MODULE – 2
The Data link layer

DATA LINK LAYER DESIGN ISSUES

- The data link layer uses the services of the physical layer to send and receivebits over communication channels. Functions of data link layer include:

• Providing a well-defined service interface to the network layer.

• Dealing with transmission errors.

• Regulating the flow of data so that slow receivers are not swampedby fast senders.

- To accomplish these goals, the data link layer takes the packets it gets from the network layer and encapsulates them into **frames** for transmission. Each frame contains a frame header, a payload field for holding the packet, and a frame trailer, as illustrated in Fig. 3-1. Frame management forms the heart of what the data link layer does.

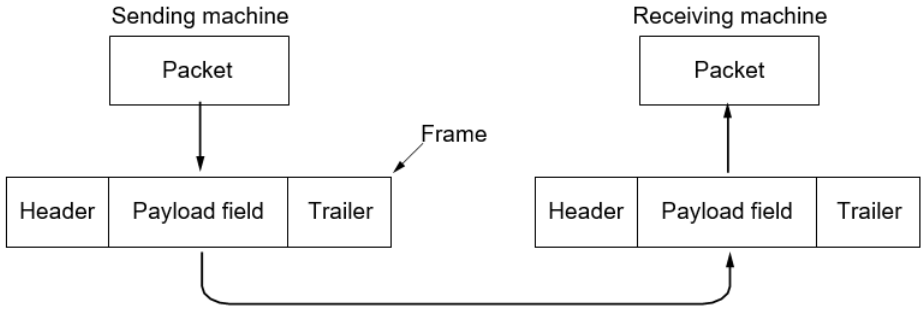


Figure 3-1. Relationship between packets and frames.

The following are the data link layer design issues

1. Services Provided to the Network Layer

The network layer wants to be able to send packets to its neighbors without worrying about the details of getting it there in one piece.

2. Framing

Group the physical layer bit stream into units called frames. Frames are nothing more than "packets" or "messages". By convention, we use the term "frames" when discussing DLL.

3. Error Control

Sender checksums the frame and transmits checksum together with data. Receiver re-computes the checksum and compares it with the received value.

4. Flow Control

Prevent a fast sender from overwhelming a slower receiver

Services Provided to the Network Layer

- The function of the data link layer is to provide services to the network layer. The principal service is transferring data from the network layer on the source machine to the network layer on the destination machine.
- On the source machine is an entity(a process), in the network layer that hands some bits to the data link layer for transmission to the destination.
- The job of the data link layer is to transmit the bits to the destination machine so they can be handed over to the network layer there, as shown in Fig. 3-2(a). The actual transmission follows the path of Fig. 3-2(b)

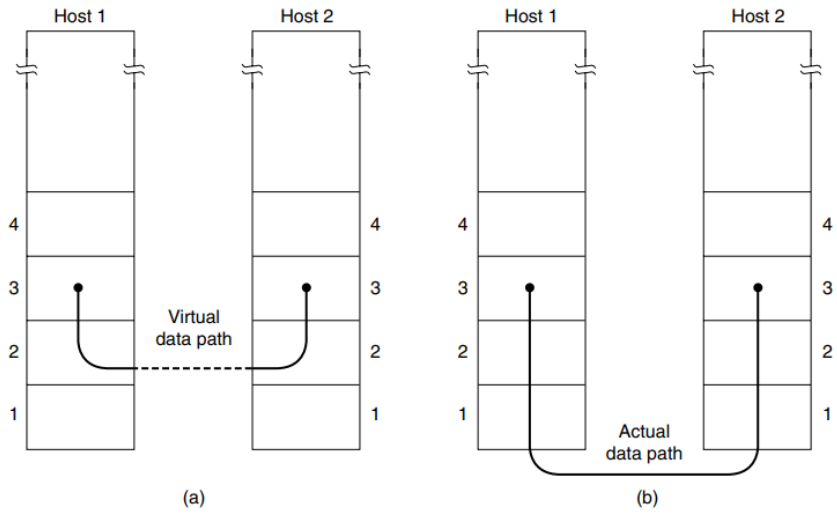


Figure 3-2. (a) Virtual communication. (b) Actual communication.

- The data link layer can be designed to offer various services:

1. Unacknowledged connectionless service.
2. Acknowledged connectionless service.
3. Acknowledged connection-oriented service.

- Unacknowledged connectionless service consists of having the source ma- chine send independent frames to the destination machine without having the destination machine acknowledge them. Ethernet is a good example of a data link layer that provides this class of service. No logical connection is established beforehand or released afterward. If a frame is lost due to noise on the line, no attempt is made to detect the loss or recover from it in the data link layer. This class of service is appropriate when the error rate is very low, so recovery is left to higher layers. It is also appropriate for real-time traffic, such as voice, in which late data are worse than bad data.
- The next step up in terms of reliability is acknowledged connectionless service. When this service is offered, there are still no logical connections used, but each frame sent is individually acknowledged. In this way, the sender knows whether a frame has arrived correctly or been lost. If it has not arrived within a specified time interval, it can be sent again. This service is useful over unreliable channels, such as wireless systems. Eg: 802.11 (WiFi)
- The most sophisticated service the data link layer can provide to the network layer is connection-oriented service. With this service, the source and destination machines establish a connection before any data are transferred. Each frame sent over the connection is numbered, and the data link layer guarantees that each frame sent is indeed

received. Furthermore, it guarantees that each frame is received exactly once and that all frames are received in the right order.

- When connection-oriented service is used, transfers go through three distinct phases.
  - First, connection is established by having both sides initialize variables and counters needed to keep track of which frames have been received and which ones have not.
  - Second, one or more frames are actually transmitted.
  - Third, connection is released, freeing up the variables, buffers, and other resources used to maintain the connection.

Framing

Data link layer breaks up the bit stream into discrete frames, compute a short token called a checksum for each frame, and include the checksum in the frame when it is transmitted. (Checksum algorithms will be discussed later in this chapter.) When a frame arrives at the destination,the checksum is recomputed. If the newly computed checksum is different from the one contained in the frame, the data link layer knows that an error has occurred and takes steps to deal with it.

Framing methods:

1. Byte count.
2. Flag bytes with byte stuffing.
3. Flag bits with bit stuffing.
4. Physical layer coding violations.

The first framing method uses a field in the header to specify the number of bytes in the frame. When the data link layer at the destination sees the byte count, it knows how many bytes follow and hence where the end of the frame is. This technique is shown in Fig. 3-3(a) for four small example frames of sizes 5, 5, 8, and 8 bytes, respectively. The trouble with this algorithm is that the count can be garbled by a transmission error. For example, if the byte count of 5 in the second frame of Fig. 3-3(b) becomes a 7 due to a single bit flip, the destination will get out of synchroniza- tion. It will then be unable to locate the correct start of the next frame.

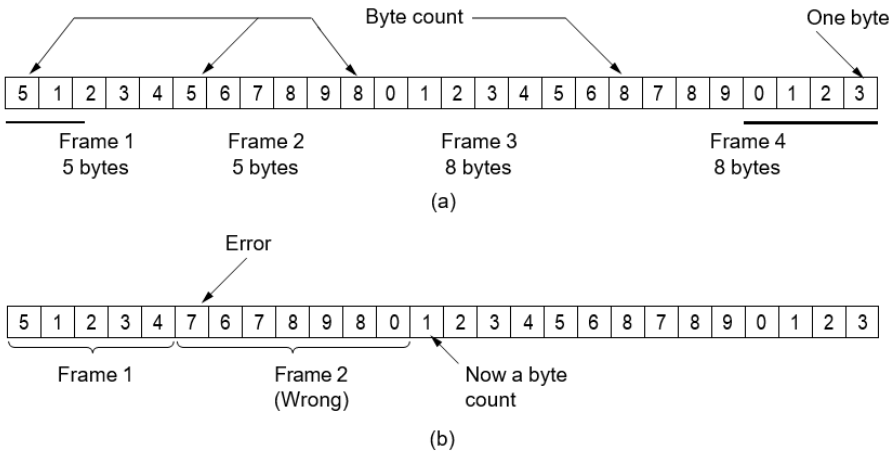


Figure 3-3. A byte stream. (a) Without errors. (b) With one error.

The second framing method gets around the problem of resynchronization after an error by having each frame start and end with special bytes. Often the same byte, called a **flag byte**, is used as both the starting and ending

delimiter. This byte is shown in Fig. 3-4(a) as FLAG. Two consecutive flag bytes indicate the end of one frame and the start of the next. Thus, if the receiver ever loses syn- chronization it can just search for two flag bytes to find the end of the current frame and the start of the next frame.

However, there is a still a problem we have to solve. It may happen that the flag byte occurs in the data, especially when binary data such as photographs or songs are being transmitted. This situation would interfere with the framing. One way to solve this problem is to have the sender's data link layer insert a special escape byte (ESC) just before each ''accidental'' flag byte in the data. Thus, a framing flag byte can be distinguished from one in the data by the absence or presence of an escape byte before it. The data link layer on the receiving end re- moves the escape bytes before giving the data to the network layer. This techni- que is called **byte stuffing**.

If an escape byte occurs in the middle of the data, it is also stuffed with an escape byte. At the receiver, the first escape byte is removed, leaving the data byte that follows it (which might be another escape byte or the flag byte). Some examples are shown in Fig. 3-4(b).

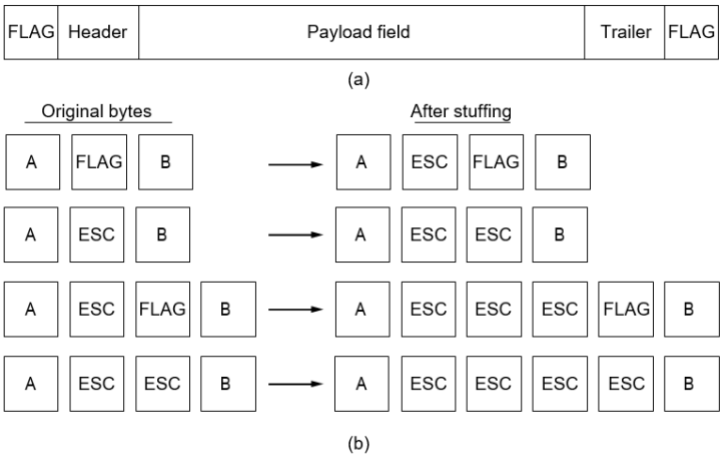


Figure 3-4. (a) A frame delimited by flag bytes. (b) Four examples of byte se- quences before and after byte stuffing.

The third method of delimiting the bit stream gets around a disadvantage of byte stuffing, which is that it is tied to the use of 8-bit bytes. Framing can be also be done at the bit level, so frames can contain an arbitrary number of bits made up of units of any size. It was developed for the once very popular **HDLC** (**High- level Data Link Control**) protocol. Each frame begins and ends with a special bit pattern, 01111110 or 0x7E in hexadecimal. This pattern is a flag byte. When- ever the sender's data link layer encounters five consecutive 1s in the data, it automatically stuffs a 0 bit into the outgoing bit stream. This **bit stuffing** is anal- ogous to byte stuffing, in which an escape byte is stuffed into the outgoing charac- ter stream before a flag byte in the data. It also ensures a minimum density of transitions that help the physical layer maintain synchronization. USB (Universal Serial Bus) uses bit stuffing for this reason.

When the receiver sees five consecutive incoming 1 bits, followed by a 0 bit,

it automatically destuffs (i.e., deletes) the 0 bit. Just as byte stuffing is completely transparent to the network layer in both computers, so is bit stuffing. If the user data contain the flag pattern, 01111110, this flag is transmitted as 011111010 but stored in the receiver's memory as 01111110. Figure 3-5 gives an example of bit stuffing.

With bit stuffing, the boundary between two frames can be unambiguously recognized by the flag pattern.

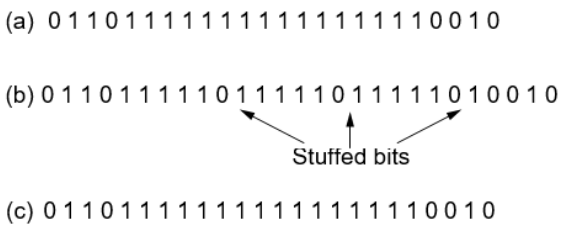


Figure 3-5. Bit stuffing. (a) The original data. (b) The data as they appear on the line. (c) The data as they are stored in the receiver's memory after destuffing.

The last method of framing is encoding of bits as signals often includes redundancy to help the receiver. This redundancy means that some signals will not occur in regular data. For example, in the 4B/5B line code 4 data bits are mapped to 5 signal bits to ensure sufficient bit transitions. This means that 16 out of the 32 signal possibilities are not used. We can use some reserved signals to indicate the start and end of frames. In effect, we are using ''coding violations'' to delimit frames. The beauty of this scheme is that, because they are reserved signals, it is easy to find the start and end of frames and there is no need to stuff the data.

Many data link protocols use a combination of these methods for safety. A common pattern used for Ethernet and 802.11 is to have a frame begin with a well-defined pattern called a **preamble**. This pattern might be quite long (72 bits is typical for 802.11) to allow the receiver to prepare for an incoming packet. The preamble is then followed by a length (i.e., count) field in the header that is used to locate the end of the frame.

Error Control

Next task of Data link layer is to make sure all frames are eventually delivered to the network layer at the destination and in the proper order.

The usual way to ensure reliable delivery is to provide the sender with some feedback about what is happening at the other end of the line. If the sender receives a positive acknowledgement about a frame, it knows the frame has arrived safely. On the other hand, a negative acknowledgement means that something has gone wrong and the frame must be transmitted again.

But certain frames can go missing due to the introduction of some noise in the signal. If the acknowledgements are lost, sender will not understand what to do. That is when the timers are useful. When the sender transmits a frame, it generally also starts a timer. The timer is set to expire after an interval long enough for the frame to reach the destination,be processed there, and have the acknowledgement propagate back to the sender. Normally, the frame will be correctly received and the acknowledgement will get back before the timer runs out, in which case the timer will be canceled.

During retransmissions receiver may get duplicate packets. To prevent this from happening, sequence numbers are assigned to outgoing frames, so that the receiver can distinguish retransmissions from originals.

The whole issue of managing the timers and sequence numbers so as to ensurethat each frame is ultimately passed to the network layer at the destination exactly once, no more and no less, is an important part of the duties of the data link layer (and higher layers)

Flow Control

Another important design issue that occurs in the data link layer (and higher layers as well) is what to do with a sender that systematically wants to transmit frames faster than the receiver can accept them. Sender should send the data

at the same speed as the receiver is capable of receiving. Otherwise receiver will loose couple of frames.

Two approaches are commonly used.

1. **Feedback-based flow control -** the receiver sends back information to the sender giving it permission to send more data, or at least telling the sender how the receiver is doing.
2. **Rate-based flow control -** the protocol has a built-in mechanism that limits the rate at which senders may transmit data, without using feedback from the receiver.

ERROR DETECTION AND CORRECTION

- Transmission errors are unavoidable. We have to learn to deal with them.
- Two basic strategies for dealing with errors. Both add redundant information to the data that is sent.

1 . Include enough redundant information to enable the receiver to deduce what the transmitted data must have been. I t uses **error-correcting codes.** The use of error-correctingcodes is often referred to as **FEC** (**Forward Error Correction**).

2. Include only enough redundancy to allow the receiver to deduce that an error has occurred (but not which error) and have it request a retransmission. It uses **error-detecting codes**.

On channels that are highly reliable, such as fiber, it is cheaper to use an error-detecting code and just retransmit the occasional block found to be faulty. However, on channels such as wireless links that make many errors, it is better to add redundancy to each block so that the receiver is able to figure out what the originally transmitted block was. FEC is used on noisy channels because retransmissions are just as likely to be in error as the first transmission.

Errors are caused by extreme values of thermal noise that overwhelm the signal briefly and occasionally, giving rise to isolated single-bit errors or errors tend to come in bursts.

PARITY METHOD

- appends a parity bit to the end of each word in the frame
- Even parity is used for asynchronous Transmission
- Odd parity is used for synchronous Transmission

Ex 1.	Character code	even parity	odd parity
	1100100	1100100 1	1100100 0
2.	0011000	0011000 0	0011000 1

If one bit or any odd no bits is erroneously inverted during Transmission, the Receiver will detect an error. How ever if two or even no of bits are inverted an undetected error occurs.

Ex 3. The Transmitted data is 10011010. The received data is 11011010.

Let both the transmitter and receiver are agreed on EVEN parity. Now an error will be detected, since the no of ones received are ODD

4. The Transmitted data is 10011010. The received data is 01011010

The received data is wrong even though the no of ones are EVEN. Since two bits are inverted error can't be detected.

Longitudinal Redundancy Check(LRC)

- The frame is viewed as a block of characters arranged in 2-dimensions. To each character is appended a parity bit. In addition a parity bit is generated for each bit position across all characters i.e., an additional character is generated in which the I^{th} bit of the character is parity bit for the I^{th} bit of all other characters in the block. This can be expressed mathematically using exclusive OR(+) operation. The parity bit at the end of each character of row parity

$R_j = b_{1j} + b_{2j} +$---------- b_{nj}

Where R_j=Parity bit of jth character

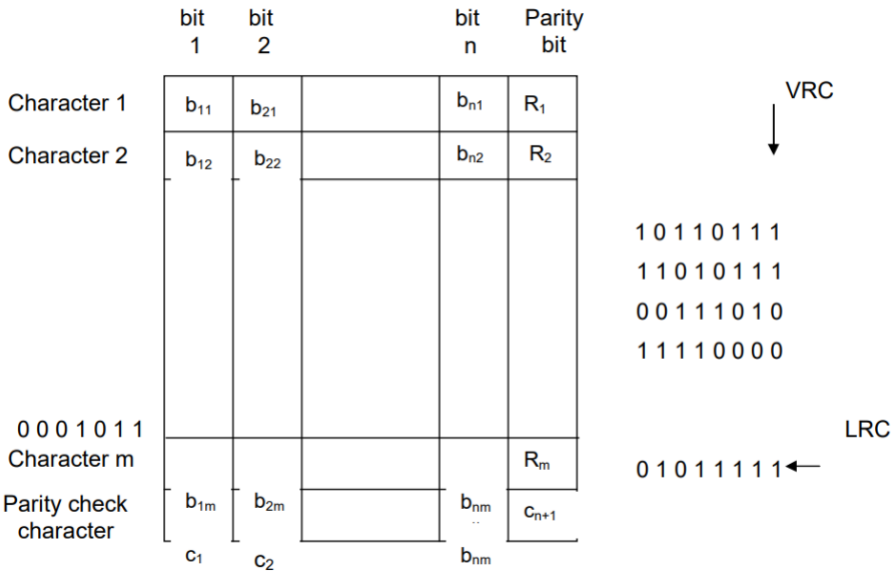
b_{ij}=ith bit in jth character

This equation generates even parity.

$C_i = b_{i1} + b_{i2} +$ ---------- $+ b_{in}$

Where C_i=ith bit of parity check character m=number of characters in a frame

In this format the parity bits at the end of each character are referred to as The Vertical Redundancy Check (VRC) and the Parity check character is referred to asthe Longitudinal Redundancy Check (LRC).



CRC Method

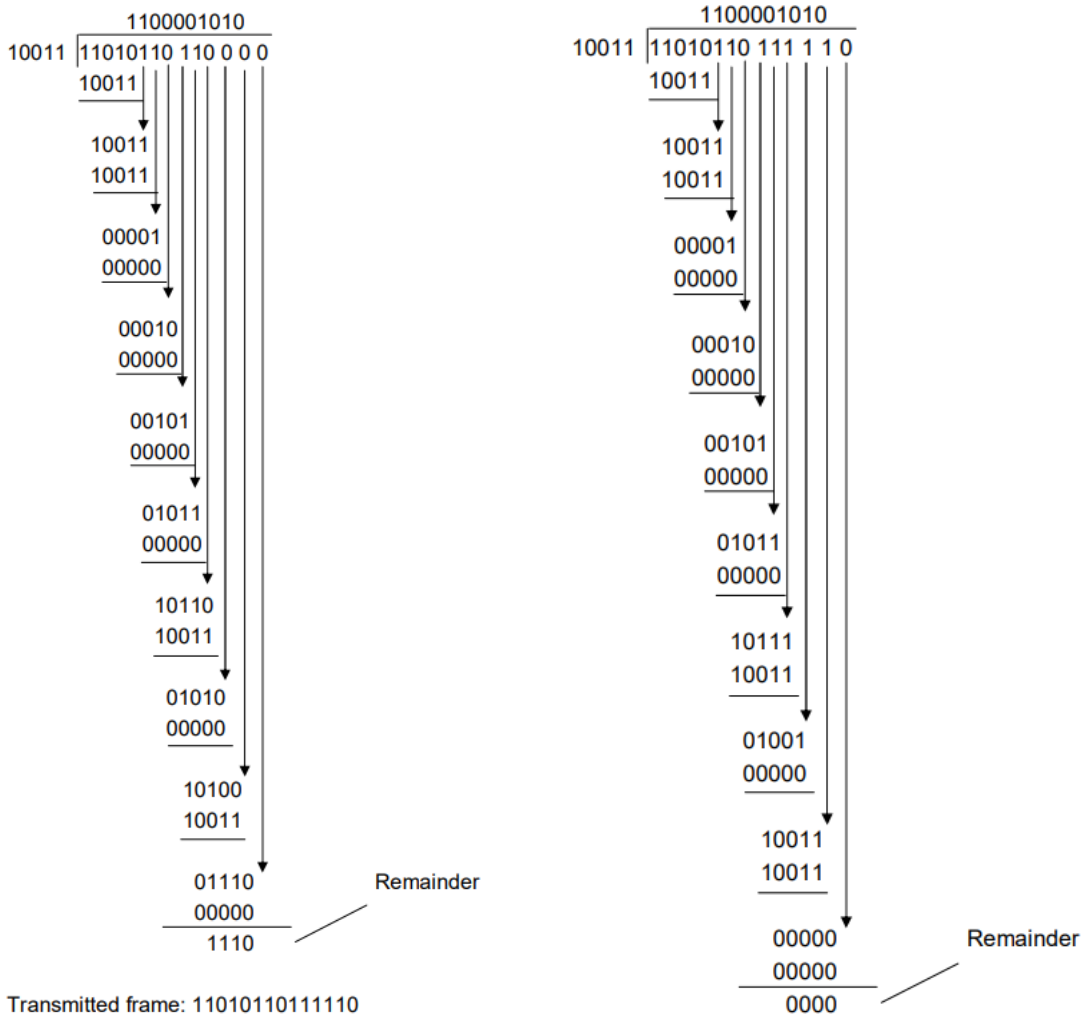
1. The frame is expressed in the form of a Polynomial F(x).0 1 1 1 1 1 1 0
2. Both the sender and receiver will agree upon a generator polynomial G(x) inadvance.
3. Let ‘r’ be the degree of G(x).Append ‘r’ zero bits to the lower — order end offrame now it contains m+r bits.

4. Divide the bit string by G(x) using Mod 2 operation.
5. Transmitted frame [T(x)] = frame + remainder

1. Divide T(x) by G(x) at the receiver end. If the result is a zero, then the frame istransmitted correctly. Ex. Frame: 1101011011

Generator: 10011

Message after appending 4 zero bits: 11010110000



Since the remainder is zero there is no error in the transmitted frame.

HAMMING CODES

Hamming codes provide another method for error correction. Error bits, called Hamming bits, are inserted into message bits at random locations. It is believed that the randomness of their locations reduces the odds that these Hamming bits themselves would be in error. This is based on a mathematical assumption that because there are so many more message bits compared with Hamming bits, there is a greater chance for a message bit to be in error than for a Hamming bit to be wrong. Determining the placement and binary value of the Hamming bits can be implemented using hardware, but it is often more practical to implement them using software. The number of bits in a message (M) are counted and used to solve the following equation to determine the number of Hamming bits (H) to be used:

$2^H \geq M + H + 1$

Once the number of Hamming bits is determined, the actual placement of the bits into the message is performed. It is important to note that despite the random nature of the Hamming bit placements, the exact sample placements

must be known and used by both the transmitter and receiver. Once the Hamming bits are inserted into their positions, the numerical values of the bit positions of the logic 1 bits in the original message are listed. The equivalent binary numbers of these values are added in the same manner as used in previous error methods by discarding all carry results. The sum produced is used as the states of the Hamming bits in the message. The numerical difference between the Hamming values transmitted and that produced at the receiver indicates the bit position that contains a bad bit, which is then inverted to correct it.

Ex. The given data 10010001100101(14- bits)

The number of hamming codes $2^H \geq M + H + 1$

H = ? M = 14 to satisfy this equation H should be 5 i.e. 5 hamming code bits should be incorporated in the data bits.

1 0 0 1 0 0 0 1 1 0 H 0 H 1 H 0 H 1 H

Now count the positions where binary 1's are present. Add using mod 2 operation (Ex-OR). The result will give the Hamming code at the transmitter end.

1's position	Binary equivalent
2 -	0 0 0 1 0
6 -	0 0 1 1 0
11 -	0 1 0 1 1
12 -	0 1 1 0 0
16 -	1 0 0 0 0
19 -	1 0 0 1 1
Hamming code =	0 0 0 0 0

This Hamming code will be incorporated at the places of 'H' in the data bits and the datawill be transmitted.

How to find out there is an error in the data?

Let the receiver received the 12^{th} bit as zero. The receiver also finds out the Hammingcode in the same way as transmitter.

1's position	Binary equivalent
2 -	0 0 0 1 0
6 -	0 0 1 1 0
11 -	0 1 0 1 1
16 -	1 0 0 0 0
19 -	1 0 0 1 1
Hamming code at the receiver	0 1 1 0 0

Hamming code at the Tx	0 0 0 0 0
Hamming code at the Rx	0 1 1 0 0
	0 1 1 0 0

The decimal equivalent for the binary is **12** so error is occurred at 12th place.

Elementary Data Link Protocols

Protocols in the data link layer are designed so that this layer can perform its basic functions: framing, error control and flow control. Framing is the process of dividing bit - streams from physical layer into data frames whose size ranges from a few hundred to a few thousand bytes.

Unrestricted Simplex Protocol:

Following assumptions are made

- Data transmission is simplex i.e. transmitted in one direction only.
- Both transmitting and receiving network layers are ready.
- Processing time is ignored.
- Infinite buffer space is available.
- An error free channel.

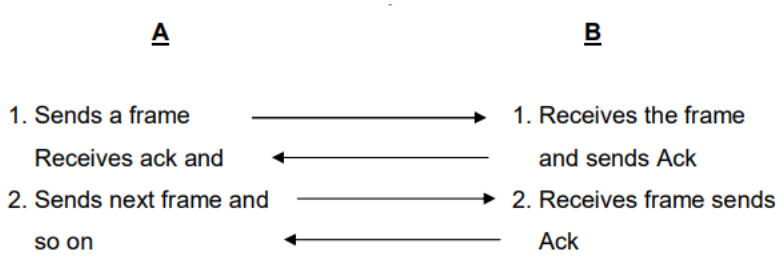
This is an unrealistic protocol, which has a nickname "Utopia".

1. A simplex stop and wait protocol:

The following assumptions are made

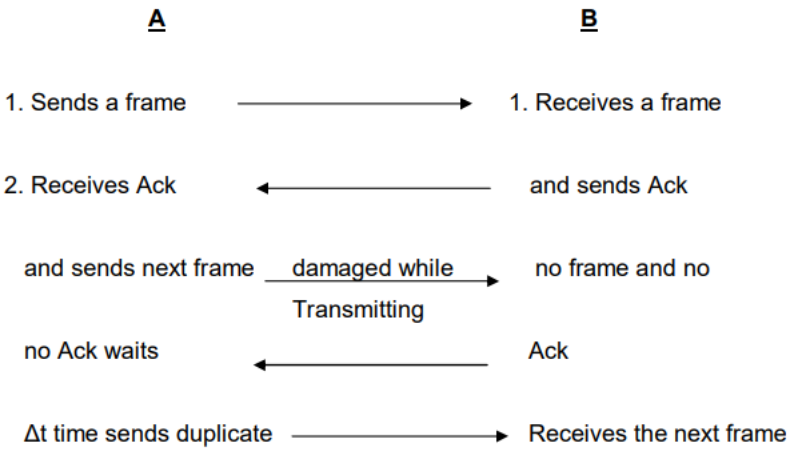
a. Error free channel.

b. Data transmission simplex.

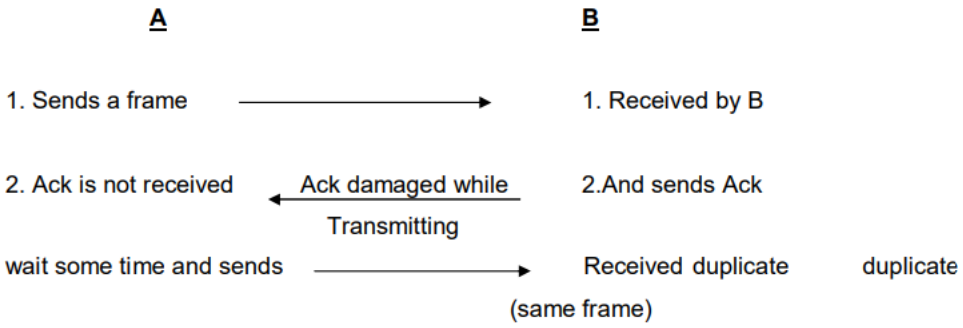


Since the transmitter waits for Δt time for an Ack this protocol is called stop and wait protocol.

3. A simplex protocol for a noisy channel



When this protocol fails?

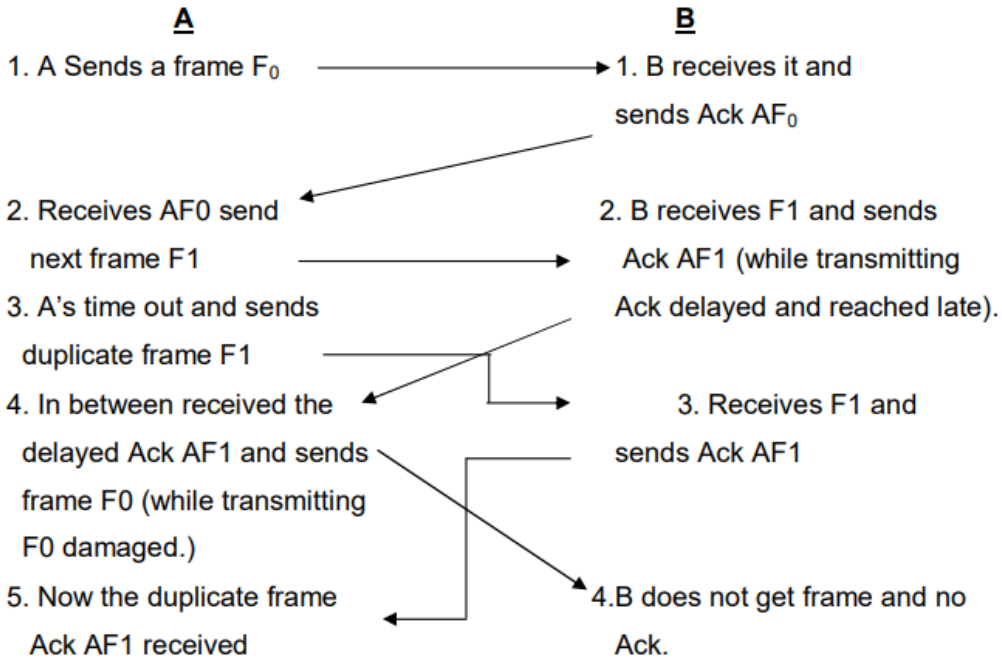


At this situation protocol fails because the receiver receives a duplicate frame and there isno way to find out whether the receiver frame is original or duplicate. So the protocol fails at this situation.

Now what is needed is some way for the Rx to distinguish a frame and a duplicate. To achieve this, the sender has to put a sequence number in the header of each frame it sends. The Rx can check the sequence number of each arriving frame to see if it is a new frame or a duplicate.

Here a question arises: What is the minimum number of bits needed for the sequence number? The ambiguity is between a frame and its successor. A 1-bit sequence number (0 or 1) is therefore sufficient. At each instant of time, the receiver excepts a particular sequence number next. Any arriving frame containing wrong sequence number is rejected as a duplicate. When a frame containing the correct sequence number arrives, it is accepted, passed to the network layer and then expected sequence number is incremented i.e. 0 becomes 1 and one becomes 0. Protocols in which a sender waits for a positive ack before advancing to the next data item are often called PAR (positive ack with retransmission) or ARQ (automatic repeat request).

When this protocol fails?



6. Now A thinks that the Ack received is the ack of new frame F0 and A sends nextframe F1. So a frame F0 is missed. At this situation this protocol fails.

PIGGY BACKING

In most practical situations there is a need of transmitting data in both directions. This canbe achieved by full duplex transmission. If this is done we have two separate physical circuits each with a 'forward ' and 'reverse' channel. In both cases, the reverse channel is almost wasted. To overcome this problem a technique called **piggy backing** is used.

The technique of temporarily delaying outgoing acknowledgements so that they can be hooked onto the next outgoing data frame is known as **piggy backing**.

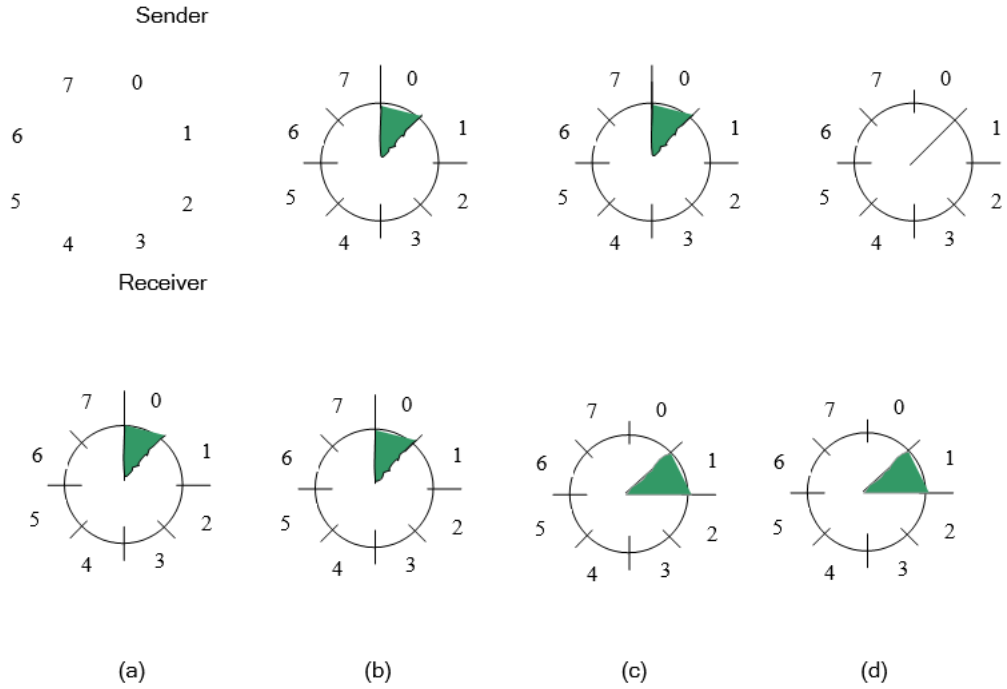
However, piggybacking introduces a complication not present with separate acknowledgements. How long should the data link layer wait longer than the sender's timeout period, the frame will be retransmitted, defeating the whole purpose of having acknowledgements. Of course, the data link layer cannot foretell the future, so it must resort to some ad hoc scheme, such as waiting a fixed number of milli seconds. If a new packet arrives quickly, the acknowledgement is piggy backed onto it; otherwise, if no new packet has arrived by the end of this time period, the data link layer just sends a separate acknowledgement frame.

SLIDING WINDOW PROTOCOLS

In all sliding window protocols, each outbound frame contains a sequence number, ranging from 0 up to some maximum. The maximum is usually $2^n - 1$ so the sequence number fits nicely in an n-bit field. The stop-and-wait sliding window protocol uses n=1, restricting the sequence numbers to 0 and 1, but more sophisticated versions can use arbitrary n.

The essence of all sliding window protocols is that at any instant of time, the sender maintains a set of sequence numbers corresponding to frames it is permitted to send. These frames are said to fall with in the sending window. Similarly the receiver also maintains a receiving window corresponding to the set of frames it is permitted to accept. The sender's window and the receiver's window need not have the same lower and upperlimits, or even have the same size. In some protocols they are fixed in size, but in others they can grow or shrink as frames are sent and received.

The sequence numbers with in the sender's window represent frames sent but as yet not acknowledged. Whenever a new packet arrives from the network layer, it is given the nexthighest sequence number, and the upper edge of the window is advanced by one. When an acknowledgement comes in, the lower edge is advanced by one. In this way the continuously maintains a list of unacknowledged frames.



(a) Initially (b) After the first frame has been sent c) After the first frame has been received. d) After the first acknowledgement has been received.

PIPELINING

1. Upto now we made the assumption that the transmission time required for a frame toarrive at the receiver plus the transmission time for the ack to come back is negligible.
2. Sometimes this is not true, when there is a long round trip propagation time is there.

3. In these cases round trip propagation time can have important implications for theefficiency of the bandwidth utilization.

Consider the below example.

Let the channel capacity b = 50Kbps.
round trip propagation delay = 500ms
Frame size = 1000bits

Without considering the round trip propagation delay
For one frame the time taken will be = 1000/500 ms = 20 ms

Considering the round trip propagation delay

For one frame the time taken will be = 500 ms +20 ms = 270 ms

The channel utilization = (20/520)*100 = 4%

i.e. We are wasting 96% of channel time. To overcome this problem we will go for a technique called **PIPELIING.**

In this technique, the sender is allowed to transmit upto ‘w ‘ frames before blocking,instead of just 1.With an appropriate choice of w the sender will be able to continuously transmit frames for a time equal to the round trip transmit time without filling up the window.

In the above example w would be at least 26 frames. (520/20 = 26 frames)

By the time it has finished sending 26 frames, at t=520 ms, the ack for frame 0 will have just arrived. Thereafter ack will arrive every 20 ms, so the sender always gets permission to continue just when it needs it. Hence, we can say the sender window size is 26.

Derivation:

Let the channel capacity = b Bps

Let the frame size = l bits

Let the round trip delay = R secs

To send one frame the time will be l/b secs

Due to round trip delay the time taken will be (l/b + R) Sec = l+Rb/b Sec

The channel utilization is l/b (l/b + R) Sec = (l / l + Rb) Sec

If l > bR the efficiency will be greater than 50%.If l < bR the efficiency will be less than 50%.

If l = bR the efficiency will be 50%.

Ex 1. A channel has a bit rate of 4 kbps and a propagation delay of 20msec.For what rage of frame sizes does stop and wait give an efficiency of at least 50 % ?

2. Consider an error free 64 kbps channel used to send 512 –byte data frames in one direction, with very short acknowledgements coming back the other way. What is the maximum throughput for window sizes of 1,7,15,and 127?

Pipelining frames over an unreliable channel raises some serious issues.

First, what happens if a frame in the middle of a long stream is damaged or lost? When a damaged frame arrives at the receiver, it obviously should discarded, but whatshould the receiver do with all the correct frames following it?

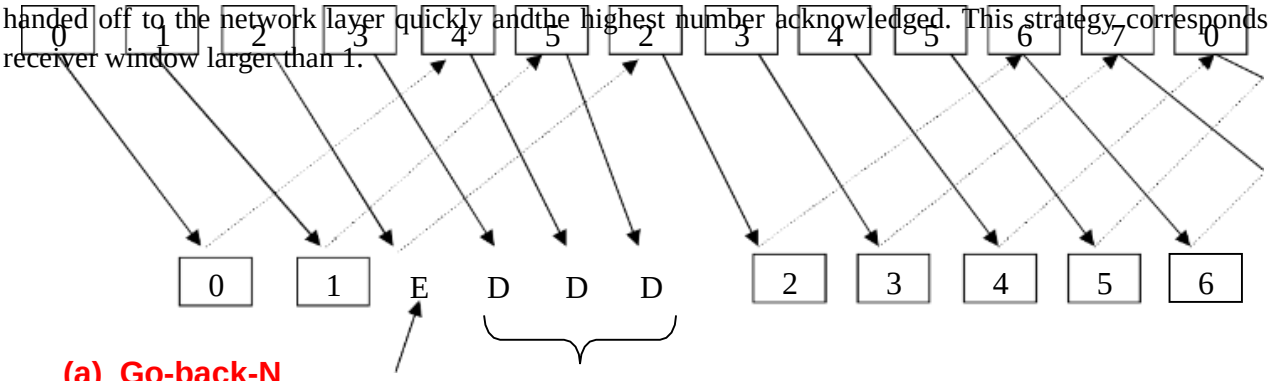
There are two basic approaches to dealing with errors .

1. Go Back 'n'
2. Selective repeat or Selective Reject

One way called in **go back n**, the receiver simply to discard all subsequent frames, sending no acknowledgements for the discard frames. In the other words, the data link layer refuses to accept any frame except the next one it must give to the network layer.

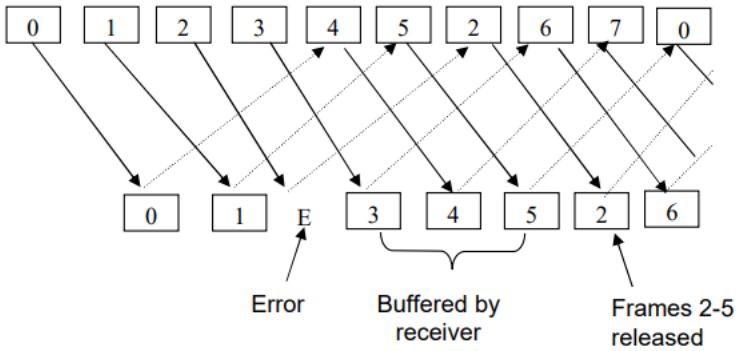
Selective Repeat:

The receiving data link layer store all the correct frames following the bad frame, not all itssuccessors. If the second try succeeds the receiving data link layer will now have many correct frames in sequence, so they can all be handed off to the network layer quickly andthe highest number acknowledged. This strategy corresponds to a receiver window larger than 1.



(a) Go-back-N

Error Discarded frames



b) Selective reject

MEDIUM ACCESS CONTROL SUBLAYER (MAC)

Networks can be categories in to two ways

a) Point to point

b) Broad cast channel

- In broadcast network, the key issue is how to share the channel amongseveral users.

- Ex *a conference call with five people*

-Broadcast channels are also called as multi-access channels or random accesschannels.

-Multi-access channel belong to a sublayer at the DL layer called the MAC sublayer.

The Channel Allocation problem:

a) **Static channel allocation** in LANs & MANs

i) **FDM** ii) **TDM**

Drawbacks: -1) Channel is wasted if one or more stations do not send data.

2) If users increases this will not support.

b) **Dynamic channel allocation**

i) **Pure ALOHA & Slotted ALOHA**

ii) **CSMA**

- CSMA/CA
- CSMA/CD

Pure ALOHA

-1970's Norman Abramson end his colleagues devised this method, used ground –basedradio broad costing. This is called the **ALOHA** system.

-The basic idea, many users are competing for the use of a single shared channel.

-There are two versions of ALOHA: **Pure and Slotted**.

-Pure ALOHA does not require global time synchronization, where as in slotted ALOHAthe time is divided into discrete slots into which all frames must fit.

-Let users transmit whenever they have data to be sent.

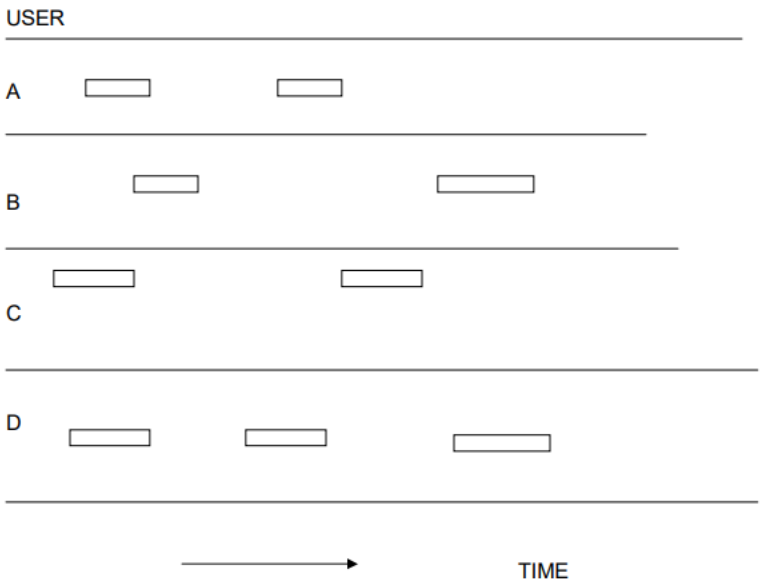
-There will be collisions and all collided frames will be damaged.

-Senders will know through feedback property whether the frame is destroyed or not bylistening channel.

[-With a LAN it is immediate, with a satellite, it will take 270m sec.]

-If the frame was destroyed, the sender waits random amount of time and again sendsthe frame.

-The waiting time must be random otherwise the same frame will collide over and over.



Frames are transmitted at completely arbitrary times

-Whenever two frames try to occupy the channel at the same time, there will be a collisionand both will be destroyed.

-We have to find out what is the efficiency of an ALOHA channel?

-Let us consider an infinite collection of interactive users sitting at their systems (stations).

-A user will always in two states **typing or waiting**.

-Let the 'Frame time' denotes the time required to transmit one fixed length frame.

-Assume that infinite populations of users are generating new frames according topossion distribution with mean N frames per frame time.

-If N>1 users are generating frames at a higher rate than the channel can handle.

-For reasonable throughput 0<N<1.

-In addition to new frames, the station also generates retransmission of frames.

-Old and new frames are G per frame time.

-$G \geq N$

-At low load there will be few collisions, so $G \sim N$

-Under all loads, the throughput $S = GP_o$, where P_o is the probability that a frame does notsuffer a collision.

-A frame will not suffer a collision if no other frames are sent with one frame time of itsstart.

-Let 't' be the time required to send a frame.

-If any other user has generated a frame between time t_o and t_o+t, the end of that framewill collide with the beginning of the shaded frame.

-Similarly, any other frame started b/w t_o+t and t_o+2t will bump into the end of the shadedframe.

-The probability that 'k' frames are generated during a given frame time is given by thepossion distribution:

$P_r[k] = \frac{G^k e^{-G}}{k!}$

-The probability of zero frames is just e^{-G}

-In an interval two frame times long, the mean number at frames generated is 2G.

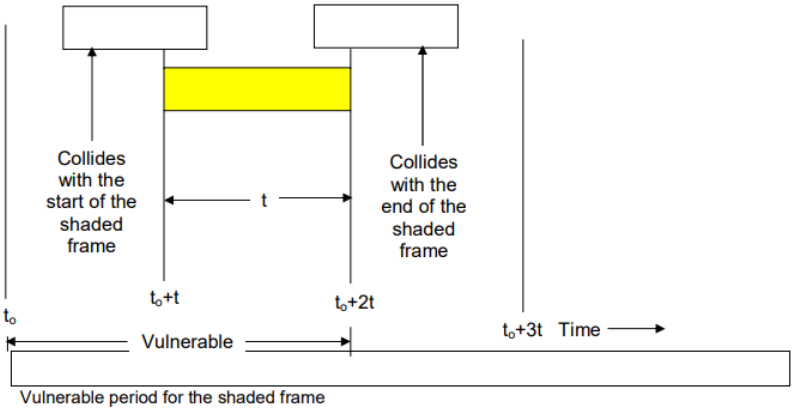
-The probability at no other traffic being initiated during the entire vulnerable period isgiven by

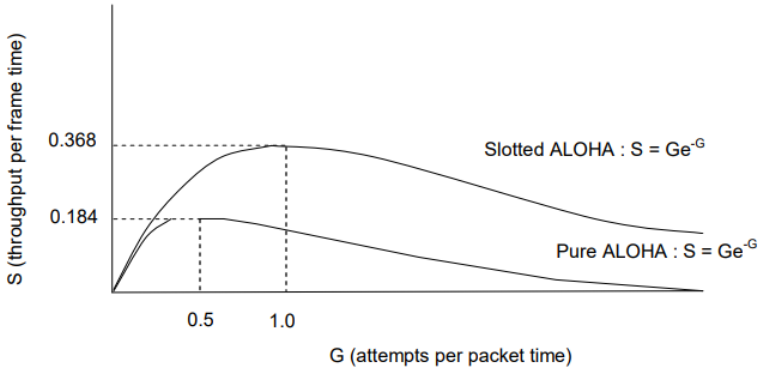
$P_O = e^{-2G}$

$S = Ge^{-2G}$ $[S=GP_o]$

The Maximum through put occurs at G=0.5 with S=1/2e = 0.184

The channel utilization at pure ALOHA =18%.





Throughput versus offered traffic for ALOHA systems

Slotted ALOHA

-In 1972, Roberts' devised a method for doubling the capacity of ALOHA system.

-In this system the time is divided into discrete intervals, each interval corresponding toone frame.

-One way to achieve synchronization would be to have one special station emit a pip at the start of each interval, like a clock.

-In Roberts' method, which has come to be known as slotted ALOHA, in contrast to Abramson's pure ALOHA; a computer is not permitted to send whenever a carriage returnis typed.

-Instead, it is required to wait for the beginning of the next slot.

-Thus the continuous pure ALOHA is turned into a discrete one.

-Since the vulnerable period is now halved, the of no other traffic during the same slot as our test frame is e^{-G} which leads to

$S = Ge^{-G}$

- At G=1, slotted ALOHA will have maximum throughput.
- So S=1/e or about 0.368, twice that of pure ALOHA.
- The channel utilization is 37% in slotted ALOHA.

Carrier Sense Multiple Access Protocols

Protocols in which stations listen for a carrier (transmission) and act accordingly are called carries sense protocols.

Persistent CSMA

When a station has data to send, it first listens to the channel to see if any one else is transmitting at that moment. If the channel is busy, the station waits until it become idle. When the station detects an idle channel, it transmits a frame. If a collision occurs, the station waits a random amount of time and starts all over again. The protocol is called 1- persistent also because the station transmits with a probability of 1 when it finds the channel idle.

The propagation delay has an important effect on the performance of the protocol. The longer the propagation delay the worse the performance of the protocol.

Even if the propagation delay is zero, there will be collisions. If two stations listen the channel, that is idle at the same, both will send frame and there will be collision.

Non persistent CSMA

In this, before sending, a station sense the channel. If no one else is sending, the station begins doing so it self. However, if the channel is busy, the station does not continually sense it but it waits a random amount of time and repeats the process.

This algorithms leads to better channel utilization but longer delays then 1-persistent CSMA.

With persistent CSMA, what happens if two stations become active when a third station is busy? Both wait for the active station to finish, then simultaneously launch a packet, resulting a collision. There are two ways to handle this problem.

a) P-persistent CSMA b) exponential backoff.

P-persistent CSMA

The first technique is for a waiting station not to launch a packet immediately when the channel becomes idle, but first toss a coin, and send a packet only if the coin comes up heads. If the coin comes up tails, the station waits

for some time (one slot for slotted CSMA), then repeats the process. The idea is that if two stations are both waiting for the medium, this reduces the chance of a collision from 100% to 25%. A simple generalization of the scheme is to use a biased coin, so that the probability of sending a packet when the medium becomes idle is not 0.5, but p, where 0< p < 1. We call such a scheme **P-persistent CSMA**. The original scheme, where p=1, is thus called 1-persitent CSMA.

Exponential backoff

The key idea is that each station, after transmitting a packet, checks whether the packet transmission was successful. Successful transmission is indicated either by an explicit acknowledgement from the receiver or the absence of a signal from a collision detection circuit. If the transmission is successful, the station is done. Otherwise, the station retransmits the packet, simultaneously realizing that at least one other station is also contending for the medium. To prevent its retransmission from colliding with the other station's retransmission, each station backs off (that is, idles) for a random time chosen from the interval [0,2*max-propagation_delay] before retransmitting its packet. If the retransmission also fails, then the station backs off for a random time in the interval [0,4* max_propagation_delay], and tries again. Each subsequent collision doubles the backoff interval length, until the retransmission finally succeeds. On a successful transmission,the backoff interval is reset to the initial value. We call this type of backoff exponential backoff.

CSMA/CA

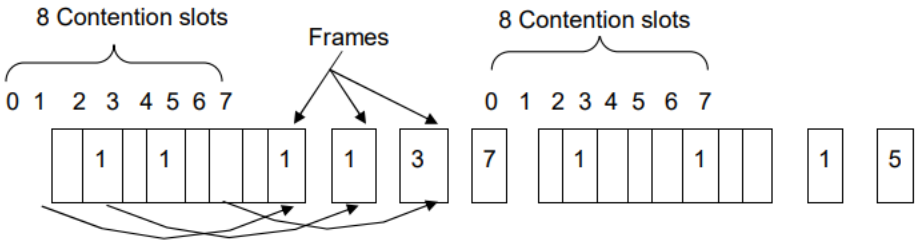
In many wireless LANS, unlike wired LANS, the station has no idea whether the packet collided with another packet or not until it receives an acknowledgement from receiver. In this situation, collisions have a greater effect on performance than with CSMA/CD, where colliding packets can be quickly detected and aborted. Thus, it makes sense to try to avoid collisions, if possible. CSMA/CA is basically p-persistence, with the twist that when the medium becomes idle, a station must wait for a time called the interframe spacing or IFS before contending for a slot. A station gets a higher priority if it is allocated smaller inter frame spacing.

When a station wants to transmit data, it first checks if the medium is busy. If it is, it continuously senses the medium, waiting for it to become idle. When the medium becomes idle, the station first waits for an interframe spacing corresponding to its priority level, then sets a contention timer to a time interval randomly selected in the range [0,CW], where CW is a predefined contention window length. When this timer expires, it transmits a packet and waits for the receiver to send an ack. If no ack is received, the packet is assumed lost to collision, and the source tries again, choosing a contention timer at random from an interval twice as long as the one before(binary exponential backoff). If the station senses that another station has begun transmission while it was waiting for the expiration of the contention timer, it does not reset its timer, but merely freezer it, and restarts the countdown when the packet completes transmission. In this way, stations that happen to choose a longer timer value get higher priority in the next round of contention.

Collision-Free Protocols

A Bit-Map Protocol

In the basic bit-map method, each contention period consists of exactly N slots. If station0 has a frame to send, it transmits a 1 bit during the zeroth slot. No other station is allowed to transmit during this slot. Regardless of what station 0 does, station 1 gets the opportunity to transmit a 1during slot 1, but only if it has a frame queued. In general, station j may announce the fact that it has a frame to send by inserting a 1 bit into slot j. after all N slots have passed by, each station has complete knowledge of which stations with to transmit.



The basic bit-map protocol

The basic bit-map protocol

Since everyone agrees on who goes next, there will never be any collisions. After the last ready station has transmitted its frame, an event all stations can easily monitor, another Nbit contention period is begun. If a station becomes ready just after its bit slot has passed by, it is out of luck and must remain silent until every station has had a chance and the bit map has come around again. Protocols like this in which the desire to transmit is broadcast before the actual transmission are called reservation protocols.

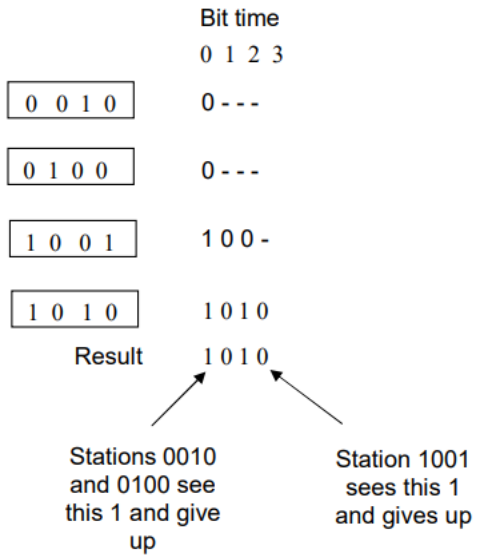
Binary Countdown

A problem with the basic bit-map protocol is that the overhead is 1 bit per station. A station wanting to use the channel now broadcasts its address as a binary bit string, starting with the high-order bit. All addresses are assumed to be the same length. Thebits in each address position from different stations are BOOLEAN ORed together. We will call this protocol binary countdown. It is used in Datakit.

As soon as a station sees that a high-order bit position that is 0 in its address has been overwritten with a 1, it gives up. For example, if station 0010,0100,1001, and 1010 are all trying to get the channel, in the first bit time the stations transmit 0,0,1, and 1, respectively. Stations 0010 and 0100 see the 1 and know that a higher-numbered station is competing for the channel, so they give up for the current round. Stations 1001 and 1010 continue.

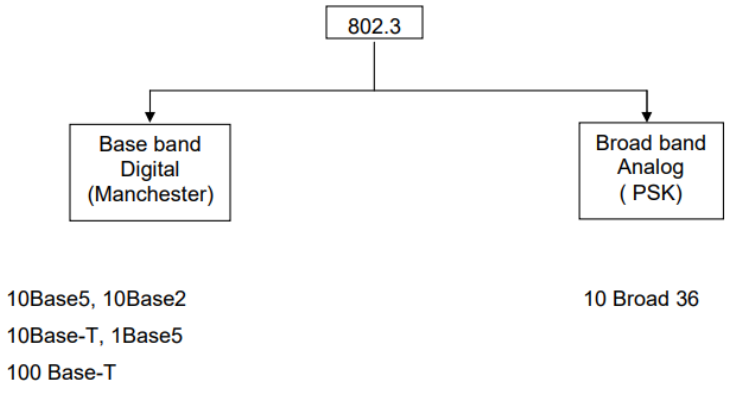
The next bit is 0, and both stations continue. The next bit is 1, so station 1001 gives up. The winner is station 1010, because it has the highest address. After winning the bidding, it may now transmit a frame, after which another bidding cycle starts.

The binary countdown protocol. A dash indicates silence



IEEE Standard 802 for LANS and MANS

The IEEE 802.3 is for a 1-persistent CSMA/CD LAN. Xerox built a 2.94 Mbps CSMA/CD system to connect over 100 personal workstations on 1-Km cable. This system was called Ethernet through which electromagnetic radiation was once thought to propagate. Xerox DEC and Intel came with another standard for 100 Mbps Ethernet. This differs from old one that it runs at speeds from 1 to 10 Mbps on various media. The second difference between these two is in one header (802.3 length field is used for packet type in Ethernet).



802.3 Cabling

Five types of cabling are commonly used, 10Base5 cabling called thick Ethernet, came first. It resembles a yellow garden hose, with markings every 2.5 m to show where thetaps go. Connections to it are generally made using **vampire taps**, in which a pin is carefully forced halfway into the coaxial cable's core. The notation 10Base5 means that it operates at 10 Mbps, uses baseband signaling, and can support segments of up to 500m.

Name	Cable	Max. segment	Nodes/seg.	Advantages
10Base5	Thick coax	500 m	100	Good for backbones
10Base2	Thin coax	200 m	30	Cheapest system
10Base-T	Twisted pair	100 m	1024	Easy maintenance
10Base-F	Fiber optics	2000 m	1024	Best between buildings

The second cable type was **10Base2** or thin Ethernet, which, in contrast to the garden- hose-like thick Ethernet, bends easily. Connections to it are made using industry standardBNC connectors to form T-junctions, rather than using vampire taps. These are easier to use and more reliable. Thin Ethernet is much cheaper and easier to install, but it can run for only 200m and can handle only 30 machines per cable segment.

Cable breaks, bad taps, or loose connectors can be detected by a devise called time domain reflectometry.

For 10Base5, a transceiver is clamped securely around the cable so that its tap makes contact with the inner core. The transceiver contains the electronics that handle carrier

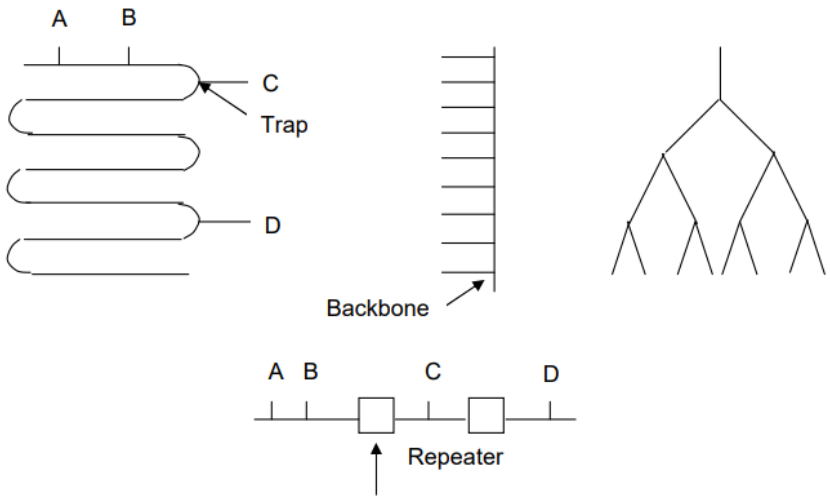
detection and collision detection. When a collision is detected, the transceiver also puts a special invalid signal on the cable to ensure that all other transceivers also realize that a collision has occurred.

The transceiver cable terminates on an interface board inside the computer. The interfaceboard contains a controller chip that transmits frames to, and receives frames from, the transceiver. The controller is responsible for assembling the data into the proper frame format, as well as computing checksums on outgoing frames and verifying them on incoming frames.

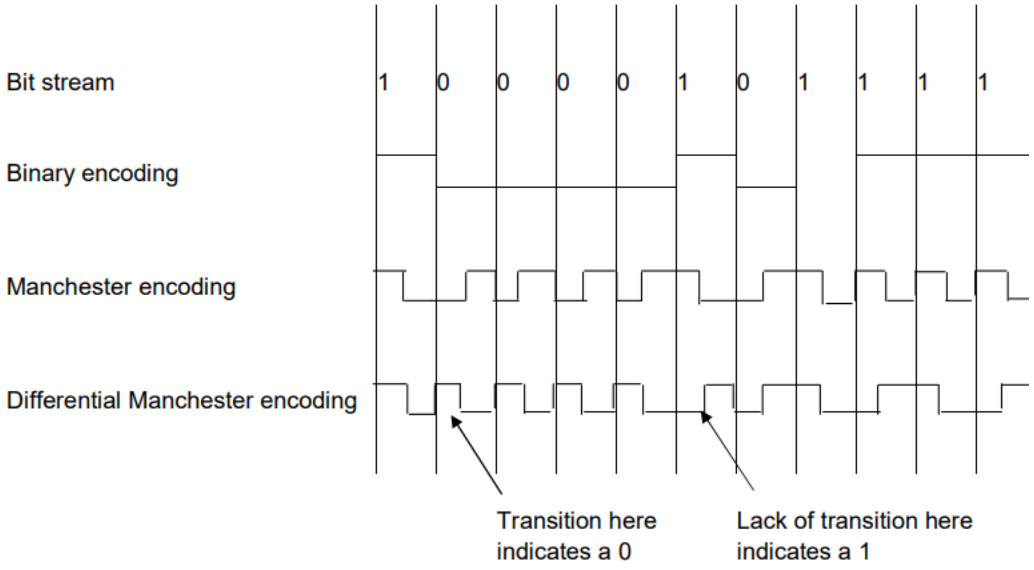
With 10Base2, the connection to the cable is just a passive BNC T-junction connector.The transceiver electronics are on the controller board, and each station always has its own transceiver.

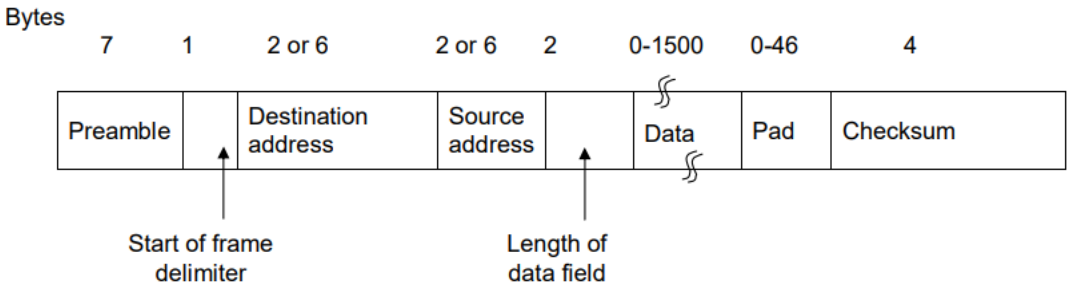
With 10Base-T, there is no cable at all, just the hub (a box full of electronics). Adding or removing a station is simple in this configuration, and cable breaks can be detected easily. The disadvantage of 10Base-T is that the maximum cable run from the hub is only 100m, may be 150m if high-quality (category 5) twisted pairs are used. 10Base-Tis becoming steadily more popular due to the ease of maintenance. 10Base-F, which uses fiber optics. This alternative is expensive due to the cost of the connectors and terminators, but it has excellent noise immunity and is the method of choice when running between buildings or widely separated hubs.

Each version of 802.3 has a maximum cable length per segment. To allow larger networks, multiple cables can be connected by repeaters. A repeater is a physical layer device. It receives, amplifies, and retransmits signals in both directions. As far as the software is concerned, a series of cable segments connected by repeaters is no different than a single cable (except for some delay introduced by the repeater). A system may contain multiple cable segments and multiple repeaters, but no two transceivers may be more than 2.5km apart and no path between any two transceivers any traverse more than four repeaters.



802.3 uses Manchester Encoding and differential Manchester Encoding





The 802.3 MAC sub layer protocol:

I) Preamble:

Each frame start with a preamble of 7 bytes each containing a bit pattern 10101010.

II) Start of frame byte:

It denotes the start of the frame itself. It contains 10101011.

III) Destination address:

This gives the destination address. The higher order bit is zero for ordinary address and 1for group address (Multi casting). All bits are 1s in the destination field frame will be delivered to all stations (Broad casting).

The 46^{th} bit (adjacent to the high-order bit) is used to distinguish local from global addresses.

IV) Length field:

This tells how many bytes are present in the data field from 0 to 1500.

V) Data field:

This contains the actual data that the frame contains.

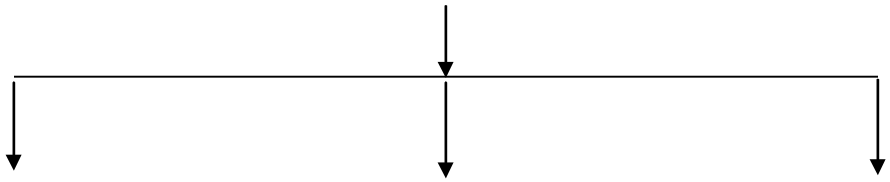
VI) Pad:

Valid frame must have 64 bytes long from destination to checksum. If the frame size less than 64 bytes pad field is used to fill out the frame to the minimum size.

VII) Checksum:

It is used to find out the receiver frame is correct or not. CRC will be used here.

Other Ethernet Networks

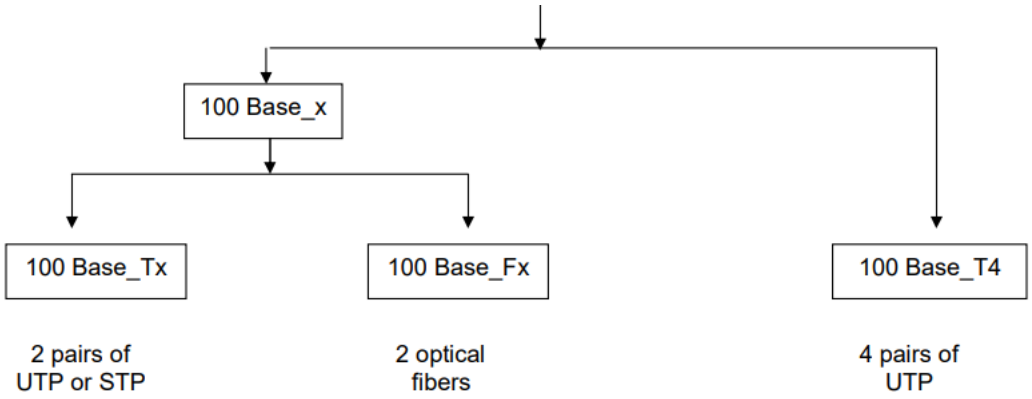


Switched Fast Gigabit

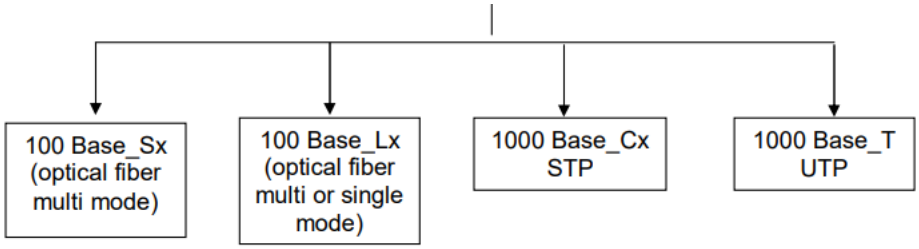
Switched Ethernet:

- 10 Base-T Ethernet is a shared media network.
- The entire media is involved in each transmission.
- The HUB used in this network is a passive device. (not intelligent).
- In switched Ethernet the HUB is replaced with switch. Which is a active device(intelligent)

Fast Ethernet

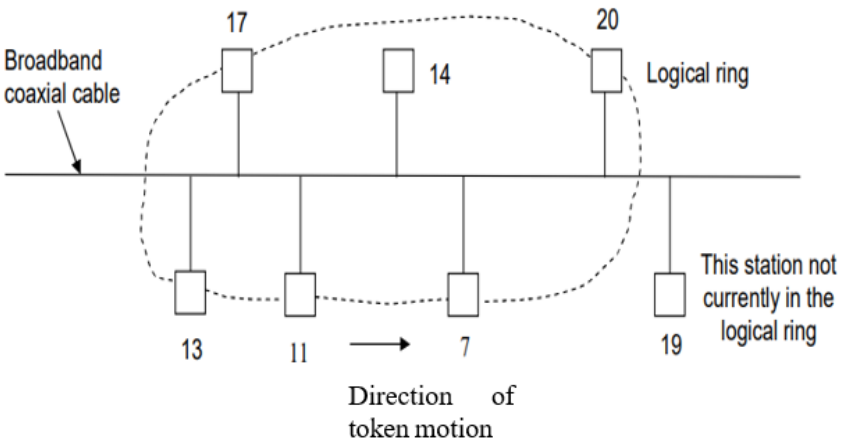


Gigabit Ethernet



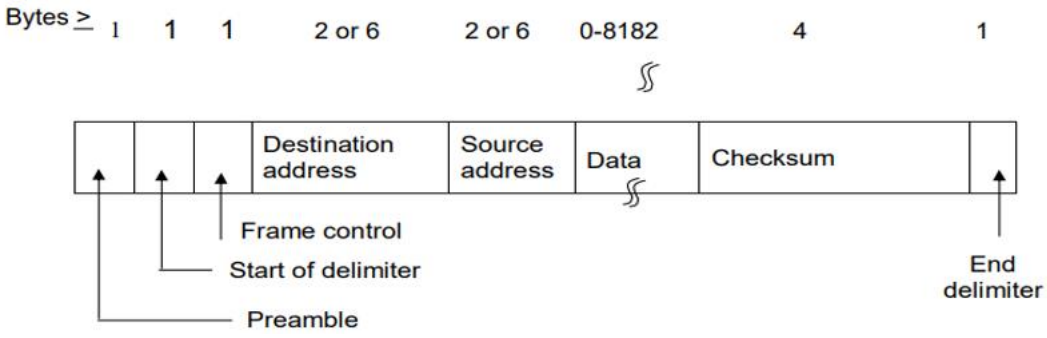
IEEE 802.4 (Token Bus)

802.3 frames do not have priorities, making them unsuited for real-time systems in which important frames should not be held up waiting for unimportant frames. A simple system with a known worst case is a ring in which the stations take turns sending frames. If there are n stations and it takes T sec to send a frame, no frame will ever have to wait more than nT sec to be sent.



This standard,802.4, described as a token bus. Physically, the token bus is a linear or tree-shaped cable onto which the stations are attached. Logically, the stations are organized into a ring, with each station knowing the address of the station to its "left" and "right." When the logical ring is initialized, the highest numbered station may send the first frame. After it is done, it passes permission to its immediate neighbor by sending the neighbor a special control frame called a token. The token propagates around the logical ring, with only the token holder being permitted to transmit frames. Since only one station at a time holds the token, collisions do not occur.

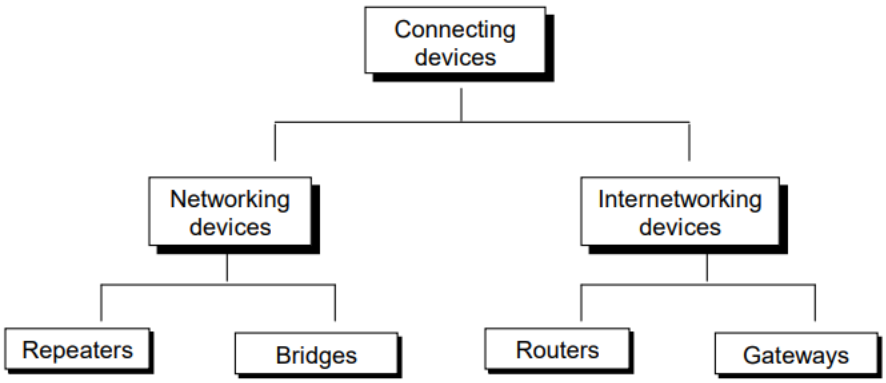
Since the cable is inherently a broadcast medium, each station receives each frame, discarding those not addressed to it. When a station passes the token, it sends a token frame specifically addressed to its logical neighbor in the ring, irrespective of where that station is physically located on the cable. It is also worth noting that when stations are firstpowered on, they will not be in the ring, so the MAC protocol has provisions for adding stations to, and deleting stations from, the ring. For the physical layer, the token bus uses the 75-ohm broadband coaxial cable used for cable television. Both single and dual-cable systems are allowed, with or without head-ends.



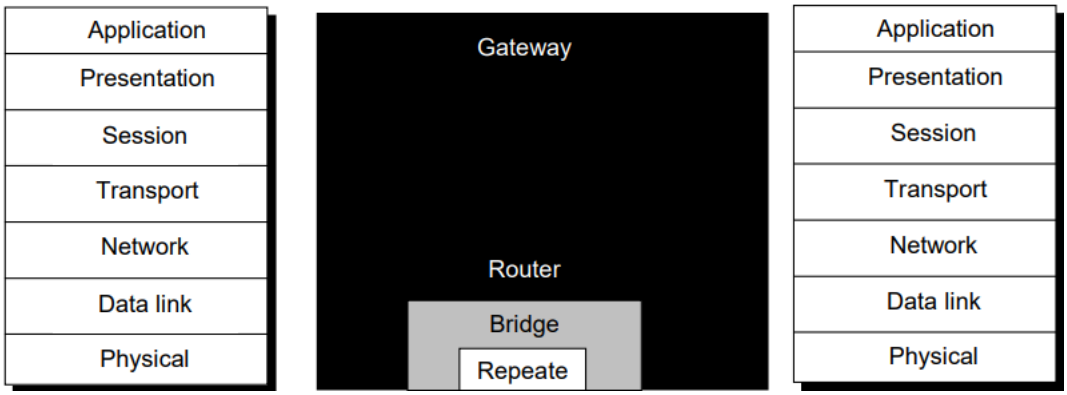
The frame control field is used to distinguish data frames from control frames. Fro data frames, it carries the frame's priority. It can also carry an indicator requiring the destination station to acknowledge correct or incorrect receipt of the frame.

For control frames, the frame control field is used to specify the frame type. The allowed types include token passing and various ring maintenance frames, including the mechanism for letting new stations enter the ring, the mechanism for allowing stations to leave the ring, and so on.

Connecting devices



Connecting devices and the OSI model



Bridges

LANS can be connected by devices called bridges, which operate in the data link layer. Bridges do not examine the network layer header and can thus copy IP, IPX, and OSI packets equally well.

The various reasons why the bridges are used.

1. Many university and corporate departments have their own LANS, primarily to connect their own personal computers, workstations, and servers. Since the goals of the various departments differ, different departments choose different LANS, without regard to what other departments are doing. Sooner or later, there is a need for interaction, so bridgesare needed.
2. The organization may be geographically spread over several buildings separated by considerable distances. It may be cheaper to have separate LANS in each building and connect them with bridges and infrared links than to run a single coaxial cable over the entire site.
3. It may be necessary to split what is logically a single LAN into separate LANS to accommodate the load. Putting all the workstations on a single LAN- the total bandwidth needed is far too high. Instead multiple LANS connected by bridges are used.
4. In some situations, a single LAN would be adequate in terms of the load, but the physical distance between the most distant machines is too great (e.g., more than 2.5km for 802.3). Even if laying the cable is easy to do, the

network would not work due to the excessively long round-trip delay. Only solution is to partition the LAN and install bridges between the segments.

5. There is the matter of reliability. On a single LAN, a defective node that keeps outputting a continuous stream of garbage will cripple the LAN. Bridges can be inserted atcritical places, to prevent a single node which has gone berserk from bringing down the entire system.
6. And last, bridges can contribute to the organization's security. By inserting bridges atvarious places and being careful not to forward sensitive traffic, it is possible to isolateparts of the network so that its traffic cannot escape and fall into the wrong hands.

Types of Bridges

Simple Bridge

Simple bridges are the most primitive and least expensive type of bridge. A simple bridge links two segments and contains a table that lists the addresses of all the stationsincluded in each of them. Before a simple bridge can be used, an operator must sit down and enter the addresses of every station. Whenever a new station is added, the table must be modified. If a station is removed, the newly invalid address must be deleted. Installation and maintenance of simple bridges are time-consuming and potentially more trouble than the cost savings are worth.

Transparent Bridge

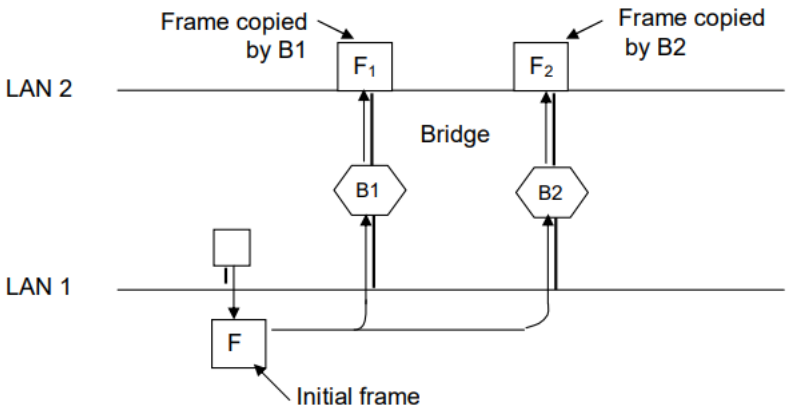
A transparent, or learning, bridge builds its table of station addresses on its own as it performs its bridge functions. When the transparent bridge is first installed, its table is empty .As it encounters each packet, it looks at both the destination and the source addresses. It checks the destination to decide where to send the packet. If it does not yet recognize the destination address, it relays the packet to all of the stations on both segments. It uses the source address to build its table. As it reads the source address, it notes which side the packet came from and associates that address with the segment to which it belongs. By continuing this process even after the table is complete, a transparent bridge is also self-updating.

This bridge uses flooding and backward landing algorithms.

The routing procedure for an incoming frame depends on the LAN it arrives on (the source LAN) and the LAN its destination is on (the destination LAN), as follows.

1) If destination and source LANS are the same, discard the frame.
2) If the destination and source LANS are different, forward the frame.
3) If the destination LAN is unknown, use flooding.

Two Parallel transparent bridges



Spanning Tree Algorithm

Bridges are normally installed redundantly, which means that two LANs may be connected by more than one bridge. In this case, if the bridges are transparent bridges, they may create a loop, which means a packet may be going round and round, from one LAN to another and back again to the first LAN. To avoid this situation, bridges today use what is called the **spanning tree algorithm.**