North South University

Department of Electrical & Computer Engineering

LAB REPORT- 01

Course Code: CSE 231L

Course Title: Digital Logic Lab

Section: 08

Lab Number: 01

Experiment Name: Digital Logic Gates and Boolean Functions

Digital Logic Gates and Boolean Functions

Experiment Date: 13 February 2023

Date of Submission: 20 February 2023

Submitted by Group Number: 05

Group members:

Name	ID	Obtained Mark Simulation [5]	Obtained Mark Lab Report [15]
1. Sazid Hasan	2211513642		
2. Joy Kumar Ghosh	2211424642		
3. Syed Tashriful Alam	2212623042		
4. Md. Nifat Hossain	2212923042		
5. Md. Rafawat Islam	2122343642		

Course Instructor: Dr. Md. Abdur Razzak (Azz)

Submitted To: Prithika Dhar

Experiments-1 Name: Introduction to Basic Logic Gates

Objective:

- Knowing the logic gates ~~And~~ AND, OR, NOT, NAND, NOR, XOR and their truth table.
- Identifying an IC and its PIN order.
- How to test an IC working or not.

Apparatus:

- IC 7400 Quadruple 2-input NAND gates.
- IC 7402 Quadruple 2-input NOR gates.
- IC 7404 Hex Inverters (NOT gates).
- IC 7408 Quadruple 2-input ~~And~~ AND gates.
- IC 7432 Quadruple 2-input OR gates.
- IC 7486 Quadruple 2-input XOR gates.
- Trainer Board.
- Wires.

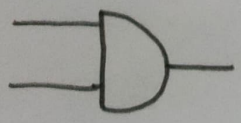
Theory:

Digital logic gates operates at two discrete voltage levels representing the binary values 0 (logical LOW) and 1 (Logical HIGH). A brief description of each gate is given below.

AND Gates:

A logical gate produces a ~~HIGH~~ HIGH output only when all of the inputs are HIGH.

Symbol:



Truth Table:

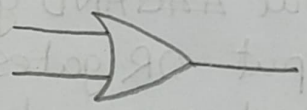
A	B	AND F=A·B
0	0	0
0	1	0
1	0	0
1	1	1

IC: 7408 Quadruple 2-input AND gates.

OR Gates:

A logic gate that produces a HIGH output when one or more inputs are HIGH.

Symbol:



Truth table:

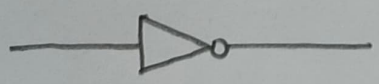
A	B	OR F=A+B
0	0	0
0	1	1
1	0	1
1	1	1

IC: 7432 Quadruple 2-input OR gates.

NOT Gates:

A logic gate that inverts or complements its input.

Symbol:



Truth table:

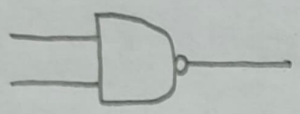
A	NOT $F=\bar{A}$
0	1
1	0

IC: 7404 Hex Inverters (NOT Gates)

NAND Gates:

A logic gate that produces a LOW output only when all the inputs are HIGH.

Symbol:



Truth Table:

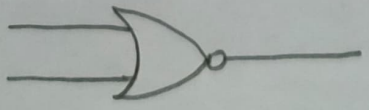
A	B	NAND $F=\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

IC: 7400 Quadruple 2-input NAND gates.

NOR Gates:

A logic gate in which the output is low when one or more inputs are HIGH.

Symbol:



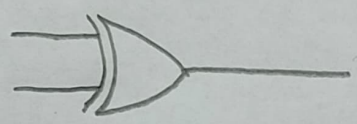
Truth Table:

A	B	NOR $F=\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

IC: 7402 Quadruple 2-input NOR gates.

XOR Gates:
A logical gate produces a High output only when its two inputs are at opposite levels.

Symbols:



Truth Table:

A	B	XOR $F=A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

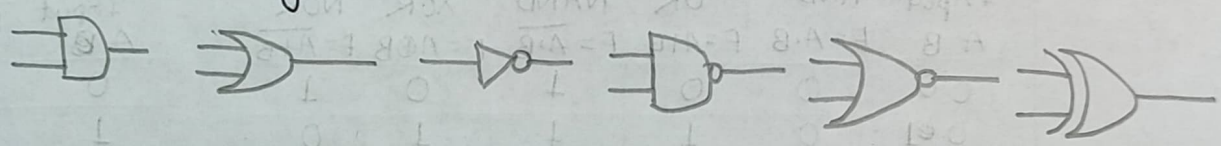
IC: 7486 Quadruple 2-input XOR gates

Identifying IC:
For IC numbers, we need to read the number of them, ignoring other letters. For example, 74HC04N is the 7404 Hex Inverter IC, where the HC denotes a high-speed TTL circuit CMOS variant.

Identifying IC PIN Numbers:

The basic rule for most ICs is that there is a polarity mark, such as the half-moon notch shown in the figure. Another standard polarity mark is a tiny dot, triangle or tab by pin 1. The rule is to move counter-clockwise around the chip from the polarity mark while numbering the pins starting at 1.

Circuit Diagram:



Experimental Procedure:

1. First, we place the 7408 AND IC on the breadboard as ~~very~~ every pin is in a separate node.
2. Then we connect the VCC and GND pins of the IC to the +5V and GND ports of the trainer board, respectively.
3. After that, we connect each input of the gate to a toggle ~~st~~switch and connect the output to an LED on the trainer board.
4. Then, we test all the combinations of inputs by the turning the toggle switches on (1) and off (0) and recording if the LED is on (1) or

off(0) as the output of the gate.

5. Then we replace the AND IC with OR, NOT, NAND, NOR and XOR ICs. Then we repeat step 4 for each ICs also maintain the input and output order by repeating step 3.

Simulation: N/A

Experimental Data Table:

Input A B	AND F=A·B	OR F=A+B	NAND F=$\overline{A \cdot B}$	XOR F=A⊕B	NOR F=$\overline{A+B}$
0 0	0	0	1	0	1
0 1	0	1	1	1	0
1 0	0	1	1	1	0
1 1	1	1	0	0	0

Input A	NOT F=$\overline{A}$
0	1
1	0

Results:

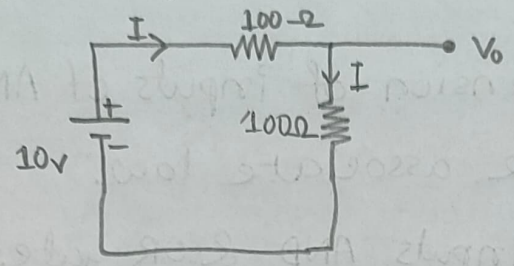
All the data are correct according to the truth table of each gate.

Questions & Answers (Q/A):

1. we need,
   7408 Quadruple 2-input AND gates IC - 5x
   7404 Hex Inverters (NOT gates) IC - 2x
   7402 Quadruple 2-input NOR gates IC - 4x

2. If the +5V port is not working, we can design a +5V power supply very easily, as below. We only require a 10V battery and two 100 Ohm

resistors so that the output voltage is +5V. This way, we can design a voltage divider network, as shown below.



Discussion:

from the first test run, our ICs were working perfectly. Then we tried to make NAND Gate by combining the AND and NOT gates ICs. But we couldn't do that; then we figure out that the breadboard was not tight enough to connect all the pins of ICs. Then for faster work, we bring NAND IC and complete out other test. Finally, we completed all our test runs ~~data~~ within the time.

Experiments-2 Name: Constructing 3-input AND & OR gates from 2-input AND & OR.

Objectives:

- Prove the extension of inputs of AND and OR gates using the associate law.
- Making three inputs AND & OR gates using two inputs AND & OR gates ICs.

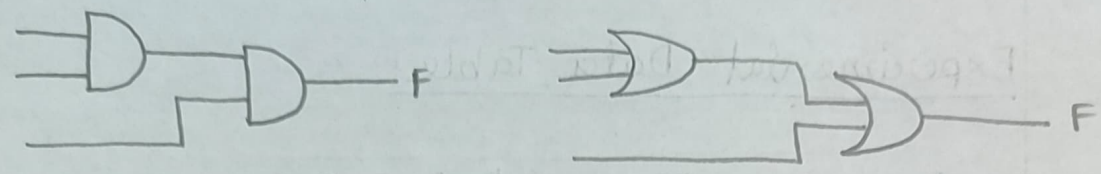
Apparatus:

- IC 7408 Quadruple 2-input AND gates.
- IC 7432 Quadruple 2-input OR gates.
- Trainer Board.
- Wires.

Theory:

We know that 7408 IC has four AND gates with two inputs. But if we need to take three inputs, we can combine two AND gates of this IC. Just connect the output pin with an input pin of another gate using a wire. Then connect the third input to the second input pin of the second gates. Then the output pin of the second gate will give us the output of three inputs AND gates.

Circuit Diagram:



Experimental Procedure:

1. First, we complete the truth table for the 3-input AND gate in Truth table.
2. Then we place the 7408 AND IC on the breadboard as every pin is in a separate node, and then we connect the VCC and GND pins of the IC to the +5V and GND ports of the trainer board, respectively.
3. After that, we connect two inputs in PINs 1 & 2. Then we connect the output pin 3 with input pin 4. After that, we connect the third ~~pin~~ input to the input pin 5.
4. Then we connect the output pin to the the LED.
5. Then, we test all the combinations of inputs by turning the toggle switches on (1) & off (0) & recording if the LED is on (1) or off (0) as the output of the gate. Then match the output to the truth table.
6. Then we replace the AND IC with OR IC and repeat step 5.

Simulation:

Attached.

Experimental Data Table:

A	B	C	F = ABC	F = A+B+C
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Results:

All the data table's output ~~red~~ results match the truth table of three inputs AND gate. Also, matched with OR gate.

Question and Answers (Q'/A):

N/A

Discussion:

After the first experiment, we were so excited to build our first circuit. Then we make a three-input AND gate using the two-input AND gate very quickly. We don't face any difficulty in this experiment.

Experiment-3 Name: Implementation of Boolean functions

Objectives:

- Get acquainted with the representation of Boolean functions using truth table, logic diagrams and Boolean Algebra.
- Become familiar with combinational logic circuits.

Apparatus:

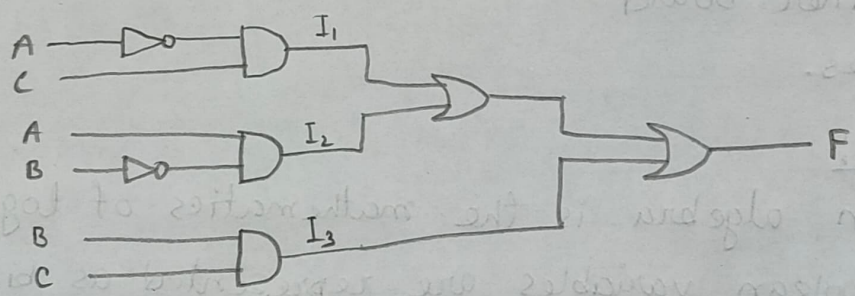
- IC 7408 Quadruple 2-input AND gates.
- IC 7432 Quadruple 2-input OR gates.
- IC 7404 Hex Inverters (NOT gates)
- Trainer Board
- Wires.

Theory:

Boolean algebra is the mathematics of logic circuits. The Boolean variables are represented as binary numbers to represents truth: 1 = true & 0 = false. Elementary algebra deals with numerical operations, whereas Boolean algebra deals with logical operations. Some Postulates and Theorems are given below:

Postulates & Theorems		Name
$A+0=A$	$A\cdot 1=A$	Identity
$A+A'=1$	$A\cdot A'=0$	
$A+A=A$	$A\cdot A=A$	
$A+1=1$	$A\cdot 0=0$	
$(A')'=A$		Involution
$A+B=B+A$	$AB=BA$	Commutative
$A+(B+C)=(A+B)+C$	$A(BC)=(AB)C$	Associative
$A(B+C)=AB+AC$	$A+BC=(A+B)(A+C)$	Distributive
$(A+B)'=A'B'$	$(AB)'=A'+B'$	De Morgan
$A+AB=A$	$A(A+B)=A$	Absorption

Circuit Diagram:



Experimental Procedure:

Consider the following Boolean Equation:

$$F = A'C + AB' + BC$$

1. Then we complete the truth table for the implicants

   $I_1 = A'C$, $I_2 = AB'$, $I_3 = BC$

2. Then, we make three input branches separately for A, B & C to take multiple inputs from one input switch.

3. then we implicant I1. First, we take an input from A and connect it with NOT gate PIN-1 & connect the output from PIN-2 to the AND gate PIN-1. Then we connect another input from C with the AND gate PIN-2. And connect ~~with~~ the output pin of AND gate PIN-3 to a led for testing purposes. After a successful test, we connect the output with OR gate PIN-1.

4. After that, we implemented I2. First we take input from A and connect with AND gate input PIN-4. Then we take another input from PIN-4 and connect with AND gate input PIN-5. Then we take the output from PIN-6 and connect it with LED for testing purposes. After a successful test, we connect the output with OR gate input PIN-2

5. After that, we implemented I3. First, we take two inputs from B and C and connect it with ~~a LED~~ AND gate input pins 9 and 10. Then take the output from PIN-8 and connect it with a LED for testing purposes. After a successful test, we connect the output ~~the~~ with OR gate input PIN-5.

6. We connect the OR gate output PIN-3 with input PIN-4. Then we connect the output PIN-6 with an LED. And test for every possible combination and varify with the truth table.

Simulation:

Attached.

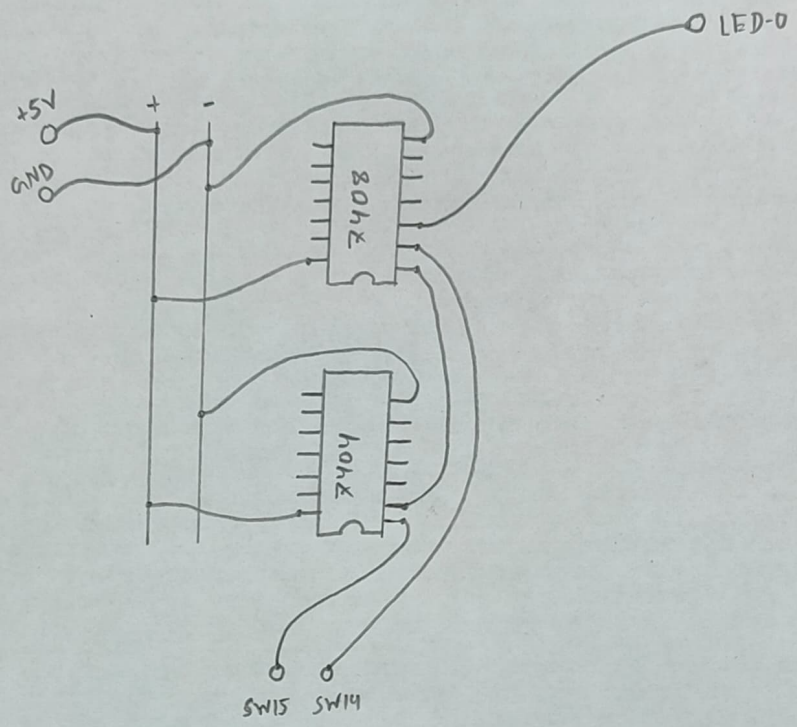
Experimental Data Table:

A	B	C	$I_1 = A'C$	$I_2 = AB'$	$I_3 = BC$	$F = I_1 + I_2 + I_3$
0	0	0	0	0	0	0
0	0	1	1	0	0	1
0	1	0	0	0	0	0
0	1	1	1	0	1	1
1	0	0	0	1	0	1
1	0	1	0	1	0	1
1	1	0	0	0	0	0
1	1	1	0	0	1	1

Results:

All data collected from step 6 are accurate with the truth table of this Boolean function. And we completed this experiment successfully.

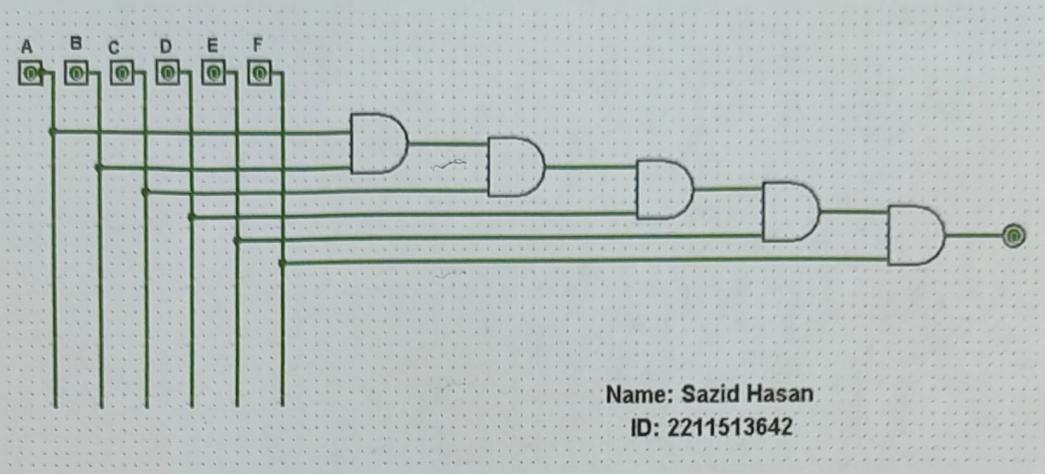
Questions & Answer (Q/A):



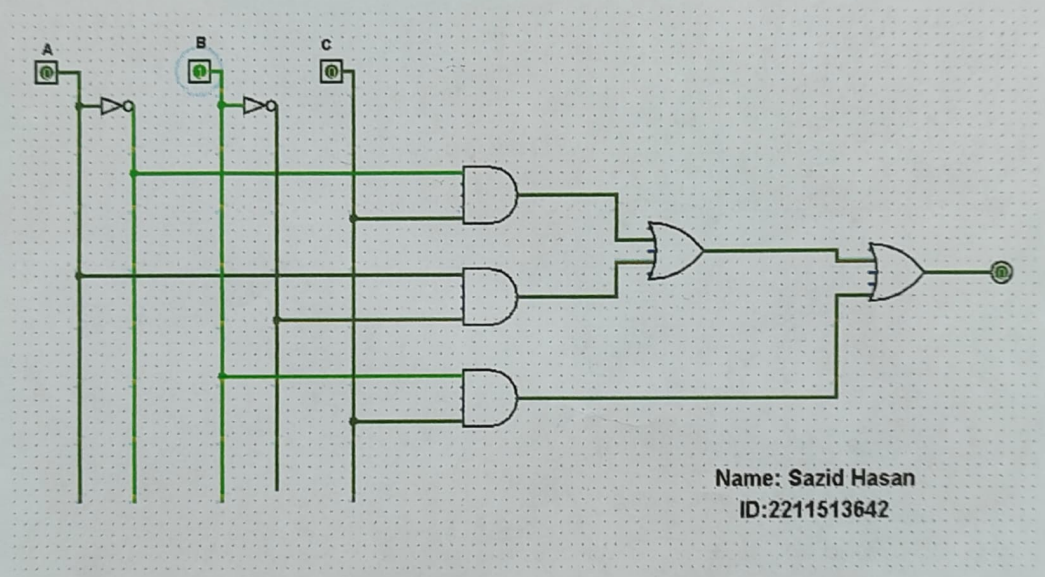
Discussion:

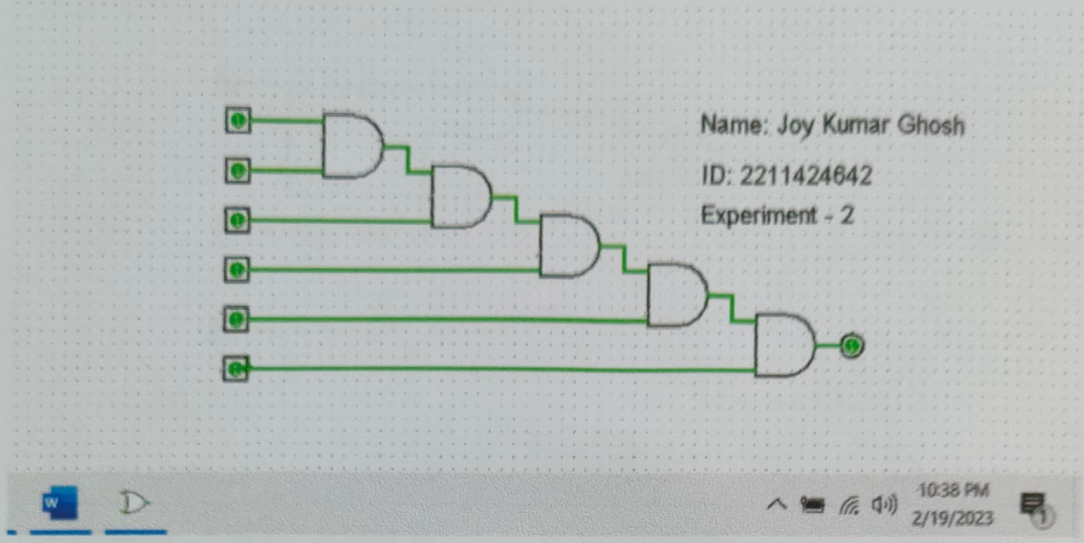
When we are doing step 6, we face some difficulties. The result was different from the truth table at the first output. Then we tried to figure out why it was not working and checked the wire connection from step 2. And we found that in the OR gate, we give the wrong input in VCC and GND. Then we reverse the inputs then test again. And then it's working perfectly.

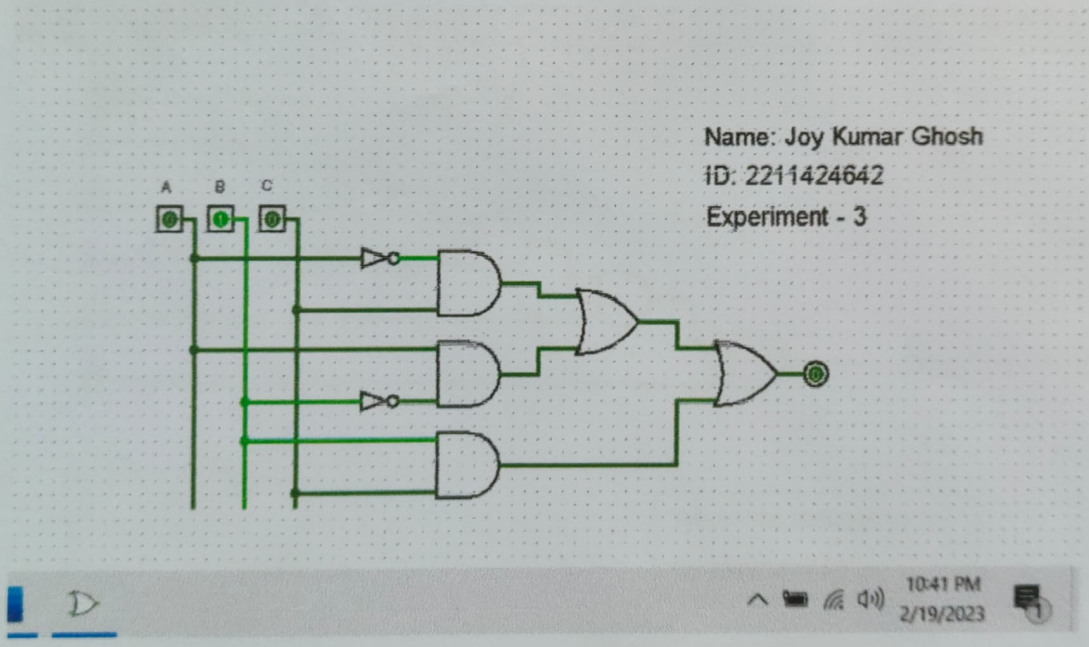
Experiment-02:

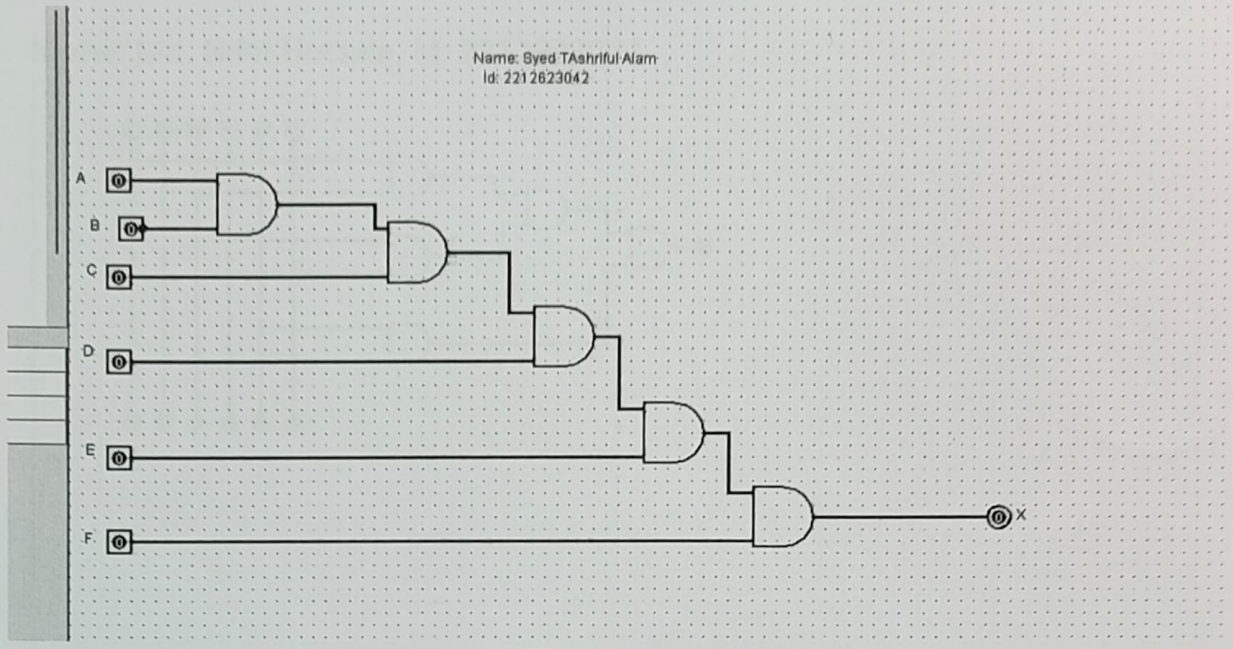


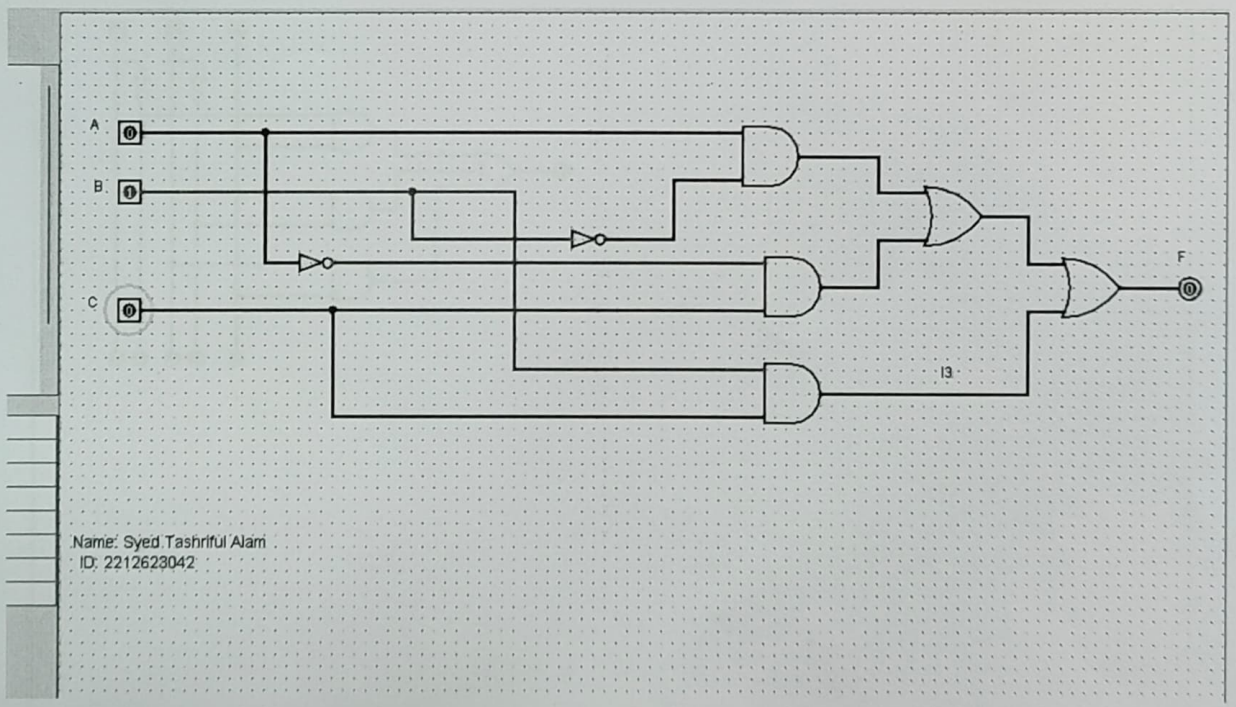
Experiment-03:



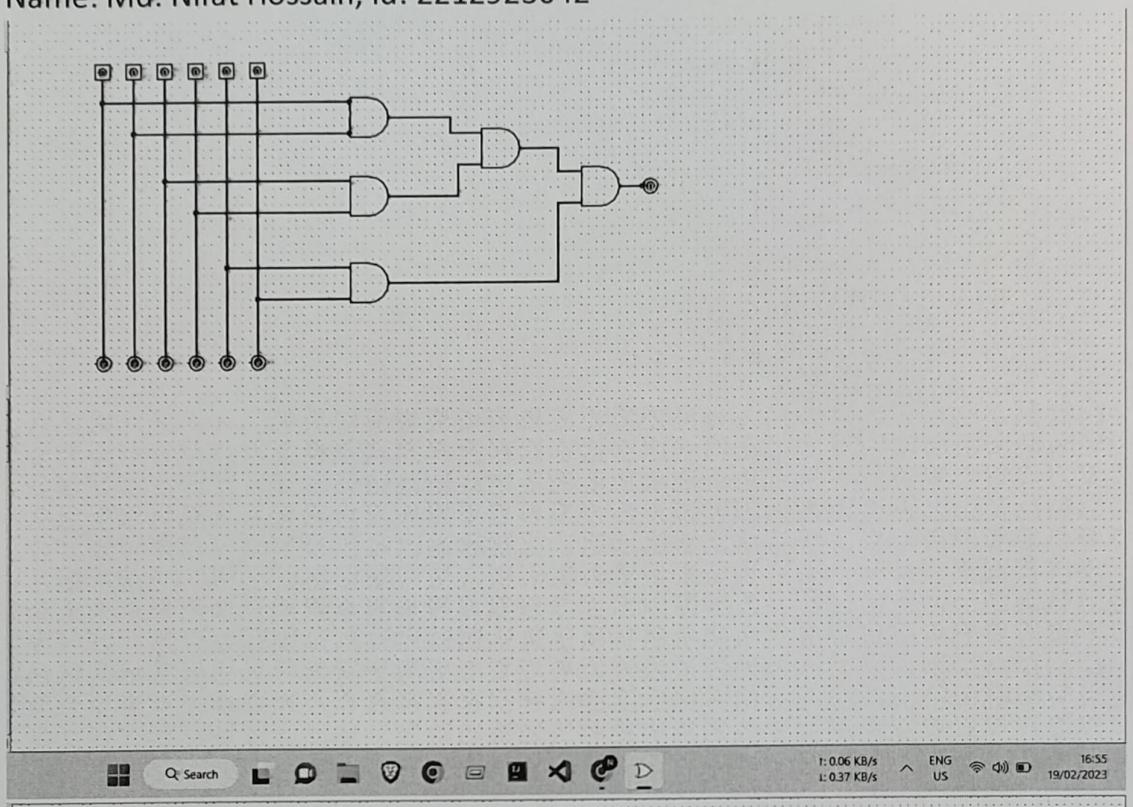

Name: Joy Kumar Ghosh
ID: 2211424642
Experiment - 2
10:38 PM
2/19/2023

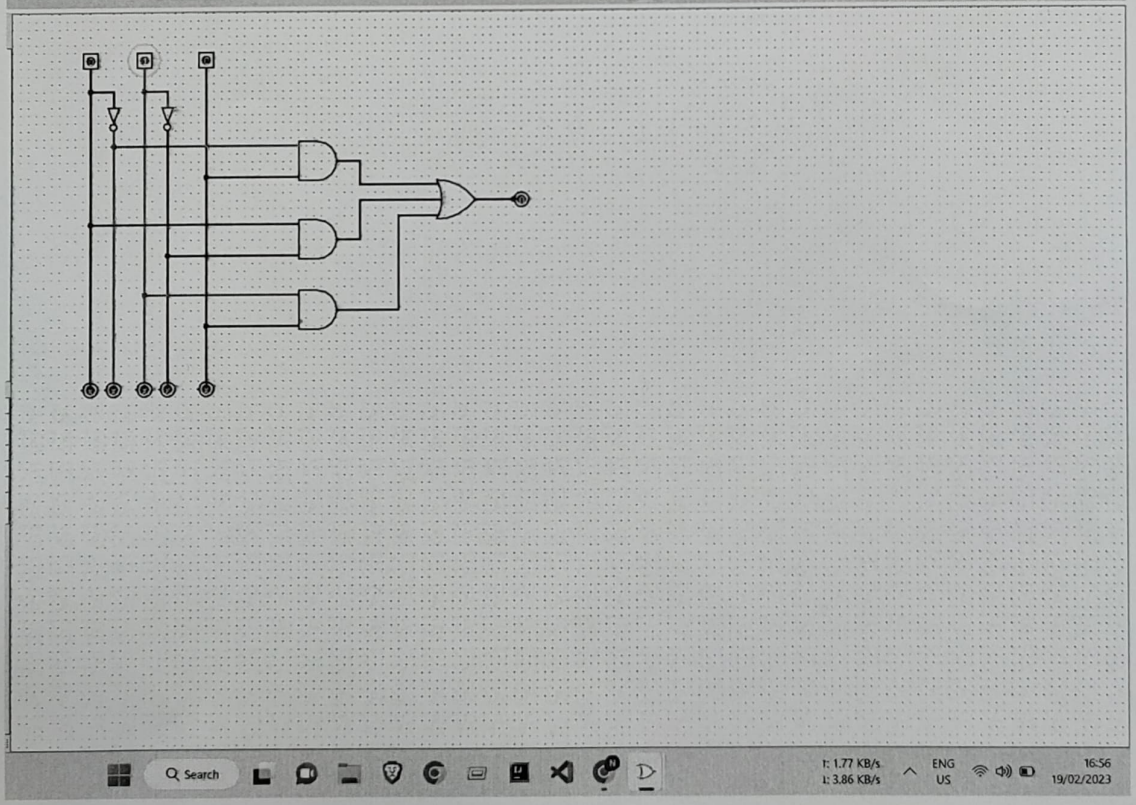

A
B
C
Name: Joy Kumar Ghosh
ID: 2211424642
Experiment - 3
10:41 PM
2/19/2023

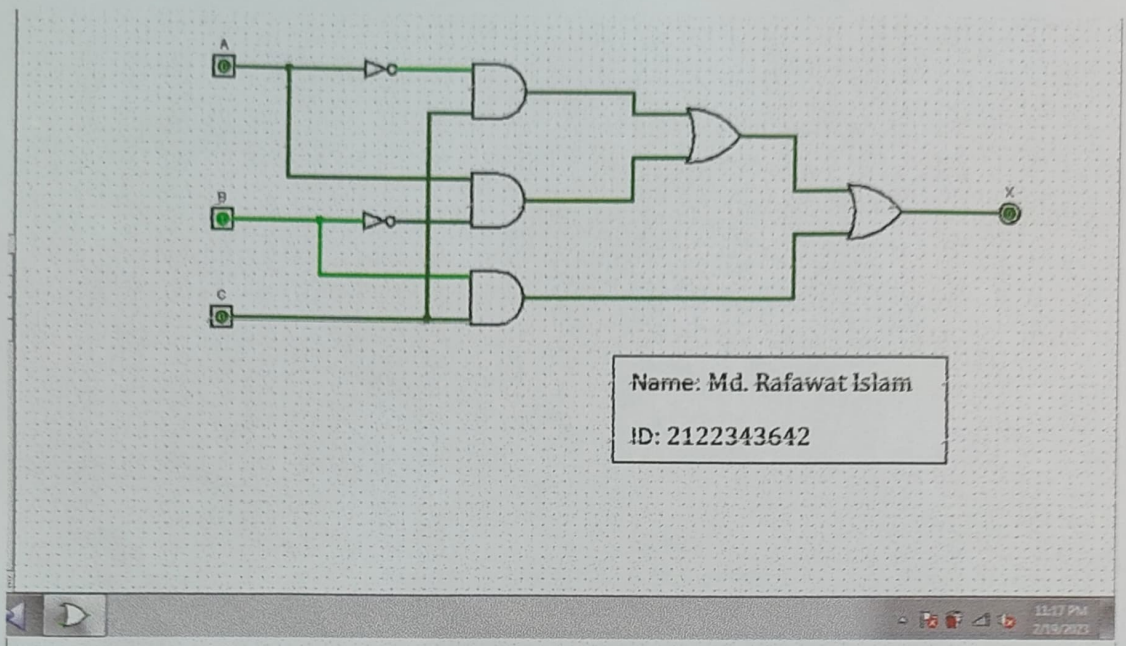

Name: Syed TAshriful Alam
Id: 2212623042
A
B
C
D
E
F
X


A
B
C
F
I3
Name: Syed Tashriful Alam
ID: 2212623042

Name: Md. Nifat Hossain, Id: 2212923042

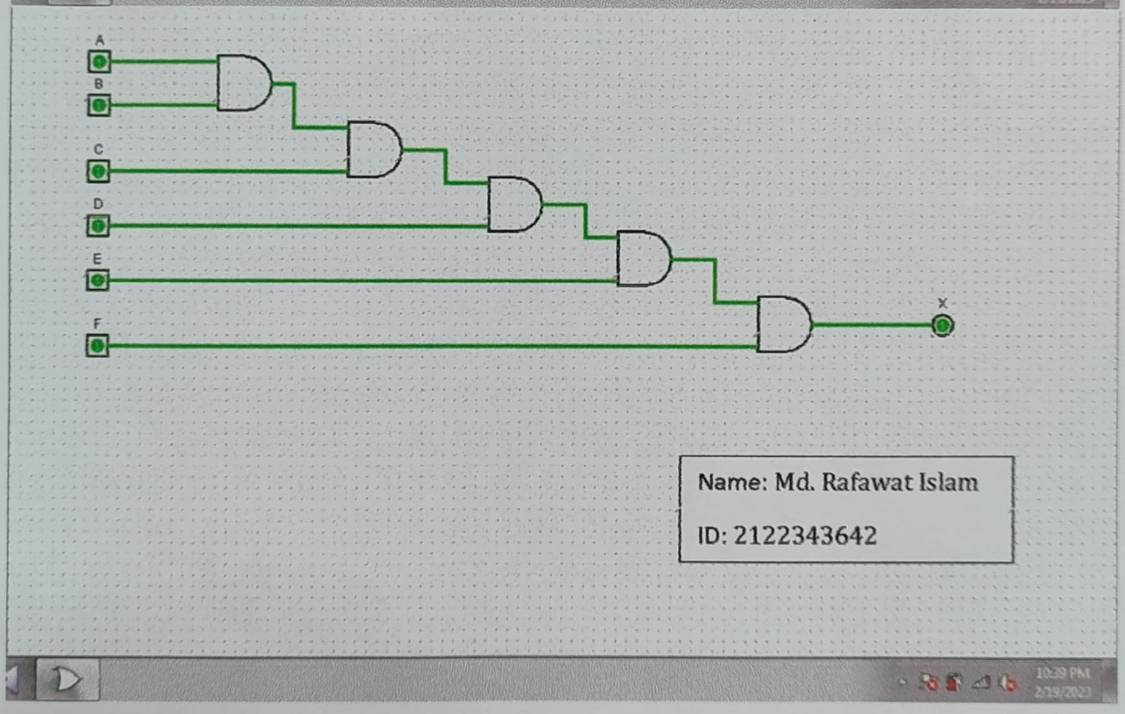






Name: Md. Rafawat Islam

ID: 2122343642



Name: Md. Rafawat Islam

ID: 2122343642

F. Data Sheet

F.1 Introduction to Basic Logic Gates

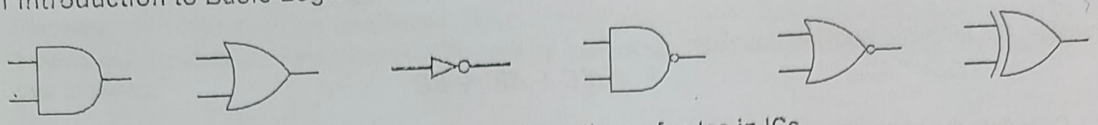


Figure F.1.1: Pin configurations of gates in ICs

Input $A\ B$	AND $F = A \cdot B$	OR $F = A + B$	NAND $F = \overline{A \cdot B}$	XOR $F = A \oplus B$	NOR $F = \overline{A + B}$
0 0	0	0	1	0	1
0 1	0	1	1	1	0
1 0	0	1	1	1	0
1 1	1	1	0	0	0

Input A	NOT $F = \overline{A}$
0	1
1	0

Table F.1.1: Truth Table of Logic Gates

F.2 Constructing 3-input AND & OR gates from 2-input AND & OR gates

$A\ B\ C$	$F = ABC$	$F = A + B + C$
0 0 0	0	0
0 0 1	0	1
0 1 0	0	1
0 1 1	0	1
1 0 0	0	1
1 0 1	0	1
1 1 0	0	1
1 1 1	1	1

Table F.2.1: Truth Tables for 3-input AND and OR

$F = ABC = (A.B).C$
$F = A + B + C$ =(A+B)+C

Table F.2.2: Expressing 3-input gates as 2-input gates using associative law.

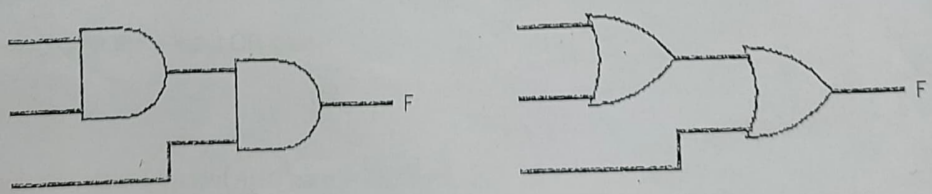


Figure F.2.1: Extension of inputs of AND and OR gates

F.3 Implementation of Boolean Functions

$A\,B\,C$	$I_1 = A'C$	$I_2 = AB'$	$I_3 = BC$	$F = I_1 + I_2 + I_3$
0 0 0	0	0	0	0
0 0 1	1	0	0	1
0 1 0	0	0	0	0
0 1 1	1	0	1	1
1 0 0	0	1	0	1
1 0 1	0	1	0	1
1 1 0	0	0	0	0
1 1 1	0	0	1	1

Figure F.3.1: Truth Table for the given Boolean Function

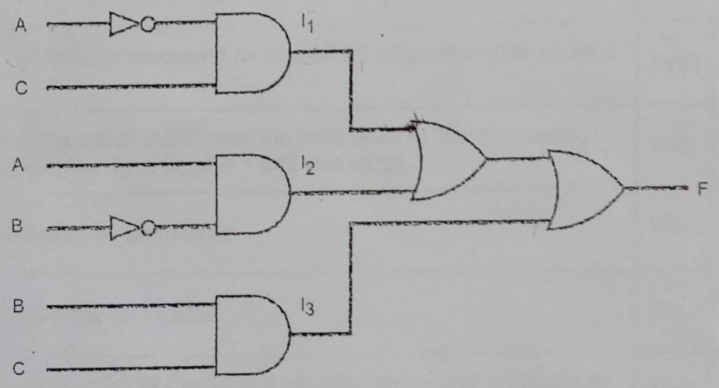


Figure F.3.1: Logic Diagram for the given Boolean Function

Proitthika Dhar
13/02/23
Bonus → 1