Unit: III

Data Link Control: Framing, Flow & Error Control, protocols, Noiseless channels (Simplest Protocol, Stop and Wait protocol); Noisy channels (Stop-and Wait ARQ).

Data Link Layer.

Responsibilities

↳ ~~framing~~, addr

Specific responsibilities of the data link layer include framing, addressing, flow control, error control, and media access control.

DLL DESIGN ISSUES.

I FRAMING

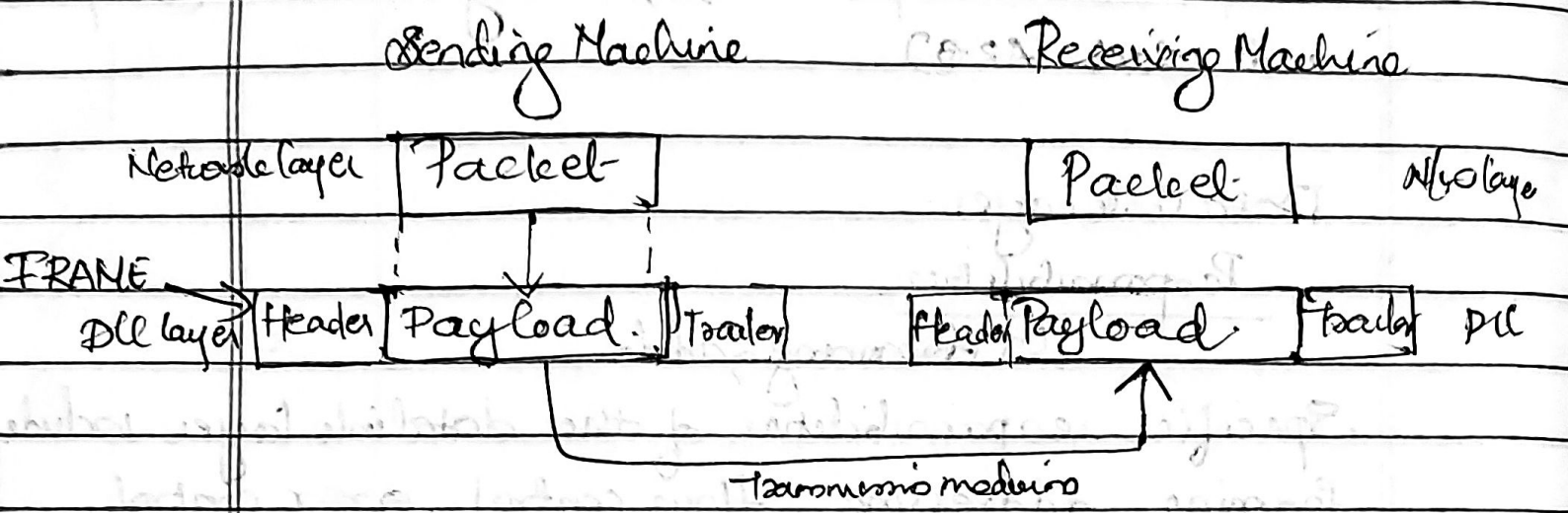
[payload]

- The data link layer divides the stream of bits received from the network layer into manageable data units called frames.

- The data link layer adds a header to the frame to define the addresses the sender and receiver of the frame.

- Each frame contains a frame header, a payload field for holding the packet and a frame trailer.

Relationship b/w packets & frames



Fixed Size Framing (easy to handle)

In fixed size framing, there is no need for defining boundaries of the frames; the size itself can be used as a delimiter.

Eg: ATM used area network.

Variable-Size Framing

In variable-size framing, we need a way to define the end of the frame and the beginning of the next.

→ The approaches used for this purpose are

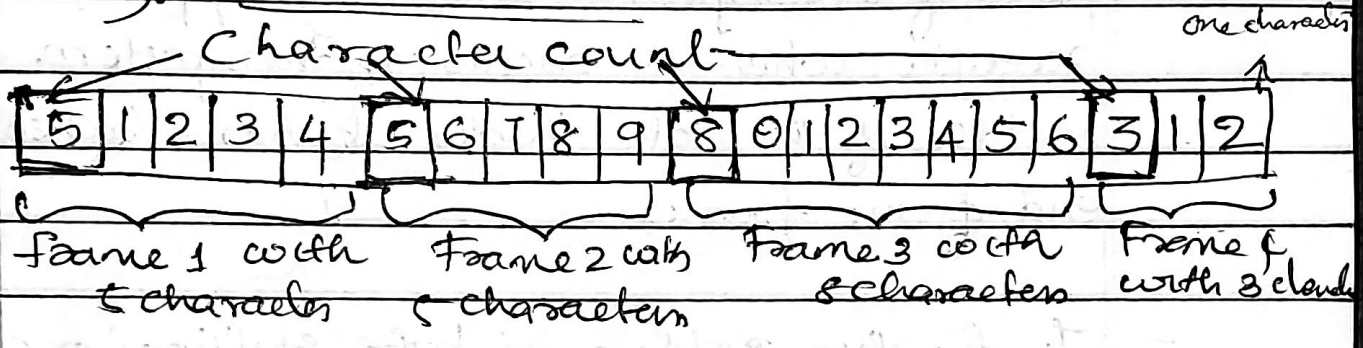
A. Character Count

- It uses a field in the header to specify

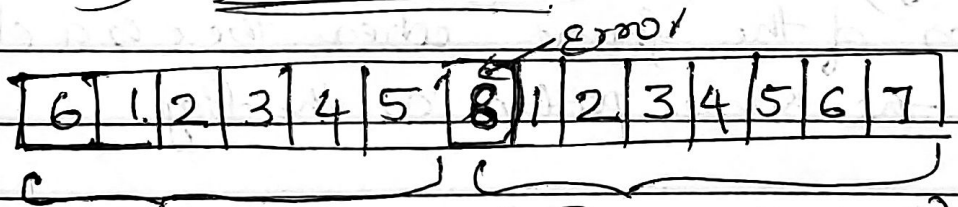
the number of characters in the frame

- 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.

a). Without Error



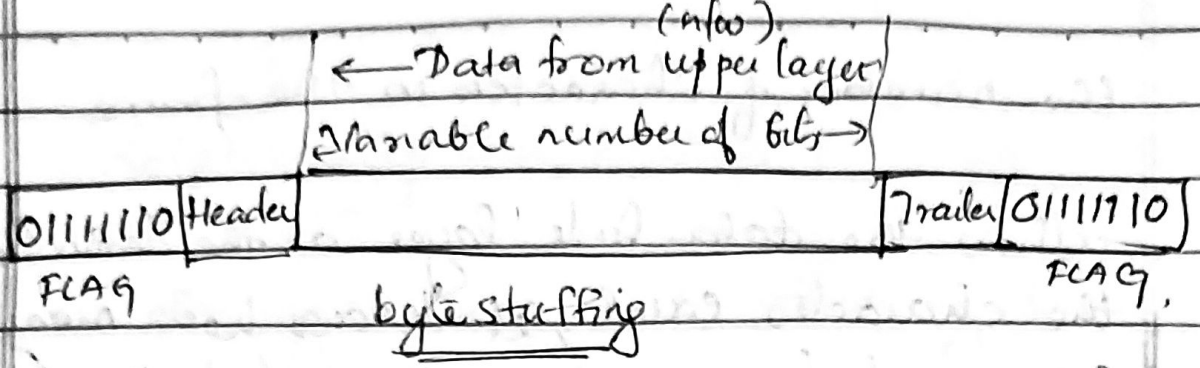
b). With Error



B. Flag bytes with byte stuffing

- To separate one frame from the next, an 8-bit (1-byte) flag is added at the beginning and end of a frame.

If the bit pattern = flag Then we have to add ESC < bit pattern > which considered as info.



byte stuffing

- Any pattern used for the flag could also be part of the information. If this happens, the receiver when it encounters this pattern in the middle of the data, thinks it has reached the end of the frame.

- To fix this problem, in byte stuffing (or character stuffing) a special byte is added to the data section of the frame when there is a character with the same pattern as the flag.

- special byte is usually escape character (ESC) which has a predefined bit pattern.

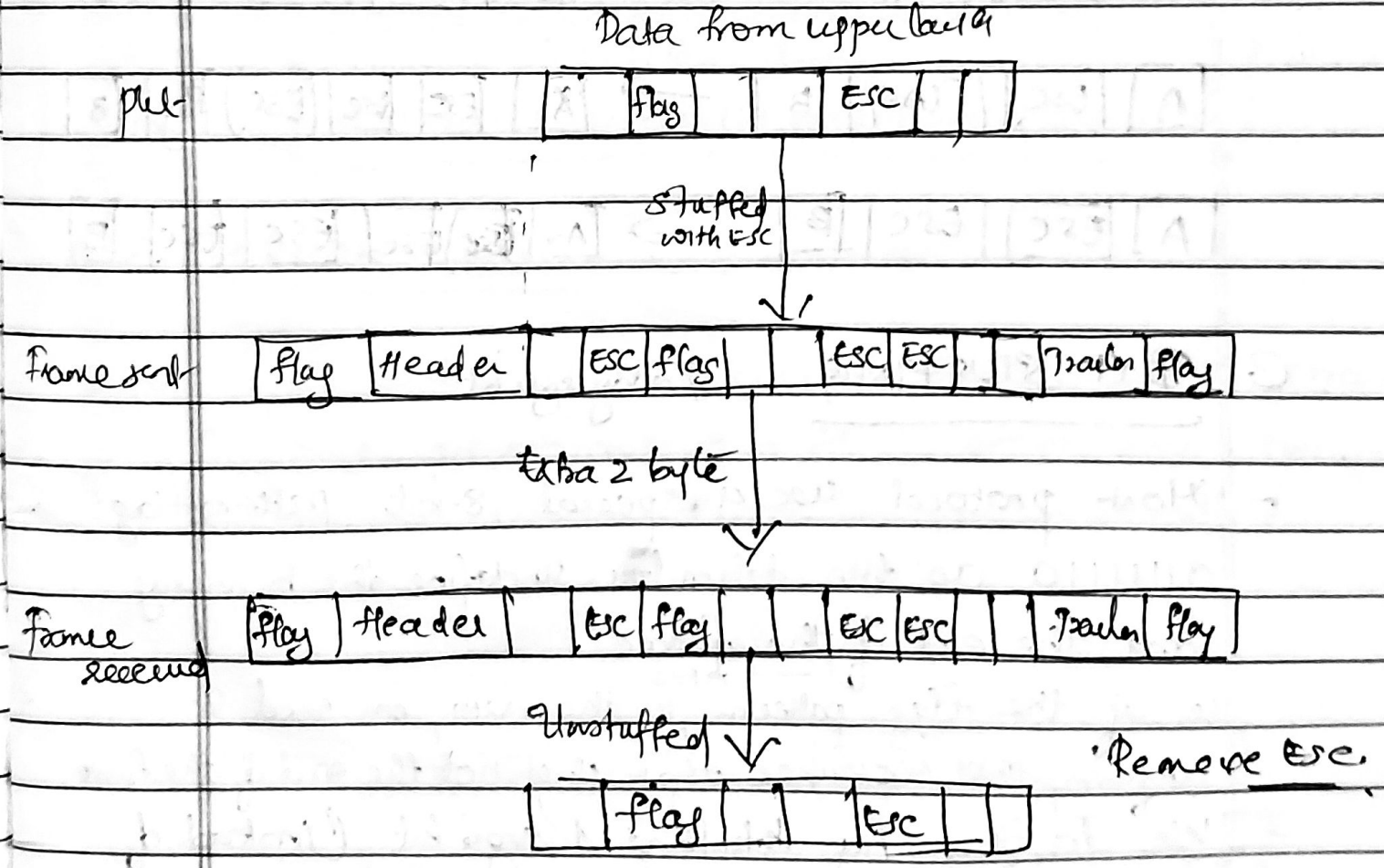
- Whenever the receiver encounter the ESC character, it removes it from the data section and treats the next character as data, not a delimiting flag.

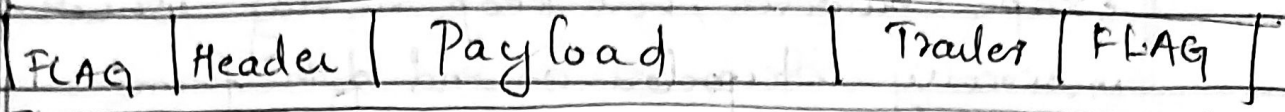
- If the text contains one or more escape characters followed by a flag, the receiver removes the

escape character, but keep the flag which is incorrectly interpreted as end of the frame.

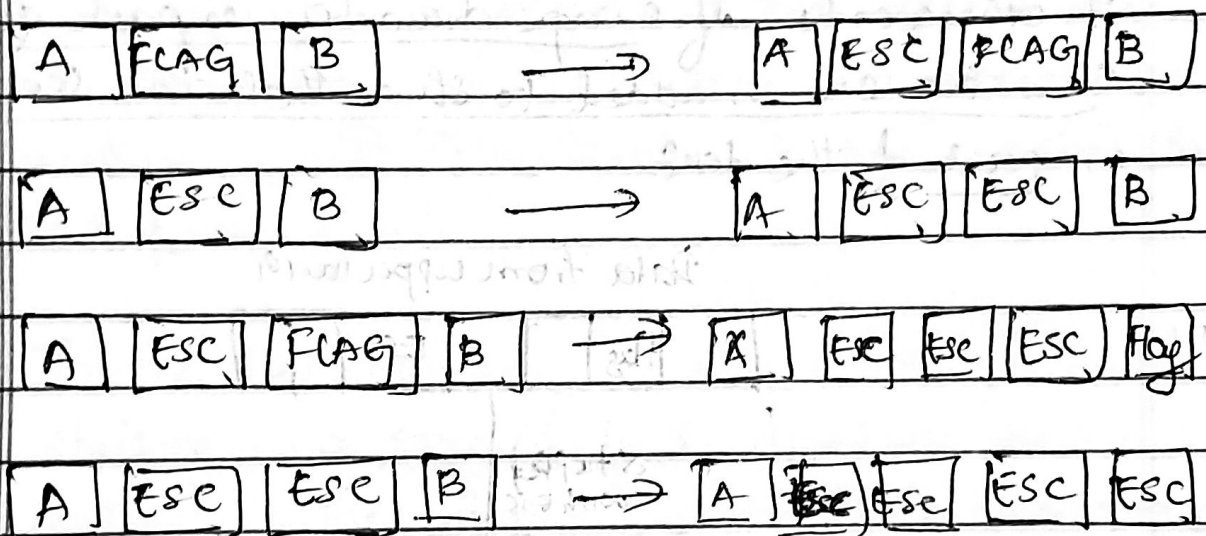
- To solve this problem, the escape characters that are part of a text must also be marked by another escape character.

In other words if escape character is part of the text an extra one is added to show that the second one is part of the text





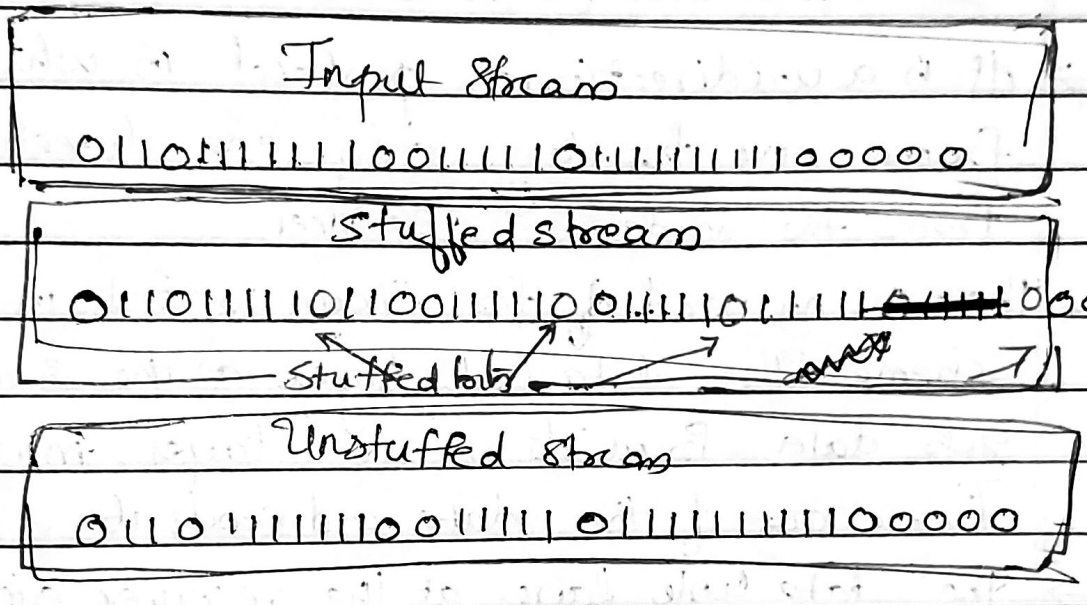
Original sequence → After stuffing



c. BIT STUFFING // addy one bit

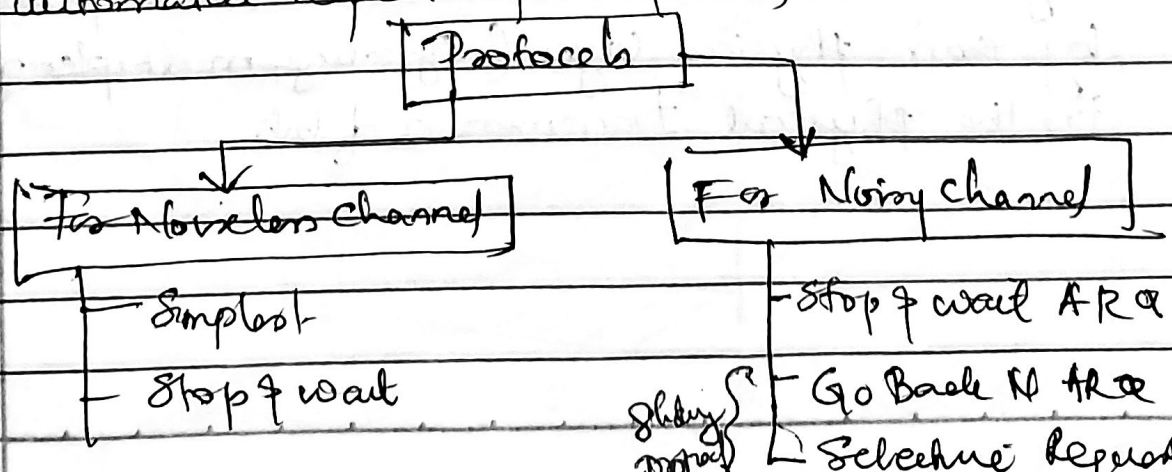
- Most protocol use a special 8-bit pattern flag 01111110 as the delimiter to define the beginning and the end of the frame.
ie if the flag pattern appear in the data, we need to inform the receiver that it is not the end of the frame
- We do this by stuffing 1 single bit (instead of 1 byte) to prevent the pattern from looking like flag. The strategy is called bit stuffing

- In bit stuffing, if a 0 and (5) consecutive 1 bit are encountered an extra '0' is added.
- This extra stuffed bit is eventually removed from data by the receiver



2. ERROR CONTROL

- Error control is both error detection & error correction
- Any time an error is detected in an exchange, specified frames are retransmitted. This process is called automatic repeat request (ARQ)

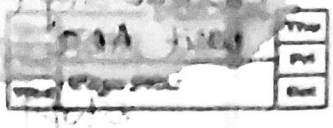


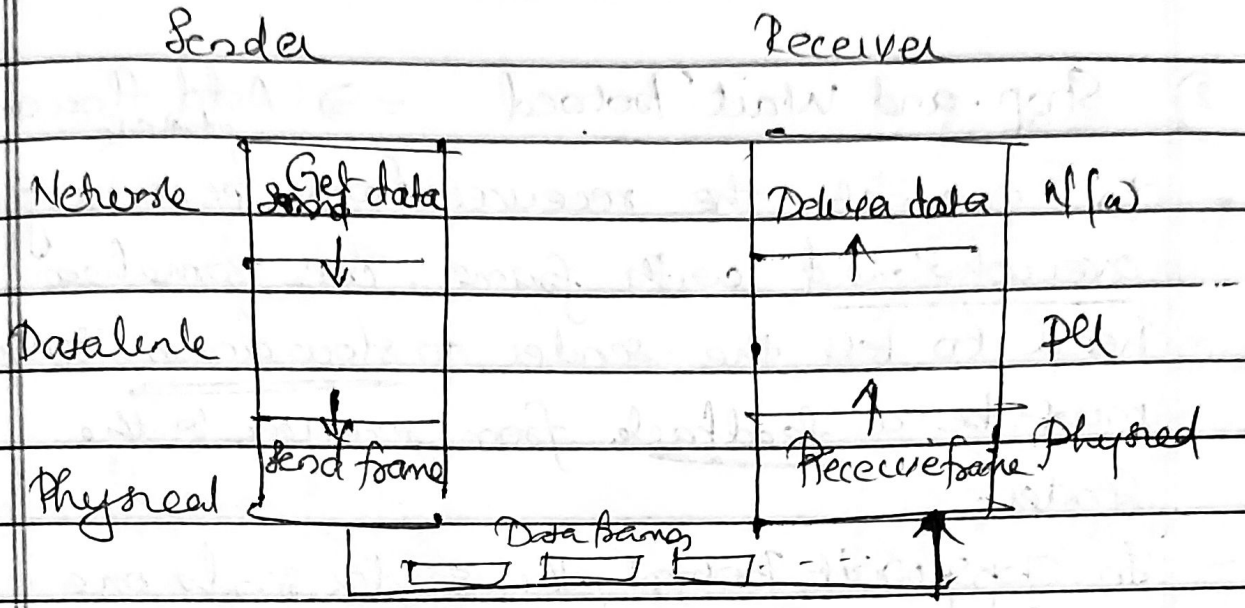
Flow Control: Suppose a situation where the sender is sending data at speed higher than the receiver is able to receive & process it then data will get lost. Flow control help is ensuring data doesn't get lost.

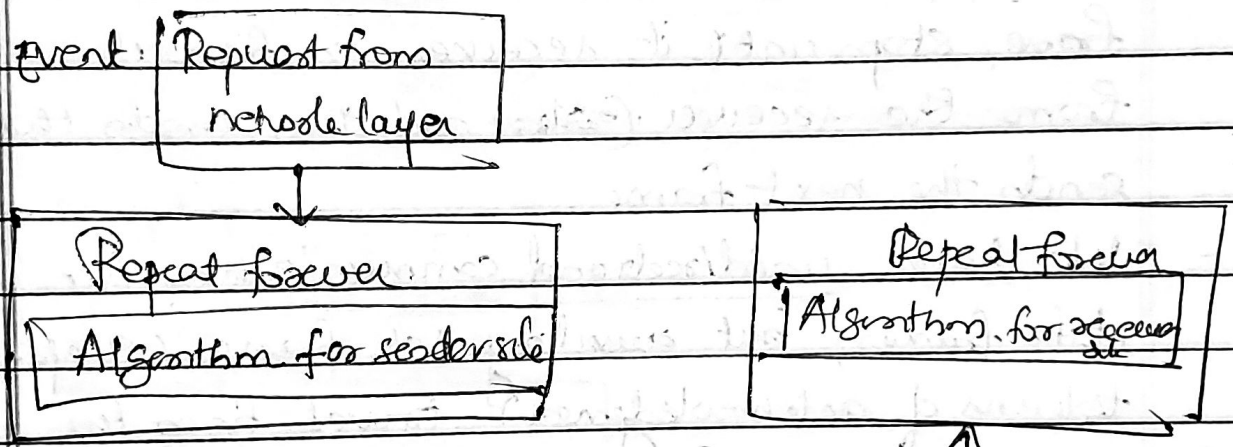
Noiseless Channel

1) Simplest Protocol (Utopian)

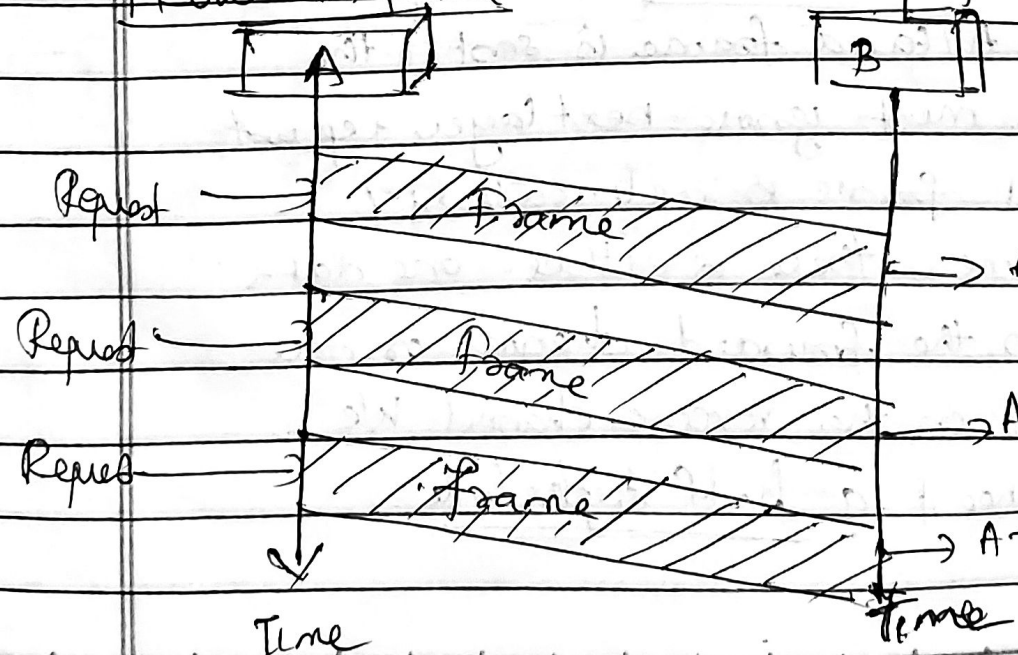
- No flow control & error control
- → It is a unidirectional protocol in which data frames are traveling only one direction from the sender to receiver
- → There is no need for flow control in this scheme. The data link layer at the sender site gets data from its network layer, makes a frame out of the data and sends it
- → The data link layer at the receiver site receives a frame from its physical layer, extracts data from the frame and delivers data to its network layer.
- The data link layer of the sender and receiver provide transmission services for their network layer. The data link use the services provided by their physical layer (signaling, multiplexing & soon) for the physical transmission of bits







FLOW DIAGRAM

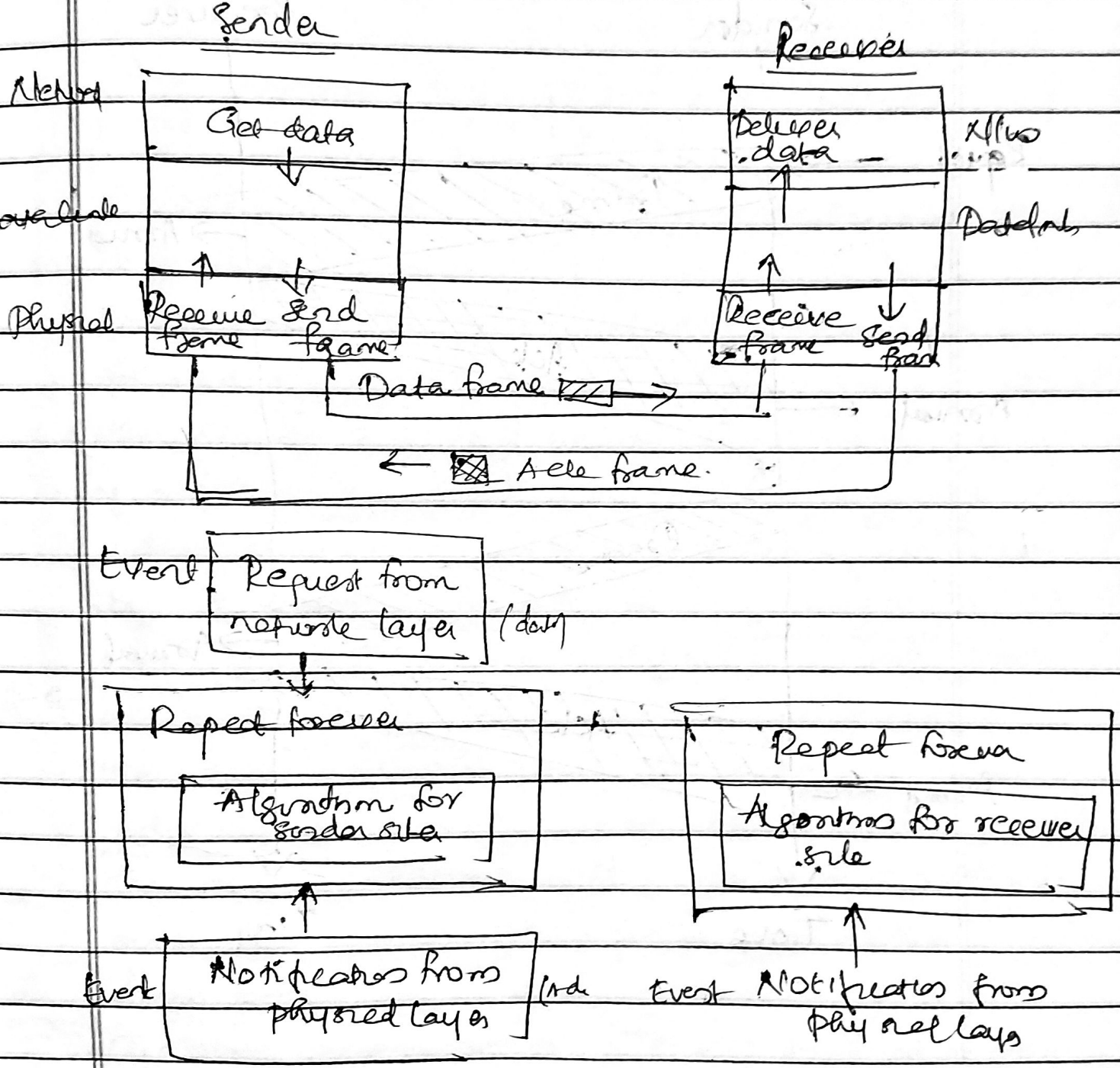


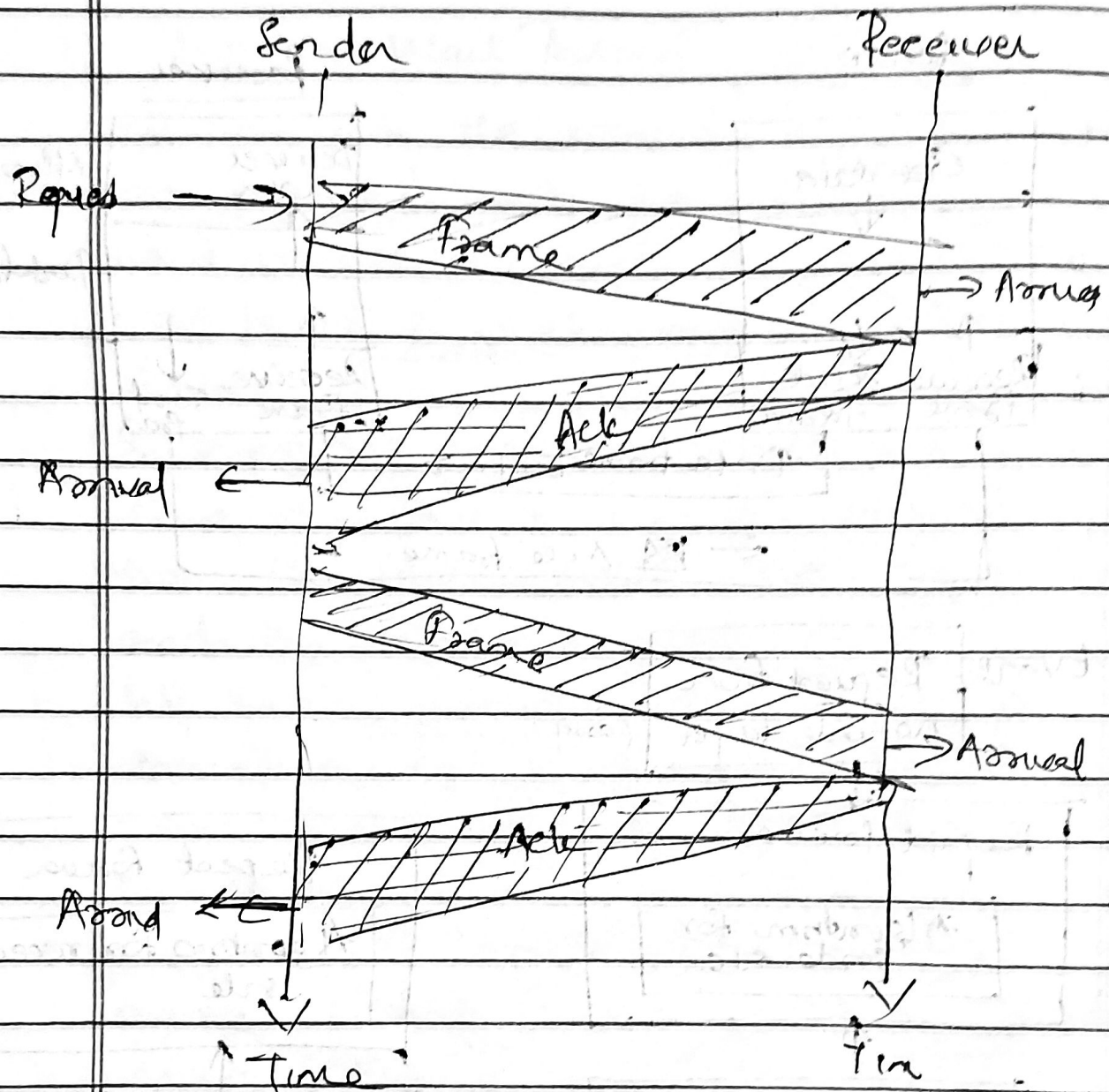
3 Problems: Lost data, Lost Ack, Delayed Ack
(Infinite time waiting) (duplicate data)

Stop & wait Ack + Timeout + SequenceNo + SequenceNo
Stop & wait (resend) times + (data) (Ack.

2) Stop-and-Wait Protocol ⟹ Add flow control + Ack

- to prevent the receiver from becoming overwhelmed with frames, we somehow need to tell the sender to slow down. There must be a feedback from receiver to the sender.
- In Stop & wait Protocol, the sender sends one frame, stops until it receives confirmation from the receiver (~~OK~~) and then sends the sends the next frame.
- We have unidirectional communication for data frames, but auxiliary Ack frames (simple tokens of acknowledgment) travel from the other direction
- We add flow control to our previous protocol. After a frame is sent, the algorithm must ignore next layer request until that frame is acknowledged
- At any time, there is either one data frame on the forward channel or one Ack frame on the reverse channel. We therefore need a half duplex link


Sender
Receiver
Network
Get data
Deliver data
Data link
Physical
Receive frame
Send frame
Receive frame
Send frame
Data frame
Ack frame
Event
Request from network layer
(data)
Repeat forever
Algorithm for sender side
Repeat forever
Algorithm for receiver side
Event
Notification from physical layer
(ack)
Event
Notification from physical layer


Sender
Receiver
Reques
Frame
Arrival
Ack
Arrival
Frame
Arrival
Ack
Arrival
Time
Tim

Noisy channel + flow control
↳ Add Error Control

1. Stop & Wait Automatic Repeat Request

- It adds a simple error control mechanism to the Stop & Wait protocol
- To detect and correct corrupted frames, we need to add redundancy bit to our data frame. When the frame arrives at the receiver site, it is checked and if it is corrupted (i), it is silently discarded. The detection of errors in this protocol is manifested by the silence of the receiver
- Lost (ii) frames are more difficult to handle than corrupted ones. The received frame could be the correct (iv) one, or a duplicate (iii), or a frame out of order (v). The solution is to number the frames. When the receiver receives a data frame that is out of order, this means that frames were either lost or duplicated
- The lost frames need to be resent in this protocol. The sender keeps a copy of the sent frame. At the same time, it starts a timer
- If the timer expires and there is no Ack for the sent frame, the frame is resent, the copy is held, and the timer is restarted

- Since the protocol uses the stop and wait mechanism, there is only one specific frame that needs an Ack even though several copies of the same frame can be in the network.
- Since an Ack frame can also be corrupted and lost, it too needs redundancy bit and a sequence number. The Ack frame for this protocol has a sequence number field. In this protocol, the sender simply discards a corrupted Ack frame or ignores an out-of-order one.

(0,1) Sequence Number = Sender window size + Receiver window size
1 + 1 = 2 //

The frames need to be numbered. This is done by using sequence numbers. A field is added to the data frame to hold the sequence number of that frame.

- One important consideration is the range of sequence numbers. Since we want to minimize the frame size, we look for the smallest range.
- Assume we have used 'x' as sequence number, we only need to use 'x+1' after that. There is no need for x+2.

Assume that sender has sent the frame numbered x. Three things can happen

(i) The frame arrives safe and sound at the receiver site, the receiver sends an acknowledgement. The acknowledgement arrives at the sender site causing the sender to send the next frame numbered 'x+1'

(ii) The frame arrives safe and sound at the receiver site; the receiver sends an acknowledgment but the acknowledgment is corrupted or lost. The sender resends the frame (numbered x) after the time-out.
Note that the frame here is a duplicate. The receiver can recognize this fact because it expects frame 'x+1' but frame x was received

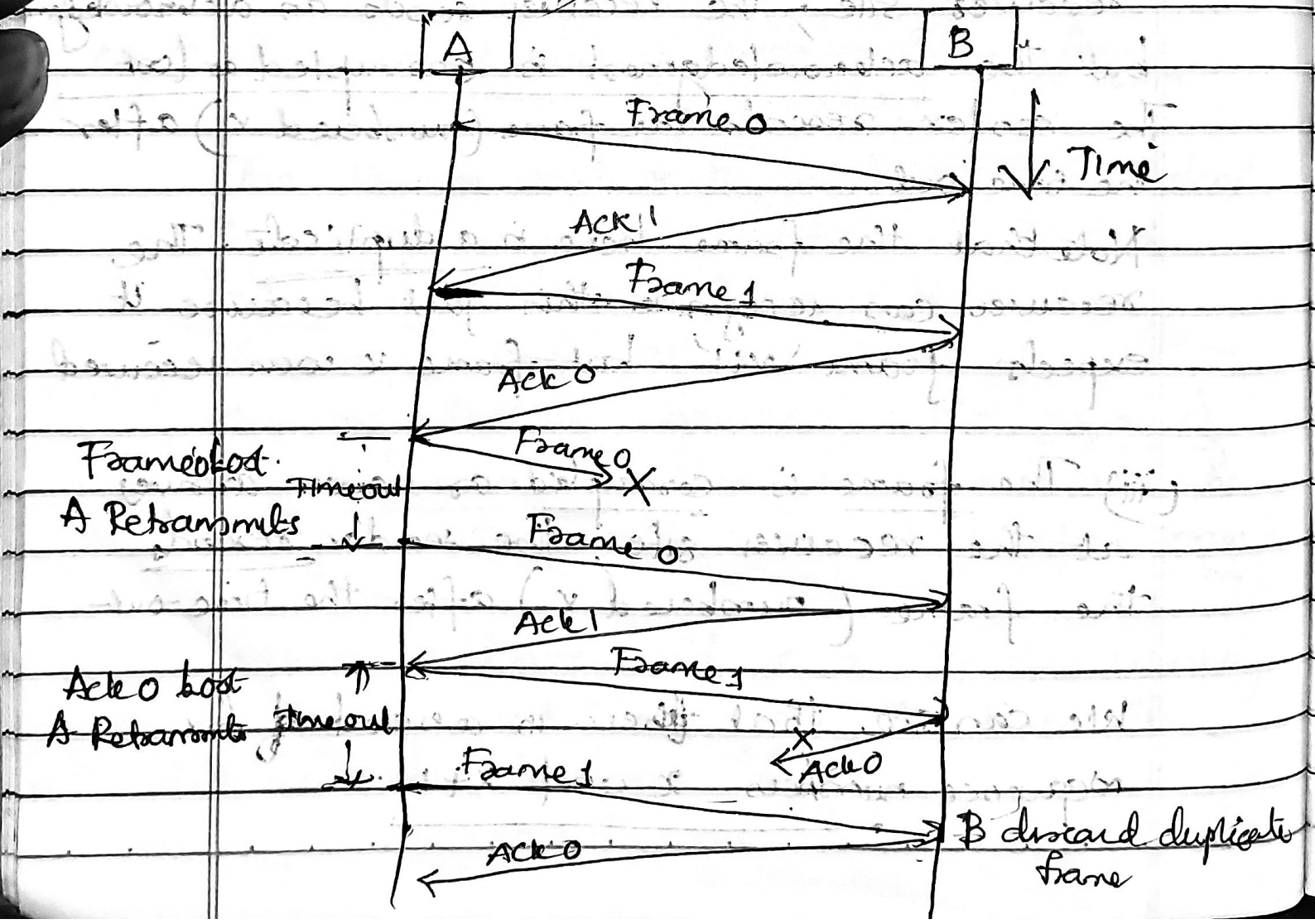
(iii) The frame is corrupted or never arrives at the receiver site; the sender resends the frame (numbered x) after the time-out

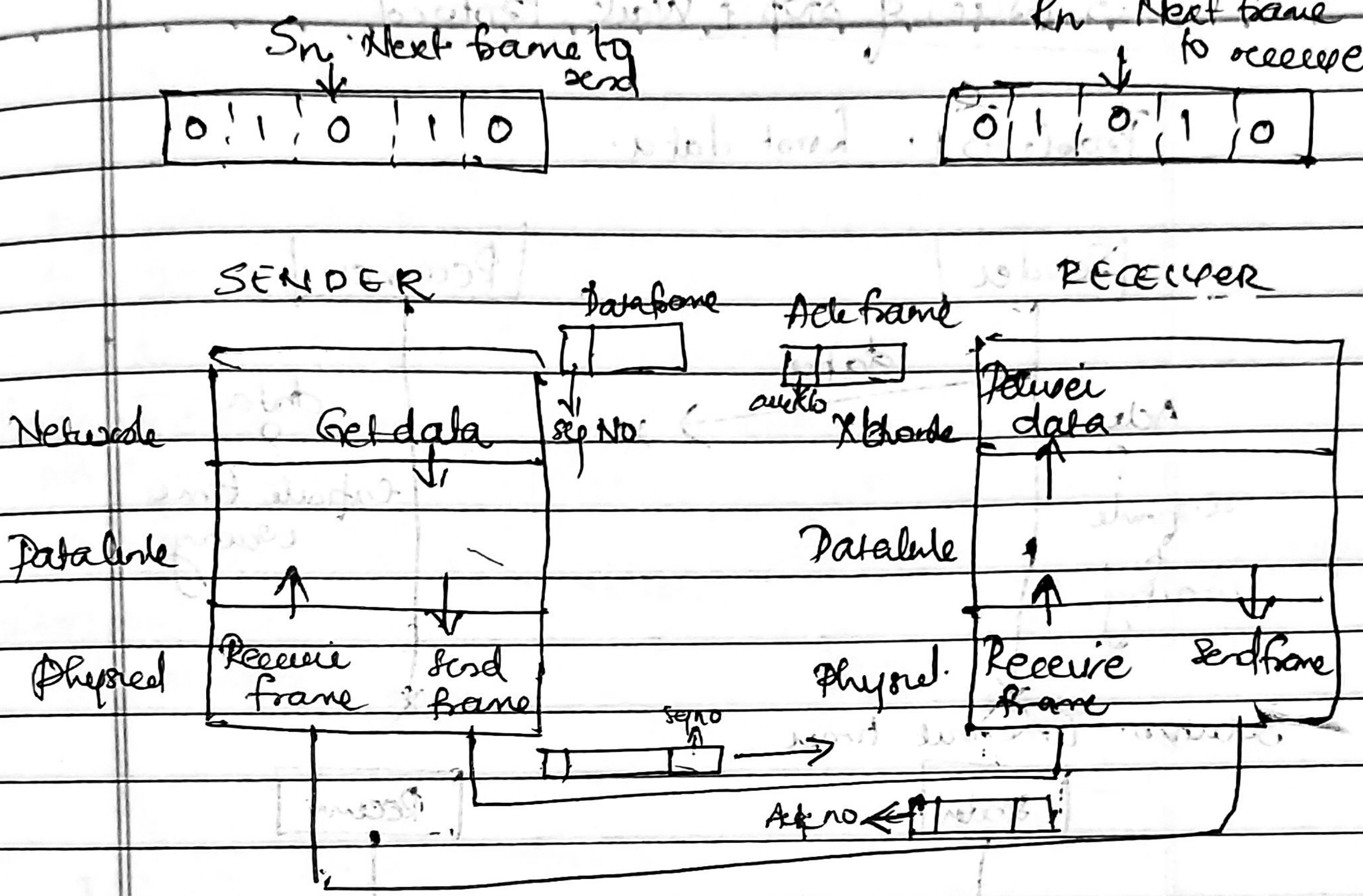
We can see that there is a need for sequence numbers x and x+1.

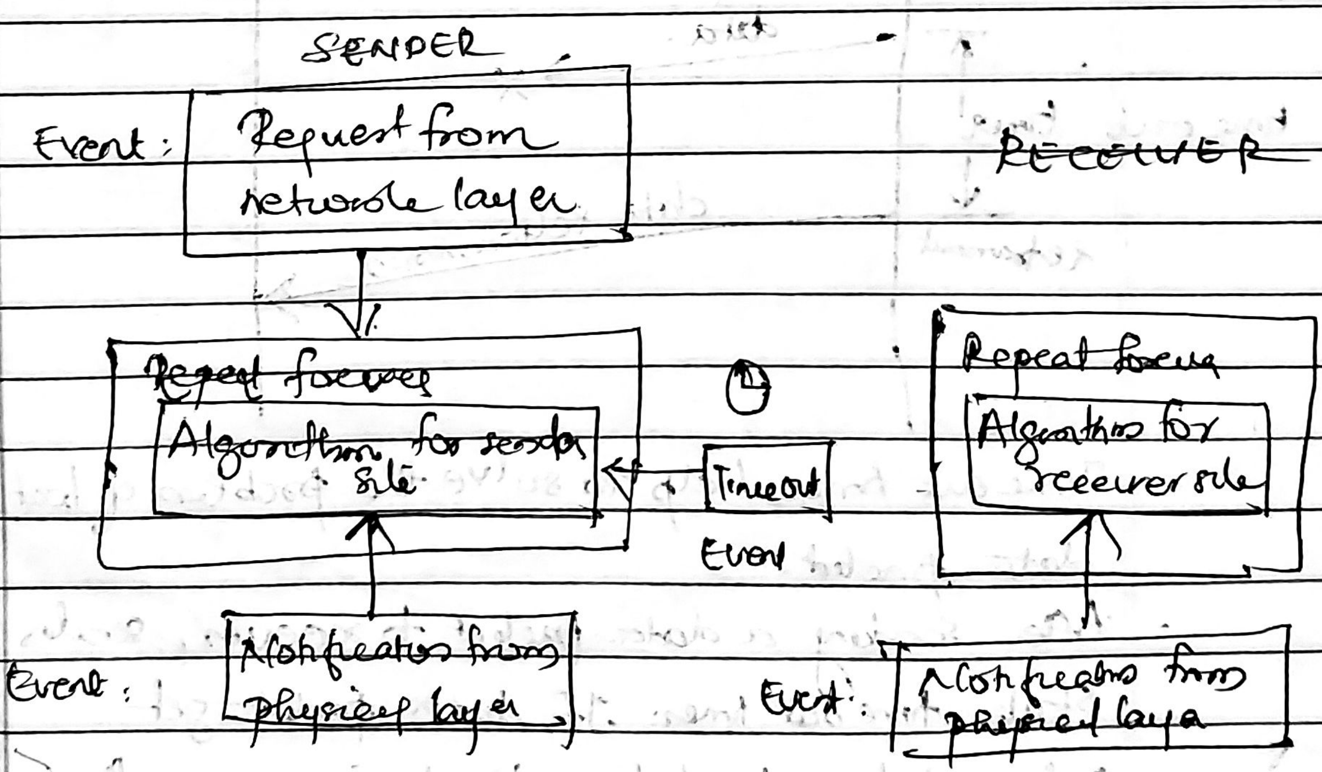
Acknowledgement Numbers

The acknowledgement numbers always announces the sequence number of the next frame expected by the receiver

For example, if frame 0 has arrived safe and sound, the receiver sends an Ack frame with acknowledgement 1 (frame 1 is expected next)



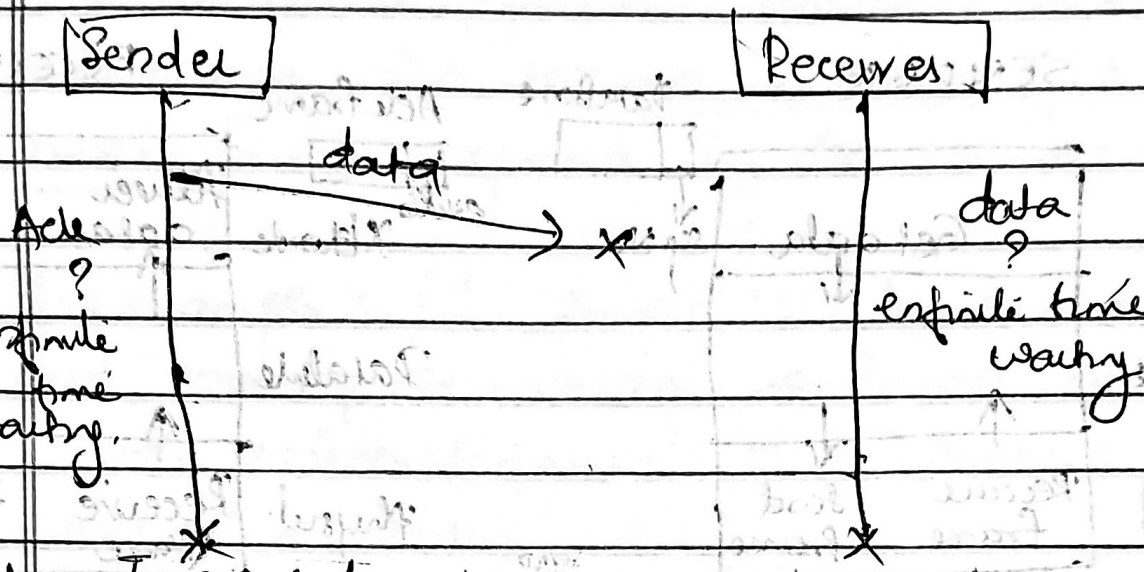

Sn Next frame to send
0 1 0 1 0
Rn Next frame to receive
0 1 0 1 0
SENDER
RECEIVER
Dataframe
Ack frame
seq No
ackNo
Network
Get data
Deliver data
Datalink
Physical
Receive frame
Send frame
seqno
AckNo


SENDER
Event: Request from network layer
Repeat forever
Algorithm for sender site
Timeout Event
Event: Notification from physical layer
RECEIVER
Repeat forever
Algorithm for receiver side
Event: Notification from physical layer

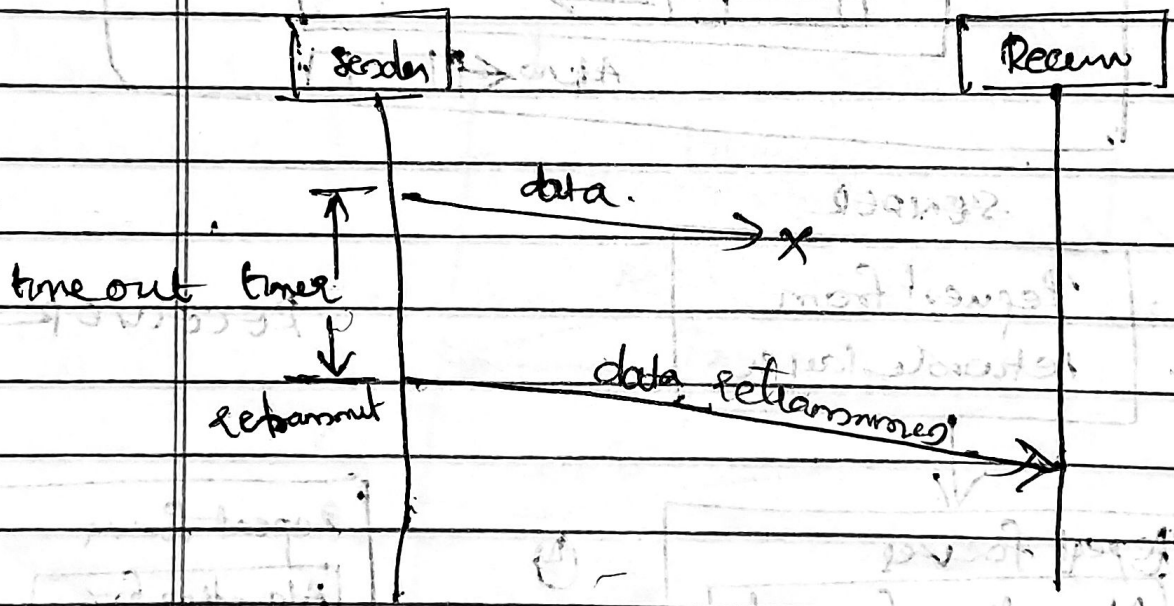
Mondal
Mon | Date 20 | Thu
Tue | Page No.: | Fri
Wed | | Sat

3 Issues of Stop & Wait Protocol

Problem 1: Lost data.



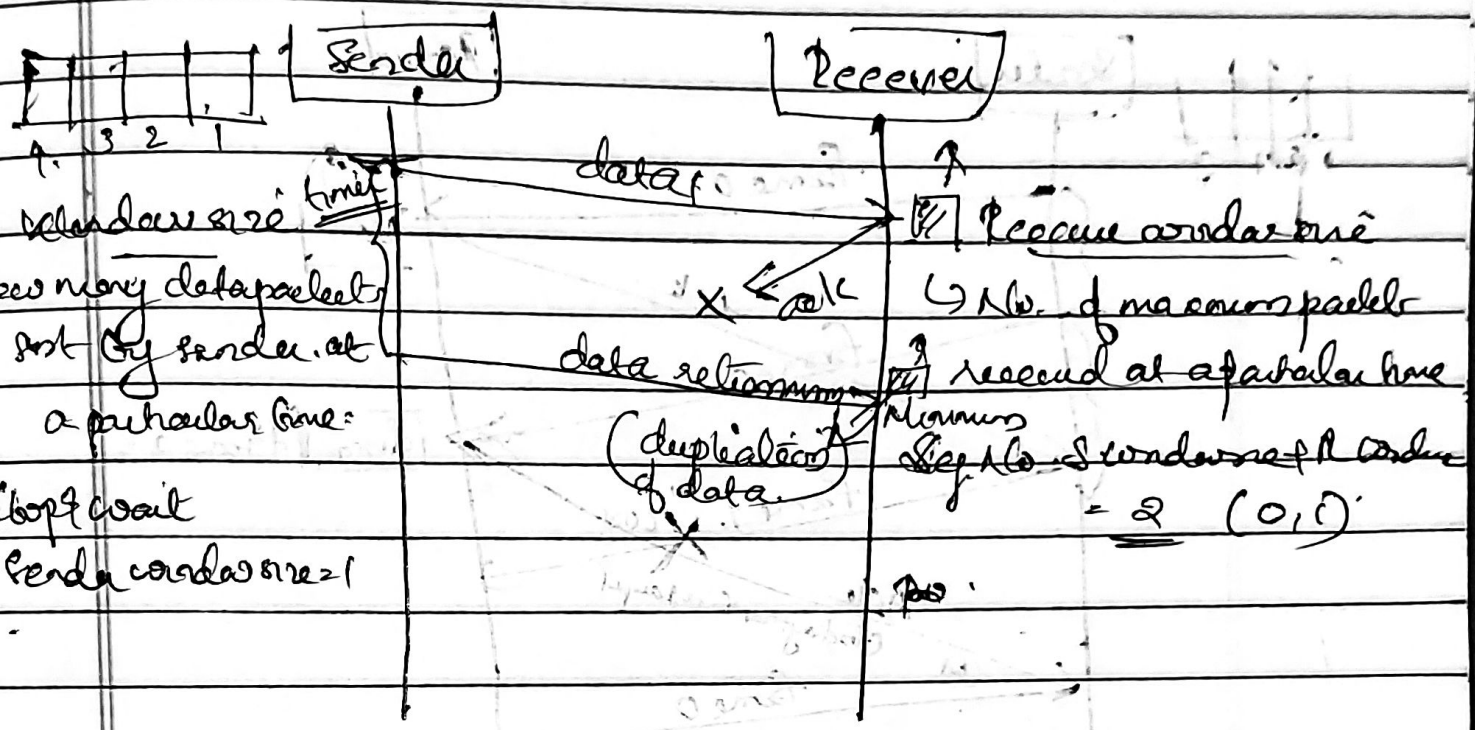
- Solution: Time out timer

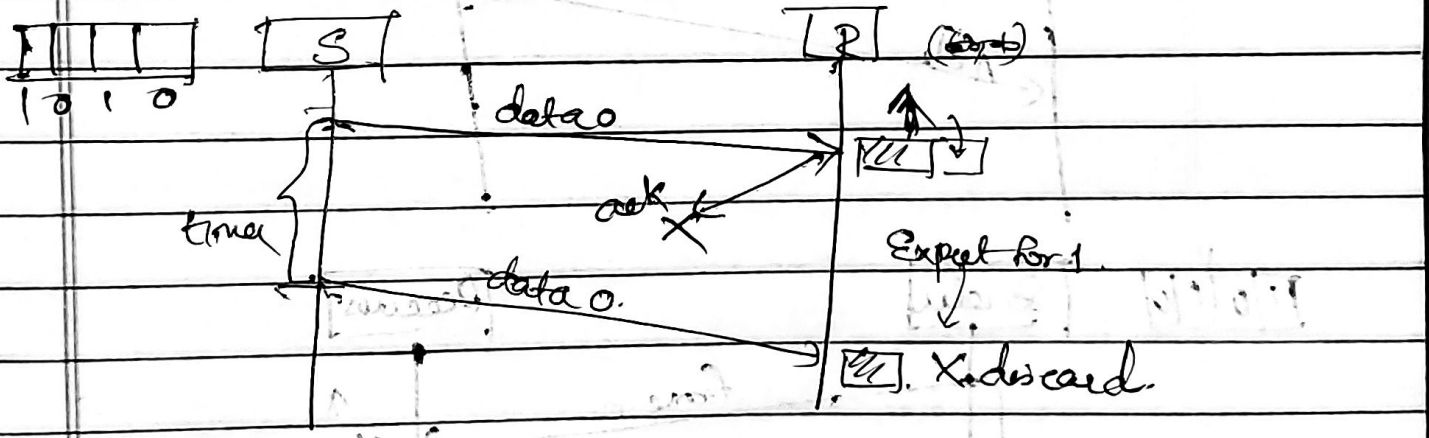


Time out timer help to solve the problem of lost data packet

- After sending a data packet to receiver, sender starts time out timer. If data packet get acknowledgement before the timer expires, the sender stops time out timer. If the timer goes off before receiving the acknowledgement, sender retransmit the data packet

Problem 2: Lost Acknowledgment [Seq. No]





The sequence number on data packet help the receiver to identify the duplicate data packet

- Receiver discards the duplicate packet and resend the same acknowledgment

Problem 3 : Delayed Acknowledgment [Seq-No of Ack]