28-12-22

Number systems

i) Decimal number system

In decimal number system we use numbers 0, 1, 2, ..., 9. And decimal number system are represented with the radix 10. These number systems are widely used in realtime applications like currency, marks etc.

* Conversion of Decimals number system to other number system:

i) Convertion of decimal to binary.

Example:

i. Convert $(47)_{10}$ the following decimal number to binary

Sol. $(47)_{10} = (\overset{MSB}{1}0111\overset{LSB}{1})_2$

2	47
2	23 - 1 (LSB)
2	11 - 1
2	5 - 1
2	2 - 1
	(MSB) 1 - 0

1 0 1 1 1 1
16 8 4 2 1

ii. Convert the decimal number $(223)_{10}$ to binary

$(223)_{10} = (11011111)_2$

$128 + 64 + 0 + 16 + 8 + 4 + 2 + 1 = 223$

2	223
2	111 - 1
2	55 - 1 (LSB)
2	27 - 1
2	13 - 1
2	6 - 1
2	3 - 0
	1 - 1
	(MSB)

iii) Convert the decimal number $(128)_{10}$ to binary

$(128)_{10} = (10000000)_2$

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

2	128 - 0
2	64 - 0
2	32 - 0
2	16 - 0
2	8 - 0
2	4 - 0
2	2 - 0
	1 - 0

Multiply by 2

Convert 130.56 decimal number to binary number

decimal
divide by 2

2	130
2	65 - 0
2	32 - 1
2	16 - 0
2	8 - 0
2	4 - 0
2	2 - 0
(MSB)	1 - 0

(LSB)

$0.56 \times 2 = 1.12$ 1

$0.12 \times 2 = 0.24$ 0

$0.24 \times 2 = 0.48$ 0

$0.48 \times 2 = 0.96$ 0

$0.96 \times 2 = 1.96$ 1

Take any 3 digits 100

2) 130 (65
-12↓
10
-10
0

29/12/2022

* Convert $(223.86)_{10} = (\ )_2$ decimal number to binary number.

2	223
2	111 - 1
2	55 - 1
2	27 - 1
2	13 - 1
2	6 - 1
2	3 - 0
(MSB)	1 - 1

(LSB)

$0.86 \times 2 = 1.72$ 1

$0.72 \times 2 = 1.44$ 1

$0.44 \times 2 = 0.88$ 0

$0.88 \times 2 = 1.66$ 1

11011111.1101

* Convert $(1024)_{10} = (\ )_2$. Convert decimal number to binary.

2	1024
2	512 - 0
2	256 - 0
2	128 - 0
2	64 - 0
2	32 - 0
2	16 - 0
2	8 - 0

2	4 - 0
2	2 - 0
	1 - 0

$(1024)_{10} = (10000000000)_2$

* convert $(12.74)_{10}$ decimal number to binary no%.-

$$\begin{array}{r|l} 2 & 12 \\ \hline 2 & 6-0 \\ \hline 2 & 3-0 \\ \hline & 12-01 \end{array}$$

$0.74 \times 2 = 1.48$ 1

$0.48 \times 2 = 0.96$ 0

$0.96 \times 2 = 1.92$ 1

$0.92 \times 2 = 1.84$ 1

1100.1011

$(12)_{10} = (1100)_2$

$(12.74)_{10} = (1100.1011)_2$

2) Conversion of Decimal to Octal

* convert the following decimal number to octal

$(1024)_{10} = (\ )_8$

$$\begin{array}{r|l} 8 & 1024 \\ \hline 8 & 128-0 \\ \hline 8 & 16-0 \\ \hline & 2-0 \end{array}$$

$(2000)_8$

* convert the following decimal number to octal $(10475)_{10} = (\ )_8$

$$\begin{array}{r|l} 8 & 10475 \\ \hline 8 & 1309-3 \\ \hline 8 & 163-5 \\ \hline 8 & 20-3 \\ \hline & 2-4 \end{array}$$

$(10475)_{10} = (24353)_8$

* Convert the following decimal number to octal

$(136.23)_{10} = (\ )_8$

$$\begin{array}{r|l} 8 & 136 \\ \hline 8 & 17-0 \\ \hline & 2-1 \end{array}$$

$0.23 \times 8 = 1.84$ 1

$0.84 \times 8 = 6.72$ 6

$0.72 \times 8 = 5.76$ 5

Scanned by TapScanner

3) Conversion of decimal to hexa decimal

Note:- In this conversion if the remainder is 10. Consider it as A, 11-B, 12-C, 13-D, 14-E, 15-F, 16-zero.

Q. Convert $(11377)_{10} = ( )_{16}$ decimal to Hexadecimal

Sol. 16 | 11377
16 | 711 - 1
16 | 44 - 7
| 2 - C

$0.0625 \times 16 = 1$
$0.4375 \times 16 = 7$

$(11377)_{10} = (2C71)_{16}$

Q. Convert $(336.85)_{10} = ( )_{16}$ decimal to Hexa decimal.

16 | 336
16 | 21 - 0
| 1 - 5

$0.85 \times 16 = 13.6$ (D)
$0.6 \times 16 = 9.6$ → 9
$0.6 \times 16 = 9.6$ → 9

Q. Convert $(8473.23)_{10} = ( )_{16}$ decimal to hexa decimal.

16 | 84723
16 | 529 - 9
16 | 33 - 1
2 - 1

$0.23 \times 16 = 3.68$ → 3
$0.68 \times 16 = 10.88$ → A
$0.88 \times 16 = 14.08$ → E

$(84723)_{10} = (2119)_8$ $(8473.23)_{10} = (2119.3AE)_{16}$

* Binary Number System

Binary number system is denoted by radix (2).
The number system used in Binary is 0 & 1.
So computers use Binary system.

Code

	4	2	1
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	01	0
7	1	1	1

	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

1) Binary to decimal

$(10111)_2 = (23)_{10}$

$1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 +$

$1 \times 2^1 + 1 \times 2^0$

$= 16 + 0 + 4 + 2 + 1$

$= 23$

Scanned by TapScanner

30-12-2022

Q- $(11011)_2 = (27)_{10}$

$16 + 8 + 2 + 1 = 27$

Q- $(1111\,0111 \cdot 0110)_2$

$1\times2^7 + 1\times2^6 + 1\times2^5 + 1\times2^4 + 1\times2^3 + 1\times2^2 + 1\times2^1 + 1\times2^0$

$(0\times2^{-1} + 1\times2^{-2} + 1\times2^{-3} + 0\times2^{-4})$

Q. 1111 0111

$128 + 64 + 32 + 16 + 4 + 2 + 1 = 247$

$0 \cdot 0110 = 0\times2^{-1} + 1\times2^{-2} + 1\times2^{-3} + 0\times2^{-4}$

$= 0 + \frac{1}{4} + \frac{1}{8} + 0 = \frac{8+4}{32} = \frac{12}{32} = 0 \cdot 375$

$(1111 0111 \cdot 0110)_2 = (247 \cdot 375)_{10}$

Q- Convert Binary to decimal $(1011 \cdot 01101)_2 - ()_{10}$

1011

$8 + 2 + 1 = 11$

$0 \cdot 01101 = 0\times2^{-1} + 1\times2^{-2} + 1\times2^{-3} + 0\times2^{-4} + 1\times2^{-5}$

$= 0 + \frac{1}{4} + \frac{1}{8} + 0 + \frac{1}{32} = \frac{13}{32} = 0 \cdot 40625$

$(1011 \cdot 01101)_2 = (11 \cdot 040625)_{10}$

2) Binary to Octal

1. $010110 = (26)_8$
2. $1011011\,0110 = (2666)_8$
3. $0101\,0111\,011\,011\,011 = (53333)_8$
4. $01010\,1101 \cdot 101\,100 = (255 \cdot 54)_8$
5. $001 \cdot 1111001 = (1 \cdot 744)_8$

3) Binary to Hexadecimal

1- 10110 = $(16)_{16}$

2- 101101101011011011011011 = $(B6B6DB)_{16}$

3- 00101101·011011000100 = $(2D.6C4)_{16}$

* Octal Number system

Octal number system is denoted with radix (8) & numbers from (0-7). These are used in microprocessors.

1) Octal to Binary

1) $(23)_8 = (\ )_2$

$\overset{2}{010}\ \overset{3}{011} \Rightarrow (010011)_2$

2) $(5655)_8 = (\ )_2$

(5) 4 2 1	(6) 4 2 1	(5) 4 2 1	(5) 4 2 1
1 0 1	1 1 0	1 0 1	1 0 1

$= (101110101101)_2$

3) $(192.38)_8 = (\ )_2$

The number conversion is not possible as it exists from (0-7)

4) 162·36

(1) 4 2 1	(6) 4 2 1	(2) 4 2 1	(3) 4 2 1
0 0 1	1 1 0	0 1 0	0 1 1

(6) 4 2 1
1 1 0

$(162.36)_8 = (00111001 \cdot 001111)_2$

Scanned by TapScanner

31-12-2022

* Octal to Decimal:-

1) $(236)_8 = (\ )_{10}$

$= 2 \times 8^2 + 3 \times 8^1 + 6 \times 8^0$

$= 128 + 24 + 6 = (158)_{10}$

2) $(552.38)_8$

$= 5 \times 8^2 + 5 \times 8^1 + 2 \times 8^0 . 3 \times 8^{-1} + 6 \times 8^{-2}$

$= 362.46875$

* Octal to Hexa decimal:-

1) $(23)_8 = (\ )_{16}$

0001 | 0011 → $(13)_{16}$
1 | 3

2) $(558)_8 = (\ )_{16}$

Not possible

3) $(557)_8 = (\ )_{16}$

5 5 7
0001 | 0110 | 1111 ← $(16F)_{16}$
1 6 F

* $(123.456)_8 = (\ )_{16}$

1 2 3
0 | 0101 | 0011 . 1001 | 0111 | 0

=

Hexa decimal numbers system:-

Hexa decimal numbers are represented by $(\ )_{16}$ and numbers from 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. These number system most widely used in microprocessors than octal system.

$\Rightarrow (2C71)_{16} = (\quad)_{10}$

$\begin{matrix} 4 & 2 & 1 \\ 0 & 1 & 0 \end{matrix}$

$= 2\times16^3 + 12\times16^2 + 7\times16^1 + 1\times16^0$

$= 8192 + 3072 + 112 + 1$

$= 11377$

$\Rightarrow (BEEE)_{16} = (\quad)_{10}$

$= 11\times16^3 + 14\times16^2 + 14\times16^1 + 14\times16^0$

$= (48878)_{10}$

$\Rightarrow (BA\cdot EF)_{16}$

$= 11\times16^1 + 10\times16^0 \cdot 14\times16^{-1} + 15\times16^{-2}$

$= 176 + 10 \cdot \frac{14}{16} + \frac{15}{256}$

$= 186 \cdot \frac{224+15}{256} = 186\cdot\frac{139}{256} = 186\cdot93359375$

$(BA\cdot EF)_{16} = (186\cdot9335937 5)_{10}$

2) Hexadecimal to Binary

Q. $(2C71)_{16}$

(2)				(C)				(7)				(1)			
8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1
0	0	1	0	1	1	0	0	0	1	1	1	0	0	0	1

$(2C71)_{16} = (0010110001110001)_{10}$

Q. $(BEEE)_{16}$

(B)=11				(E)=14				(E)=14				(E)=14			
8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1
1	0	1	1	1	1	1	0	1	1	1	0	1	1	1	0

$(1011111011101110)_{10}$

Q) $(BA \cdot EF)_{16}$

F = 15, E = 14, A = 10, B = 11

8 4 2 1 | 8 4 2 1 . 8 4 2 1 | 8 4 2 1
1 0 1 1 | 1 0 1 0 . 1 1 1 0 | 1 1 1 1

$(BA \cdot EF)_{16} = (10111010 \cdot 11101111)_{10}$

3) Hexadecimal to octal :-

Q. $(2C71b)_{16} = (\quad)_8$

Binary = 0|0 1 0|1 1 0|0 0 1|1 1 0|0 0 1 → (2616

2 6 1 6 1

Q- $(BEEE)_{16} = (\quad)_8$

1|0 1 1|1 1 1|0 1 1|1 0 1|1 1 0 = (137356)

1 3 7 3 5 6

Q- $(BA \cdot EF)_{16}$

0 1 0|1 1 1|0 1 0 · 1 1 1|0 1 1|1 1 = $(272 \cdot 736)_8$

2 7 2 7 3 6

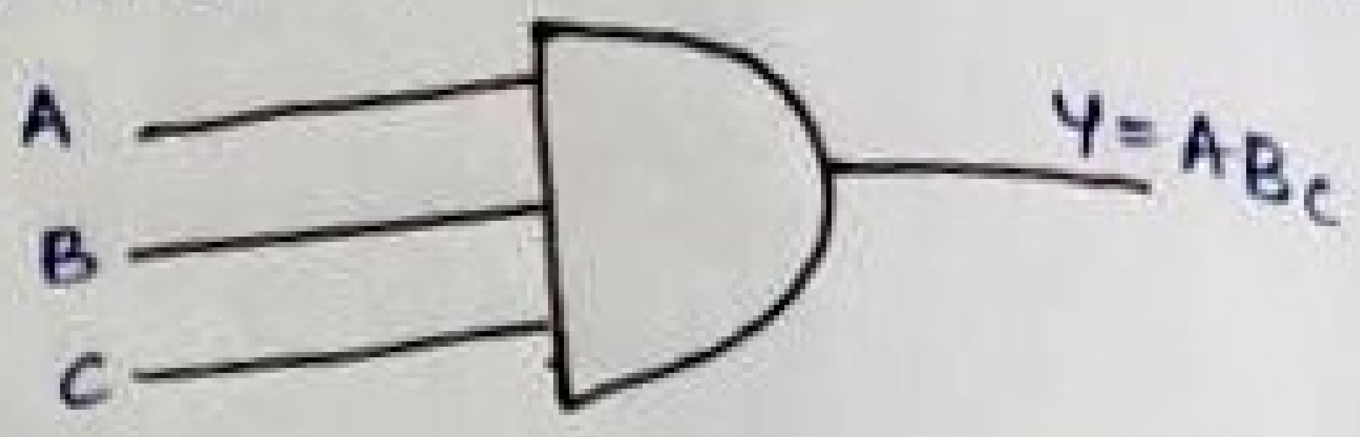
04/01/2022

LOGIC GATES

1) AND Gate

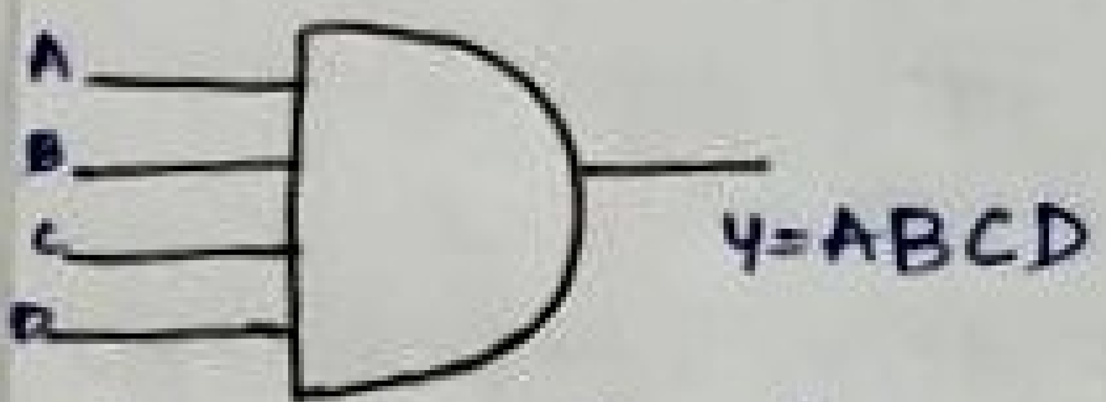
A	B	Y=AB
0	0	0
0	1	0
1	0	0
1	1	1

2) AND Gate with 3 inputs



A	B	C	Y=ABC
0	0	0	0
0	0	1	0
0	1	0	0
1	0	0	0
1	0	1	0
1	1	0	0
0	1	1	0
1	1	1	1

3) AND Gate with 4 inputs



8 A	4 B	2 C	1 D	Y=ABCD
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0

In AND gate we should do 'x'

0 x 0 = 0

0 x 1 = 0

1 x 0 = 0

1 x 1 = 1

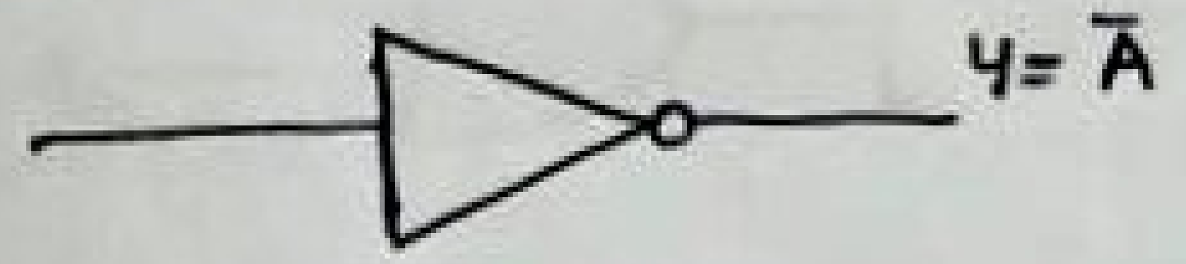
8	4	2	1	Y=A
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

1) OR Gate with 2 inputs

A	B	$Y = A+B$
0	0	0
0	1	1
1	0	1
1	1	1

2) NOT gate

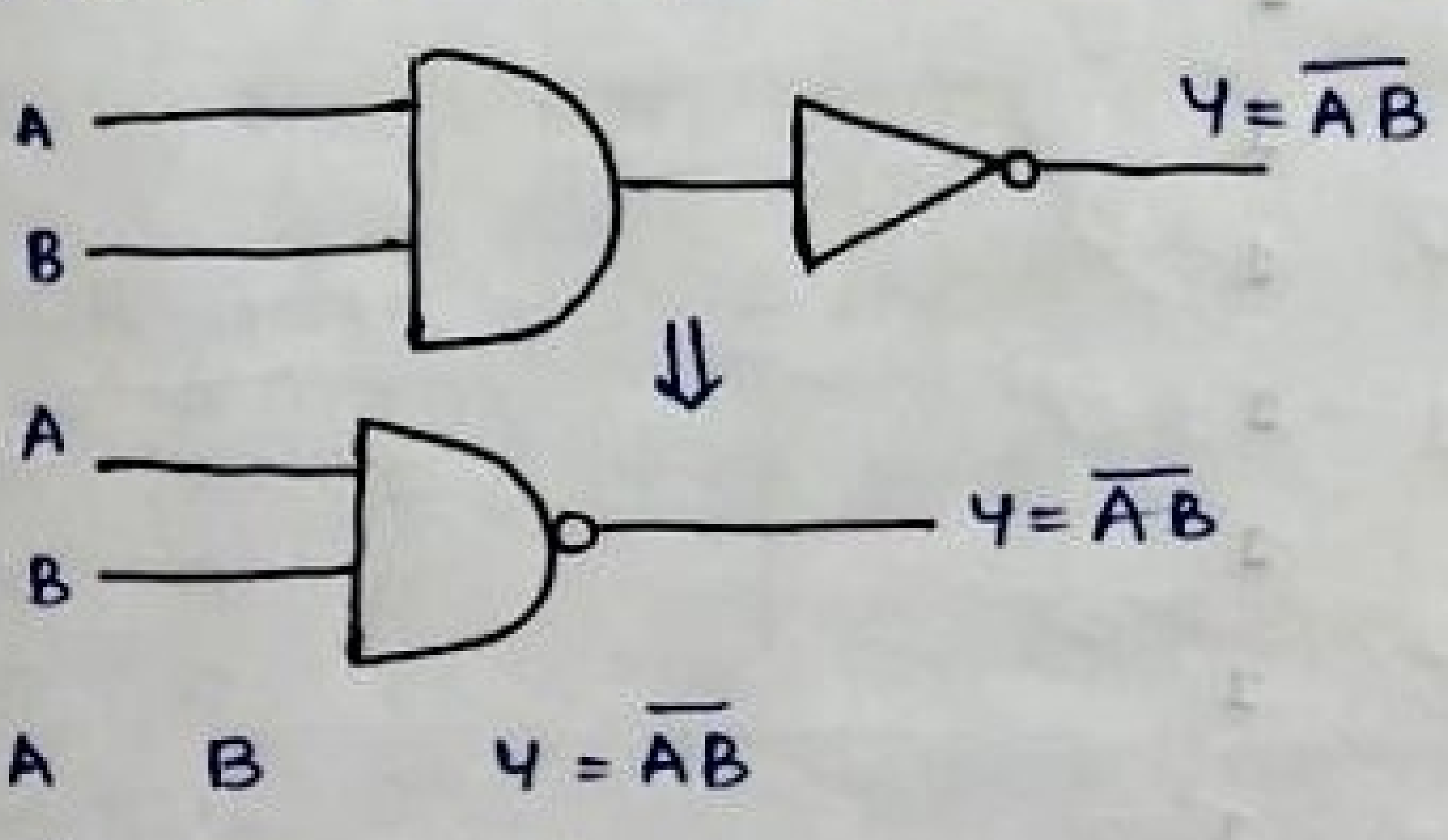
Not gate is used to compliment the given binary digit. (whether it is input or output).



A	$Y = \overline{A}$
0	1
1	0

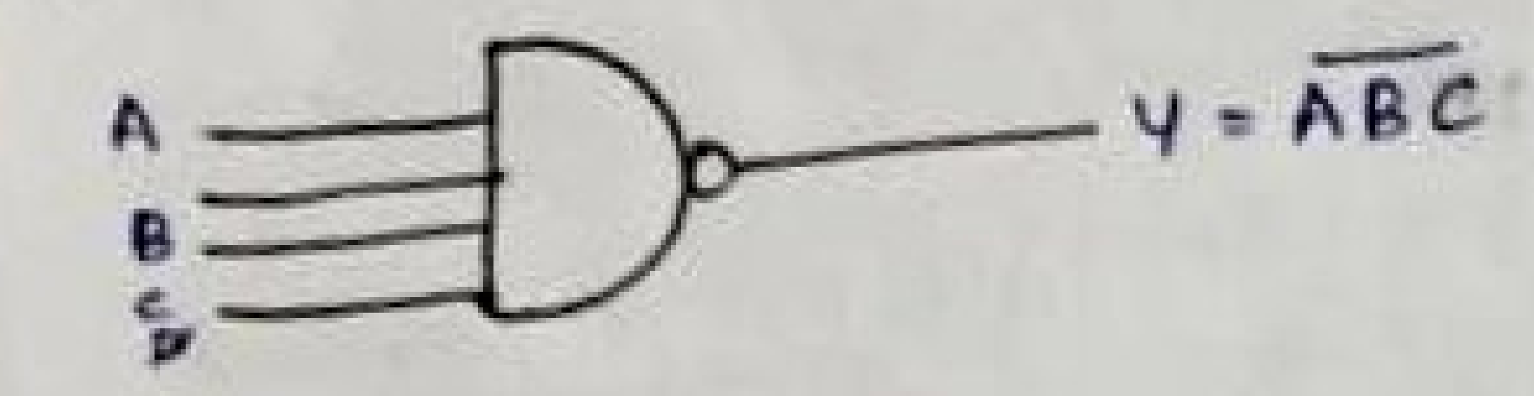
3) NAND gate :-

AND + NOT Gate is called NAND gate



A	B	$Y = \overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

NAND gate with 3 inputs :-

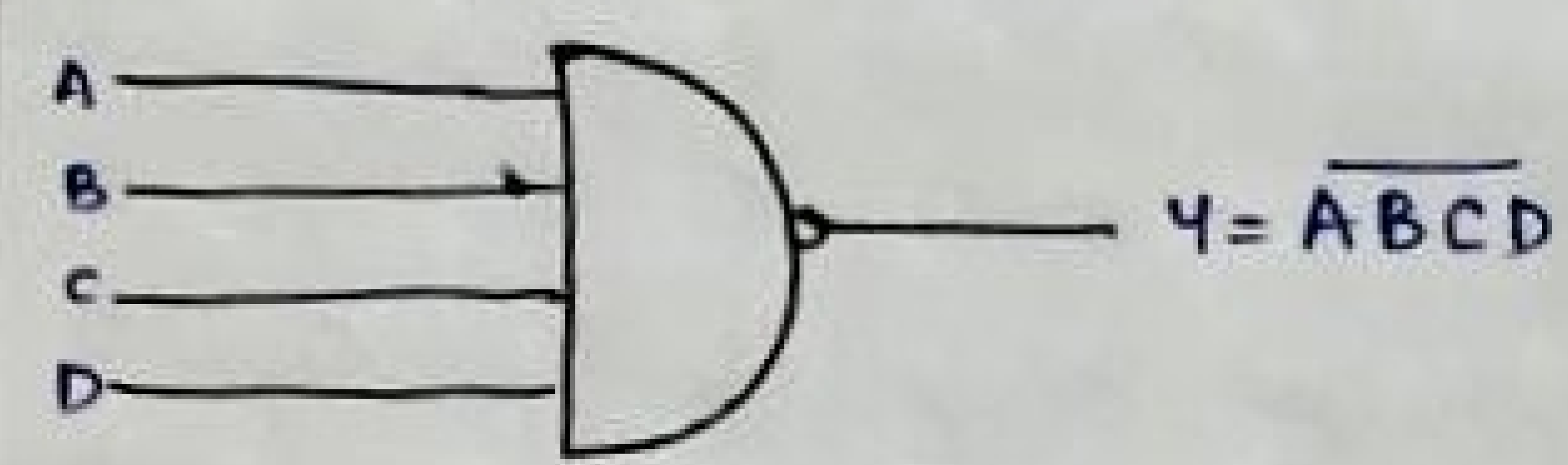


4 A	2 B	1 C	$Y = \overline{ABC}$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$\bar{0} = 1$

$\bar{1} = 0$

NAND gate with 4 inputs :-

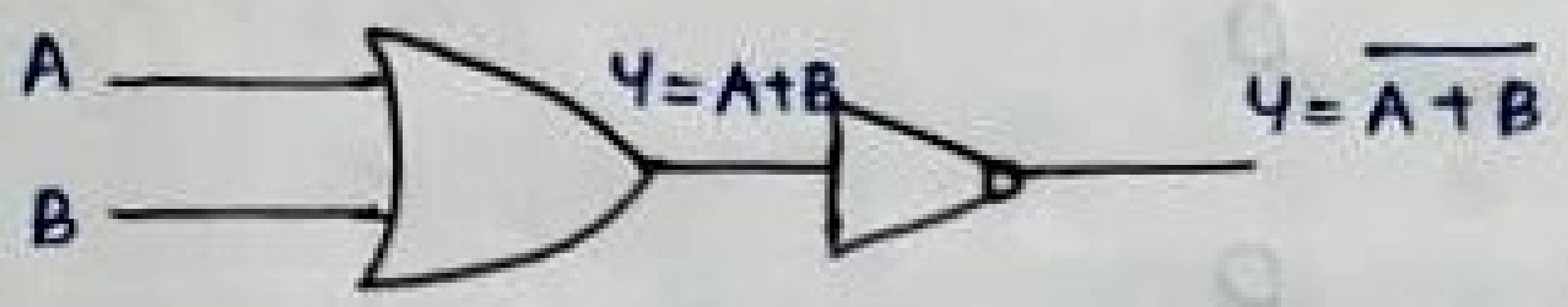


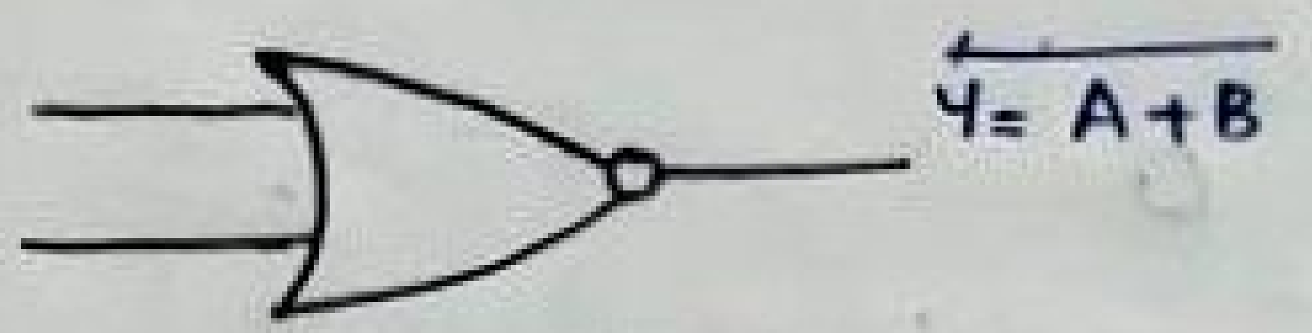
8 A	4 B	2 C	1 D	$Y = \overline{ABCD}$
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1

8	4	2	1	$Y = \overline{ABCD}$
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

1) NOR Gate

OR Gate + NOT Gate (with 2 inputs)





A	B	$Y = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

$0+0=\overline{0}=1$

$1+0=\overline{1}=0$

$1+1=\overline{1}=0$

NOR Gate with 3 inputs

4 A	2 B	1 C	$Y = \overline{A+B+C}$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0

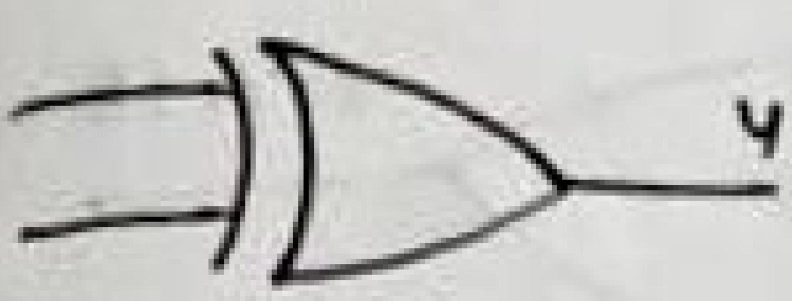
1	1	0	0
1	1	1	0

NOT Gate with 4 inputs.

$Y = A+B+C+D$

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

XOR Gate (with 2 inputs)



$Y = A + B = A\bar{B} + \bar{A}B$

$Y = A\bar{B} + \bar{A} \cdot B$

$A = 1, B = 1$

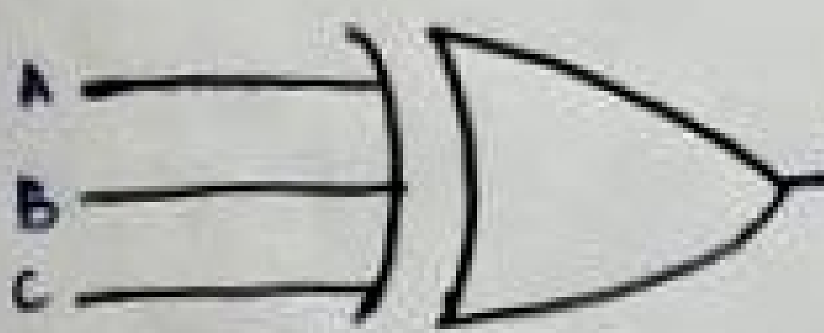
$Y = 1 \times 0 + 0 \times 1$

$Y = 0 + 0 = 0$

A	B	Y = A + B
0	0	0
0	1	1
1	0	1
1	1	0

05/01/2023

XOR Gate with 3 inputs

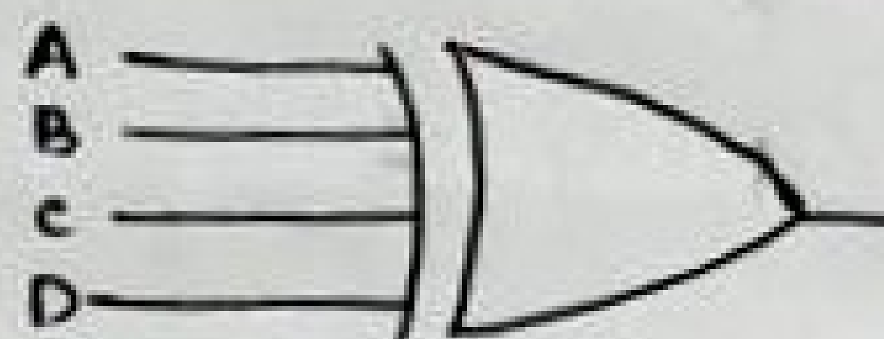


$Y = A \oplus B \oplus C$

4 2 1

A	B	C	$Y = A \oplus B \oplus C$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

XOR Gate with 4 inputs

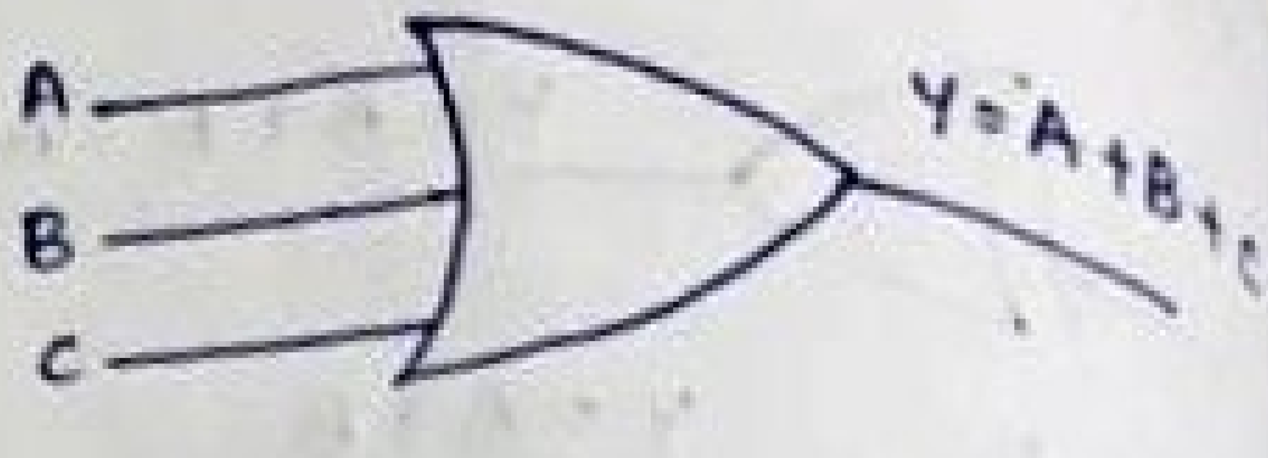


$Y = A \pm B \pm C \pm D$

8 4 2 1

A	B	C	D	$Y = A \oplus B \oplus C \oplus D$
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

OR Gate with 3 inputs



4 A	2 B	1 C	Y = A+B+C
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

OR Gate with 4 inputs

8 A	4 B	2 C	1 D	Y = A+B+C+D
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1

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

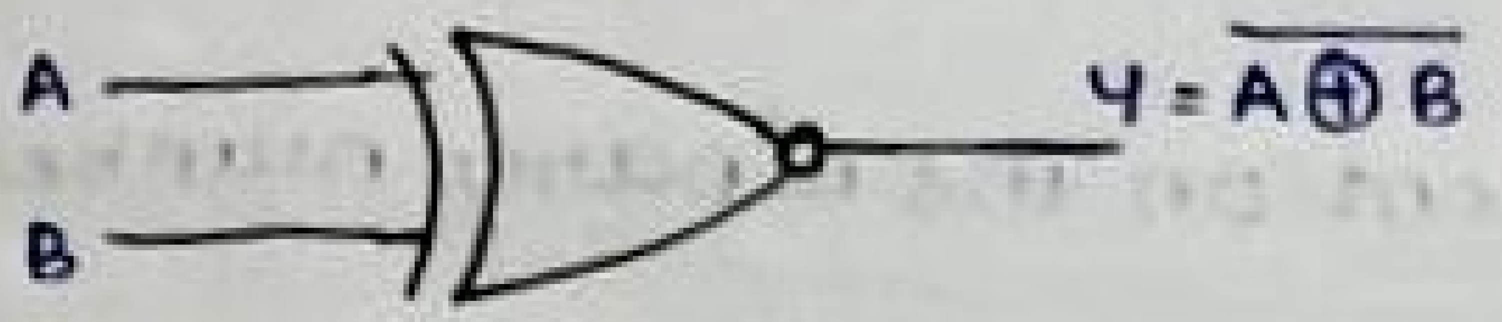
⇒ Binary subtraction using 1's complement & 2's complement

⇒ Binary Addition Bitwise

A	B	sum	carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

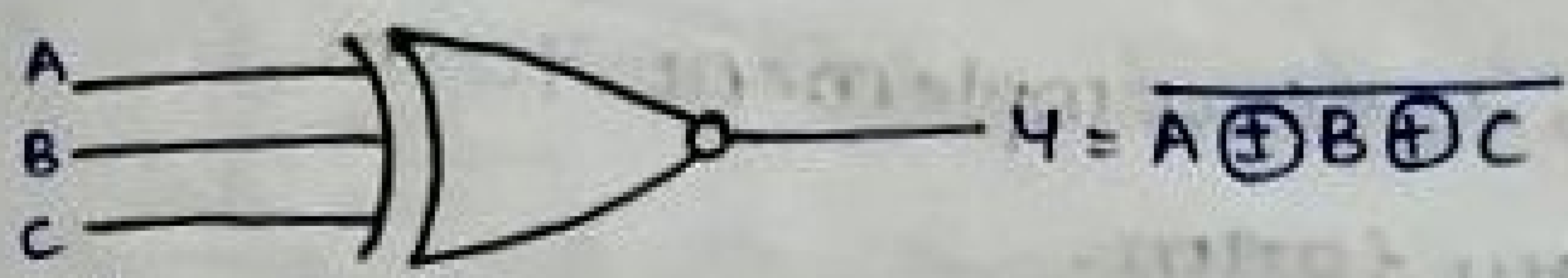
05-01-2023

* XNOR Gate with 2 inputs



A	B	$Y = \overline{A \oplus B}$
0	0	1
0	01	0
1	0	0
1	1	1

* XNOR Gate with 3 inputs



4 A	2 B	1 C	$Y = \overline{A \oplus B \oplus C}$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	01
1	1	1	0

1*1=0

6-1-23

Binary subtraction using 1's complement and 2's complement

1)

step-1

Convert given decimal numbers into binary numbers

step-2

Perform 1's compliment on the binary number to be subtracted.

step-3

If end carry is '1', the resultant number is positive and add end carry '1' to resultant binary number.

step-4

If end carry is '0', the resultant number is negative and in its '1''s complement form. Convert it into binary form.

Example:-

$(69)_{10} - (1)_{10} = (69)_{10} + (-1)_{10}$

	$(69)_{10}$ =	1	0	0	0	1	0	1
+								
	$(-1)_{10}$ =	1	1	1	1	1	1	0
		1						
	1	0	0	0	0	1	1	
							1	
	1	0	0	0	1	0	0	

Example

$(1)_{10} - (69)_{10}$

1 = 0 0 0 0 0 0 1

69 = 1 0 0 0 1 0 1

1 =	0	0	0	0	0	0	1
+ (69) =	0	1	1	1	0	1	0
	0	1	1	1	0	1	1

(1 0 0 0 1 0 0) ⇒ -68

Example

$(256)_{10} - (255)_{10}$

256	128	64	32	16	8	4	2	1	
1	0	0	0	0	0	0	0	0	= (256)
0	1	1	1	1	1	1	1	1	= (255)

	1	0	0	0	0	0	0	0	0
+	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0
									1
	0	0	0	0	0	0	0	0	1

1

$2^4 = 16$

$2^5 =$

$2 \times 2 \times 2 \times 2 \times 2$

$8 \times 4 = 32$

2^6

32 × 2 = 64

64 × 2 = 128

128 + 64 + 32 = 224

224 + 16 = 240

240 + 8 = 248

248 + 7 = 255

128 × 2 = 256

Example

$(255)_{10} - (256)_{10}$

0 1 1 1 1 1 1 1 1 $= (255)_{10}$

1 0 0 0 0 0 0 0 0 $= (256)_{10}$

0 1 1 1 1 1 1 1 1 (255)

$+$

0 1 1 1 1 1 1 1 1 (-256)

---

0 0 0 0 0 0 0 0 0

1

---

0 0 0 0 0 0 0 0 1

-1

Q- $(513)_{10} - (12)_{10}$

512 256 128 64 32 16 8 4 2 1

1 0 0 0 0 0 0 0 0 1 $= (513)_{10}$

0 0 0 0 0 0 1 1 0 0 $= (12)_{10}$

1 0 0 0 0 0 0 0 0 1 $= (513)_{10}$

1 1 1 1 1 1 0 0 1 1 $= (-12)_{10}$ $+$

1

---

0 1 1 1 1 1 0 0 01 0 0

1

---

0 1 1 1 1 1 0 1 0 1

8- $(12)_{10} - (513)_{10}$

$(12)_{10}$ = 0 0 0 0 0 0 1 1 0 0

$(-513)_{10}$ = 0 1 1 1 1 1 1 1 1 0

0 0 0 0 0 1 0 1 0

0 1~~0~~ 1 1 1 1 1 0 1

Binary's subtraction using 2's complement :-

⇒ 1's compliment + 1 → Binary '1'

Example :-

Decimal number '8' represent in 1's & 2's complement :

$(8)_{10}$ = 1 0 0 0

1's

$(-8)_{10}$ = 0 1 1 1

2's

$(-8)^2$ = 0 1 1 1

1 0 0 0

$(5)_{10}$ = 1 0 1

1's

$(-5)_{10}$ = 0 1 0

2's

$(-5)^2$ = 0 1 0

1

0 1 1

Procedure

Step 1 :

Convert given decimal numbers into binary number

Step2 :

Perform 2's complement on the binary number which is to be subtracted.

Step-3
Perform binary addition between two numbers.

Step-4
If end carry is '1' neglect the end carry and the resultant binary number is the final answer.

If end carry is '0' resultant number is negative in its 2's complement form. & convert it into binary form.

a) $(8)_{10} - (5)_{10}$

7-01-23

If end carry is '0'. Convert the resultant into its complement in its 1's & 2's complement

Binary subtraction using 2's complement

$(69)_{10} - (1)_{10}$

32	16	8	4	2	1		
1	0	0	0	1	0	1	$\Rightarrow (69)_{10}$
0	0	0	0	0	0	1	$\Rightarrow (1)_{10}$

- 1 1 1 1 1 1 0

\+ 1

1 1 1 1 1 1 1

⇒ 1 0 0 0 1 0 1

1 1 1 1 1 1 1

1 0 0 0 1 0 0

$(1)_{10} - (69)_{10}$

$(1)_{10}$ - 0 0 0 0 0 0 1

$(69)_{10}$ - 1 0 0 0 1 0 1

$(-69)_{10}$ - 0 1 1 1 0 1 0

$(-69)^2$ - 0 1 1 1 0 1 0

\+ 1

0 1 1 1 0 1 1

0 0 0 0 0 0 1

0 1 1 1 0 1 1

0 1 1 1 1 0 0

1 0 0 0 0 1 1

\+ 1

-1 0 0 0 1 0 0

= -68

Q- (256)10 - (255)10

1 0 0 0 0 0 0 0 0 = (256)10

0 1 1 1 1 1 1 1 1 = (255)10

1 0 0 0 0 0 0 0 0 = (-255)10

1

---

1 0 0 0 0 0 0 0 1

---

1 0 0 0 0 0 0 0 0

1 0 0 0 0 0 0 0 1

(1)

---

0 0 0 0 0 0 0 0 1 +1

Q- (255)10 - (256)10

0 1 1 1 1 1 1 1 1 = (255)10

1 0 0 0 0 0 0 0 0 = (256)10

(255)10 = 0 1 1 1 1 1 1 1 1

-(256)10 = 1 0 0 0 0 0 0 0 0

---

(0) 1 1 1 1 1 1 1 1 1

---

0 0 0 0 0 0 0 0 0 ← Inverse

1 → Add

---

0 0 0 0 0 0 0 0 1

(-1)

Scanned by TapScanner

6- $(223)_{10} - (223)_{10}$

128	64	32	16	8	4	2	1
1	1	0	1	1	1	1	1

$(223)_{10} = 1\,1\,0\,1\,1\,1\,1\,1$

$(-223)_{10} = 0\,0\,1\,0\,0\,0\,0\,0$

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

$$\begin{array}{r} 1\;1\;1\;1\;1\;1\;1\;\; \\ 1\;1\;0\;1\;1\;1\;1\;1 \\ 0\;1\;0\;0\;0\;0\;0\;1 \\ \hline (1)\;0\;0\;0\;0\;0\;0\;0\;0 \\ \hline \end{array}$$

2	223
2	111-1
2	55-1
2	27-1
2	13-1
2	6-1
	3-

Excess 3 code

Binary coded decimal

Decimal numbers	Binary 8	4	2	1	+3	Excess-3 code			
0	0	0	0	0		0	0	1	1
1	0	0	0	1		0	1	0	0
2	0	0	1	0		0	1	0	1
3	0	0	1	1		0	1	1	0
4	0	1	0	0		0	1	1	1
5	0	1	0	1		1	0	0	0
6	0	1	1	0		1	0	0	1
7	0	1	1	1		1	0	1	0
8	1	0	0	0		1	0	1	1
9	1	0	0	1		1	1	0	0

18-01-2022

Gray code :-

This code is called 'Unit distance Code'. The another name is 'Reflection code'.

It is non-weighted code.

2 bit code

BCD			Gray	
$\boxed{2 \quad 1}$				
0	0		0	0
0	1		0	1
1	0		1	1
1	1		1	0

MSB

write MSB as it is

Perform EX-OR for BCD

EX-OR

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

3 bit code

BCD			Gray		
4	2	1			
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

4 bit code

BCD				Gray			
8	4	2	1				
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

* Applications of Grey code
→ Digital communication
→ Error detection.

Q. 127

$(1\ 1\ 1\ 1\ 1\ 1\ 1\ 1)_2 = (01111111)_2$

Gray code = (1 0 0 0 0 0 0 0)

Q-128 ↓ Binary code

$(1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0)_2$

Gray code = (1 1 0 0 0 0 0 0 0)

Scanned by TapScanner