

## Unit no 03: Digital Systems and Logic Designs

### Long Question Answers:

#### 1. Explain the usage of Boolean functions in computers.

##### □ Usage in Computers:

There are many uses of Boolean functions in the computers for various operations.

Here are some examples of their usage:

- **Arithmetic Operations:** Boolean functions are used in Arithmetical Logic Units (ALUs) of CPUs to perform operations like addition, subtraction, multiplication, and even division.
- **Data Processing:** Boolean functions are used to process binary data in memory and storage devices, ensuring efficient data manipulation and retrieval.
- **Decision Making:** Boolean functions enable **conditional operations**, such as 'If' statements in programming. CPUs use Boolean logic to make decisions based on certain conditions (e.g., whether a value is zero or if two values are equal).
- **Control Logic:** Boolean functions are applied in computers to control various parts of the system's operation to function in a coordinated manner.
- Boolean functions are also present in our everyday devices like cell phones and calculators:
- **Cell Phones:** In cell phone processing, when you dial a number, or press a button on a phone, a Boolean function evaluates these inputs as true or false and makes the necessary output.
- **Calculators:** Basic calculators use Boolean functions. When you feed it with numbers and the operations to be performed, Boolean logic is used to arrive at the right result.

## 2. Describe how to construct a truth table for a Boolean expression with an example.

Boolean functions are algebraic statements that describe the relationship between binary variables and logical operations. These functions are particularly important for digital circuits, which are the basis of current computers, mobile phones and even simple calculator.

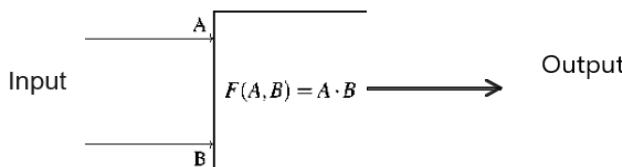
### Understanding Boolean Functions:

A Boolean function is a function which has a one or more binary inputs and produces a single binary output. The inputs and outputs can only have two values: False (represented by 0) and True (represented by 1). The construction of Boolean functions is done by employing the basic logical operations such as AND, OR and NOT, which connect the inputs to generate the correct output.

### Example 1: Simple Boolean Function

Consider a Boolean function with two inputs, A and B. We can construct a function F that represents the AND operation:

$$F(A, B) = A \cdot B$$



The diagram shown above demonstrates a basic digital circuit, which is an AND gate. The box symbolizes the AND function  $F(A, B) = A \cdot B$ . This box has two inputs A and B. If both A and B are 1, the output will be 1. In any other case, the output will be 0. The

input are shown at the entrance to the box, while the output is depicted at the exit of the block. The truth table for this function is as follows:

A	B	F(A, B)
0	0	0
0	1	0
1	0	0
1	1	1

**Example 2: Now, let us construct a more complex Boolean function with three inputs, A, B, and C:**

let us construct a complex Boolean function with three inputs, A, B, and C:

$$F(A, B, C) = A \cdot B + \bar{A} \cdot C$$

This function uses AND, OR and NOT at the same time. The truth table for this function is as follows:

**Explanation:**

- The parameters A, B, and C are included in the following example as the input columns.
- The results of AND operation between two variable A and B are presented in the column  $A \cdot B$ .
- The column  $\bar{A}$  standing for the NOT operation of A.
- Every value in the column  $\bar{A} \cdot C$  displays the result of AND operation between the values in the Fifth column and the third column.
- The final column  $F(A, B, C)$  shows the output of the Boolean function  $(A \cdot B) + (\bar{A} \cdot C)$

## COMPUTER SCIENCE

A	B	C	A·B	$\bar{A}$	$\bar{A} \cdot C$	<u>F(A, B, C)</u>
0	0	0	0	1	0	0
0	0	1	0	1	1	1
0	1	0	0	1	0	0
0	1	1	0	1	1	1
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	1	0	0	1
1	1	1	1	0	0	1

**3. Describe the concept of duality in Boolean algebra and provide an example to illustrate it.**

### Simplification of Boolean Functions

Simplification of Boolean functions is a particularly important process in designing an efficient digital circuit. Such simplified functions require fewer gates making them compact in size, energy efficient and faster than the complicated ones. Simplification means applying some Boolean algebra rules to make the functions less complicated.

#### Basic Boolean Algebra Rules:

Here is some fundamental Boolean algebra rules used for simplification:

## COMPUTER SCIENCE

## 1. Identity Laws

$$\begin{aligned} A + 0 &= A \\ \underline{A} \cdot 1 &= A \end{aligned}$$

## 2. Null Laws

$$\begin{aligned} A + 1 &= 1 \\ \underline{A} \cdot 0 &= 0 \end{aligned}$$

## 3. Idempotent Laws

$$\begin{aligned} A + A &= A \\ \underline{A} \cdot A &= A \end{aligned}$$

## 4. Complement Laws

$$\begin{aligned} A + \underline{A} &= 1 \\ \underline{A} \cdot A &= 0 \end{aligned}$$

## 5. Commutative Laws

$$\begin{aligned} A + B &= B + A \\ \underline{A} \cdot B &= B \cdot A \end{aligned}$$

## 6. Associative Laws

$$\begin{aligned} (A + B) + C &= A + (B + C) \\ (\underline{A} \cdot B) \cdot C &= \underline{A} \cdot (B \cdot C) \end{aligned}$$

## 7. Distributive Laws

$$\begin{aligned} \underline{A} \cdot (B + C) &= (\underline{A} \cdot B) + (\underline{A} \cdot C) \\ A + (\underline{B} \cdot C) &= (\underline{A} + B) \cdot (A + C) \end{aligned}$$

## 8. Absorption Laws

$$\begin{aligned} A + (\underline{A} \cdot B) &= A \\ \underline{A} \cdot (A + B) &= A \end{aligned}$$

## 9. De Morgan's Theorems

$$\begin{aligned} \overline{A + B} &= \underline{\overline{A} \cdot \overline{B}} \\ \underline{A} \cdot \overline{B} &= A + B \end{aligned}$$

## 10. Double Negation Law

$$=$$

$$A = A$$

## Simplification Examples

**Example 1**  
Simplify the expression  $A + \overline{A} \cdot B$ .

**Solution:**

$$\begin{aligned} A + \overline{A} \cdot B &= (A + \overline{A}) \cdot (A + B) \\ &= \underline{1} \cdot (A + B) \\ &= A + B \end{aligned}$$

(Distributive Law)  
(Complement Law)  
(Identity Law)

**Example 2**

Simplify the expression  $\overline{A \cdot B} + \overline{A} \cdot B$ .

**Solution:**

$$\begin{aligned} \overline{A \cdot B} + \overline{A} \cdot B &= \overline{A} + \overline{B} + \overline{A} \cdot B \\ &= (\underline{\overline{A} + B}) \\ &= \overline{A} + B \end{aligned}$$

(De Morgan's Theorem)  
Since  $\overline{A}$  is already present in  $(\underline{A} \cdot B)$ , we can use absorption law  
 $\therefore A + (\underline{A} \cdot B) = A$

**Example 3**

Simplify the expression  $(A + B) + (A + \overline{B})$ .

**Solution:**

$$\begin{aligned} (A + B) + (A + \overline{B}) &= A \cdot (A + \overline{B}) + B \cdot (A + \overline{B}) \\ &= A + A \cdot \overline{B} + B + B \cdot \overline{B} \\ &= A + A \cdot B \\ &= \underline{A} \cdot (1 + B) \\ &= A \cdot 1 \\ &= A \end{aligned}$$

(Distributive Law)  
(Absorption Law)  
(Identity Law)  
(Distributive Law)  
(Null Law)  
(Identity Law)

**Example 4**

Simplify the expression  $\overline{A} + \underline{B} \cdot (A + \overline{B})$ .

**Solution:**

$$\begin{aligned} \overline{A} + \underline{B} \cdot (A + \overline{B}) &= (\overline{A} \cdot \underline{B}) + (A + \overline{B}) \\ &= A \cdot \overline{B} + A + A \cdot \overline{B} + \overline{B} \\ &= A \cdot \overline{B} + \underline{A} \cdot \overline{B} \\ &= A \cdot \overline{B} \end{aligned}$$

(De Morgan's Theorem)  
(Distributive Law)  
(Idempotent Law)  
(Identity Law)

**4. Compare and contrast half-adders and full-adders, including their truth tables, Boolean expressions, and circuit diagrams.**

Digital logic is an essential aspect for the functioning of several modern electronic systems, such as computers, smart phones, and other digital gadgets. Digital logic optimizes in many ways to create and enhance circuits meant to perform various tasks. Two important applications of digital logic are the design of adder circuits and the use of Karnaugh maps for function simplification.

### **Half-adder and Full-adder Circuits**

Adder circuits are widely used in digital circuits to perform arithmetic calculations. There are two general forms of adder circuits known as half-adders and full adders.

#### **1. Half-adder Circuits**

A half adder is a basic circuitry unit that performs addition of two single-bit binary digits. It has two inputs, usually denoted as A and B, and two outputs: the sum (S) and the carry (C).

#### **Truth Table for Half-adder:**

A	B	Sum (S)	Carry (C)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

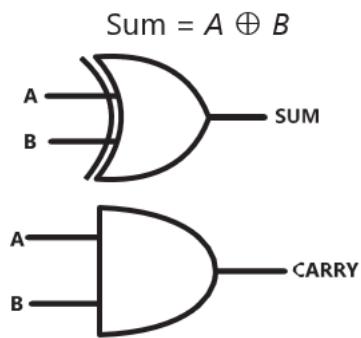
#### **Boolean Expressions for Half-adder:**

$$S = A \oplus B \\ C = A \cdot B$$

In this case the symbol  $\oplus$  represents the XOR operation. The sum output is high when only one of the inputs is high, while the carry output is high when both inputs are high.

#### **Boolean Expressions:**

COMPUTER SCIENCE



### Full-adder Circuits

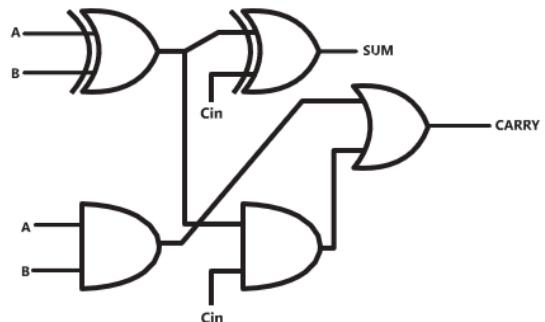
A full-adder is a more complex circuit that adds three single-bit binary numbers: two bits that belong to the sum and a carry bit from a previous addition. It has three inputs, denoted as A, B, and Cin (carry input), and two outputs: called the sum (S) and the carry (Cout) with both being integer values.

A	B	Cin	Sum (S)	Carry (C <sub>out</sub> )
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

**Boolean Expressions:**

$$\text{Sum} = A \oplus B \oplus \text{Cin}$$

$$\text{Carry} = (A \cdot B) + (\text{Cin} \cdot (A \oplus B))$$



The sum output is high if the number of high inputs is odd whereas the carry output is high if the number of high inputs is at least 2.

**5. How do Karnaugh maps simplify Boolean expressions? Provide a detailed example with steps.**

A Karnaugh map (K-map) is a graphical representation which can be used to solve Boolean algebra expressions and minimize a logic function where algebraic computations are not employed. It is a technique in which the truth value of Boolean function is plotted to enable the identification of patterns and to perform term combining for simplification.

Minterm	Variables Combination	Minterm Expression
m0	A= 0, B= 0, C= 0	$\overline{ABC}$
m1	A= 0, B= 0, C= 1	$\overline{AB}C$
m2	A= 0, B= 1, C= 0	$\overline{A}B\overline{C}$
m3	A= 0, B= 1, C= 1	$\overline{A}B C$
m4	A= 1, B= 0, C= 0	$A\overline{B}\overline{C}$
m5	A= 1, B= 0, C= 1	$A\overline{B}C$
m6	A= 1, B= 1, C= 0	$AB\overline{C}$

## COMPUTER SCIENCE

m7	A= 1, B= 1, C= 1	A B C
----	------------------	-------

A K-map is a matrix where each square is a cell, which corresponds to a positioned combination. These cells are filled with '1' or '0' in reference to the truth table of the Boolean function. The size of the K-map depends on the number of variables:

- 2 Variables: 2x2 grid
- 3 Variables: 2x4 grid
- 4 Variables: 4x4 grid least
- 5 Variables: 4x8 grid (less common for manual simplification)

Every cell in the K-map represents a minterm, and the cells in each row of the K-map differ by only one bit at any particular position, following the gray code sequence.

**6. Design a 4-bit binary adder using both half-adders and full-adders. Explain each step with truth tables, Boolean expressions, and circuit diagrams.**

**Half Adder Circuits:**

A half adder is a basic circuitry unit that performs addition of two single-bit binary digits. It has two inputs, usually denoted as A and B, and two outputs: the sum S and the carry C.

**Truth Table for Half Adder**

A	B	Sum (S)	Carry (C)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

**Boolean Expressions for Half-adder:**

$$\begin{aligned} S &= A \\ &\oplus B \ C \\ &= A \cdot B \end{aligned}$$

In this case the symbol  $\oplus$  represents the XOR operation. The sum output is high when only one of the inputs is high, while the carry output is high when both inputs are high.

#### Boolean Expressions:

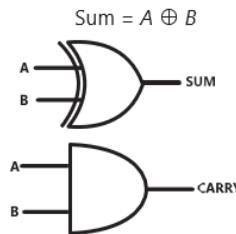


Table 3.8: Half-Adder Circuit

#### Full Adder Circuits:

A full-adder is a more complex circuit that adds three single-bit binary numbers: two bits that belong to the sum and a carry bit from a previous addition. It has three inputs, denoted as A, B, and Cin (carry input), and two outputs: called the sum (S) and the carry (Cout) with both being integer values.

A	B	Cin	Sum (S)	Carry (C <sub>out</sub> )
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

#### Boolean Expression:

$$\text{Sum} = A \oplus B \oplus \text{Cin}$$

$$Carry = (A \cdot B) + (Cin \cdot (A \oplus B))$$

The sum output is high if the number of high inputs is odd whereas the carry output is high if the number of high inputs is at least 2.

**7. Simplify the following Boolean function using Boolean algebra rules:**

$$F(A, B) = A \cdot B + A \cdot B$$

**Solution:**

$$F(A, B) = A \cdot (B + 1) \text{ Distributive Law}$$

$$B + 1 = 1 \quad \text{Null Law}$$

$$F(A, B) = A \cdot 1$$

$$F(A, B) = A \quad \text{Identity Law}$$

$$F(A, B) = A \quad \text{Answer}$$

**8. Use De Morgan's laws to simplify the following function:**

$$F(A, B, C) = A + B + AC$$

Solution:

De-Morgan's Law states:

$$\begin{aligned} \overline{A + B} &= \overline{A} \cdot \overline{B} \\ A \cdot B &= A + B \end{aligned}$$

$$F(A, B, C) = (A \cdot B) + AC$$

$$F(A, B, C) = (A \cdot B) + (A \cdot C)$$

$$F(A, B, C) = A \cdot B + A \cdot C \quad \text{Answer}$$

## COMPUTER SCIENCE

## 9. Simplify the following expressions:

(a)  $F = \overline{A + B} \cdot (A + B)$

**Ans.**

$$X \cdot \overline{X} = 0 \quad \text{Null Law}$$

**Using Complement Law**

So,  $F = 0$

(b)  $(A + \overline{B}) \cdot (\overline{A} + B)$

**Ans.**  $F = (A + \overline{B}) \cdot (\overline{A} + B)$

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

$$A \cdot \overline{A} = 0 \quad (\text{Complement Law})$$

$$\overline{B} \cdot B = 0 \quad (\text{Complement Law})$$

$$F = 0 + A \cdot B + \overline{B} \cdot \overline{A} + 0$$

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

(c)  $A + \overline{A} \cdot (\overline{B} + C)$

**Ans.**  $F = A + \overline{A} \cdot (\overline{B} + C)$

$$F = A + (\overline{A} \cdot \overline{B}) + (\overline{A} \cdot C) \quad \text{Distributive law}$$

$$F = A + \overline{B} + \overline{A} \cdot C \quad \text{Absorption Law}$$

(d)  $\overline{A + B} + A \cdot B$

**Ans.**  $F = \overline{A} \cdot \overline{B} + A \cdot B$  by using **De-Morgan's law**

(e)  $(A \cdot B) + (\overline{A} \cdot \overline{B})$

**Ans.**  $F = (A \cdot B) + (\overline{A} \cdot \overline{B})$

$$F = A \odot B$$

**OR**  $F = A \oplus B$