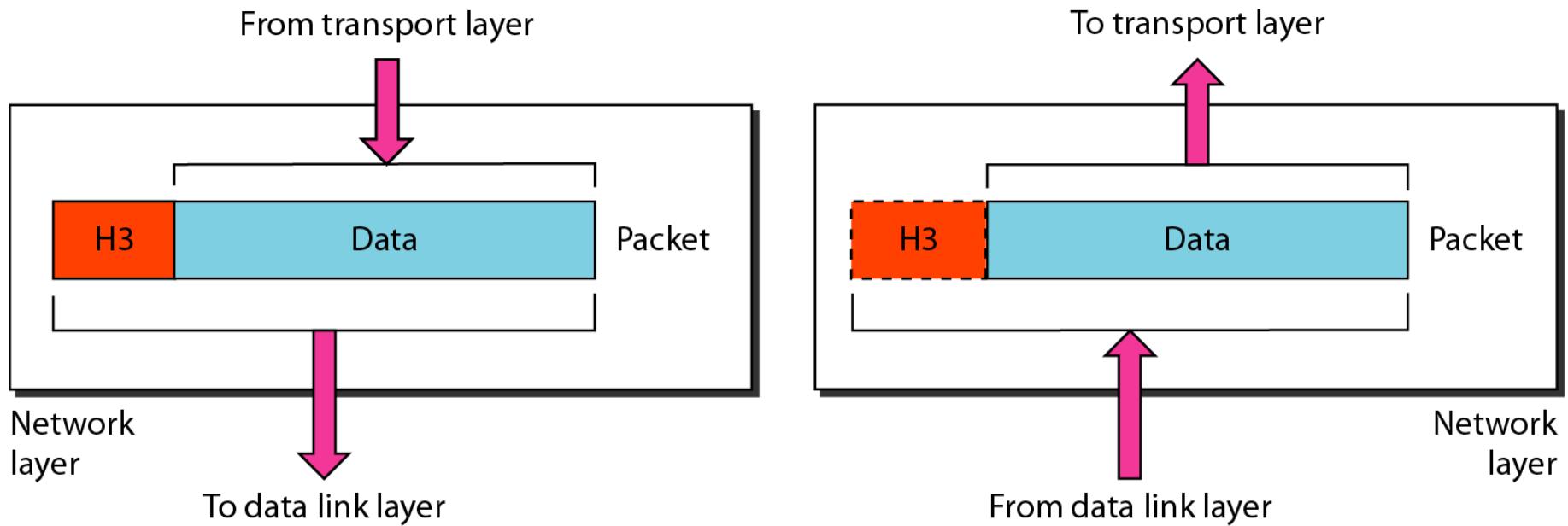
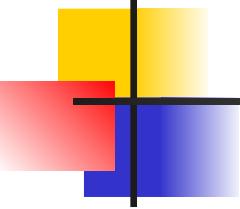


Figure 2.8 Network layer





Note

The network layer is responsible for the delivery of individual packets from the source host to the destination host.

Figure 2.9 Source-to-destination delivery

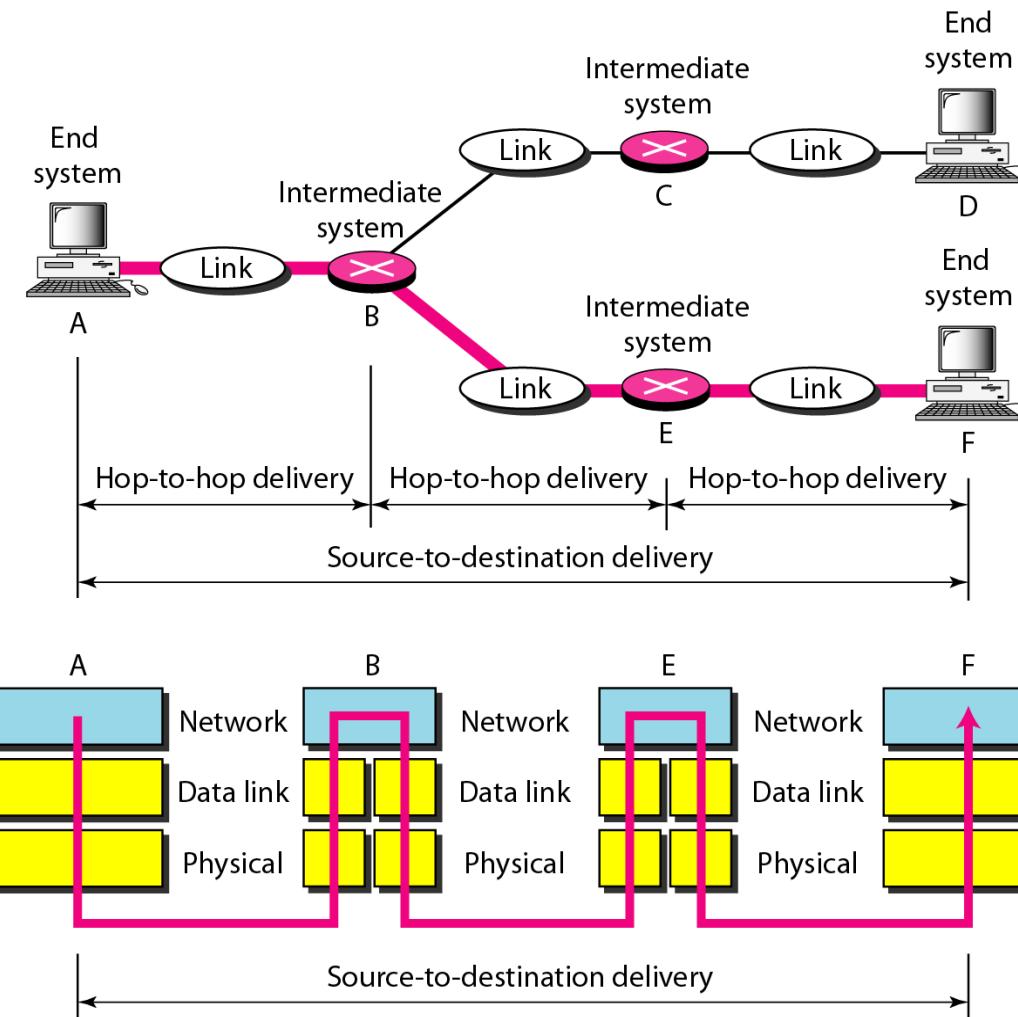
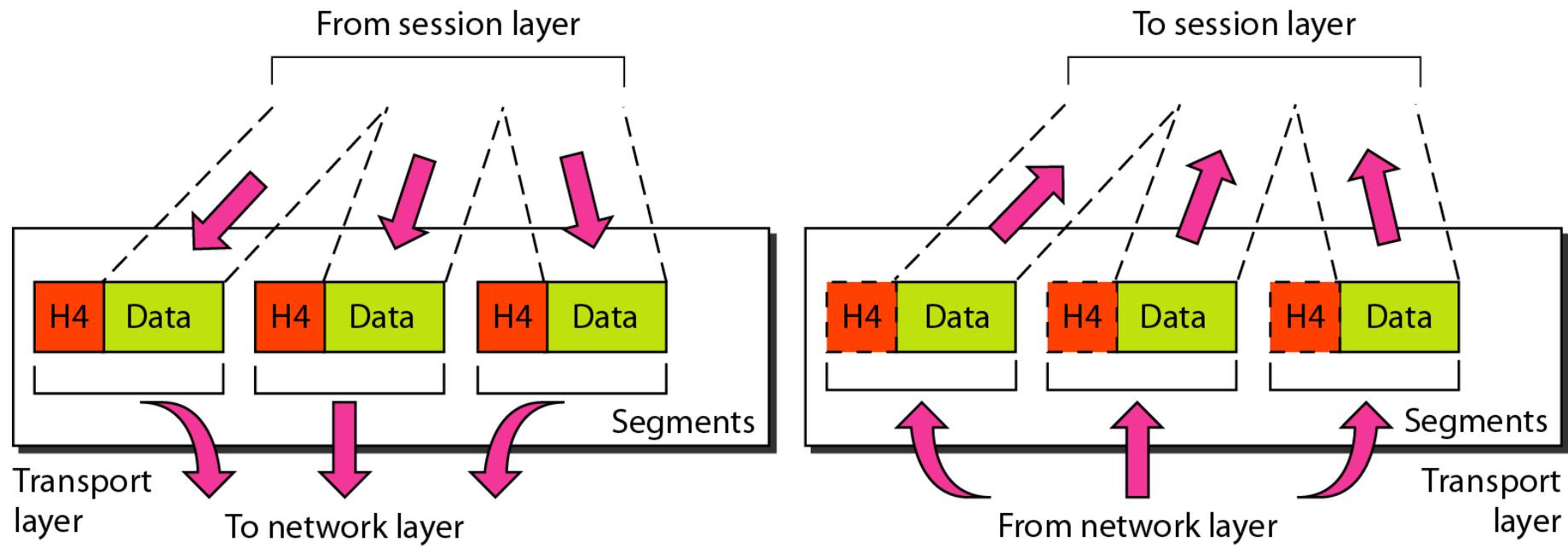
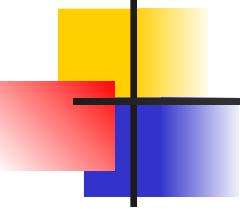


Figure 2.10 Transport layer





Note

**The transport layer is responsible for the delivery
of a message from one process to another.**

Figure 2.11 Reliable process-to-process delivery of a message

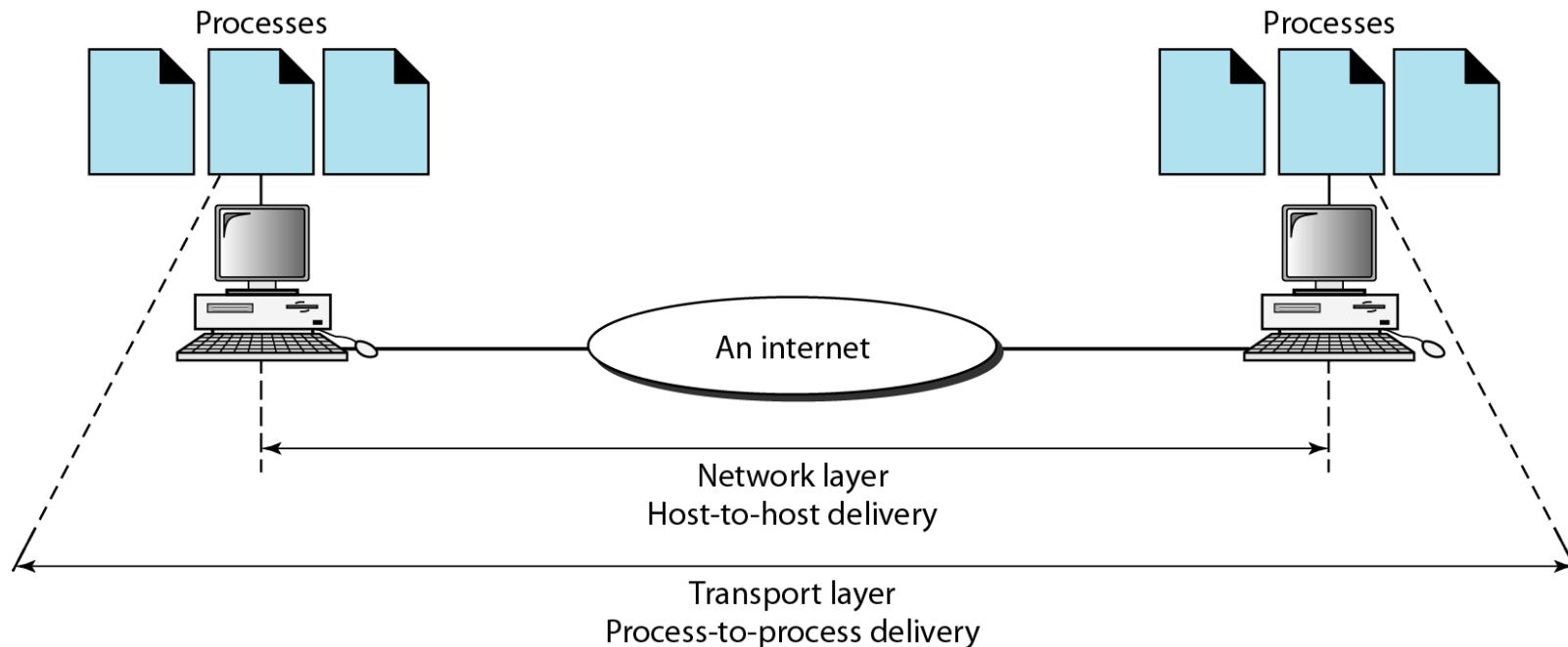
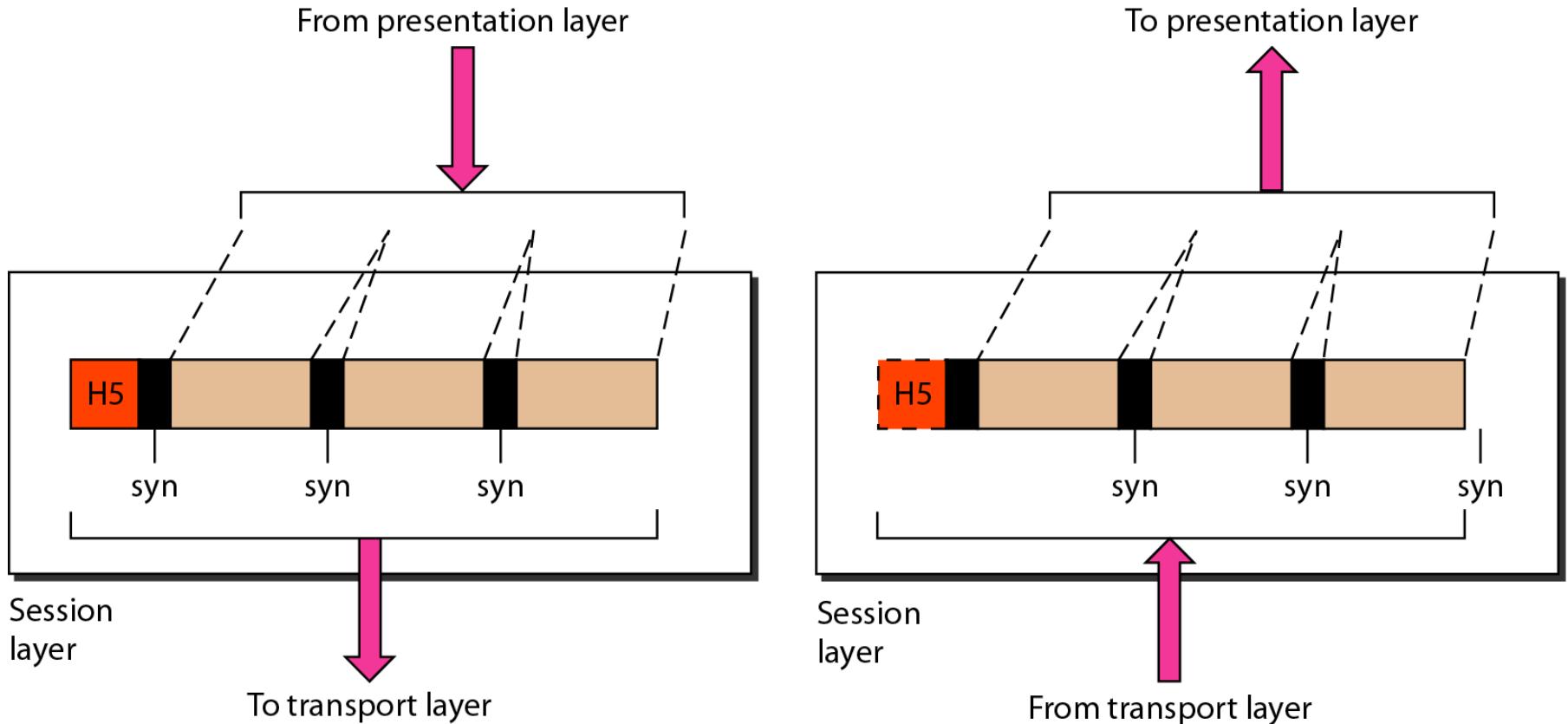


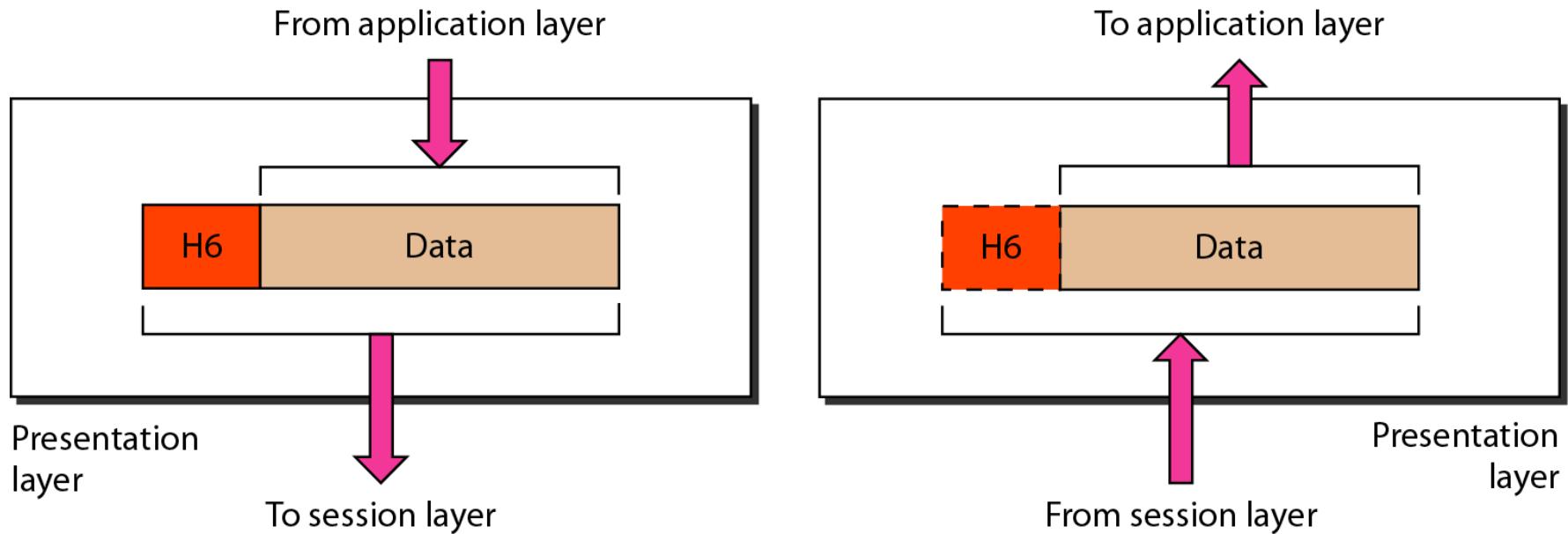
Figure 2.12 Session layer

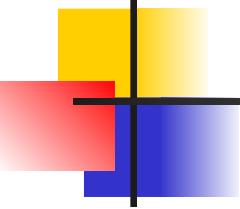


Note

The session layer is responsible for dialog control and synchronization.

Figure 2.13 Presentation layer

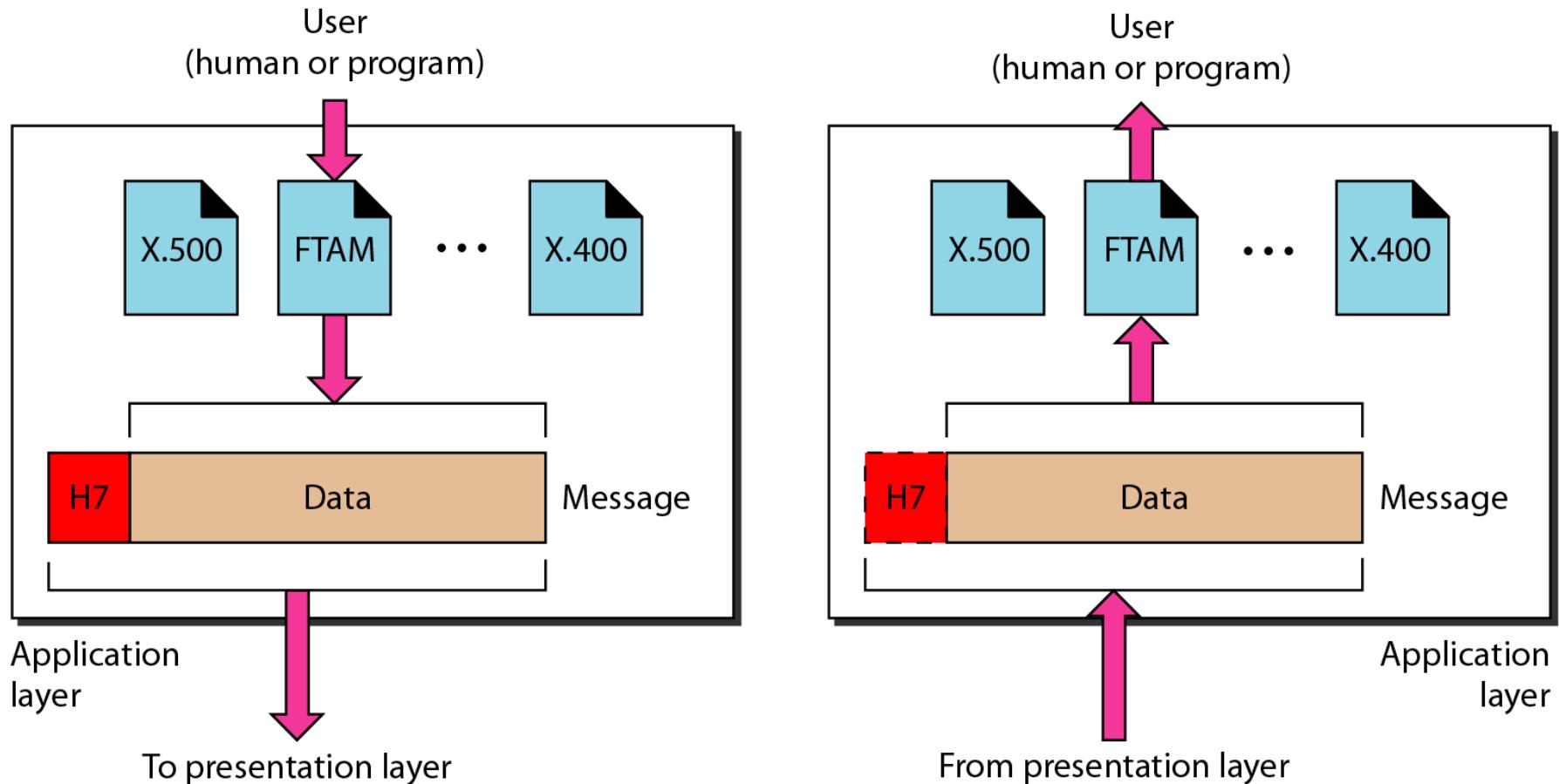


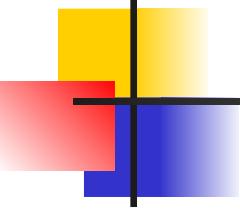


Note

The presentation layer is responsible for translation, compression, and encryption.

Figure 2.14 Application layer

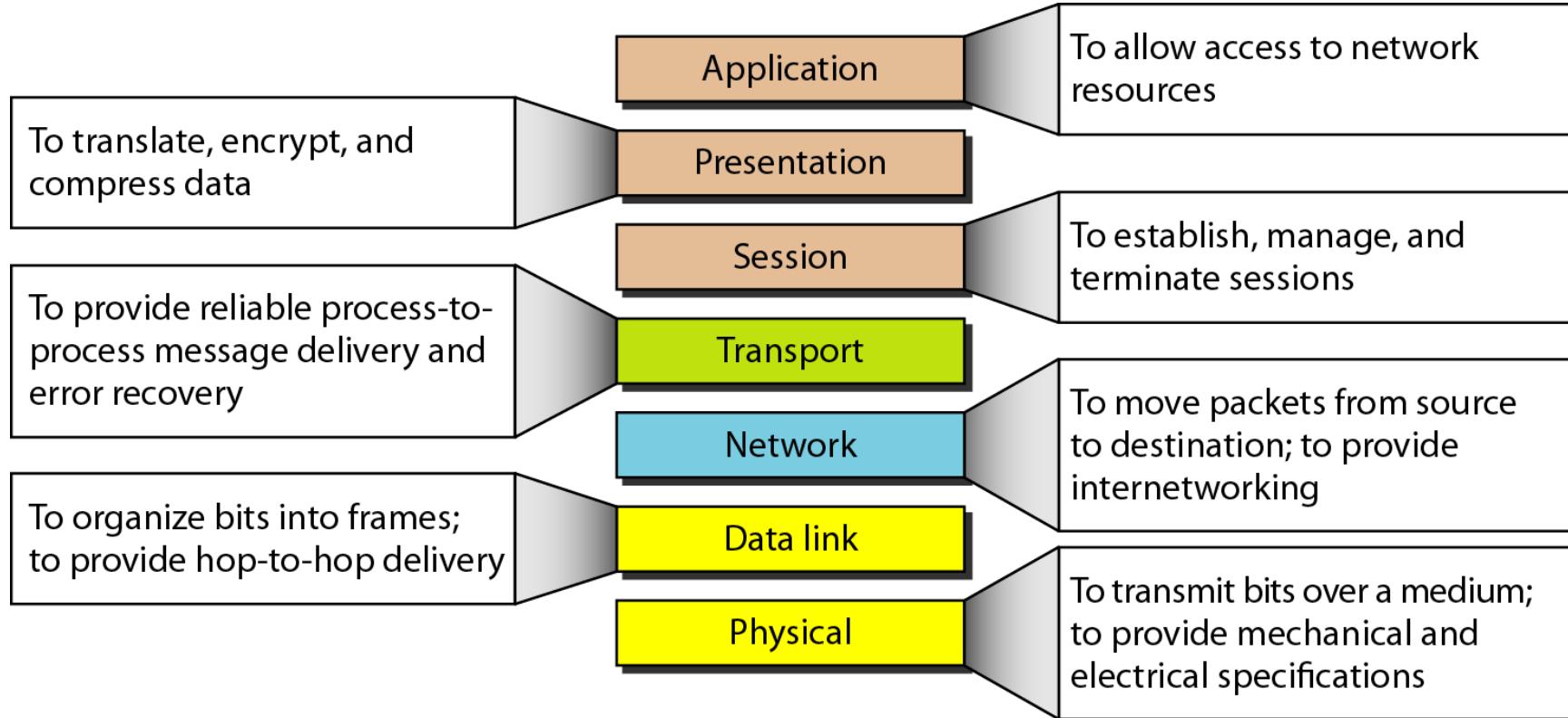




Note

The application layer is responsible for providing services to the user.

Figure 2.15 Summary of layers



2-4 TCP/IP PROTOCOL SUITE

*The layers in the **TCP/IP protocol suite** do not exactly match those in the **OSI model**. The original **TCP/IP protocol suite** was defined as having four layers: **host-to-network**, **internet**, **transport**, and **application**. However, when **TCP/IP** is compared to **OSI**, we can say that the **TCP/IP protocol suite** is made of five layers: **physical**, **data link**, **network**, **transport**, and **application**.*

Topics discussed in this section:

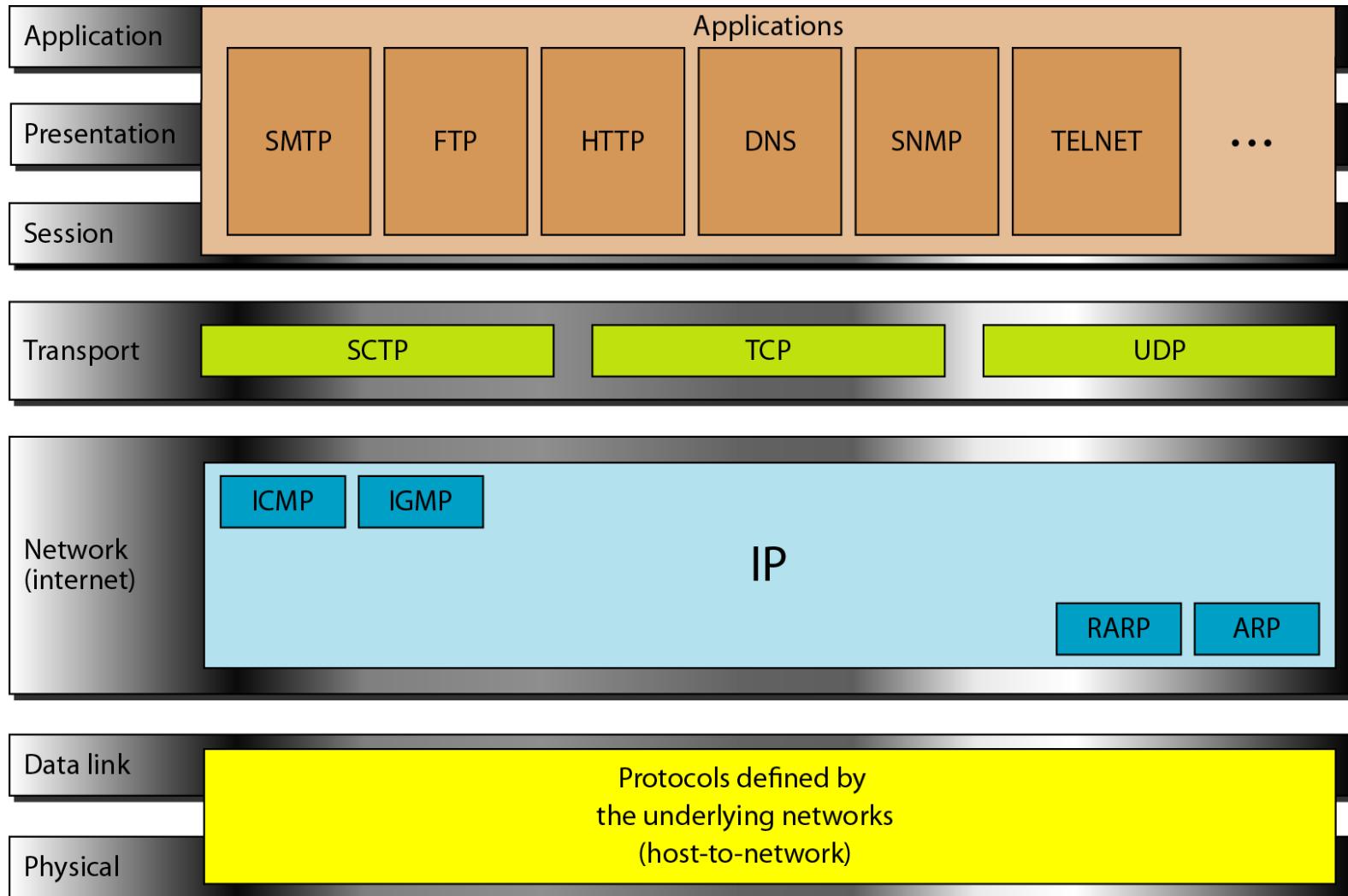
Physical and Data Link Layers

Network Layer

Transport Layer

Application Layer

Figure 2.16 TCP/IP and OSI model



2-5 ADDRESSING

*Four levels of addresses are used in an internet employing the TCP/IP protocols: **physical**, **logical**, **port**, and **specific**.*

Topics discussed in this section:

Physical Addresses

Logical Addresses

Port Addresses

Specific Addresses

Figure 2.17 Addresses in TCP/IP

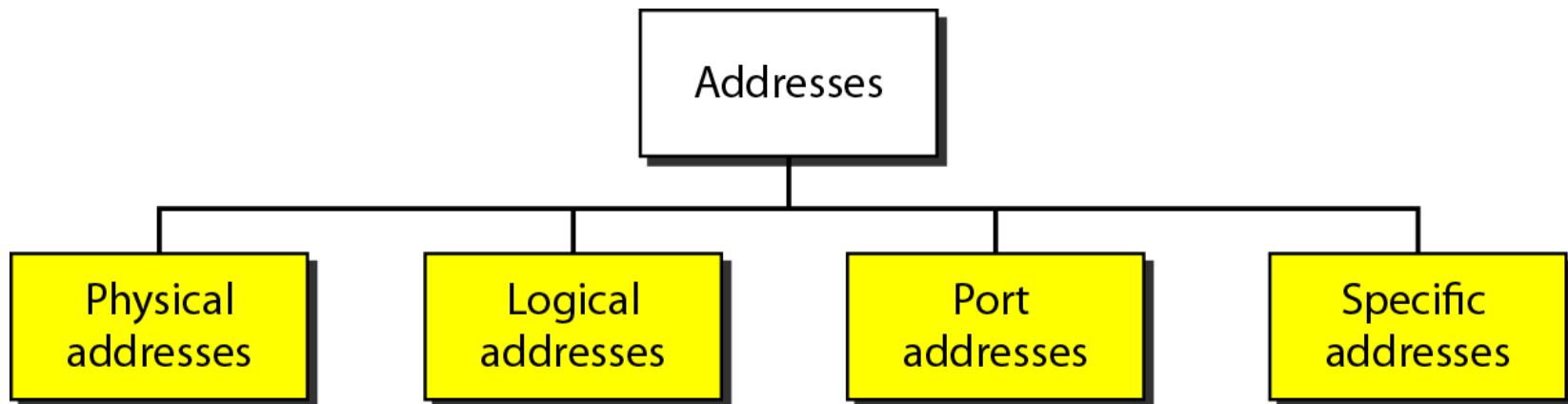
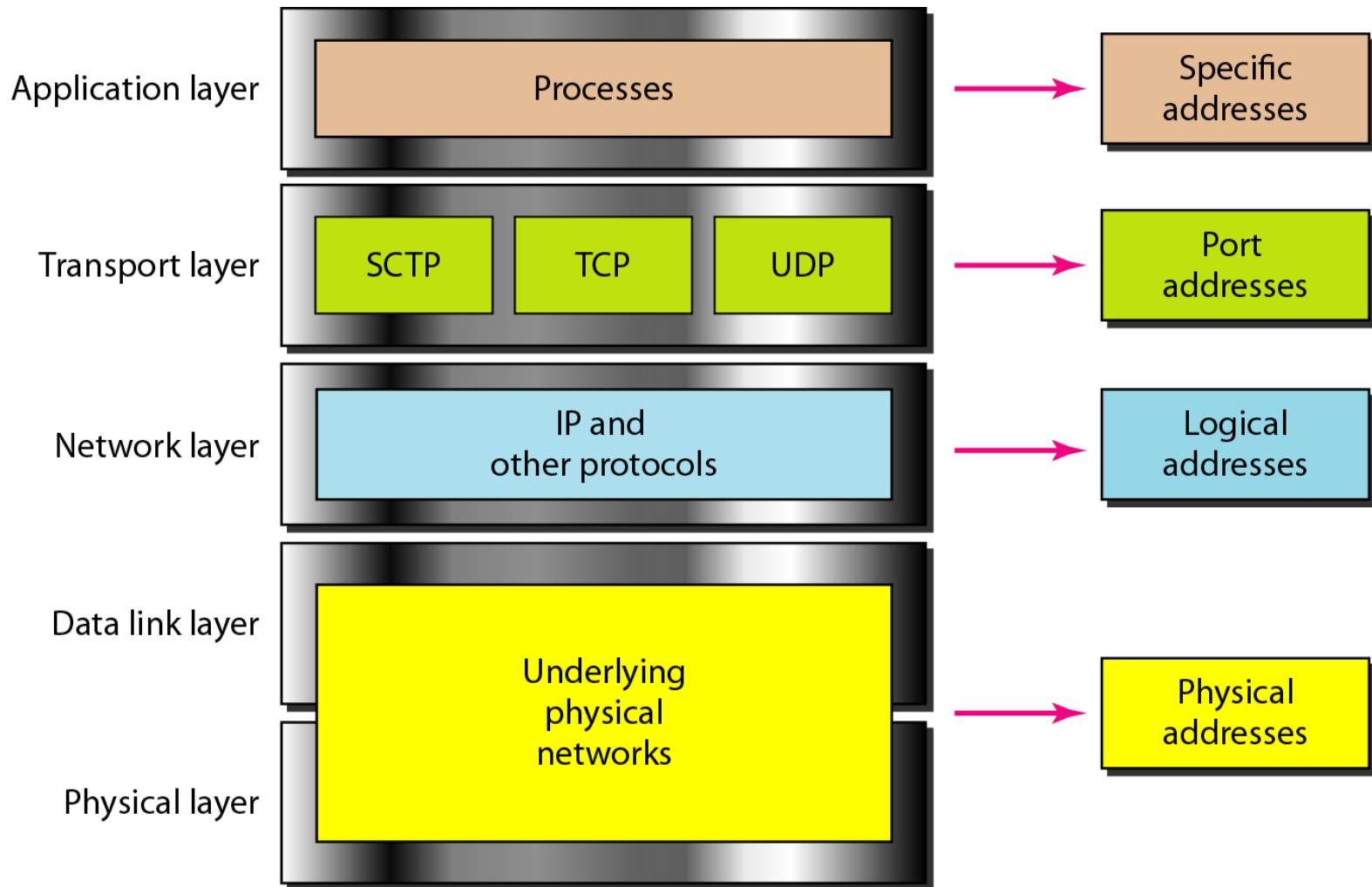
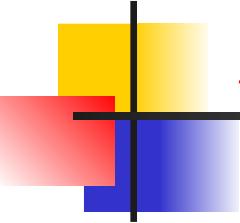


Figure 2.18 Relationship of layers and addresses in TCP/IP

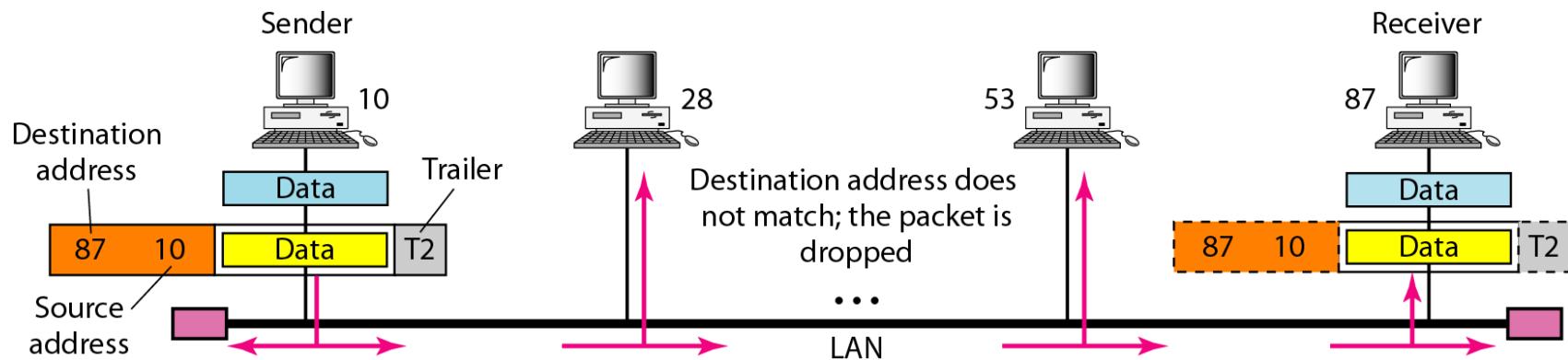


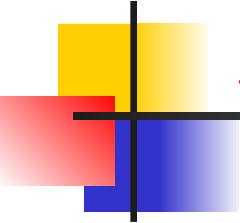


Example 2.1

In Figure 2.19 a node with physical address 10 sends a frame to a node with physical address 87. The two nodes are connected by a link (bus topology LAN). As the figure shows, the computer with physical address 10 is the sender, and the computer with physical address 87 is the receiver.

Figure 2.19 Physical addresses



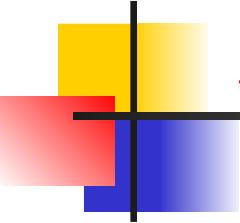


Example 2.2

*As we will see in Chapter 13, most local-area networks use a **48-bit** (6-byte) physical address written as 12 hexadecimal digits; every byte (2 hexadecimal digits) is separated by a colon, as shown below:*

07:01:02:01:2C:4B

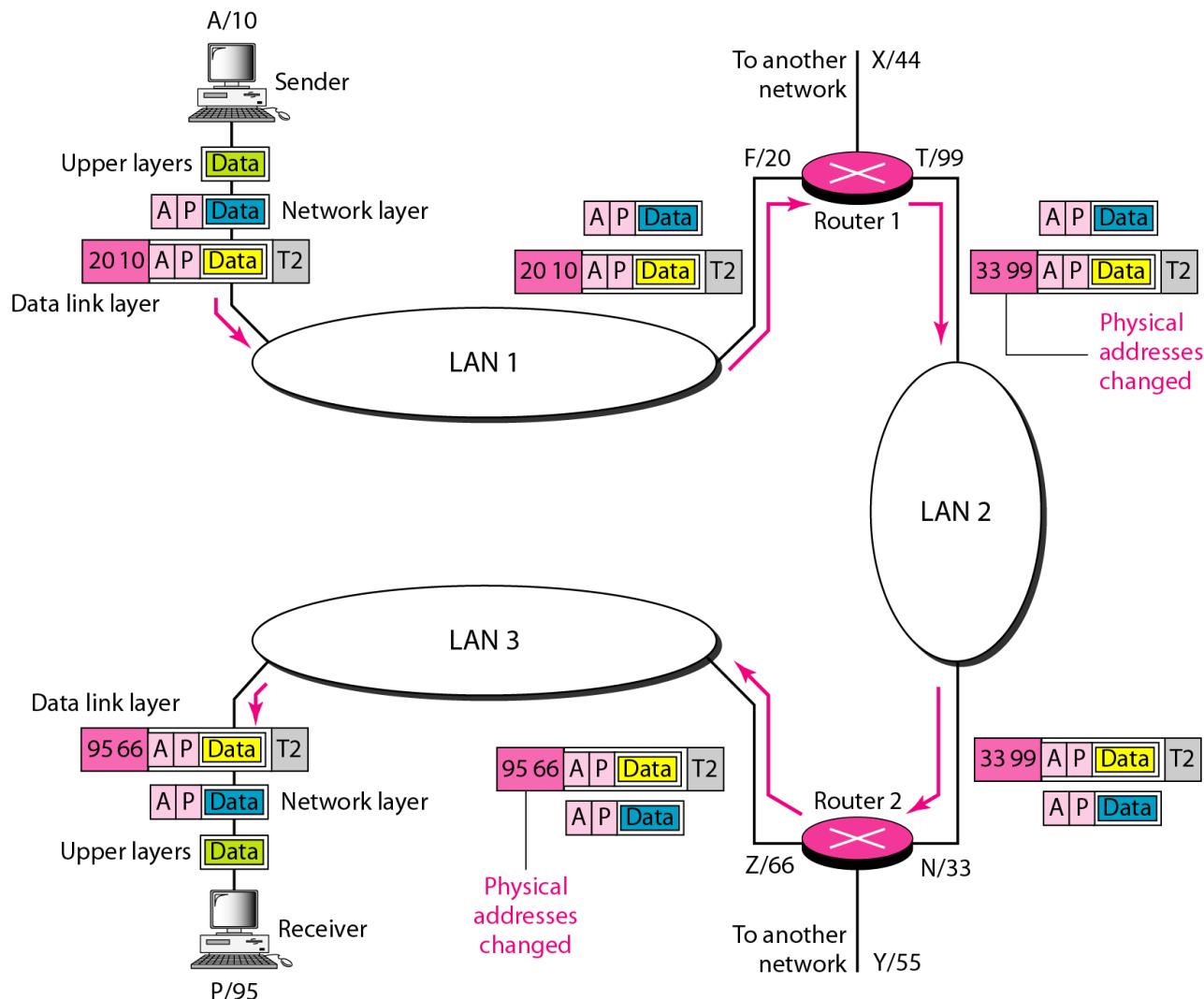
A 6-byte (12 hexadecimal digits) physical address.

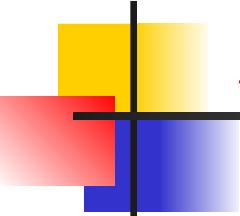


Example 2.3

Figure 2.20 shows a part of an internet with two routers connecting three LANs. Each device (computer or router) has a pair of addresses (logical and physical) for each connection. In this case, each computer is connected to only one link and therefore has only one pair of addresses. Each router, however, is connected to three networks (only two are shown in the figure). So each router has three pairs of addresses, one for each connection.

Figure 2.20 IP addresses

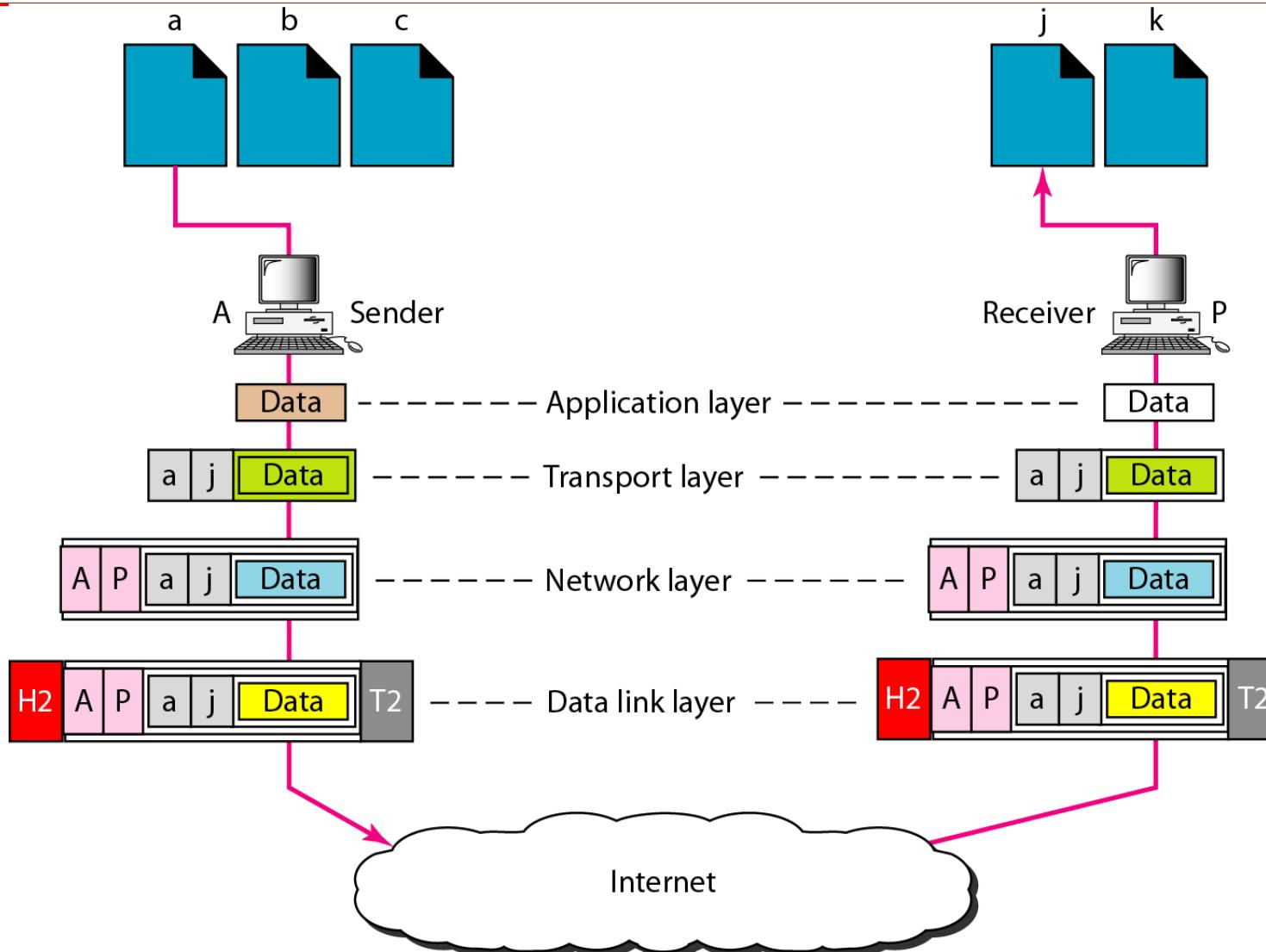


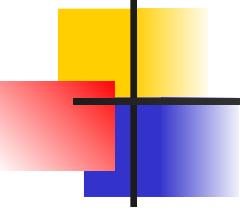


Example 2.4

Figure 2.21 shows two computers communicating via the Internet. The sending computer is running three processes at this time with port addresses a, b, and c. The receiving computer is running two processes at this time with port addresses j and k. Process a in the sending computer needs to communicate with process j in the receiving computer. Note that although physical addresses change from hop to hop, logical and port addresses remain the same from the source to destination.

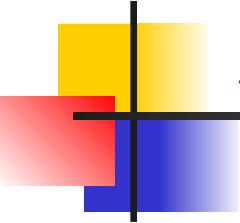
Figure 2.21 Port addresses





Note

**The physical addresses will change from hop to hop,
but the logical addresses usually remain the same.**

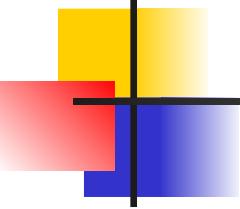


Example 2.5

As we will see in Chapter 23, a port address is a 16-bit address represented by one decimal number as shown.

753

**A 16-bit port address represented
as one single number.**

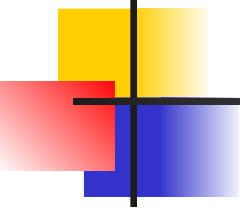


Note

**The physical addresses change from hop to hop,
but the logical and port addresses usually remain the same.**

Chapter 3

Data and Signals



Note

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

3-1 ANALOG AND DIGITAL

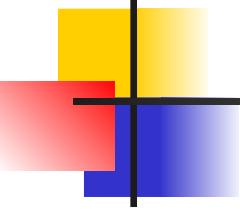
*Data can be **analog** or **digital**. The term **analog data** refers to information that is continuous; **digital data** refers to information that has discrete states. Analog data take on continuous values. Digital data take on discrete values.*

Topics discussed in this section:

Analog and Digital Data

Analog and Digital Signals

Periodic and Nonperiodic Signals

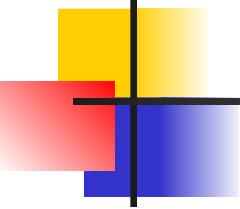


Note

Data can be analog or digital.

Analog data are continuous and take continuous values.

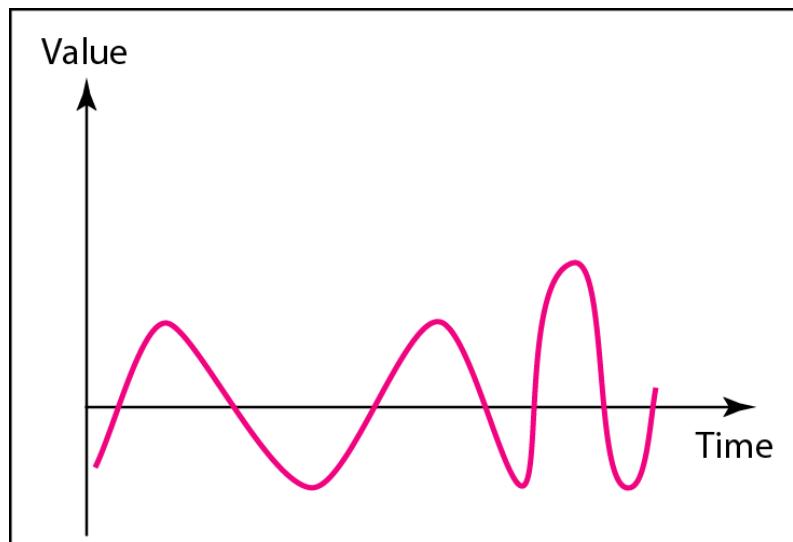
Digital data have discrete states and take discrete values.



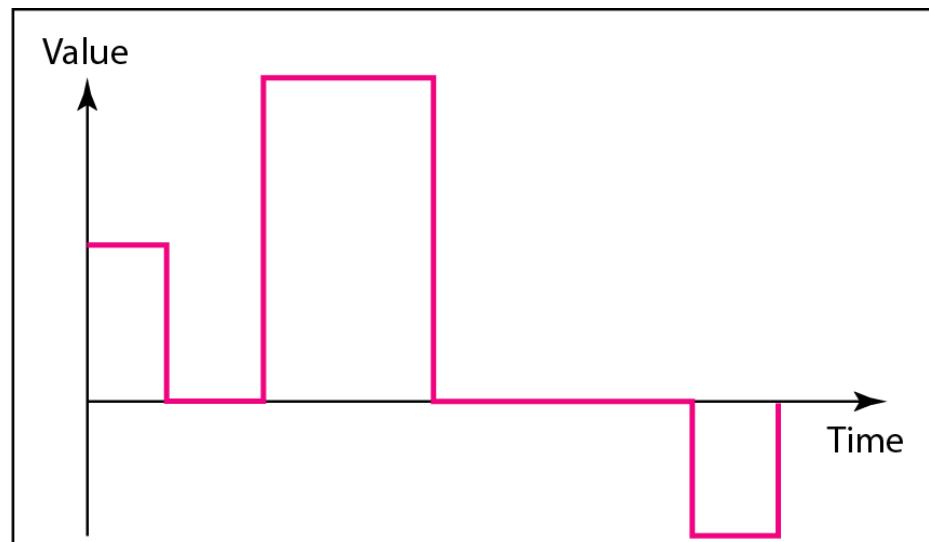
Note

Signals can be analog or digital.
Analog signals can have an infinite number of values in a range; digital signals can have only a limited number of values.

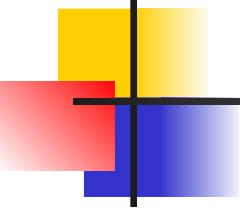
Figure 3.1 Comparison of analog and digital signals



a. Analog signal



b. Digital signal



Note

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

3-2 PERIODIC ANALOG SIGNALS

*Periodic analog signals can be classified as **simple** or **composite**. A simple periodic analog signal, a **sine wave**, cannot be decomposed into simpler signals. A composite periodic analog signal is composed of multiple sine waves.*

Topics discussed in this section:

Sine Wave

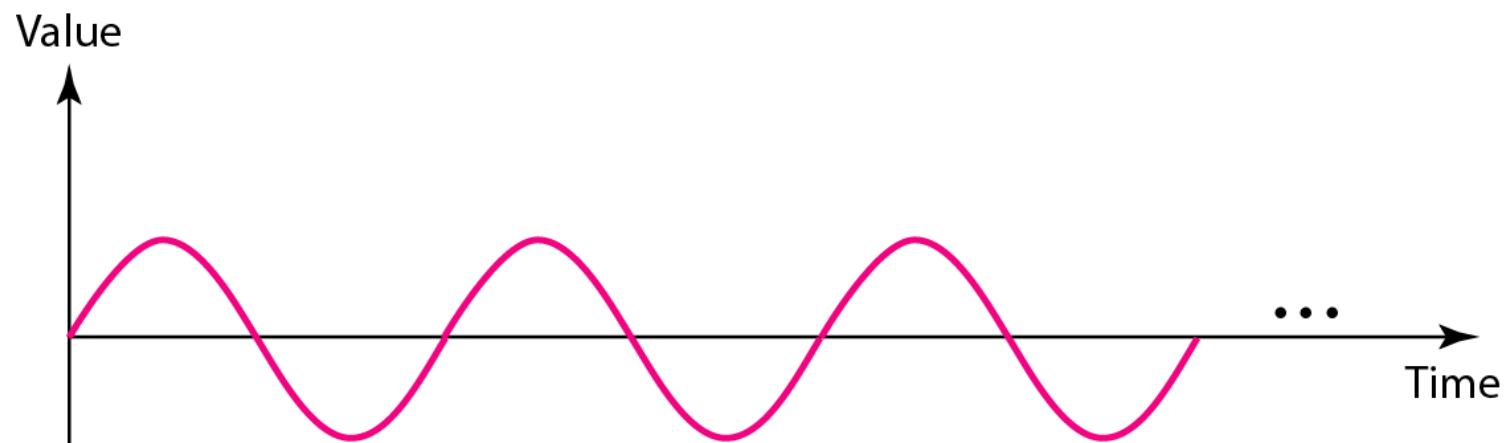
Wavelength

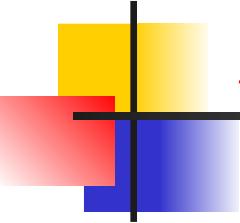
Time and Frequency Domain

Composite Signals

Bandwidth

Figure 3.2 A *sine wave*

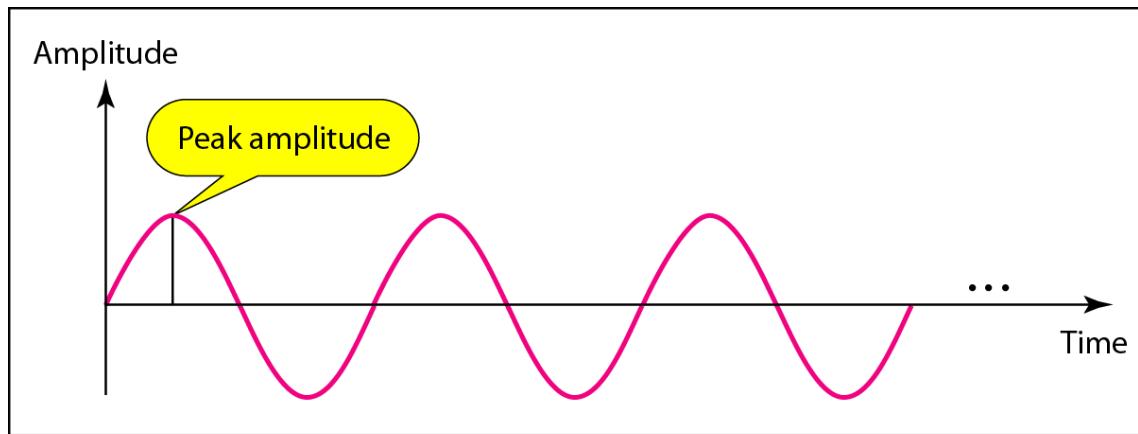




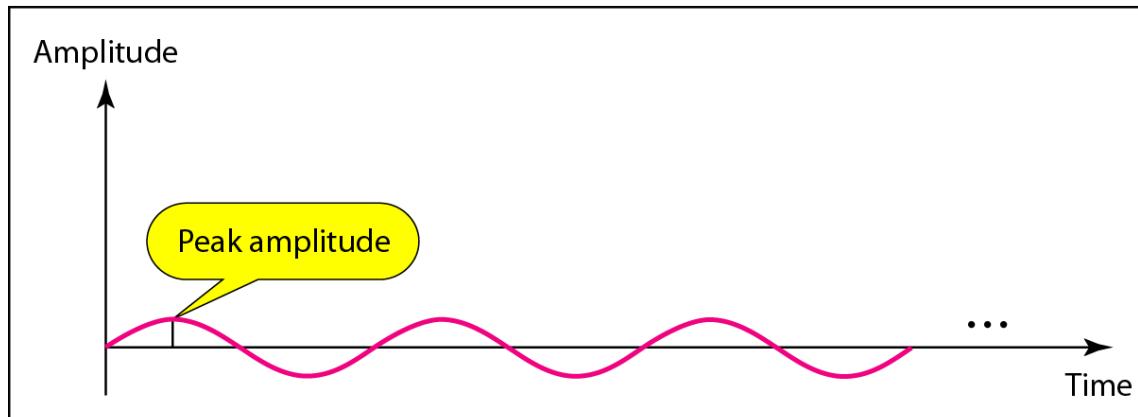
Example 3.1

The power in your house can be represented by a sine wave with a peak amplitude of 155 to 170 V. However, it is common knowledge that the voltage of the power in U.S. homes is 110 to 120 V. This discrepancy is due to the fact that these are root mean square (rms) values. The signal is squared and then the average amplitude is calculated. The peak value is equal to $2^{\frac{1}{2}} \times \text{rms}$ value.

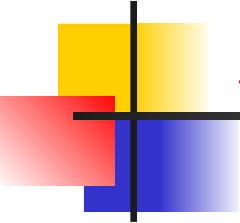
Figure 3.3 *Two signals with the same phase and frequency, but different amplitudes*



a. A signal with high peak amplitude

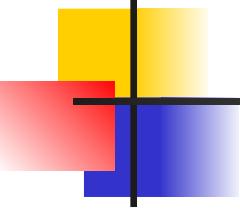


b. A signal with low peak amplitude



Example 3.2

The voltage of a battery is a constant; this constant value can be considered a sine wave, as we will see later. For example, the peak value of an AA battery is normally 1.5 V.

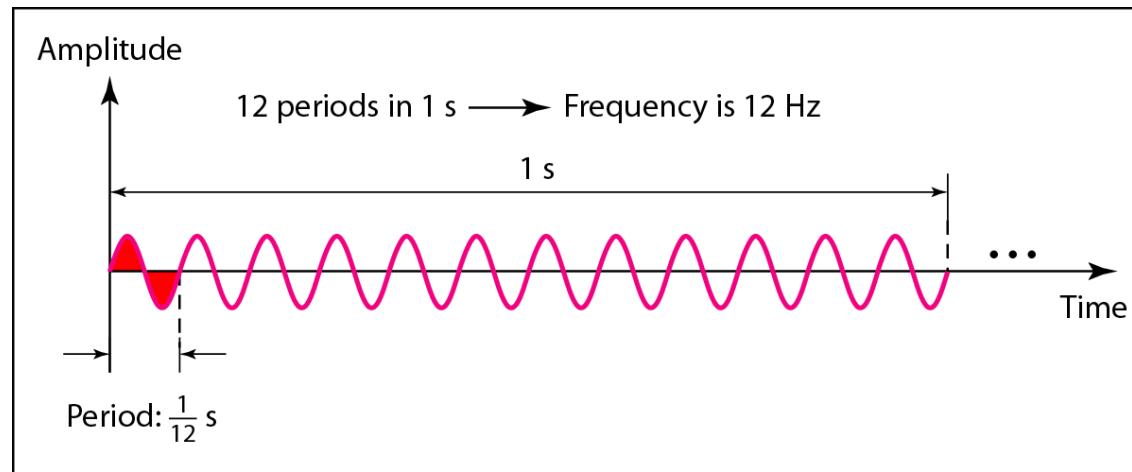


Note

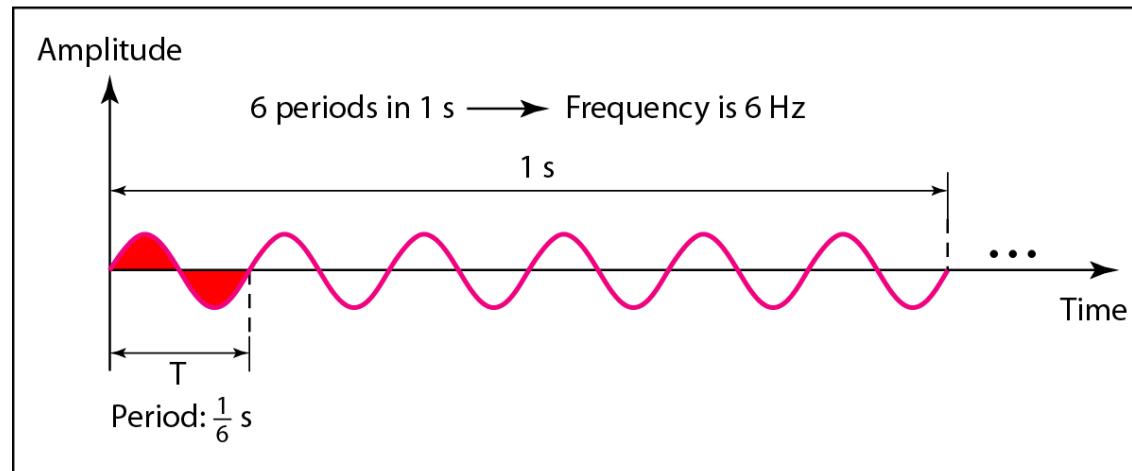
Frequency and period are the inverse of each other.

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

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



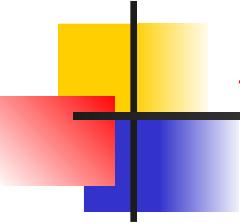
a. A signal with a frequency of 12 Hz



b. A signal with a frequency of 6 Hz

Table 3.1 *Units of period and frequency*

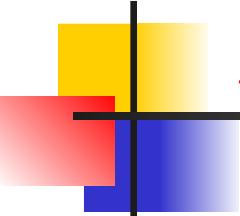
<i>Unit</i>	<i>Equivalent</i>	<i>Unit</i>	<i>Equivalent</i>
Seconds (s)	1 s	Hertz (Hz)	1 Hz
Milliseconds (ms)	10^{-3} s	Kilohertz (kHz)	10^3 Hz
Microseconds (μ s)	10^{-6} s	Megahertz (MHz)	10^6 Hz
Nanoseconds (ns)	10^{-9} s	Gigahertz (GHz)	10^9 Hz
Picoseconds (ps)	10^{-12} s	Terahertz (THz)	10^{12} Hz



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:*

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



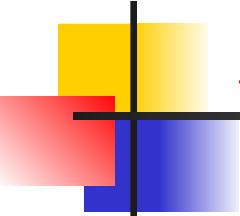
Example 3.4

Express a period of 100 ms in microseconds.

Solution

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

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



Example 3.5

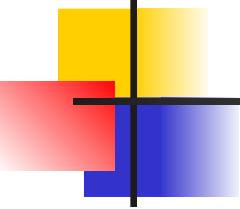
The period of a signal is 100 ms. What is its frequency in kilohertz?

Solution

First we change 100 ms to seconds, and then we calculate the frequency from the period (1 Hz = 10⁻³ kHz).

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

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

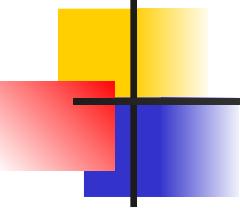


Note

Frequency is the rate of change with respect to time.

Change in a short span of time means high frequency.

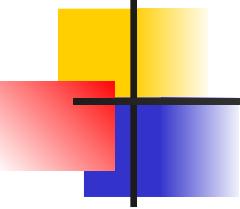
Change over a long span of time means low frequency.



Note

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

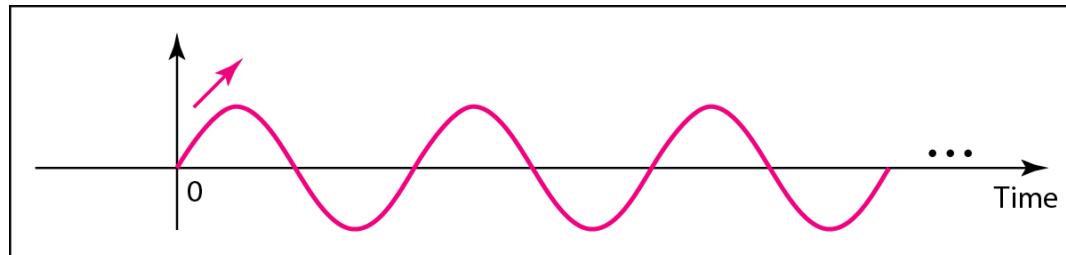
If a signal changes instantaneously, its frequency is infinite.



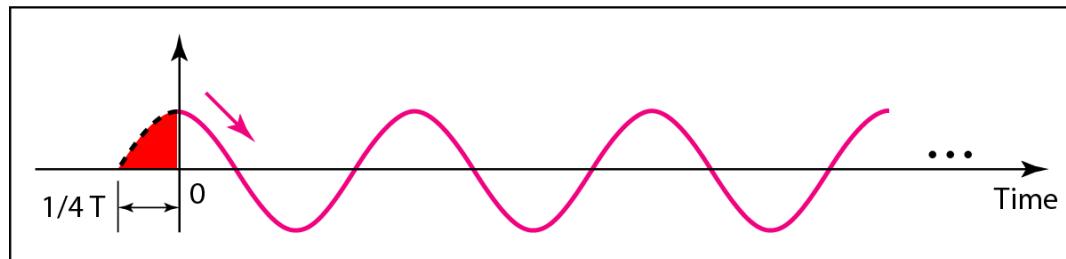
Note

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

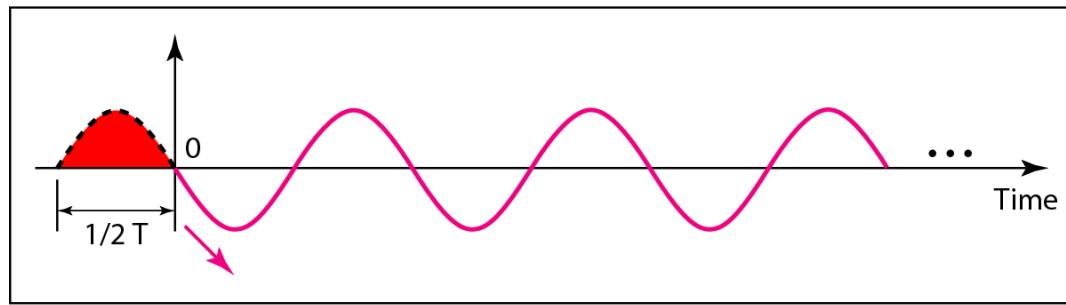
Figure 3.5 *Three sine waves with the same amplitude and frequency, but different phases*



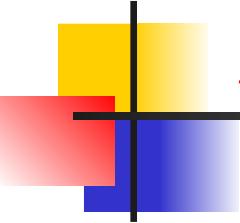
a. 0 degrees



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 3.6 *Wavelength and period*

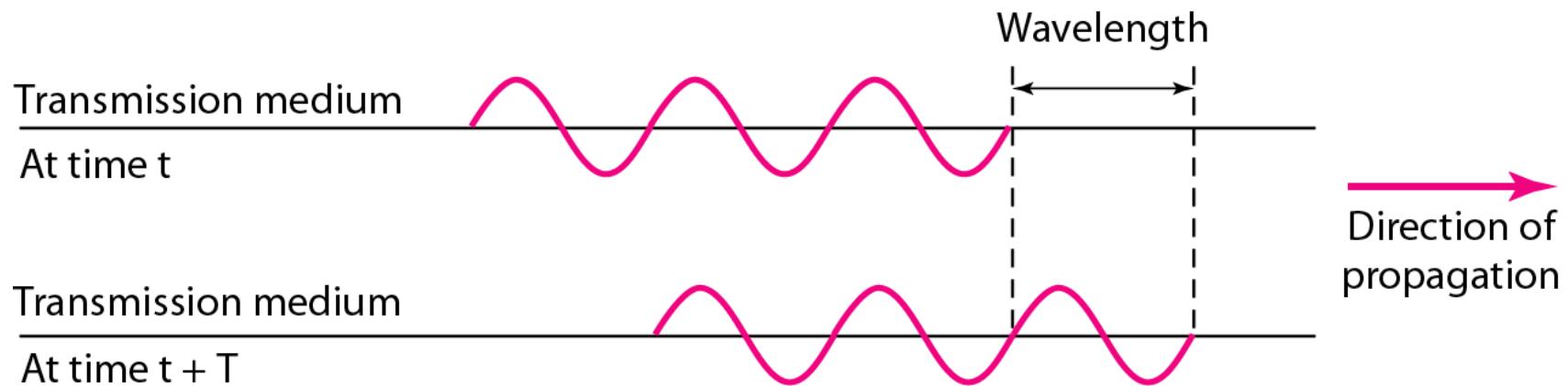
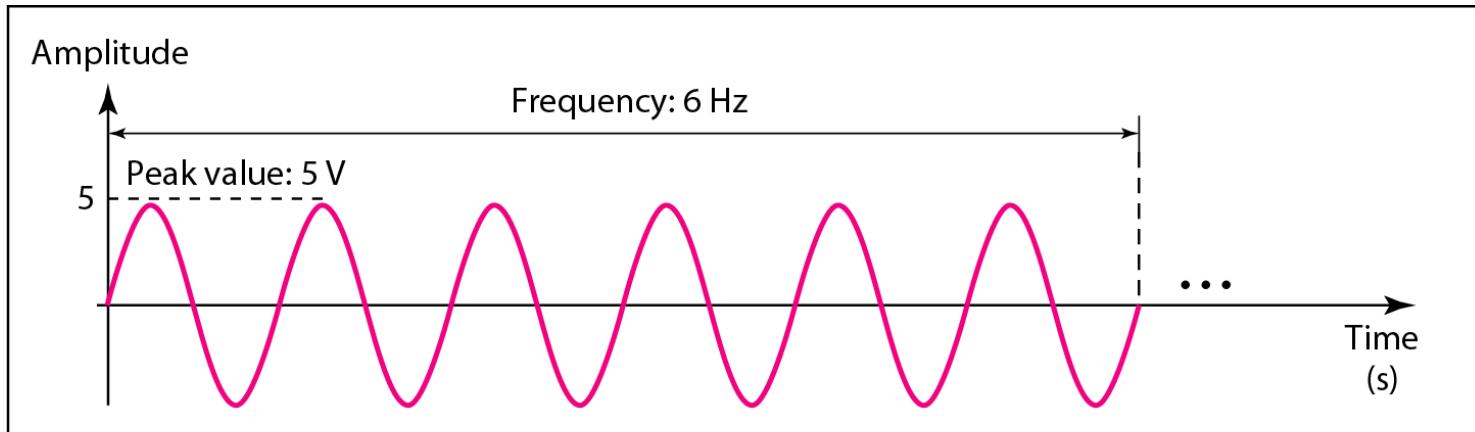
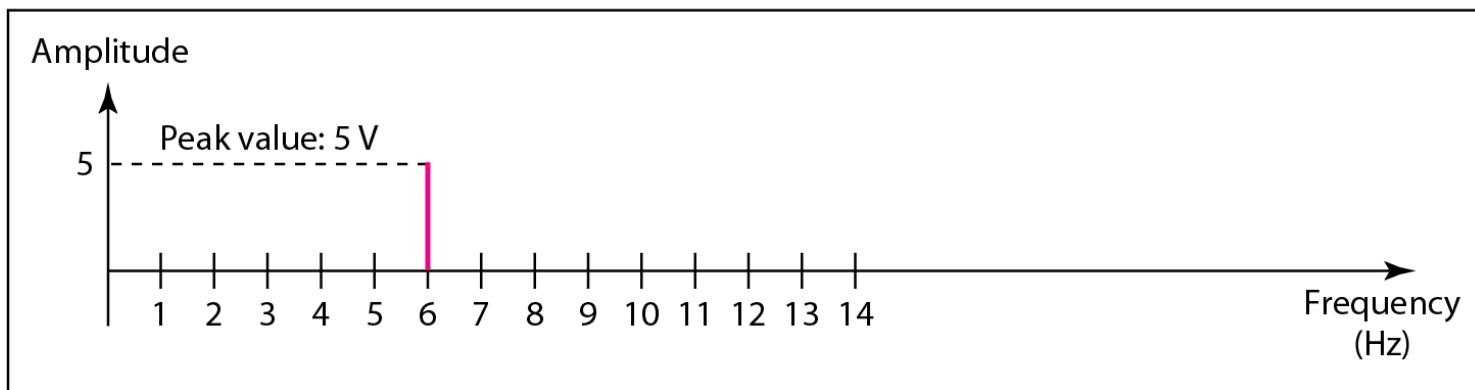


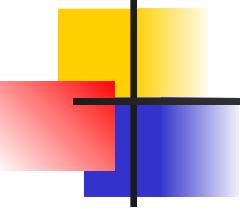
Figure 3.7 *The time-domain and frequency-domain plots of a sine wave*



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

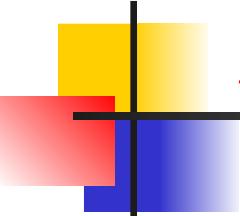


b. The same sine wave in the frequency domain (peak value: 5 V, frequency: 6 Hz)



Note

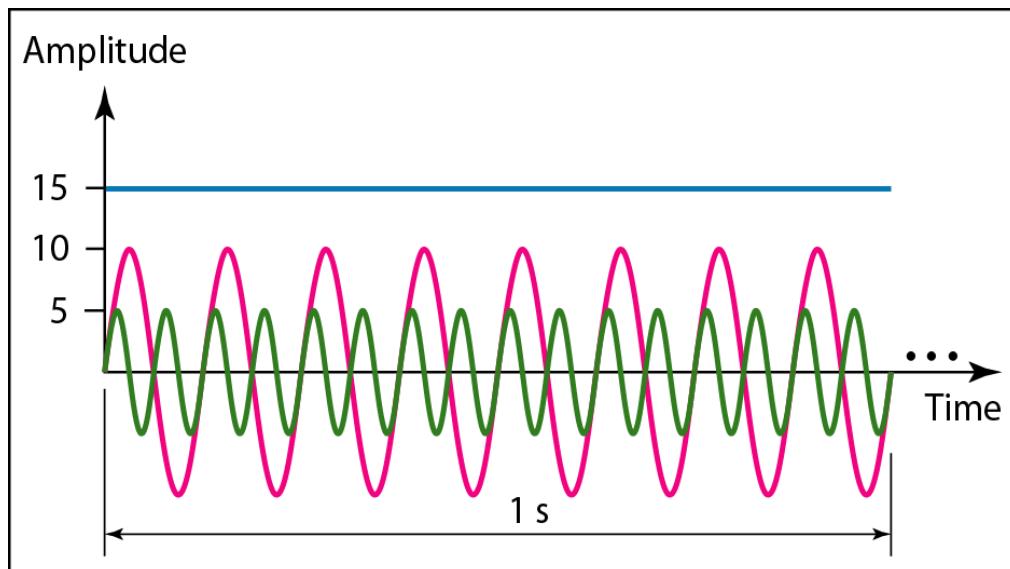
A complete sine wave in the time domain can be represented by one single spike in the frequency domain.



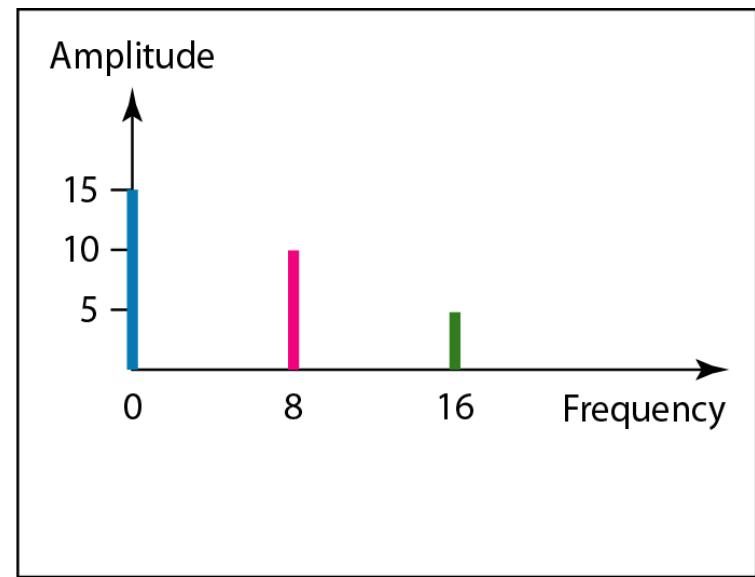
Example 3.7

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.

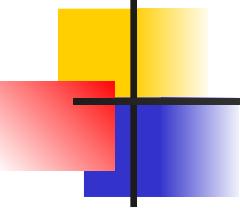
Figure 3.8 *The time domain and frequency domain of three sine waves*



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

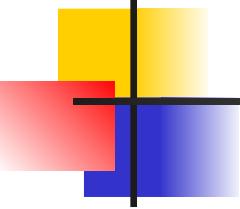


b. Frequency-domain representation of the same three signals



Note

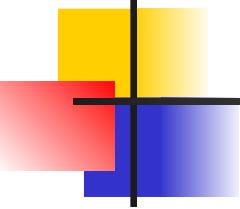
A single-frequency sine wave is not useful in data communications; we need to send a composite signal, a signal made of many simple sine waves.



Note

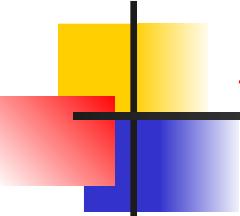
According to Fourier analysis, any composite signal is a combination of simple sine waves with different frequencies, amplitudes, and phases.

Fourier analysis is discussed in Appendix C.



Note

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



Example 3.8

Figure 3.9 shows a periodic composite signal with frequency f . This type of signal is not typical of those found in data communications. We can consider it to be three alarm systems, each with a different frequency. The analysis of this signal can give us a good understanding of how to decompose signals.

Figure 3.9 A composite periodic signal

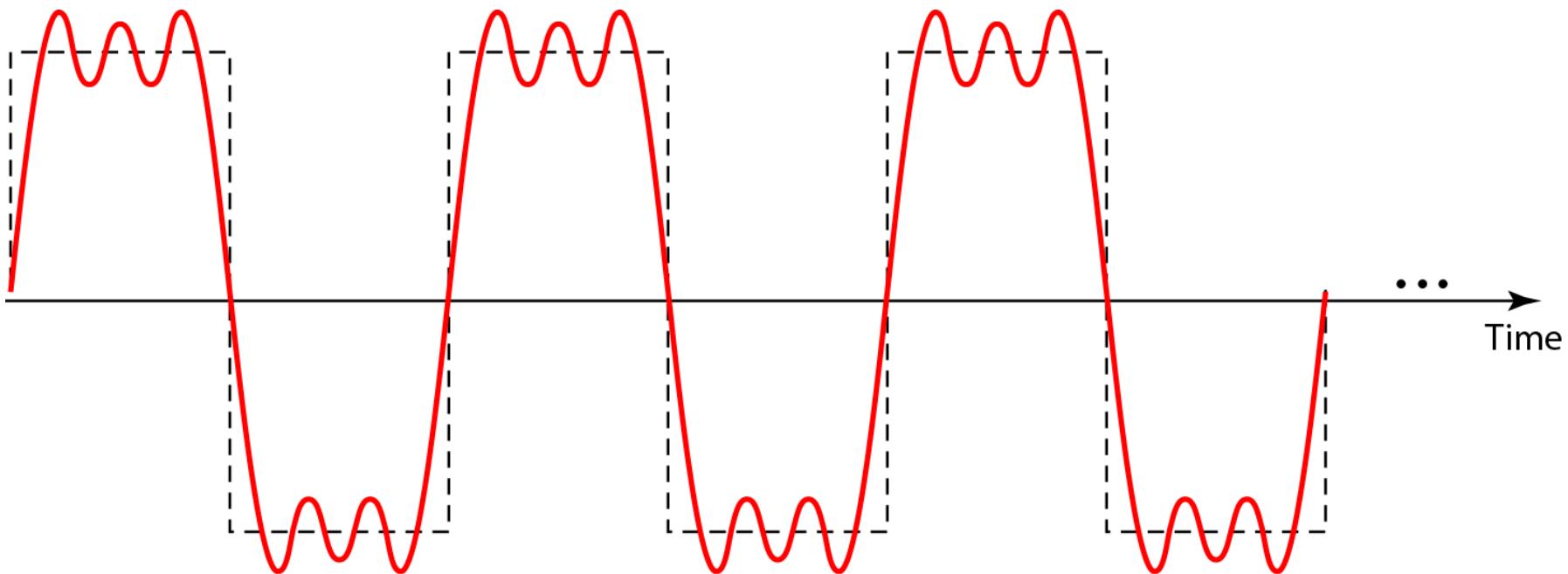
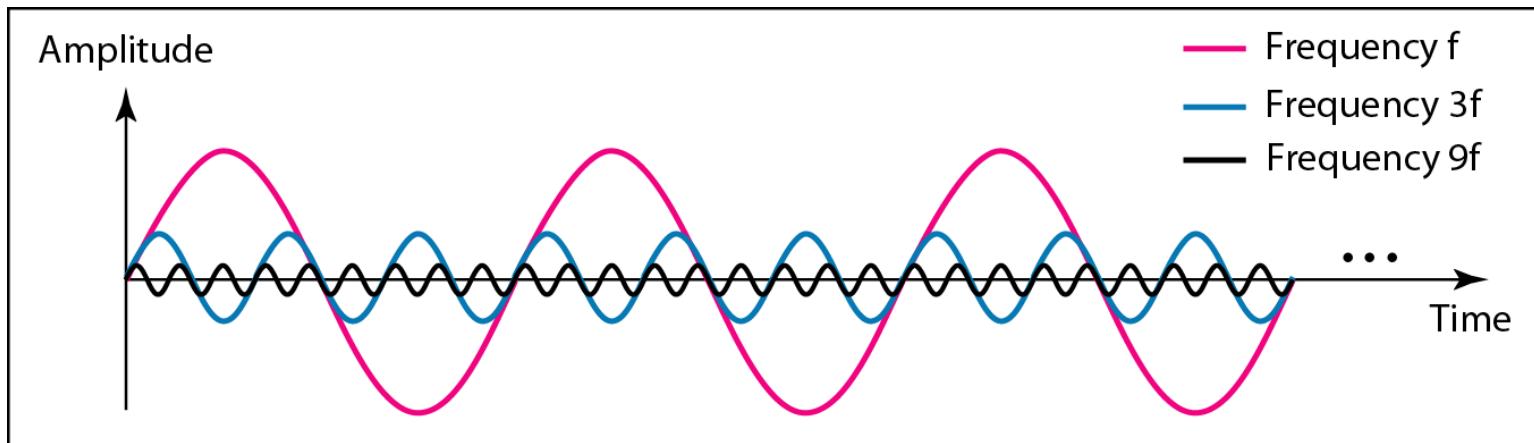
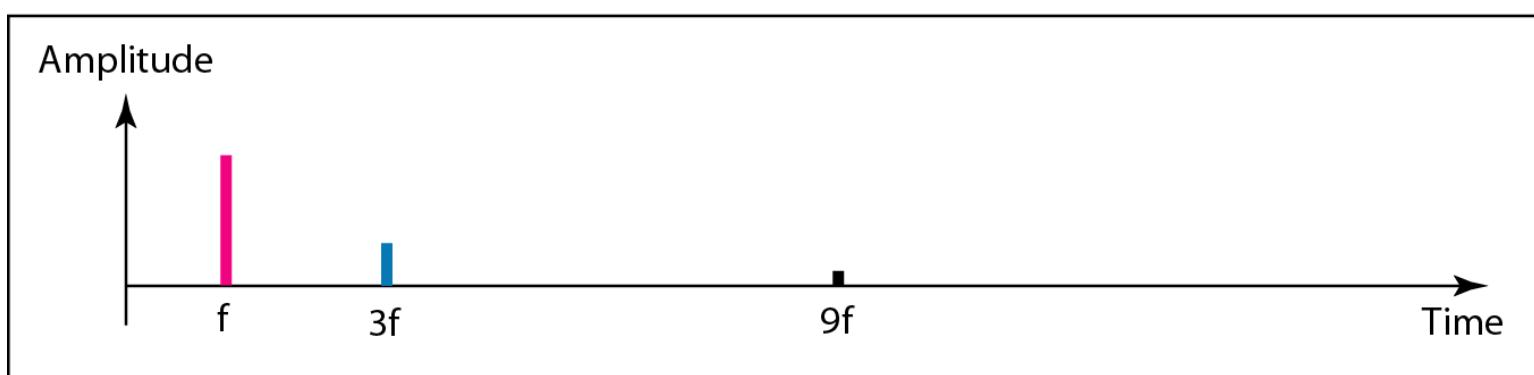


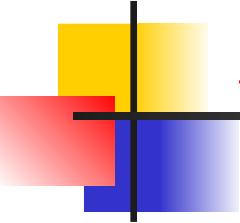
Figure 3.10 *Decomposition of a composite periodic signal in the time and frequency domains*



a. Time-domain decomposition of a composite signal



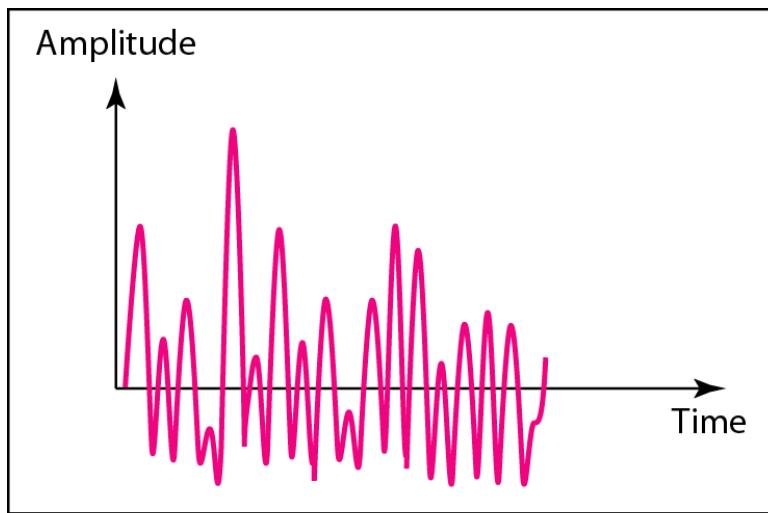
b. Frequency-domain decomposition of the composite signal



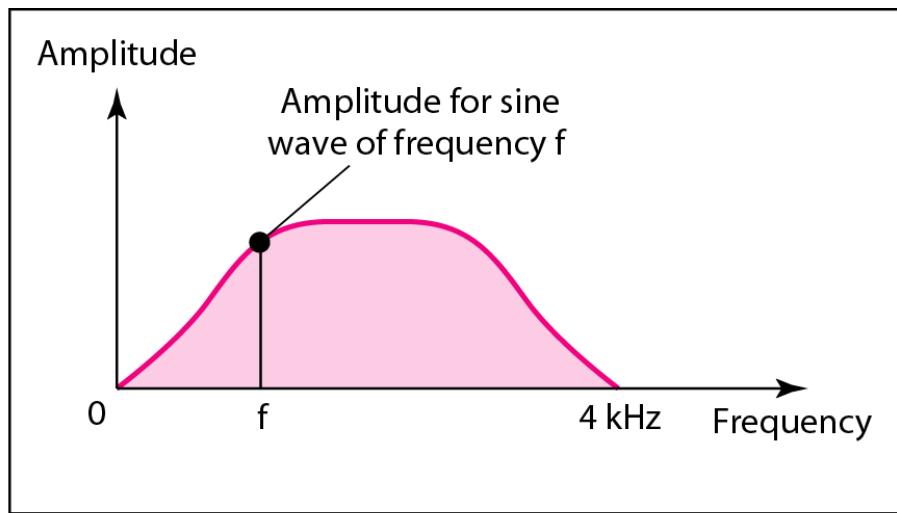
Example 3.9

Figure 3.11 shows a nonperiodic composite signal. It can be the signal created by a microphone or a telephone set when a word or two is pronounced. In this case, the composite signal cannot be periodic, because that implies that we are repeating the same word or words with exactly the same tone.

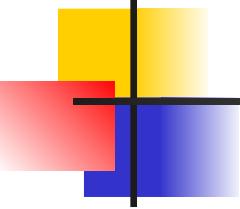
Figure 3.11 *The time and frequency domains of a nonperiodic signal*



a. Time domain



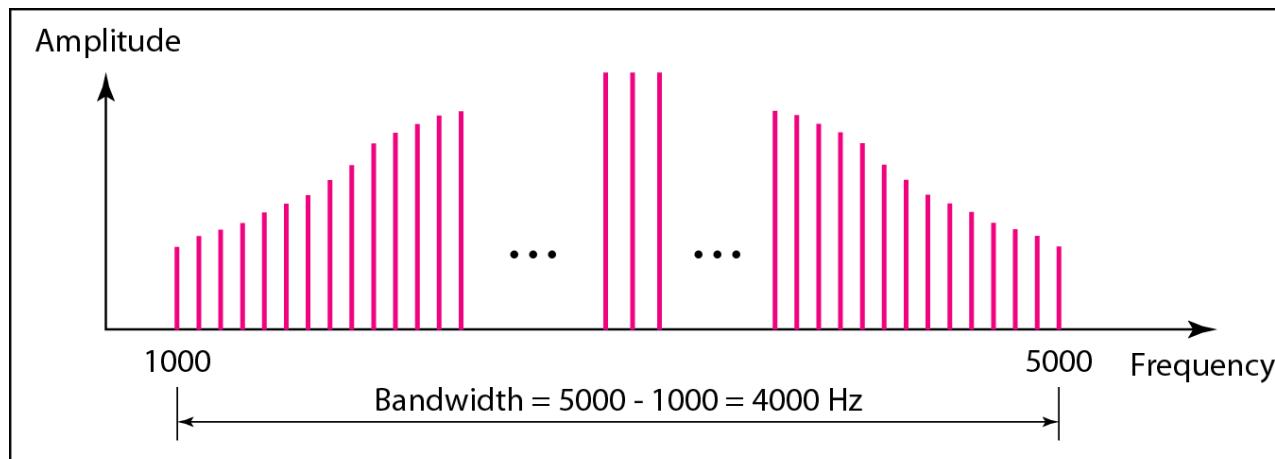
b. Frequency domain



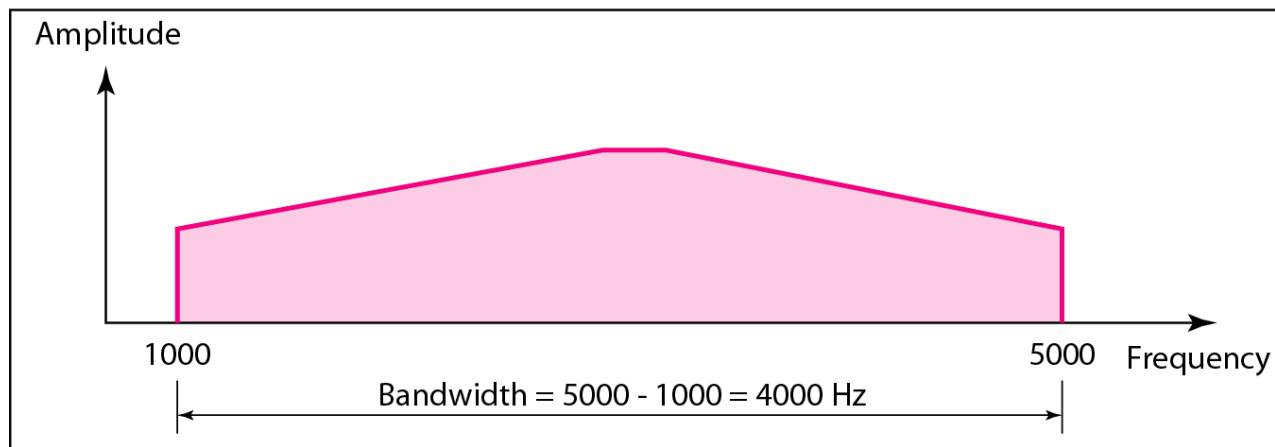
Note

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

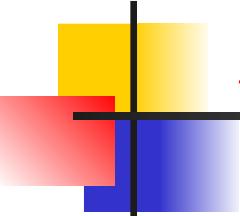
Figure 3.12 *The bandwidth of periodic and nonperiodic composite signals*



a. Bandwidth of a periodic signal



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, assuming all components have a maximum amplitude of 10 V.

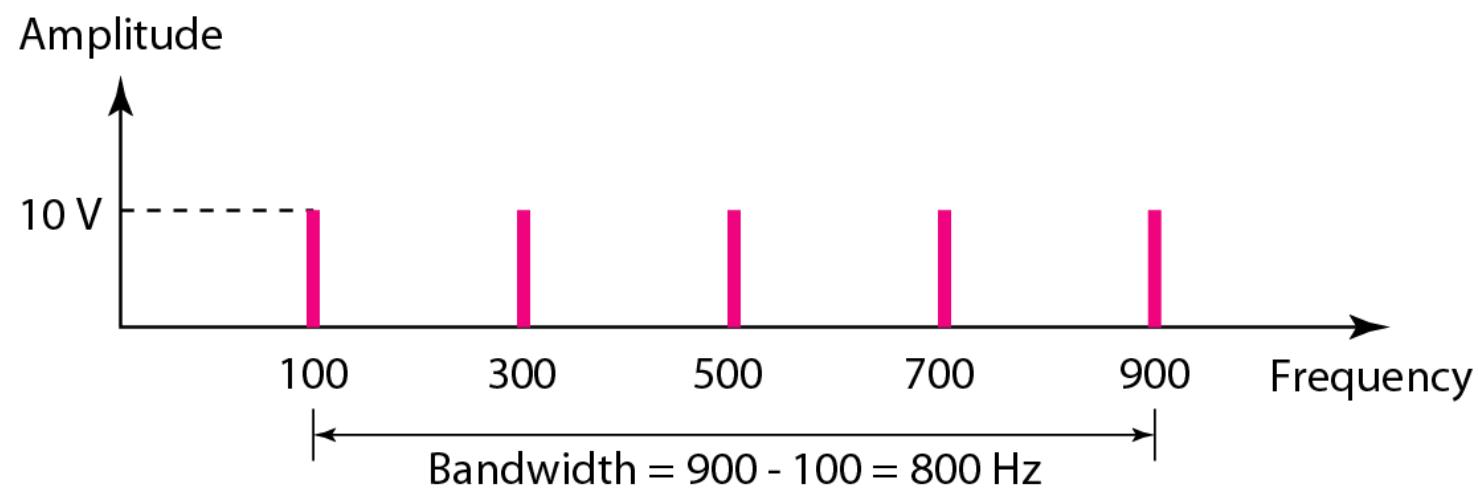
Solution

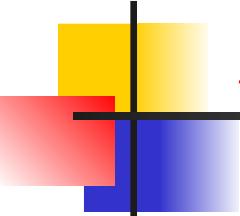
Let f_h be the highest frequency, f_l the lowest frequency, and B the bandwidth. Then

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

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 3.11

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

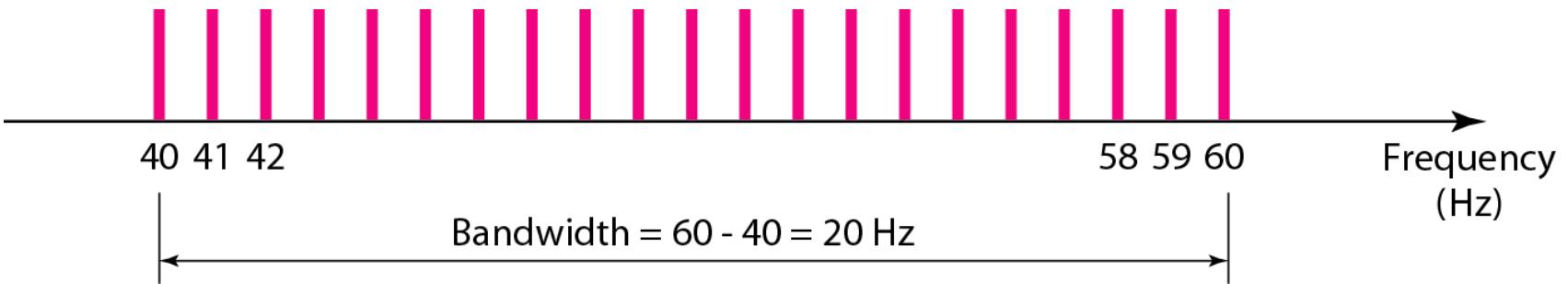
Solution

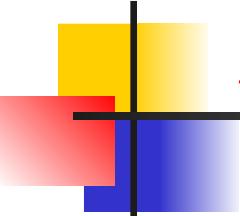
Let f_h be the highest frequency, f_l the lowest frequency, and B the bandwidth. Then

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

The spectrum contains all integer frequencies. We show this by a series of spikes (see Figure 3.14).

Figure 3.14 *The bandwidth for Example 3.11*





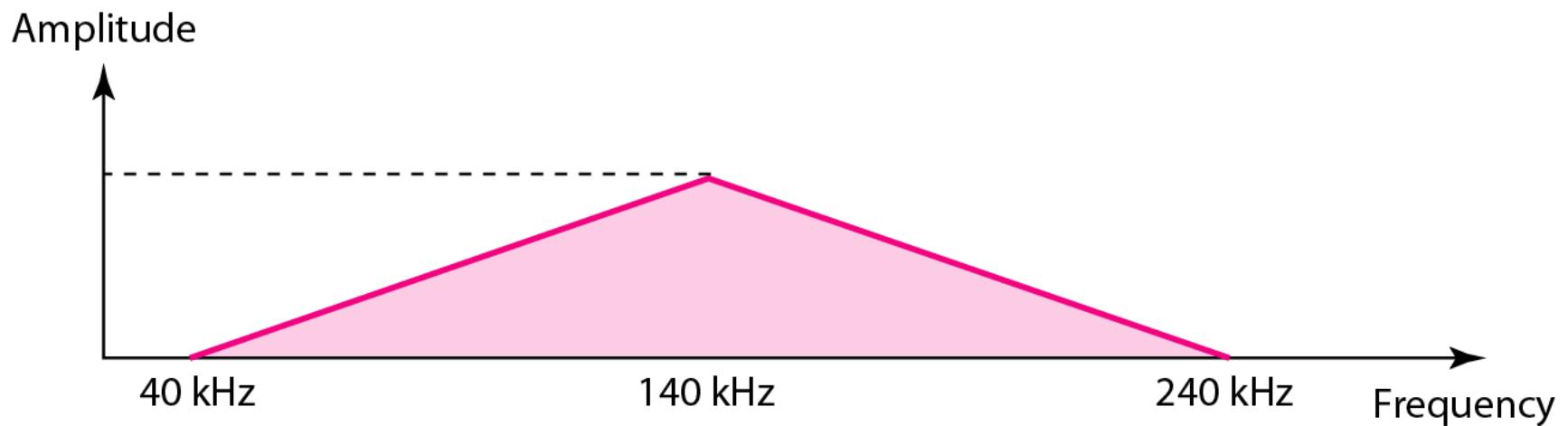
Example 3.12

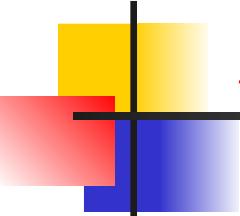
A nonperiodic composite signal has a bandwidth of 200 kHz, with a middle frequency of 140 kHz and peak amplitude of 20 V. The two extreme frequencies have an amplitude of 0. Draw the frequency domain of the signal.

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.

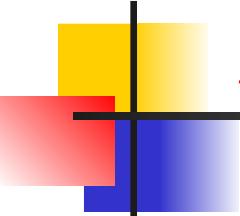
Figure 3.15 *The bandwidth for Example 3.12*





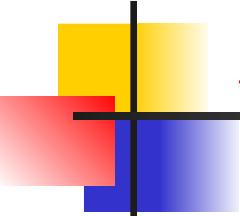
Example 3.13

An example of a nonperiodic composite signal is the signal propagated by an AM radio station. In the United States, each AM radio station is assigned a 10-kHz bandwidth. The total bandwidth dedicated to AM radio ranges from 530 to 1700 kHz. We will show the rationale behind this 10-kHz bandwidth in Chapter 5.



Example 3.14

Another example of a nonperiodic composite signal is the signal propagated by an FM radio station. In the United States, each FM radio station is assigned a 200-kHz bandwidth. The total bandwidth dedicated to FM radio ranges from 88 to 108 MHz. We will show the rationale behind this 200-kHz bandwidth in Chapter 5.



Example 3.15

Another example of a nonperiodic composite signal is the signal received by an old-fashioned analog black-and-white TV. A TV screen is made up of pixels. If we assume a resolution of 525×700 , we have 367,500 pixels per screen. If we scan the screen 30 times per second, this is $367,500 \times 30 = 11,025,000$ pixels per second. The worst-case scenario is alternating black and white pixels. We can send 2 pixels per cycle. Therefore, we need $11,025,000 / 2 = 5,512,500$ cycles per second, or Hz. The bandwidth needed is 5.5125 MHz.

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:

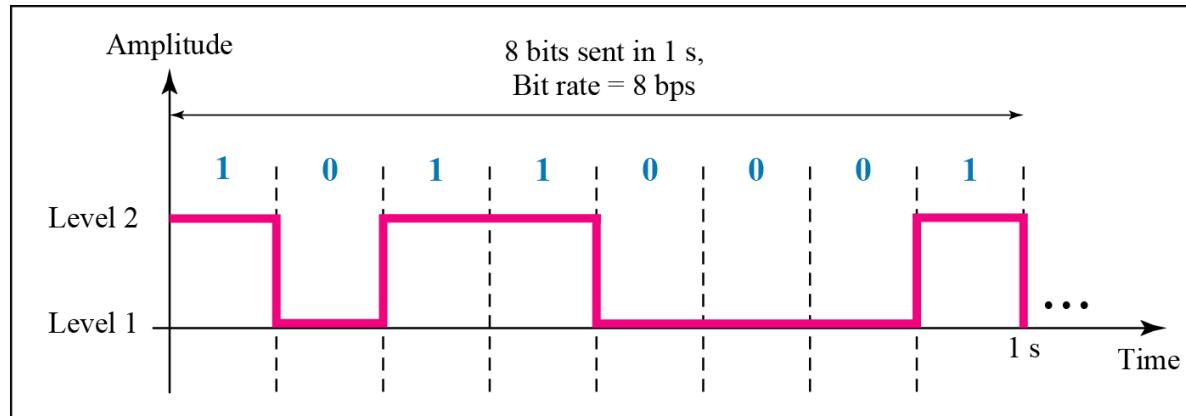
Bit Rate

Bit Length

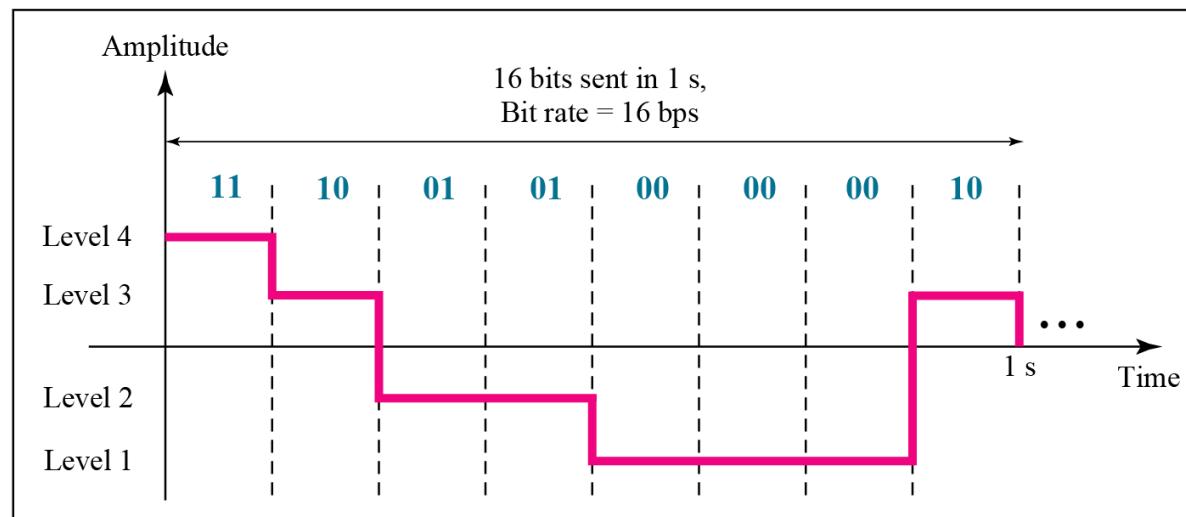
Digital Signal as a Composite Analog Signal

Application Layer

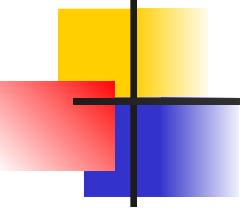
Figure 3.16 Two digital signals: one with two signal levels and the other with four signal levels



a. A digital signal with two levels

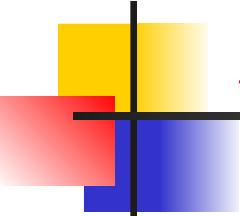


b. A digital signal with four levels



Note

Appendix C reviews information about exponential and logarithmic functions.

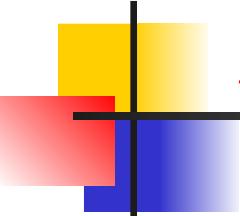


Example 3.16

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

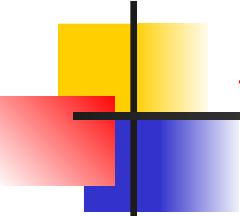
$$\text{Number of bits per level} = \log_2 8 = 3$$

Each signal level is represented by 3 bits.



Example 3.17

A digital signal has nine levels. How many bits are needed per level? We calculate the number of bits by using the formula. Each signal level is represented by 3.17 bits. However, this answer is not realistic. The number of bits sent per level needs to be an integer as well as a power of 2. For this example, 4 bits can represent one level.



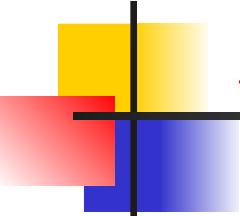
Example 3.18

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

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



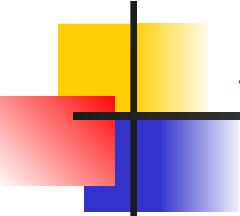
Example 3.19

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)?

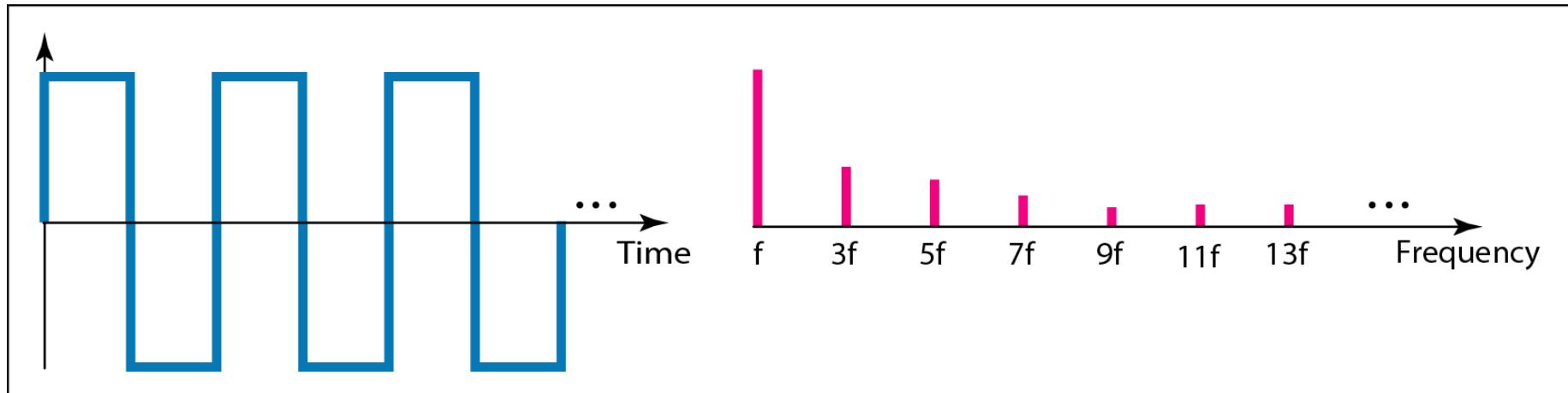
Solution

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.

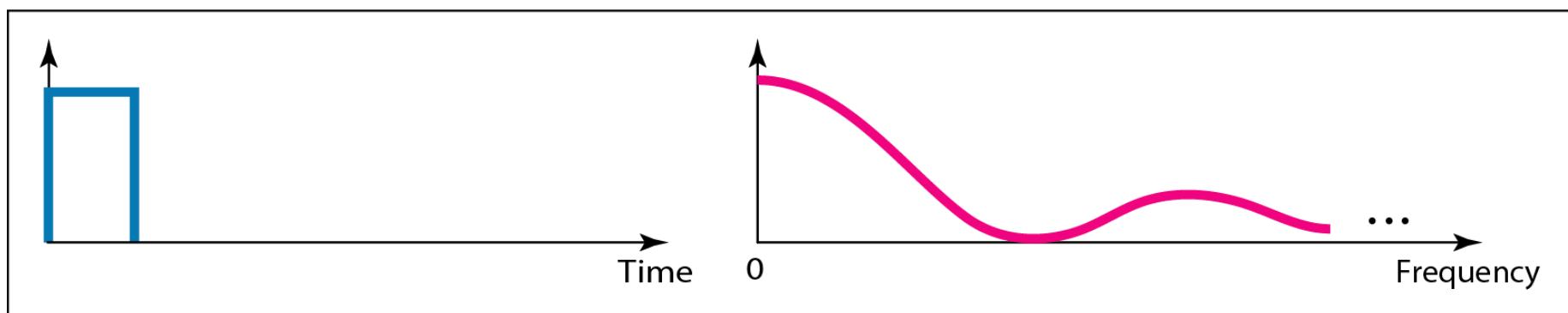
$$1920 \times 1080 \times 30 \times 24 = 1,492,992,000 \text{ or } 1.5 \text{ Gbps}$$

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

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

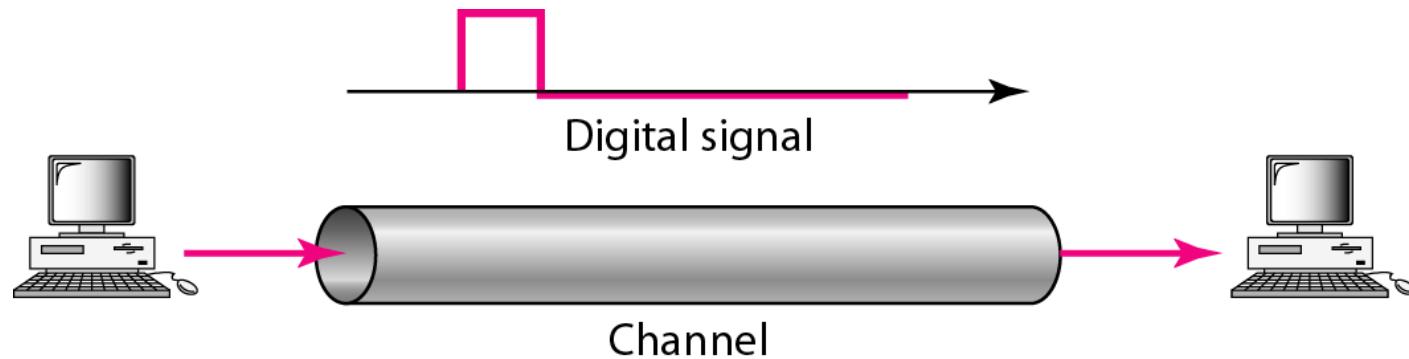


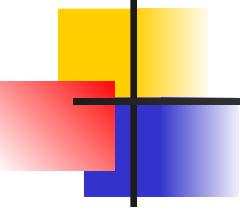
a. Time and frequency domains of periodic digital signal



b. Time and frequency domains of nonperiodic digital signal

Figure 3.18 *Baseband transmission*



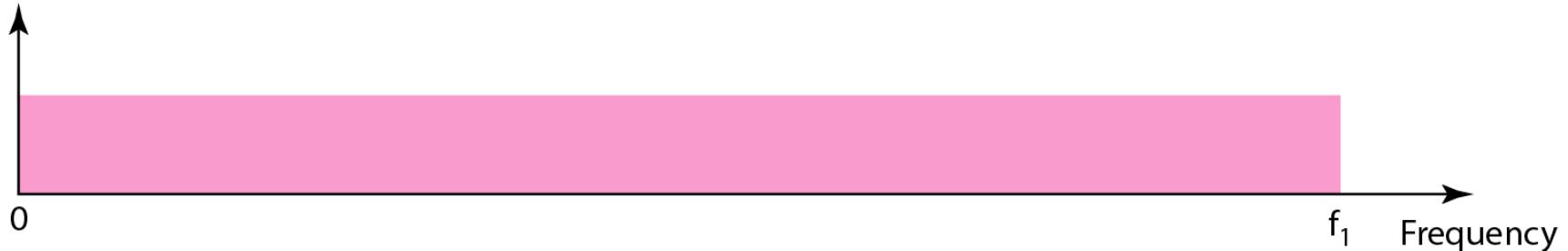


Note

A digital signal is a composite analog signal with an infinite bandwidth.

Figure 3.19 *Bandwidths of two low-pass channels*

Amplitude



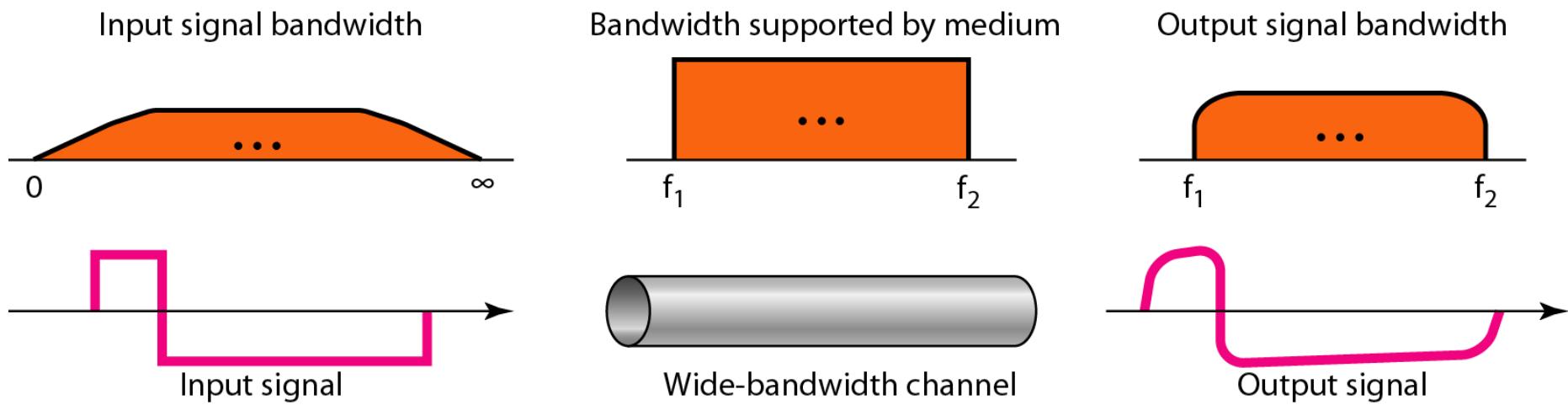
a. Low-pass channel, wide bandwidth

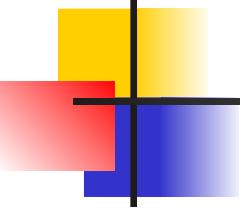
Amplitude



b. Low-pass channel, narrow bandwidth

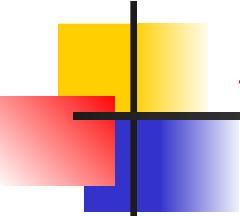
Figure 3.20 *Baseband transmission using a dedicated medium*





Note

Baseband transmission of a digital signal that preserves the shape of the digital signal is possible only if we have a low-pass channel with an infinite or very wide bandwidth.



Example 3.21

An example of a dedicated channel where the entire bandwidth of the medium is used as one single channel is a LAN. Almost every wired LAN today uses a dedicated channel for two stations communicating with each other. In a bus topology LAN with multipoint connections, only two stations can communicate with each other at each moment in time (timesharing); the other stations need to refrain from sending data. In a star topology LAN, the entire channel between each station and the hub is used for communication between these two entities. We study LANs in Chapter 14.

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

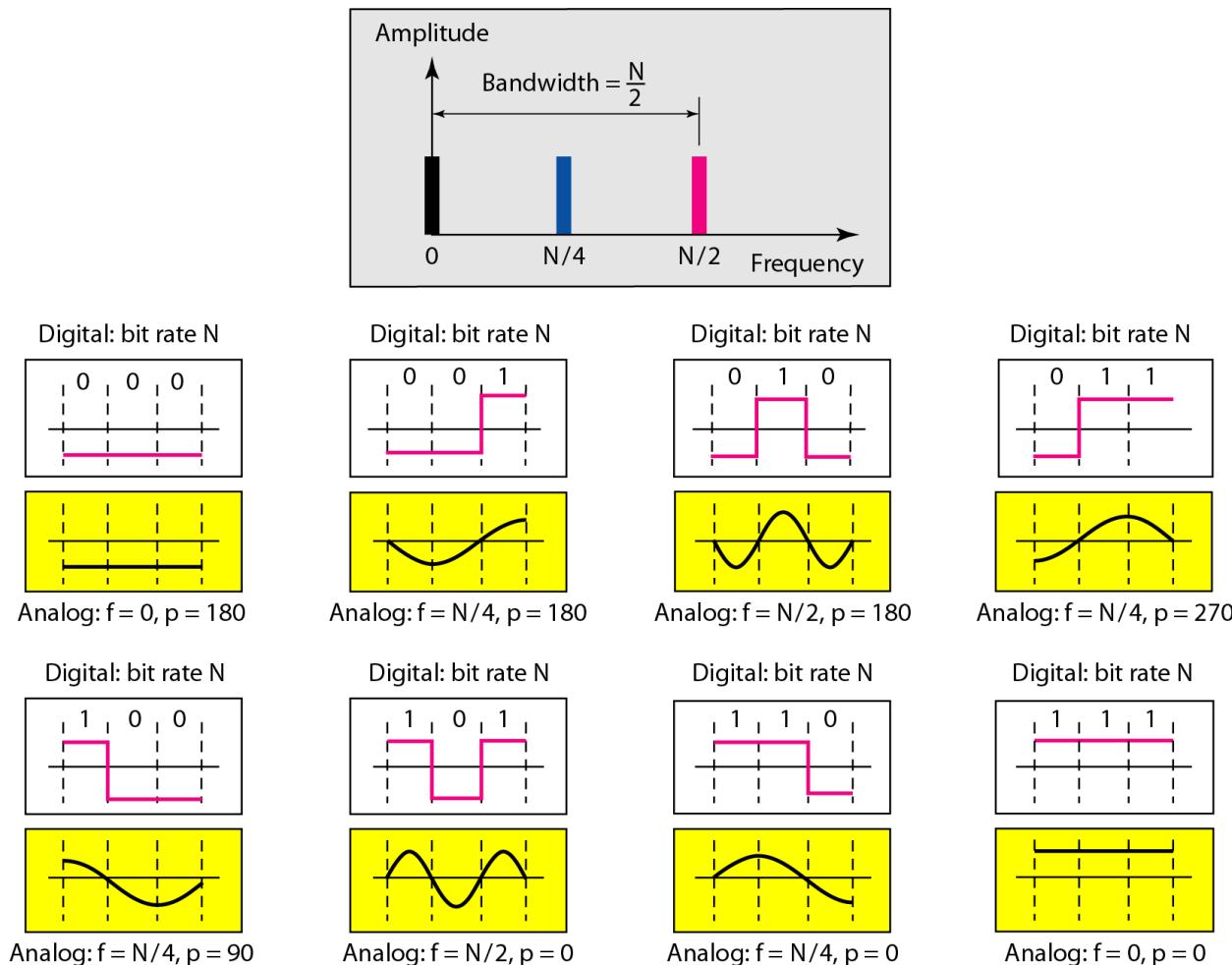
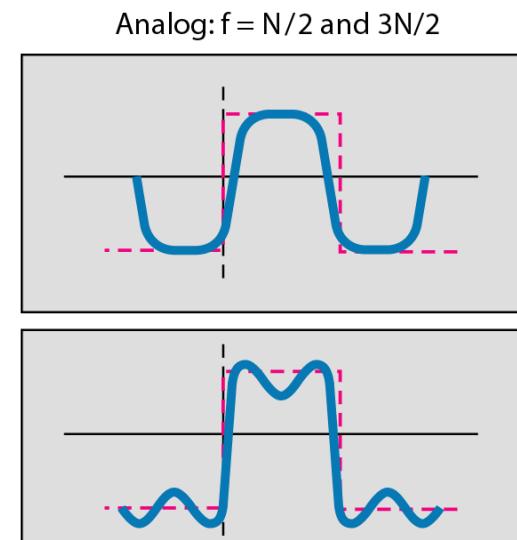
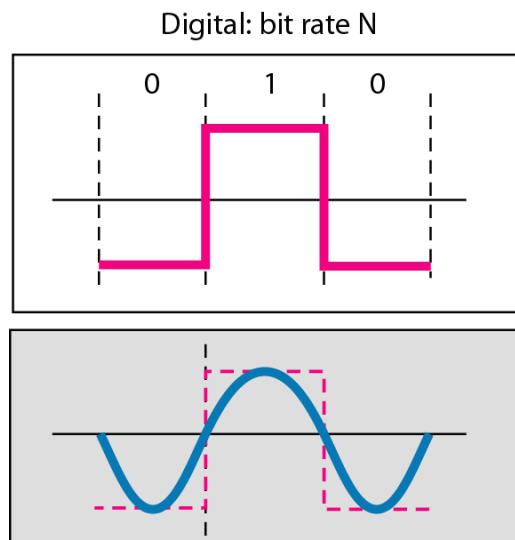
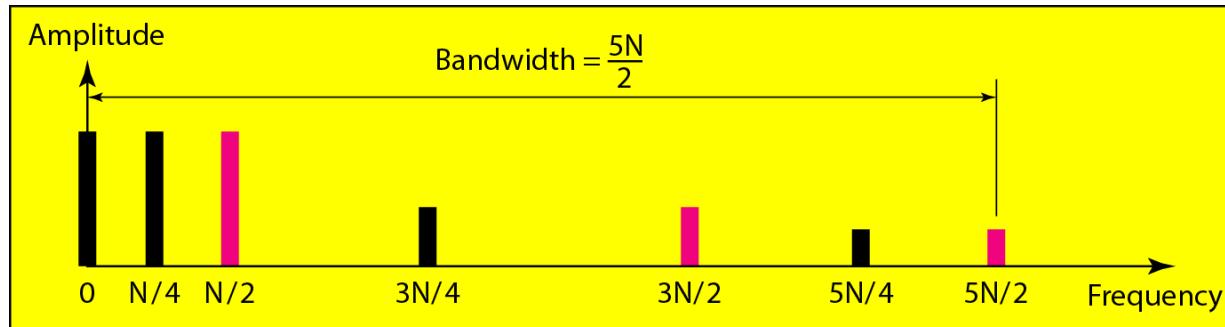
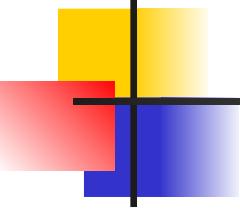


Figure 3.22 Simulating a digital signal with first three harmonics



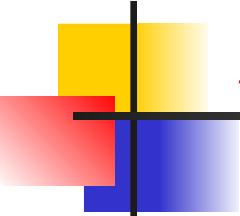


Note

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

Table 3.2 Bandwidth requirements

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



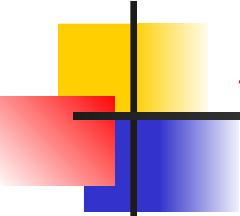
Example 3.22

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 $B = 5 \times 500 \text{ kHz} = 2.5 \text{ MHz}$.*



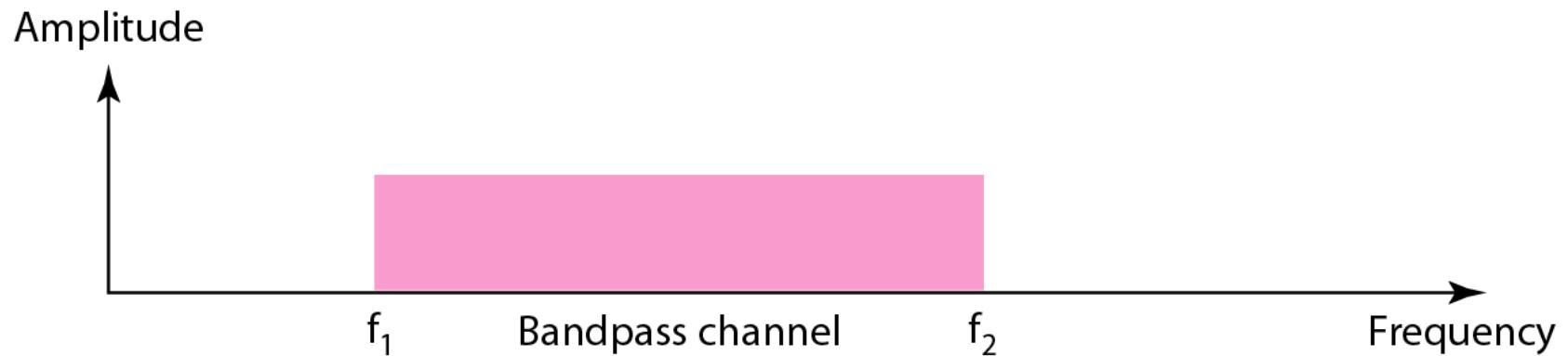
Example 3.22

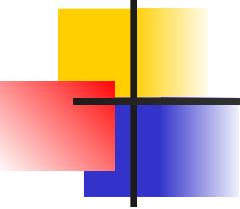
We have a low-pass channel with bandwidth 100 kHz. What is the maximum bit rate of this channel?

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*

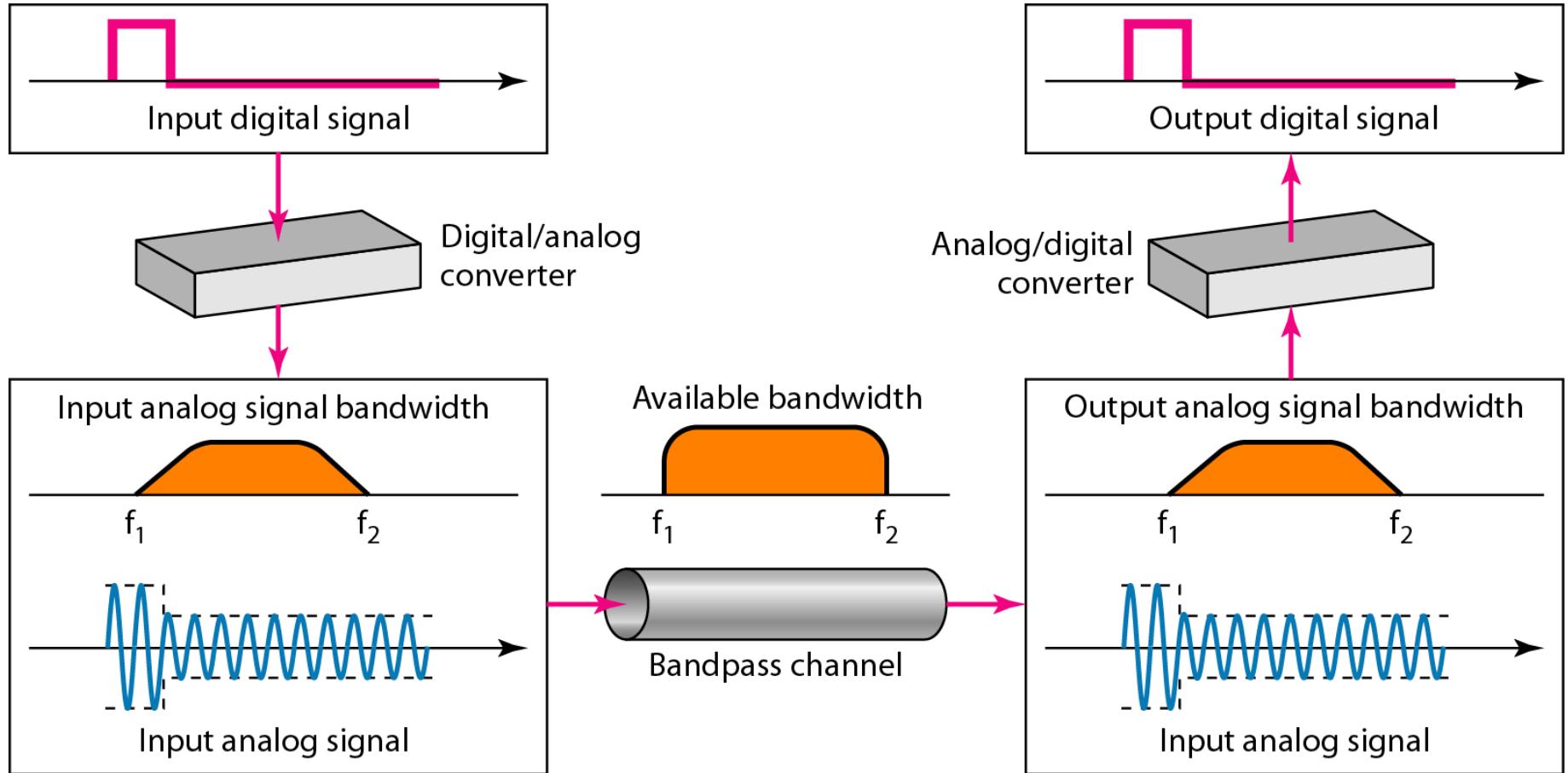


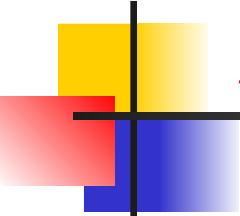


Note

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.

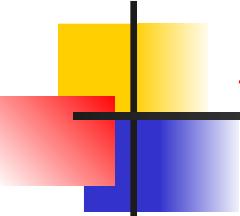
Figure 3.24 Modulation of a digital signal for transmission on a bandpass channel





Example 3.24

An example of broadband transmission using modulation is the sending of computer data through a telephone subscriber line, the line connecting a resident to the central telephone office. These lines are designed to carry voice with a limited bandwidth. The channel is considered a bandpass channel. We convert the digital signal from the computer to an analog signal, and send the analog signal. We can install two converters to change the digital signal to analog and vice versa at the receiving end. The converter, in this case, is called a modem which we discuss in detail in Chapter 5.



Example 3.25

A second example is the digital cellular telephone. For better reception, digital cellular phones convert the analog voice signal to a digital signal (see Chapter 16). Although the bandwidth allocated to a company providing digital cellular phone service is very wide, we still cannot send the digital signal without conversion. The reason is that we only have a bandpass channel available between caller and callee. We need to convert the digitized voice to a composite analog signal before sending.

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 3.25 *Causes of impairment*

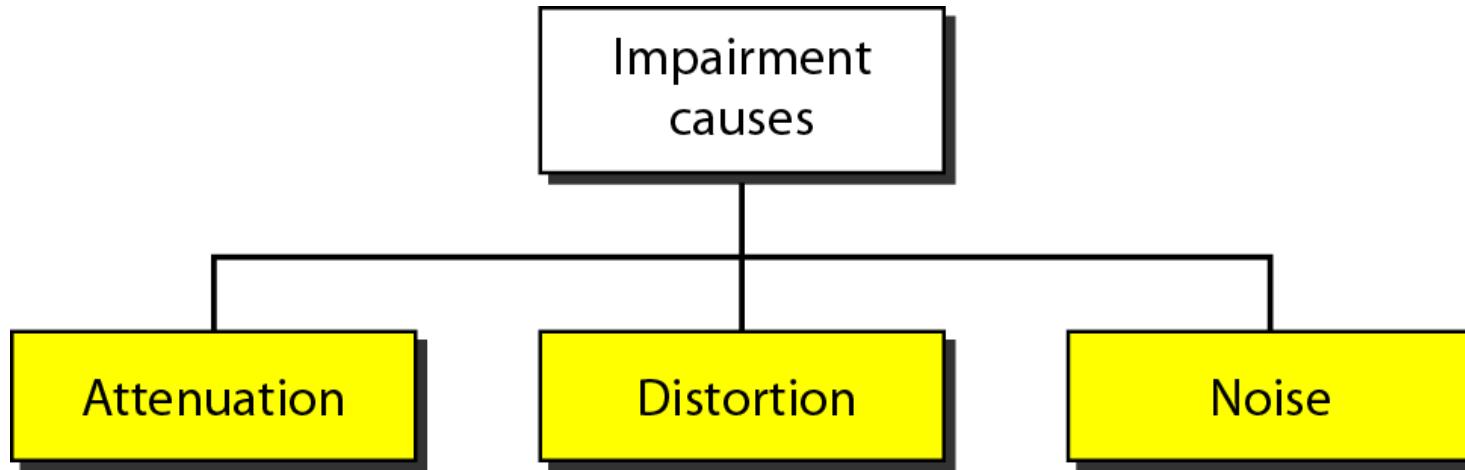
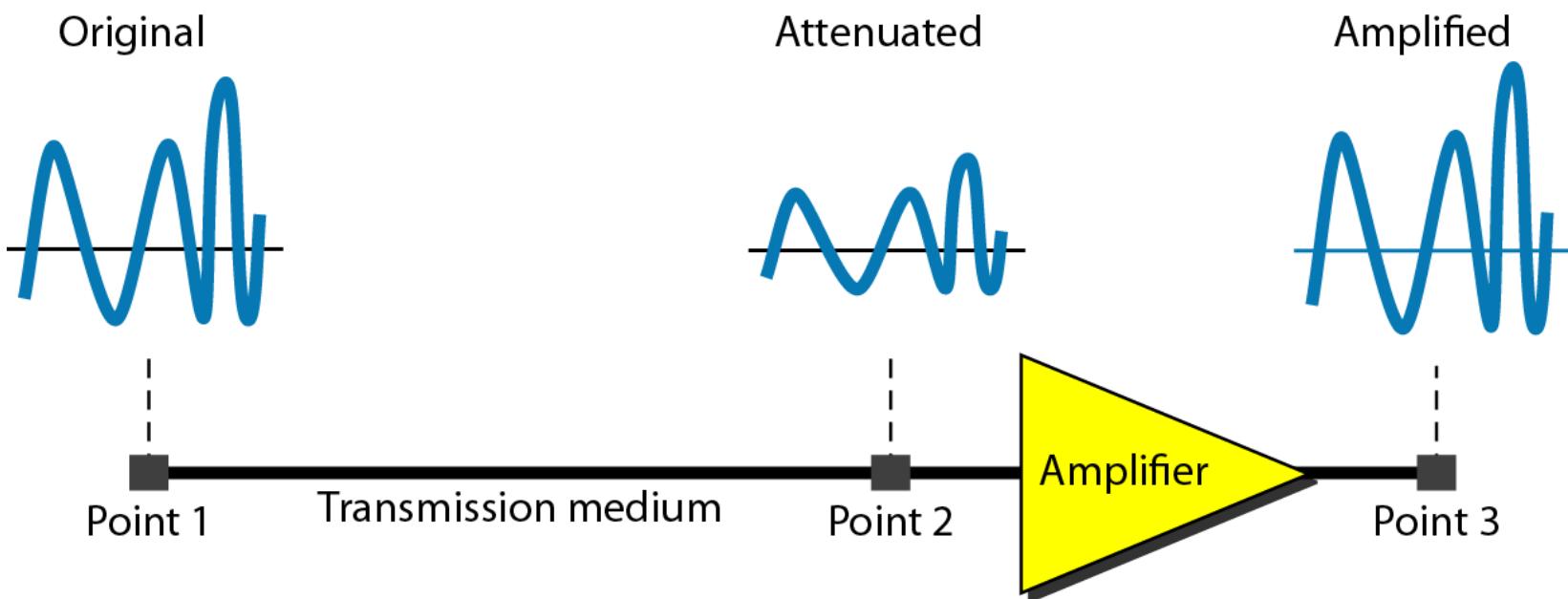
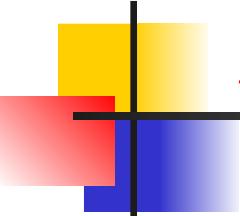


Figure 3.26 Attenuation



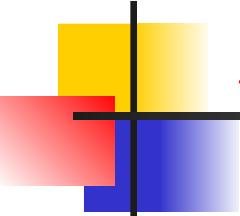


Example 3.26

Suppose a signal travels through a transmission medium and its power is reduced to one-half. This means that P_2 is $(1/2)P_1$. In this case, the attenuation (loss of power) can be calculated as

$$10 \log_{10} \frac{P_2}{P_1} = 10 \log_{10} \frac{0.5 P_1}{P_1} = 10 \log_{10} 0.5 = 10(-0.3) = -3 \text{ dB}$$

A loss of 3 dB (-3 dB) is equivalent to losing one-half the power.

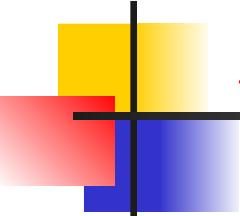


Example 3.27

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

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

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

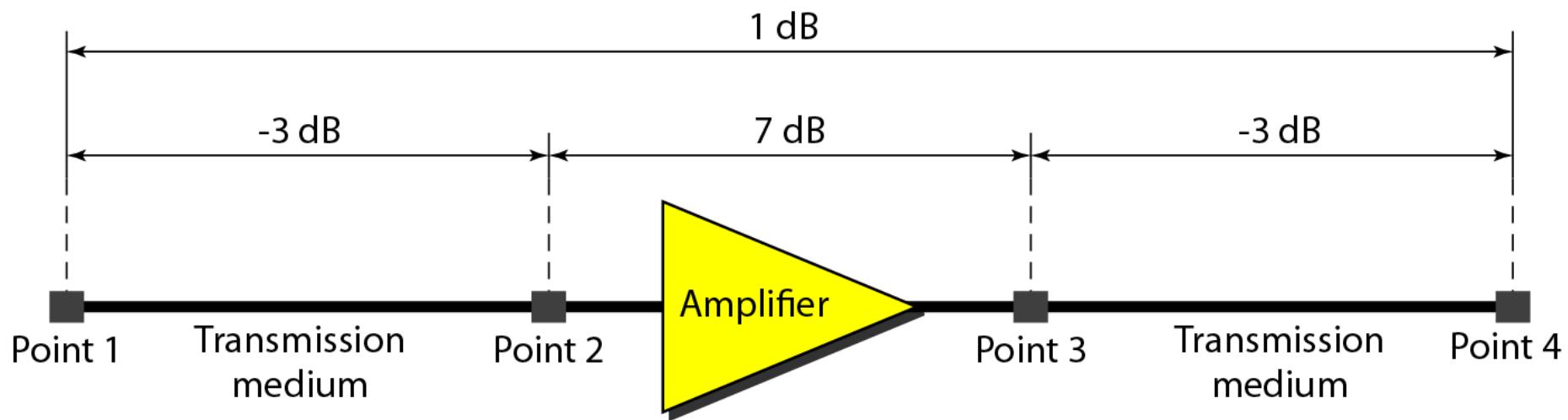


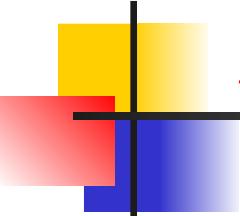
Example 3.28

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





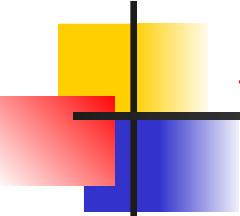
Example 3.29

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

Solution

We can calculate the power in the signal as

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



Example 3.30

The loss in a cable is usually defined in decibels per kilometer (dB/km). If the signal at the beginning of a cable with -0.3 dB/km has a power of 2 mW , what is the power of the signal at 5 km ?

Solution

The loss in the cable in decibels is $5 \times (-0.3) = -1.5 \text{ dB}$.

We can calculate the power as

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

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

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

Figure 3.28 *Distortion*

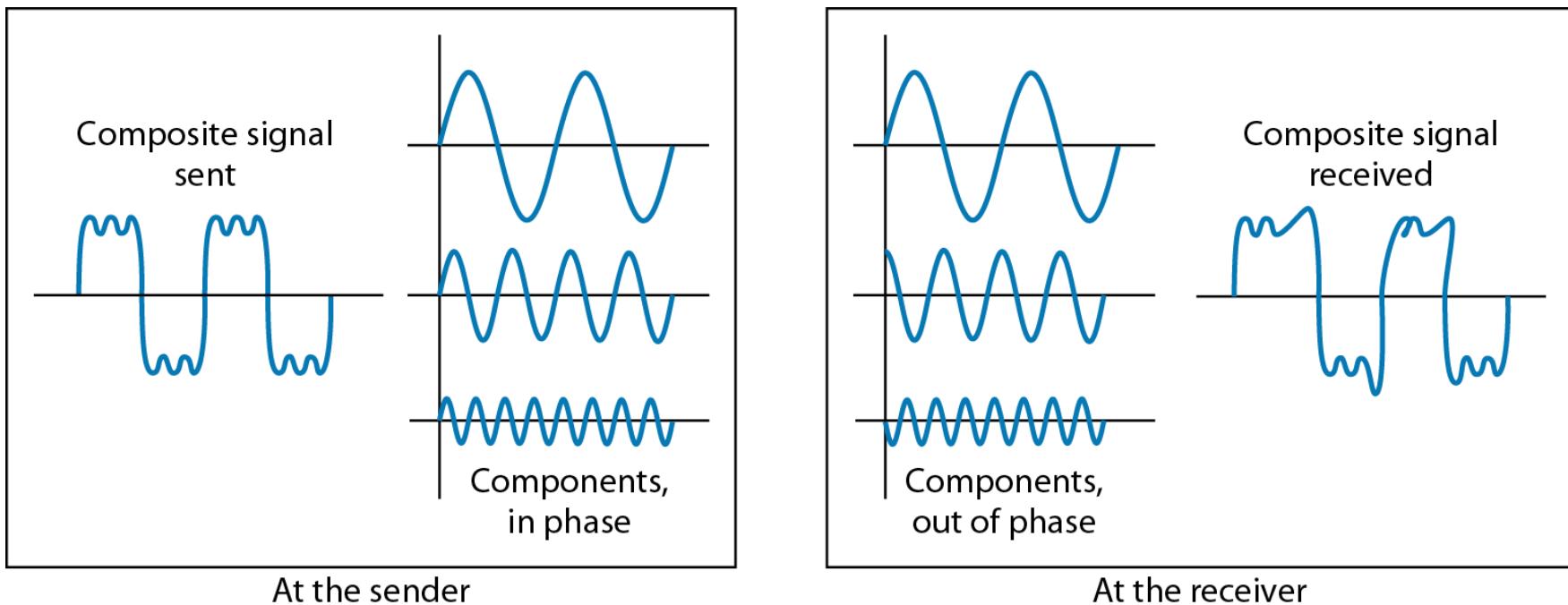
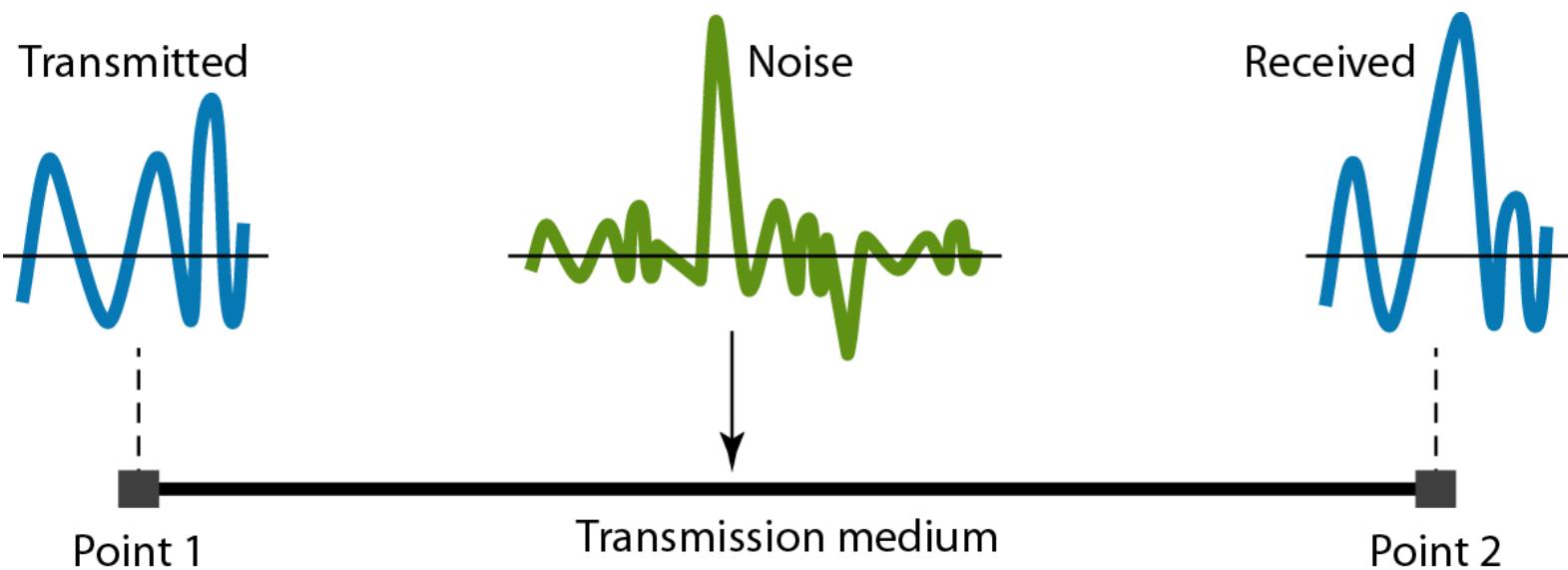
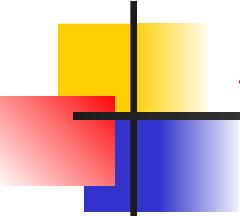


Figure 3.29 Noise





Example 3.31

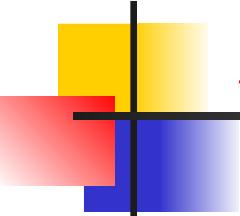
The power of a signal is 10 mW and the power of the noise is 1 μW; what are the values of SNR and SNR_{dB}?

Solution

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

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

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



Example 3.32

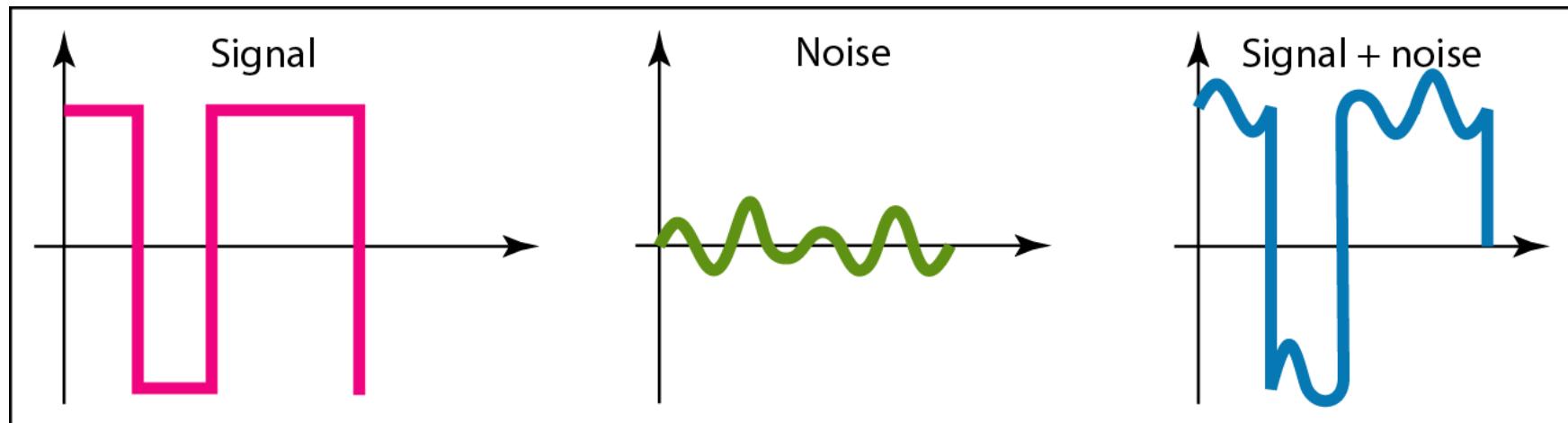
The values of SNR and SNR_{dB} for a noiseless channel are

$$\text{SNR} = \frac{\text{signal power}}{0} = \infty$$

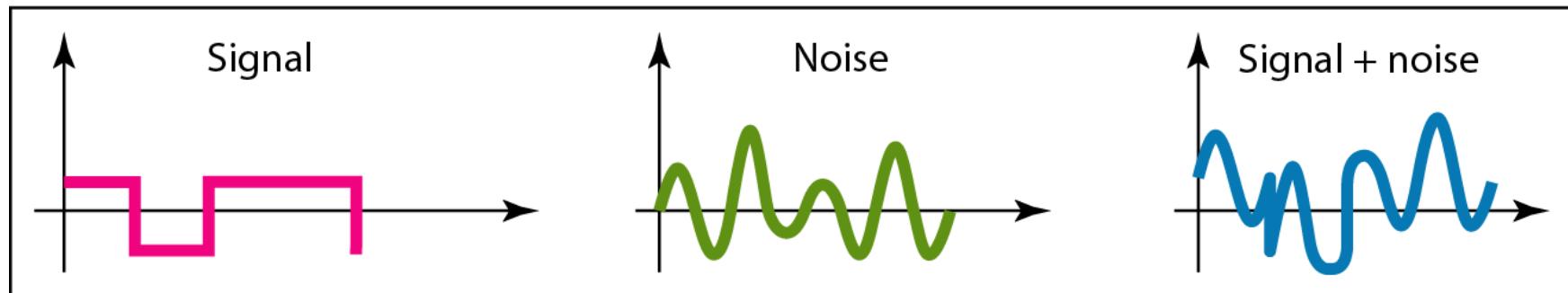
$$\text{SNR}_{\text{dB}} = 10 \log_{10} \infty = \infty$$

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

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:

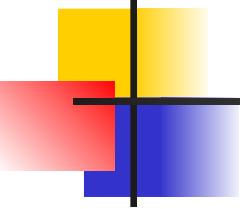
- 1. The bandwidth available**
- 2. The level of the signals we use**
- 3. The quality of the channel (the level of noise)**

Topics discussed in this section:

Noiseless Channel: Nyquist Bit Rate

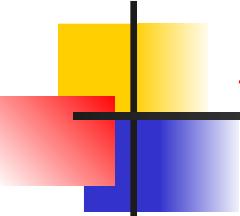
Noisy Channel: Shannon Capacity

Using Both Limits



Note

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

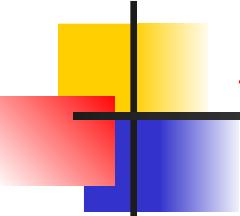


Example 3.33

Does the Nyquist theorem bit rate agree with the intuitive bit rate described in baseband transmission?

Solution

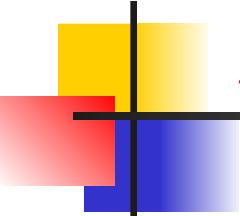
They match when we have only two levels. We said, in baseband transmission, the bit rate is 2 times the bandwidth if we use only the first harmonic in the worst case. However, the Nyquist formula is more general than what we derived intuitively; it can be applied to baseband transmission and modulation. Also, it can be applied when we have two or more levels of signals.



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

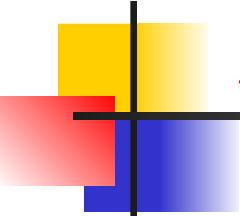
$$\text{BitRate} = 2 \times 3000 \times \log_2 2 = 6000 \text{ bps}$$



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

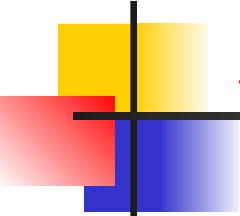
We need to send 265 kbps over a noiseless channel with a bandwidth of 20 kHz. How many signal levels do we need?

Solution

We can use the Nyquist formula as shown:

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

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.

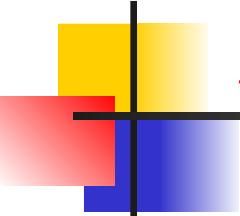


Example 3.37

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.

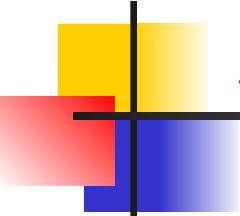


Example 3.38

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

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

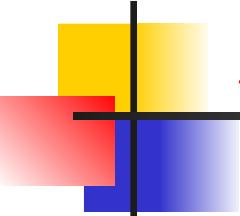
This means that the highest bit rate for a telephone line is 34.860 kbps. If we want to send data faster than this, we can either increase the bandwidth of the line or improve the signal-to-noise ratio.



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

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



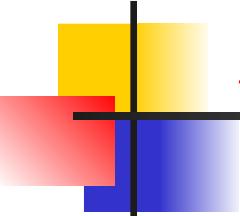
Example 3.40

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

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

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

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



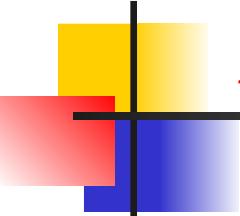
Example 3.41

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

Solution

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

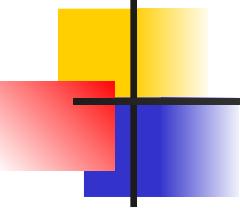
$$C = B \log_2 (1 + \text{SNR}) = 10^6 \log_2 (1 + 63) = 10^6 \log_2 64 = 6 \text{ Mbps}$$



Example 3.41 (continued)

The Shannon formula gives us 6 Mbps, the upper limit. For better performance we choose something lower, 4 Mbps, for example. Then we use the Nyquist formula to find the number of signal levels.

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



Note

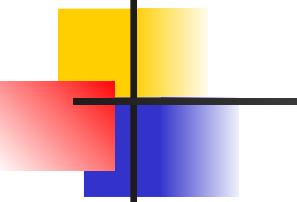
The Shannon capacity gives us the upper limit; the Nyquist formula tells us how many signal levels we need.

3-6 PERFORMANCE

*One important issue in networking is the **performance** of the network—how good is it? We discuss quality of service, an overall measurement of network performance, in greater detail in Chapter 24. In this section, we introduce terms that we need for future chapters.*

Topics discussed in this section:

- Bandwidth**
- Throughput**
- Latency (Delay)**
- Bandwidth-Delay Product**

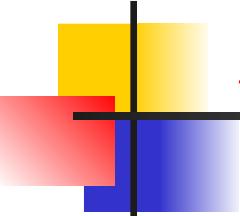


Note

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

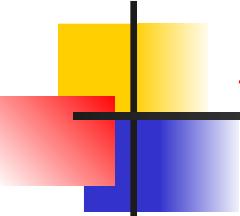
- The first, bandwidth in hertz, refers to the range of frequencies in a composite signal or the range of frequencies that a channel can pass.

- The second, bandwidth in bits per second, refers to the speed of bit transmission in a channel or link.



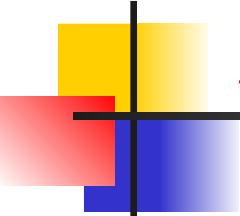
Example 3.42

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



Example 3.43

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



Example 3.44

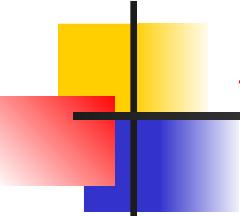
A network with bandwidth of 10 Mbps can pass only an average of 12,000 frames per minute with each frame carrying an average of 10,000 bits. What is the throughput of this network?

Solution

We can calculate the throughput as

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

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



Example 3.45

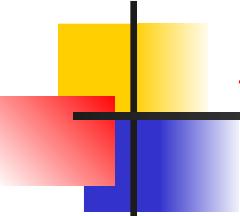
What is the propagation time if the distance between the two points is 12,000 km? Assume the propagation speed to be 2.4×10^8 m/s in cable.

Solution

We can calculate the propagation time as

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

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

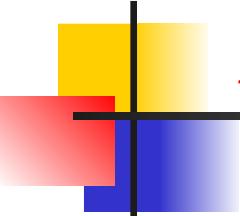


Example 3.46

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

Solution

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

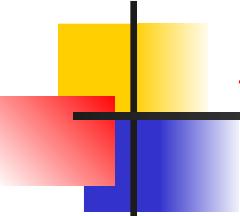


Example 3.46 (continued)

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

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

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

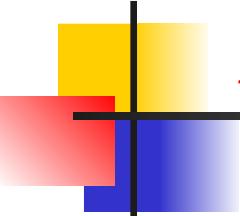


Example 3.47

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

Solution

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



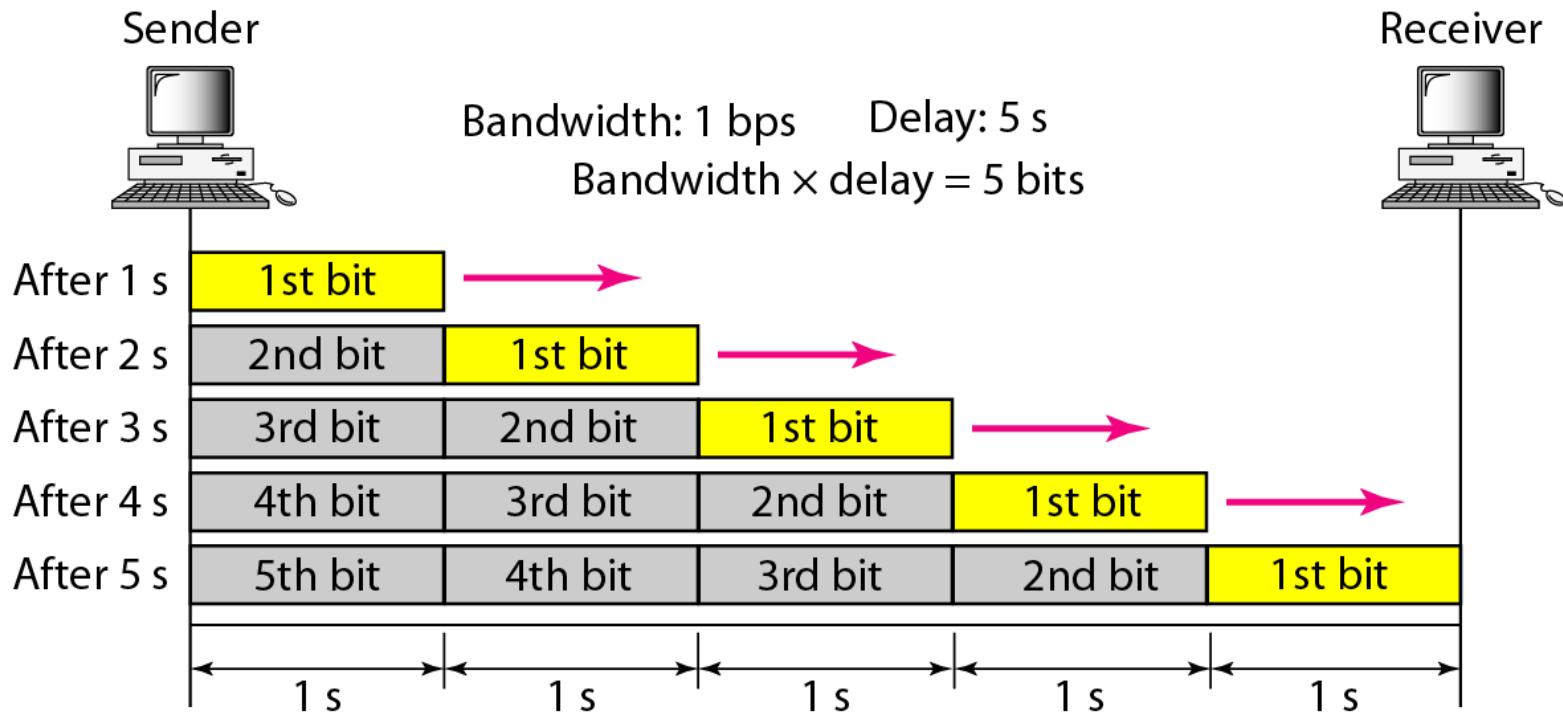
Example 3.47 (continued)

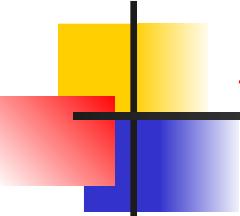
$$\text{Propagation time} = \frac{12,000 \times 1000}{2.4 \times 10^8} = 50 \text{ ms}$$

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

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

Figure 3.31 *Filling the link with bits for case 1*

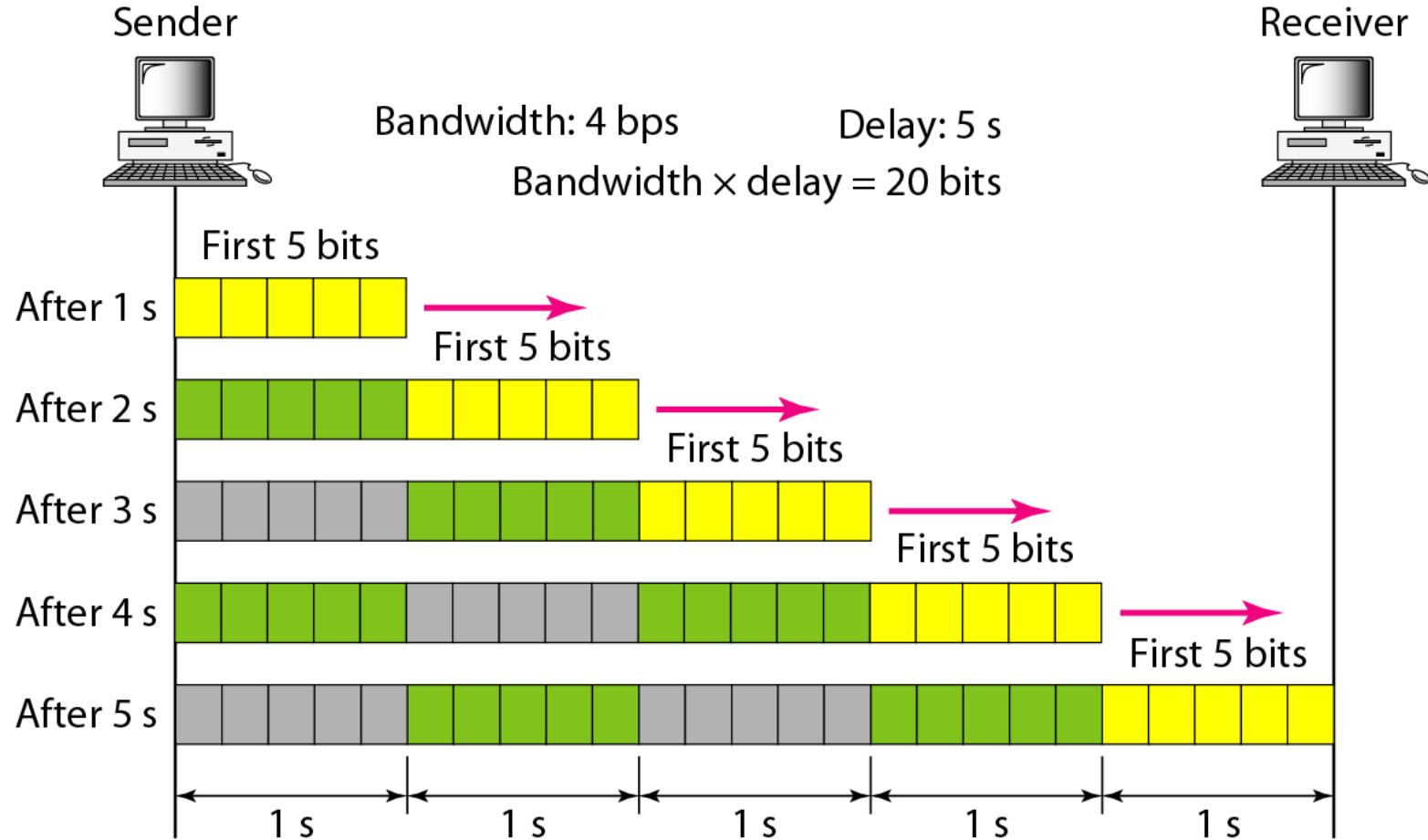


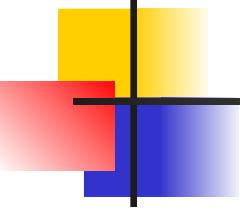


Example 3.48

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

Figure 3.32 Filling the link with bits in case 2

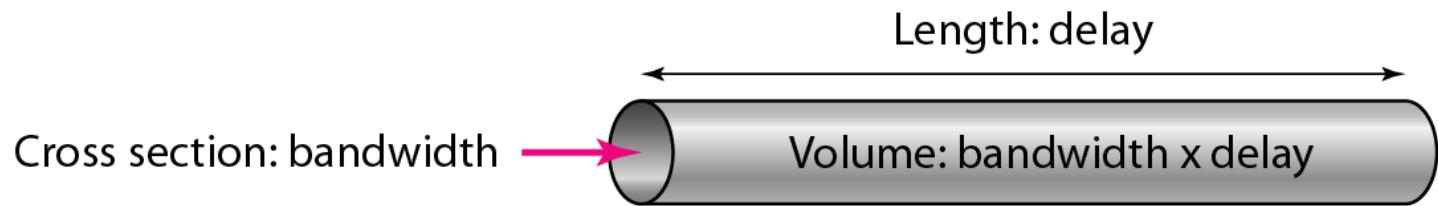




Note

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

Figure 3.33 *Concept of bandwidth-delay product*



Chapter 4

Digital Transmission

4-1 DIGITAL-TO-DIGITAL CONVERSION

*In this section, we see how we can represent digital data by using digital signals. The conversion involves three techniques: **line coding**, **block coding**, and **scrambling**. Line coding is always needed; block coding and scrambling may or may not be needed.*

Topics discussed in this section:

Line Coding

Line Coding Schemes

Block Coding

Scrambling

Figure 4.1 *Line coding and decoding*

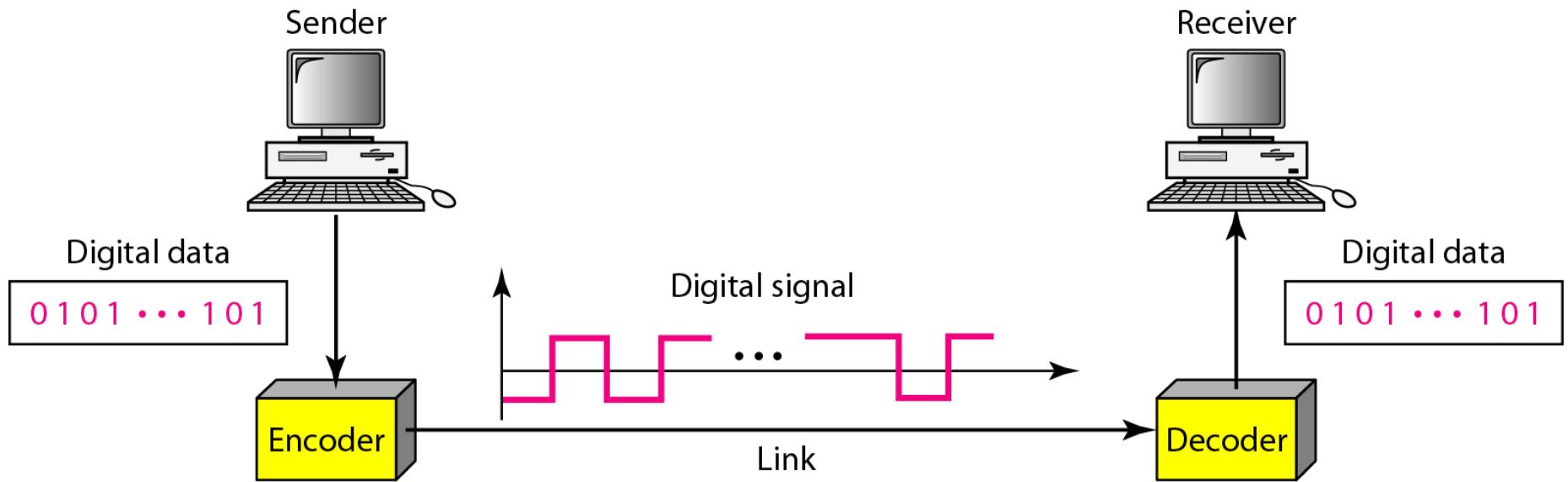
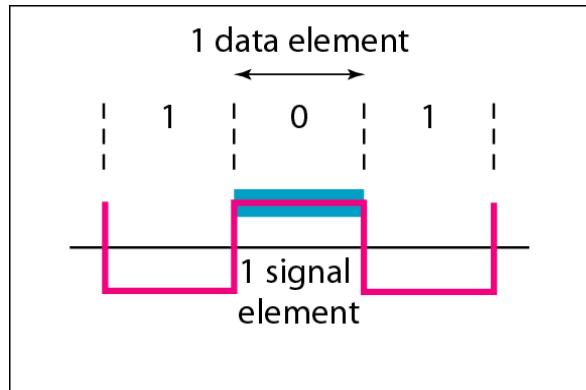
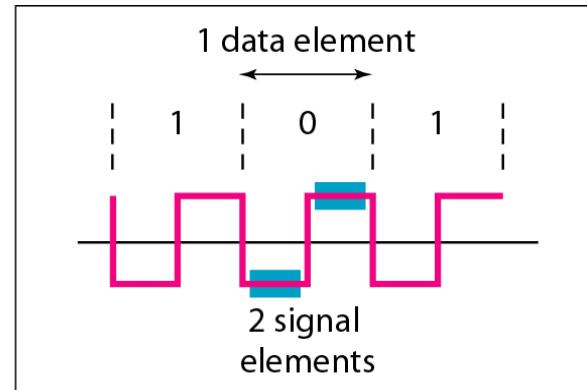


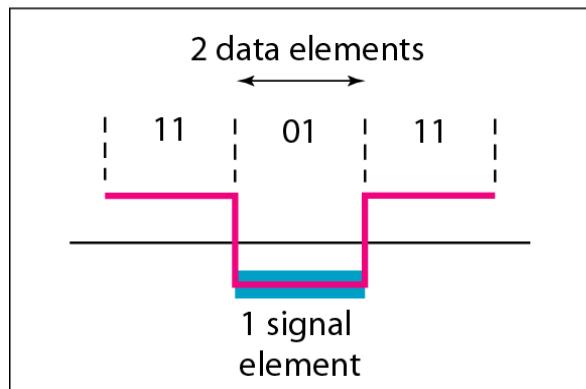
Figure 4.2 Signal element versus data element



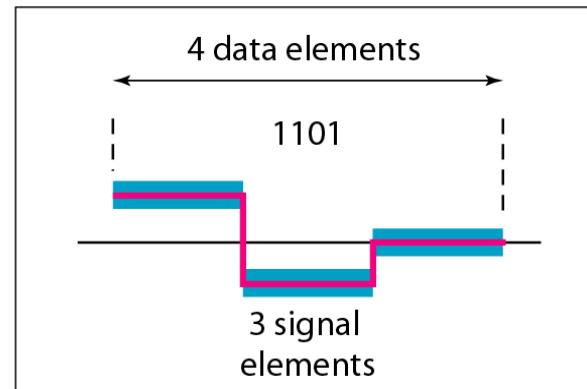
a. One data element per one signal element ($r = 1$)



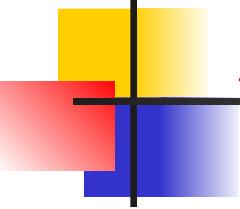
b. One data element per two signal elements ($r = \frac{1}{2}$)



c. Two data elements per one signal element ($r = 2$)



d. Four data elements per three signal elements ($r = \frac{4}{3}$)



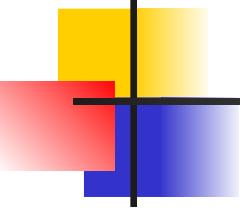
Example 4.1

A signal is carrying data in which one data element is encoded as one signal element ($r = 1$). If the bit rate is 100 kbps, what is the average value of the baud rate if c is between 0 and 1?

Solution

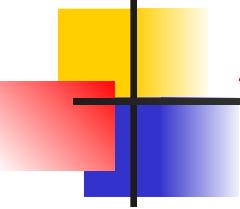
We assume that the average value of c is $1/2$. The baud rate is then

$$S = c \times N \times \frac{1}{r} = \frac{1}{2} \times 100,000 \times \frac{1}{1} = 50,000 = 50 \text{ kbaud}$$



Note

Although the actual bandwidth of a digital signal is infinite, the effective bandwidth is finite.



Example 4.2

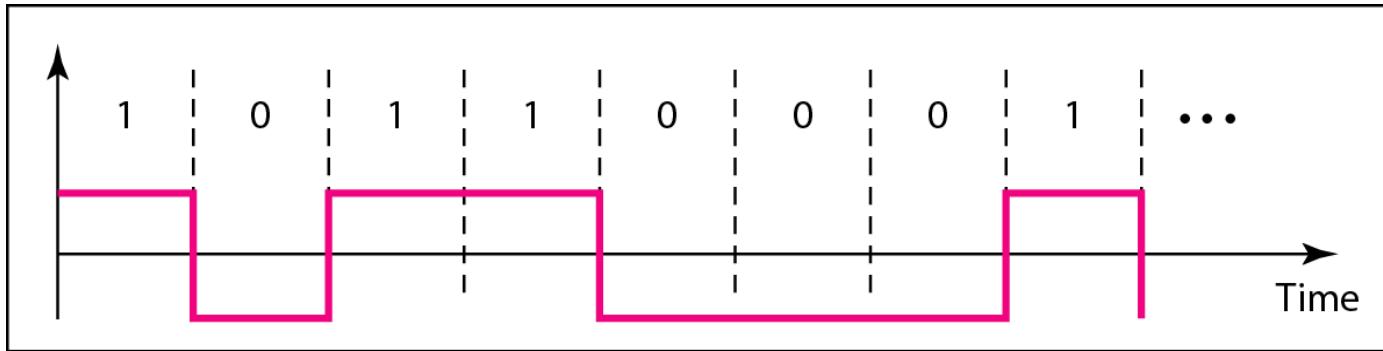
The maximum data rate of a channel (see Chapter 3) is $N_{max} = 2 \times B \times \log_2 L$ (defined by the Nyquist formula). Does this agree with the previous formula for N_{max} ?

Solution

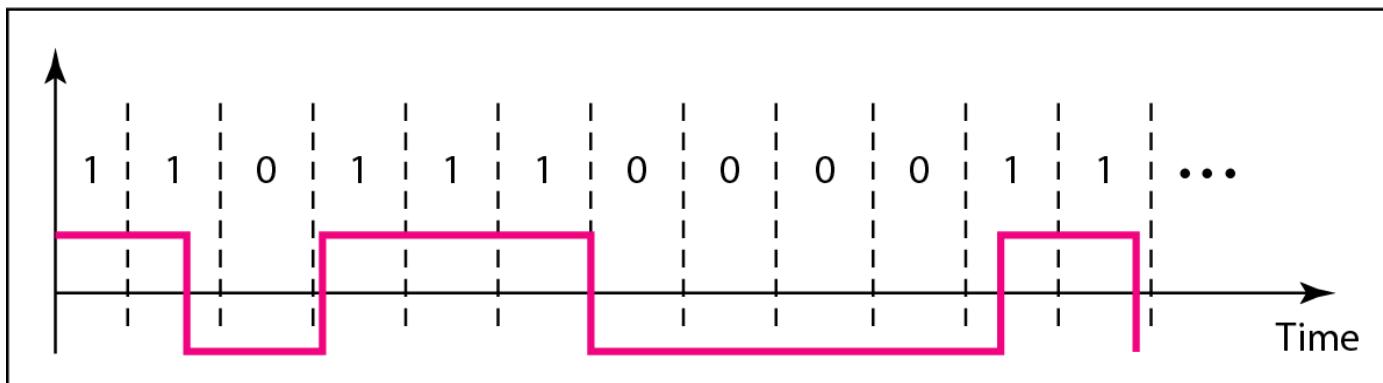
A signal with L levels actually can carry $\log_2 L$ bits per level. If each level corresponds to one signal element and we assume the average case ($c = 1/2$), then we have

$$N_{max} = \frac{1}{c} \times B \times r = 2 \times B \times \log_2 L$$

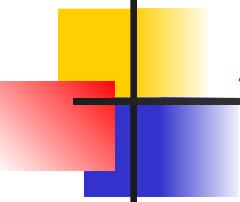
Figure 4.3 *Effect of lack of synchronization*



a. Sent



b. Received



Example 4.3

In a digital transmission, the receiver clock is 0.1 percent faster than the sender clock. How many extra bits per second does the receiver receive if the data rate is 1 kbps? How many if the data rate is 1 Mbps?

Solution

At 1 kbps, the receiver receives 1001 bps instead of 1000 bps.

1000 bits sent

1001 bits received

1 extra bps

At 1 Mbps, the receiver receives 1,001,000 bps instead of 1,000,000 bps.

1,000,000 bits sent

1,001,000 bits received

1000 extra bps

Figure 4.4 *Line coding schemes*

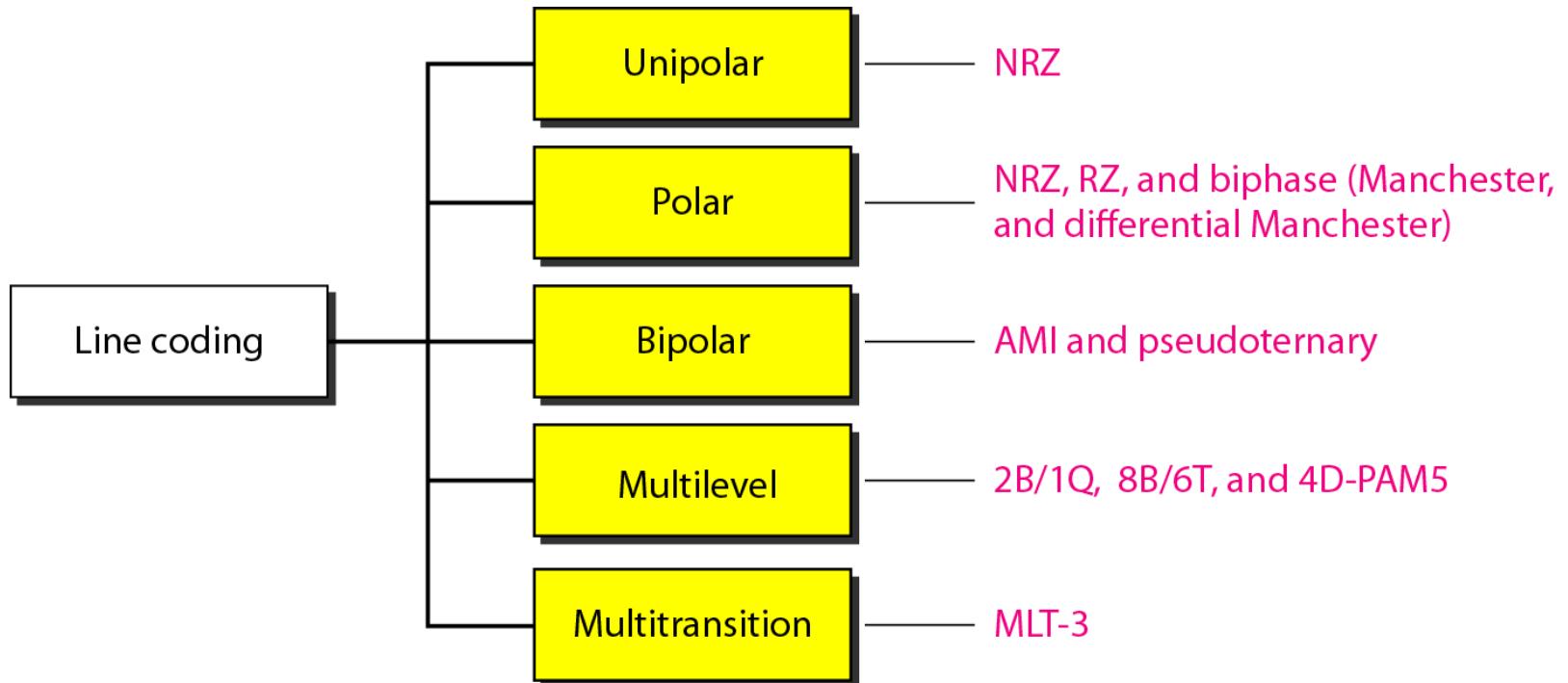
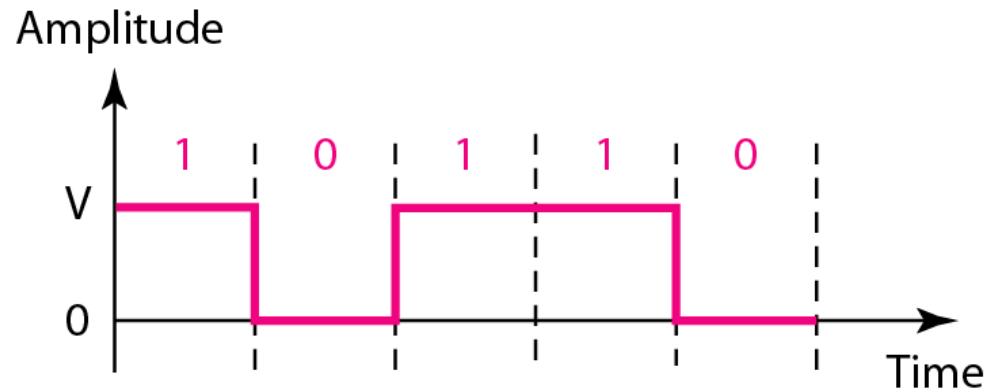


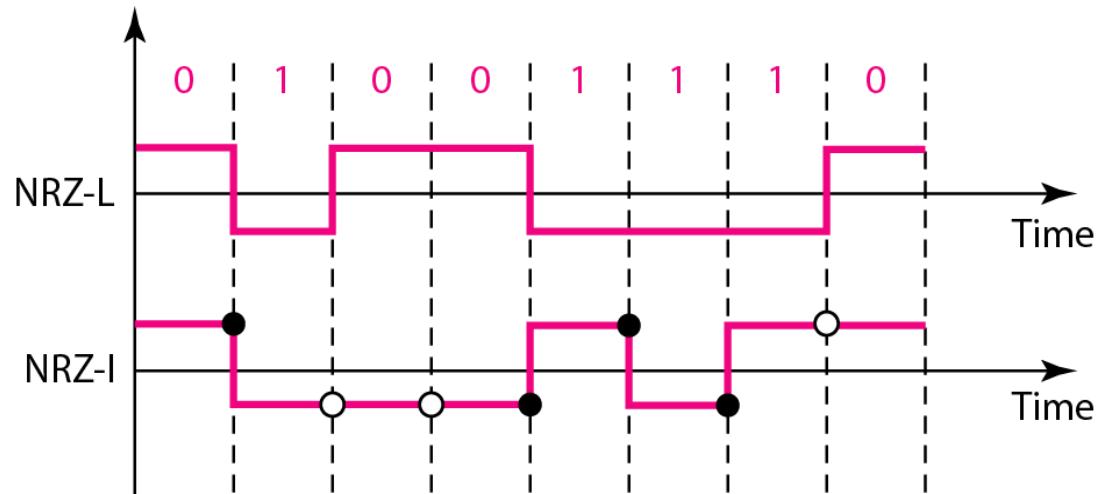
Figure 4.5 *Unipolar NRZ scheme*



$$\frac{1}{2}V^2 + \frac{1}{2}(0)^2 = \frac{1}{2}V^2$$

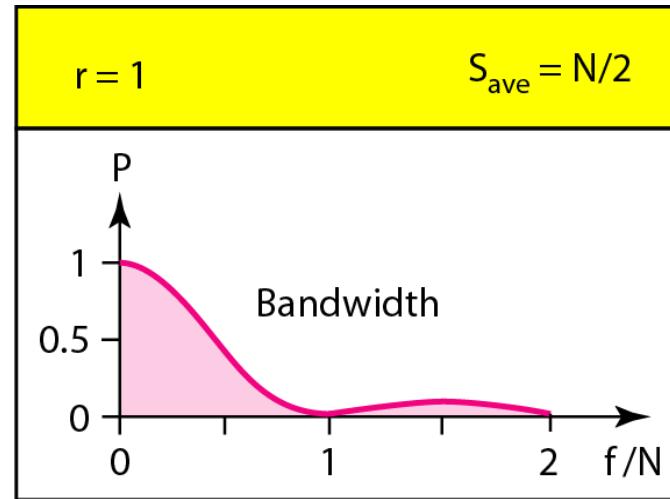
Normalized power

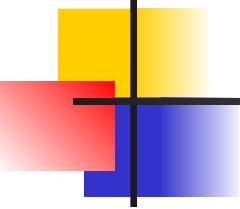
Figure 4.6 Polar NRZ-L and NRZ-I schemes



○ No inversion: Next bit is 0

● Inversion: Next bit is 1

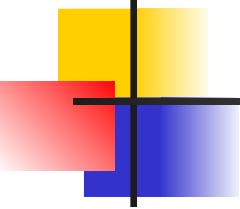




Note

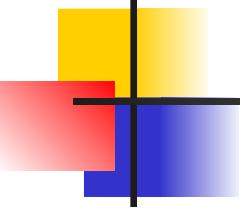
In NRZ-L the level of the voltage determines the value of the bit.

In NRZ-I the inversion or the lack of inversion determines the value of the bit.



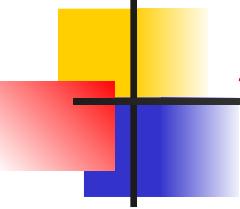
Note

NRZ-L and NRZ-I both have an average signal rate of $N/2$ Bd.



Note

NRZ-L and NRZ-I both have a DC component problem.



Example 4.4

A system is using NRZ-I to transfer 10-Mbps data. What are the average signal rate and minimum bandwidth?

Solution

The average signal rate is $S = N/2 = 500$ kbaud. The minimum bandwidth for this average baud rate is $B_{min} = S = 500$ kHz.

Figure 4.7 Polar RZ scheme

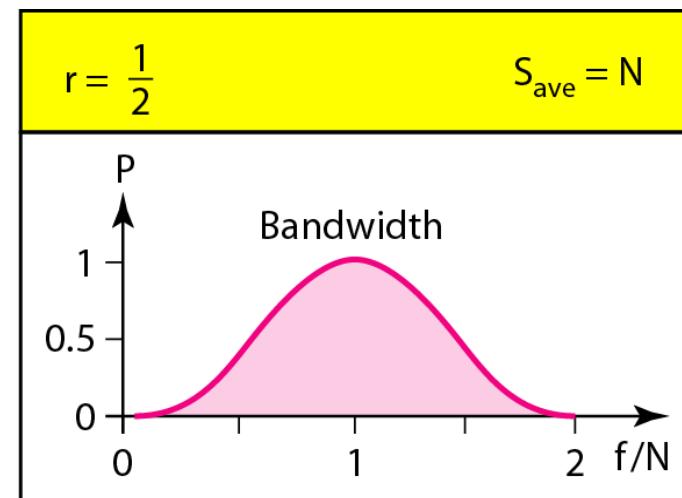
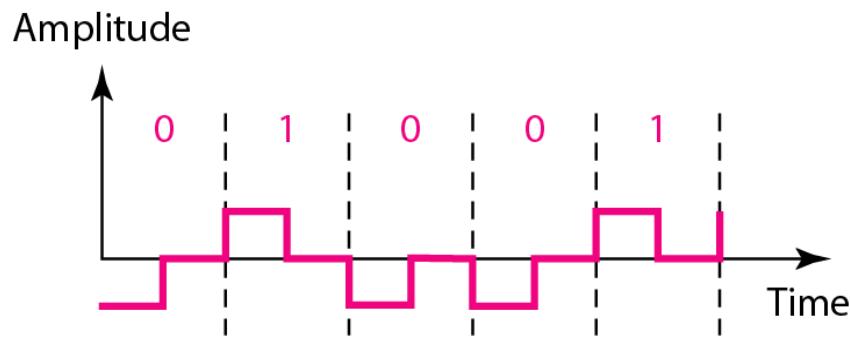
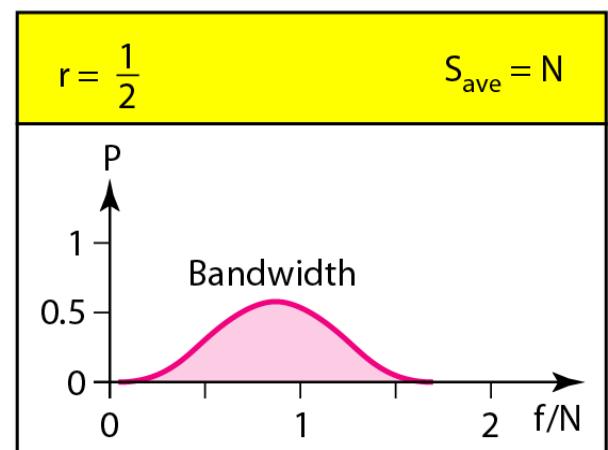
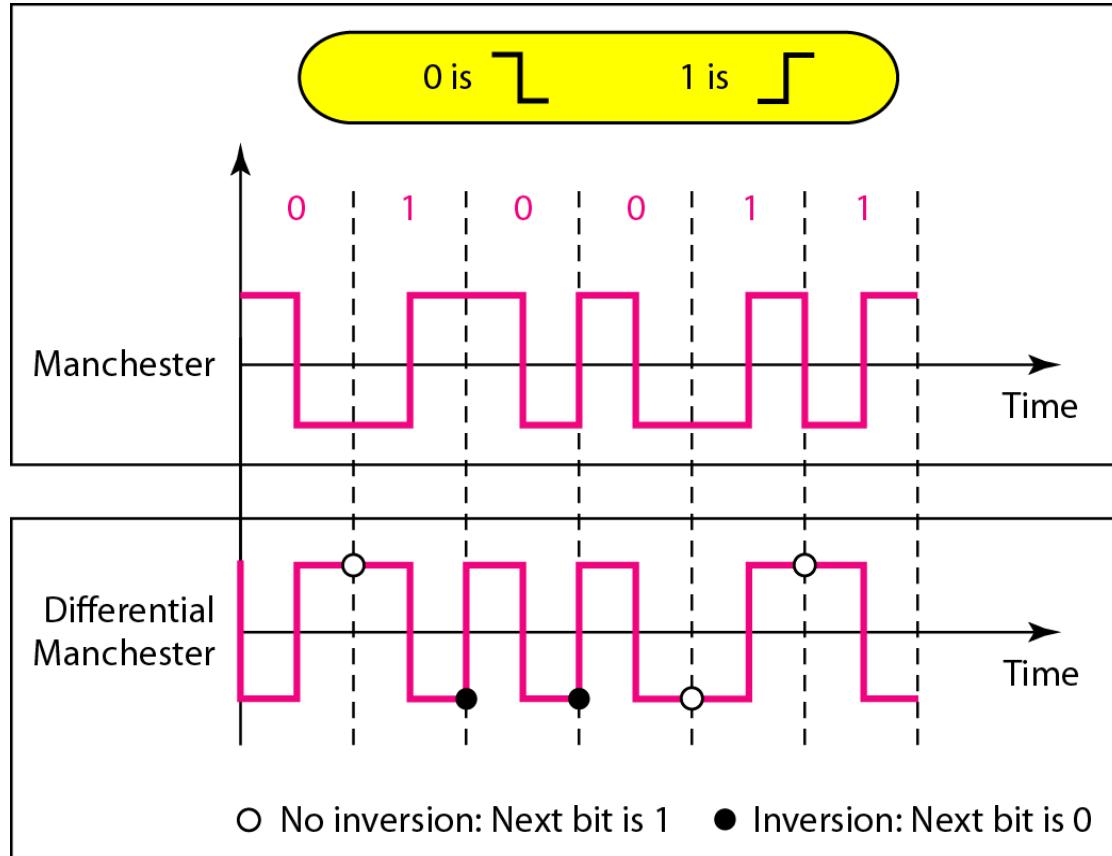
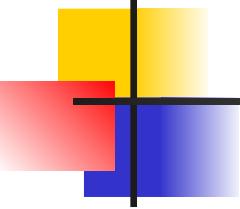


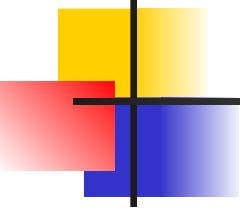
Figure 4.8 Polar biphasic: Manchester and differential Manchester schemes





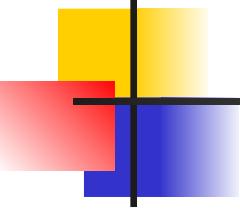
Note

In Manchester and differential Manchester encoding, the transition at the middle of the bit is used for synchronization.



Note

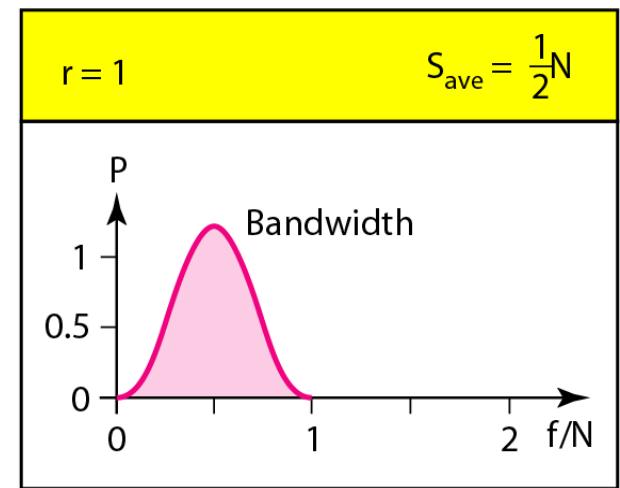
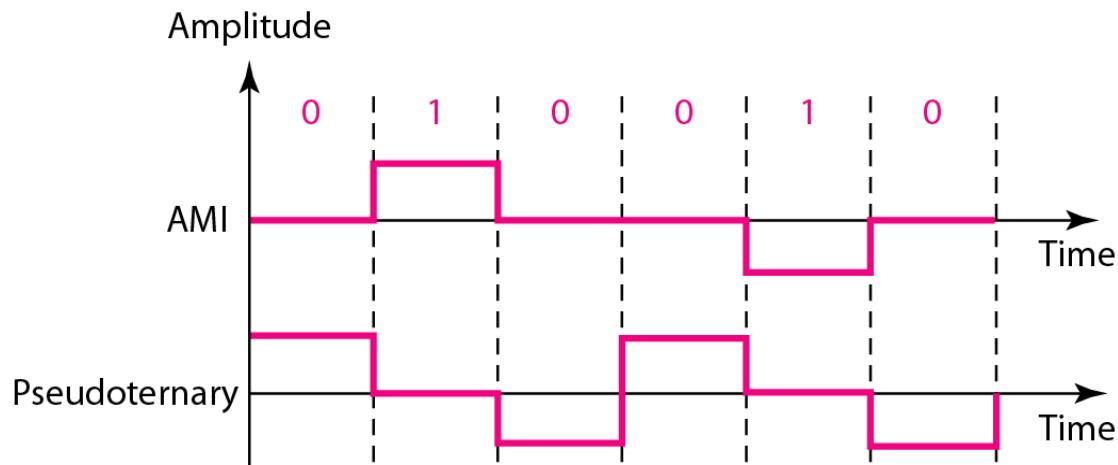
The minimum bandwidth of Manchester and differential Manchester is 2 times that of NRZ.

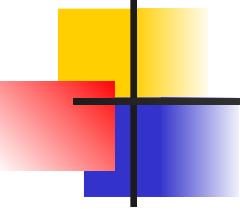


Note

**In bipolar encoding, we use three levels:
positive, zero, and negative.**

Figure 4.9 Bipolar schemes: AMI and pseudoternary





Note

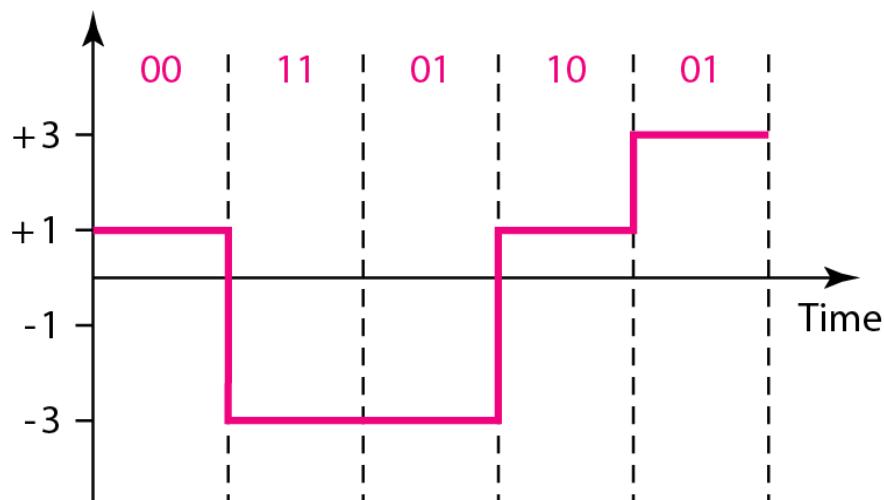
In $mBnL$ schemes, a pattern of m data elements is encoded as a pattern of n signal elements in which $2^m \leq L^n$.

Figure 4.10 Multilevel: 2B1Q scheme

Previous level:
positive Previous level:
negative

Next bits	Next level	Next level
00	+1	-1
01	+3	-3
10	-1	+1
11	-3	+3

Transition table



Assuming positive original level

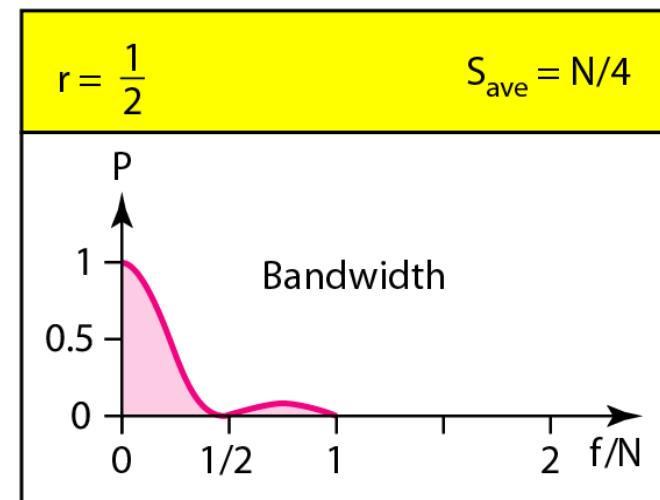


Figure 4.11 Multilevel: 8B6T scheme

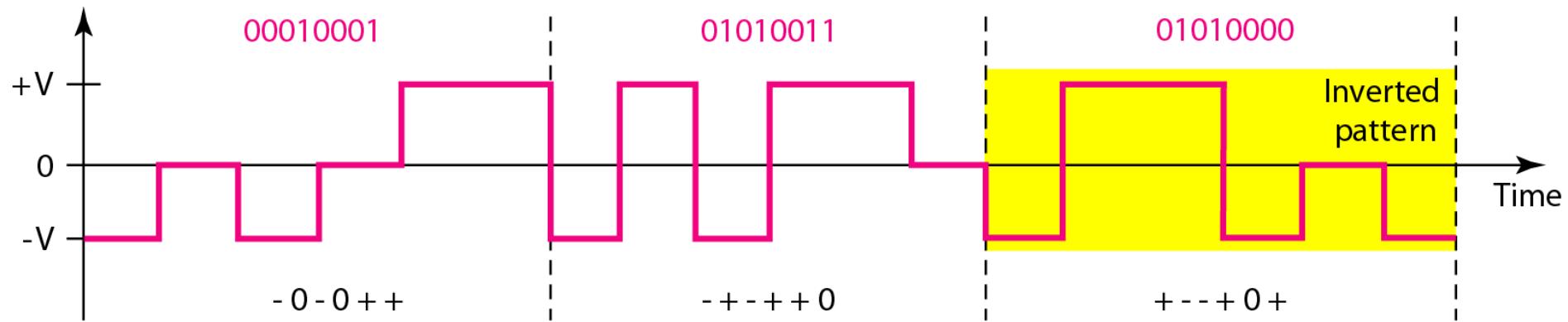


Figure 4.12 Multilevel: 4D-PAM5 scheme

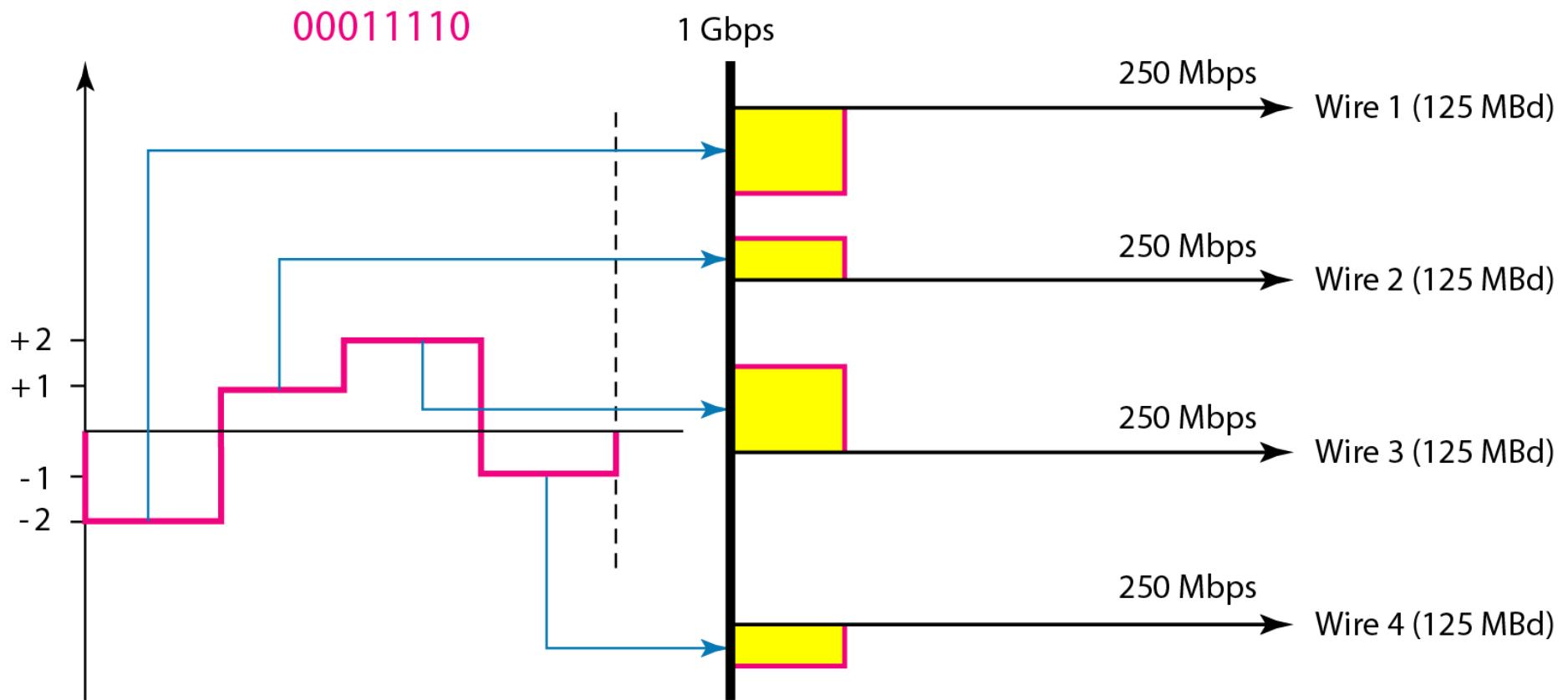
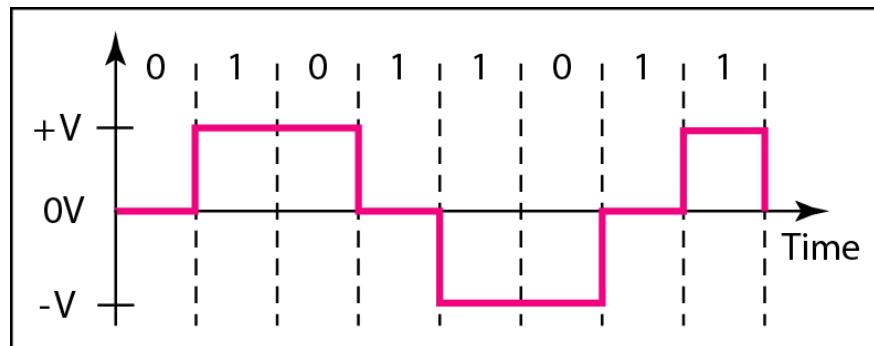
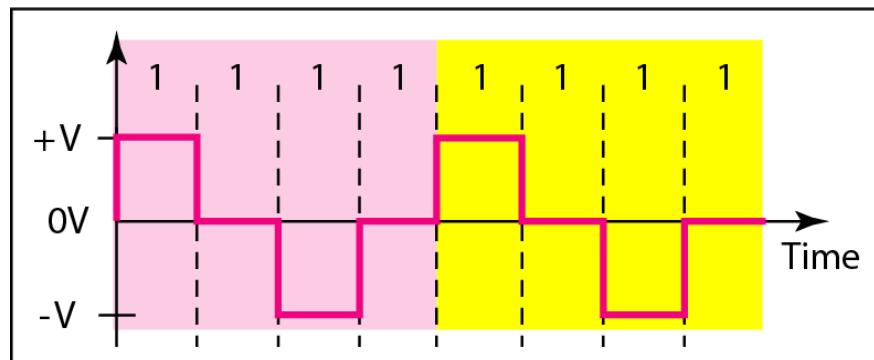


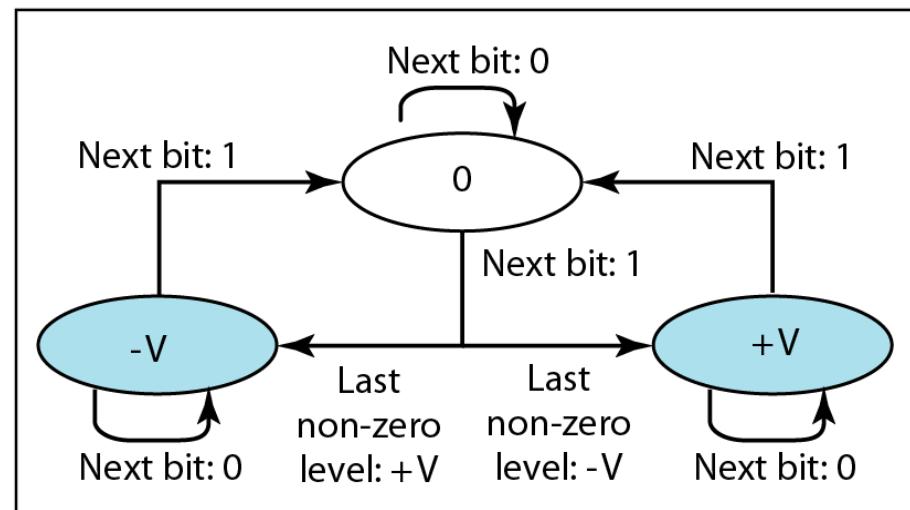
Figure 4.13 Multitransition: MLT-3 scheme



a. Typical case



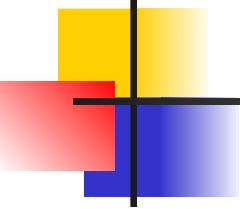
b. Worse case



c. Transition states

Table 4.1 Summary of line coding schemes

Category	Scheme	Bandwidth (average)	Characteristics
Unipolar	NRZ	$B = N/2$	Costly, no self-synchronization if long 0s or 1s, DC
Unipolar	NRZ-L	$B = N/2$	No self-synchronization if long 0s or 1s, DC
	NRZ-I	$B = N/2$	No self-synchronization for long 0s, DC
	Biphase	$B = N$	Self-synchronization, no DC, high bandwidth
Bipolar	AMI	$B = N/2$	No self-synchronization for long 0s, DC
Multilevel	2B1Q	$B = N/4$	No self-synchronization for long same double bits
	8B6T	$B = 3N/4$	Self-synchronization, no DC
	4D-PAM5	$B = N/8$	Self-synchronization, no DC
Multiline	MLT-3	$B = N/3$	No self-synchronization for long 0s



Note

**Block coding is normally referred to as
 mB/nB coding;
it replaces each m -bit group with an
 n -bit group.**

Figure 4.14 *Block coding concept*

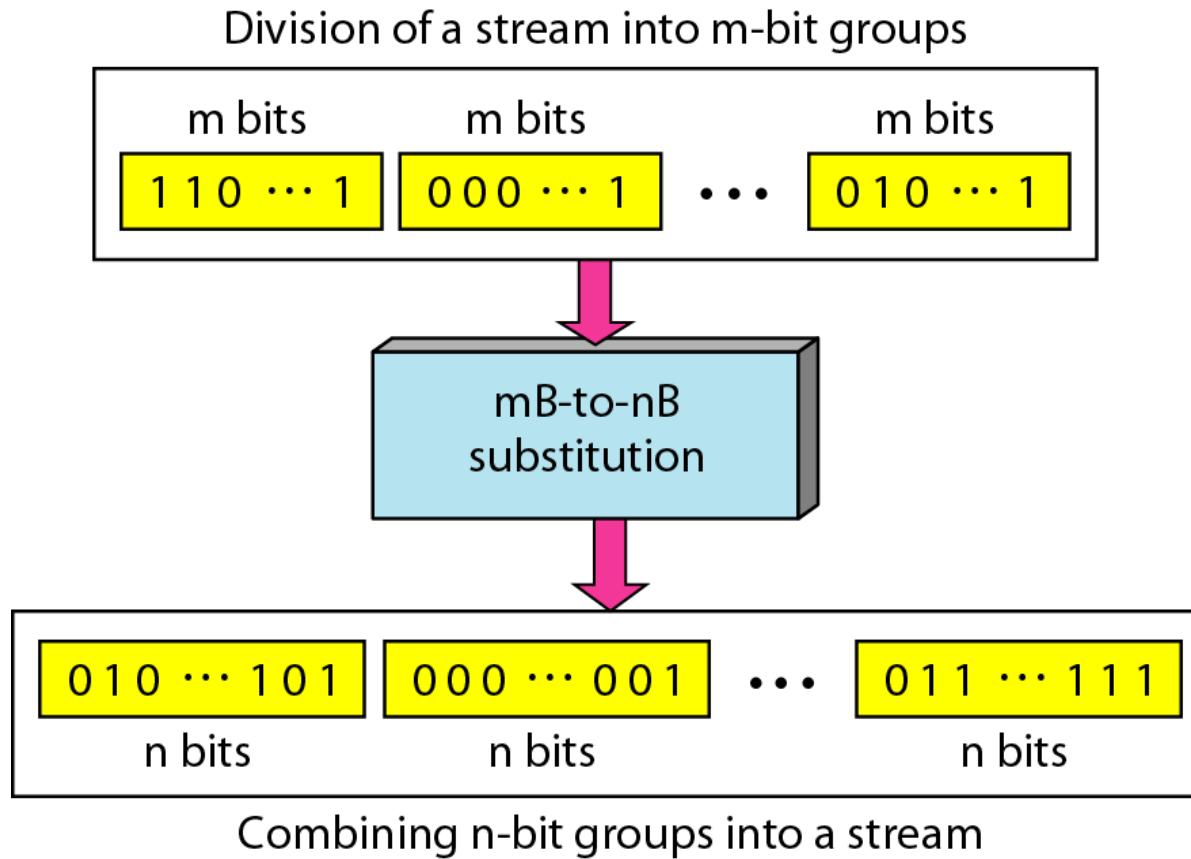


Figure 4.15 *Using block coding 4B/5B with NRZ-I line coding scheme*

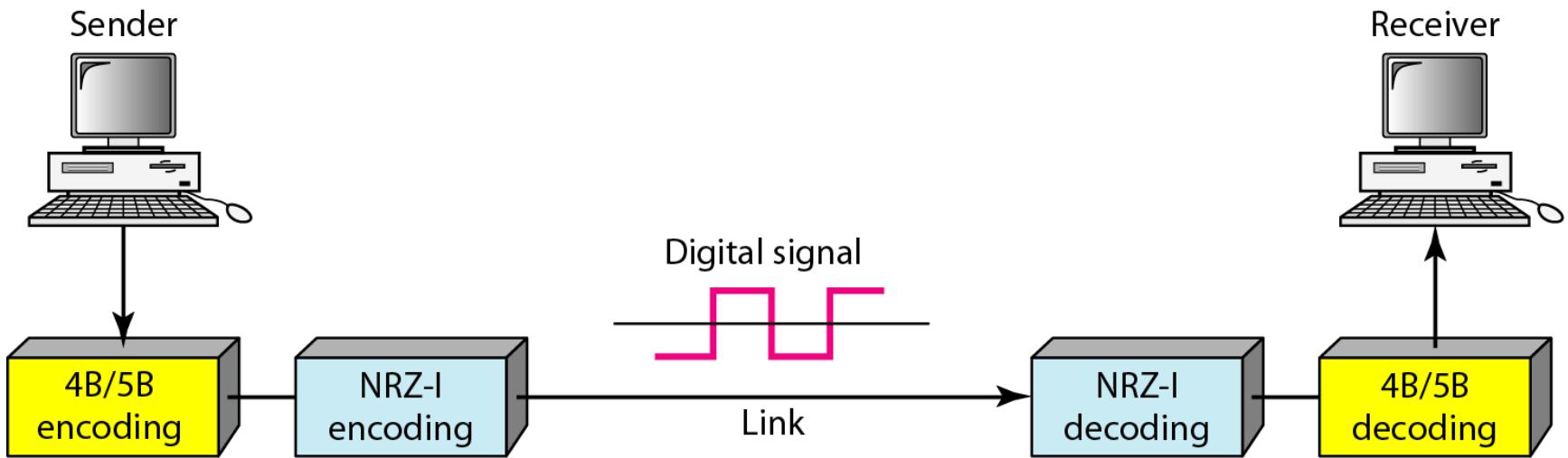
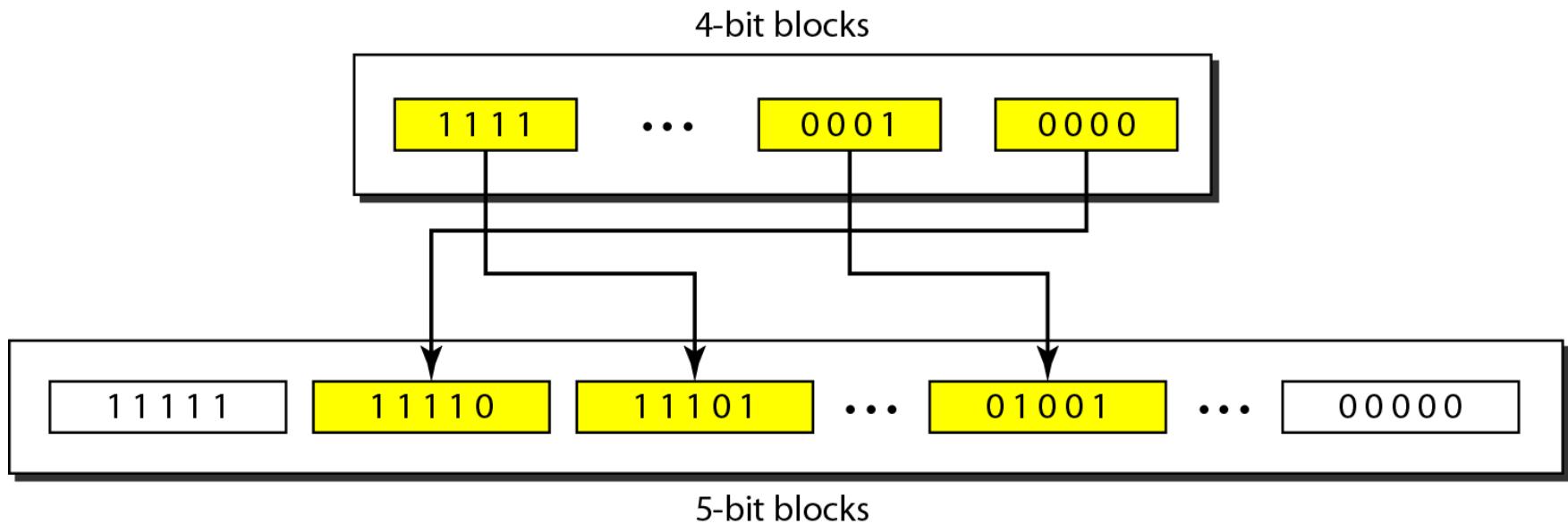
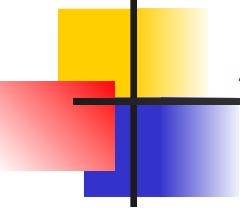


Table 4.2 4B/5B mapping codes

<i>Data Sequence</i>	<i>Encoded Sequence</i>	<i>Control Sequence</i>	<i>Encoded Sequence</i>
0000	11110	Q (Quiet)	00000
0001	01001	I (Idle)	11111
0010	10100	H (Halt)	00100
0011	10101	J (Start delimiter)	11000
0100	01010	K (Start delimiter)	10001
0101	01011	T (End delimiter)	01101
0110	01110	S (Set)	11001
0111	01111	R (Reset)	00111
1000	10010		
1001	10011		
1010	10110		
1011	10111		
1100	11010		
1101	11011		
1110	11100		
1111	11101		

Figure 4.16 Substitution in 4B/5B block coding





Example 4.5

We need to send data at a 1-Mbps rate. What is the minimum required bandwidth, using a combination of 4B/5B and NRZ-I or Manchester coding?

Solution

First 4B/5B block coding increases the bit rate to 1.25 Mbps. The minimum bandwidth using NRZ-I is $N/2$ or 625 kHz. The Manchester scheme needs a minimum bandwidth of 1 MHz. The first choice needs a lower bandwidth, but has a DC component problem; the second choice needs a higher bandwidth, but does not have a DC component problem.

Figure 4.17 *8B/10B block encoding*

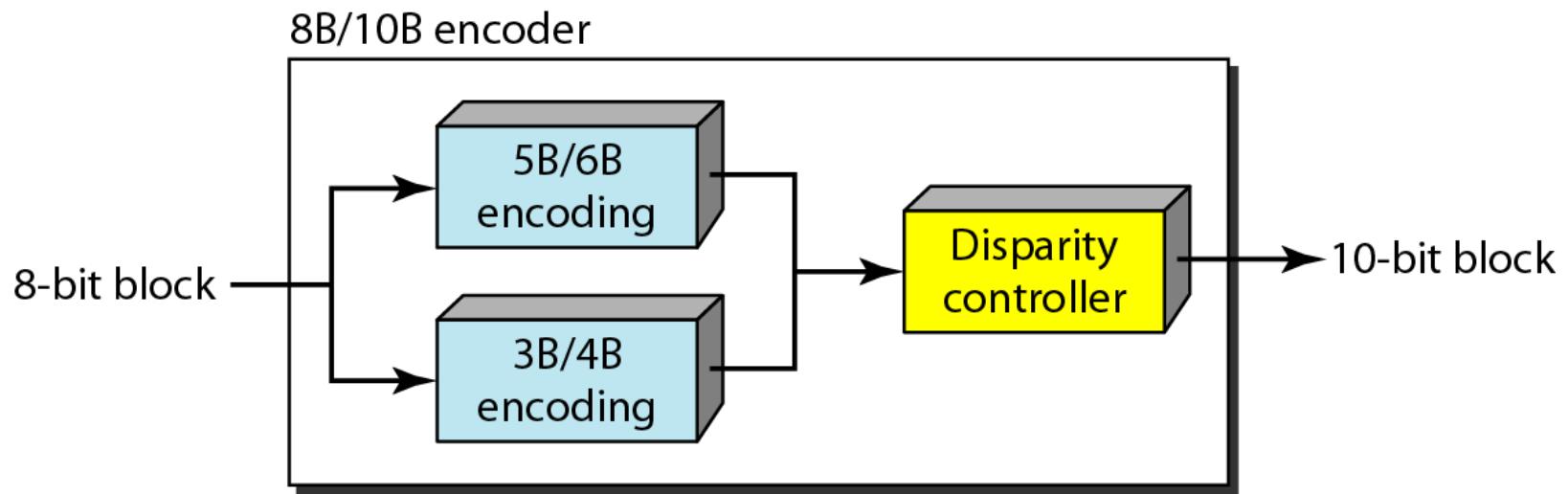


Figure 4.18 AMI used with scrambling

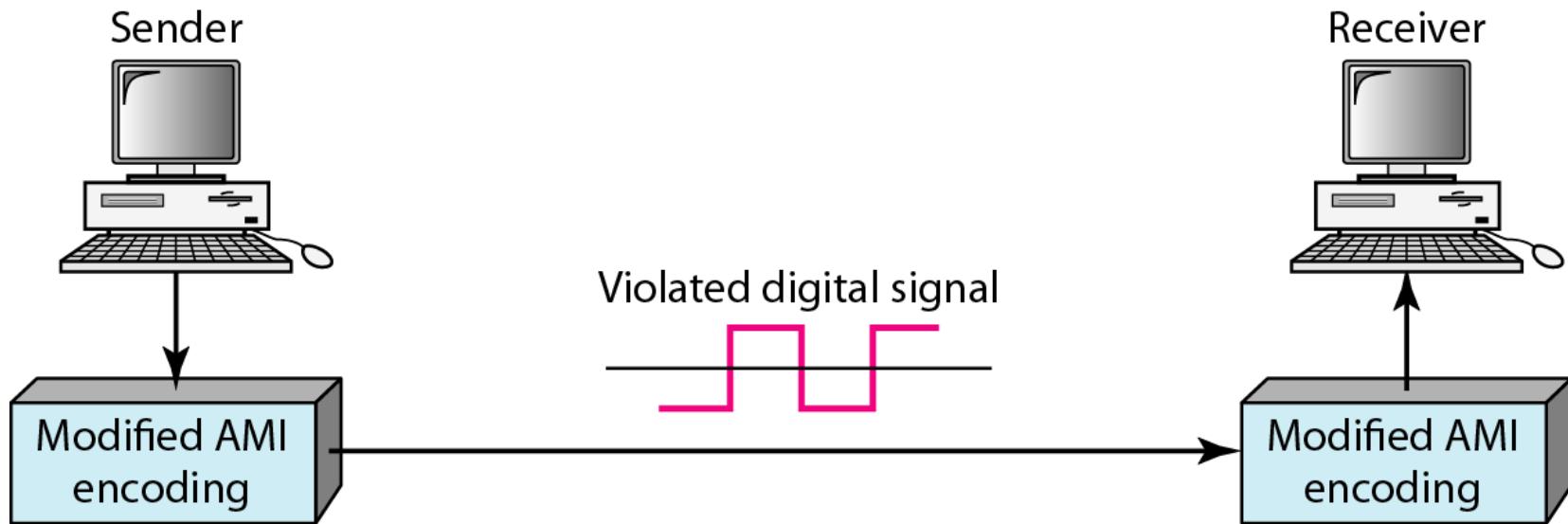
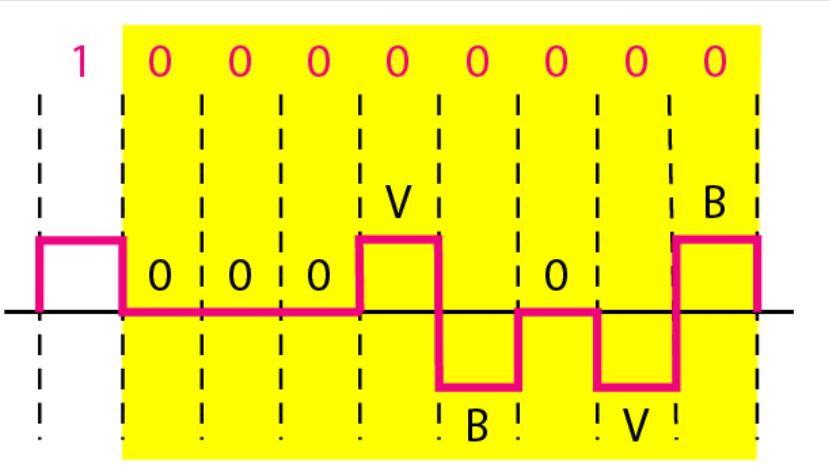
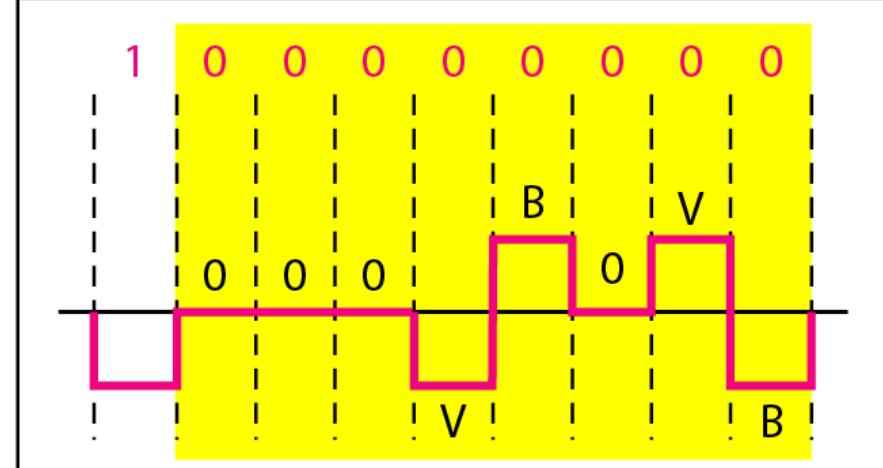


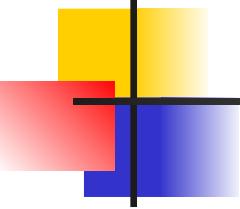
Figure 4.19 Two cases of B8ZS scrambling technique



a. Previous level is positive.



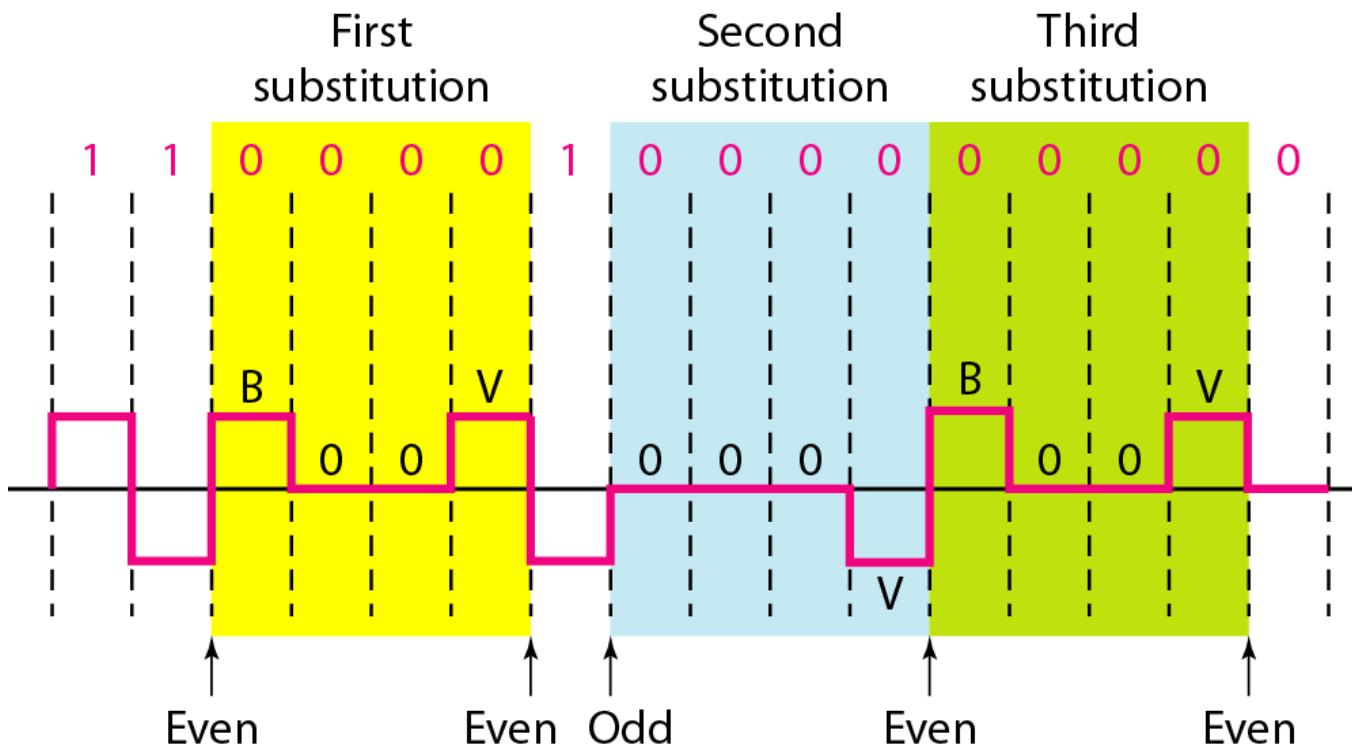
b. Previous level is negative.

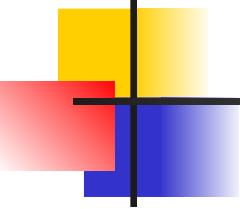


Note

B8ZS substitutes eight consecutive zeros with 000VB0VB.

Figure 4.20 *Different situations in HDB3 scrambling technique*





Note

HDB3 substitutes four consecutive zeros with 000V or B00V depending on the number of nonzero pulses after the last substitution.

4-2 ANALOG-TO-DIGITAL CONVERSION

We have seen in Chapter 3 that a digital signal is superior to an analog signal. The tendency today is to change an analog signal to digital data. In this section we describe two techniques, **pulse code modulation** and **delta modulation**.

Topics discussed in this section:

Pulse Code Modulation (PCM)

Delta Modulation (DM)

Figure 4.21 Components of PCM encoder

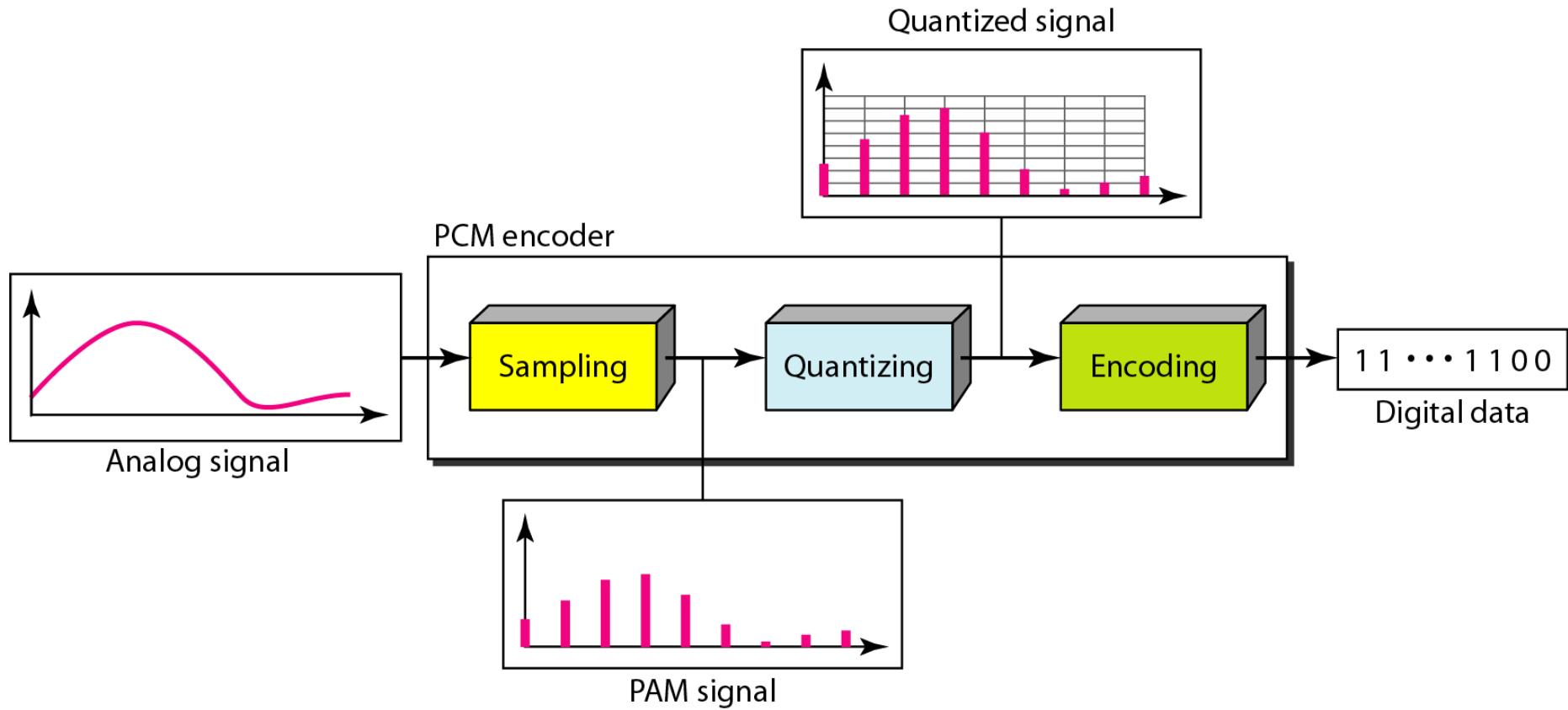
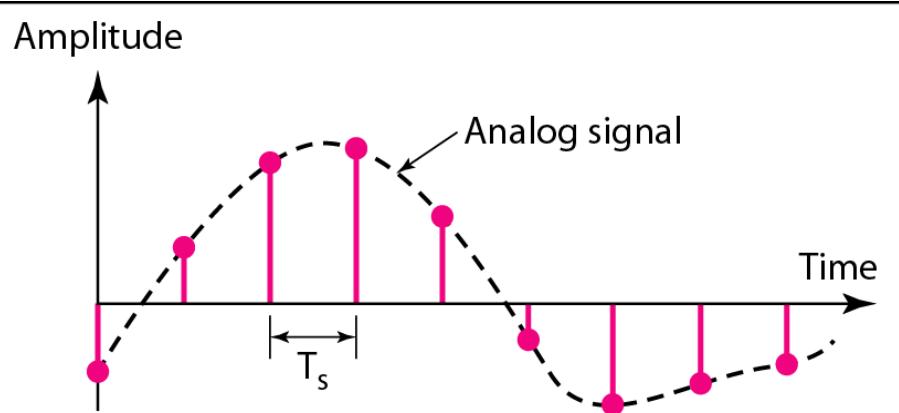
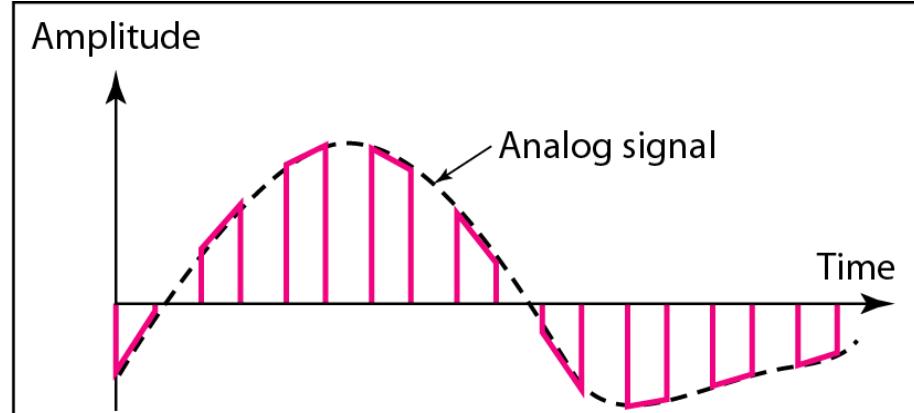


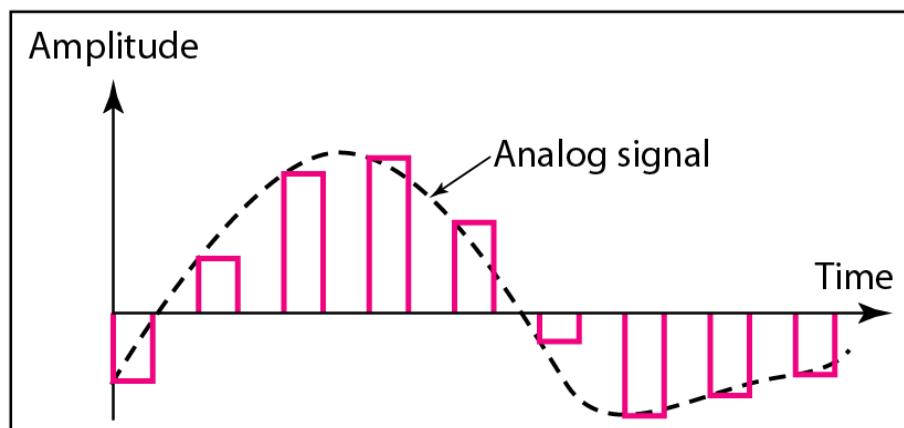
Figure 4.22 Three different sampling methods for PCM



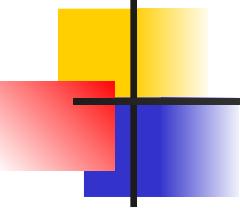
a. Ideal sampling



b. Natural sampling



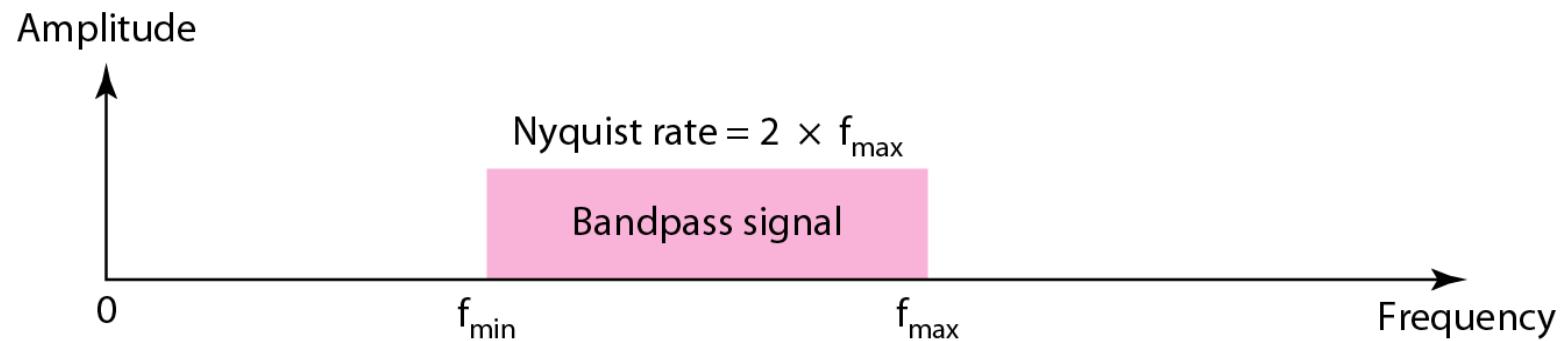
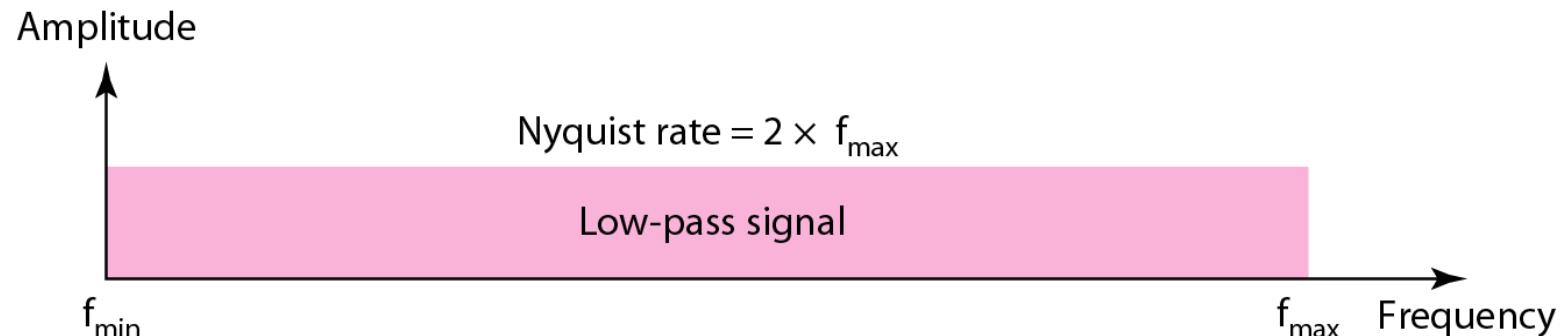
c. Flat-top sampling

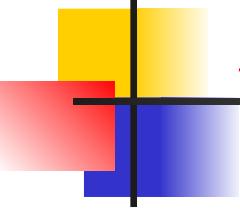


Note

According to the Nyquist theorem, the sampling rate must be at least 2 times the highest frequency contained in the signal.

Figure 4.23 Nyquist sampling rate for low-pass and bandpass signals



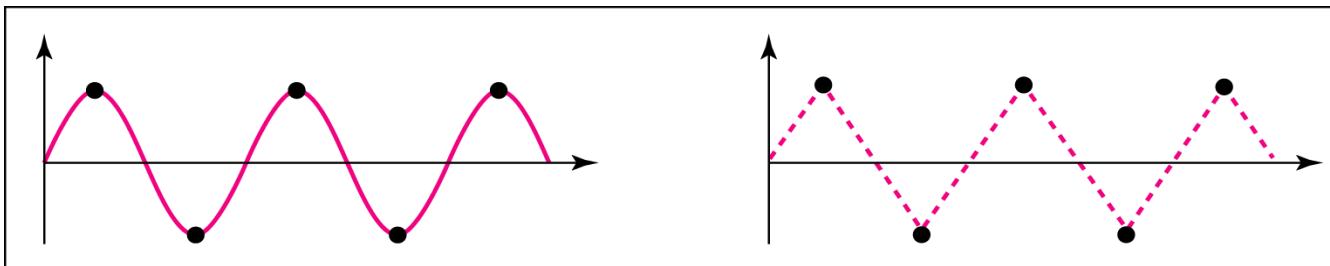


Example 4.6

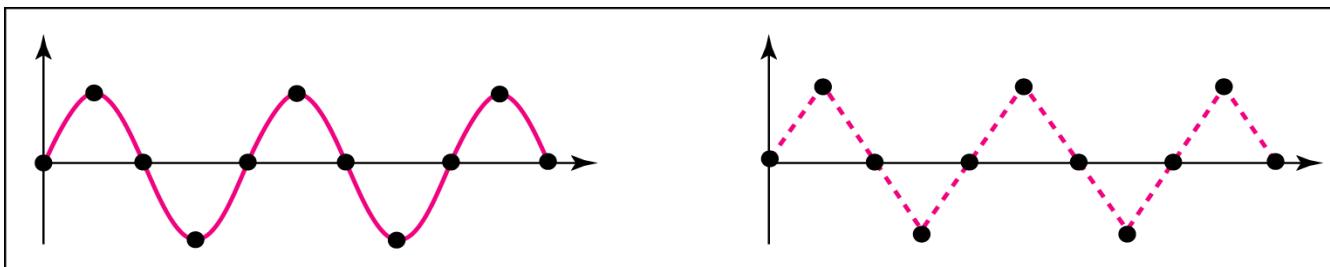
For an intuitive example of the Nyquist theorem, let us sample a simple sine wave at three sampling rates: $f_s = 4f$ (2 times the Nyquist rate), $f_s = 2f$ (Nyquist rate), and $f_s = f$ (one-half the Nyquist rate). Figure 4.24 shows the sampling and the subsequent recovery of the signal.

It can be seen that sampling at the Nyquist rate can create a good approximation of the original sine wave (part a). Oversampling in part b can also create the same approximation, but it is redundant and unnecessary. Sampling below the Nyquist rate (part c) does not produce a signal that looks like the original sine wave.

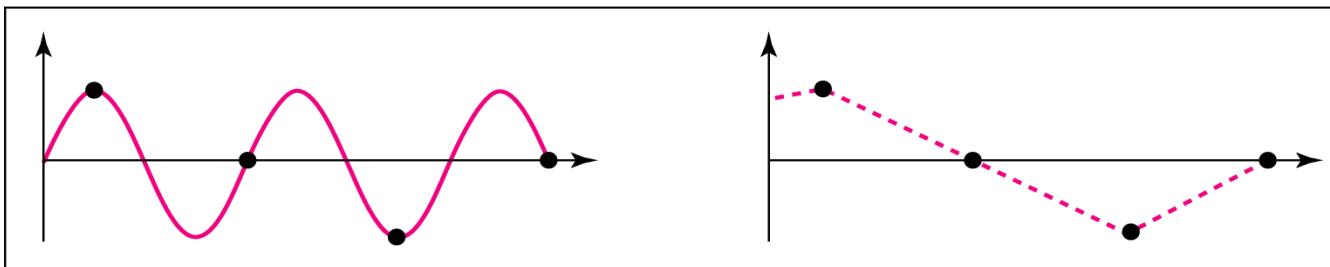
Figure 4.24 Recovery of a sampled sine wave for different sampling rates



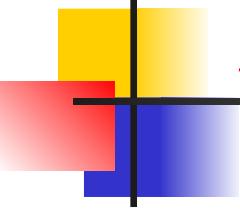
a. Nyquist rate sampling: $f_s = 2 f$



b. Oversampling: $f_s = 4 f$



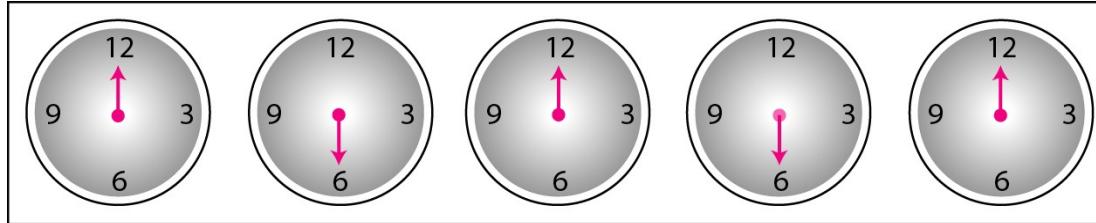
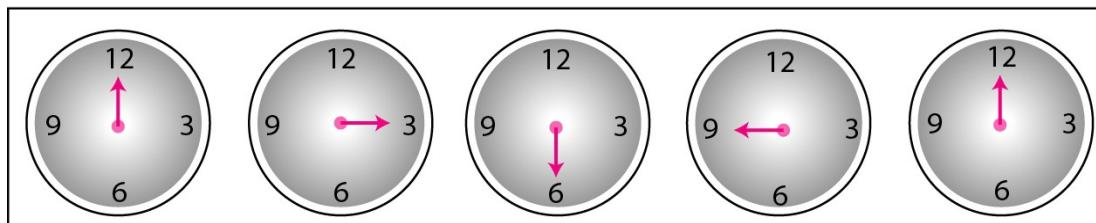
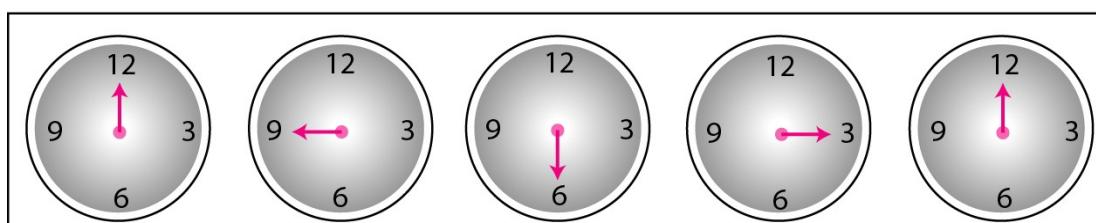
c. Undersampling: $f_s = f$

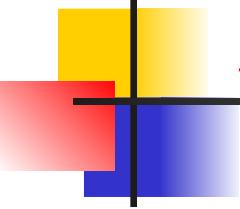


Example 4.7

Consider the revolution of a hand of a clock. The second hand of a clock has a period of 60 s. According to the Nyquist theorem, we need to sample the hand every 30 s ($T_s = T$ or $f_s = 2f$). In Figure 4.25a, the sample points, in order, are 12, 6, 12, 6, 12, and 6. The receiver of the samples cannot tell if the clock is moving forward or backward. In part b, we sample at double the Nyquist rate (every 15 s). The sample points are 12, 3, 6, 9, and 12. The clock is moving forward. In part c, we sample below the Nyquist rate ($T_s = T$ or $f_s = f$). The sample points are 12, 9, 6, 3, and 12. Although the clock is moving forward, the receiver thinks that the clock is moving backward.

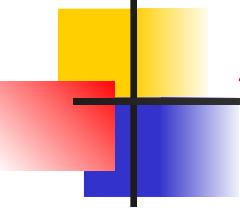
Figure 4.25 Sampling of a clock with only one hand

- a. Sampling at Nyquist rate: $T_s = T \frac{1}{2}$
- 
- Samples can mean that the clock is moving either forward or backward.
(12-6-12-6-12)
- b. Oversampling (above Nyquist rate): $T_s = T \frac{1}{4}$
- 
- Samples show clock is moving forward.
(12-3-6-9-12)
- c. Undersampling (below Nyquist rate): $T_s = T \frac{3}{4}$
- 
- Samples show clock is moving backward.
(12-9-6-3-12)



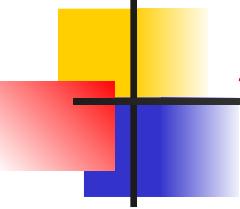
Example 4.8

An example related to Example 4.7 is the seemingly backward rotation of the wheels of a forward-moving car in a movie. This can be explained by under-sampling. A movie is filmed at 24 frames per second. If a wheel is rotating more than 12 times per second, the undersampling creates the impression of a backward rotation.



Example 4.9

Telephone companies digitize voice by assuming a maximum frequency of 4000 Hz. The sampling rate therefore is 8000 samples per second.

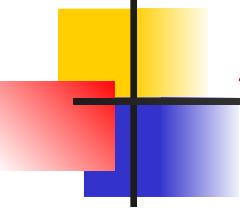


Example 4.10

A complex low-pass signal has a bandwidth of 200 kHz. What is the minimum sampling rate for this signal?

Solution

The bandwidth of a low-pass signal is between 0 and f , where f is the maximum frequency in the signal. Therefore, we can sample this signal at 2 times the highest frequency (200 kHz). The sampling rate is therefore 400,000 samples per second.



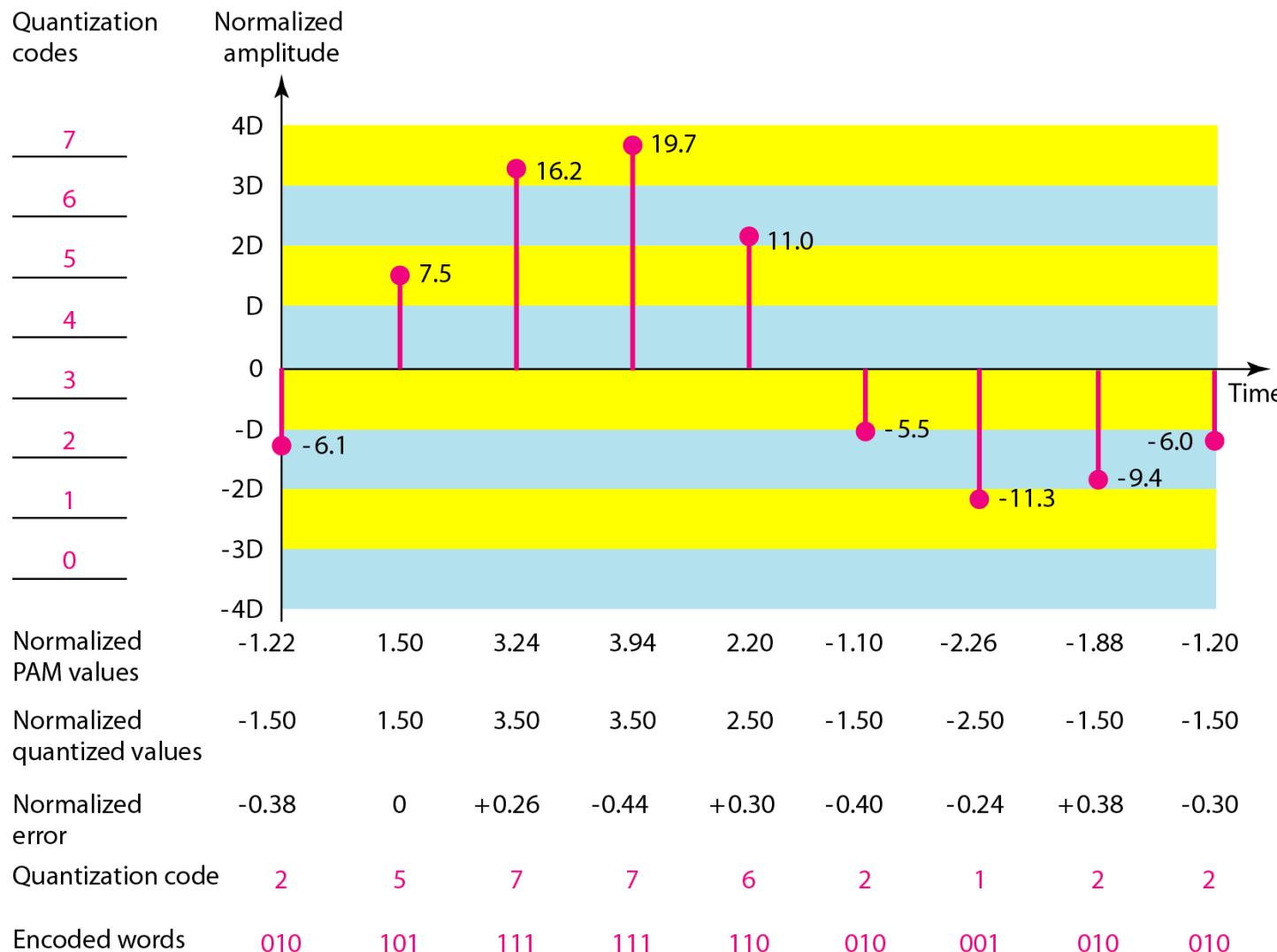
Example 4.11

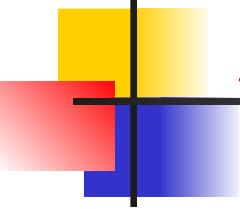
*A complex bandpass signal has a bandwidth of 200 kHz.
What is the minimum sampling rate for this signal?*

Solution

We cannot find the minimum sampling rate in this case because we do not know where the bandwidth starts or ends. We do not know the maximum frequency in the signal.

Figure 4.26 Quantization and encoding of a sampled signal





Example 4.12

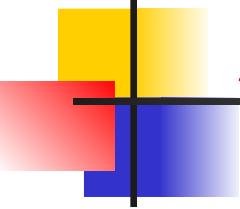
What is the SNR_{dB} in the example of Figure 4.26?

Solution

We can use the formula to find the quantization. We have eight levels and 3 bits per sample, so

$$SNR_{dB} = 6.02(3) + 1.76 = 19.82 \text{ dB}$$

Increasing the number of levels increases the SNR.



Example 4.13

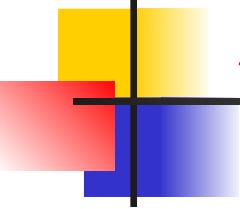
A telephone subscriber line must have an SNR_{dB} above 40. What is the minimum number of bits per sample?

Solution

We can calculate the number of bits as

$$SNR_{dB} = 6.02n_b + 1.76 = 40 \rightarrow n = 6.35$$

Telephone companies usually assign 7 or 8 bits per sample.



Example 4.14

We want to digitize the human voice. What is the bit rate, assuming 8 bits per sample?

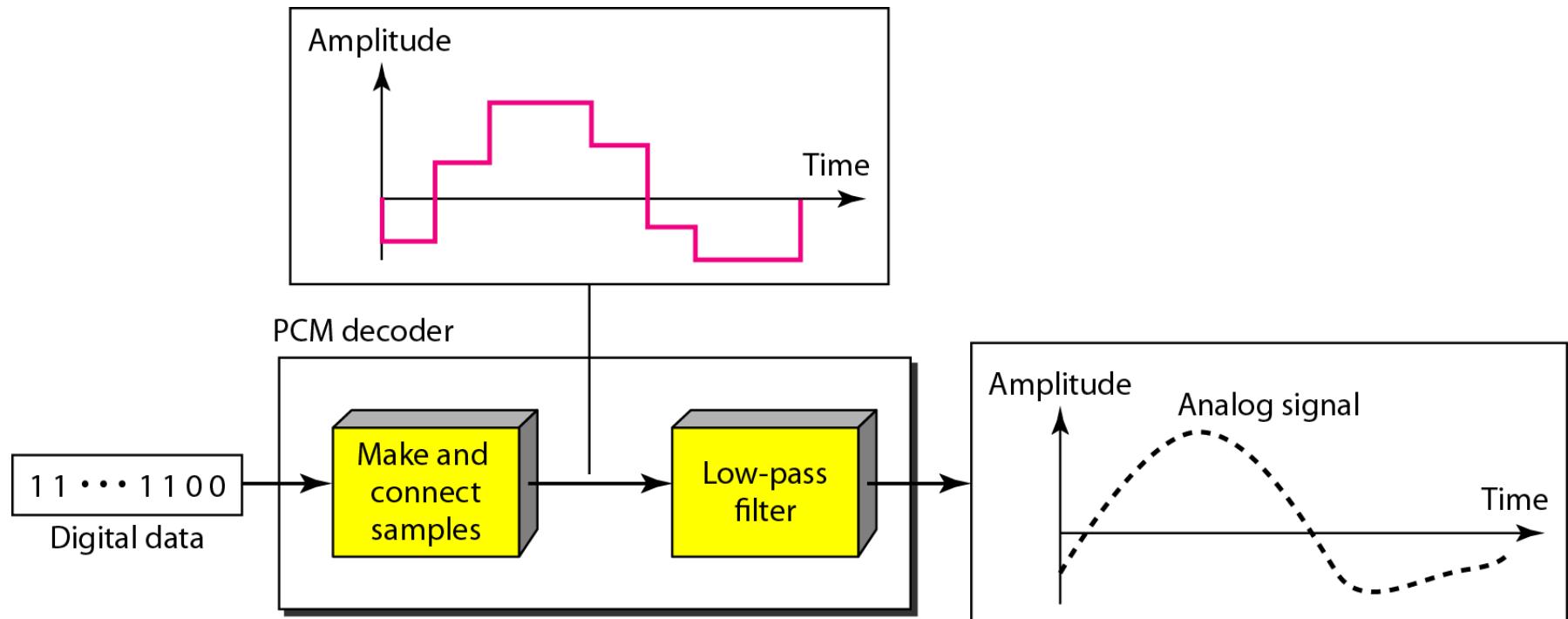
Solution

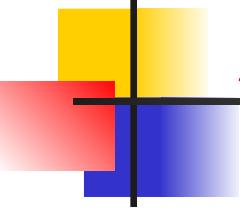
The human voice normally contains frequencies from 0 to 4000 Hz. So the sampling rate and bit rate are calculated as follows:

$$\text{Sampling rate} = 4000 \times 2 = 8000 \text{ samples/s}$$

$$\text{Bit rate} = 8000 \times 8 = 64,000 \text{ bps} = 64 \text{ kbps}$$

Figure 4.27 Components of a PCM decoder





Example 4.15

We have a low-pass analog signal of 4 kHz. If we send the analog signal, we need a channel with a minimum bandwidth of 4 kHz. If we digitize the signal and send 8 bits per sample, we need a channel with a minimum bandwidth of $8 \times 4 \text{ kHz} = 32 \text{ kHz}$.

Figure 4.28 *The process of delta modulation*

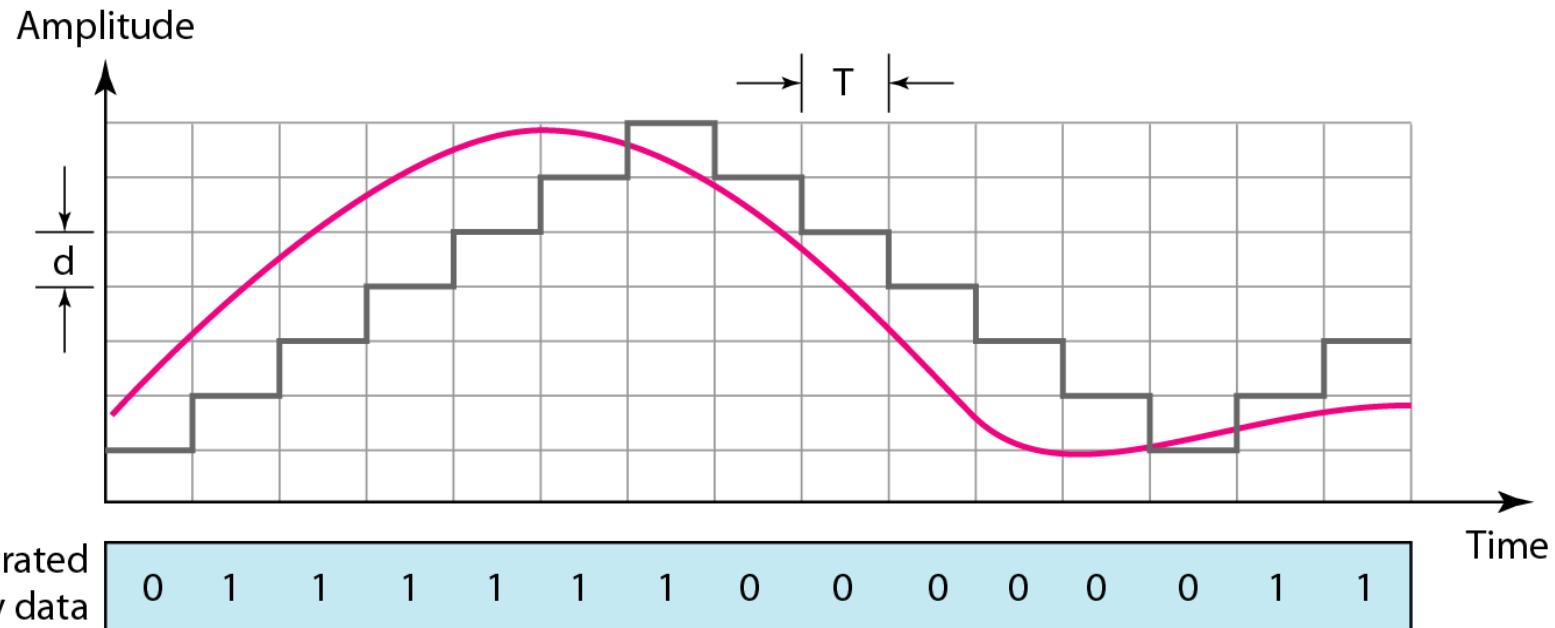


Figure 4.29 *Delta modulation components*

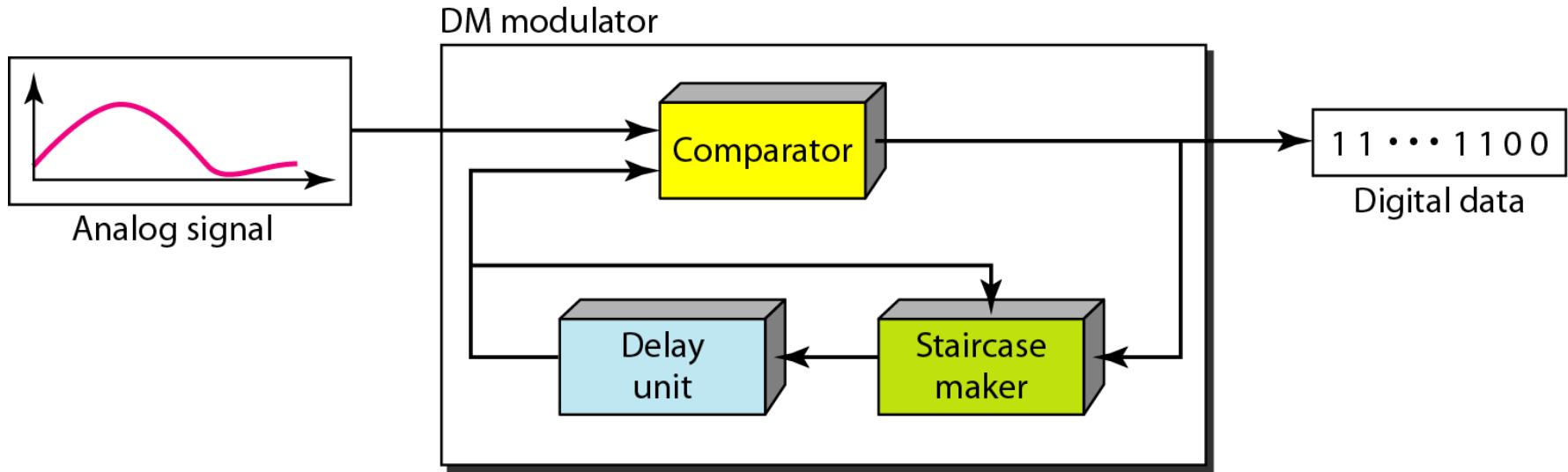
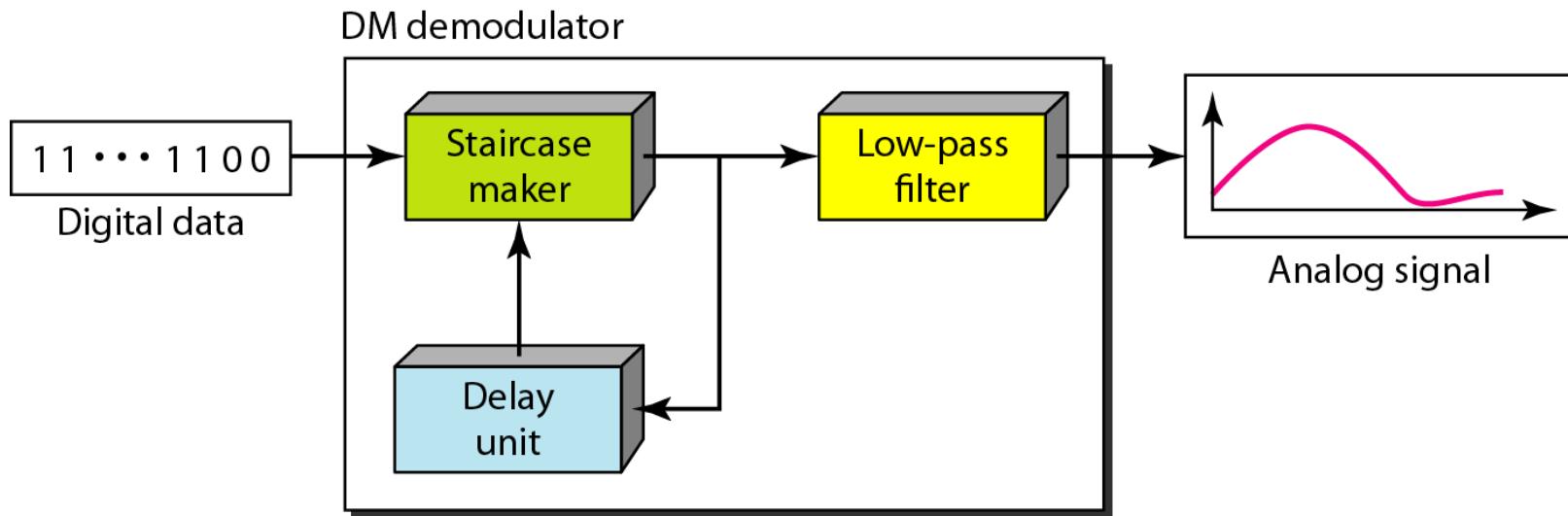


Figure 4.30 *Delta demodulation components*



4-3 TRANSMISSION MODES

The transmission of binary data across a link can be accomplished in either parallel or serial mode. In parallel mode, multiple bits are sent with each clock tick. In serial mode, 1 bit is sent with each clock tick. While there is only one way to send parallel data, there are three subclasses of serial transmission: asynchronous, synchronous, and isochronous.

Topics discussed in this section:

Parallel Transmission

Serial Transmission

Figure 4.31 *Data transmission and modes*

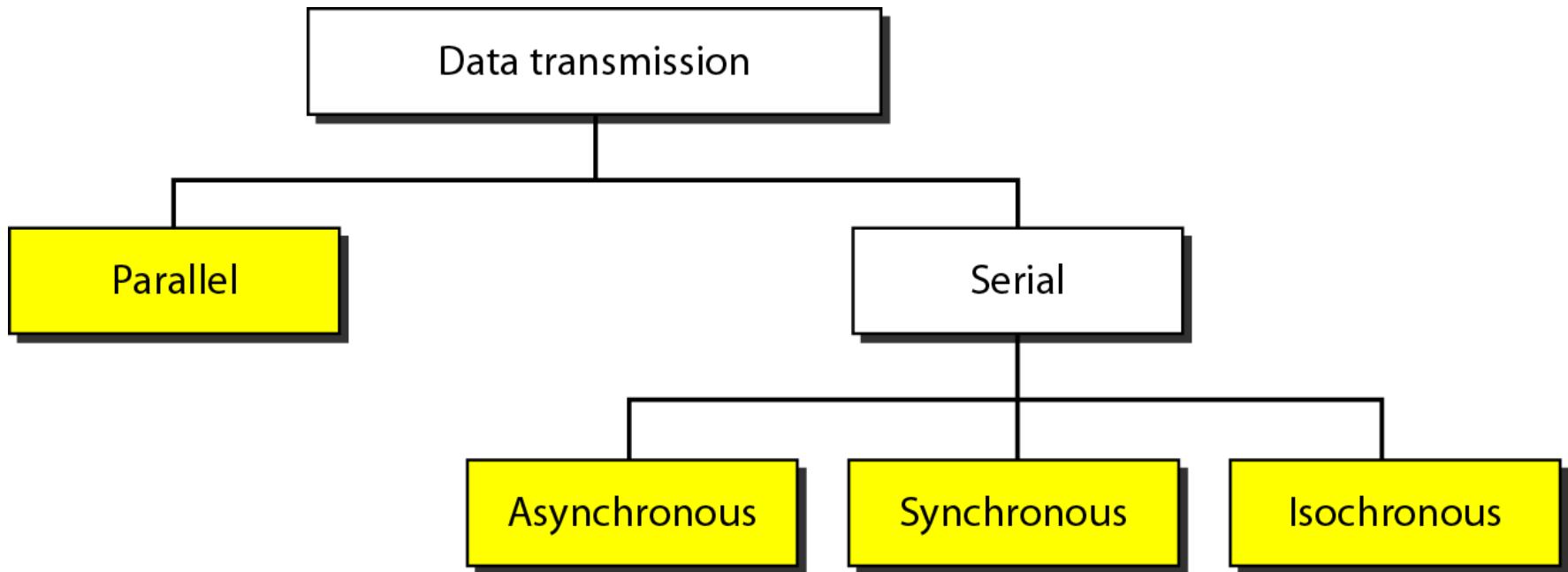


Figure 4.32 Parallel transmission

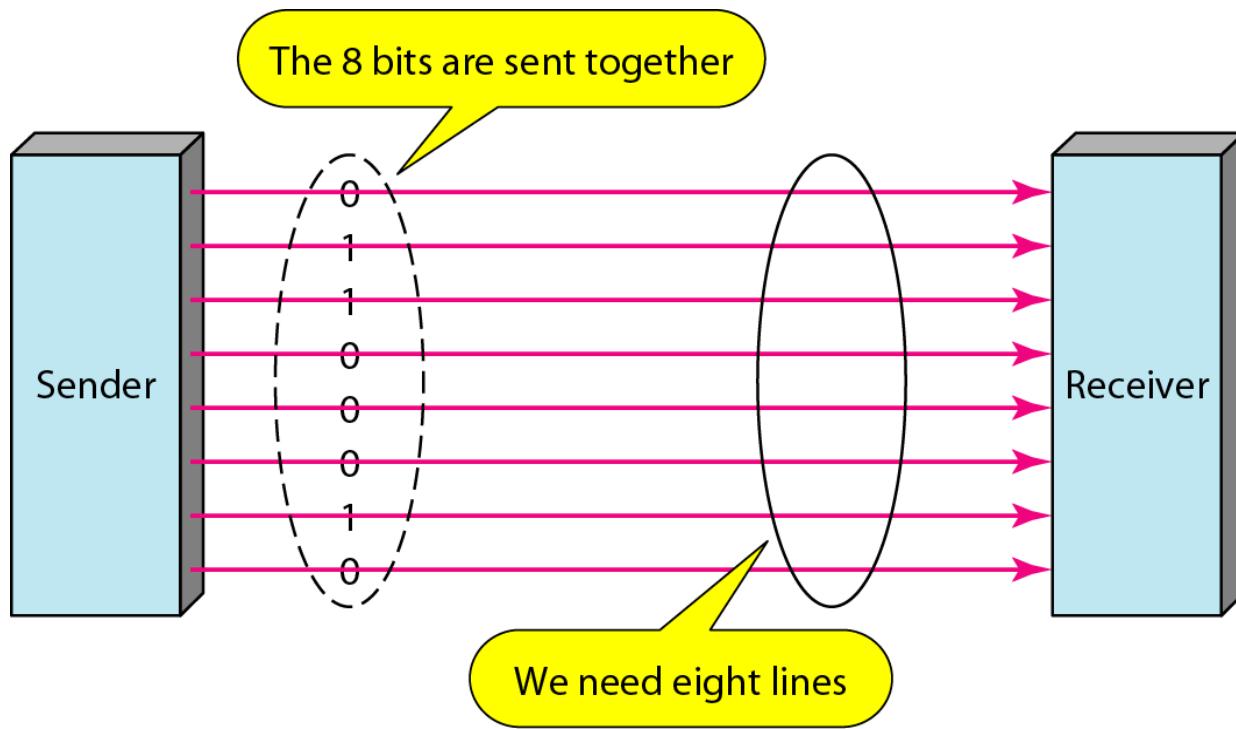
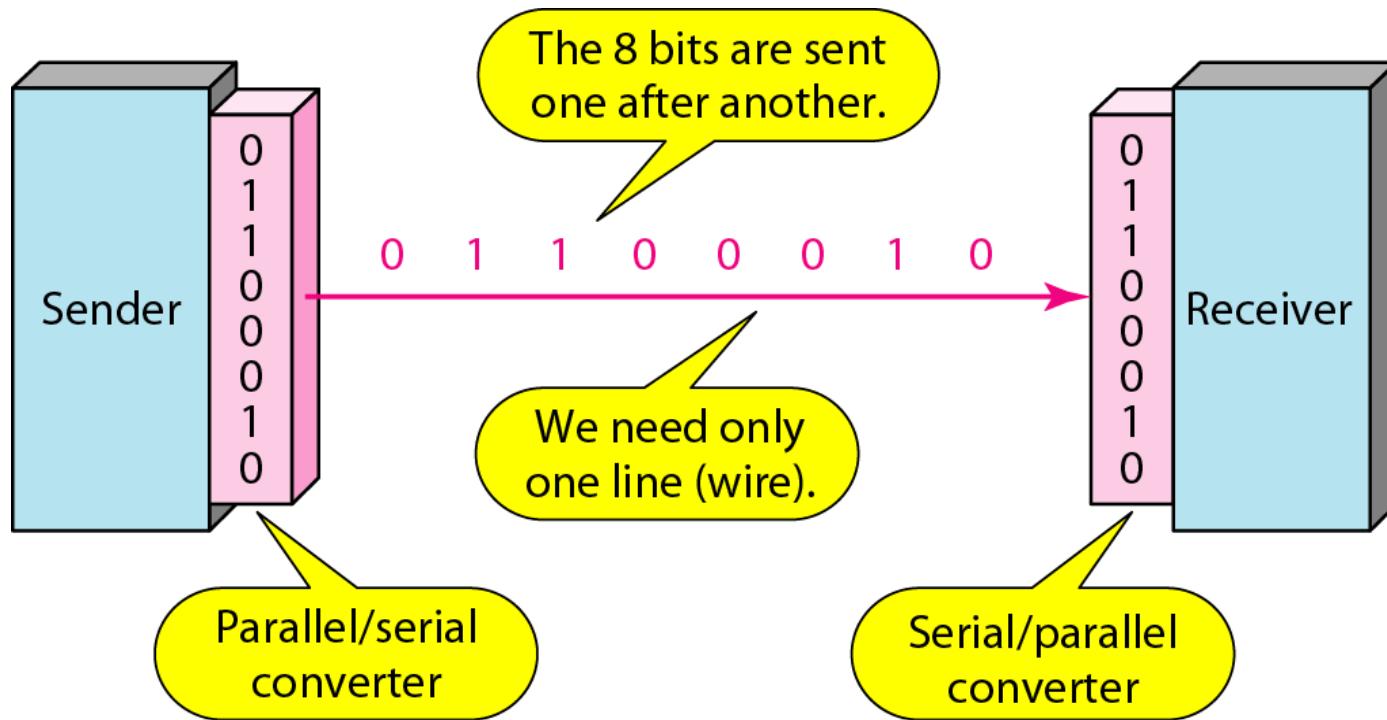
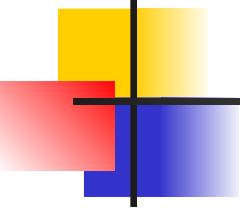


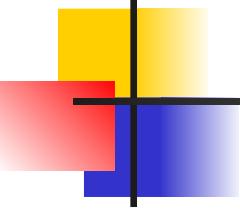
Figure 4.33 *Serial transmission*





Note

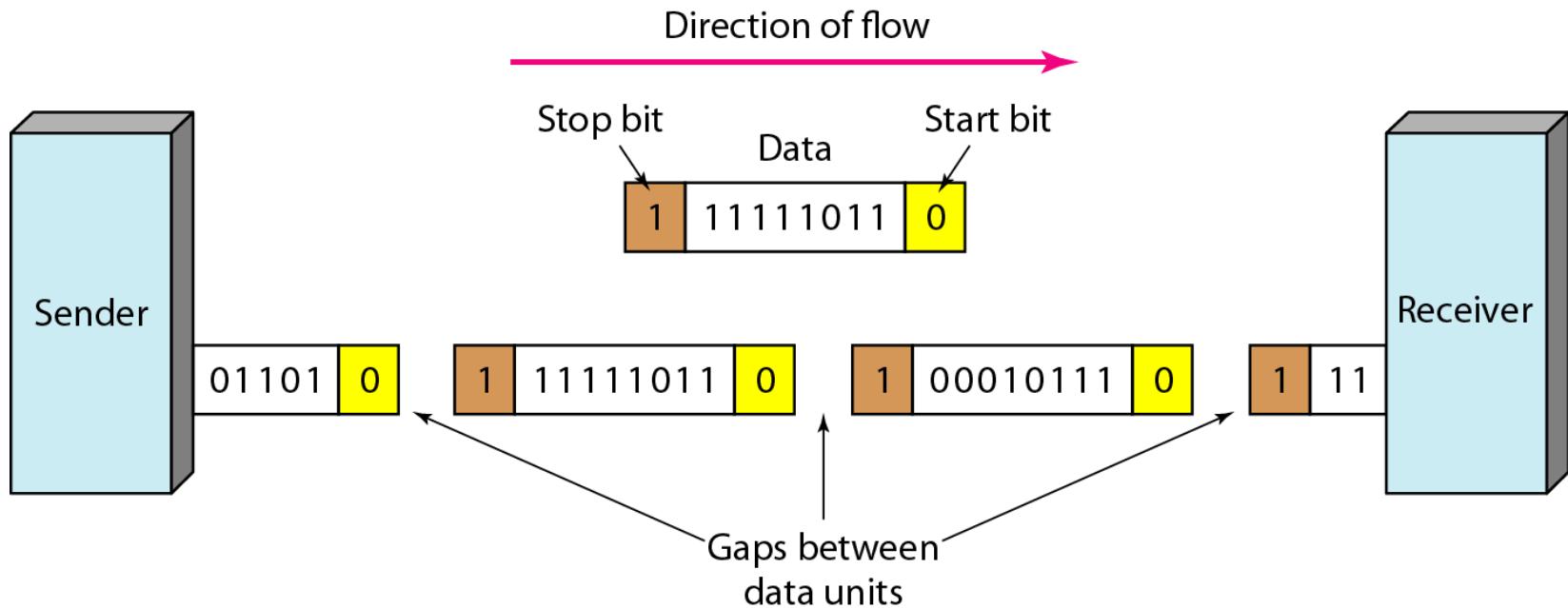
In asynchronous transmission, we send 1 start bit (0) at the beginning and 1 or more stop bits (1s) at the end of each byte. There may be a gap between each byte.

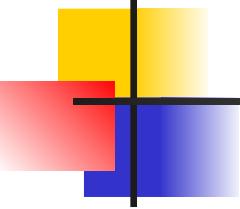


Note

**Asynchronous here means
“asynchronous at the byte level,”
but the bits are still synchronized;
their durations are the same.**

Figure 4.34 Asynchronous transmission

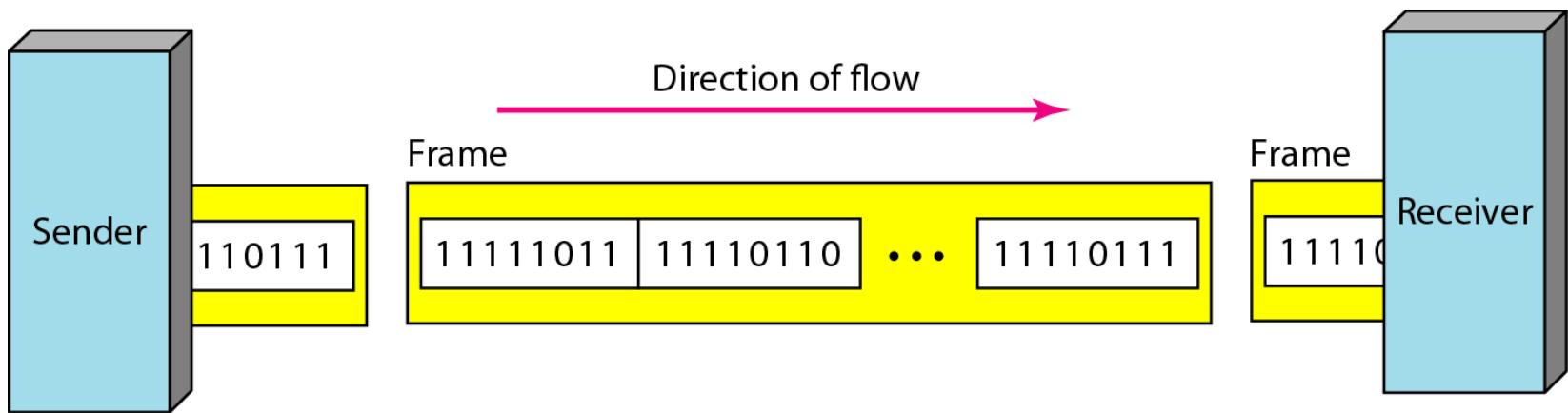




Note

In synchronous transmission, we send bits one after another without start or stop bits or gaps. It is the responsibility of the receiver to group the bits.

Figure 4.35 *Synchronous transmission*



Chapter 5

Analog Transmission

5-1 DIGITAL-TO-ANALOG CONVERSION

Digital-to-analog conversion is the process of changing one of the characteristics of an analog signal based on the information in digital data.

Topics discussed in this section:

Aspects of Digital-to-Analog Conversion

Amplitude Shift Keying

Frequency Shift Keying

Phase Shift Keying

Quadrature Amplitude Modulation

Figure 5.1 *Digital-to-analog conversion*

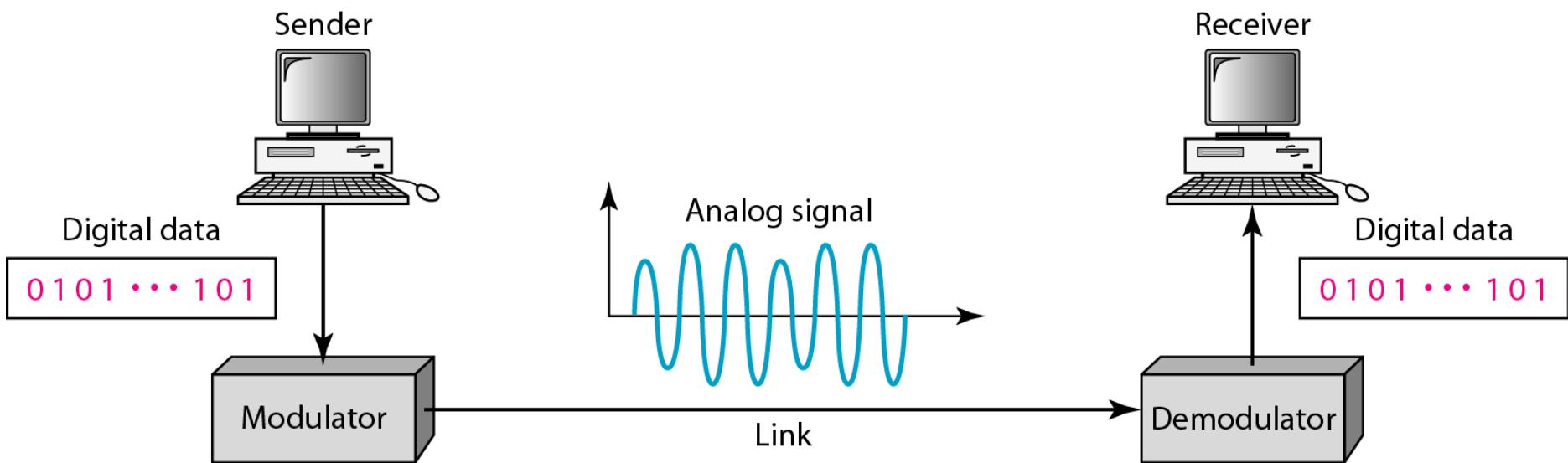
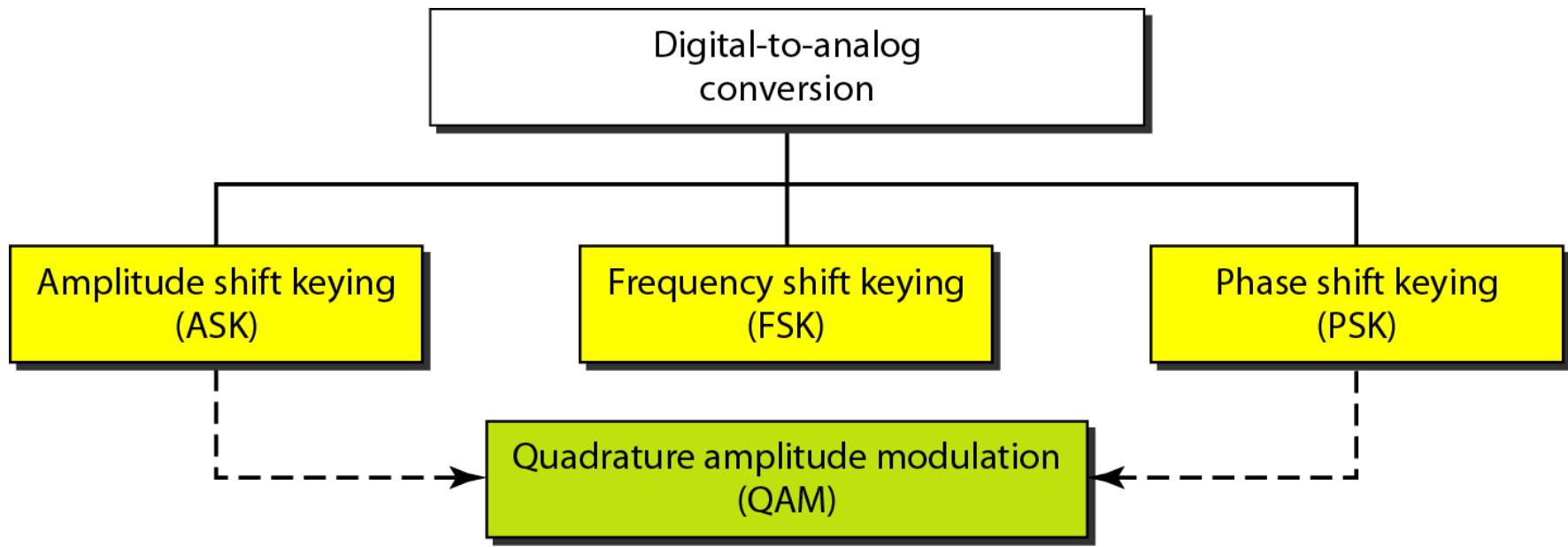
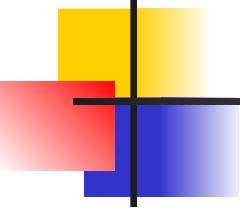


Figure 5.2 *Types of digital-to-analog conversion*

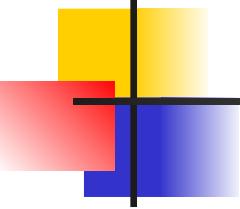




Note

Bit rate is the number of bits per second. Baud rate is the number of signal elements per second.

In the analog transmission of digital data, the baud rate is less than or equal to the bit rate.



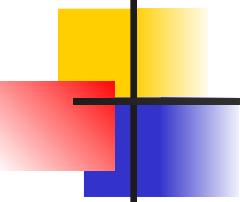
Example 5.1

An analog signal carries 4 bits per signal element. If 1000 signal elements are sent per second, find the bit rate.

Solution

In this case, $r = 4$, $S = 1000$, and N is unknown. We can find the value of N from

$$S = N \times \frac{1}{r} \quad \text{or} \quad N = S \times r = 1000 \times 4 = 4000 \text{ bps}$$



Example 5.2

An analog signal has a bit rate of 8000 bps and a baud rate of 1000 baud. How many data elements are carried by each signal element? How many signal elements do we need?

Solution

In this example, $S = 1000$, $N = 8000$, and r and L are unknown. We find first the value of r and then the value of L .

$$S = N \times \frac{1}{r} \quad \rightarrow \quad r = \frac{N}{S} = \frac{8000}{1000} = 8 \text{ bits/baud}$$

$$r = \log_2 L \quad \rightarrow \quad L = 2^r = 2^8 = 256$$

Figure 5.3 *Binary amplitude shift keying*

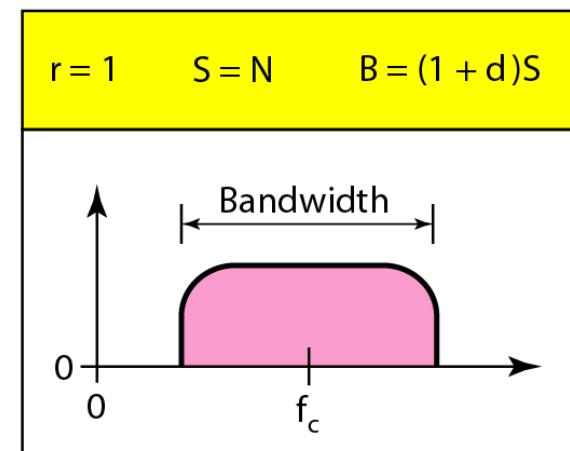
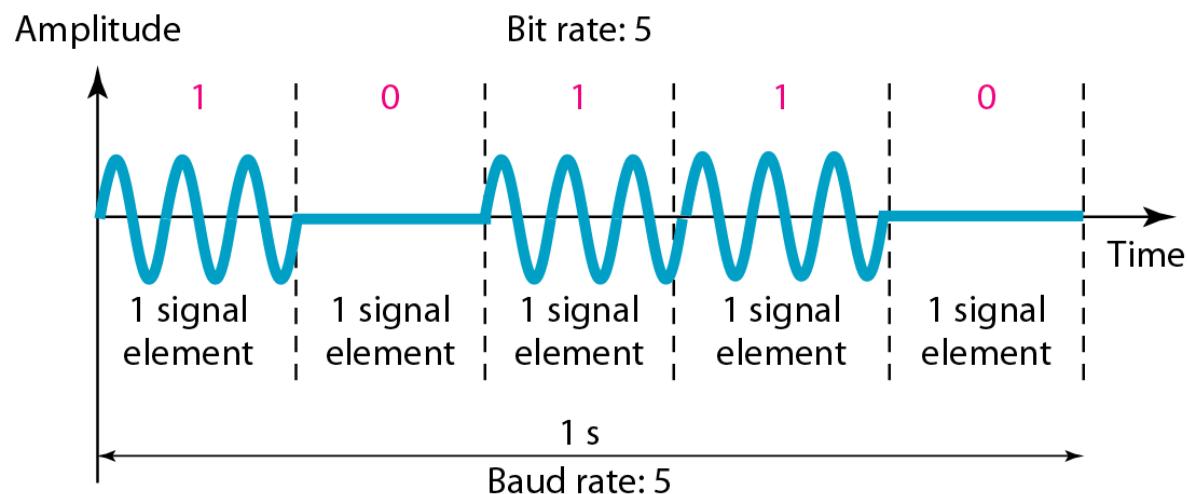
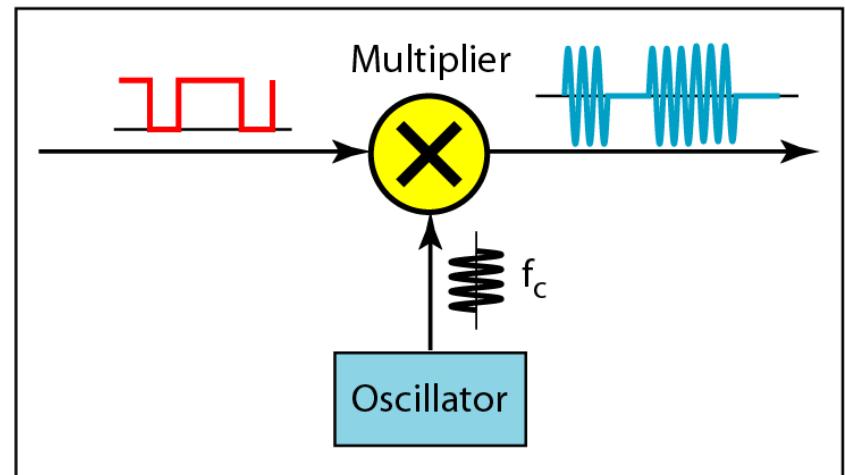
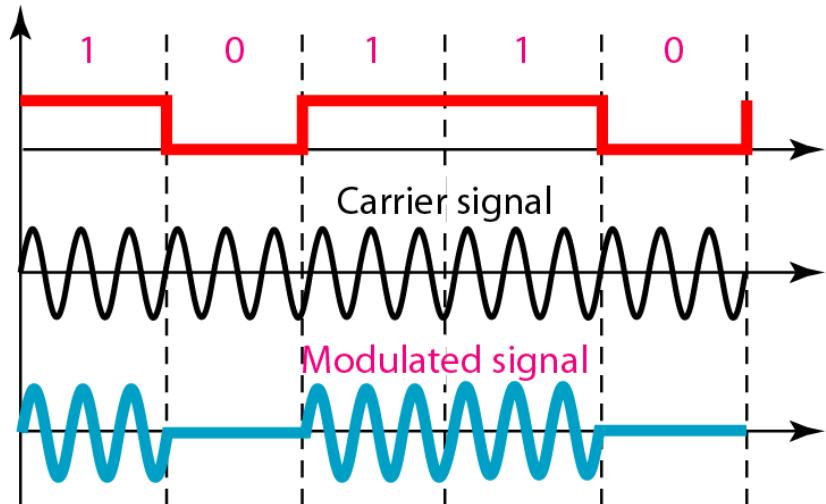
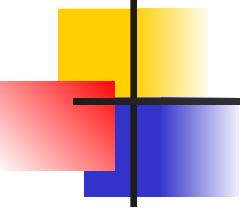


Figure 5.4 Implementation of binary ASK





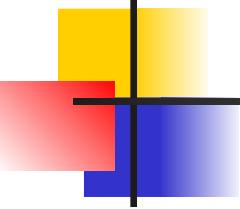
Example 5.3

We have an available bandwidth of 100 kHz which spans from 200 to 300 kHz. What are the carrier frequency and the bit rate if we modulated our data by using ASK with $d = 1$?

Solution

The middle of the bandwidth is located at 250 kHz. This means that our carrier frequency can be at $f_c = 250$ kHz. We can use the formula for bandwidth to find the bit rate (with $d = 1$ and $r = 1$).

$$B = (1 + d) \times S = 2 \times N \times \frac{1}{r} = 2 \times N = 100 \text{ kHz} \quad \rightarrow \quad N = 50 \text{ kbps}$$



Example 5.4

In data communications, we normally use full-duplex links with communication in both directions. We need to divide the bandwidth into two with two carrier frequencies, as shown in Figure 5.5. The figure shows the positions of two carrier frequencies and the bandwidths. The available bandwidth for each direction is now 50 kHz, which leaves us with a data rate of 25 kbps in each direction.

Figure 5.5 Bandwidth of full-duplex ASK used in Example 5.4

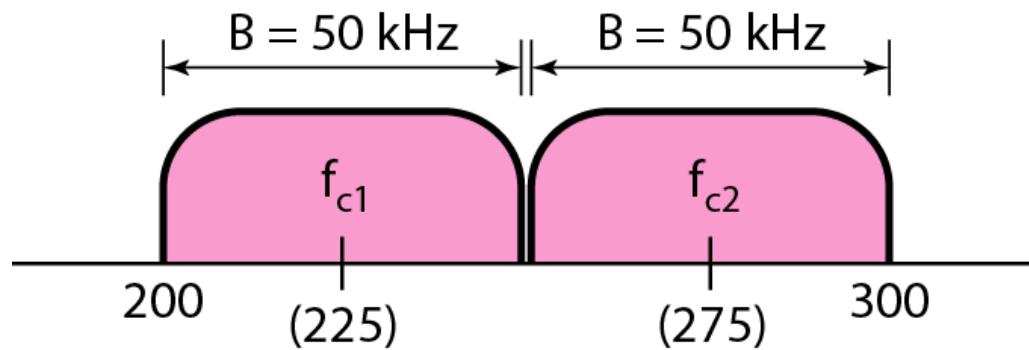
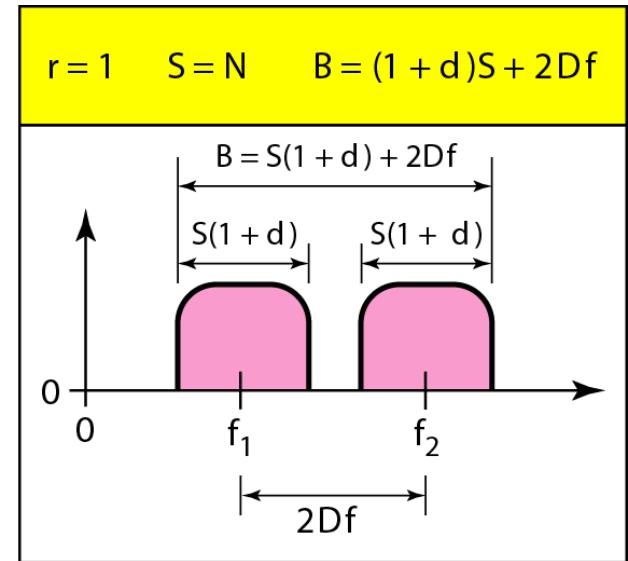
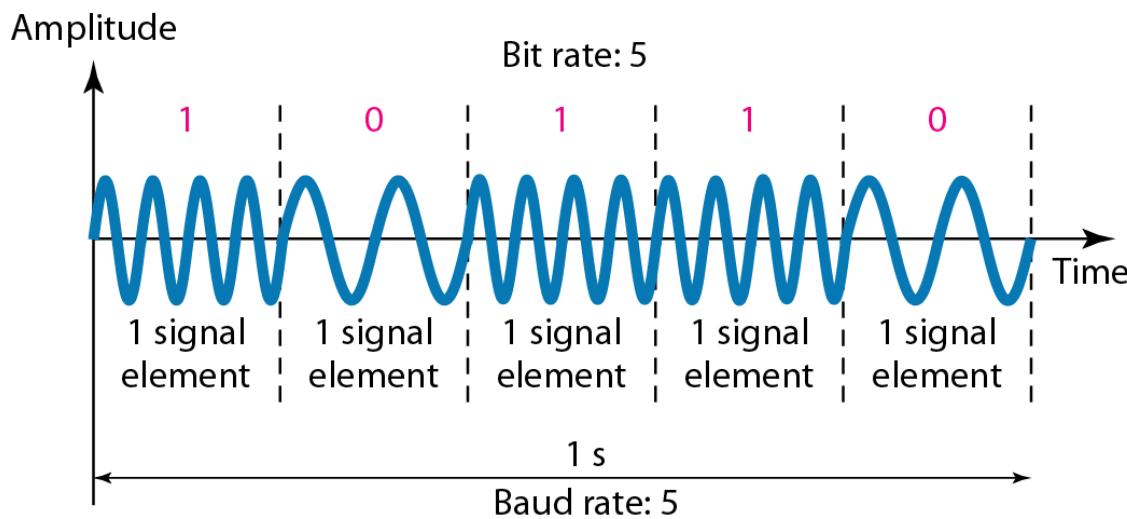
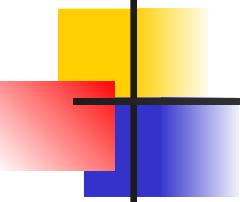


Figure 5.6 *Binary frequency shift keying*





Example 5.5

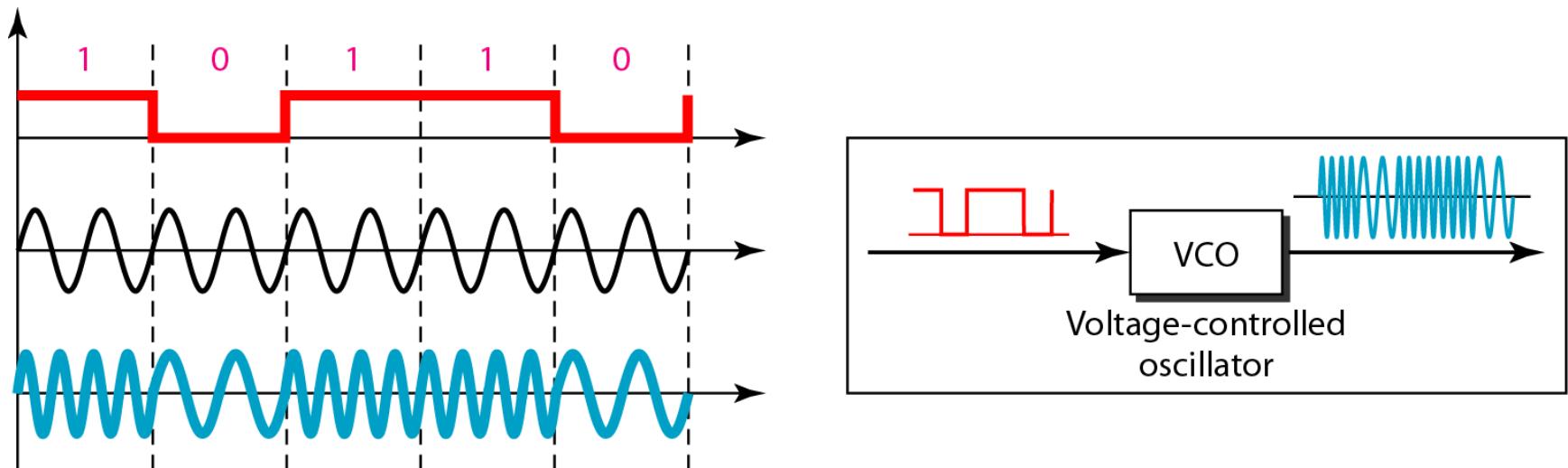
We have an available bandwidth of 100 kHz which spans from 200 to 300 kHz. What should be the carrier frequency and the bit rate if we modulated our data by using FSK with $d = 1$?

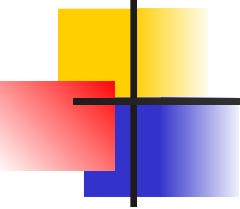
Solution

This problem is similar to Example 5.3, but we are modulating by using FSK. The midpoint of the band is at 250 kHz. We choose $2\Delta f$ to be 50 kHz; this means

$$B = (1 + d) \times S + 2\Delta f = 100 \quad \rightarrow \quad 2S = 50 \text{ kHz} \quad S = 25 \text{ baud} \quad N = 25 \text{ kbps}$$

Figure 5.7 Bandwidth of MFSK used in Example 5.6





Example 5.6

We need to send data 3 bits at a time at a bit rate of 3 Mbps. The carrier frequency is 10 MHz. Calculate the number of levels (different frequencies), the baud rate, and the bandwidth.

Solution

We can have $L = 2^3 = 8$. The baud rate is $S = 3 \text{ MHz}/3 = 1000 \text{ Mbaud}$. This means that the carrier frequencies must be 1 MHz apart ($2\Delta f = 1 \text{ MHz}$). The bandwidth is $B = 8 \times 1000 = 8000$. Figure 5.8 shows the allocation of frequencies and bandwidth.

Figure 5.8 Bandwidth of MFSK used in Example 5.6

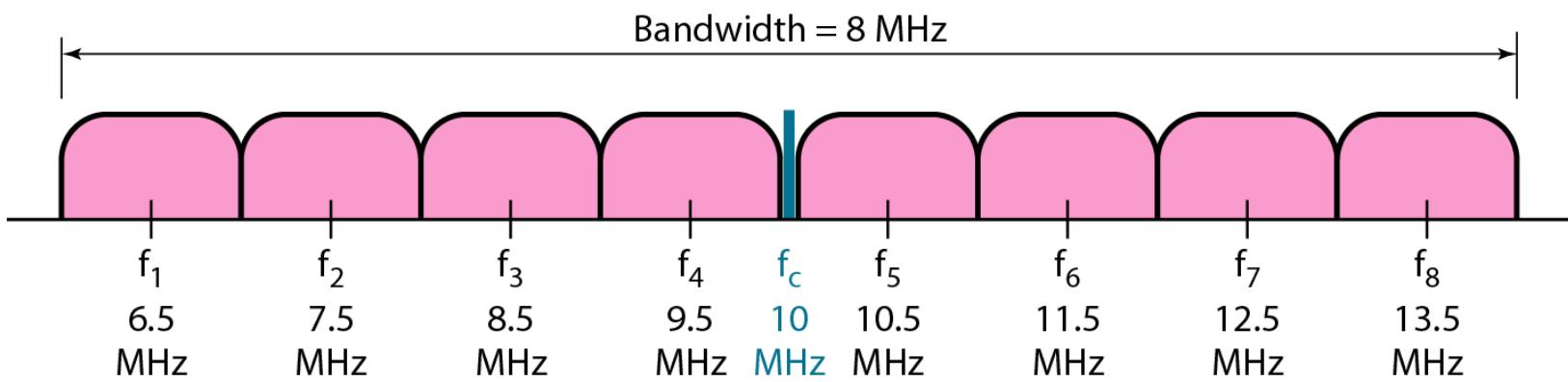


Figure 5.9 *Binary phase shift keying*

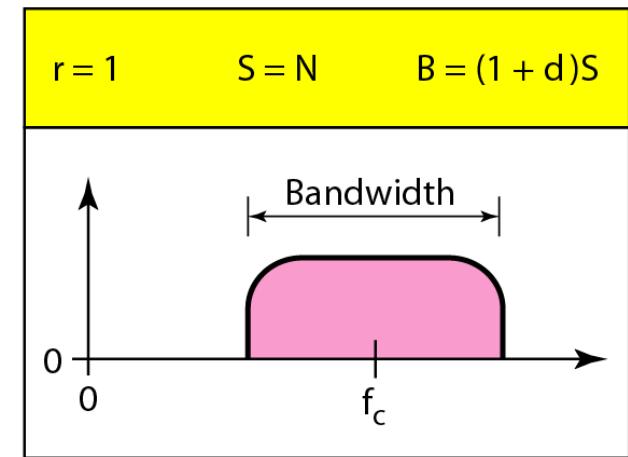
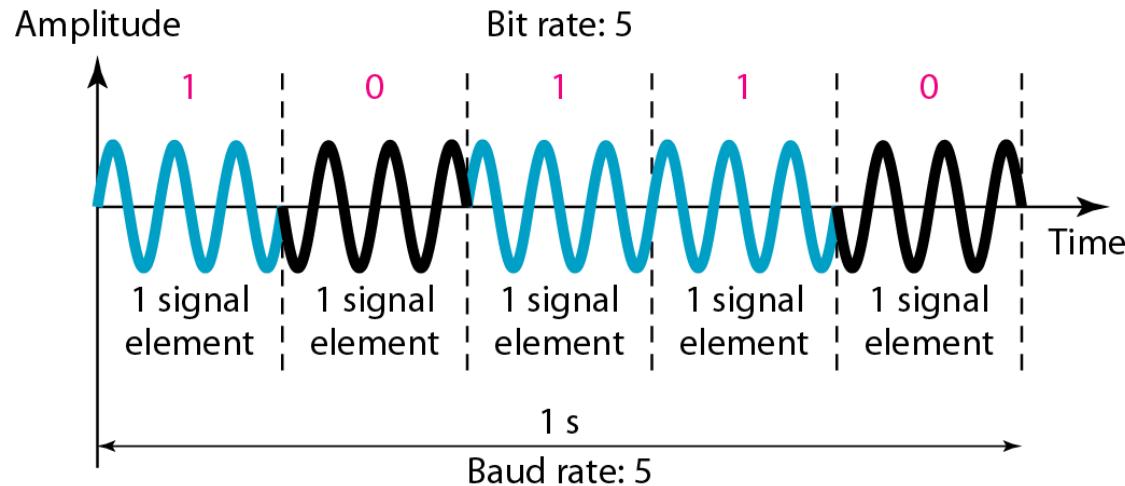


Figure 5.10 Implementation of BASK

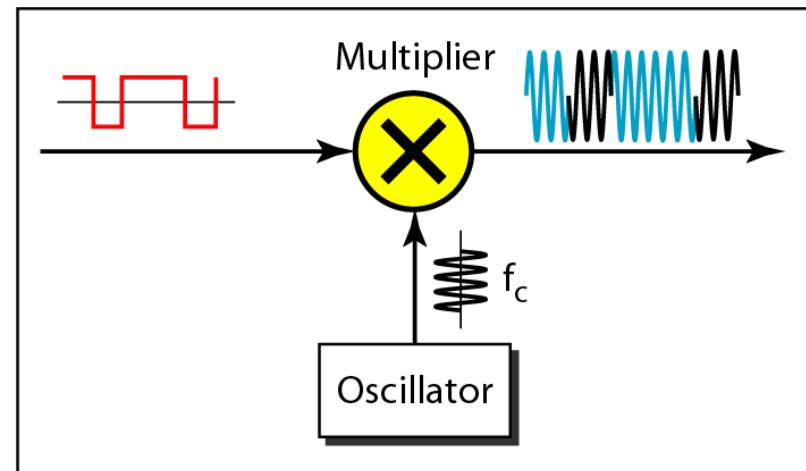
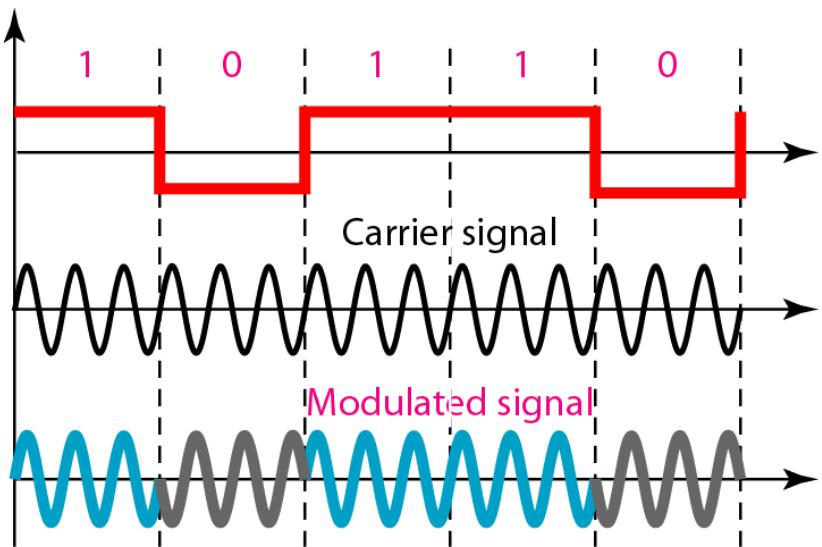
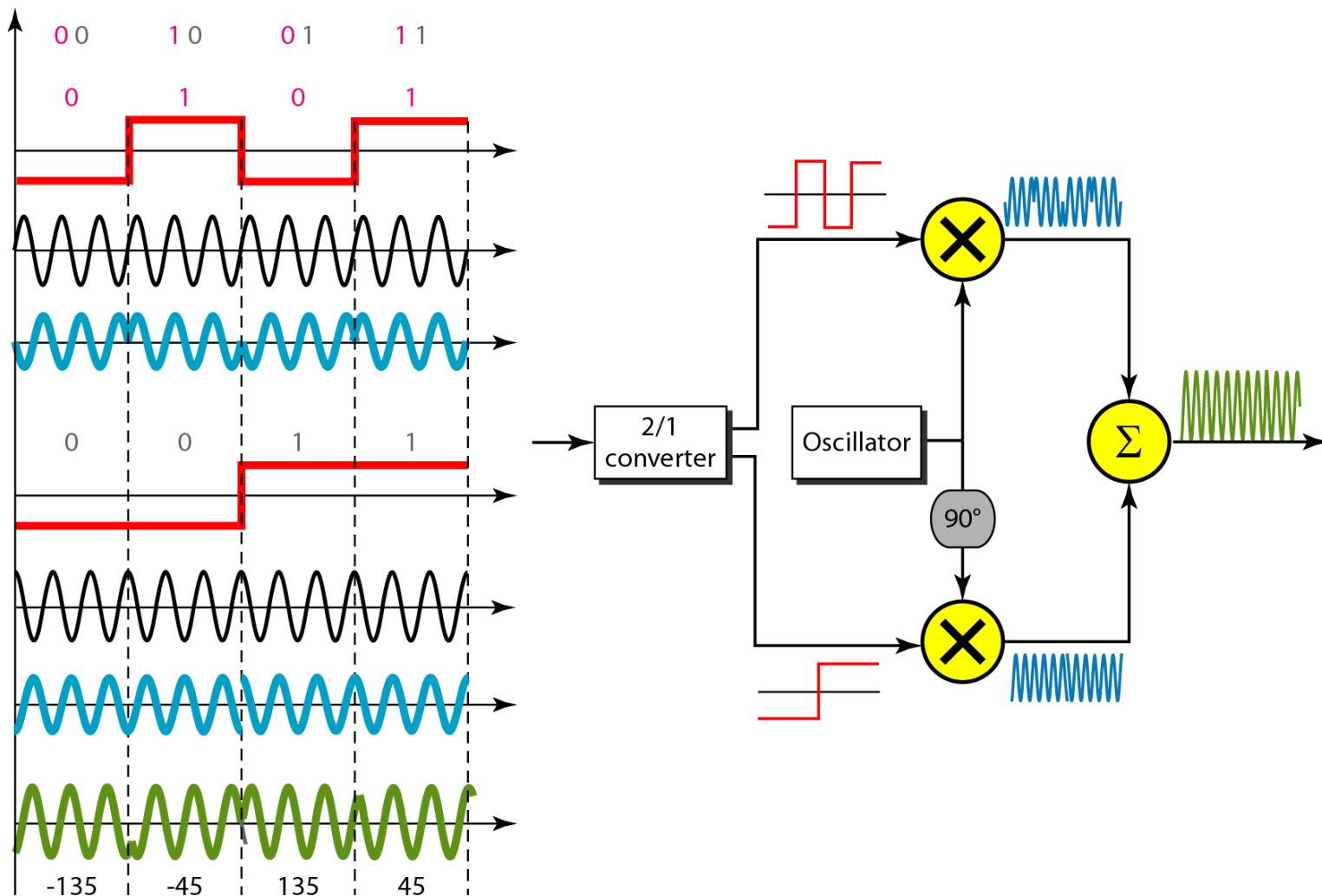
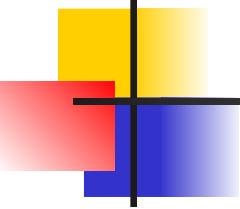


Figure 5.11 QPSK and its implementation





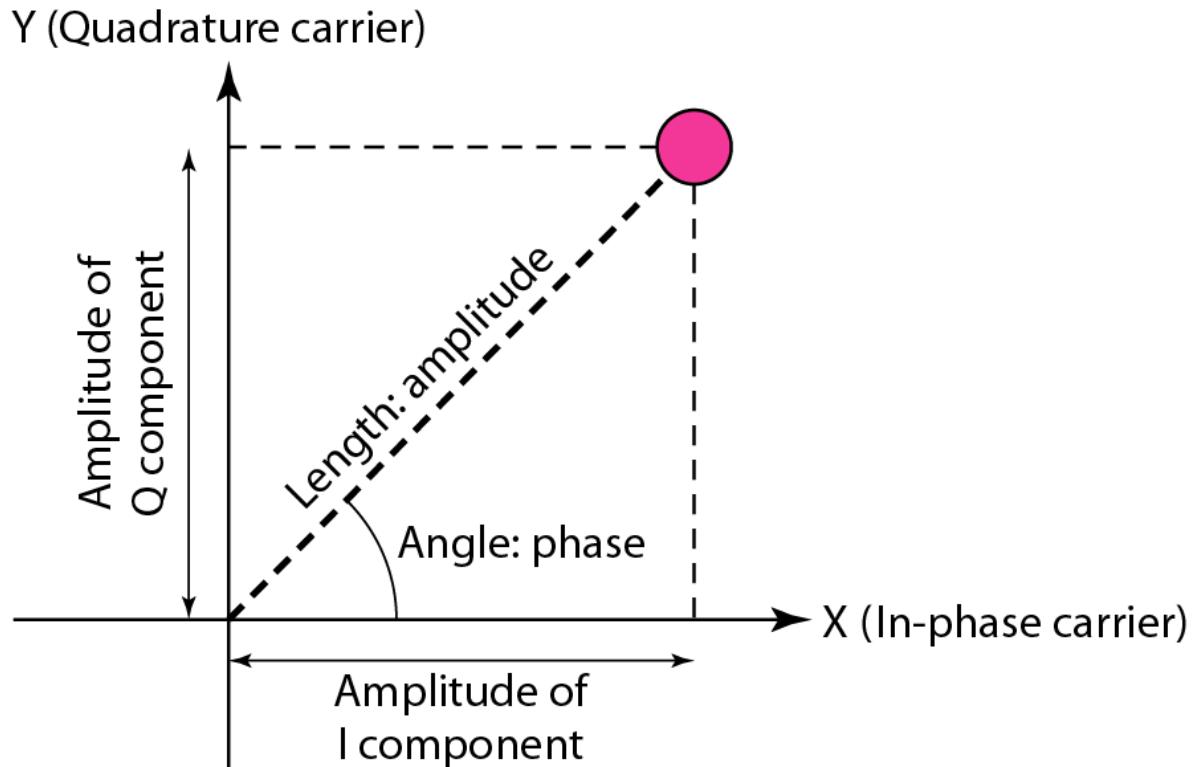
Example 5.7

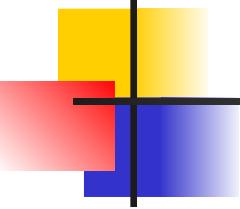
Find the bandwidth for a signal transmitting at 12 Mbps for QPSK. The value of $d = 0$.

Solution

For QPSK, 2 bits is carried by one signal element. This means that $r = 2$. So the signal rate (baud rate) is $S = N \times (1/r) = 6 \text{ Mbaud}$. With a value of $d = 0$, we have $B = S = 6 \text{ MHz}$.

Figure 5.12 Concept of a constellation diagram





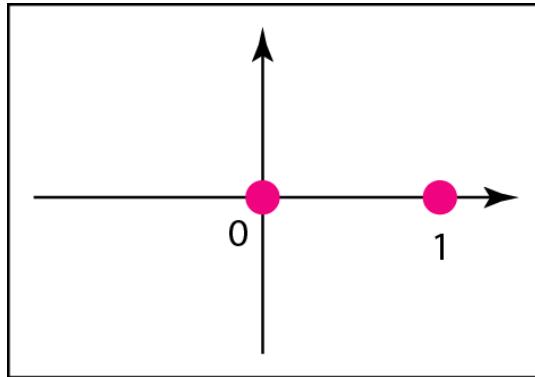
Example 5.8

Show the constellation diagrams for an ASK (OOK), BPSK, and QPSK signals.

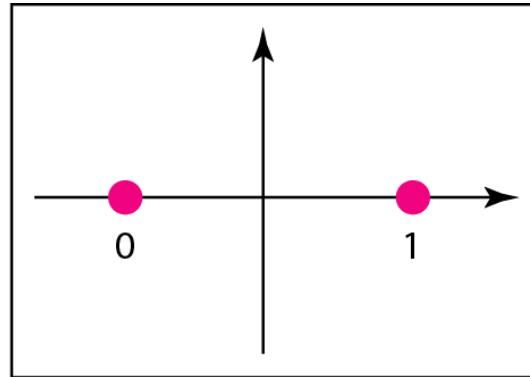
Solution

Figure 5.13 shows the three constellation diagrams.

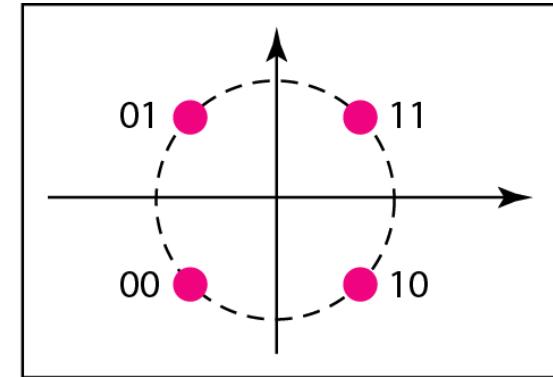
Figure 5.13 *Three constellation diagrams*



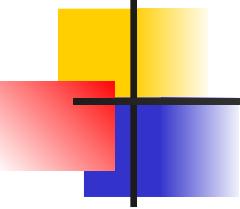
a. ASK (OOK)



b. BPSK



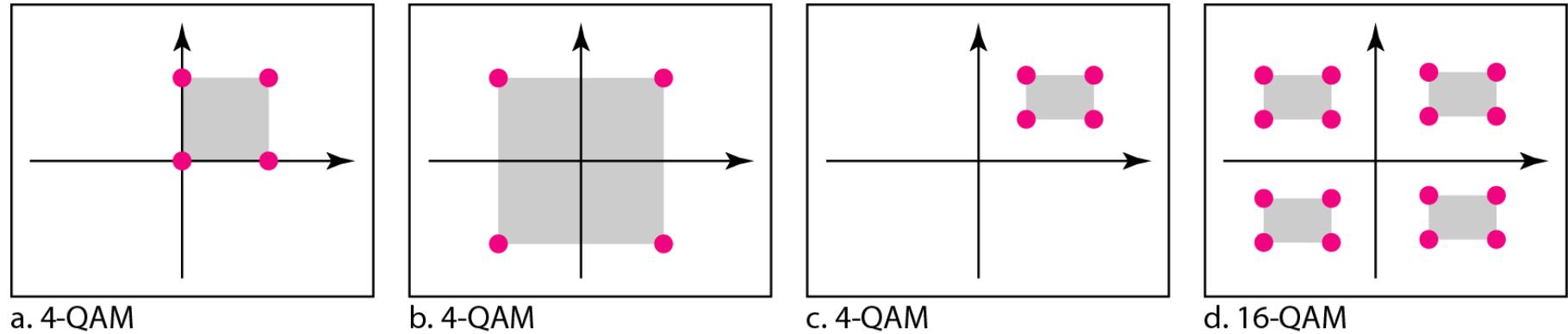
c. QPSK



Note

Quadrature amplitude modulation is a combination of ASK and PSK.

Figure 5.14 *Constellation diagrams for some QAMs*



5-2 ANALOG AND DIGITAL

Analog-to-analog conversion is the representation of analog information by an analog signal. One may ask why we need to modulate an analog signal; it is already analog. Modulation is needed if the medium is bandpass in nature or if only a bandpass channel is available to us.

Topics discussed in this section:

Amplitude Modulation
Frequency Modulation
Phase Modulation

Figure 5.15 *Types of analog-to-analog modulation*

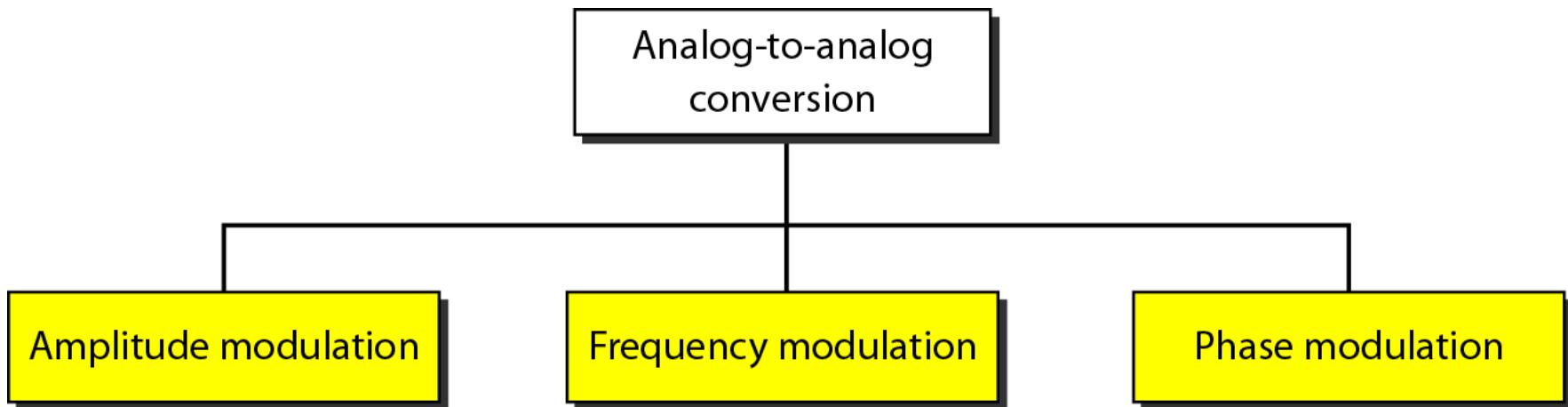
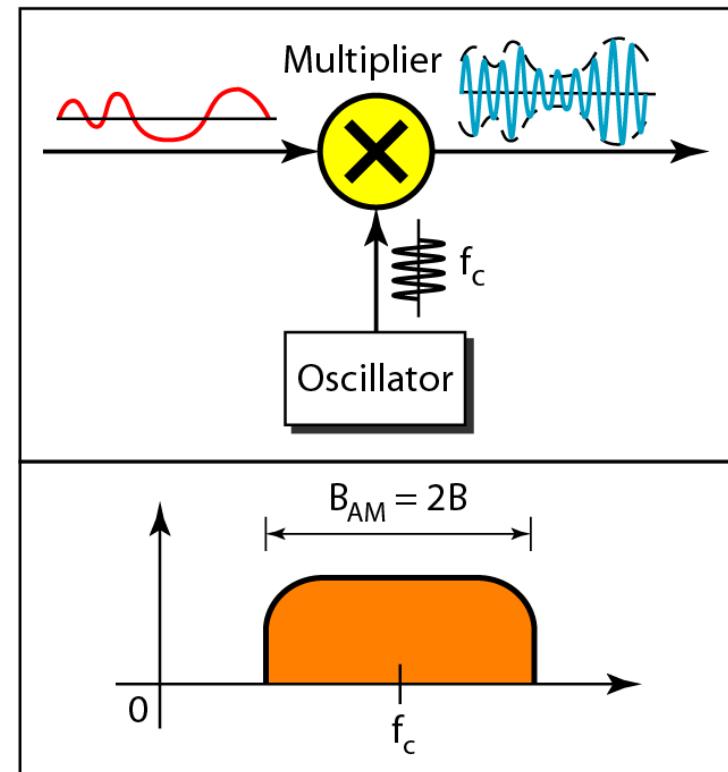
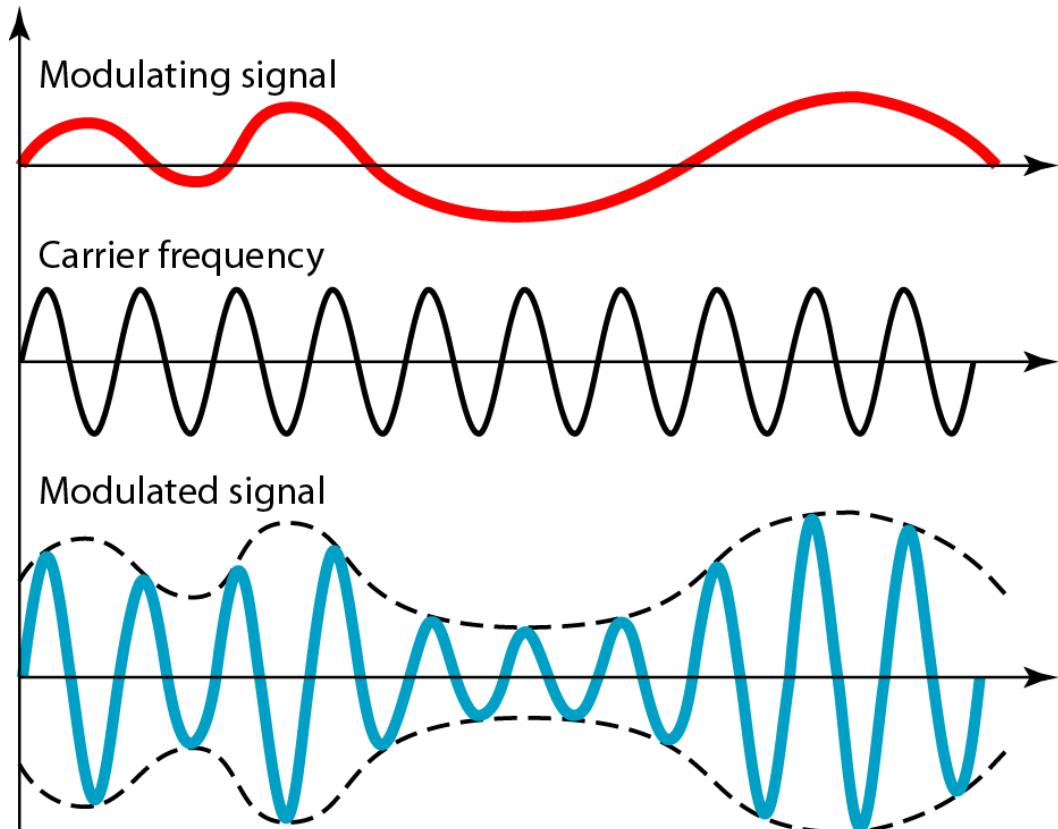
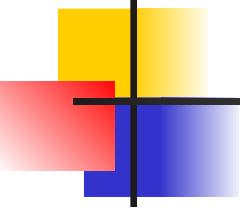


Figure 5.16 Amplitude modulation

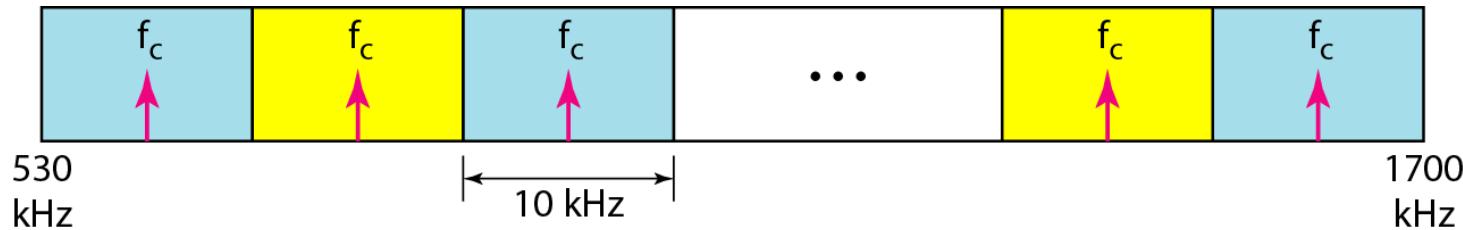


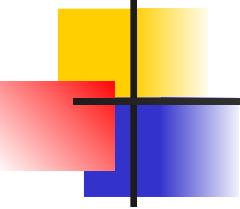


Note

**The total bandwidth required for AM
can be determined
from the bandwidth of the audio
signal: $B_{AM} = 2B.$**

Figure 5.17 AM band allocation





Note

The total bandwidth required for FM can be determined from the bandwidth of the audio signal: $B_{FM} = 2(1 + \beta)B$.

Figure 5.18 Frequency modulation

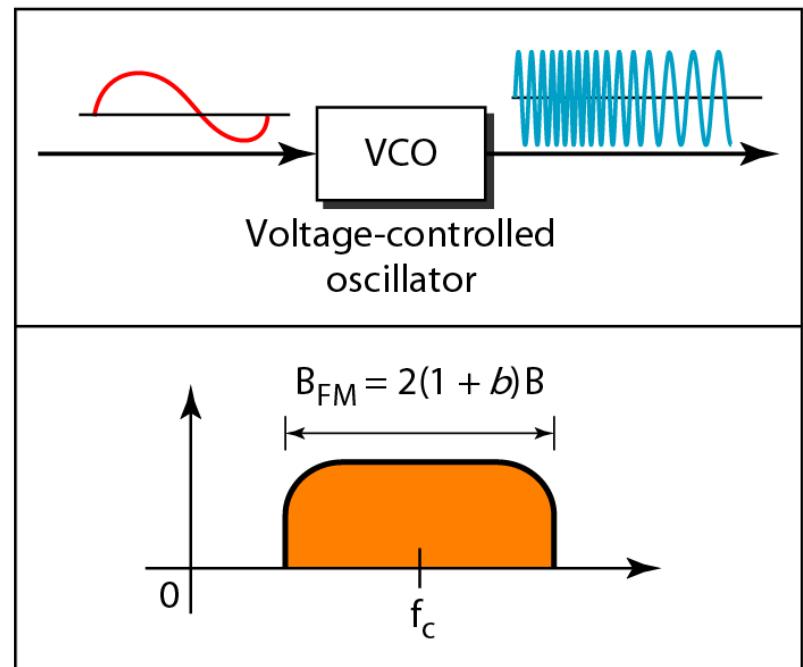
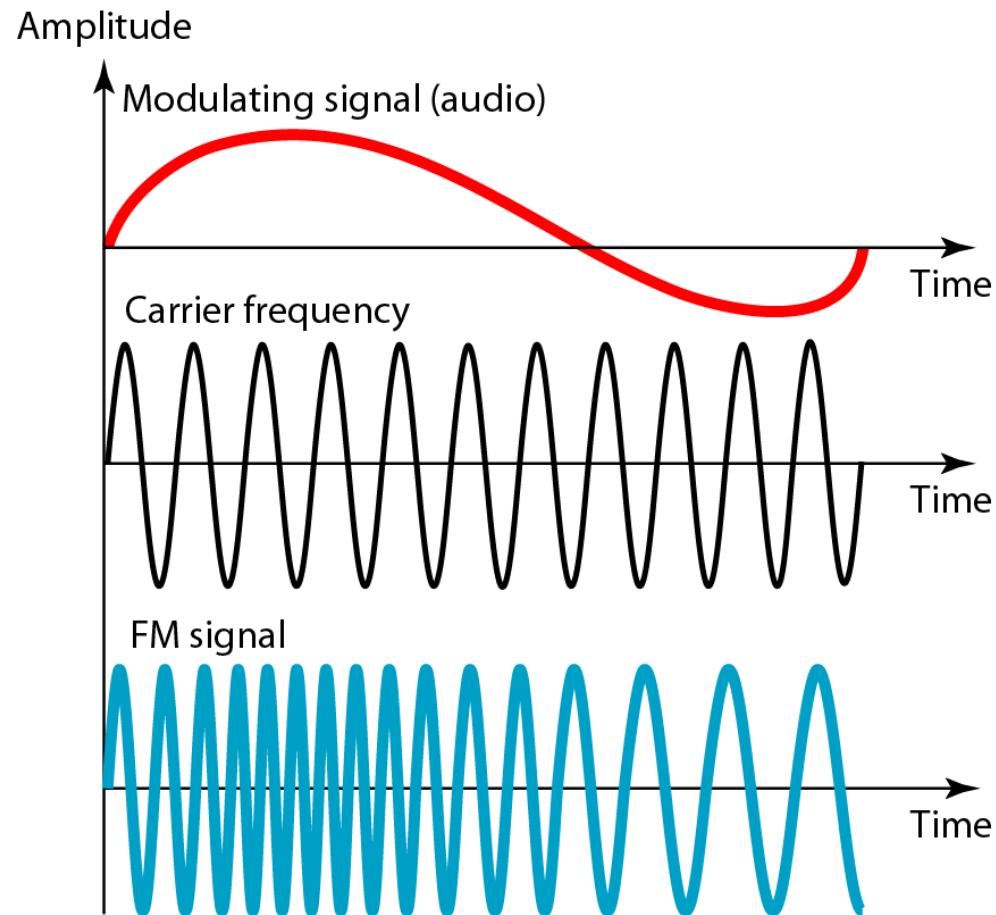


Figure 5.19 FM band allocation

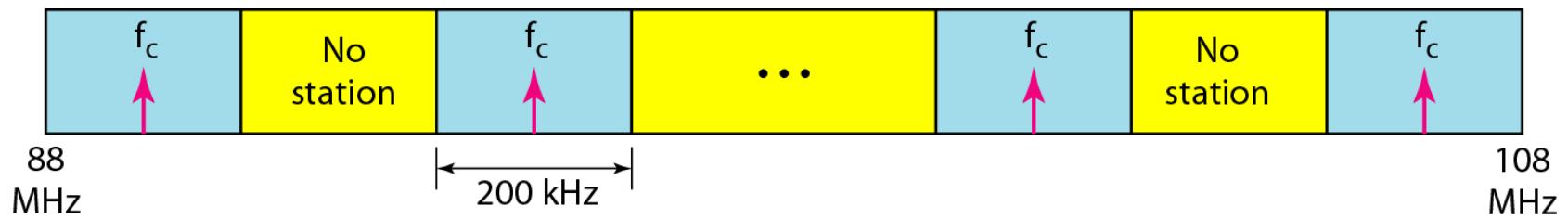
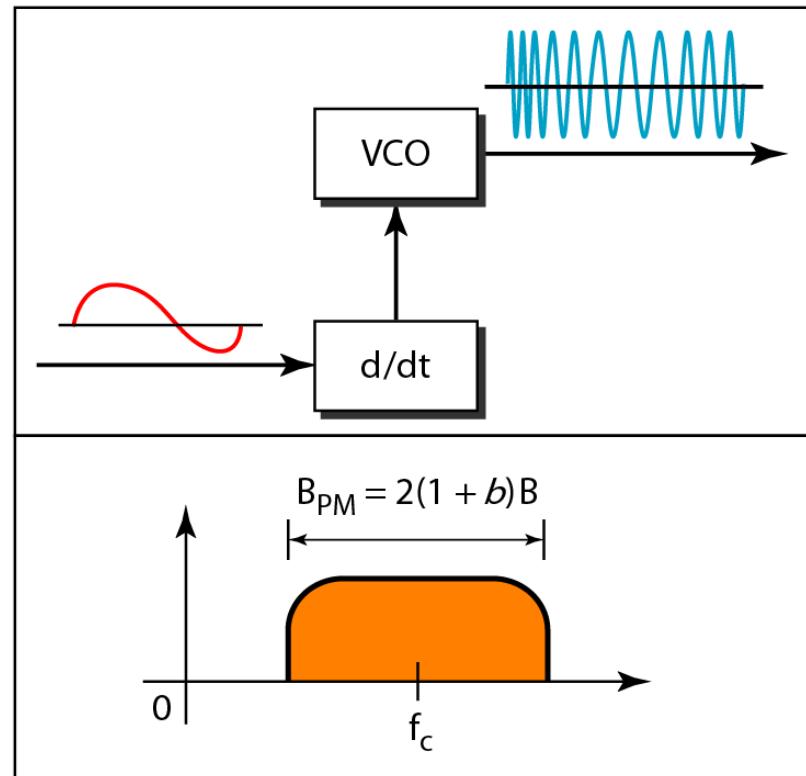
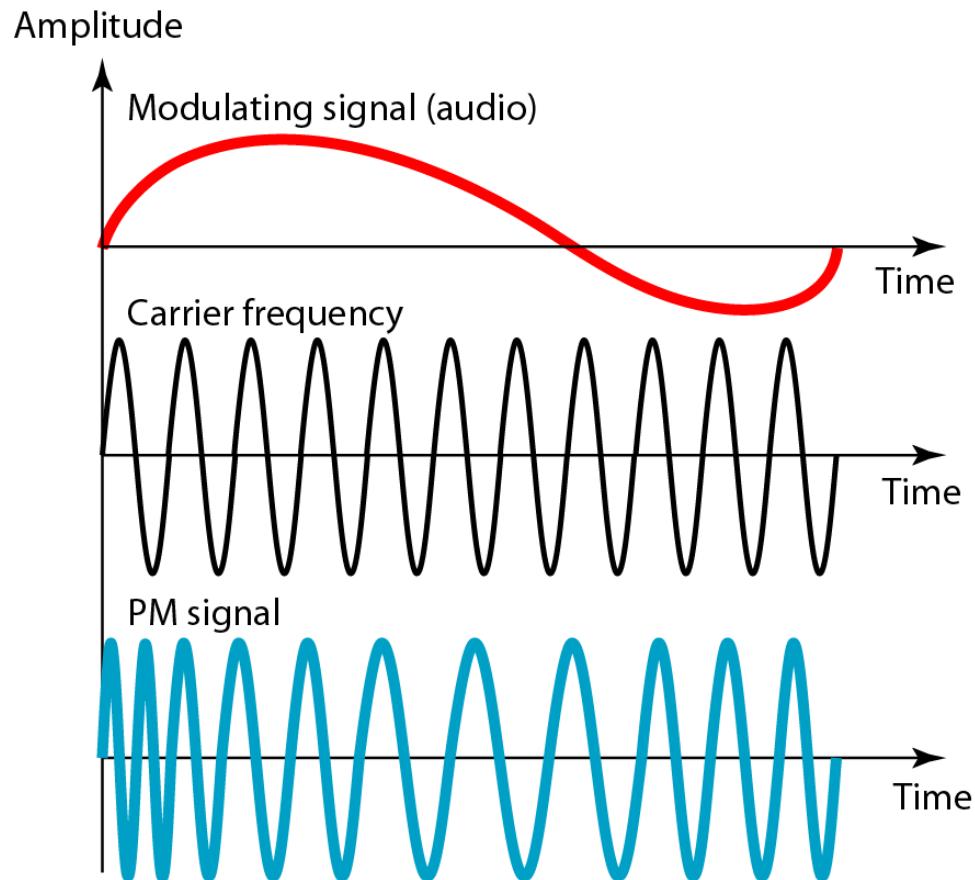
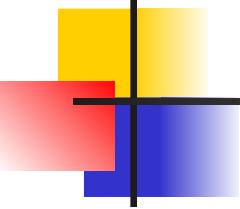


Figure 5.20 Phase modulation





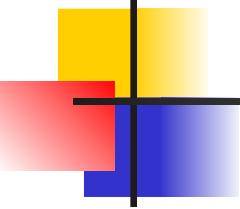
Note

The total bandwidth required for PM can be determined from the bandwidth and maximum amplitude of the modulating signal:

$$B_{PM} = 2(1 + \beta)B.$$

Chapter 6

Bandwidth Utilization: Multiplexing and Spreading



Note

Bandwidth utilization is the wise use of available bandwidth to achieve specific goals.

Efficiency can be achieved by multiplexing; privacy and anti-jamming can be achieved by spreading.

6-1 MULTIPLEXING

Whenever the bandwidth of a medium linking two devices is greater than the bandwidth needs of the devices, the link can be shared. Multiplexing is the set of techniques that allows the simultaneous transmission of multiple signals across a single data link. As data and telecommunications use increases, so does traffic.

Topics discussed in this section:

Frequency-Division Multiplexing

Wavelength-Division Multiplexing

Synchronous Time-Division Multiplexing

Statistical Time-Division Multiplexing

Figure 6.1 *Dividing a link into channels*

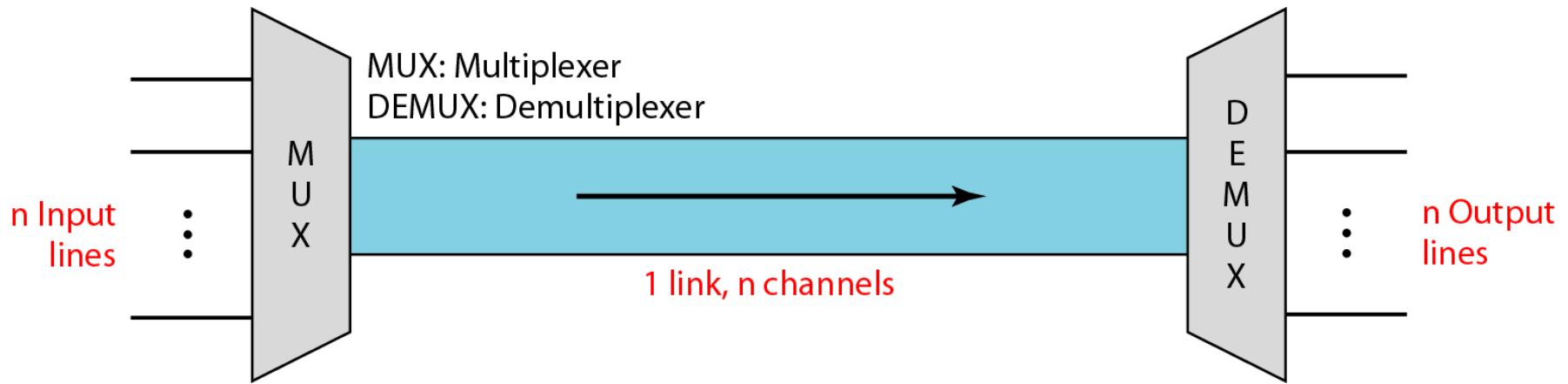


Figure 6.2 *Categories of multiplexing*

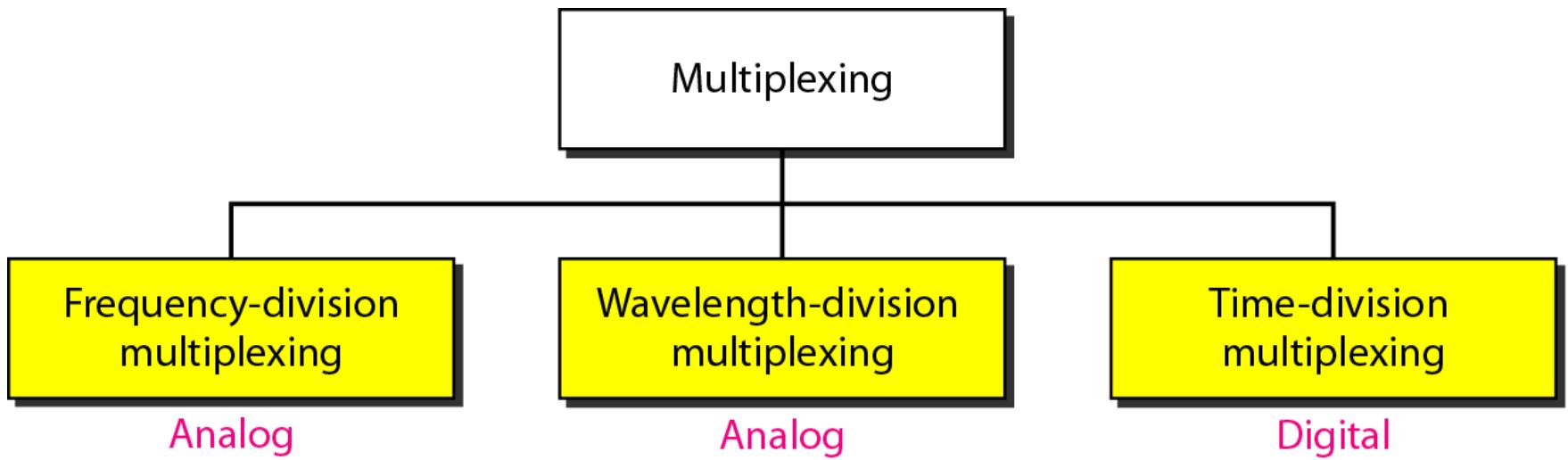
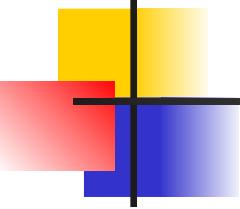


Figure 6.3 Frequency-division multiplexing





Note

**FDM is an analog multiplexing technique
that combines analog signals.**

Figure 6.4 FDM process

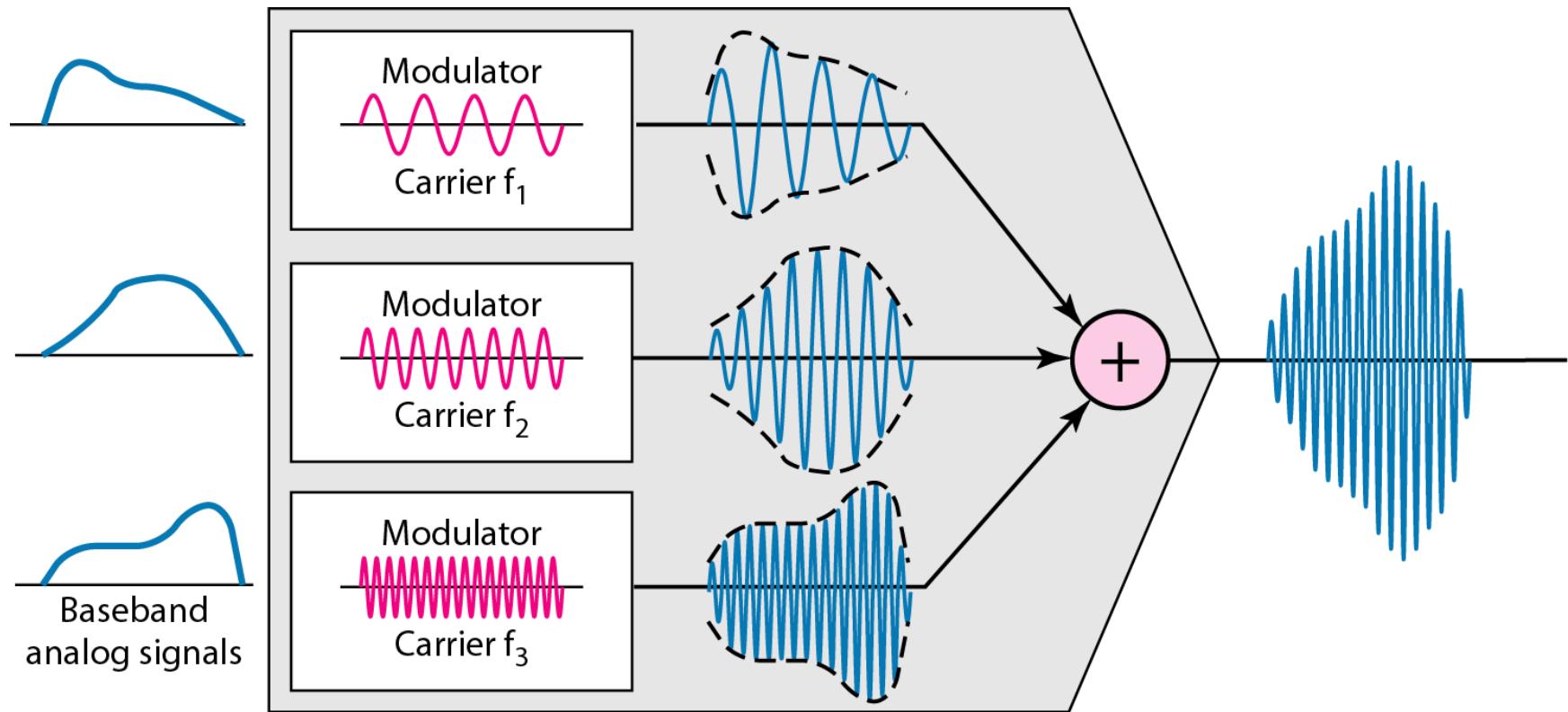
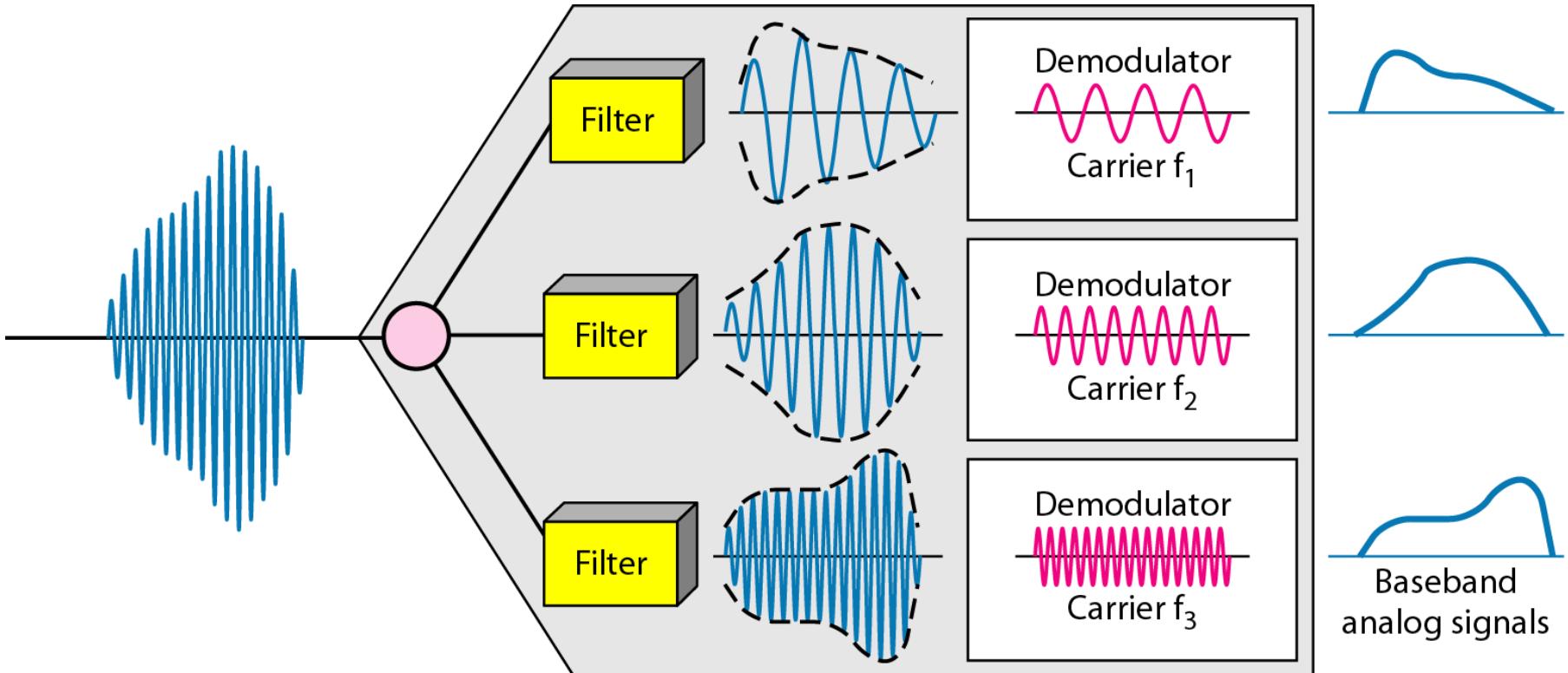


Figure 6.5 FDM demultiplexing example



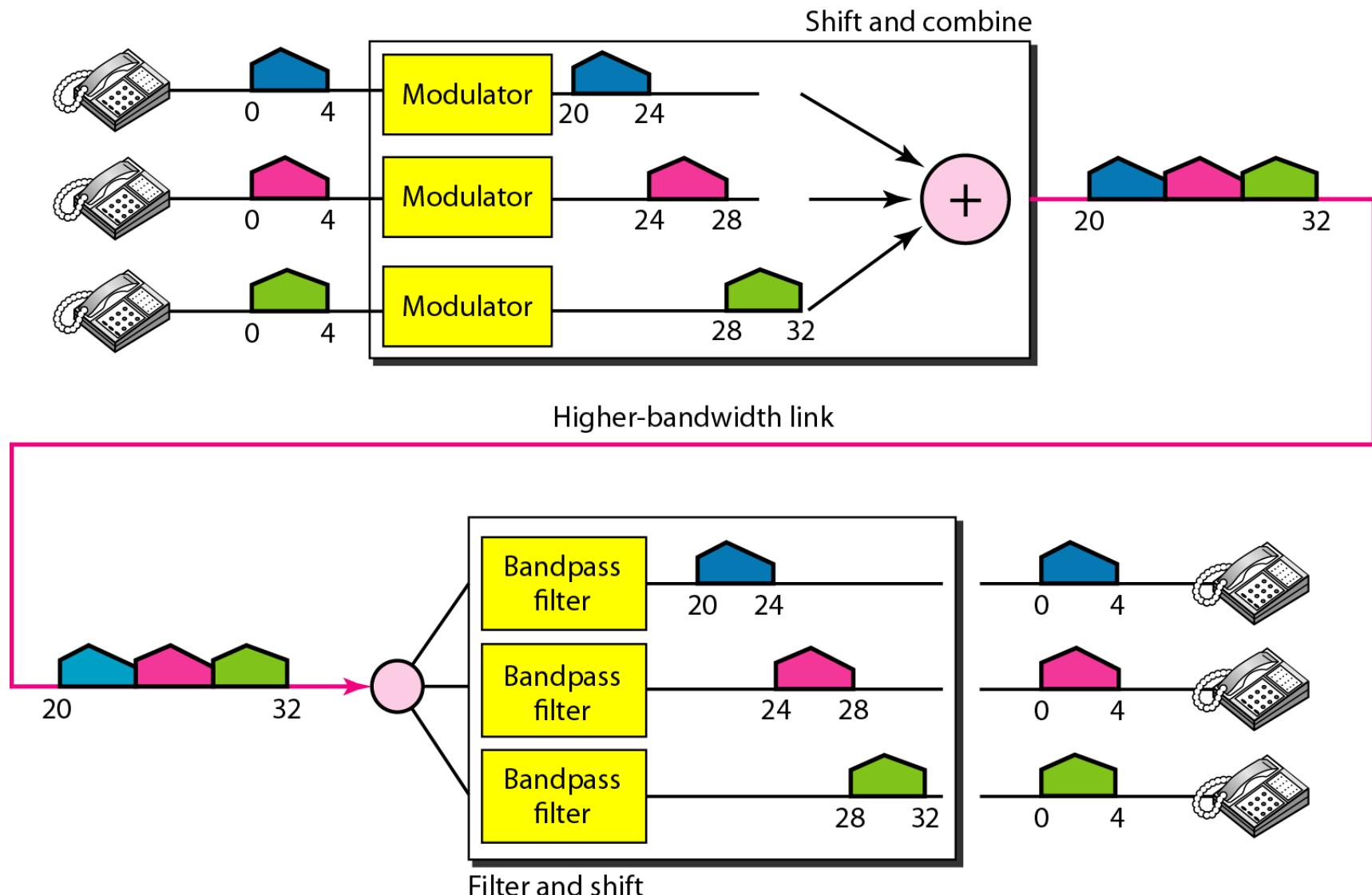
Example 6.1

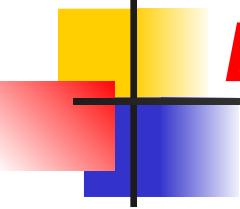
Assume that a voice channel occupies a bandwidth of 4 kHz. We need to combine three voice channels into a link with a bandwidth of 12 kHz, from 20 to 32 kHz. Show the configuration, using the frequency domain. Assume there are no guard bands.

Solution

We shift (modulate) each of the three voice channels to a different bandwidth, as shown in Figure 6.6. We use the 20- to 24-kHz bandwidth for the first channel, the 24- to 28-kHz bandwidth for the second channel, and the 28- to 32-kHz bandwidth for the third one. Then we combine them as shown in Figure 6.6.

Figure 6.6 Example 6.1





Example 6.2

Five channels, each with a 100-kHz bandwidth, are to be multiplexed together. What is the minimum bandwidth of the link if there is a need for a guard band of 10 kHz between the channels to prevent interference?

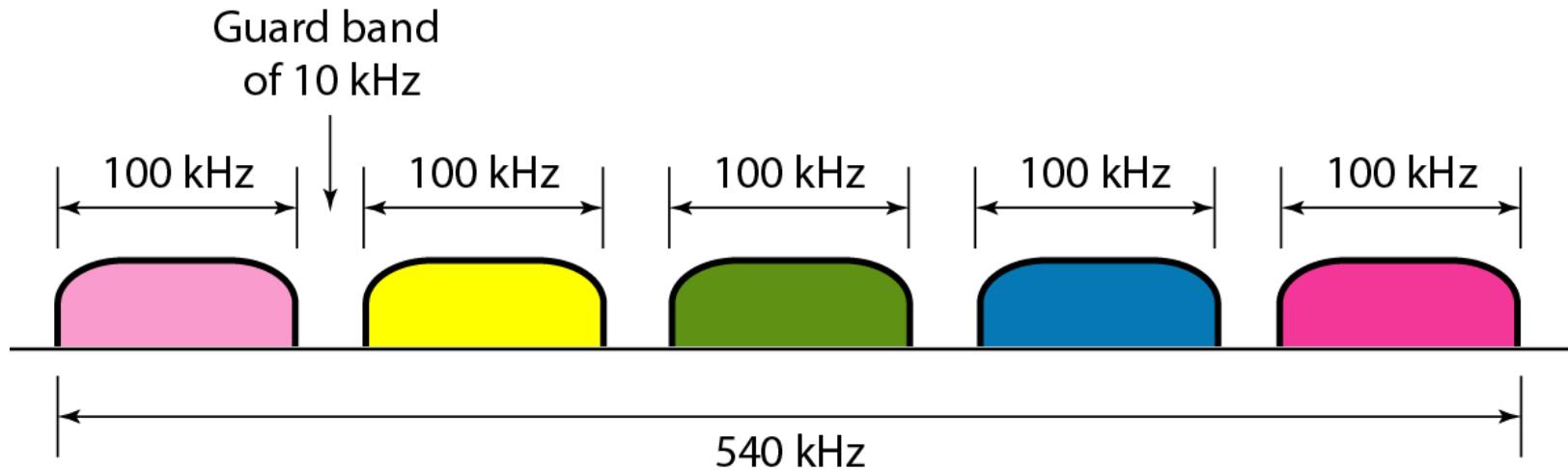
Solution

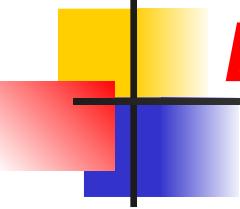
For five channels, we need at least four guard bands. This means that the required bandwidth is at least

$$5 \times 100 + 4 \times 10 = 540 \text{ kHz},$$

as shown in Figure 6.7.

Figure 6.7 Example 6.2





Example 6.3

Four data channels (digital), each transmitting at 1 Mbps, use a satellite channel of 1 MHz. Design an appropriate configuration, using FDM.

Solution

The satellite channel is analog. We divide it into four channels, each channel having a 250-kHz bandwidth. Each digital channel of 1 Mbps is modulated such that each 4 bits is modulated to 1 Hz. One solution is 16-QAM modulation. Figure 6.8 shows one possible configuration.

Figure 6.8 Example 6.3

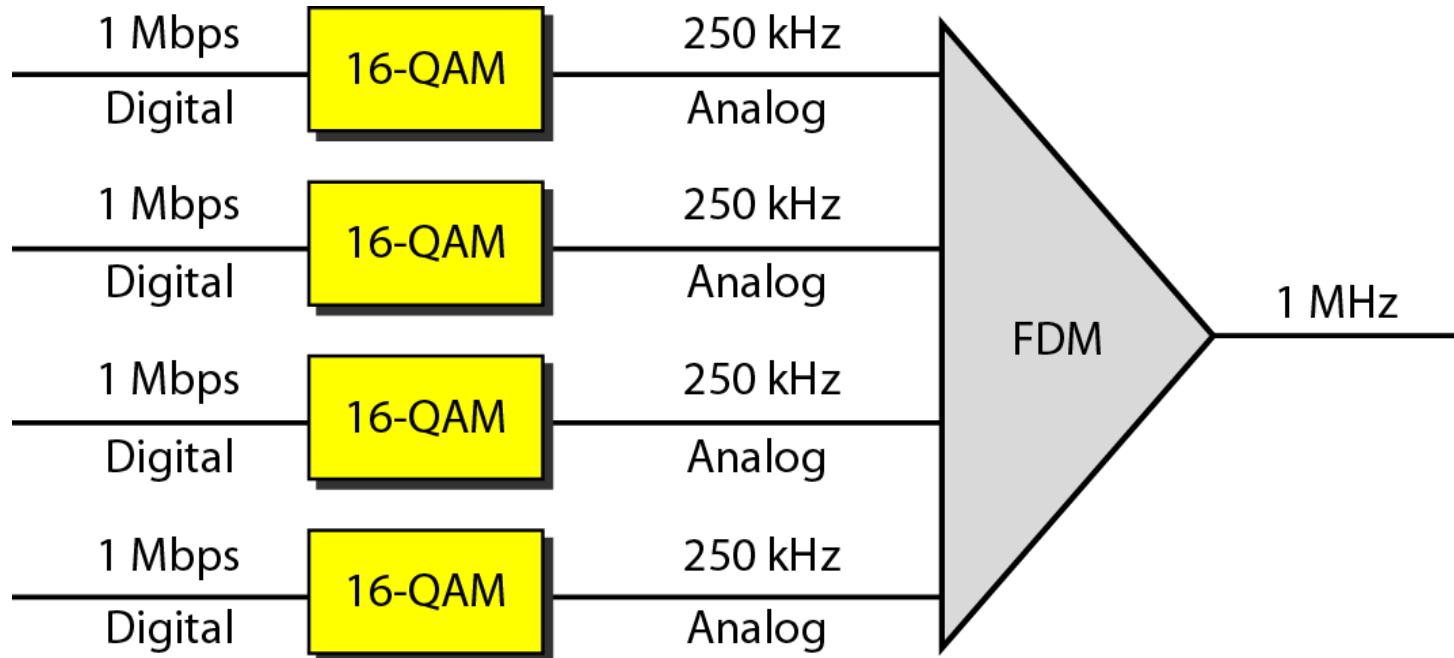
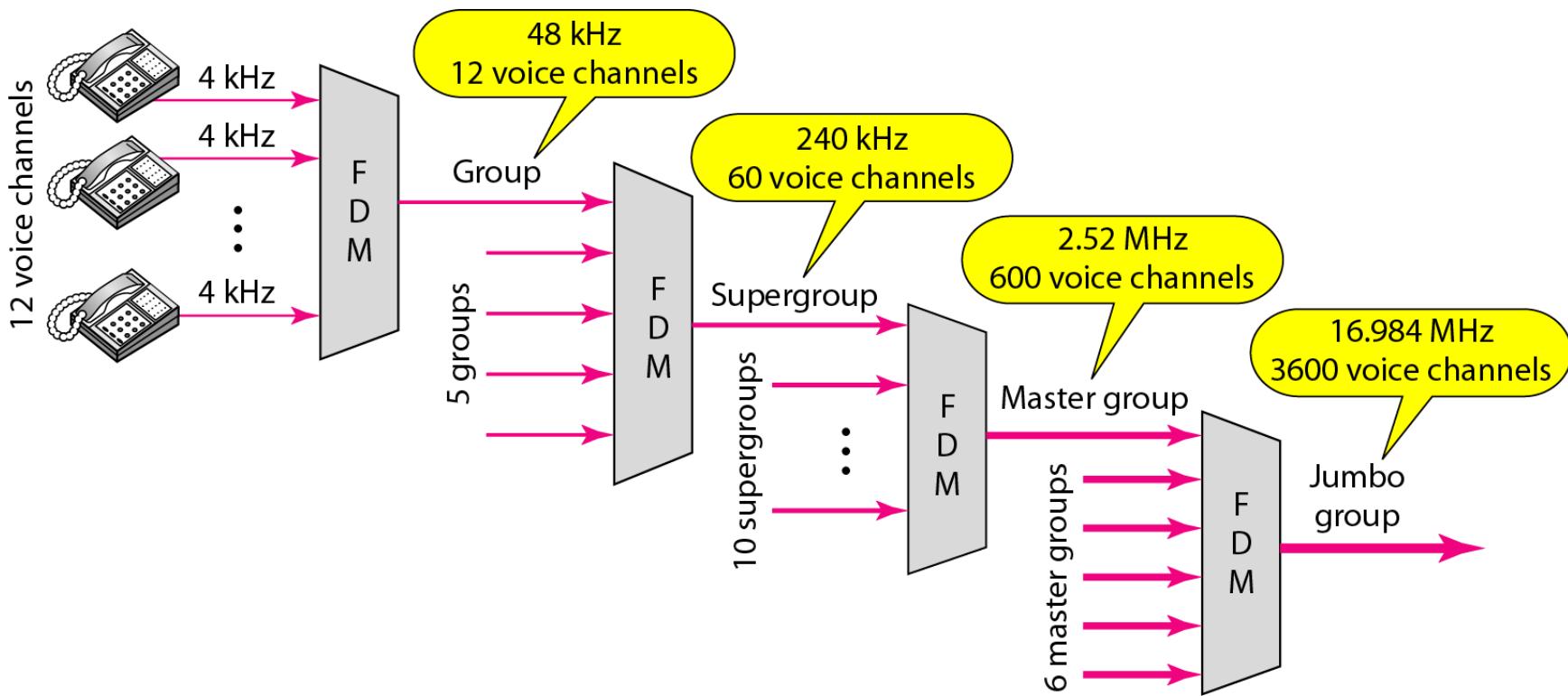
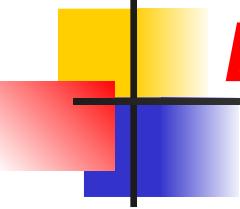


Figure 6.9 Analog hierarchy





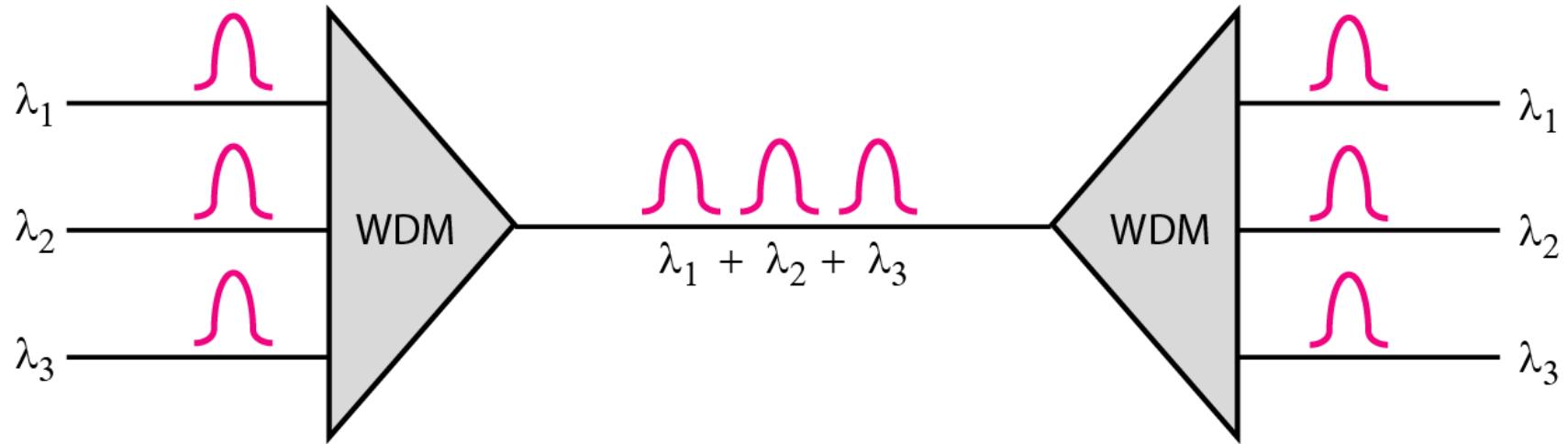
Example 6.4

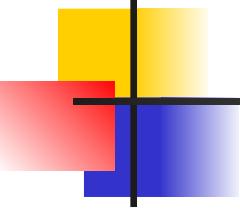
The Advanced Mobile Phone System (AMPS) uses two bands. The first band of 824 to 849 MHz is used for sending, and 869 to 894 MHz is used for receiving. Each user has a bandwidth of 30 kHz in each direction. How many people can use their cellular phones simultaneously?

Solution

Each band is 25 MHz. If we divide 25 MHz by 30 kHz, we get 833.33. In reality, the band is divided into 832 channels. Of these, 42 channels are used for control, which means only 790 channels are available for cellular phone users.

Figure 6.10 *Wavelength-division multiplexing*





Note

WDM is an analog multiplexing technique to combine optical signals.

Figure 6.11 Prisms in wavelength-division multiplexing and demultiplexing

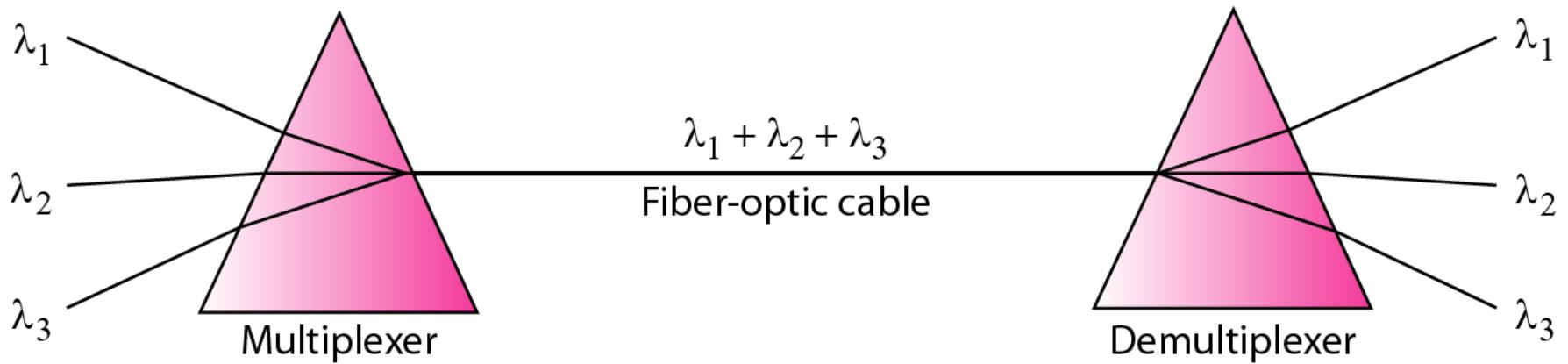
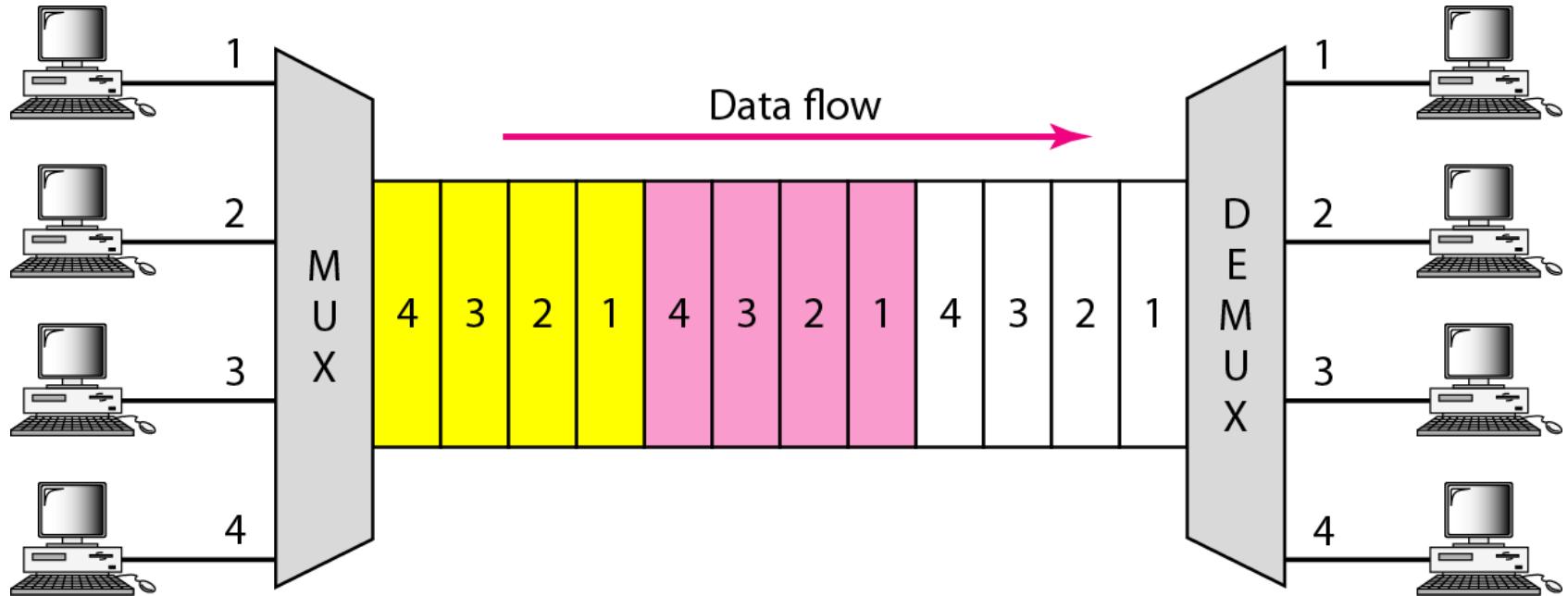
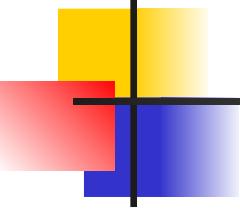


Figure 6.12 TDM

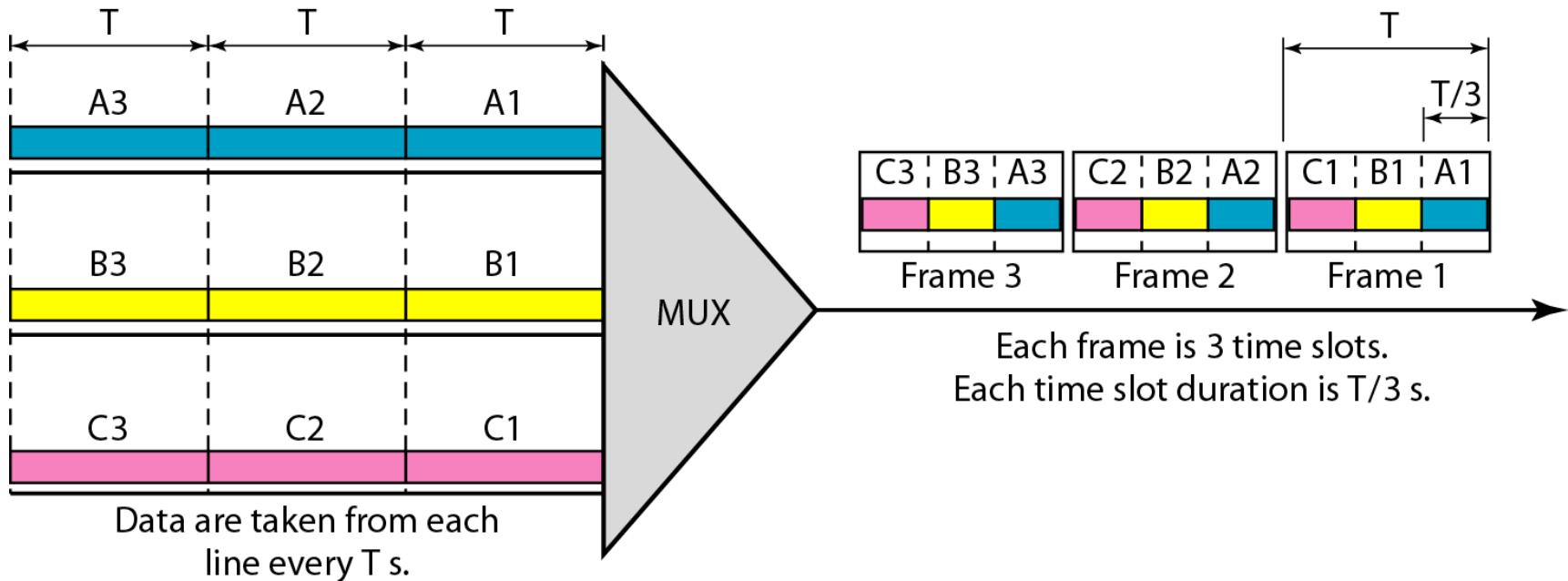


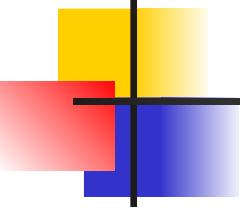


Note

**TDM is a digital multiplexing technique
for combining several low-rate
channels into one high-rate one.**

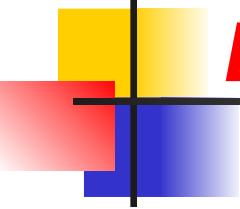
Figure 6.13 Synchronous time-division multiplexing





Note

In synchronous TDM, the data rate of the link is n times faster, and the unit duration is n times shorter.



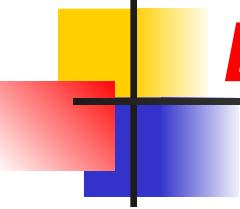
Example 6.5

In Figure 6.13, the data rate for each input connection is 3 kbps. If 1 bit at a time is multiplexed (a unit is 1 bit), what is the duration of (a) each input slot, (b) each output slot, and (c) each frame?

Solution

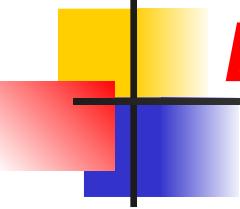
We can answer the questions as follows:

- a. *The data rate of each input connection is 1 kbps. This means that the bit duration is $1/1000$ s or 1 ms. The duration of the input time slot is 1 ms (same as bit duration).*



Example 6.5 (continued)

- b.** *The duration of each output time slot is one-third of the input time slot. This means that the duration of the output time slot is $1/3$ ms.*
- c.** *Each frame carries three output time slots. So the duration of a frame is $3 \times 1/3$ ms, or 1 ms. The duration of a frame is the same as the duration of an input unit.*



Example 6.6

Figure 6.14 shows synchronous TDM with a data stream for each input and one data stream for the output. The unit of data is 1 bit. Find (a) the input bit duration, (b) the output bit duration, (c) the output bit rate, and (d) the output frame rate.

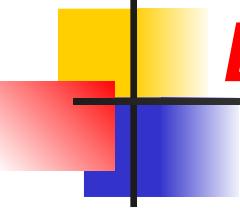
Solution

We can answer the questions as follows:

a. *The input bit duration is the inverse of the bit rate:*

$$1/1 \text{ Mbps} = 1 \mu\text{s}.$$

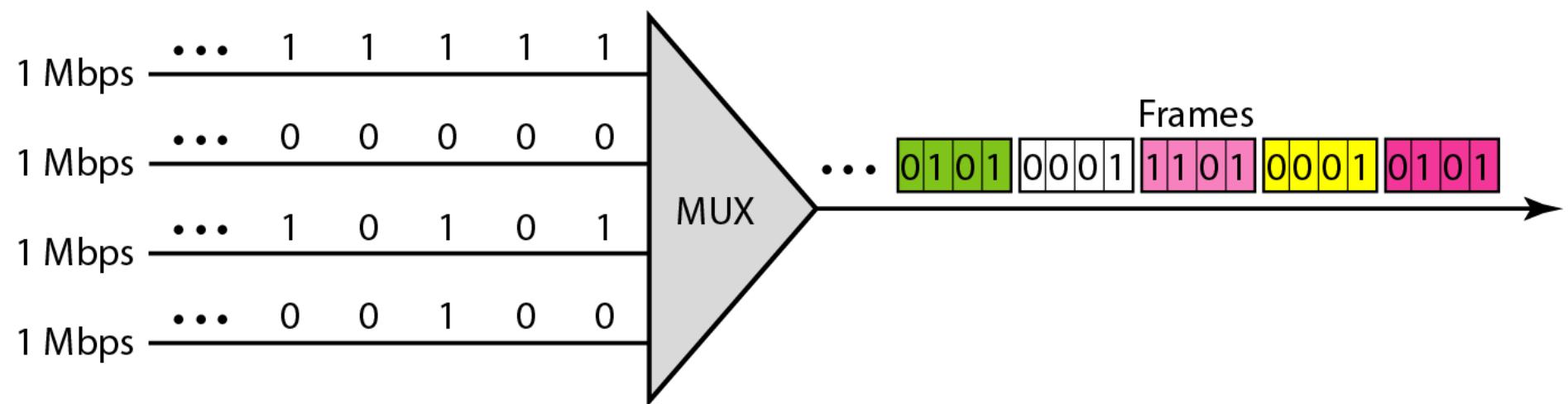
b. *The output bit duration is one-fourth of the input bit duration, or $\frac{1}{4} \mu\text{s}$.*

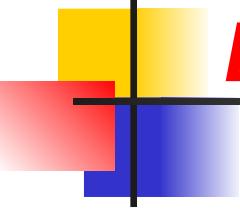


Example 6.6 (continued)

- c. *The output bit rate is the inverse of the output bit duration or $1/(4\mu s)$ or 4 Mbps. This can also be deduced from the fact that the output rate is 4 times as fast as any input rate; so the output rate = $4 \times 1 \text{ Mbps} = 4 \text{ Mbps}$.*
- d. *The frame rate is always the same as any input rate. So the frame rate is 1,000,000 frames per second. Because we are sending 4 bits in each frame, we can verify the result of the previous question by multiplying the frame rate by the number of bits per frame.*

Figure 6.14 Example 6.6





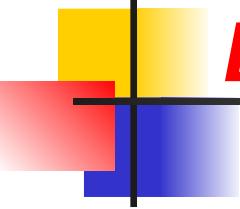
Example 6.7

Four 1-kbps connections are multiplexed together. A unit is 1 bit. Find (a) the duration of 1 bit before multiplexing, (b) the transmission rate of the link, (c) the duration of a time slot, and (d) the duration of a frame.

Solution

We can answer the questions as follows:

- a. *The duration of 1 bit before multiplexing is $1 / 1 \text{ kbps}$, or 0.001 s (1 ms).*
- b. *The rate of the link is 4 times the rate of a connection, or 4 kbps .*

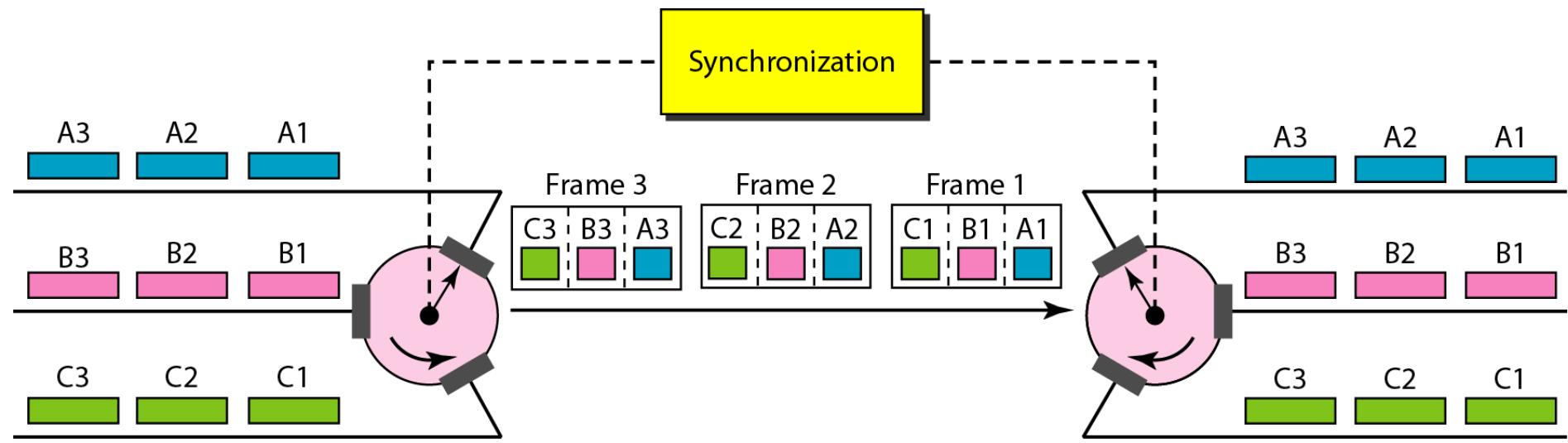


Example 6.7 (continued)

- c. The duration of each time slot is one-fourth of the duration of each bit before multiplexing, or $1/4$ ms or $250 \mu\text{s}$. Note that we can also calculate this from the data rate of the link, 4 kbps . The bit duration is the inverse of the data rate, or $1/4 \text{ kbps}$ or $250 \mu\text{s}$.

- d. The duration of a frame is always the same as the duration of a unit before multiplexing, or 1 ms . We can also calculate this in another way. Each frame in this case has four time slots. So the duration of a frame is 4 times $250 \mu\text{s}$, or 1 ms .

Figure 6.15 Interleaving



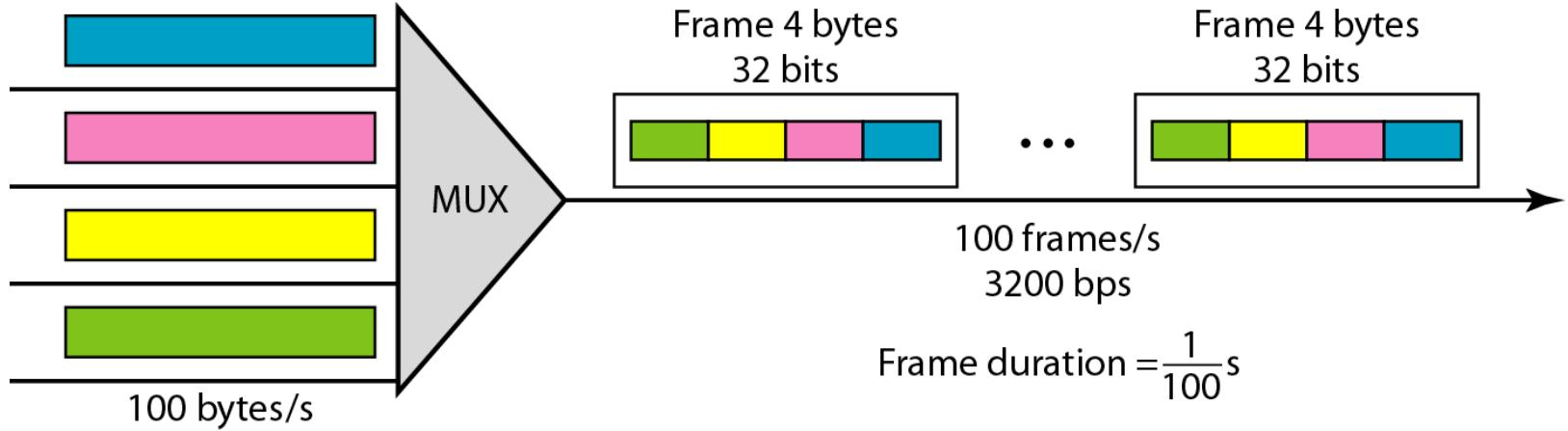
Example 6.8

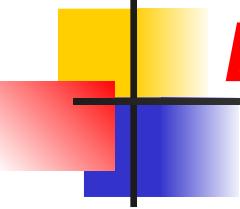
Four channels are multiplexed using TDM. If each channel sends 100 bytes /s and we multiplex 1 byte per channel, show the frame traveling on the link, the size of the frame, the duration of a frame, the frame rate, and the bit rate for the link.

Solution

The multiplexer is shown in Figure 6.16. Each frame carries 1 byte from each channel; the size of each frame, therefore, is 4 bytes, or 32 bits. Because each channel is sending 100 bytes/s and a frame carries 1 byte from each channel, the frame rate must be 100 frames per second. The bit rate is 100×32 , or 3200 bps.

Figure 6.16 Example 6.8





Example 6.9

A multiplexer combines four 100-kbps channels using a time slot of 2 bits. Show the output with four arbitrary inputs. What is the frame rate? What is the frame duration? What is the bit rate? What is the bit duration?

Solution

Figure 6.17 shows the output for four arbitrary inputs. The link carries 50,000 frames per second. The frame duration is therefore $1/50,000$ s or $20 \mu\text{s}$. The frame rate is 50,000 frames per second, and each frame carries 8 bits; the bit rate is $50,000 \times 8 = 400,000$ bits or 400 kbps. The bit duration is $1/400,000$ s, or $2.5 \mu\text{s}$.

Figure 6.17 Example 6.9

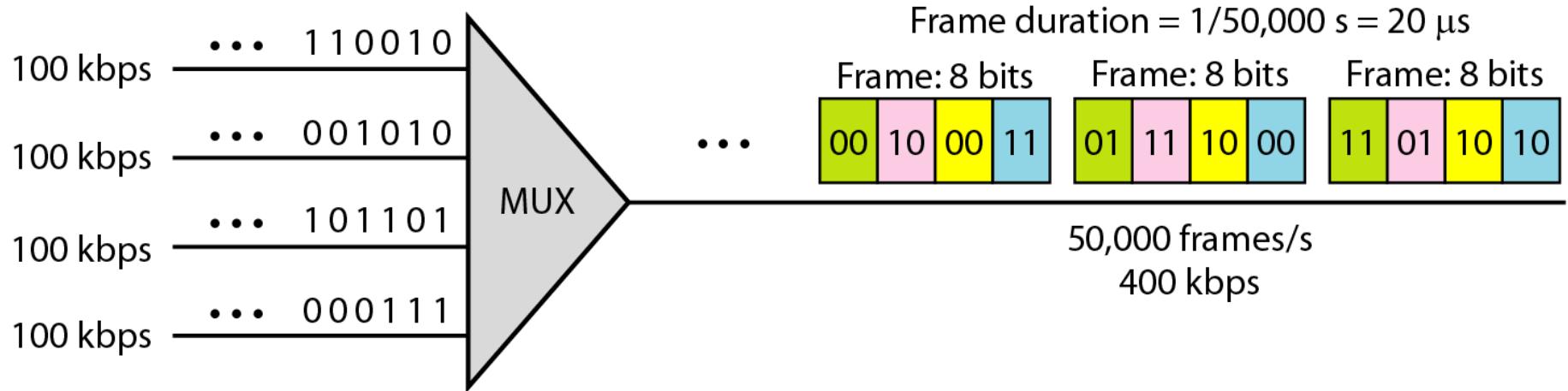


Figure 6.18 *Empty slots*

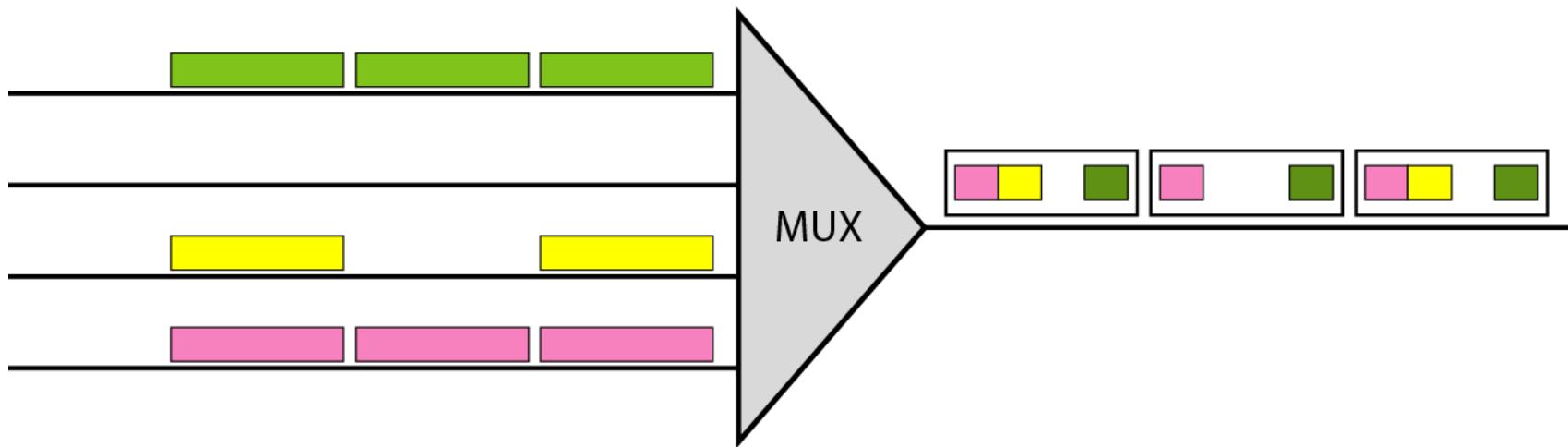


Figure 6.19 *Multilevel multiplexing*

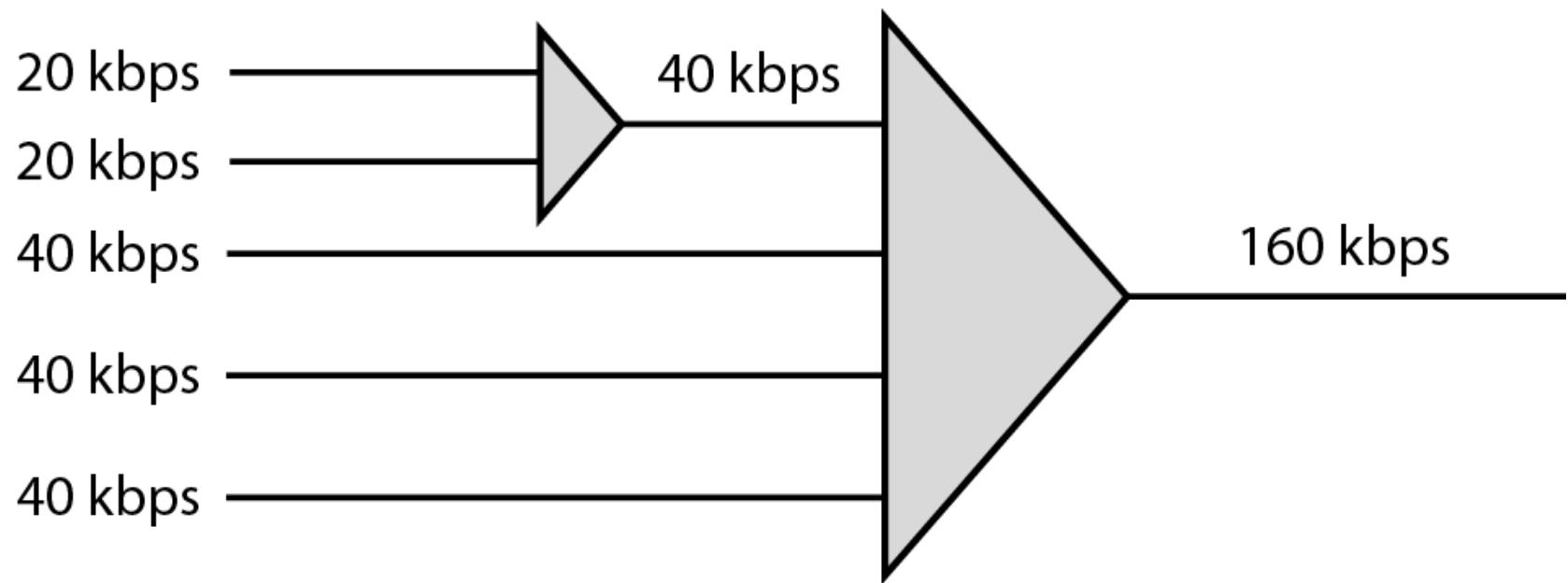


Figure 6.20 *Multiple-slot multiplexing*

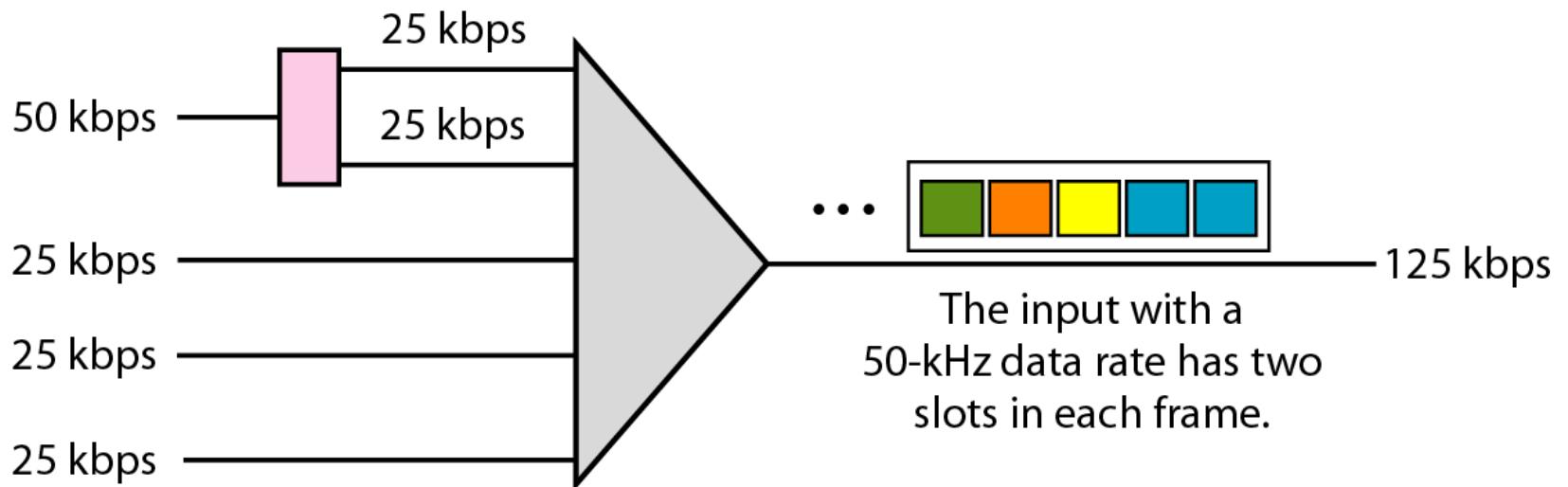


Figure 6.21 Pulse stuffing

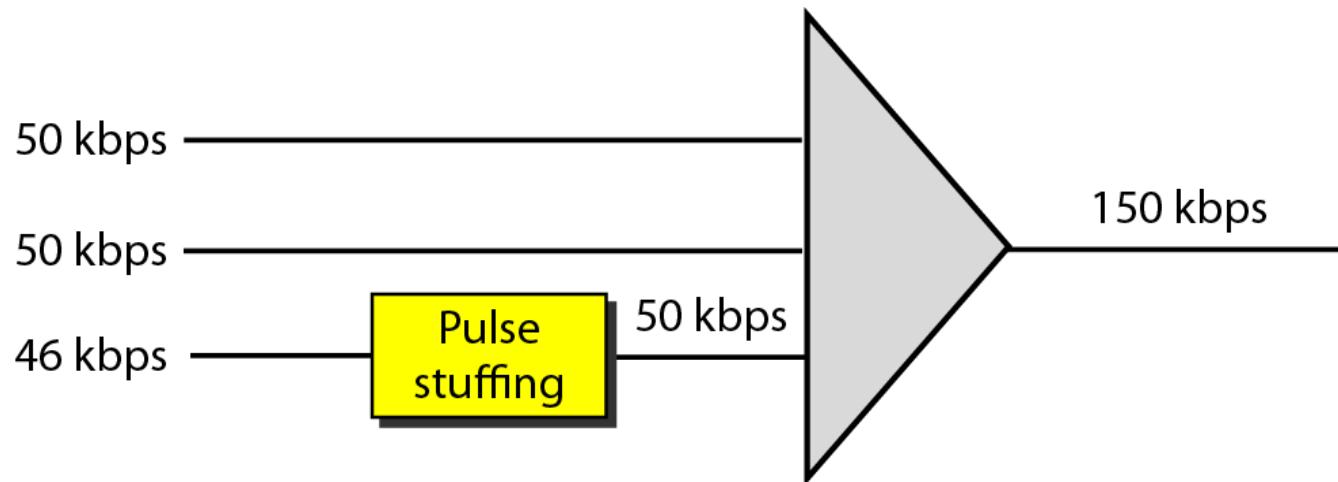
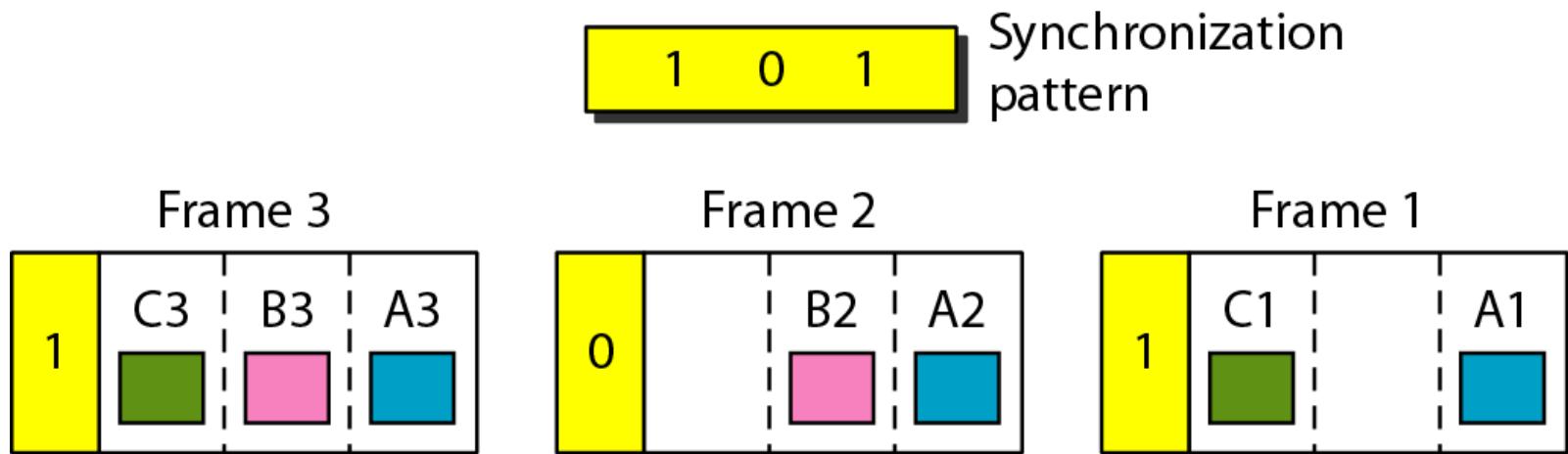
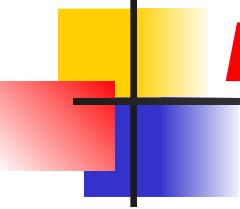


Figure 6.22 *Framing bits*





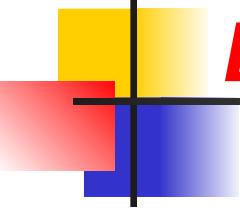
Example 6.10

We have four sources, each creating 250 characters per second. If the interleaved unit is a character and 1 synchronizing bit is added to each frame, find (a) the data rate of each source, (b) the duration of each character in each source, (c) the frame rate, (d) the duration of each frame, (e) the number of bits in each frame, and (f) the data rate of the link.

Solution

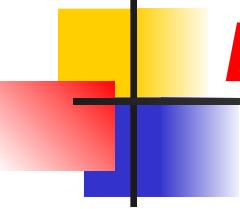
We can answer the questions as follows:

- a. The data rate of each source is $250 \times 8 = 2000 \text{ bps} = 2 \text{ kbps}$.



Example 6.10 (continued)

- b. *Each source sends 250 characters per second; therefore, the duration of a character is 1/250 s, or 4 ms.*
- c. *Each frame has one character from each source, which means the link needs to send 250 frames per second to keep the transmission rate of each source.*
- d. *The duration of each frame is 1/250 s, or 4 ms. Note that the duration of each frame is the same as the duration of each character coming from each source.*
- e. *Each frame carries 4 characters and 1 extra synchronizing bit. This means that each frame is $4 \times 8 + 1 = 33$ bits.*



Example 6.11

Two channels, one with a bit rate of 100 kbps and another with a bit rate of 200 kbps, are to be multiplexed. How this can be achieved? What is the frame rate? What is the frame duration? What is the bit rate of the link?

Solution

We can allocate one slot to the first channel and two slots to the second channel. Each frame carries 3 bits. The frame rate is 100,000 frames per second because it carries 1 bit from the first channel. The bit rate is 100,000 frames/s × 3 bits per frame, or 300 kbps.

Figure 6.23 *Digital hierarchy*

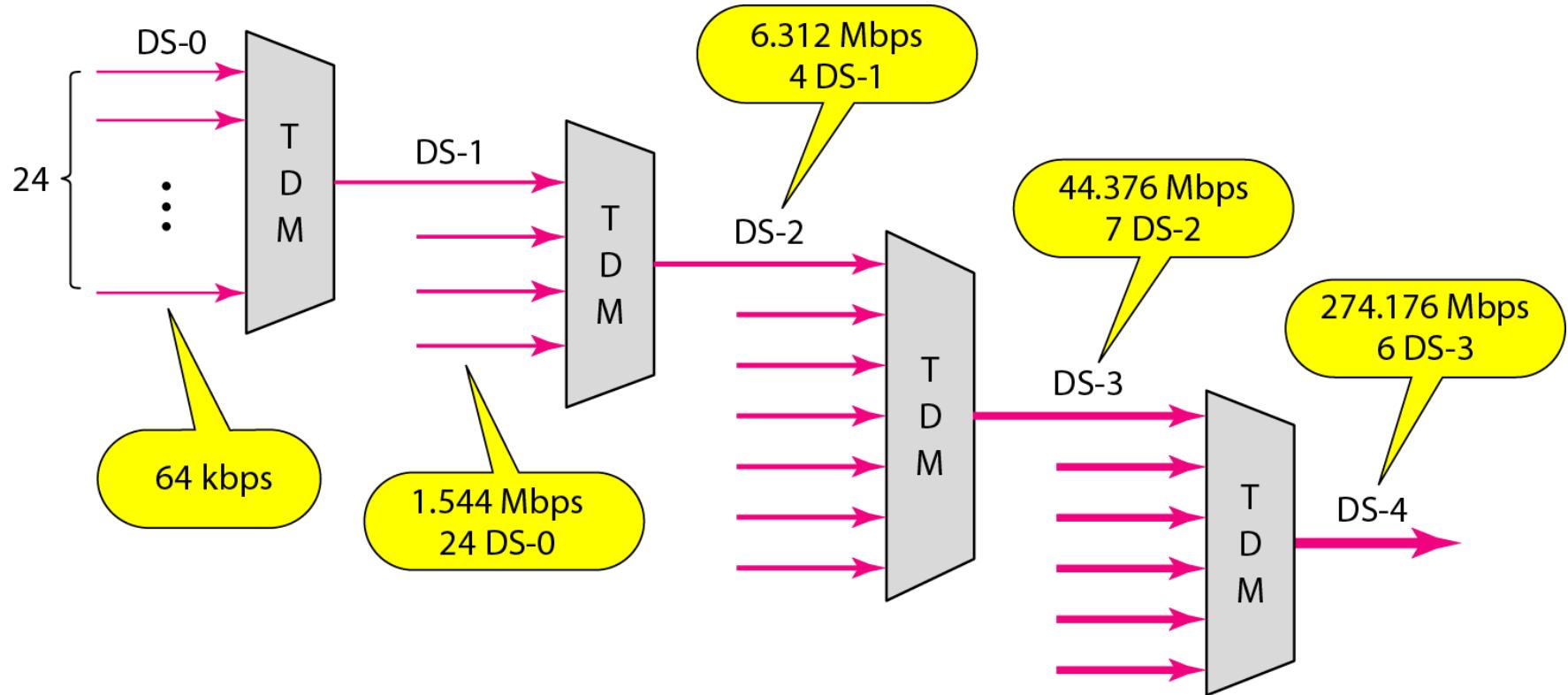


Table 6.1 DS and T line rates

<i>Service</i>	<i>Line</i>	<i>Rate (Mbps)</i>	<i>Voice Channels</i>
DS-1	T-1	1.544	24
DS-2	T-2	6.312	96
DS-3	T-3	44.736	672
DS-4	T-4	274.176	4032

Figure 6.24 T-1 line for multiplexing telephone lines

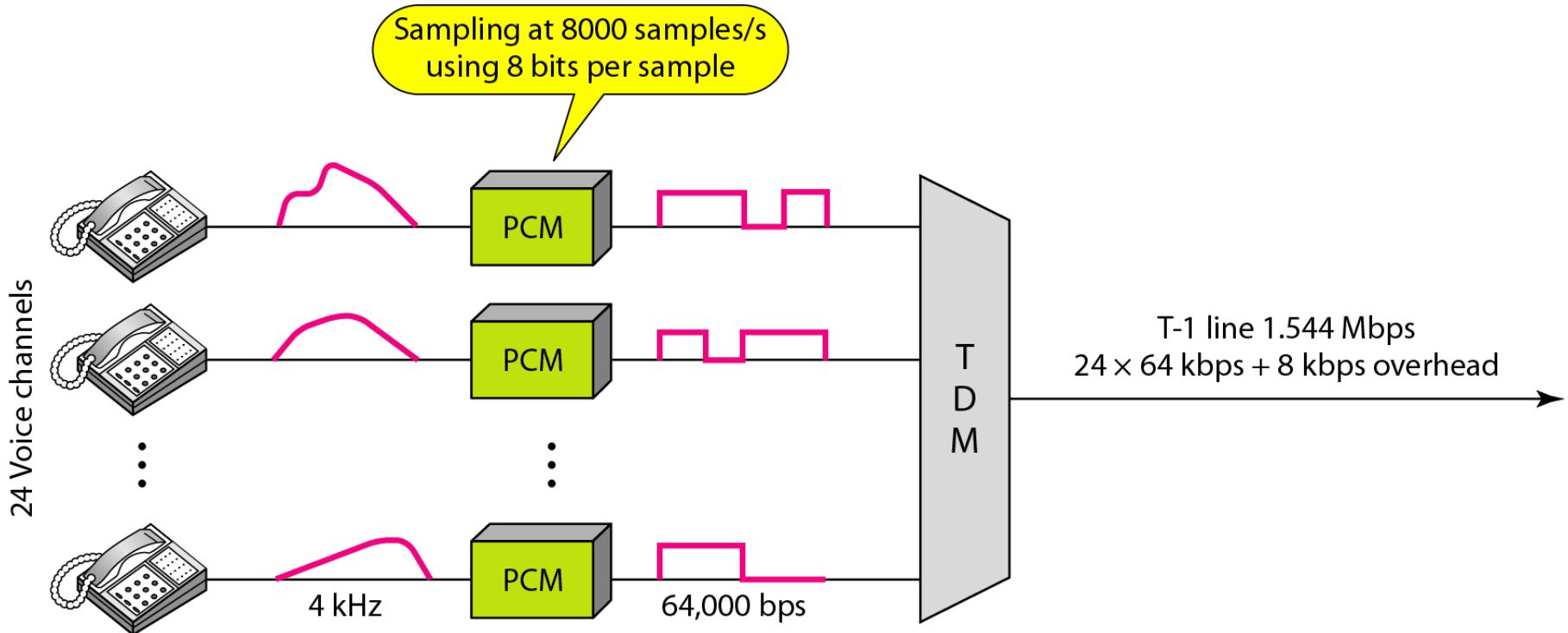


Figure 6.25 T-1 frame structure

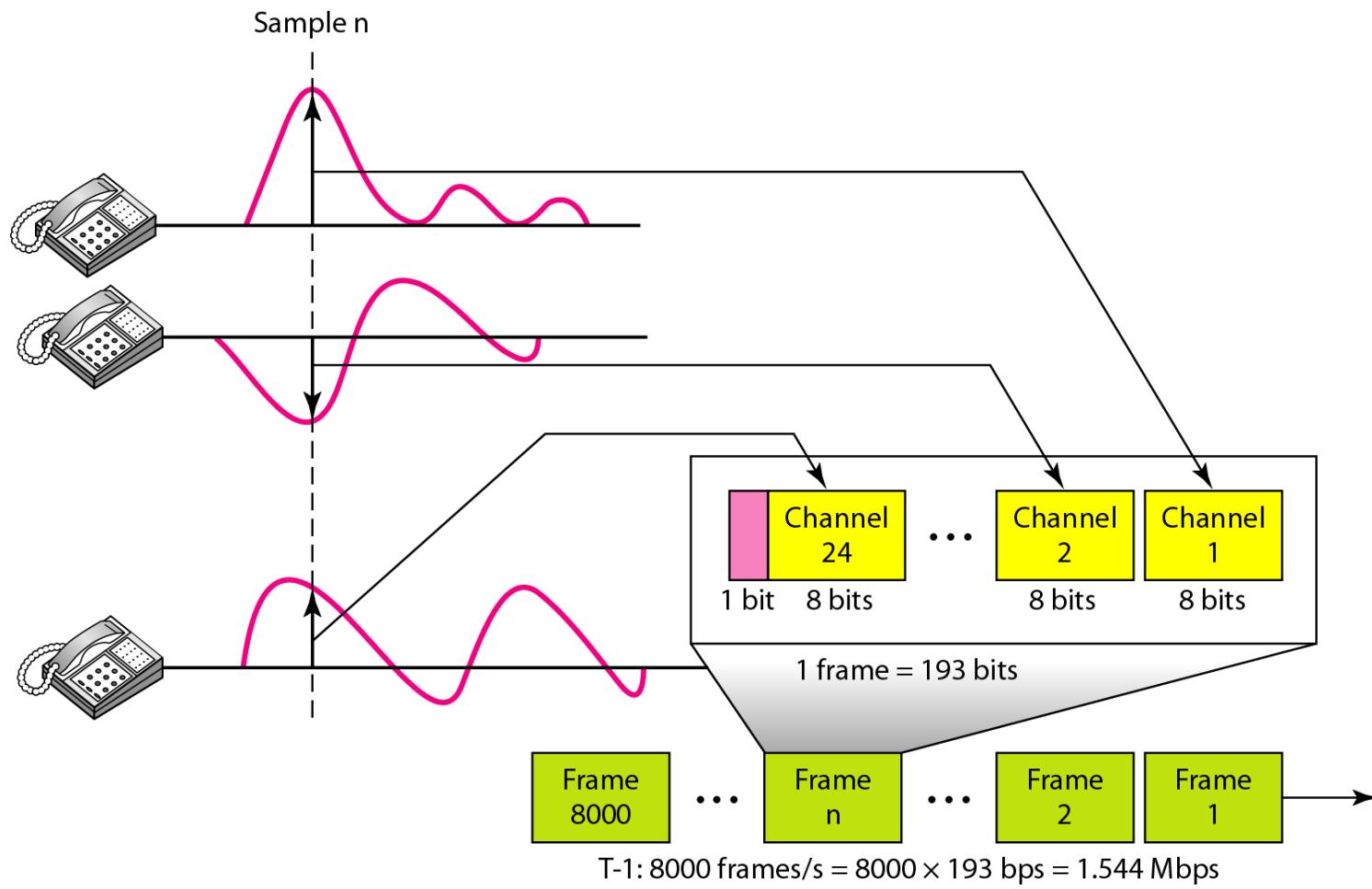
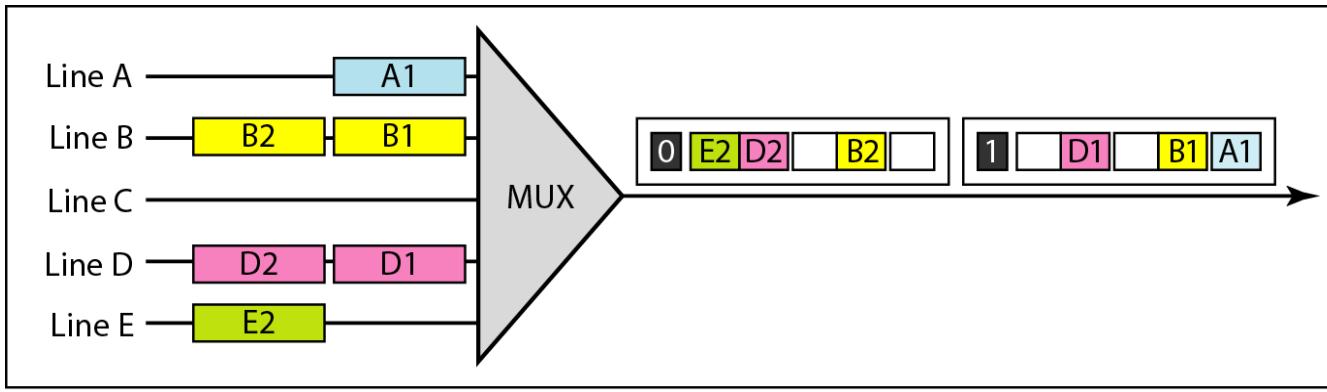


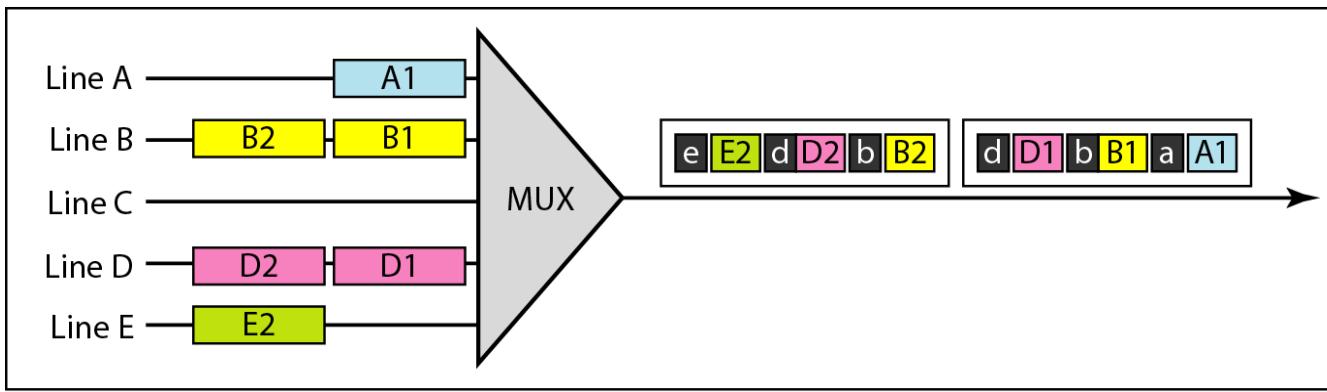
Table 6.2 E line rates

<i>Line</i>	<i>Rate (Mbps)</i>	<i>Voice Channels</i>
E-1	2.048	30
E-2	8.448	120
E-3	34.368	480
E-4	139.264	1920

Figure 6.26 TDM slot comparison



a. Synchronous TDM



b. Statistical TDM

6-1 SPREAD SPECTRUM

In spread spectrum (SS), we combine signals from different sources to fit into a larger bandwidth, but our goals are to prevent eavesdropping and jamming. To achieve these goals, spread spectrum techniques add redundancy.

Topics discussed in this section:

Frequency Hopping Spread Spectrum (FHSS)

Direct Sequence Spread Spectrum Synchronous (DSSS)

Figure 6.27 *Spread spectrum*

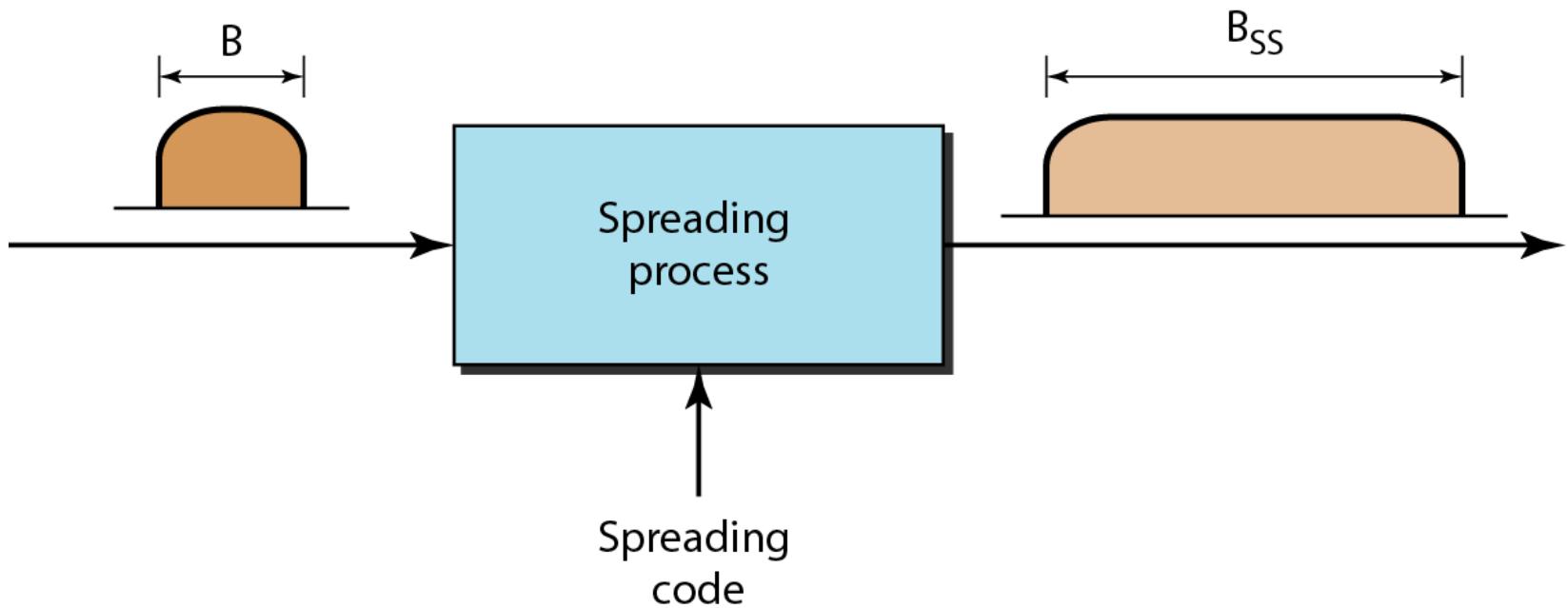


Figure 6.28 Frequency hopping spread spectrum (FHSS)

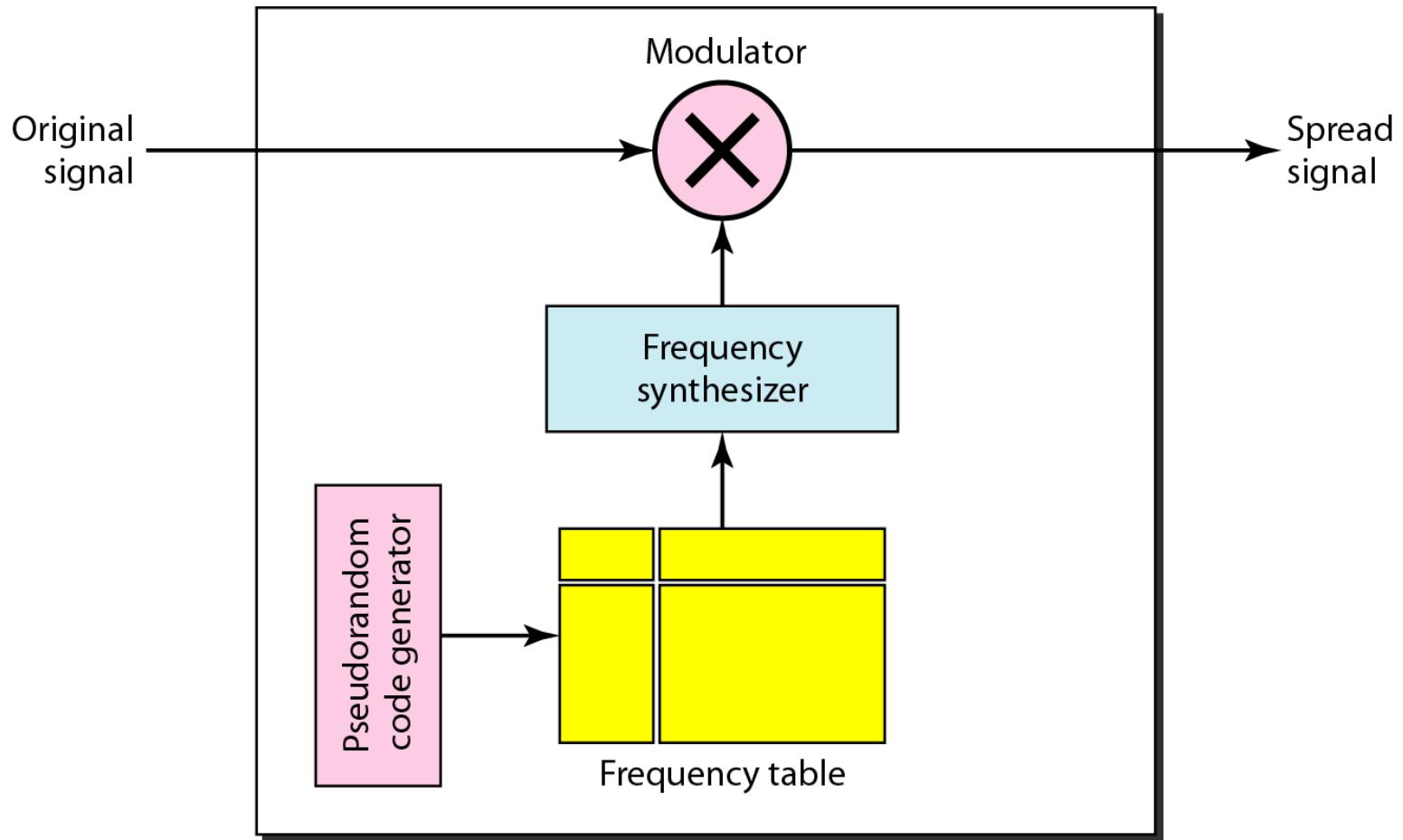


Figure 6.29 Frequency selection in FHSS

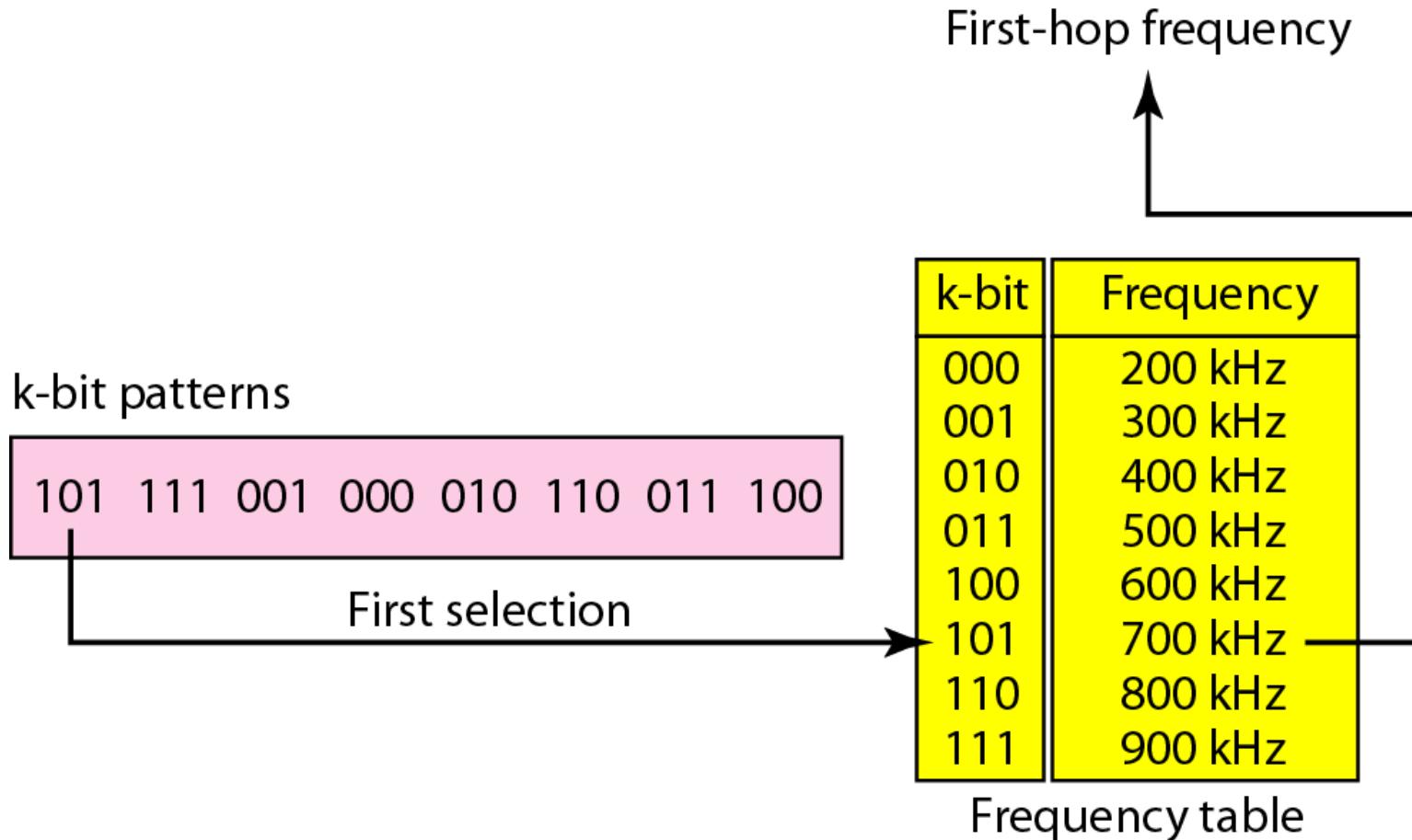


Figure 6.30 FHSS cycles

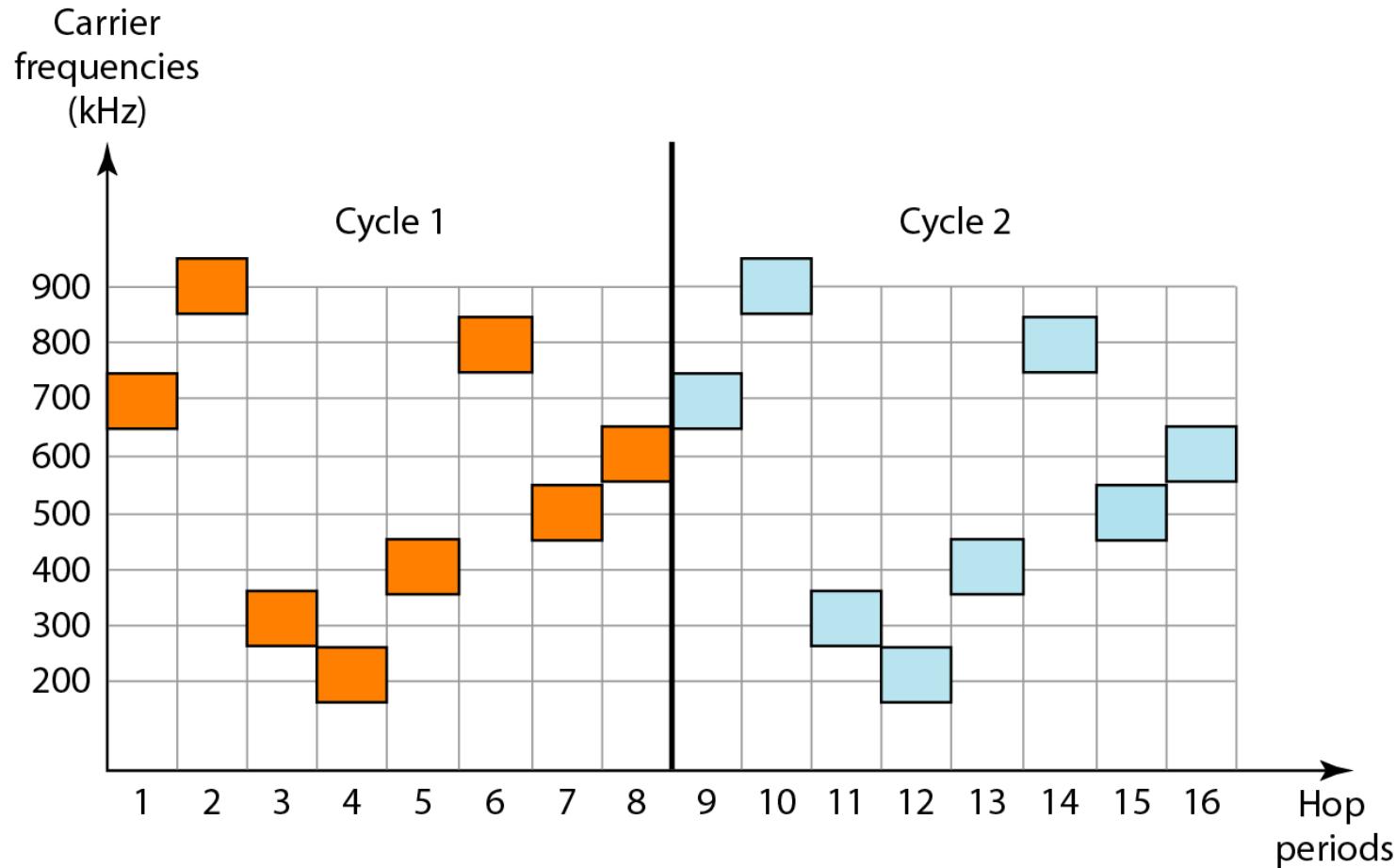
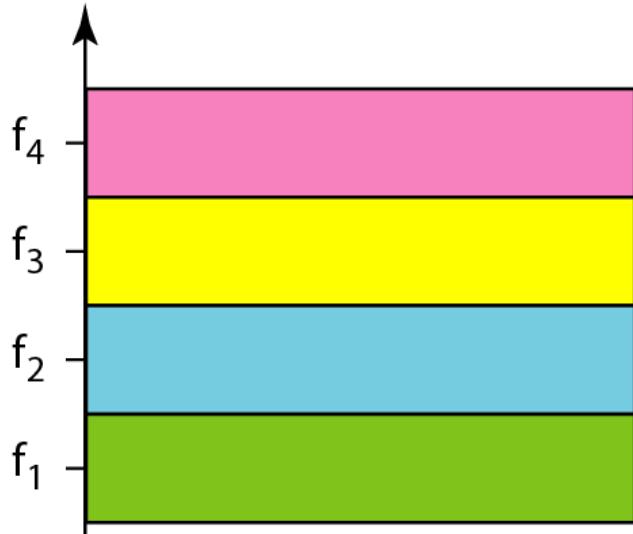


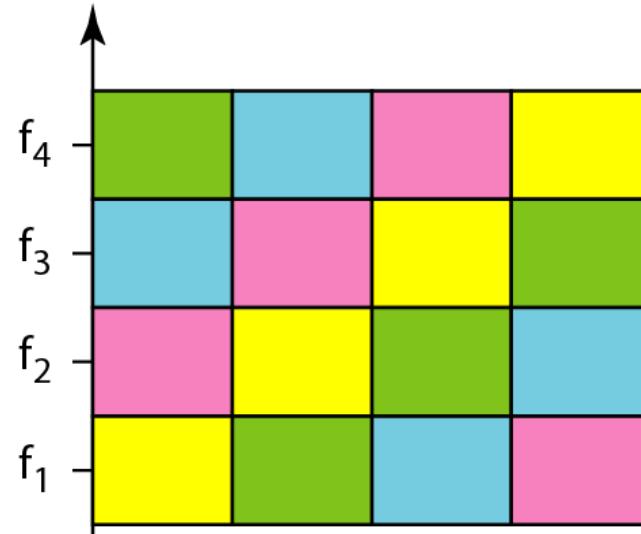
Figure 6.31 *Bandwidth sharing*

Frequency



a. FDM

Frequency



b. FHSS

Figure 6.32 DSSS

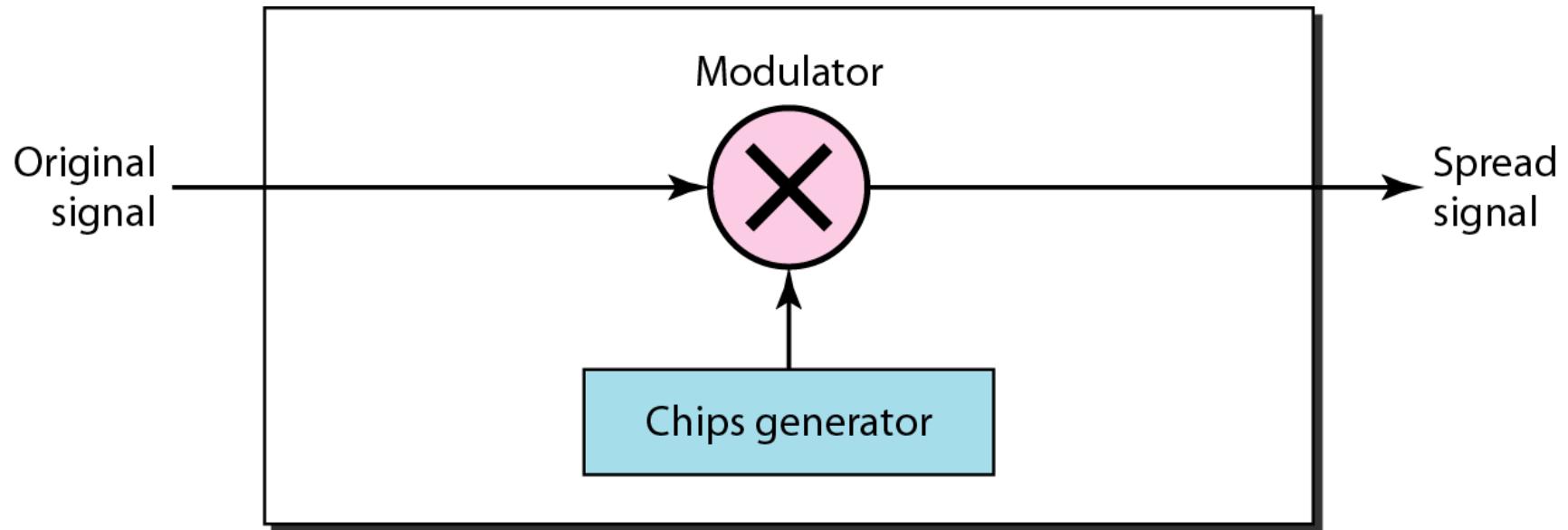
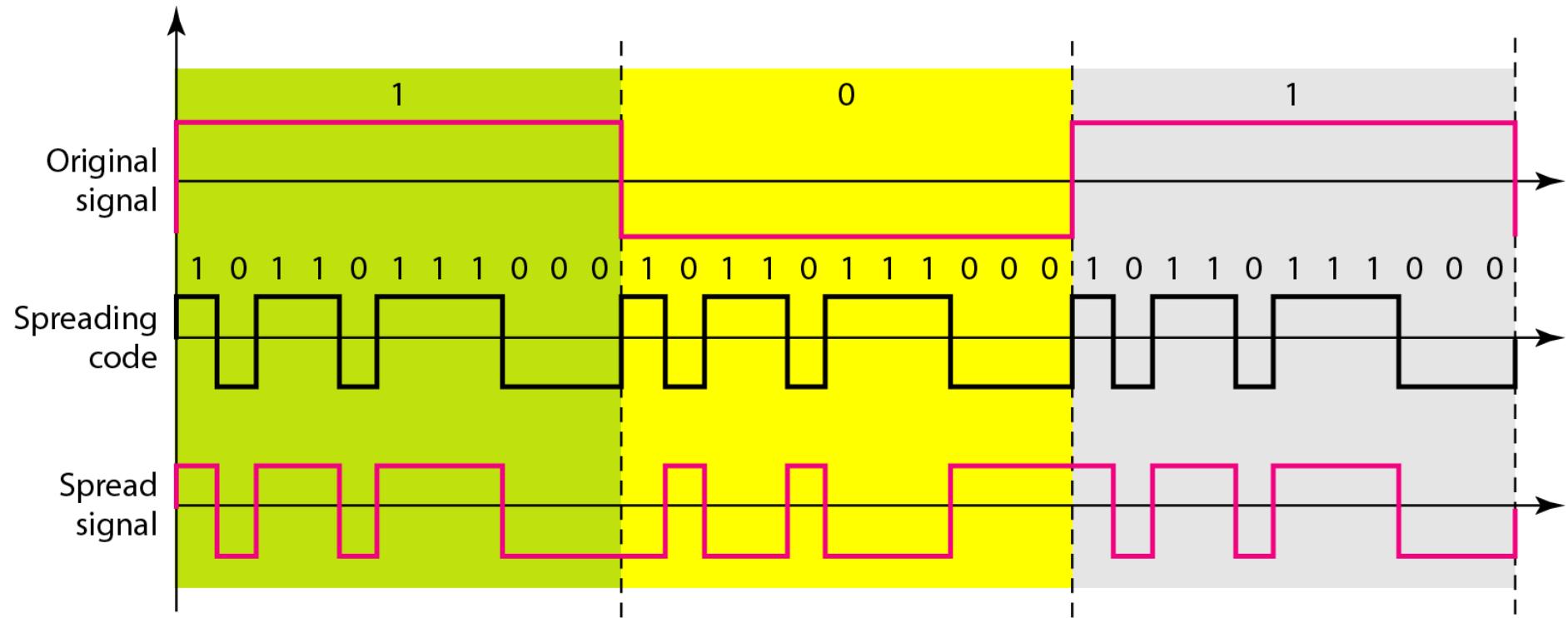


Figure 6.33 DSSS example



Chapter 7

Transmission Media

Figure 7.1 *Transmission medium and physical layer*

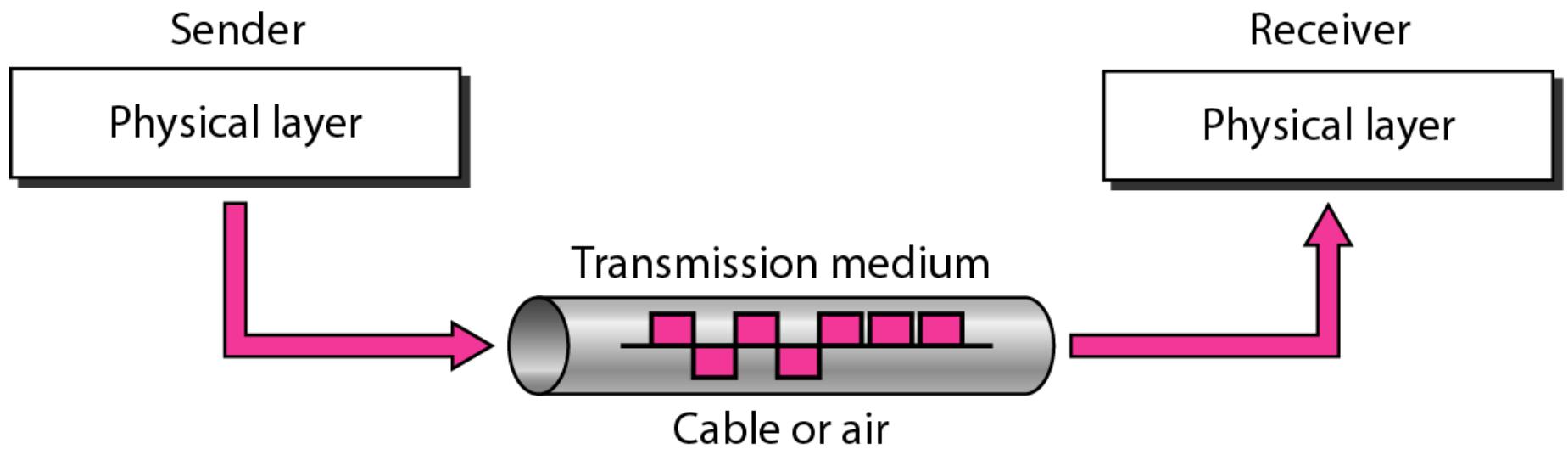
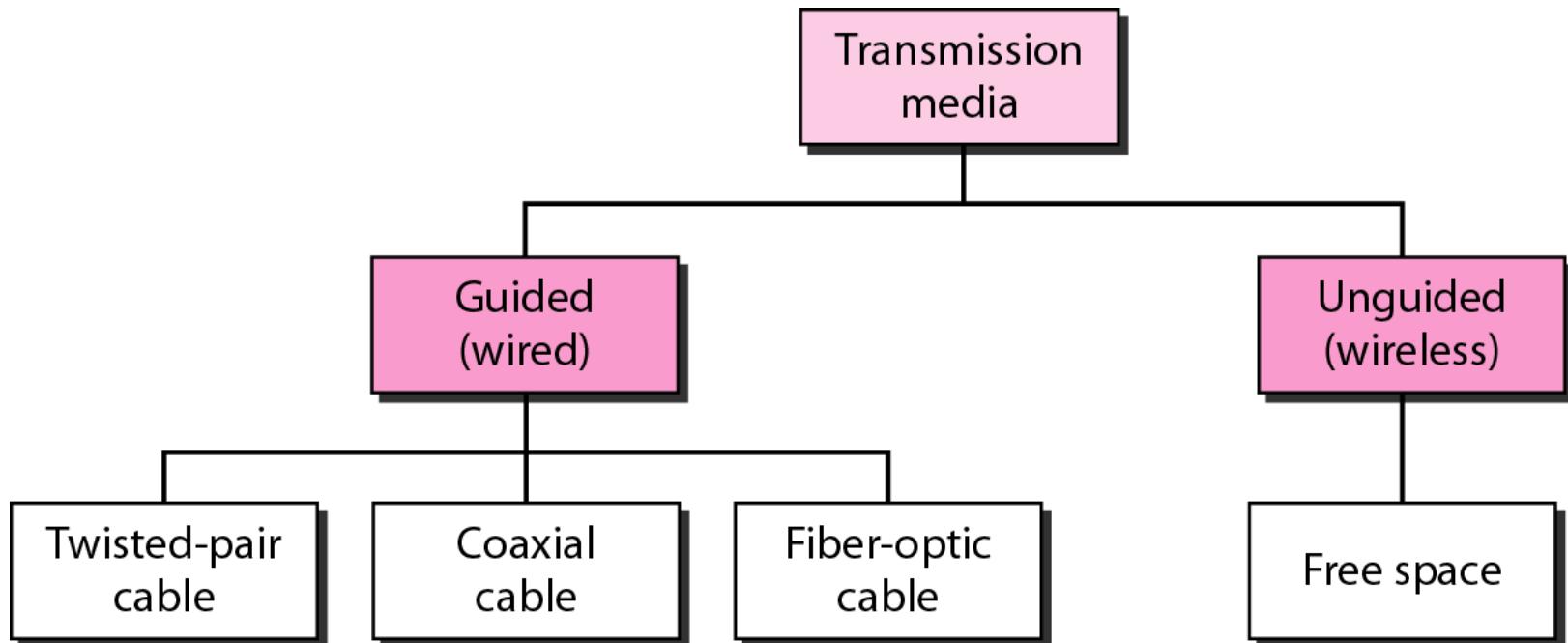


Figure 7.2 *Classes of transmission media*



7-1 GUIDED MEDIA

Guided media, which are those that provide a conduit from one device to another, include twisted-pair cable, coaxial cable, and fiber-optic cable.

Topics discussed in this section:

Twisted-Pair Cable

Coaxial Cable

Fiber-Optic Cable

Figure 7.3 *Twisted-pair cable*

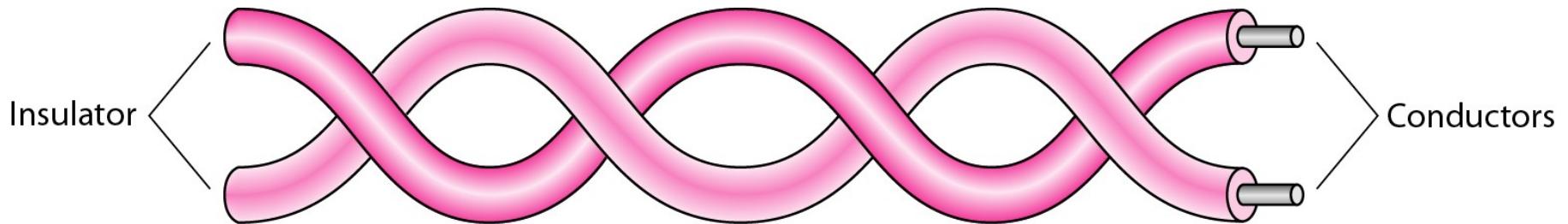
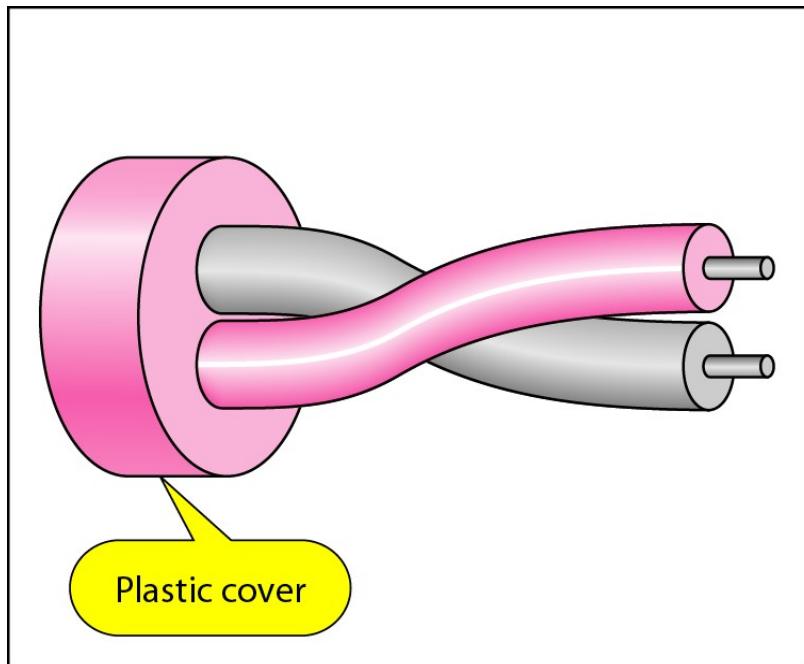
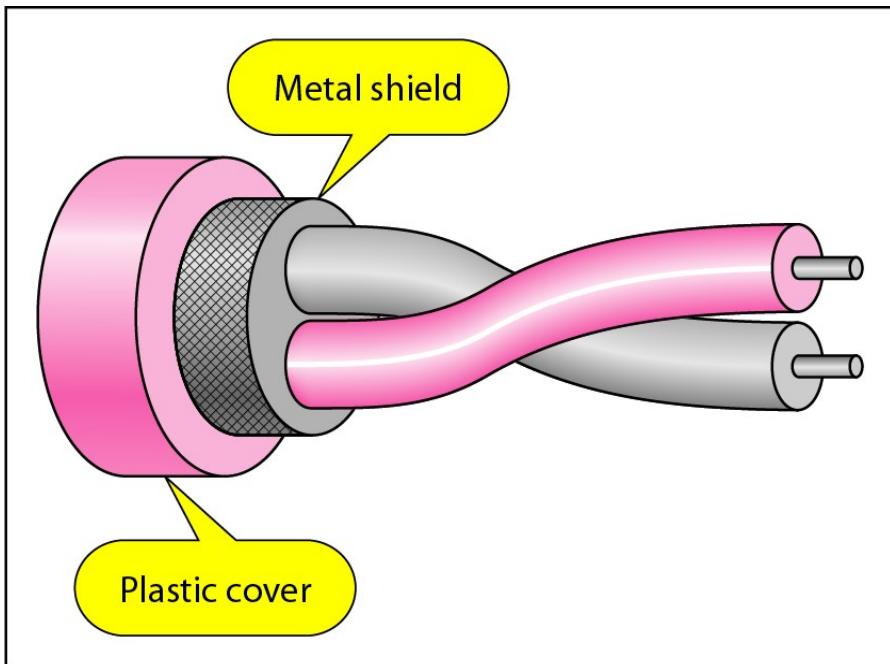


Figure 7.4 *UTP and STP cables*



a. UTP



b. STP

Table 7.1 Categories of unshielded twisted-pair cables

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

Figure 7.5 UTP connector

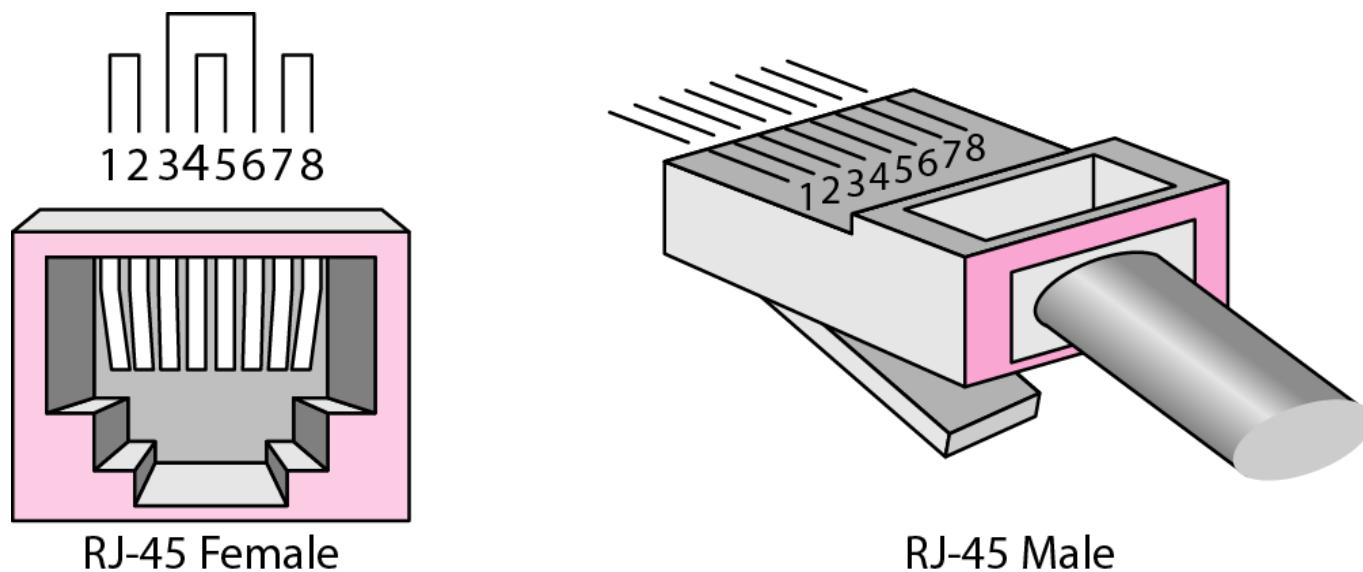


Figure 7.6 UTP performance

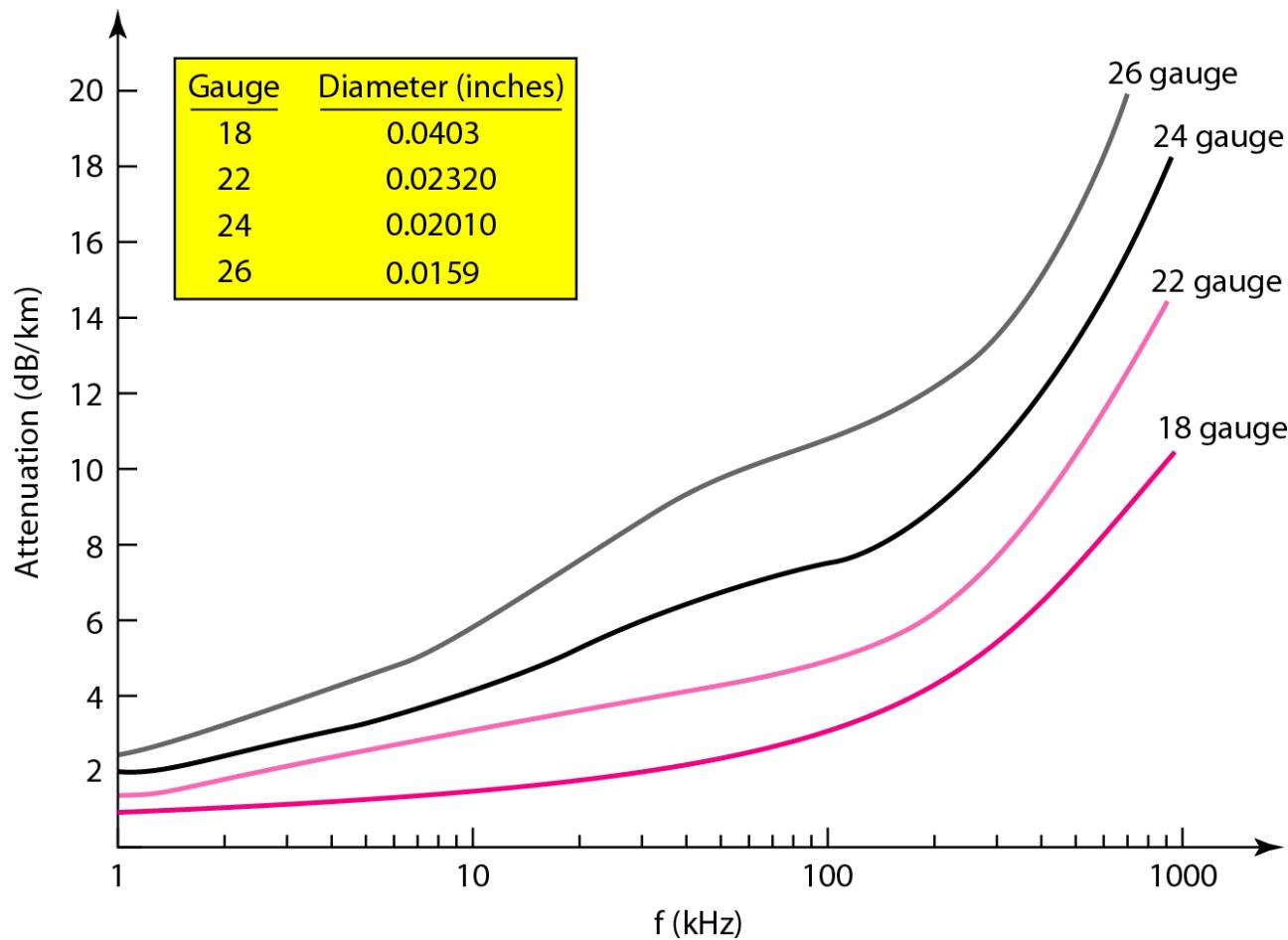


Figure 7.7 *Coaxial cable*

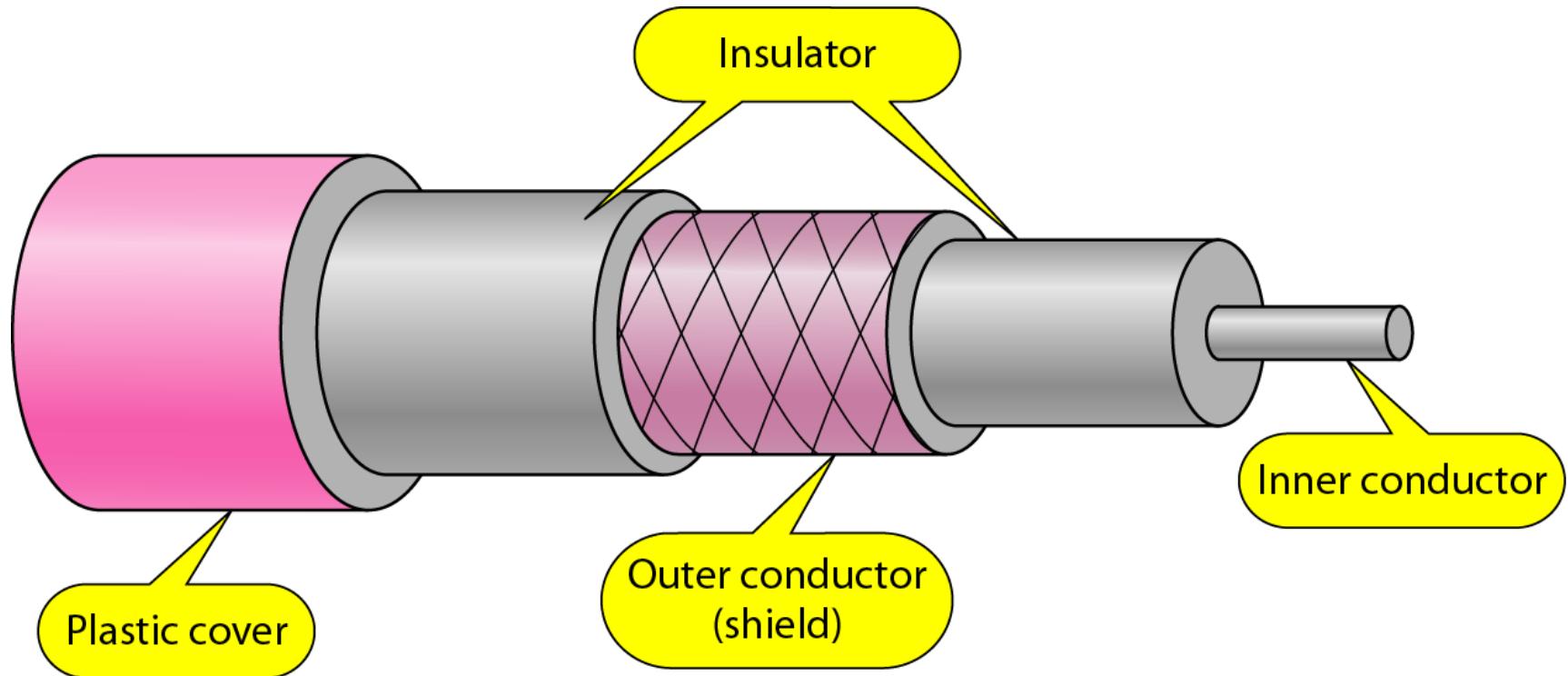


Table 7.2 *Categories of coaxial cables*

<i>Category</i>	<i>Impedance</i>	<i>Use</i>
RG-59	75Ω	Cable TV
RG-58	50Ω	Thin Ethernet
RG-11	50Ω	Thick Ethernet

Figure 7.8 *BNC connectors*

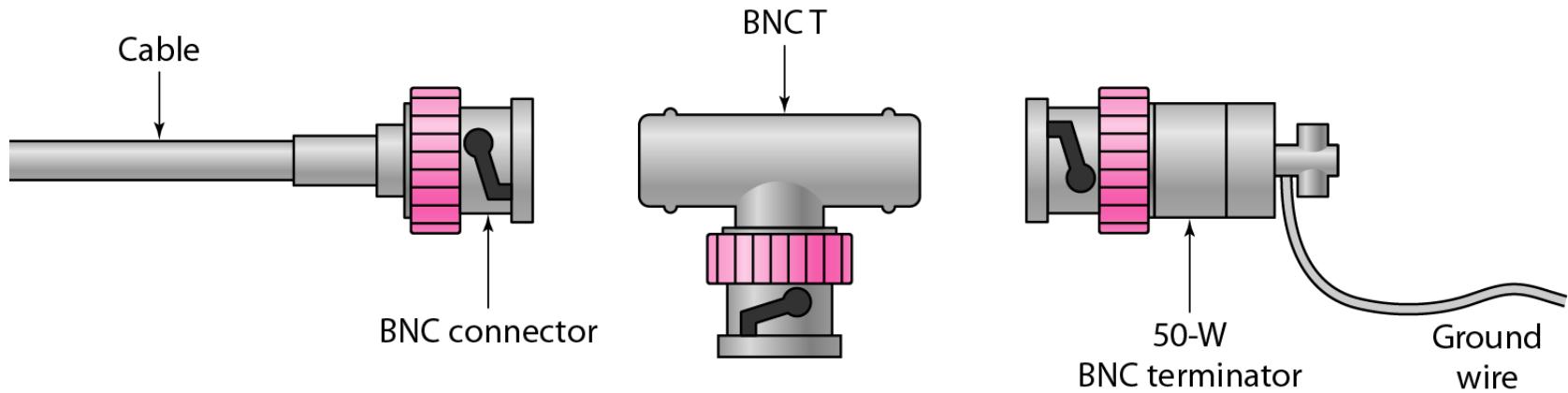


Figure 7.9 *Coaxial cable performance*

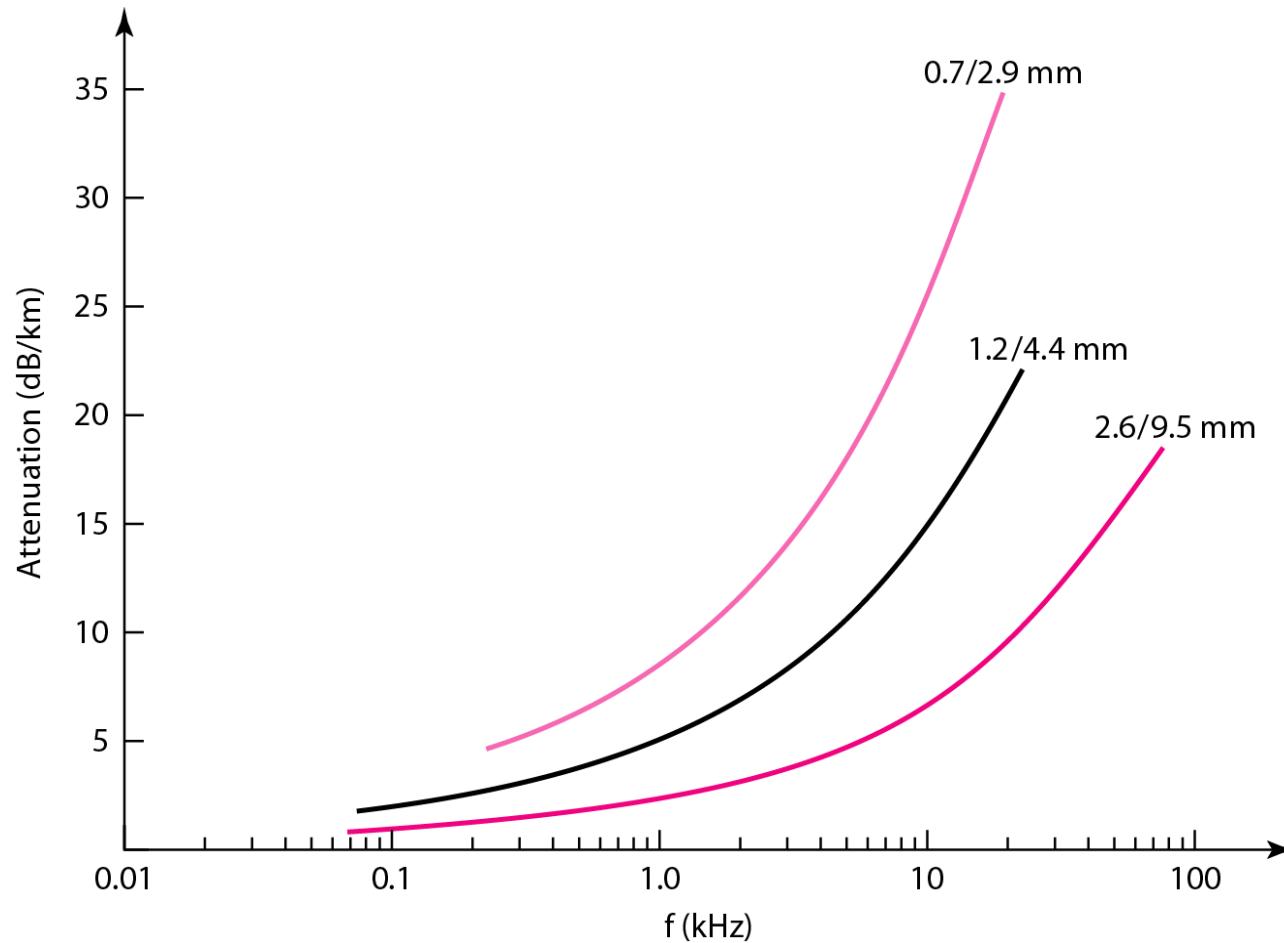
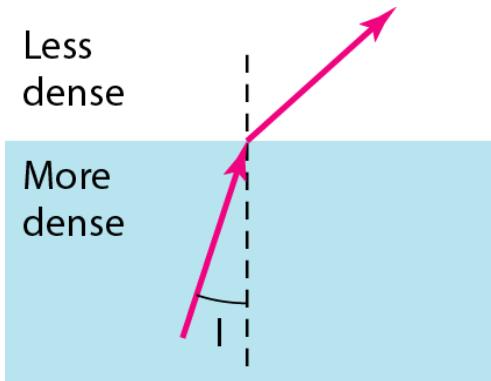
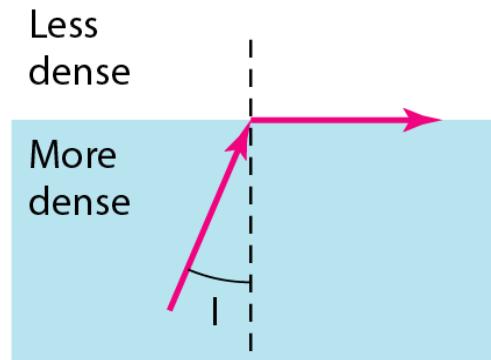


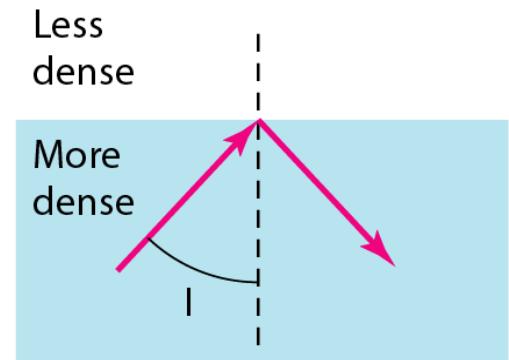
Figure 7.10 *Bending of light ray*



$I <$ critical angle,
refraction



$I =$ critical angle,
refraction



$I >$ critical angle,
reflection

Figure 7.11 *Optical fiber*

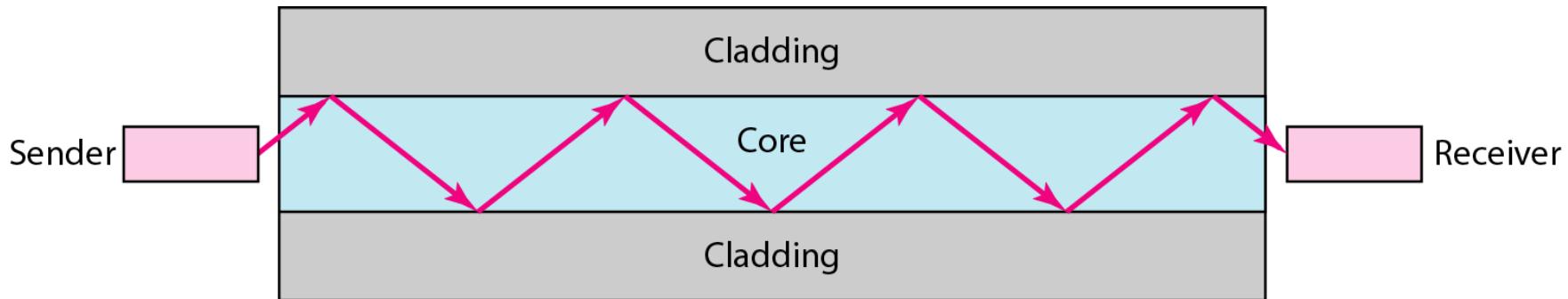


Figure 7.12 *Propagation modes*

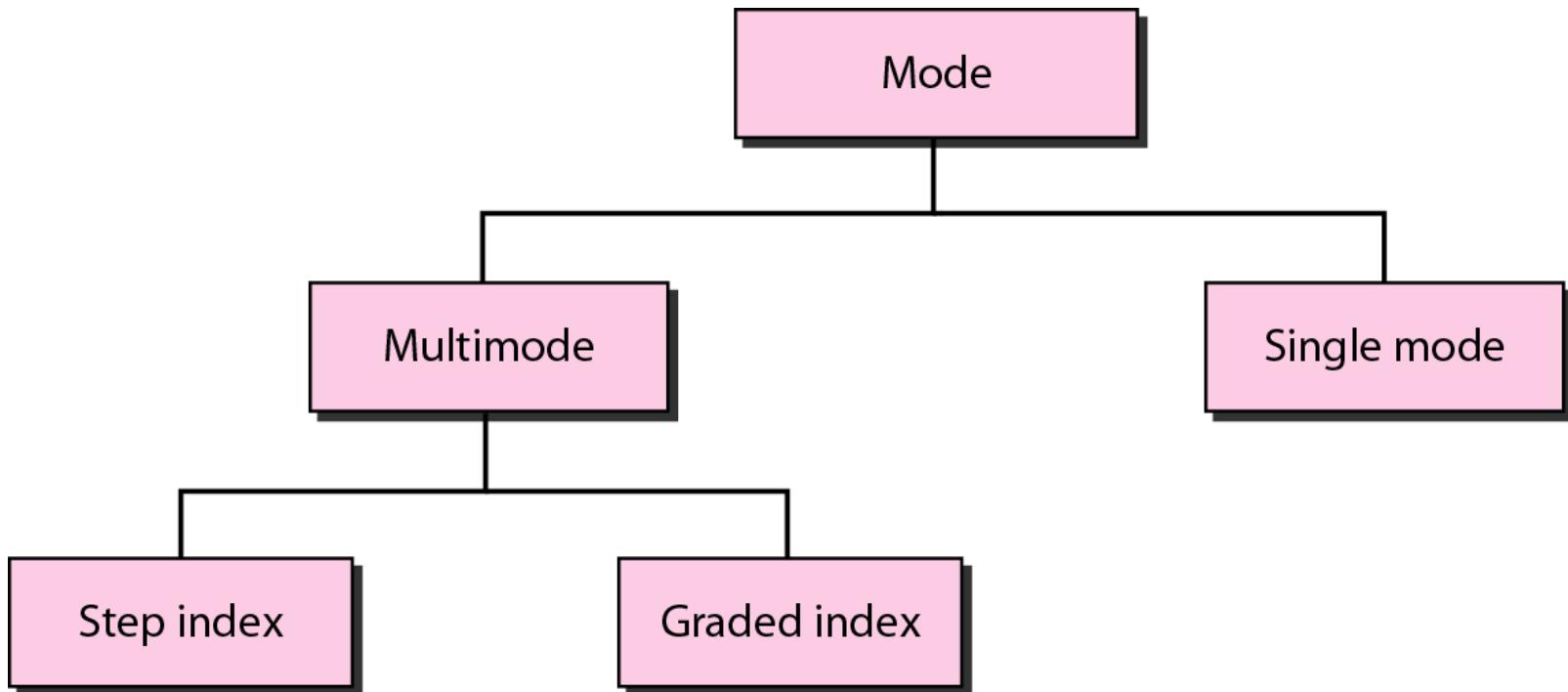
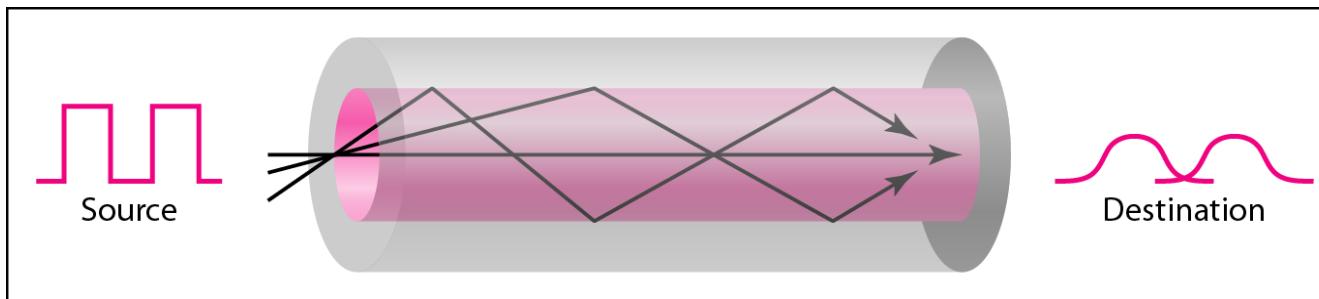
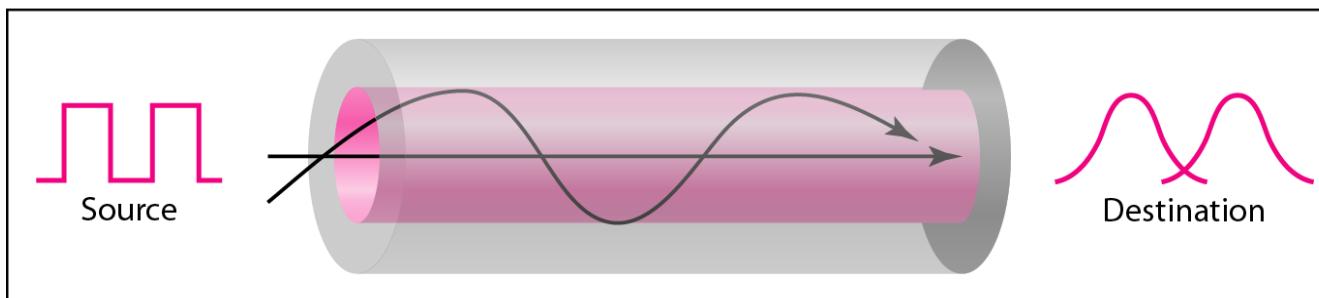


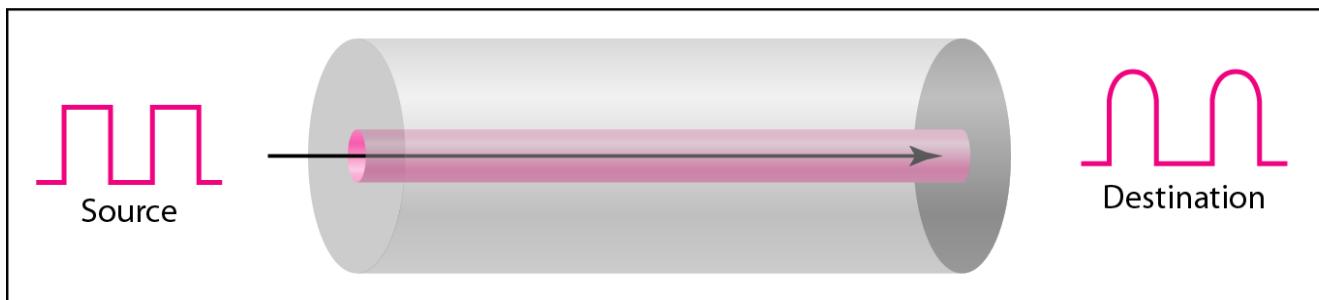
Figure 7.13 Modes



a. Multimode, step index



b. Multimode, graded index



c. Single mode

Table 7.3 *Fiber types*

Type	Core (μm)	Cladding (μm)	Mode
50/125	50.0	125	Multimode, graded index
62.5/125	62.5	125	Multimode, graded index
100/125	100.0	125	Multimode, graded index
7/125	7.0	125	Single mode

Figure 7.14 *Fiber construction*

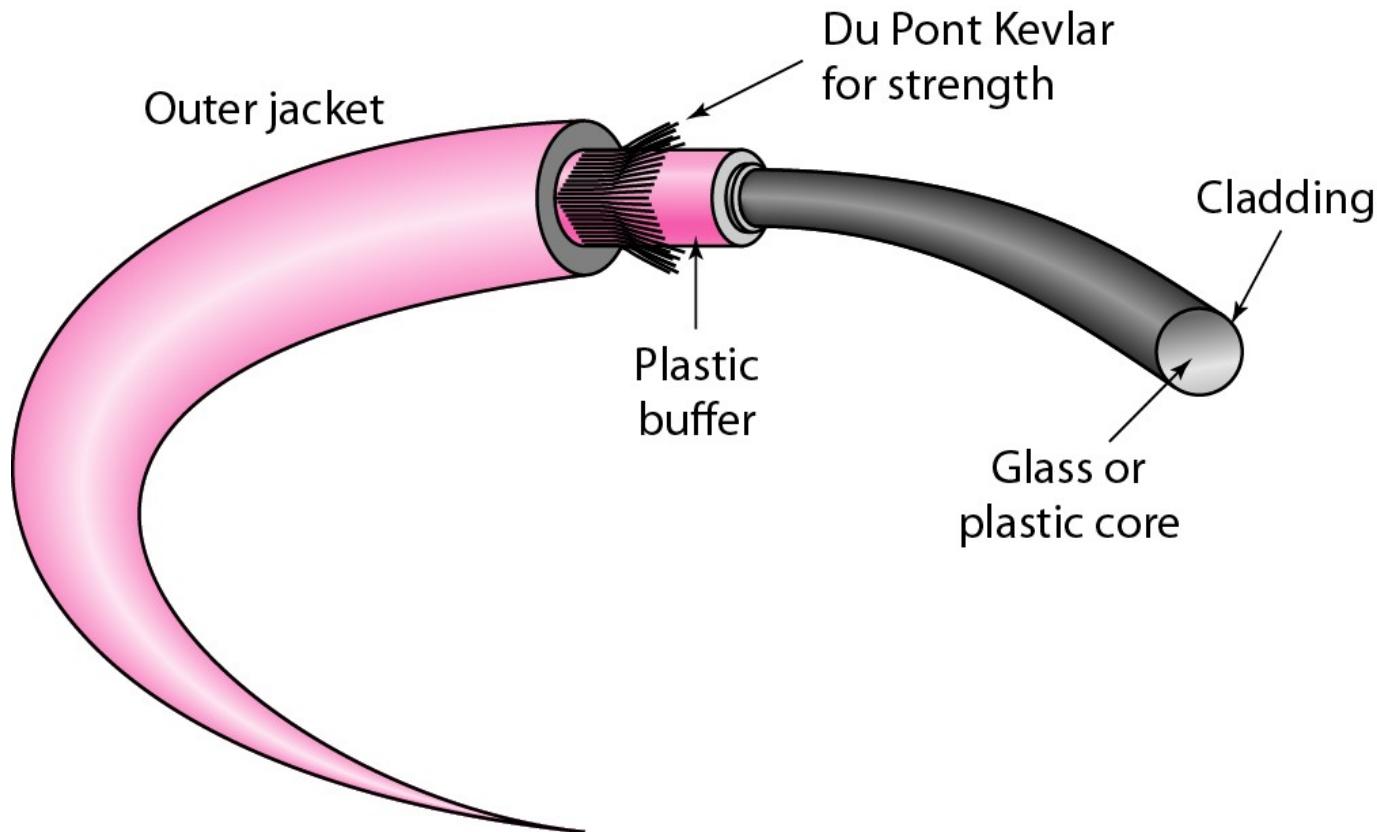
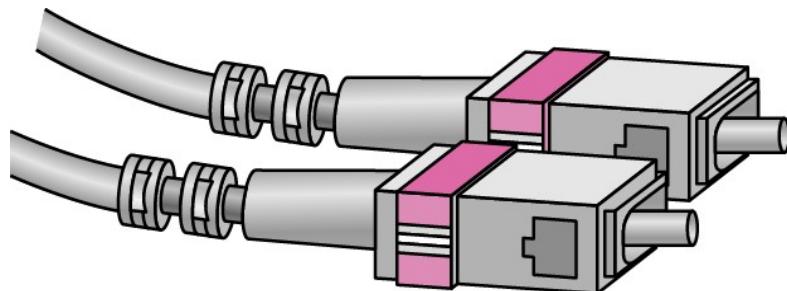
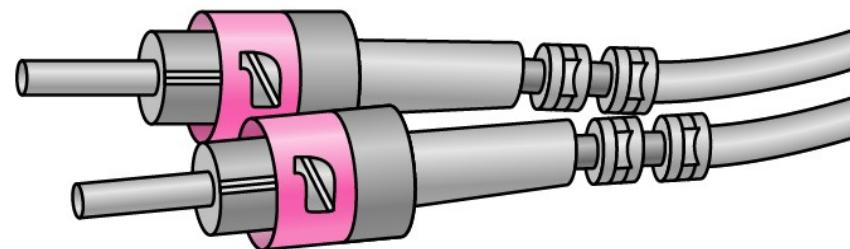


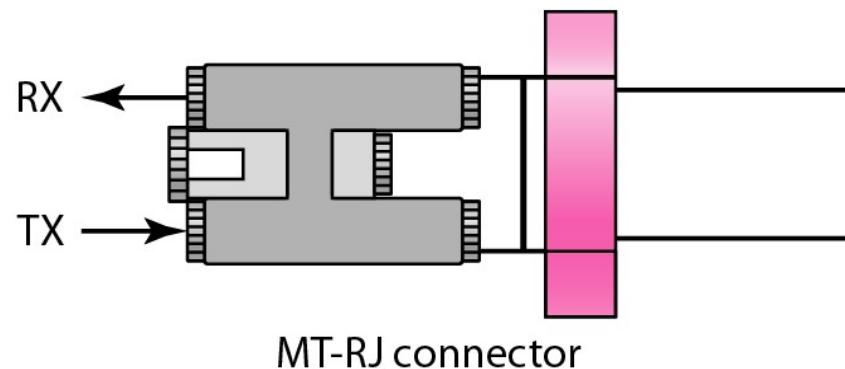
Figure 7.15 *Fiber-optic cable connectors*



SC connector

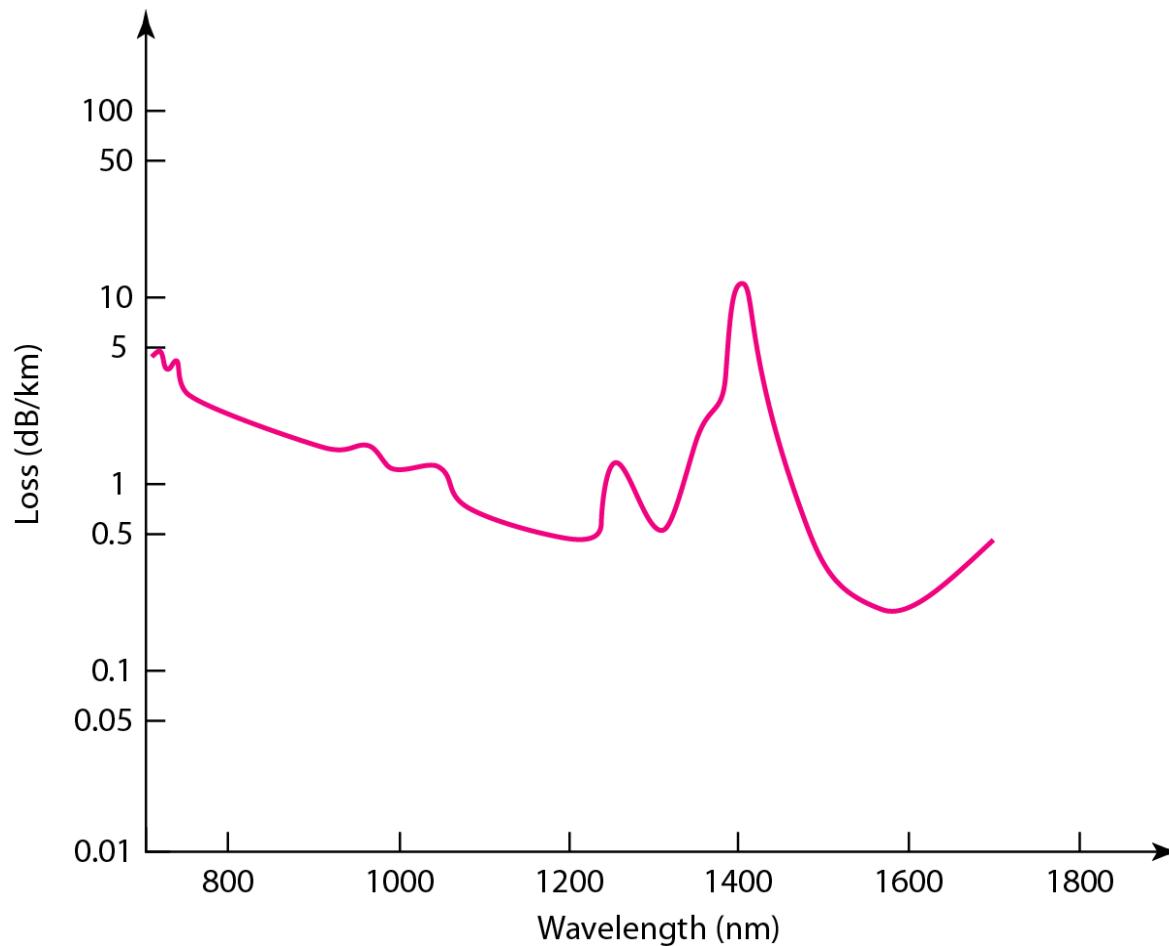


ST connector



MT-RJ connector

Figure 7.16 *Optical fiber performance*



7-2 UNGUIDED MEDIA: WIRELESS

Unguided media transport electromagnetic waves without using a physical conductor. This type of communication is often referred to as wireless communication.

Topics discussed in this section:

Radio Waves

Microwaves

Infrared

Figure 7.17 Electromagnetic spectrum for wireless communication

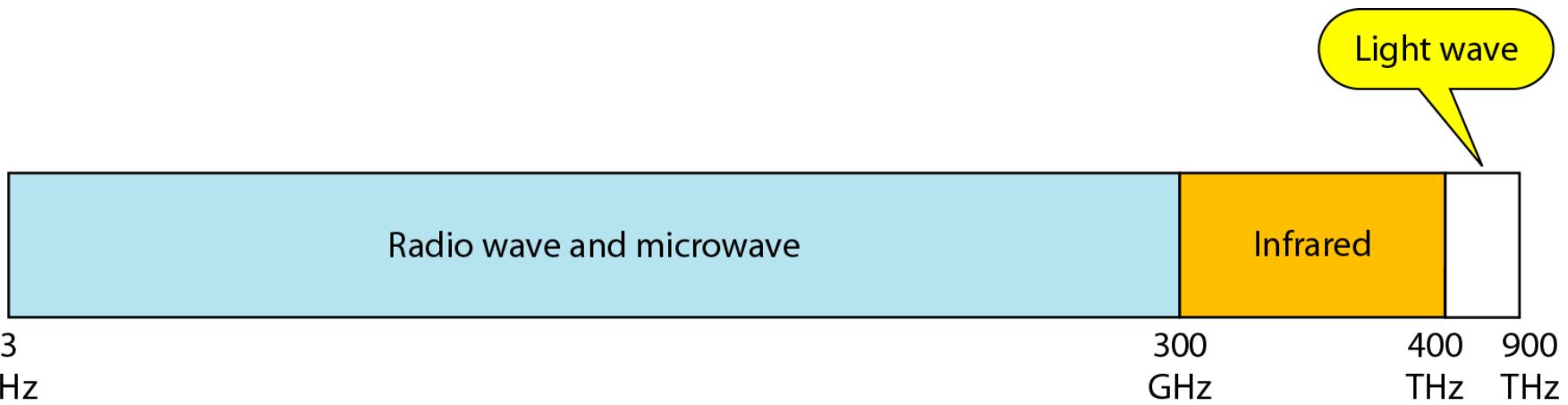


Figure 7.18 Propagation methods

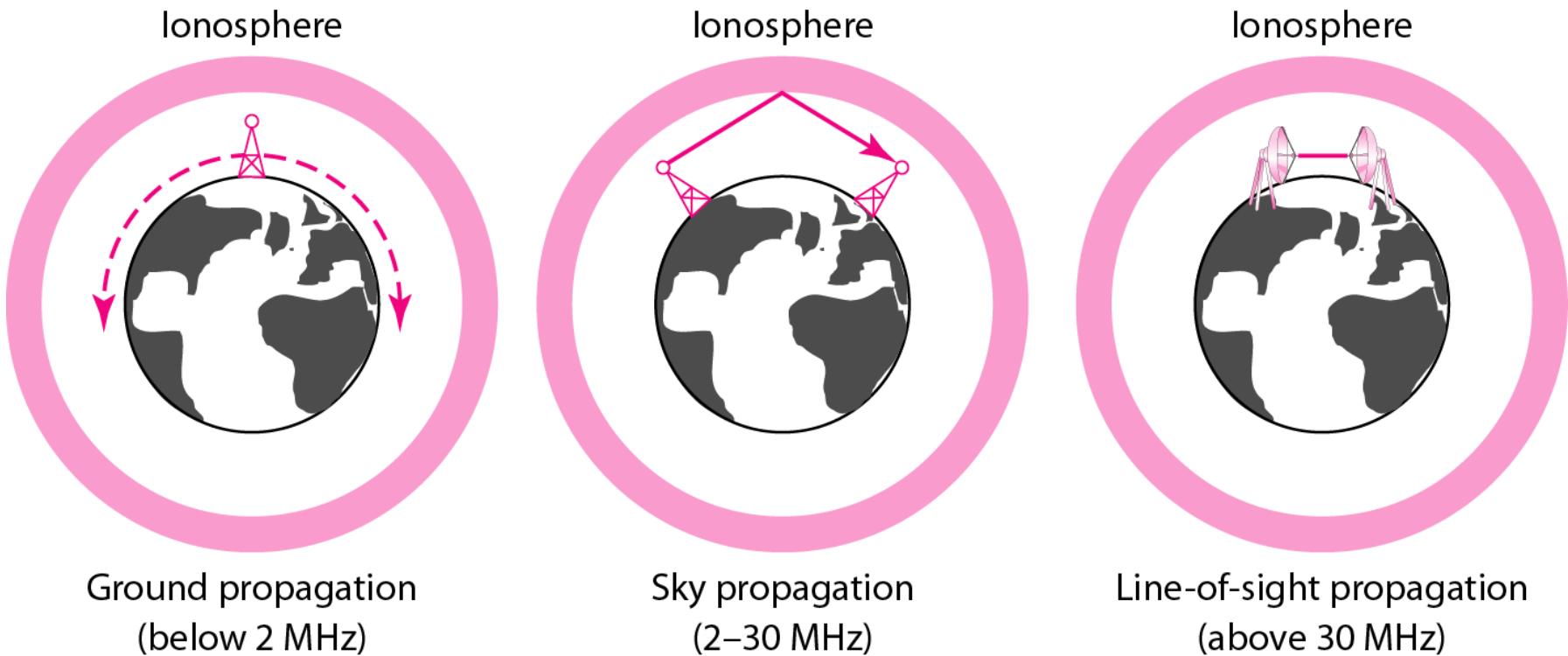


Table 7.4 Bands

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

Figure 7.19 *Wireless transmission waves*

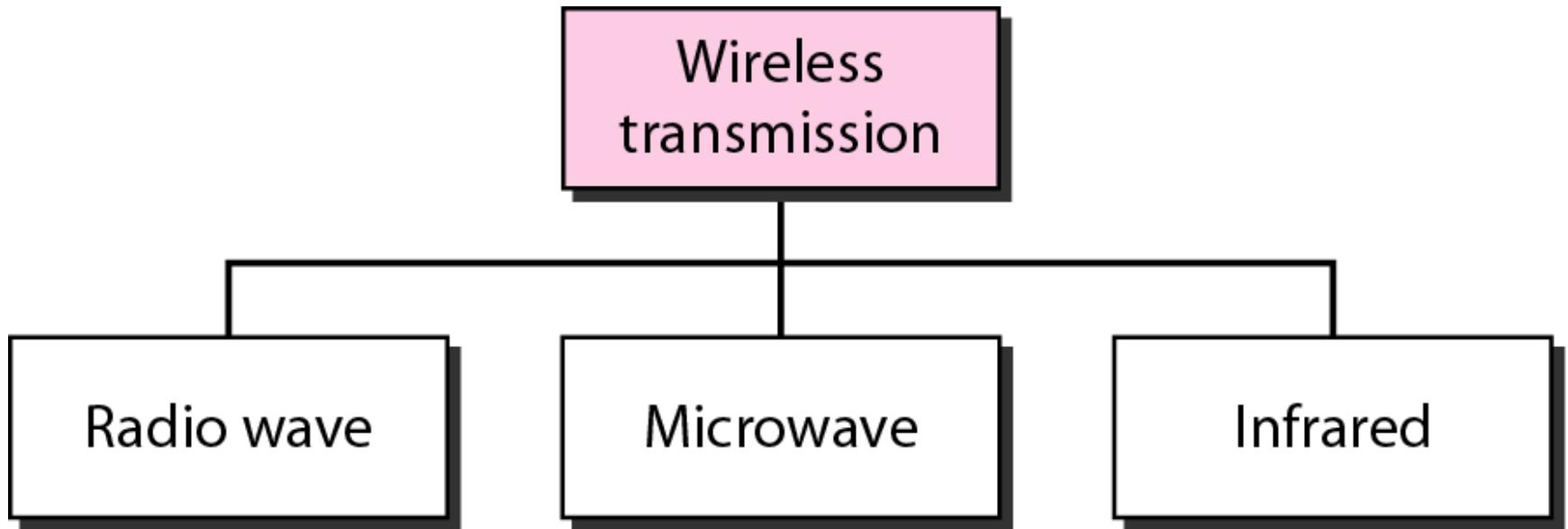
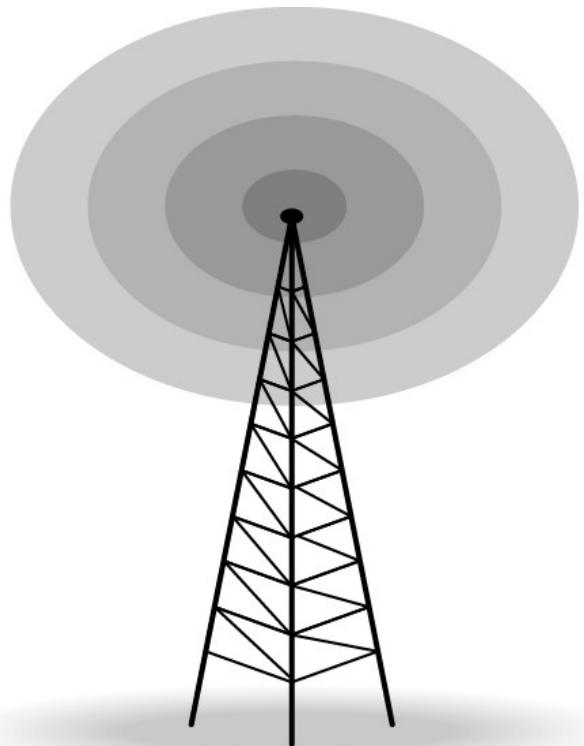
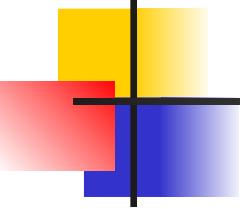


Figure 7.20 *Omnidirectional antenna*

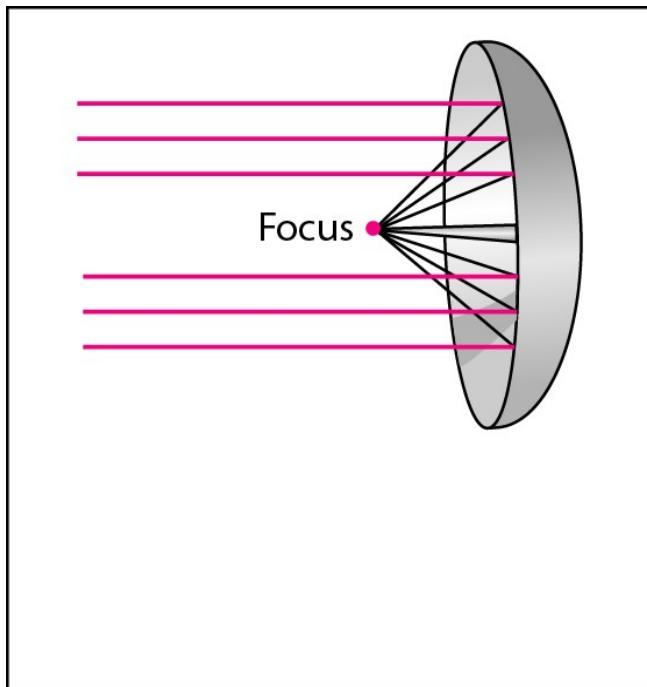




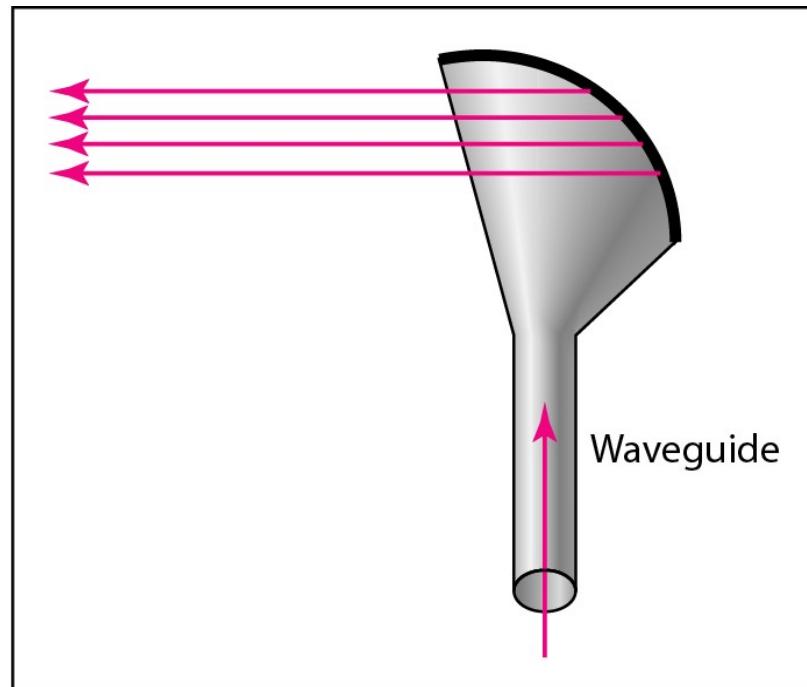
Note

Radio waves are used for multicast communications, such as radio and television, and paging systems.

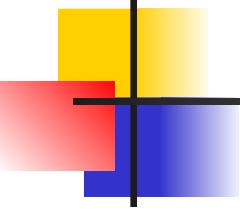
Figure 7.21 *Unidirectional antennas*



a. Dish antenna

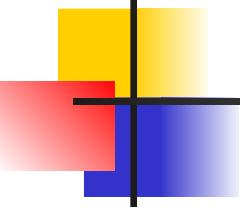


b. Horn antenna



Note

Microwaves are used for unicast communication such as cellular telephones, satellite networks, and wireless LANs.



Note

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

Chapter 8

Switching

Figure 8.1 *Switched network*

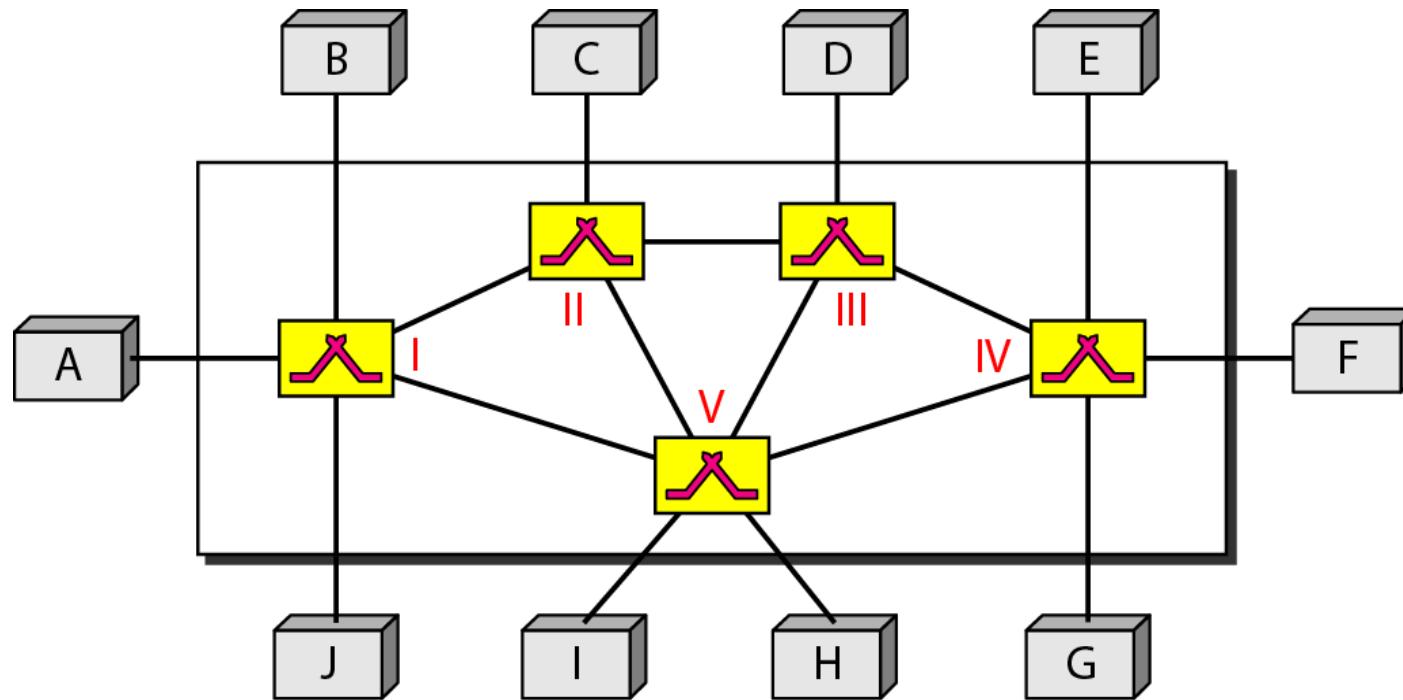
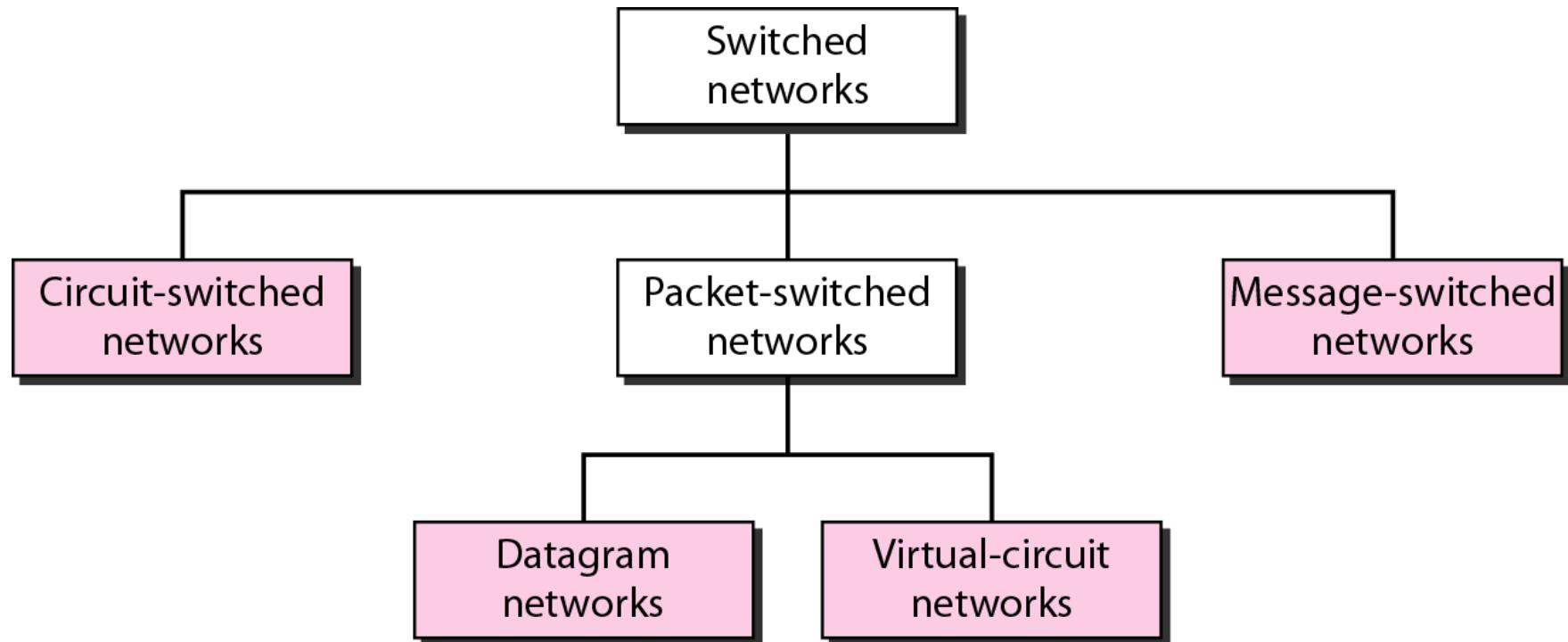


Figure 8.2 *Taxonomy of switched networks*



8-1 CIRCUIT-SWITCHED NETWORKS

A circuit-switched network consists of a set of switches connected by physical links. A connection between two stations is a dedicated path made of one or more links. However, each connection uses only one dedicated channel on each link. Each link is normally divided into n channels by using FDM or TDM.

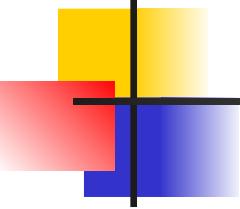
Topics discussed in this section:

Three Phases

Efficiency

Delay

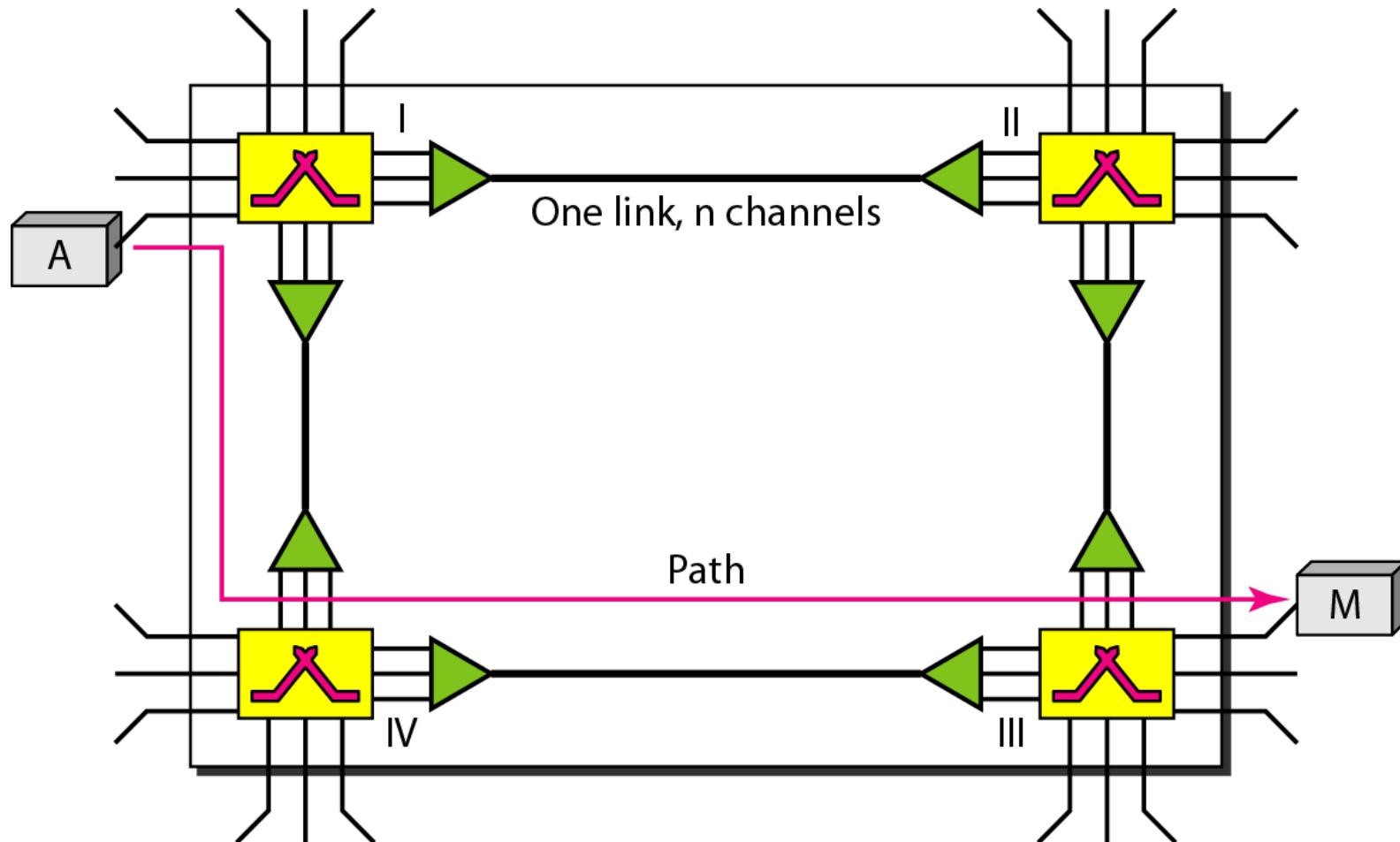
Circuit-Switched Technology in Telephone Networks

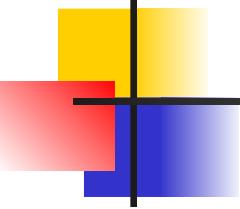


Note

A circuit-switched network is made of a set of switches connected by physical links, in which each link is divided into n channels.

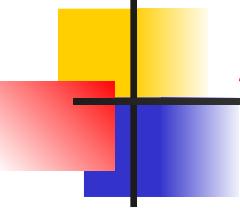
Figure 8.3 A trivial circuit-switched network





Note

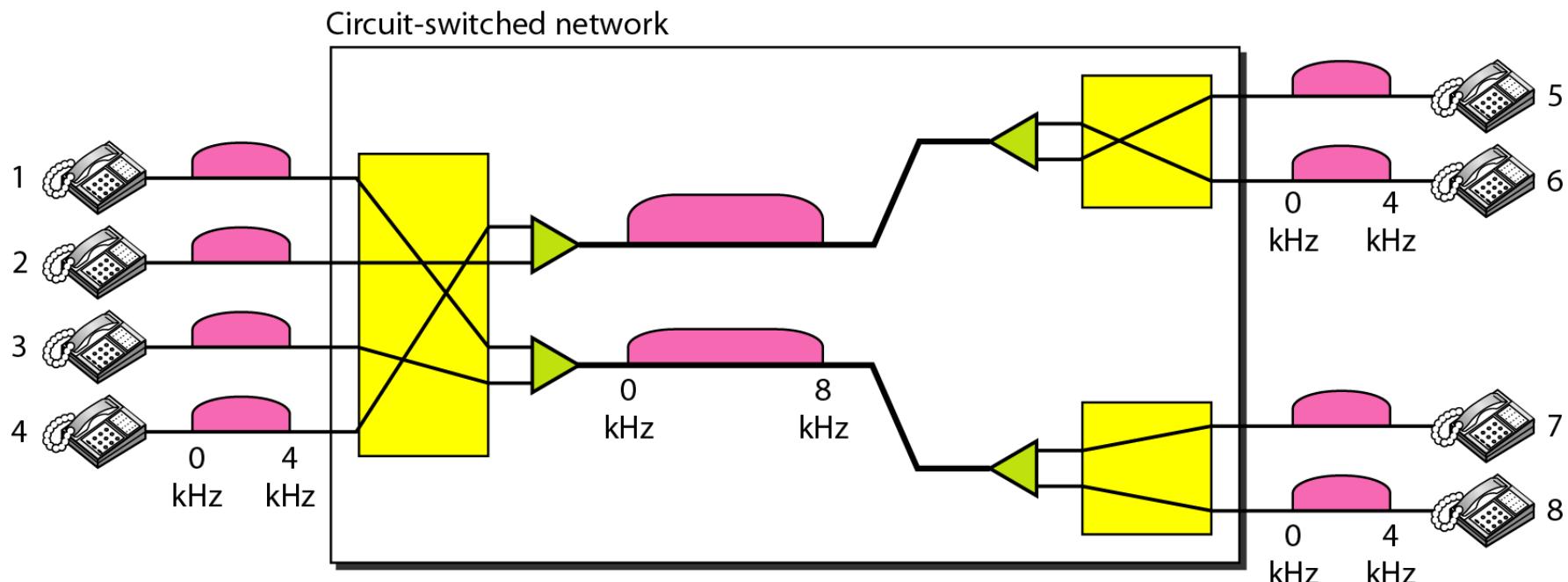
In circuit switching, the resources need to be reserved during the setup phase; the resources remain dedicated for the entire duration of data transfer until the teardown phase.

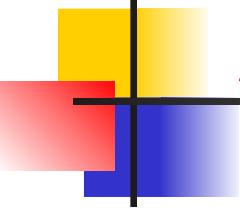


Example 8.1

As a trivial example, let us use a circuit-switched network to connect eight telephones in a small area. Communication is through 4-kHz voice channels. We assume that each link uses FDM to connect a maximum of two voice channels. The bandwidth of each link is then 8 kHz. Figure 8.4 shows the situation. Telephone 1 is connected to telephone 7; 2 to 5; 3 to 8; and 4 to 6. Of course the situation may change when new connections are made. The switch controls the connections.

Figure 8.4 Circuit-switched network used in Example 8.1





Example 8.2

As another example, consider a circuit-switched network that connects computers in two remote offices of a private company. The offices are connected using a T-1 line leased from a communication service provider. There are two 4×8 (4 inputs and 8 outputs) switches in this network. For each switch, four output ports are folded into the input ports to allow communication between computers in the same office. Four other output ports allow communication between the two offices. Figure 8.5 shows the situation.

Figure 8.5 Circuit-switched network used in Example 8.2

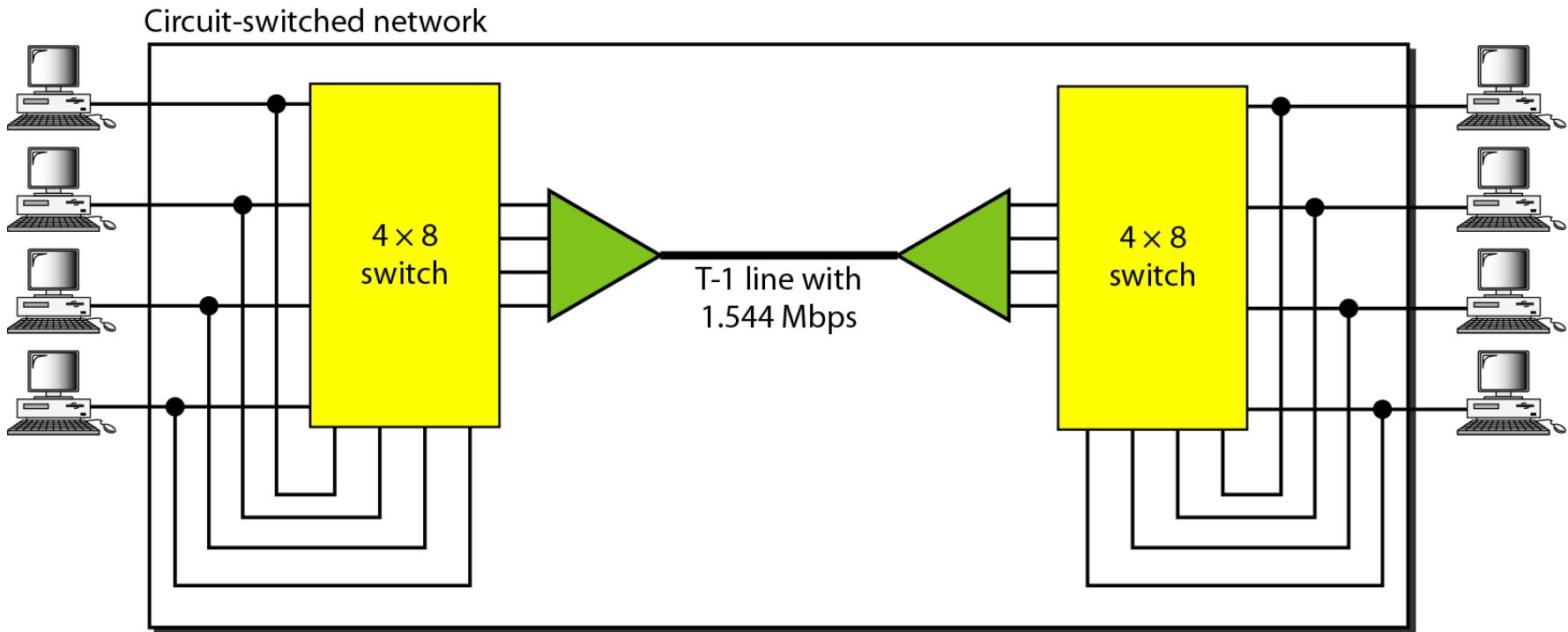
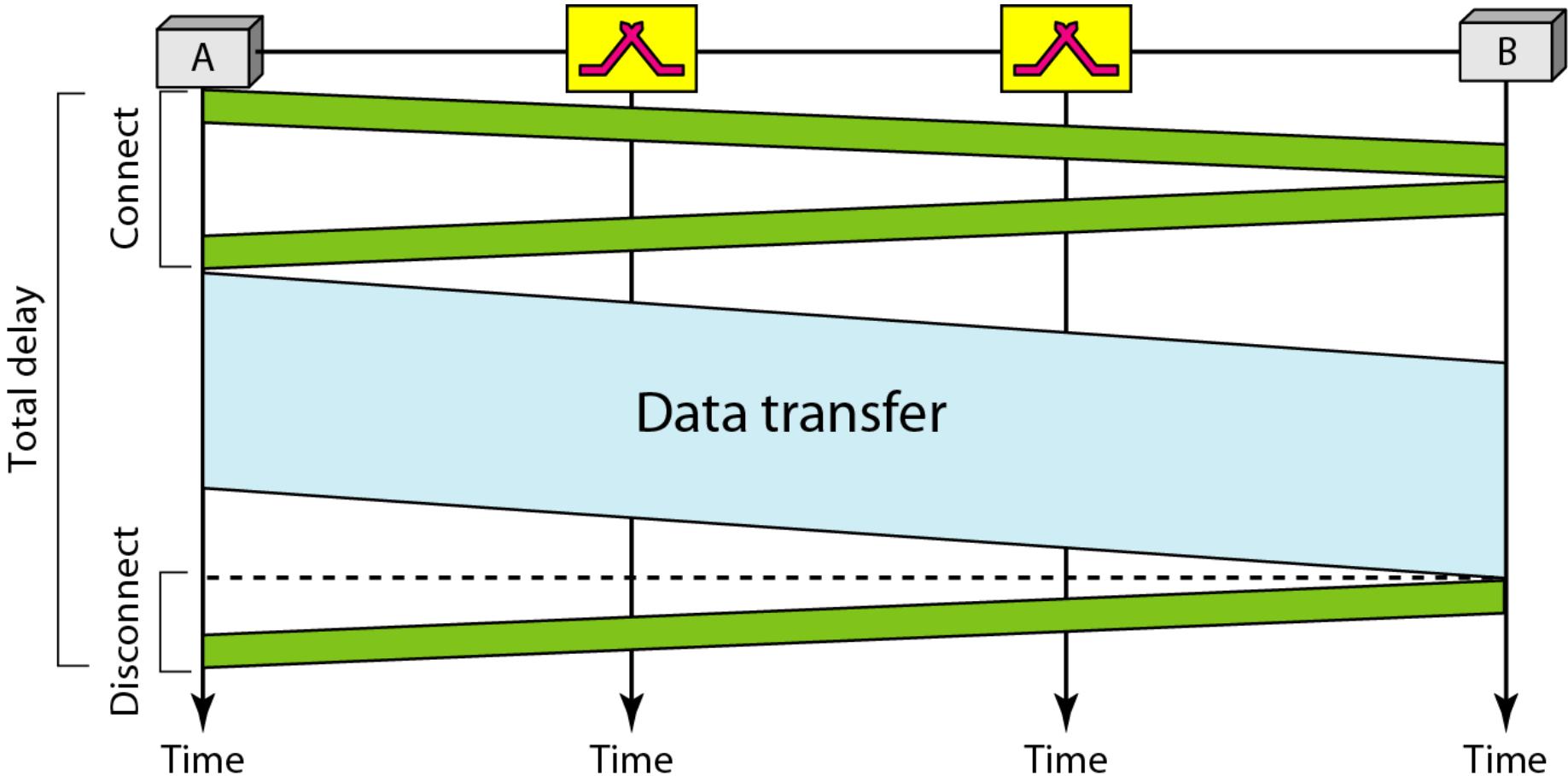
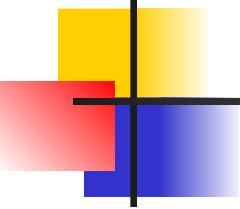


Figure 8.6 Delay in a circuit-switched network





Note

Switching at the physical layer in the traditional telephone network uses the circuit-switching approach.

8-2 DATAGRAM NETWORKS

In data communications, we need to send messages from one end system to another. If the message is going to pass through a packet-switched network, it needs to be divided into packets of fixed or variable size. The size of the packet is determined by the network and the governing protocol.

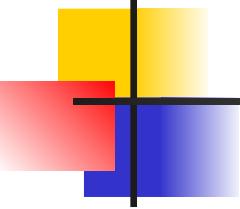
Topics discussed in this section:

Routing Table

Efficiency

Delay

Datagram Networks in the Internet



Note

**In a packet-switched network, there
is no resource reservation;
resources are allocated on demand.**

Figure 8.7 A datagram network with four switches (routers)

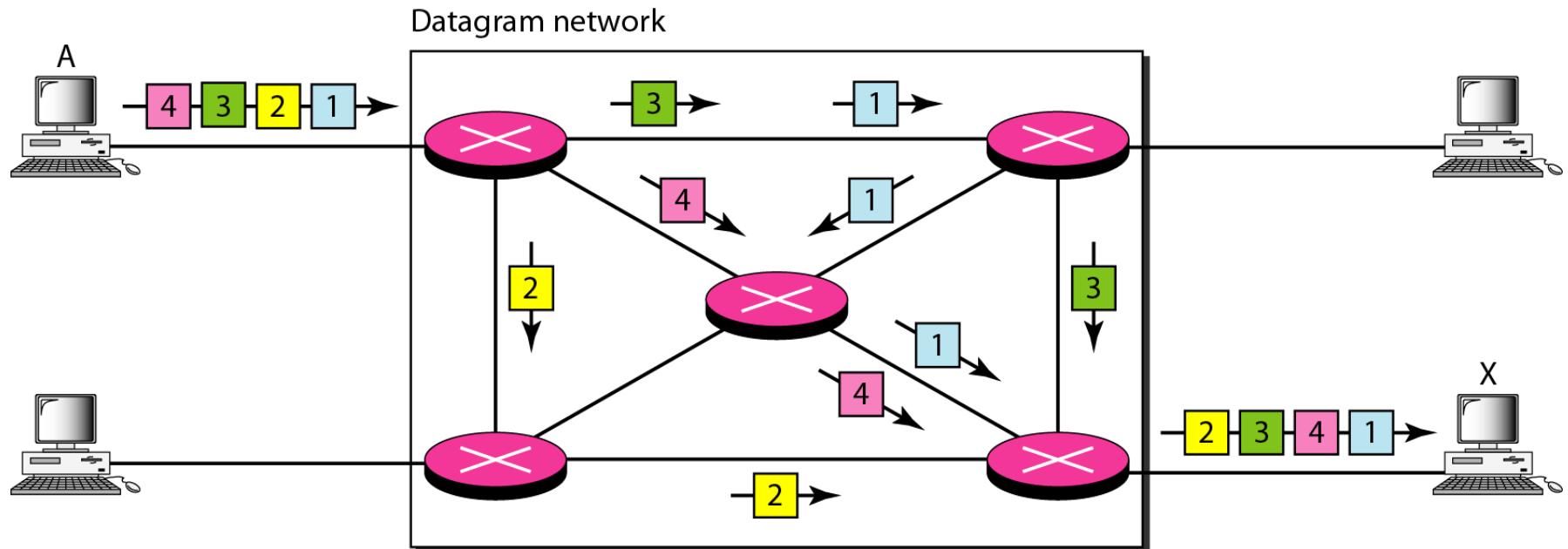
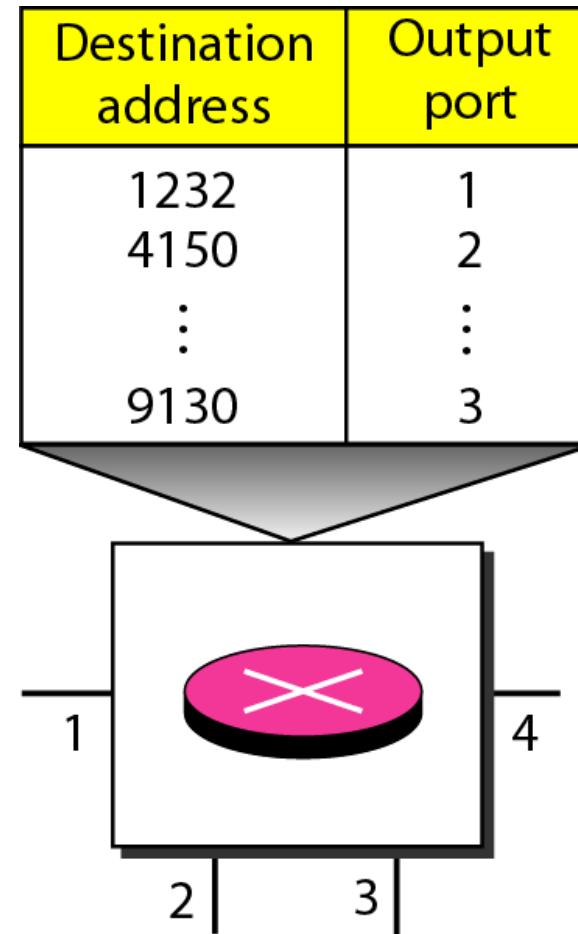
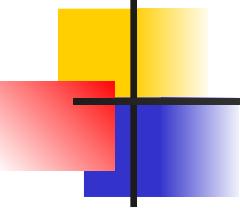


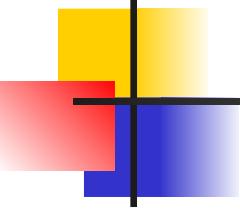
Figure 8.8 *Routing table in a datagram network*





Note

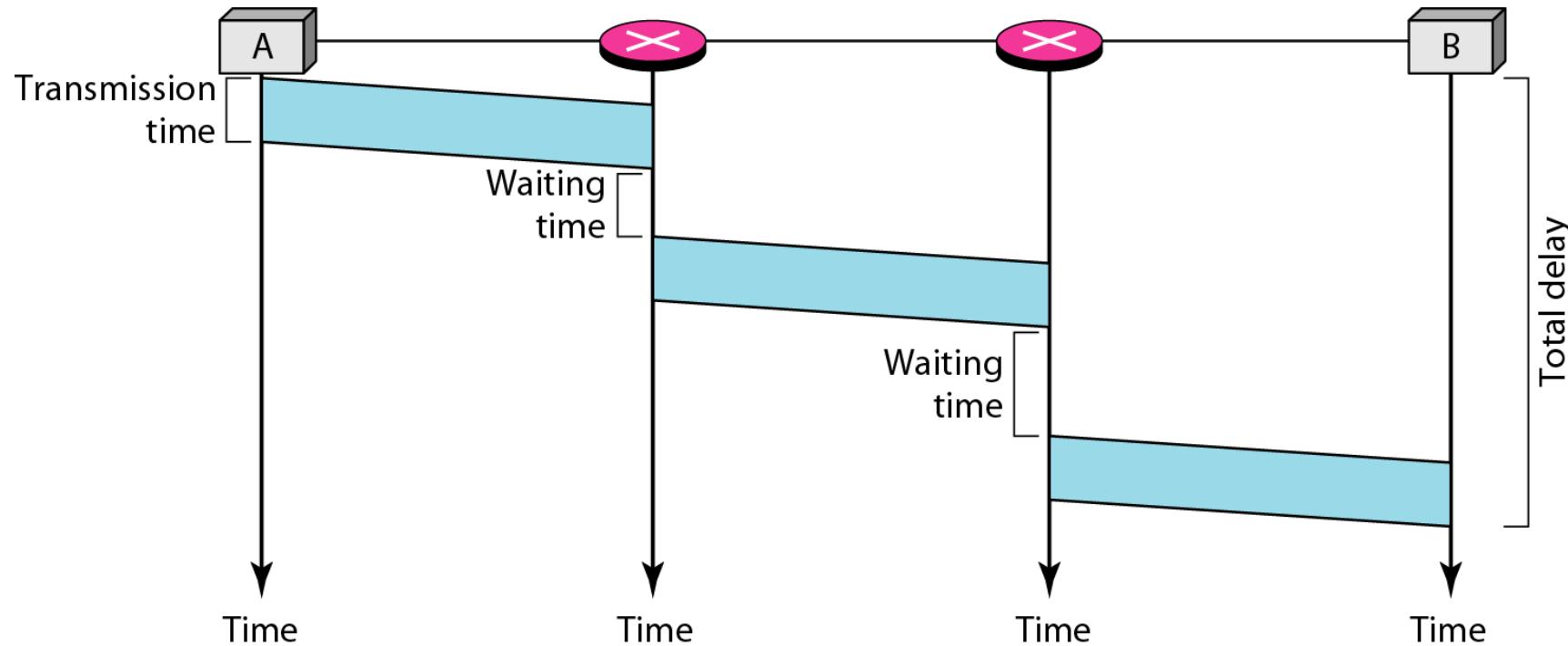
A switch in a datagram network uses a routing table that is based on the destination address.

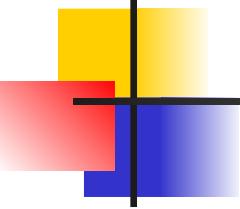


Note

The destination address in the header of a packet in a datagram network remains the same during the entire journey of the packet.

Figure 8.9 Delay in a datagram network





Note

Switching in the Internet is done by using the datagram approach to packet switching at the network layer.

8-3 VIRTUAL-CIRCUIT NETWORKS

A virtual-circuit network is a cross between a circuit-switched network and a datagram network. It has some characteristics of both.

Topics discussed in this section:

Addressing

Three Phases

Efficiency

Delay

Circuit-Switched Technology in WANs

Figure 8.10 Virtual-circuit network

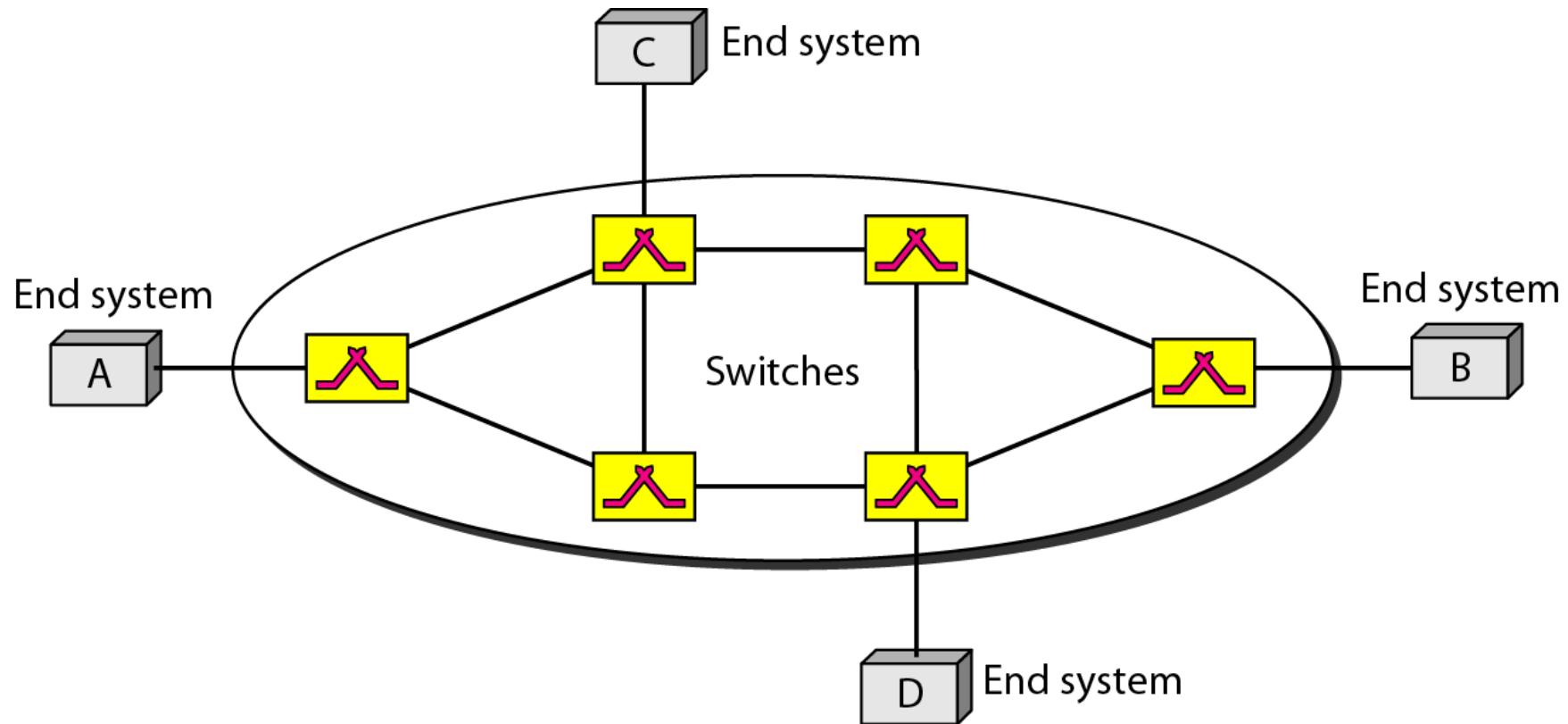


Figure 8.11 Virtual-circuit identifier

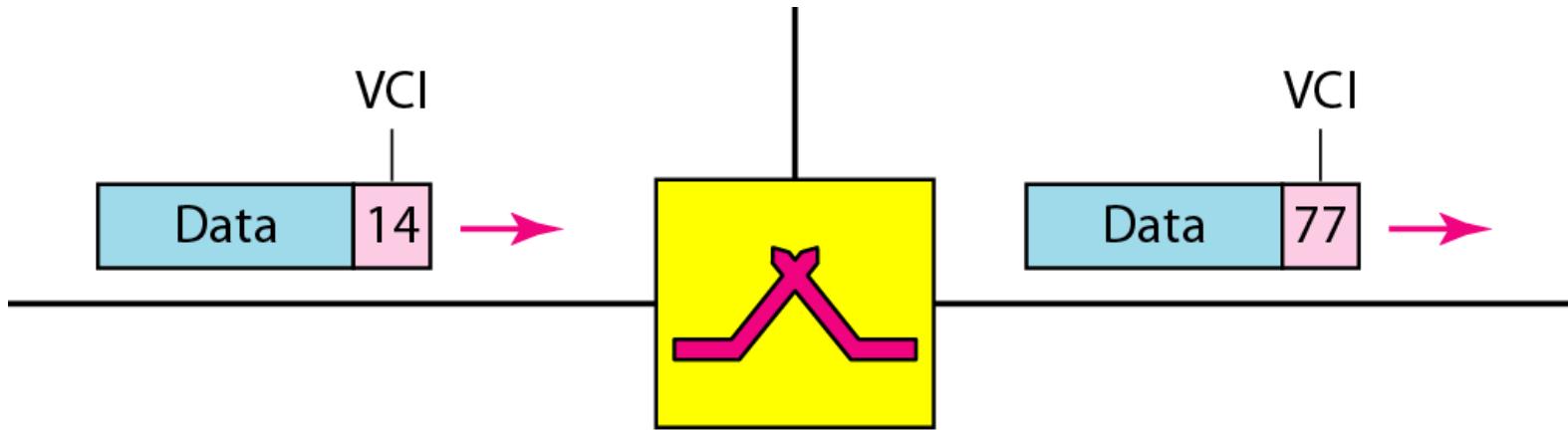


Figure 8.12 Switch and tables in a virtual-circuit network

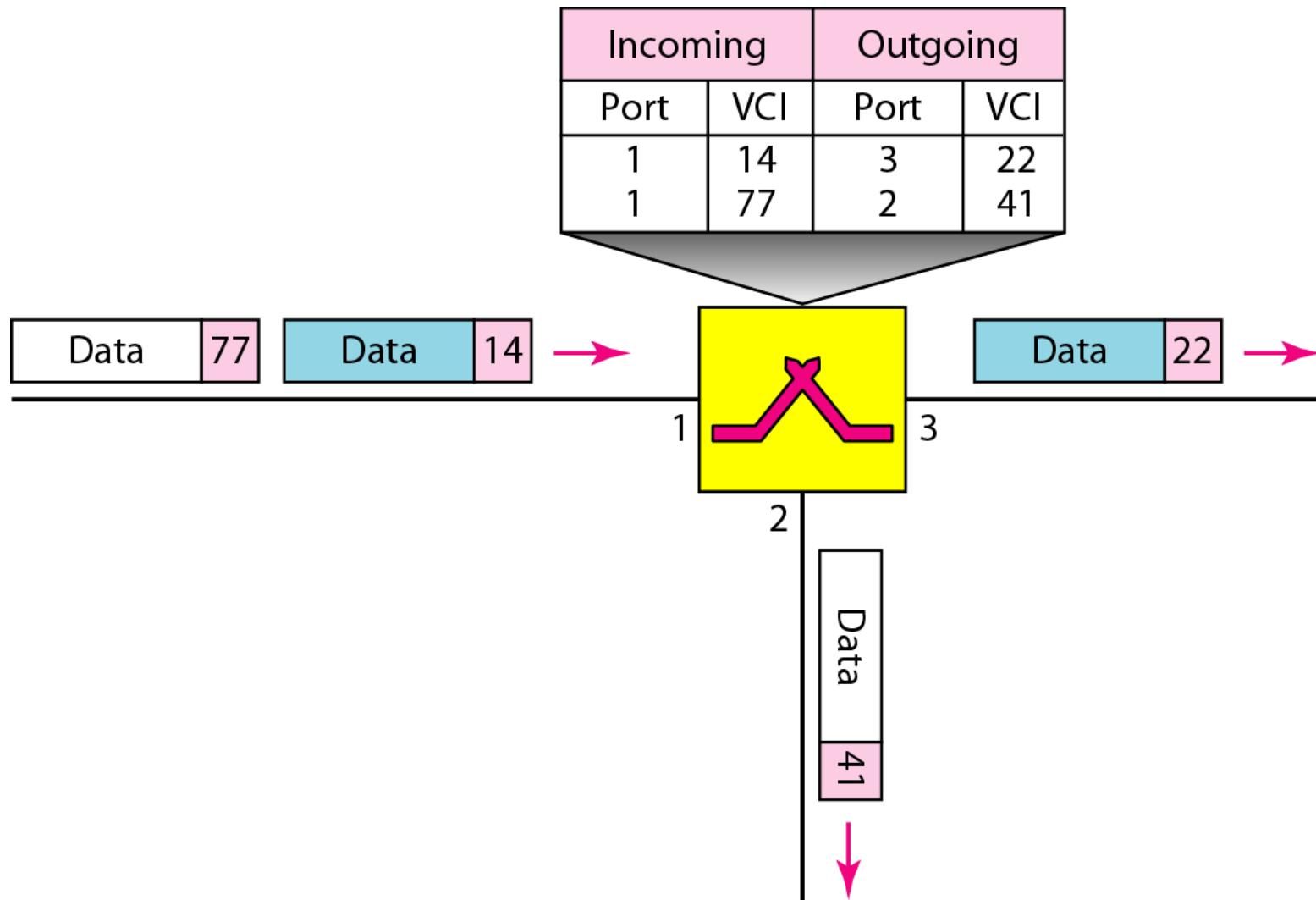


Figure 8.13 Source-to-destination data transfer in a virtual-circuit network

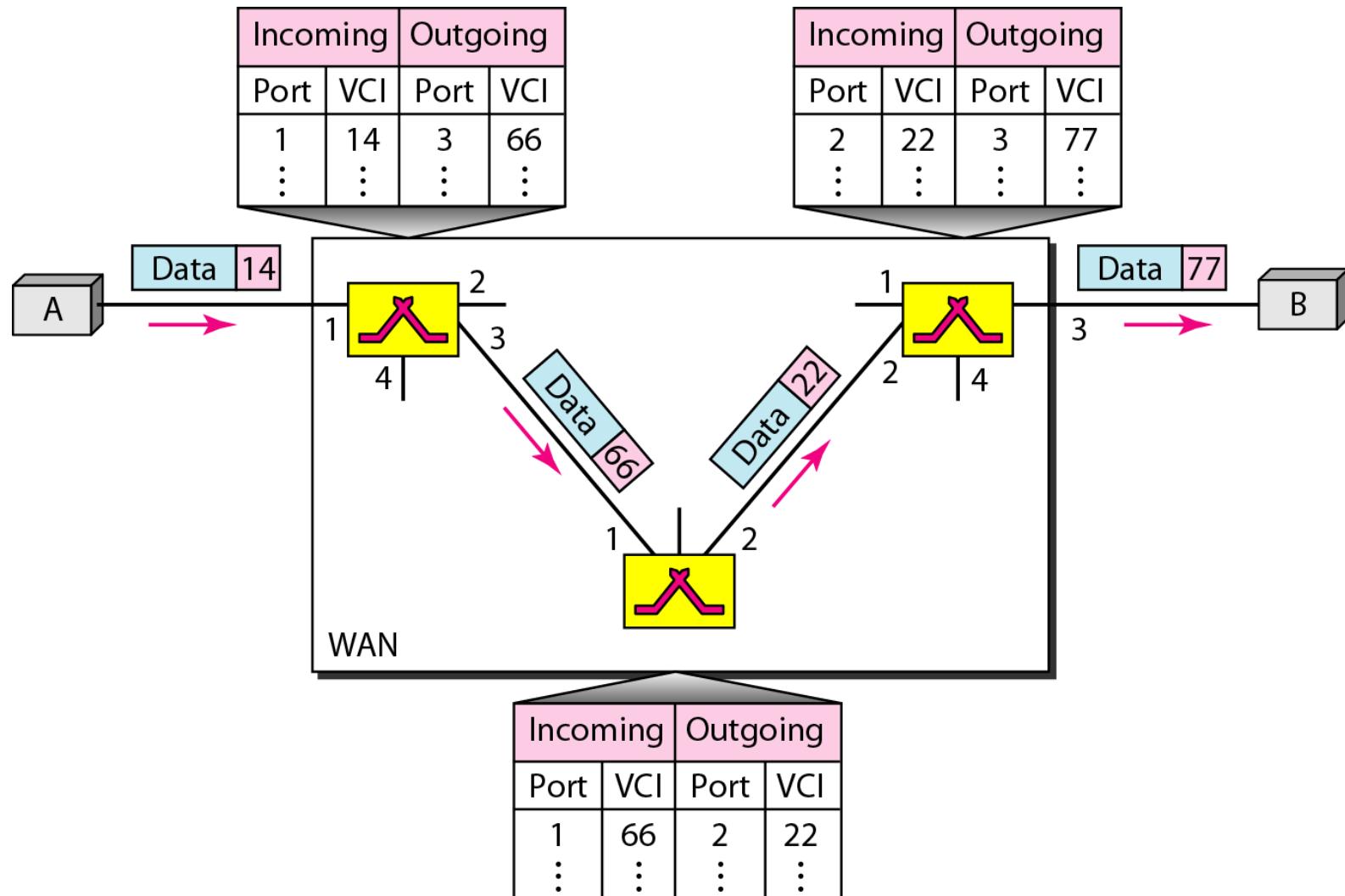


Figure 8.14 Setup request in a virtual-circuit network

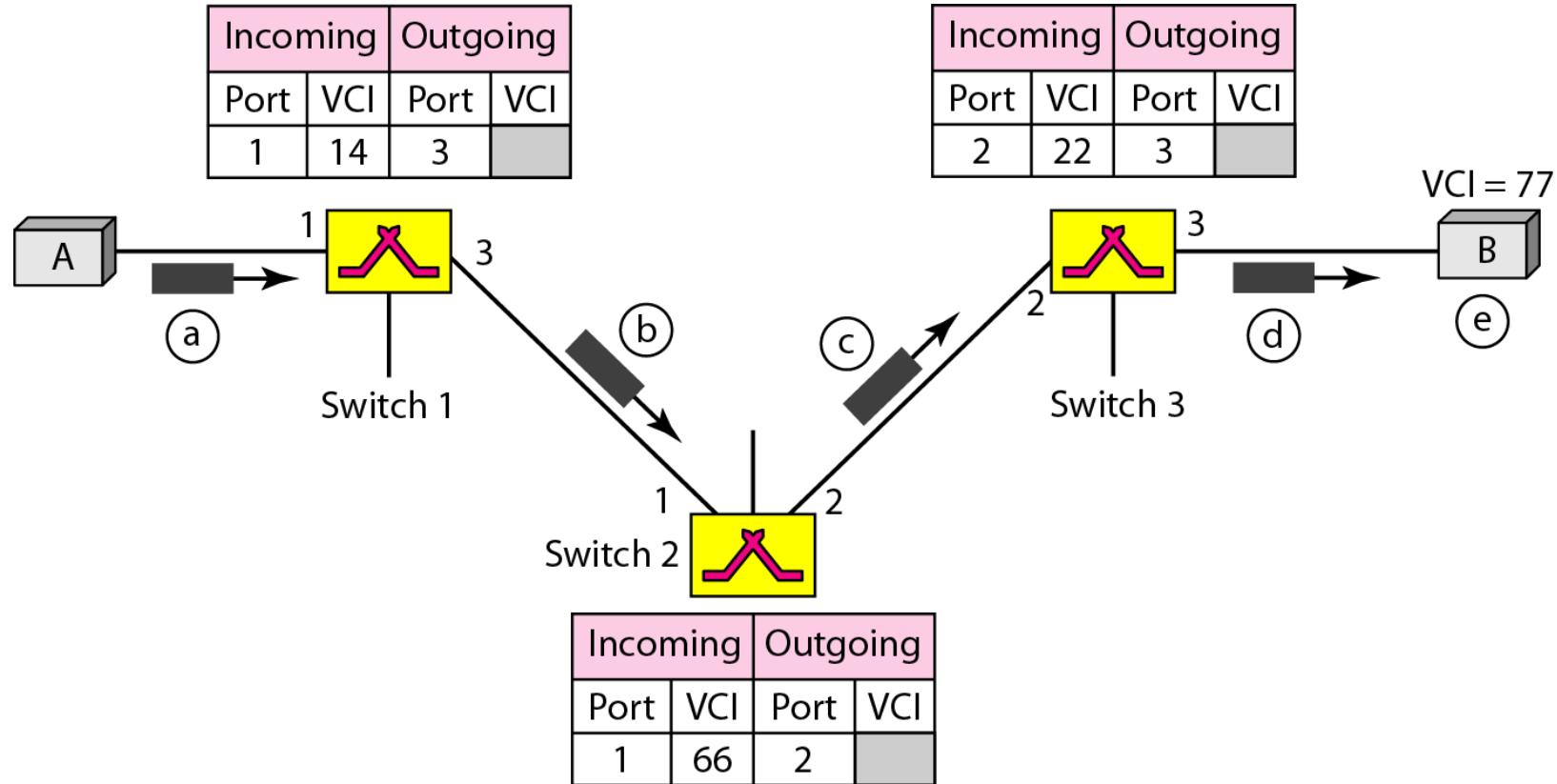
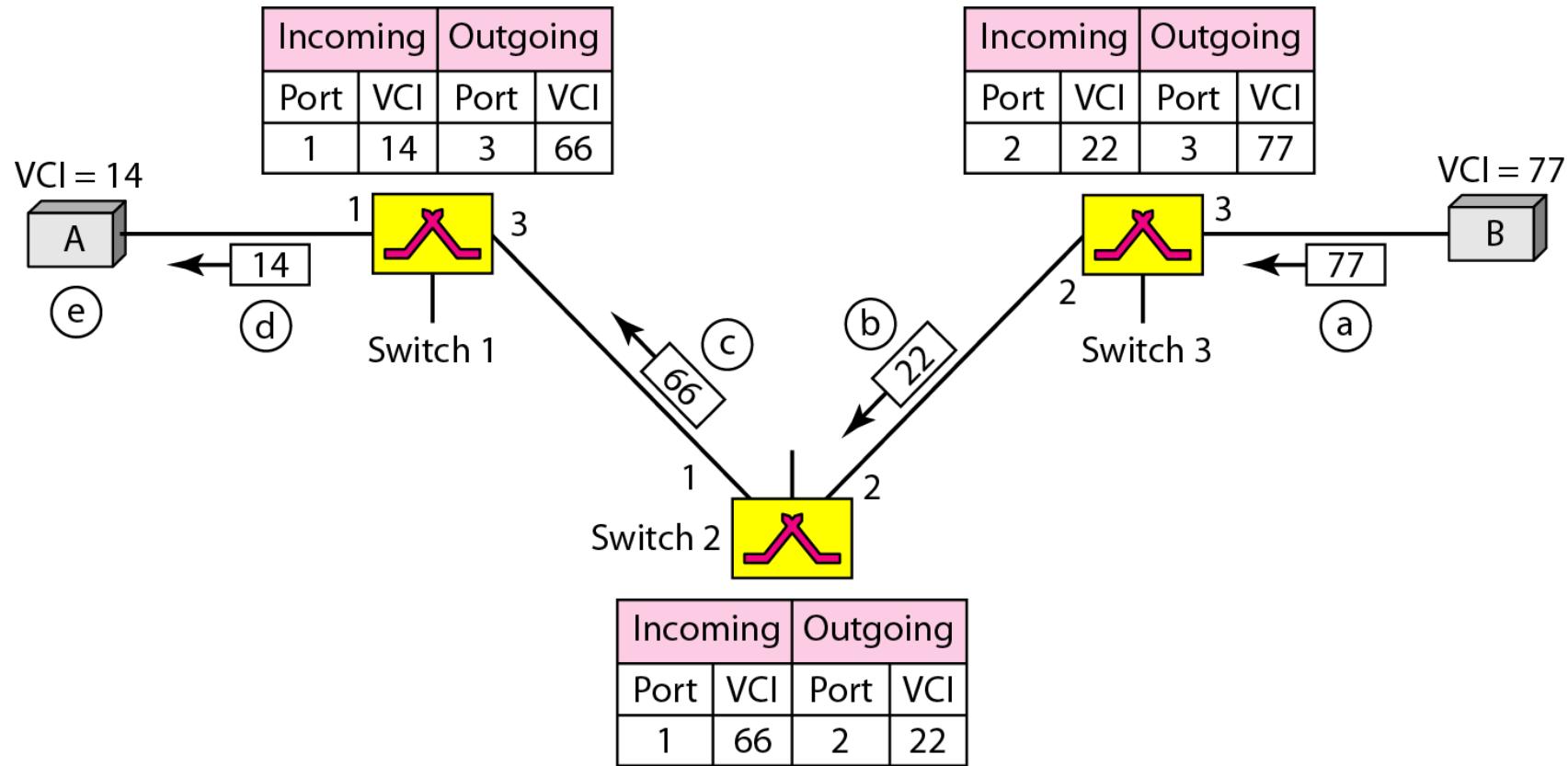
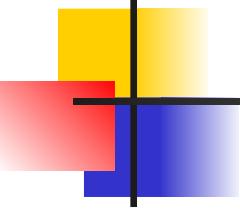


Figure 8.15 Setup acknowledgment in a virtual-circuit network

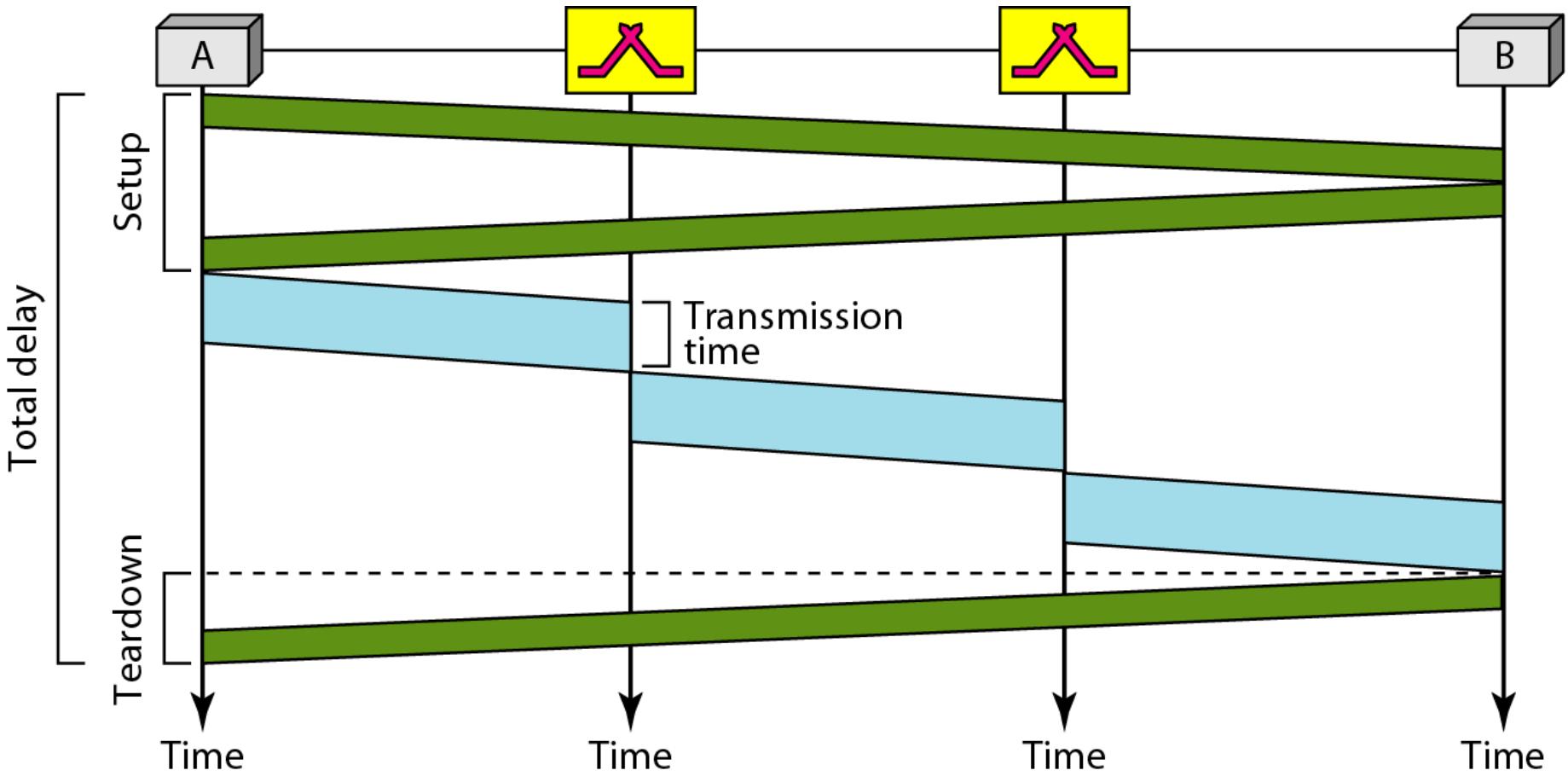


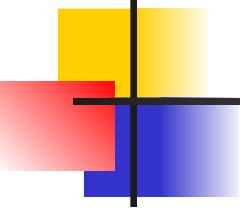


Note

In virtual-circuit switching, all packets belonging to the same source and destination travel the same path; but the packets may arrive at the destination with different delays if resource allocation is on demand.

Figure 8.16 Delay in a virtual-circuit network





Note

Switching at the data link layer in a switched WAN is normally implemented by using virtual-circuit techniques.

8-4 STRUCTURE OF A SWITCH

We use switches in circuit-switched and packet-switched networks. In this section, we discuss the structures of the switches used in each type of network.

Topics discussed in this section:

Structure of Circuit Switches

Structure of Packet Switches

Figure 8.17 Crossbar switch with three inputs and four outputs

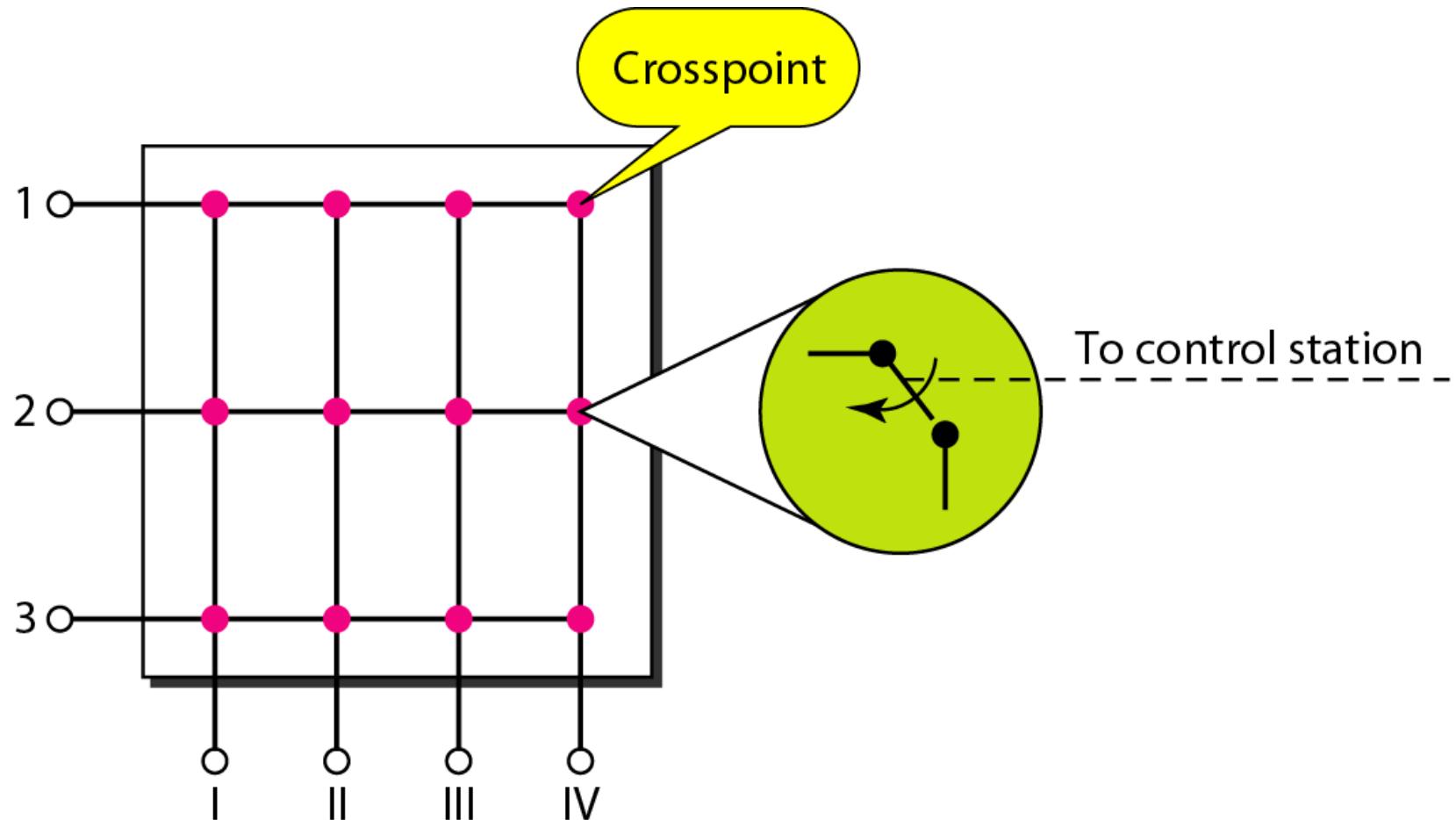
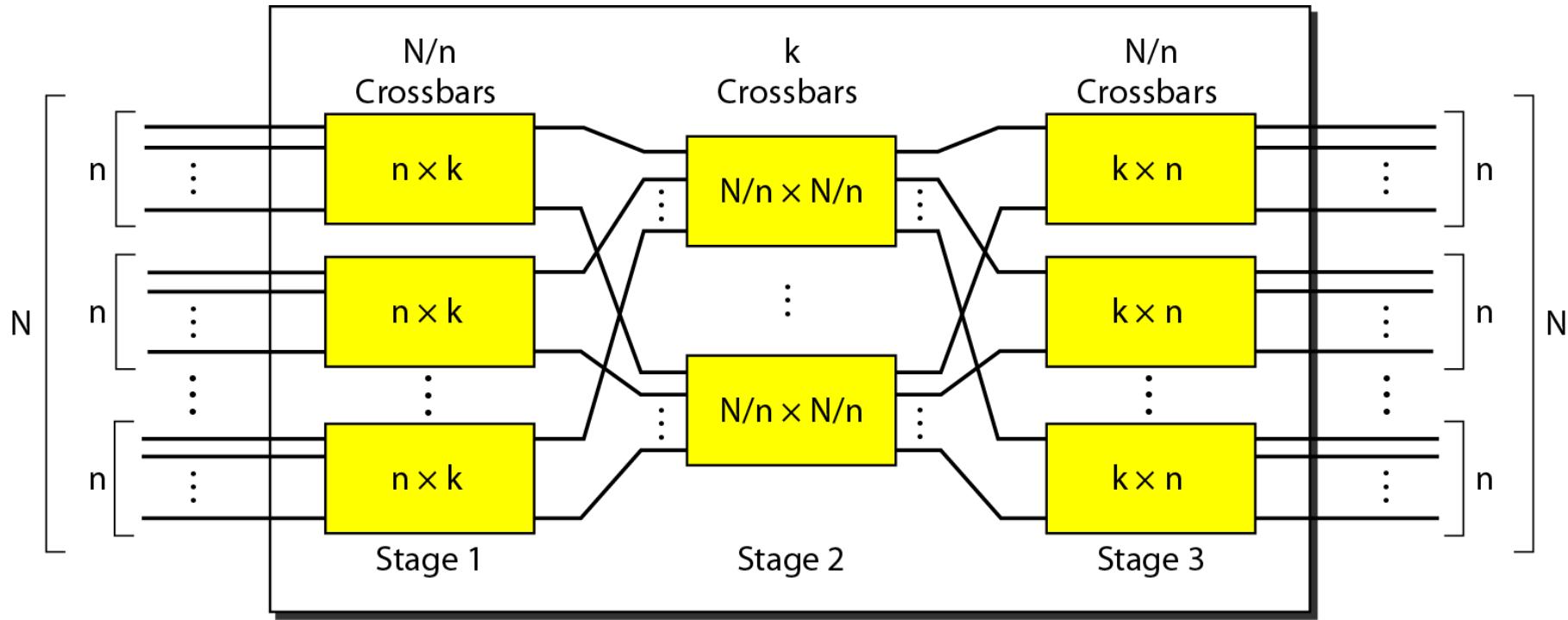
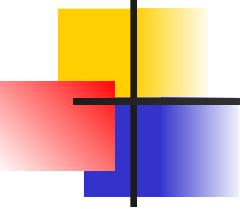


Figure 8.18 Multistage switch



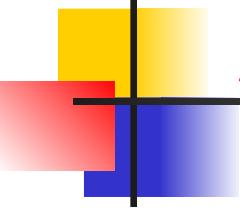


Note

In a three-stage switch, the total number of crosspoints is

$$2kN + k(N/n)^2$$

which is much smaller than the number of crosspoints in a single-stage switch (N^2).

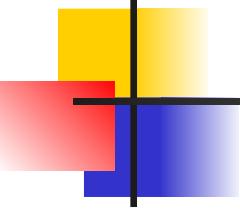


Example 8.3

Design a three-stage, 200×200 switch ($N = 200$) with $k = 4$ and $n = 20$.

Solution

*In the first stage we have N/n or 10 crossbars, each of size 20×4 . In the second stage, we have 4 crossbars, each of size 10×10 . In the third stage, we have 10 crossbars, each of size 4×20 . The total number of crosspoints is $2kN + k(N/n)^2$, or **2000** crosspoints. This is 5 percent of the number of crosspoints in a single-stage switch ($200 \times 200 = 40,000$).*



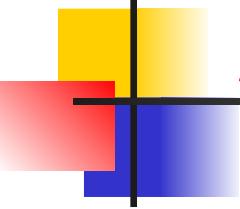
Note

According to the Clos criterion:

$$n = (N/2)^{1/2}$$

$$k > 2n - 1$$

$$\text{Crosspoints} \geq 4N [(2N)^{1/2} - 1]$$



Example 8.4

Redesign the previous three-stage, 200×200 switch, using the Clos criteria with a minimum number of crosspoints.

Solution

We let $n = (200/2)^{1/2}$, or $n = 10$. We calculate $k = 2n - 1 = 19$. In the first stage, we have $200/10$, or 20, crossbars, each with 10×19 crosspoints. In the second stage, we have 19 crossbars, each with 10×10 crosspoints. In the third stage, we have 20 crossbars each with 19×10 crosspoints. The total number of crosspoints is $20(10 \times 19) + 19(10 \times 10) + 20(19 \times 10) = 9500$.

Figure 8.19 Time-slot interchange

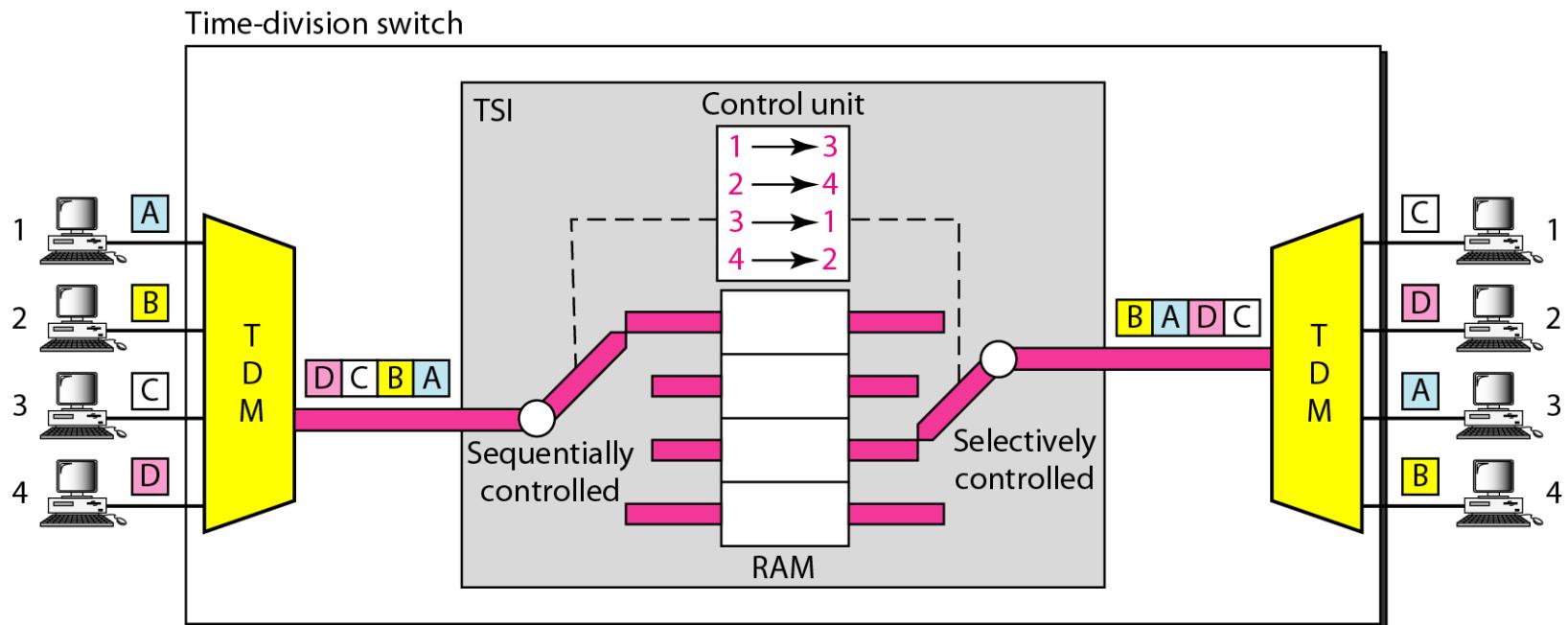


Figure 8.20 *Time-space-time switch*

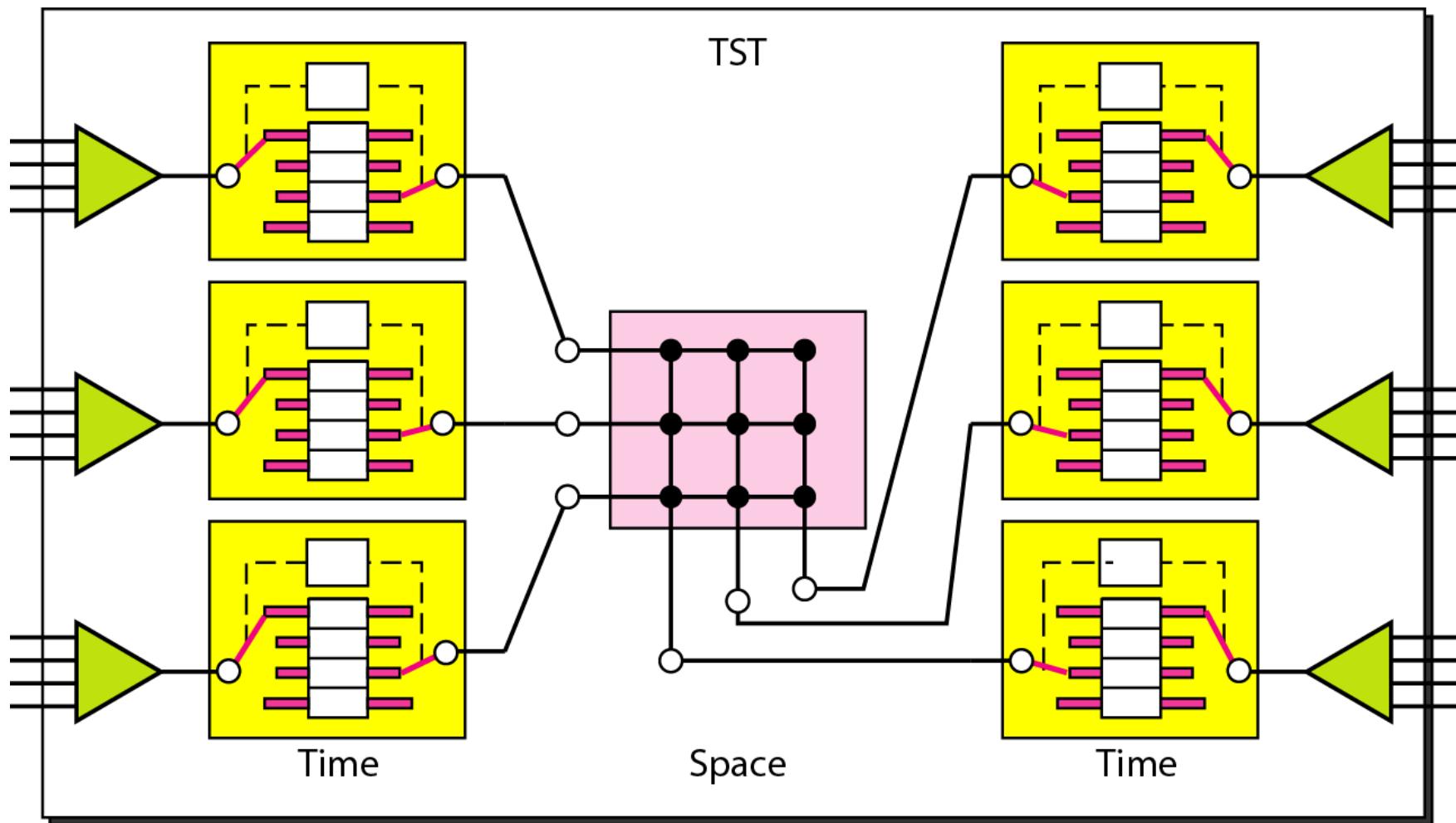


Figure 8.21 *Packet switch components*

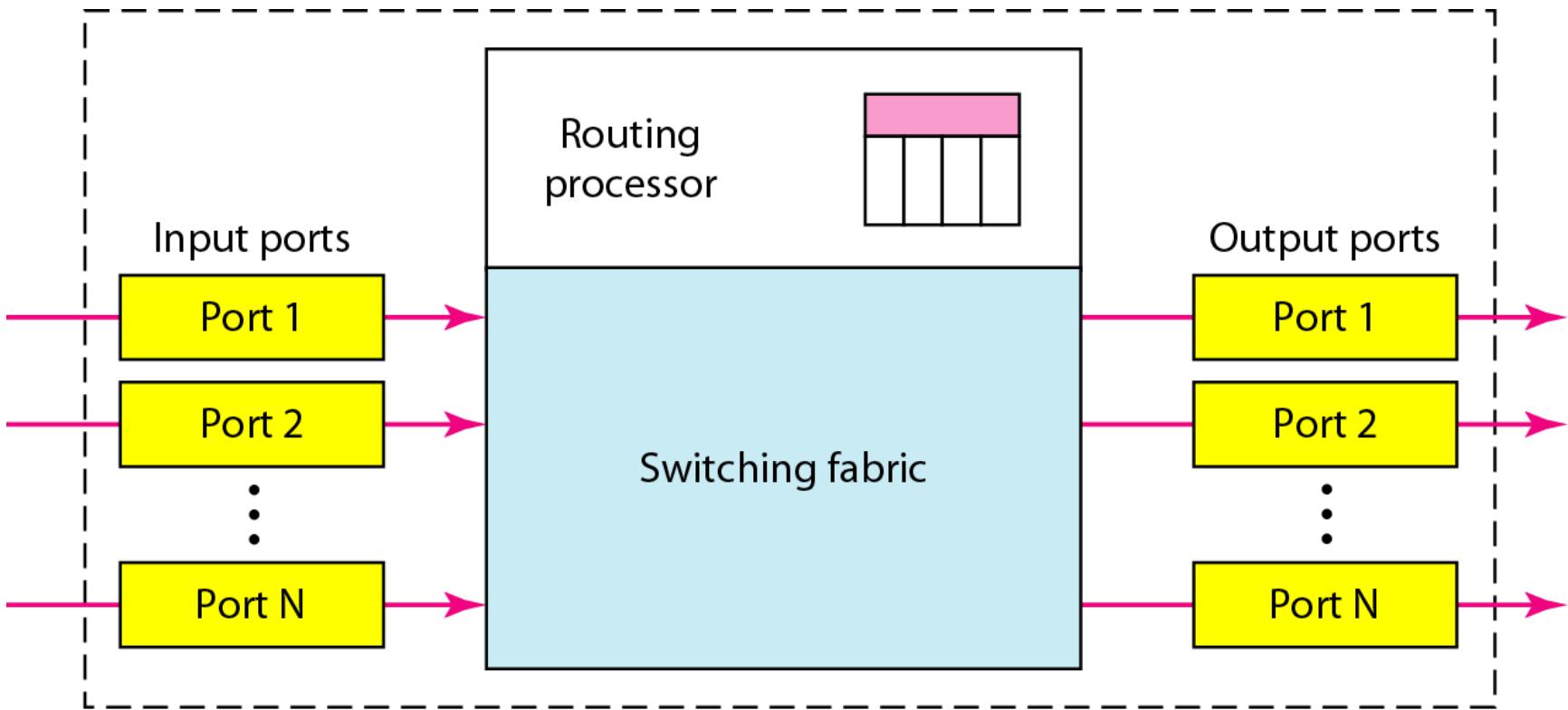


Figure 8.22 *Input port*

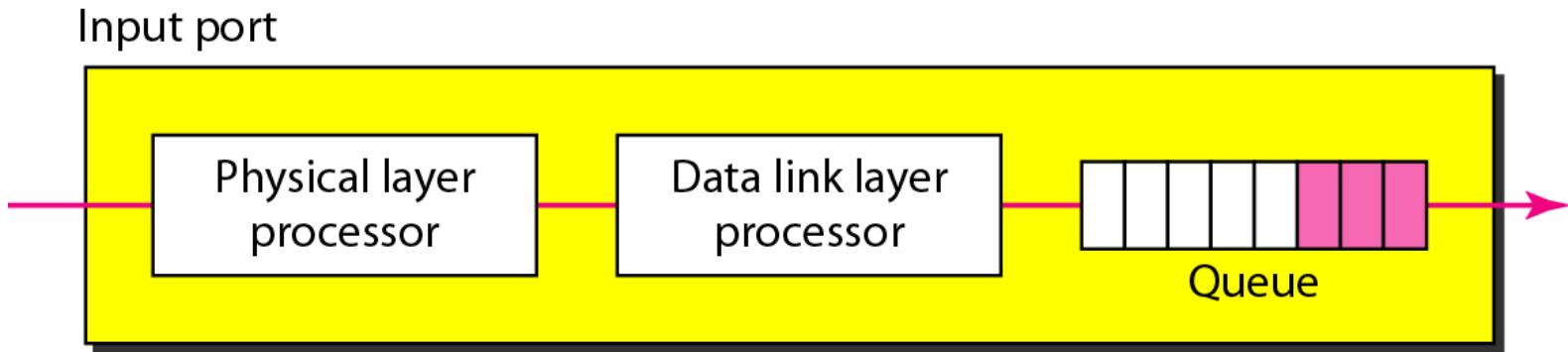


Figure 8.23 *Output port*

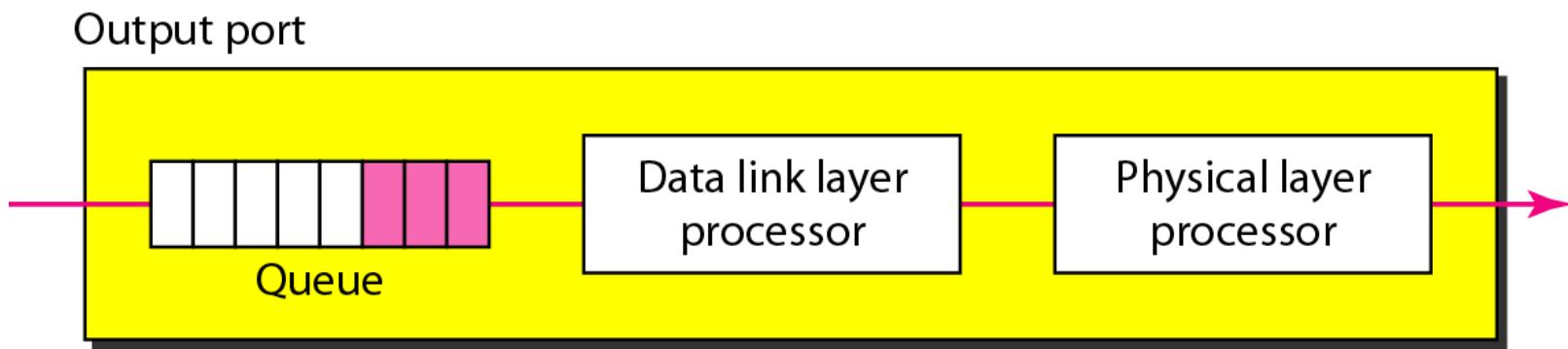


Figure 8.24 A banyan switch

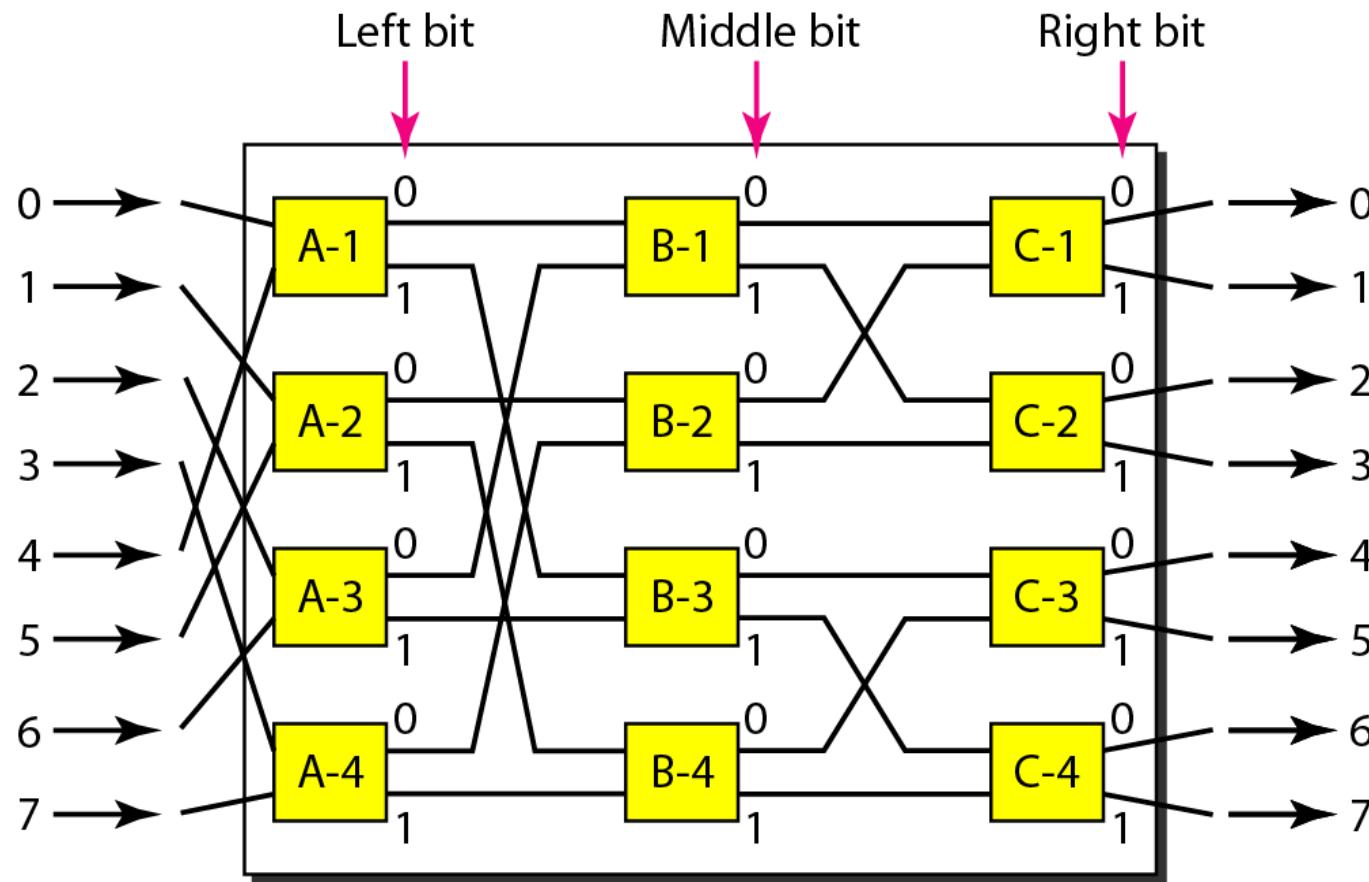
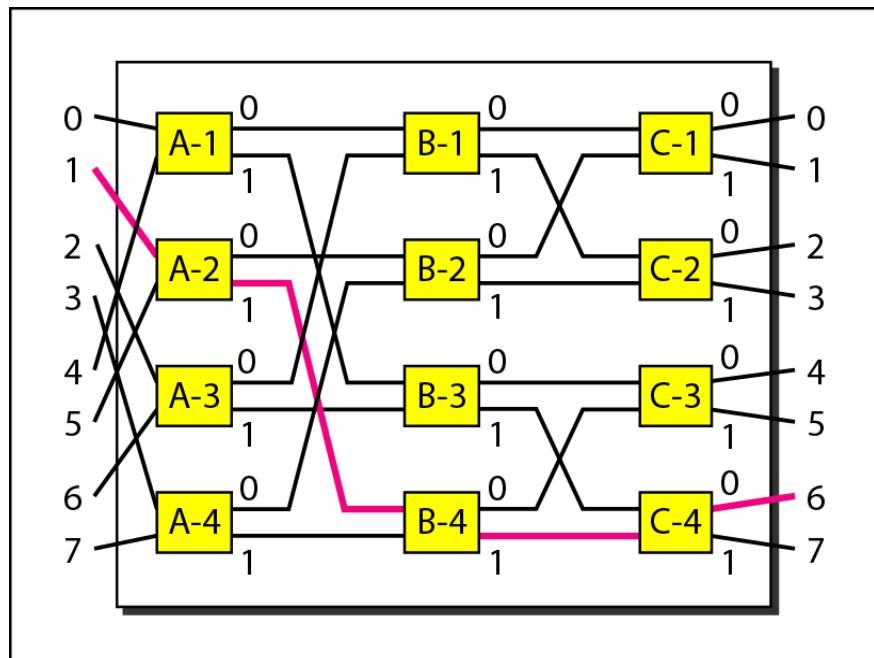
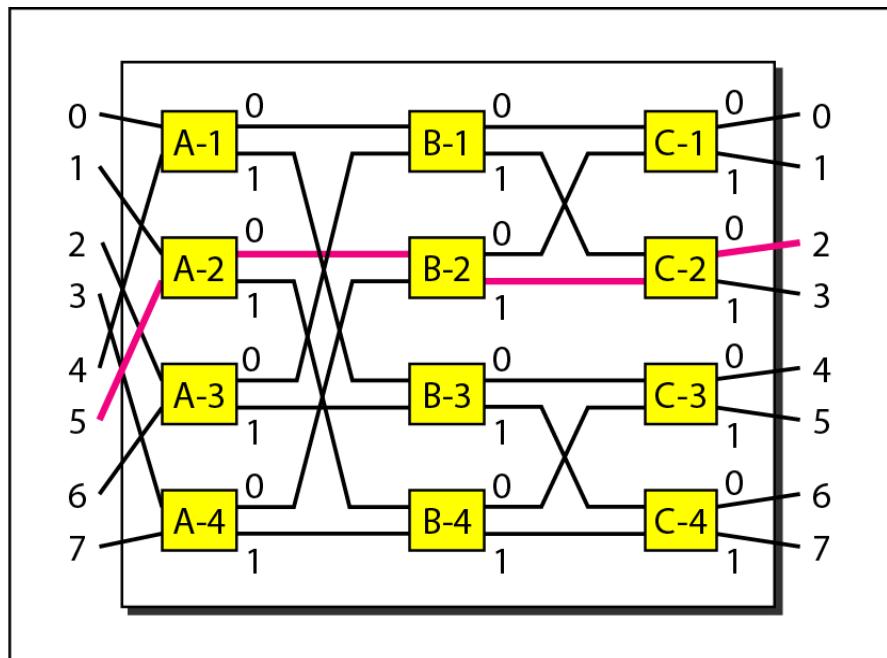


Figure 8.25 Examples of routing in a banyan switch

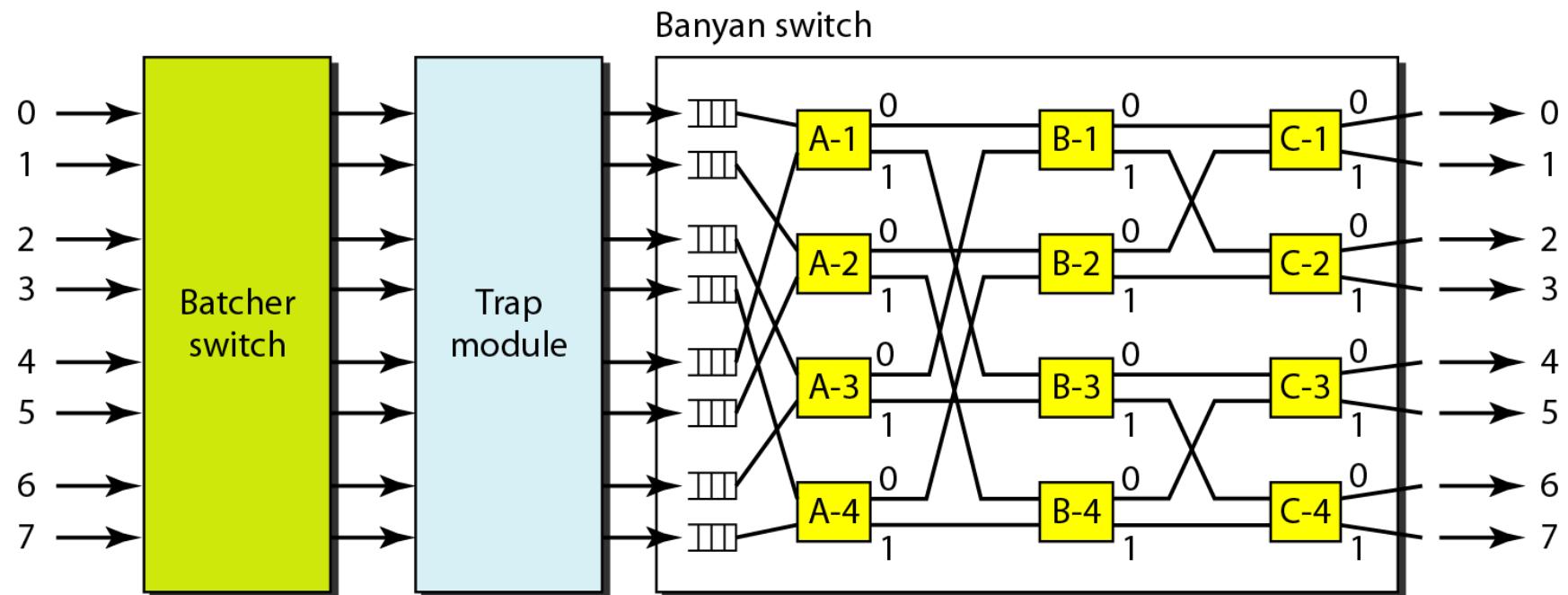


a. Input 1 sending a cell to output 6 (110)



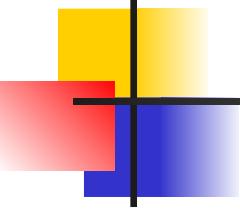
b. Input 5 sending a cell to output 2 (010)

Figure 8.26 *Batcher-banyan switch*



Chapter 10

Error Detection and Correction



Note

**Data can be corrupted
during transmission.**

**Some applications require that
errors be detected and corrected.**

10-1 INTRODUCTION

Let us first discuss some issues related, directly or indirectly, to error detection and correction.

Topics discussed in this section:

Types of Errors

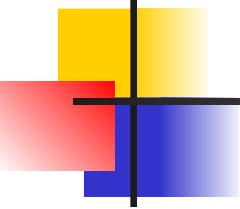
Redundancy

Detection Versus Correction

Forward Error Correction Versus Retransmission

Coding

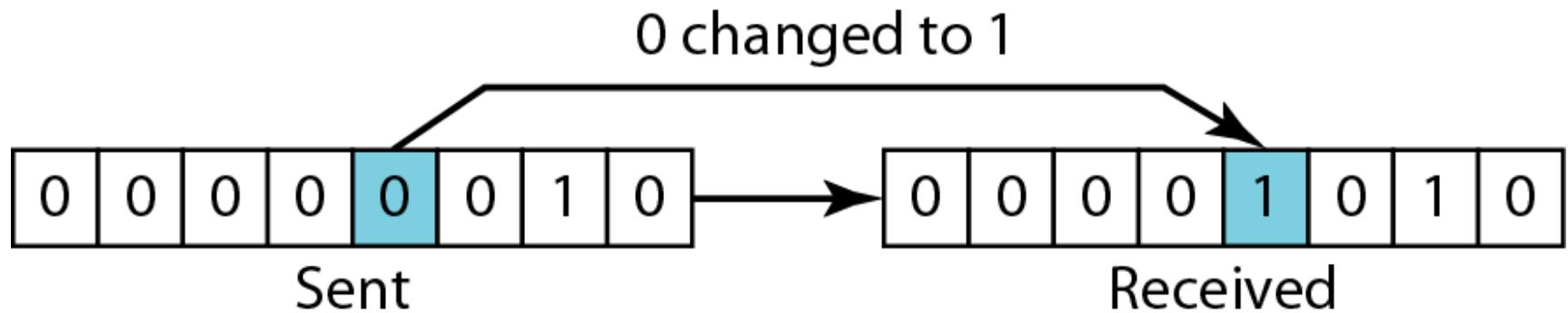
Modular Arithmetic

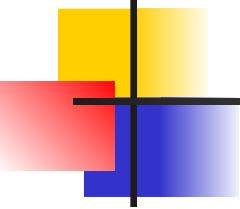


Note

In a single-bit error, only 1 bit in the data unit has changed.

Figure 10.1 Single-bit error

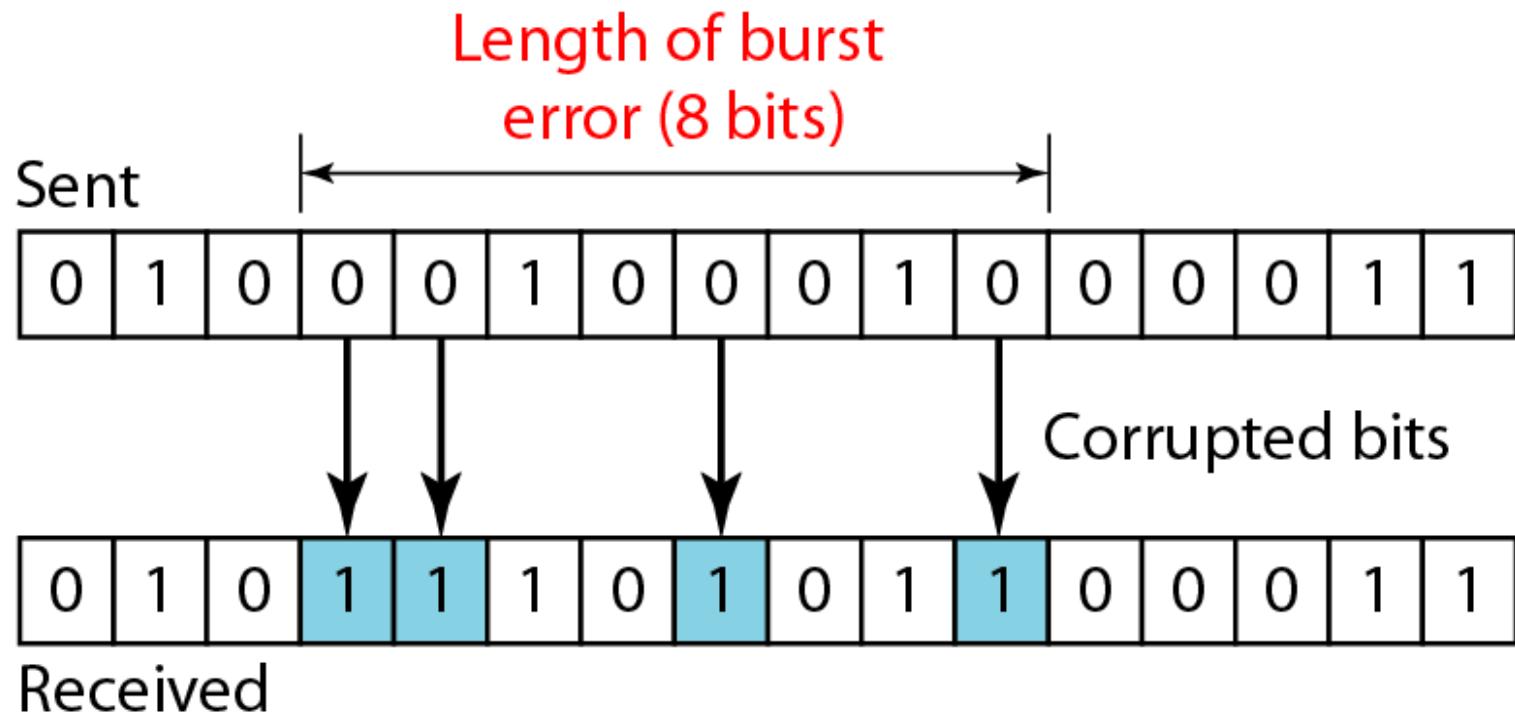


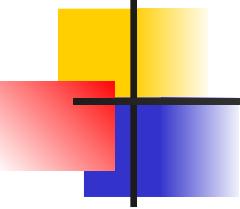


Note

A burst error means that 2 or more bits in the data unit have changed.

Figure 10.2 *Burst error of length 8*

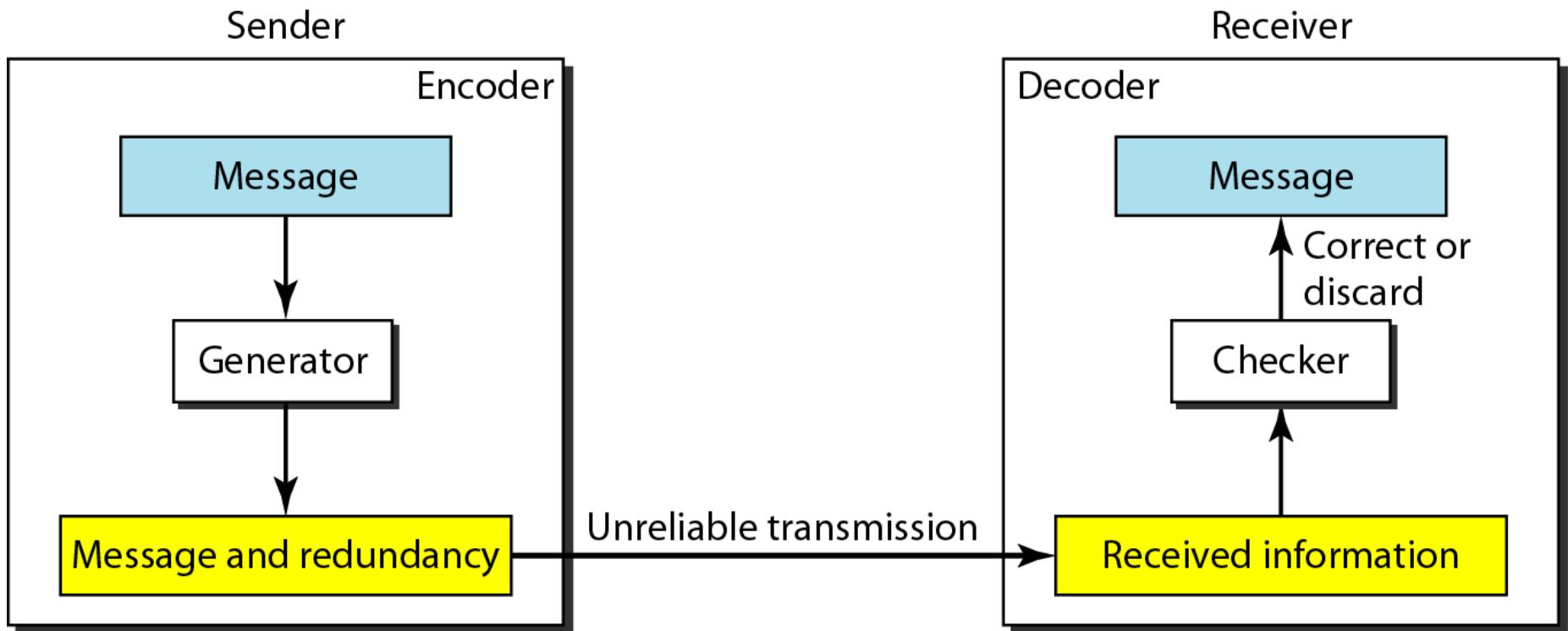


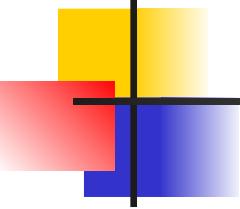


Note

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

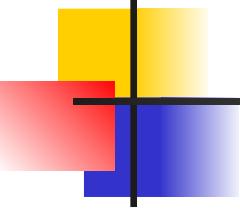
Figure 10.3 *The structure of encoder and decoder*





Note

In this book, we concentrate on block codes; we leave convolution codes to advanced texts.



Note

In modulo-N arithmetic, we use only the integers in the range 0 to $N - 1$, inclusive.

Figure 10.4 XORing of two single bits or two words

$$0 \oplus 0 = 0$$

$$1 \oplus 1 = 0$$

a. Two bits are the same, the result is 0.

$$0 \oplus 1 = 1$$

$$1 \oplus 0 = 1$$

b. Two bits are different, the result is 1.

$$\begin{array}{r} 1 & 0 & 1 & 1 & 0 \\ + & 1 & 1 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 & 0 \end{array}$$

c. Result of XORing two patterns

10-2 BLOCK CODING

*In block coding, we divide our message into blocks, each of k bits, called **datawords**. We add r redundant bits to each block to make the length $n = k + r$. The resulting n -bit blocks are called **codewords**.*

Topics discussed in this section:

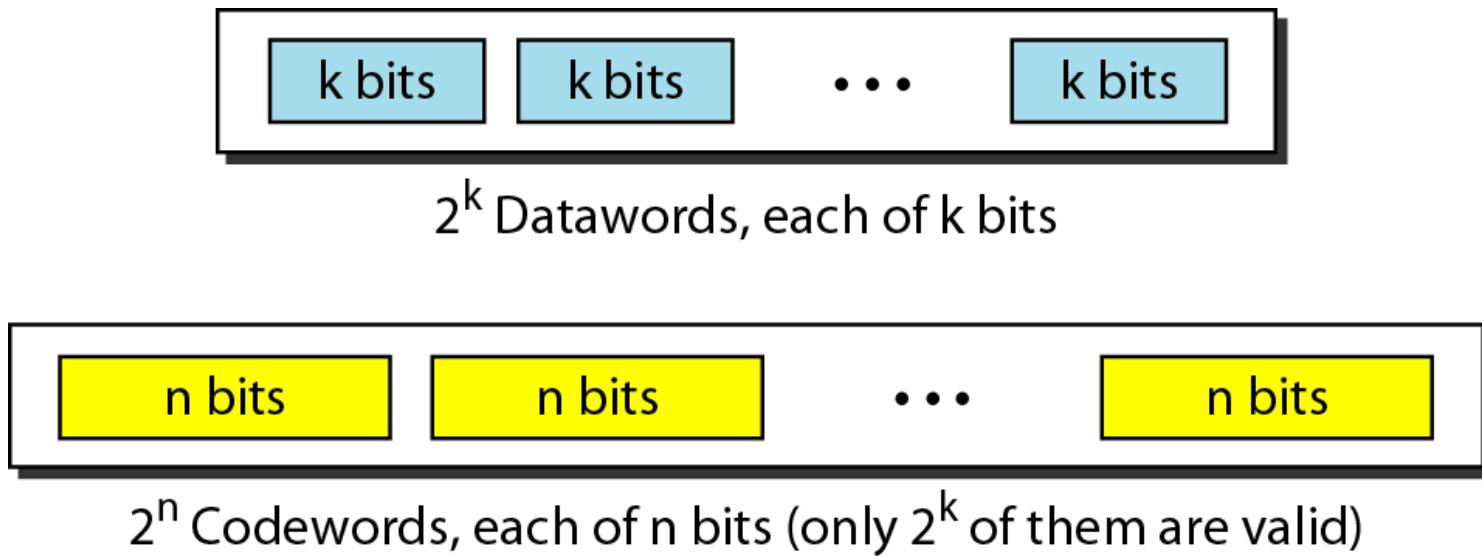
Error Detection

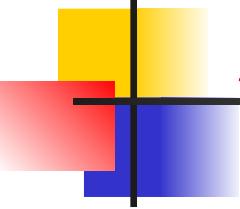
Error Correction

Hamming Distance

Minimum Hamming Distance

Figure 10.5 *Datawords and codewords in block coding*

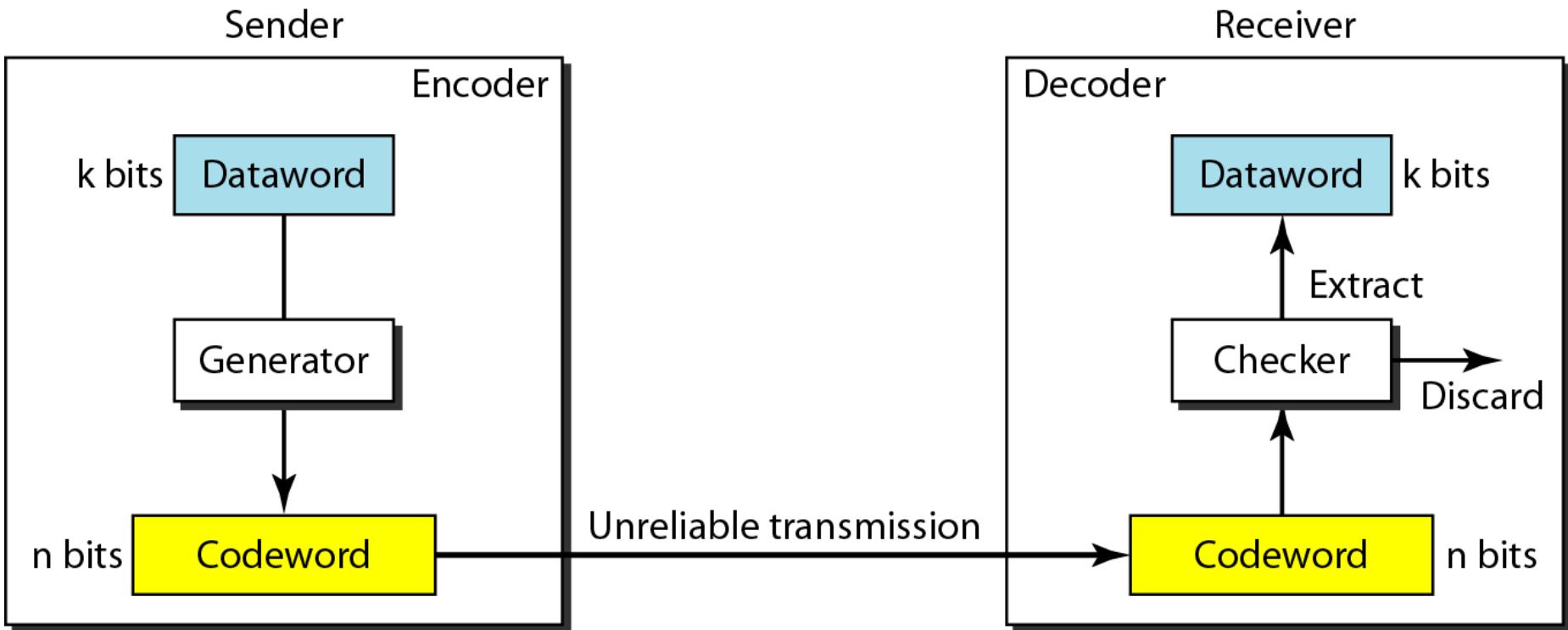


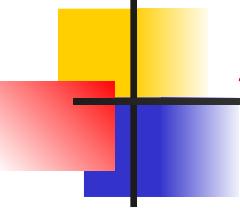


Example 10.1

The 4B/5B block coding discussed in Chapter 4 is a good example of this type of coding. In this coding scheme, $k = 4$ and $n = 5$. As we saw, we have $2^k = 16$ datawords and $2^n = 32$ codewords. We saw that 16 out of 32 codewords are used for message transfer and the rest are either used for other purposes or unused.

Figure 10.6 *Process of error detection in block coding*



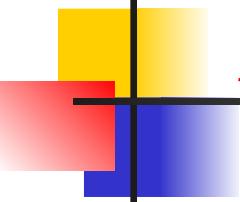


Example 10.2

Let us assume that $k = 2$ and $n = 3$. Table 10.1 shows the list of datawords and codewords. Later, we will see how to derive a codeword from a dataword.

Assume the sender encodes the dataword 01 as 011 and sends it to the receiver. Consider the following cases:

- 1. The receiver receives 011. It is a valid codeword. The receiver extracts the dataword 01 from it.*



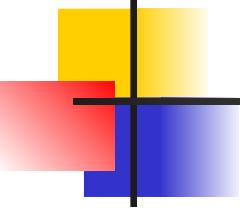
Example 10.2 (continued)

- 2.** *The codeword is corrupted during transmission, and 111 is received. This is not a valid codeword and is discarded.*

- 3.** *The codeword is corrupted during transmission, and 000 is received. This is a valid codeword. The receiver incorrectly extracts the dataword 00. Two corrupted bits have made the error undetectable.*

Table 10.1 A code for error detection (Example 10.2)

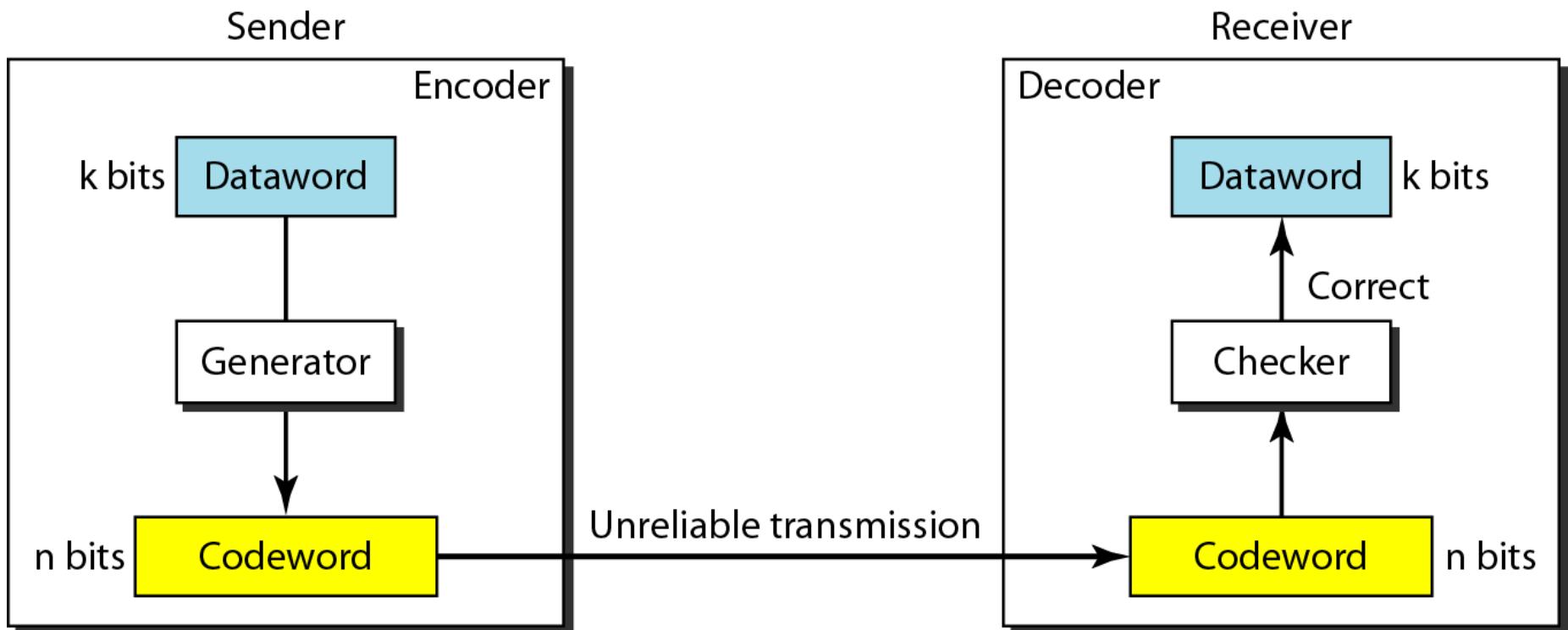
<i>Datawords</i>	<i>Codewords</i>
00	000
01	011
10	101
11	110

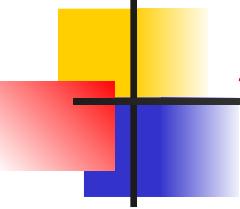


Note

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

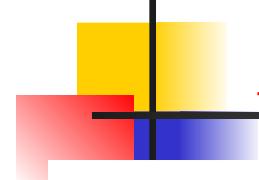
Figure 10.7 Structure of encoder and decoder in error correction





Example 10.3

Let us add more redundant bits to Example 10.2 to see if the receiver can correct an error without knowing what was actually sent. We add 3 redundant bits to the 2-bit dataword to make 5-bit codewords. Table 10.2 shows the datawords and codewords. Assume the dataword is 01. The sender creates the codeword 01011. The codeword is corrupted during transmission, and 01001 is received. First, the receiver finds that the received codeword is not in the table. This means an error has occurred. The receiver, assuming that there is only 1 bit corrupted, uses the following strategy to guess the correct dataword.

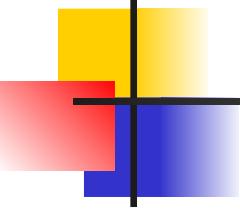


Example 10.3 (continued)

- 1.** *Comparing the received codeword with the first codeword in the table (01001 versus 00000), the receiver decides that the first codeword is not the one that was sent because there are two different bits.*
- 2.** *By the same reasoning, the original codeword cannot be the third or fourth one in the table.*
- 3.** *The original codeword must be the second one in the table because this is the only one that differs from the received codeword by 1 bit. The receiver replaces 01001 with 01011 and consults the table to find the dataword 01.*

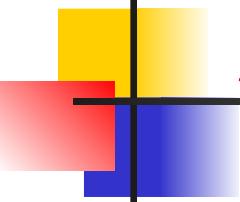
Table 10.2 A code for error correction (Example 10.3)

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



Note

The Hamming distance between two words is the number of differences between corresponding bits.



Example 10.4

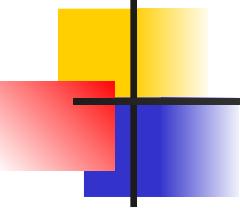
Let us find the Hamming distance between two pairs of words.

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

$$000 \oplus 011 \text{ is } 011 \text{ (two 1s)}$$

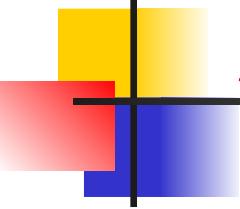
2. The Hamming distance $d(10101, 11110)$ is 3 because

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



Note

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



Example 10.5

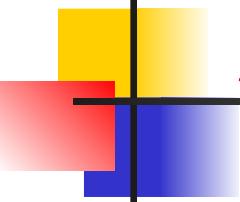
Find the minimum Hamming distance of the coding scheme in Table 10.1.

Solution

We first find all Hamming distances.

$$\begin{array}{llll} d(000, 011) = 2 & d(000, 101) = 2 & d(000, 110) = 2 & d(011, 101) = 2 \\ d(011, 110) = 2 & d(101, 110) = 2 & & \end{array}$$

The d_{min} in this case is 2.



Example 10.6

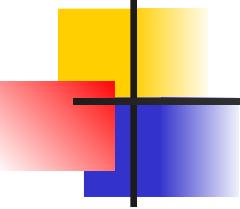
Find the minimum Hamming distance of the coding scheme in Table 10.2.

Solution

We first find all the Hamming distances.

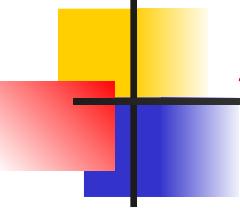
$$\begin{array}{lll} d(00000, 01011) = 3 & d(00000, 10101) = 3 & d(00000, 11110) = 4 \\ d(01011, 10101) = 4 & d(01011, 11110) = 3 & d(10101, 11110) = 3 \end{array}$$

The d_{min} in this case is 3.



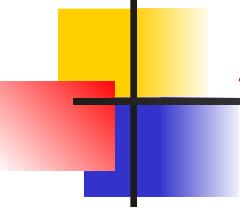
Note

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



Example 10.7

The minimum Hamming distance for our first code scheme (Table 10.1) is 2. This code guarantees detection of only a single error. For example, if the third codeword (101) is sent and one error occurs, the received codeword does not match any valid codeword. If two errors occur, however, the received codeword may match a valid codeword and the errors are not detected.



Example 10.8

Our second block code scheme (Table 10.2) has $d_{min} = 3$. This code can detect up to two errors. Again, we see that when any of the valid codewords is sent, two errors create a codeword which is not in the table of valid codewords. The receiver cannot be fooled.

However, some combinations of three errors change a valid codeword to another valid codeword. The receiver accepts the received codeword and the errors are undetected.

Figure 10.8 Geometric concept for finding d_{min} in error detection

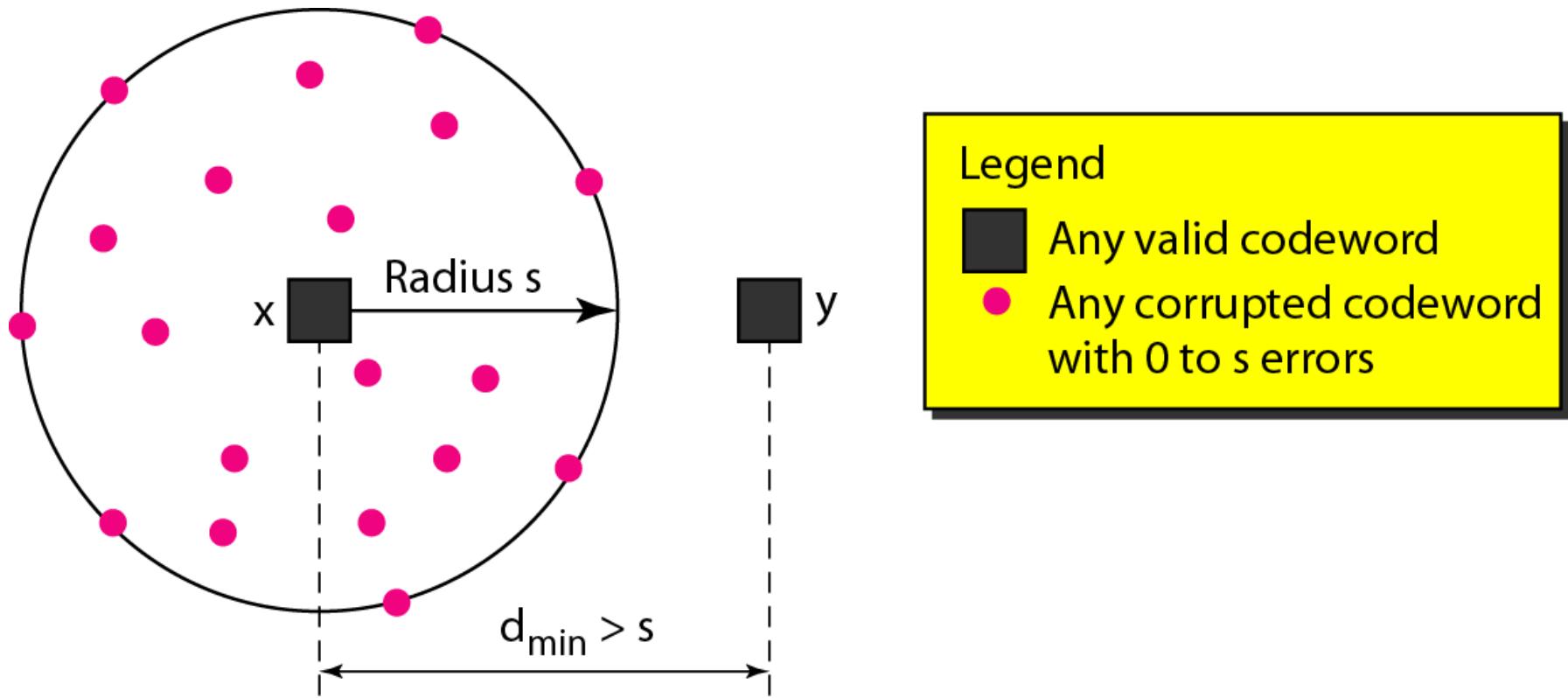
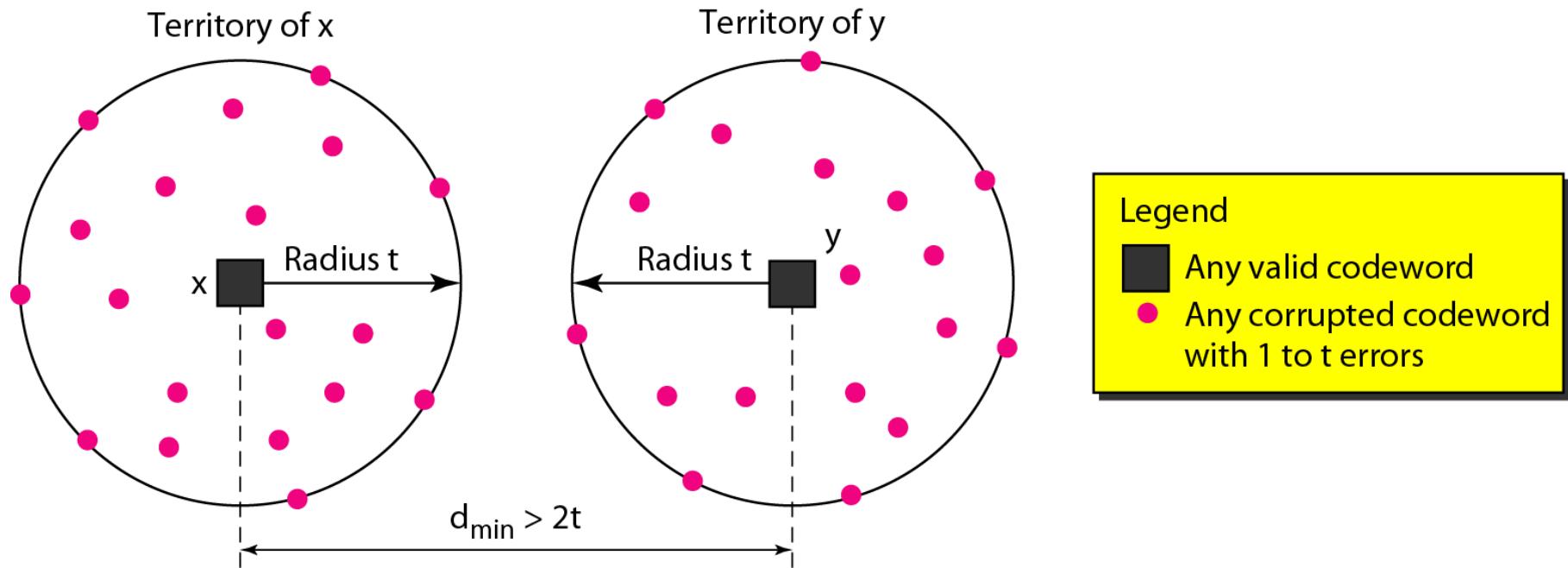
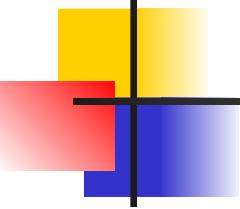


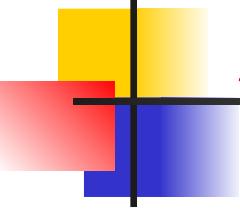
Figure 10.9 Geometric concept for finding d_{min} in error correction





Note

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



Example 10.9

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

Solution

This code guarantees the detection of up to three errors ($s = 3$), but it can correct up to one error. In other words, if this code is used for error correction, part of its capability is wasted. Error correction codes need to have an odd minimum distance (3, 5, 7, . . .).

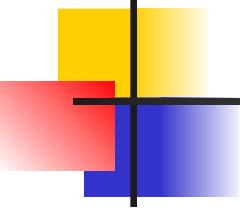
10-3 LINEAR BLOCK CODES

*Almost all block codes used today belong to a subset called **linear block codes**. A linear block code is a code in which the exclusive OR (addition modulo-2) of two valid codewords creates another valid codeword.*

Topics discussed in this section:

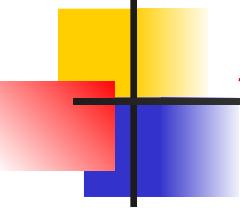
Minimum Distance for Linear Block Codes

Some Linear Block Codes



Note

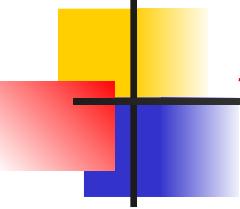
In a linear block code, the exclusive OR (XOR) of any two valid codewords creates another valid codeword.



Example 10.10

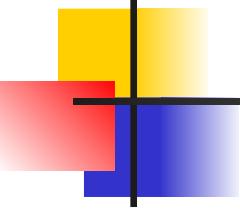
Let us see if the two codes we defined in Table 10.1 and Table 10.2 belong to the class of linear block codes.

- 1. The scheme in Table 10.1 is a linear block code because the result of XORing any codeword with any other codeword is a valid codeword. For example, the XORing of the second and third codewords creates the fourth one.*
- 2. The scheme in Table 10.2 is also a linear block code. We can create all four codewords by XORing two other codewords.*



Example 10.11

In our first code (Table 10.1), the numbers of 1s in the nonzero codewords are 2, 2, and 2. So the minimum Hamming distance is $d_{min} = 2$. In our second code (Table 10.2), the numbers of 1s in the nonzero codewords are 3, 3, and 4. So in this code we have $d_{min} = 3$.



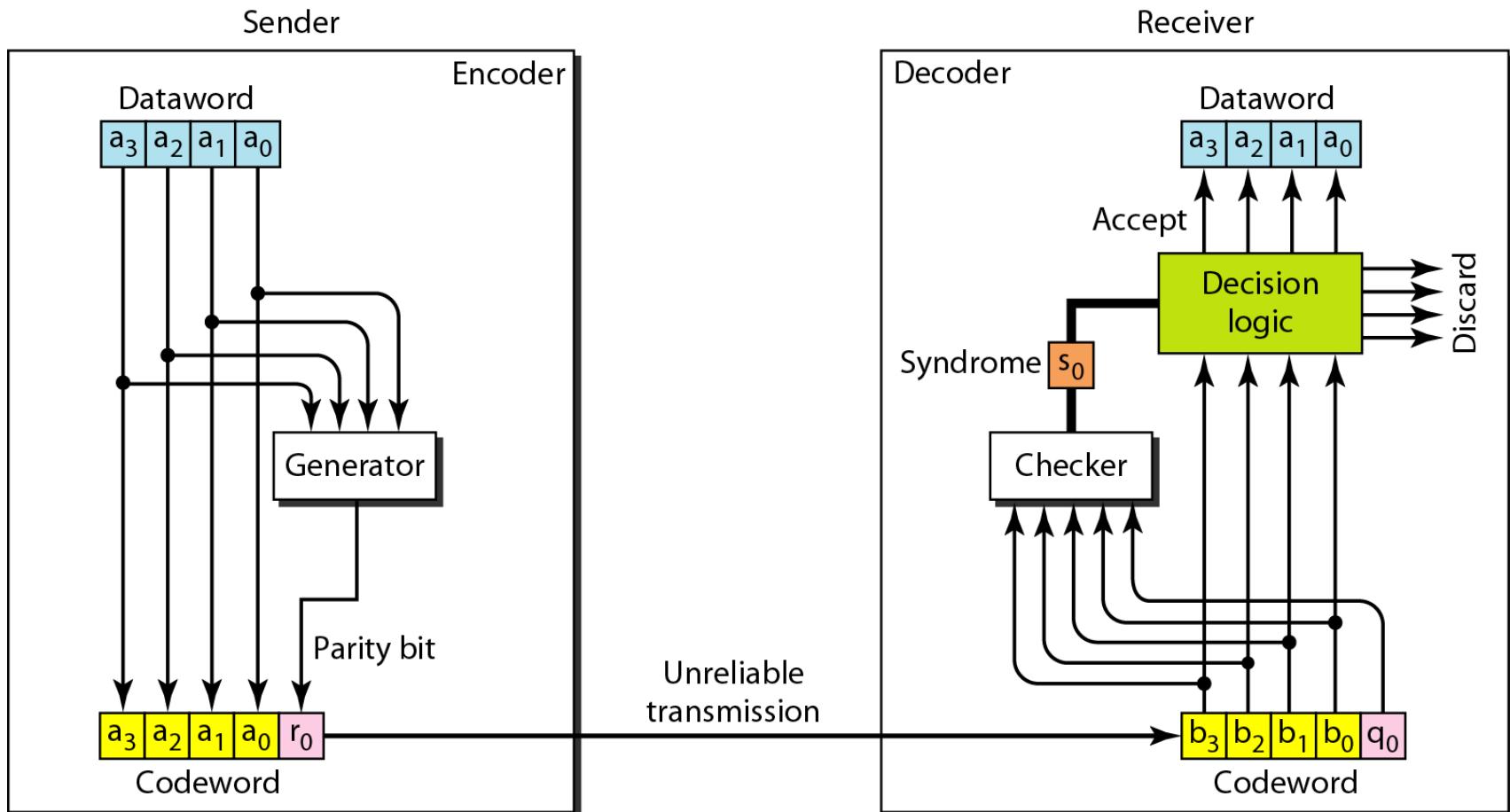
Note

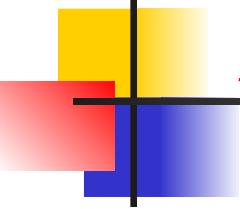
A simple parity-check code is a single-bit error-detecting code in which $n = k + 1$ with $d_{\min} = 2$.

Table 10.3 *Simple parity-check code C(5, 4)*

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

Figure 10.10 Encoder and decoder for simple parity-check code

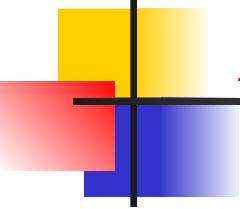




Example 10.12

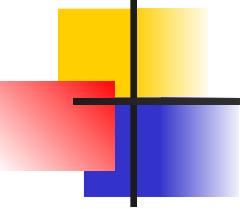
Let us look at some transmission scenarios. Assume the sender sends the dataword 1011. The codeword created from this dataword is 10111, which is sent to the receiver. We examine five cases:

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



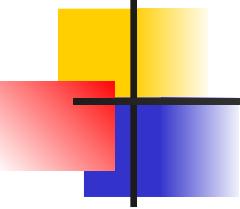
Example 10.12 (continued)

- 4. An error changes r_0 and a second error changes a_3 .**
The received codeword is 00110. The syndrome is 0.
The dataword 0011 is created at the receiver. Note that here the dataword is wrongly created due to the syndrome value.
- 5. Three bits— a_3 , a_2 , and a_1 —are changed by errors.**
The received codeword is 01011. The syndrome is 1.
The dataword is not created. This shows that the simple parity check, guaranteed to detect one single error, can also find any odd number of errors.



Note

**A simple parity-check code can detect
an odd number of errors.**



Note

All Hamming codes discussed in this book have $d_{\min} = 3$.

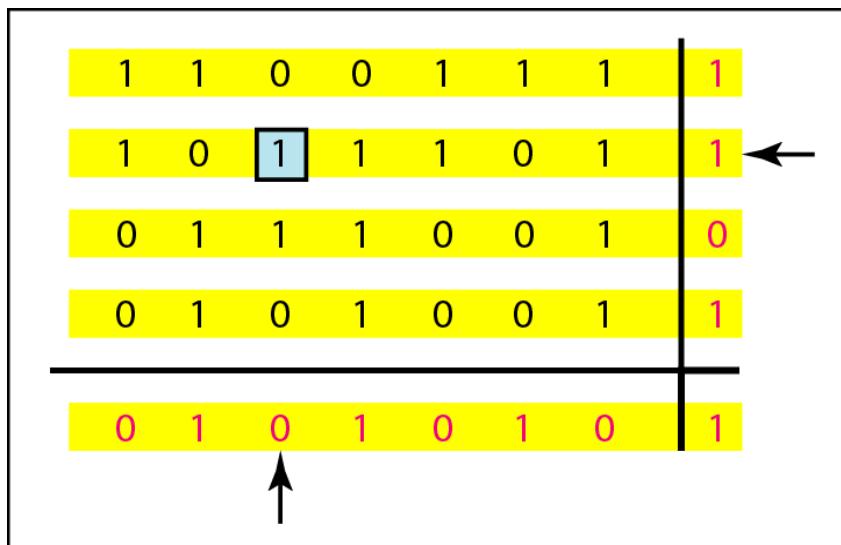
The relationship between m and n in these codes is $n = 2m - 1$.

Figure 10.11 Two-dimensional parity-check code

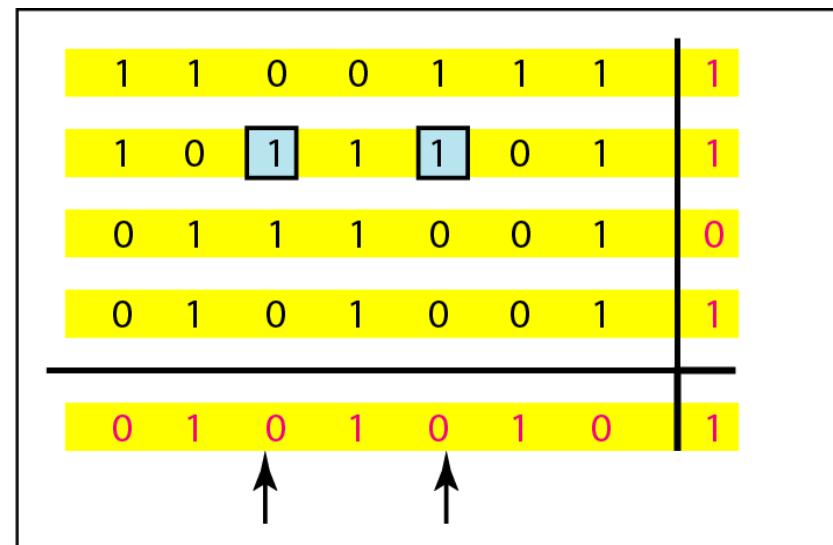
1	1	0	0	1	1	1	1	1
1	0	1	1	1	1	0	1	1
0	1	1	1	0	0	1	0	0
0	1	0	1	0	0	1	1	1
0	1	0	1	0	1	0	1	1

a. Design of row and column parities

Figure 10.11 Two-dimensional parity-check code

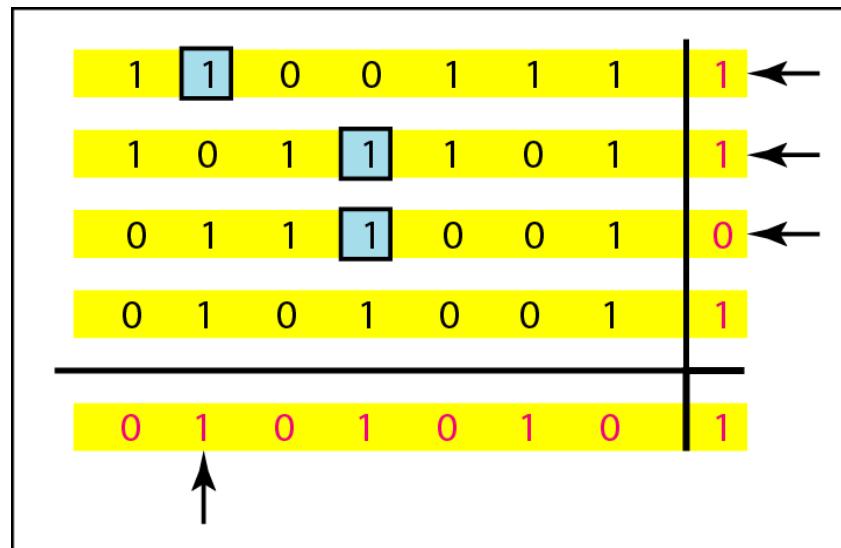


b. One error affects two parities

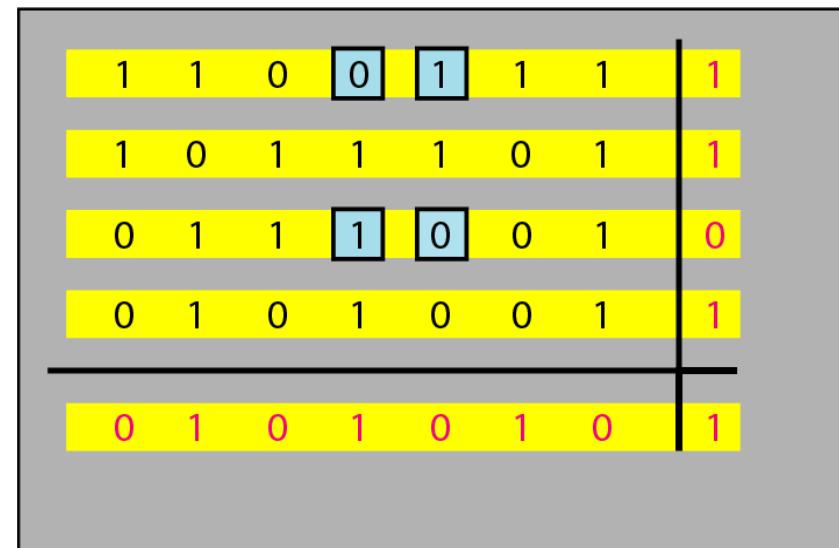


c. Two errors affect two parities

Figure 10.11 Two-dimensional parity-check code



d. Three errors affect four parities



e. Four errors cannot be detected

Table 10.4 *Hamming code C(7, 4)*

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

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

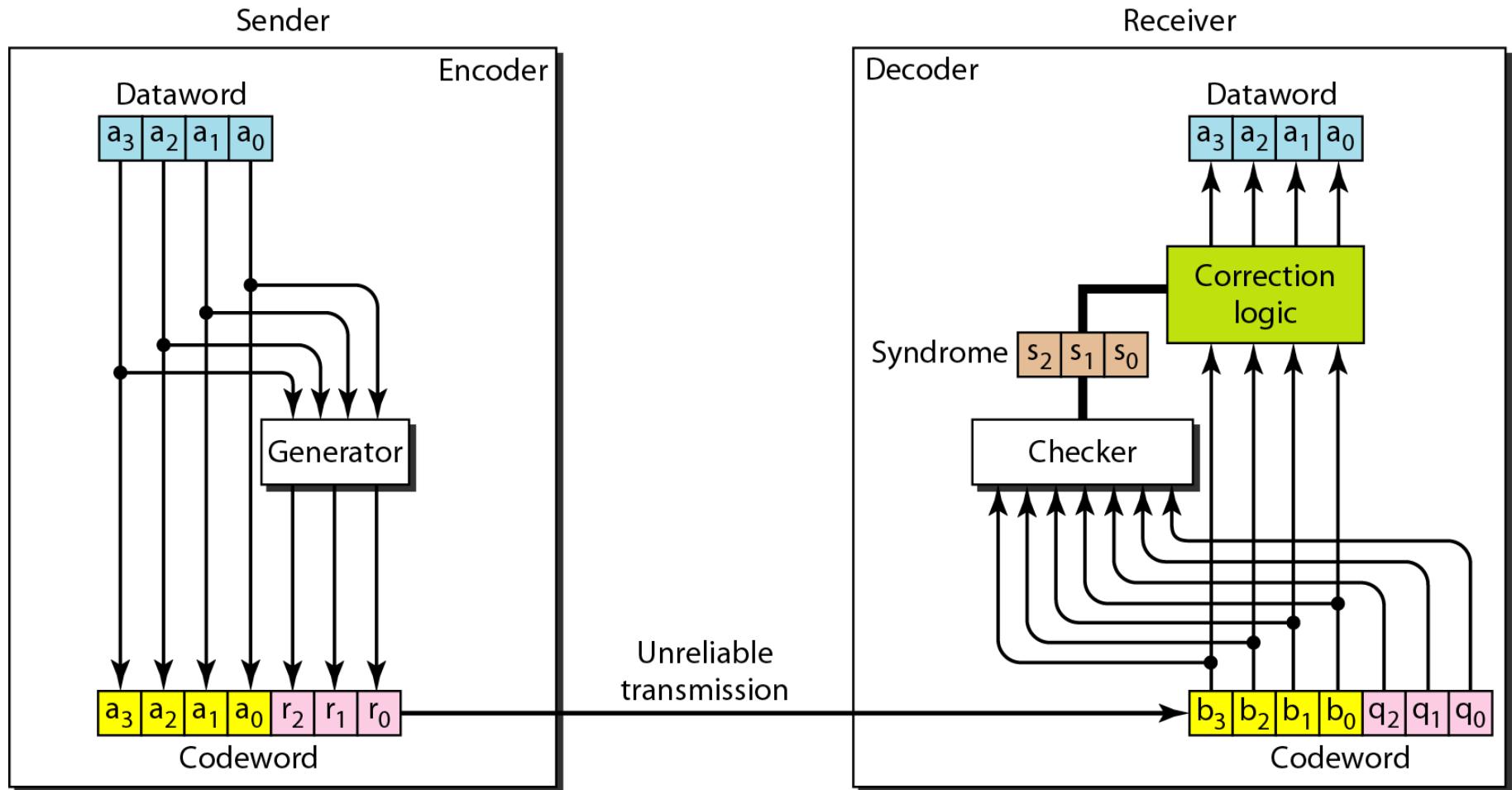
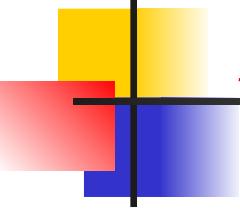


Table 10.5 *Logical decision made by the correction logic analyzer*

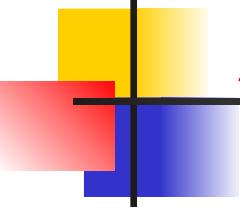
<i>Syndrome</i>	000	001	010	011	100	101	110	111
<i>Error</i>	None	q_0	q_1	b_2	q_2	b_0	b_3	b_1



Example 10.13

Let us trace the path of three datawords from the sender to the destination:

- 1. The dataword 0100 becomes the codeword 0100011.**
The codeword 0100011 is received. The syndrome is 000, the final dataword is 0100.
- 2. The dataword 0111 becomes the codeword 0111001.**
The syndrome is 011. After flipping b_2 (changing the 1 to 0), the final dataword is 0111.
- 3. The dataword 1101 becomes the codeword 1101000.**
The syndrome is 101. After flipping b_0 , we get 0000, the wrong dataword. This shows that our code cannot correct two errors.



Example 10.14

We need a dataword of at least 7 bits. Calculate values of k and n that satisfy this requirement.

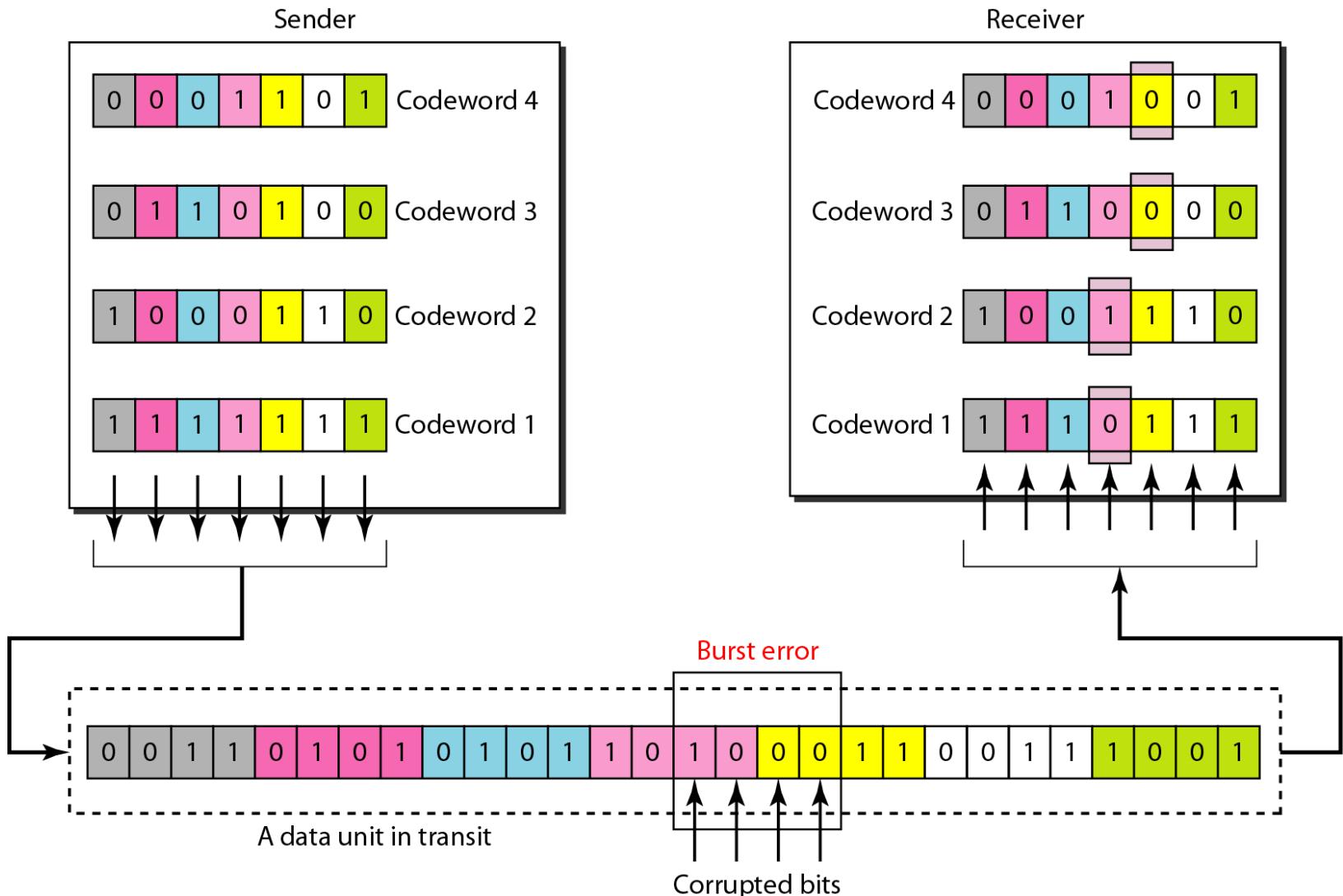
Solution

We need to make $k = n - m$ greater than or equal to 7, or $2m - 1 - m \geq 7$.

1. If we set $m = 3$, the result is $n = 23 - 1$ and $k = 7 - 3$, or 4, which is not acceptable.
2. If we set $m = 4$, then $n = 24 - 1 = 15$ and $k = 15 - 4 = 11$, which satisfies the condition. So the code is

$$C(15, 11)$$

Figure 10.13 Burst error correction using Hamming code



10-4 CYCLIC CODES

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

Topics discussed in this section:

Cyclic Redundancy Check

Hardware Implementation

Polynomials

Cyclic Code Analysis

Advantages of Cyclic Codes

Other Cyclic Codes

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

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

Figure 10.14 CRC encoder and decoder

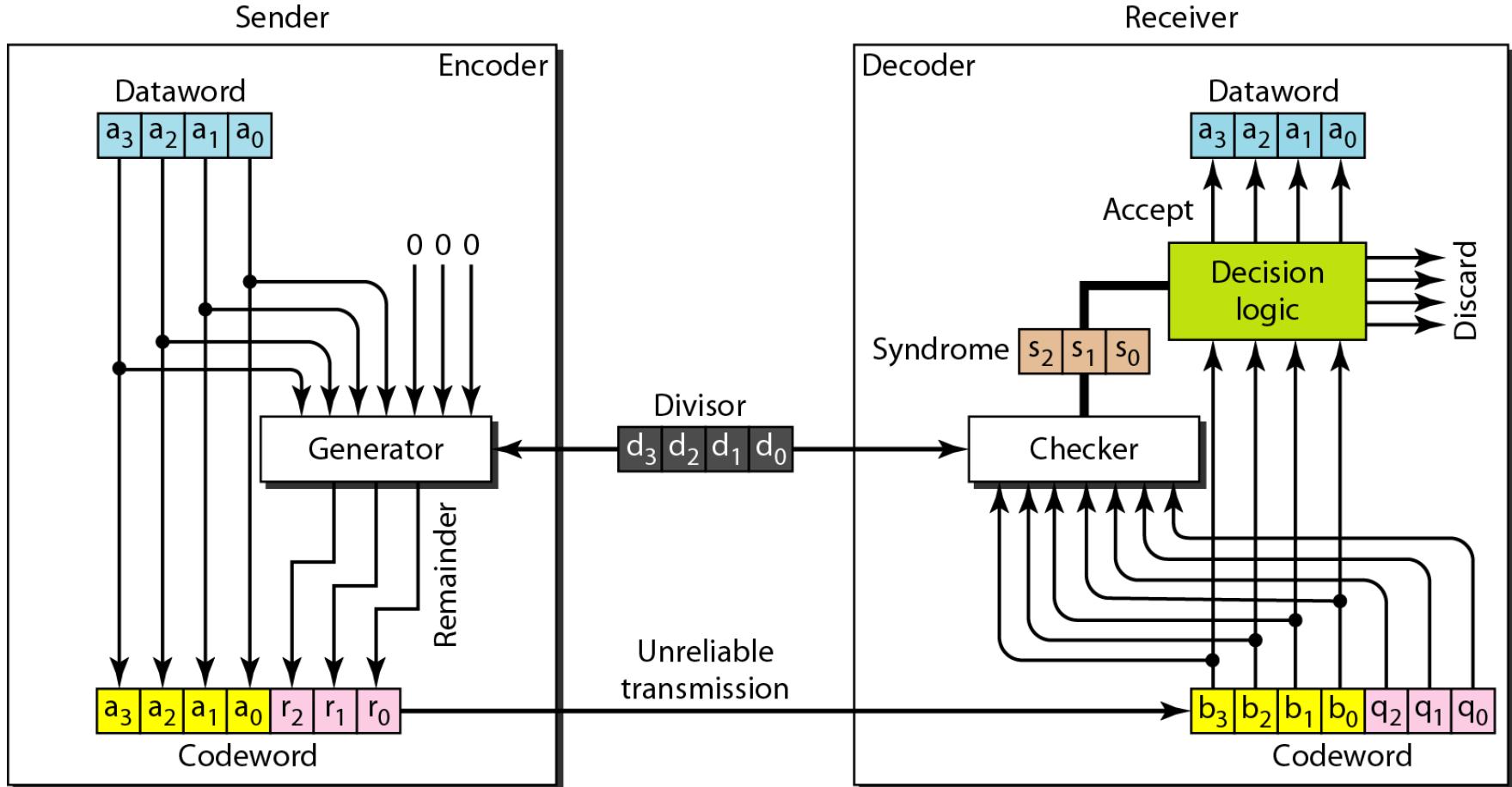


Figure 10.15 Division in CRC encoder

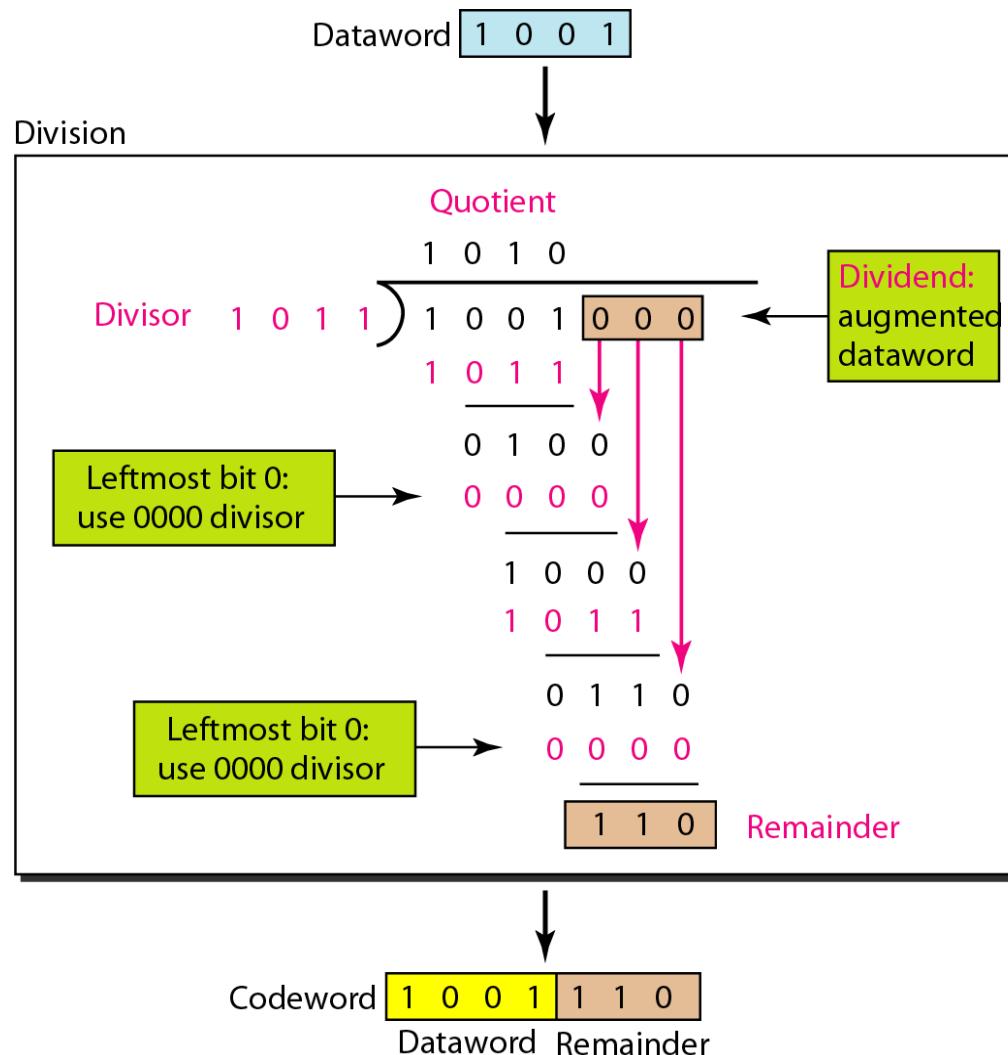


Figure 10.16 Division in the CRC decoder for two cases

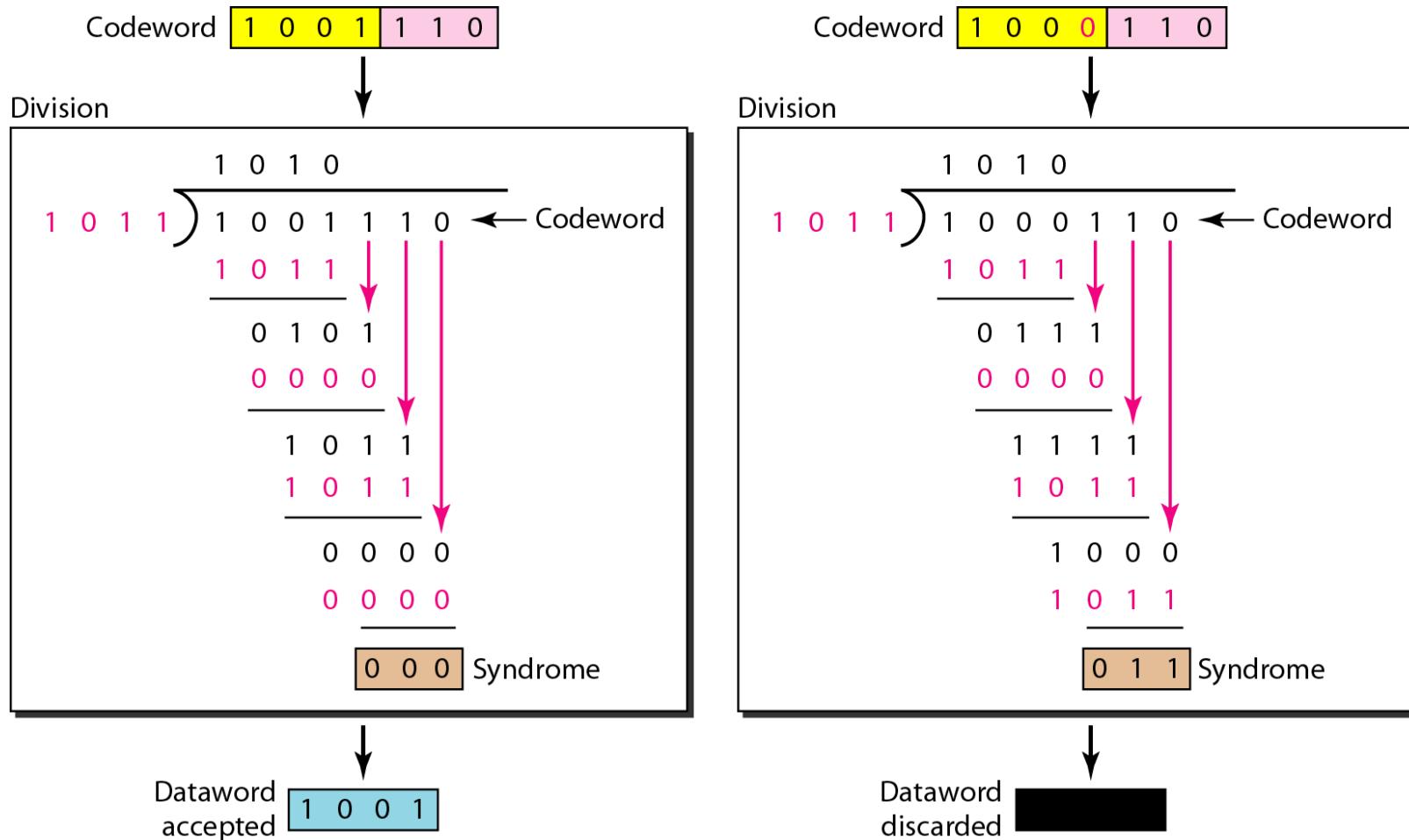


Figure 10.17 Hardwired design of the divisor in CRC

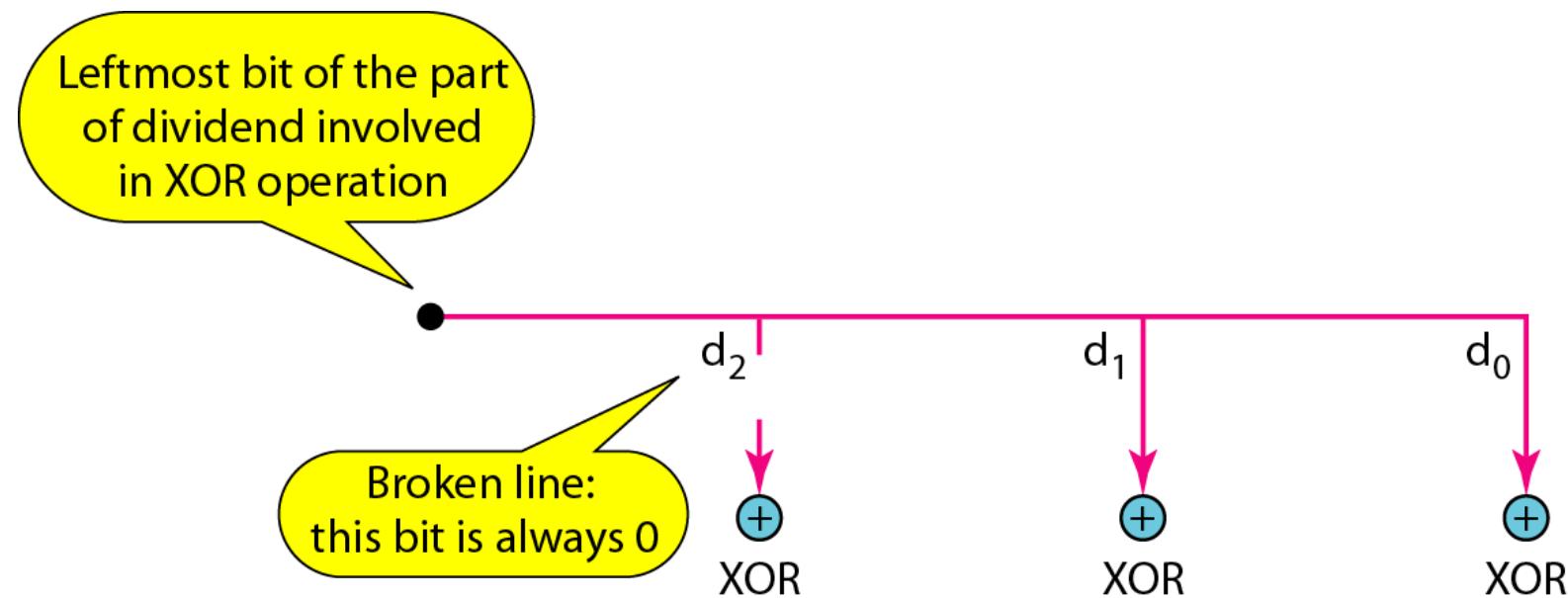


Figure 10.18 Simulation of division in CRC encoder

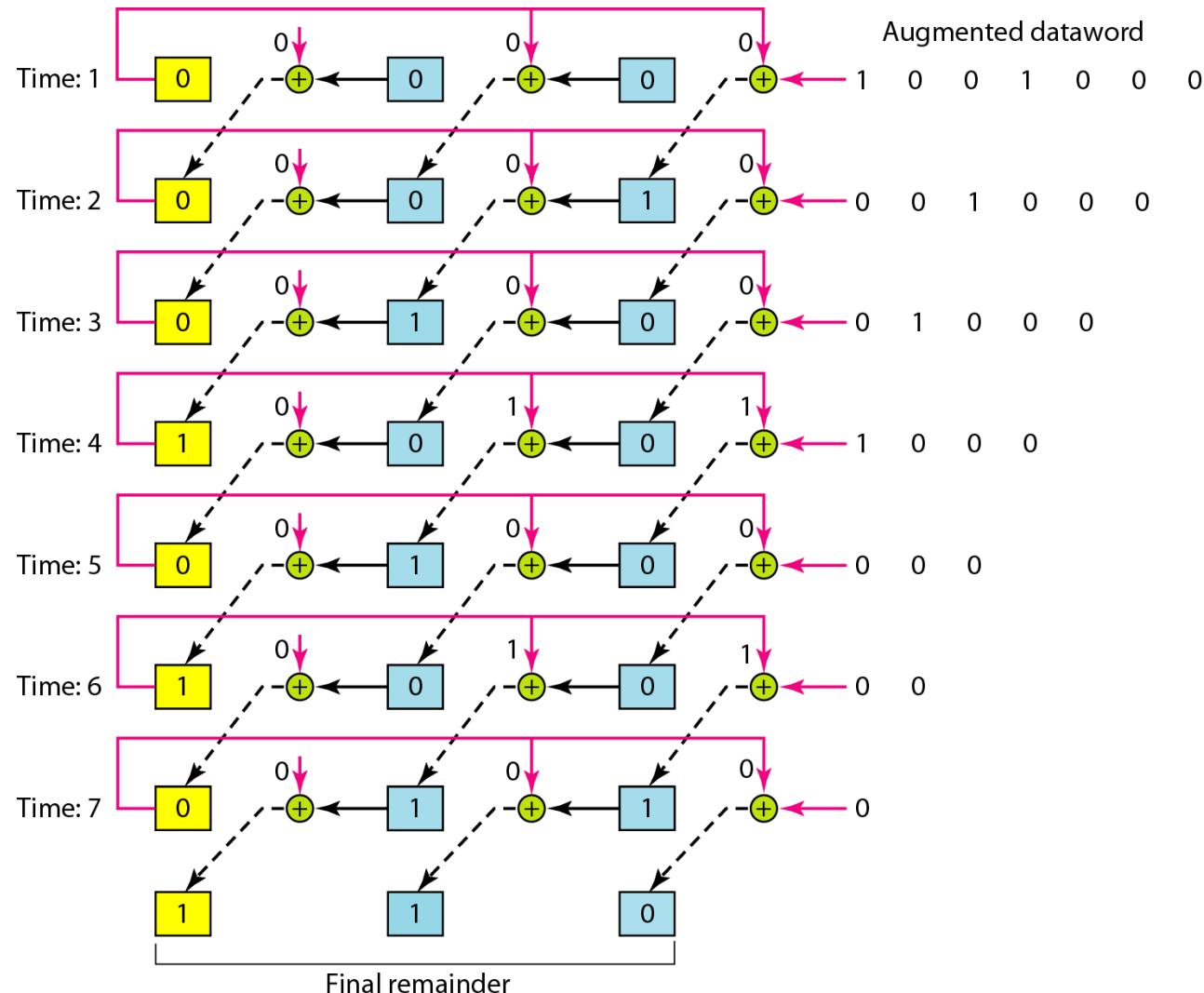


Figure 10.19 *The CRC encoder design using shift registers*

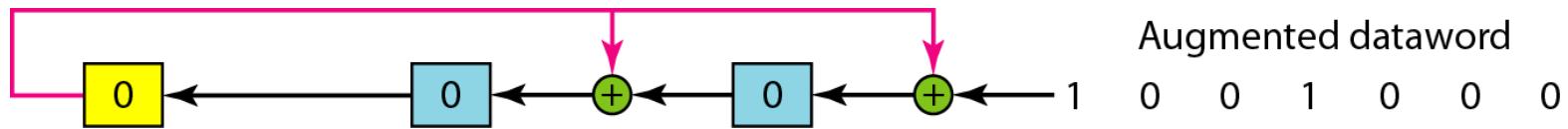
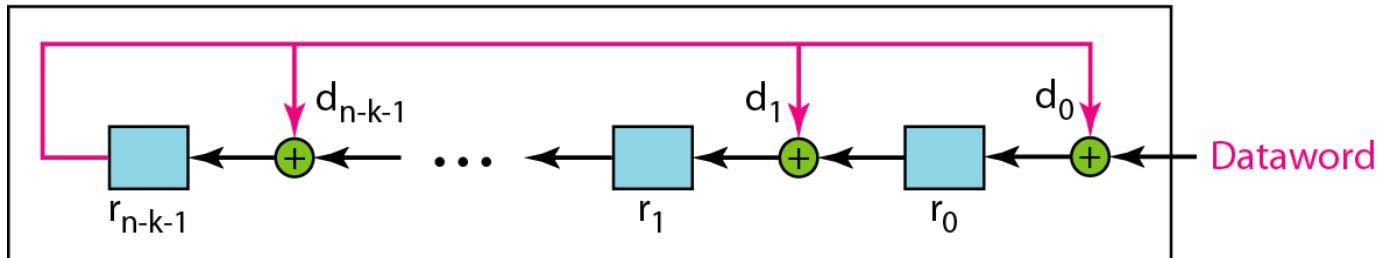


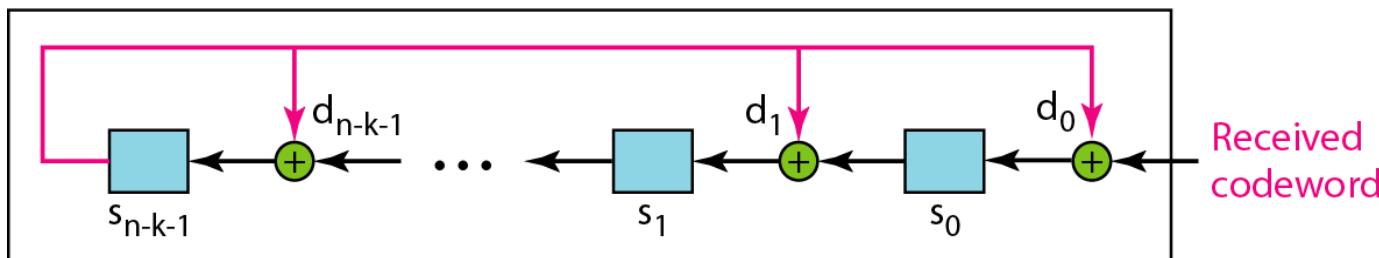
Figure 10.20 General design of encoder and decoder of a CRC code

Note:

The divisor line and XOR are missing if the corresponding bit in the divisor is 0.

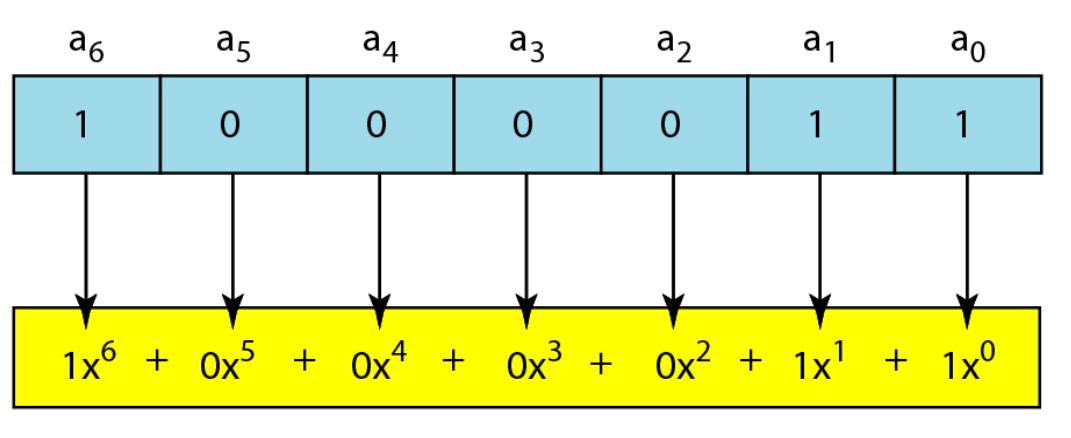


a. Encoder

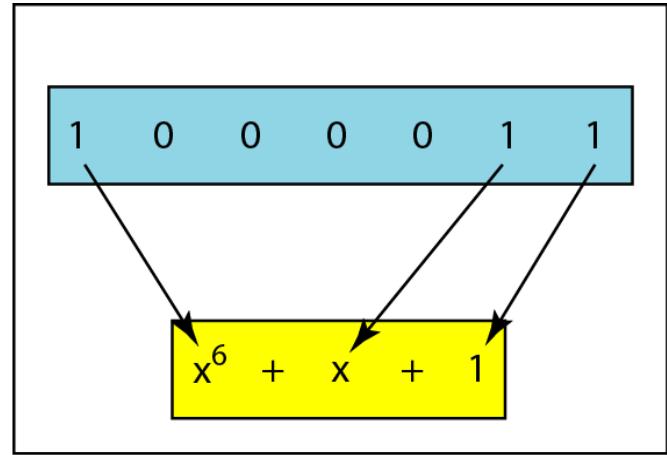


b. Decoder

Figure 10.21 A polynomial to represent a binary word

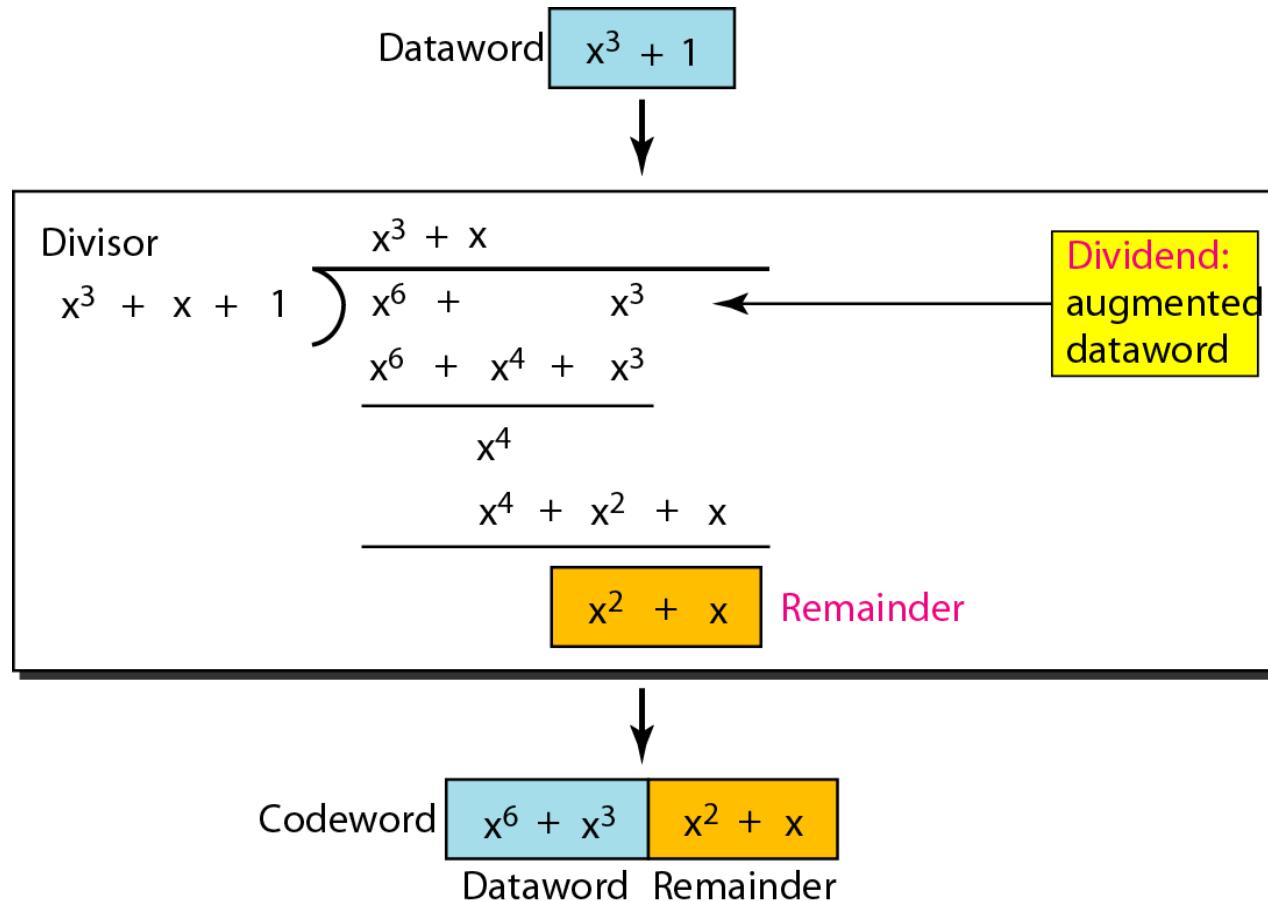


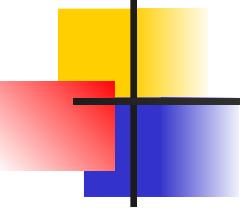
a. Binary pattern and polynomial



b. Short form

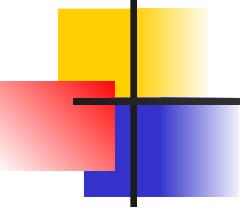
Figure 10.22 CRC division using polynomials





Note

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



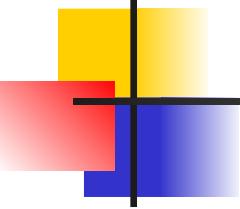
Note

In a cyclic code,

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

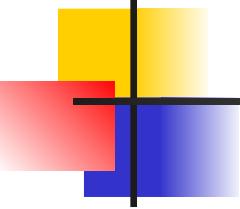
If $s(x) = 0$, either

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



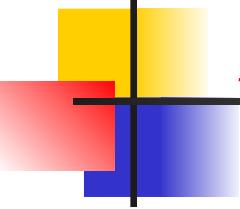
Note

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



Note

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



Example 10.15

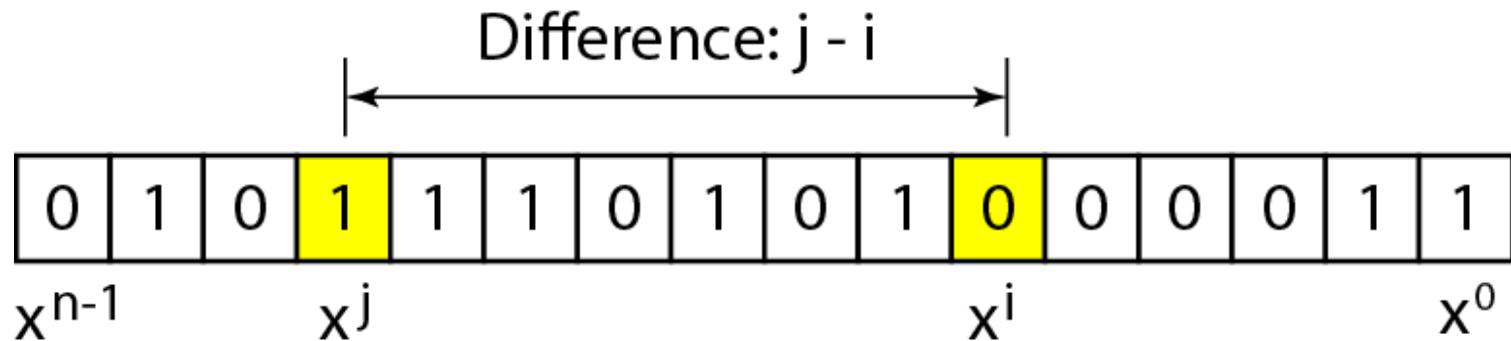
Which of the following $g(x)$ values guarantees that a single-bit error is caught? For each case, what is the error that cannot be caught?

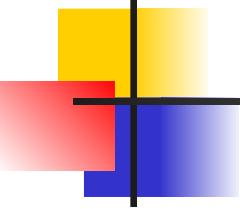
- a. $x + 1$
- b. x^3
- c. 1

Solution

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

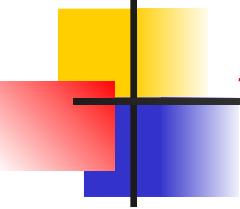
Figure 10.23 *Representation of two isolated single-bit errors using polynomials*





Note

**If a generator cannot divide $x^t + 1$
(t between 0 and $n - 1$),
then all isolated double errors
can be detected.**



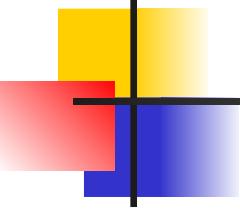
Example 10.16

Find the status of the following generators related to two isolated, single-bit errors.

- a.** $x + 1$ **b.** $x^4 + 1$ **c.** $x^7 + x^6 + 1$ **d.** $x^{15} + x^{14} + 1$

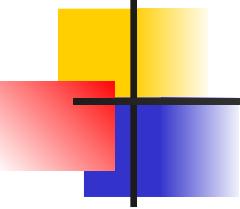
Solution

- a.** *This is a very poor choice for a generator. Any two errors next to each other cannot be detected.*
- b.** *This generator cannot detect two errors that are four positions apart.*
- c.** *This is a good choice for this purpose.*
- d.** *This polynomial cannot divide $x^t + 1$ if t is less than 32,768. A codeword with two isolated errors up to 32,768 bits apart can be detected by this generator.*



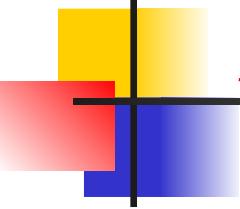
Note

A generator that contains a factor of $x + 1$ can detect all odd-numbered errors.



Note

- All burst errors with $L \leq r$ will be detected.
 - All burst errors with $L = r + 1$ will be detected with probability $1 - (1/2)^{r-1}$.
 - All burst errors with $L > r + 1$ will be detected with probability $1 - (1/2)^r$.
-



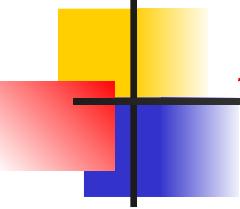
Example 10.17

Find the suitability of the following generators in relation to burst errors of different lengths.

a. $x^6 + 1$ **b.** $x^{18} + x^7 + x + 1$ **c.** $x^{32} + x^{23} + x^7 + 1$

Solution

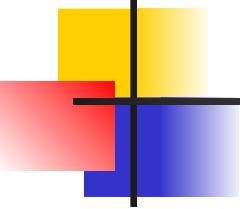
a. This generator can detect all burst errors with a length less than or equal to 6 bits; 3 out of 100 burst errors with length 7 will slip by; 16 out of 1000 burst errors of length 8 or more will slip by.



Example 10.17 (continued)

- b. This generator can detect all burst errors with a length less than or equal to 18 bits; 8 out of 1 million burst errors with length 19 will slip by; 4 out of 1 million burst errors of length 20 or more will slip by.*

- c. This generator can detect all burst errors with a length less than or equal to 32 bits; 5 out of 10 billion burst errors with length 33 will slip by; 3 out of 10 billion burst errors of length 34 or more will slip by.*



Note

A good polynomial generator needs to have the following characteristics:

- 1.** It should have at least two terms.
- 2.** The coefficient of the term x^0 should be 1.
- 3.** It should not divide $x^t + 1$, for t between 2 and $n - 1$.
- 4.** It should have the factor $x + 1$.

Table 10.7 *Standard polynomials*

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

10-5 CHECKSUM

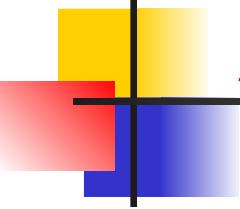
The last error detection method we discuss here is called the checksum. The checksum is used in the Internet by several protocols although not at the data link layer. However, we briefly discuss it here to complete our discussion on error checking

Topics discussed in this section:

Idea

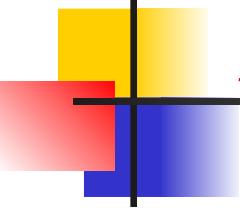
One's Complement

Internet Checksum



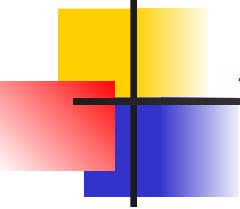
Example 10.18

Suppose our data is a list of five 4-bit numbers that we want to send to a destination. In addition to sending these numbers, we send the sum of the numbers. For example, if the set of numbers is (7, 11, 12, 0, 6), we send (7, 11, 12, 0, 6, 36), where 36 is the sum of the original numbers. The receiver adds the five numbers and compares the result with the sum. If the two are the same, the receiver assumes no error, accepts the five numbers, and discards the sum. Otherwise, there is an error somewhere and the data are not accepted.



Example 10.19

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

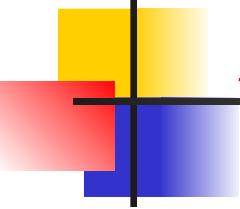


Example 10.20

How can we represent the number 21 in one's complement arithmetic using only four bits?

Solution

The number 21 in binary is 10101 (it needs five bits). We can wrap the leftmost bit and add it to the four rightmost bits. We have $(0101 + 1) = 0110$ or 6.

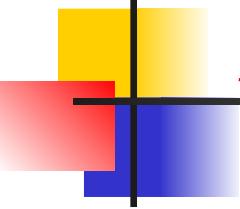


Example 10.21

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

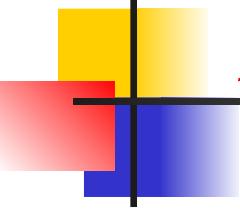
Solution

In one's complement arithmetic, the negative or complement of a number is found by inverting all bits. Positive 6 is 0110; negative 6 is 1001. If we consider only unsigned numbers, this is 9. In other words, the complement of 6 is 9. Another way to find the complement of a number in one's complement arithmetic is to subtract the number from $2^n - 1$ ($16 - 1$ in this case).



Example 10.22

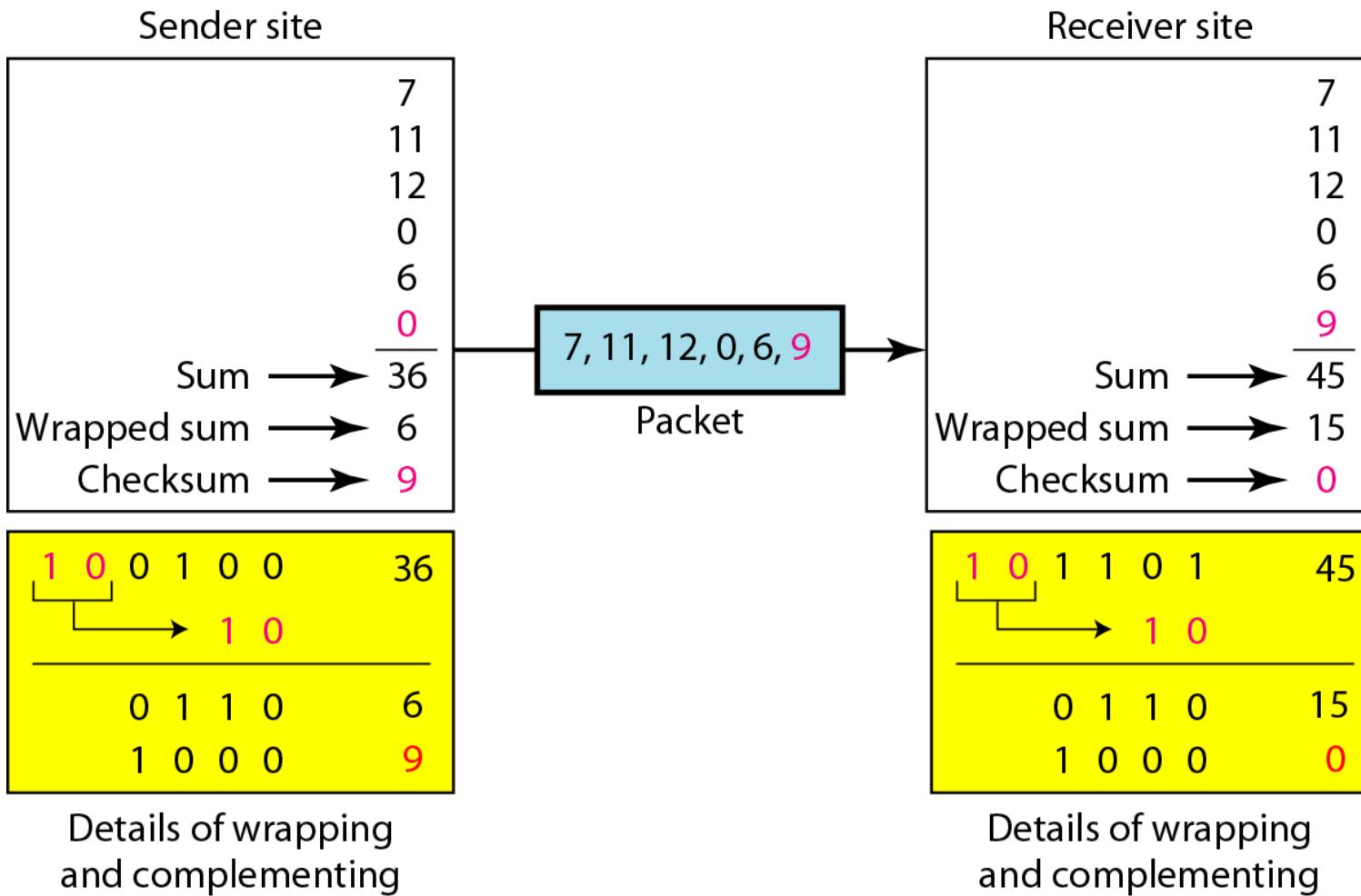
Let us redo Exercise 10.19 using one's complement arithmetic. Figure 10.24 shows the process at the sender and at the receiver. The sender initializes the checksum to 0 and adds all data items and the checksum (the checksum is considered as one data item and is shown in color). The result is 36. However, 36 cannot be expressed in 4 bits. The extra two bits are wrapped and added with the sum to create the wrapped sum value 6. In the figure, we have shown the details in binary. The sum is then complemented, resulting in the checksum value 9 ($15 - 6 = 9$). The sender now sends six data items to the receiver including the checksum 9.

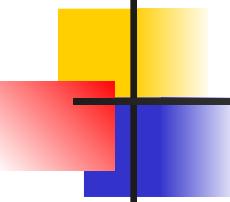


Example 10.22 (continued)

The receiver follows the same procedure as the sender. It adds all data items (including the checksum); the result is 45. The sum is wrapped and becomes 15. The wrapped sum is complemented and becomes 0. Since the value of the checksum is 0, this means that the data is not corrupted. The receiver drops the checksum and keeps the other data items. If the checksum is not zero, the entire packet is dropped.

Figure 10.24 Example 10.22

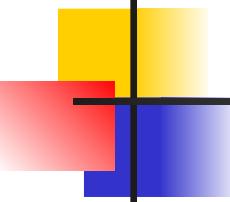




Note

Sender site:

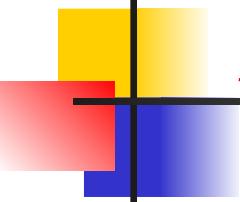
- 1. The message is divided into 16-bit words.**
- 2. The value of the checksum word is set to 0.**
- 3. All words including the checksum are added using one's complement addition.**
- 4. The sum is complemented and becomes the checksum.**
- 5. The checksum is sent with the data.**



Note

Receiver site:

- 1. The message (including checksum) is divided into 16-bit words.**
- 2. All words are added using one's complement addition.**
- 3. The sum is complemented and becomes the new checksum.**
- 4. If the value of checksum is 0, the message is accepted; otherwise, it is rejected.**



Example 10.23

Let us calculate the checksum for a text of 8 characters (“Forouzan”). The text needs to be divided into 2-byte (16-bit) words. We use ASCII (see Appendix A) to change each byte to a 2-digit hexadecimal number. For example, F is represented as 0x46 and o is represented as 0x6F. Figure 10.25 shows how the checksum is calculated at the sender and receiver sites. In part a of the figure, the value of partial sum for the first column is 0x36. We keep the rightmost digit (6) and insert the leftmost digit (3) as the carry in the second column. The process is repeated for each column. Note that if there is any corruption, the checksum recalculated by the receiver is not all 0s. We leave this an exercise.

Figure 10.25 Example 10.23

1	0	1	3	Carries
4	6	6	F	(Fo)
7	2	6	7	(ro)
7	5	7	A	(uz)
6	1	6	E	(an)
0	0	0	0	Checksum (initial)
<hr/>				
8	F	C	6	Sum (partial)
<hr/>				
8	F	C	7	Sum
7	0	3	8	Checksum (to send)

a. Checksum at the sender site

1	0	1	3	Carries
4	6	6	F	(Fo)
7	2	6	7	(ro)
7	5	7	A	(uz)
6	1	6	E	(an)
7	0	3	8	Checksum (received)
<hr/>				
F	F	F	E	Sum (partial)
<hr/>				
8	F	C	7	Sum
0	0	0	0	Checksum (new)

a. Checksum at the receiver site

Chapter 11

Data Link Control

11-1 FRAMING

*The data link layer needs to pack bits into **frames**, so that each frame is distinguishable from another. Our postal system practices a type of framing. The simple act of inserting a letter into an envelope separates one piece of information from another; the envelope serves as the delimiter.*

Topics discussed in this section:

Fixed-Size Framing

Variable-Size Framing

Figure 11.1 A frame in a character-oriented protocol

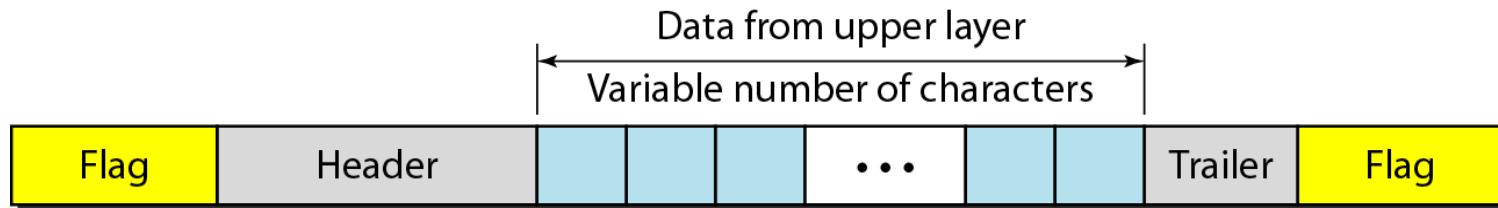
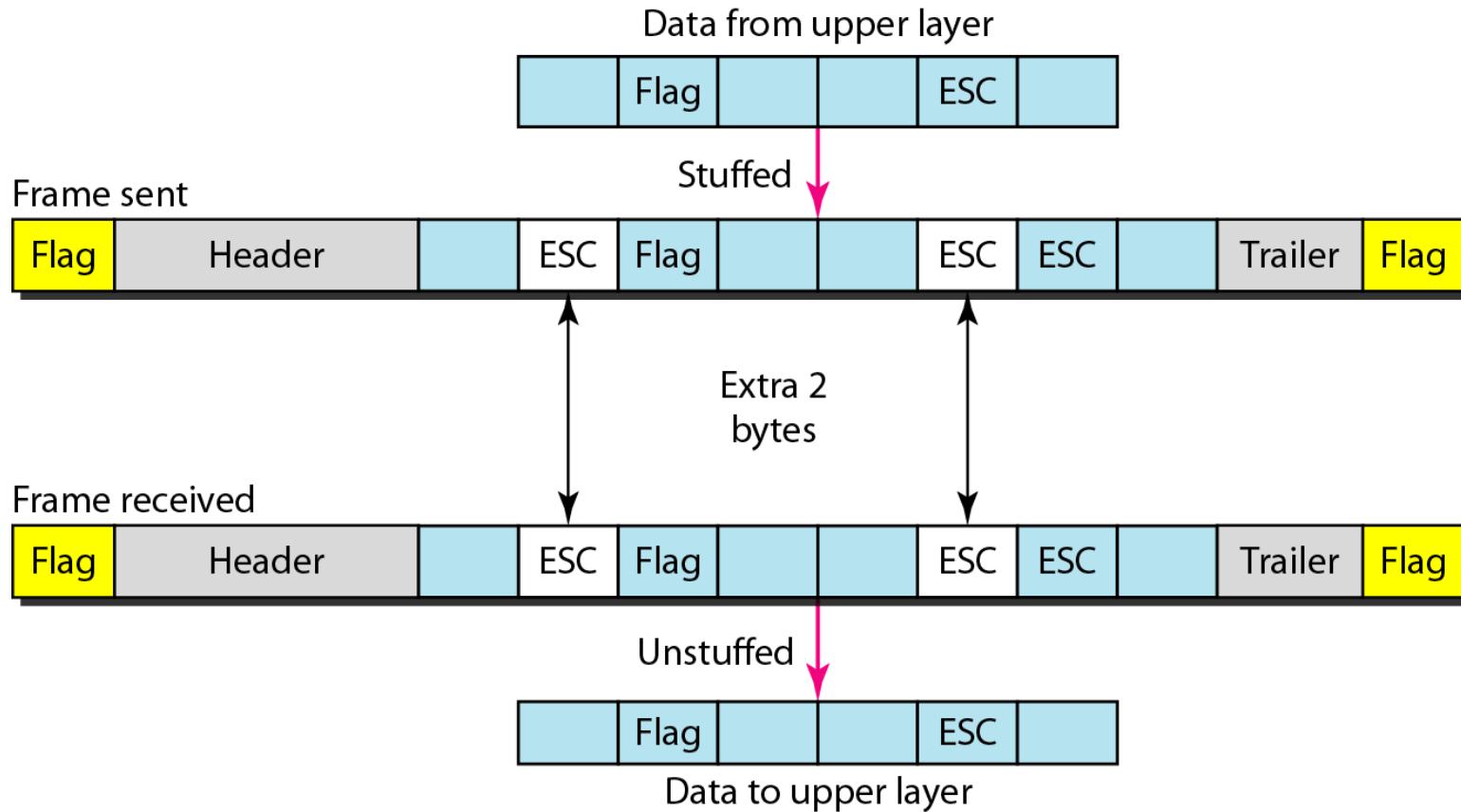
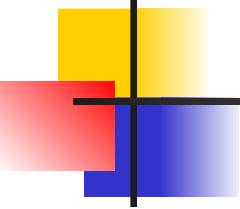


Figure 11.2 Byte stuffing and unstuffing

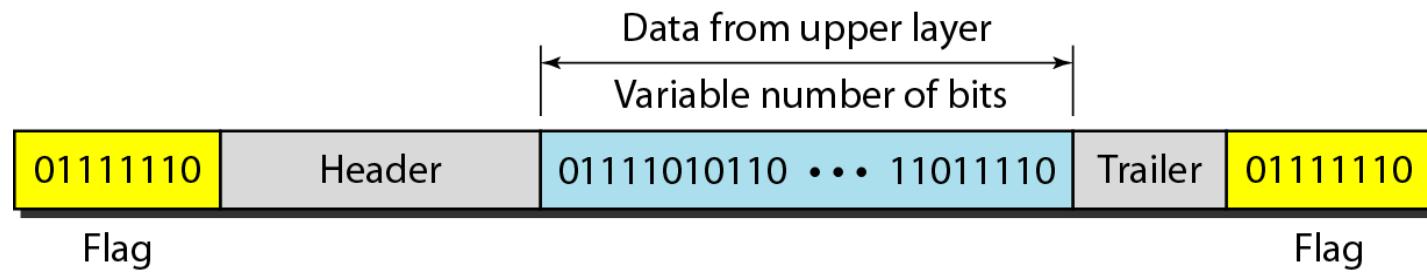


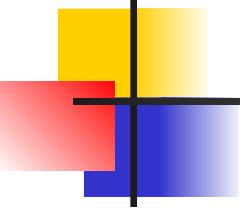


Note

Byte stuffing is the process of adding 1 extra byte whenever there is a flag or escape character in the text.

Figure 11.3 A frame in a bit-oriented protocol

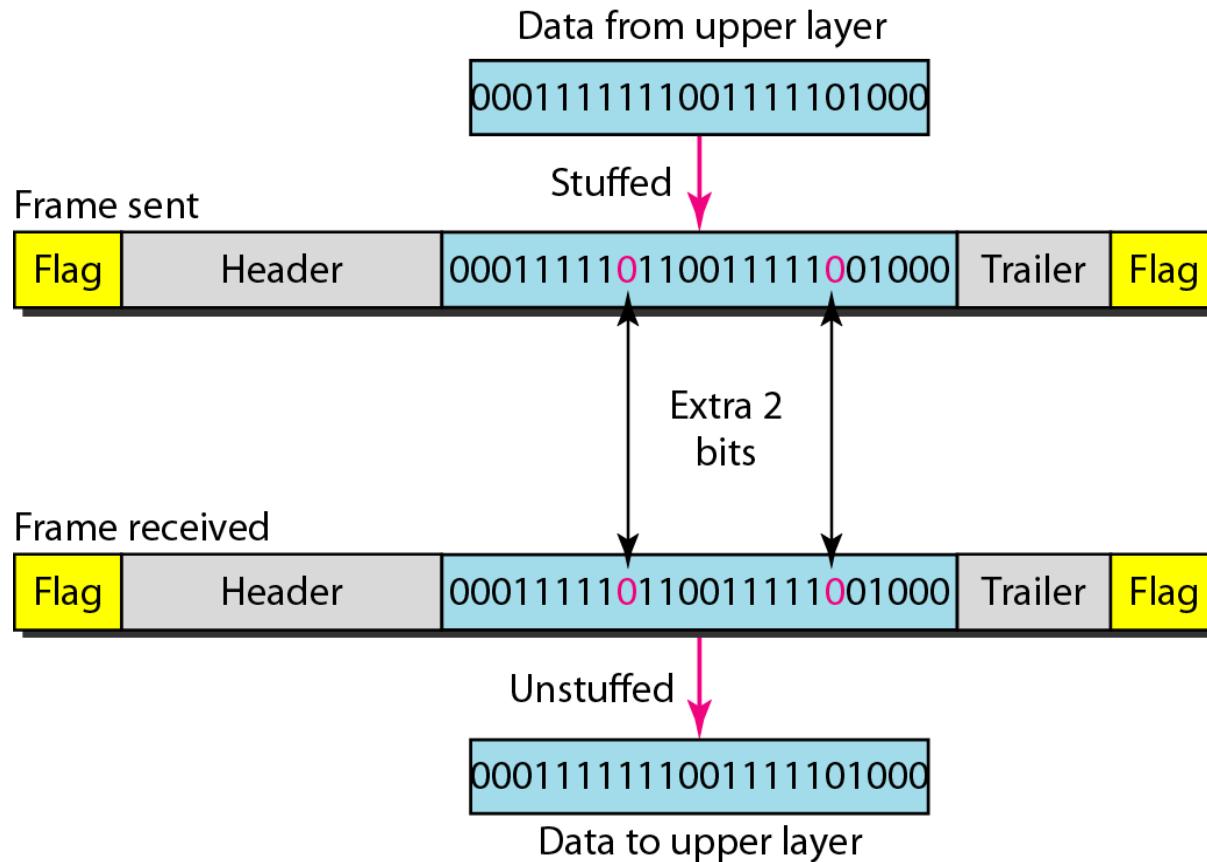




Note

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

Figure 11.4 Bit stuffing and unstuffing



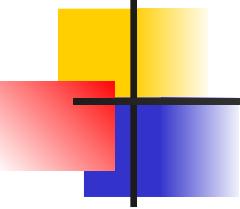
11-2 FLOW AND ERROR CONTROL

*The most important responsibilities of the data link layer are **flow control** and **error control**. Collectively, these functions are known as **data link control**.*

Topics discussed in this section:

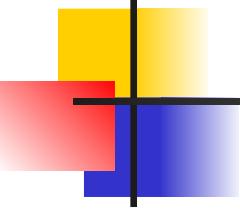
Flow Control

Error Control



Note

Flow control refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment.



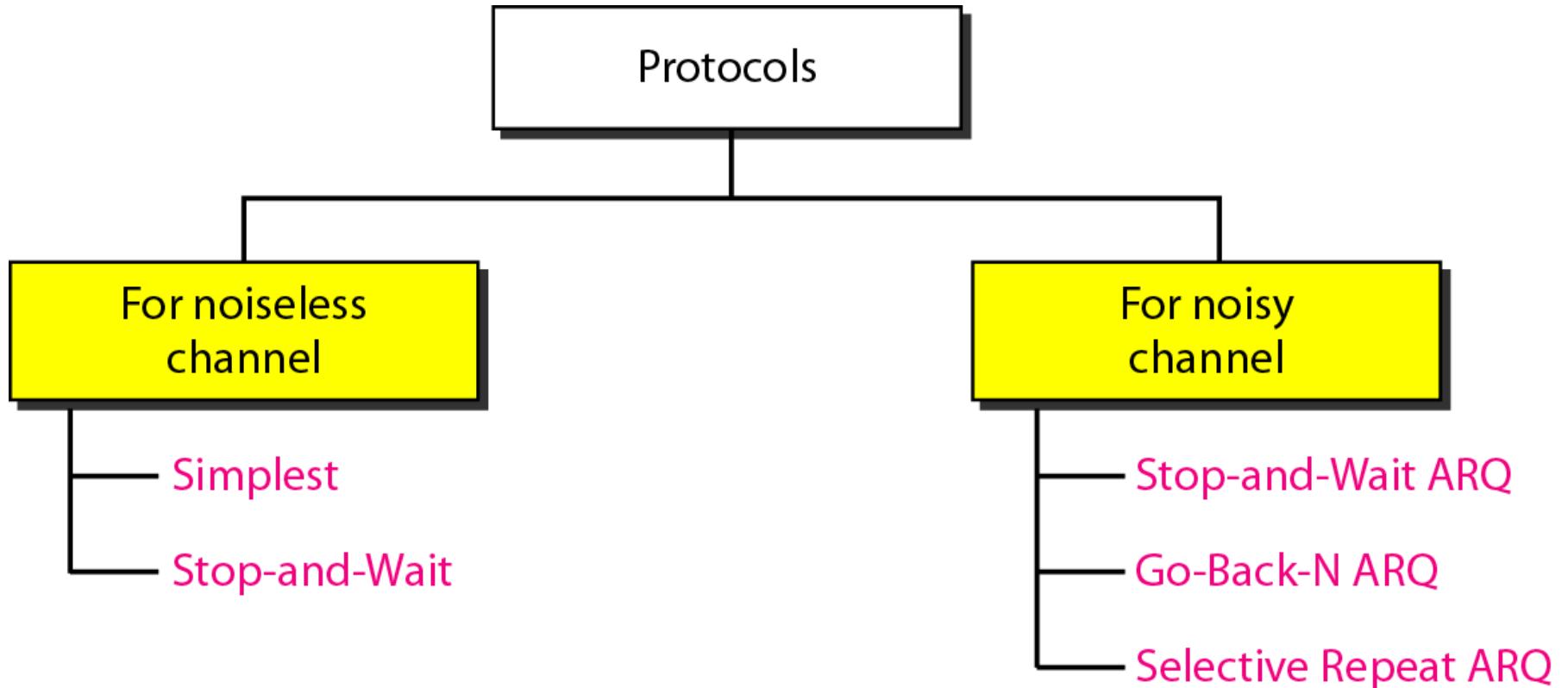
Note

Error control in the data link layer is based on automatic repeat request, which is the retransmission of data.

11-3 PROTOCOLS

Now let us see how the data link layer can combine framing, flow control, and error control to achieve the delivery of data from one node to another. The protocols are normally implemented in software by using one of the common programming languages. To make our discussions language-free, we have written in pseudocode a version of each protocol that concentrates mostly on the procedure instead of delving into the details of language rules.

Figure 11.5 *Taxonomy of protocols discussed in this chapter*



11-4 NOISELESS CHANNELS

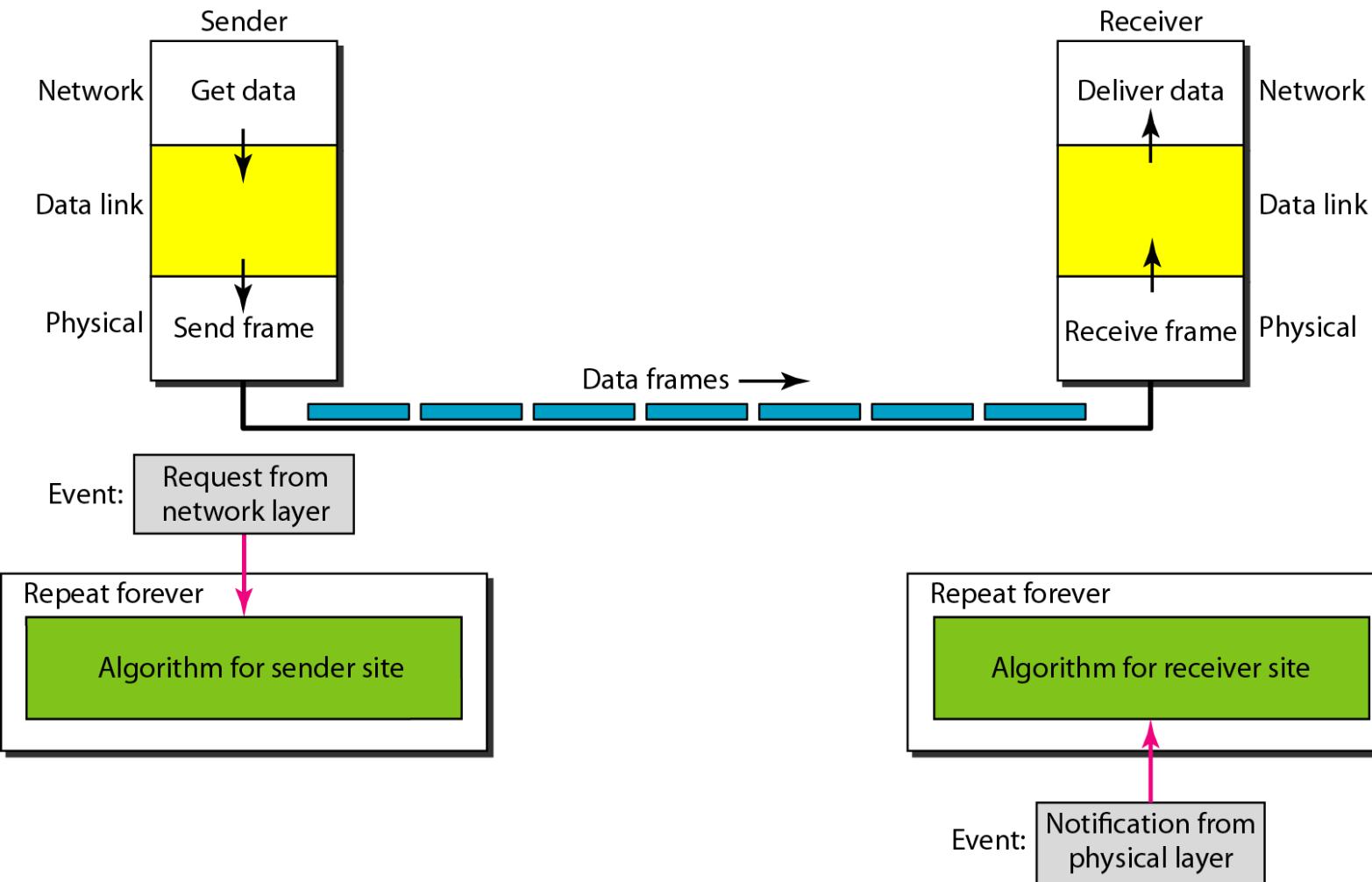
Let us first assume we have an ideal channel in which no frames are lost, duplicated, or corrupted. We introduce two protocols for this type of channel.

Topics discussed in this section:

Simplest Protocol

Stop-and-Wait Protocol

Figure 11.6 *The design of the simplest protocol with no flow or error control*

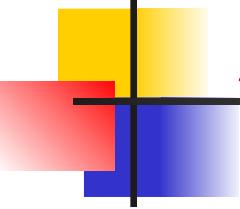


Algorithm 11.1 *Sender-site algorithm for the simplest protocol*

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

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

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



Example 11.1

Figure 11.7 shows an example of communication using this protocol. It is very simple. The sender sends a sequence of frames without even thinking about the receiver. To send three frames, three events occur at the sender site and three events at the receiver site. Note that the data frames are shown by tilted boxes; the height of the box defines the transmission time difference between the first bit and the last bit in the frame.

Figure 11.7 Flow diagram for Example 11.1

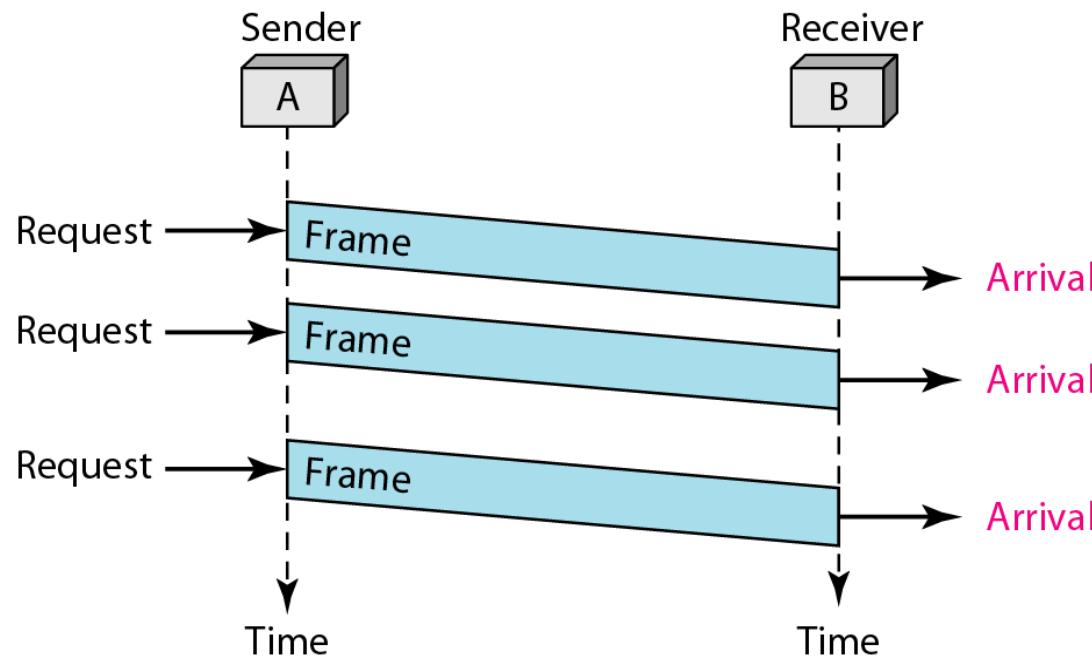
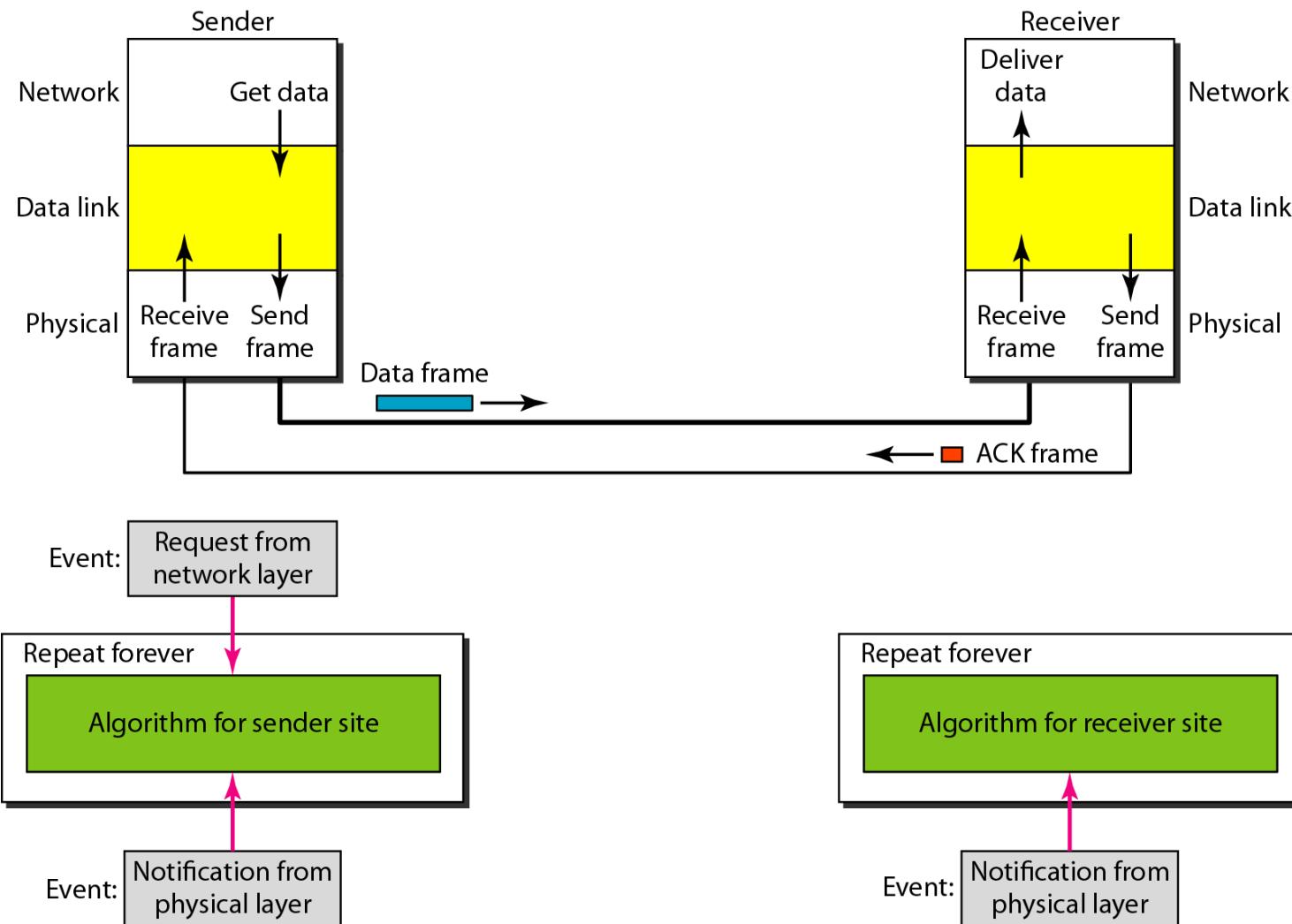


Figure 11.8 Design of Stop-and-Wait Protocol

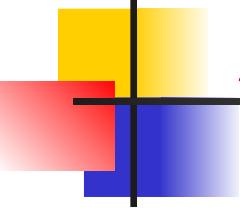


Algorithm 11.3 *Sender-site algorithm for Stop-and-Wait Protocol*

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

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

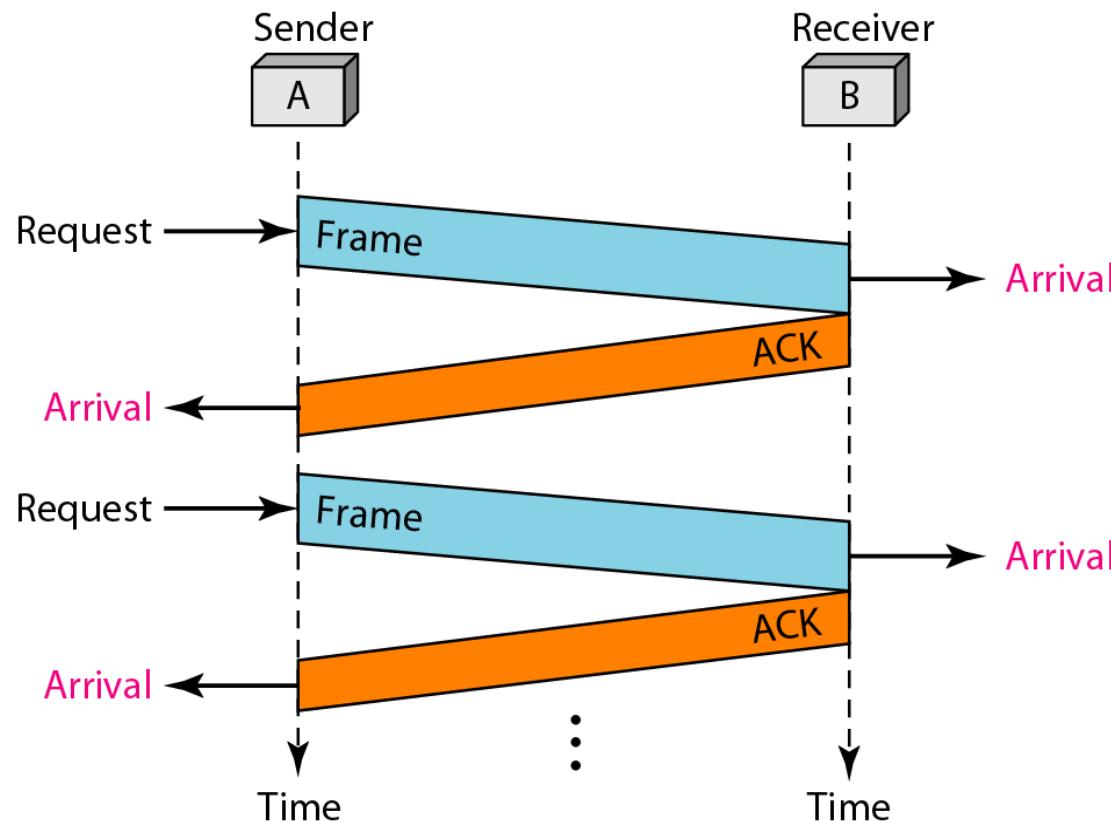
```
1 while(true)                                //Repeat forever
2 {
3     WaitForEvent();                         // Sleep until an event occurs
4     if(Event(ArrivalNotification)) //Data frame arrives
5     {
6         ReceiveFrame();
7         ExtractData();
8         Deliver(data);            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



Example 11.2

Figure 11.9 shows an example of communication using this protocol. It is still very simple. The sender sends one frame and waits for feedback from the receiver. When the ACK arrives, the sender sends the next frame. Note that sending two frames in the protocol involves the sender in four events and the receiver in two events.

Figure 11.9 Flow diagram for Example 11.2



11-5 NOISY CHANNELS

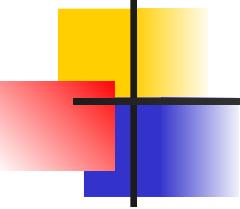
Although the Stop-and-Wait Protocol gives us an idea of how to add flow control to its predecessor, noiseless channels are nonexistent. We discuss three protocols in this section that use error control.

Topics discussed in this section:

Stop-and-Wait Automatic Repeat Request

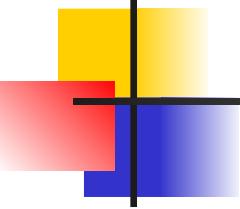
Go-Back-N Automatic Repeat Request

Selective Repeat Automatic Repeat Request



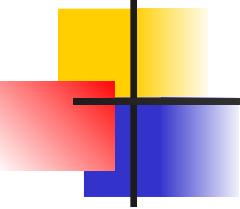
Note

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



Note

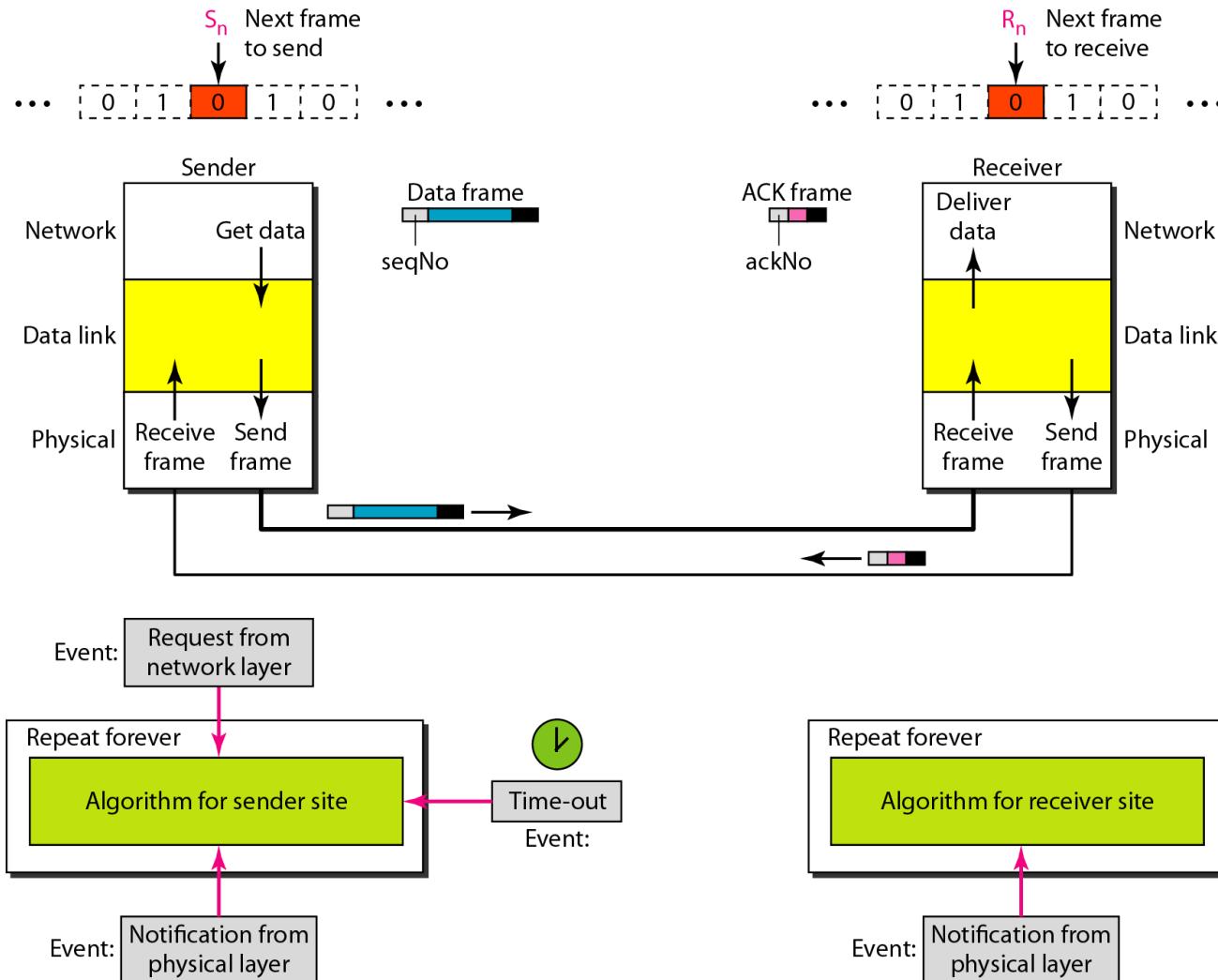
**In Stop-and-Wait ARQ, we use sequence numbers to number the frames.
The sequence numbers are based on modulo-2 arithmetic.**



Note

In Stop-and-Wait ARQ, the acknowledgment number always announces in modulo-2 arithmetic the sequence number of the next frame expected.

Figure 11.10 Design of the Stop-and-Wait ARQ Protocol



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

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

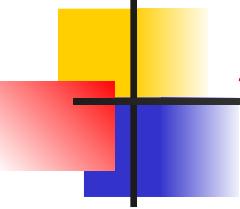
(continued)

Algorithm 11.5 Sender-site algorithm for Stop-and-Wait ARQ (continued)

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

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

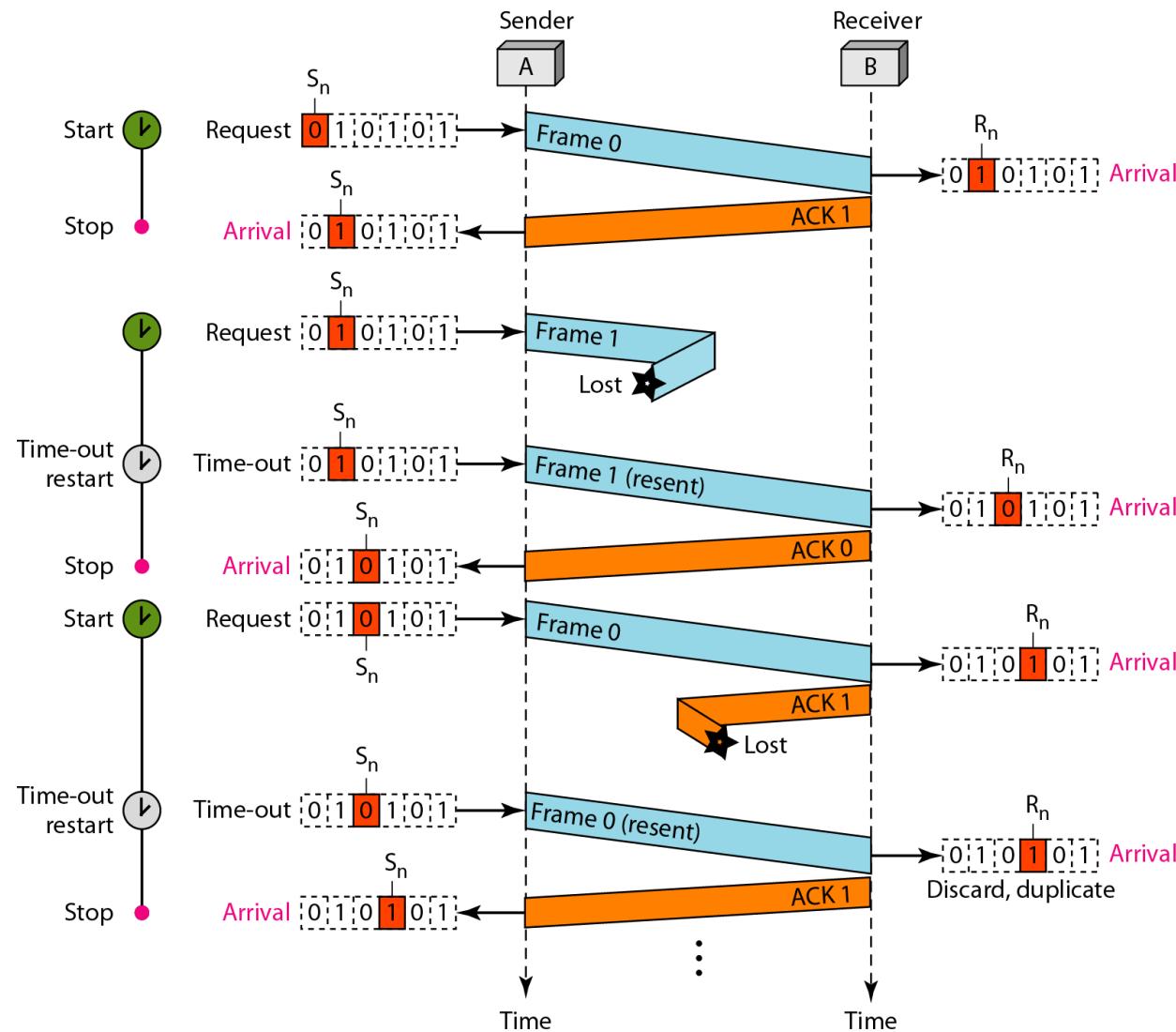
```
1 Rn = 0;                                // Frame 0 expected to arrive first
2 while(true)
3 {
4     WaitForEvent();                      // Sleep until an event occurs
5     if(Event(ArrivalNotification))      //Data frame arrives
6     {
7         ReceiveFrame();
8         if(corrupted(frame));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

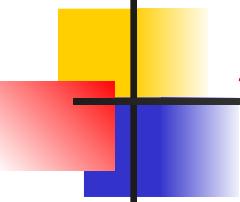


Example 11.3

Figure 11.11 shows an example of Stop-and-Wait ARQ. Frame 0 is sent and acknowledged. Frame 1 is lost and resent after the time-out. The resent frame 1 is acknowledged and the timer stops. Frame 0 is sent and acknowledged, but the acknowledgment is lost. The sender has no idea if the frame or the acknowledgment is lost, so after the time-out, it resends frame 0, which is acknowledged.

Figure 11.11 Flow diagram for Example 11.3





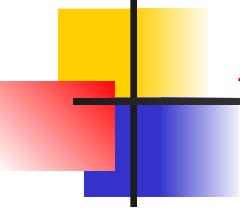
Example 11.4

Assume that, in a Stop-and-Wait ARQ system, the bandwidth of the line is 1 Mbps, and 1 bit takes 20 ms to make a round trip. What is the bandwidth-delay product? If the system data frames are 1000 bits in length, what is the utilization percentage of the link?

Solution

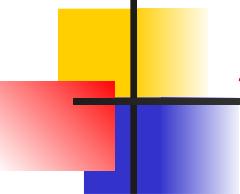
The bandwidth-delay product is

$$(1 \times 10^6) \times (20 \times 10^{-3}) = 20,000 \text{ bits}$$



Example 11.4 (continued)

The system can send 20,000 bits during the time it takes for the data to go from the sender to the receiver and then back again. However, the system sends only 1000 bits. We can say that the link utilization is only $1000/20,000$, or 5 percent. For this reason, for a link with a high bandwidth or long delay, the use of Stop-and-Wait ARQ wastes the capacity of the link.

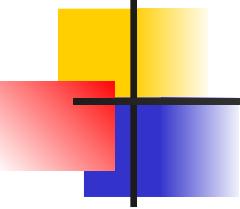


Example 11.5

What is the utilization percentage of the link in Example 11.4 if we have a protocol that can send up to 15 frames before stopping and worrying about the acknowledgments?

Solution

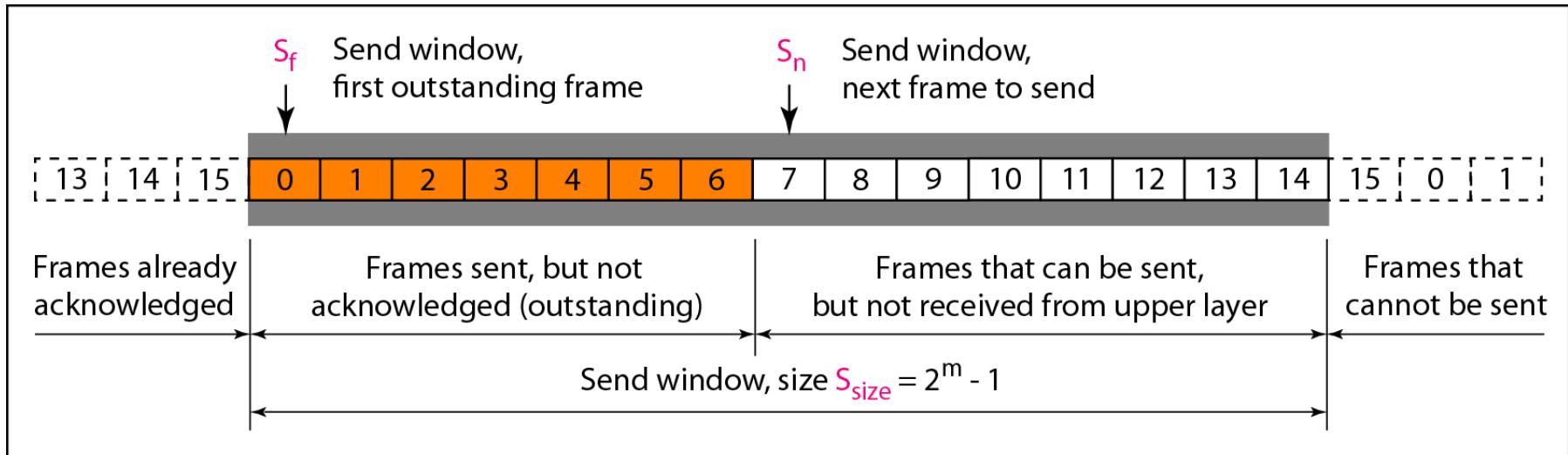
The bandwidth-delay product is still 20,000 bits. The system can send up to 15 frames or 15,000 bits during a round trip. This means the utilization is 15,000/20,000, or 75 percent. Of course, if there are damaged frames, the utilization percentage is much less because frames have to be resent.



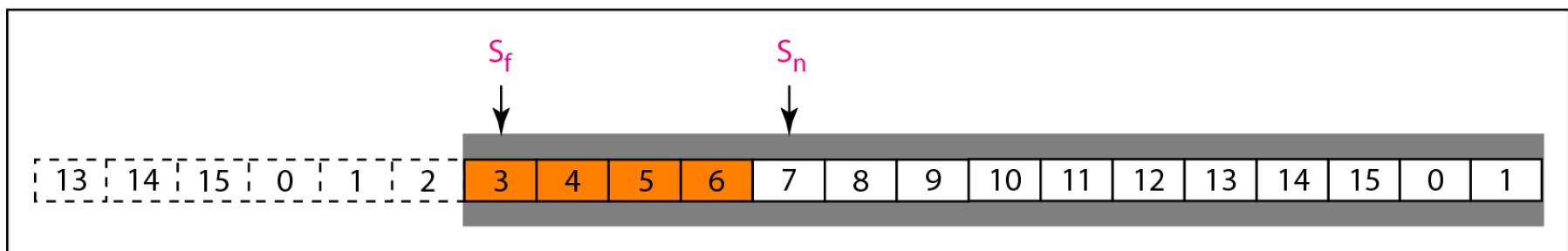
Note

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

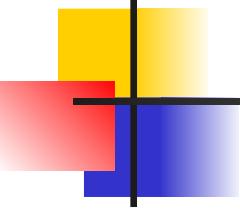
Figure 11.12 Send window for Go-Back-N ARQ



a. Send window before sliding

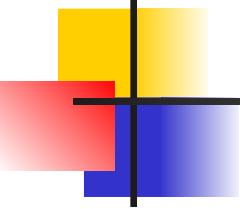


b. Send window after sliding



Note

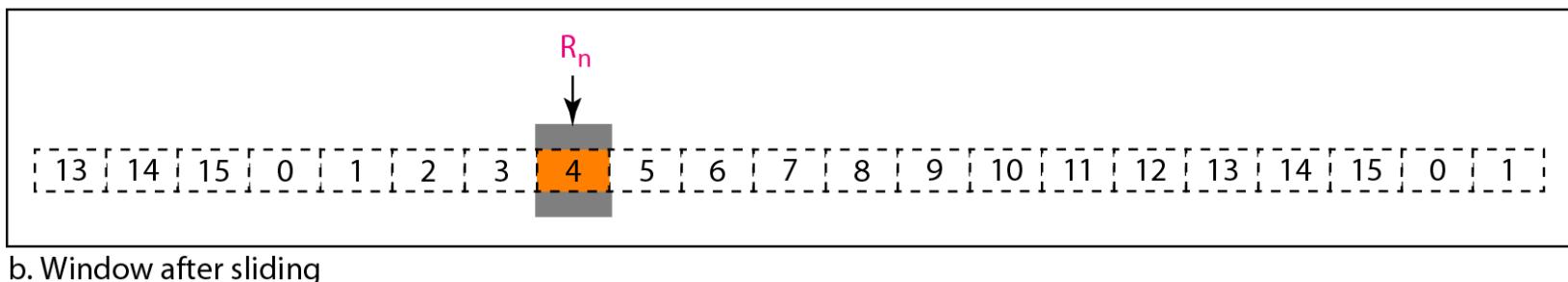
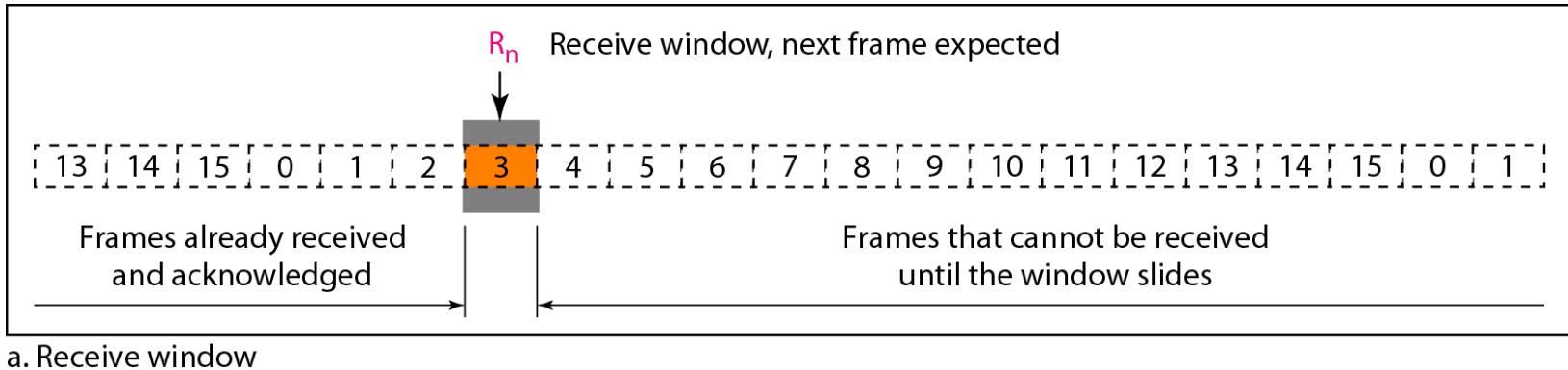
The send window is an abstract concept defining an imaginary box of size $2^m - 1$ with three variables: S_f , S_n , and S_{size} .

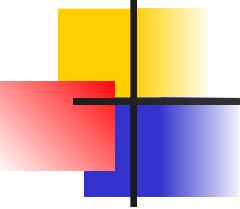


Note

The send window can slide one or more slots when a valid acknowledgment arrives.

Figure 11.13 Receive window for Go-Back-N ARQ





Note

The receive window is an abstract concept defining an imaginary box of size 1 with one single variable R_n .

The window slides when a correct frame has arrived; sliding occurs one slot at a time.

Figure 11.14 Design of Go-Back-N ARQ

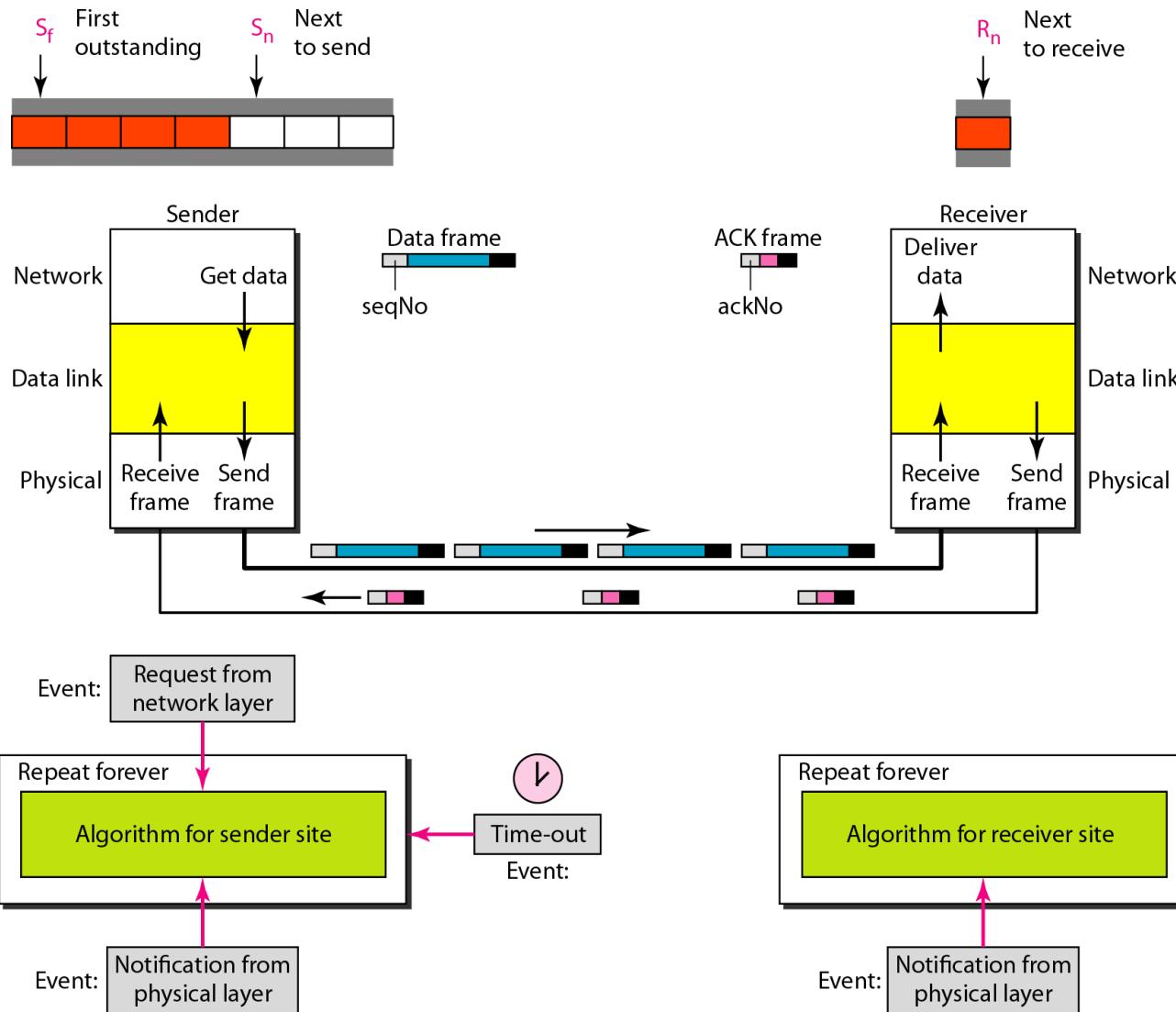
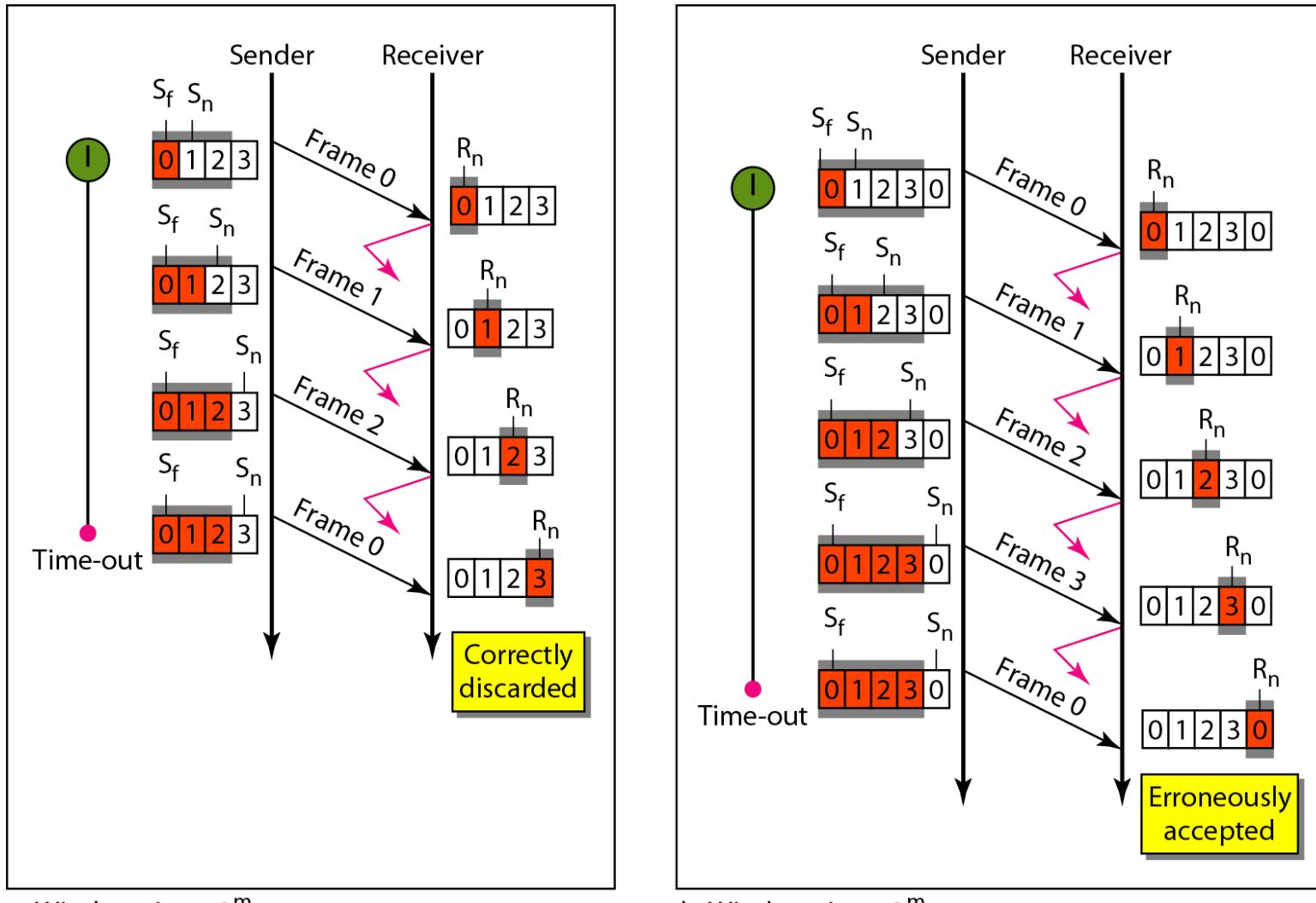
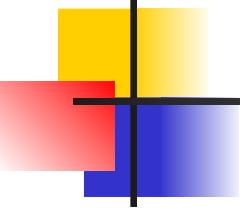


Figure 11.15 Window size for Go-Back-N ARQ



a. Window size $< 2^m$

b. Window size $= 2^m$



Note

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

Algorithm 11.7 Go-Back-N sender algorithm

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

(continued)

Algorithm 11.7 Go-Back-N sender algorithm

(continued)

```
21  if(Event(ArrivalNotification)) //ACK arrives
22  {
23      Receive(ACK);
24      if(corrupted(ACK))
25          Sleep();
26      if((ackNo>Sf)&&(ackNo<=Sn)) //If a valid ACK
27      While(Sf <= ackNo)
28      {
29          PurgeFrame(Sf);
30          Sf = Sf + 1;
31      }
32      StopTimer();
33  }

34

35  if(Event(TimeOut)) //The timer expires
36  {
37      StartTimer();
38      Temp = Sf;
39      while(Temp < Sn);
40      {
41          SendFrame(Sf);
42          Sf = Sf + 1;
43      }
44  }
45 }
```

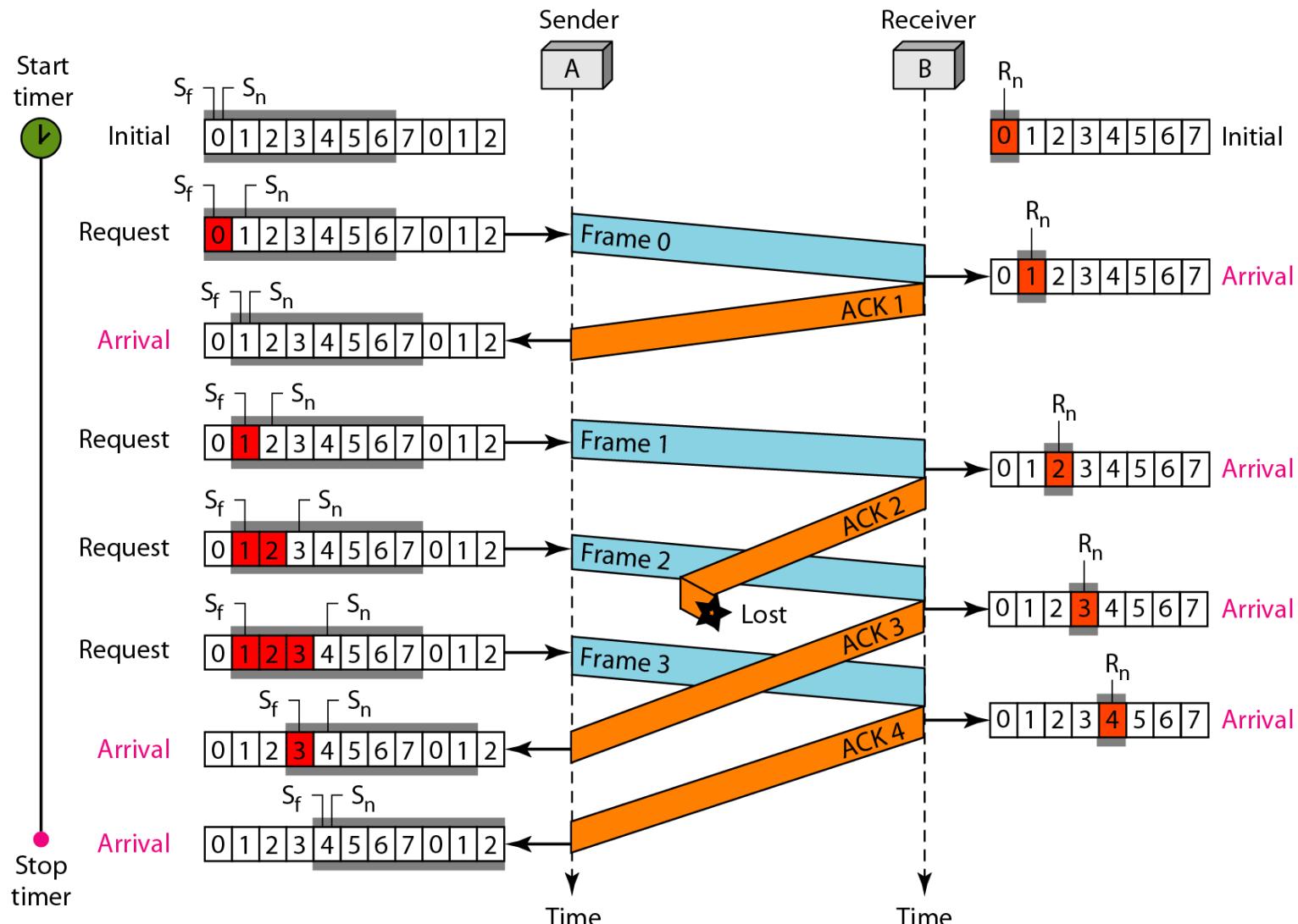
Algorithm 11.8 Go-Back-N receiver algorithm

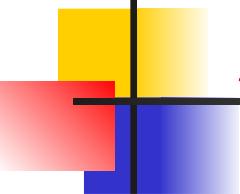
```
1 Rn = 0;  
2  
3 while (true) //Repeat forever  
4 {  
5     WaitForEvent();  
6  
7     if(Event(ArrivalNotification)) /Data frame arrives  
8     {  
9         Receive(Frame);  
10        if(corrupted(Frame))  
11            Sleep();  
12        if(seqNo == Rn) //If expected frame  
13       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

Example 11.6

Figure 11.16 shows an example of Go-Back-N. This is an example of a case where the forward channel is reliable, but the reverse is not. No data frames are lost, but some ACKs are delayed and one is lost. The example also shows how cumulative acknowledgments can help if acknowledgments are delayed or lost. After initialization, there are seven sender events. Request events are triggered by data from the network layer; arrival events are triggered by acknowledgments from the physical layer. There is no time-out event here because all outstanding frames are acknowledged before the timer expires. Note that although ACK 2 is lost, ACK 3 serves as both ACK 2 and ACK 3.

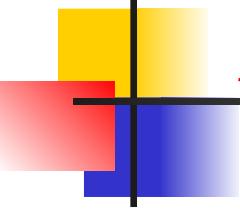
Figure 11.16 Flow diagram for Example 11.6





Example 11.7

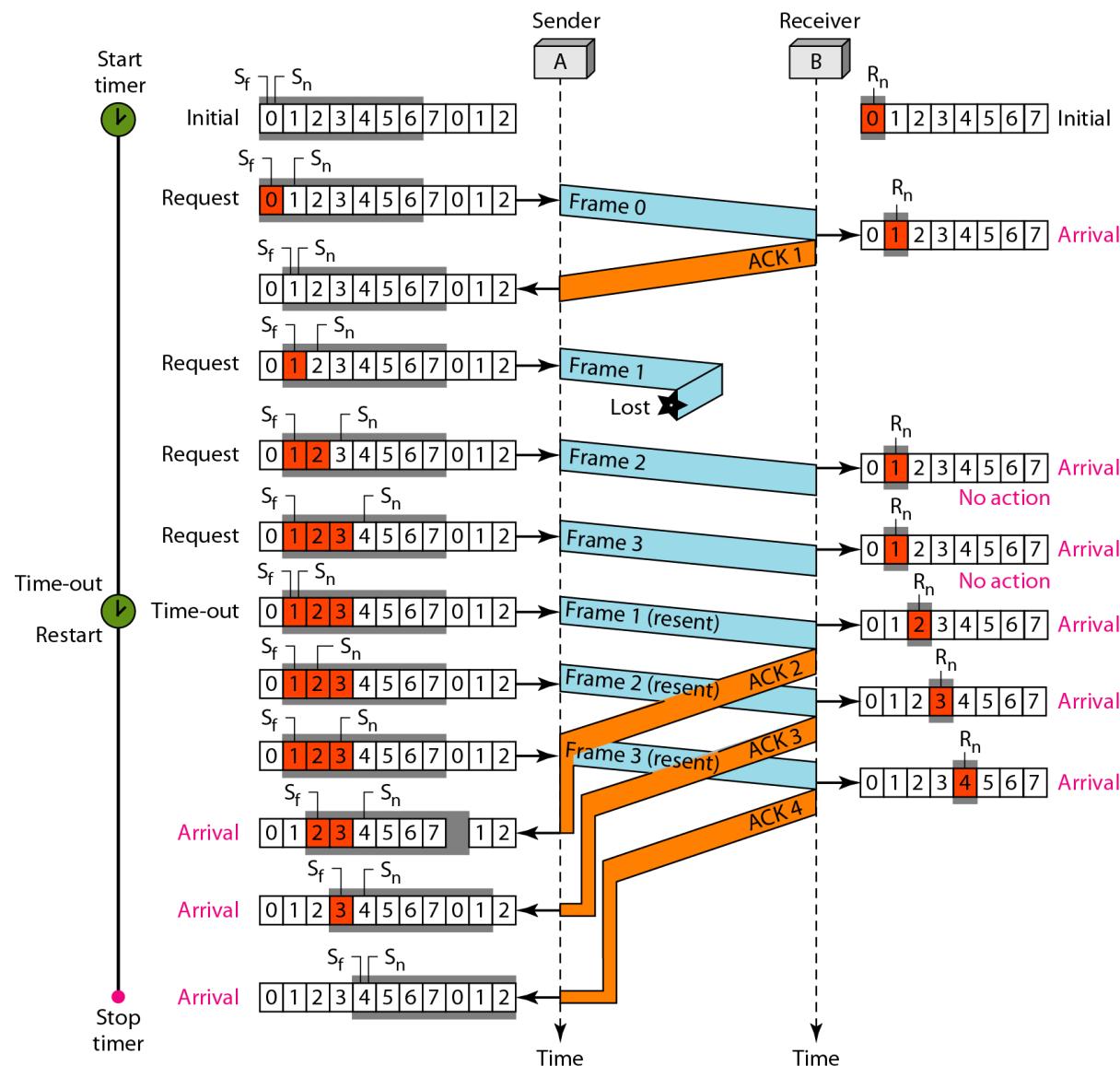
Figure 11.17 shows what happens when a frame is lost. Frames 0, 1, 2, and 3 are sent. However, frame 1 is lost. The receiver receives frames 2 and 3, but they are discarded because they are received out of order. The sender receives no acknowledgment about frames 1, 2, or 3. Its timer finally expires. The sender sends all outstanding frames (1, 2, and 3) because it does not know what is wrong. Note that the resending of frames 1, 2, and 3 is the response to one single event. When the sender is responding to this event, it cannot accept the triggering of other events. This means that when ACK 2 arrives, the sender is still busy with sending frame 3.

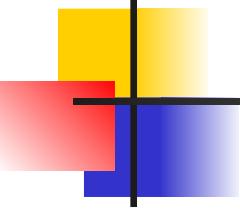


Example 11.7 (continued)

The physical layer must wait until this event is completed and the data link layer goes back to its sleeping state. We have shown a vertical line to indicate the delay. It is the same story with ACK 3; but when ACK 3 arrives, the sender is busy responding to ACK 2. It happens again when ACK 4 arrives. Note that before the second timer expires, all outstanding frames have been sent and the timer is stopped.

Figure 11.17 Flow diagram for Example 11.7





Note

Stop-and-Wait ARQ is a special case of Go-Back-N ARQ in which the size of the send window is 1.

Figure 11.18 Send window for Selective Repeat ARQ

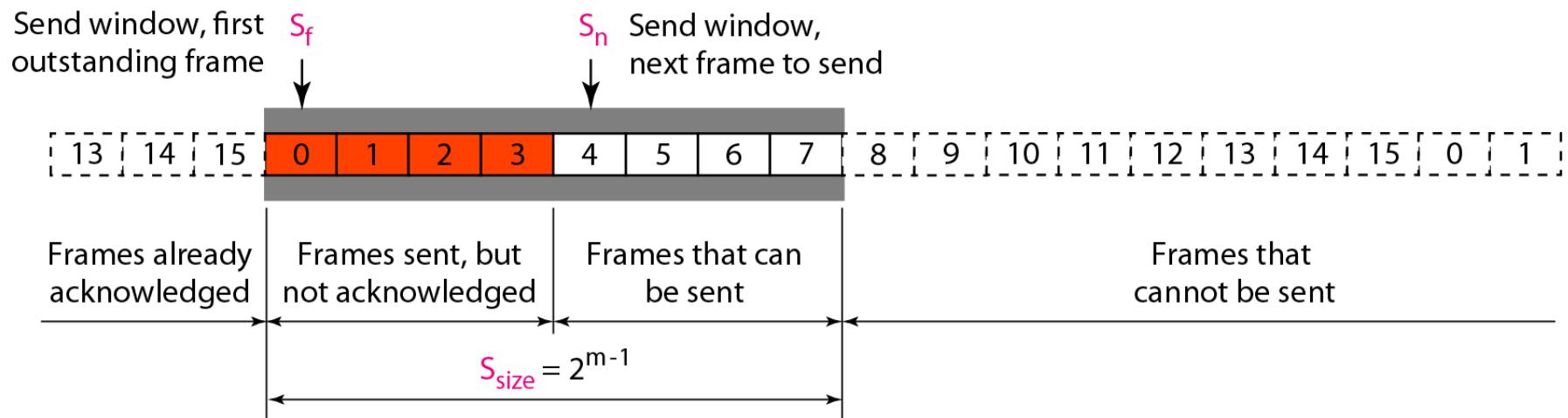


Figure 11.19 Receive window for Selective Repeat ARQ

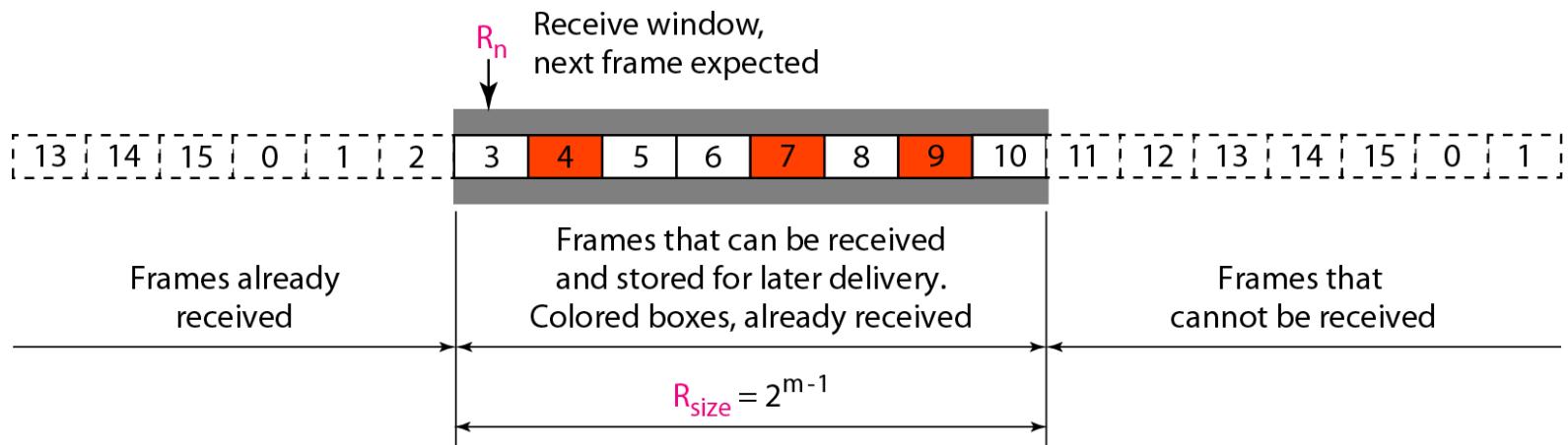


Figure 11.20 Design of Selective Repeat ARQ

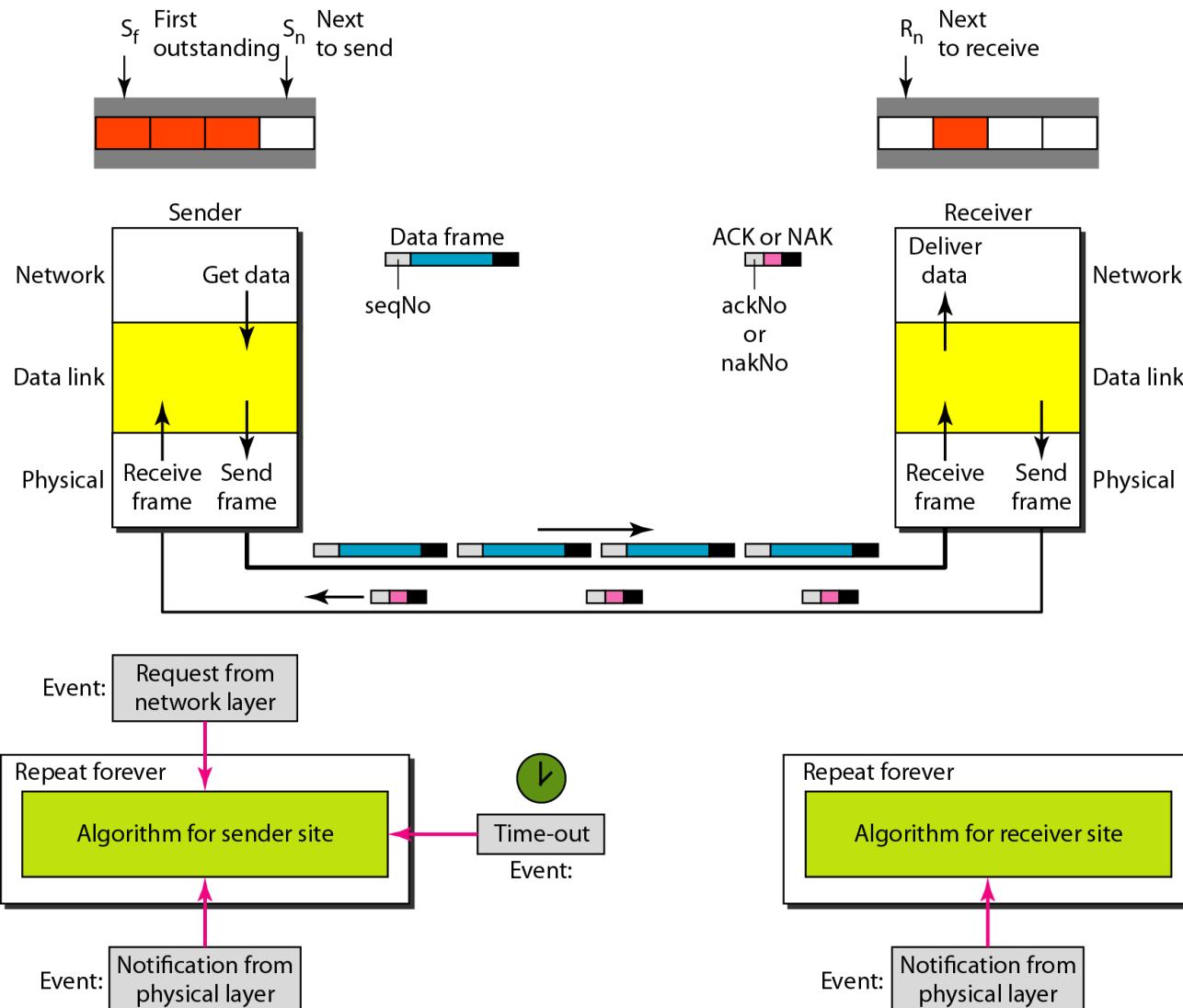
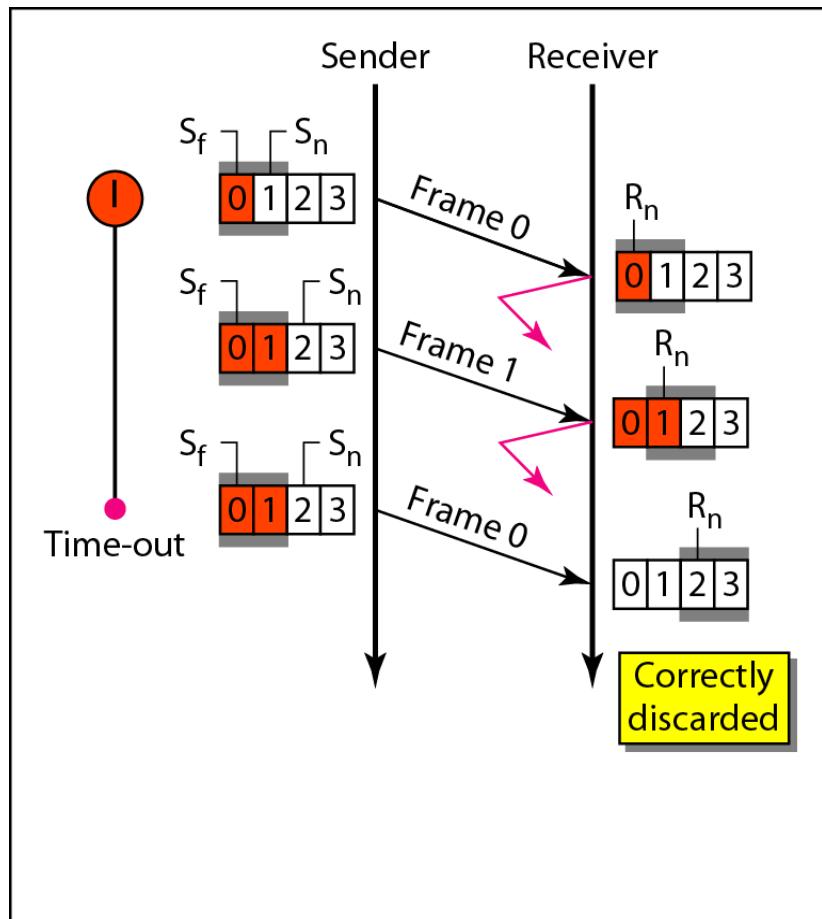
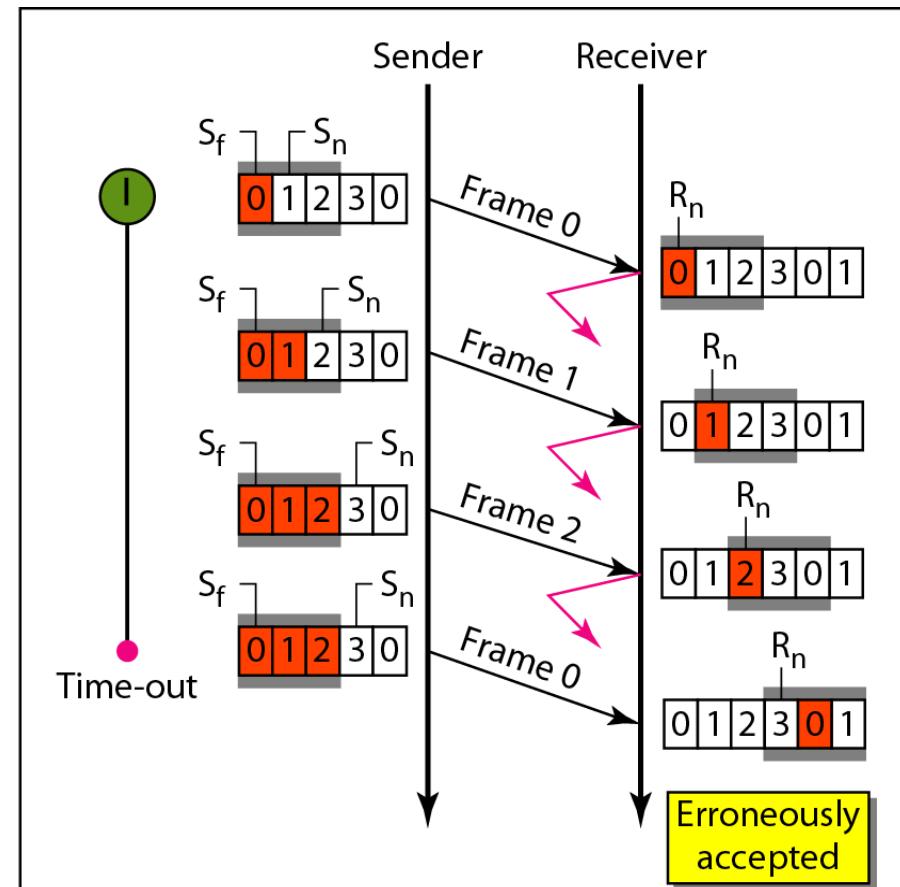


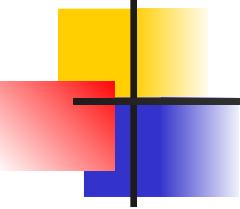
Figure 11.21 Selective Repeat ARQ, window size



a. Window size = 2^{m-1}



b. Window size > 2^{m-1}



Note

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

Algorithm 11.9 Sender-site Selective Repeat algorithm

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

(continued)

Algorithm 11.9 Sender-site Selective Repeat algorithm

(continued)

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

Algorithm 11.9 *Sender-site Selective Repeat algorithm*

(continued)

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

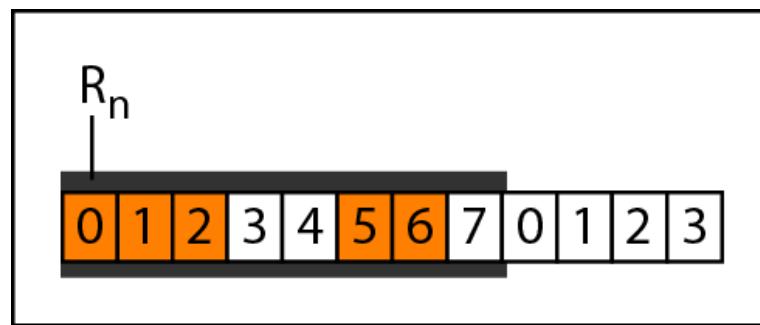
Algorithm 11.10 Receiver-site Selective Repeat algorithm

```
1 Rn = 0;
2 NakSent = false;
3 AckNeeded = false;
4 Repeat(for all slots)
5     Marked(slot) = false;
6
7 while (true)                                //Repeat forever
8 {
9     WaitForEvent();
10
11    if(Event(ArrivalNotification))           /Data frame arrives
12    {
13        Receive(Frame);
14        if(corrupted(Frame))&& (NOT NakSent)
15        {
16            SendNAK(Rn);
17            NakSent = true;
18            Sleep();
19        }
20        if(seqNo <> Rn)&& (NOT NakSent)
21        {
22            SendNAK(Rn);
```

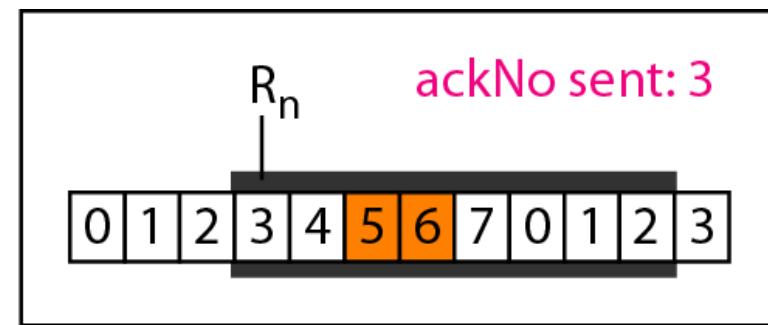
Algorithm 11.10 *Receiver-site Selective Repeat algorithm*

```
23     NakSent = true;
24     if ((seqNo in window) && (!Marked(seqNo)))
25     {
26         StoreFrame(seqNo)
27         Marked(seqNo)= true;
28         while(Marked(Rn))
29         {
30             DeliverData(Rn);
31             Purge(Rn);
32             Rn = Rn + 1;
33             AckNeeded = true;
34         }
35         if(AckNeeded);
36         {
37             SendAck(Rn);
38             AckNeeded = false;
39             NakSent = false;
40         }
41     }
42 }
43 }
44 }
```

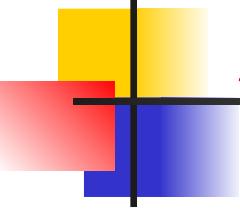
Figure 11.22 *Delivery of data in Selective Repeat ARQ*



a. Before delivery

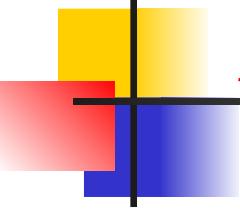


b. After delivery



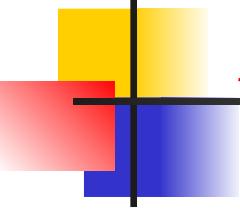
Example 11.8

This example is similar to Example 11.3 in which frame 1 is lost. We show how Selective Repeat behaves in this case. Figure 11.23 shows the situation. One main difference is the number of timers. Here, each frame sent or resent needs a timer, which means that the timers need to be numbered (0, 1, 2, and 3). The timer for frame 0 starts at the first request, but stops when the ACK for this frame arrives. The timer for frame 1 starts at the second request, restarts when a NAK arrives, and finally stops when the last ACK arrives. The other two timers start when the corresponding frames are sent and stop at the last arrival event.



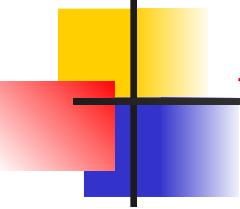
Example 11.8 (continued)

At the receiver site we need to distinguish between the acceptance of a frame and its delivery to the network layer. At the second arrival, frame 2 arrives and is stored and marked, but it cannot be delivered because frame 1 is missing. At the next arrival, frame 3 arrives and is marked and stored, but still none of the frames can be delivered. Only at the last arrival, when finally a copy of frame 1 arrives, can frames 1, 2, and 3 be delivered to the network layer. There are two conditions for the delivery of frames to the network layer: First, a set of consecutive frames must have arrived. Second, the set starts from the beginning of the window.



Example 11.8 (continued)

Another important point is that a NAK is sent after the second arrival, but not after the third, although both situations look the same. The reason is that the protocol does not want to crowd the network with unnecessary NAKs and unnecessary resent frames. The second NAK would still be NAK1 to inform the sender to resend frame 1 again; this has already been done. The first NAK sent is remembered (using the nakSent variable) and is not sent again until the frame slides. A NAK is sent once for each window position and defines the first slot in the window.



Example 11.8 (continued)

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

Figure 11.23 Flow diagram for Example 11.8

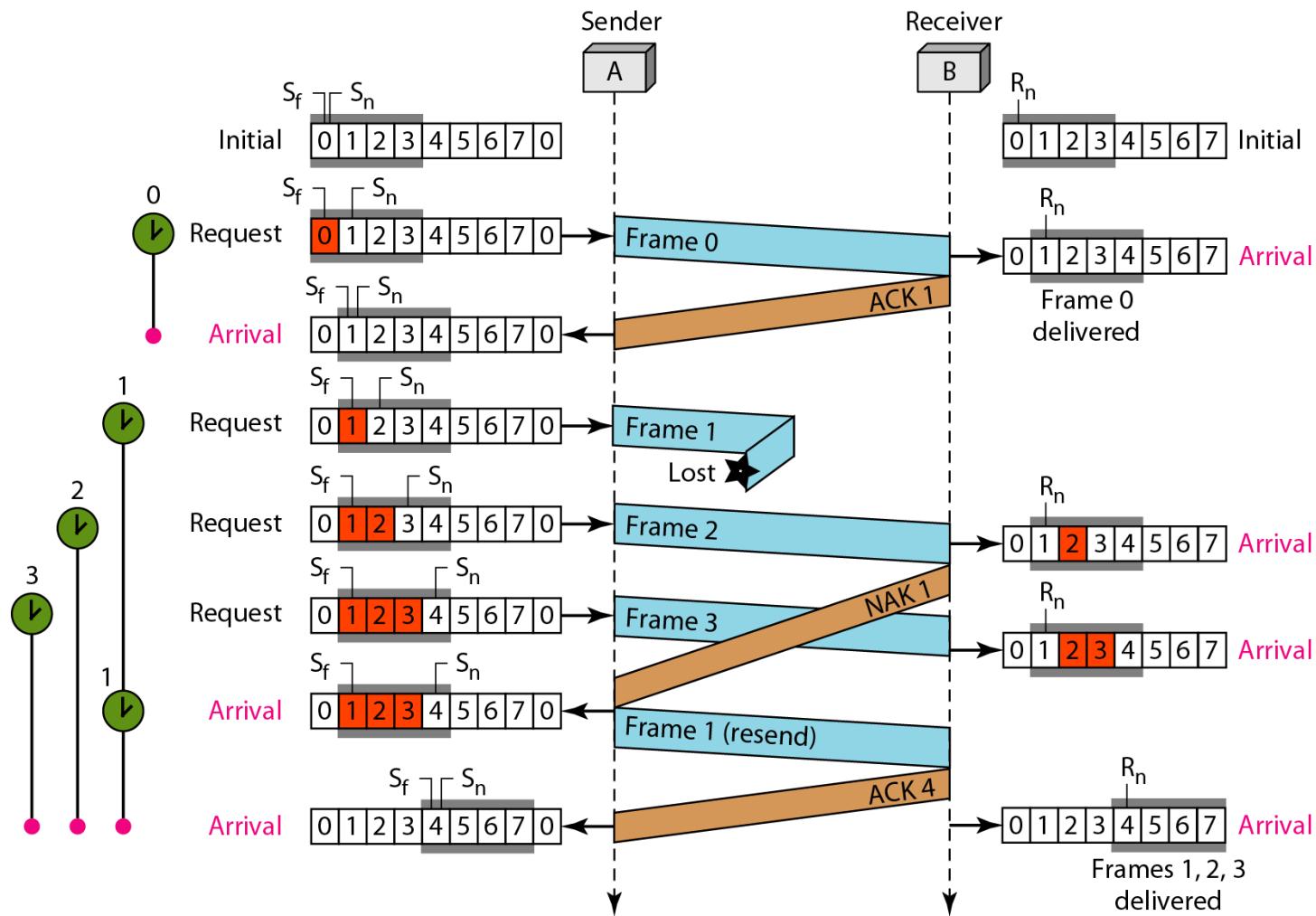
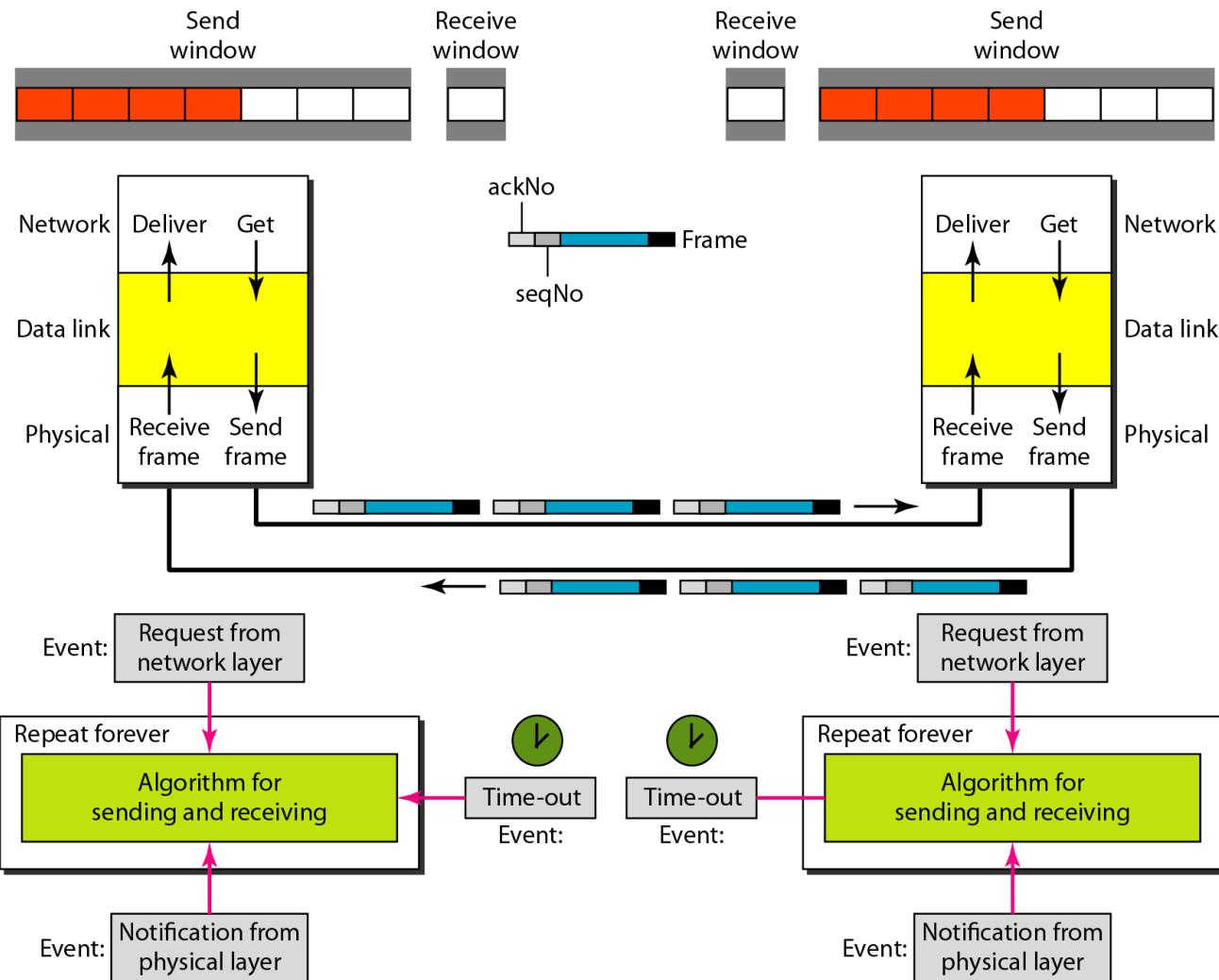


Figure 11.24 Design of piggybacking in Go-Back-N ARQ



11-6 HDLC

High-level Data Link Control (HDLC) is a bit-oriented protocol for communication over point-to-point and multipoint links. It implements the ARQ mechanisms we discussed in this chapter.

Topics discussed in this section:

Configurations and Transfer Modes

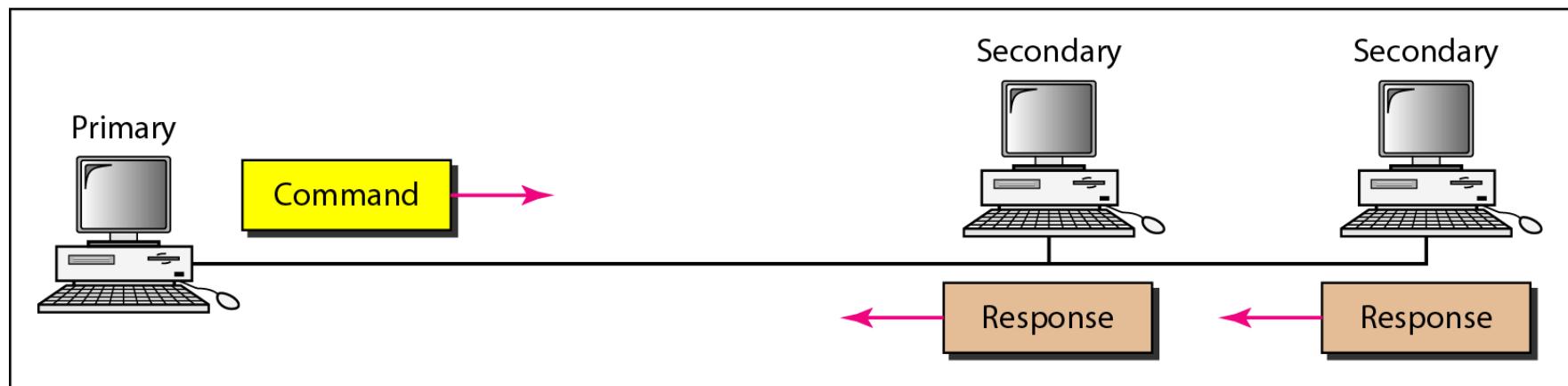
Frames

Control Field

Figure 11.25 Normal response mode



a. Point-to-point



b. Multipoint

Figure 11.26 Asynchronous balanced mode



Figure 11.27 HDLC frames

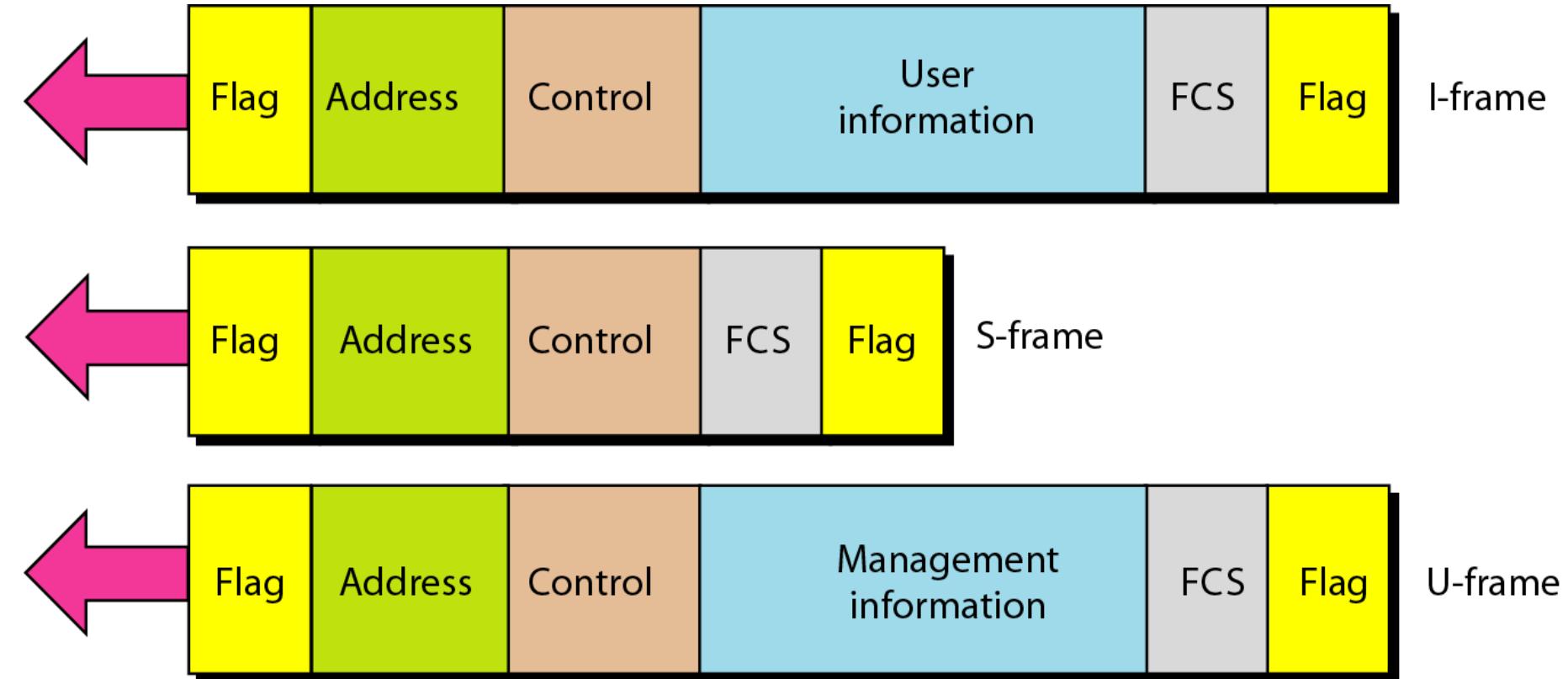


Figure 11.28 Control field format for the different frame types

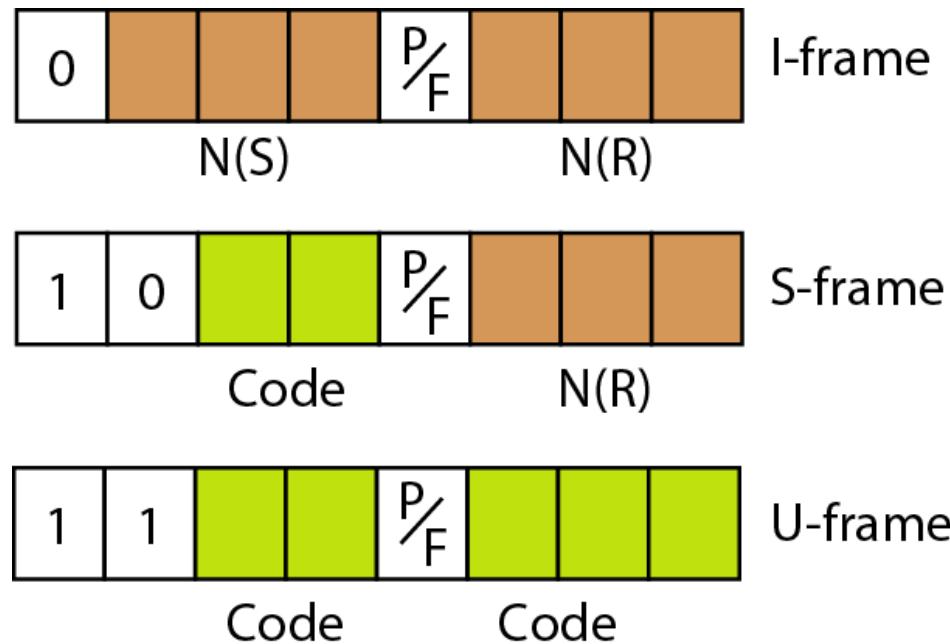
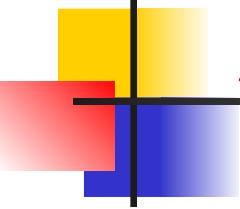


Table 11.1 U-frame control command and response

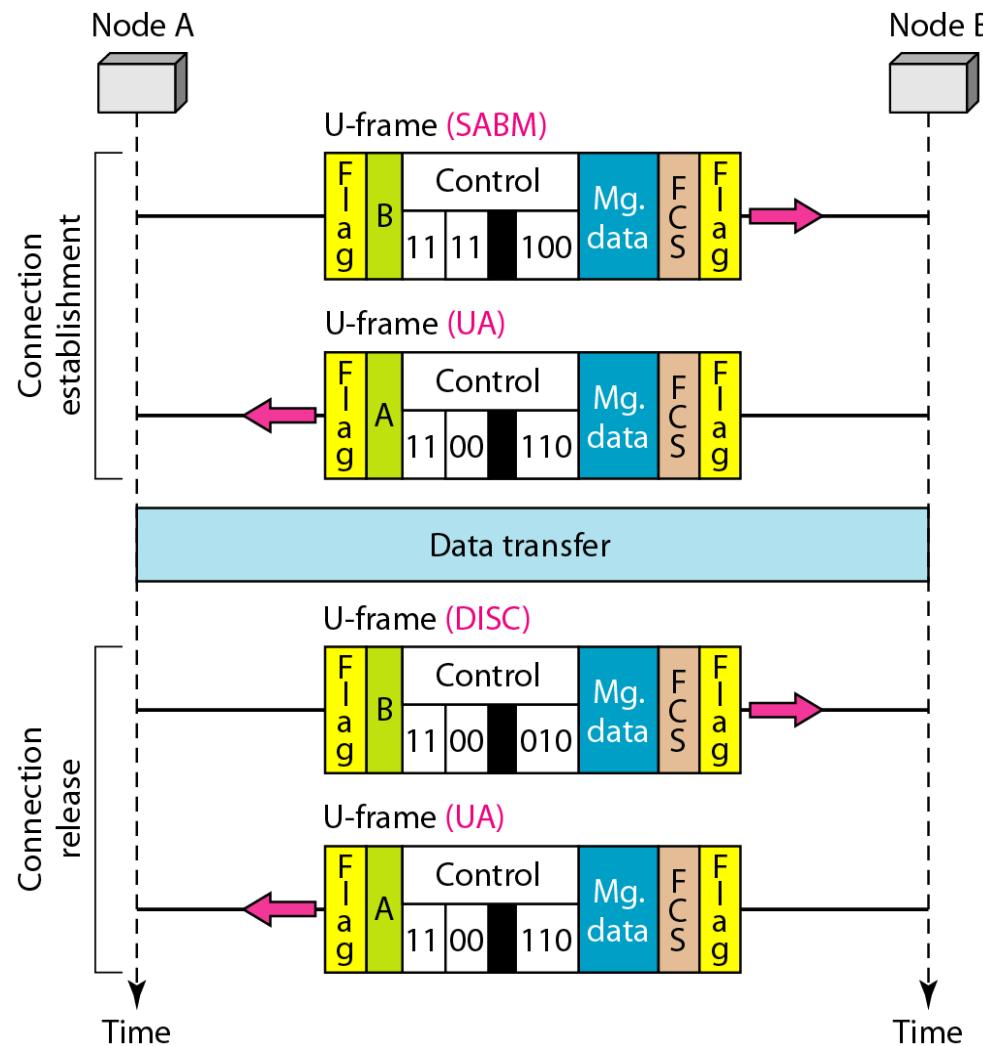
<i>Code</i>	<i>Command</i>	<i>Response</i>	<i>Meaning</i>
00 001	SNRM		Set normal response mode
11 011	SNRME		Set normal response mode, extended
11 100	SABM	DM	Set asynchronous balanced mode or disconnect mode
11 110	SABME		Set asynchronous balanced mode, extended
00 000	UI	UI	Unnumbered information
00 110		UA	Unnumbered acknowledgment
00 010	DISC	RD	Disconnect or request disconnect
10 000	SIM	RIM	Set initialization mode or request information mode
00 100	UP		Unnumbered poll
11 001	RSET		Reset
11 101	XID	XID	Exchange ID
10 001	FRMR	FRMR	Frame reject

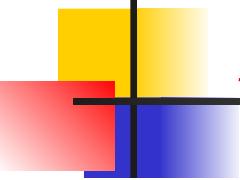


Example 11.9

*Figure 11.29 shows how **U-frames** can be used for connection establishment and connection release. Node A asks for a connection with a set asynchronous balanced mode (SABM) frame; node B gives a positive response with an unnumbered acknowledgment (UA) frame. After these two exchanges, data can be transferred between the two nodes (not shown in the figure). After data transfer, node A sends a DISC (disconnect) frame to release the connection; it is confirmed by node B responding with a UA (unnumbered acknowledgment).*

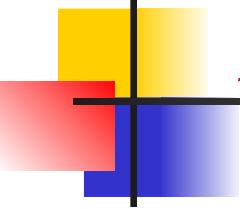
Figure 11.29 Example of connection and disconnection





Example 11.10

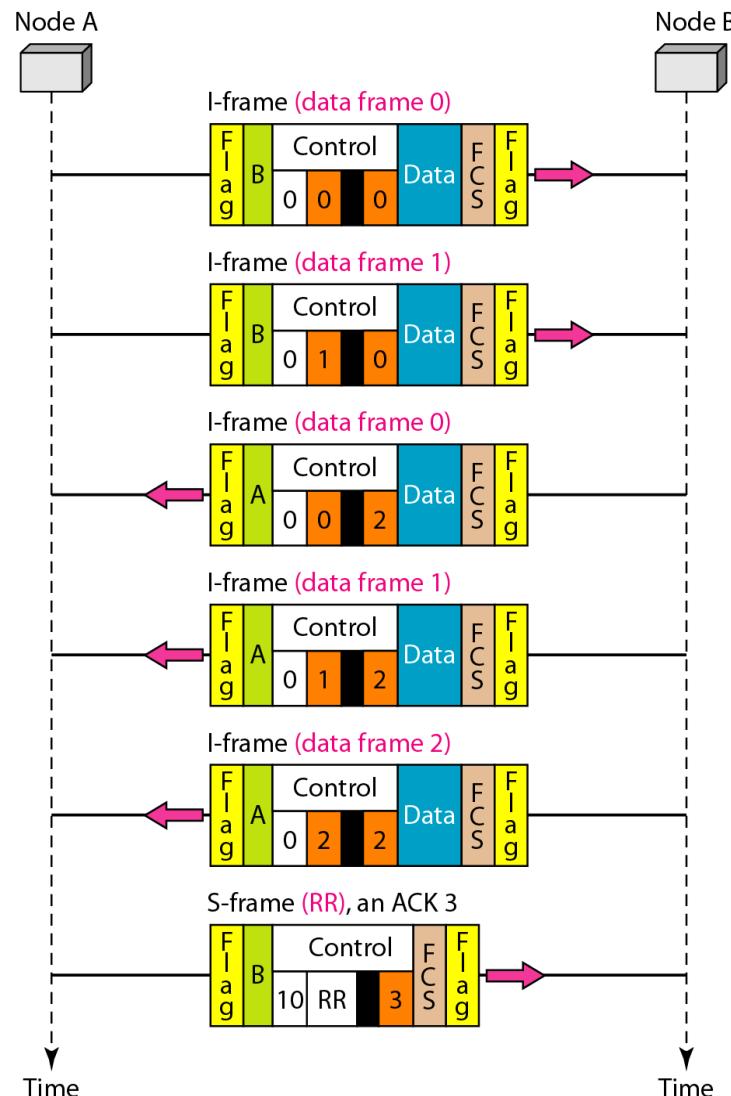
Figure 11.30 shows an exchange using piggybacking. Node A begins the exchange of information with an I-frame numbered 0 followed by another I-frame numbered 1. Node B piggybacks its acknowledgment of both frames onto an I-frame of its own. Node B's first I-frame is also numbered 0 [N(S) field] and contains a 2 in its N(R) field, acknowledging the receipt of A's frames 1 and 0 and indicating that it expects frame 2 to arrive next. Node B transmits its second and third I-frames (numbered 1 and 2) before accepting further frames from node A.

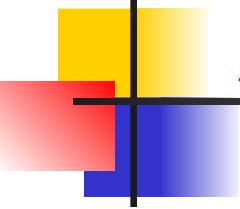


Example 11.10 (continued)

Its $N(R)$ information, therefore, has not changed: B frames 1 and 2 indicate that node B is still expecting A's frame 2 to arrive next. Node A has sent all its data. Therefore, it cannot piggyback an acknowledgment onto an I-frame and sends an S-frame instead. The RR code indicates that A is still ready to receive. The number 3 in the $N(R)$ field tells B that frames 0, 1, and 2 have all been accepted and that A is now expecting frame number 3.

Figure 11.30 Example of piggybacking without error

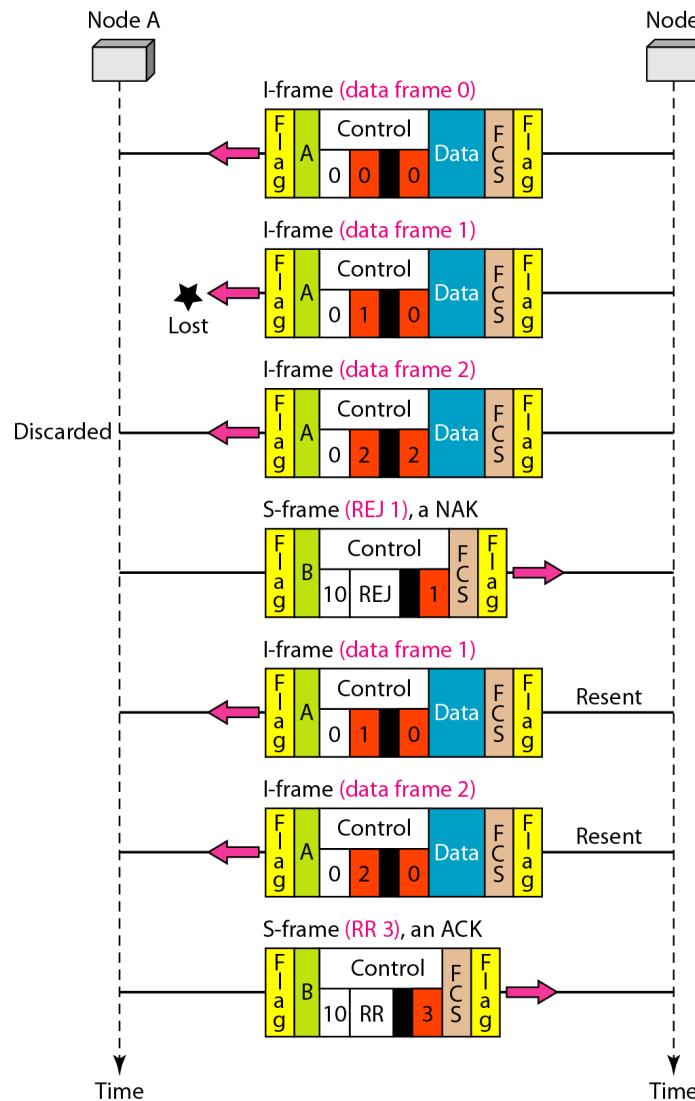




Example 11.11

Figure 11.31 shows an exchange in which a frame is lost. Node B sends three data frames (0, 1, and 2), but frame 1 is lost. When node A receives frame 2, it discards it and sends a REJ frame for frame 1. Note that the protocol being used is Go-Back-N with the special use of an REJ frame as a NAK frame. The NAK frame does two things here: It confirms the receipt of frame 0 and declares that frame 1 and any following frames must be resent. Node B, after receiving the REJ frame, resends frames 1 and 2. Node A acknowledges the receipt by sending an RR frame (ACK) with acknowledgment number 3.

Figure 11.31 Example of piggybacking with error



11-7 POINT-TO-POINT PROTOCOL

*Although HDLC is a general protocol that can be used for both point-to-point and multipoint configurations, one of the most common protocols for point-to-point access is the **Point-to-Point Protocol (PPP)**. PPP is a byte-oriented protocol.*

Topics discussed in this section:

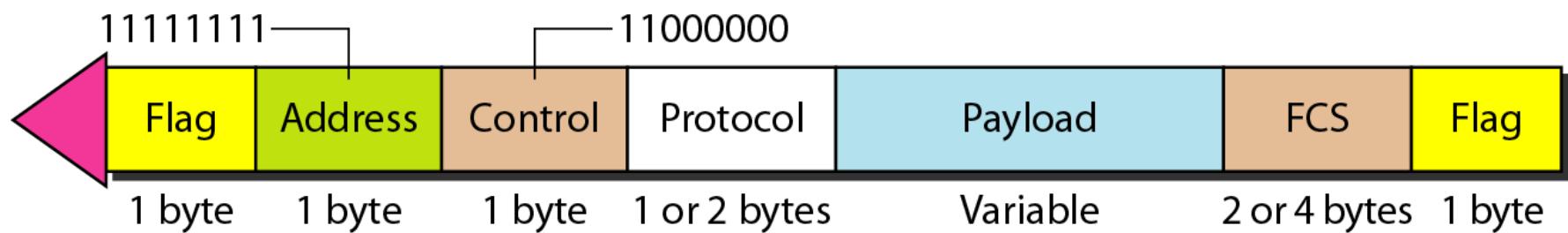
Framing

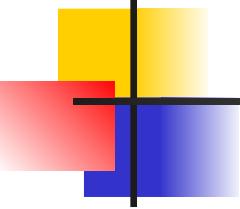
Transition Phases

Multiplexing

Multilink PPP

Figure 11.32 PPP frame format





Note

PPP is a byte-oriented protocol using byte stuffing with the escape byte 01111101.

Figure 11.33 Transition phases

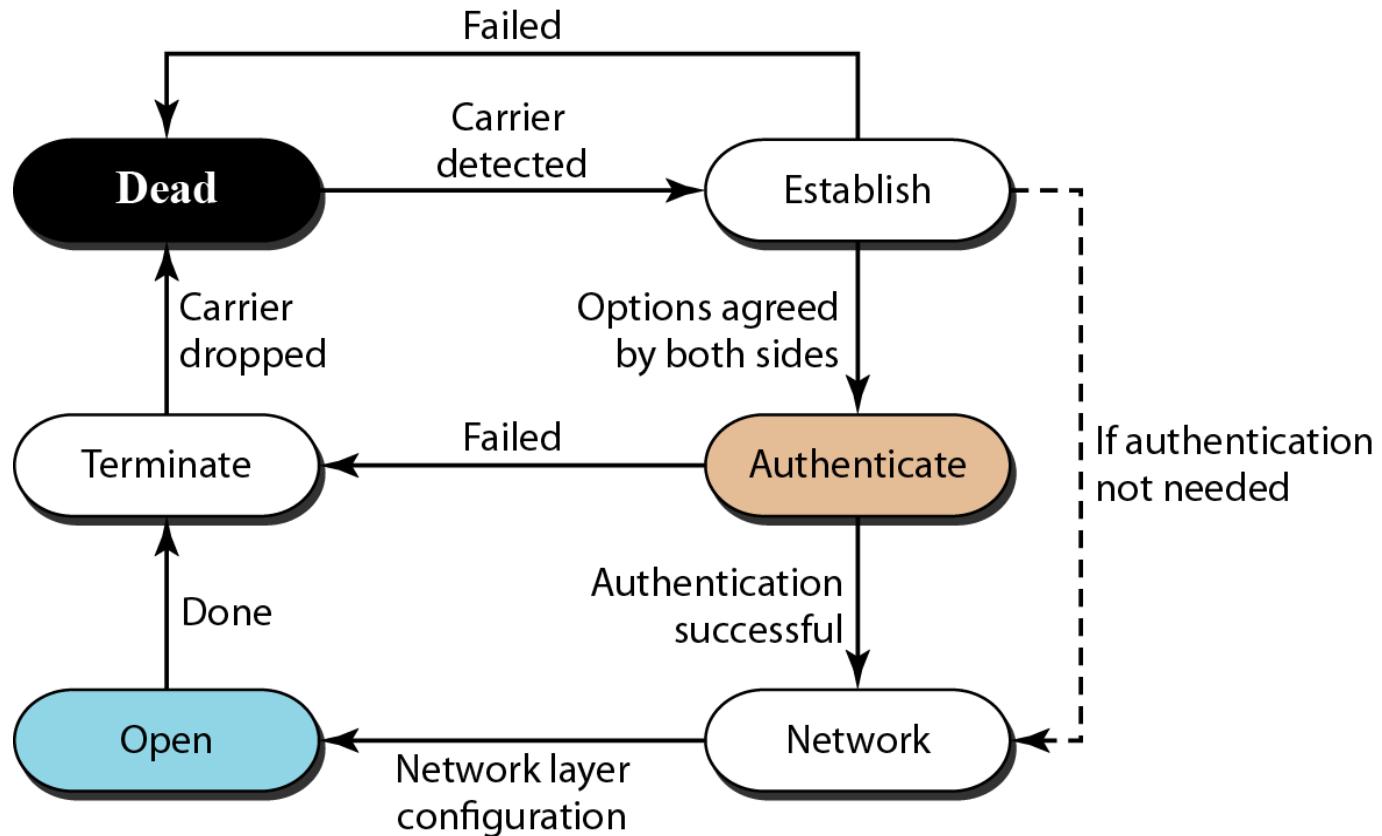


Figure 11.34 Multiplexing in PPP

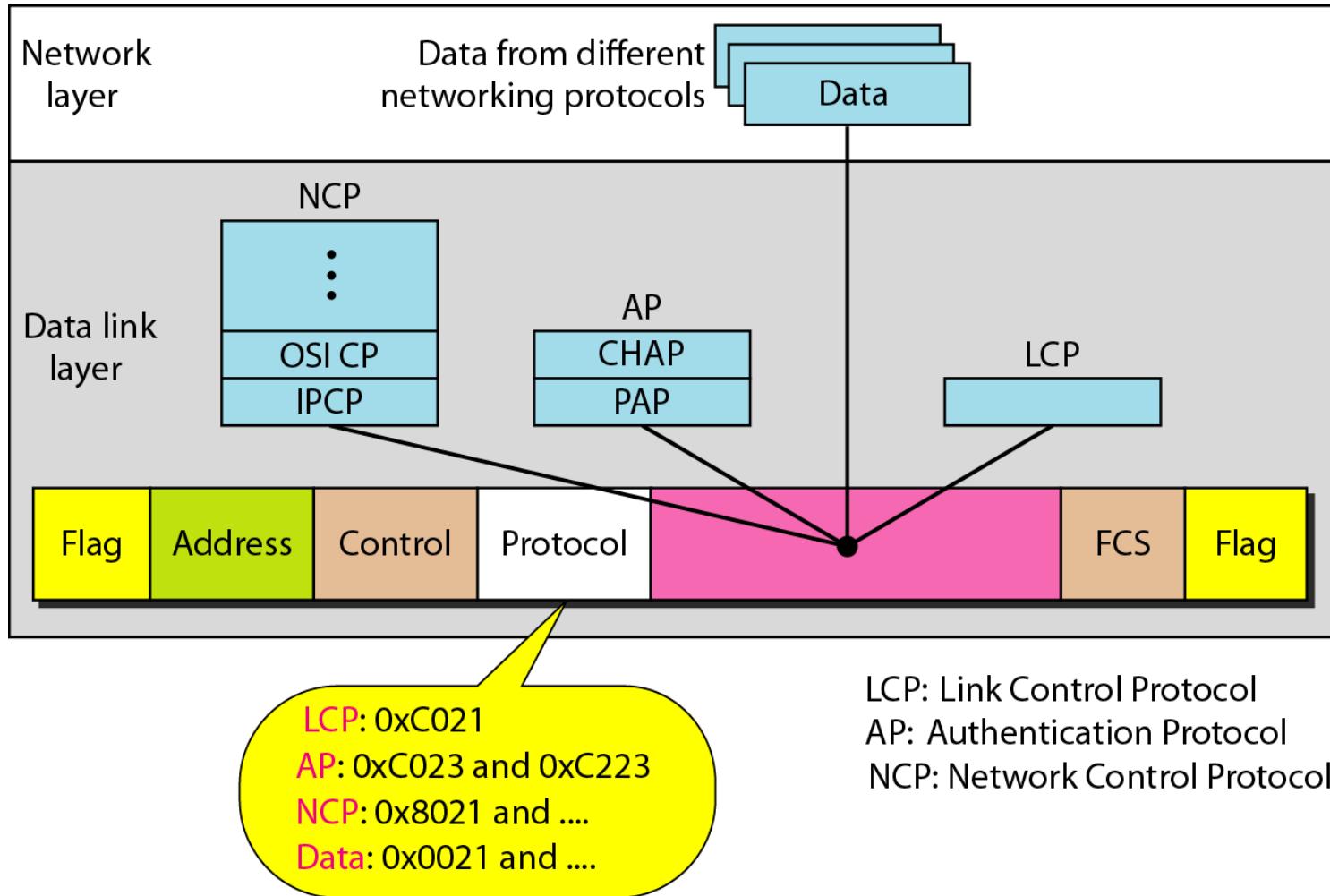


Figure 11.35 LCP packet encapsulated in a frame

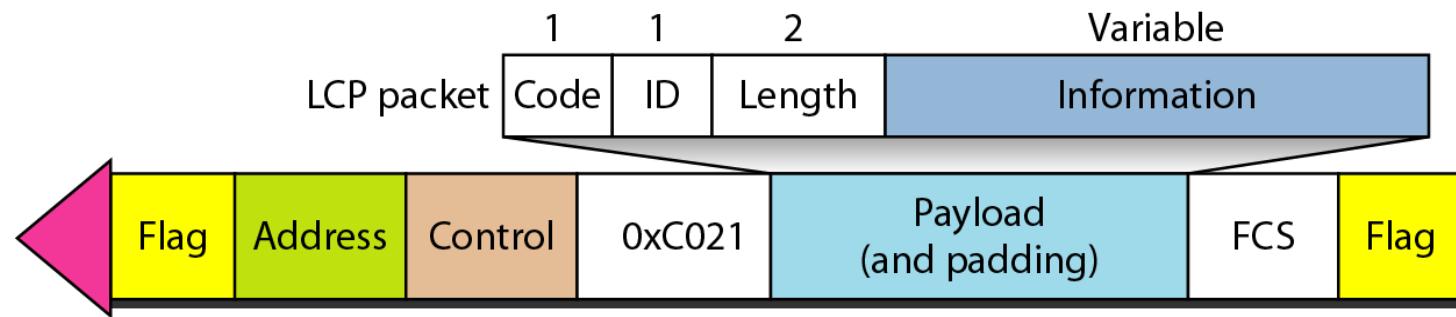


Table 11.2 LCP packets

<i>Code</i>	<i>Packet Type</i>	<i>Description</i>
0x01	Configure-request	Contains the list of proposed options and their values
0x02	Configure-ack	Accepts all options proposed
0x03	Configure-nak	Announces that some options are not acceptable
0x04	Configure-reject	Announces that some options are not recognized
0x05	Terminate-request	Request to shut down the line
0x06	Terminate-ack	Accept the shutdown request
0x07	Code-reject	Announces an unknown code
0x08	Protocol-reject	Announces an unknown protocol
0x09	Echo-request	A type of hello message to check if the other end is alive
0x0A	Echo-reply	The response to the echo-request message
0x0B	Discard-request	A request to discard the packet

Table 11.3 *Common options*

<i>Option</i>	<i>Default</i>
Maximum receive unit (payload field size)	1500
Authentication protocol	None
Protocol field compression	Off
Address and control field compression	Off

Figure 11.36 PAP packets encapsulated in a PPP frame

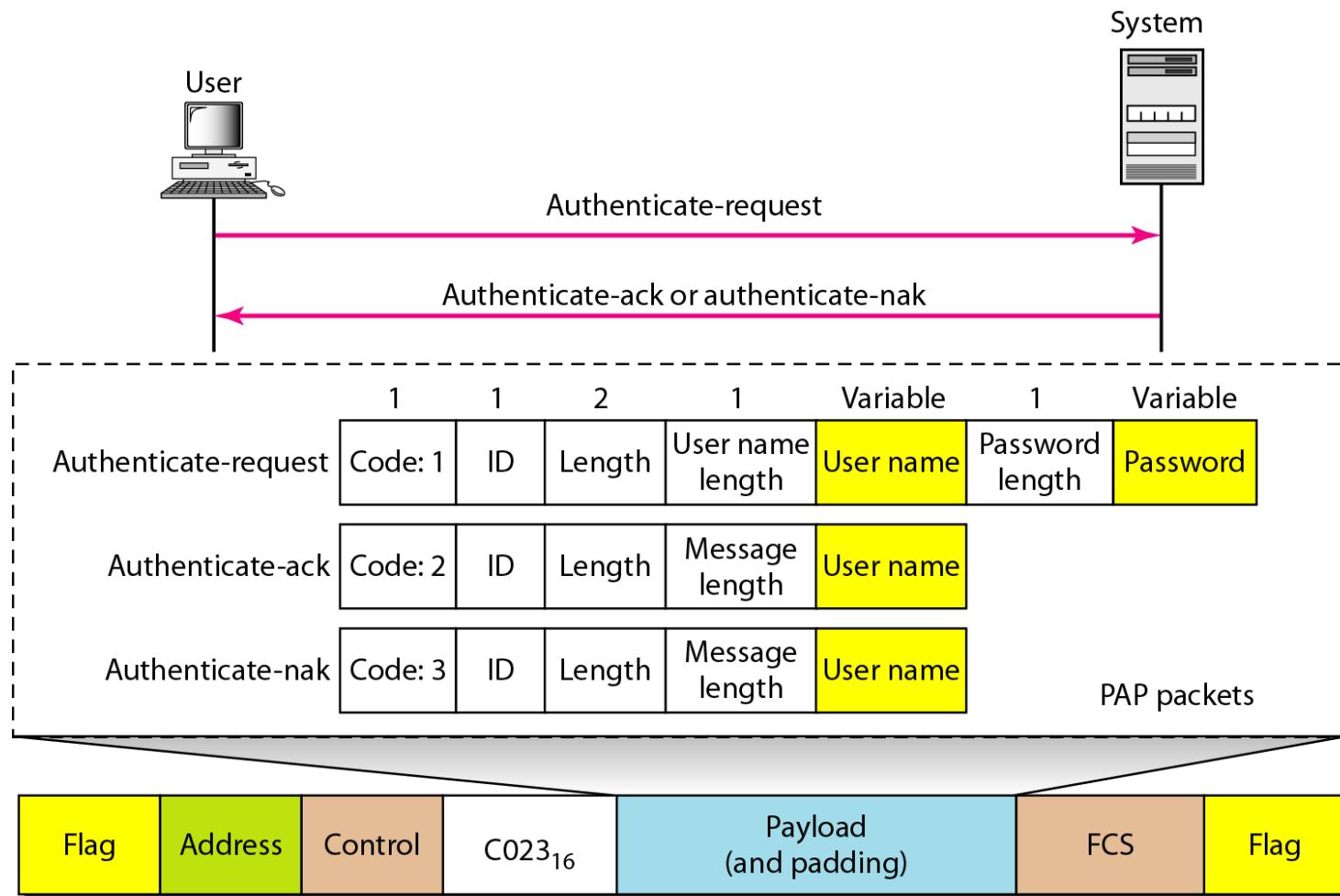


Figure 11.37 CHAP packets encapsulated in a PPP frame

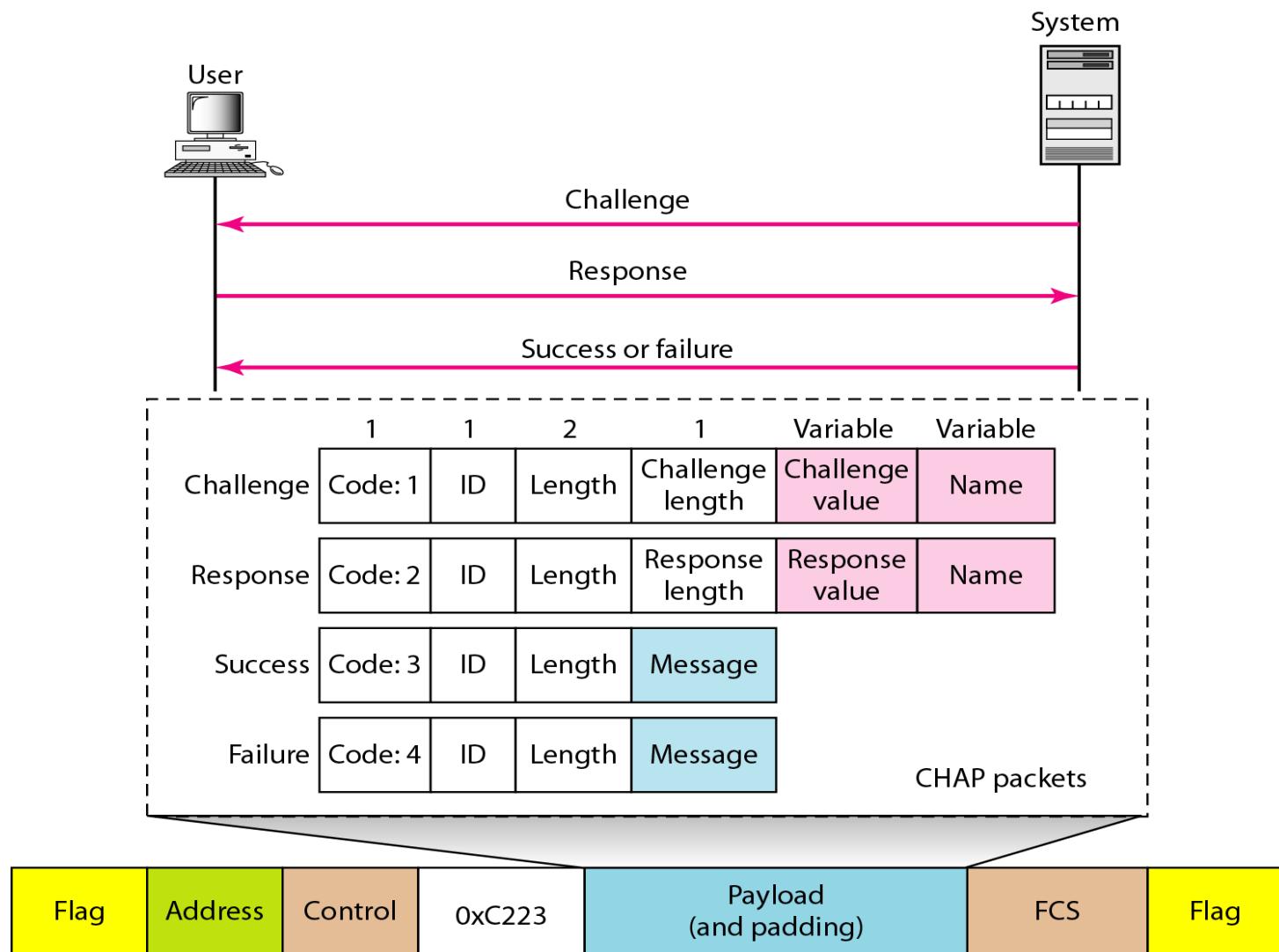


Figure 11.38 IPCP packet encapsulated in PPP frame

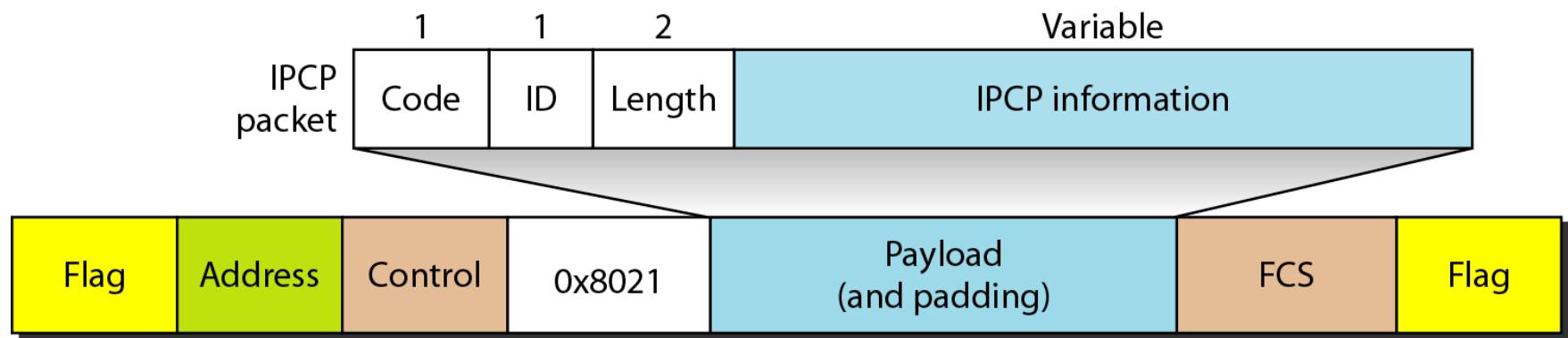


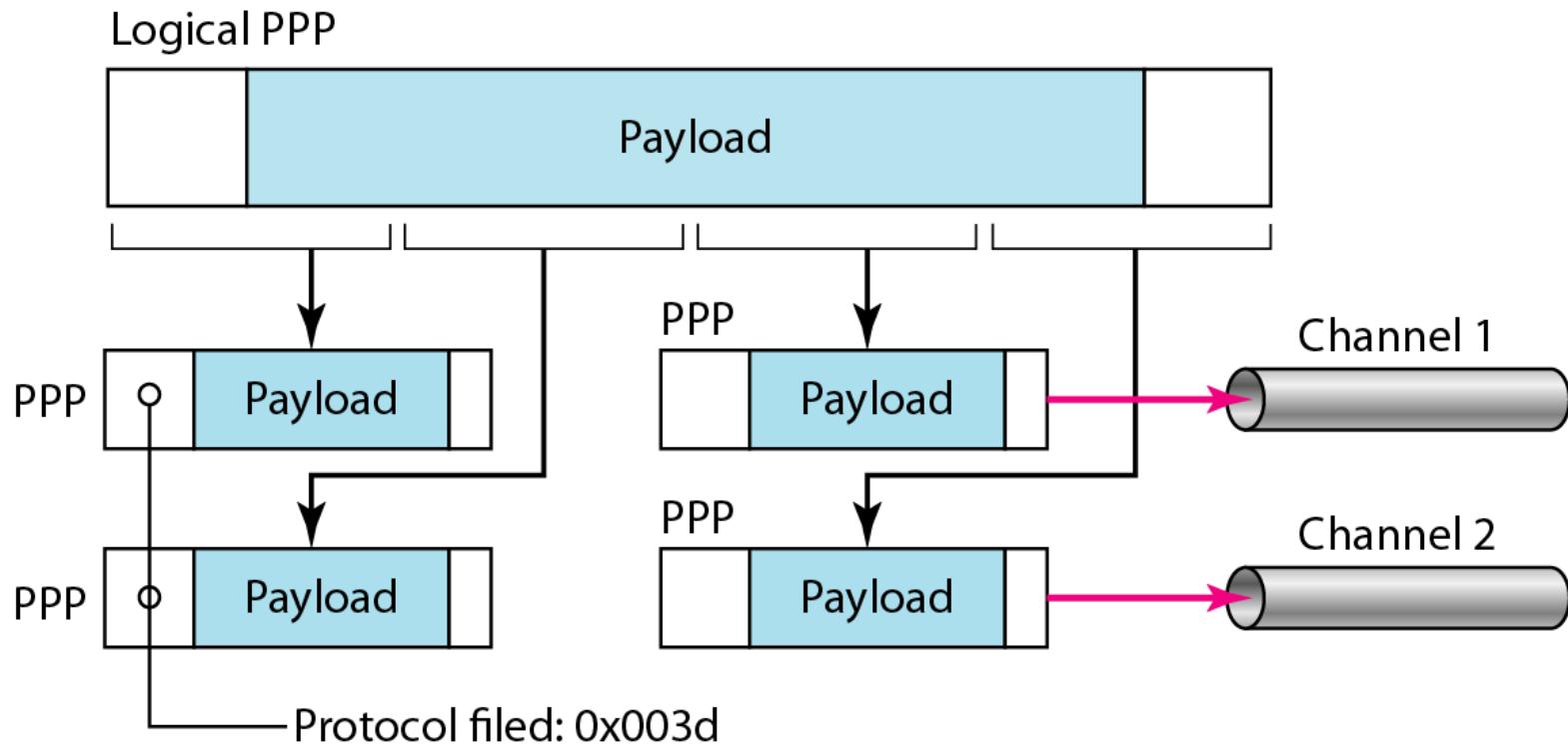
Table 11.4 *Code value for IPCP packets*

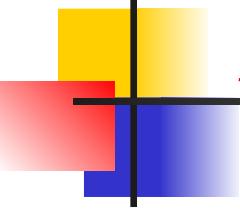
<i>Code</i>	<i>IPCP Packet</i>
0x01	Configure-request
0x02	Configure-ack
0x03	Configure-nak
0x04	Configure-reject
0x05	Terminate-request
0x06	Terminate-ack
0x07	Code-reject

Figure 11.39 *IP datagram encapsulated in a PPP frame*



Figure 11.40 Multilink PPP

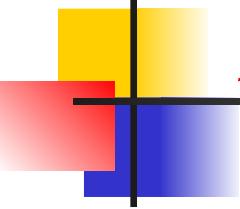




Example 11.12

Let us go through the phases followed by a network layer packet as it is transmitted through a PPP connection. Figure 11.41 shows the steps. For simplicity, we assume unidirectional movement of data from the user site to the system site (such as sending an e-mail through an ISP).

The first two frames show link establishment. We have chosen two options (not shown in the figure): using PAP for authentication and suppressing the address control fields. Frames 3 and 4 are for authentication. Frames 5 and 6 establish the network layer connection using IPCP.



Example 11.12 (continued)

The next several frames show that some IP packets are encapsulated in the PPP frame. The system (receiver) may have been running several network layer protocols, but it knows that the incoming data must be delivered to the IP protocol because the NCP protocol used before the data transfer was IPCP.

After data transfer, the user then terminates the data link connection, which is acknowledged by the system. Of course the user or the system could have chosen to terminate the network layer IPCP and keep the data link layer running if it wanted to run another NCP protocol.

Figure 11.41 An example

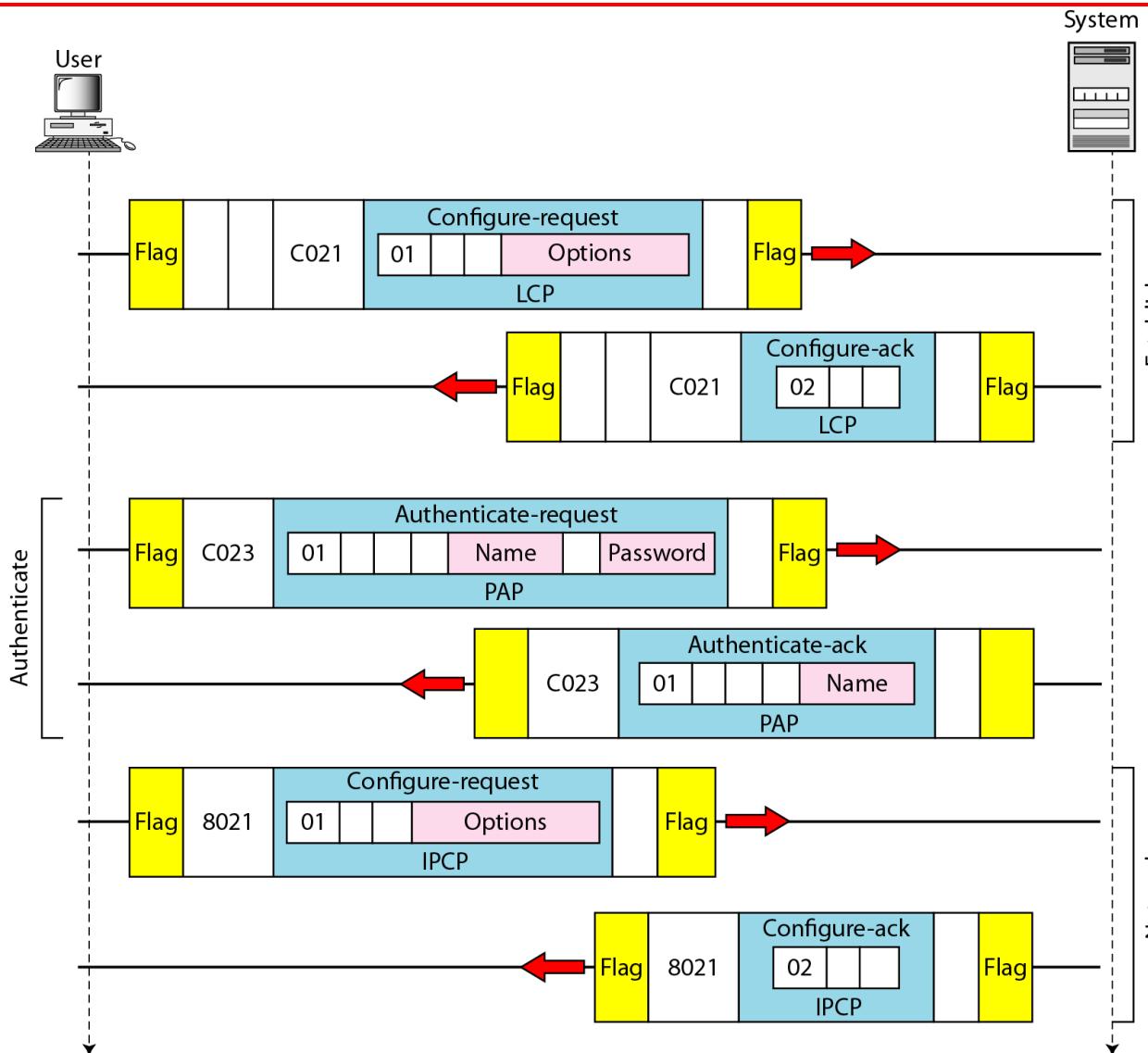
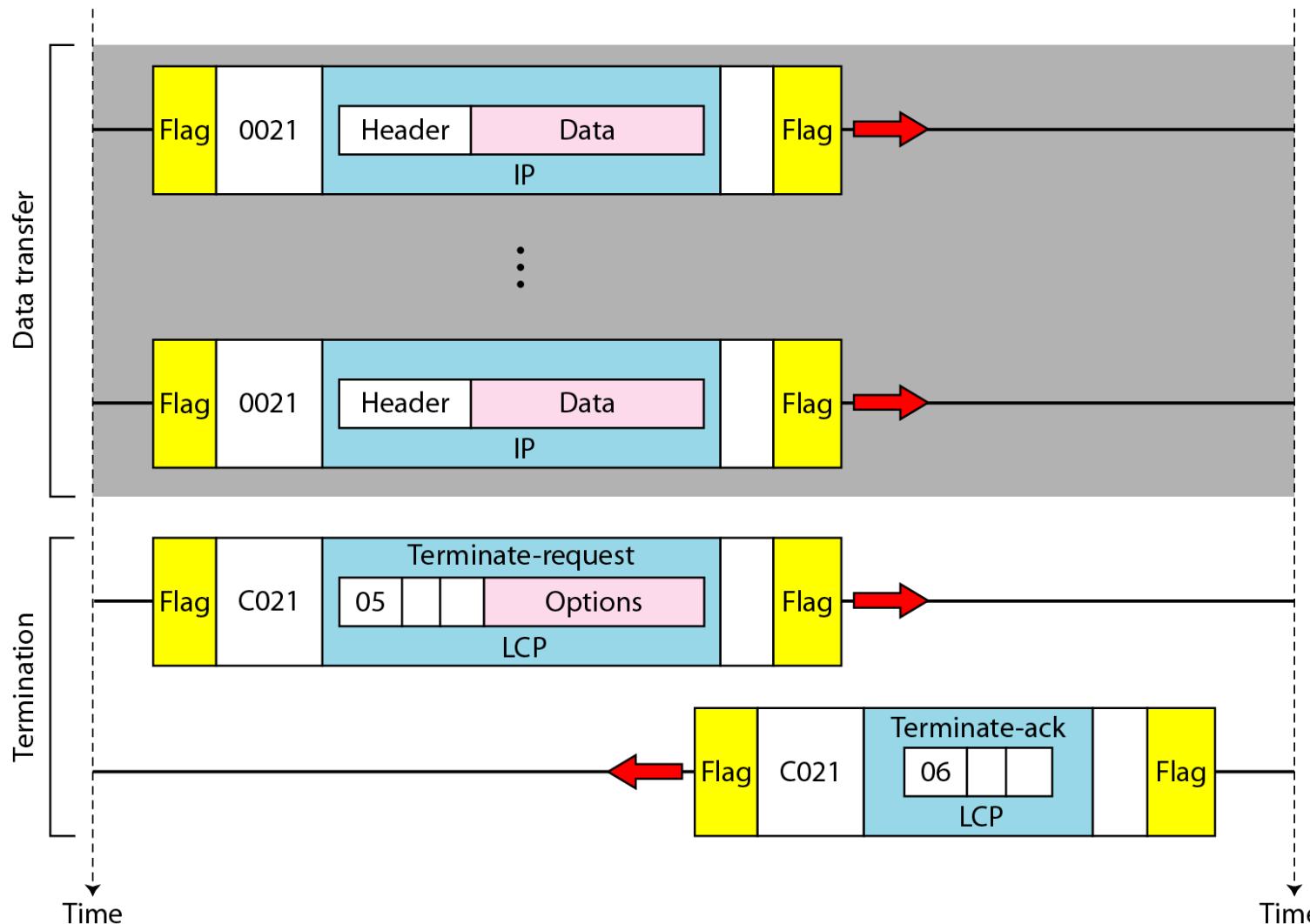


Figure 11.41 An example (continued)



Chapter 12

Multiple Access

Figure 12.1 *Data link layer divided into two functionality-oriented sublayers*

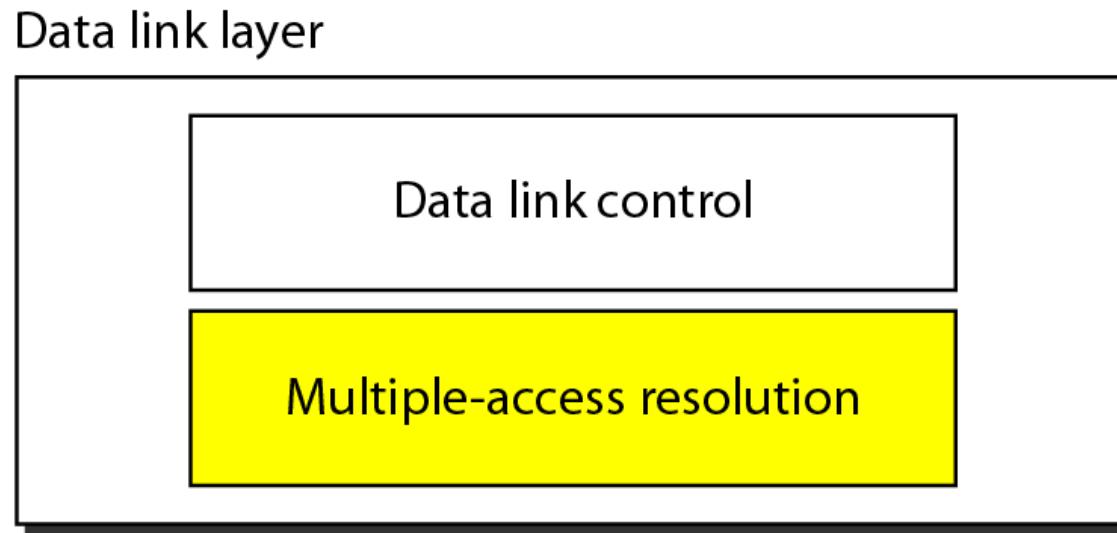
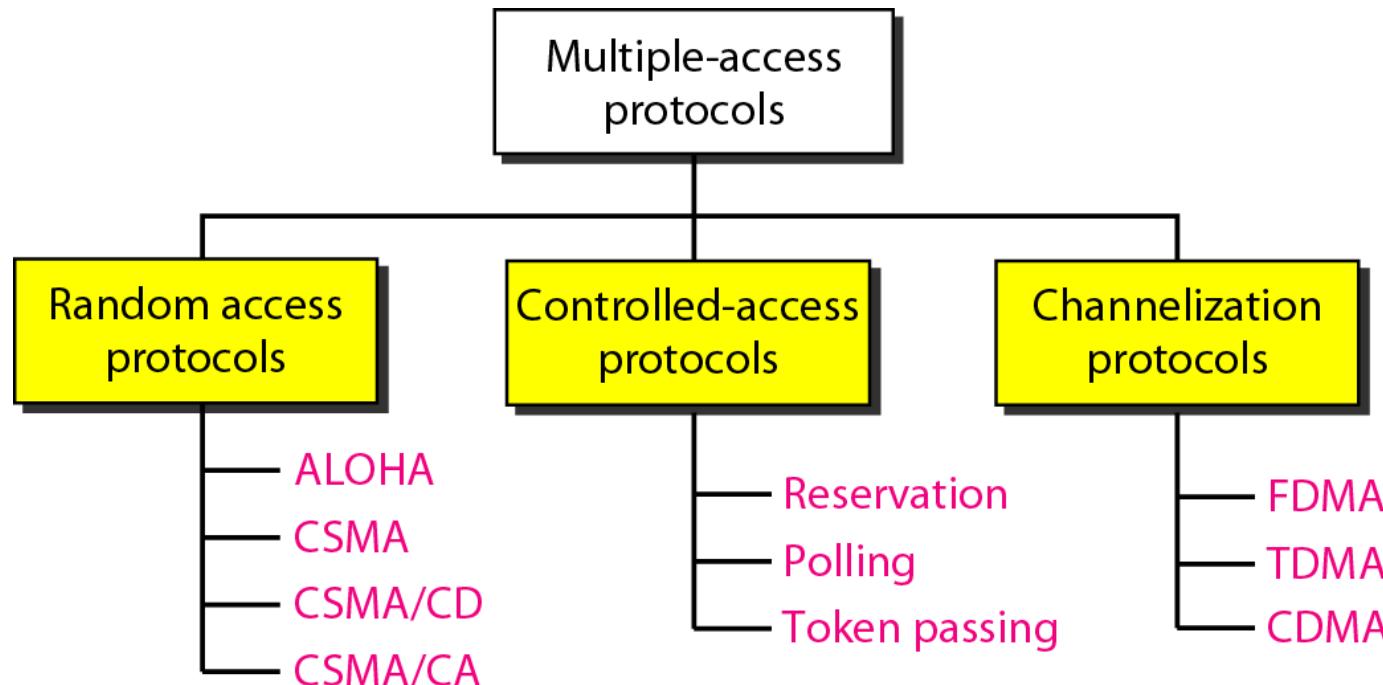


Figure 12.2 *Taxonomy of multiple-access protocols discussed in this chapter*



12-1 RANDOM ACCESS

*In **random access** or **contention** methods, no station is superior to another station and none is assigned the control over another. No station permits, or does not permit, another station to send. At each instance, a station that has data to send uses a procedure defined by the protocol to make a decision on whether or not to send.*

Topics discussed in this section:

ALOHA

Carrier Sense Multiple Access

Carrier Sense Multiple Access with Collision Detection

Carrier Sense Multiple Access with Collision Avoidance

Figure 12.3 *Frames in a pure ALOHA network*

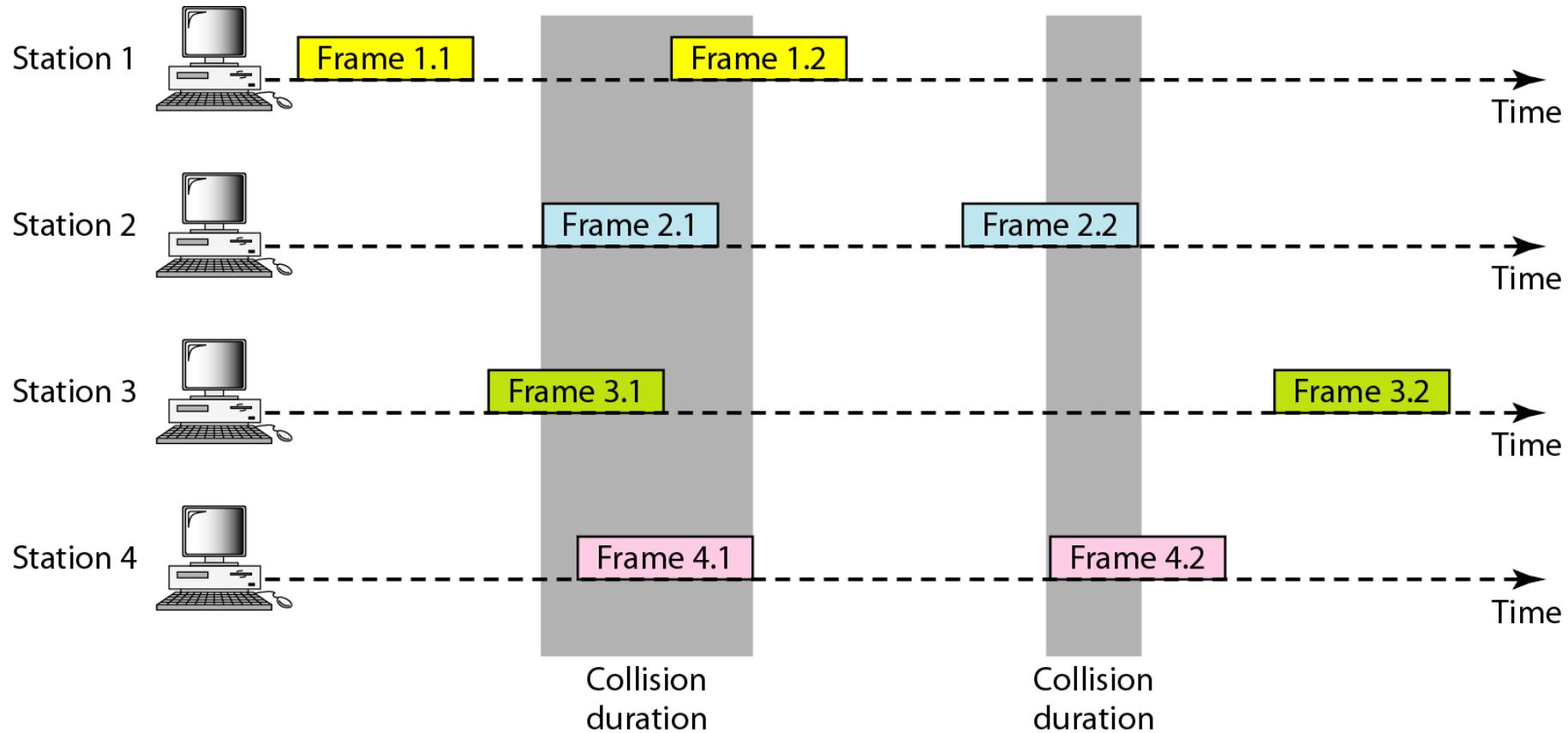
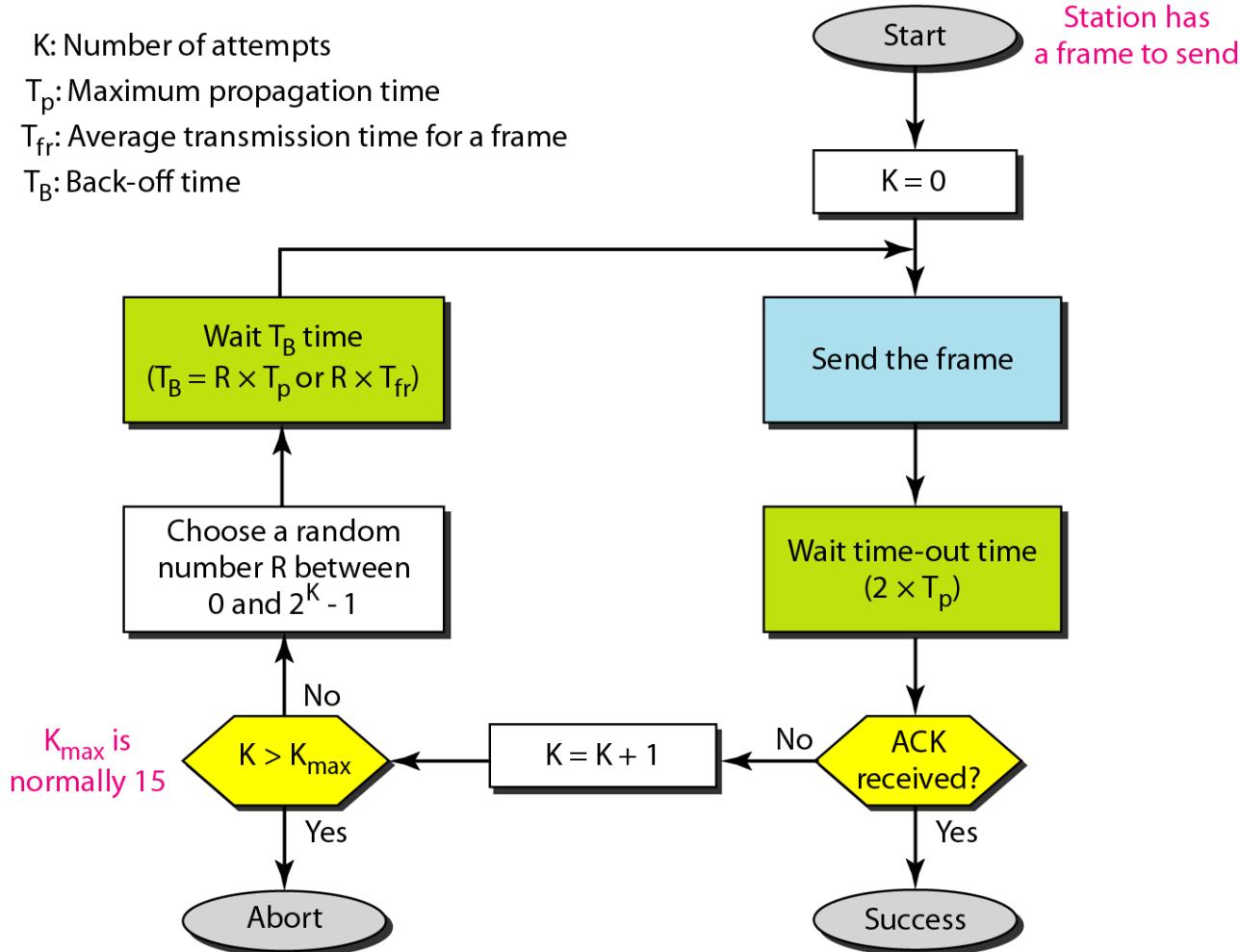
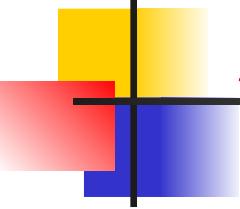


Figure 12.4 Procedure for pure ALOHA protocol





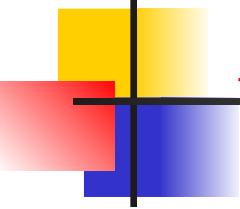
Example 12.1

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

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

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

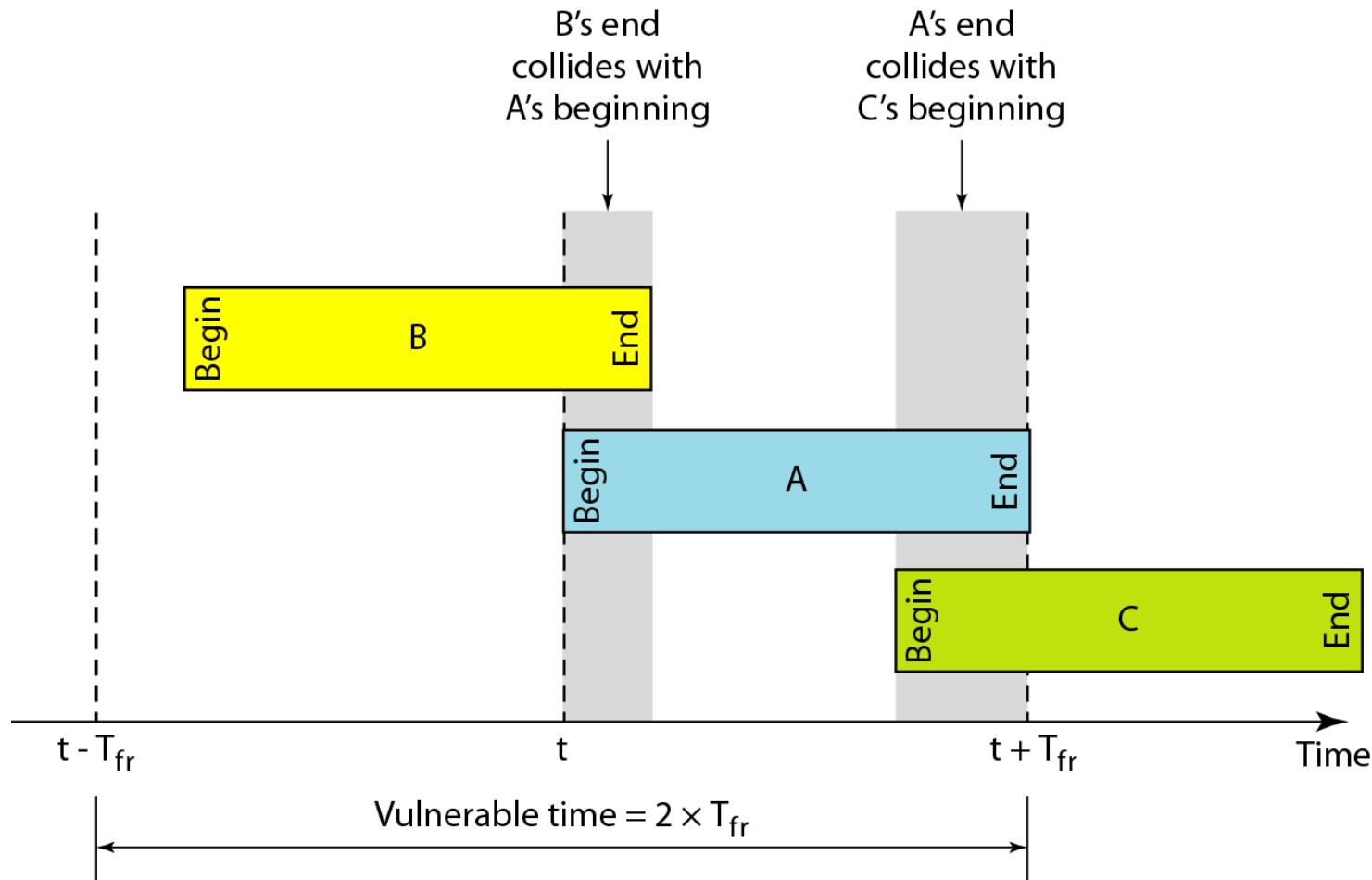
- a. For $K = 1$, the range is $\{0, 1\}$. The station needs to generate a random number with a value of 0 or 1. This means that T_B is either 0 ms (0×2) or 2 ms (1×2), based on the outcome of the random variable.

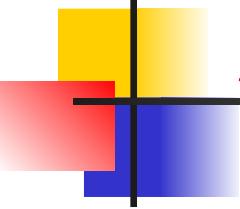


Example 12.1 (continued)

- b.** For $K = 2$, the range is $\{0, 1, 2, 3\}$. This means that T_B can be 0, 2, 4, or 6 ms, based on the outcome of the random variable.
- c.** For $K = 3$, the range is $\{0, 1, 2, 3, 4, 5, 6, 7\}$. This means that T_B can be 0, 2, 4, . . . , 14 ms, based on the outcome of the random variable.
- d.** We need to mention that if $K > 10$, it is normally set to 10.

Figure 12.5 Vulnerable time for pure ALOHA protocol



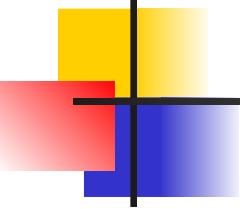


Example 12.2

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

Solution

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



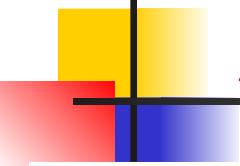
Note

The throughput for pure ALOHA is

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

The maximum throughput

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



Example 12.3

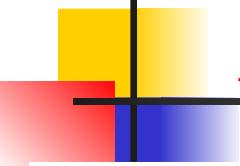
A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the throughput if the system (all stations together) produces

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

Solution

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

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

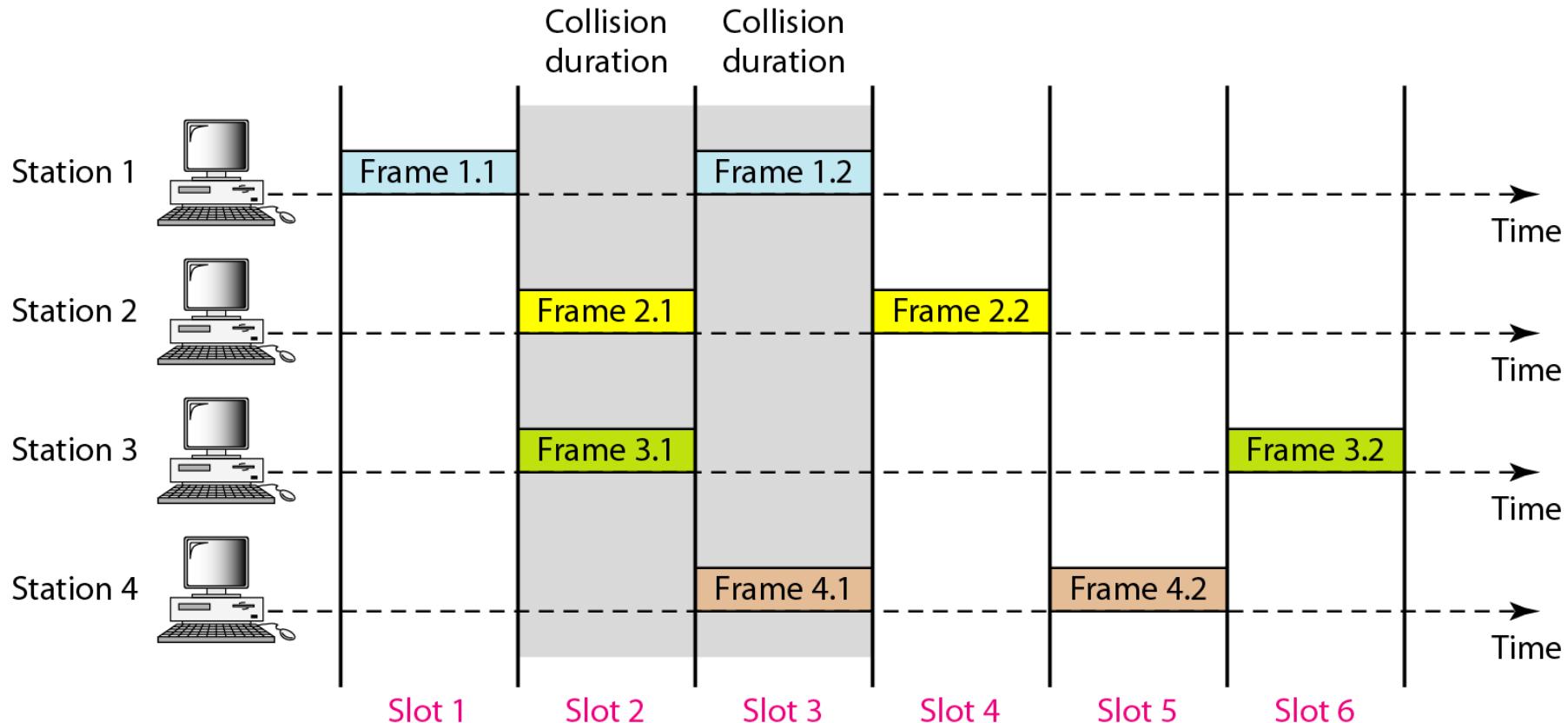


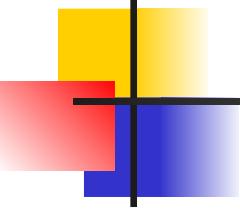
Example 12.3 (continued)

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

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

Figure 12.6 *Frames in a slotted ALOHA network*





Note

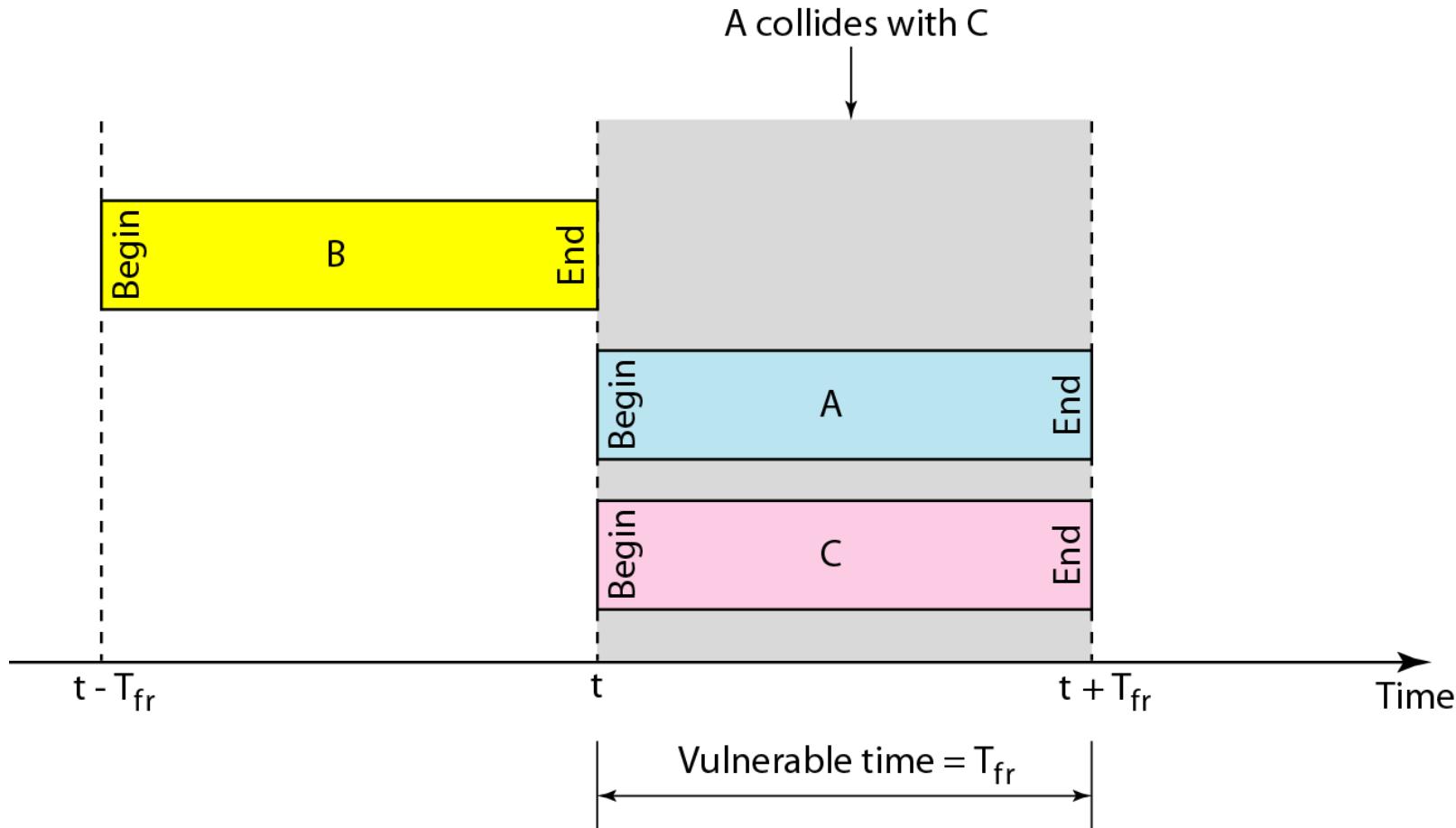
The throughput for slotted ALOHA is

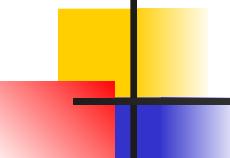
$$S = G \times e^{-G} .$$

The maximum throughput

$$S_{\max} = 0.368 \text{ when } G = 1.$$

Figure 12.7 Vulnerable time for slotted ALOHA protocol





Example 12.4

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

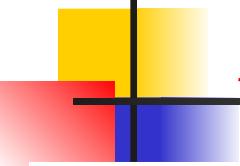
- a. 1000 frames per second
- b. 500 frames per second
- c. 250 frames per second.

Solution

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

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

Only 386 frames out of 1000 will probably survive.



Example 12.4 (continued)

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

Figure 12.8 Space/time model of the collision in CSMA

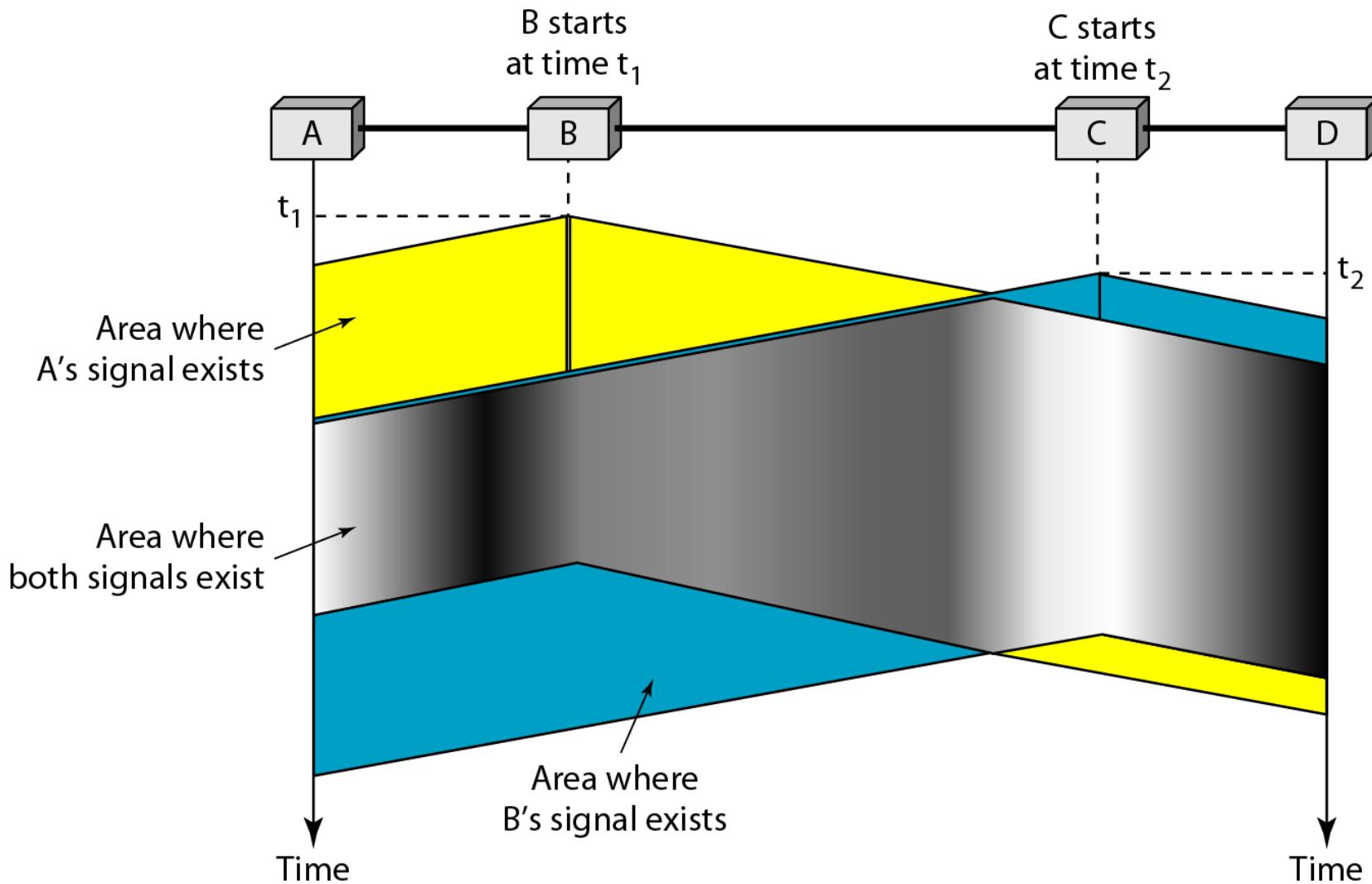


Figure 12.9 Vulnerable time in CSMA

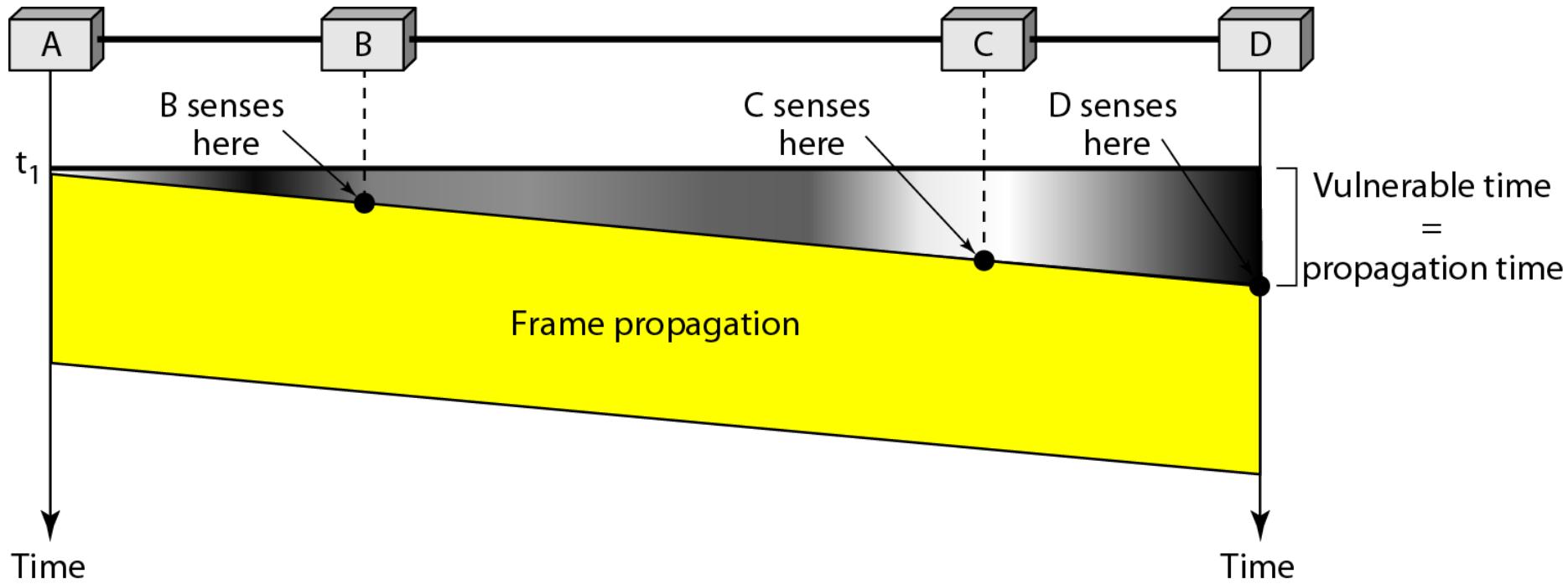
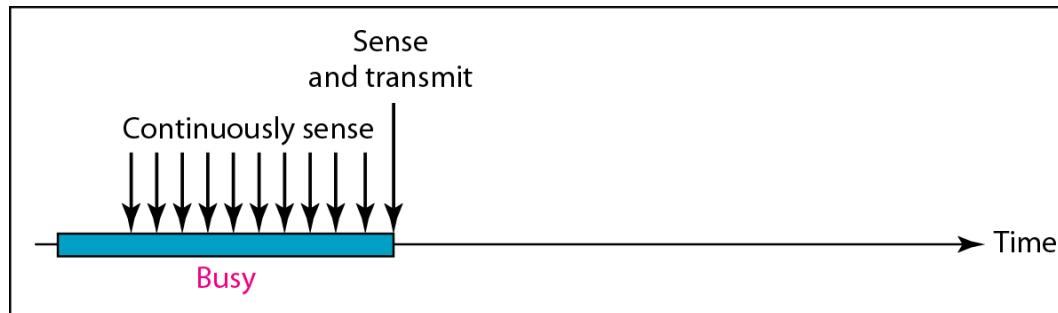
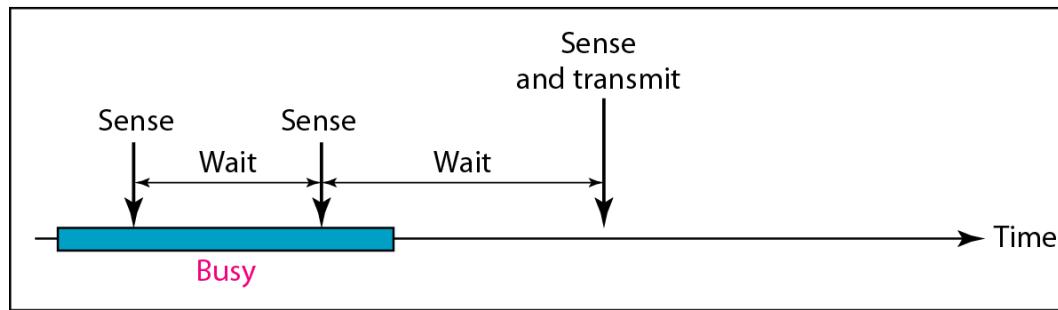


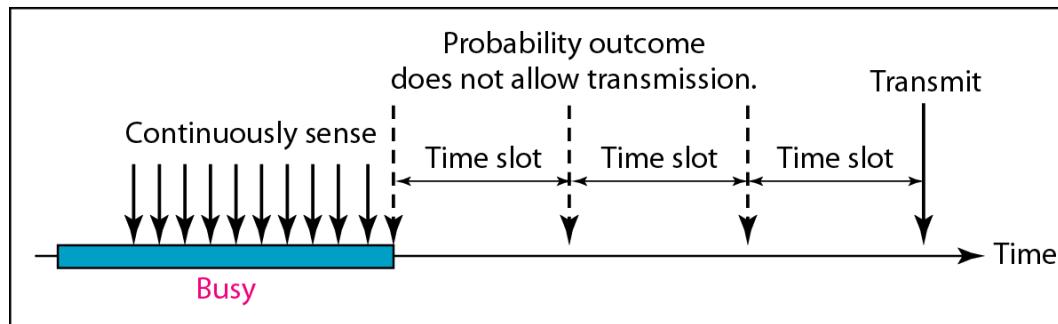
Figure 12.10 Behavior of three persistence methods



a. 1-persistent



b. Nonpersistent



c. p-persistent

Figure 12.11 Flow diagram for three persistence methods

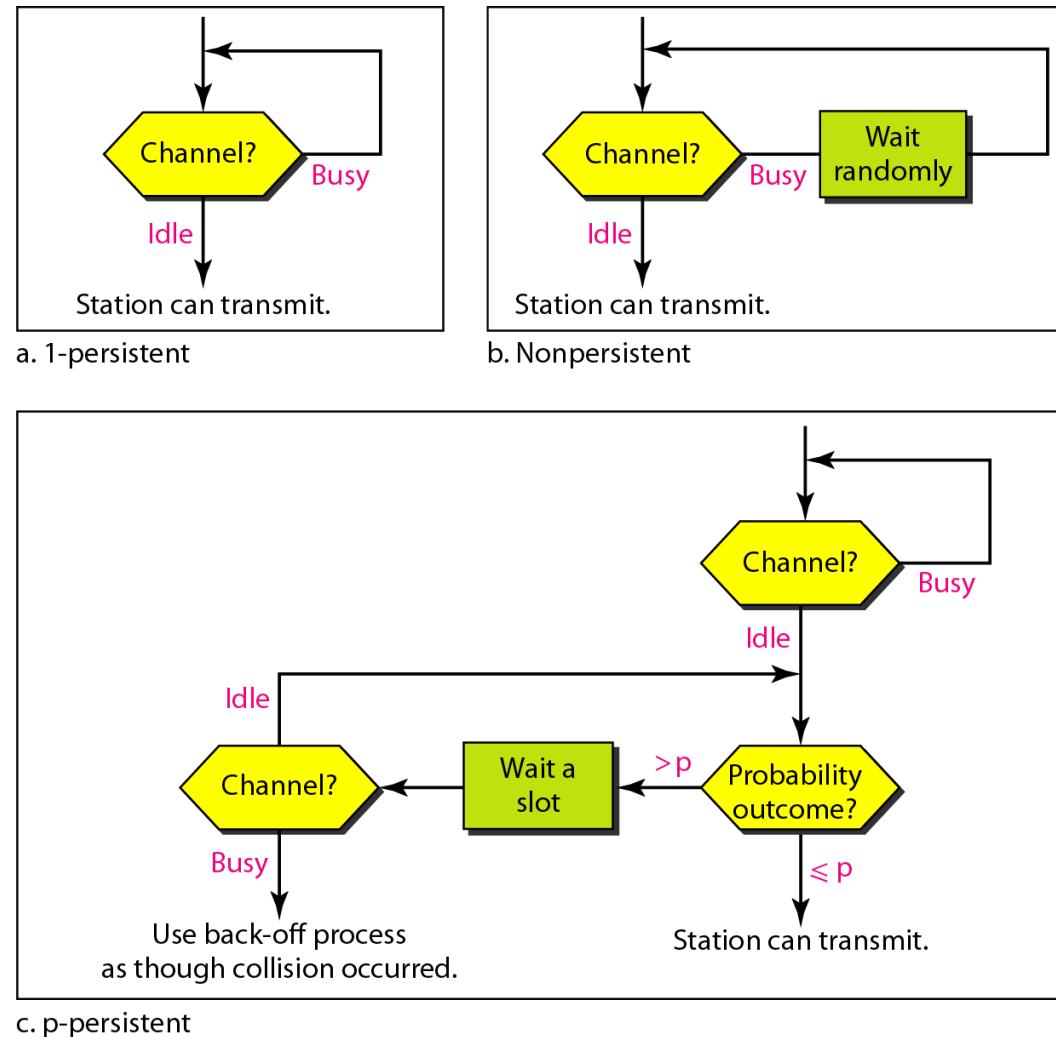


Figure 12.12 Collision of the first bit in CSMA/CD

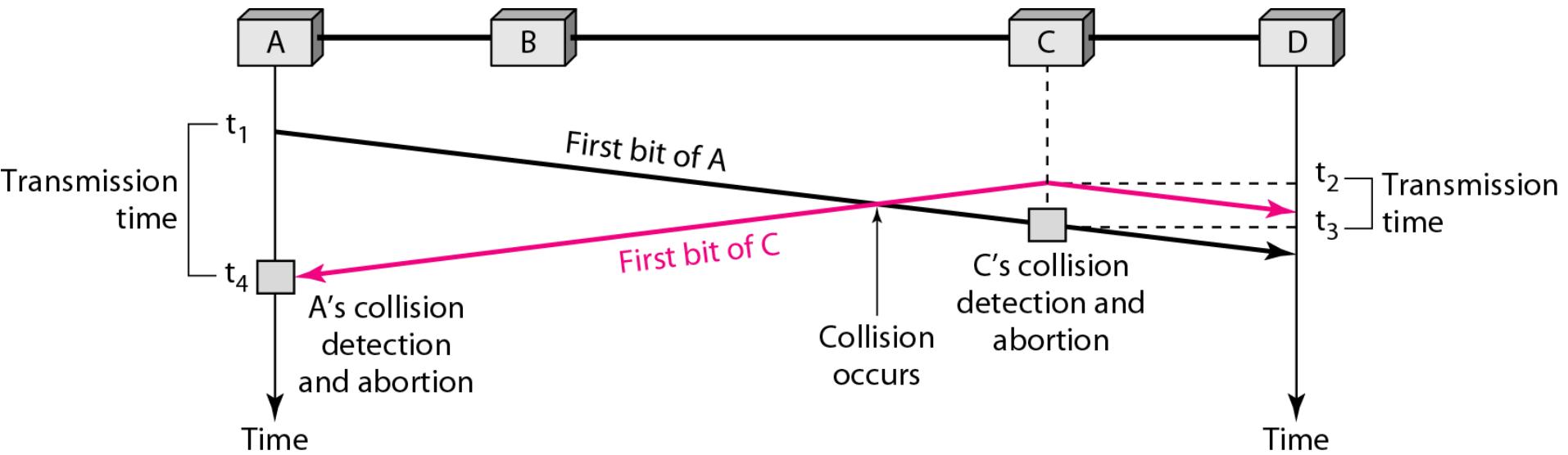
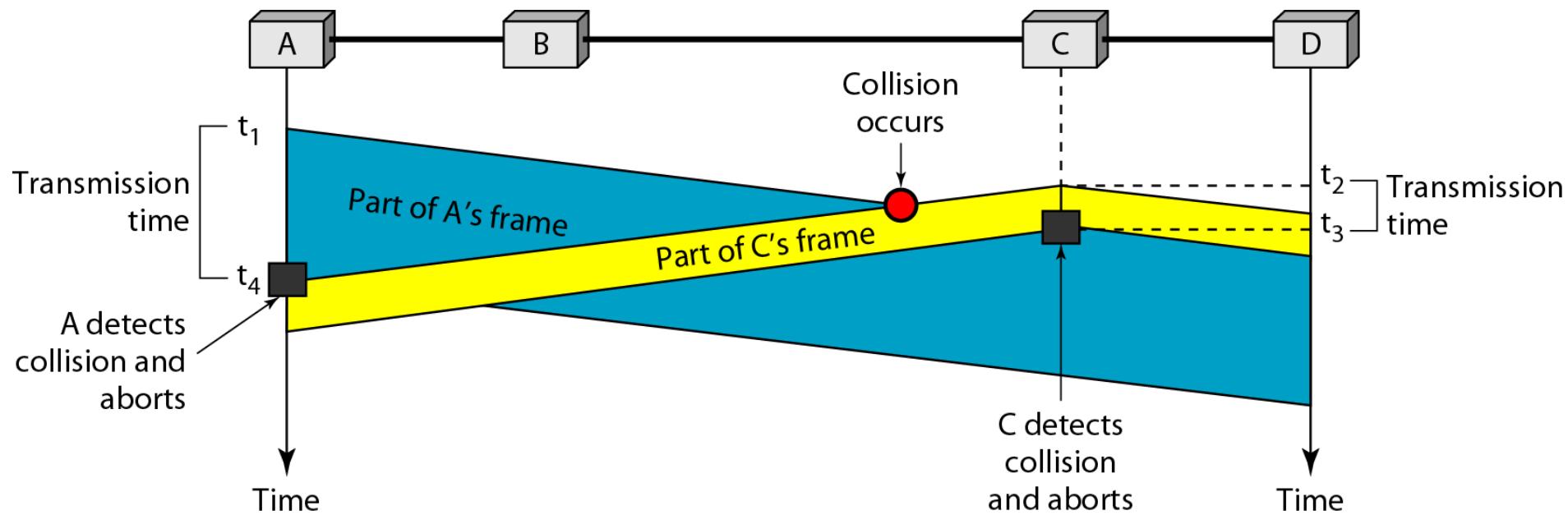
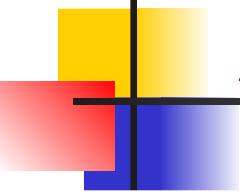


Figure 12.13 Collision and abortion in CSMA/CD





Example 12.5

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

Solution

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

Figure 12.14 Flow diagram for the CSMA/CD

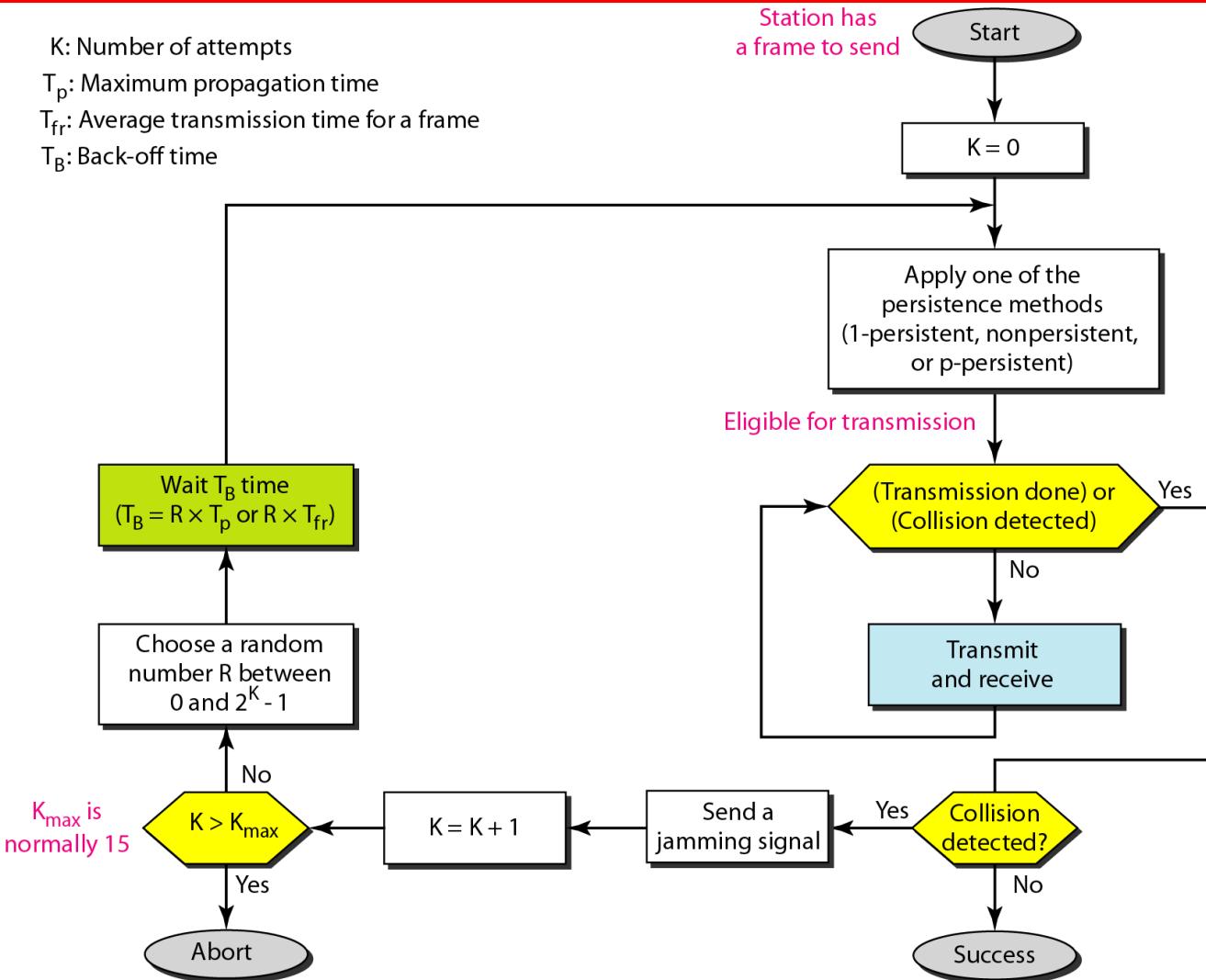


Figure 12.15 Energy level during transmission, idleness, or collision

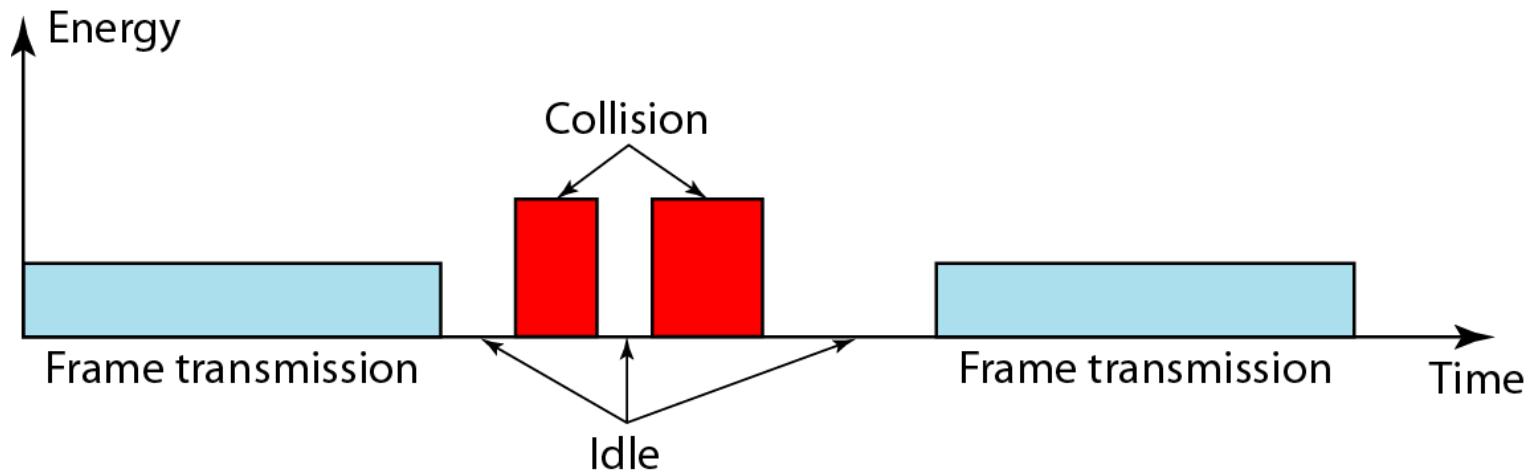
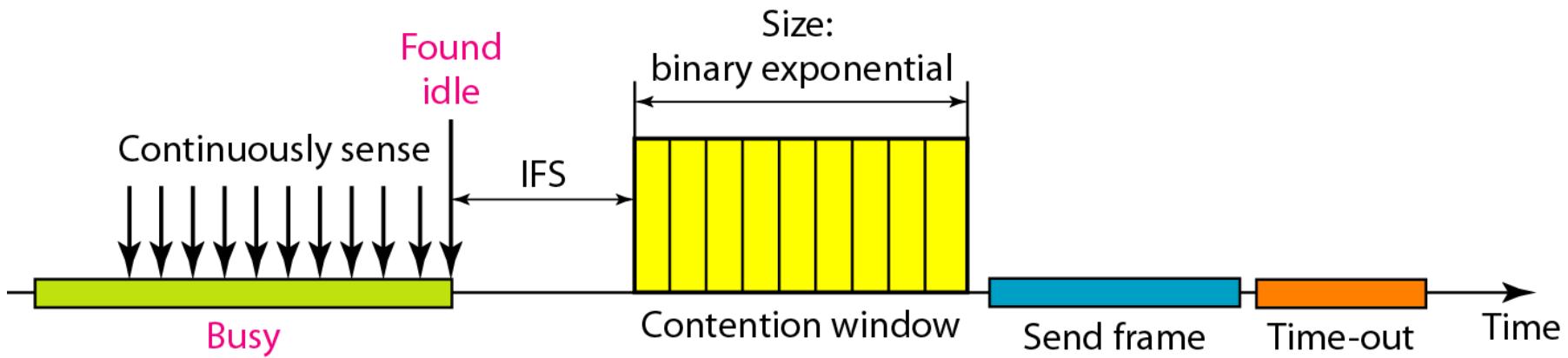
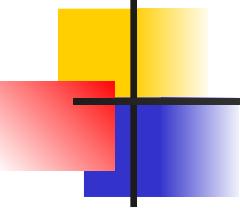


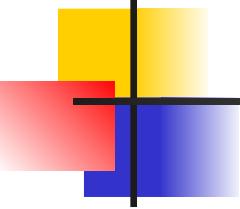
Figure 12.16 Timing in CSMA/CA





Note

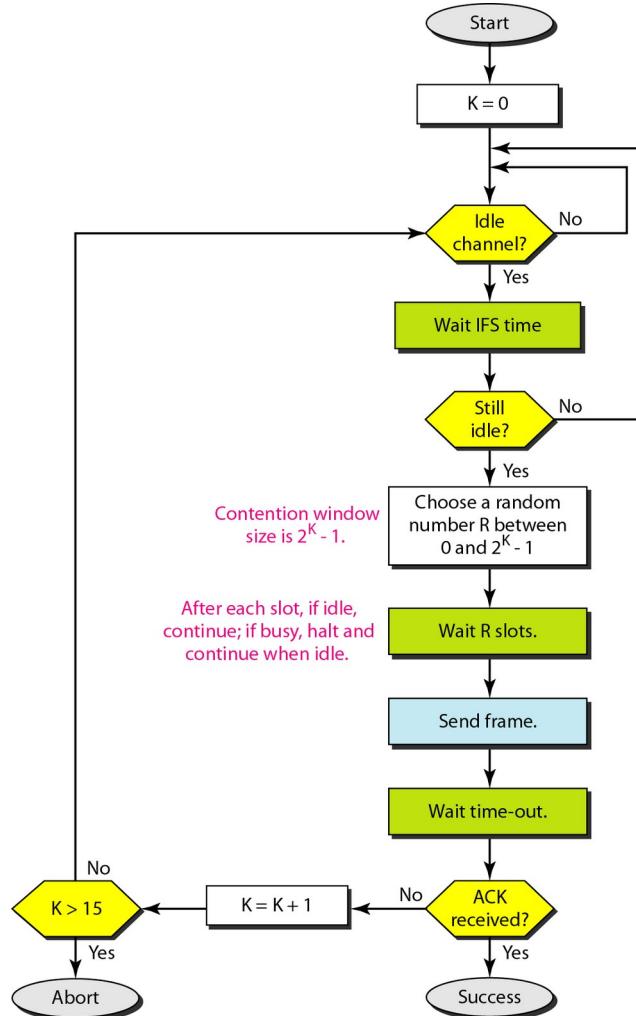
In CSMA/CA, the IFS can also be used to define the priority of a station or a frame.



Note

In CSMA/CA, if the station finds the channel busy, it does not restart the timer of the contention window; it stops the timer and restarts it when the channel becomes idle.

Figure 12.17 Flow diagram for CSMA/CA



12-2 CONTROLLED ACCESS

*In **controlled access**, the stations consult one another to find which station has the right to send. A station cannot send unless it has been authorized by other stations. We discuss three popular controlled-access methods.*

Topics discussed in this section:

Reservation

Polling

Token Passing

Figure 12.18 Reservation access method

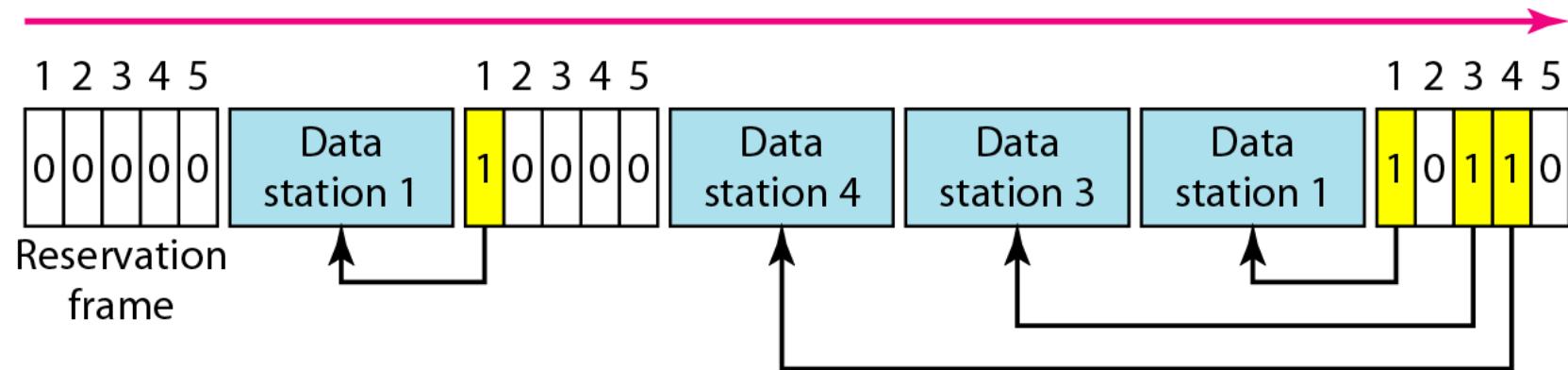


Figure 12.19 Select and poll functions in polling access method

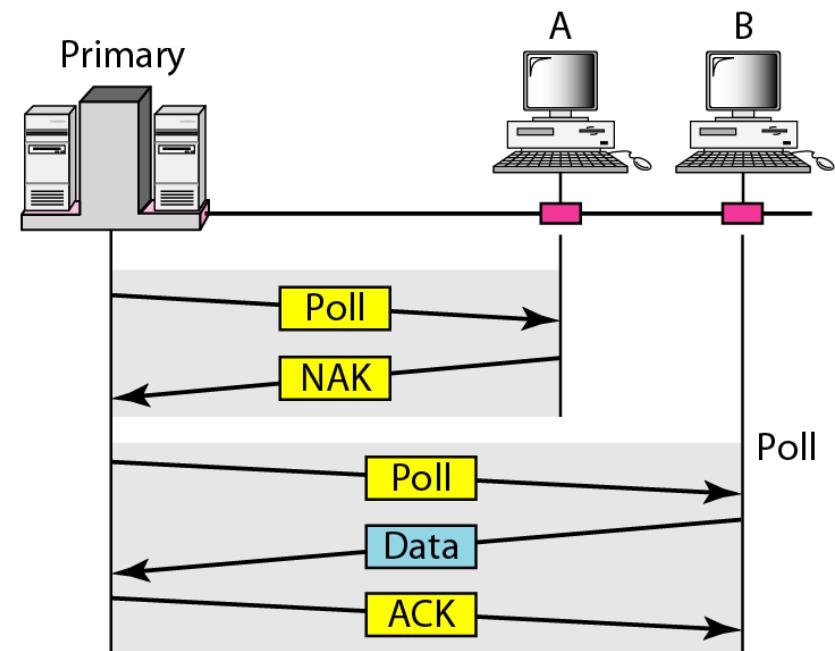
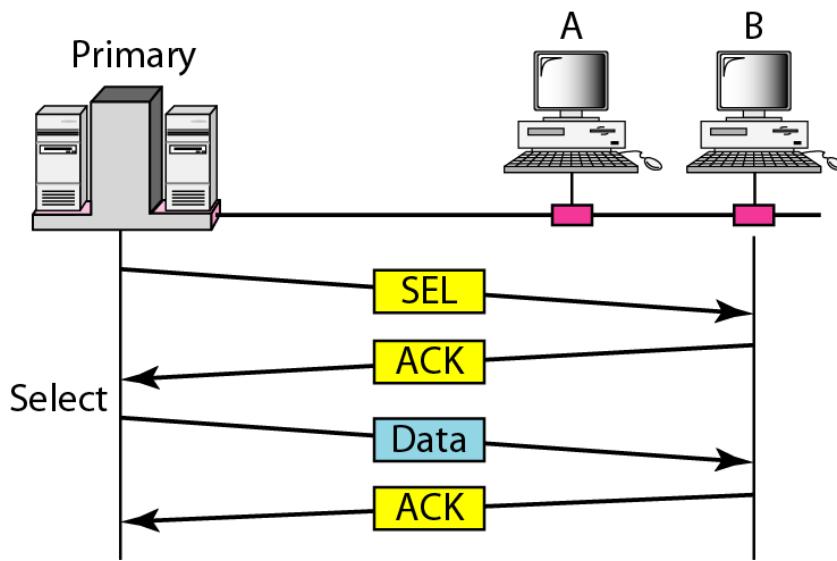
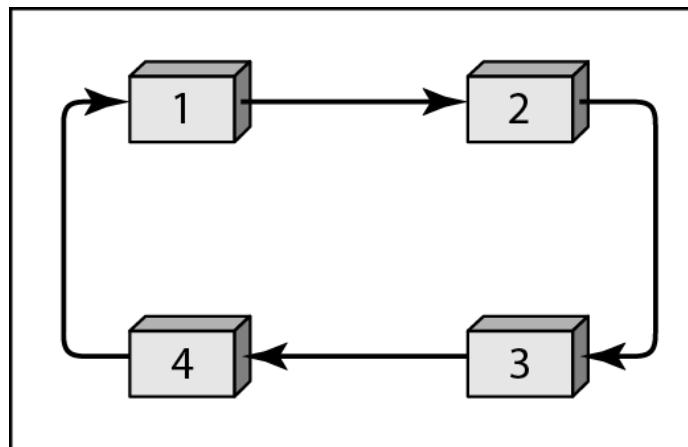
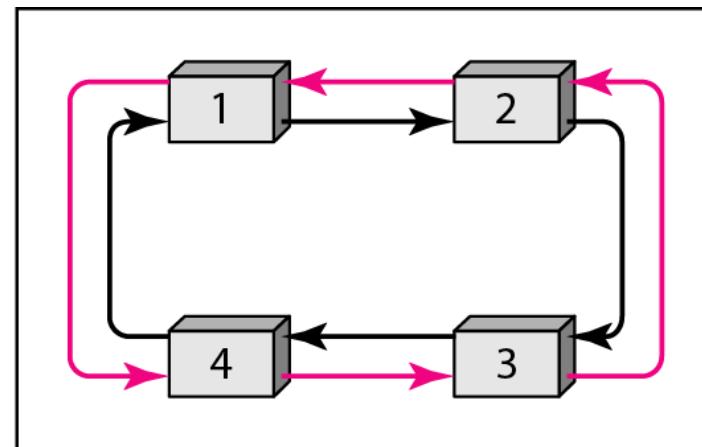


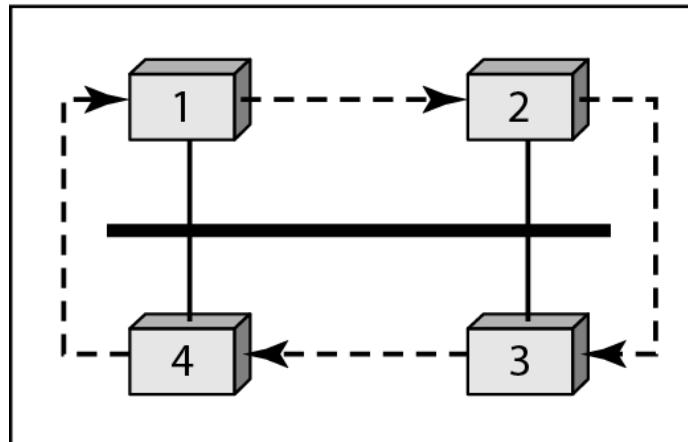
Figure 12.20 *Logical ring and physical topology in token-passing access method*



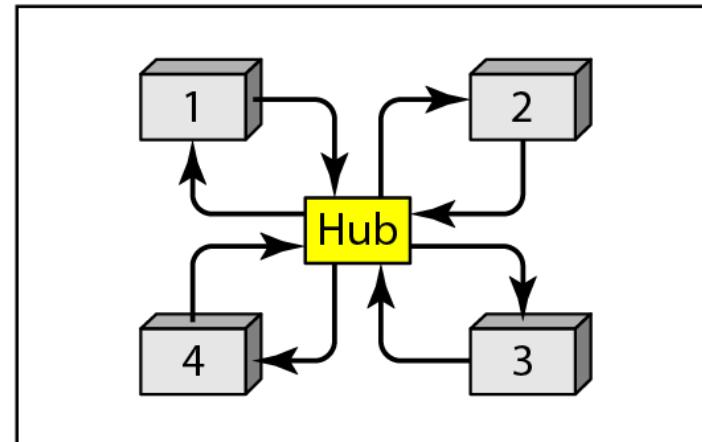
a. Physical ring



b. Dual ring



c. Bus ring



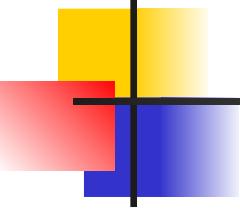
d. Star ring

12-3 CHANNELIZATION

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

Topics discussed in this section:

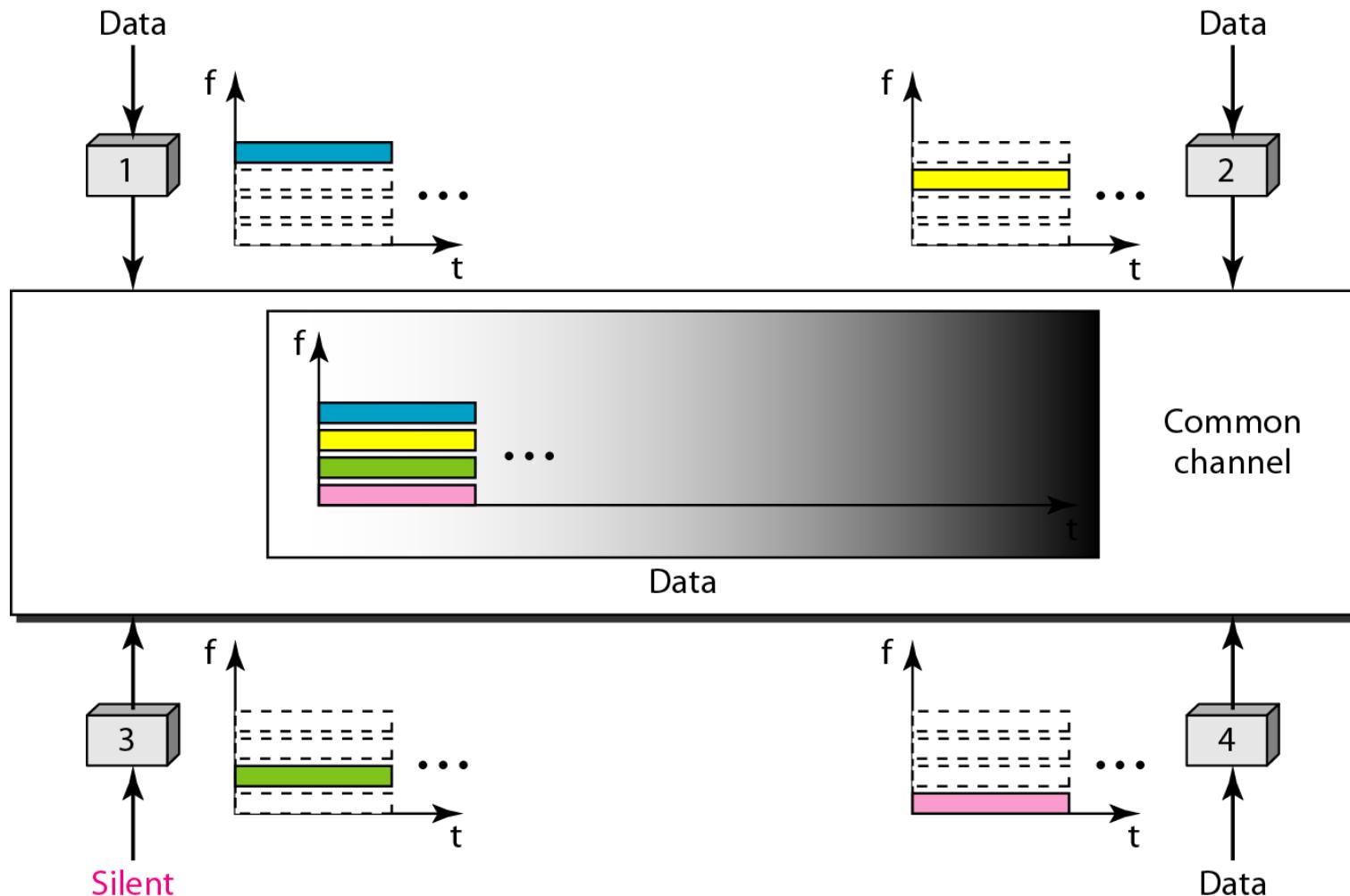
Frequency-Division Multiple Access (FDMA)
Time-Division Multiple Access (TDMA)
Code-Division Multiple Access (CDMA)

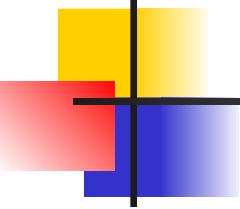


Note

We see the application of all these methods in Chapter 16 when we discuss cellular phone systems.

Figure 12.21 Frequency-division multiple access (FDMA)

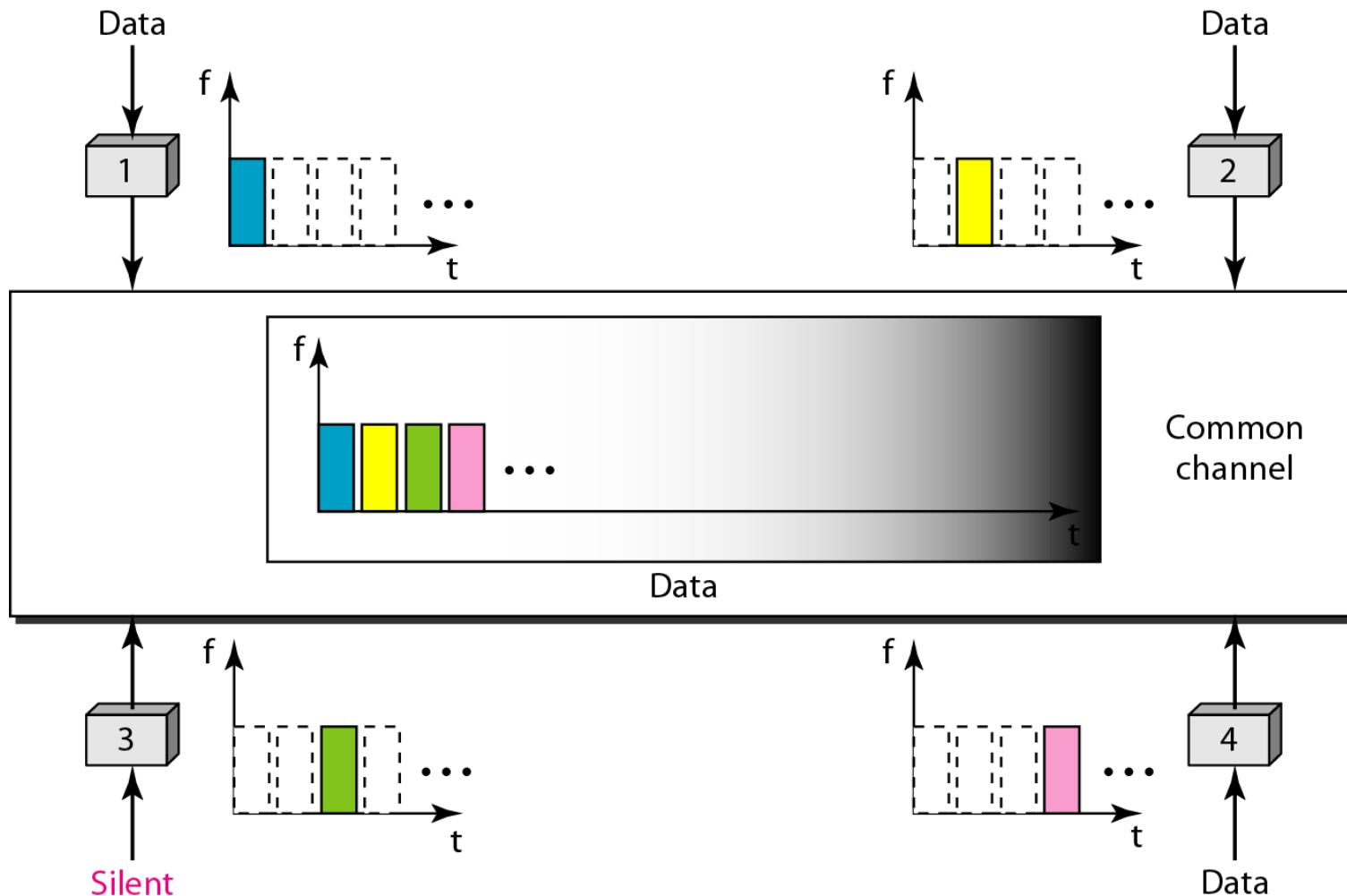


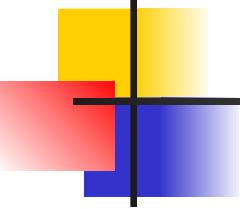


Note

In FDMA, the available bandwidth of the common channel is divided into bands that are separated by guard bands.

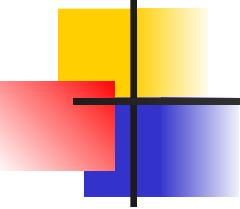
Figure 12.22 Time-division multiple access (TDMA)





Note

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



Note

In CDMA, one channel carries all transmissions simultaneously.

Figure 12.23 Simple idea of communication with code

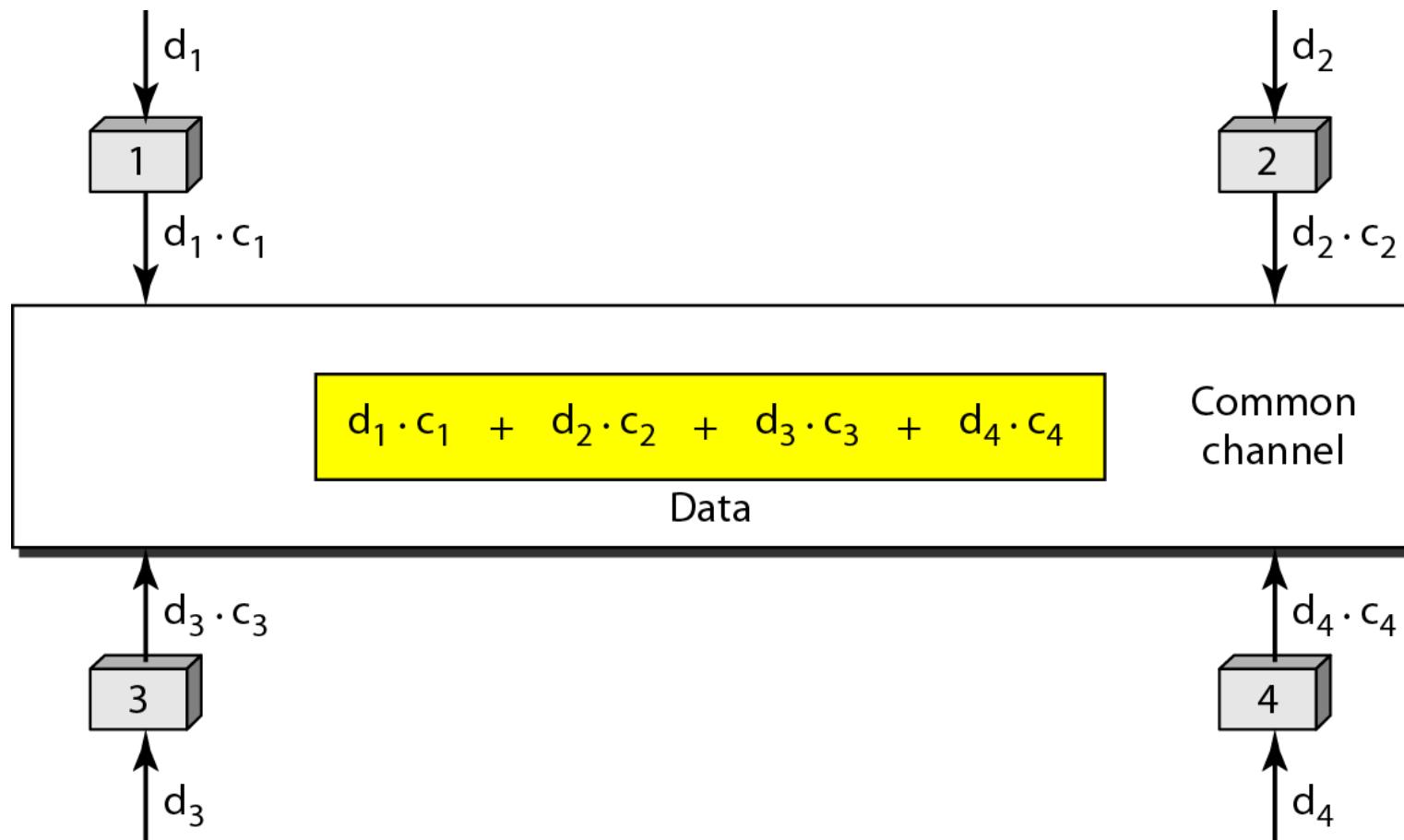


Figure 12.24 *Chip sequences*

C_1

[+1 +1 +1 +1]

C_2

[+1 -1 +1 -1]

C_3

[+1 +1 -1 -1]

C_4

[+1 -1 -1 +1]

Figure 12.25 *Data representation in CDMA*

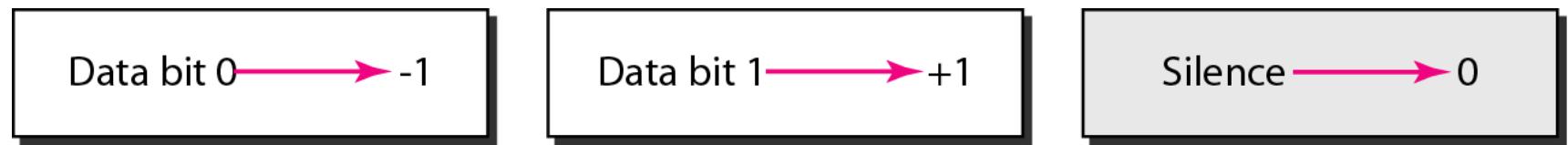


Figure 12.26 Sharing channel in CDMA

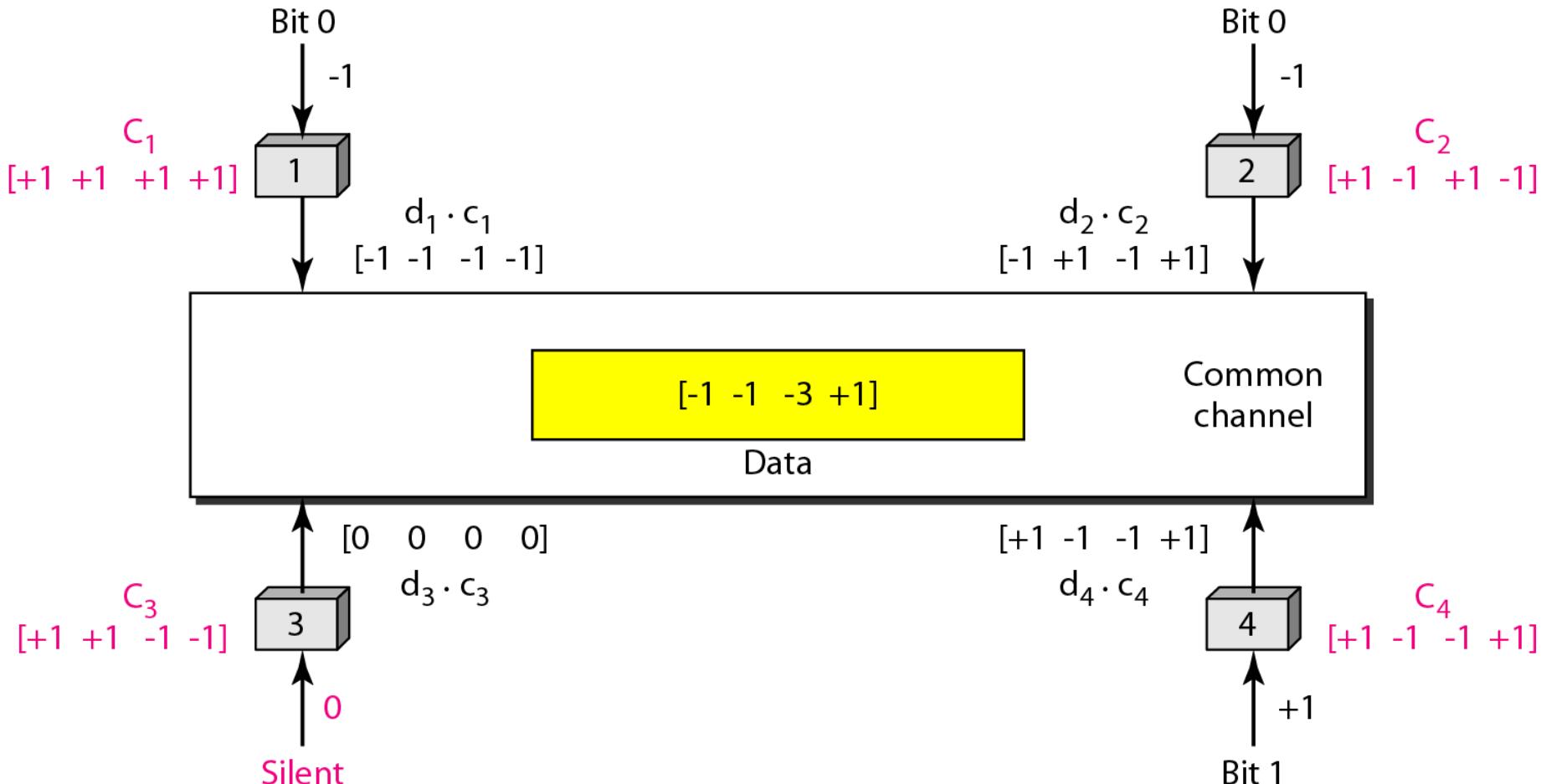


Figure 12.27 Digital signal created by four stations in CDMA

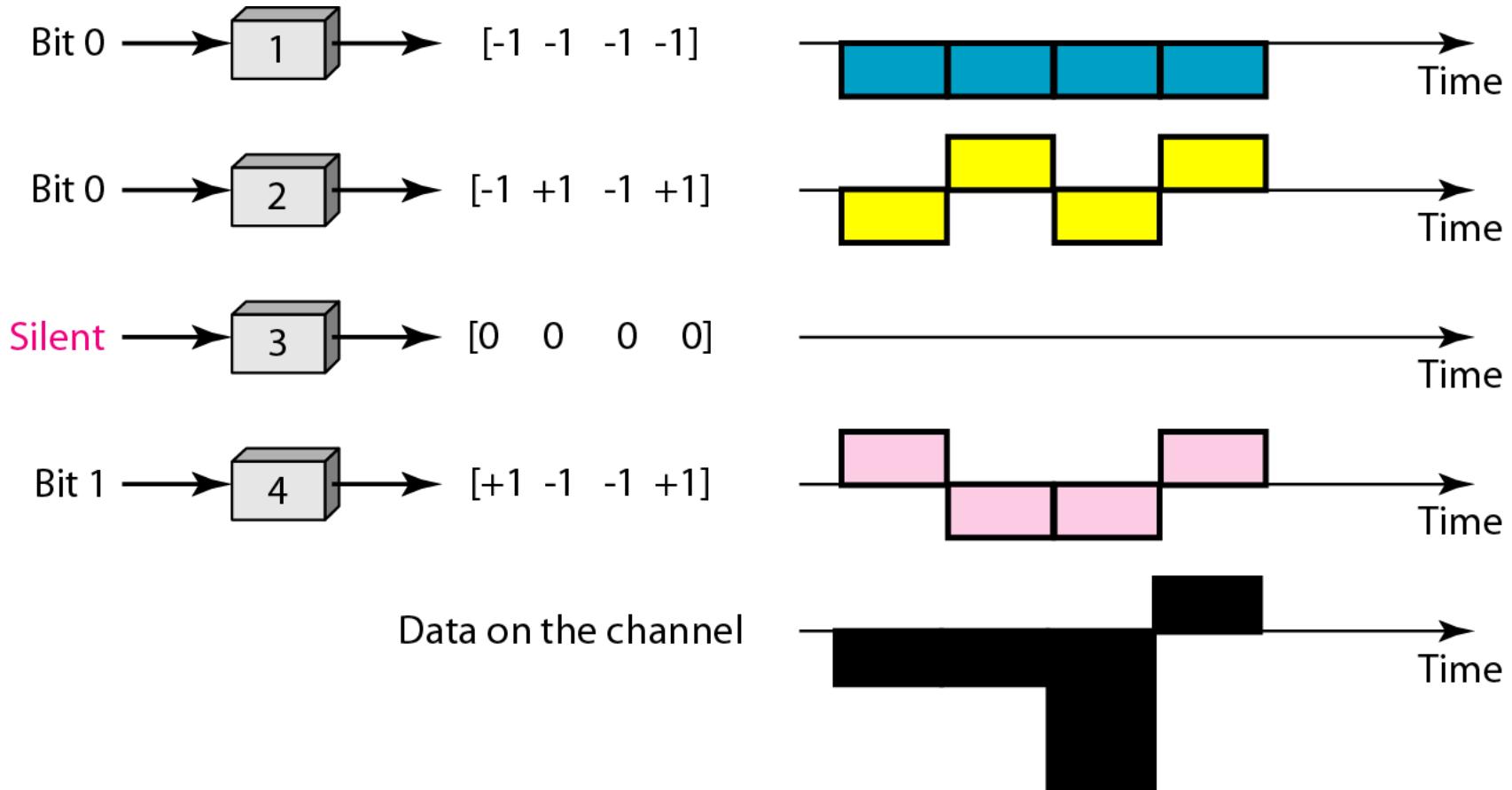


Figure 12.28 Decoding of the composite signal for one in CDMA

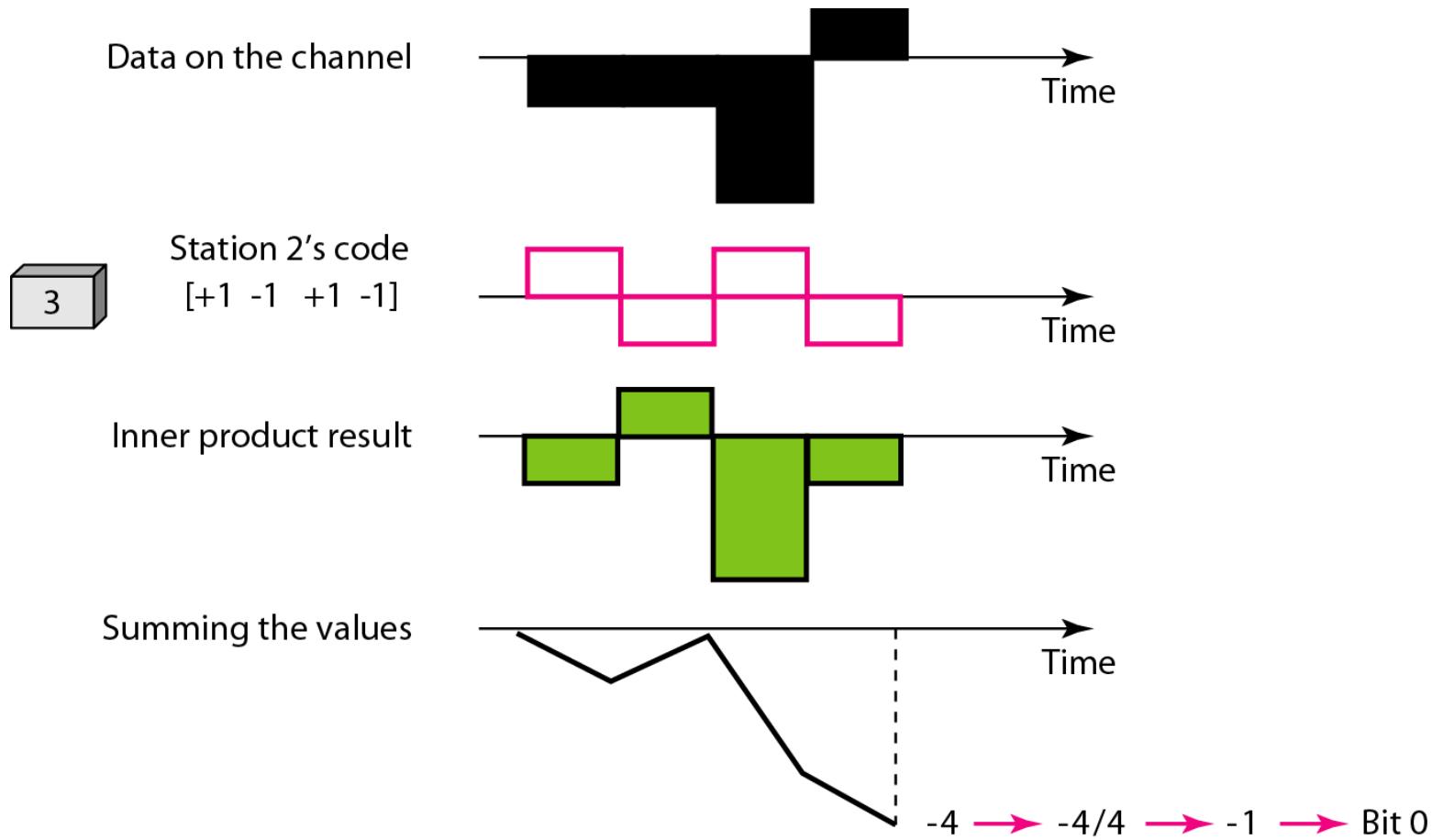


Figure 12.29 General rule and examples of creating Walsh tables

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

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

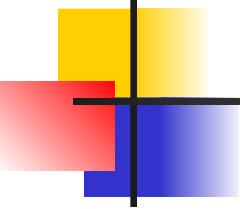
a. Two basic rules

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

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

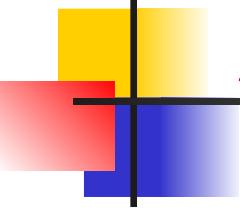
$$W_4 = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}$$

b. Generation of W_1 , W_2 , and W_4



Note

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



Example 12.6

Find the chips for a network with

- a. Two stations**
- b. Four stations**

Solution

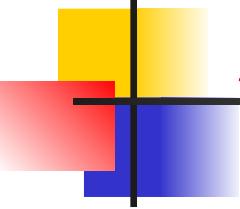
We can use the rows of W_2 and W_4 in Figure 12.29:

- a. For a two-station network, we have**

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

- b. For a four-station network we have**

$$\begin{aligned} &[+1 \ +1 \ +1 \ +1], \quad [+1 \ -1 \ +1 \ -1], \\ &[+1 \ +1 \ -1 \ -1], \text{ and } \quad [+1 \ -1 \ -1 \ +1]. \end{aligned}$$

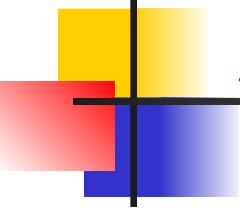


Example 12.7

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

Solution

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



Example 12.8

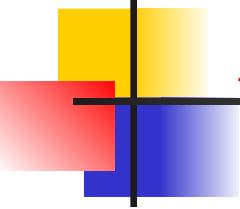
Prove that a receiving station can get the data sent by a specific sender if it multiplies the entire data on the channel by the sender's chip code and then divides it by the number of stations.

Solution

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

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

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



Example 12.8 (continued)

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

When we divide the result by N , we get d_1 .

Chapter 13

Wired LANs: Ethernet

13-1 IEEE STANDARDS

In 1985, the Computer Society of the IEEE started a project, called Project 802, to set standards to enable intercommunication among equipment from a variety of manufacturers. Project 802 is a way of specifying functions of the physical layer and the data link layer of major LAN protocols.

Topics discussed in this section:

Data Link Layer
Physical Layer

Figure 13.1 IEEE standard for LANs

LLC: Logical link control

MAC: Media access control

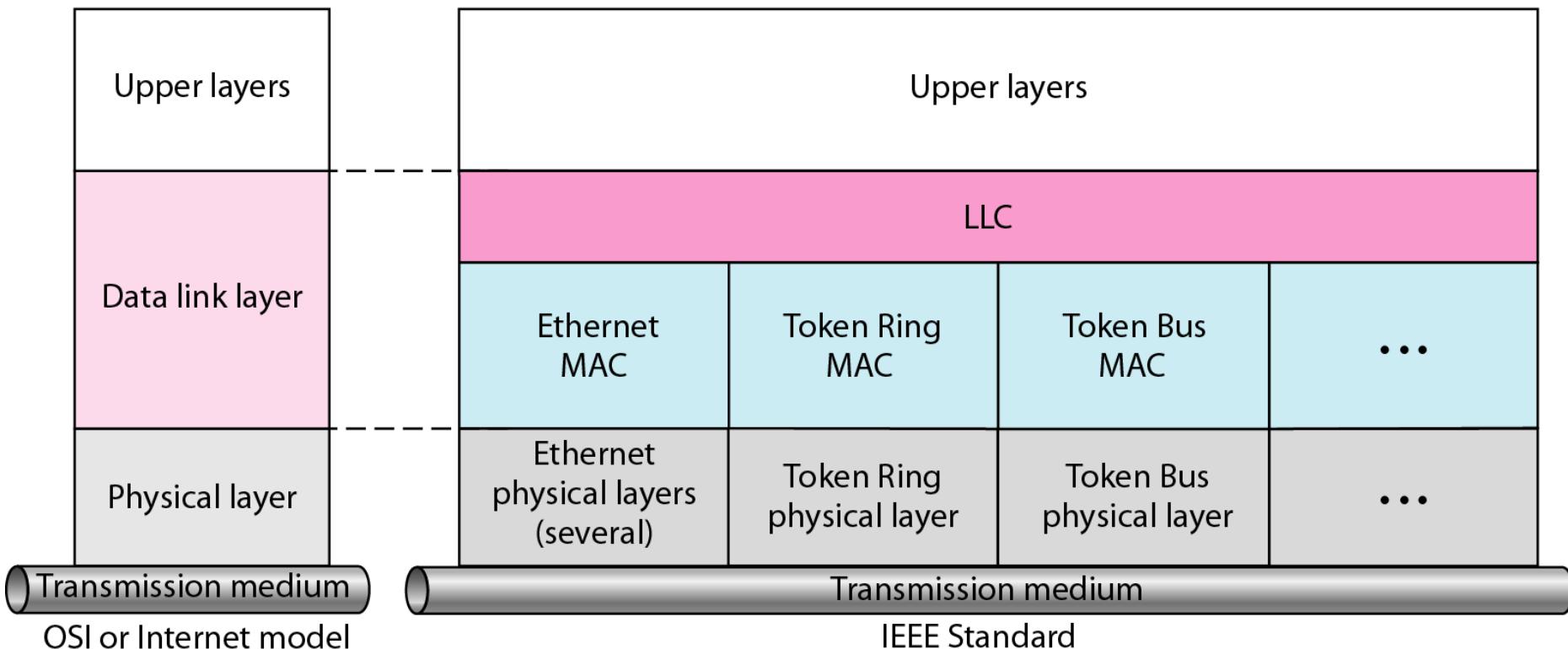
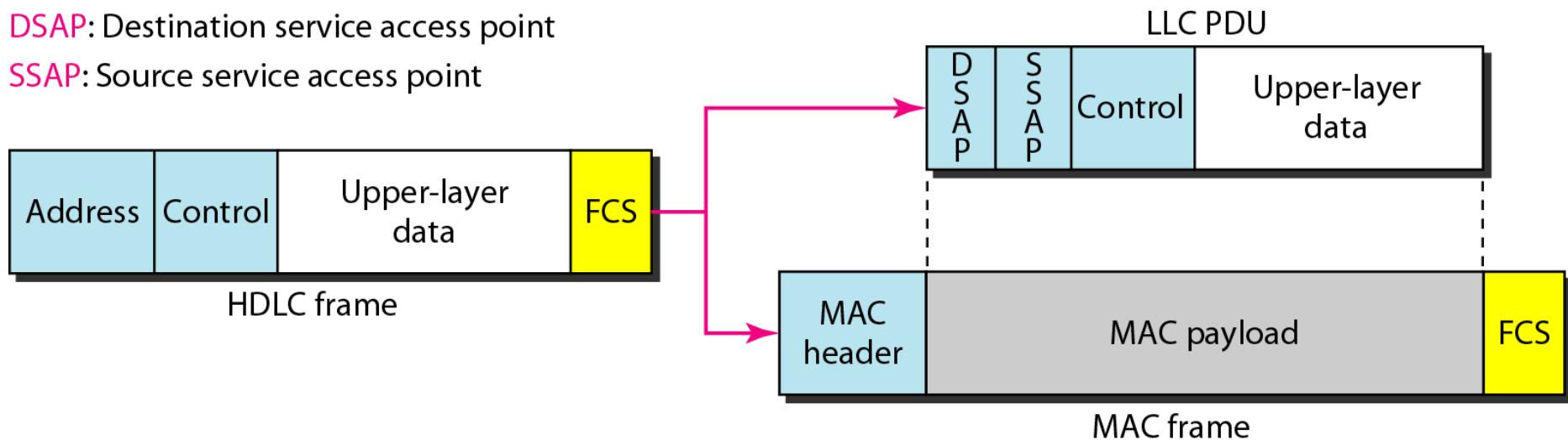


Figure 13.2 *HDLC frame compared with LLC and MAC frames*

DSAP: Destination service access point

SSAP: Source service access point



13-2 STANDARD ETHERNET

The original Ethernet was created in 1976 at Xerox's Palo Alto Research Center (PARC). Since then, it has gone through four generations. We briefly discuss the Standard (or traditional) Ethernet in this section.

Topics discussed in this section:

MAC Sublayer
Physical Layer

Figure 13.3 *Ethernet evolution through four generations*

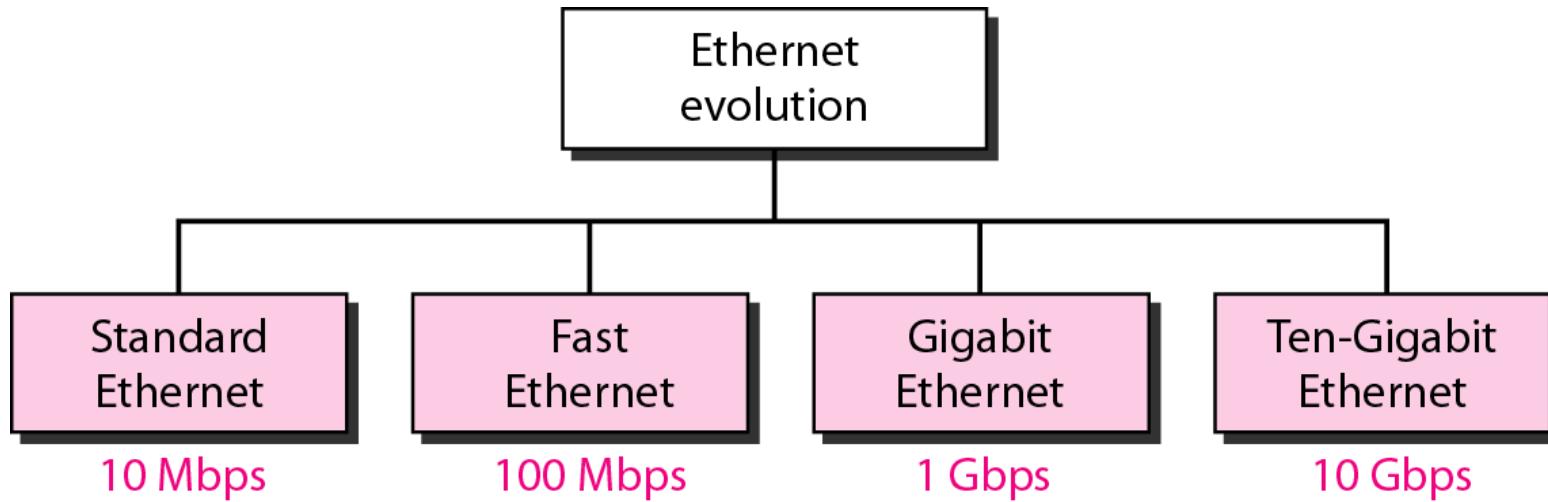


Figure 13.4 802.3 MAC frame

Preamble: 56 bits of alternating 1s and 0s.

SFD: Start frame delimiter, flag (10101011)

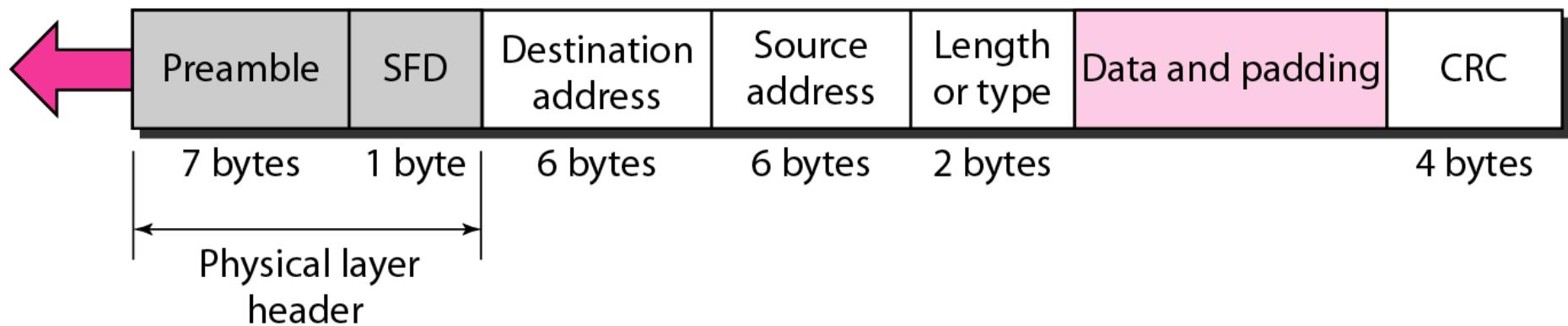
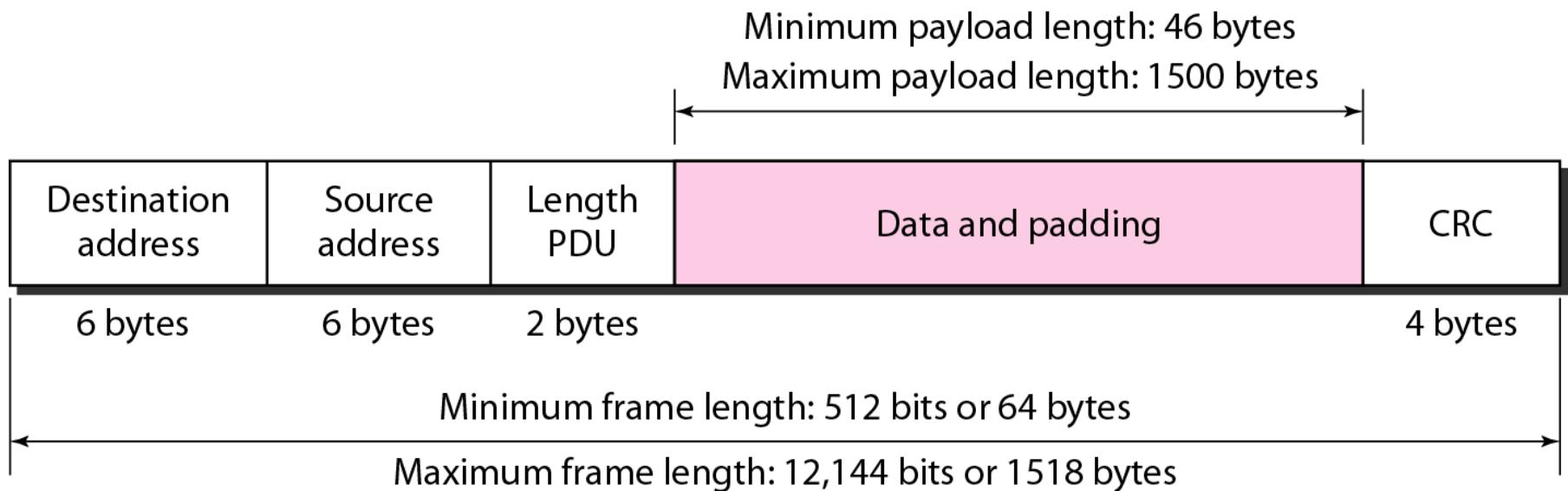
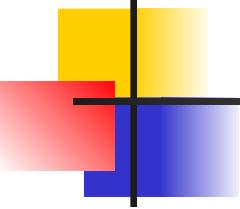


Figure 13.5 *Minimum and maximum lengths*





Note

Frame length:

Minimum: 64 bytes (512 bits)

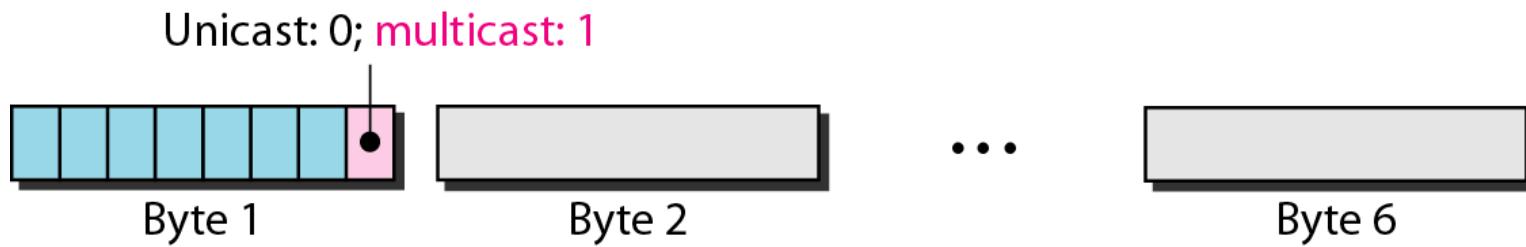
Maximum: 1518 bytes (12,144 bits)

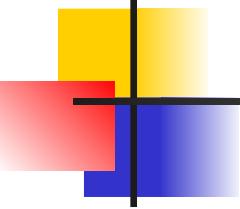
Figure 13.6 *Example of an Ethernet address in hexadecimal notation*

06 : 01 : 02 : 01 : 2C : 4B

6 bytes = 12 hex digits = 48 bits

Figure 13.7 *Unicast and multicast addresses*

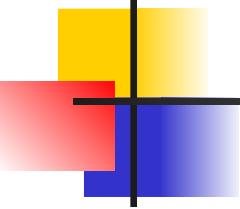




Note

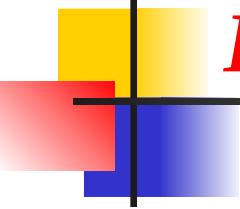
The least significant bit of the first byte defines the type of address.

**If the bit is 0, the address is unicast;
otherwise, it is multicast.**



Note

The broadcast destination address is a special case of the multicast address in which all bits are 1s.



Example 13.1

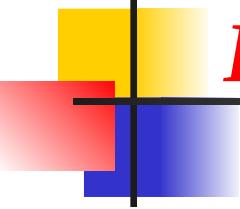
Define the type of the following destination addresses:

- a.** 4A:30:10:21:10:1A
- b.** 47:20:1B:2E:08:EE
- c.** FF:FF:FF:FF:FF:FF

Solution

To find the type of the address, we need to look at the second hexadecimal digit from the left. If it is even, the address is unicast. If it is odd, the address is multicast. If all digits are F's, the address is broadcast. Therefore, we have the following:

- a.** *This is a unicast address because A in binary is 1010.*
- b.** *This is a multicast address because 7 in binary is 0111.*
- c.** *This is a broadcast address because all digits are F's.*



Example 13.2

Show how the address **47:20:1B:2E:08:EE** is sent out on line.

Solution

The address is sent left-to-right, byte by byte; for each byte, it is sent right-to-left, bit by bit, as shown below:



11100010 00000100 11011000 01110100 00010000 01110111

Figure 13.8 *Categories of Standard Ethernet*

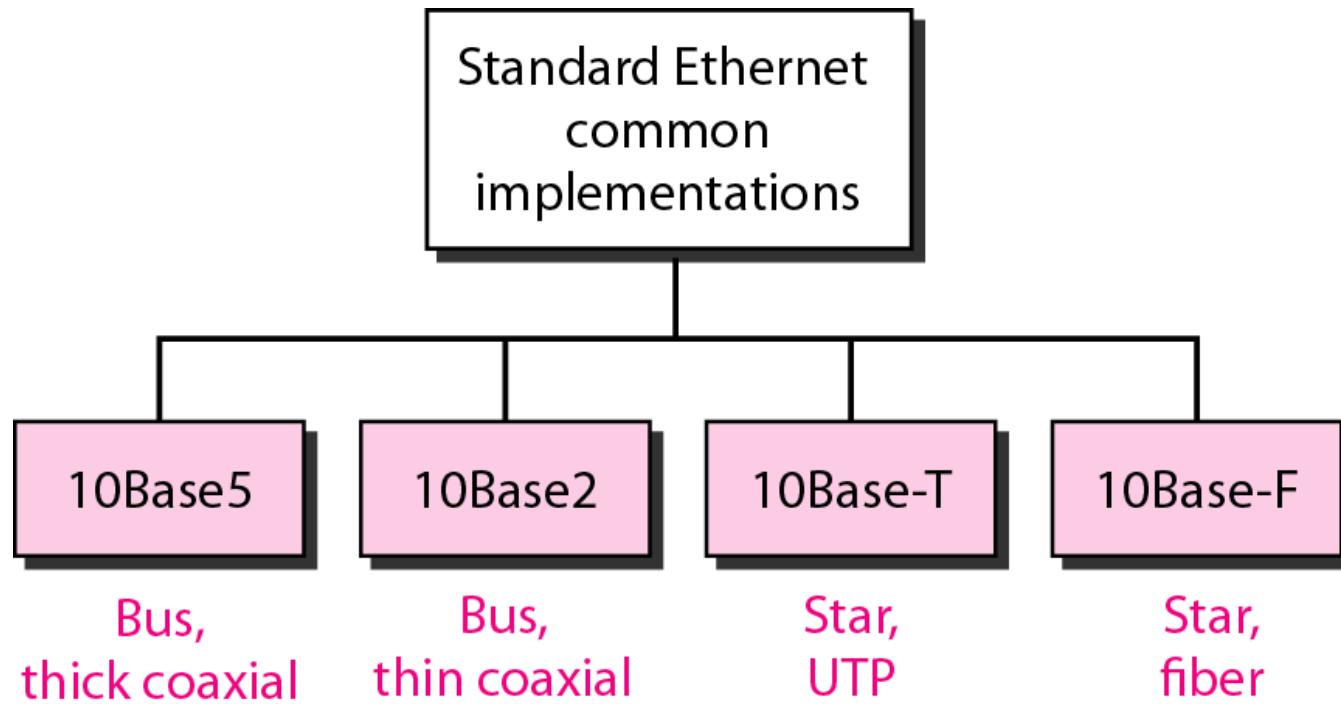


Figure 13.9 *Encoding in a Standard Ethernet implementation*

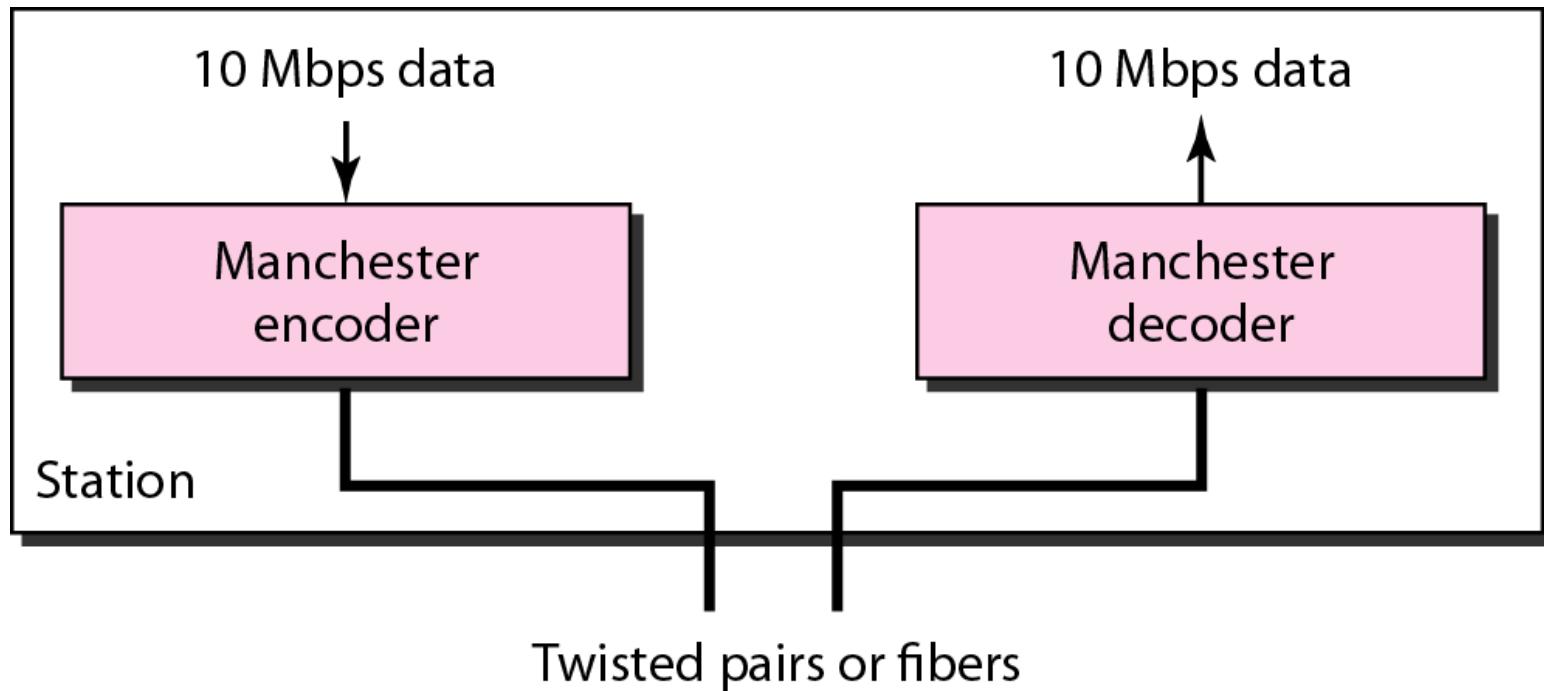


Figure 13.10 10Base5 implementation

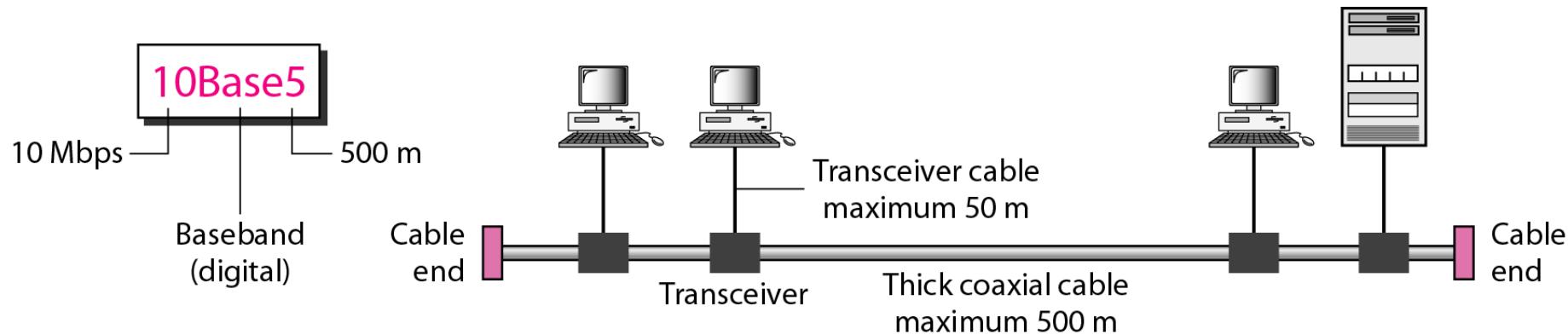


Figure 13.11 *10Base2 implementation*

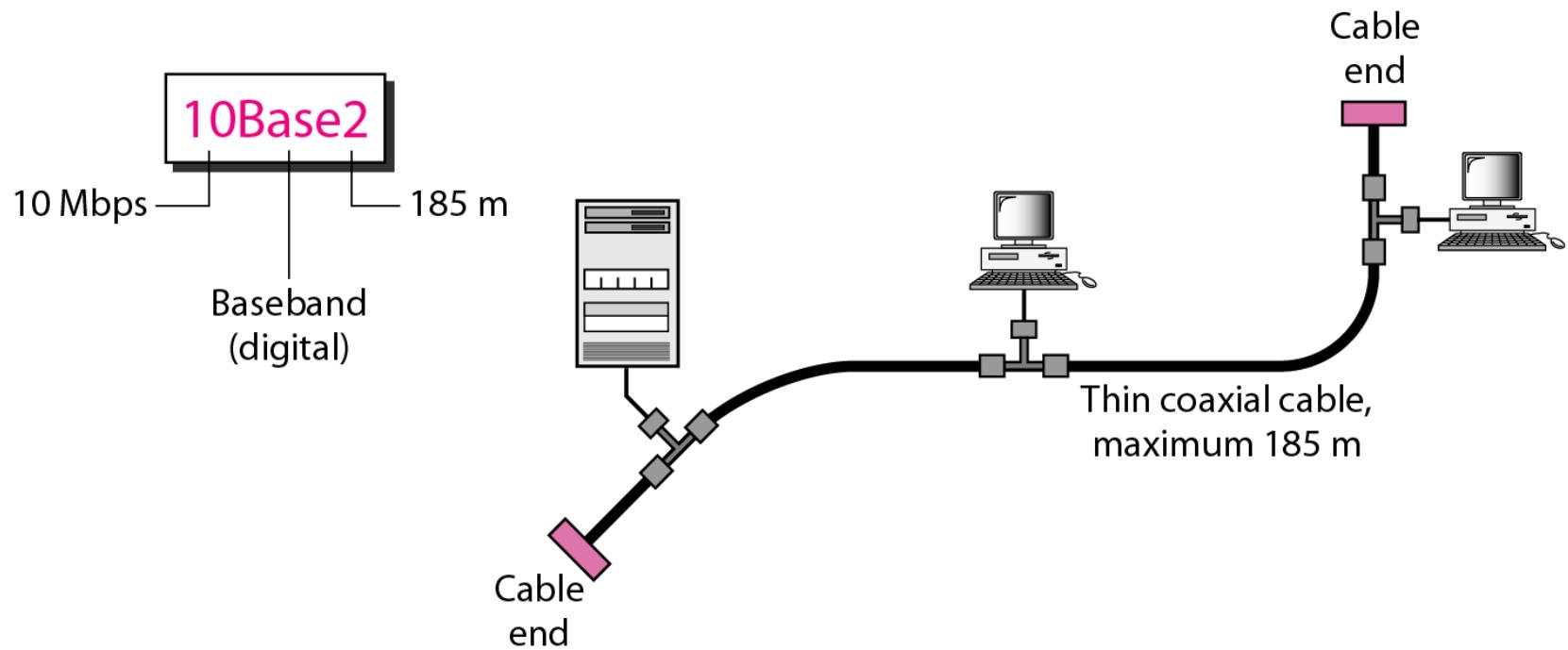


Figure 13.12 10Base-T implementation

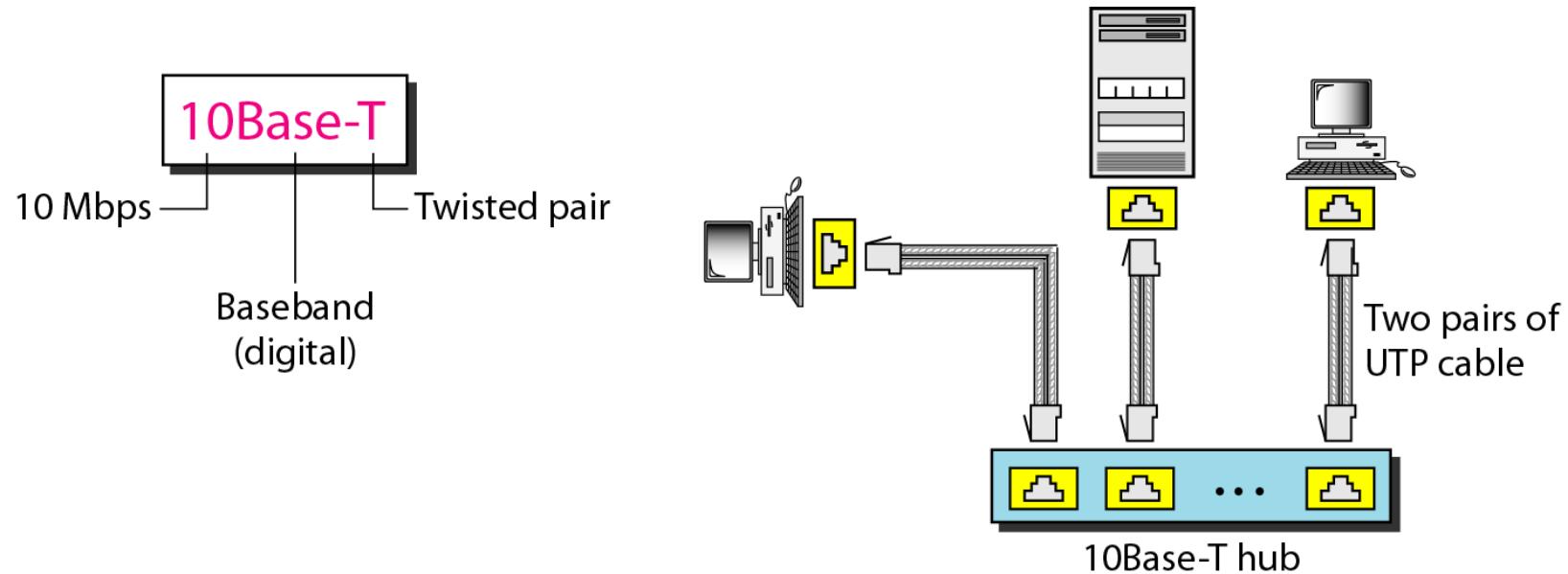


Figure 13.13 10Base-F implementation

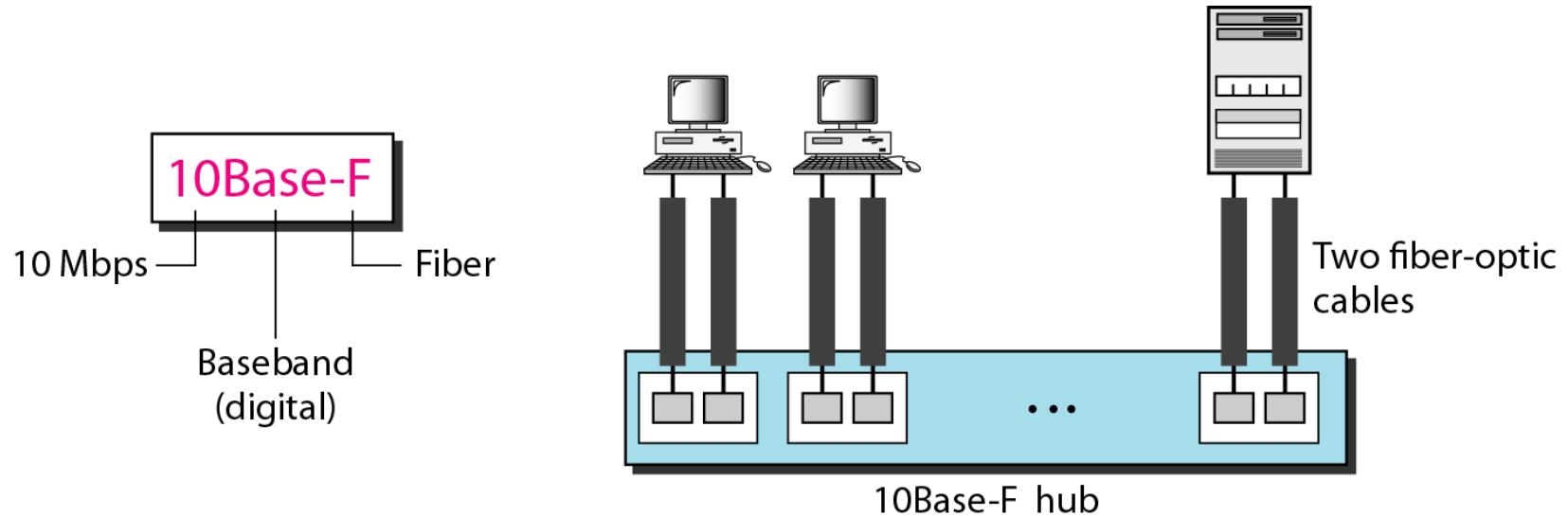


Table 13.1 Summary of Standard Ethernet implementations

<i>Characteristics</i>	<i>10Base5</i>	<i>10Base2</i>	<i>10Base-T</i>	<i>10Base-F</i>
Media	Thick coaxial cable	Thin coaxial cable	2 UTP	2 Fiber
Maximum length	500 m	185 m	100 m	2000 m
Line encoding	Manchester	Manchester	Manchester	Manchester

13-3 CHANGES IN THE STANDARD

The 10-Mbps Standard Ethernet has gone through several changes before moving to the higher data rates. These changes actually opened the road to the evolution of the Ethernet to become compatible with other high-data-rate LANs.

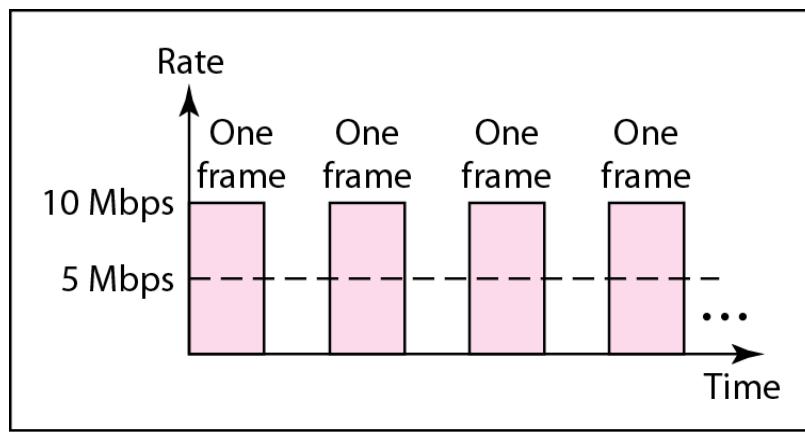
Topics discussed in this section:

Bridged Ethernet

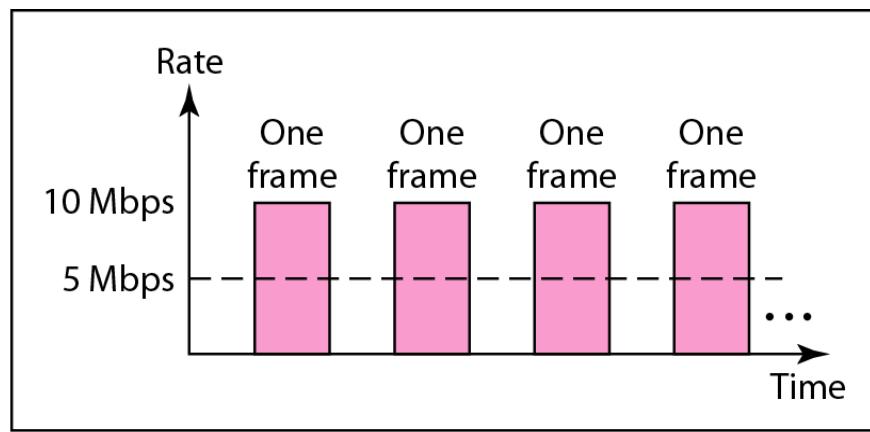
Switched Ethernet

Full-Duplex Ethernet

Figure 13.14 *Sharing bandwidth*

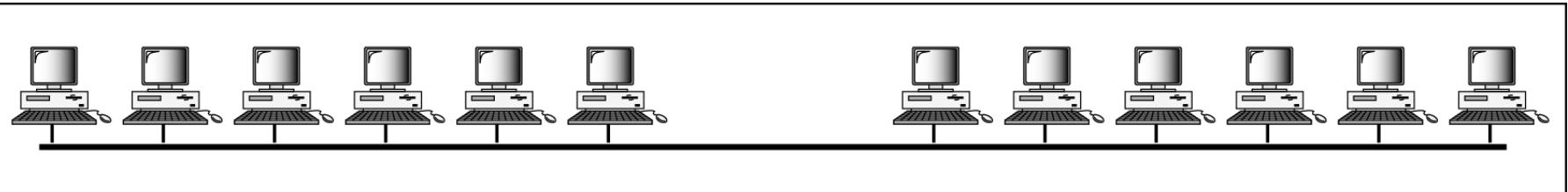


a. First station

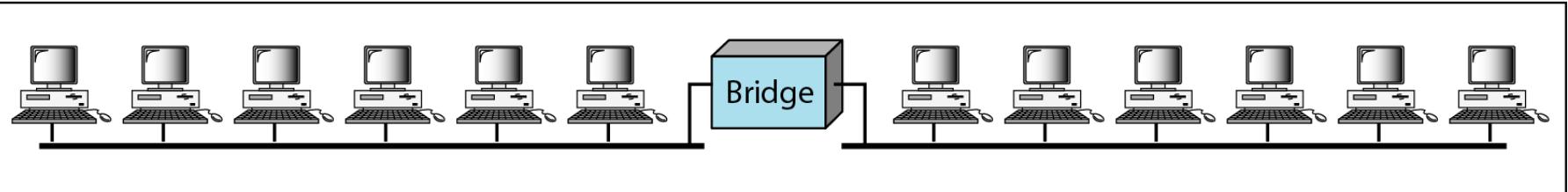


b. Second station

Figure 13.15 *A network with and without a bridge*



a. Without bridging



b. With bridging

Figure 13.16 Collision domains in an unbridged network and a bridged network

Domain



a. Without bridging

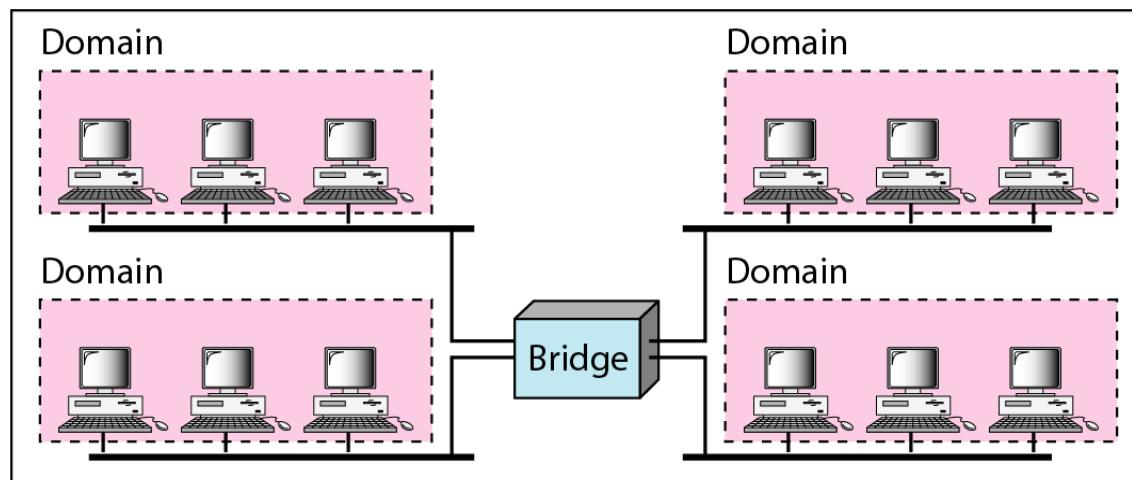
Domain

Domain

Domain

Domain

Bridge



b. With bridging

Figure 13.17 *Switched Ethernet*

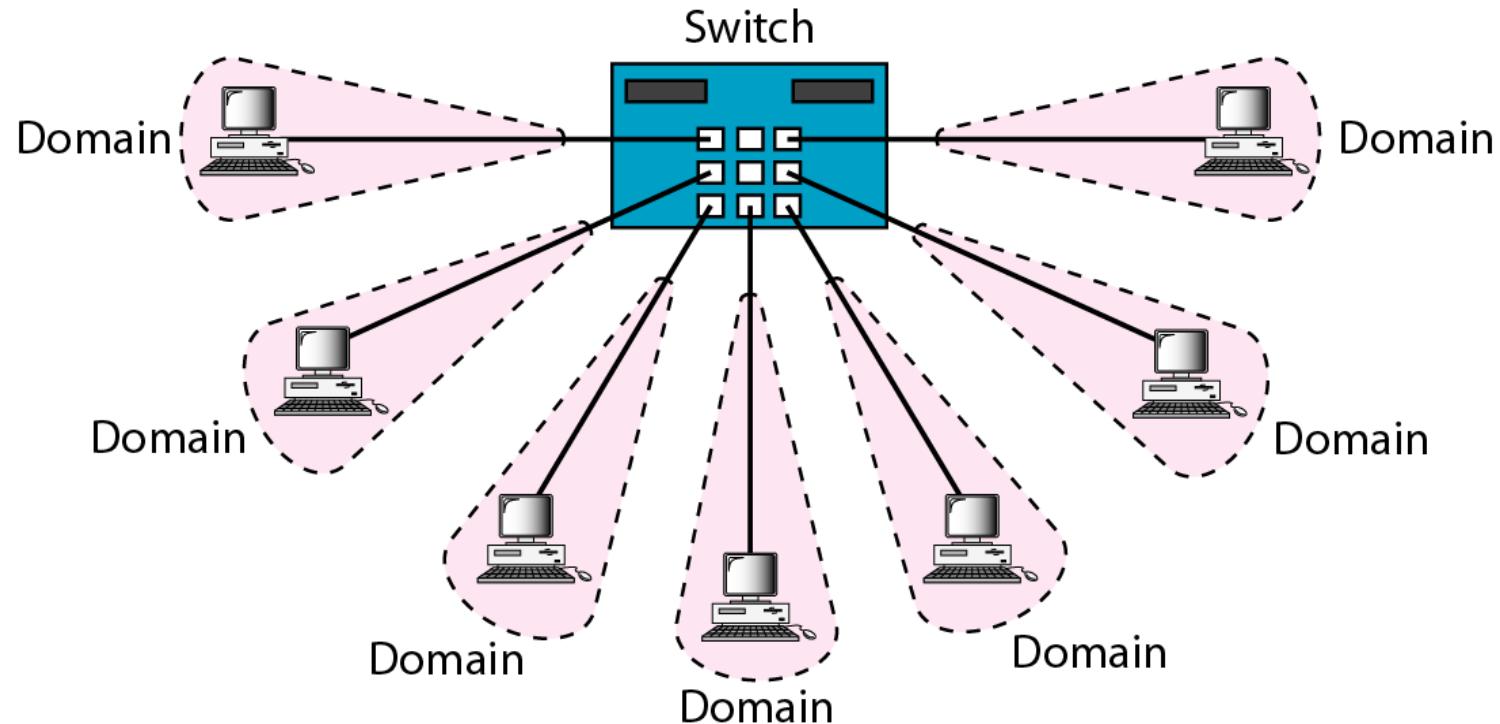
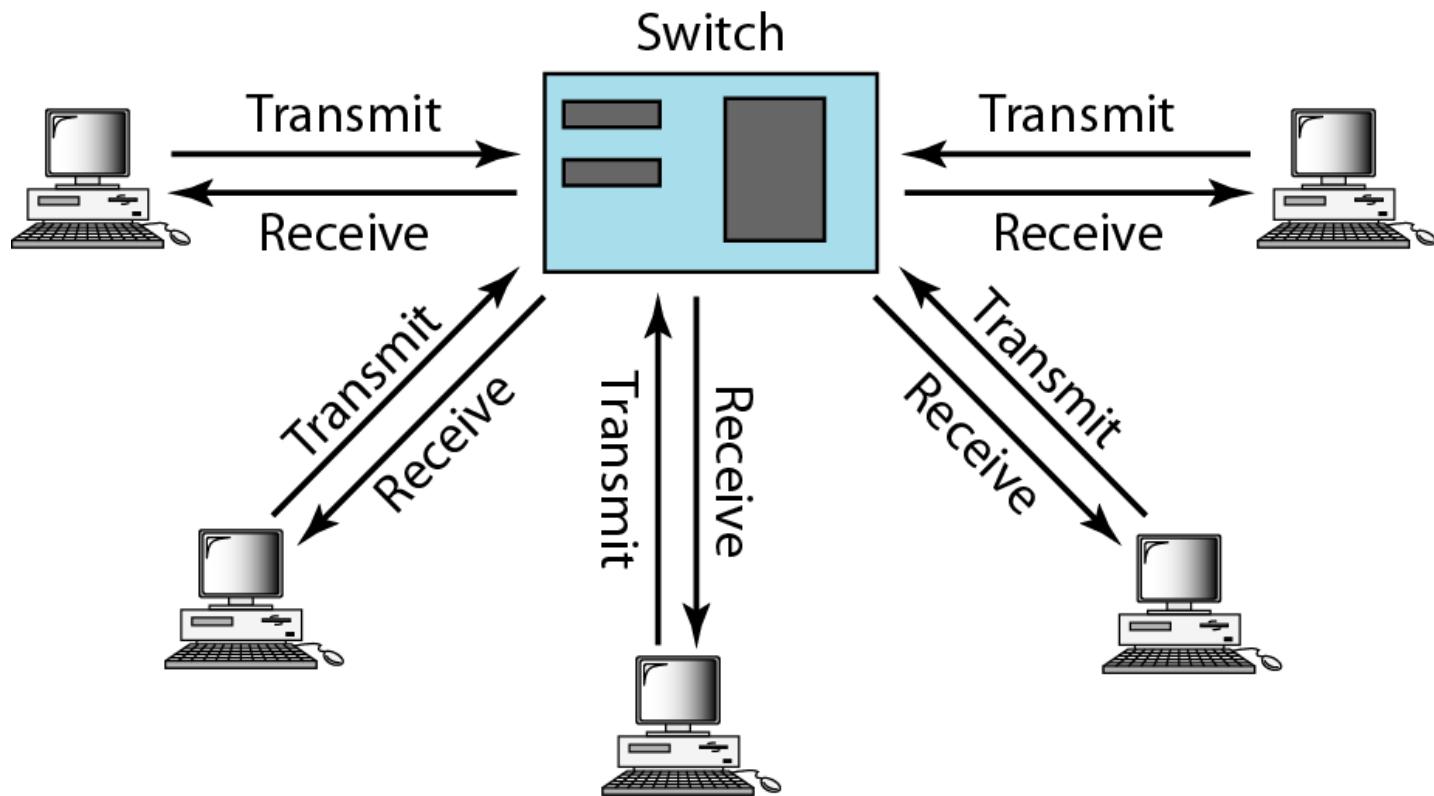


Figure 13.18 Full-duplex switched Ethernet



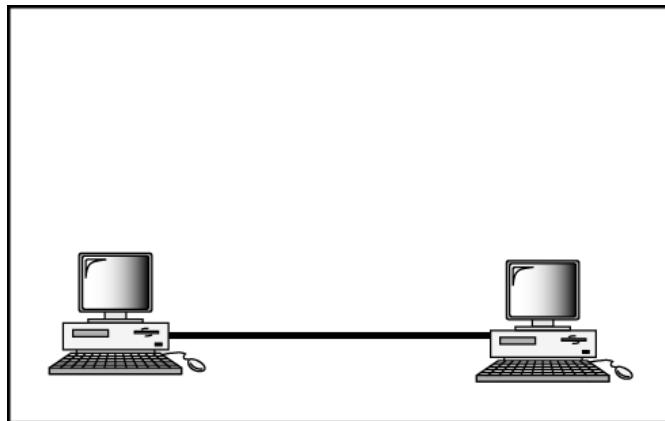
13-4 FAST ETHERNET

Fast Ethernet was designed to compete with LAN protocols such as FDDI or Fiber Channel. IEEE created Fast Ethernet under the name 802.3u. Fast Ethernet is backward-compatible with Standard Ethernet, but it can transmit data 10 times faster at a rate of 100 Mbps.

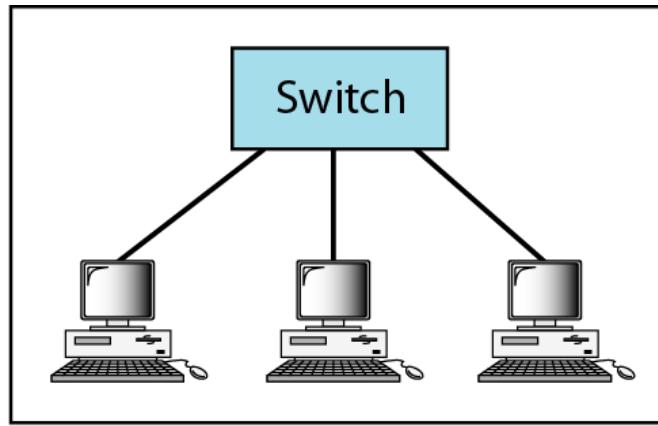
Topics discussed in this section:

MAC Sublayer
Physical Layer

Figure 13.19 *Fast Ethernet topology*



a. Point-to-point



b. Star

Figure 13.20 *Fast Ethernet implementations*

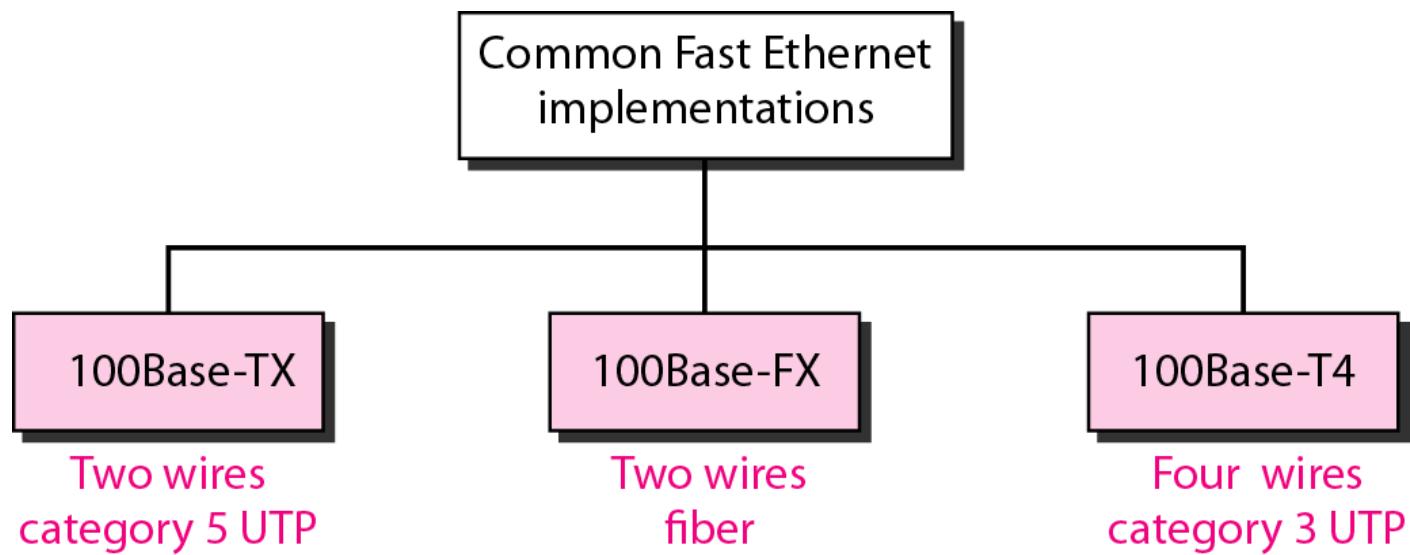


Figure 13.21 Encoding for Fast Ethernet implementation

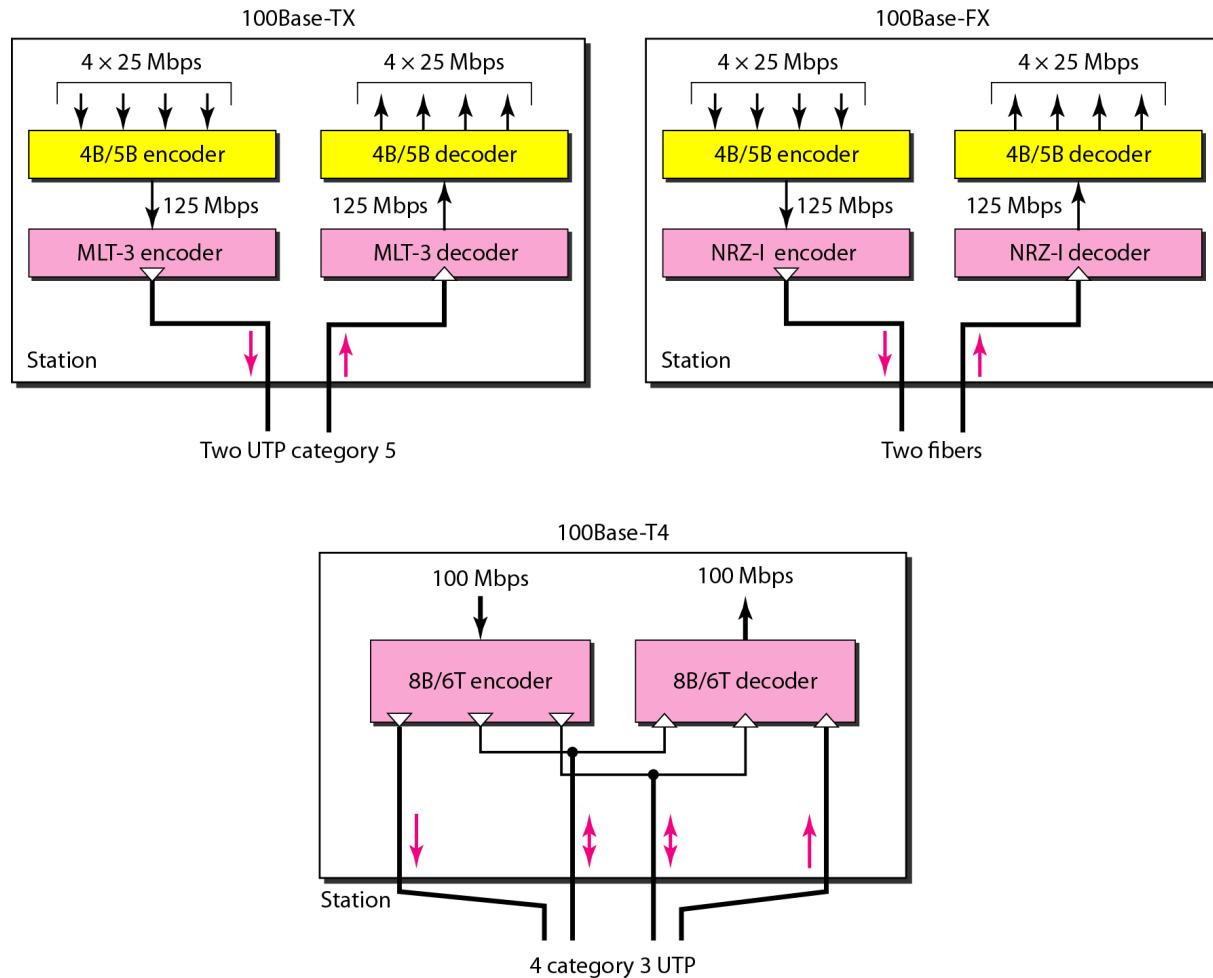


Table 13.2 *Summary of Fast Ethernet implementations*

<i>Characteristics</i>	<i>100Base-TX</i>	<i>100Base-FX</i>	<i>100Base-T4</i>
Media	Cat 5 UTP or STP	Fiber	Cat 4 UTP
Number of wires	2	2	4
Maximum length	100 m	100 m	100 m
Block encoding	4B/5B	4B/5B	
Line encoding	MLT-3	NRZ-I	8B/6T

13-5 GIGABIT ETHERNET

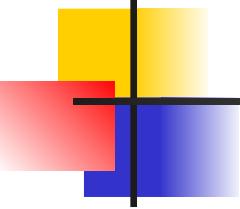
The need for an even higher data rate resulted in the design of the Gigabit Ethernet protocol (1000 Mbps). The IEEE committee calls the standard 802.3z.

Topics discussed in this section:

MAC Sublayer

Physical Layer

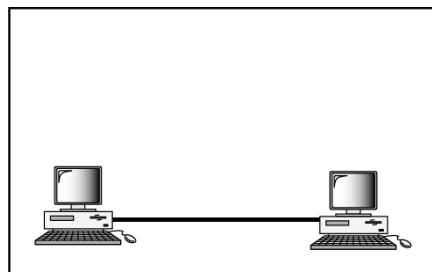
Ten-Gigabit Ethernet



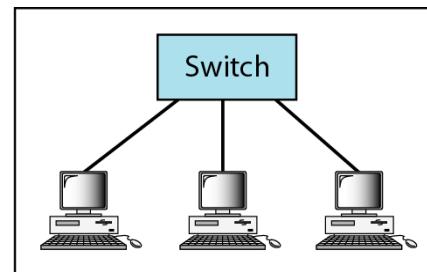
Note

In the full-duplex mode of Gigabit Ethernet, there is no collision; the maximum length of the cable is determined by the signal attenuation in the cable.

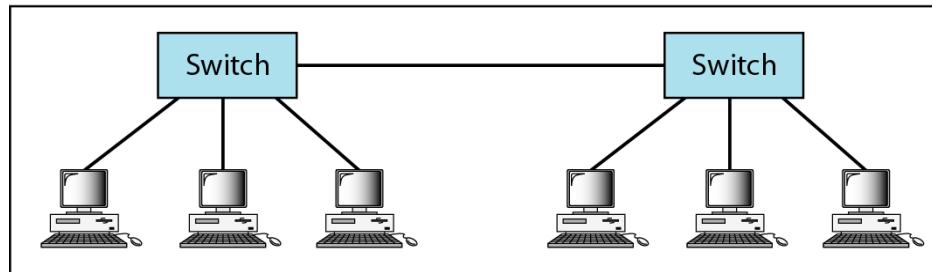
Figure 13.22 Topologies of Gigabit Ethernet



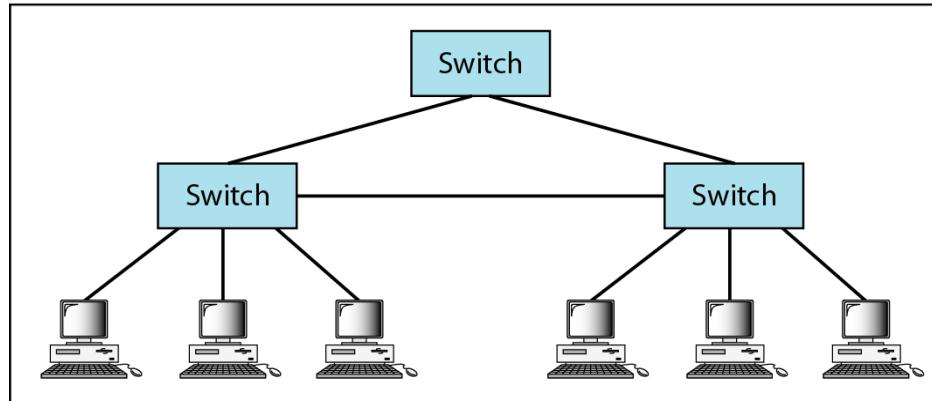
a. Point-to-point



b. Star



c. Two stars



d. Hierarchy of stars

Figure 13.23 *Gigabit Ethernet implementations*

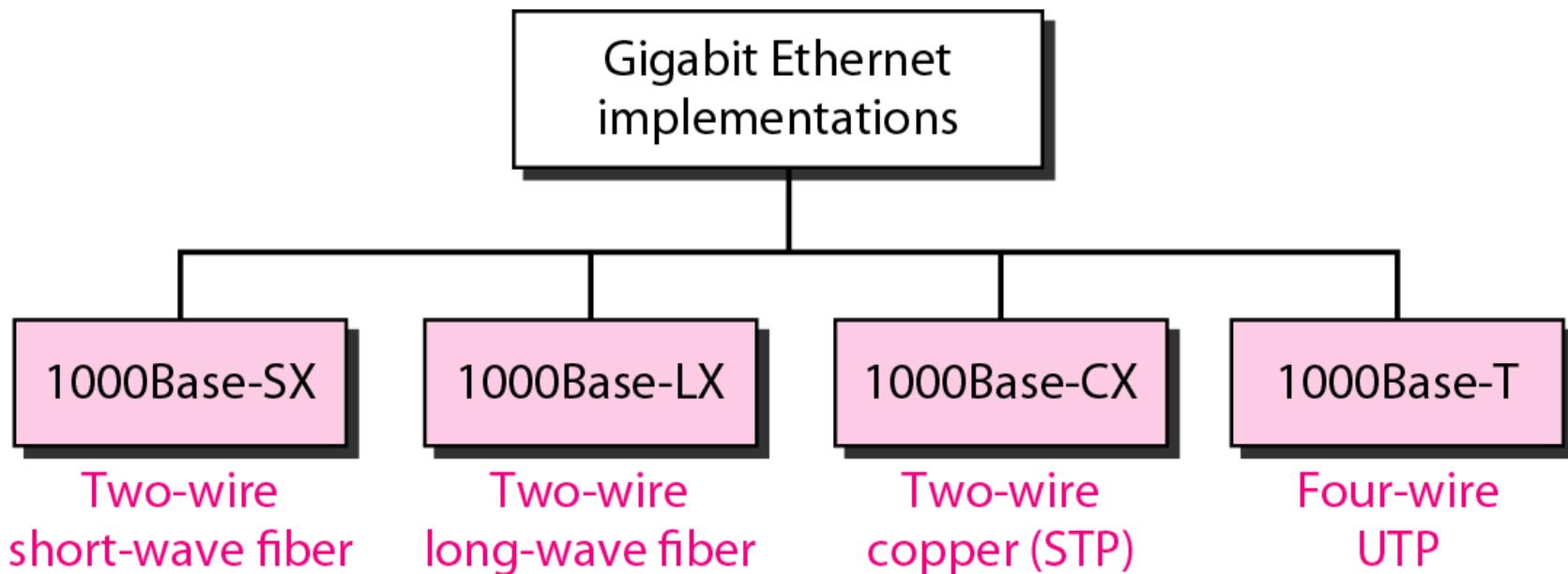


Figure 13.24 Encoding in Gigabit Ethernet implementations

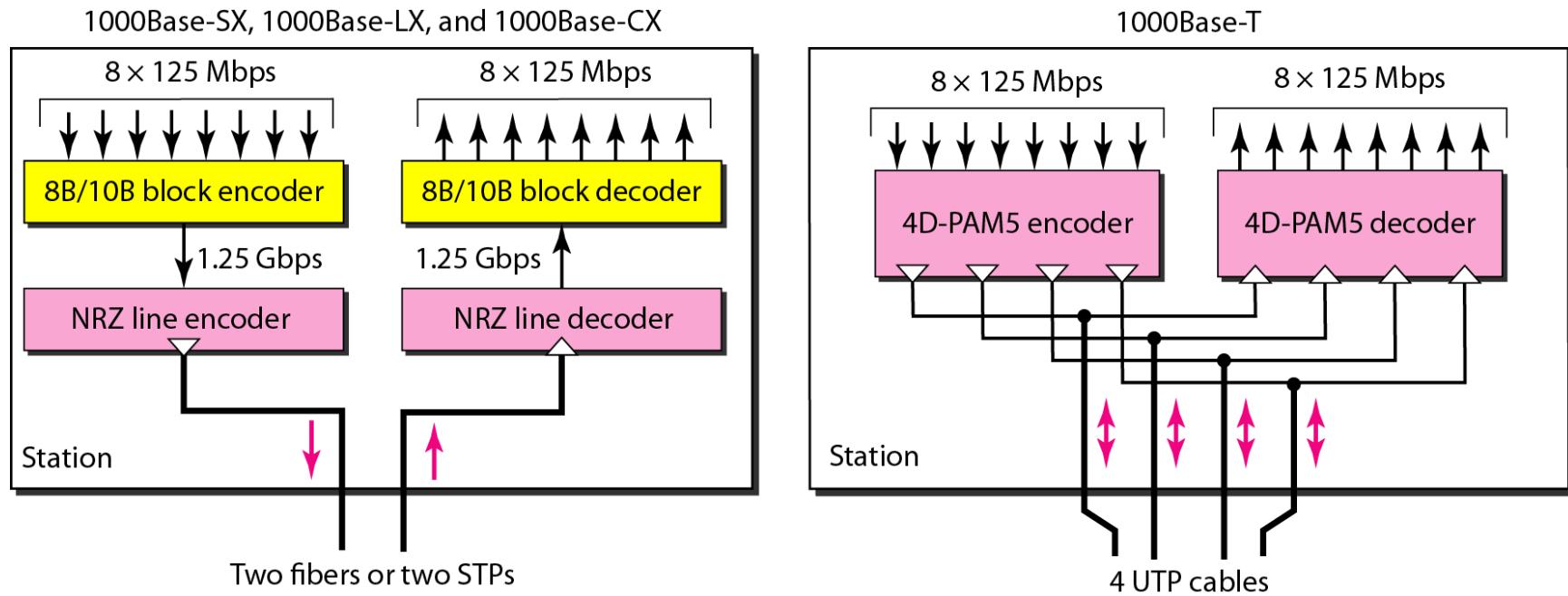


Table 13.3 Summary of Gigabit Ethernet implementations

<i>Characteristics</i>	<i>1000Base-SX</i>	<i>1000Base-LX</i>	<i>1000Base-CX</i>	<i>1000Base-T</i>
Media	Fiber short-wave	Fiber long-wave	STP	Cat 5 UTP
Number of wires	2	2	2	4
Maximum length	550 m	5000 m	25 m	100 m
Block encoding	8B/10B	8B/10B	8B/10B	
Line encoding	NRZ	NRZ	NRZ	4D-PAM5

Table 13.4 Summary of Ten-Gigabit Ethernet implementations

<i>Characteristics</i>	<i>10GBase-S</i>	<i>10GBase-L</i>	<i>10GBase-E</i>
Media	Short-wave 850-nm multimode	Long-wave 1310-nm single mode	Extended 1550-mm single mode
Maximum length	300 m	10 km	40 km

Chapter 14

Wireless LANs

14-1 IEEE 802.11

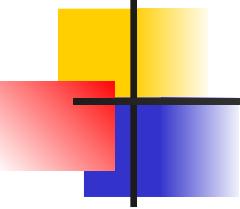
IEEE has defined the specifications for a wireless LAN, called IEEE 802.11, which covers the physical and data link layers.

Topics discussed in this section:

Architecture

MAC Sublayer

Physical Layer



Note

A BSS without an AP is called an ad hoc network;

a BSS with an AP is called an infrastructure network.

Figure 14.1 Basic service sets (BSSs)

BSS: Basic service set

AP: Access point

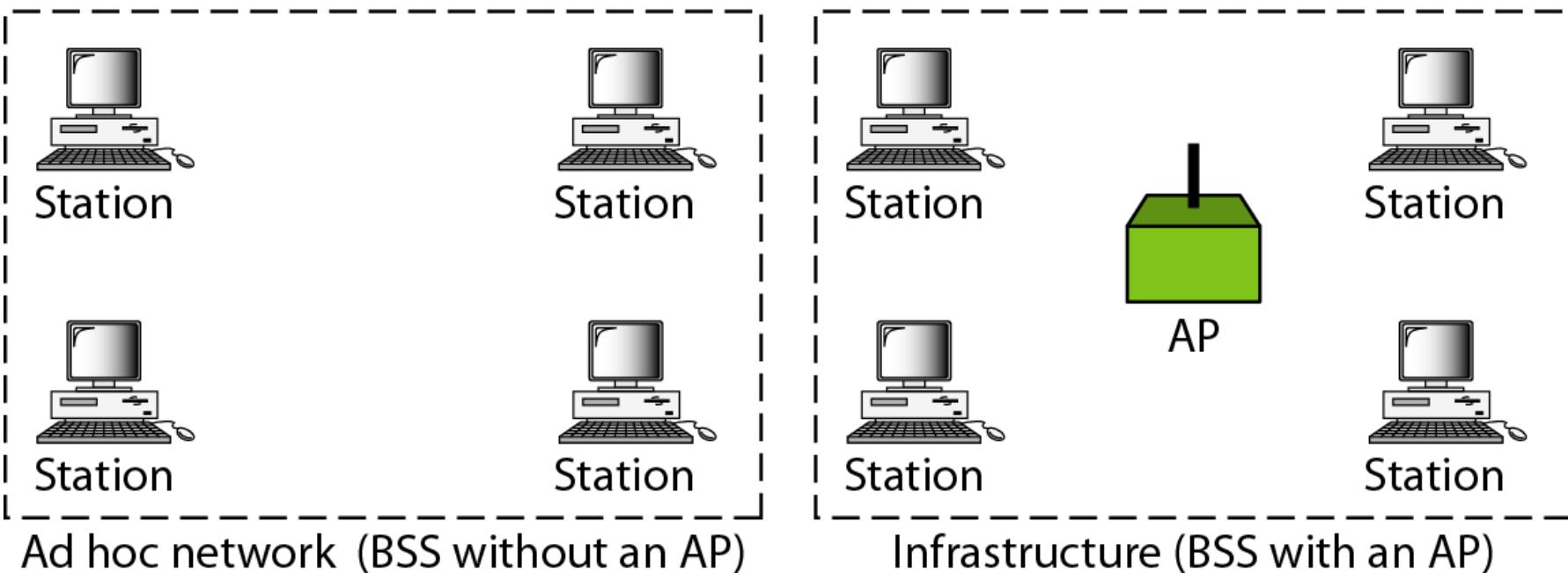


Figure 14.2 *Extended service sets (ESSs)*

ESS: Extended service set

BSS: Basic service set

AP: Access point

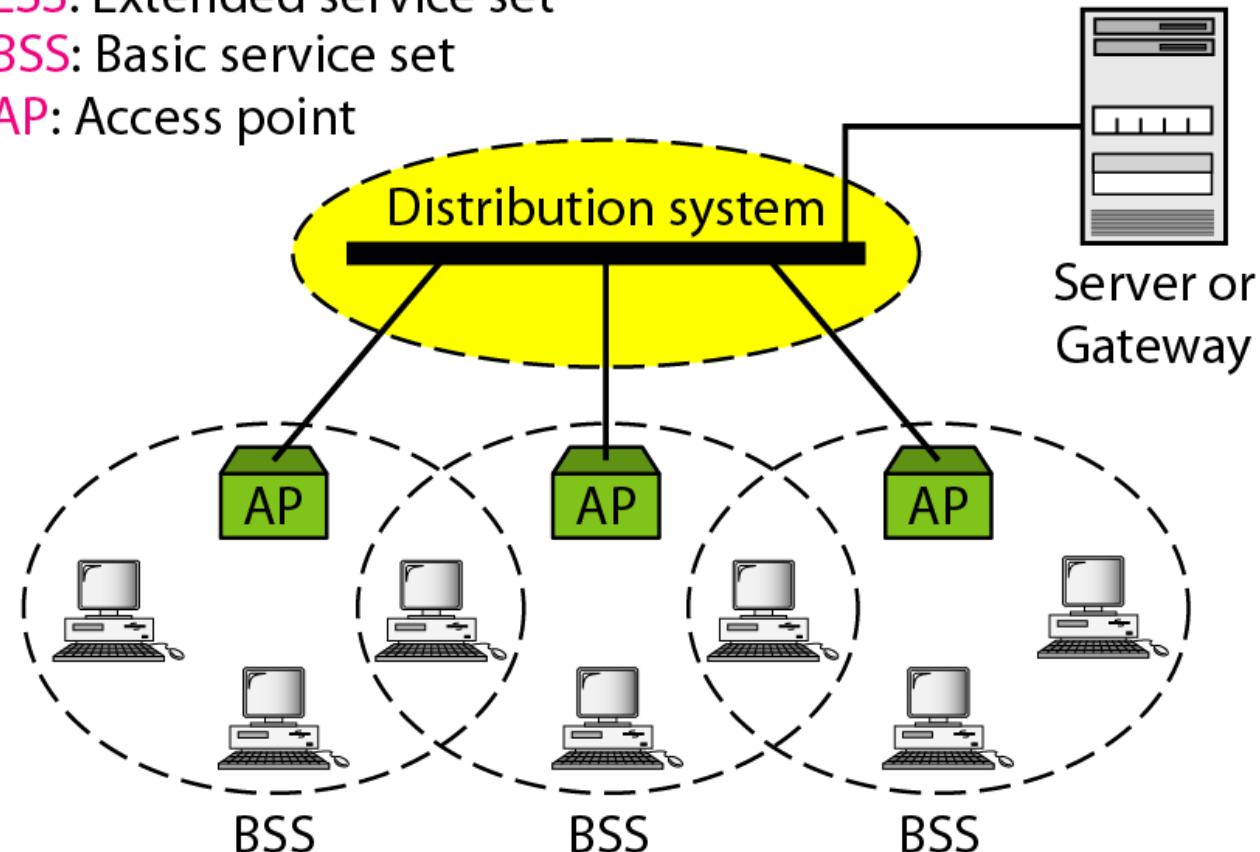


Figure 14.3 MAC layers in IEEE 802.11 standard

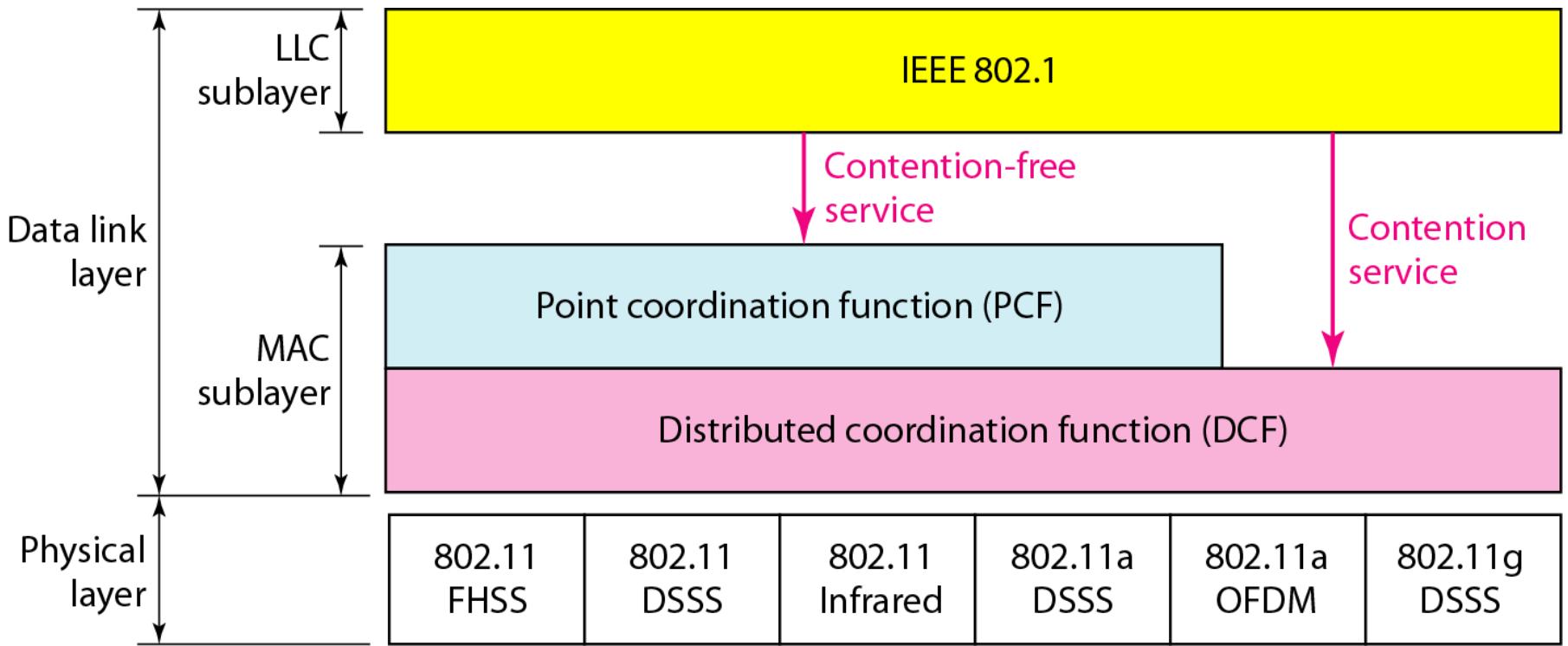


Figure 14.4 CSMA/CA flowchart

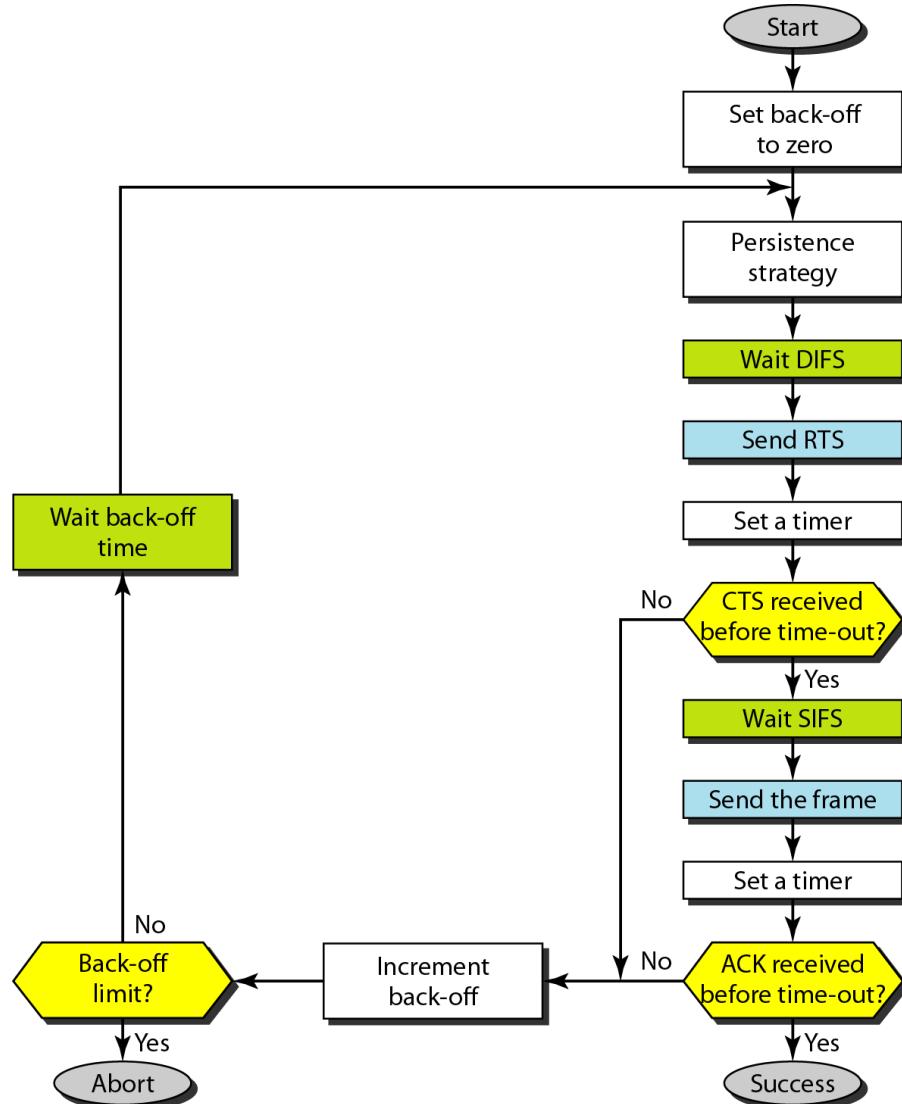


Figure 14.5 CSMA/CA and NAV

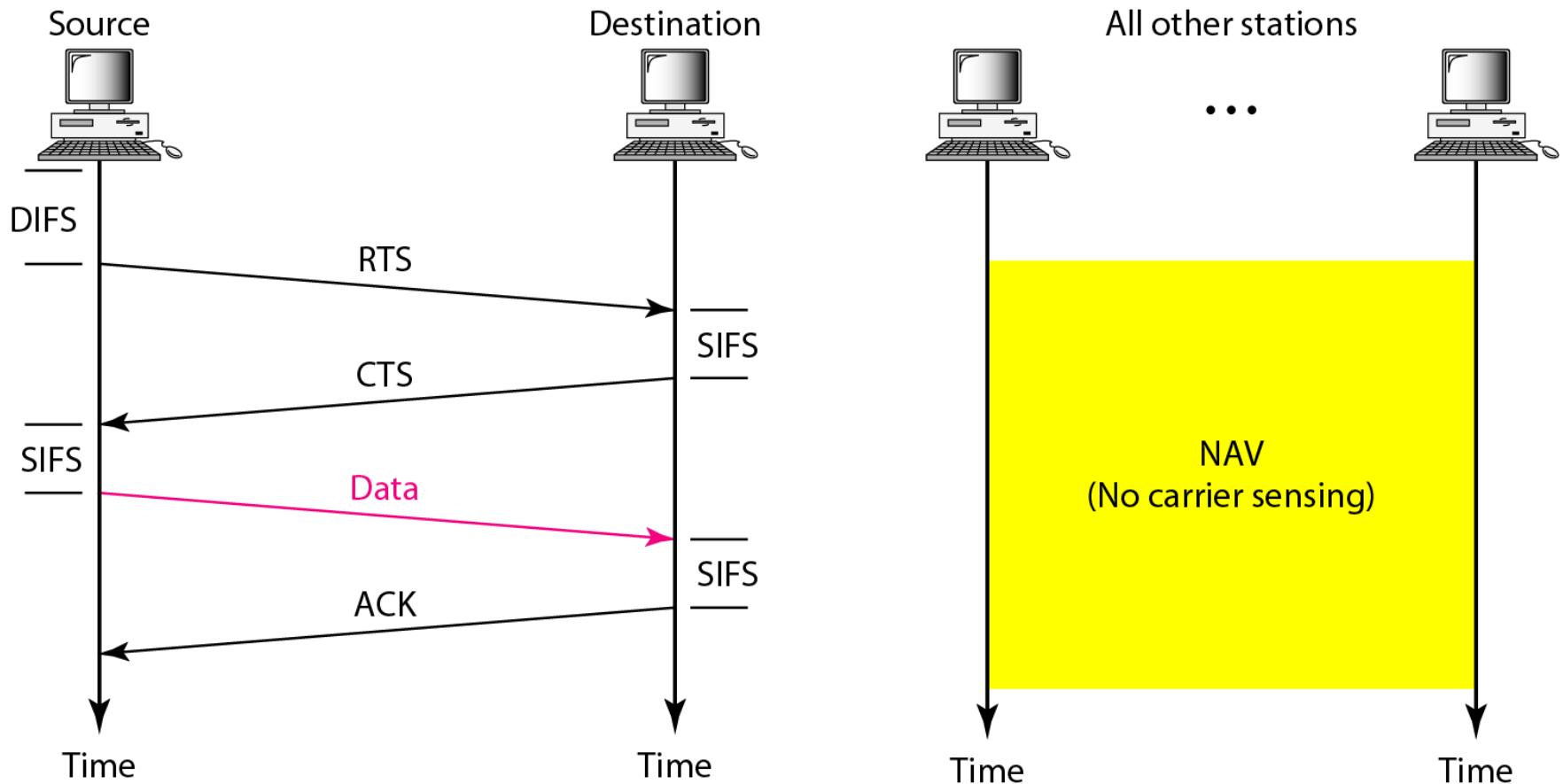


Figure 14.6 Example of repetition interval

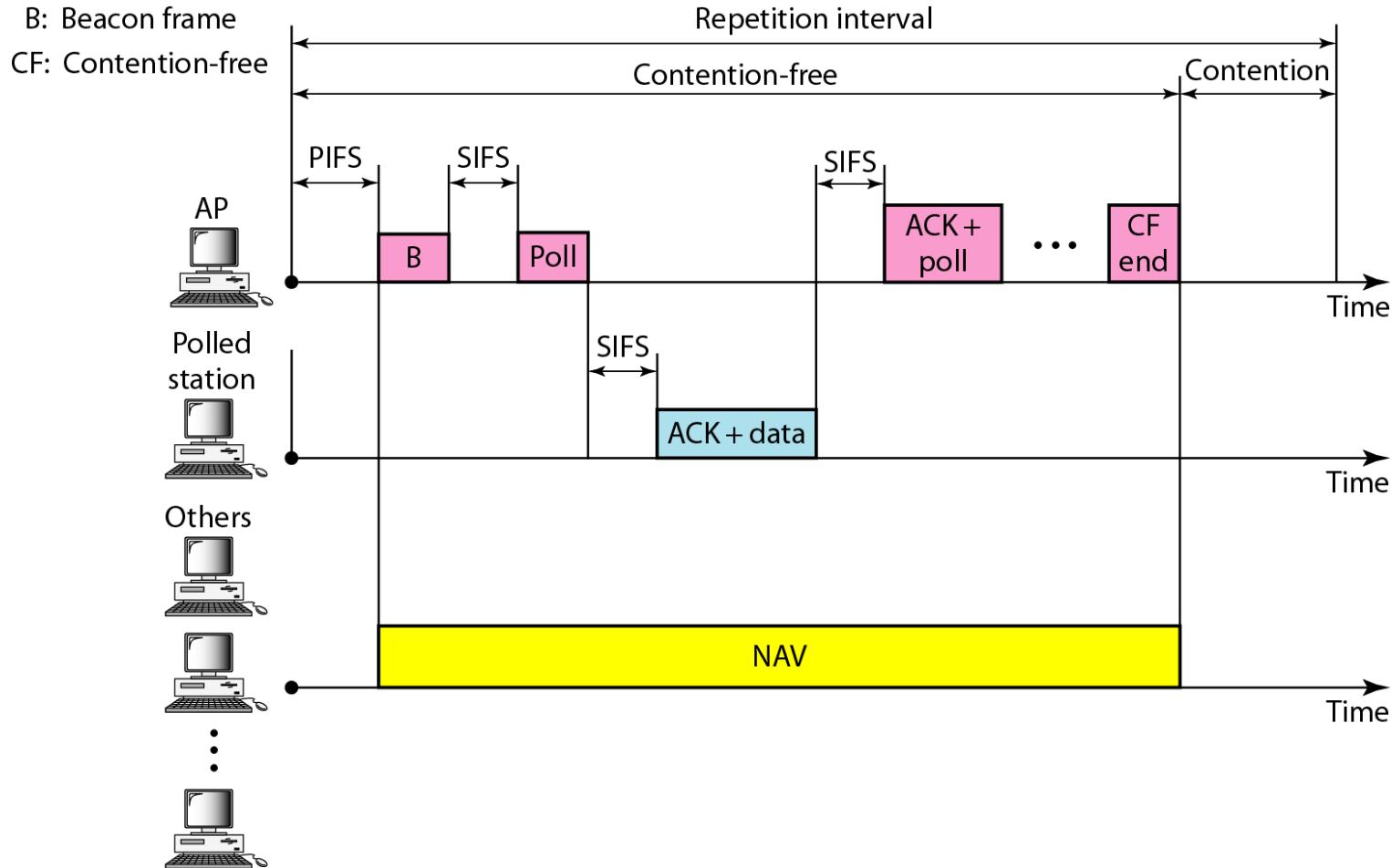


Figure 14.7 *Frame format*

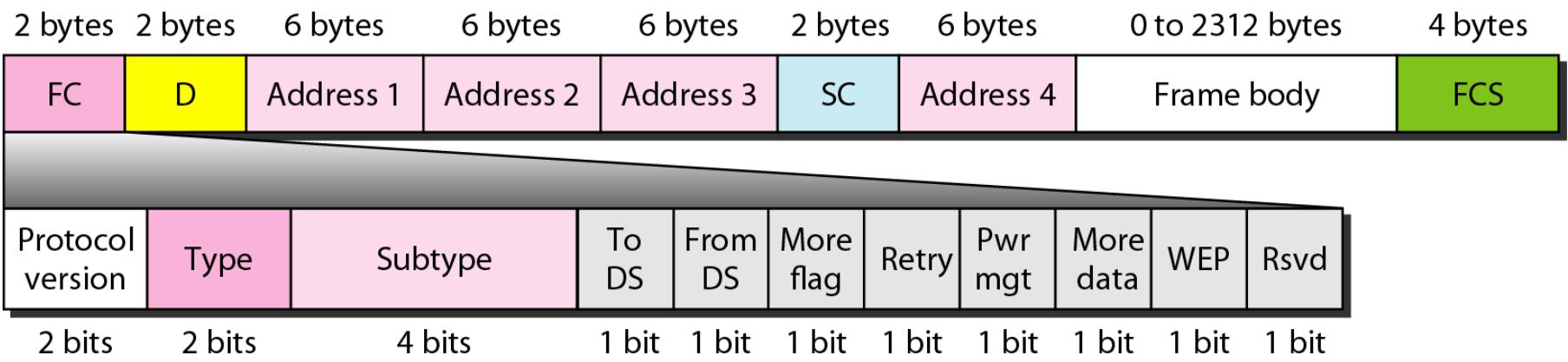


Table 14.1 *Subfields in FC field*

Field	Explanation
Version	Current version is 0
Type	Type of information: management (00), control (01), or data (10)
Subtype	Subtype of each type (see Table 14.2)
To DS	Defined later
From DS	Defined later
More flag	When set to 1, means more fragments
Retry	When set to 1, means retransmitted frame
Pwr mgt	When set to 1, means station is in power management mode
More data	When set to 1, means station has more data to send
WEP	Wired equivalent privacy (encryption implemented)
Rsvd	Reserved

Figure 14.8 *Control frames*

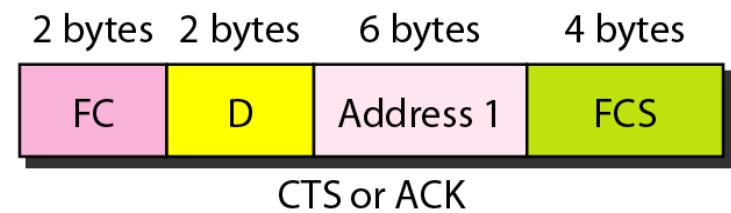
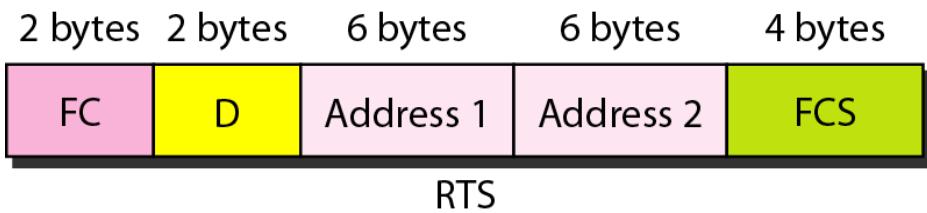


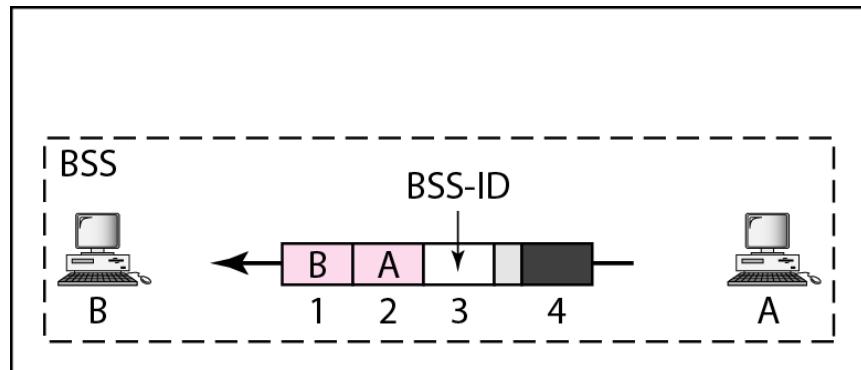
Table 14.2 *Values of subfields in control frames*

<i>Subtype</i>	<i>Meaning</i>
1011	Request to send (RTS)
1100	Clear to send (CTS)
1101	Acknowledgment (ACK)

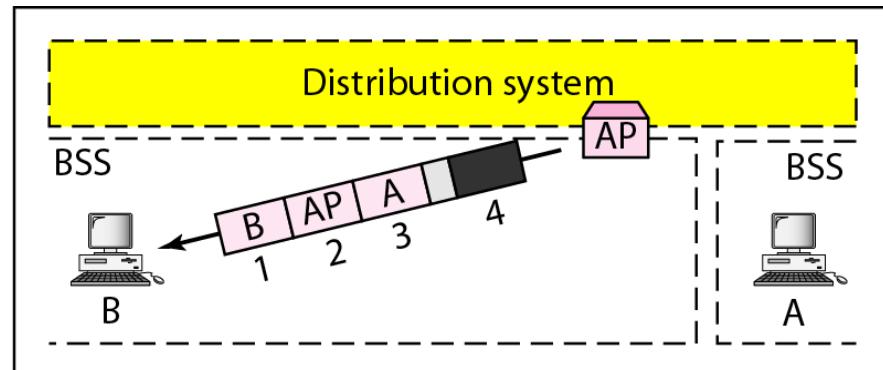
Table 14.3 Addresses

To DS	From DS	Address 1	Address 2	Address 3	Address 4
0	0	Destination	Source	BSS ID	N/A
0	1	Destination	Sending AP	Source	N/A
1	0	Receiving AP	Source	Destination	N/A
1	1	Receiving AP	Sending AP	Destination	Source

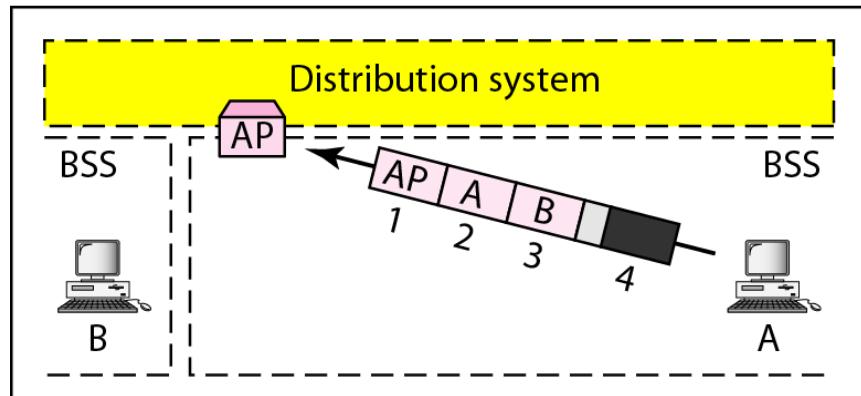
Figure 14.9 Addressing mechanisms



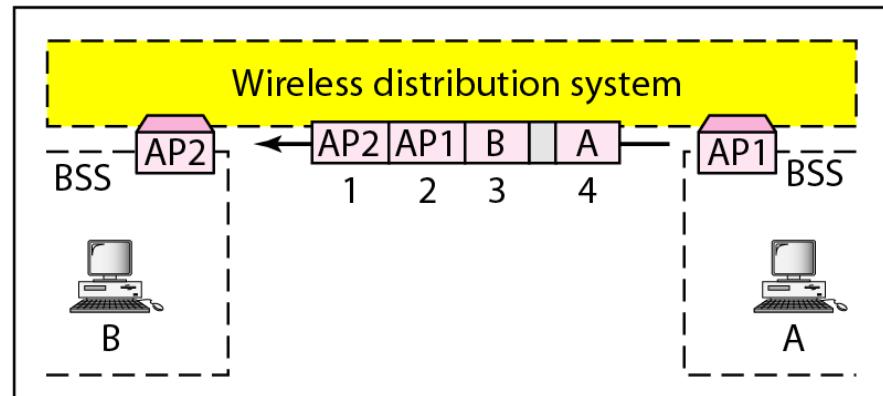
a. Case 1



b. Case 2

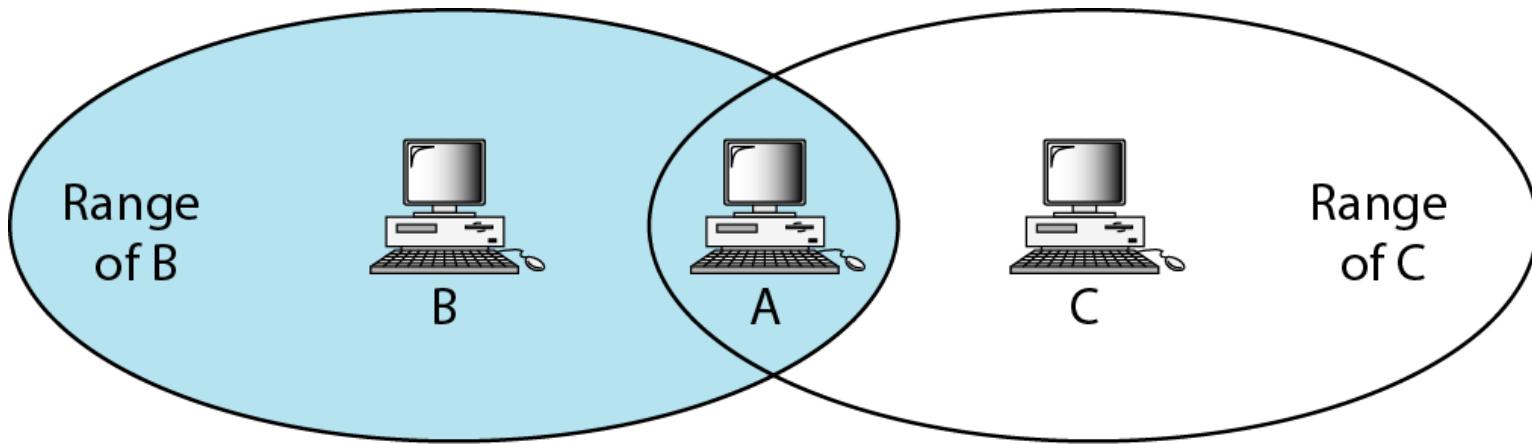


c. Case 3

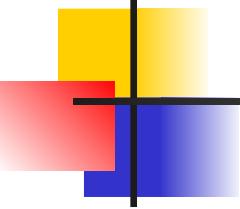


d. Case 4

Figure 14.10 *Hidden station problem*



B and C are hidden from each other with respect to A.



Note

**The CTS frame in CSMA/CA handshake
can prevent collision from
a hidden station.**

Figure 14.11 *Use of handshaking to prevent hidden station problem*

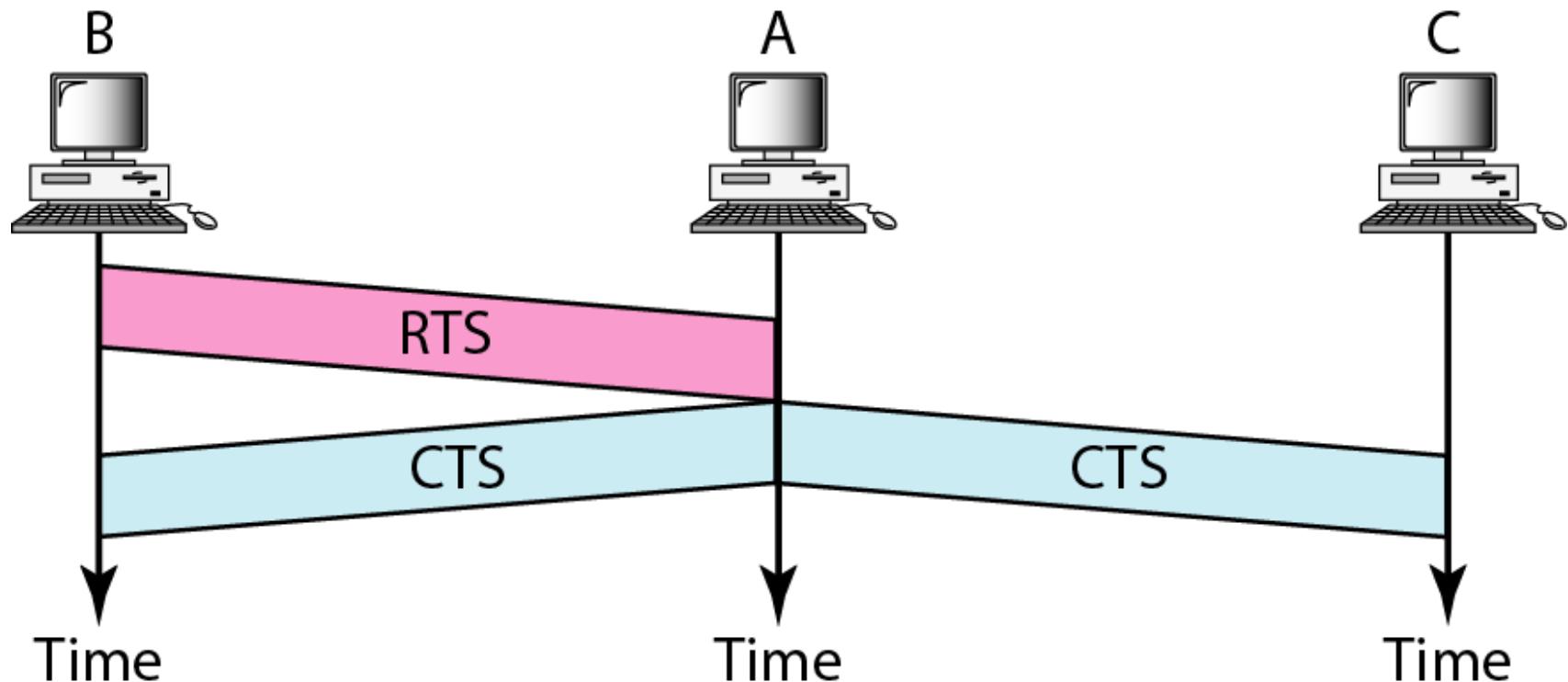


Figure 14.12 Exposed station problem

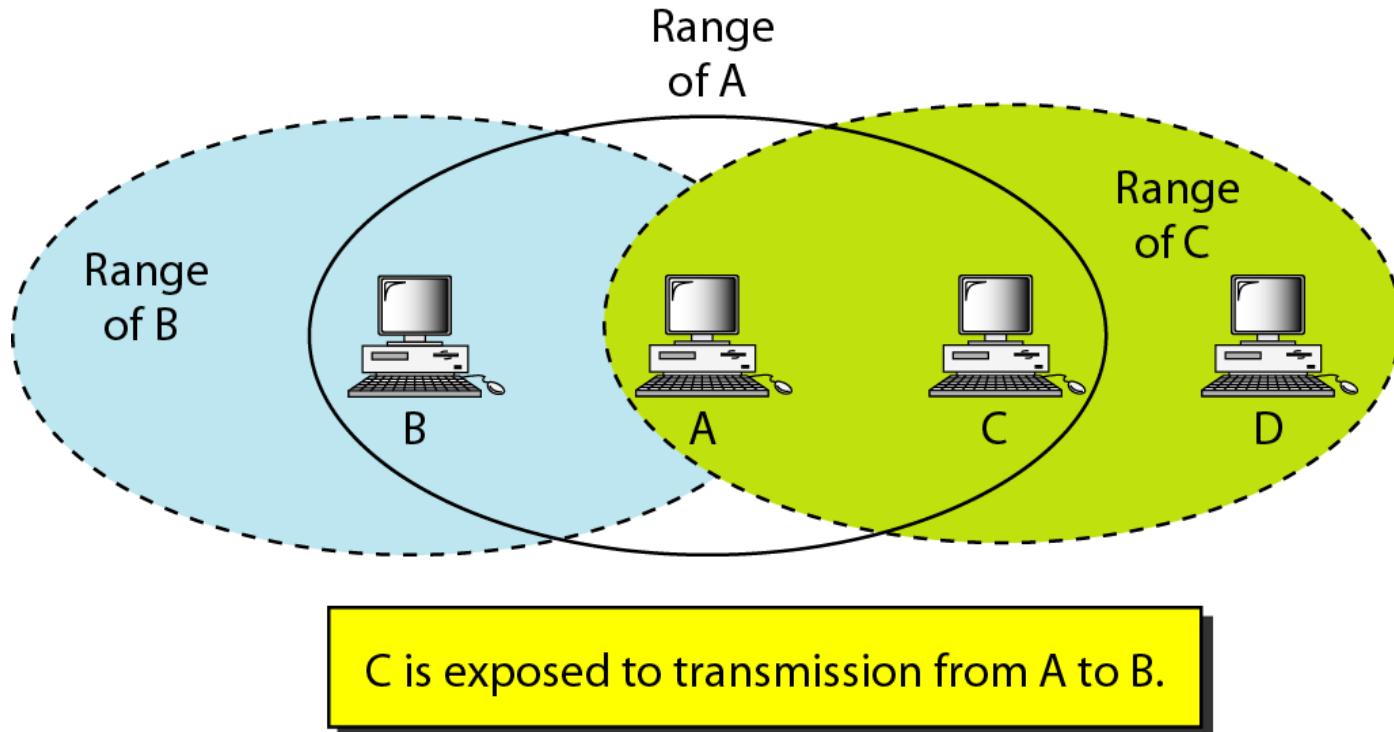


Figure 14.13 Use of handshaking in exposed station problem

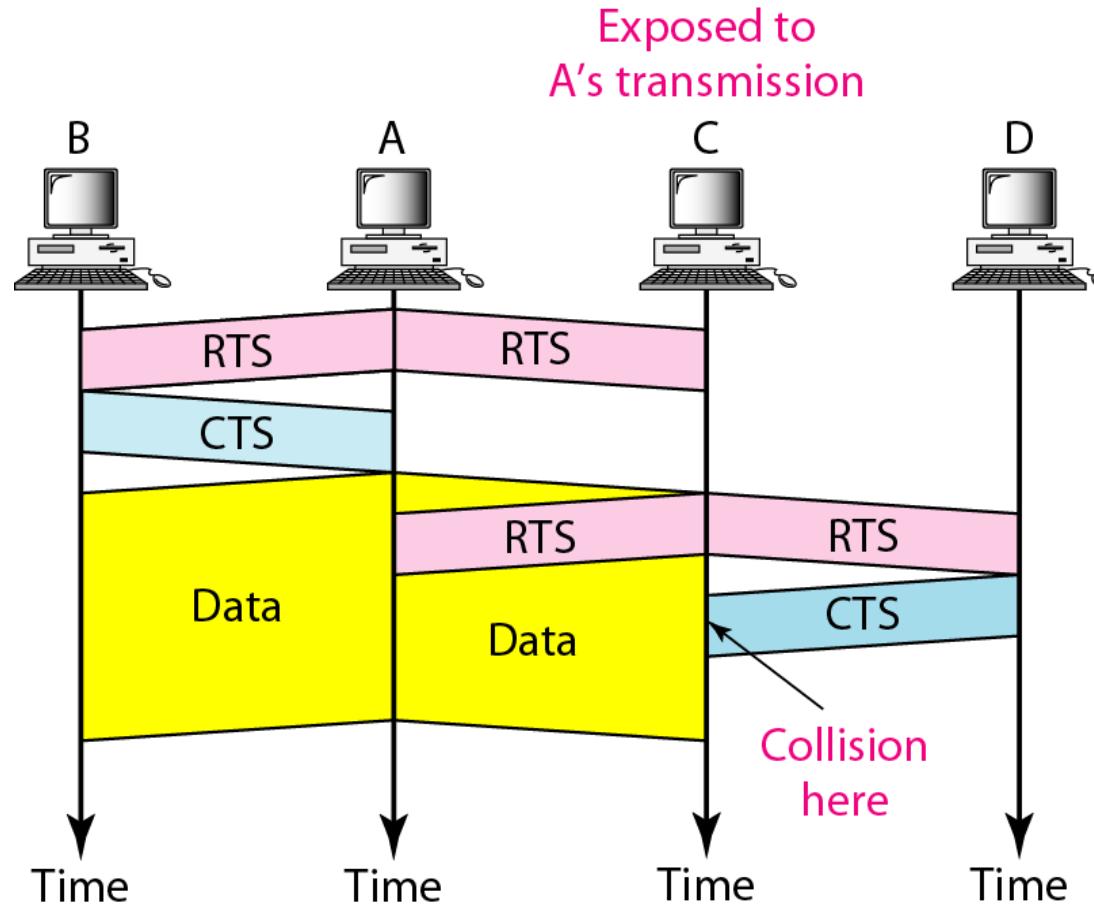


Table 14.4 Physical layers

<i>IEEE</i>	<i>Technique</i>	<i>Band</i>	<i>Modulation</i>	<i>Rate (Mbps)</i>
802.11	FHSS	2.4 GHz	FSK	1 and 2
	DSSS	2.4 GHz	PSK	1 and 2
		Infrared	PPM	1 and 2
802.11a	OFDM	5.725 GHz	PSK or QAM	6 to 54
802.11b	DSSS	2.4 GHz	PSK	5.5 and 11
802.11g	OFDM	2.4 GHz	Different	22 and 54

Figure 14.14 Industrial, scientific, and medical (ISM) band

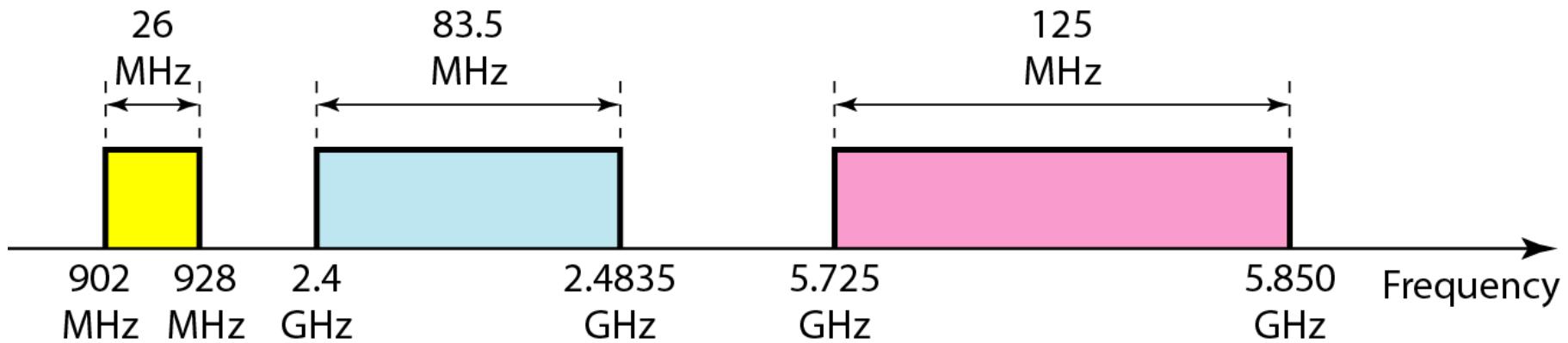


Figure 14.15 Physical layer of IEEE 802.11 FHSS

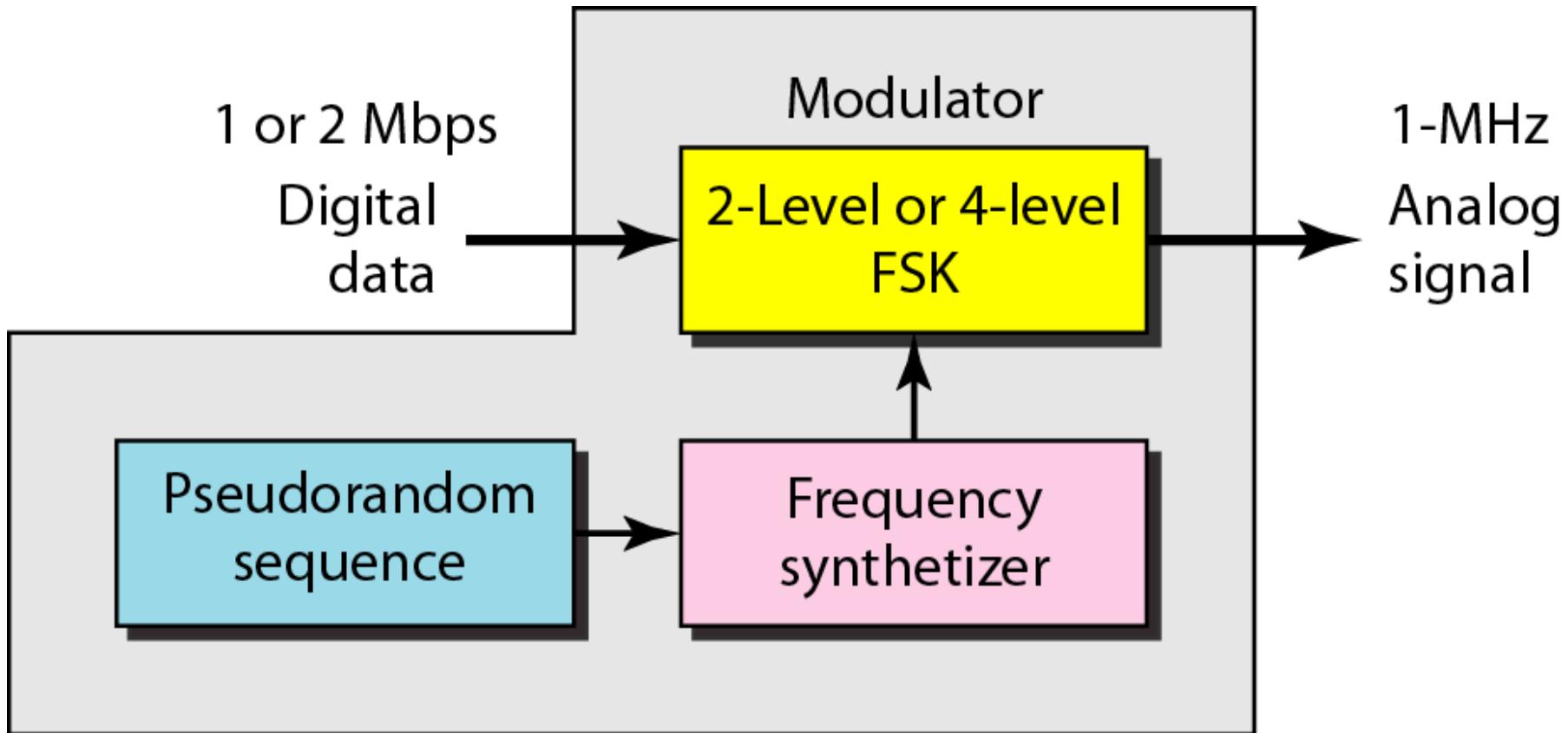


Figure 14.16 Physical layer of IEEE 802.11 DSSS

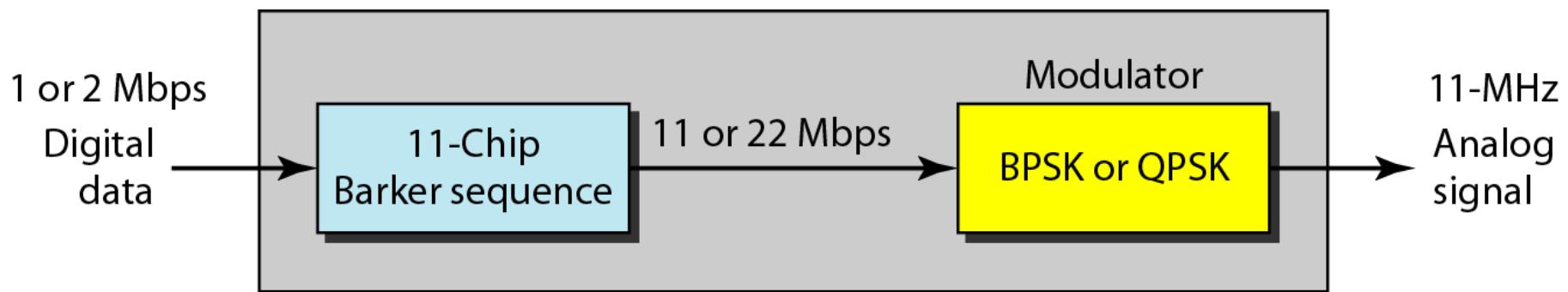


Figure 14.17 Physical layer of IEEE 802.11 infrared

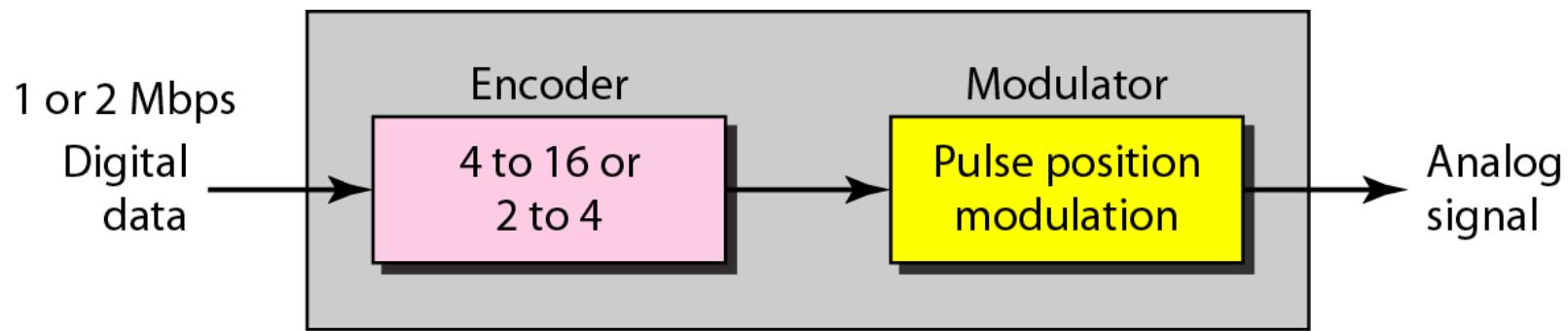
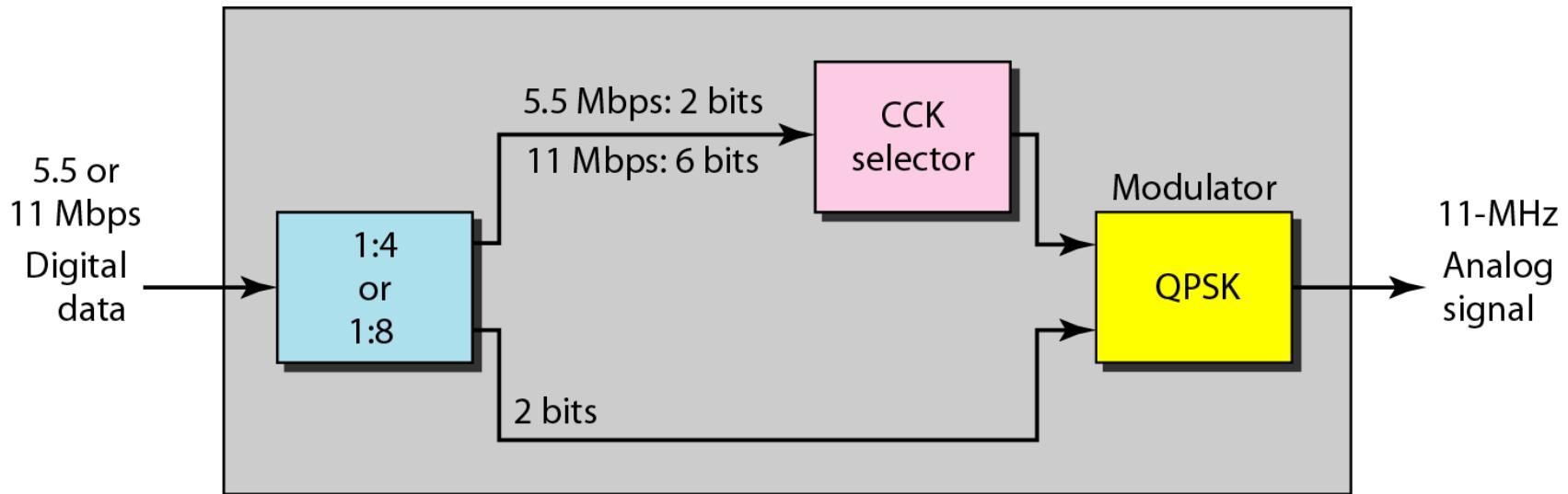


Figure 14.18 Physical layer of IEEE 802.11b



14-2 BLUETOOTH

Bluetooth is a wireless LAN technology designed to connect devices of different functions such as telephones, notebooks, computers, cameras, printers, coffee makers, and so on. A Bluetooth LAN is an ad hoc network, which means that the network is formed spontaneously.

Topics discussed in this section:

Architecture

Bluetooth Layers

Baseband Layer

L2CAP

Figure 14.19 Piconet

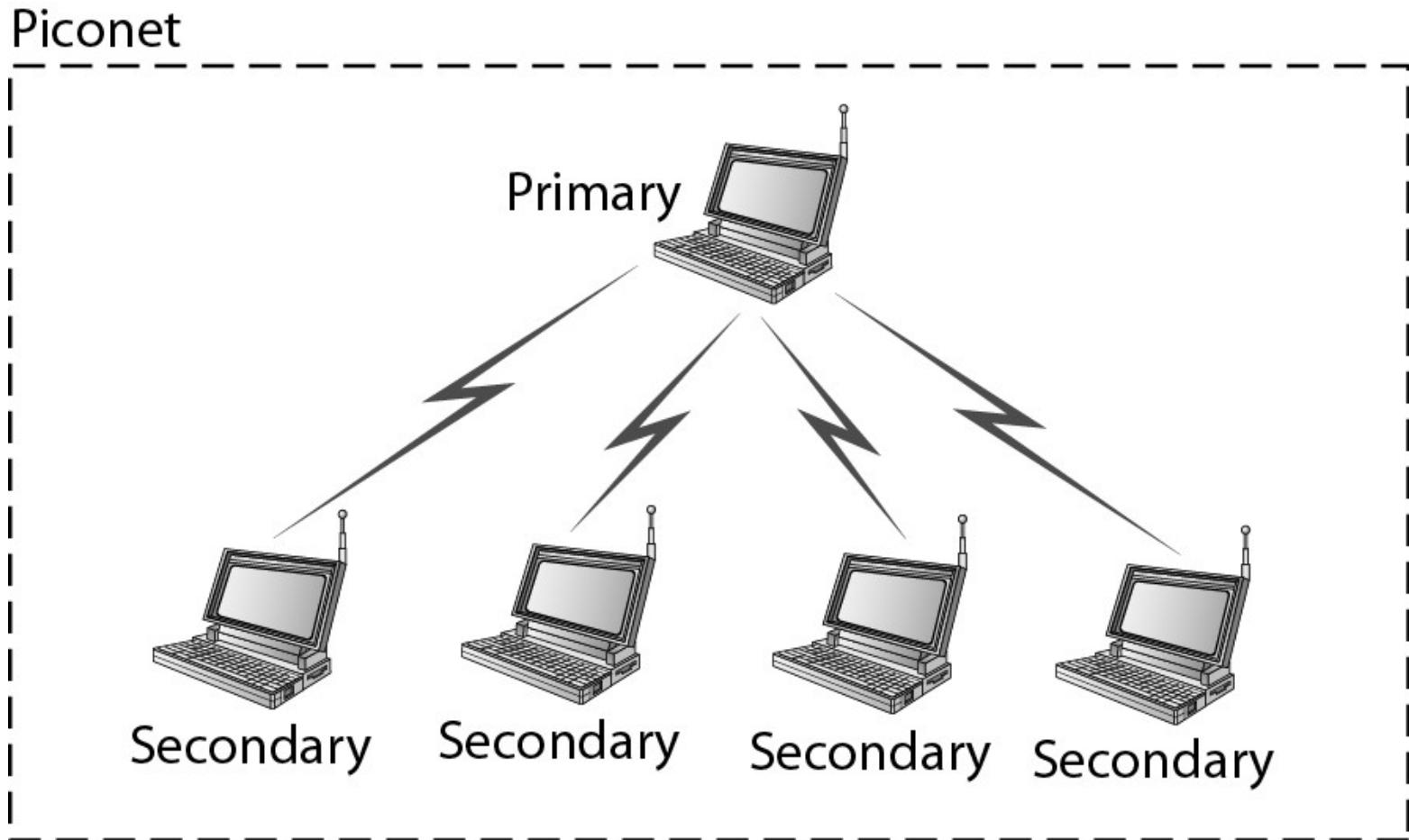


Figure 14.20 Scatternet

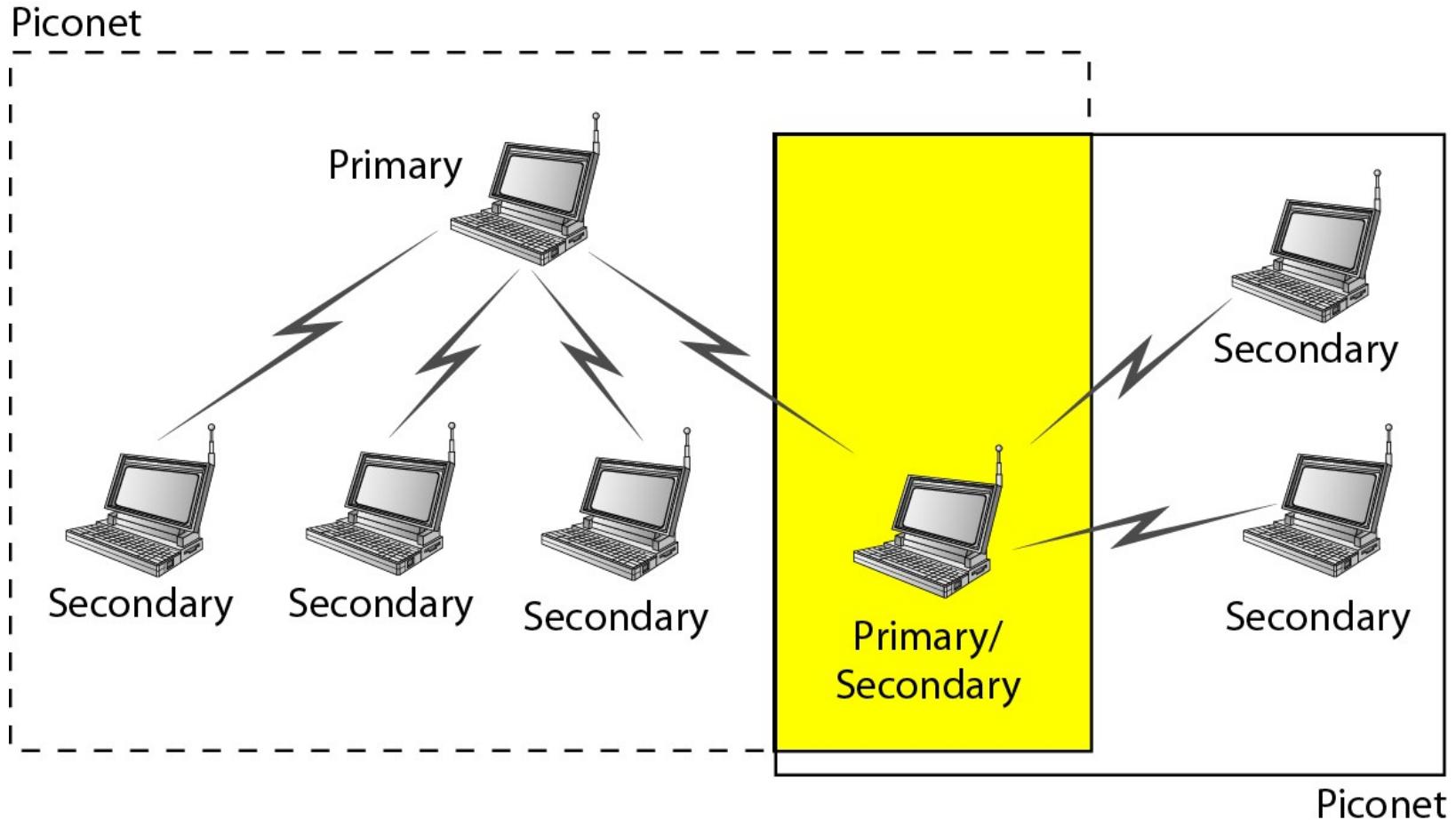


Figure 14.21 *Bluetooth layers*

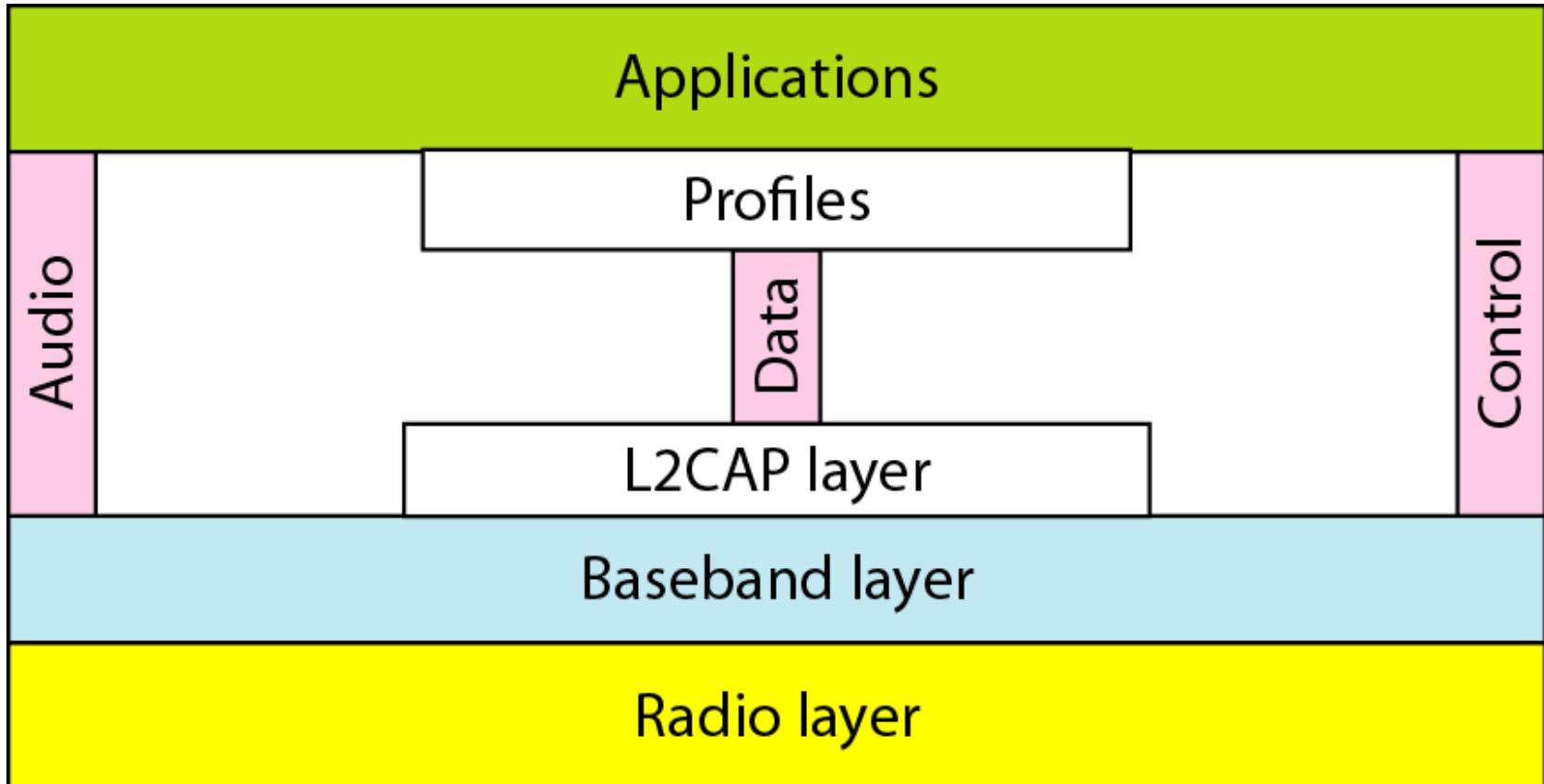


Figure 14.22 Single-secondary communication

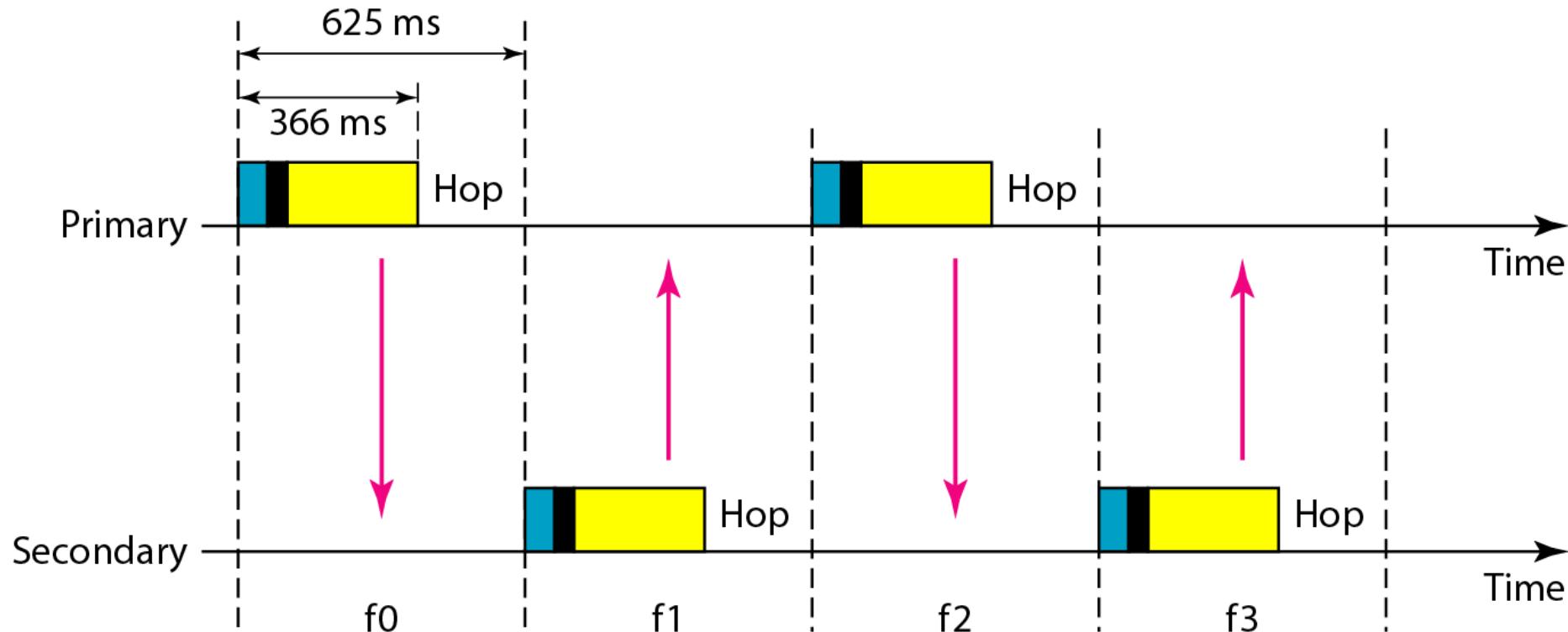


Figure 14.23 *Multiple-secondary communication*

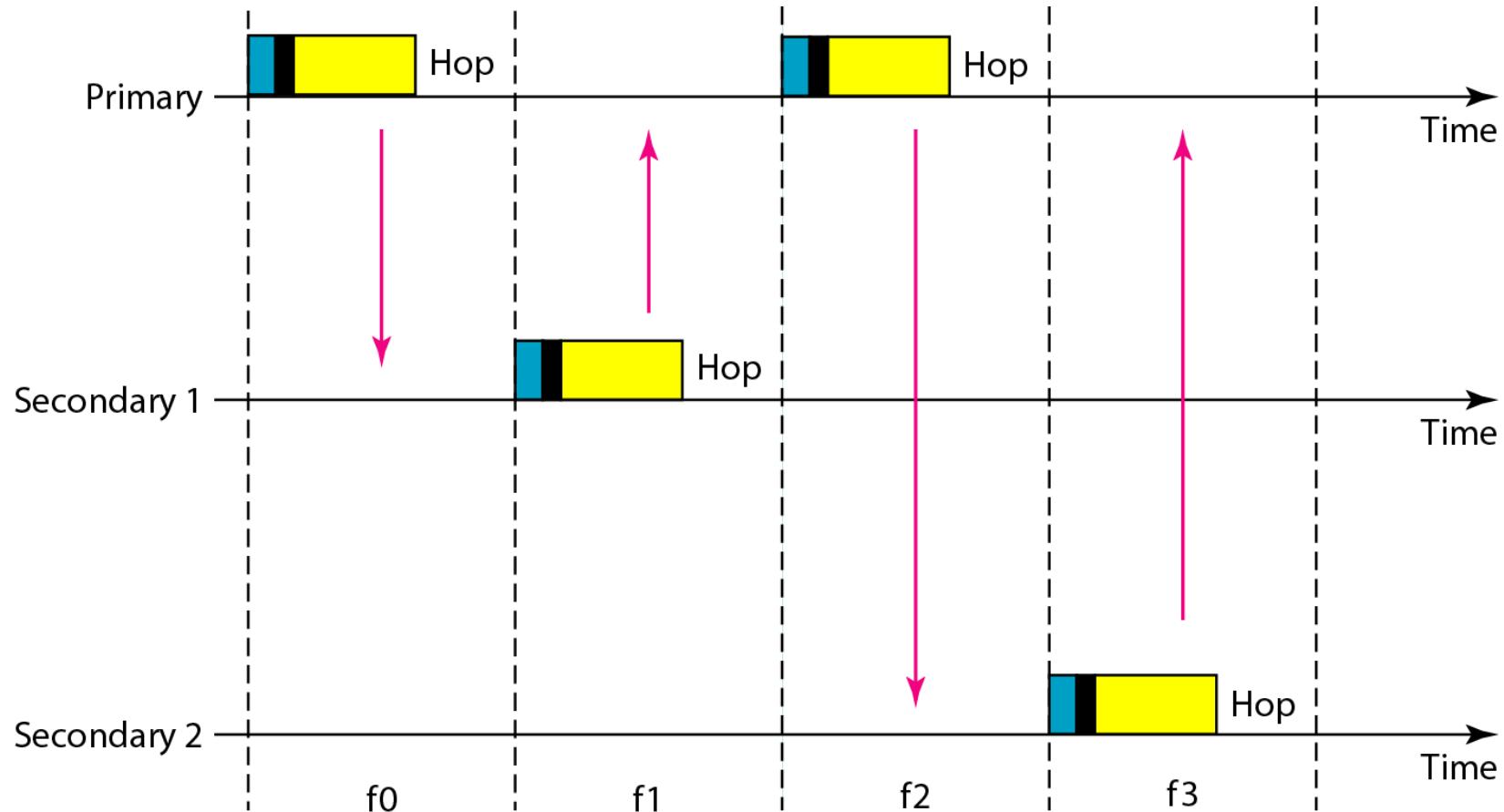


Figure 14.24 *Frame format types*

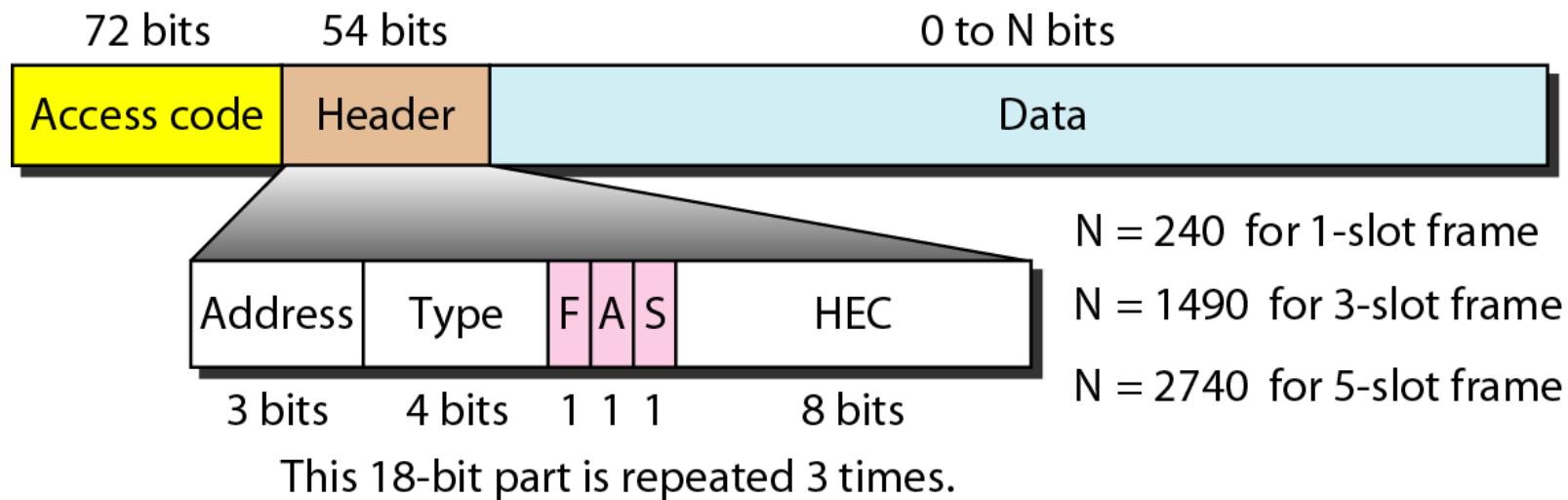


Figure 14.25 L2CAP data packet format



Chapter 15

Connecting LANs, Backbone Networks, and Virtual LANs

15-1 CONNECTING DEVICES

In this section, we divide connecting devices into five different categories based on the layer in which they operate in a network.

Topics discussed in this section:

Passive Hubs

Active Hubs

Bridges

Two-Layer Switches

Routers

Three-Layer Switches

Gateways

Figure 15.1 *Five categories of connecting devices*

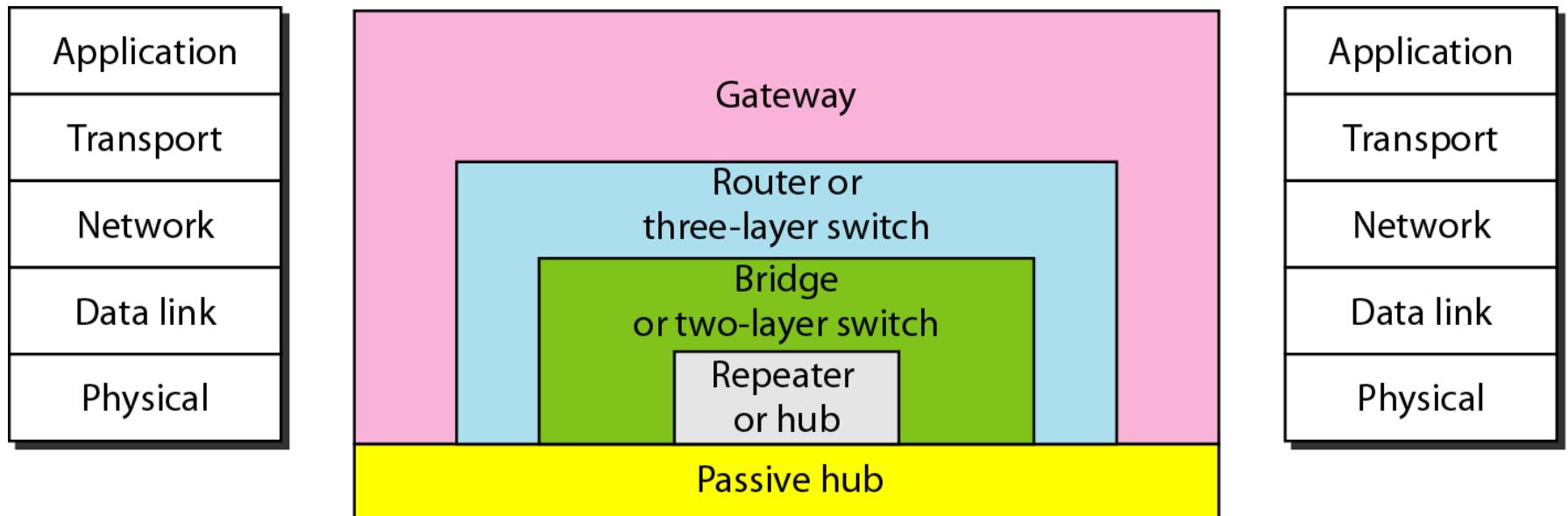
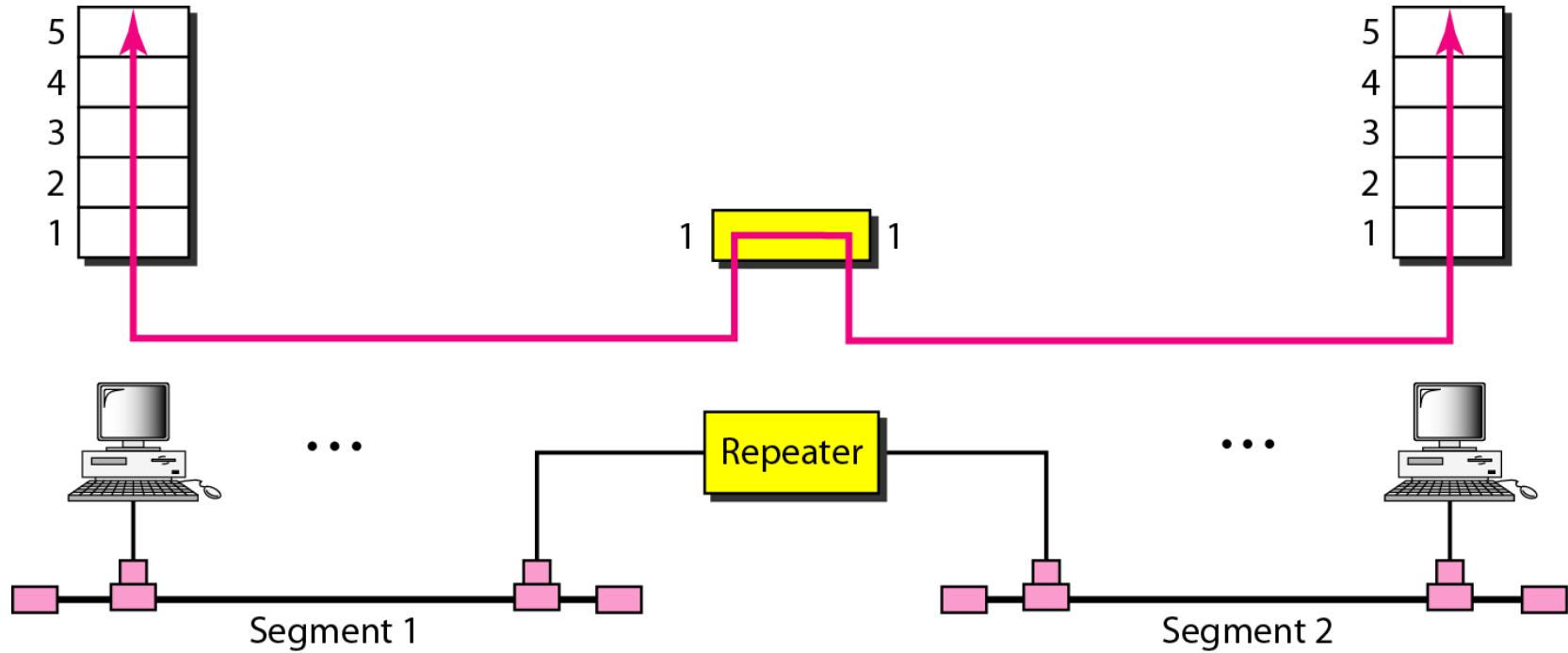
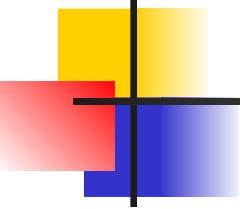


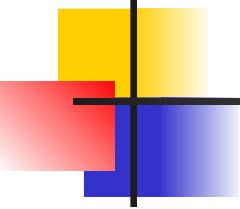
Figure 15.2 A *repeater connecting two segments of a LAN*





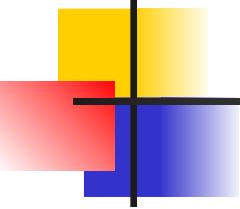
Note

A repeater connects segments of a LAN.



Note

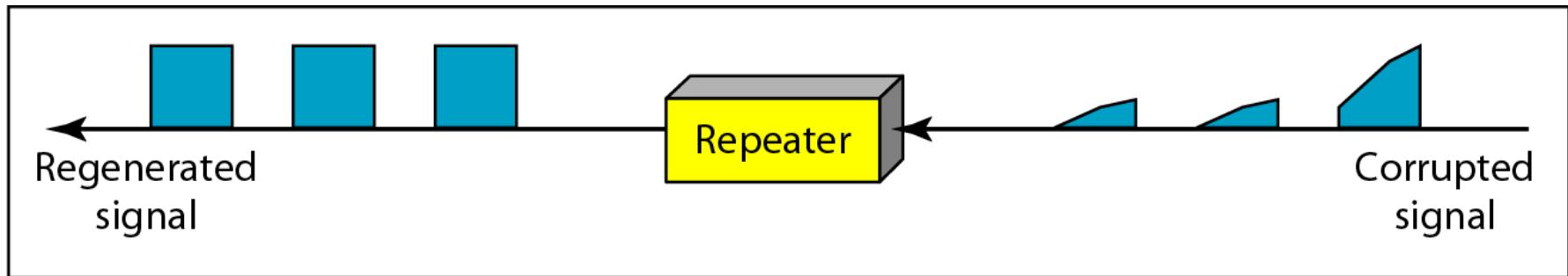
A repeater forwards every frame;
it has no filtering capability.



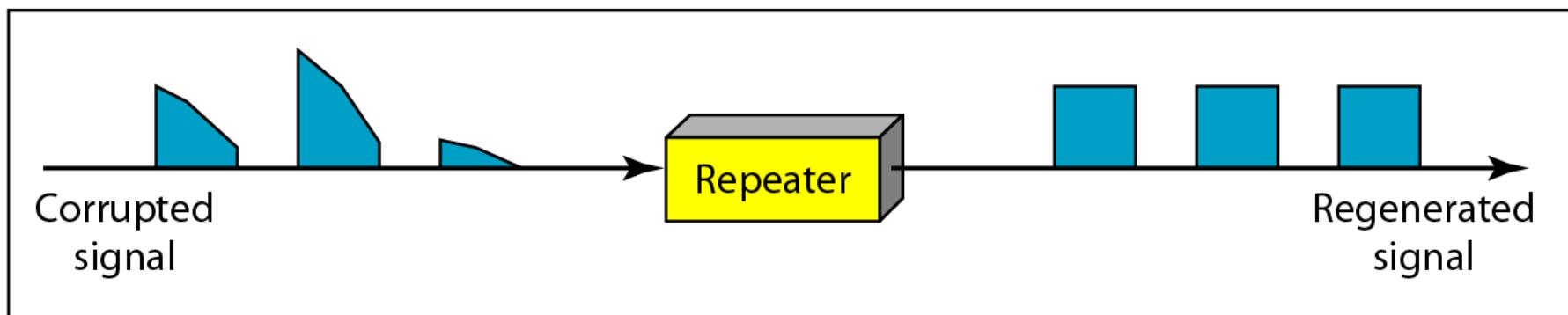
Note

**A repeater is a regenerator,
not an amplifier.**

Figure 15.3 Function of a repeater

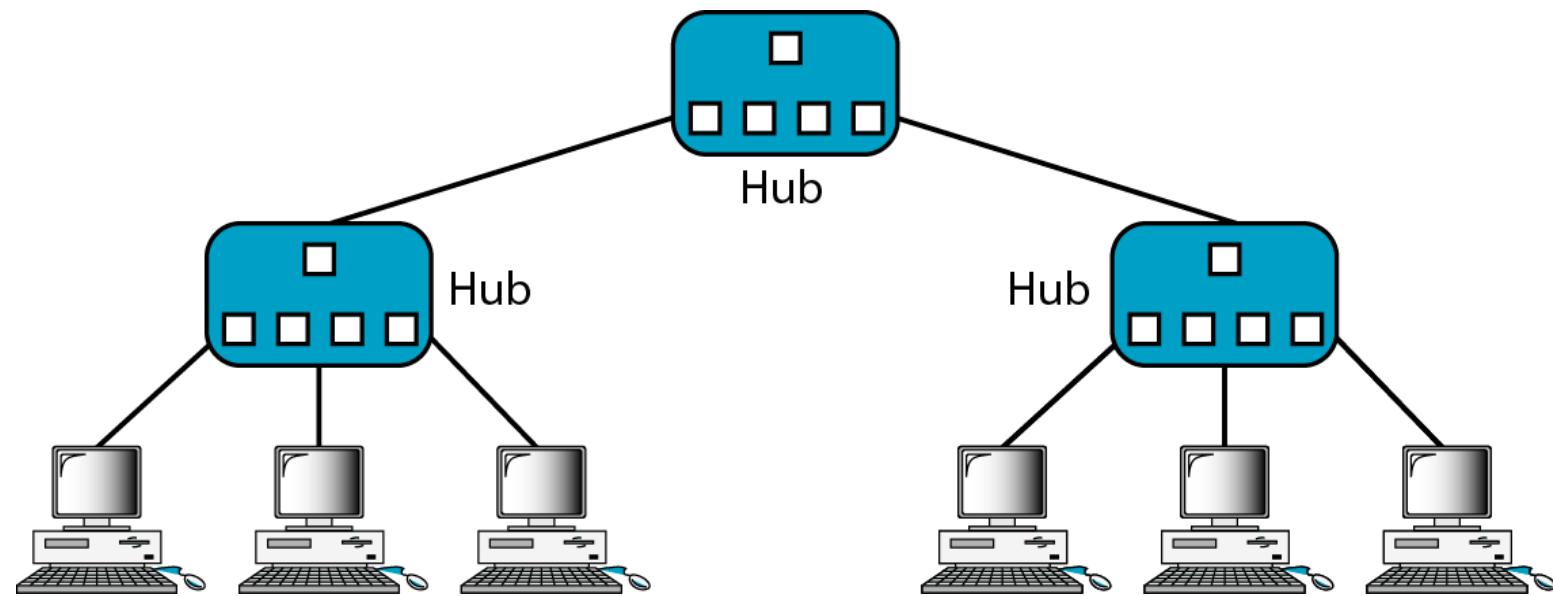


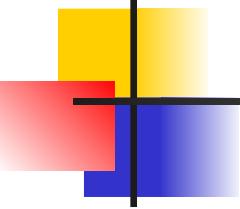
a. Right-to-left transmission.



b. Left-to-right transmission.

Figure 15.4 A *hierarchy of hubs*

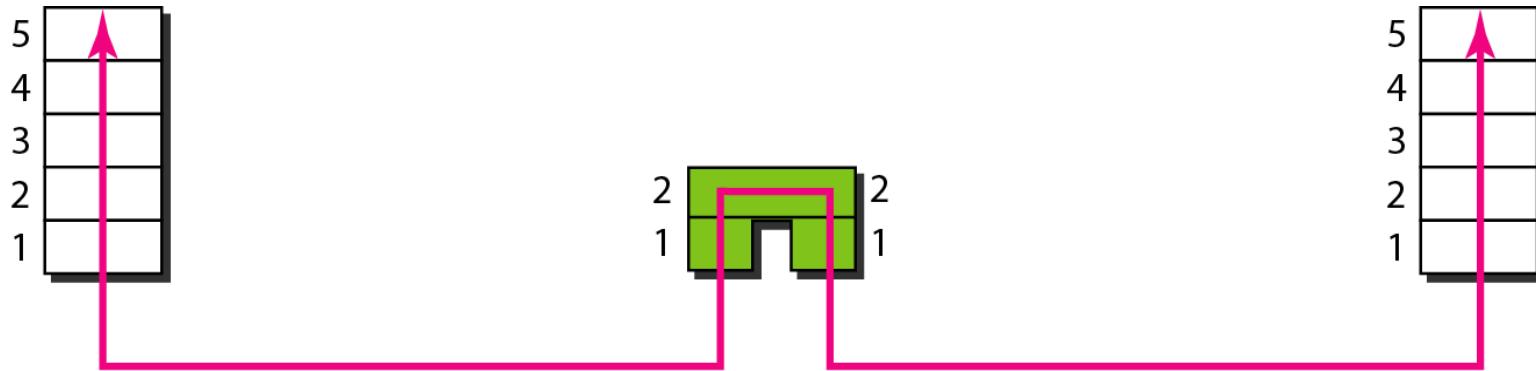




Note

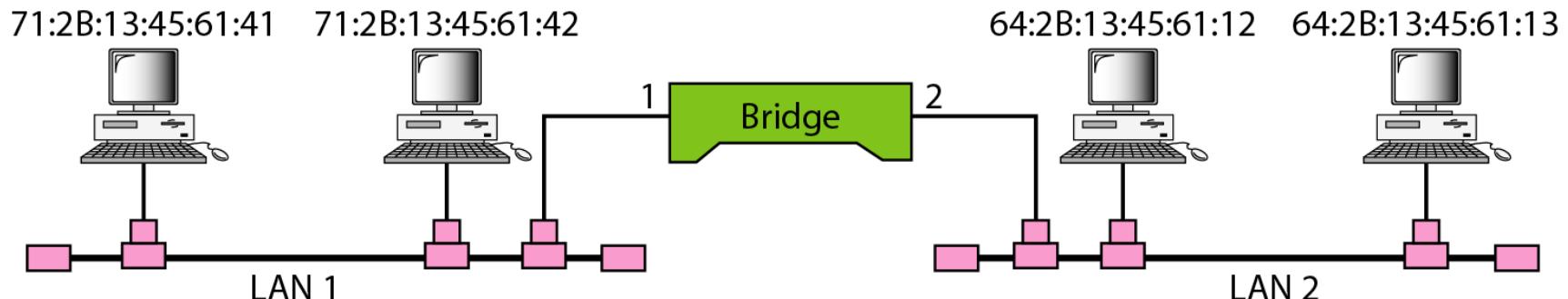
A bridge has a table used in filtering decisions.

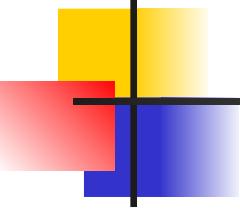
Figure 15.5 A bridge connecting two LANs



Address	Port
71:2B:13:45:61:41	1
71:2B:13:45:61:42	1
64:2B:13:45:61:12	2
64:2B:13:45:61:13	2

Bridge Table

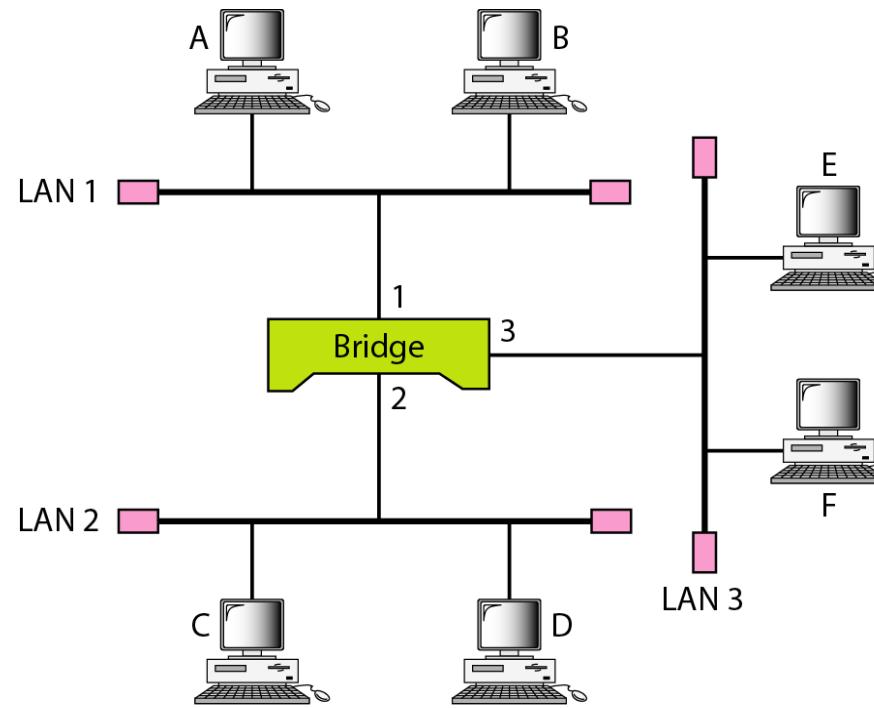




Note

**A bridge does not change the physical
(MAC) addresses in a frame.**

Figure 15.6 A learning bridge and the process of learning



Address	Port

a. Original

Address	Port
A	1

b. After A sends a frame to D

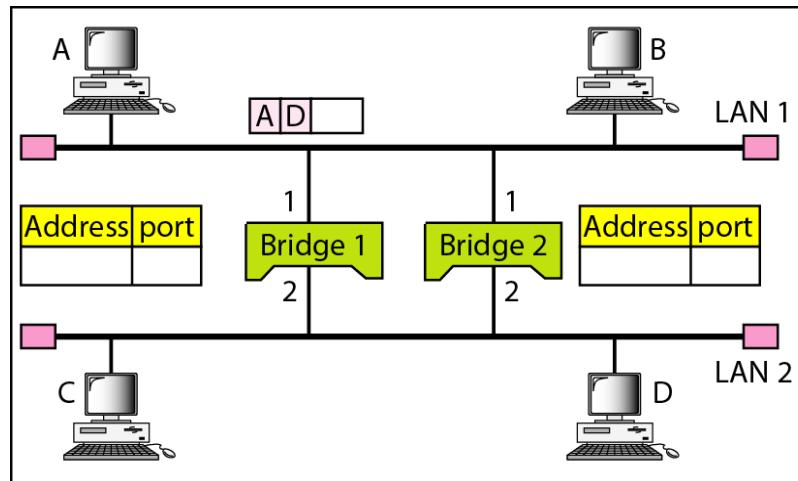
Address	Port
A	1
E	3

c. After E sends a frame to A

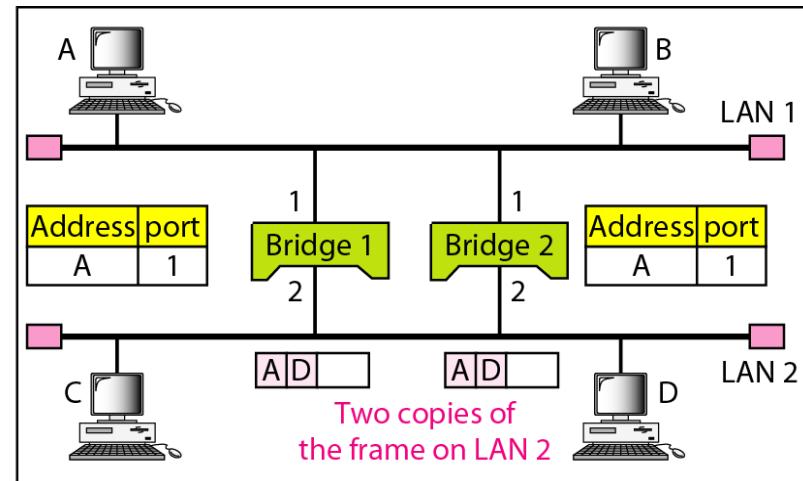
Address	Port
A	1
E	3
B	1

d. After B sends a frame to C

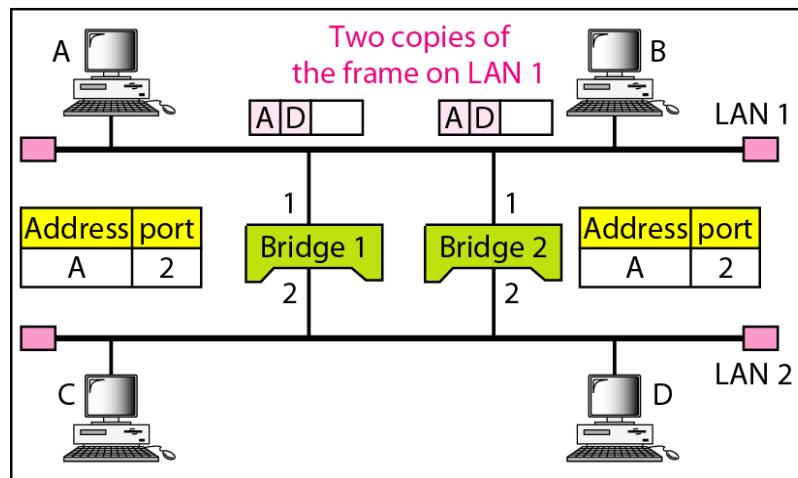
Figure 15.7 Loop problem in a learning bridge



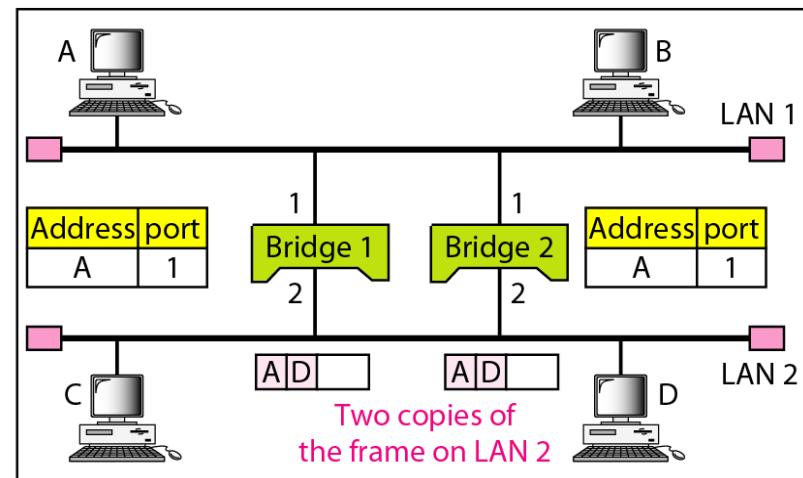
a. Station A sends a frame to station D



b. Both bridges forward the frame

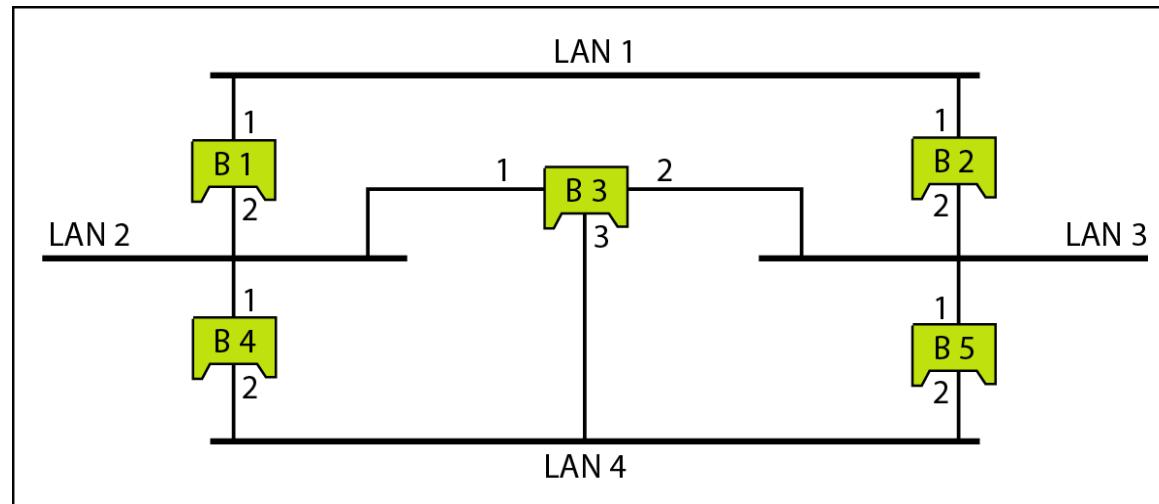


c. Both bridges forward the frame

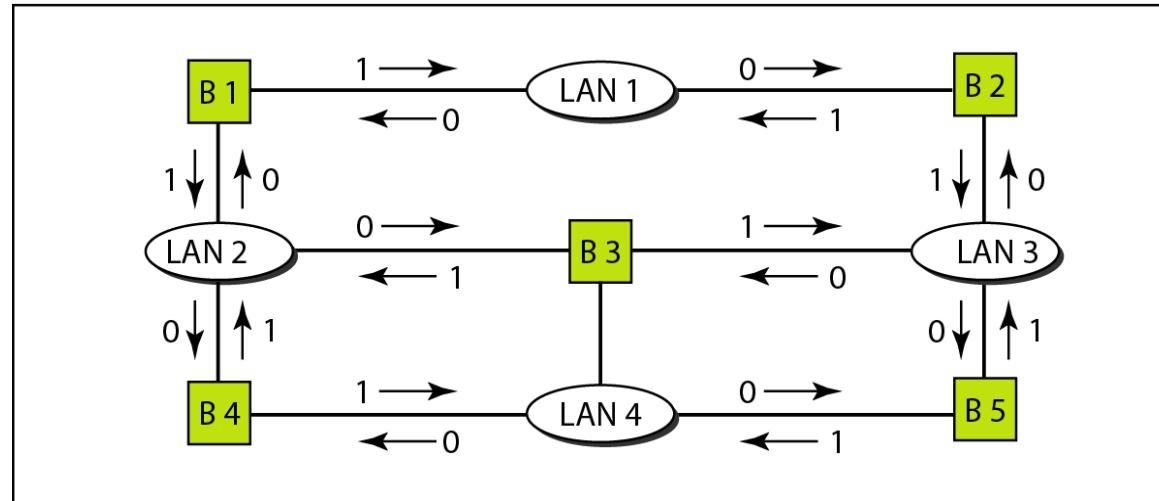


d. Both bridges forward the frame

Figure 15.8 A system of connected LANs and its graph representation



a. Actual system



b. Graph representation with cost assigned to each arc

Figure 15.9 *Finding the shortest paths and the spanning tree in a system of bridges*

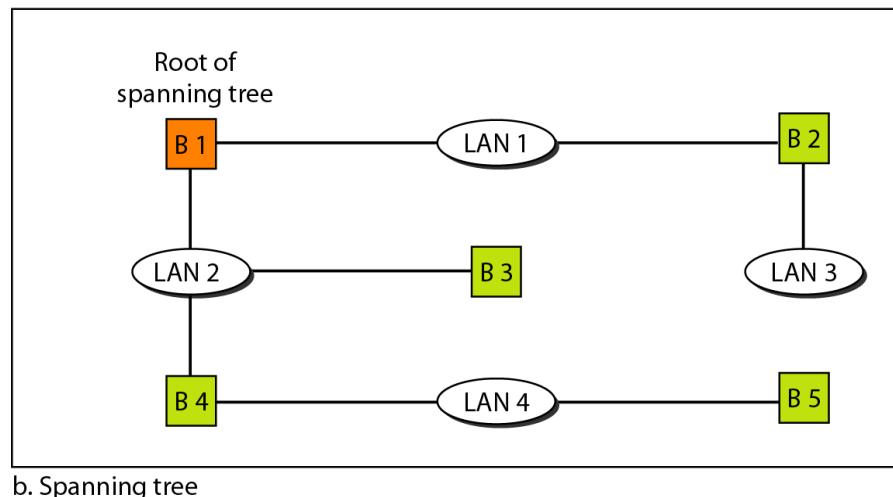
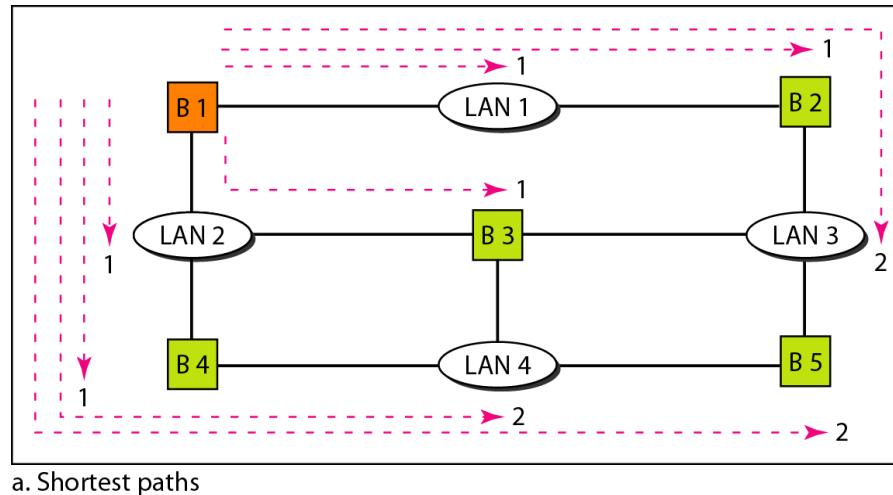
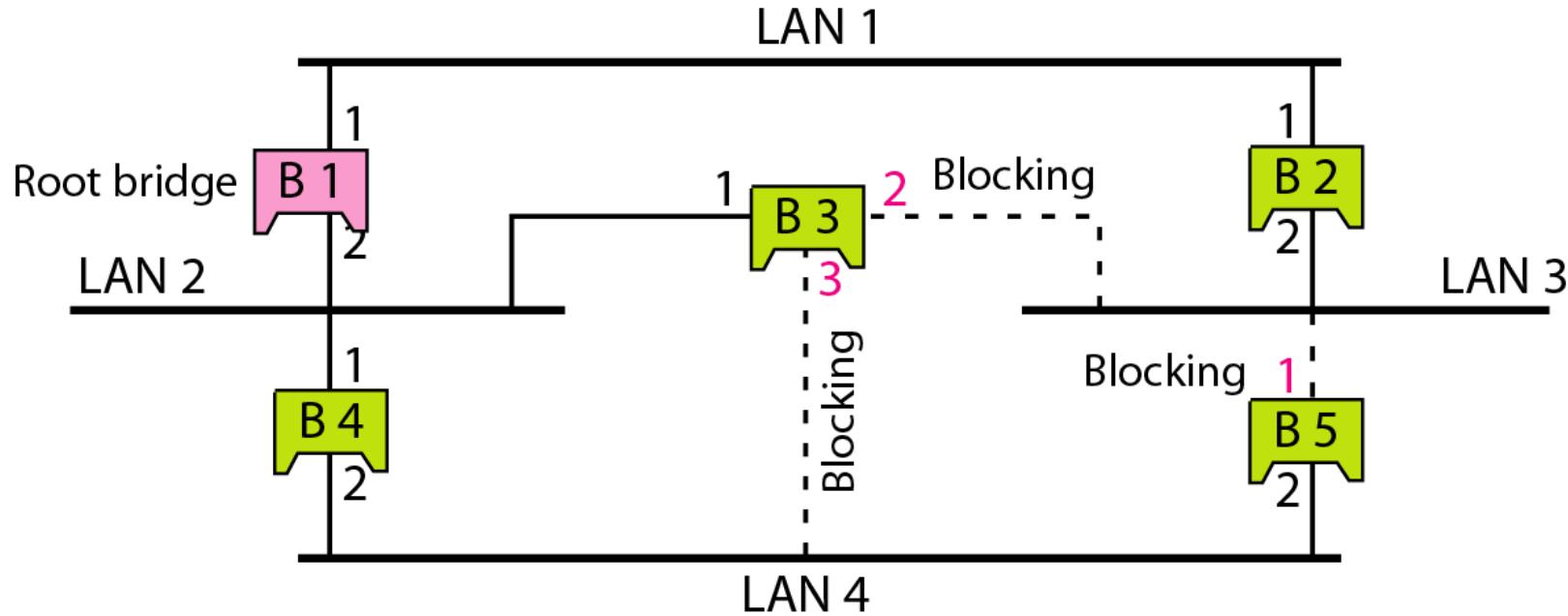
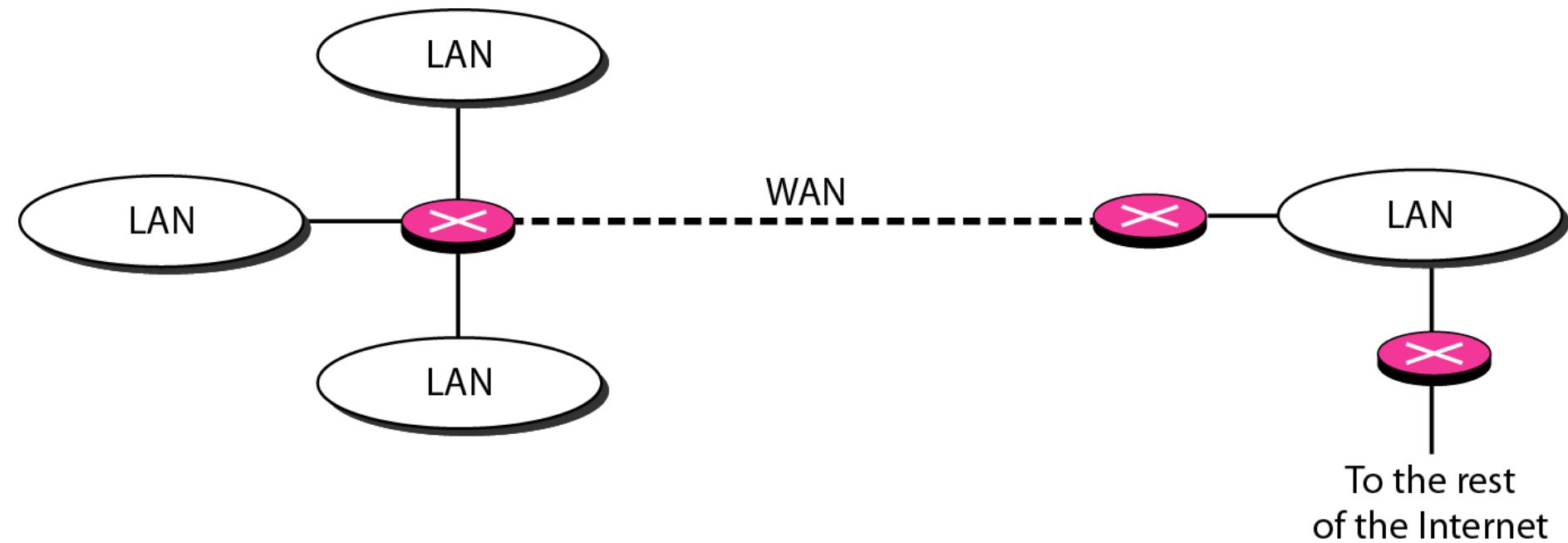


Figure 15.10 Forwarding and blocking ports after using spanning tree algorithm



Ports 2 and 3 of bridge B3 are blocking ports (no frame is sent out of these ports). Port 1 of bridge B5 is also a blocking port (no frame is sent out of this port).

Figure 15.11 *Routers connecting independent LANs and WANs*



15-2 BACKBONE NETWORKS

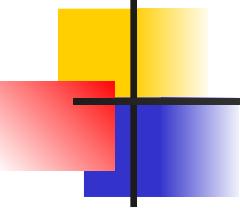
A *backbone network* allows several LANs to be connected. In a *backbone network*, no station is directly connected to the backbone; the stations are part of a LAN, and the backbone connects the LANs.

Topics discussed in this section:

Bus Backbone

Star Backbone

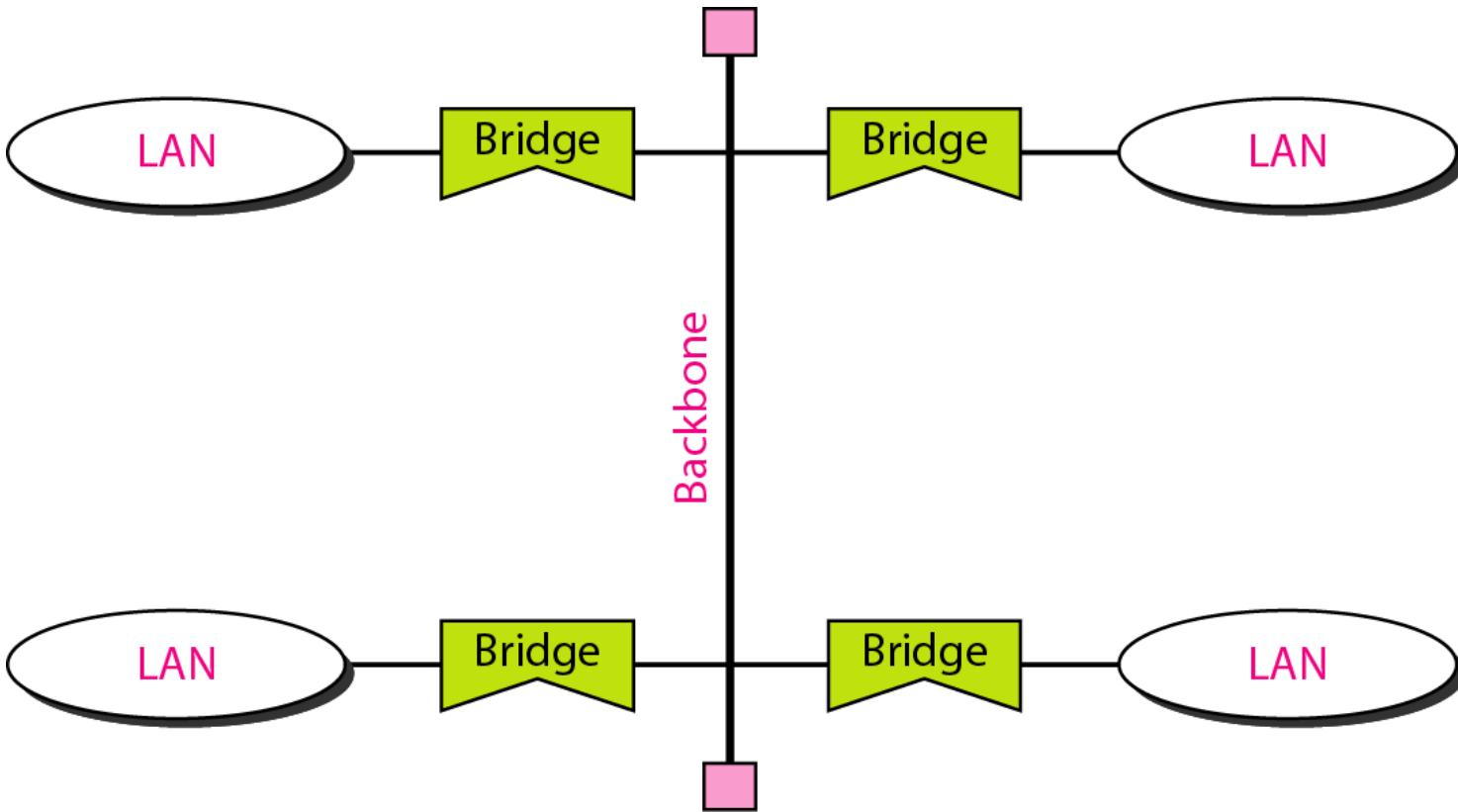
Connecting Remote LANs

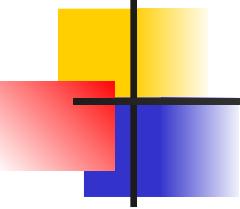


Note

**In a bus backbone, the topology
of the backbone is a bus.**

Figure 15.12 Bus backbone





Note

In a star backbone, the topology of the backbone is a star; the backbone is just one switch.

Figure 15.13 *Star backbone*

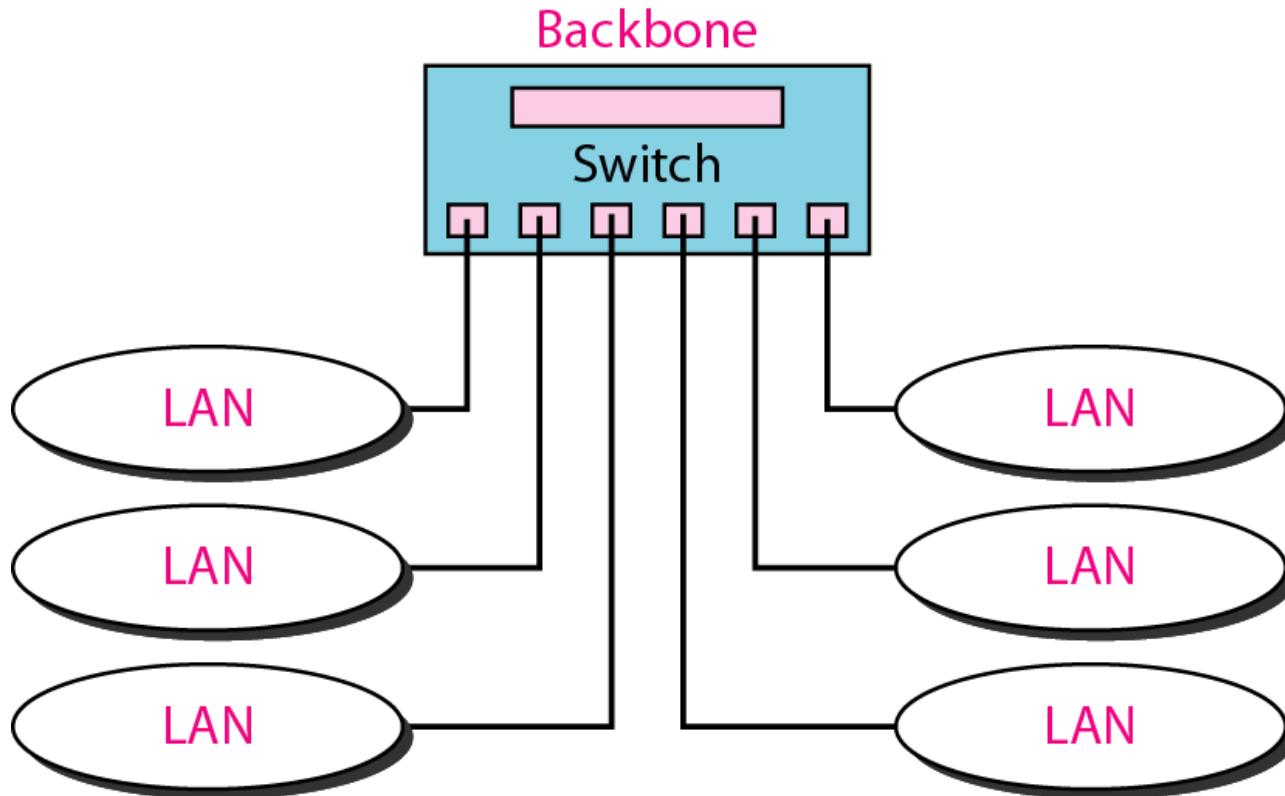
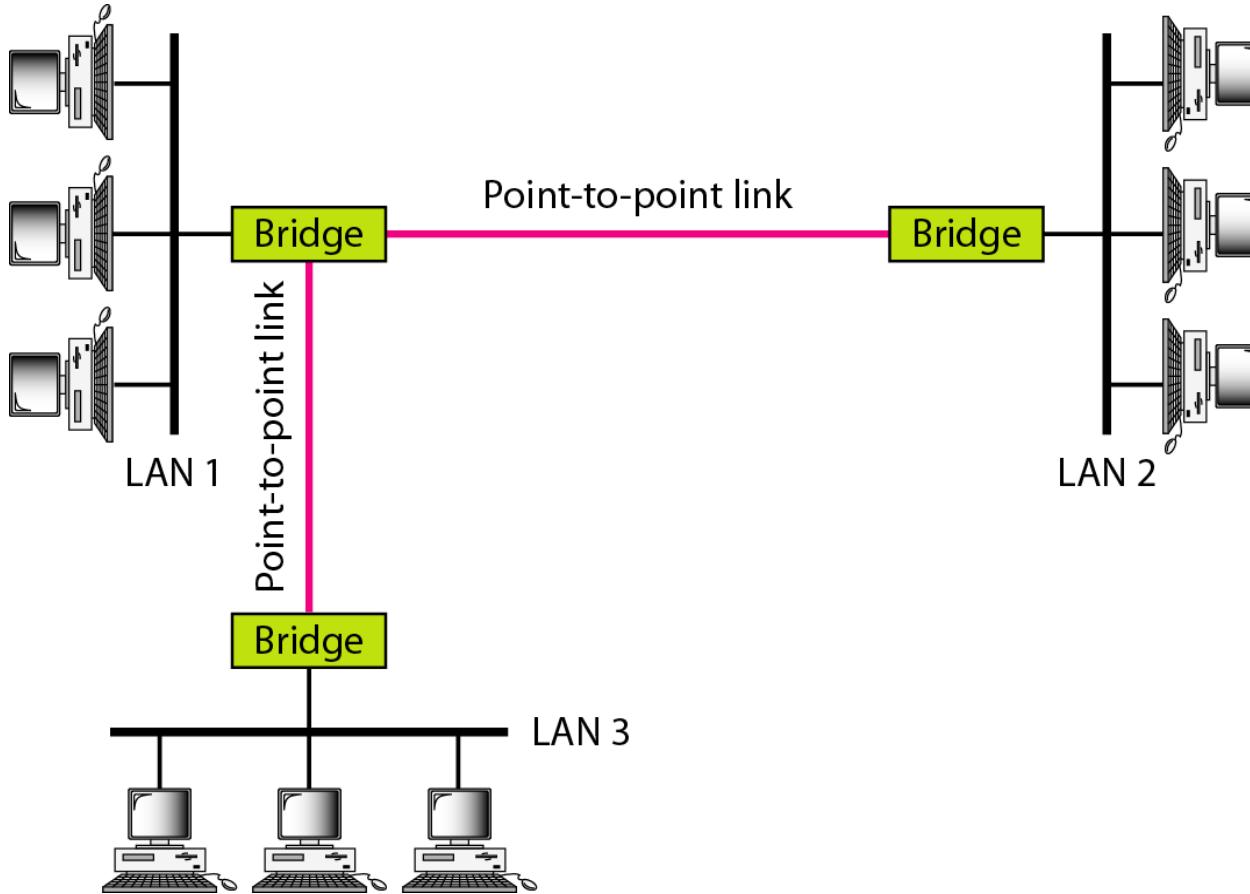
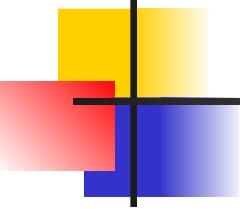


Figure 15.14 Connecting remote LANs with bridges





Note

A point-to-point link acts as a LAN in a remote backbone connected by remote bridges.

15-3 VIRTUAL LANs

*We can roughly define a **virtual local area network** (VLAN) as a local area network configured by software, not by physical wiring.*

Topics discussed in this section:

Membership

Configuration

Communication between Switches

IEEE Standard

Advantages

Figure 15.15 A switch connecting three LANs

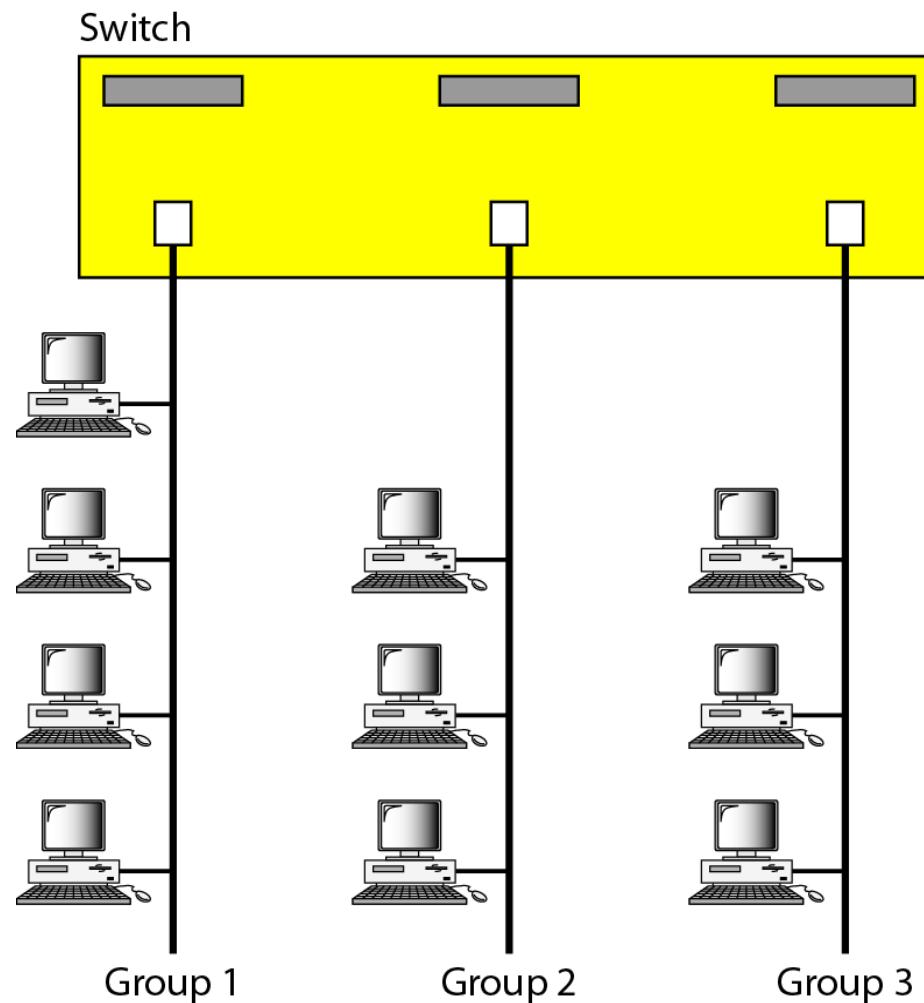


Figure 15.16 A switch using VLAN software

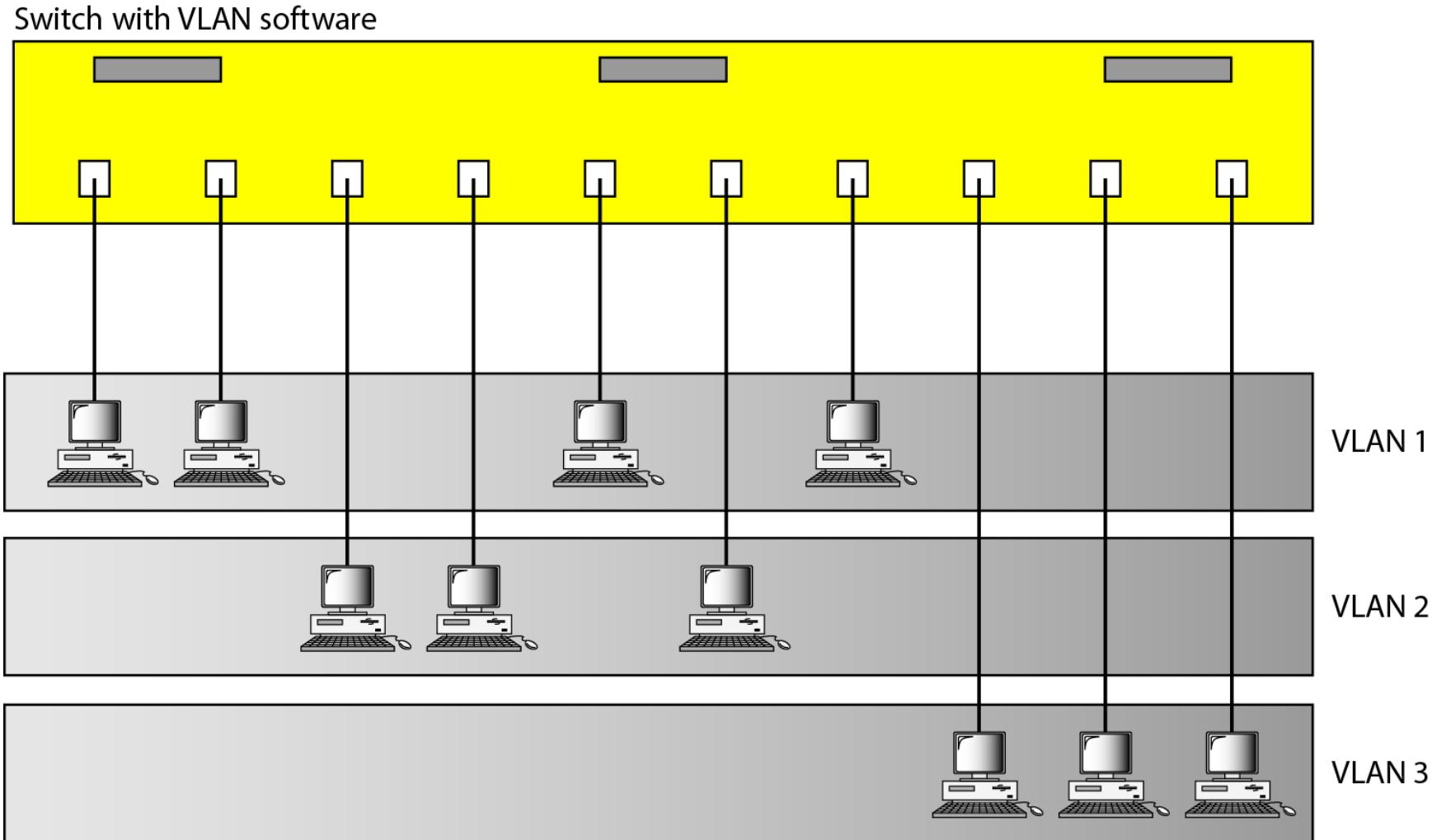
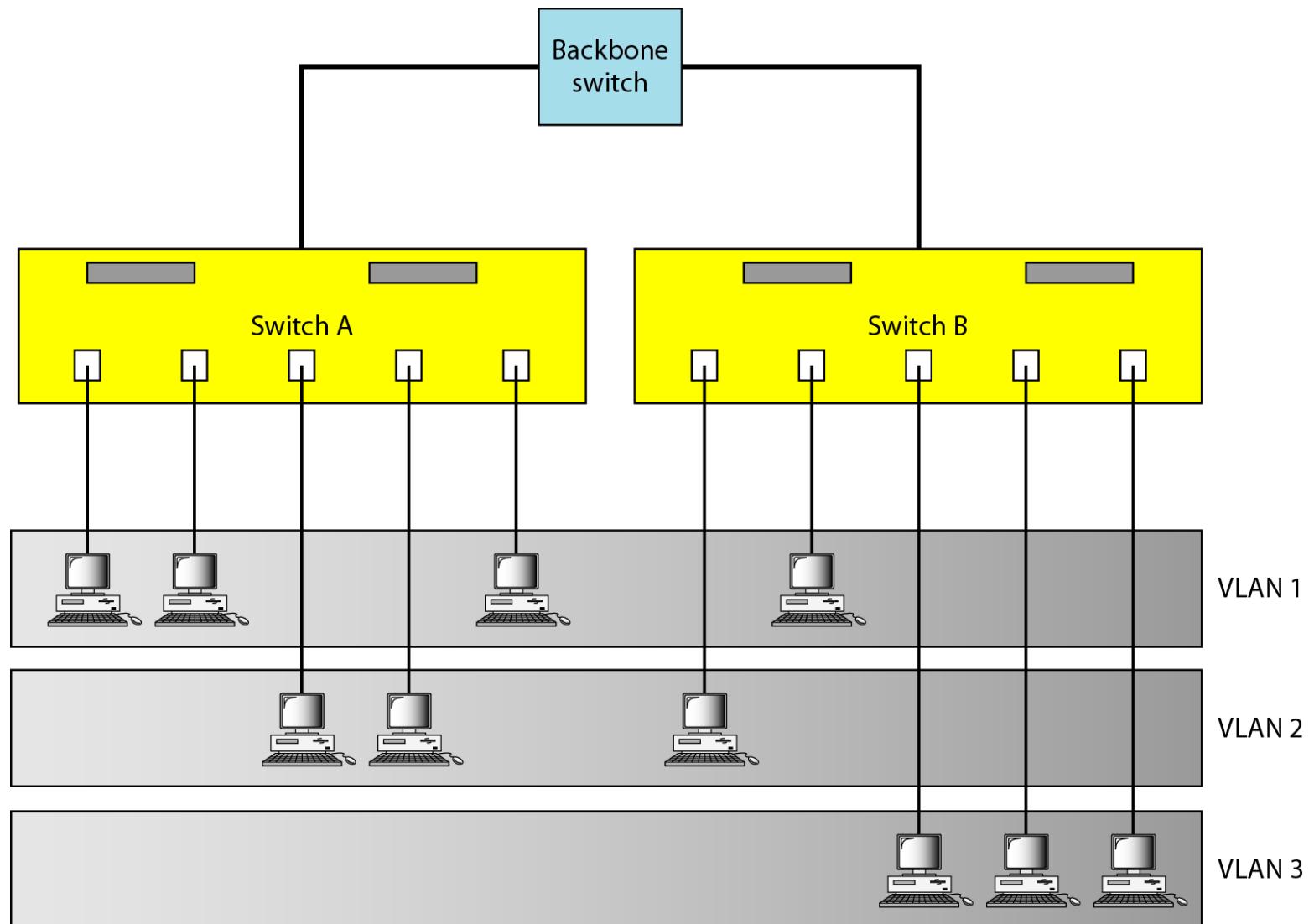
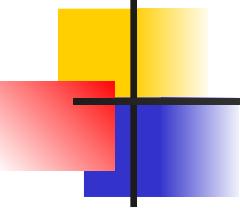


Figure 15.17 Two switches in a backbone using VLAN software





Note

VLANs create broadcast domains.
