**Experiment 3**

**Verification of De Morgan’s Theorem**

**Objective**

In this lab students will

* Verify the De-morgan’s theorem using IC 74LS08, 74LS32, 74LS04.
* Learn the application of De-Morgan’s Theorem for simplification of circuit and Boolean Expressions.

**Components**

* IC74LS32×1
* IC74LS04×1
* IC74LS08×1
* AM2000 TRAINER
* Multimeter
* Cutter
* Single core wire
* Pair of Pliers

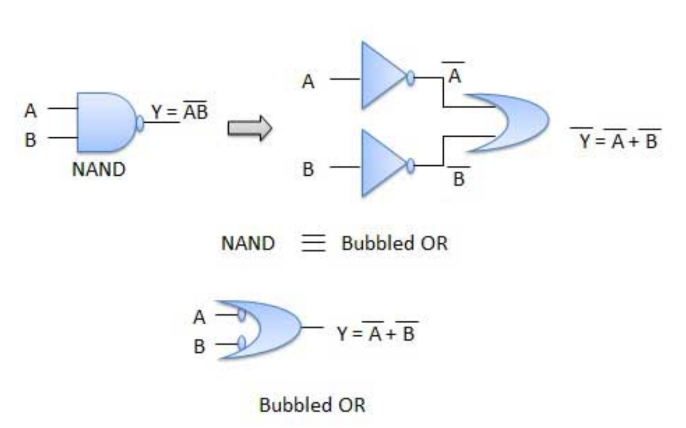
***De Morgan’s Theorem:***

De Morgan has suggested two theorems which are extremely useful in Boolean Algebra. The two theorems are discussed below.

**Theorem 1**

**NAND=Bubbled OR**

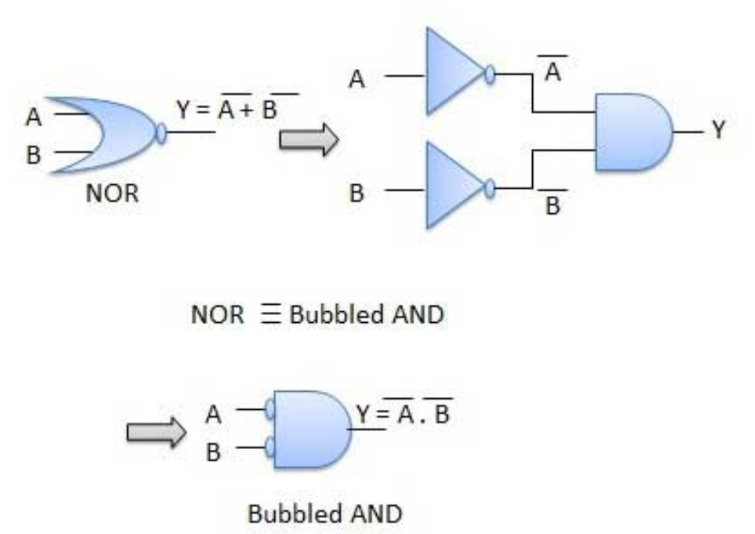
* The left hand side (LHS) of this theorem represents a NAND gate with inputs A and B, whereas the right hand side (RHS) of the theorem represents an OR gate with inverted inputs.
* This OR gate is called as **Bubbled OR**.

****

**Theorem 2**

**NOR=Bubbled AND**

* The LHS of this theorem represents a NOR gate with inputs A and B, whereas the RHS represents an AND gate with inverted inputs.
* This AND gate is called as **Bubbled AND**.



**Procedure:**

1. Connect the AM2000 trainer to the 220V AC power supply
2. Turn on the trainer and verify the voltage of the power supply usin the multimeter. It should be +5V exactly.
3. Install IC74LS08, IC74LS32, IC74LS04 on the trainer’s board.
4. Wire the circuit according to the diagram by consulting gate IC’s diagram from previous manuals.
5. Use any of the two logic switches from S2 to S9 for inputs A and Brespectively.
6. For output indication use any of the LED’s from L0 to L15.
7. Supply the +5v and GND to the pins 14 and 7 of the IC respectively.
8. Test all the possible combinations of inputs and verify the output according to the truth table of Demorgan’s Theorem.
9. Make truth table according to the results.

**In case of trouble:**

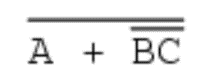
1. Check the power supply.
2. Check the Vcc and GND at pins 14 and 7 respectively.
3. Check all the wire connections.
4. Check the circuit wiring and remove the breaks.
5. Check the IC using truth table.

**Truth Table: [4]**

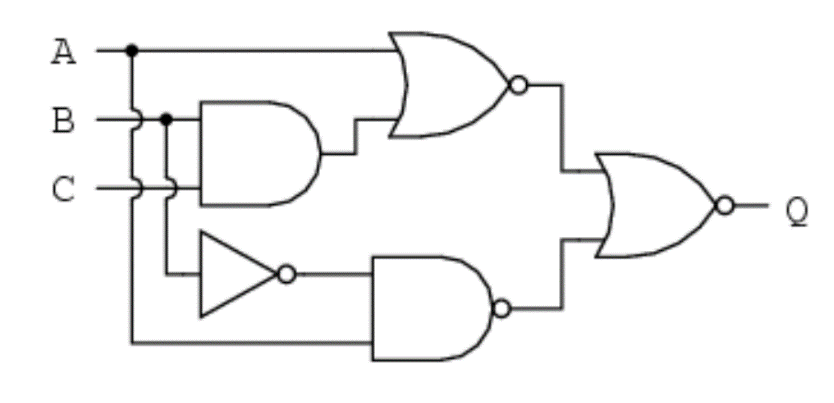
|  |  |  |  |
| --- | --- | --- | --- |
| **X** | **Y** | **(X+Y)** | **(X+Y)’** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **X’** | **Y’** | **X’.Y’** | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
| **X** | **Y** | **X.Y** | **(XY)’** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **X’** | **Y’** | **X’+Y’** | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |

**Exercise:**

**1-Simplify the following expression using De-Morgan’s Theorem [4]**

****

**2-Simplify the following circuit using De-Morgan’s theorem [4]**

****

**Solution:**

**Conclusion: [3]**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Lab Exercise (5)**

**To check the operation of 3 input XOR gate using 2 input XOR gate using IC 74LS86 IC.**

***3 Input XOR gate:***

A 3-input XOR gate can be implemented using 2-input XOR gates by cascading 2 2-input XOR gates. Two of the three inputs will feed one of the 2-input XOR gates. The output of the first gate will, then, be XORed with the third input to get the final output.

Let us say, we want to XOR three inputs A, B and C to get the output Z. First, XOR A and B together to obtain intermediate output Y. Then XOR Y and C to obtain Z. The schematic representation to obtain 3-input XOR gate by cascading 2-input XOR gates is shown in figure below:



**Figure 1 Implementation of 3-input XOR gate using 2-input XOR gate**

**Truth Table for 3-input XOR gate: [2 Marks]**

|  |  |  |  |
| --- | --- | --- | --- |
| **A(input)** | **B(input)** | **B(input)** | **Output** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Derive the expression for 3 input XOR gate and draw its gate level design. [3 Marks]**