Lab 1: Digital Logic Gates and Boolean Functions

A. Objectives

- Study the basic logic gates - AND, OR, NOT, NAND, NOR, XOR.
- Get acquainted with the representation of Boolean functions using truth tables, logic diagrams and Boolean Algebra.
- Prove the extension of inputs of AND and OR gates using the associate law.
- Become familiarized with combinational logic circuits.

B. Theory

Logic Gates

Logic gates are the elementary building blocks of digital circuits. They perform logical operations of one or more logical inputs to produce a single output. Digital logic gates operate at two discrete voltage levels representing the binary values 0 (logical LOW) and 1 (logical HIGH). **Table B.1** provides a brief description of the basic digital logic gates, their corresponding IC numbers and circuit symbols.

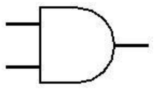
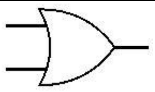
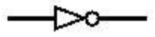
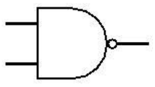
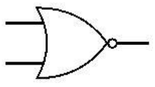
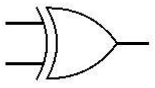
Gate	Description	IC #	Symbol
AND	Multi-input circuit producing an output of 1 if all inputs are 1.	7408	
OR	Multi-input circuit producing an output of 1 when any of its inputs is 1.	7432	
NOT	Single-input circuit that inverts the input (also called an Inverter). The output is 0 if the input is 1 and vice versa.	7404	
NAND	AND followed by an Inverter	7400	
NOR	OR followed by an Inverter	7402	
XOR	The Exclusive-OR or Ex-OR is a two-input circuit that produces an output of 0 is both inputs are same and 1 if the inputs are different.	7486	

Table B.1: Logic gates

Truth Tables

A B	$F = A \cdot B$
0 0	0
0 1	0
1 0	0
1 1	1

Table B.2: Truth table for an AND gate

A truth table shows all output logic levels of a logic circuit for every possible combination of inputs. For example, **Table B.2** shows the truth table for a two-input AND gate.

Boolean Algebra

Boolean algebra is a branch of mathematical logic that formalizes the relation between variables that take the truth values of *true* and *false*, denoted by 1 and 0 respectively. It is fundamental in the development of digital electronics. Digital electronics networks are generally expressed as Boolean functions. Discrete voltage levels are used to represent the truth values. Postulates and theorems of Boolean algebra are given in **Table B.3**.

Postulates and 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

Table B.3: Laws of Boolean algebra

Combinational Logic

Combination logic refers to digital networks where the output is solely dependent on the current input(s) and is not affected by previous states. The analysis of combination logic requires writing the Boolean functions for each element of the circuit, producing their truth tables, and subsequently combining each function for the final output and truth table.

C. 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 gates
- IC 7432 Quadruple 2-input OR gates
- IC 7486 Quadruple 2-input XOR gates
- Trainer Board
- Wires

Experiment 1: Introduction to Basic Logic Gates

D.1 Procedure

1. Place the 7408 AND IC on the breadboard. Make sure that every pin of the IC is on a separate node on the breadboard. Carefully note the location of the polarity mark of the IC. It will allow you to identify the different pins of the IC.

2. Connect the V_{CC} and GND pins of the IC to the +5 V and GND ports of the trainer board respectively.

3. Label the pin numbers of the inputs and output of the gate in **Figure F.1.1**, using the pin configurations in Error! Reference source not found..

4. Connect the logic gate:
   a. Connect each input of the gate to a toggle switch on the trainer board.
   b. Connect the output of the gate to an LED on the trainer board.

5. Apply all combinations of inputs by turning the toggle switches on (1) and off (0), and record if the LED is on (1) or off (0) as the output of the gate. Record your results in **Table F.1.1**.

6. Replace the AND IC with OR, NAND and XOR ICs without changing the connections and repeat step 5 for each.

7. Repeat steps 1-5 for the NOT and NOR ICs.

Experiment 2: Constructing 3-input AND & OR gates from 2-input AND & OR gates

D.2 Procedure

1. Complete the truth table for the 3-input AND gate in **Table F.2.1**.

2. Using the associative law given in **Table B3**, express the 3-input function using two 2-input AND gates in Table F.2.2.

3. Label the pin numbers in **Figure F.2.1**, using the pin configurations in.

4. Connect the circuit.

5. Connect the output to an LED and verify it using the truth table.

6. Repeat steps 1-5 for the 3-input OR gate.

Experiment 3: Implementation of Boolean Functions

D.3 Procedure

Consider the following Boolean Equation:

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

1. Complete the truth table for the implicants, $I_1 = A'C, I_2 = AB'$ and $I_3 = BC$ in **Table F.3.1**.

2. Using the values of the implicants, complete the truth table for the function F in **Table F.3.1**.

3. Label the pin numbers for the NOT, AND and OR gates of the function F in **Figure F.3.1**, using the pin configurations in **Figure B.2**.

4. Connect the input A to a NOT gate using the pins assigned in step 3 and check the output via an LED.

5. Wire up implicant I_1.
6. Connect the output of I_1 to an LED and verify it using the truth table.

7. Connect the input B to a NOT gate using the pins assigned in step 3 and check the output via an LED.
8. Wire up implicant I_2.
9. Connect the output of I_2 to an LED and verify it using the truth table.

10. Wire up implicant I_3.
11. Connect the output of I_3 to an LED and verify it using the truth table.

12. Connect the outputs of the three implicants as inputs to the OR gates (using the associative law).

13. Connect the output F to an LED and verify the function using the truth table.

Questions:

1) What are the names of the ICs that you would need if you wanted to use 13 AND gates, 12 NOT gates and 15 NOR gates in a circuit? How many of each IC would you need?
2) How can you power your logic ICs if the +5V port of your trainer board stops working?
3) Explain the Associative Law of Boolean algebra.
4) What is a Truth Table? Draw the Truth Table for an XNOR gate.
5) Let us assume you have two logical inputs A and B. If you pass A and B through a NAND gate and then pass the output of the NAND gate through a NOT gate, what logical operation will your final output represent? What is the name of the Boolean Algebra theorem that can be used to find this answer?
6) Draw the IC diagram for the circuit in Figure F.3.1. In place of the logic gates, draw the ICs and all the connections required to make the circuit work.
7) How can you use a 3 input AND gate as a 2-input AND gate? Can you use the same method to use a 3-input OR gate as a 2-input OR gate?

F. Data Sheet

F.1 Introduction to Basic Logic Gates

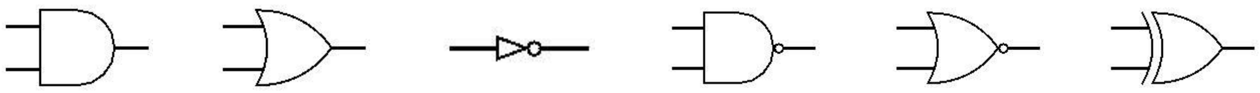


Figure F.1.1: Pin configurations of gates in ICs

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 1					
1 0					
1 1					

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

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

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

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

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

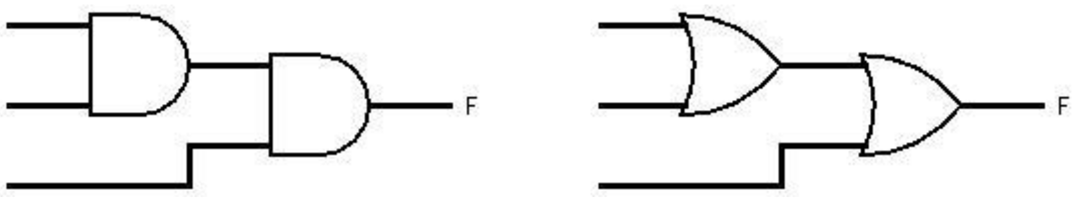


Figure F.2.1: Extension of inputs of AND and OR gates

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

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

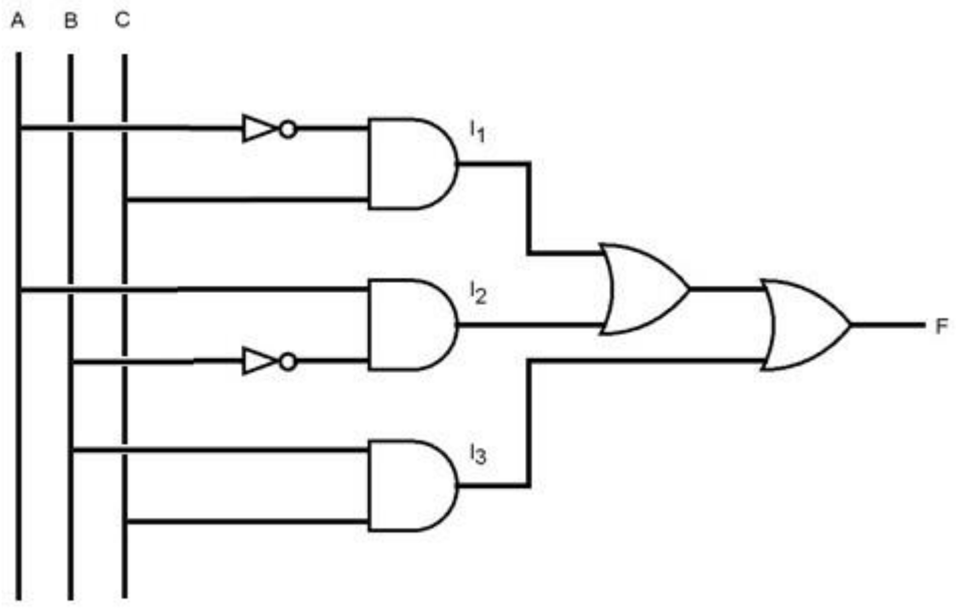


Figure F.3.1: Logic Diagram for the given Boolean Function