**DELHI TECHNOLOGICAL UNIVERSITY**

SHAHBAD DAULATPUR VILLAGE, ROHINI, DELHI, 110042



**(DIGITAL ELECTRONICS)**

**SUBMITTED BY: SUBMITTED**

**RITIK SINGH TO:**

**COE Mr. SACHIN TARAN**

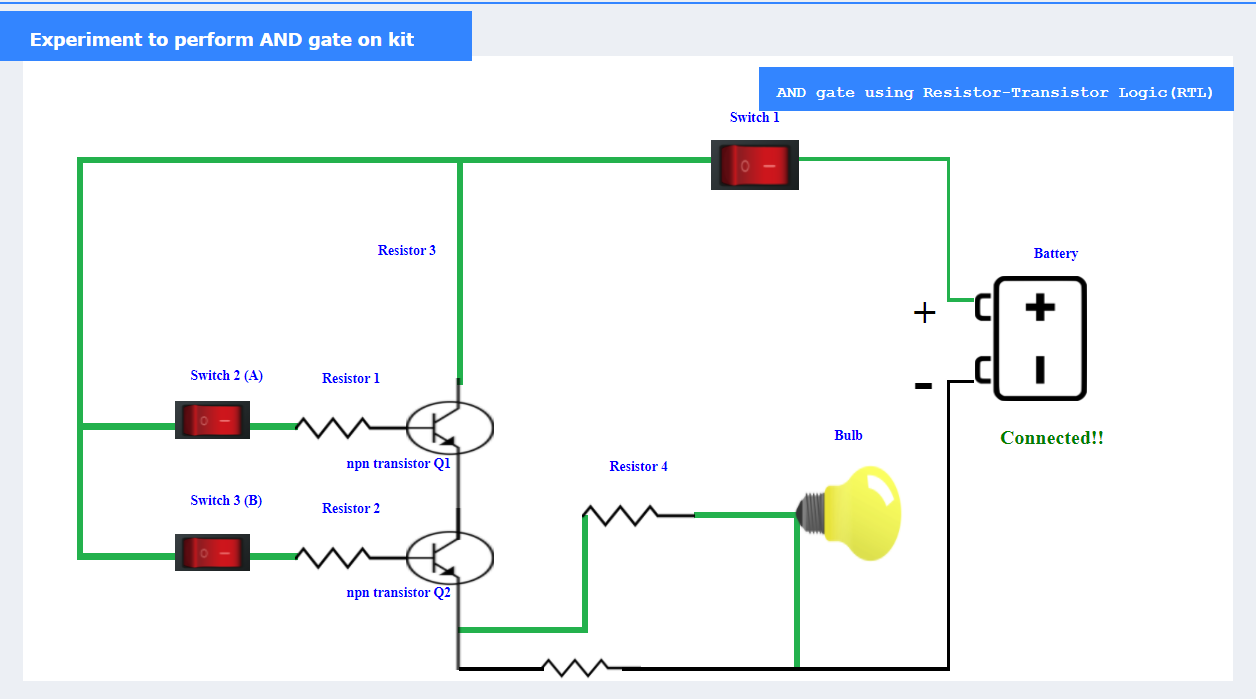
**2K19/CO/319**

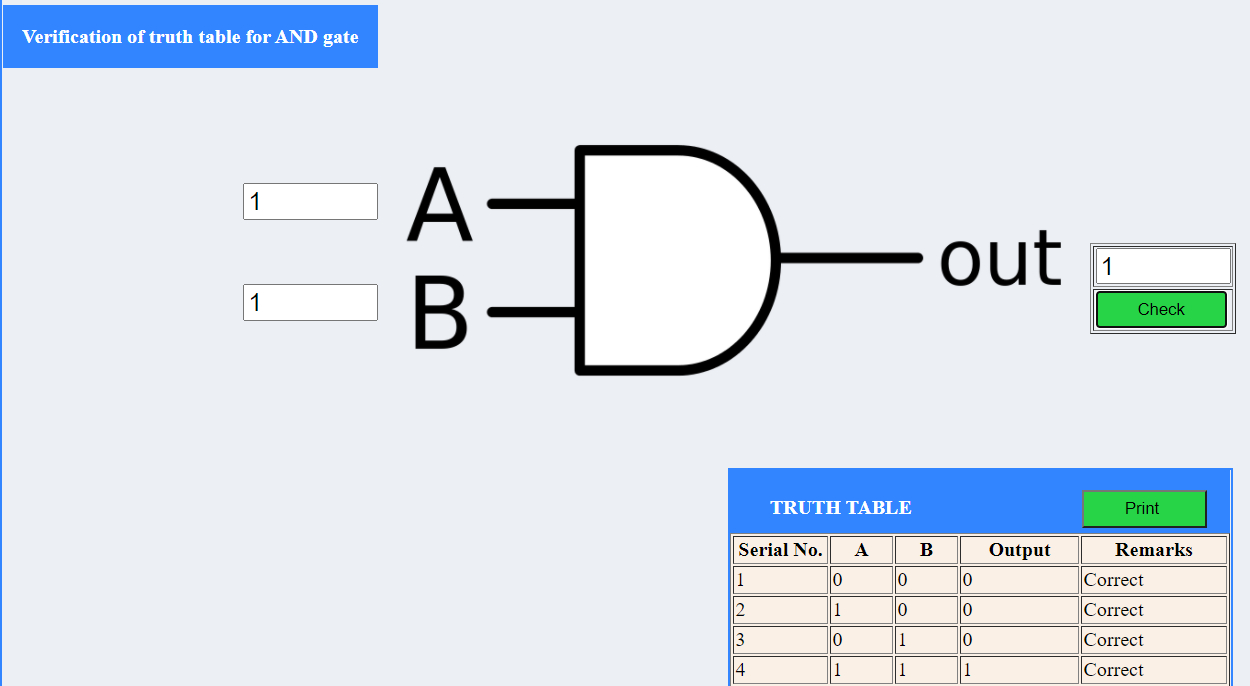
**EXPERIMENT-1**

### AIM: To verify and interpret the logic and truth table for AND, OR, NOT, NAND, NOR, Ex-OR, Ex-NOR gates using RTL (Resistor Transistor Logic), DTL (Diode Transistor Logic) and TTL (Transistor Transistor Logic) logics in simulator 1 and verify the truth table for AND, OR, NOT, NAND, NOR, Ex-OR, Ex-NOR gates in simulator 2.

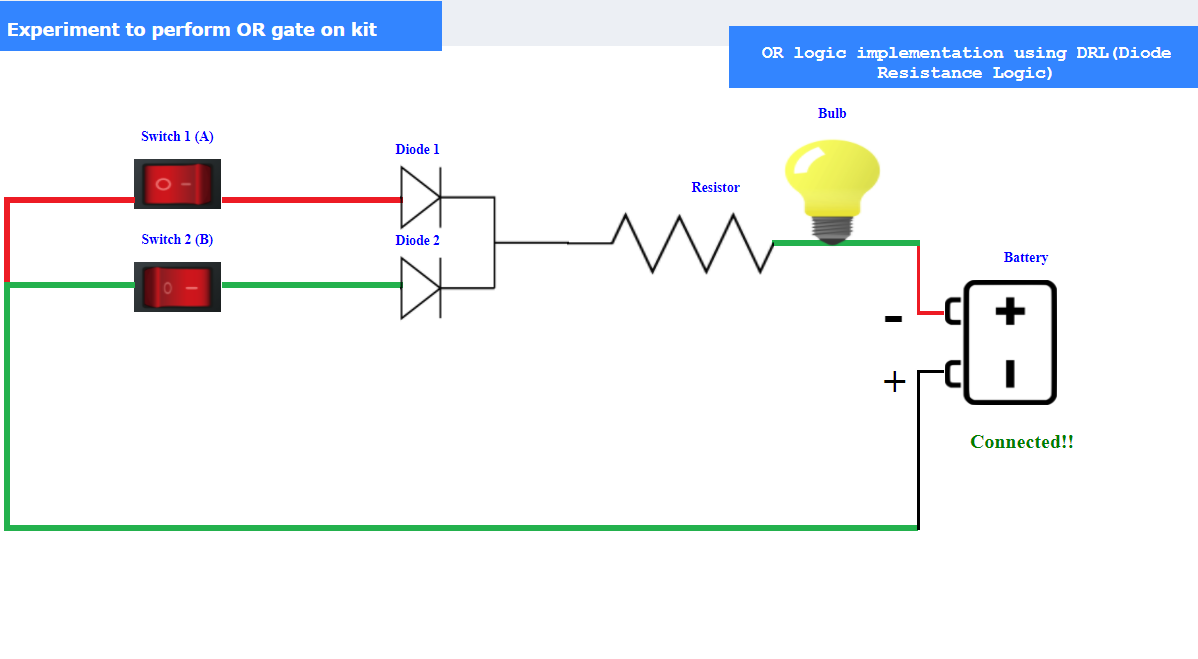
**INTRODUCTION:**

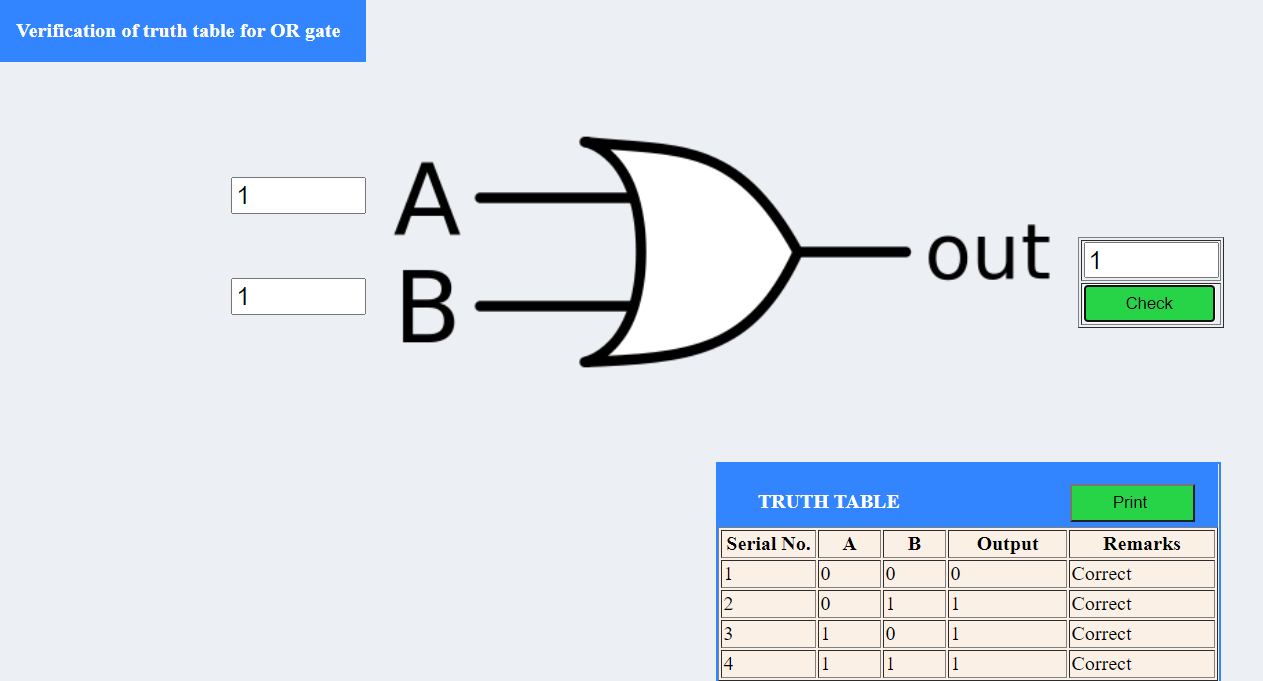
Logic gates are the basic building blocks of any digital system. Logic gates are electronic circuits having one or more than one input and only one output. The relationship between the input and the output is based on a certain logic. Based on this, logic gates are named as

1. AND gate
2. OR gate
3. NOT gate
4. NAND gate
5. NOR gate
6. Ex-OR gate
7. Ex-NOR gate
8. **AND gate:** The Logic AND Gate is a type of digital logic circuit whose output goes HIGH to a logic level 1 only when all of its inputs are HIGH.

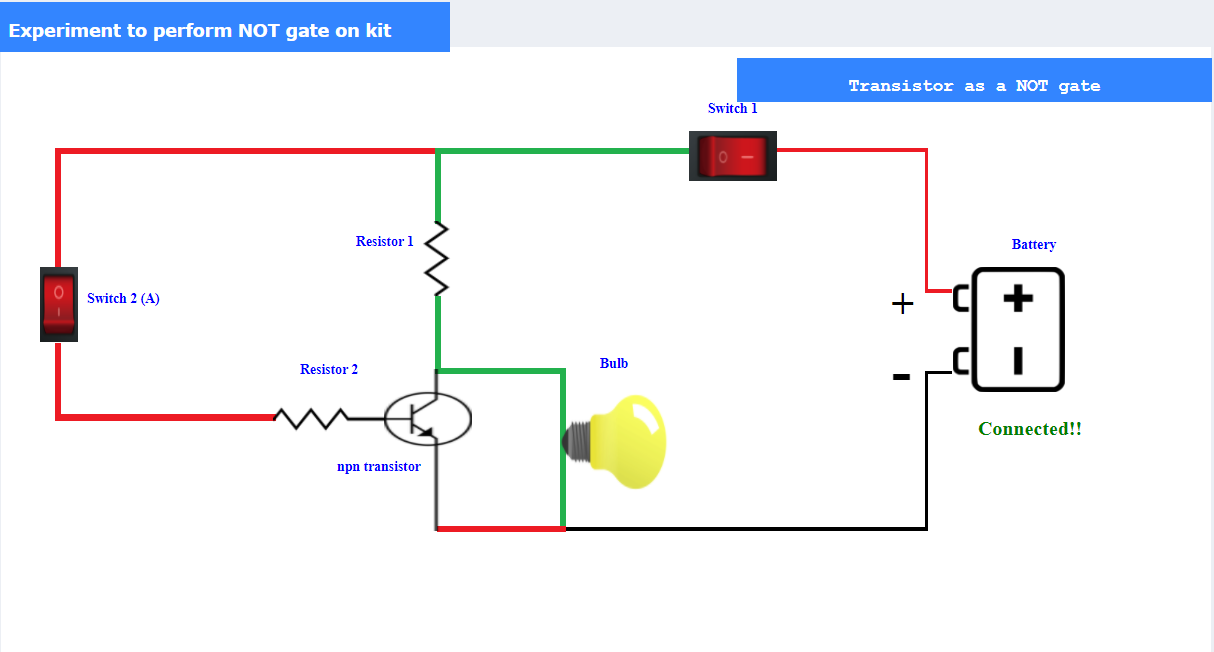


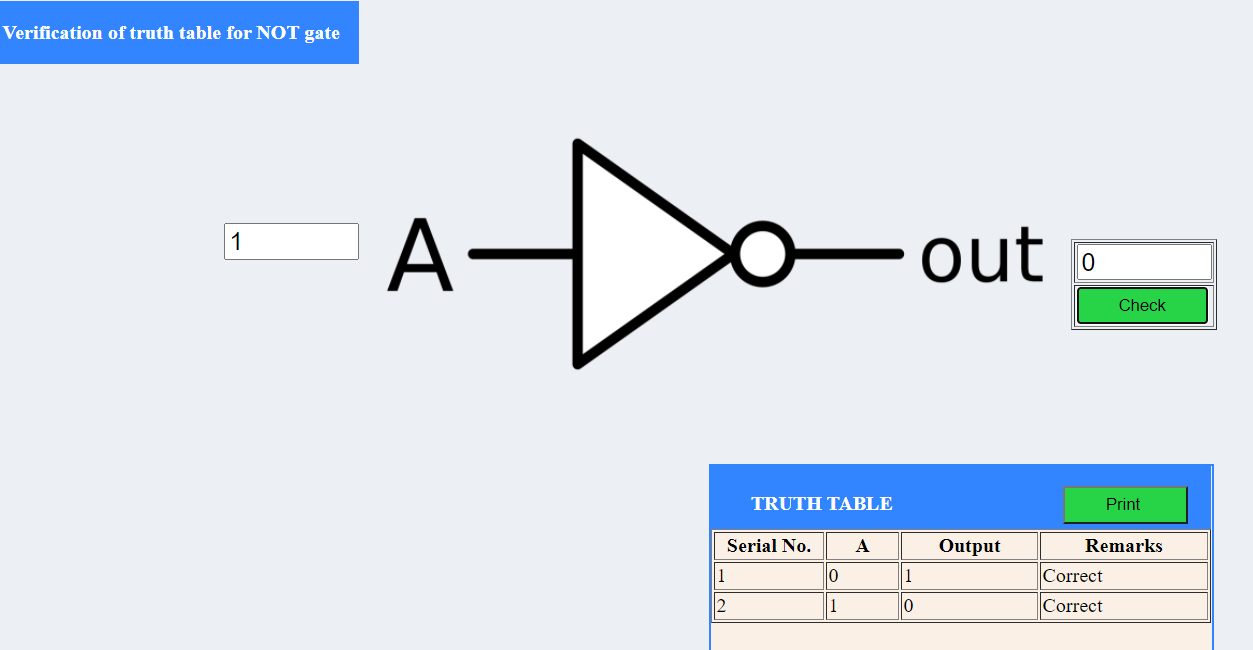
1. **OR gate**: The Logic OR Gate is a type of digital logic circuit whose output goes HIGH to a logic level 1 only when one or more of its inputs are HIGH.



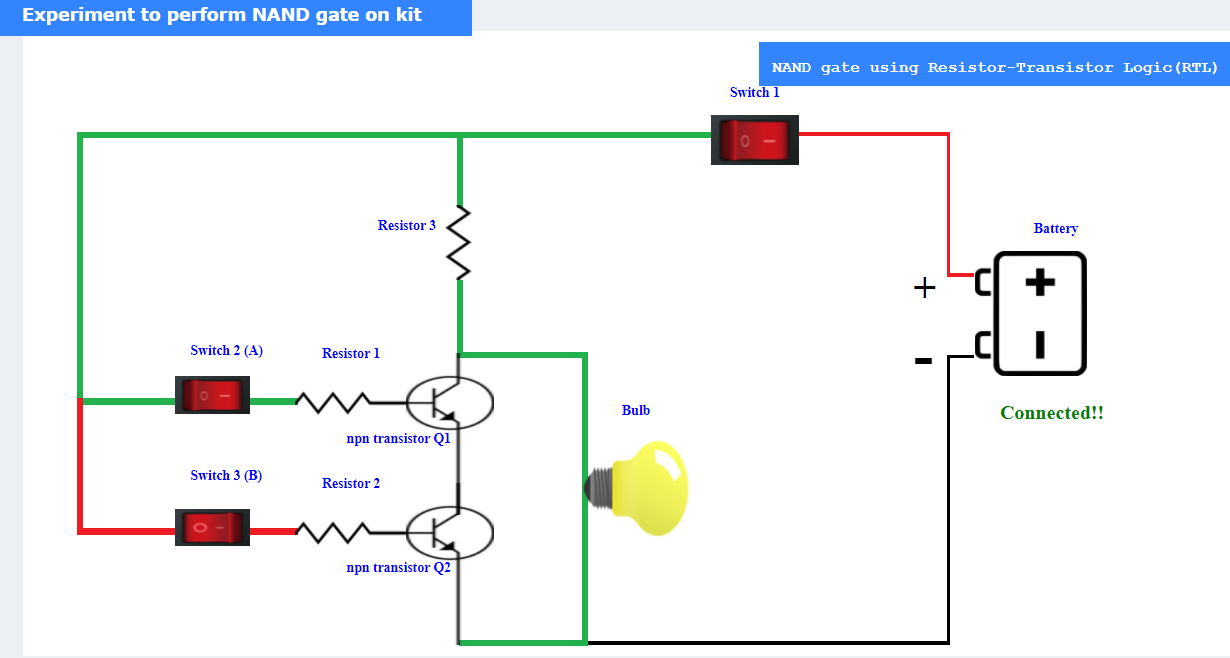


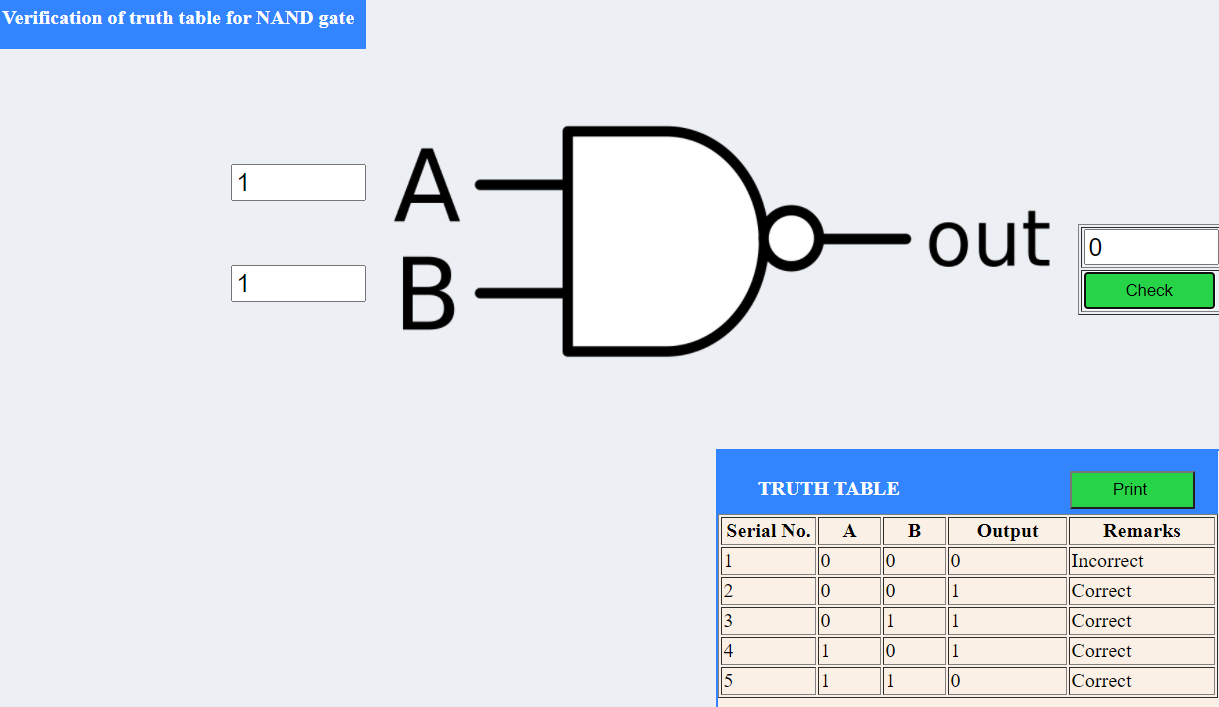
1. **NOT gate:** The Logic NOT Gate is the most basic of all the logical gates and is often referred to as an Inverting Buffer or simply an Inverter.



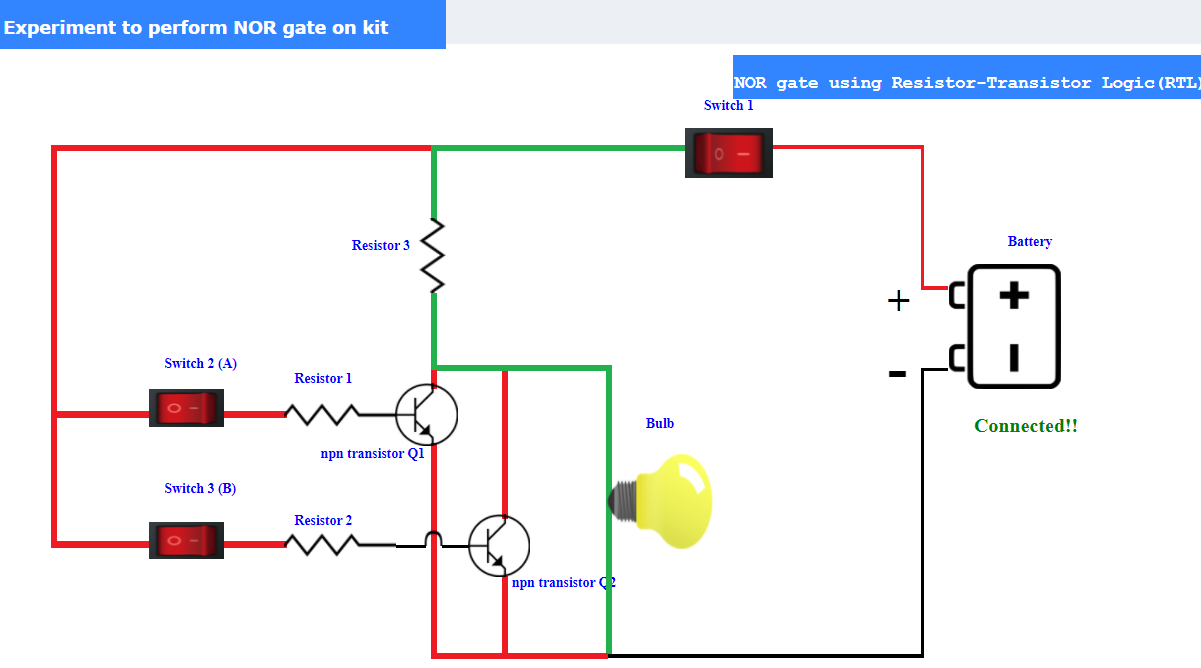


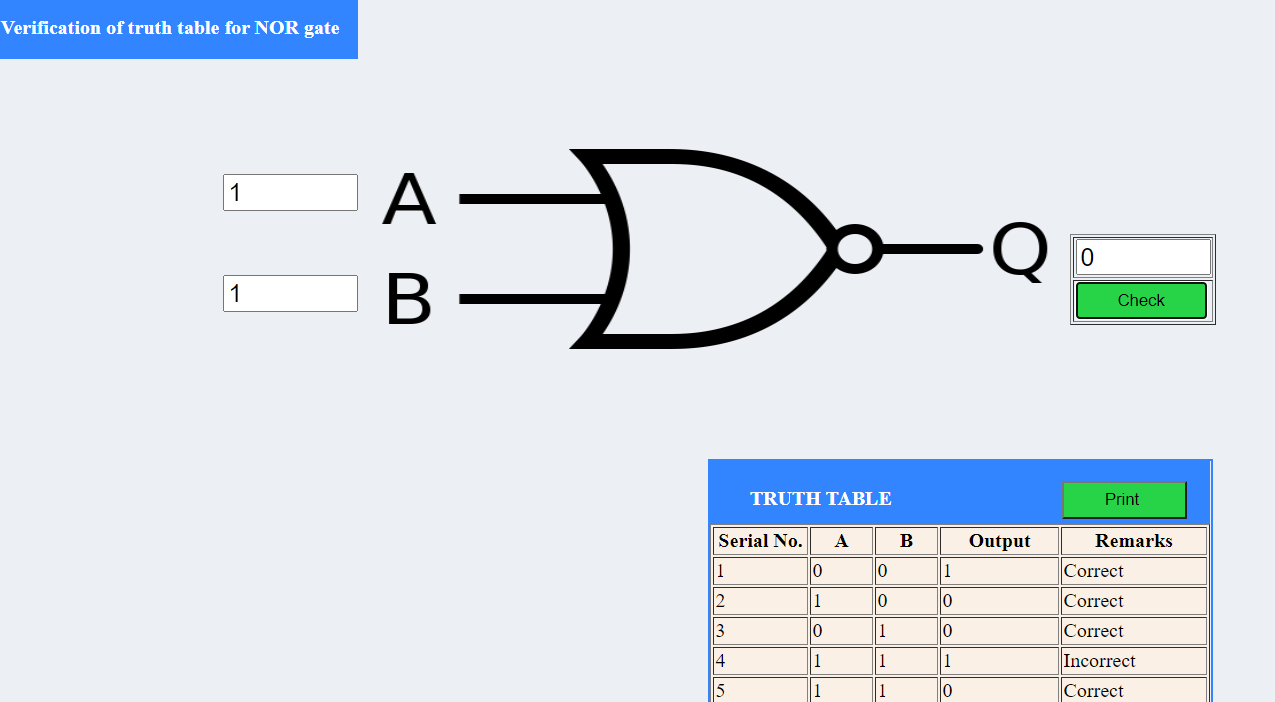
1. **NAND gate:** The Logic NAND Gate is a combination of a digital logic AND gate and a NOT gate connected together in series.



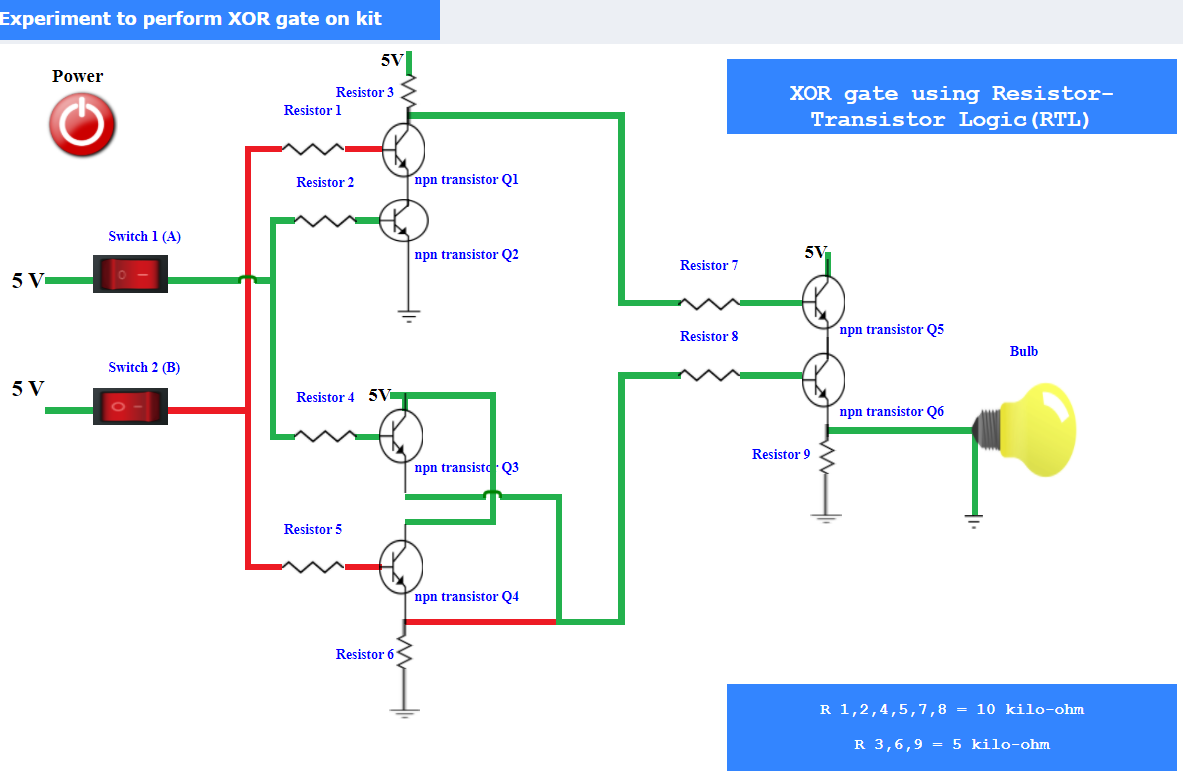


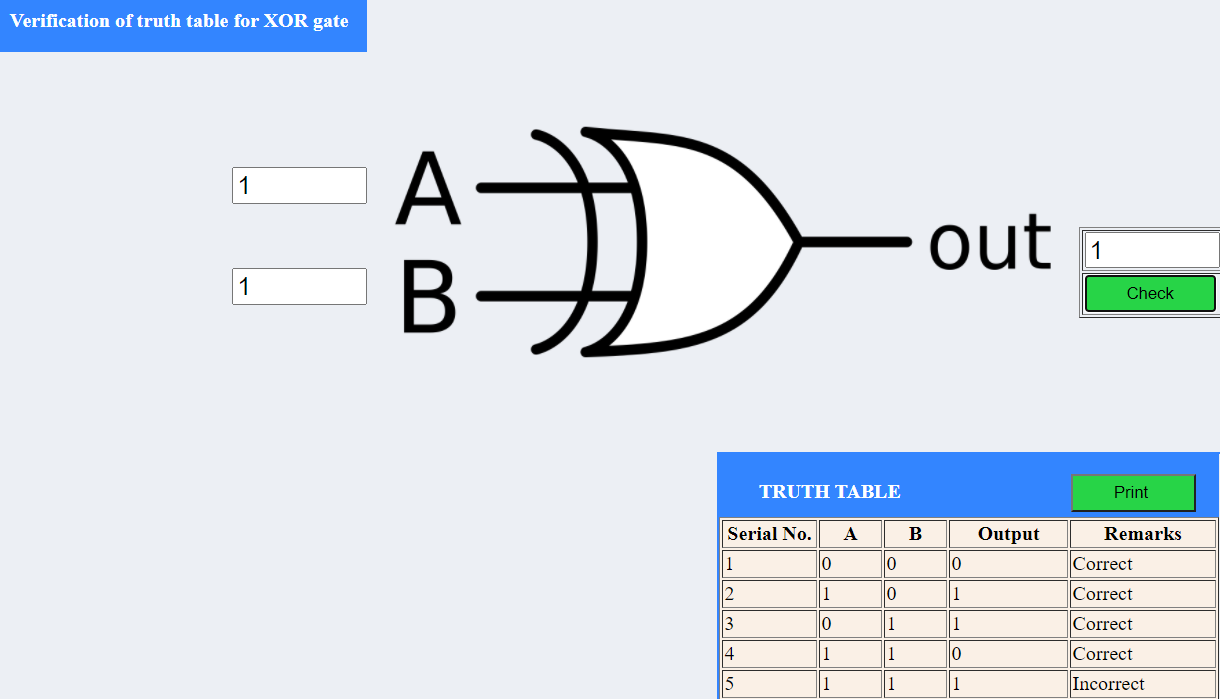
1. **NOR gate:** The Logic NOR Gate is a combination of the digital logic OR gate and an inverter or NOT gate connected together in series.



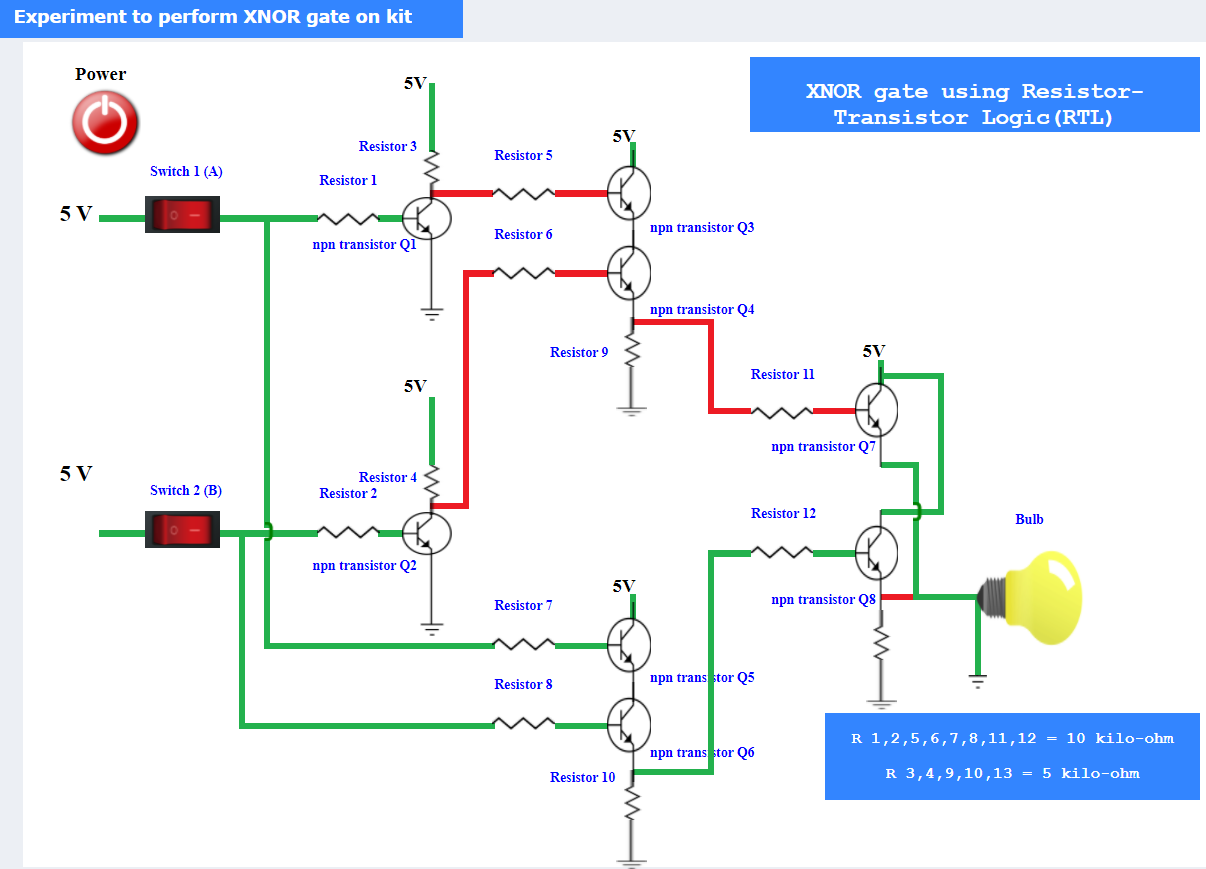


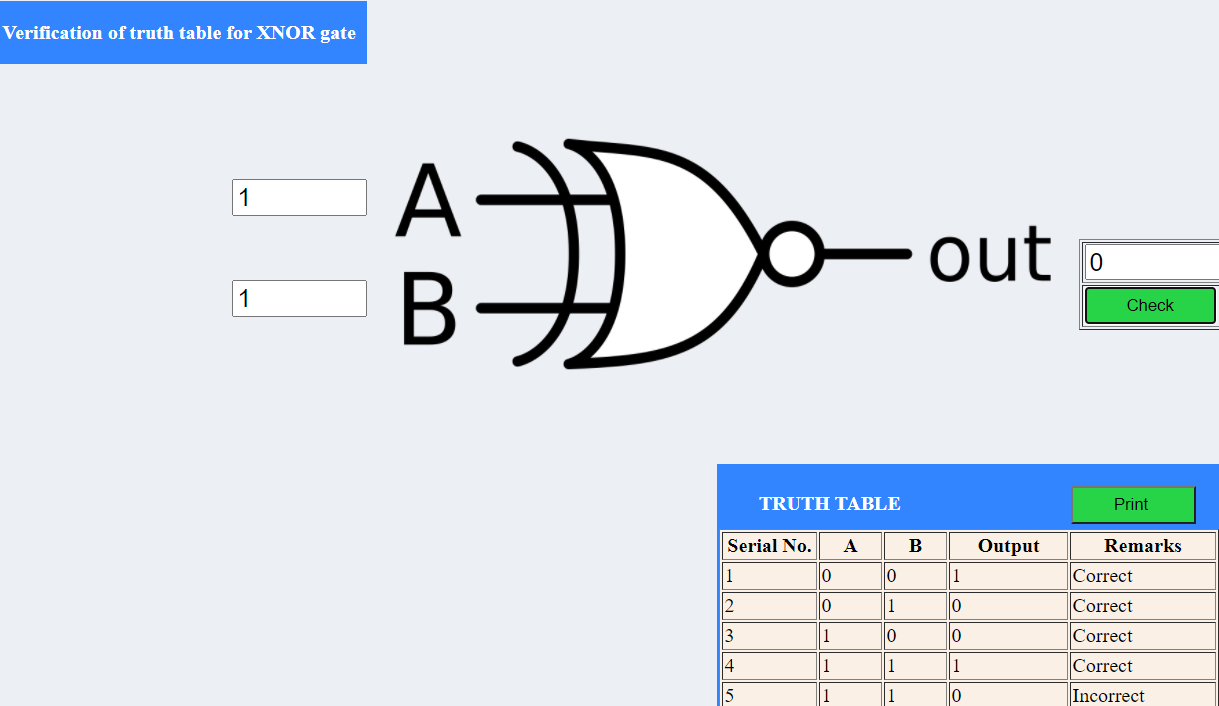
1. **EX-OR gate:** Exclusive OR gate is a digital logic gate whose output goes high to a logic level 1 only when both the inputs are different.



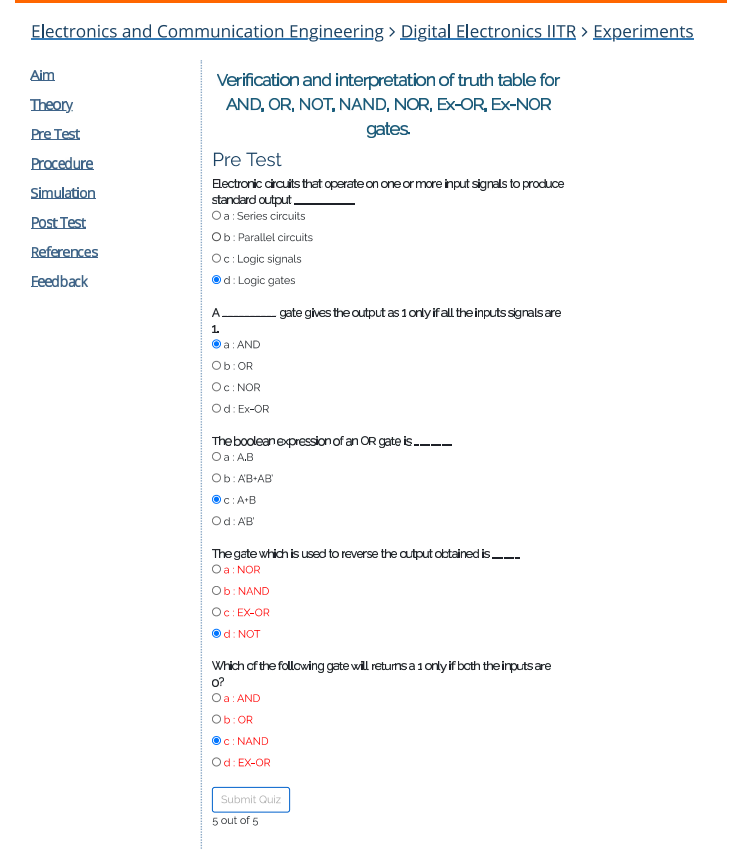


1. **EX-NOR gate:** Exclusive NOR gate is a digital logic gate whose output goes high to a logic level 1 only when both the inputs are same.

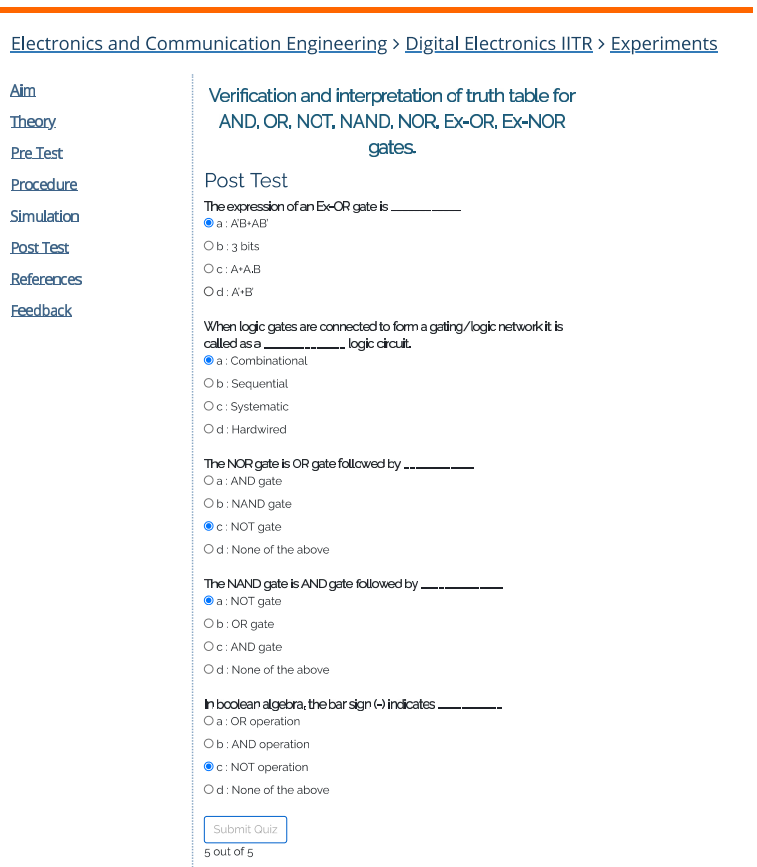




**PRE-TEST:**



**POST-TEST:**

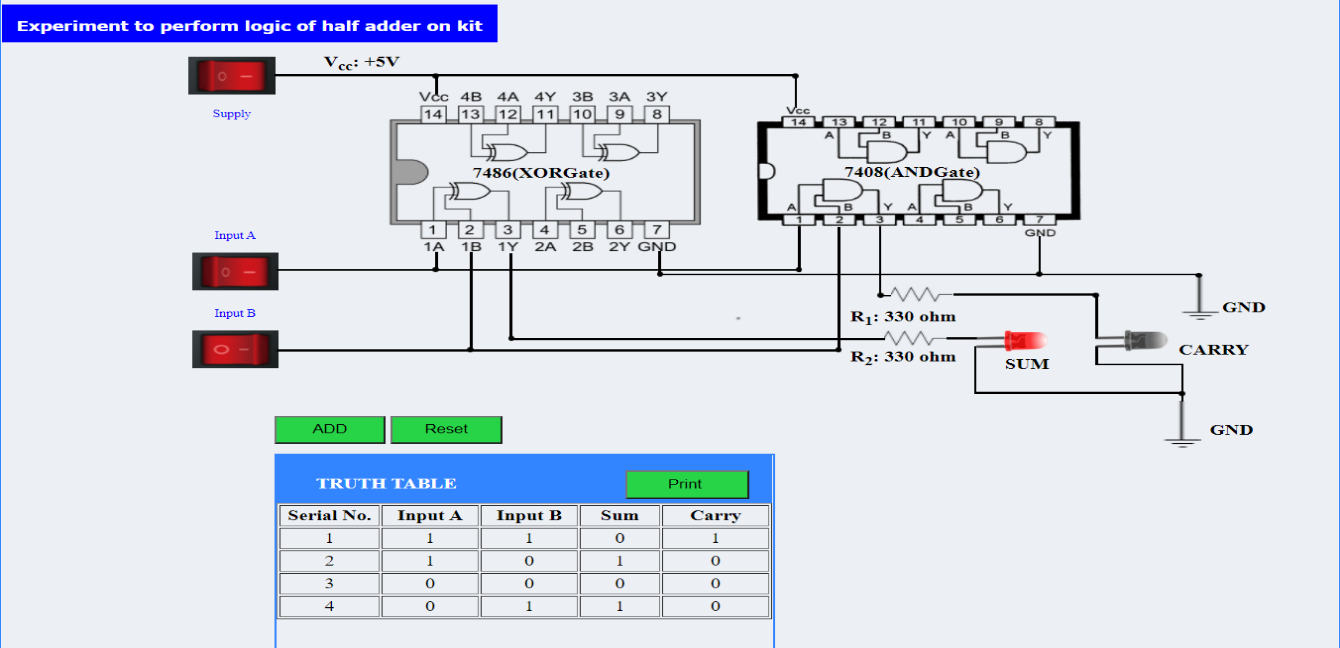


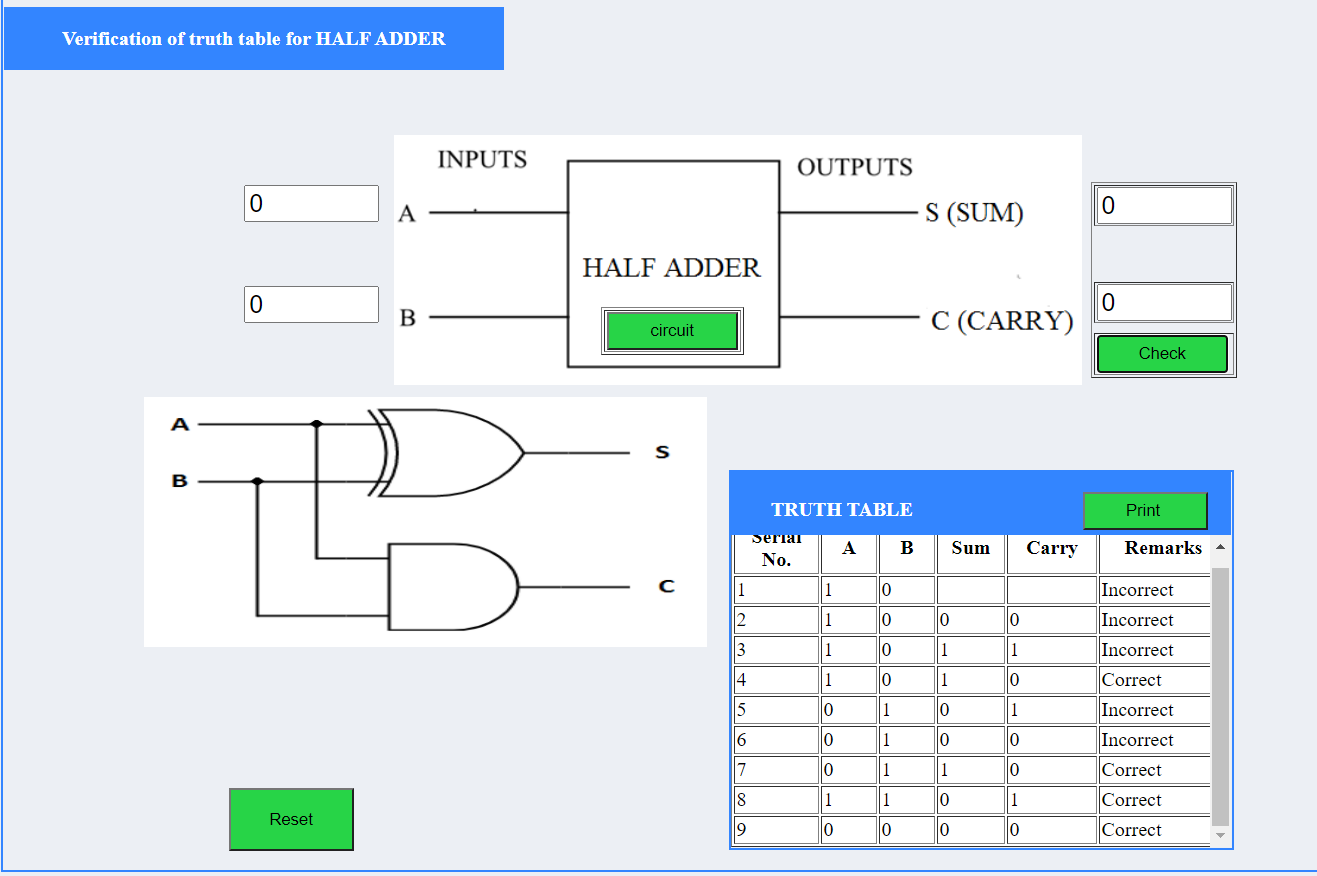
### EXPERIMENT-2

**AIM:** To verify the truth table of half adder and full adder by using XOR and NAND gates respectively and analyze the working of half adder and full adder circuit with the help of LEDs in simulator 1 and verify the truth table only of half adder and full adder in simulator 2.

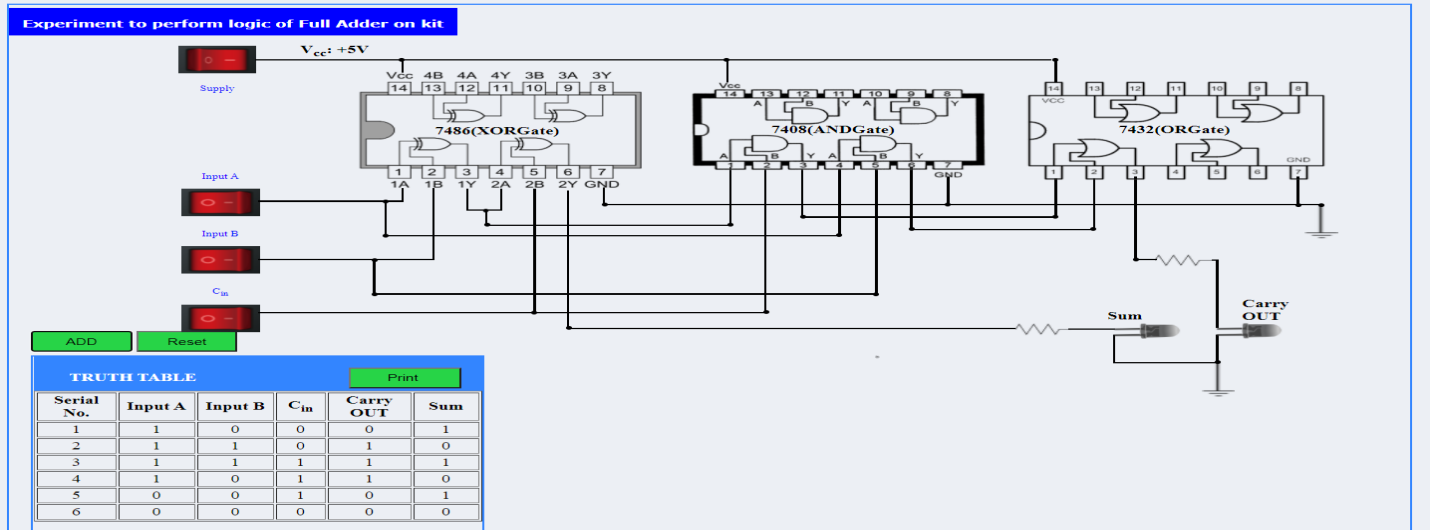
**INTRODUCTION:**

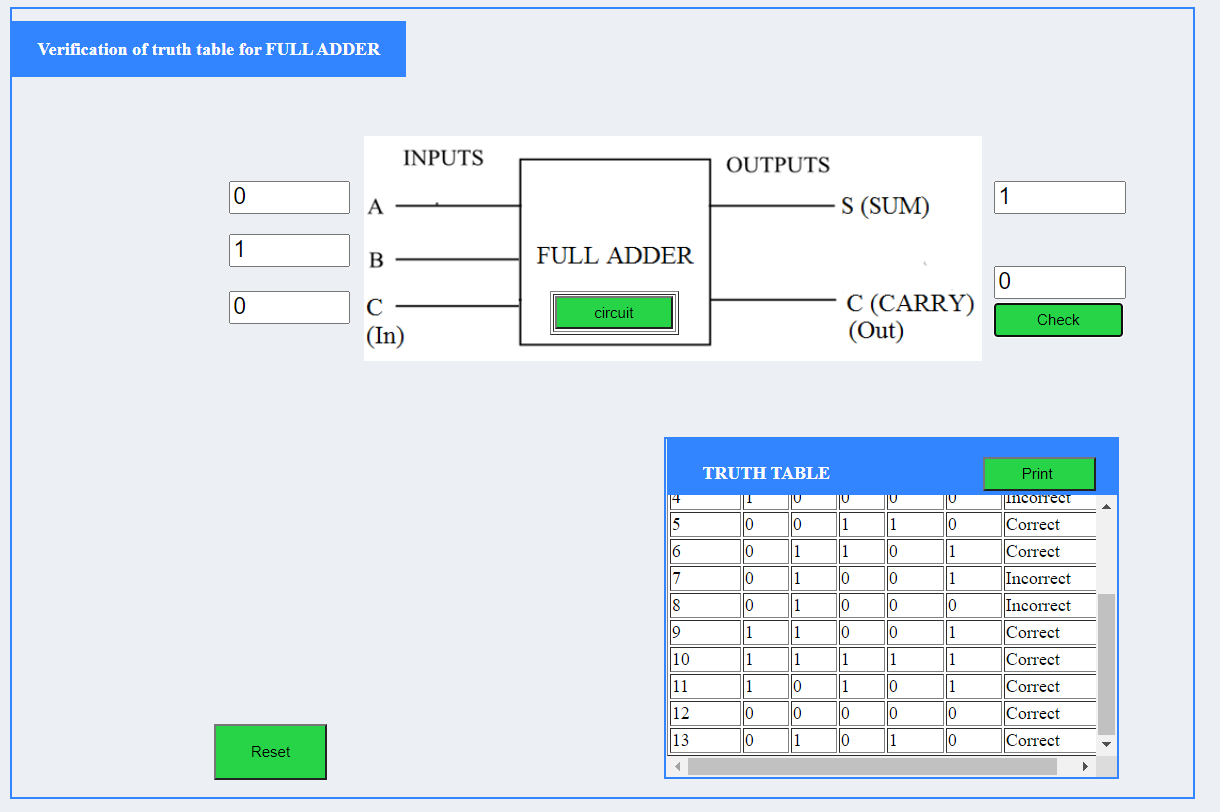
**Half Adder:** Half adder is a combinational circuit that performs simple addition of two binary numbers. If we assume A and B as the two bits whose addition is to be performed, The block diagram and a truth table for half adder with A, B as inputs and Sum, carry as outputs can be tabulated as follows





**Full adder:** Full adder is a digital circuit used to calculate the sum of three binary bits. Full adders are complex and difficult to implement when compared to half adders. Two of the three bits are same as before which are A, the augend bit and B, the addend bit. The additional third bit is carried bit from the previous stage and is called 'Carry' – in generally represented by CIN. It calculates the sum of three bits along with the carry. The output carry is called Carry – out and is represented by Carry OUT.

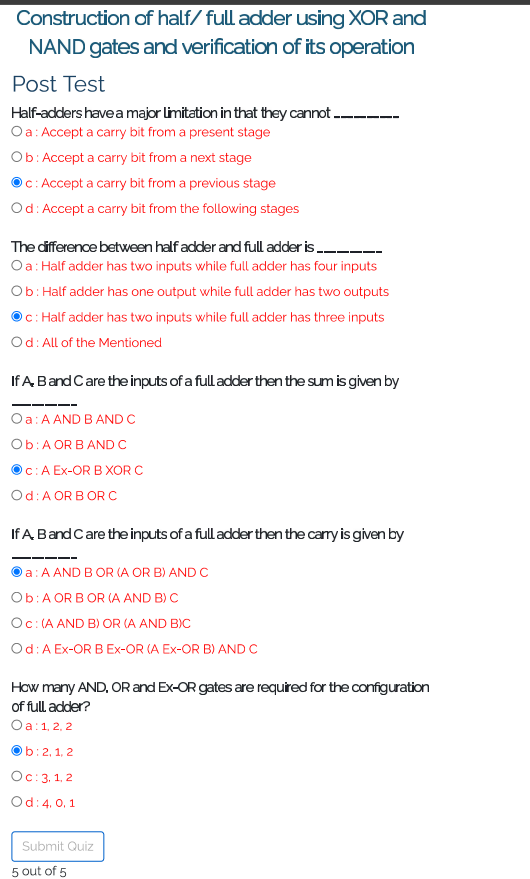




**PRE-TEST:**



**POST-TEST:**



**Experiment-3**

### AIM:

### To verify the truth table of half subtractor by using the ICs of XOR, NOT and AND gates and of full subtractor by using the ICs of XOR, AND, NOT and OR gates respectively and analyze the working of half subtractor and full subtractor circuit with the help of LEDs in simulator 1 and verify the truth table only of half subtractor and full subtractor in simulator 2.

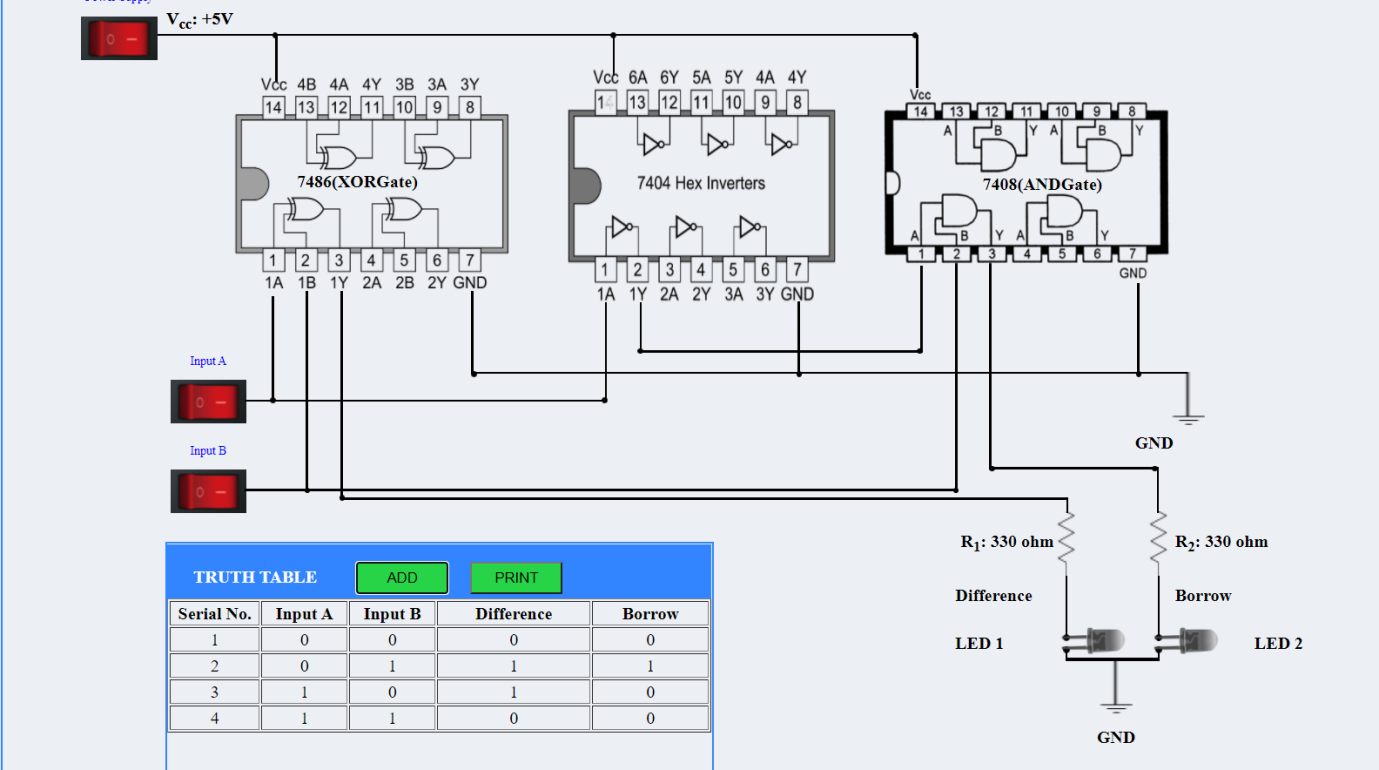
### INTRODUCTION:

### Subtractor circuits take two binary numbers as input and subtract one binary number input from the other binary number input. Similar to adders, it gives out two outputs, difference and borrow (carry-in the case of Adder). There are two types of subtractors.

**1) Half Subtractor  
2) Full Subtractor**

### 1) Half Subtractor:

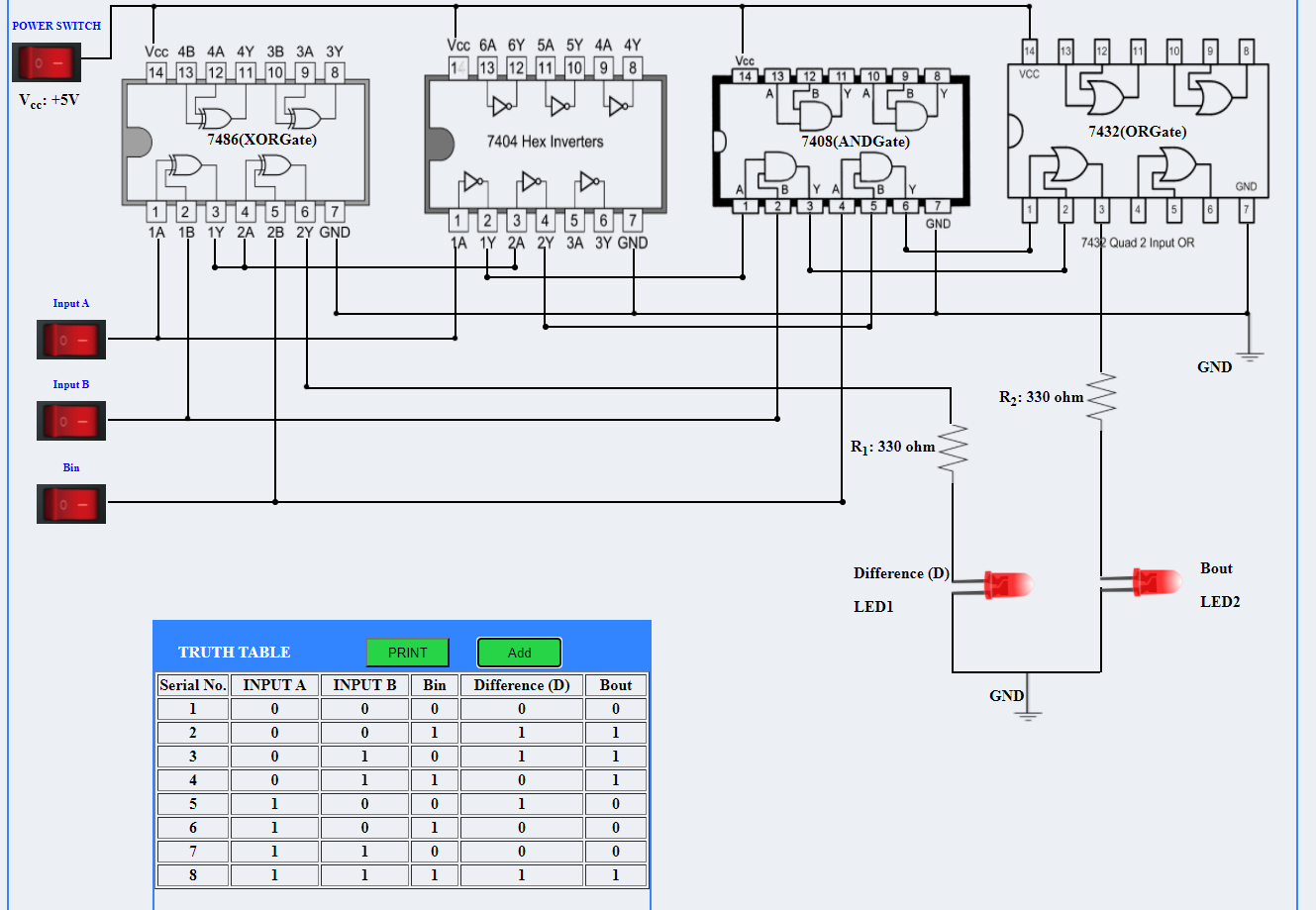
The half-subtractor is a combinational circuit which is used to perform subtraction of two bits. It has two inputs, A (minuend) and B (subtrahend) and two outputs Difference and Borrow. The logic symbol and truth table are shown below.



### 

### 2) Full Subtractor:

A full subtractor is a combinational circuit that performs subtraction involving three bits, namely A (minuend), B (subtrahend), and Bin (borrow-in) . It accepts three inputs: A (minuend), B (subtrahend) and a Bin (borrow bit) and it produces two outputs: D (difference) and Bout (borrow out). The logic symbol and truth table are shown below.



**Pre test:**

**Post test:**

### 

### PRE-TEST

### POST-TEST

### 

### Experiment-4

**Aim:**

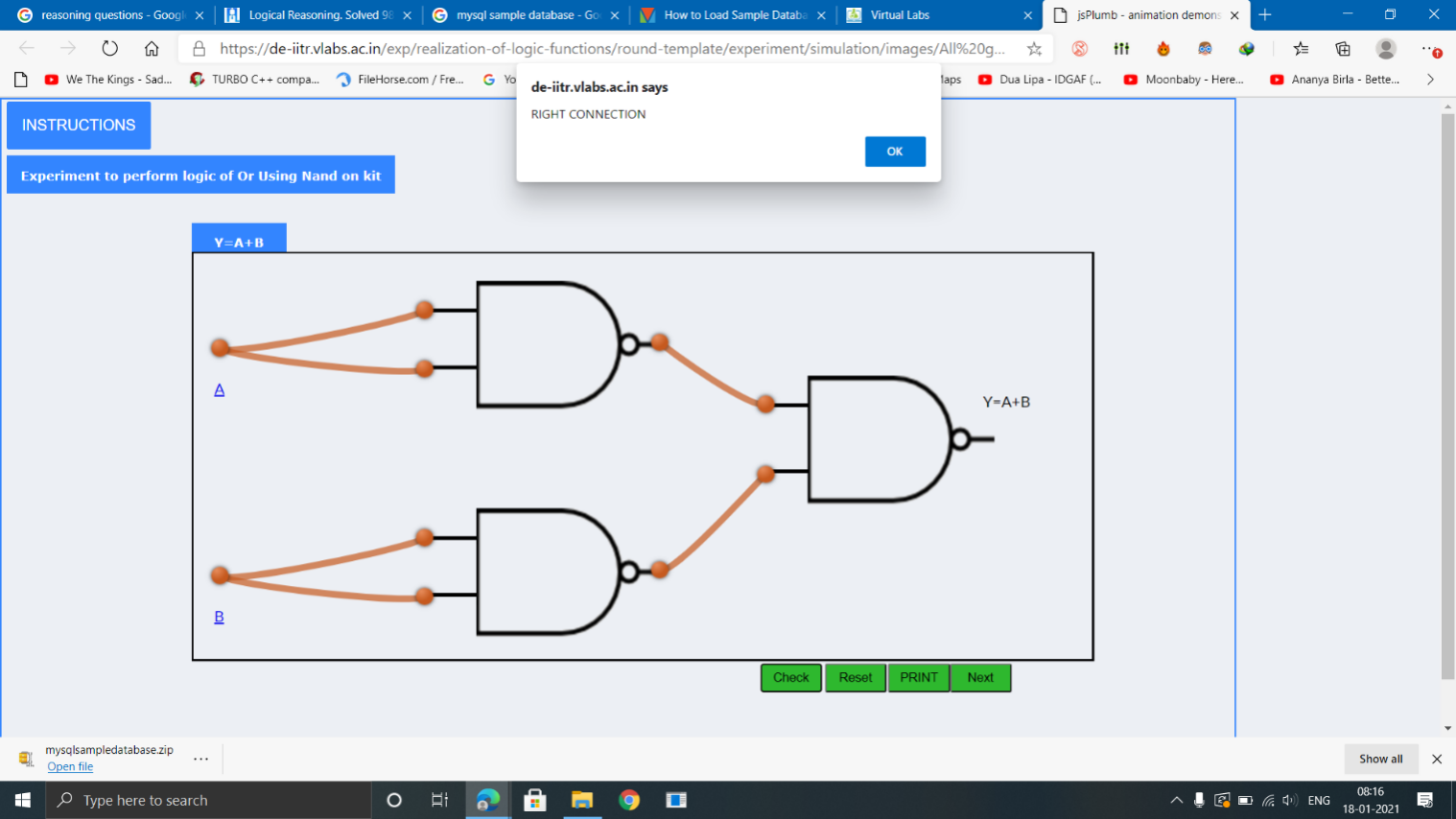
#### To implement the logic functions i.e. AND, OR, NOT, Ex-OR, Ex- NOR and a logical expression with the help of NAND and NOR universal gates respectively.

**Introduction:**

Logic gates are electronic circuits which perform logical functions on one or more inputs to produce one output. There are seven logic gates. When all the input combinations of a logic gate are written in a series and their corrresponding outputs written along them, then this input/ output combination is called Truth Table.

### Nand gate as Universal gate: NAND gate is actually a combination of two logic gates i.e., AND gate followed by NOT gate. So its output is complement of the output of an AND gate. This gate can have minimum two inputs. By using only NAND gates, we can realize all logic functions: AND, OR, NOT, Ex-OR, Ex-NOR, NOR. So, this gate is also called as universal gate.

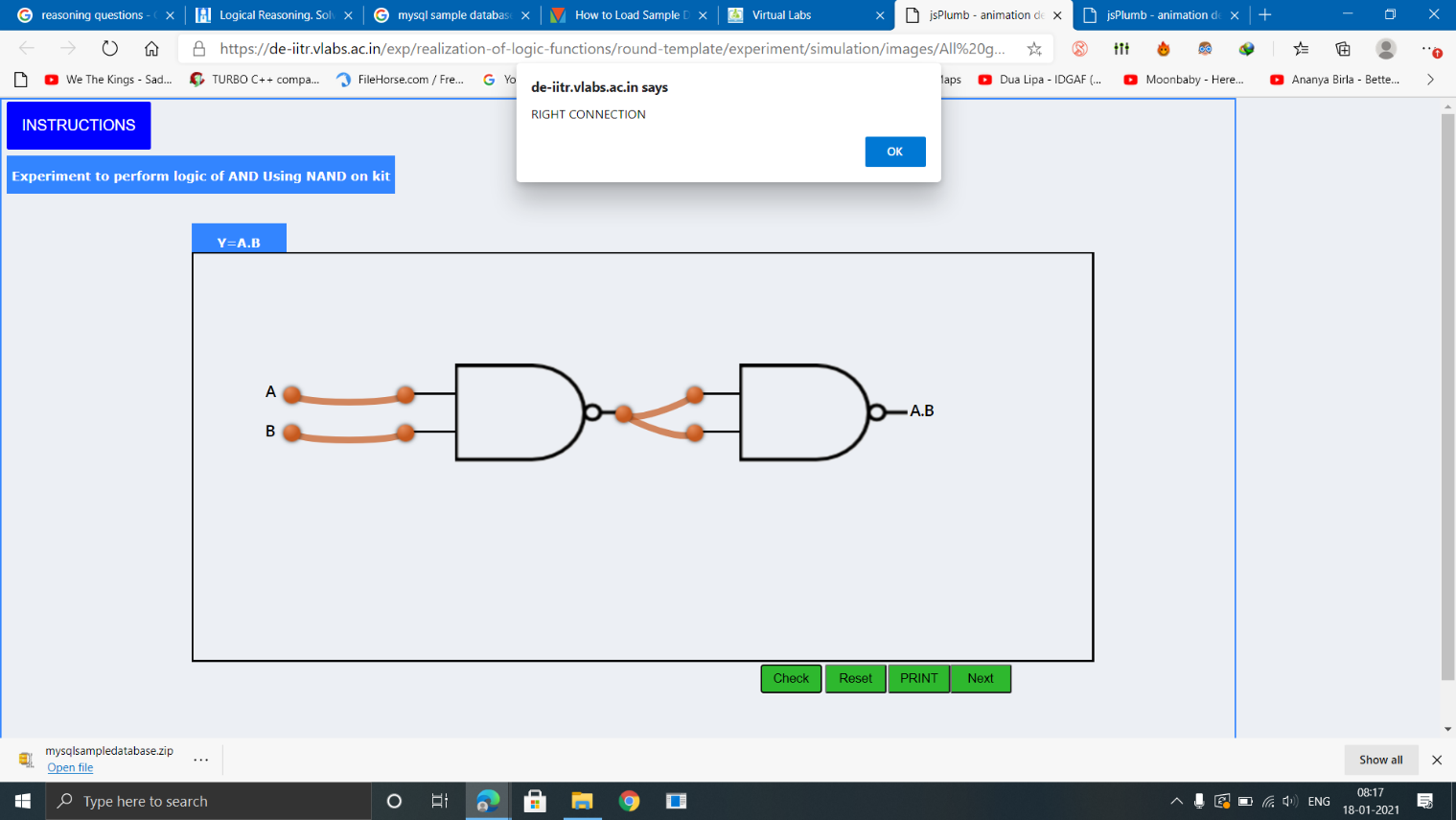
### NAND gates as OR gate:



#### **2) NAND gates as AND gate:**

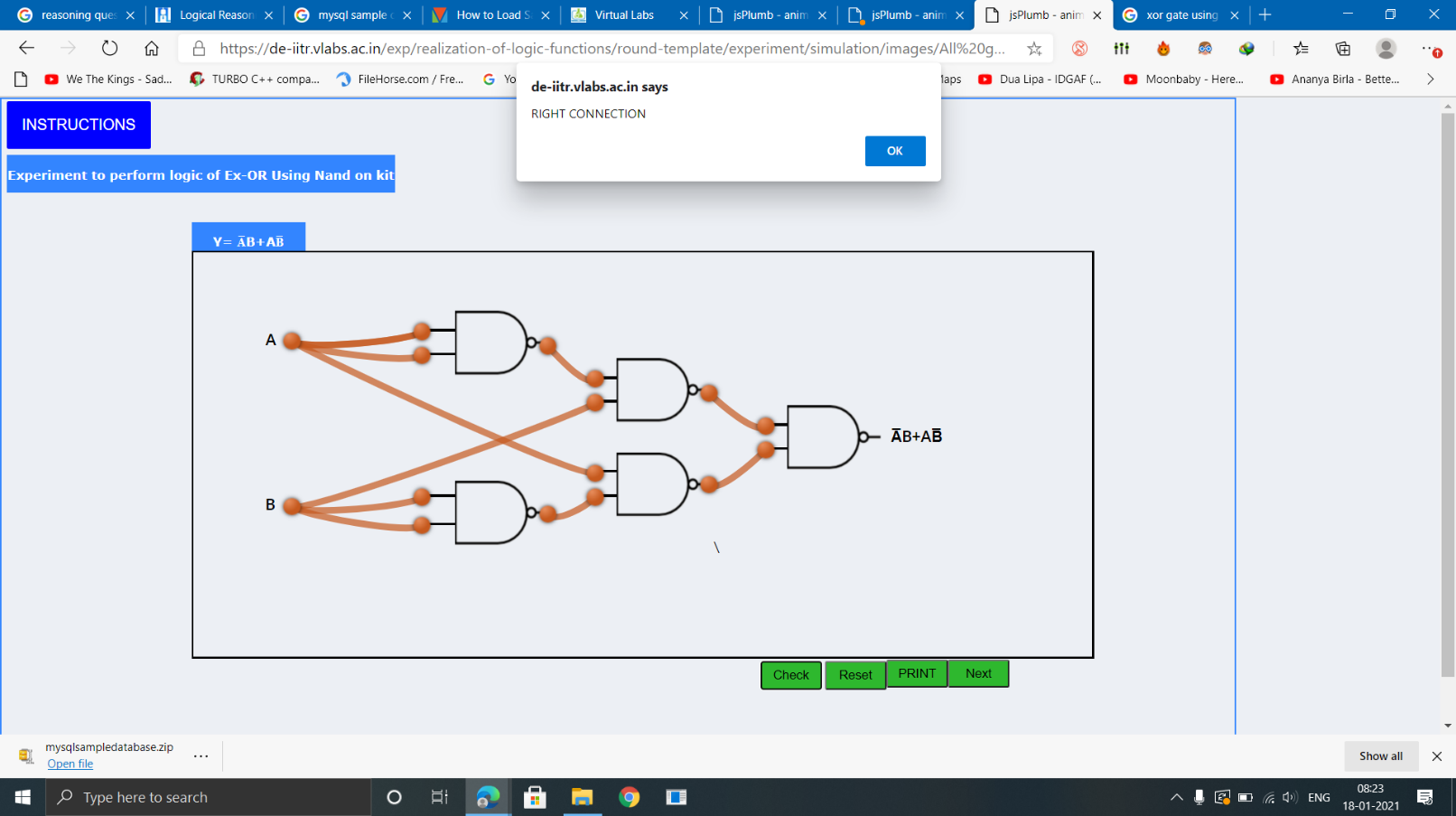
A NAND produces complement of AND gate. So, if the output of a NAND gate is inverted, overall output will be that of an AND gate.

Y = ((A.B)’)’  
Y = (A.B)



#### **3) NAND gates as Ex-OR gate:**

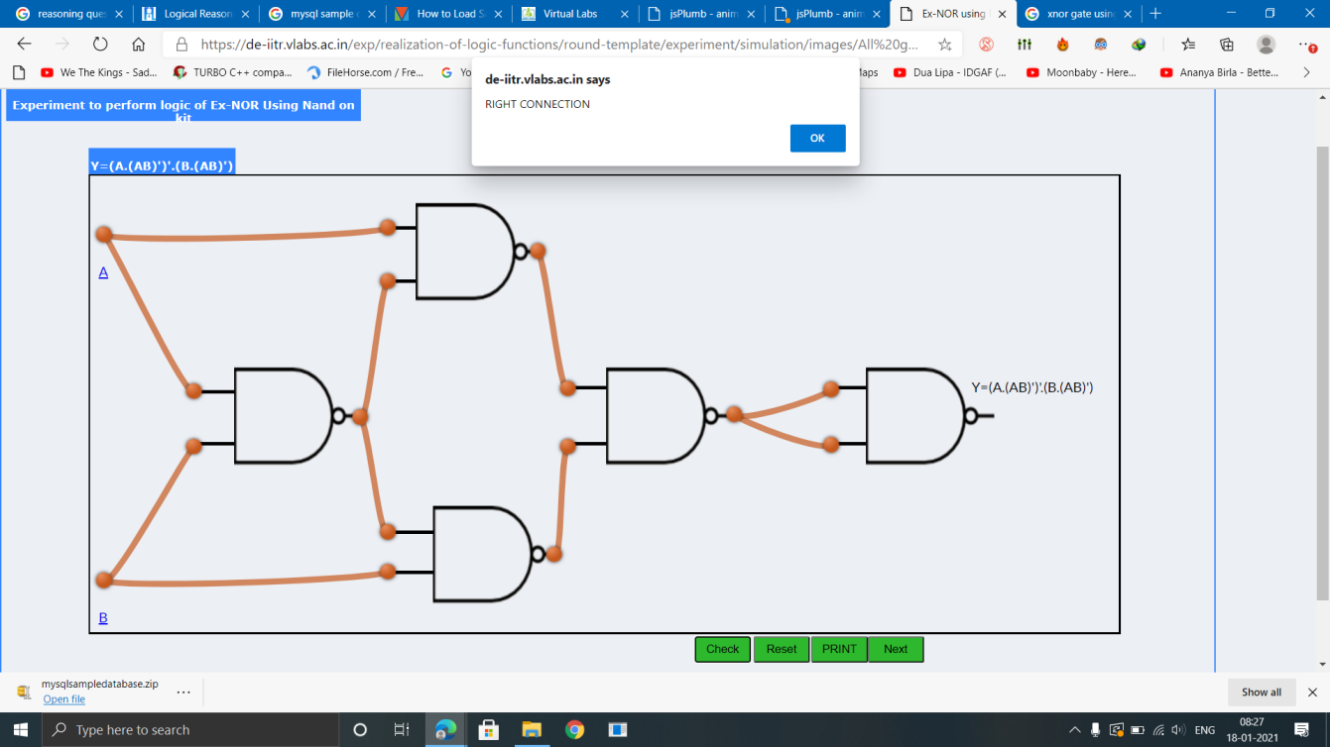
The output of a two input Ex-OR gate is shown by: Y = A’B + AB’. This can be achieved with the logic diagram shown in the left side.



#### **4) NAND gates as Ex-NOR gate:**

Ex-NOR gate is actually Ex-OR gate followed by NOT gate. So give the output of Ex-OR gate to a NOT gate, overall output is that of an Ex-NOR gate.

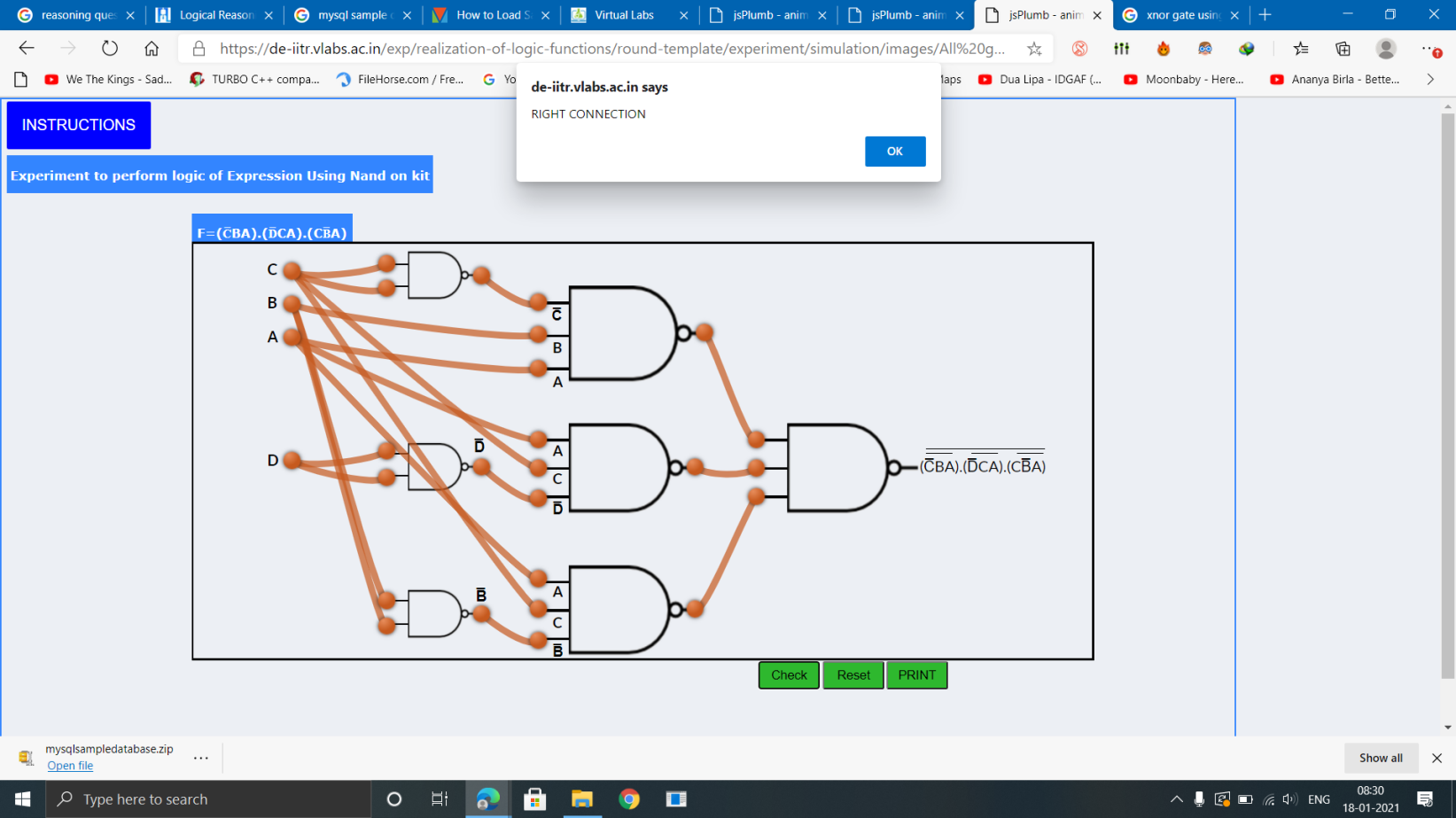
Y = AB+ A’B’



#### **5) Implementing the simplified function with NAND gates only:**

We can now start constructing the circuit. First note that the entire expression is inverted and we have three terms ANDed. This means that we must use a 3-input NAND gate. Each of the three terms is, itself, a NAND expression. Finally, negated single terms can be generates with a 2-input NAND gate acting as an inverted. Figure 8 illustrates a circuit using NAND gates only.

F=((C'.B.A)'(D'.C.A)'(C.B'.A)')'

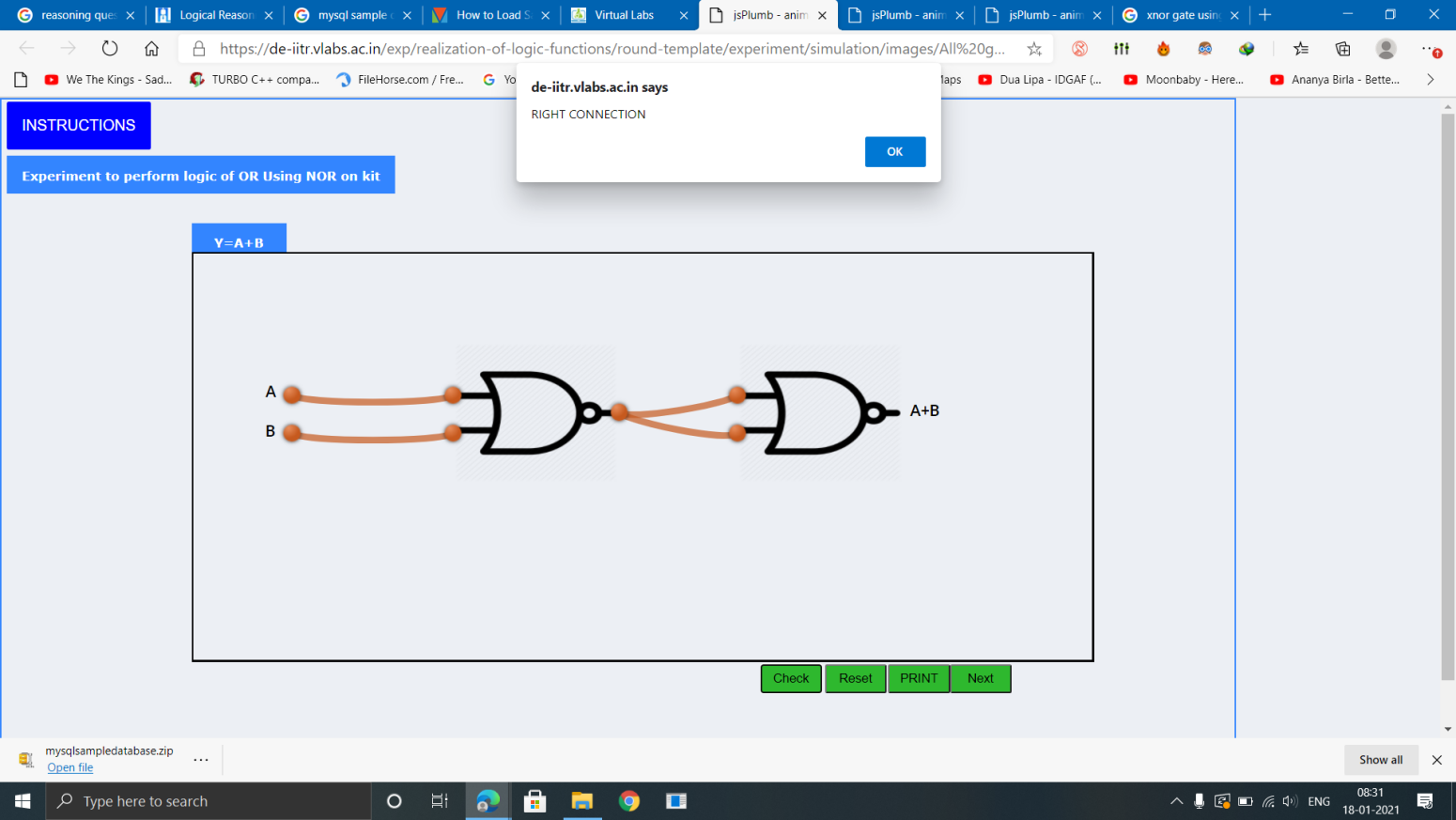
  
Implementing the simplified function with NAND gates only

### NOR gate as Universal Gate:

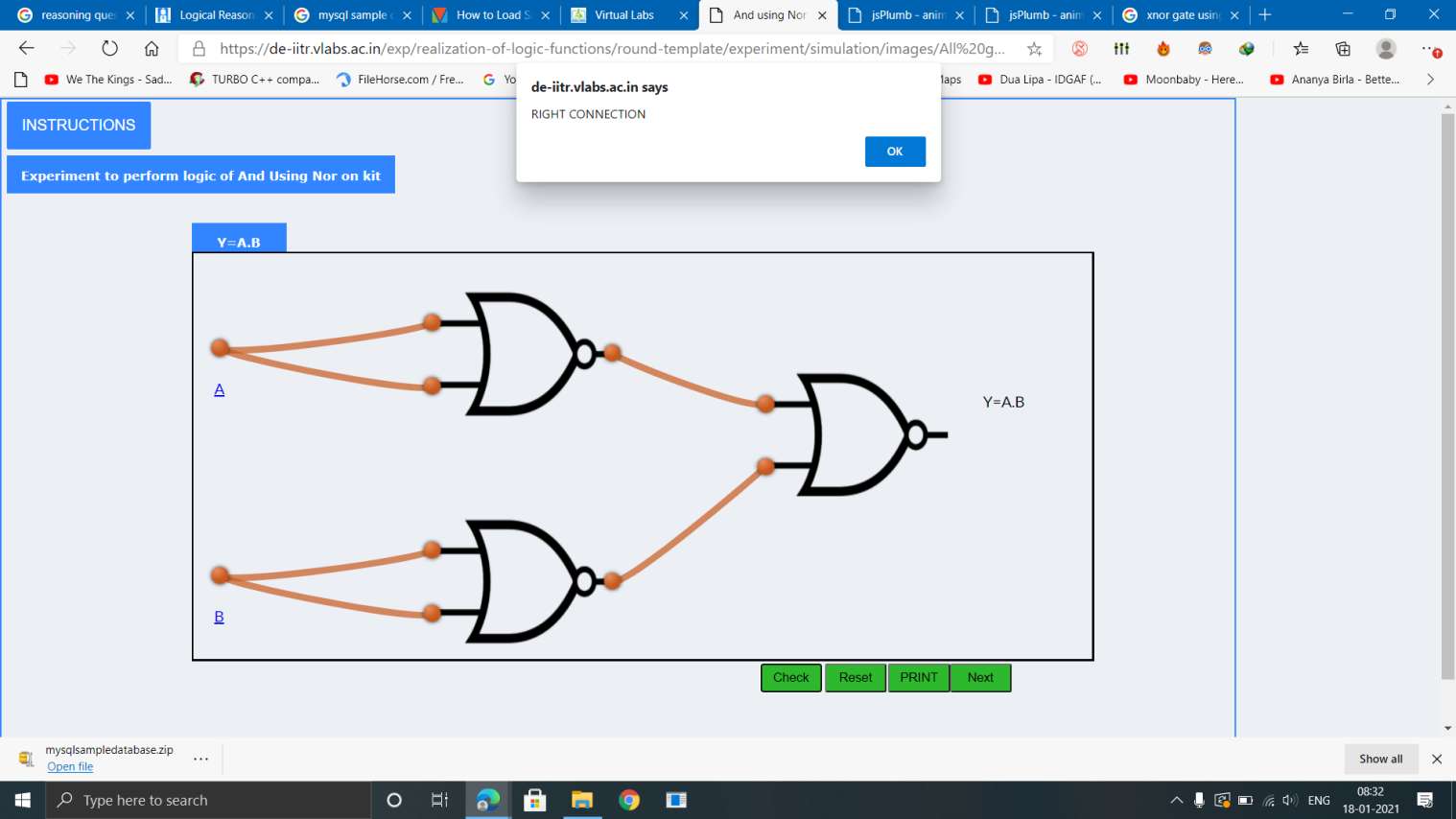
NOR gate is actually a combination of two logic gates: OR gate followed by NOT gate. So its output is complement of the output of an OR gate.This gate can have minimum two inputs, output is always one. By using only NOR gates, we can realize all logic functions: AND, OR, NOT, Ex-OR, Ex-NOR, NAND. So this gate is also called universal gate.

#### **1) NOR gates as OR gate:**

A NOR produces complement of OR gate. So, if the output of a NOR gate is inverted, overall output will be that of an OR gate.

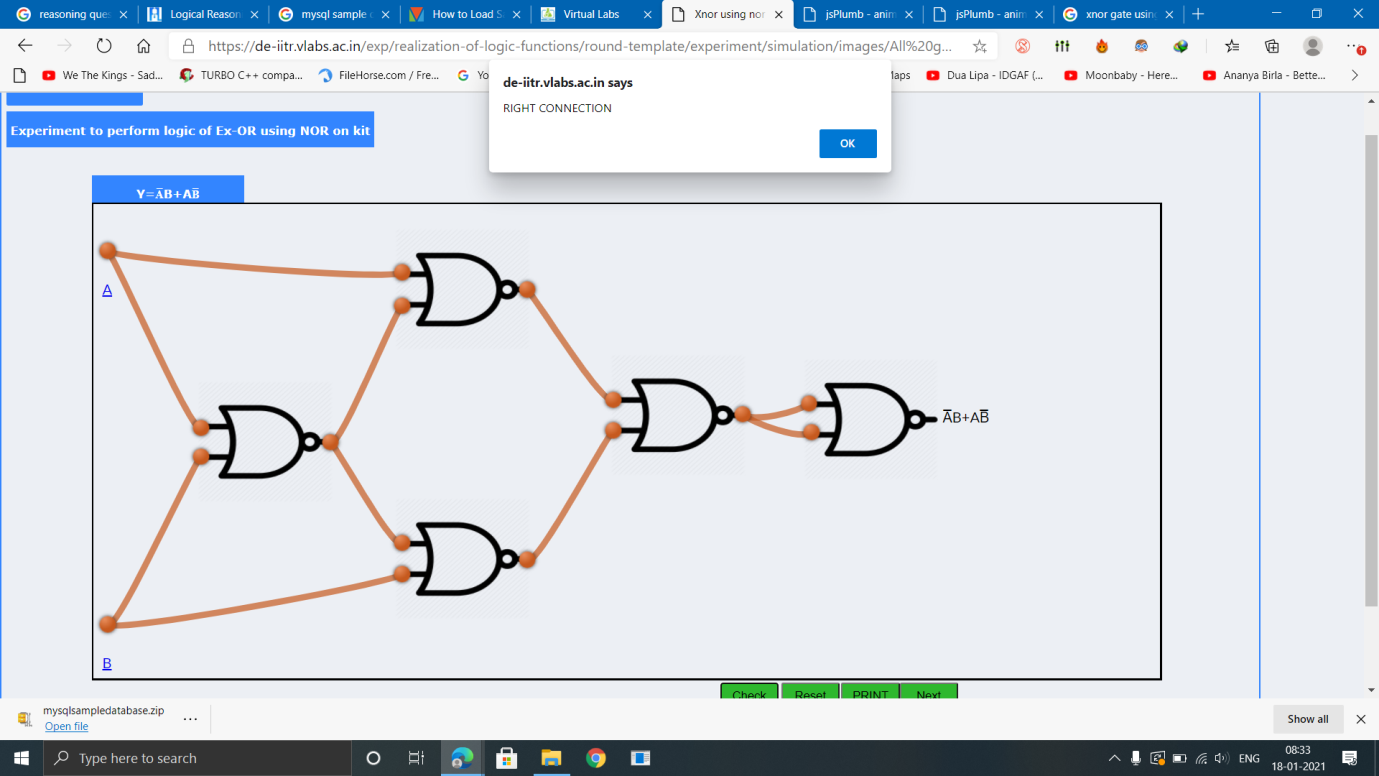
Y = ((A+B)’)’  
Y = (A+B)  


#### **2) NOR gates as AND gate:**



#### **3) NOR gates as Ex-OR gate:**

Ex-OR gate is actually Ex-NOR gate followed by NOT gate. So give the output of Ex-NOR gate to a NOT gate, overall output is that of an Ex-OR gate.  
Y = A’B+ AB’



#### **4) NOR gates as Ex-NOR gate:**

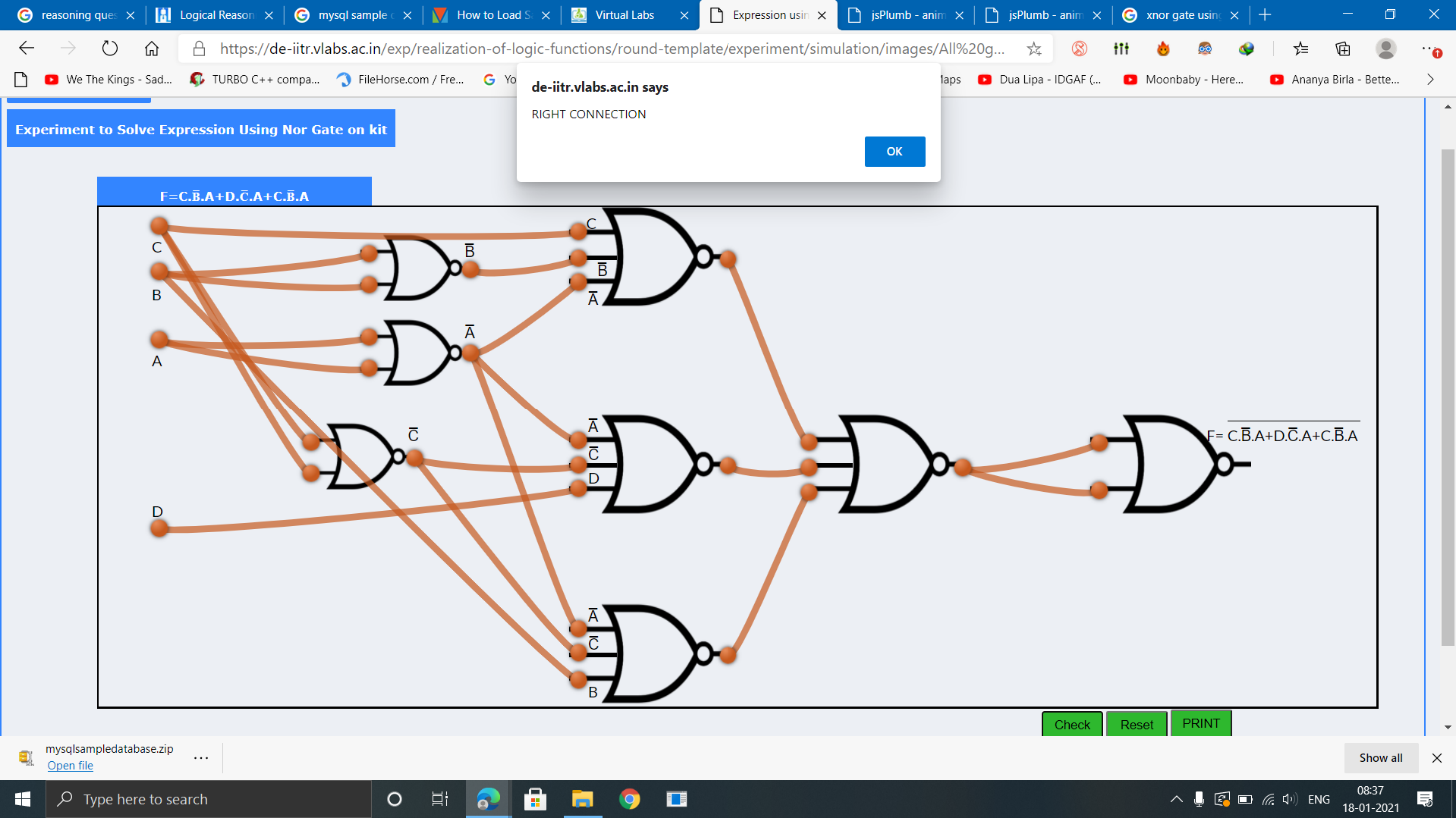
 The XNOR gate (sometimes ENOR, EXNOR or NXOR and pronounced as Exclusive NOR) is a digital logic gate whose function is the logical complement of the Exclusive OR (XOR) gate.The output of a two input Ex-NOR gate is shown by: Y = AB + A’B’.



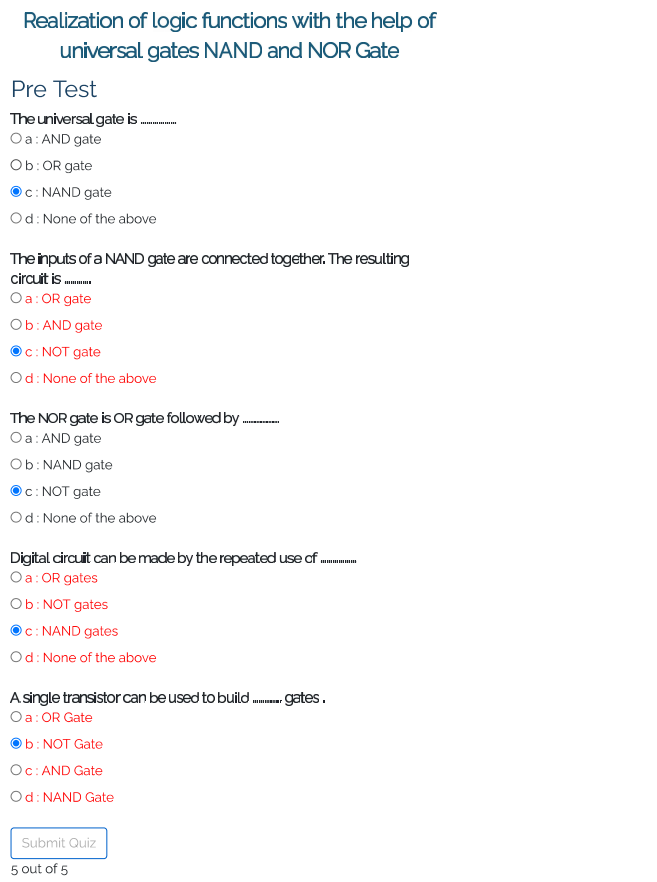
#### **5) Constructing a circuit with NOR gates only:**

Designing a circuit with NOR gates only uses the same basic techniques as designing a circuit with NAND gates; that is, the application of deMorgan’s theorem. The only difference between NOR gate design and NAND gate design is that the former must eliminate product terms and the later must eliminate sum terms.

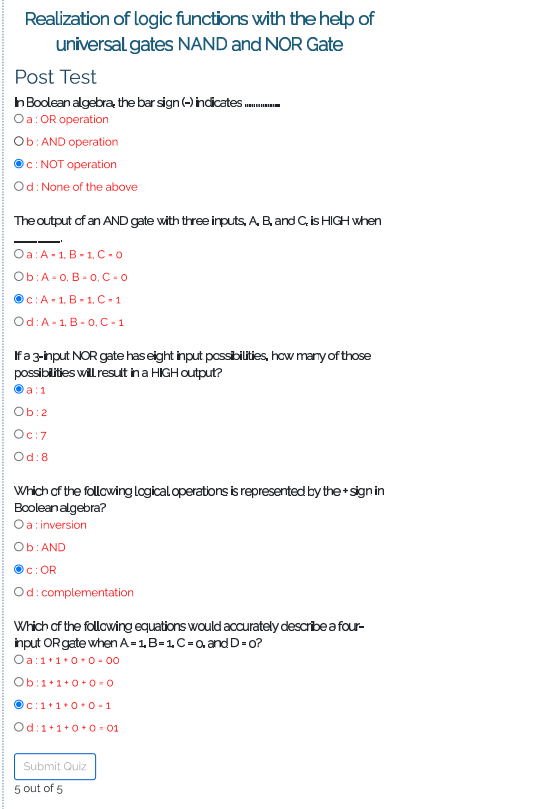
F=(((C.B'.A)+(D.C'.A)+(C.B'.A))')'

  
Implementing the simplified function with NOR gates only

**PRE-TEST:**



**POST-TEST:**



**Experiment-5**

### AIM:

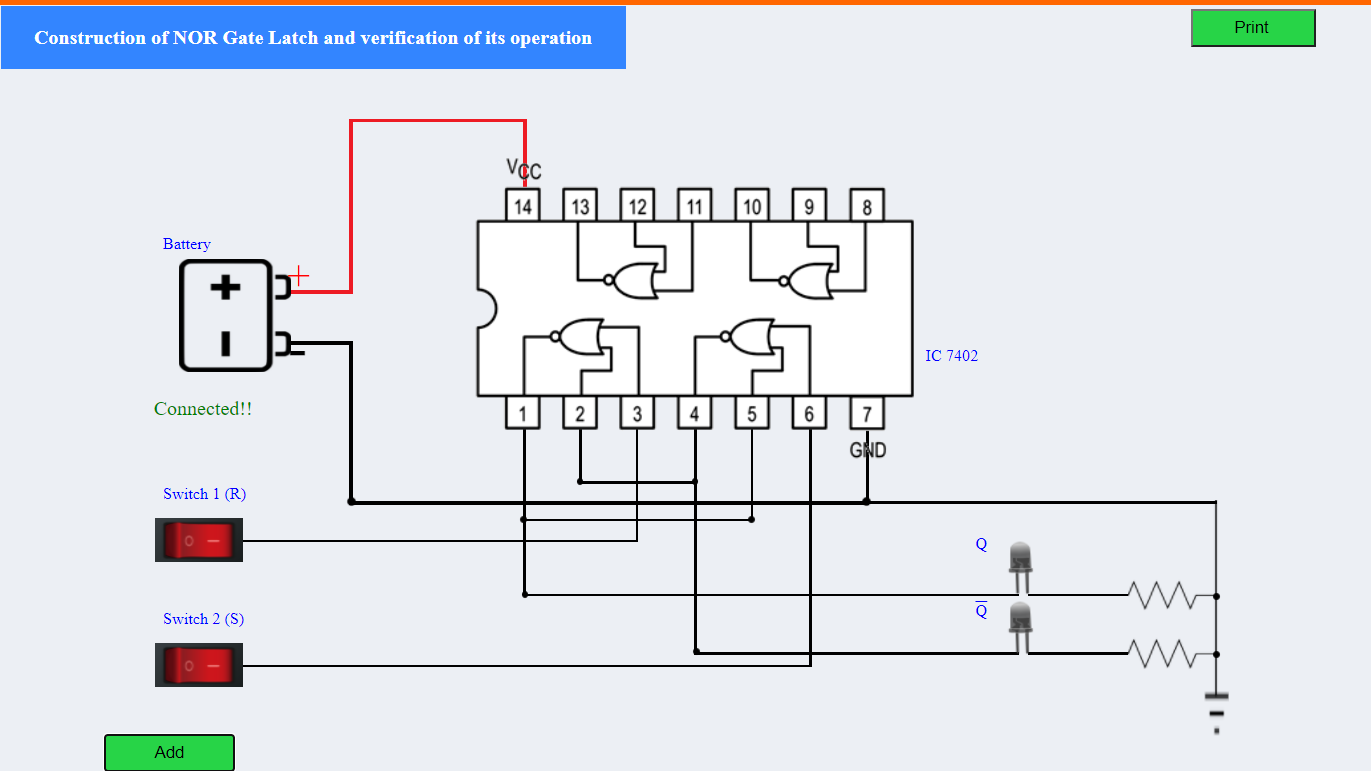
### To verify the truth table and timing diagram of NOR gate latch using NOR gate IC and analyse the circuit of NOR gate latch with the help of LEDs display.

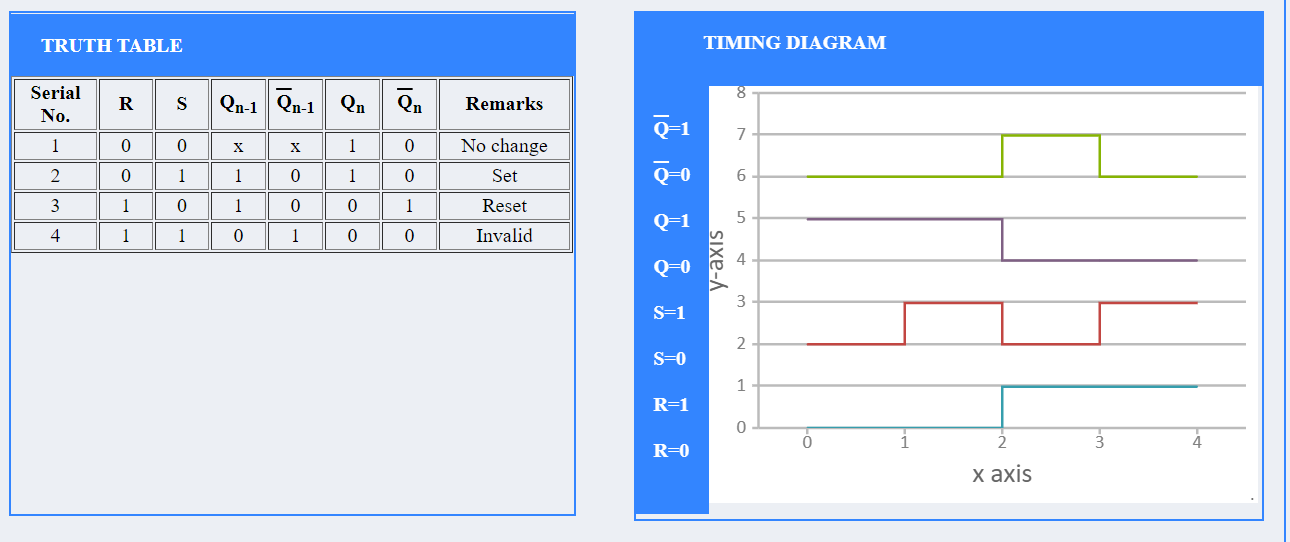
### INTRODUCTION:

Latches are basic storage elements that operate with signal levels (rather than signal transitions). Latches controlled by a clock transition are flip-flops. Latches are edge-sensitive devices. Latches are useful for the design of the asynchronous sequential circuit.

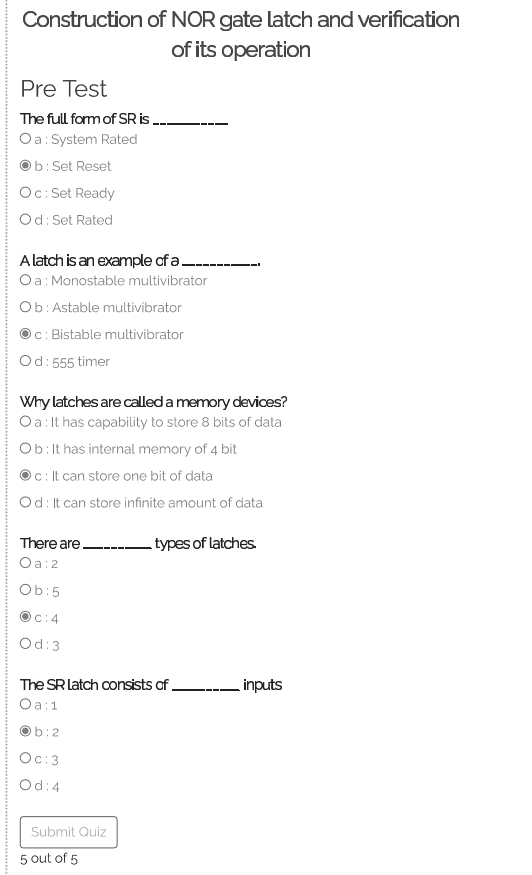
SR (Set-Reset) Latch – SR Latch is a circuit with:  
  
(i) 2 cross-coupled NOR gates or 2 cross-coupled NAND gates.  
(ii) 2 inputs S for SET and R for RESET.  
(iii) 2 outputs Q, Q.

## Construction of NOR gate latch and verification of its operation:

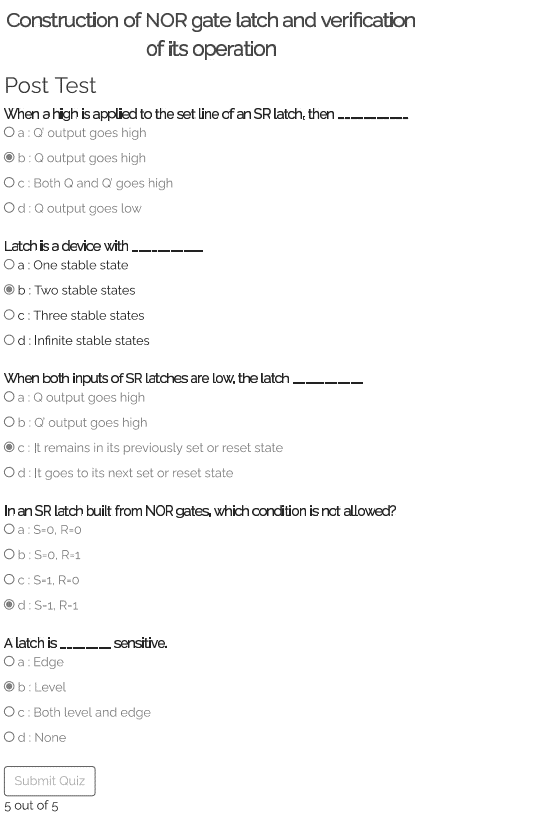




**PRE-TEST:**



**POST TEST:**



**EXPERIMENT-6**

## **AIM:**

#### To verify the truth table and timing diagram of RS, JK, T and D flip-flops by using NAND & NOR gates ICs and analyze the circuit of RS, JK, T and D flip-flops with the help of LEDs display.

**INTRODUCTION:**

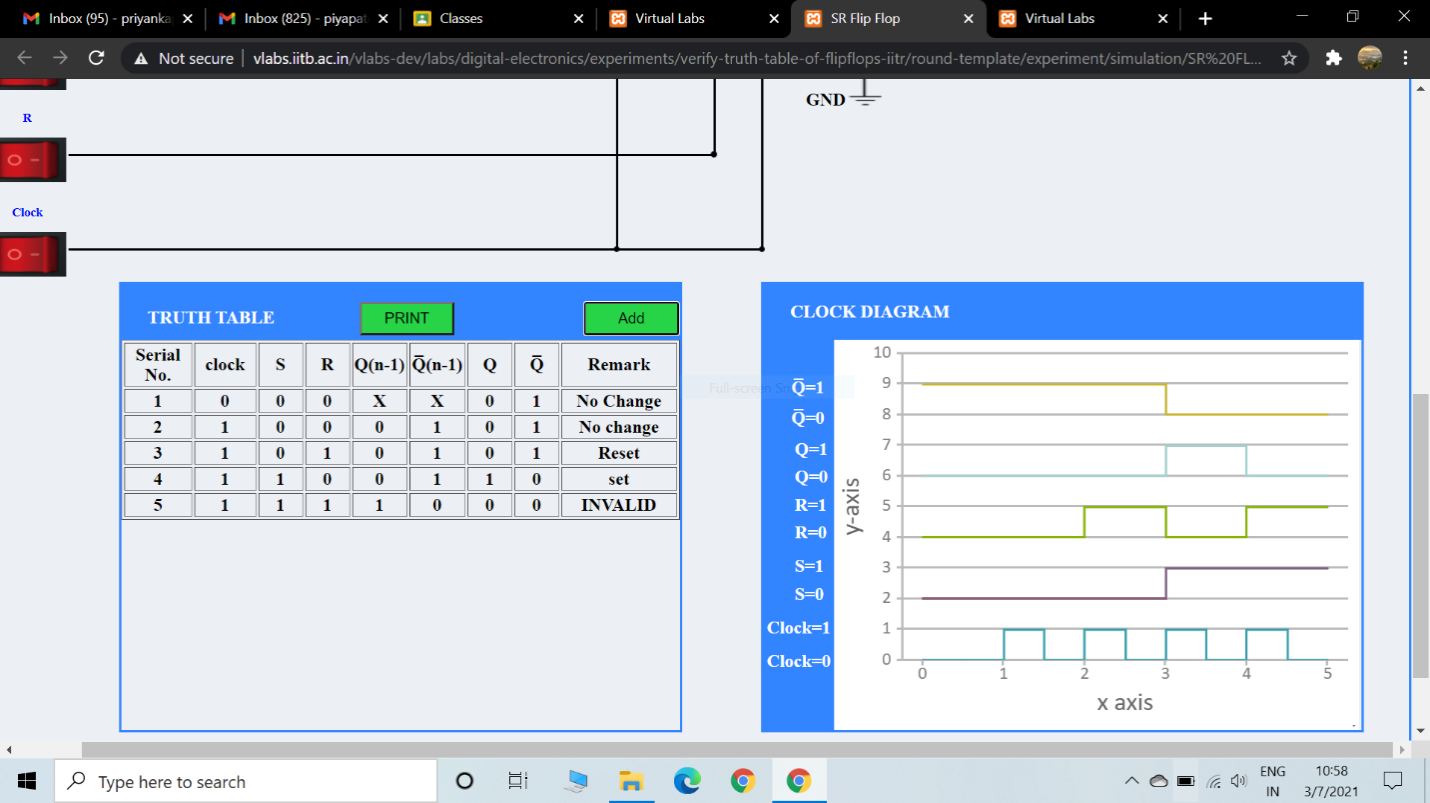
A flip flop is an electronic circuit with two stable states that can be used to store binary data. The stored data can be changed by applying varying inputs. Flip-flops and latches are fundamental building blocks of digital electronics systems used in computers, communications, and many other types of systems.

1. **R-S flip flop**
2. **D flip flop**
3. **J-K flip flop**
4. **T flip flop**

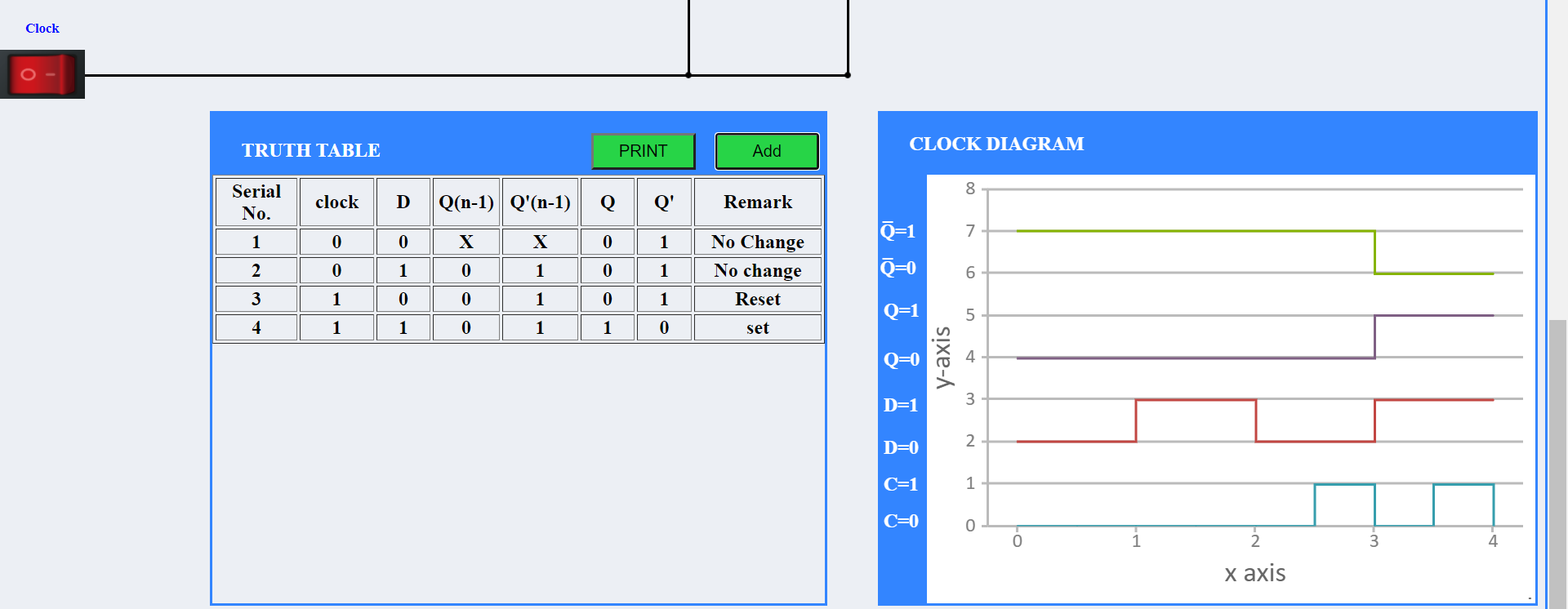
The basic NAND gate RS flip flop circuit is used to store the data and thus provides feedback from both of its outputs again back to its inputs. The RS flip flop actually has three inputs, SET, RESET and its current output Q relating to its current state as shown in figure below.

**OBSERVATIONS AND SIMULATION:**

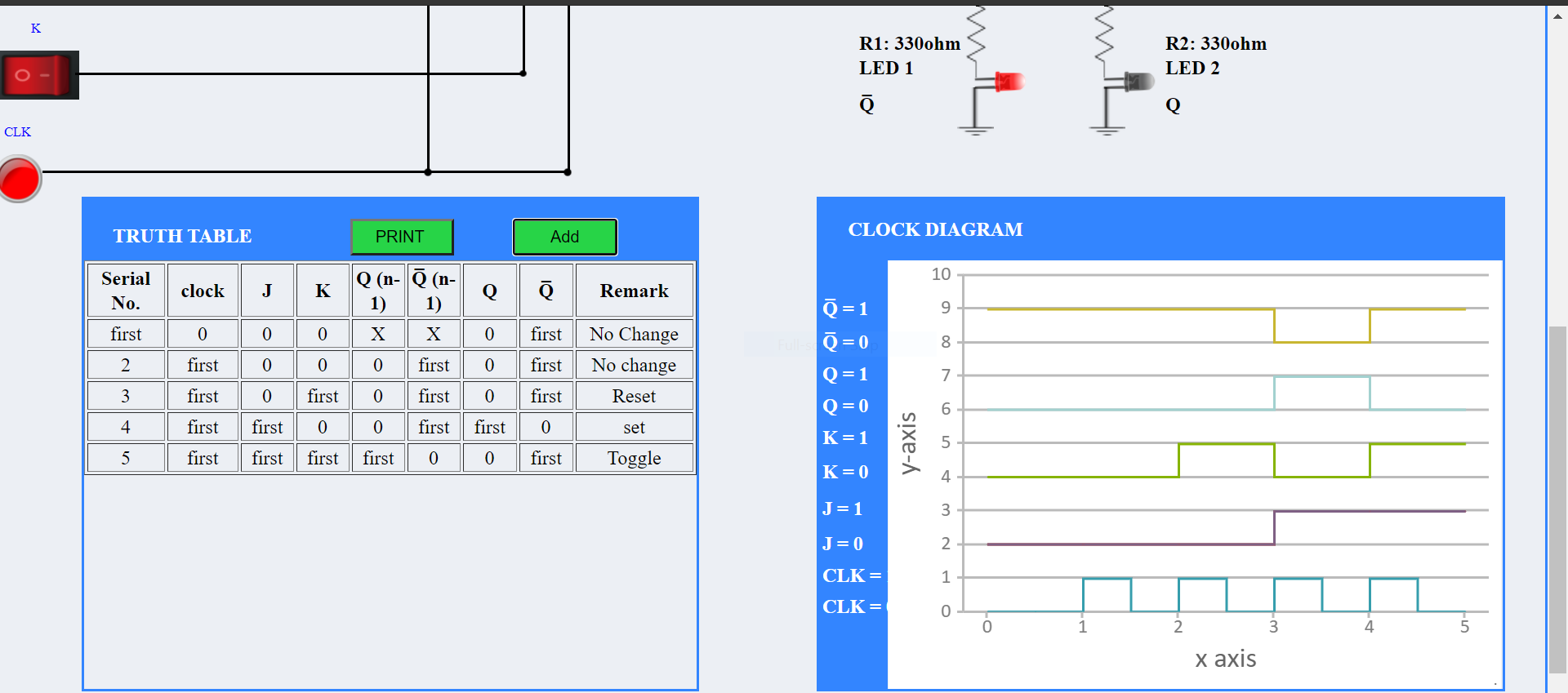
**1) R-S flip flop**

****

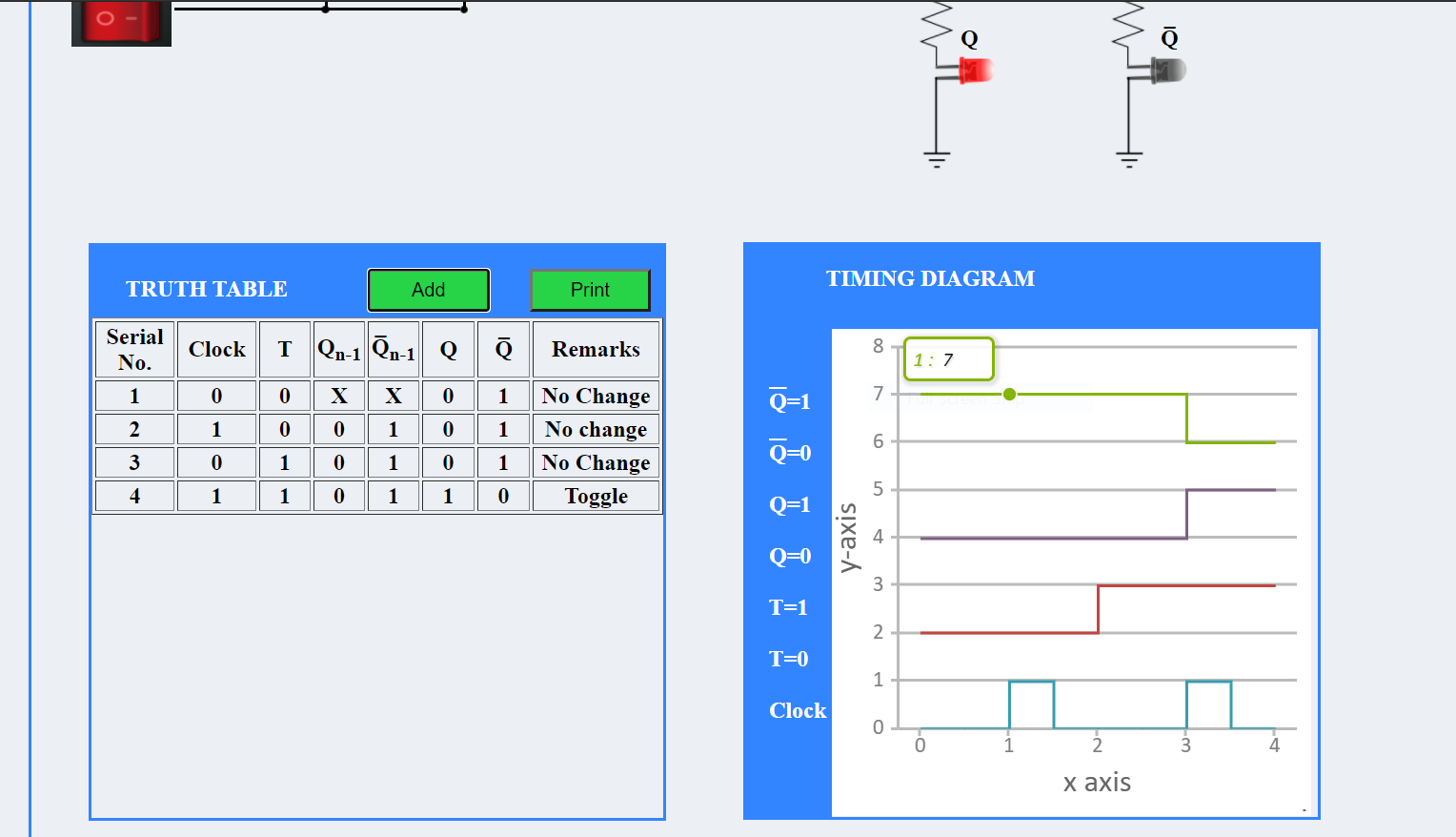
1. **D flip flop**

****

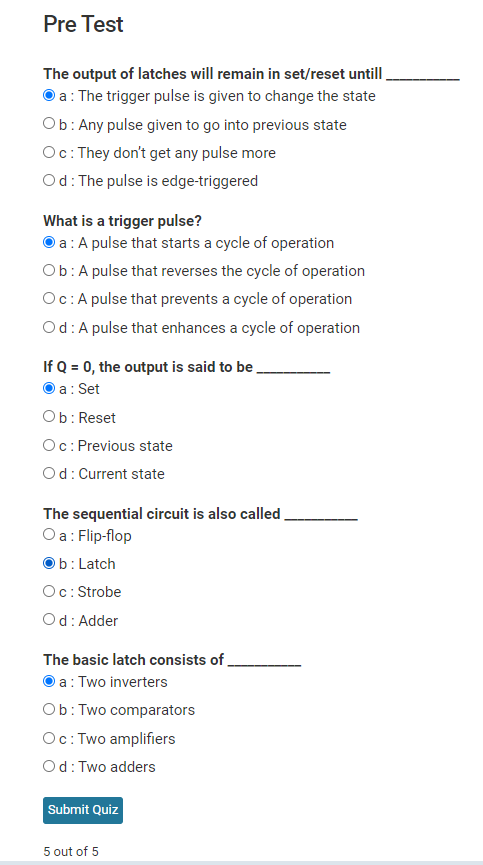
**3) J-K flip flop**

****

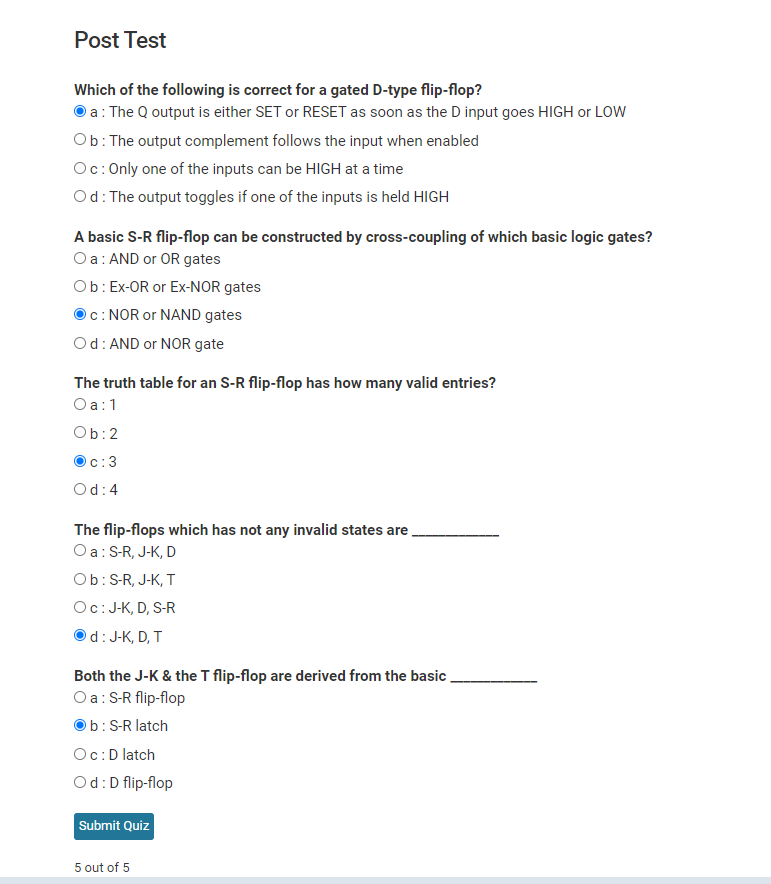
**4) T flip flop**

****

**PRE-TEST**:

****

**POST-TEST:**

****

**CONCLUSION:**

From the above experiment we get to know about latches and and get to learn how we can To verify the truth table and timing diagram of RS, JK, T and D flip-flops by using NAND & NOR gates ICs and analyse the circuit of RS, JK, T and D flip-flops. And analyzed the result with the help of a truth table and timing diagram.

**EXPERIMENT-7**

#### **AIM:**

#### To analyze the circuit and truth table of 4-bit SIPO (serial input parallel output) shift register by using IC 7474 (D flip flop).

**INTRODUCTION:**

In Serial In Parallel Out (SIPO) shift registers, the data is stored into the register serially while it is retrieved from it in parallel-fashion. Figure 1 shows an n-bit synchronous SIPO shift register sensitive to positive edge of the clock pulse. Here the data word which is to be stored (Data in) is fed serially at the input of the first flip-flop (D1 of FF1).

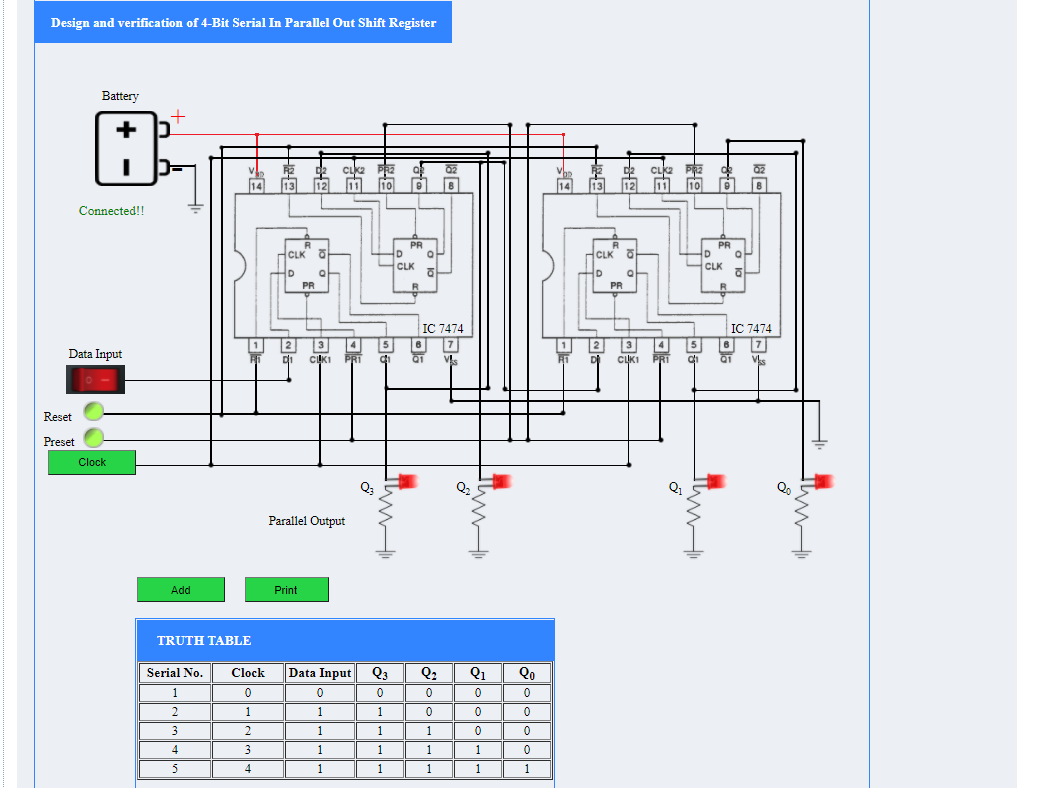
It is also seen that the inputs of all other flip-flops (except the first flip-flop FF1) are driven by the outputs of the preceding ones like the input of FF2 is driven by the output of FF1. In this kind of shift register, the data stored within the register is obtained as a parallel-output data word (Data out) at the individual output pins of the flip-flops (Q1 to Qn).

In general, the register contents are cleared by applying high on the clear pins of all the flip-flops at the initial stage. After this, the first bit, B1 of the input data word is fed at the D1 pin of FF1.This bit (B1) will enter into FF1, get stored and thereby appears at its output Q1 on the appearance of first leading edge of the clock. Further at the second clock pulse, the bit B1 right-shifts and gets stored into FF2 while appearing at its output pin Q2 while a new bit, B2 enters into FF1.

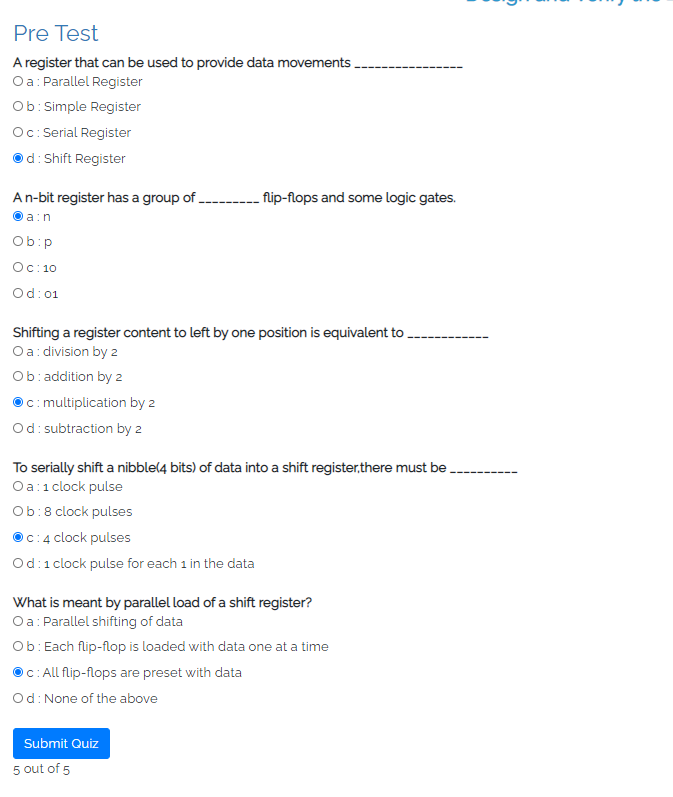
Similarly, at each clock pulse the data within the register moves towards right by a single bit while a new bit of the input word enters into the register. Meanwhile one can extract the bits stored within the register in parallel-fashion at the individual flip-flop outputs. Analyzing on the same grounds, one can note that the n-bit input data word is obtained as an n-bit output data word from the shift register at the rising edge of the nth clock pulse.

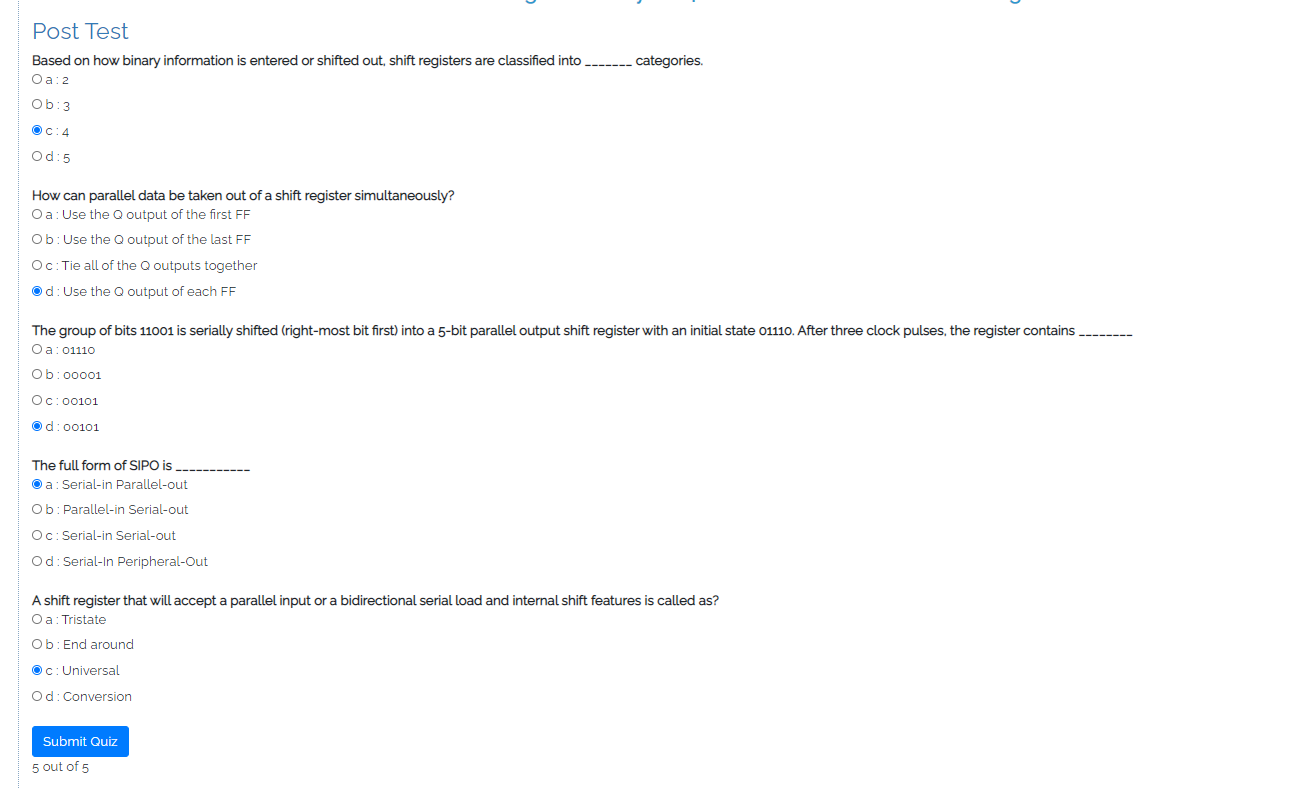
In the right-shift SIPO shift-register, data bits shift from left to right for each clock pulse. However, if the data bits are made to shift from right to left in the same design, one gets a left-shift SIPO shift-register as shown by figure 3. Nevertheless, the basic working principle remains the same except the fact that now Bn down to B1 is stored in Qn down to Q1 i.e., Q1 = B1, Q2 = B2 … Qn = Bn at the nth clock pulse.

**SIMULATION:**



**PRE-TEST**



**POST-TEST**

**EXPERIMENT-10**

#### **AIM:**

To verify the truth table and timing diagram of 4-bit synchronous parallel counter and 4-bit asynchronous parallel counter by using JK flip flop ICs and analyse the circuit of 4-bit synchronous parallel counter and 4-bit asynchronous parallel counter with the help of LEDs display.

**INTRODUCTION:**

A counter is a device which stores (and sometimes displays) the number of times a particular event or process has occurred, often in relationship to a clock signal. Counters are used in digital electronics for counting purpose, they can count specific event happening in the circuit. For example, in UP counter a counter increases count for every rising edge of clock.

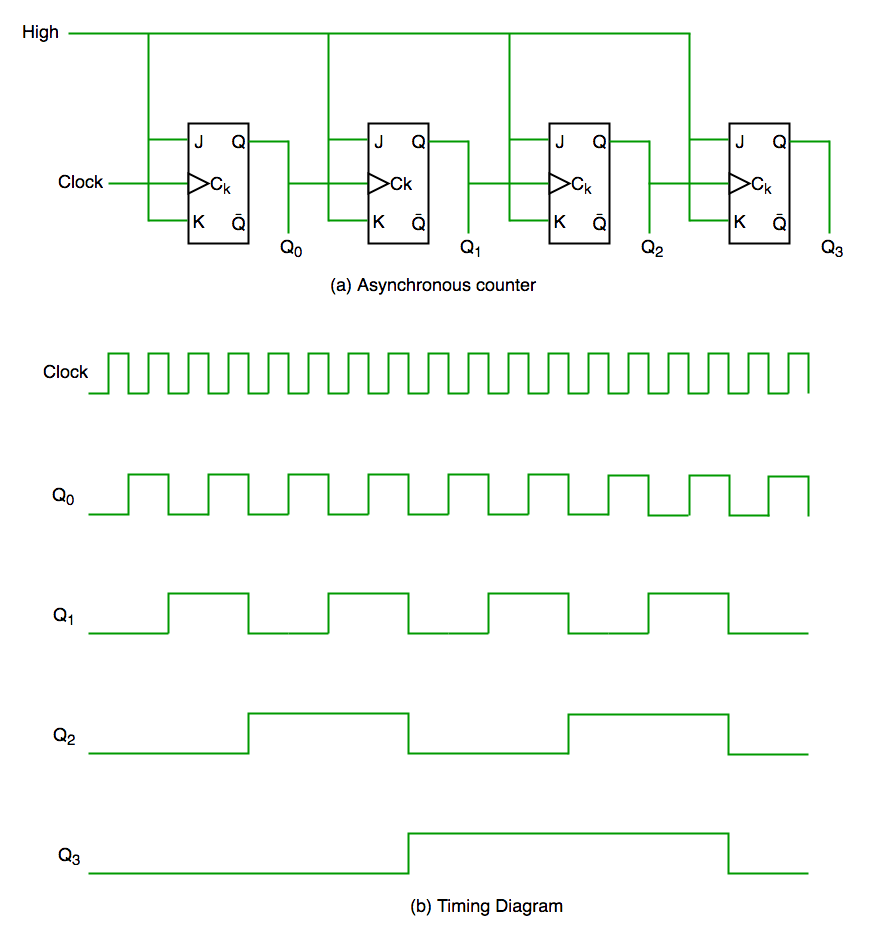
### **Classification of Counters**

Counters are broadly divided into two categories

1. Asynchronous counter
2. Synchronous counter

### **1) Asynchronous Counter**

In asynchronous counter we don’t use universal clock, only first flip flop is driven by main clock and the clock input of rest of the following counters is driven by output of previous flip flops. We can understand it by following diagram-

  
  
Figure-1: Asynchronous Counter Circuit and Timing Diagram

It is evident from timing diagram that Q0 is changing as soon as the rising edge of clock pulse is encountered, Q1 is changing when rising edge of Q0 is encountered(because Q0 is like clock pulse for second flip flop) and so on. In this way ripples are generated through Q0,Q1,Q2,Q3 hence it is also called RIPPLE counter.

### **2) Synchronous Counter**

Unlike the asynchronous counter, synchronous counter has one global clock which drives each flip flop so output changes in parallel. The one advantage of synchronous counter over asynchronous counter is, it can operate on higher frequency than asynchronous counter as it does not have cumulative delay because of same clock is given to each flip flop.

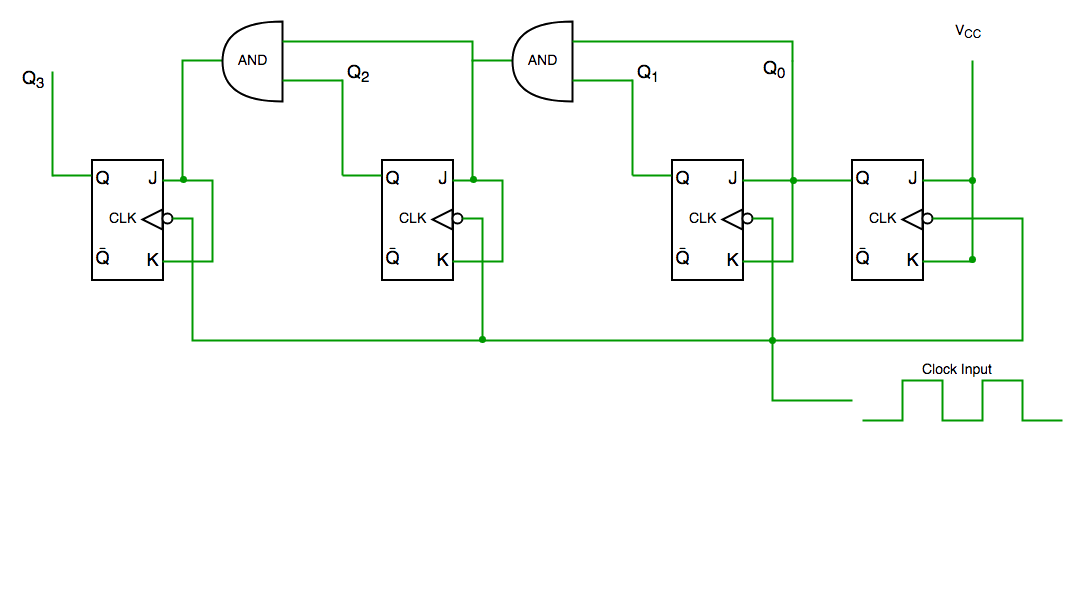
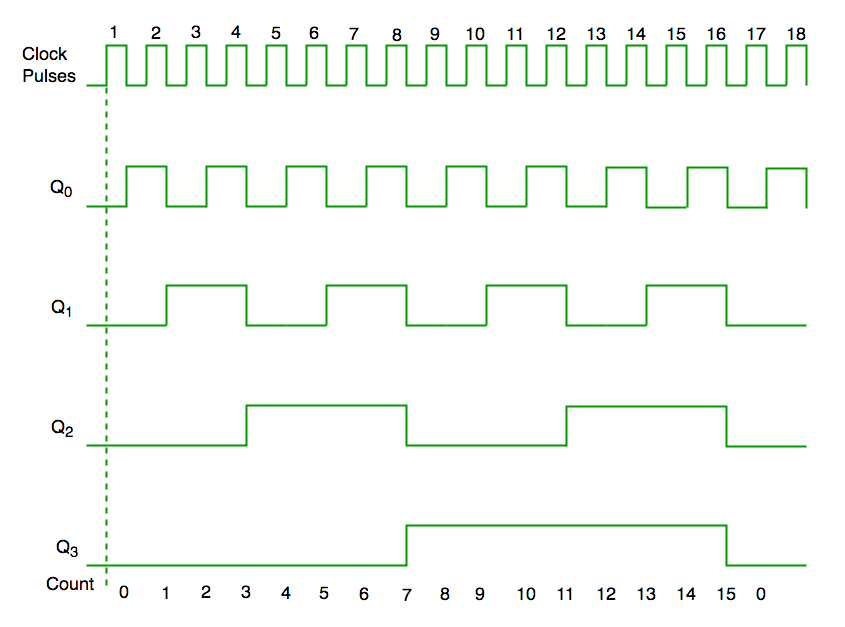
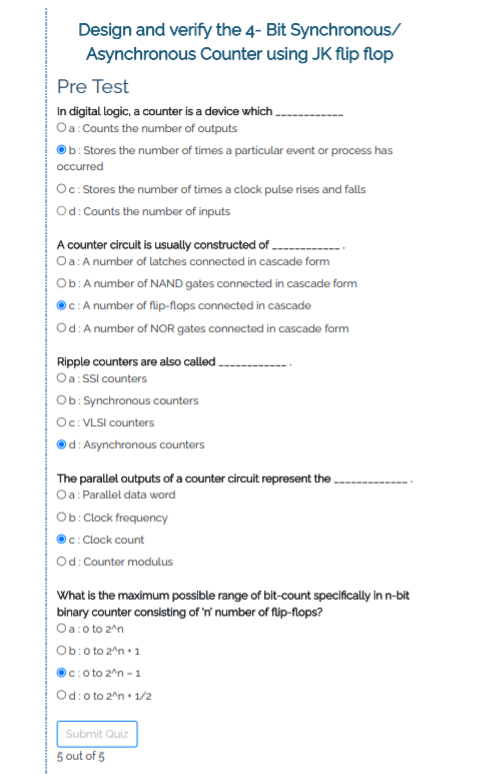
  
  


Figure-2: Synchronous Counter Circuit and Timing Diagram

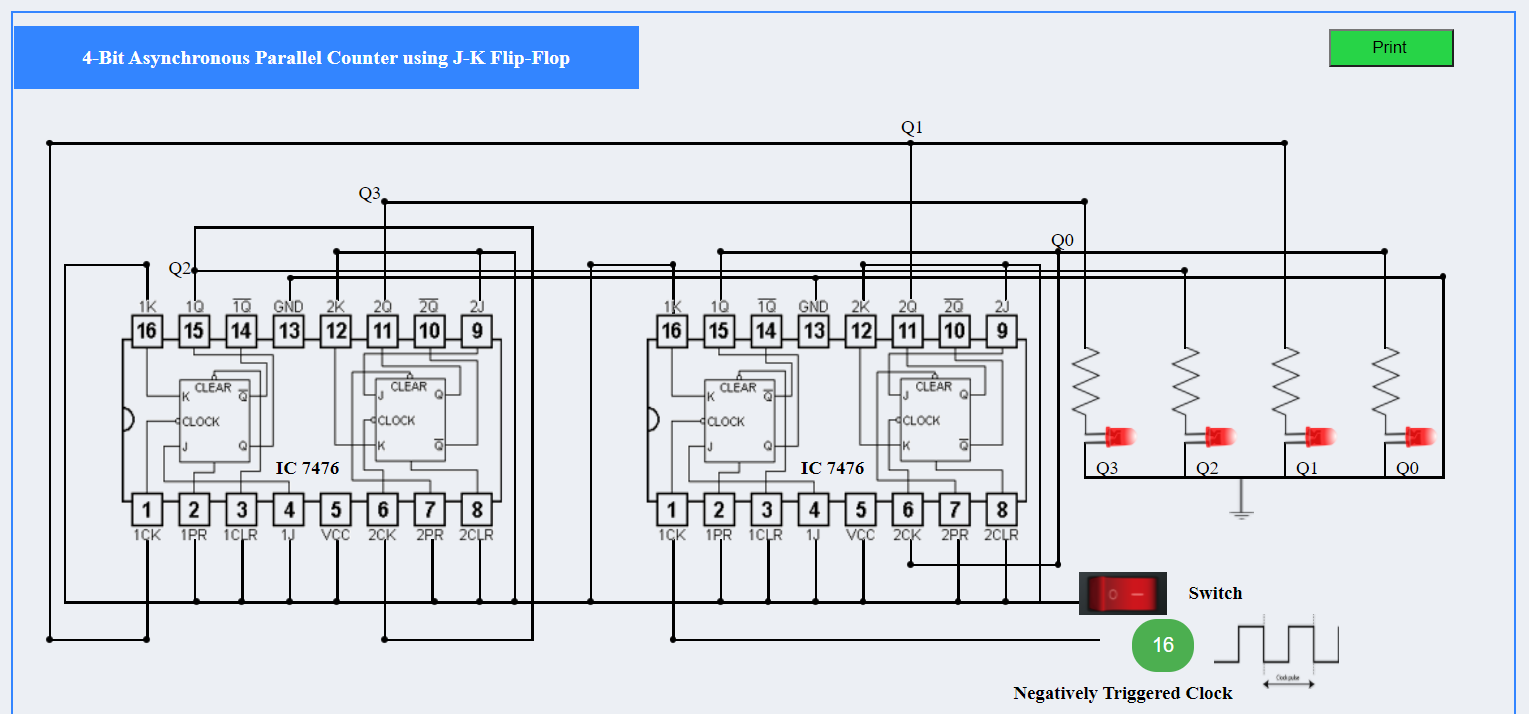
**PRETEST**

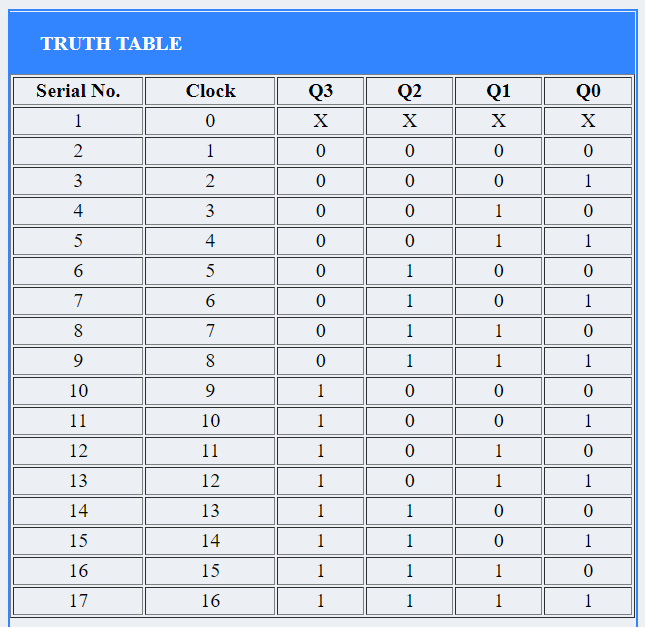


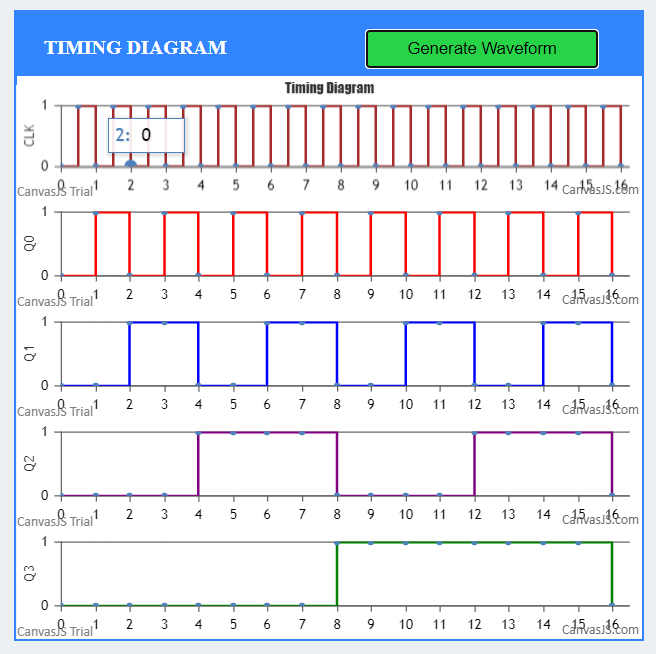
**POST TEST**

**SIMULATION**

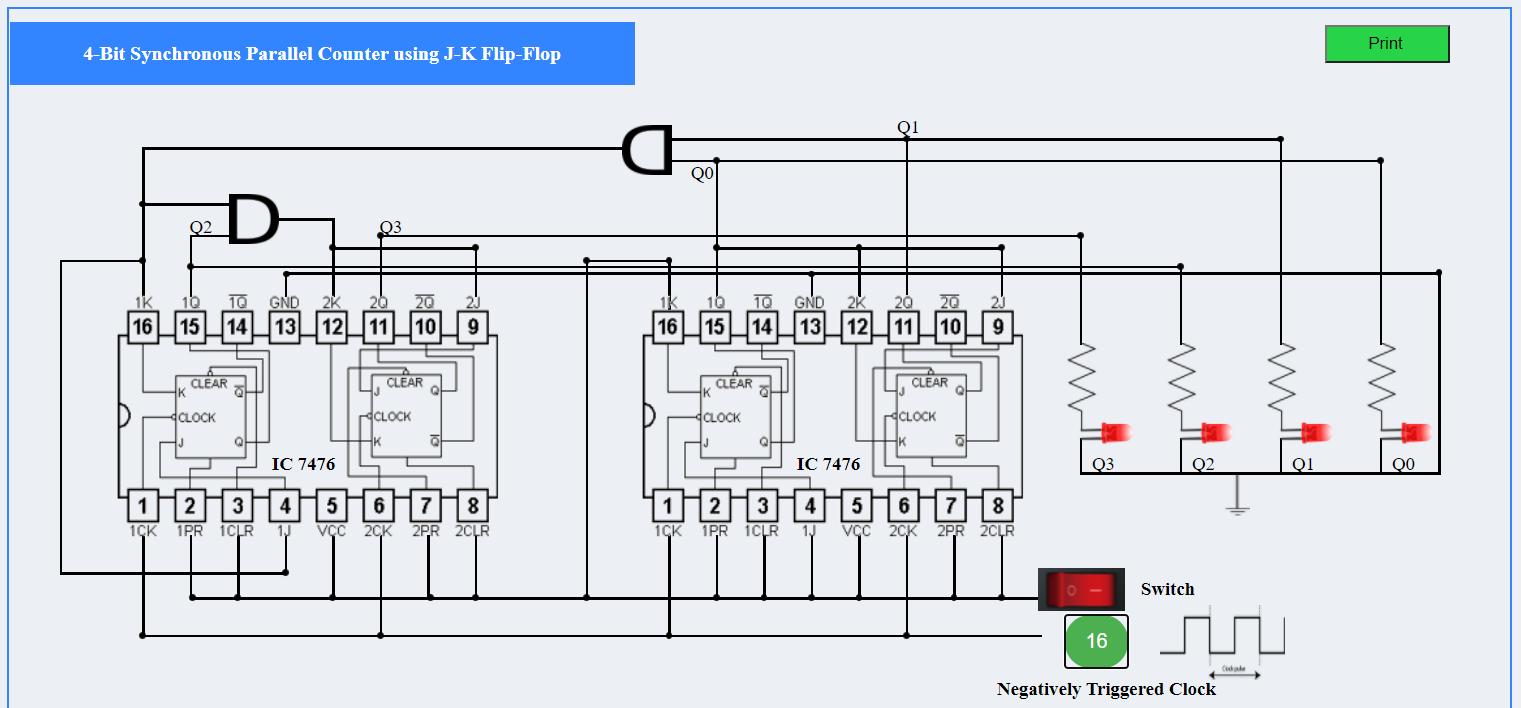
* **ASYNCHRONOUS**

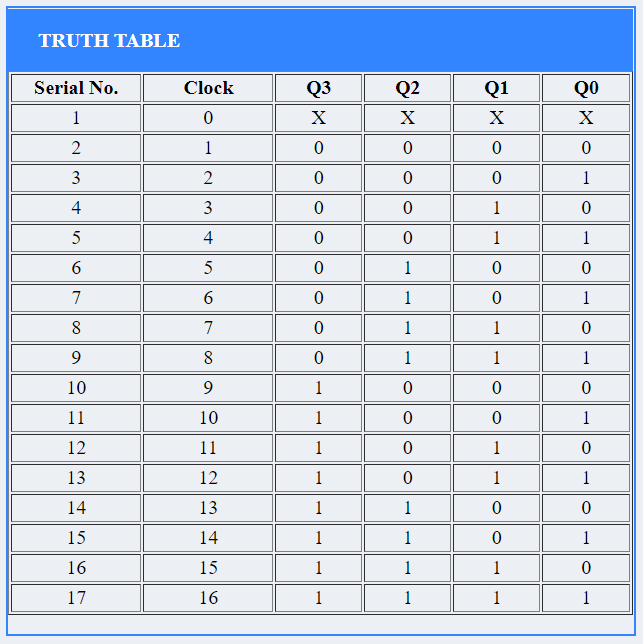
****

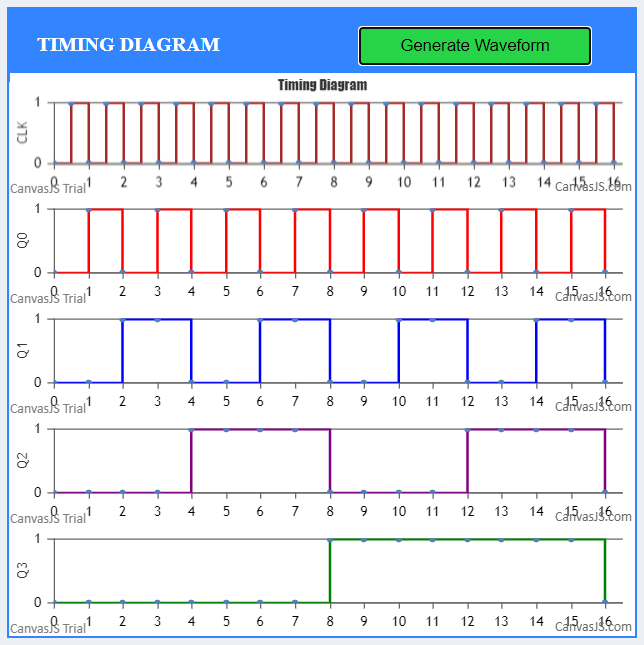
****

****

* **SYNCHRONOUS**

****

****

****