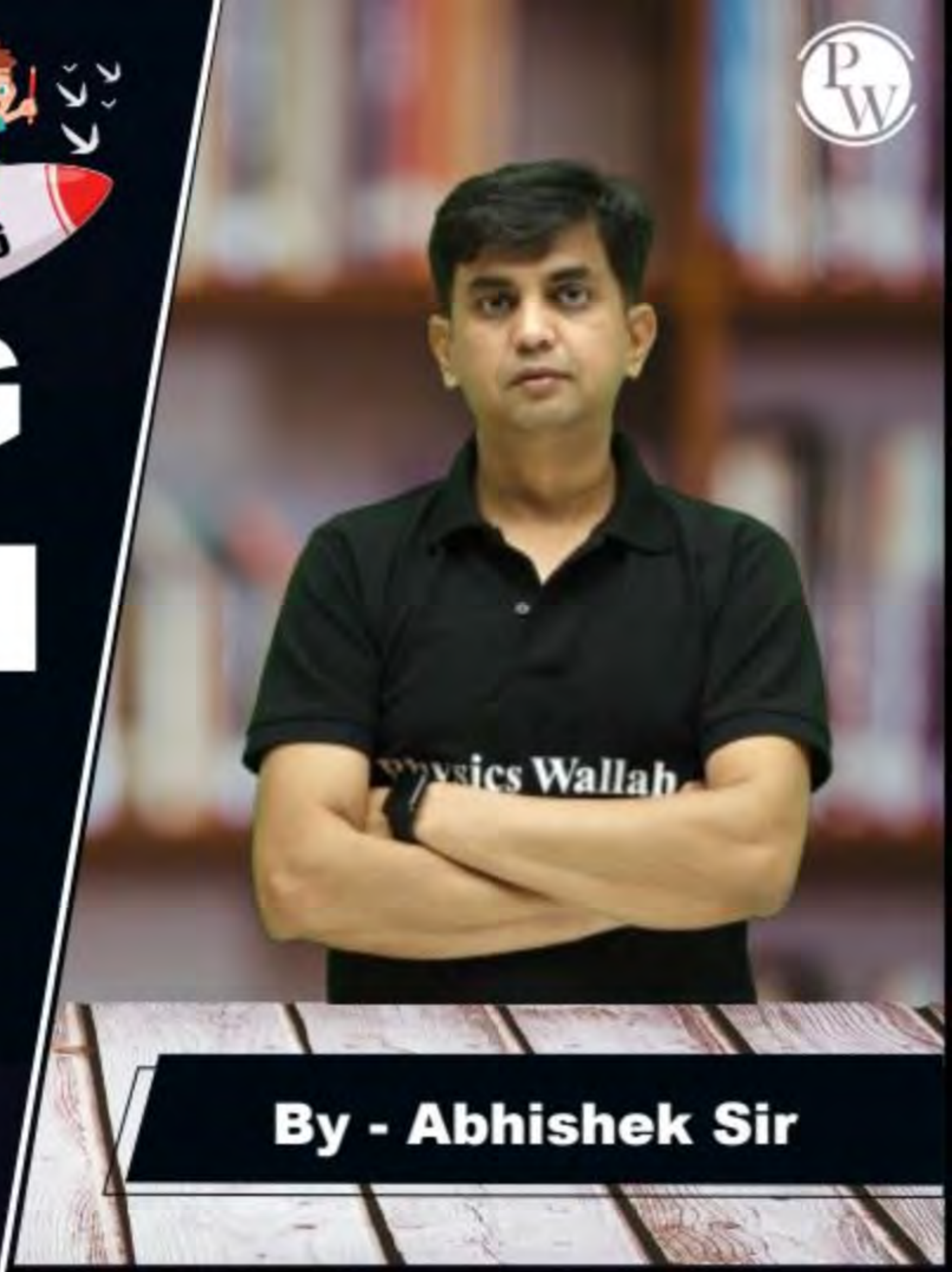
CS & IT ENGINEERING

Computer Network

Error Control

Lecture No. - 05



By - Abhishek Sir

Recap of Previous Lecture

Topic CRC

Topics to be Covered

Topic Checksum

Topic Hamming Distance

ABOUT ME

Hello, I'm Abhishek

- GATE CS AIR - 96
- M.Tech (CS) - IIT Kharagpur
- 12 years of GATE CS teaching experience

Telegram Link : https://t.me/abhisheksirCS_PW

@ABHISHEKSIRCS_PW

Topic : Checksum

⟹ LRC

→ Error detection technique, No any error correction

→ Both sender and receiver must agree on same :

1. Block size (n bits)
2. Number of blocks (k blocks) [Including checksum field]

TCP/UDP/IPv4 ⟹ Block size = 16 bits

Topic : Checksum

→ Two method :

1. Parity Word
   [Based on Modulo-2 Arithmetic]

2. Sum Complement (default)
   [Based on One's Complement Arithmetic]

Topic : Parity Word

Parity Word : Checksum

→ n bit modulo-2 sum of all n bit words (blocks)

Topic : Parity Word

Block Size = n

Receiver		Sender	
	Block 1		Block 1
	Block 2		Block 2
	Block (K – 1)		Block (K – 1)
Block K	Checksum	Block K	0 0 0 .... 0 0 0
	Result		Checksum

Data

n-bit

n-bit

Topic : Parity Word

Result : [Computed at receiver]

if **Result == ZERO** :

then Receiver concluded "**No any error detected**"

else

Receiver concluded "**Error detected**"

Example 1:-

Suppose block size = 5
No. of blocks = 6

Sender

1 0 1 1 0
0 1 1 0 1
1 1 0 1 1
1 0 0 1 1
0 1 0 1 0
0 0 0 0 0

Checksum = 1 1 0 0 1

Example 1:-

Suppose block size = 5
No. of blocks = 6

Sender

1	0	1	1	0
0	1	1	0	1
1	1	0	1	1
1	0	0	1	1
0	1	0	1	0
0	0	0	0	0

Checksum = 1 1 0 0 1

Transmitted Data : 10110 01101 11011 10011 01010 11001

Example 1:-

Suppose block size = 5
No. of blocks = 6

Receiver	Sender
1 0 1 1 0	1 0 1 1 0
0 1 1 0 1	0 1 1 0 1
1 1 0 1 1	1 1 0 1 1
1 0 0 1 1	1 0 0 1 1
0 1 0 1 0	0 1 0 1 0
1 1 0 0 1	0 0 0 0 0
Result = 0 0 0 0 0 ⇒ No any Error detected, Accept the data	1 1 0 0 1

Transmitted Data : 10110 01101 11011 10011 01010 11001
Received Data : 10110 01101 11011 10011 01010 11001

Example 1:-

Suppose block size = 5
No. of blocks = 6

Receiver	Sender
1 0 1 1 0	1 0 1 1 0
0 1 1 0 1	0 1 1 0 1
1 1 0 1 1	1 1 0 1 1
1 0 0 1 1	1 0 0 1 1
0 1 0 1 0	0 1 0 1 0
1 1 0 0 1	0 0 0 0 0
Result = 0 0 0 0 0	1 1 0 0 1

Transmitted Data : 10110 01101 11011 10011 01010 11001
Received Data : 10110 01101 11011 10011 01010 11001

Topic : Linear Code

$d_1 \, d_2$
$d_3 \, d_4$
$c_1 \, c_2$

Parity Word (Checksum with Even Parity) and 4 data bits :

Data	→	Codeword	Data	→	Codeword
$d_1d_2d_3d_4$	→	$d_1d_2d_3d_4C_1C_2$	$d_1d_2d_3d_4$	→	$d_1d_2d_3d_4C_1C_2$
0 0 0 0	→	0 0 0 0 0 0	1 0 0 0	→	1 0 0 0 1 0
0 0 0 1	→	0 0 0 1 0 1	1 0 0 1	→	1 0 0 1 1 1
0 0 1 0	→	0 0 1 0 1 0	1 0 1 0	→	1 0 1 0 0 0
0 0 1 1	→	0 0 1 1 1 1	1 0 1 1	→	1 0 1 1 0 1
0 1 0 0	→	0 1 0 0 0 1	1 1 0 0	→	1 1 0 0 1 1
0 1 0 1	→	0 1 0 1 0 0	1 1 0 1	→	1 1 0 1 1 0
0 1 1 0	→	0 1 1 0 1 1	1 1 1 0	→	1 1 1 0 0 1
0 1 1 1	→	0 1 1 1 1 0	1 1 1 1	→	1 1 1 1 0 0

Topic : Sum Complement

Sum Complement : Checksum

→ n bit one's complement of the
one's complement sum of all n bit words (blocks)

Topic : Sum Complement

Block Size = n

Receiver

Block 1

Block 2

Block (K – 1)

Block K Checksum

n - bit 1's compl. sum

1's complement

Result

Sender

Block 1

Block 2

DATA

Block (K – 1)

Block K 0 0 0 .... 0 0 0

n - bit 1's compl. sum

1's complement

Checksum

Topic : Sum Complement

Result : [Computed at receiver]

if **Result** == **ZERO** :

then Receiver concluded "**No any error detected**"

else

Receiver concluded "**Error detected**"

Example 2 :-

Suppose block size = 5
No. of blocks = 6

3 → 11
5 → 101
10 → 2
4 → 100

$$\underbrace{10}\ \underbrace{11011}$$

+

1

$$\begin{array}{r} 11011 \\ +\ \ 10 \\ \hline 11101 \end{array}$$

$$\begin{array}{r} 11111 \\ -\ 11101 \\ \hline 00010 \end{array}$$

Sender

1 0 1 0 1 0 1

1 0 1 1 0
0 1 1 0 1
1 1 0 1 1
1 0 0 1 1
0 1 0 1 0
0 0 0 0 0

---

10 11011

1's Comp. Sum = 11101

↓ 1's comp.

Checksum = $\boxed{00010}$

Example 2 :-

Suppose block size = 5
No. of blocks = 6

Sender

1 0 1 1 0
0 1 1 0 1
1 1 0 1 1
1 0 0 1 1
0 1 0 1 0
0 0 0 0 0

---

1 0 1 1 0 1 1

1's Comp. Sum = 1 1 1 0 1

1's complement

Checksum = 0 0 0 1 0

Example 2 :-

Suppose block size = 5
No. of blocks = 6

Receiver

10 10 11 1

1 0 1 1 0
0 1 1 0 1
1 1 0 1 1
1 0 0 1 1
0 1 0 1 0
0 0 0 1 0

10 11101

1's Comp. Sum = 11111

↓ 1's comp.

Result = 00000 ⟹ No any error detected;

10 11101
0 0
+

+11101
10

11111

Sender

1 0 1 1 0
0 1 1 0 1
1 1 0 1 1
1 0 0 1 1
0 1 0 1 0
0 0 0 0 0

1 0 1 1 0 1 1

1 1 1 0 1

1's complement

0 0 0 1 0

Example 2 :-

Suppose block size = 5
No. of blocks = 6

	Receiver	Sender
	1 0 1 1 0	1 0 1 1 0
	0 1 1 0 1	0 1 1 0 1
	1 1 0 1 1	1 1 0 1 1
	1 0 0 1 1	1 0 0 1 1
	0 1 0 1 0	0 1 0 1 0
	0 0 0 1 0	0 0 0 0 0
	1 0 1 1 1 0 1	1 0 1 1 0 1 1
1's Comp. Sum =	1 1 1 1 1	1 1 1 0 1
	1's complement	1's complement
Result =	0 0 0 0 0	0 0 0 1 0

Topic : Checksum → Redundent bits

→ Sender generate n-bit Checksum from all data blocks and then send data along with checksum

→ Receiver check the "received data" (including checksum) is balanced or not

Topic : Checksum

→ While computing the checksum,
the value of the checksum field should be initialized with zero

→ While transmission,
checksum field should be updated with computed checksum

Example 3 :-

Suppose block size = 5
No. of blocks = 6

$4 \rightarrow 100$
$7 \rightarrow 111$

11 01110

+

111

01110

+ 11

10001

Sender

11 10 11 10

1 1 1 1 0
0 1 1 1 1
1 1 0 1 1
1 1 0 1 1
0 1 0 1 1
0 0 0 0 0

11 01110

1's Comp. Sum = 10001

1's comp.

Checksum = $\boxed{01110}$

Example 3 :-

Suppose block size = 5
No. of blocks = 6

Sender

	1 1 1 1 0
	0 1 1 1 1
	1 1 0 1 1
	1 1 0 1 1
	0 1 0 1 1
	0 0 0 0 0
	1 1 0 1 1 1 0
1's Comp. Sum =	1 0 0 0 1
	1's complement
Checksum =	0 1 1 1 0

Example 3 :-

Suppose block size = 5

No. of blocks = 6

Receiver	Sender
1 1 1 1 0	1 1 1 1 0
0 1 1 1 1	0 1 1 1 1
1 1 0 1 1	1 1 0 1 1
1 1 0 1 1	1 1 0 1 1
0 1 0 1 1	0 1 0 1 1
0 1 1 1 0	0 0 0 0 0
11 11100	1 1 0 1 1 1 0
1's Comp. Sum = 11111	1 0 0 0 1
Result = 00000 ⇒ Accept the data	1's complement: 0 1 1 1 0

(4)(11)(4)(10)

$8 \rightarrow 1000$

$9 \rightarrow 1001$

$\underbrace{11}_{0}\ \underbrace{11100}_{0}$ +

$$\begin{array}{r} 11100 \\ +\quad 11 \\ \hline 11111 \end{array}$$

↓ 1's comp.

Example 3 :-

Suppose block size = 5
No. of blocks = 6

	Receiver	Sender
	1 1 1 1 0	1 1 1 1 0
	0 1 1 1 1	0 1 1 1 1
	1 1 0 1 1	1 1 0 1 1
	1 1 0 1 1	1 1 0 1 1
	0 1 0 1 1	0 1 0 1 1
	0 1 1 1 0	0 0 0 0 0
	1 1 1 1 1 0 0	1 1 0 1 1 1 0
1's Comp. Sum =	1 1 1 1 1	1 0 0 0 1
	↓ 1's complement	↓ 1's complement
Result =	0 0 0 0 0	0 1 1 1 0

Recv.

(2)

$$\begin{array}{r} 12 \\ 58 \\ 74 \\ 96 \\ 57 \\ \hline 297 \\ \hline 99 \end{array}$$

$99 \xrightarrow{\text{comp.}} \boxed{00}$ Result.

$$297$$

$$\begin{array}{r} 97 \\ +\ 2 \\ \hline 99 \end{array}$$

$$\begin{array}{r} 99 \\ -99 \\ \hline 00 \end{array}$$

$$240$$

$$\begin{array}{r} 40 \\ +\ 2 \\ \hline 42 \end{array}$$

$$\begin{array}{r} 99 \\ -42 \\ \hline 57 \end{array}$$

Sender

(2)

$$\begin{array}{r} 12 \\ 58 \\ 74 \\ 96 \\ 00 \\ \hline 240 \\ \hline 42 \end{array}$$

$42 \xrightarrow{\text{comp.}} \boxed{57} \Rightarrow$ Checksum

Example 4 :-

String 1 = 1 0 0 1 0 1 0 0

String 2 = 1 1 0 0 1 1 0 1

Calculate checksum ?

Example 4 :-

String 1 = 1 0 0 1 0 1 0 0

String 2 = 1 1 0 0 1 1 0 1

Calculate checksum ?

Solution :-

$$
\begin{array}{r}
\scriptstyle 1\,1\,1 \\
1\,0\,0\,1\,0\,1\,0\,0 \\
1\,1\,0\,0\,1\,1\,0\,1 \\
\hline
1\ \ 0\,1\,1\,0\,0\,0\,0\,1
\end{array}
$$

$$
\begin{array}{r}
\scriptstyle 1 \\
0\,1\,1\,0\,0\,0\,0\,1 \\
+1 \\
\hline
0\,1\,1\,0\,0\,0\,1\,0
\end{array}
$$

$$\downarrow \text{1's comp.}$$

$$\boxed{1\,0\,0\,1\,1\,1\,0\,1} \Longrightarrow \text{Checksum}$$

Example 5 :-

Suppose block size = 5
No. of blocks = 6

Sender

1 0 0 0 0
0 1 0 0 0
0 0 1 0 0
0 0 0 1 0
0 0 0 0 1
0 0 0 0 0

1 1 1 1 1

1's Comp. Sum = 1 1 1 1 1

↓ 1's comp.

Checksum = 0 0 0 0 0

Example 5 :-

Suppose block size = 5
No. of blocks = 6

Sender

$$
\begin{array}{r}
1\,0\,0\,0\,0 \\
0\,1\,0\,0\,0 \\
0\,0\,1\,0\,0 \\
0\,0\,0\,1\,0 \\
0\,0\,0\,0\,1 \\
0\,0\,0\,0\,0 \\
\hline
1\,1\,1\,1\,1 \\
\hline
\end{array}
$$

1's Comp. Sum = 1 1 1 1 1

1's complement

Checksum = 0 0 0 0 0

Example 6 :-

Suppose block size = 5
No. of blocks = 6

Sender

0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0

---

0 0 0 0 0

1's Comp. Sum = 0 0 0 0 0

1's complement

Checksum = $\boxed{1\ 1\ 1\ 1\ 1}$

Topic : Checksum

→ Checksum detect "**all single bit error**"

→ In case of burst error,
checksum able to detect "**all odd number of errors**"

→ Checksum can be "all zero bits",
but checksum can never be "all one bits"

Topic : Checksum

⇒ Not belongs to linear code

$d_1 d_2$
$d_3 d_4$
$P_1\ P_2$
↓ 1's comp.
$C_1\ C_2$

Sum Complement (Checksum) and 4 data bits :

Data	→	Codeword	Data	→	Codeword
$d_1d_2d_3d_4$	→	$d_1d_2d_3d_4C_1C_2$	$d_1d_2d_3d_4$	→	$d_1d_2d_3d_4C_1C_2$
0 0 0 0	→	0 0 0 0 1 1	1 0 0 0	→	1 0 0 0 0 1
0 0 0 1	→	0 0 0 1 1 0	1 0 0 1	→	1 0 0 1 0 0
0 0 1 0	→	0 0 1 0 0 1	1 0 1 0	→	1 0 1 0 1 0
0 0 1 1	→	0 0 1 1 0 0	1 0 1 1	→	1 0 1 1 0 1
0 1 0 0	→	0 1 0 0 1 0	1 1 0 0	→	1 1 0 0 0 0
0 1 0 1	→	0 1 0 1 0 1	1 1 0 1	→	1 1 0 1 1 0
0 1 1 0	→	0 1 1 0 0 0	1 1 1 0	→	1 1 1 0 0 1
0 1 1 1	→	0 1 1 1 1 0	1 1 1 1	→	1 1 1 1 0 0

Topic : Hamming Weight

→ The weight of a codeword is the number of nonzero elements

Hamming weight of a binary string = Number of one's in that string

Example :-

Hamming Weight["10011001"] = $\boxed{4}$

Topic : Hamming Distance

→ Metric for comparing two binary strings of equal length

Hamming distance between two binary strings of equal length

= Number of positions at which the corresponding bits are different

Topic : Hamming Distance

→ Let suppose A and B are two binary strings of equal length

d(A,B)	=	Hamming Distance between A and B
	=	Hamming Weight of [A bit-wise XOR B]

Topic : Hamming Distance

Example :- A = 1011101 and B = 0110110

$$\begin{array}{lrl} & A = & 1011101 \\ \text{bit-wise XOR} & B = & \underline{0110110} \\ & \underbrace{\text{Result}}_{0} = & 1101011 \end{array}$$

d(A,B) = Hamming Weight[Result]

$= 5$

$P = 10110101$

$Q = 01110111$

$d(P,Q) = ? = 3$

Topic : Hamming Distance

$\left({}^4C_2 \text{ Pair} \Rightarrow 6\right)$

Let suppose set of valid codewords :

$$\left\{\begin{array}{lll} \text{Codeword } C_1 & : & 00000\ 00000 \\ \text{Codeword } C_2 & : & 00000\ 11111 \\ \text{Codeword } C_3 & : & 11111\ 00000 \\ \text{Codeword } C_4 & : & 11111\ 11111 \end{array}\right.$$

valid code $\xrightarrow[\text{4 bit error}]{\text{upto}}$ Invalid code

valid code $\xrightarrow[\text{error}]{\text{5 bit}}$ valid code

valid code $\xrightarrow[\text{error}]{\text{5 bit}}$ Invalid code

minimum Hamming Distance =

Minimum[d(C_1, C_2), d(C_1, C_3), d(C_1, C_4), d(C_2, C_3), d(C_2, C_4), d(C_3, C_4)]

$= \text{Min}[5, 5, 10, 10, 5, 5]$

$= 5$

Topic : Hamming Distance

CASE I : $\underbrace{\text{No any error}}$

Transmitted Code = $\boxed{00000\ 00000}$ $\underbrace{\text{(Valid Codeword)}}$

Received Code = $\boxed{00000\ 00000}$ $\underbrace{\text{(Valid Codeword)}}$

Error Detected : "No" any error detect.

Corrected Codeword :

Codeword C_1 : 00000 00000

Codeword C_2 : 00000 11111

Codeword C_3 : 11111 00000

Codeword C_4 : 11111 11111

Topic : Hamming Distance

CASE II : One-bit error → Always detected and correct.

Transmitted Code = 00000 00000 (Valid Codeword)

Received Code = $\boxed{00000\ 00001}$ (Invalid Codeword)

Error Detected : "Yes"

Corrected Codeword : 00000 00000 (Nearest Valid Codeword)

Codeword C_1 : 00000 00000

Codeword C_2 : 00000 11111

Codeword C_3 : 11111 00000

Codeword C_4 : 11111 11111

$d(C_1, RC) = 1$

$d(C_2, RC) = 4$

$d(C_3, RC) = 6$

$d(C_4, RC) = 9$

Topic : Hamming Distance

CASE III : Two-bit error $\rightarrow$ Always detected and corrected.

Transmitted Code = 00000 00000 (Valid Codeword)

Received Code = $\boxed{00000\ 00011}$ (Invalid Codeword)

Error Detected : "Yes"

Corrected Codeword : 00000 00000 (Nearest Valid Codeword)

Codeword C_1 : 00000 00000 $\Rightarrow$ 2

Codeword C_2 : 00000 11111 $\Rightarrow$ 3

Codeword C_3 : 11111 00000 $\Rightarrow$ 7

Codeword C_4 : 11111 11111 $\Rightarrow$ 8

$C_1 \longleftrightarrow C_2$

2.5

Topic : Hamming Distance

CASE IV : Three-bit error → Always detected / Not corrected.

Transmitted Code = 00000 00000 (Valid Codeword)

Received Code = $\boxed{00000\ 00111}$ (Invalid Codeword)

Error Detected : "Yes"

Corrected Codeword : 00000 11111 (Nearest Valid Codeword)

Codeword C_1 : 00000 00000 ⇒ 3

Codeword C_2 : 00000 11111 ⇒ 2

Codeword C_3 : 11111 00000 ⇒ 8

Codeword C_4 : 11111 11111 ⇒ 7

Topic : Hamming Distance

CASE V : Four-bit error → Always detected
→ Not correction

Transmitted Code = 00000 00000 (Valid Codeword)

Received Code = $\boxed{00000\ 01111}$ (Invalid Codeword)

Error Detected : "Yes"

Corrected Codeword : 00000 11111 (Nearest Valid Codeword)

Codeword C_1 : 00000 00000

Codeword C_2 : 00000 11111

Codeword C_3 : 11111 00000

Codeword C_4 : 11111 11111

Topic : Hamming Distance

CASE VI : Five-bit error → May be detected.

Transmitted Code = 00000 00000 (Valid Codeword)

Received Code = $\boxed{00000\ 11111}$ (Valid Codeword)

Error Detected : "No"

Corrected Codeword :

Codeword C_1 : 00000 00000

Codeword C_2 : 00000 11111

Codeword C_3 : 11111 00000

Codeword C_4 : 11111 11111

Topic : Hamming Distance

CASE VII : Five-bit error → May be detected

Transmitted Code = 00000 00000 (Valid Codeword)

Received Code = $\boxed{00001\ 01111}$ (Invalid code)

Error Detected : "Yes"

Corrected Codeword : 00000 11111 (Nearest Valid Codeword)

Codeword C_1 : 00000 00000

Codeword C_2 : 00000 11111

Codeword C_3 : 11111 00000

Codeword C_4 : 11111 11111

Topic : Hamming Distance

if (minimum) Hamming Distance is D

then receiver can detect upto (D-1) bits error

and receiver can correct upto Floor[(D-1)/2] bits error

$$\left\lfloor \frac{(D-1)}{2} \right\rfloor$$

Topic : Hamming Distance

To detect up to x - bits error

-> minimum Hamming Distance should be (x+1)

To correct up to y - bits error

-> minimum Hamming Distance should be (2y+1)

#Q. An error correcting code has the following code words:

[00000000, 00001111, 01010101, 10101010, 11110000]

What is the maximum number of bit errors that can be corrected ?

[GATE 2007]

IIT-K

H.W

(A) 0

(B) 1

(C) 2

(D) 3

#Q. Consider a binary code that consists only four valid codewords as given below.

00000, 01011, 10101, 11110

Let minimum Hamming distance of code be p and maximum number of erroneous bits that can be corrected by the code be q. The value of p and q are:

[GATE 2017]

IIT-R

(A) p = 3 and q = 1

(B) p = 3 and q = 2

(C) p = 4 and q = 1

(D) p = 4 and q = 2

2 mins Summary

Topic Checksum

Topic Hamming Distance

Bit Error Prob.

THANK - YOU

@ABHISHEKSIRCS_PW