Number Systems, logic gates

Prerequisites, digital basics

Analog and digital systems

- **Electronic circuits** – analog and digital
- **Analog circuits** – voltage and current vary continuously
- **Example** - output of audio amplifier -10v to +10v at any particular instant of time.
- **Examples of analog devices-** power supplies, signal generators, Frequency transmitters and receivers, electric motors and speed controllers and any analog instruments- those having pointers

- By contrast, a **digital circuit** is one in which the voltage levels assume a finite number of distinct values.
- **Two discrete voltage levels-** All modern digital circuits
- Each voltage level- **range of voltages**
- Digital circuits – **switching circuits**

- Switching circuits – combinational switching
- or sequential switching
- In combinational switching o/p depends only on the **present inputs**, whereas in sequential switching circuits **present inputs** as well as present state of the circuit, that is on the **past inputs** also.

- Digital circuits – logic circuits
- Logic circuit – **obeys certain set of logic rules**
- The manner in which a logic circuit responds to an input is referred to as the circuit’s logic.
- Digital systems have a number of advantages over analog systems.
- Many tasks formally done by analog systems are now being performed digitally.

Advantages of digital technology

- The chief reasons for the shift to digital technology are as follows:
- Digital systems are easier to design: HIGH and LOW levels
- Information storage is easy: semiconductor and magnetic memories
- Accuracy and precision are greater: more accurate, precise, simple and analog complex
- Digital systems are more versatile: easy to design, operation is controlled by program, modify program any time, analog-costly, complex
- Digital circuits are less affected by noise :unwanted electrical signals-noise, unavoidable in any system,
- More digital circuitry can be fabricated on IC chips
- Reliability is more

Limitations of digital techniques

- Even thought the digital techniques have a number of advantages, they have only one major drawback,
- *THE REAL WORLD IS ANALOG*
- Most of the physical quantities are analog in nature
- When these quantities are processed and expressed digitally

- We are really making a digital approximation to an inherently analog quantity.
- Conversion from analog to digital, processing using digital techniques
- because of these conversions, the processing time increases and the system becomes more complex.

Representation of signed numbers

- In computing, signed number representations are required to encode negative numbers in binary number systems.
- In mathematics, **negative numbers in any base are represented by prefixing them with a minus sign ("−")**.

- The Most Significant Bit MSB of signed binary numbers is used to indicate the sign of the numbers.
- Hence, it is also called as **sign bit**. The positive sign is represented by placing ‘0’ in the sign bit. Similarly, the negative sign is represented by placing ‘1’ in the sign bit.

- If the signed binary number contains ‘N’ bits, then N–1 bits only represent the magnitude of the number since one bit MSB is reserved for representing sign of the number.
- There are three **types of representations** for signed binary numbers
- Sign-Magnitude form
- 1’s complement form
- 2’s complement form

- **Example**
- Consider the **positive decimal number +108**. The binary equivalent of magnitude of this number is 1101100.
- These 7 bits represent the magnitude of the number 108.
- Since it is positive number, consider the sign bit as zero, which is placed on left most side of magnitude.

- $+108_{10} = 01101100_{2.}$

- Therefore, the **signed binary representation** of positive decimal number +108 is $\mathbf{01101100}$.
- So, the same representation is valid in sign-magnitude form, 1’s complement form and 2’s complement form for positive decimal number +108.

- **Sign-Magnitude form**
- In sign-magnitude form, the MSB is used for representing **sign** of the number and the remaining bits represent the **magnitude** of the number.
- So, just include sign bit at the left most side of unsigned binary number.
- This representation is similar to the signed decimal numbers representation.

- **Example**
- Consider the **negative decimal number -108**.
- The magnitude of this number is 108.
- We know the unsigned binary representation of 108 is 1101100.
- It is having 7 bits.
- All these bits represent the magnitude.
- Since the given number is negative, consider the sign bit as one, which is placed on left most side of magnitude.

- $-108_{10} = 11101100_2$
- Therefore, the sign-magnitude representation of -108 is **11101100**.

- **1’s complement form**
- The 1’s complement of a number is obtained by **complementing all the bits** of signed binary number.
- So, 1’s complement of positive number gives a negative number.
- Similarly, 1’s complement of negative number gives a positive number.
- That means, if you perform two times 1’s complement of a binary number including sign bit, then you will get the original signed binary number.

- **Example**
- Consider the negative decimal number -108.
- The magnitude of this number is 108.
- We know the signed binary representation of 108 is 01101100.
- It is having 8 bits.
- The MSB of this number is zero, which indicates positive number.
- Complement of zero is one and vice-versa.
- So, replace zeros by ones and ones by zeros in order to get the negative number.

- $-108_{10} = 10010011_2$
- Therefore, the **1's complement of** $108_{\mathbf{10}}$ is $10010011_{\mathbf{2}}$.

2’s complement form

- The 2’s complement of a binary number is obtained by **adding one to the 1’s complement** of signed binary number.
- So, 2’s complement of positive number gives a negative number.
- Similarly, 2’s complement of negative number gives a positive number.

- That means, if you perform two times 2’s complement of a binary number including sign bit, then you will get the original signed binary number.

- **Example**
- Consider the **negative decimal number -108**.
- We know the 1’s complement of **(108)$_{10}$** is **(10010011)$_2$**

- *2's compliment of* 108_{10} *= 1's compliment of* 108_{10} *+ 1.*
- = 10010011 + 1
- = 10010100
- Therefore, the **2's complement of** 108_{10} is 10010100_2.

Popular number systems

- Decimal Number system
- Binary Number system
- Octal Number system
- Hexadecimal Number system

Decimal Number System

- The **base** or radix of Decimal number system is **10**.
- So, the numbers ranging from **0 to 9** are used in this number system.
- The part of the number that lies to the left of the **decimal point** is known as integer part.
- Similarly, the part of the number that lies to the right of the decimal point is known as **fractional part**.

- In this number system, the successive positions to the left of the decimal point having weights of 10^0, 10^1, 10^2, 10^3 and so on.
- Similarly, the successive positions to the right of the decimal point having weights of 10^{-1}, 10^{-2}, 10^{-3} and so on.
- That means, each position has specific weight, which is **power of base 10**

- Example
- Consider the **decimal number 1358.246**.
- Integer part of this number is 1358 and fractional part of this number is 0.246.
- **Mathematically**, we can write it as
- $1358.246 = (1 \times 10^3) + (3 \times 10^2) + (5 \times 10^1) + (8 \times 10^0) + (2 \times 10^{-1}) + (4 \times 10^{-2}) + (6 \times 10^{-3})$

Binary Number System

- All digital circuits and systems use this binary number system.
- The **base** or radix of this number system is **2**.
- So, the numbers 0 and 1 are used in this number system.

- In this number system, the successive positions to the left of the binary point having weights of 2^0, 2^1, 2^2, 2^3 and so on.
- Similarly, the successive positions to the right of the binary point having weights of 2^{-1}, 2^{-2}, 2^{-3} and so on.
- That means, each position has specific weight, which is **power of base 2**.

- Example
- Consider the **binary number 1101.011**. Integer part of this number is 1101 and fractional part of this number is 0.011.
- **Mathematically**, we can write it as
- $1101.011 = (1 \times 2^{3}) + (1 \times 2^{2}) + (0 \times 2^{1}) + (1 \times 2^{0}) + (0 \times 2^{-1}) + (1 \times 2^{-2}) + (1 \times 2^{-3})$

Octal Number System

- The **base** or radix of octal number system is **8**. So, the numbers ranging from 0 to 7 are used in this number system.
- The part of the number that lies to the left of the **octal point** is known as integer part.
- Similarly, the part of the number that lies to the right of the octal point is known as fractional part.

- In this number system, the successive positions to the left of the octal point having weights of 8^0, 8^1, 8^2, 8^3 and so on.
- Similarly, the successive positions to the right of the octal point having weights of 8^{-1}, 8^{-2}, 8^{-3} and so on.
- That means, each position has specific weight, which is **power of base 8**.

- Example
- Consider the **octal number 1457.236**. Integer part of this number is 1457 and fractional part of this number is 0.236.
- **Mathematically**, we can write it as
- $1457.236 = (1 \times 8^3) + (4 \times 8^2) + (5 \times 8^1) + (7 \times 8^0) + (2 \times 8^{-1}) + (3 \times 8^{-2}) + (6 \times 8^{-3})$

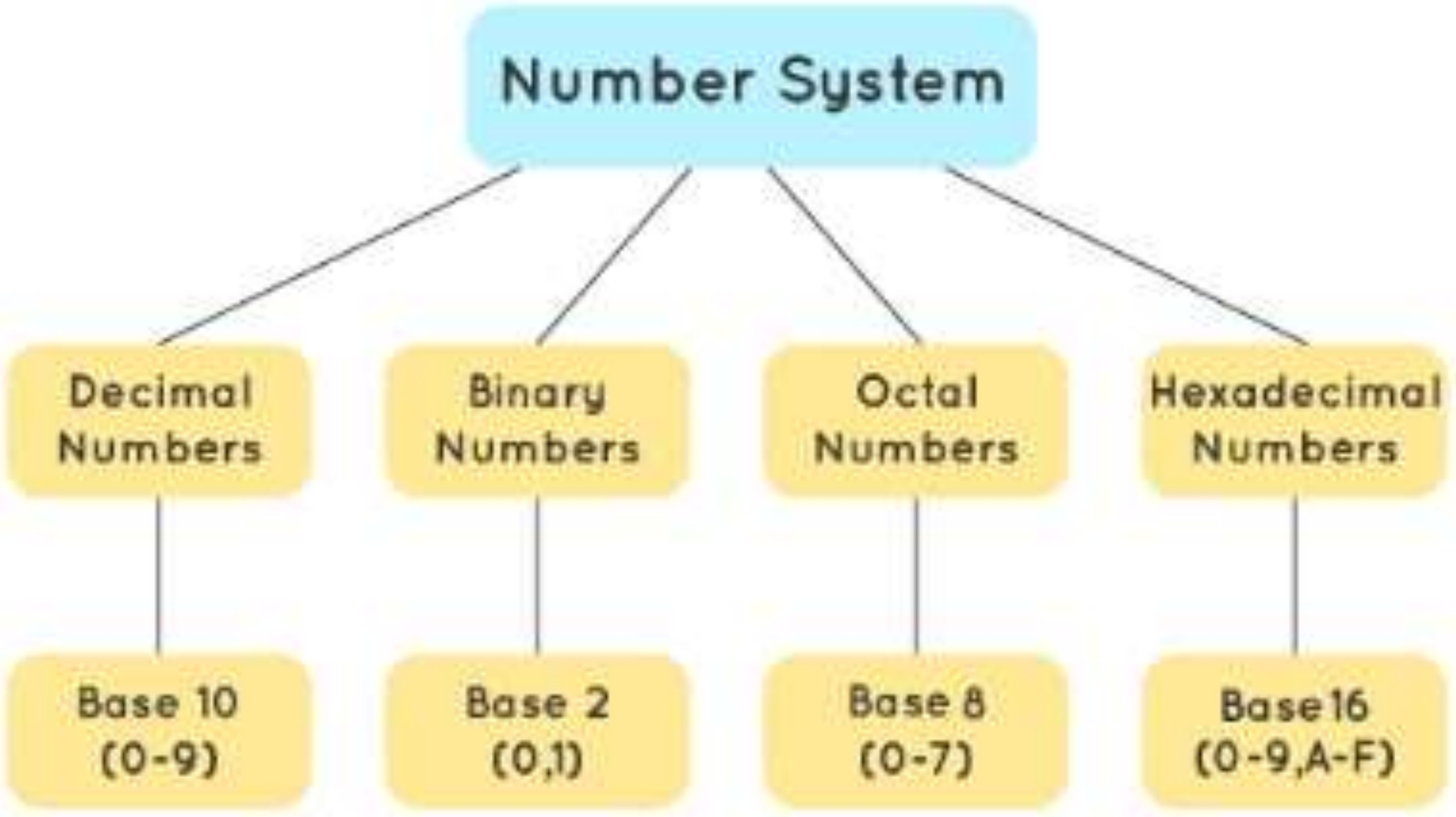
Hexadecimal Number System

- The **base** or radix of Hexa-decimal number system is **16**.
- So, the numbers ranging from 0 to 9 and the letters from A to F are used in this number system.
- The decimal equivalent of Hexa-decimal digits from A to F are 10 to 15.

- The part of the number, which lies to the left of the **hexadecimal point** is known as integer part.
- Similarly, the part of the number, which lies to the right of the Hexa-decimal point is known as **fractional part**.

- In this number system, the successive positions to the left of the Hexa-decimal point having weights of 16^0, 16^1, 16^2, 16^3 and so on.
- Similarly, the successive positions to the right of the Hexa-decimal point having weights of 16^{-1}, 16^{-2}, 16^{-3} and so on.
- That means, each position has specific weight, which is **power of base 16**.

- Example
- Consider the **Hexa-decimal number 1A05.2C4**. Integer part of this number is 1A05 and fractional part of this number is 0.2C4.
- **Mathematically**, we can write it as
- 1A05.2C4 = $(1 \times 16^{3}) + (10 \times 16^{2}) + (0 \times 16^{1}) + (5 \times 16^{0}) + (2 \times 16^{-1}) + (12 \times 16^{-2}) + (4 \times 16^{-3})$


Number System
Decimal Numbers
Binary Numbers
Octal Numbers
Hexadecimal Numbers
Base 10 (0-9)
Base 2 (0,1)
Base 8 (0-7)
Base 16 (0-9,A-F)

Decimal	Binary	Octal	Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Logic gates

- Digital electronic circuits operate with voltages of **two logic levels** namely Logic Low and Logic High.
- The range of voltages corresponding to Logic Low is represented with ‘0’.
- Similarly, the range of voltages corresponding to Logic High is represented with ‘1’.

- The basic digital electronic circuit that has one or more inputs and single output is known as **Logic gate**.
- Hence, the Logic gates are the building blocks of any digital system.
- We can classify these Logic gates into the following three categories.
- **Basic gates**
- **Universal gates**
- **Special gates**

Basic Gates

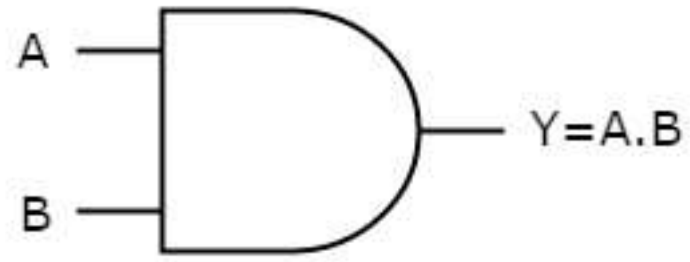
- AND gate
- An AND gate is a digital circuit that has two or more inputs and produces an output, which is the **logical AND** of all those inputs.
- It is optional to represent the **Logical AND** with the symbol ‘.’.

The following table shows the **truth table** of 2-input AND gate.

Truth table

A	B	Y = A.B
0	0	0
0	1	0
1	0	0
1	1	1

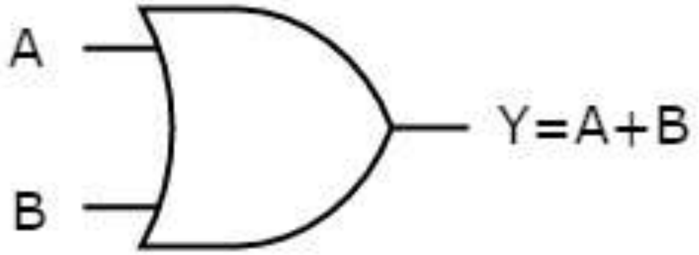
- Here A, B are the inputs and Y is the output of two input AND gate.
- If both inputs are ‘1’, then only the output, Y is ‘1’.
- For remaining combinations of inputs, the output, Y is ‘0’.
- The following figure shows the **symbol** of an AND gate, which is having two inputs A, B and one output, Y.


A
B
Y=A.B

- OR gate
- An OR gate is a digital circuit that has two or more inputs and produces an output, which is the logical OR of all those inputs.
- This **logical OR** is represented with the symbol '+'.
- The following table shows the **truth table** of 2-input OR gate.

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

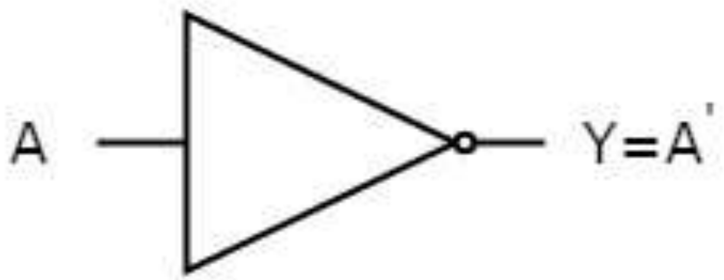
- Here A, B are the inputs and Y is the output of two input OR gate. If both inputs are ‘0’, then only the output, Y is ‘0’.
- For remaining combinations of inputs, the output, Y is ‘1’.
- The following figure shows the **symbol** of an OR gate, which is having two inputs A, B and one output, Y.


A
B
Y=A+B

- NOT gate
- A NOT gate is a digital circuit that has single input and single output.
- The output of NOT gate is the **logical inversion** of input. Hence, the NOT gate is also called as inverter.
- The following table shows the **truth table** of NOT gate.

A	Y = A’
0	1
1	0

- Here A and Y are the input and output of NOT gate respectively.
- If the input, A is ‘0’, then the output, Y is ‘1’.
- Similarly, if the input, A is ‘1’, then the output, Y is ‘0’.
- The following figure shows the **symbol** of NOT gate, which is having one input, A and one output, Y.

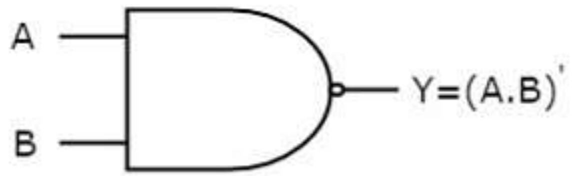

A
Y=A'

Universal gates

- NAND & NOR gates are called as **universal gates**.
- Because we can implement any Boolean function, which is in sum of products form by using NAND gates alone.
- Similarly, we can implement any Boolean function, which is in product of sums form by using NOR gates alone.

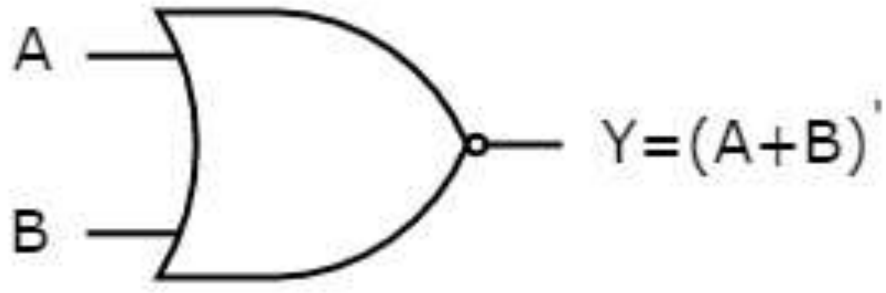
- NAND gate
- NAND gate is a digital circuit that has two or more inputs and produces an output, which is the **inversion of logical AND** of all those inputs.
- The following table shows the **truth table** of 2-input NAND gate.

A	B	Y = A.B’
0	0	1
0	1	1
1	0	1
1	1	0



- NOR gate
- NOR gate is a digital circuit that has two or more inputs and produces an output, which is the **inversion of logical OR** of all those inputs.
- The following table shows the **truth table** of 2-input NOR gate

A	B	Y = A+B’
0	0	1
0	1	0
1	0	0
1	1	0

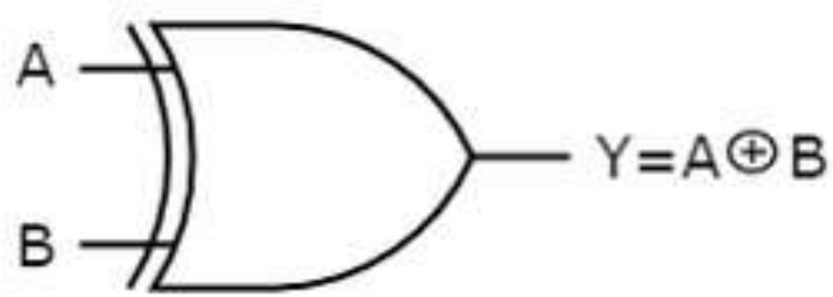


Special Gates

- Ex-OR & Ex-NOR gates are called as special gates.
- Because, these two gates are special cases of OR & NOR gates.

- Ex-OR gate
- The full form of Ex-OR gate is **Exclusive-OR** gate. Its function is same as that of OR gate except for some cases, when the inputs having even number of ones.
- The following table shows the **truth table** of 2-input Ex-OR gate.

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



- Ex-NOR gate
- The full form of Ex-NOR gate is **Exclusive-NOR** gate. Its function is same as that of NOR gate except for some cases, when the inputs having even number of ones.
- The following table shows the **truth table** of 2-input Ex-NOR gate.

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