Binary number system

Binary number system is also called base 2 system. In binary number system, any number can be represented using two digits 0 and 1. It has radix (base) 2. The binary number system is mostly used in computers, where programming language is based on two digit number system.

Ex:-

1	1	0	.	1	1	1	— Binary number
2^2	2^1	2^0	.	2^{-1}	2^{-2}	2^{-3}	— power of base
4	2	0		0.5	0.25	0.125	— Magnitude of each term

Total => 6.875 — Value.

Octal number system

In octal number system any number can be represented using eight digits 0 to 7. It has radix 8. While working on computers it is easier to write number in octal form rather than binary form.

Ex:-

1	4	0	.	2	1	0
8^2	8^1	8^0		8^{-1}	8^{-2}	8^{-3}
64	32	0		0.25	0.015	0

Decimal number system

The number system that we use in our daily life applications is decimal number system. In decimal number system any number can be represented using ten digits 0 to 9. It has radix 10. Each number in a decimal number represents a ~~x~~ power of the base (10). The advantage of the decimal number system are that the mathematic terminology and concept use is identical to the base that is used for counting everyday numbers.

1	4	0	2	1	0
10^2	10^1	10^0	10^{-1}	10^{-2}	10^{-3}
100	40	0	0.2	0.01	0

Hexadecimal Number system

In hexadecimal number system any number can be represented using digits and alphabets as 0 to 9 and A, B, C, D, E, F. It has radix 16. Each position in hexadecimal represents ~~x~~ a power of 16. The hexadecimal number system is

Commonly used by programmers to describe locations in memory because it can represent every byte as two consecutive hexadecimal digits instead of the eight digits.

2	A	0	2	1	0
16^2	16^1	16^0	16^{-1}	16^{-2}	16^{-3}
512	160	0	0.125	0.003	0

⊛ Number conversion system

In daily life we use decimal number system. But computers do not understand decimal number system. So to process computations binary number system is needed. The representation of large numbers in binary is difficult to read. In context to understand long sequence of binary digits, hexadecimal and octal number system are useful. The number system conversion is technique to convert one number in base r to another number system in equivalent base r.

Binary number system is having only zero's and ones. It has only 2 values.
→ The binary digit is known as 'bit'. The group of 4 bits is known as 'nibble' and group of 8 bits is known as 'byte'. A byte is a basic unit in data processing. The range of n bit number is $\boxed{2^n - 1}$

Binary to Decimal Conversion

Ex: convert binary number $(1011.01)_2$ to decimal

1	0	1	1	.	0	1
2^3	2^2	2^1	2^0		2^{-1}	2^{-2}
8	0	2	1	.	0	0.25

$8 + 0 + 2 + 1 + 0 + 0.25 = (11.25)$

$(1011.01)_2 = (11.25)_{10}$

Ex: convert binary number $(0100.10)_2$ to decimal

0	1	0	0	.	1	0
2^3	2^2	2^1	2^0	.	2^{-1}	2^{-2}
0	4	0	0	.	0.5	0

$0 + 4 + 0 + 0 + 0.5 + 0 = 4.5$

$(0100.10)_2 = (4.5)_{10}$.

Binary to octal number Conversion

radix of octal number is 8 and binary number is 2. radix of octal number can be mentioned as $2^3 = 8$ which indicates 3 bits produces an octal number. Append zeros incase of bits less than three.

Convert binary number $(110110.010)_2$ to octal.

$$\overset{2^2}{1}\ \overset{2^1}{1}\ \overset{2^0}{0}\quad \overset{2^2}{1}\ \overset{2^1}{1}\ \overset{2^0}{0}\ .\ \overset{2^0}{0}\ \overset{2^1}{1}\ \overset{2^2}{0}$$

$$\underbrace{110}_{6}\ \underbrace{110}_{6}\ .\ \underbrace{010}_{2}$$

$(110110.010)_2 = (66.2)_8$

Octal	
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Convert binary number $(1010.01)_2$ to octal

$$\underset{2^2\ 2^1\ 2^0}{0\ 0\ 1}\quad \underset{2^2\ 2^1\ 2^0}{0\ 1\ 0}\ .\ \underset{2^0\ 2^1\ 2^2}{0\ 1\ 0}$$

$$1 \quad 2 \ . \ 2$$

$(1010.01)_2 = (12.2)_8$

Binary to Hexadecimal

The hexadecimal number system is mostly used in digital Computation, because it deals with grouping 4 bits. the radix of hexadecimal number is 16 and binary number is 2. the radix hexadecimal number can be mentioned as

$2^4 = 16$, which indicates that grouping of 4 bits produces a hexadecimal number. Append zeros incase bits are less than 3.

(*) Convert binary number $(01011011.1010)_2$ into hexadecimal

0101	1011 . 1010
(5	B . A$)_{16}$

Hexadecimal	
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

(*) Convert binary number $(11111.01)_2$ to hexadecimal.

0001	1111	.	0100
(1	F	.	4$)_{16}$.

(*) Decimal to Binary number system

The method used for converting a decimal integer number to binary number is known as double dabble. It involves successive division. The decimal number is divided by 2. And notedown the remainders. Then use bottom to top approach for obtaining the binary number. The decimal fraction number is converted to binary number by by successive multiplication.

* Convert decimal number $(35.2)_{10}$ into binary.

$$
\begin{array}{r|ll}
2 & 35 & \\
2 & 17 & -1 \\
2 & 8 & -1 \\
2 & 4 & -0 \\
2 & 2 & -0 \\
2 & 1 & -0 \\
 & 0 & -1
\end{array}
$$

$(35)_{10} = (0100011)_2$

$0.2 \times 2 = 0.4 =$ integer 0
$0.4 \times 2 = 0.8$ 〃 0
$0.8 \times 2 = 1.6$ 〃 1
$0.6 \times 2 = 1.2$ 〃 1

$(0.2)_{10} = (0011)_2$

$(35.2)_{10} = (100011.0011)$

* Convert decimal to binary $(105.75)_{10}$

$$
\begin{array}{r|ll}
2 & 105 & \\
2 & 52 & -1 \\
2 & 26 & -0 \\
2 & 13 & -0 \\
2 & 6 & -1 \\
2 & 3 & -0 \\
2 & 1 & -1 \\
 & 1 & -1
\end{array}
$$

$(105)_{10} = (1101001)_2$

$0.75 \times 2 = 1.5$ integer 1
$0.5 \times 2 = 1.0$ integer 1
$0.0 \times 2 = 0$ integer 0

$(0.75)_{10} = (0.110)_2$

$(105.75)_{10} = (1101001.110)_2$

Decimal to octal number system

Decimal number systems can be converted into octal number by dividing by 8, incase of decimal integer and multiply by 8 for the case of decimal fraction.

Convert decimal number system (37.45) into octal.

$$8 \,|\, 37$$
$$4 - 5$$

$(37)_{10} = (45)_8$

0.45 × 8 = 3.60 integer 3
0.6 × 8 = 4.8 integer 4
0.8 × 8 = 6.4 integer 6
0.4 × 8 = 3.2 integer 3

$(0.45)_{10} = (3463)_8$

$(37.45) = (45.3463)_8$

Convert Decimal number $(412.53)_{10}$ into octal

$$8 \,|\, 412$$
$$8 \,|\, 51 - 4$$
$$6 - 3$$

$(412)_{10} = (634)_8$

0.53 × 8 = 4.24 integer 4
0.24 × 8 = 1.92 integer 1
0.92 × 8 = 7.36 integer 7
0.36 × 8 = 2.78 integer 2

$(0.53)_{10} = (4172)_8$

$(412.53)_{10} = (634.4172)_8$

* Decimal to Hexadecimal number system conversion

Decimal number canbe converted to hexadecimal number by dividing by 16, incase of decimal integer and multiply by 16 incase of decimal fraction.

* Convert Decimal number $(107.35)_{10}$ into hexadecimal

$(107)_{10}$

$$
\begin{array}{r|l}
16 & 107 \\
\hline
 & 6 - B
\end{array}
\quad (6B)_{16}
$$

$0.35 \times 16 = 5.6$	integer	5
$0.6 \times 16 = 9.6$	integer	9
$0.6 \times 16 = 9.6$	integer	9
$0.6 \times 16 = 9.6$	integer	9

$(0.35)_{10} = (5999)_{16}$

$$(107.35)_{10} = (6B.5999)_{16}.$$

* Convert Decimal number $(168.53)_{10}$ into hexadecimal

$$
\begin{array}{r|l}
16 & 168 \\
\hline
 & A - 8
\end{array}
\quad (168)_{10} = (A8)_{16}
$$

$0.53 \times 16 = 8.48$	8
$0.48 \times 16 = 7.68$	7
$0.68 \times 16 = A.88$	A
$0.88 \times 16 = E.08$	E

$(0.53)_{10} = (0.87AE)_{16}$

$$(168.53)_{10} = (A8.87AE)_{16}$$

Conversion from octal to any other state

Octal number system has base 8. It is clubbing of 3 binary digits. the range of octal number is from 0 to 7. the Conversion from octal number to other number is important for optimization purpose.

Octal to binary number system conversion.

Octal to binary is reverse to binary to octal system.

Convert octal number $(147.16)_8$ to binary

$$
\begin{array}{ccccc}
1 & 4 & 7 & . & 1 \quad 6 \\
\downarrow & \downarrow & \downarrow & & \downarrow \quad \downarrow \\
001 & 100 & 111 & . & 001 \quad 110
\end{array}
$$

$(001\,100\,111 . 001\,110)_2$

0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Convert octal number $(213.12)_8$ to binary

$$
\begin{array}{cccccc}
2 & 1 & 3 & . & 1 & 2 \\
\downarrow & \downarrow & \downarrow & & \downarrow & \downarrow \\
010 & 001 & 011 & & 001 & 010
\end{array}
$$

$(213.12)_8 = (010\,001\,011 . 001\,010)_2$

Octal to decimal number

As the decimal system is widely used number system in our day to day life. The conversion from octal to decimal number is multiplication of octal bits by its positional weight and summing up the product.

Convert octal to decimal $(6230.41)_8$

$$\begin{array}{cccccccc} 6 & 2 & 3 & 0 & . & 4 & 1 \\ 8^3 & 8^2 & 8^1 & 8^0 & & 8^{-1} & 8^{-2} \end{array}$$

$3072 + 128 + 24 + 0 + 0.5 + 0.015$

$(6230.41)_8 = (3224.575)_{10}$

Convert octal number $(1211.20)_8$ to decimal

$$\begin{array}{cccccccc} 1 & 2 & 1 & 1 & . & 2 & 0 \\ 8^3 & 8^2 & 8^1 & 8^0 & . & 8^{-1} & 8^{-2} \end{array}$$

$512 + 128 + 8 + 1 + 0.25 + 0$

$(1211.20)_8 = (649.25)_{10}$.

Octal to Hexadecimal Conversion

Octal number system is clubbing of 3 binary digits and hexadecimal number system is clubbing of 4 binary digits. Most of the computational techniques are based on hexadecimal number system, because 4 binary digits are grouped into form. first group into 3bits then regroup into 4 bits

Convert octal number $(427.11)_8$ to Hexadecimal.

first convert to octal of 3bits and then convert to 4bits

	4	2	7	. 1	1
	100	010	111	001	001 00
0001					
1		1	7	2	4

$(427.11)_8 = (117.24)_{16}$

Octal	
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Convert octal number $(111.11)_8$ to hexadecimal

1	1	1 .	1	1
001	001	001	001	001
0000 0100	1001		0010	0100
0 4	9	.	2	4

$(111.11)_8 = (049.24)_{16}$

Hexa	
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Conversion from Hexadecimal number system to any other base

Hexadecimal number system has base 16. The range of hexadecimal numbers are from 0-9, A, B, C, D, E, F due to wide range of applications the conversion from hexadecimal number system to another number system is important.

Hexadecimal to binary

Hexadecimal to binary is reverse of binary to hexadecimal.

Convert hexadecimal number $(7A7.B2)_{16}$ to binary

7	A	7	.	B	2
0111	1010	0111	.	1011	0010

$(7A7.B2)_{16} = (011110100111.10110010)_2$

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Convert hexadecimal $(ABC.DE)_{16}$ to binary

A	B	C	.	D	E
1010	1011	1100	.	1101	1110

$(ABC.DE)_{16} = (101010111100.11011110)_2$

Scanned by CamScanner

Hexa decimal to Decimal number

The decimal number system is the major requirement of real time applications. The conversion from hexadecimal to decimal number system is multiplication of bit value by its positional weight and summing up the product

Convert hexadecimal $(6AAA.C1)_{16}$ to decimal

6 A A A . C 1

16^3 16^2 16^1 16^0 16^{-1} 16^{-2}

24576 + 2560 + 160 + 10 + 0.75 + 0.0039

$(27306.7539)_{10}$.

$(6AAA.C1)_{16} = ($~~[illegible]~~$)$.

Convert hexadecimal number $(2381.FF)_{16}$ to decimal

2 3 8 1 . F F

16^3 16^2 16^1 16^0 . 16^{-1} 16^{-2}

8192 + 768 + 128 + 1 + 0.93 + 0.05

$(9089.98)_{10}$

$(2381.FF)_{16} = (9089.98)_{10}$.

hexadecimal to octal system

The hexadecimal number is having radix 16 and it is group of 4 binary digits, whereas octal number system is clubbing of 3 binary digits. first group 4 bits then regroup 3 bits

Convert hexadecimal number $(AA7.1A)_{16}$ to octal

first convert to 4 bits then group 3 bits

A A 7 . 1 A

1010 1010 0111 . 0001 1010

101	010	100	111	.	000	110	100
5	2	4	7	.	0	6	4

$(AA7.1A)_8 = (5247.064)_8$

Convert hexadecimal number $(1CA.EF)_{16}$ to octal

1 C A . E F

0001 1100 1010 . 1110 1111

000	111	001	010	.	111	011	110
0	7	1	2	.	7	3	6

$(1CA.EF)_{16} = (0712.736)_8$

Hexa

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Octa

0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Q No. 9 (a) - Convert the number $(2048)_{10}$ into binary, octal and hexadecimal.

$(2048)_{10}$ into binary

2 | 2048
2 | 1024 - 0
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
2 | 1 - 0
  | 0 - 1

$(2048)_{10} = (100000000000)_2$

$(2048)_{10}$ to octal

8 | 2048
8 | 256 - 0
8 | 32 - 0
8 | 4 - 0
  | 0 - 4

$(2048)_{10} = (4000)_8$

$(2048)_{10}$ to hexadecimal

16 | 2048
16 | 128 - 0
16 | 8 - 0
    | 0 - 8

$(2048)_{10} = (800)_{16}$

Binary arithmatic

Binary number system is the foundation of all computer related operations. Arithmatic operations are possible on binary number system. The procedure for binary arithmatic system is similar to mathematical calculation.

Binary addition

When we add two numbers the resultant is sum and if that sum is having an extra bit, that bit is known as carry bit. The binary addition deals with string of 0 and 1.

Rules

0+0 = 0	Sum 0	carry 0
0+1 = 1	Sum 1	carry 0
1+0 = 1	Sum 1	carry 0
1+1 = 10	Sum 0	carry 1
1+1+1 = 11	Sum 1	carry 1

Add the following binary numbers.

(a) 10 + 10 (b) 11 + 111 (c) 111 + 1111 (d) 101 + 101

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

Sum = $(100)_2$ carry = 1

(b) 11 + 111

```
   011
  1 1 1
 --------
  1010
```

Sum 1010 Carry 1

(c)

```
   0111
   1111
 --------
  10110
```

Sum (10110) Carry = 1

(d)

```
   101
   101
 --------
  1010
```

Sum 1010 Carry = 1

Binary Subtraction

If the value of a is less than b then a digit is taken from more significant position is known as borrow. The binary subtraction is performed on binary bits.

Rules

0 - 0 = 0 Difference = 0 borrow = 0

1 - 0 = 1 Difference = 1 borrow = 0

1 - 1 = 0 Difference = 0 Borrow = 0

0 - 1 = 1 Difference = 1 Borrow = 1

(*) Subtract the following binary bits

(a) 1011 − 1001 (b) 10000 − 101

(a)
```
  1011
- 1001
  ----
  0010
```

Difference (0010) Borrow 0

(b)
```
  10000
-   101
  -----
   1011
```

Borrowing from a number gives bit 10
10 means $\underset{2^1}{1}\ \underset{2^0}{0} = 2$ again to borrow from
this $\cancel{1} \rightarrow 1$ given to next number becomes again
10, 10 means $\cancel{1}$ again to borrow next
number 10 10 mean 2 it will continue
up to higher number comes where borrow
stops. $2-1=1$

Difference $= (1011)_2$

Borrow = 0.

(*) Binary Multiplication

Binary Multiplication is performed on same manner as decimal multiplication. partial products created and successive partial product is shifted to left. then add all the partial products

Rules

$0 \times 1 = 0$

$1 \times 0 = 0$

$1 \times 1 = 1$

$0 \times 0 = 0$

7) perform binary multiplication

(a) 111×111 (b) 1001×1111

(a)

$$
\begin{array}{r}
111 \\
\times \quad 111 \\
\hline
111 \\
111\times \\
111\times\times \\
\hline
110001
\end{array}
$$

$(110001)_2$

(b)

$$
\begin{array}{r}
1001 \\
1111 \\
\hline
1001 \\
1001\times \\
1001\times\times \\
1001\times\times\times \\
\hline
10000111
\end{array}
$$

$(10000111)_2$

(4) Express the following numbers in decimal

① $(10110.0101)_2$, $(16.5)_{16}$, $(26.24)_8$.

① $(10110.0101)_2$

1 0 1 1 0 . 0 1 0 1

$2^4 + 2^3 + 2^2 + 2^1 + 2^0 \;.\; 2^{-1} + 2^{-2} + 2^{-3} + 2^{-4}$.

$16 + 0 + 4 + 2 + 0 \;.\; 0 + 0.25 + 0 + 0.0625$

$(22.3125)_{10}$.

② $(16.5)_{16}$

1 6 . 5

$16^1 + 16^0 \;.\; 16^{-1}$

$16 + 6 \;.\; 0.3125$

$(22.3125)_{10}$

③ $(26.24)_8$

2 6 . 2 4

$8^1 + 8^0 \;.\; 8^{-1} + 8^{-2}$

$16 + 6 \;.\; 0.25 + 0.0625$

$(22.3125)_{10}$.

(7) Solve for x

① $(F3A7C2)_{16} = (x)_{10}$

② ~~(1011111)~~ $(2AC5)_{16} = (10949)_x$

③ $(0.93)_{10} = (x)_8$

(4) $(4057.06)_8 = (x)_{10}$

① $(F3A7C2)_{16} = (x)_{10}$

F 3 A 7 C 2

$16^5 + 16^4 + 16^3 + 16^2 + 16^1 + 16^0$

$+ 2$

$\Rightarrow 15968194$

② $(2AC5)_{16} = (10949)_x$

$2\times16^3 + A\times16^2 + C\times16^1 + 5\times16^0 = 1x^4 + 0x^3 + 9x^2 + 4x^1 + 9\times x^0$

$2\times16^3 + 10\times16^2 + 16\times12 + 5 = x^4 + 9x^2 + 4x + 9$

③ $(0.93)_{10} = (x)_8$

$0.93\times8 \Rightarrow 7.44 \to 7$

$0.44\times8 = 3.52 \to 3$

$0.52\times8 = 4.16 \to 4$

$0.16\times8 = 1.28 \to 1$

$\Rightarrow (0.7341)_8$

③ $(4057.6)_8 = (x)_{10}$

$$\begin{array}{ccccc} 4 & 0 & 5 & 7. & 6. \\ 8^3 + & 8^2 + & 8^1 + & 8^0 . & 8^{-1} \end{array}$$

$2048 + 0 + 40 + 7 . 0.75$

$(2095.75)_{10}$

④ Given 2 binary numbers $x = (1010100)_2$
$y = (1000011)_2$ perform 2's complement subtraction
① $x-y$ ② $y-x$.

① $x-y$

$x = 1010100$ $y = 1000011$

take 1's complement for y - 0111100

$$\begin{array}{r} 0111100 \\ 1 \\ \hline 0111101 \end{array}$$

2's complement

$x-y \Rightarrow$

$$\begin{array}{r} 1010100 \\ 0111101 \\ \hline 10010001 \end{array}$$ Ans

... 1's complement for x 0101011

2's complement for x

$$
\begin{array}{r}
0101011 \\
+\quad 1 \\
\hline
0101100
\end{array}
$$

Y-X =

$$
\begin{array}{r}
1000011 \\
0101100 \\
\hline
1101111
\end{array}
$$

Ans

(7) Given that $(81)_{10} = (100)_b$ find the value of b.

$(9^2) = (10^2)$

$8\times10^1+1\times10^0 = 1\times b^2+0\times b^1+0\times b^0$

$80+1 = b^2$

$81 = b^2$

$b = \sqrt{81} = 9.$

$\underline{\underline{b=9.}}$

(7) Convert the following to Decimal and then octal

(1) $(125F)_{16}$ (2) $(1011111)_2$ (3) $(4234)_5$

(1) $(125F)_{16}$. Hexadecimal to Decimal

1 2 5 F

$16^3+16^2+16^1+16^0 \Rightarrow 4096+256+80+1 \Rightarrow 4433$

Decimal to octal.

$(4433)_{10}$

$$\begin{array}{r|l} 8 & 4433 \\ 8 & 554 - 1 \\ 8 & 69 - 2 \\ 8 & 8 - 5 \\ 8 & 1 - 0 \\ & 0 - 1 \end{array}$$

$(4433)_{10} = (10521)_8.$

Scanned by CamScanner

(2) $(10111111)_2$ Binary to Decimal.

1 0 1 1 1 1 1 1

$2^7 + 2^6 + 2^6 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0$

$\Rightarrow 128 + 0 + 32 + 16 + 8 + 4 + 2 + 1$

$= (191)_{10}$

$(191)_{10}$ Decimal to octal.

$$\begin{array}{l|ll} 8 & 191 & \\ 8 & 23 & - 7 \\ 8 & 2 & - 7 \\ & 0 & - 2 \end{array}$$

$(191)_{10} = (277)_8$

(3)

(*) Solve for x

(1) $(367)_8 = (x)_2$. (2) $(378.93)_{10} = (x)_8$

(3) $(B9F.AE)_{16} = (x)_8$ (4) $(16)_{10} = (100)_x$

(5) Convert $(163.875)_{10}$ to binary octal and hexadecimal.

(1) $(367)_8 = (x)_2$

$$\left(\begin{array}{ccc} 3 & 6 & 7 \\ 011 & 110 & 111 \end{array}\right)_8 = (011\ 110\ 111)_2$$

(2) $(378.93)_{10} = (x)_8$

$$\begin{array}{r|ll} 8 & 378 & \\ 8 & 47 & -2 \\ 8 & 5 & -7 \\ & 0 & -5 \end{array}$$

$(572)_8$

$(378.93)_{10} = (572)_8$

(3) $(B9F.AE)_{16} = (x)_8$.

B 9 F . A E

1011 1001 1111 . 1010 1110

Regroup into 3 bits

$$\underset{5}{\underline{101}}\ \underset{6}{\underline{110}}\ \underset{3}{\underline{011}}\ \underset{7}{\underline{111}} . \underset{5}{\underline{101}}\ \underset{3}{\underline{011}}\ \underset{4}{\underline{100}} = (5637.534)_8.$$

$1\times10+6\times10 = 1\times x^2+0\times x+0\times x$

$10+6 = x^2$

$16 = x^2$

$x=\sqrt{16}=4$

(4) $(16)_{10} = (100)_x$

$(16)_{10} = (100)_x$

(4^2) (10^2)

$x = 4$

(5) Convert $(163.875)_{10}$ into binary, octal, hexadecimal.

Binary

2 | 163
2 | 81 − 1
2 | 40 − 1
2 | 20 − 0
2 | 10 − 0
2 | 5 − 0
2 | 2 − 1
2 | 1 − 0
  | 0 − 1

$(10100011)_2$

$0.875\times2 = 1.750 \rightarrow 1$

$0.750\times2 = 1.50 \rightarrow 1$

$0.5\times2 = 1.0 \rightarrow 1$

$0.0\times2 = 0.0 \rightarrow 0.$

$(.1110)$

$(163.875)_{10} = (10100011.1110)_2$

Octal

8 | 163
8 | 20 − 3
8 | 2 − 4
  | 0 − 2

$(243)_8$

$0.875\times8 = 7.00 \rightarrow 7$

$0.0\times8 = 0.0 \rightarrow 0$

$0.0\times8 = 0.0 \rightarrow 0$

$0.0\times8 = 0.0 \rightarrow 0.$

$(163.875)_{10} = (243.7000)_8$

hexadecimal

16 | 163
16 | 10 − 3
  | 0 − A

$(A3)_{16}$

$0.875\times16 = 14.00 \rightarrow E$

$0.0\times16 = 0.0 \rightarrow 0$

$(163.875)_{10} = (A3.E)_{16}$

(6) Convert $(946)_{10}$ into binary and hexadecimal.

$$
\begin{array}{r|l}
2 & 946 \\
2 & 473 - 0 \\
2 & 236 - 1 \\
2 & 118 - 0 \\
2 & 59 - 0 \\
2 & 29 - 1 \\
2 & 14 - 1 \\
2 & 7 - 0 \\
2 & 3 - 1 \\
2 & 1 - 1 \\
 & 0 - 1
\end{array}
$$

$(946)_{10} = (\quad)_2$

$(1110110010)_2$

$$
\begin{array}{r|l}
16 & 946 \\
16 & 59 - 2 \\
16 & 3 - B \\
 & 0 - 3
\end{array}
$$

$(946)_{10} = (3B2)_{16}$.

(7) Convert the given octal number $(2564.603)_8$ to hexadecimal number.

2	5	6	4	.	6	0	3
010	101	110	100	.	110	000	011 000

regroup into 4 bits

0101	0111	0100	.	1100	0001	1000
5	7	4	.	C	1	8

$(574.C18)_{16}$

Octal

000 – 0
001 – 1
010 – 2
011 – 3
100 – 4
101 – 5
110 – 6
111 – 7

Hexa

0000 – 0	1010 – A
0001 – 1	1011 – B
0010 – 2	1100 – C
0011 – 3	1101 – D
0100 – 4	1110 – E
0101 – 5	1111 – F
0110 – 6	
0111 – 7	
1000 – 8	
1001 – 9	

(7) Perform the following conversions $(476.64)_{10} = (\quad)_2 = (\quad)_8$

$(476.64)_{10} = (\quad)_2$

$$
\begin{array}{r|l}
2 & 476 \\
2 & 238 - 0 \\
2 & 119 - 0 \\
2 & 59 - 1 \\
2 & 29 - 1 \\
2 & 14 - 1 \\
2 & 7 - 0 \\
2 & 3 - 1 \\
2 & 1 - 1 \\
 & 0 - 1
\end{array}
$$

(111011100)

$0.64 \times 2 = 1.28 \rightarrow 1$

$0.28 \times 2 = 0.56 \rightarrow 0$

$0.56 \times 2 = 1.12 \rightarrow 1$

$0.12 \times 2 = 0.24 \rightarrow 0$

$(.1010)$

$(476.64)_{10} = (111011100.1010)_2$

$(476.64)_{10} = (\quad)_8$

$$
\begin{array}{r|l}
8 & 476 \\
8 & 59 - 4 \\
8 & 7 - 3 \\
 & 0 - 7
\end{array}
$$

(734)

$0.64 \times 8 = 5.12 \rightarrow 5$

$0.12 \times 8 = 0.96 \rightarrow 0$

$0.96 \times 8 = 7.68 \rightarrow 7$

$0.68 \times 8 = 5.44 \rightarrow 5$

$(.5075)$.

$(476.64)_{10} = (734.5075)_8$

Scanned by CamScanner

1's Complement Representation

The positive numbers are represented in similar manner for sign magnitude and 1's complement form. For negative numbers, complements have been taken for each bit including the sign bit. The 1's complement can be calculated by complementing each bit i.e 0 for 1 and 1 for 0. The 1's complement of A is $\bar{A}$.

Represent the following number in 8 bit 1's complement form.

(a) $(44)_{10}$ (b) $(57)_{10}$ (c) $(37)_{10}$ (d) $(72)_{10}$.

(a) $(44)_{10}$ 8bit representation of $(44)_{10} = (00101100)_2$

1's complement of $(44)_{10} = (11010011)_2$

(b) $(57)_{10}$ 8 bit representation $(57)_{10} = (00111001)_2$

1's complement of $(57)_{10} = (11000110)$

(c) $(37)_{10}$ 8 bit representation $(37)_{10} = (00100101)_2$

1's complement of $(37)_{10} = (11011010)_2$

(d) $(72)_{10}$ 8 bit representation $(72)_{10} = (01001000)_2$

1's complement of $(72)_{10} = (10110111)_2$

2's complement

+ve numbers are represented using in similar manner for sign magnitude and 2's complement form. For negative numbers complements have been taken for each bit including the sign bit

2's complement = 1's complement + 1.

Represent the following number in 8 bit 2's complement form

(a) $(-44)_{10}$ (b) $(-57)_{10}$ (c) $(-37)_{10}$ (d) $(-72)_{10}$.

(a) $(-44)_{10}$

8 bit representation of 44 $(00101100)_2$

1's complement of $(44)_{10}$ – $(11010011)_2$

2's complement of $(44)_{10}$ –
```
  11010011
        +1
  --------
  11010100
```

Answer = 11010100

(b) $(-57)_{10}$

8 bit representation of 57 – 00111001

1's complement of $(57)_{10}$ – 11000110

2's complement of $(57)_{10}$ –
```
  11000110
        +1
  --------
  11000111
```

Answer 11000111

(c) 8 bit representation of $(37)_{10}$ = (00100101)

1's complement of $(37)_{10}$ = 11011010

2's complement of $(37)_{10}$ =

$$\begin{array}{r} 11011010 \\ +1 \\ \hline 11011011 \end{array}$$

(d) 8 bit representation of $(72)_{10}$ = $(01001000)_2$

1's complement of $(72)_{10}$ = (10110111)

2's complement of $(72)_{10}$ =

$$\begin{array}{r} 10110111 \\ 1 \\ \hline 10111000 \end{array}$$

(*) Find 9's complement for the given decimal number

(a) 532 (b) 4367 (c) 71654

(a) 532 is having 3 digits n = 3
9's complement of 532 = $(10^3 - 1) - 532$
$= (1000 - 1) - 532$
$= 999 - 532 = 467$

$(r-1)$'s complement of $k = (r^n - 1) - k$. where n = No. of digits
r is radix or base k is number

(b) 9's complement of 4367 there are 4 digits $= (10^4 - 1) - 4367$
$= (10000 - 1) - 4367 = 9999 - 4367$
$= 5632$

(c) 9's complement of 71654 have 5 digits

$= (10^5 - 1) - 71654$
$= (100000 - 1) - 71654$
$= 99999 - 71654$
$= \underline{\underline{28345}}$

(*) Find 10's complement of following numbers.

(a) 532 (b) 4367 (c) 71654

(a) 532 having no. of digits 3

10's complement of 532 is $= (10^3 - 532)$
$= 1000 - 532$
$= 468$

(b) 4367 having 4 digits

10's complement of 4367 is $= (10^4 - 532)$
$= 10000 - 532$
$= 5633$

(c) 71654 is having 5 digits

10's complement of 71654 is $= (10^5 - 71654)$
$= 100000 - 71654$
$= 28346$

Subtract 11011 − 1001 using 1's compliment using (r−1)'s complement

Equate numbers of digits of both binary numbers

11011 − 01001

Take 1's complement of 01001 = 10110

Add 1's complement of subtrahend to minuend.

```
   10110
   11011
  ------
 1 10001
```

Check wether carry present or not

there is a carry add the carry to LSB of the sum and that is final result.

```
  10001
     +1
  -----
  10010
```

— Answer

Subtract using 2's complement 1011 − 10011 using (r−1)'s complement

Equate both binary numbers

01011 − 10011

Take 1's complement of 10011 = 01100

Add 1's complement of subtrahend to minuend.

```
  01100
  01011
  -----
  10111
```

Check wether carry present or not

The addition does not present carry taken again 1's complement of 10111 → 01000 Answer

* Subtract 62516 − 4234 using 9's complement using (r−1)'s complement

Equate both numbers 62516 − 04234

Take 9's complement of 04234

$= (10^5 - 1) - 04234$

$= 95765$

Add 9's complement of subtrahend to minuend

$$
\begin{array}{r}
95765 \\
62516 \\
\hline
158281
\end{array}
$$

Check whether carry present (or) not

Addition produces a carry.

Add carry to LSB

$$
\begin{array}{r}
58281 \\
1 \\
\hline
58282
\end{array}
$$

Answer

* Subtract 1011 − 10011 using 2's complement using 2's complement

Equate 01011 − 10011

Take 2's complement of 10011

1's complement 01100

2's complement 01101

Add 2's complement of subtrahend to minuend.

$$
\begin{array}{r}
01101 \\
01011 \\
\hline
11000
\end{array}
$$

Check whether carry is present (or) not

Addition does not produce carry

Take again 2's complement 11000 − 00111

and 00111
+1

01000 Answer if carry present discard carry and write answer

~~BCD~~ Codes

In digital number systems for the fast processing of information data is used in binary format. Various binary codes are used to represent data which may be numeric, alphabets (or) special characters. Some are weighted and few are nonweighted codes, although the information used in every code is represented in binary form, yet the information of this binary information is possible only if the code in which this information is available is known.

Binary coded decimal

For fast calculation and conversion binary codes decimal numbers are used. It is used to display decimal number in binary form. In this decimal number is written by replacing decimal digit in integers and fractions with its four bit binary equivalent value. The range of BCD numbers are 0-9. The most commonly used BCD code is 8421, where the digits 8, 4, 2, 1 depicts the weights of different bits in combination of 4 bits, starting from MSB to LSB. Other codes are 4221 and 5421 these Codes are weighted binary Codes because each position of a number represents a specific weight

Decimal	BCD code 8421	4221	5421
0	0000	0000	0000
1	0001	0001	0001
2	0010	0010	0010
3	0011	0011	0011
4	0100	1000	0100
5	0101	1001	1000
6	0110	1100	1001
7	0111	1101	1010
8	1000	1110	1011
9	1001	1111	1100

⊛ Convert BCD numbers into Decimal numbers

1) 10010001 2) 00010001

1) Group 4 bits and then convert to decimal

$10010001 \Rightarrow \underbrace{1001}_{9}\ \underbrace{0001}_{1} = (91)_{10}$

2) Group 4 bits and then convert to decimal

$00010001 = \underbrace{0001}_{1}\ \underbrace{0001}_{1} = (11)_{10}$.

(8)

BCD addition

BCD code is commonly used in arithmetic operation. In BCD addition, if the sum exceeds than 9 it is invalid result which can be avoided by adding 6 (0110). The six digits 10 to 15 are not allowed in BCD numbers so 6 is added.

Perform BCD addition

(a) 1001 + 0001 (b) 00010001 + 01010101

(a)

```
   1001
   0001
  -----
   1010   — invalid BCD sum is out of limits
   0110        add 0110 = 6.
  -----
 1 0000   — represent this in 4 bits
```

0001 0000 = $(00010000)_2$

1 0

(b)

```
 0001 0001
 0101 0101
 ---------
 0110 0110   BCD code is within limits
```

$= (0110\,0110)_2$

BCD Subtraction

BCD subtraction follows subtraction of r's complement and (r-1)'s complement rules.

* Perform BCD subtraction 888 − 243 using
(a) 9's complement (b) 10's complement

(a) 9's complement of 243 = $(10^3 - 1) - 243$
$= (1000 - 1) - 243$
$= 756$

Add to minuend 888 = 756 + 888 = 1644

there is carry add carry to LSD

= 644 + 1 = 645 Answer

(b) 10's complement of 243 = $(10^3) - 243$
$= 757$

Add to minuend 888 = 757 + 888 = 1645

there is carry discard carry remaining result is answer

= 645

Unit distant code

It is an unweighted code that change at only one digit position when going from one number to the next in a consecutive sequence of numbers. Examples of unit distance codes are cyclic codes, Gray code, excess -3 code.

Excess-3 code

Excess -3 code is one type of non weighted code where 3 is added to each decimal digit and result is represented in four binary bits. Excess -3 code is self complementing code which is helpful in subtraction. self complementing is a code which represents 9's complement of a decimal codes when its digits are inverted. the decimal digit 3 is added to each value to extract excess -3 code

Digit	8421	Excess-3
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

⊛ perform excess-3 addition of following numbers

(a) 1011 + 1001 (b) 0101 + 1010

(a) both numbers are 4 digits

$$\begin{array}{r} 1011 \\ 1001 \\ \hline 10100 \end{array}$$

carry generated so group in to 4 digits and add 3 (0011) to the sum

$$\begin{array}{r} 0001 \\ 0011 \\ \hline 0100 \end{array} \quad \begin{array}{r} 0100 \\ 0011 \\ \hline 0111 \end{array}$$

Result $(01000111)_2$

(b) Both numbers are in 4 digits

$$\begin{array}{r} 0101 \\ 1010 \\ \hline 1111 \end{array}$$

— No carry generated subtract 3(0011) from sum

$$\begin{array}{r} 1111 \\ -\ 0011 \\ \hline 1100 \end{array}$$

— Result

⊛ perform excess-3 addition

(a) 46 (b) 430

(a)

$$\begin{array}{rr} 4 & 6 \\ +3 & +3 \\ \hline 7 & 9 \end{array}$$

0111 1001

= (0111 1001)

(b)

$$\begin{array}{rrr} 4 & 3 & 0 \\ +3 & 3 & 3 \\ \hline 7 & 6 & 3 \end{array}$$

0111 0110 0011

$= (0111\ 0110\ 0011)_2$

Excess-3 Subtraction

Perform Excess-3 subtraction

(a) $(3)_{10} - (6)_{10}$ (b) $(6)_{10} - (3)_{10}$

(a) 3 = 3 + 3 = 6 = 0110 → minuend.
6 = 6 + 3 = 9 = 1001 — subtrahend

1001 is

1's complement of Excess-3 subtrahend 0110

Add complemented value to minuend

$$\begin{array}{r} 0110 \\ +0110 \\ \hline 1100 \end{array}$$

Carry not generated
subtract 0011 from result

$$\begin{array}{r} 1100 \\ -0011 \\ \hline 1001 \end{array}$$

It is negative number again take

1's complement $\underline{0110}$.

(b) Excess-3 of 6 = 6+3 = 9 = 1001 minuend
" 3 = 3+3 = 6 = 0110 — Subtrahend

1's complement of Excess-3 of subtrahend — 1001

Add 1's complement of subtrahend to minuend — 1001

$$\begin{array}{r} 1001 \\ 1001 \\ \hline (1)\,0010 \end{array}$$

Carry generated add carry to LSB

$$\begin{array}{r} 0010 \\ +1 \\ \hline 0011 \end{array}$$ Result

10 (a)

Gray code

Gray code is one type of Non weighted binary code. It is not a positional weight. It is a binary code that progresses in such a way that between two successive codes, only one bit is changed. It is not arithmetic code. Single bit change property is important in some applications Ex: shaft position encoders. In these applications the chances of error increases if more than one bit change occurs. Gray codes are also called reflected codes.

Binary to Gray Code Conversion

1. The left most bit (MSB) in gray code is same as the left most bit in binary

MSB 1 0 1 1 Binary → 1 in Gray

2. Add the left most bit to the adjacent bit

1 + 0 1 1 → 1 1 in Gray

3. Add the next adjacent bit and discard carry

1 0 + 1 1 → 1 1 1 in Gray.

4. Continue the above process till completion.

1 0 1 + 1 → 1 1 1 0

(*) Convert the following binary to Gray numbers.

(a) 10110 (b) 01010

(a) 10110

$\downarrow$

$(11101)_{Gray}$

(b) 01010

$\downarrow$

01111_{Gray}

(*) Gray to Binary Conversion.

① 11101 ② 01101

11101

1 0 1 1 0

0 1 1 0 1

0 1 0 0 1

Combinational logic System

(*) Examples of Combinational circuits are adder, subtractor, Multiplexer, demultiplexer, encoder, decoder etc.

A combinational logic system is a type of digital logic which is implemented using Boolean circuits and output of combinational logic system is dependant purely on present input. It is also known as time independant logic. Combinational logic is used to build circuits that produce specified outputs from certain inputs. The logic gates are building blocks of combinational logic system. The three main ways of specifying the function of a combinational logic circuit are-

① Boolean algebra : It is an algebraic expression that depicts the operation of the logic circuit for each input variable either True or False that results in a logic 1 o/p.

② Truth table : A truth table

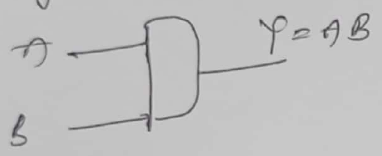
logic Gates

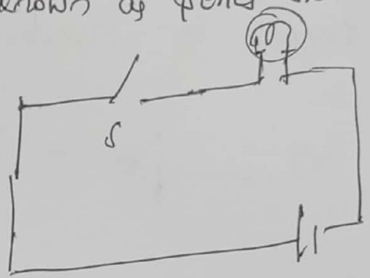
logic gates are different types. they are
① AND ② OR ③ NOT ④ NAND ⑤ NOR ⑥ EXOR
⑦ EXNOR. out of these AND, OR, NOT gates are called basic gates, NAND, NOR gates are called universal gates, EXOR, EXNOR gates are called special gates.

Basic gates

① AND gate:

AND gate can be represented by a symbol shown in fig. It is also known as series switch.





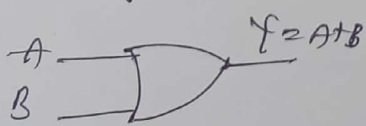
Truth table for AND gate

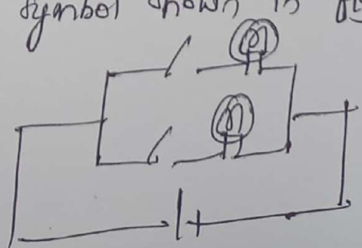
Inputs

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

② OR gate:

OR gate can be represented by the symbol shown in fig. It is also known as parallel switch.



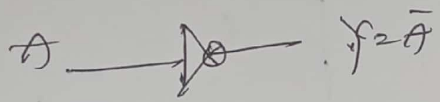


Truth table for OR gate

Inputs		Output
A	B	Y = A+B
0	0	0
0	1	1
1	0	1
1	1	1

NOT gate :

NOT gate is represented by a symbol as shown in fig. It is also known as inverter gate.



Truth table

Input A	Output $Y = \bar{A}$
0	1
1	0

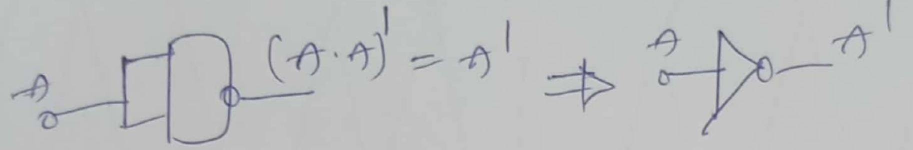
(*) Discuss about universal gates

Ans:- A universal gate is a gate which can be implemented any boolean function without need to use any other gate type.

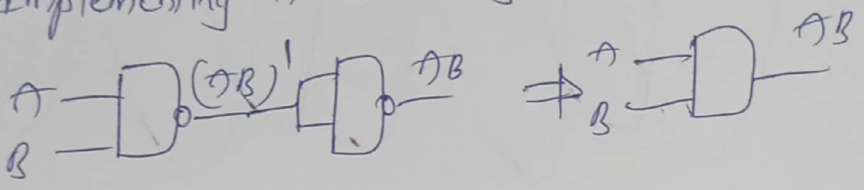
The NAND and NOR gates are universal gates. In practice this is advantageous since NAND and NOR gates are economical and easier to fabricate and are the basic gates used in all IC digital logic families.

NAND gate as a universal gate

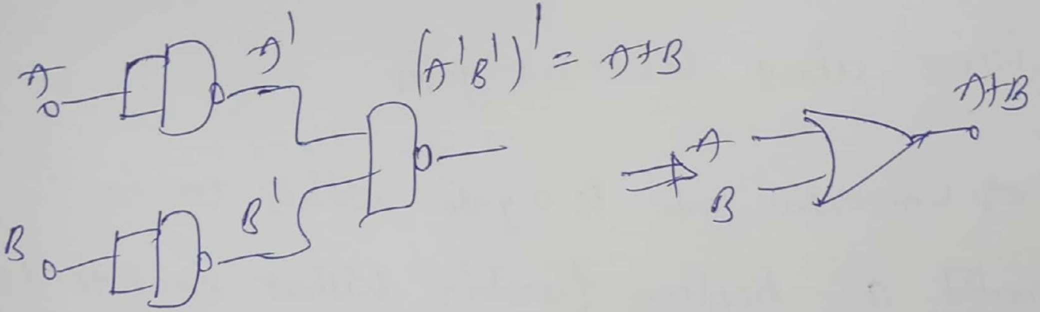
Implementing inverter using NAND gate



Implementing AND using NAND.

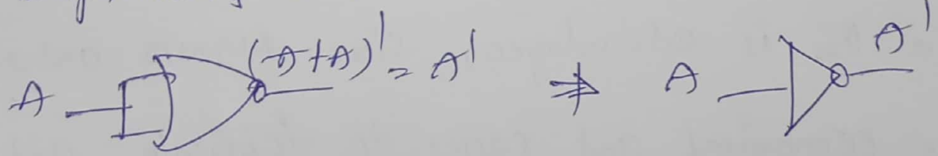


Implementing OR using NAND

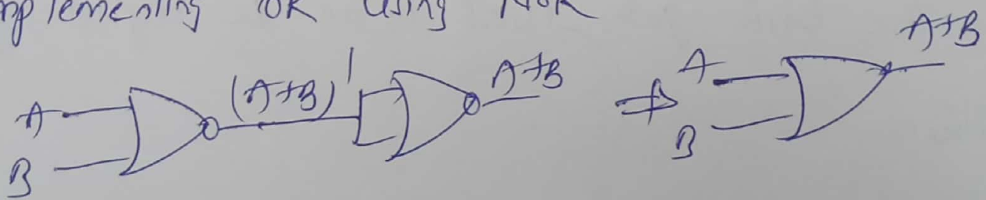


NOR gate as a universal gate

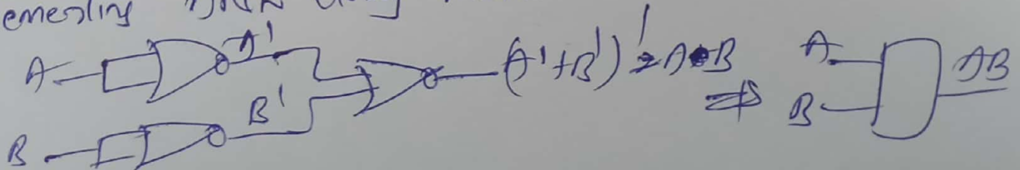
Implementing ~~...~~ inverter as NOR



Implementing OR using NOR



Implementing AND using NOR



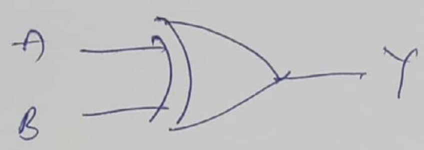
(*) Discuss about special gates Explain them in detail

Ans XOR and XNOR gates are special types of gates. these can be used in half adder, Full adder Full subtractor.

XOR gate.

Ex-OR (or) X-OR gate is a special type of gate

Logic diagram

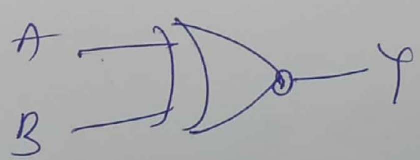


Truth table

Inputs		Outputs
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Ex NOR gate

Logic diagram



Truth table

Inputs		Outputs
A	B	$Y = A \odot B$
0	0	1
0	1	0
1	0	0
1	1	1