

# 1 Number Systems & Logic Gates

- Number Base
- Number Base Conversion
- Basic Gates

# DIGITAL SYSTEMS

Digital systems have such a prominent role in everyday life used in Communication, Medical treatment, Weather monitoring, the Internet, and many other commercial, industrial, and scientific enterprises.

Examples of Digital Devices are digital telephones, digital televisions, digital versatile discs, digital cameras, handheld devices, and, of course, digital computers.

One characteristic of digital systems is to represent discrete information.

- the 10 decimal digits: 0 - 9,
- the 26 letters of the alphabet: a – z; A – Z ,
- the 52 playing cards,
- the 64 squares of a chessboard.

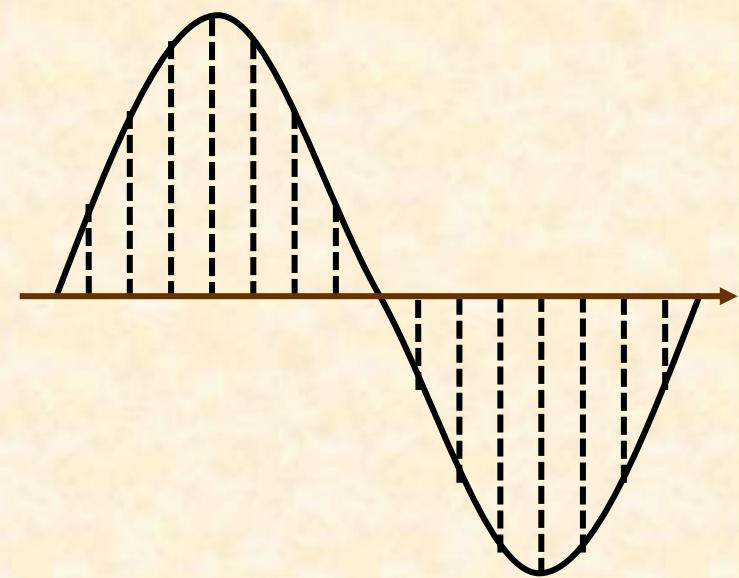
CONTROL CHARACTERS				GRAPHIC SYMBOLS											
NAME	DEC	BINARY	HEX	SYMBOL	DEC	BINARY	HEX	SYMBOL	DEC	BINARY	HEX	SYMBOL	DEC	BINARY	HEX
NUL	0	0000000	00	space	32	0100000	20	@	64	1000000	40	`	96	1100000	60
SOH	1	0000001	01	!	33	0100001	21	A	65	1000001	41	a	97	1100001	61
STX	2	0000010	02	"	34	0100010	22	B	66	1000010	42	b	98	1100010	62
ETX	3	0000011	03	#	35	0100011	23	C	67	1000011	43	c	99	1100011	63
EOT	4	0000100	04	\$	36	0100100	24	D	68	1000100	44	d	100	1100100	64
ENQ	5	0000101	05	%	37	0100101	25	E	69	1000101	45	e	101	1100101	65
ACK	6	0000110	06	&	38	0100110	26	F	70	1000110	46	f	102	1100110	66
BEL	7	0000111	07	,	39	0100111	27	G	71	1000111	47	g	103	1100111	67
BS	8	0001000	08	(	40	0101000	28	H	72	1001000	48	h	104	1101000	68
HT	9	0001001	09	)	41	0101001	29	I	73	1001001	49	i	105	1101001	69
LF	10	0001010	0A	*	42	0101010	2A	J	74	1001010	4A	j	106	1101010	6A
VT	11	0001011	0B	+	43	0101011	2B	K	75	1001011	4B	k	107	1101011	6B
FF	12	0001100	0C	,	44	0101100	2C	L	76	1001100	4C	l	108	1101100	6C
CR	13	0001101	0D	-	45	0101101	2D	M	77	1001101	4D	m	109	1101101	6D
SO	14	0001110	0E	.	46	0101110	2E	N	78	1001110	4E	n	110	1101110	6E
SI	15	0001111	0F	/	47	0101111	2F	O	79	1001111	4F	o	111	1101111	6F
DLE	16	0010000	10	0	48	0110000	30	P	80	1010000	50	p	112	1110000	70
DC1	17	0010001	11	1	49	0110001	31	Q	81	1010001	51	q	113	1110001	71
DC2	18	0010010	12	2	50	0110010	32	R	82	1010010	52	r	114	1110010	72
DC3	19	0010011	13	3	51	0110011	33	S	83	1010011	53	s	115	1110011	73
DC4	20	0010100	14	4	52	0110100	34	T	84	1010100	54	t	116	1110100	74
NAK	21	0010101	15	5	53	0110101	35	U	85	1010101	55	u	117	1110101	75
SYN	22	0010110	16	6	54	0110110	36	V	86	1010110	56	v	118	1110110	76
ETB	23	0010111	17	7	55	0110111	37	W	87	1010111	57	w	119	1110111	77
CAN	24	0011000	18	8	56	0111000	38	X	88	1011000	58	x	120	1111000	78
EM	25	0011001	19	9	57	0111001	39	Y	89	1011001	59	y	121	1111001	79
SUB	26	0011010	1A	:	58	0111010	3A	Z	90	1011010	5A	z	122	1111010	7A
ESC	27	0011011	1B	;	59	0111011	3B	[	91	1011011	5B	{	123	1111011	7B
FS	28	0011100	1C	<	60	0111100	3C	\	92	1011100	5C		124	1111100	7C
GS	29	0011101	1D	=	61	0111101	3D	]	93	1011101	5D	}	125	1111101	7D
RS	30	0011110	1E	>	62	0111110	3E	^	94	1011110	5E	~	126	1111110	7E
US	31	0011111	1F	?	63	0111111	3F	_	95	1011111	5F	Del	127	1111111	7F

# DIGITAL SYSTEMS

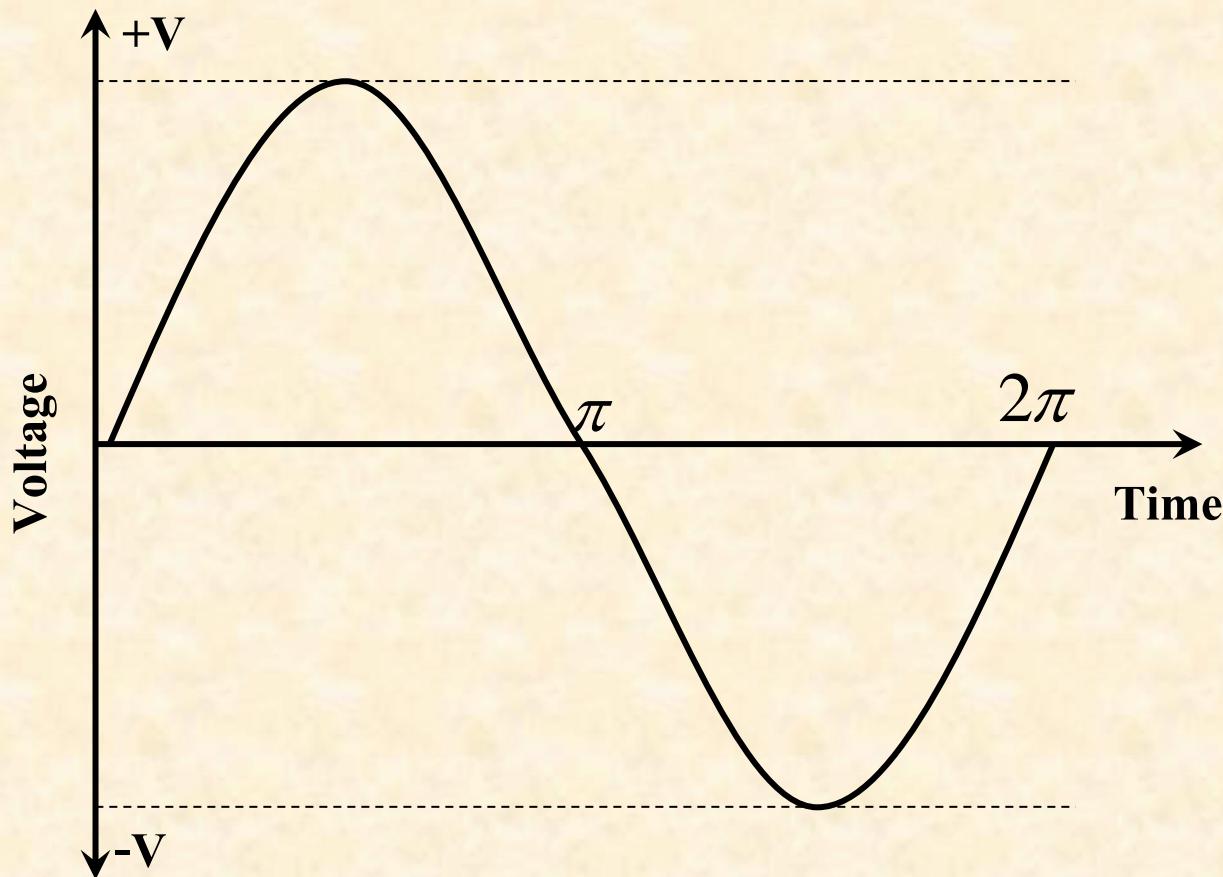
Discrete information is represented in a digital system by physical quantities called **signals**.

Electrical signals such as voltages and currents are the most common.

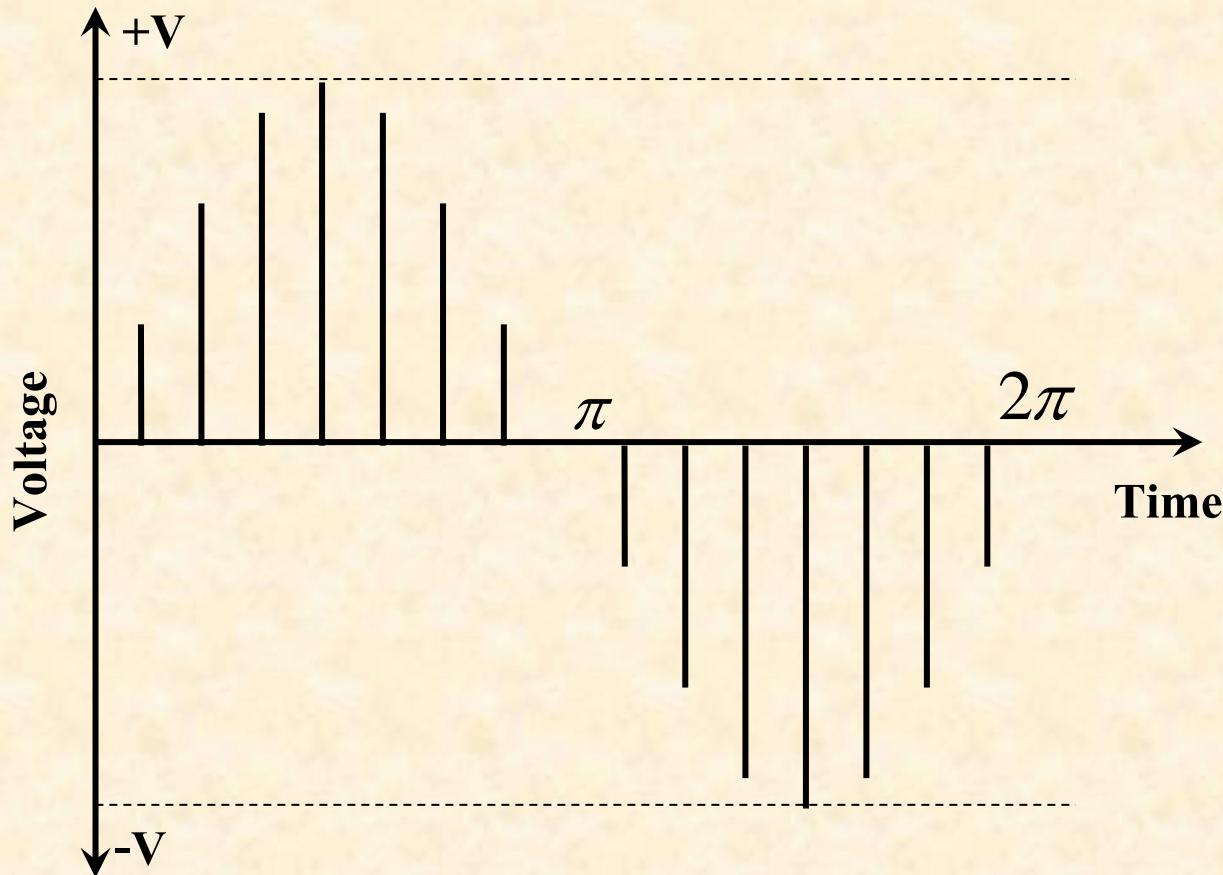
- **Analog Signals**
  - Analog signals: Continuous
- **Digital Signals**
  - Digital Signals: Discrete



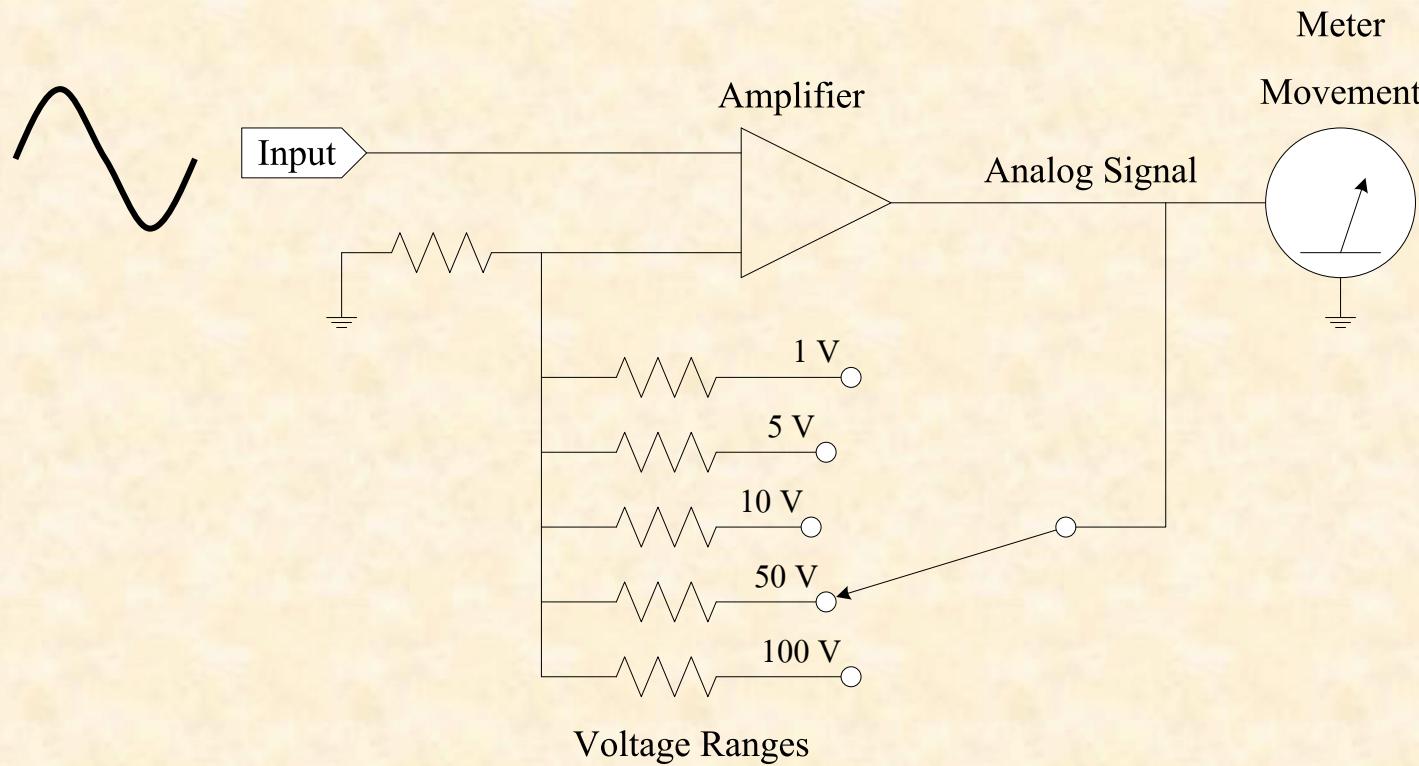
# Analog Signals



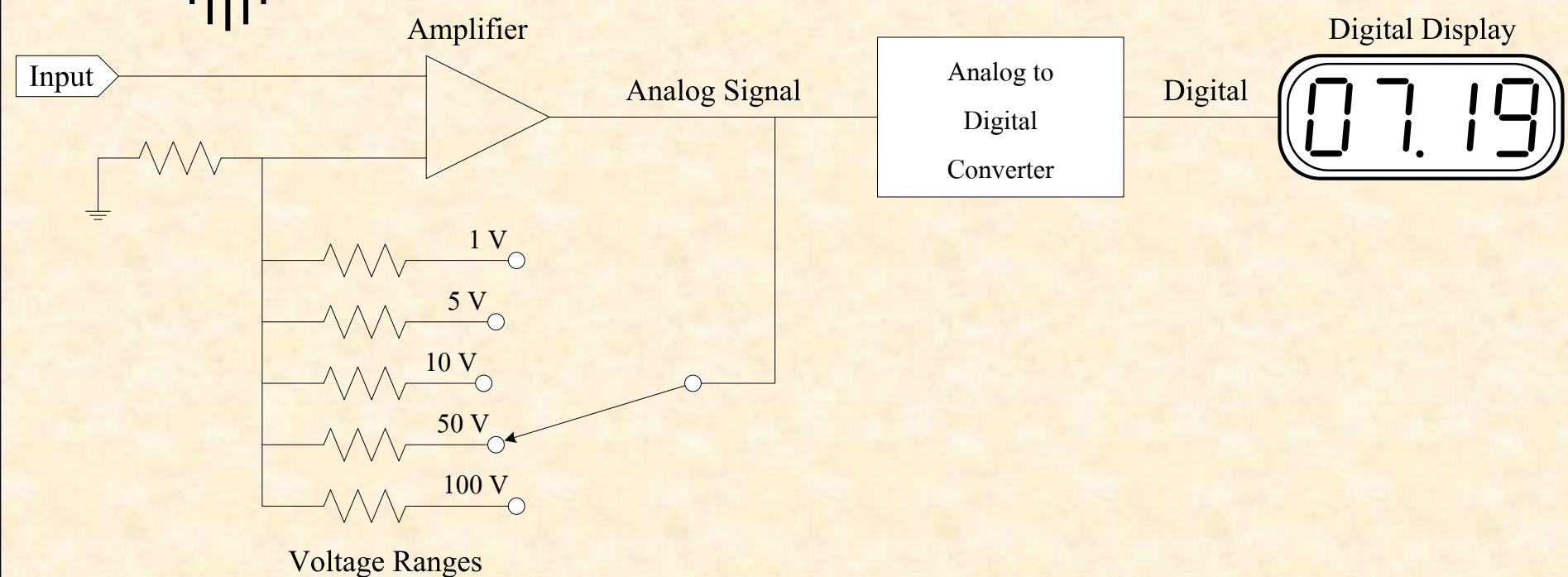
# Digital Signals



# Analog Voltmeter



# Digital Voltmeter



# Digital Electronic System

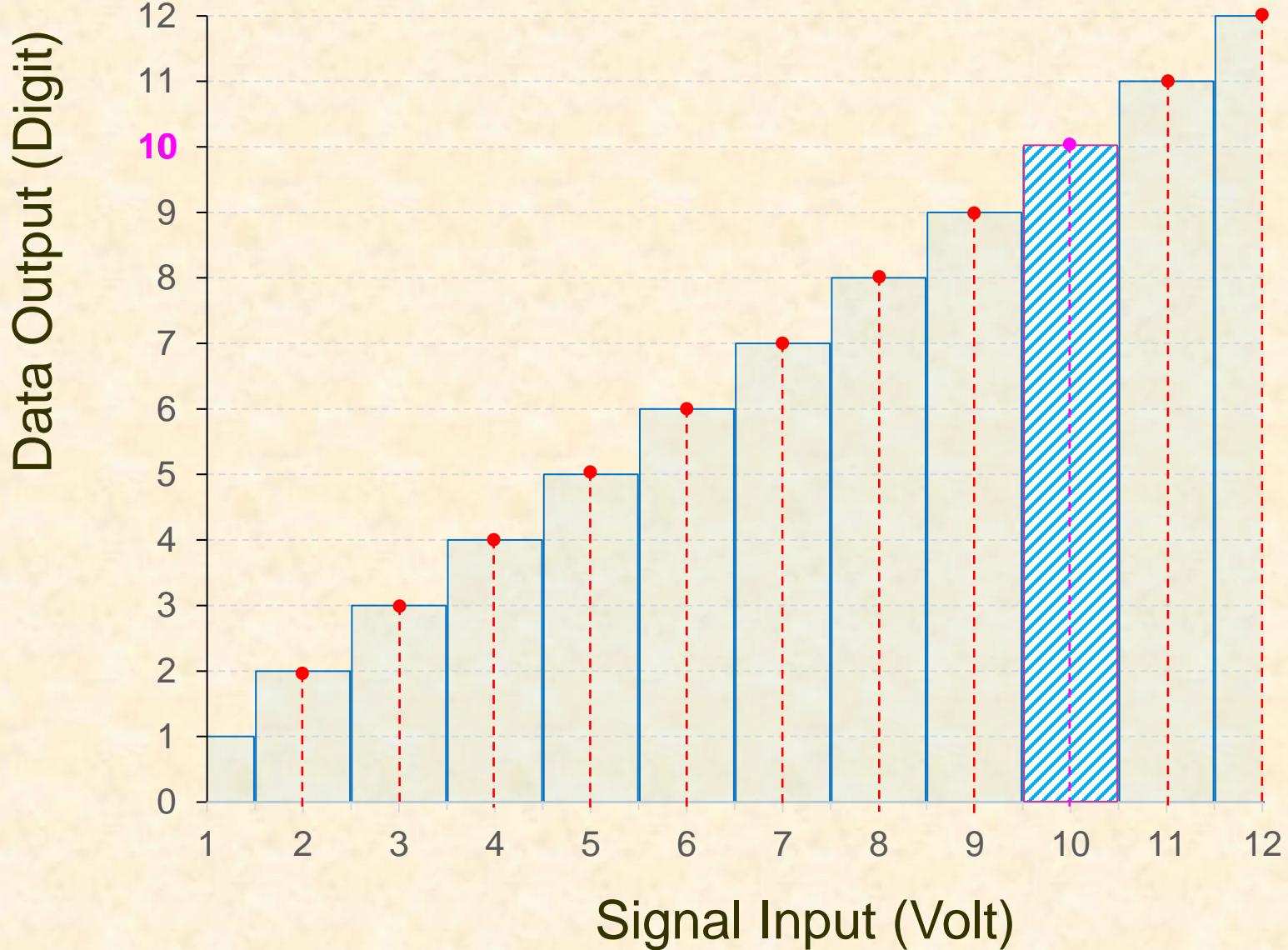
- Digital electronic system is an electronic system in which operations are performed with a certain value for a level of voltage.
- e.g. using the value of 10, to represent the voltage of 9.5-10.4 Volt for the system resolution of 1 Volt/unit.

In electrical system, the signal is created using a switch to make ON/OFF status:

➤ two status  $\Rightarrow$  two numbers  $\Rightarrow$  **Binary System**.

whereas OFF status = 0, and ON status = 1

- e.g. using the **Binary of 1010**, to represent the voltage of 9.5-10.4 Volt for the resolution of 1 Volt/unit system.



# Number Systems

- Number System used in the **human-life system** is **Decimal Number.**
- Number System used in the **computer system** is
  - **Binary Number** comprises with 0 and 1 (2 Numbers).
  - Another Number System derived from Binary Number is **Octal ( $2^3$ ) number** and **Hexadecimal ( $2^4$ ) number.**

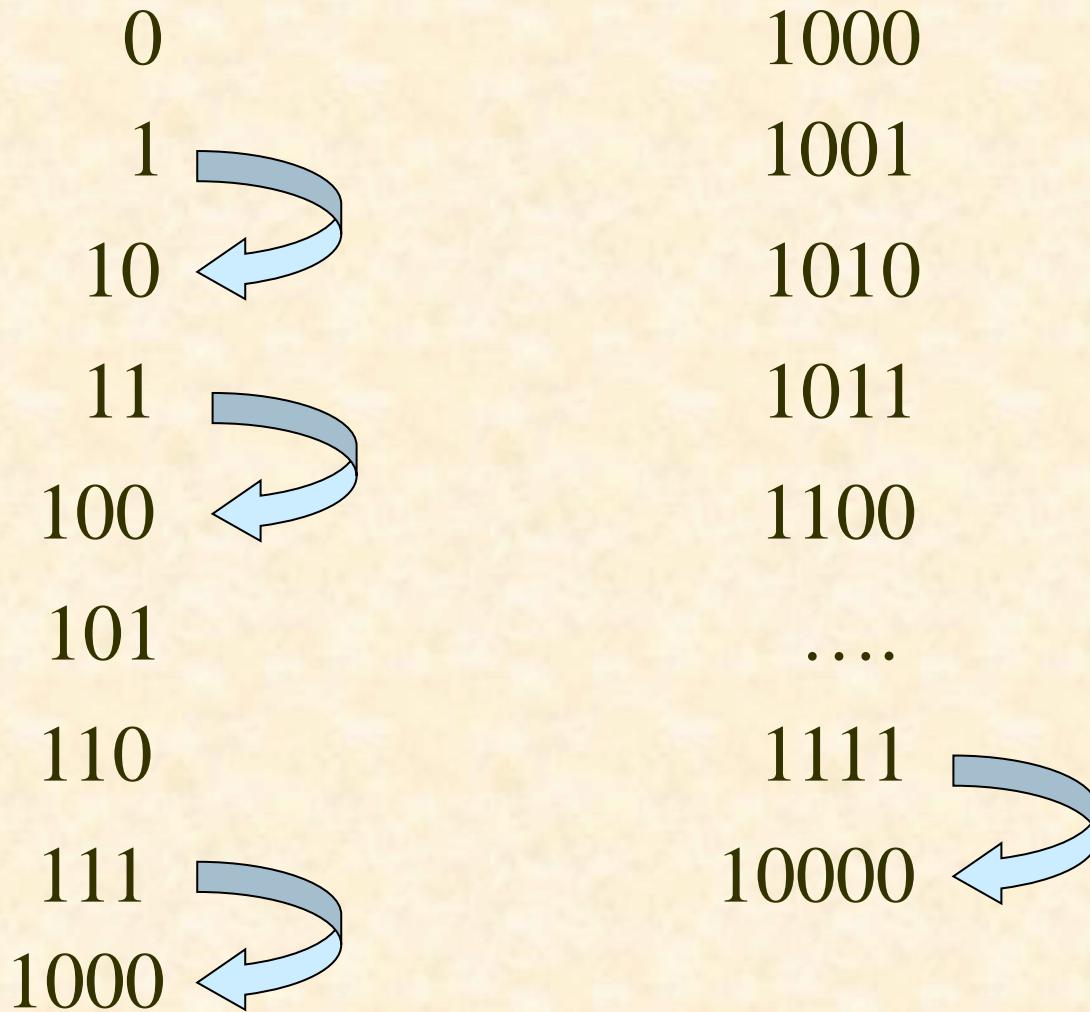
# Number Systems

- Decimal                      Base 10
- Binary                        Base 2
- Octal                        Base 8
- Hexadecimal                Base 16

## Binary Number Systems

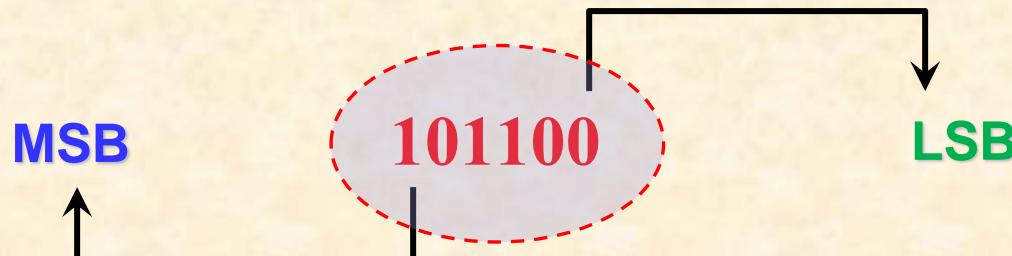
- **Binary Number** comprises with 2 numbers of 0 and 1
- e.g. 0, 1001, 111001, 111110001 etc.

# Binary Number System: Counting.



# Binary Number System

## Most Significant Bit



## Least Significant Bit

For the Number of

$N$  bits

The system has overall the NUMBER of

$2^N$

The NUMBER starts from 0 to the possible Maximum of  $2^N - 1$

# Number-Base Conversions

- The number with base  $X$ , composed of  $n$  bits on the left and  $m$  bits on the right of the decimal point, can be expressed as:

$$a_n X^{n-1} + a_{n-1} X^{n-2} + a_{n-2} X^{n-3} + \dots + a_2 X^1 + a_1 X^0 + b_1 X^{-1} + b_2 X^{-2} + b_3 X^{-3} + \dots + b_m X^{-m}$$

- where  $a_k$  is the number at the  $k^{\text{th}}$  bit
  - The number  $a_n$  is the most significant bit (MSB)
  - and  $b_m$  is the least significant bit (LSB)

- The Decimal Number

$$10^{n-1} \quad \dots \quad 10^2 \quad 10^1 \quad 10^0$$

- The Binary Number

$$2^{n-1} \quad \dots \quad 2^2 \quad 2^1 \quad 2^0$$

**MSB**

**LSB**

# Number-Base Conversions

**395.4<sub>10</sub>**

3            9            5            4

$$(3 \times 10^2) + (9 \times 10^1) + (5 \times 10^0) + (4 \times 10^{-1}) = 300 + 90 + 5 + 0.4 = 395.4_{10}$$

**101.10<sub>2</sub>**

1            0            1       .       1            0

$$(1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) + (1 \times 2^{-1}) + (0 \times 2^{-2}) = 4 + 0 + 1 + 0.5 + 0 = 5.5_{10}$$

# Binary to Decimal Number Conversions

**Example:** Convert  $1000111_2$  to the Decimal Number

1	0	0	0	1	1	1
×	×	×	×	×	×	×
$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
<b>64</b>	<b>32</b>	<b>16</b>	<b>8</b>	<b>4</b>	<b>2</b>	<b>1</b>

$$(1 \times 64) + (0 \times 32) + (0 \times 16) + (0 \times 8) + (1 \times 4) + (1 \times 2) + (1 \times 1)$$

$$= 64 + 0 + 0 + 0 + 4 + 2 + 1$$

$$= 71$$

# Binary to Decimal Number Conversions

Example Convert  $1000.111_2$  to the Decimal Number

1	0	0	0	.	1	1	1
×	×	×	×	.	×	×	×
$2^3$	$2^2$	$2^1$	$2^0$	.	$2^{-1}$	$2^{-2}$	$2^{-3}$
<b>8</b>	<b>4</b>	<b>2</b>	<b>1</b>		<b>0.5</b>	<b>0.25</b>	<b>0.125</b>

$$\begin{aligned}(1 \times 8) + (0 \times 4) + (0 \times 2) + (0 \times 1) + (1 \times 0.5) + (1 \times 0.25) + (1 \times 0.125) \\= 8 + 0 + 0 + 0 + 0.5 + 0.25 + 0.125 \\= 8.875\end{aligned}$$

# Decimal to Binary Number Conversions

Example Convert  $45_{10}$  to the Binary Number.

0	1	0	1	1	0	1
$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
64	32	16	8	4	2	1

$$45 - 32 = 13$$

$$13 - 8 = 5$$

$$5 - 4 = 1$$

$$1 - 1 = 0$$

$$45_{10} = 101101_2$$

*Successive Approximation*

# Decimal to Binary Number Conversions

Example Convert  $45_{10}$  to the Binary Number

The second method

$$\begin{array}{r} 2 ) \underline{45} \\ 2 ) \underline{22} \text{ remain } 1 \quad \text{LSB} \\ 2 ) \underline{11} \text{ remain } 0 \\ 2 ) \underline{5} \text{ remain } 1 \\ 2 ) \underline{2} \text{ remain } 1 \\ \textcolor{red}{1} \text{ remain } 0 \\ \text{MSB} \end{array}$$

↑

$$45_{10} = \textcolor{red}{1}01101_2$$

*Double Dabble for INTEGER*

- *Divided by 2 until getting point zero;*
- *Collect the Remainders.*

# Decimal to Binary Number Conversions

Example Convert  $0.75_{10}$  to the Binary Number.

$$\begin{array}{rcl} 0.75 \times 2 & = & 1.5 \\ 0.5 \times 2 & = & 1.0 \end{array}$$

MSB  
  
LSB

$$0.75_{10} = 0.\textcolor{red}{1}1_2$$

*Double Dabble for FRACTION*

- Multiplied by 2 until getting point zero;
- Collect the Front Numbers

# Decimal to Binary Number Conversions

**Example:** Convert  $41.53125_{10}$  to the Binary Number.

**Solution:** Find the Integer Part:

$$\begin{array}{r} 2 ) \underline{41} \\ 2 ) \underline{20} \text{ remain } 1 \text{ LSB} \\ 2 ) \underline{10} \text{ remain } 0 \\ 2 ) \underline{5} \text{ remain } 0 \\ 2 ) \underline{2} \text{ remain } 1 \\ 1 \text{ remain } 0 \\ \text{MSB } \end{array}$$


Then  $41_{10} = 101001_2$

# Decimal to Binary Number Conversions

Then convert the Fractional Part of  $0.53125_{10}$  to the Binary Number

**Method:**

$$0.53125 \times 2 = 1.06250$$

$$0.06250 \times 2 = 0.12500$$

$$0.12500 \times 2 = 0.25000$$

$$0.25000 \times 2 = 0.50000$$

$$0.50000 \times 2 = 1.00000$$

MSB

LSB

$$0.53125_{10} = 0.10001_2 \text{ (The Fractional Part)}$$

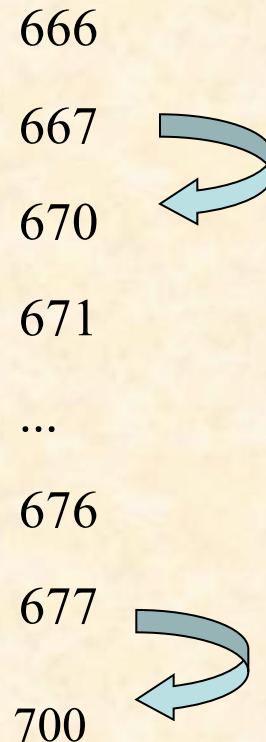
Then the complete answer:

$$41.53125_{10} = 101001.10001_2$$

# Octal Number System

- Octal Number System has 8 digits: 0, 1, 2, 3, 4, 5, 6 and 7.

Example Write the number base 8 starting from  $666_8$ .



# Octal Number System

Numbers in the Octal Number System:

0	1	2	3	4	5	6	7
10	11	12	13	14	15	16	17
20	21	22	23	24	25	26	27
30	31	32	33	34	35	36	37
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
<b>n0</b>	<b>n1</b>	<b>n2</b>	<b>n3</b>	<b>n4</b>	<b>n5</b>	<b>n6</b>	<b>n7</b>

# Binary to Octal Number Conversions

- Since  $8 = 2^3$
- A **3-bit** binary number can represent 8 values.
- A **3-bit** binary number is equivalent to 1 digit of the octal number.
- The transformation is done by grouping **3-bit** binary.
- From the right to left for the Integer,
- From the left to right for the Fraction,
- And convert each group to octal numbers

**Example:** convert  $1\ 0\ 1\ 1\ 1\ 1\ 0\ 1_2$  to an octal number

$$\begin{array}{r} \textcircled{1} \ 0\ 1\ 1\ 1\ 1\ 0\ 1 \\ \underbrace{\quad}_{2} \ \underbrace{\quad}_{7} \ \underbrace{\quad}_{5} \\ 1011101_2 = 275_8 \end{array}$$

# Binary to Octal Number Conversions

Example, convert  $10101.01111_2$  to an octal number

1 0 1 0 1 . 0 1 1 1 1 0  
2    5    .    3    6

Answer:  $10101.01111_2 = 25.36_8$

# Octal to Binary Number Conversions

To convert from an octal back to a binary number system by reversing the previous process.

Each digit of an octal number can be written as a group of **3-bit binary**.

**Example:** convert  $3062_8$  to binary number.

It can be done by writing:

The diagram shows the conversion of the octal number  $3062_8$  to its binary equivalent. Above the binary digits, the octal digits  $3$ ,  $0$ ,  $6$ , and  $2$  are listed. Four red arrows point from these octal digits down to the corresponding 3-bit binary groups below:  $011$ ,  $000$ ,  $110$ , and  $010$ . Together, these four groups form the binary number  $011\ 000\ 110\ 010$ .

3 0 6 2  
011 000 110 010

Then

$$3062_8 = 011\ 000\ 110\ 010_2$$

# Hexadecimal Number System

Hexadecimal Number System has 16 digits:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(10), B(11), C(12), D(13), E(14) and F(15)

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
n0	n1	n2	n3	n4	n5	n6	n7	n8	n9	nA	nB	nC	nD	nE	nF

# Hexadecimal counting

**Example:** Count the hexadecimal numbers from AE8 to B00.

AE8

AE9

AEA

.....

AEE

AEF

AF0

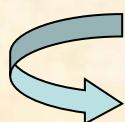
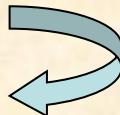
AF1

.....

AFE

AFF

B00



# Binary to Hexadecimal Number Conversions

- Since  $16 = 2^4$
- A **4-bit** binary number can represent 16 values.
- A **4-bit** binary number is equivalent to 1 digit of the hexadecimal number.
- The transformation is done by grouping **4-bit** binary;
- From the right to left for the Integer,
- From the left to right for the Fraction,
- And convert each group to hexadecimal numbers.

**Example:** Convert  $1\ 0\ 1\ 1\ 1\ 0\ 0\ 1_2$  to a hexadecimal number .

1 0 1 1 1 0 0 1  
  \underbrace{    }\_{B} \underbrace{    }\_{9}

Then

$$1\ 0\ 1\ 1\ 1\ 0\ 0\ 1_2 = B9_{16}$$

# Binary to Hexadecimal Number Conversions

**Example:** Convert  $1011.1001_2$  to an hexadecimal number

$$\begin{array}{c} 1011.1001 \\ \brace{1011} \quad \brace{1001} \\ B \quad . \quad 9 \end{array}$$

Then

$$1011.1001_2 = B.9_{16}$$

# Hexadecimal to Binary Number Conversion

To convert from a Hexadecimal back to a binary number system by reversing the previous process.

Each digit of a Hexadecimal number can be written as a group of **4-bit** binary.

**Example:** convert  $C3A6_{16}$  to binary number.

It can be done by writing:    C            3            A            6

1100    0011    1010    0110

Then

$$C3A6_{16} = 1100001110100110_2$$

# Hexadecimal to Binary Number Conversion

**Example:** convert  $C3.A6_{16}$  to binary number.

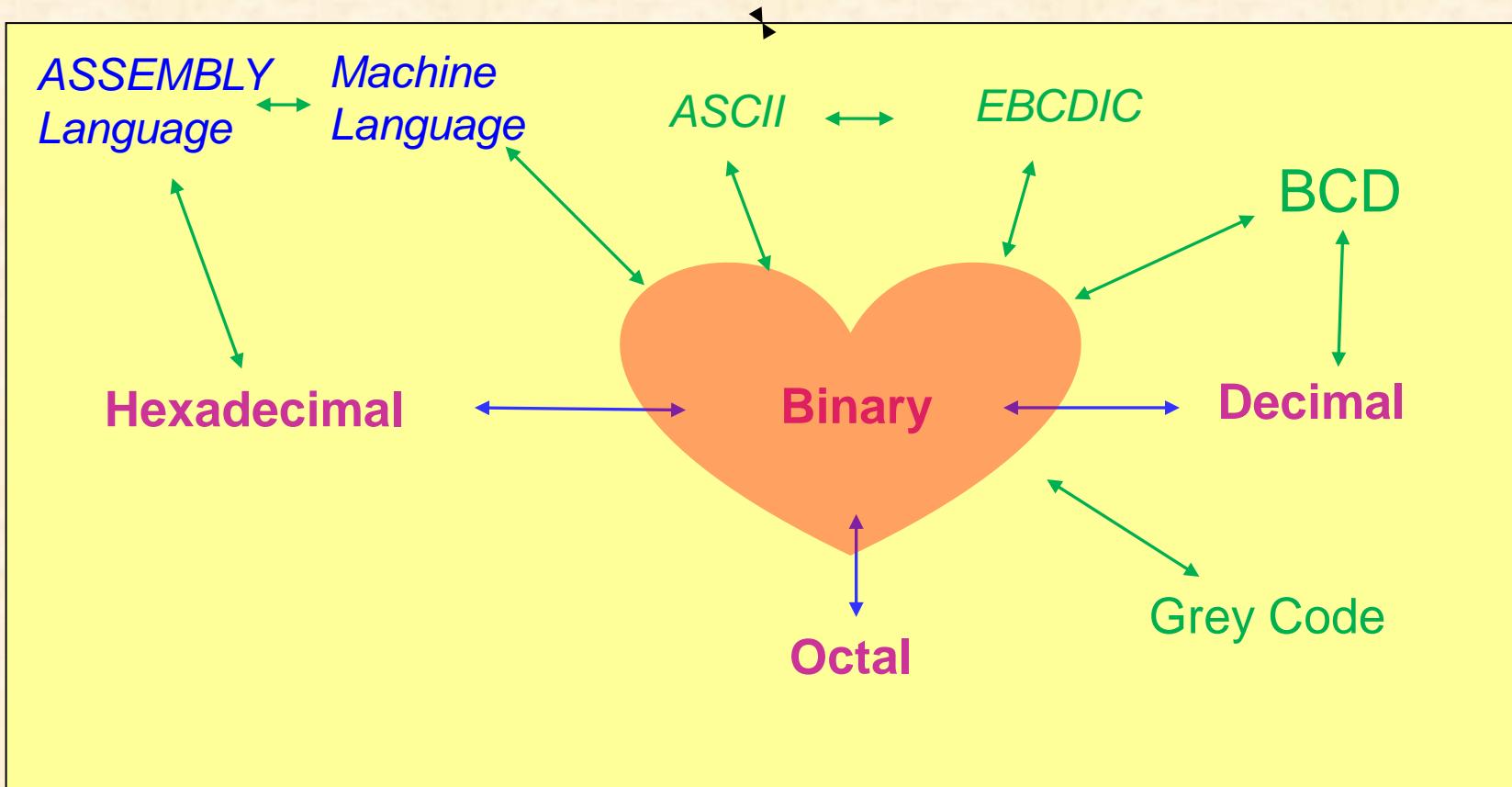
It can be done by writing:    C        3    .    A        6

1100    0011    .    1010    0110

Then

$$C3.A6_{16} = 11000011.10100110_2$$

# Number System and Applications



# **LOGIC GATE**

The gate is a basic logic circuit that has input(s) and output(s) as logic signal either 0 or 1.

The basic gates are followings:

- Inverter or NOT Gate
- AND Gate
- NAND Gate
- OR Gate
- NOR Gate
- Exclusive Or (XOR) Gate
- Exclusive Nor (XNOR) Gate

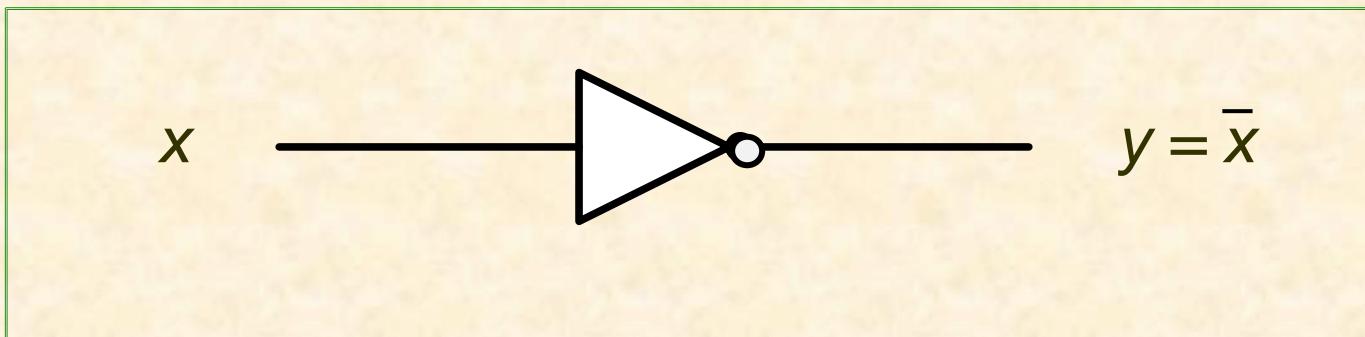
# Inverter / NOT Gate

The inverter is a gate that has 1 input and 1 output.

The circuit will give the logical output opposite to the logical input,

i.e. If input = 0, Output = 1

And if input = 1, Output = 0

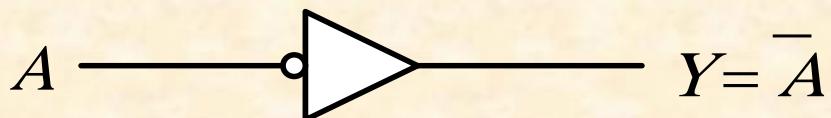
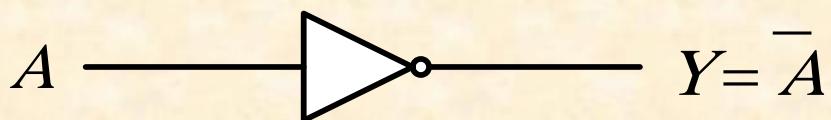


# Inverter / NOT Gate

Inverter / NOT Gate can be expressed by the **symbols**, and is characterized by the **truth table**.

## Symbols

$$Y = A'$$

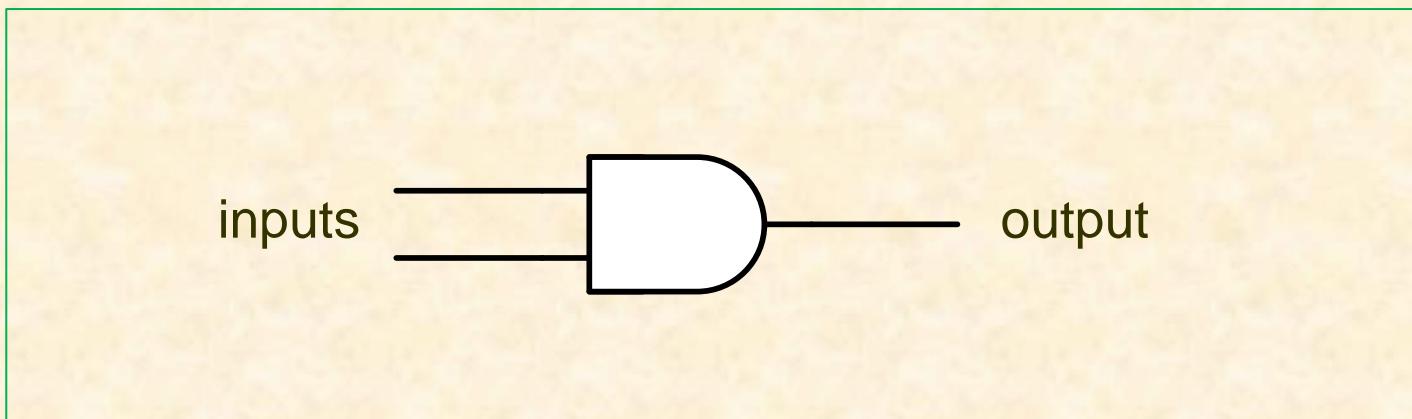


## Truth Table for the Inverter

Input	Output
$A$	$Y$
0	1
1	0

# AND Gate

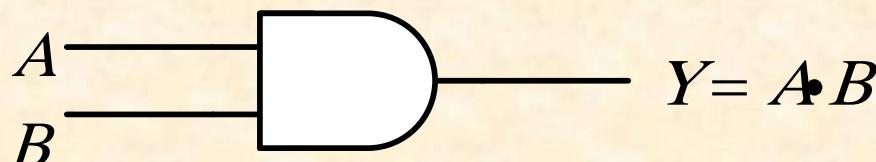
- **AND Gate** is a gate with 2 inputs or more, but 1 output.
- The circuit will give the output of 1 when every input is 1.
- In other cases, the output will be 0.



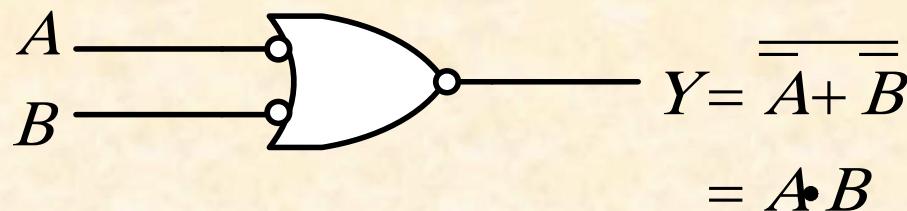
# AND Gate

AND Gate has the **symbol**, characterized by the **truth table** below.

Symbol



Equivalent Symbol

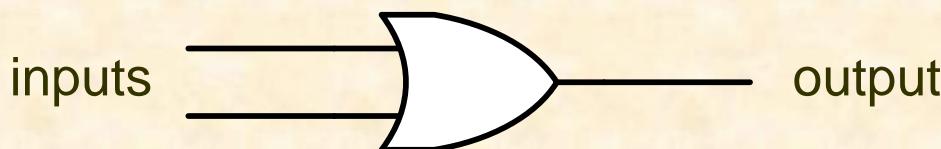


Truth Table for the AND gate

Input		Output
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

# OR Gate

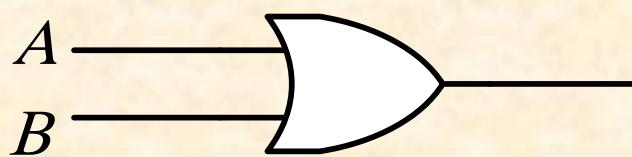
- **OR Gate** is a gate with 2 inputs or more, but 1 output.
- The circuit will give the output of 0 when every input is 0.
- In other cases, the output will be 1.



# OR Gate

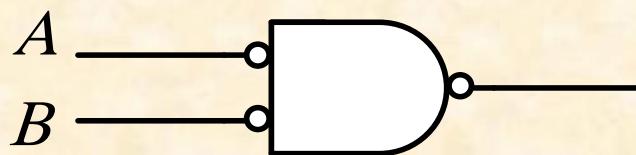
OR Gate has the **symbol**, characterized by the **truth table** below.

Symbol



$$Y = A + B$$

Equivalent Symbol



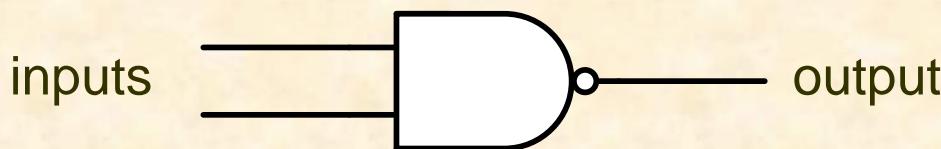
$$Y = \overline{\overline{A} \cdot \overline{B}} \\ = A + B$$

Truth Table for the OR gate

Input		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

# NAND Gate

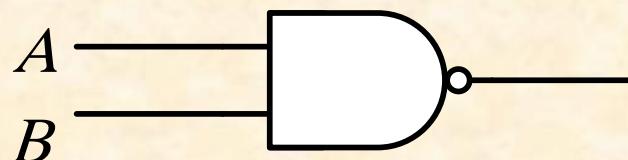
- **NAND Gate** is a gate with 2 inputs or more, but 1 output.
- The circuit will give the output of 0 when every input is 1.
- In other cases, the output will be 1.



# NAND Gate

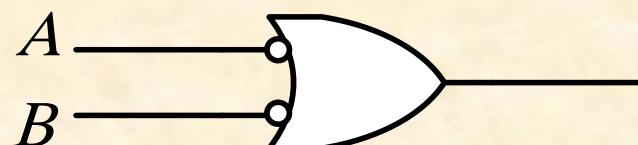
NAND Gate has the **symbol**, characterized by the **truth table** below.

Symbol



$$Y = \overline{A \bullet B}$$

Equivalent Symbol



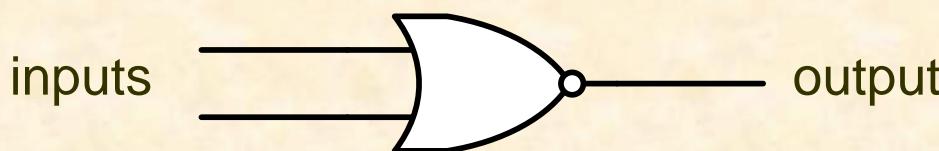
$$Y = \overline{A} + \overline{B} \\ = \overline{A \bullet B}$$

Truth Table for the NAND gate

Input		Output
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

# NOR Gate

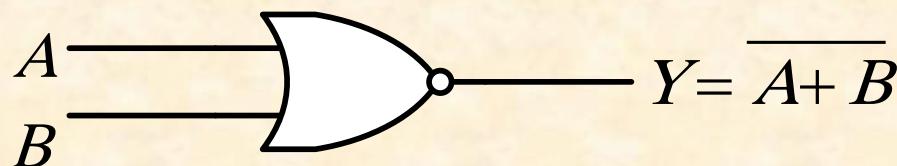
- **NOR Gate** is a gate with 2 inputs, or more, but 1 output.
- The circuit will give the output of 1 when every input is 0.
- In other cases, the output will be 0.



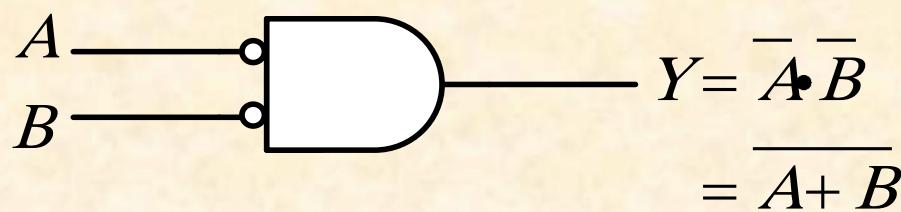
# NOR Gate

NOR Gate has the symbol, characterized by the truth table below.

Symbol



Equivalent Symbol

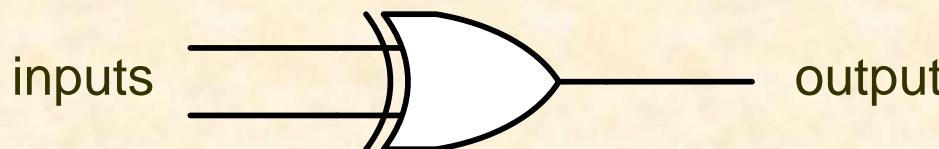


Truth Table for the NOR gate

Input		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

# Exclusive OR (XOR) Gate

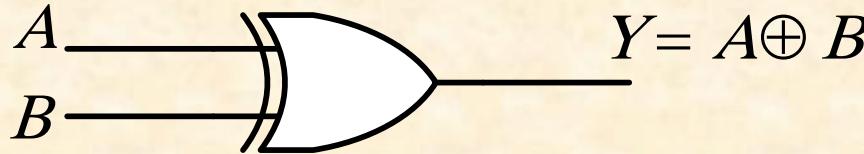
- **XOR Gate** is a gate with 2 inputs and 1 output.
- The circuit will give the output of 1 when the inputs have different logic.
- And the output is 0 when the inputs have same logic.



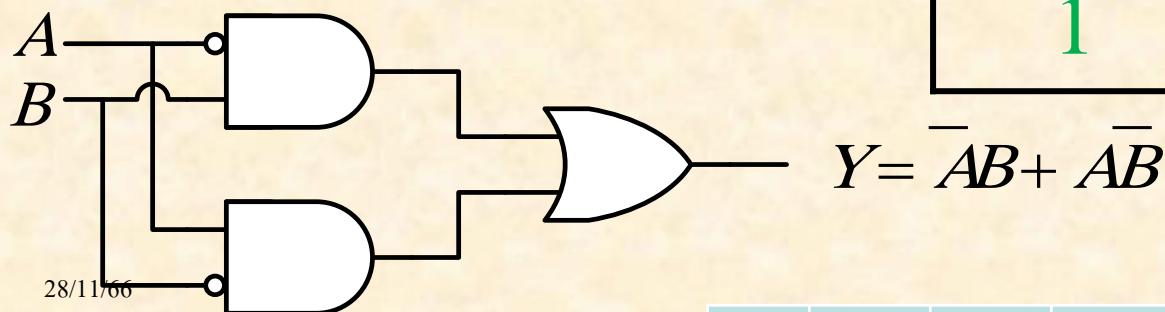
# Exclusive OR (XOR) Gate

XOR Gate has the symbol, characterized by the truth table below.

Symbol



Equivalent Circuit



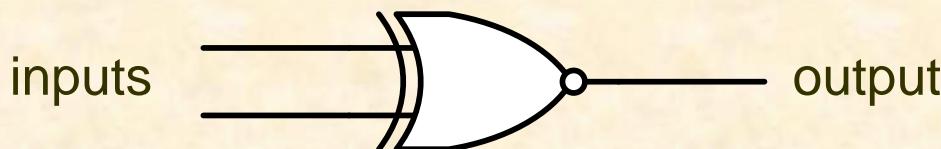
Truth Table for the XOR gate

Input		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

A	B	A'	B'	AB'	A'B	Y
0	0	1	1	0	0	0

# Exclusive NOR (XNOR) Gate

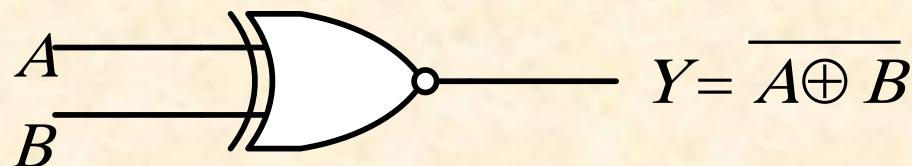
- **XNOR Gate** is a gate with 2 inputs and 1 output.
- The circuit will give the output of 1 when the inputs have same logic.
- And the output is 0 when the inputs have different logic.



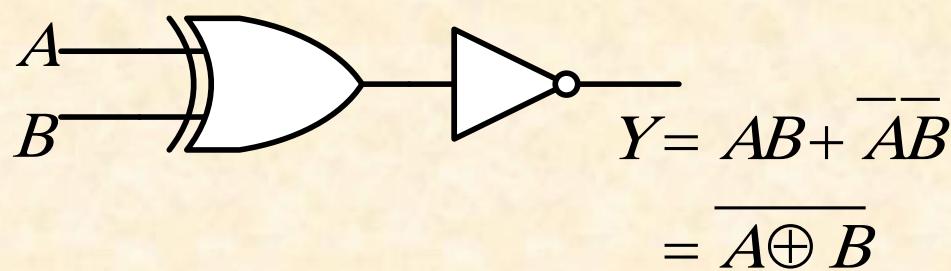
# Exclusive NOR (XOR) Gate

XNOR Gate has the symbol, characterized by the truth table below.

Symbol



Equivalent Circuit



Truth Table for the XNOR gate

Input		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1