**Comparison of the OSI and TCP/IP Reference Models**

**Similarities**

The OSI and TCP/IP reference models have much in common. Both are based on the concept of a stack of independent protocols. Also, the functionality of the layers is roughly similar. For example, in both models the layers up through and including the transport layer are there to provide an end-to-end, network-independent transport service to processes wishing to communicate. These layers form the transport provider. Again in both models, the layers above transport are application-oriented users of the transport service.

**Differences**

Despite these fundamental similarities, the two models also have many differences.

1. The OSI model makes the distinction between the three concepts, i.e., services, interfaces and protocols explicitly. The TCP/IP model did not originally clearly distinguish between services, interfaces, and protocols, although people have tried to retrofit it after the fact to make it more OSI-like.
2. The protocols in the OSI model are better hidden than in the TCP/IP model and can be replaced relatively easily as the technology changes. Being able to make such changes transparently is one of the main purposes of having layered protocols in the first place.
3. The OSI reference model was devised *before* the corresponding protocols were invented. This ordering meant that the model was not biased toward one particular set of protocols, a fact that made it quite general. But the model is not suitable for internetworking. With TCP/IP, the protocols came first, and the model was really just a description of the existing protocols. There was no problem with the protocols fitting the model. They fit perfectly. The only trouble was that the *model* did not fit any other protocol stacks. Consequently, it was not especially useful for describing other, non-TCP/IP networks.
4. An obvious difference between the two models is the number of layers: the OSI model has seven layers and the TCP/IP model has four. Both have (inter)network, transport, and application layers, but the other layers are different.
5. Another difference is in the area of connectionless versus connection-oriented communication. The OSI model supports both connectionless and connection-oriented communication in the network layer, but only connection-oriented communication in the transport layer, where it counts (because the transport service is visible to the users). The TCP/IP model supports only one mode in the network layer (connectionless) but both in the transport layer, giving the users a choice. This choice is especially important for simple request-response protocols.

**Design Issues for the Layers**

Reliability is the design issue of making a network that operates correctly even though it is made up of a collection of components that are themselves unreliable. Think about the bits of a packet traveling through the network. There is a chance that some of these bits will be received damaged (inverted) due to fluke electrical noise, random wireless signals, hardware flaws, software bugs and so on. It is important to find and fix these errors? One mechanism for finding errors in received information uses codes for **error detection**. Information that is incorrectly received can then be retransmitted until it is received correctly. More powerful codes allow for **error correction**, where the correct message is recovered from the possibly incorrect bits that were originally received. Both of these mechanisms work by adding redundant information. They are used at low layers, to protect packets sent over individual links, and high layers, to check that the right contents were received. Another reliability issue is finding a working path through a network. Often there are multiple paths between a source and destination, and in a large network, there may be some links or routers that are broken. Suppose that the network is down in Germany. Packets sent from London to Rome via Germany will not get through, but we could instead send packets from London to Rome via Paris. The network should automatically make this decision. This topic is called **routing**.

A second design issue concerns the evolution of the network. Over time, networks grow larger and new designs emerge that need to be connected to the existing network. We have recently seen the key structuring mechanism used to support change by dividing the overall problem and hiding implementation details: **protocol layering**. There are many other strategies as well. Since there are many computers on the network, every layer needs a mechanism for identifying the senders and receivers that are involved in a particular message. This mechanism is called **addressing** or **naming**, in the low and high layers, respectively. An aspect of growth is that different network technologies often have different limitations. For example, not all communication channels preserve the order of messages sent on them, leading to solutions that number messages. Another example is differences in the maximum size of a message that the networks can transmit. This leads to mechanisms for disassembling, transmitting, and then reassembling messages. This overall topic is called **internetworking**. When networks get large, new problems arise. Cities can have traffic jams, a shortage of telephone numbers, and it is easy to get lost. Not many people have these problems in their own neighborhood, but citywide they may be a big issue. Designs that continue to work well when the network gets large are said to be **scalable**.

A third design issue is resource allocation. Networks provide a service to hosts from their underlying resources, such as the capacity of transmission lines. To do this well, they need mechanisms that divide their resources so that one host does not interfere with another too much. Many designs share network bandwidth dynamically, according to the short term needs of hosts, rather than by giving each host a fixed fraction of the bandwidth that it may or may not use. This design is called **statistical multiplexing**, meaning sharing based on the statistics of demand. It can be applied at low layers for a single link, or at high layers for a network or even applications that use the network. An allocation problem that occurs at every level is how to keep a fast sender from swamping a slow receiver with data. Feedback from the receiver to the sender is often used. This subject is called **flow control**. Sometimes the problem is that the network is oversubscribed because too many computers want to send too much traffic, and the network cannot deliver it all. This overloading of the network is called **congestion**. One strategy is for each computer to reduce its demand when it experiences congestion. It, too, can be used in all layers. It is interesting to observe that the network has more resources to offer than simply bandwidth. For uses such as carrying live video, the timeliness of delivery matters a great deal. Most networks must provide service to applications that want this **real-time** delivery at the same time that they provide service to applications that want high throughput. **Quality of service** is the name given to mechanisms that reconcile these competing demands.

The last major design issue is to secure the network by defending it against different kinds of threats. One of the threats we have mentioned previously is that of eavesdropping on communications. Mechanisms that provide **confidentiality** defend against this threat, and they are used in multiple layers. Mechanisms for **authentication** prevent someone from impersonating someone else. They might be used to tell fake banking Web sites from the real one, or to let the cellular network check that a call is really coming from your phone so that you will pay the bill. Other mechanisms for **integrity** prevent surreptitious changes to messages, such as altering ‘‘debit my account $10’’ to ‘‘debit my account $1000.’’ All of these designs are based on cryptography

**MAC Addresses**

The MAC address, also known as the physical address or link address, is the address of a node as defined by its LAN or WAN. It is included in the frame used by the data link layer. It is the lowest-level address. The physical addresses have authority over the network (LAN or WAN). The size and format of these addresses vary depending on the network. For example, Ethernet uses a 6-byte (48-bit) physical address that is imprinted on the network interface card (NIC) and nominally written in **hexadecimal notation** is shown below.

06:01:02:01:2C:4B

LocalTalk (Apple), however, has a 1-byte dynamic address that changes each time the station comes up.

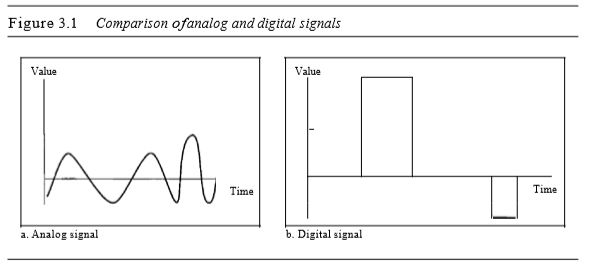
**IP Addresses**

IP addresses also called logical addresses are necessary for universal communications that are independent of underlying physical networks. Physical addresses are not adequate in an internetwork environment where different networks can have different address formats. A universal addressing system is needed in which each host can be identified uniquely, regardless of the underlying physical network. The logical addresses are designed for this purpose. A logical address in the Internet is currently a 32-bit address that can uniquely define a host connected to the Internet. No two publicly addressed and visible hosts on the Internet can have the same IP address.

**Analog and Digital signals.**

Data can be analog or digital. The term analog data refers to information that is continuous; digital data refers to information that has discrete states.

Digital data take on discrete values.



**Periodic and Non-Periodic Signals.**

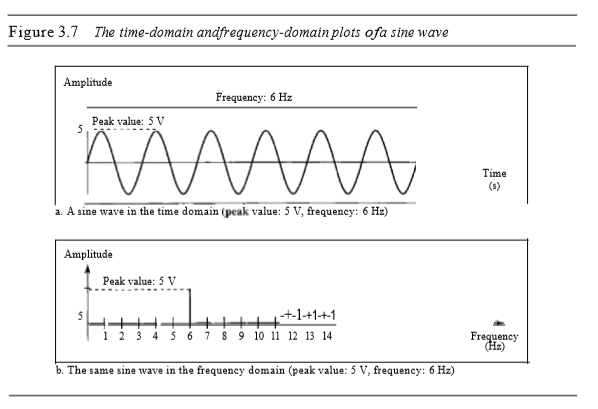
A periodic signal completes a pattern within a measurable time frame, called a period, and repeats that pattern over subsequent identical periods. The completion of one full pattern is called a cycle. A non-periodic signal changes without exhibiting a pattern or cycle that repeats over time. Both analog and digital signals can be periodic or non-periodic.

**Period and Frequency**

Period refers to the amount of time, in seconds, a signal needs to complete 1 cycle. Frequency refers to the number ofperiods in I s. Note that period and frequency are just one characteristic defined in two ways. Period is the inverse offrequency, and frequency is the inverse ofperiod, as the following formulas show

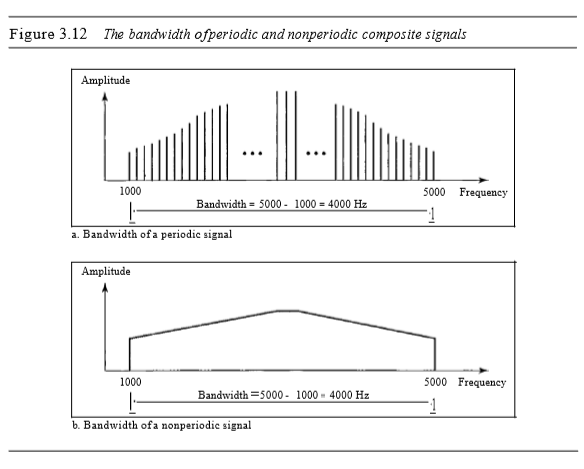
**Time and Frequency Domains**

A sine wave is comprehensively defined by its amplitude, frequency, and phase. We have been showing a sine wave by using what is called a time-domain plot. The time-domain plot shows changes in signal amplitude with respect to time (it is an amplitude-versus-time plot). Phase is not explicitly shown on a time-domain plot. To show the relationship between amplitude and frequency, we can use what is called a frequency-domain plot. A frequency-domain plot is concerned with only the peak value and the frequency.



**Bandwidth**

The range of frequencies contained in a composite signal is its bandwidth. The bandwidth is normally a difference between two numbers.The bandwidth ofa composite signal is the difference between the highest and the lowest frequencies contained in thatsignal.



**Baud Rate**

The **data rate** (**Bit Rate**) defines the number of data elements (bits) sent in one second. The unit is bits per second (bps). The **baud** (**Symbol**) **rate** is the number of signal elements (symbols) sent in one second. The unit is the **baud**. The baud rate is sometimes called the **pulse rate**, the **modulation rate**, or the **signal rate**. The bit rate is the symbol rate multiplied by the number of bits per symbol.

**Latency (Delay)**

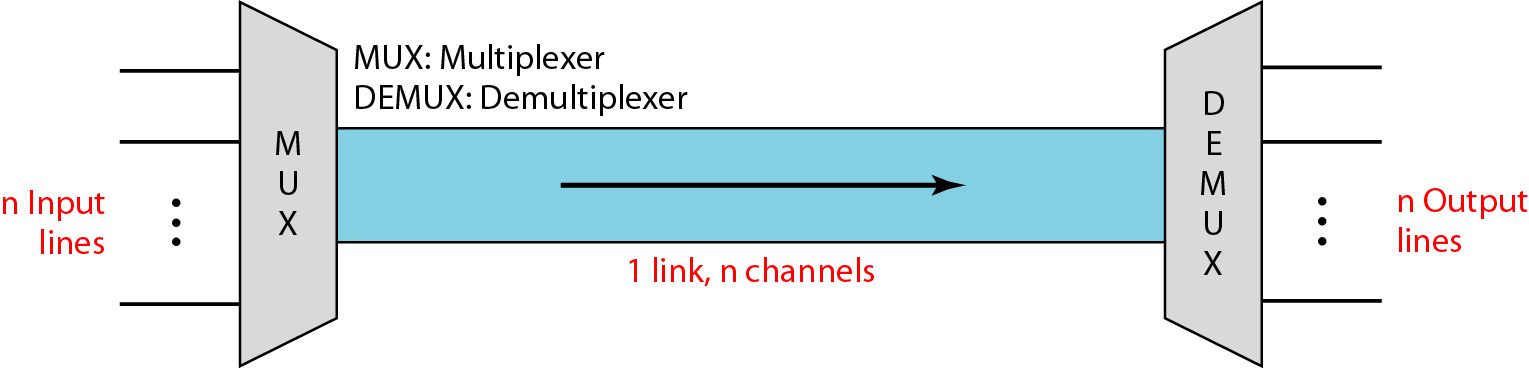
The latency or delay defines how long it takes for an entire message to completely arrive at the destination from the time the first bit is sent out from the source. We can say that latency is made of four components: propagation time, transmission time, queuing time and processing delay.

Latency = propagation time + transmission time + queuing time + processing delay

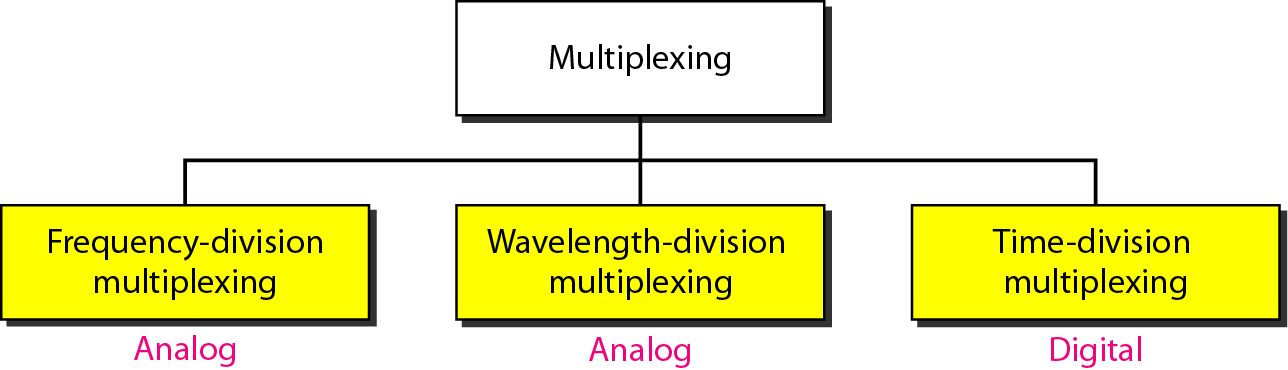
**Q. Explain types of multiplexing**

Whenever the bandwidth of a medium linking two devices is greater than the bandwidth needs ofthe devices, the link can be shared. Multiplexingis the setoftechniques that allows the simultaneous transmission of multiple signals across a single data link. As data and telecommunications use increases, so does traffic. We can accommodate this increase by continuing to add individual links each time a new channel is needed; or we can install higher-bandwidth links and use each to carry multiple signals.

In a multiplexed system, n lines share the bandwidth ofone link. Figure 6.1 shows the basic format ofa multiplexed system. The lines on the left direct their transmission streams to a multiplexer (MUX), which combines them into a single stream (many-to-one). At the receiving end, that stream is fed into a demultiplexer (DEMUX), which separates the stream back into its component transmissions (one-to-many) and directs them to their corresponding lines. In the figure, the word link refers to the physical path. The word channel refers to the portion of a link that carries a transmission between a given pair oflines. One link can have many (n) channels.

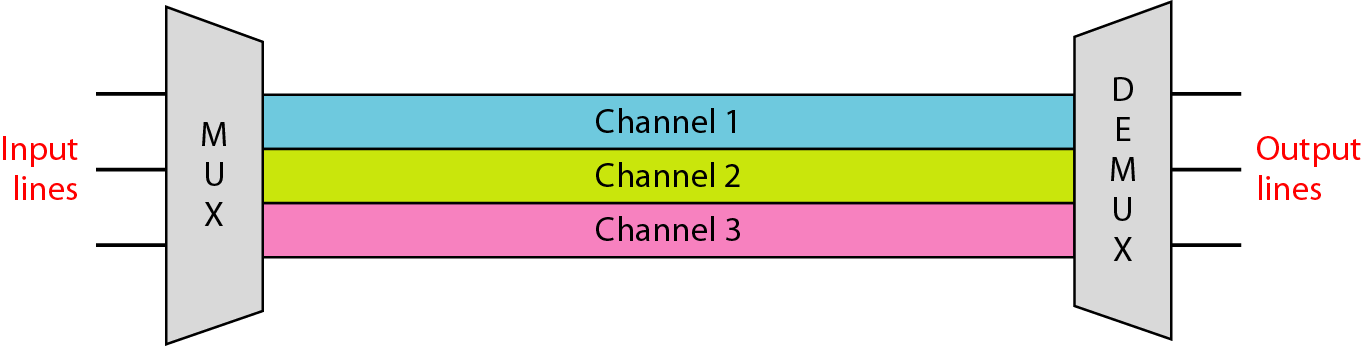


There are three basic multiplexing techniques: frequency-division multiplexing, wavelength-division multiplexing, and time-division multiplexing. The first two are techniques designed for analog signals, the third, for digital signals.



**Frequency-Division Multiplexing**

Frequency-division multiplexing (FDM) is an analog technique that can be applied when the bandwidth of a link (in hertz) is greater than the combined bandwidths of the signals to be transmitted. In FOM, signals generated by each sending device modulate different carrier frequencies. These modulated signals are then combined into a single composite signal that can be transported by the link. Carrier frequencies are separated by sufficient bandwidth to accommodate the modulated signal. These bandwidth ranges are the channels through which the various signals travel. Channels can be separated bystrips of unused bandwidth-guard bands-to prevent signals from overlapping. In addition, carrier frequencies must not interfere with the original data frequencies. Figure 6.3 gives a conceptual view ofFDM. In this illustration, the transmission path is divided into three parts, each representing a channel that carries one transmission.



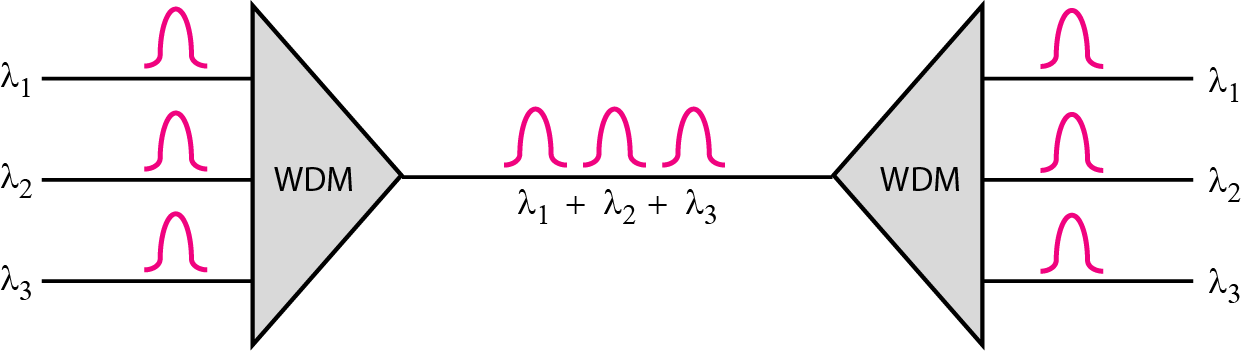
We consider FDM to be an analog multiplexing technique; however, this does not mean that FDM cannot be used to combine sources sending digital signals. A digital signal can be converted to an analog signal (with the techniques discussed in Chapter 5) before FDM is used to multiplex them.

NOTE:FDM is an analog multiplexing technique that combines analog signals.

**Wavelength-Division Multiplexing**

Wavelength-division multiplexing (WDM) is designed to use the high-data-rate capability of fiber-optic cable. The optical fiber data rate is higher than the data rate of metallic transmission cable. Using a fiber-optic cable for one single line wastes the available bandwidth. Multiplexing allows us to combine several lines into one.

WDM is conceptually the same as FDM, except that the multiplexing and demultiplexing involve optical signals transmitted through fiber-optic channels. The idea is the same: We are combining different signals of different frequencies. The difference is that the frequencies are very high.

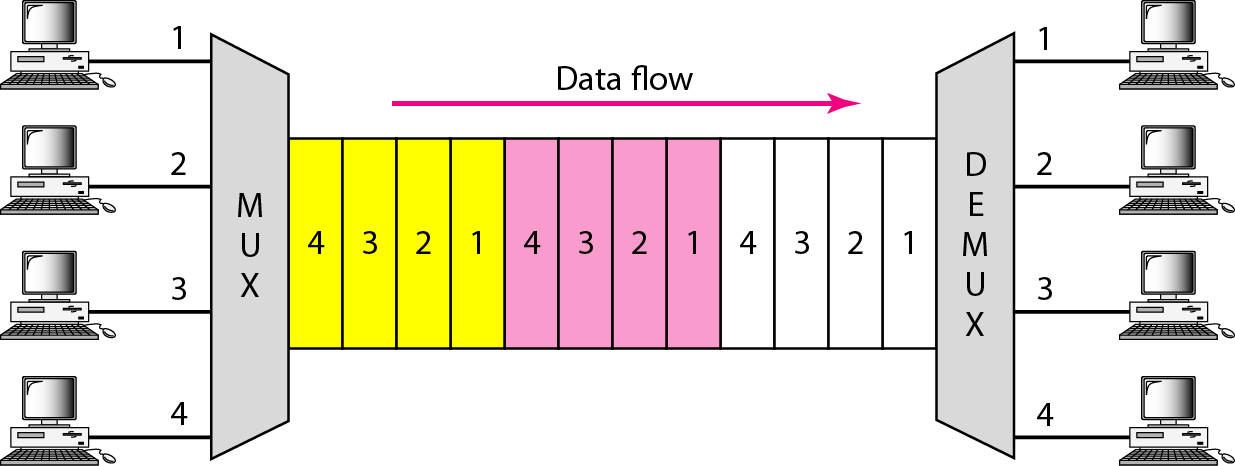


Although WDM technology is very complex, the basic idea is very simple. We want to combine multiple light sources into one single light at the multiplexer and do the reverse at the demultiplexer. The combining and splitting of light sources are easily handled by a prism. Recall from basic physics that a prism bends a beam oflight based on the angle ofincidence and the frequency. Using this technique, a multiplexer can be made to combine several input beams of light, each containing a narrow band of frequencies, into one output beam of a wider band of frequencies. A demultiplexer can also be made to reverse the process.

One application of WDM is the SONET network in which multiple optical fiber lines are multiplexed and demultiplexed. We discuss SONET in Chapter 17. A new method, called dense WDM (DWDM), can multiplex a very large number of channels by spacing channels very close to one another. It achieves even greater efficiency.

**Synchronous Time-Division Multiplexing**

Time-division multiplexing (TDM) is a digital process that allows several connections to share the high bandwidth of a line Instead ofsharing a portion ofthe bandwidth as in FDM, time is shared. Each connection occupies a portion oftime in the link. Figure 6.12 gives a conceptual view of TDM. Note that the same link is used as in FDM; here, however, the link is shown sectioned by time rather than by frequency. In the figure, portions ofsignals 1,2,3, and 4 occupy the link sequentially.



This means that all the data in a message from source 1 always go to one specific destination, be it 1, 2, 3, or 4. The delivery is fixed and unvarying, unlike switching. We also need to remember thatTDM is, in principle, a digital multiplexing technique. Digital data from different sources are combined into one timeshared link. However, this does not mean that the sources cannot produce analog data; analog data can be sampled, changed to digital data, and then multiplexed by using TDM.

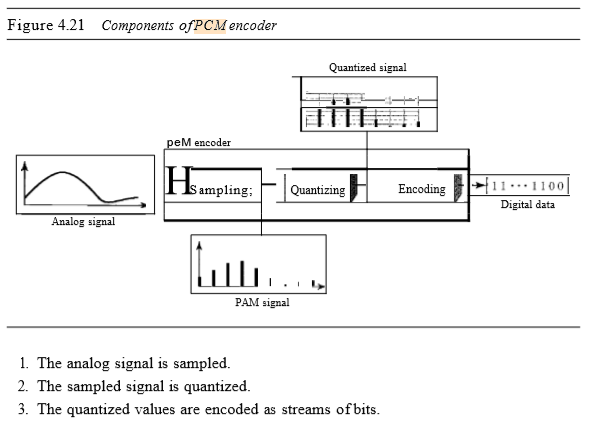
TDM is a digital multiplexing technique for combining several low-rate channels into one high-rate one.We can divide TDM into two different schemes: synchronous and statistical. We first discuss synchronous TDM and then show how statistical TDM differs. In synchronous TDM, each input connection has an allotment in the output even ifit is not sending data.

**ANALOG-TO-DIGITAL CONVERSION:**

1. **Pulse Code Modulation (PCM)**

The most common technique to change an analog signal to digital data (digitization) is called pulse code modulation (PCM). A PCM encoder has three processes:

* Sampling
* Quantizing
* Encoding



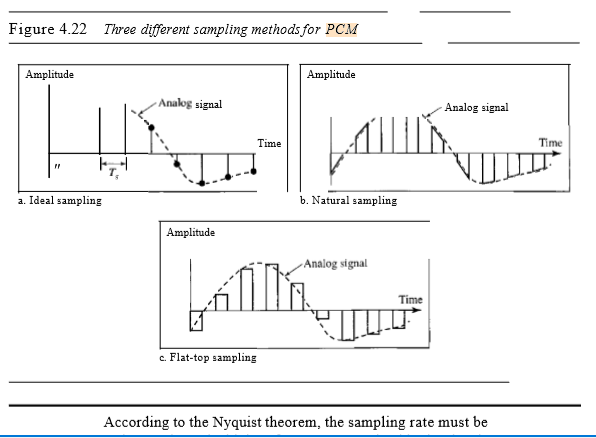
**Sampling:**

The first step in PCM is sampling. The analog signal is sampled every Ts s, where Ts is the sample interval or period. The inverse of the sampling interval is called the sampling rate or sampling frequency and denoted byis, whereis = IITs ' There are three sampling methods-ideal, natural, and flat-top.

In ideal sampling, pulses from the analog signal are sampled. This is an ideal sampling method and cannot be easily implemented. In natural sampling, a high-speed switch is turned on for only the small period of time when the sampling occurs. The result is a sequence of samples that retains the shape of the analog signal. The most common sampling method, called sample and hold, however, creates flat-top samples by using a circuit.

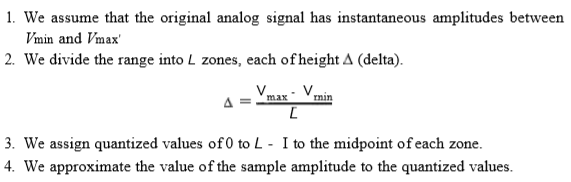
The sampling process is sometimes referred to as pulse amplitude modulation (PAM). We need to remember, however, that the result is still an analog signal with nonintegral values.

**According to the Nyquist theorem, to reproduce the original analog signal, one necessary condition is that the sampling rate be at least twice the highest frequency in the original signal.**



**Quantization**

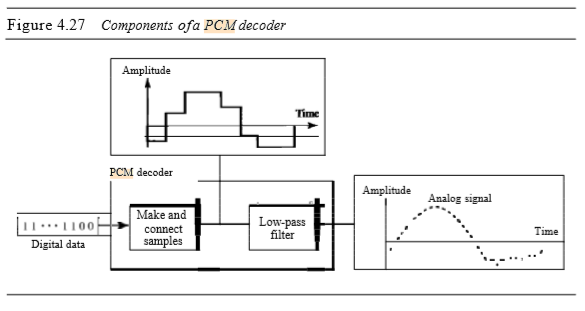
The result of sampling is a series of pulses with amplitude values between the maximum and minimum amplitudes of the signal. The set ofamplitudes can be infinite with nonintegral values between the two limits. These values cannot be used in the encoding process. The following are the steps in quantization:



As a simple example, assume that we have a sampled signal and the sample amplitudes are between -20 and +20 V. We decide to have eight levels (L = 8). This means that ~ =5 V.

**Encoding**

The last step in PCM is encoding. After each sample is quantized and the number of bits per sample is decided, each sample can be changed to an llb-bit code word. In Figure 4.26 the encoded words are shown in the last row. A quantization code of 2 is encoded as 010; 5 is encoded as 101; and so on. Note that the number of bits for each sample is determined from the number of quantization levels. If the number of quantization levels is L, the number of bits is llb =log2 L. In our example L is 8 and llb is therefore 3. The bit rate can be found from the formula.



**PCM decoder**

The decoder first uses circuitry to convert the code words into a pulse that holds the amplitude until the next pulse. After the staircase signal is completed, it is passed through a low-pass filter tosmooth the staircase signal into an analog signal. The filter has the same cutoff frequency as the original signal at the sender. If the signal has been sampled at (or greater than) the Nyquist sampling rate and if there are enough quantization levels, the original signal will be recreated. Note that the maximum and minimum values of the original signal can be achieved by using amplification. Figure 4.27 shows the simplified process.

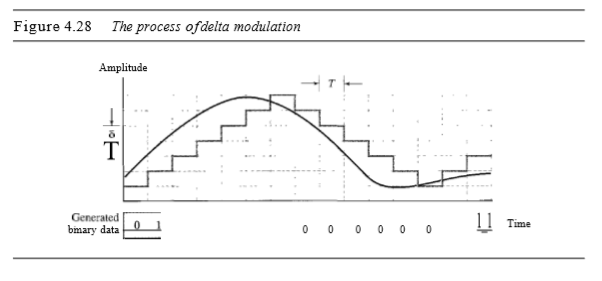
**VIRTUAL LANs**

A virtual local area network (VLAN) can be defined as a local area network configured by software, not by physical wiring.

Following figure shows a switched LAN in an engineering firm in which 10 stations are grouped into three LANs that are connected by a switch. The first four engineers work together as the first group, the next three engineers work together as the second group, and the last three engineers work together as the third group. The LAN is configured to allow this arrangement.

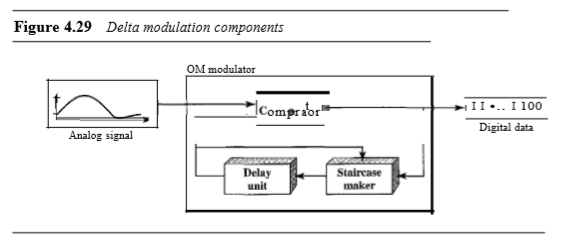
1. **Delta Modulation (DM):**

PCM is a very complex technique. Other techniques have been developed to reduce the complexity of PCM. The simplest is delta modulation. PCM finds the value of the signal amplitude for each sample; DM finds the change from the previous sample. Figure 4.28 shows the process. Note that there are no code words here; bits are sent one after another.



**Modulator:**

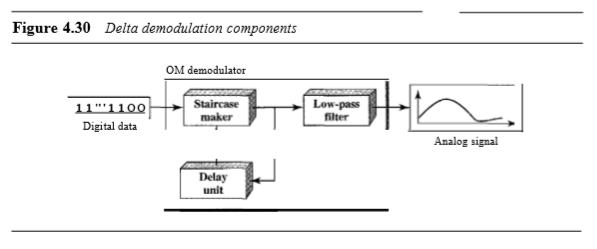
The modulator is used at the sender site to create a stream ofbits from an analog signal. The process records the small positive or negative changes, called delta O. Ifthe delta is positive, the process records a I; ifit is negative, the process records a O. However, the process needs a base against which the analog signal is compared. The modulator builds a second signal that resembles a staircase. Finding the change is then reduced to comparing the input signal with the gradually made staircase signal. Figure 4.29 shows a diagram of the process.



The modulator, at each sampling interval, compares the value ofthe analog signal with the last value ofthe staircase signal. Ifthe amplitude ofthe analog signal is larger, the next bit in the digital data is 1; otherwise, it is O. The output ofthe comparator, however, also makes the staircase itself. If the next bit is I, the staircase maker moves the last point ofthe staircase signal 0 up; it the next bit is 0, it moves it 0 down. Note that we need a delay unit to hold the staircase function for a period between two comparisons.

**Demodulator:**

The demodulator takes the digital data and, using the staircase maker and the delay unit, creates the analog signal. The created analog signal, however, needs to pass through a low-pass filter for smoothing. Figure 4.30 shows the schematic diagram.



A better performance can be achieved if the value of 0 is not fixed. In adaptive delta modulation, the value of 0 changes according to the amplitude ofthe analog signal.

**DATA LINK LAYER (DLL)**

**Data linkLayer** organizes bits into frames and provides hop-to-hop (node-to-node) delivery.

**Functions (Duties) of Data Link Layer**

* **Framing:** The data link layer divides the stream of bits received from the network layer into manageable data units called **frames*.***
* **Physical addressing:** The physical address, also known as the link address, is the address of a node as defined by its LAN (host address) or WAN (Router address). It is included in the frame used by the data link layer. It is the lowest-level address. For example, Ethernet uses a 6-byte physical address that is imprinted on the network interface card (NIC) of host or router.
* **Flow control:** If the rate at which the data is absorbed by the receiver is less than the rate produced at the sender, the data link layer imposes a flow control mechanism to prevent overwhelming the receiver.
* **Error control:** The data link layer adds reliability to the physical layer by adding mechanisms to detect and retransmit damaged or lost frames. It also uses a mechanism to recognize duplicate frames. Error control is normally achieved through a trailer added to the end of the frame.
* **Access control:** When two or more devices are connected to the same link, data link layer protocols are necessary to determine which device has control over the link at any given time.

**Services provided by DLL**

Three **services** commonly offered by data link layer are:

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

**Unacknowledged connectionless service** consists of having the source machine send independent frames to the destination machine without having the destination machine acknowledge them. 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 that 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. Most LANs use unacknowledged connectionless service in the data link layer.

**Acknowledged connectionless service** is a reliable service in which no logical connections are used but each frame sent is acknowledged. In this way, the sender knows whether a frame has arrived correctly. If it has not arrived within a specified time interval, it can be sent again. Thus reliability is achieved through acknowledgement. This type of service is used in the datagram approach to packet switching. This service is useful over unreliable channels, such as wireless systems. The Internet has chosen this type of service at the network layer.

Providing acknowledgement in the data link layer is just an optimization, never a requirement. The network layer can always send a packet and wait for it to be acknowledged. If the acknowledgement is not forthcoming before the time expires, the sender can just send the entire message again. The trouble with this strategy is that frames usually have a maximum length imposed by the hardware and network layer packets do not. If the average packet is broken up into, say, 10 frames, and 20 percent of all frames are lost, it may take a very long time for the packet to get through. If individual frames are acknowledged and retransmitted, entire packet gets through much faster. On reliable channels, such as fiber, the overhead of a heavyweight data link protocol may be unnecessary, but on wireless channels, with their inherent unreliability, it is well worth the cost.

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 they are sent over the same path in sequential order. 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. With connectionless service, in contrast, it is conceivable that a lost acknowledgement causes a packet to be sent several times and thus received several times. Connection-oriented service, in contrast, provides the network layer processes with the equivalent of a reliable bit stream.

When a connection-oriented service is used, transfers go through three distinct phases. In the first phase, the 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. In the second phase, one or more frames are actually transmitted. In the third or final phase, the connection is released, freeing up the variables, buffers, and other resources used to maintain the connection. This type of service is used in a virtual circuit approach to packet switching such as to Frame Relay and ATM.

**Framing**

The data link layer packs bits into frames, so that each frame is distinguishable from another. **Framing** in the data link layer separates a message from one source to a destination, or from other messages to other destinations, by adding a sender address and a destination address. The destination address defines where the packet is to go; the sender address helps the recipient acknowledge the receipt.

Although the whole message could be packed in one frame that is not normally done. One reason is that a frame can be very large, making flow and error control very inefficient. When a message is carried in one very large frame, even a single-bit error would require the retransmission of the whole message. When a message is divided into smaller frames, a single-bit error affects only that small frame

**Fixed-Size Framing**

Frames can be of fixed or variable size. In fixed-size framing, there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter. An example of this type of framing is the ATM wide-area network, which uses frames of fixed size called cells.

**Variable-Size Framing**

In variable-size framing which is prevalent in LANs, we need a way to define the end of the frame and the beginning of the next. Historically, two approaches were used for this purpose: a character-oriented approach and a bit-oriented approach. Following are four methods of framing.

1. Character count.
2. Flag bytes with byte stuffing.
3. Starting and ending flags, with bit stuffing.
4. Physical layer coding violation.

The **character count method** uses a field in the header to specify the number of characters in the frame. When the data link layer at the destination sees the character count, it knows how many characters follow and hence where the end of the frame is. This technique is shown in Fig. 3-4(a) for four frames of sizes 5, 5, 8, and 8 characters, respectively.

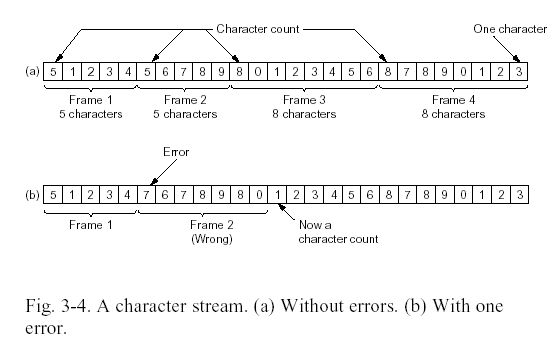


Figure 3.1: A character stream (a) without errors (b) with one error

The trouble with this algorithm is that the count can be garbled by transmission error. For example, if the character count of 5 in the second frame of Fig. 3-1(b) becomes a seven, the destination will get out of synchronization and will be unable to locate the start of next frame. Even if the checksum is incorrect so the destination knows that the frame is bad, it still has no way of telling where the next frame starts. Sending a frame back to the source asking for a retransmission does not help either, since the destination does not know how many characters to skip over to get to the start of the retransmission. For this reason, the character count method is rarely used any more.

The second framing method, i. e. **flag byte with byte stuffing,** gets around the problem of resynchronization after an error by having each frame start and end with special bytes called the **flag bytes**. The protocol uses the same flag byte as both the starting and ending delimiter of the frame which is shown in Fig. 3.2(a) as FLAG. In this way, if the receiver ever loses synchronization, it can just search for the flag byte to find the end of the current frame. Two consecutive flag bytes indicate the end of one frame and start of the next frame.

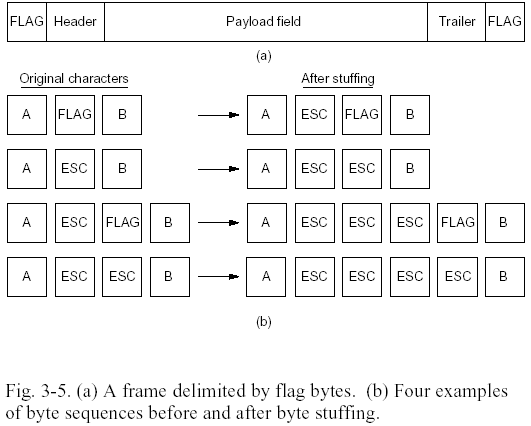


Fig 3.2: (a) A frame delimited by flag bytes (b) Four examples of byte sequences before and after byte stuffing.

A serious problem occurs with this method when binary data, such as object programs or floating-point numbers, are being transmitted. It may easily happen that the flag byte bit pattern occurs in the data. This situation will usually 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. The data link layer on the receiving end removes the escape byte before the data are given to the network layer. This technique is called **byte stuffing** or **character stuffing.** Thus, a framing flag byte can be distinguished from one in the data by the absence or presence of an escape byte before it.

If an escape byte occurs in the middle of the data, it too is stuffed with an escape byte. Thus, any single escape byte is part of an escape sequence, whereas a double one indicates that a single escape occurred naturally in the data. Some examples are shown in Fig. 3.2(b). In all cases, the byte sequence delivered after destuffing is exactly the same as the original byte sequence.

The major **disadvantage** of using this framing method is that it is closely tied to the use of 8-bit characters. Not all character codes use 8-bit characters (UNICODE uses 16-bit characters).

In **bit stuffing** technique, each frame begins and ends with a special bit pattern, 01111110 (in fact, a flag byte). Whenever 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 analogous to byte stuffing, in which an escape byte is stuffed into the outgoing character stream before a flag byte in the data.

When the receiver sees five consecutive incoming 1 bits, followed by a zero 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. Fig.3-6 gives an example of bit stuffing.

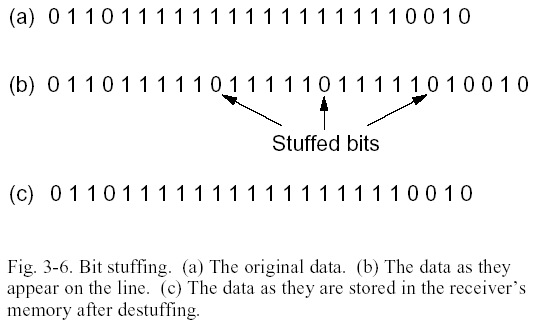


Figure 3.3. Bit stuffing. (a) The original data. (b) The data as they appear on the line. (c) The data as they are stored in the receiver’s memory after destuffing.

With bit stuffing, the boundary between two frames can be unambiguously reorganized by the flag pattern. Thus, if the receiver loses track of where it is, all it has to do is scan the input for flag sequences, since they can only occur at frame boundaries and never within the data.

The last method of framing (**physical layer coding violation scheme**) is only applicable to networks in which the encoding on the physical medium contains some redundancy. For example, some LANs encode 1 bit of data by using two physical bits (signal levels). Normally, a 1 bit is a high-low pair and a 0 bit is a low-high pair (Manchester encoding). The scheme means that every data bit has a transition in the middle, making it easy for the receiver to locate the bit boundaries. The combination high-high and low-low are not used for data but are used for delimiting frames in some protocols.

Many data link protocols use a combination of character count with one of the other methods for extra safety. When a frame arrives, the count field is used to locate the end of the frame. Only if the appropriate delimiter is present at that position and the checksum is correct is the frame accepted as valid. Otherwise, the input stream is scanned for the next delimiter.

**Exercise Questions:**

Q1. The following character encoding is used in a data link protocol:

A: 01000111; B: 11100011; FLAG: 01111110; ESC: 11100000

Show the bit sequence transmitted (in binary) for the four-character frame: A B ESC FLAG when each of the following framing methods are use:

1. Character count.
2. Flag bytes with byte stuffing.
3. Starting and ending flag bytes, with bit stuffing.

Ans: (a) 00000100 01000111 11100011 11100000 01111110

(b) 01111110 01000111 11100011 11100000 11100000 11100000 01111110 01111110

(c) 01111110 01000111 110100011 111000000 011111010 01111110

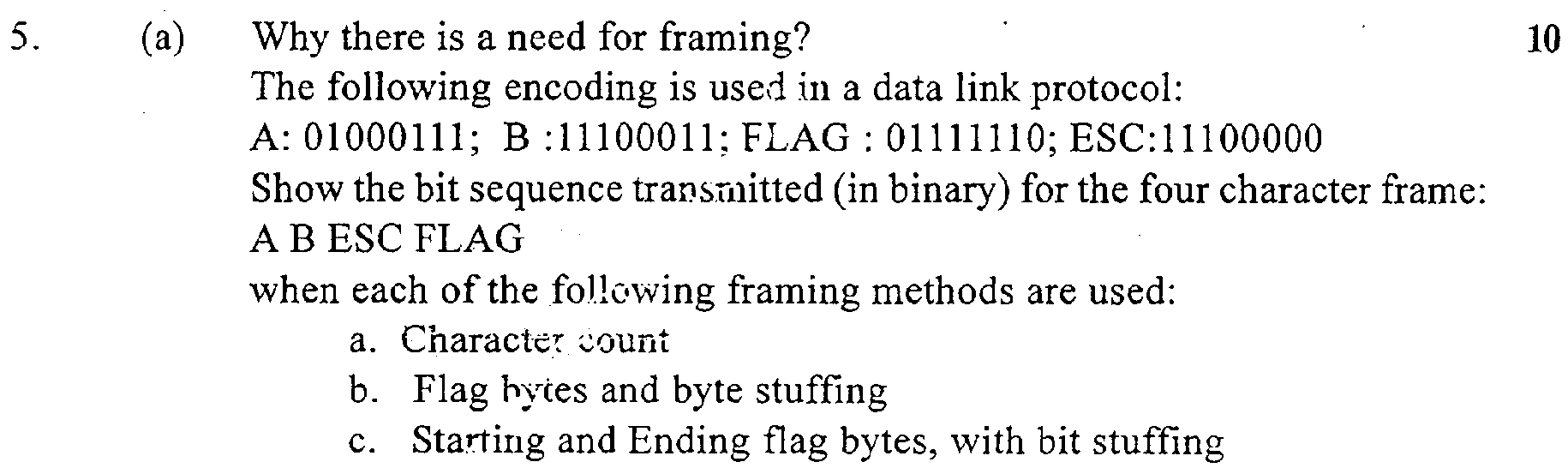
Q2. The following data fragment occurs in the middle of a data stream for which the byte stuffing

algorithm described is used: A B ESC C ESC FLAG FLAG D. What is the output after

stuffing?

Ans: A B ESC ESC C ESC ESCESC FLAG ESC FLAG D.

Q3.



4. A bit string, 0111101111101111110, needs to be transmitted at the data link layer. What is the string actually transmitted after bit stuffing?

Ans: 011110111110011111010.

**DATA LINK CONTROL AND PROTOCOLS**

The most important responsibilities of the data link layer are **flow control** and **error control**. Collectively, these functions are known as **data link control.**

**Flow Control**

**Flow control** refers to a set of procedures to restrict the amount of data that the sender can send before waiting for acknowledgement from the receiver. The flow of data must not be allowed to overwhelm the receiver. Any receiving device has a limited speed at which it can process incoming data and a limited amount of memory (called a **buffer**) for storing incoming data until they are processed and passed to network layer. If the buffer begins to fill up, the receiving device must be able to inform the transmitting device to transmit fewer frames or halt the transmission temporarily until it is once again able to receive.

**Error Control**

**Error control** is both error detection and error correction. In the data link layer, error control refers primarily to methods of error detection and retransmission. It allows the receiver to inform the sender of any frames lost or damaged in transmission and coordinates the retransmission of those frames by the sender. Then, if an error is detected in an exchange, specified frames are retransmitted. This process is called **Automatic Repeat Request** (**ARR**).

There are three common flow and error control protocols.

* **Stop-And-Wait ARQ** (also called **One-Bit-Sliding Window protocol**)
* **Go-Back-N ARQ**
* **Selective Repeat ARQ**

**STOP-AND-WAIT ARQ** (**ONE-BIT-SLIDING WINDOW PROTOCOL**)

**Normal Operation**

In a normal transmission, the sender sends frame 0 and waits to receive ACK1. When ACK 1 is received, it sends frame 1 and then waits to receive ACK 0, and so on. The ACK must be received before the timer set for each frame expires. Figure 3.4 shows successful frame transmission.

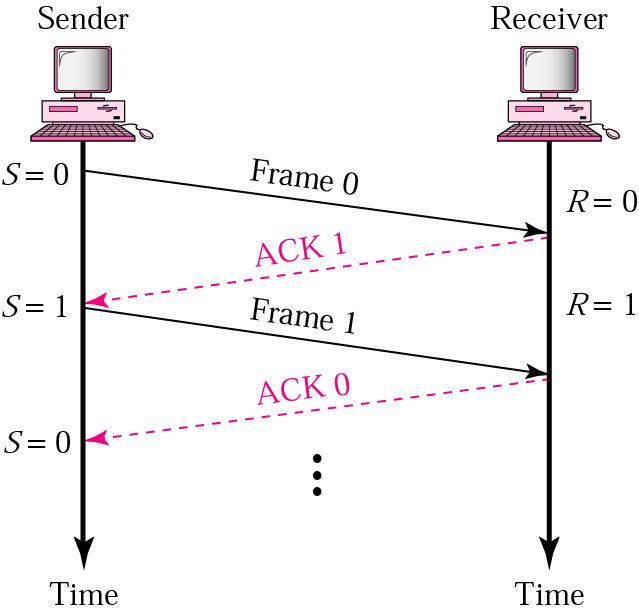


Figure 3.4. Normal operation

**Lost or Damaged Frame**

When the receiver receives a damaged frame it discards it and remains silent. It also remains silent about a lost frame and keeps its value of variable R. For example, in Figure 3.5, the sender transmits frame 1, but it is lost. The receiver does nothing, retaining the value of R (1). After the timer at the sender site expires, another copy of frame 1 is sent.

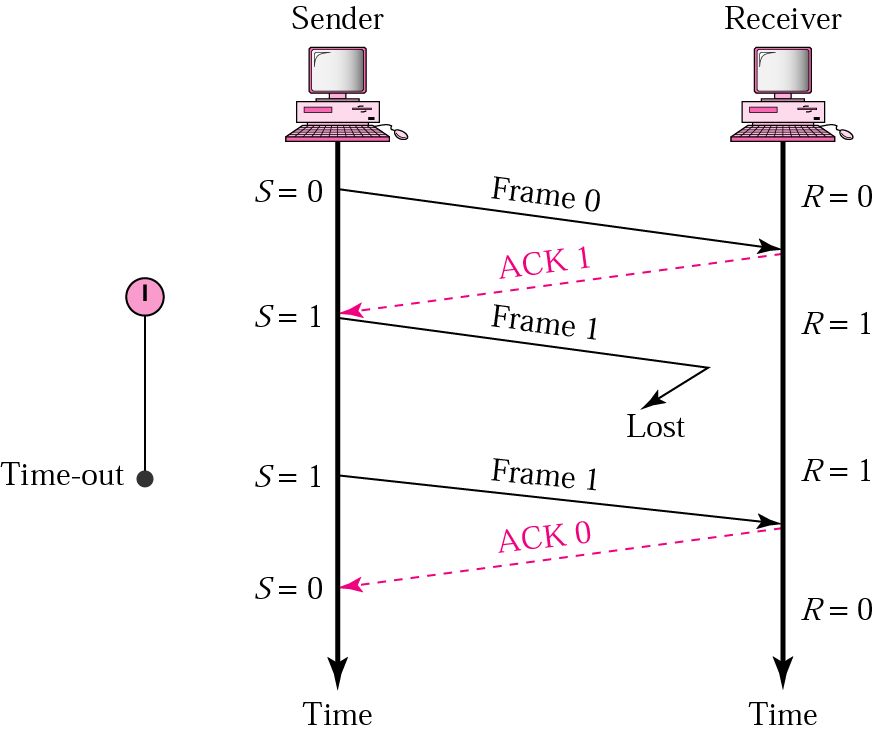


Figure 3.5. frame Lost

**Lost Acknowledgement**

A lost or damaged acknowledgement is handled in the same way by the sender; if the sender receives a damaged acknowledgement, it discards it. Figure 3.6 shows a lost ACK 0. The waiting sender does not know if frame 1 has been received. When the timer for frame 1 expires, the sender retransmits frame 1. But the receiver is expecting to receive frame 0, therefore, it silently discards the second copy of frame 1.

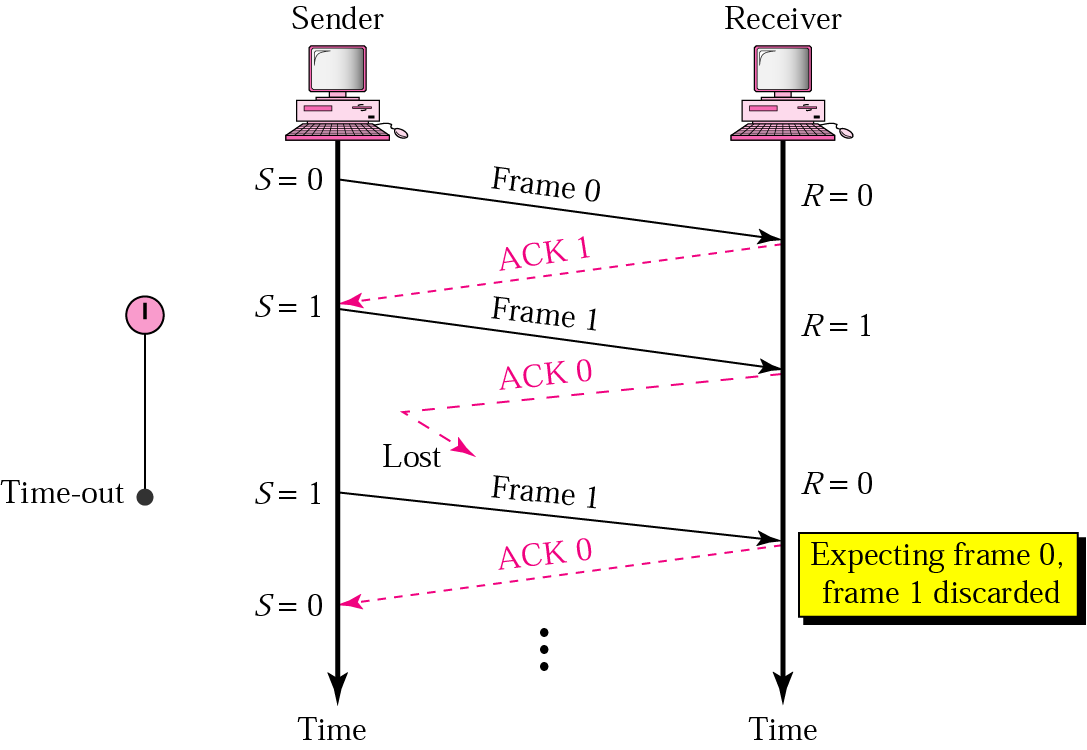


Figure 3.6. Lost ACK frame

This example shows the need to number frames. If the frames were not numbered, the receiver, thinking that frame 1 is a new frame (not a duplicate), keeps it. **Thus, numbering frames prevents the retaining of duplicate frames**.

**Delayed Acknowledgement**

An acknowledgement can be delayed at the receiver or by some problem with the link. Figure 3.7 shows the delay of ACK 1; it is received after the timer for frame 0 has already expired. The sender has already retransmitted a copy of frame 0. However, the value of R at the receiver site is still 1, which means that the receiver expects to see frame 1. The receiver, therefore, discards the duplicate frame 0.

The sender has now received two ACKs, one that was delayed and one that was sent after the duplicate frame 0 arrived. The second ACK 1 is discarded.

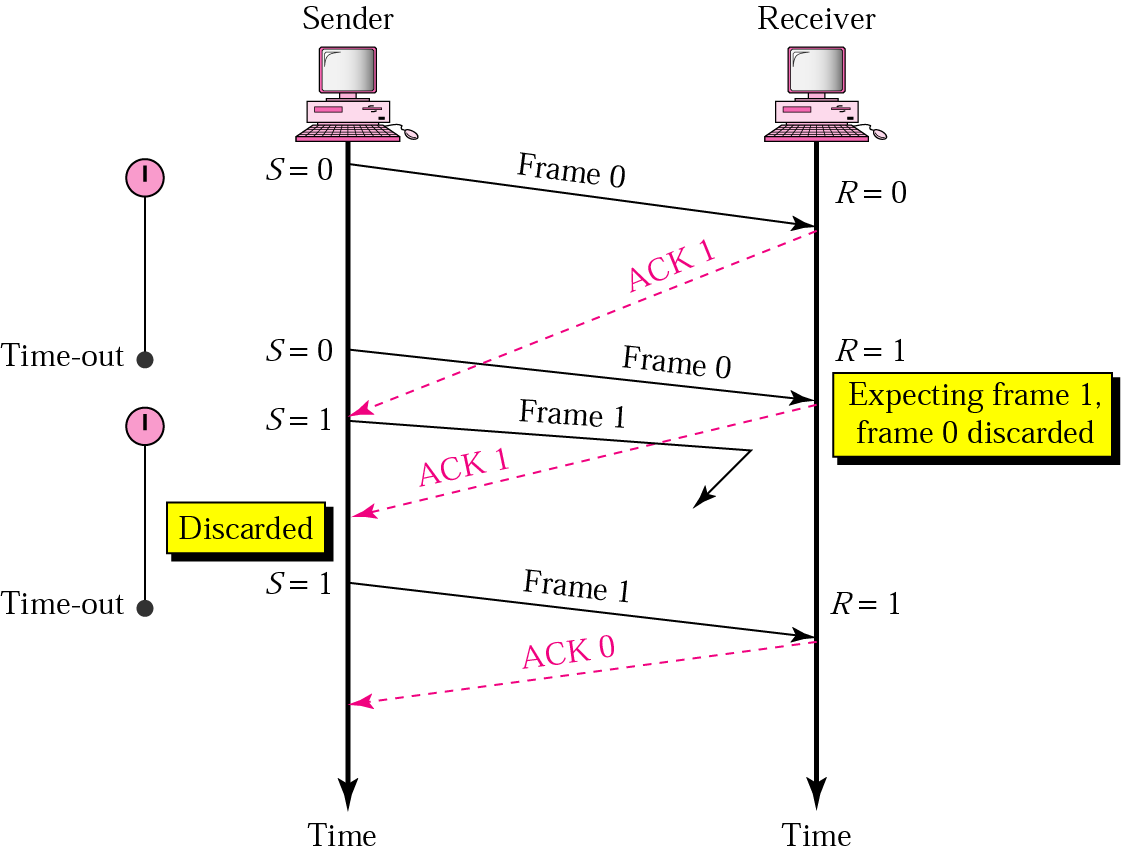


Figure 3.7. Delayed ACK

**SLIDING - WINDOW PROTOCOLS**

In Stop-and-Wait ARQ, at any point in time, there is only one outstanding frame that is sent and waiting for acknowledgement. This is not a good use of the transmission medium. To improve the efficiency, multiple frames should be in transition while waiting for acknowledgement. In other words, we need to let more than one frame be outstanding. In these protocols, we can send up to W frames before worrying about acknowledgements; we keep a copy of these frames until the acknowledgements arrive.

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 before receiving an acknowledgement. These frames are said to fall within 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 or upper limits or even have the same size. In some protocols they are fixed in size, but in others (such as TCP protocol) they can grow or shrink over the course of time as frames are sent and received.

The sequence numbers within the sender’s window represent frames that have been sent or can be sent but are as yet not acknowledged. Whenever a new packet arrives from the network layer, it is given the next higher 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 window continuously maintains a list of unacknowledged frames.

This procedure requires additional features to be added to Stop-and-Wait ARQ. Two protocols use this concept:

* **Go-Back-N ARQ and**
* **Selective Repeat ARQ**

**Go –Back-N ARQ**

**Sender Sliding Window**

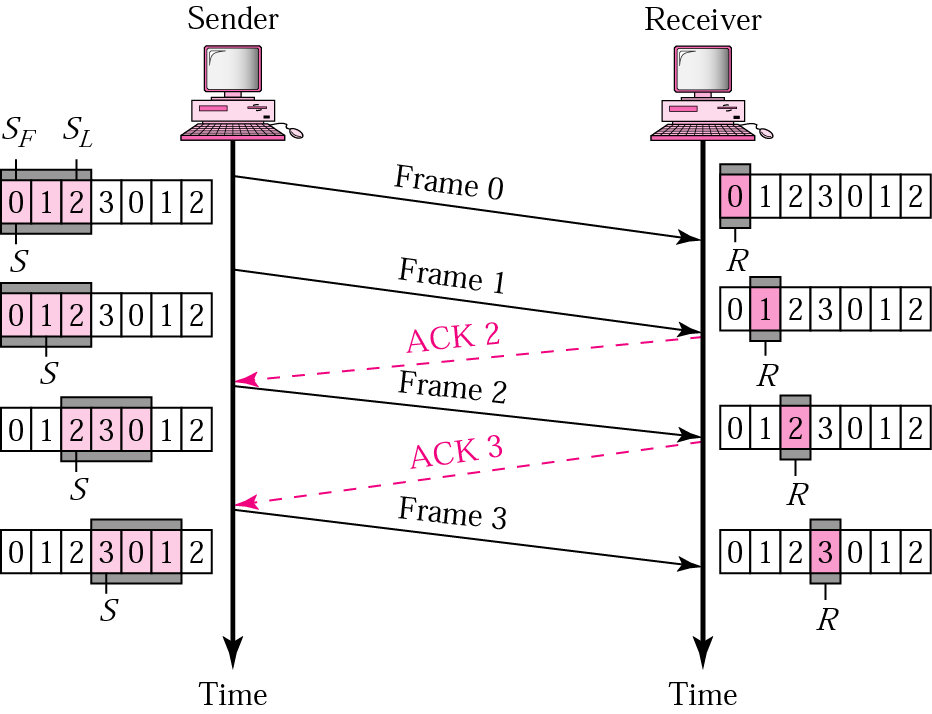
At the sender site, to hold the outstanding frames until they are acknowledged, all outstanding frames are stored in a buffer called **window**. The frames to the left of the window are those that have already been acknowledged and can be purged; those to the right of the window can not be sent until the window slides over them. The size of the window is at most 2*m*-1. The window **slides** to include new unsent frames when the correct acknowledgements are received that is why it is called a **sliding window.**

**Receiver Sliding Window**: The size of the receiver window is 1. The window slides by one slot whenever a correct frame has arrived.

**Operation**

**Normal Operation**

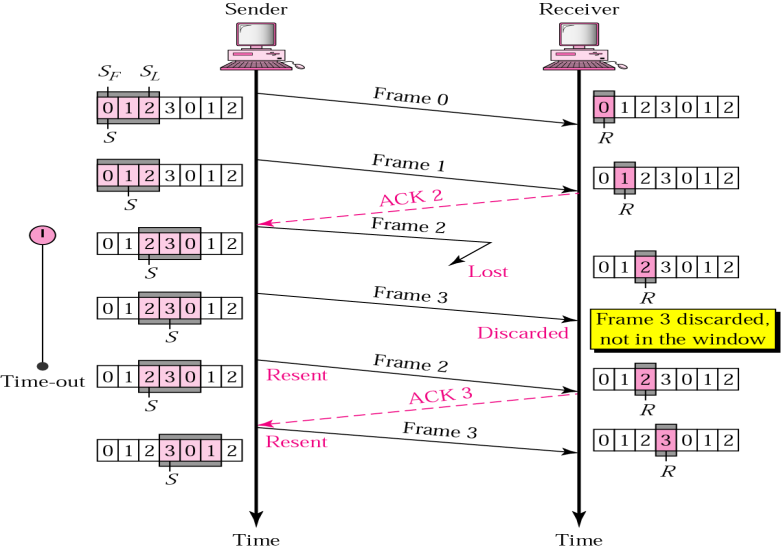
Fig. 3.8 shows a normal operation of this mechanism. The sender keeps track of outstanding frames and updates the variables and windows as the acknowledgements arrive. When an acknowledgement comes in, the lower edge is advanced by one. In this way the window continuously maintains a list of unacknowledged frames.



**Fig. 3.8** Go-Back-N ARQ, **normal operation**

**Damaged or Lost Frames**

Fig. 3.9 shows that frame 2 is lost. When the receiver receives frame 3, it is discarded because the receiver is expecting frame 2, not frame 3. After the timer for frame 2 expires at the sender site, the sender goes back to frame 2 and sends all frames starting from frame 2, i.e., frames 2 and 3.



**Fig. 3.9** Go-Back-N ARQ, **lost frame**

**Damaged or Lost Acknowledgements**

If an acknowledgement is damaged or lost, we ca have two situations. If the next acknowledgement arrives before the expiration of any timer, there is no need for retransmission of frames because acknowledgements are **cumulative** in this protocol. ACK 4 means ACK 1 to ACK 4. So if ACK 1, ACK 2, and ACK 3 are lost, ACK 4 covers them. However, if the next ACK arrives after the time-out, the frame and all the frames after that are resent. Receiver never resends an ACK.

**Delayed Acknowledgement**

A delayed acknowledgement also triggers the resending of frames after time-out.

**Selective Repeat ARQ:**

Go back-N ARQ simplifies the process at the receiver site. The receiver keeps track of only one variable, and there is no need to buffer out-of-order frames; they are simply discarded. However, this protocol is very inefficient for a noisy link. In a noisy channel a frame has a higher probability of damage, which means the resending of multiple frames. The resending uses up the bandwidth and slows down the transmission. For noisy channels, there is another protocol that does not resend N frames when just one frame is damaged; only the damaged frame is resent. This protocol is called **selective repeat**. When it is used, a bad frame that is received is discarded, but good frames received after it are buffered. When the sender times out, only the oldest unacknowledged frame is retransmitted. If that frame arrives correctly, the receiver can deliver to the network layer, in sequence, all the frames it has buffered. Selective repeat is often combined with having the receiver send a **negative acknowledgement (NAK)** when it detects any error, for example, when it receives a checksum error or a frame out of sequence. NAKs stimulate retransmission before the corresponding timer expires and this improves performance.

**Sender and receiver window size in selective repeat:**

The size of the sender and receiver windows must be at most one-half of the sequence number range (2m). For example, for a 2 bit sequence number, the size of the window should be at most 22/2 = 2.

**Operation**

In figure 3.11, frame 0 and 1 are accepted when received because they are in the range specified by the receiver window. When frame 3 is received, it is also accepted for the same reason. However, the receiver sends a NAK 2 to show that frame 2 has not been received. When the sender receives the NAK 2, it resends only frame 2, which is then accepted because it is in the range of the window. Similar is the case with lost and delayed ACKs and NAKs. Sender also sets a timer for each frame sent.

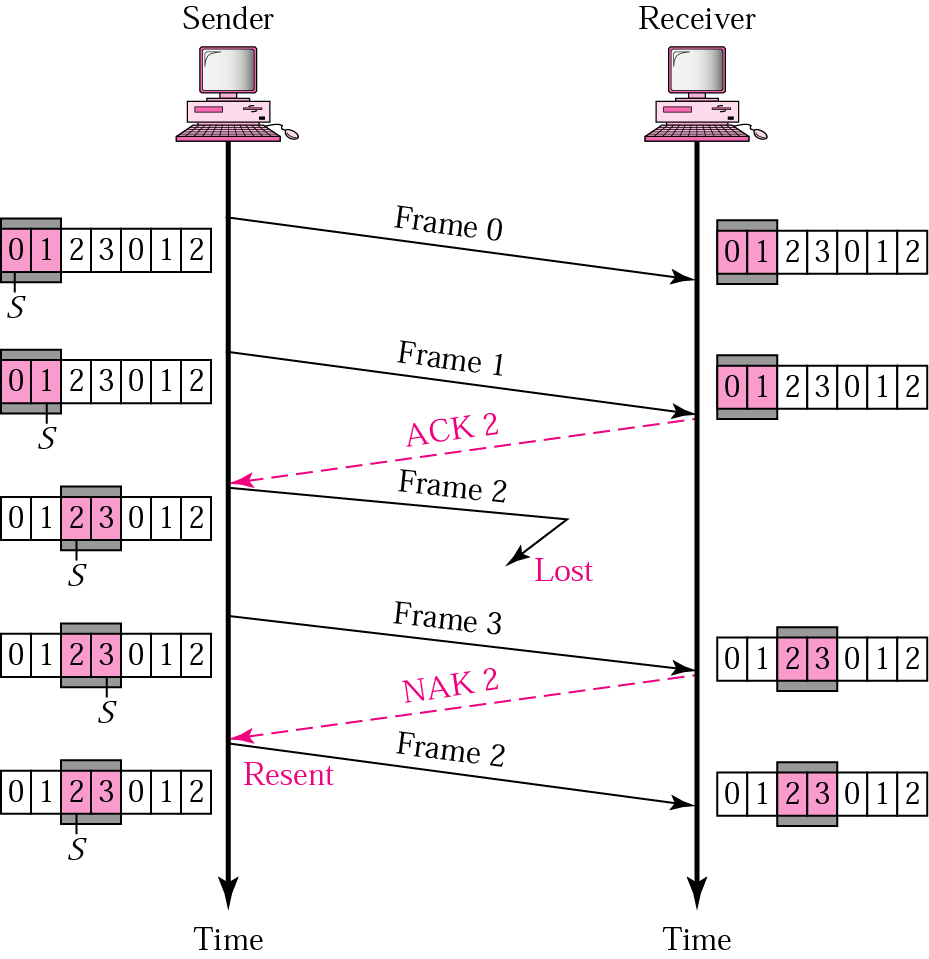


Fig 3.11 Selective Repeat ARQ, lost frame

**Q. Explain with suitable example CRC algorithm for computing checksum. [10 marks][M13][D12]**

**Q. Explain the error detection and error correction algorithms.**

**Error Detecting Code**

The **CRC** (**Cyclic Redundancy Check**), also known as a **polynomial code** is an error detecting code used at data link layer. Polynomial codes are based upon treating bit strings as representations of polynomials with coefficients of 0 and 1 only. For example, 110001 has 6 bits and thus represents a six-term polynomial with coefficients 1, 1, 0, 0, 0, and 1: 1*x*5+1x4+0x3+0x2 + 0*x* 1 + 1*x* 0= *x*5+ x4 + 1.

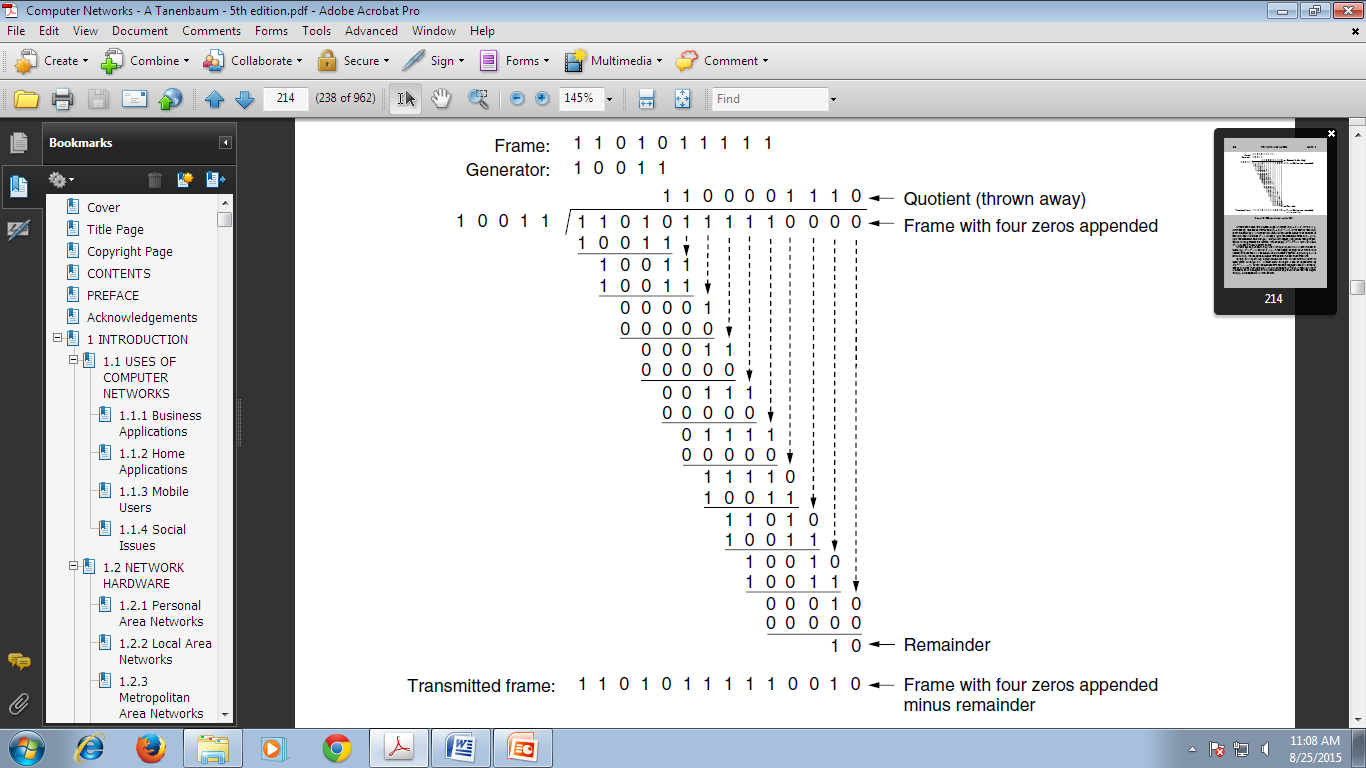
When the polynomial code method is employed, the sender and receiver must agree upon a **generator polynomial**, *G*(*x*), in advance. Both the high- and low order bits of the generator must be 1. To compute the CRC for some frame with *m* bits corresponding to the polynomial *M*(*x*), the frame must be longer than the generator polynomial. The idea is to append a CRC to the end of the frame in such a way that the polynomial represented by the checksummed frame is divisible by *G*(*x*)*.* When the receiver gets the checksummed frame, it tries dividing it by *G*(*x*)*.* If there is a remainder, there has been a transmission error. The algorithm for computing the CRC is as follows:

1. Let *r* be the degree of *G*(*x*)*.* Append *r* zero bits to the low-order end of the frame so it now contains *m* + *r* bits and corresponds to the polynomial *x* r*M*(*x*)*.*

2. Divide the bit string corresponding to *G*(*x*) into the bit string corresponding to *x*r*M*(*x*), using modulo 2 division.

3. Subtract the remainder (which is always *r* or fewer bits) from the bit string corresponding to *xrM*(*x*) using modulo 2 subtraction. The result is the checksummed frame to be transmitted. Call its polynomial *T*(*x*)*.*

Following figure illustrates the calculation for a frame 1101011111 using the generator *G*(*x*) = *x*4 + *x* + 1*.*



**Figure:** Example calculation of the CRC.

**Error Correction Algorithm**

The use of error-correcting codes is often referred to as **FEC** (**Forward Error Correction**). **Hamming Code** is a block code whichadds redundancy to the information that is sent. A frame consists of *m* data (i.e., message) bits and *r* redundant (i.e. check) bits. Here *r*check bits are computed solely as a function of the *m* data bits with which they are associated. Let the total length of a block be n (i.e., n = m + r). This is described as an (n, m) code. An n-bit unit containing data and check bits is referred to as an n bit codeword. The code rate, or simply rate, is the fraction of the codeword that carries information that is not redundant, or m/n. The rates used in practice vary widely. They might be 1/2 for a noisy channel, in which case half of the received information is redundant, or close to 1 for a high-quality channel, with only a small number of check bits added to a large message.

The number of bit positions in which two codewords differ is called the **Hamming distance** (Hamming, 1950). In the case of two codewords 0001001 and 10110001— they differ by 3 bits. Thus their Hamming distance is 3. Its significance is that if two codewords are a Hamming distance *d* apart, then it can correct (d-1)/2 bits of error..

The key to the Hamming Code is the use of extra parity bits to allow the identification of a single error. The code word is created as follows:

1. All bit positions that are powers of two are marked as parity bits. (Positions 1, 2, 4, 8, 16, 32, 64, etc.)
2. All other bit positions are for the data to be encoded. (Positions 3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17, etc.)
3. Each parity bit calculates the parity for some of the bits in the code word. The position of the parity bit determines the sequence of bits that it alternately checks and skips.   
   Position 1: check 1 bit, skip 1 bit, check 1 bit, skip 1 bit, etc. (1,3,5,7,9,11,13,15,...)  
   Position 2: check 2 bits, skip 2 bits, check 2 bits, skip 2 bits, etc. (2,3,6,7,10,11,14,15,...)  
   Position 4: check 4 bits, skip 4 bits, check 4 bits, skip 4 bits, etc. (4,5,6,7,12,13,14,15,20,21,22,23,...)  
   Position 8: check 8 bits, skip 8 bits, check 8 bits, skip 8 bits, etc. (8-15,24-31,40-47,...)  
   Position 16: check 16 bits, skip 16 bits, check 16 bits, skip 16 bits, etc. (16-31,48-63,80-95,...)  
   Position 32: check 32 bits, skip 32 bits, check 32 bits, skip 32 bits, etc. (32-63,96-127,160-191,...)  
   etc.
4. Set a parity bit to 1 if the total number of ones in the positions it checks is odd. Set a parity bit to 0 if the total number of ones in the positions it checks is even.

Here is an example:

A byte of data: 10011010  
Create the data word, leaving spaces for the parity bits: \_ \_ 1 \_ 0 0 1 \_ 1 0 1 0  
Calculate the parity for each parity bit (a ? represents the bit position being set):

* Position 1 checks bits 1,3,5,7,9,11:  **?** \_ **1** \_ **0** 0 **1**\_ **1** 0 **1** 0. Even parity so set position 1 to a 0: **0** \_ **1** \_ **0** 0 **1**\_ **1** 0 **1** 0
* Position 2 checks bits 2,3,6,7,10,11:  
  0 **? 1**\_ 0 **0 1** \_ 1 **0 1** 0. Odd parity so set position 2 to a 1: 0 **1 1** \_ 0 **0 1** \_ 1 **0 1** 0
* Position 4 checks bits 4,5,6,7,12:  
  0 1 1 **? 0 0 1** \_ 1 0 1 **0**. Odd parity so set position 4 to a 1: 0 1 1 **1 0 0 1**\_ 1 0 1 **0**
* Position 8 checks bits 8,9,10,11,12:  
  0 1 1 1 0 0 1 **? 1 0 1 0**. Even parity so set position 8 to a 0: 0 1 1 1 0 0 1 **0 1 0 1 0**
* Code word: 011100101010.

If a receiver receives a wrong codeword, then the receiver could calculate which bit was wrong and correct it by verifying each check bit.

**HIGH-LEVEL DATA LINK CONTROL (HDLC)**

HDLC is a classical bit-oriented actual protocol designed to support both half-duplex and full-duplex communication over point-to-point and multipoint links. It supports ARQ mechanism.

**Basic characteristics**

HDLC defines three types of stations, two link configurations, and three transfer modes of operations.

The three **station types** are:

* **Primary station.** It has the responsibility for controlling the operation of the link. Frames issued by the primary are called *commands*.
* **Secondary station.** It operates under the control of the primary station. Frames issued by the secondary are called *responses*. The primary maintains a separate logical link with each secondary station on the line.
* **Combined station.** It combines the features of both primary and secondary. A combined station may issue both commands and responses.

The two **link configurations** are:

* **Unbalanced configuration.** It consists of one primary and one or more secondary stations and supports both full duplex and half duplex transmission.
* **Balanced configuration.** It consists of two combined stations on a point-to-point link and supports both full duplex and half duplex communication.

The three **transfer modes** are**:**

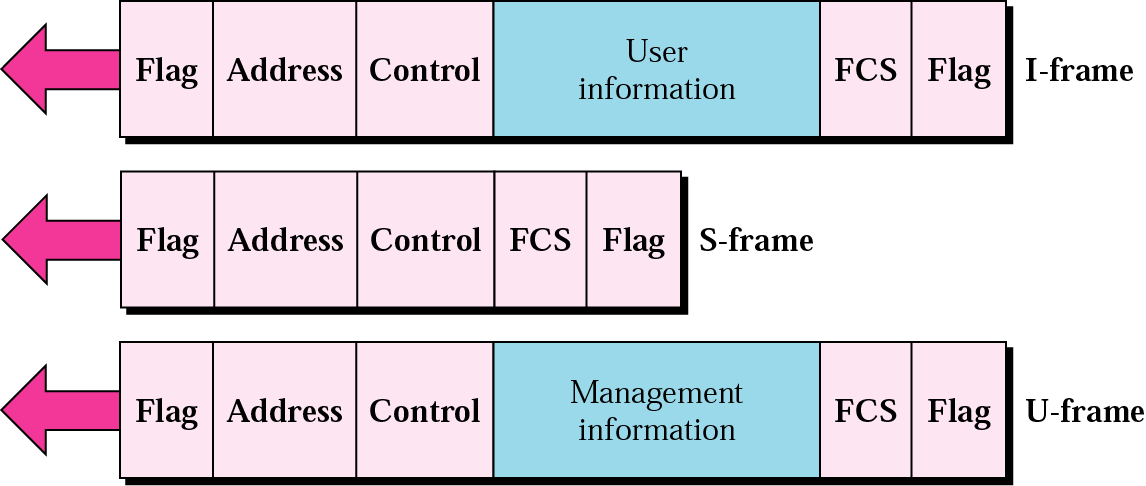
* **Normal Response mode (NRM):** In this mode the station configuration is unbalanced. We have one primary station and multiple secondary stations. The primary may initiate data transfer to a secondary, but secondary may only transmit data in response to a command from the primary.
* **Asynchronous Balanced Mode (ABM):** The configuration is balanced. The link is point-to-point, and each station can function as a primary or secondary. Either combined station may initiate transmission without receiving permission from the other combined station.
* **Asynchronous Response Mode (ARM):**It is used with an unbalanced configuration. The secondary may initiate transmission without explicit permission of the primary. The primary still retains responsibility for the line, including initialization, error recovery, and logical disconnection.

**HDLC Frame Format**

To provide the flexibility necessary to support all the options possible in modes and configurations, HDLC defines three types of frames based on the value of bits in the control field.

* Information frames (I-frames)
* Supervisory frames (S-frames) and
* Unnumbered frames (U-frames)

Each type of frame works as an envelop for the transmission of a different type of message. I-frames are used to transport user data and control information related to user data (piggybacking). S-frames are used only to transport control information. U-frames are reserved for system management. Information carried by U-frames is intended for managing the link itself. These are shown in Fig. 3-18.



**Fig. 3-18** HDLC frame formats

Function of each field is described below:

**Flag Field** (8 bits)**:** It is an 8-bit sequence with a bit pattern 01111110 that identifies both as beginning and end of a frame and serves as a synchronization pattern for the receiver.

**Address Field** (8 bits – extendable)**:** It contains the address of the secondary station. If a primary station creates a frame, it contains a ‘*to*’ address (destination address). If a secondary creates a frame, it contains a ‘*from’* address (originator address

**Control Field** (8 or 16 bits)**:** It is a 1- or 2-byte segment of the frame used for flow or error control. It is used for sequence numbers, acknowledgements, and other purposes. The interpretation of bits in this field is different for different frame types.

**Information Field** (variable length)**:** It contains the user’s data from the network layer or network management information. Its length can vary from one network to another but is always fixed within each network.

**FCS Field** (16 or 32 bits)**:** It is HDLC’s error detection field and contains cyclic redundancy code. It contains either a 2- or 4-byte ITU-T CRC.

**Point-to-Point Protocol (PPP)**

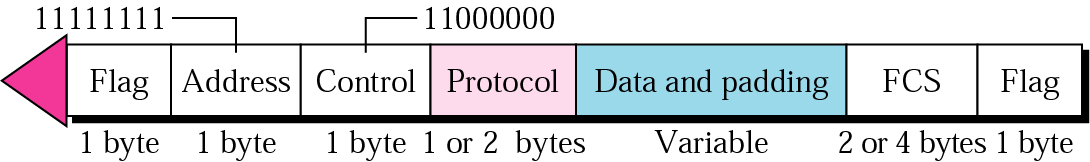
The PPP is an Internet standard protocol for transporting IP datagram over a point-to-point link.

PPP provides several services such as:

1. It defines the format of the frame to be exchanged between devices.
2. It defines how two devices can negotiate the establishment of the link and the exchange of data.
3. It defines how network layer data are encapsulated in the data link frame.
4. It defines how two devices can authenticate each other.

**PPP Frame Format:**

The PPP frame format was chosen to closely resemble the HDLC frame format. The major difference between PPP and HDLC is that PPP is character oriented rather than bit oriented. In particular, PPP uses byte stuffing on dial-up modem lines, so all frames are an integral number of bytes. It is not possible to send a frame consisting of 30.25 bytes, as it is with HDLC. Not only can PPP frames be sent over dial-up telephone lines but they can also be sent over SONET or true bit-oriented HDLC lines (e.g., for router-router connections). The PPP frame format is shown in fig. 3.23. The description of each field follows.



**Fig. 3-23** PPP frame

* **Flag Field: -** All PPP frames begin and end with the standard HDLC flag byte (01111110), which is byte stuffed if it occurs within the payload (data) field.
* **Address field**: Because PPP is used for a point-to-point connection, it uses the broadcast address of HDLC, 11111111, to avoid data link address in the protocol.
* **Control field**: The control field uses the format of U-frame in HDLC. The value is 11000000 to show that the frame does not contain any sequence numbers and that there is no flow or error control.
* **Protocol field:** The protocol field defines what is being carried in the data field: user data or other information. Codes are defined for LCP, NCP, IPX, Apple Talk and other protocols. Protocols starting with a 0 bit are network layer protocols such as IP,IPX, OSI CLNP, XNS. Those starting with a 1 bit are used to negotiate other protocols. These include LCP and a different NCP for each network layer protocol supported. The default size of the protocol field is 2 bytes, but it can be negotiated down to 1 byte using LCP.
* **Payload or data field:** Payload field carries either the user data or user control information**.** It is variable length up to some negotiated maximum. Its length is not negotiated using LCP during line set up; a default length of 1500 bytes is used. Padding may follow the payload if need be.
* **Frame check sequence (FCS):** The FCS as in HDLC is simply 2-byte or 4- byte CRC.

**CHAPTER 3.2**

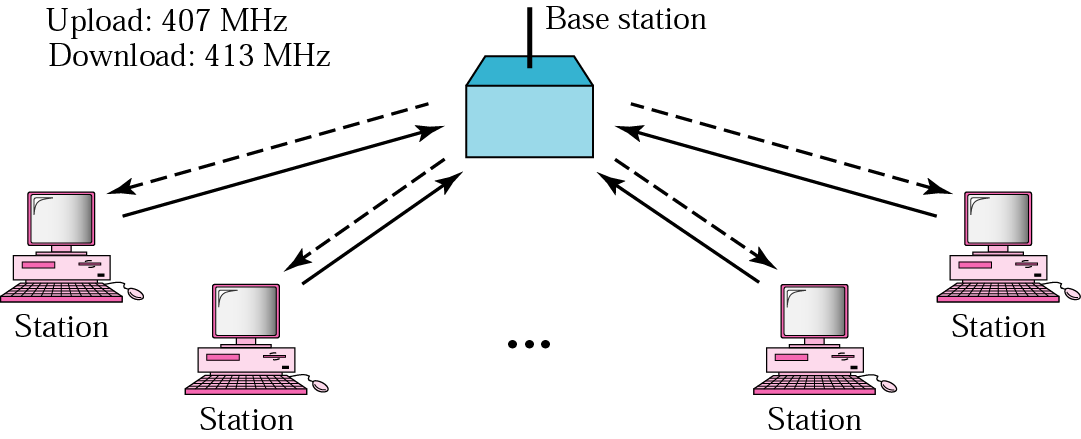
**MEDIA ACCESS CONTROL (MAC) LAYER**

**MULTIPLE ACCESS PROTOCOLS**

**ALOHA**

This earliest random access method was developed at the University of Hawaii in the early 1970s by Norman Abramson and colleagues. It was designed to be used on a radio (wireless) LAN with a data rate of 9600 bps as shown in fig 4.1 in which uncoordinated users are competing for the use of a single shared channel.

Here a base station is the central controller. Every station that needs to send a frame to another station first sends to the base station. The base station receives the frame and relays it to the intended destination. ALOHA system can be categorized with **pure** and **slotted**. They differ with respect to whether time is divided into discrete slots into which all frames must fit. Pure ALOHA doesn’t require global synchronization while slotted ALOHA does.



**Figure 4.1 ALOHA network**

**Pure ALOHA**: In this system, frame transmission can begin at any instant. There is no master clock dividing time into discrete intervals. Any station sends a frame (data) when it has a frame to send. After sending the frame, the station waits for an acknowledgement. If it does not receive an acknowledgement during the allotted time, which is 2 times the maximum propagation delay (the time it takes for the first bit of the frame to reach every station), the station assumes that the frame (or the acknowledgement) has been destroyed and resends the frame again after a random amount of time. After several tries, if there is no acknowledgement the station gives up.

It is obvious that there are potential collisions in this arrangement. The medium is shared between the stations. Two or more stations may send the data at the same time. In this case the data from the two stations collide and become garbled. Systems in which multiple users share a common channel in a way that can lead to conflicts (collision) are widely known as **contention** systems.

A sketch of frame generation in an ALOHA system is given in Fig. 4-1. The throughput of ALOHA system is maximized by having a uniform frame size rather than by allowing variable length frames.

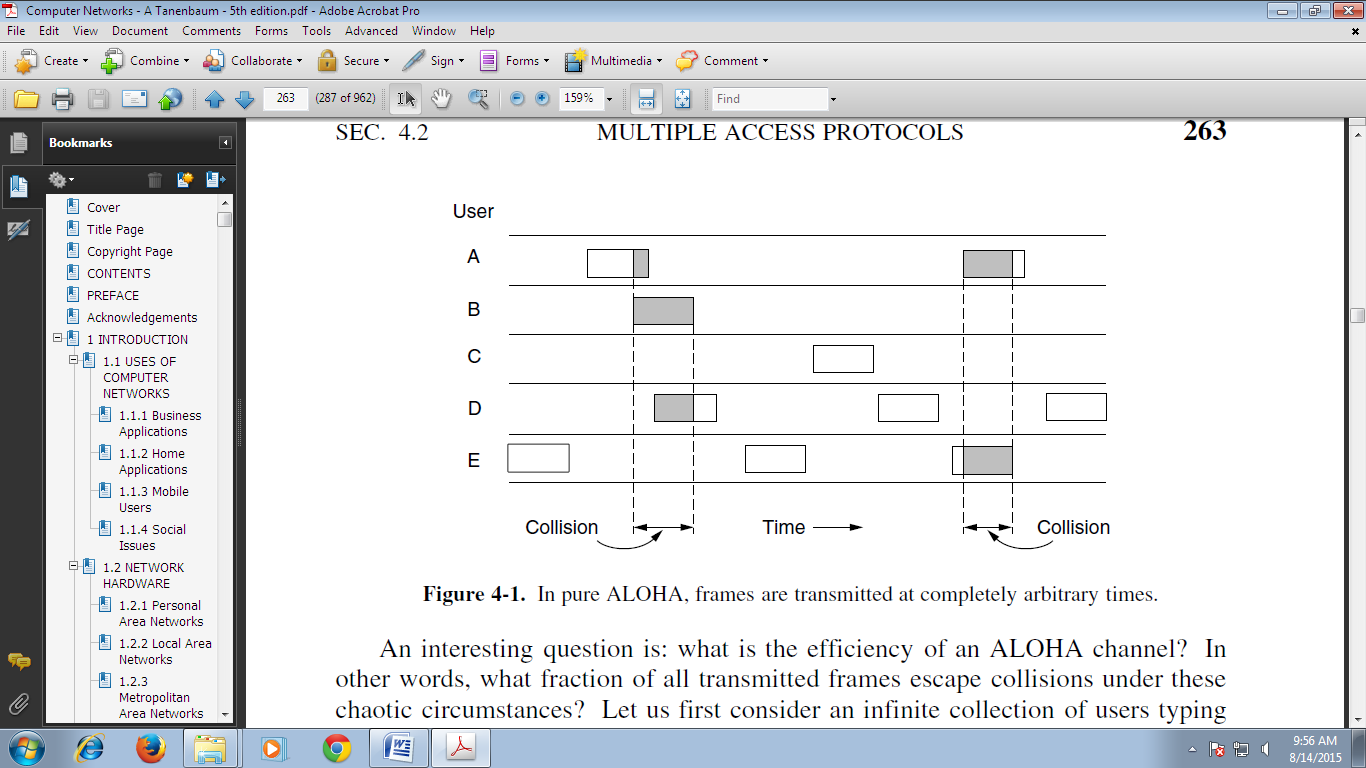
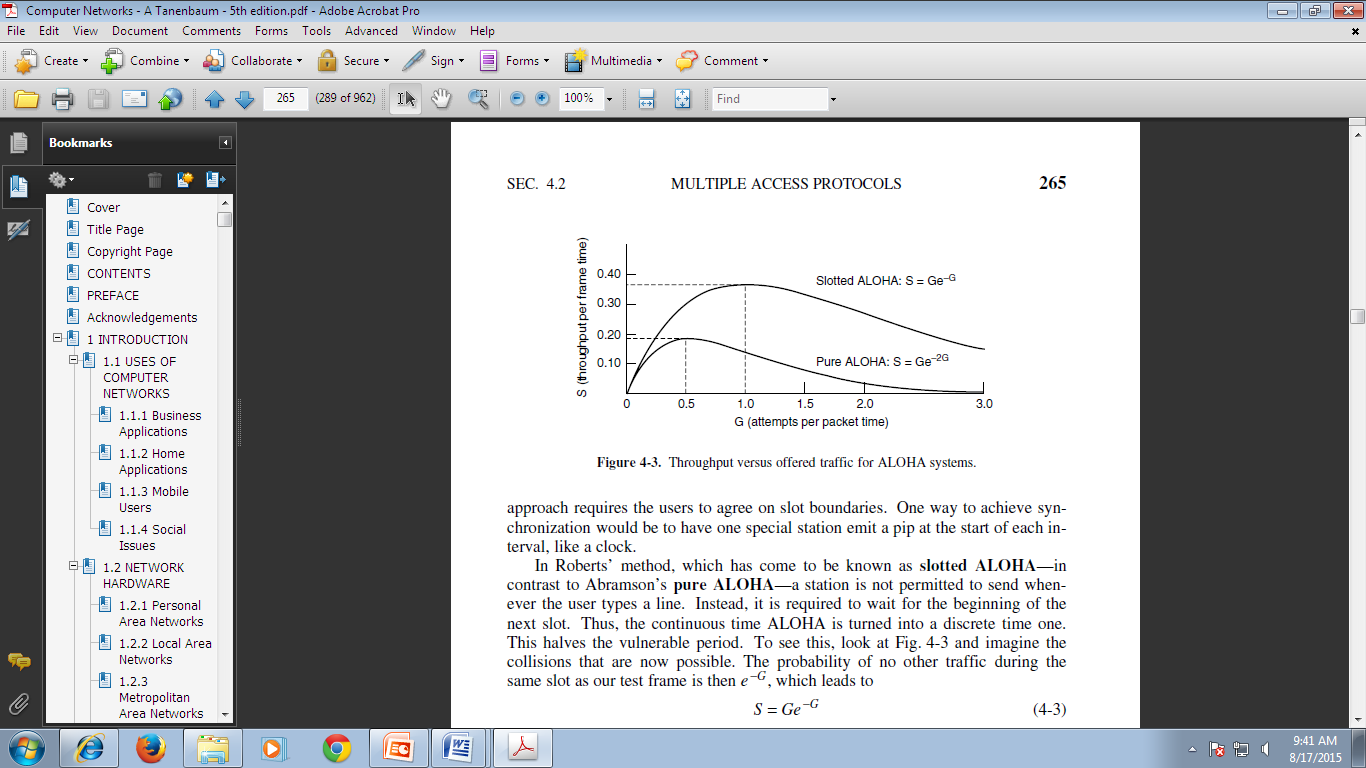


Figure 4.1: In pure ALOHA, frames are transmitted at completely arbitrary times.

Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be garbled. If the first bit of the new frame overlaps with just the last bit of a frame almost finished, both frames will be totally destroyed and both will have to be retransmitted later.



**Slotted ALOHA**

Slotted ALOHA (proposed by Roberts in 19720), divides time into discrete intervals, each interval corresponding to one frame. In contrast to pure ALOHA, slotted ALOHA does not permit a station to send the frame whenever it is ready to send. Instead, it is required to wait for the beginning of the next slot. Following figure shows an example of frame collision in slotted ALOHA.

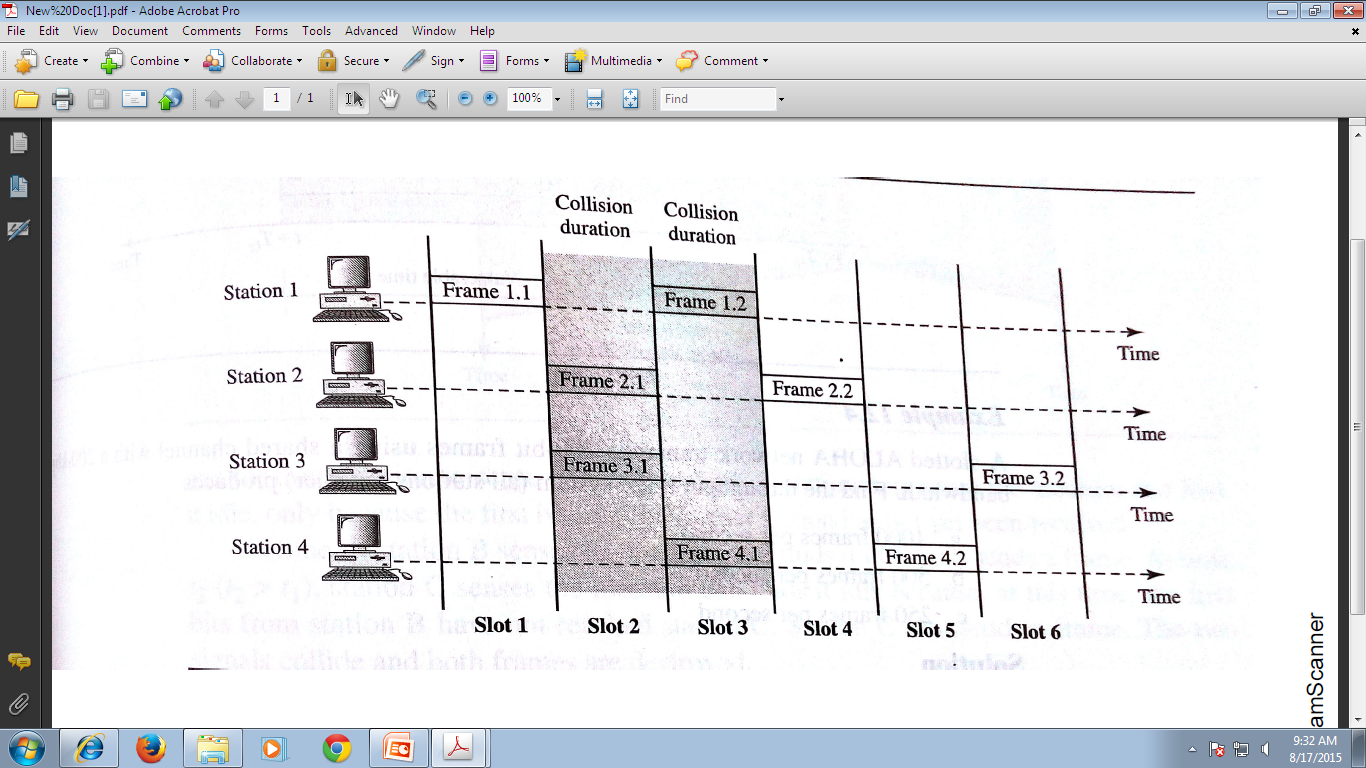


Figure: Frames in a slotted ALOHA network

Slotted ALOHA was invented to improve the efficiency of pure ALOHA.

**Efficiency of an ALOHA Channel**

Let frame (transmission) time = amount of time needed to transmit the standard fixed-length frame (i.e.,

frame length (in bits) divided by the bit rate).

N = Mean number of frames per frame time generated by infinite population of users according to Poisson

distribution.

Infinite population assumption is needed to ensure that N does not decrease as users become blocked. If N > 1, the user community is generating frames at a higher rate than the channel can handle, and nearly every frame will suffer a collision. For reasonable throughput (bits transmitted per second successfully), we would expect 0 < N < 1.

In addition to new frames, the stations also generate transmission of frames that previously suffered collisions. Let us further assume that the probability of k transmission attempts per frame time, old and new combined, is also Poisson, which mean G frames per frame time. Obviously, G ≥ N.

At low load (i.e., N ≈ 0), there will be few collisions, hence few transmissions, so G ≈ N. At high load there will be many collisions, so G > N.

Let P0 be the probability of successful transmission. Thus under all loads, the throughput, S, is just the offered load, G, times the probability, P0.

Therefore S = G P0

A frame will not suffer a collision if no other frames are sent within one frame time of its start, as shown in Fig.4-2.

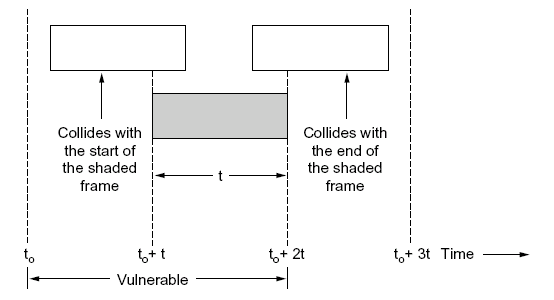


Fig.4-2. Vulnerable period for the shaded frame

Let *t* be the time required to send a frame. If any other user has generated a frame between time t0 and and*t0*+ *t*, the end of that frame will collide with the beginning of the shaded one. Similarly, any other frame started between *t0*+ *t*and *t0*+*2t* will bump into the end of the shaded frame.

The probability that *k* frames are generated during a given frame time is given by the Poission distribution:

*Pr*[*k*]= *G k e-G*/ *k*!

So the probability of zero frames is just *G 0 e-G*/ *0*! = *e-G*. In an interval two frame times long, the mean number of frames generated is 2G. The probability of no other traffic being initiated during the entire vulnerable period is thus given by

P0 = e -2G

Using S = GP0, we get

S = e -2G

Smax= dS/dG = d[Ge -2G]/dG = [e -2G - 2Ge -2G] = e -2G [1- 2G] = 0

or 1- 2G = 0

or G = ½ = 0.5

Thus the maximum throughput occurs at G = ½.

At G = ½, Smax b = ½ e-2\*1/2  = 1/2e = 0.184.

In other words, the best we can hope for is a channel utilization of 18 percent.

**Efficiency Calculation of Slotted ALOHA**

Since in Slotted ALOHA, it is required to wait for the beginning of the next slot, the vulnerable period is now halved which leads to

S = Ge -2G

Smax= dS/dG = e -G [1- G] = 0

or G = 1 for Smax

In this case (for G = 1),

Smax= 1 Ge -1 = 1/e = 0.368

Thus the best we can hope for 37 percent success. Operating at higher values of G reduces the number of empties but increases the number of collisions exponentially.

**Q.** Compare the performance of pure ALOHA at low and high load.

**Ans:** The relation between the offered traffic and the throughput for pure and slotted ALOHA systems is shown in Fig.4-3.

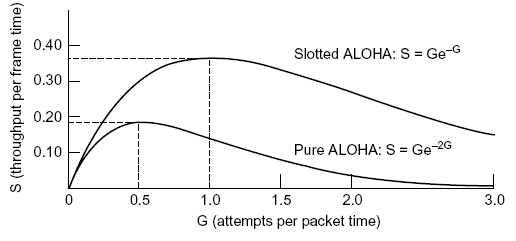


Fig. 4-3. Throughput versus offered traffic for ALOHA systems

**Q.** Consider the delay of pure ALOHA versus slotted ALOHA at low load. Which one is less? Explain your answer. [5 marks]

**Ans:** With pure ALOHA, transmission can start instantly. At low load, no collisions are expected so the transmission is likely to be successful. With slotted ALOHA, it has to wait for the next slot. This introduces half a slot time of delay.

**4.2.2 CARRIER SENSE MULTIPLE ACCESS PROTOCOL (CSMA)**

Protocols In which stations listen for a carrier (i.e., a transmission) and act accordingly are called **carrier sense protocols**. If the channel is sensed as busy, no station will attempt to use it until it goes idle. Several versions of carrier sense protocols are discussed below.

# Persistent and Non persistent CSMA

In a **persistentstrategy**, a station that has a frame to send senses the channel. If the channel is idle, the station sends a frame. This method has two variations: 1-persistent and p-persistent.

In **1-persistent** method, if the station finds the channel idle, the station sends its frame immediately. If collision occurs, the station waits a random amount of time and starts all over again (with a probability of 1). This method increases the chance of collision because two or more stations may send their frames after finding the medium idle. The protocol is called 1 – persistent because the station transmits with a probability of 1 when it finds channel idle. The longer the propagation delay, the more important this effect becomes, and the worse performance of the protocol. This approach leads to a higher performance than pure ALOHA and slotted ALOHA which do not sense carrier.

The **p-persistent** method is applied to a slotted channel and works as follow: If the station finds the channel idle, the station may or may not send. It sends with probability p and refrain from sending with probability q = 1 – p until the next slot. For example, if p is 0.2, means that each station, after sensing an idle channel, sends with a probability of 0.2 (20 percent of the time) and refrains from sending with a probability of 0.8 ( 80 percent of the time) until the next slot. The station generates a random number between 1 and 100. If the random number is less than 20, the station will send. Otherwise the station refrains from sending. The p – persistent strategy combines the advantage of the other two strategies (non persistent and 1 – persistent). It reduces the chance of collision and improves the efficiency.

**Non persistent CSMA**: - In this strategy, a station that has frame to send senses the channel. If the channel is idle, the station sends immediately. If the channel is not idle, the station waits a random period of time without sensing the medium and then repeats the algorithm by sensing the channel again. The non persistent approach reduces the chance of collision because it is unlikely that two or more stations wait the same amount of time and retry again simultaneously. Consequently, this algorithm leads to better channel utilization but longer delays than 1 – persistent CSMA.

The behavior of the above three persistent methods are shown by the following figure.

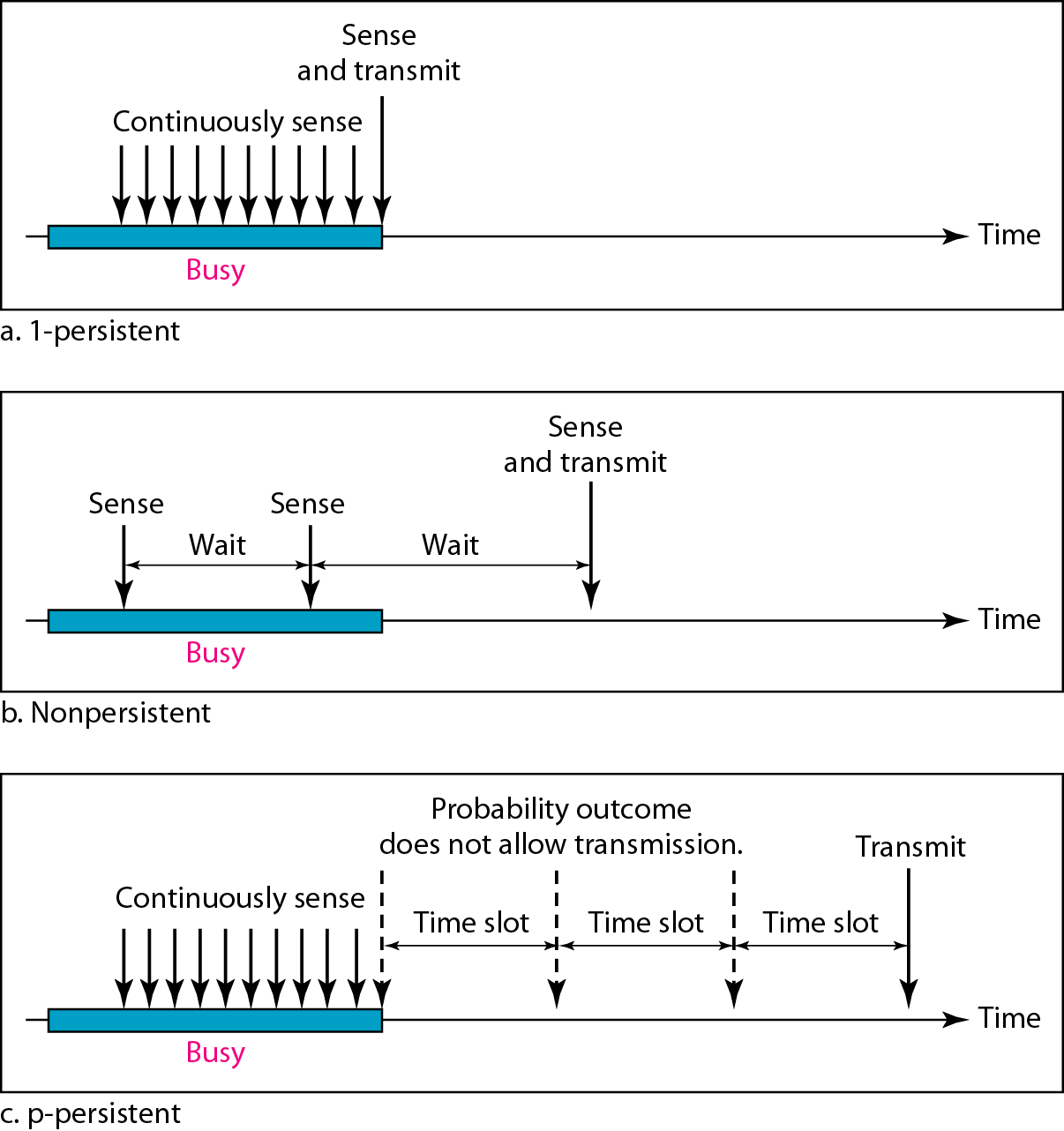


Figure :Behaviour of three persistence methods.

**CSMA with Collision Detection (CSMA/CD)**

Persistent and non Persistent CSMA protocols are an improvement over ALOHA because they ensure that no station begins to transmit when it senses the channel busy. Another improvement is for stations to abort their missions as soon as they detect a collision. In other words, if two stations sense the channel to be idle and begin transmitting simultaneously, they will both detect the collision almost immediately. Rather than finish transmitting their frames, which are irretrievably garbled anyway, they should abruptly stop transmitting as soon as the collision is detected. Quickly transmitting damaged frames saves time and bandwidth. This protocol known as **CSMA/CD (CSMA with Collision Detection)** is widely used in LAN’S in the MAC sub layer. In particular, it is the basis of the popular Ethernet LAN.

Conceptual model of CSMA/CD as well as many other LAN protocols is shown in fig 4.5

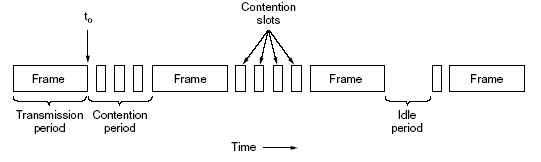


Fig. 4-5. CSMA/CD can be in one of three states: contention, transmission, or idle.

At the point marked *t0*, a station has finished transmitting its frame. Any other station having a frame to send may now attempt to do so. If two or more stations decide to transmit simultaneously, there will be a collision. Collisions can be detected by looking at the power or pulse width of the received signal and comparing it to the transmitted signal.

After a station detects a collision, it aborts its transmission, waits a random period of time, and then tries again, assuming that no other station has started transmitting in the meantime. Therefore, the method for CSMA/ CD will consists of alternating contention and transmission periods, with idle periods occurring when all stations are quite (i.e., for lack of work).

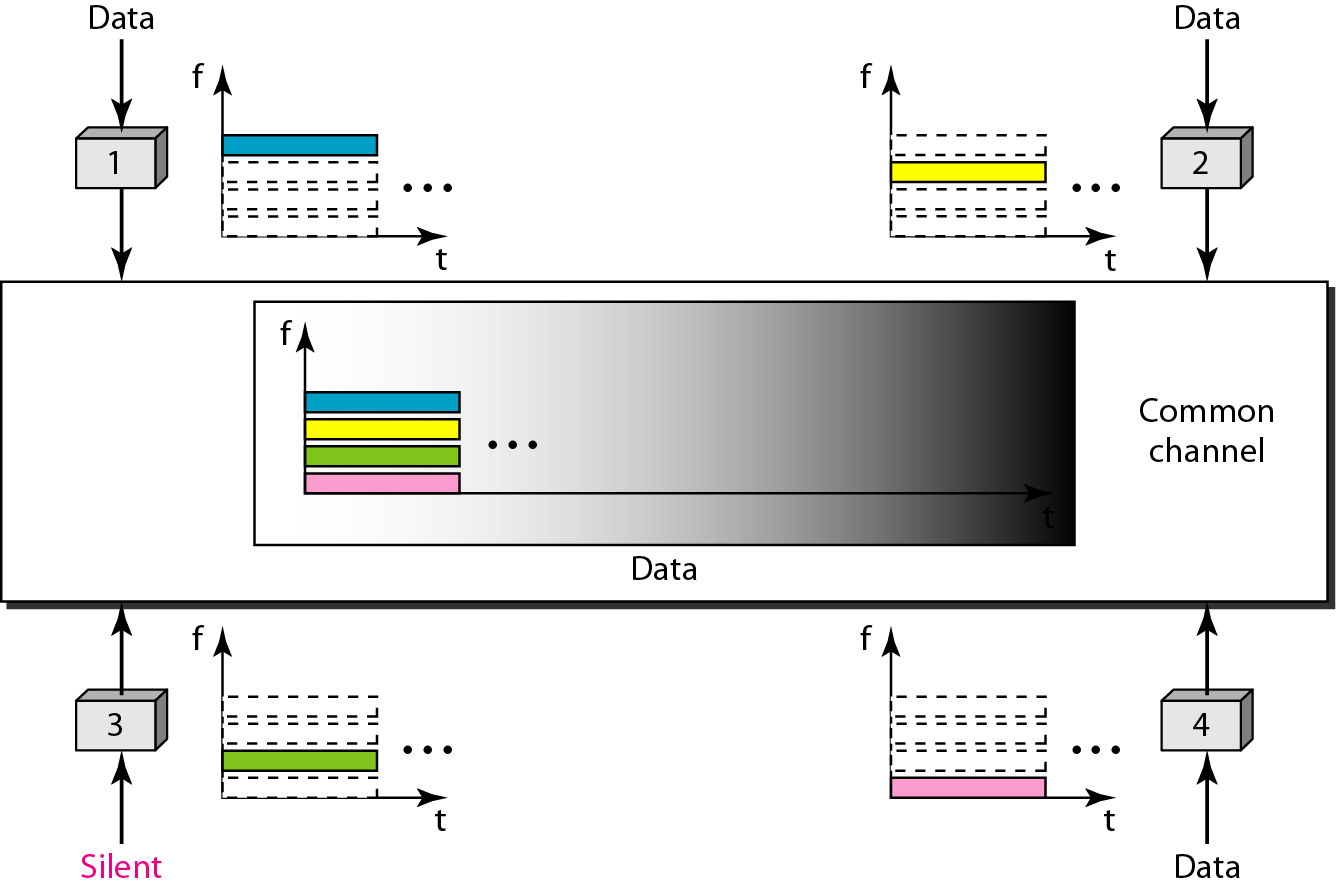
A station not hearing a collision for a time equal to 2τ (τ is the time a signal takes to travel between two stations) after starting its transmission could be same it has seized the cable and can transmit frames at any rate it wants to..

**Q. Explain FDMA, TDMA, AND CDMA. [10 marks][M13][D13][M12]**

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

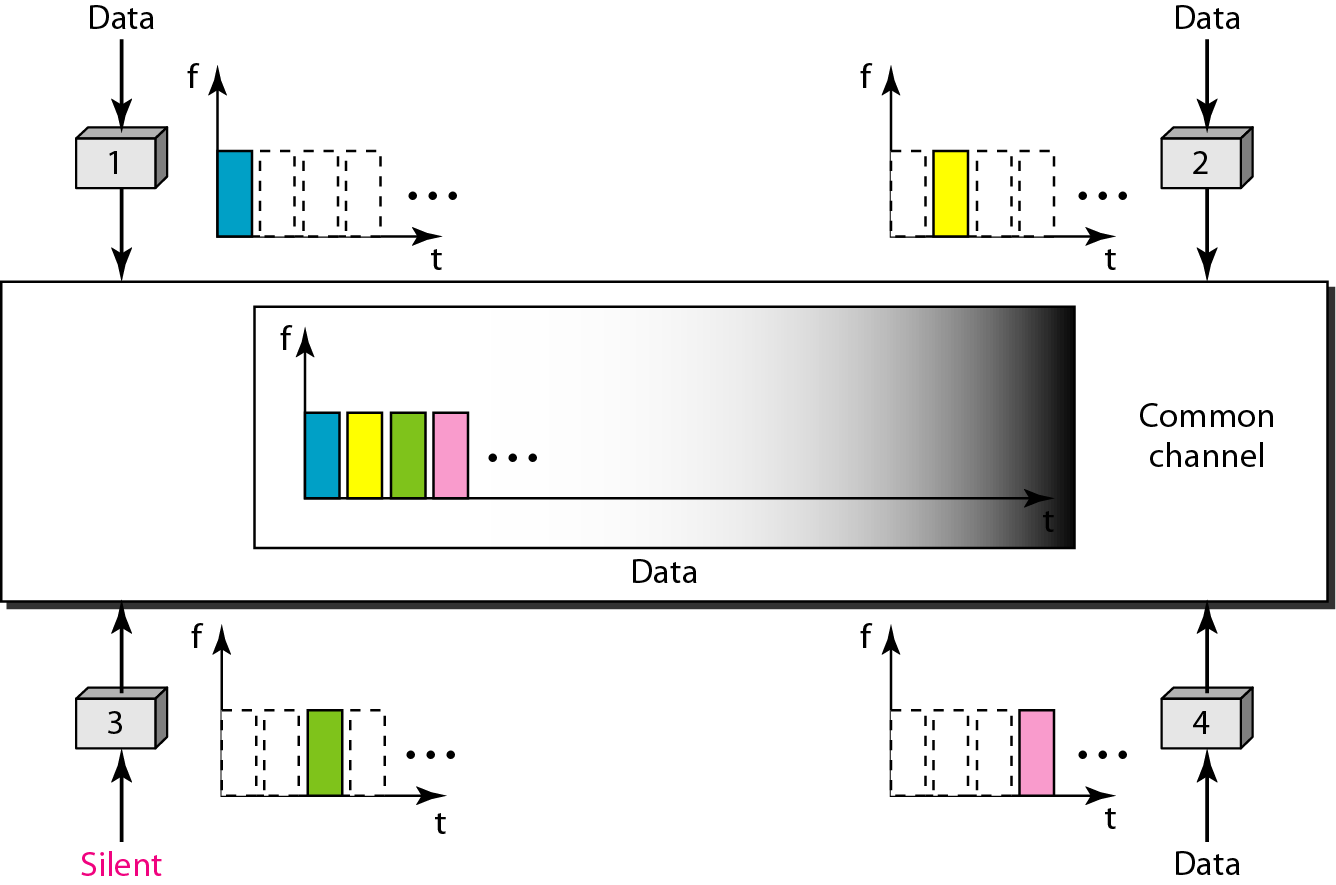
In FDMA, the available bandwidth is divided into frequency bands. Each station is allocated a band to send its data. In other words, each band is reserved for a specific station, and it belongs to the station all the time.

To prevent station interferences, the allocated bands are separated from one another by small guard bands.



**Time-Division Multiple Access (TDMA)**

In 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. Following figure shows the idea behind TDMA.



**CODE DIVISION MULTIPLE ACCESS (CDMA)**

In code division multiple access (CDMA) all channels use the same frequency at the same time for transmission. Separation of channels is achieved by assigning each channel its own *code* to enable it to access the shared medium. Guard spaces are realized by using codes with the necessary distance in the space, e. g., orthogonal codes. A code for certain user should have a good autocorrelation and should be orthogonal to other codes. An example of CDMA is shown by the following table

|  |  |  |
| --- | --- | --- |
| **Stations** | **Codes** | **Encoded and transmitted message 1101**  **(By XORing individual bit with code)** |
| A | 100101 | 011010 011010 100101 011010 |
| B | 110011 | 001100 001100 110011 001100 |
| C | 110110 | 001001 001001 110110 |

**Ethernet**

Ethernet is the most widely used local area network protocol. The IEEE 802.3 Standard defines 1- persistent *CSMA/CD* as the access method for first-generation 10-Mbps. The data link layer of Ethernet consists of the LLC sublayer and the MAC sublayer. The MAC sublayer is responsible for the operation of the *CSMAlCD*access method and framing. Each station on an Ethernet network has a unique 48-bit address imprinted on its network interface card (NIC). The original Ethernet was created in 1976 at Xerox's Palo Alto Research Center (PARC). Since then, it has gone through four generations: **Standard Ethernet** (l0 Mbps), **Fast Ethernet** (100 Mbps), **Gigabit Ethernet** (l Gbps), and **Ten-Gigabit Ethernet** (l0 Gbps).

**STANDARD ETHERNET**

**MAC Sublayer**

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

***Addressing***

Each station on an Ethernet network (such as a PC, workstation, or printer) has its own network interface card (NIC). The NIC fits inside the station and provides the station with a 6-byte physical address. As given below, the Ethernet address is 6 bytes (48 bits), normally written in hexadecimal notation, with a colon between the bytes.

06:01 :02:01:2C:4B

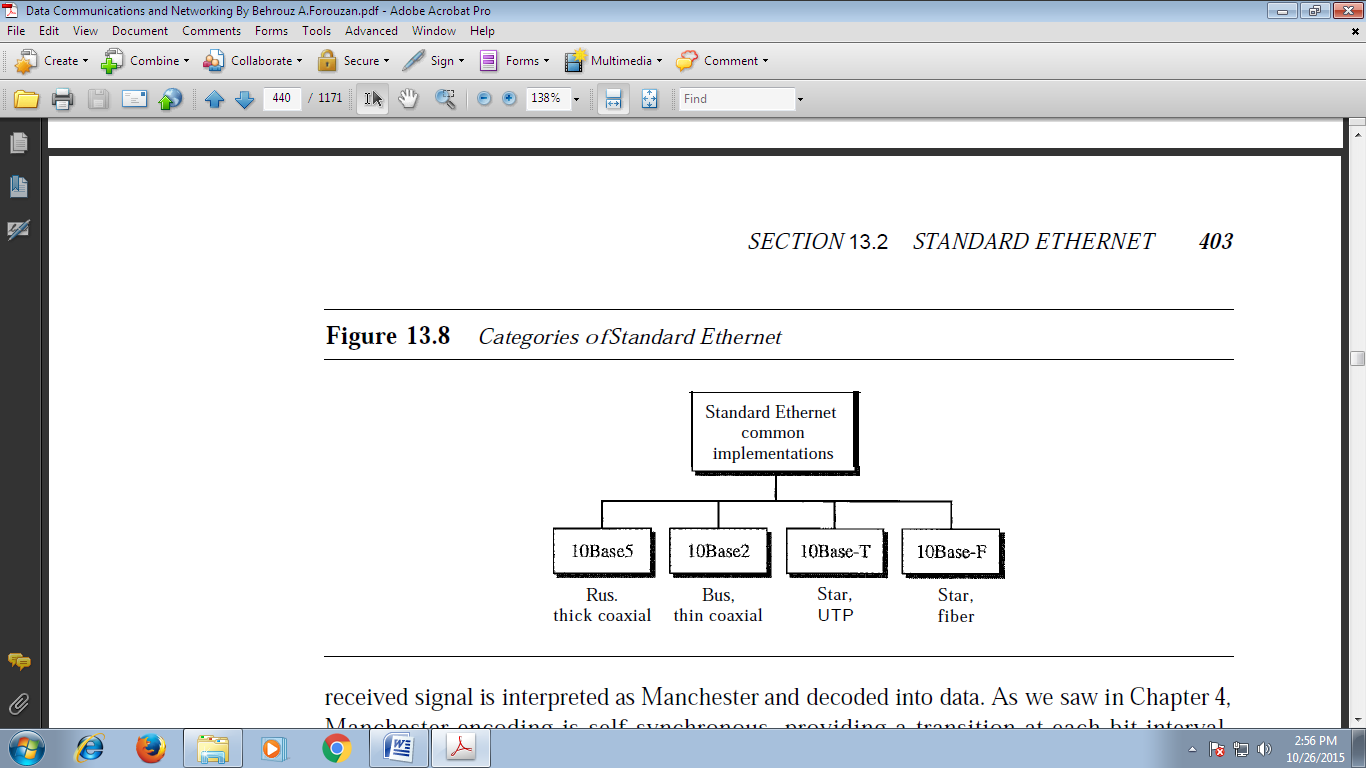
A source address is always a unicast address-the frame comes from only one station. The destination address, however, can be unicast, multicast, or broadcast.

**Access Method: CSMAICD**

Standard Ethernet uses 1-persistent CSMA/CD

**Physical Layer**

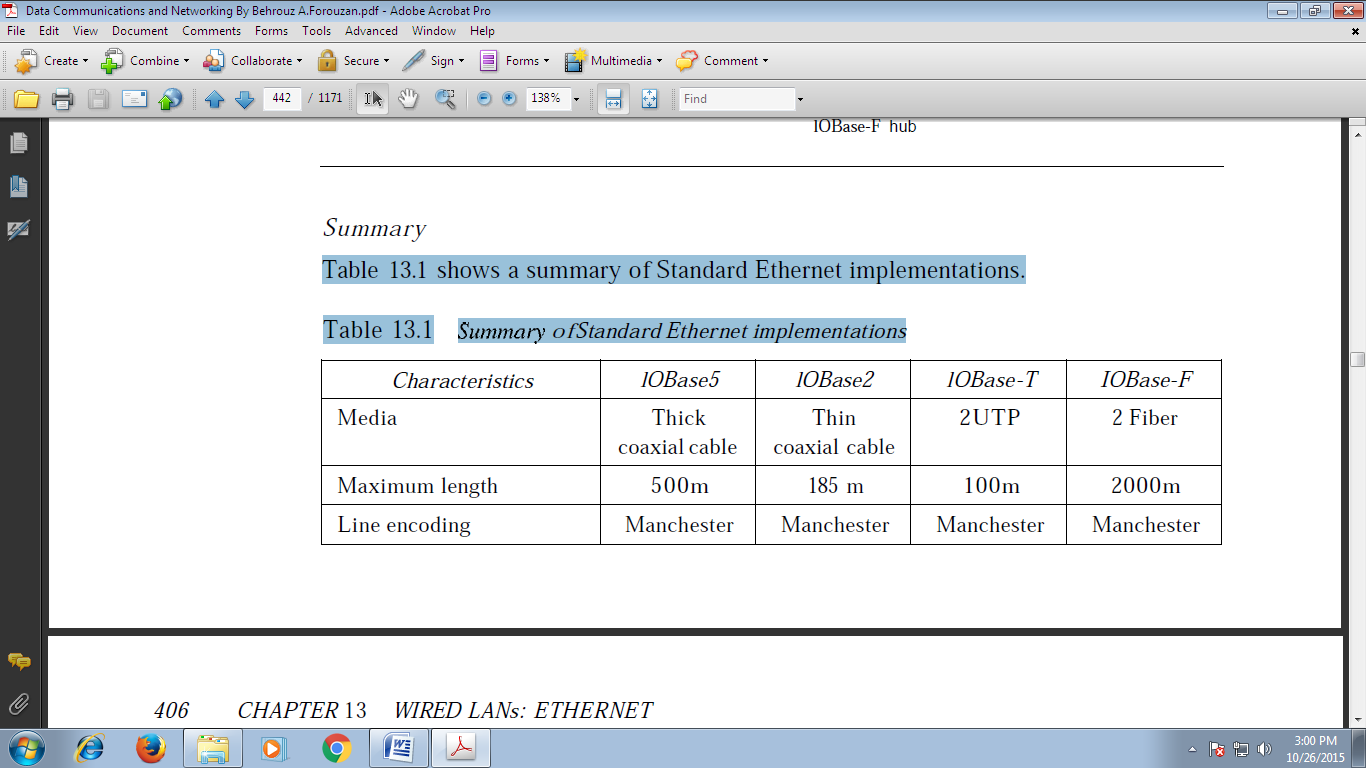
The Standard Ethernet defines several physical layer implementations; four of the most common, are shown in Figure given below.

.

**Figure:** *Categories of Standard Ethernet*

Following table shows a summary of Standard Ethernet implementations.

Table: Summary of Standard Ethernet implementations



**Evolution (Changes) in the Standard of Ethernet**

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

**Bridged Ethernet**

The first step in the Ethernet evolution was the division of a LAN by bridges. Bridges have two effects on an Ethernet LAN: They raise the bandwidth and they separate collision domains.

**Switched Ethernet**

The idea of a bridged LAN can be extended to a switched LAN. Instead of having two to four networks, why not have N networks, where N is the number of stations on the LAN? In other words, if we can have a multiple-port bridge, why not have an N-port switch? In this way, the bandwidth is shared only between the station and the switch (5 Mbps each). In addition, the collision domain is divided into N domains.

**Full-Duplex Ethernet**

One of the limitations of 10Base5 and l0Base2 is that communication is half-duplex (l0Base-T is always full-duplex); a station can either send or receive, but may not do both at the same time. The next step in the evolution was to move from switched Ethernet to full-duplex switched Ethernet. The full-duplex mode increases the capacity of each domain from 10 to 20 Mbps.

In full-duplex switched Ethernet, there is no need for the CSMA**/**CD method. In a full-duplex switched Ethernet, each station is connected to the switch via two separate links. Each station or switch can send and receive independently without worrying about collision. Each link is a point-to-point dedicated path between the station and the switch. There is no longer a need for carrier sensing; there is no longer a need for collision detection. The job of the MAC layer becomes much easier. The carrier sensing and collision detection functionalities of the MAC sublayer can be turned off.

**Q.1. Write note on (a) Ethernet [5 marks][D14] (b) Ethernet frame format [10 marks][M12]**

**Q.2. Make a comparative study of switched ethernet, fast Ethernet and gigabit Ethernet. [10 marks][D13]**

**FAST ETHERNET**

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

1. Upgrade the data rate to 100 Mbps.

2. Make it compatible with Standard Ethernet.

3. Keep the same 48-bit address.

4. Keep the same frame format.

5. Keep the same minimum and maximum frame lengths.

A new feature added to Fast Ethernet is called **autonegotiation**which allows two devices to negotiate the mode or data rate of operation

*Implementation*

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

**GIGABIT ETHERNET**

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

1. Upgrade the data rate to 1 Gbps.

2. Make it compatible with Standard or Fast Ethernet.

3. Use the same 48-bit address.

4. Use the same frame format.

5. Keep the same minimum and maximum frame lengths.

6. To support autonegotiation as defined in Fast Ethernet.

***Implementation***

Gigabit Ethernet can be categorized as either a two-wire or a four-wire implementation.The two-wire implementations use fiber-optic cable (1000Base-SX, short-wave, or l000Base-LX, long-wave), or STP (1000Base-CX). The four-wire version uses category 5 twisted-pair cable (l000Base-T).

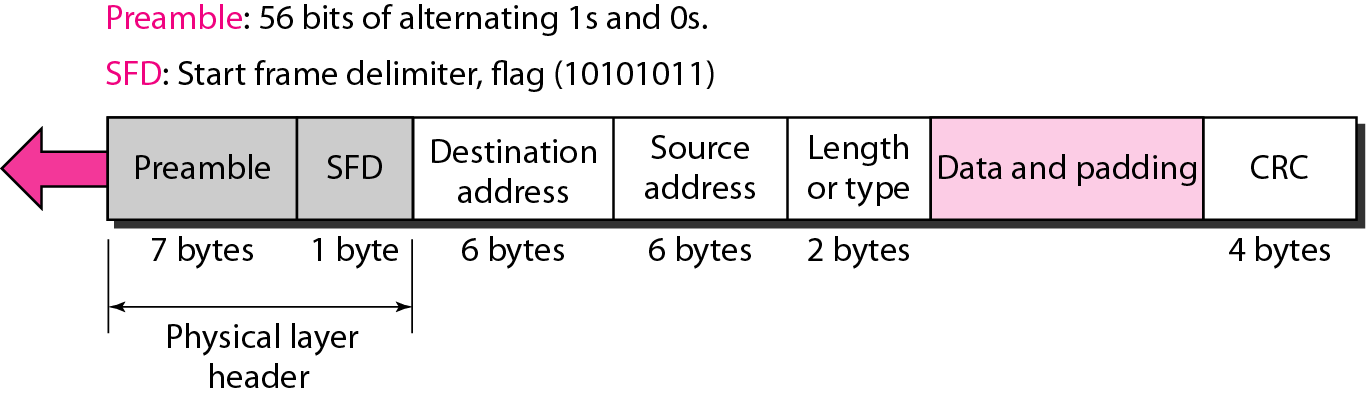
**Ten-Gigabit Ethernet**

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

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

**Ethernet Frame Format**

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



**Figure: IEEE *802.3 MAC frame***

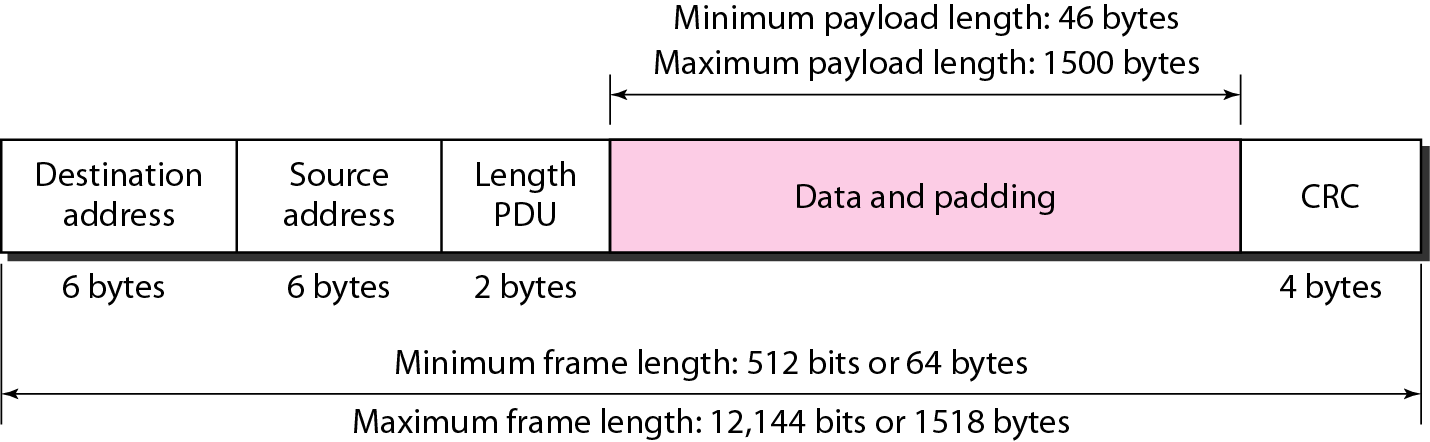
Function of each field is described below:

* ***Preamble***: The first field of the 802.3 frame contains 7 bytes (56 bits) of alternating 0s and 1s that alerts the receiving system to the coming frame and enables it to synchronize its input timing. The pattern provides only an alert and a timing pulse. The 56-bit pattern allows the stations to miss some bits at the beginning of the frame. The preamble is actually added at the physical layer and is not (formally) part of the frame.
* ***Start frame delimiter (SF*:** The second field (l byte: 10101011) signals the beginning of the frame. The SFD warns the station or stations that this is the last chance for synchronization. The last 2 bits is 11 and alerts the receiver that the next field is the destination address.
* ***Destination address (D*:*.*** The DA field is 6 bytes and contains the physical address of the destination station or stations to receive the packet.
* ***Source address (SA):*** The SA field is also 6 bytes and contains the physical address of the sender of the packet. o Length or type. This field is defined as a type field or length field. The original
* ***Length or type*:** This field is defined as a type field or length field. The original Ethernet used this field as the type field to define the upper-layer protocol using the MAC frame. The IEEE standard used it as the length field to define the number of bytes in the data field. Both uses are common today.
* ***Dat*:** This field carries data encapsulated from the upper-layer protocols. It is a minimum of 46 and a maximum of 1500 bytes.
* ***CR*:** The last field contains error detection information, in this case a CRC-32.

**Q. State the reason for having a minimum length requirement for a frame in Ethernet. How is it achieved? [5 marks][D13]**

***Frame Length***

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

****

**Figure**: *Minimum and maximum lengths*

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

The standard defines the **maximum length** of a frame (without preamble and SFD field) as 1518 bytes. If we subtract the 18 bytes of header and trailer, the maximum length of the payload is 1500 bytes. The maximum length restriction has two historical reasons. First, memory was very expensive when Ethernet was designed: a maximum length restriction helped to reduce the size of the buffer. Second, the maximum length restriction prevents one station from monopolizing the shared medium, blocking other stations that have data to send.