value	protocol
1	ICMP
2	IGMP
6	TCP
17	UDP
⋮	

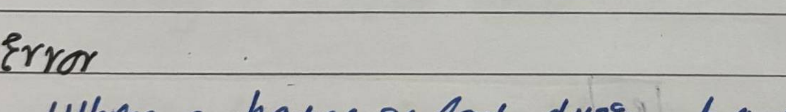
header checksum : used for error detection in 'HEADER' not data

Module-3

* Flow Control (only for conn^n oriented protocol)

Stop-and-wait technique

1. No Error



2. Error

When a packet is lost during transmission, there is no negative acknowledgement.
A timer is defined if no packet recieved by reciever then it ~~tells transmitter ab~~ doesn't send +ve ack.
So transmitter retransmits it until reciever gives +ve ack.

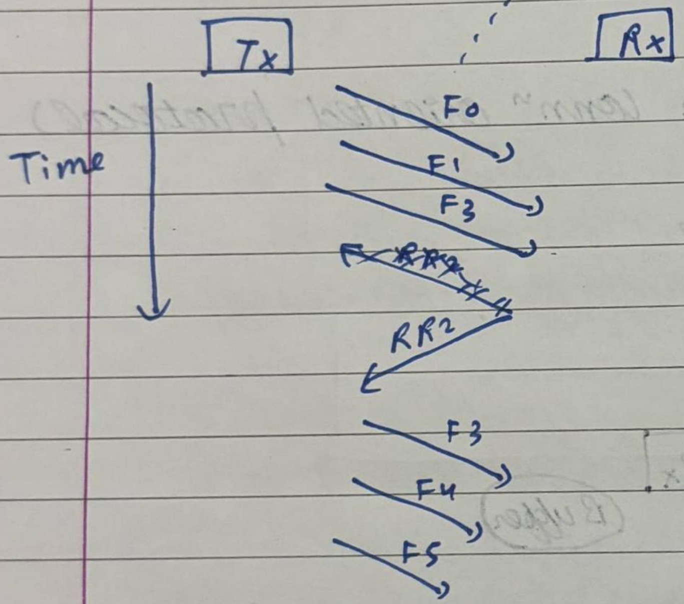
If a damaged packet is send to tr no +ve ack is sent backas to transmitter so transmitter retransmits it.

If ack is lost then transmitter retransmits it so at reciever after checking sequence number reciever detects it's a duplicate packet and handles it accordingly.

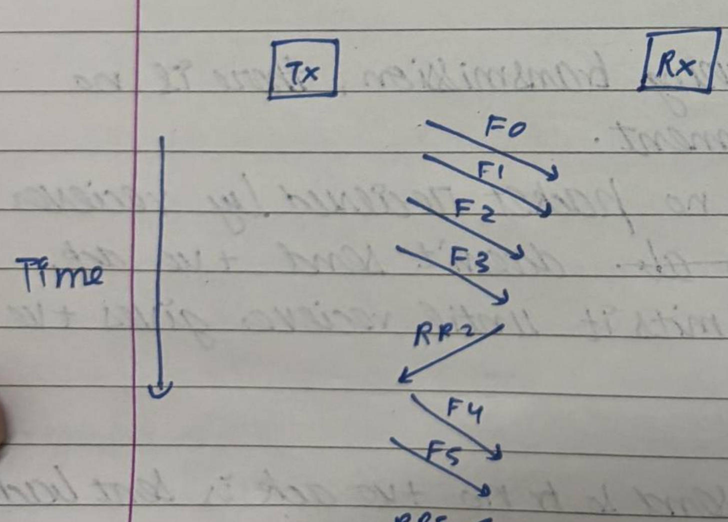
* Sliding Window

1. No Error

→ frames are alway 0-Based indexed

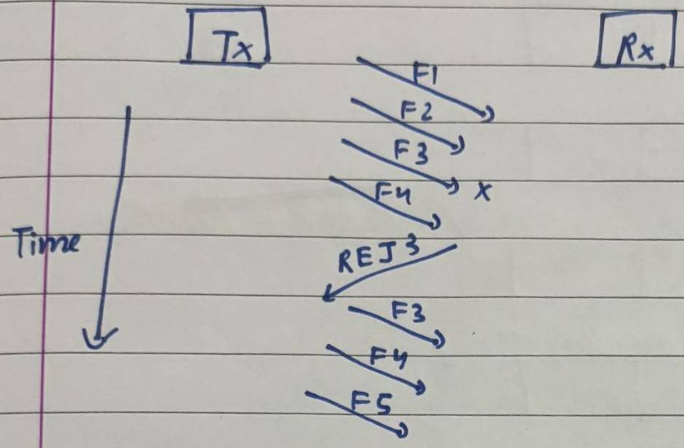


RR → +ve ack
(~~Read~~ Recieved Ready)
~~RR~~ REJ 7 → -ve ack



here ack was till packet 2
so Tx should look what to do about F3 but it sends next frames waiting to get ack for frame 3 as well

2. Error



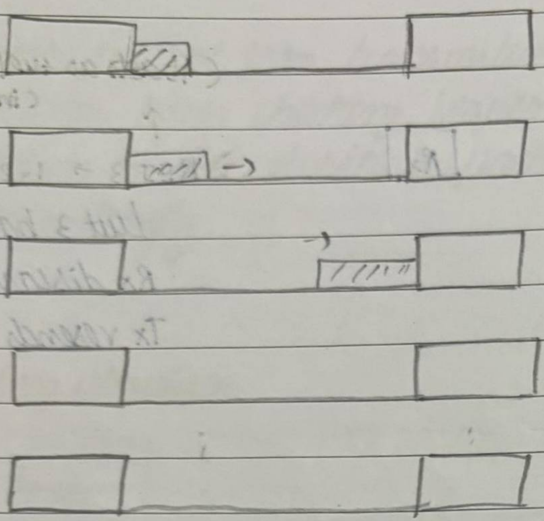
(it acts as +ve/-ve ack both)
(indirectly)

REJ 3 → 1, 2, were successful but 3 has error so Rx discards 3, 4
Tx resends from 3

if a given packet has error so what Rx do is that it sends REJ(x) also it discards all packets it recieves after that until that packet with error was resend. - Tx needs to resend all packets starting from error packet again.

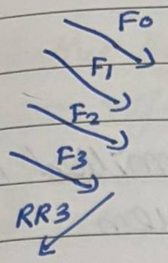
RNR → Recieved Not Ready
ex. RNR5 → Recieved till 5 but don't send more im not ready to recieve more

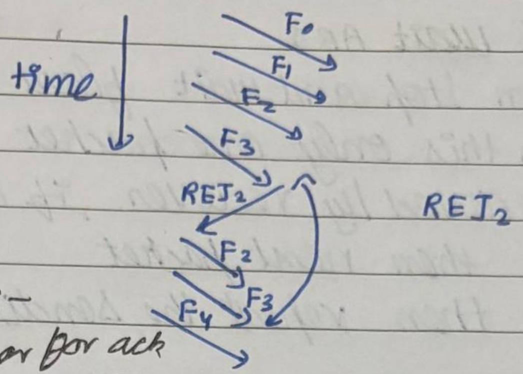
Stop and wait link utilization
Let
transmission time = 1
propagation time/delay = a



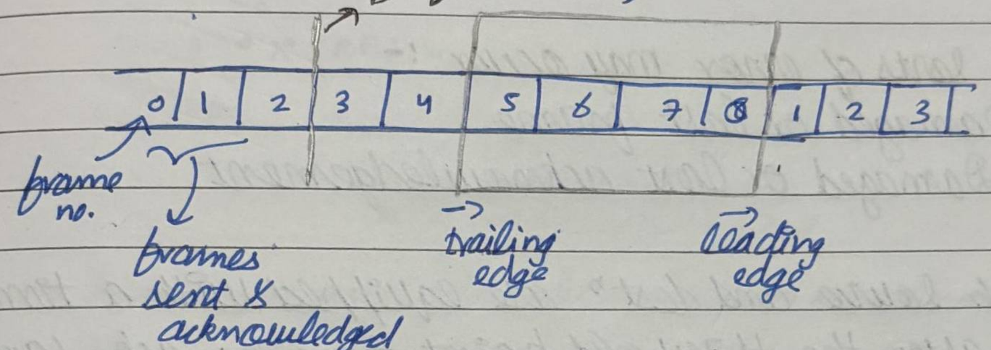
Sliding-window

Positive ack of given frame is equivalent to acknowledgement of previous frames as well.





Sender's perspective :-



Reciever's perspective :-

Error Control :-

Technique for the error detection is based on either some or all of the following ingredients :-

1. Error detection
2. +ve ack
3. -ve ack & retransmission
4. Retransmission after timeout

} collectively referred as ARQ ↳ Automatic, Repeat Request

1. Stop and wait ARQ
2. Go-back-N ARQ
3. Selective reject ARQ

1. Stop & wait ARQ
- Based on Stop and wait flow control
- Here in this only one packet is transmitted at a time
- It is recieved by reciever, if ack is given
  if +ve then next packet
  if -ve then repeat the sending of same packet

→ Two sorts of error may occur :-
  1. Damaged or lost frame
  2. Damaged or lost acknowledgement

- Both source and dest^n are equipped with a timer
- If after the threshold point time no ack came then also packet is resend
- Here in stop & wait ARQ no ack is equivalent to -ve ack

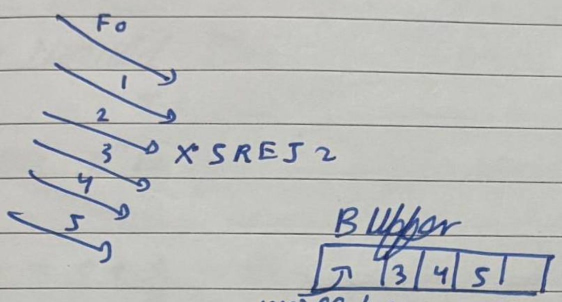
2. Go-back-N ARQ
- Based on sliding window ~~ARQ~~ flow control
  There are two sub cases
1. Damaged / lost ~~frame~~ ack
   a)

If there is a comm^n break b/w transmitter dest^n & reciever then a command packet / frame is sent by transmitter dest^n to reciever where p-bit is set to 1. The transmitter asks the dest^n about the last message. This is used to establish comm^n again.

2. Damaged / lost (-ve ack) REJ
   In this case comm^n definitely breaks

3. Selective Reject - N ARQ

here reject is written as SREJ instead of REJ
only those frames are retransmitted for which negative ack was issued



we'll have 2 when resend

here as SREJ 2 so only frame 2 is resend also the frames after SREJ are stored in buffer as they are correct

8 Cyclic Redundancy Check (CRC)

Let T = (K+n) bit to be transmitted

M = K-bit message (1^{st} K-bit of T)

F = n-bit FCS (last n-bit of T)

p = pattern of n-bit (predetermined divisor)

we would like T/P to have no remainder

$$T = 2^n M + F$$

Divide $2^n M$ by P

$$\frac{2^n M}{P} = \varphi + \frac{R}{P} \quad - (1)$$

Remainder is always 1-bit shorter than divisor. use remainder as FCS

Then, $T = 2^n M + R$ → in modulo 2 arithemetic adding a no. to itself results in zero.

Then $\frac{T}{P} = \varphi + \frac{R}{P} + \frac{R}{P} = \varphi$

(modulo-2 arithemetic is used)

ex. Message M = 1010001101 (10-bit)

Pattern P = 110101 (6-bit)

FCS, R = to be calculated (1 bit less than p ⇒ 5bit)

1. $M \times 2^n$ n = 5-bit

$2^5 = 32 = 100000_2$

Let $M \times 2^n$

$\Rightarrow 101000110100000_2$

2. divide by P

$$\frac{2^n \times M}{P} = \frac{101000110100000}{110101}$$

$= 1101010110 = \varphi$

R = 01110 = FCS

3. $2^n M + R \rightarrow T$ (transmitted)

1010001101000000
+ 01110
101000110101110 → transmitted message

4. now divide T/P

110101 ⟌ 101000110101110
0 ← R

at the reciever if as well T will be divided by P if remainder does not comes 0 then there has been an error

(NOTE: This entire question is in modulo 2)

* multiplexing
1. Time division multiplexing → digital signal → asynchronous → synchronous
2. Frequency division multiplexing → analog signal