# CONCEPTS AND INTRO

- Computer N/w Usage:
  - Data Communication
  - Resource Utilization

- Importance of CN
  User views
  - Throughput (Efficiency)
    # of correctly xferred bytes / time
  - Reliability
    - Not manipulated / lost / corrupted / duplicated
    - Ensures integrity
  - Security — Authorization
    Encryption
  - Robustness: Provision of sustainability
  - Scalability: Easy to expand

- Classification
  - Connection Oriented / Less
  - Reliable / Not

  MAIN

  - Wired / Less
  - Link type
  - Area or Region
  - Topologies

P.S: All these are independent attributes of one another

Cellular N/w : - Wireless
- Conn. Oriented
- Unreliable

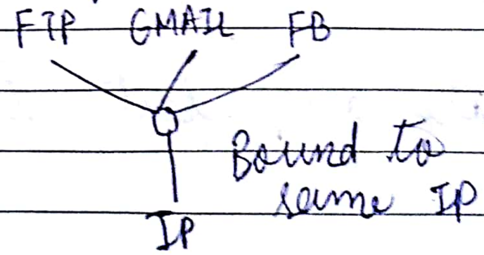
✓ TASK I: Advantages of layered approach

OSI Layer
- Lower layer provides service to upper layer

- Current layer utilizes the services of the layer lower to it.

- Data flows are opposite at transmitter and receiver

- Design issues parameters required for network to communicate
  - Unique addressing
  - Efficient routing
  - Security (At all layers)
  - Error control
  - Flow control
  - Establishment of channel
  - Congestion control
  - Multiplexing : Multiple apps on one node




TASK II : Design issues - Tenanbaum

IP address (Internet Protocol) N/w Layer
192.168.1.1
- 32-bit address
- Logical address - Identify comp. in n/w

MAC address (H/w address / Physical Address)
F1:20:13:3C:B0:02
- 48-bit address
- Data-Link layer - Identify comp. in n/w

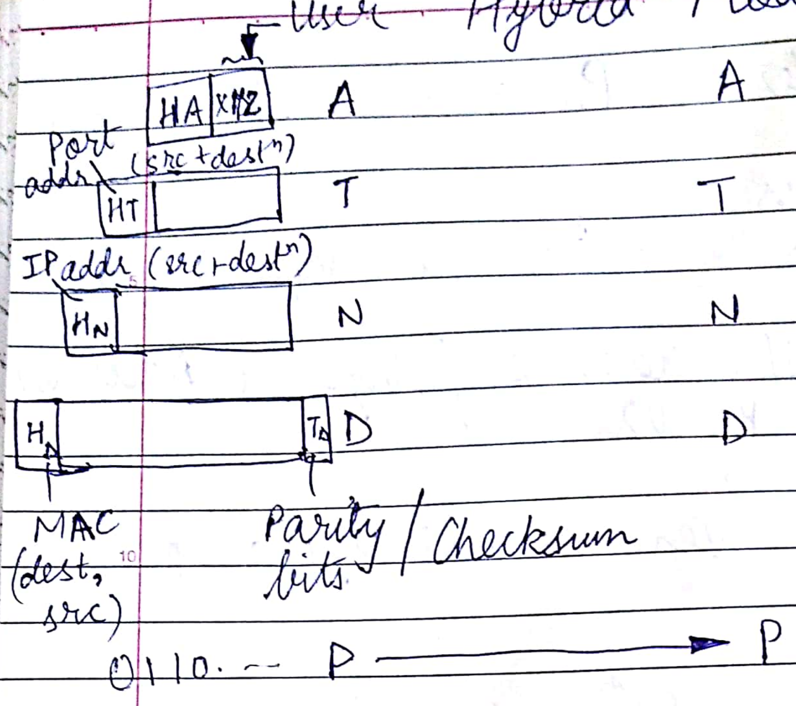
Port address
- identify an application in system
- 16 bit address
- Unique in the system
- Transport layer

TASK III : Role of MAC + IP and problem with only IP

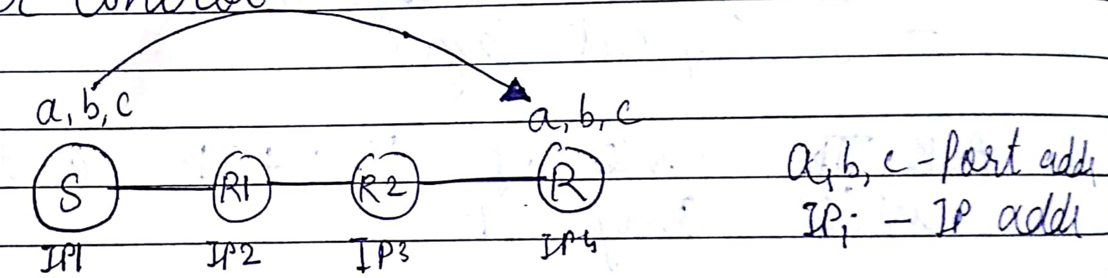
| Layer | Data Unit |
|---|---|
| Appⁿ | message |
| Transport | segment |
| N/W | packet |
| DL | frame |
| Physical | bits |

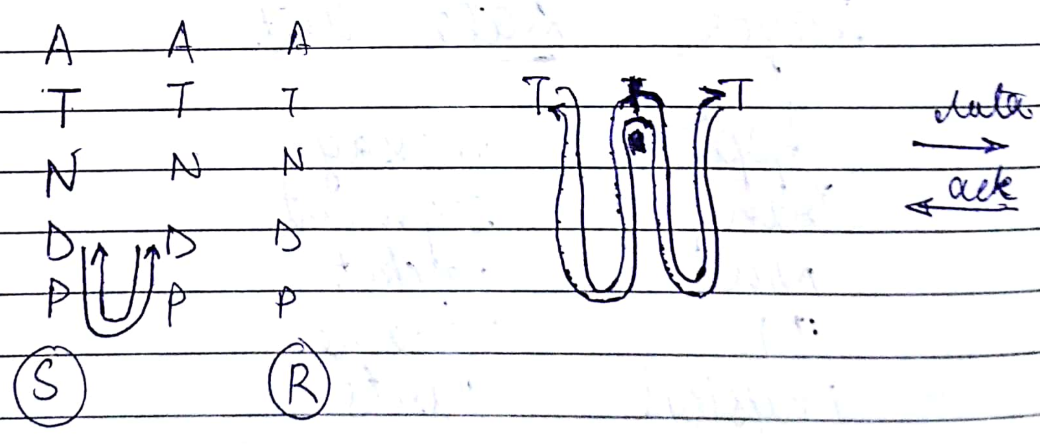
MAC sub layer
- Deals with issues of access control

# Hybrid Model




- Error Control




Transport : { End-to-end commn
Process-to-process "

Network : { Host-to-host commn
Source-to-Dest "

Data Link : { Hop-to-hop comm$^n$
{ Node-to-node "

Transport layer error control is mandatory, DLL error control is for optimization

| A | R1 | R2 | B |
|---|---|---|---|
| A | | | |
| T | | | |
| N | | | |
| D | | | |
| P | | | |
| X | | | |

error occurs here.

- If DLL level error control not present, reliability is still achieved but is too exhaustive.

∴ There is no meaning in sending wrong data ahead and after reaching B, error is detected; ~~and~~ which would have been done at R1 itself if DLL error control was provided

# Datalink Layer

- 'out of order delivery ~~order~~' : ~~problem~~ not a problem at this layer

Ethernet : 1500 bytes / frame

- ~~Framing~~ Framing helps in error control

FRAMING

(I) time delay
  - exact time delay generation is difficult.
  - throughput decreases due to delay

(II) character count

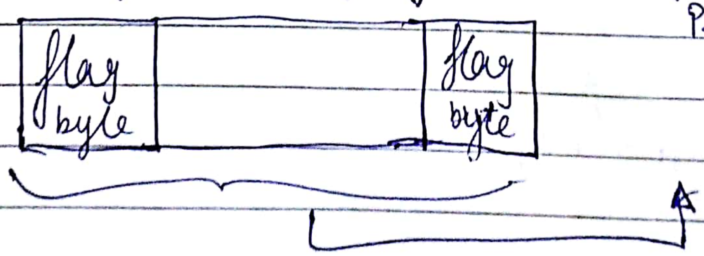
110 → 010 } error

$\underbrace{\boxed{6}\ 2\ 3\ 4\ 5\ 0}_{6}\ \underbrace{\boxed{6}\ 7\ 8\ 9\ 3\ 4}_{6}\ \underbrace{\boxed{7}\ 5\ 6\ 0\ 2\ 2\ 3}_{7}$

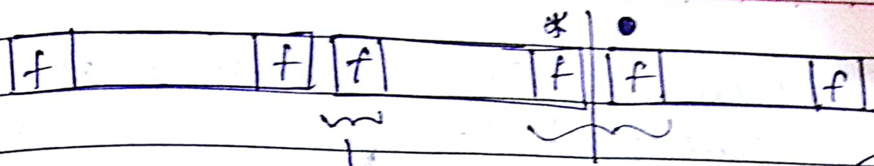
- when ~~cha~~ count bit gets corrupted, propagated error occurs thus basic aim of framing not achieved.

P.T.O

(III) byte stuffing




- works only with 8 bit formats




if gets corrupted ∴ when next two frames occur consequently,

algorithm knows then * is invalid & new count started from •

- receiver is misguided if flag byte appears in data

Sol$^n$: Escape characters are used

(1) frame: $\boxed{\text{flag}}$ a b esc flag d . e $\boxed{\text{Flag}}$

Data: { a b flag d e }

(2) Data [ a b c esc flag d e esc ]

Frame: $\boxed{\text{Flag}}$ a b c esc esc esc flag d e

esc esc $\boxed{\text{flag}}$

## (IV) bit stuffing

$\boxed{01111110}$ 0111110 0 1 0 0 1 1 0 0 1 1 1 1 1 0 0 0 1 $\boxed{01111110}$

0 0

Fixed

whenever five 1's together,

111110

append 0

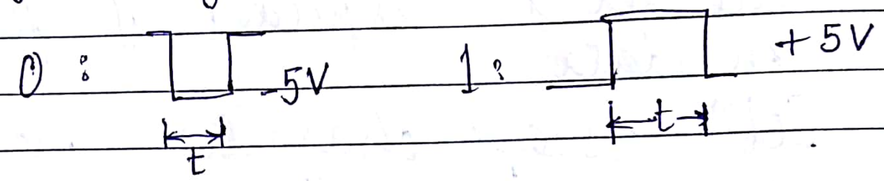
Date / /

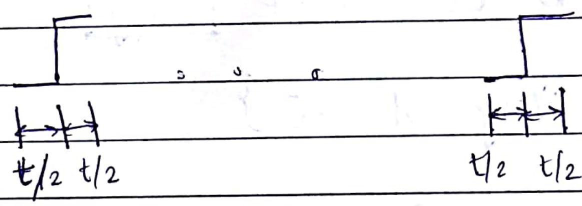
At receiver, after occurance of five 1s, remove the

01111110010

0111110010

(V) physical symbol violation




- Use different line code than that used for data

- aquiring hardware is required
- we need to access physical layer for this and this violates layering approach
- we cannot ask receiver to use the line code similar to ours
- theoritical concept

Purpose of Framing
- Hardware may not support large amount of data. Thus, we need to divide into chunks and transfer without errors

Self study: CRC: Cyclic Redundancy Check

# ERROR CONTROL

- Error detection
- Error correction

## (1) Hamming distance

1 ⟨ 0011, 1100, 0111

0011, 1100 > 4

1100, 0111 > 3

3 ⟨ a - 001, b - 100, c - 110

a - 001, b - 100 > 2

b - 100, c - 110 > 1

cab
110001100
→ cac

Hamming distance = 1 — This shows that 1 bit error cannot be detected

2 ⟨ a - 0011, b - 1100, c - 1111

a - 0011, b - 1100 > 4

b - 1100, c - 1111 > 2

c a b
1111 0011 1100
error

Hamming distance = 2 => 1 bit error detection

n+1 => n

If n bits change, 1 bit is still there by which we can differentiate

- Correction

a - 000 — 001, 010, 100

b - 111 — 110, 101, 011

a a b

000 000 111

1

Allowing 1 bit error

010 — More similar to 000 because it has to change only 1 bit

| Correct | | Hamming distance |
|---|---|---|
| 1 | => | 3 |
| 2 | => | 5 |

a - 0000 — 0011 $\qquad n \Rightarrow 2n+1$

b - 1111 — 0011

a - 00000 — 00011

b - 11111 — 00111

Error detection is better option because they are less expensive w.r.t to sending data per character.

Error correction is required when we have too much noise channel.

## (II) Hamming Code

data : 0111001101

data bits
check bits $= 2^n \; ; n \geq 0$

~~0~~ 1 2 3 4 5 6 7 8 9 10 11 12 13 14

0 1 1 1 0 0 1 1 0 1

Even parity

1- 011010 = 1
2- 011011 = 0
4- 111101 = 1
8- 001101 = 1

1 0 1 1 1 1 1 0 0 1 1 0 1
1 2 3 4 5 6 7 8 9 10 11 12 13

1
1 0 ~~0~~ 1 1 1 1 1 0 0 1 1 0 1
1 2 4 8

1 = 111010 = 0
2 = 111011 = 1
4 = 111101 = 1
8 = 001101 = 1

~~1110~~ ~~1011~~

0
11~~1~~0
1101
0001