Design Issues and Framing techniques in Data Link Layer

Dr. G. Omprakash

Assistant Professor, ECE, KLEF

Aim of the session

To familiarize students with the basic concept of Data link layer and its design issues

Learning Outcomes

At the end of this session, you should be able to:
Describe the design issues of the DLL
Framing techniques in DLL

Data Link Layer

Functions of the Data Link Layer

- Provides a **well-defined service** interface to the network layer
- **Framing**: Convert Packets from Network layer to frames
- **Error control**: Detecting and correcting transmission errors
- **Flow control**: Regulating the flow of data so that slow receivers are not swamped by fast senders

Services Provided to the Network Layer

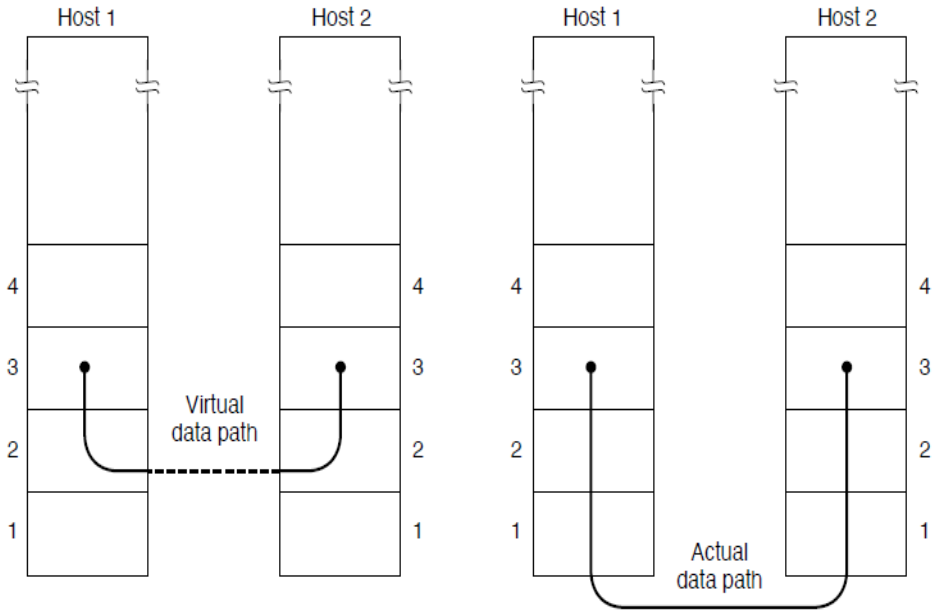


Figure: Virtual and Actual Communication

- Principal service: Transfer data from the network layer on the source machine to the network layer on the destination machine

Services Provided to Network Layer

Services vary from protocol to protocol. Three possible services are

- **Unacknowledged connectionless service**
  - Source machine sends frames to destination machine without having the destination machine acknowledge them
  - No logical connection is established beforehand
  - This class of service is appropriate when the error rate is very low
  - Example: Ethernet (No logical connection is established beforehand)
- **Acknowledged connectionless service**.
  - No logical connections are used
  - Each frame sent is individually acknowledged.
  - This service is useful over unreliable channels
  - Example: 802.11 (WiFi)

Services Provided to Network Layer

- Network layer can send a packet and wait for Acknowledgement.
- Send a large packet that is broken up into, say, ten frames, of which two are lost on average.
- If no acknowledgement comes within some time, send the packet (ten frames) again!!
- Instead, if individual frames are acknowledged then errors can be corrected more quickly
- **Acknowledged connection-oriented service**
  - In this service, the source and destination machines establish a connection before any data are transferred
  - Each frame sent over the connection is numbered
  - DLL guarantees that all frames are received in the right order
  - Example: Unreliable links such as a satellite channel or a long-distance telephone circuit.

Framing

A good design must make it easy for a receiver to find the start of new frames

- Byte count
- Flag bytes with byte stuffing.
- Flag bits with bit stuffing
- Physical layer coding violations.

Framing

Byte count

- Use field in the header to specify the number of bytes in the frame.
- Data link layer at the destination sees the byte count and knows where the frame ends
- Due to a single bit flip (5 to 7), the destination will get out of synchronization
- checksum is incorrect so the destination knows that the frame is bad
- After loosing sync, there is no way of telling where the next frame starts
- Send the entire bit stream again from beginning!!
- Byte count method is rarely used

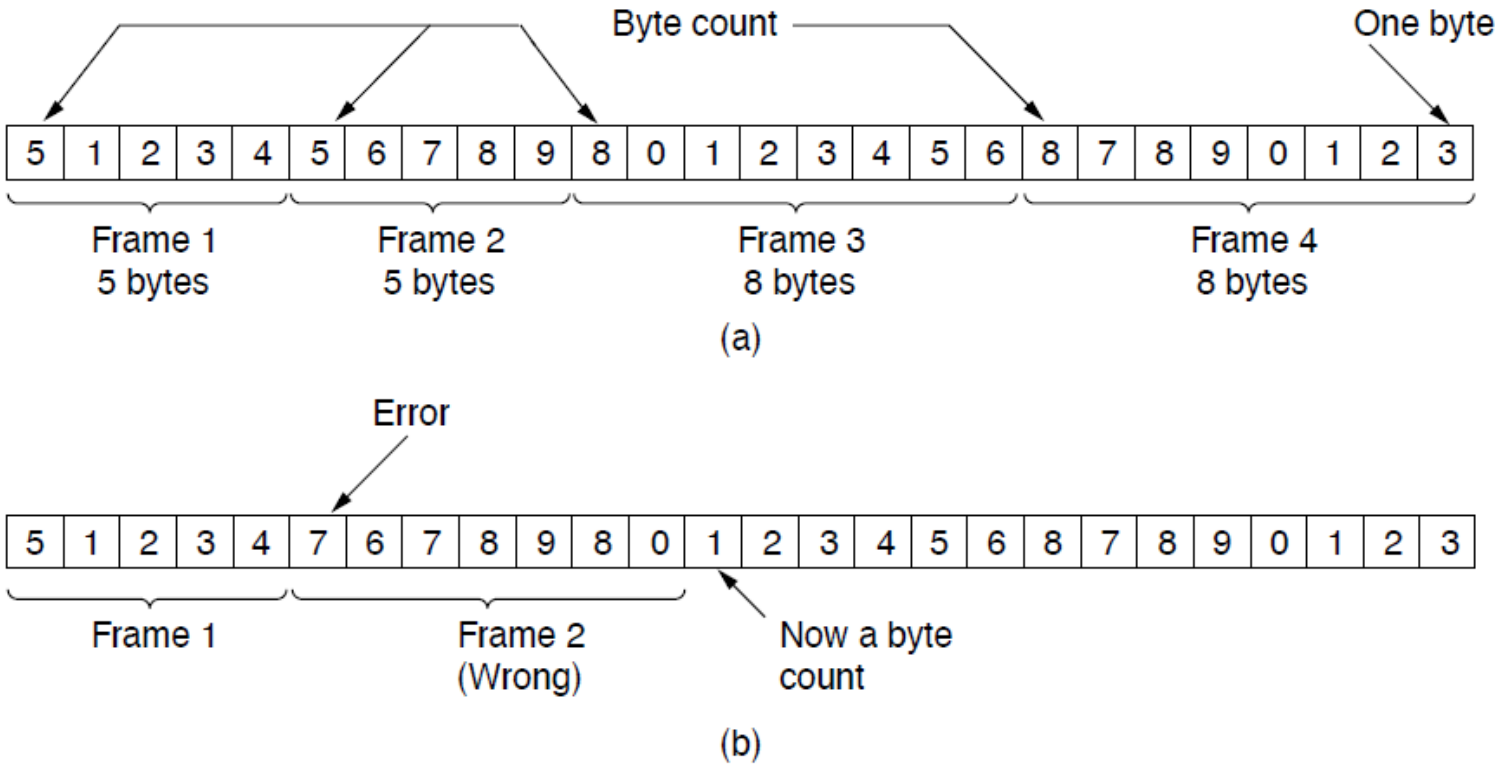


Figure: (a) Without errors (b) With one error

Flag bytes with byte stuffing

Each frame start and end with special bytes called **flag byte**

- Flag byte is used as both the starting and ending delimiter
- Two consecutive flag bytes indicate the end of one frame and the start of the next.
- if the receiver ever loses synchronization, it can just search for two flag bytes to find the end of the current frame and the start of the next frame.

Challenge Flag byte occurs in the data (photos or songs).
Byte stuffing is the process of adding one special escape byte whenever there is a flag byte in the text

- Presence of ESC byte $\Longrightarrow$ flag byte is data
- Absence of ESC byte $\Longrightarrow$ flag byte is frame delimiter

The data link layer on the receiving end removes the escape bytes before giving the data to the network layer

Byte stuffing

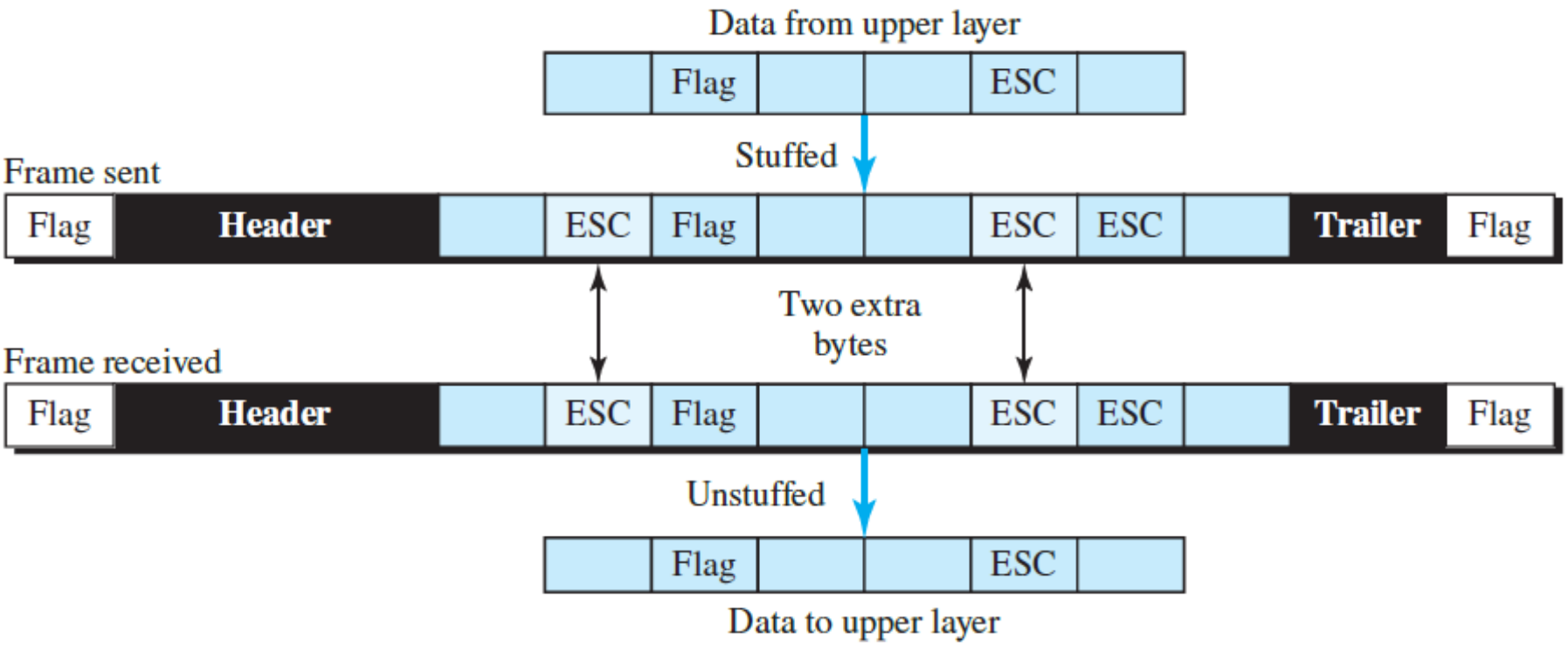
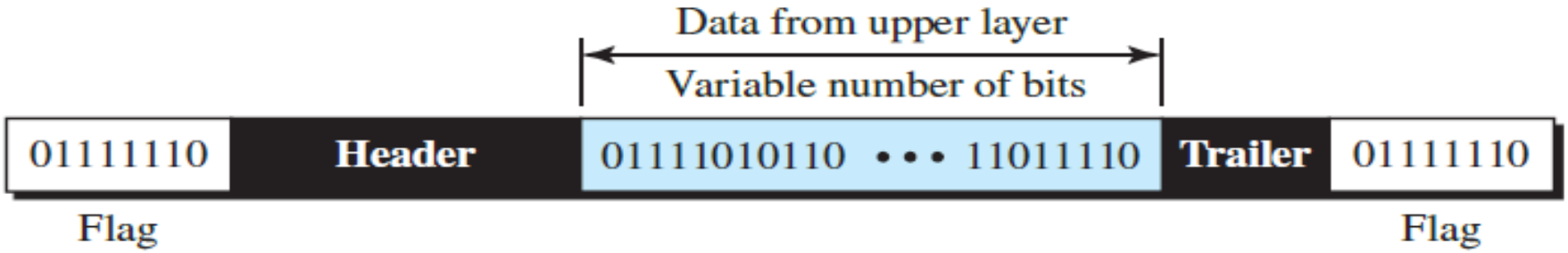


Figure: (a) Frame delimited by flag bytes (b) Byte sequences before and after byte stuffing.

Bit-Oriented Framing

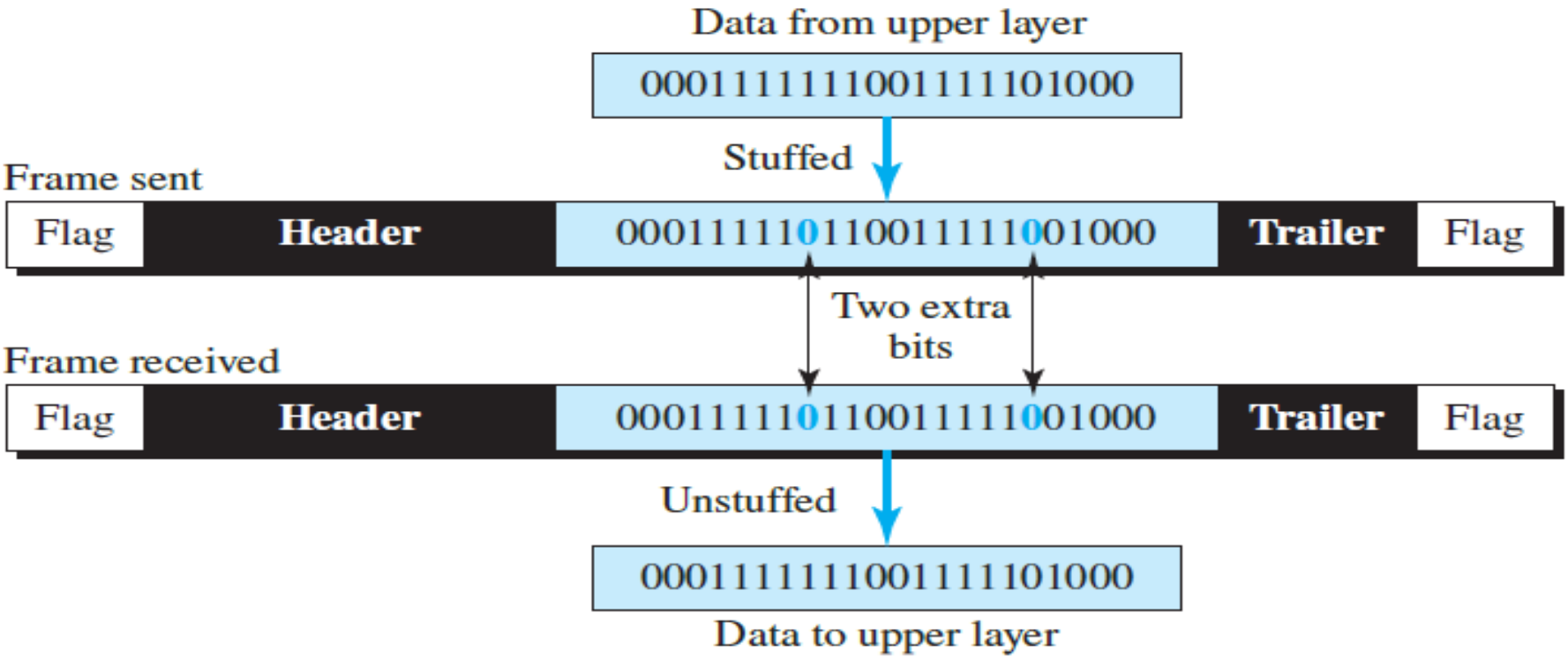
Protocols use a special 8-bit pattern flag, 01111110, as the delimiter to define the beginning and end of the frame



If the flag pattern appears in the data, we need to somehow inform the receiver that this is not the end of the frame

Bit stuffing

Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern 0111110 for a flag.



Physical layer coding violations

- Encoding of bits as signals often includes redundancy to help the receiver.
- 4 data bits are mapped to 5 signal bits
- This means that 16 out of the 32 signal possibilities are not used.
- Use *invalid characters* or **coding violations** to indicate the start and end of frames.

Error Control

- How to make sure all frames are delivered to the network layer at the destination and in the proper order?
- For **reliable delivery**: provide the sender with some **feedback (Positive/Negative)**
- Complication 1 : Possibility that a frame may vanish completely (noise burst)-Rx will not react at all
- Complication 2: If the acknowledgement frame is lost, the sender doesn't know how to proceed
- **Introduce timers**: long enough for the frame to reach the destination and have the acknowledgement propagate back
- When frames may be transmitted multiple times receiver will accept and pass it to the network layer more than once
- It is necessary to **assign sequence numbers** to outgoing frames
- Managing the timers and sequence numbers so as to ensure that each frame is ultimately passed to the network layer

Flow Control

What to do with a sender that systematically wants to transmit frames faster than the receiver can accept them?
Two approaches

- **Feedback-based flow control**: The receiver sends back information to the sender
- **Rate-based flow control:** This protocol has a built-in mechanism that limits the rate at which senders may transmit data, without using feedback from the receiver

Acknowledge various sources for the images.
Thankyou