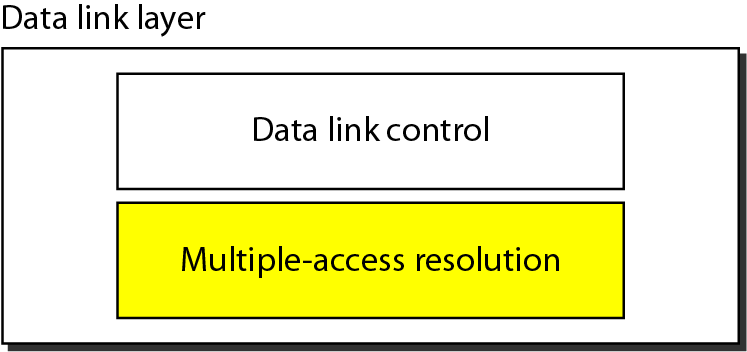
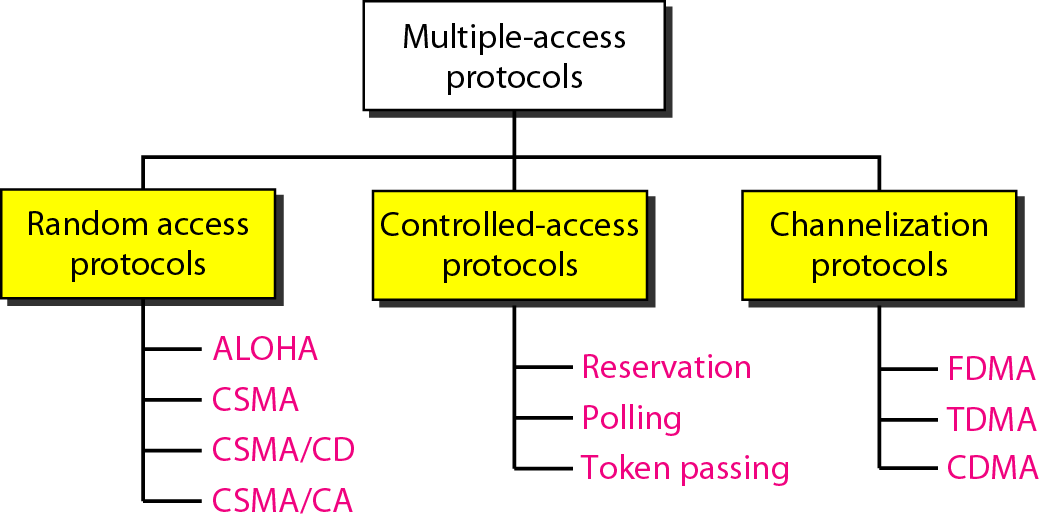
If we use our cellular phone to connect to another cellular phone, the channel (the bandallocated to the vendor company) is not dedicated. A person a few feet away from us may beusing the same channel to talk to her friend.

We can consider the data link layer as two sublayers. The upper sublayer is responsible for data link control, and the lower sublayer is responsible for resolving access to the shared media. Ifthe channel is dedicated, we do not need the lower sublayer.



Many formal protocols have been devised to handle access to a shared link. We categorize them into three groups. Protocols belonging to each group are shown in below figure:



**1. 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. 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. 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?

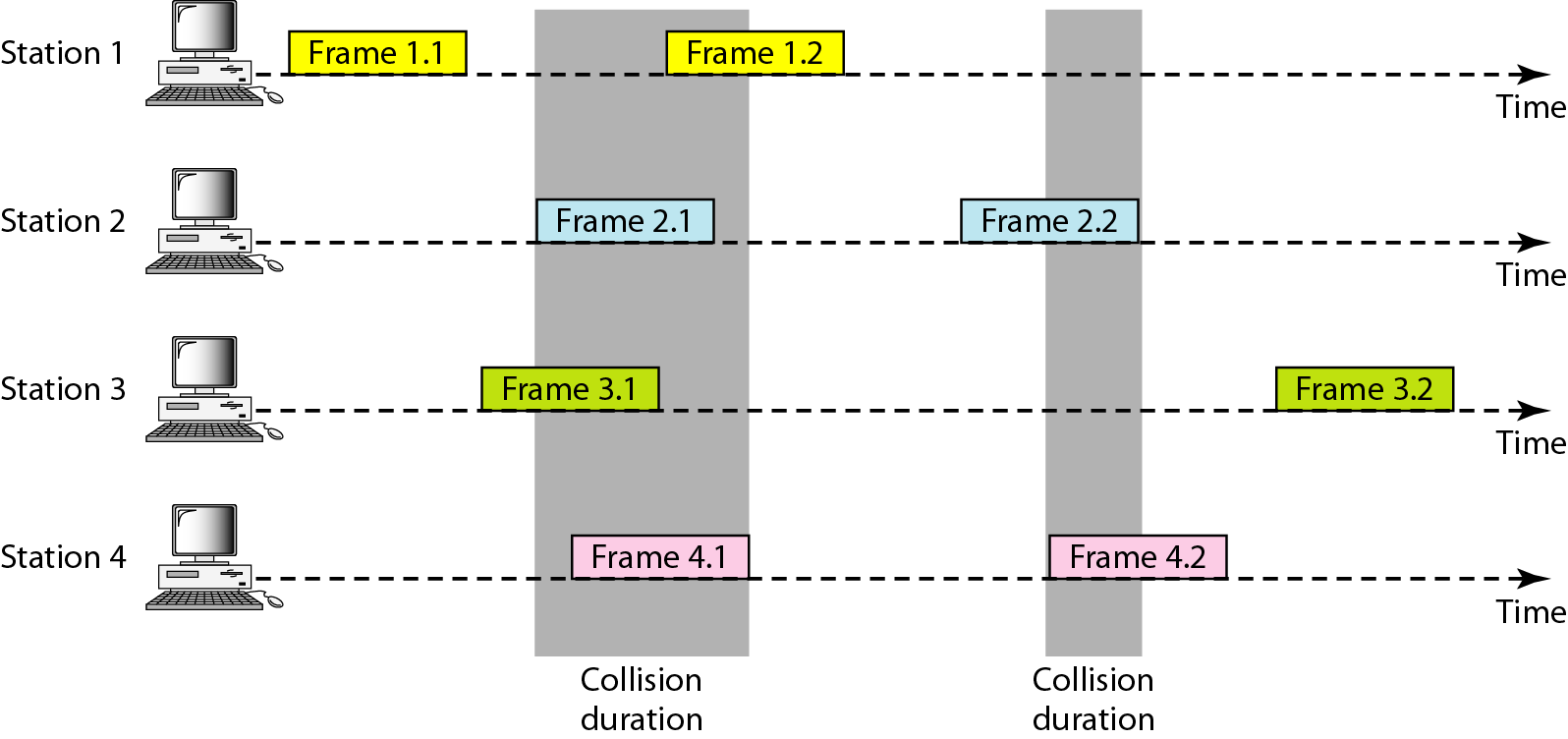
**I. ALOHA**

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.

**a. Pure ALOHA**

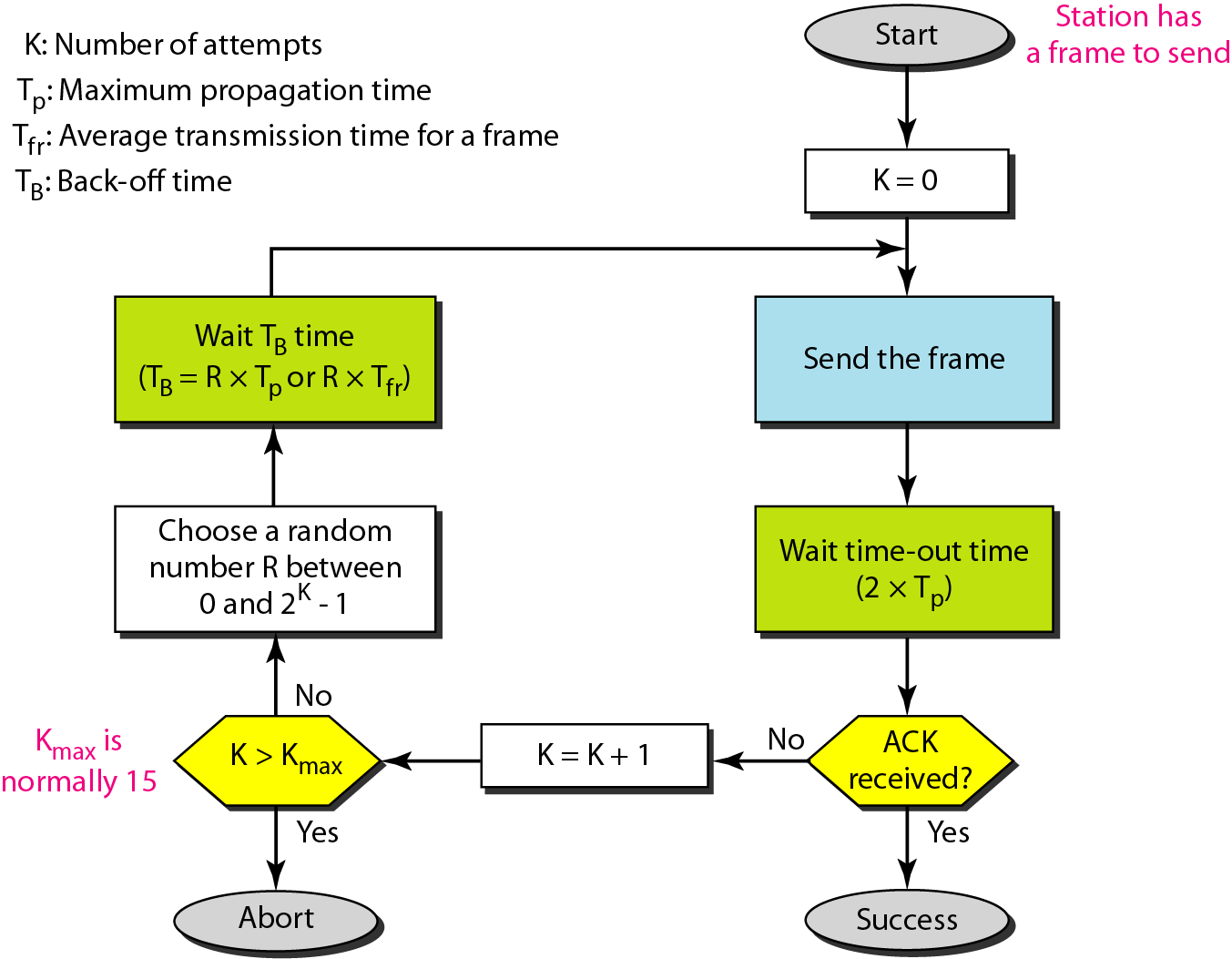
* The original ALOHA protocol is called pure ALOHA. This is a simple, but elegant protocol.
* The idea is that each station sends a frame whenever it has a frame to send.
* However, since there is only one channel to share, there is the possibility of collision between frames from different stations.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

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. Below figure 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.



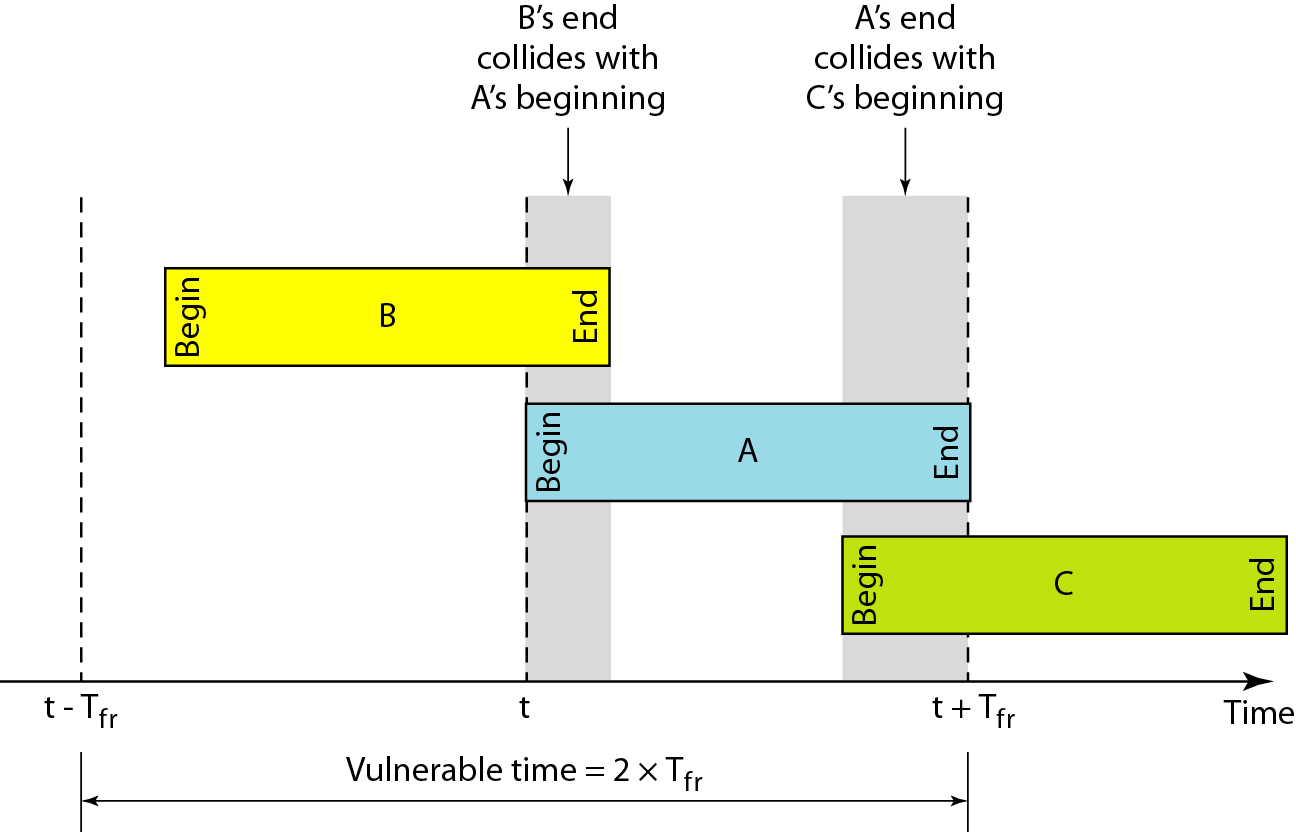
**Procedure for pure ALOHA protocol**

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 (2xTp). The back-off time TBis a random value that normally depends on *K* (the number of attempted unsuccessful transmissions). The formula for *TB* depends on the implementation. One common formula is the **binary exponential back-off.** In thismethod, for each retransmission, a multiplier in the range 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 *TB.* The value of Kmaxis usually chosen as 15.



**Vulnerable time:**Let us find the length of time, the vulnerable time, in which there is a possibility of collision. We assume that the stations send fixed-length frames with each frame taking Tfr s to send.

**Pure ALOHA vulnerable time = 2 x Tfr**

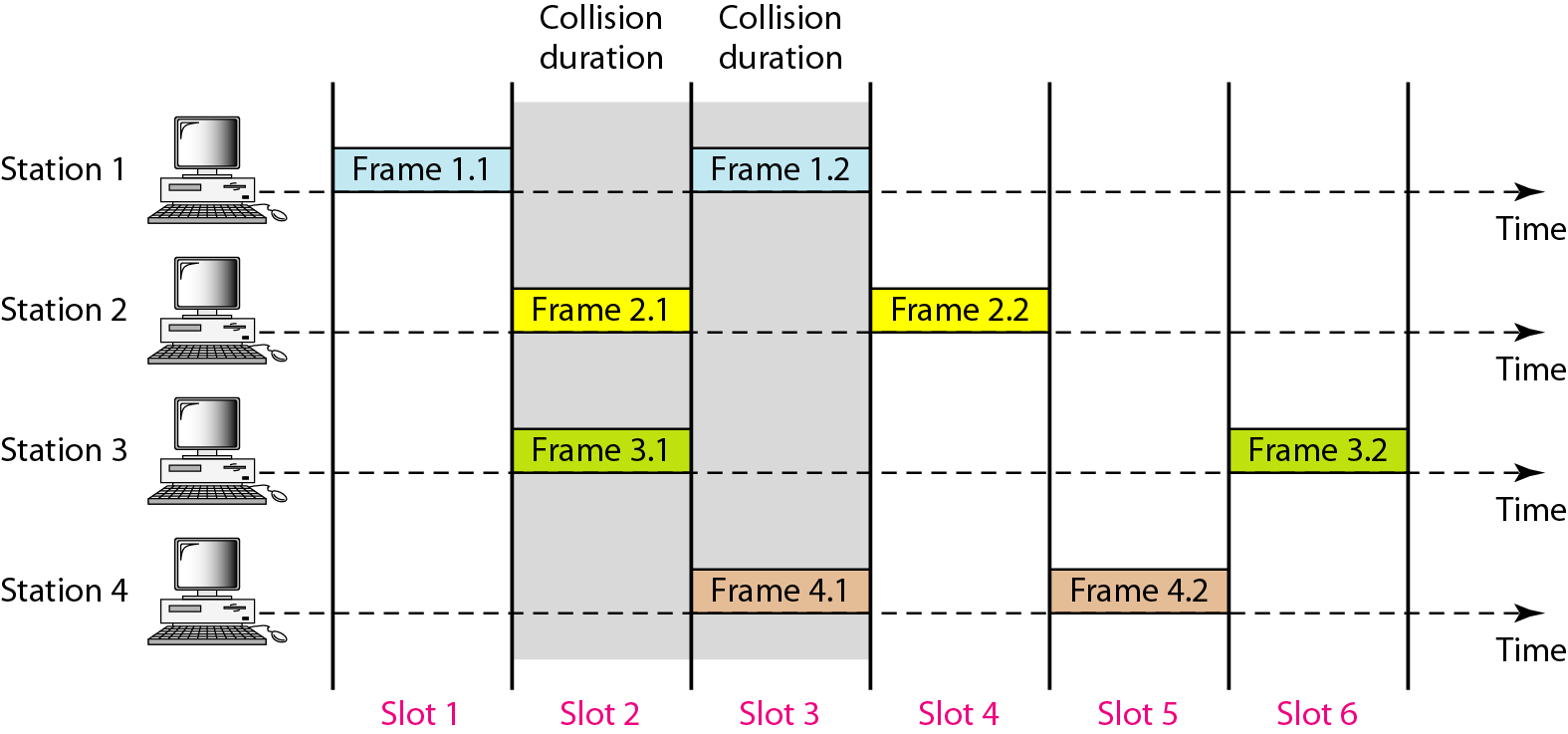


**Throughput:** Let us call G the average number of frames generated by the system during one frame transmission time. Then it can be proved that the average number of successful transmissions for pure ALOHA is **S = G x e-2G***.* The maximum throughput **Smax is 0.184, for G = 1/2**. In other words, if one-half a frame is generated during one frame transmission time (in other words, one frame during two frame transmission times), then 18.4 percent of these frames reach their destination successfully.

|  |
| --- |
| **Example:** A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the requirement to make this frame collision-free?  **Solution:**  Average frame transmission time Tfr is 200 bits/200 kbps or 1 ms. The vulnerable time is 2 x 1 ms =2ms. This means no station should send later than 1 ms before this station starts transmission and no station should start sending during the one 1ms period that this station is sending.  **Example:** A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is thethroughput if the system (all stations together) produces  a. 1000 frames per second  b. 500 frames per second  c. 250 frames per second  **Solution:**  The frame transmission time is *200/200* kbps or 1 ms (milliseconds).   1. If the system creates 1000 frames per second, this is 1 frame per millisecond. The load is G=1. In this case S =G x *e-2G*or S =0.135 (13.5 percent). This means that the throughput is 1000 X 0.135 =135 frames. Only 135 frames out of 1000 will probably survive. 2. If the system creates 500 frames per second, this is (1/2) frame per millisecond.The load is G=*(1/2).* In this case S = G x *e-2G*or S = 0.184 (18.4 percent). This means that the throughput is 500 x 0.184 =92 and that only 92 frames out of 500 will probably survive. Note that this is the maximum throughput case, percentagewise. 3. If the system creates 250 frames per second, this is (1/4) frame per millisecond. The load is G= (1/4). In this case S = G x *e-2G*or S =0.152 (15.2 percent). This means that the throughput is 250 x 0.152 = 38. Only 38 frames out of 250 will probably survive. |

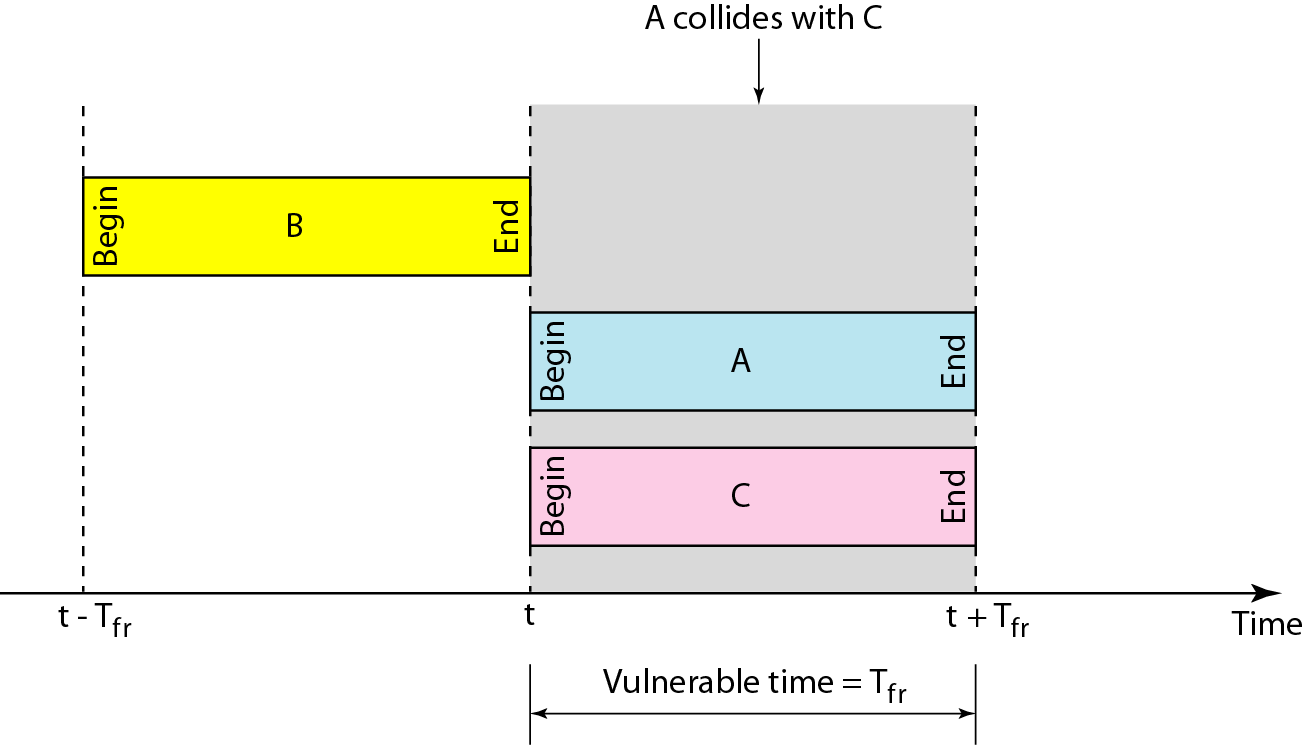
**b. Slotted ALOHA**

Slotted ALOHA was invented to improve the efficiency of pure ALOHA. In slotted ALOHA we divide the time into slots of Tfr s 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.

Below figure shows that the vulnerable time for slotted ALOHA is one-half that of pure ALOHA.



**Slotted ALOHA vulnerable time = Tfr**

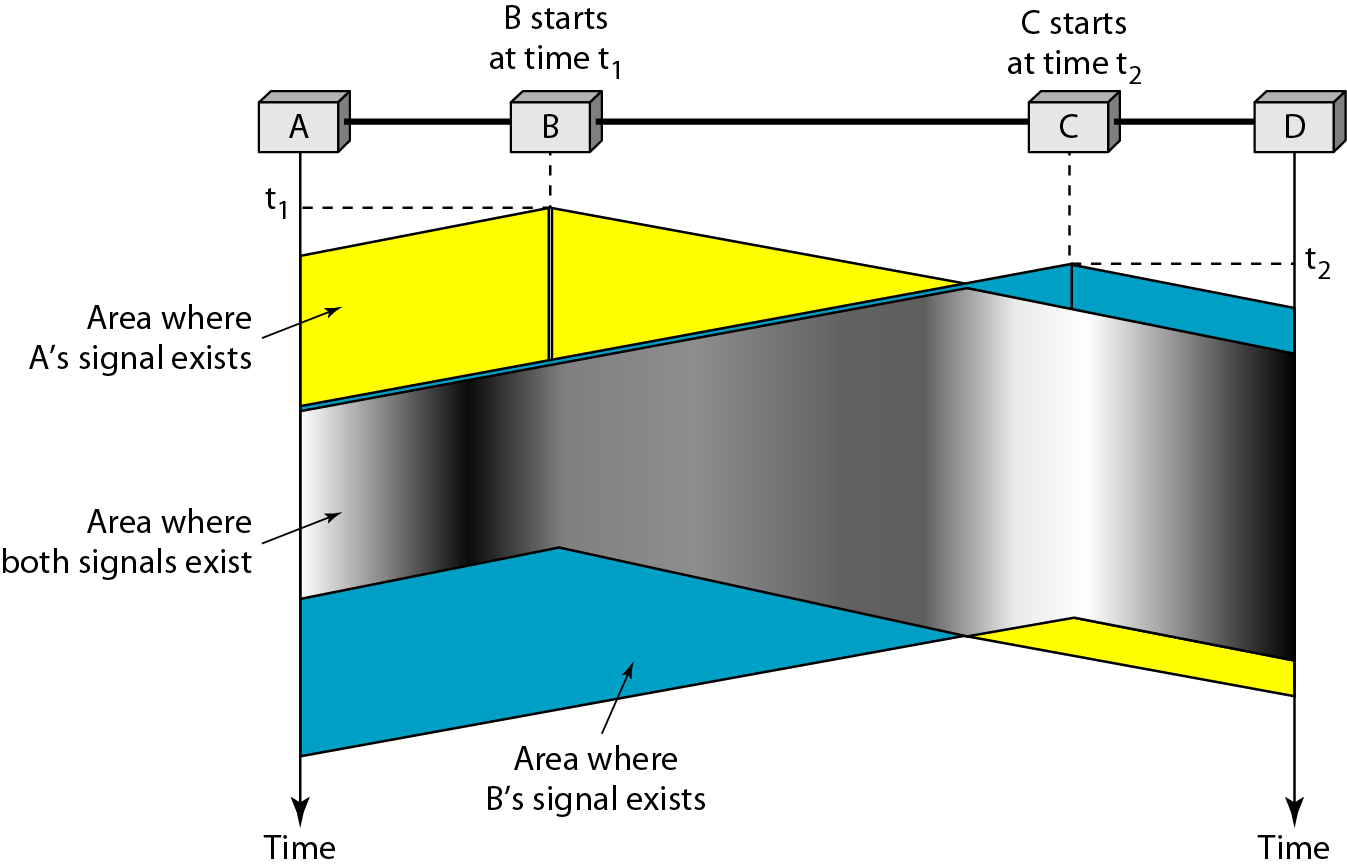
**The throughput for slotted ALOHA is S =G x *e-G.***

**The maximum throughput Smax =0.368when 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.   1. If the system creates 1000 frames per second, this is 1 frame per millisecond. The load is G=1. So S =G x *e-G*or *S* =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. 2. If the system creates 500 frames per second, this is (1/2) frame per millisecond. The load is G= *(1/2).* In this case S =G x *e-G*or S =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. 3. If the system creates 250 frames per second, this is (1/4) frame per millisecond. The load is G= (1/4). In this case S =G x *e-G*or S =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. |

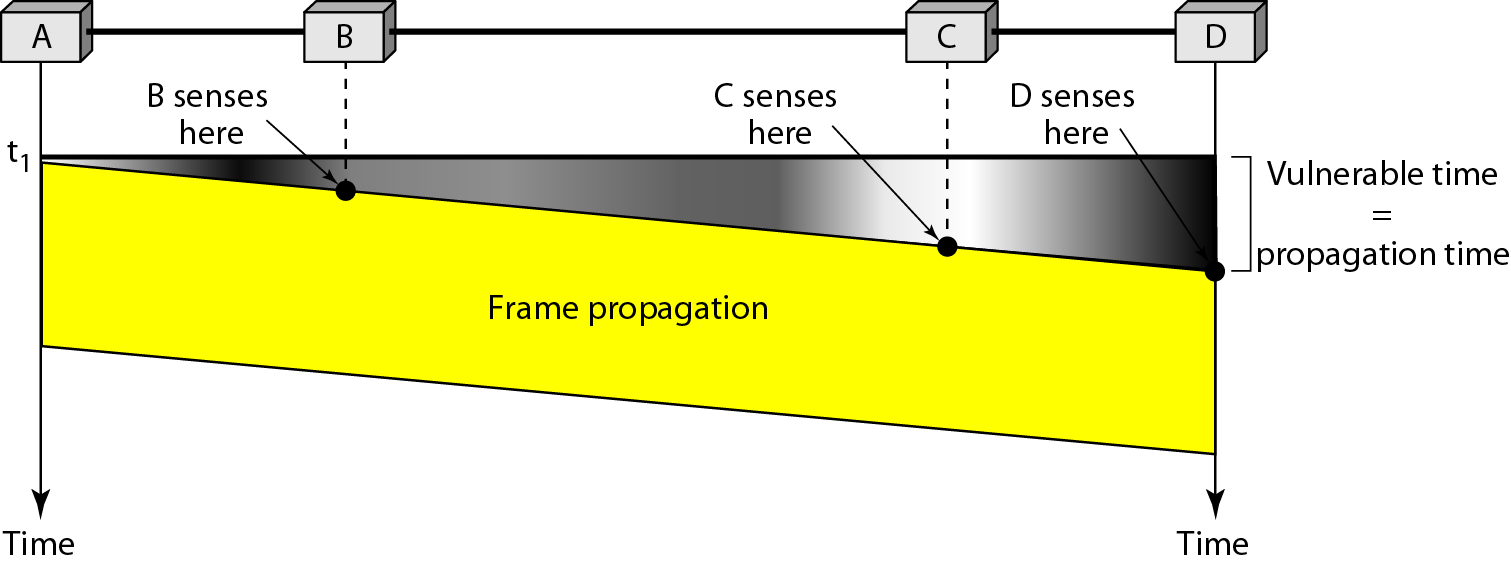
**II. CARRIER SENSE MULTIPLE ACCESS (CSMA)**

* 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**

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.

**1-Persistent:** 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 I). This method has the highest chance of collision because two or more stations may find the line idle and send their frames immediately.

**Non-persistent:** 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.

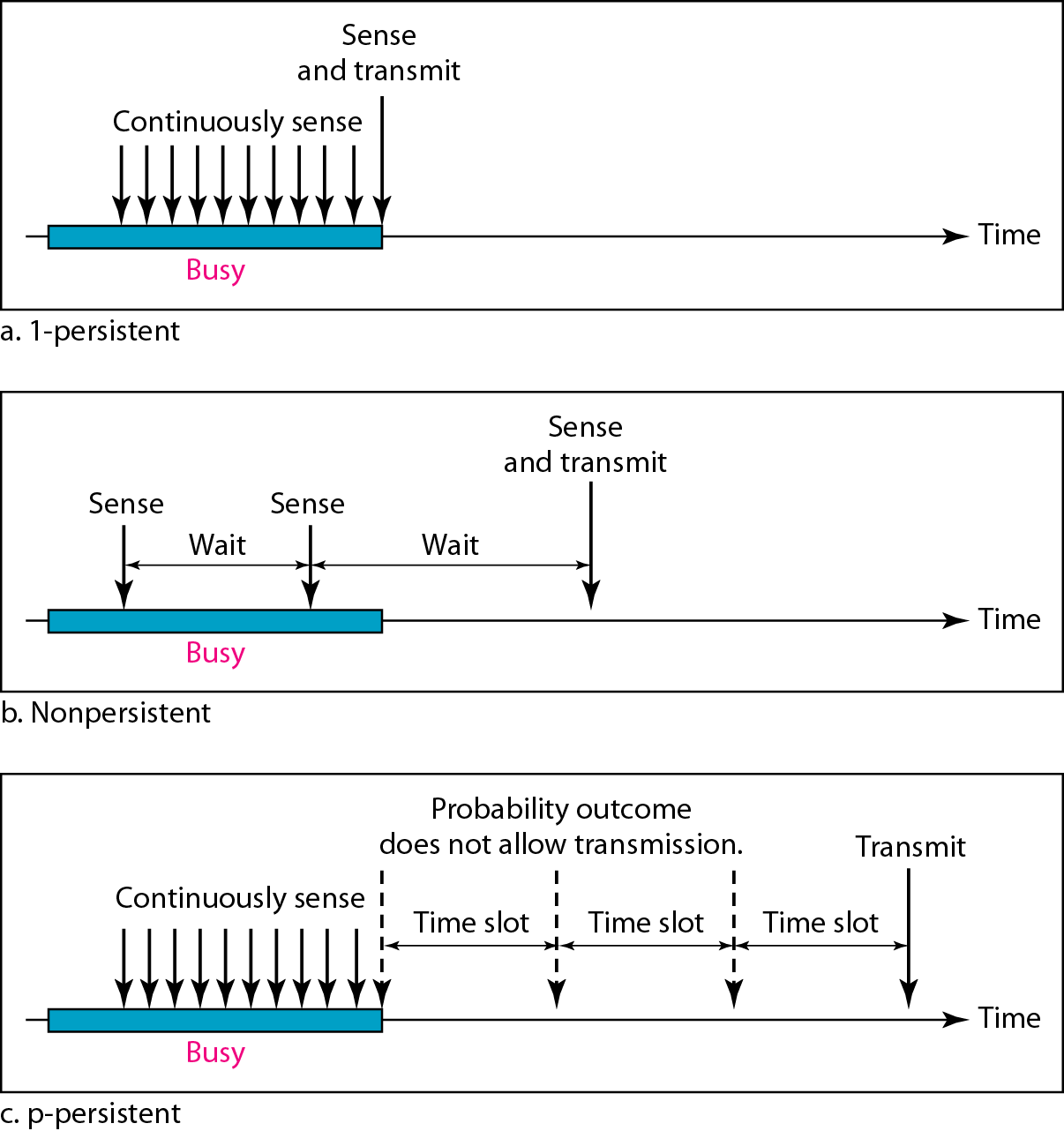
**p-Persistent: The p-persistent method** is used if the channel has time slots with a slot duration equal to or greater the maximum propagation time. The p-persistent approach combines the advantages of the other two strategies. It reduces the chance of collision and improves efficiency. In this method, after the station finds the than 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.

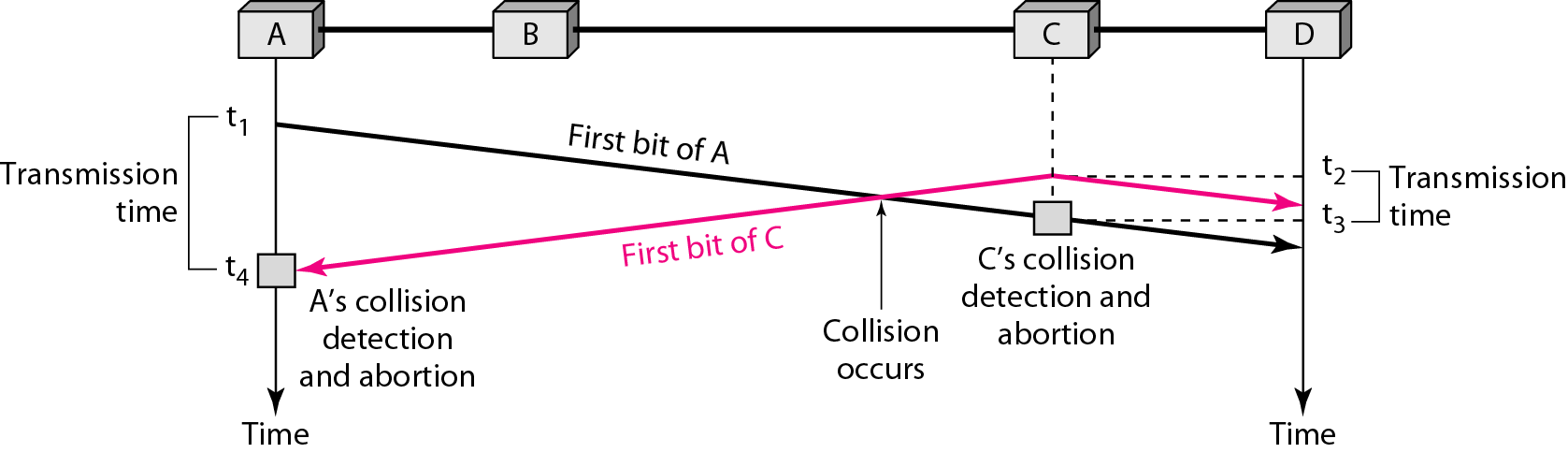
b. If the line is busy, it acts as though a collision has occurred and uses the backoff procedure.



**III. CARRIER SENSE MULTIPLE ACCESS / COLLISION DETECTION (CSMA/CD)**

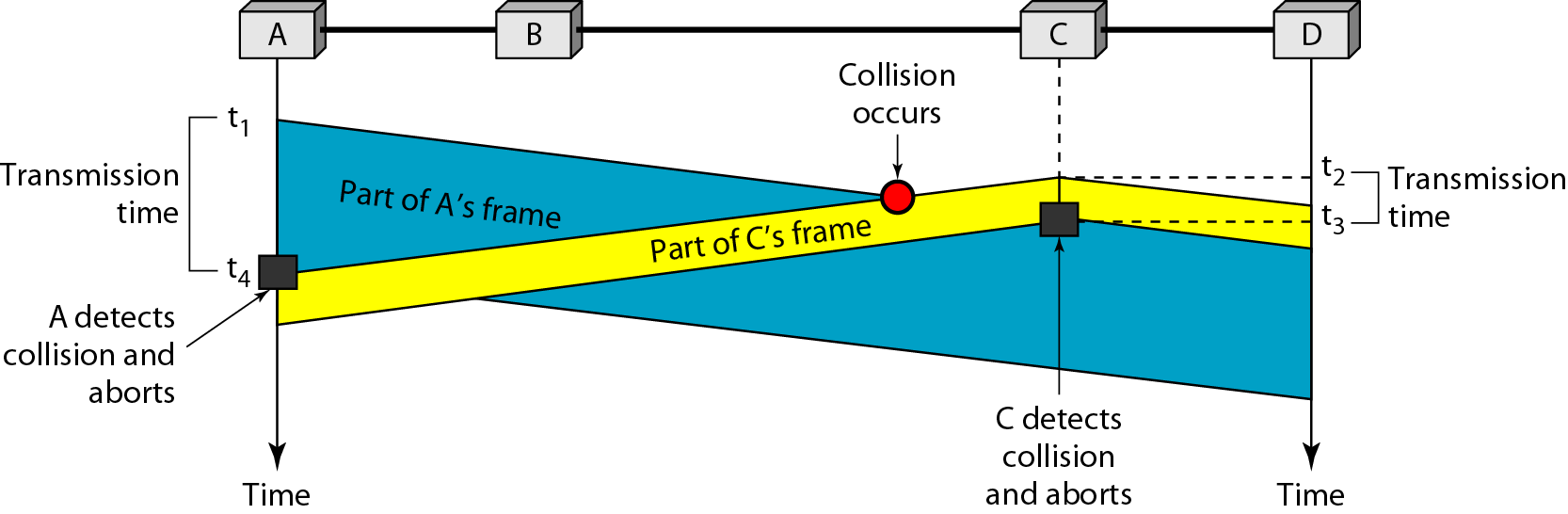
The CSMA method does not specify the procedure following a collision. Carrier sense multiple access with collision detection (CSMA/CD) augments the algorithm to handle the collision.

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.



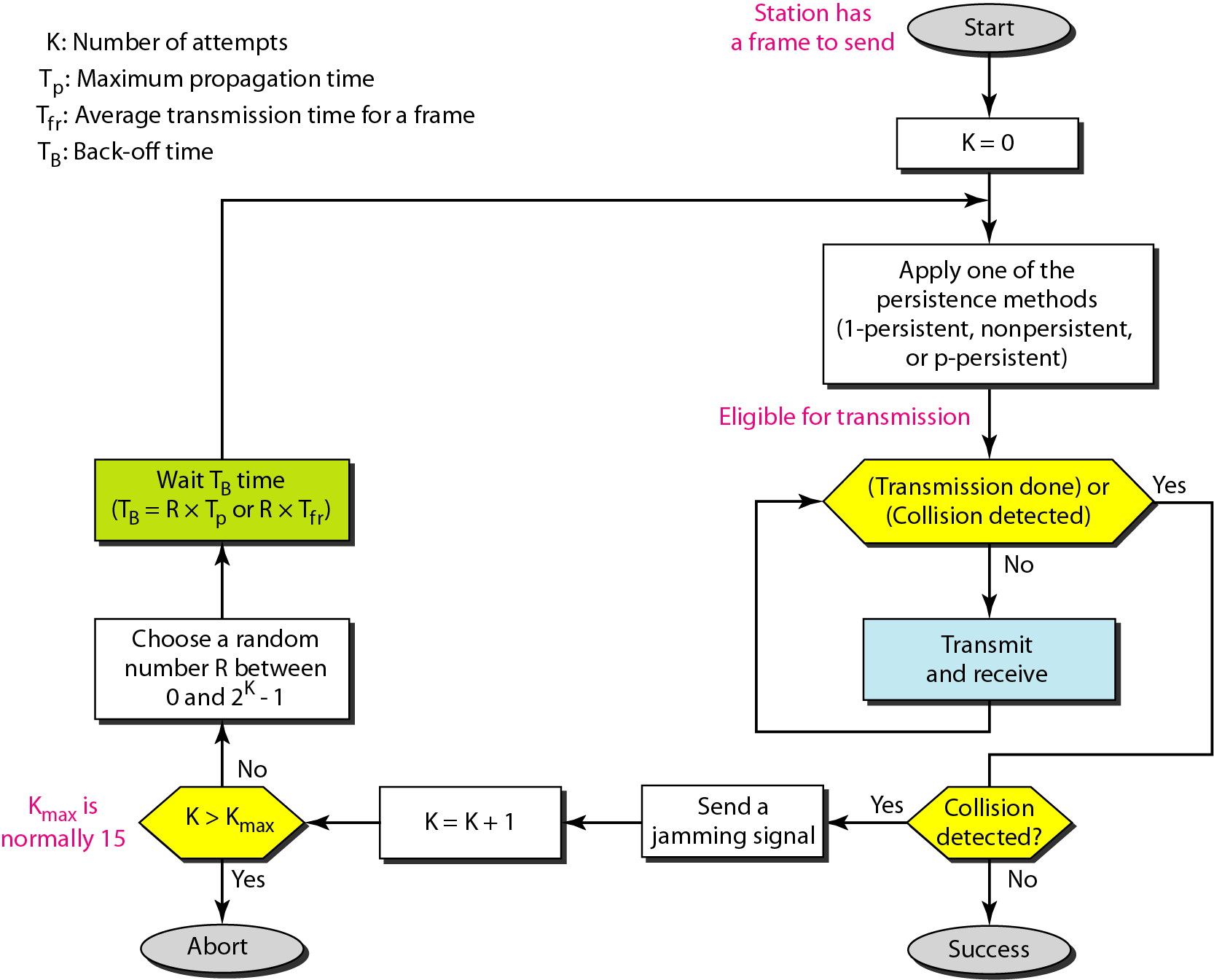
At time *t1*, station A has executed its persistence procedure and starts sending the bits of its frame. At time *t2,* station C has not yet sensed the first bit sent by A. Station C executes its persistence procedure and starts sending the bits in its frame, which propagate both to the left and to the 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 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.* At time *t4,* the transmission of *A’s* frame, though incomplete, is aborted; at time *t3,* the transmission of B's frame, though incomplete, is aborted.

**Collision and abortion in CSMA/CD**



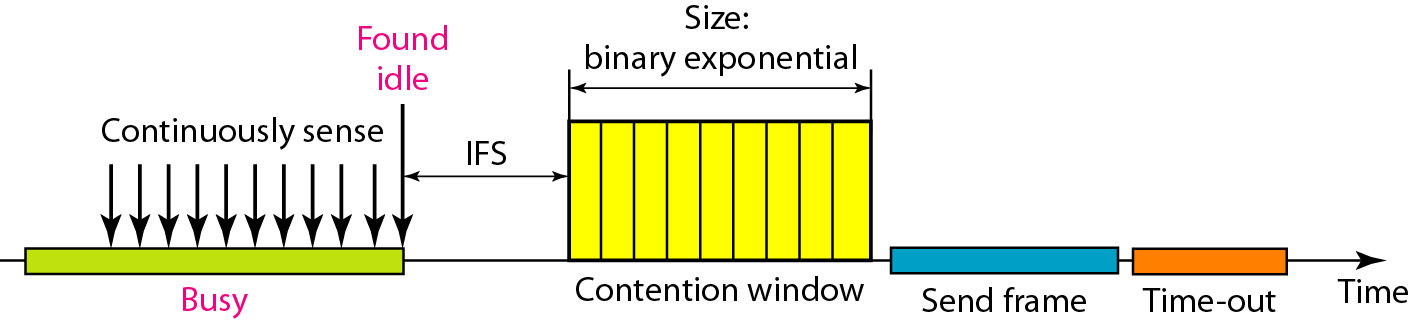
|  |
| --- |
| **Example:** A network using *CSMA/CD* has a bandwidth of 10 Mbps. If the maximum propagation time is 25.6µS, what is the minimum size of the frame?  **Solution:** The frame transmission time is *Tfr* = 2 x *Tp* =51.2 s. This means, in the worst case, a station needs to transmit for a period of 51.2 s to detect the collision. The minimum size of the frame is 10 Mbps x 51.2 µs =10\*106\*512\*10-6=512 bits or 64 bytes. |

**Flow diagram for the CSMA/CD**



**IV. CARRIER SENSE MULTIPLE ACCESS / COLLISION AVOIDANCE (CSMA/CA)**

We need to avoid collisions on wireless networks because they cannot be detected. Carrier sense multiple access with collision avoidance *(CSMA/CA)* was invented for this network. Collisions are avoided through the use of CSMAICA's three strategies: the inter frame space, the contention window, and acknowledgments, as shown in Figure.



**Interframe 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 interframe space or IFS.

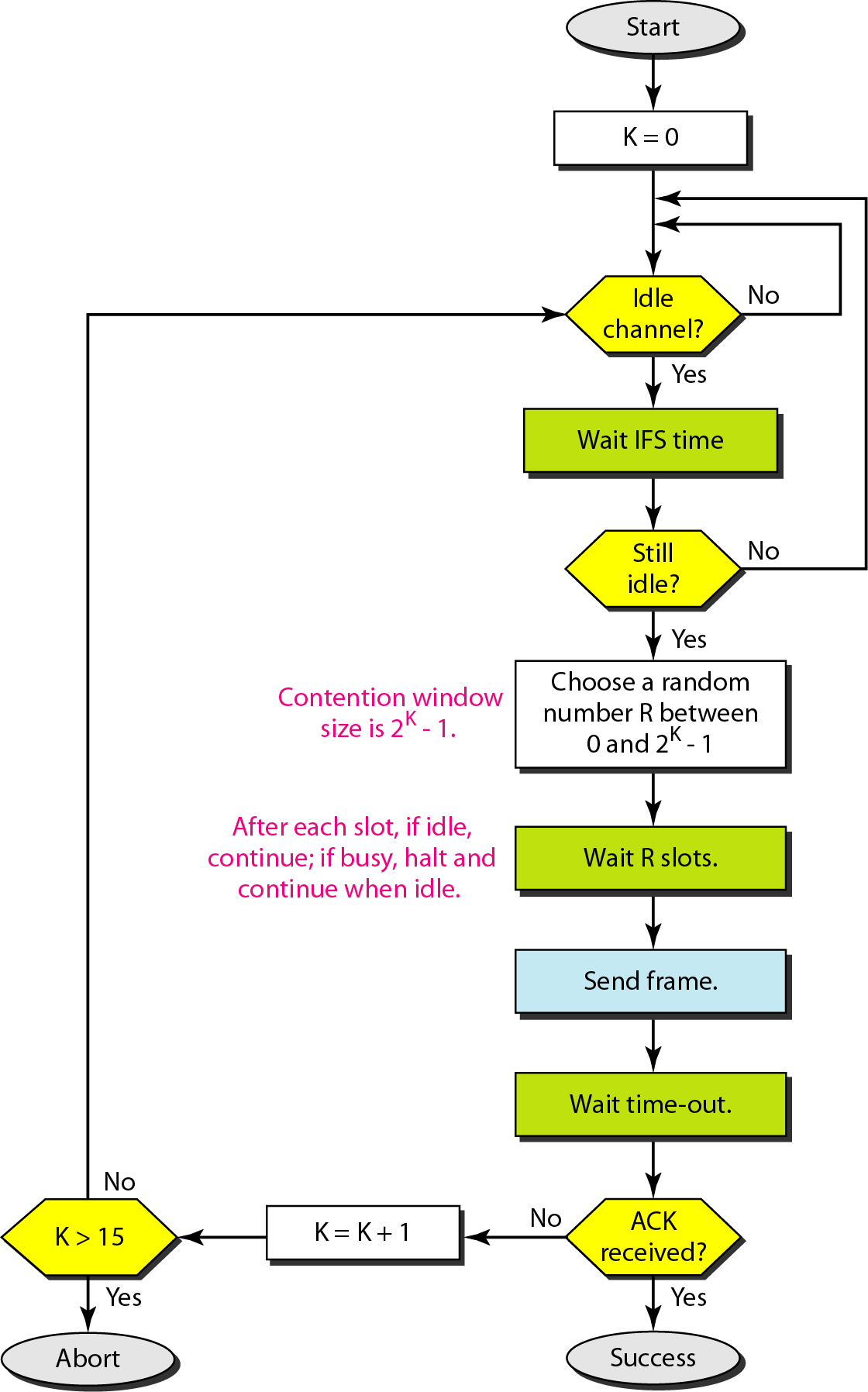
**Contention Window**

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.

**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.

**Flow diagram for CSMA/CA**

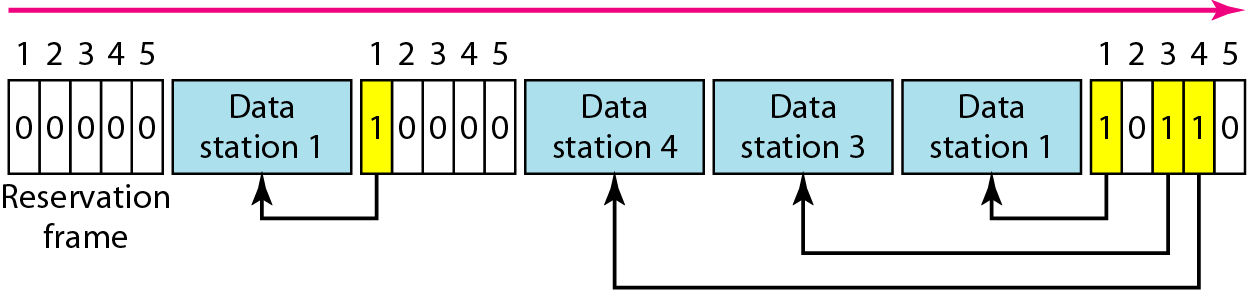


**2. CONTROLLED ACCESS PROTOCOLS**

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:

**I. 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.



**II. 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.

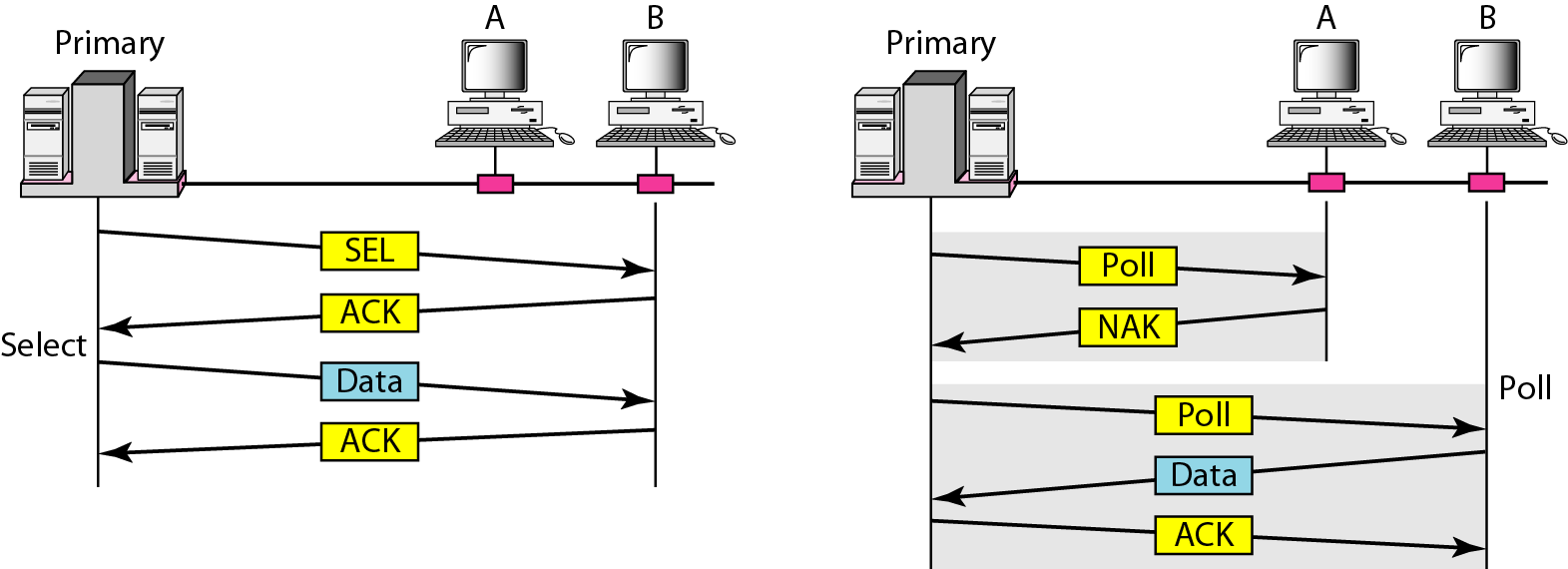
If the primary wants to receive data, it asks the secondary if they have anything to send; this is called poll function. If the primary wants to send data, it tells the secondary to get ready to receive; this is called select function.

**1. 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.

**2. 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.

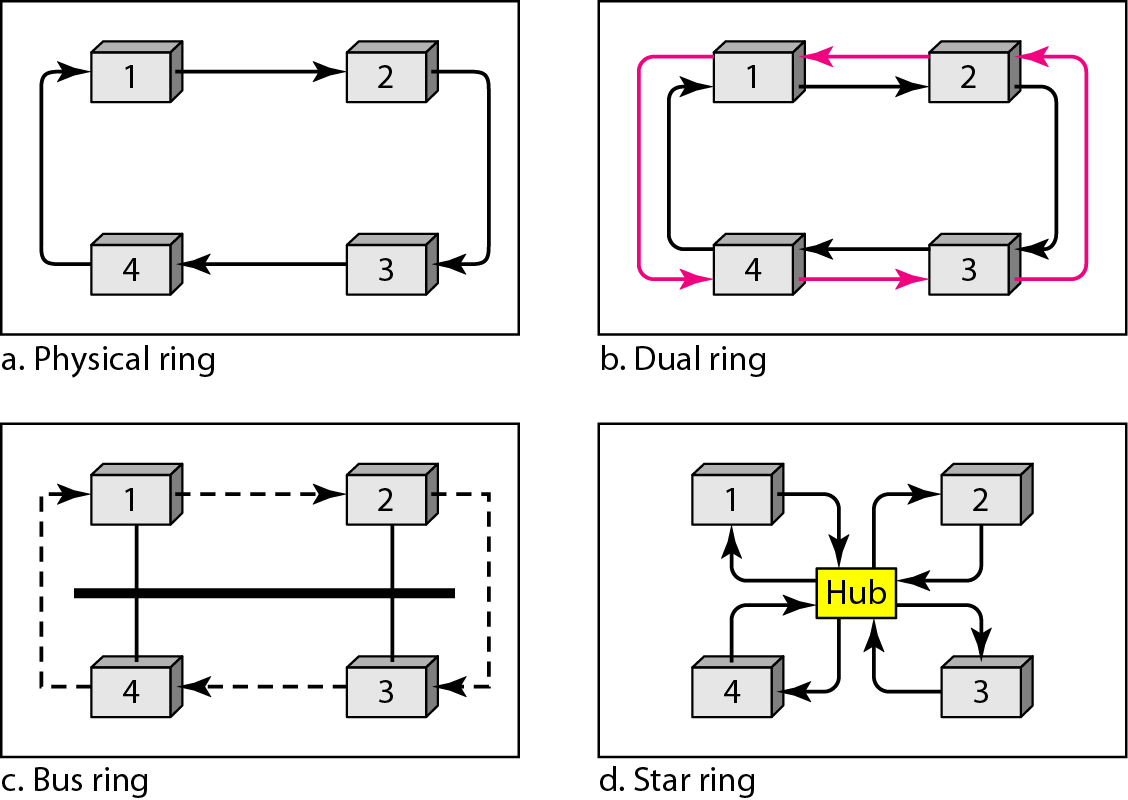


**III. Token Passing:** In the token-passing method, the stations in a network are organized in a logical ring. In other words, for each station, there is a *predecessor* and a *successor.* The predecessor is the station which is logically before the station in the ring; the successor is the station 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.

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 (4 tasks).

1. Stations must be limited in the time they can have possession of the token.
2. 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.
3. Assign priorities to the stations and to the types of data being transmitted.
4. And finally, token management is needed to make low-priority stations release the token to high priority stations.

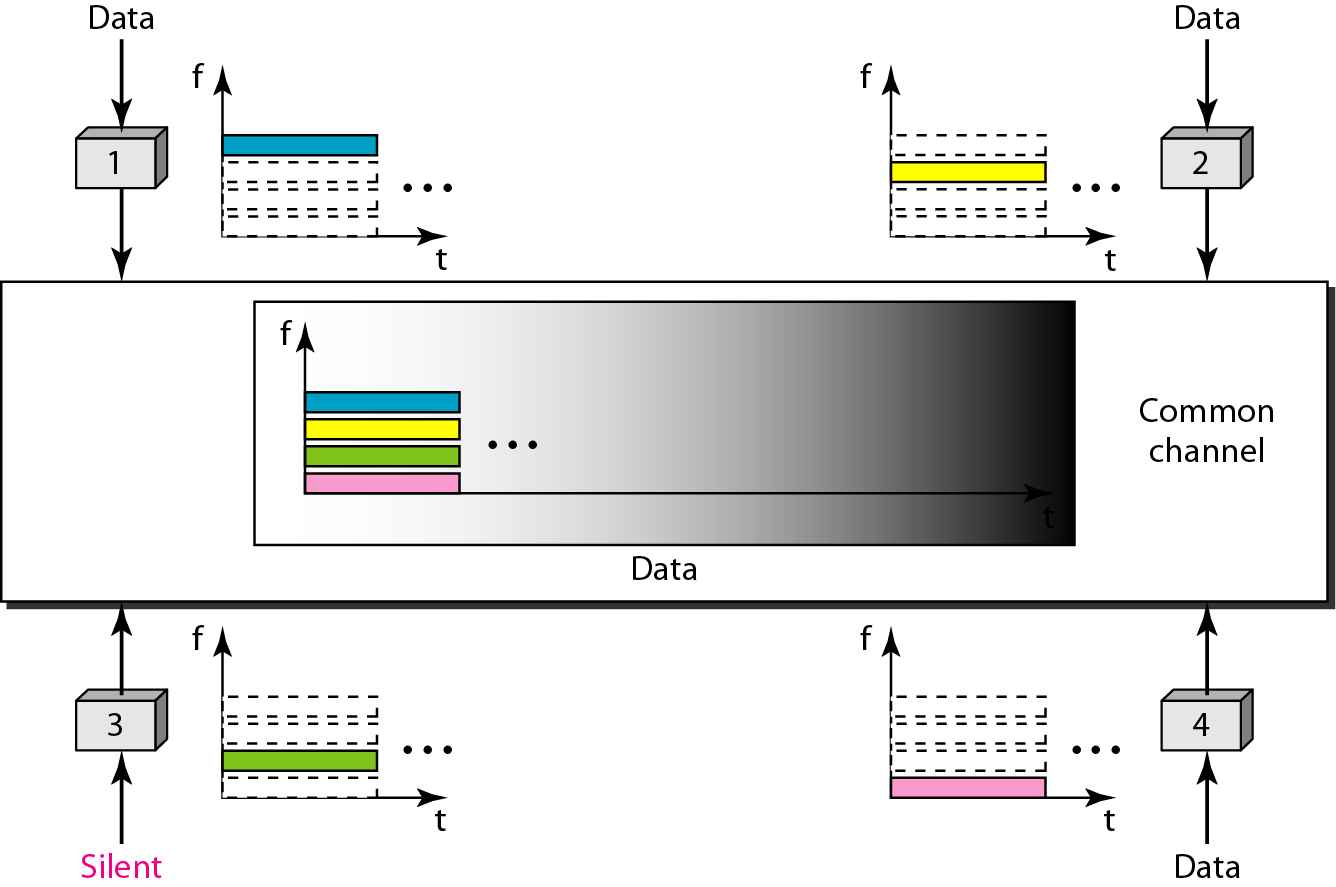


**3. CHANNELLIZATION PROTOCOLS**

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.

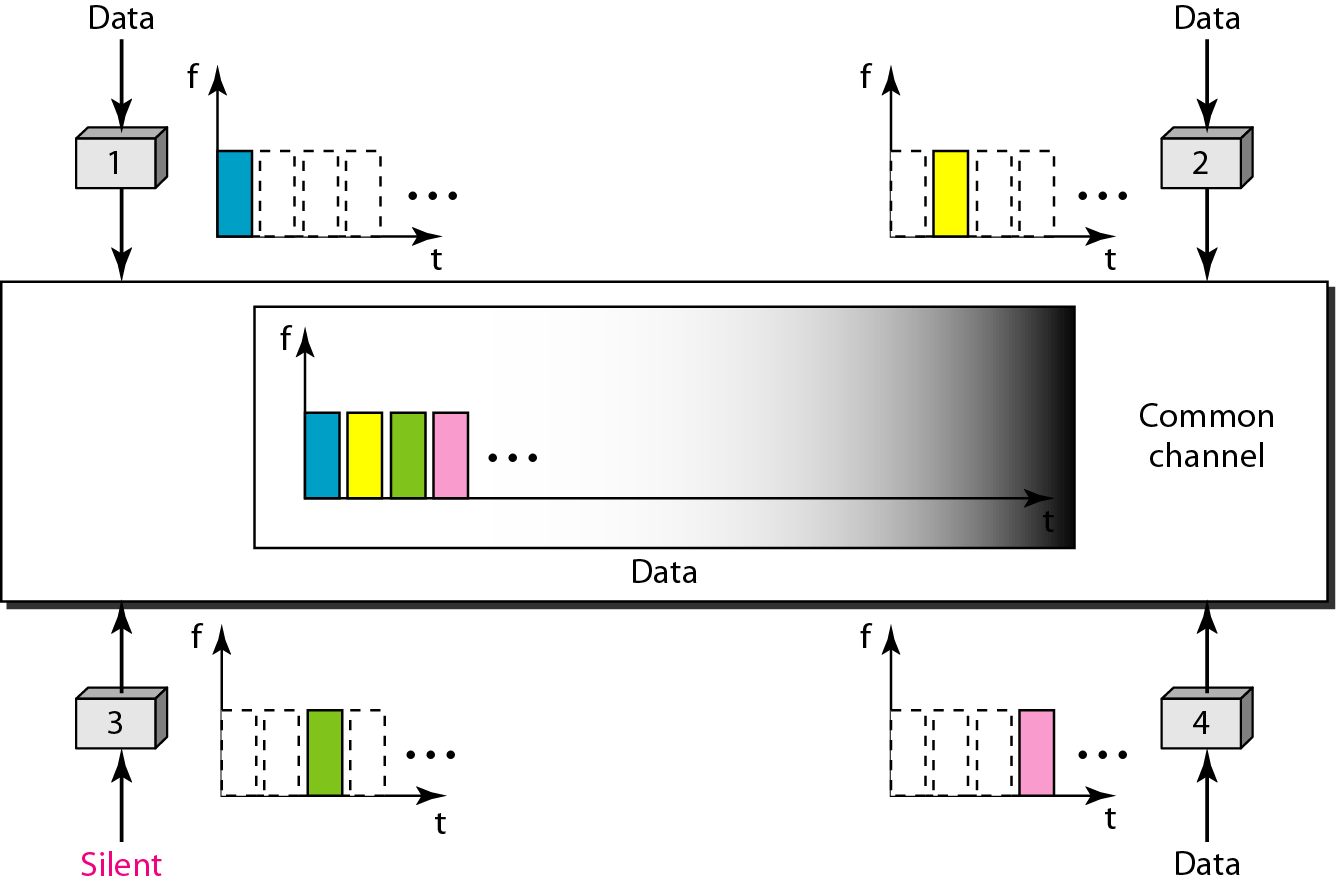
**I. Frequency-Division Multiple Access (FDMA)**

In frequency-division multiple access (FDMA), the available bandwidth is divided into frequency bands. Each station is allocated a band to send its data. Each station also uses a bandpass filter to confine the transmitter frequencies. To prevent station interferences, the allocated bands are separated from one another by small *guard bands.*



**II. 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 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 *guardtimes.* Synchronization is normally accomplished by having some synchronization bits(normally referred to as preamble bits) at the beginning of each slot.

**III. Code-Division Multiple Access (CDMA)**

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. If we multiply each code by another, we get 0.
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 below Figure.

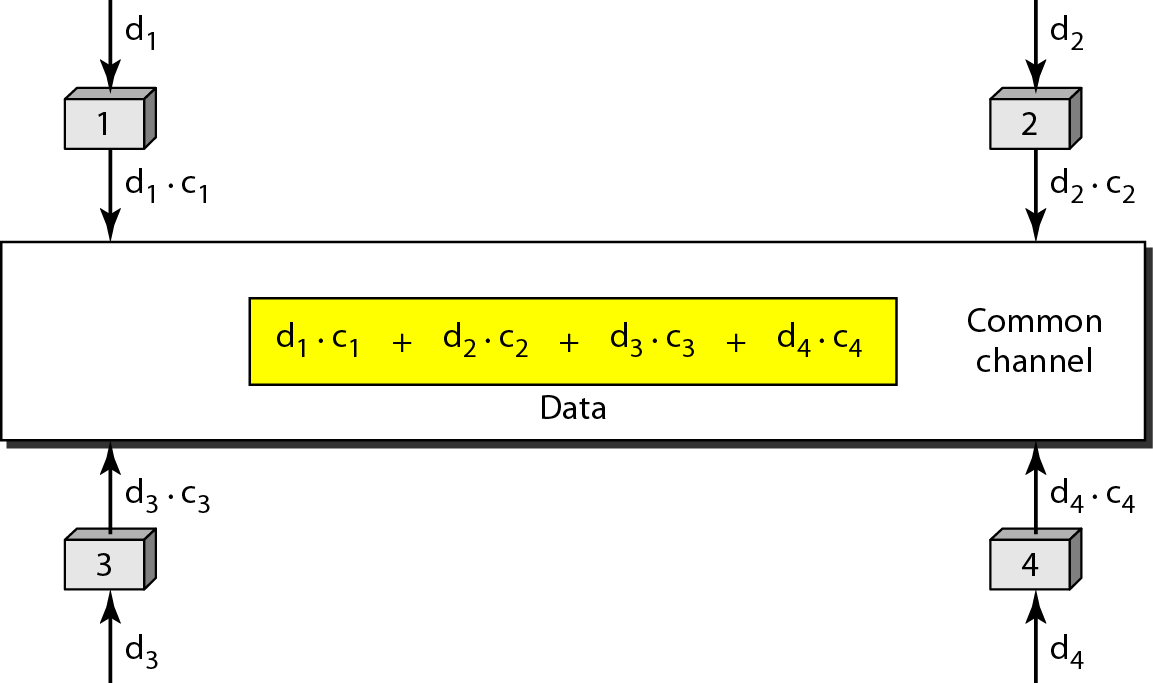
Station 1 multiplies its data by its code to get dl\*c1. Station 2 multiplies its data by its code to get *d2* \**c2* And so on. The data that go on the channel are the sum of all these terms, as shown in the box. Any station that wants to receive data from one of the other three multiplies the data 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 c1 the code of station 1.

Because (cl \*cl) is 4, but (c2\*c1), (c3\*c1), and (c4\*cl) are all 0s, station 2 divides the result by 4 to get the data from station 1.

Data =(d1\*c1 + d2\*c2 +d3 \*c3 + d4\*c4)\*c1

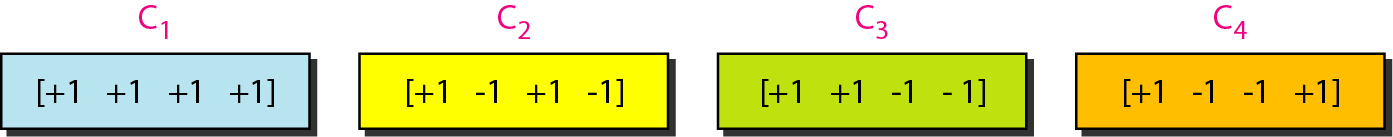
= (d1\*c1\*c1+ d2\*c2\*c1 +d3 \*c3\*c1 + d4\*c4\*c1)

= 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.



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]

1. 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

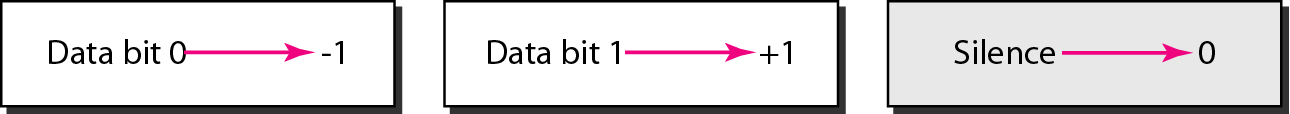
1. If we multiply two different sequences, element by element, and add the results, we get O. This is called inner product of two different sequences.

For example, [+1 +1 -1 -1] • [+1 +1 +1 +1] = 1 + 1 - 1 - 1 = 0

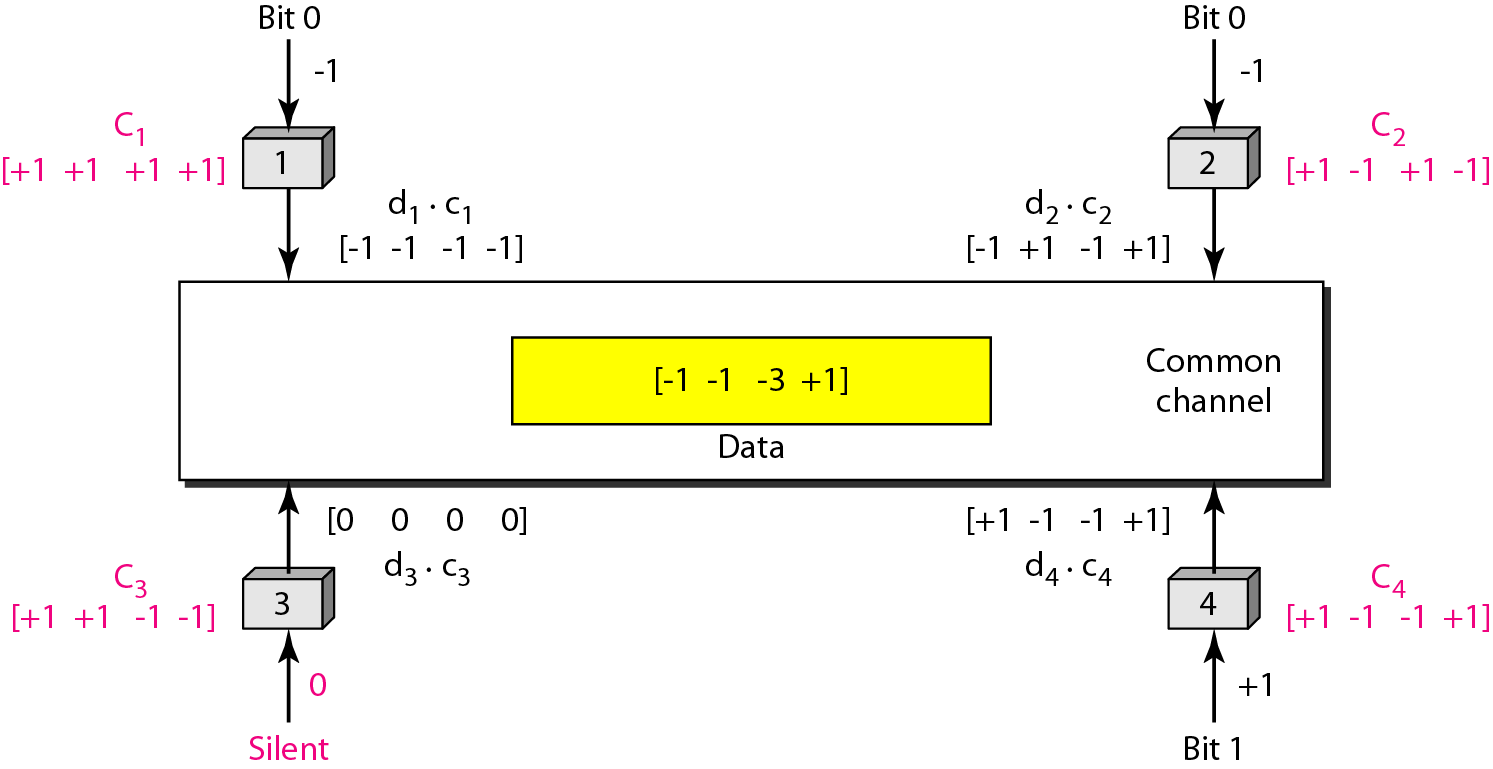
1. Adding two sequences means adding the corresponding elements. The result is another sequence. For example, [+1 +1 -1 -1] + [+1 +1 +1 +1]=[+2 +2 0 0]

**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.



**Encoding and Decoding**

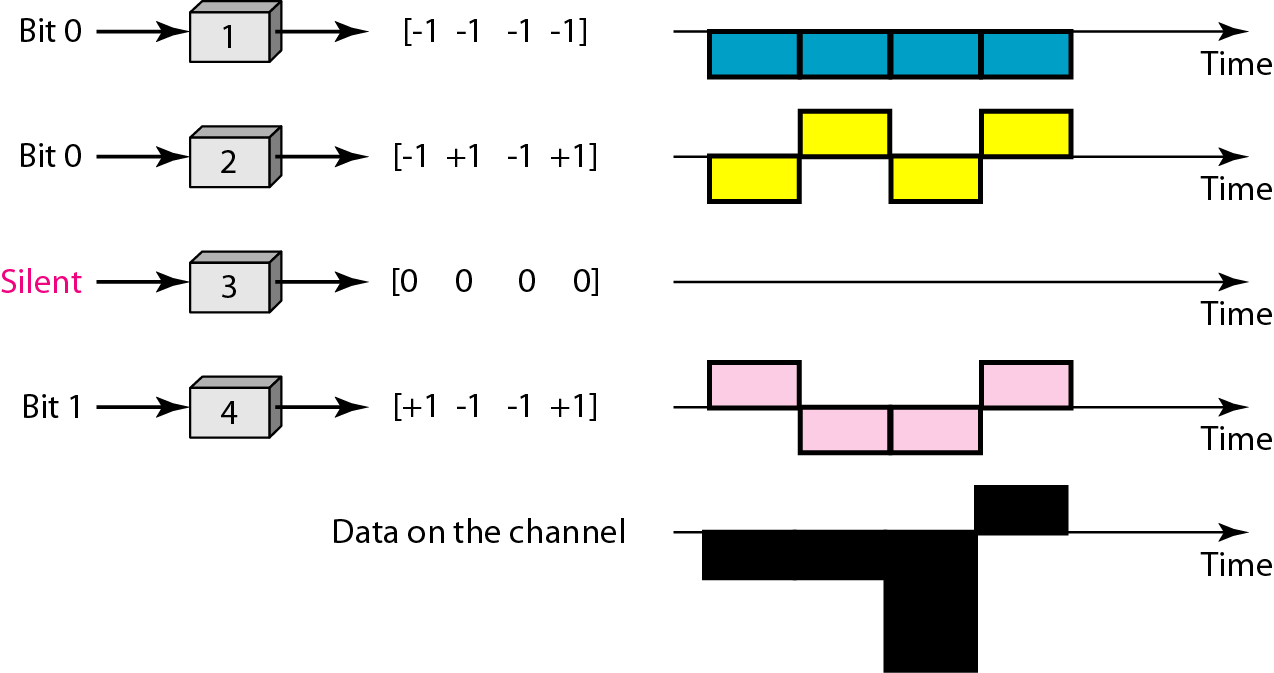


As a simple example, we can see how four stations share the link during a 1-bit interval. The procedure can easily be repeated for additional intervals. We assume that stations 1 and 2 are sending a 0 bit and channel 4 is sending a 1 bit. Station 3 is silent. The data at the sender site are translated 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. Figure shows the situation.

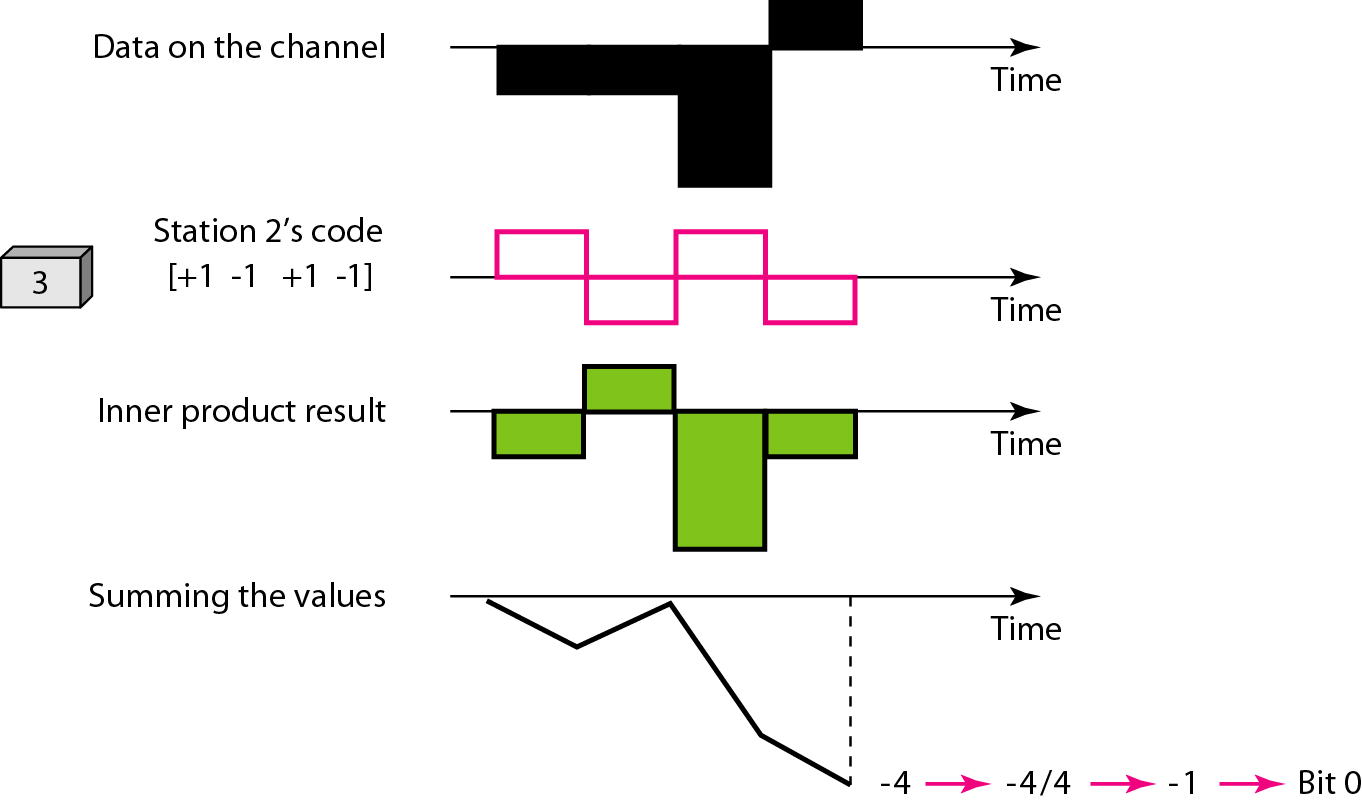
Now imagine station 3, which we said is silent, is listening to station 2. Station 3 multiplies the total data on the channel by the code for station 2, which is [+1 -1 +1-1], to get [-1-1-3 +1]· [+1-1 +1-1] =-4/4 =-1 ...... bit 0

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

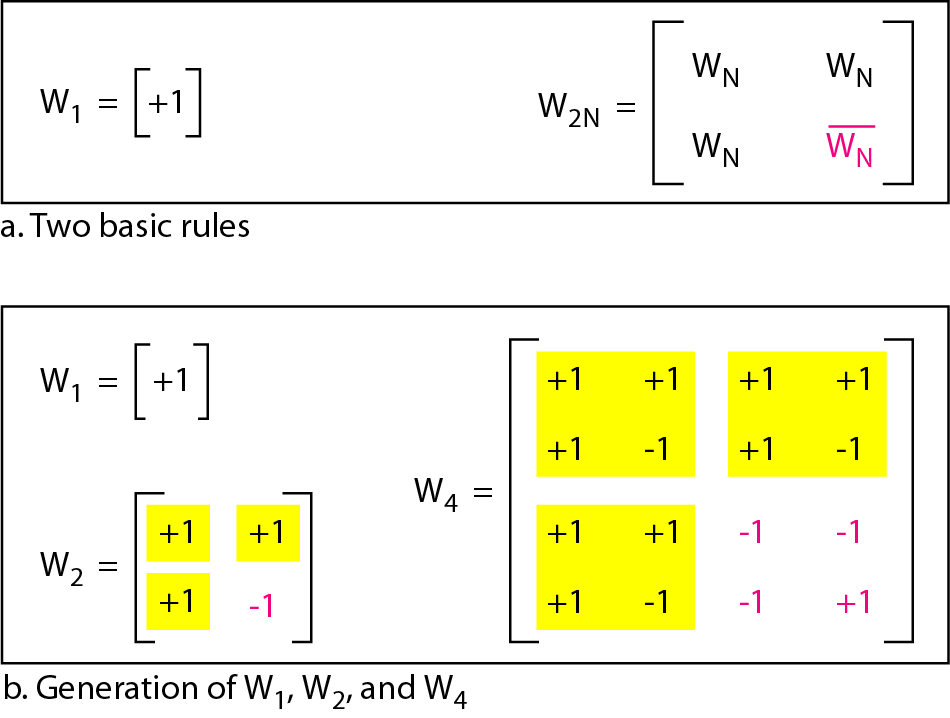


Below figure shows how station 3 can detect the data sent by station 2 by using the code for station 2. The total data on the channel are multiplied (inner product operation) by the signal representing station 2 chip code to get a new signal. The station then integrates and adds the area under the signal,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



**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. W1 for a one-chip sequence has one row and one column. We can choose –l or +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. The WN with the over bar WN stands for the complement of WN where each +1 is changed to -1 and vice versa.

Figure also shows how we can create W2 and W4 from W1.After we select W1, W2 can be made from four W1’s, with the last one the complement of W1.After W2 is generated, W4 can be made of four W2's, with the last one the complement of W2. Of course, W8 is composed of four W4's, and so on. Note that after WN 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 2.

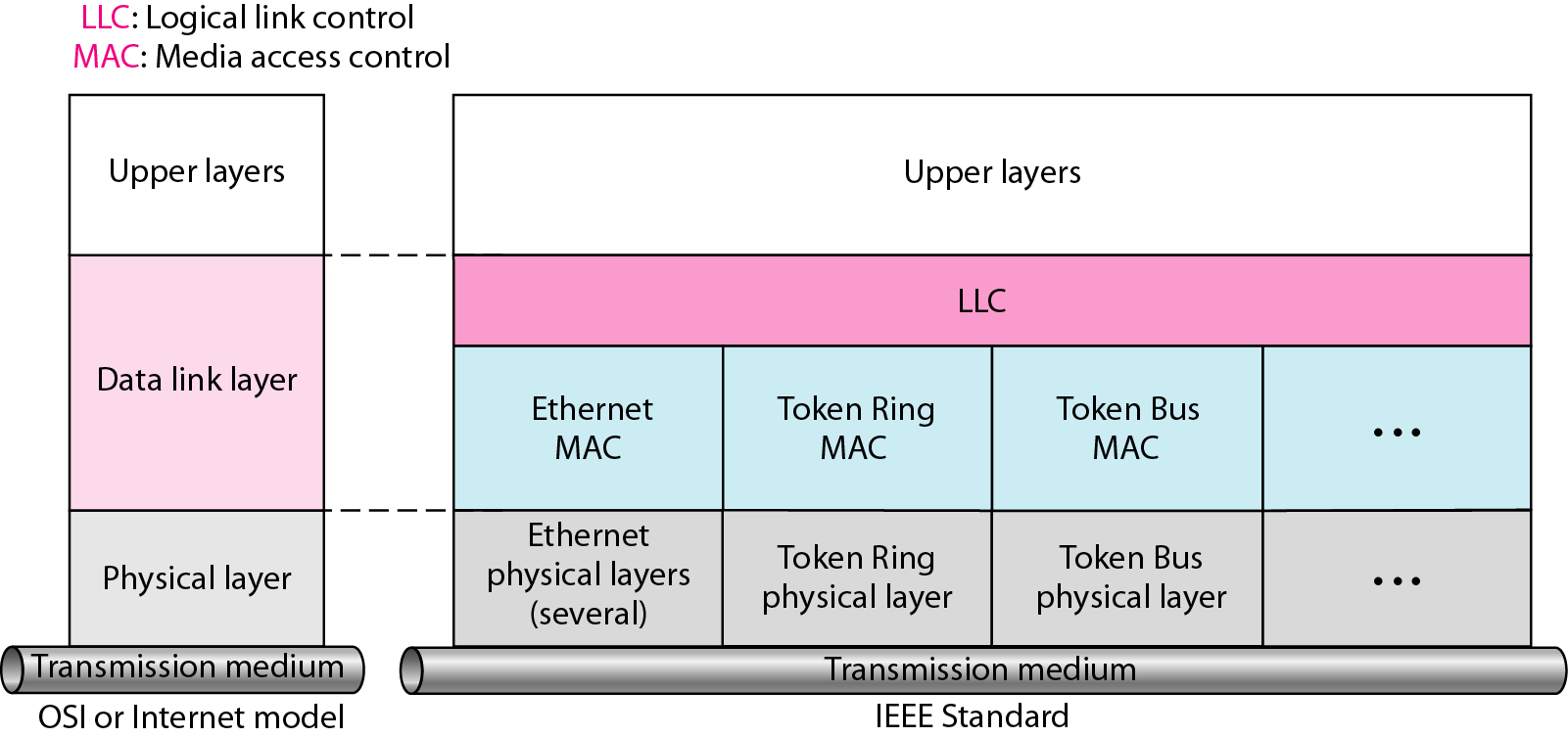
In other words, we need to have N = 2m*.*

|  |
| --- |
| **Example:** Find the chips for a network with   1. Two stations 2. Four stations   **Solution:**  We can use the rows of *W2* and *W4* in above Figure:   1. For a two-station network, we have [+1 +1] and [+1 -1]. 2. For a four-station network we have [+1 +1 +1 +1], [+1 -1 +1 -1], [+1 +1 -1 -1], and [+1-1-1 +1].   **Example:** What is the number of sequences if we have 90 stations in our network?  **Solution:**  The number of sequences needs to be *2m.* We need to choose *m* = 7 and *N* = 27 or 128. We can then use 90 of the sequences as the chips.  **Example:** Prove that a receiving station can get the data sent by a specific sender if it multiplies the entire data on the channel by the sender's chip code and then divides it by the number of stations.  **Solution**  Let us prove this for the first station, using our previous four-station example. We can say that the data on the channel  D = (d1⋅ c1 + d2⋅ c2 + d3⋅ c3 + d4⋅ c4).  The receiver which wants to get the data sent by station 1 multiplies these data by c1.    When we divide the result by N, we get d1. |

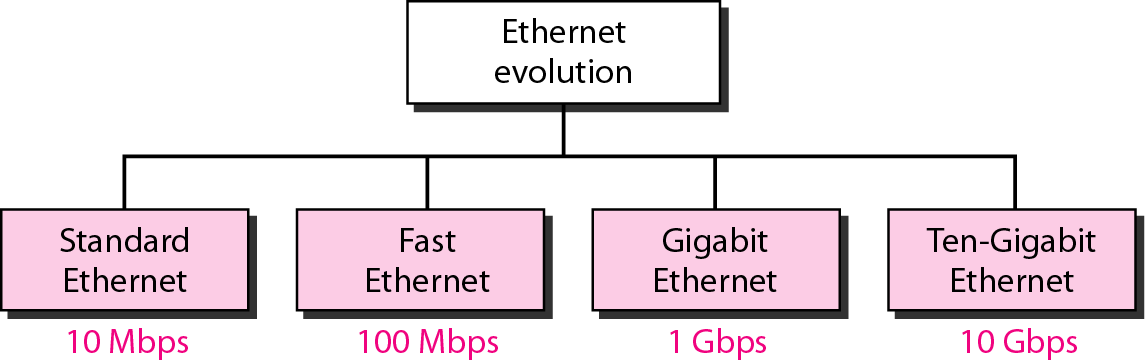
**WIRED LAN’s: ETHERNET**

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

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



The original Ethernet was created in 1976 at Xerox's Palo Alto Research Center (PARC). Since then, it has gone through four generations: Standard Ethernet (10 Mbps), Fast Ethernet (100 Mbps), Gigabit Ethernet (1 Gbps), and Ten-Gigabit Ethernet (10 Gbps).

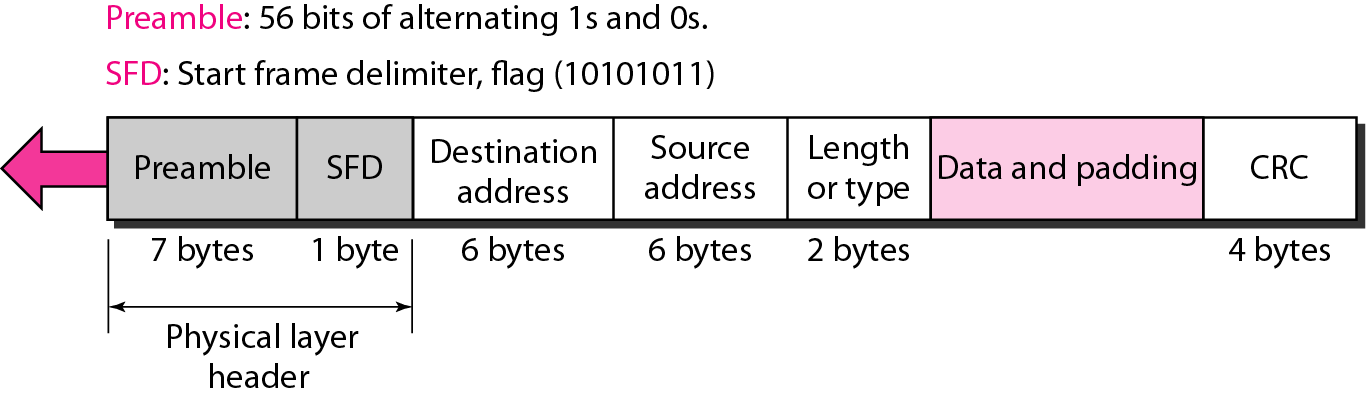


**MAC SUBLAYER**

In Standard Ethernet, the MAC sublayer governs the operation of the access method. It also frames data received from the upper layer and passes them to the physical layer.

**802.3 MAC Frame Format**

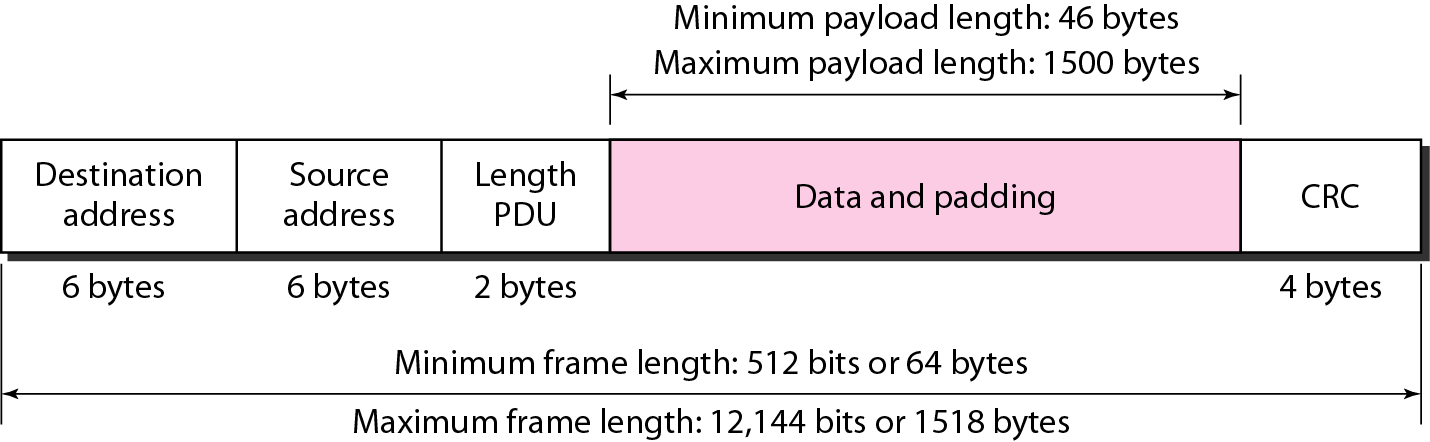
The Ethernet frame contains seven fields: preamble, Start Frame Delimiter (SFD), Destination Address, Source Address, length or type of protocol data unit (PDU), upper-layer data, and the CRC. Ethernet does not provide any mechanism for acknowledging received frames, making it what is known as an unreliable medium. Acknowledgments must be implemented at the higher layers. The format of the MAC frame is shown in below figure:



* **Preamble:** The first field of the 802.3 frame contains 7 bytes (56 bits) of alternating 0s and 1s that alerts the receiving system to the coming frame and enables it to synchronize its input timing.
* **Start frame delimiter (SFD):** The second field (1byte: 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 is 11 and alerts the receiver that the next field is the destination address.
* **Destination address (DA):** The DA field is 6 bytes and contains the physical address of the destination station or stations to receive the packet.
* **Source address (SA):** The SA field is also 6 bytes and contains the physical address of the sender of the packet.
* **Length or type:** This field is defined as a type field or length field. The original Ethernet used this field as the type field to define the upper-layer protocol using the MAC frame. The IEEE standard used it as the length field to define the number of bytes in the data field. Both uses are common today.
* **Data:** This field carries data encapsulated from the upper-layer protocols. It is a minimum of 46 and a maximum of 1500 bytes.
* **CRC:** The last field contains error detection information, in this case a CRC-32.

**Frame Length**

Ethernet has imposed restrictions on both the minimum and maximum lengths of a frame, as shown in Figure.



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.

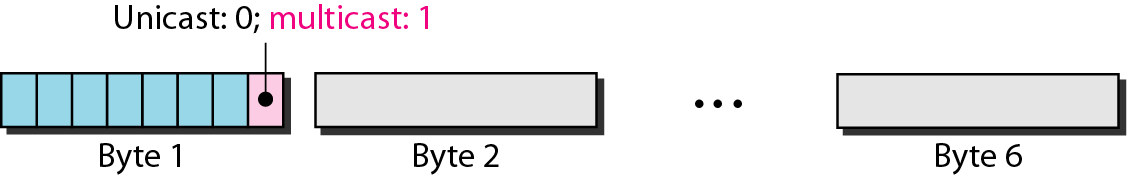
**Addressing**

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 6-byte physical address.

Example of an Ethernet address in hexadecimal notation



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. Figure 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. If all digits are F, then address is broadcast.



The least significant bit of the first byte defines the type of address. If the bit is 0, the address is unicast; otherwise, it is multicast.

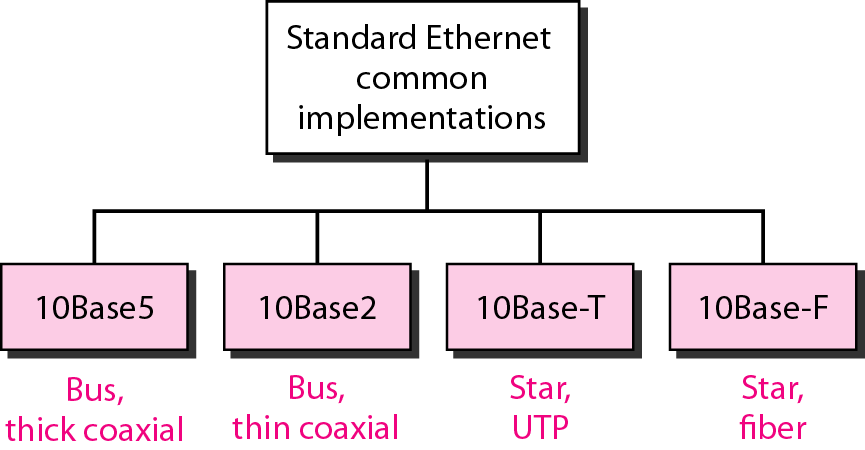
|  |
| --- |
| **Example**  Define the type of the following destination addresses:   1. 4A:30:10:21:1O:1A 2. 47:20:1B:2E:08:EE 3. FF:FF:FF:FF:FF:FF   **Solution**  To find the type of the address, we need to look at the second hexadecimal digit from the left. If it is even, the address is unicast. If it is odd, the address is multicast. If all digits are F's, the address is broadcast. Therefore, we have the following:   1. This is a unicast address because A in binary is 1010 (even). 2. This is a multicast address because 7 in binary is 0111 (odd). 3. This is a broadcast address because all digits are F's.   The way the addresses are sent out on line 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.  **Example**  Show how the address 47:20:1B:2E:08: EE is sent out on line.  **Solution**  The address is sent left-to-right, byte by byte; for each byte, it is sent right-to-left, bit by bit, as  shown below:  🡨11100010 00000100 11011000 01110100 00010000 01110111 |

**Standard Ethernet(Physical Layer)**

The Standard Ethernet defines several physical layer implementations; four of the most common, are shown in below figure.

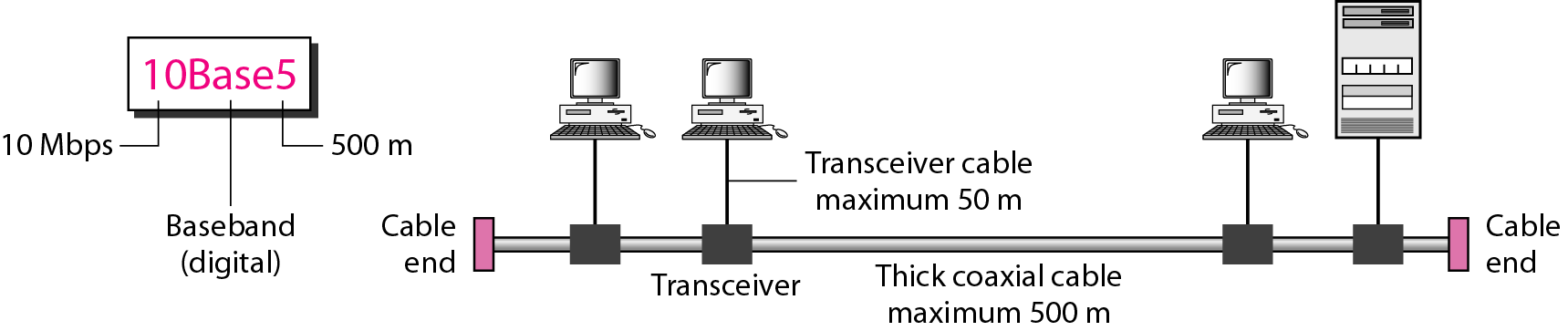
**Encoding and Decoding**

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.



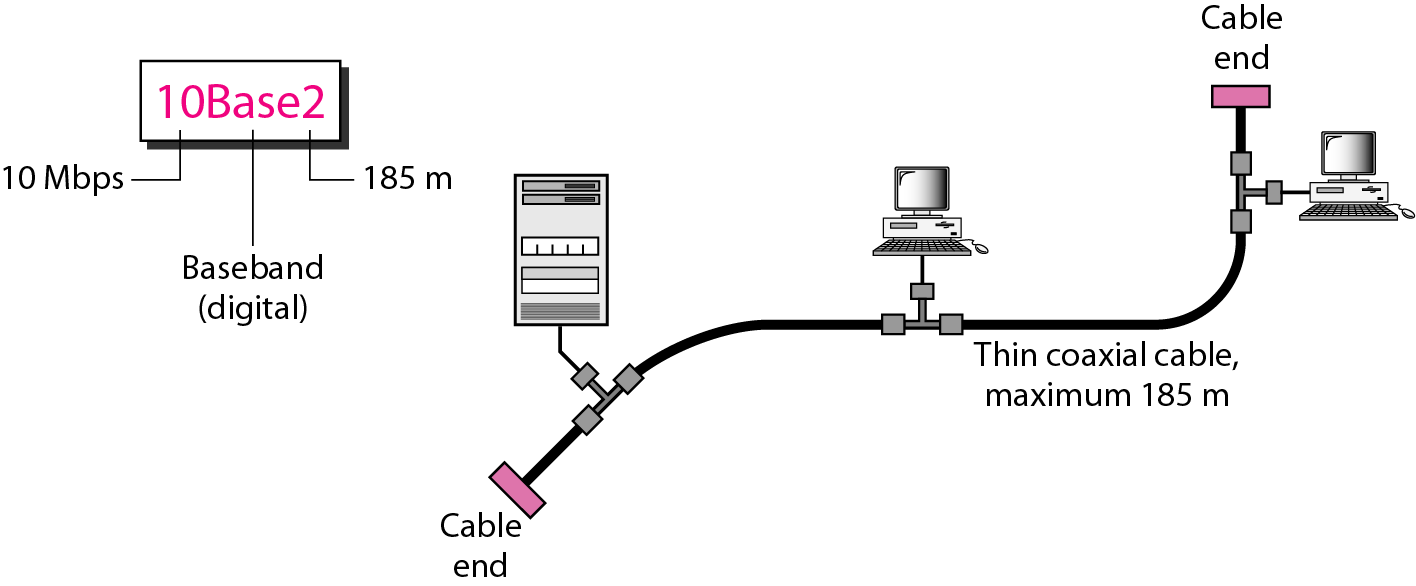
**10Base5: Thick Ethernet**

* The first implementation is called **10Base5, thick Ethernet, or Thicknet.**
* 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.
* 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.
* The maximum length of the coaxial cable must not exceed *500* m, otherwise, there is excessive degradation of the signal.



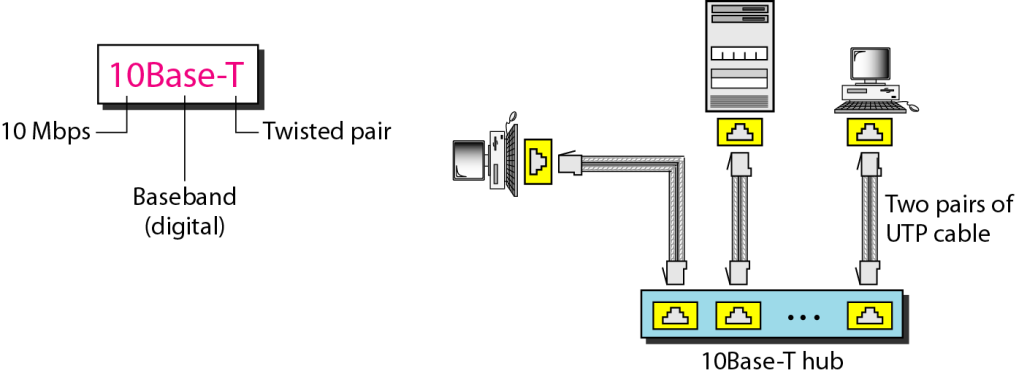
**10Base2: Thin Ethernet**

* The second implementation is called **10Base2**, **thin Ethernet**, or **Cheapernet**.
* 10Base2 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.
* 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 185m (close to 200 m) due to the high level of attenuation in thin coaxial cable.

****

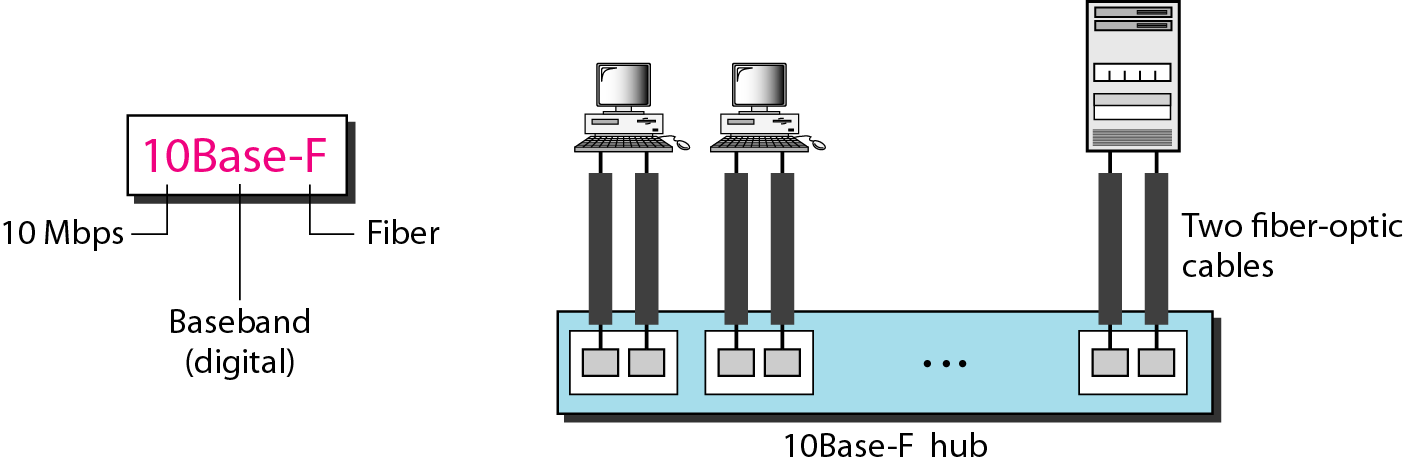
**10Base-T: Twisted-Pair Ethernet**

* The third implementation is called 10Base-T or twisted-pair Ethernet. 10Base-T uses a physical star topology.
* Any collision here happens in the hub.
* Compared to 10Base5 or l0Base2, 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 10Base-F.
* 10Base-F uses a star topology to connect stations to a hub.



**Summary of Standard Ethernet implementations**



**CHANGES IN THE STANDARD**

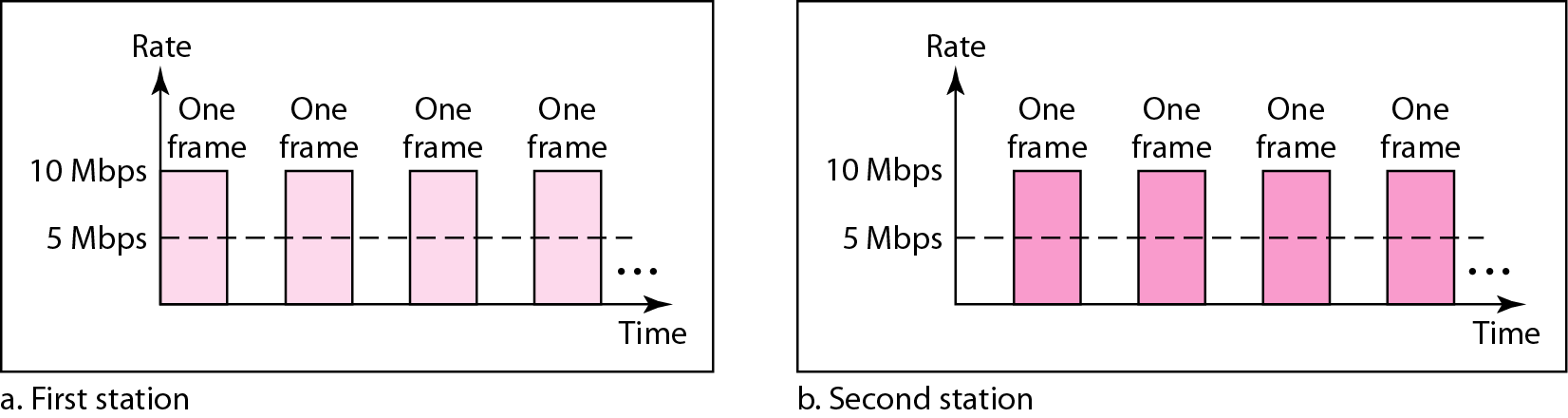
The 10-Mbps Standard Ethernet has gone through several changes before moving to the higher data rates.

**Bridged Ethernet**

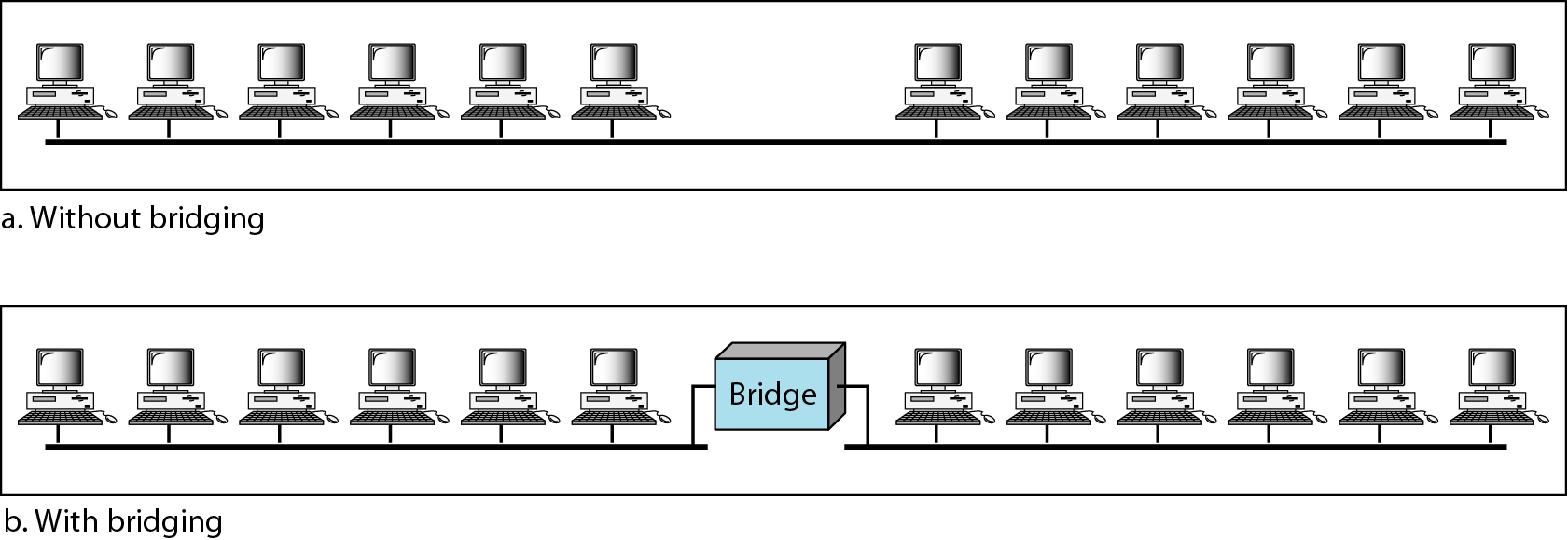
Bridges have two effects on an Ethernet LAN: They raise the bandwidth and they separate collision domains.

**Raising the Bandwidth**

In an unbridged Ethernet network, the total capacity (10 Mbps) is shared among all stations with a frame to send; the stations share the bandwidth of the network. If only one station has frames to send, it benefits from the total capacity (10 Mbps). But if more than one station needs to use the network, the capacity is shared. For example, if two stations have a lot of frames to send, they probably alternate in usage. When one station is sending, the other one refrains from sending. We can say that, in this case, each station on average, sends at a rate of 5 Mbps. Figure shows the situation.

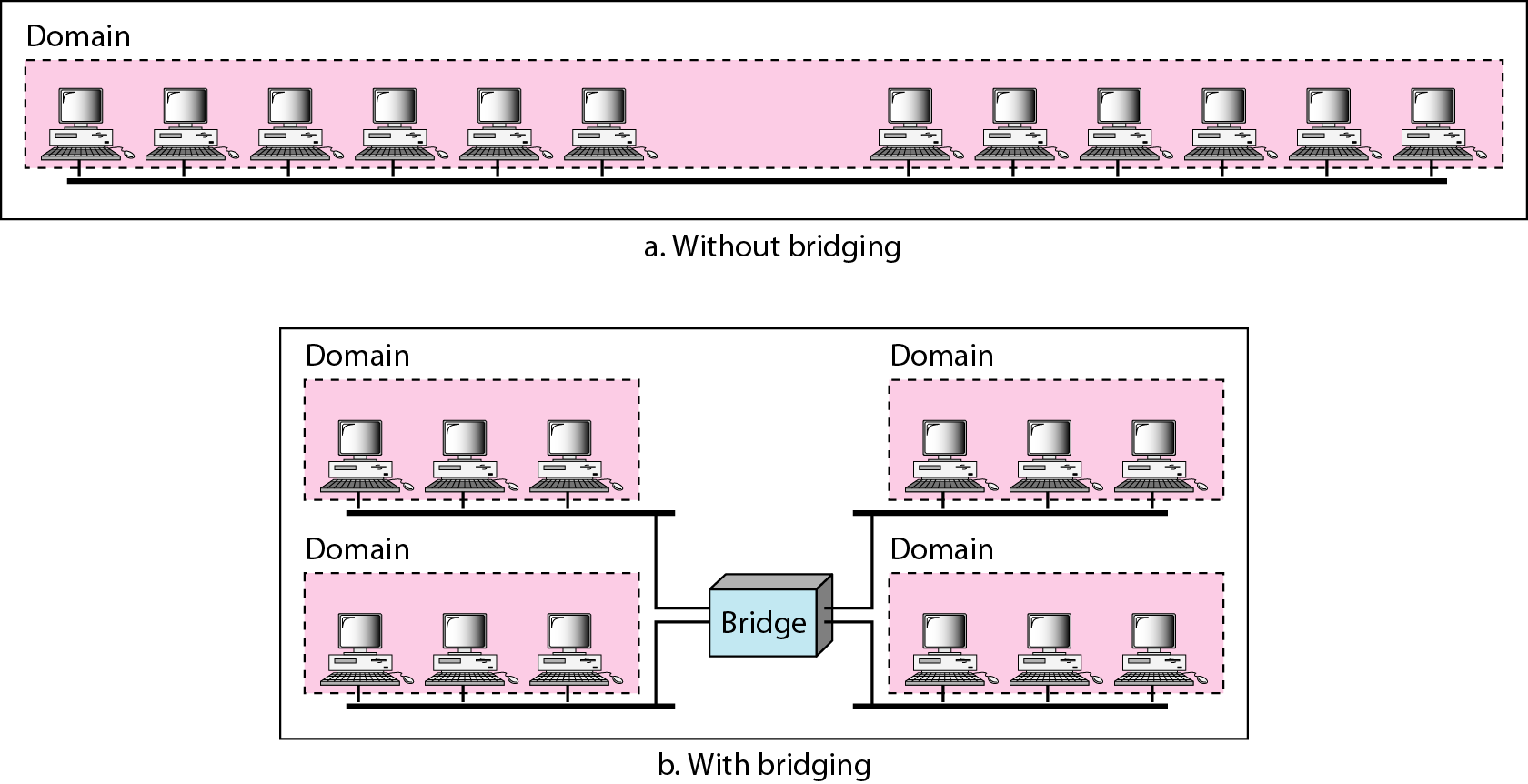


A bridge divides the network into two or more networks. Bandwidth-wise, each network is independent. For example, in below Figure, a network with 12 stations is divided into two networks, each with 6 stations. Now each network has a capacity of 10 Mbps. The 10-Mbps capacity in each segment is now shared between 6 stations (actually 7 because the bridge acts as a station in each segment), not 12 stations. In a network with a heavy load, each station theoretically is offered 10/6 Mbps instead of 10/12 Mbps, assuming that the traffic is not going through the bridge.



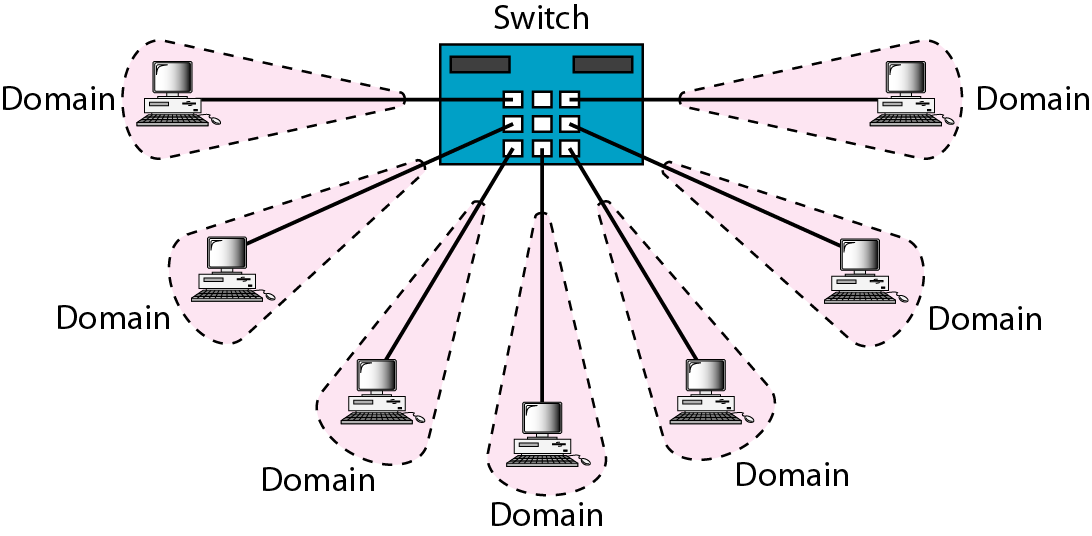
**Separating Collision Domains**

Another advantage of a bridge is the separation of the collision domain. Figure shows the collision domains for an unbridged and a bridged network. You can see that the collision domain becomes much smaller and the probability of collision is reduced tremendously. Without bridging, 12 stations contend for access to the medium; with bridging only 3 stations contend for access to the medium.



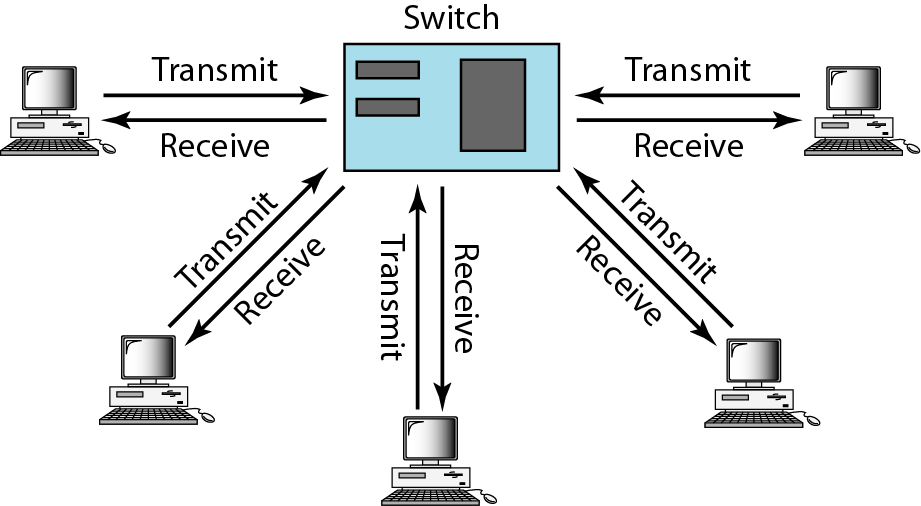
**Switched Ethernet**

The idea of a bridged LAN can be extended to a switched LAN. Instead of having two to four networks, why not have *N* networks, where *N* is the number of stations on the LAN? In other words, if we can have a multiple-port bridge, why not have an *N-port* switch? In this way, the bandwidth is shared only between the station and the switch (5 Mbps each). In addition, the collision domain is divided into *N* domains. A layer 2 switch is an *N-port* bridge with additional sophistication that allows faster handling of the packets. Evolution from a bridged Ethernet to a switched Ethernet was a big step that opened the way to an even faster Ethernet, as we will see. Figure shows a switched LAN.



**Full-Duplex Ethernet**

One of the limitations of 10Base5 and 10Base2 is that communication is half-duplex (l0Base-T is always full-duplex); a station can either send or receive, but may not do both at the same time. The next step in the evolution was to move from switched Ethernet to full-duplex switched Ethernet. The full-duplex mode increases the capacity of each domain from 10 to 20 Mbps. Figure shows a switched Ethernet in full-duplex mode. Note that instead of using one link between the station and the switch, the configuration uses two links: one to transmit and one to receive.



**FAST ETHERNET**

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

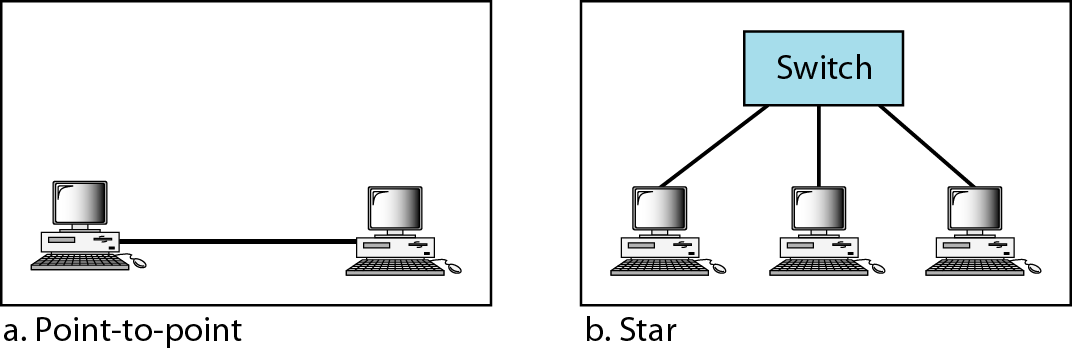
1. Upgrade the data rate to 100 Mbps.
2. Make it compatible with Standard Ethernet.
3. Keep the same 48-bit address.
4. Keep the same frame format.
5. Keep the same minimum and maximum frame lengths.

**Physical Layer**

The physical layer in Fast Ethernet is more complicated than the one in Standard Ethernet.

We briefly discuss some features of this layer.

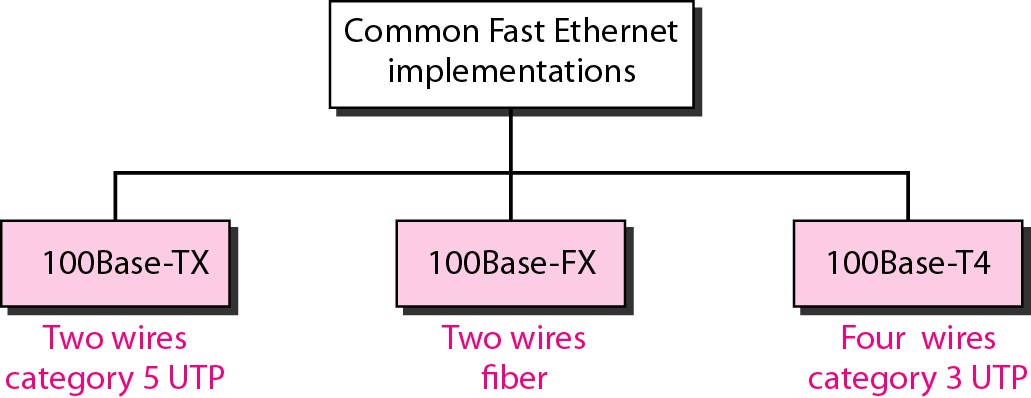
* **Topology:** Fast Ethernet is designed to connect two or more stations together. If there are only two stations, they can be connected point-to-point. Three or more stations need to be connected in a star topology with a hub or a switch at the center.



* **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 such as NRZ-I, 4B/5B block coding.

**Implementation**

Fast Ethernet implementation at the physical layer can be categorized as either two-wire or four-wire. The two-wire implementation can be either category 5 UTP (100Base-TX) or fiber-optic cable (100Base-FX). The four-wire implementation is designed only for category 3 UTP (l00Base-T4).



**GIGABIT ETHERNET**

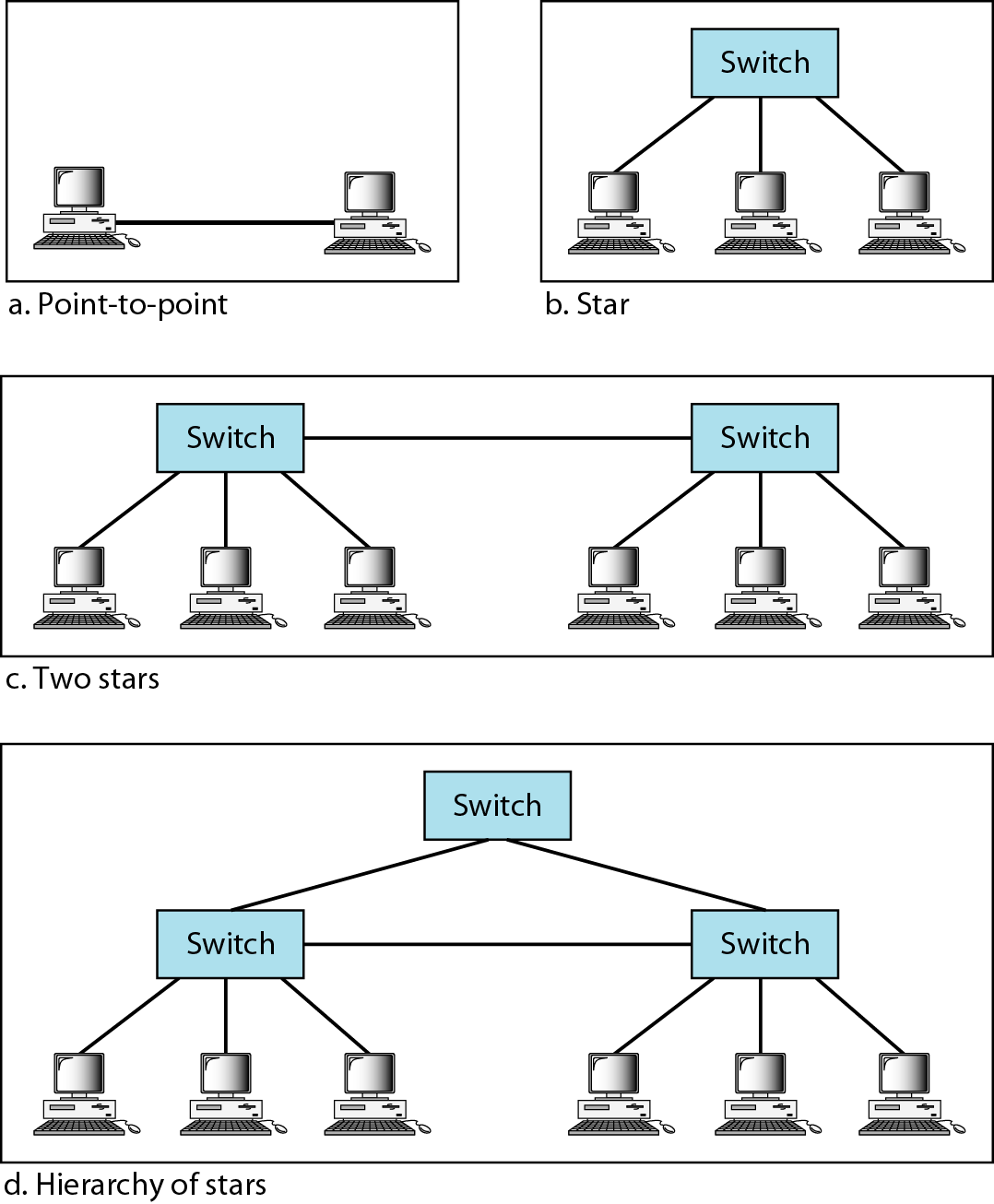
The need for an even higher data rate resulted in the design of the Gigabit Ethernet protocol (1000 Mbps). The IEEE committee calls the Standard 802.3z. The goals of the Gigabit Ethernet design can be summarized as follows:

1. Upgrade the data rate to 1 Gbps.
2. Make it compatible with Standard or Fast Ethernet.
3. Use the same 48-bit address.
4. Use the same frame format.
5. Keep the same minimum and maximum frame lengths.
6. To support auto negotiation as defined in Fast Ethernet.

**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.

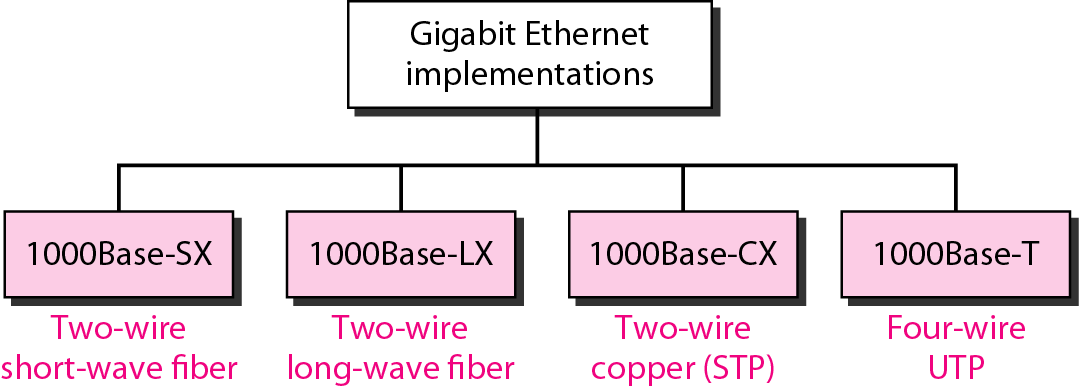
* **Topology:** Gigabit Ethernet is designed to connect two or more stations. If there are only two stations, they can be connected point-to-point. Three or more stations need to be connected in a star topology with a hub or a switch at the center. Another possible configuration is to connect several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



* **Encoding:** Gigabit Ethernet cannot use the Manchester encoding scheme because it involves a very high bandwidth. The two-wire implementations use an NRZ scheme, but NRZ does not self-synchronize properly. To synchronize bits at this high data rate 8B/10B block coding is used.

**Implementation**

Gigabit Ethernet can be categorized as either a two-wire or a four-wire implementation. The two-wire implementations use fiber-optic cable (1000Base-SX, short-wave, or 1000Base-LX, long-wave), or STP (1000Base-CX). The four-wire version uses category 5 twisted-pair cable (1000Base-T). In other words, we have four implementations, as shown in Figure 13.23. 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.



**TEN-GIGABIT ETHERNET**

The IEEE committee created Ten-Gigabit Ethernet and called it Standard 802.3ae. The goals of the Ten-Gigabit Ethernet design can be summarized as follows:

1. Upgrade the data rate to 10 Gbps.
2. Make it compatible with Standard, Fast, and Gigabit Ethernet.
3. Use the same 48-bit address.
4. Use the same frame format.Keep the same minimum and maximum frame lengths.
5. Allow the interconnection of existing LANs into a metropolitan area network (MAN) or a wide area network (WAN).
6. Make Ethernet compatible with technologies such as Frame Relay and ATM.

**Questions from recent VTU Papers**

1. Describe the different controlled access methods. (Dec 2010)
2. Explain 802.3 MAC frame format and frame length.

(Dec 2008 / June 2009 / Dec 2010 / 2012 / June 2013)

1. Explain i) CSMA ii) CSMA/CD ii) Slotted ALOH (Dec 2008 / June 2009 / 2010)
2. What do you mean by channelization? Explain the protocols used for channelization.

(June 2010)

1. A network transmits 200 bit frame on a shared channel of 200 kbps. For aloha and slotted aloha, what is the, i) Requirement to make the frame collision free? ii) Throughput if the system produces 1000 frames/sec? (Dec 2009 / June 2012)
2. With a neat diagram explain CSMA/CD protocol. (June2012)
3. Define channelization and list and explain its three protocols? (June 2009 / Dec 2009)
4. Explain why collision is an issue in a random access protocol but not in controlled access or channelizing protocols? (Dec 2012)
5. Explain any two popular control access methods, with a neat diagram. (Dec 2012)
6. A SLOTTED ALOHA network transmits 200 bit frame on a shared channel of 200 kbps. Find the throughput if the system produces i) 1000 Frames / sec ii) 500 Frames / sec and iii) 250 Frames / sec. (June 2013)
7. What is channelization? Explain CDMA. (Dec 2012 / June 2013)