DATA LINK LAYER DESIGN ISSUES
(FUNCTIONS OF DLL)

1. FRAME SYNCHRONISATION
2. FLOW CONTROL
3. ERROR CONTROL
4. PHYSICAL ADDRESSING
5. NETWORK TOPOLOGY

FRAME SYNCHRONISATION — The Source machine sends data in blocks called Frames to the destination machine. The starting & ending of each frame should be recognised by the destination machine

FLOW CONTROL — The Source machine must not send data frames at a rate faster than the destination machine can accept them

ERROR CONTROL — The errors made in bits during Transmission from Source to Destination machines must be detected & corrected.
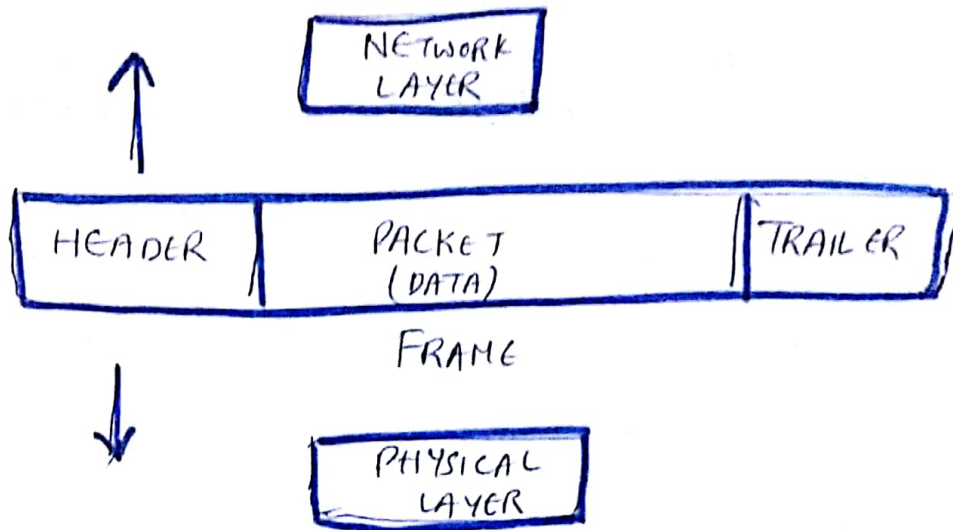
PHYSICAL ADDRESSING — On a Multipoint line, Such as many machines connected together (LAN), the identity of the individual machines must be specified while transmitting the data frames.

NETWORK TOPOLOGY — It is defined by DLL.
Example — TOKEN RING, BUS TOPOLOGY, STAR TOPOLOGY

FRAMING IN DATA LINK LAYER

- FRAMING is a Point-to-Point connection b/w Two computers in which data is transmitted as a stream of bits.

```
              ┌─────────────┐
              │  NETWORK    │
              │   LAYER     │
              └─────────────┘
                   ↑
┌──────────┬────────────────┬───────────┐
│ HEADER   │    PACKET      │  TRAILER  │
│          │    (DATA)      │           │
└──────────┴────────────────┴───────────┘
                  FRAME
                   ↓
              ┌─────────────┐
              │  PHYSICAL   │
              │   LAYER     │
              └─────────────┘
```

- At data link layer, it extracts messages from sender & provide it to receiver by providing sender's & receiver's address.

PROBLEMS IN FRAMING

- Detecting Start of the Frame
- Detecting end of the Frame

TYPES OF FRAMING

① FIXED SIZE

- The frame is of fixed size & there is no need to provide boundaries to the Frame.

- The length of the frame itself acts as a delimiter

Drawback — Suffers internal Fragmentation if data size is less than Frame Size

SOLUTION — PADDING

② VARIABLE SIZE

- In this there is need to define end of frame as well as begining of next frame.

  Length field   End delimiter
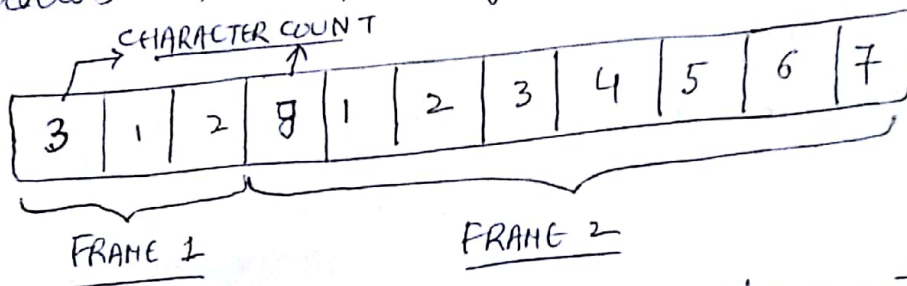
FRAMING METHODS

① CHARACTER COUNT ② CHARACTER STUFFING ③ BYTE STUFFING ④ BIT STUFFING
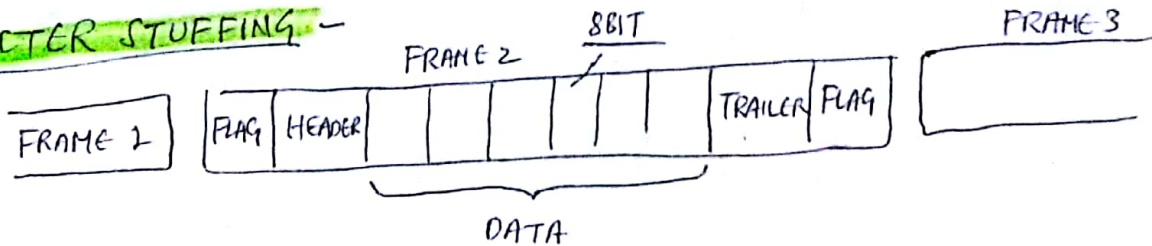
CHARACTER COUNT

- In This method, a field in the header is used to specify no, of characters in the frame.
- This number helps the receiver to know the no, of characters in the frame following this count.

```
        CHARACTER COUNT
      ┌─────────────────┐
   →  | 3 | 1 | 2 | 9 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
      └───────────────┘ └───────────────────────┘
         FRAME 1              FRAME 2
```

CHARACTER COUNT → above 3 1 2 9 1 2 3 4 5 6 7

DISADVANTAGE - Error can change the character count

So NEXT SAME ↓

② CHARACTER STUFFING -

```
              FRAME 2          8 BIT              FRAME 3
  ┌────────┐ ┌────┬──────┬──┬──┬──┬──┬──┬──────┬────┐ ┌────────┐
  │ FRAME 1│ │FLAG│HEADER│  │  │  │  │  │TRAILER│FLAG│ │        │
  └────────┘ └────┴──────┴──┴──┴──┴──┴──┴──────┴────┘ └────────┘
                        └──────────────┘
                             DATA
```

- The Problem of character count is solved by adding Flag (8 BIT) at the start f end of Header & Trailer respectively.
- But what if, the bit Pattern of flag is Same as the data bits inside the data
- This Problem is solved by using a technique called **BYTE STUFFING**

③ BYTE STUFFING

- A BYTE (usually Escape character (ESC)), which has a predefined bit Pattern is added to the data section of the frame when there is a character with the same Pattern as the flag.

- Whenever the receiver encounter's the ESC character, it removes from the data section & treats the next character as data, not a flag.

BUT

- Problem arises when text contains one or more escape character's followed by a Flag.

- To solve this problem, escape character's that are Part of the text are marked with another escape character ie if the escape character is part of the text, an extra one is added to show that the second one is Part of the text
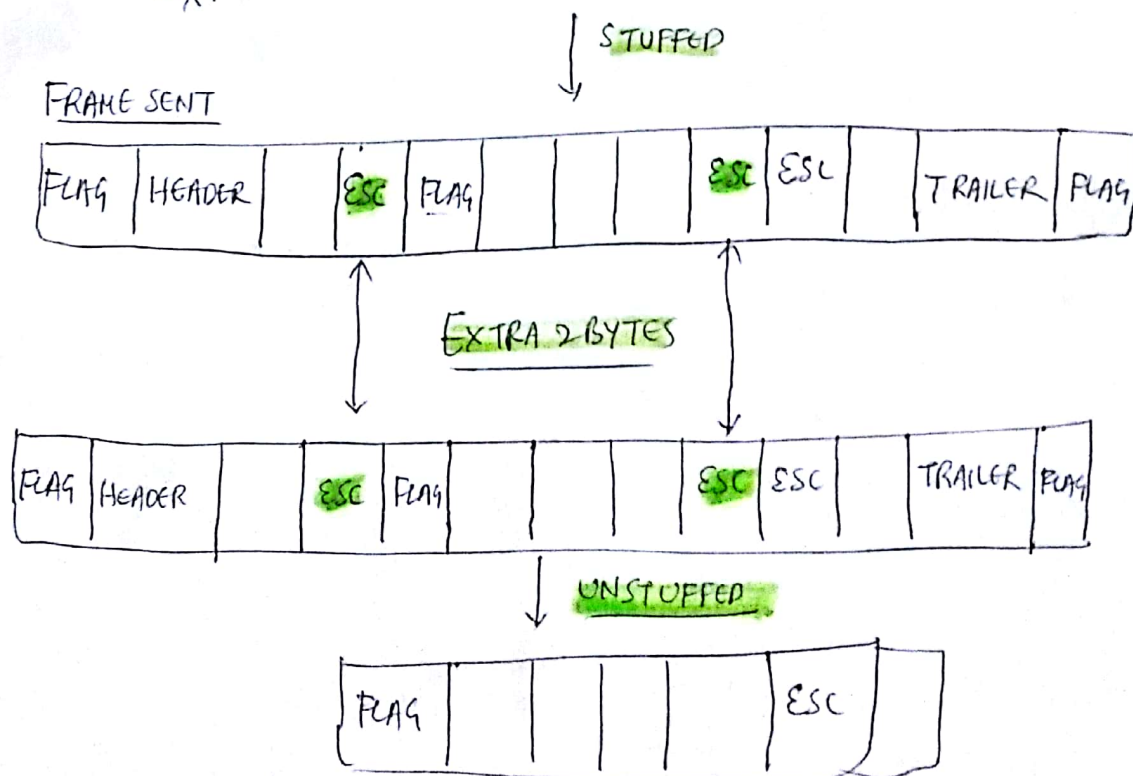
NOTE - POINT-TO-POINT PROTOCOL (PPP) is a byte oriented Protocol

BIT STUFFING -

- Mostly flag is a special 8-bit Pattern '01111110' used to define the begining & the end of the frame.

- Problem with the flag is same as that was in case of Byte stuffing. So in this Protocol what we do is, if we encounter 0 & five consecutive 1 bits, an extra 0 is added after these bits. This extra stuffed bit is removed from the data by the receiver.

- The extra bit is added after one 0 followed by five bits regardless of the value of the next bit.

- Also, as sender side always knows which sequence is data & which is flag it will only add this extra bit in the data sequence not in flag sequence.

DATA FROM UPPER LAYER

```
0 0 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 1 0 0 0
```

FRAME SENT   STUFFED

| FLAG | HEADER | 0 0 0 1 1 1 1 1 0 1 1 0 0 1 1 1 1 1 0 0 1 0 0 0 | TRAILER | FLAG |

EXTRA 2 BITS

FRAME RECEIVED

| FLAG | HEDER | 0 0 0 1 1 1 1 1 0 1 1 0 0 1 1 1 1 1 0 0 1 0 0 0 | TRAILER | FLAG |

UNSTUFFED

```
0 0 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 1 0 0 0
```

DATA TO UPPER LAYER