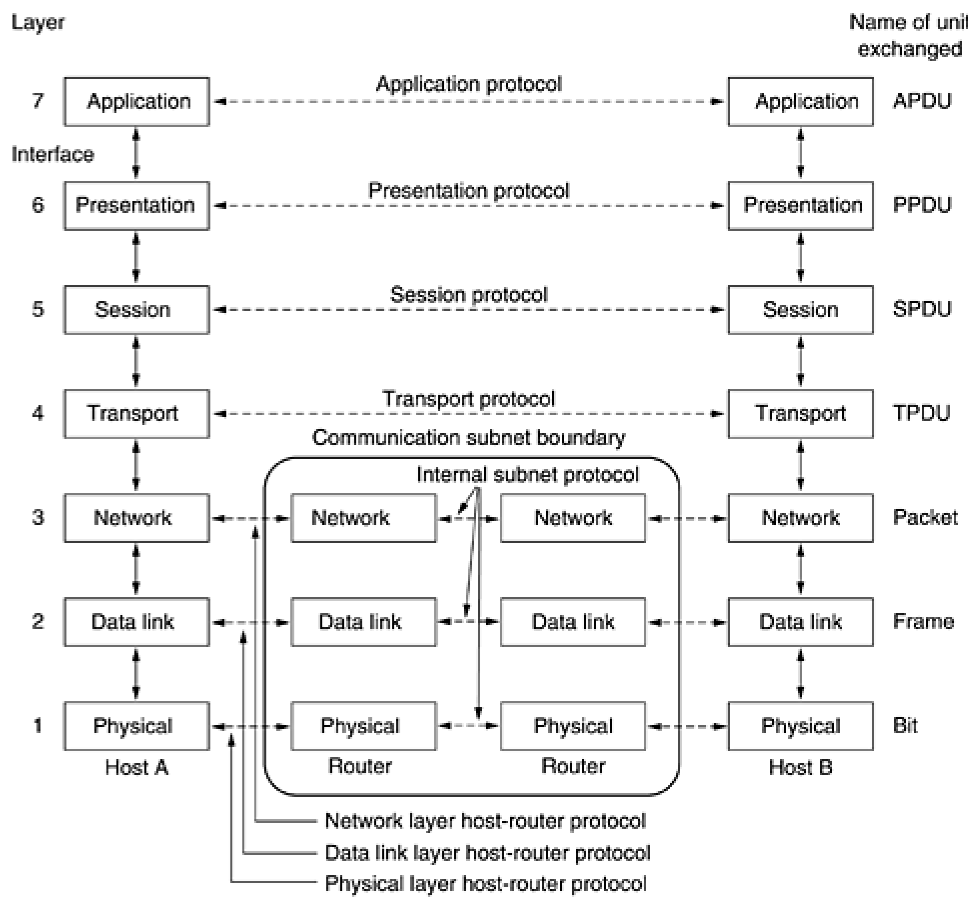
Chapter 3. The Data Link Layer

The OSI reference model



The Data Link Layer

- Data link layer is responsible for converting data stream to signals bit by bit and to send that over the underlying hardware.
- At the receiving end, Data link layer picks up data from hardware which are in the form of electrical signals, assembles them in a recognizable frame format, and hands over to upper layer.
- Data link layer Transforms the raw data bits to a data frame (few hundred/thousand bits).
- It accomplishes this task by having the sender break up the input data into data frames (typically a few hundred or a few thousand bytes) and transmit the frames sequentially.
- Is responsible for moving frames from node to node or computer to computer.

- Two sublayers: Logical Link Control (LLC) and the Media Access Control (MAC)

Logical Link Control (LLC)

–Data Link layer addressing, flow control, address notification, error control

Media Access Control (MAC)

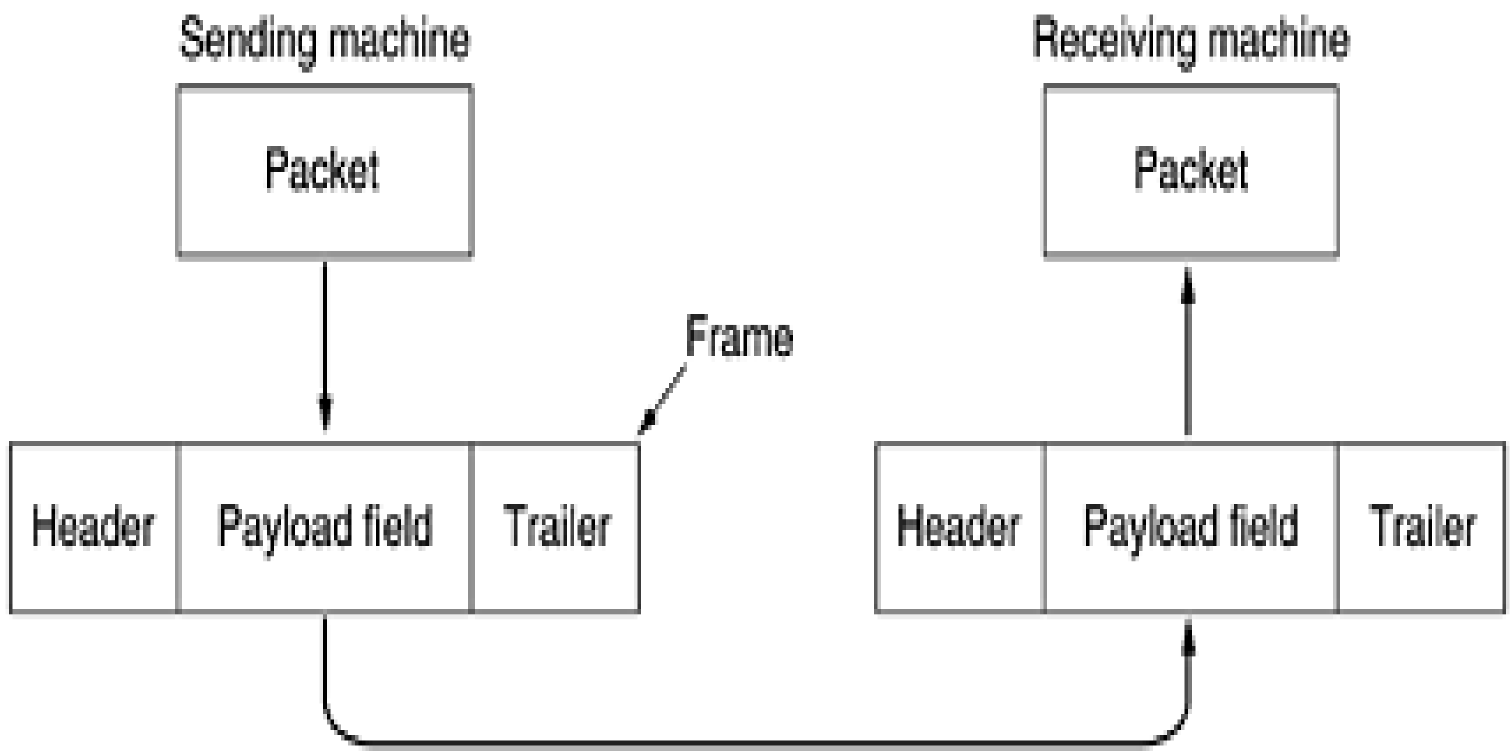
–Determines which computer has access to the network media at any given time

–Determines where one frame ends and the next one starts, called frame synchronization

3.1 Data Link Layer Design Issues

- The data link layer has a number of specific functions it can carry out. These functions 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 swamped by 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.
- Frame management forms the heart of what the data link layer does.

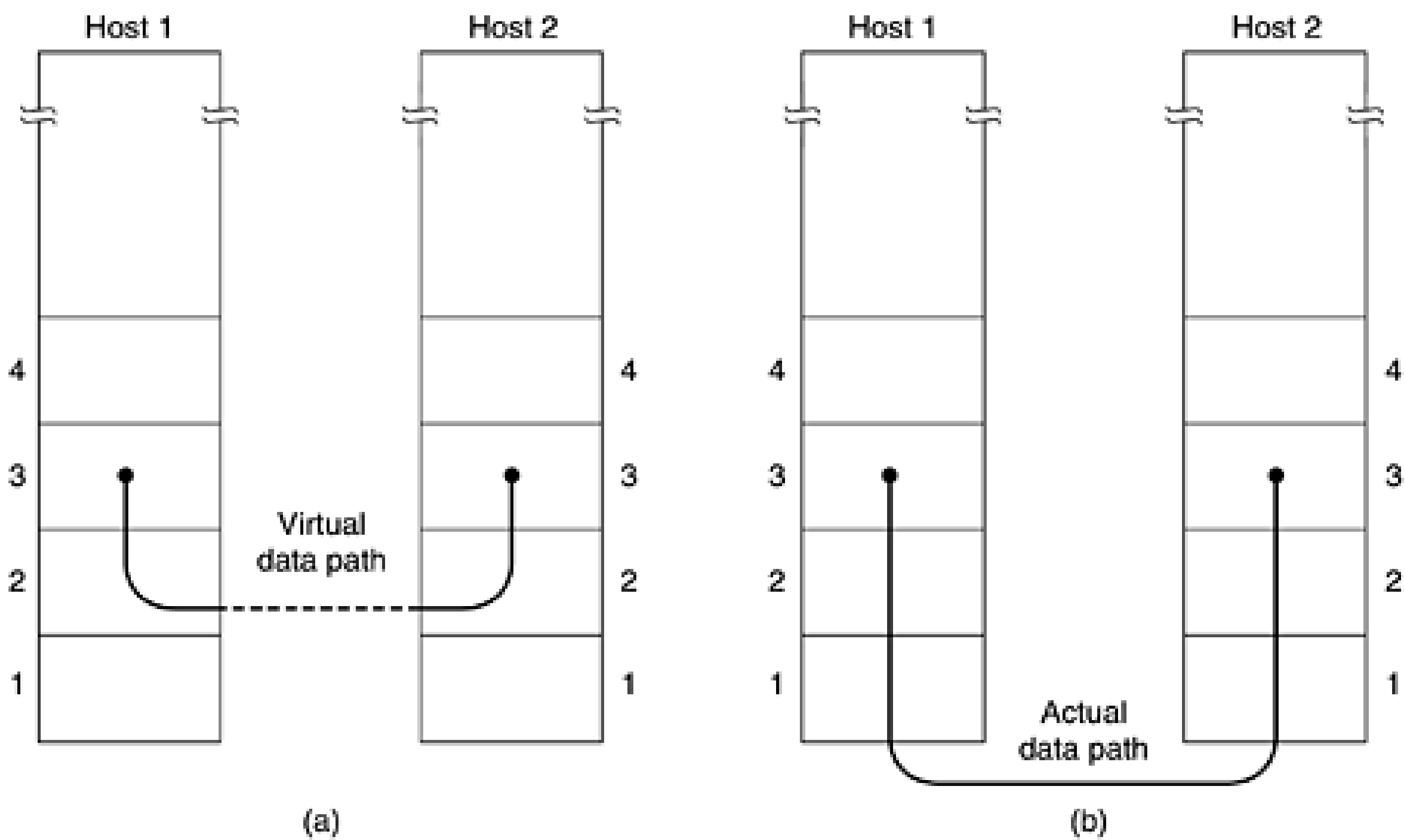
Figure 3-1. Relationship between packets and frames



3.1.1 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, call it 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.

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



- The data link layer can be designed to offer various services.
- The actual services offered can vary from system to system.
- Three reasonable possibilities that are commonly provided are
  - Unacknowledged connectionless service.
  - Acknowledged connectionless service.
  - Acknowledged connection-oriented service.

Unacknowledged connectionless 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

- 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. 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.
- The network layer always send a packet and wait for it to be acknowledged. If the acknowledgement is not forthcoming before the timer expires, the sender can just send the entire message again.
- The trouble with this strategy is that frames usually have a strict 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.
- On reliable channels, such as fiber, the overhead of a heavyweight data link protocol may be unnecessary.

Acknowledged connection-oriented service.

- 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.
- 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 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 and final phase, the connection is released, freeing up the variables, buffers, and other resources used to maintain the connection.

- Consider a typical example: a WAN subnet consisting of routers connected by point-to-point leased telephone lines.
- When a frame arrives at a router, the hardware checks it for errors, then passes the frame to the data link layer software (which might be embedded in a chip on the network interface board).
- The data link layer software checks to see if this is the frame expected, and if so, gives the packet contained in the payload field to the routing software.
- The routing software then chooses the appropriate outgoing line and passes the packet back down to the data link layer software, which then transmits it.

Framing

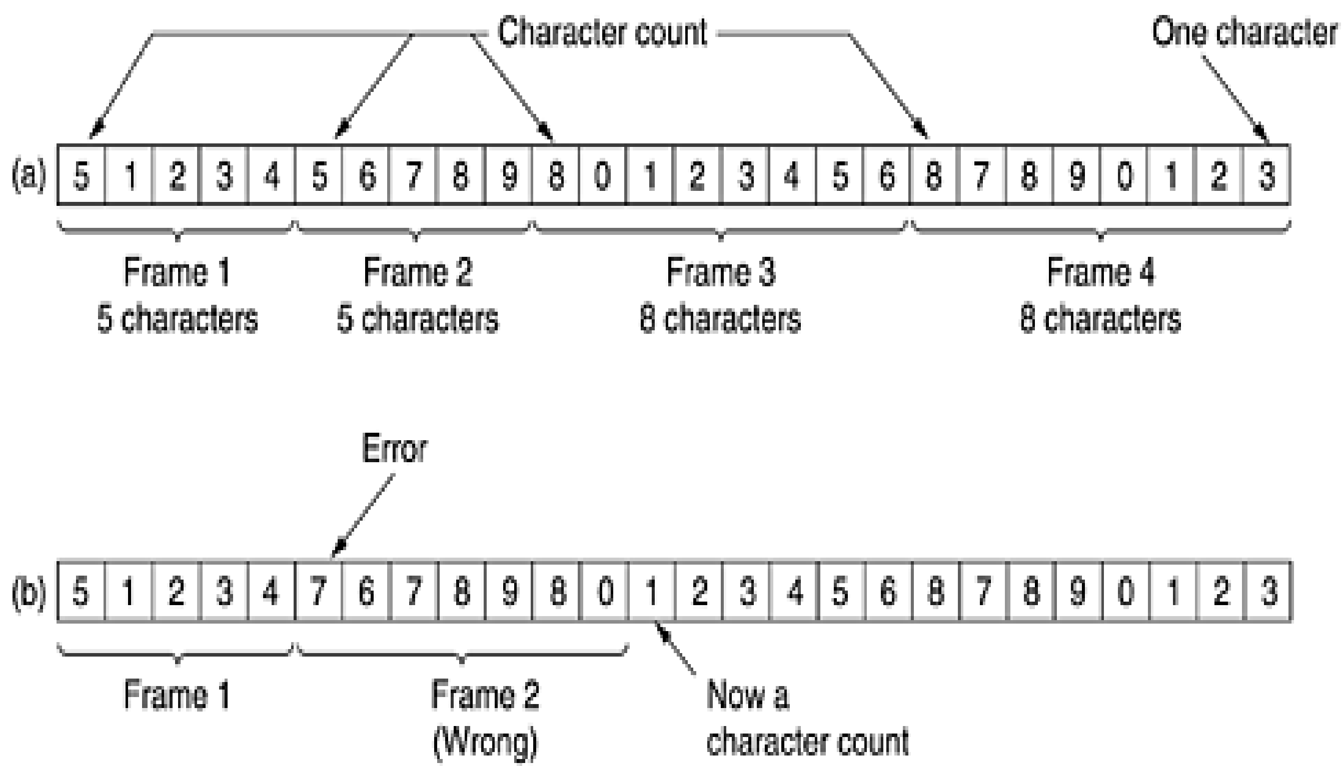
- The physical layer accept a raw bit stream and attempt to deliver it to the destination.
- This bit stream is not guaranteed to be error free. It is up to the data link layer to detect and, if necessary, correct errors.
- The usual approach is for the data link layer to break the bit stream up into discrete frames and compute the checksum for each frame.
- 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.

- One way to achieve this framing is to insert time gaps between frames, much like the spaces between words in ordinary text.
- However, networks rarely make any guarantees about timing, so it is possible these gaps might be squeezed out or other gaps might be inserted during transmission.
- Since it is too risky to count on timing to mark the start and end of each frame, other methods have been devised.
-
  - Character count.
  - Flag bytes with byte stuffing.
  - Starting and ending flags, with bit stuffing.

Character count method

- The first framing 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.
- The trouble with this algorithm is that the count can be garbled by a transmission error.
- 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.
- For this reason, the character count method is rarely used anymore.

Figure 3-4. A character stream. (a) Without errors. (b) With one error.



Flag bytes with byte stuffing

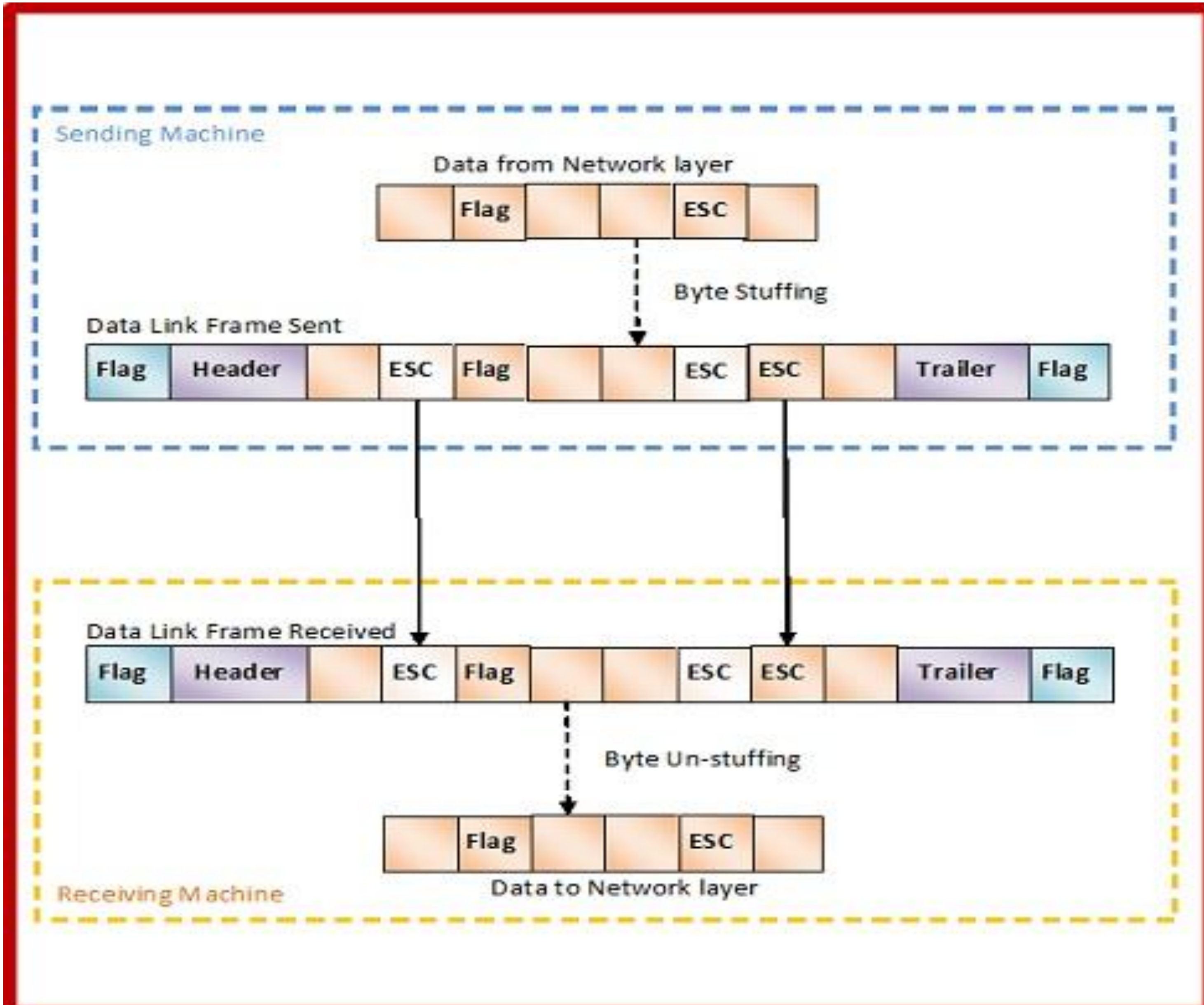
- The second framing method gets around the problem of resynchronization after an error by having each frame start and end with special bytes.
- In the past, the starting and ending bytes were different, but in recent years most protocols have used the same byte, called a flag byte, as both the starting and ending delimiter, known 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 one.

A frame has the following parts −

- Frame Header − It contains the source and the destination addresses of the frame.
- Payload field − It contains the message to be delivered.
- Trailer − It contains the error detection and error correction bits.
- Flags − 1- byte (8-bits) flag at the beginning and at end of the frame.

It is a protocol – dependent special character, signalling the start and end of the frame.

- In byte stuffing, a special byte called the escape character (ESC) is stuffed before every byte in the message with the same pattern as the flag byte.
- If the ESC sequence is found in the message byte, then another ESC byte is stuffed before it.


Sending Machine
Data from Network layer
Flag
ESC
Byte Stuffing
Data Link Frame Sent
Flag
Header
ESC
Flag
ESC
ESC
Trailer
Flag
Data Link Frame Received
Flag
Header
ESC
Flag
ESC
ESC
Trailer
Flag
Byte Un-stuffing
Flag
ESC
Data to Network layer
Receiving Machine

Original characters

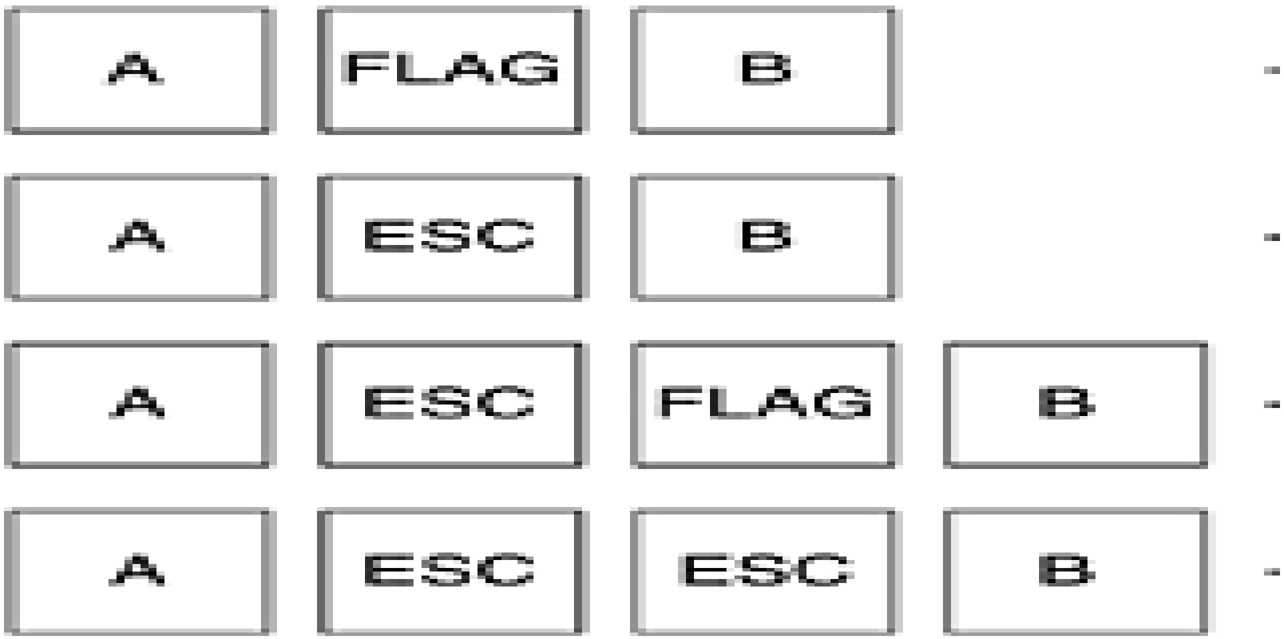
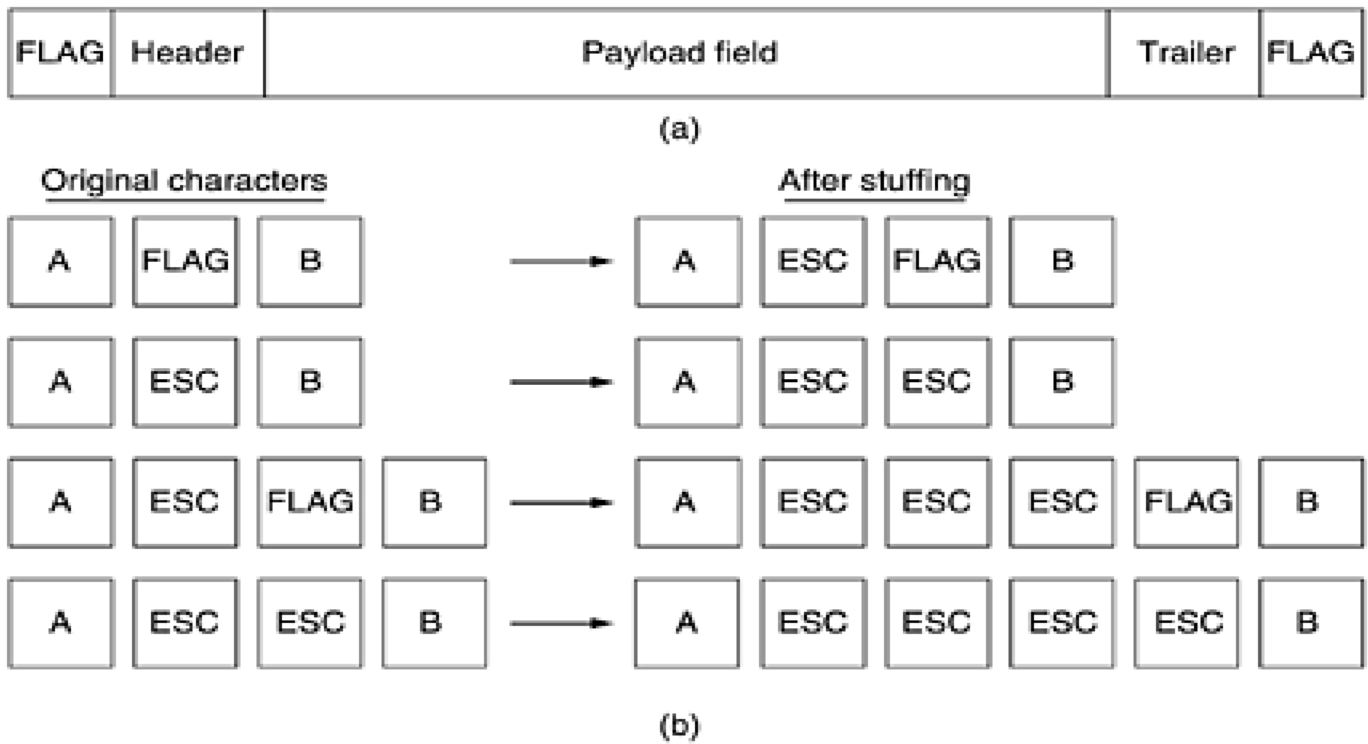


Figure 3-5. (a) A frame delimited by flag bytes. (b) Four examples of byte sequences before and after byte stuffing.

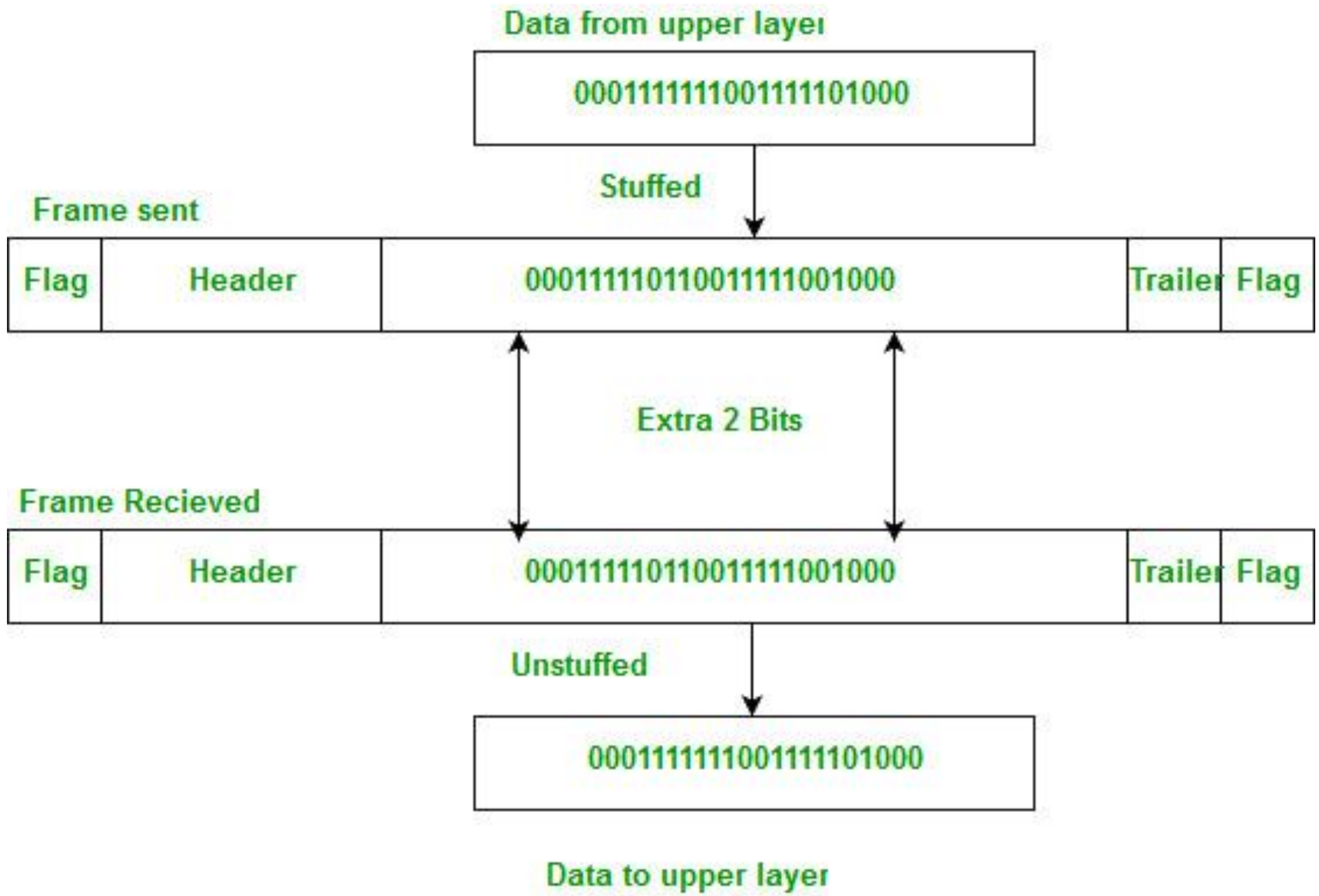


- A serious problem occurs with this method when binary data are being transmitted and the flag byte's bit pattern occurs in the data. This situation will usually interfere with the framing.
- One way to solve this problem is to have the sender's data link layer insert a special escape byte (ESC) just 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.

- Of course, the next question is: What happens if an escape byte occurs in the middle of the data?
- The answer is that it, too, is stuffed with an escape byte. Thus, any single escape byte is part of an escape sequence, whereas a doubled one indicates that a single escape occurred naturally in the data.
- In all cases, the byte sequence delivered after destuffing is exactly the same as the original byte sequence.
- A 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. For example. UNICODE uses 16-bit characters.
- So a new technique is used that allow arbitrary sized characters.

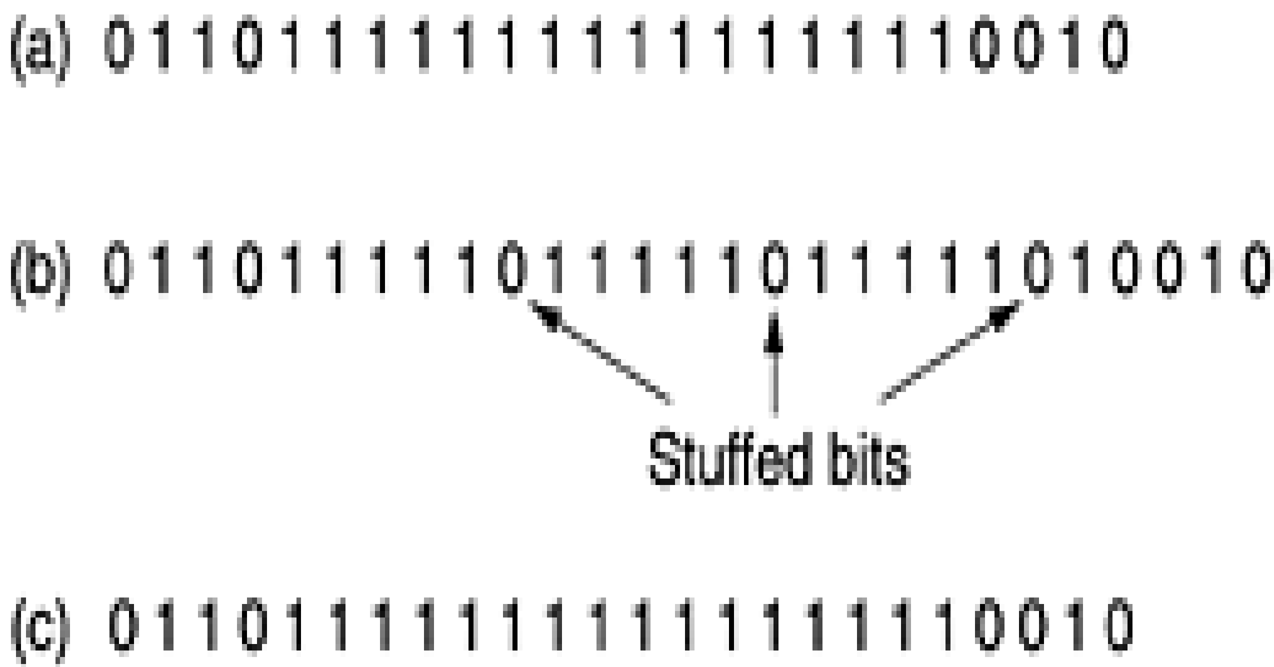
Starting and ending flags, with bit stuffing.

- The new technique allows data frames to contain an arbitrary number of bits and allows character codes with an arbitrary number of bits per character.
- It works like this. 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 0 bit, it automatically destuffs (i.e., deletes) the 0 bit.
- If the user data contain the flag pattern, 01111110, this flag is transmitted as 011111010 but stored in the receiver's memory as 01111110.



Data from upper layer
0001111111001111101000
Stuffed
Frame sent
Flag
Header
000111110110011111001000
Trailer
Flag
Extra 2 Bits
Frame Recieved
Flag
Header
000111110110011111001000
Trailer
Flag
Unstuffed
0001111111001111101000
Data to upper layer

- **Figure 3-6. 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 recognized by the flag pattern.
- As a final note on framing, many data link protocols use a combination of a 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.

Error Control

- Make sure all frames are 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.
- Typically, the protocol calls for the receiver to send back **special control frames** bearing **positive or negative** acknowledgements about the incoming frames.
- 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.

- An additional complication comes from the possibility that hardware troubles may cause a frame to vanish completely.
- In this case, the receiver will not react at all, since it has no reason to react
- This possibility is dealt with by introducing **timers** into the data link layer.
- If either the frame or the acknowledgement is lost, the timer will go off, alerting the sender to a potential problem. The obvious solution is to just transmit the frame again.
- However, when frames may be transmitted multiple times there is a danger that the receiver will accept the same frame two or more times and pass it to the network layer more than once.
- To prevent this from happening, it is generally necessary to **assign sequence numbers** 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 ensure that 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 data link layer's duties.

Flow Control

- Another important design issue that occurs in the data link layer is what to do with a sender that systematically wants to transmit **frames faster than the receiver can accept them**.
- This situation can easily occur when the sender is running on a fast (or lightly loaded) computer and the receiver is running on a slow (or heavily loaded) machine.
- The sender keeps pumping the frames out at a high rate until the receiver is completely swamped.
- Even if the transmission is error free, at a certain point the receiver will simply be unable to handle the frames as they arrive and will start to lose some.
- Two approaches are commonly used.

❑ In the first one, **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.

❑ In the second one, **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.