Experiment No 4

Introduction to Boolean Algebra and Logic Simplification

4.1 Objectives

After completing this experiment, student will be able to:

- Comprehend different Boolean Algebra postulates and laws.
- Experimentally verify different rules of Boolean Algebra.
- Experimentally determine the truth tables for circuits with three input variables, and use DeMorgan's theorem to prove algebraically whether they are equivalent.
- Comprehend Logic simplification techniques and implement it.
- Construct circuits from simplified logic expressions.

4.2 Background Theory

A set of rules or Laws of Boolean Algebra expressions have been invented to help reduce the number of logic gates needed to perform a particular logic operation resulting in a list of functions or theorems known commonly as the Laws of Boolean Algebra. Boolean Algebra is the mathematics we use to analyze digital gates and circuits. We can use these "Laws of Boolean" to both reduce and simplify a complex Boolean expression in an attempt to reduce the number of logic gates required. Boolean Algebra is therefore a system of mathematics based on logic that has its own set of rules or laws which are used to define and reduce Boolean expressions.

The variables used in Boolean Algebra only have one of two possible values, a logic "0" and a logic "1" but an expression can have an infinite number of variables all labelled individually to represent inputs to the expression, For example, variables A, B, C etc, giving us a logical expression of A + B = C, but each variable can ONLY be a 0 or a 1.

4.2.1 Laws of Boolean Algebra

The basic Laws of Boolean Algebra that relate to the Commutative Law allowing a change in position for addition and multiplication, the Associative Law allowing the removal of brackets for addition and multiplication, as well as the Distributive Law allowing the factoring of an expression, are the same as in ordinary algebra. Each of the Boolean Laws above are given with just a single or two variables, but the number of variables defined by a single law is not limited to this as there can be an infinite number of variables as inputs too the expression. These Boolean laws detailed above can be used to prove any given Boolean expression as well as for simplifying complicated digital circuits. A brief description of the various Laws of Boolean are given below with A representing a variable input.

4.2.1.1 Description of the Laws of Boolean Algebra

A brief description of various laws of Boolean Algebra is given below

4.2.1.1.1 Distributive Law

This law permits the multiplying or factoring out of an expression.

A(B + C) = A.B + A.C (OR Distributive Law)

A + (B.C) = (A + B).(A + C) (AND Distributive Law)

4.2.1.1.2 Associative Law

This law allows the removal of brackets from an expression and regrouping of the variables.

A + (B + C) = (A + B) + C = A + B + C (OR Associate Law)

A(B.C) = (A.B)C = A . B . C (AND Associate Law)

4.2.1.1.3 Commutative Law

The order of application of two separate terms is not important

A . B = B . A (The order in which two variables are AND'ed makes no difference)

A + B = B + A (The order in which two variables are OR'ed makes no difference)

4.2.1.1.4 Absorptive Law

This law enables a reduction in a complicated expression to a simpler one by absorbing like terms.

A + (A.B) = (A.1) + (A.B) = A(1 + B) = A (OR Absorption Law)

A(A + B) = (A + 0).(A + B) = A + (0.B) = A (AND Absorption Law)

4.2.1.1.5 Annulment Law

A term AND'ed with a "0" equals 0 or OR'ed with a "1" will equal 1

A . 0 = 0 (A variable AND'ed with 0 is always equal to 0)

A + 1 = 1 (A variable OR'ed with 1 is always equal to 1)

4.2.1.1.6 Identity Law

A term OR'ed with a "0" or AND'ed with a "1" will always equal that term

A + 0 = A (A variable OR'ed with 0 is always equal to the variable)

A . 1 = A (A variable AND'ed with 1 is always equal to the variable)

4.2.1.1.7 Idempotent Law

An input that is AND'ed or OR´ed with itself is equal to that input

A + A = A (A variable OR'ed with itself is always equal to the variable)

A . A = A (A variable AND'ed with itself is always equal to the variable)

4.2.1.1.8 Complement Law

A term AND'ed with its complement equals "0" and a term OR´ed with its complement equals "1"

$A . \overline{A} = 0$ (A variable AND'ed with its complement is always equal to 0)

$A + \overline{A} = 1$ (A variable OR'ed with its complement is always equal to 1)

4.2.1.1.9 Double Negation Law

A term that is inverted twice is equal to the original term

$\overline{\overline{A}} = A$ (A double complement of a variable is always equal to the variable)

4.2.1.1.10 De Morgan's Law

There are two "de Morgan's" rules or theorems,

1) Two separate terms NOR'ed together is the same as the two terms inverted (Complement) and AND'ed for example: $\overline{A + B} = \overline{A} . \overline{B}$
2) Two separate terms NAND'ed together is the same as the two terms inverted (Complement) and OR'ed for example: $\overline{A . B} = \overline{A} + \overline{B}$

4.2.1.2 Boolean Postulates

There are a set of Mathematical Laws which can be used in the simplification of Boolean Expressions.

- 0 . 0 = 0 (A 0 AND'ed with itself is always equal to 0)
- 1 . 1 = 1 (A 1 AND'ed with itself is always equal to 1)
- 1 . 0 = 0 (A 1 AND'ed with a 0 is equal to 0)
- 0 + 0 = 0 (A 0 OR'ed with itself is always equal to 0)
- 1 + 1 = 1 (A 1 OR'ed with itself is always equal to 1)
- 1 + 0 = 1 (A 1 OR'ed with a 0 is equal to 1)
- $\overline{1} = 0$ The Inverse (Complement) of a 1 is always equal to 0
- $\overline{0} = 1$ The Inverse (Complement) of a 0 is always equal to 1

4.2.2 Sum of Products (SOP)

The short form of the sum of the product is SOP, and it is one kind of **Boolean algebra** expression. In this, the different product inputs are being added together. The product of inputs is Boolean logical AND whereas the sum or addition is Boolean logical OR. Before going to understand the concept of the sum of products, we have to know the concept of minterm.

A **minterm**, denoted as mi, where $0 \leq i < 2n$, is a product (AND) of the n variables in which each variable is complemented if the value assigned to it is 0, and uncomplemented if it is 1.

- 1-minterms = minterms for which the function F = 1.
- 0-minterms = minterms for which the function F = 0.

The truth table of the min term is shown below.

Table 4.1: Truth table of Min term

Sr. No.	X	Y	Z	F	Min Term (m)
1.	0	0	0	0	X'Y'Z' = m0
2.	0	0	1	0	X'Y'Z = m1
3.	0	1	0	0	X'YZ' = m2
4.	0	1	1	1	X'YZ = m3
5.	1	0	0	0	XY'Z' = m4
6.	1	0	1	1	XY'Z = m5
7.	1	1	0	1	XYZ' = m6
8.	1	1	1	1	XYZ = m7

In the above table 4.1, there are three inputs namely X, Y, Z and the combinations of these inputs are 8. Every combination has a minterm that is specified with m.

4.2.2.1 Types of Sum of Product (SOP)

The **sum of products** is available in **three different forms** which include the following.

- Canonical Sum of Products
- Non-Canonical Sum of Products
- Minimal Sum of Products

4.2.2.1.1 Canonical Sum of Products

This is a normal form of SOP, and it can be formed with grouping the minterms of the function for which the o/p is high or true, and it is also called as the sum of minterms. The expression of the canonical SOP is denoted with sign summation (∑), and the minterms in the bracket are taken when the output is true. The truth table of the canonical sum of the product is shown below.

Table 4.2: Truth table of the canonical sum of the product

Sr. No.	X	Y	Z	F
1.	0	0	0	0
2.	0	0	1	1
3.	0	1	0	1
4.	0	1	1	1
5.	1	0	0	0
6.	1	0	1	1
7.	1	1	0	0
8.	1	1	1	0

For the above table, the **canonical SOP form** can be written as:

$$F = \sum (m1, m2, m3, m5)$$

By expanding the above summation we can get the following function.

$$F = m1 + m2 + m3 + m5$$

By substituting the minterms in the above equation we can get the below expression

$$F = X'Y'Z + X'YZ' + X'YZ + XY'Z$$

The product term of the canonical form includes both complemented and non-complimented inputs.

4.2.2.1.2 Canonical Sum of Products

In the non-canonical sum of product form, the product terms are simplified. For example, let’s take the above canonical expression.

$$F = X'Y'Z + X'YZ' + X'YZ + XY'Z$$

$$F = X'Y'Z + X'Y (Z'+Z) + XY'Z$$

Here **Z’+Z =1** (Standard function)

$$F = X'Y'Z + X'Y (1) + XY'Z$$

$$F = X'Y'Z + X'Y + XY'Z$$

This is still in the form of SOP, but it is the non-canonical form

4.2.2.1.3 Minimal Sum of Products

The minimum sum of products (MSOP) of a function, f, is a SOP representation of f that contains the fewest number of product terms and fewest number of literals of any SOP representation of f. This is the most simplified expression of the sum of the product, and it is also a type of non-canonical. This type of can is made simplified with the Boolean algebraic theorems although it is simply done by using K-map (Karnaugh map).

This form is chosen due to the number of input lines & gates are used in this is minimum. It is profitably useful due to its solid size, quick speed, along with low manufacture price.

- f= (xyz +x`yz+ xy`z+ …..)

This is called sum of products. The + is sum operator which is an OR gate. The product such as xy is an AND gate for the two inputs x and y.

Example: Minimize the following Boolean function using sum of products (SOP):

f(a,b,c,d) = Σ m(3,7,11,12,13,14,15)

=a`b`cd + a`bcd + ab`cd + abc`d`+ abc`d + abcd` + abcd

=cd(a`b` + a`b + ab`) + ab(c`d` + c`d + cd` + cd)

=cd(a`[b` + b] + ab`) + ab(c`[d` + d] + c[d` + d])

=cd(a`[1] + ab`) + ab(c`[1] + c[1])

=ab+ab`cd + a`cd

=ab+cd(ab` + a`)

=ab+ cd(a + a`)(a`+b`)

= ab + a`cd + b`cd

= ab +cd(a` + b`)

4.2.2.2 Schematic Design of Sum of Product

The expression of the sum of product executes two-level AND-OR design, and this design requires a collection of AND gates and one OR gate. Each expression of the sum of the product has similar designing.

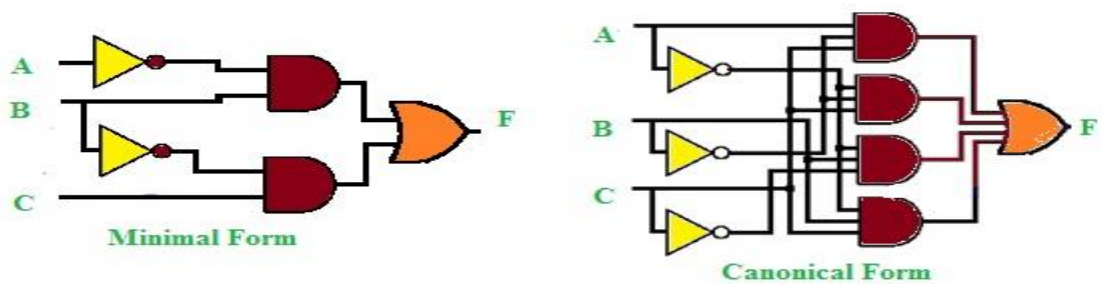

Figure 4.1: Schematic Design of SOP

The number of inputs and the number of AND gates depend upon the expression one is implementing. The design for a minimal sum of product & canonical expression using AND-OR gates is shown above.

4.2.3 Product of Sum (POS)

The short form of the product of the sum is POS, and it is one kind of Boolean algebra expression. In this, it is a form in which products of the dissimilar sum of inputs are taken, which are not arithmetic result & sum although they are logical Boolean AND & OR correspondingly. Before going to understand the concept of the product of the sum, we have to know the concept of the max term.

A **maxterm**, denoted as Mi, where $0 \leq i < 2n$, is a sum (OR) of the n variables (literals) in which each variable is complemented if the value assigned to it is 1, and uncomplemented if it is 0.

- 1-maxterms = maxterms for which the function F = 1.
- 0-maxterms = maxterms for which the function F = 0.

In the table 4.3, there are three inputs namely X, Y, Z and the combinations of these inputs are 8. Every combination has a max term that is specified with M. In max term, every input is complemented as it provides only ‘0’ while the stated combination is applied & complement of minterm is a max term.

M3 =m3’

(X’YZ)’ = M3

X+Y’+Z’=M3 (De Morgan’s Law)

Table 4.3: Truth table of Max Term

Sr. No.	X	Y	Z	F	Max Term (m)
1.	0	0	0	0	X+Y+Z = M0
2.	0	0	1	1	X+Y+Z’ = M1
3.	0	1	0	1	X+Y’+Z = M2
4.	0	1	1	1	X+Y’+Z’ = M3
5.	1	0	0	0	X’+Y+Z = M4
6.	1	0	1	1	X’+Y+Z’ = M5
7.	1	1	0	0	X’+Y’+Z = M6
8.	1	1	1	0	X’+Y’+Z’ = M7

4.2.3.1 Types of Product of Sum (POS)

The product of the sum is classified into three types which include the following.

- Canonical Product of Sums
- Non – Canonical Product of Sums
- Minimal Product of Sums

4.2.3.1.1 Canonical Product of Sum

The canonical POS is also named as a product of max term. These are AND jointly for which o/p is low or false. The expression this is denoted by ∏ and the max terms in the bracket are taken when the output is false. The truth table of the canonical product of sum is shown below.

Table 4.4: Truth table of the Canonical Product of Sum

Sr. No.	X	Y	Z	F
1.	0	0	0	0
2.	0	0	1	1
3.	0	1	0	1
4.	0	1	1	1
5.	1	0	0	0
6.	1	0	1	1
7.	1	1	0	0
8.	1	1	1	0

For the above table 4.4, the canonical POS can be written as

F = ∏ (M0, M4, M6, M7)

By expanding the above equation we can get the following function.

F = M0, M4, M6, M7

By substituting the max terms in the above equation we can get the below expression

F = (X+Y+Z) (X’+Y+Z)(X’+Y’+Z)(X’+Y’+Z’)

The product term of the canonical form includes both complemented and non-complimented inputs

4.2.3.1.2 Non-Canonical Product of Sum

The expression of the **product of sum (POS)** is not in normal form is named as non-canonical form. For example, let’s take the above expression

F = (X+Y+Z) (X’+Y+Z)(X’+Y’+Z)(X’+Y’+Z’)

F = (Y+Z) (X’+Y+Z) (X’+Y’+Z’)

Similar although reversed terms remove from two Max terms & forms only term to show it here is an instance.

= (X+Y+Z) (X’+Y+Z)
= XX’+XY+XZ+X’Y+YY+YZ+X’Z+YZ+ZZ
= 0+XY+XZ+X’Y+YY+YZ+X’Z+YZ+Z
= X (Y+Z) + X’ (Y+Z) + Y(1+Z) +Z
= (Y+Z) (X+X’) + Y (1) +Z
= (Y+Z) (0) +Y+Z
= Y+Z

The above final expression is still in the form of Product of Sum; however, it is in the form of non-canonical.

4.2.3.1.3 Minimal Product of Sum

This is the most simplified expression of the product of the sum, and it is also a type of non-canonical. This type of can is made simplified with the Boolean algebraic theorems although it is simply done by using K-map (Karnaugh map). This form is chosen due to the number of input lines & gates are used in this is minimum. It is profitably useful due to its solid size, quick speed, along with low manufacture price.

- The minimum product of sums (MPOS) of a function, f, is a POS representation of f that contains the fewest number of sum terms and the fewest number of literals of any POS representation of f.
- The zeros are considered exactly the same as ones in the case of sum of product (SOP)

Example:

f(a,b,c,d) = Π M(0,1,2,4,5,6,8,9,10)

= Σ m(3,7,11,12,13,14,15)

=[(a+b+c+d)(a+b+c+d`)(a+b`+c`+d`)(a`+b+c`+d`)(a`+b`+c+ d)(a`+b`+c+ d`)(a`+b`+c`+d)(a`+b`+c`+d`)]

4.2.3.2 Schematic Design of Product of Sum

The expression of the product of the sum executes two levels OR- AND design and this design requires a collection of OR gates and one AND gate. Each expression of the product of the sum has similar designing.

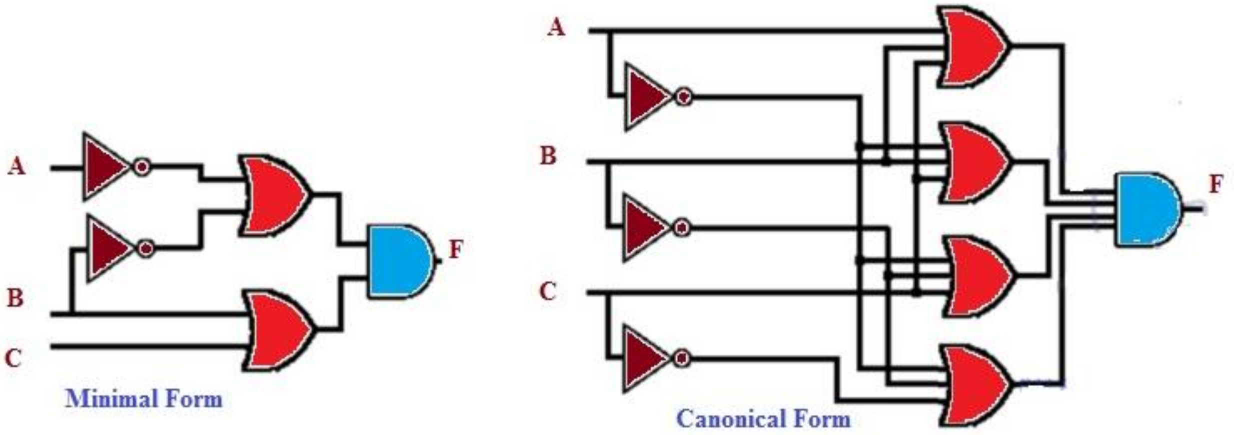

Figure 4.2: Schematic Design of POS

The number of inputs and the number of AND gates depend upon the expression one is implementing. The design for a minimal sum of product & canonical expression using OR-AND gates is shown above.

Thus, this is all about Canonical Forms: Sum of Products and Product of Sums, schematic design, K-map, etc. From the above information finally, we can conclude that a Boolean expression consists completely any of minterm otherwise maxterm is named as the canonical expression.

4.2.3.3 Conversion between Minterms and Maxterms

To convert from one canonical form to its other **equivalent** form, interchange the symbols Σ and Π, and list the index numbers that were excluded from the original form.

Example: 3-variable Minterms and Maxterms Conversion

Table 4.5: 3-Variable Truth table

Sr. #	Inputs			Output	
	x	Y	Z	F	F '
1.	0	0	0	0	1
2.	0	0	1	0	1
3.	0	1	0	0	1
4.	0	1	1	1	0
5.	1	0	0	0	1
6.	1	0	1	1	0

7.	1	1	0	1	0
8.	1	1	1	1	0

- F = x' y z + x y' z + x y z' + x y z
  = m3 + m5 + m6 + m7
  or
  F (x, y, z) = Σ(3, 5, 6, 7)

- F ' = x' y' z' + x' y' z + x' y z' + x y' z'
  = m0 + m1 + m2 + m4
  or
  F ' (x, y, z) = Σ(0, 1, 2, 4)

- F = (x+y+z) • (x+y+z') • (x+y'+z) • (x'+y+z)
  = M0 • M1 • M2 • M4
  or
  F (x, y, z) = Π(0, 1, 2, 4)

- F ' = (x+y'+z') • (x'+y+z') • (x'+y'+z) • (x'+y'+z')
  = M3 • M5 • M6 • M7
  or
  F ' (x, y, z) = Π(3, 5, 6, 7)

4.2.4 Logic Simplification using Boolean Laws

Here are some examples of Boolean Algebra simplifications. Each lines gives a form of the expression, and the rule or rules used to derive it from the previous one. Generally, there are several ways to reach the result.

4.2.4.1 Simplify: C+$\overline{BC}$

Expression	Rule(s) Used
C + $\overline{BC}$	Original Expression
C + ($\overline{B}$ + $\overline{C}$)	DeMorgan's Law.
(C + $\overline{C}$) + $\overline{B}$	Commutative, Associative Laws.
T + $\overline{B}$	Complement Law.
T	Identity Law.

4.2.4.2 Simplify: $\overline{AB}$ ($\overline{A}$ + B)($\overline{B}$ + B)

Expression	Rule(s) Used
$\overline{AB}$ ($\overline{A}$ + B)($\overline{B}$ + B)	Original Expression
$\overline{AB}$ ($\overline{A}$ + B)	Complement law, Identity law.
($\overline{A}$ + $\overline{B}$)($\overline{A}$ + B)	DeMorgan's Law

$\overline{A} + \overline{B}B$	Distributive law. This step uses the fact that or distributes over and. It can look a bit strange since addition does not distribute over multiplication.
$\overline{A}$	Complement, Identity.

4.3 Lab Activities

4.3.1 Task-1

Experimentally determine the truth table 4.6 and 4.7 for circuits shown in figure 4.5 and 4.6 with three input variables, and use DeMorgan's theorem to prove algebraically whether they are equivalent.

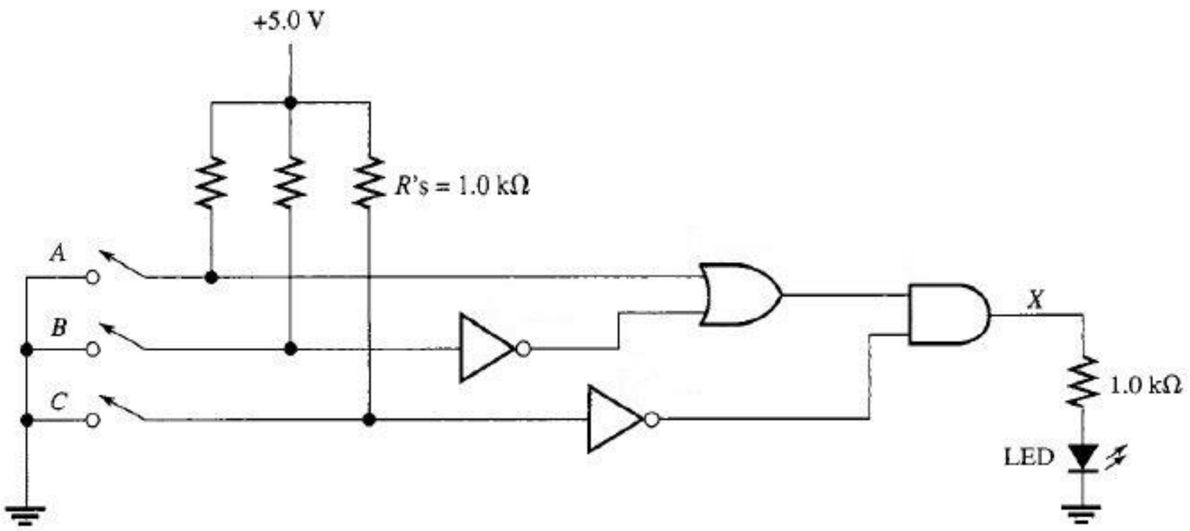

Figure 4.3

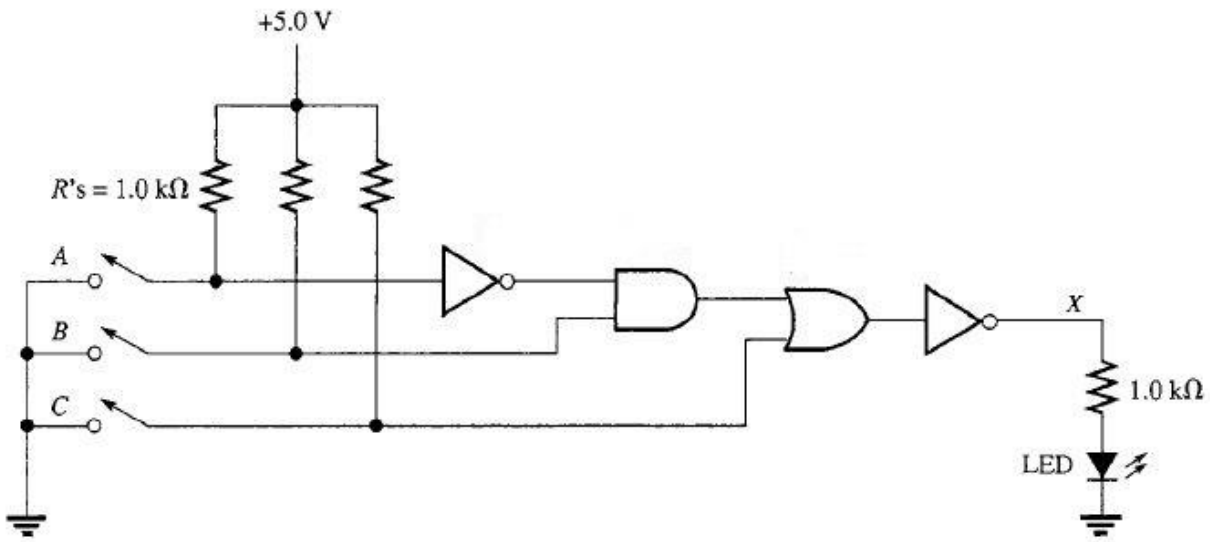

Figure 4.4

Table 4.6: Truth table for Figure 4.3

Sr. No.	Inputs			Output
	A	B	C	X
1.	0	0	0	
2.	0	0	1	
3.	0	1	0	
4.	0	1	1	
5.	1	0	0	
6.	1	0	1	
7.	1	1	0	
8.	1	1	1	

Table 4.7: Truth table for Figure 4.4

Sr. No.	Inputs			Output
	A	B	C	X
1.	0	0	0	
2.	0	0	1	
3.	0	1	0	
4.	0	1	1	
5.	1	0	0	
6.	1	0	1	
7.	1	1	0	
8.	1	1	1	

4.3.2 Task-2

Simplify the given expression to minimum number of literals using Boolean Algebra simplifications and draw logic diagram. Implement the resultant logic circuit and perform truth table based verification.

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

4.3.3 Task-3

Convert given expression into Sum of Minterms (SOM). Also convert it into Product of Maxterms. Implement resultant expression and perform truth table based verification of minterms and maxterms.

$$F(x,y,z) = x'y'z + x'z$$

Lab Exercise and Summary

Summary should cover Introduction, Procedure, Data Analysis and Evaluation.

Student’s Signature: ________________ **Date: _________________**

LABORATORY SKILLS ASSESMENT (Psychomotor)

Total Marks: 100

Criteria (Max Marks)	Level 1 0% ≤ S < 50%	Level 2 50% ≤ S< 70%	Level 3 70% ≤ S< 90%	Level 4 90%≤ S ≤100%	Score (S)
Procedural Awareness (20)	Selects inappropriate skills and/or strategies required by the task	Selects and applies appropriate skills and/or strategies required by the task with some errors	Selects and applies the appropriate strategies and/or skills specific to the task without significant errors	Selects and applies appropriate strategies and/or skills specific to the task without any error	
Practical Implementation (30)	Makes several critical errors in applying procedural knowledge of Boolean Algebra and Logic Simplification	Makes few critical errors in applying procedural knowledge of Boolean Algebra and Logic Simplification	Makes some non-critical errors in applying procedural knowledge of Boolean Algebra and Logic Simplification	Applies the procedural knowledge of Boolean Algebra and Logic Simplification in perfect ways	
Safety (10)	Requires constant reminders to follow safety procedures	Requires some reminders to follow safety procedures	Follows safety procedures with only minimal reminders	Routinely follows safety procedures	
Use of Tool/Equipment (20)	Uses tools, equipment and materials with limited competence	Uses tools, equipment and materials with some competence	Uses tools, equipment and materials with considerable competence	Uses tools, equipment and materials with a high degree of competence	
Participation to Achieve Group Goals (10)	Shows little commitment to group goals and fails to perform assigned roles	Demonstrates commitment to group goals, but has difficulty performing assigned roles	Demonstrates commitment to group goals and carries out assigned roles effectively	Actively helps to identify group goals and works effectively to meet them in all roles assumed	
Interpersonal Skills in Group Work (10)	Rarely interacts positively within a group, even with prompting	Interacts with other group members if prompted	Interacts with all group members spontaneously	Interacts positively with all group members and encourages such interaction in others	
				Marks Obtained	

Instructor's Signature: ________________ **Date: _________________**

LABORATORY SKILLS ASSESMENT (Affective)

Total Marks: 40

Criteria (Max. Marks)	Level 1 $0\% \leq S < 50\%$	Level 2 $50\% \leq S < 70\%$	Level 3 $70\% \leq S < 90\%$	Level 4 $90\% \leq S \leq 100\%$	Score (S)
Introduction (5)	Very little background information provided or information is incorrect	Introduction is brief with some minor mistakes	Introduction is nearly complete, missing some minor points	Introduction complete and well-written; provides all necessary background principles for the experiment	
Procedure (5)	Many stages of the procedure are not entered on the lab report.	Many stages of the procedure are entered on the lab report.	The procedure could be more efficiently designed but most stages of the procedure are entered on the lab report.	The procedure is well designed and all stages of the procedure are entered on the lab report.	
Data Record (10)	Data is brief and missing significant pieces of information.	Data provides some significant information and has few critical mistakes.	Data is almost complete but has some minor mistakes.	Data is complete and relevant. Tables with units are provided. Graphs are labeled. All questions are answered correctly.	
Data Analysis (10)	Data is presented in very unclear manner. Error analysis is not included.	Data is presented in ways (charts, tables, graphs) that are not clear enough. Error analysis is included.	Data is presented in ways (charts, tables, graphs) that can be understood and interpreted. Error analysis is included.	Data are presented in ways (charts, tables, graphs) that best facilitate understanding and interpretation. Error analysis is included.	
Report Quality (10)	Report contains many errors.	Report is somewhat organized with some spelling or grammatical errors.	Report is well organized and cohesive but contains some grammatical errors.	Report is well organized and cohesive and contains no grammatical errors. Presentation seems polished.	
				Marks Obtained	

LABORATORY SKILLS ASSESSMENT (Cognitive)

Total Marks: 10

(If any) **Marks Obtained**	

Instructor's Signature: ________________ **Date: __________________**