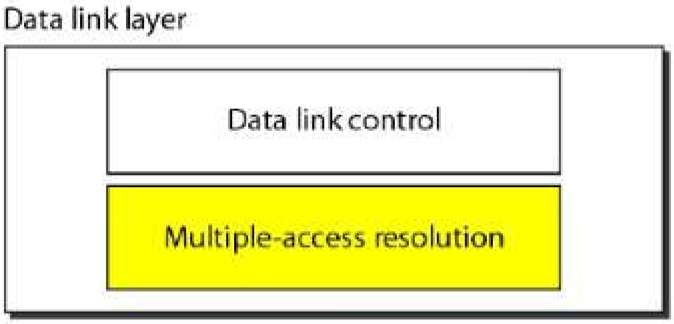
UNIT-4
Multiple Access

Syllabus: Multiple Access
Random Access: ALOHA, Carrier sense multiple access (CSMA), CSMA with Collision Detection, CSMA with Collision Avoidance, Controlled Access: Reservation, Polling, Token Passing, Channelization: frequency division multiple access(FDMA), time division multiple access(TDMA), code division multiple access(CDMA).

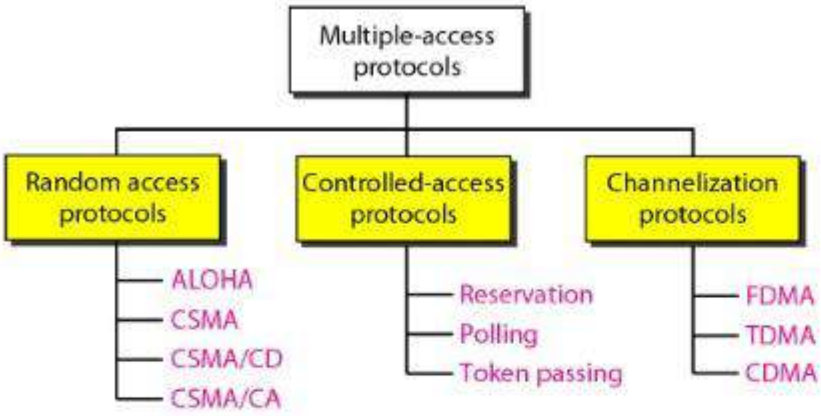
Multiple Access:

- Data link layer in the OSI model is divided into two layers,



- The upper sublayer is responsible for flow and error control is called as LLC (Logical Link Control) layer.
- The lower sublayer is responsible for Multiple access resolution is called Media Access Control(MAC) layer.

Multiple Access Protocols:



Random Access Protocols:

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:

❑ When can the station access the medium?
❑ What can the station do if the medium is busy?
❑ How can the station determine the success or failure of the transmission?
❑ What can the station do if there is an access conflict?

- We have different Random-Access methods listed as
  - ALOHA
  - Carrier Sense Multiple Access (CSMA)
  - CSMA CD (Collision Detection)
  - CSMA CA (Collision Avoidance)

ALOHA:

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

- **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.***
  - Since there is only one channel to share, there is the possibility of collision between frames from different stations.
  - A collision involves two or more stations.

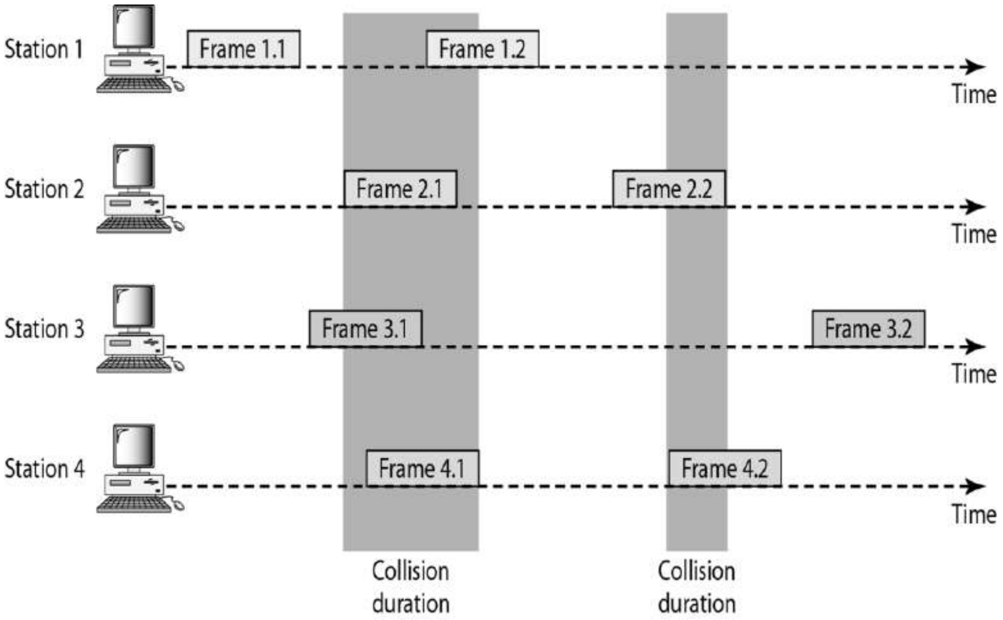


Figure: Pure ALOHA

  - 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.
    - If all these stations try to resend their frames after the time-out, the frames will collide again.

- **Collision Prevention in Pure ALOHA:**
  - Pure ALOHA dictates that when the time-out period passes, each station waits a random amount of time before resending its frame.
  - After a maximum number of retransmission attempts Kmax' a station must give up and try later.
  - **Vulnerable Time:** The length of the collision is given by the Vulnerable Time.

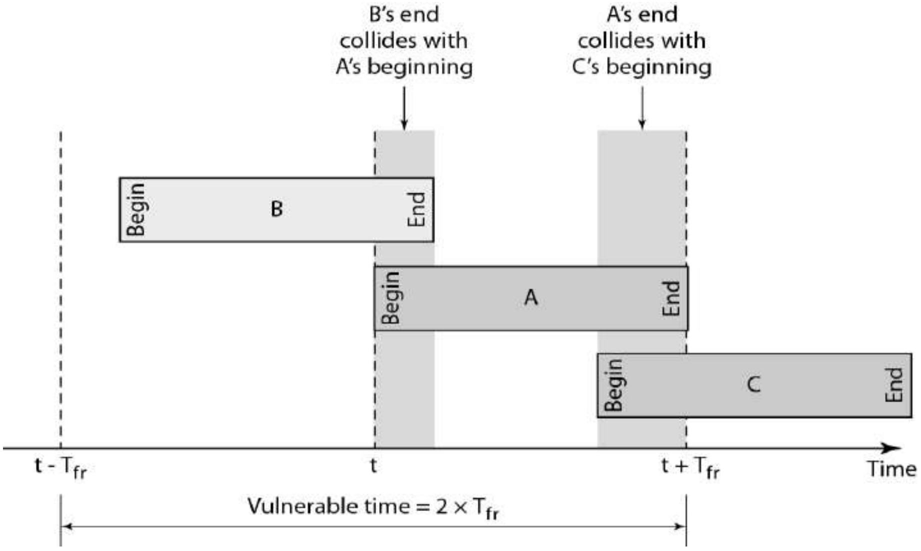

B's end collides with A's beginning
A's end collides with C's beginning
Begin
B
End
Begin
A
End
Begin
C
End
$t - T_{fr}$
t
$t + T_{fr}$
Time
Vulnerable time = $2 \times T_{fr}$

Figure: Vulnerable Time of Pure ALOHA

  - Let us assume that the stations send fixed-length frames with each frame taking T_{fr} s to send.
  - The Above figure gives the vulnerable time of the Station A, Station A sends a frame at time *t.* Now imagine station B has already sent a frame between *t* - T_{fr} and *t.* This leads to a collision between the frames from station A and station B. The end of B's frame collides with the beginning of A's frame.
  - On the other hand, suppose that station C sends a frame between *t* and *t*+T_{fr}. Here, there is a collision between frames from station A and station C. The beginning of C's frame collides with the end of A's frame.
  - 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:**
  - Slotted ALOHA was invented to improve the efficiency of pure ALOHA.
  - In slotted ALOHA we divide the time into slots of T_{fr} 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.

- There is still the possibility of collision if two stations try to send at the beginning of the same time slot.

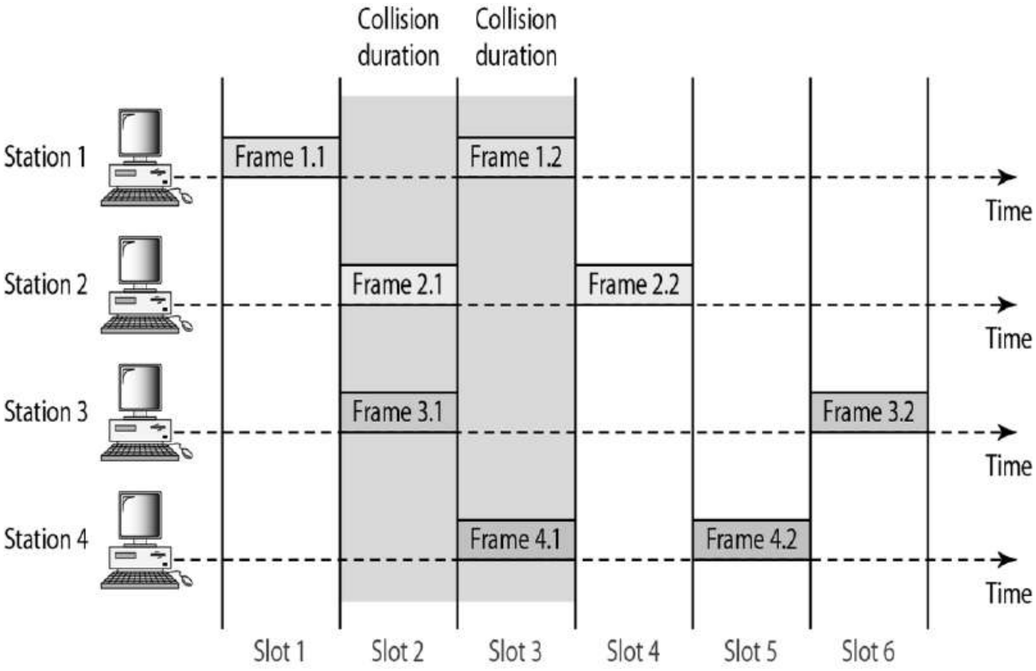


Figure: Slotted ALOHA

- **Vulnerable Time:** The vulnerable time for slotted ALOHA is one-half that of pure ALOHA.

Slotted ALOHA vulnerable time = T_{fr}

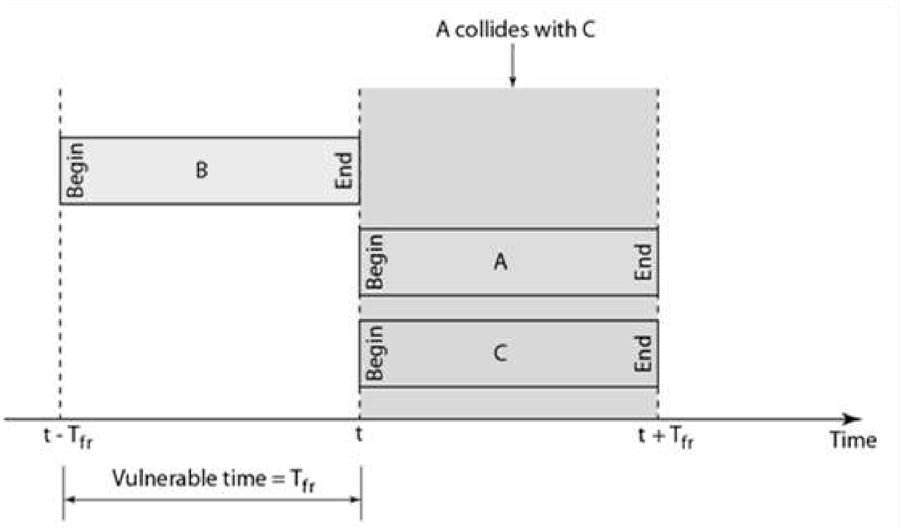


Figure: Vulnerable Time of Slotted ALOHA

- **Throughput:**
  - 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 a frame is generated during one frame transmission time, then 36.8 percent of these frames reach their destination successfully. This result can be expected because the vulnerable time is equal to the frame transmission time.
  - 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.

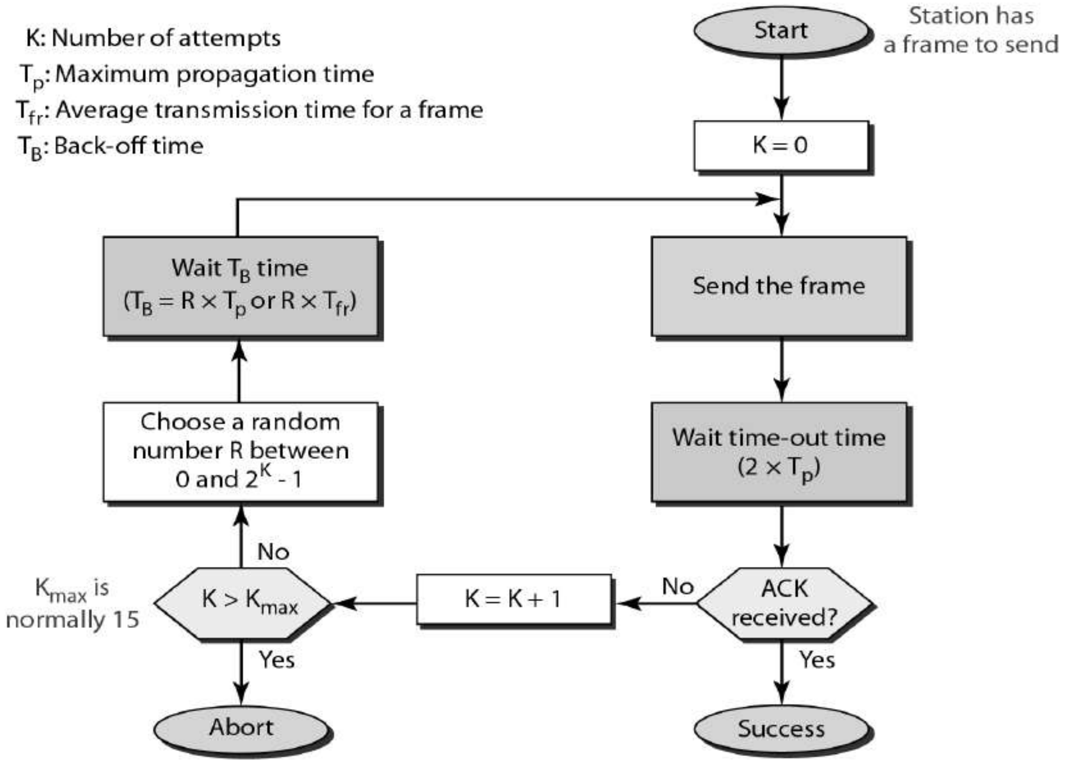

K: Number of attempts
T_p: Maximum propagation time
T_{fr}: Average transmission time for a frame
T_B: Back-off time
Start
Station has a frame to send
K = 0
Send the frame
Wait time-out time ($2 \times T_p$)
ACK received?
No
Yes
Success
K = K + 1
$K > K_{max}$
K_{max} is normally 15
Yes
Abort
No
Choose a random number R between 0 and $2^K - 1$
Wait T_B time ($T_B = R \times T_p$ or $R \times T_{fr}$)

Figure: Flow Diagram for ALOHA

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.
- 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.

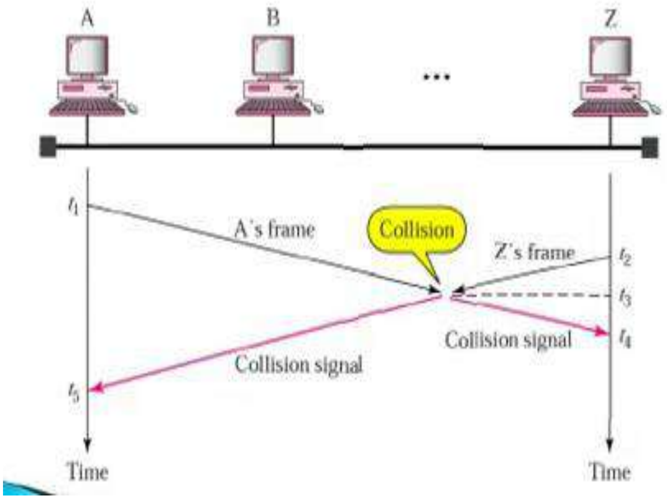


Figure: CSMA Collision(The two signals collide and both frames are destroyed.)

- **Vulnerable Time of CSMA:**
  - The vulnerable time for CSMA is the **propagation time T_p** . 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.

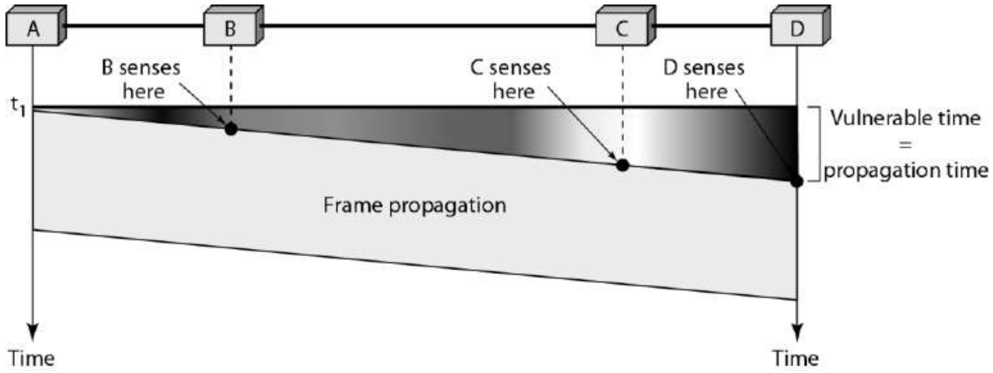


Figure: Vulnerable Time of CSMA

- **Persistence Methods of CSMA:**
  - There are three methods for CSMA persistence
    1. **1-persistent CSMA**
    2. **Non Persistent CSMA**
    3. **P-persistent CSMA**
  - **1-Persistennt CSMA:**
    - The **I-persistent method** is simple and straightforward.
    - **In** this method, after the station finds the line idle, it sends its frame immediately (with probability 1).

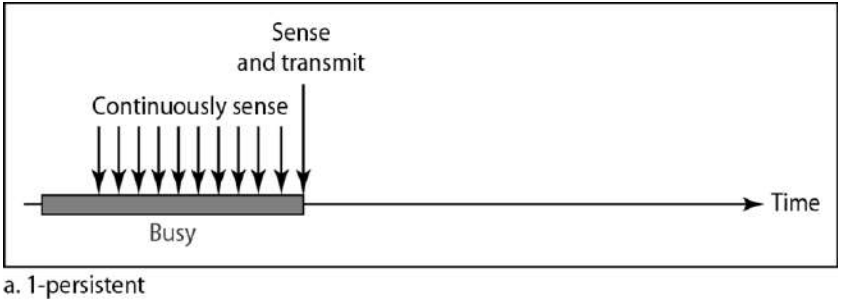


Figure: 1-Persistent CSMA

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

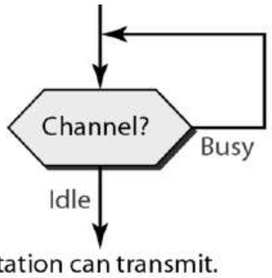
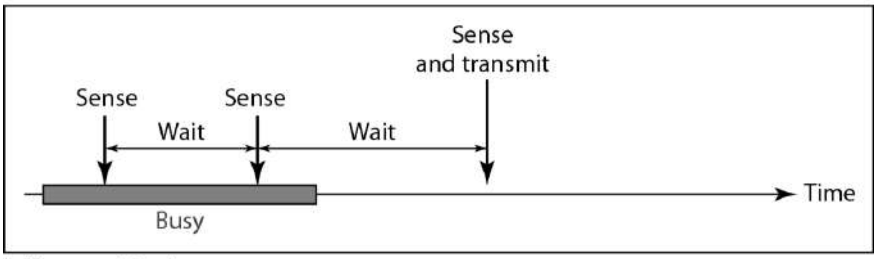
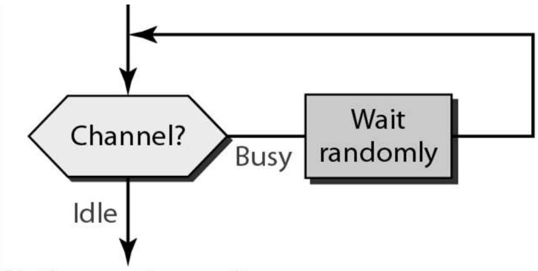


Figure: Flow Diagram of 1-Persistent CSMA

- **Non Persistent CSMA:**
  - 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.
  - This method reduces the efficiency of the network because the medium remains idle when there may be stations with frames to send.



b. Nonpersistent

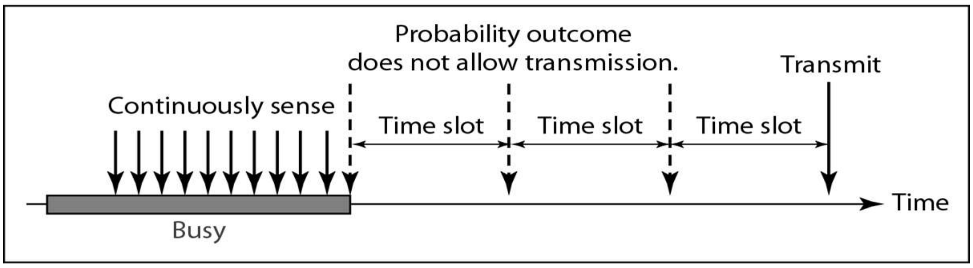


Station can transmit.

Figure: Flow diagram for Non Persistent CSMA

- **P-Persistent CSMA:**
  - 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.



c. p-persistent

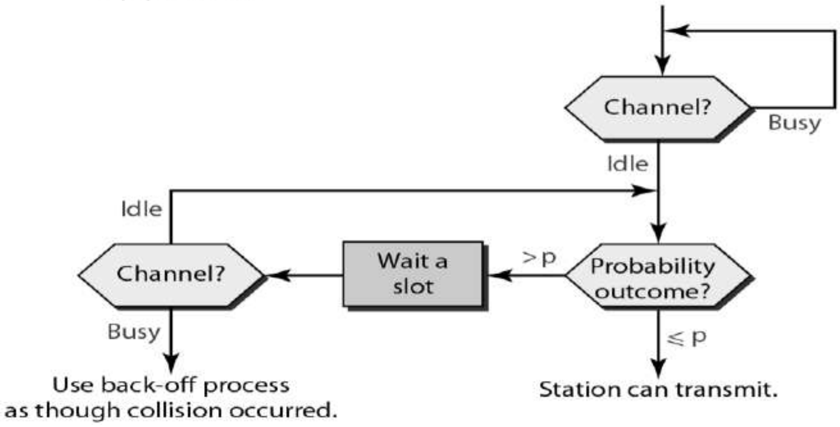


Figure: Flow Diagram for P-Persistent CSMA

CSMA/CD (Carrier Sense Multiple Access with Collision Detection):

- 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.

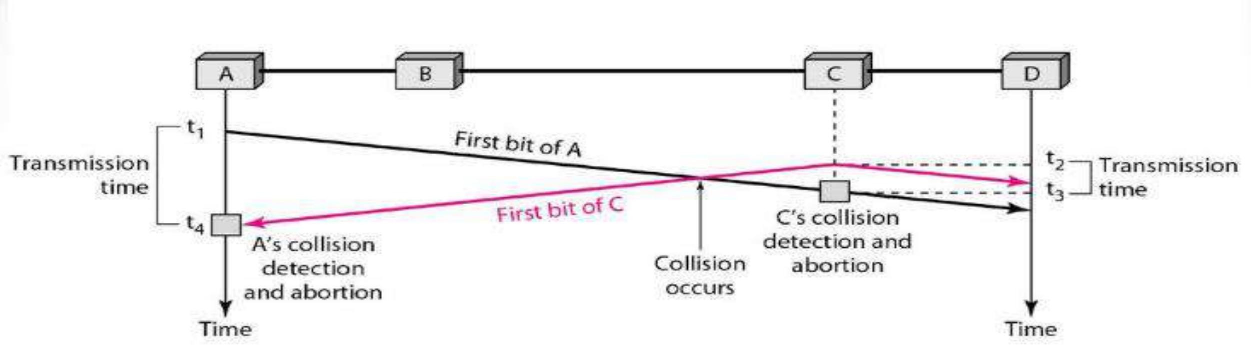


Figure: Collision of the first bit in CSMA/CD

- At time t_1, station A has executed its persistence procedure and starts sending the bits of its frame.
- At time t_2, 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 t_2 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.

- **Minimum Frame Size:** Each frame must be large enough for a sender to detect a collision.
- **Energy Level:**
    - The level of energy in a channel can have three values: zero, normal, and abnormal.
    - At the zero level, the channel is idle. This level is also called ***Idle Period.***
    - At the normal level, a station has successfully captured the channel and is sending its frame. This level is called ***Transmission Period.***
    - At the abnormal level, there is a collision and the level of the energy is twice the normal level. This is called ***Contention Period.***
    - A station that has a frame to send or is sending a frame needs to monitor the energy level to determine if the channel is idle, busy, or in collision mode.

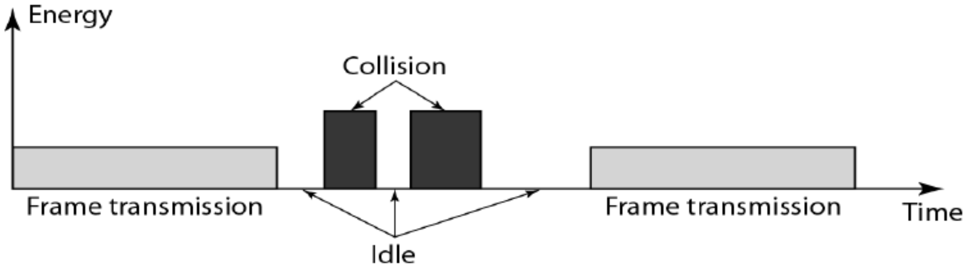


Figure: Energy levels in CSMA/CD

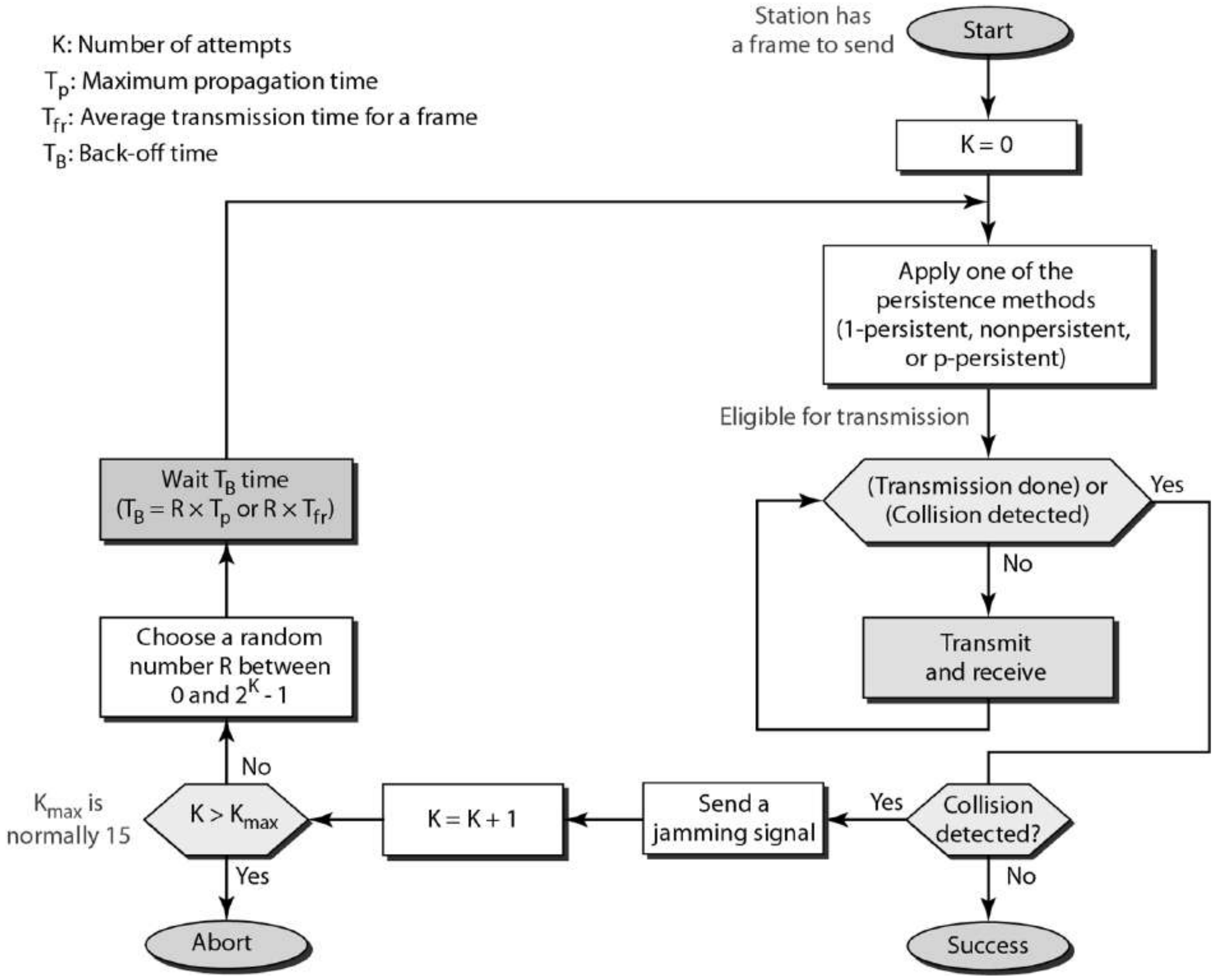

K: Number of attempts
T_p: Maximum propagation time
T_{fr}: Average transmission time for a frame
T_B: Back-off time
Station has a frame to send
Start
K = 0
Apply one of the persistence methods (1-persistent, nonpersistent, or p-persistent)
Eligible for transmission
(Transmission done) or (Collision detected)
Yes
No
Transmit and receive
Wait T_B time ($T_B = R \times T_p$ or $R \times T_{fr}$)
Choose a random number R between 0 and $2^K - 1$
No
K_{max} is normally 15
$K > K_{max}$
Yes
Abort
K = K + 1
Send a jamming signal
Yes
Collision detected?
No
Success

Figure: Flow Diagram for CSMA/CD

CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance):

- The basic idea behind *CSMA/CD* is that a station needs to be able to receive while transmitting to detect a collision. The signal from the second station needs to add a significant amount of energy to the one created by the first station.
- In a wired network, the received signal has almost the same energy as the sent signal because either the length of the cable is short or there are repeaters that amplify the energy between the sender and the receiver. This means that in a collision, the detected energy almost doubles.
- In a wireless network, much of the sent energy is lost in transmission. The received signal has very little energy. Therefore, a collision may add only 5 to 10 percent additional energy. This is not useful for effective collision detection.
- Carrier sense multiple access with collision avoidance *(CSMA/CA)* was invented for this network.
- Collisions are avoided through the use of CSMA/CA's three strategies:
  The Interframe Space, The Contention Window, and Acknowledgments
  - **Interframe Space:**
    - 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.
    - The IFS time allows the front of the transmitted signal by the distant station to reach this station.
    - If after the IFS time the channel is still idle, the station can send, but it still needs to wait a time equal to the contention time.
    - The IFS variable can also be used to prioritize stations or frame types. For example, a station that is assigned a shorter IFS has a higher priority.
  - **The 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.
    - The contention window is that the station needs to sense the channel after each time slot.
    - 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.
  - **Acknowledgements:**
    - 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.

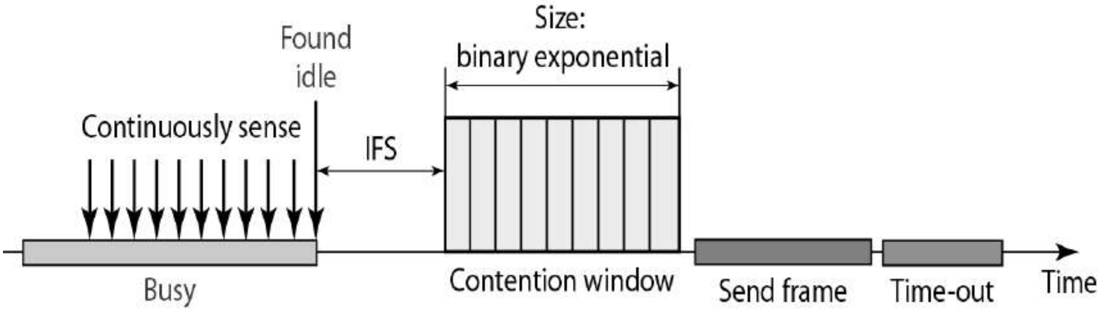


Figure: The Interframe Space, The Contention Window, and Acknowledgments

- ***CSMA/CA*** was mostly intended for use in **wireless networks.**

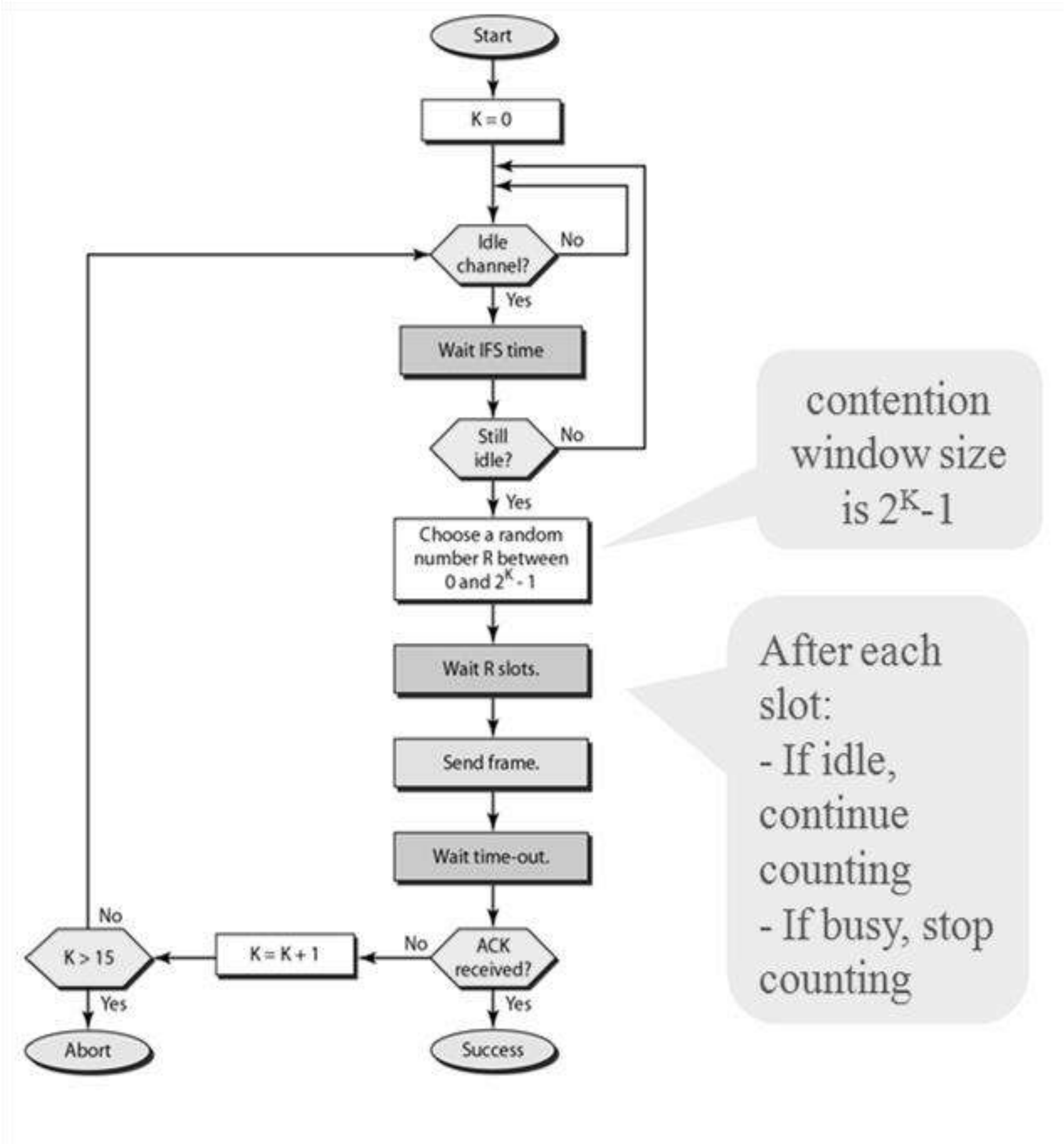


Figure: Flow Diagram of CSMA/CA

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 have three popular controlled-access methods.

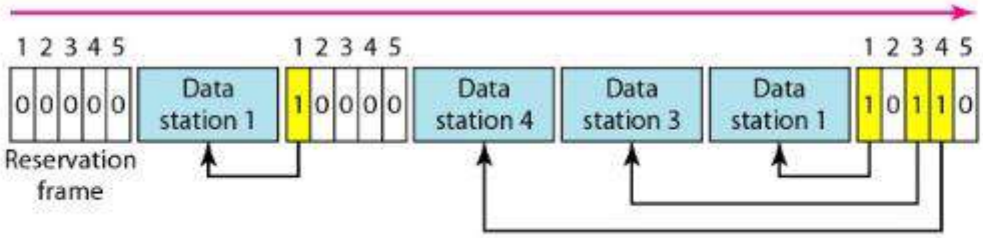
1. Reservation 2. Polling 3. Tokenpassing

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 minislots in the reservation frame. Each minislot belongs to a station. When a station needs to send a data frame, it makes a reservation in its own minislot. 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-minislot 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.



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

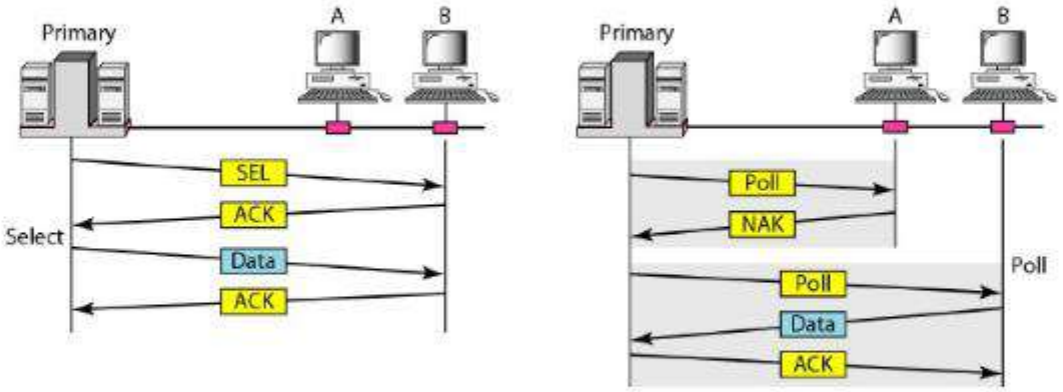
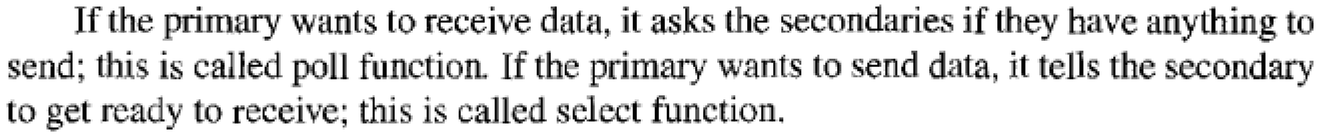
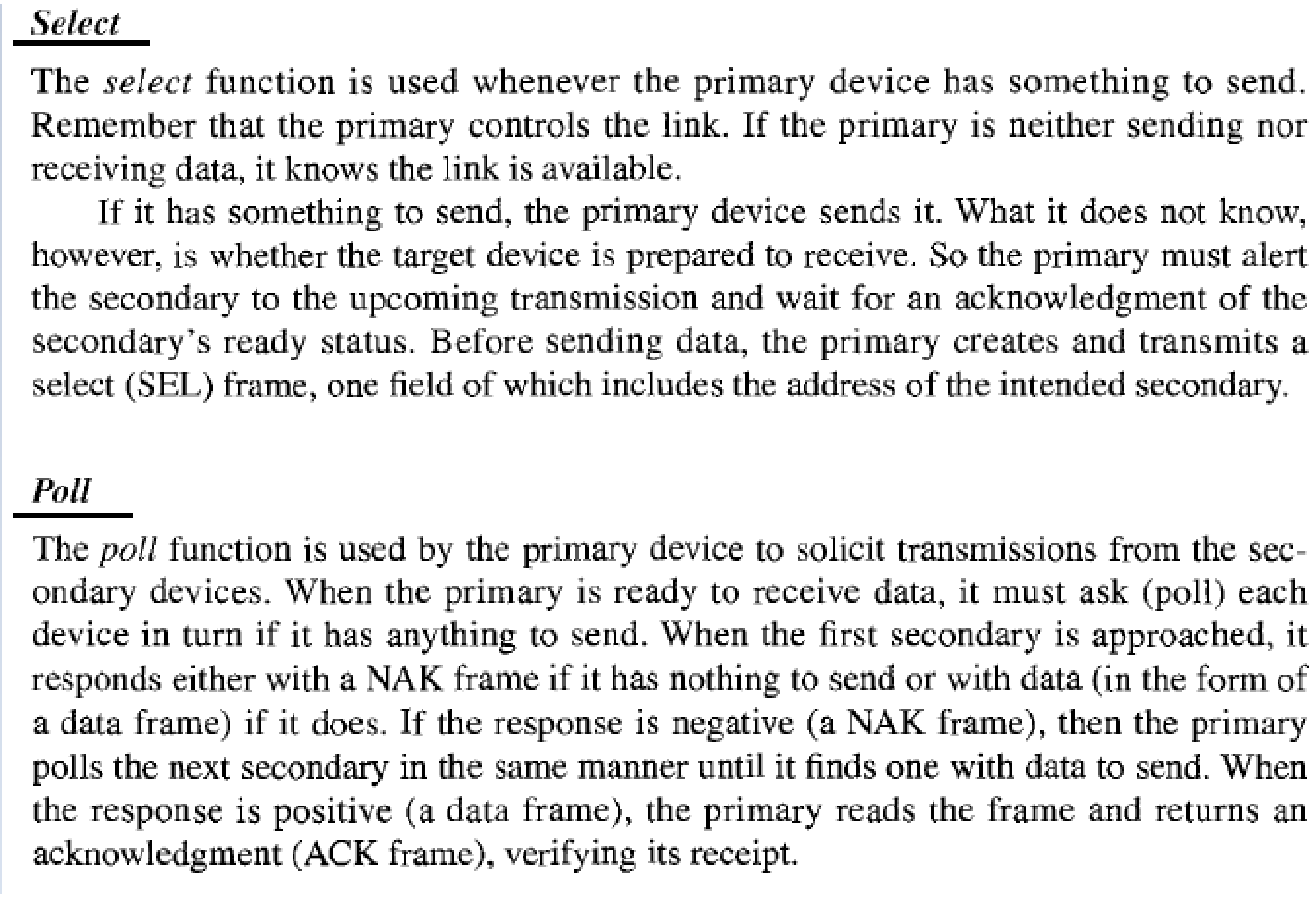


Figure: Select and poll functions in polling access method



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



Select

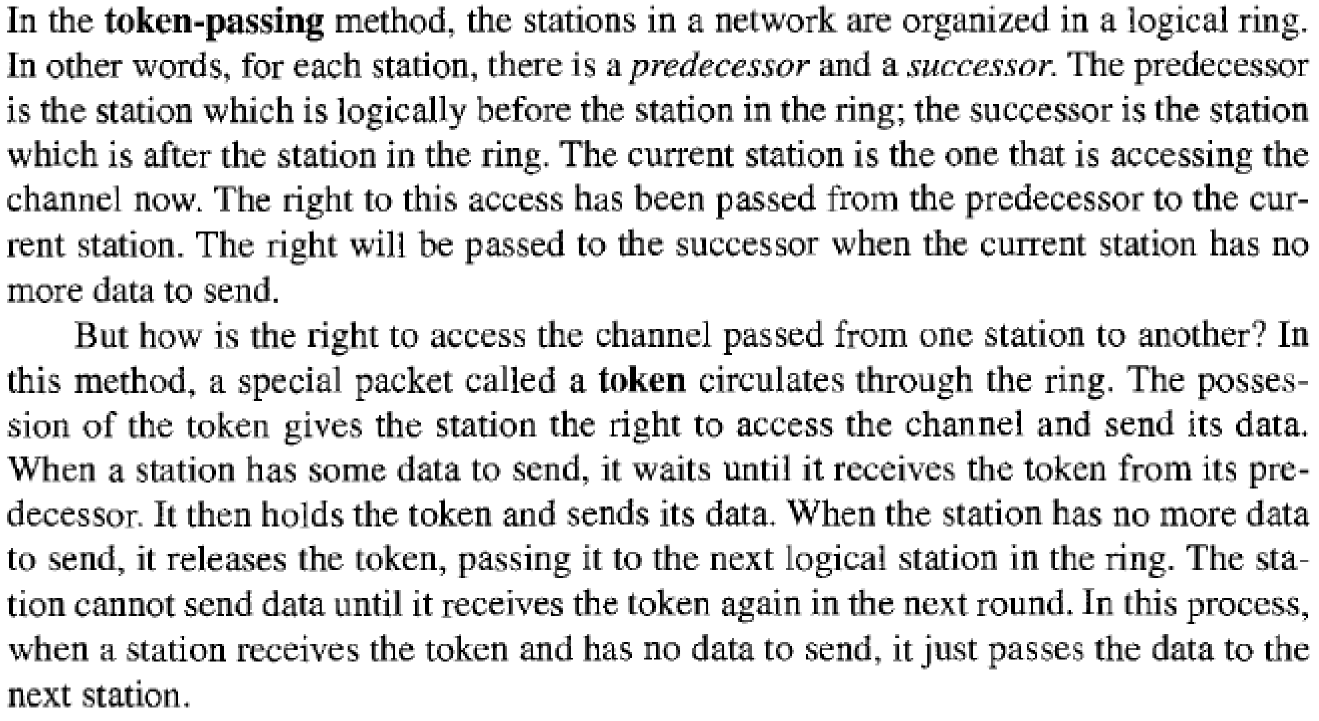
The *select* function is used whenever the primary device has something to send. Remember that the primary controls the link. If the primary is neither sending nor receiving data, it knows the link is available.

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.

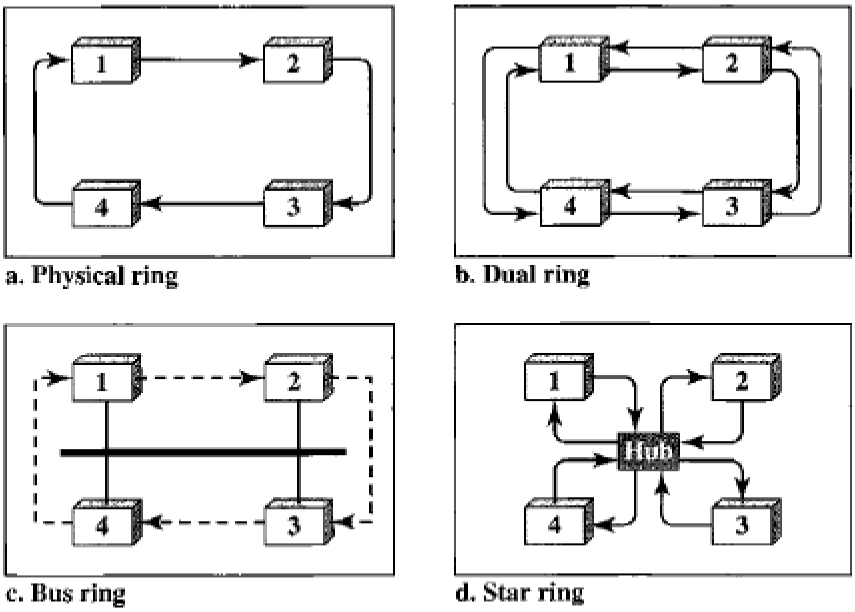
Token management is needed for this access method. Stations must be limited in the time they can have possession of the token. The token must be monitored to ensure it has not been lost or destroyed. For example, if a station that is holding the token fails, the token will disappear from the network. 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. Figure 12.20 show four different physical topologies that can create a logical ring.

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



a. Physical ring

b. Dual ring

c. Bus ring

d. Star ring

Channelization:

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

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

Frequency Division Multiple Access (FDMA):

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

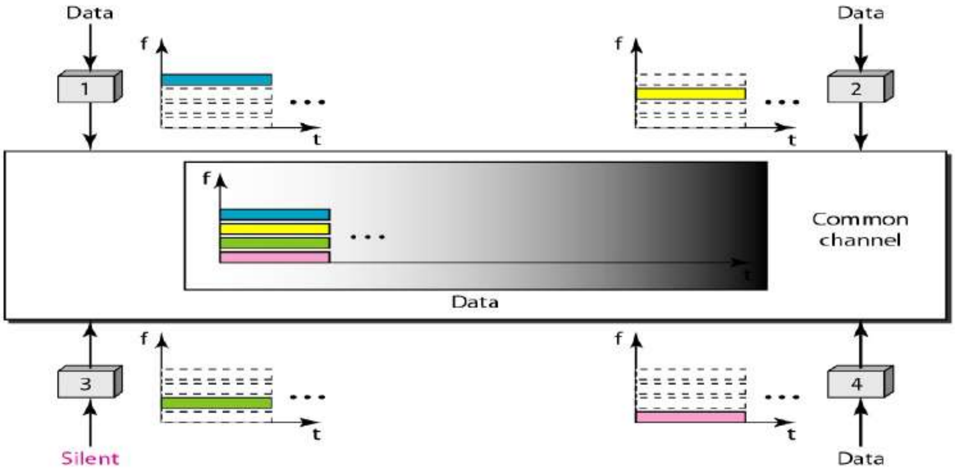


Figure: FDMA

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.

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.

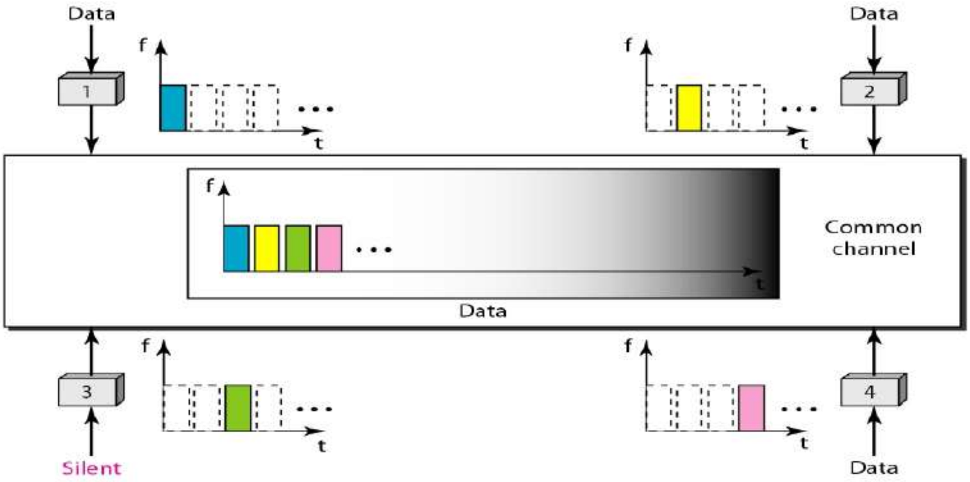
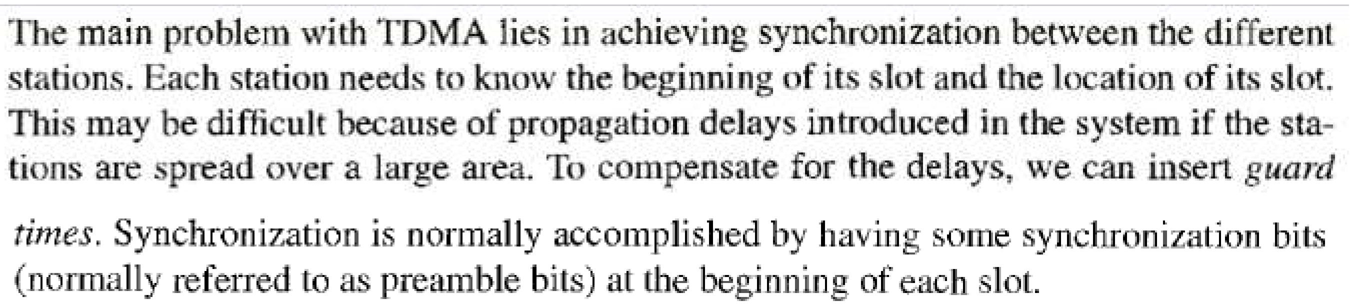
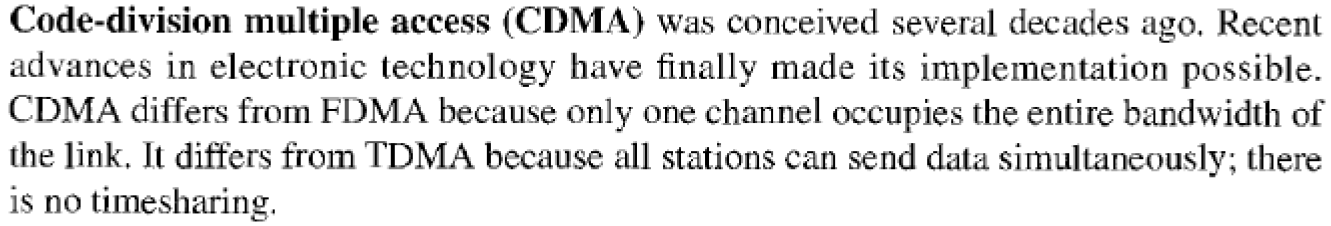


Figure: TDMA



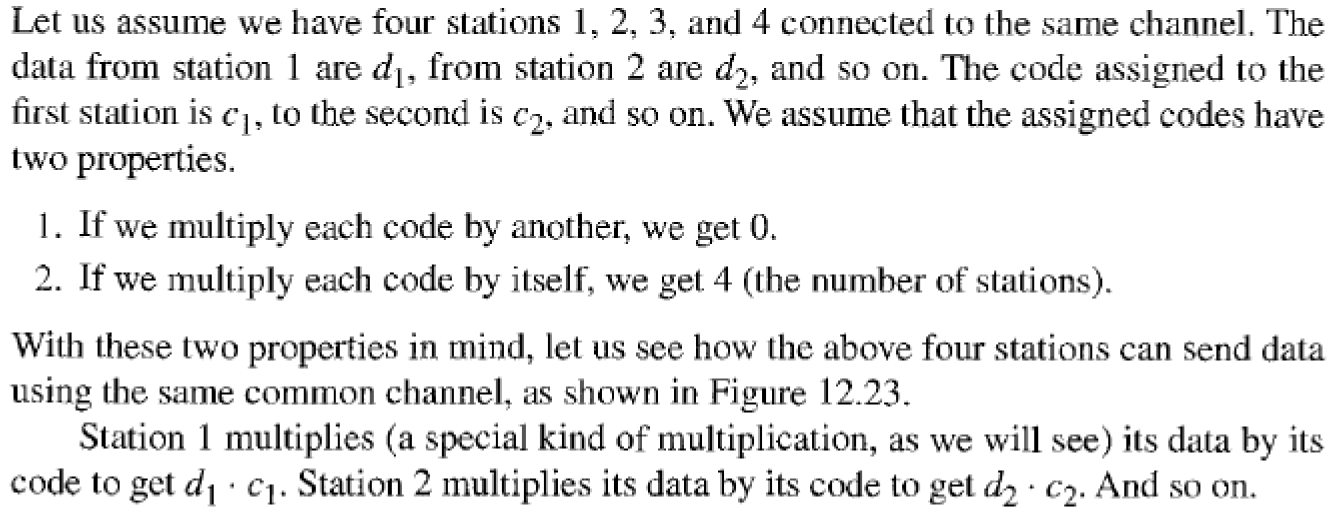
The main problem with TDMA lies in achieving synchronization between the different stations. Each station needs to know the beginning of its slot and the location of its slot. This may be difficult because of propagation delays introduced in the system if the stations are spread over a large area. To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

Code Division Multiple Access (CDMA):



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

CDMA means communication with different codes.

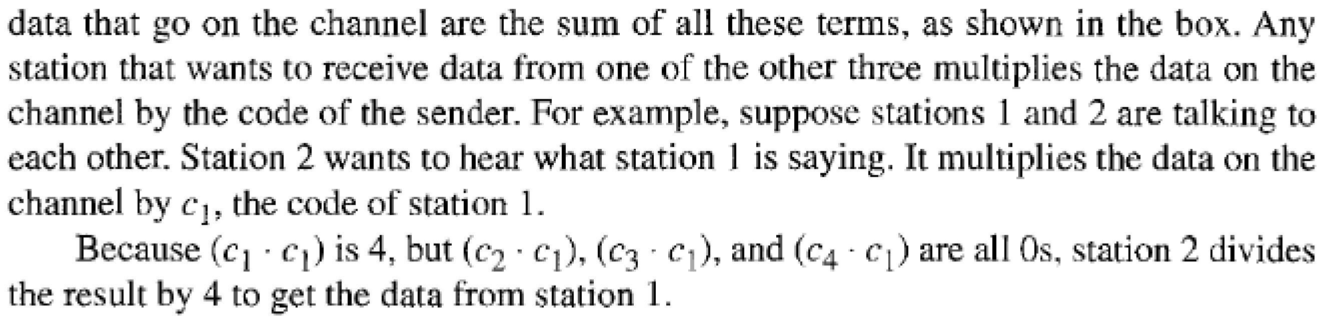


Let us assume we have four stations 1, 2, 3, and 4 connected to the same channel. The data from station 1 are d_1, from station 2 are d_2, and so on. The code assigned to the first station is c_1, to the second is c_2, and so on. We assume that the assigned codes have two properties.

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).

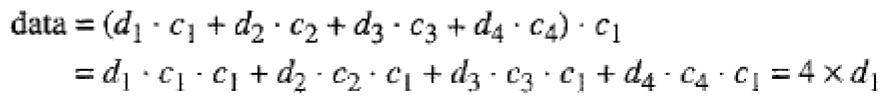
With these two properties in mind, let us see how the above four stations can send data using the same common channel, as shown in Figure 12.23.

Station 1 multiplies (a special kind of multiplication, as we will see) its data by its code to get $d_1 \cdot c_1$. Station 2 multiplies its data by its code to get $d_2 \cdot c_2$. And so on.



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_1, the code of station 1.

Because $(c_1 \cdot c_1)$ is 4, but $(c_2 \cdot c_1)$, $(c_3 \cdot c_1)$, and $(c_4 \cdot c_1)$ are all 0s, station 2 divides the result by 4 to get the data from station 1.



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

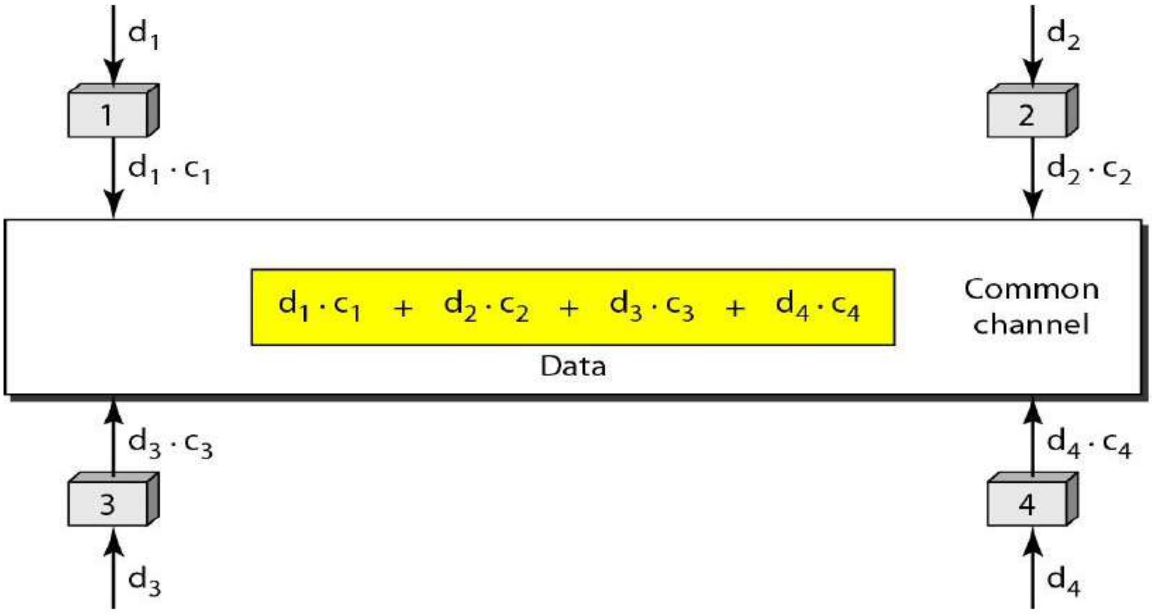


Figure: CDMA

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 (for 4 stations).

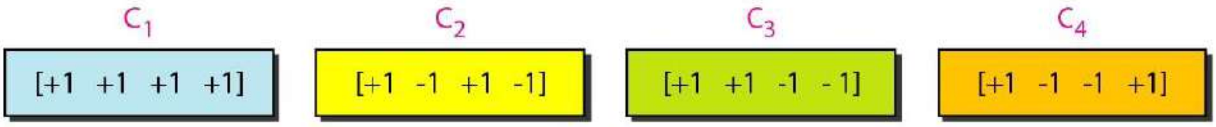


Figure: Chip Sequences

The above sequences are orthogonal sequences, the orthogonal sequences 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,

$$2 \cdot [+1\ +1\ -1\ -1] = [+2\ +2\ -2\ -2]$$

3. If we multiply two equal sequences, element by element, and add the results, we get *N*, where *N* is the number of elements in the each sequence. This is called the **inner product** of two equal sequences. For example,

$$[+1\ +1\ -1\ -1] \cdot [+1\ +1\ -1\ -1] = 1 + 1 + 1 + 1 = 4$$

4. If we multiply two different sequences, element by element, and add the results, we get 0. This is called inner product of two different sequences. For example,

$$[+1\ +1\ -1\ -1] \cdot [+1\ +1\ +1\ +1] = 1 + 1 - 1 - 1 = 0$$

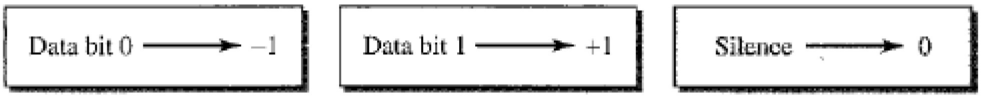
5. Adding two sequences means adding the corresponding elements. The result is another sequence. For example,

$$[+1\ +1\ -1\ -1] + [+1\ +1\ +1\ +1] = [+2\ +2\ \ 0\ \ 0]$$

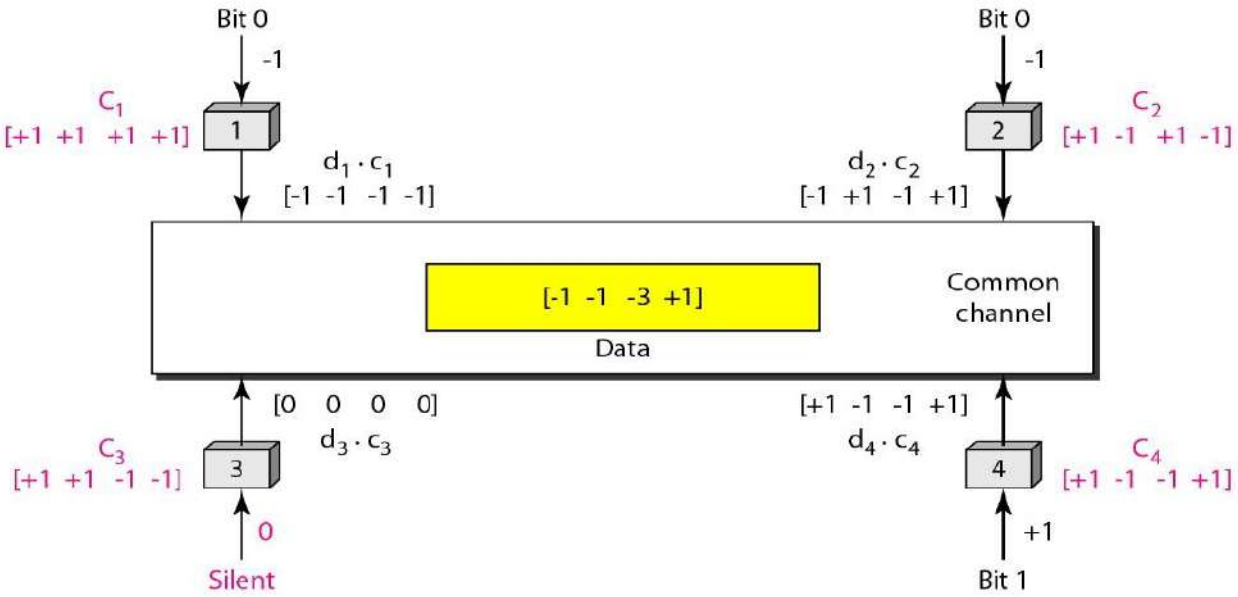
Data□Representation

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

Figure *Data representation in CDMA*



Example: Sharing channel in CDMA



****