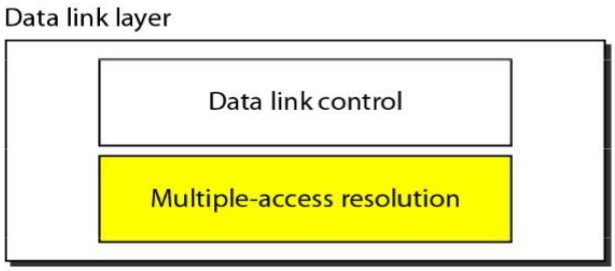
UNIT- II

MULTIPLE ACCESS PROTOCOLS

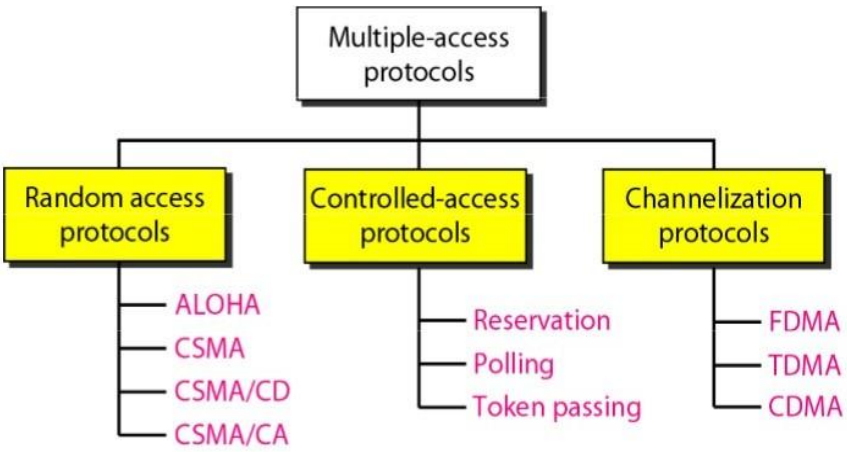
Data link layer is divided into two sub layers:

1. **Logical link control (LLC) layer**: The upper sub layer is responsible for data link control i.e. for flow and error control.
2. **Media access control (MAC) layer**: The lower sub layer is responsible for resolving access to the shared media.

If the channel is dedicated, we do not need the lower sub layer. Figure 12.1 shows these two sub layers in the data link layer.



When nodes or stations are connected and use a common link, called a multipoint or broadcast link, we need a multiple-access protocol to coordinate access to the link. The problem of controlling the access to the medium is similar to the rules of speaking in an assembly. The procedures guarantee that the right to speak is upheld and ensure that two people do not speak at the same time, do not interrupt each other, and do not monopolize the discussion, and so on. The situation is similar for multipoint networks. Many formal protocols have been devised to handle access to a shared link. We categorize them intothree groups.



2. RANDOM ACCESS PROTOCOLS

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. This decision depends on the state of the medium (idle or busy). In other words, each station can transmit when it desires on the condition that it follows the predefined procedure, including the testing of the state of the medium. Two features give this method its name.

1. First, there is no scheduled time for a station to transmit. Transmission is random among the stations. That is why these methods are called random access.
2. Second, no rules specify which station should send next. Stations compete with one another to access the medium. That is why these methods are also called contention methods.

In a random access method, each station has the right to the medium without being controlled by any other station. However, if more than one station tries to send, there is an access conflict-collision-and the frames will be either destroyed or modified. To avoid access conflict or to resolve it when it happens, each station follows a procedure that answers the following questions:

1. *When* can the station access the medium?
2. *What* can the station do if the medium is busy?
3. *How* can the station determine the success or failure of the transmission?
4. What can the station do if there is an access conflict?

ALOHA

ALOHA, the earliest random access method was developed at the University of Hawaii in early 1970. It was designed for a radio (wireless) LAN, but it can be used on any shared medium. It is obvious that there are potential collisions in this arrangement. The medium is shared between the stations. When a station sends data, another station may attempt to do so at the same time. The data from the two stations collide and become garbled. There are two variations of ALOHA protocol:

Pure ALOHA

The original ALOHA protocol is called *pure ALOHA*. This is a simple, but elegant protocol. The idea is that each station sends a frame whenever it has a frame to send. However, since there is only one channel to share, there is the possibility of collision between frames from different stations. Figure 12.3 shows an example of frame collisions in pure ALOHA.

There are four stations (unrealistic assumption) that contend with one another for access to the shared channel. The figure shows that each station sends two frames; there are a total of eight frames on the shared medium. Some of these frames collide because multiple frames are in contention for the shared channel. Figure 12.3 shows that only two frames survive: frame 1.1 from station 1 and frame 3.2 from station 3. We need to mention that even if one bit of a frame coexists on the channel with one bit from another frame, there is a collision and both will be destroyed. It is obvious that we need to resend the frames that have been destroyed during transmission.

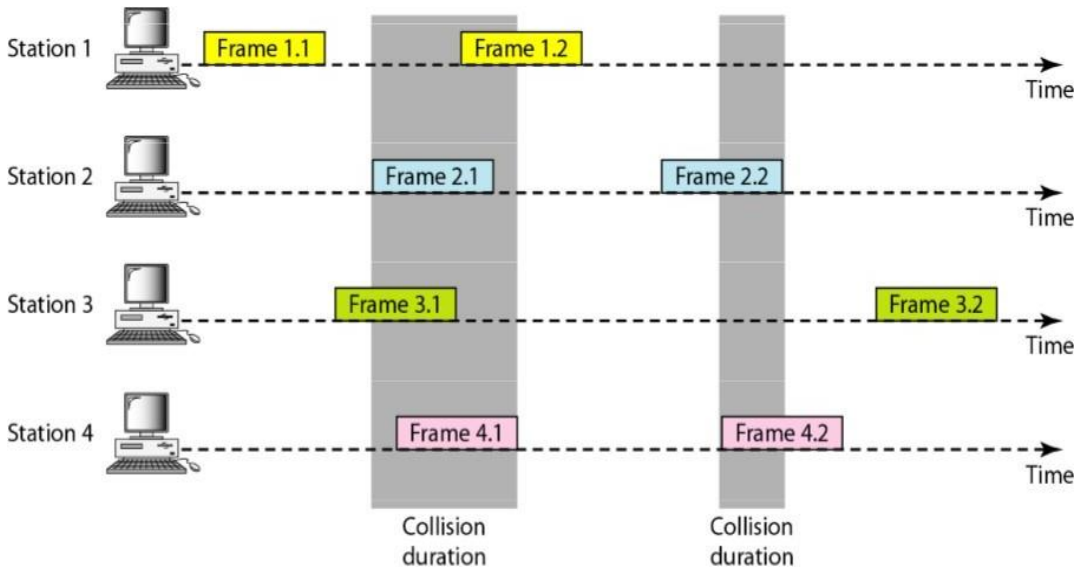


Figure 12.3 Frames in a pure ALOHA network

The pure ALOHA protocol relies on acknowledgments from the receiver. When a station sends a frame, it expects the receiver to send an acknowledgment. If the acknowledgment does not arrive after a time-out period, the station assumes that the frame (or the acknowledgment) has been destroyed and resends the frame. A collision involves two or more stations. If all these stations try to resend their frames after the time-out, the frames will collide again.

Pure ALOHA dictates that when the time-out period passes, each station waits a random amount of time before resending its frame. The randomness will help avoid more collisions. We call this time the back-off time T_B. Pure ALOHA has a second method to prevent congesting the channel with retransmitted frames. After a maximum number of retransmission attempts K_{max} a station must give up and try later. Figure 12.4 shows the procedure for pure ALOHA based on the above strategy.

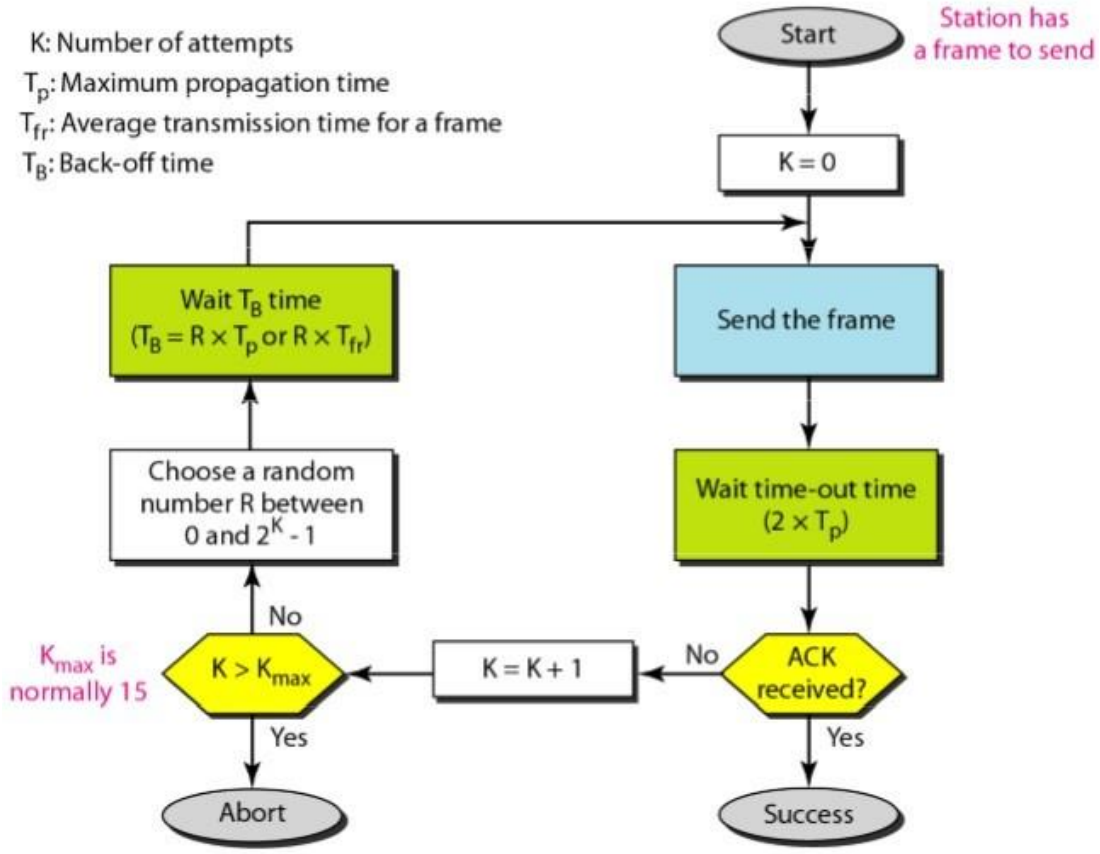


Figure 12.4 Procedure for pure ALOHA protocol

The time-out period is equal to the maximum possible round-trip propagation delay, which is twice the amount of time required to send a frame between the two most widely separated stations (2 x Tp) The back-off time T_B is a random value that normally depends on K (the number of attempted unsuccessful transmissions). One common formula for calculating T_B is the *binary exponential back-off*. In this method, for each retransmission, a multiplier in the range 0 to 2^K - 1 is randomly chosen and multiplied by Tp (maximum propagation time) to find T_B. Note that in this procedure, the range of the random numbers increases after each collision. The value of K_{max} is usually chosen as 15.

Slotted ALOHA

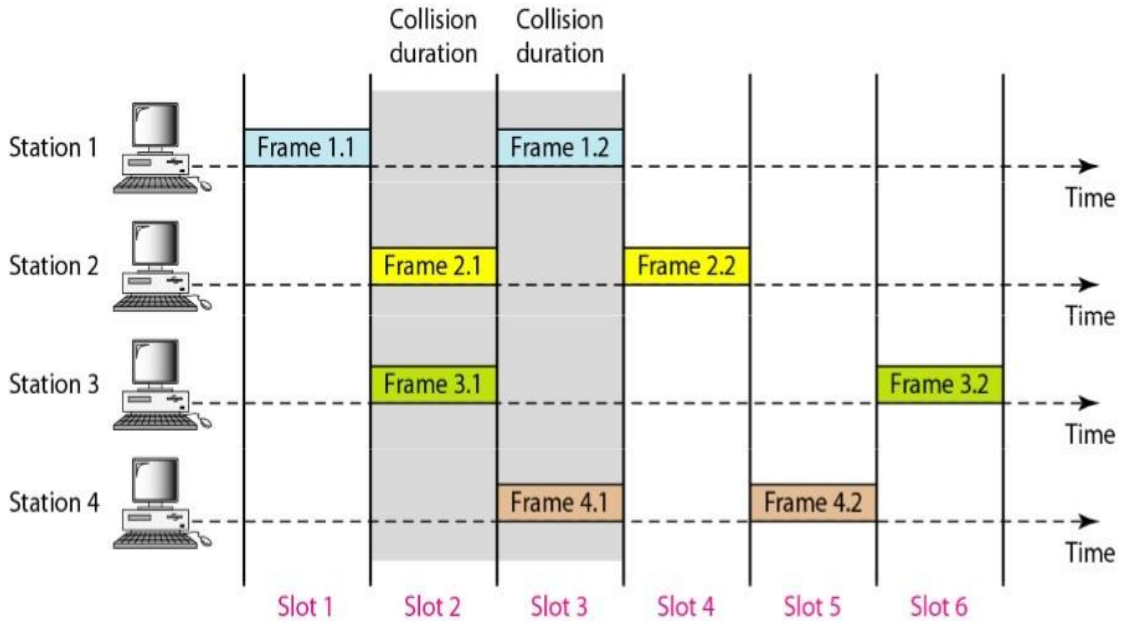
Slotted ALOHA was invented to improve the efficiency of pure ALOHA. In slotted ALOHA we divide the time into slots of T_{fr} s(time to send each fixed size frame) and force the station to send only at the beginning of the time slot.

Because a station is allowed to send only at the beginning of the synchronized time slot, if a station misses this moment, it must wait until the beginning of the next time slot. This means that the station which started at the beginning of this slot has already finished sending its frame. Of course, there is still the possibility of collision if two stations try to send at the beginning of the same time slot.

Vulnerable time

Vulnerable time for slotted ALOHA is one-half that of pure ALOHA i.e.

Slotted ALOHA vulnerable time $= T_{fr}$



Frames in a slotted ALOHA network

CARRIER SENSE MULTIPLE ACCESS (CSMA)

To minimize the chance of collision and, therefore, increase the performance, the CSMA method was developed. The chance of collision can be reduced if a station senses the medium before trying to use it. Carrier sense multiple access (CSMA) requires that each station first listen to the medium (or check the state of the medium) before sending. In other words, CSMA is based on the principle "sense before transmit" or "listen before talk." CSMA can reduce the possibility of collision, but it cannot eliminate it.

Vulnerable Time

The vulnerable time for CSMA is the propagation time Tp. This is the time needed for a signal to propagate from one end of the medium to the other. When a station sends a frame, and any other station tries to send a frame during this time, a collision will result. But if the first bit of the frame reaches the end of the medium, every station will already have heard the bit and will refrain from sending.

Persistence Methods

Three methods have been devised:

1. 1-persistent method,
2. Non-persistent method, and
3. P-persistent method.

1-Persistent Method

The 1-persistent method is simple and straightforward. In this method, after the station finds the line idle, it sends its frame immediately (with probability 1). This method has the highest chance of collision because two or more stations may find the line idle and send their frames immediately. Ethernet, a LAN standard uses this method.

Non-persistent Method

In the non-persistent method, a station that has a frame to send senses the line. If the line is idle, it sends immediately. If the line is not idle, it waits a random amount of time and then senses the line again. The non-persistent approach reduces the chance of collision because it is unlikely that two or more stations will wait the same amount of time and retry to send simultaneously. However, this method reduces the efficiency of the network because the medium remains idle when there may be stations with frames to send.

p-Persistent Method

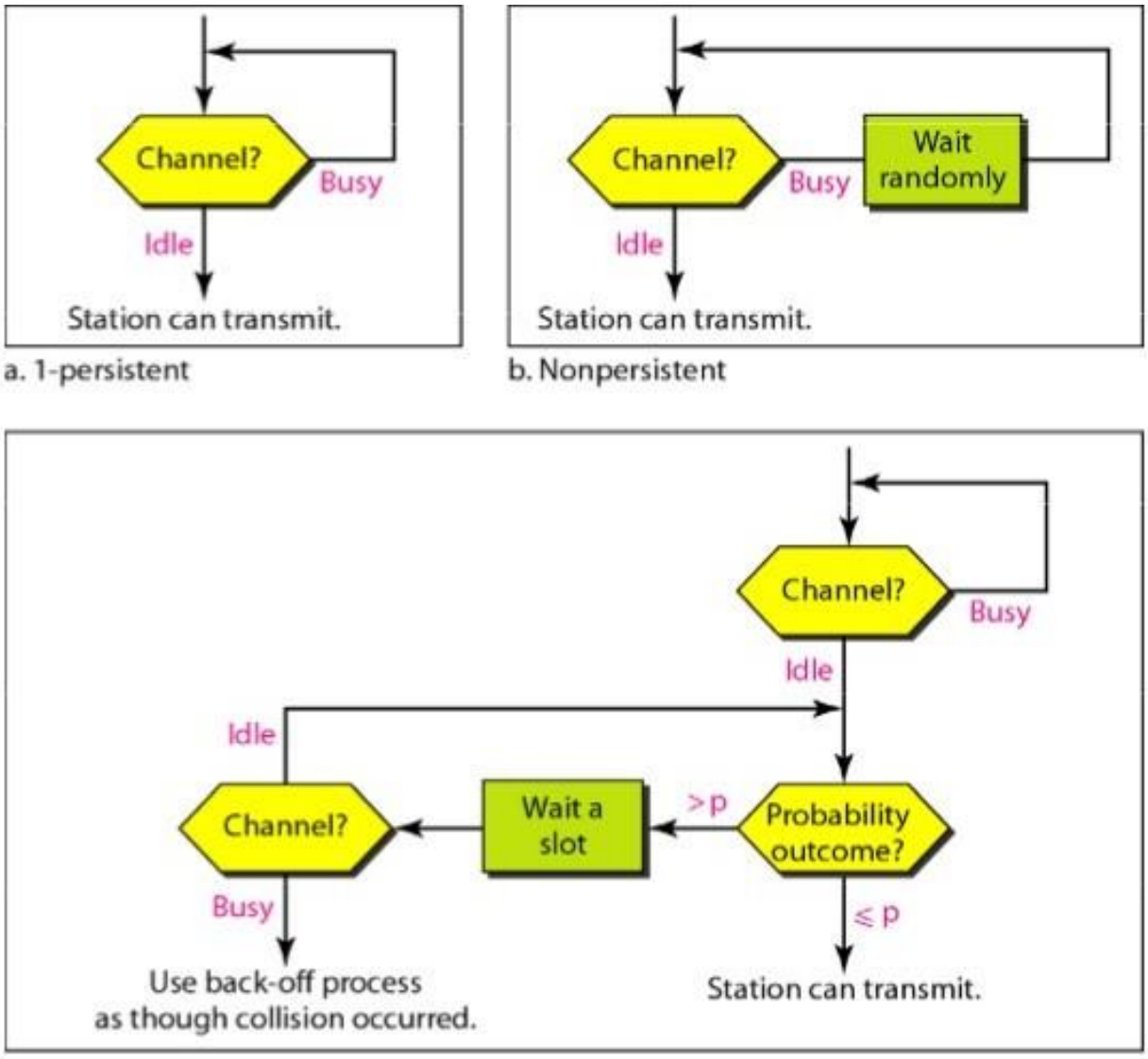
The p-persistent method is used if the channel has time slots with slot duration equal to or greater than the maximum propagation time. The p-persistent approach combines the advantages of the other two strategies. It reduces the chance of collision and improves efficiency.

In this method, after the station finds the line idle it follows these steps:

1. With probability p, the station sends its frame.
2. With probability q = 1 - p, the station waits for the beginning of the next time slot and checks the line again.

a) If the line is idle, it goes to step **(i)**.

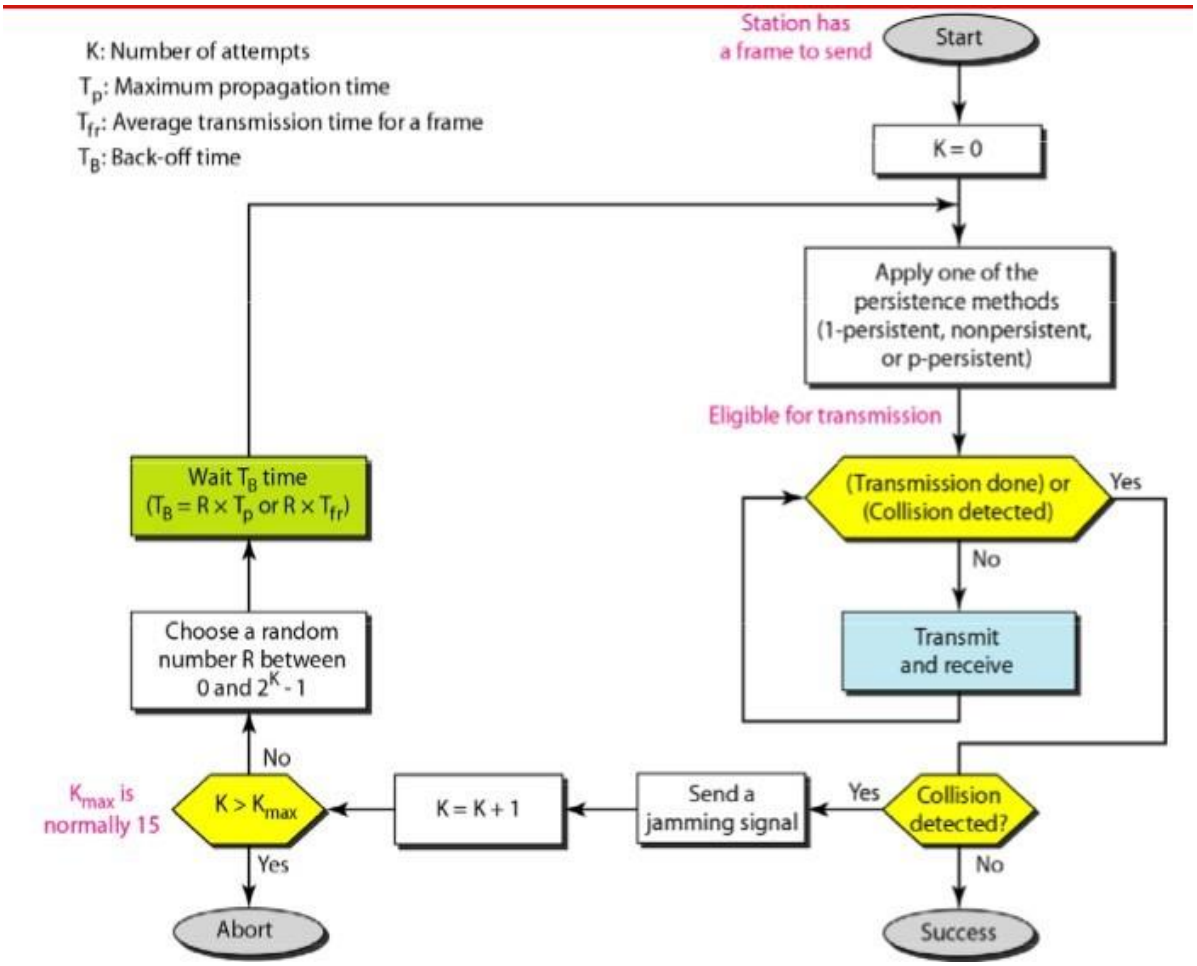
b) If the line is busy, it acts as though a collision has occurred and uses the backoff procedure.



CARRIER SENSE MULTIPLE ACCESS WITH COLLISION DETECTION (CSMA/CD)

The CSMA method does not specify the procedure following a collision. Carriers sense multiple access with collision detection (CSMA/CD) augments the algorithm to handle the collision.

1. In this method, a station monitors the medium after it sends a frame to see if the transmission was successful. If so, the station is finished.
2. If, however, there is a collision, the frame is sent again.



It is similar to the one for the ALOHA protocol, but there are differences:

(i). The first difference is the addition of the persistence process. We need to sense the channel before we start sending the frame by using one of the persistence processes we discussed previously (non-persistent, 1-persistent, or p-persistent).

(ii). The second difference is the frame transmission. In ALOHA, we first transmit the entire frame and then wait for an acknowledgment. In CSMA/CD, transmission and collision detection is a continuous process. We do not send the entire frame and then look for a collision. The station transmits and receives continuously and simultaneously (using two different ports). We constantly monitor in order to detect one of two conditions: either transmission is finished or a collision is detected. Either event stops transmission. When we come out of the loop, if a collision has not been detected, it means that transmission is complete; the entire frame is transmitted. Otherwise, a collision has occurred.

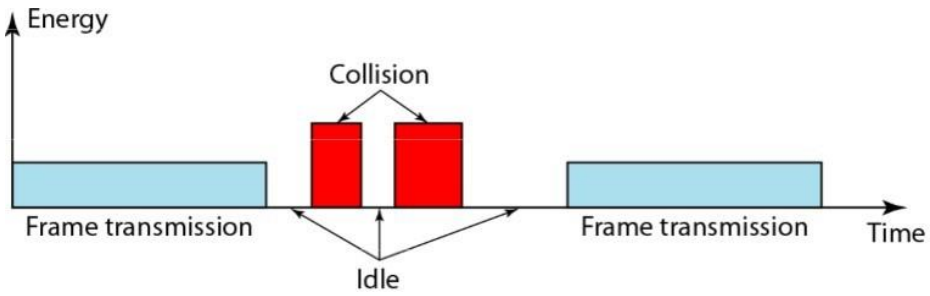
(iii). The third difference is the sending of a short jamming signal that enforces the collision in case other stations have not yet sensed the collision.

Energy Level

Energy level of a channel is used to monitor the current status of a channel. The level of energy in a channel can have three values: zero, normal, and abnormal.

1. At the zero level, the channel is idle.
2. At the normal level, a station has successfully captured the channel and is sending its frame.
3. At the abnormal level, there is a collision and the level of the energy is twice the normal level.

A station that has a frame to send or is sending a frame needs to monitor the energy level to determine if the channel is idle, busy, or in collision mode. Figure 12.8 shows the situation.



Energy level during transmission, idleness, or collision

Throughput

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

(i) The throughput for **pure ALOHA** is S =G x e^{-2G}. The maximum throughput **Smax =0.184** when G=(1/2).

(ii) The throughput for **slotted ALOHA** is S =: G x e^{-G}. The maximum throughput **Smax = 0.368** when G=1.

(iii) The throughput of **CSMA/CD** is greater than that of pure or slotted ALOHA. The maximum throughput occurs at a different value of G and is based on the persistence method and the value of p in the p-persistent approach.

(a) For **1-persistent** method the maximum throughput is around **50** percent when G =1.

(b) For **non-persistent** method, the maximum throughput can go up to **90** percent when G is between 3 and 8.

CONTROLLED ACCESS METHODS

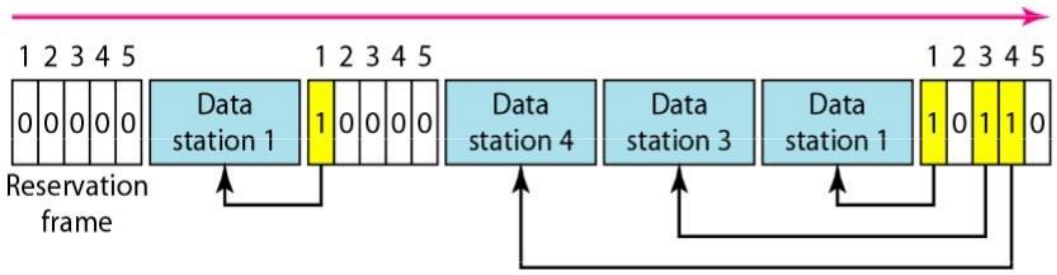
In controlled access, the stations consult one another to find which station has the right to send. A station cannot send unless it has been authorized by other stations. There are three popular controlled-access methods:

1. Reservation
2. Polling
3. Token passing

1. RESERVATION

In the reservation method, a station needs to make a reservation before sending data. Time is divided into intervals. In each interval, a reservation frame precedes the data frames sent in that interval.

If there are N stations in the system, there are exactly N reservation mini-slots in the reservation frame. Each mini-slot belongs to a station. When a station needs to send a data frame, it makes a reservation in its own mini-slot. The stations that have made reservations can send their data frames after the reservation frame. Figure shows a situation with five stations and a five mini-slot reservation frame. In the first interval, only stations 1, 3, and 4 have made reservations. In the second interval, only station 1 has made a reservation.

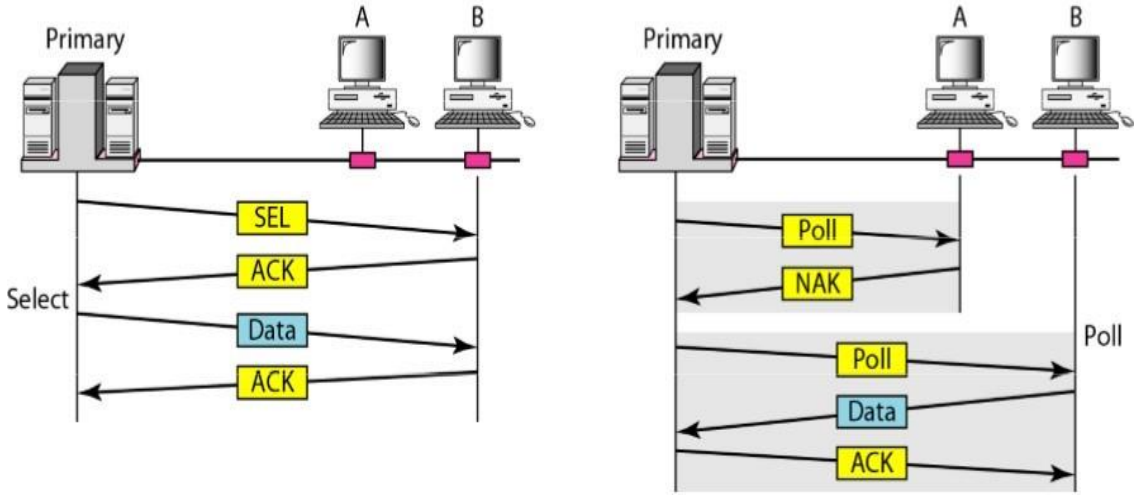


Reservation access method

2. POLLING

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

If the primary wants to receive data, it asks the secondaries if they have anything to send; this is called **poll function**. If the primary wants to send data, it tells the secondary to get ready to receive; this is called **select function**.



Select and poll functions in polling access method

Select

The select function is used whenever the primary device has something to send. Remember that the primary controls the link. If the primary is neither sending nor receiving data, it knows the link is available. If it has something to send, the primary device sends it. What it does not know, however, is whether the target device is prepared to receive. So the primary must alert the secondary to the upcoming transmission and wait for an acknowledgment of the secondary's ready status. Before sending data, the primary creates and transmits a select (SEL) frame, one field of which includes the address of the intended secondary.

Poll

The poll function is used by the primary device to solicit transmissions from the secondary devices. When the primary is ready to receive data, it must ask (poll) each device in turn if it has anything to send. When the first secondary is approached, it responds either with a NAK frame if it has nothing to send or with data (in the form of a data frame) if it does. If the response is negative (a NAK frame), then the primary polls the next secondary in the same manner until it finds one with data to send. When the response is positive (a data frame), the primary reads the frame and returns an acknowledgment (ACK frame), verifying its receipt.

3. TOKEN PASSING

In the token-passing method, the stations in a network are organized in a *logical ring*. In other words, for each station, there is a predecessor and a successor.

1. The predecessor is the station which is logically before the station in the ring;
2. The successor is the station which is after the station in the ring.
3. The current station is the one that is accessing the channel now.

The right to this access has been passed from the predecessor to the current station. The right will be passed to the successor when the current station has no more data to send.

In this method, a special packet called a **token** circulates through the ring. The **possession** of the token gives the station the right to access the channel and send its data. When a station has some data to send, it waits until it receives the token from its predecessor. It then holds the token and sends its data.

When the station has no more data to send, it releases the token, passing it to the next logical station in the ring. The station cannot send data until it receives the token again in the next round. In this process, when a station receives the token and has no data to send, it just passes the data to the next station.

Token management is needed for this access method. Stations must be limited in the time they can have possession of the token. The token must be monitored to ensure it has not been lost or destroyed. For example, if a station that is holding the token fails, the token will disappear from the network. Another function of token management is to assign priorities to the stations and to the types of data being transmitted. And finally, token management is needed to make low-priority stations release the token to high priority stations.

Logical Ring

In a token-passing network, stations do not have to be physically connected in a ring; the ring can be a logical one. Figure 12.11 show four different physical topologies that can create a logical ring.

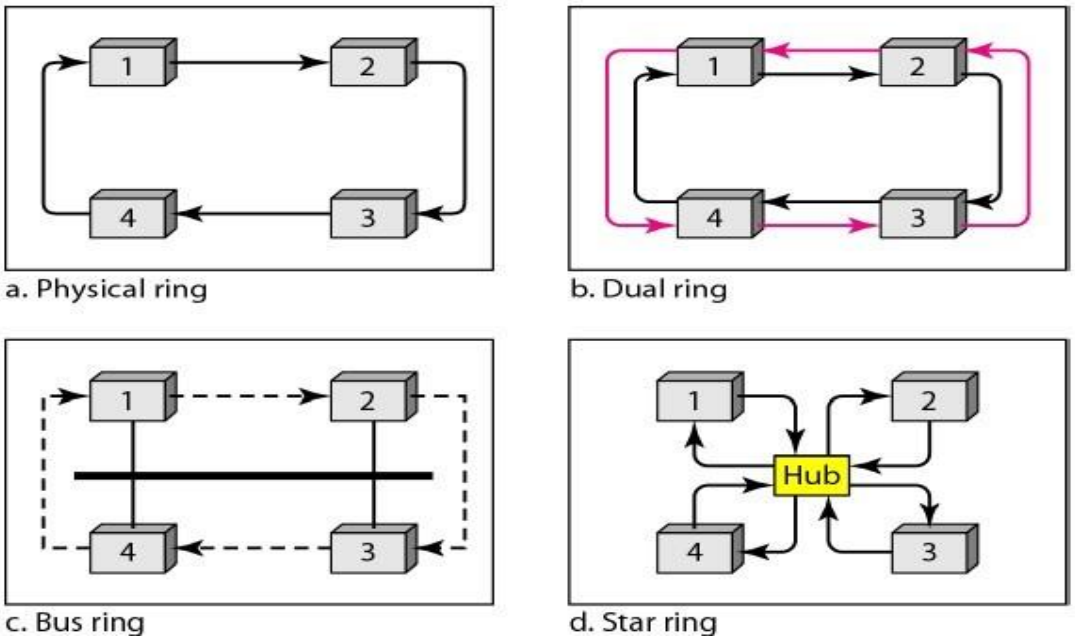


Figure 12.11 Logical ring and physical topology in token-passing access method

In the **Physical Ring Topology**, when a station sends the token to its successor, the token cannot be seen by other stations; the successor is the next one in line. This means that the token does not have to have the address of the next successor. The problem with this topology is that if one of the links-the medium between two adjacent stations fails, the whole system fails. The dual ring topology uses a second (auxiliary) ring which operates in the reverse direction compared with the main ring. The second ring is for emergencies only (such as a spare tire for a car). If one of the links in the main ring fails, the system automatically combines the two rings to form a temporary ring. After the failed link is restored, the auxiliary ring becomes idle again. Note that for this topology to work, each station needs to have two transmitter ports and two receiver ports. The high-speed Token Ring networks called FDDI (Fiber Distributed Data Interface) and CDDI (Copper DistributedData Interface) use this topology.

In the **Bus Ring Topology**, also called a token bus, the stations are connected to a single cable called a bus. They, however, make a logical ring, because each station knows the address of its successor (and also predecessor for token management purposes). When a station has finished sending its data, it releases the token and inserts the address of its successor in the token. Only the station with the address matching the destination address of the token gets the token to access the shared media. The Token Bus LAN, standardized by IEEE, uses this topology.

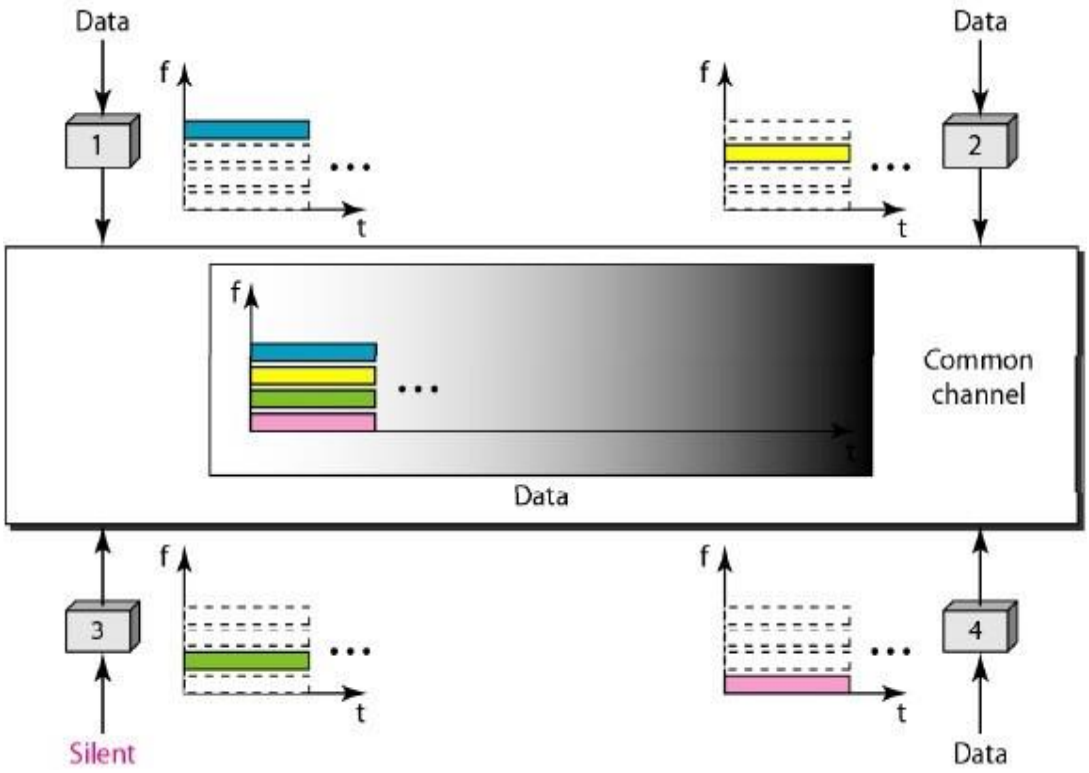
In a **Star Ring Topology**, the physical topology is a star. There is a hub, however, that acts as the connector. The wiring inside the hub makes the ring; the stations are connected to this ring through the two wire connections. This topology makes the network less prone to failure because if a link goes down, it will be bypassed by the hub and the rest of the stations can operate. Also adding and removing stations from the ring is easier. This topology is still used in the Token Ring LAN designed by IBM.

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: FDMA, TDMA, and CDMA.

Frequency-Division Multiple Access (FDMA)

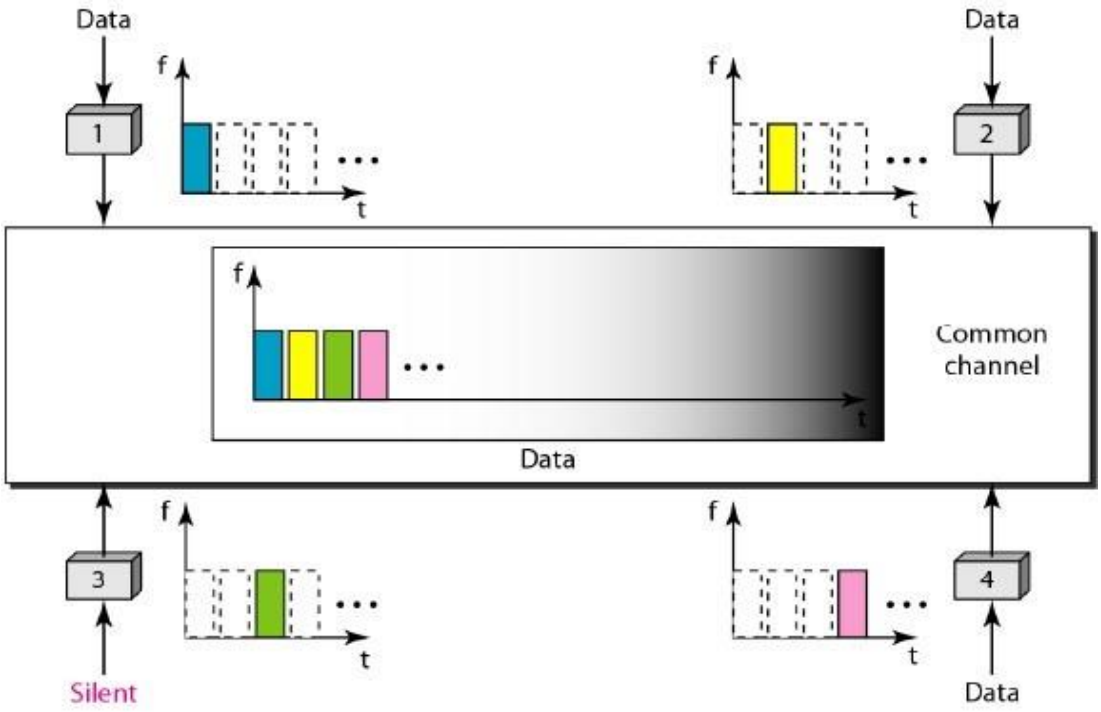
In frequency-division multiple access (FDMA), the available bandwidth is divided into frequency bands. Each station is allocated a frequency band to send its data. In other words, each band is reserved for a specific station, and it belongs to the station all the time. To prevent station interferences, the allocated bands are separated from one another by small *guard bands.*



Time-Division Multiple Access (TDMA)

In time-division multiple access (TDMA), the stations share the bandwidth of the channel in time. Each station is allocated a time slot during which it can send data. Each station transmits its data in is assigned time slot. Figure 12.13 shows the idea behind TDMA.

The main problem with TDMA lies in achieving synchronization between the different stations. Each station needs to know the beginning of its slot and the location of its slot. This may be difficult because of propagation delays introduced in the system if the stations are spread over a large area. To compensate for the delays, we can insert *guard times.* Synchronization is normally accomplished by having some synchronization bits (normally referred to as preamble bits) at the beginning of each slot.



Code-Division Multiple Access (CDMA)

Code-division multiple access (CDMA) was conceived several decades ago. Recent advances in electronic technology have finally made its implementation possible. CDMA differs from FDMA because only one channel occupies the entire bandwidth of the link. It differs from TDMA because all stations can send data simultaneously; there is no timesharing.

IDEA

Let us assume we have four stations 1, 2, 3, and 4 connected to the same channel. The data from station 1 are *d1* , from station 2 are *d2*, and so on. The code assigned to the first station is *c1*, to the second is *c2*, and so on. We assume that the assigned codes have **two properties**:

1. cixcj=0 for all i≠j .i.e. If we multiply each code by another, we get 0.
2. cixci=n(number of stations) i.e. If we multiply each code by itself, we get 4 (the number of stations).

With these two properties in mind, let us see how the above four stations can send data using the same common channel, as shown in Figure 12.14.

1. Station 1 multiplies (a special kind of multiplication, as we will see) its data by its code to get *d1* . *c1*. Station 2 multiplies its data by its code to get *d2* . *c2 a*nd so on. The data that go on the channel are the sum of all these terms, as shown in the box.

2. Any station that wants to receive data from one of the other three multiplies the data on the channel by the code of the sender. For example, suppose stations 1 and 2 are talking to each other. Station 2 wants to hear what station 1 is saying. It multiplies the data on the channel by c1, the code of station 1. Because (*c1 . c1*) is 4, but *(*c2 . c1), *(*c3 . c1), and *(c4 . c1*) are all 0s, station 2 divides the result by 4 to getthe data from station 1.

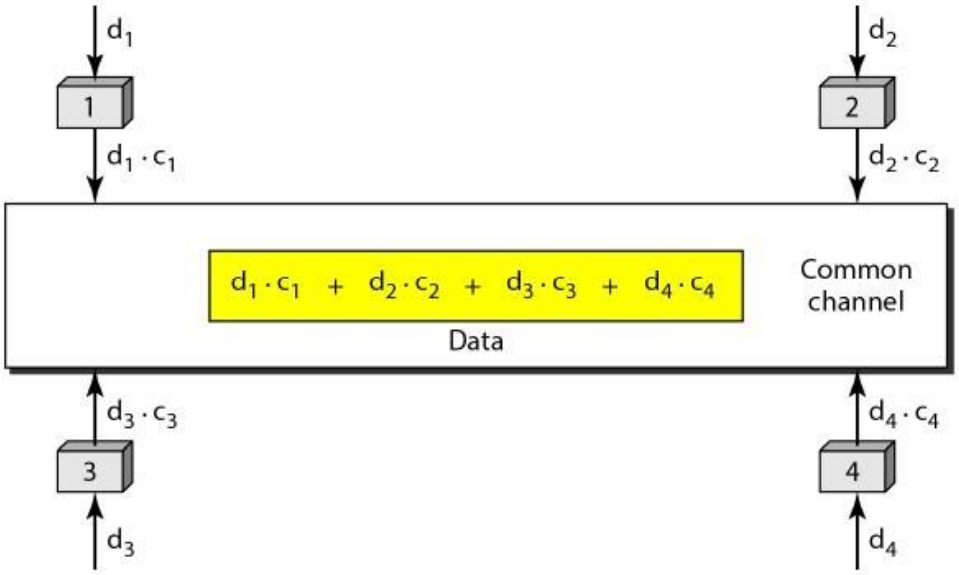


Figure 12.14 Simple idea of communication with code

Data = [*(d1 . c1 + d2 . c2 +d3 . c3 + d4 . c4) . c1]/4*

= [*d 1. c1* . c*1* + d2. c2 . c1 + *d3 . c3* . c1 + *d4 . c4.* c1]/4

= [4 x *d1+0+0+0]/4*

= *4d1/4*

= *d1*

Chips

CDMA is based on coding theory. Each station is assigned a code, which is a sequence of numbers called chips, as shown in Figure 12.15. The codes are for the previous example.

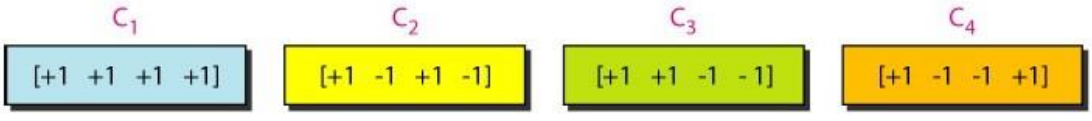


Figure 12.15 Chip sequences

We need to know that we did not choose the sequences randomly; they were carefully selected. They are called orthogonal sequences and have the following properties:

1. Each sequence is made of *N* elements, where *N* is the number of stations.
2. If we multiply a sequence by a number, every element in the sequence is multiplied by that element. This is called multiplication of a sequence by a scalar. For example,
   2.[+1 +1-1-1]=[+2+2-2-2]
3. If we multiply two equal sequences, element by element, and add the results, we get *N*, where *N* is the number of elements in the each sequence. This is called the inner product of two equal sequences. For example,
   [+1 +1-1 -1]· [+1 +1 -1 -1] = 1 + 1 + 1 + 1 = 4
4. If we multiply two different sequences, element by element, and add the results, we get 0. This is called inner product of two different sequences. For example,
   [+1 +1 -1 -1]. [+1 +1 +1 +1] = 1 + 1 - 1 - 1 = 0
5. Adding two sequences means adding the corresponding elements. The result is another sequence. For example,
   [+1+1-1-1]+[+1+1+1+1]=[+2 +2 0 0]

Data Representation

We follow these rules for encoding: If a station needs to send a 0 bit, it encodes it as -1; if it needs to send a 1 bit, it encodes it as +1. When a station is idle, it sends no signal, which is interpreted as a 0. These are shown in Figure 12.16.

Figure 12.16 Data representation in CDMA

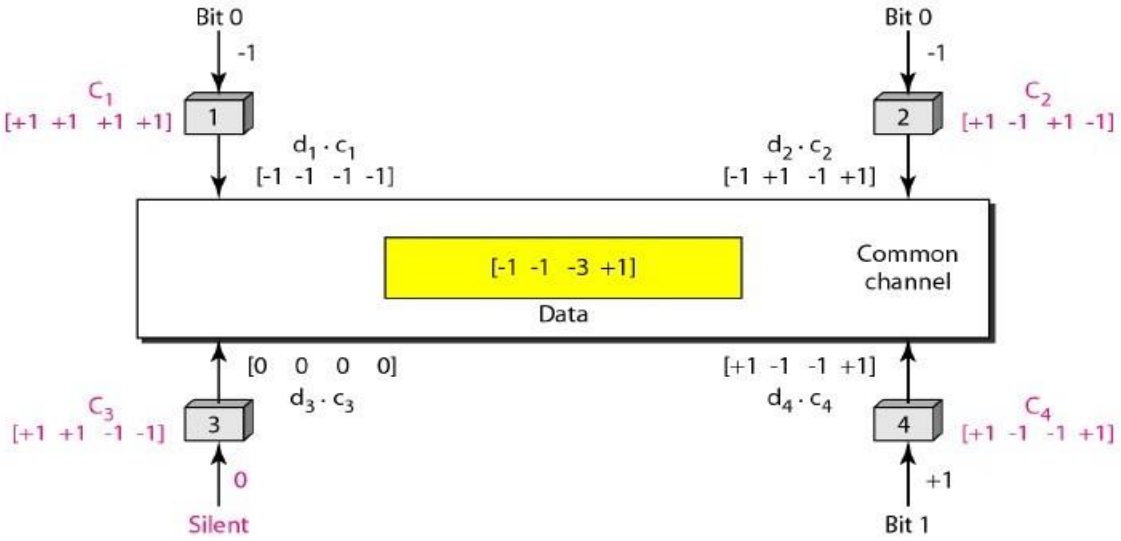
Encoding and Decoding

As a simple example, we show how four stations share the link during a 1-bit interval. The procedure can easily be repeated for additional intervals. We assume that stations 1 and 2 are sending a 0 bit and channel 4 is sending a 1 bit. Station 3 is silent. The data at the sender site are translated to -1, -1, 0, and +1. Each station multiplies the corresponding number by its chip (its orthogonal sequence), which is unique for each station.

The result is a new sequence which is sent to the channel. For simplicity, we assume that all stations send the resulting sequences at the same time. The sequence on the channel is the sum of all four sequences as defined before.

Now imagine station 3, which we said is silent, is listening to station 2. Station 3 multiplies the total data on the channel by the code for station 2, which is [+1 -1 +1-1], to get

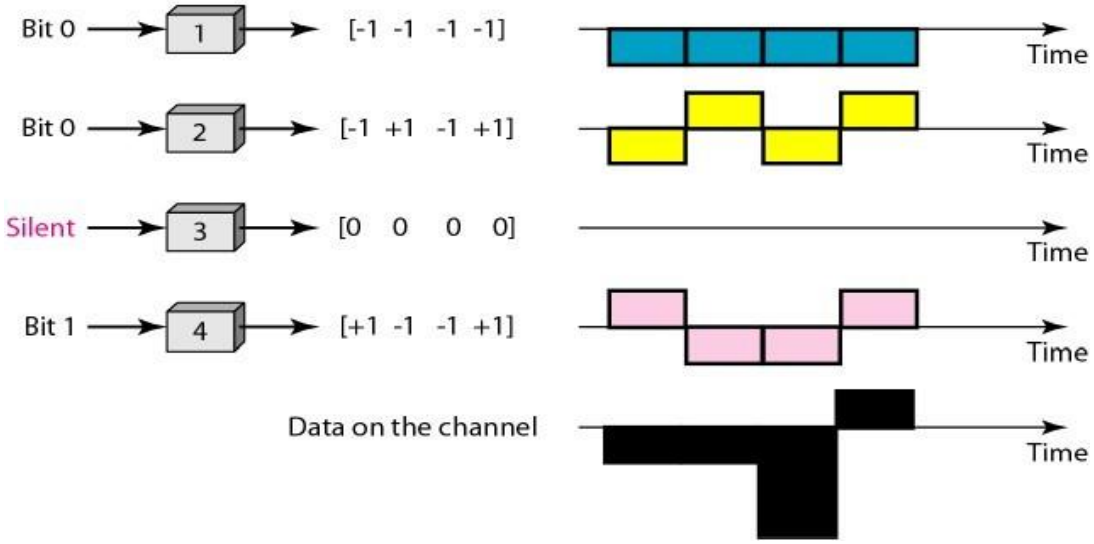
[-1 -1 -3 +1]· [+1 -1 +1 -1] = -4/4 = -1 □ bit 1



Sharing channel in CDMA

Signal Level

The process can be better understood if we show the digital signal produced by each station and the data recovered at the destination (see Figure 12.18). The figure shows the corresponding signals for each station (using NRZ-L for simplicity) and the signal that is onthe common channel.



Digital signal created by four stations in CDMA

Figure 12.19 shows how station 3 can detect the data sent by station 2 by using the code for station 2. The total data on the channel are multiplied (inner product operation) by the signal representing station 2 chip code to get a new signal. The station then integrates and adds the area under the signal, to get the value -4, which is divided by 4 and interpreted as bit 0.

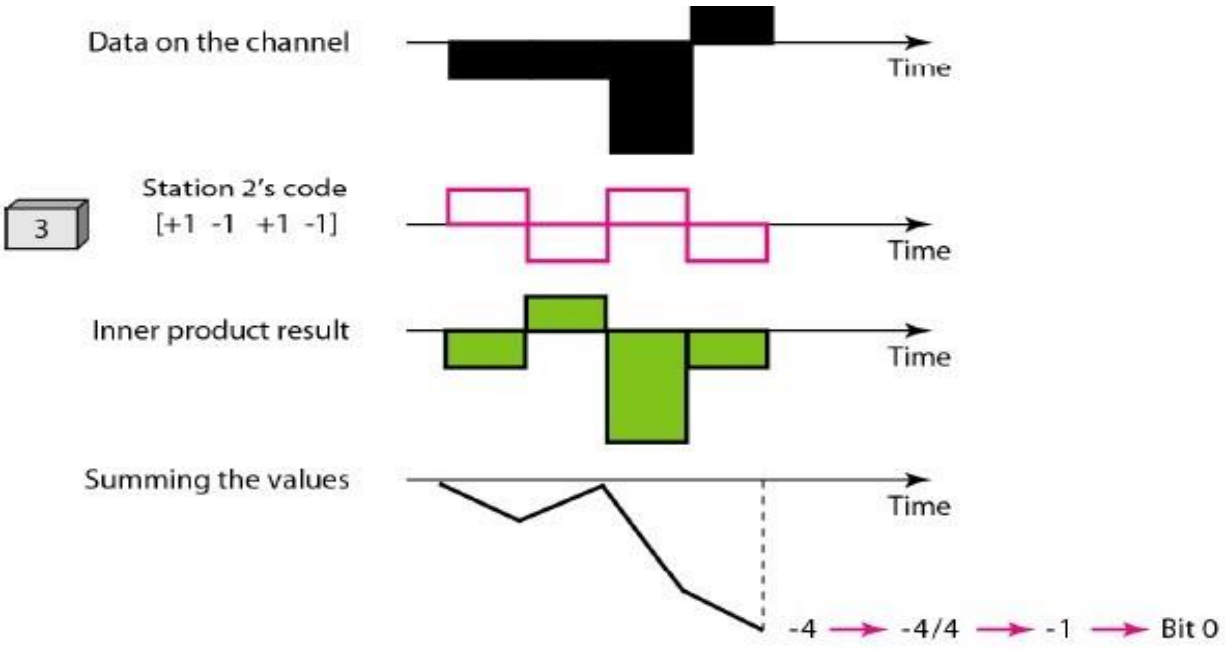
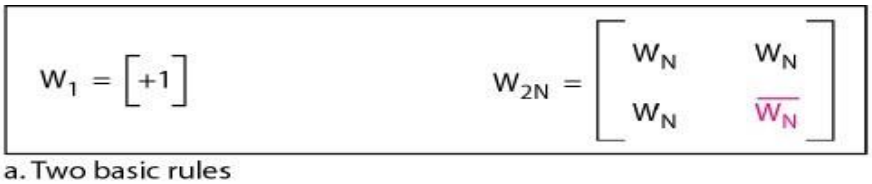


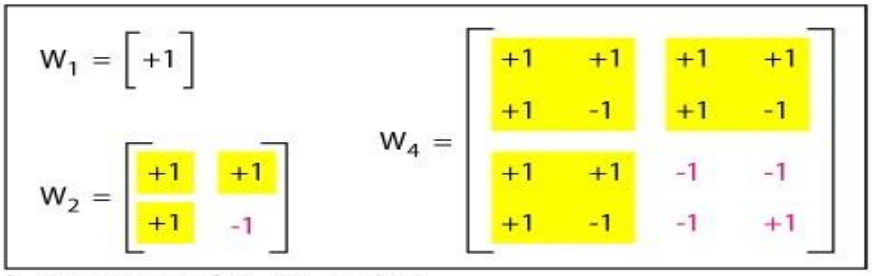
Figure 12.19 Decoding of the composite signal for one in CDMA

Sequence Generation

To generate chip sequences, we use a **Walsh table,** which is a two-dimensional table with an equal number of rows and columns, as shown in Figure 12.20.



a. Two basic rules



b. Generation of W_1, W_2, and W_4

Figure 12.20 General rule and examples of creating Walsh tables

In the Walsh table, each row is a sequence of chips. W_1 for a one-chip sequence has one row and one column. We can choose -1 or +1 for the chip for this trivial table (we chose +1). According to Walsh, if we know the table for N sequences W_N we can create the table for $2N$ sequences W_{2N}, as shown in Figure 12.20. The W_N with the overbar ($\bar{}$) stands for the complement of W_N where each +1 is changed to -1 and vice versa. Figure 12.20 also shows how we can create W_2 and W_4 from W_1 .After we select W_1, W_2 can be made from four W_j's, with the last one the complement of W_1 .After W_2 is generated, W_4 can be made of four W_2's, with the last one the complement of W_2. Of course, W_8 is composed of four W_4's, and so on. Note that after W_N is made, each station is assigned a chip corresponding to a row.

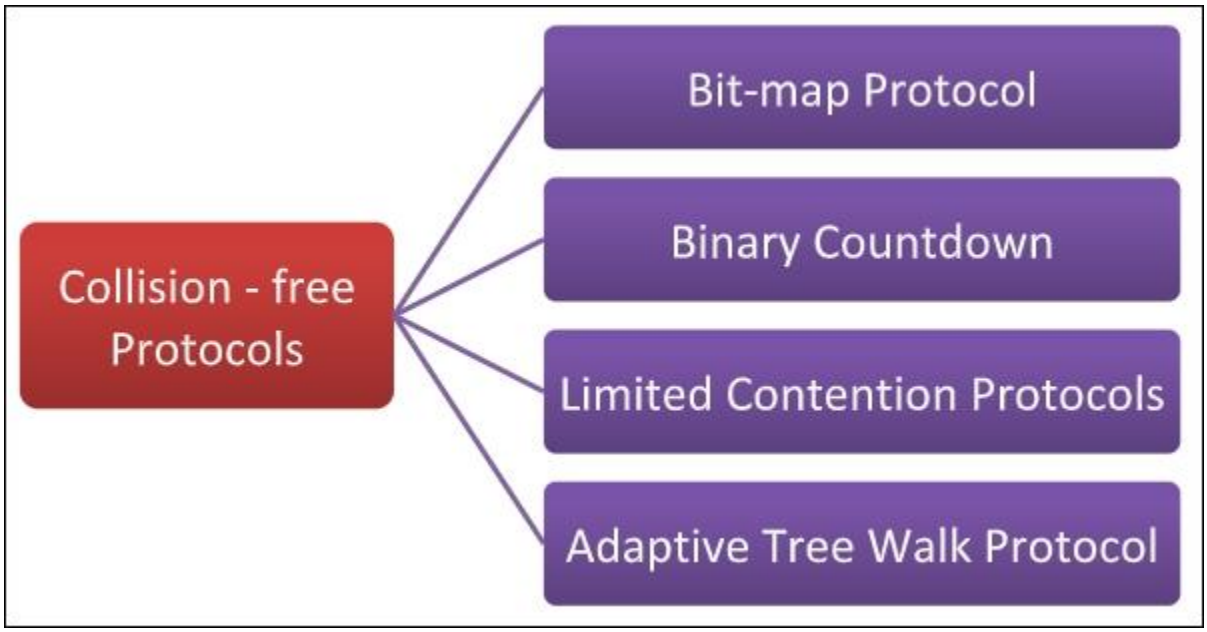
Something we need to emphasize is that the number of sequences N needs to be a power of

In other words, we need to have $N = 2^m$.

COLLISION-FREE PROTOCOLS

In computer networks, when more than one station tries to transmit simultaneously via a shared channel, the transmitted data is garbled. This event is called collision. The Medium Access Control (MAC) layer of the OSI model is responsible for handling collision of frames. Collision – free protocols are devised so that collisions do not occur. Protocols like CSMA/CD and CSMA/CA nullifies the possibility of collisions once the transmission channel is acquired by any station. However, collision can still occur during the contention period if more than one stations starts to transmit at the same time. Collision – free protocols resolves collision in the contention period and so the possibilities of collisions are eliminated.

TYPES OF COLLISION – FREE PROTOCOLS



BIT – MAP PROTOCOL

In bit map protocol, the contention period is divided into N slots, where N is the total number of stations sharing the channel. If a station has a frame to send, it sets the corresponding bit in the slot. So, before transmission, each station knows whether the other stations want to transmit. Collisions are avoided by mutual agreement among the contending stations on who gets the channel.

BINARY COUNTDOWN

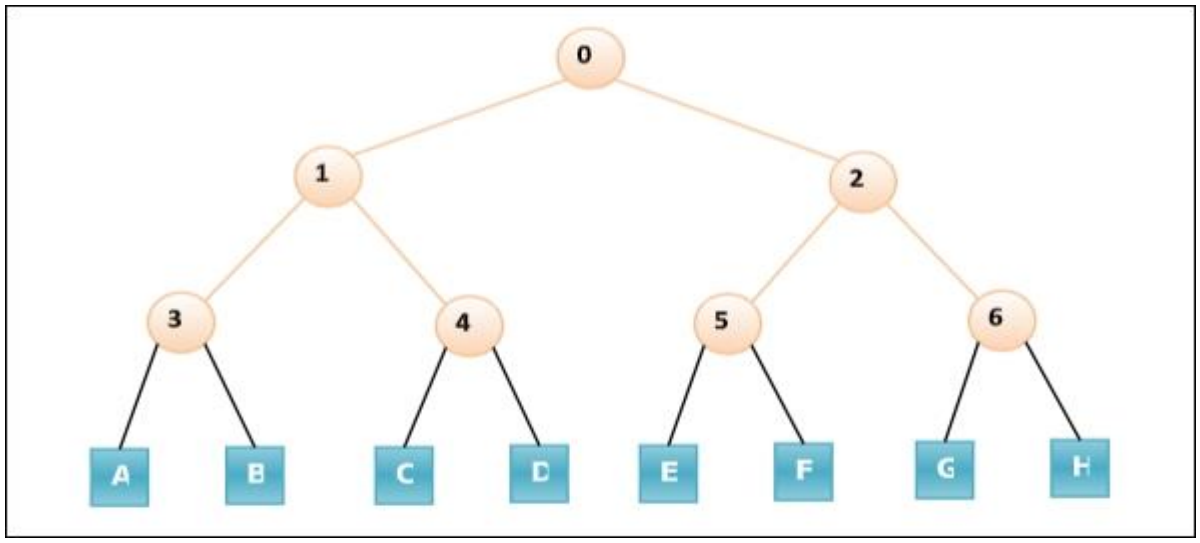
This protocol overcomes the overhead of 1 bit per station of the bit – map protocol. Here, binary addresses of equal lengths are assigned to each station. For example, if there are 6 stations, they may be assigned the binary addresses 001, 010, 011, 100, 101 and 110. All stations wanting to communicate broadcast their addresses. The station with higher address gets the higher priority for transmitting.

LIMITED CONTENTION PROTOCOLS

These protocols combines the advantages of collision based protocols and collision free protocols. Under light load, they behave like ALOHA scheme. Under heavy load, they behave like bitmap protocols.

ADAPTIVE TREE WALK PROTOCOL

In adaptive tree walk protocol, the stations or nodes are arranged in the form of a binary tree as follows -



Initially all nodes (A, B ……. G, H) are permitted to compete for the channel. If a node is successful in acquiring the channel, it transmits its frame. In case of collision, the nodes are divided into two groups (A, B, C, D in one group and E, F, G, H in another group). Nodes belonging to only one of them are permitted for competing. This process continues until successful transmission occurs.

ETHERNET

IEEE Project 802

The IEEE has subdivided the data-link layer into two sublayers: logical link control (LLC) and media access control (MAC). IEEE has also created several physical-layer standards for differentLAN protocols.

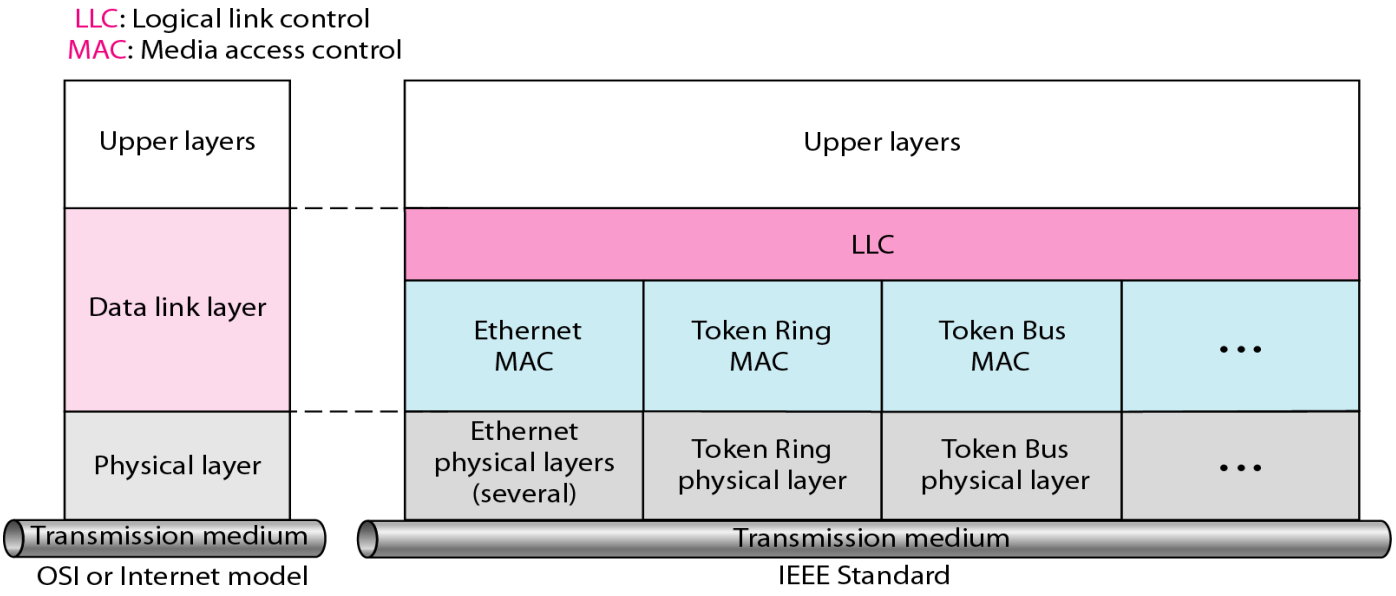
Logical Link Control (LLC)

1. In IEEE Project 802, flow control, error control, and part of the framing duties are collected into one sub layer called the logical link control(LLC). Framing is handled in both the LLC sub layer and the MAC sub layer
2. The LLC provides a single link-layer control protocol for all IEEE LANs. This means LLC protocol can provide interconnectivity between different LANs because it makes the MAC sub layer transparent.

Media Access Control (MAC)

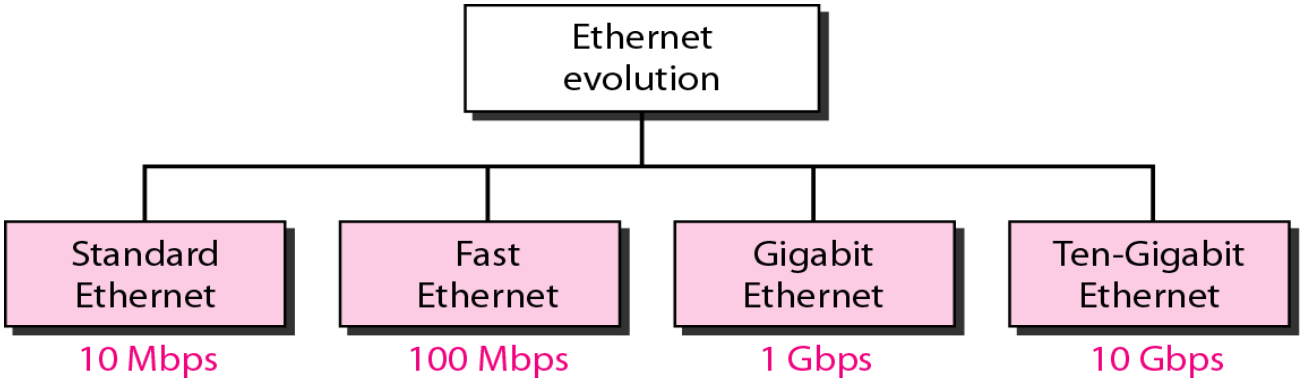
1. IEEE Project 802 has created a sub layer called media access control that defines the specific access method for each LAN.
2. For example, it defines CSMA/CD as the media access method for Ethernet LANs and defines the token-passing method for Token Ring and Token Bus LANs

IEEE standard for LANs



ETHERNET EVOLUTION:

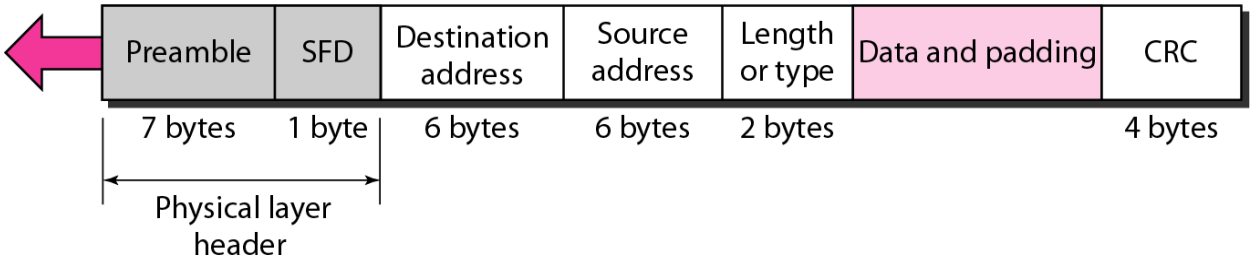
The Ethernet LAN was developed in the 1970s by Robert Metcalfe and David Boggs. Since then, it has gone through four generations: Standard Ethernet (10 Mbps), Fast Ethernet (100 Mbps), Gigabit Ethernet (1 Gbps), and 10 Gigabit Ethernet(10 Gbps)



802.3 MAC frame

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

SFD: Start frame delimiter, flag (10101011)



STANDARD ETHERNET

The original Ethernet technology with the data rate of 10 Mbps as the *Standard Ethernet is referred* . Although most implementations have moved to other technologies in the Ethernet evolution, there are some features of the Standard Ethernet that have not changed during the evolution.

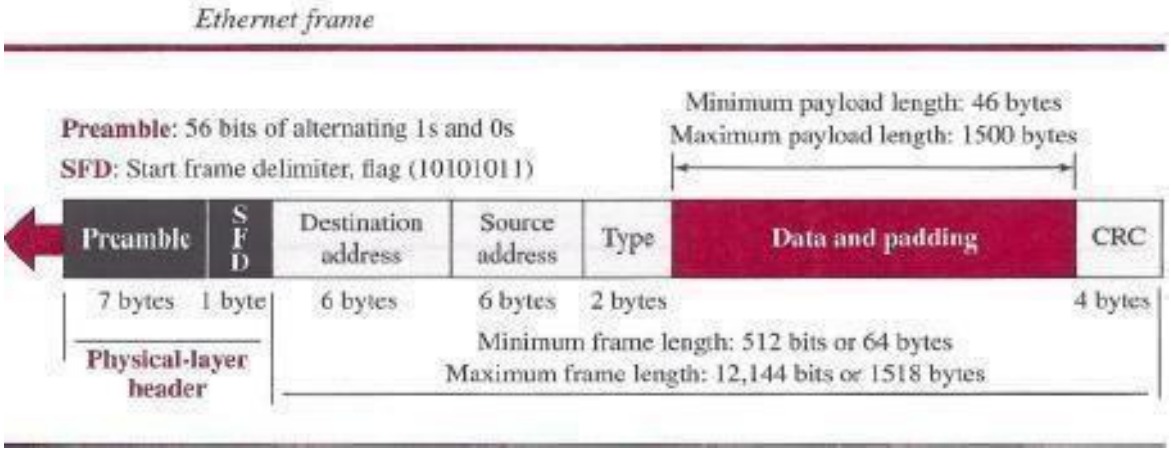
CHARACTERISTICS

Connectionless and Unreliable Service

Ethernet provides a connectionless service, which means each frame sent is independent of the previous or next frame. Ethernet has no connection establishment or connection termination phases. The sender sends a frame whenever it has it; the receiver mayor may not be ready for it. The sender may overwhelm the receiver with frames, which may result in dropping frames. If a frame drops, the sender will not know about it. Since IP, which is using the service of Ethernet, is also connectionless, it will not know about it either. If the transport layer is also a connectionless protocol, such as UDP, the frame is lost and salvation may only come from the application layer. However, if the transport layer is TCP, the sender TCP does not receive acknowledgment for its segment and sends it again. Ethernet is also unreliable like IP and UDP. If a frame is corrupted during transmission and the receiver finds out about the corruption, which has a high level of probability of happening because of the CRC-32, the receiver drops the frame silently. It is the duty of high-level protocols to find out about it.

ETHERNET FRAME FORMAT

The Ethernet frame contains seven fields, as shown in below figure.



Preamble This field contains 7 bytes (56 bits) of alternating Os and Is that alert the receiving system to the coming frame and enable it to synchronize its clock if it's out of synchronization. The pattern provides only an alert and a timing pulse. The 56-bit pattern allows the stations to miss some bits at the beginning of the frame. The *preamble* is actually added at the physical layer and is not (formally) part of the frame.

Start frame delimiter (SFD) This field (1 byte: 10101011) signals the beginning of the frame. The SFD warns the station or stations that this is the last chance for synchronization. The last 2 bits are (llh and alert the receiver that the next field is the destination address. This field is actually a flag that defines the beginning of the frame. We need to remember that an Ethernet frame is a variable-length frame. It needs a flag to define the beginning of the frame. The SFD field is also added at the physical layer.

Destination address (DA) This field is six bytes (48 bits) and contains the link layer address of the destination station or stations to receive the packet. When the receiver sees its own link-layer address, or a multicast address for a group that the receiver is a member of, or a broadcast address, it de-capsulates the data from the frame and passes the data to the upper layer protocol defined by the value of the type field.

Source address (SA) This field is also six bytes and contains the link-layer address of the sender of the packet.

Type This field defines the upper-layer protocol whose packet is encapsulated in the frame. This protocol can be IP, ARP, OSPF, and so on. In other words, it serves the same purpose as the protocol field in a datagram and the port number in a segment or user datagram. It is used for multiplexing and de-multiplexing.

Data This field carries data encapsulated from the upper-layer protocols. It is a minimum of 46 and a maximum of 1500 bytes. We discuss the reason for these minimum and maximum values shortly. If the data coming from the upper layer is more than 1500 bytes, it should be fragmented and encapsulated in more than one frame. If it is less than 46 bytes, it needs to be padded with extra Os. A padded data frame is delivered to the upper-layer protocol as it is (without removing the padding), which means that it is the responsibility of the upper layer to remove or, in the case of the sender, to add the padding. The upper-layer protocol needs to know the length of its data. For example, a datagram has a field that defines the length of the data.

CRC The last field contains error detection information, in this case a CRC-32. The CRC is calculated over the addresses, types, and data field. If the receiver calculates the CRC and finds that it is not zero (corruption in transmission), it discards the frame.

ETHERNET FRAME LENGTH

Ethernet has imposed restrictions on both the minimum and maximum lengths of a frame. The minimum length restriction is required for the correct operation of CSMAlCD. An Ethernet frame needs to have a minimum length of 512 bits or 64 bytes. Part of this length is the header and the trailer. If we count 18 bytes of header and trailer (6 bytes of source address, 6 bytes of destination address, 2 bytes of length or type, and 4 bytes of CRC), then the minimum length of data from the upper layer is 64 - 18 = 46 bytes. If the upper-layer packet is less than 46 bytes, padding is added to make up the difference.

Minimum frame length: 64 bytes **Minimum data length: 46 bytes**

Maximum frame length: 1518 bytes **Maximum data length: 1500 bytes**

The standard defines the maximum length of a frame (without preamble and SFD field) as 1518 bytes. If we subtract the 18 bytes of header and trailer, the maximum length of the payload is 1500 bytes.

The maximum length restriction has two historical reasons. First, memory was very expensive when Ethernet was designed; a maximum length restriction helped to reduce the size of the buffer. Second, the maximum length restriction prevents one station from monopolizing the shared medium, blocking other stations that have data to send.

Addressing

Each station on an Ethernet network (such as a PC, workstation, or printer) has its own network interface card (NIC). The NIC fits inside the station and provides the station with a link-layer address. The Ethernet address is 6 bytes (48 bits), normally written in hexadecimal notation, witha colon between the bytes. For example, the following shows an Ethernet MAC address:
4A:30:10:21:10:1A

Transmission of Address Bits

The way the addresses are sent out online is different from the way they are written in hexadecimal notation. The transmission is left to right, byte by byte; however, for each byte, the least significant bit is sent first and the most significant bit is sent last. This means that the bit that defines an address as unicast or multicast arrives first at the receiver. This helps the receiver to immediately know if the packet is unicast or multicast.

Unicast, Multicast, and Broadcast Addresses

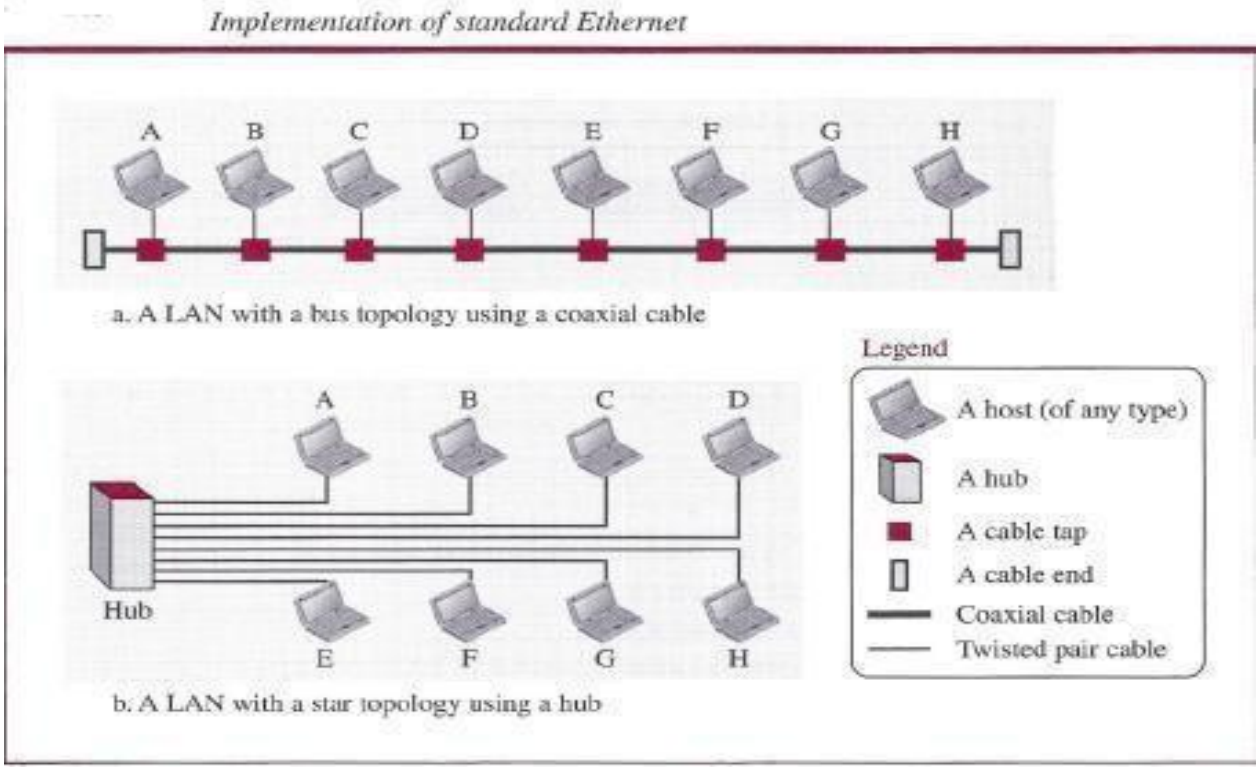
A source address is always a *unicast address-the* frame comes from only one station. The destination address, however, can be *unicast, multicast,* or *broadcast. Below f*igure shows how to distinguish a unicast address from a multicast address. If the least significant bit of the first byte in a destination address is 0, the address is unicast; otherwise, it is multicast. Note that with the way the bits are transmitted, the unicast/multicast bit is the first bit which is transmitted or received. The broadcast address is a special case of the multicast address: the recipients are all the stations on the LAN. A broadcast destination address is forty-eight 1s.

Distinguish Between Unicast, Multicast, and Broadcast Transmission

Standard Ethernet uses a coaxial cable (bus topology) or a set of twisted-pair cables with a hub (star topology) as shown in Figure. We need to know that transmission in the standard Ethernet is always broadcast no matter if the intention is unicast, multicast, or broadcast. In the bus topology, when station A sends a frame to station B, all stations will receive it. In the star topology, when station A sends a frame to station B, the hub will receive it. Since the hub is a passive element, it does not check the destination address of the frame; it regenerates the bits (if they have been weakened) and sends them to all stations except station A.

In fact, it floods the network with the frame. The question is, then, how the actual unicast, multicast, and broadcast transmissions are distinguished from each other. The answer is in the way the frames are kept or dropped.

1. In a unicast transmission, all stations will receive the frame, the intended recipient keeps and handles the frame; the rest discard it.
2. In a multicast transmission, all stations will receive the frame, the stations that are members of the group keep and handle it; the rest discard it.
3. In a broadcast transmission, all stations (except the sender) will receive the frame and all stations (except the sender) keep and handle it.



Access Method

Since the network that uses the standard Ethernet protocol is a broadcast network, we need to use an access method to control access to the sharing medium. The standard Ethernet choose *CSMA/CD* with l-persistent method.

Let us use a scenario to see how this method works for the Ethernet protocol.

1. Assume station A in above figure has a frame to send to station D. Station A first should check whether any other station is sending (carrier sense). Station A measures the level of energy on the medium (for a short period of time, normally less than 100 us). If there is no signal energy on the medium, it means that no station is sending (or the signal has not reached station A). Station A interprets this situation as idle medium. It starts sending its frame. On the other hand, if the signal energy level is not zero, it means that the medium is being used by another station. Station A continuously monitors the medium until it becomes idle for 100 us, It then starts sending the frame. However, station A needs to keep a copy of the frame in its buffer until it is sure that there is no collision. When station A is sure of this is the subject.
2. The medium sensing does not stop after station A has started sending the frame. Station A needs to send and receive continuously. Two cases may occur:
    a. Station A has sent 512 bits and no collision is sensed (the energy level did not go above the regular energy level), the station then is sure that the frame will go through and stops sensing the medium. Where does the number 512 bits come from? If we consider the transmission rate of the Ethernet as 10 Mbps, this means that it takes the station 512/(10 Mbps) = 5l.2 us to send out 512 bits. With the speed of propagation in a cable (2 x 108 meters), the first bit could have gone 10,240 meters (one way) or only 5120 meters (round trip), have collided with a bit from the last station on the cable, and have gone back. In other words, if a collision were to occur, it should occur by the time the sender has sent out 512 bits (worst case) and the first bit has made a round trip of 5120 meters. We should know that if the collision happens in the middle of the cable, not at the end, station A hears the collision earlier and aborts the transmission. We also need to mention another issue. The above assumption is that the length of the cable is 5120 meters. The designer of the standard Ethernet actually put a restriction of 2500 meters because we need to consider the delays encountered throughout the journey. It means that they considered the worst case. The whole idea is that if station A does not sense the collision before sending 512 bits, there must have been no collision, because during this time, the first bit has reached the end of the line and all other stations know that a station is sending and refrain from sending.

In other words, the problem occurs when another station (for example, the last station) starts sending before the first bit of station A has reached it. The other station mistakenly thinks that the line is free because the first bit has not yet reached it. The reader should notice that the restriction of 512 bits actually helps the sending station: The sending station is certain that no collision will occur if it is not heard during the first 512 bits, so it can discard the copy of the frame in its buffer.

b. Station A has sensed a collision before sending 512 bits. This means that one of the previous bits has collided with a bit sent by another station. In this case both stations should refrain from sending and keep the frame in their buffer for resending when the line becomes available. However, to inform other stations that there is a collision in the network, the station sends a 48-bit jam signal. The jam signal is to create enough signal (even if the collision happens after a few bits) to alert other stations about the collision. After sending the jam signal, the stations need to increment the value of K (number of attempts). If after increment $K = 15$, the experience has shown that the network is too busy, the station needs to abort its effort and try again. If $K < 15$, the station can wait a back off time and restart the process. As Figure 12.l3 shows, the station creates a random number between 0 and $2K - 1$, which means each time the collision occurs, the range of the random number increases exponentially. After the first collision $(K = 1)$ the random number is in the range (0, 1). After the second collision $(K = 2)$ it is in the range (0, 1, 2, 3). After the third collision $(K = 3)$ it is in the range (0, 1, 2,3, 4,5,6,7). So after each collision, the probability increases that the back off time becomes longer. This is due to the fact that if the collision happens even after the third or fourth attempt, it means that the network is really busy; a longer back off time is needed.

Efficiency of Standard Ethernet

The efficiency of the Ethernet is defined as the ratio of the time used by a station to send data to the time the medium is occupied by this station. The practical efficiency of standard Ethernet has been measured to be **Efficiency = 1 / (1 + 6.4 x *a)*** in which the parameter *"a"* is the number of frames that can fit on the medium. It can be calculated as a = (propagation delay)/(transmission delay) because the transmission delay is the time it takes a frame of average size to be sent out and the propagation delay is the time it takes to reach the end of the medium.

Note that as the value of parameter *a* decreases, the efficiency increases. This means that if the length of the media is shorter or the frame size longer, the efficiency increases. In the ideal case, $a = 0$ and the efficiency is 1.

Implementation

The Standard Ethernet defined several implementations, but only four of them became popular during the 1980s. Table shows a summary of Standard Ethernet implementations.

Summary of Standard Ethernet implementations

Implementation	*Medium*	*Medium Length*	*Encoding*
10Base5	Thick coax	500 m	Manchester
10Base2	Thin coax	185 m	Manchester
10Base-T	2 UTP	100 m	Manchester
10Base-F	2 Fiber	2000 m	Manchester

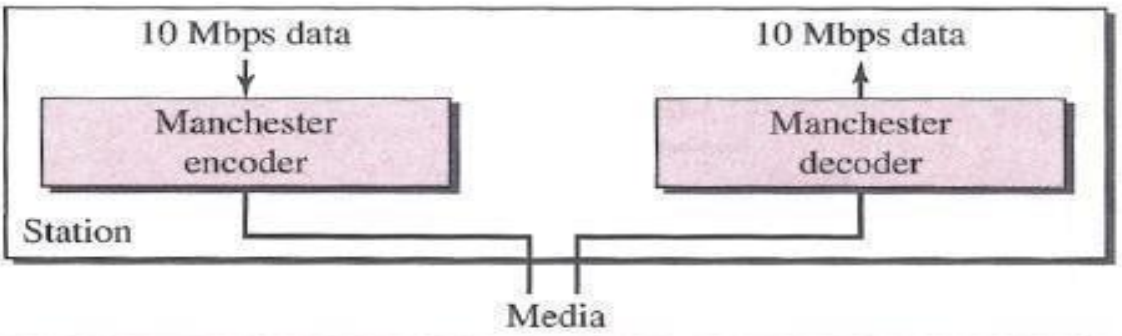
Table

In the nomenclature 10BaseX, the number defines the data rate (10 Mbps), the term *Base* means baseband (digital) signal, and X approximately defines either the maximum size of the cable in 100 meters (for example 5 for 500 or 2 for 185 meters) or the type of cable, T for unshielded twisted pair cable (UTP) and F for fiber-optic. The standard Ethernet uses a baseband signal, which means that the bits are changed to a digital signal and directly sent on the line.

Encoding and Decoding

All standard implementations use digital signaling (baseband) at 10 Mbps. At the sender, data are converted to a digital signal using the Manchester scheme; at the receiver, the received signal is interpreted as Manchester and decoded into data. Manchester encoding is self-synchronous, providing a transition at each bit interval. Figure shows the encoding scheme for Standard Ethernet.

Encoding in a Standard Ethernet implementation

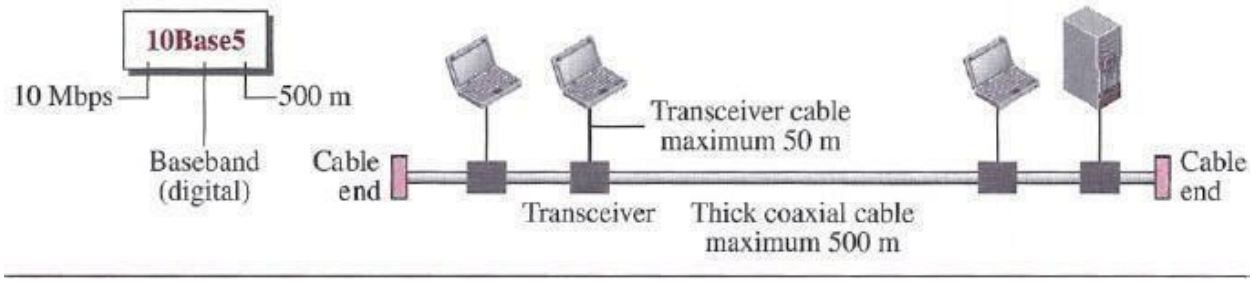


ETHERNET CABLING

10Base5: Thick Ethernet

The first implementation is called *1OBaseS, thick Ethernet,* or *Thicknet.* The nickname derives from the size of the cable, which is roughly the size of a garden hose and too stiff to bend with your hands. 10Base5 was the first Ethernet specification to use a bus topology with an external transceiver (transmitter/receiver) connected via a tap to a thick coaxial cable. Figure shows a schematic diagram of a 10Base5 implementation.

10Base5 implementation

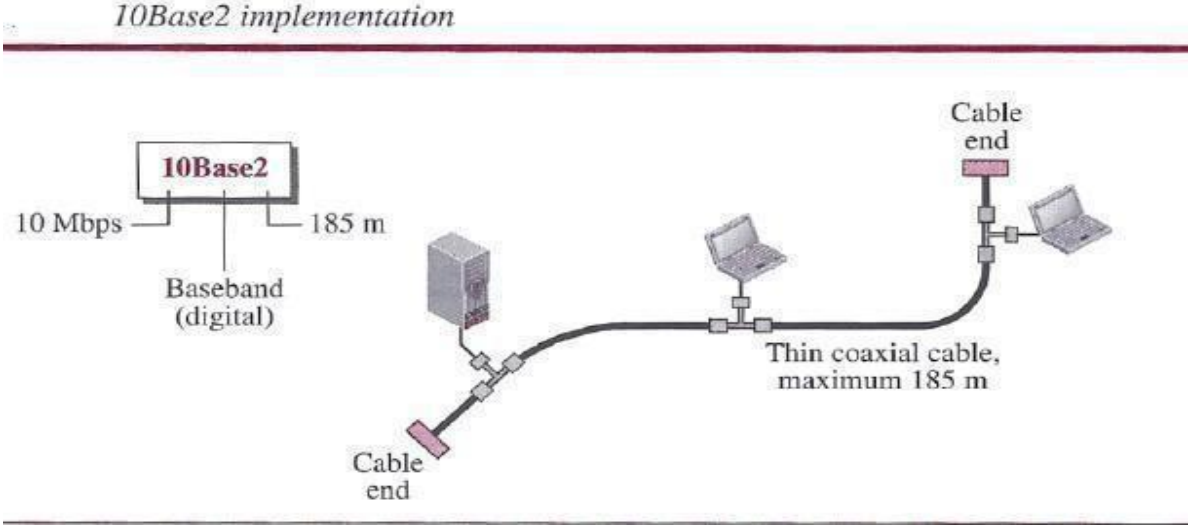


The transceiver is responsible for transmitting, receiving, and detecting collisions. The transceiver is connected to the station via a transceiver cable that provides separate paths for sending and receiving. This means that collision can only happen in the coaxial cable. The maximum length of the coaxial cable must not exceed 500 m, otherwise, there is excessive degradation of the signal. If a length of more than 500 m is needed, up to five segments, each a maximum of 500 meters, can be connected using repeaters.

10Base2: Thin Ethernet

The second implementation is called *lOBase2, thin Ethernet,* or *Cheaper net.* lOBase2 also uses a bus topology, but the cable is much thinner and more flexible. The cable can be bent to pass very close to the stations. In this case, the transceiver is normally part of the network interface card (NIC), which is installed inside the station. Figure shows the schematic diagram of a 10Base2 implementation.

10Base2 implementation

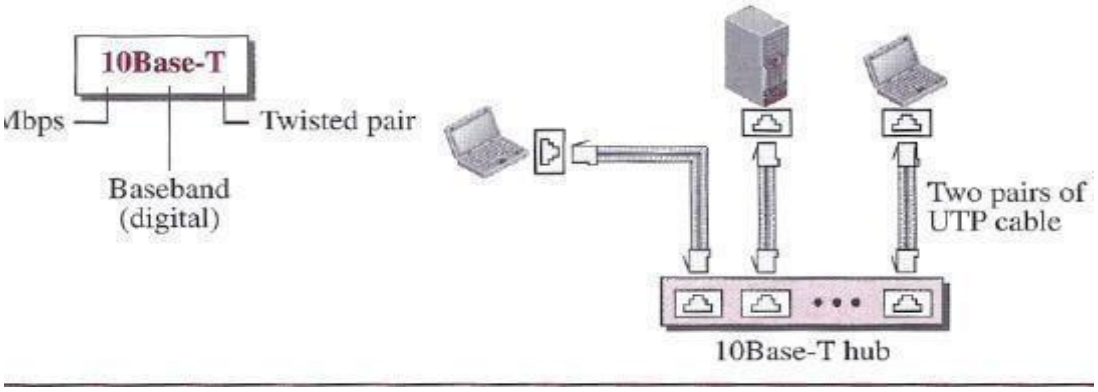


Note that the collision here occurs in the thin coaxial cable. This implementation is more cost effective than 10Base5 because thin coaxial cable is less expensive than thick coaxial and the tee connections are much cheaper than taps. Installation is simpler because the thin coaxial cable is very flexible. However, the length of each segment cannot exceed 185 m (close to 200 m) due to the high level of attenuation in thin coaxial cable.

10Base-T: Twisted-Pair Ethernet

The third implementation is called *1OBase- T* or *twisted-pair Ethernet.* 1OBase-T uses a physical star topology. The stations are connected to a hub via two pairs of twisted cable, as shown in Figure.

10Base-T implementation

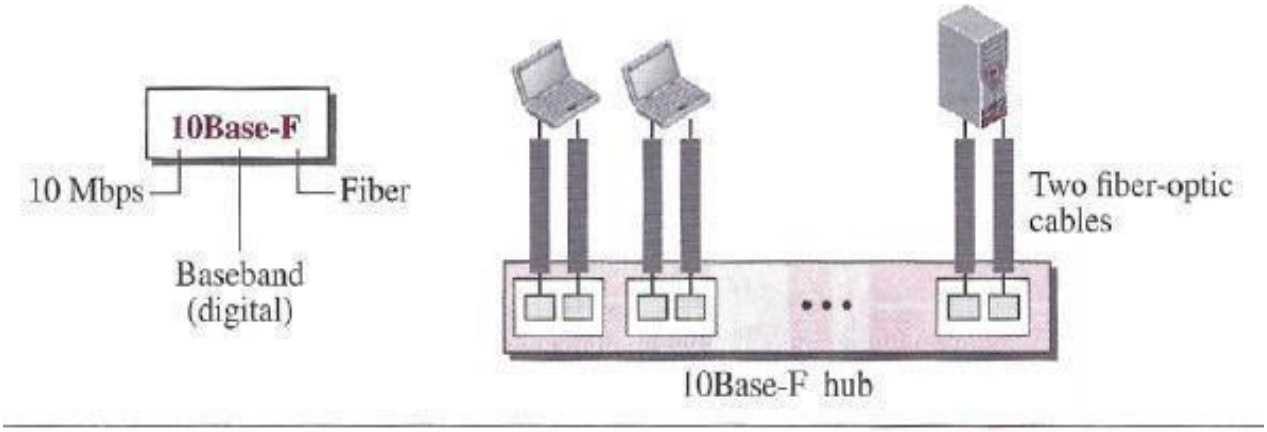


Note that two pairs of twisted cable create two paths (one for sending and one for receiving) between the station and the hub. Any collision here happens in the hub. Compared to lOBase5 or lOBase2, we can see that the hub actually replaces the coaxial cable as far as a collision is concerned. The maximum length of the twisted cable here is defined as 100 m, to minimize the effect of attenuation in the twisted cable.

10Base-F: Fiber Ethernet

Although there are several types of optical fiber 10-Mbps Ethernet, the most common is called *lOBase-F*. 1OBase-F uses a star topology to connect stations to a hub. The stations are connectedto the hub using two fiber-optic cables, as shown in Figure.

10Base-F implementation



PHYSICAL LAYER

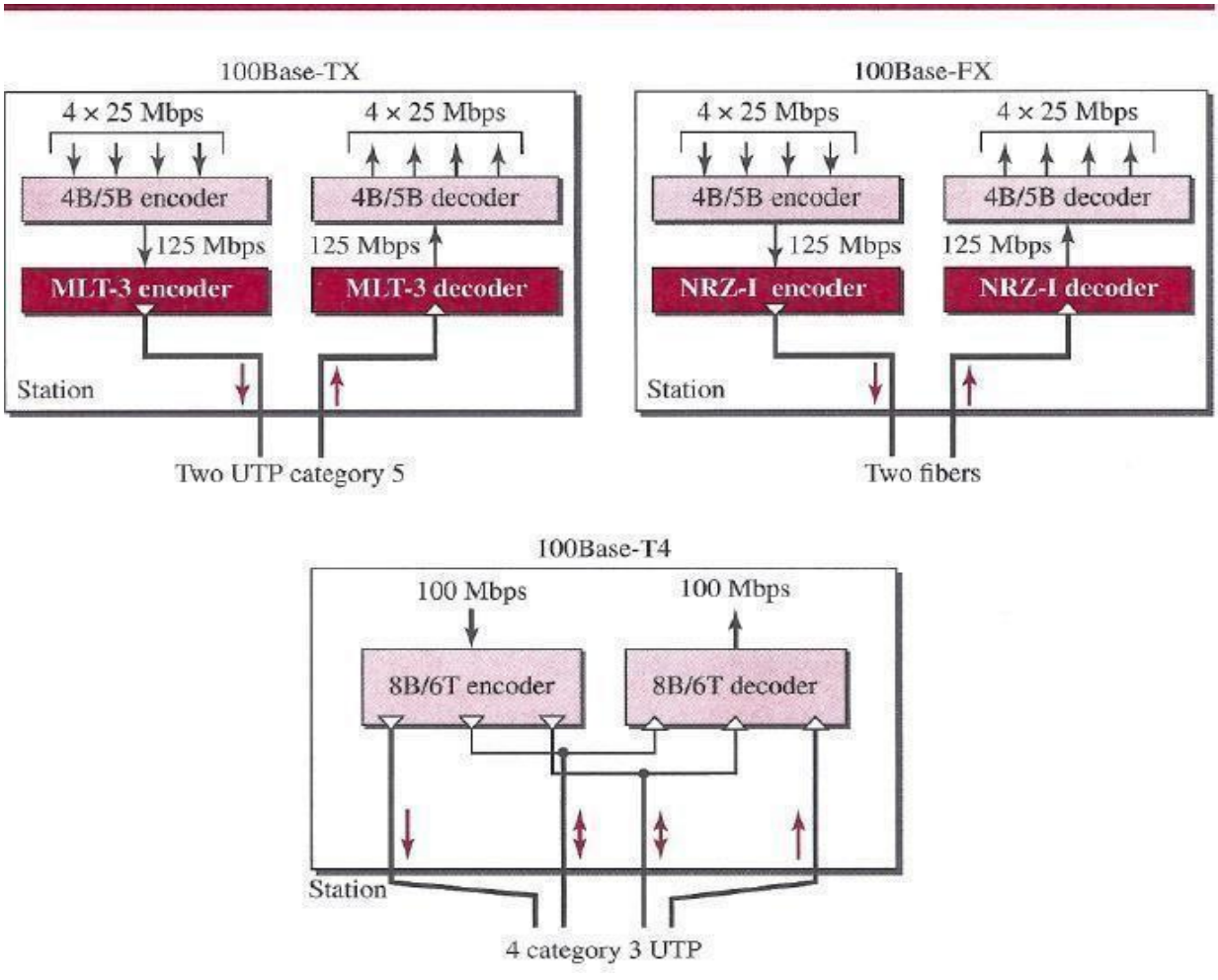
To be able to handle a 100 Mbps data rate, several changes need to be made at the physical layer.

Topology

Fast Ethernet is designed to connect two or more stations. If there are only two stations, they can be connected point-to-point. Three or more stations need to be connected in a star topology with a hub or a switch at the center.

Encoding

Manchester encoding needs a 200-Mbaud bandwidth for a data rate of 100 Mbps, which makes it unsuitable for a medium such as twisted-pair cable. For this reason, the Fast Ethernet designers sought some alternative encoding/decoding scheme. However, it was found that one scheme would not perform equally well for all three implementations. Therefore, three different encoding schemes were chosen



100Base- TX uses two pairs of twisted-pair cable (either category S UTP or STP). For this implementation, the MLT-3 scheme was selected since it has good bandwidth performance. (However, since MLT-3 is not a self-synchronous line coding scheme, 4B/SB block coding is used to provide bit synchronization by preventing the occurrence of a long sequence of Os and Is. This creates a data rate of 125 Mbps, which is fed into MLT-3 for encoding.

100Base-FX uses two pairs of fiber-optic cables. Optical fiber can easily handle high bandwidth requirements by using simple encoding schemes. The designers of 100Base-FX selected the NRZ-I encoding scheme for this implementation. However, NRZ-I has a bit synchronization problem for long sequences of 0s (or Is, based on the encoding). To overcome this problem, the designers used 4B/5B block encoding, as we described for 100Base- TX. The block encoding increases the bit rate from 100 to 125 Mbps, which can easily be handled by fiber-optic cable. A 100Base-TX network can provide a data rate of 100 Mbps, but it requires the use of category 5 UTP or STP cable. This is not cost-efficient for buildings that have already been wired for voice- grade twisted-pair (category 3). A new standard, called ***lOOBase-T4,*** was designed to use category 3 or higher UTP. The implementation uses four pairs of UTP for transmitting 100 Mbps. Encoding/decoding in 100Base-T4 is more complicated. As this implementation uses category 3 UTP, each twisted-pair cannot easily handle more than 25 Mbaud. In this design, one pair switches between sending and receiving. Three pairs of UTP category 3, however, can handle only 75 Mbaud (25 Mbaud) each. We need to use an encoding scheme that converts 100 Mbps to a 75 Mbaud signal. 8B/6T satisfies this requirement. In 8B/6T, eight data elements are encoded as six signal elements. This means that 100 Mbps uses only (6/8) x 100 Mbps, or 75 Mbaud.

MAC SUBLAYER

A main consideration in the evolution of Ethernet was to keep the MAC sub layer untouched. However, to achieve a data rate of 1 Gbps, this was no longer possible. Gigabit Ethernet has two distinctive approaches for medium access: half-duplex and full duplex. Almost all implementations of Gigabit Ethernet follow the full-duplex approach, so we mostly ignore the half-duplex mode.

Full-Duplex Mode

In full-duplex mode, there is a central switch connected to all computers or other switches. In this mode, for each input port, each switch has buffers in which data are stored until they are transmitted. Since the switch uses the destination address of the frame and sends a frame out of the port connected to that particular destination, there is no collision. This means that *CSMA/CD* is not used. Lack of collision implies that the maximum length of the cable is determined by the signal attenuation in the cable, not by the collision detection process. 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.

Half-Duplex Mode

Gigabit Ethernet can also be used in half-duplex mode, although it is rare. In this case, a switch can be replaced by a hub, which acts as the common cable in which a collision might occur. The half-duplex approach uses *CSMA/CD*. However, as we saw before, the maximum length of the network in this approach is totally dependent on the minimum frame size. Three methods have been defined: traditional, carrier extension, and frame bursting.

Traditional

In the traditional approach, we keep the minimum length of the frame as in traditional Ethernet (512 bits). However, because the length of a bit is *1/100* shorter in Gigabit Ethernet than in 10- Mbps Ethernet, the slot time for Gigabit Ethernet is 512 bits x 11 1000 us, which is equal to 0.512 J.lS. The reduced slot time means that collision is detected 100 times earlier. This means that the maximum length of the network is 25 m. This length may be suitable if all the stations are in one room, but it may not even be long enough to connect the computers in one single office.

Carrier Extension

To allow for a longer network, we increase the minimum frame length. The carrier extension approach defines the minimum length of a frame as 512 bytes (4096 bits). This means that the minimum length is 8 times longer. This method forces a station to add extension bits (padding) to any frame that is less than 4096 bits. In this way, the maximum length of the network can be increased 8 times to a length of 200 m. This allows a length of 100 m from the hub to the station.

Frame Bursting

Carrier extension is very inefficient if we have a series of short frames to send; each frame arries redundant data. To improve efficiency, frame bursting was proposed. Instead of adding an extension to each frame, multiple frames are sent. However, to make these multiple frames look like one frame, padding is added between the frames (the same as that used for the carrier extension method) so that the channel is not idle. In other words, the method deceives other stations into thinking that a very large frame has been transmitted.

Physical Layer

The physical layer in Gigabit Ethernet is more complicated than that in Standard or Fast Ethernet. We briefly discuss some features of this layer.

Topology

Gigabit Ethernet is designed to connect two or more stations. If there are only two stations, they can be connected point-to-point. Three or more stations need to be connected in a star topology with a hub or a switch at the center. Another possible configuration is to connect several star topologies or let one star topology be part of another.

Implementation

Gigabit Ethernet can be categorized as either a two-wire or a four-wire implementation. The two- wire implementations use fiber-optic cable (l000Base-SX, short-wave, or l000Base-LX, long-wave), or STP (l000Base-CX). The four-wire version uses category 5 twisted-pair cable (l000Base-T). In other words, we have four implementations. 1000Base-T was designed in response to those users who had already installed this wiring for other purposes such as Fast Ethernet or telephone services.

Encoding

Encoding in Gigabit Ethernet implementations

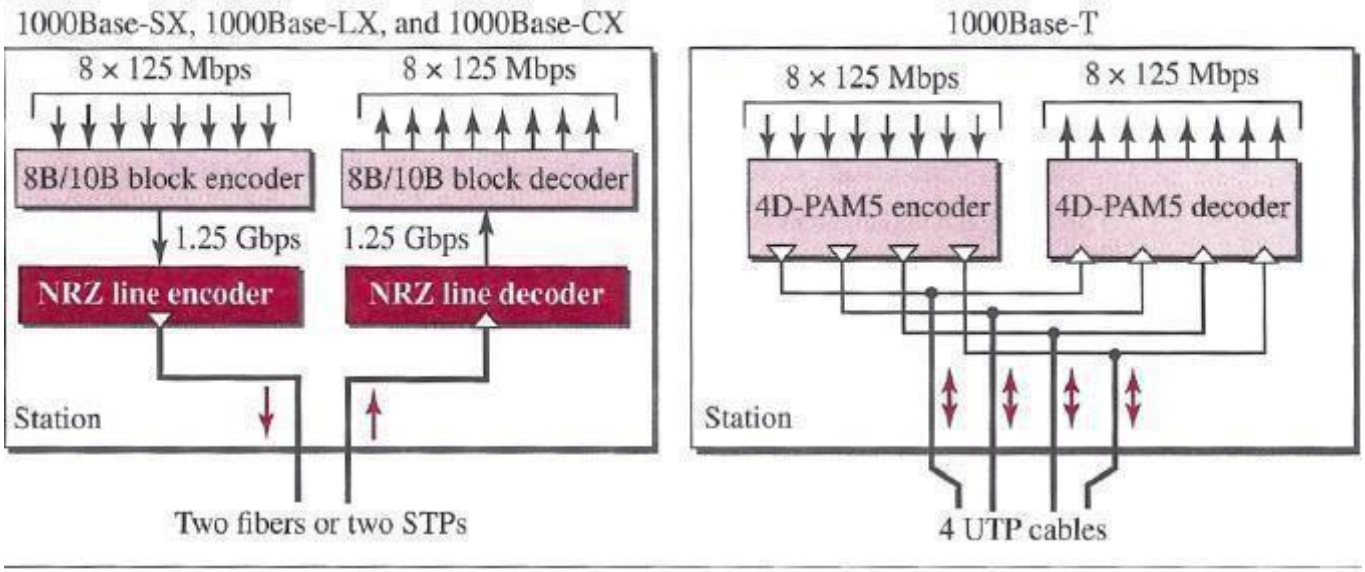


Figure shows the encoding/decoding schemes for the four implementations. Gigabit Ethernet cannot use the Manchester encoding scheme because it involves a very high bandwidth (2 GBaud). The two-wire implementations use an NRZ scheme, but NRZ does not self-synchronize properly. To synchronize bits, particularly at this high data rate, SBIl0B block encoding, discussed in Chapter 4, is used. This block encoding prevents long sequences of 0s or Is in the stream, but the resulting stream is 1.25 Gbps. Note that in this implementation, one wire (fiber or STP) is used for sending and one for receiving. In the four-wire implementation it is not possible to have 2 wires for input and 2 for output, because each wire would need to carry 500 Mbps, which exceeds the capacity for category 5 UTP. Thus, all four wires are involved in both input and output; each wire carries 250 Mbps, which is in the range for category 5 UTP cable.

DATA LINK LAYER SWITCHING

Many organizations have multiple LANs and wish to connect them. Would it not be convenient if we could just join the LANs together to make a larger LAN? In fact, we can do this when the connections are made with devices called **bridges**. are a modern name for bridges; they provide functionality that goes beyond classic Ethernet and Ethernet hubs to make it easy to join multipleLANs into a larger and faster network.

We shall use the terms ‘‘bridge’’ and ‘‘switch’’ interchangeably. Bridges operate in the data link layer, so they examine the data link layer addresses to forward frames. Since they are not supposed to examine the payload field of the frames they forward, they can handle IP packets as well as other kinds of packets, such as AppleTalk packets. In contrast, *routers* examine the addresses in packets and route based on them, so they only work with the protocols that they were designed to handle. physical LANs into a single logical LAN. We will also look at how to do the reverse and treat one physical LAN as multiple logical LANs, called **VLANs** (**Virtual LANs**). Both technologies provide useful flexibility for managing networks. For a comprehensive treatment of bridges, switches, and related topics, see Seifert and Edwards (2008)and Perlman (2000).

USES OF BRIDGES

Before getting into the technology of bridges, let us take a look at some common situations in which bridges are used. We will mention three reasons why a single organization may end up with multiple LANs.

First, many university and corporate departments have their own LANs to connect their own personal computers, servers, and devices such as printers. Since the goals of the various departments differ, different departments may set up different LANs, without regard to what other departments are doing. Sooner or later, though, there is a need for interaction, so bridges are needed. In this example, multiple LANs come into existence due to the autonomy of their owners.

Second, the organization may be geographically spread over several buildings separated by considerable distances. It may be cheaper to have separate LANs in each building and connect them with bridges and a few long-distance fiber optic links than to run all the cables to a single central switch.

Even if laying the cables is easy to do, there are limits on their lengths (e.g., 200 m for twisted-pair gigabit Ethernet). The network would not work for longer cables due to the excessive signal attenuation or round-trip delay. The only solution is to partition the LAN and install bridges to join the pieces to increase the total physical distance that can be covered.

Third, it may be necessary to split what is logically a single LAN into separate LANs (connected by bridges) to accommodate the load. At many large universities, for example, thousands of workstations are available for student and faculty computing. Companies may also have thousands of employees. The scale of this system precludes putting all the workstations on a single LAN—there are more computers than ports on any Ethernet hub and more stations than allowed on a single classic Ethernet.

Even if it were possible to wire all the workstations together, putting more stations on an Ethernet hub or classic Ethernet would not add capacity. All of the stations share the same, fixed amount of bandwidth. The more stations there are, the less average bandwidth per station. However, two separate LANs have twice the capacity of a single LAN. Bridges let the LANs be joined together while keeping this capacity. The key is not to send traffic onto ports where it is not needed, so that each LAN can run at full speed. This behavior also increases reliability, since on a single LAN a defective node that keeps outputting a continuous stream of garbage can clog up the entire LAN.

By deciding what to forward and what not to forward, bridges act like fire doors in a building, preventing a single node that has gone berserk from bringing down the entire system. To make these benefits easily available, ideally bridges should be completely transparent. It should be possible to go out and buy bridges, plug the LAN cables into the bridges, and have everything work perfectly, instantly.

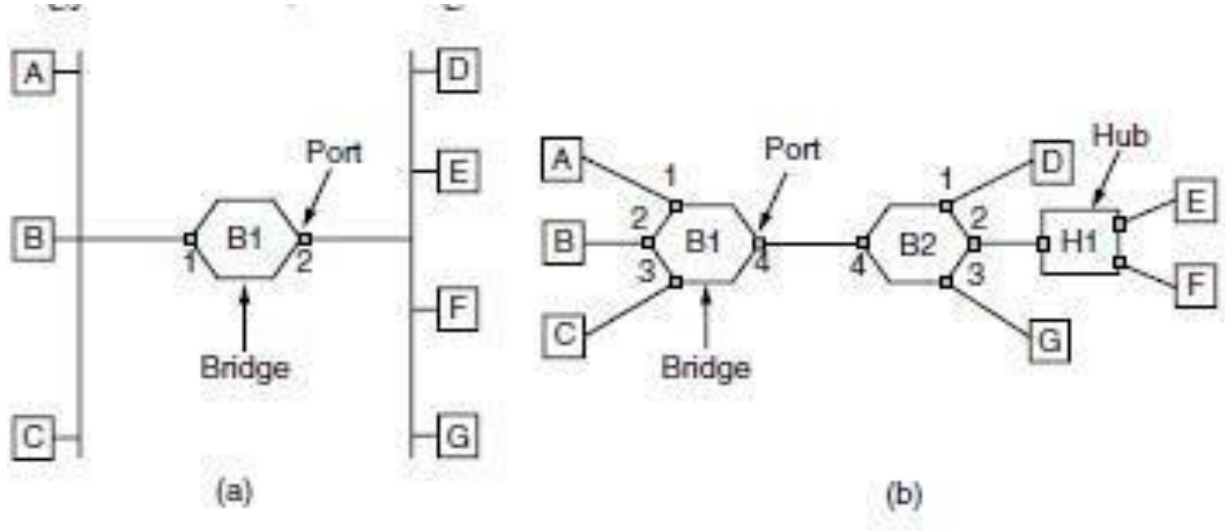
There should be no hardware changes required, no software changes required, no setting of address switches, no downloading of routing tables or parameters, nothing at all. Just plug in the cables and walk away. Furthermore, the operation of the existing LANs should not be affected by the bridges at all. As far as the stations are concerned, there should be no observable difference whether or not they are part of a bridged LAN. It should be as easy to move stations around the bridged LAN as it is to move them around a single LAN.

Surprisingly enough, it is actually possible to create bridges that are transparent. Two algorithms are used: a backward learning algorithm to stop traffic being sent where it is not needed; and a spanning tree algorithm to break loops that may be formed when switches are cabled together willy-nilly.

LEARNING BRIDGES

The topology of two LANs bridged together is shown in figure. On the left-hand side, two multidrop LANs, such as classic Ethernets, are joined by a special station—the bridge—that sits on both LANs. On the right-hand side, LANs with point-to- point cables, including one hub, are joined together. The bridges are the devices to which the stations and hub are attached. If the LAN technology is Ethernet, the bridges are better known as Ethernet switches.

Bridges were developed when classic Ethernets were in use, so they are often shown in topologies with multidrop cables, as in Fig. 4-41(a). However, all the topologies that are encountered today are comprised of point-to-point cables and switches. The bridges work the same way in both settings. All of the stations attached to the same port on a bridge belong to the same collision domain, and this is different than the collision domain for other ports. If there is more than one station, as in a classic Ethernet, a hub, or a half-duplex link, the CSMA/CD protocol is used to send frames.



(**a) Bridge connecting two multidrop LANs. (b) Bridges (and a hub) connecting seven point-to-point stations**

There is a difference, however, in how the bridged LANs are built. To bridge multidrop LANs, a bridge is added as a new station on each of the multidrop LANs, as in Fig. 4-41(a). To bridge point-to-point LANs, the hubs are either connected to a bridge or, preferably, replaced with a bridge to increase performance. In Fig. bridges have replaced all but one hub. Different kinds of cables can also be attached to one bridge. For example, the cable connecting bridge *B1* to bridge *B2* in Fig. might be a long-distance fiber optic link, while the cable connecting the bridges to stations might be a short-haul twisted-pair line. This arrangement is useful for bridging LANs in different buildings.

Now let us consider what happens inside the bridges. Each bridge operates in promiscuous mode, that is, it accepts every frame transmitted by the stations attached to each of its ports. The bridge must decide whether to forward or discard each frame, and, if the former, on which port to output the frame. This decision is made by using the destination address. As an example, consider the topology of Fig. If station *A* sends a frame to station *B*, bridge *B1* will receive the frame on port 1. This frame can be immediately discarded without further ado because it is already on the correct port. However, in the topology of Fig. suppose that *A* sends a frame to *D*. Bridge *B1* will receive the frame on port 1 and output it on port 4. Bridge *B2* will then receive the frame on its port 4 and output it on its port 1.

A simple way to implement this scheme is to have a big (hash) table inside the bridge. The table can list each possible destination and which output port it belongs on. For example, in Fig, the table at *B1* would list *D* as belonging to port 4, since all *B1* has to know is which port to put frames on to reach *D*. That, in fact, more forwarding will happen later when the frame hits *B2* is not of interest to *B1*..

As mentioned above, the bridges operate in promiscuous mode, so they see every frame sent on any of their ports. By looking at the source addresses, they can tell which machines are accessible on which ports. For example, if bridge *B1* in Fig sees a frame on port 3 coming from *C*, it knows that *C* must be reachable via port 3, so it makes an entry in its hash table. Any subsequent frame addressed to *C* coming in to *B1* on any other port will be forwarded to port 3.The topology can change as machines and bridges are powered up and down and moved around. To handle dynamic topologies, whenever a hash table entry is made, the arrival time of the frame is noted in the entry.

Whenever a frame whose source is already in the table arrives, its entry is updated with the current time. Thus, the time associated with every entry tells the last time a frame from that machine was seen. Periodically, a process in the bridge scans the hash table and purges all entries more than a few minutes old. In this way, if a computer is unplugged from its LAN, moved around the building, and plugged in again somewhere else, within a few minutes it will be back in normal operation, without any manual intervention. This algorithm also means that if a machine is quiet for a few minutes, any traffic sent to it will have to be flooded until it next sends a frame itself. The routing procedure for an incoming frame depends on the port it arrives on (the source port) and the address to which it is destined (the destination address).

The procedure is as follows:

1. If the port for the destination address is the same as the source port, discard the frame.
2. If the port for the destination address and the source port are different, forward the frame on tothe destination port.
3. If the destination port is unknown, use flooding and send the frame on all ports except thesource port.

An example is shown in Fig. 4-41(b) where stations *E* and *F* are connected to hub *H*1, which is in turn connected to bridge *B2*. If *E* sends a frame to *F*, the hub will relay it to *B2* as well as to *F*. That is what hubs do—they wire all ports together so that a frame input on one port is simply output on all other ports. The frame will arrive at *B2* on port 4, which is already the right output port to reach the destination. Bridge *B2* need only discard the frame. As each frame arrives, this algorithm must beapplied, so it is usually implemented with special-purpose VLSI chips.

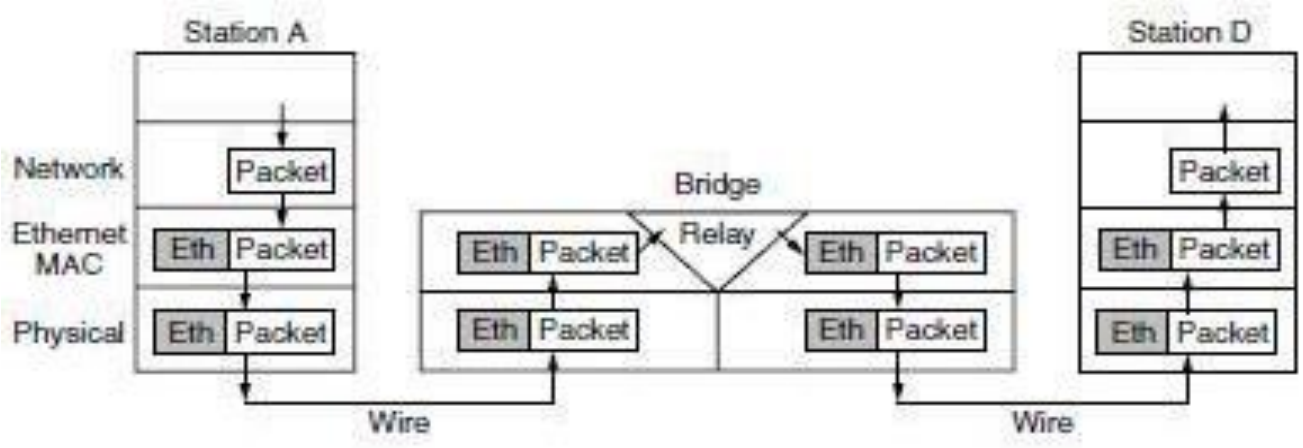


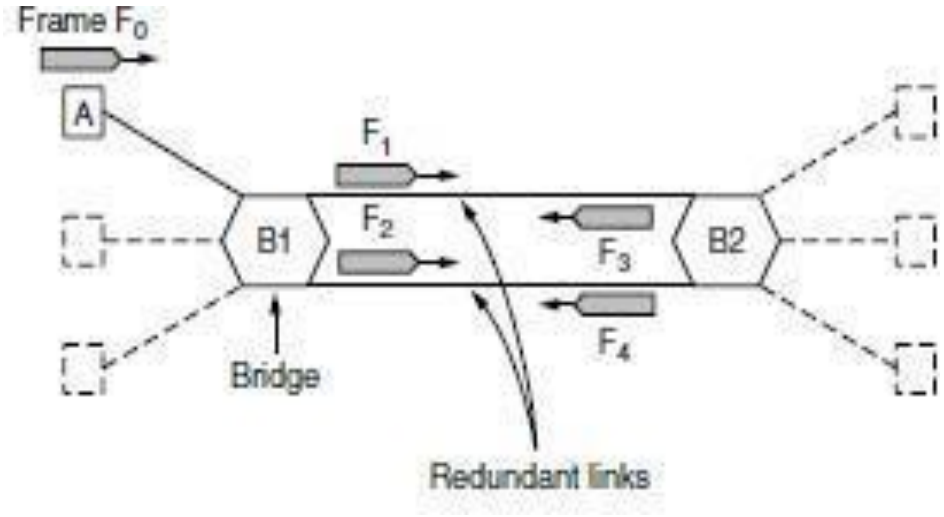
Figure 4-42. Protocol processing at a bridge.

The chips do the lookup and update the table entry, all in a few microseconds. Because bridges only look at the MAC addresses to decide how to forward frames, it is possible to start forwarding as soon as the destination header field has come in, before the rest of the frame has arrived. This design reduces the latency of passing through the bridge, as well as the number of frames that the bridge must be able to buffer. It is referred to as cut-through switching or wormhole routing and is usually handled in hardware.

It acquires an Ethernet header (and also a trailer, not shown in the figure). This unit is passed to the physical layer, goes out over the cable, and is picked up by the bridge. In the bridge, the frame is passed up from the physical layer to the Ethernet MAC layer. This layer has extended processing compared to the Ethernet MAC layer at a station. It passes the frame to a relay, still within the MAC layer. The bridge relay function uses only the Ethernet MAC header to determine how to handle the frame. In this case, it passes the frame to the Ethernet MAC layer of the port used to reach station *D*, and the frame continues on its way. In the general case, relays at a given layer can rewrite the headers for that layer. VLANs will provide an example shortly. In no case should the bridge look inside the frame and learn that it is carrying an IP packet; that is irrelevant to the bridge processing and would violate protocol layering. Also note that a bridge with *k* ports will have *k* instances of MAC and physical layers. The value of *k* is 2 for our simple example.

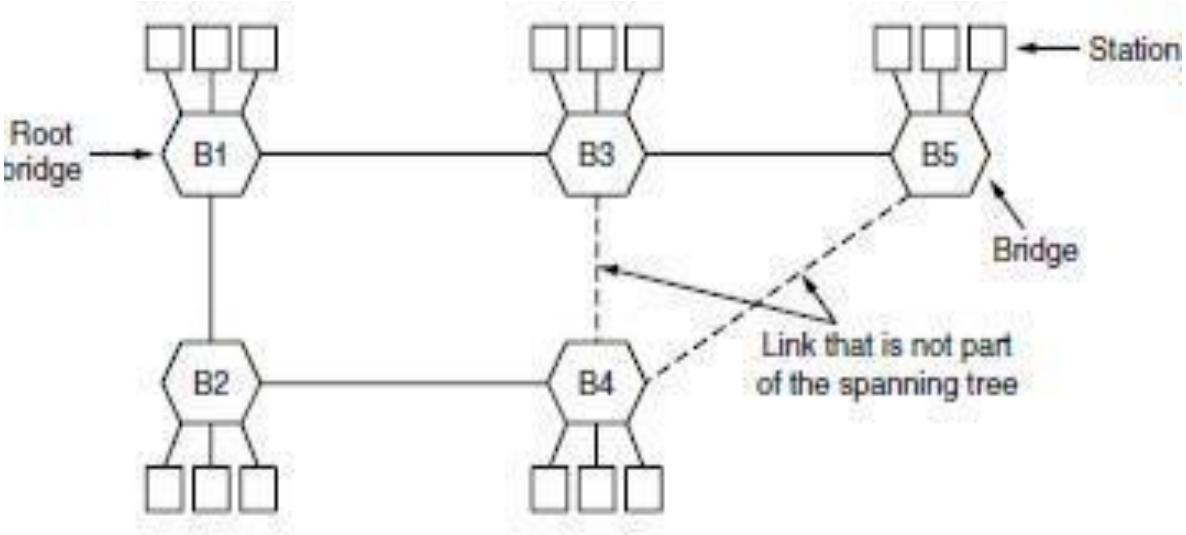
SPANNING TREE BRIDGES

To increase reliability, redundant links can be used between bridges. In the example of Fig, there are two links in parallel between a pair of bridges. This design ensures that if one link is cut, the network will not be partitioned into two sets of computers that cannot talk to each other.



However, this redundancy introduces some additional problems because it creates loops in the topology. An example of these problems can be seen by looking at how a frame sent by *A* to a previously unobserved destination is handled in Fig..

Each bridge follows the normal rule for handling unknown destinations, which is to flood the frame. Call the frame from *A* that reaches bridge *B1* frame *F*0. The bridge sends copies of this frame out all of its other ports. We will only consider the bridge ports that connect *B1* to *B2* (though the frame will be sent out the other ports, too). Since there are two links from *B1* to *B2*, two copies of the frame will reach *B2*. They are shown in Fig as *F*1 and *F*2. Shortly thereafter, bridge *B2* receives these frames. However, it does not (and cannot) know that they are copies of the same frame, rather than two different frames sent one after the other. So bridge *B2* takes *F*1 and sends copies of it out all the other ports, and it also takes *F*2 and sends copies of it out all the other ports. This produces frames *F*3 and *F*4 that are sent along the two links back to *B1*. Bridge *B1* then sees two new frames with unknown destinations and copies them again. This cycle goes on forever. The solution to this difficulty is for the bridges to communicate with each other and overlay the actual topology with a spanning tree that reaches every bridge. In effect, some potential connections between bridges are ignored in the interest of constructing a fictitious loop-free topology that is a subset of the actual topology.



A spanning tree connecting five bridges. The dashed lines are links that are not part of the spanning tree.

For example, in Fig. we see five bridges that are interconnected and also have stations connected to them. Each station connects to only one bridge. There are some redundant connections between the bridges so that frames will be forwarded in loops if all of the links are used. This topology can be thought of as a graph in which the bridges are the nodes and the point-to-point links are the edges.

The graph can be reduced to a spanning tree, which has no cycles by definition, by dropping the links shown as dashed lines in Fig. Using this spanning tree, there is exactly one path from every station to every other station. Once the bridges have agreed on the spanning tree, all forwarding between stations follows the spanning tree. Since there is a unique path from each source to each destination, loops are impossible.

To build the spanning tree, the bridges run a distributed algorithm. Each bridge periodically broadcasts a configuration message out all of its ports to it neighbors and processes the messages it receives from other bridges, as described next. These messages are not forwarded, since their purpose is to build the tree, which can then be used for forwarding. The bridges must first choose one bridge to be the root of the spanning tree. To make this choice, they each include an identifier based on their MAC address in the configuration message, as well as the identifier of the bridge they believe to be the root. MAC addresses are installed by the manufacturer and guaranteed to be unique worldwide, which makes these identifiers convenient and unique. The bridges choose the bridge with the lowest identifier to be the root. After enough messages have been exchanged to spread the news, all bridges will agree on which bridge is the root. In Fig., bridge *B1* has the lowest identifier and becomes the root. Next, a tree of shortest paths from the root to every bridge is constructed.

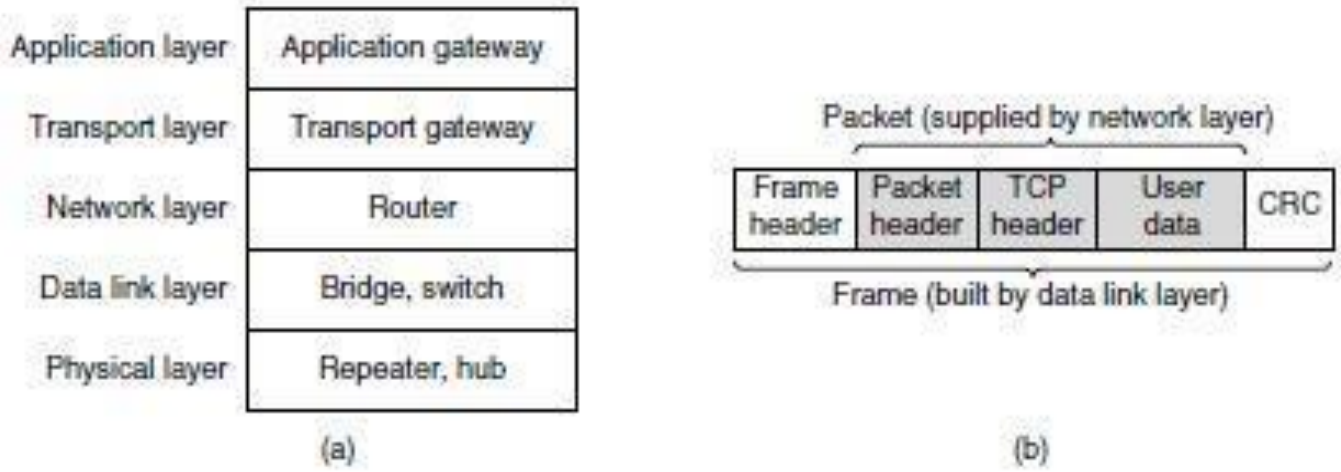
In Fig. bridges *B2* and *B3* can each be reached from bridge *B1* directly, in one hop that is a shortest path. Bridge *B4* can be reached in two hops, via either *B2* or *B3*. To break this tie, the path via the bridge with the lowest identifier is chosen, so *B4* is reached via *B2*. Bridge *B5* can be reached in two hops via *B3*. To find these shortest paths, bridges include the distance from the root in their configuration messages. Each bridge remembers the shortest path it finds to the root. The bridges then turn off ports that are not part of the shortest path. Although the tree spans all the bridges, not all the links (or even bridges) are necessarily present in the tree. This happens because turning off the ports prunes some links from the network to prevent loops. Even after the spanning tree has been established, the algorithm continues to run during normal operation to automatically detect topology changes and updatethe tree.

The algorithm for constructing the spanning tree was invented by Radia Perlman. Her job was to solve the problem of joining LANs without loops. She was given a week to do it, but she came up with the idea for the spanning tree algorithm in a day. Fortunately, this left her enough time towrite it as a poem (Perlman, 1985):

The spanning tree algorithm was then standardized as IEEE 802.1D and used for many years. In 2001, it was revised to more rapidly find a new spanning tree after a topology change. For a detailed treatment of bridges, see Perlman (2000).

REPEATERS, HUBS, BRIDGES, SWITCHES, ROUTERS, AND GATEWAYS

The key to understanding these devices is to realize that they operate in different layers, as illustrated in Fig. 4-45(a).



(a) Which device is in which layer. (b) Frames, packets, and headers.

The layer matters because different devices use different pieces of information to decide how to switch. In a typical scenario, the user generates some data to be sent to a remote machine. Those data are passed to the transport layer, which then adds a header (for example, a TCP header) and passes the resulting unit down to the network layer.

The network layer adds its own header to form a network layer packet (e.g., an IP packet). In Fig. we see the IP packet shaded in gray. Then the packet goes to the data link layer, which adds its own header and checksum (CRC) and gives the resulting frame to the physical layer for transmission, for example, over a LAN.

Now let us look at the switching devices and see how they relate to the packets and frames. At the bottom, in the physical layer, we find the repeaters. These are analog devices that work with signals on the cables to which they are connected. A signal appearing on one cable is cleaned up, amplified, and put out on another cable. Repeaters do not understand frames, packets, or headers. They understand the symbols that encode bits as volts.

Classic Ethernet, for example, was designed to allow four repeaters that would boost the signal to extend the maximum cable length from 500 meters to 2500 meters. Next we come to the hubs. A hub has a number of input lines that it joins electrically. Frames arriving on any of the lines are sent out on all the others. If two frames arrive at the same time, they will collide, just as on a coaxial cable. All the lines coming into a hub must operate at the same speed. Hubs differ from repeaters in that they do not (usually) amplify the incoming signals and are designed for multiple input lines, but the differences are slight. Like repeaters, hubs are physical layer devices that do not examine the linklayer addresses or use them in any way.

Now let us move up to the data link layer, where we find bridges and switches. We just studied bridges at some length. A bridge connects two or more LANs. Like a hub, a modern bridge has multiple ports, usually enough for 4 to 48 input lines of a certain type.

Unlike in a hub, each port is isolated to be its own collision domain; if the port has a full-duplex point-to-point line, the CSMA/CD algorithm is not needed. When a frame arrives, the bridge extracts the destination address from the frame header and looks it up in a table to see where to send the frame. For Ethernet, this address is the 48-bit destination address shown in Fig. The bridge only outputs the frame on the port where it is needed and can forward multiple frames at the same time.

Bridges offer much better performance than hubs, and the isolation between bridge ports also means that the input lines may run at different speeds, possibly even with different network types. A common example is a bridge with ports that connect to 10-, 100-, and 1000-Mbps Ethernet. Buffering within the bridge is needed to accept a frame on one port and transmit the frame out on a different port. If frames come in faster than they can be retransmitted, the bridge may run out of buffer space and have to start discarding frames. For example, if a gigabit Ethernet is pouring bits into a 10-Mbps Ethernet at top speed, the bridge will have to buffer them, hoping not to run out of memory. This problem still exists even if all the ports run at the same speed because more than one port may be sending frames to a given destination port.

Bridges were originally intended to be able to join different kinds of LANs, for example, an Ethernet and a Token Ring LAN. However, this never worked well because of differences between the LANs. Different frame formats require copying and reformatting, which takes CPU time, requires a new checksum calculation, and introduces the possibility of undetected errors due to bad bits in the bridge's memory. Different maximum frame lengths are also a serious problem with no good solution. Basically, frames that are too large to be forwarded must be discarded. So much for transparency.

Two other areas where LANs can differ are security and quality of service. Some LANs have link-layer encryption, for example 802.11, and some do not, for example Ethernet. Some LANs have quality of service features such as priorities, for example 802.11, and some do not, for example Ethernet. Consequently, when a frame must travel between these LANs, the security or quality of service expected by the sender may not be able to be provided. For all of these reasons, modern bridges usually work for one network type, and routers, which we will come to soon, are used instead to join networks of different types. Switches are modern bridges by another name.

The differences are more to do with marketing than technical issues, but there are a few pointsworth knowing.

Bridges were developed when classic Ethernet was in use, so they tend to join relatively few LANs and thus have relatively few ports. The term ''switch'' is more popular nowadays. Also, modern installations all use point-to-point links, such as twisted-pair cables, so individual computers plug directly into a switch and thus the switch will tend to have many ports.

Finally, ‘‘switch’’ is also used as a general term. With a bridge, the functionality is clear. On the other hand, a switch may refer to an Ethernet switch or a completely different kind of device that makes forwarding decisions, such as a telephone switch. So far, we have seen repeaters and hubs, which are actually quite similar, as well as bridges and switches, which are even more similar to each other. Now we move up to routers, which are different from all of the above. When a packet comes into a router, the frame header and trailer are stripped off and the packet located in the frame’s payload field is passed to the routing software. This software uses the packet header to choose an output line. For an IP packet, the packet header will contain a 32-bit (IPv4) or 128-bit (IPv6) address, but not a 48-bit IEEE 802 address. The routing software does not see the frame addresses and does not even know whether the packet came in on a LAN or a point-to-point line.

Up another layer, we find transport gateways. These connect two computers that use different connection-oriented transport protocols. For example, suppose a computer using the connection- oriented TCP/IP protocol needs to talk to a computer using a different connection-oriented transport protocol called SCTP. The transport gateway can copy the packets from one connectionto the other, reformatting them as need be.

Finally, application gateways understand the format and contents of the data and can translate messages from one format to another. An email gateway could translate Internet messages into SMS messages for mobile phones, for example. Like ‘‘switch,’’ ‘‘gateway’’ is somewhat of a general term. It refers to a forwarding process that runs at a high layer.