Revision:

Switching always send data in order.

10|2|25 Mon (sir)

1st
* Physical address (mac address) [it is compulsory to connecting]
(It depends on h/w that's why it is physical address)
48 bits (Hexadecimal code)

Format 3A : 2B : 2A : 6A : 3C : 2C :- 48 bits
8bit 8bit 8bit 8bit 8bit 8bit

→ Assign by Datalink layer

2nd
* IP address (use for identify device)
It is logical address [not depend on h/w]
→ It assigns by ISP (Internet service provider)
Ex:- JIO

assign by Network Layer

→ We need Globaly Unique address & it is IP address (physical address is not globally accesible)

→ IP add :- IPV4 , IPV6. (V5 is not successed)

↓ size :- 32 bit

IPV4 :- Format :- 190.10.25.10 → 32 bit
(8 8 8 8)

max val :- 1111 1111 1111 1111 → 255 decimal >> total :- 256
min val :- 0000 0000 0000 0000 → 0 decimal

means 256.10.25.10 is a wrong.

IPV6 :- 128 bit (2^{128} X number of IPs).

3rd

* Port address (size :- 16 bit) 0 to 355~~8~~35
(Process Id) it will identify Application.

→ assign by Transport layer

Note → device identify by physical & IP address

→ Application ~~ed~~ identify by por

18/2/25 Tuesday

→ data transmits into ↓ (form of) Electro magnetic signal.
→ data may be currupt or lost while transmission is called Error Detection.

(1) Single bit error.

Sender		Received
1010101	⟶	101010[0]

only one bit is error while transforming

(2) Burst Error.
2 bit or more bit corrupted
Length of error :- 1st corrupted bit – last corrupted bit
here in figure :- 8 bit

→ before Receiving data Receiver must check data is wrong or right, if it is ✓ then accept, if ✗ then reject)
→ In this case, Sender Retransmits the data (when sender send same data again its called Retransmission)

* Redundancy
→ To detect or correct errors, we need to send extra (redundant) bits with data.

(i) → error detection + error correction = Error control

Error detection → Informs to receiver that there is an error in data
methods → CRC, checksum, parity bit

(ii) Error correction → detects error as well as corrects the error.
To correct :- How many bits are currepted and the location of currpted bit.

→ If location 2, 4, 6 bit are corrupted then if there is 0 → change to 1 & vice versa.
101010 ⟶ 11 11 11 (2, 4, 6)

Error detection	Error correction
simple	complex

* Block coding.
Data ⟶ generator (code) ⟶ Data + code (sended)

1010101 — 101010 ↓ og data, 1 ↓ code (to verify og data)

Calculate code → attach code with data → send

→ To calculate code use Block coding.
divide data into equal parts.
data :- [1010] (1st) [1010] (2nd) [1100] (3rd) :- k = 4 (length of one block)

no of blocks = 2^4

Calculate code for each block

length of codeword dataword = 2^k

$n = k + r$
data + code

→ (block) data = datawords = 2^k
→ $k + r = n$ = codeword, where n is higher than k
→ Here, $k = 4$, $r = 1$, $n = 4 + 1 = 5$, dataword = 16 (2^k)
→ 2^n = codeword = 32
→ codeword required exactly same as dataword.
→ Required codeword - wasted = $2^n - 2^k$

* CRC (Cyclic Redundancy check)

→ K=2

Dataword = $2^k = 2^2 = 4$ datawords.

1 :- 01
2 :- 10
3 :- 00
4 :- 11

→ K=3

dataword = $2^k = 2^3 = 8$ datawords.

001 011
010
100
110
101
101
110

* Parity bit (Error detection) method.

Parity bit
- Even
- Odd

no of 1 in data must be Even

data :- 1011 Here 3 = Odd

∴ 1011 (1) = 4 = Even.

↑ code.

data :- 1010 ∴ Parity bit :- 1010 (0)

यदि Even 1 होय तो code = 0

नहीतर Even असताना code = 1.

→ Odd purity works with Same but Here, We consider Even purity.

→ C(n, K)

Simple Purity check code C(5,4).

0111 ← codeword.

→ Sender → Receiver.

Codeword.

0111 ← if 0 then ✓ otherwise X.

011 | 1
data word | code word

Suppose data :- 0011 ← it is 1 then data is X.

disadvantage :- It only checks Single bit Errors.

Suppose :- Sender 10111 → Receiver 00110 ← ✓ but data is wrong.

wrong data.

* CRC

111010 → 111010 | 1 (left most bit shift to right)

Ex :- 1001 + code = codeword.

⊕ :- XOR. if 0 0 → 0
1 1 → 0
0 1 → 1
1 0 → 1

Length of divisor - 1 = 000 (data will not change)

Length of CRC :- 1 of divisor - 1 = 4-1 = 3.

Sender :-

1010
1011 | 1 0 0 1 0 0 0 ← for CRC. (initially we don't know CRC, so 000)
1 0 1 1
0 0 1 0 0
0 0 0 0
0 1 0 0 0
1 0 1 1
0 0 1 1 0
0 0 0 0
0 1 1 0

avoid this

After Removing left most bit if 1st bit is 0 then og divisor change to 0000

← CRC :- 110

Receiver:-

codeword :- 1001110

```
          1010
1011 | 1001110
       1011
       -----
      x0010
        0000
        -----
       x01011
         1011
         -----
        x00000
          0000
          -----
         x0000  <- 0 then data is correct.
```

Ex :-

```
        1100
1011 | 1110000
       1011
       -----
      x01010
        1011
        -----
       x00010
         0000
         -----
        x00100
          0000
          -----
         x0100
          CRC
```

```
        1100
1011 | 1110100
       1011
       -----
      x01011
        1011
        -----
       x00000
         0000
         -----
        x00000
          0000
          -----
         x0000
```

24/2/25 Mon

Error detection method.

* checksum (4 bit) (page 299 Ex :- 10.22).

Sender

data :- 7, 11, 12, 0, 6, 0 ← convert data in numbers.

Sum :- 7 + 11 + 12 + 0 + 6 + 0 = 36.

binary :-

32	16	8	4	2	1
1	0	0	1	0	0

} in 4 bit we can write only 15 max. (1111)

100100 (this is 6 bit, so convert in 4 bit)

∴ 100100 → 10 (left most bits)

```
   0100
+    10      (binary addition)
  -----
   0110
1's comp. → 1001   (conversion neg no)
```

Use of compliment in binary.
:- To Represent negative number
4 = 100. -4 = 011

1001 :- 9 in decimal

checksum :- 9

Logic :- if sending 36 & adding neg. (-36) ans will 0.

Receive

7 + 11 + 12 + 0 + 6 + 0 + 9 = 45
↑ checksum

32	16	8	4	2	1
1	0	1	1	0	1

= 45.

```
   1101
+    10
  -----
   1111  -- compliment -->  0000  (data is correct)
```

Hamming code (Pg-274)

* $n = k + r$

- n ↑ length of code word
- k ↑ length of data word
- r ↑ length of code

→ Suppose data :-

①	②	③	④	⑤	
1	0	1	0	0	1
1	1	0	0	1	1	0
	↑ diff	↑ diff	↑ diff	↑ diff	↑ diff	

minimum difference of 2 corresponding bits = Hamming distance

To Identify Hamming dis ↓ go Perform XOR

```
  1 0 1 0 0 0 1
⊕ 1 1 0 0 1 1 0
---------------
  0 1 1 0 1 1 1   ← 5 times one so Hamming distance is 5.
```

Ex :- $K = 2$, no of datawords $= 2^k = 2^2 = 4$

①	0	0	1
②	1	0	0
③	1	1	0
④	0	1	1

b/w 1 & 2 → 0 0 1 ⊕ 1 0 0 = 1 0 1 ← 2

b/w 1 & 3 → 0 0 1 ⊕ 1 1 0 = 1 1 1 ← 3

b/w 1 & 4 → 0 0 1 ⊕ 0 1 1 = 0 1 0 ← 1

min of this is minimum Hamming distance.

→ $d_{min} = S + 1$ (S ↑ no of bit)

Require minimum Hamming code to detect error.

Error detection

data : S if S error then S+1 Hamming dis is Require

5 Error → 6 Hamming code

Error correction

1 Error = 1 bit Error. $d_{min} = 2t + 1$

where if $t = 1$ (no of Errors)

$d_{min} = 3$ [2(1) + 1]

Correct t Error, we need $2t + 1$ minimum Hamming distance.

→ Hamming dis is require to get How many Error in CRC, checksum or parity bit.

→ In parity bit method we can detect only 1 bit Error (single bit). so to solve this we will see 2d parity check (2 dimension)

Ex :- (Pg-285)

column wise ↓ / raw wise →								
	1	1	0	0	1	1	1	1
	1	0	1	1	1	0	1	1
	0	1	1	1	0	0	1	0
	0	1	0	1	0	0	1	1
	0	1	0	1	0	1	0	1

Advantage :- It checks raw wise & column wise. If 2 bits are change then raw parity & Column parity get effected so it detects "Burst" Error also.

disadv :- Even no of bits changes in same raw & same column so it can't detect.

In above Ex If we changes underline 1's then parity is not effected. So It is disadvantage.