Experiment / Assignment / Tutorial No. _1_____

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Batch:B2	Roll No.:	16010121110	Experiment / assignment / tutorial No.: 1

Title: Basic Gates & Universal Gates

Objective: To study the basic gates: AND, OR, NOT and universal gates: NAND, NOR, XOR, XNOR

Expected Outcome of Experiment:

CO1: Recall basic gates and binary, octal & hexadecimal calculations and conversions.

Books/ Journals/ Websites referred:

- Vlab Link: http://vlabs.iitkgp.ernet.in/dec/#
- R. P. Jain, “Modern Digital Electronics”, Tata McGraw Hill
- http://www.ee.surrey.ac.uk/Projects/Labview/gatesfunc/
- http://www.electronics-tutorials.ws/boolean/bool_6.html

Pre Lab/ Prior Concepts:

Gate is a logic circuit with one or more inputs but only one output. Gates are digital (two state) circuit because the input & output are either low or high. Gates provide high output for certain combinations of input & for other combinations the output is low. Total number of combinations for a gate is 2^n; where n is number of input.

Classification: The two types of gate are:

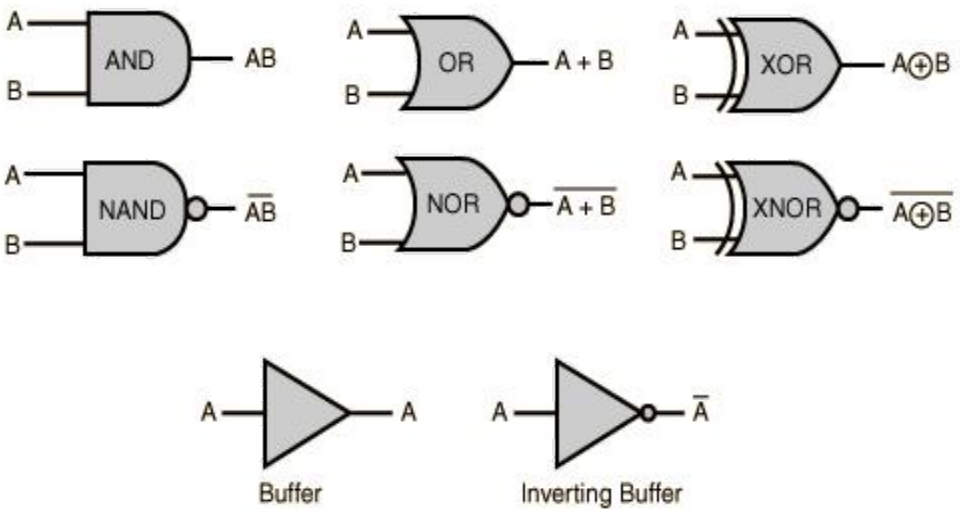
1. **Basic or Fundamental Gates:**

AND, OR, XOR and NOT.

2. Derived Gates:

NOR, XNOR, NAND

Symbols of gates



Type of IC	Specification

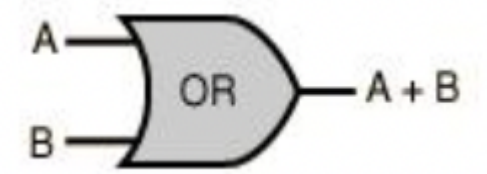
Implementation Details:

Basic Gates

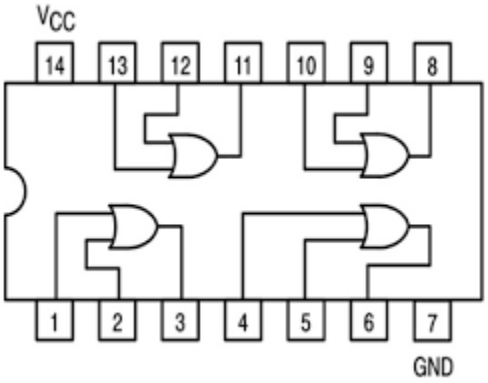
1. **OR gate:** The OR gate has two or more inputs but only 1 output. If any or all the inputs are high, the output is high. If all the inputs are low, the output is low.

Y=A+B

Symbol for OR gate



Pin Diagram For IC 7432



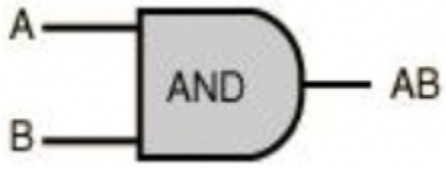
The truth table for OR operations are:

Input		Output
A	B	Y=A+B
0	0	0
0	1	1
1	0	1
1	1	1

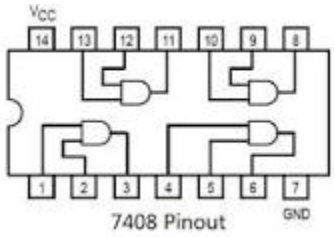
2. **AND gate:** The AND gate has two or more inputs but only one output. If any or all inputs are high then output is also high

Y=A.B

Symbol for AND gate



Pin Diagram For IC 7408



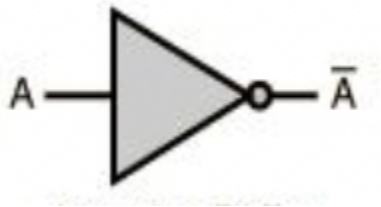
The truth table for AND operations are:

Input		Output
A	B	Y=A.B
0	0	0
0	1	0
1	0	0
1	1	1

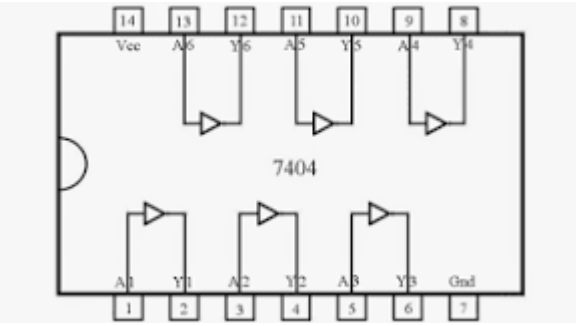
3. **NOT gate:** The Not gate is a gate with only one input and one output. The output is always in opposite state of an input. A NOT gate is also called as Inverter because it performs inversion.

Y= Y'

Symbol for NOT gate



Pin Diagram For IC 7404



The truth table for NOT operations is:

Input	Output
A	Y
0	1
1	0

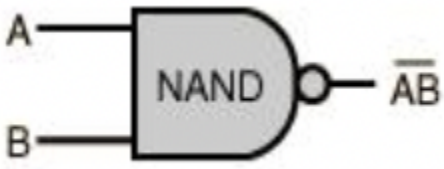
Derived Gates/Universal Gates

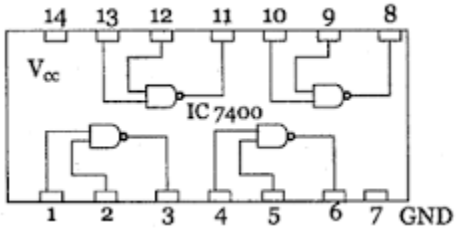
- NAND gate
- NOR gate
- EX-OR gate
- EX-NOR gate

1. **NAND gate:** This is a NOT-AND gate which is equal to an AND gate followed by a NOT gate. The outputs of all NAND gates are high if any of the inputs are low. The symbol is an AND gate with a small circle on the output. The small circle represents inversion.

$Y = (A.B)'$

Symbol **Pin Diagram for IC 7400**





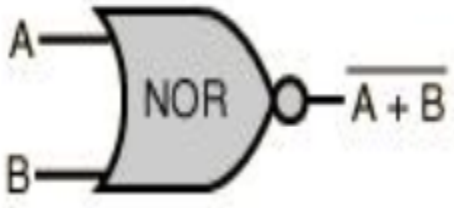
The truth table for NAND operations is:

Input		Output
A	B	Y= $\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

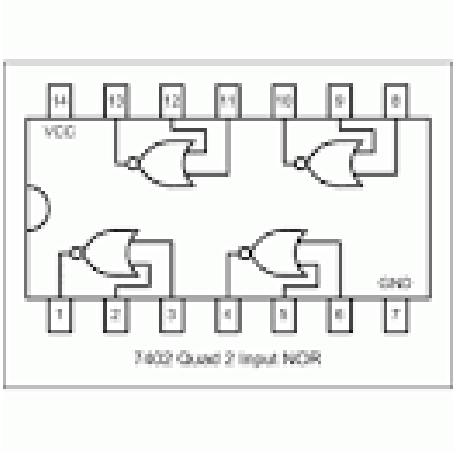
2. **NOR gate:** This is a NOT-OR gate which is equal to an OR gate followed by a NOT gate. The outputs of all NOR gates are low if any of the inputs are high. The symbol is an OR gate with a small circle on the output. The small circle represents inversion.

Y= (A+B)'

Symbol for NOR gate



Pin Diagram For IC 7402

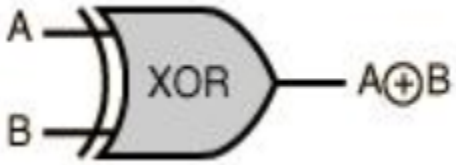


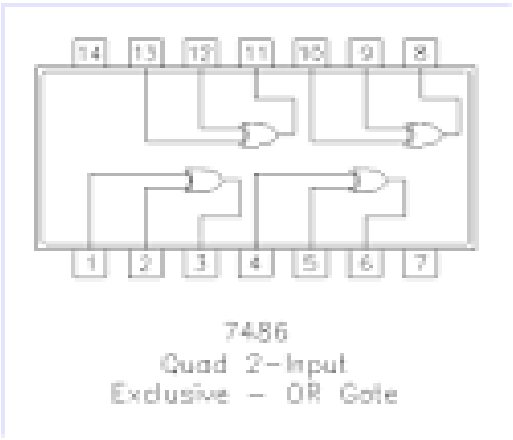
The truth table for NOR operations are:

Input		Output
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

3. **EX-OR gate**: The 'Exclusive-OR' gate is a circuit which will give a high output if either, but not both, of its two inputs are high. An encircled plus sign (⊕) is used to show the EX-OR operation

Symbol for Ex-OR gate **Pin Diagram For IC 7486**



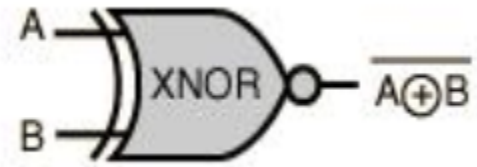


The truth table for XOR operations is:

XOR Truth Table		
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

4. **EX-NOR gate**: The 'Exclusive-NOR' gate circuit does the opposite to the EOR gate. It will give a low output if either, but not both, of its two inputs are high. The symbol is an EXOR gate with a small circle on the output. The small circle represents inversion

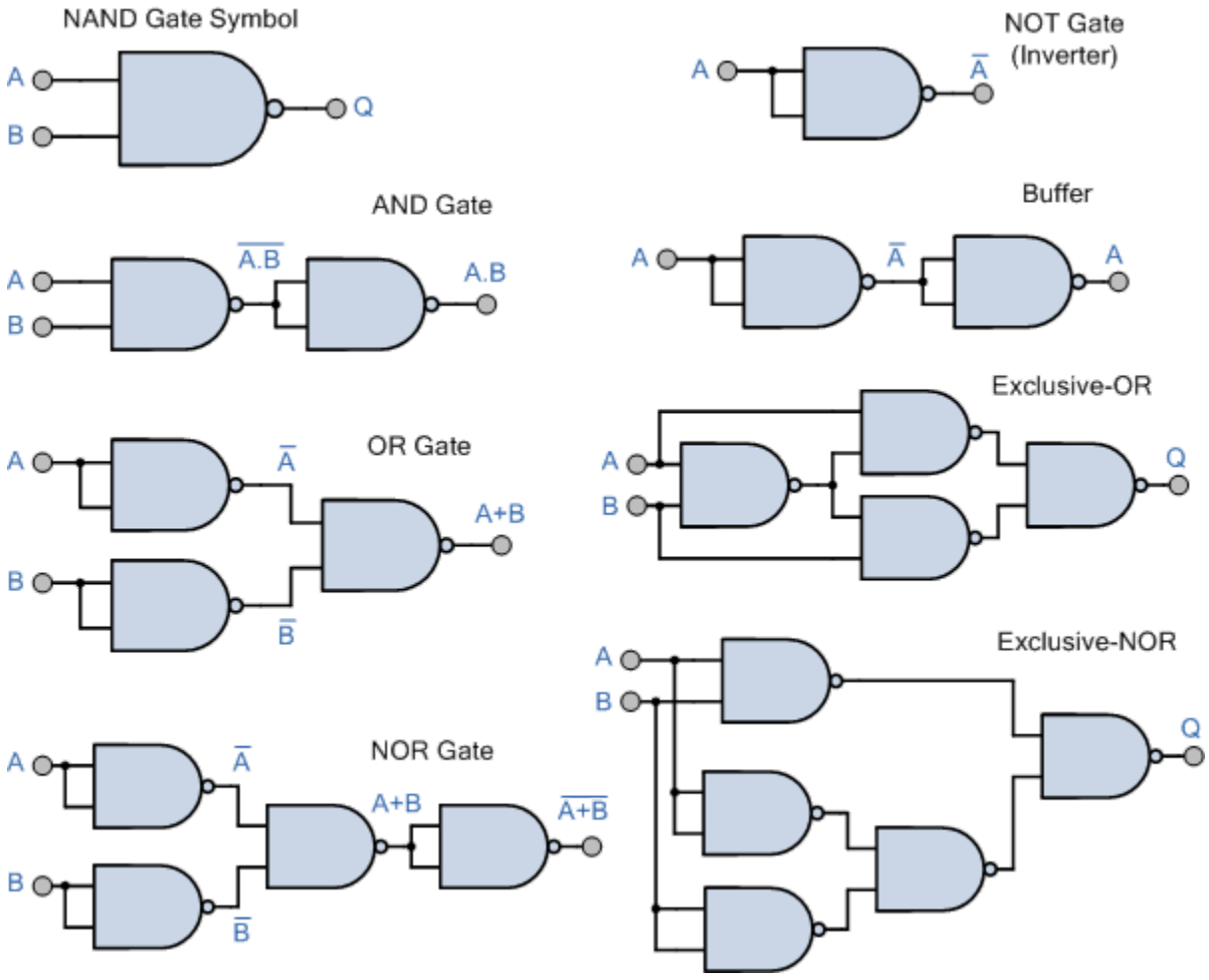
Symbol for Ex-NOR gate



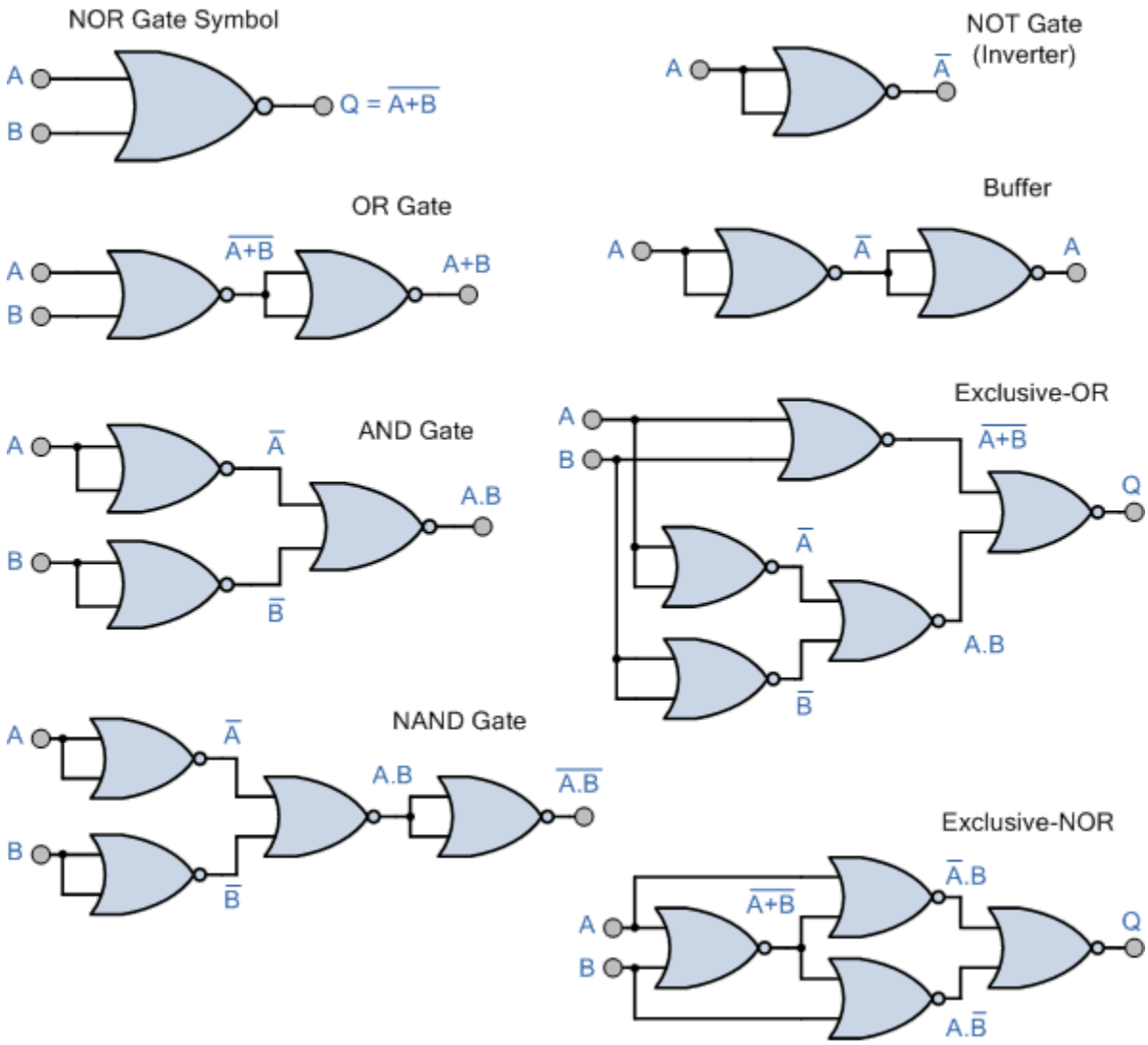
The truth table for XNOR operations is:

A	B	A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1

Implementation Using NAND Gate



IMPLEMENTATION USING NOR GATE

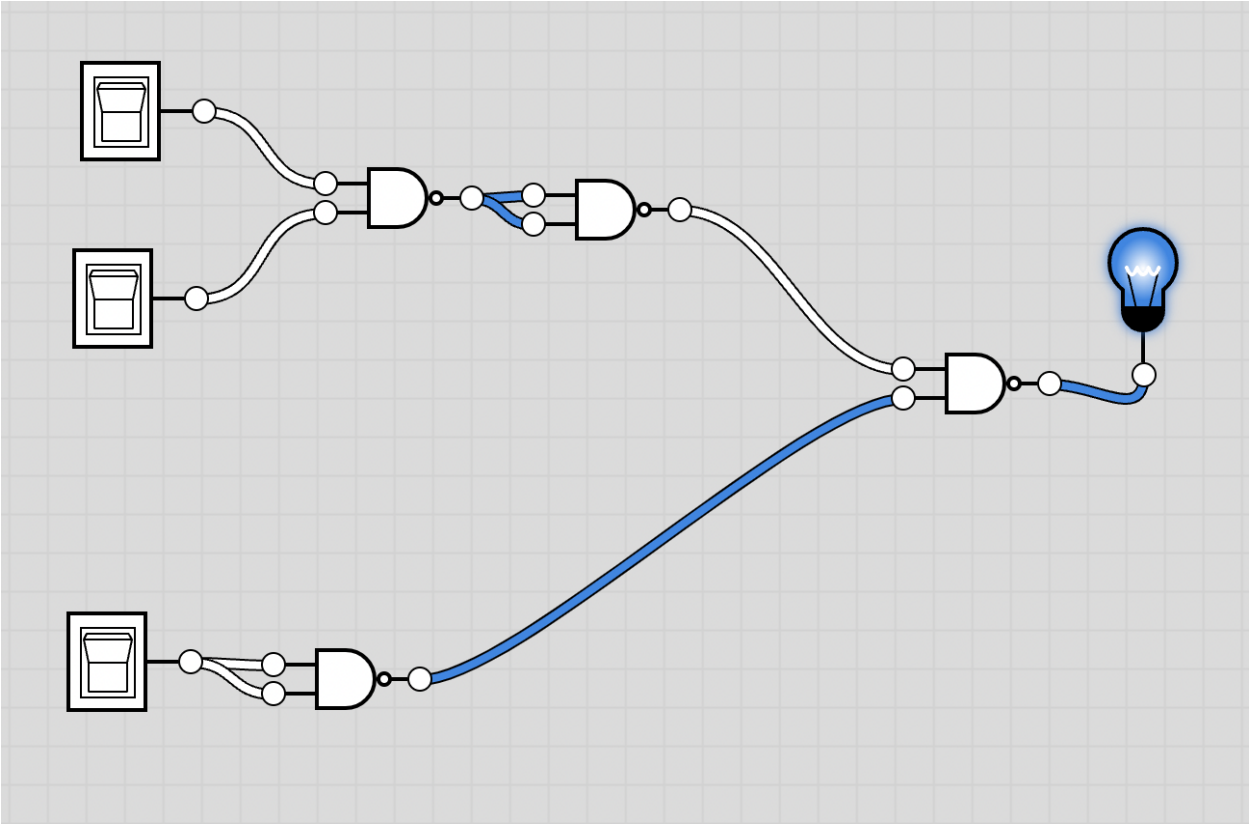


Conclusion:

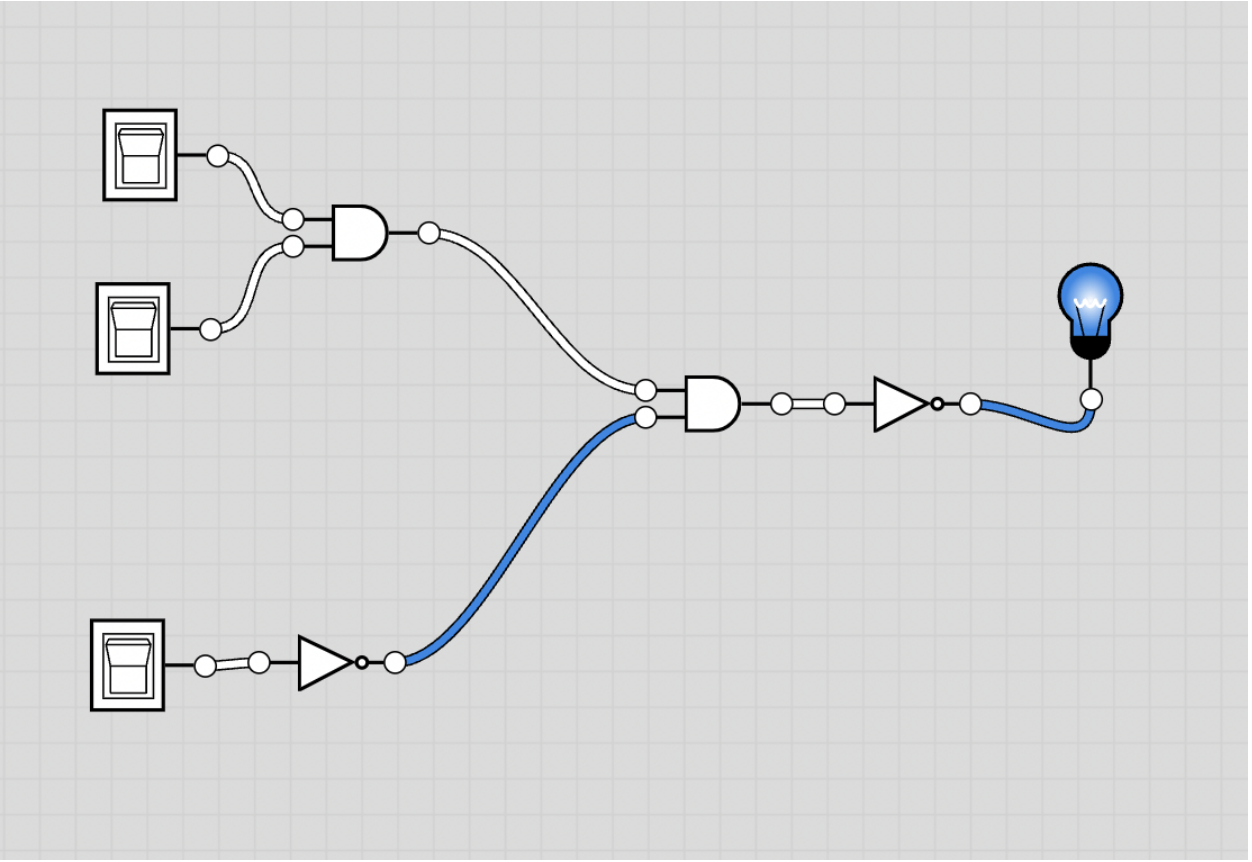
Thus we have implemented the logic of various logic gates using their respective ICs. We also learnt how to implement the logic gates using two fundamental gates- NANA and NOR. We understood the logic behind the truth tables of the gates.

Post Lab Descriptive Questions

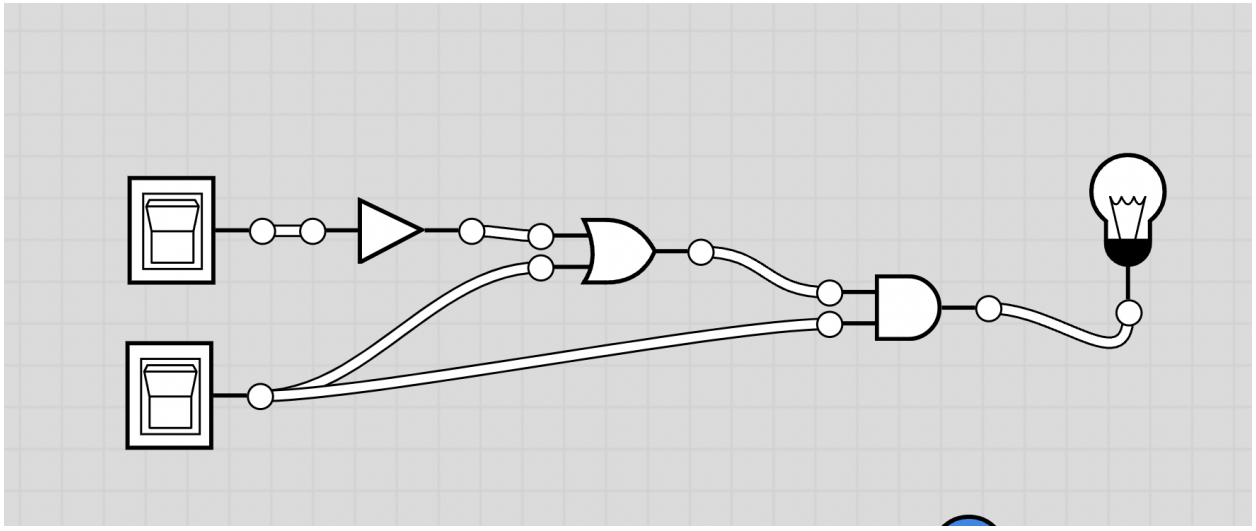
1. Verify the expression (A·B)' + C by:
   a) Using NAND Gate directly.

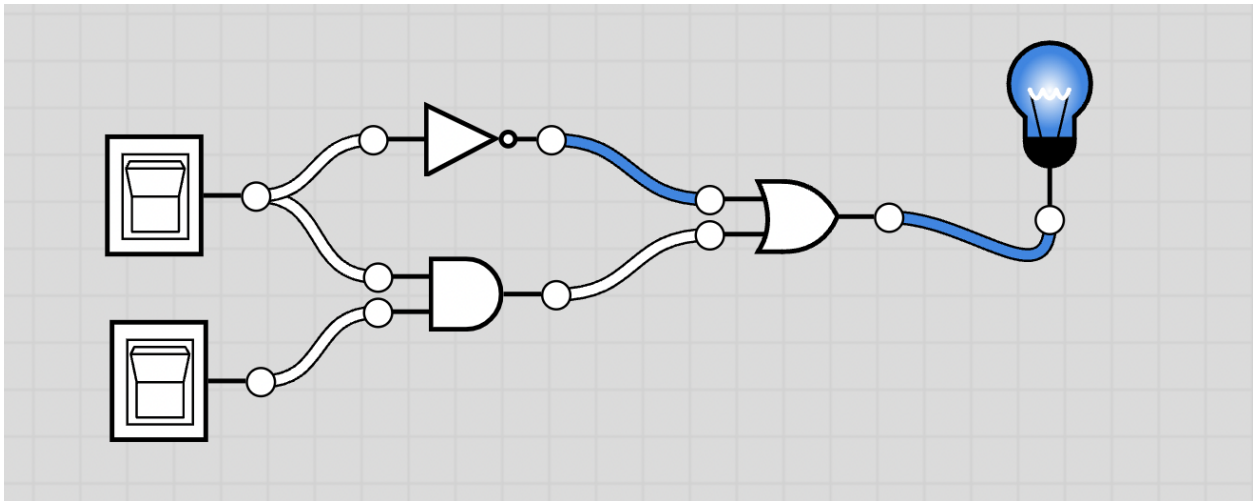


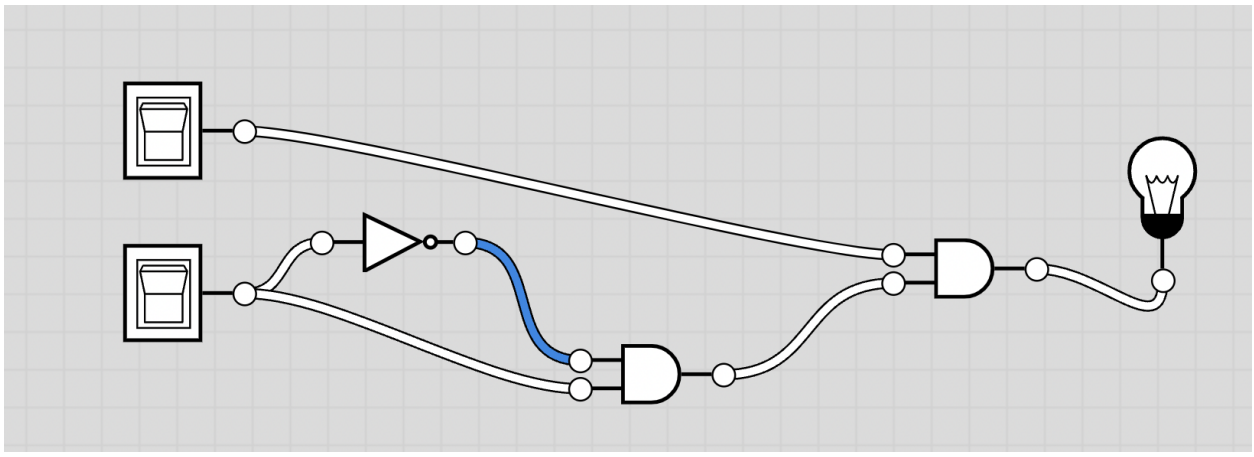
b) Using AND & NOT gate consecutively.

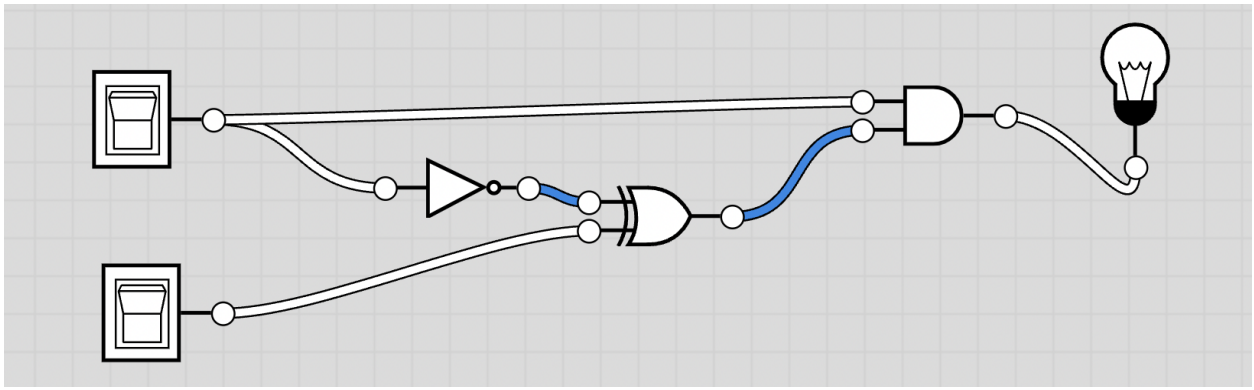


2. Implement the following expressions using combination of gates:
   a) (A'+B)·B
   b) (A·B)+A'
   c) A· (B·B')
   d) (A'⊕B)·A









Experiment / Assignment / Tutorial No. 2

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Batch: B2 Roll No.: 16010121110	Experiment / assignment / tutorial No.: 2

Title: Binary Adders and Subtractors

Objective: To implement half and full adder–subtractor using gates and IC 7483

Expected Outcome of Experiment:

CO2: Use different minimization technique and solve combinational circuits, synchronous & asynchronous sequential circuits.

Books/ Journals/ Websites referred:

- **VLab Link:** http://vlabs.iitb.ac.in/vlabs-dev/labs/dldesignlab/experimentlist.html
- R. P. Jain, “Modern Digital Electronics”, Tata McGraw Hill
- M .Morris Mano, “Digital Logic & computer Design”, PHI
- http://physics.niser.ac.in/labmanuals/sem5/elect/7_ADDER%20SUBTRACTOR%20CIRCUITS.pdf

Pre Lab/ Prior Concepts:

Adder: Addition of two binary digits is most basic operation performed by the digital computer. There are two types of adder:

- Half adder
- Full adder

Half Adder: Half adder is combinational logic circuit with two inputs and two outputs. It is the basic building block for addition of two single bit numbers.

Full adder: A half adder has a provision not to add a carry coming from the lower order

bits when multi bit addition is performed. for this purpose a third input terminal is added and this circuits is to add A,B,C where A and B are the nth order bits of the number A and B respectively and C is the carry generated from the addition of (n-1) order bits. This circuit is referred to as full adder.

Subtractor: Subtraction of two binary digits is one of the most basic operations performed by digital computer .there are two types of subtractor:

- Half subtractor
- Full subtractor

Half subtractor: Logic circuit for the subtraction of B from A where A,B are 1 bit numbers is referred to as half subtract or .the subtract or process has two input and difference and borrow are the two outputs.

Full subtractor: As in the case of the addition using logic gates, a full subtractor is made by combining two half-sub tractors and an additional OR-gate. A full subtractor has the borrow in capability (denoted as BOR_{IN}) and so allows cascading which results in the possibility of multi-bit subtraction.

IC 7483

For subtraction of one binary number from another, we do so by adding 2's complement of the former to the latter number using a full adder circuit.

IC 7483 is a 16 pin, 4-bit full adder. This IC has a provision to add the carry output to transfer and end around carry output using Co and C4 respectively.

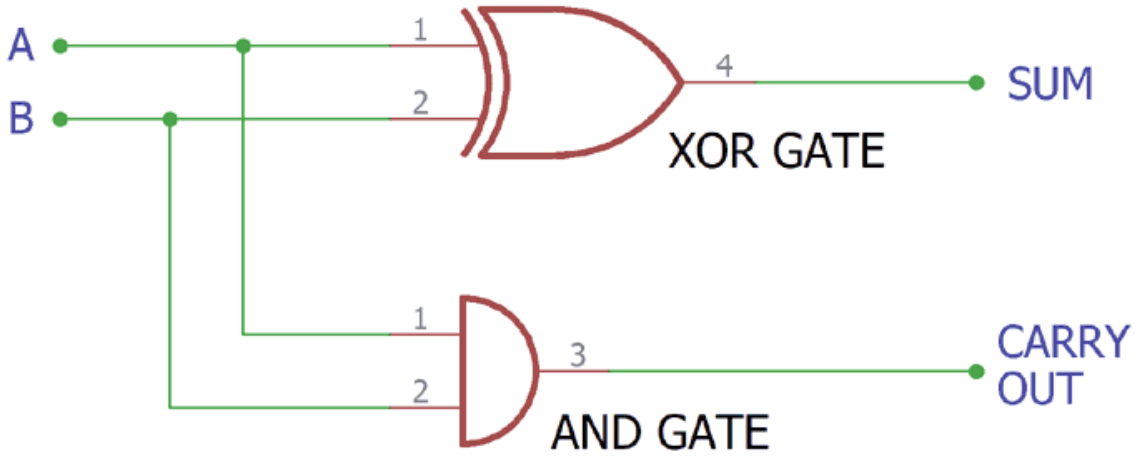
2's complement: 2's complement of any binary no. can be obtained by adding 1 in 1'scomplement of that no.

e.g. 2's complement of $+(10)_{10}$ =1010is

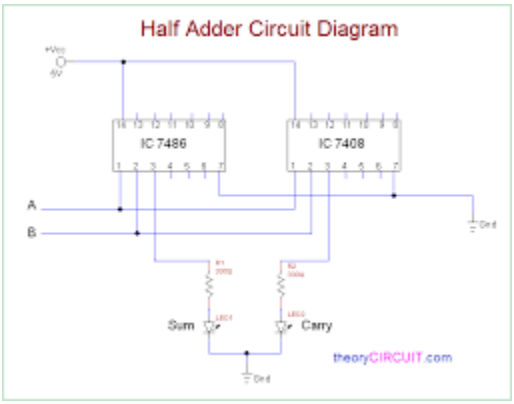
1C of 1010	0101
	+ 1
$-(10)_{10}$	0110

In 2's complement subtraction using IC 7483, we are representing negative number in 2's complement form and then adding it with 1st number.

Implementation Details:
Half Adder Block Diagram



Half Adder Circuit



Truth Table for Half Adder

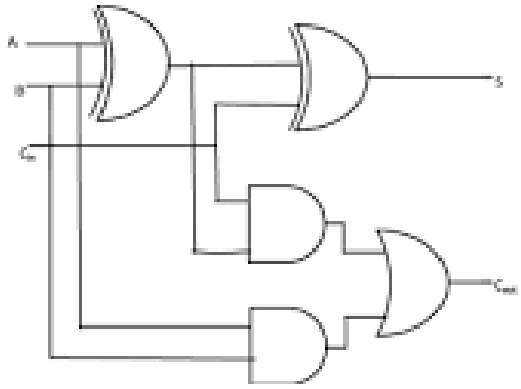
Inputs		Outputs	
A	**B**	**A**	**B**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

From the truth table (with steps):

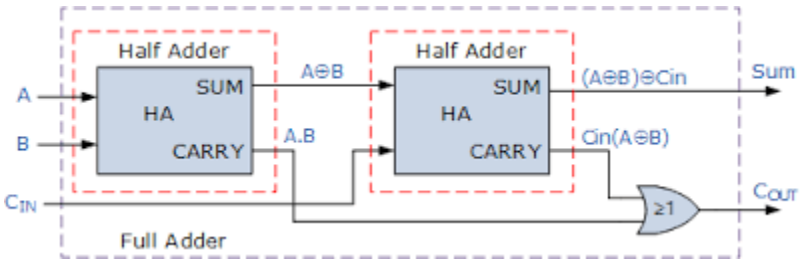
0+0=0 no carry
0+1=1 no carry
1+0=1 no carry
1+1= 2 = 1 0 = 1 carry, sum 0

sum = A xor B
Carry = A and B

Full Adder Block Diagram



Full Adder Circuit



Truth Table for Full Adder

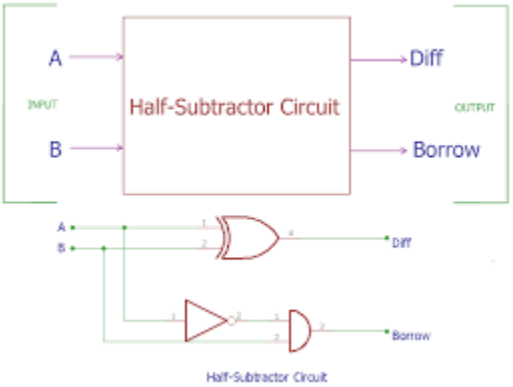
Inputs			Outputs	
A	B	C – IN	Sum	C - Out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

From the truth table (with steps):

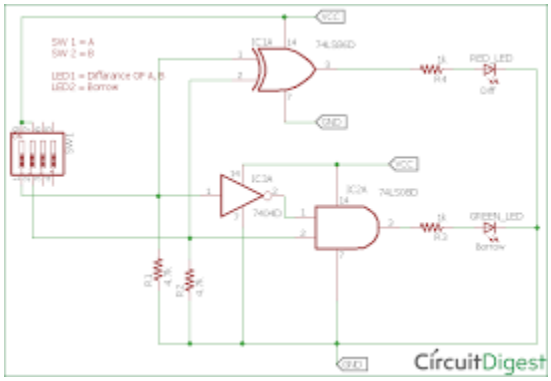
0+0+0 = 0
0+0+1 = 1
0+1+1 = 2 = 1 carry + 0 sum
0+1+0 = 1
1+1+0 = 2 1 carry + 0 sum
1+1+1 = 3 1 carry + 1 sum
1+0+0 = 1
1+0+1 = 2 1 carry + 0 sum

Sum = A' B' C-IN + A' B C-IN' + A B' C-IN' + A B C-IN
C-out = A' B C-IN + A B' C-IN + A B C-IN' + A B C-IN

Half Subtractor Block Diagram



Half Subtractor Circuit



Truth Table for Half Subtractor

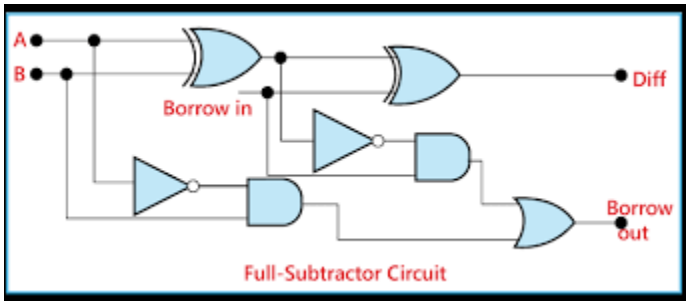
A	B	DIFFERENCE (D)	BORROW(Bo)
1	0	1	0
1	1	0	0
0	0	0	0
0	1	0	1

From the truth table (with steps) :

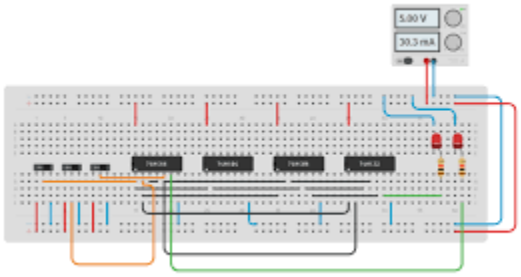
1-0=1
1-1=0
0-0=0
0-1 not possible so borrow =1 and difference=1

Diff= A'B+AB'
Borrow = A'B

Full Subtractor Block Diagram



Full Subtractor Circuit



Truth Table for Full subtractor

A	B	B_{IN}	D	BOR_{OUT}
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

From the truth table (with steps):

bout = A'Bin + A'B + BBin

diff = Bin(A'B' + AB) + Bin'(AB' + A'B)

IC 7483

Procedure:

1) Locate the IC 7483 and 4-not gates block on trainer kit.
2) Connect 1st input no. to A4-A1 input slot and 2nd (negative) no. to B4-B1 through 4-not gates (1C of 2nd no.)
3) Connect high input to Co so that it will get added with 1C of 2nd no. to get 2C.
4) Connect 4-bit output to the output indicators.
5) Switch ON the power supply and monitor the output for various input combinations.

Example:

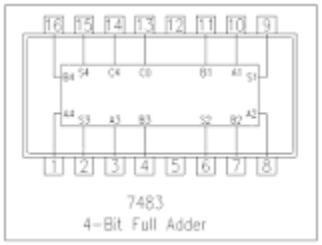
1) $7_{10} - 2_{10} = 5_{10}$

7	0111
2	0010
1'C of 2	1101
	+ 1
2'C of 2	1110

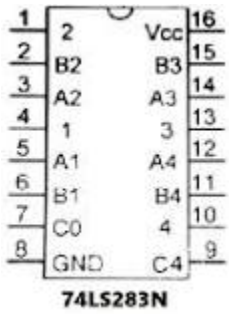
0111 +
1110 1 0101

Pin Diagram IC7483

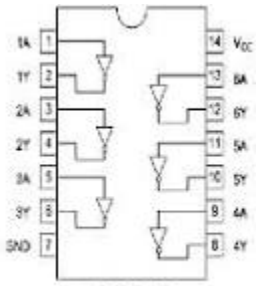
Adder



Subtractor



74LS283N



74LS04

Conclusion:

Thus we have understood the working behind half adder, full adder , half subtracter and full subtractor. We understood how the logic was implemented and how the operations worked in principle..

Post Lab Descriptive Questions

1. What is difference between half and full adder, half and full subtractor?

1. Half Adder is a combinational logic circuit that adds two 1-bit digits. The half adder produces a sum of the two inputs. A full adder is a combinational logic circuit that performs an addition operation on three one-bit binary numbers.

2. Half subtractor subtracts two numbers while full adder subtracts two numbers including borrow.

2. Perform the following Binary subtraction with the help of appropriate ICs:
   a) 7-5
   b) 5-7
   c) 9-4

Experiment / Assignment / Tutorial No. _____

Grade: AA / AB / BB / BC / CC / CD /DD

Experiment / Assignment / Tutorial No. _____

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Experiment / Assignment / Tutorial No. _____

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Batch:	Roll No.:	16010121110	Experiment / assignment / tutorial No.: 3

Title: Design 4:1 Multiplexer and 1: 4 De-multiplexer

Objective: To design and implement a 4:1 multiplexer and 1:4 de-multiplexer using logic gates and MUX IC

Expected Outcome of Experiment:

CO2: Use different minimization technique and solve combinational circuits, synchronous & asynchronous sequential circuits.

Books/ Journals/ Websites referred:

- VLab Links: http://vlabs.iitb.ac.in/vlabs-dev/labs/dldesignlab/experimentlist.html
- R. P. Jain, “Modern Digital Electronics”, Tata McGraw Hill
- M .Morris Mano, “Digital Logic & computer Design”, PHI
- https://wiki.engr.illinois.edu/download/attachments/84770821/08-Multiplexers.pdf?version=2&modificationDate=1285128827000

Pre Lab/ Prior Concepts:

Multiplexer: Multiplexer is a special type of combinational circuit. It is a digital circuit which selects one of the n data inputs and routes it to the output. The selection of one of the n inputs is done by the select lines. To select n inputs we require m select lines, such that 2^m=n. Depending on the digital code applied at the select inputs, one out of the n data sources is selected and transmitted to a single output . E is called as the strobe or enable input which is useful for cascading. It is generally on active low terminal that means it will perform the required operation when it is low. The multiplexer act like a digitally controlled single pole, multiple way switches. The output gets

connected to only one input at a time. In most of the electronic system the digital data is available on more than one line. It is necessary to route the data over a single line, under such circumstances input at a time

Types of Multiplexer:

1. 2:1 Multiplexer
2. 4:1 Multiplexer
3. 8:1 Multiplexer
4. 16:1 Multiplexer
5. 32:1 Multiplexer

De-multiplexer: It has only one input, n output and m select lines. A demultiplexer performs the reverse operation of a multiplexer i.e. it receives one input and distributes it over several outputs. The demultiplexer converts a serial data signal at the input to a parallel data at its output lines. The relation between the output lines and select lines is as follows: $N=2^m$

Types of Demultiplexers:

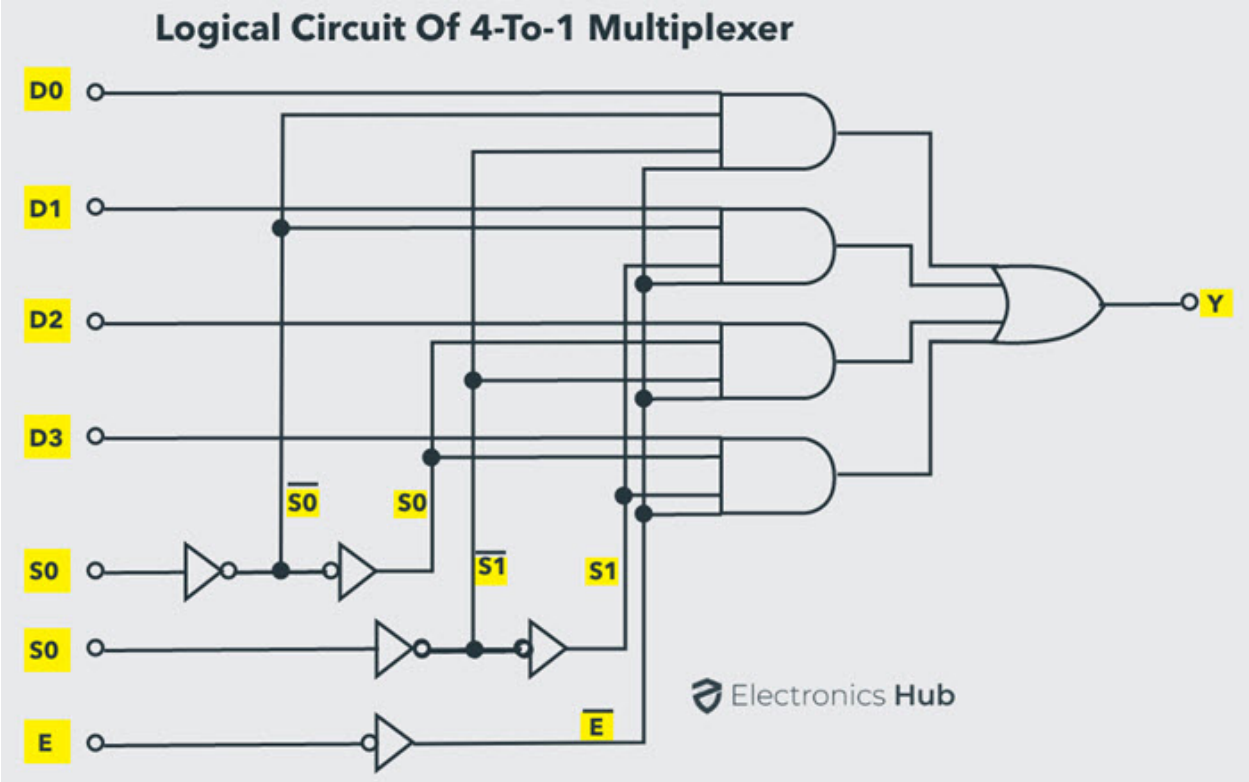
1. 1:2 DEMUX
2. 1:4 DEMUX
3. 1:8 DEMUX
4. 1:16 DEMUX

Implementation Details of 4:1 MUX

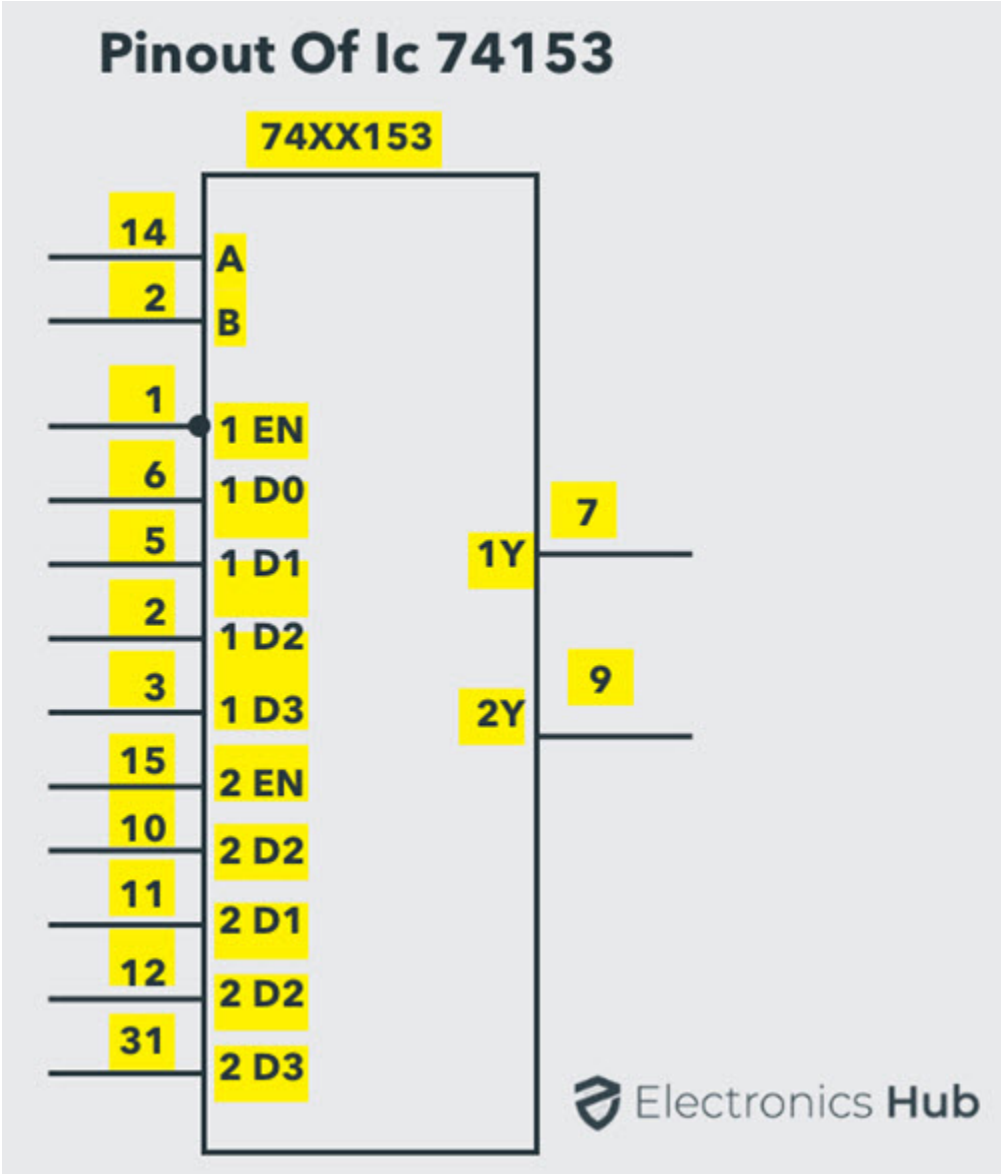
A 4-to-1 multiplexer can be implemented by using basic logic gates. The below figure shows the logic circuit of 4:1 MUX which is implemented by four 3-inputs AND gates, two 1-input NOT gates, and one 4-inputs OR gate.

In this circuit, each data input line is connected as input to an AND gate and two select lines are connected as other two inputs to it. Additionally, there is also an Enable Signal. The output of all the AND gates are connected to inputs of OR gate in order to produce the output Y.

Block Diagram of 4:1 MUX



Circuit Diagram of 4:1 MUX


Pinout Of Ic 74153
74XX153
14
A
2
B
1
1 EN
6
1 D0
5
1 D1
2
1 D2
3
1 D3
15
2 EN
10
2 D2
11
2 D1
12
2 D2
31
2 D3
7
1Y
9
2Y
Electronics Hub

Truth table

S0	S1	D0	D1	D2	D3	Y
0	0	0	X	X	X	0
0	0	1	X	X	X	1
0	1	X	0	X	X	0
0	1	X	1	X	X	1
1	0	X	X	0	X	0
1	0	X	X	1	X	1
1	1	X	X	X	0	0
1	1	X	X	X	1	1

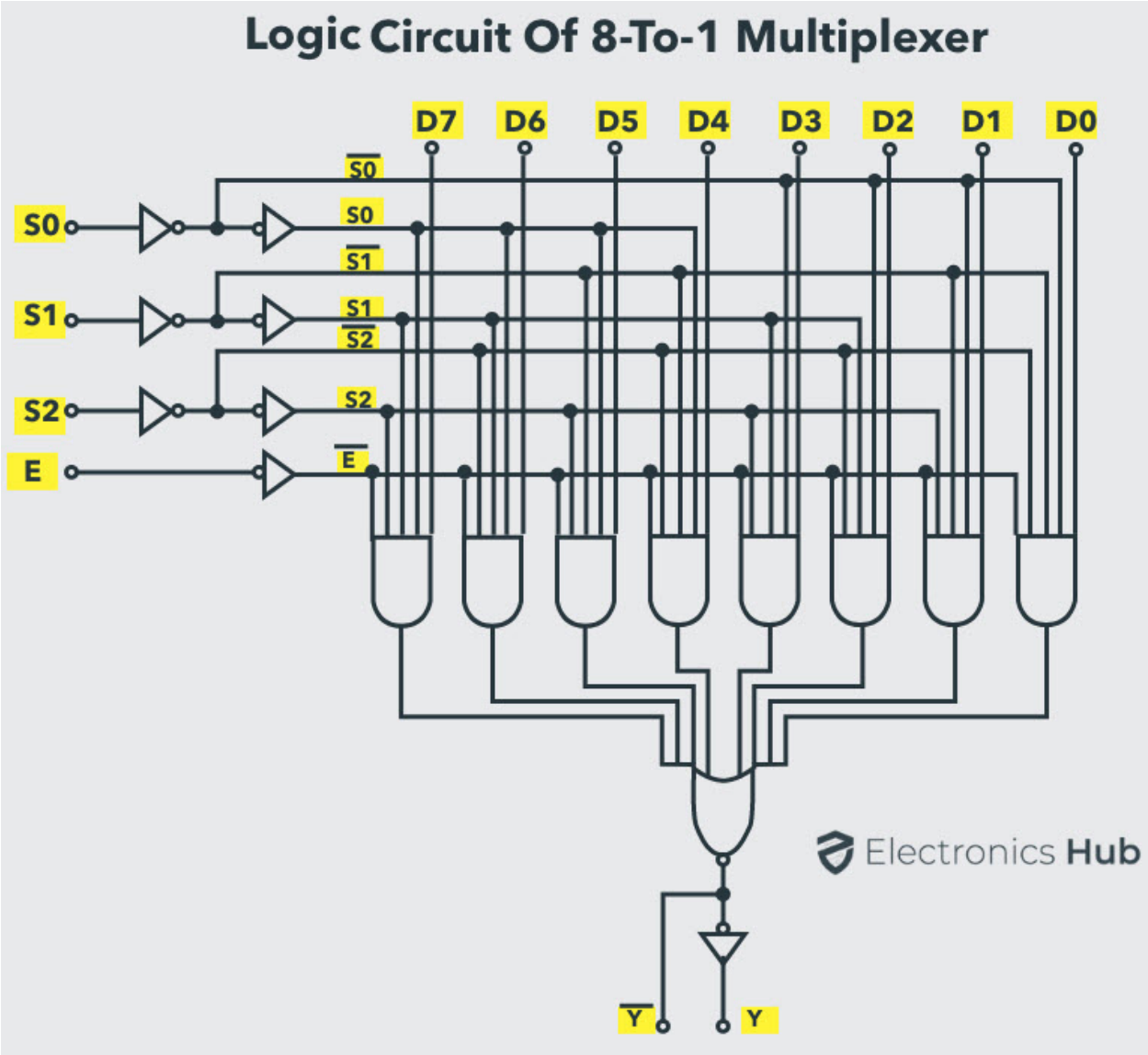
From Truth Table:

$$Y = S0\ S1\ D0 + S0\ S1\ D1 + S0\ S1\ D2 + S0\ S1\ D3$$

Implementation Details of 8:1 MUX

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

Circuit Diagram of 8:1 MUX


Logic Circuit Of 8-To-1 Multiplexer
D7
D6
D5
D4
D3
D2
D1
D0
S0
S1
S2
E
Y
Electronics Hub

Truth Table for 8:1 Multiplexer

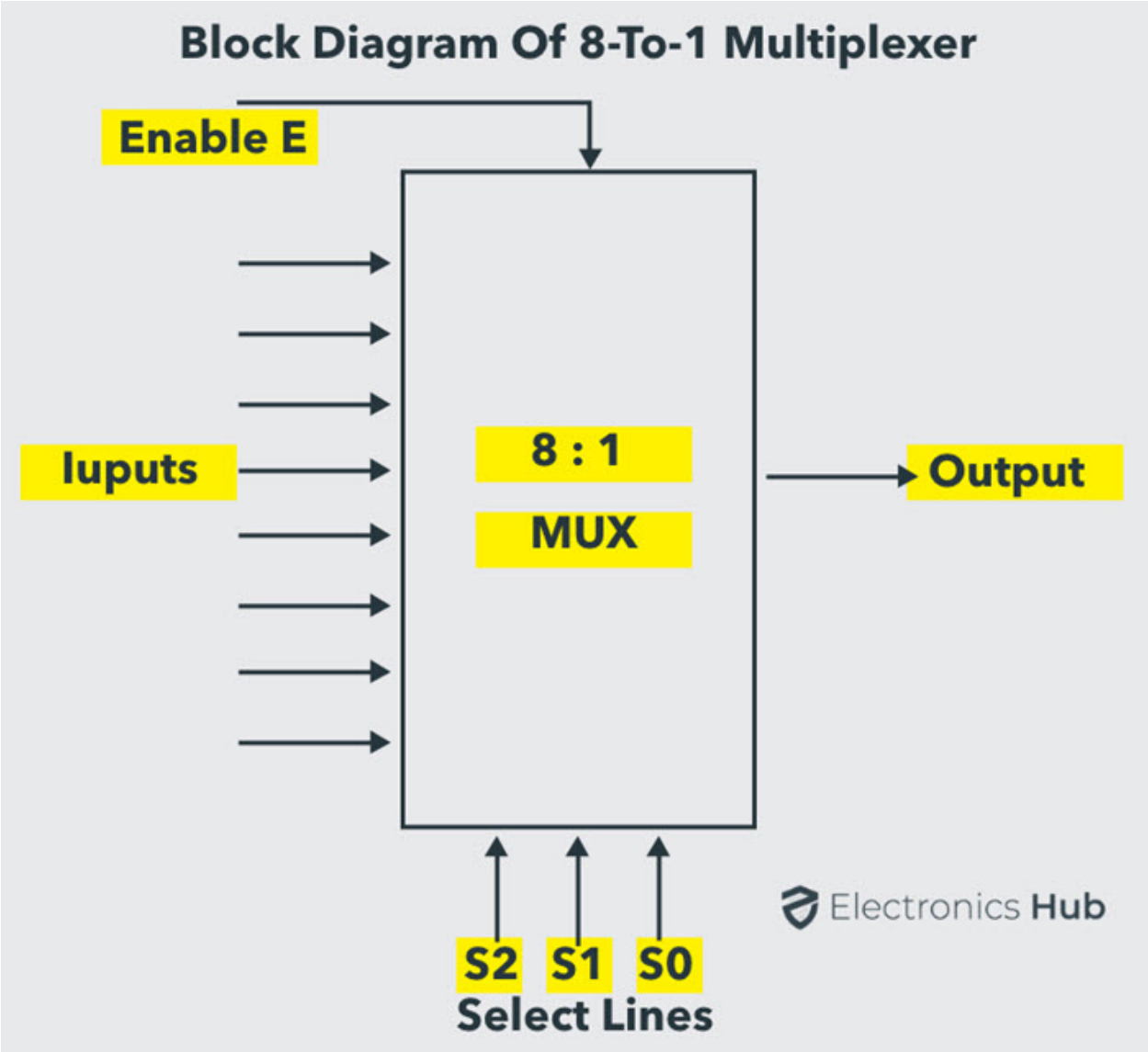
S0	S1	S2	D0	D1	D2	D3	D4	D5	D6	D7	Y
0	0	0	0	X	X	X	X	X	X	X	0
0	0	0	1	X	X	X	X	X	X	X	1
0	0	1	X	0	X	X	X	X	X	X	0
0	0	1	X	1	X	X	X	X	X	X	1
0	1	0	X	X	0	X	X	X	X	X	0
0	1	0	X	X	1	X	X	X	X	X	1
0	1	1	X	X	X	0	X	X	X	X	0
0	1	1	X	X	X	1	X	X	X	X	1
1	0	0	X	X	X	X	0	X	X	X	0
1	0	0	X	X	X	X	1	X	X	X	1

1	0	1	X	X	X	X	X	0	X	X	0
1	0	1	X	X	X	X	X	1	X	X	1
1	1	0	X	X	X	X	X	X	0	X	0
1	1	0	X	X	X	X	X	X	1	X	1
1	1	1	X	X	X	X	X	X	X	0	0
1	1	1	X	X	X	X	X	X	X	1	1

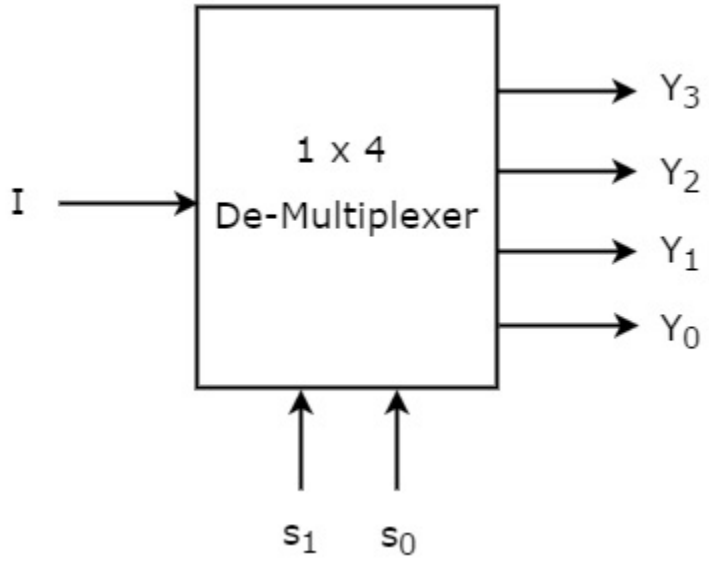
From Truth Table:

Y = S0 S1 S2 D0 + S0 S1 S2 D1 + S0 S1 S2 D2 + S0 S1 S2 D3 + S0 S1 S2 D4 + S0 S1 S2 D5 + S0 S1 S2 D6 + S0 S1 S2 D7

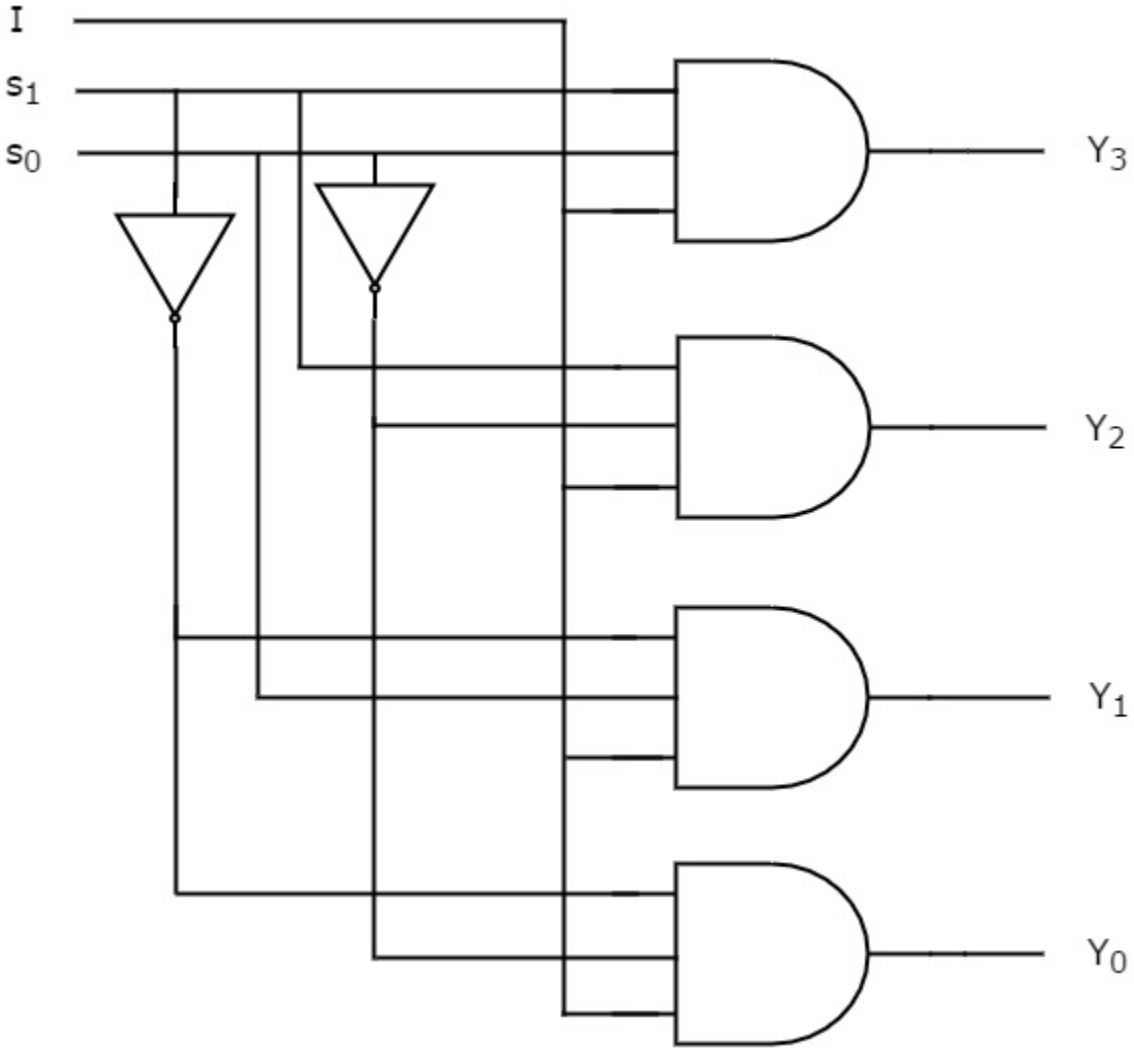
Pin diagram: IC 74151


Block Diagram Of 8-To-1 Multiplexer
Enable E
Iuputs
8 : 1
MUX
Output
Electronics Hub
S2
S1
S0
Select Lines

Block Diagram of 1:4 DE MUX



Circuit Diagram of 1:4 DE MUX



Truth Table for 1:4 Demultiplexers

S_1	S_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	I
0	1	0	0	I	0

1	0	0	I	0	0
1	1	I	0	0	0

From Truth Table:
Y0 = S D

Y1 = S D

Conclusion:

Thus we have understood the principle behind multiplexer and demultiplexer. We implemented the multiplexer and demultiplexers using basic gates and understood the working.

Post Lab Descriptive Questions

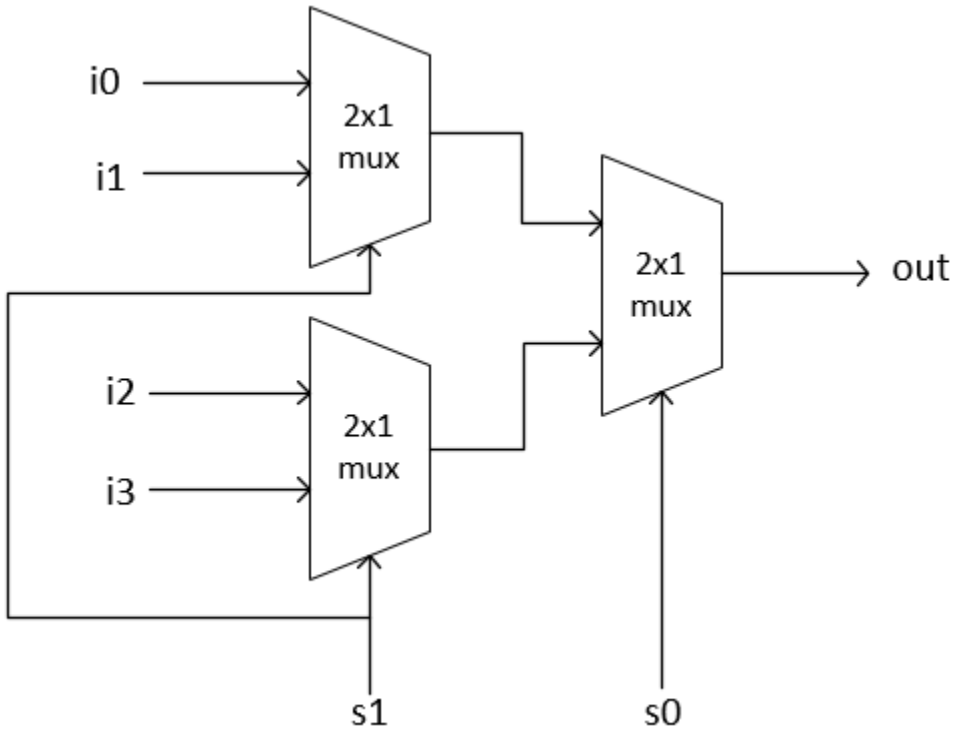
1. How many select lines are required for 64:1 MUX?

   2^6 is 64 so 6 lines are required.

2. State some applications of MUX and DEMUX.

Mux and demux both are **used in communication systems to carry out the process of data transmission**. A De-multiplexer receives the output signals from the multiplexer and at the receiver end, it converts them back to the original form.

3. Build a 4:1 MUX using only 2:1 MUX.



Credits:
"Design of 4×2 Multiplexer Using 2×1 Mux in Verilog | Brave Learn." *Design of 4×2 Multiplexer Using 2×1 Mux in Verilog | Brave Learn*, bravelearn.com/design-of-4x2-multiplexer-using-2x1-mux-in-verilog. Accessed 15 Sept. 2022.

Experiment / Assignment / Tutorial No. _____

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Batch:B2 Roll No.: 1601012110 Experiment / assignment / tutorial No.: 4

Title: 4 bit Magnitude Comparator

Objective: Design a 2-bit comparator using logic gates and verify 4-bit magnitude comparator using IC 7485

Expected Outcome of Experiment:

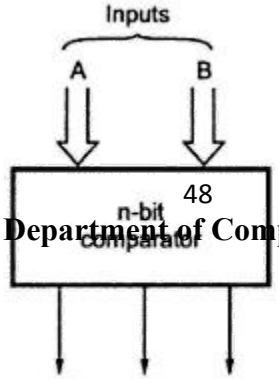
CO2: Use different minimization technique and solve combinational circuits, synchronous & asynchronous sequential circuits.

Books/ Journals/ Websites referred:

- VLab Link: http://vlabs.iitb.ac.in/vlabs-dev/labs/dldesignlab/experimentlist.html
- R. P. Jain, “Modern Digital Electronics”, Tata McGraw Hill
- M .Morris Mano, “Digital Logic & computer Design”, PHI
- http://elnsite.teilam.gr/ebooks/digital_design/lab/dataSheets_page/7485.pdf

Pre Lab/ Prior Concepts:

The comparison of two numbers is an operator that determines one number is greater than, less than (or) equal to the other number. A magnitude comparator is a combinational circuit that compares two numbers A and B and determines their relative magnitude. The outcome of the comparator is specified by three binary variables that indicate whether A>B, A=B (or) A<B.



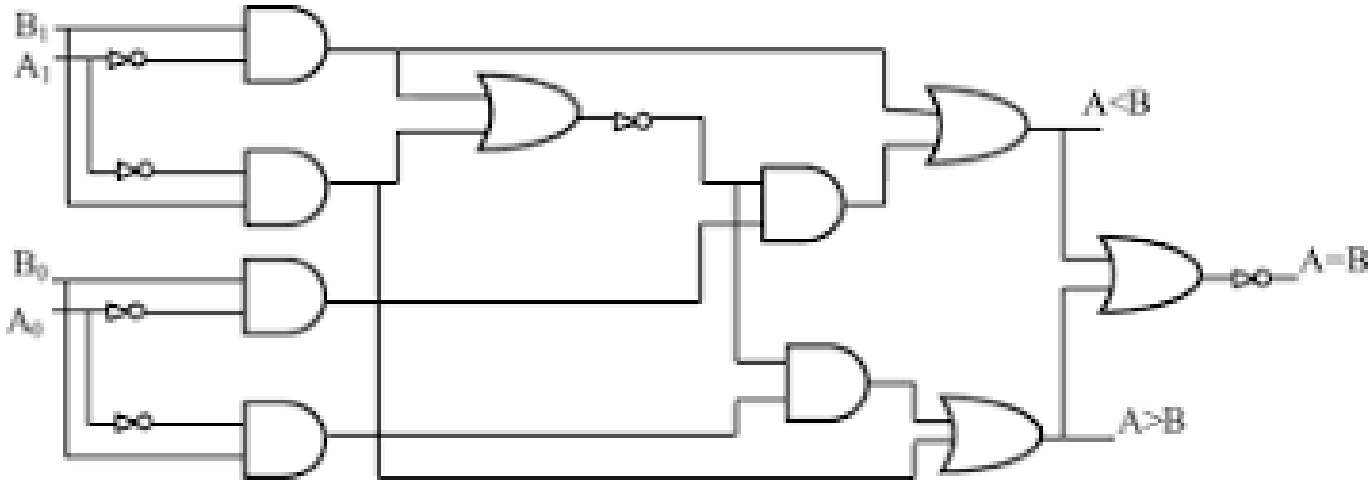
Two Bit Magnitude Comparator Implementation Details:

Inputs		Outputs		
B	A	A > B	A = B	A < B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

From the Truth Table:

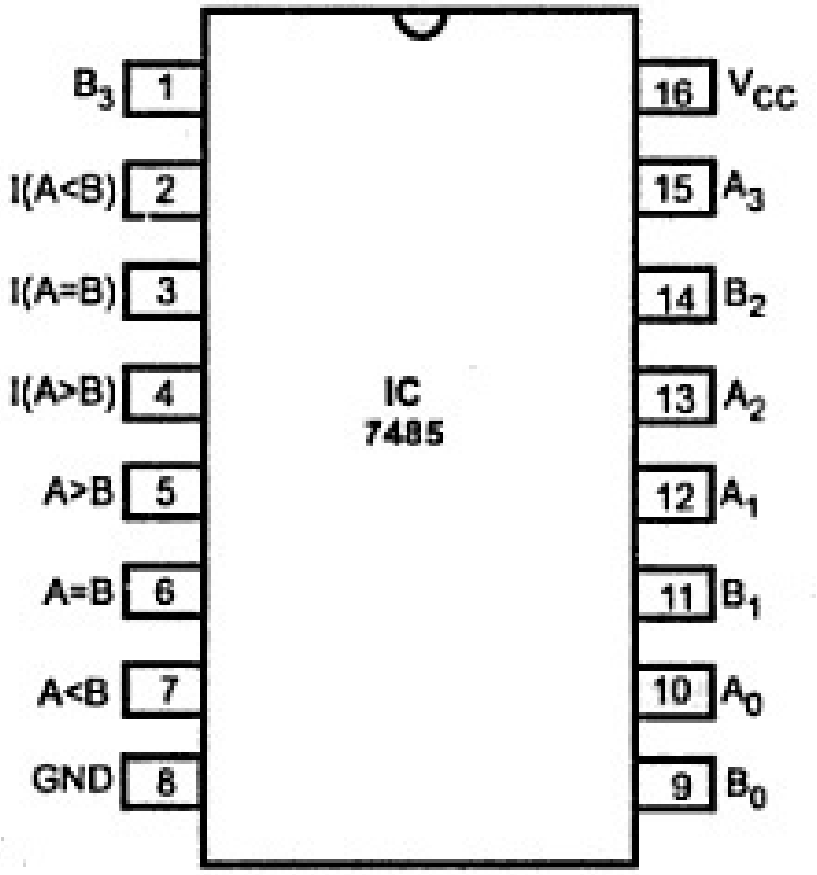
A>B: AB'
A<B: A'B
A=B: A'B' + AB

Logic Diagram of 2 bit Comparator



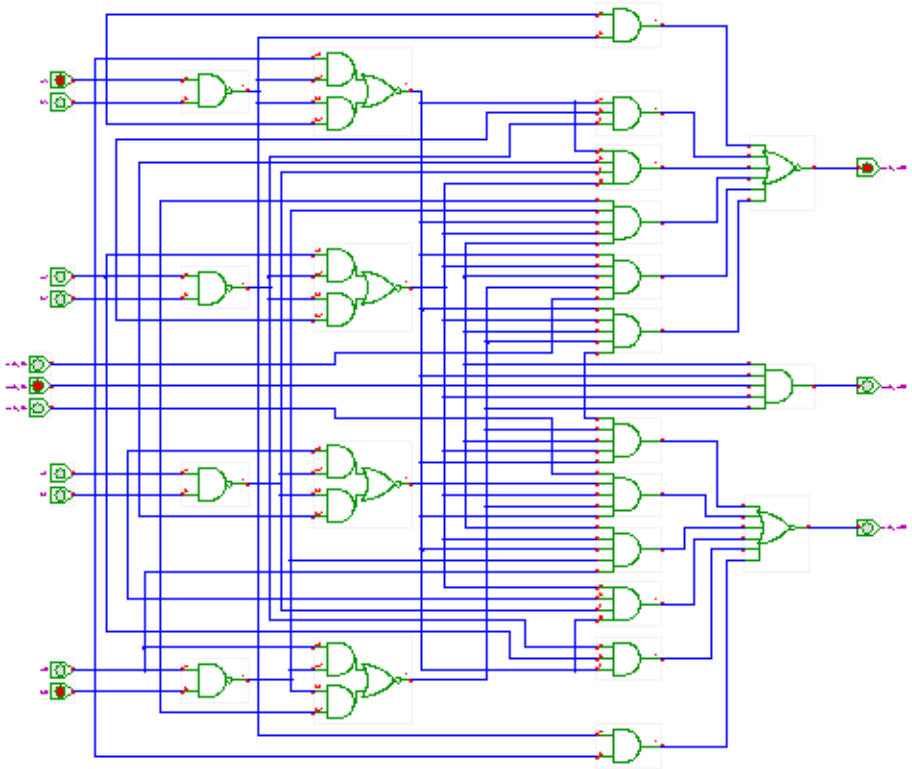
Four Bit Magnitude Comparator Implementation Details

Pin Diagram of IC 7485



(a) Pin diagram (IC 7485)

Logic Diagram of IC 7485



Comparing Table

COMPARING INPUTS				OUTPUT		
A3, B3	**A2, B2**	**A1, B1**	**A0, B0**	**A > B**	**A < B**	**A = B**
A3 > B3	X	X	X	H	L	L
A3 < B3	X	X	X	L	H	L
A3 = B3	A2 >B2	X	X	H	L	L
A3 = B3	A2 < B2	X	X	L	H	L
A3 = B3	A2 = B2	A1 > B1	X	H	L	L
A3 = B3	A2 = B2	A1 < B1	X	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 > B0	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 < B0	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	H
H = High Voltage Level, L = Low Voltage, Level, X = Don't Care						

Conclusion:

In this experiment, we have understood how comparators work. We designed various comparators, for example 2 bit and 4 bit comparators were designed by us. We understood the working behind the comparators and implemented them in the circuit using ICs.

Post Lab Descriptive Questions

1. Design a 1- bit magnitude comparator using logic gates.

A	B	A=B	A>B	A<B
1	1	1	0	0
1	0	0	1	0
0	1	0	0	1
0	0	1	0	0

From the truth table

A=B : (A Xor B)'

A>B: A*B'

A<B: A'*B

Experiment / Assignment / Tutorial No. _____
Grade: AA / AB / BB / BC / CC / CD /DD
Signature of the Staff In-charge with date

Batch:B2 Roll No.: 16010121110	Experiment / assignment / tutorial No.: 5

Title: Flip Flops

Objective: Design of JK Flip flop, D flip flop, T flip flop using NAND Gates & verification of the same flip flop using IC7476

Expected Outcome of Experiment:

CO2: Use different minimization technique and solve combinational circuits, synchronous & asynchronous sequential circuits.

Books/ Journals/ Websites referred:

- VLab Link: http://vlabs.iitkgp.ernet.in/dec/#
- R. P. Jain, “Modern Digital Electronics”, Tata McGraw Hill
- M .Morris Mano, “Digital Logic & computer Design”, PHI
- A.P.Godse, D.A.Godse, “Digital Logic Design”

Pre Lab/ Prior Concepts:

Flip-flop is the common name given to two-state devices which offer basic memory for sequential logic operations. Flip-flops are heavily used for digital data storage and transfer and are commonly used in banks called "registers" for the storage of binary numerical data.

JK-flip flop: has two inputs, traditionally labeled J and K. IC 7476 is a dual JK master slave flip flop with preset and clear inputs. If J and K are different then the output Q

takes the value of J at the next clock edge. If J and K are both low then no change occurs. If J and K are both high at the clock edge, then the output will toggle from one state to the other. It can perform the functions of the set/reset flip-flop and has the advantage that there are no ambiguous states.

D Flip Flop: tracks the input, making transitions with match those of the input D. The D stands for "data"; this flip-flop stores the value that is on the data line. It can be thought of as a basic memory cell. D flip-flop can be made from J-K flip-flop by connecting both inputs through a not gate.

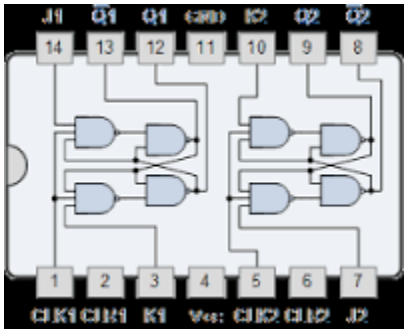
T Flip Flop: T or "toggle" flip-flop changes its output on each clock edge, giving an output which is half the frequency of the signal to the T input. It is useful for constructing binary counters, frequency dividers, and general binary addition devices. It can be made from a J-K flip-flop by tying both of its inputs high.

Implementation Details:

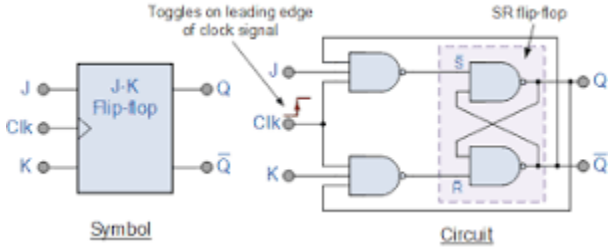
Procedure

1) Locate IC 7476 on Digital trainer kit
2) Apply various inputs to J & K pins by means of the output on logic output indicator.
3) Connect a pulsar switch to the clock input.
4) Connect the J&K as D and T flip flop as shown in diagrams and verify the respective truth tables.

Pin Diagram of IC 7476 JK Master- Slave FF



Logic Symbol



Truth Table

J	K	Q_n	Q_{n+1}	State
0	0	0	0	Q_n (Hold)
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	1	Toggel
1	1	1	0	

JKFF

D FF **Truth Table**

D	O/P
0	0
1	1

TFF **Truth Table**

T	O/P
0	1
1	0

Diagram of JK Flip Flop using NAND gates

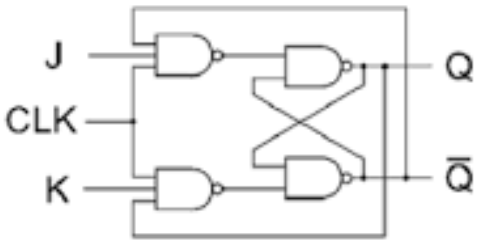


Figure 4: JK Flip Flop

Conclusion:

Thus we have implemented JK flip flop using basic nand gates s well as the Ic. We have understood the working behind the JK flip flop and how they are used for storing bits. We also understood the types of flipflops

Post Lab Descriptive Questions

1. How does a JK flip-flop differ from an SR flip-flop in its basic operation?

The only difference between JK flip flop and SR flip flop is that when both inputs of SR flip flop is set to 1, the circuit produces the invalid states as outputs, but in case of JK flip flop, there are no invalid states even if both 'J' and 'K' flip flops are set to 1

2. What is use of characteristic and excitation table?

Excitation table is used for design of flip-flops and counters. Truth table contains inputs and excitation table takes outputs as inputs. A characteristic table has the control input (D or T) as the first column, the current state as the middle column, and the next state as the last column

3. How many flip flops due you require storing the data 1101?

1 flipflop per bit so total of 4 flipflops

4. Describe the basic difference between pulse-triggered and edge-triggered flip-flops.

In pulse triggered flip flops only one latch is used whereas it is two in normal edge triggered flip flops. The only type of flip flops which has time borrowing capability with negative set-up time is pulse triggered flip flops

.

Experiment / Assignment / Tutorial No. _____
Grade: AA / AB / BB / BC / CC / CD /DD
Signature of the Staff In-charge with date

Batch:B2	Roll No.: 110	Experiment / assignment / tutorial No.: 6

Title: Shift Register

Objective: To implement the SISO, SIPO, PISO, PIPO shift register using D flips flop

Expected Outcome of Experiment:

CO2: Use different minimization technique and solve combinational circuits, synchronous & asynchronous sequential circuits.

Books/ Journals/ Websites referred:

- VLab Link: http://vlabs.iitkgp.ernet.in/dec/#
- R. P. Jain, “Modern Digital Electronics”, Tata McGraw Hill
- M .Morris Mano, “Digital Logic & computer Design”, PHI
- A.P.Godse, D.A.Godse, “Digital Logic Design”

Pre Lab/ Prior Concepts:

A register is capable of shifting its binary information in one or both directions is known as shift register. The logical configuration of shift register consist of a D-Flip flop cascaded with output of one flip flop connected to input of next flip flop. All flip flops receive common clock pulses which causes the shift in the output of the flip flop.The simplest possible shift register is one that uses only flip flop. The output of a given flip flop is connected to the input of next flip flop of the register. Each clock pulse shifts the content of register one bit position to right.

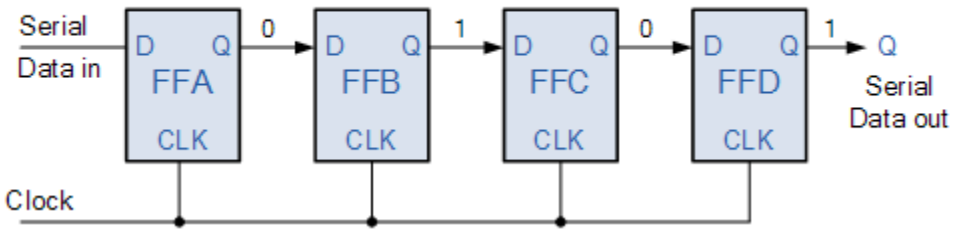
The basic types of shift registers are

- Serial In - Serial Out
- Serial In - Parallel Out
- Parallel In - Serial Out
- Parallel In - Parallel Out
- Bidirectional shift registers.

Implementation Details:

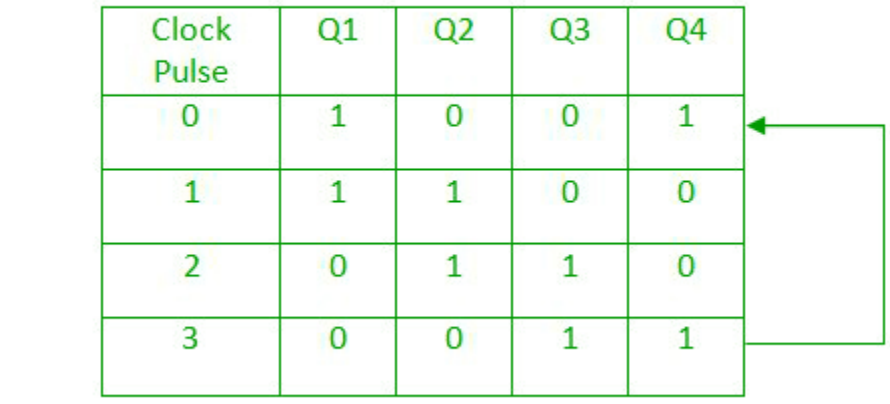
Logic Diagram

Serial in Serial Out

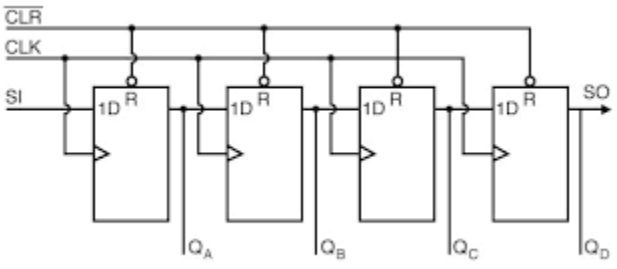


Truth table

Clock Pulse	Q1	Q2	Q3	Q4
0	1	0	0	1
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1



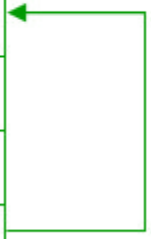
Serial In - Parallel Out



Serial-in/ Parallel-out shift register details

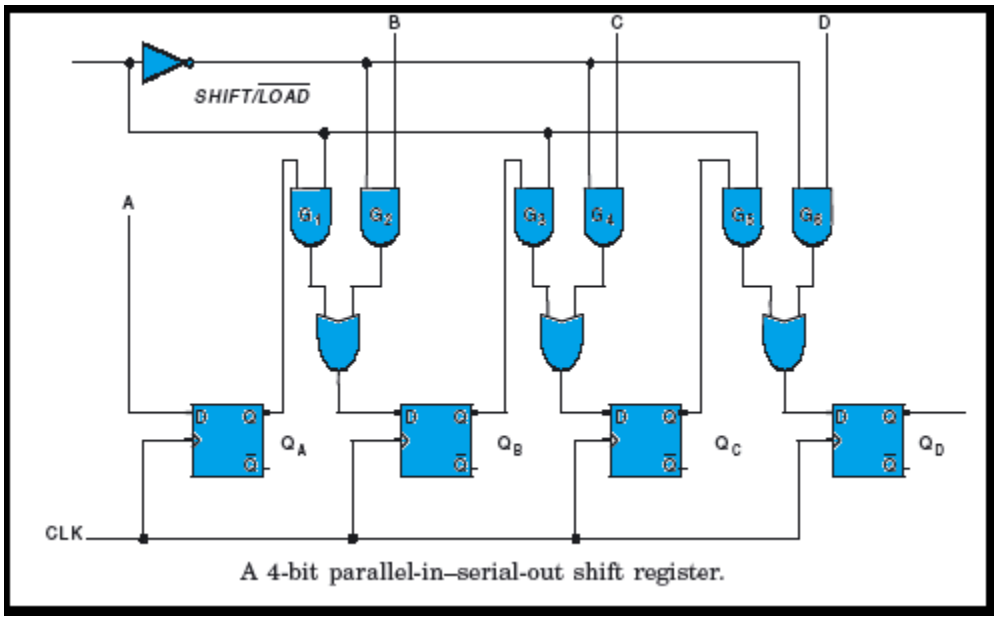
Truth table

Clock Pulse	Q1	Q2	Q3	Q4
0	1	0	0	1
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1



Note this is same TT as serial in Serial out, only difference is that Q1 Q2 Q3 Q4 are outputs.

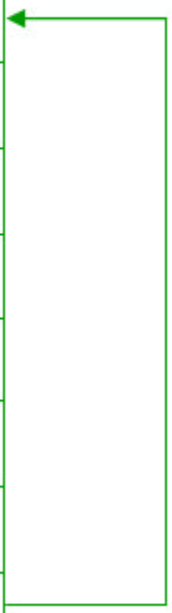
Parallel In Serial Out



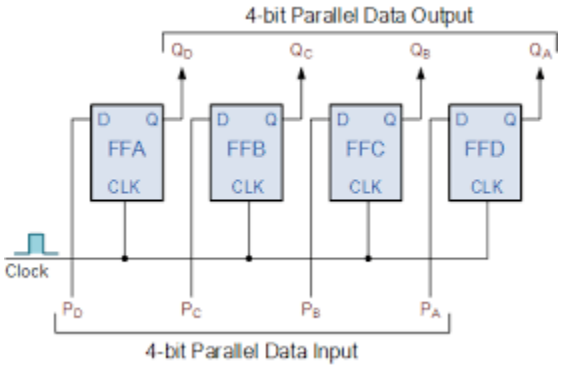
A 4-bit parallel-in–serial-out shift register.

Truth table

Clock Pulse	Q1	Q2	Q3	Q4
0	0	0	0	1
1	0	0	0	0
2	1	0	0	0
3	1	1	0	0
4	1	1	1	0
5	1	1	1	1
6	0	1	1	1
7	0	0	1	1

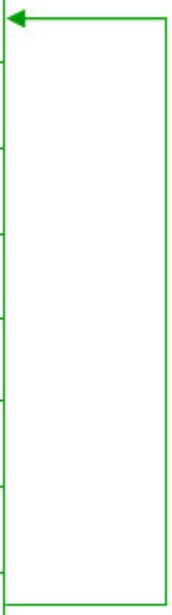


Parallel In Parallel Out


4-bit Parallel Data Output
Q_D
Q_C
Q_B
Q_A
D Q
FFA
CLK
D Q
FFB
CLK
D Q
FFC
CLK
D Q
FFD
CLK
Clock
P_D
P_C
P_B
P_A
4-bit Parallel Data Input

Truth table

Clock Pulse	Q1	Q2	Q3	Q4
0	0	0	0	1
1	0	0	0	0
2	1	0	0	0
3	1	1	0	0
4	1	1	1	0
5	1	1	1	1
6	0	1	1	1
7	0	0	1	1

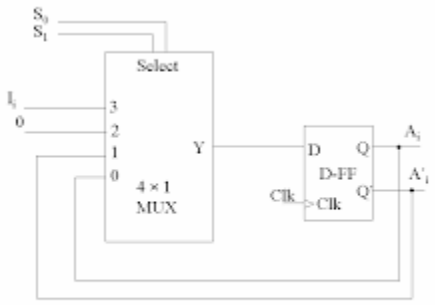


Conclusion:

Thus we have understood the working of shift registers. we studied four types of registers, parallel in parallel out, serial in serial out, serial in parallel out, parallel in serial out.
These shift registers are made up of comparators.

Post Lab Descriptive Questions

1. Draw logic diagram for universal shift register using 4:1 MUX.



2. Develop the logic diagram for the shift register using JK flip-flop to replace the D flip flop?
   Put inverted input in D by using a not gate
3. How many clock pulses are required to enter a byte of data serially into an 8-bit shift register?
   8

Experiment / Assignment / Tutorial No. 7

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Batch: B2	**Roll No.:** 110	**Experiment / assignment / tutorial No.: 7**

Title: To simulate OR gate using VHDL

Objective:

To Implement digital networks using VHDL. To simulate OR gate using VHDL

Expected Outcome of Experiment:

CO4: Implement digital networks using VHDL.

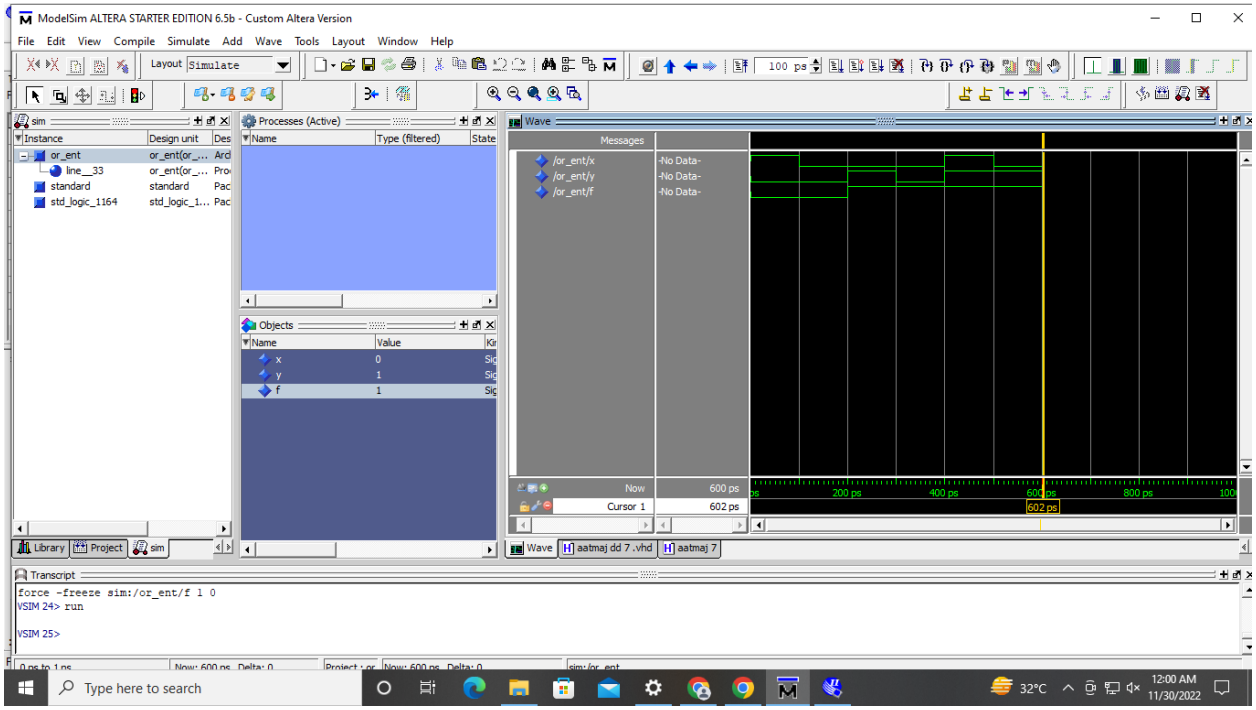
Books/ Journals/ Websites referred:

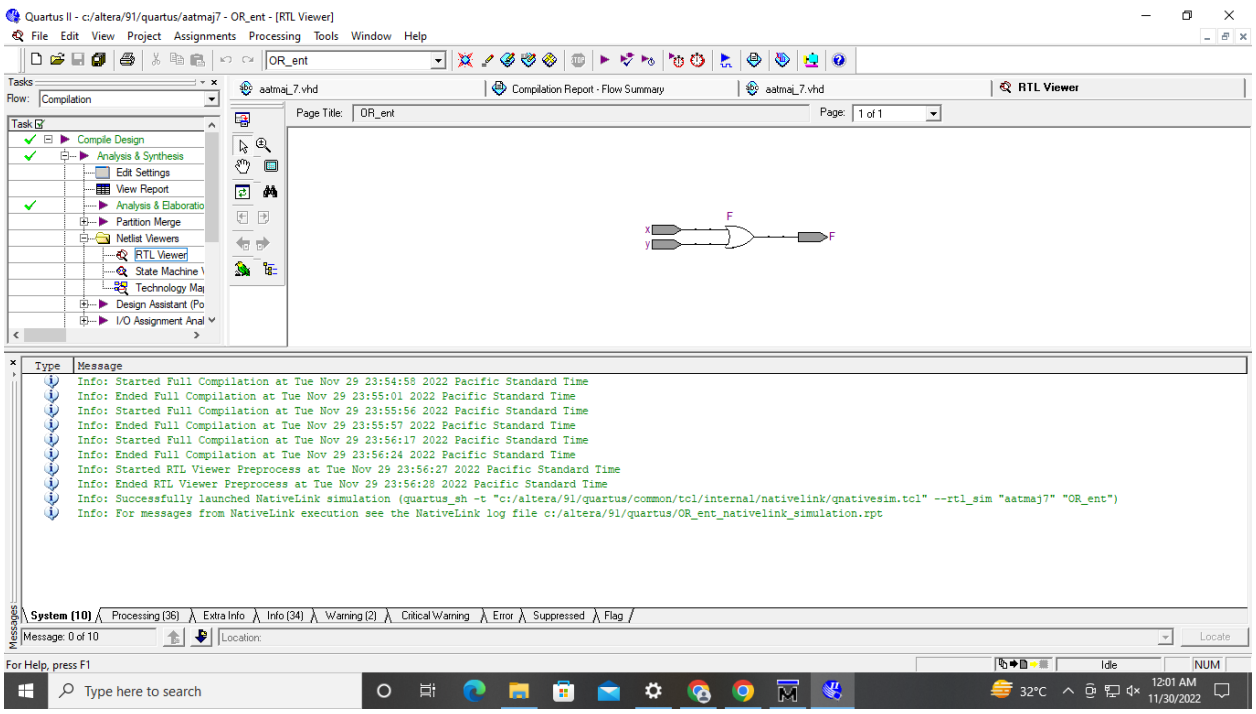
- ModelSim Software Link: https://www.mentor.com/company/higher_ed/modelsim-student-edition
- J. Bhasker, “VHDL Primer”, Pearson Education
- Douglas L. Perry, “VHDL Programming by Example”, Tata McGraw Hill

- http://esd.cs.ucr.edu/labs/tutorial/

Pre Lab/ Prior Concepts:

Gate is a logic circuit with one or more inputs but only one output. Gates are digital (two state) circuit because the input & output are either low or high. Gates provide high output for certain combinations of input & for other combinations the output is low. Total number of combinations for a gate is 2^n; where n is number of input.





VHDL Programming Structure

```
library ieee;
use ieee.std_logic_1164.all;
```

```
entity OR_ent is
port( x: in std_logic;
      y: in std_logic;
      F: out std_logic
);
end OR_ent;



architecture OR_arch of OR_ent is
begin

    process(x, y)
    begin
        -- compare to truth table
        if ((x='0') and (y='0')) then
          F <= '0';
      else
          F <= '1';
      end if;
    end process;

end OR_arch;

architecture OR_beh of OR_ent is
begin

    F <= x or y;

end OR_beh;
```

Conclusion:

Thus we implemented gates in VHDL. We also understood the VHDL structure and coding.

Post Lab Descriptive Questions

1. What are two types of HDL?

The two most widely used and well-supported HDL varieties used in industry are **Verilog and VHDL**

Experiment / Assignment / Tutorial No. __8_____

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Batch:	**B2**	**Roll No.:**	**110**	**Experiment / assignment / tutorial No.: 8**

Title: 4:1 Mux in VHDL

Objective: Design of 3 bit asynchronous counter using JK flip flop in VHDL

Expected Outcome of Experiment:

CO2: Use different minimization technique and solve combinational circuits, synchronous & asynchronous sequential circuits.

CO4: Implement digital networks using VHDL

Books/ Journals/ Websites referred:

- VLab Links: http://vlabs.iitb.ac.in/vlabs-dev/labs/dldesignlab/experimentlist.html
- R. P. Jain, “Modern Digital Electronics”, Tata McGraw Hill
- M .Morris Mano, “Digital Logic & computer Design”, PHI
- https://wiki.engr.illinois.edu/download/attachments/84770821/08-Multiplexers.pdf?version=2&modificationDate=1285128827000

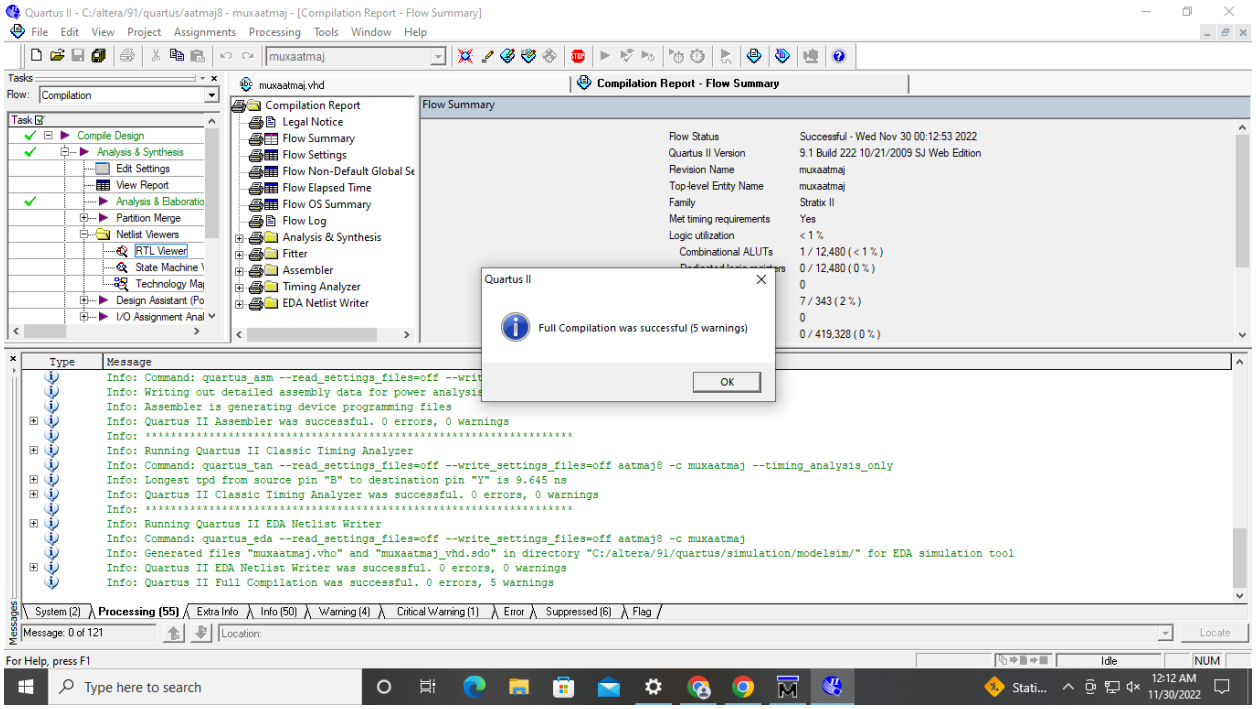
Pre Lab/ Prior Concepts:

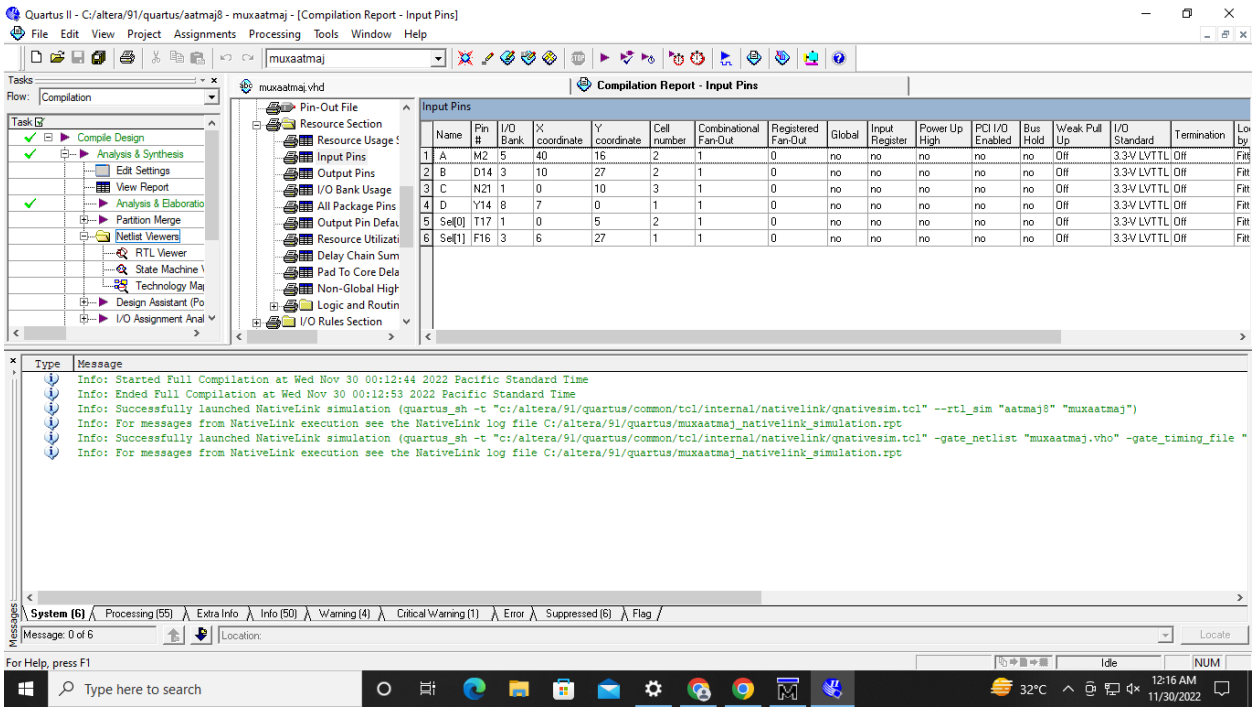
Multiplexer: Multiplexer is a special type of combinational circuit. It is a digital circuit which selects one of the n data inputs and routes it to the output. The selection of one of the n inputs is done by the select lines. To select n inputs we require m select lines, such that 2^m=n. Depending on the digital code applied at the select inputs, one out of the n data sources is selected and transmitted to a single output . E is called as the strobe or enable input which is useful for cascading. It is generally on active low terminal that means it will perform the required operation when it is low. The multiplexer act like a digitally controlled single pole, multiple way switches. The output gets connected to only one input at a time. In most of the electronic system the digital data is available on more than one line. It is necessary to route the data over a single line, under such circumstances input at a time

Types of Multiplexer:

6. 2:1 Multiplexer
7. 4:1 Multiplexer
8. 8:1 Multiplexer
9. 16:1 Multiplexer
10. 32:1 Multiplexer

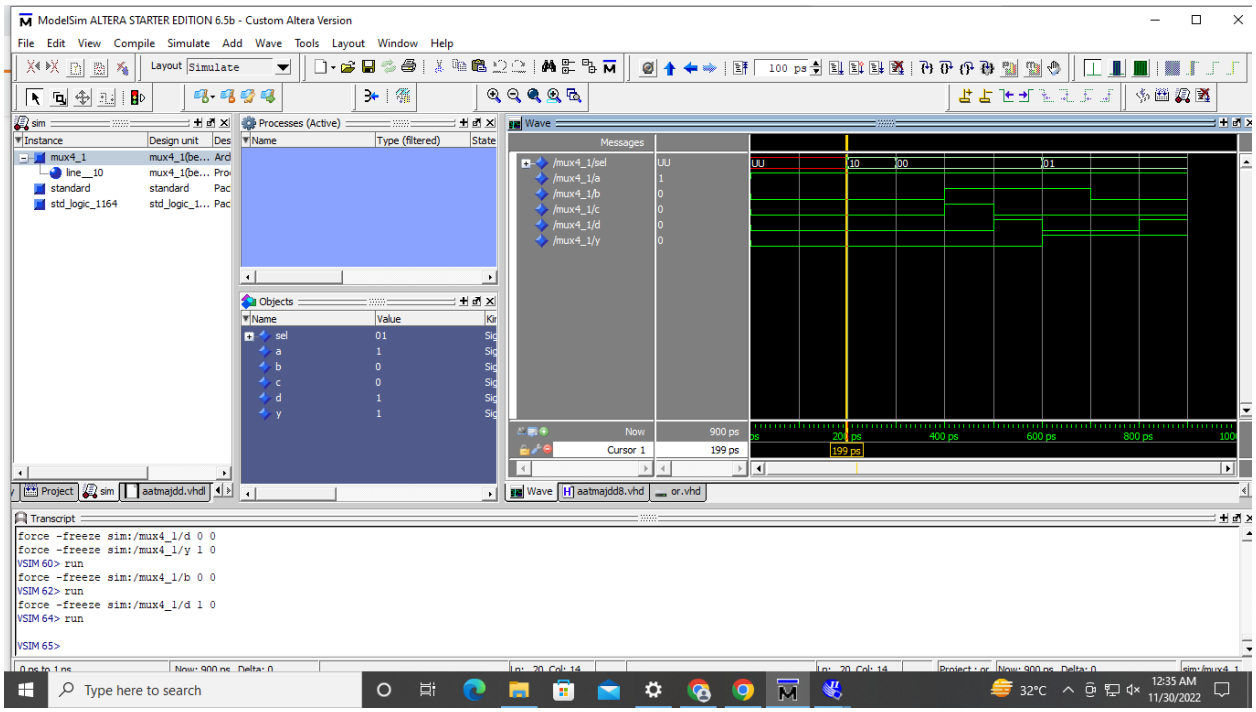
Implementation Details of 4:1 MUX






```
library ieee;
use ieee.std_logic_1164.all;
entity MUX4_1 is
```

```
port (Sel : in std_logic_vector(1 downto 0);
A, B, C, D : in std_logic;
Y : out std_logic );
end MUX4_1;
architecture behavior2 of MUX4_1 is
begin
process (Sel, A, B, C, D)
begin
case Sel is
when &amp;quot;00&amp;quot; =&amp;gt; Y&amp;lt;=A;
when &amp;quot;01&amp;quot; =&amp;gt; Y&amp;lt;=B;
when &amp;quot;10&amp;quot; =&amp;gt; Y&amp;lt;=C;
when &amp;quot;11&amp;quot; =&amp;gt; Y&amp;lt;=D;
when others =&amp;gt; Y&amp;lt;=A;
end case ;
end process ;
end behavior2;
```



Conclusion:

Thus we have simulated 4:1 MUX in VHDL. We understood how VHDL works.

Post Lab Descriptive Questions

1. Application Mux?

Multiplexer is used in various applications that require single input data to be converted to multiple output streams.

In this VHDL experiment we have simulated mux in VHDL.