**LAB # 9**

**Introduction to Basic Instruments and Study of Logic Gates**

**OBJECTIVE:**

1. Familiarization with the Lab Instruments.
2. To Study and Verify the Truth Tables of the Logic Gates.

**EQUIPMENT:**

* 1. Bread Board.
  2. Digital Design Module.
  3. Digital Logic Probe.
  4. Wire Stripper.

5. ICs: 74LS00, 74LS02, 74LS04, 74LS08, 74LS32, 74LS86 and 74LS266.

6. DC supply (0 and +5V).

**Part – I**

***(*Familiarization with Lab Equipments*)***

**THEORY:**

**Bread Board:**

Bread Board is used to design and test different circuits. It is an array that contains the holes where simply the components and the connecting wires are pushed to form a circuit. One of the major advantage of using the Bread Board is that the components are not soldered. So, they can be plugged in / out easily if they are connected incorrectly. The Bread Board with its internal connections / structure is shown in Figure. 1.1.

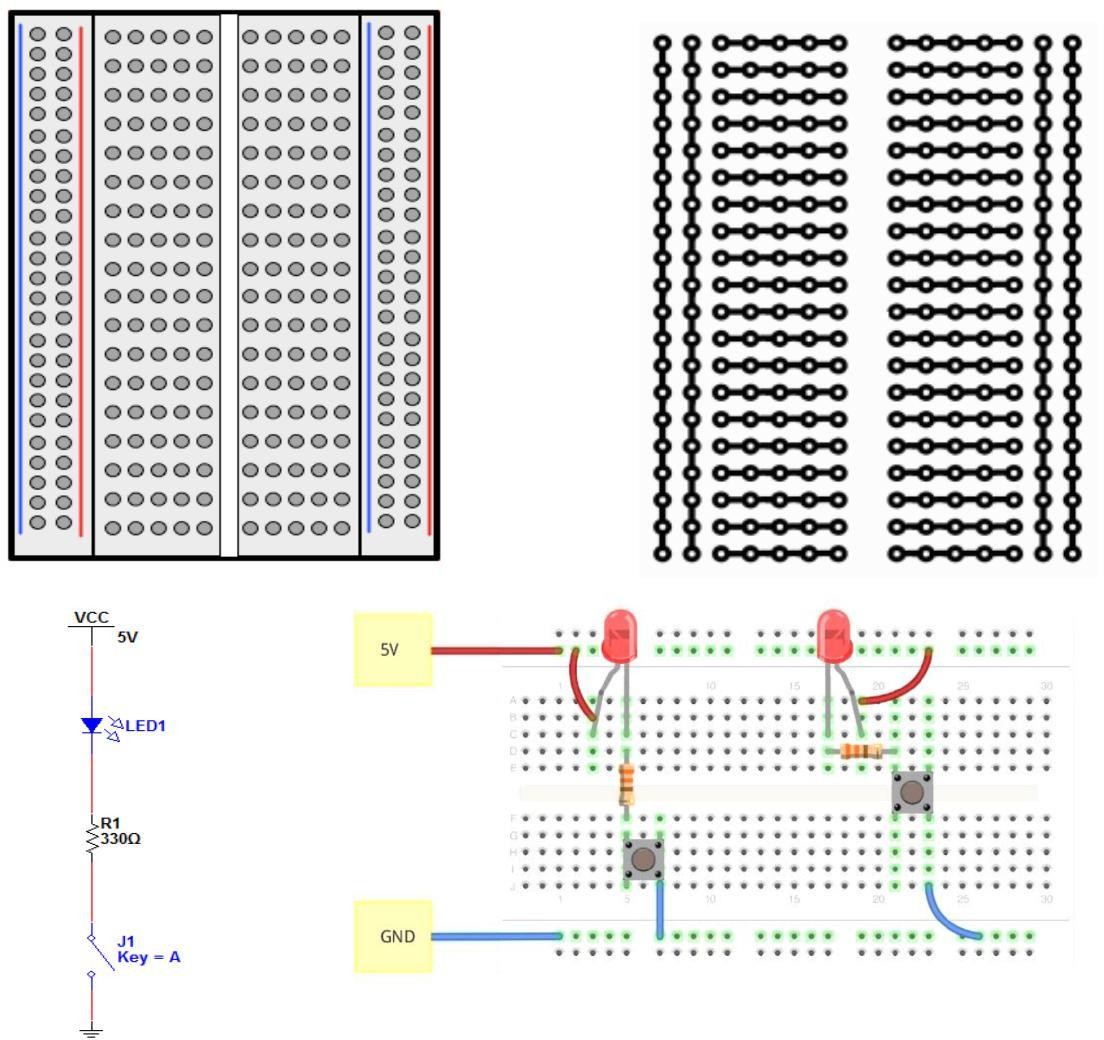


Figure. 1.1: Bread Board and its Internal Connections / Structure.

**Digital Design Module (NI myDAQ):**

###### NI myDAQ is a low-cost portable Data Acquisition device that uses the NI LabVIEW based software instruments, allowing the user to measure and analyze the real world signals. NI myDAQ is ideal for the Electronics and taking the sensor measurements. Combined with NI LabVIEW on the PC, one can analyze and process the required signals and control simple processes anytime, anywhere.

myDAQ provides Analog Input (AI), Analog Output (AO), Digital Input and Output (DIO), Audio Input and Output (AIO), DC Power Supplies, and Digital Multimeter (DMM) functions in a compact USB device. The myDAQ connection diagram is shown in Figure. 1.3.



Figure. 1.2: NI myDAQ

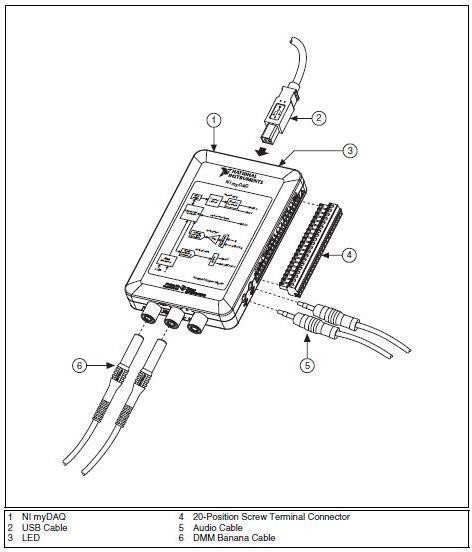


Figure. 1.3: myDAQ Connection Diagram

The myDAQ provides a soft front panel (from NI LabVIEW) to control the functionality of the device, which can launch several instruments including Digital Multimeter (DMM), Oscilloscope, Function Generator, Bode Analyzer, Dynamic Signal Analyzer, Arbitrary Waveform Generator and Digital Reader / Writer

**Application Software Details:**

The software that is used to interface the myDAQ card with PC is *NI ELVISmx for NI ELVIS*

###### *NI myDAQ.* Through this software, the different device components of NI myDAQ can be operated which includes Digital Multimeter, Bode Analyzer, Oscilloscope, Function Generator, Impedance Analyzer, etc. etc.

For opening this application software, follow the path as:

*Start All Programs National Instruments NI ELVISmx for NI ELVIS & NI myDAQ NI ELVISmx Instrument Launcher. As shown in figure 1.4*

All the myDAQ devices are shown in this interface that shows that it can be used for analog applications as well besides the digital. We can access any of the device by just clicking on its icon. For the Digital Lab, we will use just three applications devices. i.e. Digital Multimeter, Digital Reader and Digital Writer. These are shown as in figures Figure 1.5 – 1.7.

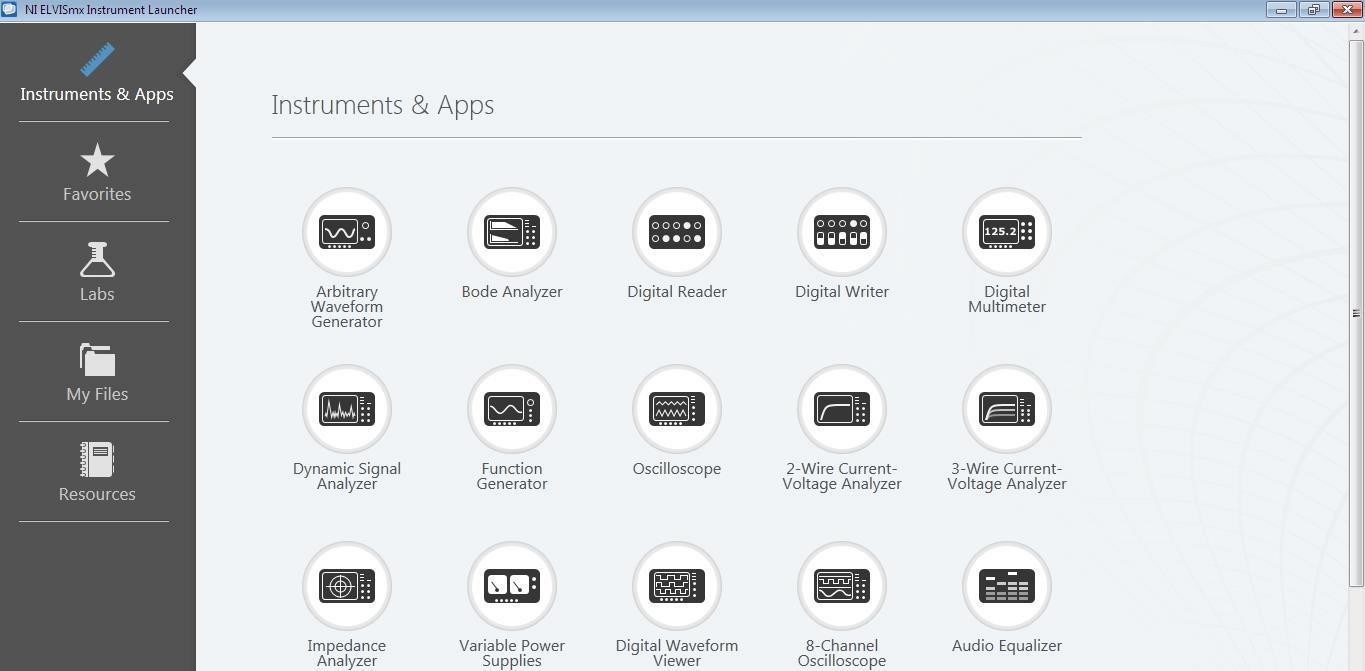


Figure. 1.4: NI Instrument Launcher

###### .

Figure. 1.5: Digital Multimeter Interface

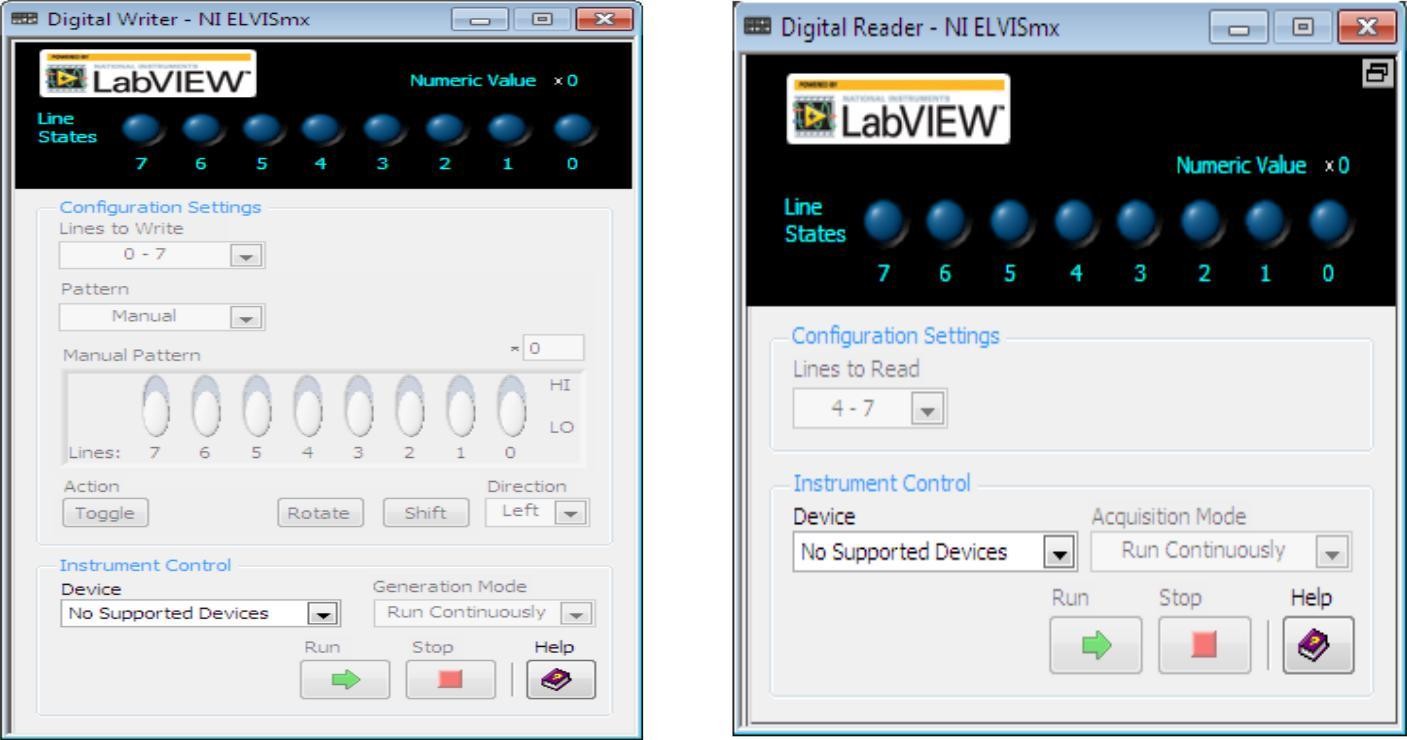


Figure. 1.6: Digital Reader & Digital Writer Interfaces

###### After making all the necessary connections for the Signal Reading / Writing purpose, the RUN button should be pressed in order to get the readings on interface display.

**Digital Logic Probe:**

A **logic probe** is a hand-held pen-like test probe used for analyzing and troubleshooting the logical states (Boolean 0 or 1) of a digital circuit. While most are powered by the circuit under test, some devices use batteries. They can be used on either TTL (transistor-transistor logic) or CMOS (complementary metallic oxide semiconductor) integrated circuit devices. There are usually three differently-colored LEDs on the probe's body:

Red and green LEDs indicate high and low states respectively. An amber LED indicates a pulse.

The pulse-detecting electronics usually has a pulse-stretcher circuit, so that even very short pulses become visible on the amber LED. A control on the logic probe allows either to capture and storage of a single event or continuous running. Probe is shown in Figure. 1.7.

When the logic probe is either connected to an invalid logic level (a fault condition or a tri-stated output) or not connected at all, none of the LEDs lights up. Another control on the logic probe allows selection of either TTL or CMOS family logic. This is required as these families have different thresholds for VIH and VIL. Some logic probes have a separate audible tone for each of the logical states. An oscillating signal causes the probe to alternate between high-state and low- state tones. A logic probe is a cheap, versatile and convenient digital test instrument, but can test only a single signal at a time. When many logic levels need to be observed or recorded simultaneously, a logic analyzer is used.



Figure. 1.7: Digital Logic Probe.

##### Wire Stripper:

###### It is a tool that is used to remove the insulation from the wire or cut the wire.

Figure. 1.8: Wire Stripper

##### Safety:

###### Following proper safety practices are a must when working with electronic equipment. Not only is there the danger of electrical shock, but the components can explode if not connected properly. Many of today’s electronic components are easily damaged by improper handling. The test equipment used in the electronic service industry is expensive and easily damaged if proper operating procedures are not followed. Make sure TEST INSTRUMENTS are set for proper FUNCTION AND RANGE prior to taking a measurement.

1. Always cut wire leads so the clipped wire falls on the table top and not toward others.
2. Avoid skin contact.
3. Only work with powered units when necessary for troubleshooting.
4. Avoid pinching wires when putting equipment back together.
5. Never solder a circuit that has the power applied.
6. Double check circuits for proper connections and polarity prior to applying the power.
7. Observe polarity when connecting polarized components or test equipment into a circuit.

**Part – II**

***(*Study of Logic Gates*)***

**THEORY:**

**AND Gate:**

The AND function is similar to the multiplication in mathematics. This is the all or nothing operator and it provides a logic 1 output only when all the inputs of the gate are at logic 1, and logic 0 output for all other input combinations. The logic operator for the AND function is a dot (**∙**) sign. The AND function is described in terms of the following “truth table”.

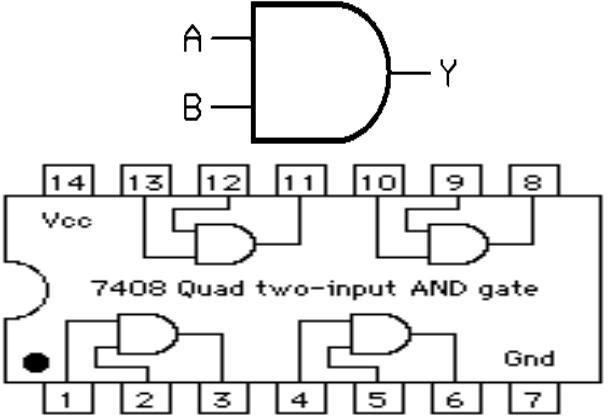
**(Boolean Expression** is *“ Output = A●B ”***).**

TABLE 1.1: Function of AND Gate

|  |  |  |
| --- | --- | --- |
| **Input: A** | **Input: B** | **Output** |
| **Y = A۰B** |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Figure. 1.9: AND Gate Symbol & TTL IC

#### OR Gate:

###### The OR function is similar to the mathematical function of addition and the output for the OR gate may be analyzed using the laws of addition. The logic operator for the OR function is a plus (+) sign. The output will be logic 0 only if all the inputs are logic 0, and the output will be logic 1 anytime any input is at logic 1.

**(Boolean Expression** is as “Output *= A+B”***).**

TABLE 1.2: Function of OR Gate

|  |  |  |
| --- | --- | --- |
| **Input: A** | **Input: B** | **Output** |
| **Y= A+B** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |



Figure. 1.10: OR Gate Symbol & TTL IC

###### 7

**NOT Gate:**

The NOT circuit or inverter performs the basic logic function of complementation. It may be identified by the presence of a bubble on the input or the output of the traditional logic symbol. The output of NOT gate is the inverse of the input. Unlike the others it only has one input and one output.

**(Boolean Expression** is as ***“****Output =* A**’** *”***).**

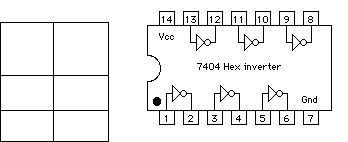
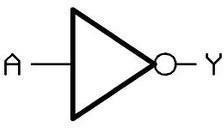


TABLE 1.3: Function of NOT Gate

**Input: Output:**

**A Y = A’**

###### 0 1

1 0

**NAND Gate:**

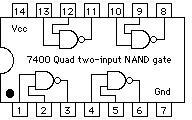
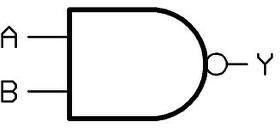
Figure. 1.11: NOT Gate Symbol & TTL IC

The NAND function is the complement of the AND function and the logic symbols have the inversion on the output. NAND gate is constructed by adding an inverter after AND operator. The NAND function provides logic 0 on the output only when all inputs are logic 1, and logic

1 output for all other combinations.

̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅

**(Boolean Expression** is as ***“****Output =* (A۰B)**’** *”***).**

TABLE 1.4: Function of NAND Gate 

|  |  |  |
| --- | --- | --- |
| **Input: A** | **Input: B** | **Output:** |
| **Y = (A.B)’** |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Figure. 1.12: NAND Gate Symbol & TTL IC

##### NOR Gate:

###### The complement of the OR function is the NOR function and the logic symbol has the inversion present on the output. NOR gate is constructed by adding an inverter after OR operator. The NOR function provide logic 1 when all input are 0 and 0 for all other combinations.

**(Boolean Expression** is as *“Output =* ̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅*”***).**

+

TABLE 1.5: Function of NOR Gate

|  |  |  |
| --- | --- | --- |
| **Input:A** | **Input: B** | **Output: Y= (A+B)’** |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

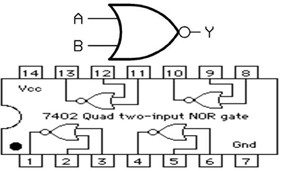


Figure. 1.13: NOR Gate Symbol & TTL IC

##### Exclusive OR Gate:

###### The Exclusive OR (also known as XOR) gate is a logic function that results true if one and only one input to the gate is 1. In other words, when the inputs are at different logic levels, the output is 1 and when the inputs are at the same logic levels, the output is 0.

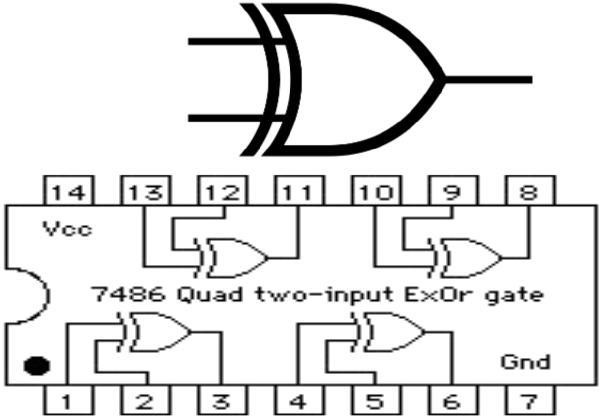
(**Boolean Expression** is as *“Output =* A⊕B *”*).

TABLE 1.6: Function of XOR Gate

|  |  |  |
| --- | --- | --- |
| **Input: A** | **Input: B** | **Output**: |
| **(A**⊕**B)** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Figure. 1.14: XOR Gate Symbol & TTL IC

##### Exclusive NOR Gate:

The Exclusive NOR (also known as XNOR) is a function that is the complement of the XOR Function that results in 1 when the inputs are same and results 0 when the inputs are different.

̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅

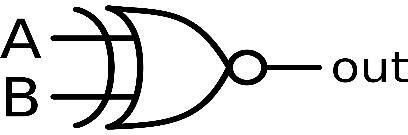
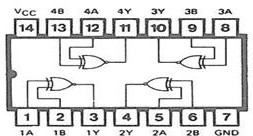
(**Boolean Expression** is as *“Output* = (A⊕B)**’** *”*).

TABLE 1.7: Function of XNOR Gate



|  |  |  |
| --- | --- | --- |
| **Input:A** | **Input: B** | **Output:**  **Y= (A**⊕**B)’** |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Figure. 1.15: XNOR Gate Symbol & TTL IC

**PROCEDURE:**

###### Place the respective ICs (74LS400, 74LS02, 74LS04, 74LS08, 74LS32, 74LS86 and 74LS266) on the trainer board.

1. Connect VCC and ground to the respective pins on the trainer board.
2. Connect the inputs to the input switches provided in the trainer board.
3. Connect the outputs to the switches of output LEDs and apply various combinations of inputs as shown in the truth table and observe the conditions of LEDs.

**OBSERVATIONS / RESULTS & DISCUSSION:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input: A** | **Input: B** | ̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅  ۰  **7400**  **NAND** | ̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅̅  +  **7402**  **NOR** | **A**●**B 7408**  **AND** | **A+B 7432**  **OR** | **(A⊕B) 7486**  **XOR** | **(A⊕B)’**  ⊕  **74266**  **XNOR** |
| 0 | 0 |  |  |  |  |  |  |
| 0 | 1 |  |  |  |  |  |  |
| 1 | 0 |  |  |  |  |  |  |
| 1 | 1 |  |  |  |  |  |  |
| 0 | Not Connected |  |  |  |  |  |  |
| 1 | Not Connected |  |  |  |  |  |  |
| Not Connected | 0 |  |  |  |  |  |  |
| Not Connected | 1 |  |  |  |  |  |  |
| Not Connected | Not Connected |  |  |  |  |  |  |

**CONCLUSION:**

1. The output of AND Gate is only high when all inputs are high.
2. The output of OR Gate is low when all inputs are low.
3. The output of NOT Gate is inverse of input.
4. The output of NAND Gate is low when all inputs are high.
5. The output of NOR Gate is high when all inputs are low.
6. The output of XOR Gate is high when the inputs are at different logic levels.
7. The output of XNOR Gate is high when the inputs are at same logic levels.

**QUESTIONS:**

1. How do these gate treat when inputs are unconnected (Low or High)? Briefly explain why?
2. Do swapping A and B make any difference?
3. What is difference between an Inverter and a **NOT** gate?
4. How can you make an inverter using a **NOR** gat? And using **NAND** gate?
5. Implement XOR gate function using basic gates?
6. Implement XNOR gate function using basic gates?