B-3.

Boolean Algebra

Objectives:

This experiment will verify experimentally some of the Boolean Laws, DeMorgan's Theorem, and the XOR function.

READING:

MATERIALS NEEDED:

74 LS 04 1 EA

74 LS 32 1 EA

74 LS 08 1 EA

LED 1 EA

330Ω 1 EA

POWER SUPPLY

SUMMARY OF THEORY:

Generally you will find that the basic logic functions AND, OR, NAND, NOR, and NOT are not sufficient to implement complex digital logic functions. These gates are the basis for building more complex logic circuits that are constructed using various combinations of gates which is known as Combinational Logic. Combinational logic requires the use of two or more gates to form a useful, complex function.

These complex functions usually begin as a Boolean Equation and the logic circuit may be implemented directly from this equation. However, there are times when the same function may be generated from a less complex equation. It is to the logic designer’s advantage to implement a logic function using as few logic gates as necessary. This reduces cost and fewer parts may mean a more reliable circuit and one that is easier to build and repair.

The text lists the three Laws and twelve Rules of Boolean Algebra. The student is advised to refer to the text as necessary to review this material. These rules of Boolean Algebra allow certain variables to be combined or eliminated to form a simpler equivalent circuit. An equivalent digital circuit may be defined as one that yields the same truth table as the more complex circuit.

DeMorgan developed a theorem that allows conversion between a logic expression that has inversions on the output to a different logic expression with the inversions on each of the inputs. This may allow for the simplification of a Boolean Expression by the cancellation of some redundant inversions. There are two Boolean Equations that represent DeMorgan's Theorem:

(1A)

(1B)

When simplifying Boolean Equations, it is sometimes convenient use DeMorgan's Theorem to proceed in the opposite direction. After using DeMorgan's Theorem, further simplification

An example of a simple combinational logic function is the Exclusive−OR gate. The XOR is a gate with two or more inputs that will give a logic 1 output when there are an odd number of inputs at logic 1, and a logic 0 output when there are and even of inputs at logic 1.

The logic symbol and the truth table for the 2-input XOR may be found in Figure 1 and the equation is:

Q = A ⊕ B (2)

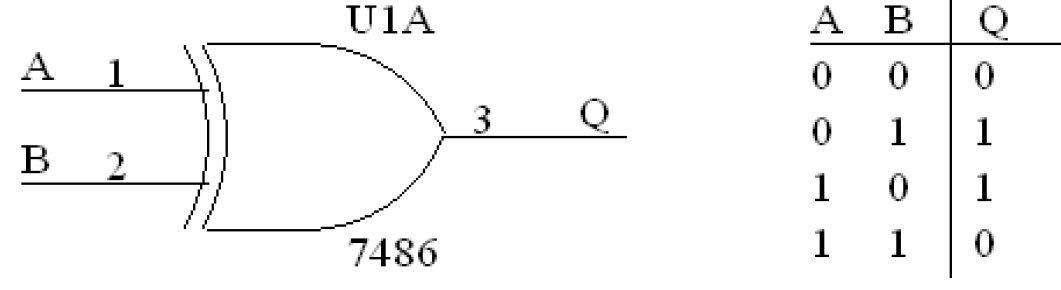


Figure 1 XOR

The XOR function is not a basic gate but is a combinational logic function. A minimum of 3 logic gates are required to construct the gate. The Boolean Equation for the XOR function is:

Q = (3)

This function may be generated using either the circuit of Figure 2 or Figure 3. To prove that these two circuits are equivalent, all that is required is to construct the truth table for both and compare the outputs. If the all outputs give the same logic levels for the appropriate input, the two circuits are equivalent.

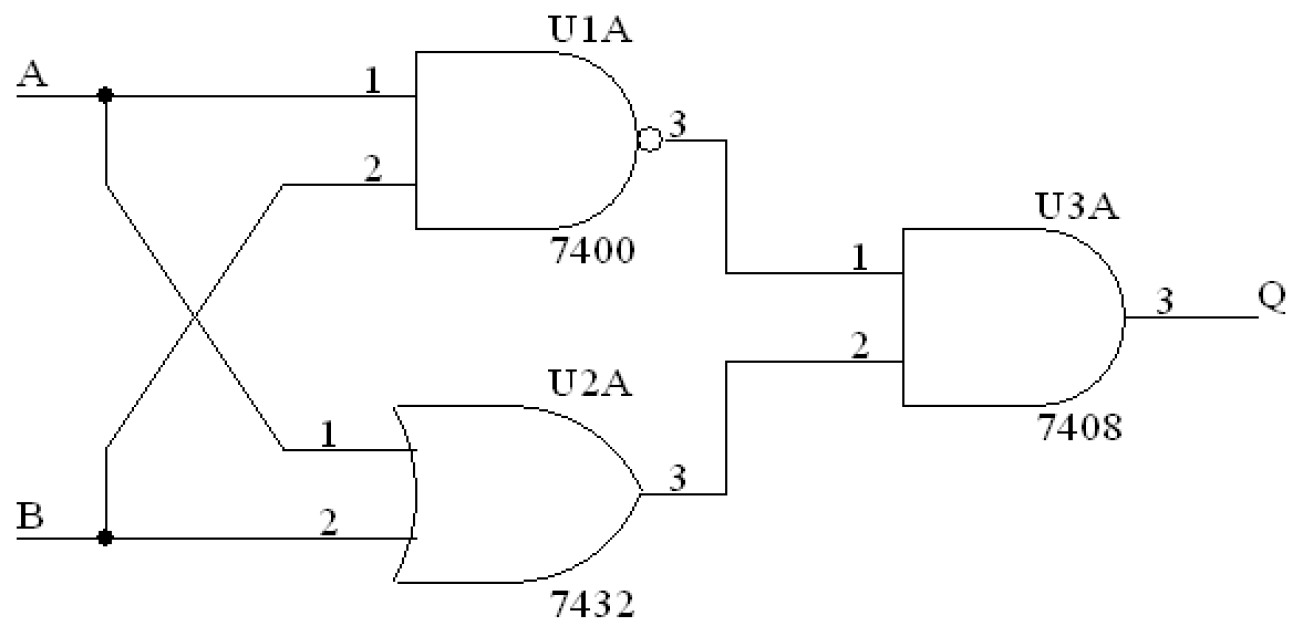


Figure 2

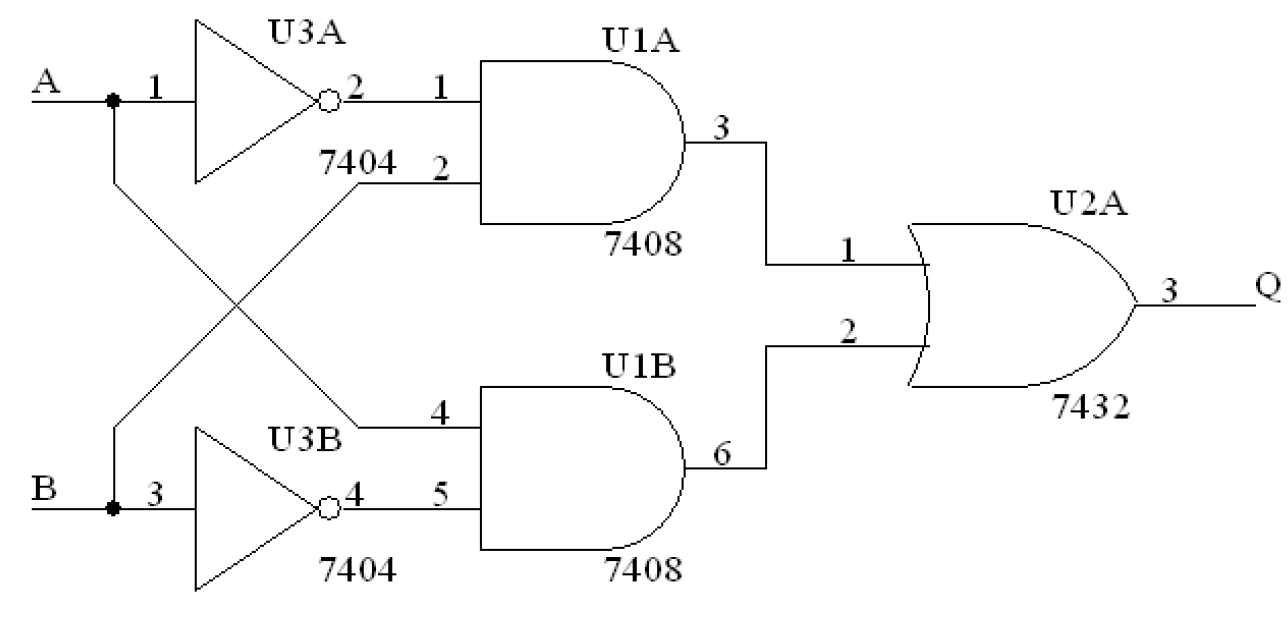


Figure 3

The Boolean Equation for Figure 2 will give the Boolean Equation for the XOR gate. We can derive the Boolean Equation for Figure 3 from the equation for Figure 2 by applying the Rules of Boolean Algebra as follows:

(4)

From DeMorgan's Theorem:

From the Distributive Law

Applying Rule 8, we obtain

The complement of the Exclusive−OR is the Exclusive−NOR. The XNOR will provide a logic 0 on the output when an odd number of inputs are at logic 1, and a logic 1 on the output when there are all 0's or an even number of 1's on the input. The logic symbol and truth table may be found in Figure 4. From the truth table for the XOR, we may write the Boolean Equation as:

Q = A B + AB (5)

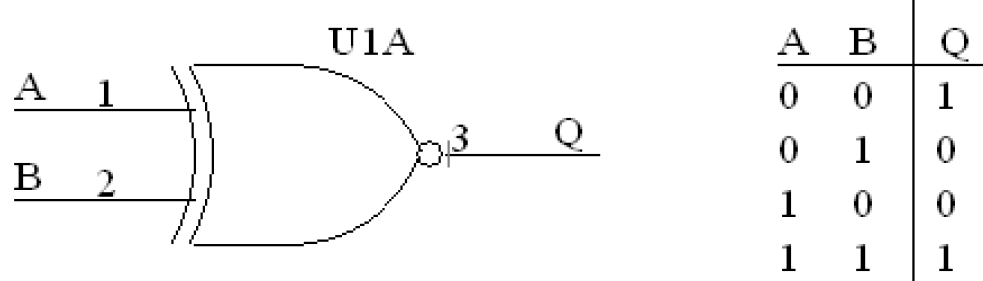


Figure 4 XOR

The logic diagram for the XNOR based on the Boolean Equation is shown in Figure 5.

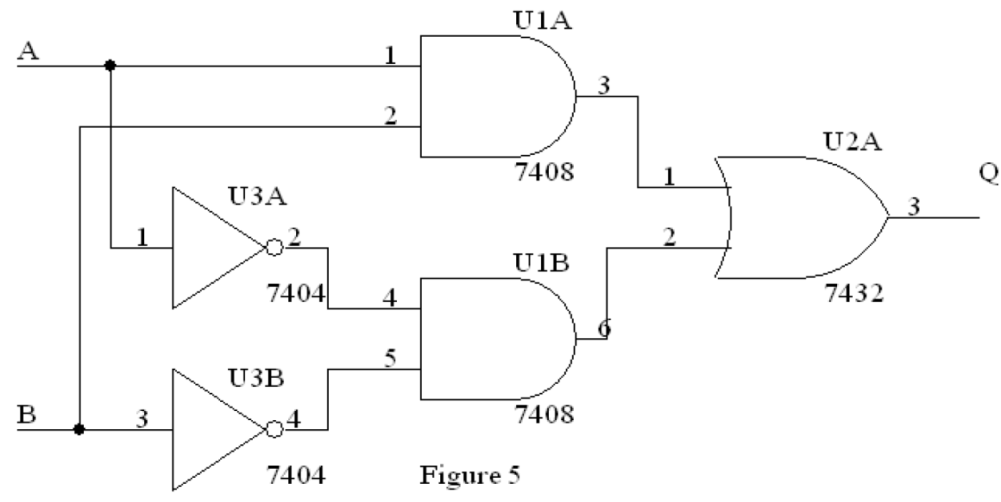


Figure 5 XNOR

Theorem of Boolean Algebra

* Theorem 1 “ Double Complementation or Double Negation Theorem”

a)

* Theorem 2 “Idempotency Theorem”

a)

b)

* Theorem 3 “Identity Element Theorem”

a)

b)

* Theorem 4 “Absorption Theorem”

a)

b)

* Theorem 5 “Associative Theorem”

a)

b) 

* Theorem 6 “Adjacency Theorem”

a) 

b)

* Theorem 7 “Consensus Theorem”

a)

b)

* Theorem 8 “Simplification Theorem”

a)

b)

* Theorem 9 “DeMorgan’s Theorem (2-Variable form)”

a)

b)

* Theorem 10 “DeMorgan’s Theorem (General form)”

a)

b)

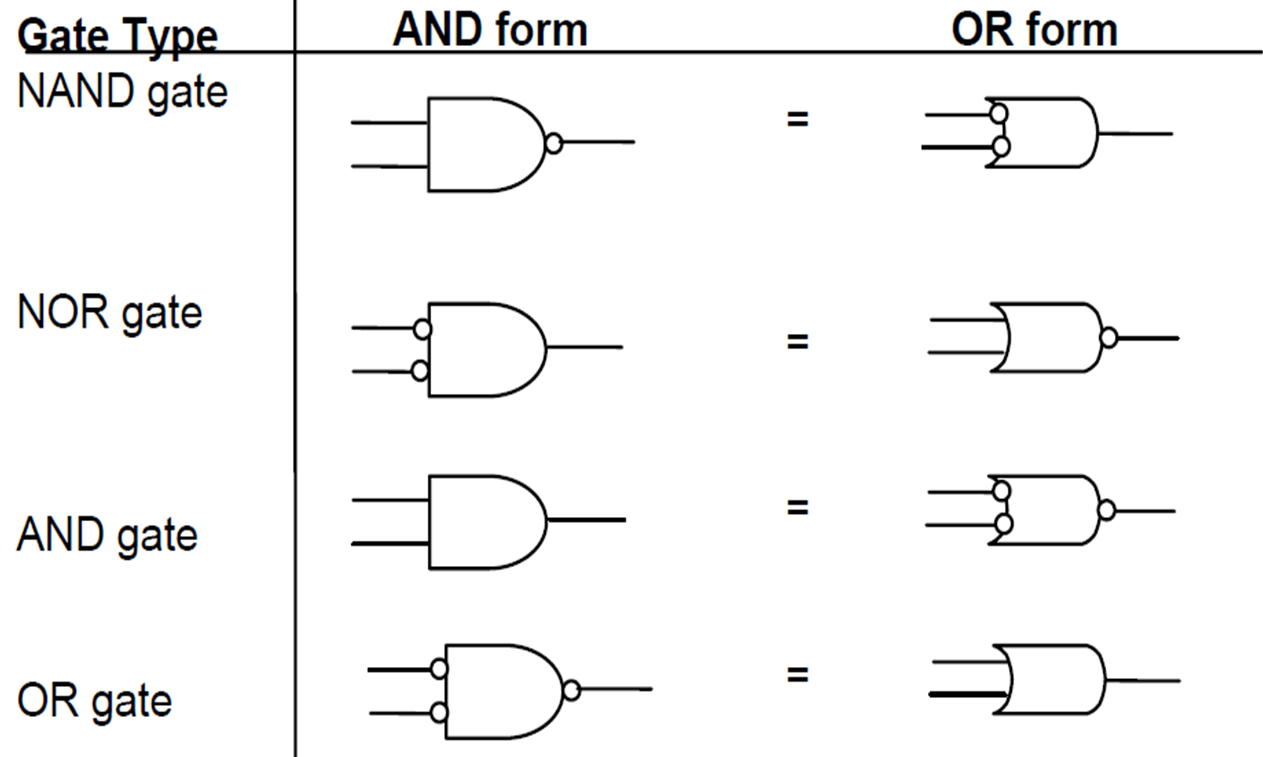
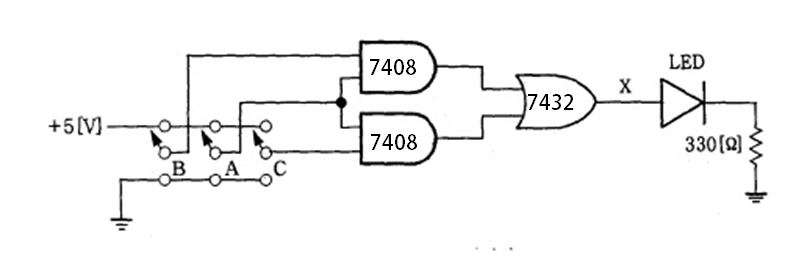


Figure 6 AND. OR, NOR

PROCEDURE

1. Build the below circuit. Fill in below Table with the logic levels using LED lamp.



C:\Users\Administrator\Desktop\3-1.emf

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | | | **Output** | |
| **A** | **B** | **C** | **X** | **Y** |
| **0** | **0** | **0** |  |  |
| **0** | **0** | **1** |  |  |
| **0** | **1** | **0** |  |  |
| **0** | **1** | **1** |  |  |
| **1** | **0** | **0** |  |  |
| **1** | **0** | **1** |  |  |
| **1** | **1** | **0** |  |  |
| **1** | **1** | **1** |  |  |

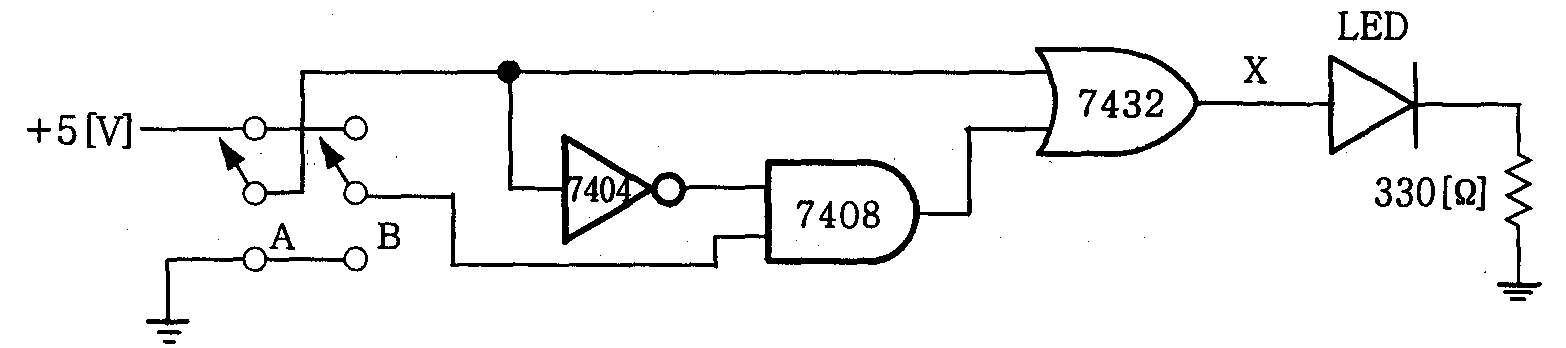
1. Build the below circuit. Fill in below Table with the logic levels using LED lamp.

C:\Users\Administrator\Desktop\디지털논리실험 그림수정\3-2.emf

C:\Users\Administrator\Desktop\디지털논리실험 그림수정\3-3.emf

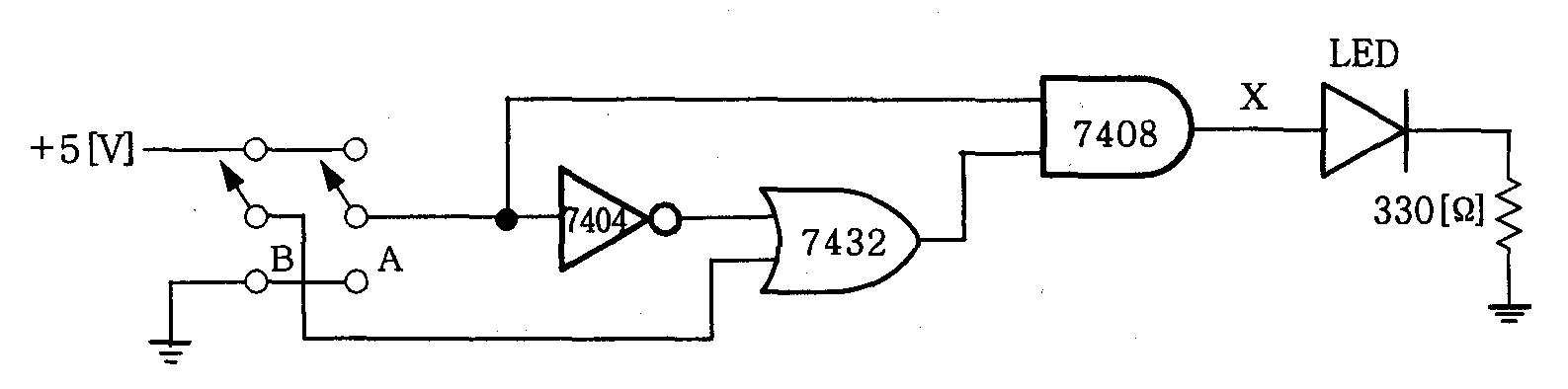
|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | **Output** | |
| **A** | **B** | **X** | **Y** |
| **0** | **0** |  |  |
| **0** | **1** |  |  |
| **1** | **0** |  |  |
| **1** | **1** |  |  |

1. Build the below circuit. Fill in below Table with the logic levels using LED lamp.



|  |  |  |
| --- | --- | --- |
| **Input** | | **Output** |
| **A** | **B** | **X** |
| **0** | **0** |  |
| **0** | **1** |  |
| **1** | **0** |  |
| **1** | **1** |  |

1. Build the below circuit. Fill in below Table with the logic levels using LED lamp.



|  |  |  |
| --- | --- | --- |
| **Input** | | **Output** |
| **A** | **B** | **X** |
| **0** | **0** |  |
| **0** | **1** |  |
| **1** | **0** |  |
| **1** | **1** |  |

1. Draw the logic diagram for the Boolean equation .
2. Build the circuit of ⑤ and construct a truth table based on the logic outputs for every possible input.

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | | **Output** |
| **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** |  |

1. Draw the logic diagrams of below equation
2. **Additional work (if needed):** Build the circuits, and demonstrates both Boolean Equations are equivalent.

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | | **Output ()** |
| **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** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | | **Output ()** |
| **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** |  |