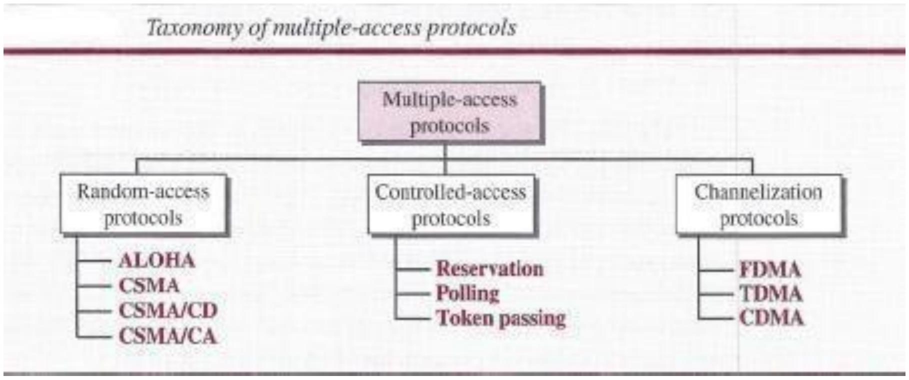
UNIT- III
Multiple Access Protocols

Multiple Access Protocols

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.
Many protocols have been devised to handle access to a shared link. All of these protocols belong to a sub layer in the data-link layer called *media access control (MAC).* We categorize them into three groups:



Taxonomy of multiple-access protocols

➢ 1 The first section discusses random-access protocols. Four protocols, ALOHA, CSMA, *CSMA/CD,* and *CSMA/CA,* are described in this section. These protocols are mostly used in LANs and WANs.

➢ 2 The second section discusses controlled-access protocols. Three protocols, reservation, polling, and token-passing, are described in this section. Some of these protocols are used in LANs, but others have some historical value.

➢ 3 The third section discusses channelization protocols. Three protocols, FDMA, TDMA, and CDMA are described in this section. These protocols are used in cellular telephony.

RANDOM ACCESS

In random-access or contention methods, no station is superior to another station and none is assigned control over another. 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 testing the state of the medium.

Two features give this method its name:

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.*

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.

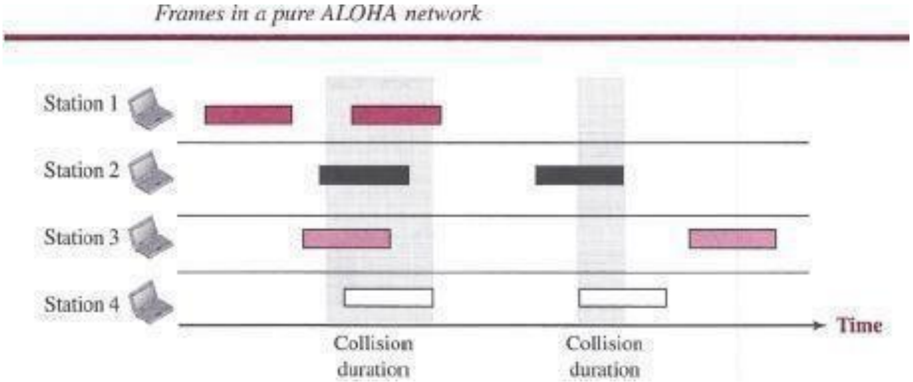
In a random-access method, each station has the right to the medium without being controlled by any other station. However, if more than one station tries to send, there is an access conflict-collision-and the frames will be either destroyed or modified.

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.

Pure ALOHA

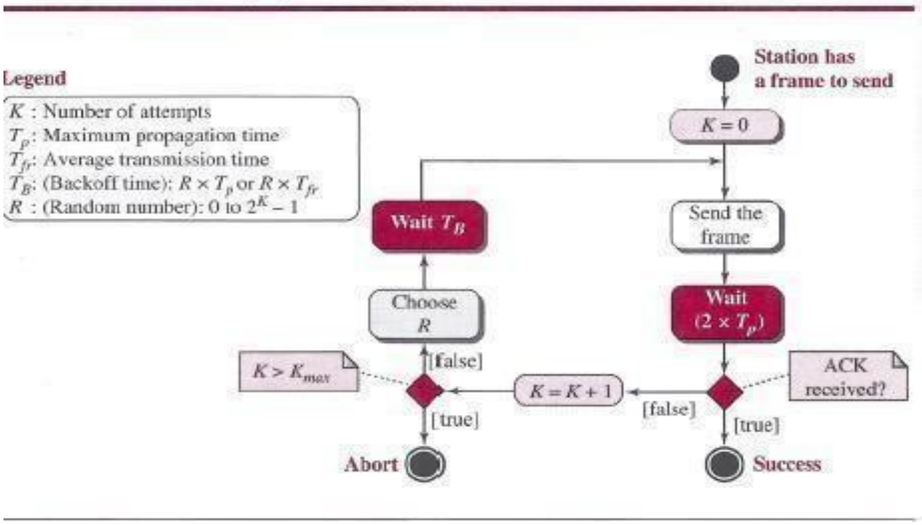
The original ALOHA protocol is called *pure ALOHA*. This is a simple but elegant protocol. The idea is that each station sends a frame whenever it has a frame to send (multiple access). However, since there is only one channel to share, there is the possibility of collision between frames from different stations. Below figure 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. Above Figure shows that only two frames survive: one frame from station 1 and one frame 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. 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 Ts.*

Pure ALOHA has a second method to prevent congesting the channel with retransmitted frames. After a maximum number of retransmission attempts *Kmax'* a station must give up and try later. 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 backoff time *Ts* is a random value that normally depends on *K* (the number of attempted unsuccessful transmissions). The formula for *Ts* depends on the implementation. One common formula is the *binary exponential backoff*. In this method, for each retransmission, a multiplier *R* = 0 to *2K* - 1 is randomly chosen and multiplied by *Tp*(maximum propagation time) or *Tfr* (the average time required to send out a frame) to find *Ts*. Note that in this rocedure, the range of the random numbers increases after each collision. The value of *Kmax* is usually chosen as 15.

Procedure for pure ALOHA protocol


Legend
K : Number of attempts
T_p: Maximum propagation time
T_{fr}: Average transmission time
T_B: (Backoff time): $R \times T_p$ or $R \times T_{fr}$
R : (Random number): 0 to $2^K - 1$
Station has a frame to send
$K = 0$
Send the frame
Wait ($2 \times T_p$)
ACK received?
[true]
Success
[false]
$K = K + 1$
$K > K_{max}$
[true]
Abort
[false]
Choose R
Wait T_B

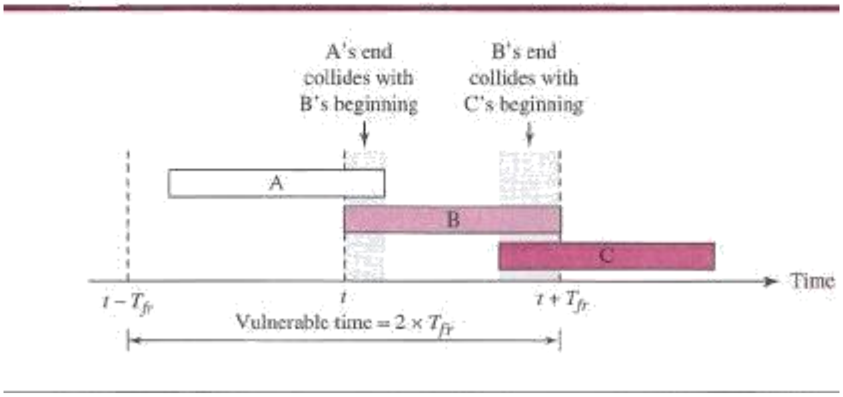
Example

The stations on a wireless ALOHA network is a maximum of 600 km apart. If we assume that signals propagate at 3 x J08 *mis,* we find *Tp* = (600 x J03) *I* (3 x J08) = 2 ms. For *K* = 2, the range of *R* is (O, 1, 2, 3). This means that *TB* can be 0, 2, 4, or 6 ms, based on the outcome of the random variable *R.*

Vulnerable time

Let us find the ***vulnerable time,*** the length of time in which there is a possibility of collision.We assumes that the stations send fixed-length frames with each frame taking *Tjr* seconds to send. Following figure shows the vulnerable time for station B.

Vulnerable time for pure ALOHA protocol



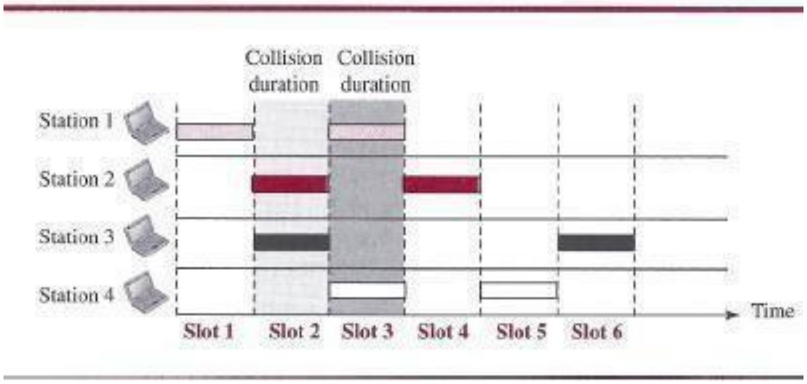
Station B starts to send a frame at time *t*. Now imagine station A has started to send its frame after *t* - *TjT'* This leads to a collision between the frames from station Band station A. On the other hand, suppose that station C starts to send a frame before time *t* + *Tjr*- Here, there is also a collision between frames from station B and station C. Looking at Figure 12.4, we see that the vulnerable time during which a collision may occur in pure ALOHA is 2 times the frame transmission time.

Pure ALOHA vulnerable time = 2 x T_{fr}

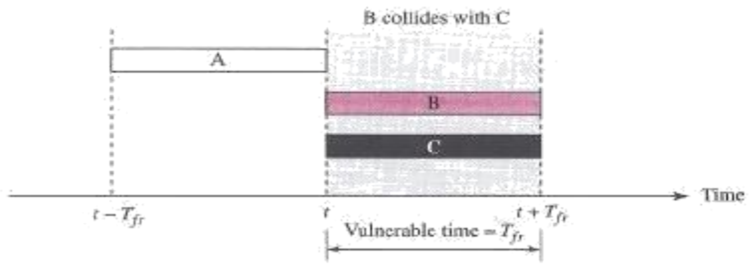
Slotted ALOHA

Pure ALOHA has a vulnerable time of 2 x T_p This is so because there is no rule that defines when the station can send. A station may send soon after another station has started or just before another station has finished. Slotted ALOHA was invented to improve the efficiency of pure ALOHA. In slotted ALOHA we divide the time into slots of T_{fr} seconds and force the station to send only at the beginning of the time slot. Below figure shows an example of frame collisions in slotted ALOHA.

Frames in a slotted ALOHA network



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. However, the vulnerable time is now reduced to one-half, equal to *Tjr. Below f*igure shows the situation.



Vulnerable time for slotted ALOHA protocol

Slotted ALOHA vulnerable time = *Tlr*

Throughput

It can be proven that the average number of successful transmissions for slotted ALOHA is S = G x e^{-G} .The maximum throughput *Smax* is 0.368, when G = 1. In other words, if one frame is generated during one frame transmission time, then 36.8 percent of these frames reach their destination successfully. We expect G = 1 to produce maximum throughput because the vulnerable time is equal to the frame transmission time. Therefore, if a station generates only one frame in this vulnerable time (and no other station generates a frame during this time), the frame will reach its destination successfully.

The throughput for slotted ALOHA is S = G x e^{-G}
The maximum throughput S_{max} = 0.368 when G = 1.

Example

A slotted ALOHA network transmits 200-bit frames using a shared channel with a 200-kbps bandwidth. Find the throughput if the system (all stations together) produces
a. 1000 frames per second.
b. 500 frames per second.
c. 250 frames per second.

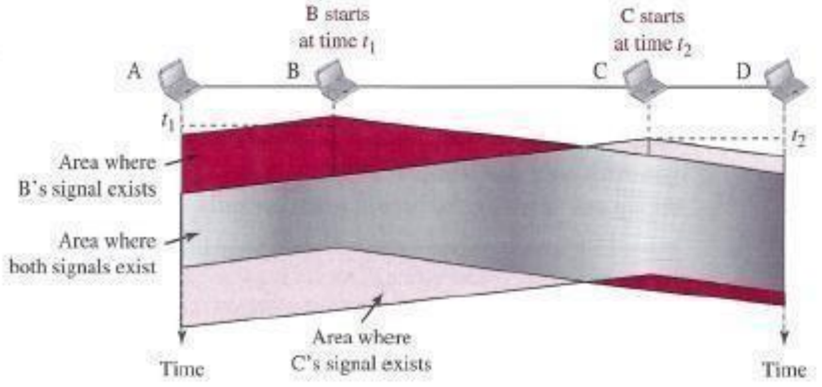
Solution

This situation is similar to the previous exercise except that the network is using slotted ALOHA instead of pure ALOHA. The frame transmission time is *200/200* kbps or 1 ms.
a. In this case G is 1. So S = G x *e-G* = 0.368 (36.8 percent). This means that the throughput is 1000 x 0.0368 = 368 frames. Only 368 out of 1000 frames will probably survive. Note that this is the maximum throughput case, percentagewise.
b. Here G is 112. In this case S = G x *e-G* = 0.303 (30.3 percent). This means that the throughput is 500 x 0.0303 = 151. Only 151 frames out of 500 will probably survive. c. Now G is 114. In this case S = G x *e-G* = 0.195 (19.5 percent). This means that the throughput is 250 x 0.195 = 49. Only 49 frames out of 250 will probably survive.

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. The reason for this is shown in below figure, a space and time model of a CSMA network. Stations are connected to a shared channel (usually a dedicated medium). The possibility of collision still exists because of propagation delay; when a station sends a frame, it still takes time (although very short) for the first bit to reach every station and for every station to sense it. In other words, a station may sense the medium and find it idle, only because the first bit sent by another station has not yet been received.

Space/time model of a collision in CSMA

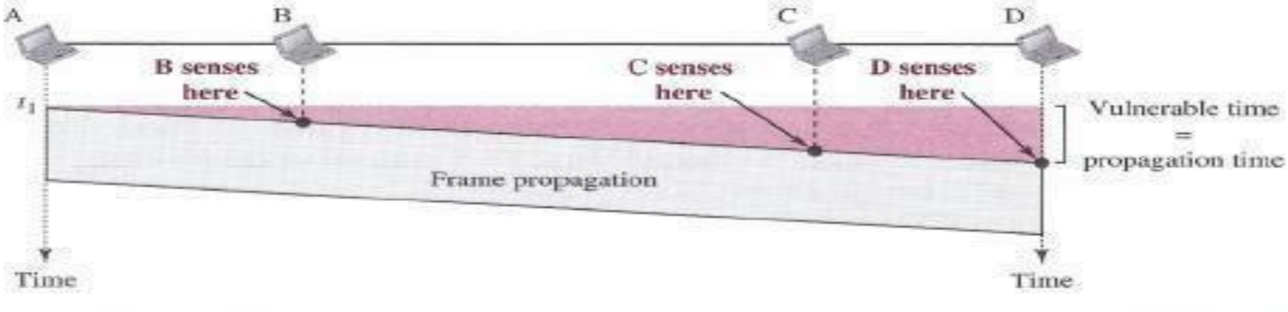


At time fl, station B senses the medium and finds it idle, so it sends a frame. At time *t2 (t2 > tl)*, station C senses the medium and finds it idle because, at this time, the first bits from station B have not reached station C. Station C also sends a frame. The two signals collide and both frames are destroyed.

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 bi t of the frame reaches the end of the medium, every station will already have heard the bit and will refrain from sending. Below Figure shows the worst case. The leftmost station, A, sends a frame at time *fl,* which reaches the rightmost station, D, at time *tl* + *Tp.* The gray area shows the vulnerable area in time and space.

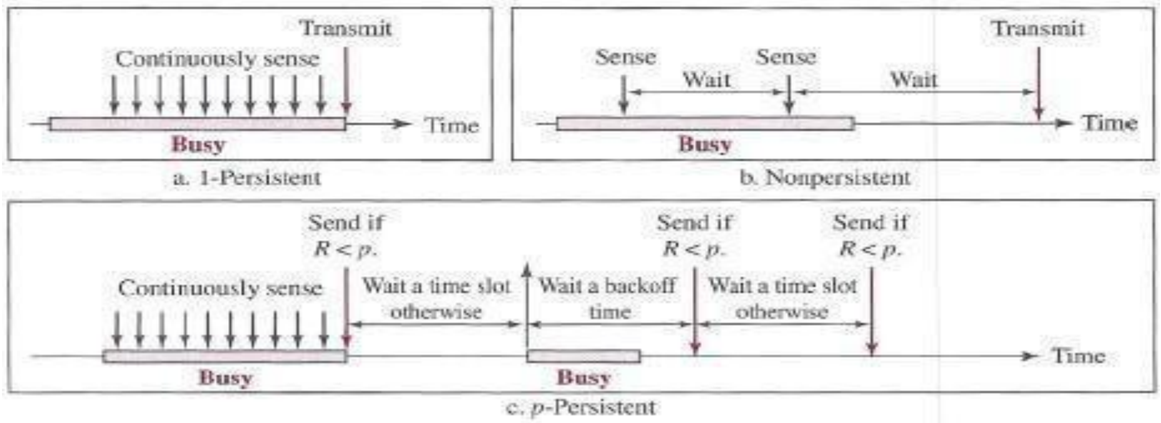
Vulnerable time in CSMA



Persistence Methods

What should a station do if the channel is busy? What should a station do if the channel is idle? Three methods have been devised to answer these questions: the I-persistent method, the non persistent method, and the p-persistent method. Below Figure shows the behavior of three persistence methods when a station finds a channel busy.

Behavior of three persistence methods



a. 1-Persistent

b. Nonpersistent

c. *p*-Persistent

Above Figure shows the flow diagrams for these methods.

I- Persistent

The *l-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. We will see later that Ethernet uses this method.

Non persistent

In the *nonpersistent 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 nonpersistent 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.

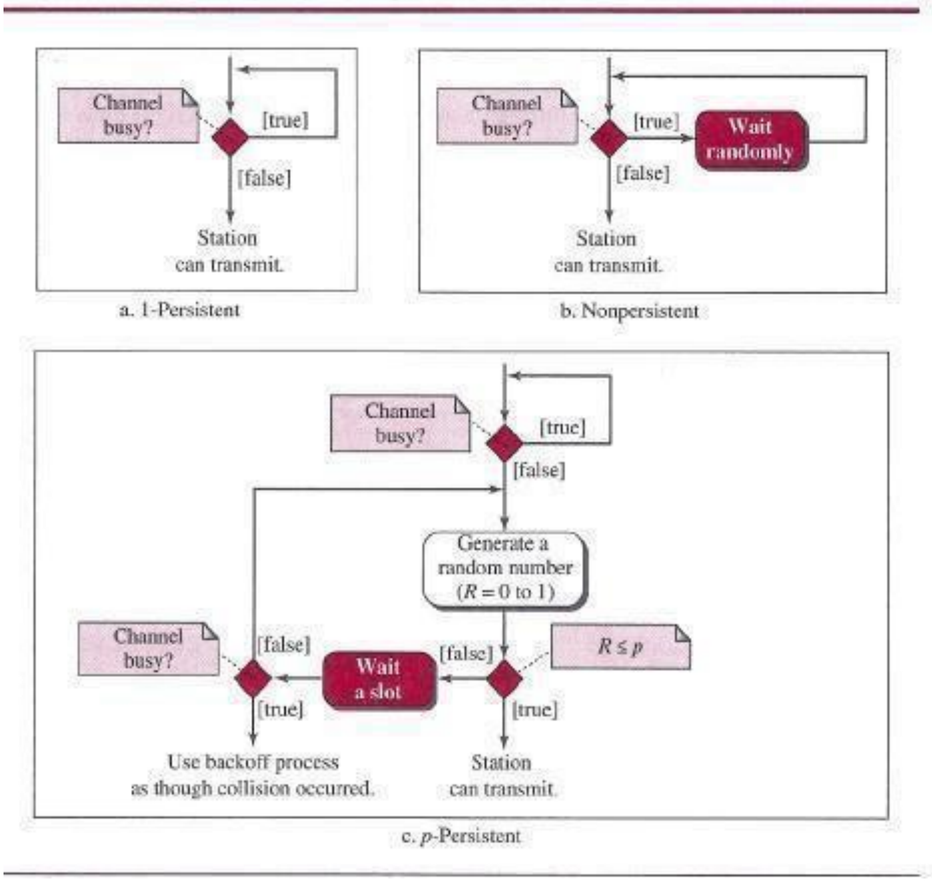
P- Persistent

The *p-persistent method* is used if the channel has time slots with a 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 1.
   b. If the line is busy, it acts as though a collision has occurred and uses the back off procedure.

Flow diagram for three persistence methods



a. 1-Persistent

b. Nonpersistent

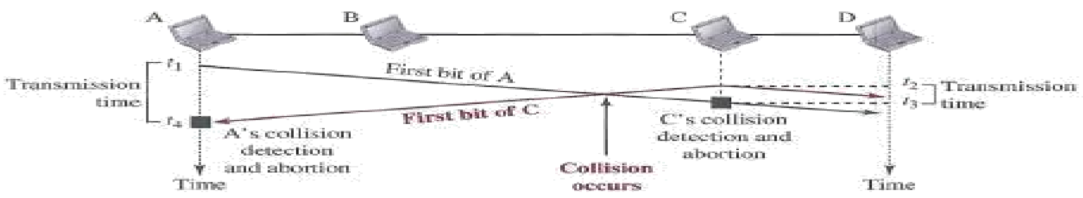
c. *p*-Persistent

CSMA/CD

The CSMA method does not specify the procedure following a collision. Carrier sense multiple access with collision detection *(CSMA/CD)* augments the algorithm to handle the collision. 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. If, however, there is a collision, the frame is sent again.

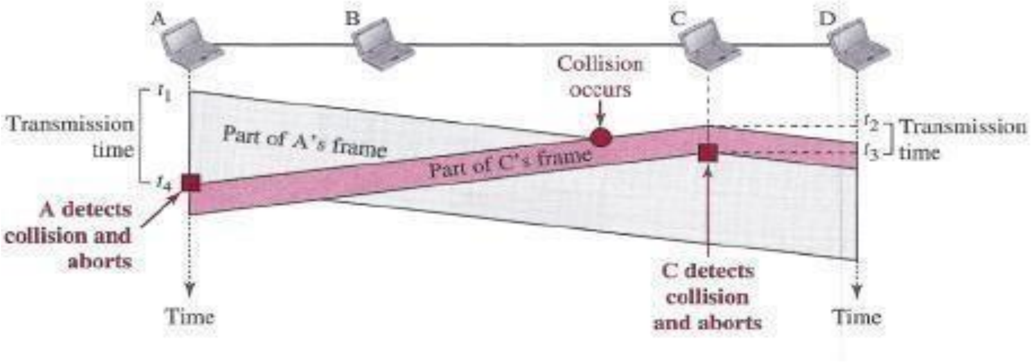
To better understand *CSMA/CD,* let us look at the first bits transmitted by the two stations involved in the collision. Although each station continues to send bits in the frame until it detects the collision, we show what happens as the first bits collide. In below figure, stations A and Care involved in the collision.

Collision of the first bits in CSMA/CD



At time *t*1, station A has executed its persistence procedure and starts sending the bits of its frame. At time *t2,* station C has not yet sensed the first bit sent by A. Station C executes its persistence procedure and starts sending the bits in its frame, which propagate both to the left and to the right. The collision occurs sometime after time *t2'* Station C detects a collision at time *t3* when it receives the first bit of A's frame. Station C immediately (or after a short time, but we assume immediately) aborts transmission. Station A detects collision at time *t4* when it receives the first bit of C's frame; it also immediately aborts transmission. Looking at the figure, we see that A transmits for the duration *t4 - t1;* C transmits for the duration *t3 - t2'* Now that we know the time durations for the two transmissions, we can show a more complete graph in below figure.

Collision and abortion in CSMA/CD



Minimum Frame Size

For *CSMAJCD* to work, we need a restriction on the frame size. Before sending the last bit of the frame, the sending station must detect a collision, if any, and abort the transmission. This is so because the station, once the entire frame is sent, does not keep a copy of the frame and does not monitor the line for collision detection. Therefore, the frame transmission time *Tfr* must be at least two times the maximum propagation time *Tp.* To understand the reason, let us think about the worst-case scenario. If the two stations involved in a collision are the maximum distance apart, the signal from the first takes time *Tp* to reach the second, and the effect of the collision takes another time *Tp* to reach the first. So the requirement is that the first station must still be transmitting after *2Tp.*

Example

A network using *CSMA/CD* has a bandwidth of 10 Mbps. If the maximum propagation time (including the delays in the devices and ignoring the time needed to send a jamming signal, as we see later) is 25.6I1s, what is the minimum size of the frame?

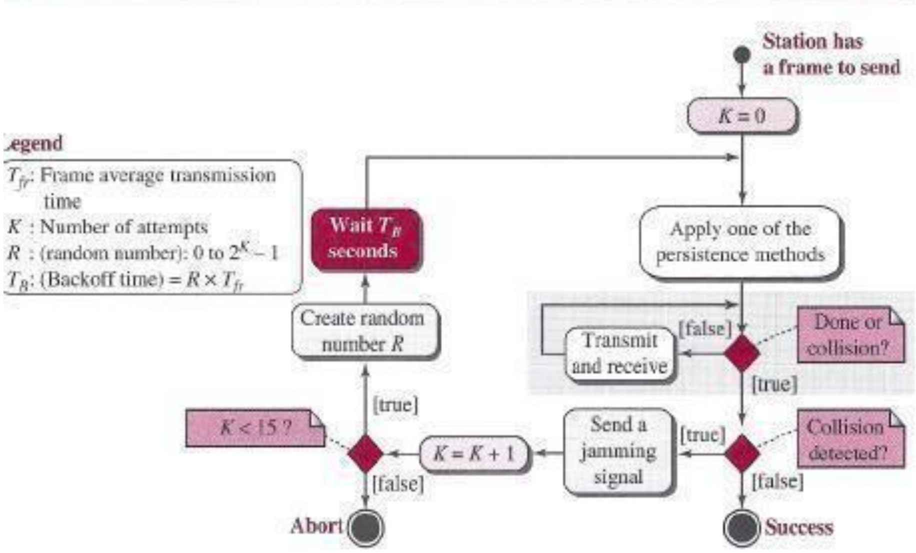
Solution

The minimum frame transmission time is *Tfr* = 2 x *Tp* = 51.2 I1s.This means, in the worst case, a station needs to transmit for a period of 51.2 I1sto detect the collision. The minimum size of the frame is 10Mbps x 51.211s= 512 bits or 64 bytes. This is actually the minimum size of the frame for Standard Ethernet.

Procedure

Now let us look at the flow diagram for *CSMA/CD* in Figure. It is similar to the one for the ALOHA protocol, but there are differences.

Flow diagram for the CSMA/CD

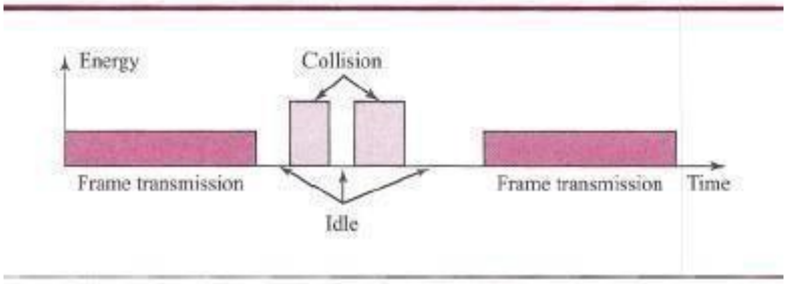


The first difference is the addition of the persistence process. We need to sense the channel before we start sending the frame by using one of the persistence processes we discussed previously (non persistent, l-persistent, or p-persistent). The second difference is the frame transmission. In ALOHA, we first transmit the entire frame and then wait for an acknowledgment. In *CSMA/CD*, transmission and collision detection are continuous processes. 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 or a bidirectional port). We use a loop to show that transmission is a continuous process. 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. The third difference is the sending of a short jamming signal to make sure that all other stations become aware of the collision.

Energy Level

We can say that the level of energy in a channel can have three values: zero, normal, and abnormal. At the zero level, the channel is idle. At the normal level, a station has successfully captured the channel and is sending its frame. At the abnormal level, there is a collision and the level of the energy is twice the normal level. A station that has a frame to send or is sending a frame needs to monitor the energy level to determine if the channel is idle, busy, or in collision mode. Below figure shows the situation.

Energy level during transmission, idleness, or collision



Throughput

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. For the l-persistent method, the maximum throughput is around 50 percent when G = 1. For the non persistent method, the maximum throughput can go up to 90 percent when G is between 3 and 8.

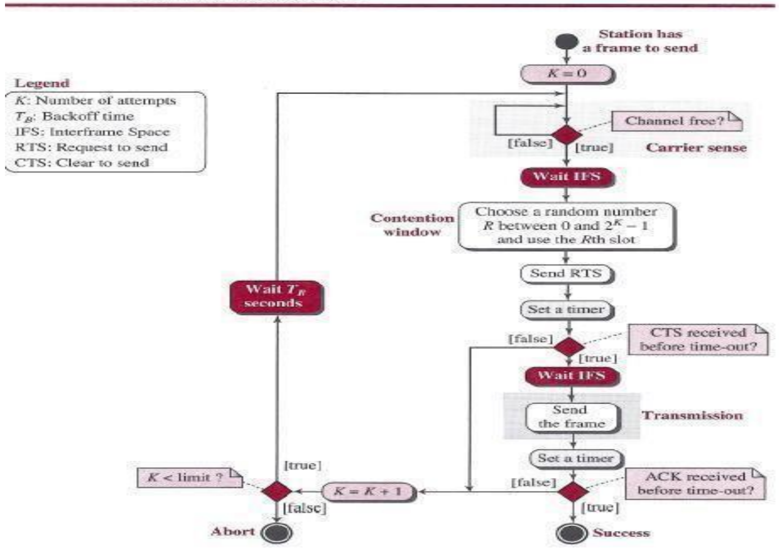
Traditional Ethernet

One of the LAN protocols that used *CSMA/CD* is the traditional Ethernet with the data rate of 10 Mbps. The traditional Ethernet was a broadcast LAN that used the l-persistence method to control access to the common media. Later versions of Ethernet try to move from *CSMA/CD* access methods.

CSMA/CA

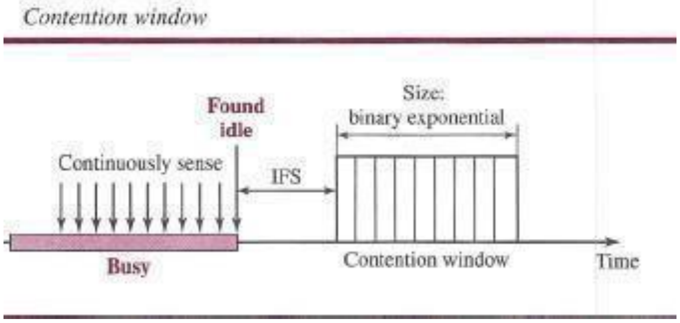
Carrier sense **multiple** access **with collision avoidance** *(CSMA/CA)* was invented for wireless networks. Collisions are avoided through the use of *CSMA/CA's* three strategies: the inter frame space, the contention window, and acknowledgments, as shown in below figure.

Flow diagram of CSMA/CA



Inter frame Space (IFS) First, collisions are avoided by deferring transmission even if the channel is found idle. When an idle channel is found, the station does not send immediately. It waits for a period of time called the *inter frame space* or *IFS.* Even though the channel may appear idle when it is sensed, a distant station may have already started transmitting. The distant station's signal has not yet reached this station. The IFS time allows the front of the transmitted signal by the distant station to reach this station. After waiting an IFS time, if the channel is still idle, the station can send, but it still needs to wait a time equal to the contention window (described next).The IFS variable can also be used to prioritize stations or frame types. For example, a station that is assigned shorter IFS has a higher priority.

Contention Window The contention window is an amount of time divided into slots. A station that is ready to send chooses a random number of slots as its wait time. The number of slots in the window changes according to the binary exponential back off strategy. This means that it is set to one slot the first time and then doubles each time the station cannot detect an idle channel after the IFS time. This is very similar to the p-persistent method except that a random outcome defines the number of slots taken by the waiting station. One interesting point about the contention window is that the station needs to sense the channel after each time slot. However, if the station finds the channel busy, it does not restart the process; it just stops the timer and restarts it when the channel is sensed as idle. This gives priority to the station with the longest waiting time. See below figure.



Acknowledgment

With all these precautions, there still may be a collision resulting in destroyed data. In addition, the data may be corrupted during the transmission. The positive acknowledgment and the time-out timer can help guarantee that the receiver has received the frame.

Frame Exchange Time Line

Below figure shows the exchange of data and control frames in time.

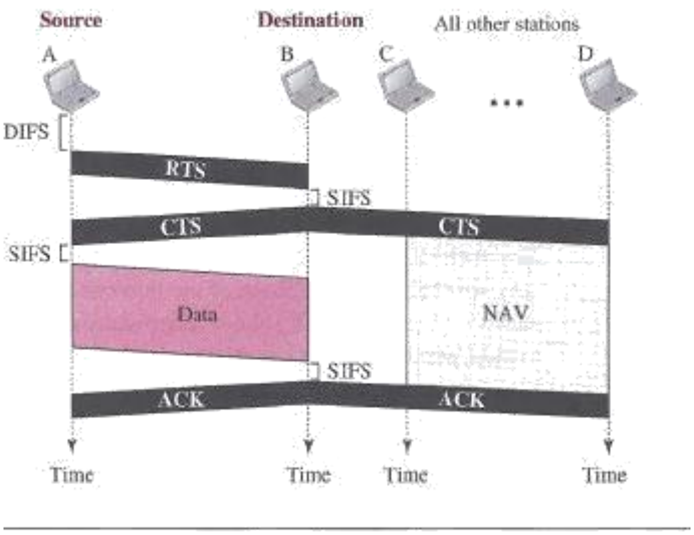
1. Before sending a frame, the source station senses the medium by checking the energy level at the carrier frequency.

a. The channel uses a persistence strategy with back off until the channel isidle.

b. After the station is found to be idle, the station waits for a period of time called the *DCF inter frame space (DIFS);* then the station sends a control frame called the *request to send (RTS).*

2. After receiving the RTS and waiting a period of time called the *short inter frame space (SIFS),* the destination station sends a control frame, called the *clear to send (CTS),* to the source station. This control frame indicates that the destination station is ready to receive data.

CSMA/CA and NAV



3. The source station sends data after waiting an amount of time equal to SIFS.
4. The destination station, after waiting an amount of time equal to SIFS, sends an acknowledgment to show that the frame has been received. Acknowledgment is needed in this protocol because the station does not have any means to check for the successful arrival of its data at the destination. On the other hand, the lack of collision in *CSMA/CD* is a kind of indication to the source that data have arrived.

Network Allocation Vector

How do other stations defer sending their data if one station acquires access? In other words, how is the *collision avoidance* aspect of this protocol accomplished? The key is a feature called *NA V.* When a station sends an RTS frame, it includes the duration of time that it needs to occupy the channel. The stations that are affected by this transmission create a timer called a network allocation vector (NAV) that shows how much time must pass before these stations are allowed to check the channel for idleness. Each time a station accesses the system and sends an RTS frame, other stations start their NAY. In other words, each station, before sensing the physical medium to see if it is idle, first checks its NAY to see if it has expired.

Collision During Handshaking

What happens if there is a collision during the time when RTS or CTS control frames are in transition, often called the *handshaking period?* Two or more stations may try to send RTS frames at the same time. These control frames may collide. However, because there is no mechanism for collision detection, the sender assumes there has been a collision if it has not received a CTS frame from the receiver. The back off strategy is employed, and the sender tries again.

Hidden-Station Problem

The solution to the hidden station problem is the use of the handshake frames (RTS and CTS).Above figure also shows that the RTS message from B reaches A, but not C. However, because both Band C are within the range of A, the CTS message, which contains the duration of data transmission from B to A, reaches C. Station C knows that some hidden station is using the channel and refrains from transmitting until that duration is over.

CSMAICA and Wireless Networks

CSMA/CA was mostly intended for use in wireless networks. The procedure described above, however, is not sophisticated enough to handle some particular issues related to wireless networks, such as hidden terminals or exposed terminals.

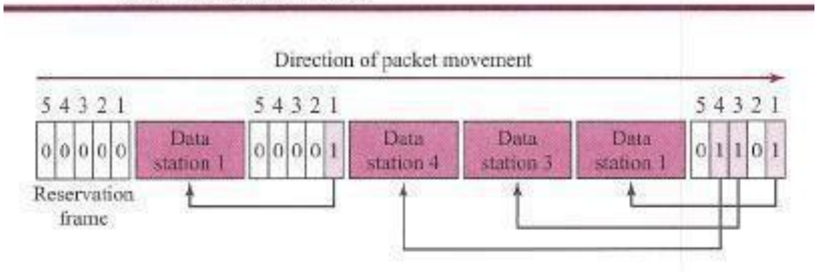
CONTROLLED ACCESS

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

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. Below 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.

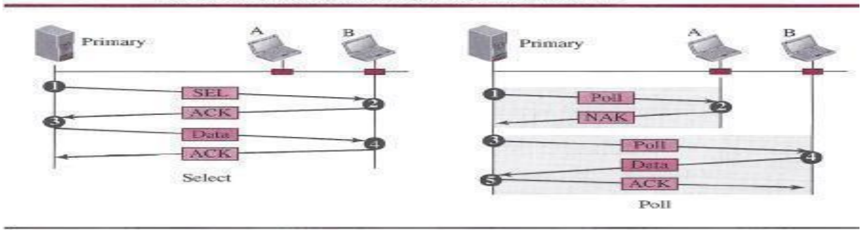
Reservation access method



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 This method uses poll and select functions to prevent collisions. However, the drawback is if the primary station fails, the system goes down.

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.

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*. The predecessor is the station which is logically before the station in the ring; the successor is the station which is after the station in the ring. 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.

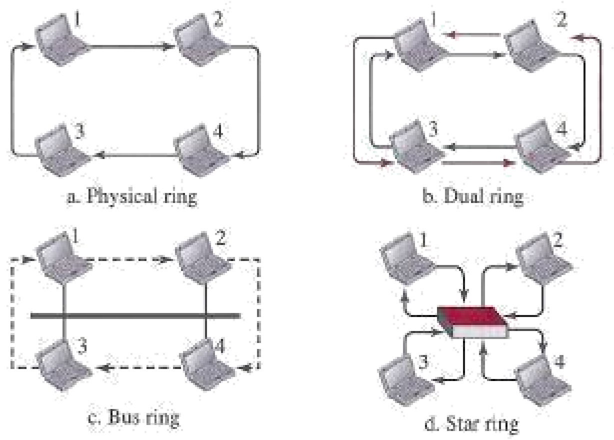
But how is the right to access the channel passed from one station to another? In this method, a special packet called a *token* circulates through the ring. The possession of the token gives the station the right to access the channel and send its data. When a station has some data to send, it waits 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. Below figure shows four different physical topologies that can create a logical ring.

Logical ring and physical topology in token-passing access method



a. Physical ring

b. Dual ring

c. Bus ring

d. Star ring

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 Distributed Data 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 and 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.

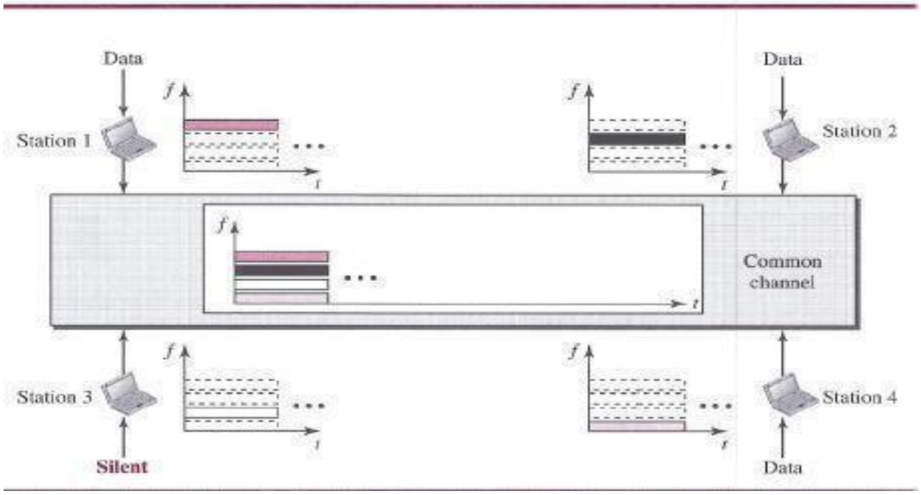
CHANNELIZATION

Channelization (or *channel partition*, as it is sometimes called) is a multiple-access method in which the available bandwidth of a link is shared in time, frequency, or through code, among different stations. In this section, we discuss three channelization protocols: FDMA, TDMA, and CDMA.

FDMA

In frequency-division multiple access (FDMA), the available bandwidth is divided into frequency bands. Each station is allocated a 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. Each station also uses a band pass filter to confine the transmitter frequencies. To prevent station interferences, the allocated bands are separated from one another by small *guard bands. Below f*igure shows the idea of FDMA.

Frequency-division multiple access (FDMA)



- FDMA specifies a predetermined frequency band for the entire period of communication. This means that stream data (a continuous flow of data that may not be packetized) can easily be used with FDMA. We will see in Chapter 16 how this feature can be used in cellular telephone systems. We need to emphasize that although FDMA and frequency-division multiplexing

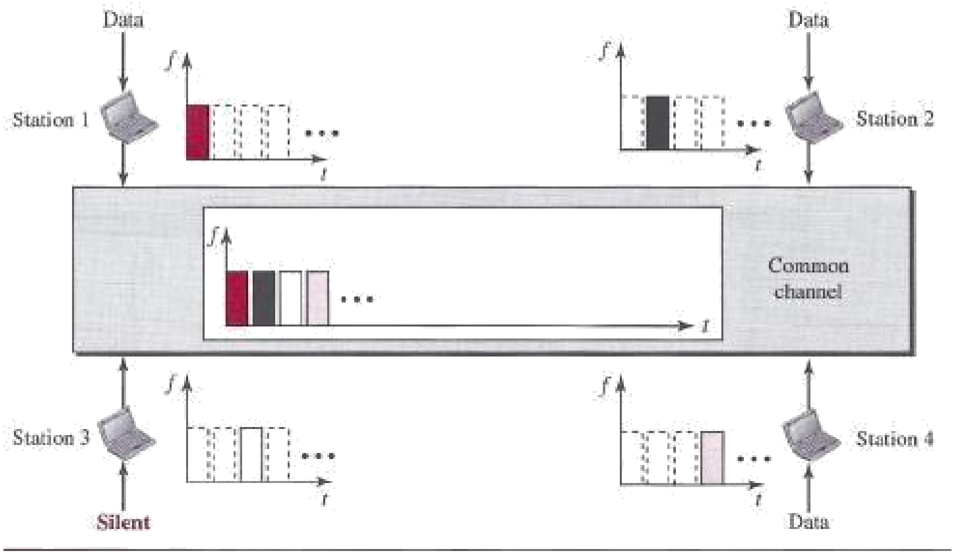
- (FDM) conceptually seem similar, there are differences between them FDM. The channels that are combined are low-pass. The multiplexer modulates the signals, combines them, and creates a band pass signal. The bandwidth of each channel is shifted by the multiplexer.

- FDMA, on the other hand, is an access method in the data-link layer. The data link layer in each station tells its physical layer to make a band pass signal from the data passed to it. The signal must be created in the allocated band. There is no physical multiplexer at the physical layer. The signals created at each station are automatically band pass-filtered. They are mixed when they are sent to the common channel.

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 its assigned time slot. Below figure shows the idea behind TDMA.

Time-division multiple access (TDMA)



The main problem with TDMA lies in achieving synchronization between the different stations. Each station needs to know the beginning of its slot and 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. We also need to emphasize that although TDMA and time-division multiplexing (TDM) conceptually seem the same, there are differences between them. TDM, as, is a physical layer technique that combines the data from slower channels and transmits them by using a faster channel. The process uses a physical multiplexer that interleaves data units from each channel. TDMA, on the other hand, is an access method in the data-link layer. The data-link layer in each station tells its physical layer to use the allocated time slot. There is no physical multiplexer at the physical layer.

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 in that only one channel occupies the entire bandwidth of the link. It differs from TDMA in that all stations can send data simultaneously; there is no timesharing.

In CDMA, one channel carries all transmissions simultaneously

Analogy

Let us first give an analogy. CDMA simply means communication with different codes. For example, in a large room with many people, two people can talk privately in English if nobody else understands English. Another two people can talk in Chinese if they are the only ones who understand Chinese, and so on. In other words, the common channel, the space of the room in this case, can easily allow communication between several couples, but in different languages (codes).

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 Cj, to the second is *c2,* and so on. We assume that the assigned codes have two properties.

1. If we multiply each code by another, we get O.

2. 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.

Station 1 multiplies (a special kind of multiplication, as we will see) its data by its code to get *d,*. Cj. Station 2 multiplies its data by its code to get *d2* . <z- and so on. The data that go on the channel are the sum of all these terms, as shown in the box. 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 C r- the code of station 1.

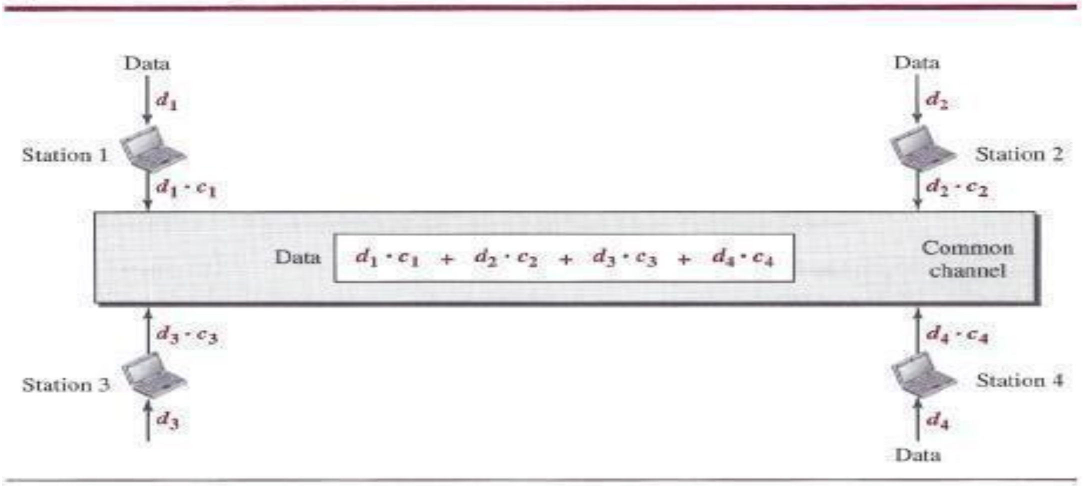
Because (cI . cI) is 4, but *(c2* . cl), *(c3* . cl)' and *(c4* . cI) are all Os, station 2 divides the result by 4 to get the data from station 1.

Chips

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

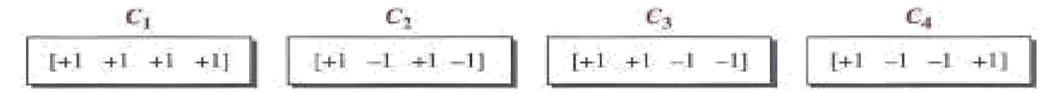
Later in this chapter we show how we chose these sequences. For now, we need to know that we did not choose the sequences randomly; they were carefully selected. They are called *orthogonal sequences* and have the following properties:

Simple idea of communication with code



$$\begin{aligned} \text{data} &= (d_1 \cdot c_1 + d_2 \cdot c_2 + d_3 \cdot c_3 + d_4 \cdot c_4) \cdot c_1 \\ &= d_1 \cdot c_1 \cdot c_1 + d_2 \cdot c_2 \cdot c_1 + d_3 \cdot c_3 \cdot c_1 + d_4 \cdot c_4 \cdot c_1 = 4 \times d_1 \end{aligned}$$

Chip sequences



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 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

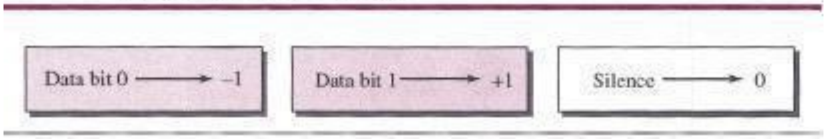
4. If we multiply two different sequences, element by element, and add the results, we get O. This is called the *inner product* of two different sequences. For example,
[+ 1 +1 -1 -1] • [+1 +1 + 1 + 1] = 1 + 1 - 1 - 1 = 0

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.

Data representation in CDMA



Encoding and Decoding

As a simple example, we show how four stations share the link during a I-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. Figure shows the situation. Now imagine that 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 --7 bit1

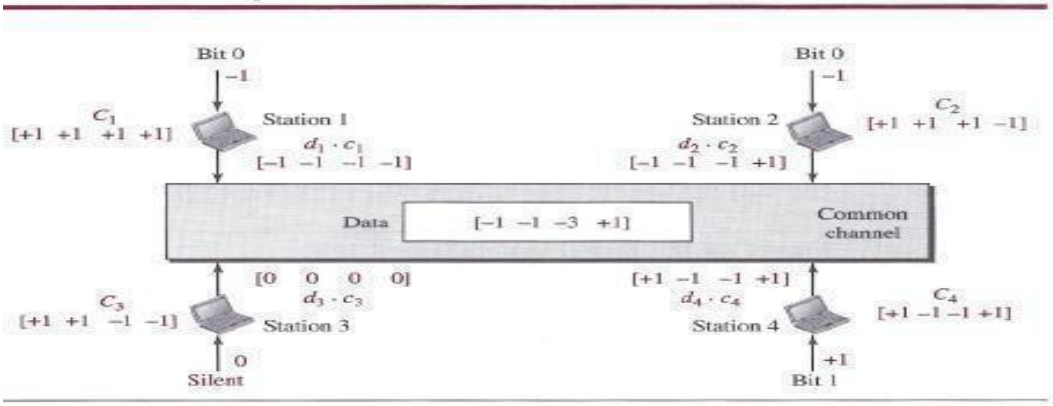
Signal Level

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 12.27). The figure shows the corresponding signals for each station (using NRZ-L for simplicity) and the signal that is on the common channel. Below figure 12.28 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 O.

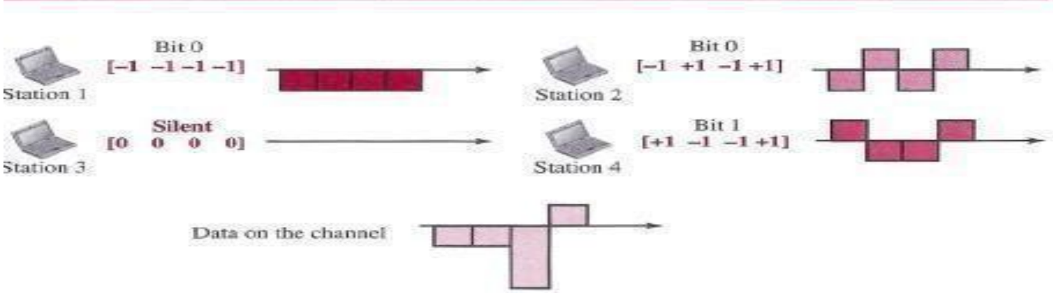
Sequence Generation

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. In the Walsh table, each row is a sequence of chips. *WI* for a one-chip sequence has one row and one column. We can choose -lor +1 for the chip for this trivial table (we chose +1). According to Walsh, if we know the table for *N* sequences *WN,* we can create the table for *2N* sequences *W2N,* as shown in Figure 12.29. The *WN* with the over bar *WN* stands for the complement of *WN,* where each +1 is changed to -1 and vice versa. Below figure also shows how we can create *W*2 and *W*4 from *WI'* After we select *WI' W2* can be made from four *Wls,* with the last one the complement of *WI'* After *W2* is generated, *W4* can be made of four *W2s,* with the last one the complement of *W2.* Of course, *Ws* is composed of four *W4s,* and so on. Note that after *WN* is made, each station is assigned a chip corresponding to a row.

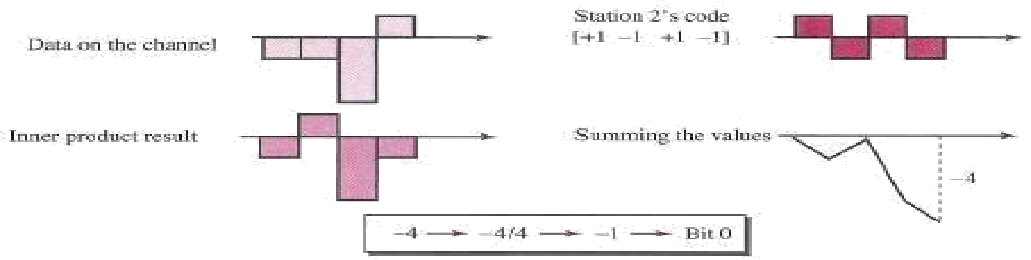
Sharing channel in CDMA



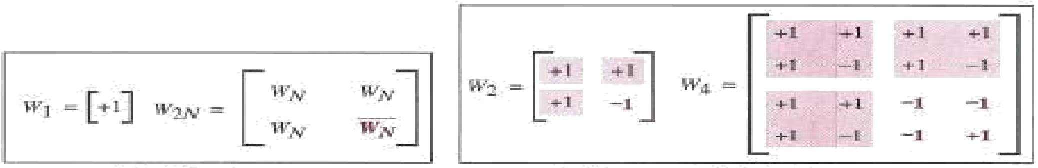
Digital signal created by four stations in CDMA



Decoding of the composite signal for one in CDMA



General rule and examples of creating Walsh tables


$W_1 = [+1]$
$W_{2N} = \begin{bmatrix} W_N & W_N \\ W_N & \overline{W_N} \end{bmatrix}$
$W_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix}$
$W_4 = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}$

a. Two basic rules

b. Generation of W_2 and W_4

Something we need to emphasize is that the number of sequences, N, needs to be a power of 2. In other words, we need to have $N = 2^m$.

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

Example

Find the chips for a network with

a. Two stations

b. Four stations

Solution

We can use the rows of *W2* and *W4* in Figure:

a. For a two-station network, we have l+ 1 +1] and l+ 1-1].

b. For a four-station network we have [+1 +1 +1 +1], [+1-1 +1-1], [+1 +1-1-1], and [+1-1-1 +1].

ETHERNET

IEEE Project 802

Before we discuss the Ethernet protocol and all its generations. . In 1985, the Computer Society of the IEEE started a project, called *Project 802*, to set standards to enable intercommunication among equipment from a variety of manufacturers. Project 802 does not seek to replace any part of the OSI model or TCPIIP protocol suite. Instead, it is a way of specifying functions of the physical layer and the data-link layer of major LAN protocols. 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 different LAN protocols.

Logical Link Control (LLC)

Earlier we discussed *data link control.* We said that data link control handles framing, flow control, and error control. 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. 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.

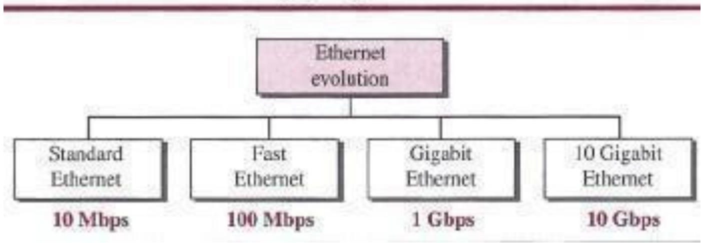
Media Access Control (MAC)

Earlier we discussed multiple access methods including random access, controlled access, and channelization. IEEE Project 802 has created a sub-layer called *media access control* that defines the specific access method for each LAN. 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. As we mentioned in the previous section, part of the framing function is also handled by the MAC layer.

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.

Ethernet evolution through four generations



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

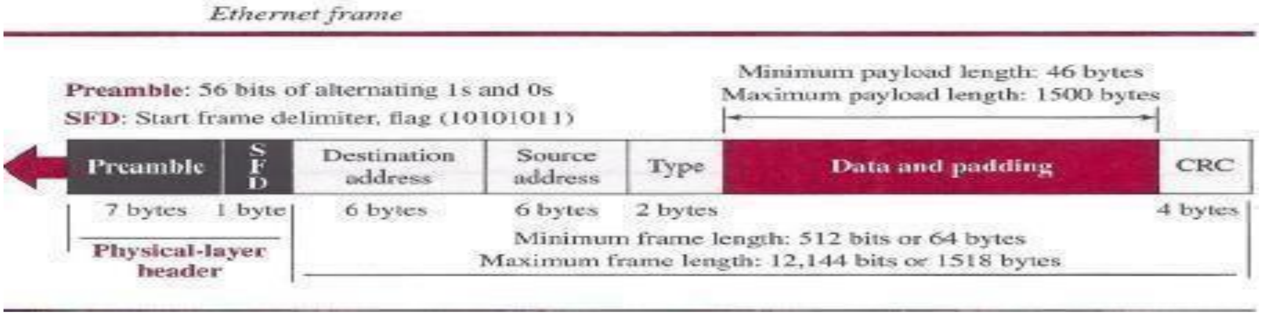
Let us first discuss some characteristics of the Standard Ethernet.

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.

Frame Format

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


Ethernet frame
Preamble: 56 bits of alternating 1s and 0s
SFD: Start frame delimiter, flag (10101011)
Minimum payload length: 46 bytes
Maximum payload length: 1500 bytes
Preamble
S F D
Destination address
Source address
Type
Data and padding
CRC
7 bytes
1 byte
6 bytes
6 bytes
2 bytes
4 bytes
Physical-layer header
Minimum frame length: 512 bits or 64 bytes
Maximum frame length: 12,144 bits or 1518 bytes

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.

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.

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.

Minimum frame length: 64 bytes	**Minimum data length: 46 bytes**
Maximum frame length: 1518 bytes	**Maximum data length: 1500 bytes**

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, with a 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.

Example

Show how the address 47:20:IB:2E:08:EE is sent out online.

Solution

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

Hexadecimal 47 20 IB 2E 08 EE
Binary 01000111 00100000 00011011 00101110 00001000 11101110
Transmitted ~ 11100010 00000100 11011000 01110100 00010000 01110111

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.

Example

Define the type of the following destination addresses:
a. 4A:30:10:21:10:1A
b. 47:20:IB:2E:08:EE
c. FF:FF:FF:FF:FF:FF

Solution

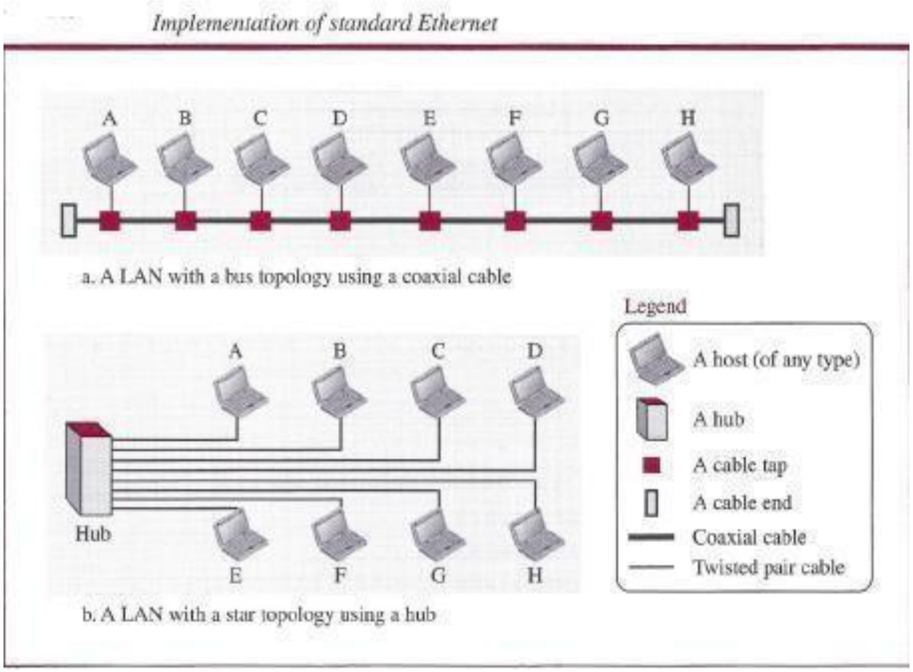
To find the type of the address, we need to look at the second hexadecimal digit from the left. If it is even, the address is unicast. If it is odd, the address is multicast. If all digits are Fs, the address is broadcast. Therefore, we have the following:
a. This is a unicast address because A in binary is 1010 (even).
b. This is a multicast address because 7 in binary is 0111 (odd).
c. This is a broadcast address because all digits are Fs in hexadecimal

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.

- In a unicast transmission, all stations will receive the frame, the intended recipient keeps and handles the frame; the rest discard it.
- 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.

- In a broadcast transmission, all stations (except the sender) will receive the frame andall stations (except the sender) keep and handle it.


Implementation of standard Ethernet
A
B
C
D
E
F
G
H
a. A LAN with a bus topology using a coaxial cable
A
B
C
D
Hub
E
F
G
H
b. A LAN with a star topology using a hub
Legend
A host (of any type)
A hub
A cable tap
A cable end
Coaxial cable
Twisted pair cable

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 *(TB* in Figure 12.l3) 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.

Example

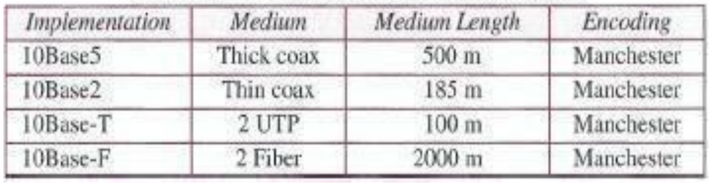
In the Standard Ethernet with the transmission rate of 10 Mbps, we assume that the length of the medium is 2500 m and the size of the frame is 512 bits. The propagation speed of a signal in a cable is normally 2 x 108 mls. Propagation delay = 2500/ (2 x 108) = 12.5 fJ.s Transmission delay = 512/ (107) = 51.2 fJ.S *a* = 12.5/51.2 = 0.24 Efficiency = 39%

The example shows that *a* = 0.24, which means only 0.24 of a frame occupies the whole medium in this case. The efficiency is 39 percent, which is considered moderate; it means that only 61 percent of the time the medium is occupied but not used by a station.

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

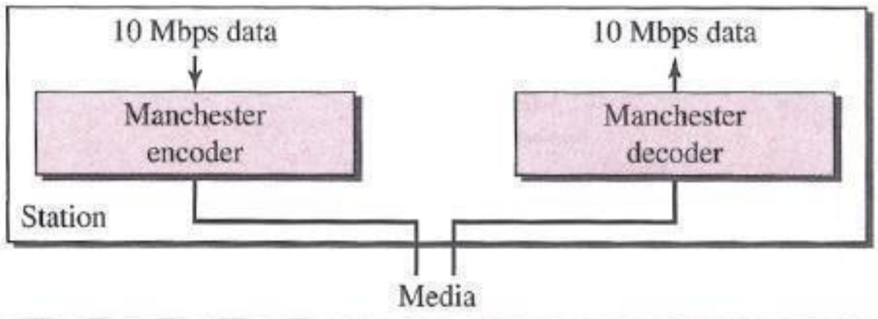
Table

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.

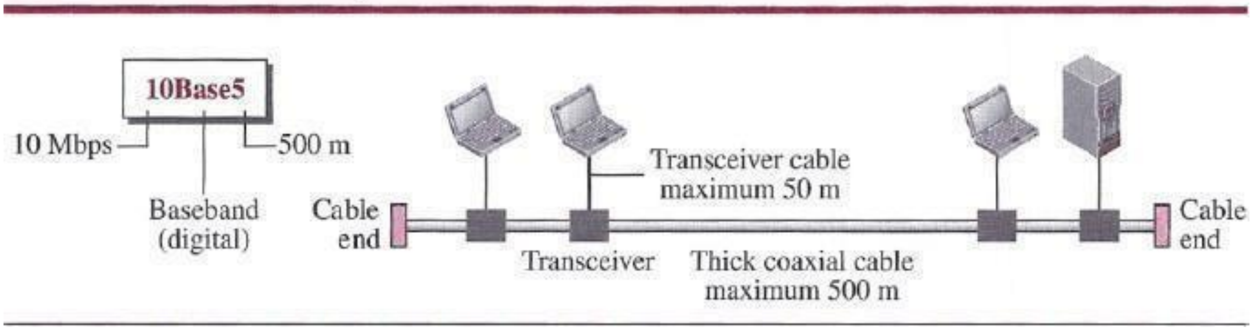
Encoding in a Standard Ethernet implementation



10BaseS: 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

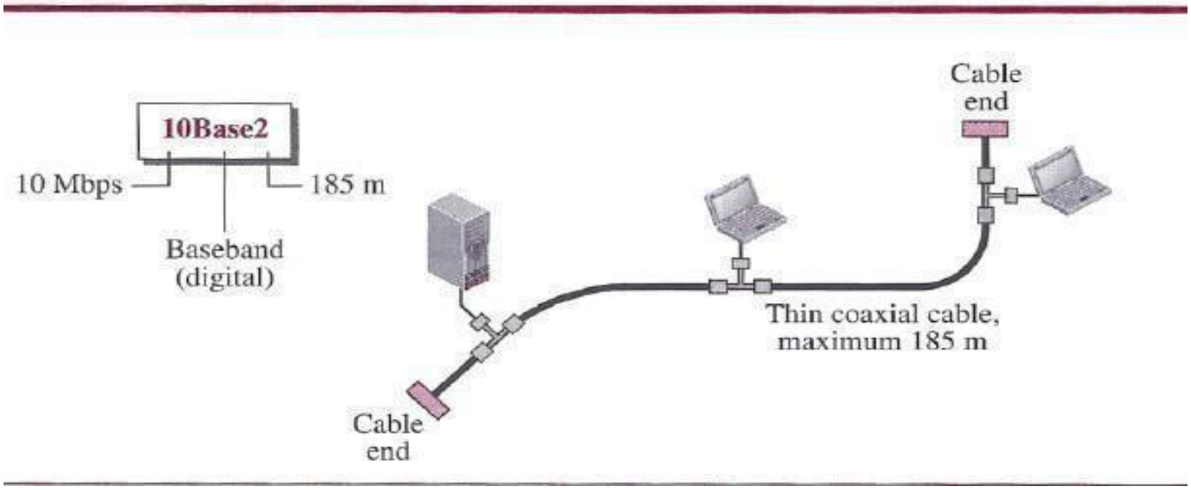


The transceiver is responsible for transmitting, receiving, and detecting collisions. The transceiver is connected to the station via a transceiver cable that provides separate paths for sending and receiving. This means that collision can only happen in the coaxial cable. The maximum length of the coaxial 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

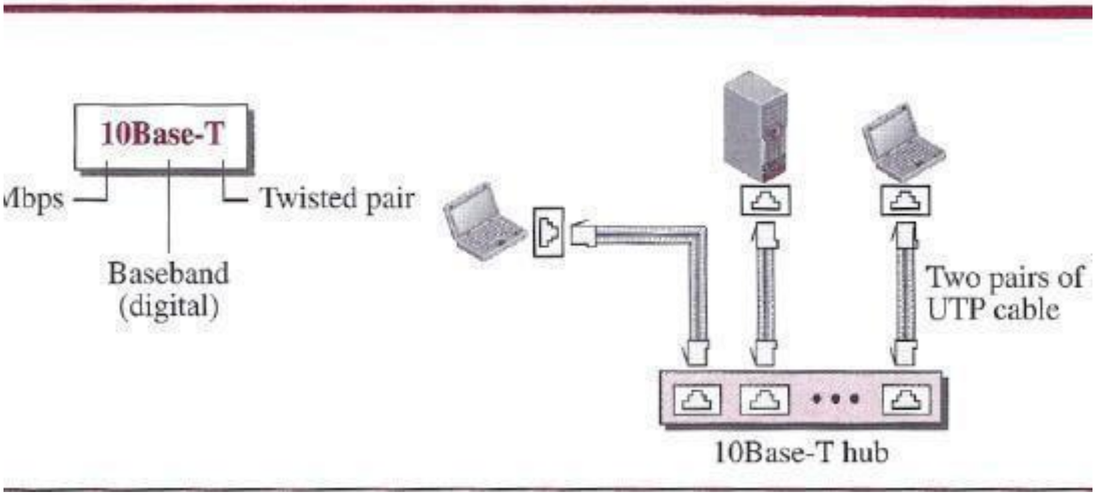


Note that the collision here occurs in the thin coaxial cable. This implementation is more cost effective than 10Base5 because thin coaxial cable is less expensive than thick coaxial and the tee connections are much cheaper than taps. Installation is simpler because the thin coaxial cable is very flexible. 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

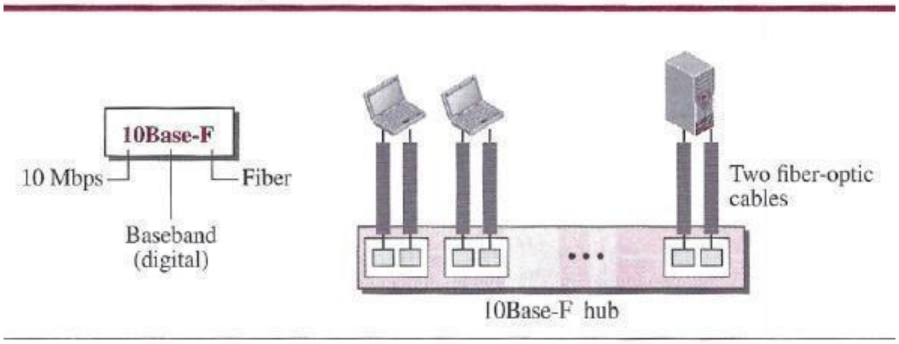


Note that two pairs of twisted cable create two paths (one for sending and one for receiving) between the station and the hub. Any collision here happens in the hub. Compared to lOBase5 or lOBase2, we can see that the hub actually replaces the coaxial cable as far as a collision is concerned. The maximum 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 connected to the hub using two fiber-optic cables, as shown in Figure.

10Base-F implementation



Physical Layer

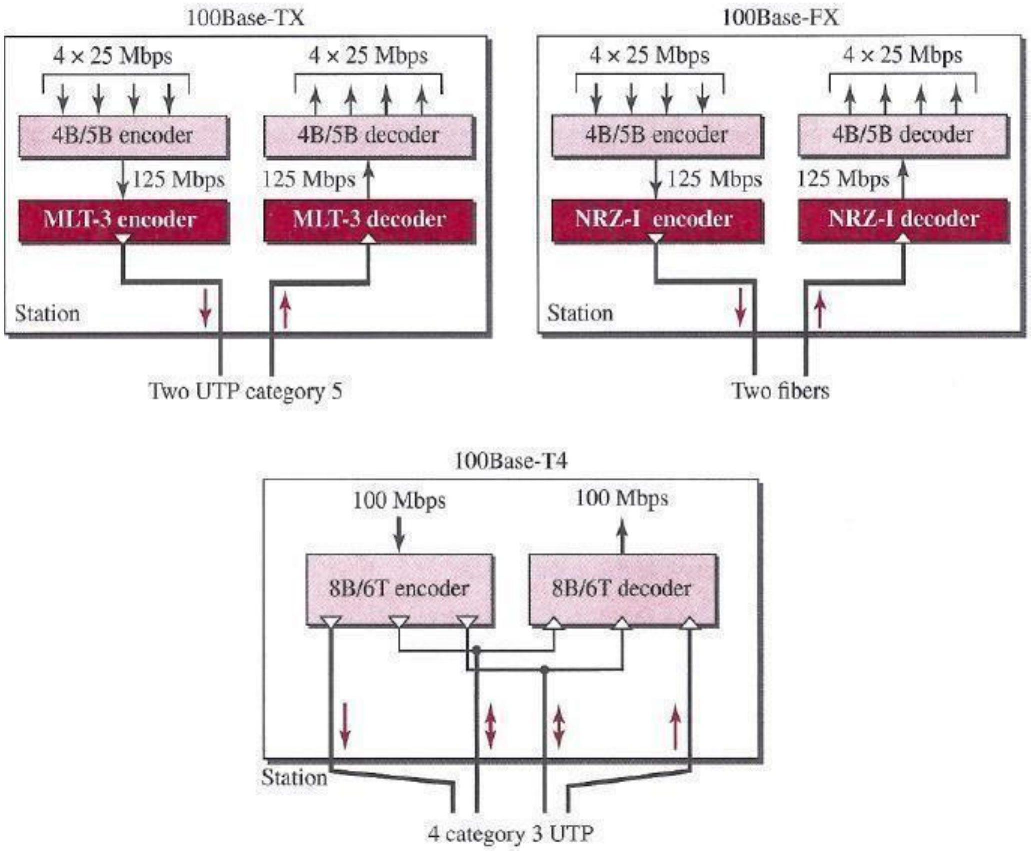
To be able to handle a 100 Mbps data rate, several changes need to be made at the physical layer.

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.