AMERICAN INTERNATIONAL UNIVERSITY–BANGLADESH (AIUB)
FACULTY OF SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING
DIGITAL LOGIC AND CIRCUITS LABORATORY

FALL 2023-2024
Section: Q Group: 2 Exp No: 2

LAB REPORT ON

Deriving logic equation and truth table from a given statement or expression and construction of combinational circuits.

Supervised By

DR. TANBIR IBNE ANOWAR

Submitted By:

Name	ID
NUSHRAT JAHAN	22-46149-1
S.M. MUJAHID SOUROV	22-49679-3
TRIDIB SARKAR	22-46444-1
MD. ATIK ISHRAK SUJON	22-46684-1
MD. FAHIM MURSHED	22-46695-1

Date of Submission: October 19th, 2023

TABLE OF CONTENTS

ABSTRACT:

This experiment is designed to help students implement the logic circuits derived from a given statement in the breadboard, using gate ICs and observe whether the output verifies the truth table of the given logic statement or not. This experiment performs relevant theoretical work by deriving the logic circuit and truth table from the given logic equation /statement and get familiarized with Boolean algebra and De Morgan's Law. Helps simplify the logic expressions with K-Map and verify accuracy by breadboard implementation.

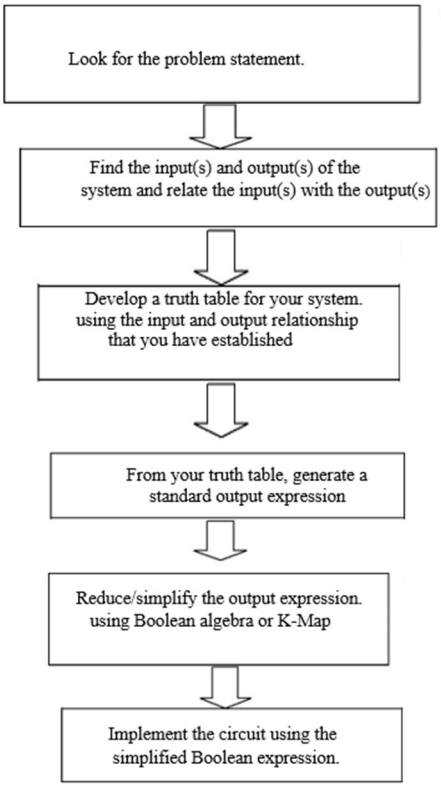
THEORY:

From any given logic statement, it is possible to construct a digital logic circuit. The first step in this process is to construct a truth table and then determine a standard SOP (Sum of product) or POS (product of sum). At the same time, it is also possible to derive a logic expression from a given combinational circuit diagram by observing the individual logic operations performed in the circuit and matching them with their corresponding logic gates. Expressions are simplified using Boolean algebra and De Morgan's law or K-Map to reduce the number of gates used. Then the circuit is implemented in the breadboard using gate ICs and observed whether the output verifies the truth table of the given statement.

This experiment shows the students a practical verification of deriving logic equations and truth table from combinational circuits. Knowing how to derive logic equations and truth tables from combinational circuits helps a person with detecting the output logic expressions from any unknown logic circuit.

Methodology:

Combinational circuits are built with logic gates and other components. It does not include any values to be taken from a previous state of the circuit. Designing such a combinational digital system requires use of the following methods:



Boolean algebra: Boolean Algebra is a mathematical system based on logic that utilizes a set of rules and laws to simplify and reduce complex Boolean expressions, enabling the analysis and optimization of digital gates and circuits with variables that can only take on the values of 0 or 1. [1]

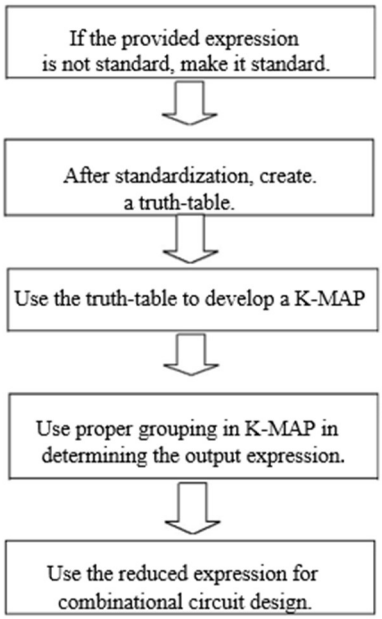
Variable: Boolean variables, such as A, B, and C, referred to as literals, can be represented by any symbol and can only have values of 0 or 1. [2]

Complement: The complement is defined as the inverse of a variable, which is represented by a bar over the variable. [1]

Sum term: The OR function, representing addition in Boolean Algebra, is called a sum term and is symbolized by the plus sign (+). [1]

Product term: The "product" function refers to the AND operation of terms with variables or constants, resulting in min-terms that cannot be further simplified, and various combinations of these can generate product results. [1]

Or if an expression is given, the following steps will help in designing the system.



Sum of Products (SOP):

When two or more product terms are summed by Boolean addition, the resulting expression is a sum of product. Implementing an SOP expression simply requires ORing the outputs of two or more AND gates. Product term is produced by an AND operation, and the sum (addition) of two or more product terms is produced by an OR operation. Therefore, an SOP expression can be implemented by AND-OR logic in which the outputs of a number (equal to the number of product terms in the
expression) of AND gates connect to the inputs of an OR gate. A standard SOP expression is one in which all the variables in the domain appear in each product term. Ex. Standard SOP expressions are important in constructing truth-tables and in Karnaugh map simplification method. The SOP expression is equal to 1 only if one or more of the product terms in the expression are equal to 1. [1]

Product of Sums (POS):

The OR function produces the logical sum of Boolean addition, and the AND function produces the logical sum of Boolean multiplication. But when dealing with combinational logic circuits in which AND gates, OR gates and NOT gates are connected, the expressions of Product-of-Sum are widely used. The Product of Sum (POS) expression comes from the fact that two or more sums (OR's) are added (AND 'ed) together. That is the outputs from two or more OR gates are connected to the input of an AND gate so that they are effectively AND' ed together to create the final (OR AND) output. [1]

Karnaugh Map:

Karnaugh map or K-map is a map of a function used in a technique used for minimization or simplification of a Boolean expression. It results in a smaller number of logic gates and inputs to be used during the fabrication. Booleans expression can be simplified using Boolean algebraic theorems but there are no specific rules to make the most simplified expression. However, K-map can easily minimize the terms of a Boolean function. Unlike an algebraic method, K-map is a pictorial method, and it does not need any Boolean algebraic theorems. K-map is basically a diagram made up of squares. Each of these squares represents a min-term of the variables. If n = number of variables, then the number of squares in its K-map will be 2n. K-map is made using the truth table. In fact, it is a special form of the truth table that is folded upon itself like a sphere. Every two adjacent squares of the k-map have a difference of 1-bit including the corners. Karnaugh map can produce Sum of product (SOP) or product of Sum (POS) expression considering which of the two (0,1) outputs are being grouped in it. The grouping of 0's result in Product of Sum expression & the grouping of 1's result in Sum of Product expression. The expression produced by K-map may be the most simplified expression but not unique. There can be more than 1 simplified expression for a single function, but they all perform the same. [1]

Apparatus:

- Digital trainer board.
- IC or Integrated Circuits:
    1. 7432 (1 pcs)
    2. 7408 (1pcs)
    3. 7404 (2pcs)
    4. 7402(1 pcs)
    5. 7400(1 pcs)
    6. 7486(1pcs)
- Connecting wires.
- LEDs
- Switches

Experimental Procedure:

a) Use output Y to form standard SOP expression.
b) Minimize the SOP expression using Boolean algebra and K-Map. Perform hardware implementation of the circuit and compare with your truth table output.

Problem Set:

Problem1: A Building has 4 floors which share the same water tank for water supply. To start the motor, each floor has a designated switch- Ground Floor with switch A, 1st Floor with switch B, 2nd Floor with switch C and 3rd Floor with switch D. The motor starts if someone presses the switch from the 3rd floor or from both ground and 2nd floor or from 1st and 2nd floor. Your job is to design the system.
a) Draw a truth table to represent the output Y. b) Use the truth table outputs to form standard SOP and POS expressions. c) Minimize the SOP expression using Boolean algebra and K-Map. Perform hardware implementation of the circuit and compare with your truth table output.

Problem II: For the active high input in 0,1,3,7,11,13,15 find the truth-table, reduced expression using K-MAP and the logic gate diagram.
a) Draw a step-by-step truth table to represent the outputs at each gate (1-6) and then the final output at Y. b) Use output Y to form standard SOP expression. c) Minimize the SOP expression using Boolean algebra and K-Map. Perform hardware implementation of the circuit and compare with your truth table output.

Problem 1.a) Truth table-

Ground Floor	1st Floor	2nd Floor	3rd Floor	Motor State
A	B	C	D	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

1.b) Expressions-
SOP = A'B'CD' + A'BCD + AB'CD' + AB'CD + AB'CD + ABC'D' + ABCD' + ABCD + ABD' + ABCD'

POS = (A+B+C+D') (A+B'+C+D) (A + B + C' + D) (A + B' + C' + D) (A + B + C' + D')

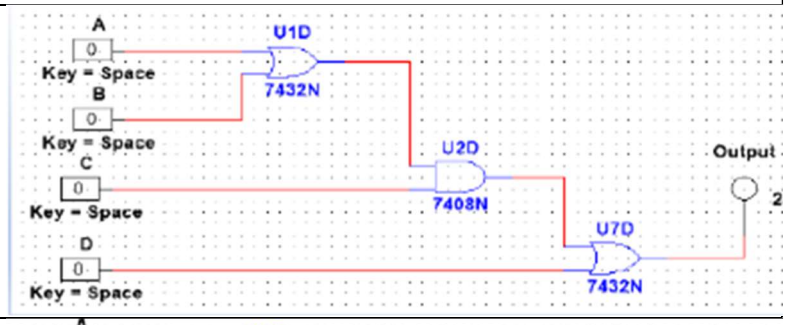
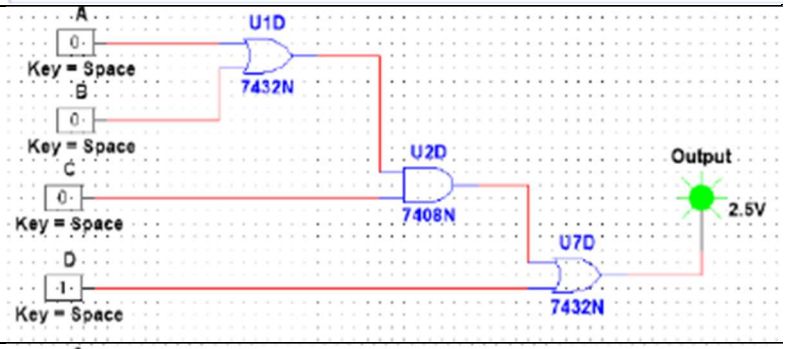
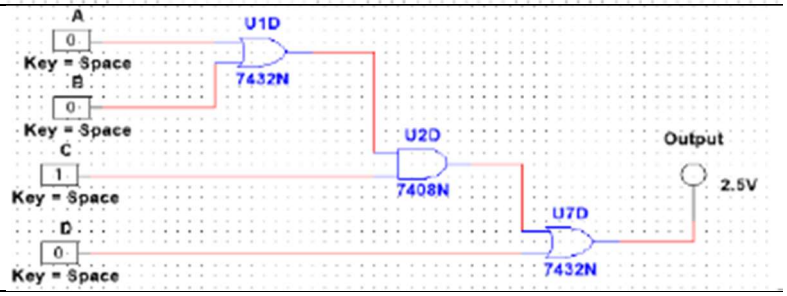
1.c) Minimizing the SOP expression –

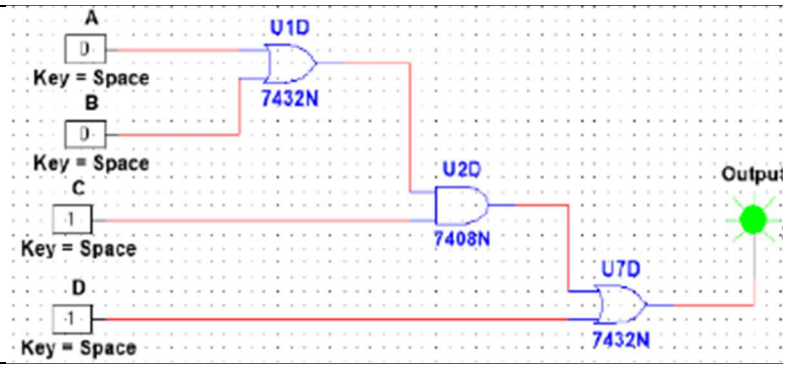
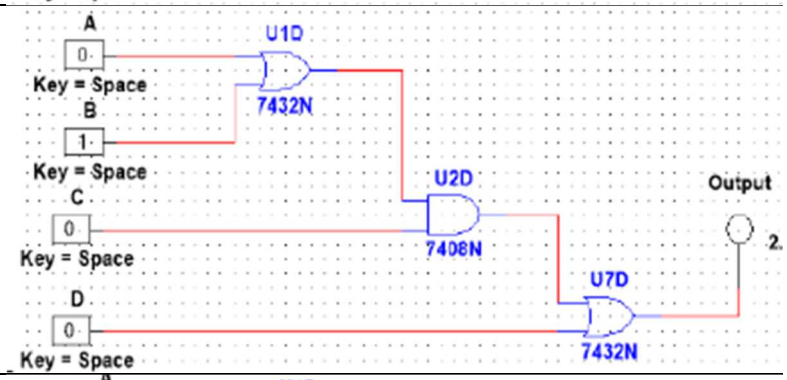
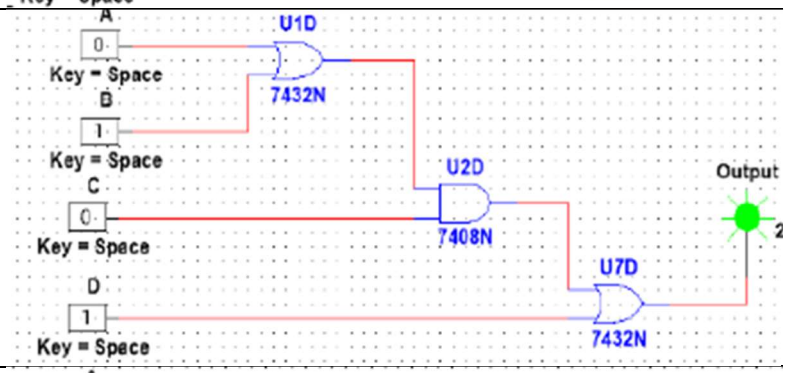
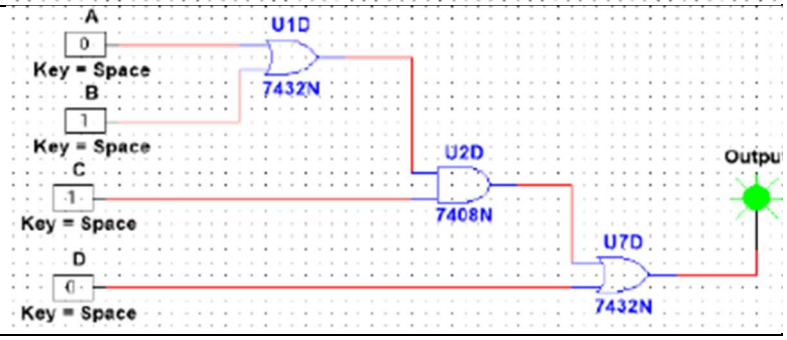
AB \ CD	00	01	11	10
00	0	1	1	0
01	0	1	1	1
11	0	1	1	1
10	0	1	1	1

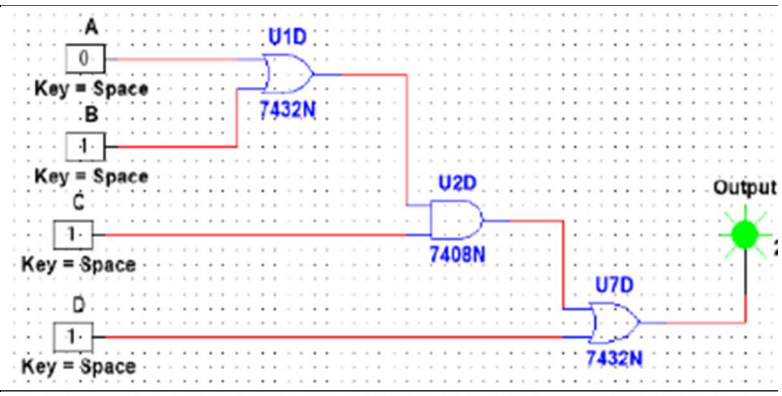
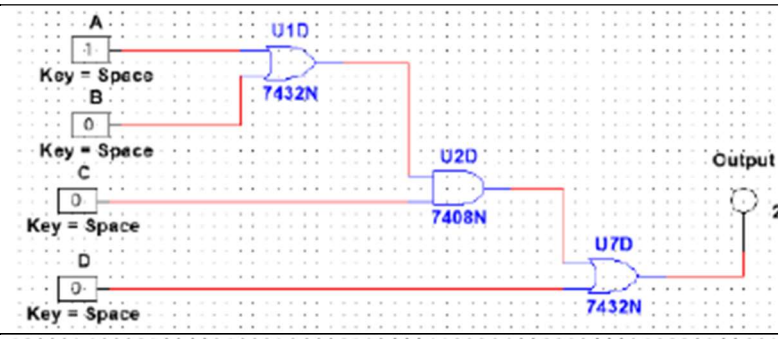
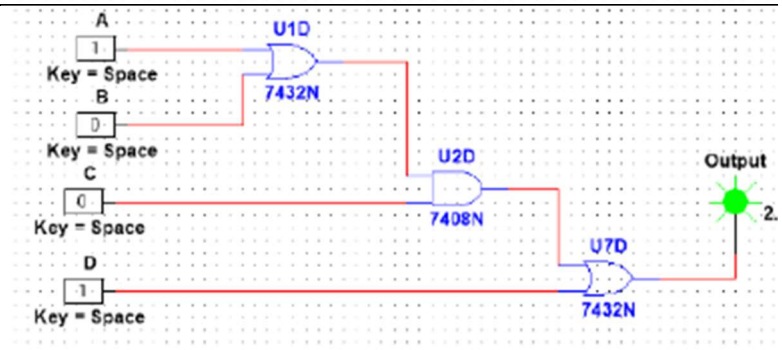
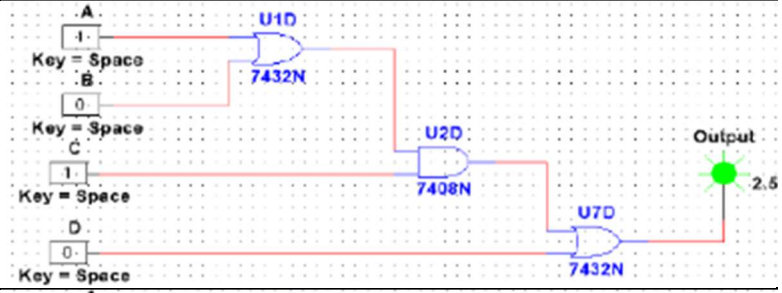
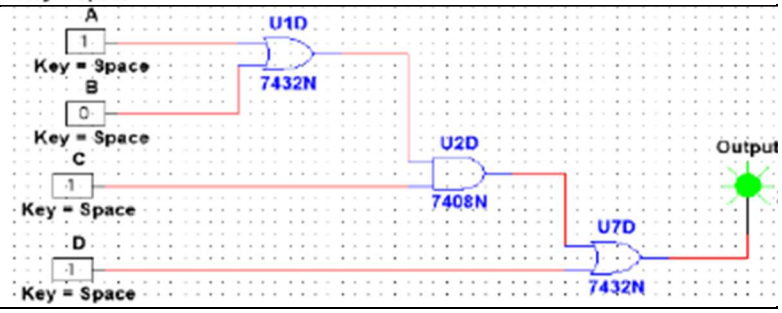
Minimized SOP = A'B' + A'CD + ABCD + ACD'

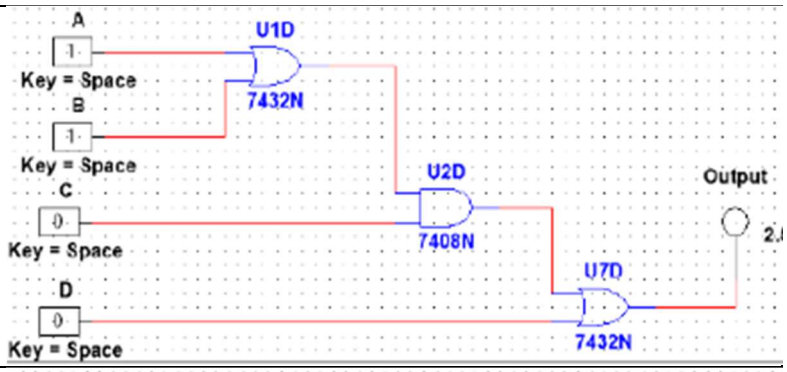
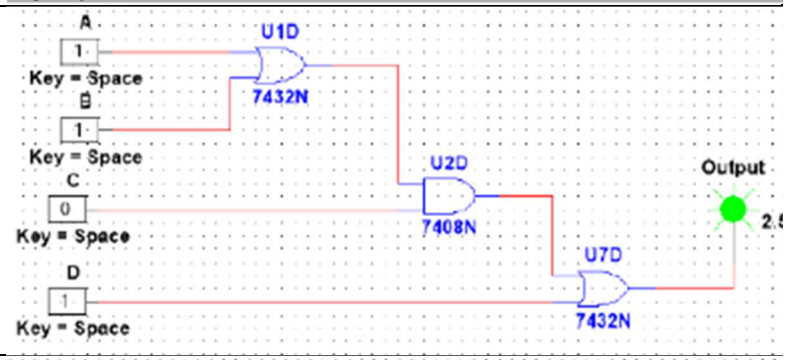
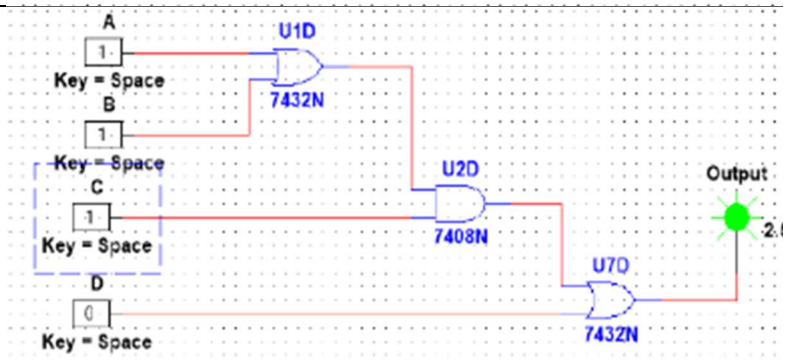
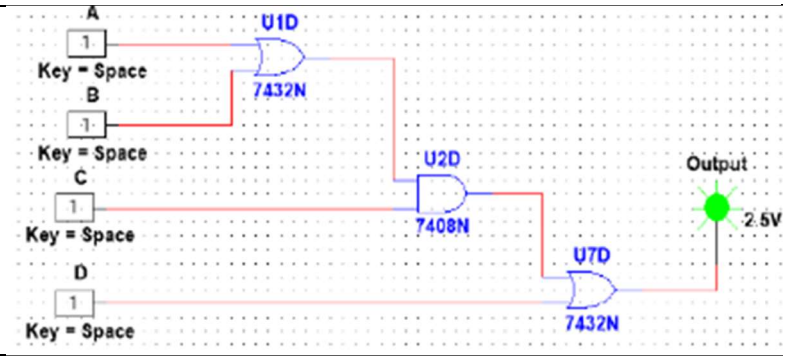
F=D+AC+BC

1.d) Simulation:

Ground Floor	1st Floor	2nd Floor	3rd Floor	Motor State	SIMULATION
A	B	C	D		
0	0	0	0	0	
0	0	0	1	1	
0	0	1	0	0	

0	0	1	1	1	
0	1	0	0	0	
0	1	0	1	1	
0	1	1	0	1	

0	1	1	1	1	
1	0	0	0	0	
1	0	0	1	1	
1	0	1	0	1	
1	0	1	1	1	

1	1	0	0	0	
1	1	0	1	1	
1	1	1	0	1	
1	1	1	1	1	

Problem 2.a) Truth table-

SL	A	B	C	D	F
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	1

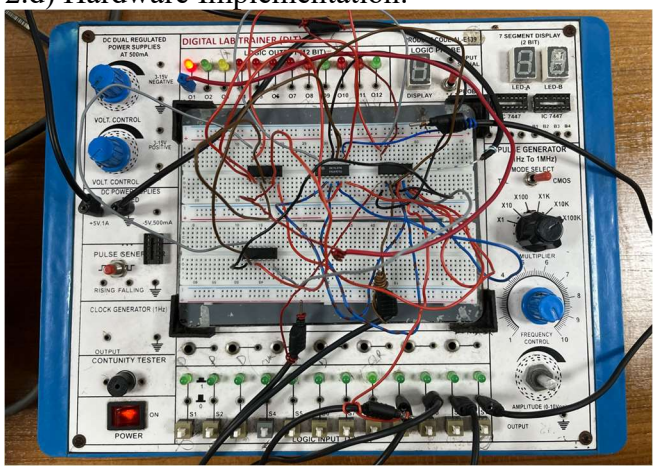
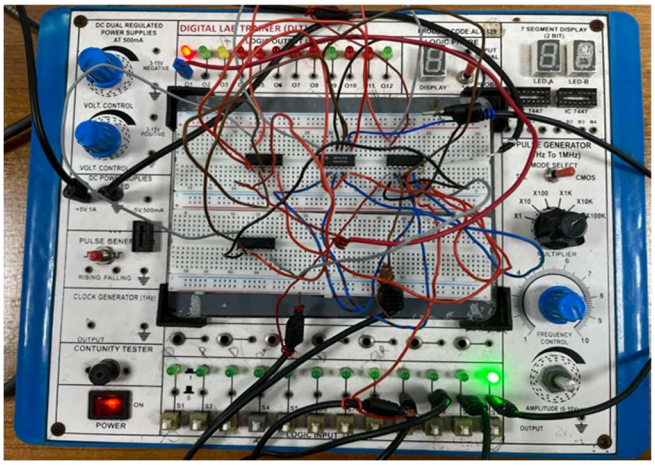
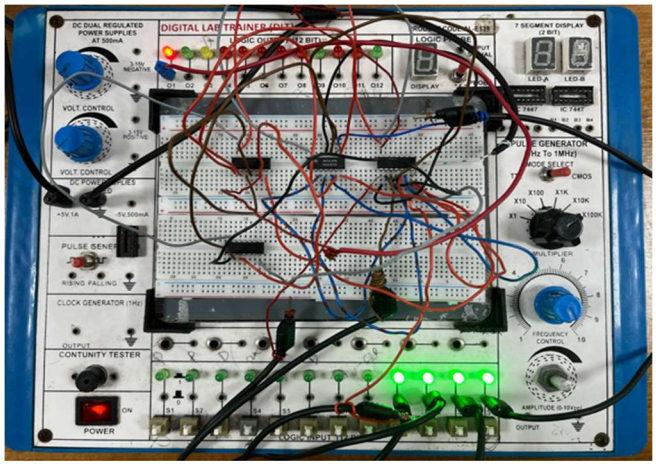
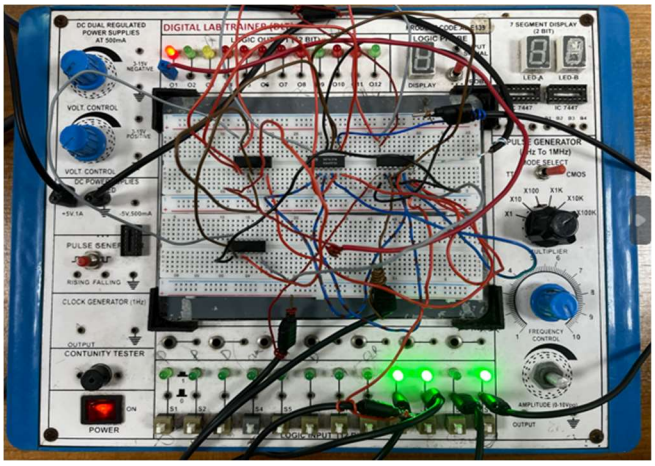
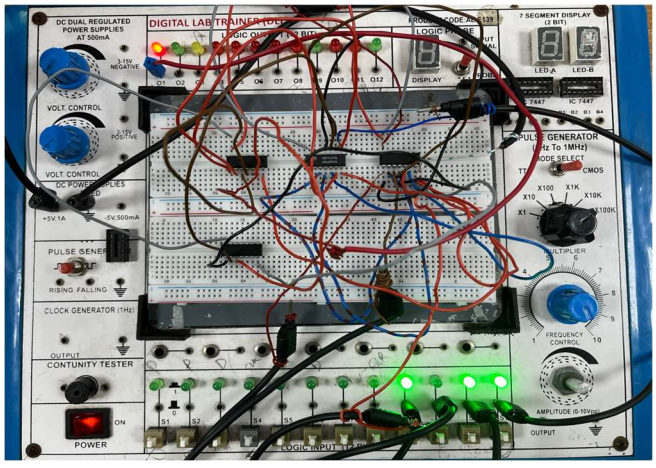
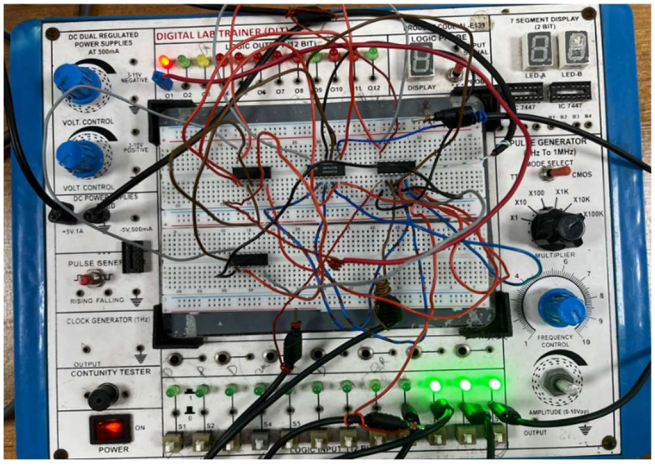
2.b) Expressions-
SOP = A'B'C'D' + A'B'C'D + A'B'CD + A'BCD + AB'CD + ABC'D+ABCD

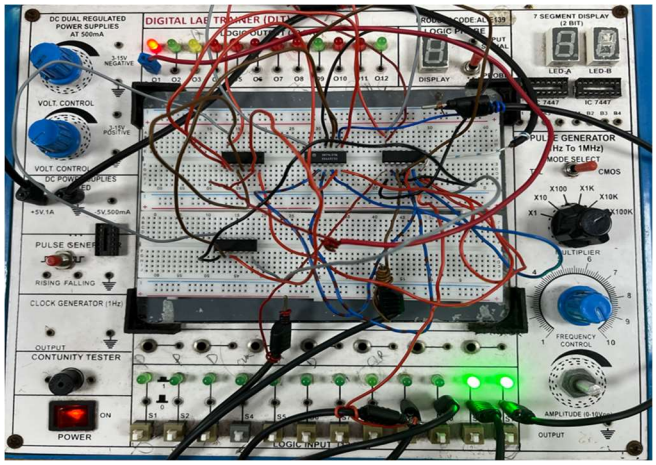
2.c) Minimizing the SOP expression –

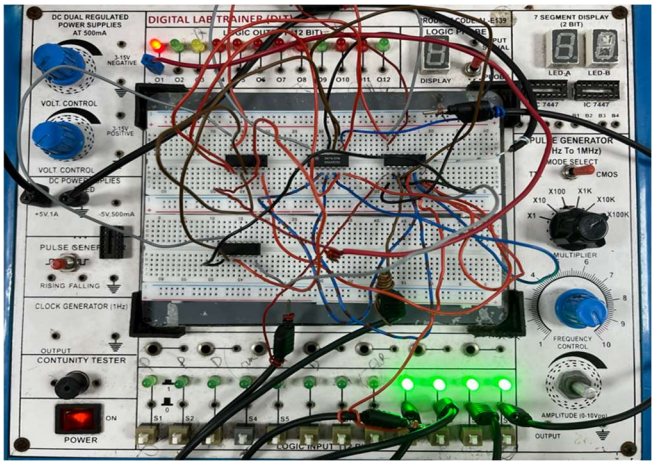
	CD				
AB		00	01	11	10
	00	1	1	1	0
	01	0	0	1	0
	11	0	1	1	0
	10	0	0	1	0

Minimized SOP =F= A'B'C'+ABD+CD

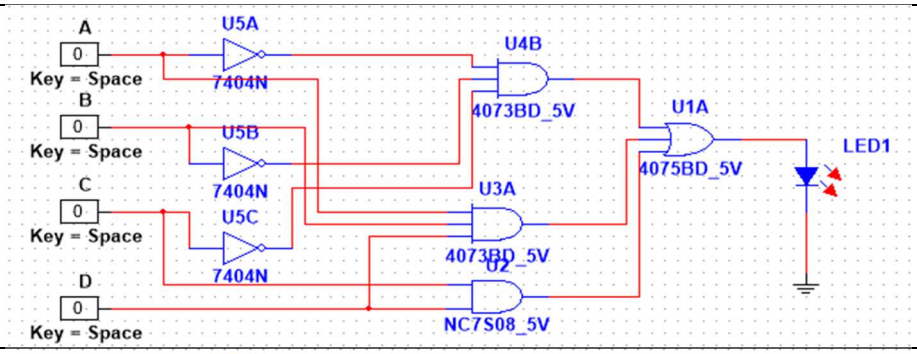
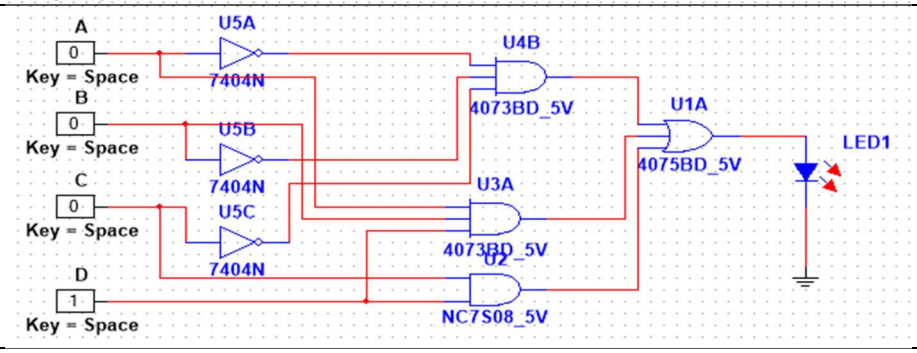
2.d) Hardware Implementation:

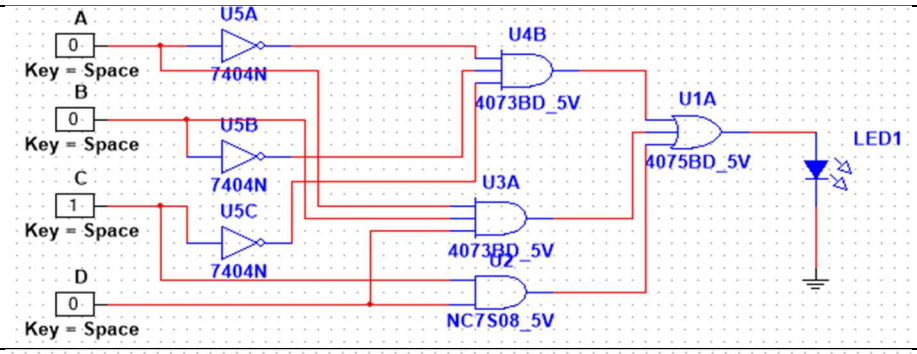
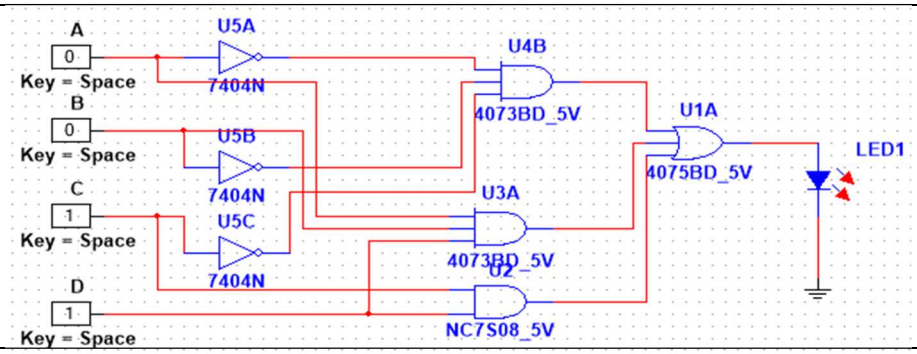
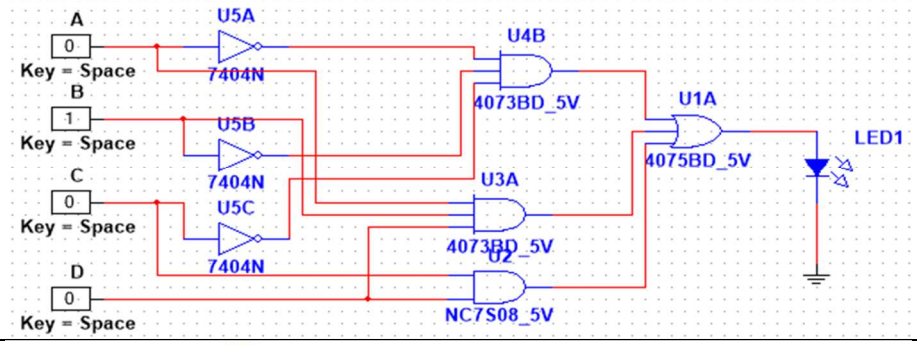
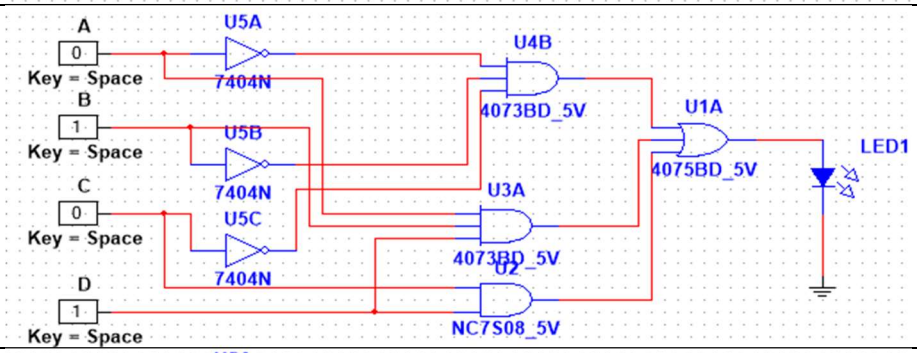
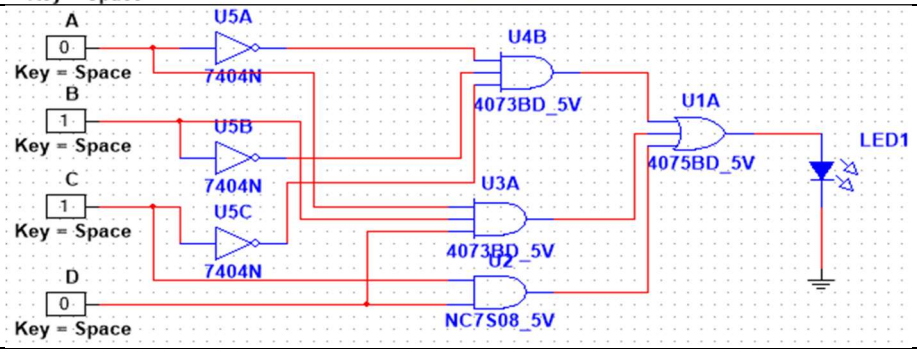







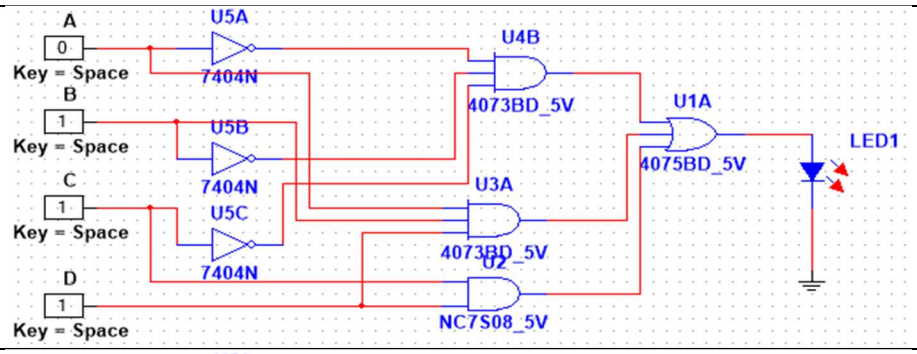
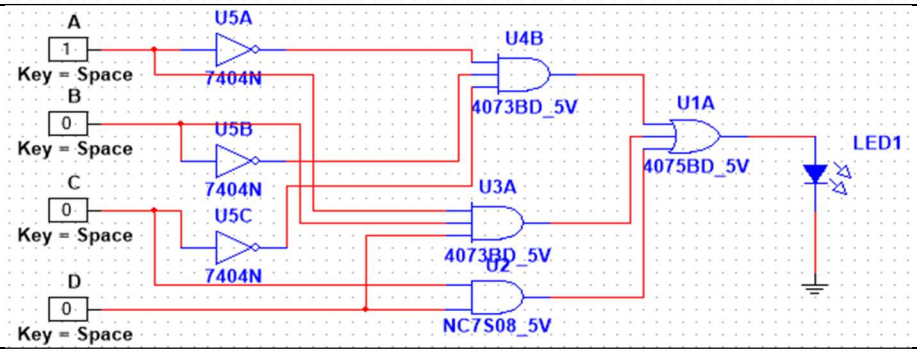
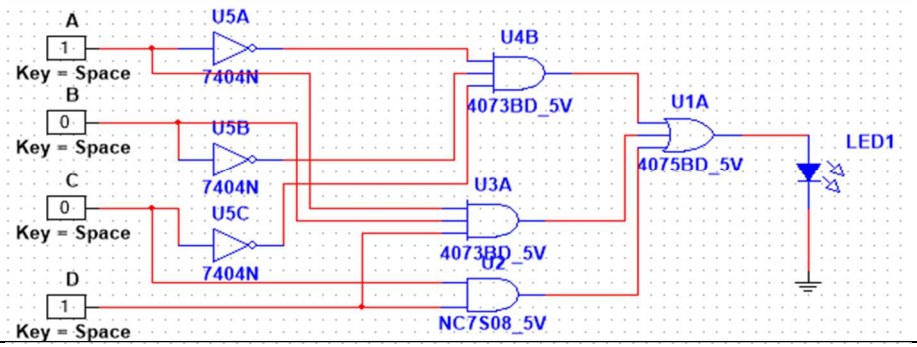
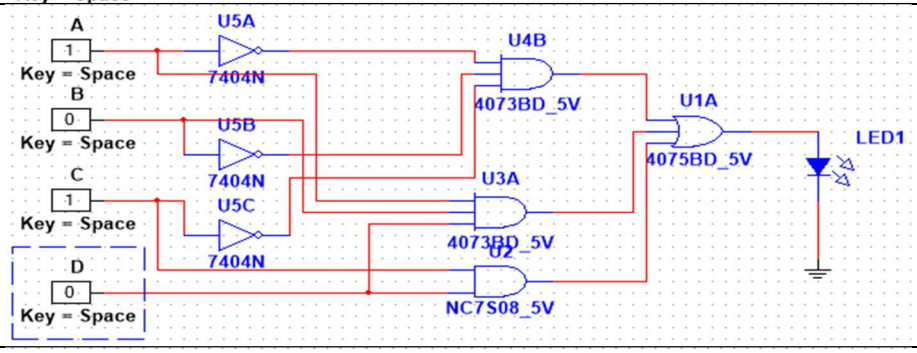
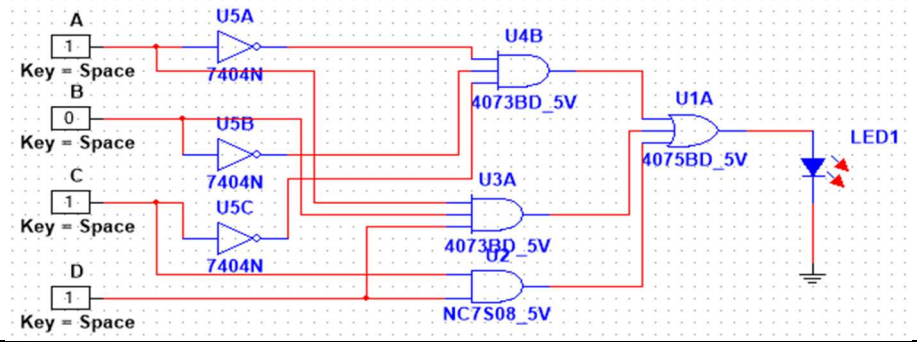


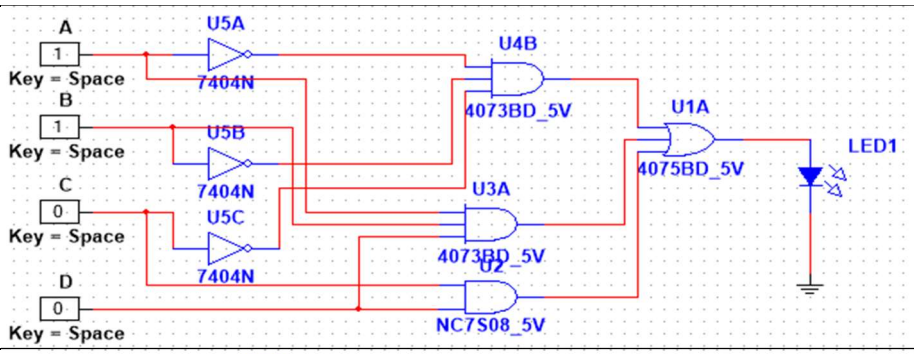
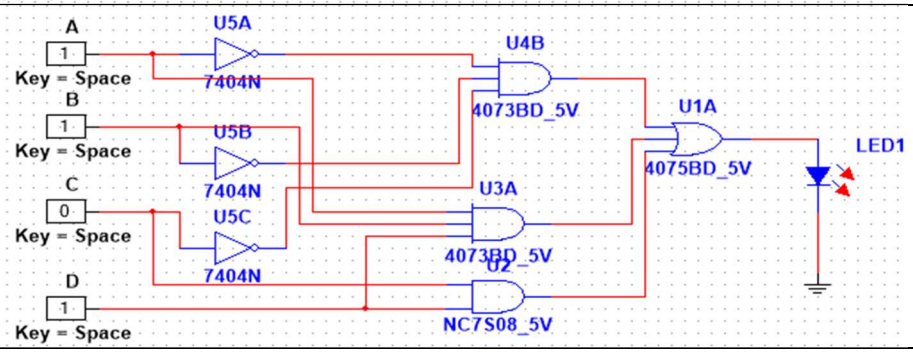
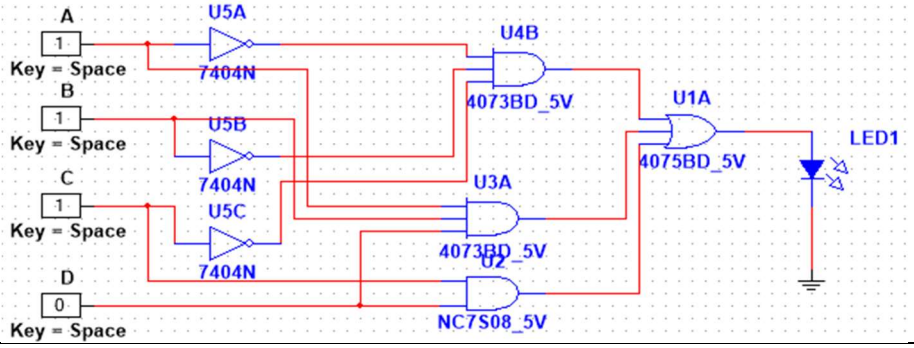
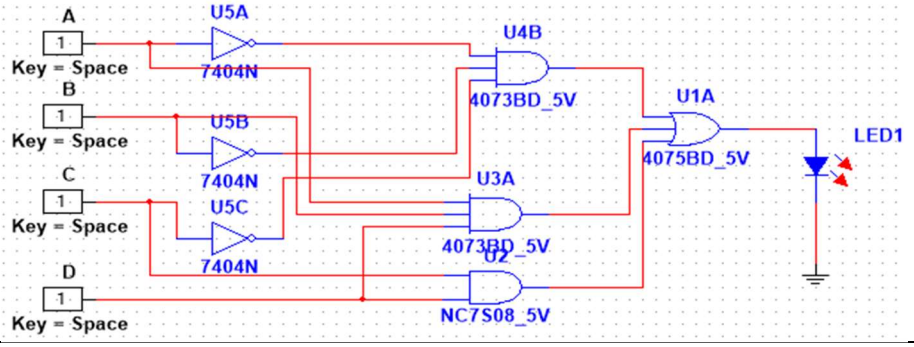


2.e) Simulation:

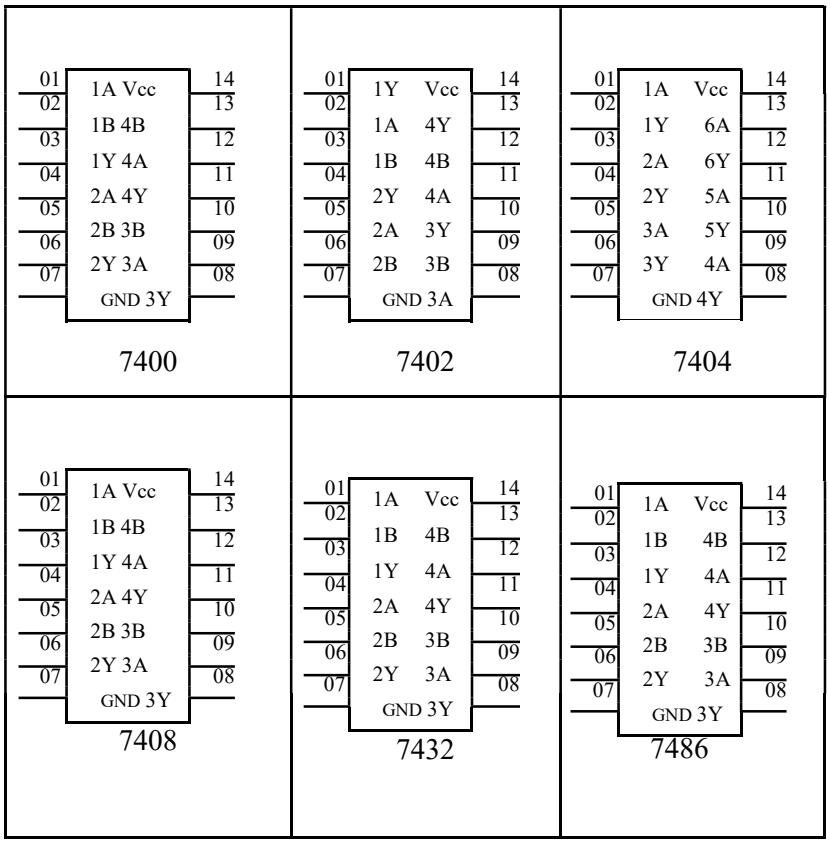
SL	A	B	C	D	F	Simulation
0	0	0	0	0	1	
1	0	0	0	1	1	

2	0	0	1	0	0	
3	0	0	1	1	1	
4	0	1	0	0	0	
5	0	1	0	1	0	
6	0	1	1	0	0	

7	0	1	1	1	1	
8	1	0	0	0	0	
9	1	0	0	1	0	
10	1	0	1	0	0	
11	1	0	1	1	1	

12	1	1	0	0	0	
13	1	1	0	1	1	
14	1	1	1	0	0	
15	1	1	1	1	1	

IC Configuration:



Report Questions:

1. Construct the derived equations (i) and (ii), using Universal gates (both NAND and NOR).
2. Develop the truth table for a certain three-input logic circuit with the output expression
   a. Y=ABC+(AB)'C+A'BC+AB'C+A(B'+C).
3. Implement the following logic expressions with logic gates Y=ABC+AB+AC

Report Answers:

Ans 1: **Implementation of Y = D + C (A + B) using NAND and NOR gate only -**

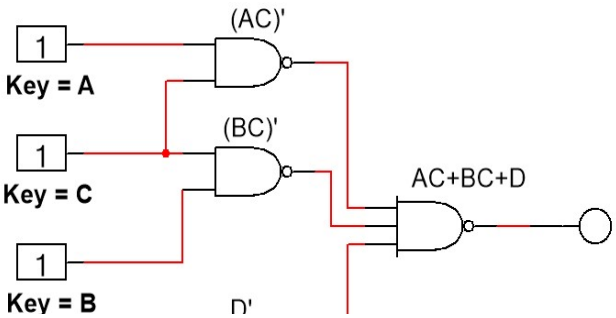


Figure 1: Implementation using NAND

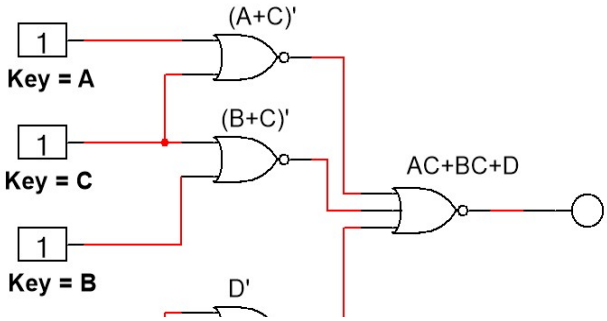


Figure 2: Implementation using NOR

Ans 2:

After simplifying the expression:

Y = ABC+(AB)'C+A'BC+AB'C+A(B'+C)
= ABC + (A'+B') C+ A'BC+ AB'C+AB'+AC
= ABC + A'C + B'C + A'BC+ AB'C+AB'+AC
= ABC+ A'BC + A'C+ AC+B'C+ AB'C+ AB'
= BC (A+A') + C (A'+A) + B'C(1+A) + AB'
= BC.1+ C.1+B'C.1+ AB' = BC+C+B'C+AB'
= C(B+1) + B'C+AB' = C.1+ B'C+AB'
= C(1+B') + AB'
= C+ AB'

Truth Table for the expression:

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Ans 3:

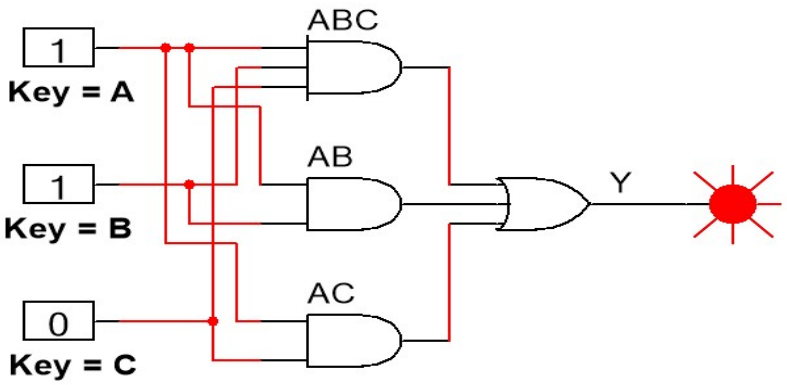


Figure 1:

Discussion:

The theory and expression part of the excrement were tricky to solve. A little mistake can lead to a big problem. During the experiment we encountered some difficulties such as bread board have some issue all the pin holes weren't working. The pin arrangement of all gates was not the same which also causes some issues but after checking the PIN arrangement Of the ICs twice before assembling the circuit will help to avoid this problem. Some gates also cause some issues, after changing the ICs solves the problem. Incorrect connections can also be very likely to happen as there are lots of connections to make. To solve the problem all the connections should be checked more than once.

Reference:

"Digital Fundamentals" by Thomas L. Floyd

- www.tutorialspoint.com
- www.electronics-tutorials.ws
- www.faculty.kfupm.edu.sa