UNIT-2

{Data Link Layer}

* Data Link Layer (DLL) is the second layer in OSI (and) TCP/IP models, whose primary function is to provide reliable communication b/w two directly connected nodes over a physical layer link.
* DDL is responsible for:-
  - ↳ Framing the Packets from NL
  - ↳ Physical Addressing
  - ↳ Error detection (and) Correction [Error control]
  - ↳ Controlling the flow of data [flow control].

* **functionalities of DLL:-**

[Services Provided by DLL]

1. Framing
2. Physical Addressing
3. Flow Control
4. Error Control
5. Access Control.

① Framing:-

The process of encapsulating the data packets recieved from NL, with a header and a trailer, is known as framing.

header	data packet	trailer

Frame

* The header (and) trailer of a frame acts as delimeters.
* Delimeters are the sequence of bits

which are encapsulated with datapackets, in order to distinguish it from other data Packets.

i.e., Every frame has unique delimeter sequences.

* The header (and) trailer of a frame Contains various types of Control informations such as, Source (and) destination MAC addresses, frame type, and error-checking information.
* At reciever's node, the header and trailer and decapsulated and sent to the Network Layer, by the DLL.

② Physical Addressing:-

The process encapsulating the MAC addresses of source node (sender) and destination node (Reciever) with the data Packet from NL, is known as Physical Addressing.

* These MAC addresses are specified in the header part of the frame.

$(MAC)_S$	$(MAC)_D$	data Packet	trailer

$\underbrace{(MAC)_S \;\; (MAC)_D}_{\text{header}}$

Frame

where,

$(MAC)_S \Rightarrow$ MAC address of Source

$(MAC)_D \Rightarrow$ MAC address of Destination

* A MAC [Media Access Control] Address is a unique sequence of hexadecimal digits, assigned to a device by the hardware Manufacturer.
* Hence, the MAC addresses of the devices are used as identifiers to recognize and Communicate over a network.

③ Flow Control:-

The mechanism of Controlling the Rate of data transmission from the Sender to a Reciever, is known as flow control.

* Hence, this mechanism is Controlled by the DLL to match the Data transmission speeds of the Sender and reciever, so that none of the data gets lost (or) recieving the invalid data is avoided.

Ex:-

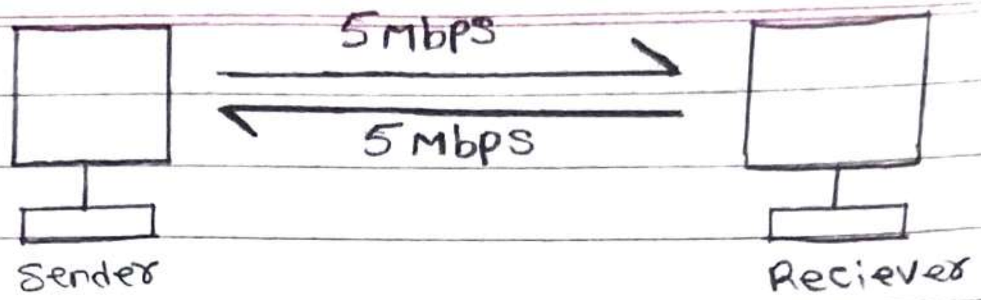
Sender's transmission speed = 10Mbps

Reciever's transmission speed = 5Mbps

∴ for Every Second, the Sender Sends 10Mb of data to the reciever (and) reciever recieves only 5Mb of data.

i.e., Remaining 5Mb of data is lost.

∴ DDL at Sender's side, must decrease the speed by 5Mbps.



[data loss = 0Mbps]

④ Error Control :-

The Process of detecting and Correcting the faults in the transmitted data by the DLL, is known as Error Control.

* This is due to the fact that the transmitted data can get Corrupted due to various reasons like noise, attenuation, etc., during transmission.
* This mechanism is performed by incuding 'error detection bits' into the trailer of a frame, known as Frame Check Sequence (FCS).
* This FCS is Common for all the frames at all nodes of a network.

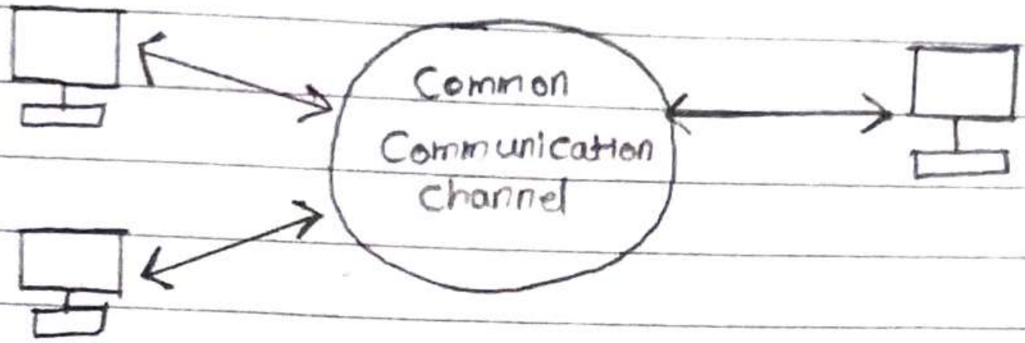
header	data packet	FCS

$\underbrace{\text{FCS}}_{\text{trailer}}$

Frame

⑤ Access Control :-

The Process of Controlling the Communication Access permissions by the DLL, to avoid Collisions when multiple (more than 2) devices Communicate over the same Channel, is known as Access Control.



Sub-Layers of DLL:-

LLC Sub-Layer
MAC Sub-Layer

Data Link Layer (DLL)

① LLC [Logical Link Control] Sub-Layer:-
This Layer is responsible for flow Control mechanism.
i.e., this layer provides 'Flow Control' service.

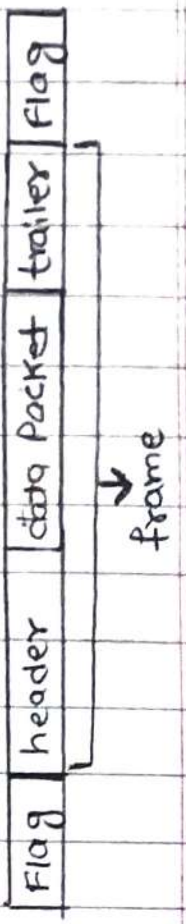
② MAC [Media Access Control] Sub-layer:-
This Layer is responsible for framing, Physical Addressing, Error control and Access Control Services.

Framing:-

[Pg.no: 1-2 ⇒ Framing]

* At low level of OSI (or) TCP/IP model [i.e, At physical layer], the data (frames) are transmitted in the form of bits.
* The transmission is sequential and synchronized.
i.e., bit-by-bit transmission.

* Hence, to indicate the start and end the additional sequence of bits are added at the start and end of a frame, known as '<u>Flag</u>'

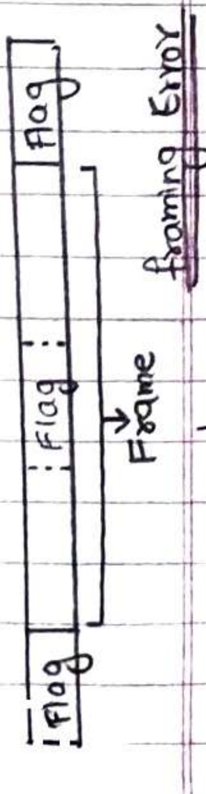


* Same as the Data Packet is a sequence of bits, the flag is also the sequence of bits.
* Hence, there is a possibility of the occurance of the flag sequence in the frame sequence, due to which a part of data (frame) is lost at the reciever's side.
* This senario is known as Framing Error.

Framing Error is caused when the sequence of bits of flag — — occurs in its frame (and) a part of frame data is lost at the receiver end.

<u>Ex</u>:- Consider,

flag = 01111110
frame = 0111011001111110011



→ at Sender's End:-

$\underbrace{01111110}_{flag}\,\underbrace{0111011\,0\,\boxed{01111110}\,011}_{frame}\,\underbrace{01111110}_{flag}$

→ at Reciever's End:-

frame recieved = 0111011

data lost = 01111110011

e
e
er
ral.

* If the Frame size of all frames are fixed (same), then the Framing Error cannot occur.
* Framing Error only occurs when the frames are variable sized.
* 'Bit Stuffing' and 'Byte Stuffing' techniques are used to resolve framing errors.

* **Types of Framing:-**

Based on the length/size/no. of bits of frames, there are two types, viz:-

1. Fixed-size Framing
2. Variable size Framing

① **Fixed-size Framig [Static Framing]:-**

This is when every payload (data packet) recieved from NL is of fixed length.

* Hence, the length of every frame, will be the same.
* Hence, there is no need to use additional bits as the delimiters, at header and trailer of a frame.

i.e., the length of a frame is used as the delimiter, at the reciever end.

i.e., frames are recieved by counting the no. of bits recieved.

* The fixed length of frames must be pre-known in this method. (At reciever end)

Ex:- Sending three frames of length (4)

$f_1 = 0110$, $f_2 = 1100$, $f_3 = 0001$

∴ At reciever End:- $\underbrace{0110}_{\substack{4\text{-bits} \\ (f_1)}}\underbrace{1100}_{\substack{4\text{-bits} \\ (f_2)}}\underbrace{0001}_{\substack{4\text{bits} \\ (f_3)}}$

② **Variable-Size Framing:-** [Dynamic Framing]

This is when the lengths of payloads (data packets) recieved from NL, are of vairable lengths.

* Hence, in order to distinguish the frames from one another, the additional bits are specified in the header and trailer of a frame, as delimiters.

* **Framing Appraaches:-**

It refers to the represention/format of frames used at the low level, to perform transmission.

i.e., the frames can be converted and transmitted in two forms, viz:-

① Bit-oriented Appfach

② Byte/Char-oriented Approach.

① **Bit-oriented Approach:-**

This is when the frames are converted and transmitted in the form of bits, by the physical layer.

* Hence, in this case, a frame is a sequence of bits.

* The protocol used for this approach is:-

→ HDLC [High-Level Data Link Control] protocol.

② **Byte-oriented Approach:-**

This is when the frames are Converted and transmitted in the form of byte (or) Character.

∵ [1 Character size = 1 byte]

* Hence, in this case, a frame is a sequence of bytes/characters.
* This approach is used when transmitting textual (and) multi-media (image, video, mp3) data.
* The protocols used for this approach are:-
  - → BISYNC Protocol [Binary Synchronous Communication]
  - → DDCMP Protocol [Digital Data Communication Message]
  - → PPP Protocol [Point-to-Point]

* **Resolving Framing Error:-**

→ Framing error occurs only when Variable size framing is used and when the flag sequence occurs in frame sequence.

→ To resolve, we add an additional bit/char when it occurs in the frame, and remove it at the reciever end.

→ This method of adding additional bit/byte in the frame when the flag sequence occurs, to resolve framing error, is known as Stuffing.

→ Based on the Approach used for

transmission, the stuffing is of two types

① Bit Stuffing

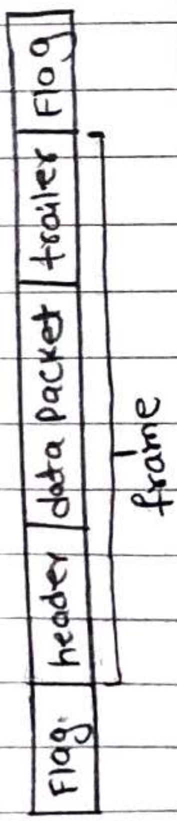
② Byte/Character Stuffing

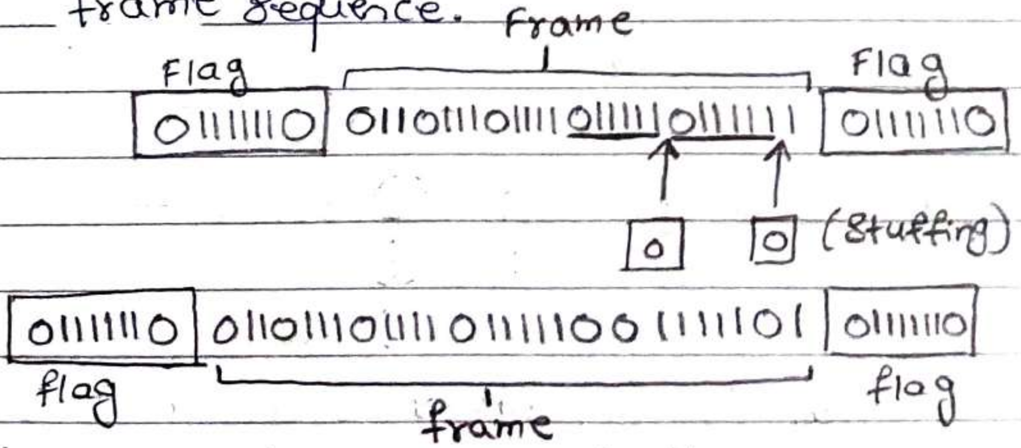
① Bit-Stuffing :-

* This is when Bit-oriented approach is used for data transmission.
* usually, the flag sequence used for indicating the Start and End of a frame is:-

$\boxed{\text{Flag} = 01111110}$

* After the formation of frame in DLL — , the MAC sub layer sequentially checks for a zero and five consecutive 1's [011111], and inserts a (0) bit every time it occurs in the — frame sequence.

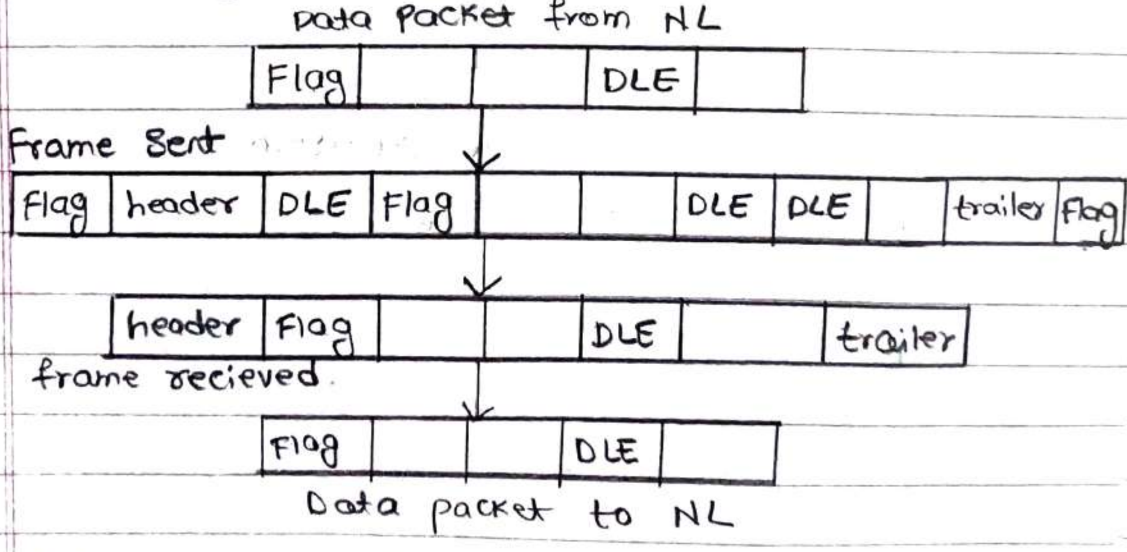




* Inversely, at receiver end, the MAC Sub layer removes the (0) bit after every occurrance of (011111), from the frame received, to get the valid frame.

(2) Byte/Character Stuffing :-

* This is when Byte-oriented approach is used for data transmission.
* A Special character/ is used to resolve the framing error, known as DLE [Data Link Escape].
* (DLE) is inserted before every occurance of (Flag) in the frame, indicating to escape the next character.
  i.e, do not consider the next character as end of frame, but read it as the part of frame.
* There is a possibility of occurance of (DLE) character too, in the frame sequence, Hence, the (DLE) is also specified before the occurance of (DLE) character in the frame.



* At receiver end, the MAC Sub Layer, removes the (DLE) and the next character is considered as the part of frame's data.

//

Error Control :-

[Error Detection and Correction]

The process of detecting and Correcting the faults in the transmitted data by the DLL, is Known as Error Control.

* This is due to the fact that the transmitted data can get Corrupted due to various reasons like noise, attenuation, etc., during transmission.

(i) Error Detection :-

* The process identifying faulty transmission (due to lost bits (or) Change of bits in the frame) is Known as Error detection.
* There are various methods of Error detection :-
  - (a) Parity Check
  - (b) VRC
  - (c) LRC
  - (d) CRC
  - (e) Checksum
* The Sender and reciever are both aware of the method being used.
* In all these methods, the Sender determines a bit/sequece of bits for error detection purpose (at the reciever end), and appends it with the frames.
* The reciever (being aware of the method used) will perform the error detection process

(a) Parity Check Method:-

This method is used for single bit error detection.

i.e., when only one bit will get lost or changed.

* In this method, the measure of no. of (1's) of a frame, is used to perform error detection.
* i.e., whether the no. of (1's) are even (or) not [or] whether the no. of (1's) are odd (or) not

[parity ⇒ being Equal]

* Hence, there are two ways:-
  1. Even Parity
  2. Odd Parity
* The Sender and Reciever are aware of the type of Parity checking method being used.
* But, the frame can have even/odd no. of 1's. Hence, we use an extra bit and append at the end of frame, known as Parity bit.

frame	Parity bit

(1) Even parity:-

→ If the no. of 1's of a frame are even, then the [parity bit = 0].

→ If the no. of 1's are odd, then the [Parity bit = 1] (to make the no. of 1's even).

Ex:-

Sender		Reciever	
1010110	⟶	1010110	✓
0101101	⟶	0101101	✓
1101100	$\xrightarrow{\text{noice}}$	0101100	X
0011101	$\xrightarrow{\text{noice}}$	1011101	X

② odd parity:-

→ If the no.of 1's are odd, then the [parity bit = 0]

→ If the no.of 1's are even, then the [parity bit = 1] (to make the no.of 1's odd)

Ex:-

Sender		Reciever	
1010111	⟶	1010111	✓
0101100	⟶	0101100	✓
1101101	$\xrightarrow{\text{noice}}$	0101101	X
0011100	$\xrightarrow{\text{noice}}$	1011100	X

*disadvantage:- This method is not suitable when multiple bits are changed / lost. [multi-bit error detection]. [burst error]

ⓑ Vertical Redundancy Check (VRC) method:-

This is similar to Parity check method, but the frame is dividended into fixed ~~no~~ no. of bits sizes and then Parity bit is assigned to every part of the frame.

* This method is efficient than Parity check, as is does not need to perform error correction on the whole frame.

Ex:- 0110 1110 0010 [Even Parity]

Sender ⇒ | 0110 | 0 | | 1110 | 1 | | 0010 | 1 |

noice

Reciever ⇒ | 0110 | 0 | | 0110 | 1 | | 0010 | 1 |

✓ ✗ ✓

↳ Perform error correction only on this part of the frame.

* disadvantage:-
* Still not suitable for burst error.

(c) Longitudinal Redundancy Check (LRC) method:-

This is simalar to parity check method, but the frame is divided into fixed no. of bits sizes and arranged Row wise to determine the parity check for each column.

* This method over comes the disadvatage

of VRC and parity [i.e, Suitable for burst error].

0110 1110 0010 (Even parity)

0110
1110
0010
Parity bits → 1010

Sender ⇒ | 1010 | 0110 1110 0010 |
↳ parity bits.

* disadvantages:- This method is not suitable when two bits on the same column change.

(d) Cyclic Redundancy check (CRC) Method :-

* In this method, an error detection sequence known as (CRC) is used.
* At sender side, the CRC is generated by dividing the frame sequence with a divisor of length (L).
* This divisor is pre-known at sender and reciever side and are same.
* The length of the CRC will be (L-1).
* The division is performed followed by the XOR operation.
* CRC is nothing but the final remainder at the sender side.

* XOR operation:-

Same → 0

different → 1

A	B	A(xor)B
0	0	0
0	1	1
1	0	1
1	1	0

* After determining the CRC, it is appended with the frame and sent to reciever.
* At reciever end, the frame (with CRC) is divided with the divisor followed by XOR operation.
* If the remainder is a sequence of 0's, then we can say that there is no error in the frame, (and) vice versa.

i.e., At reciever End:-

reminder → 0 ⟹ no error

reminder → not (0) ⟹ error.

Ex:-

frame = 1001000

divisor = 1101 (L=4)

* At Sender's End:- (generate and append CRC)

→ Append (L-1 = 3) 0's on the frame.

i.e., $100100\underbrace{000}$

```
            111101
      ____________
1101 ) 100100000
       1101
       -----
       01000
        1101
        -----
        01010
         1101
         -----
         01110
          1101
          -----
          00110
           0000
           -----
           01100
            1101
            -----
            0001
```

∴ Reminder (CRC) = 001 [(L-1) bits]

∴ transmitted frame = $100100\underbrace{001}_{CRC}$

*** At Reciver End:-** (Perform division and verify the Reminder)

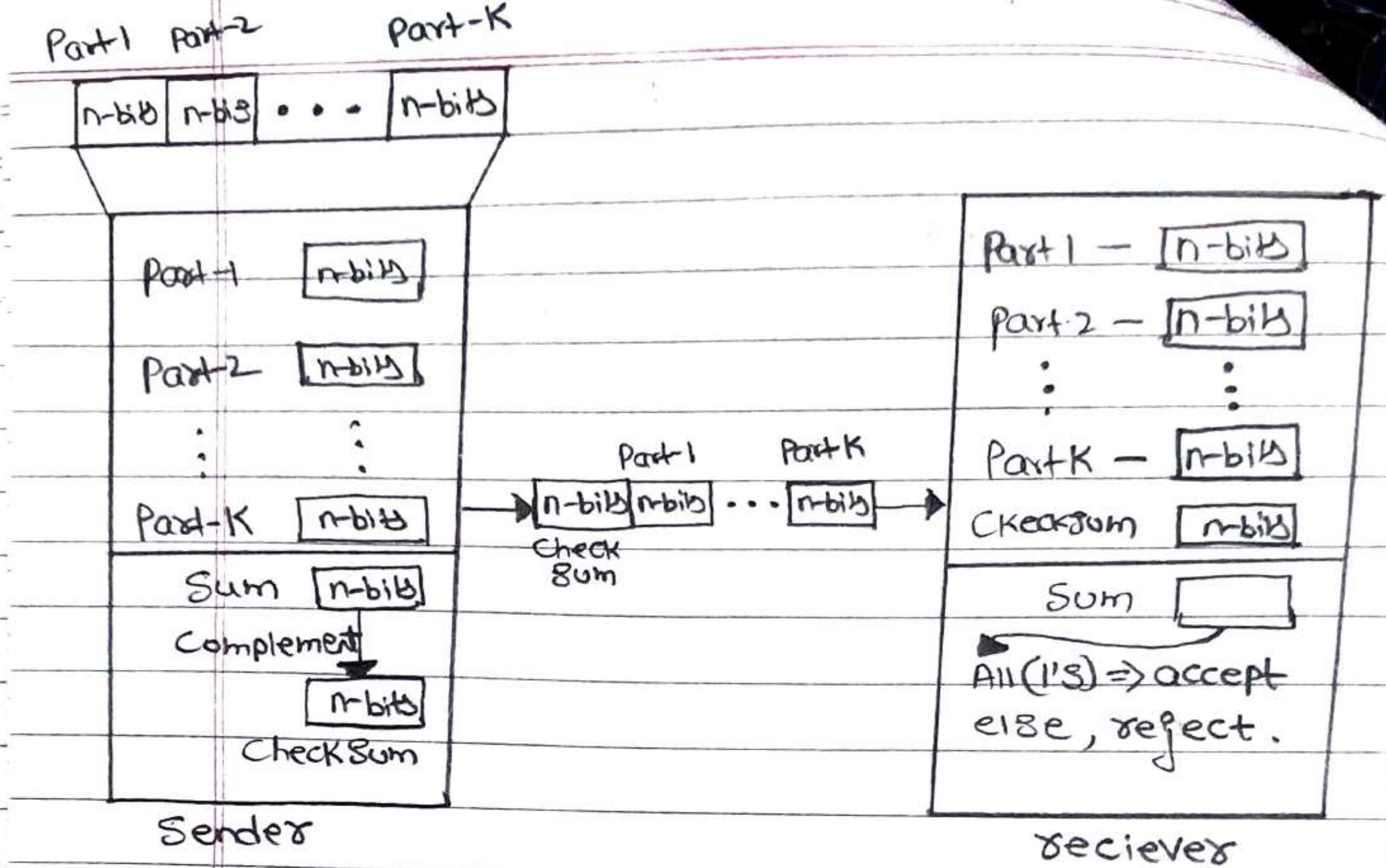
frame recived = 100100001

```
         111101
1101 ) 100100001
       1101
       01000
        1101
        01010
         1101
         01110
          1101
          00110
           0000
           01101
            1101
            0000  ---> Reminder.
```

∴ No errors.//

(e) Check-Sum Method:-

* the frame to be Sent, is divided into (K) no. of parts with (n) bits in each part.
* Addition operation is performed on all the parts of frame.
* In case of any carry bits at the last, add those carry bits into the result.
* Complement (1's) of final result of Addition, is known as a checksum, which is used for error detection.
* As a result, checksum will also be of (n)-bits.



* The checksum is appended with the frame and sent to reciever.
* At reciever end, the (K) parts and the checksum are added.
* Based on the resultant sum, the occurance of error is decided.

i.e.,

Sum (at reciever end) → all (1's) ⇒ No Error (Accept)

Sum (at reciever end) → else ⇒ Error (Reject)

$\left[\frac{32}{4} = 8\text{-bits}\right]$ Ex:- frame = 10011001111000100010010010000100

$\left[\frac{32}{8} = 4 \text{ parts}\right]$

10011001	11100010	00100100	10000100

0+0=0
0+1=1
1+0=1
1+1=10

*** At Sender's End:-**

```
   1  1  1  1  1
   1  0  0  1  1  0  0  1
   1  1  1  0  0  0  1  0
   0  0  1  0  0  1  0  0
(+) 1  0  0  0  0  1  0  0
-----------------------------
[10] 0  0  1  0  0  0  1  1
```

⇒ Add carry (10) into (00100011):-

```
    00100011
(+)       10
------------
    00100101
```

1's Complement

→ 11011010 ⇒ Checksum.

∴ | Checksum | frame | is sent to reciever.

*** At reciever End:-**

```
             1  1  1  1  1  1
             1  0  0  1  1  0  0  1
             1  1  1  0  0  0  1  0
             0  0  1  0  0  1  0  0
             1  0  0  0  0  1  0  0
Checksum ⇒   1  1  0  1  1  0  1  0  (+)
--------------------------------------
        [10] 1  1  1  1  1  1  0  1
```

→ Add carry (10) into (11111101) :-

$$\begin{array}{r} 11111101 \\ (+)\quad\quad 10 \\ \hline 11111111 \\ \hline \end{array} \rightarrow \text{All ones [Accept] (no Error)}$$

(ii) Error Correction:-

This case is when the frame is received and an error is detected at the receiver End.

* In this scenario, there are two ways of Correcting the error:-

(i) The receiver identifies the Exact positions of error and inverts the bits (with sending any ack).

Ex:- Hamming Code method

(ii) The receiver sends negative ack to the sender to re-transmit the whole frame (or) a part of frame.

Ex:- Vertical Redundancy Check (VRC)
Stop-and-wait ARQ protocol
Go-Back-N ARQ protocol
Selective Repeat ARQ protocol

* The Error is only possible when the medium is noisy, for which noisy communication channel is used to detect and correct errors.

Protocols in DLL:-

Framing and Physical addressing doesn't require any protocols, but error control and flow Control does.

* These are of two categories:-

- Protocols
  - for noiseless Channels
    - Simplest
    - Stop-and-wait
  - For noisy Channel
    - Stop-and-wait ARQ
    - Sliding window Protocols
      - Go-Back-N ARQ
      - Selective Repeat ARQ

* A noiseless Channel is a Communication link where data can be transmitted without any errors due to absence of interferences and distorsion in the medium.
* A noisy channel is a Communication link where data transmission Causes errors due to interferences and distorsions in the medium.

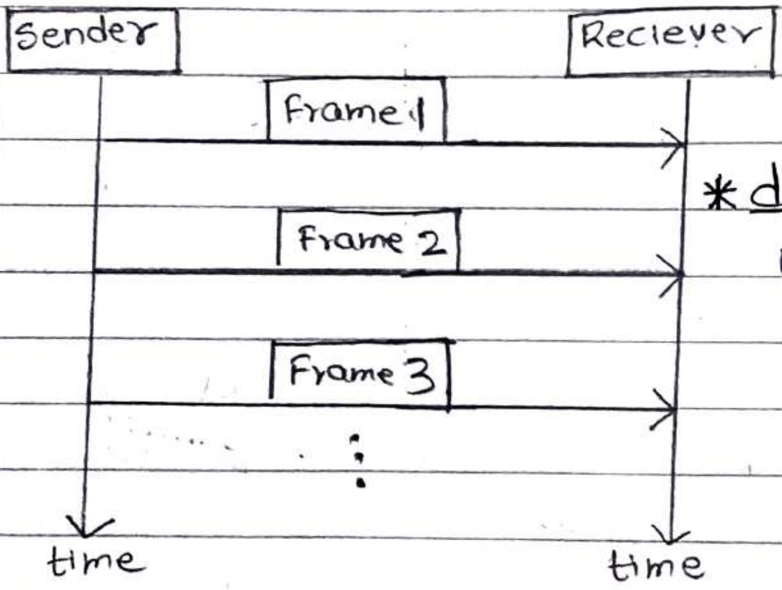
⇒ **For noiseless Channels**

(i) Simplest Protocol:-

This is a uni-directional (only sender to reciever), without any error Control and flow Control mechanisms.

i.e, the Seder will Continuously (no time) Send the frames, irrespective of the acknoledgements (not possible to Send) from the reciever.

* This protocol is Suitable for a reciever which Constantly listens for the incoming frames (and) which may not be overwelmed with the flow rate of incoming frames.



* ~~disadvantage~~:- no flow Control Can be implemented when reciever overwelmes. [data loss]

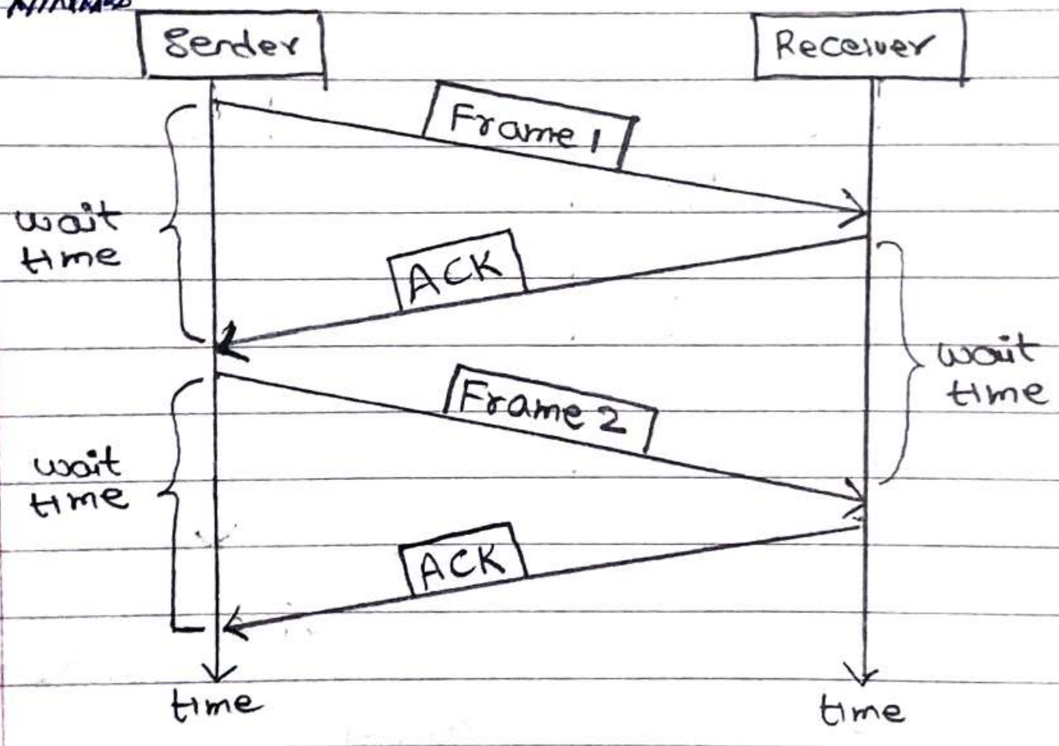
(ii) Stop-and-wait Protocol :-
This is a bi-directional protocol [Sender-Reciever (and) Receiver-Sender], with a flow Control mechanism.
i.e, the Reciever can send Acknoledgements.

~~[illegible]~~

* In this protocol, the Sender sends a frame to the reciever and then waits for an acknoledgement from the reciever.

i.e, the sender sends next frame only if the possitive acknoledgement is recieved.

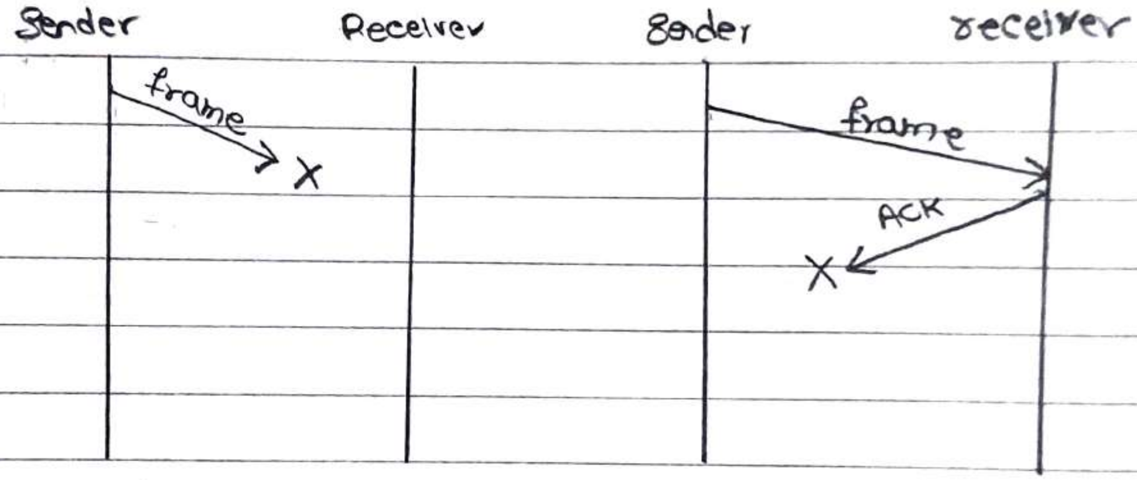
* In case of negative acknoledgement, the sender will retransmit the frame.
* negative acknoledgement means that there was an error in the received frame, and vice-versa.



* This protocol is suitable for the which can be overwelmed with the transmission rate of frames, due to :-
  - ↳ Processing the recieved frame
  - ↳ multiple senders sending data to one reciever.
  - ↳ low internet speed.

***disadvantage:-**

→ In Case of frame loss (or) ACK loss, the sender and reciever waits for an infinite amount of time.
i.e., the sender never recieves an ACK (and) reciever never recieves a frame.



(i) frame loss (ii) ACK loss

infinite waiting

i.e., this protocal is not suitable for noicy channels.

⇒ **For Noisy Channels:-**

All protocals for noisy channels, can be used for both Flow control and Error Control mechanisms.

All of these protocols implements ARQ (Automatic Repeat request) mechanism to achieve reliable data transmission over an unreliable communication channel. [i.e., noisy channel]

In these Protocols, the frames to be sent are assigned with a number sequentially in increasing order, to uniquely identify and order the frames, known as the <u>Sequence Numbers</u>

* ARQ mechanism uses the concept of timeout (at the sender side) to re-transmit the frame (data), assuming that the frame (or) ack is lost.

  i.e., the sender re-transmits the frame if the ACK is not-recieved within a specific period of time.

(one-bit sliding window Protocol)

(i) Stop-and-Wait ARQ Protocol:-

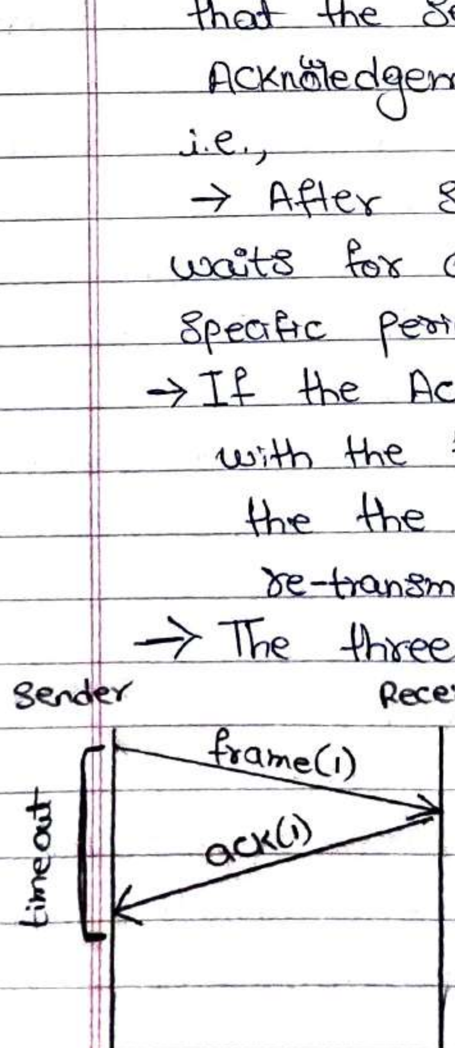
This protocol is similar to stop-and-wait (noiseless) Protocol, the only difference is that the sender is assigned with an Acknoledgement timeout.

i.e.,

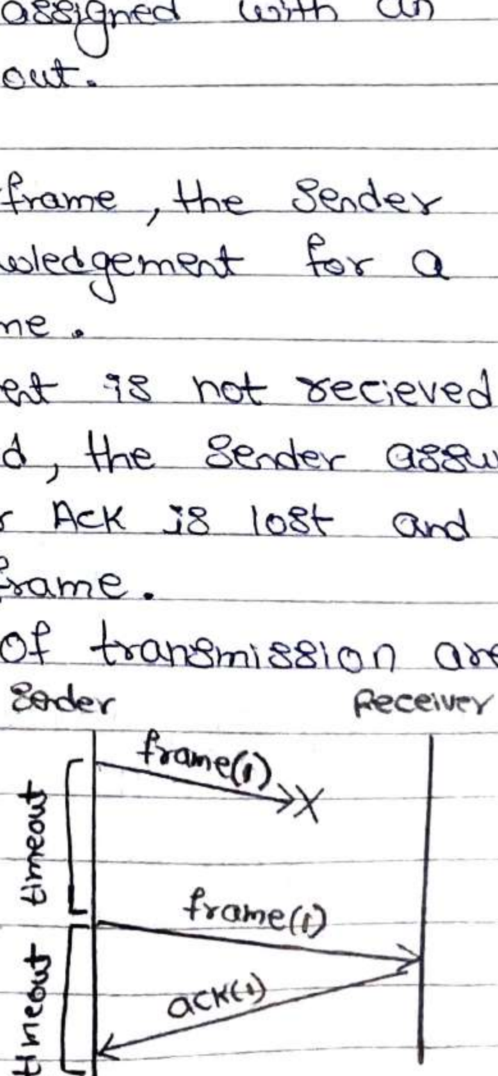
→ After sending a frame, the sender waits for an acknowledgement for a specific period of time.

→ If the Acknoledgement is not recieved with the time period, the sender assumes the the frame or ACK is lost and re-transmits the frame.

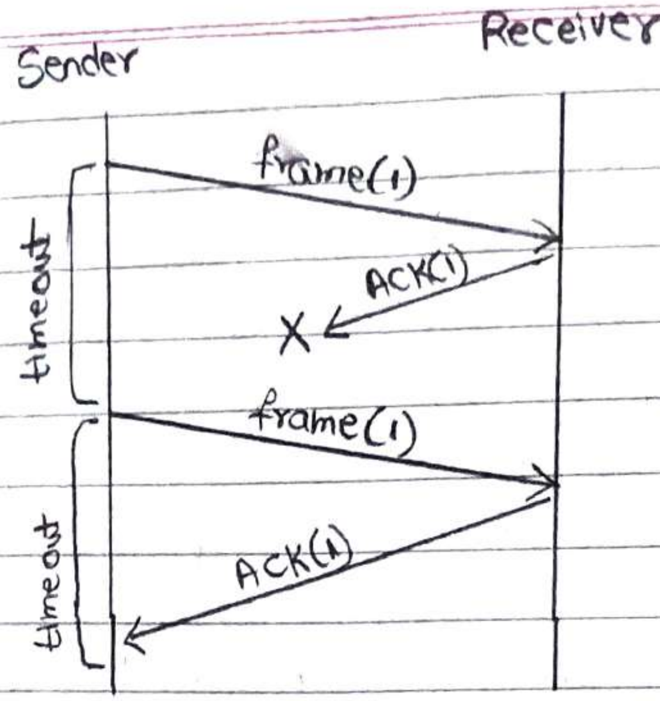
→ The three cases of transmission are:-



(i) ACK is recieved within timeout



(ii) Frame lost



(iii) ACK lost.

*disadvantage:-

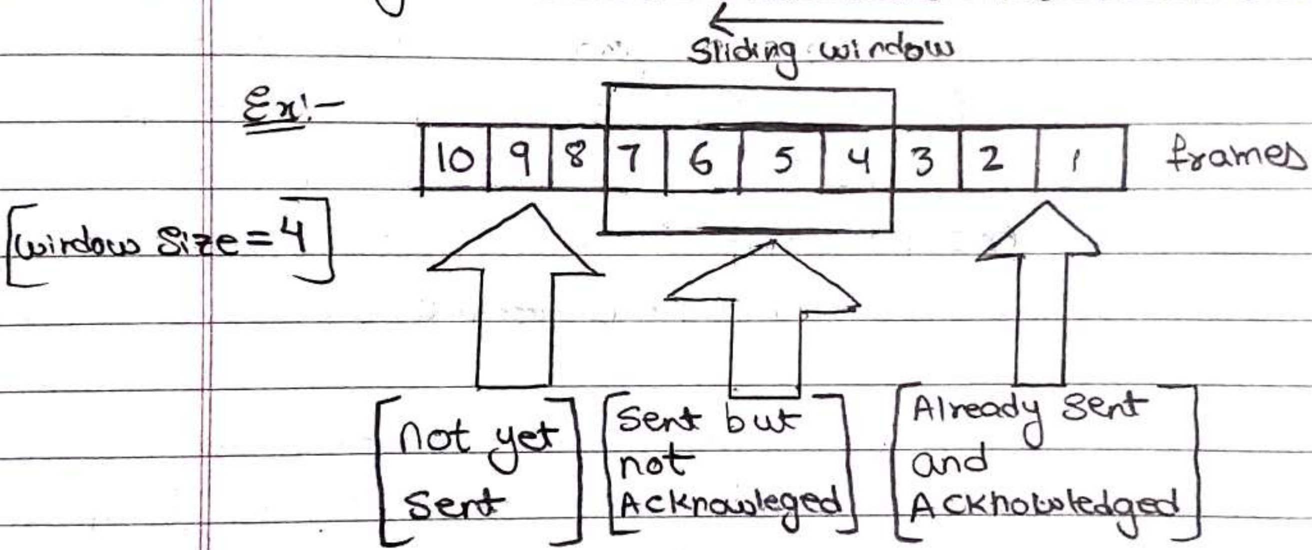
↳ Poor utilization of Receiver's Bandwidth:-
If the receiver is capable of recieving more than (1) frames at a time (due to higher bandwidth (transmission speed)), ~~and~~ receiving only one frame at a time makes the reciever non-utilized device.
i.e., even if the recieving speed is high, the device couldnt get the data fastly.

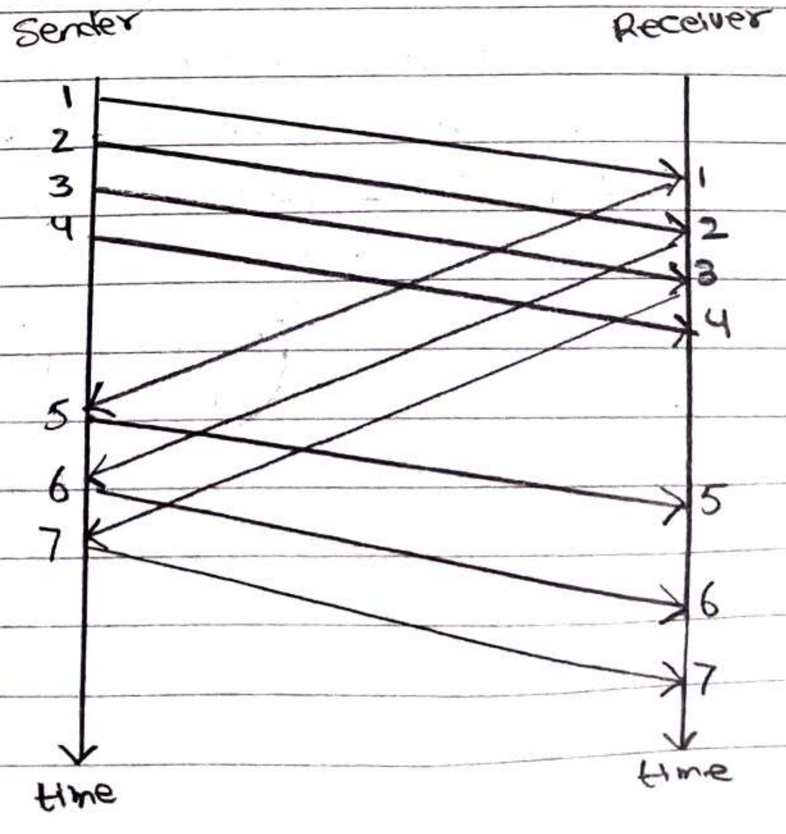
(ii) Sliding Window Protocol :-
These protocols solve the poor utilization problem of Receiver's Bandwidth, by implementing the Sliding window mechanism.

A window of a specified width (no. of frames) which slides over the sequence of frames to be sent, as the Acknoledgements are received, is known as sliding window.

* All of the frames under the window are sent at a time and expected to receive an ack to slide over by by 1-frame and send the next frame.

Ex:-





* There are two types of Sliding window Protocols :-

(a) Go-Back-N ARQ
(b) Selective Repeat ARQ.

(a) Go-Back-(N) ARQ Protocol :-

* here, (N) refers to the window width.

* In this protocol, if the waiting ack is not received on a specific time period, the pointer (which points to the next frame to be sent) is moved (N) frame positions back (and) re-transmitted the frames sequentially.
i.e., if the ack of the first frame in the window, is not received, then all of the frames that are inside the window are re-transmitted.

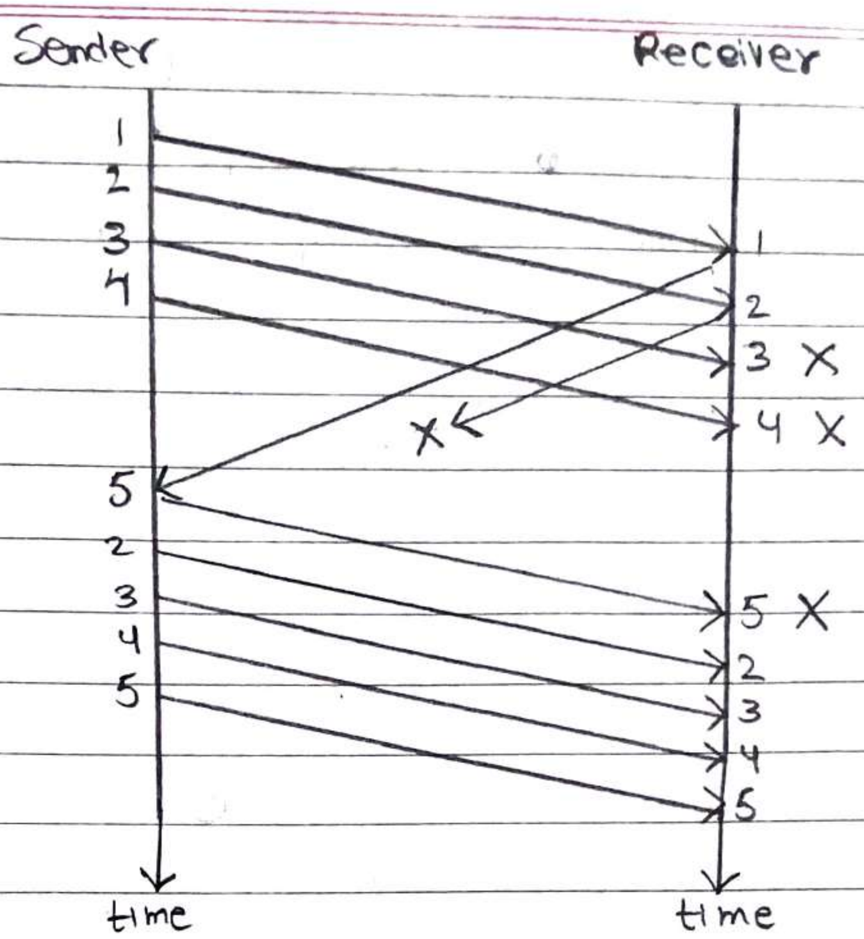
Go-Back-(N) ⇒ [move back the pointer by (N) frame positions.]
(to first position of window)

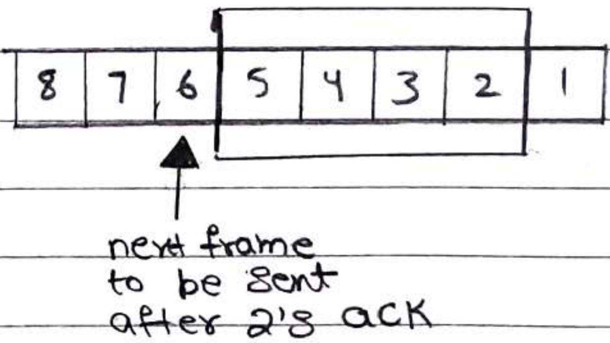
Ex:-

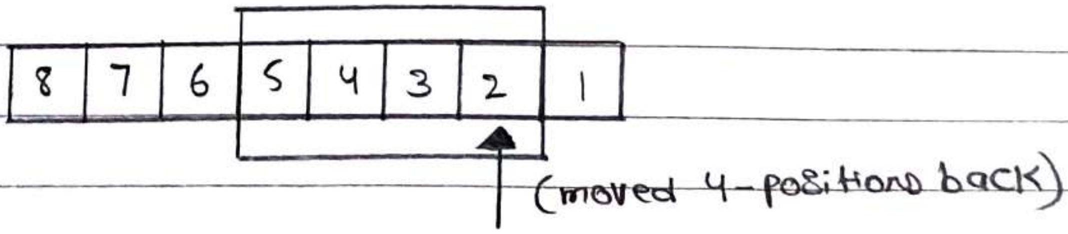
frames ⇒

8	7	6	5	4	3	2	1

window size (N) = 4







* here, the previously recieved frame (3, 4, 5) are also rejected by the receiver to receive data sequentially.

* <u>disadvantage</u>:-

↳ <u>In-efficient</u>:

If one frame or its Ack is lost then all of the frames under the sliding window are retransmitted to follow the sequential transmission

(b) Selective Repeat ARQ Protocol :-

This protocol solves the disadvantage of Go-Back-(N) ARQ protocol.

* In this protocol, instead of re-transmitting all the protocols, only non-Acknowledged and negative Acknowleged frames are re-transmitted to improve the Efficiency.

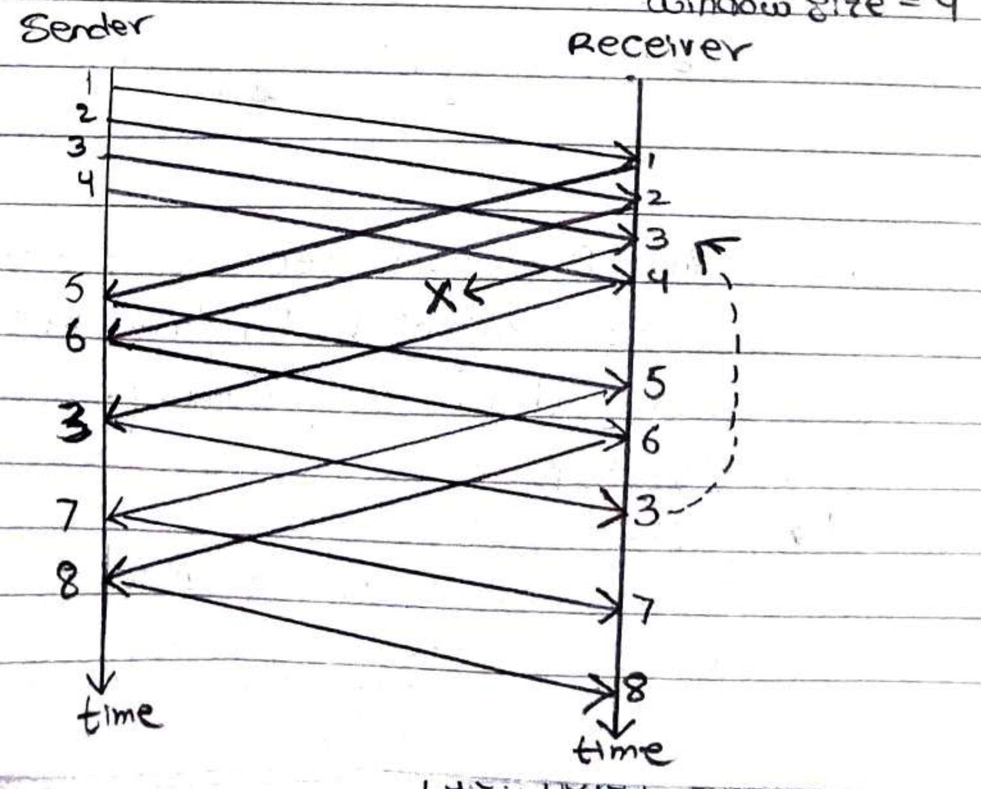
* The reciever keeps track of the missing frames and places the correct frame when the sender re-transmits it.

* This is the most efficient protocol for noisy channels.

Ex:-

8	7	6	5	4	3	2	1

window size = 4



→ when sender was waiting for ACK of (3), (4)'s ACK is received. So, sender assumes that (3) is lost, and retransmits the frame(3).

→ when the next ack (4) is received, the sender continous the transmission from (7)

→ The receiver places the re-transmitted frame (3) back to its sequential position.

⇒ Media/Medium Access Sub-Layer:-

This is a sub-layer of DLL which is responsible for framing, physical Addressing, Error Control and Access Control Services.

* Devices in a Network transmit data through a communication path / medium, known as a channel.
* Hence, in-order to communicate, a channel must be allocated.
* This channel can be wired (or) wireless.
* Since, the Networking devices are mostly long distant devices, hence wireless frequency is used for channelization.
* As long as there are only two devices in a network, the channelization is simple.

* But in vast WAN, the Channelization becomes difficult as there will be multiple devices in the network, trying to get the Correct transmitted data.
* Hence, in WAN, it is the responsibility of DLL on Sender and receiver side to Channelize and transmit the Correct data.
* Multi-Access Control protocols are Categorized into three types:-
  1. Random Access
  2. Controlled Access
  3. Channelization

1. Random Access protocols:-

In these protocols, the Sender is allowed to transmitt the data without worrying about the access Control and destination device location.

→ The receiver is assigned full Control over the Sender's transmission.

Ex- ALOHA, CSMA, CSMA-CA

2. Controlled Access protocols:-

In these protocols, no device is allowed Send the data without the permission of its receiving device.

Ex:- Reservation protocol
Polling protocol
Token Passing Protocol.

③ Channelization protocols:-

In these protocols, the transmitting bandwidth is divided into time, frequency (or) Code.

Ex:- FDMA [frequency division]
TDMA [time division]
CDMA [Code division.]

* By using any of these protocols, a specific channel can be allocated for communication of two devices in WAN.

Channel Allocation Problem

* Based on the length of communication session, the channel Allocation is of two types:-

① Static Channel Allocation
② Dynamic channel Allocation

① Static Channel Allocation:-

In this type of allocation, the communication session b/w devices is infinite, untill changed manually. i.e, This Allocation is applied among the divices which tramits data only among themselves.

* That is, once the communication channel is established, it canned be ended [fixed channels].
[Channels cannot be uprooted (removed)]

* The protocols used are:-

(i) **FDMA [Frequency division Multiple Access] protocol :-**

The available frequency is divided into non-overlapping frequency bands, and each user is assigned a specific ~~ban~~ frequency band for communication.

(ii) **TDMA [Time Division Multiple Access] protocol :-**

The available time is divided into multiple time-slots, and each user is assigned a specific time slot during which it can transmit data.

(iii) **CDMA [Code Division Multiple Access] protocol :-**

A unique code is assigned to each user, allowing multiple user to transmit simultaneously on the same frequency.

i.e., identification is done based on the unique code.

② **Dynamic Channel Allocation :-**

In this type of Allocation the communication session b/w devices is finite.

* That is, the communication channel can be established and uprooted (removed) whenever required.

* The protocols used are:-

(i) <u>CSMA [Carrier Sence Multiple Access] Protocol:-</u>
The sending devices waits and listens to the channel for permission to transmit data, from Receiving device.

(ii) <u>CSMA-CA [Collision Avoidance] protocol:-</u>
This protocol is used for controlling the simultaneous transmissions to avoid collisions.

(iii) <u>Token passing Protocol:-</u>
A token is circulated among multiple devices in the network, (and) only the device identifying the specific token is allowed to receive data.

Some Multiple-Access Protocols

⇒ ALOHA Protocol:-

* It is a Random Access Protocol which allows data to be transmitted via a shared network channel, at any time.
* That is, Each node/station using ALOHA Protocol, transmits a frame without determining whether the transmission channel is idle (or) busy.
* If the channel is idle, the frame will be successfully transmilted.
* If the channel is busy, then the two frames occupy the channel at the same time and collide.
* Types of ALOHA protocols:-

(i) Pure ALOHA Protocol:-

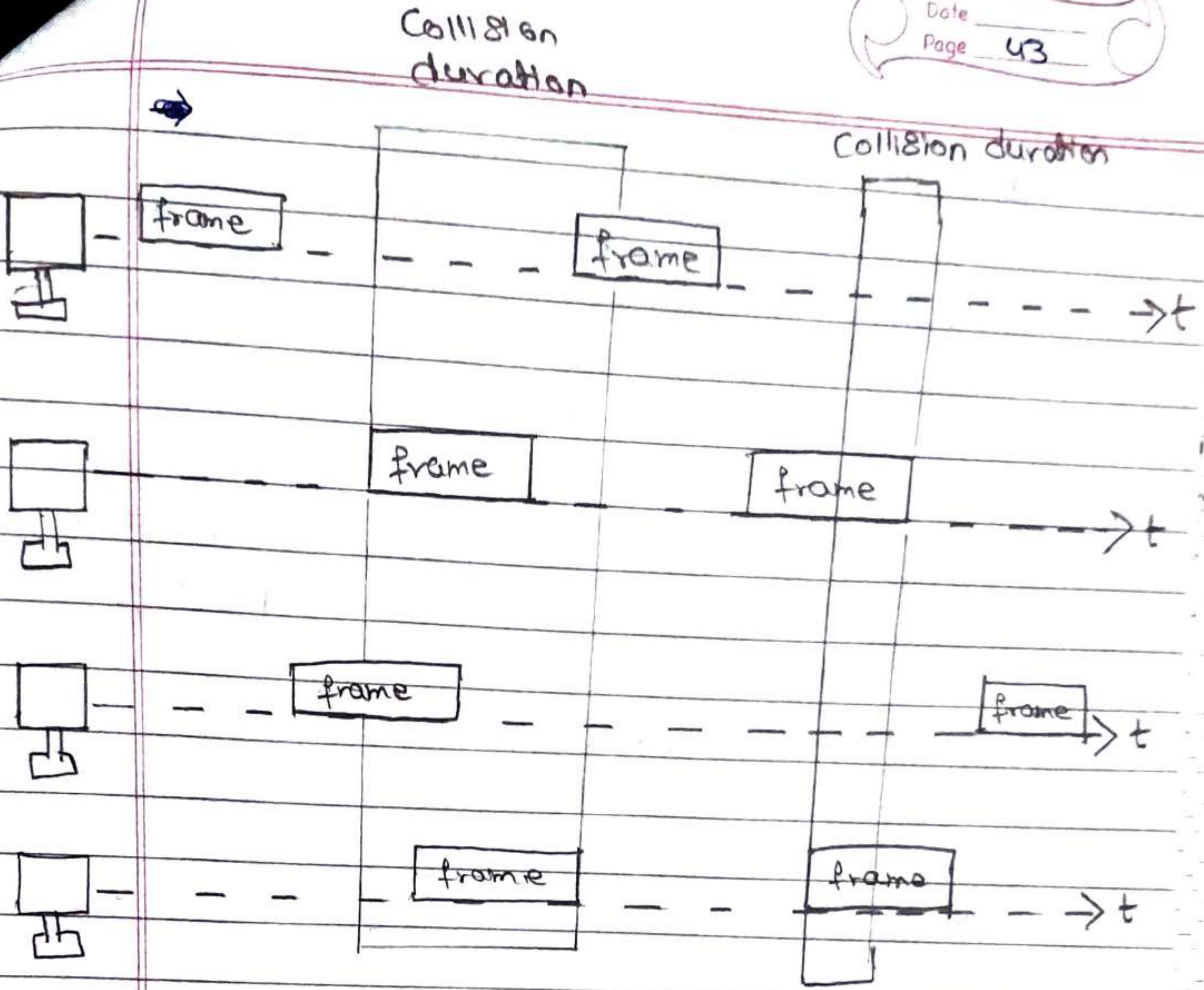
→ The sender can transmit frames at any time, due to which the probability of occurrence of collision increases.

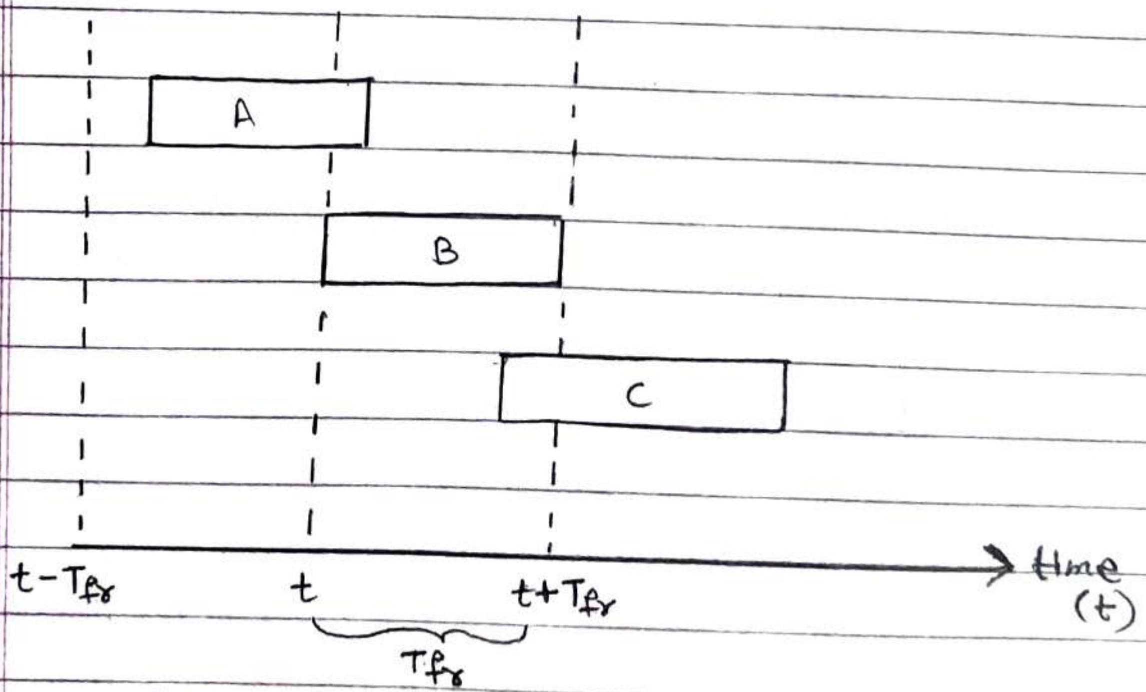
→ If a frame is collided and lost and no (ACK) is received, the sender will wait for a Random amount of time to re-transmit the frame.

→ If (T_{fr}) is the transmission time of a frame, then the Vulnerable time (V_t) is:-

$$V_t = 2 \cdot T_{fr}$$

Vulnerable time ⇒ duration during which a collision can occur in a network.





→ If (G) is the Avg no. of frames, then the throughput of the transmitting station (S) is :-

$$S = G \cdot e^{-2G}$$

→ max. value of (G) is:- $G = \frac{1}{2}$

∴ the maximum throughput is:-

$$S_{max} = 0.184$$

∴ the Efficiency of Pure ALOHA is:- 18.4%

(ii) Slotted ALOHA Protocol:-

Slotted ALOHA is more efficient version of Pure ALOHA.

→ The frames size in the Protocol are fixed.

→ There is a global clock in this Protocol which divides the time into fixed no. of time slots

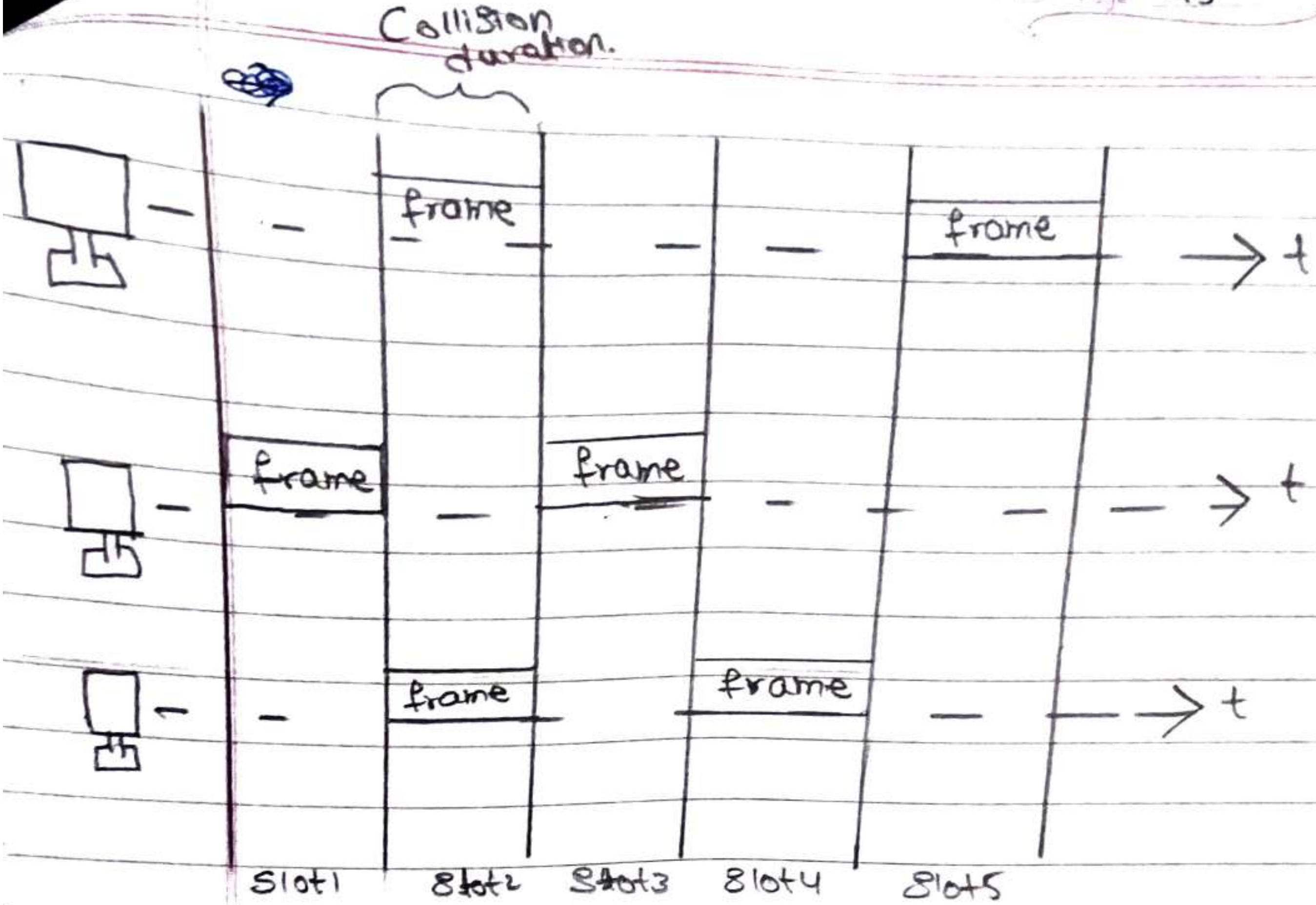
→ Each time-slot is same as the transmision time of a frame (T_{fr})

→ Hence, the seder can transmit the frame only on the start of a time-slot.

→ If a time-slot is missed, the sender has to wait and transmit the frame in next time slot.

→ The Vulnerable time is:-

$$\boxed{V_t = T_{fr}}$$



→ Collisions are possible but lesser than that of in pure ALOHA.