SRM INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

DEPT. Of Computer Science Engineering

SRM IST, Ramapuram

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	01
Title of Experiment	Design and Implementation of Half Wave and Full Wave Rectifiers using simulation package and demonstrate its working
Name of the candidate	Sathya L J K
Register Number	RA2011026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva	5	
2	Execution	10	
3	Model Calculation / Result Analysis	5	
	Total	**20**	

Staff Signature with date

Aim:

To construct a Half wave and Full wave rectifier using diode and to draw its performance characteristics.

Apparatus Required:

S.No	Particulars	Type	Range	Quantity
1	Diode	1N4001		4
2	Resistor		100 to 10000Ω	As per required
3	Capacitor		470μF	1
4	AC voltage source		4V, 50Hz	1
5	Voltage Measurement probe.			2

Software Required:

https://www.multisim.com/

1)a) Half wave rectifier:

Theory:

The process of converting an alternating current into direct current is known as rectification. The unidirectional conduction property of semiconductor diodes (junction diodes) is used for rectification. Rectifiers are of two types: (a) Half wave rectifier and (b) Full wave rectifier.

In a half-wave rectifier circuit, during the positive half-cycle of the input, the diode is forward biased and conducts. Current flows through the load and a voltage is developed across it. During the negative half cycle, it is reverse bias and does not conduct. Therefore, in the negative half cycle of the supply, no current flows in the load resistor as no voltage appears across it. Thus the dc voltage across the load is sinusoidal for the first half cycle only and a pure a.c. input signal is converted into a unidirectional pulsating output signal.

Another type of circuit that produces the same output as a full-wave rectifier is that of the Bridge Rectifier. This type of single-phase rectifier uses 4 individual rectifying diodes connected in a "bridged" configuration to produce the desired output but does not require a special center tapped transformer, thereby reducing its size and cost. The single secondary winding is connected to one side of the diode bridge network and the load to the other side. The 4 diodes labeled D arranged in "series pairs" with only two diodes conducting current during each half cycle. During the positive half cycle of the supply, diodes D1 and D2 conduct in se D3 and D4 are reverse biased and the current flows through the load as shown below . During the negative half cycle of the supply, diodes D3 and D4 conduct in series, but

diodes D1 and D2 switch of as they are now reverse biased. The current flowing through the load is the same direction as before.

Formula:

Half wave rectifier without filter:

I. $V_{rms}=\frac{V_m}{2}$; V_m = Peak voltage magnitude

II. $V_{dc}=\frac{V_m}{\pi}$

III. Ripple factor = $\sqrt{\left(\frac{V_{rms}}{V_{dc}}\right)^2 -1}$

IV. % Efficiency = $\left(\frac{V_{dc}}{V_{rms}}\right)^2 \times 100\%$

Half wave rectifier with filter:

I. $V_{rms}=\frac{V_{rpp}}{(\sqrt{3}\times 2)}$; V_{rpp} = Peak to peak voltage magnitude

II. $V_{dc}=V_m - \frac{V_{rpp}}{2}$

III. Ripple factor = $\frac{V_{rms}}{V_{dc}}$

Procedure:

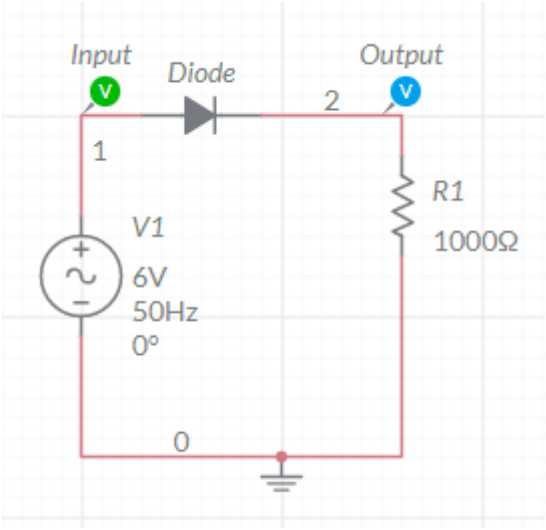
Without Filter

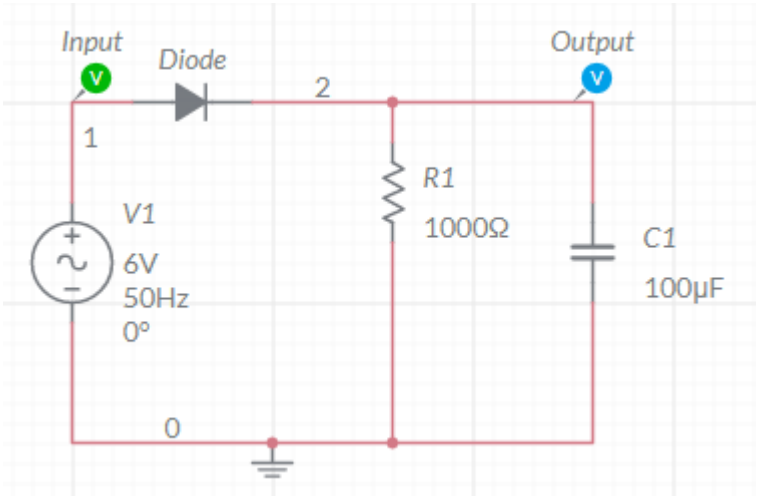
I. Give the connections as per the circuit diagram.
II. Give 6 V, 50Hz Input to the circuit.
III. Measure the rectifier output across the Load and input voltage.
IV. Plot its performance graph.

With Filter

I. Give the connections as per the circuit diagram.
II. Give 6 V, 50Hz Input to the circuit.
III. Connect the Capacitor across the load.
IV. Measure the rectifier output across the different load and input voltage
V. Plot its performance graph.

Circuit Diagram:

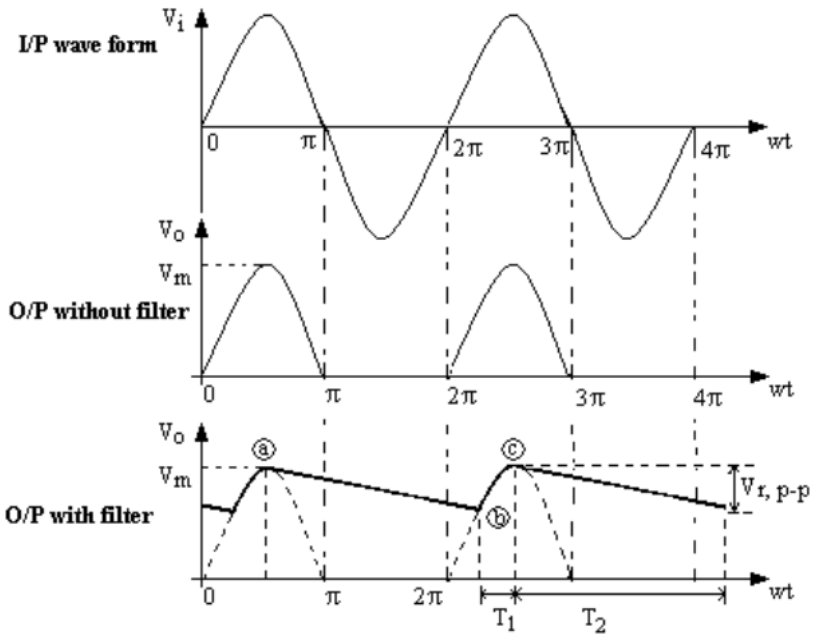




Half wave Rectifier – Without filter

Half wave Rectifier – With filter

Model graph for half wave rectifier



Tabulation:

Without Filter:

V_m (V)	V_{rms} (V)	V_{dc} (V)	Ripple Factor	Efficiency (%)
6	3	1.909	1.212	40.4

With filter:

Load Resistor	V_{rpp} (V)	V_{rms} (V)	V_{dc} (V)	Ripple factor
10 Ω	5.165	1.491	3.417	0.436
25 Ω	5.177	1.494	3.411	0.437
50Ω	4.915	1.418	3.542	0.400
100Ω	4.059	1.171	3.970	0.294
1000Ω	0.854	0.246	5.572	0.044

Model Calculation:

Without filter:

Vm=6

Vrms=6/2=3

Vdc=6/π =1.909

Ripple factor = $\sqrt{\left(\frac{3}{1.909}\right)^2 - 1}$=1.212

% Efficiency = $\left(\frac{1.909}{3}\right)^2 \times 100\%$=40.4

With filter:

For 10 ohms:
Vrpp=5.165

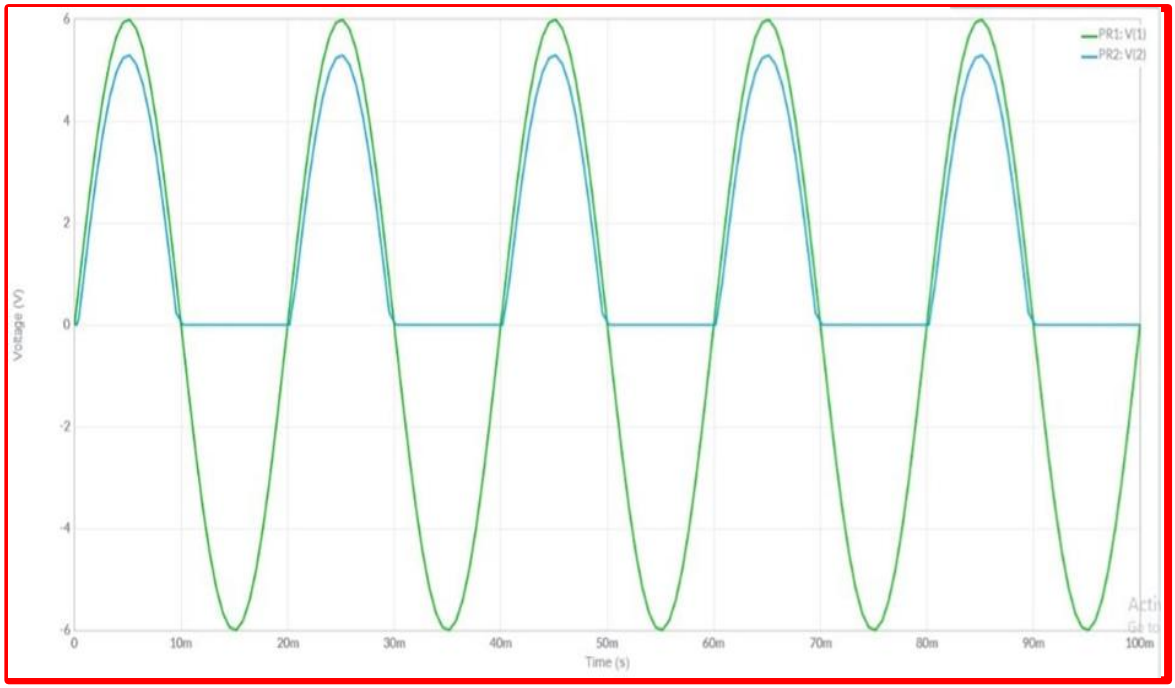
Vrms=$\frac{5.165}{2\sqrt{3}}$

Vdc=6-(5.165/2)=3.417

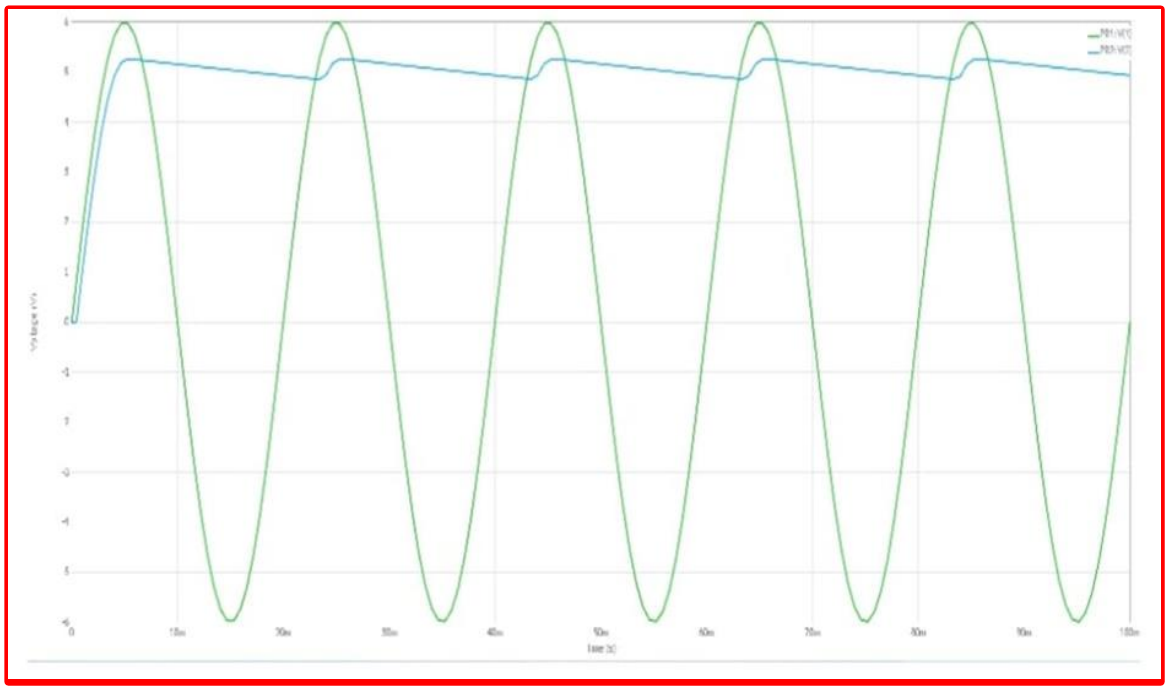
Ripple factor=1.491/3.417=0.436

Similarly done for the rest.

Simulation waveform for without filter:



Simulation waveform for with filter:

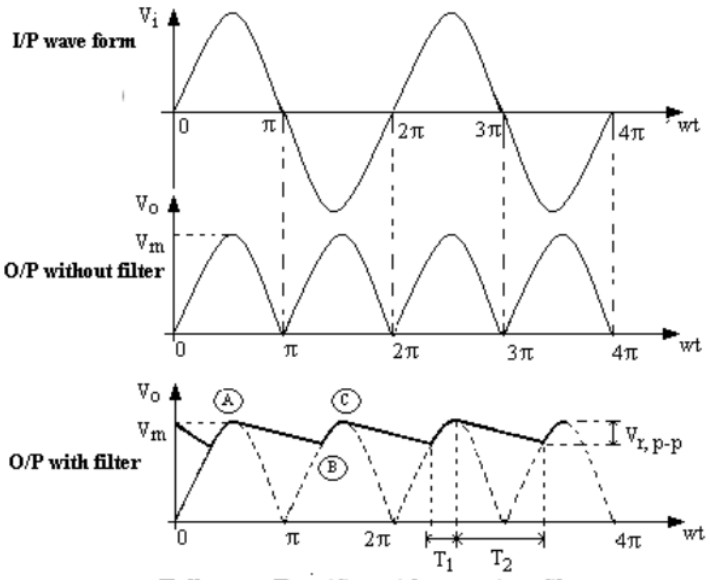


Full wave rectifier:

Theory:

Another type of circuit that produces the same output as a full-wave rectifier is that of the Bridge Rectifier. This type of single-phase rectifier uses 4 individual rectifying diodes connected in a "bridged" configuration to produce the desired output but does not require a special center tapped transformer, thereby reducing its size and cost. The single secondary winding is connected to one side of the diode bridge network and the load to the other side. The 4 diodes labeled D arranged in "series pairs" with only two diodes conducting current during each half cycle. During the positive half cycle of the supply, diodes D1 and D2 conduct in se D3 and D4 are reverse biased and the current flows through the load as shown below . During the negative half cycle of the supply, diodes D3 and D4 conduct in series, but diodes D1 and D2 switch of as they are now reverse biased. The current flowing through the load is the same direction as before.

Model Graph:



Formula:

Full wave rectifier without filter:

I. $V_{rms}=\frac{V_m}{\sqrt{2}}$; V_m = Peak voltage magnitude

II. $V_{dc}=\frac{2V_m}{\pi}$

III. Ripple factor = $\sqrt{\left(\frac{V_{rms}}{V_{dc}}\right)^2 - 1}$

IV. % Efficiency = $\left(\frac{V_{dc}}{V_{rms}}\right)^2 \times 100\%$

Full wave rectifier with filter:

I. $V_{rms}=\frac{V_{rpp}}{(\sqrt{3}\times 2)}$; V_{rpp} = Peak to peak voltage magnitude

II. $V_{dc}=V_m - V_{rpp}$

III. Ripple factor = $\frac{V_{rms}}{V_{dc}}$

Procedure:

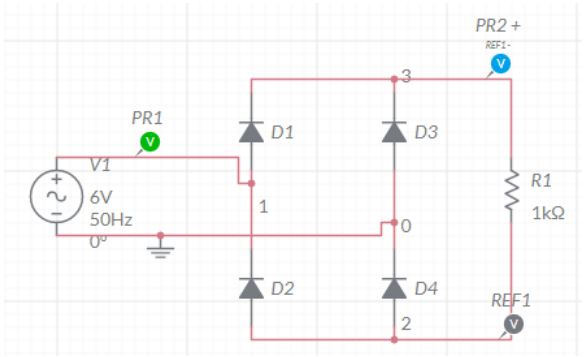
Without Filter

I. Give the connections as per the circuit diagram.
II. Give 6 V, 50Hz Input to the circuit.
III. Measure the rectifier output across the Load and input voltage.
IV. Plot its performance graph.

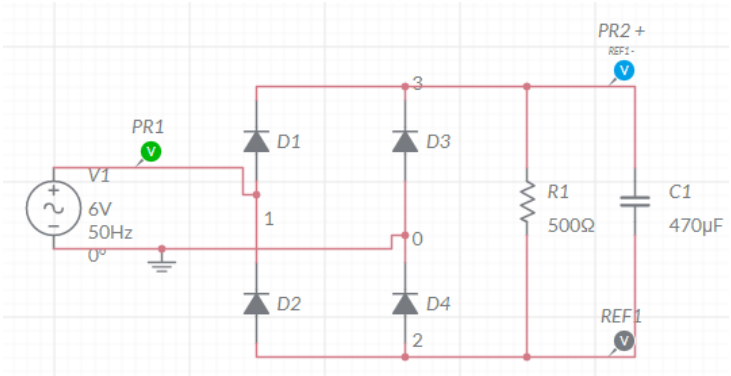
With Filter

I. Give the connections as per the circuit diagram.
II. Give 6 V, 50Hz Input to the circuit.
III. Connect the Capacitor across the load.
IV. Measure the rectifier output across the different Load and input voltage.
V. Plot its performance graph.

Circuit Diagram:



Full wave Rectifier – Without filter



Full wave Rectifier – With filter

Tabulation:

Without Filter:

V_m (V)	V_{rms} (V)	V_{dc} (V)	Ripple Factor	Efficiency (%)
6	4.242	3.819	0.483	81.05

With filter:

Load Resistance	V_{rpp} (V)	V_{rms} (V)	V_{dc} (V)	Ripple factor
10 Ω	3.022	0.872	2.978	0.292
25 Ω	1.874	0.540	4.126	0.131
50Ω	1.167	0.336	4.833	0.069
100Ω	0.5834	0.168	5.417	0.032
1000Ω	0.076	0.022	5.924	0.003

Model Calculation:

Without Filter:

Vrms = $6/\sqrt{2}$ =4.242

Vdc=(2x6)/π=3.819

Ripple factor = $\sqrt{\left(\frac{4.242}{3.819}\right)^2 - 1}$=0.483

% Efficiency = $\left(\frac{3.819}{4.242}\right)^2 \times 100\%$=81.05

With Filter:

For 10 ohms,

Vrms=3.022/(2x$\sqrt{3}$)=0.872

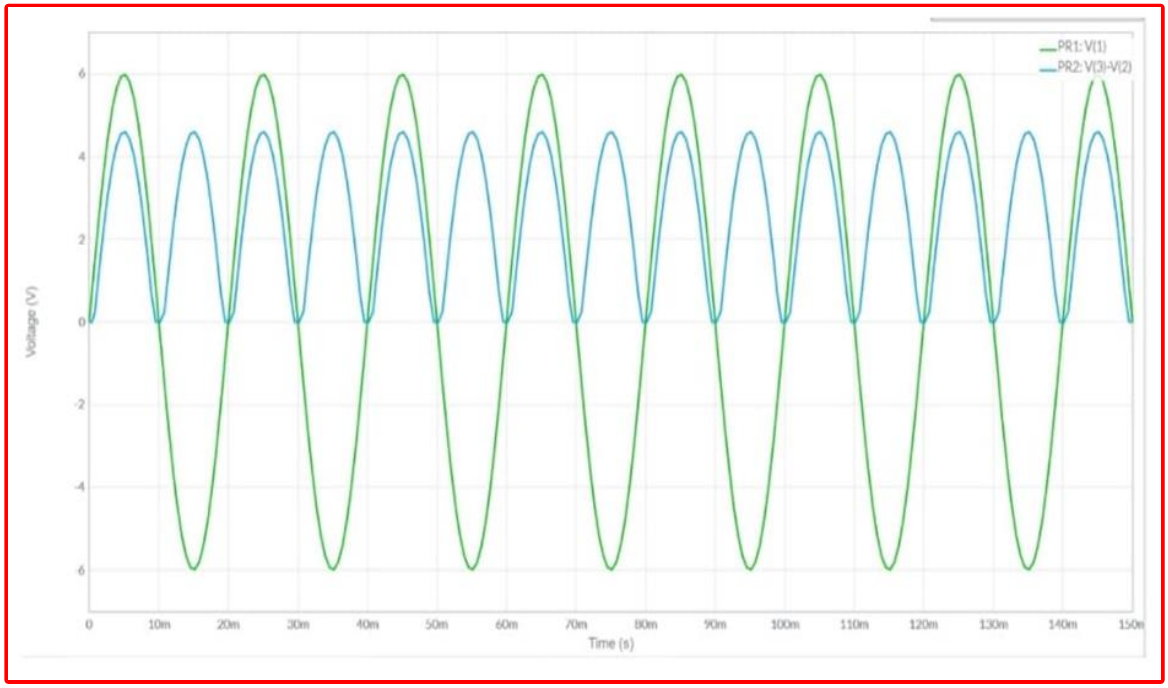
Vrpp=3.022

Vdc=6-3.022=2.978

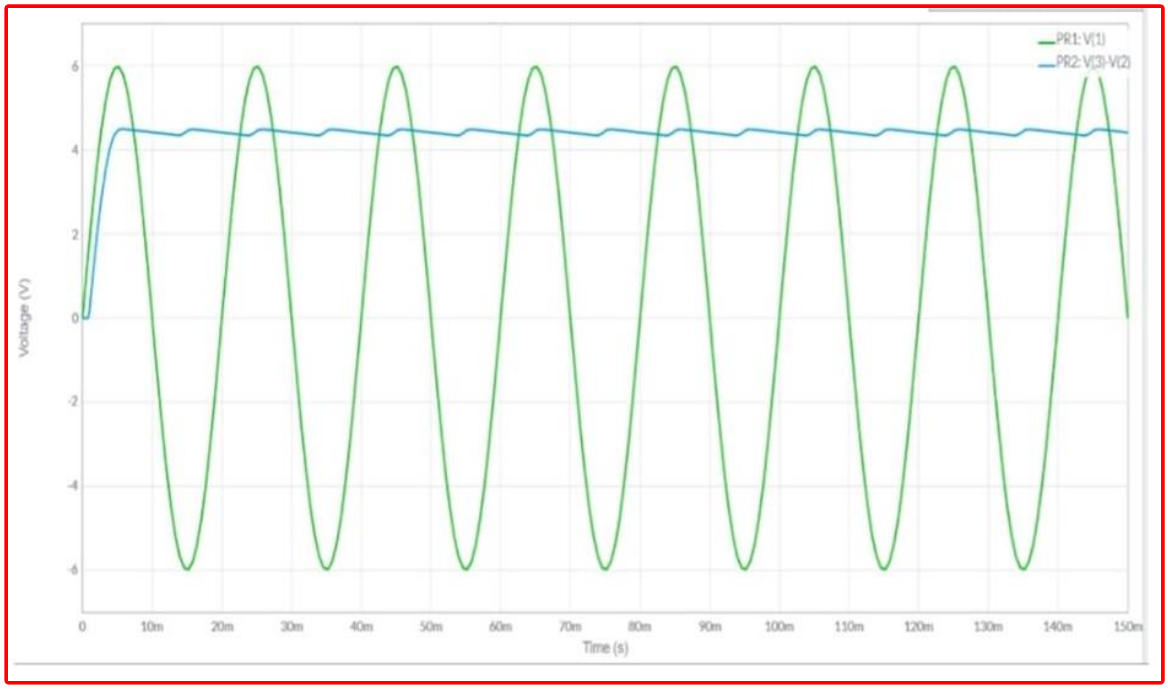
Ripple factor=0.872/2.978=0.292

Similarly done for the rest.

Simulation waveform for without filter:



Simulation waveform for with filter:



Result:

Thus, the performance characteristics of single phase Half wave and Full wave rectifier were obtained.

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

DEPT. Of Computer Science Engineering

SRM IST, Ramapuram

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	02
Title of Experiment	Design and implement a Schmitt trigger using Op-Amp using a simulation package and demonstrate its working.
Name of the candidate	Sathya L J K
Register Number	RA2011026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva	5	
2	Execution	10	
3	Model Calculation / Result Analysis	5	
	Total	**20**	

Staff Signature with date

Aim:

Design and implement a Schmitt trigger using Op-Amp using a simulation package and demonstrate its working.

Apparatus Required:

S.No	Particulars	Type	Range	Quantity
1	Op amp	UA 741 CD		1
2	Resistor		10K to 50KΩ	As per required
3	AC voltage source		10V, 500Hz	1
4	DC voltage source		15 V, 3V	3
5	Voltage Measurement			2

Software Required:

https://www.multisim.com/

Theory:

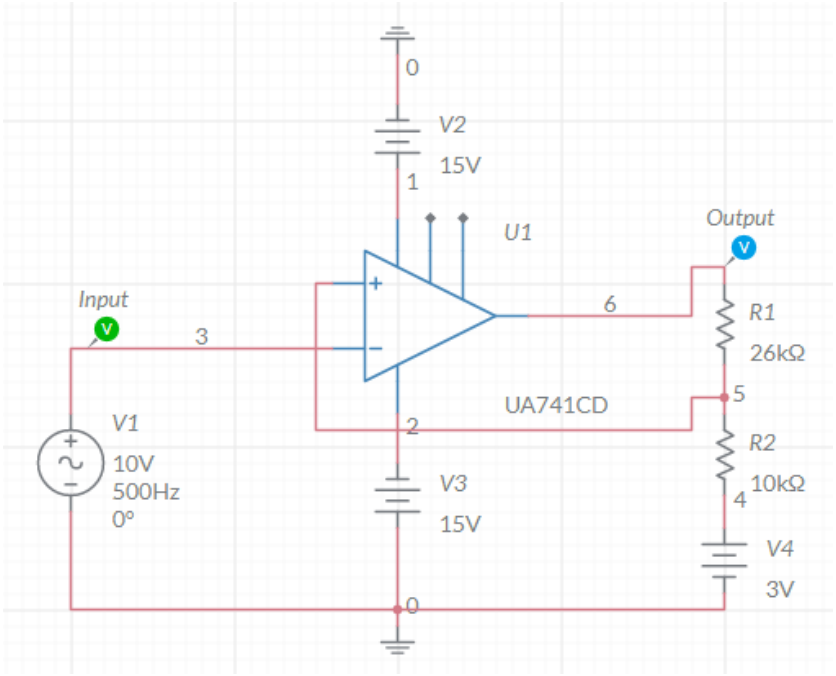
Schmitt trigger is essentially a multivibrator having two stable states. The output remains in one of the stable states indefinitely. The transition from one stable state to the other takes place when the input signal changes appropriately (triggers appropriately). Bistable peration needs an amplifier with a regenerative (positive) feedback with loop gain greater than unity. The circuit is often used to convert square waves with slowly varying edges to sharp edges required in digital circuits. It is also used for debouncing the switches. Schmitt trigger is otherwise called regenerative comparator. In this comparator circuit a positive feedback is added. The input voltage Vi triggers the output Vo very time it exceeds certain voltage levels. These voltages are known as upper threshold voltage (V_{UT}) and lower threshold voltage V_{LT} . The difference between the two threshold voltages ($V_{UT} - V_{LT}$) gives the hysteresis width.

- The Schmitt trigger is also called regenerative comparator.
- Schmitt trigger is a comparator with hysteresis.
- As it compares the input analog waveform with respect to preset values of V_{UT} and V_{LT}, Schmitt trigger is also known as two level comparator..
- A non-inverting Schmitt trigger circuit is obtained by interchanging Vi and Vref
- When an input sinusoidal signal of frequency f is applied, a square wave of same frequency is produced at the output.
- The square wave amplitude is symmetrical about zero level

Procedure:

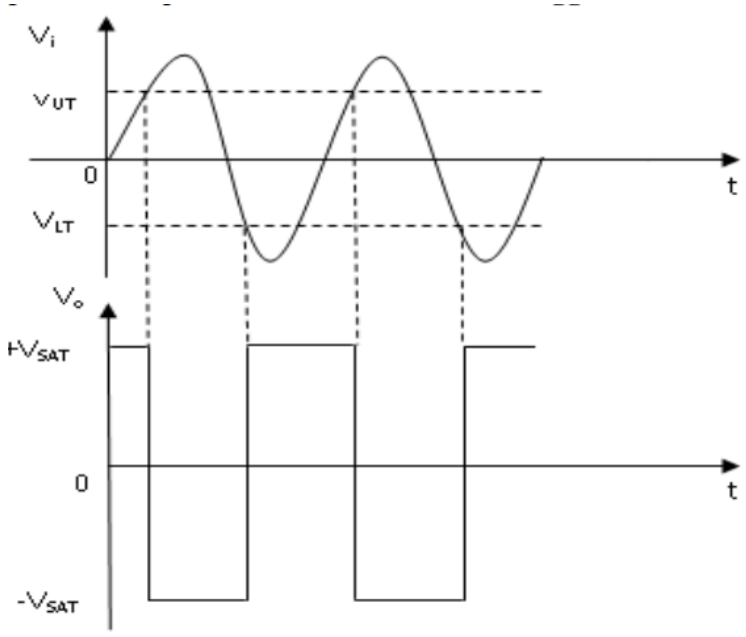
1. Connections are made as shown in the circuit diagram.
2. A sinusoidal input whose amplitude is greater than the magnitude of the V_{UT} and V_{LT}, is applied, a square wave output is obtained and tabulated the various value.
3. V_{UT} and V_{LT}, points are noted.

Circuit Diagram:

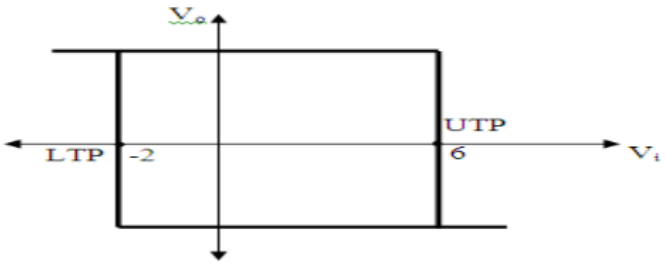


Model graph:

Input and Output waveform



Transfer Characteristics:



Design Specification

$$V_{UT} = \frac{V_R R_1}{R_1+R_2} + \frac{V_{sat} R_2}{R_1+R_2} \quad (1)$$

$$V_{LT} = \frac{V_R R_1}{R_1+R_2} - \frac{V_{sat} R_2}{R_1+R_2} \quad (2)$$

(1) – (2) $V_{UT} - V_{LT} = 2\frac{V_{sat} R_2}{R_1+R_2}$

(1) + (2) $V_{UT} + V_{LT} = 2\frac{V_R R_1}{R_1+R_2}$

Simplify the above equation,

$$V_R R_1 = V_{sat} R_2 \times \frac{V_{UT} + V_{LT}}{V_{UT} - V_{LT}}$$

Where,

V_{sat} = Saturation voltage = 13 V

R_2 = 10kΩ

V_R = Reference voltage = 3 V

Tabulation:

S.No	R_1 (kΩ)	V_{UT}		V_{LT}	
		Theoretical	Simulation	Theoretical	Simulation
1	26	4	5.5	-1	-1.1
2	28.888	5	5.2	-1	-1.4
3	30.952	6	5.5	-1	-1.05
4	21.666	6	5.6	-2	-2.1
5	32.5	7	6.6	-1	-1.1

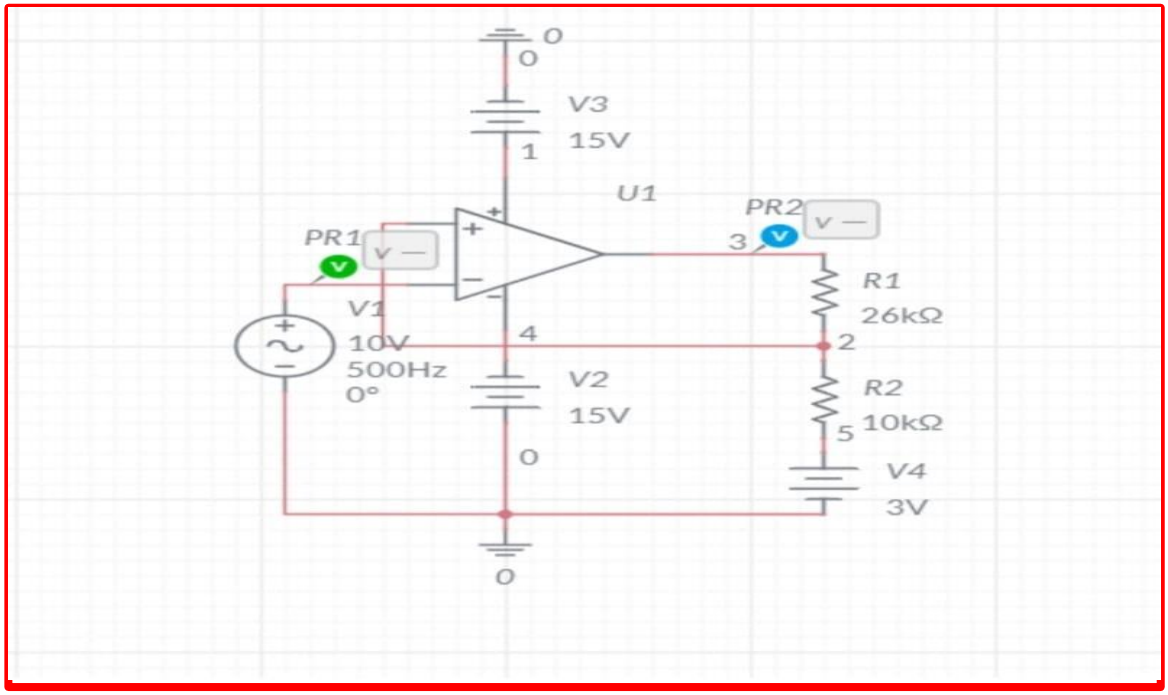
Model Calculation:

$$V_R R_1 = V_{sat} R_2 \times \frac{V_{UT} + V_{LT}}{V_{UT} - V_{LT}}$$

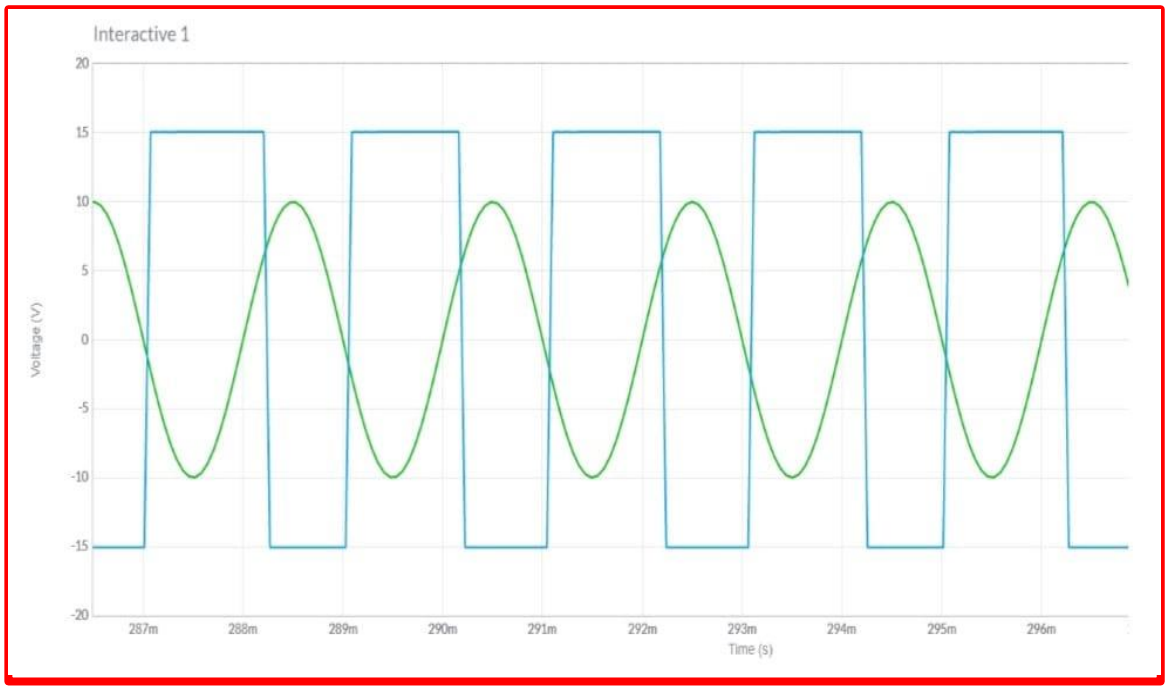
V_{sat} = 13V R_2 = 10K ohm V_R = 3V

1. R_1 = (13 x 10 x 3) / (3 x 5) = 26
2. R_1 = (13 x 10 x 4) / (3 x 6) = 28.999
3. R_1 = (13 x 10 x 5) / (3 x 7) = 30.952
4. R_1 = (13 x 10 x 4) / (3 x 8) = 21.666
5. R_1 = (13 x 10 x 5) / (3 x 8) = 32.5

Simulation Diagram:



Simulation waveform:



Result:

Thus, the design and performance of the Schmitt trigger were obtained.

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

DEPT. Of Computer Science Engineering

SRM IST, Ramapuram

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	03
Title of Experiment	Design and implement a rectangular waveform generator (Op-Amp relaxation oscillator) using a simulation package and demonstrate the working of it
Name of the candidate	Sathya L J K
Register Number	RA2011026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva	5	
2	Execution	10	
3	Model Calculation / Result Analysis	5	
	Total	**20**	

Staff Signature with date

Design and implement a rectangular waveform generator (Op-Amp relaxation oscillator) using a simulation package and demonstrate the working of it

Aim:

To design and implement a rectangular waveform generator (Op-Amp relaxation oscillator) using a simulation package and demonstrate the working of it.

Apparatus Required:

S.No	Apparatus	Type	Range	Quantity
1	OP-AMP	IC741		1
2	Resistor		1 kΩ, 1 kΩ, 100 kΩ	Each 1
3	Capacitor		10 nF	1
4	Voltage source		12 V DC	2
5	Voltage probe			1

Software Required:

https://www.multisim.com/

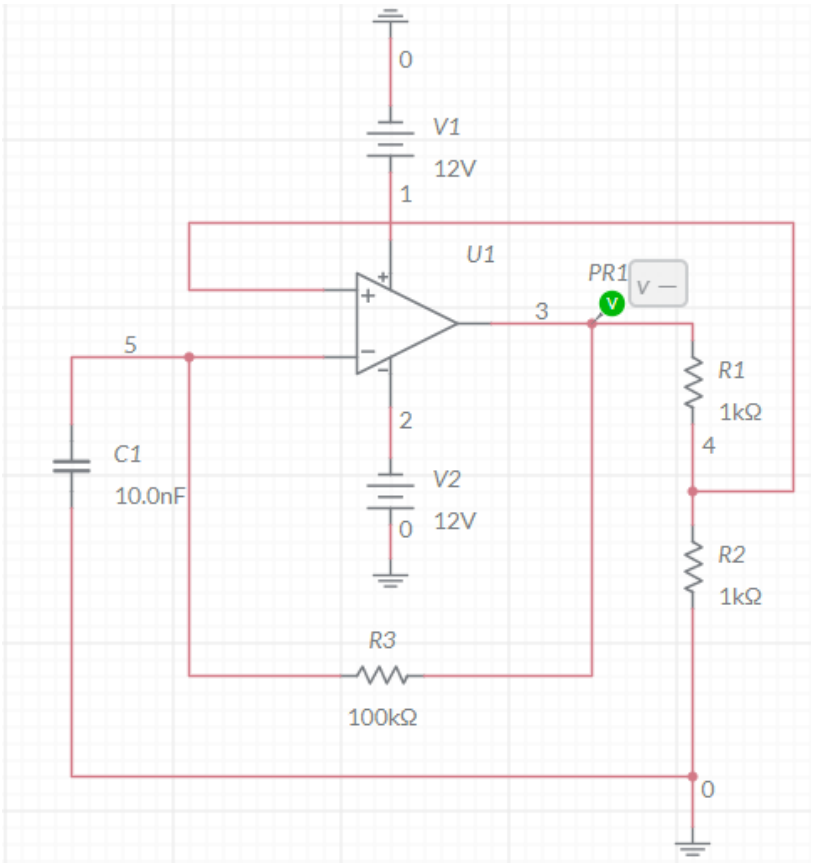
Theory:

Rectangular Waves are generated when the Op-Amp is forced to operate in the saturation region. That is, the output of the op-amp is forced to swing respectively between +Vsat And -Vsat resulting in the generation of square wave. The square wave generator is also called a free-running or astable Multivibrator Assuming the voltage across capacitor C is zero at the instant the d.c Supply voltage at +Vcc and VEE are applied. Initially the capacitance C acts, as a short circuit. The gain of the Op-Amp is very large hence V_1 drives the output of the Op-Amp to its saturation.

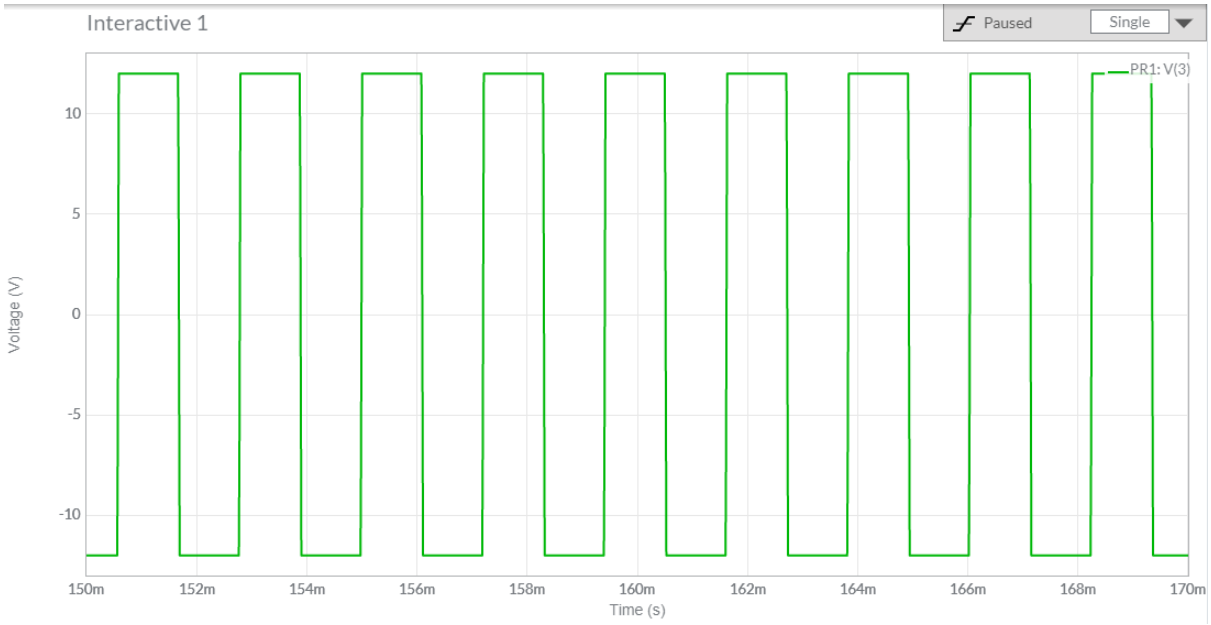
Procedure:

I. Make the connections as per the circuit diagram.
II. Adjust the values of resister and capacitor to the desired value.
III. Measure the output voltage using voltage probe and obtain the graph in grapher window.
IV. Tabulate the readings.

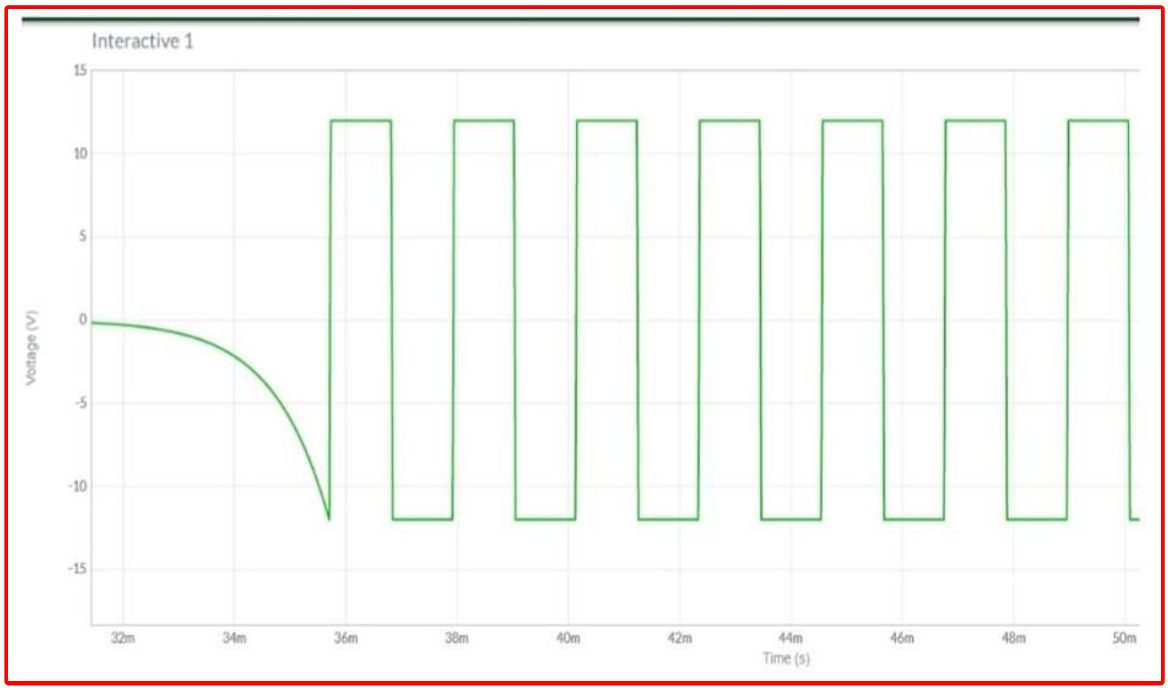
Circuit Diagram:



Model graph:



Simulation waveform:



Tabulation:

Amplitude(V)	Time period (ms)	Frequency (Hz)
12	2.209	452.6935

Result:

Thus, the rectangular wave generator was designed, and the corresponding values are tabulated.

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

DEPT. Of Computer Science Engineering

SRM IST, Ramapuram

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	04
Title of Experiment	Design and implementation of transistor as a switch
Name of the candidate	Sathya L J K
Register Number	RA2011026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva	5	
2	Execution	10	
3	Model Calculation / Result Analysis	5	
	Total	**20**	

Staff Signature with date

Aim:

1. To observe the action of a Transistor as an electronic switch.

2. To measure the voltage across the transistor when it is ON and when it is OFF.

Apparatus Required:

S.No	Apparatus	Type	Range	Quantity
1	Transistor	Q1 100 A/A		1
2	Resistor		1 kΩ,10kΩ	1 each
3	DC power source		1.5 V, 12 V	1 each
4	Switch	SPST		1
5	Probes	Voltage, current		1 each
6	LED			1
7	Ground connection			1
8	Connecting wires			As required

Software Required:

https://www.multisim.com/

Theory:

The computers of today do not process numbers in the base 10 (i.e., 0, 1, 2, 3, ...,9). Computers instead use binary logic of base 2 (0 and 1) to perform their functions. One fundamental circuit is the transistor switch, also known as an inverter. Here, a transistor connected in a common-emitter fashion inverts a signal. That is, if a high-input signal is applied, a low-output signal is created. If a low-input signal is applied, then a high output signal is created.

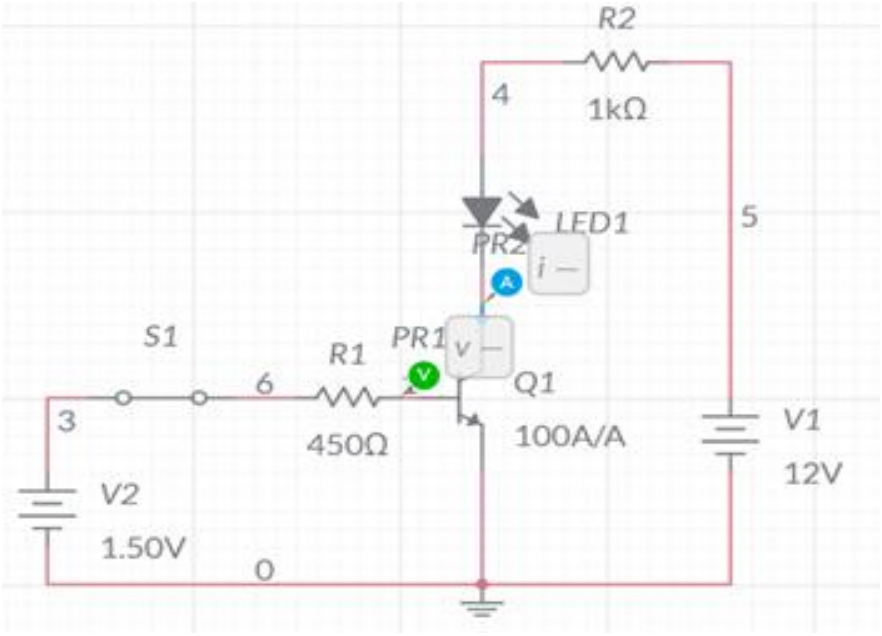
In a transistor switch circuit, a voltage level applied to the base terminal will control the potential at the collector. In this fashion, the transistor can be used to turn on or off circuitry connected to the collector. This common-emitter circuit is being switched from cutoff to saturation. In this experiment, a transistor will be connected to demonstrate this switching ability.

Procedure:

1. Log in Multisim Live Online Circuit Simulator.
2. Click create circuit button.
3. Change the untitled circuit as Transistor as a switch.
4. Click search for component and type components. Select it and drag to the Schematic window

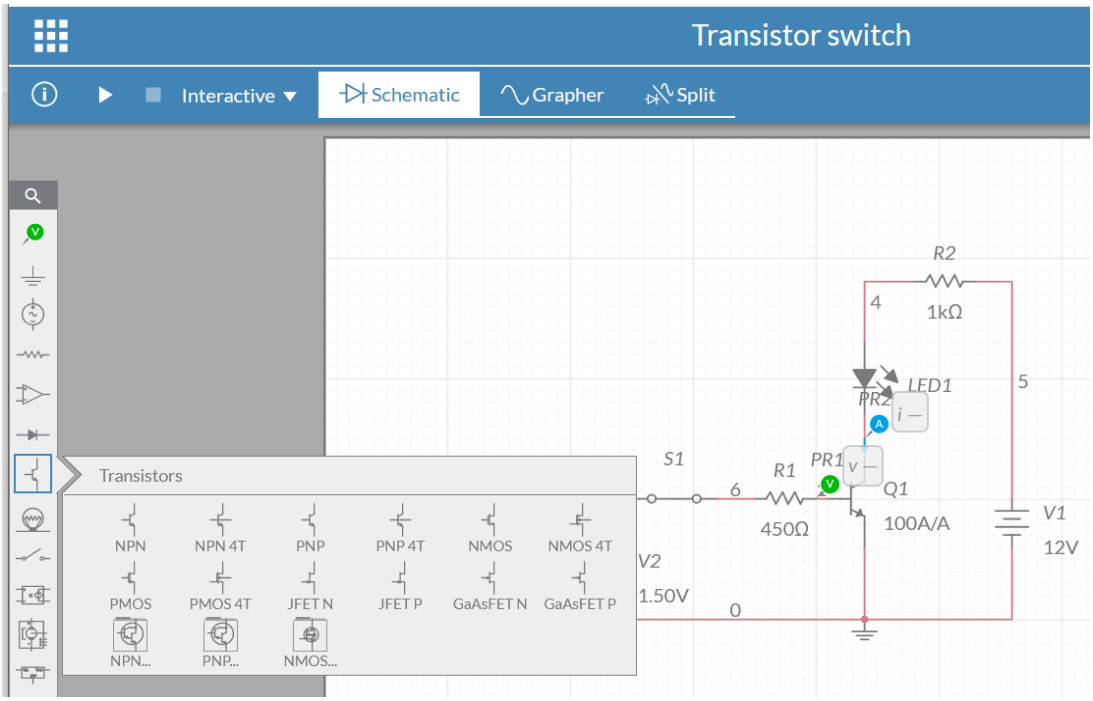
5. Click search for transistor and NPN. Select it and drag to the Schematic window. Follow this step to select the entire apparatus given in table to complete the circuit.
6. Click schematic connector and select junction drag to the Schematic window and left click at the point and drag to the other point to make the wire connection. Complete the connection according to the diagram.
7. Click analysis and annotation and select voltage probe and drag to the schematic window and place at the input source and left click. Another current probe place at the top side of the collector of transistor and left click.
8. Click the value of the components in circuit and enter the value in the circuit document.
9. Save the file by clicking the file navigation menu at the left top and save with a file name.
10. Click grapher. Enter end time as 1e29 s.
11. Run the simulation by clicking the run simulation. Switch on and off SPST switch at regular intervals.
12. Observe the input and output wave form and note down the values during ON and OFF condition of the transistor.

Circuit Diagram:



Transistor as a switch

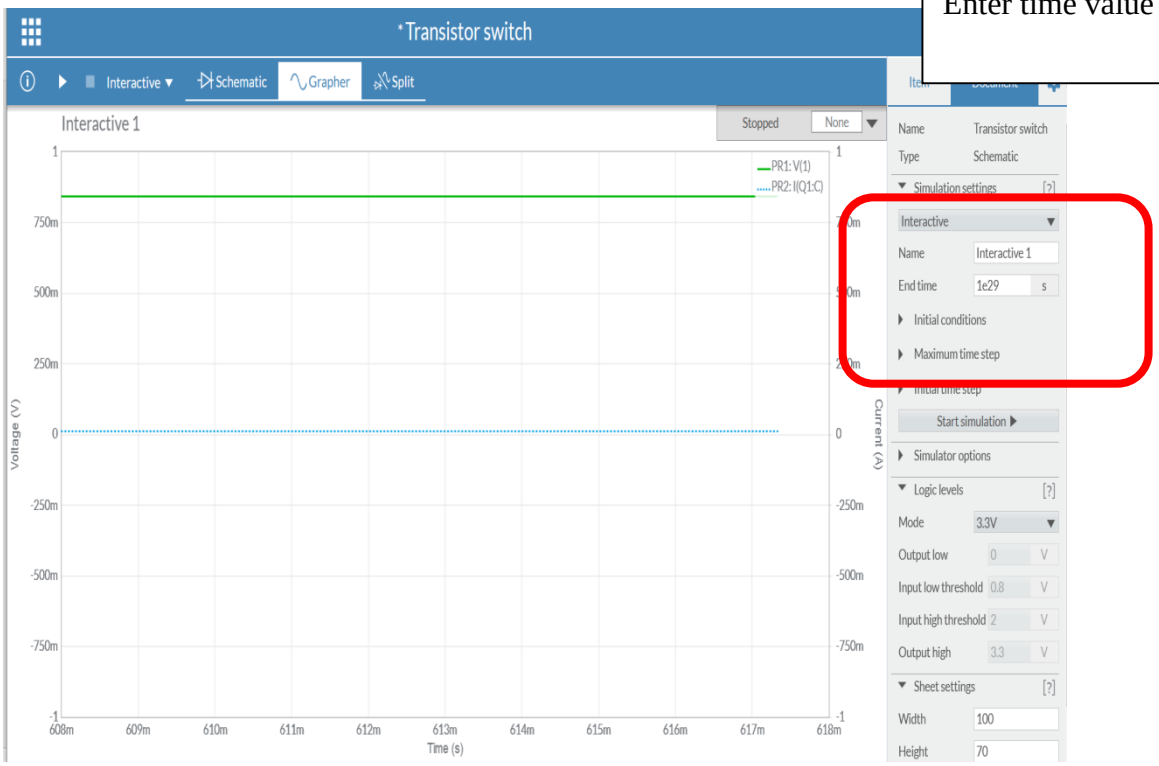
Method for component selection



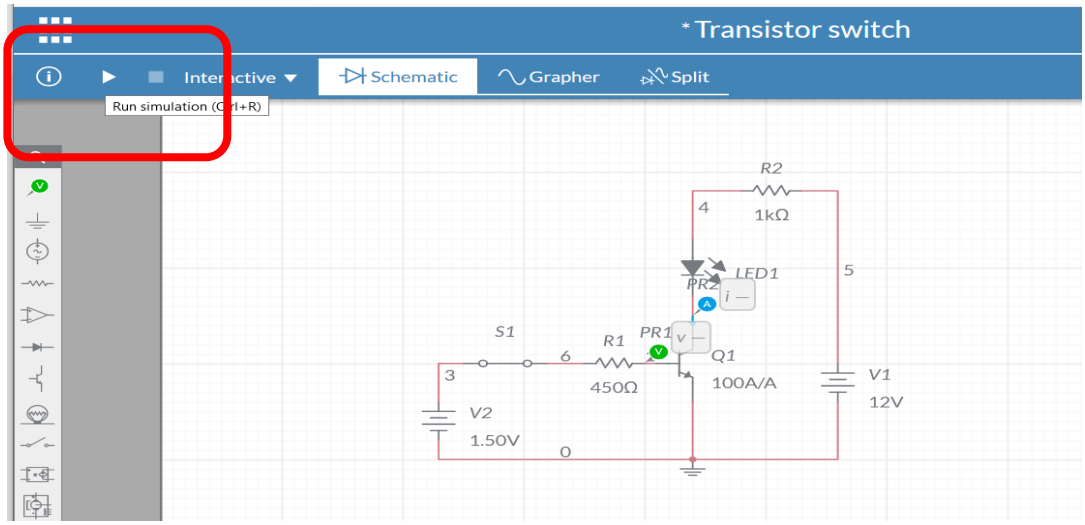
- Click search for transistor and NPN. Select it and drag to the Schematic window. Follow this step to make entire circuit as shown in table

Method for observing the grapher to provide time details

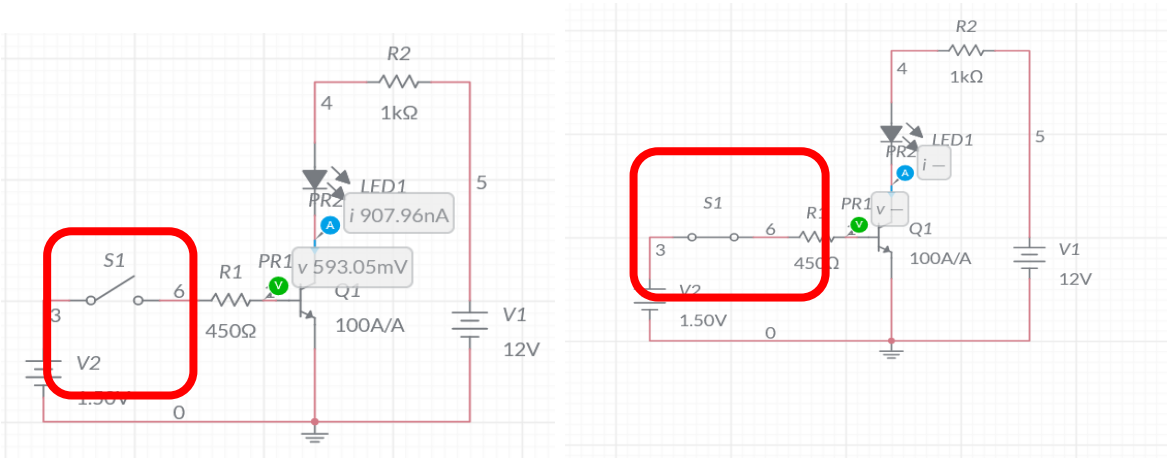
Enter time value here



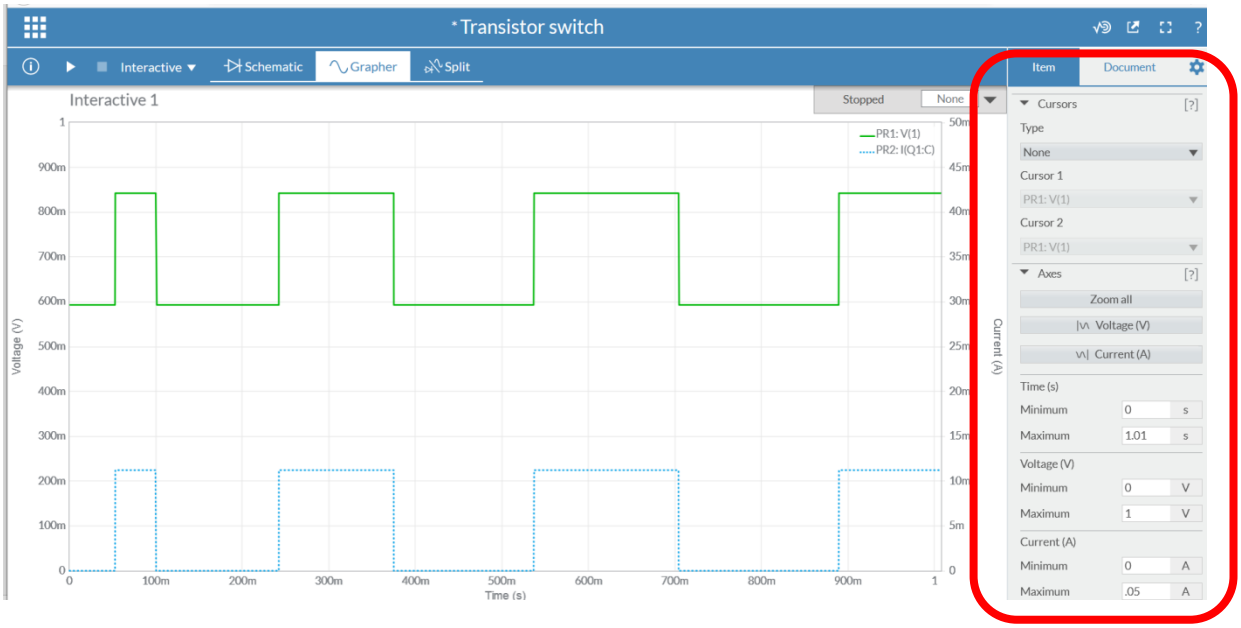
Method to run the simulation



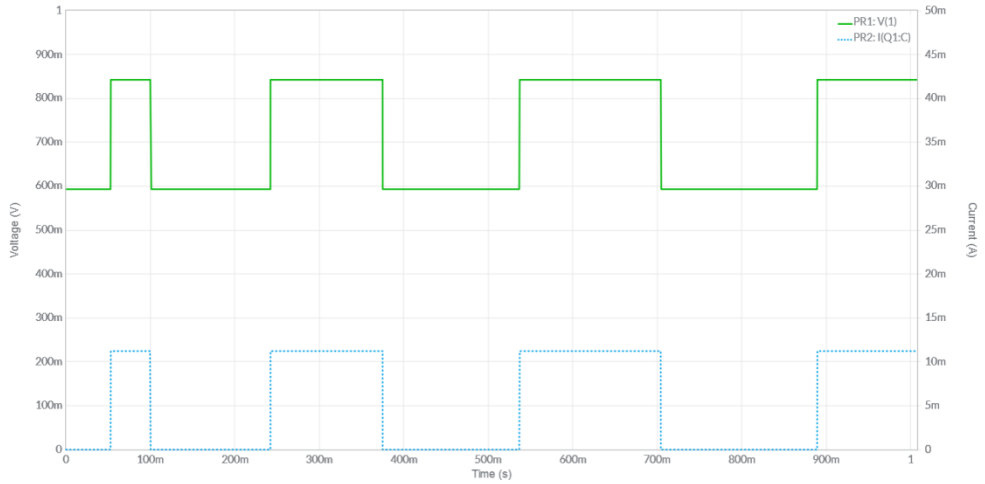
Switch ON and OFF condition



Method to get proper scales for the waveforms



Expected output waveforms:



Tabulation:

Switch	Status of probe	Voltage value
ON	Glowing	842.22
OFF	Not glowing	593.05

Result:

Thus, the transistor as a switch was designed and the output voltage and status of the Probe was tabulated.

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

DEPT. Of Computer Science Engineering

SRM IST, Ramapuram

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	05
Title of Experiment	Design CMOS Inverter and measure its propagation delay using Multisim Live Online Circuit Simulator.
Name of the candidate	Sathya L J K
Register Number	RA22011026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva	5	
2	Execution	10	
3	Model Calculation / Result Analysis	5	
	Total	**20**	

Staff Signature with date

Aim:

To Design CMOS Inverter and measure its propagation delay.

Apparatus Required:

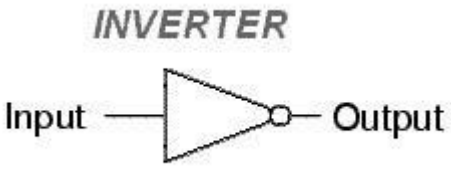
S.No	Apparatus	Type	Range
1	Transistor	Pmos4T	
2	Transistor	Nmos4T	
3	Clock Voltage		
4	Capacitor		500fF

Software Required:

https://www.multisim.com/

THEORY

The inverter is universally accepted as the most basic logic gate doing a Boolean operation on a single input variable. Fig.1 depicts the symbol, truth table and a general structure of a CMOS inverter. As shown, the simple structure consists of a combination of an pMOS transistor at the top and a nMOS transistor at the bottom.



Input	Output
1	0
0	1

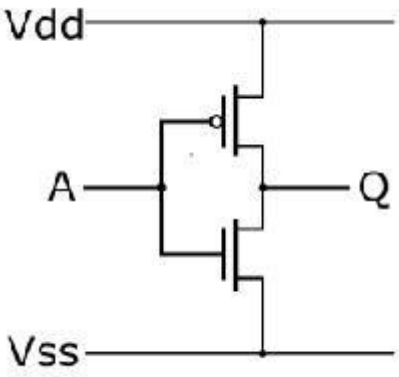


Fig.1: Symbol, circuit structure and truth table of a CMOS inverter

CMOS is also sometimes referred to as complementary-symmetry metal–oxide– semiconductor. The words "complementary-symmetry" refer to the fact that the typical digital design style with CMOS uses complementary and symmetrical pairs of p-type and n-type metal oxide semiconductor field effect transistors (MOSFETs) for logic functions. Two important characteristics of CMOS devices are high noise immunity and low static power consumption. Significant power is only drawn while the transistors in the CMOS device are switching between on and off states. Consequently, CMOS devices do not produce as much waste heat as other forms of logic, for example transistor-transistor logic (TTL) or NMOS logic, which uses all n-channel devices without p-channel devices. Fig. 2 shows the propogation delay graph.

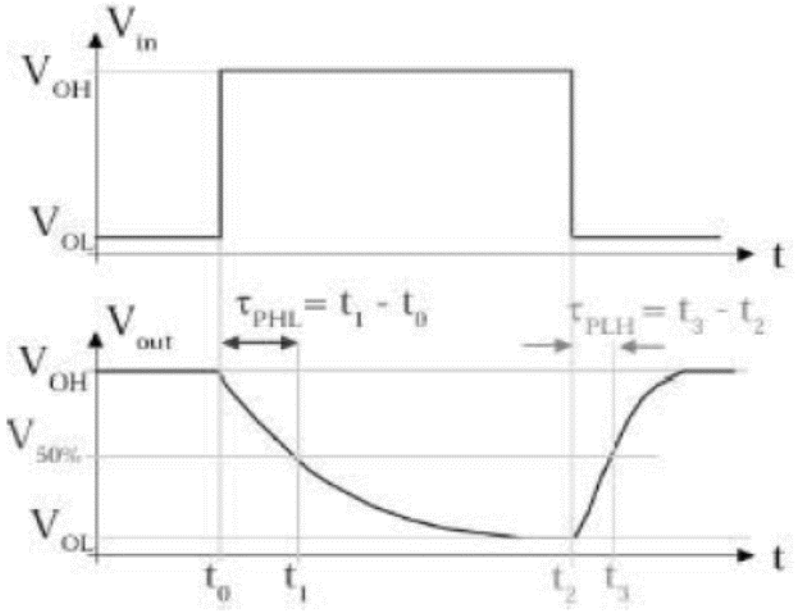


Fig.2 Propogation delay graph

The propagation delay t_p of a gate defines how quickly it responds to a change at its inputs. It express the delay experienced by a signal when passing through a gate. It is measured between the 50% transition points of the input and output waveforms. The τ_{pLH} defines the response time of the gate for a low to high output transition. The τ_{pHL} defines the response time of the gate for a high to low output transition. The propagation delay t_p is the average of the two.

Formula:

$$\tau_p = \left(\frac{\tau_{pHL} + \tau_{pLH}}{2} \right)$$

Procedure:

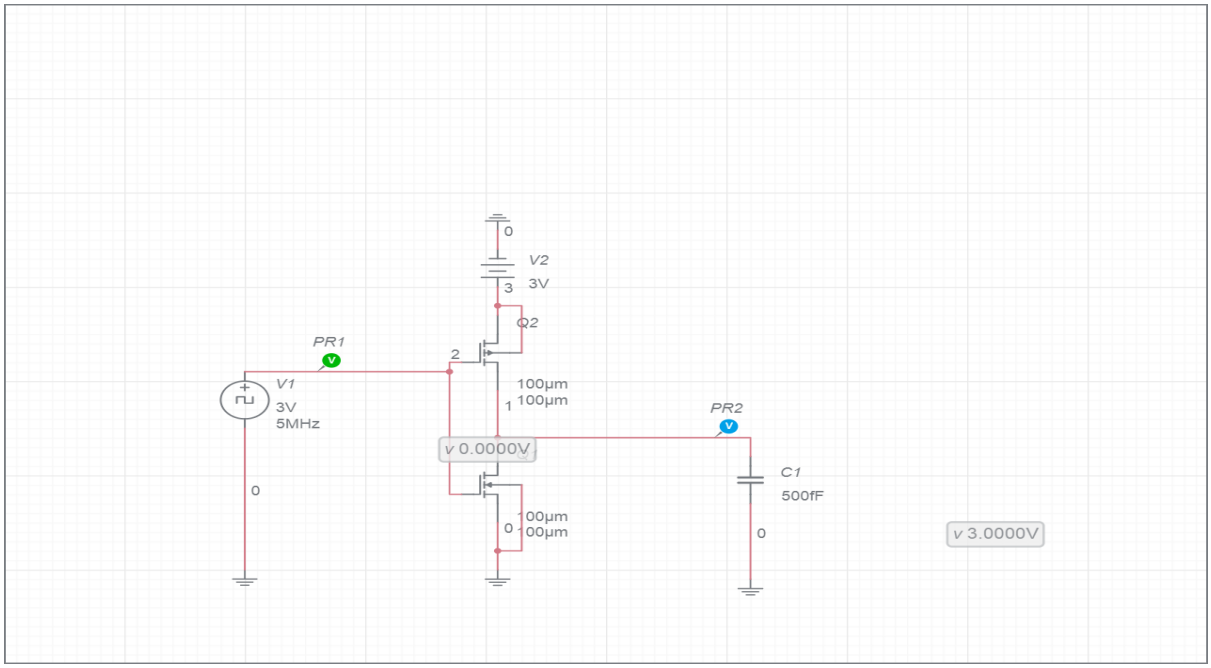
I. Give the connections as per the circuit diagram.

II. Give 3 V, 5MHz Input to the circuit.

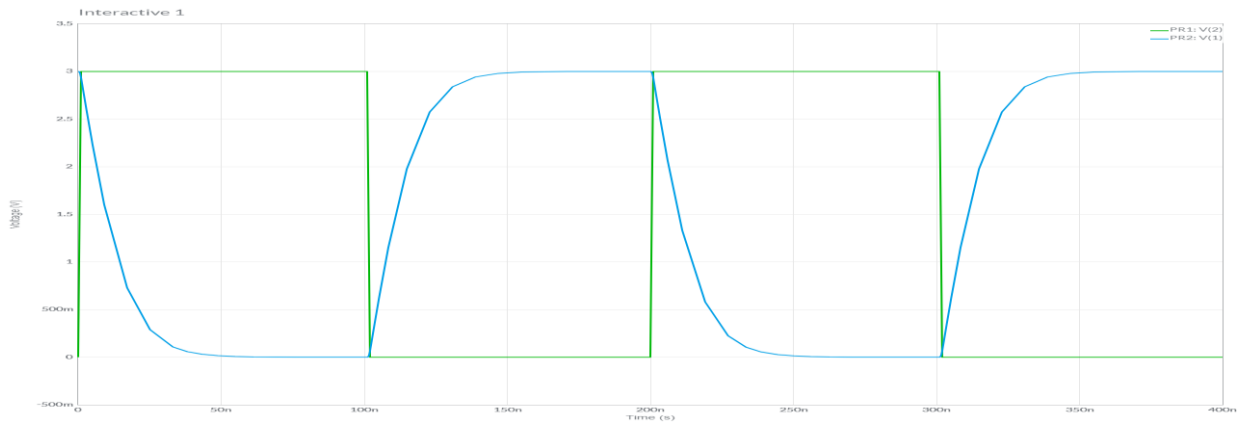
III. Measure the inverter output across the capacitor and input voltage.

IV. Plot its performance graph and measure the propagation delay from the output waveform.

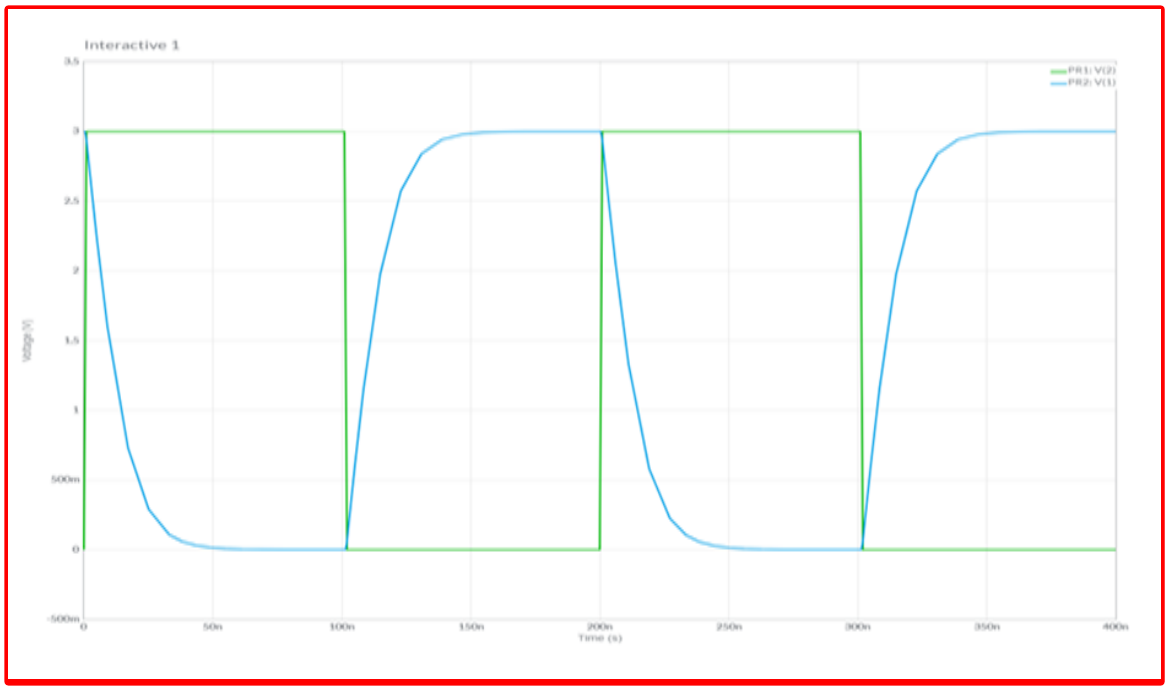
Circuit Diagram:



Model graph:



Simulation waveform for the inverter:



Model Calculation:

$$\tau_p = \left(\frac{\tau_{pHL} + \tau_{pLH}}{2} \right)$$

τ_{pHL} = 9.13 – 0 = 9.13

τ_{pLH} = 1.12 – 1.01 = 0.11

$$\tau_p = \left(\frac{9.13 + 0.11}{2} \right)$$

$$= \frac{9.24}{2}$$

τ_p = **4.62**

Result:

Thus, the CMOS Inverter is simulated and the propagation delay is measured.

DEPT. of Computer Science Engineering

SRM IST, Ramapuram

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	06
Title of Experiment	**Design and implementation of Binary to gray code converters and vice versa using logic gates**
Name of the candidate	Sathya L J K
Register Number	RA2011026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva	5	
2	Circuit Connection and Execution	10	
3	Verification of truth table	5	
	Total	**20**	

Staff Signature with date

6.a. Design and implementation of Binary to gray code converters using logic gates

Aim:

1. To design and implementation of Binary to gray code converters using Multisim-online software.
2. Hardware Implementation of the same with virtual Lab - IIT Bombay

Software Required:

https://www.multisim.com/

Apparatus Required:

S.No	Apparatus	Type	Range	Quantity
1	IC	IC 7486		1
2	LED			4
3	Switch			4
4	DC Power Source			1
5	Multisim online virtual lab IIT Bombay			
6	Wires			As Required

Theory:

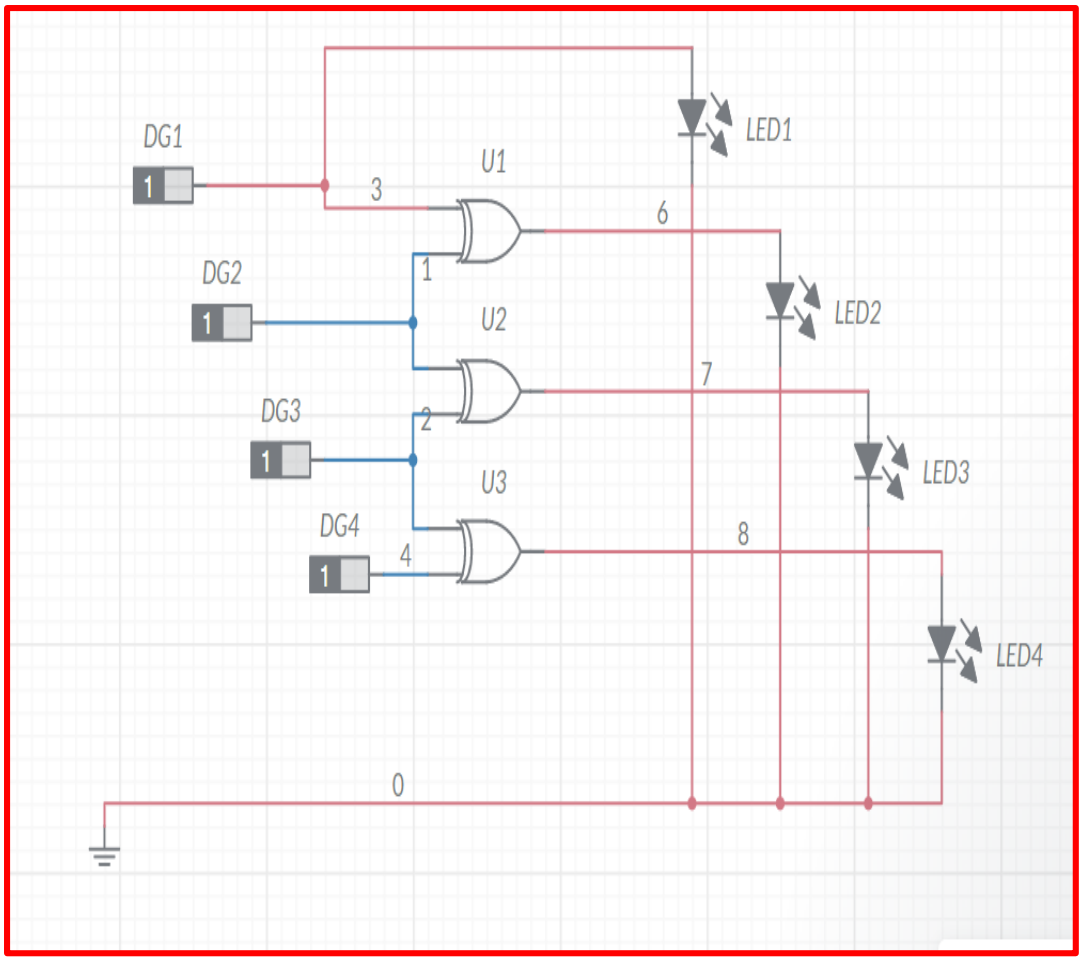
The logical circuit which converts binary code to equivalent gray code is known as binary to gray code converter. The gray code is a non-weighted code. The successive gray code differs in one-bit position only that means it is a unit distance code. It is also referred as cyclic code. It is not suitable for arithmetic operations. It is the most popular of the unit distance codes. It is also a reflective code. An n-bit Gray code can be obtained by reflecting an n-1-bit code about an axis after 2n-1 rows, and putting the MSB of 0 above the axis and the MSB of 1 below the axis. This method uses an Ex-OR gate to perform among the binary bits. In this conversion method, take down the MSB bit of the present binary number, as the primary bit or MSB bit of the gray code number is similar to the binary number.

To get the straight gray coded bits for generating the corresponding gray coded digit for the given binary digits, add the primary digit or the MSB digit of binary number toward the second digit & note down the product next to the primary bit of gray code, and add the next binary bit to third bit then note down the product next to the 2^{nd} bit of gray code. Similarly, follow this procedure until the final binary bit as well as note down the outcomes depending on EX-OR logic operation to generate the corresponding gray coded binary digit.

Procedure:

1. Open Multisim. Sign in.
2. Open a new circuit file
3. Select the components
    - Go to digital, choose digital constants
    - Go to digital, choose XOR 2 input gate
    - Go to indicator, choose LED bulbs
    - Go to schematic connectors, choose ground
4. Duplicate the components and connect them as per circuit diagram.
5. From analysis and annotation choose digital probe and place them where the output bits are to be seen
6. Run the simulation and verify the output
7. To change the input values, vary the values of high and low on the digital constants and verify the complete truth table illustrated below.

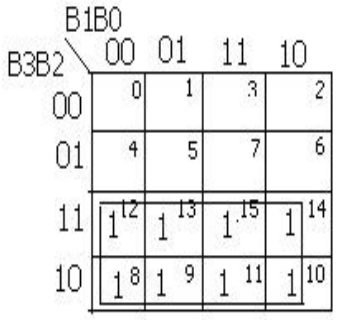
Circuit Diagram:



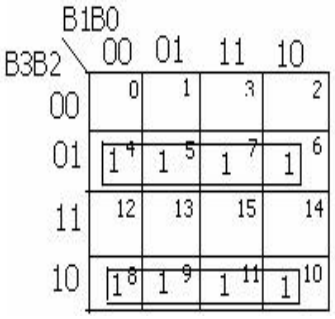
Truth Table:

BINARY				GRAY CODE			
B3	**B2**	**B1**	**B0**	**G3**	**G2**	**G1**	**G0**
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

$G3 = \sum(8,9,10,11,12,13,14,15)$



$G2 = \sum(4,5,6,7,8,9,10,11)$

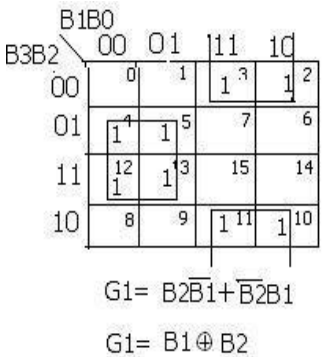


$G3 = B3$

$G2 = \overline{B3}B2 + B3\overline{B2}$

$G2 = B3 \oplus B2$

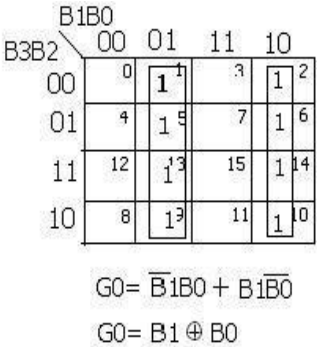
$G1 = \sum(2,3,4,5,10.,11,12,13)$



$G1 = B2\overline{B1} + \overline{B2}B1$

$G1 = B1 \oplus B2$

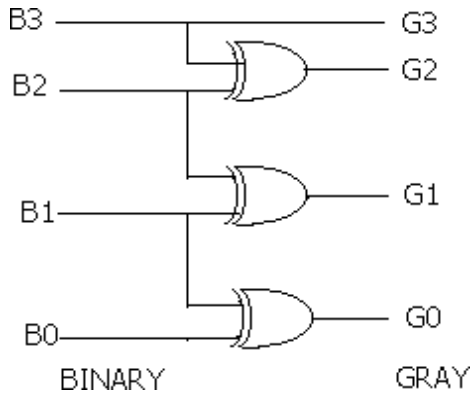
$G0 = \sum(1,2,3,5,6,9,10,13,14)$



$G0 = \overline{B1}B0 + B1\overline{B0}$

$G0 = B1 \oplus B0$

Binary to Gray code converter Using XOR Gates Only



$G3 = B3$

$G2 = \overline{B3}B2 + B3\overline{B2}$

$G2 = B3 \oplus B2$

$G1 = B2\overline{B1} + \overline{B2}B1$

$G1 = B1 \oplus B2$

$G0 = \overline{B1}B0 + B1\overline{B0}$

$G0 = B1 \oplus B0$

6.b. Design and implementation of Gray to Binary code converters using logic gates

Aim:

To design and implementation of Gray to Binary code converters using Multisim.

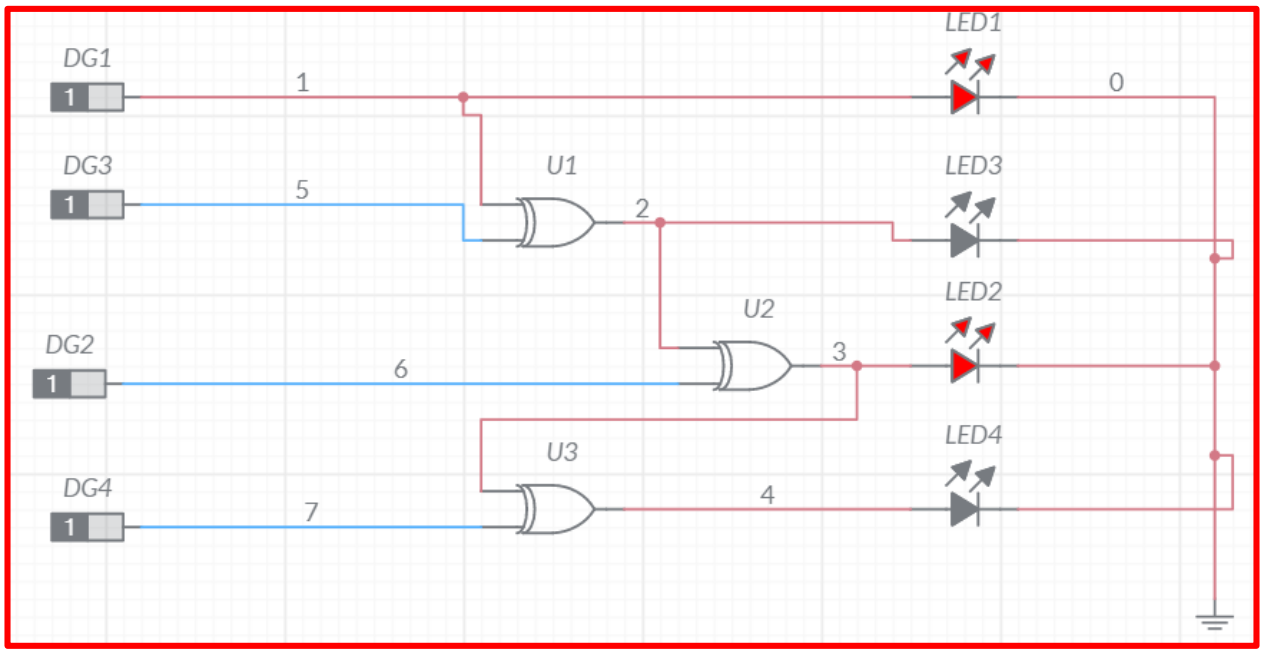
Software Required:

https://www.multisim.com/

PROCEDURE:

1. Open Multisim. Sign in.
2. Open a new circuit file
3. Select the components
    - Go to digital, choose digital constants
    - Go to digital, choose XOR 2 input gate
    - Go to indicator, choose LED bulbs
    - Go to schematic connectors, choose ground
4. Duplicate the components and connect them as per circuit diagram.
5. From analysis and annotation choose digital probe and place them where the output bits are to be seen
6. Run the simulation and verify the output
7. To change the input values, vary the values of high and low on the digital constants and verify the complete truth table illustrated below.

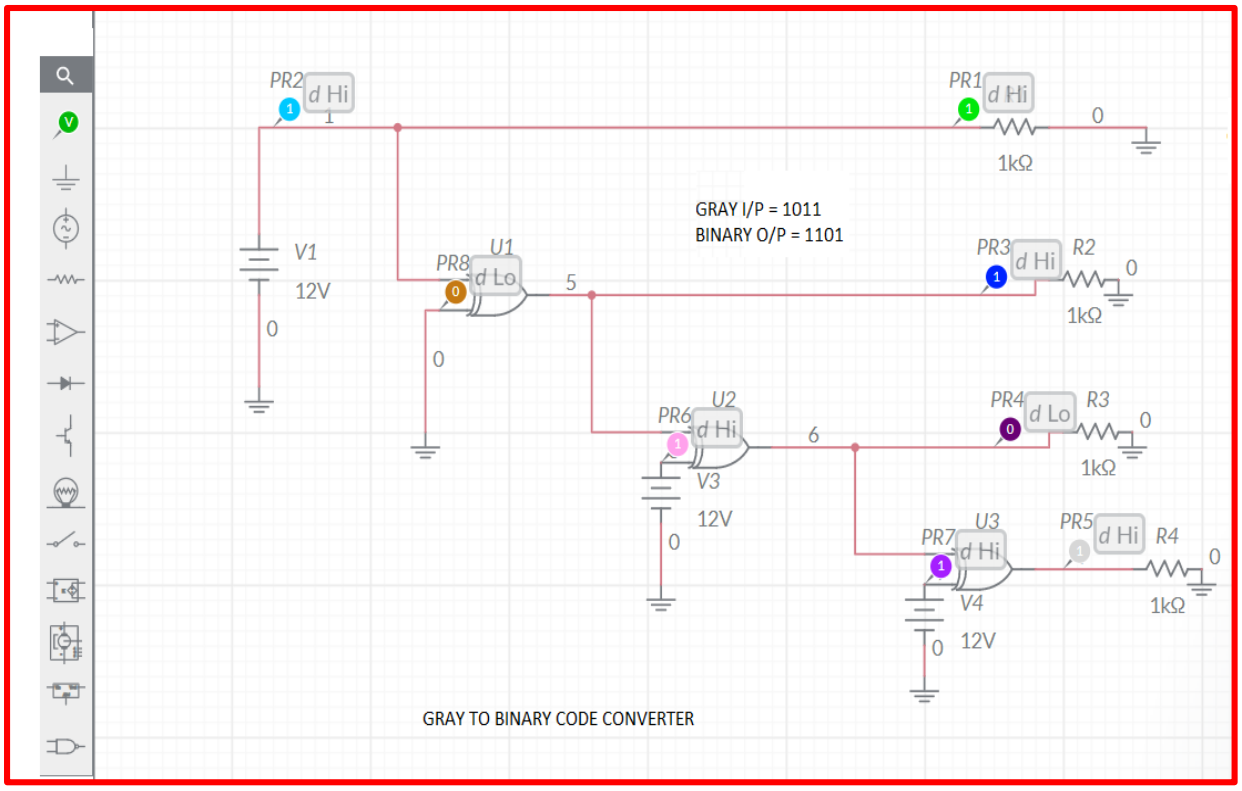
Circuit Diagram:



ALTERNATE WAY

Procedure:

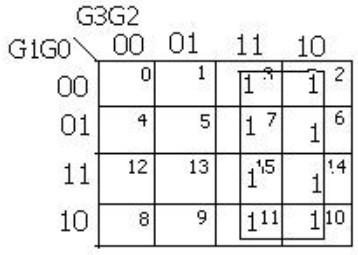
1. Open Multisim.
2. Select file navigation menu → New file
3. Select the components
    a. Go to source choose DC source and give its value as 5V
    b. Go to digital choose XOR 2 input gate
    c. Go to passive choose resistor and give its value as 100Ω.
    d. Go to schematic connectors and choose ground
4. Duplicate the components and connect them as per circuit diagram.
5. From analysis and annotation choose digital probe and place them where the output bits are to be seen.
6. Run the simulation and verify the output
7. To change the input values connect it to the dc source or connect it to ground and verify the complete truth table given below.



Truth Table:

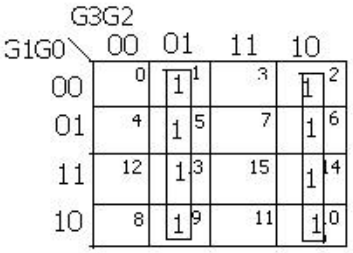
GRAY CODE				BINARY CODE			
G3	**G2**	**G1**	**G0**	**B3**	**B2**	**B1**	**B0**
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
1	0	0	0	1	1	1	1
1	0	0	1	1	1	1	0
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	0	1	0	1	1
1	1	1	1	1	0	1	0

K MAP FOR B3



B3=G3

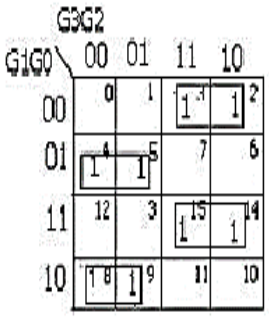
K MAP FOR B2



$B2=\overline{G3}G2 + G3\overline{G2}$

$B2=G3 \oplus G2$

K MAP FOR B1



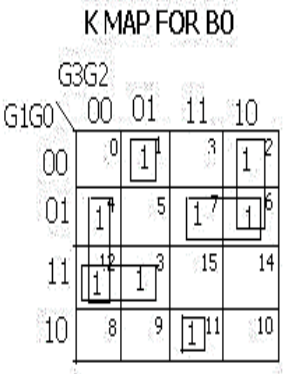
$B1=\overline{G1}G0\overline{G3}+\overline{G1}\overline{G0}G3+G1G0G3+G1\overline{G0}\overline{G3}$

$=\overline{G1}(G0\overline{G3}+\overline{G0}G3)+G1(G0G3+\overline{G0}\overline{G3})$

$=\overline{G1}(G0 \oplus G3)+G1(\overline{G0 \oplus G3})$

$B1=G3 \oplus G2 \oplus G1$

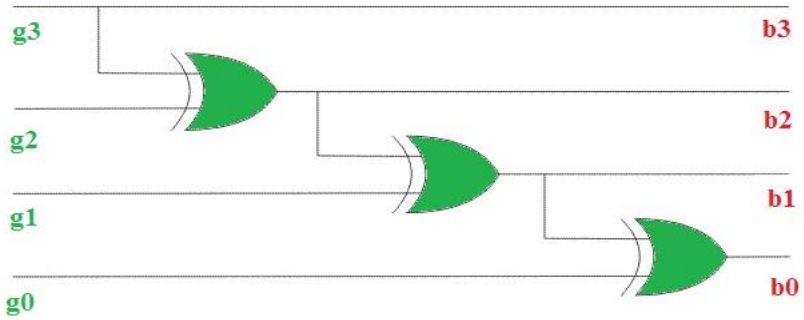
K MAP FOR B0



$B0=\overline{G1}\overline{G0}\overline{G3}G2+G1\overline{G0}G2G3+G0G3G2+G1G0\overline{G3}+\overline{G1}G0G3+\overline{G1}G3\overline{G2}$

$B0=G0 \oplus G1 \oplus G2 \oplus G3$

Gray to Binary code converter Using XOR Gates Only



6.c Hardware Implementation of Code Converters Using NI Analog Discovery 2

Aim:

Hardware Implementation of the code converter using NI Analog Discovery 2.

Apparatus Required:

S.No	Apparatus	Types	Range	Quantity
1	IC	IC 7486		
2	NI Analog Discovery 2			
3	Wires			As Required
4	Bread board			1

Theory:

Introduction to NI Analog Discovery 2(AD 2):

The Analog Discovery 2 transforms any PC into an electrical engineering workstation. This USB-powered device enables students to build and test analog and digital circuits in any environment with the functionality of traditional benchtop instruments. In addition to the 100 MS/s two-channel oscilloscope, the Analog Discovery 2 provides a two-channel waveform generator, 16-channel logic analyzer, 16-channel digital pattern generator, spectrum analyzer, network analyzer, voltmeter, and ±5 VDC adjustable power supplies.

Introduction to Virtual Lab – IIT Bombay

The objective of VLabsDev is to involve the community to re-think on the best practices regarding pedagogy, storyboards, lab manuals, documentation and the technologies for building a high-quality simulator.

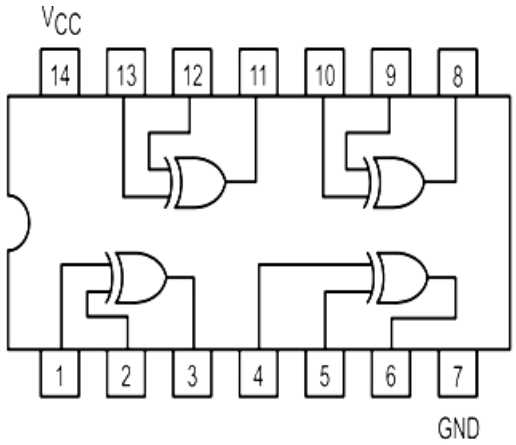
The main activities of the VLabsDev portal currently are content creation through community, hosting the community created content, the source codes, hackathons for code and content development, events for instructors for pedagogy and storyboard design.

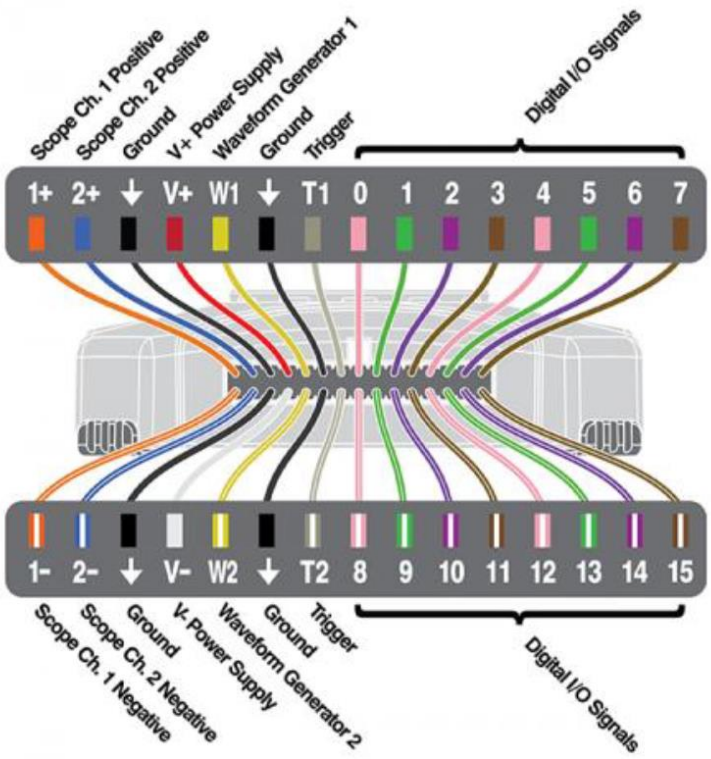
The academic community of students and instructors from Engineering, Science and Polytechnic institutes are the potential stakeholders as well as the beneficiaries. The objective is to provide quality Virtual Lab experiences for education.

The data collected through major universities in India indicates that, there are over 500 Labs which need to be created; (only ~120 labs of the MHRD Virtual Labs project are available for use). Therefore, there exists a huge gap in the syllabus which has to be filled by the community itself.

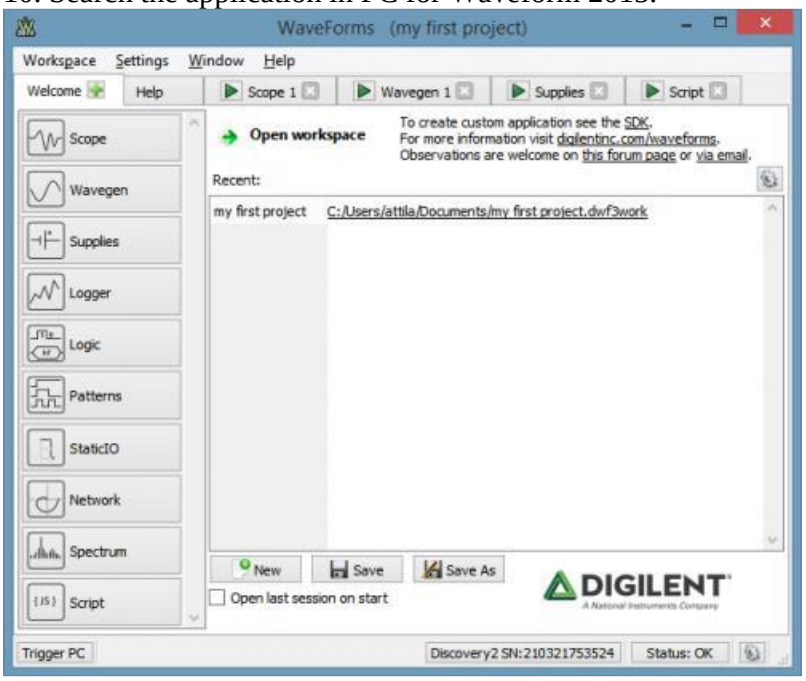
Procedure:

1. Build the Binary to Gray and Vice Versa Circuit in the breadboard.
2. Use the below pin diagram for circuit connection in breadboard.

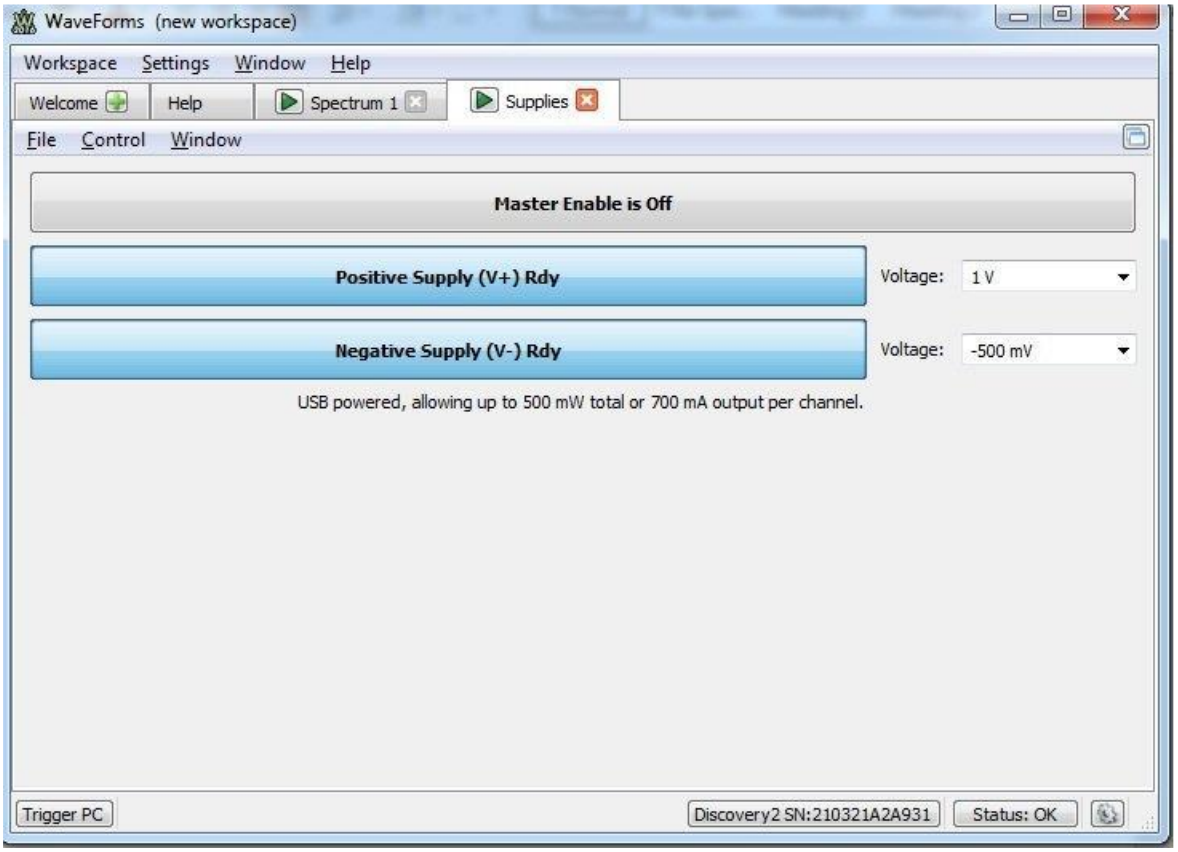




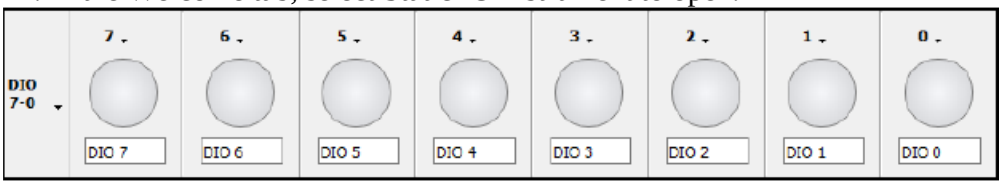
3. Use the above pin connection from AD2.
4. Red wire belongs to power. Take a wire connect to red wire and wire it to pin 14.
5. Black wire belongs to ground. Take a wire connect to Black wire and wire it to pin 7.
6. Use Pin 0- Pin 3 of AD2 as Input.
7. Connect Pin 0- Pin 3 of AD2 to B0, B1, B2, B3.
8. Use Pin 4- Pin 7 of AD2 as Output.
9. Connect Pin 4- Pin 7 of AD2 to G0, G1, G2, G3.
10. Search the application in PC for Waveform 2015.



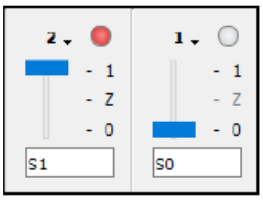
11. In the above window click the Supplies Instrument.



12. Use only positive supply. Change the voltage as 5.
13. Click Master Enable button to enable the Instrument.
14. In the Welcome tab, select Static IO Instrument to open.

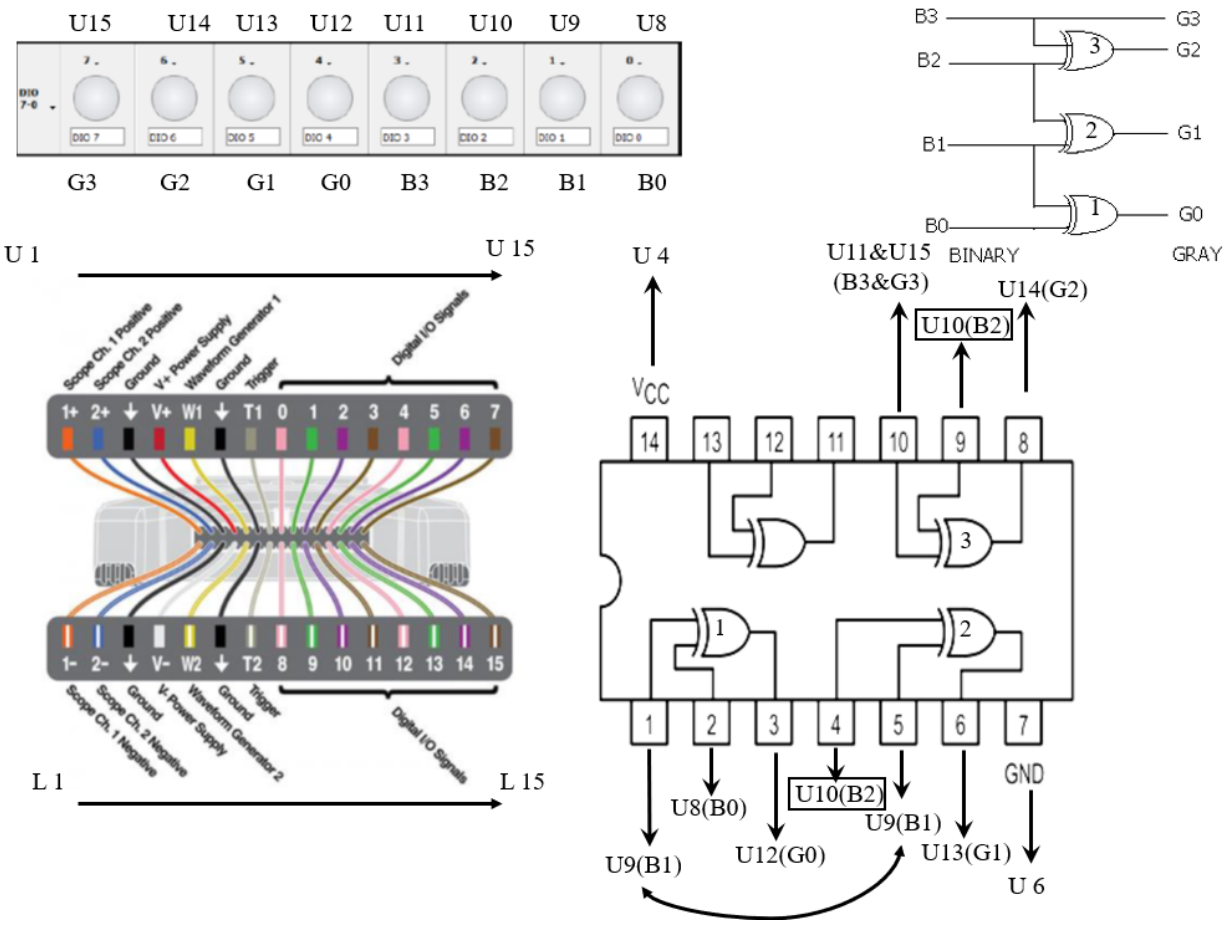


15 Configure Digital I/O signal into a switch by selecting 0, Switch, Push/Pull (1/0) as seen in Figure below for DIO 0-DIO3



16. Run both Static IO and Power Supplies Instrument.
17. Verify the truth table by changing the switch position.

Connection Diagram:



Result:

Thus, design and implementation of Binary to gray code converters and Vice Versa using logic gates using Multisim and NI Analog Discovery 2.

DEPT. of Computer Science Engineering

SRM IST, Ramapuram

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	**07**
Title of Experiment	**Design and implementation of magnitude comparator**
Name of the candidate	Sathya L J K
Register Number	RA2011026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva	5	
2	Circuit Connection and Execution	10	
3	Verification of truth table	5	
	Total	**20**	

Staff Signature with date

7. a. Design and implementation of Magnitude Comparator Combinational circuits using simulation package

Aim:

To Design a magnitude comparator using Multisim software and to verify its truth table.

Apparatus / Software Required:

MULTISIM SOFTWARE

Theory:

A magnitude digital comparator is a combinational circuit that compares two digital or binary numbers in order to find out whether one binary number is equal, less than or greater than the other binary number. We logically design a circuit for which we will have two inputs one for A and other for B and have three output terminals, one for A > B condition, one for A = B condition and one for A < B condition.

Figure-1: Block Diagram of Comparator

2-Bit Magnitude Comparator:

A comparator used to compare two binary numbers each of two bits is called a 2-bit magnitude comparator. It consists of four inputs and three outputs to generate less than, equal to and greater than between two binary numbers.

Truth Table:

The truth table for a 2-bit comparator is given below:

INPUT				OUTPUT		
A1	A0	B1	B0	A<B	A=B	A>B
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

Figure-2: Truth Table of 2-Bit Comparator

The logical expressions for each output can be expressed as follows:

$A > B : A_1B_1' + A_0B_1'B_0' + A_1A_0B_0'$

$A = B : A_1'A_0'B_1'B_0' + A_1'A_0B_1'B_0 + A_1A_0B_1B_0 + A_1A_0'B_1B_0'$

$: A_1'B_1' (A_0'B_0' + A_0B_0) + A_1B_1 (A_0B_0 + A_0'B_0')$

$: (A_0B_0 + A_0'B_0') (A_1B_1 + A_1'B_1')$

$: (A_0 \text{ Ex-Nor } B_0) (A_1 \text{ Ex-Nor } B_1)$

$A < B : A_1'B_1 + A_0'B_1B_0 + A_1'A0'B0$

Logical Diagram:

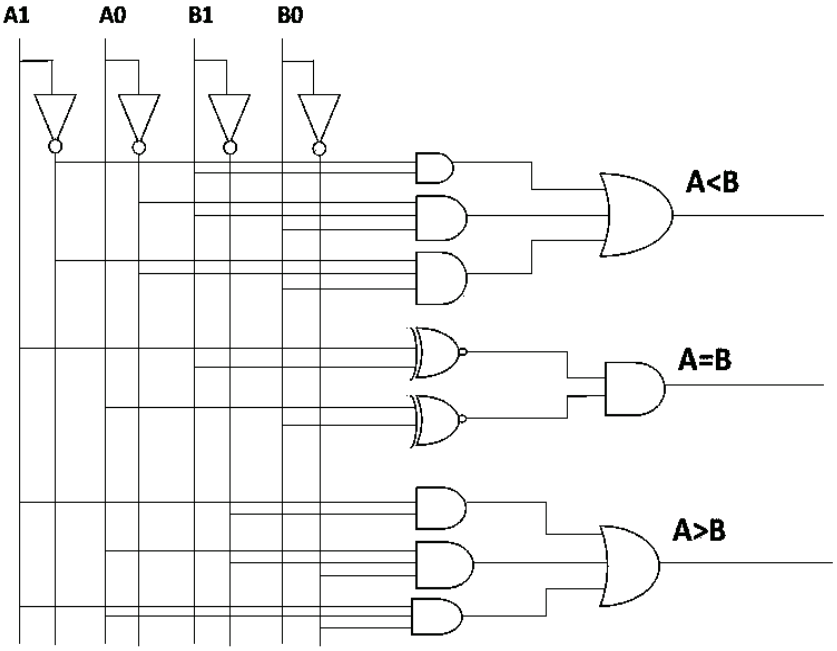


Figure-3: Logic Circuit of 2-Bit Magnitude Comparator

Multisim Diagram:

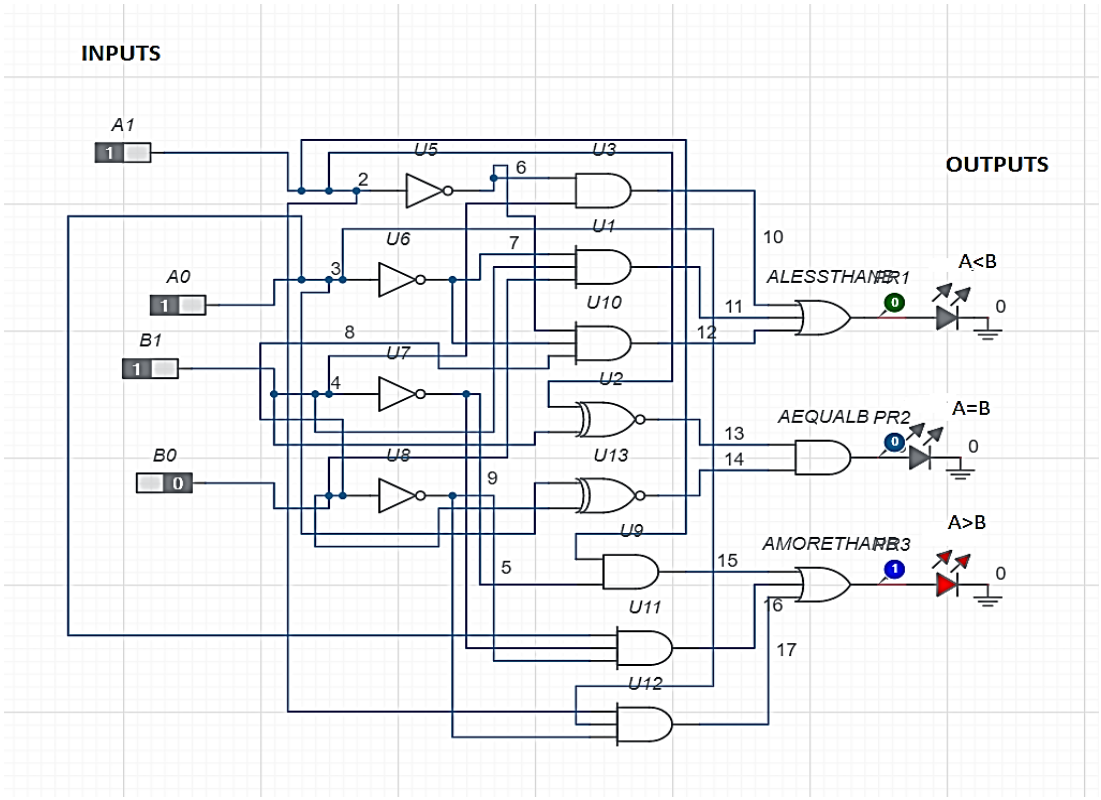
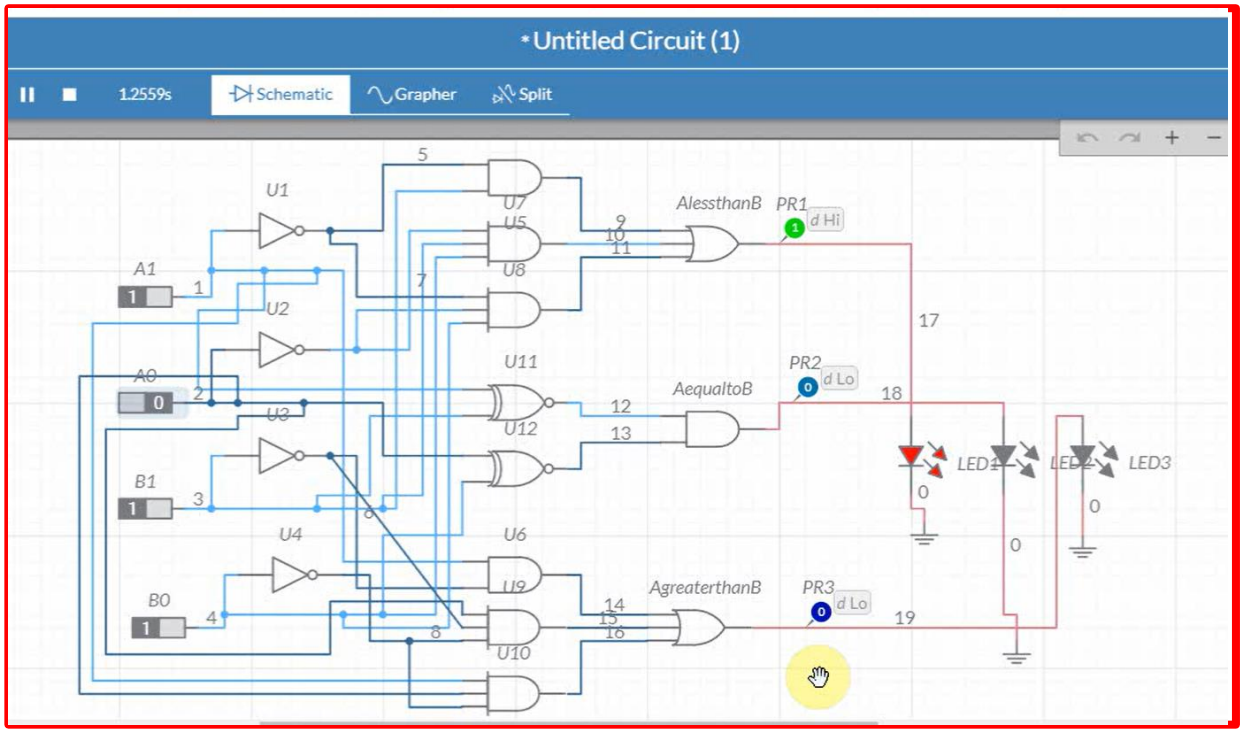
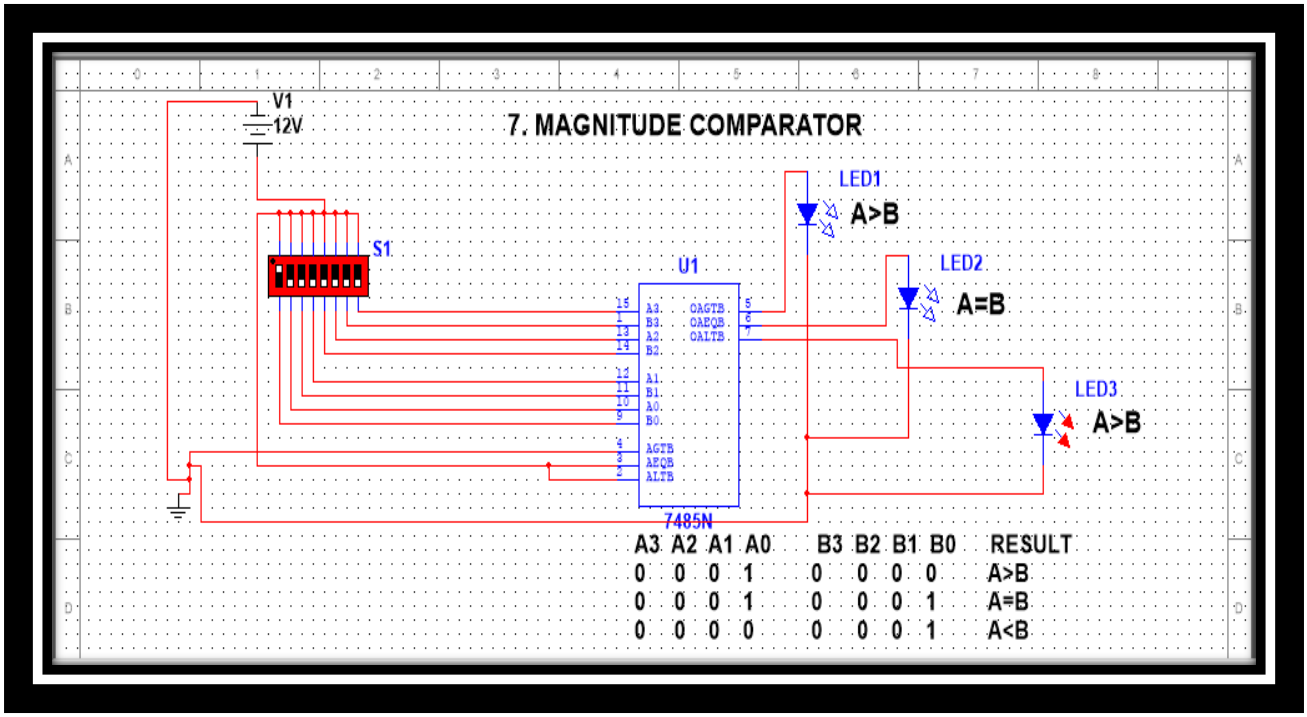


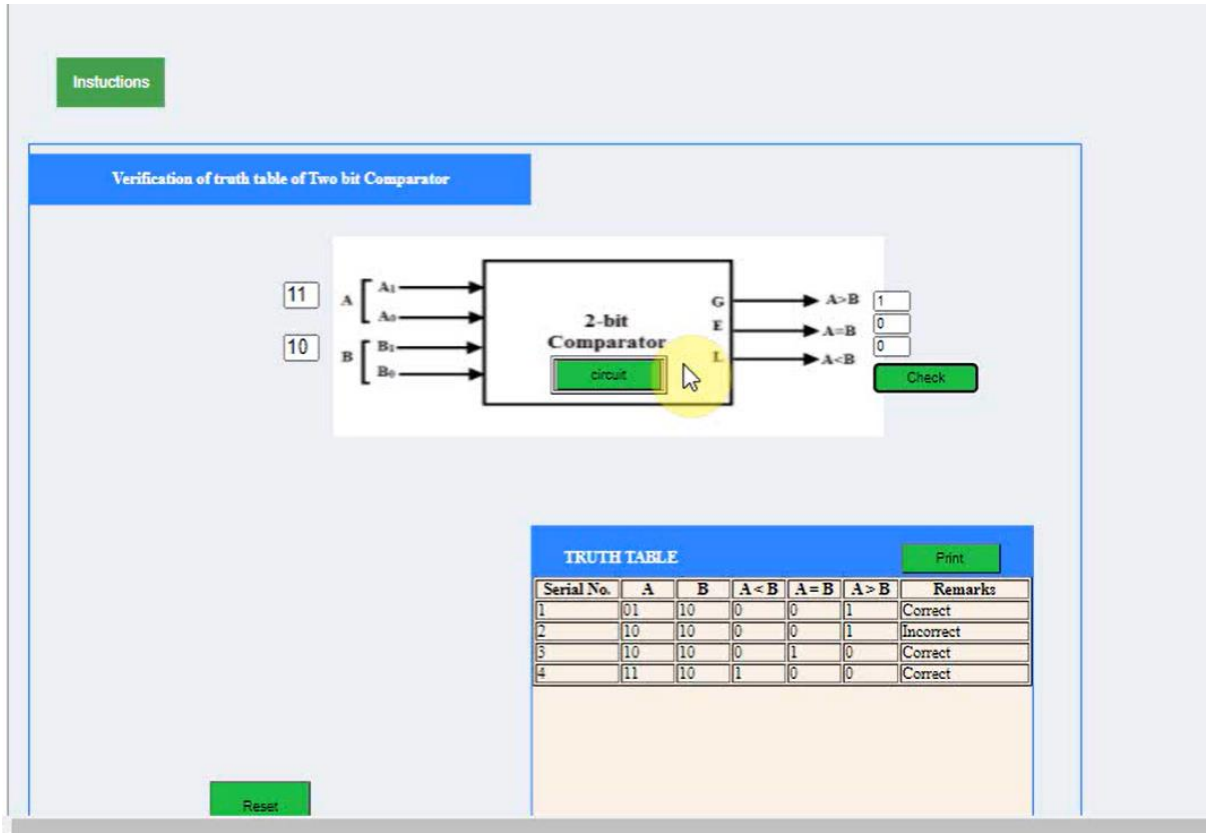
Figure-4: Multisim Circuit of 2-Bit Magnitude Comparator

Simulation diagram:



Output:





Serial No.	A	B	A < B	A = B	A > B	Remarks
1	01	10	0	0	1	Correct
2	10	10	0	0	1	Incorrect
3	10	10	0	1	0	Correct
4	11	10	1	0	0	Correct

Result:

Thus the 2 bit magnitude comparator was designed and verified with the truth table using Multisim software.

Expt No: 7b

Date:

Hardware Implementation Using NI Analog Discovery 2

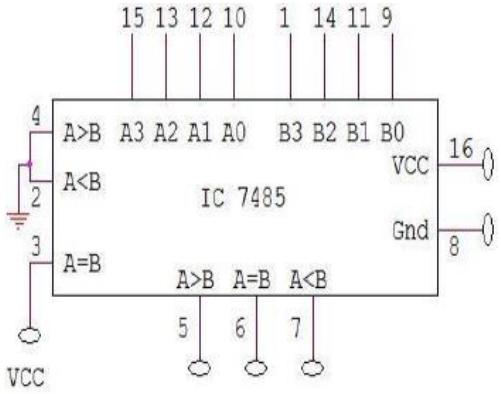
Aim:

Hardware Implementation of the same with NI Analog Discovery 2.

Apparatus Required:

S.No	Apparatus	Types	Range	Quantity
1	IC	IC 7486		
2	NI Analog Discovery 2			
3	Wires			As Required
4	Bread board			1

Circuit Connection:



Truth Table:

A				B				Result
A3	A2	A1	A0	B3	B2	B1	B0	
0	0	0	1	0	0	0	0	A > B
0	0	0	1	0	0	0	1	A = B
0	0	0	0	0	0	0	1	A < B

Procedure:

1. Fix the IC 7485 in the breadboard.
2. Red wire belongs to power. Take a wire connect to red wire and wire it to pin 16.
3. Black wire belongs to ground. Take a wire connect to Black wire and wire it to pin 8.
4. Short pins 4,2 of the IC and wire it to ground.
5. Interconnect pin 3 and 16.

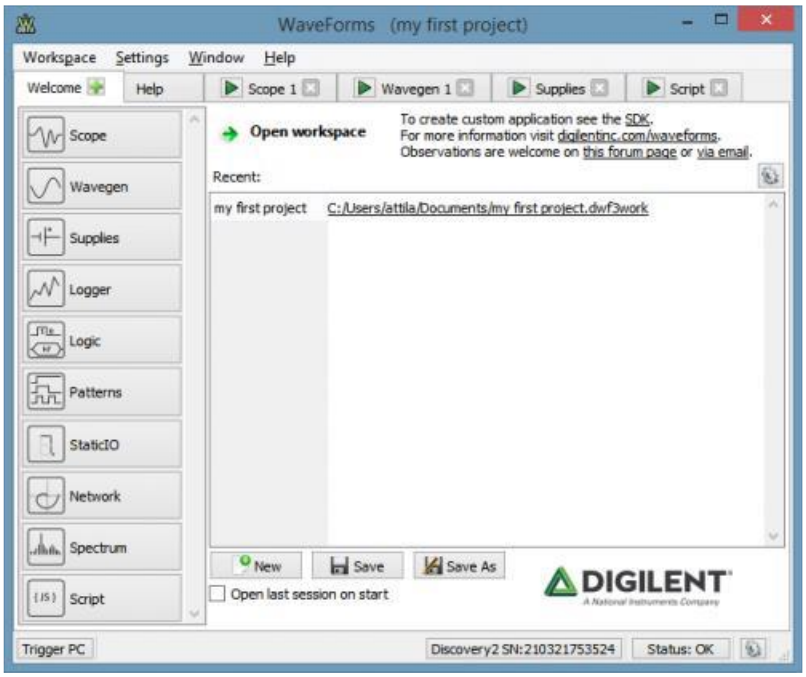
6. Use Pin 0 - Pin 7 of AD2 as Input.

7. Connect Pin 0 - Pin 7 of AD2 to IC Pin 15,13,12,10,1,14,11,9 (A3, A2, A1, A0, B3, B2, B1, B0) respectively.

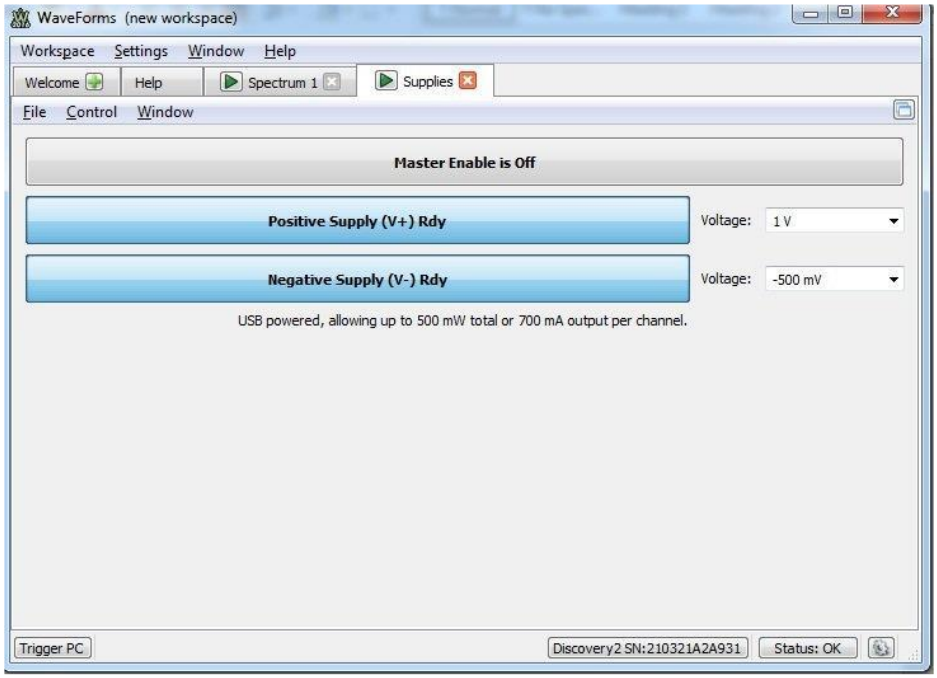
8. Use Pin 13- Pin 15 of AD2 as Output.

9. Connect Pin13- Pin15 of AD2 to IC pin 5,6,7 respectively.

10. Search the application in PC for Waveform 2015.



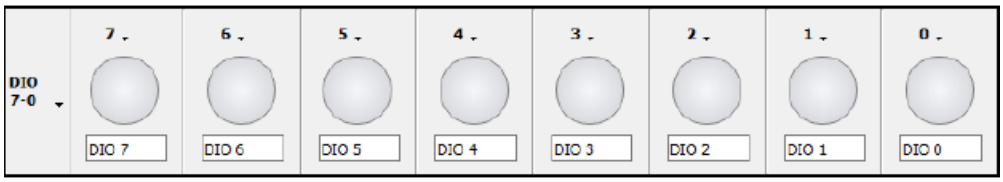
11. In the above window click the Supplies Instrument.



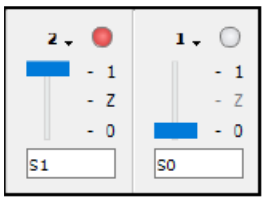
12. Use only positive supply. Change the voltage as 5.

13. Click Master Enable button to enable the Instrument.

14. In the Welcome tab, select Static IO Instrument to open.



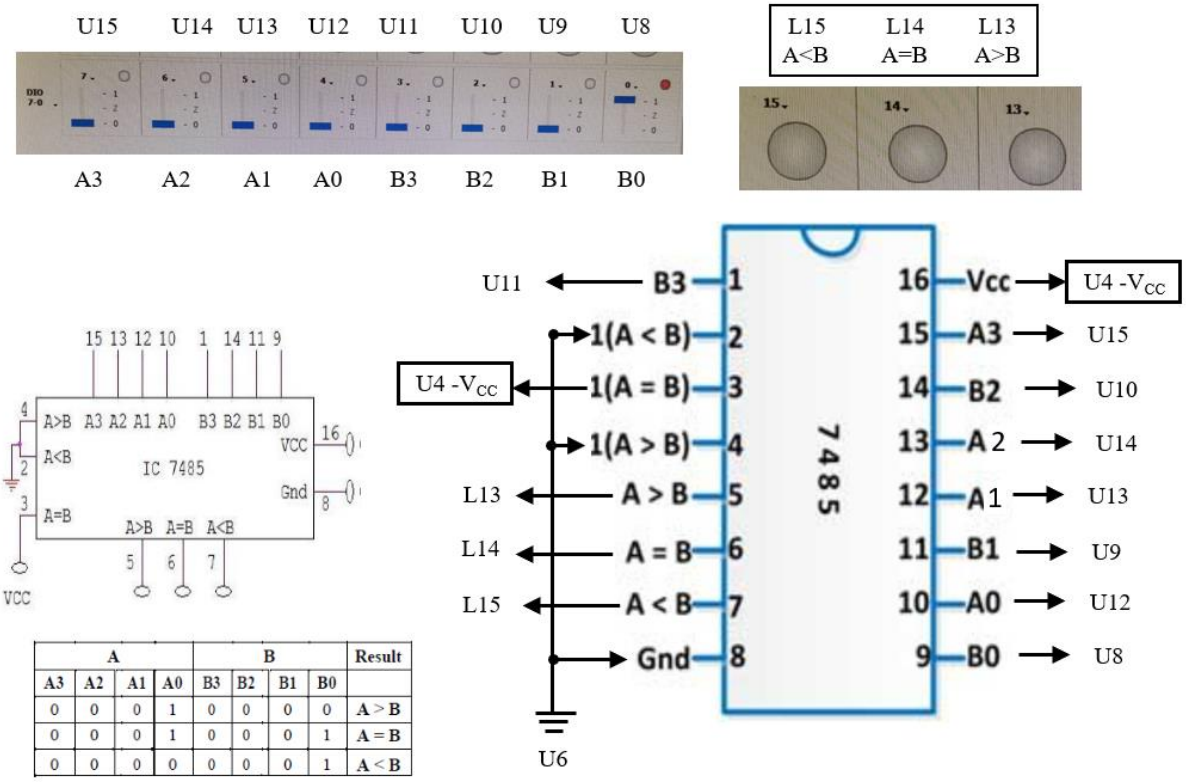
15. Configure Digital I/O signal into a switch by selecting 0-7 Switch to Push/Pull (1/0) as seen in Figure below for DIO 0-DIO7



16. Run both Static IO and Power Supplies Instrument.

17. Verify the truth table by changing the switch position.

Connection Diagram:



A				B				Result
A3	A2	A1	A0	B3	B2	B1	B0	
0	0	0	1	0	0	0	0	A > B
0	0	0	1	0	0	0	1	A = B
0	0	0	0	0	0	0	1	A < B

Result:

Thus, design and implementation of Magnitude Comparator using Multisim and NI Analog Discovery 2 is done.

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

Department of Computer Science Engineering

SRMIST, Ramapuram

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	08
Title of Experiment	Design and implementation of Synchronous sequential circuits using Simulation Package
Name of the candidate	Sathya L J K
Register Number	RA2011026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva	5	
2	Execution	10	
3	Model Calculation / Result Analysis	5	
	Total	**20**	

Staff Signature with date

Experiment No:8 Date:

Design and implementation of Synchronous sequential circuits using Simulation Package

Aim:

To design and implementation of D Flip Flop using Multisim.

Apparatus Required:

S.No	Apparatus	Type	Range	Quantity
1)	IC	IC 7474		1
2)	LED			4
3)	Switch			4
4)	DC Power Source			1
5)	Digital Clock			1

Software Required:

https://www.multisim.com/

Theory:

A D-type flip-flop is a clocked flip-flop which has two stable states. A D-type flip-flop operates with a delay in input by one clock cycle. Thus, by cascading many D-type flip-flops delay circuits can be created, which are used in many applications such as in digital television systems.

A D-type flip-flop is also known as a D flip-flop or delay flip-flop. A D-type flip-flop consists of four inputs:

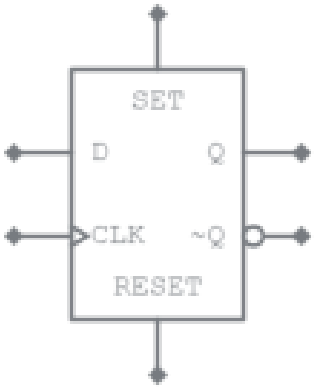
- Data input
- Clock input
- Set input
- Reset input

It also has two outputs, with one being logically inverse of other. The data input is either logic 0 or 1, meaning low or high voltage. The clock input helps in synchronizing the circuit to an external signal. The set input and reset input are mostly held low. A D-type flip-flop can have two possible values. When input D = 0, the flip-flop undergoes a reset, which means the output would be set to 0. When input D = 1, the flip-flop does a set, which makes the output There are several applications in which a D-type flip-flop is used, such as in frequency dividers and data latches.

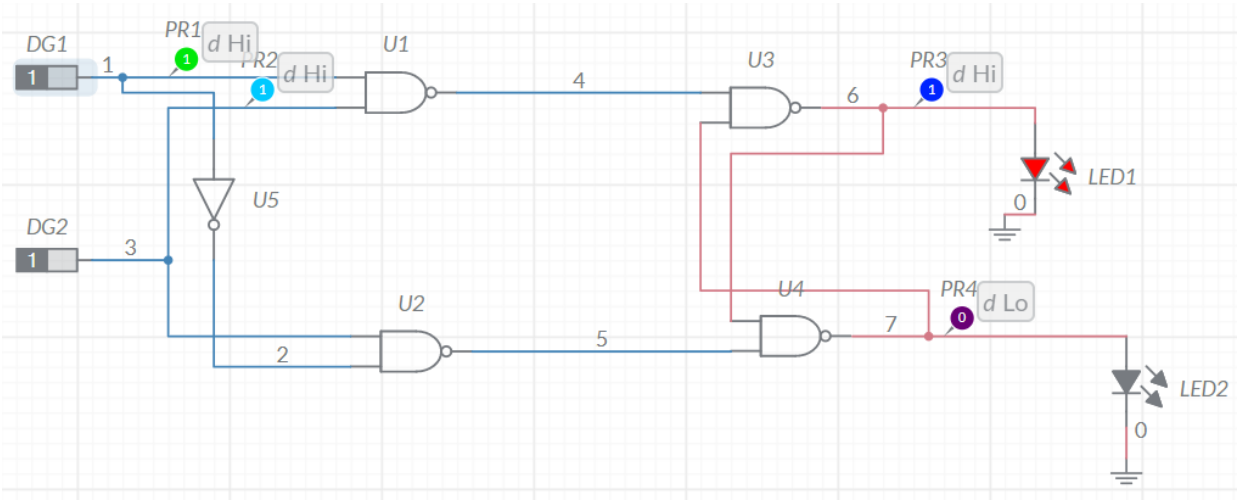
Procedure:

1. Log in Multisim Live Online Circuit Simulator.
2. Click create circuit button.
3. Click search for component and type components. Select it and drag to the Schematic window
4. Select the entire apparatus given in table to complete the circuit.
5. Click schematic connector and select junction drag to the Schematic window and left click at the point and drag to the other point to make the wire connection. Complete the connection according to the diagram.
6. Click analysis and annotation and select digital probe and drag to the schematic window and place at the output side.
7. Save the file by clicking the file navigation menu at the left top and save with a file name.
8. Run the simulation change the value of the switches to verify the truth table.

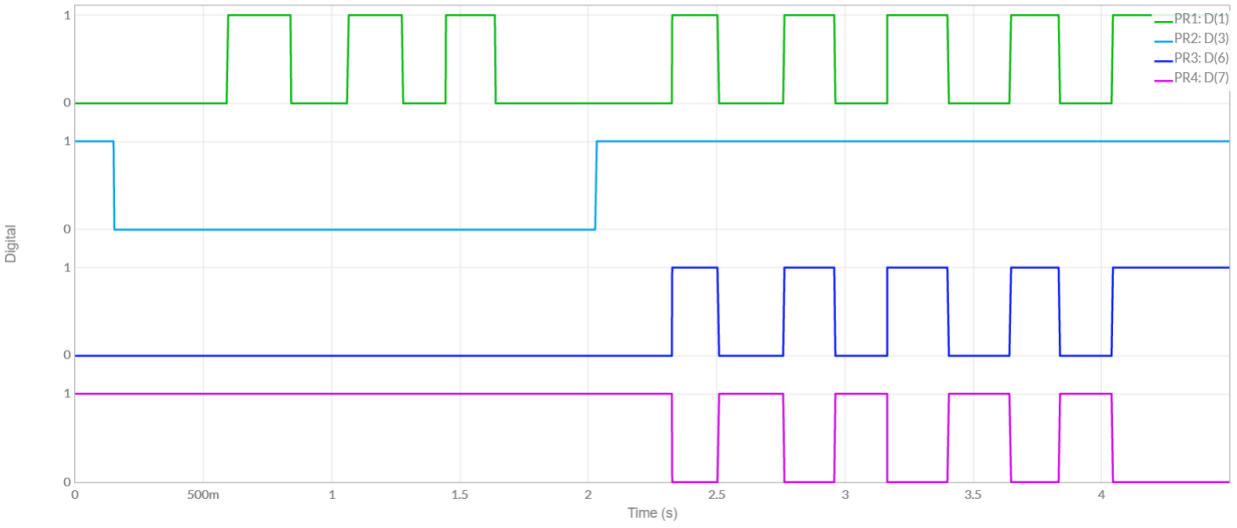
Pin Diagram:



Circuit Diagram:



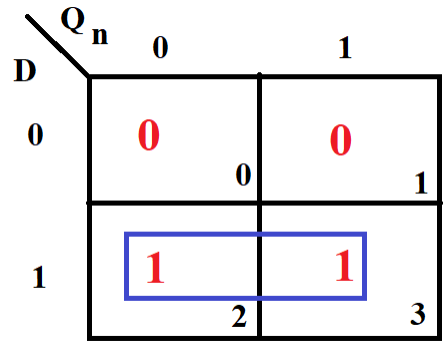
Timing Diagram:



Truth (Characteristic) Table:

CLK	D	Q_n	Q_{n+1}	Q'_{n+1}
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	1	0
1	1	1	1	0

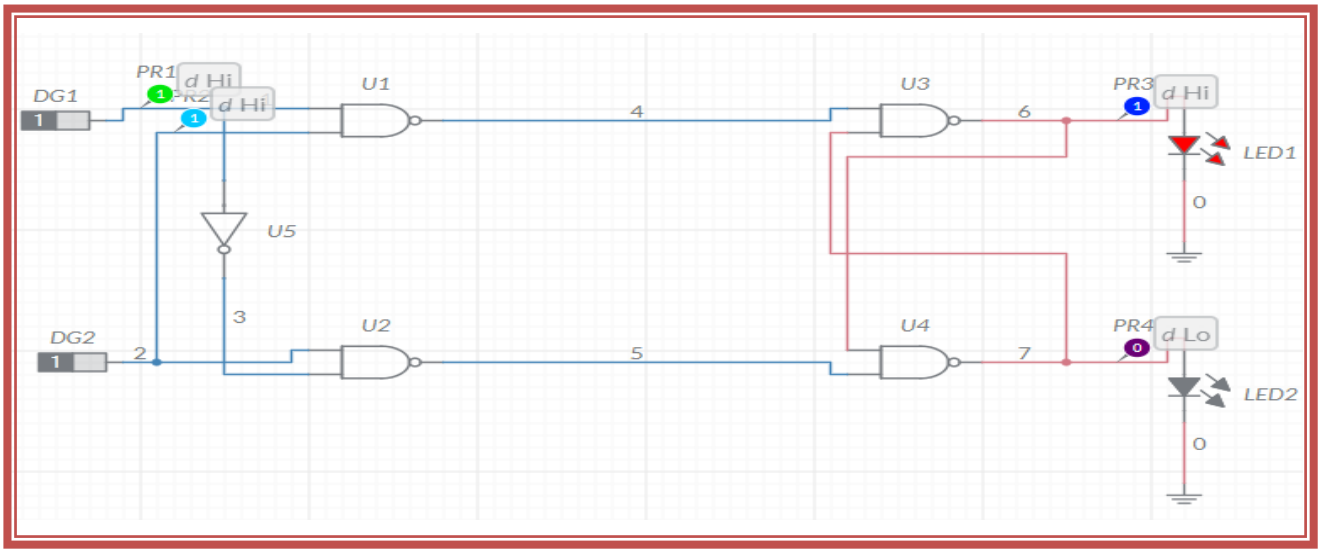
Characteristic Equation:



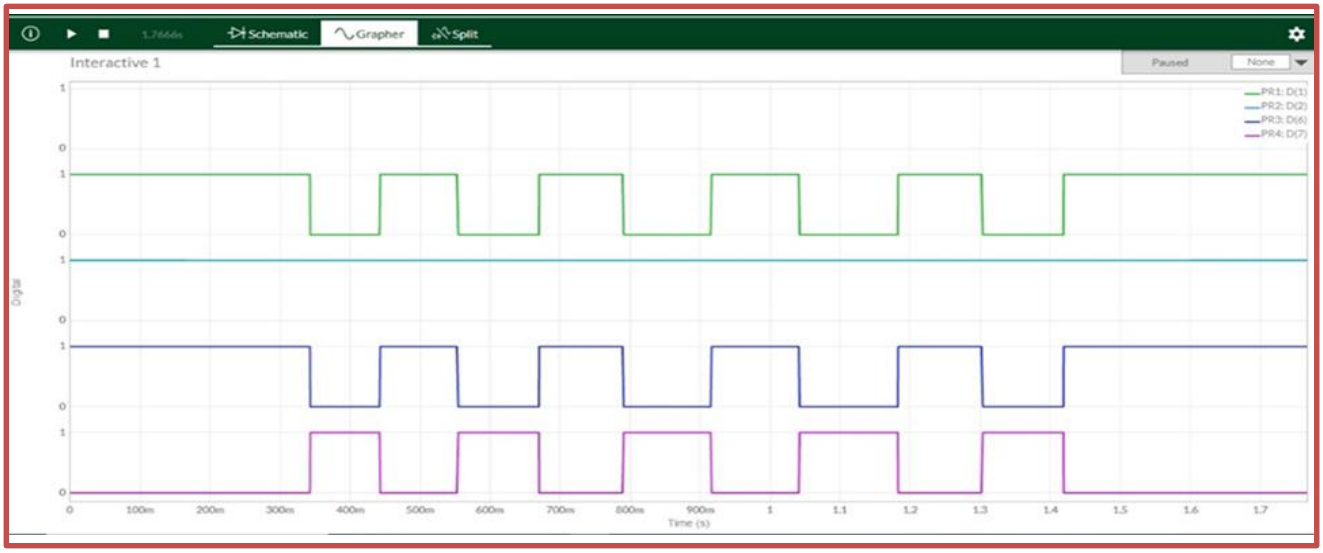
$Q_{n+1} = D$

Simulation Results:

Circuit Diagram:



Timing Diagram:



Result:

Thus, the implementation of D flip flop using Multisim is verified.

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

Department of Computer Science Engineering

SRMIST, Ramapuram

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	09
Title of Experiment	Implementation of SISO, SIPO, PISO and PIPO shift registers using Flip Flops
Name of the candidate	Sathya L J K
Register Number	RA2011026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva / Online Quiz	5	
2	Execution	10	
3	Model Calculation / Result Analysis	5	
	Total	**20**	

Staff Signature with date

Experiment No: 9 Date:

Implementation of SISO, SIPO, PISO and PIPO shift registers using Flip Flop

Aim:

To Implement SISO, SIPO, PISO and PIPO shift registers using Flip Flop.

Apparatus Required:

S.No	Apparatus	Type	Range	Quantity
1)	D Flip Flop			4
2)	LED			4
3)	Switch			4
4)	DC Power Source			1
5)	Digital Clock			1

Software Required:

https://www.multisim.com/

Theory:

A D-type flip-flop is a clocked flip-flop which has two stable states. A D-type flip-flop operates with a delay in input by one clock cycle. Thus, by cascading many D-type flip-flops delay circuits can be created, which are used in many applications such as in digital television systems.

A D-type flip-flop is also known as a D flip-flop or delay flip-flop. A D-type flip-flop consists of four inputs:

- Data input
- Clock input
- Set input
- Reset input

It also has two outputs, with one being logically inverse of other. The data input is either logic 0 or 1, meaning low or high voltage. The clock input helps in synchronizing the circuit to an external signal. The set input and reset input are mostly held low. A D-type flip-flop can have two possible values. When input D = 0, the flip-flop undergoes a reset, which means the output would be set to 0. When input D = 1, the flip-flop does a set, which makes the output There are several applications in which a D-type flip-flop is used, such as in frequency dividers and data latches.

Serial-in to Serial-out (SISO) Shift Register

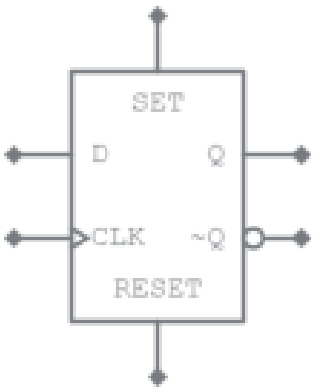
This **shift register** is very similar to the SIPO above, except were before the data was read directly in a parallel form from the outputs Q_A to Q_D, this time the data is allowed to flow straight through the register and out of the other end. Since there is only one output, the DATA leaves the shift register one bit at a time in a serial pattern, hence the name **Serial-in to Serial-Out Shift Register** or **SISO**.

The SISO shift register is one of the simplest of the four configurations as it has only three connections, the serial input (SI) which determines what enters the left hand flip-flop, the serial output (SO) which is taken from the output of the right hand flip-flop and the sequencing clock signal (Clk). The logic circuit diagram below shows a generalized serial-in serial-out shift register.

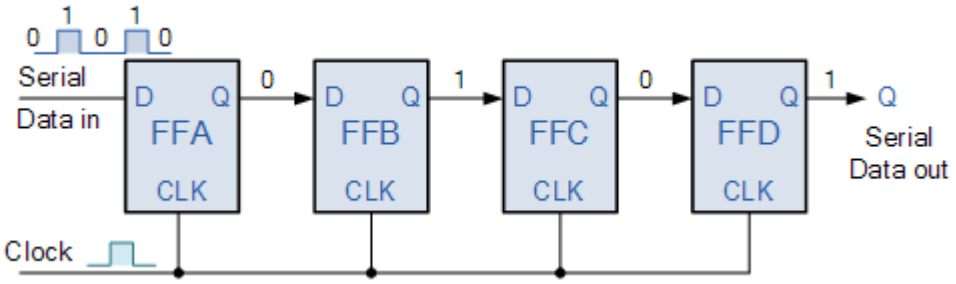
SISO PROCEDURE

1. Log in Multisim Live Online Circuit Simulator.
2. Built-in D Flip Flop is available in the below link.
   https://www.multisim.com/content/TU4wEJ8VgRsRNyxeecQ3Wd/d-flip-flop/open/
3. Connect four D flip flops in cascaded form.
4. Connect LED at the output 'Q' terminal of fourth flip flop.
5. Connect common ground for the 'Reset' pin for four D flip flops and leave the 'set' pin of all four D flip flops floating.
6. Give input to first flip flop using Digital Switch (Select Digital Switch Digital components list) , For Clock signal generation use digital switch itself for all four flip flops.
7. Observe the movement of input from one flip flop to another by Switching on and off the clock digital switch, if input is set as '1', after four clock pulses you can observe LED at the fourth flip flop will be 'ON' this shows the movement of logic '1' from first flip flop to fourth flip flop.
8. Save the file by clicking the file navigation menu at the left top and save with a file name.
9. Run the simulation change the value of the switches to verify the truth table.

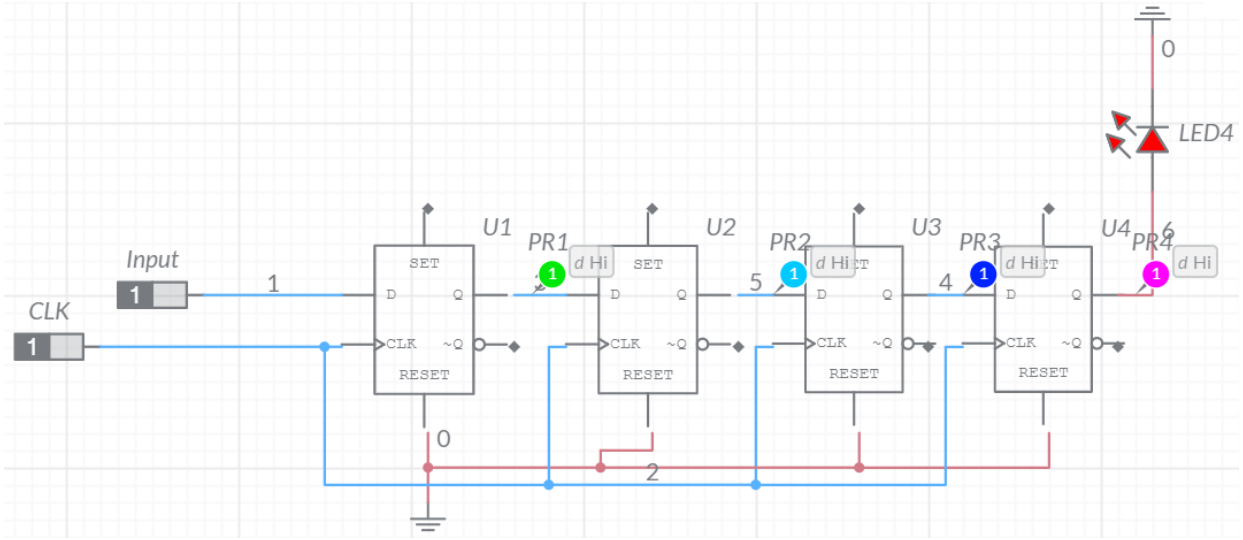
Pin Diagram:

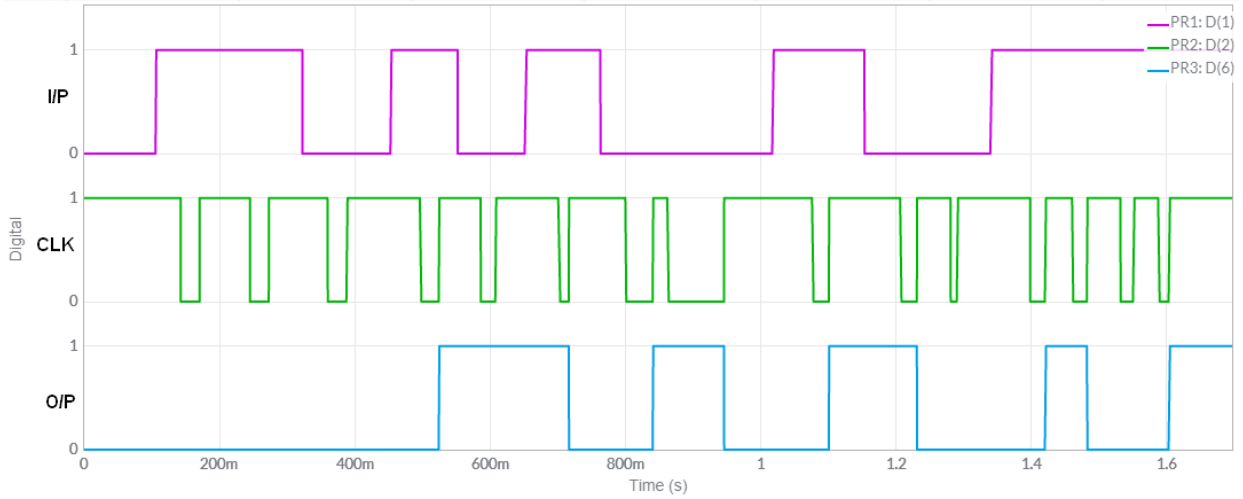


4-bit Serial-in to Serial-out Shift Register

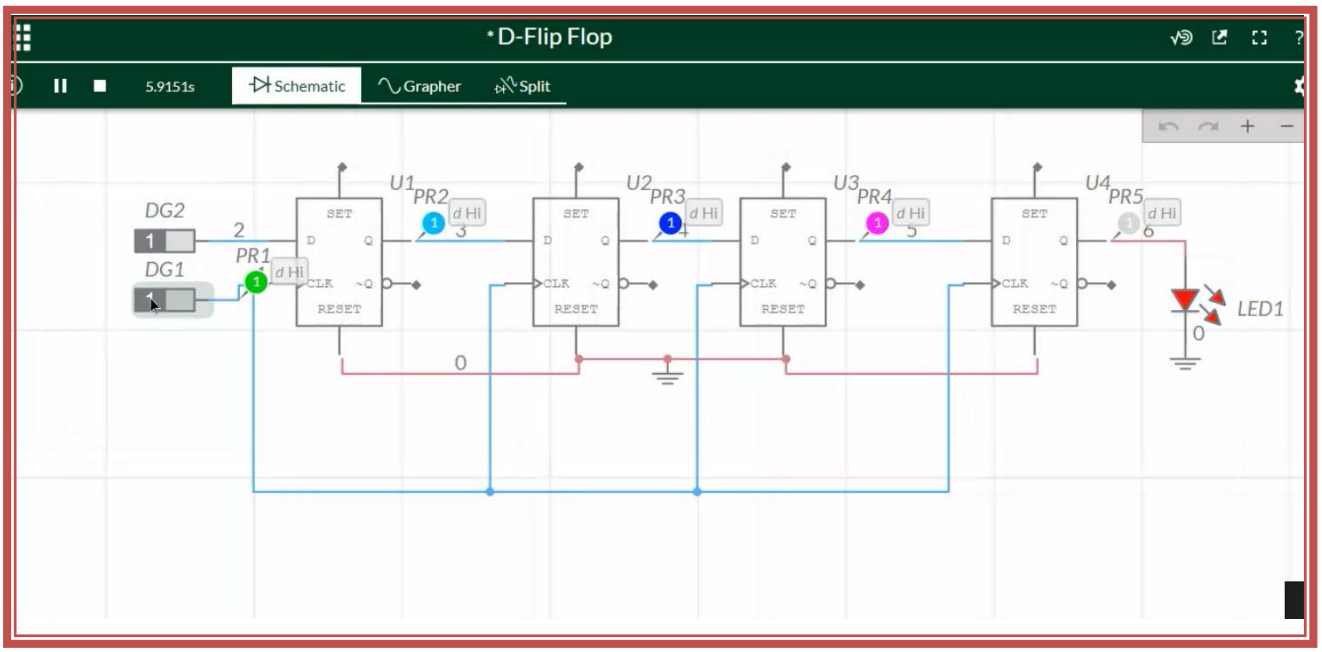


SISO CIRCUIT DIAGRAM in Multisim:

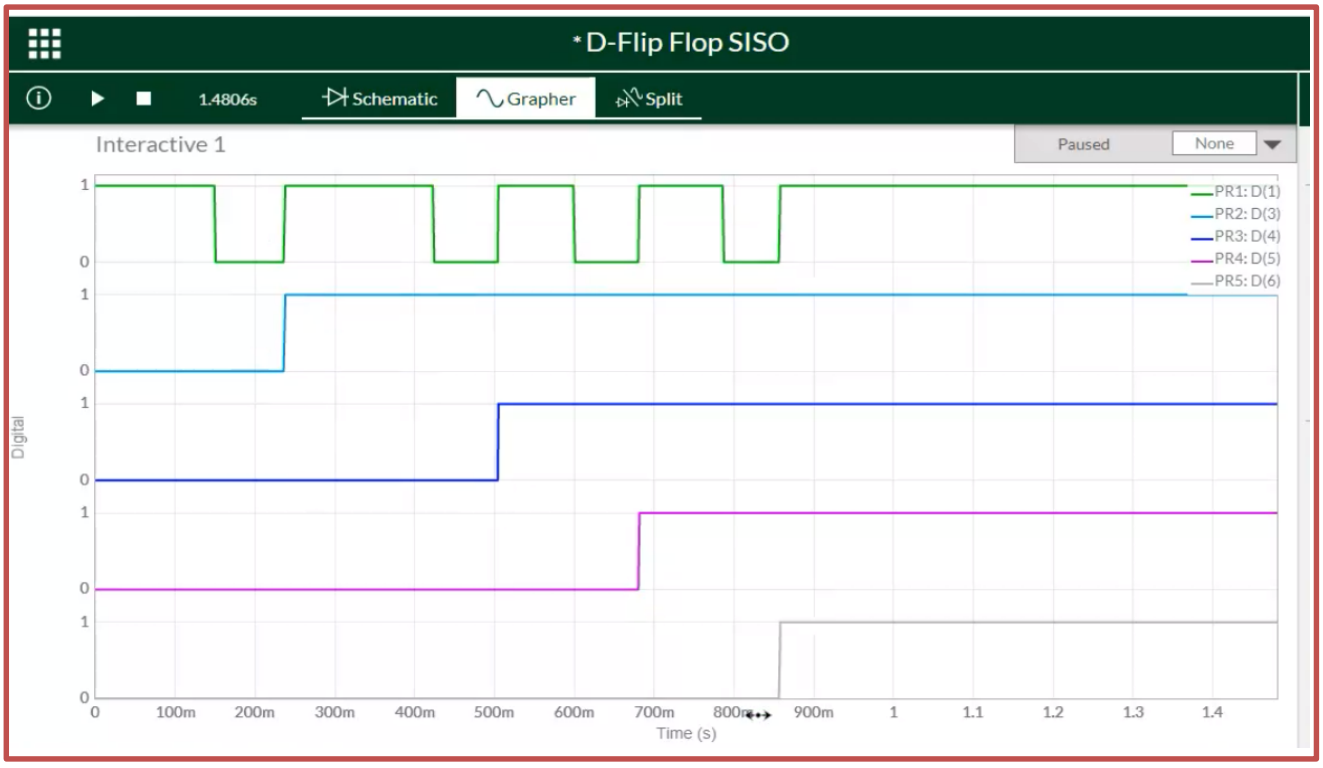




Simulation Results:
Circuit Diagram:

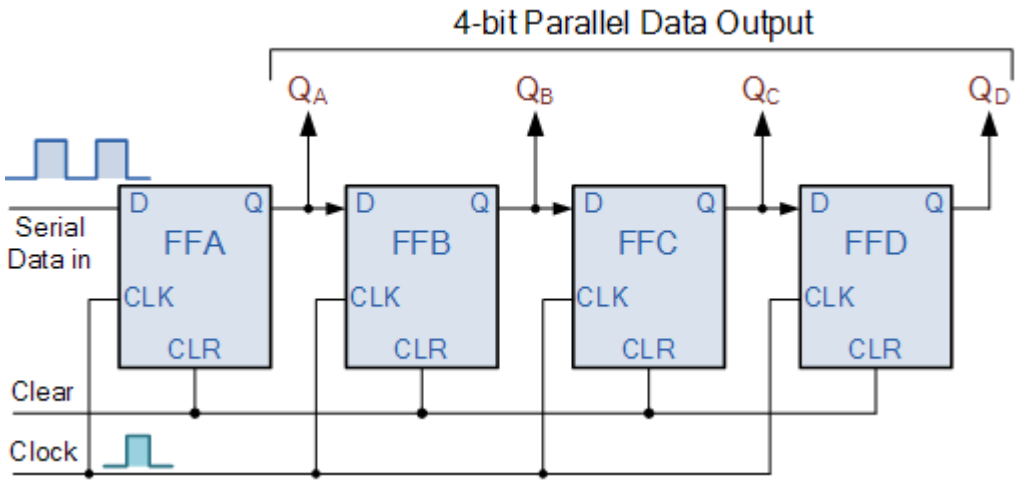


Timing Diagram:



Serial-in to Parallel-out (SIPO) Shift Register

4-bit Serial-in to Parallel-out Shift Register



Theory:

The operation is as follows. Lets assume that all the flip-flops (FFA to FFD) have just been RESET (CLEAR input) and that all the outputs Q_A to Q_D are at logic level “0” ie, no parallel data output.

If a logic “1” is connected to the DATA input pin of FFA then on the first clock pulse the output of FFA and therefore the resulting Q_A will be set HIGH to logic “1” with all the other outputs still remaining LOW at logic “0”. Assume now that the DATA input pin of FFA has returned LOW again to logic “0” giving us one data pulse or 0-1-0.

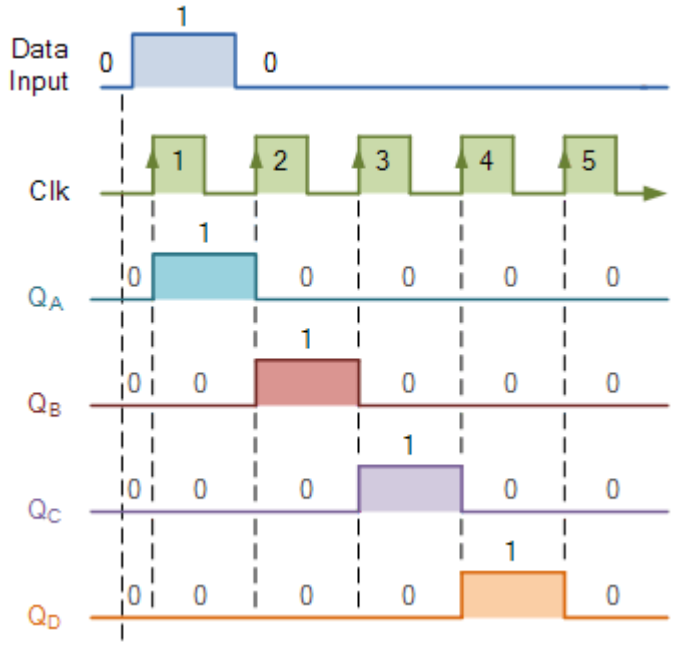
The second clock pulse will change the output of FFA to logic “0” and the output of FFB and Q_B HIGH to logic “1” as its input D has the logic “1” level on it from Q_A. The logic “1” has now moved or been “shifted” one place along the register to the right as it is now at Q_A.

When the third clock pulse arrives this logic “1” value moves to the output of FFC (Q_C) and so on until the arrival of the fifth clock pulse which sets all the outputs Q_A to Q_D back again to logic level “0” because the input to FFA has remained constant at logic level “0”.

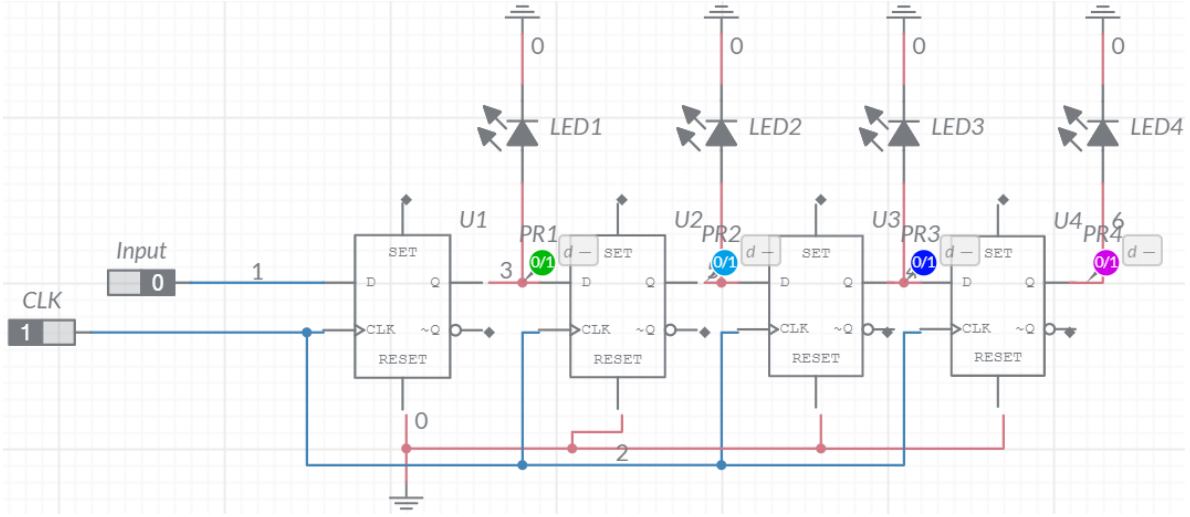
The effect of each clock pulse is to shift the data contents of each stage one place to the right, and this is shown in the following table until the complete data value of 0-0-0-1 is stored in the register. This data value can now be read directly from the outputs of Q_A to Q_D.

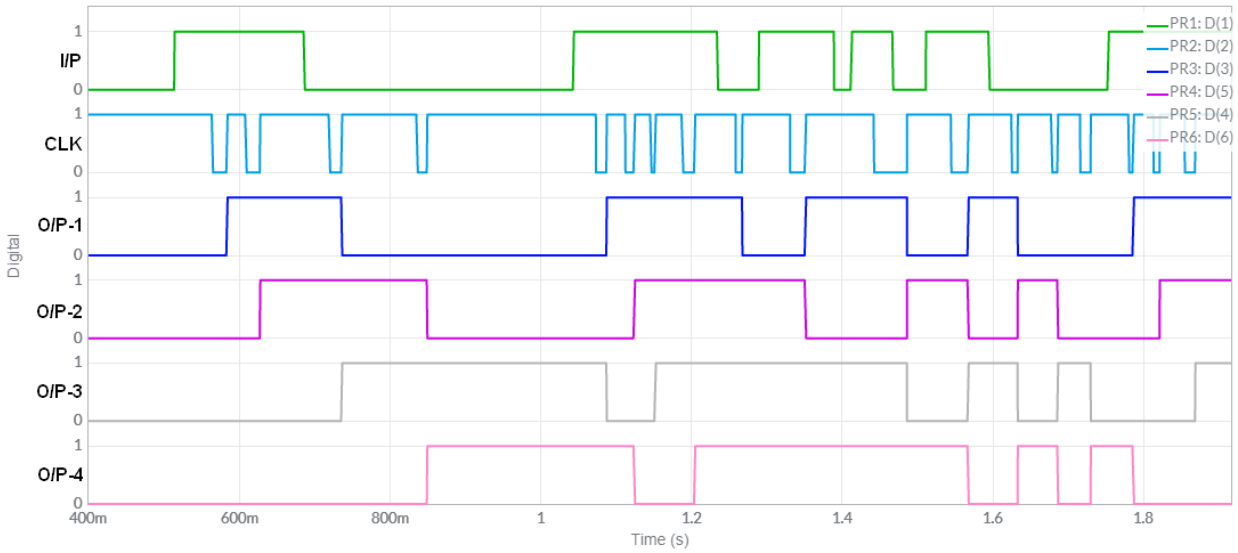
Then the data has been converted from a serial data input signal to a parallel data output. The truth table and following waveforms show the propagation of the logic “1” through the register from left to right as follows.

Data Movement

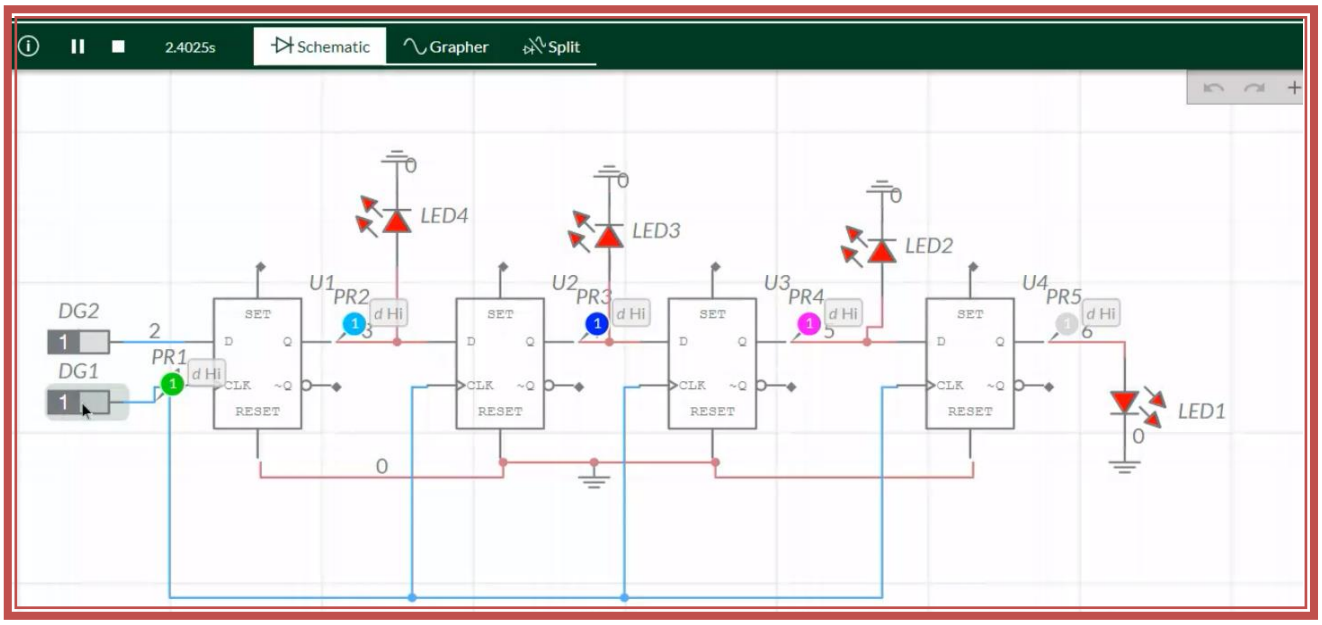


SIPO CIRCUIT DIAGRAM in Multisim:

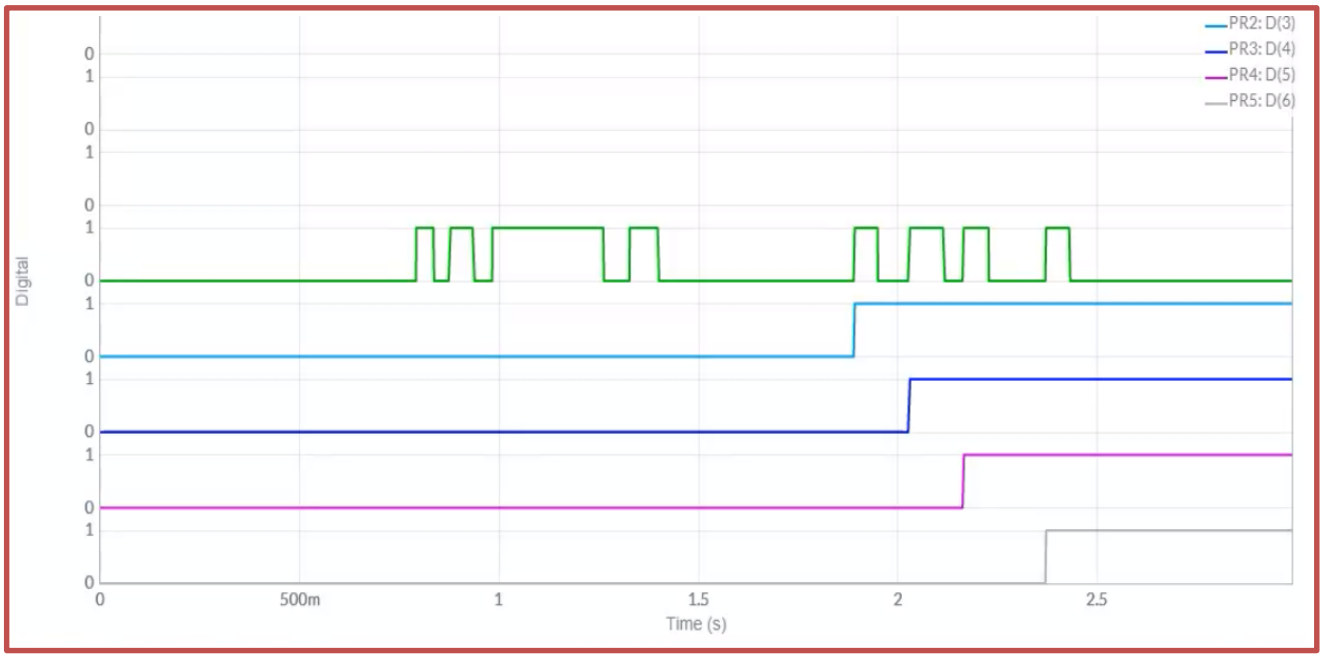




Simulation Result:
Circuit Diagram:



Timing Diagram:

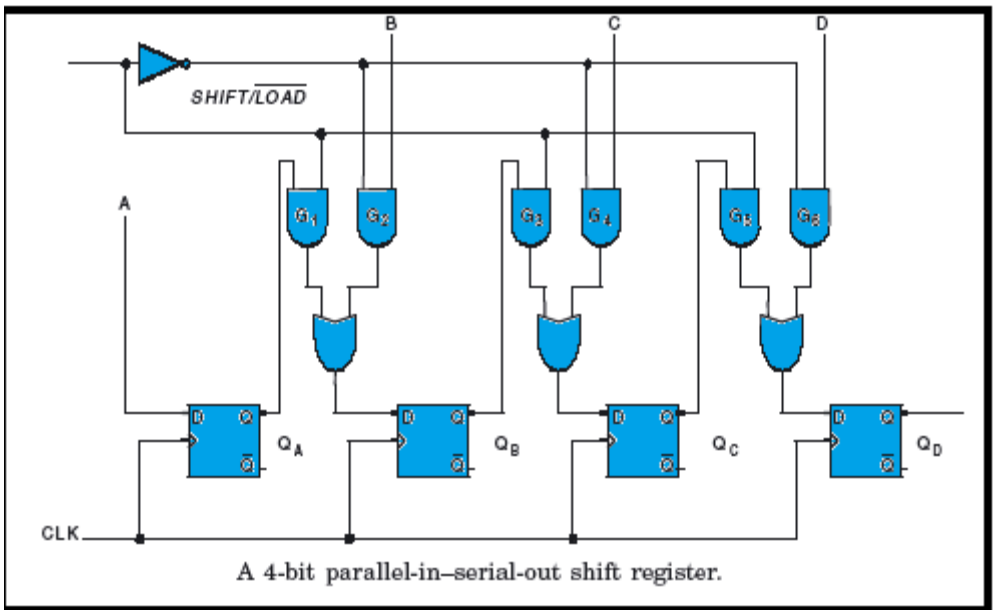


Parallel-in to Serial-out (PISO) Shift Register

The Parallel-in to Serial-out shift register acts in the opposite way to the serial-in to parallel-out one above. The data is loaded into the register in a parallel format in which all the data bits enter their inputs simultaneously, to the parallel input pins P_A to P_D of the register. The data is then read out sequentially in the normal shift-right mode from the register at Q representing the data present at P_A to P_D.

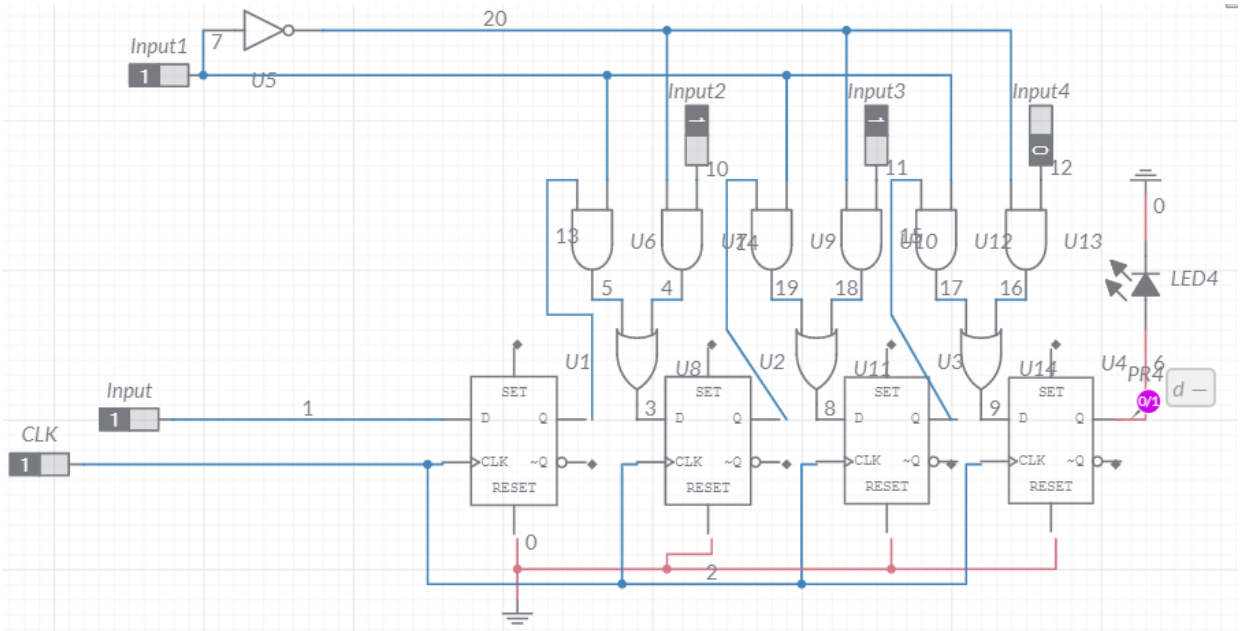
This data is outputted one bit at a time on each clock cycle in a serial format. It is important to note that with this type of data register a clock pulse is not required to parallel load the register as it is already present, but four clock pulses are required to unload the data.

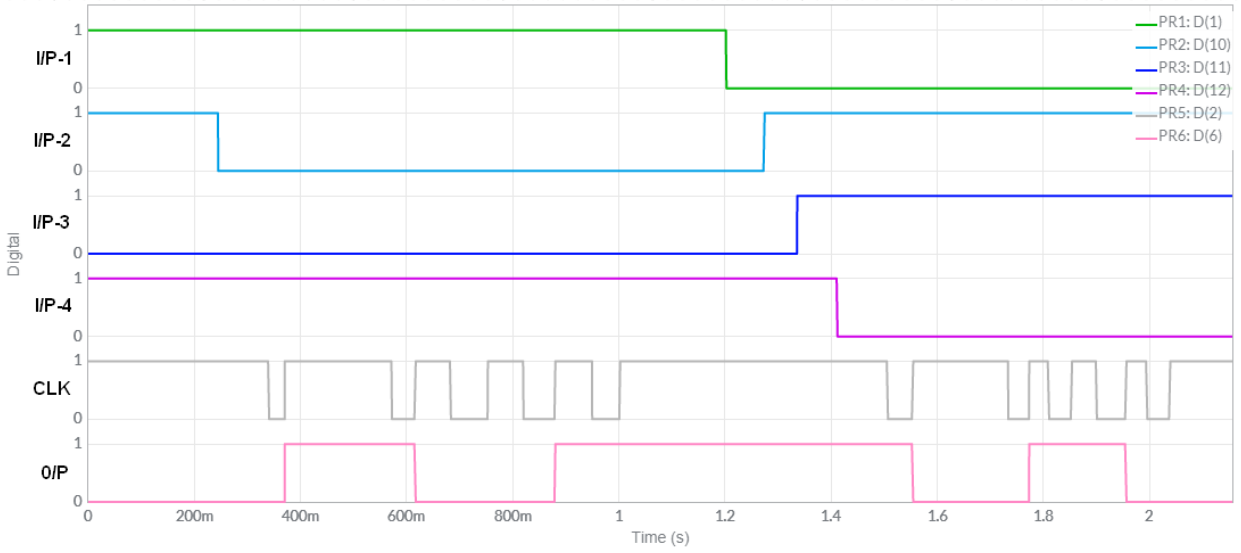
4-bit Parallel-in to Serial-out Shift Register



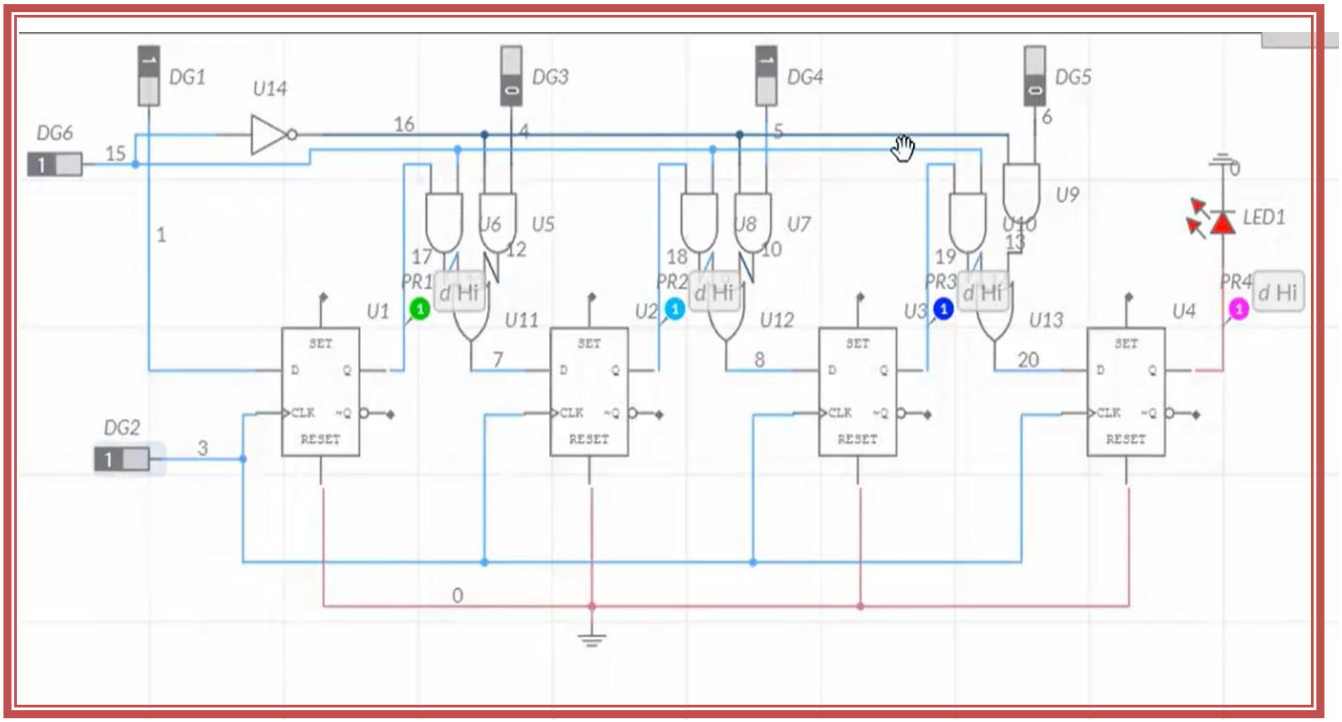
A 4-bit parallel-in–serial-out shift register.

PISO CIRCUIT DIAGRAM in Multisim:

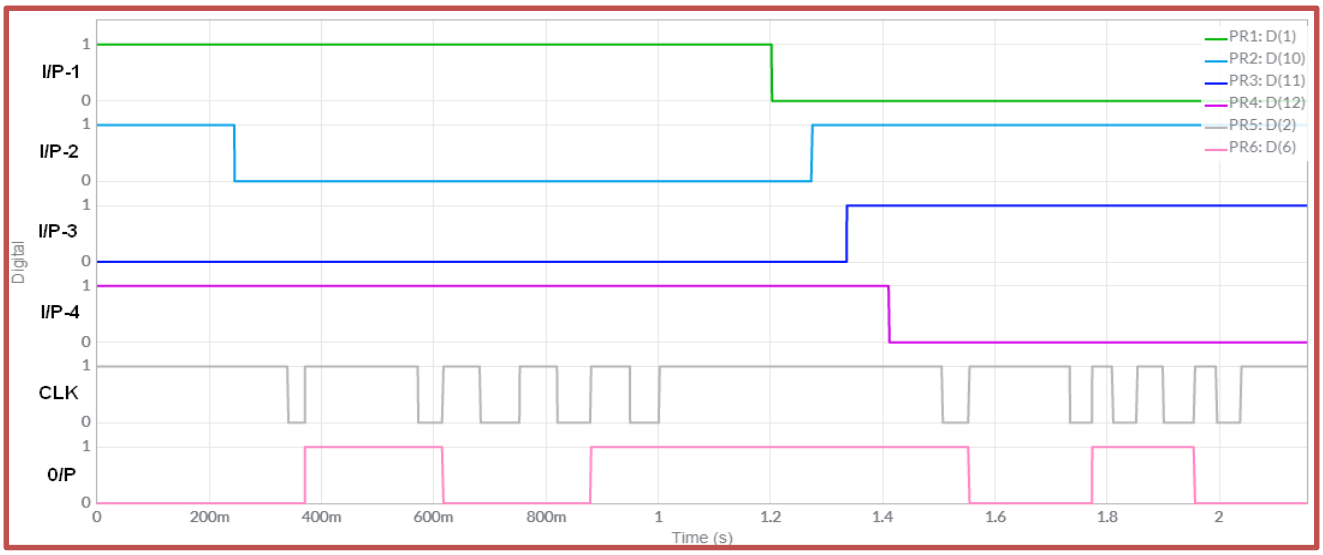




Simulation Results:
Circuit Diagram:



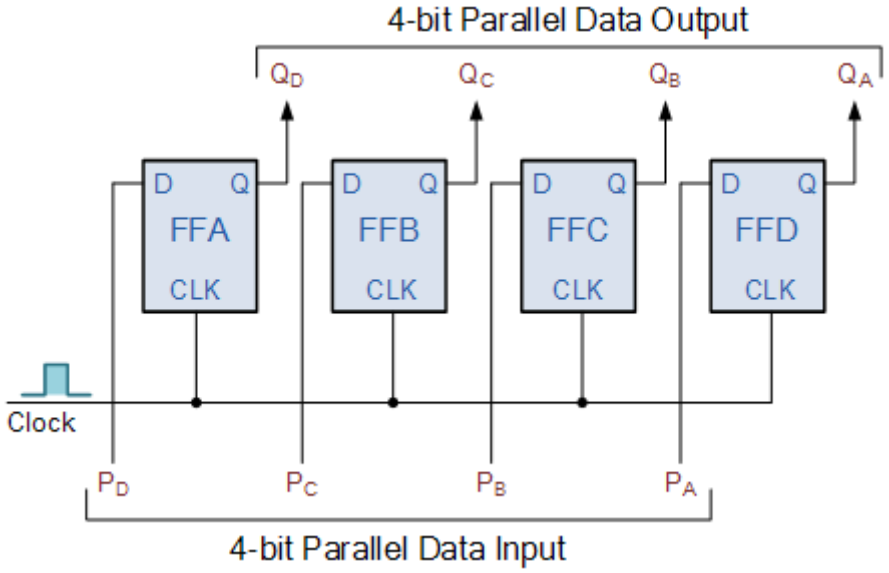
Timing Diagram:



Parallel-in to Parallel-out (PIPO) Shift Register

The final mode of operation is the Parallel-in to Parallel-out Shift Register. This type of shift register also acts as a temporary storage device or as a time delay device similar to the SISO configuration above. The data is presented in a parallel format to the parallel input pins P_A to P_D and then transferred together directly to their respective output pins Q_A to Q_D by the same clock pulse. Then one clock pulse loads and unloads the register. This arrangement for parallel loading and unloading is shown below.

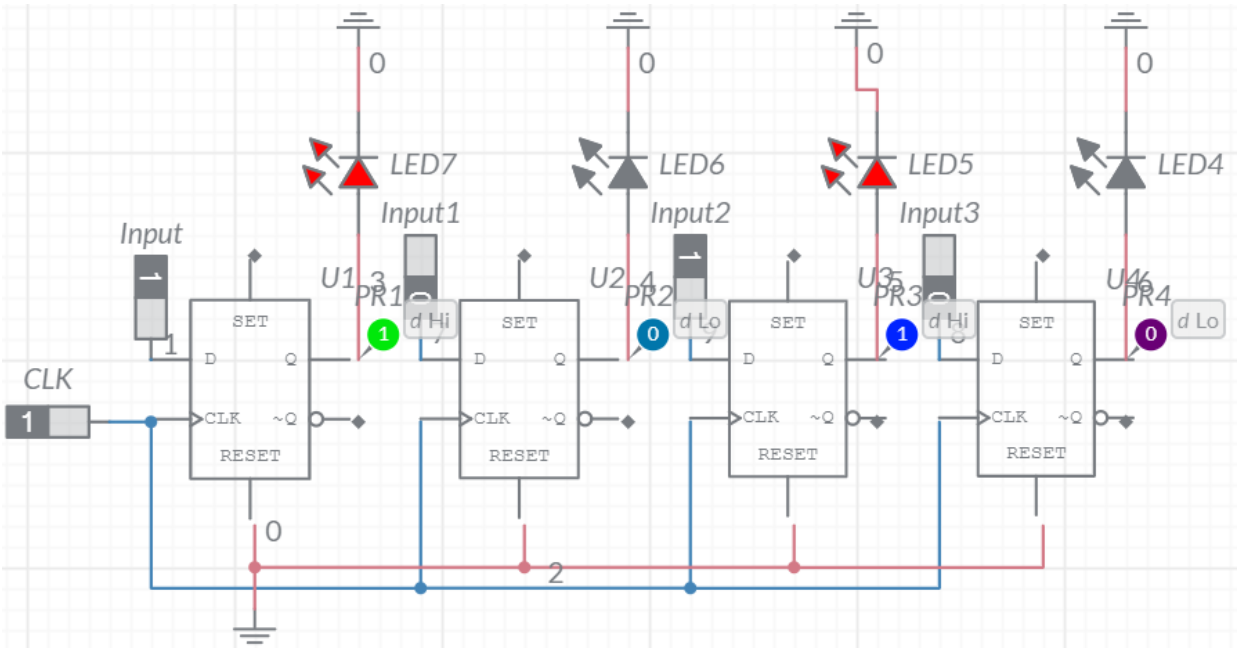
4-bit Parallel-in to Parallel-out Shift Register

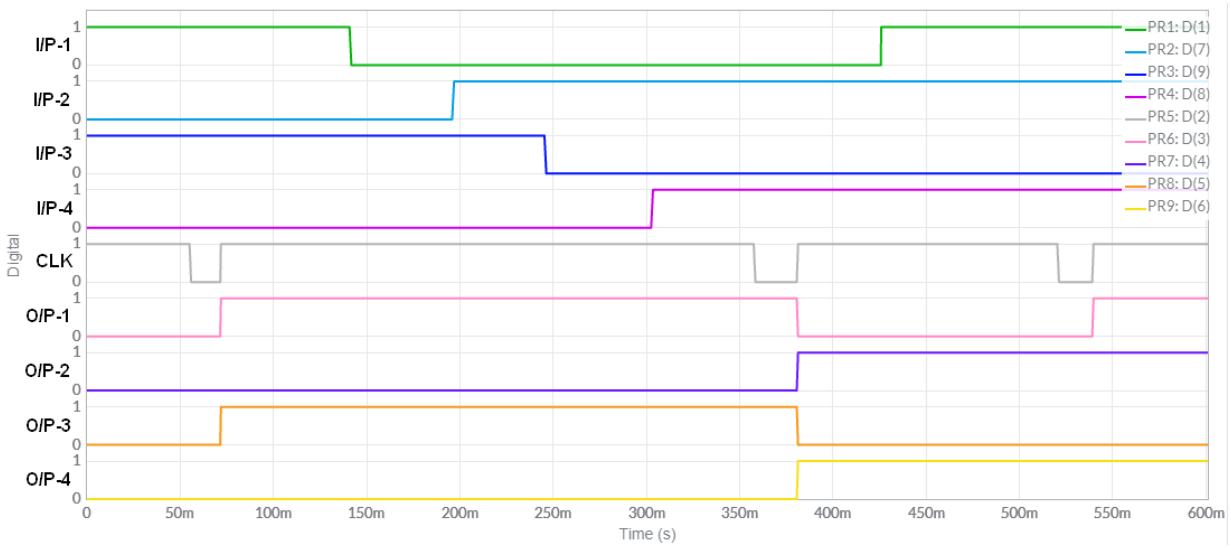


The PIPO shift register is the simplest of the four configurations as it has only three connections, the parallel input (PI) which determines what enters the flip-flop, the parallel output (PO) and the sequencing clock signal (Clk).

Similar to the Serial-in to Serial-out shift register, this type of register also acts as a temporary storage device or as a time delay device, with the amount of time delay being varied by the frequency of the clock pulses. Also, in this type of register there are no interconnections between the individual flip-flops since no serial shifting of the data is required.

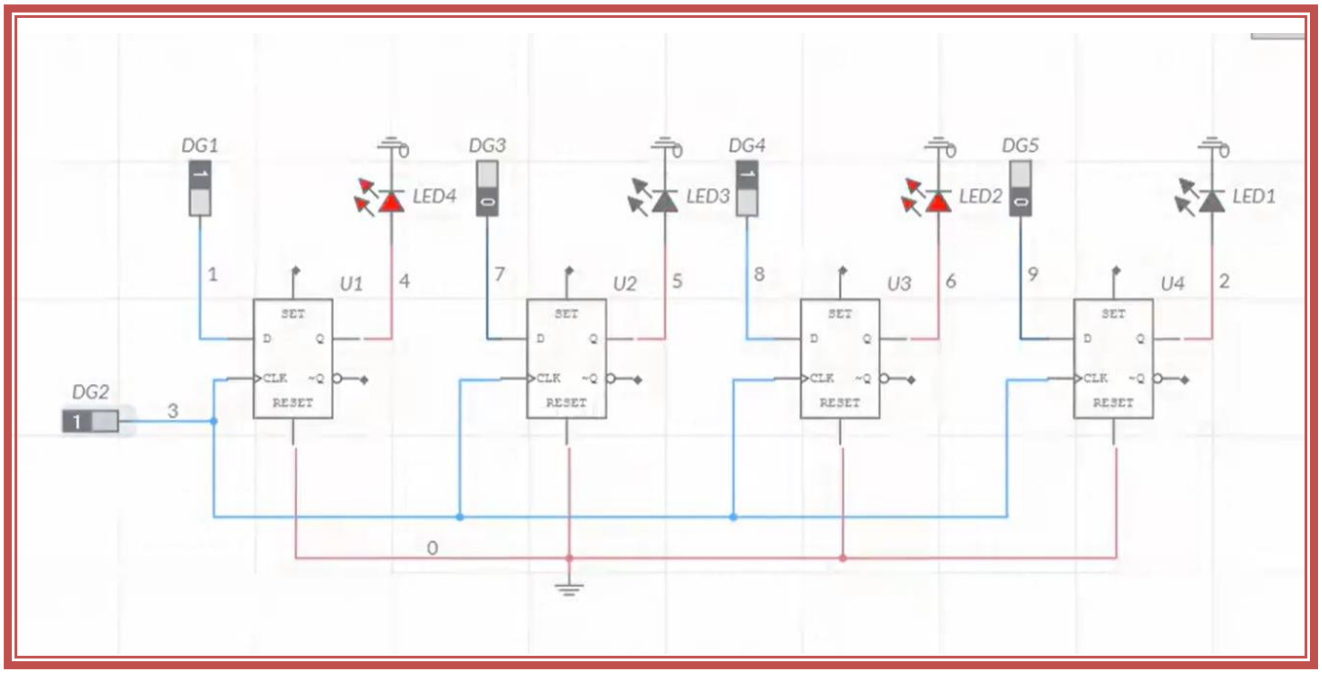
PIPO CIRCUIT DIAGRAM in Multisim:



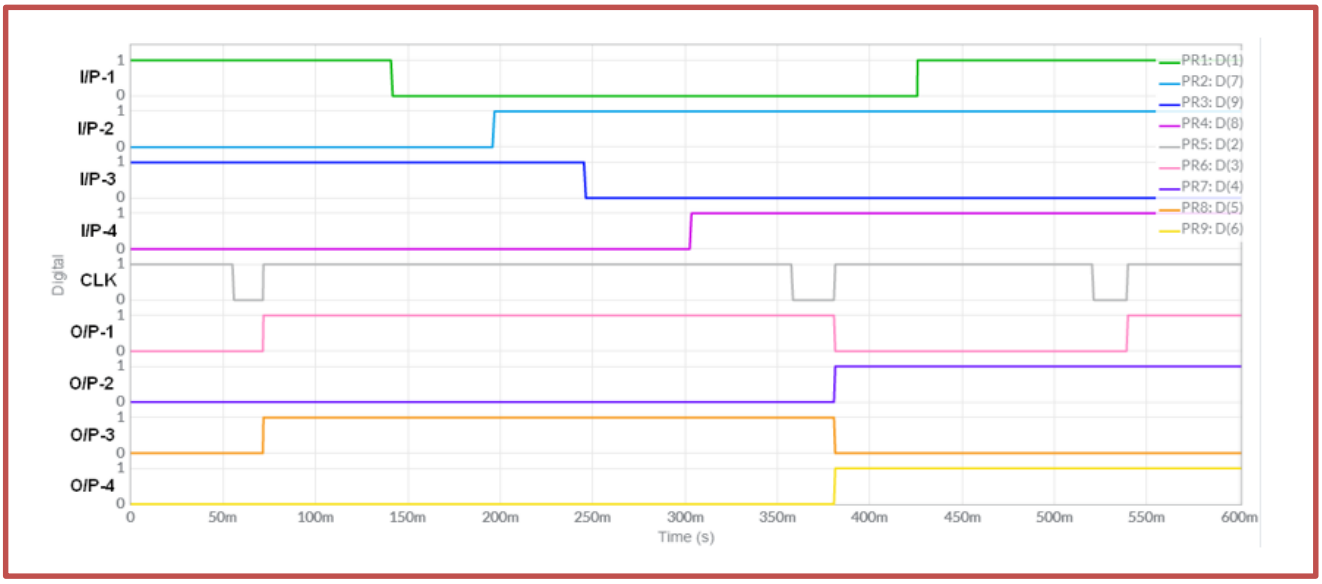


Simulation Results:

Circuit Diagram:



Timing Diagram:



Reference for Theory: https://www.electronics-tutorials.ws/sequential/seq_5.html
https://www.electronicsengineering.nbcafe.in/parallel-in-serial-out-shift-register-piso/

Result:

Thus, the implementation of D flip flop using Multisim is verified.

DEPT. of Computer Science Engineering

SRM IST, Ramapuram

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	**10**
Title of Experiment	**Design and simulation of 3-bit Synchronous up and down counter using multisim**
Name of the candidate	Sathya L J K
Register Number	RA2011026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva	5	
2	Circuit Connection and Execution	10	
3	Verification of Results	5	
	Total	**20**	

Staff Signature with date

10. Design and simulation of 3-bit Synchronous up and down counter using multisim

AIM:

To design and simulation 3-bit synchronous up and down counter using multisim software.

APPARATUS REQUIRED:

S.No	Apparatus	Type	Quantity
1)	JK - Flipflop	Link attached	3
2)	Multisim online		

THEORY: COUNTER

Counter is the most useful and versatile subsystem of digital branch. Counter is going to count number of clock pulses applied to it. Maximum count that binary counter can count is 2^n-1. Clock pulses occur at regular time interval, so that counter can be used to measure time or frequency. Digital counters are integrated circuits (ICs) that count events in computers and other digital systems. Because they must remember past states, digital counters include memory. Generally, digital counters consist of bistable devices or bistable multi vibrators called flip-flops. The number of flip-flops and the way in which they are connected determines the number of states and the sequence of states that digital counters complete in each full cycle.

Counters can be subdivided into 2 groups:

1. Asynchronous Counters
2. Synchronous Counters

The way in which devices are clocked determines whether digital counters are categorized as synchronous or asynchronous. In synchronous devices (such as synchronous BCD counters and synchronous decade counters), one clock triggers all of the flip-flops simultaneously.

With asynchronous devices, often called asynchronous ripple counters an external clock pulse triggers only the first first-flop. Each successive flip-flop is then clocked by one of the

outputs (Q or Q') of the previous flip-flop.

Digital counters are configured as UP (counting in increasing sequence), DOWN (counting in decreasing sequence) or Bidirectional (UP / DOWN).

Synchronous / Asynchronous counter can be subdivided into following subgroups:

- Sequential Counters: States of counter are sequential.
- Non-sequential Counters: Sequence or states of counter are sequential but irregular.
- Regular Counters: In this counters, FFs are used. There is direct relation between number of states and number of FFs used i.e. N=2m.
- Decade counter – counts through ten states per stage.
- Up down counter – counts both up and down, under command of a control input

Some of the commercial ICs used for design of Counters:

- ✓ IC 7490-Decade Counter
- ✓ IC 7492 Divide by 10 Counter
- ✓ IC 7493 4 - bit binary Counter
- ✓ IC 74190 Up -Down Decade Counter
- ✓ IC74191 Binary Up-down Counter

a. 3-BIT SYNCHRONOUS UP COUNTER

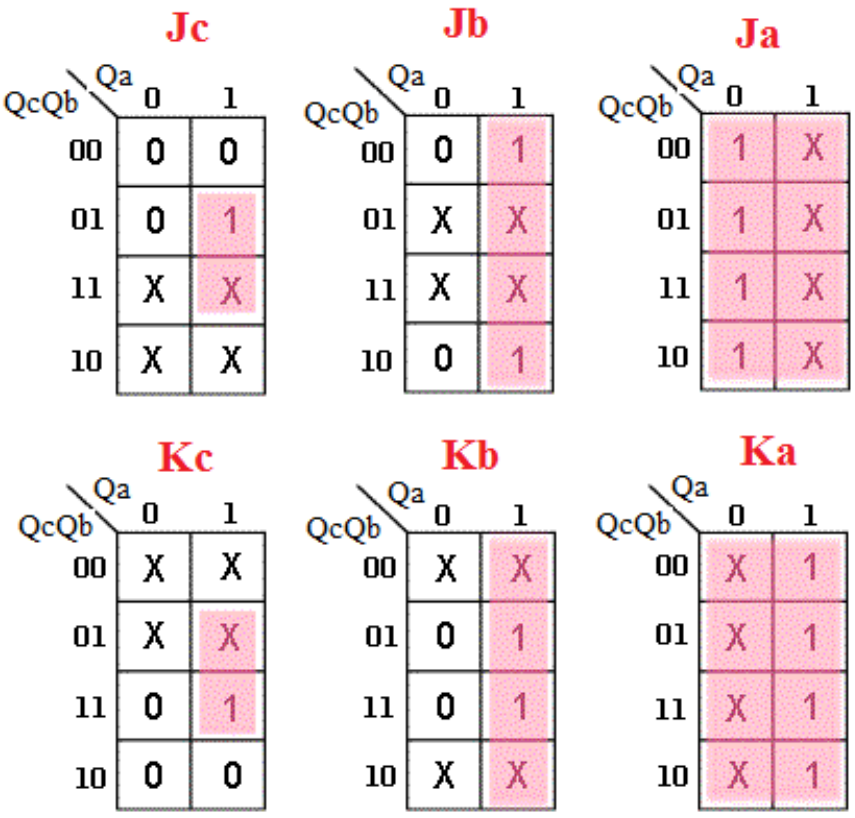
The 3 bit up counter shown in below diagram is designed by using JK flip flop. External clock pulse is connected to all the flip flops in parallel. For designing the counters JK flip flop is preferred.

The significance of using JK flip flop is that it can toggle its state if both the inputs are high, depending on the clock pulse. The inputs of first flip flop are connected to HIGH (logic 1), which makes the flip flop to toggle, for every clock pulse entered into it. So the synchronous counter will work with single clock signal and changes its state with each pulse.

Truth Table:

Clk	Present State			Next state			Flipflop Inputs					
	Qc	Qb	Qa	Q_{c+1}	Q_{b+1}	Q_{a+1}	Jc	Kc	Jb	Kb	Ja	Ka
1	0	0	0	0	0	1	0	X	0	X	1	X
2	0	0	1	0	1	0	0	X	1	X	X	1
3	0	1	0	0	1	1	0	X	X	0	1	X
4	0	1	1	1	0	0	1	X	X	1	X	1
5	1	0	0	1	0	1	X	0	0	X	1	X
6	1	0	1	1	1	0	X	0	1	X	X	1
7	1	1	0	1	1	1	X	0	X	0	1	X
8	1	1	1	0	0	0	X	1	X	1	X	1

K-Map Simplification:

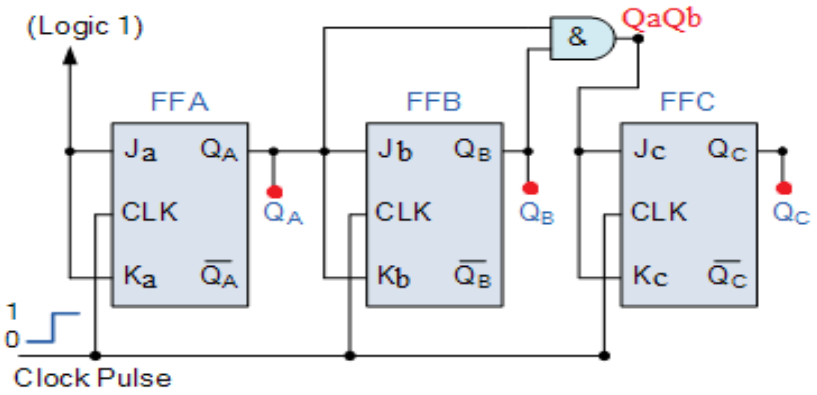


Ja = 1; Ka = 1

Jb = Qa; Kb = Qa

Jc = Qa Qb; Kc = Qa Qb

Logic Diagram:



b. 3-BIT SYNCHRONOUS DOWN COUNTER

Truth Table:

Clk	Present State			Next state			Flipflop Inputs					
	Qc	Qb	Qa	Q_{c+1}	Q_{b+1}	Q_{a+1}	Jc	Kc	Jb	Kb	Ja	Ka
1	1	1	1	1	1	0	X	0	X	0	X	1
2	1	1	0	1	0	1	X	0	X	1	1	X
3	1	0	1	1	0	0	X	0	0	X	X	1
4	1	0	0	0	1	1	X	1	1	X	1	X
5	0	1	1	0	1	0	0	X	X	0	X	1
6	0	1	0	0	0	1	0	X	X	1	1	X
7	0	0	1	0	0	0	0	X	0	X	X	1
8	0	0	0	1	1	1	1	X	1	X	1	X

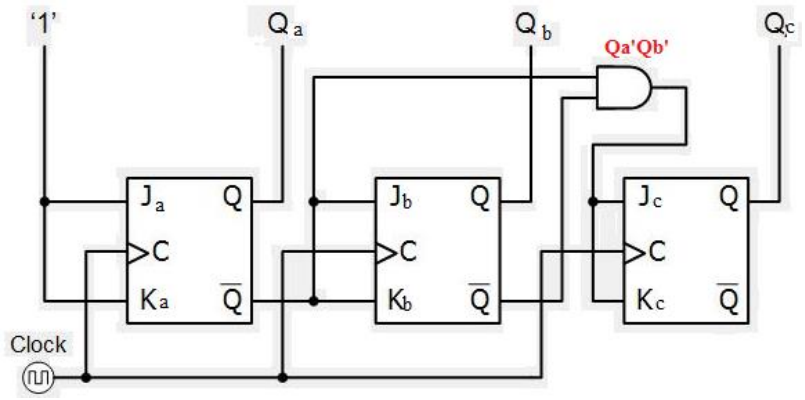
K-Map Simplification:

Ja = 1; Ka = 1

Jb = Qa'; Kb = Qa'

Jc = Qa' Qb'; Kc = Qa' Qb'

Logic Diagram:

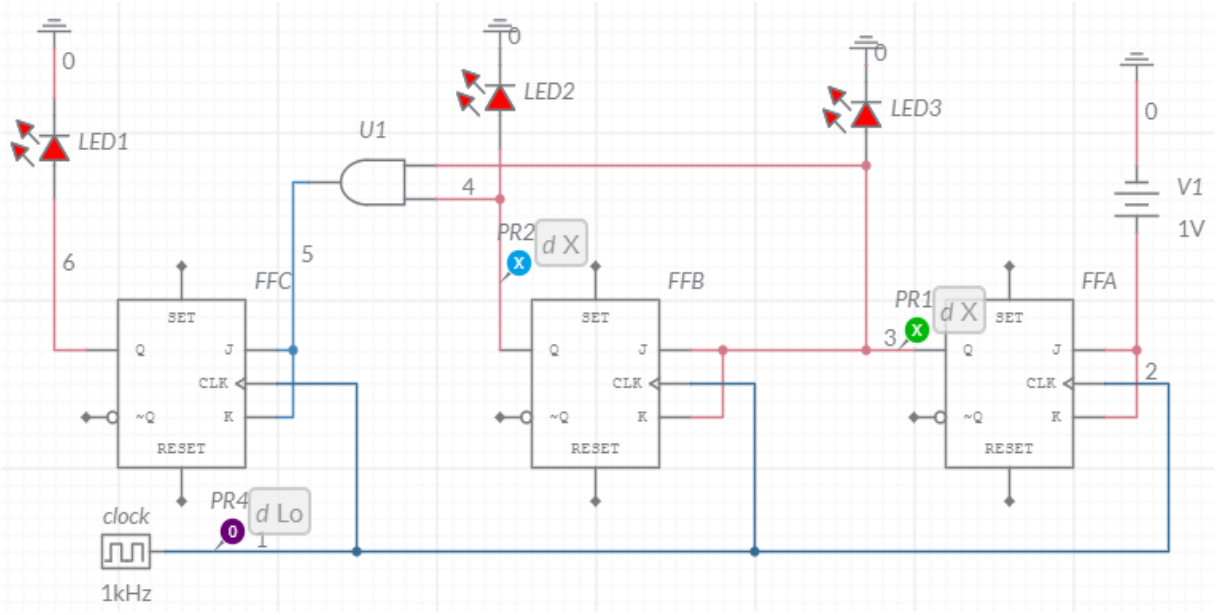


PROCEDURE:

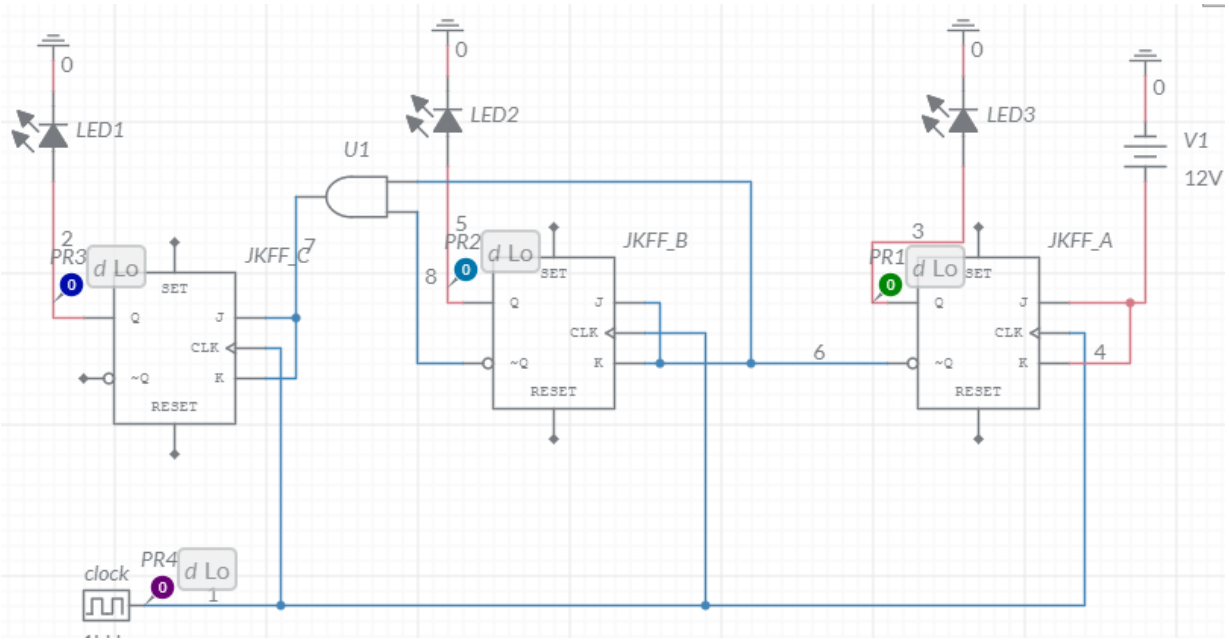
1. Open online Multisim software and create a new circuit
2. Use the following link to add JK- flipflop and Clock Pulse (it's not available in online multisim)
   https://www.multisim.com/content/R47jgTRSmr4SeacynyyJ26/jk-ff-clock/open/
3. Connect the circuit as per given logic diagram.
4. Apply the Clock Pulse parameters as Duty=50%, Frequency =1kHz.
5. Verify results of 3-bit synchronous up and down counter.

Simulation Diagram:

3-BIT SYNCHRONOUS UP COUNTER:

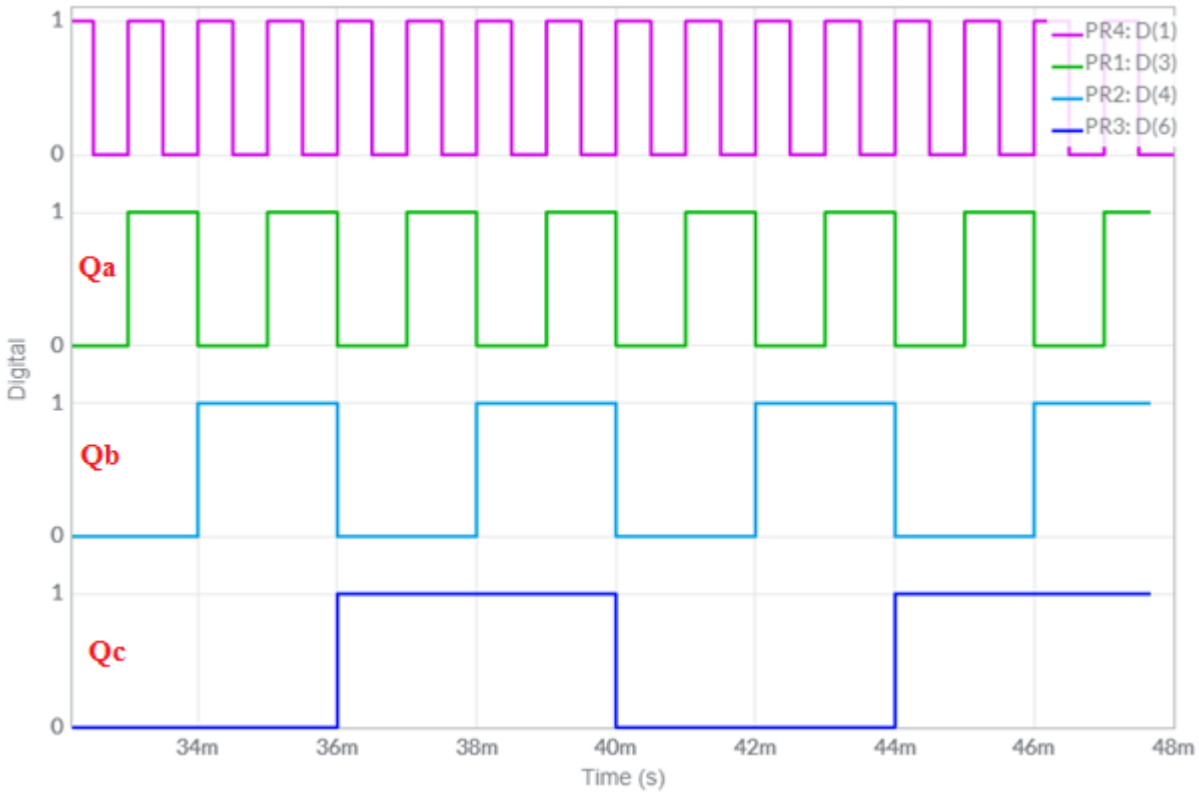


3-BIT SYNCHRONOUS DOWN COUNTER:

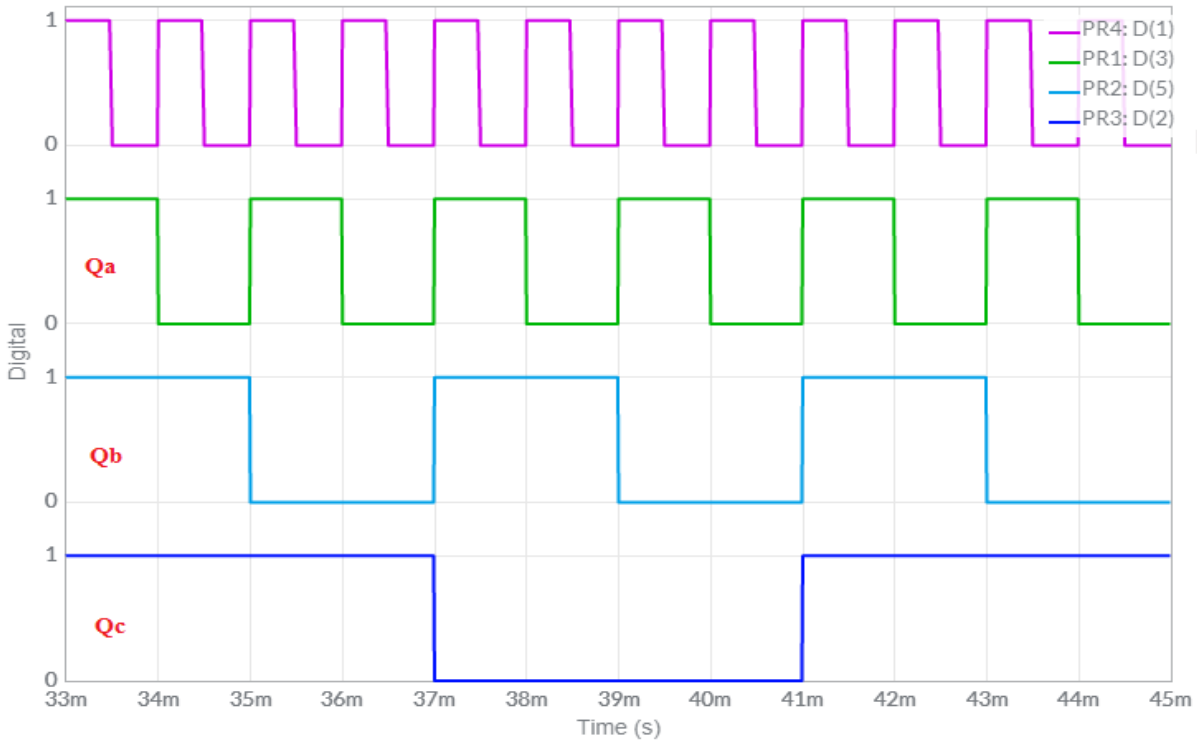


Simulation Result:

3-BIT SYNCHRONOUS UP COUNTER:



3-BIT SYNCHRONOUS DOWN COUNTER:



Result:

Thus, the design and simulation of simulation 3-bit synchronous up and down counter using multisim software.

DEPT. Of Computer Science Engineering

SRM IST, RAMAPURAM

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	11
Title of Experiment	HDL PROGRAM FOR COMBINATIONAL CIRCUITS
Name of the candidate	Sathya.LJK
Register Number	RA20110026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva / Online Quiz	5	
2	Execution	10	
3	Model Calculation / Result Analysis	5	
	Total	20	

Staff Signature with date

Experiment no: 10
Date:

HDL PROGRAM FOR COMBINATIONAL CIRCUITS

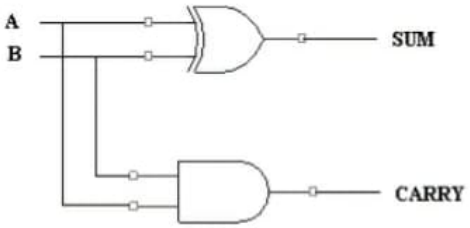
AIM:
To develop the source code for adders and subtractors by using VERILOG and obtain the simulation & synthesis.

ALGORITM:
Step1: Define the specifications and initialize the design.
Step2: Declare the name of the entity and architecture by using VHDL source code.
Step3: Write the source code in VERILOG.
Step4: Check the syntax and debug the errors if found, obtain the synthesis report.
Step5: Verify the output by simulating the source code.
Step6: Write all possible combinations of input using the test bench.
Step7: Obtain the place and route report.

BASIC ADDERS & SUBTRACTORS:

HALF ADDER:

LOGIC DIAGRAM:



TRUTH TABLE:

A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

VHDL SOURCE CODE:

Dataflow Modeling:

```
module HF(sum,carry,a,b);
  output sum, carry;
  input a, b;
  assign sum = a ^ b; // assigning sum
  assign carry = a & b; // assigning carry
endmodule
```

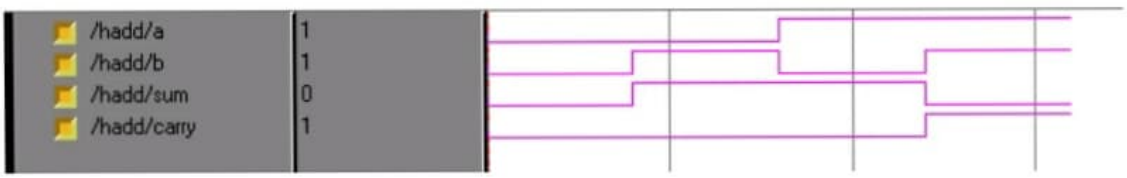
Test bench

```
module HF_TB();
  reg  a, b;
 wire  sum, carry;
  HF uut(sum,carry,a,b);
  initial
  begin
 a = 0; b =0;
```

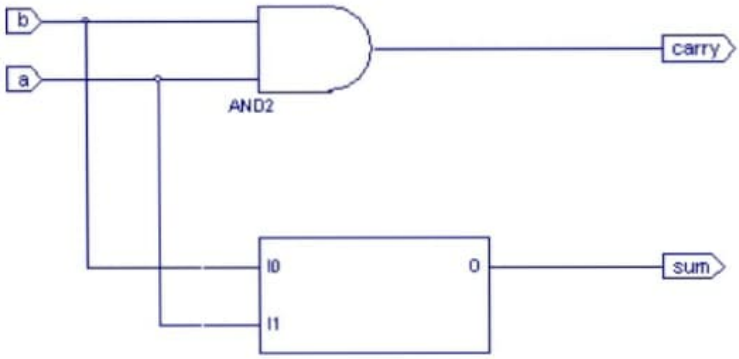
```
  #5; a = 0; b =1;
  #5; a = 1; b =0;
  #5; a = 1; b =1;
  #5;
   end
  //Dump waves (only required here)
initial
  begin
   $dumpfile("dump.vcd");
   $dumpvars(1);
  end

endmodule
```

Simulation output:

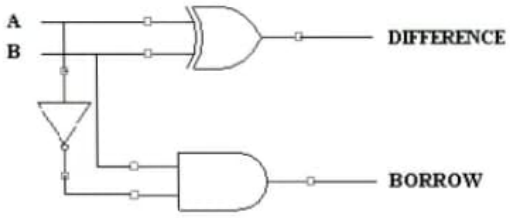


Synthesis RTL Schematic:



HALF SUBTRACTOR:

LOGIC DIAGRAM:



TRUTH TABLE

A	B	DIFFERENCE	BORROW
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

VERILOG SOURCE CODE:

Dataflow Modeling:

```
module Half_Subtractor_2(output D, B, input X, Y);
```

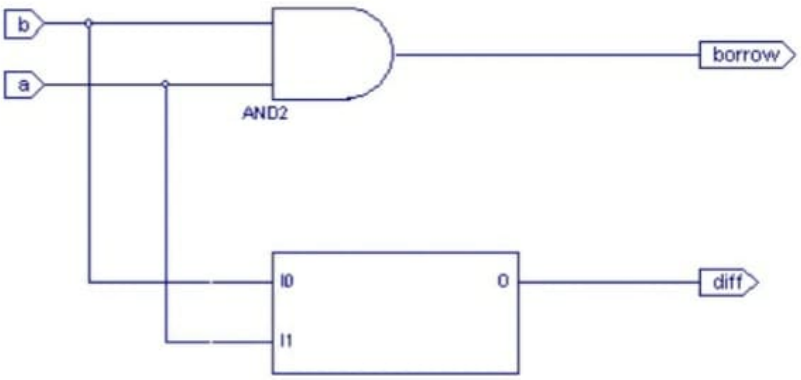
```
assign D = X ^ Y;
assign B = ~X & Y;
endmodule
```

Test Bench

```
module Half_Subtractor_2_tb;
wire D, B;
reg X, Y;
Half_Subtractor_2 Instance0 (D, B, X, Y);
initial begin
   X = 0; Y = 0;
#1  X = 0; Y = 1;
#1  X = 1; Y = 0;
#1  X = 1; Y = 1;
end
initial begin
   $monitor ("%t, X = %d| Y = %d| B = %d| D = %d", $time, X, Y, B, D);
   $dumpfile("dump.vcd");
   $dumpvars();
end
endmodule
```

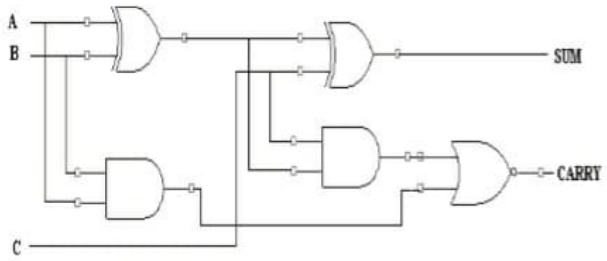
Simulation output:

Synthesis RTL Schematic:



FULL ADDER:

LOGIC DIAGRAM:



TRUTH TABLE:

A	B	C	SUM	CARRY
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

VERILOG SOURCE CODE:

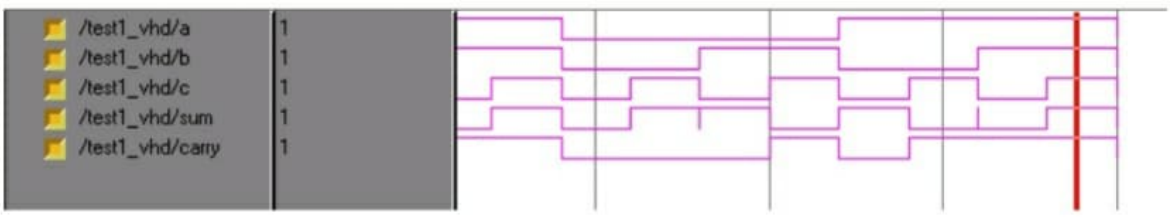
Dataflow Modeling:

```
module Full_Adder (a, b, c, sum, cout);
 input a;
input b;
  input c;
  output sum;
  output cout;
assign sum=a^b^c;
assign cout=(a & b) | (b & c) | (c & a);
endmodule
```

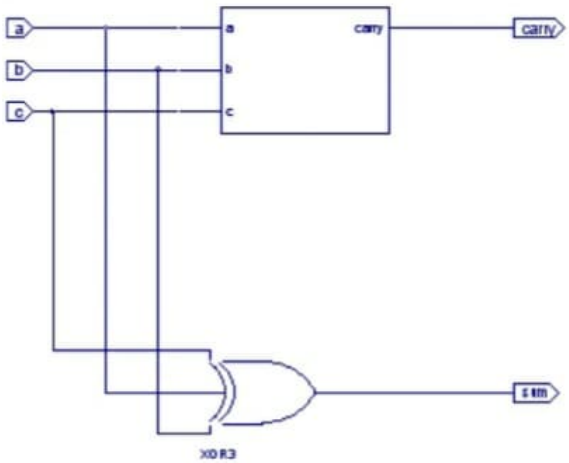
Test Bench

```
module Test_Full_Adder;
  reg  a, b, c;
  wire sum, cout;
  Full_Adder FA (a, b, c, cout, sum);
  initial begin
 $dumpfile("Test_Full_Adder.vcd");
    $dumpvars(1, FA);
    a=0; b=0; c=0;
    #20 a=1; b=1;
    #20 a=0; b=0; c=1;
    #20 a=1; c=0;
    #20 $finish;
  end
endmodule
```

Simulation output:

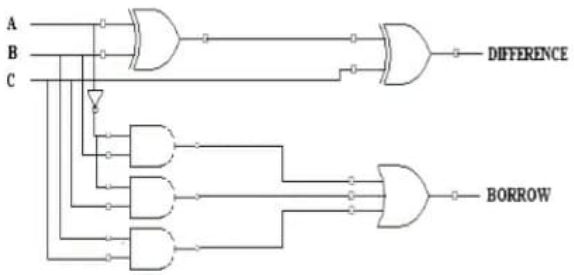


Synthesis RTL Schematic:



FULL SUBTRACTOR:

LOGIC DIAGRAM:



TRUTH TABLE:

A	B	C	DIFFERENCE	BORROW
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

VERILOG SOURCE CODE:

Dataflow Modeling:

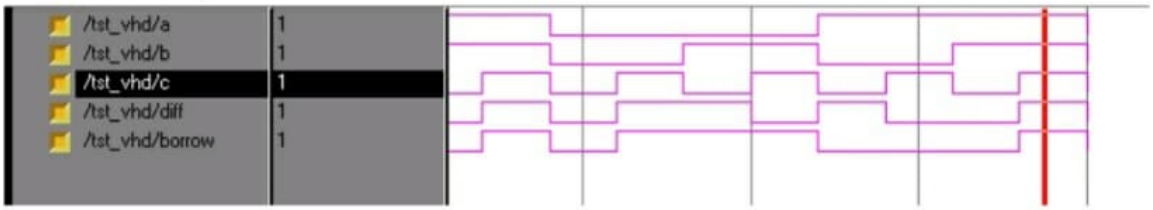
```
module Full_Subtractor_3(output D, B, input X, Y, Z);
assign D = X ^ Y ^ Z;
assign B = ~X & (Y^Z) | Y & Z;
endmodule
```

Test bench

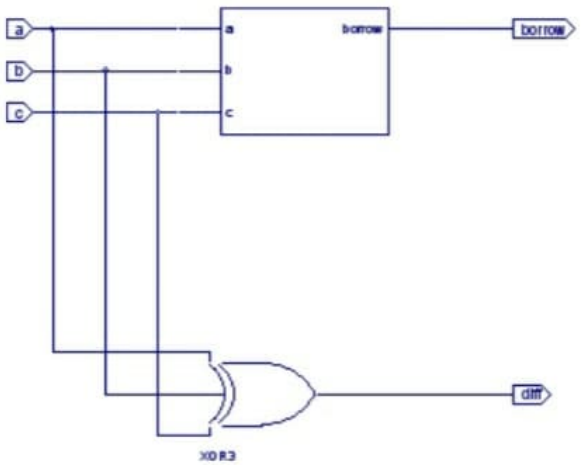
```
module Full_Subtractor_3_tb;
wire D, B;
reg X, Y, Z;
Full_Subtractor_3 Instance0 (D, B, X, Y, Z);
initial begin
  X = 0; Y = 0; Z = 0;
```

```
    #1  X = 0; Y = 0; Z = 1;
    #1  X = 0; Y = 1; Z = 0;
    #1  X = 0; Y = 1; Z = 1;
    #1  X = 1; Y = 0; Z = 0;
    #1  X = 1; Y = 0; Z = 1;
    #1  X = 1; Y = 1; Z = 0;
    #1  X = 1; Y = 1; Z = 1;
    end
    initial begin
      $monitor ("%t, X = %d| Y = %d| Z = %d| B = %d| D = %d", $time, X, Y, Z, B,
D);
      $dumpfile("dump.vcd");
      $dumpvars();
    end
    endmodule
```

Simulation output:



Synthesis RTL Schematic:



RESULT:

Thus the OUTPUT of HDL program for Combinational circuits is done and verified.

DEPT. Of Computer Science Engineering

SRM IST, RAMAPURAM

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	12
Title of Experiment	HDL program for Binary counters
Name of the candidate	Sathya.LJK
Register Number	RA20110026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva / Online Quiz	5	
2	Execution	10	
3	Model Calculation / Result Analysis	5	
	Total	20	

Staff Signature with date

AIM:

To write a verilog HDL program for binary counter and verify its output.

SOFTWARE REQUIRED:

Xilinx ISE
10.1

ALGORITHM:

Step1: Define the specifications and

initialize the design. Step2: Write the

source code in VERILOG.

Step3: Check the syntax and perform synthesis .

Step4: Write different combinations of input using the test

bench.

Step5:Verify the output by simulating the source code.

VERILOG SOURCE CODE:

```
module counter (  input clk,
input rstn,
output reg[3:0] out);
always @ (posedge clk) begin
if (rstn <= 0)
out <= 0;
else
out <= out + 1;
end
endmodule
```

TESTBENCH:

```
module tb_counter;
reg clk;
reg rstn;
wire [3:0] out;

counter uut(clk,rstn,out);
always #5 clk = ~clk;

// This initial block forms the stimulus of the testbench
initial
```

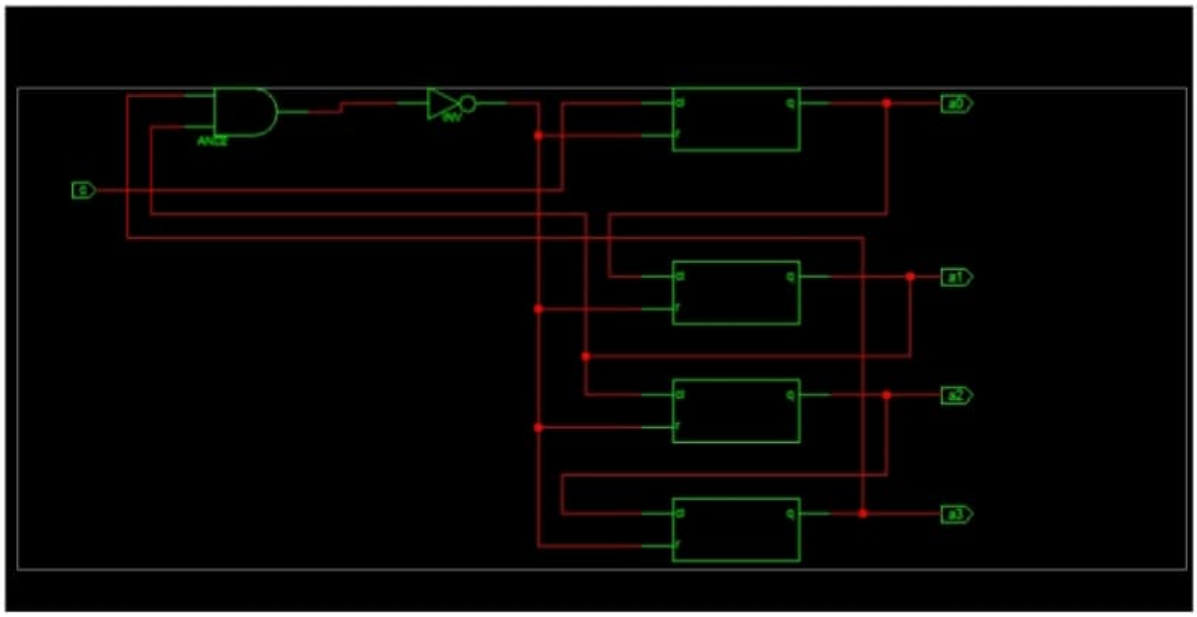
```
begin

clk <= 0;
rstn <= 0;
#20
rstn <= 1;
#80
rstn <= 0;
#50
rstn <= 1;

#200
$finish;
end

initial begin
$dumpvars(0,uut);
$dumpfile("dump.vcd");
end
endmodule
```

RTL SCHEMATIC:



SYNTHESIS REPORT:

* Final Report *

==========

Final Results

RTL Top Level Output File

Name : count1.ngr Top Level Output File Name

:

count1 Output Format :

NGC

Optimization Goal : Speed

Keep Hierarchy : NO

Design Statistics

IOs 6

Cell Usage :

BELS 6

INV 1

LUT2 : 1

LUT2_L : 1

LUT3 : 1

LUT4 : 2

FlipFlops/Latches 4

FDR 4

Clock Buffers 1

BUFGP 1

IO Buffers 5

IBUF 1

OBUF 4

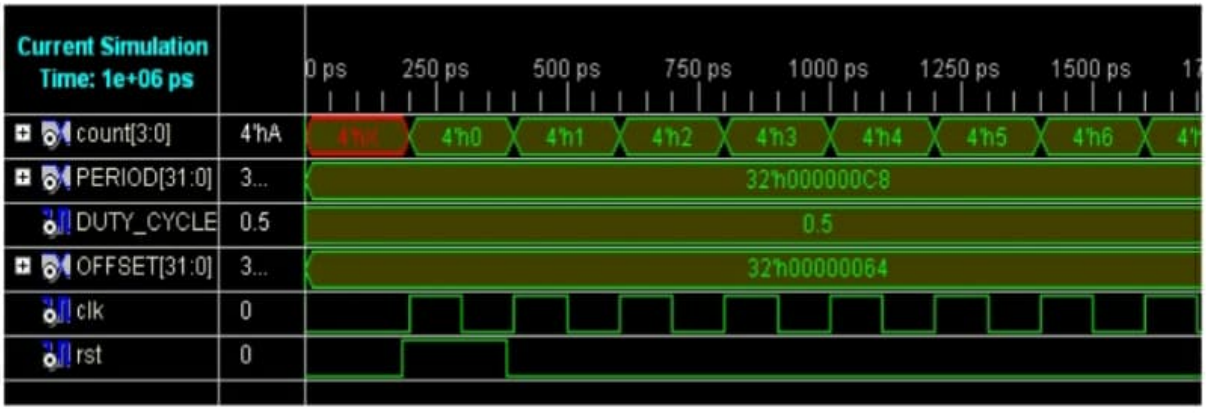
==========

Device utilization summary:

Selected Device : 3s100evq100-4

Number of Slices:	3	out of 960	0%
Number of Slice Flip Flops:	4	out of 1920	0%
Number of 4 input LUTs:	6	out of 1920	0%
Number of IOs:	6		
Number of bonded IOBs:	6	out of 66	9%
Number of GCLKs:	1	out of 24	4%

SIMULATION OUTPUT:



RESULT:

Thus a verilog HDL program was written for binary counter and its output was verified.

DEPT. Of Computer Science Engineering

SRM IST, RAMAPURAM

Sub Code & Name: 18CSS201J - ANALOG AND DIGITAL ELECTRONICS

Experiment No	13
Title of Experiment	HDL program for Mod-10 counters
Name of the candidate	Sathya.LJK
Register Number	RA20110026020032
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Oral Viva / Online Quiz	5	
2	Execution	10	
3	Model Calculation / Result Analysis	5	
	Total	20	

Staff Signature with date

AIM:

To write a verilog HDL program for mod-10 counter and verify its output.

SOFTWARE REQUIRED:

Xilinx ISE 10.1

ALGORITHM:

Step1: Define the specifications and initialize the design.
Step2: Write the source code in VERILOG.
Step3: Check the syntax and perform synthesis .
Step4: Write different combinations of input using the test bench.
Step5:Verify the output by simulating the source code

VERILOG SOURCE CODE:

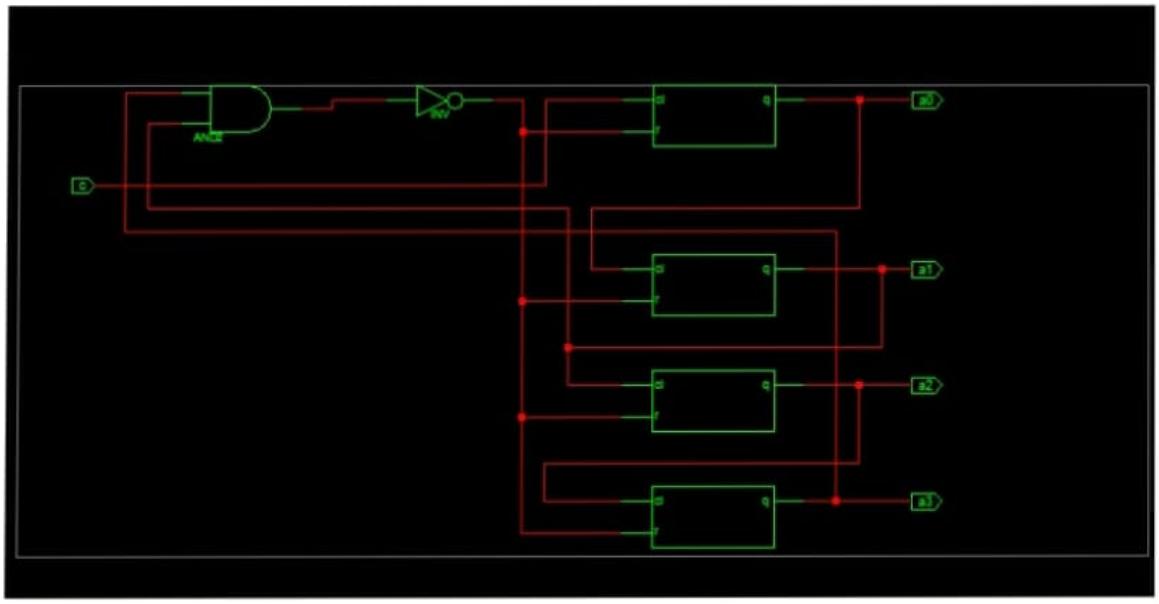
```
module counter(clk,rst,count);
input clk,rst;
output reg [3:0]count;
always@(posedge clk) begin
if(rst == 1 || count == 9)
count <= 0;
else
count <= count + 1;
end
endmodule
```

Test Bench code

```
module tb();
reg clk,rst;
wire [3:0]count;
counter c1(clk,rst,count);
initial begin
clk = 0;
forever #5 clk = ~clk;
end
initial begin
rst = 0;
#5 rst = 1;
#10 rst = 0;
#200 $finish;
end
initial
$monitor("%d%d",clk,count);
initial begin
$dumpfile("dump.vcd");
```

$dumpvars;
end
endmodule

RTL schematic



SYNTHESIS REPORT:

==========

* Final Report *

==========

Final Results

RTL Top Level Output File Name :

modten.ngr Top Level Output File Name :

modten Output Format : NGC

Optimization Goal : Speed

Keep Hierarchy : NO

Design Statistics

IOs 6

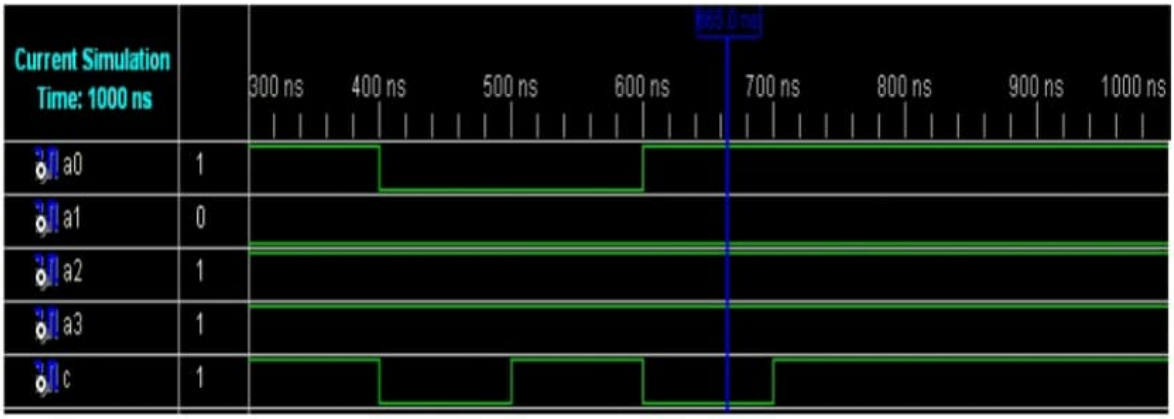
Cell Usage :

```
#  BELS                 : 6
#    INV                : 1
#    LUT2               : 1
#    LUT2_L             : 1
#    LUT3               : 1
#    LUT4               : 2
# FlipFlops/Latches     : 4
#    FDR                : 4
# Clock Buffers         : 1
#    BUFGP              : 1
# IO Buffers            : 5
#    IBUF               : 1
#    OBUF               : 4
=========================================================================
```

Device utilization summary:

Selected Device :	3s100evq100-4	
Number of Slices:	3 out of 960	0%
Number of Slice Flip Flops:	4 out of 1920	0%
Number of 4 input LUTs:	6 out of 1920	0%
Number of IOs:	6	
Number of bonded IOBs:	6 out of 66	9%
Number of GCLKs:	1 out of 24	4%

SIMULATION OUTPUT:



RESULT:

Thus a verilog HDL program was written for mod-10 counter and its output was verified.