Savitribai Phule Pune University

F. Y. B. C. A. (Science) Semester-II

Lab Course

Computer Organization Workbook

Name: _____

College Name: _____

Roll No. : _____ Division: _____

Academic Year: _____

F. Y. B. C. A. (Science)

Lab Course

Computer Organization Workbook

Editors:

Ms. Punam P. Warke
Dr. D. Y. Patil ACS College, Pimpri, Pune

Ms. Chaitali Tikhe
Dr. D. Y. Patil ACS College, Pimpri, Pune

Reviewed By:

Mrs. Jayshree S. Sonawane
Dr. D. Y. Patil ACS College, Pimpri,Pune

Mr. Deepak S. Kumbhar
Modern College, Ganeshkhind, Pune-16

Introduction

1. About the Workbook:

This workbook is intended to be used by FYBCA (Science) students for the Computer Organization Laboratory Assignments in Semester–II. This workbook is designed by considering all the practical concepts / topics mentioned in syllabus.

2. The Objectives of this Workbook are:

- Defining the scope of the course.
- To bring the uniformity in the practical conduction and implementation in all colleges affiliated to SPPU.
- To have continuous assessment of the course and students.
- Providing ready reference for the students during practical implementation.
- Provide more options to students so that they can have good practice before facing the examination.
- Catering to the demand of slow and fast learners and accordingly providing the practice assignments to them.

3. How to use this Workbook:

Computer Organization Laboratory workbook is divided into eleven assignments. The assignments comprise of activities to be carried out while doing each practical. The students have to study the respective practical kit, make connections, insert appropriate records or observations and then perform the activities specified in each of the assignments.

4. Instructions to the students:

Please read the following instructions carefully and follow them.

- Students are expected to carry this workbook every time they come to the lab for practical.
- Students should prepare for the assignment by reading the relevant material which is mentioned in ready reference.
- Instructor will specify which problems to solve in the lab during the allotted slot and student should complete them and get verified by the instructor. However, student

should spend additional hours in Lab and at home to cover all workbook assignments if needed.

- Students will be assessed for each assignment on a scale from 0 to 5

Not done	0
Incomplete	1
Late Complete	2
Needs improvement	3
Complete	4
Well Done	5

5. Instruction to the Instructors:

- Make sure that students should follow above instructions.
- Explain the practical assignment and related concepts using white board if required or by demonstrating on the circuit board.
- Evaluate each assignment carried out by a student on a scale of 5 as specified above by ticking appropriate box.
- The value should also be entered on assignment completion page of the respective Lab course.

Savitribai Phule Pune University

Section-II

F. Y. B. C. A. (Science)

SEMESTER II

BCA - 125

Lab Course

Computer Organization Laboratory

Table of Contents

CERTIFICATE

This is to certify that

Mr./Ms. _____

Has successfully completed the Computer Organization Laboratory course work and has scored _____ Marks out of 15.

Instructor **H.O.D. / Coordinator**

Internal Examiner **External Examiner**

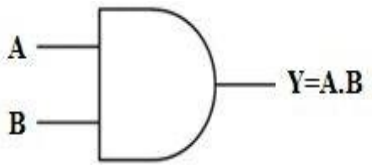
Assignment 1

Aim: **To Study & Verify the Truth Tables of Logic Gates.**

Requirements: Circuit Board/ Digital Trainer kit, Logic Gates ICs (IC 7400, IC7402, IC7404, IC7408, IC7432, and IC7486) Digital multi-meter and Connecting Wires.

Logic Gates:

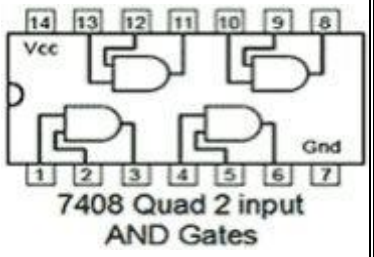
1. AND Gate:



Symbol

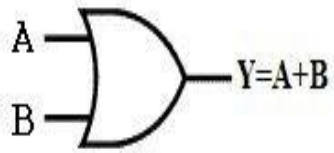
Inputs		Output
A	B	Y=A.B
0	0	0
0	1	0
1	0	0
1	1	1

Truth Table



Pin Diagram for IC 7408

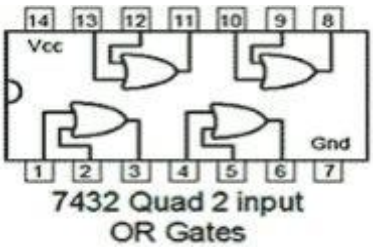
2. OR Gate:



Symbol

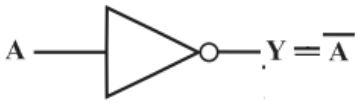
Inputs		Output
A	B	Y=A+B
0	0	0
0	1	1
1	0	1
1	1	1

Truth Table



Pin Diagram for IC 7432

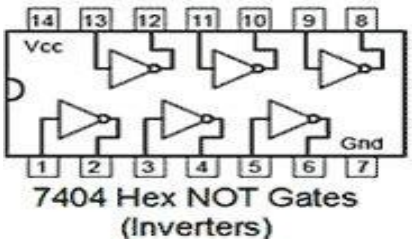
3. NOT Gate:



Symbol

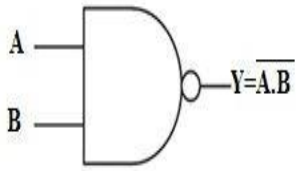
A	Y = A'
0	1
1	0

Truth Table



Pin Diagram for IC 7404

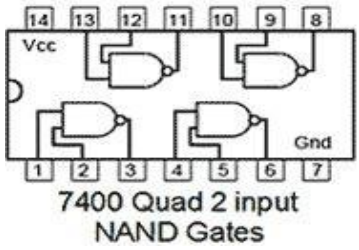
4. NAND Gate:



Symbol

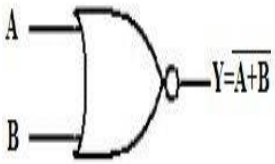
Inputs		Output
A	B	$Y=\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

Truth Table



Pin Diagram for IC 7400

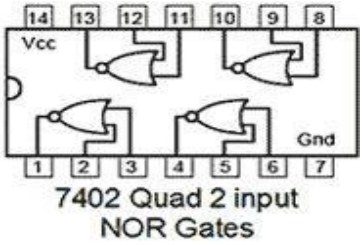
5. NOR Gate:



Symbol

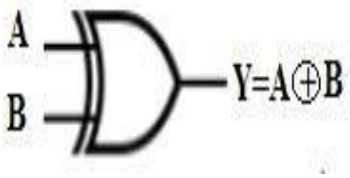
Inputs		Output
A	B	$Y=\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

Truth Table



Pin Diagram for IC 7402

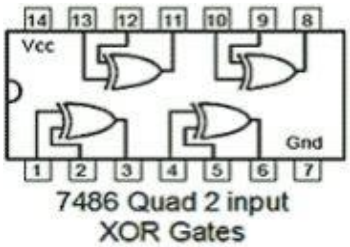
6. EX-OR Gate:



Symbol

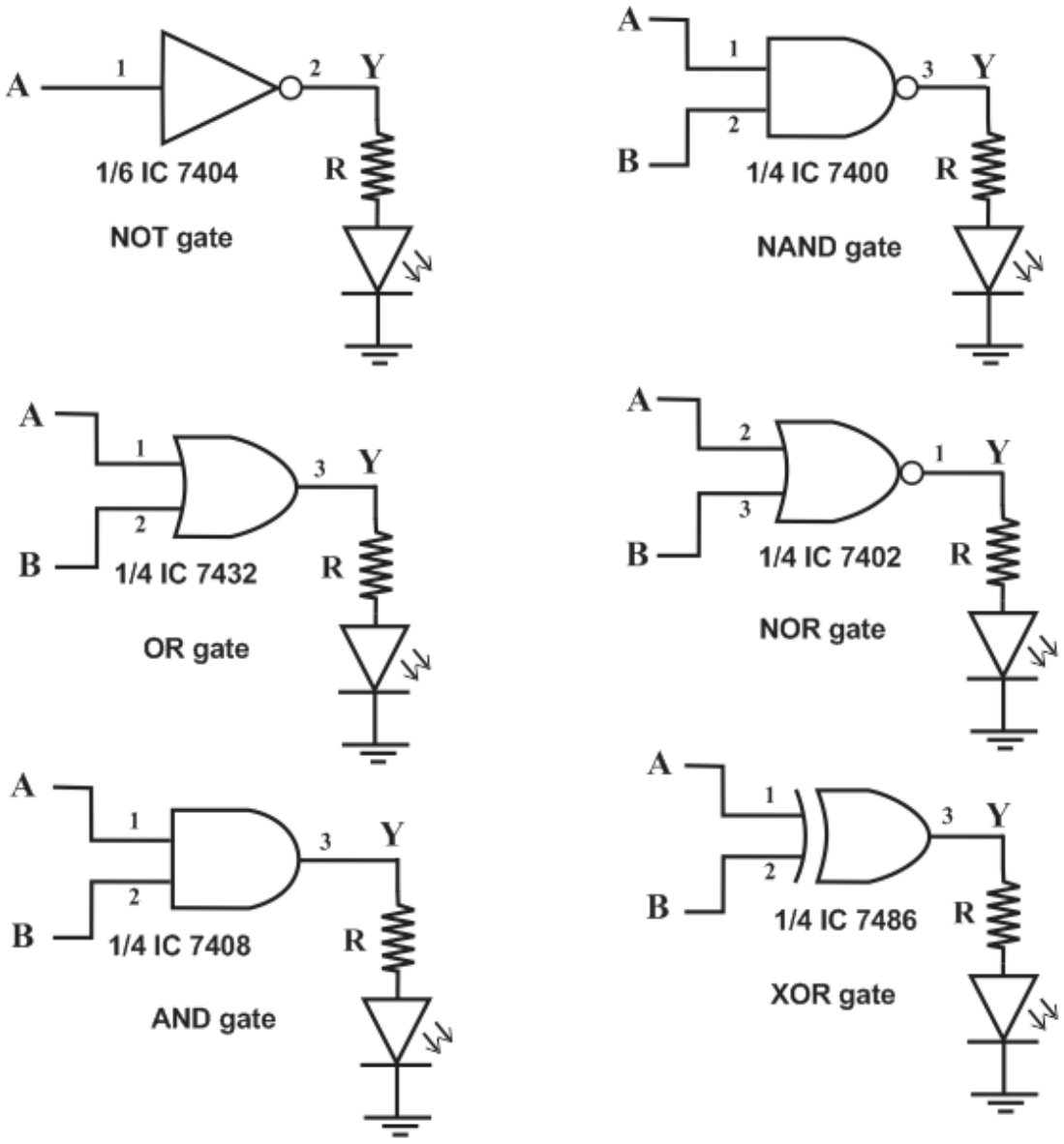
Inputs		Output
A	B	$Y=A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Truth Table



Pin Diagram for IC 7486

Circuit Diagram:



Procedure:

Setup the circuits shown in figure to analyze the operation of the various logic Gates. For each gate: Use +5V for logic 1 and 0V (Ground) for logic 0, as input conditions. Connect the output pin to LED. Vary the input conditions of each gate and observe the output. For the output, if LED is ON, consider it as logic 1 and if it is OFF, consider it as logic 0. Do this for all possible Combinations of inputs Repeat the same procedure for all other Gates. Verify the truth Table of the each Logic Gate.

Precautions:

All the connections should be made properly.

Observation Table:

Adopting Positive Logic

Input condition:

Voltage Level (V.L.) =5V → Logic Level (L.L) =1,

Voltage Level (V.L.) =0V → Logic Level (L.L.) =0

Output condition:

LED ON→ Logic Level (L.L.) =1,

LED OFF→ Logic Level (L.L.) =0

1. **NOT gate**

Input A		**Output Y=A**	
L.L.	V.L.	LED	L.L.
0			
1			

Conclusion: Output is the COMPLEMENT of input.

2. **OR gate**

Input A		**Input B**		**Output Y=A+B**	
L.L.	V.L.	L.L.	V.L.	LED	L.L.
0		0			
0		1			
1		0			
1		1			

Conclusion: Any HIGH input drives the output HIGH

3. **AND gate**

Input A		**Input B**		**Output Y=A . B**	
L.L.	V.L.	L.L.	V.L.	LED	L.L.
0		0			
0		1			
1		0			
1		1			

Conclusion: Any LOW input drives the output LOW.

4. NAND gate

Input A		**Input B**		**Output Y=A . B**	
L.L.	V.L.	L.L.	V.L.	LED	L.L.
0		0			
0		1			
1		0			
1		1			

Conclusion: Any LOW input drives the output HIGH

5. NOR gate

Input A		Input B		Output Y=A+B	
L.L.	V.L.	L.L.	V.L.	LED	L.L.
0		0			
0		1			
1		0			
1		1			

Conclusion: Any HIGH input drives the output LOW.

6. XOR gate

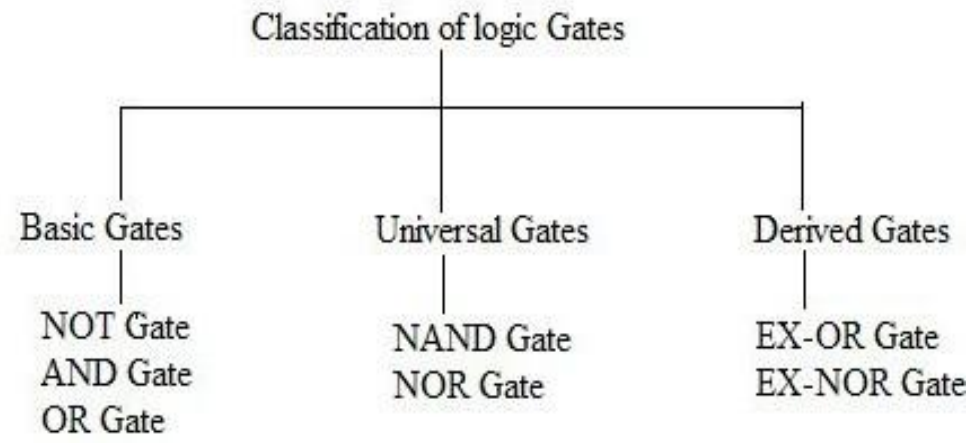
Input A		Input B		Output Y=A+B	
L.L.	V.L.	L.L.	V.L.	LED	L.L.
0		0			
0		1			
1		0			
1		1			

Conclusion: Dissimilar inputs drive the output HIGH.

Theory:

Logic gates are electronic circuits that implement basic logic operations. Each gate has a symbol associated with it. The logic gates accept the input and give outputs only in two states 0 and 1.

Classification of logic gates:



Basic Gates:

There are 3 basic logic gates AND, OR &NOT. They are called basic logic gates because they cannot be derived by taking combinations of one another. Each logic function is independent of each other.

Universal Gates:

The NAND and NOR Gates are called as "Universal Gates" because it is possible to implement AND, OR & NOT Gates using only NAND and NOR Gates. Any combinational circuit can be design using these gates.

Derived Gates:

A derived logic gate is obtained by considering combinations of basic logic gates AND, OR & NOT. For ex.: NAND Gate is derived by using OR gate whose output is given to a NOT gate.

Result:

Conclusion:

Questions:
What do you mean by Logic gate? List Universal Logic gates.

State Propagation delay/Power dissipation/Fan Out for a Logic Gate.

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor

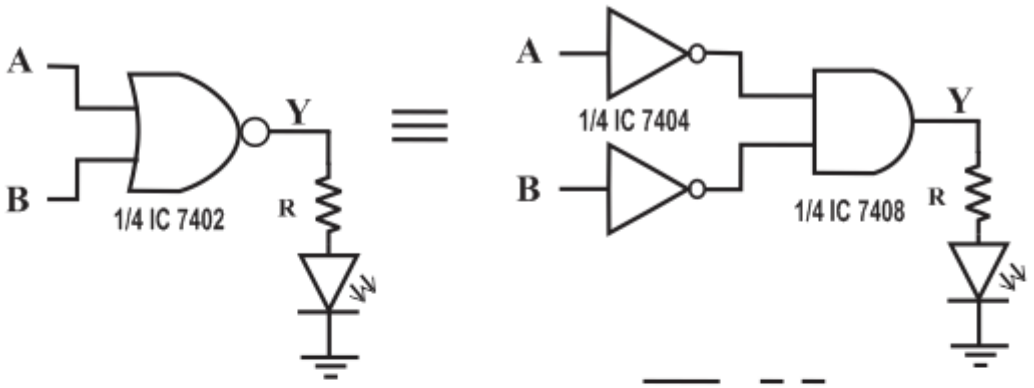
Assignment 2

Aim: Study of De-Morgan's Theorems

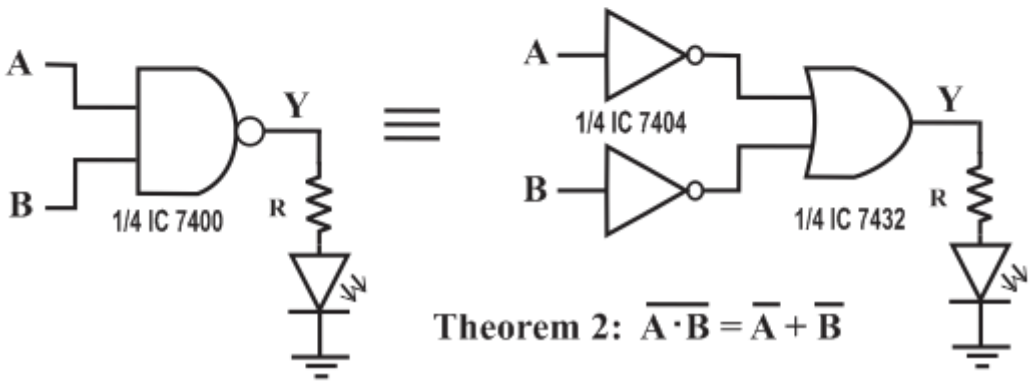
Requirements: Circuit Board/ Digital Trainer kit and connecting wires

Circuit

Diagram:



Theorem 1: $\overline{A+B} = \overline{A} \cdot \overline{B}$



Theorem 2: $\overline{A \cdot B} = \overline{A} + \overline{B}$

Proof:

1) A + B = A * B

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

2) A * B = A + B

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

Procedure:-

1) Take a circuit board
2) Do the connections as per the circuit diagram.
3) Give the proper inputs.
4) To apply high logic to the inputs, connect the inputs to +Vcc & apply low logic connect them to GND.
5) Note down the output.
6) LED glows, when the output is high, otherwise it is low.
7) Repeat the same procedure for all other circuit diagram & verify the truth table.

Observation Table:

Theorem 1: A + B = A * B

A	B	A+B		A * B	
L.L	L.L.	LED	L.L.	LED	L.L.
0	0				
0	0				
1	1				
1	1				

Theorem 2: A * B = A + B

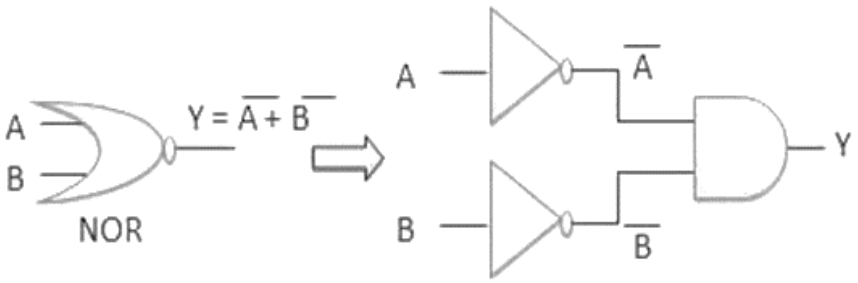
A	B	A * B		A + B	
L.L	L.L.	LED	L.L.	LED	L.L.
0	0				
0	0				
1	1				
1	1				

Theory:

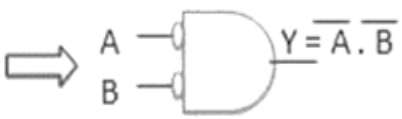
Theorem 1:

$\overline{A+B} = \overline{A}.\overline{B}$

NOR = Bubbled AND



NOR ☰ Bubbled AND



Bubbled AND

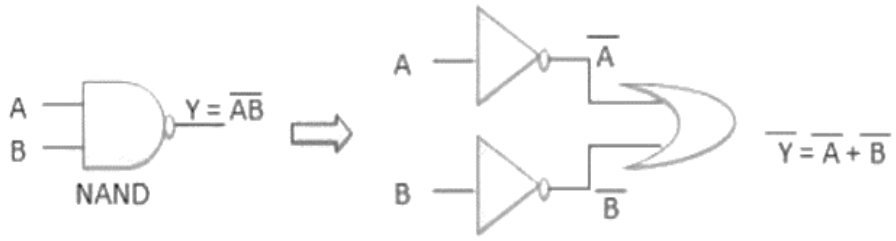
It states that the sum of the product of two variables is equal to product of the

compliment of each variable. A NOR Gate is same as a bubbled OR Gate.

Theorem 2:

$$\overline{A.B} = \overline{A} + \overline{B}$$

NAND = Bubbled OR



NAND ≡ Bubbled OR

Bubbled OR

It states that the compliment of the product of two variables is equal to sum of the compliment of each variable. A NAND Gate is same as a bubbled OR Gate.

Result:

Conclusion:

Questions:

1. **State and Prove De-Morgan's Theorems.**

2. **Write applications of De-Morgan's Theorem.**

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor

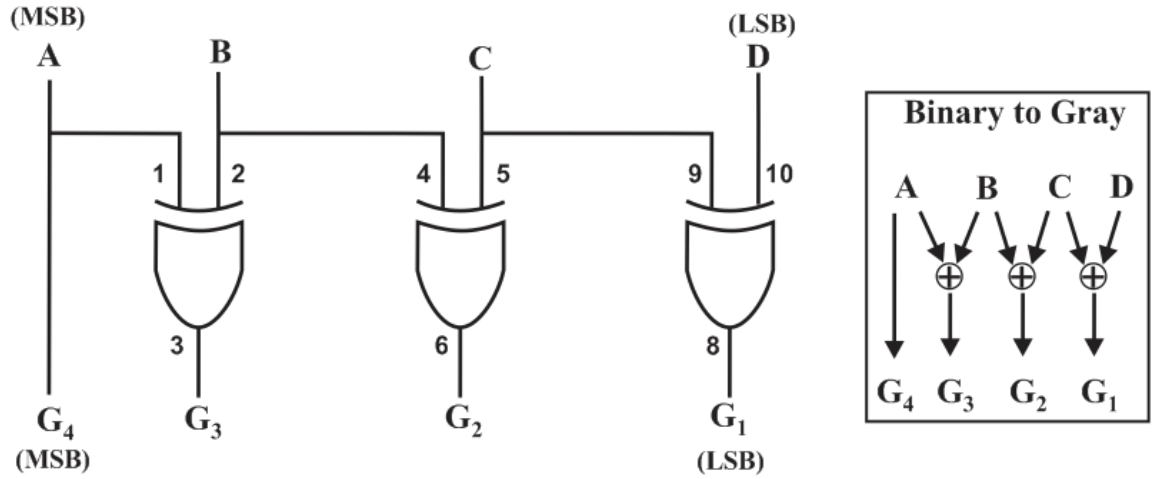
Assignment 3

Aim: **Implementation of Code Converters (binary to gray and gray to binary) using K-Map.**

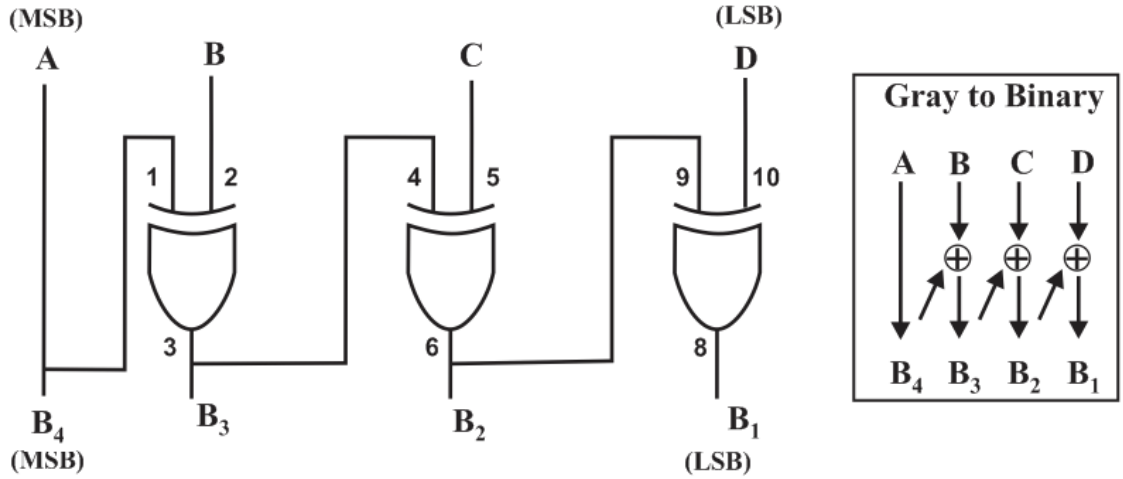
Requirements: Circuit Board/ Digital Trainer kit, logic gate IC's and connecting wires etc.

Circuit diagram

A) Binary to Gray Conversion



B) Gray to Binary Conversion



Procedure:

1. Draw proper K-map for binary to gray and vice versa and draw the logic diagram from the minimized output equation.
2. Make the circuit connections according to logic diagram shown in Fig a. & b. on

circuit/bread board using wires.

3. In the case of **binary to gray conversion**, apply Logic '0' or Logic '1' level input to A, B, C and D using toggle switches and observe the LED condition at the outputs G4, G3, G2, G1 for all the 16 combinations of the inputs.
4. In the case of **gray to binary conversion**, apply Logic '0' or Logic '1' level input to A, B, C and D using toggle switches and observe the LED condition at the outputs B4, B3, B2, and B1 for all the 16 combinations of inputs.

Precautions:

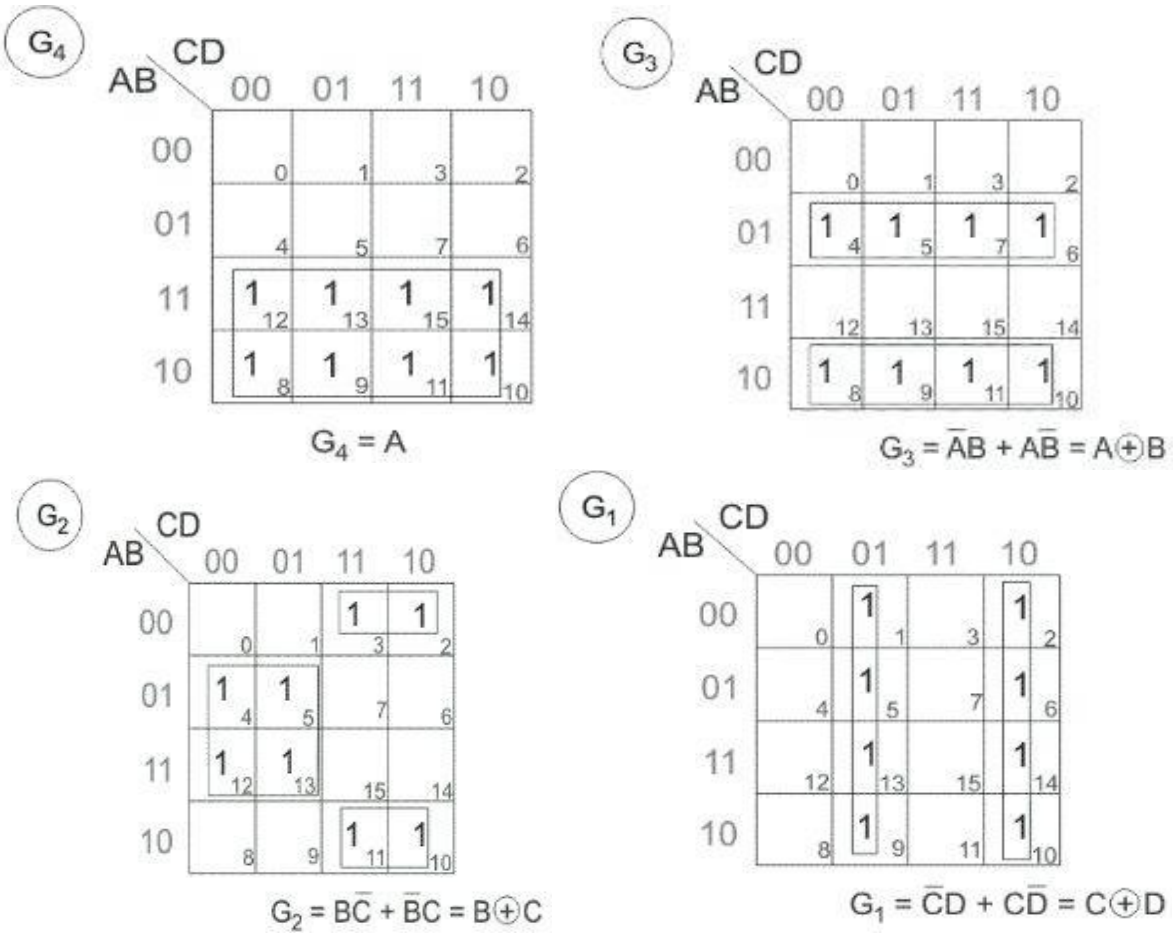
All the connections should be made properly.

Observations:

A) Binary to Gray Conversion

Decimal Numbers	**Input** 4- bit Binary Code				**Output** 4- bit Code				**Observe output** 4- bit Gray Code			
	A	B	C	D	G4	G3	G2	G1	G4	G3	G2	G1
0	0	0	0	0	0	0	0	0				
1	0	0	0	1	0	0	0	1				
2	0	0	1	0	0	0	1	1				
3	0	0	1	1	0	0	1	0				
4	0	1	0	0	0	1	1	0				
5	0	1	0	1	0	1	1	1				
6	0	1	1	0	0	1	0	1				
7	0	1	1	1	0	1	0	0				
8	1	0	0	0	1	1	0	0				
9	1	0	0	1	1	1	0	1				
10	1	0	1	0	1	1	1	1				
11	1	0	1	1	1	1	1	0				
12	1	1	0	0	1	0	1	0				
13	1	1	0	1	1	0	1	1				
14	1	1	1	0	1	0	0	1				
15	1	1	1	1	1	0	0	0				

K-Maps For Out Put Expressions

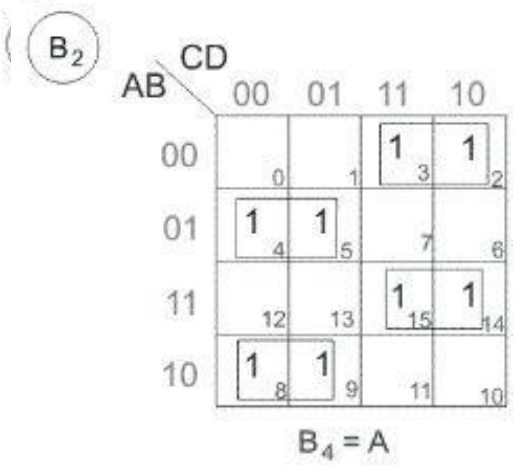


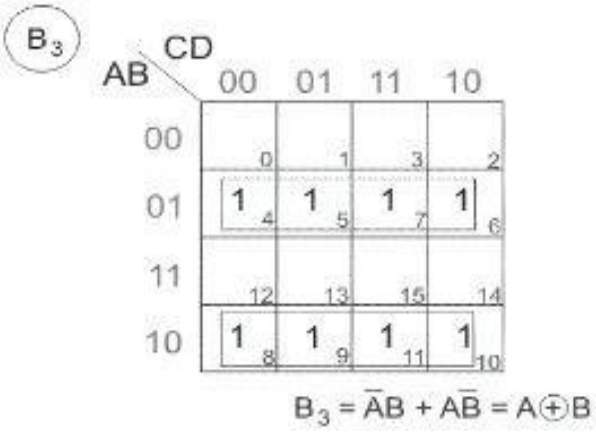
B) Gray to Binary Conversion

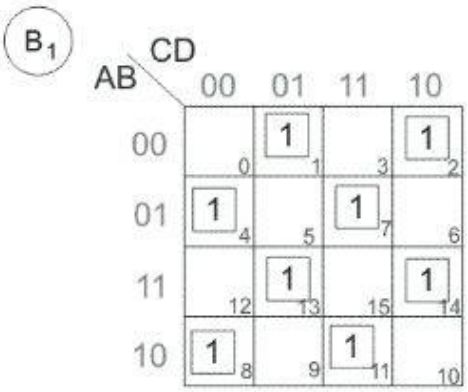
Decimal Numbers	**Input** 4- bit Gray Code				**Output** 4- bit Binary Code				**Observe output** 4- bit Binary Code			
	A	B	C	D	B3	B2	B1	B0	B3	B2	B1	B0
0	0	0	0	0	0	0	0	0				
1	0	0	0	1	0	0	0	1				
2	0	0	1	1	0	0	1	0				
3	0	0	1	0	0	0	1	1				
4	0	1	1	0	0	1	0	0				
5	0	1	1	1	0	1	0	1				
6	0	1	0	1	0	1	1	0				
7	0	1	0	0	0	1	1	1				
8	1	1	0	0	1	0	0	0				
9	1	1	0	1	1	0	0	1				
10	1	1	1	1	1	0	1	0				

11	1	1	1	0	1	0	1	1				
12	1	0	1	0	1	1	0	0				
13	1	0	1	1	1	1	0	1				
14	1	0	0	1	1	1	1	0				
15	1	0	0	0	1	1	1	1				

K-Maps For Out Put Expressions







Theory:

Code is a way to represent data. the codes can be classified as weighted code and non-weighted code.

WEIGHTED CODES: it is a code in which a weight is assigned to each symbol position of the code. e.g. hexadecimal code, binary code etc

NON WEIGHTED CODES: it is a code in which no fixed weight is assign to each symbol position of the code. Gray code is non weighted code it is a code in terms of 1 and 0 and every new code differs from the previous code only in one bit position. Gray code is not used for performing arithmetic operations. Instead it is used in applications like encoding the position of shaft or wheels. It is also used extensively to represent a cell in K-map.

$$B_2 = \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + ABC$$
$$= A(\overline{B}\,\overline{C} + BC) + \overline{A}(B\overline{C} + \overline{B}C)$$
$$= A\overline{(B\overline{C} + \overline{B}C)} + \overline{A}(B\overline{C} + \overline{B}C)$$
$$= A\overline{(B \oplus C)} + \overline{A}\overline{(B \oplus C)} = A \oplus B \oplus C$$

Result:

Conclusion:

$$B_1 = \overline{A}\,\overline{B}\,\overline{C}\,D + \overline{A}\,\overline{B}C\,\overline{D} + \overline{A}B\overline{C}\,\overline{D} + \overline{A}BCD + AB\overline{C}\,D + ABC\overline{D} + A\overline{B}\,\overline{C}\,\overline{D}$$
$$+ A\overline{B}CD = A \oplus B \oplus C \oplus D$$

Questions:

1. **Give difference between weighted and non-weighted number systems.**

2. **Draw symbol of Ex-OR gate with Truth table.**

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

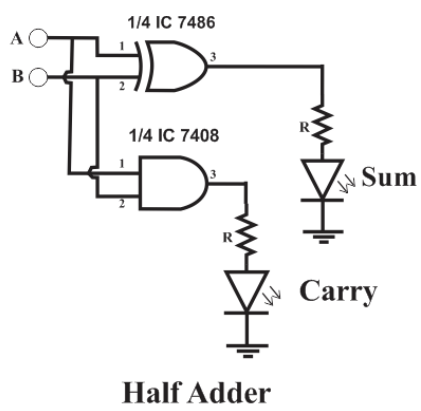
Signature of Instructor

Assignment 4

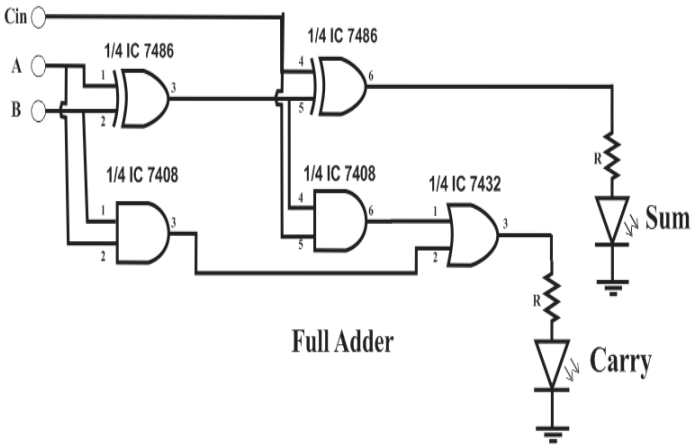
Aim: To study and verify the truth tables of half adder and full adder circuit.

Requirements: Circuit Board/ Digital Trainer kit, Digital multi-meter and connecting wires.

Circuit Diagram:



Half Adder



Full Adder

Truth table for Half adder

INPUTS		OUTPUTS	
A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Full Adder Truth Table

INPUT			OUTPUT	
C_P	A	B	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

Procedure:

1) Make connections as shown in circuit diagram
2) Connect the sum and carry outputs to the anodes of LEDs while their cathodes are grounded
3) Apply various combinations of input according to truth table and observe the corresponding sum and carry output.
4) Note down the sum and carry output readings of half adder and full adder for different combinations of inputs and verify their truth tables.

Observation Table:

a. Half Adder

Input A		Input B		Sum		Carry	
L.L.	**V.L.**	**L.L.**	**V.L.**	**LED**	**L.L.**	**LED**	**L.L.**
0		0					
0		1					
1		0					
1		1					

b. Full Adder

Input Cp		Input A		Input B		Sum		Carry	
L.L.	**V.L.**	**L.L.**	**V.L.**	**L.L.**	**V.L.**	**LED**	**L.L.**	**LED**	**L.L.**
0		0		0					
0		0		1					
0		1		0					
0		1		1					
1		0		0					
1		0		1					
1		1		0					
1		1		1					

Theory:

An adder is a digital circuit that performs addition of numbers. The half adder adds two binary digits and produces two outputs as sum and carry; XOR is applied to both inputs to produce sum and AND gate is applied to both inputs to produce carry. The full adder adds 3 one bit numbers, and produces 2-bit output, and these can be referred to as output carry and sum. With the help of half adder, we can design circuits that are capable of performing simple addition with the help of logic gates.

Let us first take a look at the addition of single bits.

0+0 = 0

0+1 = 1

1+0 = 1

1+1 = 10

These are the least possible single-bit combinations. But the result for 1+1 is 10. Though this problem can be solved with the help of an EXOR Gate, if you do care about the output, the sum result must be re-written as a 2-bit output.

Here the output '1'of '10' becomes the carry-out. 'SUM' is the normal output and 'CARRY' is the carry-out. 1-bit adder can be easily implemented with the help of EXOR Gate for the output 'SUM' and an AND Gate for the carry.

Thus, we can implement a full adder circuit with the help of two half adder circuits. The first will half adder will be used to add A and B to produce a partial Sum. The second half adder logic can be used to add C_{IN} to the Sum produced by the first half adder to get the final S output. If any of the half adder logic produces a carry, there will be an output carry. Thus, COUT will be an OR function of the half-adder Carry outputs.

The main difference between the Full Adder and the Half Adder is that a full adder has three inputs. The same two single bit data inputs A and B as before plus an additional Carry-in (C-in) input to receive the carry from a previous stage

Result:

Conclusion:

Questions:

1. **State the applications of Full Adder circuit.**

2. How many Half and Full Adders will be required for 2-Nibble addition?

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

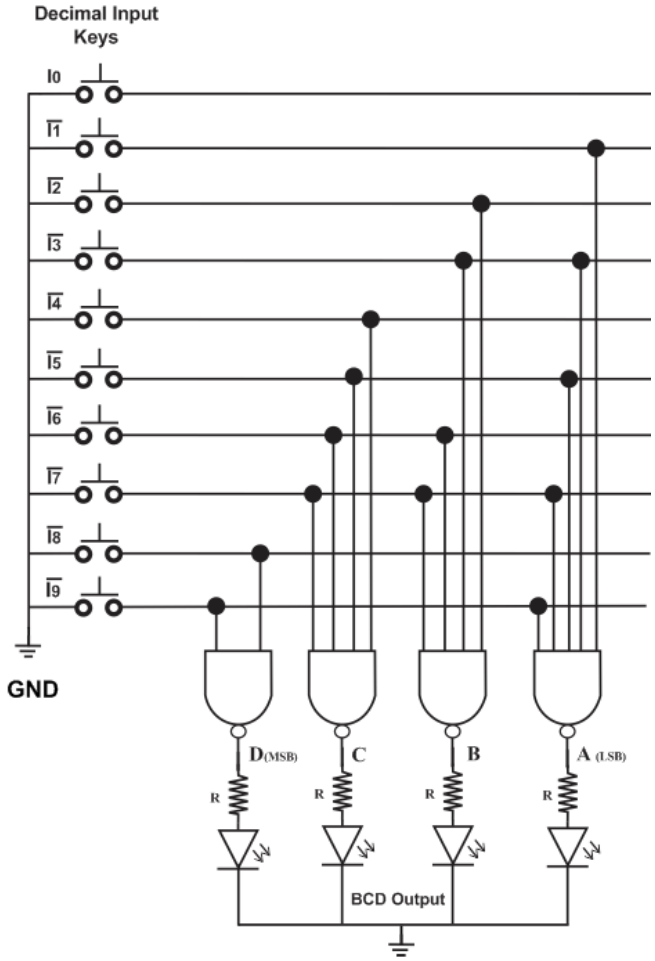
Signature of Instructor

Assignment 5

Aim: Build & test Decimal to BCD Encoder using Logic Gates.

Requirements: Circuit Board/ Digital Trainer kit, DMM, Connecting Wires etc.

Circuit Diagram:



Decimal To BCD Encoder

Boolean Equations:

For Output,

$$A = \overline{I1*I3*I5*I7*I9}$$
$$A = \overline{I1}+\overline{I3}+\overline{I5}+\overline{I7}+\overline{I9}$$
$$B = \overline{I3*I4*I6*I7}$$
$$B = \overline{I3}+\overline{I4}+\overline{I6}+\overline{I7}$$
$$C = \overline{I4*I5*I6*I7}$$
$$C = \overline{I4}+\overline{I5}+\overline{I6}+\overline{I7}$$
$$D = \overline{I8*I9}$$
$$D = \overline{I8}+\overline{I9}$$

Procedure:

1) Construct the circuits of **Decimal to BCD Encoder with active LOW inputs** as shown in diagram.
2) Active the input line I_0, i.e. apply logic 0 to line no. I_0 & all other input lines are at logic 1.
3) Note down the outputs & complete the Respective truth table.
4) Note down the outputs & complete the Respective truth table.
5) Make your necessary COMMENT.

Observation Table:

Decimal to BCD Encoder with active LOW inputs

Decimal No.	**Active Low Inputs**										**BCD outputs**			
	I0	I1	I2	I3	I4	I5	I6	I7	I8	I9	(MSB) D	C	B	(LSB) A
0	0	1	1	1	1	1	1	1	1	1				
1	1	0	1	1	1	1	1	1	1	1				
2	1	1	0	1	1	1	1	1	1	1				
3	1	1	1	0	1	1	1	1	1	1				
4	1	1	1	1	0	1	1	1	1	1				
5	1	1	1	1	1	0	1	1	1	1				
6	1	1	1	1	1	1	0	1	1	1				
7	1	1	1	1	1	1	1	0	1	1				
8	1	1	1	1	1	1	1	1	0	1				
9	1	1	1	1	1	1	1	1	1	0				

Theory:

Decimal to BCD Encoder:

The Decimal to BCD conversion can be implemented by the truth table for the conversion.

Decimal	BCD			
	D(MSB)	C	B	A(LSB)
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

NAND Gate Truth Table:

2 Input NAND gate		
A	B	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

From the truth table, it is clear that D=1, for the decimal inputs 8, 9. C=1, for the decimal inputs 4,5,6,7. B=1 for the decimal inputs 2,3,6,7. A=1 for the decimal inputs 1, 3,5,7,9.

For active low inputs, we have to use NAND gates and the respective decimal inputs must be connected to those NAND gates to get the BCD output.

In the circuit, all the inputs are active low i.e., when the key is ON the corresponding line will become LOW. When the key is OFF, that corresponding line will become HIGH. When all the keys are ON, the output of all NAND gate will be LOW.

For Example, when the key corresponding to I_1 is OFF (logic 0), then output of NAND gate A will become high and all other NAND gate output will be low i.e. we will get the BCD output 0001 corresponding to decimal 1. When the key corresponding to I_2 is OFF (logic 0), then output of NAND gate B will become high and all other NAND gate output will be low. i.e. we will get the BCD output 0010 corresponding to decimal 2 and so on.

Result:

Conclusion:

Questions:

1. **Define Encoder? Give types of Encoders.**

2. What is Priority Encoder?

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

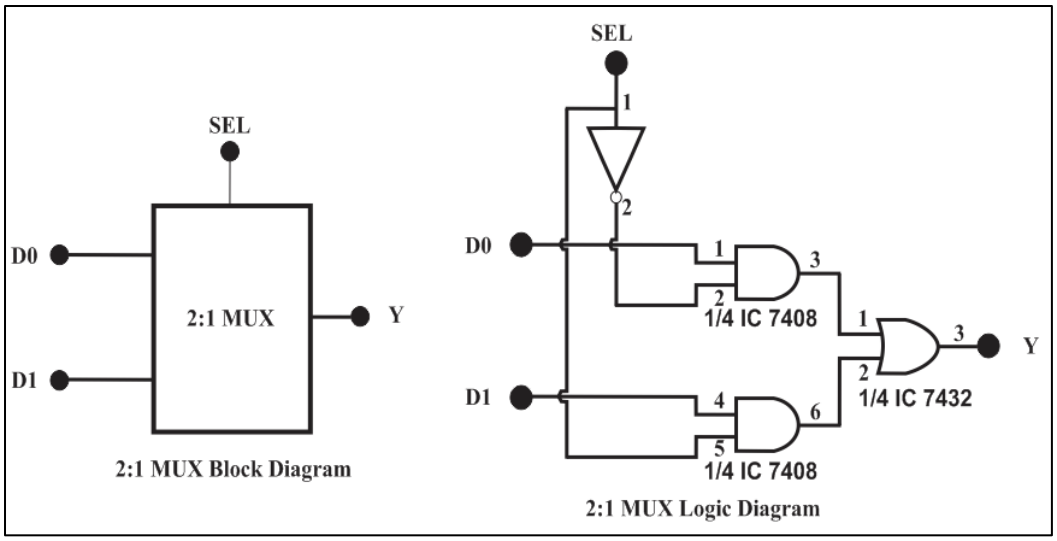
Signature of Instructor

Assignment 6

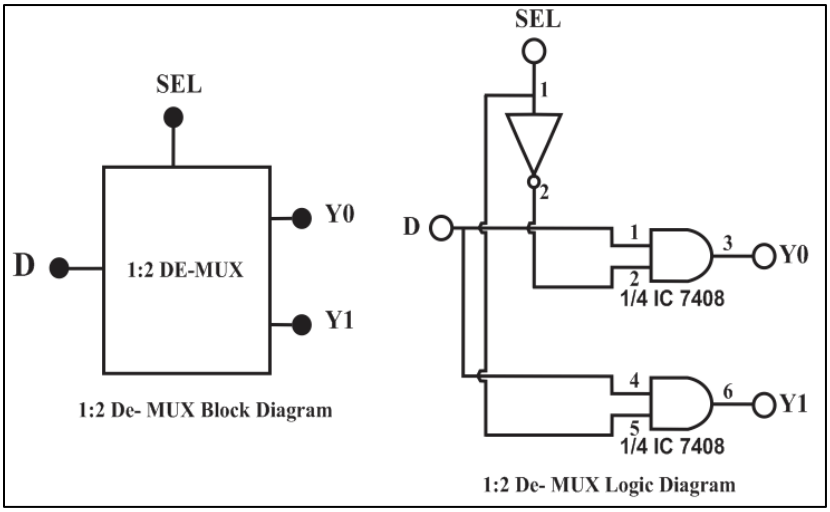
Aim: To study Multiplexer and De-multiplexer.

Requirements: Circuit Board/ Digital Trainer kit and connecting wires.

Circuit Diagrams:



Multiplexer



De-Multiplexer

2:1 MUX Truth Table: -

SEL	Data Inputs		Output	Selected input
	D_0	D_1	Y	
0	1/0	x	1/0	D_0
1	x	1/0	1/0	D_1

1:2 De- MUX Truth Tables: -

SEL	Data Input D	Data Inputs		Selected output
		Y_0	Y_1	
0	0	1/0	x	Y_0
1	0	x	1/0	Y_1

Procedure:

1. Construct the circuit as shown in figure.
2. Make connections as shown in the circuit diagram.
3. Verify the Truth Table and observe the outputs

Observation Table:

a. 2:1 Multiplexer

Input SEL	Input Do	Input D1	Output Y		Selected Input
L.L.	L.L.	L.L.	LED	L.L.	
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

b. 1:2 De-Multiplexer

Input SEL	Input D	Output Yo		Output Y1		Selected Output
L.L.	L.L.	LED	L.L.	LED	L.L.	
0	0					
0	1					
1	0					
1	1					

Theory:

Multiplexer:

Multiplexer means many into one. A multiplexer is a circuit used to select and route any one of the several input signals to a single output. Multiplexer has several data inputs and only one output and number of select lines. One of the inputs is selected depending upon the condition of select lines and transfer to the output.

De-Multiplexer:

De-Multiplexer means one to many. A De-Multiplexer is a logic circuit with one input and many outputs and number of select lines. By applying control signal, the input is transferred to one of the several outputs.

Result:

Conclusion:

Questions:

1. **What is Multiplexer and De-Multiplexer?**

2. Write applications of Multiplexers.

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor

Assignment 7

Aim: Study of flip- flop:

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

Requirements: Circuit Board/ Digital Trainer kit and connecting wires.

Circuit Diagram with Truth table:

1. **SR Flip-flop**

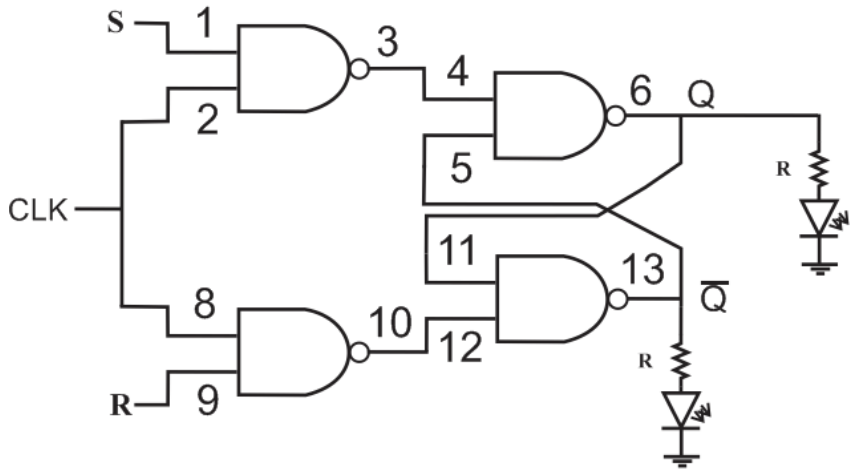
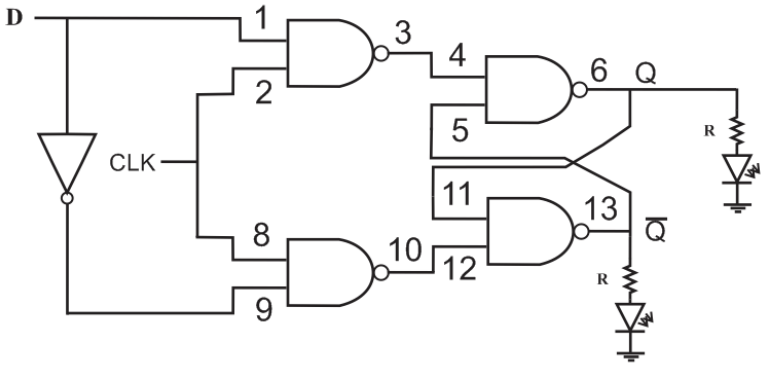


Fig. a. S-R Flip-Flop

Truth table: -

Inputs			Output		Inference
CLK	**S**	**R**	**Q**	**Q**	
0	X	X	X	X	No change
1	0	0	X	X	No change
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	1	1	Invalid

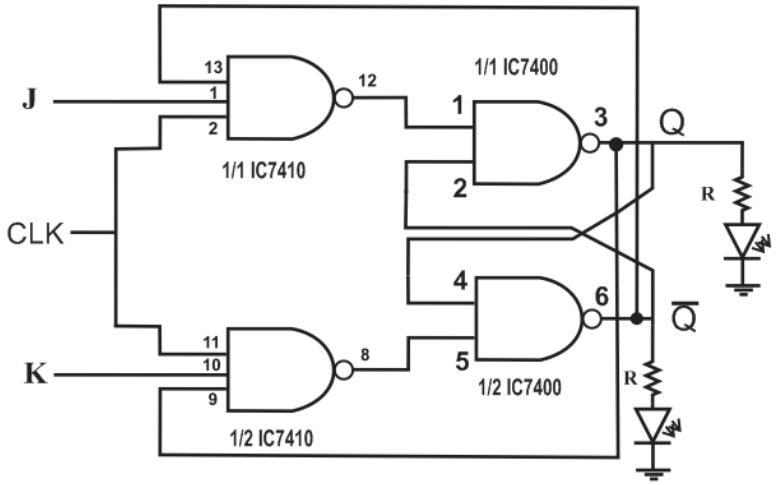
2. D Flip-flop



Truth table:

Inputs		Output Q	Inference
CLK	D		
0	X	X	No change
1	0	0	Reset
1	1	1	Set

3. JK Flip-Flop



Truth table: -

Inputs			Output		Inference
CLK	J	K	Q	$\overline{Q}$	
0	X	X	X	X	No change
1	0	0	X	X	No change
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	1	1	Toggle

Procedure:

1. Make the connection for RS flip flop
2. Apply the proper inputs and clock signal.
3. Observe the corresponding outputs and compare with the table given
4. Repeat the same procedure for JK and D-F.F.

Observation Table:

1. SR Flip-flop

Inputs			Q (Output)		Q	
CLK	**S**	**R**	**LED State**	**Logic Level**	**LED State**	**Logic Level**
0	X	X				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

2. D FF

Inputs		Q (Output)	
CLK	**D**	**LED State**	**Logic Level**
0	X		
1	0		
1	1		

3. J-K Flip-flop

Inputs			Q (Output)		Q	
CLK	**J**	**K**	**LED State**	**Logic Level**	**LED State**	**Logic Level**
0	X	X				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

Theory:

A flip-flop is a bi-stable electronic circuit. Flip-flop has 2 stable output states, 0 and 1. The stable state of a flip-flop can be changed only by changing the set of its inputs.

R-S Flip-flop:

Is basically a device that has 2 inputs along with the clock and 2 outputs, one output being the complement of the other. 2 inputs are called Set and Reset.

The clocked RS flip flop has additional input called clock which control the action of flip-flop.

When clock goes low, we get a stable output. Circuit will operate as an RS flip-flop only when clock goes high.

- **Case 1:** When S=0,R=0; clock pulse have no effect. Output Q retains its last value. So No change in the output
- **Case 2:** When S=0 and R=1; Q=0 and Q=1... Hence flip-flop is in Reset condition.
- **Case 3:** When S=1 and R=0, Q=1 and Q=0...Hence flip-flop is in Set condition.
- **Case 4:** When S=R=1,outputs of NAND gates A and B are 0.So both gates C and D receive one input as 0.Hence their outputs go to 1 .

Thus Q=Q=1, which is not possible as Q and Q should be complementary. This is not allowed condition

D Flip Flop:-

The D Flip-Flop is very useful when a signal bit is to be stored. It can be developed from RS flip –flop using a signal inverter.

The flip flop has a1 i/p & a clock.

- **Case1:** When D=0 S=0, R=1 Q=0. The flip-flop is in Reset state.
- **Case 2:** When D=1 S=1, R=0 So Q=1 hence flip-flop is in Set state.

Q will always follow D input after some time delay. Hence it is also called as Delay flip-flop.

J-K flip-flop:-

In JK flip-flop, output Q is fed as an additional input to gate A and Q is fed as an additional input to gate B apart from JK inputs and clock.

- **Case 1:** When J & K both are low both the NAND gate are disabled, c/k pulse have no effect. The o/p of Q retains its last value i.e. No change in the outputs
- **Case 2:** If 'J' is low & 'K 'is high the upper gate is disabled. So flip-flop is in Reset condition.

- **Case 3:** When J is high & K is low, Q=1 and Q-0 i.e. flip-flop is in Set state.
- **Case 4:** When J & K is high, Q and Q are inverted. This is called as toggling mode. That means the flip flop will toggle.

Result:

Conclusion:

Questions:

1. **Define Flip-Flop and mention its types.**

2. **Explain how T-Flip-flop is formed by using JKFlip-Flop.**

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

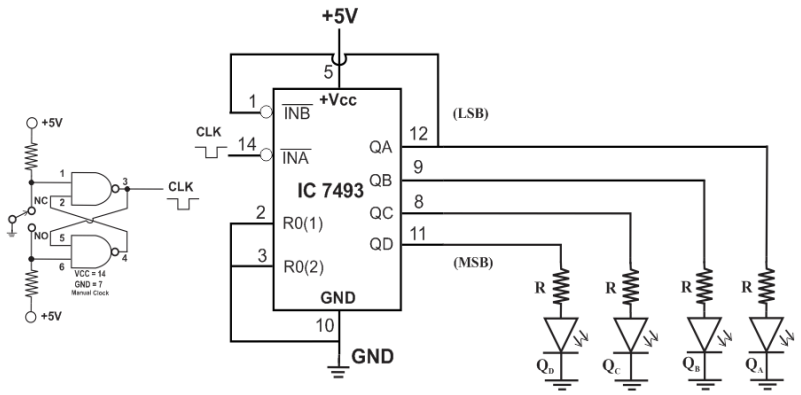
Signature of Instructor

Assignment 8

Aim: Study of 4-bit binary asynchronous counter using IC 7493

Requirements: Circuit Board/ Digital Trainer kit and Connecting Wires etc.

Circuit Diagram:



Procedure:

1. Construct the circuit as shown in figure.
2. Do the connections as shown in figure.
3. Apply Clock inputs and note down the Q**D, QC, QB** and **QA** in tabular format.
4. Verify it with expected output.

Observations:

Clock	**Outputs**			
INA	**QD**	**QC**	**QB**	**QA**

Theory:

- **Counter**

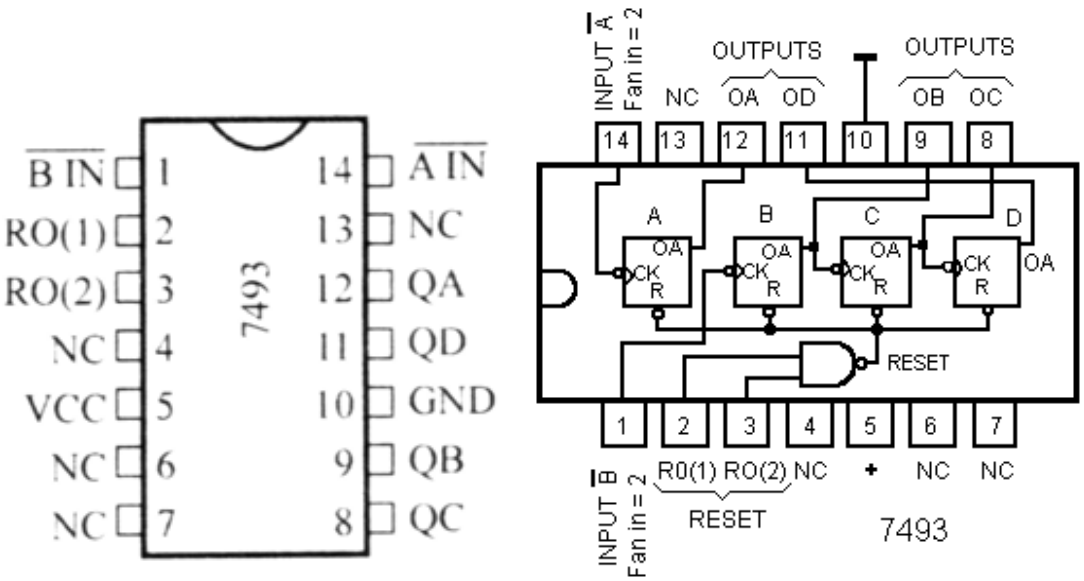
A counter is a device which can count any particular event on the basis of how many times the particular event(s) is occurred. In a digital logic system or computers, this counter can count and store the number of time any particular event or process have occurred, depending on a clock signal. Most common type of counter is sequential digital logic circuit with a single clock input and multiple outputs. The outputs represent binary or binary coded decimal numbers. Each clock pulse either increases the number or decreases the number.

- **Asynchronous Counter:**

Asynchronous stands for the absence of synchronization. Something that is not existing or occurring at the same time. In computing or telecommunication stream, Asynchronous stands for controlling the operation timing by sending a pulse only when the previous operation is completed rather than sending it in regular intervals.

An Asynchronous counter can count using **Asynchronous clock input**. Counters can be easily made using flip-flop. As the count depends on the clock signal, in case of an Asynchronous counter, changing state bits are provided as the clock signal to the subsequent flip-flops. Those Flip- flops are serially connected together, and the clock pulse ripples through the counter. Due to the ripple clock pulse, it's often called a ripple counter. An Asynchronous counter can count $\mathbf{2^n - 1}$ possible counting state

IC 7493 Pin Diagram



This is a 4-bit ripple type decade binary counter, which consists of four master/slave JK flip-flops connected to provide a divide-by-two section and a divide-by-eight section. Each section has a separate clock input, which initiates state changes of the counter on the high-to-low clock transition. For every 16 pulses at the input, it will generate one pulse at the output. Pin 1 connects to pin 12 so that the output of the first stage is fed to the input of the second stage. The input is at pin 14 (CP), and the final output is at pin 11 (Q3).

A divide-by-16 counter, also called divide-by-sixteen counter, is a counter circuit used in the field of digital electronics, which produces a single pulse at the output for every 16 pulses at the input.

Advantages and Disadvantages of Asynchronous Counter:

- While using the Asynchronous counter, an **additional re-synchronizing output flip-flops required** for desynchronizing the flip-flops. Also, for the truncated sequence count, when it is not equal to, extra feedback logic is needed.
- When counting a large number of bits, due to the chain system, **propagation delay** by successive stages became too large which is very difficult to get rid off. In such a situation, Synchronous counters are faster and reliable. There are also **counting errors** in Asynchronous Counter when high clock frequencies are applied across it.

Result:

Conclusion:

Questions:

1. **What is difference between Asynchronous and Synchronous Counter?**

2. Give applications of Counters.

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

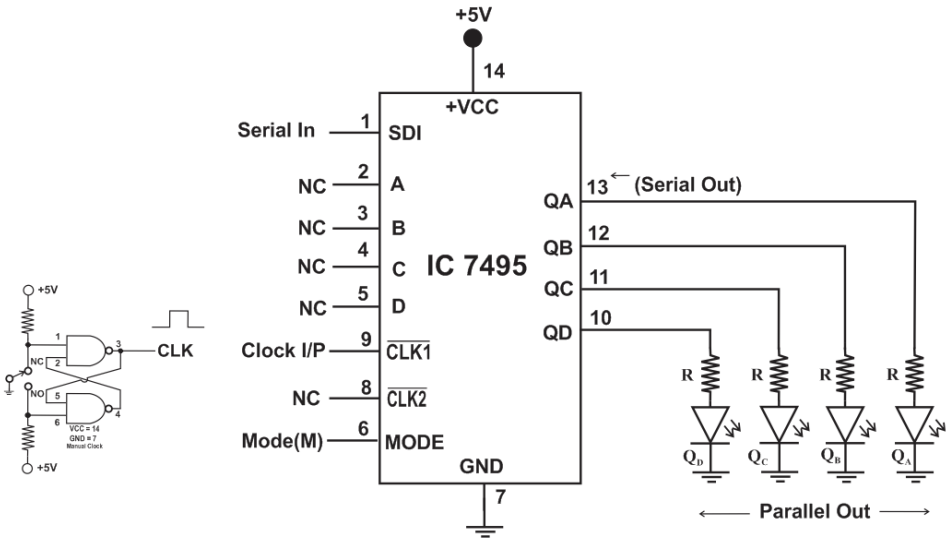
Signature of Instructor

Assignment 9

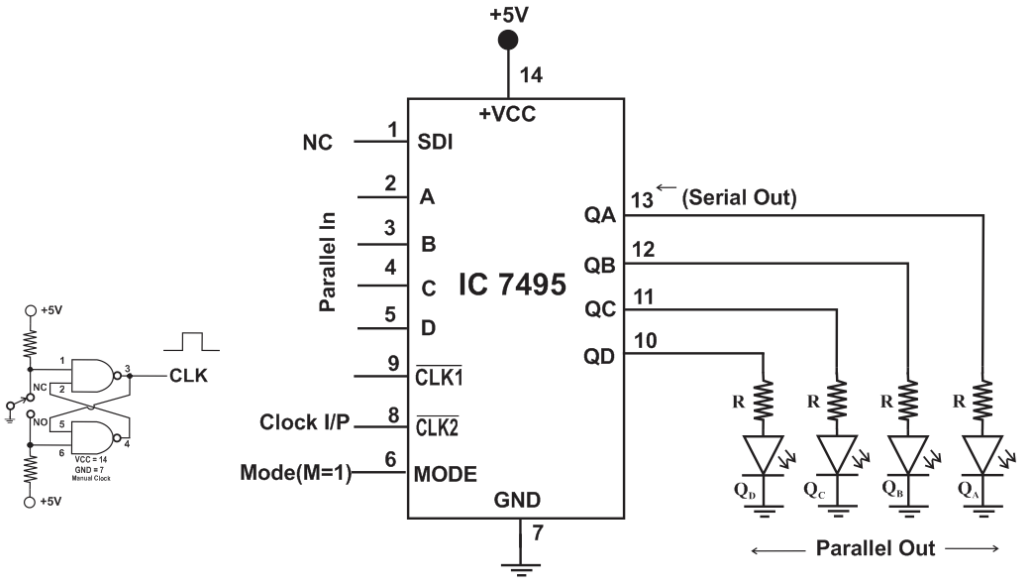
Aim: Study of Shift Register using IC 7495

Requirements: Circuit Board / Digital Trainer Kit and Connecting Wires, etc.

Circuit Diagram:



SISO and SIPO



PISO and PIPO

Procedure:

1. Connect the wires as shown in the logic circuit diagram.
2. Keep the logic control Mode on '0' for SISO and SIPO. For PISO and PIPO keep the logic control Mode on '1'.
3. Apply other inputs to the logic circuit via the input logic switches according to operation.
4. Apply the clock input and note down the corresponding outputs in tabular format.

Observations:

SISO:

Clock	Data in	Outputs of Shift Registers
	SDI	QA

SIPO:

Clock	Data in	Outputs of Shift Registers			
	SDI	QA	QB	QC	QD

PIPO:

Clock	Data in				Outputs of Shift Registers			
	A	B	C	D	QA	QB	QC	QD

PISO:

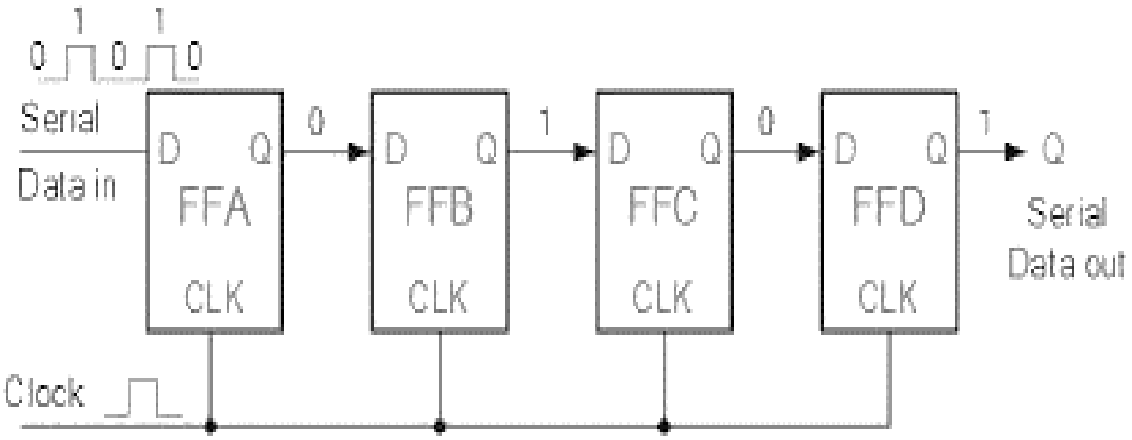
Clock	Data in				Outputs of Shift Registers			
	A	B	C	D	QA	QB	QC	QD

Theory:

Shift registers are a type of sequential logic circuit, mainly for storage of digital data. They are a group of flip-flops connected in a chain so that the output from one flip-flop becomes the input of the next flip-flop. All the flip-flops are driven by a common clock, and all are set or reset simultaneously. The serial in/serial out shift register accepts data serially –that is, one bit at a time on a single line. It produces the stored information on its output also in serial form. The serial in/parallel out shift register accepts data serially –that is, one bit at a time on a single line. It produces the stored information on its output in parallel form. The parallel in/parallel out shift register accepts data in parallel. It produces the stored information on its output in parallel form.

SISO (Serial in Serial out):

The input to this register is given in serial fashion i.e. one bit after the other through a single data line and the output is also collected serially. The data can be shifted only left or shifted only right. Hence it is called Serial in Serial out shift register or a SISO shift register. The registers which will shift the bits to left are called "Shift left registers". The registers which will shift the bits to right are called "Shift right registers"

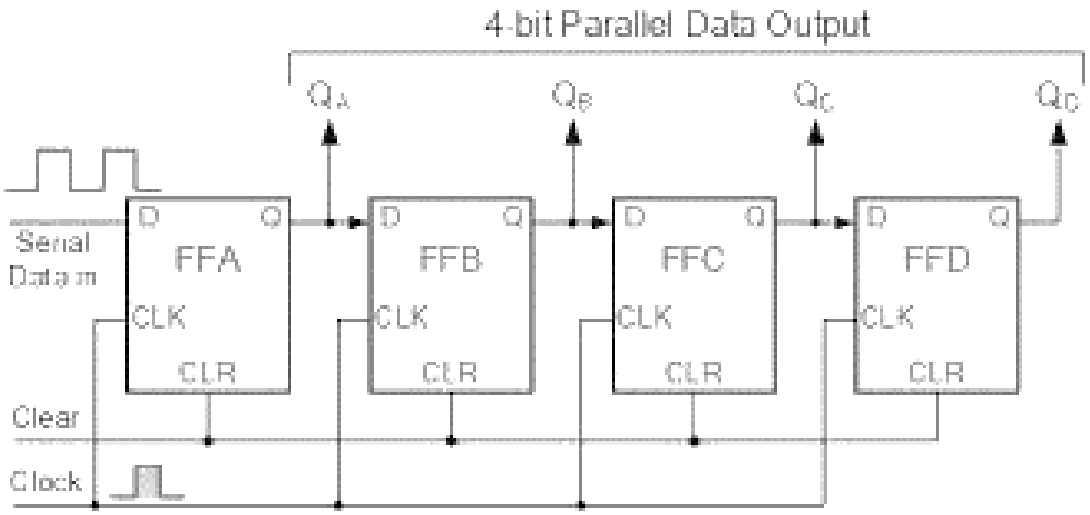


In the above diagram the serial data is applied at the left side of the flip flop. In this shift register, when the clock signal is applied and the serial data is given; only one bit will be available at output at a time in the order of the input data. The SISO shift register can be used as temporary data storage device. But the main use of a SISO is to act as a delay element.

SIPO (Serial in Parallel out):

In such types of operations, the data is entered serially and taken out in parallel fashion. Data is loaded bit by bit. The outputs are disabled as long as the data is loading. As soon as the data loading gets completed, all the flip-flops contain their required data; the outputs are enabled so that all the loaded data is made available over all the output lines at the same time. 4 clock cycles are required to load a four bit word. Hence the speed of operation of SIPO mode is same as that of SISO mode. The main application of Serial in Parallel out shift register is to convert serial data into parallel data. Hence they are used in communication lines where de-multiplexing of a data line into several parallel line is required.

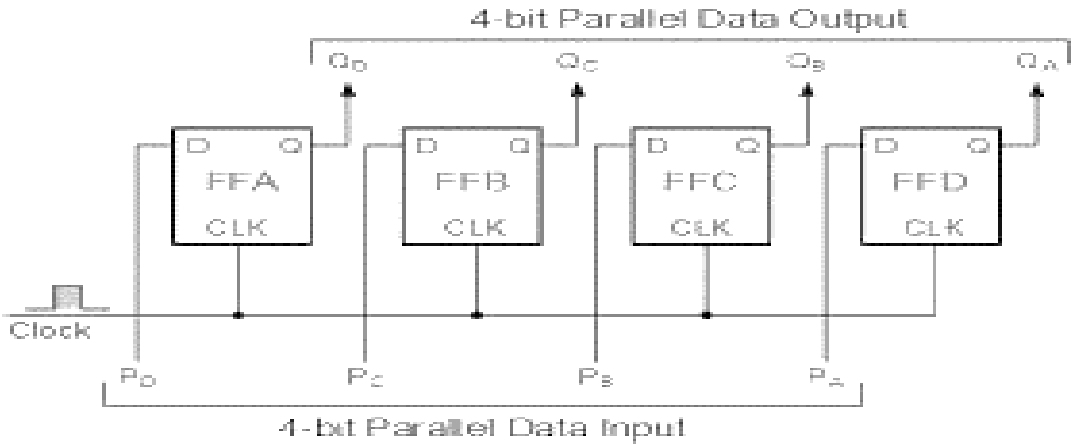
PISO Parallel in to Serial-out:



In this type of register the data is loaded into the register in a parallel format in which all the data bits enter their inputs simultaneously, to the parallel input pins P_A to P_D of the register. The data is then read out sequentially in the normal shift-right mode from the register at Q representing the data present at P_A to P_D. This data is outputted one bit at a time on each clock cycle in a serial format.

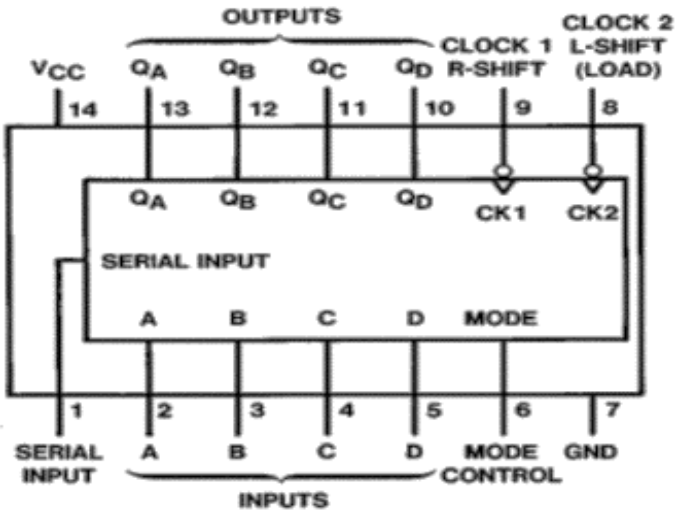
Parallel in Parallel out shift register:

In this register, the input is given in parallel and the outputalso collected in parallel. The clear (CLR) signal and clock signals are connected to all the 4 flip flops. Data is given as input separately for each flip flop and in the same way, output also collected individually from each flip flop.



As shown in the figure the data is presented in a parallel format to the parallel input pins P_A to P_D and then transferred together directly to their respective output pins Q_A to Q_D by the same clock pulse. Then one clock pulse loads and unloads the register. This type of register also acts as a temporary storage device or as a time delay device.

Pin diagram of Shift Register IC 7495



Result:

Conclusion:

Questions:

1. **Write Applications of Shift Registers.**

2. **Draw SISO shift register output waveforms.**

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

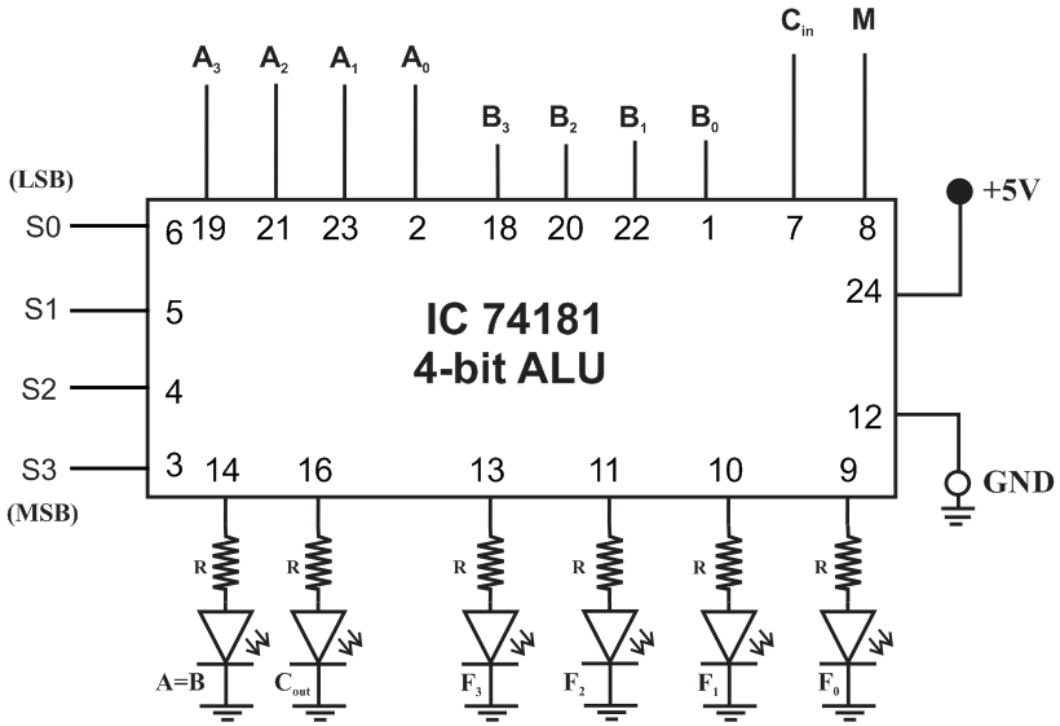
Signature of Instructor

Assignment 10

Aim: To Study basic Logical and Arithmetic operations using 4-bit ALU

Requirements: Circuit Board/ Digital Trainer kit and Connecting Wires etc.

Circuit Diagram:



4-Bit Arithmetic Logic Unit

Procedure:

1. Connect A0-A3 and B0-B3 to eight switches.
2. Connect F0-F3 to four LEDs.
3. The select lines S0-S3 an mode M are set to the logical function.
4. Change the connections for select lines (S0-S3) and perform few Logical functions by setting the input switches A & B to the appropriate values.
5. Record the output data from the LEDs status of F3 to F0.
6. Then perform the arithmetic function of 74181 by setting the input switches to the appropriate values.
7. Record the output data from the LEDs (F3 - F0) and verify the outcomes.

Observations:

1. For Active High Inputs and Outputs: Logical operation **Logic (M = H)**

Mode Select input				A-input				B-input				Output			
S3	**S2**	**S1**	**S0**	**A3**	**A2**	**A1**	**A0**	**B3**	**B2**	**B1**	**B0**	**F3**	**F2**	**F1**	**F0**

2. For Active High Inputs and Outputs: Arithmetic operation **Arithmetic (M = L; Cn= H)**

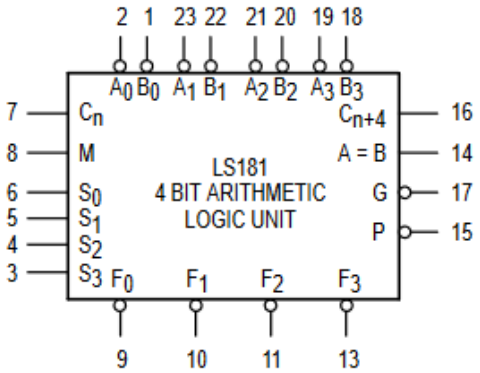
Mode Select input				A-input				B-input				Output			
S3	**S2**	**S1**	**S0**	**A3**	**A2**	**A1**	**A0**	**B3**	**B2**	**B1**	**B0**	**F3**	**F2**	**F1**	**F0**

Theory:

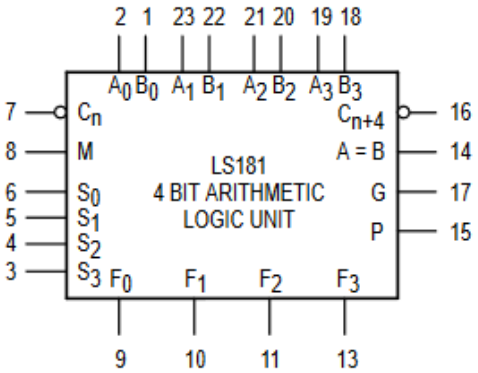
An **arithmetic logic unit (ALU)** is a digital circuit used to perform arithmetic and logic operations. It represents the fundamental building block of the **central processing unit (CPU)** of a computer. Modern CPUs contain very powerful and complex ALUs.

LOGIC SYMBOLS

ACTIVE LOW OPERANDS



ACTIVE HIGH OPERANDS



Most of the operations of a CPU are performed by one or more ALUs, which load data from input registers. A **register** is a small amount of storage available as part of a CPU. The control unit tells the ALU what operation to perform on that data, and the ALU stores the result in an output register. The control unit moves the data between these registers, the ALU, and memory.

The ALU device used in this experiment is 74181 and its pin assignment is shown in Fig. 74LS181 is a 4-bit high speed parallel Arithmetic Logic Unit (ALU). It consists of four parallel full adder/subtractor circuits. Data applied on the inputs Ai's and Bi's are processed and the sum or difference is available at the output in a parallel format. Controlled by the four Function Select Inputs (S0 . . . S3) and the Mode Control Input (M), it can perform all the 16 possible logic operations or 16 different arithmetic operations on active HIGH or active LOW operands. When the Mode Control Input is HIGH, all internal carries are inhibited and the device performs logic operations on the individual bits as listed. When the Mode Control Input is LOW, the carries are enabled and the device performs arithmetic operations on the two 4-bit words.

Functional Description:

FUNCTION TABLE

MODE SELECT INPUTS				ACTIVE LOW INPUTS & OUTPUTS		ACTIVE HIGH INPUTS & OUTPUTS	
S_3	S_2	S_1	S_0	LOGIC (M = H)	ARITHMETIC** (M = L) (C_n = L)	LOGIC (M = H)	ARITHMETIC** (M = L) (C_n = H)
L	L	L	L	$\overline{A}$	A minus 1	$\overline{A}$	A
L	L	L	H	$\overline{AB}$	AB minus 1	$\overline{A}$ + B	A + B
L	L	H	L	A + B	AB minus 1	$\overline{AB}$	A + B
L	L	H	H	Logical 1 minus 1		Logical 0 minus 1	
L	H	L	L	$\overline{A}$ + B	A plus $\overline{(A + B)}$	$\overline{AB}$	A plus $\overline{AB}$
L	H	L	H	$\overline{B}$	AB plus (A + B)	B	(A + B) plus AB
L	H	H	L	A ⊕ B	A minus B minus 1	$\overline{A \oplus B}$	A minus B minus 1
L	H	H	H	$\overline{A}$ + B	A + B	$\overline{AB}$	AB minus 1
H	L	L	L	AB	A plus (A + B)	$\overline{A + B}$	A plus AB
H	L	L	H	A ⊕ B	A plus B	A ⊕ B	A plus B
H	L	H	L	B	AB plus (A + B)	B	(A + B) plus AB
H	L	H	H	A + B	A + B	AB	AB minus 1
H	H	L	L	Logical 0 A plus A*		Logical 1 A plus A*	
H	H	L	H	AB	AB plus A	A + B	(A + B) plus A
H	H	H	L	AB	AB plus A	A + B	(A + B) Plus A
H	H	H	H	A	A	A	A minus 1

L = LOW Voltage Level
H = HIGH Voltage Level
*Each bit is shifted to the next more significant position
**Arithmetic operations expressed in 2s complement notation

Result:

Conclusion:

Questions:

1. **What is ALU?**

2. **Write steps to perform multiplication using 4-bit ALU IC74181.**

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

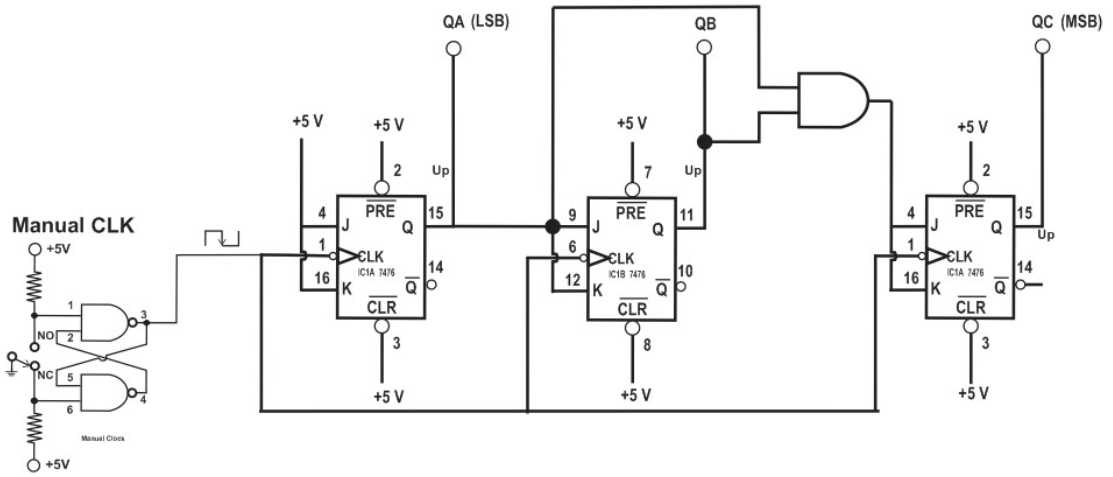
Signature of Instructor

Assignment 11

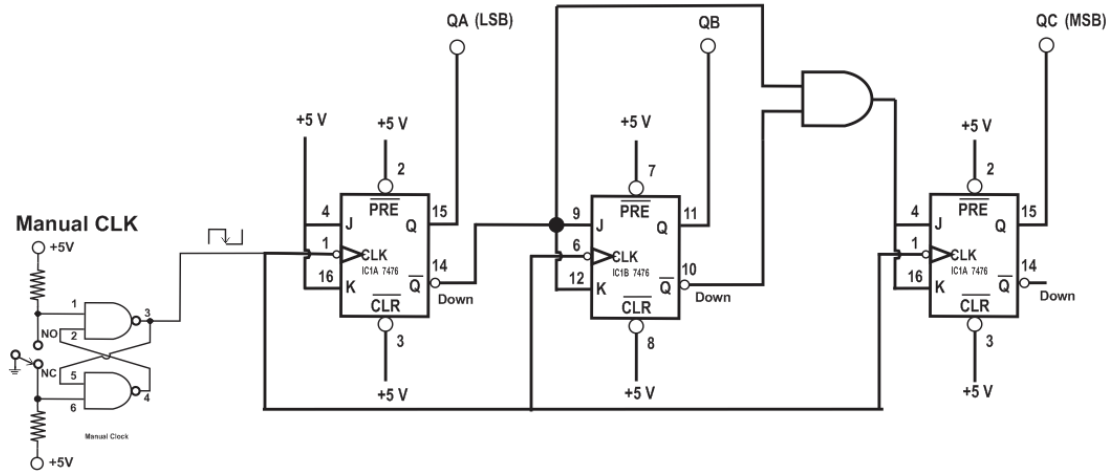
Aim: Study of 3-bit Synchronous Up-Down counter

Requirements: Circuit Board/ Digital Trainer kit and Connecting Wires etc.

Circuit Diagram:



3-bit synchronous Up Counter



3-bit synchronous Down Counter

Procedure:

1. Construct the circuit as shown infigure.
2. Do the connections as shown in figure.
3. Apply Clock inputs and note down the Q**D, QC, QB** and **QA** in tabular format.
4. Verify it with truth table.
5. Repeat same procedure.

Observations:

A. 3-bit Synchronous UP counter:

CLk	Present State			Next State		
↓	QC	QB	QA	QC	QB	QA
	0	0	0	0	0	1

LED ON = logic'1' LED OFF= logic'0'

B. 3-bit Synchronous Down counter:

CLk	Present State			Next State		
↓	QC	QB	QA	QC	QB	QA
	1	1	1	1	1	0

LED ON = logic'1' LED OFF= logic'0'

Theory:

Counter

A counter is a device which can count any particular event on the basis of how many times the particular event(s) is occurred. In a digital logic system or computers, this counter can count and store the number of time any particular event or process have occurred, depending on a clock signal. Most common type of counter is sequential digital logic circuit with a single clock input and multiple outputs. The outputs represent binary or binary coded decimal numbers. Each clock pulse either increases the number or decreases the number.

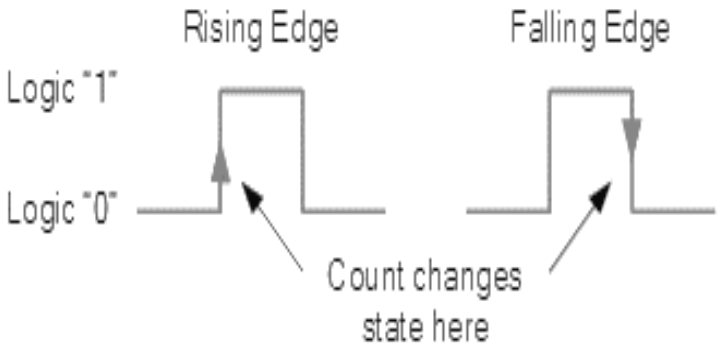
Synchronous Counter

Synchronous Counters are so called because the clock input of all the individual flip-flops within the counter are all clocked together at the same time by the same clock signal

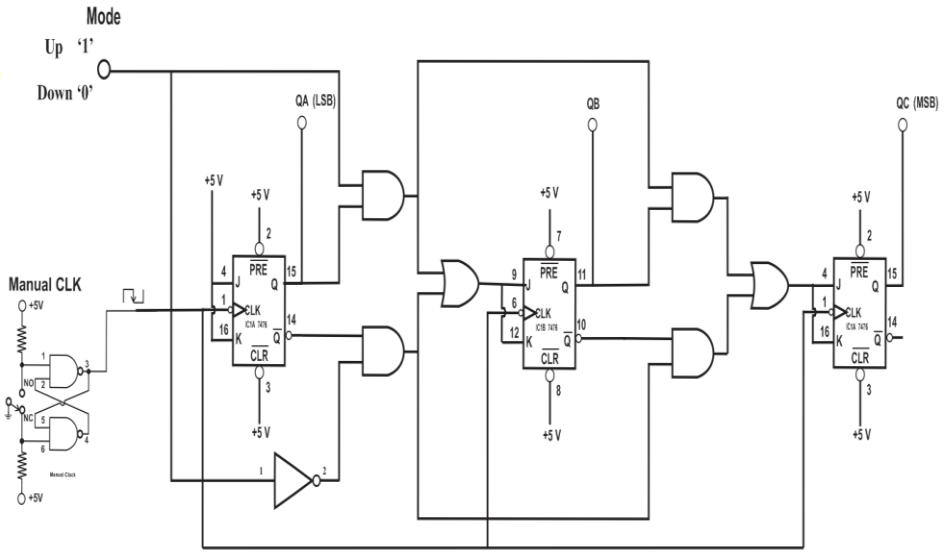
Triggering a Synchronous Counter

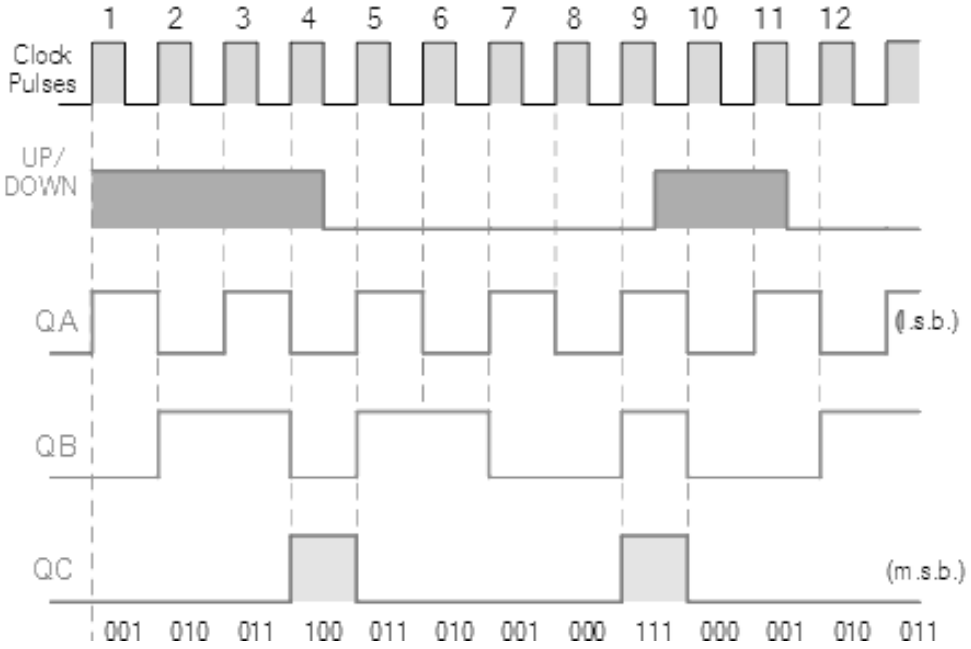
Synchronous Counters use edge-triggered flip-flops that change states on either the "positive-edge" (rising edge) or the "negative-edge" (falling edge) of the clock pulse on the control input resulting in one single count when the clock input changes state.

Generally, synchronous counters count on the rising-edge which is the low to high transition of the clock signal and asynchronous ripple counters count on the falling-edge which is the high to low transition of the clock signal.



Synchronous 3-bit Up/Down Counter





The circuit above is of a simple 3-bit Up/Down synchronous counter using JK flip-flops configured to operate as toggle or T-type flip-flops giving a maximum count of zero (000) to seven (111) and back to zero again. Then the 3-Bit counter advances upward in sequence (0,1,2,3,4,5,6,7) or downwards in reverse sequence (7,6,5,4,3,2,1,0).

Generally most bidirectional counter chips can be made to change their count direction either up or down at any point within their counting sequence. This is achieved by using an additional input pin which determines the direction of the count, either Up or Down and the timing diagram gives an example of the counters operation as this Up/Down input changes state.

Result:

Conclusion:

Questions:

1. **What is Synchronous Counter?**

2. **Give applications of Synchronous Counters.**

Assignment Evaluation		
0: Not Done [ ]	1: Incomplete [ ]	2:Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor