Visvesvaraya Technological University
ACS College of Engineering

Kambipura, Mysore Road, Bangaolre-560074

Department of Electronics & Communication Engineering

Digital System Design Laboratory Manual

[**18ECL38**]

III SEMESTER – B. E

Academic Year 2019-2020

Compiled By:

Mr. NAGESH H B

Assistant Professor

Checked and Approved by

Dr. H B Bhuvaneswari

Professor & HOD, ECE

ACS College of Engineering

Kambipura, Mysore Road, Bangaolre-560074

Department of Electronics &Communication Engineering

III SEMESTER

Digital System Design Laboratory Manual

Sub Code: 18ECL38

Name: ………………………………….

USN: …………………………………….

Sem : ………………………………..

Safety Measure

- Execution of Lab work in a safe manner is even more important than performing accurate electronic measurements and construction neat circuits.
- You should also know all equipment's and components that are used in the Lab to take the necessary precautions.
- Always power down the electrical equipment, disconnect the power cord, and wait for a few seconds before touching exposed wires. And make sure that your hands are dry.
- Do not wear rings, watches, necklace, and any other loose metallic objects. Rings and watches are especially dangerous as the skin beneath them is wet by sweat, making the resistance of skin much lower.
- In case of electric shock, cut the power and/or remove the victim as quickly as possible without endangering yourself.

Do's&Don'ts

Do's

- Study the theory behind the experiment before coming to the lab.
- Submit the completed record of previously conducted experiment & update the index card.
- Maintain observation book written with experiments to be conducted for the day.
- Take the signature of the staff in-charge before taking the components.
- Note down the specifications of the devices and equipment's used in the observation book.
- Handle the equipment's, devices and components carefully.
- Checked and Okayed the circuit rigged up, by the staff in-charge before energizing.
- To make any changes in the circuit, disconnected the power supply and the input signal
- Note down the proper readings in the observation book.
- After completion of the experiments switch of the power supply
- Do the necessary verification using truth table if any
- Show the obtained results to the staff in-charge and get initialed
- Handover the components to the lab instructor / mechanic
- Should Prepare for viva question regularly
- Should maintain discipline and silence in the lab
- Attend lab sessions regularly

Don'ts

- Do not come late the lab
- Do not exceed the Voltage / Current ratings of the devices
- Do not energize theequipment's and circuits without getting it checked by the staff in-charge
- Do not spoil the components, connecting wires and patch card
- Avoid loose connection and short circuit

DIGITAL SYSTEM DESIGN LABORATORY

[As per Choice Based Credit System (CBCS) scheme]
SEMESTER – III (EC/TC)

Laboratory Code	**18ECL38**	**CIE Marks**	**40**
Number of Lecture Hours/Week	**02Hr Tutorial (Instructions) + 02 Hours Laboratory**	**SEE Marks**	**60**
RBT Level	**L1, L2, L3,L4,L5,L6**	**ExamHours**	**03**

CREDITS – 02

Course objectives: This laboratory course enables students to get practical experience in design, realization and verification of

- Demorgan‘s Theorem, SOP, POS forms
- Full/Parallel Adders,and Magnitude Comparator
- Demultiplexers and Decoders applications
- Flip-Flops, Shift registers and Counters

NOTE: 1. Use discrete components to test and verify the logic gates. The IC umbers given are suggestive. Any equivalent IC can be used.
2. For experiment No. 11 and 12 any open source or licensed simulation tool may be used.

Laboratory Experiments:

1. Verify
   (a) Demorgan‘s Theorem for 2 variables.
   (b) The sum-of product and product-of-sum expressions using universal gates.

2. Design and implement
   (a) Half Adder & Full Adder using (i) basic logic gates and (ii) NAND gates.
   (b) Half substractor& Full substractor using (i) basic logic gates and (ii) NANAD gates.

3. Design and implement
   (i) 4-bit Parallel Adder/ Substractor using IC 7483.
   (ii) BCD to excess-3 code conversion an vice-versa

4. Design and Implementation of
   (i) 1- bit comparator
   (ii) 5-bit Magnitude Comparator using IC 7485.

5. Realize
   (a)Adder &Substractor using IC 74153.
   (b) 4-variable function using IC 74151(8:1MUX).

6. Realize
   (i) Adder and Substractors using IC74139.
   (ii) Binary to Gray code conversion & vice versa (74139)

7. Realize the following flip-flops using NAND Gates.
Master-Slave JK, D & T Flip-Flops.

8. Realize the following shift registers using IC7474/IC 7495
(a) SISO (b) SIPO (c) PISO (d) PIPO (e) Ring and (f) Johnson counter.

9. Realize
(i) Design Mod-N synchronous Up counter and Down Counter using 7476JK Flip-flop.
(ii) Mod-N Asynchronous Counter using IC7490 and
(iii) Mod-N Synchronous counter using IC74192

10. Design Pseudo Random Sequence generator using 7495.

11. Simulate Full- Adder using simulation tool.

12. Simulate Mod-8 Synchronous UP/DOWN Counter using simulation tool.

Course Outcomes: On the completion of this laboratory course, the students will be able to:

- **CO1:**Demonstrate the truth table of various expressions and combinational circuits using logic gates.
- **CO2:**Design and test various combinational circuits such as adders, Substractors, comparators, multiplexers and De-MUX.
- **CO3:**Construct Flip Flop using Universal Gates.
- **CO4:**Explain the Operation of counter and shift register
- **CO5:**Simulate serial adder and Binary Multiplier.

Conduct of Practical Examination:

- All laboratory experiments are to be included for practical examination.
- Students are allowed to pick one experiment from the lot.
- Strictly follow the instructions as printed on the cover page of answer script for breakup of marks.
- Change of experiment is allowed only once and 15% Marks allotted to the procedure part to be made zero.

LIST OF EXPERIMENTS

EXP.NO	NAME OF THE EXPERIMENT	CO	RBT	PAGE NO
Cycle - 1				
1	Verify a) Demorgan's theorem for 2 variables. b) The Sum-of-Product and Product-of-sum expression using universal gates	CO1,CO3	L1,L2,L3	6- 13
2	Design and implement (a) Half Adder & Full Adder using (i) basic logic gates and (ii) NAND gates. (b) Half substractor& Full substractor using (i) basic logic gates and (ii) NANAD gates.	CO2,CO3	L3,L6	14 - 20
Cycle – 2				
3	Design and implement (i) 4-bit Parallel Adder/ Substractor using IC 7483. (ii) BCD to excess-3 code conversion an vice-versa	CO3,CO5	L3,L6	21 - 28
4	Design and Implementation of (i) 1- bit comparator (ii) 5-bit Magnitude Comparator using IC 7485.	CO3,CO4	L3,L4,L5	29 - 34
5	Realize (a)Adder &Substractor using IC 74153. (b) 4-variable function using IC 74151(8:1MUX).	CO1,CO2,CO3	L2,L3,L6	35 - 43
6	Realize (I) Adder and Substractors using IC74139. (ii) Binary to Gray code conversion & vice versa (74139)	CO1,CO2,CO3	L2,L3,L6	44 - 49
Cycle – 3				
7	Realize the following flip-flops using NAND Gates. Master-Slave JK, D & T Flip-Flops.	CO1,CO3,CO5	L2,L3,L4	50 - 53
8	Realize the following shift registers using IC7474/IC 7495 (a) SISO (b) SIPO (c) PISO	CO1,CO3,CO5	L2,L3,L4	54 - 59

	(d) PIPO (e) Ring and (f) Johnson counter.			
9	Realize (i) Design Mod-N synchronous Up counter and Down Counter using 7476JK Flip-flop. (ii) Mod-N Counter using IC7490 and (iii) Mod-N Synchronous counter using IC74192	CO2,CO3,CO5	L2,L3,L5	60 - 69
10	Design Pseudo Random Sequence generator using 7495.	CO3,CO4	L3,L5	70 - 74
Cycle - 4				
	Introduction to MULTISIM			
11	Design and Simulate serial adder with accumulator using simulation tool.	CO3,CO5	L3,L4	77 - 78
12	Design and Simulate Binary multiplier using simulation tool.	CO1,CO3,CO5	L2,L3,L4	79 - 81

INTRODUCTION

VERIFICATION OF LOGIC GATES

COMPONENTS REQUIRED:

Sl. No	Particulars	Quantity
1	IC 7400, IC 7408, IC 7432, IC 7406, IC 7402, IC 7404, IC 7486,IC 7410	1 each
2	Logic gates (IC) trainer kit	1
3	Connecting patch chords.	About 20

THEORY:

The basic logic gates are the building blocks of more complex logic circuits. These logicgates perform the basic Boolean functions, such as AND, OR, NAND, NOR, Inversion, Exclusive-OR, Exclusive-NOR. The small circle on the output of the circuit symbols designates the logiccomplement. The AND, OR, NAND, and NOR gates can be extended to have more than twoinputs. A gate can be extended to have multiple inputs if the binary operation it represents iscommutative and associative.

These basic logic gates are implemented as small-scale integrated circuits (SSICs) or as partof more complex medium scale (MSI) or very large-scale (VLSI) integrated circuits. DigitalIC gates are classified not only by their logic operation, but also the specific logic-circuitfamily to which they belong. Each logic family has its own basic electronic circuit uponwhich more complex digital circuits and functions are developed. The following logicfamilies are the most frequently used.

TTL - Transistor-transistor logic

ECL - Emitter-coupled logic

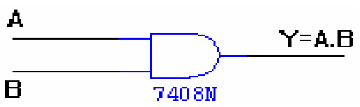
MOS - Metal-oxide semiconductor

CMOS - Complementary metal-oxide semiconductor

TTL and ECL are based upon bipolar transistors. TTL has a well establishedpopularityamong logic families. ECL is used only in systems requiring high-speed operation. MOS andCMOS, are based on field effect transistors. They are widely used in large scale integratedcircuitsbecause of their high component density and relatively low power consumption.CMOS logic consumes far less power than MOS logic. There are various commercial integrated circuit chips available. TTL ICs are usually distinguished by numerical designation 5as the 5400 and 7400 series.

AND GATE

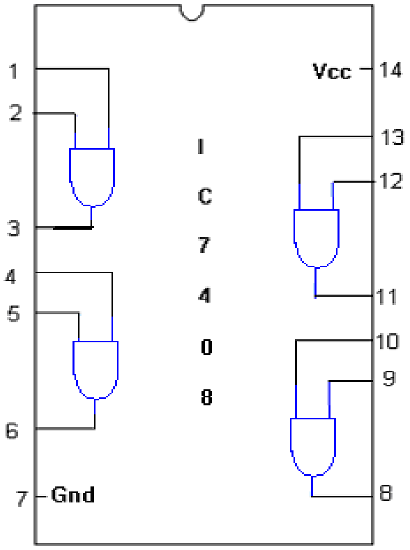
SYMBOL



TRUTH TABLE

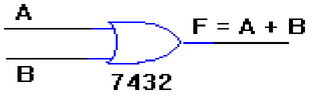
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

PIN DIAGRAM



OR GATE:

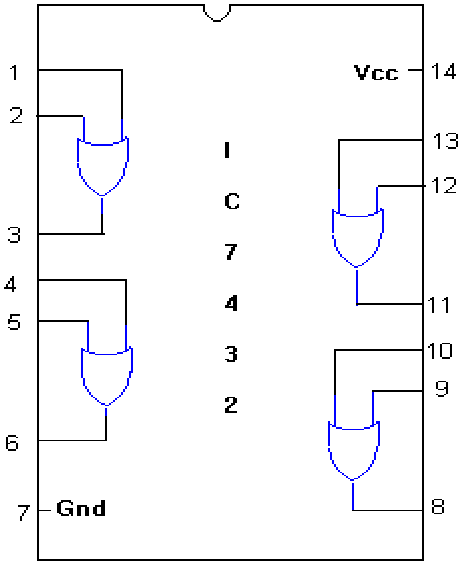
SYMBOL :



TRUTH TABLE

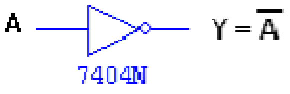
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

PIN DIAGRAM :

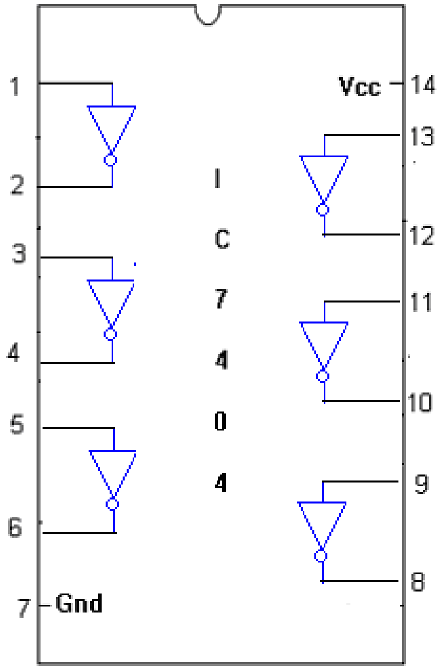


NOT GATE

SYMBOL



PIN DIAGRAM

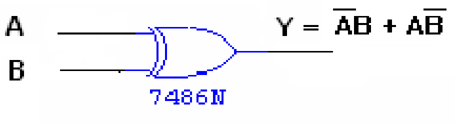


TRUTH TABLE :

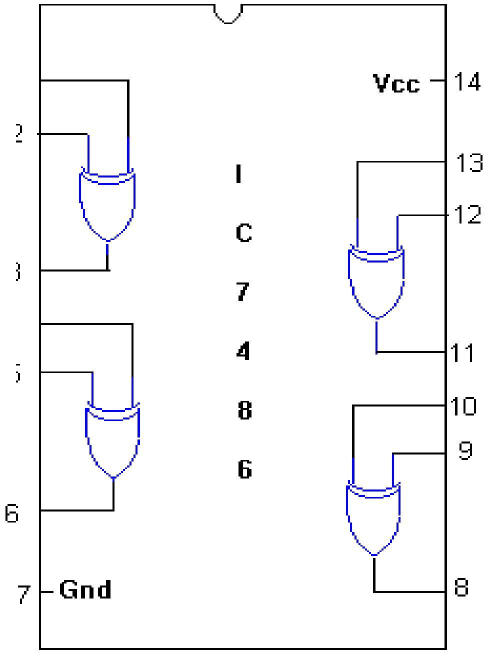
A	$\bar{A}$
0	1
1	0

X-OR GATE:

SYMBOL



PIN DIAGRAM

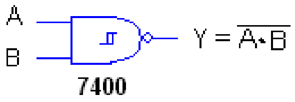


TRUTH TABLE :

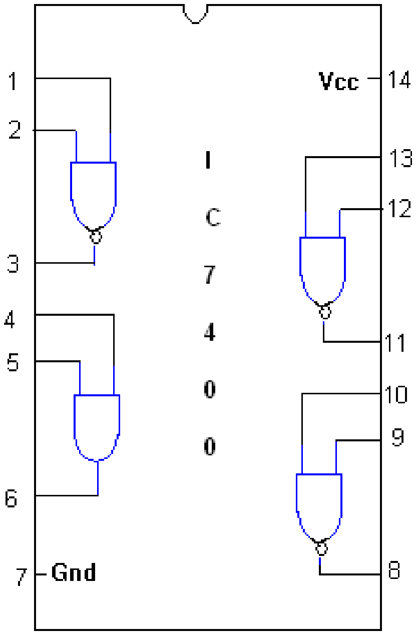
A	B	$\bar{A}B + A\bar{B}$
0	0	0
0	1	1
1	0	1
1	1	0

2-INPUT NAND GATE:

SYMBOL



PIN DIAGRAM

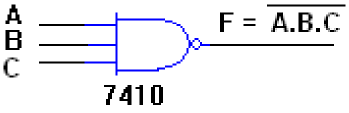


TRUTH TABLE

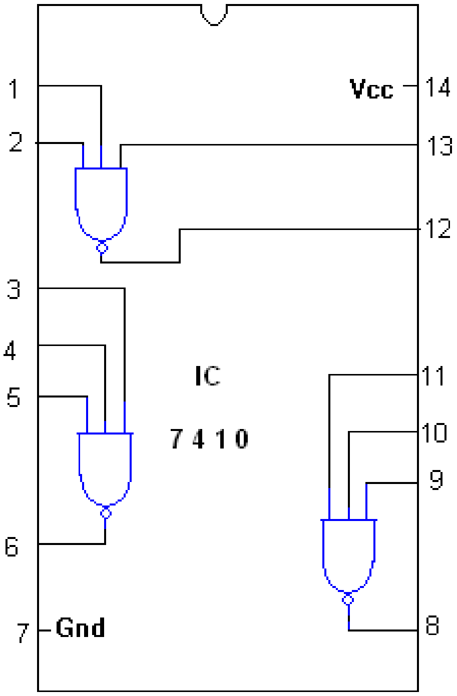
A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

3-INPUT NAND GATE

SYMBOL :



PIN DIAGRAM :

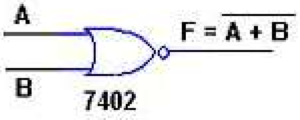


TRUTH TABLE

A	B	C	$\overline{A.B.C}$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

NOR GATE

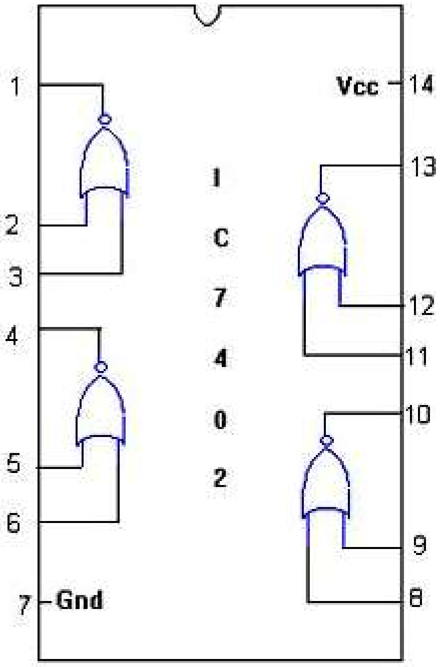
SYMBOL :



TRUTH TABLE

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

PIN DIAGRAM :



***Result*: ……………………………………………………………………………………**

Staff Signature

Experiment No.1

DEMORGAN'S THEOREM

Aim: a) VerificationofDemorgan's theorem for 2 variables

b)Verification of sum of products (SOP) and product of sum (POS) expressions using basic gates and universal gates.

Components Required:

Particulars	Quantity
IC 7404, 7408,7432	01 each

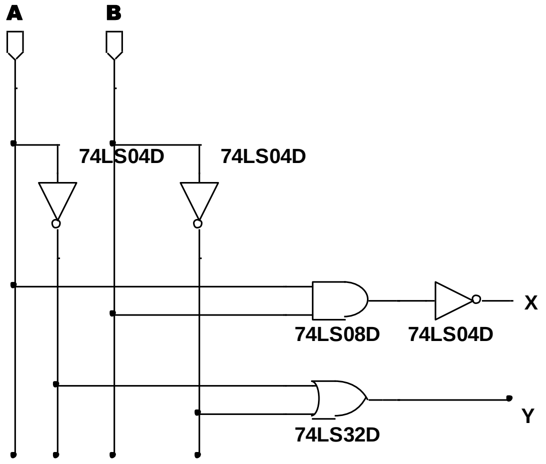
Demorgan's1st theorem Statement:

"The compliment of product of 2 or more variables is equal to the sum of complimentof 2 or more variables"

Truth Table:

Inputs		Outputs	
A	B	X = $\overline{\boldsymbol{A}.\boldsymbol{B}}$	Y= $\overline{\boldsymbol{A}} + \overline{\boldsymbol{B}}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

Logic Diagram:



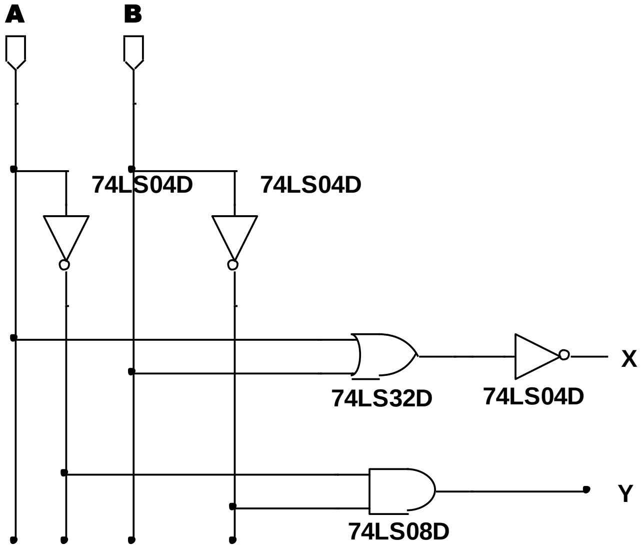
Demorgan’s 2^{nd} theorem Statement:

“The compliment of sum of 2 or more variables is equal to the product of complimentof 2 or more variables”

Truth Table:

Inputs		Outputs	
A	B	X $=\overline{A+B}$	Y= $\overline{A}.\overline{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

Logic Diagram:



Procedure:-

1. Place the IC in the socket of the trainer kit.

2. Make the connections as shown in the circuit diagram.

3. Apply the different combinations of input according to truth table and verify the output.

Result: ………………………………………………………………………………………

Staff Signature

b) Verification of SOP and POS expressions

Components Required:

Particulars	Quantity
IC 7404, 7408, 7432,7400,7410, 7402,,7427	01 each

i) SOP expression

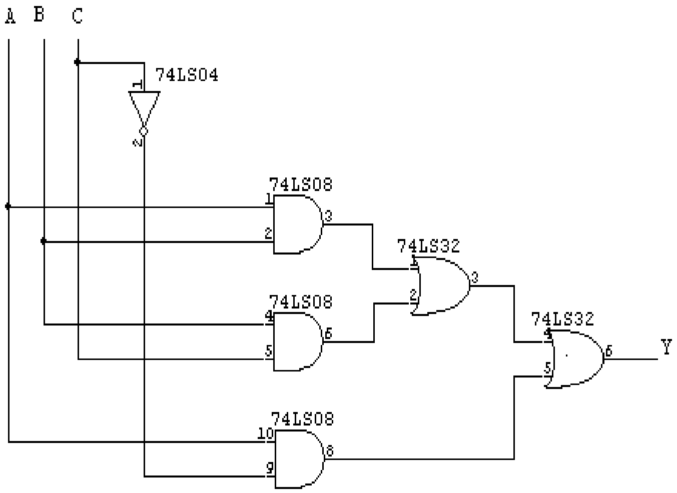
Y = AB + A$\overline{C}$+BC

Truth Table:

Inputs			Output
A	**B**	**C**	**Y**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Logic Diagram:

Realization Using Basic gates

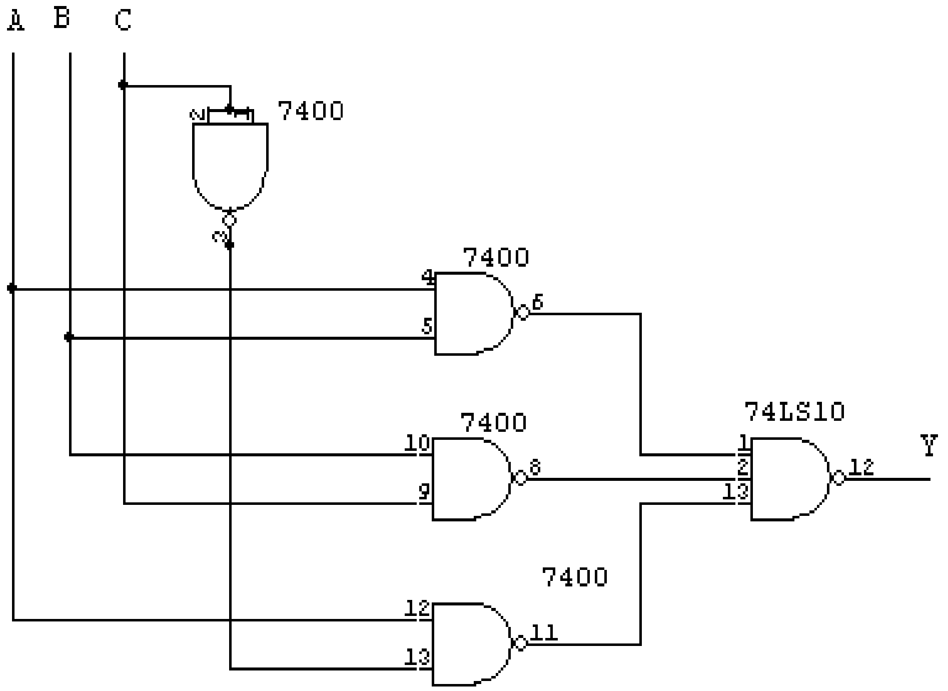


Realization Using NAND gates

$Y = AB + A\bar{C} + BC$

$Y = \overline{\overline{AB + A\bar{C} + BC}}$

$Y = \overline{\overline{AB}\,.\,\overline{A\bar{\bar{C}}}\,.\,\overline{BC}}$ (using Demorgan’stheorem)



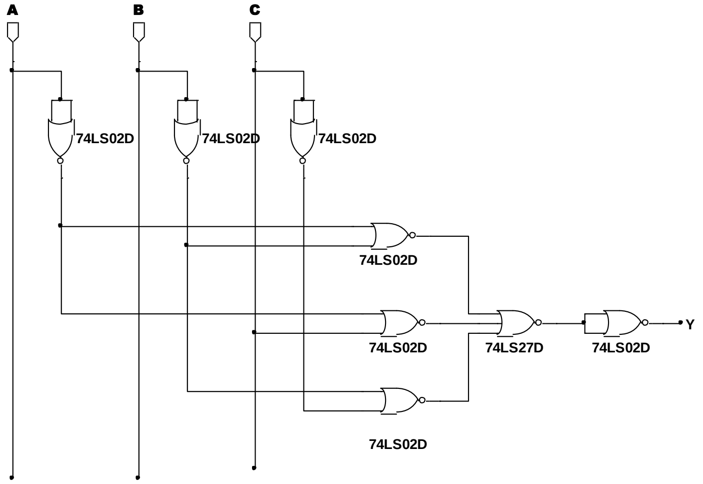
Realization Using NOR gates

$Y = AB + A\bar{C} + BC$

$Y = \overline{\overline{AB + A\bar{C} + BC}}$

$Y = \overline{\overline{AB}\,.\,\overline{A\bar{\bar{C}}}\,.\,\overline{BC}}$ (using Demorgan’stheorem)

$Y = \overline{(\bar{A} + \bar{B})} + \overline{(\bar{A} + C)} + \overline{(\bar{B} + \bar{C})}$



i) POSexpression

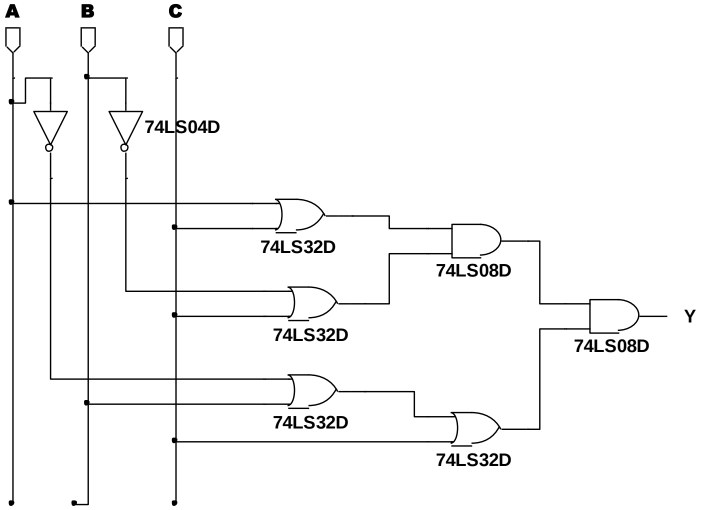
$$\boldsymbol{Y = (A + C) (\overline{B} + C) \ (\overline{A} + B + C)}$$

Truth Table:

Inputs			Output
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Logic Diagram:

Realization Using Basic gates



Realization Using NAND gates

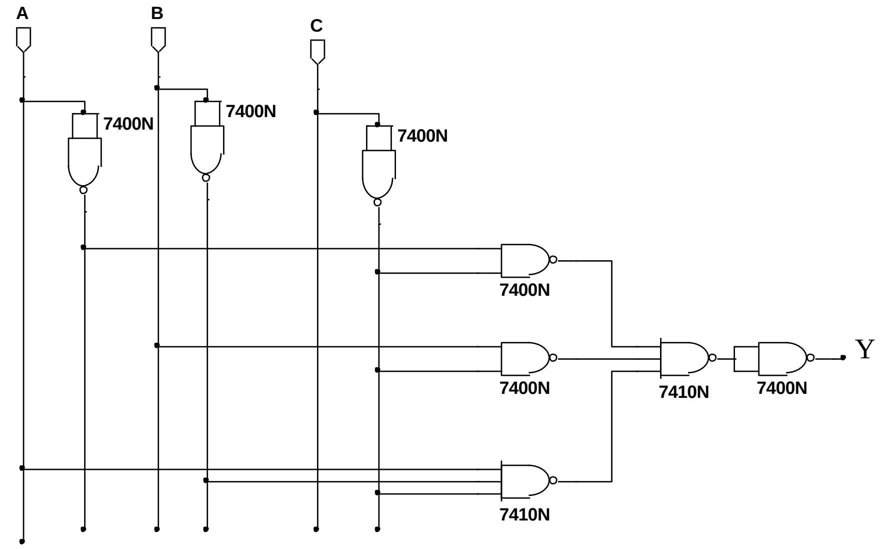
$$\boldsymbol{Y = (A + C)(\overline{B} + C)\ (\overline{A} + B + C)}$$

$$Y = \overline{\overline{(\boldsymbol{A} + \boldsymbol{C})(\overline{\boldsymbol{B}} + \boldsymbol{C})\ (\overline{\boldsymbol{A}} + \boldsymbol{B} + \boldsymbol{C})}}$$

$$Y = \overline{\overline{(\boldsymbol{A} + \boldsymbol{C})} + \overline{(\overline{\boldsymbol{B}} + \boldsymbol{C})} + \overline{(\overline{\boldsymbol{A}} + \boldsymbol{B} + \boldsymbol{C})}} \text{(using Demorgan'stheorem)}$$

$$\boldsymbol{Y} = \overline{(\overline{\boldsymbol{A}}.\overline{\boldsymbol{C}}) + (\boldsymbol{B}.\overline{\boldsymbol{C}}) + (\boldsymbol{A}.\overline{\boldsymbol{B}}.\overline{\boldsymbol{C}})}$$

$$\mathrm{Y} = \overline{(\overline{\boldsymbol{A}}.\overline{\boldsymbol{C}})}\ .\ \overline{(\boldsymbol{B}.\overline{\boldsymbol{C}})}\ .\ \overline{(\boldsymbol{A}.\overline{\boldsymbol{B}}.\overline{\boldsymbol{C}})}$$

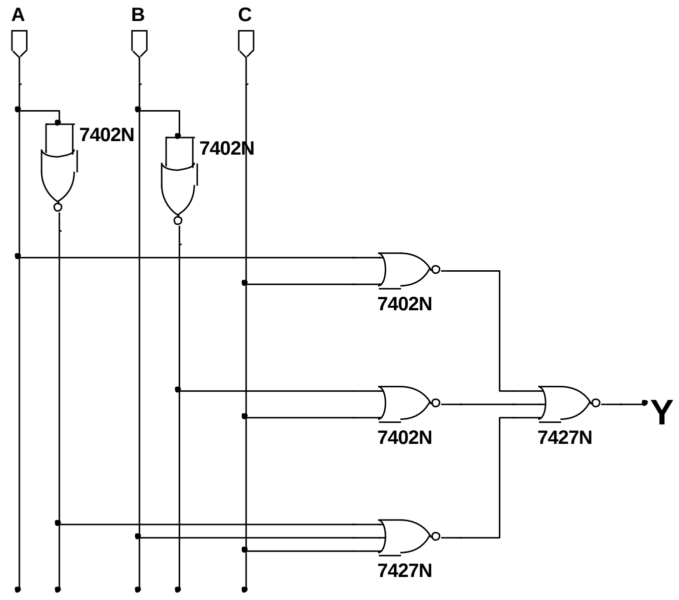


Realization Using NOR gates

$$\boldsymbol{Y = (A + C)(\overline{B} + C)\ (\overline{A} + B + C)}$$

$$Y = \overline{\overline{(\boldsymbol{A} + \boldsymbol{C})(\overline{\boldsymbol{B}} + \boldsymbol{C})\ (\overline{\boldsymbol{A}} + \boldsymbol{B} + \boldsymbol{C})}}$$

$$Y = \overline{\overline{(\boldsymbol{A} + \boldsymbol{C})} + \overline{(\overline{\boldsymbol{B}} + \boldsymbol{C})} + \overline{(\overline{\boldsymbol{A}} + \boldsymbol{B} + \boldsymbol{C})}} \text{(using Demorgan'stheorem)}$$



Procedure:-

1. Place the IC in the socket of the trainer kit.
2. Make the connections as shown in the circuit diagram.
3. Apply the different combinations of input according to truth table and verify the output.

Result: ……………………………………………………………………………………

Staff Signature

VIVA Question

1.Write the difference between digital and analog electronics?

2.Explain the weighted and non weighted code?

3.Explain a)Distributivelawb) Commutative Property and c) Complement Property with an example.

4.Why is called Boolean Algebra?

5.Write the properties of Boolean Algebra?

6.Construct the truth table for 4-input.

a).NAND gate b).NOR gate c).XOR gated).XNOR gate

7.How do you convert a XOR gate into buffer and an inverter(Use only one XOR gate for each)?

Experiment No. 2

FULL ADDER AND FULL Substractor

Aim : Design and implement

(a) Half Adder & Full Adder using (i) basic logic gates and (ii) NAND gates.

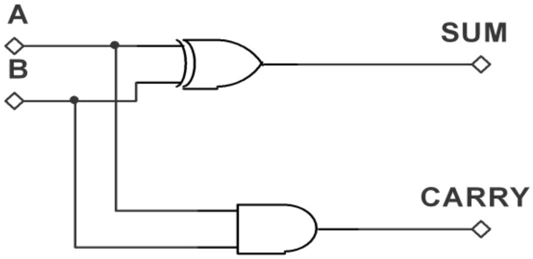
(b) Half subtractor& Full subtractor using (i) basic logic gates and (ii) NANAD gates.

Components Required:

Particulars	Quantity
IC 7408, 7432, 7486 ,7404,	01 each
IC7400	03

Half Adder:

Logic Diagram: Using Logic Gates



Truth Table

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

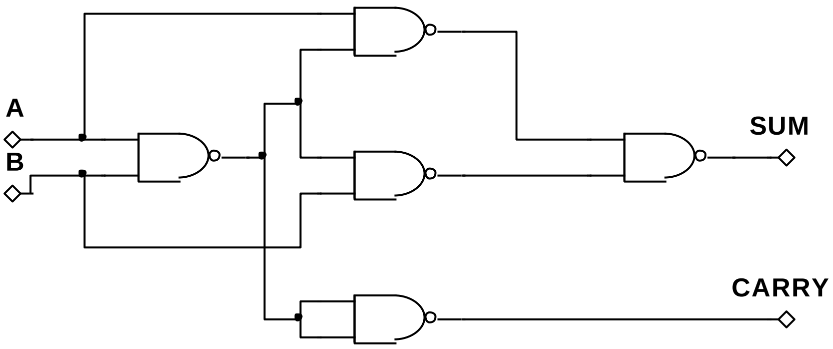
$\boldsymbol{S = A \oplus B}$

$\boldsymbol{C = A.B}$

$\boldsymbol{SUM = \overline{A}B + A\overline{B}}$

$\boldsymbol{CARRY = A.B}$

Logic Diagram: Using NAND Gates



Full Adder:

Truth Table

Inputs			Outputs	
A	**B**	**C**	**Sum (S)**	**Carry Out (Co)**
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

Design:
SUM

$S = A\bar{B}\bar{C}+\bar{A}\bar{B}C+\bar{A}B\bar{C}+ABC$

$S = \bar{C}(A\bar{B}+\bar{A}B)+C(\bar{A}\bar{B}+AB)$

$S = \bar{C}(A\oplus B)+C(\overline{A\oplus B})$

$S=A \oplus (B\oplus C)$

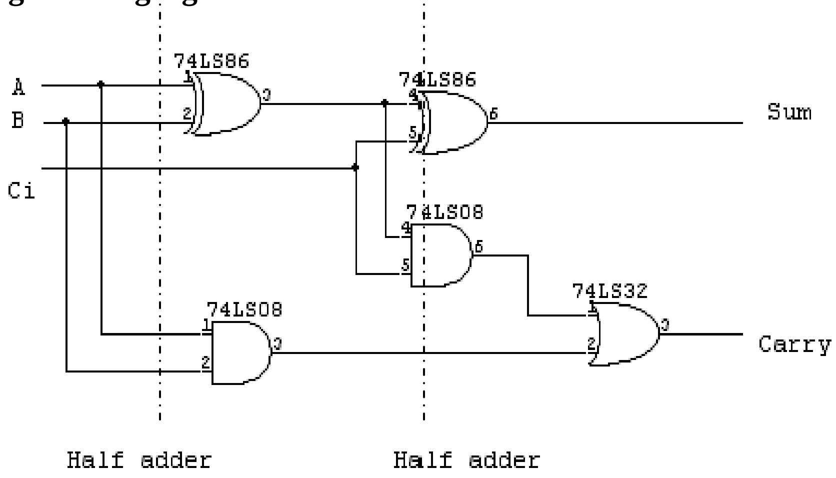
CARRY

$Co=\bar{A}BC+A\bar{B}C+AB\bar{C}+ABC$

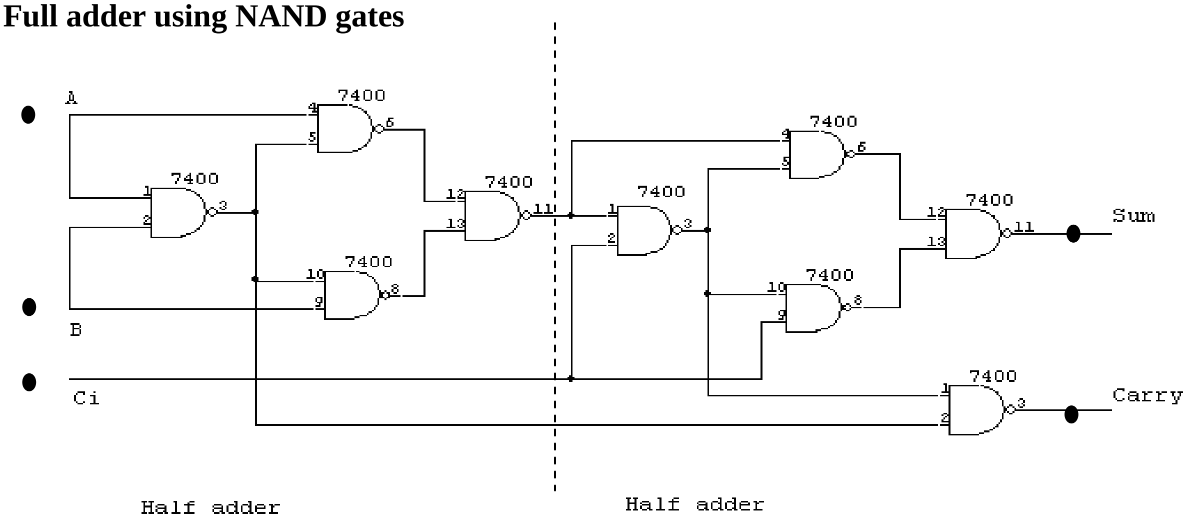
$Co=AB(C+\bar{C}) + C(A\oplus B)$

$Co=AB+ C(A\oplus B)$

Full adder using basic logic gates

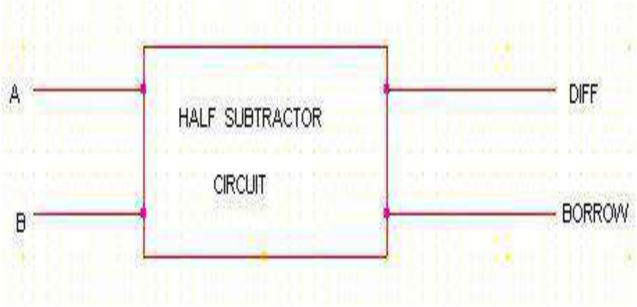


Full adder using NAND gates

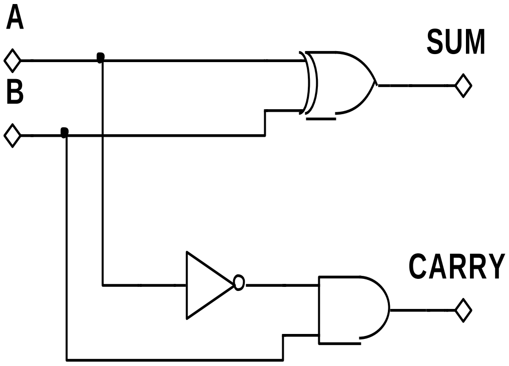


Half Substractor

Block Diagram:

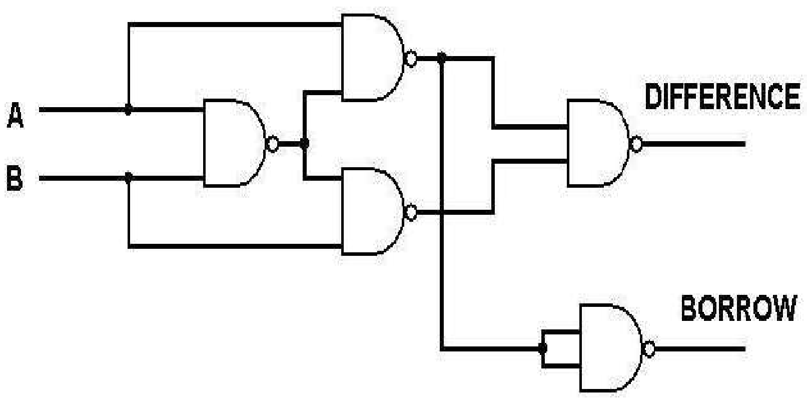


Logic Diagram: Using Logic Gates Truth Table



A	B	DIFF.	BORROW
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Logic Diagram: Using NAND Gates



FULL Substractor

Truth Table

Inputs			Outputs	
A	**B**	**C**	**Diff (D)**	**Borrow(Bo)**
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

DIFFERENCE

$D = A\bar{B}\bar{C}+\bar{A}\bar{B}C+\bar{A}B\bar{C}+ABC$

$D = \bar{C}(A\bar{B}+\bar{A}B)+C(\bar{A}\bar{B}+AB)$

$D = \bar{C}(A\oplus B) +C(\overline{A\oplus B})$

$D = A \oplus (B\oplus C)$

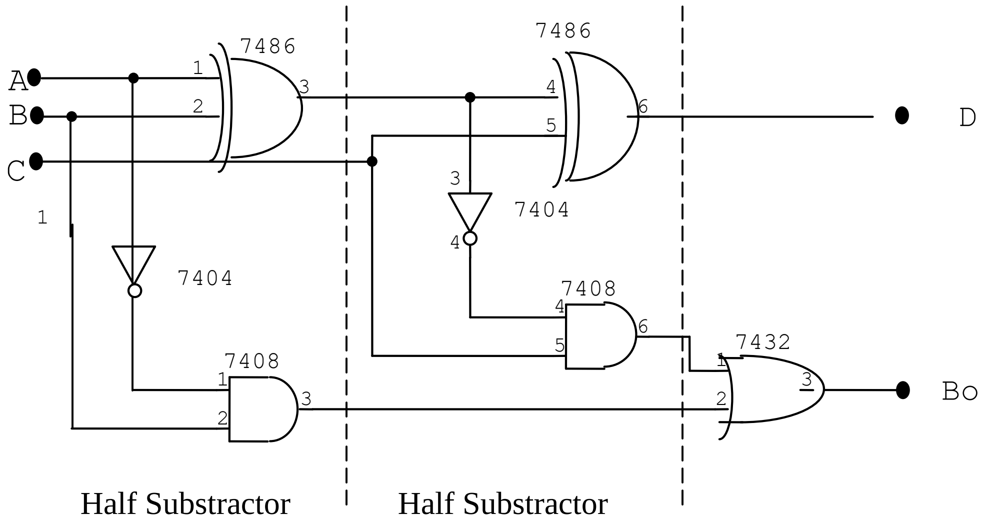
BORROW:

$Bo = \bar{A}\bar{B}C+\bar{A}B\bar{C}+\bar{A}BC+ABC$

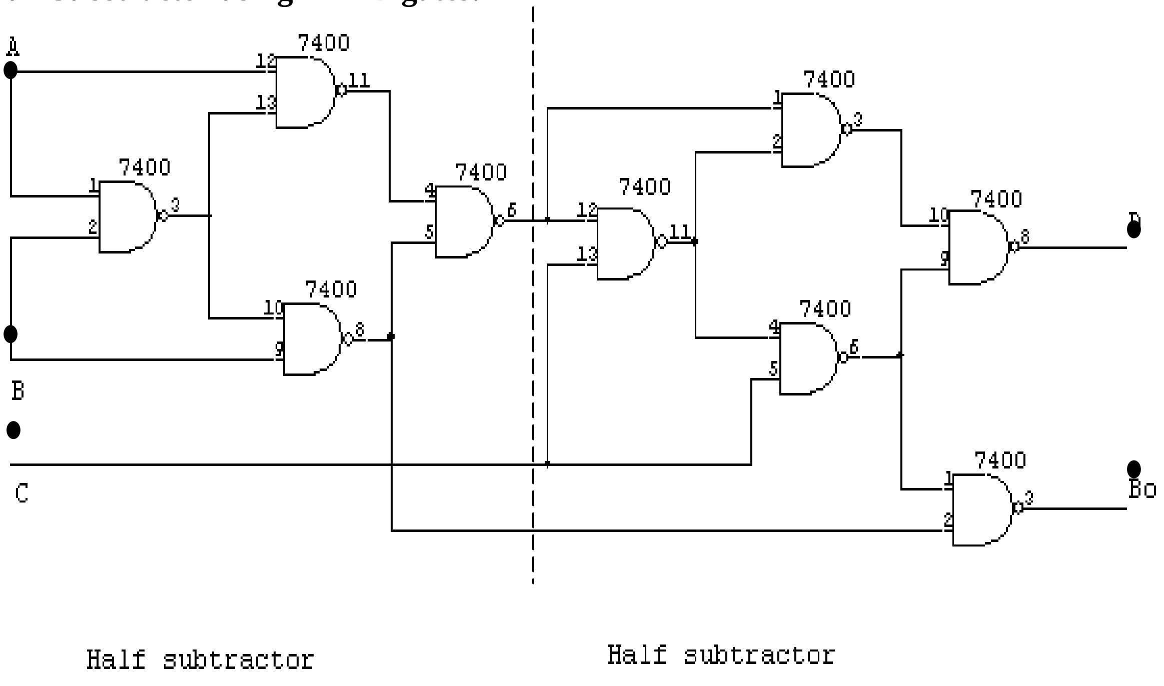
$Bo = \bar{A}B(C+\bar{C})+C\overline{(A\oplus B)}$

$Bo=\bar{A}B + C\overline{(A\oplus B)}$

Full Substractor using basic logic gates



Full Substractor using NAND gates:



Procedure

1. Place the IC in the socket of the trainer kit.
2. Make the connections as per the circuit diagram.
3. Switch on V_{CC} and apply various combinations of input according to the truth table.
4. Note down the output readings for full adder and full Substractor sum/difference and the carry/borrow bit for different combinations of inputs.

Result: **…………………………………………………………………………………….**

Staff Signature

VIVA Question

1. Define half adder and full adder?
2. Define half Substractor and full Substractor?
3. Explain the different types of canonical form with example?
4. Define prime implicants and essential prime implicants?
5. Find the prime and essential prime implicants from the switching equation?
6. D=f(W,X,Y,Z)=Σm(5,7,8,9,13) b). U=f(W,X,Y,Z)=Σm(1,5,7,8,9,10,11,13,15)
7. Explain incompletely specified functions(Don't care terms)?
8. Explain Combinational and Sequential circuits?

Experiment No.3

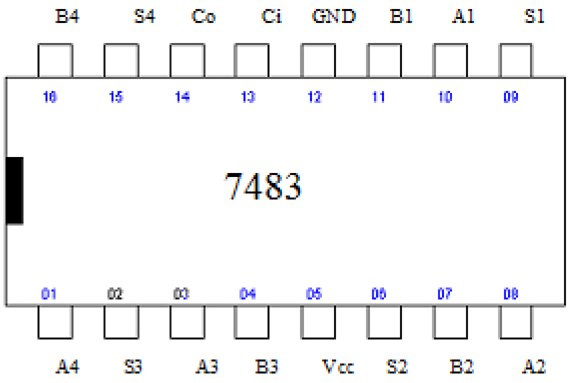
PARALLEL ADDER/SUBTRACTOR

Aim: Design and implement 4-bit Parallel Adder/ Substractor using IC 7483.

Components Required:

Particulars	Quantity
IC 7483, 7486	1 each

Pin Diagram of IC 7483



Design:

A - B= A + 2's B

Example1: Minuend is greater than Subtrahend

A=1100, B= 0010

Take 2's compliment of B, so B= 1110

A+ 2's B= 1100

+1110

Co=1 1010

If Co = 1 answer is +ve

Example 2: Subtrahend is greater than minuend

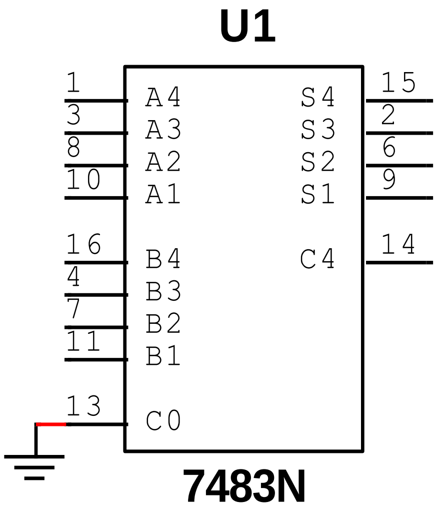
A=0010, B= 1100

Take 2’s compliment of B, so B= 0100

A+ 2’s B= 0010
+ 0100
Co=0 0110

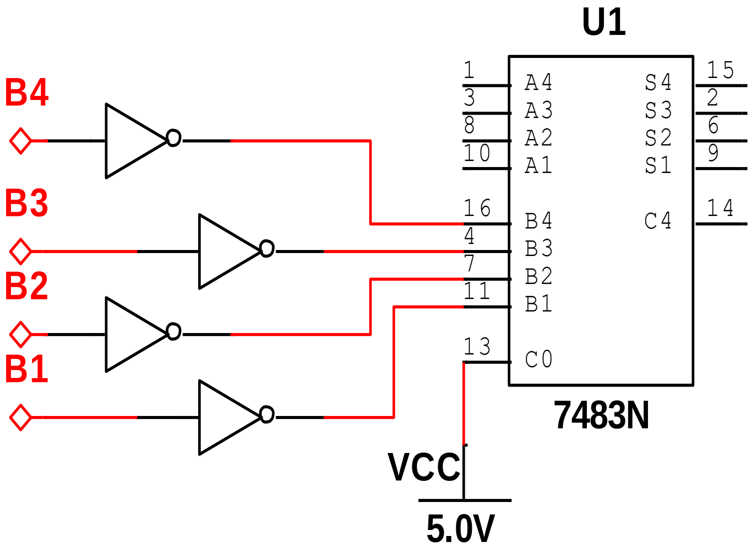
If Co=0 answer is –ve, take 2’s complement of the result to get the magnitude.

ADDER:



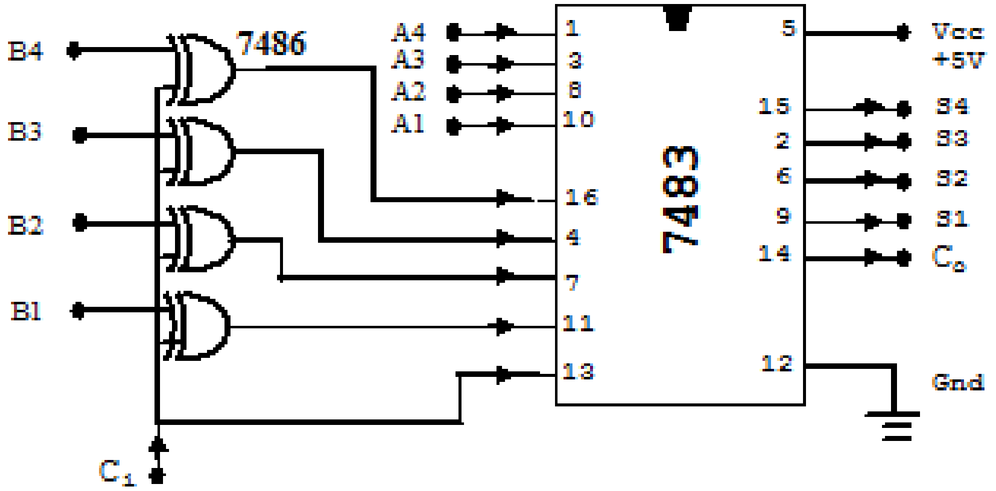
OUTPUT: S1,S2,S3,S4
INPUTS : A- A1,A2,A3,A4
B- B1,B2,B3,B4
VCC : PIN No = 05
GND : PIN No =12

SUBTRACTION:



OUTPUT: S1,S2,S3,S4
INPUTS : A- A1,A2,A3,A4
B- B1,B2,B3,B4
VCC : PIN No = 05
GND : PIN No =12

Logic diagram for parallel adder / Substractor



If control Ci = 0, addition can be performed, if Ci=1, subtraction can be performed.

Truth Table for parallel adder

		Inputs									Outputs				
Decimal No		**Carry in**	**Binary Digit – A**				**Binary Digit - B**				**Carry out**	**Sum**			
A	**B**	**C_i**	**A4**	**A3**	**A2**	**A1**	**B4**	**B3**	**B2**	**B1**	**C_o**	**S4**	**S3**	**S2**	**S1**
9	5	0	1	0	0	1	0	1	0	1	0	1	1	1	0
12	8	0	1	1	0	0	1	0	0	0	1	0	1	0	0

Truth Table for parallel Substractor

		Inputs									Outputs				
Decimal No		**Borrow in**	**Binary Digit – A**				**Binary Digit - B**				**Borrow out**	**Difference**			
A	**B**	**C_i**	**A4**	**A3**	**A2**	**A1**	**B4**	**B3**	**B2**	**B1**	**C_o**	**S4**	**S3**	**S2**	**S1**
9	5	1	1	0	0	1	0	1	0	1	1	0	1	0	0
8	12	1	1	0	0	0	1	1	0	0	0	1	1	0	0

ii) BCD TO EXCESS-3 CODE CONVERSION AND VISE VERSA USING IC 7483

Theory:

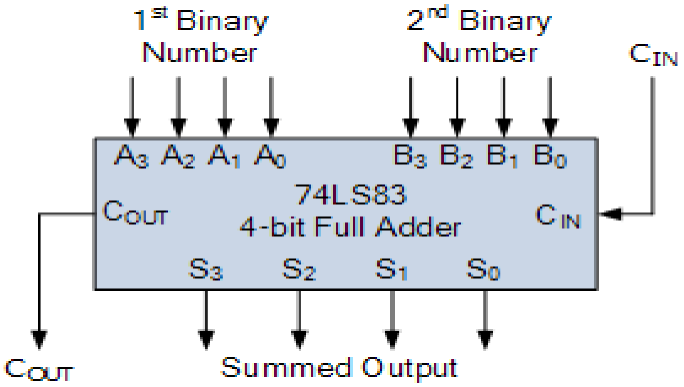
Code converter is a combinational circuit that translates the input code word into a new corresponding word. The excess-3 code digit is obtained by adding three to the corresponding BCD digit. To Construct a BCD-to-excess-3-code converter with a 4-bit adder feed BCD code to the 4- bit adder as the first operand and then feed constant 3 as the second operand. The output is the corresponding excess-3 code.

To make it work as an excess-3 to BCD converter, we feed excess-3 code as the first operand and then feed 2's complement of 3 as the second operand. The output is the BCD code.

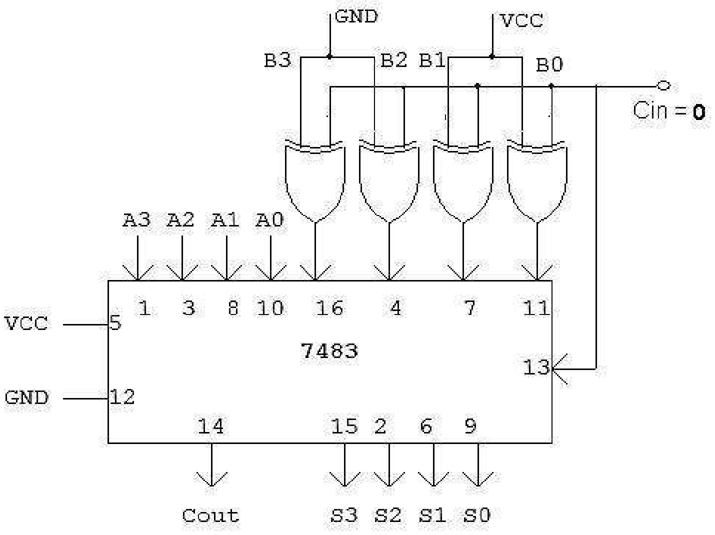
Truth Table

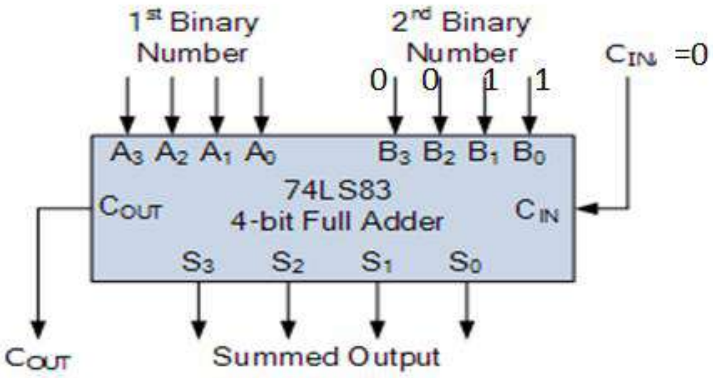
BCD (8421)				**Excess-3**			
A	**B**	**C**	**D**	**W**	**X**	**Y**	**Z**
0	**0**	**0**	**0**	**0**	**0**	**1**	**1**
0	**0**	**0**	**1**	**0**	**1**	**0**	**0**
0	**0**	**1**	**0**	**0**	**1**	**0**	**1**
0	**0**	**1**	**1**	**0**	**1**	**1**	**0**
0	**1**	**0**	**0**	**0**	**1**	**1**	**1**
0	**1**	**0**	**1**	**1**	**0**	**0**	**0**
0	**1**	**1**	**0**	**1**	**0**	**0**	**1**
0	**1**	**1**	**1**	**1**	**0**	**1**	**0**
1	**0**	**0**	**0**	**1**	**0**	**1**	**1**
1	**0**	**0**	**1**	**1**	**1**	**0**	**0**
1	**0**	**1**	**0**	**X**	**X**	**X**	**X**
1	**0**	**1**	**1**	**X**	**X**	**X**	**X**
1	**1**	**0**	**0**	**X**	**X**	**X**	**X**
1	**1**	**0**	**1**	**X**	**X**	**X**	**X**
1	**1**	**1**	**0**	**X**	**X**	**X**	**X**
1	**1**	**1**	**1**	**X**	**X**	**X**	**X**

Pin Diagram:



Logic Diagram:



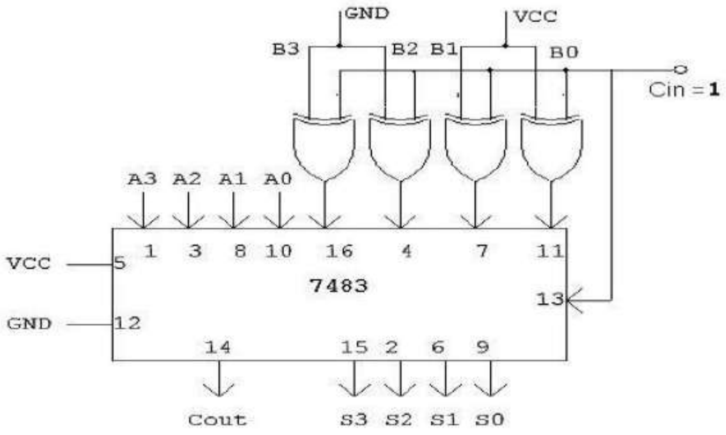


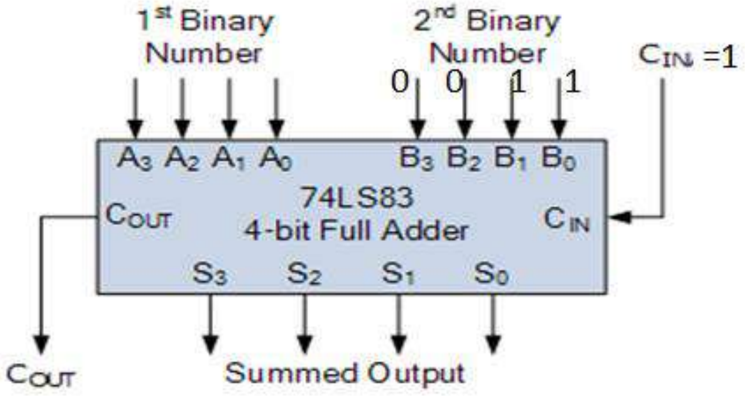
Note: Keep the input (B3,B2,B1,B0=0011) constant and Cin=0

To realize EXCESS-3 CODE to BCD conversion

Excess-3				BCD (8421)			
W	X	Y	Z	A	B	C	D
0	0	0	0	X	X	X	X
0	0	0	1	X	X	X	X
0	0	1	0	X	X	X	X
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	1
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
	1	1	1	X	X	X	X

Logic Diagram:





Note: Keep the input (B3,B2,B1,B0=0011) constant and Cin=1

Procedure

1. The IC is fixed on the IC base board and VCC & GND connections are given from 5V supply.
2. Connections are made as shown in the Logic diagram.
3. The truth table is verified for different combinations of input.

***Result:* ……………………………………………………………………………………....**

Staff Signature

VIVA Question:

1. Explain how MEV map differs from K-map?
2. Explain the terms.
3. a) Arithmetic logic unit (ALU) b) Array multiplier c) BCD adder d)Comparator
4. Draw and explain the block diagram of n-bit parallel adder?
5. What do you mean by carry propagation delay?
6. Define clock skew, negative clock skew, positive clock skew?
7. Explain the method used for carry look ahead generation?

Experiment No. 4

COMPARATOR

Aim:- Design and Implementation of
a) To realize a one bit comparator.
b) 5-bit magnitude comparator using IC 7485

Components required

Particulars	Quantity
IC 7400, , 7404, 7408, 7432, 7485	2 No

a) One bit Comparator

Truth table

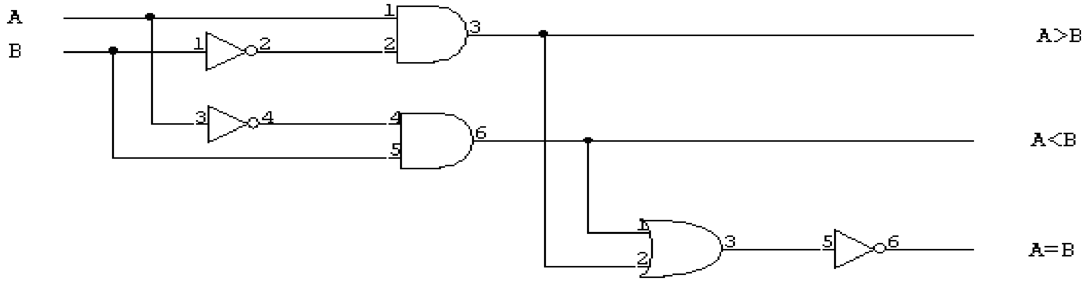
Inputs		Outputs		
A	B	A<B	A=B	A>B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

Design

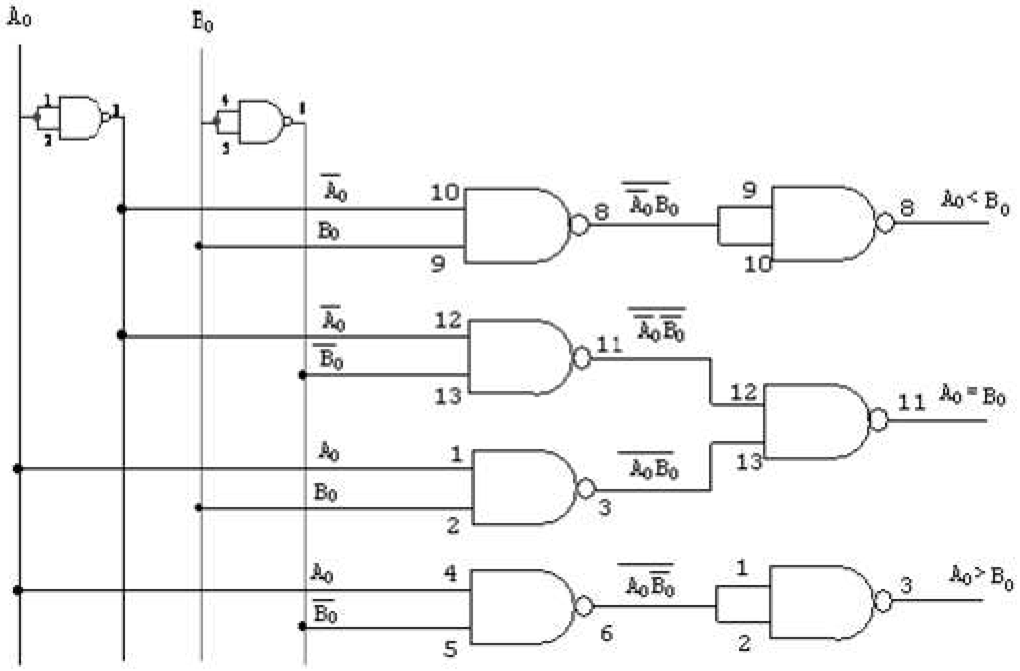
A>B			**A=B**				**A<B**		
B				B				B	
A	0	1		A	0	1		A 0	1
0	0	0		0	1	0		0 0	1
1	1	0		1	0	1		1 0	0
$A>B = A\,\overline{B}$				$A=B = \overline{A}\ \ \overline{B} + AB$			$A<B = \overline{A}\,B$		

Logic diagram

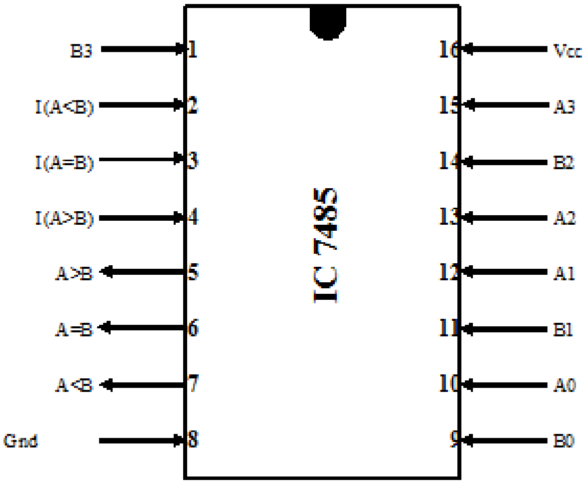
Realization using Basic Gates



Realization using NAND Gates



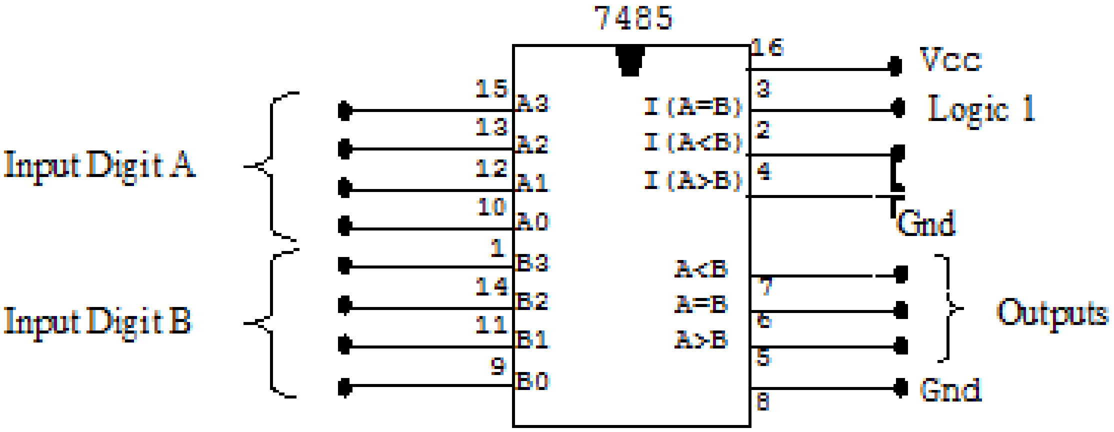
Pin Configuration IC 7485



Truth table for 4-bit Comparator

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

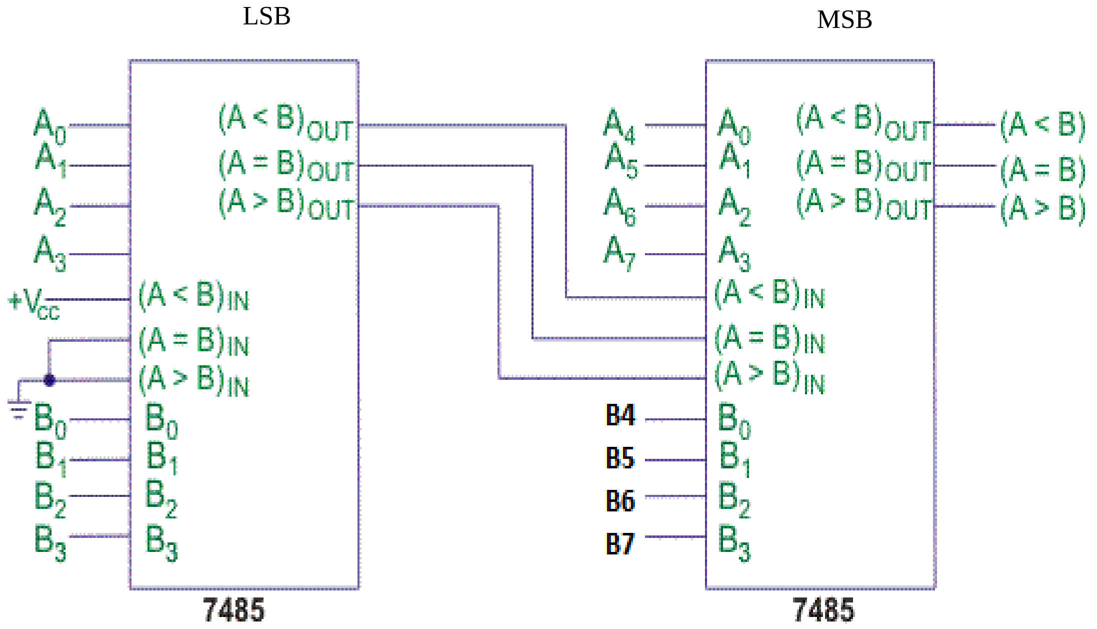
4-bit Comparator using IC7485



***Note1**: For 1 bit comparision using IC7485 provide data on A0 & B0 pins, connect rest all input pins to logic ‘0’*

***Note 2:** For 2 bit comparision using IC7485 provide data on A1, A0 ,& B1, B0 pins, connect rest all input pins to logic ‘0’*

Logic circuit of 5-bit Comparator using IC7485:



Note: *For 5 bit comparison using IC7485 provide data on A0- A4& B0-B4 pins, connect rest all input pins* (**A_5, A_6,A_7, B_5, B_6,B_7**) *to logic '0'*

Truth table for 5-bit Comparator:

INPUTS										OUTPUTS		
A					B					A>B	A=B	A<B
A4	A3	A2	A1	A0	B4	B3	B2	B1	B0			
1	0	0	0	1	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	1	0	1	0
0	0	0	0	1	1	0	0	0	1	0	0	1

Procedure:

1. The IC is fixed on the IC zip socket and Vcc&Gnd connections are given from 5V Supply.
2. Connections are made as shown in the Logic diagram.
3. All the inputs are connected to the switches & output to the LEDs.
4. Truth table of comparator is verified for different combinations of input.

Result: ……………………………………………………………………………………….

Staff Signature

VIVA Question:

1. What is a 1 bit comparator?
2. What is a logic comparator?
3. How does a comparator work?
4. What are comparator and its types?
5. What is the use of Cascade inputs in IC 7485?

Experiment No. 5

MULTIPLEXER

Aim : Realize

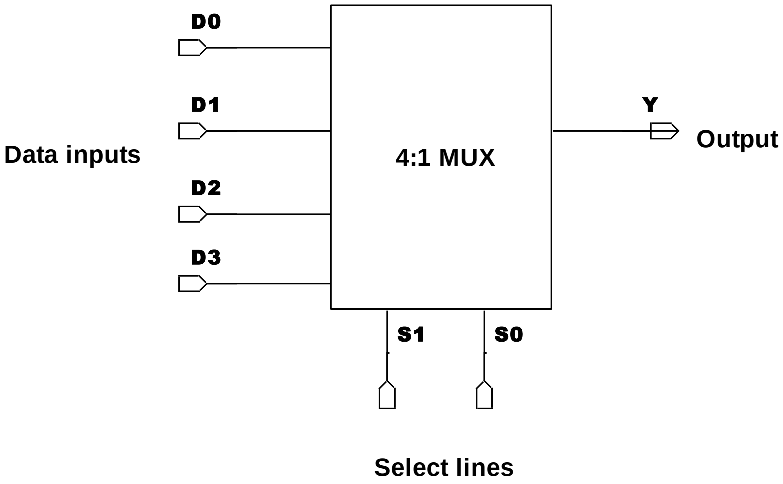
a) Adder &Substractor using IC 74153

b) 4 - variable function using IC 74151 (8:1 MUX)

Components Required:

Particulars	Quantity
IC 7404, 7415, IC 7432,74151 ,74153	1 No.each

Block diagram of 4:1Multiplexer

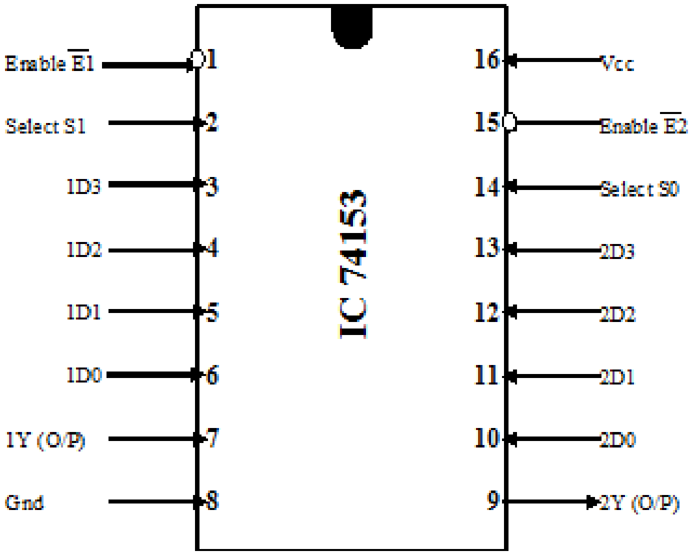


Truth Table

Select lines		Output
S1	**S0**	**Y**
0	0	D0
0	1	D1
1	0	D2
1	1	D3

a) Realization of Adder &Substractor using IC 74153

Pin Diagram of IC 74153(Dual 4: 1 Mux)



Half adder using 74153

Truth Table:

Input		Output	
A	**B**	**Sum**	**Carry**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Sum is realized on Mux A, Carry on Mux B, hence enable both the Mux by connecting Ea&Eb to logic 0

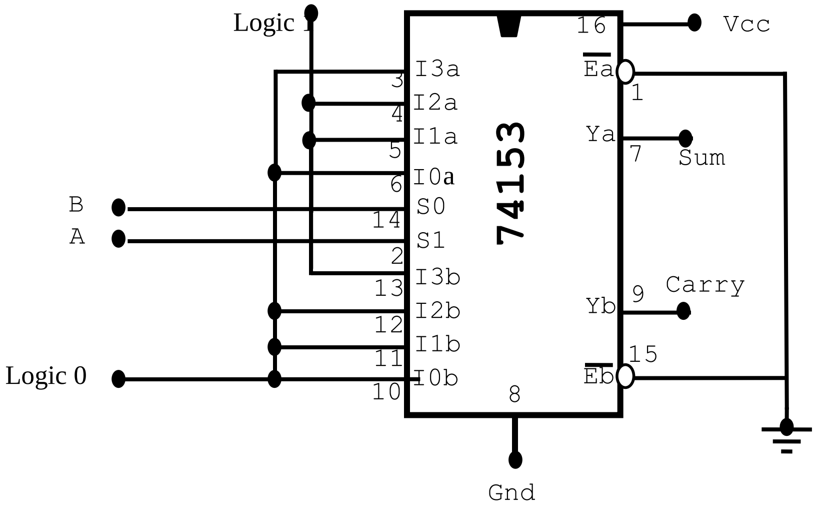
Realization: Connect A & B input to S1, S0 lines of Mux respectively.

According to input A & B different data input lines(I0 to I3) are selected.

Connect the data input line to logic 0 or 1 according to the required outputs.

Sum=Σ m(1,2) Carry =Σm(3)

Logic diagram



Full adder using 74153

Truth table

Input			Output	
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

Sum is realized on Mux A, Carry on Mux B, hence enable both the Mux by connecting Ea&Eb to logic 0

Realization: Connect A & B input to S1, S0 lines of Mux respectively.

According to input A & B different data input lines(I0 to I3) are selected. Express sum & carry in terms input variable C

SUM= ∑m (1, 2, 4, 7) CARRY= ∑ m (3, 5, 6, 7)

Implementation table

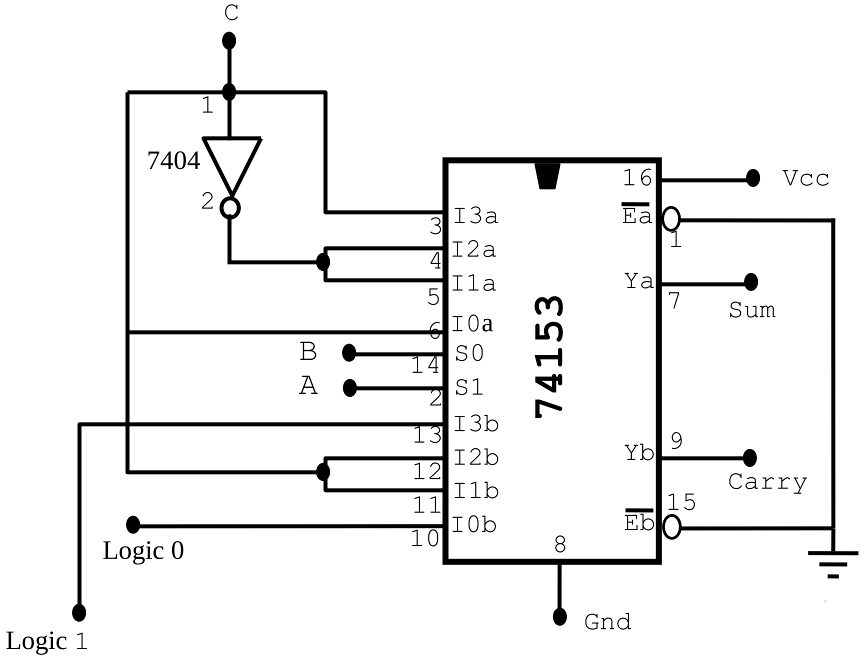
For Sum:

	I0a	I1a	I2a	I3a
C’	0	1	1	0
C	1	0	0	1
	C	C’	C’	C

For Carry:

	I0b	I1b	I2b	I3b
C’	0	0	0	1
C	0	1	1	1
	0	C	C	1

Logic diagram



Half Substractor using 74153

Truth Table

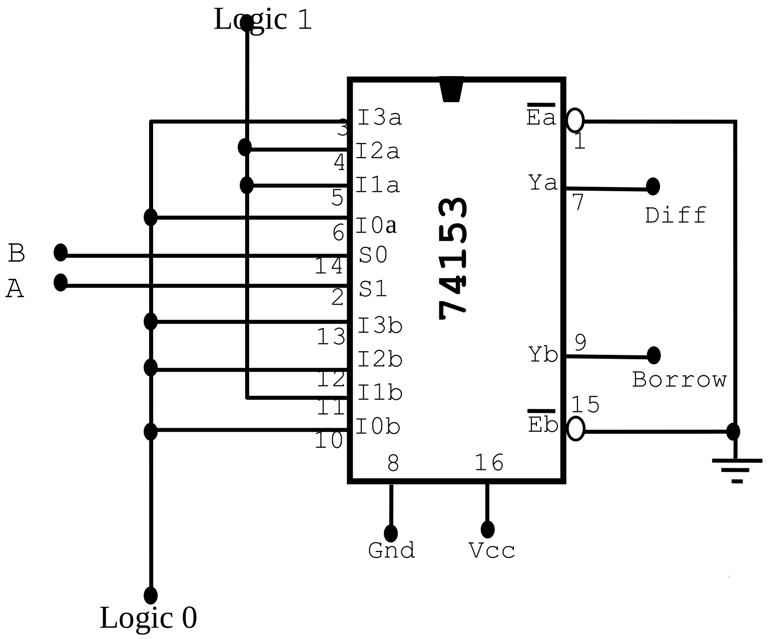
Input		Output	
A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Diff is realized on Mux A, Borrow on Mux B, hence enable both the Mux by connecting Ea&Eb to logic 0

Realization: Connect A & B input to S1, S0 lines of Mux respectively.

Difference = Σ m (1, 2) **Borrow = Σ m (1)**

Logic diagram



Full Substractor using IC 74153

Truth Table

Input			Output	
A	**B**	**C**	**Diff**	**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

Diff is realized on Mux A, Borrow on Mux B, hence enable both the Mux by connecting Ea&Eb to logic 0

Realization: Connect A & B input to S1, S0 lines of Mux respectively.

According to input A & B different data input lines(I0 to I3) are selected. Express Diff & Borrow in terms input variable C.

Diff= ∑ m (1, 2, 4, 7) Borrow = ∑m (1, 2, 3, 7)

Implementation table:

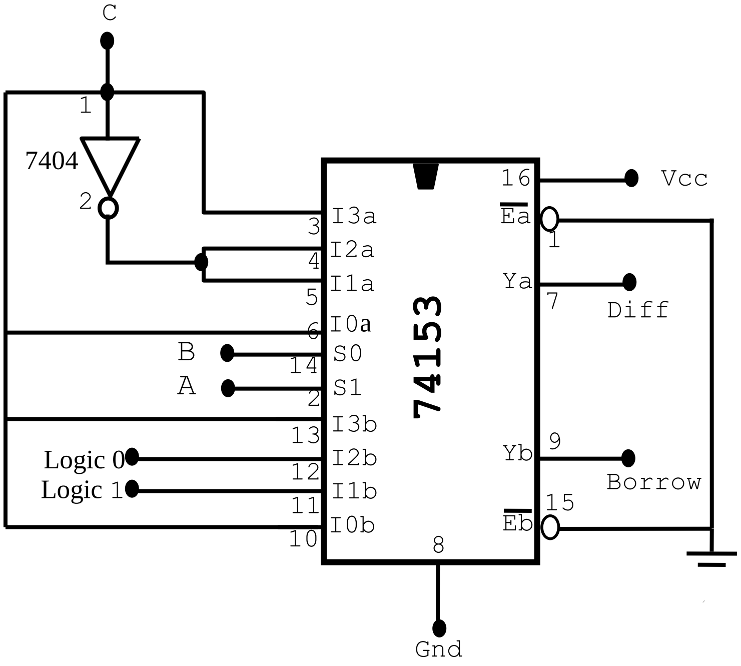
For Difference:

	I0a	I1a	I2a	I3a
C’	0	1	1	0
C	1	0	0	1
	C	C’	C’	C

For Borrow:

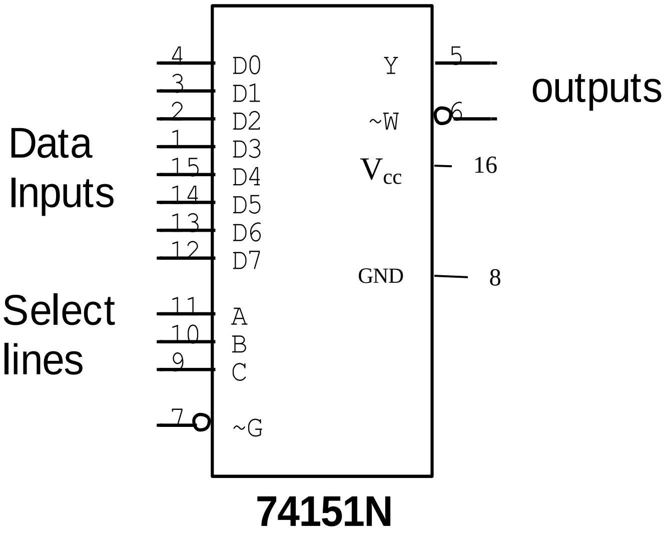
	I0b	I1b	I2b	I3b
C’	0	1	0	0
C	1	1	0	1
	C	1	0	C

Logic Diagram



(b) 4 -variable function using IC 74151(8:1MUX).

Pin configuration

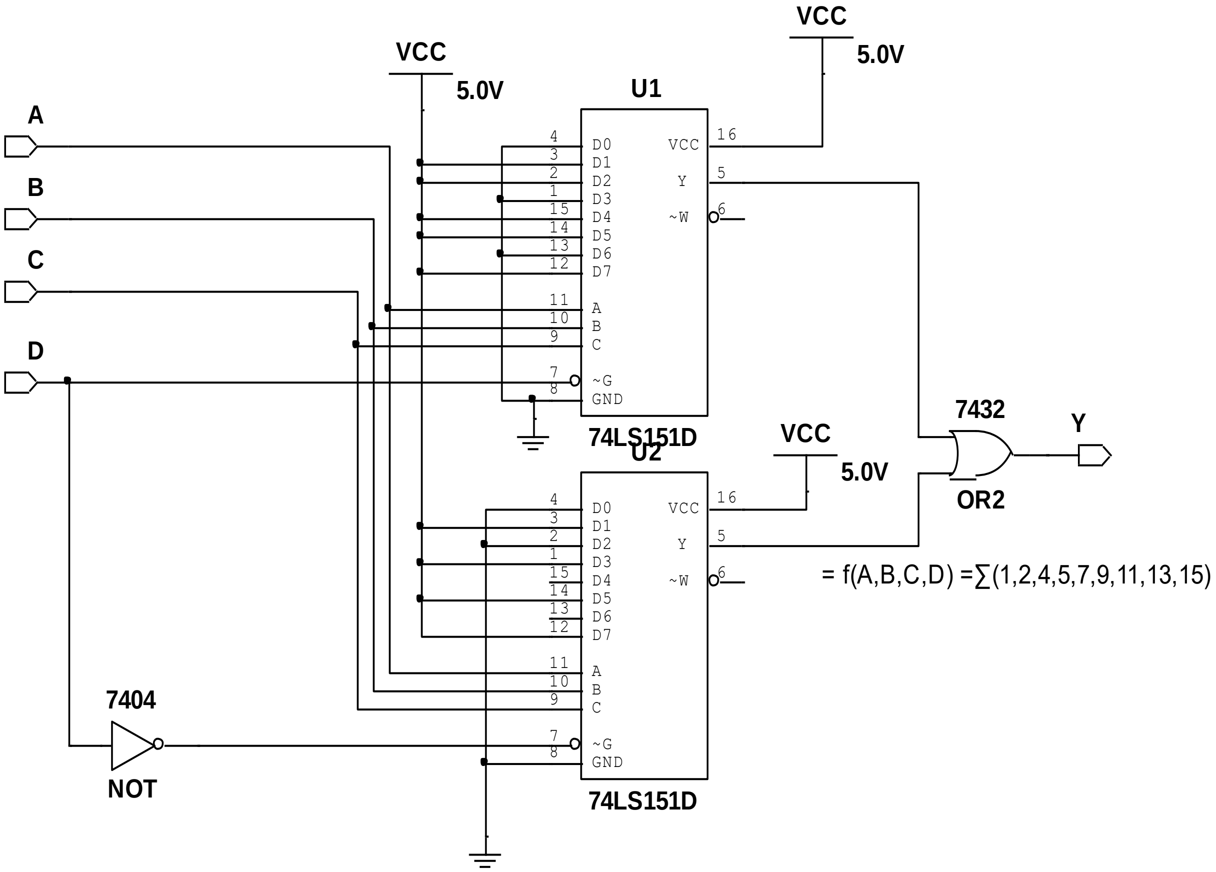


Truth Table

Enable input	Select Inputs			Outputs	
G	A	B	C	Y	W
1	x	x	x	0	1
0	0	0	0	D0	$\overline{D0}$
0	0	0	1	D1	$\overline{D1}$
0	0	1	0	D2	$\overline{D2}$
0	0	1	1	D3	$\overline{D3}$
0	1	0	0	D4	$\overline{D4}$
0	1	0	1	D5	$\overline{D5}$
0	1	1	0	D6	$\overline{D6}$
0	1	1	1	D7	$\overline{D7}$

c) Realize the given function F= f(A,B,C,D) =∑(1,2,4,5,7,9,11,13,15) using IC 74151(8:1 MUX)

Logic diagram:



Truth Table:

Select Inputs				**Output**
D	C	B	A	Y
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Procedure:

1. The IC is fixed on the IC zip socket and VCC &GND connections are given from 5V Supply.
2. Connections are made as shown in the Logic diagram.
3. All the inputs are connected to the switches & output to the LEDs.
4. Truth table is verified for different combinations of input.

Result: …………………………………………………………………………………

Staff Signature

VIVA Question

1. What is the difference between Adder and Substractor?
2. Applications of Adders and Substractor
3. What is multiplexer?
4. Give the applications of multiplexer?
5. What are the advantages of multiplexer?
6. Give the design of 8X1 multiplexer using 2X1 multiplexers?
7. What is difference between decoder and multiplexer?

Experiment No. 6

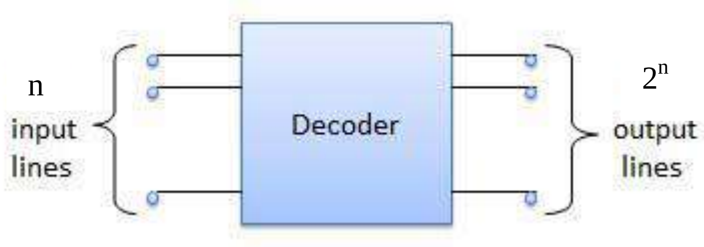
DECODER or De-MUX

Aim: To realize a Boolean expression using decoder IC74139.

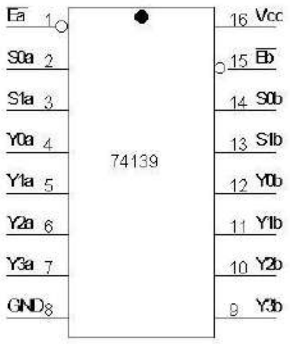
Components required:

Sl.No.	Particulars	Quantity
1	IC 74139	2 nos.
2	IC 7400, 7410,7427, 7432	1 each

Block diagram:



Pin Configuration:

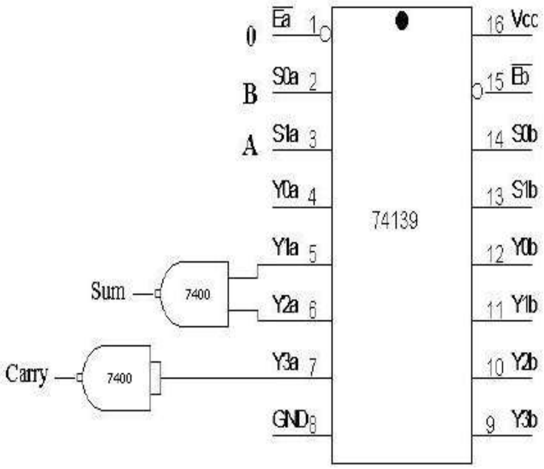


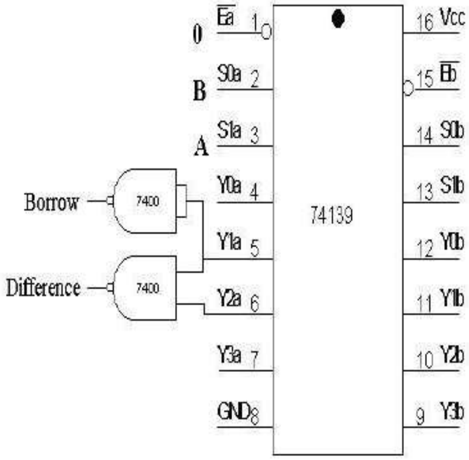
$\overline{Ea}$	S_{1a}	S_{0a}	$\overline{Y3a}$	$\overline{Y2a}$	$\overline{Y1a}$	$\overline{Y0a}$
1	X	X	1	1	1	1
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1

$\overline{Eb}$	S_{1b}	S_{0b}	$\overline{Y3b}$	$\overline{Y2b}$	$\overline{Y1b}$	$\overline{Y0b}$
1	X	X	1	1	1	1
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1

a) Implement Half adder and half Substractor using 74139

i) Half adder ii) Half Substractor



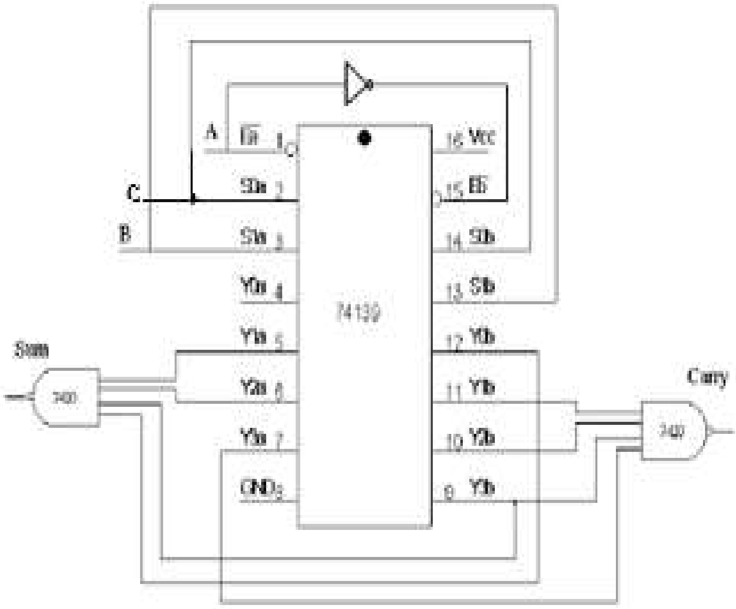


Sum =Σ m (1, 2) , Carry = Σ m (3) Difference = Σ m (1, 2) , Borrow = Σ m (1)

b) Implement Full adder and full Substractor using decoder 74139

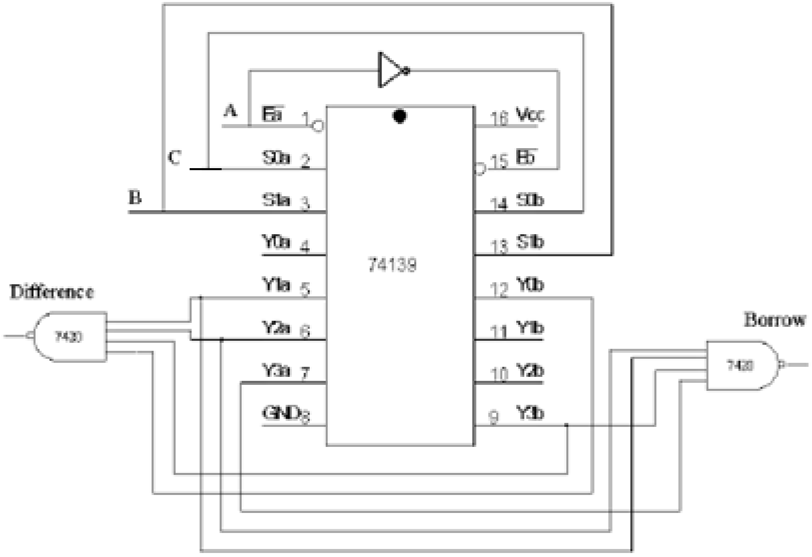
i) Full adder

SUM= ∑m (1, 2, 4, 7) CARRY= ∑ m (3, 5, 6, 7)



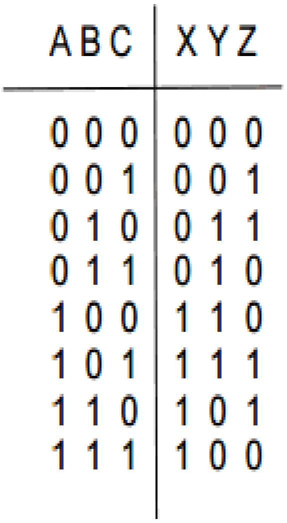
ii) Full Substractor

Diff= ∑ m (1, 2, 4, 7) Borrow = ∑m (1, 2, 3, 7)

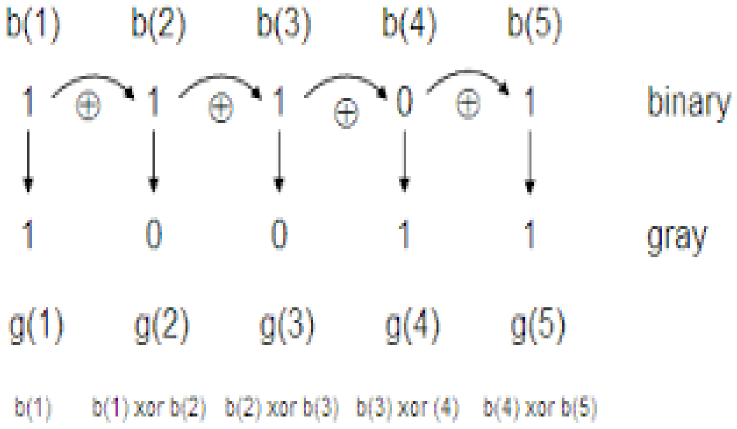


Note: Refer **truth tables** of adders and substractorsfromexperiment no. 5

i) Realize BINARY TO GREYCODE conversion using decoder 74139

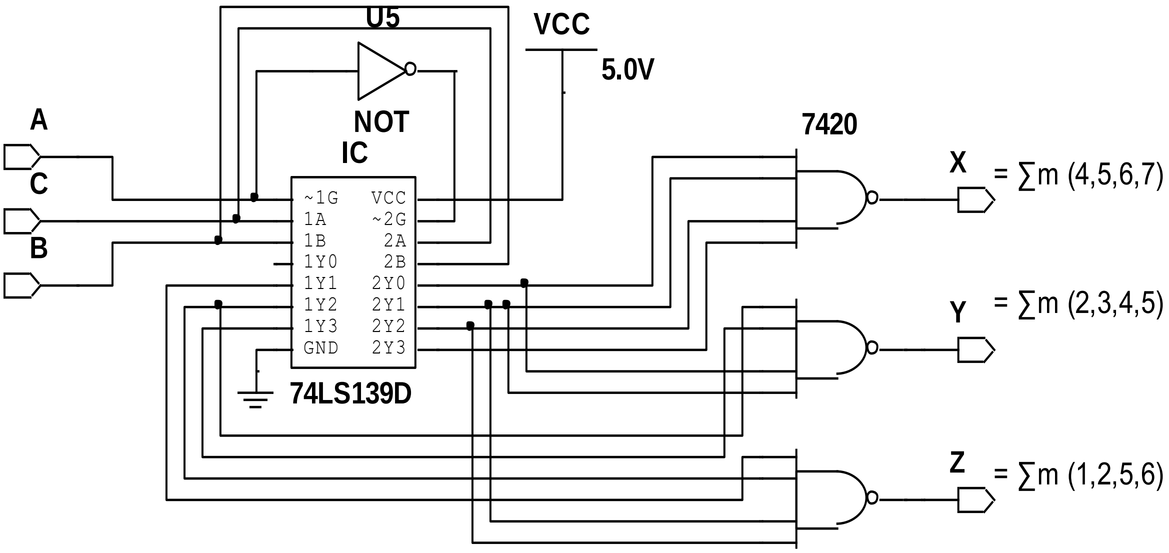


ABC	XYZ
0 0 0	0 0 0
0 0 1	0 0 1
0 1 0	0 1 1
0 1 1	0 1 0
1 0 0	1 1 0
1 0 1	1 1 1
1 1 0	1 0 1
1 1 1	1 0 0

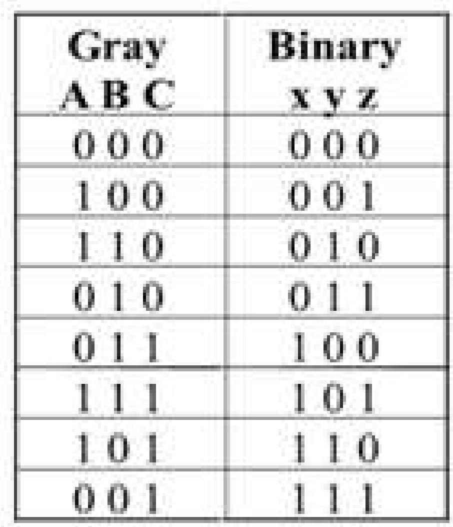


X (A,B,C)= ∑m (4,5,6,7) Y (A,B,C)= ∑m (2,3,4,5) Z (A,B,C)= ∑m (1,2,5,6)

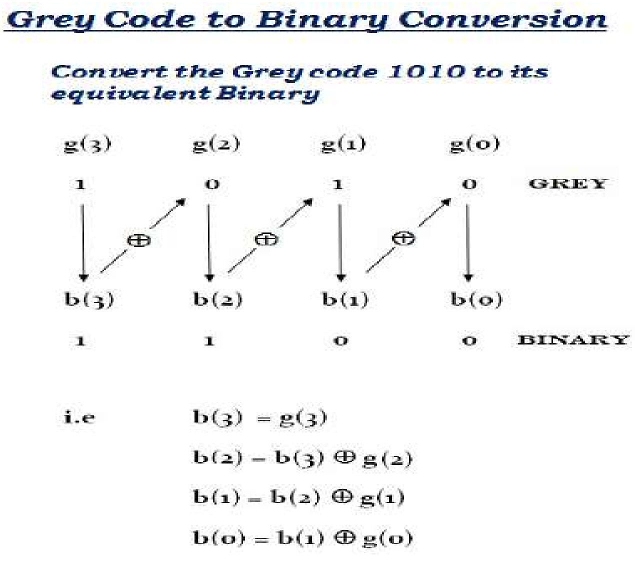
Logic diagram:



i) Realize GREY TO BINARY CODE conversion using decoder 74139



Gray A B C	Binary x y z
0 0 0	0 0 0
1 0 0	0 0 1
1 1 0	0 1 0
0 1 0	0 1 1
0 1 1	1 0 0
1 1 1	1 0 1
1 0 1	1 1 0
0 0 1	1 1 1



Grey Code to Binary Conversion

Convert the Grey code 1010 to its equivalent Binary

g(3)	g(2)	g(1)	g(0)	
1	0	1	0	GREY
b(3)	b(2)	b(1)	b(0)	
1	1	0	0	BINARY

i.e b(3) = g(3)

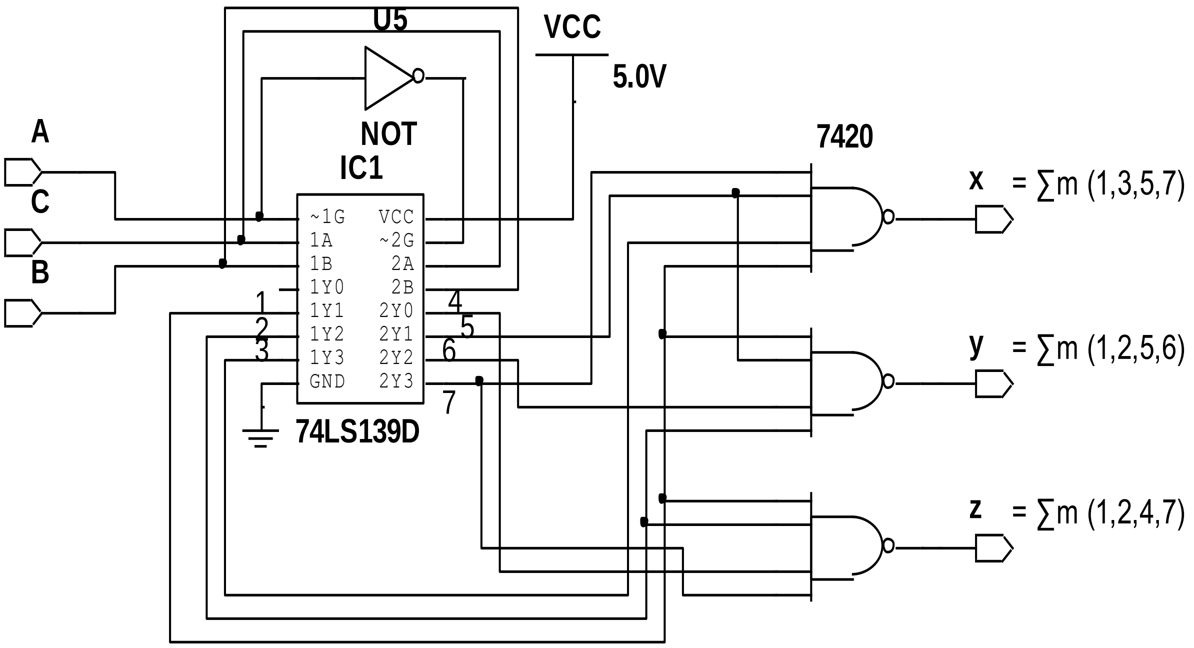
b(2) = b(3) ⊕ g(2)

b(1) = b(2) ⊕ g(1)

b(0) = b(1) ⊕ g(0)

x(A,B,C)= ∑m (1,3,5,7) y(A,B,C)= ∑m (1,2,5,6) z(A,B,C)= ∑m (1,2,4,7)

Logic diagram:



Procedure:

1. The IC is fixed on the IC zip socket and VCC &GND connections are given from 5V Supply.
2. Connections are made as shown in the Logic diagram.
3. All the inputs are connected to the switches & output to the LEDs.
4. Truth table is verified for different combinations of input.

Result: ……………………………………………………………………………………….

Staff Signature

VIVA Question

1. What is decoder?
2. What is priority encoder? What is difference between encoder and priority encoder?
3. What is difference between decoder and encoder?
4. Give the applications of multiplexer?
5. Give the design of 8X1 multiplexer using 2X1 multiplexers?
6. What is difference between decoder and multiplexer?

Experiment No.7

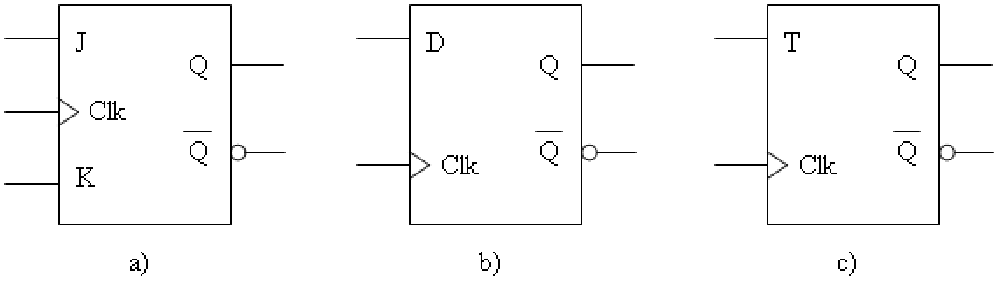
FLIP-FLOPS

Aim: Realization of Master-Slave JK, D & T Flip-Flops using NAND Gates.

Components Required:

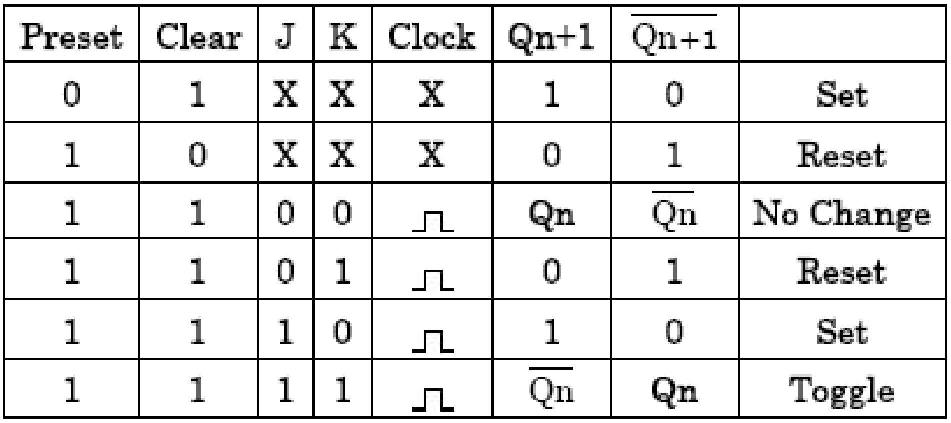
Sl.No	Particulars	Quantity
01	IC 7410	2 No
02	IC 7400	1 No

Logic symbol of Flip flops:



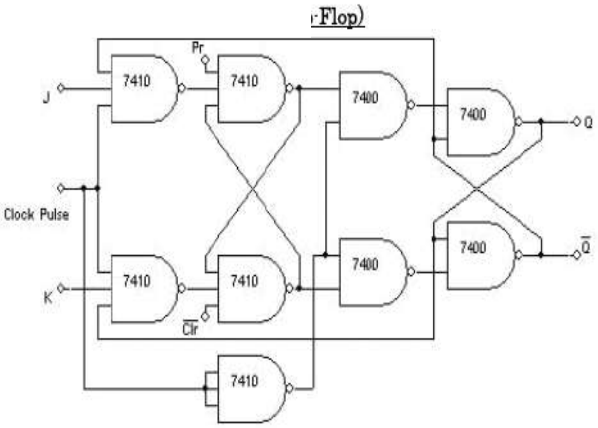
(a) Master Slave JK Flip flop using NAND gates

Truth Table:



Preset	Clear	J	K	Clock	Qn+1	$\overline{Qn+1}$	
0	1	X	X	X	1	0	Set
1	0	X	X	X	0	1	Reset
1	1	0	0	⎍	Qn	$\overline{Qn}$	No Change
1	1	0	1	⎍	0	1	Reset
1	1	1	0	⎍	1	0	Set
1	1	1	1	⎍	$\overline{Qn}$	Qn	Toggle

Logic Diagram:



Note:

Case 1: Asynchronous:

1. In the absence of clock.
   a) With preset = 0 and clear =1, the output is set.
   b) With preset = 1 and clear = 0, the output is reset.

Case 2: Synchronous:

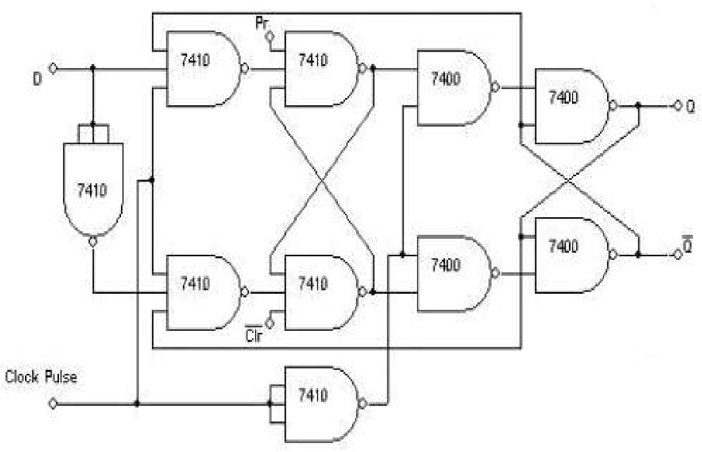
2. Both present and clear are made high.
3. All combination of inputs is applied at J & K.

(b) D-Flip flop using NAND gates

Truth table:

Inputs				Output	
Preset	**Clear**	**D**	**Clock**	**Qn + 1**	**$\overline{\text{Qn + 1}}$**
1	**1**	**0**	⎍	**0**	**1**
1	**1**	**1**	⎍	**1**	**0**

Logic Diagram:

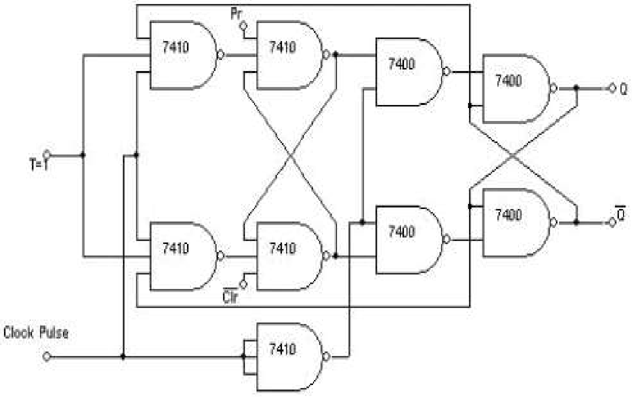


(c) T-Flip flop using NAND gates

Truth table:

Inputs				Output	
Preset	**Clear**	**T**	**Clock**	**Qn + 1**	**$\overline{Qn + 1}$**
1	**1**	**0**	⎍	**Qn**	**$\overline{Qn}$**
1	**1**	**1**	⎍	**$\overline{Qn}$**	**Qn**

Logic Diagram:



Procedure:

1.Connections are made as shown in Logic diagram.

2.The Truth Tables of flip flops are verified for various combinations of inputs.

Result: ……………………………………………………………………………………….

Staff Signature

VIVA Question

1. Give the difference between latches and flip-flops?
2. Draw and explain the working of

   a) SR flip-flop, b) Gated SR flip-flop, c)Gated D latch
3. Explain any one application of SR latch?
4. Draw the logic diagram, construct the excitation table and write the characteristic equations for the following flip-flop?

   a) SR flip-flop, b) JK flip-flop, c) T or Toggle flip-flop, d) D or Delay flip-flop
5. What is race around condition? How it is avoided?
6. Explain the advantage of JK flip-flop over SR flip-flop?
7. Sketch the logic diagram of a master-slave flip-flop?.Explain its operation and features?

Experiment No. 8

SHIFT REGISTERS

Aim: To realize

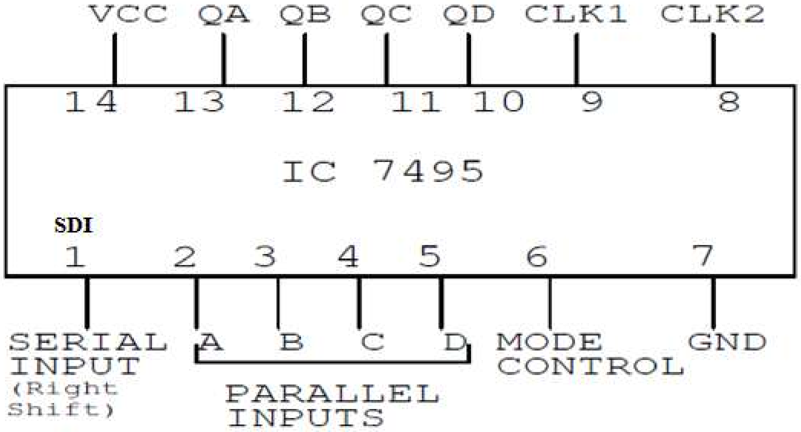
a) The following shift operations using IC 7474/7495.i) SISO ii) SIPO (Right shift)iii) PISO iv)PIPO

b) The ring and Johnson counter using IC 7495

Components required:

Particulars	Quantity
IC 7495	1 No
IC 7404	1 No

Pin configuration of IC 7495



Truth Table:

Operating mode	INPUT					OUTPUT			
	Mode control (S)	**Clk2 (CP1)**	**Clk1 (CP2)**	**SDI (DS)**	**PARALLEL INPUT (Pn) (A B C D)**	**QA**	**QB**	**QC**	**QD**
SHIFT	L	↓	X	L	X	L	qb	qc	qd
	L	↓	X	H	X	H	qb	qc	qd
Parallel Load	H	X	↓	X	Pn (1011)	P0 1	P1 0	P2 1	P3 1

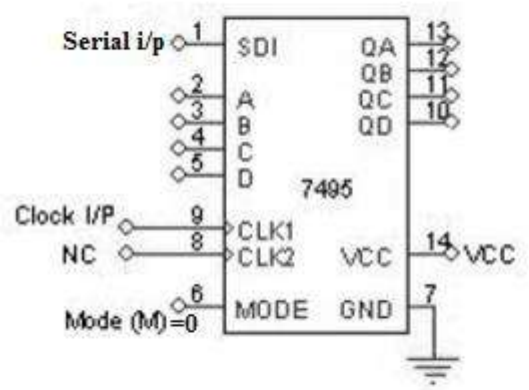
a) Shift registers operations

- **SDI:** Serial data input(to be shifted)
- **A,B,C,D:** Parallel data inputs to be loaded into the shift register.
- **Mode control (M)**

  Keep, M=1 for loading parallel data and to enable clock 2.

  M=0 for enabling clock 1
- **Clk 2:** For loading parallel input data and for shift left of data.
- **Clk 1:** For right shift of data.
- **QA, QB, QC and QD:** Parallel outputs of the shift register.

(i) Serial In Serial Out (SISO)



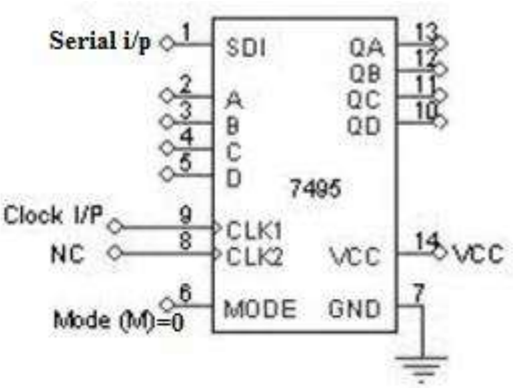
Tabular Column

Inputs		Outputs			
Clock	**Serial i/p**	**QA**	**QB**	**QC**	**QD**
1	**d0=0**	0	X	X	X
2	**d1=1**	1	0	X	X
3	**d2=1**	1	1	0	X
4	**d3=1**	1	1	1	**0=d0**
5	X	X	1	1	**1=d1**
6	X	X	X	1	**1=d2**
7	X	X	X	X	**1=d3**

Procedure:

1. Connections are made as per circuit diagram.
2. Load the shift register with 4 bits of data one by one serially.
3. At the end of 4th clock pulse the first data ‘d0’ appears at QD.
4. Apply another clock pulse; the second data ‘d1’ appears at QD.
5. Apply another clock pulse; the third data appears at QD.
6. Application of next clock pulse will enable the 4th data ‘d3’ to appear at QD. Thus the data applied serially at the input comes out serially at QD.

(ii) Serial In Parallel Out SIPO (Right Shift)



Tabular Column

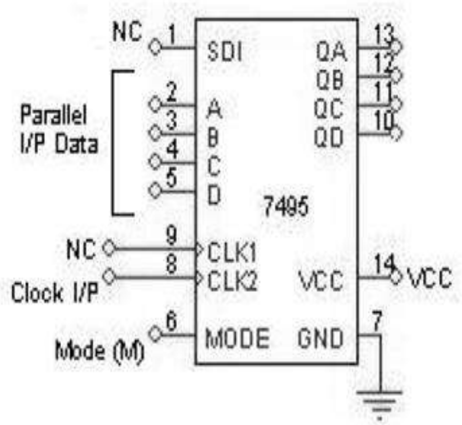
Inputs		Outputs			
Clock	**Serial i/p**	**QA**	**QB**	**QC**	**QD**
1	**d0=0**	0	X	X	X
2	**d1=1**	1	0	X	X
3	**d2=1**	1	1	0	X
4	**d3=1**	**d0=1**	**d1=1**	**d2=1**	**d3=0**

Procedure:

1. Connections are made as per circuit diagram.
2. Apply the data at serial i/p
3. Apply one clock pulse at clock 1 (Right Shift) observe this data at QA.
4. Apply the next data at serial i/p.
5. Apply one clock pulse at clock 2, observe that the data on QA will shift to QB and the new data applied will appear at QA.
6. Repeat steps 2 and 3 till all the 4 bits data are entered one by one into the shift register.

(iii) Parallel In Serial Out (PISO)

Tabular Column



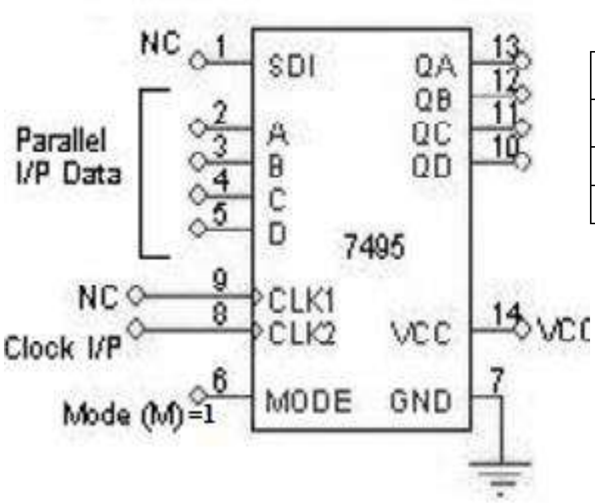
Mode	Clk	Parallel i/p				Serial o/p			
		A	**B**	**C**	**D**	**QA**	**QB**	**QC**	**QD**
1	1	1	0	1	1	1	0	1	1=d0
0	2	X	X	X	X	X	1	0	1=d1
0	3	X	X	X	X	X	X	1	0=d2
0	4	X	X	X	X	X	X	X	1=d3

Note: Mode M = 1 for Parallel loading.
Mode M = 0 for serial shifting.

Procedure:

1. Connections are made as per circuit diagram.
2. Apply the desired 4 bit data at A, B, C and D.
3. Keeping the mode control M=1 apply one clock pulse. The data applied atA, B, C and D will appear at QA, QB, QC and QD respectively.
4. Now mode control M=0. Apply clock pulses one by one and observe the data coming out serially at QD.

(iv) Parallel In Parallel Out (PIPO)



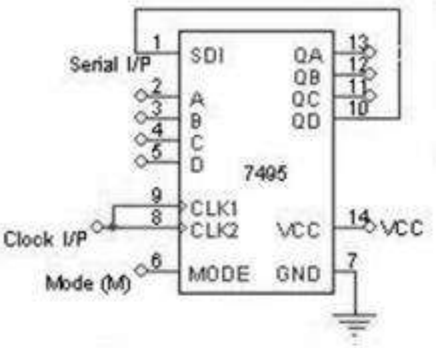
Tabular Column

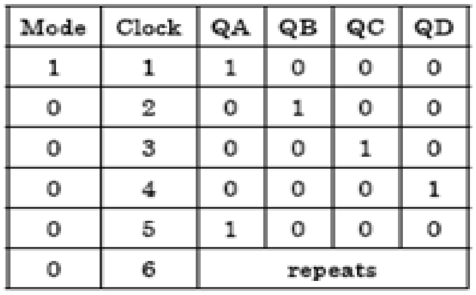
Inputs					Outputs			
Clock	**Parallel i/p**				**Parallel o/p**			
	A	**B**	**C**	**D**	**QA**	**QB**	**QC**	**QD**
1	1	0	1	1	1	0	1	1

Procedure: -

1. Connections are made as per circuit diagram.
2. Apply the 4 bit data at A, B, C and D.
3. Apply one clock pulse at Clock 2 (Note: Mode control M=1).
4. The 4 bit data at A, B, C and D appears at QA, QB, QC and QD respectively.

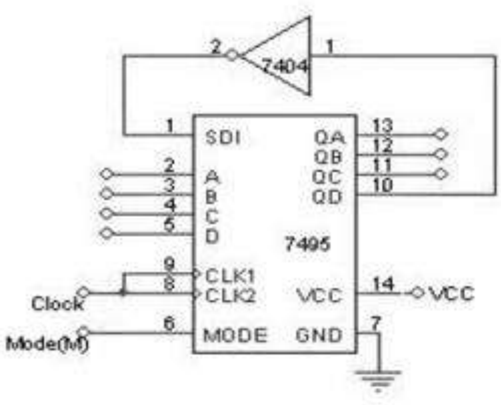
b) Ring counter using IC 7495 Counting sequence

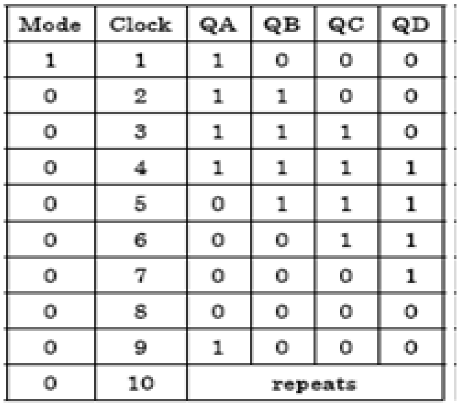




Mode	Clock	QA	QB	QC	QD
1	1	1	0	0	0
0	2	0	1	0	0
0	3	0	0	1	0
0	4	0	0	0	1
0	5	1	0	0	0
0	6	repeats			

c) Johnson counter using IC 7495 Counting sequence





Mode	Clock	QA	QB	QC	QD
1	1	1	0	0	0
0	2	1	1	0	0
0	3	1	1	1	0
0	4	1	1	1	1
0	5	0	1	1	1
0	6	0	0	1	1
0	7	0	0	0	1
0	8	0	0	0	0
0	9	1	0	0	0
0	10	repeats			

Procedure: -

1. Connections are made as per the circuit diagram.
2. Apply the data 1000 at A, B, C and D respectively.
3. Keeping the mode M = 1, apply one clock pulse.
4. Now the mode M is made 0 and clock pulses are applied one by one and the truth table is verified.
5. Above procedure is repeated for Johnson counter also.

Result: **---**

Staff Signature

VIVA Question

1. What is the function of shift register
2. What is shift left register?
3. How does a Johnson counter work?
4. What is shift register application?
5. Explain the different types of triggering?
6. What do you mean by sequential circuits?
7. Give the comparison between combinational and sequential circuits?
8. Give the comparison between synchronous and asynchronous circuits?
9. Draw and explain the working of basic bistable element?

Experiment No.9

COUNTERS

Aim: Realize i) Design MOD- N synchronous UP counter and DOWN Counter Using 7476 JK Flip-flop

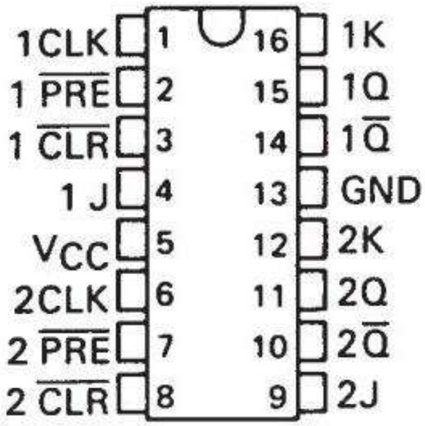
ii) Mod N Asynchronous counter using IC7490 and

(iii) Mod-N Synchronous counter using IC74192

Components required:

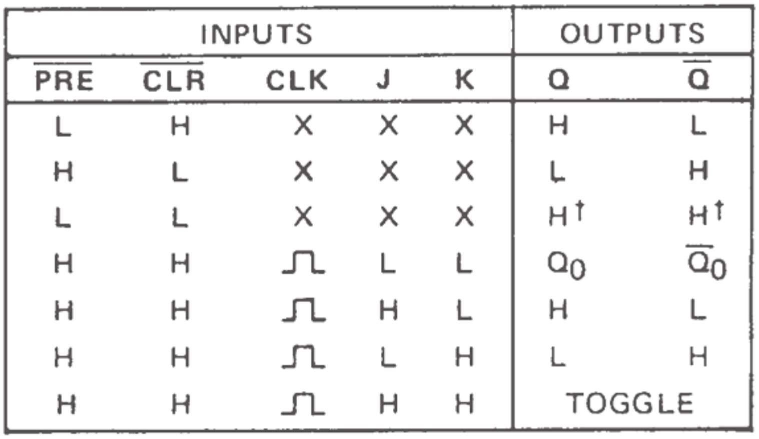
Sl.No	Particulars	Quantity
01	IC 7490	1
02	IC 74192	1
03	IC 7476,7404	1 Each

i) Design MOD- N synchronous UP counter and DOWN Counter using 7476 JK Flip-flop



Pin diagram of 7476

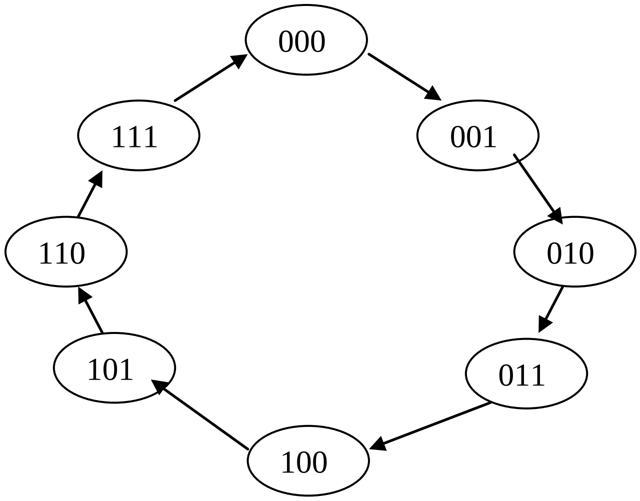
FUNCTION TABLE



INPUTS					OUTPUTS	
$\overline{PRE}$	$\overline{CLR}$	CLK	J	K	Q	$\overline{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H†	H†
H	H	⎍	L	L	Q_0	$\overline{Q}_0$
H	H	⎍	H	L	H	L
H	H	⎍	L	H	L	H
H	H	⎍	H	H	TOGGLE	

3 bit/ Mod 8 Synchronous Up counter for the given sequence using IC 7476.

Given State Diagram



Design:

State Table

Clock	QC	QB	QA
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Excitation Table for JK Flip flop

State Change		J-K Input	
Qn	Qn+1	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Transition Table

Present State			Next State			Flip flop Inputs					
						FF-2		FF-1		FF-0	
QC	QB	QA	QC	QB	QA	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	1	1	1	X	0	X	0	1	X
1	1	1	0	0	0	X	1	X	1	X	1

Simplification for Flip Flop inputs in terms of present state:

For J_A:

	QB'QA'	QB'QA	QBQA	QBQA'
QC'	1	X	X	1
QC	1	X	X	1

$J_A = 1$

For K_A:

	QB'QA'	QB'QA	QBQA	QBQA'
QC'	X	1	1	X
QC	X	1	1	X

$K_A = 1$

For J_B:

	QB'QA'	QB'QA	QBQA	QBQA'
QC'	0	1	X	X
QC	0	1	X	X

$J_B = QA$

For K_B :

	QB'QA'	QB'QA	QBQA	QBQA'
QC'	X	X	1	0
QC	X	X	1	0

$K_B = QA$

For J_C:

	QB'QA'	QB'QA	QBQA	QBQA'
QC'	0	0	1	0
QC	X	X	X	X

$J_C = QA.QB$

For K_C:

	QB'QA'	QB'QA	QBQA	QBQA'
QC'	X	X	X	X
QC	0	0	1	0

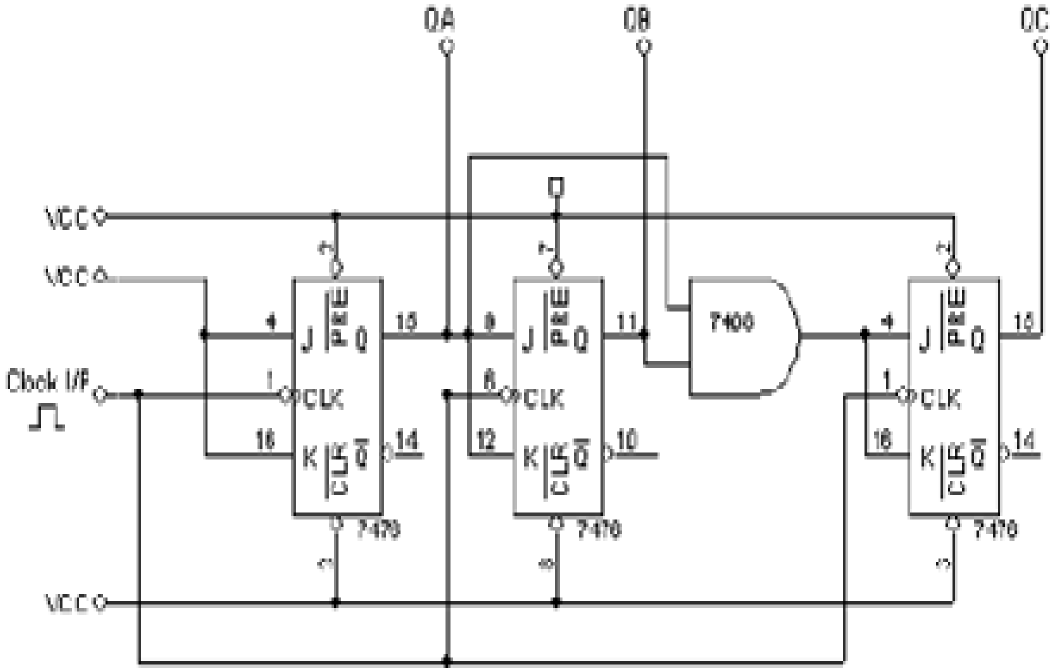
$K_C=QA.QB$

FF-0 :- $J_A= 1, K_A = 1$

FF-1 :- $J_B = QA, K_B = QA$

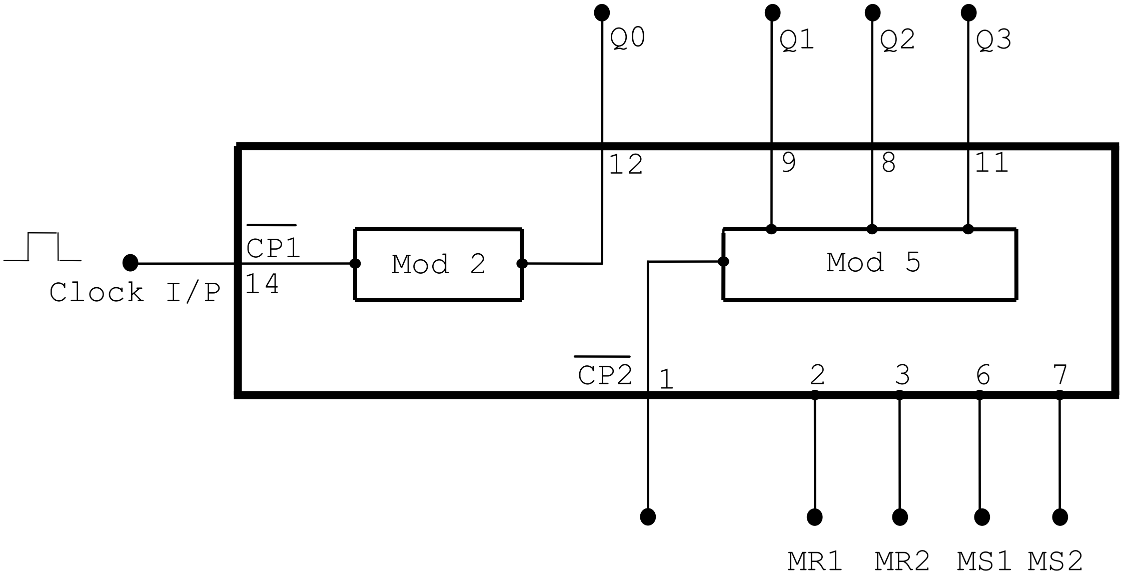
FF-2 :- $J_C= QA.QB, K_C = QA.QB$

Logic Diagram for 3-bit Synchronous up counter using discrete components:



ii)Asynchronous counter using IC7490

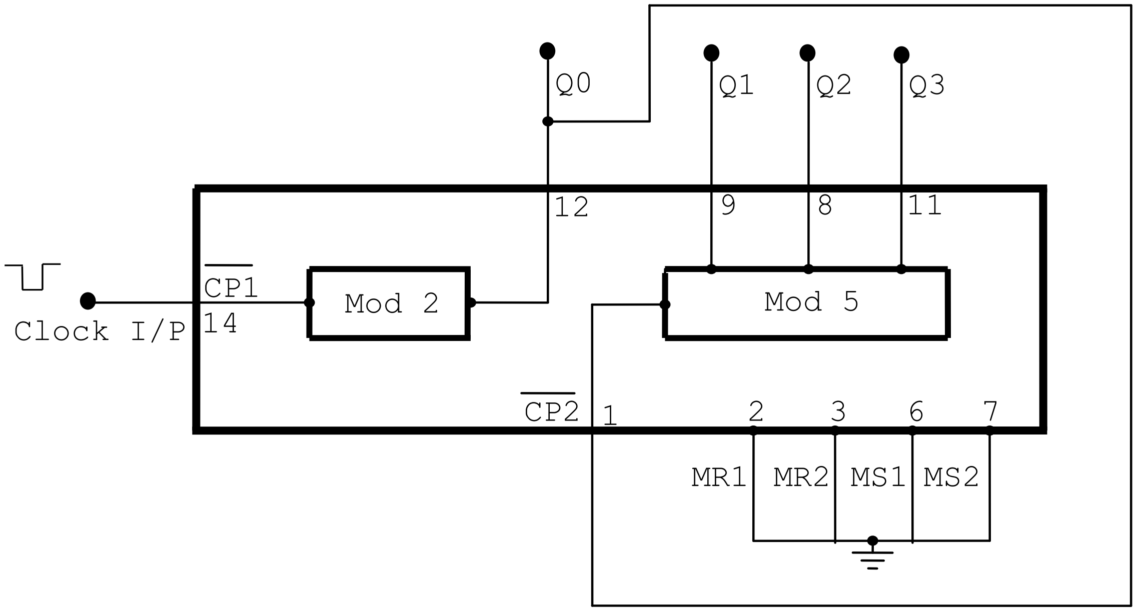
Internal Diagram of 7490:



Function Table:

Clock	MR1	MR2	MS1	MS2	Q3	Q2	Q1	Q0	Remarks
X	1	1	0	X	0	0	0	0	Reset
X	1	1	X	0	0	0	0	0	Reset
X	X	X	1	1	1	0	0	1	Set to 9
⎍	X	0	X	0	Count				
⎍	0	X	0	X	Count				
⎍	0	X	X	0	Count				
⎍	X	0	0	X	Count				

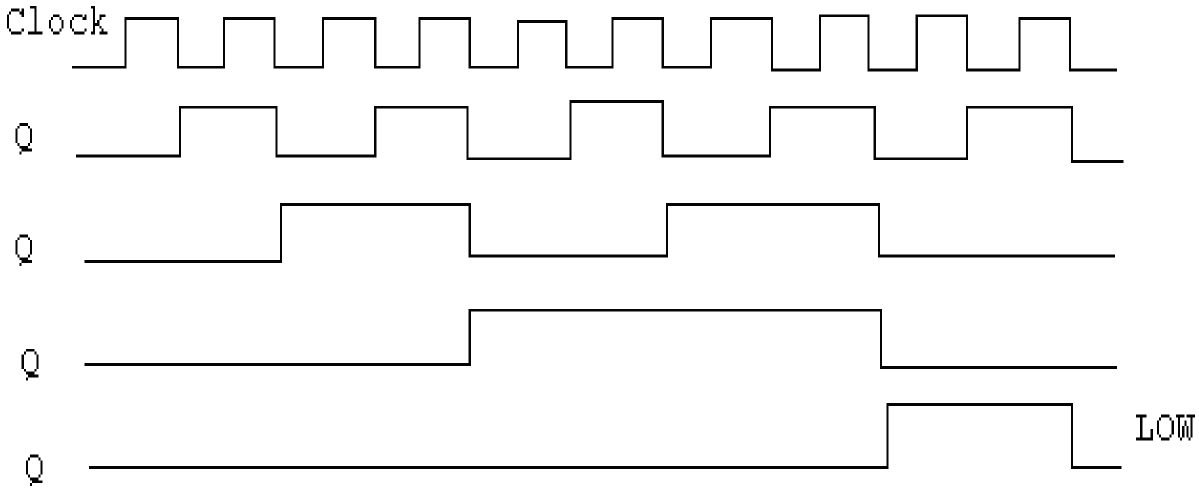
Logic Diagram of Mod-10 counter(Decade Counter):



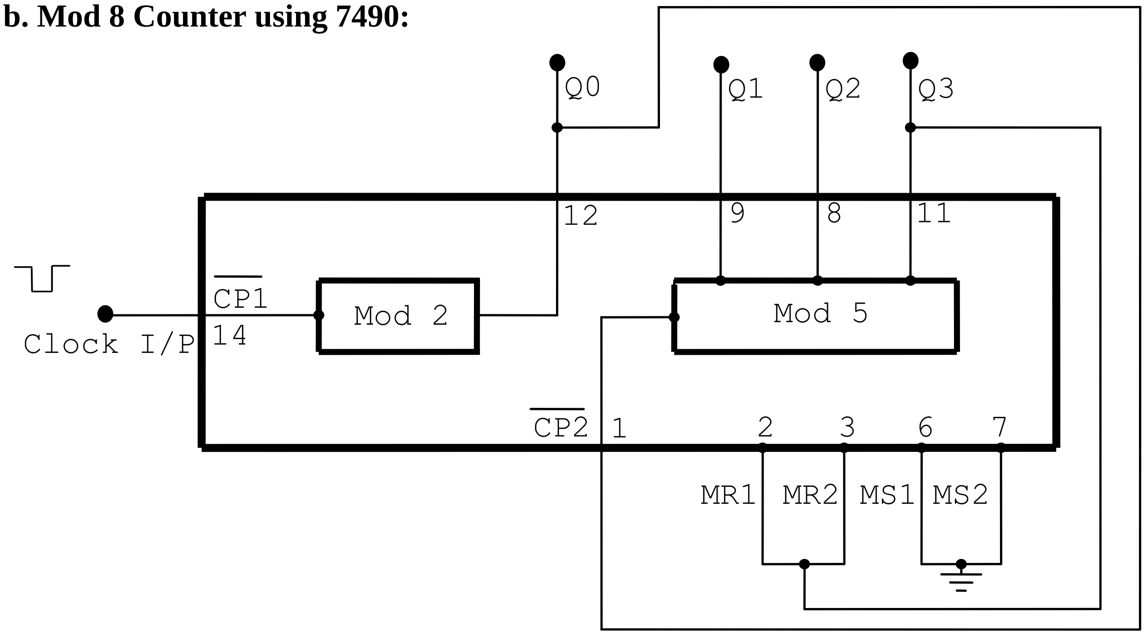
Truth Table of Mod-10 Counter:

Clock	Q3	Q2	Q1	Q0
⎍	0	0	0	0
⎍	0	0	0	1
⎍	0	0	1	0
⎍	0	0	1	1
⎍	0	1	0	0
⎍	0	1	0	1
⎍	0	1	1	0
⎍	0	1	1	1
⎍	1	0	0	0
⎍	1	0	0	1
⎍	0	0	0	0

Timing diagram of Mod 10 Counter:



b. Mod 8 Counter using 7490:

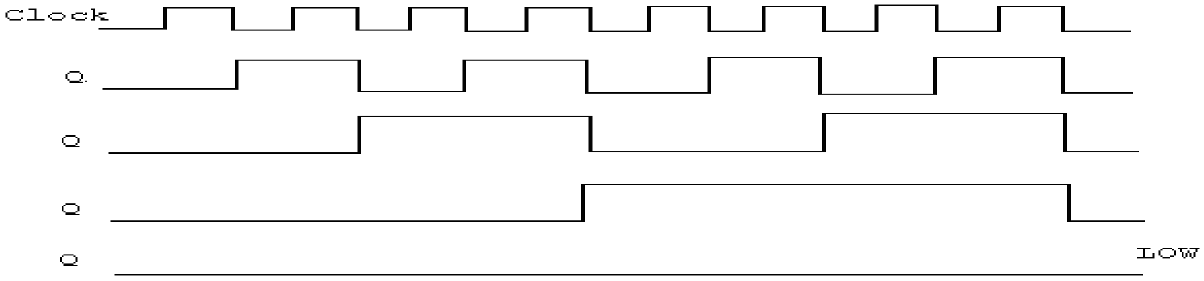


Truth Table for Mod 8 Counter:

Inputs	Outputs			
Clock	Q3	Q2	Q1	Q0
⎍	0	0	0	0
⎍	0	0	0	1
⎍	0	0	1	0
⎍	0	0	1	1
⎍	0	1	0	0
⎍	0	1	0	1
⎍	0	1	1	0
⎍	0	1	1	1
⎍	0	0	0	0

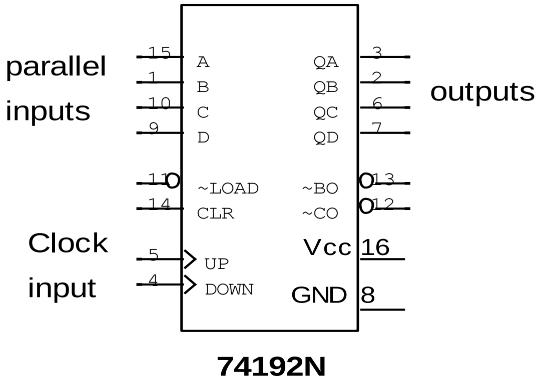
At the 8th clock pulse reset the counter

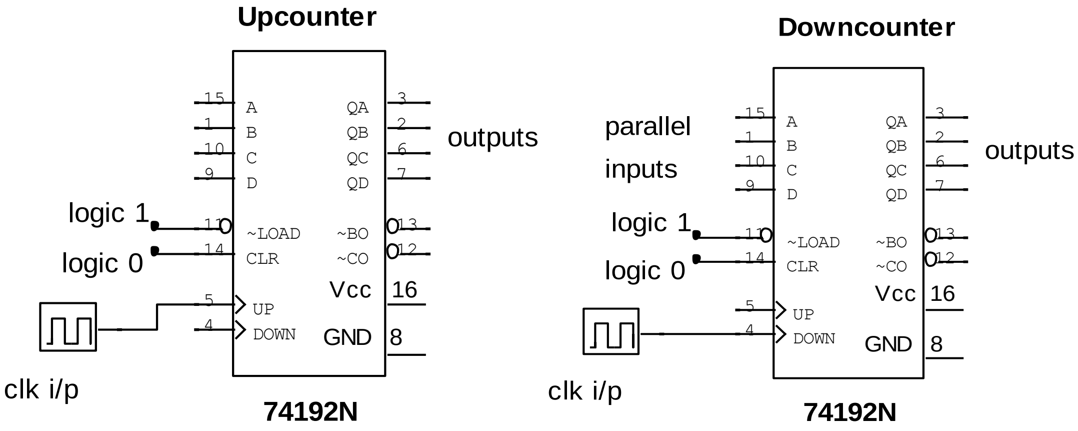
Timing diagram of Mod-8 counter:



iii) Synchronous counter using IC74192

Pin Configuration:





Function table:

Clr	Load	UP(C_U)	Down(C_D)	Mode
1	X	X	X	Reset
0	0	X	X	Preset
0	1	1	1	No change
0	1	Clock i/p (L-H)	1	Count up
0	1	1	Clock i/p (L-H)	Count down

Truth table:

Input(clock) (C_U / C_D)	Up counter				Down counter			
	Q_D	Q_C	Q_B	Q_A	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0	1	0	0	1
1	0	0	0	1	1	0	0	0
2	0	0	1	0	0	1	1	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	0	1
5	0	1	0	1	0	1	0	0
6	0	1	1	0	0	0	1	1
7	0	1	1	1	0	0	1	0
8	1	0	0	0	0	0	0	1
9	1	0	0	1	0	0	0	0

Procedure:

1. Connections are made as shown in Logic diagram.
2. Verify the function table
3. Verify the truth table of logic circuit of up/down counter by applying respective clock pulses.

Result: ……………………………………………………………………………………….

Staff Signature

VIVA Question:

1. What is counter?
2. What is ring counter?
3. What are synchronous counters?
4. How many Flip-Flops are required for mod–16 counter?
5. Difference between synchronous and asynchronous?
6. Explain the working of 4 bit synchronous counter?
7. Explain the working of 4 bit asynchronous counter?
8. What is advantages and disadvantages of counter?
9. What is difference between synchronous counter and asynchronous counter?
10. Why counter is called as clock divider? Explain?
11. Design a BCD counter with JK flip-flops?
12. Design a counter with the following binary sequence 0,1,9,3,2,8,4 and repeat? Use T flip-flops?
13. Design a counter with the following binary sequence 0,1,9,3,2,8,4 and repeat? Use JK flip-flops?
14. Design a counter which counts 1 to 10 using flip-flops.
15. Define Metastablity?

Experiment No.10

SEQUENCE GENERATOR

Aim: Design a pseudo random sequence generator using IC7495

Components required:

Sl. No	Particulars	Quantity
01	IC 7495, 7486, 7410	1 No

Sequence generator A digital logic circuit whose purpose is to produce a prescribed **sequence** of outputs. Each output will be one of a number of symbols or of binary or q-ary logic levels. The **sequence** may be of indefinite length or of predetermined fixed length. A binary counter is a special type of **sequence generator**.

$$L \leq 2^n - 1$$

L = Length of sequence

n = No of Flip Flop

(i) Design for the sequence, S = 1101011(Sequence length = 7 bits)

$$7 \leq 2^n - 1$$

No of flip flop = 3
Note: If same sequence is repeated to differentiate add one more flip flop.

Truth table:

Inputs				Output	
QA	**QB**	**QC**	**QD**	**Y**	**Decimal no.**
1	1	1	0	1	14
1	1	1	1	0	15
0	1	1	1	1	7
1	0	1	1	1	11
0	1	0	1	1	5
1	0	1	0	1	10
1	1	0	1	0	13

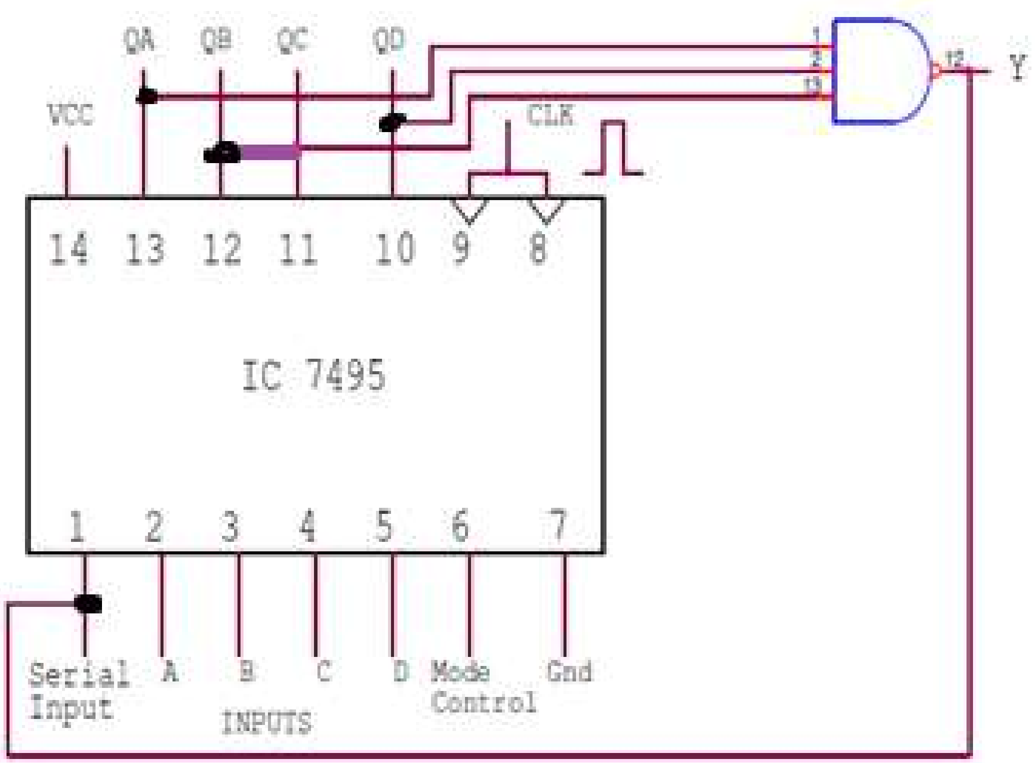
K- Map for above truth table

QA QB \ QC QD	00	01	11	10
00	X	X	X	X
01	X	1	1	X
11	X	0	0	1
10	X	X	1	1

$$Y = \overline{QA} + \overline{QB} + \overline{QD}$$

$$Y = \overline{QA\ QB\ QD}$$

Logic Diagram:



(ii) Design for the sequence, S = 100010011010111(Sequence length = 15 bits)

$$\mathbf{15 \le 2^{n} - 1}$$

No of flip flop = 4

Truth table:

Map value	Clock	Inputs				Output
		QA	QB	QC	QD	Y
15	1	1	1	**1**	**1**	0
7	2	0	1	1	**1**	1
3	3	0	0	1	1	1
1	4	0	0	0	1	1
8	5	1	0	0	0	1
4	6	0	1	0	0	0
2	7	0	0	1	0	0
9	8	1	0	0	1	0
12	9	1	1	0	0	1
6	10	0	1	1	0	0
11	11	1	0	1	1	0
5	12	0	1	0	1	1
10	13	1	0	1	0	1
13	14	1	1	0	1	0
14	15	1	1	1	0	**1**

K- Map for above truth table

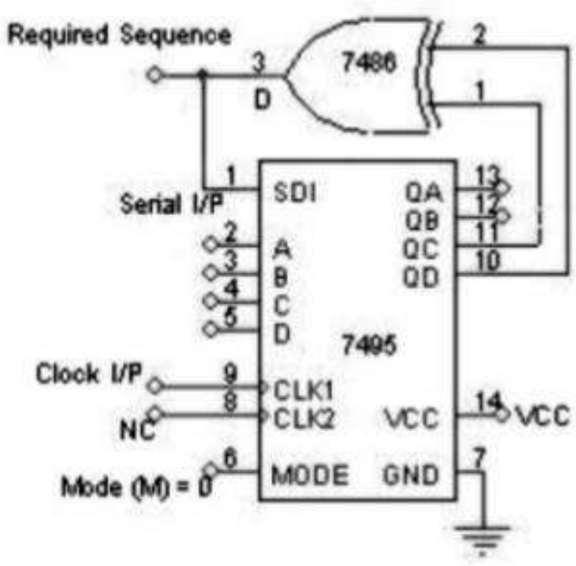
QC QD / QA QB	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

$$Y = \overline{QC}\,QD + \overline{QD}\,QC$$

$Y = QC \oplus QD$

Note: For 2nd sequence S = 100010011010111 write the logic circuit same as previous replacing NAND gate by X-OR gate as per simplified expression.

Logic Diagram:



Procedure :

1. Make the connection as shown in the circuit diagram.
2. By Keeping mode=1, Load the input A, B, C, D as in Truth table 1st Row and apply a clock pulse.
3. For count mode make mode=0 and apply the clock pulses
4. Verify the Truth Table and observe the outputs.

Result: **---**

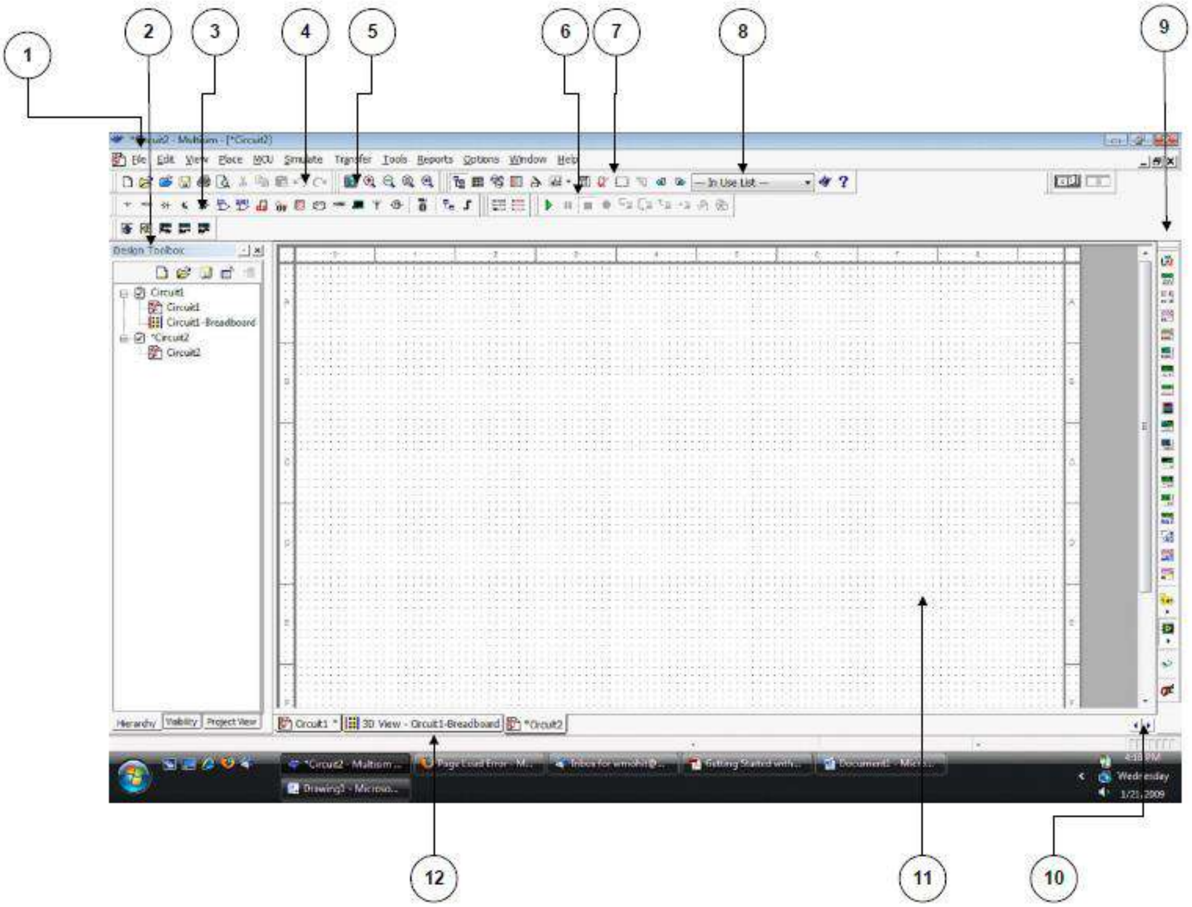
Staff Signature

VIVA Question

1. What is a MOD N counter?
2. How many flip flops are needed for MOD 32 binary counter?
3. Why flip flops are used in Counters?
4. What is BCD counter?
5. How many flip flops are needed for MOD 6 counter?
6. Sketch a full period of a clock waveform on sketch identify the following
7. Rise time,b)fall time,c)Period time, d)one pulse width
8. Draw and explain the block diagram of Moore model.
9. Draw and explain the block diagram of Melay model.
10. What is the difference between Melay and Moore FSM?
11. Explain the termsInput variable, b)Output variable, c)State variable, d)Excitation variable
12. Explain the present state and next state condition?
13. Explain the state diagram with an example?
14. Explain the state table with an example?
15. Is it able to interchange integrated circuits from the TTL and CMOS families? Justify your answer.

INTRODUCTION TO MULTISIM

Multisim is the schematic capture and simulation application of National Instruments Circuit Design Suite,a suite of EDA (Electronic Design Automation) tools. It is similar to PSpice, but it is more easy to use inpractical sense and has lots of features to make circuit drawing/simulating, a really simple task. Here iswindow of multisim, as it appears first time when you start the software.



1. The Menu Bar is where you find commands for all functions.

2. The Design Toolbox lets you navigate through the different types of files in a project (schematics,PCBs, reports), view a schematic’s hierarchy and show or hide different layers.

3. The Component toolbar contains buttons that let you select components from the Multisim databases for placement in your schematic.

4. The Standard toolbar contains buttons for commonly-performed functions such as Save, Print, Cut,and Paste.

5. The View toolbar contains buttons for modifying the way the screen is displayed.

6. The Simulation toolbar contains buttons for starting, stopping, and other simulation functions.

7. The Main toolbar contains buttons for common Multisim functions.

8. The In Use List contains a list of all components used in the design.

9. The Instruments toolbar contains buttons for each instrument.

10. Scroll Left –right is to ensure ease in handling larger designs.

11. The Circuit Window (or workspace) is where you build your circuit.

12. Active tab indicates the current active circuit window.

Experiment No.11

Serial Adder with Accumulator

Aim: Design Serial Adder with accumulator and simulate using NI Multi simulation tool.

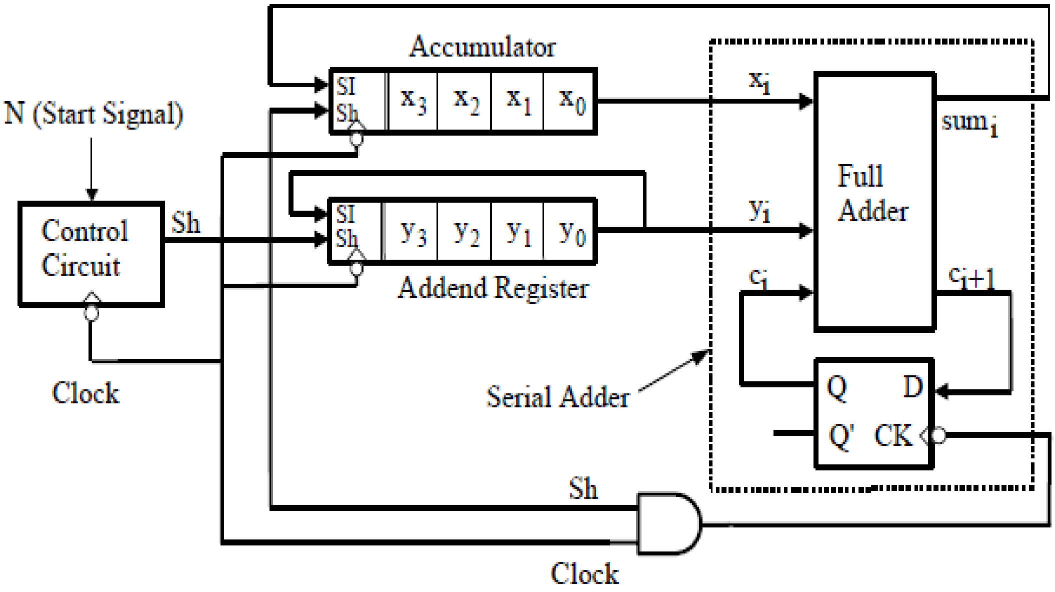
Introduction:

- The full adder is used to perform bit by bit addition and D-Flip flop is used to store the carry output generated after addition.

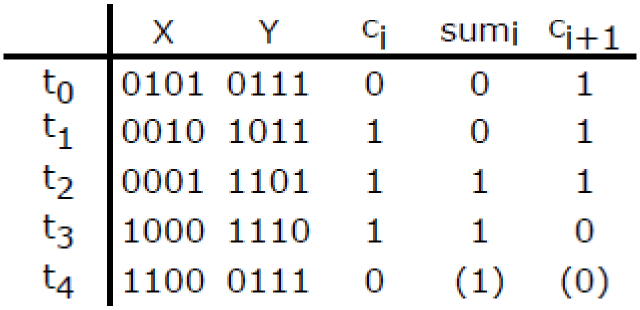
- This carry is used to carry input for the next addition. Initially the D Flip flop is cleared and addition starts with the least significant bits of both register.

- After each clock pulse data within the right shift registers are shifted right 1-bit and We get from next digit and carry of precious addition as new inputs for the full adder.

Block Diagram:

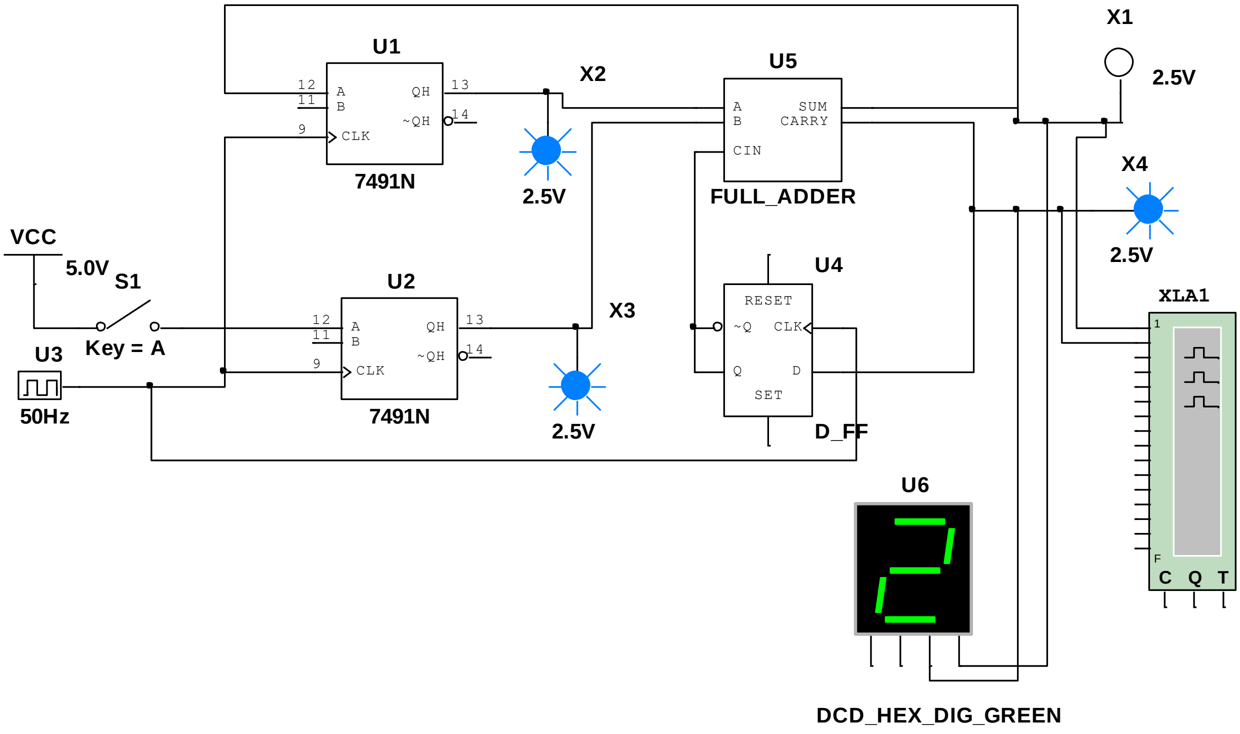


Truth table:



	X	Y	c_i	sum_i	c_{i+1}
t_0	0101	0111	0	0	1
t_1	0010	1011	1	0	1
t_2	0001	1101	1	1	1
t_3	1000	1110	1	1	0
t_4	1100	0111	0	(1)	(0)

Simulation Diagram:



Result: --

Staff Signature

Experiment No.12

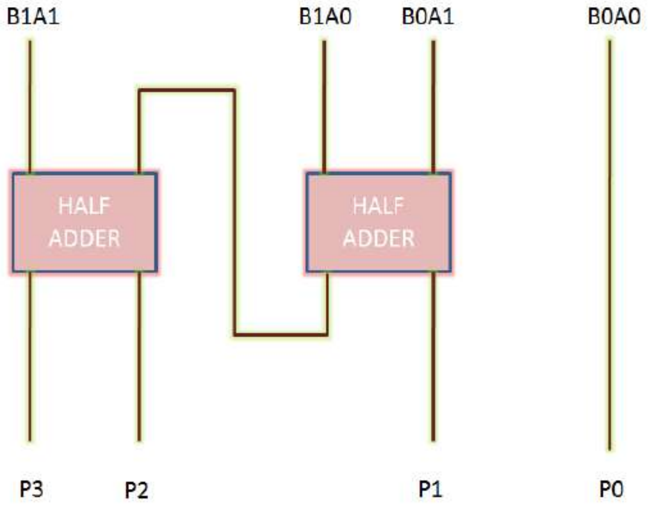
BINARY MULTIPLIER

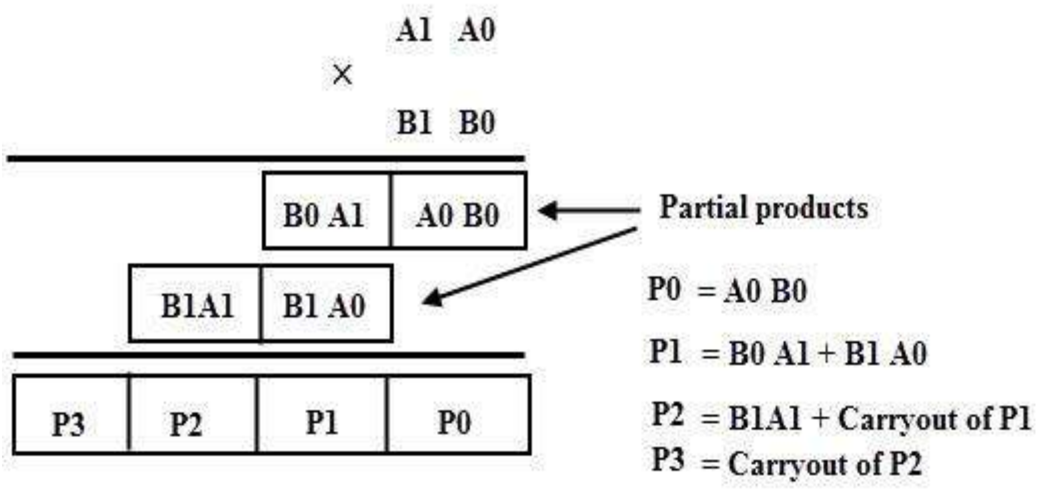
Aim: Design Binary Multiplier and simulate using NI Multi simulation tool.

2 BIT MULTIPLIER:

Let us consider two unsigned 2 bit binary numbers A and B to generalize the multiplication process. The multiplicand A is equal to A1, A0 and the multiplier B is equal to B1, B0. The belowexample shows the multiplication process of two 2 bit binary numbers. This process involves the multiplication of two digits and the addition of digits with or without carry. After the multiplication of the each bit to the multiplicand, partial products are generated, and then these products are added to produce the total sum which represents the binary multiplication value.

This multiplication is implemented by combinational circuit such that the multiplication is performed with AND gates whereas the addition is carried out by using half adders as shown in figure.

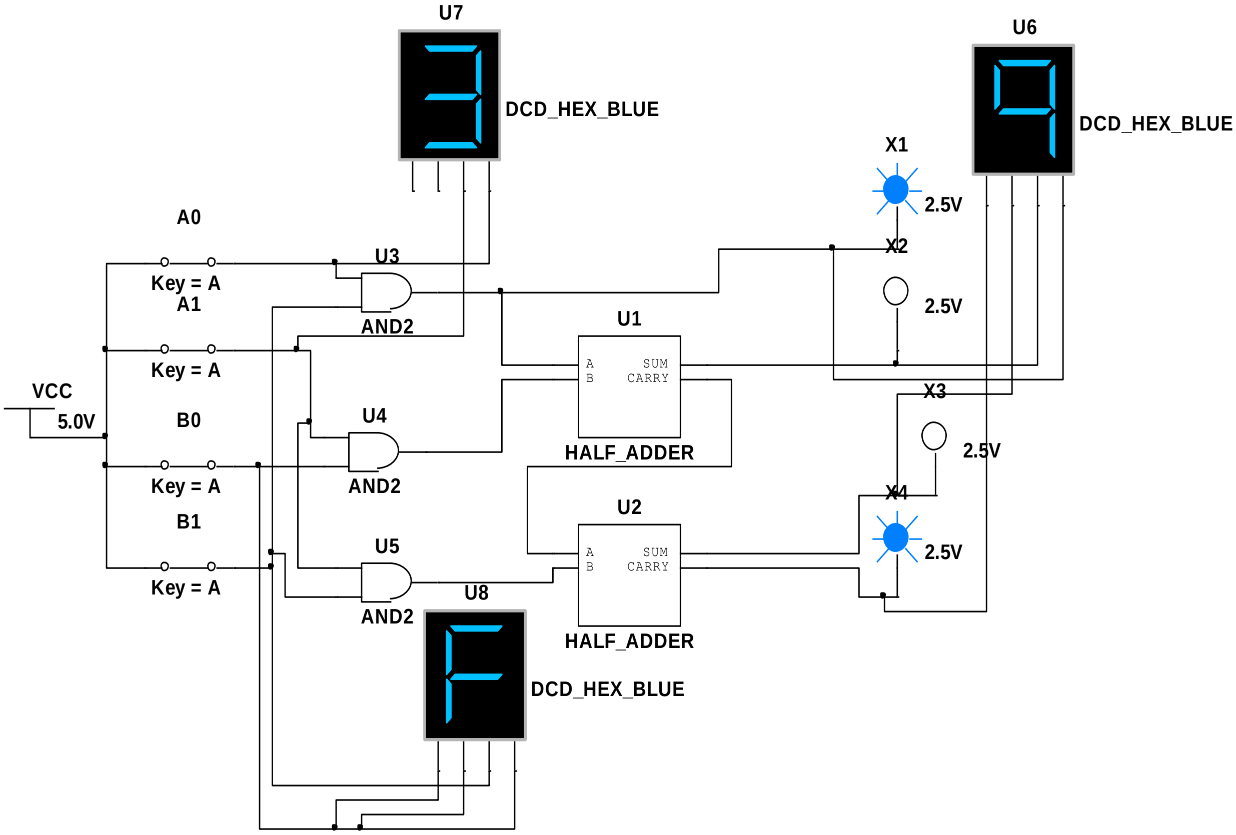




The first partial product is obtained by the AND gate which is nothing but a least significant bit of the multiplication result. Since the second partial product is shifted to the left position, the first partial second term and second partial product first term is added by half adder and produce the sum output along with the carry out.

This carry out is added at the next half adder as an input as shown in figure. Likewise, it produces the multiplication result of two binary numbers by using the simple circuit configuration. The multiplication of the two 2 bit number results a 4-bit binary number.

Simulation diagram:



Result: ---

Staff Signature

VA QUESTIONS AND ANSWERS

1. **Distinguish between min terms and max terms.**

Ans: (i) Each individual term in standard Sum Of Products form is called as minterm whereas each individual term in standard Product Of Sums form is called maxterm.

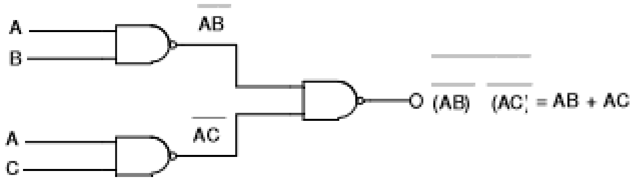
(ii) The unbarred letter represent 1’s and the barred letter represent 0’s in min terms, whereas the unbarred letter represent 0’s and the barred represent 1’s in maxterms.

(iii) If a system has variables A, B, C then the minterms would be in the form ABC, whereas the maxterm would be in the form A+B+C.

(iv) Theminterm designation for three variable expression be $Y= \sum m\ (1, 3, 5, 7)$ Whereas the Maxterm designation for three variable expression be $Y= \prod M\ (0, 1, 3, 4)$

2. What are universal gates. Construct a logic circuit using NAND gates only for the expression x = A .(B + C).

Ans: NAND and NOR Gates are known as Universal gates. The AND, OR, NOT gates can be realized using any of these two gates. The entire logic system can be implemented by using any of these two gates. These gates are easier to realize and consume less power than other gates.



3. What is a decoder?

Ans: A Decoder is a combinational logic circuit that converts Binary words into alphanumeric characters. Thus the inputs to a decoder are the bits 1, 0 and their combinations. The output is the corresponding decimal number. It converts binary information from n input lines to a maximum of 2 n unique output lines. If the n-bit decoded information has unused or don't-care combinations, the decoder output will have less than 2 n outputs.

4. What is an encoder?

Ans: An Encoder is a combinational logic circuit which converts Alphanumeric characters into Binary codes. It has 2n (or less) input lines and n output lines. An Encoder may be Decimal to Binary, Hexadecimal to Binary, Octal to BCD etc.

5. What is a flip-flop? What is the difference between a latch and a flip-flop? List out the application of flip-flop.

Ans: Flip-Flop: A flip-flop is a basic memory element used to store one bit of information. Both Flip-flops and latches are bistable logic circuits and can reside in any of the two stable states due to a feedback arrangement. The main difference between them is in the method used for changing the state.

Applications of Flop-Flops: (1) Bounce elimination switch (2) Parallel Data Storage in Registers (3) Transfer of Data from one bit to another. (4) Counters (5) Frequency Division

6. What is a demultiplexer? Discuss the differences between a demultiplexer and a decoder

Ans: Demultiplexer: It is a logic circuit that accepts one data input and distributes it over several outputs. A demultiplexer has one data input, m select lines, and n output lines, whereas a decoder does not have the data input but the select lines are used as input lines.

7. What is a shift register? Can a shift register be used as a counter?

Ans: Shift Register: A register in which data gets shifted towards left or right when clock pulses are applied is known as a Shift Register. A shift register can be used as a counter. If the output of a shift register is fed back to serial input, then the shift register can be used as a Ring Counter.

8. What are synchronous counters?

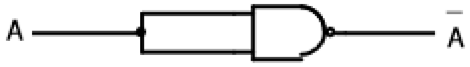
Synchronous Counters: The term synchronous means that all flip-flops are clocked simultaneously. The clock pulses drive the clock input of all the flip-flops together so that there is no propagation delay.

9. What is a universal gate? Give examples. Realize the basic gates with any one universal gate.

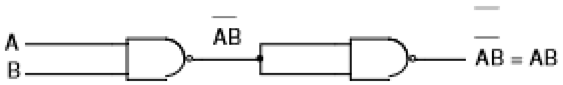
Ans: NAND and NOR are known as Universal gates The AND, OR, NOT gates can be realized using any of these two gates. The entire logic system can be implemented by using any of these two gates. These gates are easier to realize and consume less power than other gates.

Realizations of NOT, AND and OR gates using NAND gates

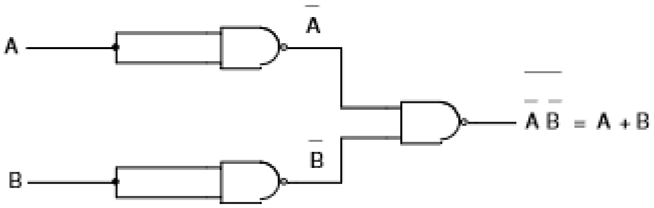
NOT GATE:Figshows the realization of Inverter (NOT) gate using NAND gate. Both the inputs to the NAND gates are tied together so that the gate works as an inverter (NOT) gate.



AND GATE: Fig. shows the realization of AND gate using two NAND gates. It has combination of two NAND gates gives AND operation. The first NAND gate has two inputs A and B. The two inputs to the second NAND gate are tied together and the output AB of the first gate is fed to this common terminal. The output is AB thus giving AND operation.



OR GATE: Fig. shows the realization of OR gate using NAND gates. The two inputs from each of the first two NAND gates are tied together and fed by A and B as shown in the figure. The outputs are A and B. They are fed to as inputs to third NAND gate. The final output is A OR B thus giving OR operation.



10. Define a register

Ans: A register consists of a group of flip-flops and gates that effect their transition. The flip flops hold the binary information and the gates control when and how new information is transformed into the register.

11. The simplification of the Boolean expression (ABC) + (ABC)

Ans: The Boolean expression is (*ABC*)+(*ABC*) is equivalent to 1

(*ABC*)+ (*ABC*) **=** *A* + *B* +*C* + *A* + *B* +*C* = A + *B* + C + *A* + B + *C*

= (A+ *A) (*B+ *B)* (C+*C)* = 1X1X1 = 1

12. The number of control lines for an 8 – to – 1 multiplexer

Ans: The number of control lines for an 8 to 1 Multiplexer is 3

13. What is the binary equivalent of the decimal number 368

Ans: The Binary equivalent of the Decimal number 368 is 101110000

14. How many Flip-Flops are required for mod–16 counter?

Ans: The number of flip-flops is required for Mod-16 Counter is 4.

15. The Gray code for decimal number 6 is equivalent to

Ans: The Gray code for decimal number 6 is equivalent to 0101

16. The Boolean expression A'.B+ A.B'+ A.B is equivalent to

Ans: A + B

17. The digital logic family which has minimum power dissipation

Ans: The digital logic family which has minimum power dissipation is CMOS.

18. The output of a logic gate is 1 when all its inputs are at logic 0. the gate is either

Ans: The output of a logic gate is 1 when all inputs are at logic 0. The gate is either a NOR or anEX-NOR.

19. A ring counter consisting of five Flip-Flops will have

Ans: A ring counter consisting of Five Flip-Flops will have 5 states.

20. The 2's complement of the number 1101101 is

Ans: The 2's complement of the number 1101101 is 0010011

21. When simplified with Boolean Algebra (x + y) (x + z) simplifies to

Ans: When simplified with Boolean Algebra (x + y)(x + z) simplifies to x + yz

[(x + y) (x + z)] = xx + xz + xy + yz = x + xz + xy + yz (Qxx = x)

= x(1+z) + xy + yz = x + xy + yz {Q(1+z) = 1}

= x(1 + y) + yz = x + yz {Q(1+y) = 1}]

22. The gates required to build a half adder are

Ans: The gates required to build a half adder are EX-OR gate and AND gate.

23. The code where all successive numbers differ from their preceding number by single bit

Ans: The code where all successive numbers differ from their preceding number by single bit is Gray Code.

24. If the input to T-flipflop is 100 Hz signal, the final output of the three T-flipflops in cascade

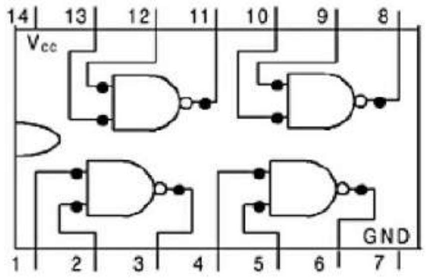
Ans: If the input to T-flip-flop is 100 Hz signal, the final output of the three Tflip-
flops in cascade is 12.5 Hz

{The final output of the three T-flip-flops in cascade is

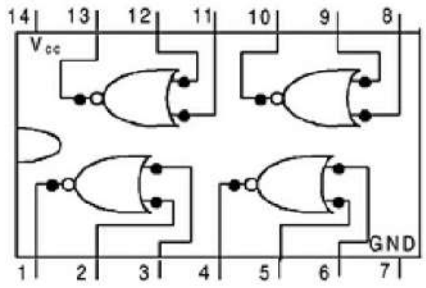
(T) = Frequency= 100=12.5 HZ

APPENDIX

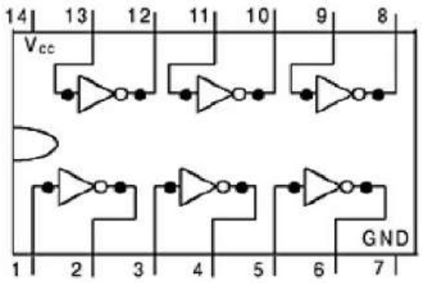
PIN CONFIGURATION OF ICS



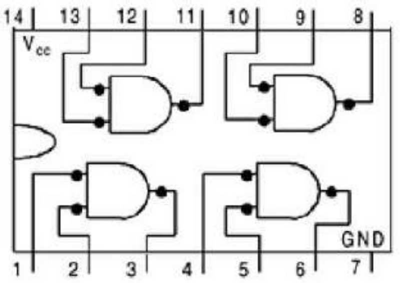
IC 7400/7403*/7426*/7437/7438*/74132



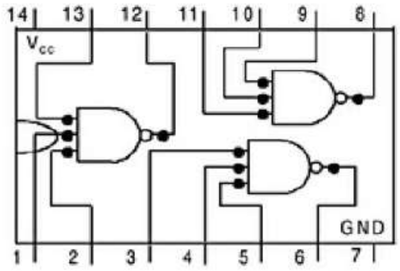
IC 7402/7428/7433*/74128



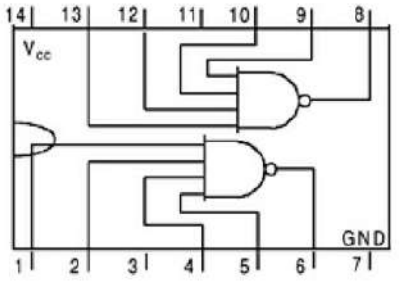
IC 7404/7405*/7406*/7416*



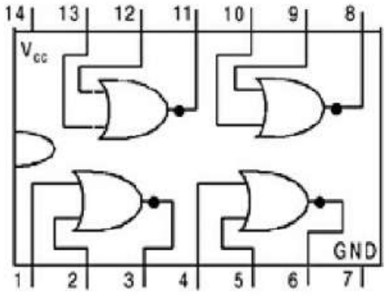
IC 7408/7409*



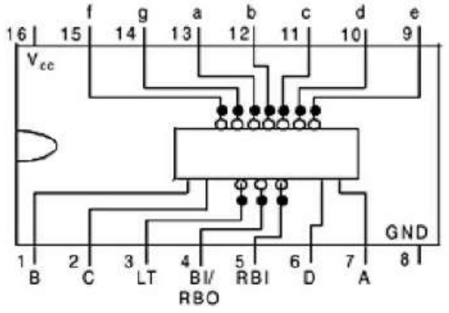
IC 7410/7412*



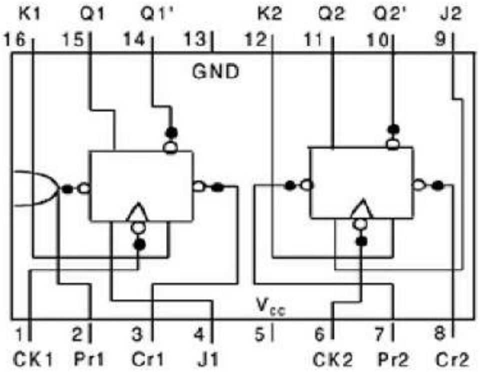
IC 7420/7422*/7440



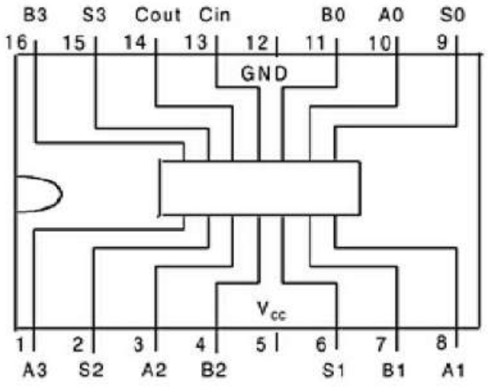
IC 7432



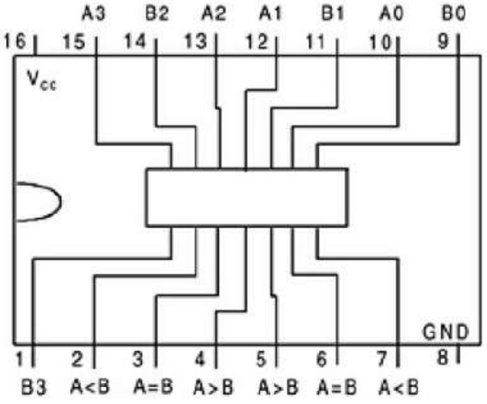
IC 7447/7446*/74246*/74247*



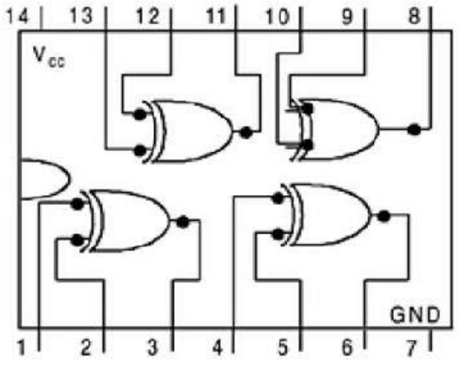
IC 7476/74106



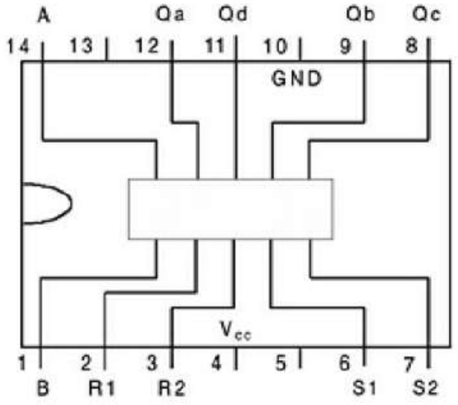
IC 7483



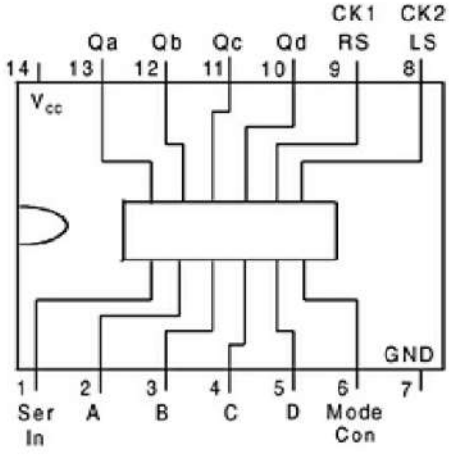
IC 7485



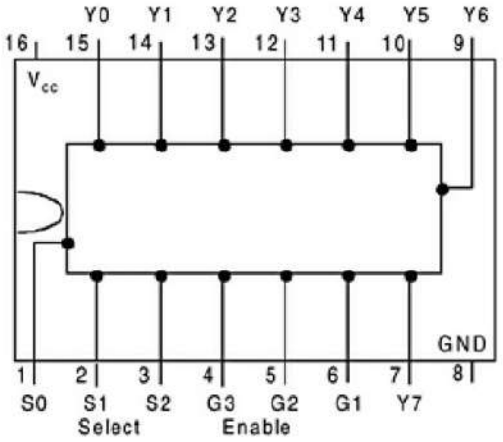
IC 7486



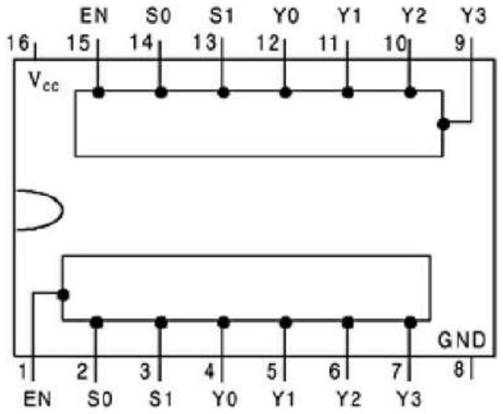
IC 7490



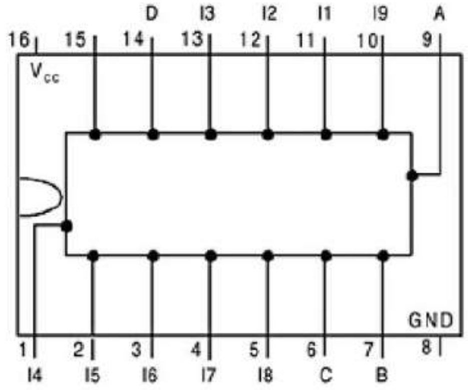
IC 7495



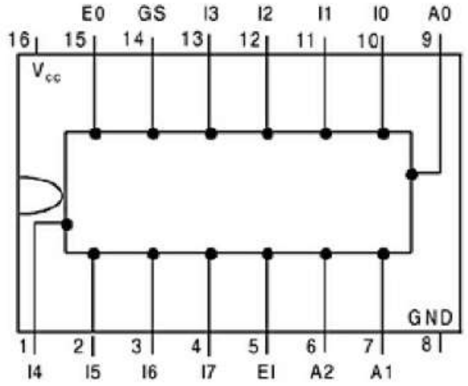
IC 74138



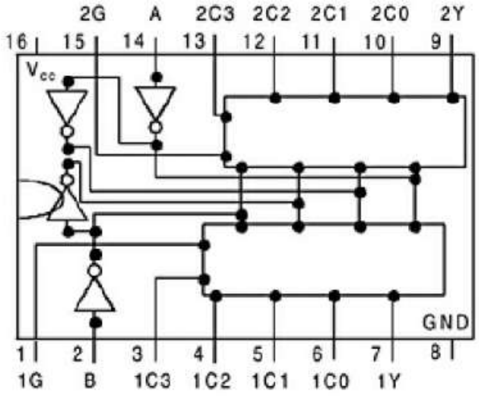
IC 74139



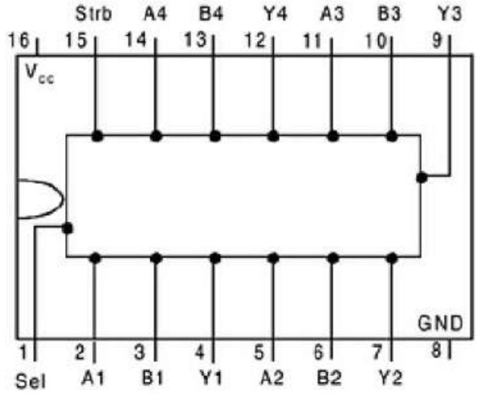
IC 74147



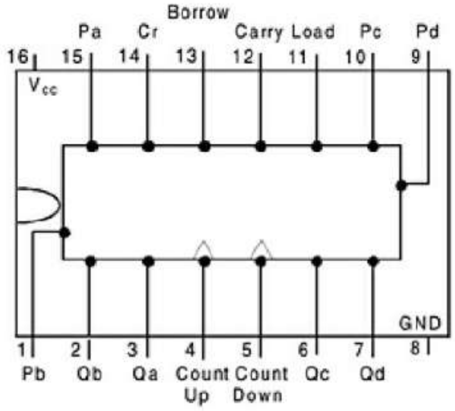
IC 74148



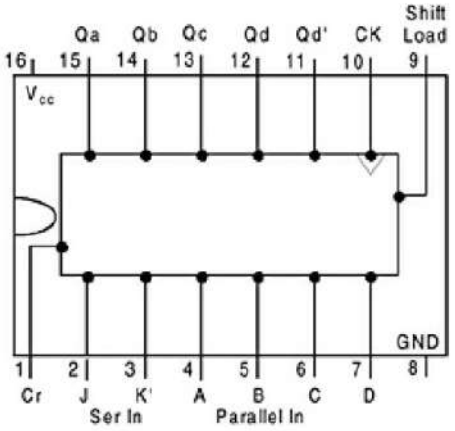
IC 74153



IC 74158/74157



IC 74192 / IC 74193



IC 74195