UNIT-III

The Data Link Layer

Data link layer Design Issues

i. Services provided to the n/w layer
ii. Framing
iii. Error Control
iv. Flow Control

i. Services provided to the n/w layer

- Unacknowledged Connectionless Service
- Acknowledged Connectionless Service
- Acknowledged Connection-oriented Service.

(a) Acknowledged Connectionless Service: It consists of having the source machine send independent frames to the destination machine without having the destination machine acknowledge them.

Ex:- Ethernet.

- If a frame is lost due to noise, no attempt is made to detect the loss or recover from it.
- This service is used when the error rate is very low.

(b) Acknowledged Connectionless Service

- There are no logical connections used b/w the sender and the receiver.

- Each and every frame sent is individually acknowledged.
- So, the sender knows whether a frame has arrived correctly or been lost.
- If it has not arrived within a specified time interval it can be sent again.
- This service is useful over reliable channels such as wireless channels Eg:- 802.11 (wifi).

(c) Acknowledged Connection-Oriented Service:

- In this service, source and destination establish a connection before any data is transferred.
- Each frame sent over the connection is numbered, & the data link layer guarantees that each frame sent is received.
- It guarantees that each frame is received exactly once and that all frames are received in right order.
- This service is used over long, unreliable links such as a satellite channel or long-distance telephone circuit.

(ii) Framing: The DLL should detect and correct the errors.

- For this purpose, DLL will break up the bit stream into discrete frames, compute a short token called a checksum for each frame & include the checksum in the frame when it is transmitted.

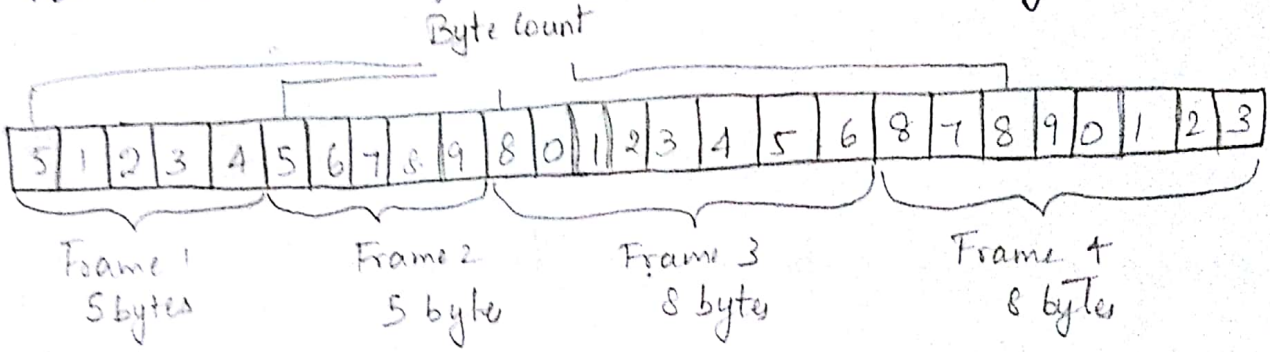
- When the frame arrives at the destination, the checksum is recomputed.
- If the newly computed checksum is different from the one contained in the frame, DLL finds that error has occured & retransmits the frame.
- After dividing the data into frames, we should be able to identify the starting & ending of each frame.
- There are four methods for this purpose:

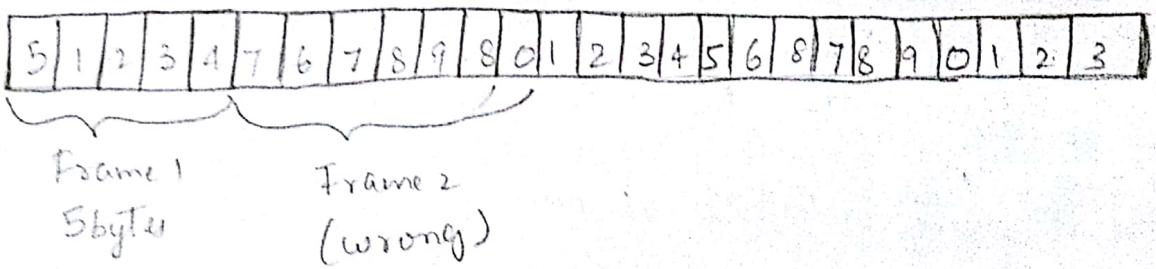
Framing methods
- Byte Count
- Flag bytes with byte stuffing
- Flag bits with bit stuffing
- Physical layer coding violations

(a) Byte Count :- This method uses a field in the header to specify the number of bytes in the frame.
- When the DLL at the destination sees the byte count, it knows how many bytes follow & hence where the end of the frame is.
- This problem occurs if the byte count is changed by any transmission error.

Ex:- if the byte count of 5 becomes 7 due to error, the destination will get out of synchronization.
- It will be unable to locate the correct start of next frame.

- Using checksum destination determines the error has occured, but retransmission is not possible since we are unable to locate the correct start of the frame.
- For this reason, this method is rarely used.



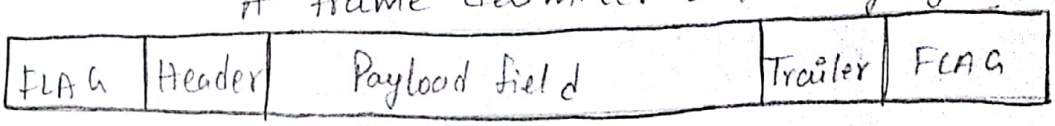


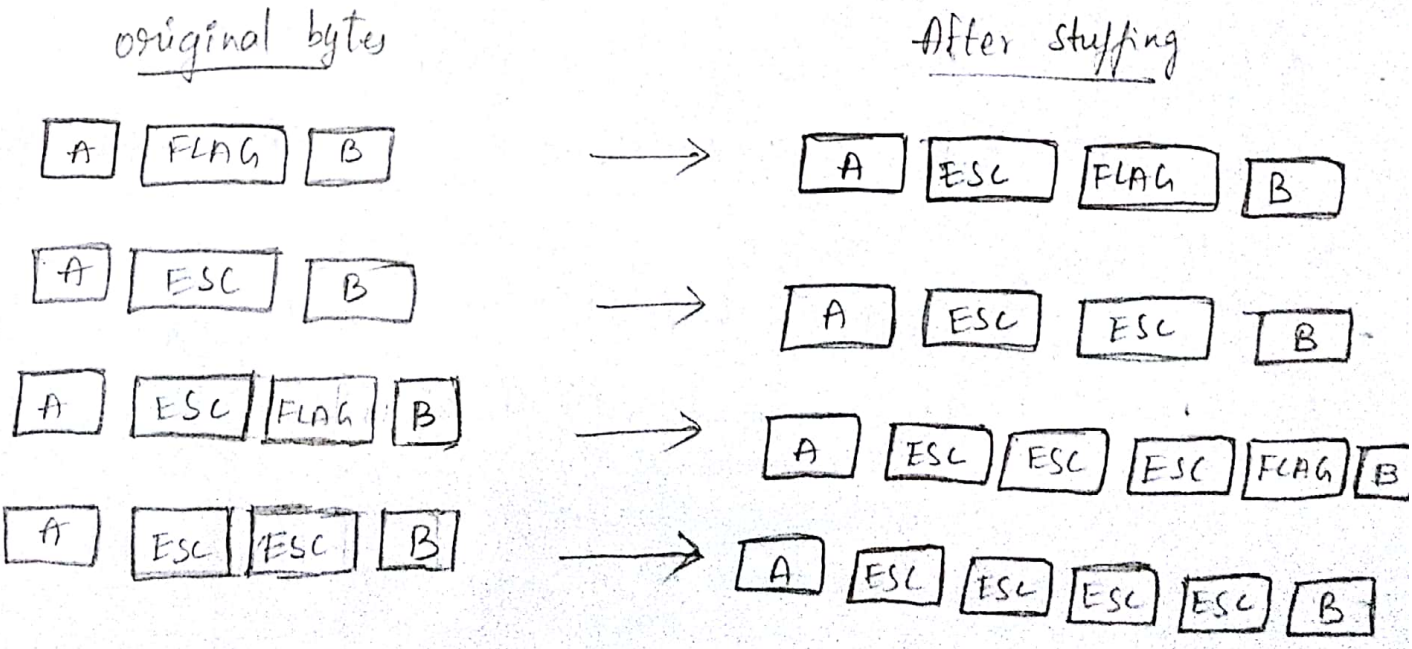
(b) Flag bytes with byte stuffing:

- In this method, a special byte called flag byte is used as both the starting & ending delimiter of each frame.
- Two consecutive flag bytes indicate the end of one frame and the start of the next.
- If the receiver looses synchronization, it can just search for two flag bytes to find the end of current frame & the start of the next frame.

- There may be a situation in which the flag byte occurs in the data.
- One way to solve this problem is to insert a special escape byte (ESC) just before each flag byte in the data.
- Thus, a framing flag byte can be distinguished from the flag byte in the data by the absence or presence of escape byte before it.
- The DLL on the receiving end removes the escape bytes before giving the data to the N/w Layer.
- This technique is called byte stuffing.

A frame delimiter with flag bytes



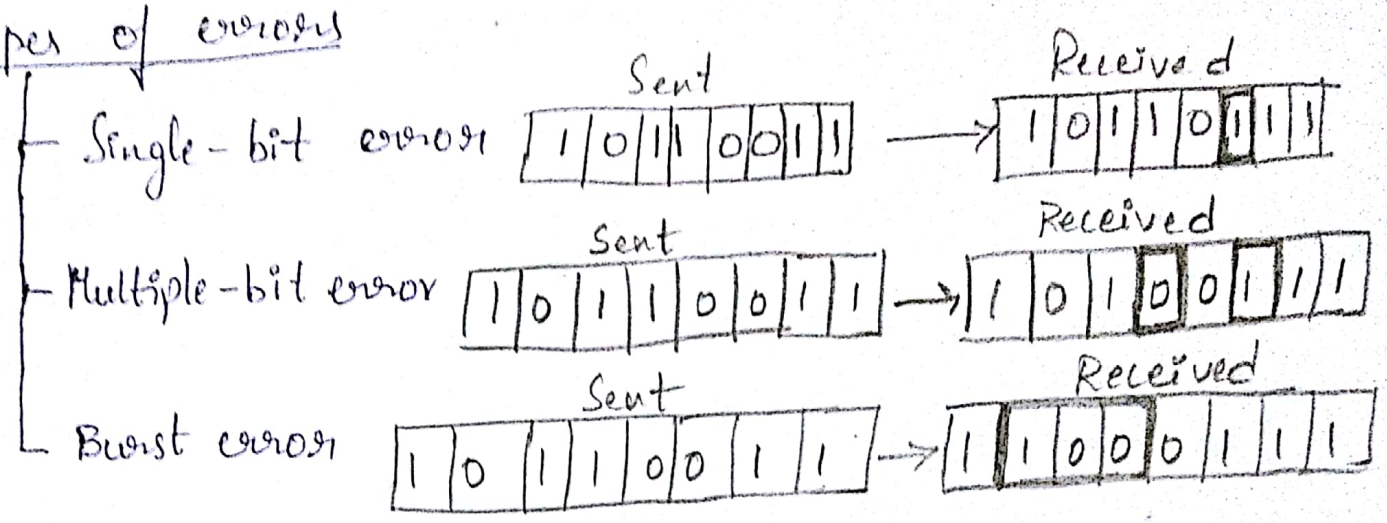


Four examples of byte sequences before & after byte stuffing

(iii) Error Control:

- To ensure reliable delivery, the sender should be provided with some feedback about what is happening at the receiver.
- For this purpose, receiver sends special control frames having positive or negative acknowledgement.
- If the sender receives positive acknowledgement, it means that the frame has transmitted safely.
- If the sender receives negative acknowledgement, it means the frame is lost and the sender must retransmit the frame.
- But if the ACK frame is lost, the sender indefinitely waits for +ve/-ve ACK & may hang forever.
- To overcome this, timers are used in DLL
- When the sender transmits a frame, it also starts a timer.
- The timer is set to the time interval required for the data to reach the destination and the ACK to reach the source.
- If the timer expires, it means that either the frame is lost or ACK is lost, then the sender retransmits the frame.
- Sequence numbers are used to distinguish b/w the original frame & the retransmitted frame.

- Types of errors



(iv) Flow Control: Flow is controlled by sending the data according to the capability of the receiver. There are two ways:-

(a) Feedback-based flow control: In this, the receiver sends some feedback to the receiver. This feedback includes:-

- when to send the data.
- how much data the sender can transmit.
- at what rate data can be transmitted.

(b) Rate-based flow control: In this, there is a built-in mechanism that limits the rate at which senders can transmit data, without using feedback from the receiver.

Error Detection & Correction :

- Error Correcting codes
- Error Detecting Codes

Error - Correcting codes :

- Hamming codes
- Binary convolutional codes
- Reed - Solomon codes
- Low - Density Parity check codes.

Hamming codes :

Hamming distance : Consider two codewords

10001001 & 10110001

it is possible to determine how many corresponding bits differ. To determine how many bits differ, XOR the two codewords and count the number of '1' bits in the result.

Ex:-

```
  1 0 0 0 1 0 0 1
  1 0 1 1 0 0 0 1
-----------------
  0 0 1 1 1 0 0 0
```

3 bits differ (No of 1's = 3)

XOR

0	0	0
0	1	1
1	0	1
1	1	0

- The number of bit positions in which two codewords differ is called the Hamming distance.

- It means that if two codewords are at a Hamming distance 'd' apart, it will require 'd' single-bit errors to convert one into the other.

Codeword ⟶ m + r
(message bits) (check/redundant bits)

- The number of check bits is calculated using the relation: $(m+r+1) \leq 2^r$

Ex:- m = 1 1 0

$\Rightarrow (m+r+1) \leq 2^r$

$\Rightarrow (3+r+1) \leq 2^r$

$\Rightarrow (4+r) \leq 2^r$ if $r=3$,

$\Rightarrow 7 \leq 2^3$

$\Rightarrow \boxed{7 \leq 8} \Longrightarrow \boxed{r=3} \Rightarrow$ r_1 r_2 r_3
2^0 2^1 2^2

Codeword $\Rightarrow$ m + r

$\Rightarrow$ 3+3 = 6

Hamming code

1	2	3	4	5	6
r_1	r_2	M_1	r_3	M_2	M_3
2^0	2^1	1+2	2^2	1+4	2+4

$\Rightarrow$

1	2	3	4	5	6
r_1	r_2	1	r_3	1	0

r_3	r_2	r_1	
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6

$\Rightarrow r_1 = 3+5 = M_1 + M_2 = 1 \text{ XOR } 1 = 0$

$\boxed{r_1 = 0}$

$\Rightarrow r_2 = 3+6 = M_1 + M_3 = 1 \text{ XOR } 0 = 1$

$\boxed{r_2 = 1}$

$\Rightarrow r_3 = 5+6 = M_2 + M_3 = 1 \text{ XOR } 0 = 1$

$\boxed{r_3 = 1}$

Hamming code $\Rightarrow$ 011110 $\Rightarrow$ transmitted.

$r_1\ r_2\ M_1\ r_3\ M_2\ M_3$

(1) If the codeword is received as 010110

(data is received with one-bit error)

Calculate r_1, r_2, r_3 (check bits)

$r_1 = M_1 + M_2 = 0 \text{ XOR } 1 = 1$ ✗ r_1 is wrong. [In receive data, $r_1 = 0$]

$r_2 = M_1 + M_3 = 0 \text{ XOR } 0 = 0$ ✗ r_2 is wrong

$r_3 = M_2 + M_3 = 1 \text{ XOR } 0 = 1$

So, error is detected in the codeword.

$r_1\ r_2\ M_1\ r_3\ M_2\ M_3$

(2) If the received codeword is 011100

$r_1 = M_1 + M_2 = 1 \text{ XOR } 0 = 1$ ✗ {In received data, $r_1 = 0$ So r_1 is wrong.]

$r_2 = M_1 + M_3 = 1 \text{ XOR } 0 = 1$ ✓

$r_3 = M_2 + M_3 = 0 \text{ XOR } 0 = 0$ ✗ r_3 is wrong.

So, the codeword is received with an error.

Hence error is detected.

Error Detecting codes:

- Error Correcting codes are used when the error rate is low.
- Error Detecting codes are used when the error rate is high.

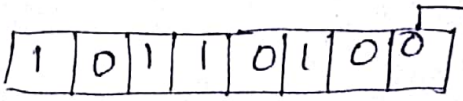
Error-Detecting codes

- Parity
- Checksums
- Cyclic Redundancy Checks (CRCs)

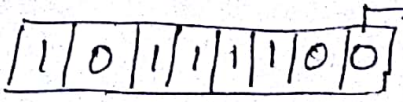
(a) **Parity** :- It can detect single-bit errors.

There are two types :- even parity
odd parity

Data word Original data	even parity	Codeword Transmitted data	odd parity	Codeword transmitted data
1011010	0	10110100	1	10110101
100101	1	10010111	0	10010100

If transmitted data is 

Ex: At the receiver,

If the received data is 

Receiver calculates the parity bit

⇒ parity bit = 1 [Consider even parity]

the transmitted parity bit & receiver calculated parity bit are not equal. So, error is occured.

Cyclic Redundancy Check (CRC):

- Polynomial code : bit strings are representations of polynomials with coefficients of 0 and 1 only.
- When the polynomial code is employed, the sender and receiver must agree upon a generator polynomial G(x).
- The result is a checksummed frame to be Transmitted T(x).

Ex :- Frame M(x) = 1101011111

Generator G(x) = 10011

code to be appended at the end of M(x)

= [(No of bits in G(x)) − 1] = 5 − 1 = 4 (0000)

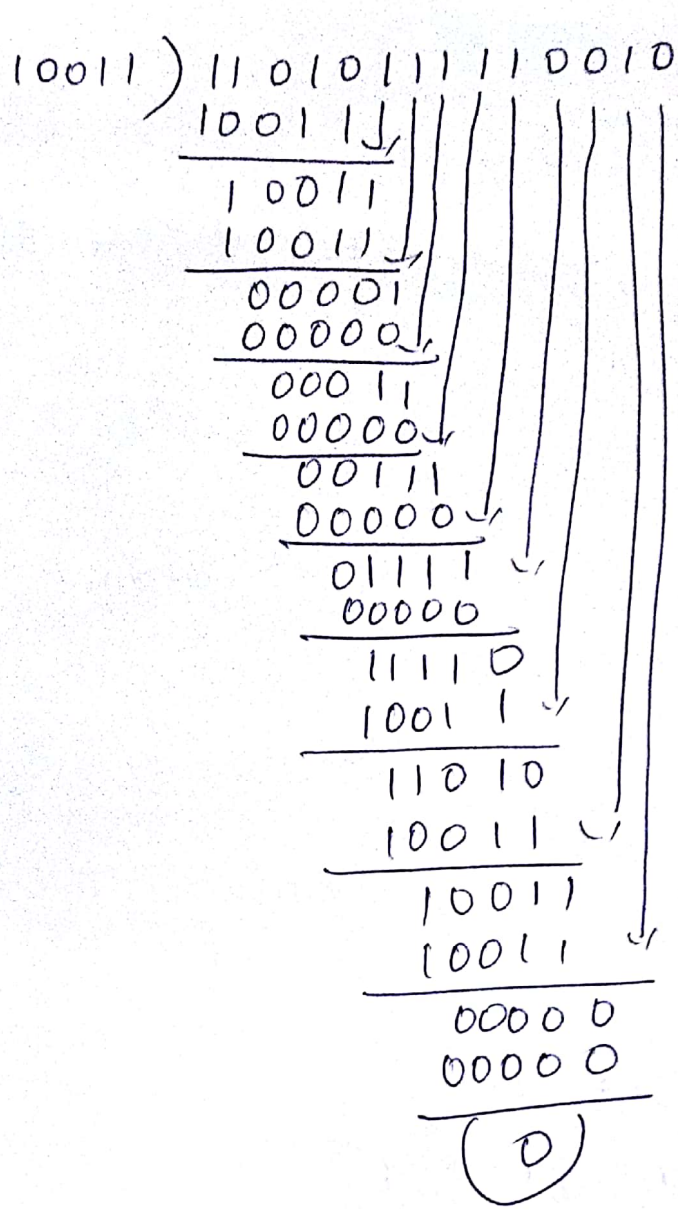
Sender side

```
10011 ) 11010111110000 (110000111
        10011
        -----
         10011
         10011
         -----
          00001
          00000
          -----
           00011
           00000
           -----
            00111
            00000
            -----
             01111
             00000
             -----
              11110
              10011
              -----
               11010
               10011
               -----
                10010
                10011
                -----
                 00010
                 00000
                 -----
                 (0010)
```

Transmitted frame T(x) =

11010111110010

Receiver side :



At the receiver, the above calculation is done. If the remainder is '0'. It means there is no error.

Elementary Data link protocols :

All the design issues/functions of DLL are fulfilled using some protocols. They are :-

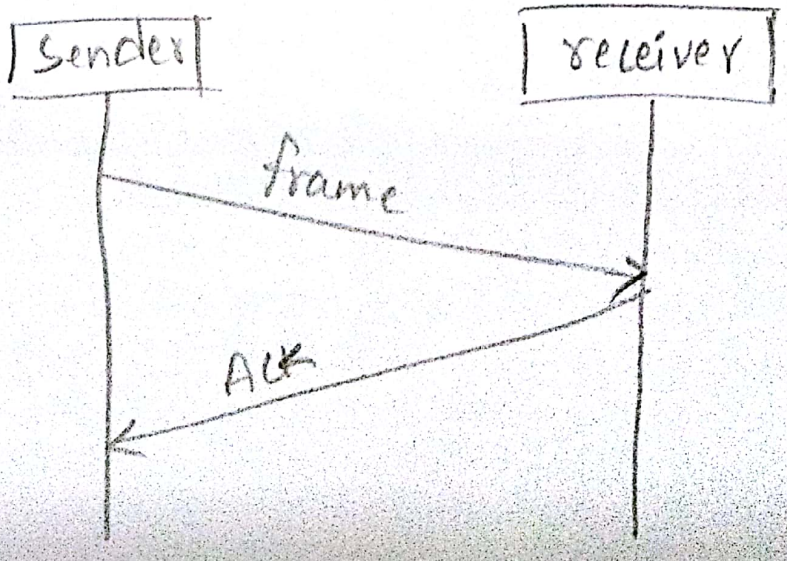
(a) Unrestricted Simplex protocol
(b) Simplex stop-and-wait protocol for an Error-free channel
(c) Simplex stop-and-wait protocol for a Noisy channel.

(a) Unrestricted Simplex protocol :

- It is also called as Utopian Simplex protocol.
- This protocol can be used if the following conditions exist :-
  1. Data are transmitted in one direction only.
  2. Both transmitting (Sender) and receiving (receiver) systems are always ready
  3. Processing time can be ignored.
  4. Infinite buffer space is available
  5. Communication channel never damages or looses frames
  6. No sequence numbers or Acknowledgements are used here.
- This protocol is unrealistic because it doesn't handle either flow control or error control.
- Its processing is close to that of an unacknowledged connectionless service.

(b) Simplex stop-and-wait protocol for an Error-free channel :-

- The communication channel is assumed to be error free.
- The data traffic is half-duplex.
- In this protocol, receiver provides a feedback to the sender.
- It means that when the sender sends the data, the receiver receives it & sends a little dummy frame back to the sender giving permission to the sender to transmit the next frame.
- After having sent a frame, the sender is required by the protocol to wait until the dummy (ACK) frame arrives.
- Protocols in which the sender sends one frame & then waits for an Ack before proceeding to the next frame are called stop-and-wait protocols.



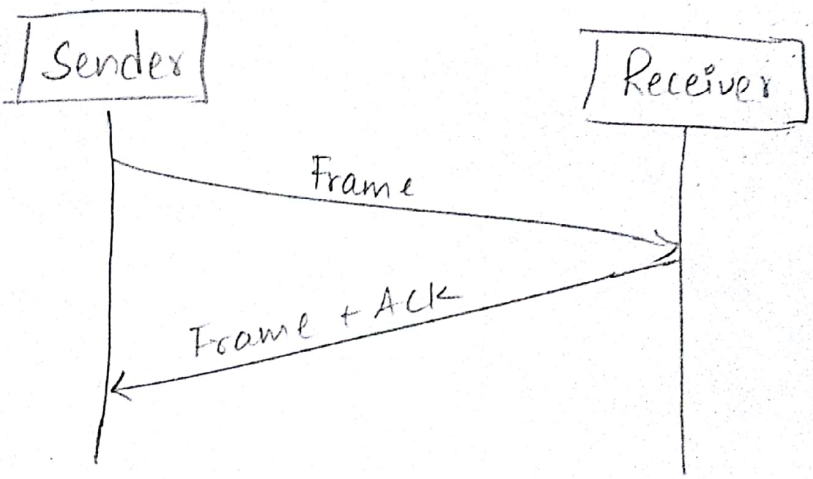
(c) Simplex stop-and-wait protocol for a Noisy channel:

- Consider the communication channel is a Noisy channel (ie., the channel that makes errors).
- Frames may be either damaged or lost completely.
- If a frame is damaged, the error is detected by using the checksum.
- If the data is lost or the ACK is lost, the send it can be identified by using timers.
- When the sender transmits a frame, it also starts a timer.
- If the timer expires, then the sender retransmits the frame.
- Sequence numbers are used to distinguish b/w the original frame & the retransmitted frame.
- Protocols in which the sender waits for a +ve Ack before advancing to the next data item are often called ARQ (Automatic Repeat ReQuest) or PAR (Positive Acknowledgement with Retransmission).

Sliding Window Protocols

Piggybacking: when a data frame arrives, instead of immediately sending a separate ACK, the receiver waits until the next frame.

- The ack is attached to the outgoing data frame.
- The technique of temporarily delaying outgoing acknowledgements so that they can be attached onto the next outgoing data frame is known as piggybacking



adv :- better use of channel bandwidth

disadv :- If the receiver waits too long, then at the sender the timer will be off & the sender retransmits the frame.

- **Sending window** :- At any instant of time, the sender maintains a set of sequence numbers corresponding to the frames it is permitted to send.

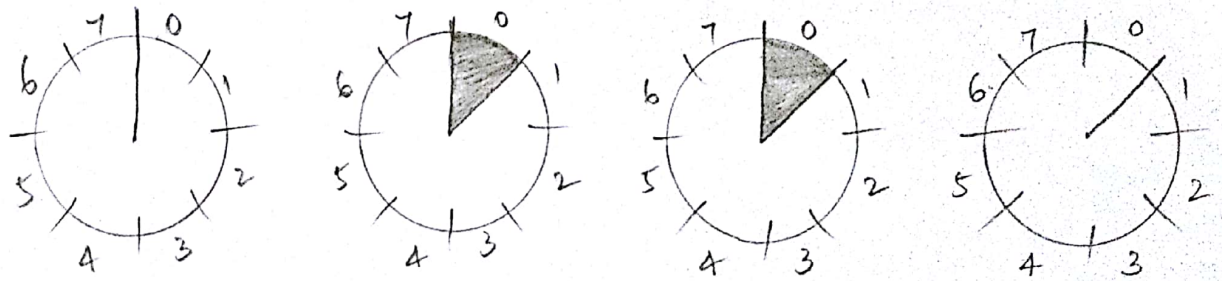
- receiving window : At any instant of time, the receiver maintains a set of frames it is permitted to receive.
- Sequence number range is 0 to $2^n - 1$

Sliding window protocols

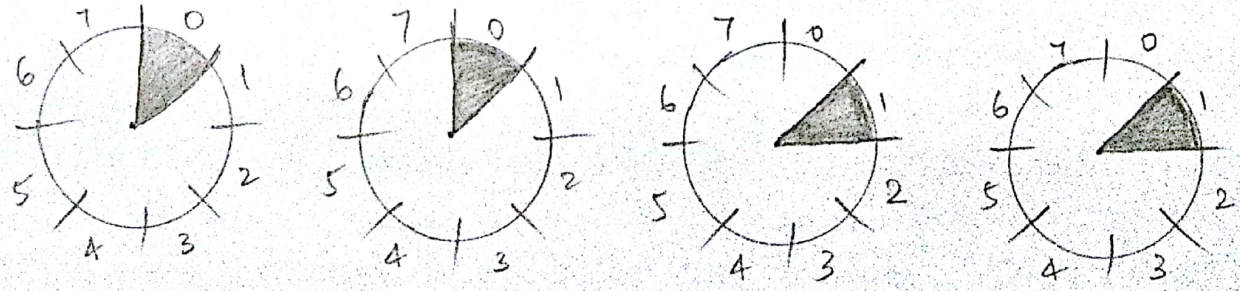
- one-bit sliding window protocol
- A protocol using Go-Back-N
- A protocol using Selective Repeat.

(a) one-bit sliding window protocol :

Sender



Receiver



(a) (b) (c) (d)

- The above example has a sliding window of size 1, with a 3-bit sequence number.
  ↳ sequence number range is 0 to $2^3 - 1$
  = 0 to 7

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

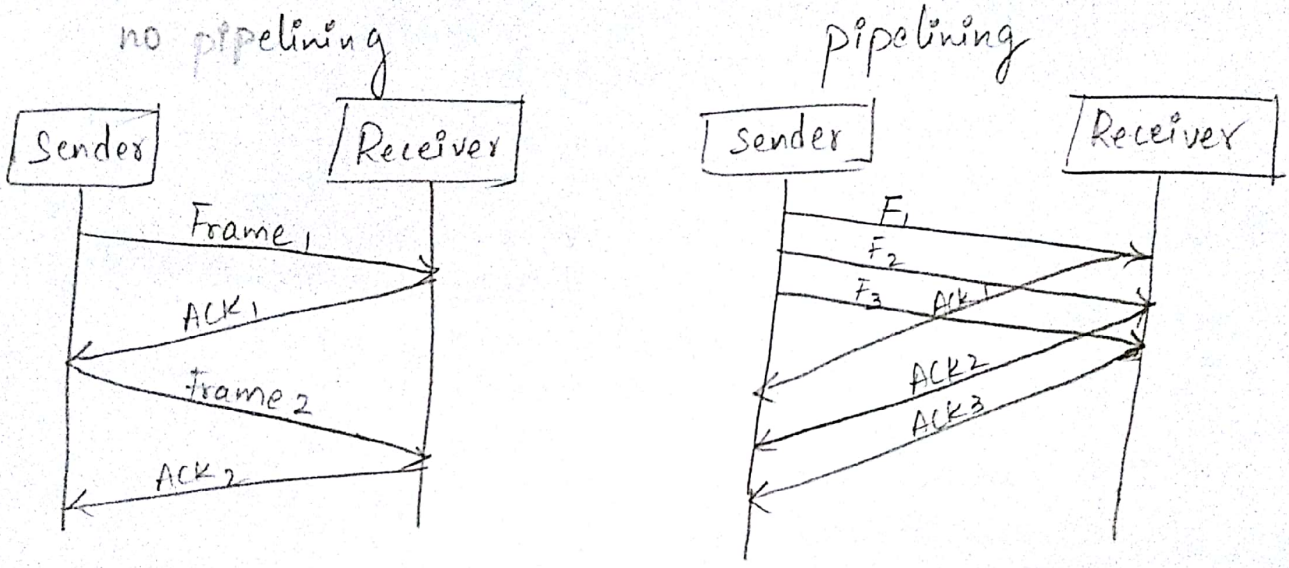
(a) Sender :- Initially when data transmission is not yet started.
Receiver :- waiting for a frame of sequence num = 0.

(b) Sender :- Sender sends a frame of sequence num = 0
Receiver :- waiting for a frame of sequence num = 0

(c) Sender :- Sender waiting for an Ack for frame of seq num = 0
Receiver :- sends the Ack and waits for the next frame of seq num = 1

(d) Sender :- Data transmission of frame with seq num = 0 is completed & not sending any data (idle).
Receiver : waiting for a frame of seq num = 1.

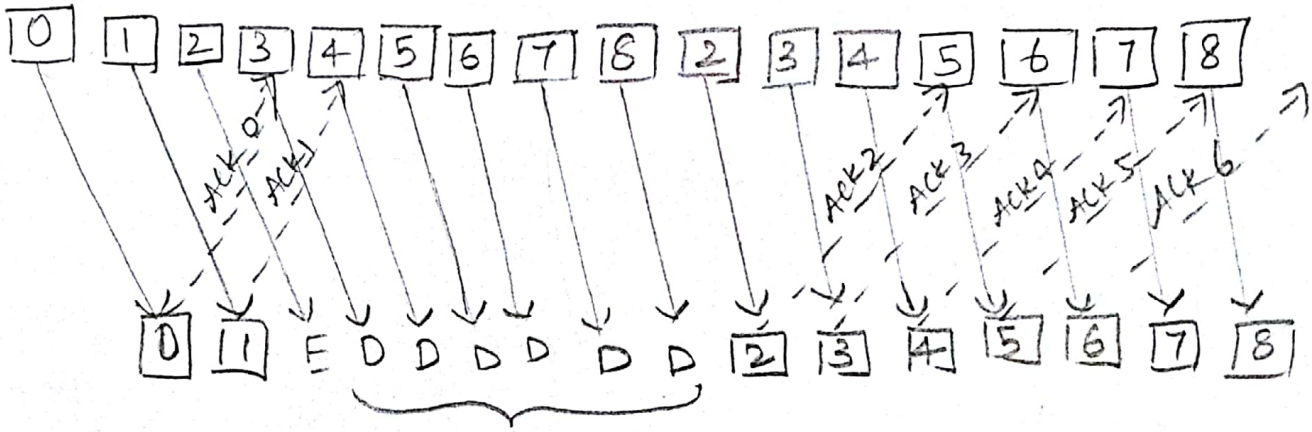
Pipelining :- It is a technique in which multiple frames are sent at a time without waiting for the corresponding individual acknowledgements.



(b) A protocol using Go-Back-N :-

- In this protocol, the sender retransmits all the frames that are transmitted after the damaged/lost frame.
- If error rate is high, it wastes a lot of bandwidth.
- In this protocol, the receiver do not store the frames received after the damaged frame until the damaged frame is retransmitted.
- It is a mechanism to detect & control the errors.
- Go-Back-N protocol is shown in the below diagram.
- Frame 2 is lost, so all the frames followed by frame 2 are deleted (discarded).
- All the frames from frame 2 to 8 are retransmitted.

- Go-Back-N protocol performs pipelining. Hence all the frames from 0 to 8 are sent at a time without waiting for individual acknowledgements.
- The damaged or discarded frames will be retransmitted after all the 8 frames are transmitted.



receiver's window size is 1

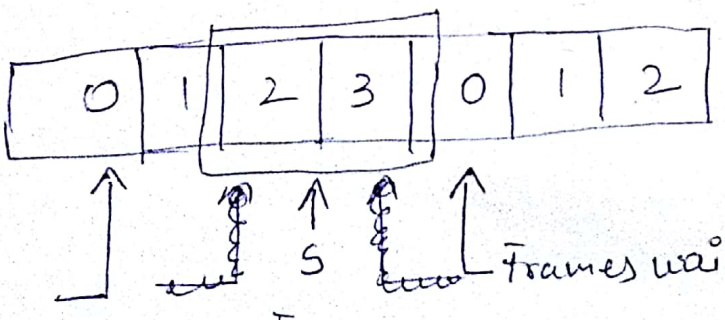
(c) A protocol using selective repeat.

- The go-back-n protocol works well if errors are rare, but if the channel has high error rate, it wastes lot of bandwidth on retransmitted frames.
- An alternative approach is selective repeat.
- The selective repeat protocol retransmits only that frame which is damaged or lost.
- The sender maintains a buffer (sender window) having the

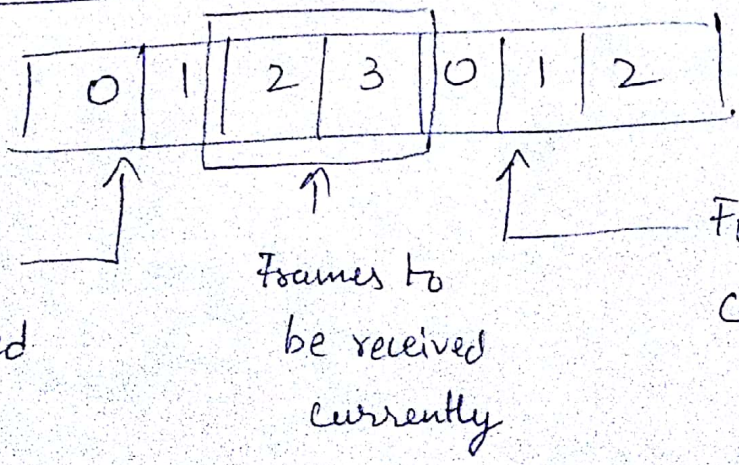
Sequence numbers of the frames that the sender is allowed to send.

- The receiver also maintains a buffer (receiver window) having the sequence numbers of the frames that the receiver is allowed to receive.
- In this protocol, receiver stores the frames received after the damaged frame in the buffer until the damaged frame is replaced.

Sender window :



receiver window :



- The selective repeat protocol is shown in the below example. The Frame 2 is lost, only that frame is retransmitted.