Unit 5

Digital Electronics fundamentals: Difference between Analog and Digital Circuits—Number system—Binary—Hexadecimal —Conversions –Decimal to Binary—Hexadecimal and vice versa. Sequential and combinational circuits. Logic gates—Universal gates—Full Adder— Half Adder. MUX—DeMUX— Flip Flops (D—T—JK—& SR)

Basic Principle of Communication: Communication system Block diagram & working principles—Evolution of Communication systems

5.1 Digital Electronics Fundamentals

Introduction

Digital electronics is a type of electronics that deals with the digital systems which processes the data/information in the form of binary (0s and 1s) numbers, whereas analog electronics deals with the analog systems which processes the data/information in the form of continuous signals.

Characteristics of Digital systems

- Digital systems manipulate discrete elements of information.
- Discrete elements are nothing but the digits such as 10 decimal digits or 26 letters of alphabets and so on.
- Digital systems use physical quantities called signals to represent discrete elements.
- In digital systems, the signals have two discrete values and are therefore said to be binary.
- A signal in digital system represents one binary digit called a bit. The bit has a value either 0 or 1.

5.2 Difference between Analog Circuit and Digital Circuit

Analog Circuit

An analog circuit is a type electronic circuit that can process any analog signal or data and produce an output in analog form. Analog circuits are composed of **resistors, inductors and capacitors, etc**.

The type of signal which is a continuous function of time is known as **analog signal**.

All the real-world signals are the analog signals; therefore, the analog circuit do not require any conversion of the input signal i.e. the analog input signal can be directly fed to the analog circuit without any loss and it can be directly processed by the given analog circuit. Also, the output signal produced by the analog circuit is an analog signal.

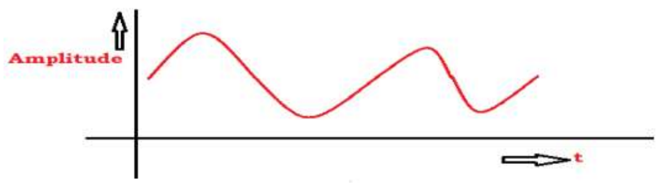


Figure 5.1(a): Analog Signal

Based on the circuit behaviour and the components used, the analog circuit can be of two types:

1. **Active Circuit**
   Ex: Amplifiers

2. **Passive Circuit**
   Ex: Low Pass Filter

The main drawback of the analog circuits is that the analog signals are very susceptible to the noise which may cause distortion of the signal waveform and causing the loss of information.

Digital Circuit

A digital circuit is an electronic circuit that processes digital signals. A signal that is a discrete function of time are known as **digital signals**.

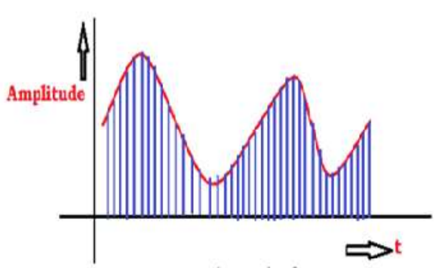


Figure 5.1(b1): Discrete Signal

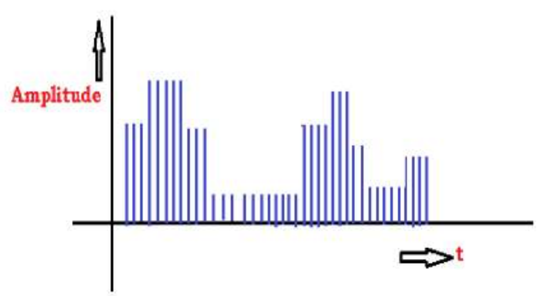


Figure 5.1(b2): Digital Signal

The basic building blocks of digital circuits are digital logic gates. The digital circuit can process only digital signals, but the real-world signals are of analog nature. Therefore, they need to be converted into digital signals using special electronic circuit known as **ADC (Analog to Digital Converter)**. The output of the digital circuits is also digital signals, which is required to be converted back into the analog signal.

There may be loss of information in the digital circuit during sampling process. The digital circuits can only be active circuits which means they require an additional power source to power the circuit.

The following table highlight the major differences between analog circuits and digital circuits

Table 5.1: Difference between Analog Circuit and Digital Circuit

Parameter	Analog Circuit	Digital Circuit
Definition	The electronic circuit which can process only analog signals is known as analog circuit.	The circuit which has ability to process only digital signals is known as digital circuit.
Input signal	The input signal to the analog circuit must be a continuous time signal or analog signal.	The input to the digital circuit is a discrete time signals or digital signal.
Output signal	Analog circuits produce output in the form of analog signals.	The output of the digital circuit is a digital signal.
Circuit components	The circuit components of the analog circuits are resistors, inductors, capacitors, etc.	The main circuit components of the digital circuits are logic gates.
Need of converters	The analog circuits can process the analog signals directly which present in the nature. Therefore, analog circuits do not require signal converters.	As the real world signals are analog, but the digital circuits can process signals only in digital form. Thus, digital circuits require signal converter, i.e. Analog to Digital Converter (ADC) and Digital to Analog Converter (DAC).

Parameter	Analog Circuit	Digital Circuit
Susceptibility to noise	The analog signals are more susceptible to noise.	Digital signals are immune to the noise.
Design	The analog circuits are complex to design because their circuit components need to be placed manually.	The designing of complicated digital circuits is relatively easier by using multiple software.
Flexibility	The implementation of analog circuit is not flexible.	The digital circuits offer more flexible implementation process.
Types	Analog circuits can be of two viz.: active circuit and passive circuit.	The digital circuits are of only one type named active circuit.
Processing speed	The processing speed of analog circuits is relatively low.	The digital circuits have higher processing speed than analog circuits.
Power consumption	The analog circuits consume more power.	The power consumed by the digital circuits is relatively less.
Accuracy & precision	The analog circuits are less accurate and precise.	The digital circuits are comparatively more accurate and precise.
Observational errors	In case of analog circuits, there may be an observational error in the output.	The digital circuits are free from observational errors in the output.
Signal transmission	In case of analog circuits, the signals are transmitted in the form of waves either wirelessly or with wires.	In the digital circuits, the signals can only be transmitted through wires in the digital form.

Parameter	Analog Circuit	Digital Circuit
Form of information storage	The analog circuits store the information in the form of waves.	Digital circuits store the information in binary form.
Logical operations	The analog circuit are not able to perform the logical operations efficiently.	Digital circuit performs logical operations efficiently.

5.3 Number System

Number systems are the technique to represent numbers in the computer system architecture, every value that you are saving or getting into/from computer memory has a defined number system.

Modern computers communicate and operate with binary numbers which use only the digits 0 &1. Basic number system used by humans is Decimal number system.

For Ex: Let us consider decimal number 18. This number is represented in binary as 10010.

In the digital computer, there are various types of number systems used for representing information.

1. Binary Number System
2. Decimal Number System
3. Hexadecimal Number System
4. Octal Number System

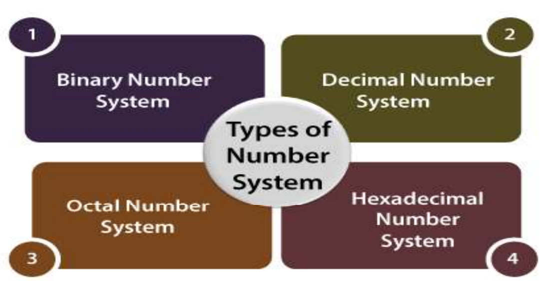


Figure 5.2: Types of Number System

Binary Number System

A Binary number system has only two digits that are **0 and 1**. Every number (value) represents with 0 and 1 in this number system. The base of binary number system is 2, because it has only two digits.

Octal Number System

Octal number system has only eight (8) digits from **0 to 7**. Every number (value) represents with 0,1,2,3,4,5,6 and 7 in this number system. The base of octal number system is 8, because it has only 8 digits.

Decimal Number System

Decimal number system has only ten (10) digits from **0 to 9**. Every number (value) represents with 0,1,2,3,4,5,6, 7,8 and 9 in this number system. The base of decimal number system is 10, because it has only 10 digits.

Hexadecimal Number System

A Hexadecimal number system has sixteen (16) alphanumeric values from **0 to 9** and **A to F**. Every number (value) represents with 0,1,2,3,4,5,6, 7, 8, 9, A, B, C, D, E and F in this number system. The base of hexadecimal number system is 16, because it has 16 alphanumeric values.

Table 5.2: Radix of Number System

Number system	Base(Radix)	Used digits	Example
Binary	2	0,1	$(11110000)_2$
Octal	8	0,1,2,3,4,5,6,7	$(360)_8$
Decimal	10	0,1,2,3,4,5,6,7,8,9	$(240)_{10}$
Hexadecimal	16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F	$(F0)_{16}$

Note - Here A is 10, B is 11, C is 12, D is 14, E is 15 and F is 16.

5.4 Number Base Conversion

In our previous section, we learned different types of number systems such as binary, decimal, octal, and hexadecimal. In this part of the tutorial, we will learn how we can change a number from one number system to another number system.

As, we have four types of number systems so each one can be converted into the remaining three systems. There are the following conversions possible in Number System

1. Binary to other Number Systems.
2. Decimal to other Number Systems.
3. Octal to other Number Systems.
4. Hexadecimal to other Number Systems.

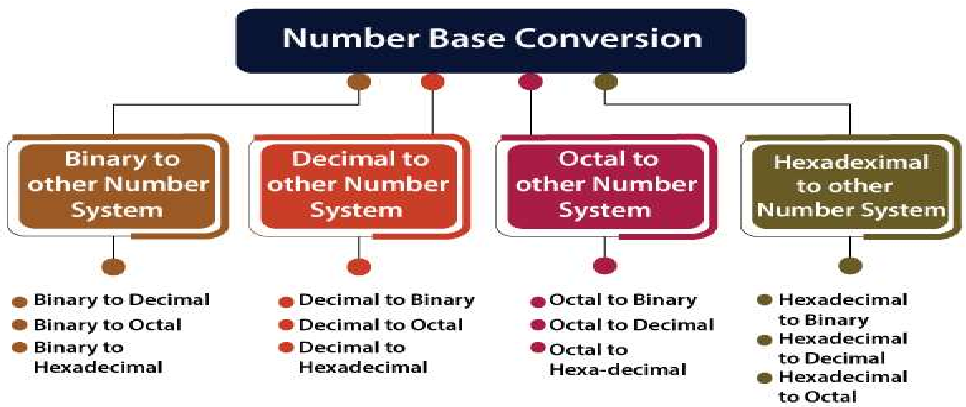


Figure 5.2: Flowchart of Number Base Conversion

5.4.1 Binary to other Number Systems

There are three conversions possible for binary number, i.e., binary to decimal, binary to octal, and binary to hexadecimal. The conversion process of a binary number to decimal differs from the remaining others. Let's take a detailed discussion on Binary Number System conversion.

- **Binary to Decimal Conversion**

The process of converting binary to decimal is quite simple. The process starts from multiplying the bits of binary number with its corresponding positional weights. And lastly, we add all those products.

Let's take an example to understand how the conversion is done from binary to decimal.

Example 1: $(10110.001)_2$

We multiplied each bit of $(10110.001)_2$ with its respective positional weight, and last we add the products of all the bits with its weight.

$$\mathbf{(10110.001)_2} = (1\times 2^4)+(0\times 2^3)+(1\times 2^2)+(1\times 2^1)+(0\times 2^0)+$$
$$(0\times 2^{-1})+(0\times 2^{-2})+(1\times 2^{-3})$$
$$\mathbf{(10110.001)_2} = (1\times 16)+(0\times 8)+(1\times 4)+(1\times 2)+(0\times 1)+$$
$$(0\times ½)+(0\times ¼)+(1\times ⅛)$$
$$\mathbf{(10110.001)_2} = 16+0+4+2+0+0+0+0.125$$
$$\mathbf{(10110.001)_2} = \mathbf{(22.125\ )_{10}}$$

- **Binary to Octal Conversion**

The base numbers of binary and octal are 2 and 8, respectively. In a binary number, the pair of three bits is equal to one octal digit. There are only two steps to convert a binary number into an octal number which are as follows:

1. In the first step, we have to make the pairs of three bits on both sides of the binary point. If there will be one or two bits left in a pair of three bits pair, we add the required number of zeros on extreme sides.

2. In the second step, we write the octal digits corresponding to each pair.

Example 1: $(111110101011.0011)_2$

1. Firstly, we make pairs of three bits on both sides of the binary point.

111 110 101 011.001 1

On the right side of the binary point, the last pair has only one bit. To make it a complete pair of three bits, we added two zeros on the extreme side.

111 110 101 011.001 100

2. Then, we wrote the octal digits, which correspond to each pair.

$(111110101011.0011)_2 = (7653.14)_8$

- **Binary to Hexadecimal Conversion**

The base numbers of binary and hexadecimal are 2 and 16, respectively. In a binary number, the pair of four bits is equal to one hexadecimal digit. There are also only two steps to convert a binary number into a hexadecimal number which are as follows:

1. In the first step, we have to make the pairs of four bits on both sides of the binary point. If there will be one, two, or three bits left in a pair of four bits pair, we add the required number of zeros on extreme sides.

2. In the second step, we write the hexadecimal digits corresponding to each pair.

Example 1: (10110101011.0011)$_2$

1. Firstly, we make pairs of four bits on both sides of the binary point.

111 1010 1011.0011

On the left side of the binary point, the first pair has three bits. To make it a complete pair of four bits, add one zero on the extreme side.

0111 1010 1011.0011

2. Then, we write the hexadecimal digits, which correspond to each pair.

(011110101011.0011)$_2$=(7AB.3)$_{16}$

5.4.2 Decimal to other Number System

The decimal number can be an integer or floating-point integer. When the decimal number is a floating-point integer, then we convert both part (integer and fractional) of the decimal number in the isolated form (individually). There are the following steps that are used to convert the decimal number into a similar number of any base **'r'**.

1. In the first step, we perform the division operation on integer and successive part with base **'r'**. We will list down all the remainders till the quotient is zero. Then we find out the remainders in reverse order for getting the integer part of the equivalent number of base **'r'**. In this, the least and most significant digits are denoted by the first and the last remainders.
2. In the next step, the multiplication operation is done with base **'r'** of the fractional and successive fraction. The carries are noted until the result is zero or when the required number of the equivalent digit is obtained. For getting the fractional part of the equivalent number of base **'r'**, the normal sequence of carrying is considered.

- **Decimal to Binary Conversion**

For converting decimal to binary, there are two steps required to perform, which are as follows:

1. In the first step, we perform the division operation on the integer and the successive quotient with the base of binary (2).

2. Next, we perform the multiplication on the integer and the successive quotient with the base of binary (2).

Example 1: $(152.25)_{10}$

Step 1:

Divide the number 152 and its successive quotients with base 2.

Operation	Quotient	Remainder
152/2	76	0 (LSB)
76/2	38	0
38/2	19	0
19/2	9	1
9/2	4	1
4/2	2	0
2/2	1	0
1/2	0	1(MSB)

$(152)_{10}=(10011000)_2$

Step 2:

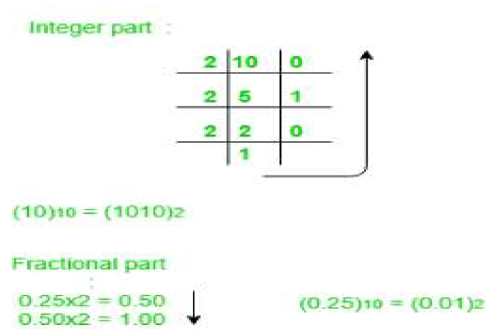
Now, perform the multiplication of 0.27 and successive fraction with base 2.

Operation	Result	carry
0.25×2	0.50	0
0.50×2	0	1

OR

Example

$(10.25)_{10}$



Note: Keep multiplying the fractional part with 2 until decimal part 0.00 is obtained. $(0.25)_{10} = (0.01)_2$

Answer: $(10.25)_{10} = (1010.01)_2$

- **Decimal to Octal Conversion**

For converting decimal to octal, there are two steps required to perform, which are as follows:

1. In the first step, we perform the division operation on the integer and the successive quotient with the base of octal(8).

2. Next, we perform the multiplication on the integer and the successive quotient with the base of octal(8).

Example 1: $(152.25)_{10}$

Step 1:

Divide the number 152 and its successive quotients with base 8.

Operation	Quotient	Remainder
152/8	19	0
19/8	2	3
2/8	0	2

$(152)_{10}=(230)_8$

Step 2:

Now perform the multiplication of 0.25 and successive fraction with base 8.

Operation	Result	carry
0.25×8	**0**	**2**

$(0.25)_{10}=(2)_8$

So, the octal number of the decimal number 152.25 is **230.2**

- **Decimal to hexadecimal conversion**

For converting decimal to hexadecimal, there are two steps required to perform, which are as follows:

1. In the first step, we perform the division operation on the integer and the successive quotient with the base of hexadecimal (16).

2. Next, we perform the multiplication on the integer and the successive quotient with the base of hexadecimal (16).

Example 1: $(152.25)_{10}$

Step 1:

Divide the number 152 and its successive quotients with base 8.

Operation	Quotient	Remainder
152/16	9	8
9/16	0	9

$(152)_{10}=(98)_{16}$

Step 2:

Now perform the multiplication of 0.25 and successive fraction with base 16.

Operation	Result	carry
0.25×16	0	4

5.4.3 Octal to other Number System

Like binary and decimal, the octal number can also be converted into other number systems. The process of converting octal to decimal differs from the remaining one. Let's start understanding how conversion is done.

- **Octal to Decimal Conversion**

The process of converting octal to decimal is the same as binary to decimal. The process starts from multiplying the digits of octal numbers with its corresponding positional weights. And lastly, we add all those products.

Let's take an example to understand how the conversion is done from octal to decimal.

Example 1: $(152.25)_8$

Step 1:

We multiply each digit of **152.25** with its respective positional weight, and last we add the products of all the bits with its weight.

$(152.25)_8$ $=(1\times8^2)+(5\times8^1)+(2\times8^0)+(2\times8^{-1})+(5\times8^{-2})$

$(152.25)_8$ $=64+40+2+(2\times1/8)+(5\times1/64)$

$(152.25)_8$ $=64+40+2+0.25+0.078125$

$(152.25)_8$ $=106.328125$

So, the decimal number of the octal number 152.25 is **106.328125**

- **Octal to Binary Conversion**

The process of converting octal to binary is the reverse process of binary to octal. We write the three bits binary code of each octal number digit.

Example 1: $(152.25)_8$

We write the three-bit binary digit for 1, 5, 2, and 5.

$(152.25)_8$ $=(001101010.010101)_2$

So, the binary number of the octal number 152.25 is **$(001101010.010101)_2$**

- **Octal to hexadecimal conversion**

For converting octal to hexadecimal, there are two steps required to perform, which are as follows:

1. In the first step, we will find the binary equivalent of number **25**.

2. Next, we have to make the pairs of four bits on both sides of the binary point. If there will be one, two, or three bits left in a pair of four bits pair, we add the required number of zeros on extreme sides and write the hexadecimal digits corresponding to each pair.

Example 1: $(152.25)_8$

Step 1:

We write the three-bit binary digit for 1, 5, 2, and 5.

$(152.25)_8$ $=(001101010.010101)_2$

So, the binary number of the octal number 152.25 is **$(001101010.010101)_2$**

Step 2:

1. Now, we make pairs of four bits on both sides of the binary point.

0 0110 1010.0101 01

On the left side of the binary point, the first pair has only one digit, and on the right side, the last pair has only two-digit. To make them complete pairs of four bits, add zeros on extreme sides.

0000 0110 1010.0101 0100

2. Now, we write the hexadecimal digits, which correspond to each pair.

$(0000\ 0110\ 1010.0101\ 0100)_2=(6A.54)_{16}$

5.4.4 Hexa-decimal to other Number System

Like binary, decimal, and octal, hexadecimal numbers can also be converted into other number systems. The process of converting hexadecimal to decimal differs from the remaining one. Let's start understanding how conversion is done.

- **Hexa-decimal to Decimal Conversion**

The process of converting hexadecimal to decimal is the same as binary to decimal. The process starts from multiplying the digits of hexadecimal numbers with its corresponding positional weights. And lastly, we add all those products.

Let's take an example to understand how the conversion is done from hexadecimal to decimal.

Example 1: $(152A.25)_{16}$

Step 1:

We multiply each digit of **152A.25** with its respective positional weight, and last we add the products of all the bits with its weight.

$(152A.25)_{16}$ $=(1\times16^3)+(5\times16^2)+(2\times16^1)+(A\times16^0)+(2\times16^{-1})+(5\times16^{-2})$

$(152A.25)_{16}$ $=(1\times4096)+(5\times256)+(2\times16)+(10\times1)+(2\times16^{-1})+(5\times16^{-2})$

$(152A.25)_{16}$ $=4096+1280+32+10+(2\times1/16)+(5\times1/256)$

$(152A.25)_{16}$ $=5418+0.125+0.125$

$(152A.25)_{16}$ $=5418.14453125$

So, the decimal number of the hexadecimal number 152A.25 is **5418.14453125**

- **Hexadecimal to Binary Conversion**

The process of converting hexadecimal to binary is the reverse process of binary to hexadecimal. We write the four bits binary code of each hexadecimal number digit.

Example 1: $(152A.25)_{16}$

We write the four-bit binary digit for 1, 5, A, 2, and 5.

$(152A.25)_{16}$=$(0001\ 0101\ 0010\ 1010.0010\ 0101)_2$

So, the binary number of the hexadecimal number 152.25 is **$(1010100101010.00100101)_2$**

- **Hexadecimal to Octal Conversion**

For converting hexadecimal to octal, there are two steps required to perform, which are as follows:

1. In the first step, we will find the binary equivalent of the hexadecimal number.

2. Next, we have to make the pairs of three bits on both sides of the binary point. If there will be one or two bits left in a pair of three bits pair, we add the required number of zeros on extreme sides and write the octal digits corresponding to each pair.

Example 1: $(152A.25)_{16}$

Step 1:

We write the four-bit binary digit for 1, 5, 2, A, and 5.

$(152A.25)_{16}$=$(0001\ 0101\ 0010\ 1010.0010\ 0101)_2$

So, the binary number of hexadecimal number 152A.25 is **$(0011010101010.010101)_2$**

Step 2:

3. Then, we make pairs of three bits on both sides of the binary point.

001 010 100 101 010.001 001 010

4. Then, we write the octal digit, which corresponds to each pair.

$(001010100101010.001001010)_2$=$(12452.112)_8$

So, the octal number of the hexadecimal number 152A.25 is **12452.112**

5.5 Boolean Algebra

Boolean algebra is the category of algebra in which the variable's values are the truth values, true and false, ordinarily denoted 1 and 0 respectively. It is used to analyse and simplify digital circuits or digital gates. It is also called **Binary Algebra** or **Logical Algebra.**

It has been fundamental in the development of digital electronics and is provided for in all modern programming languages.

The basic operations of Boolean algebra are as follows:

1. Conjunction or AND operation
2. Disjunction or OR operation
3. Negation or Not operation

Below is the table defining the symbols for all three basic operations.

Operator	Symbol
NOT	' (or) ¬
AND	. (or) ∧
OR	+ (or) ∨

Table 5.2: Basic Operations

Suppose A and B are two Boolean variables, then we can define the three operations as;

- A conjunction B or A AND B, satisfies A ∧ B = True, if A = B = True or else A ∧ B = False.
- A disjunction B or A OR B, satisfies A ∨ B = False, if A = B = False, else A ∨ B = True.
- Negation A or ¬A satisfies ¬A = False, if A = True and ¬A = True if A = False

5.5.1 Boolean Expression

A logical statement that results in a Boolean value, either be True or False, is a Boolean expression. Sometimes, synonyms are used to express the statement such as 'Yes' for 'True' and 'No' for 'False'. Also, 1 and 0 are used for digital circuits for True and False, respectively.

Boolean expressions are the statements that use logical operators, i.e., AND, OR, XOR and NOT. Thus, if we write X AND Y = True, then it is a Boolean expression.

5.5.2 Boolean Algebra Terminologies

Now, let us discuss the important terminologies covered in Boolean algebra.

- ✓ **Boolean algebra**: Boolean algebra is the branch of algebra that deals with logical operations and binary variables.

- ✓ **Boolean Variables:** A Boolean variable is defined as a variable or a symbol defined as a variable or a symbol, generally an alphabet that represents the logical quantities such as 0 or 1.

- ✓ **Boolean Function:** A Boolean function consists of binary variables, logical operators, constants such as 0 and 1, equal to the operator, and the parenthesis symbols.

- ✓ **Literal:** A literal may be a variable or a complement of a variable.

- ✓ **Complement**: The complement is defined as the inverse of a variable, which is represented by a bar over the variable.

- ✓ **Truth Table:** The truth table is a table that gives all the possible values of logical variables and the combination of the variables. It is possible to convert the Boolean equation into a truth table. The number of rows in the truth table should be equal to 2^n, where "n" is the number of variables in the equation. For example, if a Boolean equation consists of 3 variables, then the number of rows in the truth table is 8. (i.e.,) $2^3 = 8$.

5.5.3 Boolean Algebra Truth Table

Now, if we express the above operations in a truth table, we get;

A	B	$A \wedge B$	$A \vee B$
TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE
FALSE	TRUE	FALSE	TRUE
FALSE	FALSE	FALSE	FALSE

A	¬A
TRUE	FALSE
FALSE	TRUE

Table 5.3: Boolean algebra Truth Table

5.5.4 Boolean Algebra Rules

Following are the important rules used in Boolean algebra.

- Variable used can have only two values. Binary 1 for HIGH and Binary 0 for LOW.
- The complement of a variable is represented by an overbar $Thus,\ complement\ of\ variable\ B\ is\ represented\ as\ B.\ Thus\ if\ B = 0\ then\ B = 1$ $Thus\ if\ B = 0\ then\ \bar{B} = 1\ and\ B\ = 1\ then\ \bar{B} = 0.$
- OR-ing of the variables is represented by a plus (+) sign between them. For example, the OR-ing of A, B, and C is represented as A + B + C.
- Logical AND-ing of the two or more variables is represented by writing a dot between them, such as A.B.C. Sometimes, the dot may be omitted like ABC.

5.5.5 Laws of Boolean algebra

There are six types of Boolean algebra laws. They are:

- Commutative law
- Associative law
- Distributive law
- AND law
- OR law
- Inversion law

➢ **Commutative Law**

Any binary operation which satisfies the following expression is referred to as a commutative operation. Commutative law states that changing the sequence of the variables does not have any effect on the output of a logic circuit.

- A. B = B. A
- A + B = B + A

- **Associative Law**

It states that the order in which the logic operations are performed is irrelevant as their effect is the same.

- (A. B). C = A . (B . C)
- (A + B) + C = A + (B + C)

- **Distributive Law**

Distributive law states the following conditions:

- A. (B + C) = (A. B) + (A. C)
- A + (B. C) = (A + B) . (A + C)

- **AND Law**

These laws use the AND operation. Therefore, they are called AND laws.

- A .0 = 0
- A . 1 = A
- A. A = A
- $A.\bar{A}=0$

- **OR Law**

These laws use the OR operation. Therefore, they are called OR laws.

- A + 0 = A
- A + 1 = 1
- A + A = A
- $A+\bar{A}=1$

- **Inversion Law**

In Boolean algebra, the inversion law states that double inversion of variable results in the original variable itself.

- $\bar{\bar{A}}=A$

5.5.6 Boolean Algebra Theorems

The two important theorems which are extremely used in Boolean algebra are **De Morgan's First law and De Morgan's second law**. These two theorems are used to change the Boolean expression. This theorem basically helps to reduce the given Boolean expression in the simplified form. These two De Morgan's laws are used to change the expression from one form to another form. Now, let us discuss these two theorems in detail.

➢ **De Morgan's First Law:**

- De Morgan's First Law states that (A.B)' = A'+B'.

- The first law states that the complement of the product of the variables is equal to the sum of their individual complements of a variable.

- The truth table that shows the verification of De Morgan's First law is given as follows:

Table 5.4: De Morgan's First Law Truth Table

A	B	A'	B'	(A.B)'	A'+B'
0	0	1	1	1	1
0	1	1	0	1	1
1	0	0	1	1	1
1	1	0	0	0	0

The last two columns show that (A.B)' = A'+B'.
Hence, De Morgan's First Law is proved.

De Morgan's Second Law:

- De Morgan's Second law states that (A+B)' = A'. B'.

- The second law states that the complement of the sum of variables is equal to the product of their individual complements of a variable.

- The following truth table shows the proof for De Morgan’s second law.

Table 5.5: De Morgan’s Second Law Truth Table

A	B	A’	B’	(A+B)’	A’. B’
0	0	1	1	1	1
0	1	1	0	0	0
1	0	0	1	0	0
1	1	0	0	0	0

The last two columns show that (A+B)’ = A’. B’.

Hence, De Morgan’s second law is proved.

Examples

1. **Simplify the following expression:**

$c + \bar{B}\bar{C}$

Solution:

Given:

$C + \bar{B}\bar{C}$

According to Demorgan’s law, we can write the above expressions as

$C + (\bar{B} + \bar{C})$

From Commutative law:

$(C + \bar{C}) + \bar{B}$

From Complement law

$1 + \bar{B} = 1$

Therefore,

$C + \bar{B}\bar{C} = 1$

2. **Draw a truth table for A(B+D).**

A	B	D	B+D	A(B+D)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

5.6 Logic Gates

A **logic gate** is an electronic circuit designed by using electronic components like diodes, transistors, resistors, and more. As the name implies, a logic gate is designed to perform logical operations in digital systems like computers, communication systems, etc.

Therefore, we can say that the building blocks of a digital circuit are logic gates, which execute numerous logical operations that are required by any digital circuit. A logic gate can take two or more inputs but only produce one output. The output of a logic gate depends on the combination of inputs and the logical operation that the logic gate performs.

The logic gates can be classified into the following major types:

5.6.1 Basic Logic Gates

There are three basic logic gates:

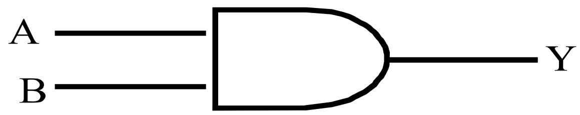
1. AND Gate
2. OR Gate
3. NOT Gate

- **AND Gate**:

  The AND gate performs logical multiplication, commonly known as AND function. The AND gate has two or more inputs and a single output. The output of an AND gate is

HIGH only when all the inputs are HIGH. Even if any one of the inputs is LOW, the output will be LOW. If a & b are input variables of an AND gate and c is its output, then Y=A.B.

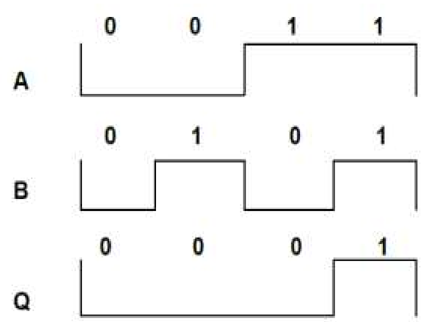
Logic Symbol



Truth Table

Input		Output
A	B	Y=A.B
0	0	0
0	1	0
1	0	0
1	1	1

Timing Diagram

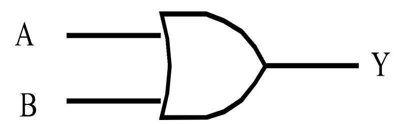


- **OR Gate**

The OR gate performs logical additions commonly known as OR function. The OR gate has two or more inputs and only one output. The operation of OR gate is such that a HIGH (1) on the output is produced when any of the input is HIGH. The output is LOW (0) only when all the inputs are LOW.

- If A & B are the input variables of an OR gate and c is its output, then A+B. similarly for more than two variables, the OR function can be expressed as Y=A+B+C.

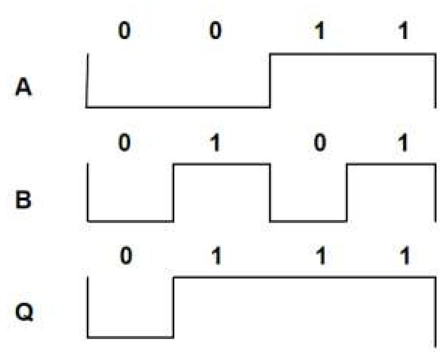
Logic Symbol



Truth Table

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

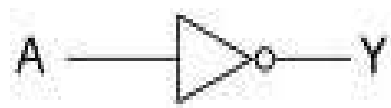
Timing Diagram



- **Not Gate (Inverter)**

The NOT gate performs the basic logical function called inversion or complementation. The purpose of this gate is to convert one logic level into the opposite logic level. It has one input and one output. When a HIGH level is applied to an inverter, a LOW level appears at the output and vice-versa.

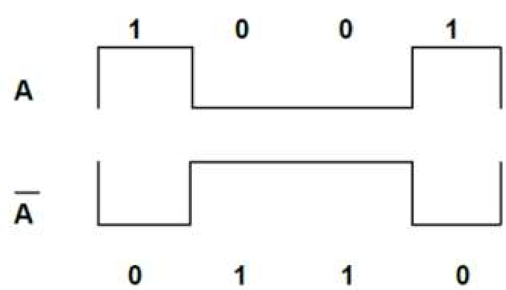
Logic Symbol



Truth Table

Input	Output
A	$\overline{A}$
0	1
1	0

Timing Diagram



5.6.2 Universal Logic Gates

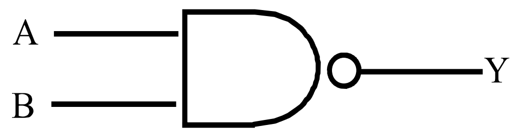
In digital electronics, the following two logic gates are considered as universal logic gates:

1. NAND Gate
2. NOR Gate

- **NAND Gate**

  The output of a NAND gate is LOW only when all inputs are HIGH and output of the NAND is HIGH if one or more inputs are LOW.

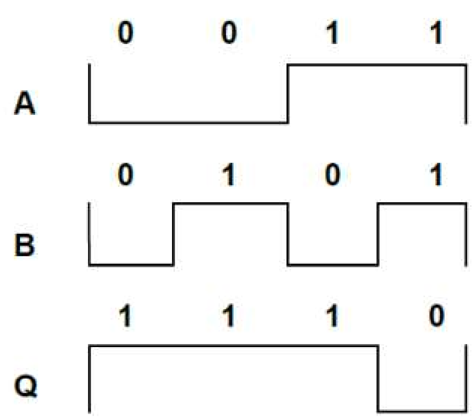
Logic Symbol



Truth Table

INPUT		OUTPUT
A	**B**	**Q=A + B**
0	0	0
0	1	1
1	0	1
1	1	1

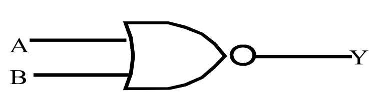
Timing Diagram



- **NOR Gate**

The output of the NOR gate is HIGH only when all the inputs are LOW.

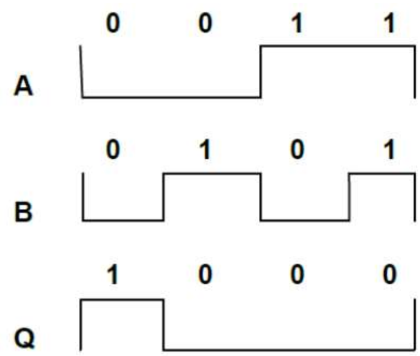
Logic Symbol



Truth Table

INPUT		OUTPUT
A	B	Q= $\overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

Timing Diagram



5.6.3 Derived Logic Gates

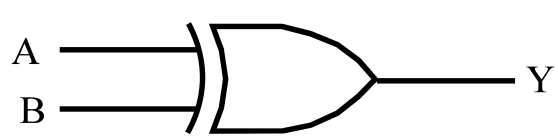
The following two are the derived logic gates used in digital systems:

1. XOR Gate
2. XNOR Gate

➢ **XOR Gate or Exclusive OR Gate**

In this gate output is HIGH only when any one of the input is HIGH. The circuit is also called as inequality comparator, because it produces output when two inputs are different. When both the inputs are high, then the output is low.

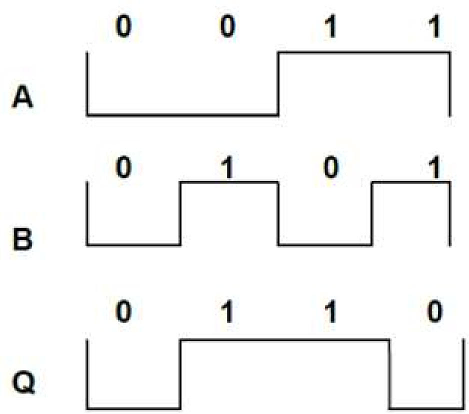
Logic Symbol



Truth Table

INPUT		OUTPUT
A	B	$Q = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

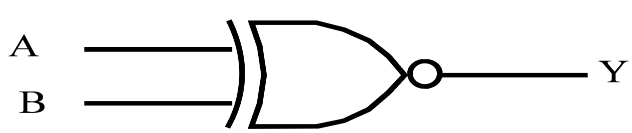
Timing Diagram



XNOR Gate or Exclusive NOR Gate:

An XNOR gate is a gate with two or more inputs and one output. XNOR operation is complimentary of XOR operation. i.e. The output of XNOR gate is High, when all the inputs are identical; otherwise, it is low.

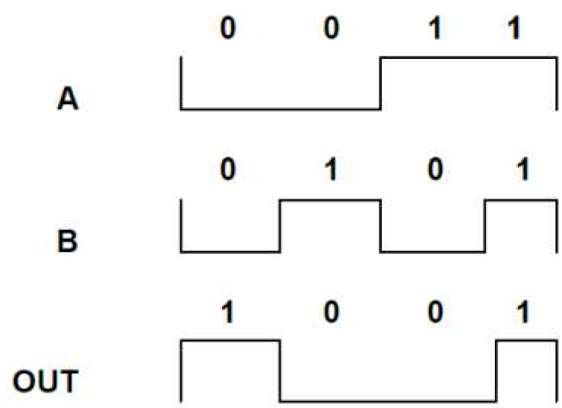
Logic Symbol



Truth Table

INPUT		OUTPUT
A	B	OUT =A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1

Timing Diagram



Logic gates are the fundamental building blocks of all digital circuits and devices like computers. Here are some key digital devices in which logic gates are utilized to design their circuits:

- Computers
- Microprocessors
- Microcontrollers
- Digital and smart watches
- Smartphones, etc.

5.7 Combinational Circuits

Combinational circuit is a circuit in which we combine the different gates in the circuit, for example encoder, decoder, multiplexer and demultiplexer. Some of the characteristics of combinational circuits are following

- The output of combinational circuit at any instant of time, depends only on the levels present at input terminals.
- The combinational circuit do not use any memory. The previous state of input does not have any effect on the present state of the circuit.
- A combinational circuit can have an n number of inputs and m number of outputs.

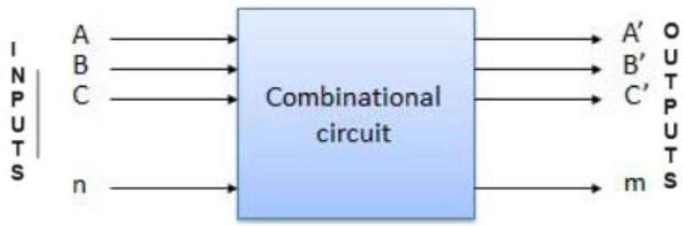


Figure 5.3: Block Diagram of Combinational Circuits

Adder

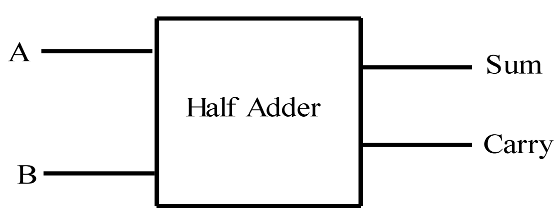
- The Basic operation in digital computer is binary addition. The circuit which perform the addition of binary bits are called as **Adder**.
- The logic circuit which performs the addition of two bit is called **Half adder** and three bit is called **Full adder**.

Rules of 2-Bit Addition

$0 + 0 = 0$
$0 + 1 = 1$
$1 + 0 = 1$
$1 + 1 = 10_2$

5.7.1 Half Adder

A combinational circuit which performs the arithmetic addition of two binary digits is called Half Adder. In the half adder circuit, there are two inputs, one is addend and augends and two outputs are **Sum** and **Carry**.



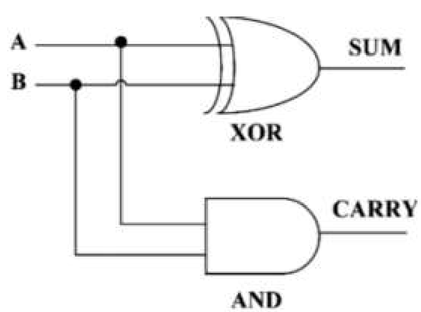
Truth Table

Inputs		Outputs	
A	**B**	**Carry**	**Sum**
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Logic Expression

$$\text{Sum, } S = A'B + AB' = A \oplus B$$
$$\text{Carry, } C = A \cdot B$$

Logic Diagram



Advantages of Half Adder

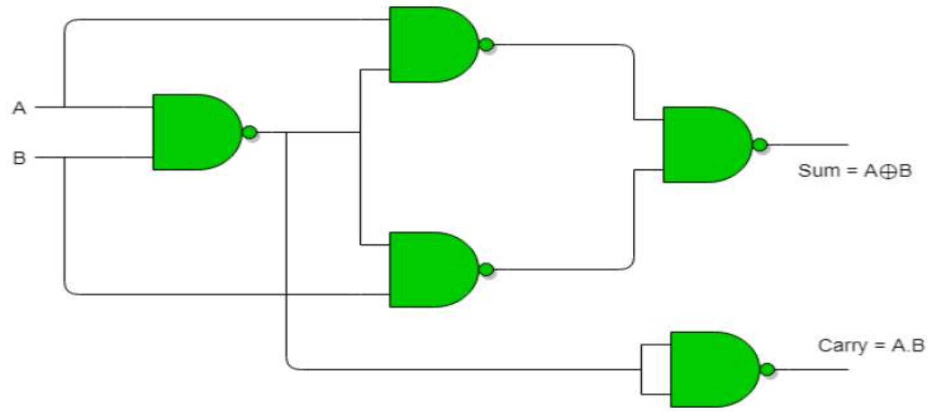
- The half adder and half subtractor circuits use only a few gates, which reduces the cost and power consumption compared to more complex circuits.
- Easy integration: The half adder and half subtractor can be easily integrated with other digital circuits and systems.

Limitations

- The main limitation of a half adder is that it can only add two single bits; it cannot handle a carry bit from the previous step.

- Half adders have no scope of adding the carry bit resulting from the addition of previous bits. This is a major drawback of half adders. This is because real time scenarios involve adding the multiple number of bits which cannot be accomplished using half adders.

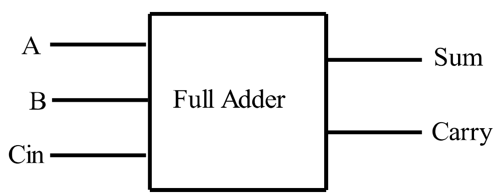
Realization of Half-Adder using NAND Gates



5.7.2 Full Adder

The full adder is a combinational circuit that performs the arithmetic sum of three input bits.

- It consists of three inputs and two outputs. Two of the inputs are variables, denoted by A and B, represent the two significant bits to be added. The third input C_{in} represents carry form the previous lower significant position.



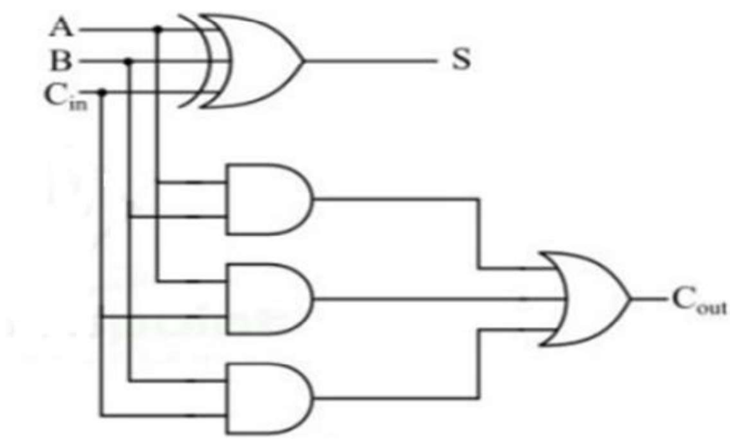
Truth Table

Inputs			Outputs	
A	B	Cin	Cout	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

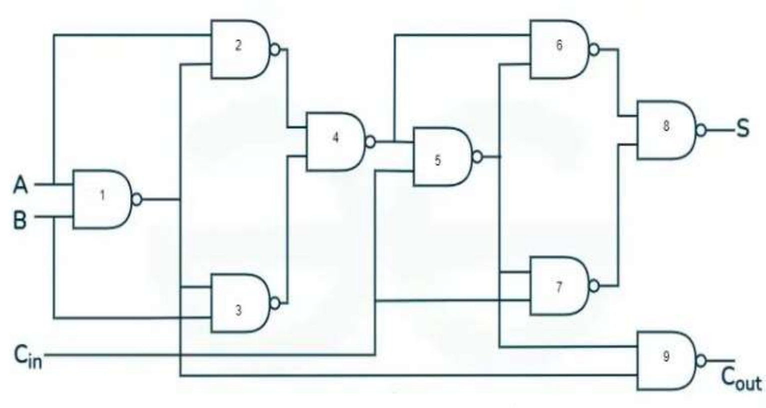
Logic Expression

$$\text{Sum, } S = A'B'C_{in} + A'BC'_{in} + AB'C'_{in} + ABC_{in}$$
$$\text{Carry, } C_{out} = AB + AC_{in} + BC_{in}.$$

Logic Diagram



Realization of Full-Adder using NAND Gates



5.8 Multiplexer

A multiplexer is a combinational circuit that has 2^n input lines and a single output line. Simply, the multiplexer is a multi-input and single-output combinational circuit. The binary information is received from the input lines and directed to the output line. On the basis of the values of the selection lines, one of these data inputs will be connected to the output.

Unlike encoder and decoder, there are n selection lines and 2^n input lines. So, there is a total of 2^N possible combinations of inputs. A multiplexer is also treated as Mux.

There are various types of the multiplexer which are as follows:

- 2-to-1(1selectline)
- 4-to-1(2selectlines)
- 8-to-1(3selectlines)
- 16-to-1(4selectlines)

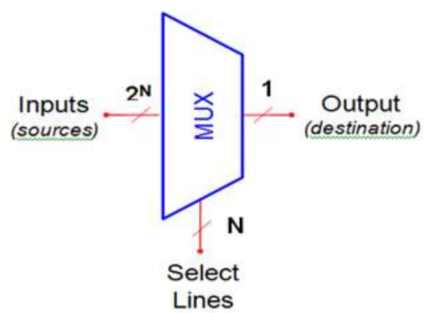
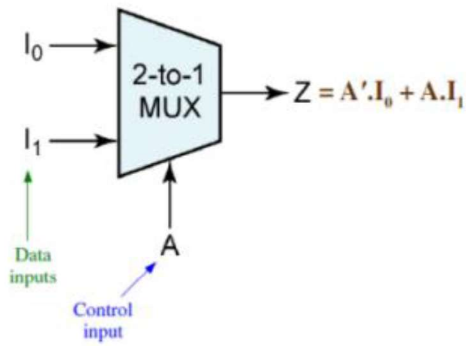


Figure 5.4: Block Diagram of Multiplexer

5.8.1 2×1 Multiplexer

In 2×1 multiplexer, there are only two inputs, i.e., A_0 and A_1, 1 selection line, i.e., S_0 and single outputs, i.e., Y. On the basis of the combination of inputs which are present at the selection line S_0, one of these 2 inputs will be connected to the output. The block diagram and the truth table of the 2×1 multiplexer is given below.

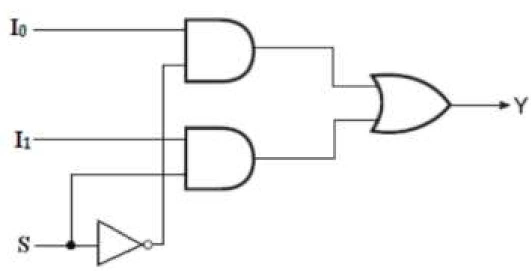
Circuit Diagram



Truth Table

S	Y
0	I_0
1	I_1

Logic Diagram



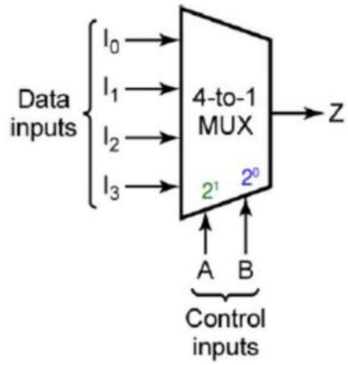
The **Logical Expression** of the term **Y** is as follows:

$\mathbf{Y=S_0'.\ A_0 + S_0.\ A_1}$

5.8.2 4×1 Multiplexer:

In the 4×1 multiplexer, there is a total of four inputs, i.e., A_0, A_1, A_2, and A_3, 2 selection lines, i.e., S_0 and S_1 and single output, i.e., Y. On the basis of the combination of inputs that are present at the selection lines S^0 and S_1, one of these 4 inputs are connected to the output. The block diagram and the truth table of the 4×1 multiplexer are given below.

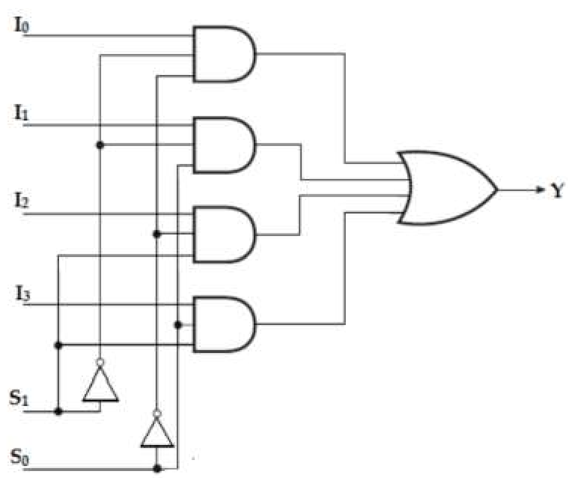
Block Diagram



Truth Table

S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Logic Diagram



The **Logical Expression** of the term **Y** is as follows:

$$Y=S_1' S_0' A_0+S_1' S_0 A_1+S_1 S_0' A_2+S_1 S_0 A_3$$

5.9 De-Multiplexer

A De-multiplexer is a combinational circuit that has only 1 input line and 2^N output lines. Simply, the multiplexer is a single-input and multi-output combinational circuit. The information is received from the single input lines and directed to the output line. On the basis of the values of the selection lines, the input will be connected to one of these outputs. De-multiplexer is opposite to the multiplexer.

Unlike encoder and decoder, there are n selection lines and 2^n outputs. So, there is a total of 2^n possible combinations of inputs. De-multiplexer is also treated as **De-mux**.

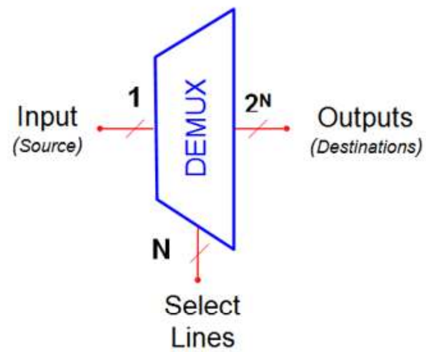
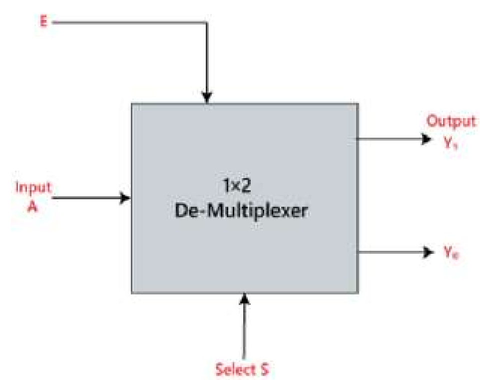


Figure 5.5: Block Diagram of De-Multiplexer

5.9.1 1×2 De-Multiplexer

In the 1 to 2 De-multiplexers, there are only two outputs, i.e., Y_0, and Y_1, 1 selection lines, i.e., S_0, and single input, i.e., A. On the basis of the selection value, the input will be connected to one of the outputs. The block diagram and the truth table of the 1×2 multiplexer is given below.

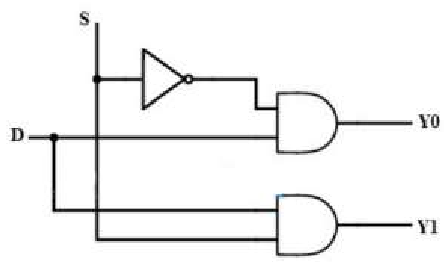
Block Diagram



Truth Table

INPUTS	Output	
S_0	Y_1	Y_0
0	0	A
1	A	0

Logic Diagram



The **Logical Expression** of the term **Y** is as follows:

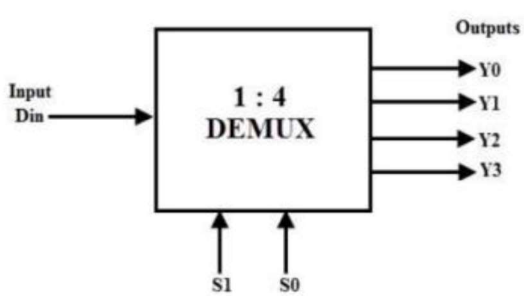
$\mathbf{Y_0 = S_0' A}$

$\mathbf{Y_1 = S_0 A}$

5.9.2 1×4 De-multiplexer

In 1 to 4 De-multiplexer, there are total of four outputs, i.e., Y_0, Y_1, Y_2, and Y_3, 2 selection lines, i.e., S_0 and S_1 and single input, i.e., A. On the basis of the combination of inputs which are present at the selection lines S_0 and S_1, the input be connected to one of the outputs. The block diagram and the truth table of the 1×4 multiplexer is given below.

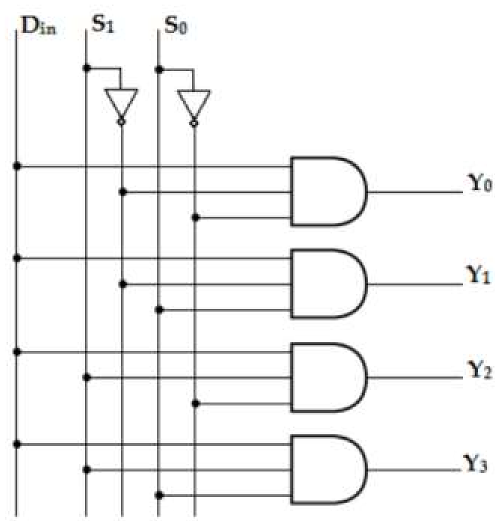
Block Diagram



Truth Table

INPUTS		Output			
S_1	S_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	A
0	1	0	0	A	0
1	0	0	A	0	0
1	1	A	0	0	0

Logic Diagram



The **Logical Expression** of the term **Y** is as follows:

$\mathbf{Y_0 = S_1' \, S_0' \, A}$

$\mathbf{Y_1 = S_1' \, S_0 \, A}$

$\mathbf{Y_2 = S_1 \, S_0' A}$

$\mathbf{Y_3 = S_1 \, S_0 \, A}$

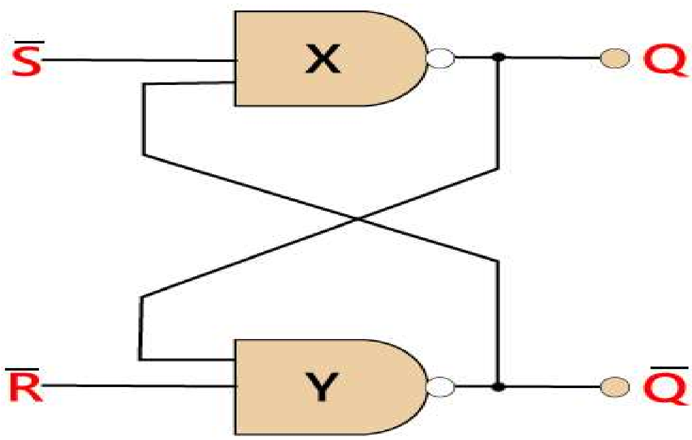
5.10 Flip Flop

A circuit that has two stable states is treated as a **Flip - Flop**. These stable states are used to store binary data that can be changed by applying varying inputs. The flip flops are the fundamental building blocks of the digital system. Flip flops and latches are examples of data storage elements. In the sequential logical circuit, the flip flop is the basic storage element. The latches and flip flops are the basic storage elements but different in working. There are the following types of flip flops:

5.10.1 SR Flip Flop

The **S-R flip flop** is the most common flip flop used in the digital system. In SR flip flop, when the set input "S" is true, the output Y will be high, and Y' will be low. It is required that the wiring of the circuit is maintained when the outputs are established. We maintain the wiring until set or reset input goes high, or power is shutdown.

Logic Diagram



The S-R flip flop is the simplest and easiest circuit to understand.

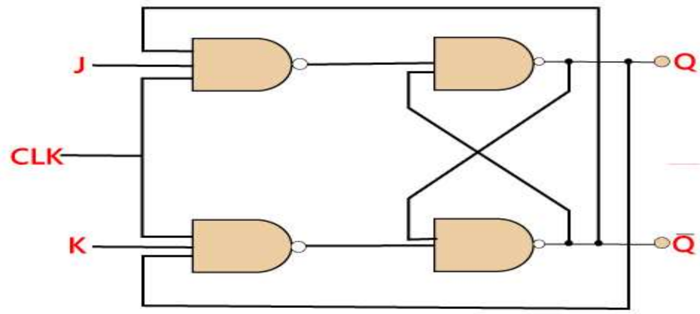
Truth Table

S	R	Y	Y'
0	0	0	1
0	1	0	1
1	0	1	0
1	1	∞	∞

5.10.2 J-K Flip-Flop

The **JK flip flop** is used to remove the drawback of the S-R flip flop, i.e., undefined states. The JK flip flop is formed by doing modification in the SR flip flop. The S-R flip flop is improved in order to construct the J-K flip flop. When S and R input is set to true, the SR flip flop gives an inaccurate result. But in the case of JK flip flop, it gives the correct output.

Logic Diagram



In J-K flip flop, if both of its inputs are different, the value of J at the next clock edge is taken by the output Y. If both of its input is low, then no change occurs, and if high at the clock edge, then from one state to the other, the output will be toggled. The JK Flip Flop is a Set or Reset Flip flop in the digital system.

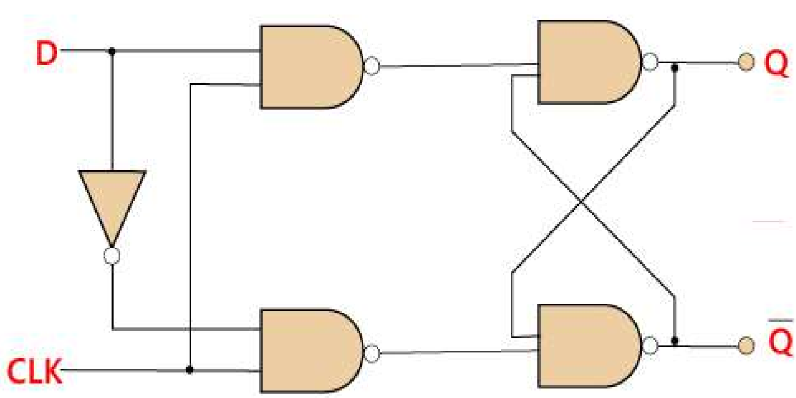
Truth Table

J	K	Y	Y'
0	0	0	0
0	1	0	0
1	0	0	1
1	1	0	1
0	0	1	1
0	1	1	0
1	0	1	1
1	1	1	0

5.10.3 D Flip Flop

D flip flop is a widely used flip flop in digital systems. The D flip flop is mostly used in shift-registers, counters, and input synchronization.

Logic Diagram



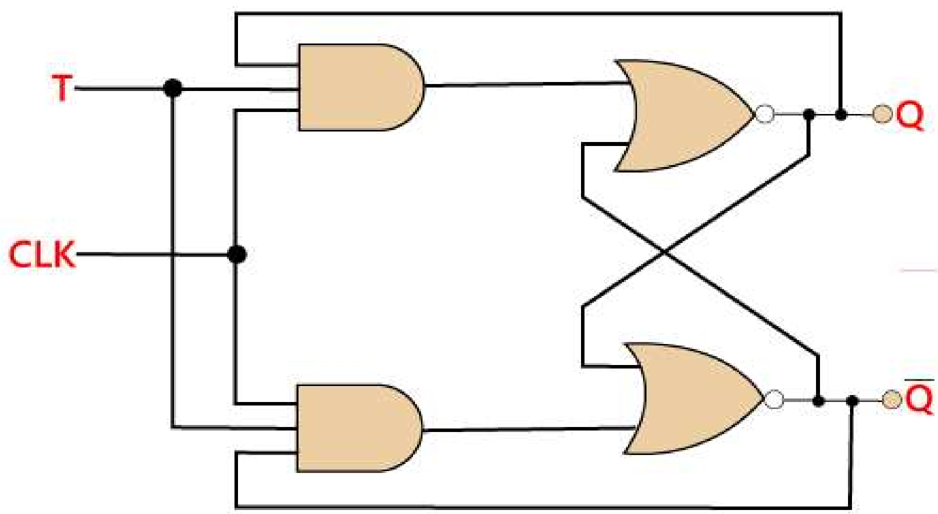
Truth Table

Clock	D	Y	Y’
↓ » 0	0	0	1
↑ » 1	0	0	1
↓ » 0	1	0	1
↑ » 1	1	1	0

5.10.4 T Flip Flop

Just like JK flip-flop, T flip flop is used. Unlike JK flip flop, in T flip flop, there is only single input with the clock input. The T flip flop is constructed by connecting both of the inputs of JK flip flop together as a single input.

Logic Diagram



The **T flip flop is also known as Toggle flip-flop**. These T flip-flops are able to find the complement of its state.

Truth Table

T	Y	Y (t+1)
0	0	0
1	0	1
0	1	1
1	1	0

5.11 Basic Principle of Communication

Sending, receiving, and processing data among two devices are referred to as communication. A communication system is a group of components (devices) that work together to establish a connection between both the sender and recipient. Radio and television, satellite broadcasting, wireless telegraphy, mobile communication, and computer communication are some examples of communication systems.

5.11.1 Principles of Electronic Communication Systems

- Communication is the process of establishing connection or link between two points for information exchange or Communication is simply the basic process of exchanging information.
- The electronics equipment which are used for communication purpose, are called communication equipment. Different communication equipment when assembled together form a communication system.
- Typical example of communication system are line telephony and line telegraphy, radio telephony and radio telegraphy, radio broadcasting, point-to-point communication and mobile communication, computer communication, radar communication, television broadcasting, radio telemetry, radio aids to navigation, radio aids to aircraft landing etc. **Figure 5.6** shows the block diagram of a general communication system, in which the different functional elements are represented by blocks.

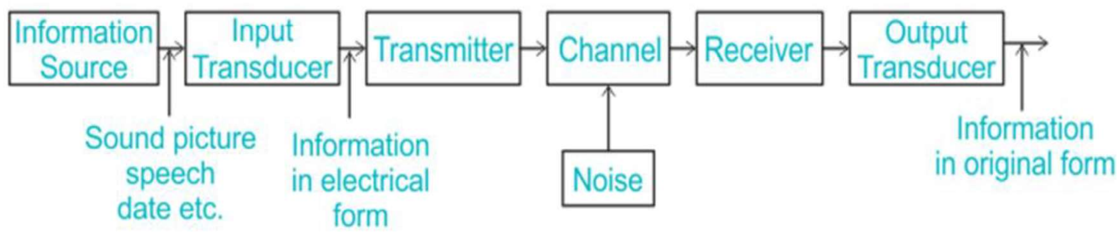


Figure 5.6: Block diagram of Electronic Communication System

The main constituents of basic communication system are:

- Information source and input transducer
- Transmitter
- Channel or medium
- Noise
- Receiver
- Output transducer and final destination

Information Source and Input Transducer

- The physical form of information is represented by a message that is originated by an information source.
- For example, a sentence or paragraph spoken by a person is a message that contains some information. The person, in this case, acts as information source.
- If the information produced by the source is not in an electrical form, it has to be converted into an electrical form using a transducer.

  Eg. Microphone.
- The electrical signal produced by transducer is called the baseband signal. It is also called a message signal, an information signal and is usually designated by s(t).
- There are two types of signals:

  Analog signal and Digital signal.

Transmitter

- Transmitter processes the base band signal received from transducer prior transmission
- There are two following options for processing signals prior transmission:

  o the baseband signal, which lies in the low frequency spectrum, is translated to a higher frequency spectrum (Carrier communication system).

o the baseband signal is transmitted without translating it to a higher frequency spectrum (Baseband communication system).

• The carrier communication system is based on the principle of translating a low frequency baseband signal to high frequency spectrum. This process is **Modulation**.

Channel or Medium

• After the required processing, the transmitter section passes the signal to the transmission medium. The signal propagates through the transmission medium and is received at the other side by the receiver section.

• The transmission medium between the transmitter and the receiver is called a channel.

• Most of the noise is added to the signal during its transmission through the channel.

• Depending on physical implementations, channels can be classified into two groups:

Hardware Channels: These channels are manmade structure. The three possible implementations of the hardware channels are: Transmission lines, Waveguides, and Optical Fiber Cables (OFC)

Software Channels: These are certain natural resources. The natural resources that can be used as software channels are: air or open space and sea water.

Noise

• Noise is defined as unwanted electrical energy of random and unpredictable nature.

• Noise is an electrical disturbance, which does not contain any useful information.

• Noise is a highly undesirable part of a communication system, and has to be minimized.

• When noise is mixed with transmitted signal, it rides over it & deteriorates it waveform.

• This results in alteration of original information so that wrong information is received.

• The designer provides adequate signal strength at the time of transmission so that a high SNR (Signal to Noise Ratio) is available at the receiver.

Receiver

• The function of the receiver section is to separate the noise from the received signal, and then recover the original baseband signal by performing demodulation process.

• A voltage amplifier first amplifies the received signal so that it becomes strong enough for further processing, and then recovers the original information.

• The demodulation process removes the high frequency carrier from the received signal and retrieves the original baseband.

Output Transducer & Final Destination

• The recovered baseband signal is handed over to the final destination, which uses a transducer to convert this electrical signal to its original form.

• Prior to handing over the recovered baseband signal to its final destination, the voltage and power are amplified by the amplifier stages.

5.11.2 Evolution of Communication systems

1G – First-Generation System

1G laid the foundation and opened the door to wireless telephone communication before us. It used analog technology, was limited to voice calls, and offered a maximum speed of 2.4 kbps. During 1G, cell phones were big, heavy, and expensive. Also, battery drainage and poor voice quality were other limitations.

2G – Second-generation Communication System (GSM)

The Global System for Mobile Communications (GSM), the second generation (2G) standard developed by the European Telecommunications Standards Institute (ETSI), is based on Time Division Multiple Access (TDMA).

The 2G mobile phones used digital modulation and enabled a maximum speed of 14.4 kbps. Voice calls and SMS were supported, and mobile phones got smaller and more secure.

2.5G and 2.75G Communication System

The transition from 2G to 2.5G marked an advancement in mobile communication technology, where enhancements were introduced to the existing 2G networks.

Also, 2.5G represented a notable shift from primarily catering to voice communication in 2G, as it integrated packet-switched data services, enabling basic internet usage and data applications alongside traditional circuit-switched voice services. Further, in 2.5G GPRS, the subscriber data transfer rates got enhanced up to 171 kbps.

EDGE (Enhanced Data Rates for GSM Evolution) or 2.75G is an enhancement of GPRS for data transmission. Also, it works on GSM networks, an extension of GPRS and allows for speeds up to 384 kbps.

3G – Third-generation Communication System

Later, second-generation (2G) cellular technology evolved into third-generation (3G) cellular technology based on the Universal Mobile Telecommunications System (UMTS). Furthermore, this technology changed the primary focus from voice and text to mobile data.

The advent of 3G networks in the first decade of the century paved the foundation for high-speed internet and wireless applications. Further, it resulted in a digitally powered era in communications.

UMTS is a third-generation cellular technology. It allows 2G GSM networks to migrate to 3G. UMTS uses Wideband CDMA (WCDMA) for its radio interface. Further, it enables peak download data rates of up to 2 Mbps and average download speeds of around 384 kbps.

Also, 3G paved the way for video call and streaming services. With the latest enhancements in High-Speed Packet Access (HSPA and HSPA+), UMTS networks can enable peak data rates of up to 71.6 Mbps.

How Does the 3G UMTS Network Function?

The User Equipment (UE) includes two components such as,

1. i) Mobile equipment and
2. ii) Universal Subscriber Identity Module (USIM)

Using the Air interface, the User Equipment connects to the UTRAN (Universal Terrestrial Radio Access Network). UTRAN consists of Node Bs and RNCs, and each RNC (Radio Network Controller) manages multiple Node Bs. In UMTS, there exists a circuit-switched core and a packet-switched core.

In the circuit-switched core, the Mobile Switching Center (MSC) is responsible for voice calls, delivering text messages and tracking down mobile locations.

Gateway MSC (GMSC) offers the connection to other service providers (mobile or fixed). The Home Location Register (HLR) keeps a repository of all the subscribers belonging to a service provider.

The Serving GPRS Support Node (SGSN) in the packet-switched core manages the data connection between the mobile and the Packet Data Network. It also tracks the location of the mobile for data services.

The Gateway GPRS Support Node (GGSN) provides the connection to external data networks. Also, it is an anchor point as the user moves to a different SGSN due to mobility.

How is 4G Different from 3G?

4G was commercially deployed in 2009. The 3G network only uses IP for data, enabling voice with a circuit-switched network. On the other hand, 4G is an all-IP-based standard for both voice and data. For this reason, 4G is more efficient for mobile network providers to operate and optimize instead of managing different network technologies for data and voice.

There are two flavors of 4G – LTE and WiMax.

The Long-Term Evolution (LTE) is fully packet-switched, which uses Orthogonal Frequency Division Multiple Access (OFDMA). LTE is designed to provide connectivity between a user's equipment and a Packet Data network with a data rate of up to 100 Mbps. LTE Advanced (LTEA) is an enhancement that improves the original LTE technology and could deliver up to 1000 Mbps. In addition, LTE supports VoIP, video conferencing, HD Mobile TV, online gaming, mobile broadband and mobile apps.
The key components include the E-UTRAN and the Evolved Packet Core (EPC), collectively forming the LTE network. The evolved NODE B (eNodeB) manages scheduling, handovers and security. The Serving gateway(S-GW) handles mobility between E-UTRAN and EPC.

The PDN Gateway (P-GW) connects the EPC to the Packet Data Network. The controlling entity is the Mobility Management Entity (MME), which tracks the location of UE. Also, it is responsible for session management. The HSS is the central repository of subscriber information.

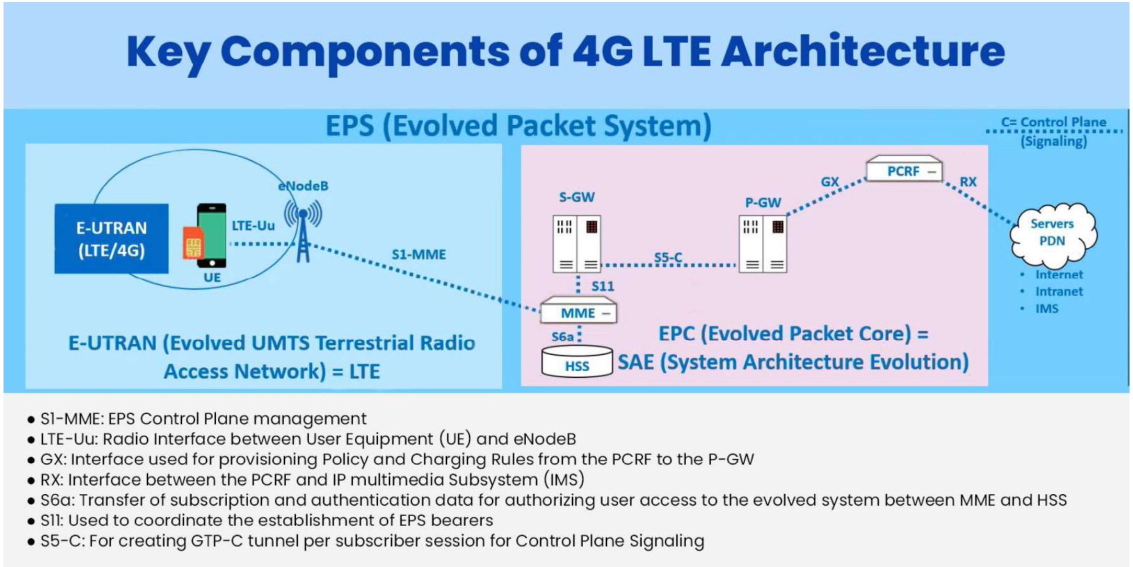


Figure 5.7: Key Components of 4G LTE Architecture

5G – Fifth-generation Communication System

5G network is not just about providing huge data rates. It can create an adaptive, flexible network that can connect virtually everything, including machines, objects and devices. Also, it can provide different features to different customers with many potential and possibilities.

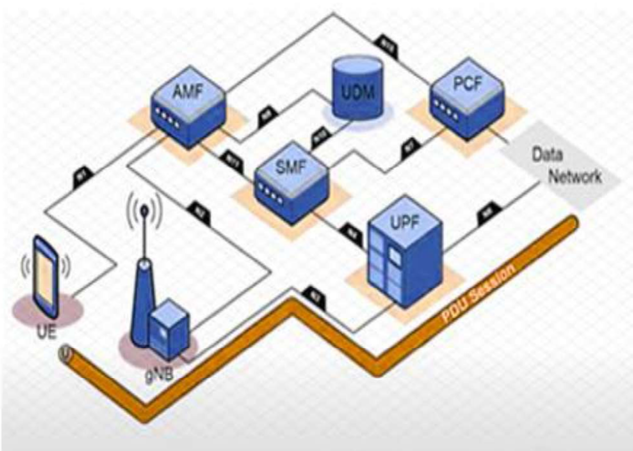
The Radio Access Network belonging to 5G is known as New Radio (NR). The connectivity of 5G NR to a 5G core network is provided by Standalone architecture. The network architecture, known as Non-Standalone architecture, is based on tight interworking with LTE and NR, allowing a smooth evolution towards an end-to-end 5G system.

What Makes 5G Exciting?

The 5G NR technology, including millimeter wave (mmWave) and massive Multiple-Input Multiple-Output (MIMO) with beamforming, enables a network to deliver very high speed, reduced low latency and more data capacity.

The peak data rate of the network for the download link is 20 Gbps and 10 Gbps for uplink and offers a latency of less than 1 millisecond. The new use cases of 5G networks include enhanced Mobile Broadband (eMBB), ultra Reliable Low Latency Communication (uRLLC)and mMTC massive Machine Type Communication (mMTC).

As traffic volume increases exponentially, eMBB can deliver speeds in multi-Gbps peak data rates much faster than its previous generation. With uRLCC, latency can be minimal for applications like self-driving cars. In such cases, the response time can make much difference, facilitating decisions in real-time. Further, it is easy to connect Many smart devices to the network for an extended period, and mMTC delivers a network capable of handling this type of demand.



gNB	gNodeB
AMF	Access and Mobility Function
SMF	Session Management Function
UPF	User Plane Function
PCF	Policy Control Function
UDM	Unified Data Management

Figure 5.8: 5G Network Architecture

Access and Mobility Function (AMF) knows the cell or tracking area where the subscriber is located. AMF ensures that the subscriber is allowed on the network and authenticates the subscriber. Also, AMF allocates the user equipment with a Globally Unique Temporary ID (GUTI) during mobility and periodic updates.

The Session Management Function (SMF) is responsible for the user equipment's session management and IP address allocation. Directions based on decisions related to creating, modifying or terminating a session to the UPF are given.

SMF liaisons with PCF for policy and QoS enforcement. Also, SMF performs the selection and control of UPF.Unified Data Management (UDM) stores subscriber profiles and data network profiles. User Plane Function (UPF) is responsible for processing and forwarding data. If the user moves from one g-Node B to another, the traffic continues on the same connected UPF.Based on the rules from SMF, UPF ensures the quality of service. Packet Data Unit (PDU) sessions provide connectivity between the device and the Data Network. Quality of service (QoS) flow within a PDU session offers different QoS levels for different services.The Policy Control Function (PCF) takes dynamic decisions based on network conditions. Hence, it decides the correct resource allocation for a user to access a particular service.