**EXPERIMENT NO:- 1.**

OBJECT:- Implementation of AND gate.

SOFTWARE REQUIRED**:-** Logisim software.

THEORY:-

GATE:- Binary information is represented in digital computers by Physical quantities called signals.

The manipulation of binary Information is done by logic circuits called gates.

Gates are blocks of hardware that produce signals of binary 1 or 0 when input logic requirements are satisfied. A variety of logic gates are commonly used in digital Computer systems. The input output relationship of the binary variables for each gate can be represented in tabular form by a truth table.

The **AND gate** is a basic digital [logic gate](https://en.wikipedia.org/wiki/Logic_gate) that implements [logical conjunction](https://en.wikipedia.org/wiki/Logical_conjunction) - it behaves according to the [truth table](https://en.wikipedia.org/wiki/Truth_table) to the right. A HIGH output (1) results only if both the inputs to the AND gate are HIGH (1). If neither or only one input to the AND gate is HIGH, a LOW output results. In another sense, the function of AND effectively finds the *minimum* between two binary digits, just as the [OR](https://en.wikipedia.org/wiki/OR_gate) function finds the *maximum* between two binary digits. Therefore, the output is always 0, except when all the inputs are 1.

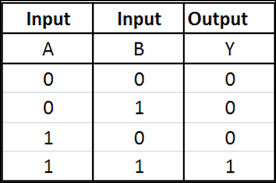


Figure 1- Truth table

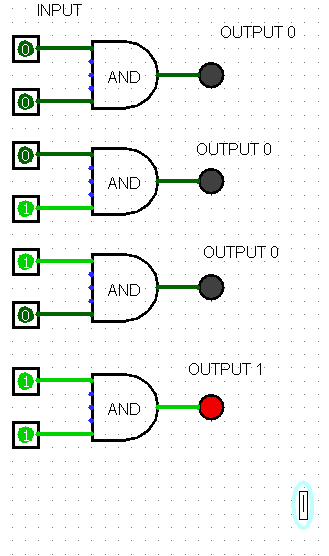


Figure 2- NAND gate

**EXPERIMENT NO:- 2.**

OBJECT:- Implementation of OR Gate.

SOFTWARE REQUIRED:- Logisim software

THEORY:-

GATE:- Binary information is represented in digital computers by Physical quantities called signals.

The manipulation of binary Information is done by logic circuits called gates.

Gates are blocks of hardware that produce signals of binary 1 or 0 when input logic requirements are satisfied. A variety of logic gates are commonly used in digital Computer systems. The input output relationship of the binary variables for each gate can be represented in tabular form by a truth table.

The **OR gate** is a basic digital [logic gate](https://en.wikipedia.org/wiki/Logic_gate) that implements [logical conjunction](https://en.wikipedia.org/wiki/Logical_conjunction) - it behaves according to the [truth table](https://en.wikipedia.org/wiki/Truth_table) to the right. A HIGH output (1) results only if both the inputs to the AND gate are HIGH (1). If neither or only one input to the OR gate is HIGH, a LOW output results. In another sense, the function of OR effectively finds the *minimum* between two binary digits, just as the [OR](https://en.wikipedia.org/wiki/OR_gate) function finds the *maximum* between two binary digits. Therefore, the output is always 0, except when all the inputs are 1.

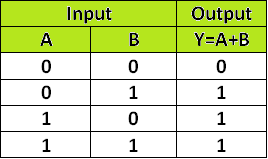
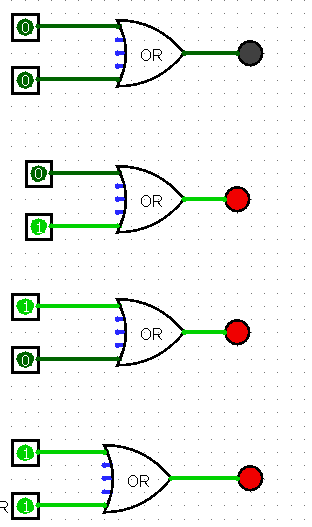


Figure 1- truth table



**Figure2- OR gate**

**EXPERIMENT NO:-3.**

OBJECT:-Implementation of NAND Gate.

SOFTWARE REQUIRED:-Logisim software.

THEORY:-

GATE:-

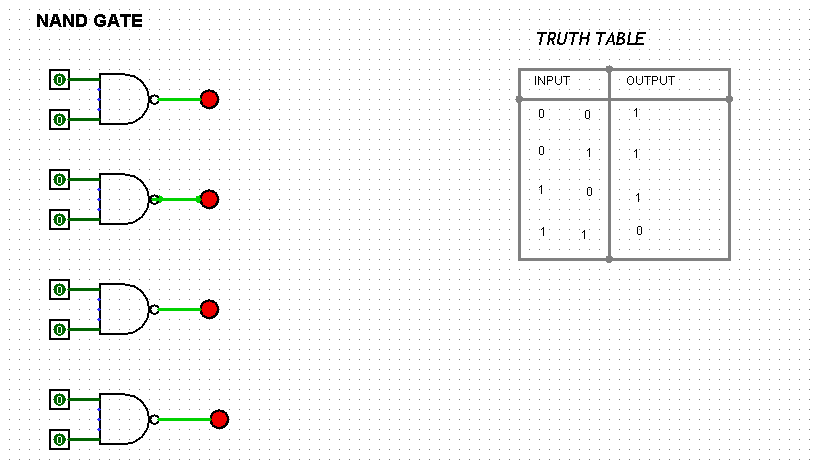
Binary information is represented in digital computers by Physical quantities called signals.

The manipulation of binary Information is done by logic circuits called gates.

Gates are blocks of hardware that produce signals of binary 1 or 0 when input logic requirements are satisfied. A variety of logic gates are commonly used in digital Computer systems.

The input output relationship of the binary variables for each gate can be represented in tabular form by a truth table.

The basic logic gates are AND ann [electronics](https://en.wikipedia.org/wiki/Electronics), a **logic gate** is an idealized or physical device implementing a [Boolean function](https://en.wikipedia.org/wiki/Boolean_function); that is, it performs a [logical operation](https://en.wikipedia.org/wiki/Logical_operation) on one or more [binary](https://en.wikipedia.org/wiki/Binary_number) inputs, and produces a single binary output. Depending on the context, the term may refer to an **ideal logic gate**, one that has for instance zero [rise time](https://en.wikipedia.org/wiki/Rise_time) and unlimited [fan-out](https://en.wikipedia.org/wiki/Fan-out), or it may refer to a non-ideal physical device(see [Ideal and real op-amps](https://en.wikipedia.org/wiki/Ideal_and_real_op-amps) for d inclusive OR with multiple inputs and NOT with a single input)

****

**Figure 1- NAND gate Figure 2- truth table**

**EXPERIMENT NO:-4.**

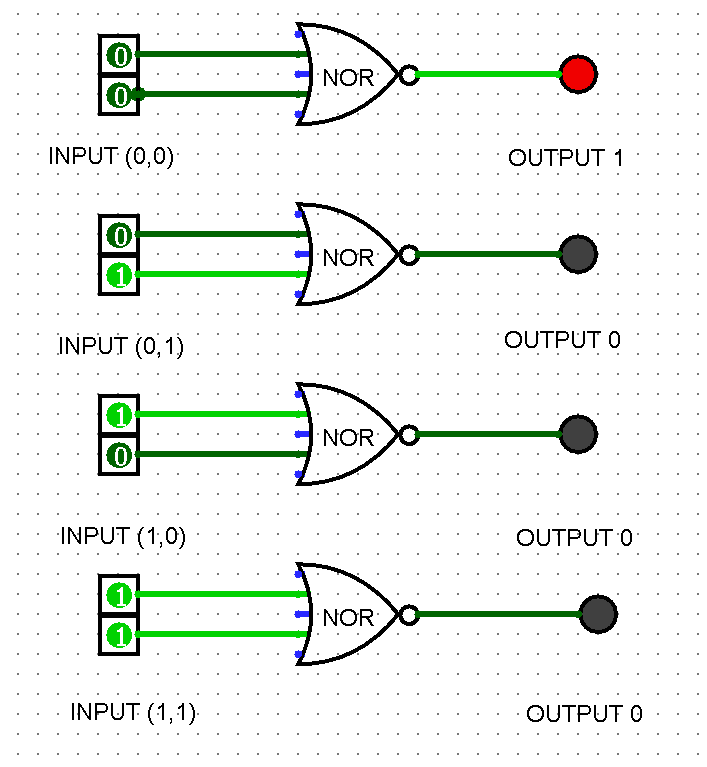
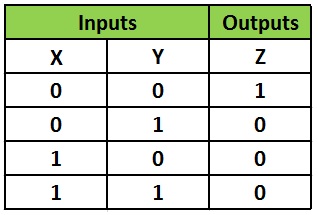
OBJECT: **-** Implementation of NOR GATE.

SOFTWARE REQUIRED: logisim software.

THEORY:

Multiplexer The NOR gate is a digital logic gate that implements logical NOR - it behaves according to the truth table to the right. A HIGH output (1) results if both the inputs to the gate are LOW (0); if one or both input is HIGH (1), a LOW output (0) results. NOR is the result of the negation of the OR operator. It can also be seen as an AND gate with all the inputs inverted. NOR is a functionally complete operation—NOR gates can be combined to generate any other logical function. it shares this property with the NAND gate. By contrast, the OR operator is monotonic as it can only change LOW to HIGH but not vice versa.

In most, but not all, circuit implementations, the negation comes for free—including CMOS and TTL. In such logic families, OR is the more complicated operation; it may use a NOR followed by a NOT. A significant exception is some forms of the domino logic family.

 ****

**Fig: NOR GATE Fig: NOR Truth Table**

**EXPERIMENT NO:-5.**

OBJECT:- Implementation of NOT Gate.

SOFTWARE REQUIRED:- Logisim software

THEORY:-

GATE:-

Binary information is represented in digital computers by Physical quantities called signals.

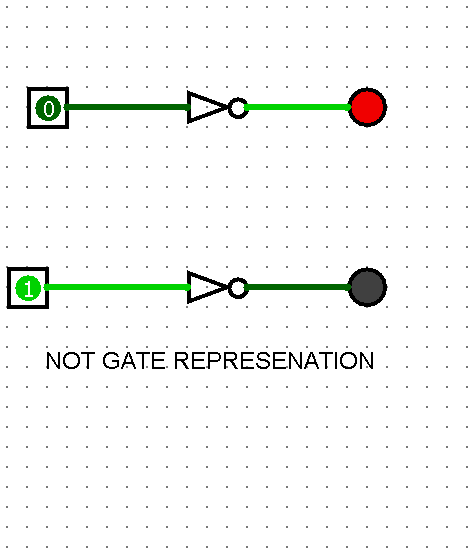
The manipulation of binary Information is done by logic circuits called gates.

Gates are blocks of hardware that produce signals of binary 1 or 0 when input logic requirements are satisfied. A variety of logic gates are commonly used in digital Computer systems. The input output relationship of the binary variables for each gate can be represented in tabular form by a truth table.

A **NOT gate** (also often called **Inverter**) is a [logic gate](https://simple.wikipedia.org/wiki/Logic_gate). Each NOT gate has only one input signal. Logically with NOT gates, the input and the output swap, so if you input *1* it outputs as *0*; likewise if you input *0* it outputs as *1*.

Generally, actual NOT gates with a [voltage](https://simple.wikipedia.org/wiki/Voltage) below 0.5V is *0*, and a voltage of 4–5V is *1*.

The inverter can be made of a discrete [transistor](https://simple.wikipedia.org/wiki/Transistor) with other components, or several inverters may be packaged in an [integrated circuit](https://simple.wikipedia.org/wiki/Integrated_circuit).



**Figure2- NAND gate**

EXPERIMENT NO:-6.

OBJECT: - Implementation of XOR Gate.

SOFTWARE REQUIRED: - Logistic software

THEORY:-

GATE:-

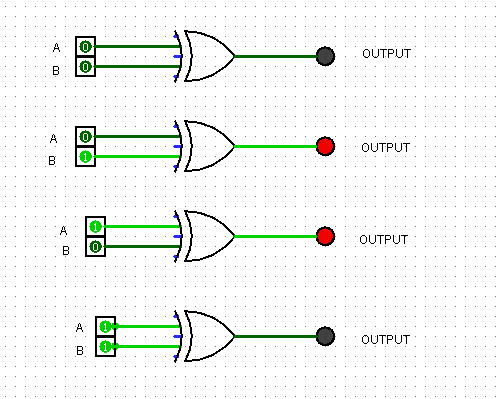
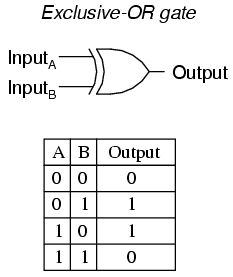
Binary information is represented in digital computers by Physical quantities called signals.

The manipulation of binary Information is done by logic circuits called gates.

Gates are blocks of hardware that produce signals of binary 1 or 0 when input logic requirements are satisfied. A variety of logic gates are commonly used in digital Computer systems. The input output relationship of the binary variables for each gate can be represented in tabular form by a truth table.

The **XOR gate** (sometimes EOR **gate**, or EXOR **gate** and pronounced as **Exclusive OR gate**) is a digital logic **gate** that gives a true (1/HIGH) output when the number of true inputs is odd. An **XOR gate** implements an **exclusive or**; that is, a true output results if one, and only one, of the inputs to the **gate** is true.

If both inputs are false (0/LOW) and both are true, a false output results. XOR represents the inequality function, i.e., the output is true if the inputs are not alike otherwise the output is false. A way to remember XOR is "one or the other but not both".

**Figure1- XOR gate figure 2- XOR truth table**

**EXPERIMENT NO:-7.**

OBJECT: - Implementation of -XNOR Gate.

SOFTWARE REQUIRED: - Logistic software

THEORY:-

GATE:-

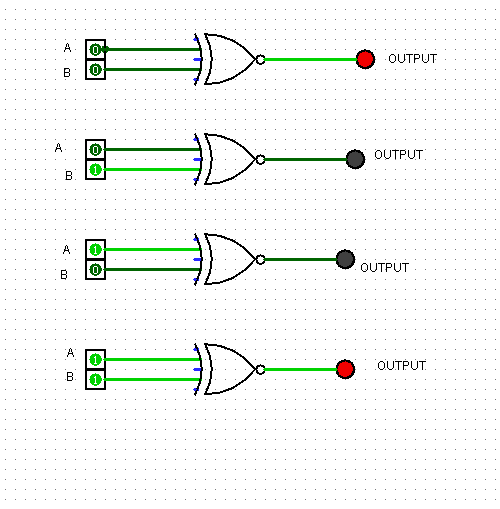
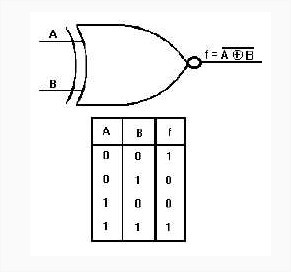
Binary information is represented in digital computers by Physical quantities called signals.

The manipulation of binary Information is done by logic circuits called gates.

Gates are blocks of hardware that produce signals of binary 1 or 0 when input logic requirements are satisfied. A variety of logic gates are commonly used in digital Computer systems. The input output relationship of the binary variables for each gate can be represented in tabular form by a truth table.

The **XNOR gate** (sometimes, EXNOR, ENOR, and, rarely, NXOR, [XAND](https://en.wikipedia.org/wiki/XAND)) is a digital [logic gate](https://en.wikipedia.org/wiki/Logic_gate) whose function is the logical complement of the exclusive OR ([XOR](https://en.wikipedia.org/wiki/XOR_gate)) gate. The two-input version implements [logical equality](https://en.wikipedia.org/wiki/Logical_equality), behaving according to the truth table to the right.

A high output (1) results if both of the inputs to the gate are the same. If one but not both inputs are high (1), a low output (0) results. The inverter can be made of a discrete [transistor](https://simple.wikipedia.org/wiki/Transistor) with other components, or several inverters may be packaged in an [integrated circuit](https://simple.wikipedia.org/wiki/Integrated_circuit).



**Figure 1- truth table Figure2- -XNOR gate**

Experiment No:-8.

Object:- Implementation of JK flip flop and verify their characteristics table.

SOFTWARE REQUIRD: - logistic software.

Theory:-

The J-K flip-flop is the most versatile of the basic [flip-flops](http://hyperphysics.phy-astr.gsu.edu/hbase/Electronic/flipflop.html#c1). It has the input- following character of the clocked [D flip-flop](http://hyperphysics.phy-astr.gsu.edu/hbase/Electronic/dflipflop.html#c3) but has two inputs, traditionally labeled J and K. If J and K are different then the output Q takes the value of J at the next clock edge.When the clock rises from 0 to 1, the value remembered by the flip-flop toggles if the *J* and *K* inputs are both 1, remains the same if they are both 0, and changes to the *K* input value if *J* and *K* are not equal. (The names *J* and *K* do not stand for anything.)

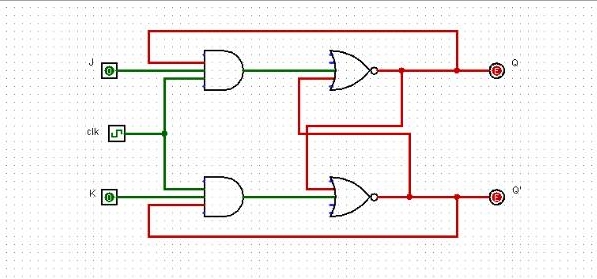
****

Figure: JK-Flip Flop

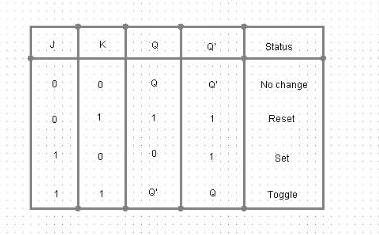
****

Figure: Truth Table

EXPERIMENT NO:- 9.

OBJECTIVE: - Implementation of S-R flip flop and verify their characteristics table.

SOFTWARE REQUIRED: - Logism software.

THEORY:-

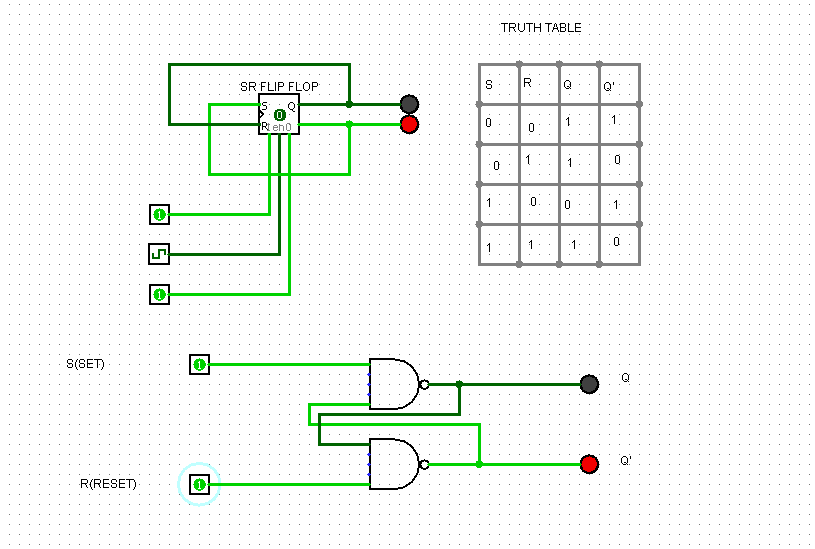
Flip flop and latches are used as data storage elements. A flip flop stores a single bit of data, one of its two states represent a one and the other represents the zero.

Flip flop can be either simple or clocked although the term flip flop has historical referred generically to both simple and clocked circuits, in modern usage it is common to reserve the term flip flop exclusively for discussing clocked circuit, the simple once are commonly called latches.

**S-R FLIP FLOP:-**

The S-R flip flop , also known as a S –R latch, Can be considered as one of the most basic sequential logic circuit possible. This simple flip flop is basically a one bit memory bistable space device that has two inputs, one which will SET the device(meaning the output equals to 1),and is labelled as another which will RESET the device labelled R.

Than the S-R description stands for “SET and RESET”. The reset input reset the flip flop back to its original state with an input Q that will be either at a logic level 1 or logic 0.

****

**Figure: SR Flip flop**

**EXPERIMENT NO:- 10.**

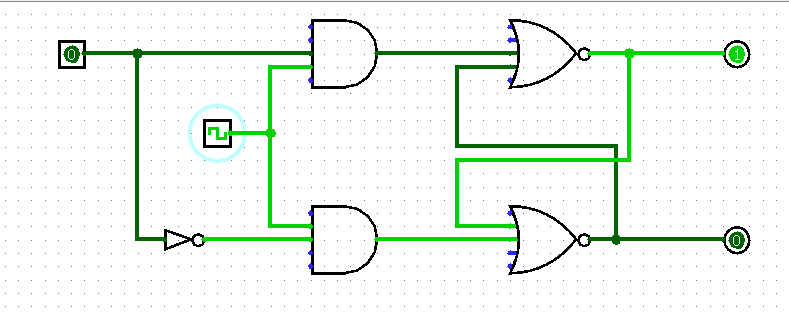
OBJECT:-Implementation of D-flip-flop and verify their characteristics table.

THEORY:-

**D-FlipFlop:**

The storage elements employed in clocked sequential circuits are called flipflops. In order to deal with the indeterminant state of a flipflop we proposed a new flipflop called D or data flipflo. Put the data and, and enable the clock it will get store and remove the clock is stays.

It ismost frequently used flipflop, as the name suggest data flipflop, because every data, doesn’t matter how much, will always stores in a memory as bits and D-flipflop is used to store bits.



**Figure :D flip flop circuit diagram**

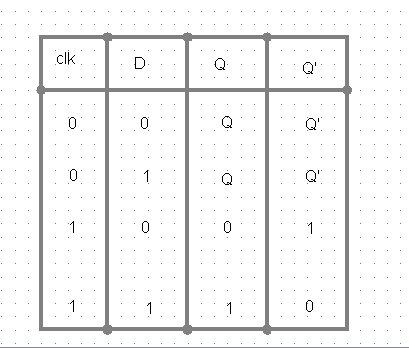
****

Figure: D flip flop truth table

**EXPRIMENT:- 11.**

OBJECT:- Implementation of T flip-flop and verify their characteristics table.

SOFTWARE REQUIRD:- logisim software.

THEORY:-

**T Flip-flop:-** The T flip-flop has one input in addition to the clock. *T* stands for toggle for the obvious reason. When *T* is asserted (*T* = 1), the flip-flop state toggles back and forth, and when *T* is de-asserted, the flip-flop keeps its current state. The T flip-flop can be constructed using a D flip-flop with the two outputs *Q* and *Q'* feedback to the *D* input through a multiplexer that is controlled by the *T* input.

If the T input is high, the T flip-flop changes state ("toggles") whenever the clock input is strobed. If the T input is low, the flip-flop holds the previous value. This behavior is described by the characteristic.

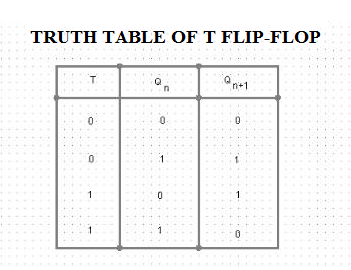
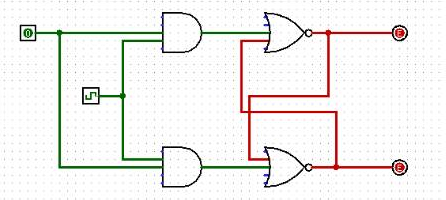
**­­ **

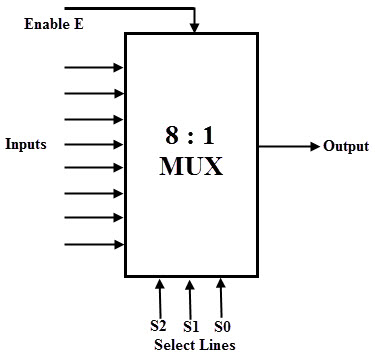
Fig:- T FLIP-FLOP

**EXPERIMENT NO. 12.**

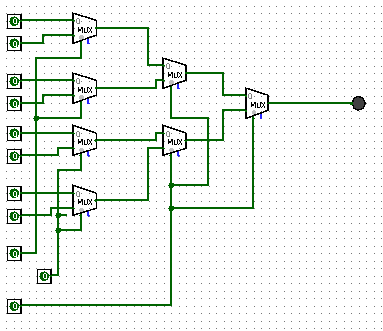
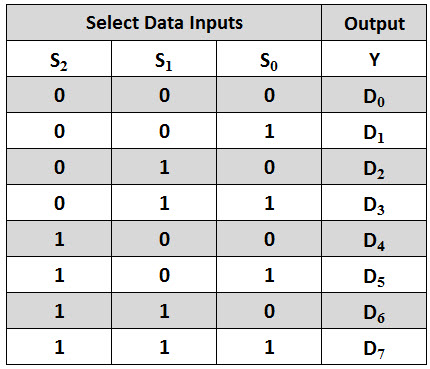
OBJECT:- Implementation of 8x1 Multiplexer.

THEORY:

An **8-to-1 multiplexer** consists of eight data inputs D0 through D7, three input select lines S2 through S0 and a single output line Y. Depending on the select lines combinations, multiplexer decodes the inputs.

[](http://www.electronicshub.org/wp-content/uploads/2015/07/8-to-1-MUX.jpg)

**Figure: 8x1 Multiplexer block Diagram**

 ****

**Figure : 8x1 Multiplexer Fig :8x1 Multiplexer Truth Table**