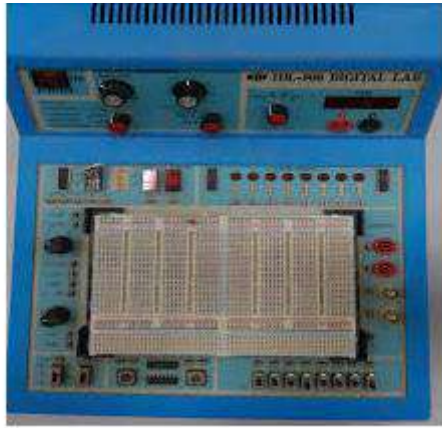


EC 230 LOGIC CIRCUIT DESIGN LAB MANUAL



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Vision

To nurture the talents of electronics and communication engineers, making them highly competent for growth of the society.

Mission

- To deliver excellence in teaching - learning process.
- Promote safe, orderly, caring and supportive environment to learners.
- Development of skilled engineers to perform innovative Research for betterment of the society.
- To encourage industry - institute interaction, career advancement, innovation and entrepreneurship development.

Program Educational Outcome (PEO)

PEO1: To acquire a strong foundation in mathematics and scientific fundamentals, to develop an ability to analyze various functional elements of different disciplines of electronics and communication engineering.

PEO2: Develop technical competence to move in pace with rapid changes in technology.

PEO3: Equip learners to strengthen knowledge and soft skills for carrier advancement.

PEO4: Adhere to ethics to contribute for betterment of the society.

Program Specific Outcomes (PSO)

PSO1. To understand principles and applications of various electronic components/devices and circuits.

PSO2. Enable learners to solve complex problems using modern hardware and software tools.

COURSE CODE	COURSE NAME	L-T-P-C	YEAR OF INTRODUCTION
EC230	LOGIC CIRCUIT DESIGN LAB	0-0-3-1	2016
Prerequisite: EC207 Logic circuit design			
Course objectives: <ul style="list-style-type: none"> To study the working of standard digital ICs and basic building blocks To design and implement combinational circuits To design and implement sequential circuits 			
List of Experiments: (Minimum 12 experiments are to be done) <ol style="list-style-type: none"> Realization of functions using basic and universal gates (SOP and POS forms). Design and Realization of half /full adder and subtractor using basic gates and universal gates. 4 bit adder/subtractor and BCD adder using 7483. 2/3 bit binary comparator. Binary to Gray and Gray to Binary converters. Study of Flip Flops: S-R, D, T, JK and Master Slave JK FF using NAND gates Asynchronous Counter: Realization of 4-bit counter Asynchronous Counter: Realization of Mod-N counters. Asynchronous Counter: 3 bit up/down counter Synchronous Counter: Realization of 4-bit up/down counter. Synchronous Counter: Realization of Mod-N counters. Synchronous Counter: 3 bit up/down counter Shift Register: Study of shift right, SIPO, SISO, PIPO, PISO (using FF & 7495) Ring counter and Johnson Counter. (using FF & 7495) Realization of counters using IC's (7490, 7492, 7493). Multiplexers and De-multiplexers using gates and ICs. (74150, 74154), Realization of combinational circuits using MUX & DEMUX. Random sequence generator. LED Display: Use of BCD to 7 Segment decoder / driver chip to drive LED display Static and Dynamic Characteristic of NAND gate (MOS/TTL) 			
Expected outcome:			
The student should be able to:			
<ol style="list-style-type: none"> Design and demonstrate functioning of various combination circuits Design and demonstrate functioning of various sequential circuits Function effectively as an individual and in a team to accomplish the given task 			

EC 230 - LOGIC CIRCUIT DESIGN LAB**COURSE OUTCOME, CO-PO MAPPING**

C230.1	The students will be able to demonstrate the function of standard digital ICs and basic building block
C230.2	The students will be able to do the design and implementation of combinational circuits.
C230.3	The students will be able to do the design and implementation of sequential circuits.
C230.4	The students will be able to function as an individual and in a team to accomplish the given task.

CO/ PO	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
EC2 30.1	2	-	-	-	-	-	-	-	2	-	-	2	2	-
EC2 30.2	2	-	-	-	-	-	-	-	3	-	-	-	3	-
EC2 30.3	2	-	-	-	-	-	-	-	3	-	-	-	3	-
EC2 30.4	-	-	-	-	-	-	-	-	3	-	-	-	-	-

LIST OF EXPERIMENTS

1. FAMILIARIZATION OF LOGIC GATES
2. IMPLEMENTATION OF GIVEN BOOLEAN EXPRESSION USING LOGIC GATES IN SOP AND POS FORMS
3. DESIGN AND IMPLEMENTATION OF ARITHMETIC CIRCUITS
4. BIT ADDER/SUBTRACTOR AND BCD ADDER USING 7483
5. 2/3 BIT BINARY COMPARATOR
6. BINARY TO GRAY AND GRAY TO BINARY CONVERTERS
7. FLIP FLOPS
8. ASYNCHRONOUS COUNTERS
9. SYNCHRONOUS COUNTERS
10. RANDOM SEQUENCE GENERATOR
11. SHIFT REGISTERS
12. RING COUNTER AND JOHNSON COUNTER
13. REALIZATION OF COUNTERS USING IC'S(7490,7492,7493)
14. MULTIPLEXERS AND DEMULTIPLEXERS USING GATES AND ICS(74150,74154)
15. REALIZATION OF COMBINATIONAL CIRCUITS USING MUX AND DEMUX
16. USE OF BCD TO 7 SEGMENT DECODER/ DRIVER CHIP TO DRIVE LED DISPLAY
17. TRANSFER CHARACTERISTICS OF TTL AND CMOS NAND GATE

EXPT NO: 1_

DATE: __/__/__

FAMILIARIZATION OF LOGIC GATES**OBJECTIVE:**

To familiarize logic gates.

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	7408	1
		7432	1
		7404	1
		7400	1
		7402	1
		7486	1
		7410	1
		7411	1

INTRODUCTION:

A **logic gate** is an idealized or physical device implementing a Boolean function, that is, it performs a logical operation on one or more logic inputs and produces a single logic. There are seven basic logic gates: AND, OR, XOR, NOT, NAND, NOR, and XNOR.

The **AND gate** is so named because, if 0 is called "false" and 1 is called "true," the gate acts in the same way as the logical "and" operator. The output is "true" when both inputs are "true." Otherwise, the output is "false."

The **OR gate** gets its name from the fact that it behaves after the fashion of the logical inclusive "or." The output is "true" if either or both of the inputs are "true." If both inputs are "false," then the output is "false."

A **NOT gate** sometimes called a logical inverter to differentiate it from other types of electronic inverter devices, has only one input. It reverses the logic state

The **NAND gate** operates as an AND gate followed by a NOT gate. It acts in the manner of the logical operation "and" followed by negation. The output is "false" if both inputs are "true." Otherwise, the output is "true."

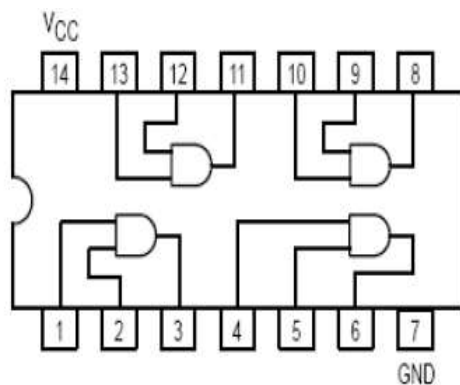
The **NOR gate** is a combination OR gate followed by an inverter. Its output is "true" if both inputs are "false." Otherwise, the output is "false."

The **XOR(exclusive-OR) gate** acts in the same way as the logical "either/or." The output is "true" if either, but not both, of the inputs are "true." The output is "false" if both inputs are "false" or if both inputs are "true." Another way of looking at this circuit is to observe that the output is 1 if the inputs are different, but 0 if the inputs are the same.

The **XNOR (exclusive-NOR) gate** is a combination XOR gate followed by an inverter. Its output is "true" if the inputs are the same, and "false" if the inputs are different

CIRCUIT DIAGRAM AND OBSERVATIONS:

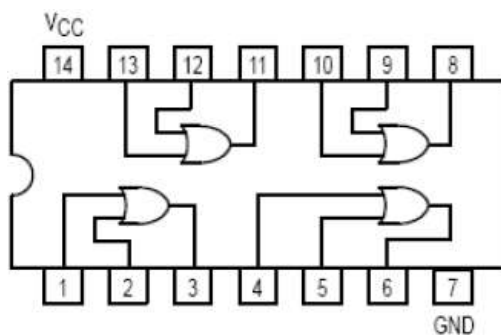
1) AND Gate-7408LS



Truth table

A	B	O/P
0	0	0
0	1	0
1	0	0
1	1	1

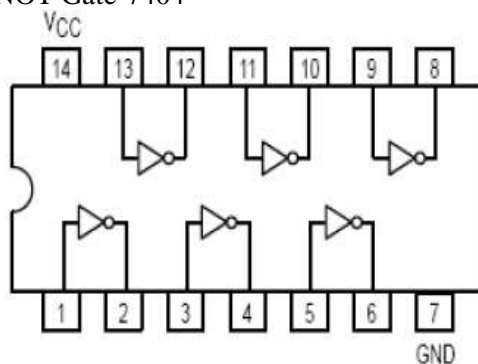
2) OR Gate-7432LS



Truth table

A	B	O/P
0	0	0
0	1	1
1	0	1
1	1	1

3) NOT Gate-7404

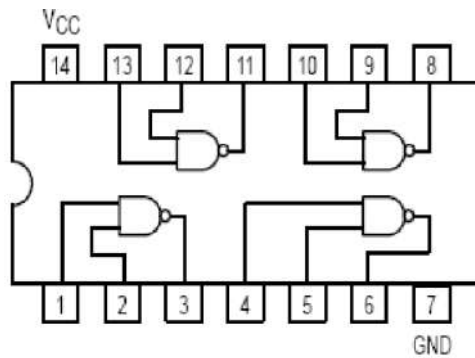


Truth table

A	O/P
0	1
1	0

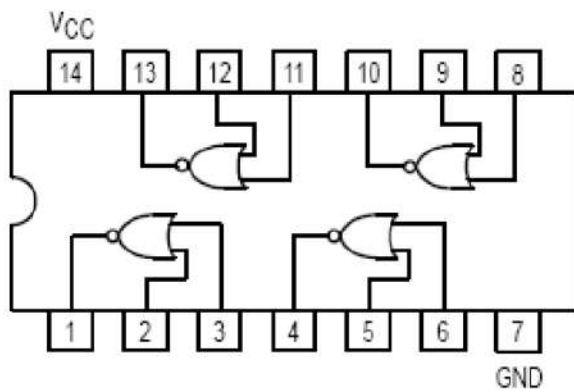
4) NAND Gate-7400LS

Truth table



A	B	O/P
0	0	1
0	1	1
1	0	1
1	1	0

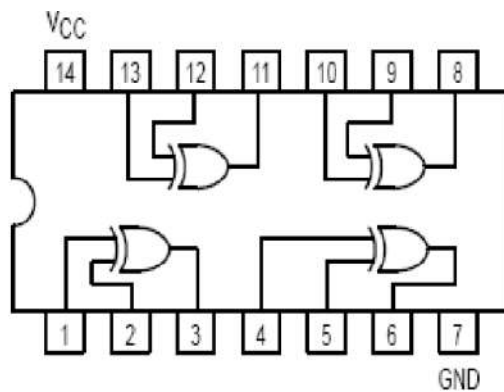
5) NOR Gate-7402LS



Truth table

A	B	O/P
0	0	1
0	1	0
1	0	0
1	1	0

6) EX-OR Gate-7486LS

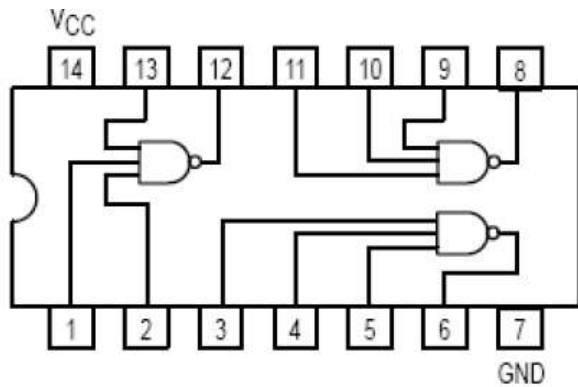


Truth table

A	B	O/P
0	0	0
0	1	1
1	0	1
1	1	0

7) 3 INPUT NAND Gate-7410LS

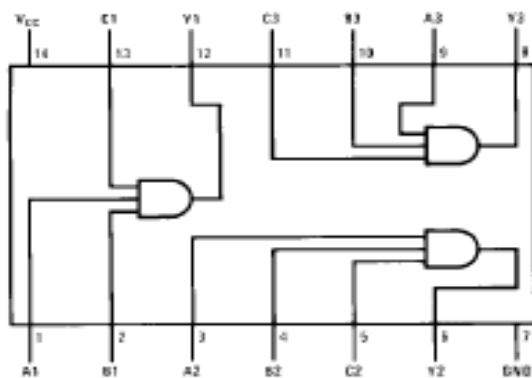
Truth table



A	B	C	O/P
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

8) 3 INPUT AND Gate-7411LS

Truth table



A	B	C	O/P
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

EXPERIMENT:**Procedure**

1. Place the IC on IC Trainer Kit.
2. Connect V_{CC} and ground to respective pins of IC Trainer Kit.
3. Connect the inputs to the input switches provided in the IC Trainer Kit.
4. Connect the outputs to the switches of output LEDs
5. Apply various combinations of inputs according to the truth table and observe condition of LEDs
6. Disconnect output from the LEDs and note down the corresponding multimeter voltage readings for various combinations of inputs.

CONCLUSION

Logic gates are familiarized.

EXPT NO: 2

DATE: __/__/__

IMPLEMENTATION OF GIVEN BOOLEAN EXPRESSION USING LOGIC GATES IN SOP AND POS FORMS

OBJECTIVE:

To design and implement the given Boolean function using logic gates in sum of product and product of sum forms.

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	7400	1
		7402	1
		7404	1
		7408LS	2
		7432	2

INTRODUCTION:

Boolean algebra provides a concise way to express the operation of a logic circuit formed by a combination of logic gates so that the output can be determined for various combinations of input values.

All Boolean expressions regardless of their form can be converted into either of two standard forms: the sum of products form or the product of sum form. Standardization makes the evaluation, simplification and implementation of Boolean expressions much more systematic and easier.

THE SUM OF PRODUCTS (SOP) FORM

When two or more product terms are summed by Boolean addition, the resulting expression is a sum of products (sop).

Example:

$$\overline{AB} + ABC$$

$$\overline{ABC} + CD + \overline{AC}$$

$$AB + ABC + AC$$

Implementation of SOP Expression

Implementing SOP expression simply requires ORing the outputs of two or more AND gates. A product term is produced by an AND operation and the sum (addition) of two or more product terms is produced by an OR operation. Therefore, an SOP expression can be implemented by AND-OR logic in which the outputs of a number (equal to the number of product terms in the expression) of AND gates connect to the inputs of an OR gate. The output X of the OR gate equals the

SOP expression. A SOP expression can always be implemented with one OR gate and two or more AND gates.

The standard POS form:

Any logic expression can be changed in to SOP form by applying Boolean algebra techniques.

The expression $A(B+CD)$ can be converted to SOP form by applying the distributive law:

Example:

$$A(B+CD) = AB + ACD$$

A standard SOP expression is one in which all the variables in the domain appear in each product term in the expression. Any nonstandard SOP expression can be converted to standard form using Boolean algebra.

THE PRODUCT-OF-SUM (POS) FORM

When two or more sum terms are multiplied, the resulting expression is a product-of-sums (POS).

Examples:

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

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

Implementation of POS expression:

Implementing a POS expression simply requires ANDing the outputs of two or more OR gates. A sum term is produced by an OR operation and the product of two or more sum term is produced by an AND operation. Therefore a POS expression can be implemented by a logic in which the outputs of number of OR gates connect to the input of an AND gate.

The standard POS form:

A standard POS expression is one in which all the variables in the domain appear in each sum term in the expression.

Example:

$$(A+B+C+D)(A+B+C+D)$$

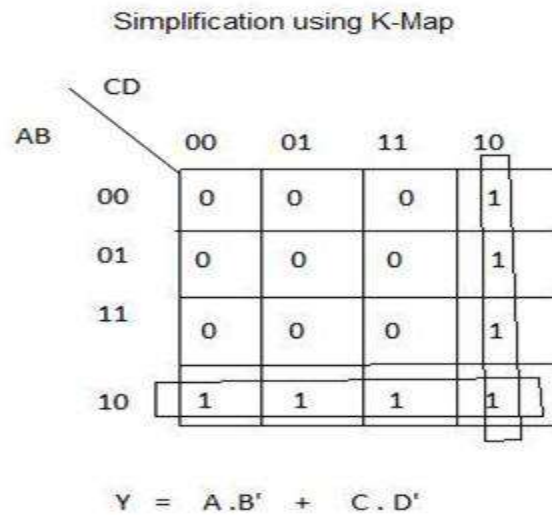
Any nonstandard POS expression can be converted to the standard form using Boolean algebra.

CIRCUIT DIAGRAM AND OBSERVATIONS:

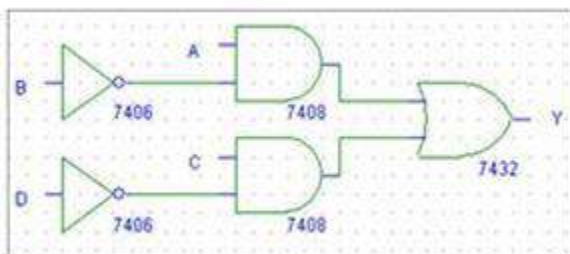
1) Realization of SOP Boolean expression:

$$Y = A'B'CD' + A'BCD' + ABCD' + AB'CD' + AB'C'D' + AB'C'D + AB'CD$$

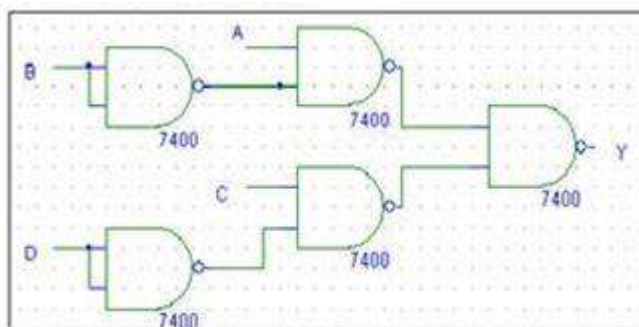
inputs				O/P
A	B	C	D	O
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0



Realization using basic Gates



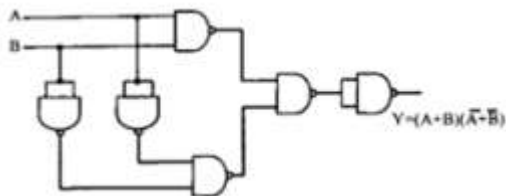
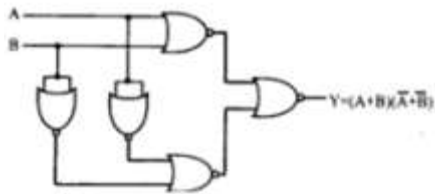
Realization using NAND Gates



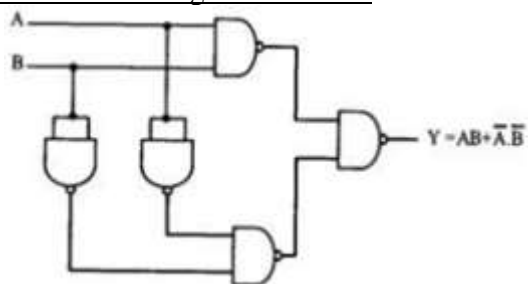
2) Realization of POS expression $Y = (A+B)(A'+B')$

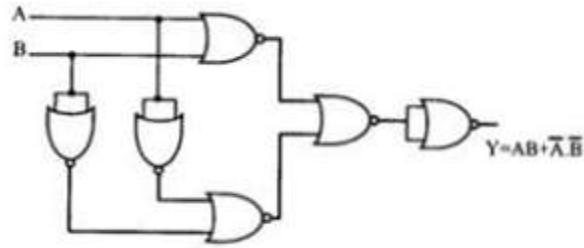
Truthtable

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Realization using NAND GatesRealization using NOR Gates**3) Realisation of SOP expression $Y=AB+A'B'$** Truthtable

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Realization using NAND GatesRealization using NOR Gates

**EXPERIMENT:****Procedure**

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit .
- 3) Set up the circuit obtained from the expression
- 4) Connect the inputs to the input switches provided in the IC Trainer Kit.
- 5) Connect the outputs to the switches of output LEDs
- 6) Apply various combinations of inputs according to the truth table and observe condition of LEDs

CONCLUSION

Implemented given boolean expression using logic gates in sop and pos forms.

REVIEW QUESTIONS:

- 1) $F = \sum m(0,1,2,3,4,5,6)$
- 2) $F = \sum m(1,2,3,6,8,12,14,15)$
- 3) $F = \prod M(0,3,5)$
- 4) $F = \prod M(1,2,3,6,8,12,14,15)$

EXPT NO: 3

DATE: __/__/__

DESIGN AND IMPLEMENTATION OF ARITHMETIC CIRCUITS**OBJECTIVE:**

To design and implement the Arithmetic Circuits

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	7400	3
		7408	1
		7486	1
		7410	1
		7432	1

INTRODUCTION:**Half adder**

With the help of half adder, we can design circuits that are capable of performing simple addition with the help of logic gates. The truth table of half adder is shown below. Sum, s is the normal output and carry, c is the carry-out.

Full adder

This type of adder is a little more difficult to implement than a half-adder. The main difference between a half-adder and a full-adder is that the full-adder has three inputs and two outputs. The first two inputs are A and B and the third input is an input carry designated as CIN. When a full adder logic is designed we will be able to string eight of them together to create a byte-wide adder and cascade the carry bit from one adder to the next. The output carry is designated as C and the normal output is designated as S.

Half subtractor:

The half subtractor is constructed using X-OR and AND Gate. The half subtractor has two input and two outputs. The outputs are difference and borrow. The difference can be applied using X-OR Gate, borrow output can be implemented using an AND Gate and an inverter.

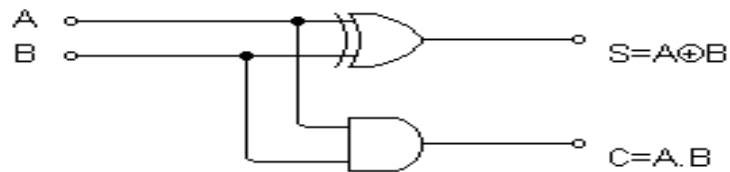
Full subtractor:

The full subtractor is a combination of X-OR, AND, OR, NOT Gates. In a full subtractor the logic circuit should have three inputs and two outputs. The two half subtractor put together gives a full

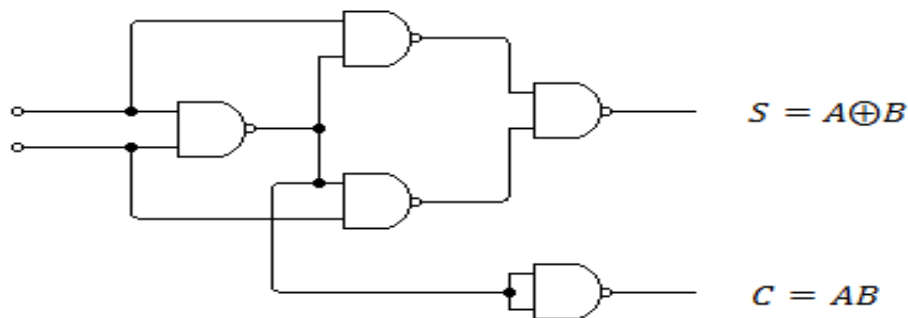
subtractor .The first half subtractor will be C and A B. The output will be difference output of full subtractor. The expression AB assembles the borrow output of the half subtractor and the second term is the inverted difference output of first X-OR.

CIRCUIT DIAGRAM:

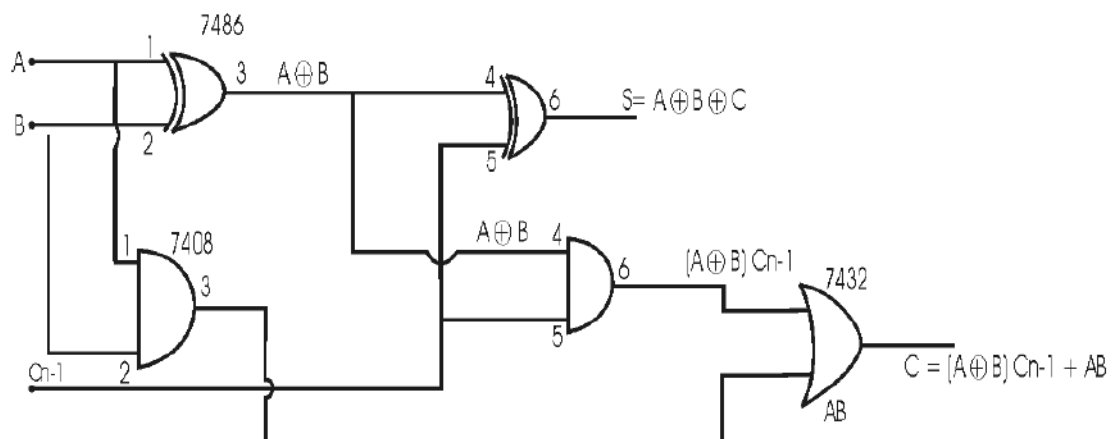
1. Half-Adder using basic gates



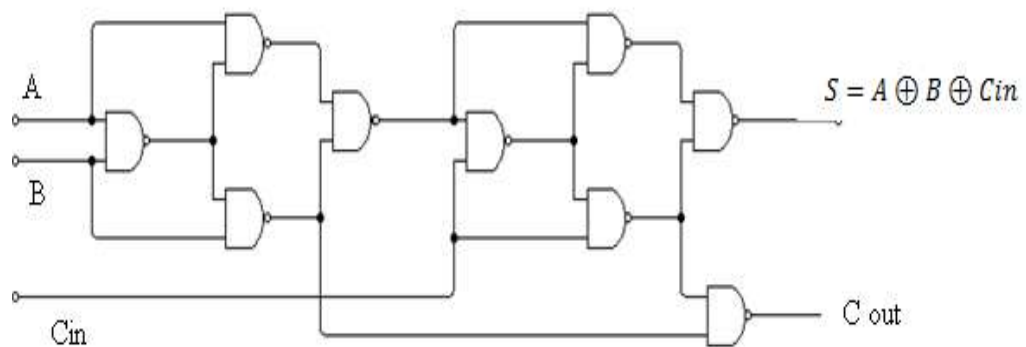
2. Half-Adder using NAND gates only



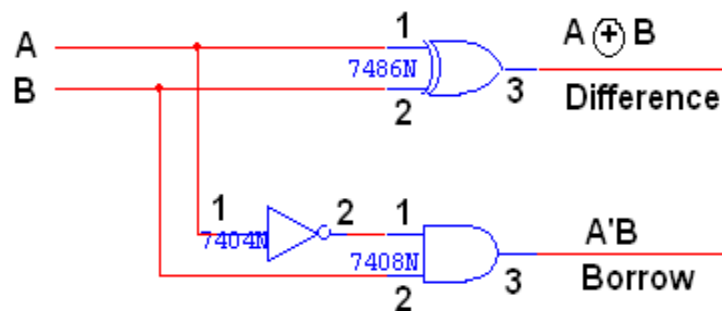
3. Full-Adder using basic gates



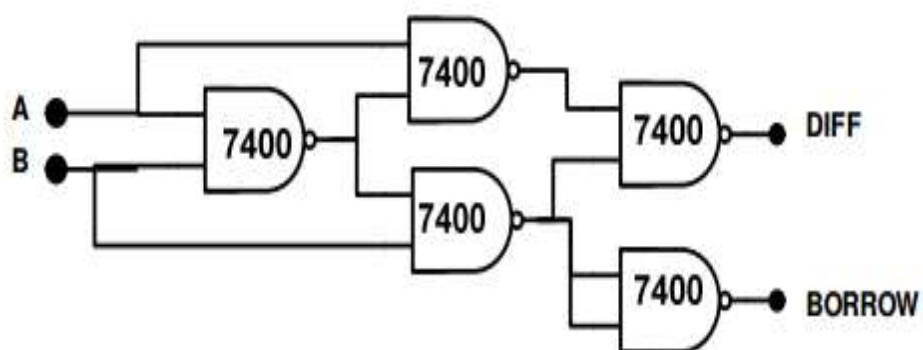
4. Full-Adder using NAND gates only



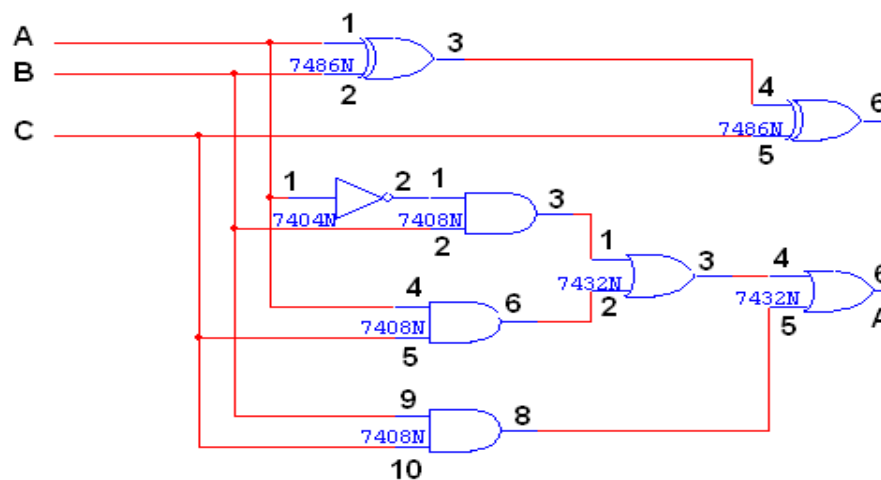
5. Half-subtractor using basic gates



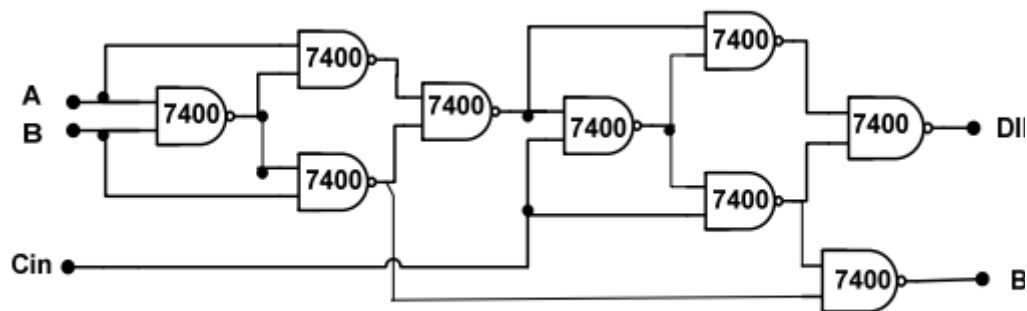
6. Half-subtractor using NAND gates only



7. Full-subtractor using Basic gates.

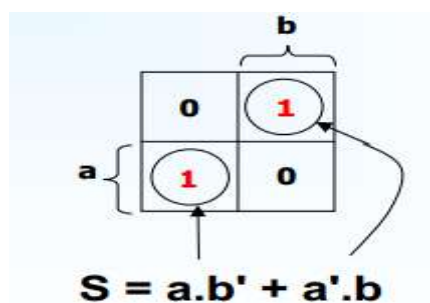
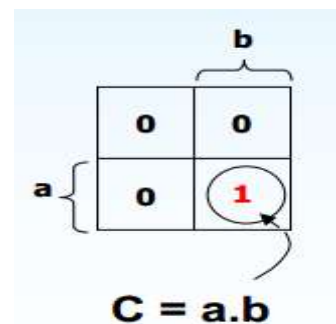


8. Full-subtractor using NAND gates only

**DESIGN:**

1) Half adder

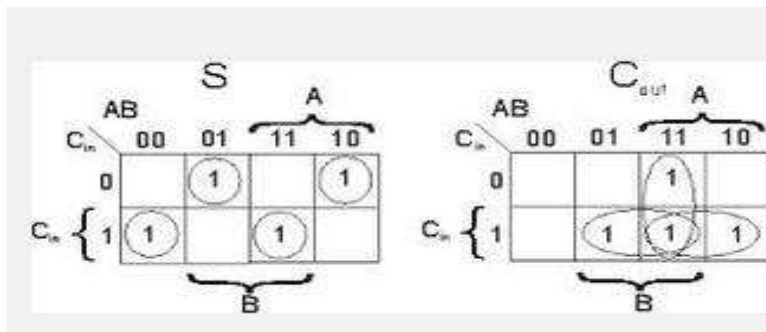
Inputs		Outputs	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Using basic gatesUsing NAND Logic

$$\begin{aligned}
 S &= A\bar{B} + \bar{A}B = A\bar{B} + A\bar{A} + \bar{A}B + B\bar{B} \\
 &= A(\bar{A} + \bar{B}) + B(\bar{A} + \bar{B}) \\
 &= A \cdot \overline{AB} + B \cdot \overline{AB} \\
 &= \overline{A \cdot AB \cdot B \cdot AB} \\
 C &= AB = \overline{AB}
 \end{aligned}$$

2) Full adder

Inputs			Sum	Carry
A	B	C _{in}	S	C _{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Using basic gates

$$\begin{aligned}
 S &= \bar{A}\bar{B}C_{in} + \bar{A}B\bar{C}_{in} + A\bar{B}\bar{C}_{in} + ABC_{in} \\
 &= (\bar{A}\bar{B} + \bar{A}B)\bar{C}_{in} + (A\bar{B} + \bar{A}B)C_{in} = (A \oplus B)\bar{C}_{in} + (\overline{A \oplus B})C_{in} = A \oplus B \oplus C_{in}
 \end{aligned}$$

$$C_{out} = \bar{A}BC_{in} + A\bar{B}C_{in} + AB\bar{C}_{in} + ABC_{in} = AB + (A \oplus B)C_{in} = AB + AC_{in} + BC_{in}$$

Using NAND Logic

$$S = A \oplus B \oplus C_{in} = \overline{(A \oplus B) \cdot (A \oplus B)C_{in} \cdot C_{in} \cdot (A \oplus B)C_{in}}$$

$$C_{out} = C_{in}(A \oplus B) + AB = \overline{C_{in}(A \oplus B) \cdot AB}$$

3) Half –Subtractor

Truth table:

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

K-Map for DIFFERENCE:

		B	
		A	
A	B	00	01
	00		1
	01	1	

$$\text{Difference} = a'b + ab'$$

K-map for borrow:

		B	
		A	
A	B	00	01
	00		1
	01		

$$\text{Borrow} = a'b$$

USING NAND ONLY:

$$\text{Difference } D = A \oplus B = \overline{A \cdot \overline{AB}} \cdot \overline{B \cdot \overline{AB}}$$

$$\text{Borrow } B = \overline{A \cdot B} = \overline{B(\overline{A+B})} = \overline{B(\overline{AB})} = \overline{B \cdot \overline{AB}}$$

4)Full-Subtractor**TRUTH TABLE:**

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

K-Map for Difference:

A \ BC				
	00	01	11	10
0		1		1
1	1		1	

$$\text{Difference} = A'B'C + A'BC' + AB'C' + ABC$$

K-Map for Borrow:

A \ BC				
	00	01	11	10
0		1	1	1
1			1	

$$\text{Borrow} = A'B + BC + A'C$$

USING NAND ONLY:

$$\text{Difference } D = A \oplus B \oplus C = \overline{\overline{A \oplus B} \oplus C} = \overline{(A \oplus B)(A \oplus B)C} \cdot C \cdot \overline{(A \oplus B)C}$$

$$\text{Borrow } B = \overline{AB} \cdot C \cdot \overline{(A \oplus B)} = \overline{AB} + C \cdot \overline{(A \oplus B)} = \overline{B(A+B)} \cdot C \cdot \overline{(C + A \oplus B)}$$

$$= \overline{B \cdot \overline{A} \cdot C} \{ \overline{C \cdot (A \oplus B)} \}$$

EXPERIMENT:

Procedure

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Connect the inputs to the input switches provided in the IC Trainer Kit.
- 4) Connect the outputs to the switches of output LEDs
- 5) Apply various combinations of inputs according to the truth table and observe condition of LEDs

OBSERVATIONS:

Truthtable

Half-Adder

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Full Adder

A	B	C_{IN}	S	C_{OUT}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Half –Subtractor

A	B	Borrow	Difference
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

Full-Subtractor

A	B	C	Borrow	Difference
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

CONCLUSION

Arithmetic circuits are designed and implemented

EXPT NO:4

DATE: __/__/__

4 BIT ADDER/SUBTRACTOR AND BCD ADDER USING 7483**OBJECTIVE:**

To design a 4 bit adder/subtractor and BCD Adder using 7483 .

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	7483	2
		7408	1
		7432	1
		7486	1

INTRODUCTION:**4 bit binary adder/subtractor:**

A binary adder is a digital circuit that produces the arithmetic sum of two binary numbers. It can be constructed with full adders connected in cascade, with the output carry from each full adder connected to the input carry of next full adder in chain. The augends bits of 'A' and the addend bits of 'B' are designated by subscript numbers from right to left, with subscript 0 denoting the least significant bits. The carries are connected in chain through the full adder. The input carry to the adder is C_0 and it ripples through the full adder to the output carry C_4 .

The circuit for subtracting A-B consists of an adder with inverters, placed between each data input 'B' and the corresponding input of full adder. The input carry C_0 must be equal to 1 when performing subtraction.

The addition and subtraction operation can be combined into one circuit with one common binary adder. The mode input M controls the operation. When $M=0$, the circuit is adder circuit. When $M=1$, it becomes subtractor.

4 bit BCD adder:

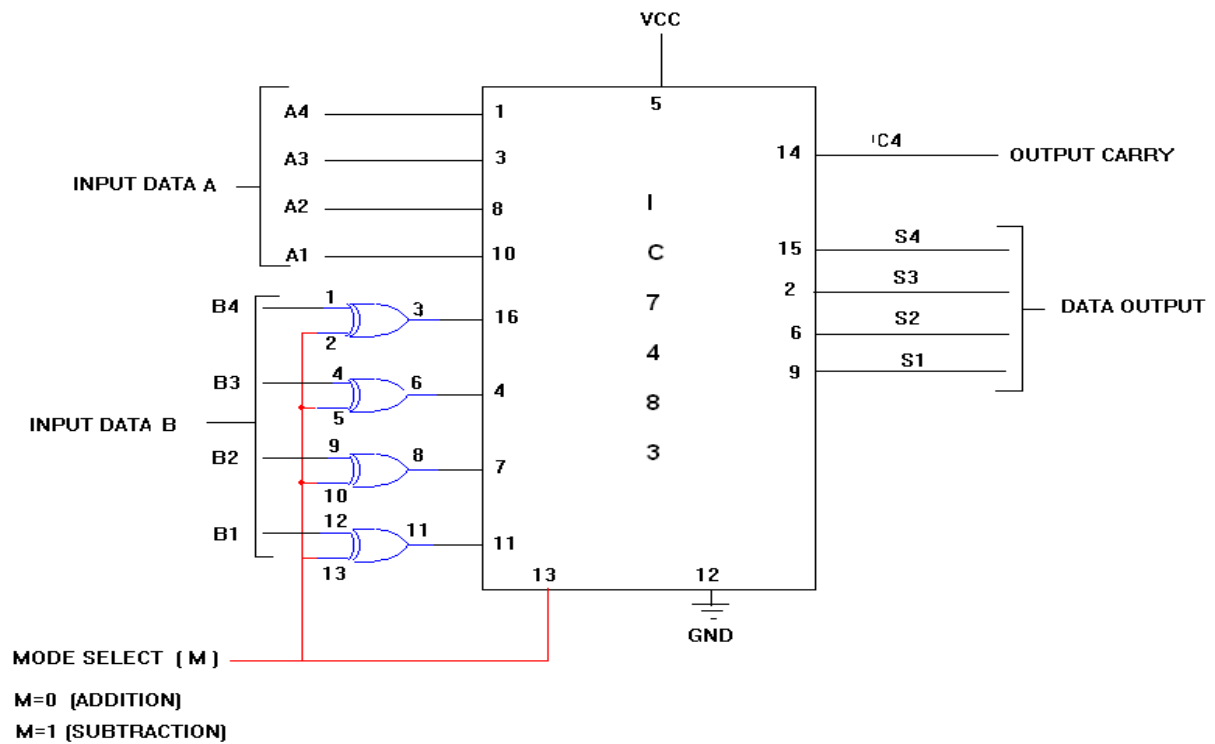
Consider the arithmetic addition of two decimal digits in BCD, together with an input carry from a previous stage. Since each input digit does not exceed 9, the output sum cannot be greater than 19, the 1 in the sum being an input carry. The output of two decimal digits must be represented in BCD and should appear in the form listed in the columns.

ABCD adder that adds 2 BCD digits and produce a sum digit in BCD. The 2 decimal digits, together with the input carry, are first added in the top 4 bit adder to produce the binary sum.

Pin diagram for IC 7483:

1	A4		B4	16
2	S3	I	S4	15
3	A3	C	C4	14
4	B3	7	C1	13
5	VCC	4	GND	12
6	S2	8	B1	11
7	B2	3	A1	10
8	A2		S1	9

CIRCUIT DIAGRAM AND OBSERVATIONS:**4-BIT BINARY ADDER/SUBTRACTOR**



Truth table:

S1 S2		S3 S4			
		00	01	11	10
00	0	0	0	0	0
01	0	0	0	0	0
11	1	1	1	1	1
10	0	0	1	1	1

$$Y = C4 + S4 (S3 + S2)$$

Truth table:

BCD SUM				CARRY
S4	S3	S2	S1	C
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

EXPERIMENT:

Procedure

- 1) Connections were given as per circuit diagram.
- 2) Logical inputs were given as per truth table
- 3) Observe the logical output and verify with the truth tables.

CONCLUSION

The 4 bit adder/subtractor and BCD adder are designed and implemented using 7483.

EXPT NO:5

DATE: __/__/__

2/3 BIT BINARY COMPARATOR**OBJECTIVE:**

To design a 2/3 bit binary comparator

.

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	IC 7408	2
		IC 7486	1
		IC 7432	1
		IC 7404	1
		IC 7485	2

INTRODUCTION:

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

$$A = A_3 A_2 A_1 A_0$$

$$B = B_3 B_2 B_1 B_0$$

The equality of the two numbers and B is displayed in a combinational circuit designated by the symbol ($A=B$).

This indicates A greater than B, then inspect the relative magnitude of pairs of significant digits starting from most significant position. A is 0 and that of B is 0.

We have $A < B$, the sequential comparison can be expanded as

$$A > B = A_3 B_3^1 + X_3 A_2 B_2^1 + X_3 X_2 A_1 B_1^1 + X_3 X_2 X_1 A_0 B_0^1$$

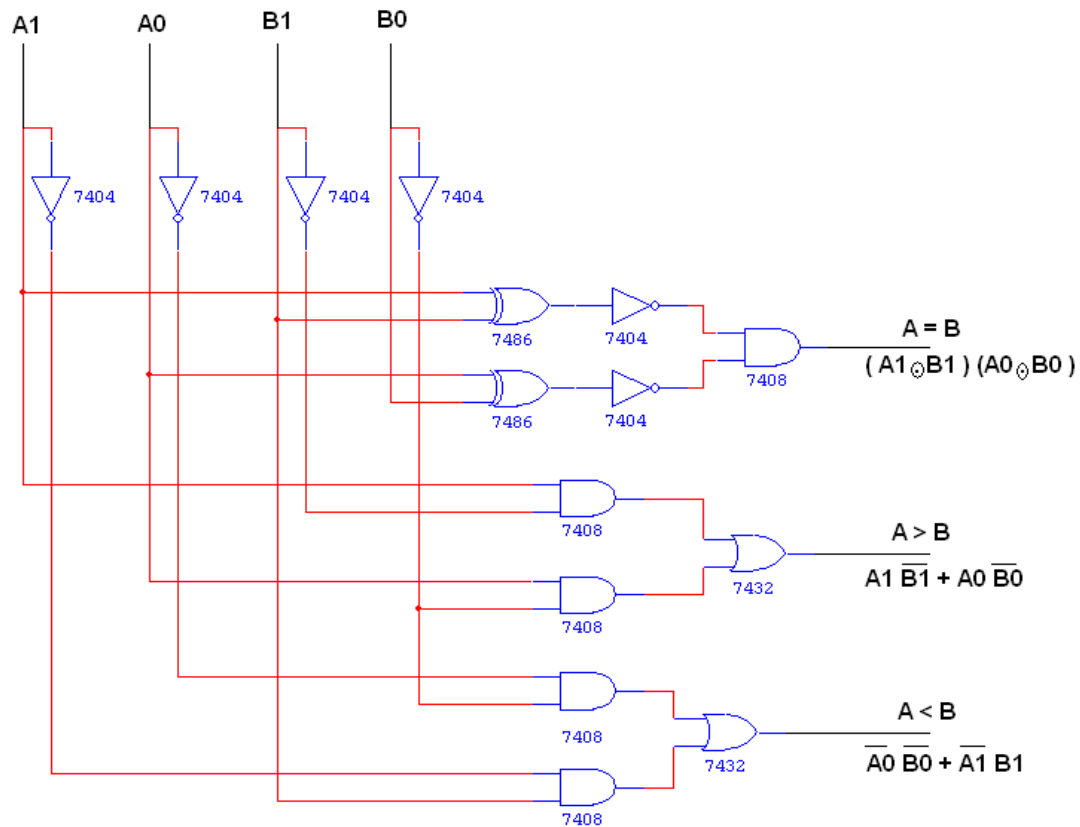
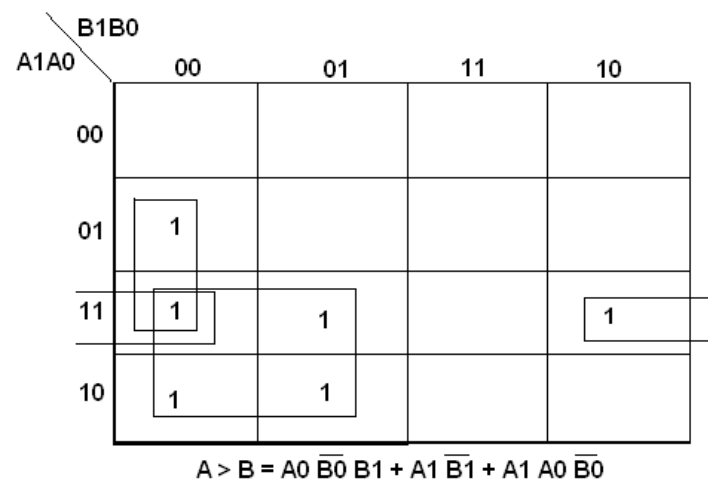
$$A < B = A_3^1 B_3 + X_3 A_2^1 B_2 + X_3 X_2 A_1^1 B_1 + X_3 X_2 X_1 A_0^1 B_0$$

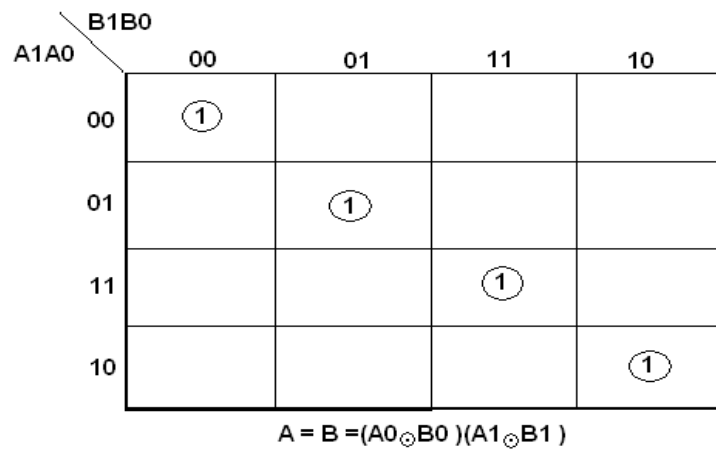
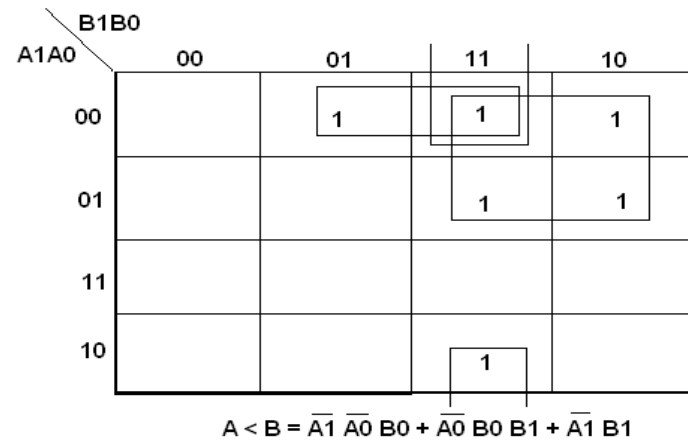
The same circuit can be used to compare the relative magnitude of two BCD digits.

Where, $A = B$ is expanded as,

$$A = B = (A_3 + B_3) (A_2 + B_2) (A_1 + B_1) (A_0 + B_0)$$

$$\begin{array}{cccc} \downarrow & \downarrow & \downarrow & \downarrow \\ X_3 & X_2 & X_1 & X_0 \end{array}$$

CIRCUIT DIAGRAM AND OBSERVATIONS:**2 BIT MAGNITUDE COMPARATOR****K MAP**



TRUTH TABLE

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

EXPERIMENT:**Procedure**

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Test all the components and IC packages using multimeter or digital IC tester. Set up the circuits and verify their truth table.

CONCLUSION

Designed and implemented a 2/3 bit binary comparator.

EXPT NO:6

DATE: __/__/__

BINARY TO GRAY AND GRAY TO BINARY CONVERTERS**OBJECTIVE:**

To design the following circuits

(a) binary to gray converter

(b) gray to binary converter

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	IC 7486	1
		IC 7408	1
		IC 7432	1
		IC 7404	1

INTRODUCTION:

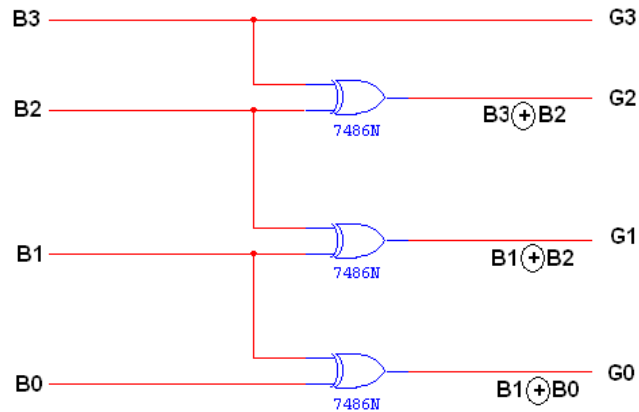
The availability of large variety of codes for the same discrete elements of information results in the use of different codes by different systems. A conversion circuit must be inserted between the two systems if each uses different codes for same information. Thus, code converter is a circuit that makes the two systems compatible even though each uses different binary code.

The bit combination assigned to binary code to gray code. Since each code uses four bits to represent a decimal digit. There are four inputs and four outputs. Gray code is a non-weighted code.

The input variable are designated as B3, B2, B1, B0 and the output variables are designated as C3, C2, C1, C0. from the truth table, combinational circuit is designed. The Boolean functions are obtained from K-Map for each output variable.

A code converter is a circuit that makes the two systems compatible even though each uses a different binary code. To convert from binary code to Excess-3 code, the input lines must supply the bit combination of elements as specified by code and the output lines generate the corresponding bit combination of code. Each one of the four maps represents one of the four outputs of the circuit as a function of the four input variables.

A two-level logic diagram may be obtained directly from the Boolean expressions derived by the maps. These are various other possibilities for a logic diagram that implements this circuit. Now the OR gate whose output is $C+D$ has been used to implement partially each of three outputs.

CIRCUIT DIAGRAM AND OBSERVATIONS:**BINARY TO GRAY CODE CONVERTOR****K-Map for G_3 :**

		B_1B_0			
		00	01	11	10
B_3B_2	00				
	01				
	11	1	1	1	1
	10	1	1	1	1

$$G_3 = B_3$$

K-Map for G_2 :

		B_1B_0			
		00	01	11	10
B_3B_2	00				
	01	1	1	1	1
	11				
	10	1	1	1	1

$$G_2 = B_3 \oplus B_2$$

K-Map for G_1 :

		B1B0			
		00	01	11	10
B3B2	00			1	1
	01	1	1		
	11	1	1		
	10			1	1

$$G_1 = B_1 \oplus B_2$$

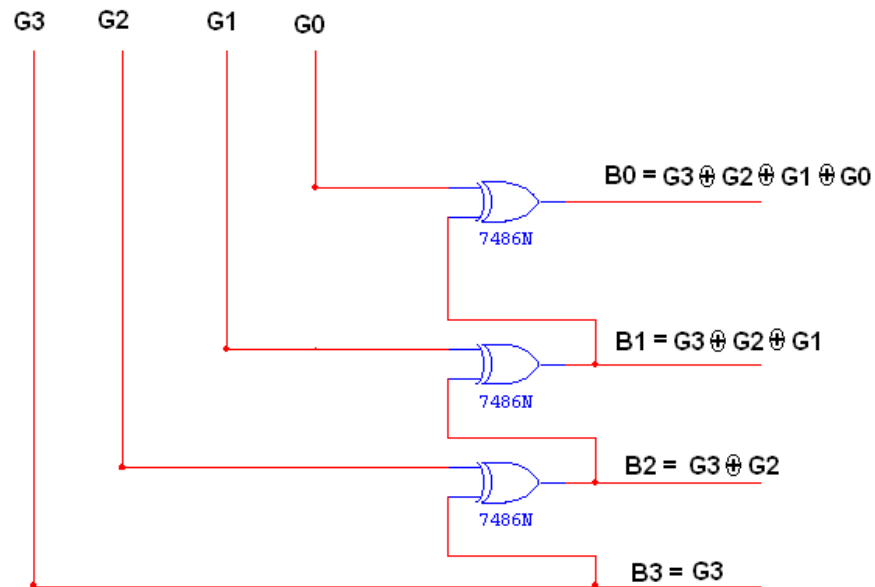
K-Map for G_0 :

		B1B0			
		00	01	11	10
B3B2	00		1		1
	01		1		1
	11		1		1
	10		1		1

$$G_0 = B_1 \oplus B_0$$

TRUTH TABLE:

Binary input				Gray code output			
B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

GRAY CODE TO BINARY CONVERTOR**K-Map for B_3 :**

		G_1G_0			
		00	01	11	10
G_3G_2	00	0	0	0	0
	01	0	0	0	0
	11	1	1	1	1
	10	1	1	1	1

$$B_3 = G_3$$

K-Map for B_2 :

		G_1G_0			
		00	01	11	10
G_3G_2	00	0	0	0	0
	01	1	1	1	1
	11	0	0	0	0
	10	1	1	1	1

$$B_2 = G_3 \oplus G_2$$

K-Map for B₁:

		G1G0			
		00	01	11	10
G3G2	00	0	0	1	1
	01	1	1	0	0
	11	0	0	1	1
	10	1	1	0	0

$$B_1 = G_3 \oplus G_2 \oplus G_1$$

K-Map for B₀:

		G1G0			
		00	01	11	10
G3G2	00	0	①	0	①
	01	①	0	①	0
	11	0	①	0	①
	10	①	0	①	0

$$B_0 = G_3 \oplus G_2 \oplus G_1 \oplus G_0$$

TRUTH TABLE:

Gray Code				Binary Code			
G3	G2	G1	G0	B3	B2	B1	B0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	1	0	0	1	0
0	0	1	0	0	0	1	1
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
0	1	0	1	0	1	1	0
0	1	0	0	0	1	1	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	0	1	0
1	1	1	0	1	0	1	1
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	0
1	0	0	0	1	1	1	1

EXPERIMENT:**Procedure**

- 1) Connections were given as per circuit diagram.
- 2) Logical inputs were given as per truth table
- 3) Observe the logical output and verify with the truth tables

CONCLUSION

Designed and implemented a binary to gray and gray to binary converter.

. EXPT NO:7

DATE: __/__/__

FLIP FLOPS**OBJECTIVE:**

To implement various flip flops using NAND gates and to familiarize the ICs 7474 and 7476

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	7400	2
		7410	2
		7404	1
		7474	1
		7476	1

INTRODUCTION:

Flip flops are the basic building blocks in any memory system, since its output will remain in its state until it is forced to change it by some means.

CLOCKED SR FLIP FLOP

S and R stands for set and reset. There are 4 input combinations possible. But $S=R=1$ is forbidden, since the output will be invalid. When the flip flop is switched on, its output state will be uncertain. When an initial state is to be assigned, two separate inputs called preset and clear are used. They are active low inputs.

JK FLIP FLOPS

The invalid output state of S-R flip flop, when $S=R=1$ is avoided by converting it into a J-K flip flop.

MASTER SLAVE JK FLIP FLOPS

The race around condition of JK flip flop is rectified in Master Slave JK flip flop. Racing is the toggling of the output more than once during a positive clock edge. Master Slave JK flip flop is created by cascading two JK flip flops. The clock fed to the first stage (master) is inverted and fed to the second stage (slave). This ensures that the slave follows the master eliminates the chance of racing.

D FLIP FLOP

It has only one input referred to as D input or data input. The input data is transferred to the output after a clock pulse applied. D flip flop can be derived JK flip flops by using J inputs as D input and J is inverted and fed to K input.

T FLIP FLOP

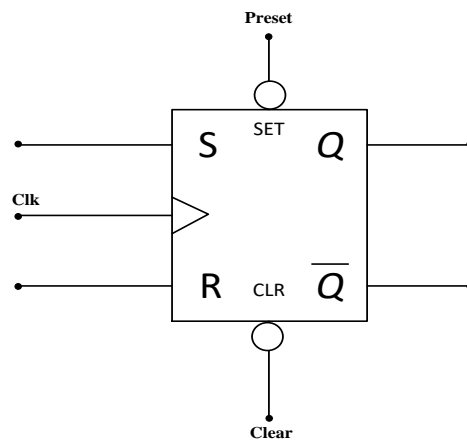
T stands for toggle. The output toggles when a clock pulse is applied. That is the output of the flip flop changes state for an input pulse. T flip flop can be derived from JK flip flop by shorting J and K inputs.

FLIP FLOP ICs

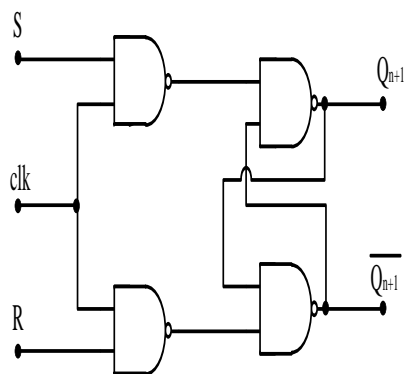
IC 7476 is dual negative edge triggered MS JK flip flop with preset and clear facility. It has a 16 pin DIP chip. IC 7473 is a dual negative edge triggering Master Slave JK flip flop with clear in 14 pin DIP. It does not have preset input. IC 7474 is positive edge triggered dual D flip flop with preset and clear in 14 pin DIP.

CIRCUIT DIAGRAM :**1) SR FLIPFLOP**

LOGIC SYMBOL

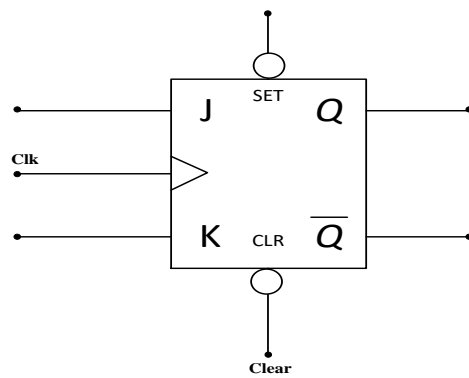


SR FLIP FLOP USING GATES

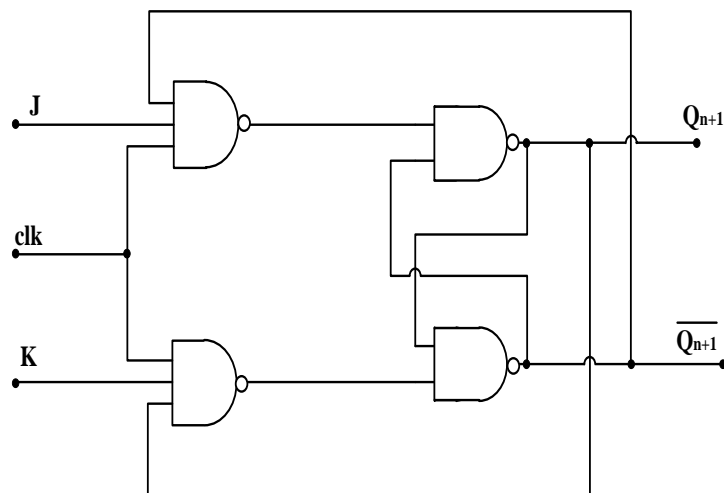


2) JK FLIP FLOP

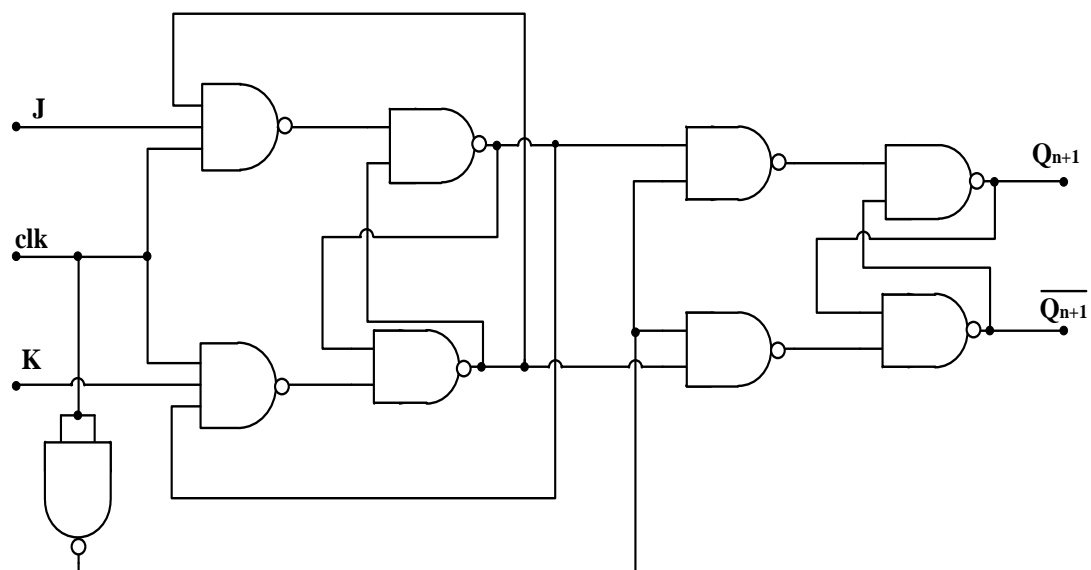
LOGIC SYMBOL



JK FLIP FLOP USING GATES

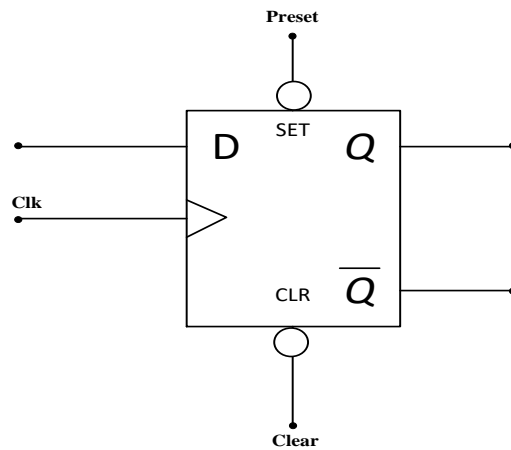


3) MASTER SLAVE JK FLIP FLOP NAND GATES

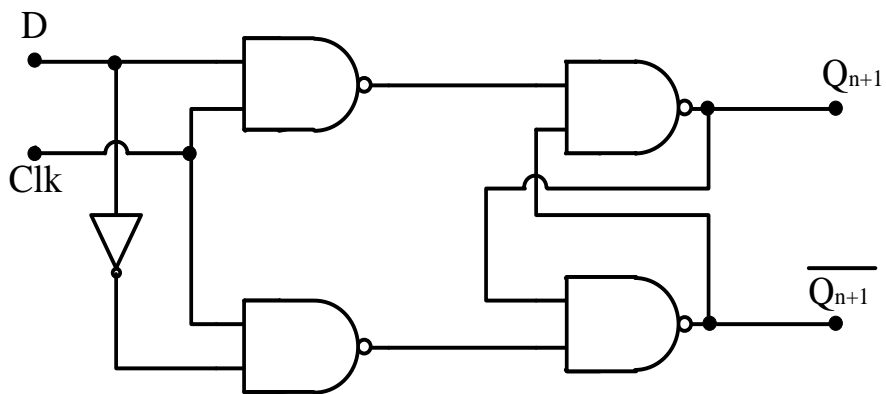


4) D FLIPFLOP

LOGIC SYMBOL

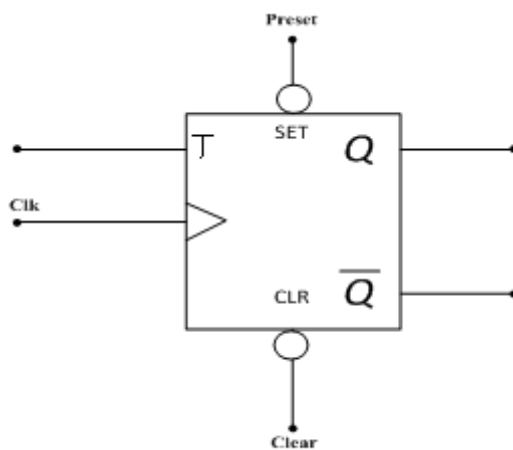


D FLIP FLOP USING GATES

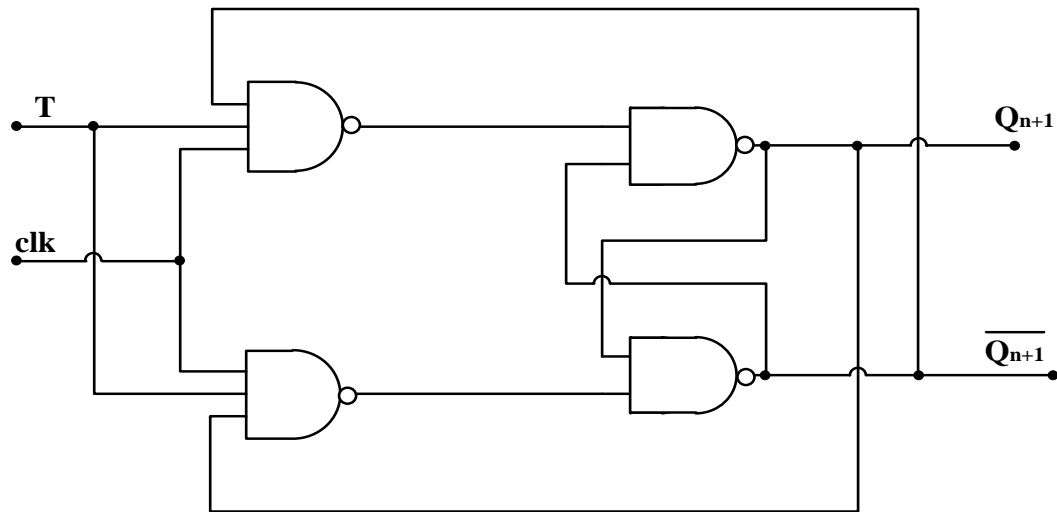


5) T FLIP FLOP

LOGIC SYMBOL

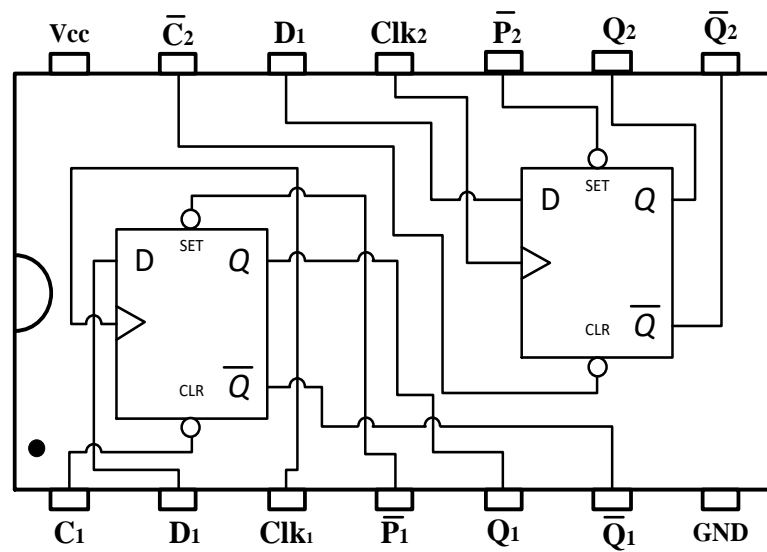


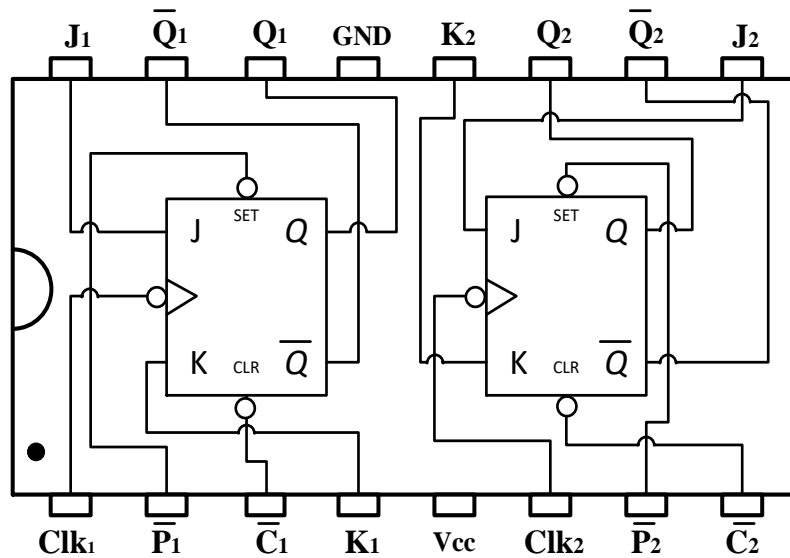
T FLIP FLOP USING GATES



FLIP FLOP ICS

D FLIP FLOP IC- 7474



JK FLIPFLOP IC-7476**EXPERIMENT:****Procedure**

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Test all the components and IC packages using multimeter or digital IC tester. Set up the flip flops using gates and verify their truth table.
- 4) Verify the truth tables of 7474, 7476

OBSERVATIONS:**Truth table****1) SR flipflop**

Inputs		
S	R	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	?

2) JK flipflop

Inputs			Outputs		
J	K	C	Q	Q'	Comments
0	0	↑	Q	Q'	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	↑	Q'	Q	Toggle

3) Master slave JK flip-flop

J	K	$Q_{(t+1)}$
0	0	$Q_{(t)}$ unchanged
0	1	0 reset
1	0	1 set
1	1	$\bar{Q}_{(t)}$ output inversion

4) D flip flop

Input	Output
D_n	Q_{n+1}
0	0
1	1

5) T flip flop

T	Q	\bar{Q}
0	Q	\bar{Q}
1	\bar{Q}	Q
0	\bar{Q}	Q
1	Q	\bar{Q}

CONCLUSION

The flip-flops are implemented using NAND gates and outputs are verified
EXPT NO: 8

DATE: __/__/__

ASYNCHRONOUS COUNTERS

OBJECTIVE:

To realize the following asynchronous counters

- (a) 4-bit up counter\
- (b) 4 bit down counter
- (c) Mod-10 counters
- (d) 3 bit up/down counter

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	7476,	1
		7400	1
		7473	1
		7486	1

INTRODUCTION:

Asynchronous counter

The term asynchronous refers to events that do not occur at the same time. With respect to counter operation, asynchronous means that the flip-flops within the counter are not connected in a way to cause all flip-flops at exactly the same time. They are wired in a way that links the clock of the next flip-flop to the Q of the current device. This causes the output count states to ripple through the counter. A counter is a circuit that produces a set of unique output combinations corresponding to the number of applied input pulses. The number of unique outputs of a counter is known as its modulus or mod number. Each flip flop is triggered by the output from the previous flip flop except for the first flip flop (LSB) which receives an external clock. Asynchronous counters are commonly referred to as ripple counters because the effect of the input clock pulse is first felt by FF0. This effect cannot get to FF1 immediately because of the propagation delay through FF0.

4 bit asynchronous up-counter

An asynchronous 4 bit binary up counter, a circuit made up of 4 J-K flip-flop cascaded to generate four bits counting sequence. An up counter is basically a digital counting circuit which counts up in an incremental mode.

4 bit asynchronous down-counter.

An asynchronous 4 bit binary up counter, a circuit made up of 4 J-K flip-flop cascaded to generate four bits counting sequence. A down counter is basically a digital counting circuit which counts down in a decremental mode.

Asynchronous decade counter

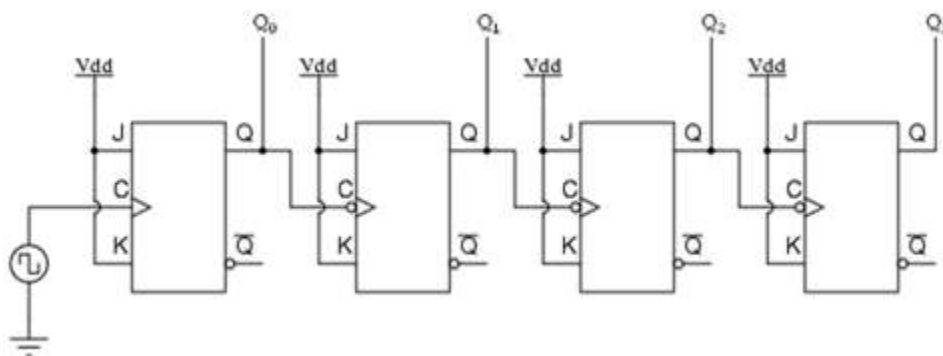
The modulus for counters for counters is the number of unique states through which the counter will sequence. The maximum possible number of states of a counter is 2^n , where n is the number of flip flops in the counter. A decade counter with a count sequence of zero (0000) through nine (1001) is a BCD decade counter because its ten-state sequence produces the BCD code. This type of counter is useful in display applications in which BCD is required for conversion to a decimal readout.

3-bit up/down counter

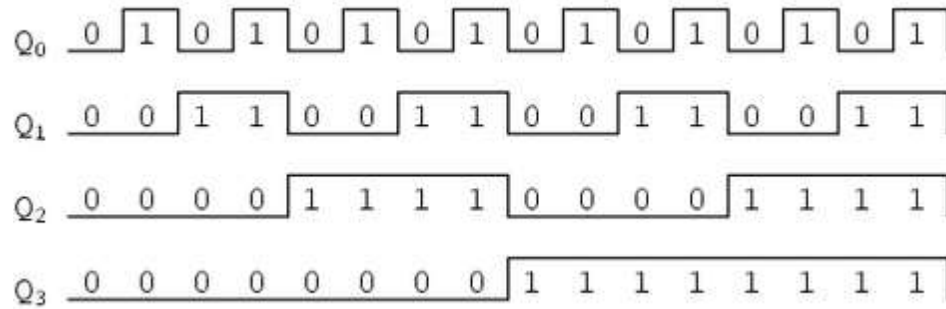
The direction of counting is decided by a mode control input M . An XOR gate between flip flop functioning as a controlled inverter connects either of Q or Q bar to the clock input of the succeeding flip flop as decided by the logic state at M . When mode control is 0, Q outputs get connected to the clock inputs of the succeeding flip flop and counter counts up. When mode control is 1, Q bar outputs are connected to the clock inputs and counter counts down.

CIRCUIT DIAGRAM AND DESIGN:

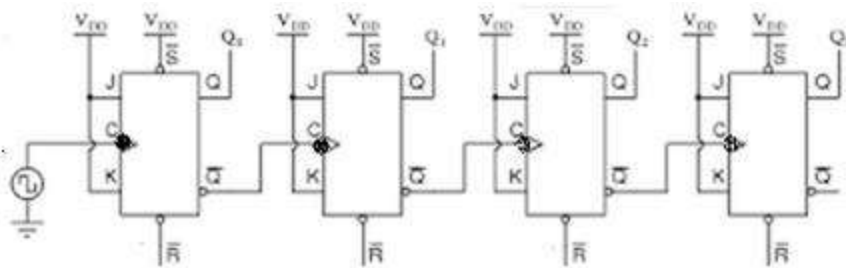
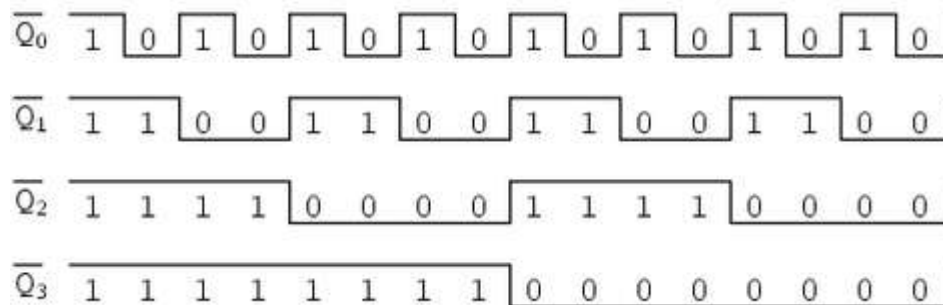
1) 4 bit asynchronous up-counter



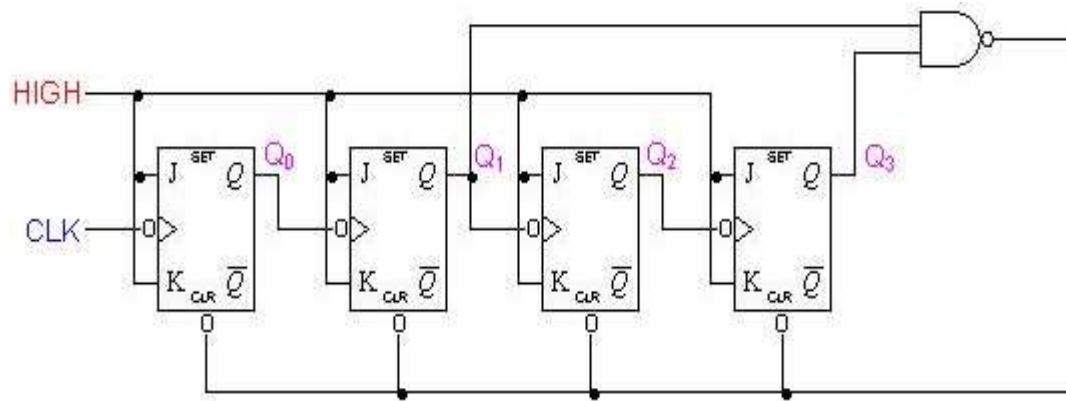
Timing diagram

"Up" count sequence

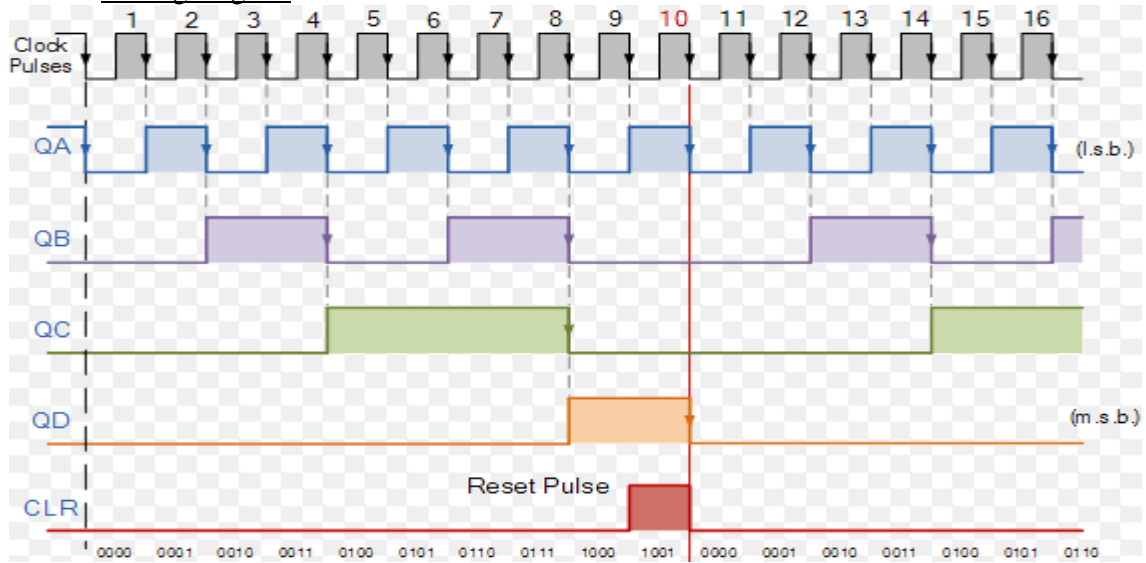
2) 4 bit asynchronous down counter

Timing diagram*"Down" count sequence*

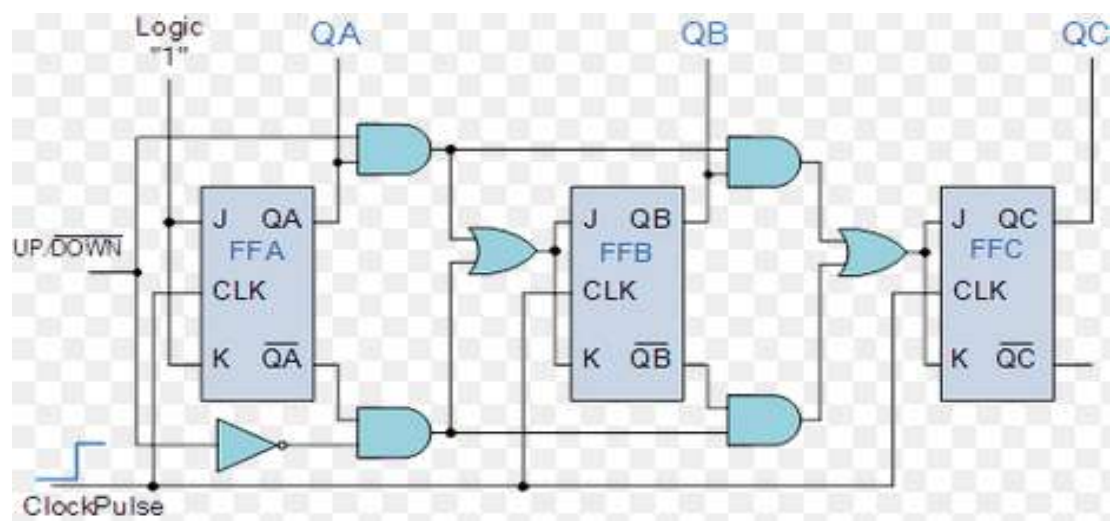
3) Decade counter



Timing diagram



4)3-bit up/down counters using mode control



Timing diagram



EXPERIMENT:

Procedure

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Connect the inputs to the input switches provided in the IC Trainer Kit.
- 4) Connect the outputs to the switches of output LEDs
- 5) Apply various combinations of inputs according to the truth table and observe condition of LEDs

OBSERVATIONS:

Truth table**1) 4 bit asynchronous up-counter**

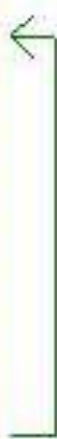
Clock	Q3	Q2	Q1	Q0
1	0	0	0	0
2	0	0	0	1
3	0	0	1	0
4	0	0	1	1
5	0	1	0	0
6	0	1	0	1
7	0	1	1	0
8	0	1	1	1
9	1	0	0	0
10	1	0	0	1
11	1	0	1	0
12	1	0	1	1
13	1	1	0	0
14	1	1	0	1
15	1	1	1	0
16	1	1	1	1

3) 4 bit asynchronous down-counter

CK	Q₃	Q₂	Q₁	Q₀
0	1	1	1	1
1	1	1	1	0
2	1	1	0	1
3	1	1	0	0
4	1	0	1	1
5	1	0	1	0
6	1	0	0	1
7	1	0	0	0
8	0	1	1	1
9	0	1	1	0
10	0	1	0	1
11	0	1	0	0
12	0	0	1	1
13	0	0	1	0
14	0	0	0	1
15	0	0	0	0

3) Decade counter

Clock Pulse	Q3	Q2	Q1	Q0
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



4) 3-bit up/down counter using mode control

CLK	M	Q ₂	Q ₁	Q ₀
0	1	0	0	0
1	1	1	1	1
2	1	1	1	0
3	1	1	0	1
4	1	1	0	0
5	1	0	1	1
6	1	0	1	0
7	1	0	0	1
8	1	0	0	0
9	0	0	0	1
10	0	0	1	0
11	0	0	1	1
12	0	1	0	0
13	0	1	0	1
14	0	1	1	0
15	0	1	1	1

CONCLUSION

Asynchronous counters are designed and outputs are obtained.

EXPI NO: 9

DATE: __/__/__

SYNCHRONOUS COUNTERS**OBJECTIVE:**

Realization of following synchronous counters

- (1) 4 bit Synchronous Counter
- (2) Synchronous Mod-N counters
- (3) 3 bit synchronous up-down counter

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	7476	1
		7400	1
		7408	1

INTRODUCTION:**Synchronous counter**

A synchronous **counter**, in contrast to an asynchronous **counter**, is one whose output bits change state simultaneously, with no ripple. The only way we can build such a **counter** circuit from J-K flip-flops is to connect all the clock inputs together, so that each and every flip-flop receives the exact same clock pulse at the exact same.

Mod -N Counter

A mod-n counter has n possible states .that means it counts from 0 to n and rolls over it. There must be a way to force the counter stops counting at n and roll over to 0. This is where the asynchronous inputs come into play. The asynchronous inputs can override the synchronous inputs and force the output either at high or low.

3 bit synchronous Up-down Counter

A counter is a register capable of counting number of clock pulse arriving at its clock input. Counter represents the number of clock pulses arrived. An up/down counter is one that is capable of progressing in increasing order or decreasing order through a certain sequence. An up/down counter is also called bidirectional counter. Usually up/down operation of the counter is controlled by up/down

signal. When this signal is high counter goes through up sequence and when up/down signal is low counter follows reverse sequence.

DESIGN , CIRCUIT DIAGRAM AND OBSERVATIONS:

1) 4 BIT SYNCHRONOUS UP COUNTER

Truth Table

Clock pulse	Present state		
	Q2	Q1	Q0
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

Present state	Next state	J3	K3	J2	K2	J1	K1	J0	K0
0000	0001	0	X	0	X	0	X	1	X
0001	0010	0	X	0	X	1	X	X	1
0010	0011	0	X	0	X	X	0	1	X
0011	0100	0	X	1	X	X	1	X	1
0100	0101	0	X	X	0	0	X	1	X
0101	0110	0	X	X	0	1	X	X	1
0110	0111	0	X	X	0	X	0	1	X
0111	1000	1	X	X	1	X	1	X	1
1000	1001	X	0	0	X	0	X	1	X
1001	1010	X	0	0	X	1	X	X	1
1010	1011	X	0	0	X	X	0	1	X
1011	1100	X	0	1	X	X	1	X	1
1100	1101	X	0	X	0	0	X	1	X
1101	1110	X	0	X	0	1	X	X	1
1110	1111	X	0	X	0	X	0	1	X
1111	0000	X	1	X	1	X	1	X	1

K Map:*K-maps:*

$Q_1 Q_0$ $Q_3 Q_2$	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	x	x	x	X
10	x	x	x	x

$$J_3 = Q_2 Q_1 Q_0$$

$Q_1 Q_0$ $Q_3 Q_2$	00	01	11	10
00	x	x	x	X
01	x	X	x	x
11	0	0	1	0
10	0	0	0	0

$$K_3 = Q_2 Q_1 Q_0$$

$Q_1 Q_0$ $Q_3 Q_2$	00	01	11	10
00	0	0	1	0
01	X	X	X	X
11	X	X	X	X
10	0	0	1	0

$$J_2 = Q_1 Q_0$$

$Q_1 Q_0$ $Q_3 Q_2$	00	01	11	10
00	x	x	x	X
01	0	0	1	0
11	0	0	1	0
10	x	x	x	X

$$K_2 = Q_1 Q_0$$

$Q_1 Q_0$ $Q_3 Q_2$	00	01	11	10
00	0	1	x	X
01	0	1	x	X
11	0	1	x	x
10	0	1	x	x

$$J_1 = Q_0$$

$Q_1 Q_0$ $Q_3 Q_2$	00	01	11	10
00	x	x	1	0
01	x	x	1	0
11	x	x	1	0
10	x	x	1	0

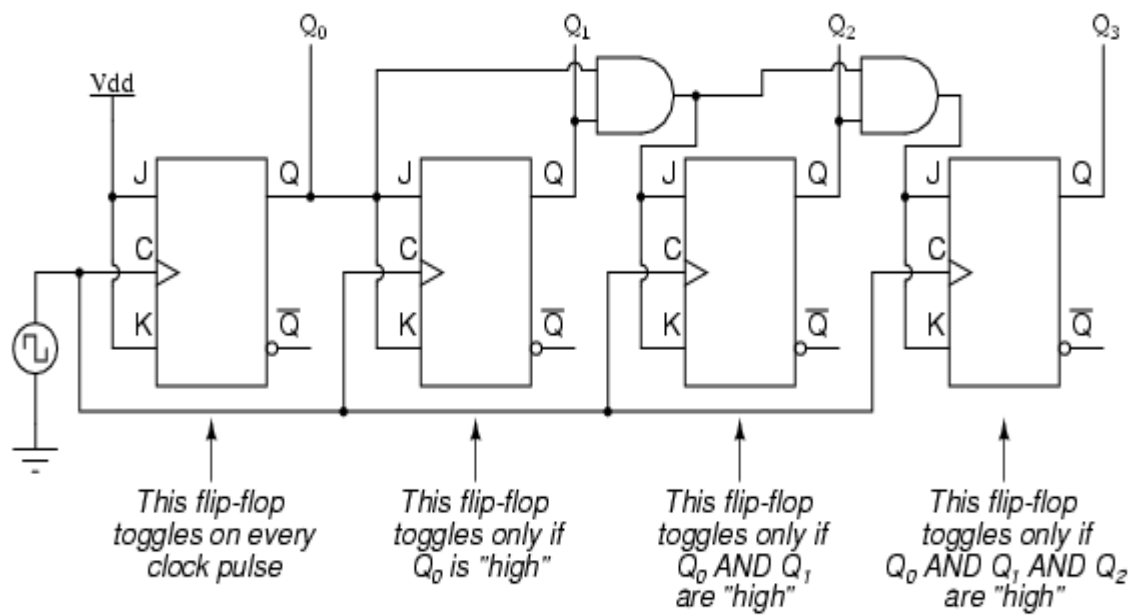
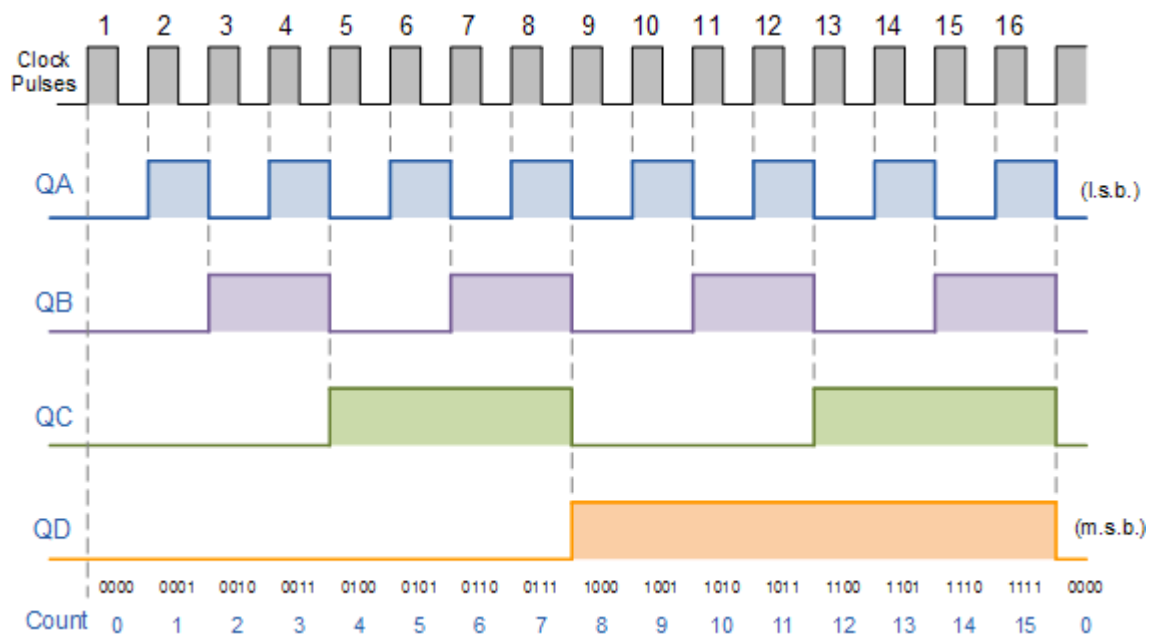
$$K_1 = Q_0$$

$Q_1 Q_0$ $Q_3 Q_2$	00	01	11	10
00	1	1	1	1
01	x	x	x	x
11	x	x	x	X
10	1	1	1	1

$$J_0 = 1$$

$Q_1 Q_0$ $Q_3 Q_2$	00	01	11	10
00	x	x	x	X
01	1	1	1	1
11	1	1	1	1
10	x	x	x	x

$$K_0 = 1$$

Circuit daigram**Timing diagram**

2) 3 BIT SYNCHRONOUS UP/DOWN COUNTER

Truth Table

Clock pulse	Present state			
	M	Q2	Q1	Q0
0	1	0	0	0
1	1	0	0	1
2	1	0	1	0
3	1	0	1	1
4	1	1	0	0
5	1	1	0	1
6	1	1	1	0
7	1	1	1	1
8	0	1	1	1
9	0	1	1	0
10	0	1	0	1
11	0	1	0	0
12	0	0	1	1
13	0	0	1	0
14	0	0	0	1
15	0	0	0	0

Excitation Table

Clock pulse	Present state				Next state			Inputs To Flipflops					
	M	Q2	Q1	Q0	Q2	Q1	Q0	J2	K2	J1	K1	J0	K0
0	0	0	0	0	1	1	1	1	X	1	X	1	X
1	0	0	0	1	0	0	1	0	X	0	X	1	X
2	0	0	1	0	0	0	0	0	X	0	X	X	1
3	0	0	1	1	0	1	0	0	X	1	X	X	1
4	0	1	0	0	0	0	1	0	X	X	1	1	X
5	0	1	0	1	0	1	1	0	X	X	0	1	X
6	0	1	1	0	0	1	0	0	X	X	0	X	1
7	0	1	1	1	1	0	0	1	X	X	1	X	1
8	1	0	0	0	0	1	1	X	1	1	X	1	X
9	1	0	0	1	1	0	1	X	0	0	X	1	X
10	1	0	1	0	1	0	0	X	0	0	X	X	1
11	1	0	1	1	1	1	0	X	0	1	X	X	1
12	1	1	0	0	1	0	1	X	0	X	1	1	X
13	1	1	0	1	1	1	1	X	0	X	0	1	X
14	1	1	1	0	1	1	0	X	0	X	0	X	1
15	1	1	1	1	0	0	0	X	1	X	1	X	1

K MAP:

Q3Q2 \ Q1M

1			
		1	
X	X	X	X
X	X	X	X

$$J3 = Q2'Q1'M' + Q2Q1M$$

Q3Q2 \ Q1M

X	X	X	X
X	X	X	X
		1	
1			

$$K3 = Q2'Q1'M' + Q2Q1M$$

Q3Q2 \ Q1M

1		1	
X	X	X	X
X	X	X	X
1		1	

$$J2 = Q1'M' + Q1M$$

Q3Q2 \ Q1M

X	X	X	X
1		1	
1		1	
X	X	X	X

$$K2 = Q1'M' + Q1M$$

Q3Q2 \ Q1M

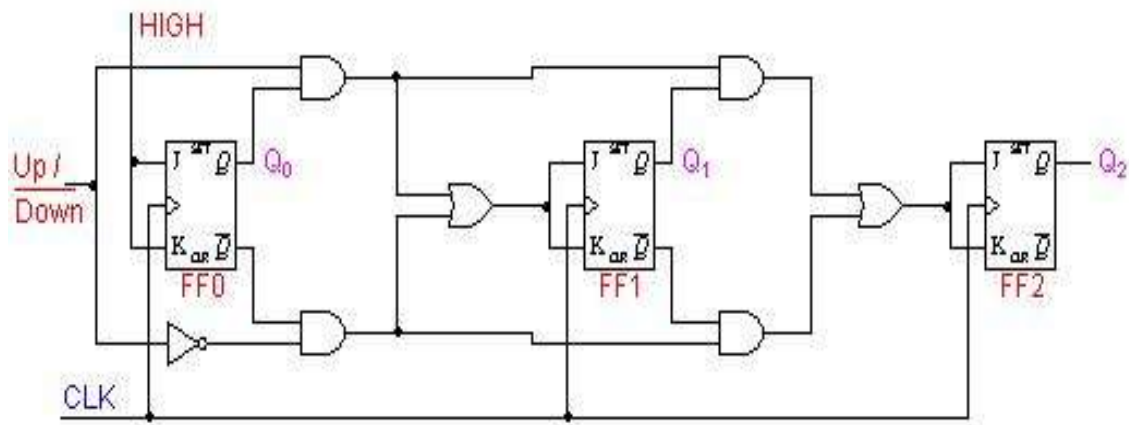
1	1	X	X
1	1	X	X
1	1	X	X
1	1	X	X

$$J1 = 1$$

Q3Q2 \ Q1M

X	X	1	1
X	X	1	1
X	X	1	1
X	X	1	1

K1=1

Circuit daigram

3) Mod 10 synchronous counter

Truth Table

Clock Pulse	Q3	Q2	Q1	Q0
Initially	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
10 (recycles)	0	0	0	0

Excitation table

Present State				Next state				Inputs To Flipflops							
Q3	Q2	Q1	Q0	Q3	Q2	Q1	Q0	J3	K3	J2	K2	J1	K1	J0	K0
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	X	0	0	X	0	X	1	X
1	0	0	1	0	0	0	0	X	1	0	X	0	X	X	1

K MAP:**Q4Q3 \ Q2Q1**

		1	
X	X	X	X
X	X	X	X

J4=Q3Q2Q1**Q4Q3 \ Q2Q1**

X	X	X	X
X	X	X	X
X	X	X	X
	1	X	X

K4=Q1**Q4Q3 \ Q2Q1**

		1	
X	X	X	X
X	X	X	X
		X	X

J3=Q2Q1

Q4Q3 \ Q2Q1

X	X	X	X
		1	
X	X	X	X
		X	X

K3=Q2Q1**Q4Q3 \ Q2Q1**

	1	X	X
	1	X	X
X	X	X	X
		X	X

J2=Q4'Q1**Q3Q2 \ Q1M**

X	X	1	
X	X	1	
X	X	X	X
X	X	X	X

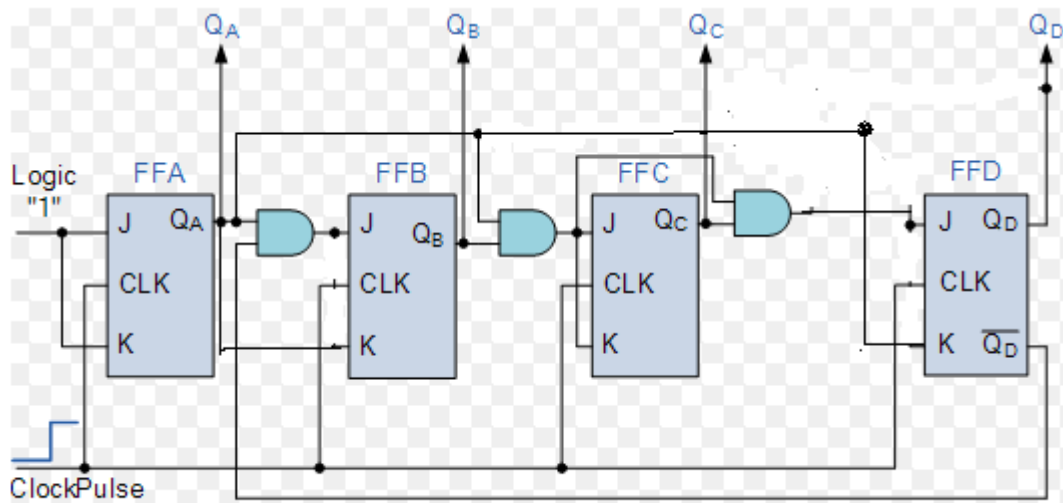
K2=Q1**Q4Q3 \ Q2Q1**

1	X	X	1
1	X	X	1
X	X	X	X
1	X	X	X

J1=1**Q4Q3 \ Q2Q1**

X	1	1	X
X	1	1	X
X	X	X	X
X	1	X	X

K1=1

Circuit daigram**EXPERIMENT:****Procedure**

1. Place the IC on IC Trainer Kit.
2. Connect V_{CC} and ground to respective pins of IC Trainer Kit.
3. Connect the inputs to the input switches provided in the IC Trainer Kit.
4. Connect the outputs to the switches of output LEDs
5. Apply various combinations of inputs according to the truth table and observe condition of LEDs

CONCLUSION

The synchronous counters are designed and implemented.

EXPI NO: 10

DATE: __/__/__

RANDOM SEQUENCE GENERATOR**OBJECTIVE:**

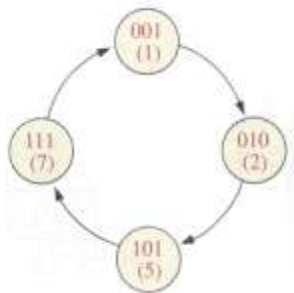
Realization of random sequence generator

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	7476	2

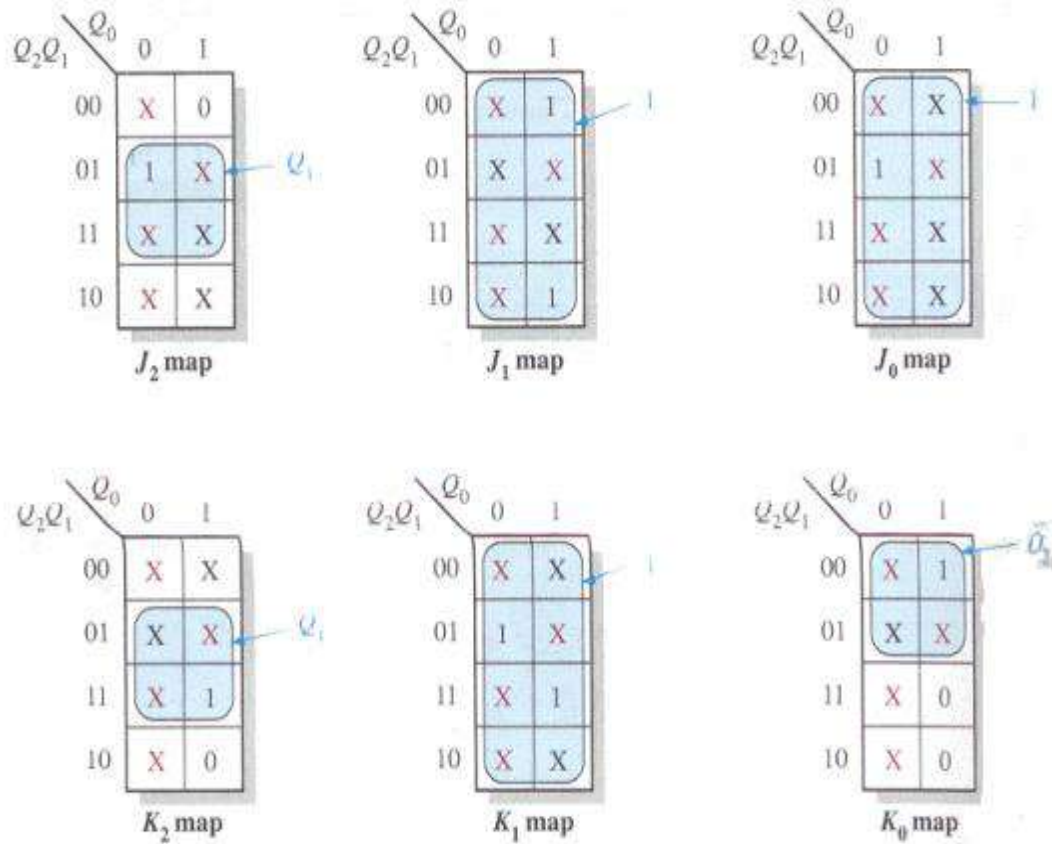
INTRODUCTION:**Random sequence 1,2, 5, 7,.....**

This counter has only four stable states, but it requires three Flipflops, because it counts 101 and 111. Three flips can have 8 states, out of which states 000,011,100,101 are invalid. The entries for excitations corresponding to invalid states are don't cares.

State Diagram**DESIGN and CIRCUIT DIAGRAM :****Excitation Table**

Present State			Next State			Inputs to Flip-flop					
Q2	Q1	Q0	Q2	Q1	Q0	J2	K2	J1	K1	J0	K0
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	1	0	1	1	X	X	1	1	X
1	0	1	1	1	1	X	0	1	X	X	0
1	1	1	0	0	1	X	1	X	1	X	0

Karnaugh Map



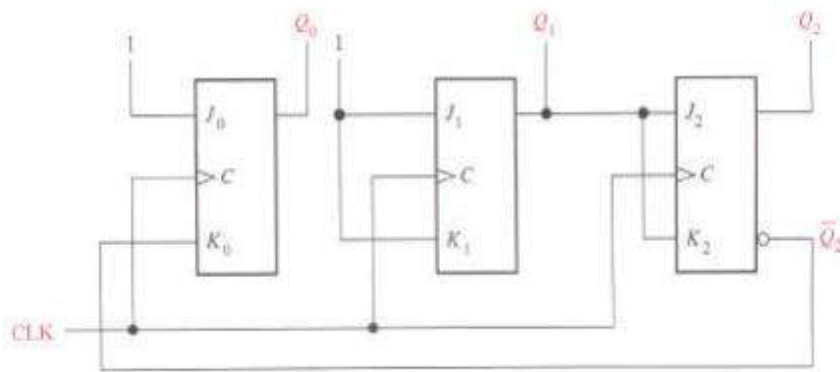
Logic Expressions for Flip- flop Inputs

The expression for each J and K input taken from the maps is as follows:

$$J_0=1, K_0=Q_2$$

$$J_1=K_1=1$$

$$J_2=K_2=Q_1$$

Circuit daigram**EXPERIMENT:****Procedure**

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Connect the inputs to the input switches provided in the IC Trainer Kit.
- 4) Connect the outputs to the switches of output LEDs
- 5) Apply various combinations of inputs according to the truth table and observe condition of LEDs
- 6) Disconnect output from the LEDs and note down the corresponding multimeter voltage readings for various combinations of inputs.

CONCLUSION

The random sequence generator is designed and implemented using JK flipflop.

EXPT NO: 11

DATE: __/__/__

SHIFT REGISTERS**OBJECTIVE:**

To design and Implement the Shift Registers.

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	7474	1
		7400	3

INTRODUCTION:**Shift register**

In digital circuits, a shift register is a cascade of flip flops, sharing the same clock, which has the output of anyone but the last flip-flop connected to the "data" input of the next one in the chain, resulting in a circuit that shifts by one position the one-dimensional "bit array" stored in it, shifting in the data present at its input and shifting out the last bit in the array, when enabled to do so by a transition of the clock input. More generally, a shift register may be multidimensional; such that its "data in" input and stage outputs are themselves bit arrays: this is implemented simply by running several shift registers of the same bit-length in parallel. Shift registers can have both parallel and serial inputs and outputs. These are often configured as serial-in, parallel-out (SIPO) or as parallel-in, serial-out (PISO). There are also types that have both serial and parallel input and types with serial and parallel output.

Serial in serial out shift registers

These are the simplest kind of shift registers. The data string is presented at 'Data In', and is shifted right one stage each time 'Data Advance' is brought high. At each advance, the bit on the far left (i.e. 'Data In') is shifted into the first flip-flop's output. The bit on the far right (i.e. 'Data Out') is shifted out and lost.

Serial in parallel out shift registers

This configuration allows conversion from serial to parallel format. Data is input serially, as described in the SISO section above. Once the data has been input, it may be either read off at each output simultaneously, or it can be shifted out and replaced.

Parallel In Parallel Out (PIPO) shift registers

Parallel in Parallel out (PIPO) shift registers are the type of storage devices in which both data loading as well as data retrieval processes occur in parallel mode

Parallel in serial out shift registers

This configuration has the data input on lines D1 through D4 in parallel format. To write the data to the register, the Write/Shift control line must be held LOW. To shift the data, the W/S control line is brought HIGH and the registers are clocked. The arrangement now acts as a SISO shift register, with D1 as the Data Input. However, as long as the number of clock cycles is not more than the length of the data-string, the Data Output, Q, will be the parallel data read off in order.

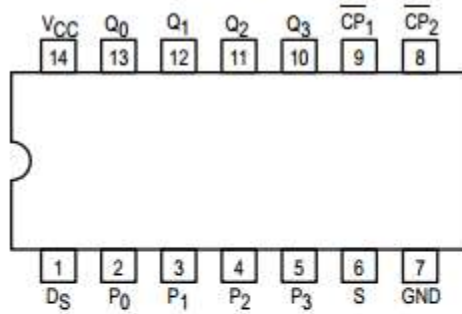
4-Bit Shift Register, 7495

These 4-bit registers feature parallel and serial inputs, parallel outputs, mode control, and two clock inputs. The registers have three modes of operation:

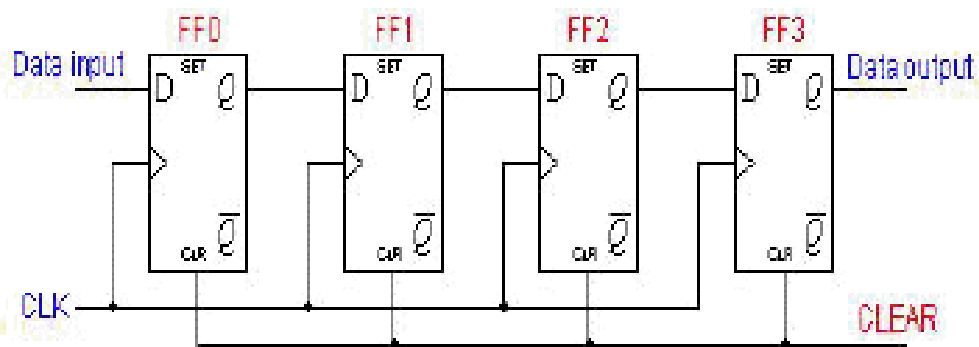
- Parallel (broadside) load
- Shift right (the direction Q_A toward Q_D)
- Shift left (the direction Q_D toward Q_A)

Parallel loading is accomplished by applying the four bits of data and taking the mode control input high. The data is loaded into the associated flip-flops and appears at the outputs after the high-to-low transition of the clock-2 input. During loading, the entry of serial data is inhibited.

Shift right is accomplished on the high-to-low transition of clock 1 when the mode control is low; shift left is accomplished on the high-to-low transition of clock 2 when the mode control is high by connecting the output of each flip-flop to the parallel input of the previous flip-flop (Q_D to input C, etc.) and serial data is entered at input D. The clock input may be applied commonly to clock 1 and clock 2 if both modes can be clocked from the same source. Changes at the mode control input should normally be made while both clock inputs are low; however, conditions described in the last three lines of the function table will also ensure that register contents are protected.

CONNECTION DIAGRAM**PIN NAMES**

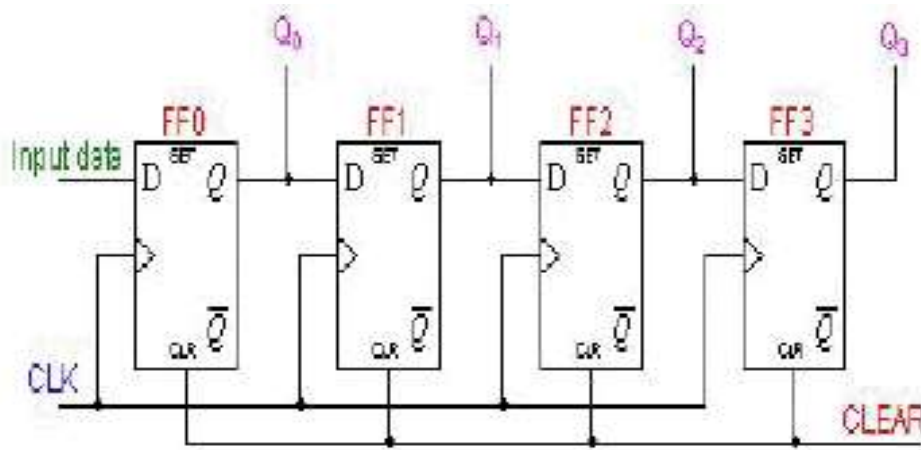
S	Mode Control Input
D _S	Serial Data Input
P ₀ –P ₃	Parallel Data Inputs
CP ₁	Serial Clock (Active LOW Going Edge) Input
CP ₂	Parallel Clock (Active LOW Going Edge) Input
Q ₀ –Q ₃	Parallel Outputs (Note b)

CIRCUIT DIAGRAM AND OBSERVATIONS :**1) Serial in serial out shift register**

Truthtable

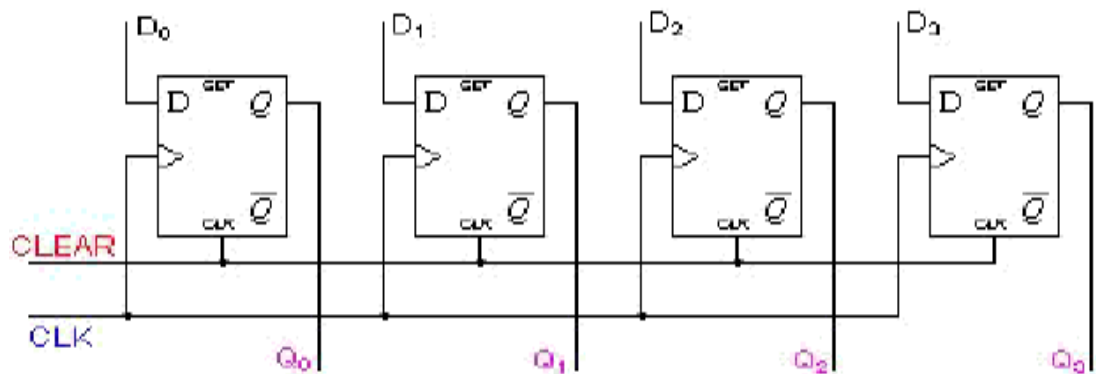
Clock	Serial i/p	QA	QB	QC	QD
1	d ₀ =0	0	X	X	X
2	d ₁ =1	1	0	X	X
3	d ₂ =1	1	1	0	X
4	d ₃ =1	1	1	1	0=d ₀
5	X	X	1	1	1=d ₁
6	X	X	X	1	1=d ₂
7	X	X	X	X	1=d ₃

2) Serial in parallel out shift register

Truthtable

Clock	Serial i/p	QA	QB	QC	QD
1	0	0	X	X	X
2	1	1	0	X	X
3	1	1	1	0	X
4	1	1	1	1	0

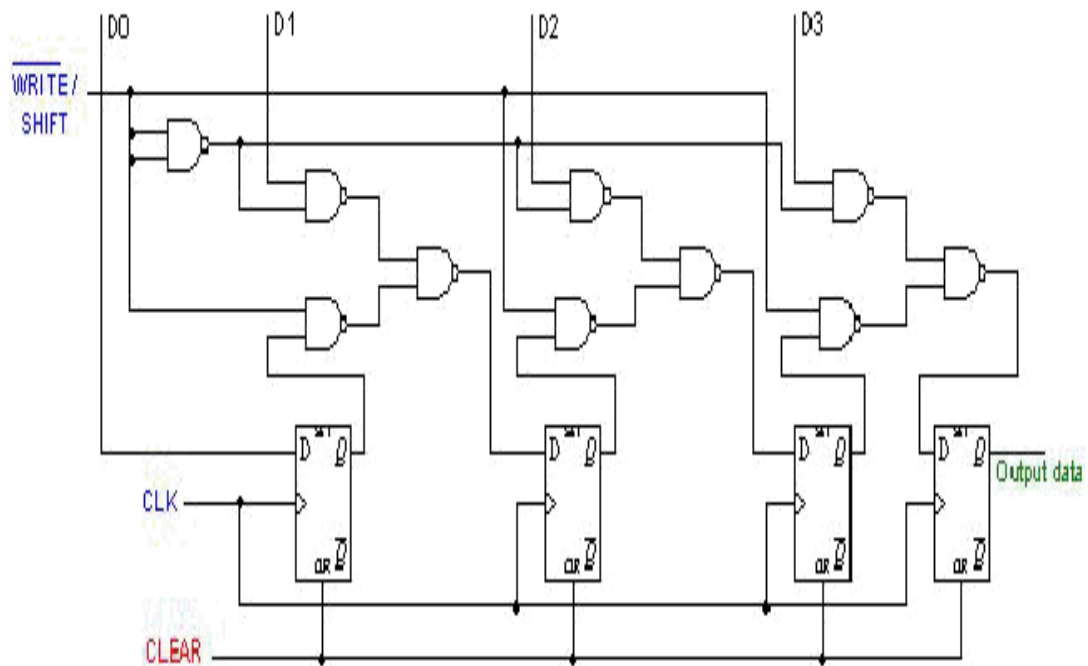
3) Parallel in parallel out shift register



Truthtable

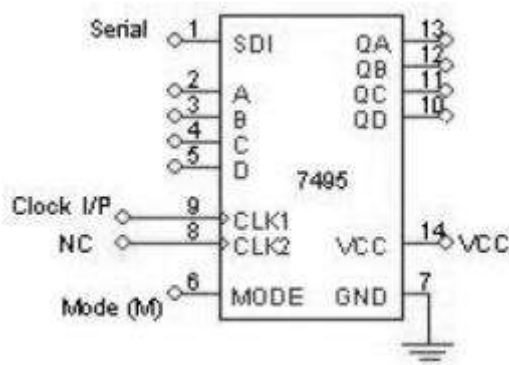
Clock	Parallel i/p				Parallel o/p			
	A	B	C	D	QA	QB	QC	QD
1	1	0	1	1	1	0	1	1

4) Parallel in serial out shift register

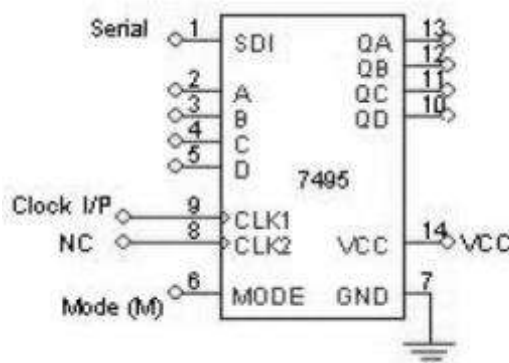


Truthtable

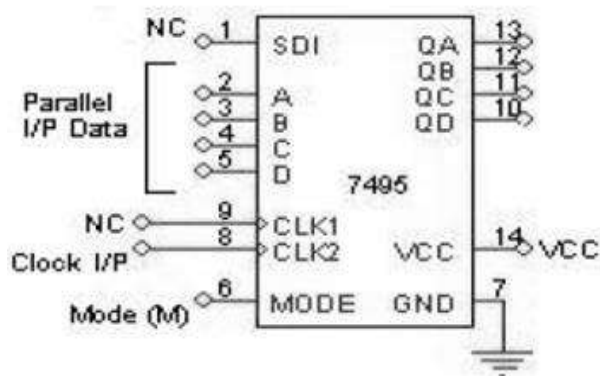
Mode	Clock	Parallel i/p				Parallel o/p			
		A	B	C	D	QA	QB	QC	QD
1	1	1	0	1	1	1	0	1	1
0	2	X	X	X	X	X	1	0	1
0	3	X	X	X	X	X	X	1	0
0	4	X	X	X	X	X	X	X	1

SHIFT REGISTERS USING 7495**SISO:**

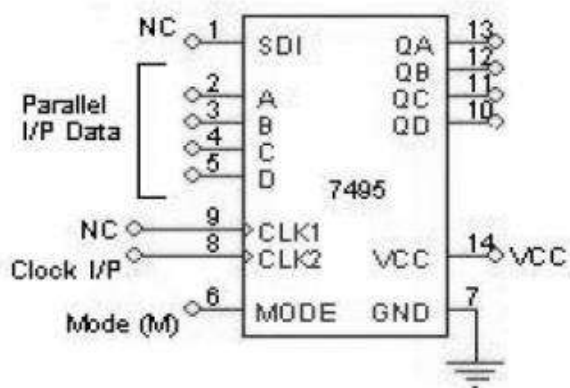
Clock	Serial i/p	QA	QB	QC	QD
1	do=0	0	X	X	X
2	d1=1	1	0	X	X
3	d2=1	1	1	0	X
4	d3=1	1	1	1	0=do
5	X	X	1	1	1=d1
6	X	X	X	1	1=d2
7	X	X	X	X	1=d3

SIPO(Right shift):

Clock	Serial i/p	QA	QB	QC	QD
1	0	0	X	X	X
2	1	1	0	X	X
3	1	1	1	0	X
4	1	1	1	1	0

PISO:

Mode	Clock	Parallel i/p				Parallel o/p			
		A	B	C	D	QA	QB	QC	QD
1	1	1	0	1	1	1	0	1	1
0	2	X	X	X	X	X	1	0	1
0	3	X	X	X	X	X	X	1	0
0	4	X	X	X	X	X	X	X	1

PIPO:

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

EXPERIMENT:**Procedure**

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Test all the components and IC packages using multimeter or digital IC tester. Set up the shift registers using flipflops and 4 bit shift register IC.
- 4) Verify the truth tables.

CONCLUSION

The shift registers are implemented and outputs are obtained as per truth table.

EXPT NO:12

DATE: __/__/__

RING COUNTER AND JOHNSON COUNTER**OBJECTIVE:**

To design and Implement counters using shift registers – Ring Counter and Johnson Counter.

HARDWARE REQUIRED:

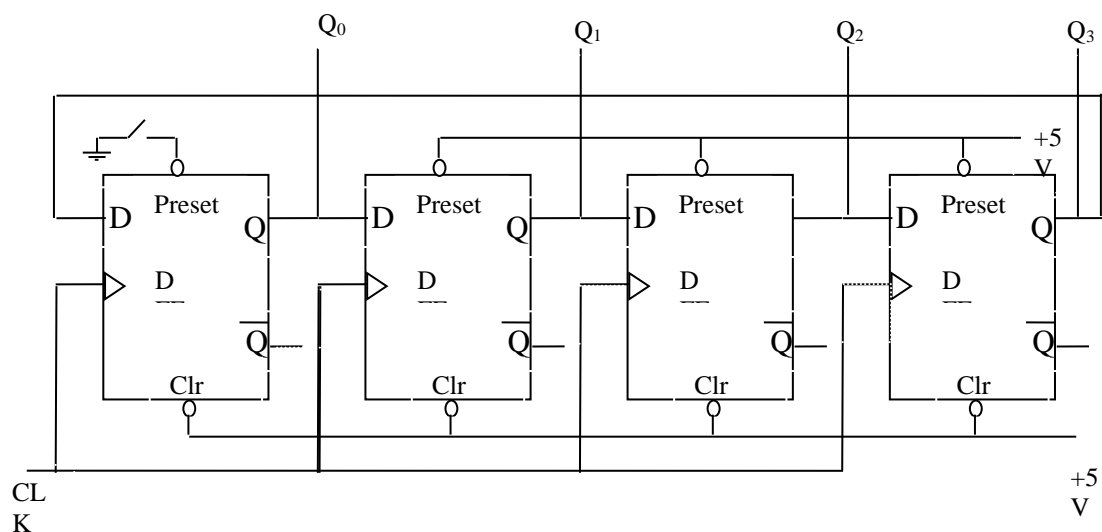
SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	7474	1

INTRODUCTION:**Ring Counter**

A ring counter is basically a circulating shift register in which the output of the MSB is fed back to the input of the LSB. Here the 4 bit ring counter is constructed using D flip-flops. The output of each stage is shifted into the next stage on the positive edge of a clock pulse. Here the clear signal is high, all the flip-flops except first one FF are reset to 0. FF0 is preset to 1 instead.

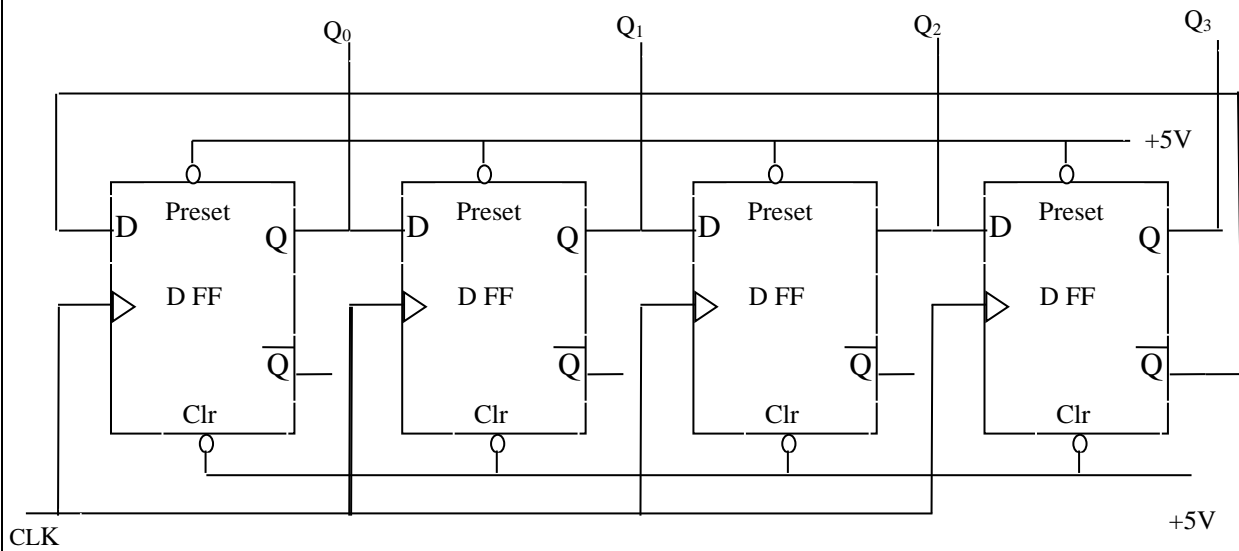
Johnson Counter

They are a variation of standard ring counters, with the inverted output of the last stage fed back to the first stage. They are also known as twisted ring counters. An n-stage Johnson counter yields a count sequence of length $2n$, so it may be considered to be mod $2n$ counter.

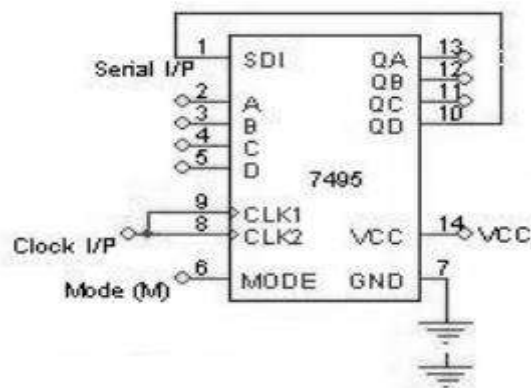
CIRCUIT DIAGRAM AND OBSERVATIONS:**1) RING COUNTER**

Truthtable

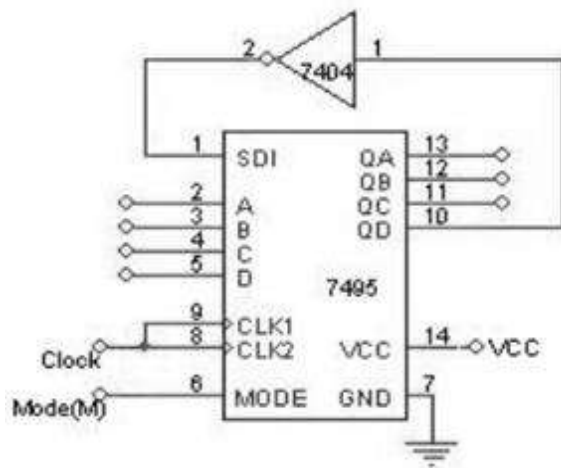
Clock Pulse	Q ₃	Q ₂	Q ₁	Q ₀
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	1	0	0	0

2) JOHNSON COUNTER**Truthtable**

Clock Pulse	Q ₃	Q ₂	Q ₁	Q ₀
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	1	1	1
4	1	1	1	1
5	1	1	1	0
6	1	1	0	0
7	1	0	0	0
8	0	0	0	0

RING COUNTER USING 7495:

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			

JOHNSON COUNTER USING 7495

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			

EXPERIMENT:**Procedure**

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Connect the inputs to the input switches provided in the IC Trainer Kit.
- 4) Connect the outputs to the switches of output LEDs
- 5) Verify the truthtable.

CONCLUSION:

Ring counter and Johnson counter using shift registers are designed and implemented

EXPT NO: 13

DATE: __/__/__

REALIZATION OF COUNTERS USING IC'S(7490,7492,7493)**OBJECTIVE:**

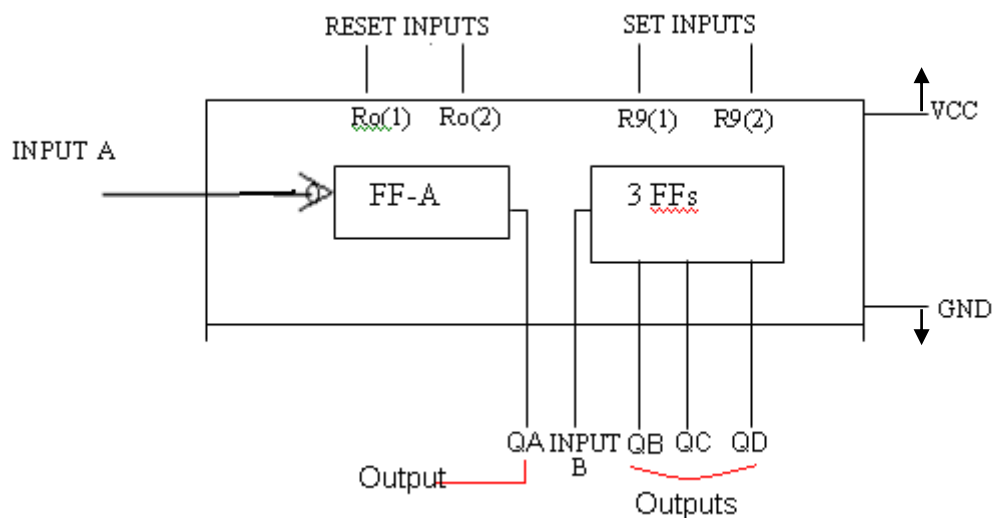
To realize counters using IC'S 7490,7492 and 7493.

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		
2.	IC	7490	1
		7492	1
		7493	1
		7473	2

INTRODUCTION:**Ripple counter IC-7490 (decade counter):**

This counter having various name like mod-n counter, decade counter ,BCD counter. IC-7490 is a TTL MSI decade counter. It contains four master slave flip flops and additional gating to provide a divide-by-two counter and a three stage binary counter which provides a divide by 5 counter.

**The Basic internal structure of IC 7490**

1. If both the reset input Ro(1) & Ro(2) are at logic 1 then all the flip-flop will be reset and the output is given by

$$QD \ QC \ QB \ QA = 0000$$

2. If both the reset input R9(1) & R9(2) are at logic 1 then the counter output is set to decimal 9.

$$QD \ QC \ QB \ QA = 1001$$

3. If any one pin of Ro(1) & Ro(2) and one of R9(1) & R9(2) are at low, then the counter

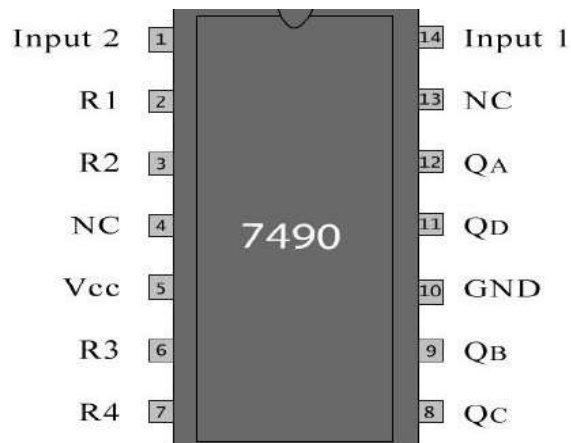
will be in counting mode.

The reset/count function table of IC7490 is shown in table 1.

Table 1:-Reset/count truth table

Reset inputs				Output			
R0(1)	R0(2)	R9(0)	R9(1)	QD	QC	QB	QA
1	1	0	X	0	0	0	0
1	1	X	0	0	0	0	0
X	X	1	1	1	0	0	1
X	0	X	0	COUNTER			
0	X	0	X	COUNTER			
0	X	X	0	COUNTER			
X	0	0	x	COUNTER			

IC 7490 is MOD-10 or decade counter. It is a 14 pin IC with the pin configuration as shown in fig:



Pin configuration of IC7490.

Table 2:-Pin name and description of IC 7490

Pin name	Description
Input B	This is clock input to the internal MOD-5 ripple counter, which is negative edge triggered.
R0(1),R0(2)	Gated zero reset inputs
R9(1),R9(2)	These are gated set to nine inputs
QD,QC,QB	Output of internal MOD-5 counter with QD as MSB.
QA	Output of internal MOD-2 counter with QA as LSB.
Input A	Clock input to FF-A which is negative edge triggered.

Decade Counter Operation :

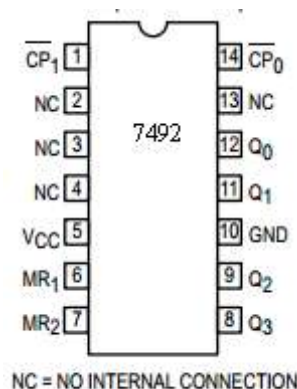
1. The output of MOD-2 is externally connected to the input B which is the clock input of the internal MOD-5 counter.
2. Hence QA toggles on every falling edge of clock input whereas the output QD, QC, QB of the MOD-5 counter will increment from 000 to 100 on low going change of QA output.
3. Due to cascading of MOD-2 and MOD-5 counter, the overall configuration becomes a MOD-10 i.e. decade counter.
4. The reset inputs Ro(1), Ro(2) and preset inputs R9(1), R9(2) are connected to ground so as to make them inactive.

IC-7492 (Divide by 12 counter):

The 7492 / 74LS92 counter IC is a 4-Bit ripple counter (4 cascaded counting elements). The individual counting circuits inside are partitioned into two blocks, one is a divide by two counter and the other capable of divide by 6, which when combined together effectively implements a divide by 12, perfect for the hour tracking register for a digital clock.

Modulo 12, Divide-By-Twelve Counter:- The CP1' input must be externally connected to the Q0 output. The CP0' input receives the incoming count and Q3 produces a symmetrical divide-by-twelve square wave output.

Divide by two and Divide-By-Six Counter:- No external interconnections are required. The first flip-flop is used as a binary element for the divide-by-two- function. The CP1' input is used to obtain divide-by-two operation at Q1 and Q2 outputs and divide-by-six operation at the Q3 output.

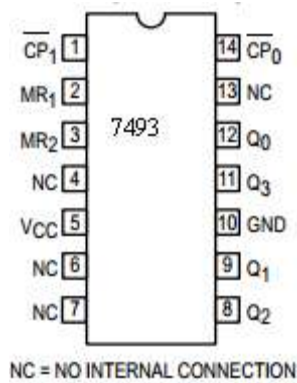


Pin configuration of IC7492

IC-7493(Divide by 16 counter):

This counter contains four master-slave flip-flops and additional gating to provide a divide-by-two counter and a three-stage binary counter for which the count cycle length is divide-by-eight for the 7493A.

- State changes of Q_n outputs do not occur simultaneously.
- Setting both MR_1 and MR_2 high resets the counter to zero.
- For a 4-bit counter, connect Q_0 to CP_1 , and apply count pulses to CP_0 .
- For a 3-bit counter, apply count pulses to CP_1 .



Pin configuration of IC7492

Pin	Symbol	Description
1	CP ₁	clock input, 2nd, 3rd and 4th section (high-to-low edge-triggered)
2	MR ₁	asynchronous master reset
3	MR ₂	asynchronous master reset
4	NC	no connection
5	V _{cc}	supply voltage
6	NC	no connection
7	NC	no connection
8	Q ₂	counter output
9	Q ₁	counter output
10	GND	Ground
11	Q ₃	counter output
12	Q ₀	counter output
13	NC	no connection
14	CP ₀	clock input, 1st section (high-to-low edge-triggered)

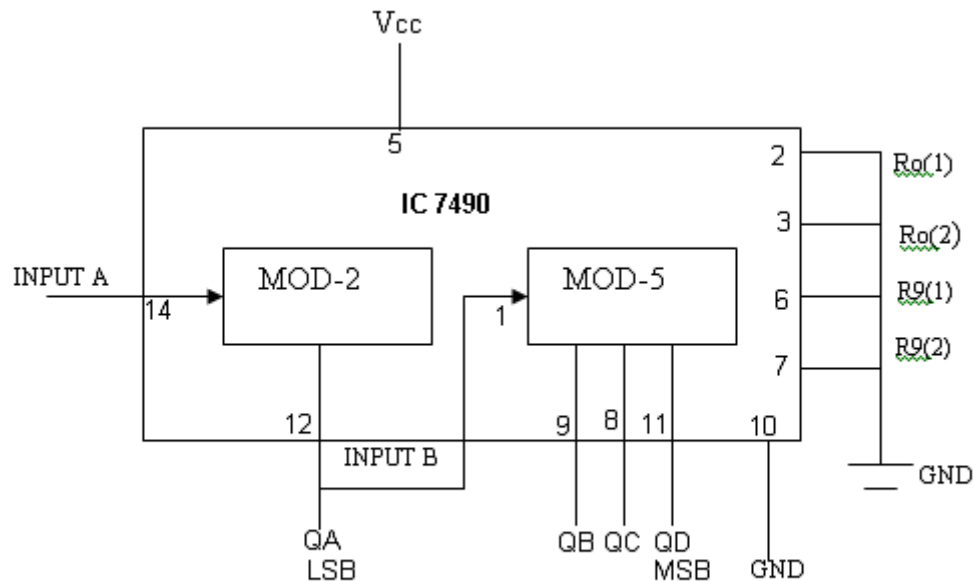
LS92 AND LS93 MODE SELECTION

RESET INPUTS		OUTPUTS			
MR ₁	MR ₂	Q ₀	Q ₁	Q ₂	Q ₃
H	H	L	L	L	L
L	H	Count			
H	L	Count			
L	L	Count			

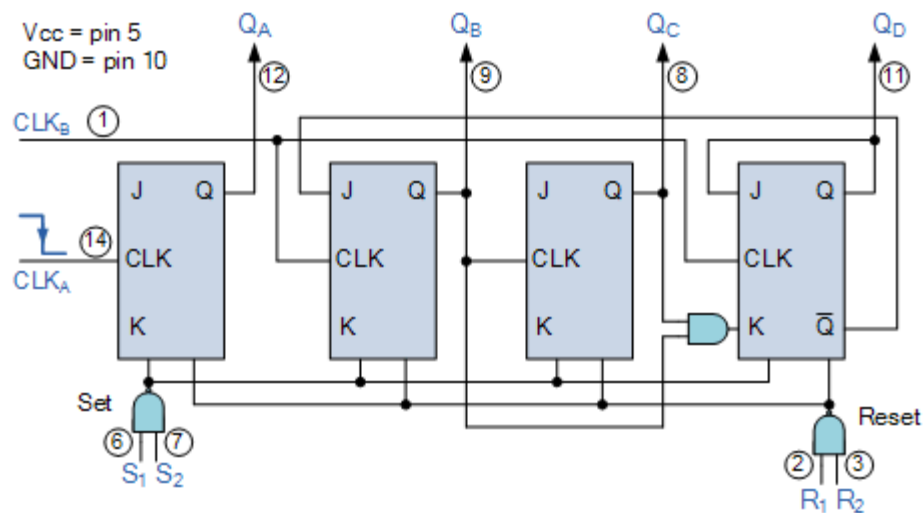
H = HIGH Voltage Level

L = LOW Voltage Level

X = Don't Care

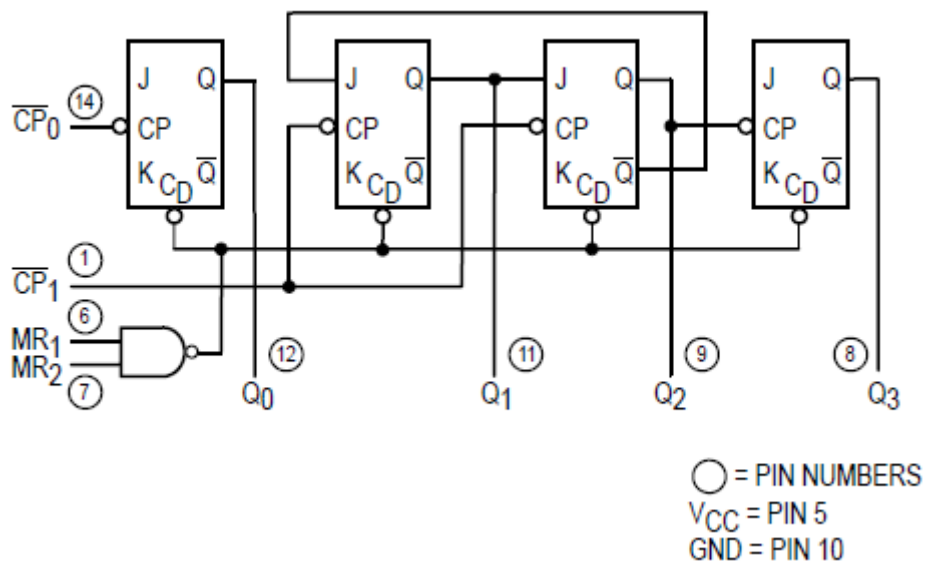
CIRCUIT DIAGRAM AND OBSERVATIONS:**(1) 7490 BCD COUNTER**Realization of MOD 10 counter using IC 7490

Note:- Pin 4 & 13 For NC

7490 BCD COUNTER USING 7473

Truth Table of the 7490 as decade counter

O/p of MOD-5			O/p of MOD-2	CLK	Count
QD	QC	QB	QA		
0	0	0	0	0	0
0	0	0	1	1	1
0	0	1	0	2	2
0	0	1	1	3	3
0	1	0	0	4	4
0	1	0	1	5	5
0	1	1	0	6	6
0	1	1	1	7	7
1	0	0	0	8	8
1	0	0	1	9	9

(2) 7492(DIVIDE BY 12 COUNTER)

COUNT	OUTPUT			
	Q ₀	Q ₁	Q ₂	Q ₃
0	L	L	L	L
1	H	L	L	L
2	L	H	L	L
3	H	H	L	L
4	L	L	H	L
5	H	L	H	L
6	L	L	L	H
7	H	L	L	H
8	L	H	L	H
9	H	H	L	H
10	L	L	H	H
11	H	L	H	H

(3)7493(DIVIDE BY 16 COUNTER)



COUNT	OUTPUT			
	Q ₀	Q ₁	Q ₂	Q ₃
0	L	L	L	L
1	H	L	L	L
2	L	H	L	L
3	H	H	L	L
4	L	L	H	L
5	H	L	H	L
6	L	H	H	L
7	H	H	H	L
8	L	L	L	H
9	H	L	L	H
10	L	H	L	H
11	H	H	L	H
12	L	L	H	H
13	H	L	H	H
14	L	H	H	H
15	H	H	H	H

84

EXPERIMENT:**Procedure**

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Connect the inputs to the input switches provided in the IC Trainer Kit.
- 4) Connect the outputs to the switches of output LEDs
- 5) Verify the truthtables
- 6)

CONCLUSION:

Realized the counters using IC's 7490, 7492 and 7493.

EXPT NO:14

DATE: __/__/__

MULTIPLEXERS AND DEMULTIPLEXERS USING GATES AND ICS(74150,74154)**OBJECTIVE:**

To design and implement multiplexer and demultiplexer using gates and IC's.

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		
2.	IC	74150	1
		74154	1

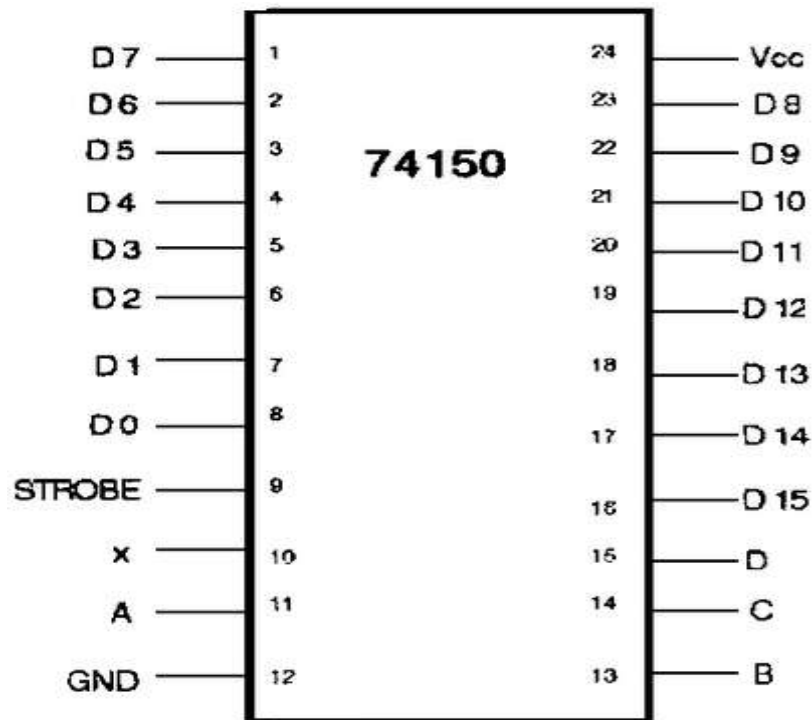
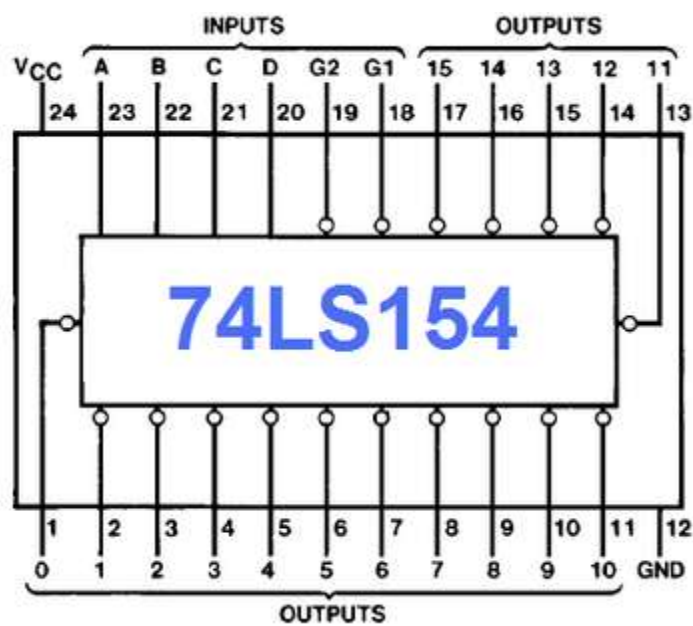
INTRODUCTION:**MULTIPLEXER:**

Multiplexer means transmitting a large number of information units over a smaller number of channels or lines. A digital multiplexer is a combinational circuit that selects binary information from one of many input lines and directs it to a single output line. The selection of a particular input line is controlled by a set of select lines. Normally there are 2^n input lines and n selection lines whose bit combination determines which input is selected.

DEMULTIPLEXER:

The function of Demultiplexer is in contrast to multiplexer function. It takes information from one line and distributes it to a given number of output lines. For this reason, the demultiplexer is also known as a data distributor. Decoder can also be used as demultiplexer.

In the 1: 4 demultiplexer circuit, the data input line goes to all of the AND gates. The data select lines enable only one gate at a time and the data on the data input line will pass through the selected gate to the associated data output line.

CIRCUIT DIAGRAM :**16:1 MULTIPLEXER****1:16 DEMULTIPLEXER /DECODER**

EXPERIMENT:**Procedure**

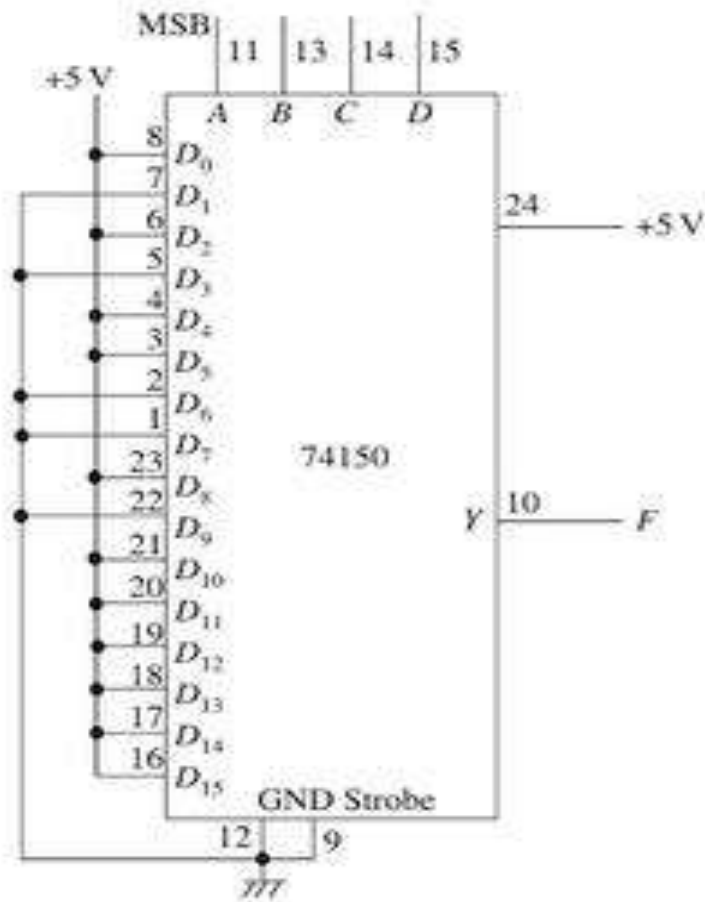
- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Connect the inputs to the input switches provided in the IC Trainer Kit.
- 4) Connect the outputs to the switches of output LEDs
- 5) Apply various combinations of inputs according to the truth table and observe condition of LEDs

DESIGN:

1. Implement $F(A,B,C,D) = A'B'C'D + A'B'CD + A'BCD' + A'BCD + AB'C'D$ using IC 74151

Truth table

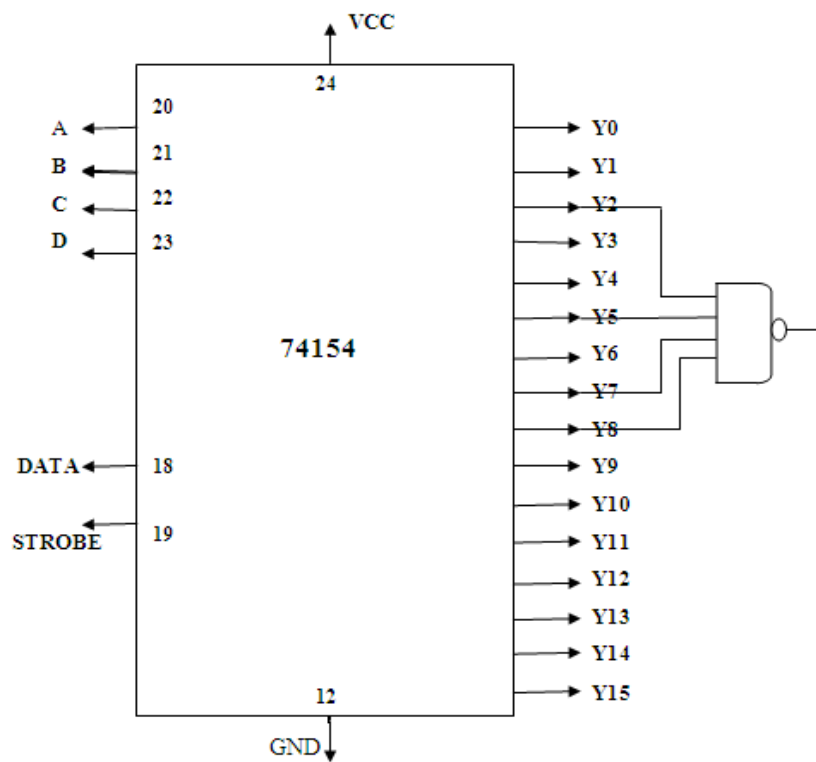
Data select				Output
<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>F</i>
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0



1. Implement $F = A'B'CD' + A'BC'D' + A'BCD + AB'C'D'$ using IC 74154.

Truth table

A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0



OBSERVATIONS:

Truth table for 74150 multiplexer

Inputs					Output Y
Select				Enable E	
S3	S2	S1	S0		
X	X	X	X	H	H
L	L	L	L	L	$\overline{D_0}$
L	L	L	H	L	$\overline{D_1}$
L	L	H	L	L	$\overline{D_2}$
L	L	H	H	L	$\overline{D_3}$
L	H	L	L	L	$\overline{D_4}$
L	H	L	H	L	$\overline{D_5}$
L	H	H	L	L	$\overline{D_6}$
L	H	H	H	L	$\overline{D_7}$
H	L	L	L	L	$\overline{D_8}$
H	L	L	H	L	$\overline{D_9}$
H	L	H	L	L	$\overline{D_{10}}$
H	L	H	H	L	$\overline{D_{11}}$
H	H	L	L	L	$\overline{D_{12}}$
H	H	L	H	L	$\overline{D_{13}}$
H	H	H	L	L	$\overline{D_{14}}$
H	H	H	H	L	$\overline{D_{15}}$

Truth table for 74154 Demultiplexer

INPUTS					OUTPUTS															
\bar{E}	A_0	A_1	A_2	A_3	\bar{O}_0	\bar{O}_1	\bar{O}_2	\bar{O}_3	\bar{O}_4	\bar{O}_5	\bar{O}_6	\bar{O}_7	\bar{O}_8	\bar{O}_9	\bar{O}_{10}	\bar{O}_{11}	\bar{O}_{12}	\bar{O}_{13}	\bar{O}_{14}	\bar{O}_{15}
H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	H	L	L	L	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	H	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H
L	H	H	L	L	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H
L	H	L	H	L	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H
L	H	H	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H
L	L	L	L	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H
L	H	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H
L	L	H	L	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H
L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H
L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H
L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L

CONCLUSION:

Designed MUX and DEMUX using gates and IC's

EXPT NO:15

DATE: __/__/__

REALIZATION OF COMBINATIONAL CIRCUITS USING MUX AND DEMUX**OBJECTIVE:**

To realize combinational circuits using MUX and DEMUX.

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		
2.	IC	7411	2
		7432	1
		7404	1

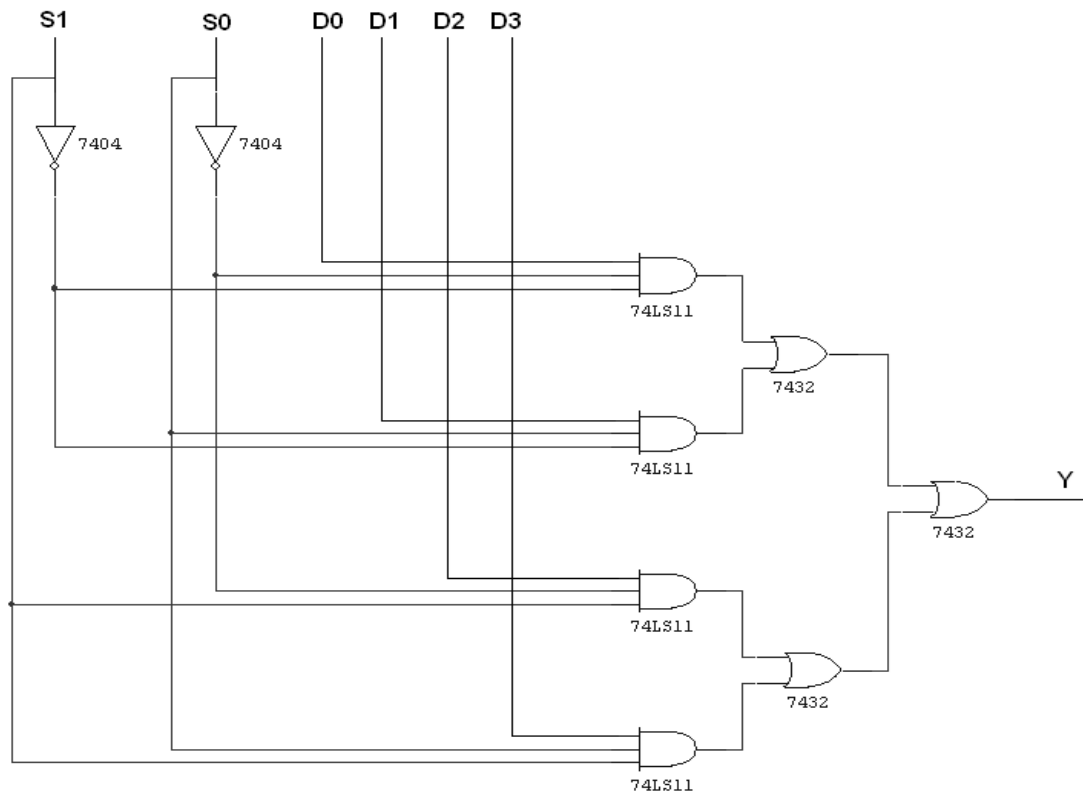
INTRODUCTION:**MULTIPLEXER:**

Multiplexer means transmitting a large number of information units over a smaller number of channels or lines. A digital multiplexer is a combinational circuit that selects binary information from one of many input lines and directs it to a single output line. The selection of a particular input line is controlled by a set of selection lines. Normally there are 2^n input line and n selection lines whose bit combination determine which input is selected.

DEMULTIPLEXER:

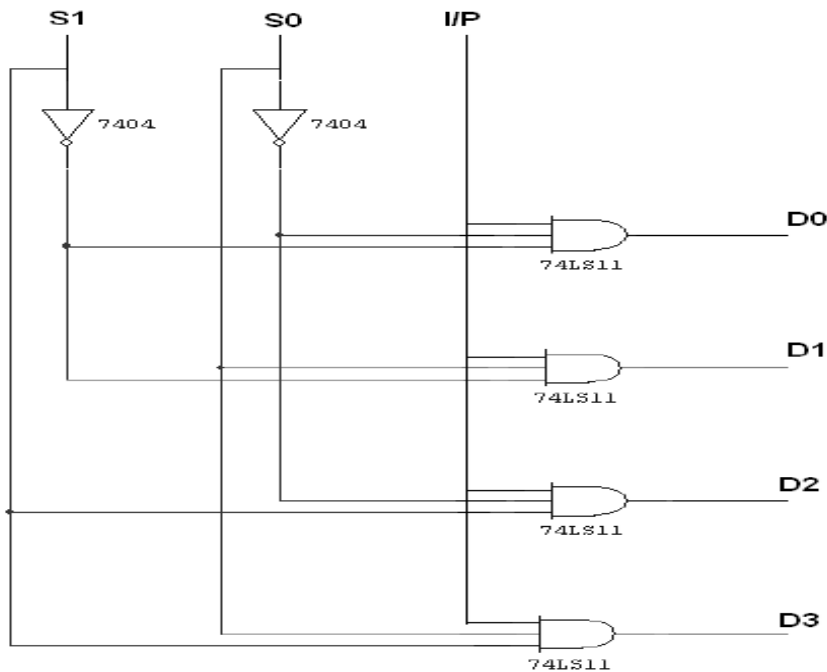
The function of Demultiplexer is in contrast to multiplexer function. It takes information from one line and distributes it to a given number of output lines. For this reason, the demultiplexer is also known as a data distributor. Decoder can also be used as demultiplexer.

In the 1: 4 demultiplexer circuit, the data input line goes to all of the AND gates. The data select lines enable only one gate at a time and the data on the data input line will pass through the selected gate to the associated data output line.

CIRCUIT DIAGRAM :**4:1 MULTIPLEXER:****FUNCTION TABLE:**

S1	S0	INPUTS Y
0	0	$D0 \rightarrow D0 S1' S0'$
0	1	$D1 \rightarrow D1 S1' S0$
1	0	$D2 \rightarrow D2 S1 S0'$
1	1	$D3 \rightarrow D3 S1 S0$

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

1:4 DEMULTIPLEXER:**FUNCTION TABLE:**

S1	S0	INPUT
0	0	$X \rightarrow D0 = X S1' S0'$
0	1	$X \rightarrow D1 = X S1' S0$
1	0	$X \rightarrow D2 = X S1 S0'$
1	1	$X \rightarrow D3 = X S1 S0$

EXPERIMENT:**Procedure**

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Connect the inputs to the input switches provided in the IC Trainer Kit.
- 4) Connect the outputs to the switches of output LEDs
- 5) Apply various combinations of inputs according to the truth table and observe condition of LEDs.

OBSERVATIONS:

Truth table for multiplexer using gates.

S1	S0	Y = OUTPUT
0	0	D0
0	1	D1
1	0	D2
1	1	D3

Truth table for Demultiplexer using gates.

INPUT			OUTPUT			
S1	S0	I/P	D0	D1	D2	D3
0	0	0	0	0	0	0
0	0	1	1	0	0	0
0	1	0	0	0	0	0
0	1	1	0	1	0	0
1	0	0	0	0	0	0
1	0	1	0	0	1	0
1	1	0	0	0	0	0
1	1	1	0	0	0	1

CONCLUSION:

Combinational circuits using MUX and DEMUX are realized

EXPT NO: 16

DATE: __/__/__

USE OF BCD TO 7 SEGMENT DECODER/ DRIVER CHIP TO DRIVE LED DISPLAY**OBJECTIVE:**

To set up and test seven segment static display system to display decimal numbers.

HARDWARE REQUIRED:

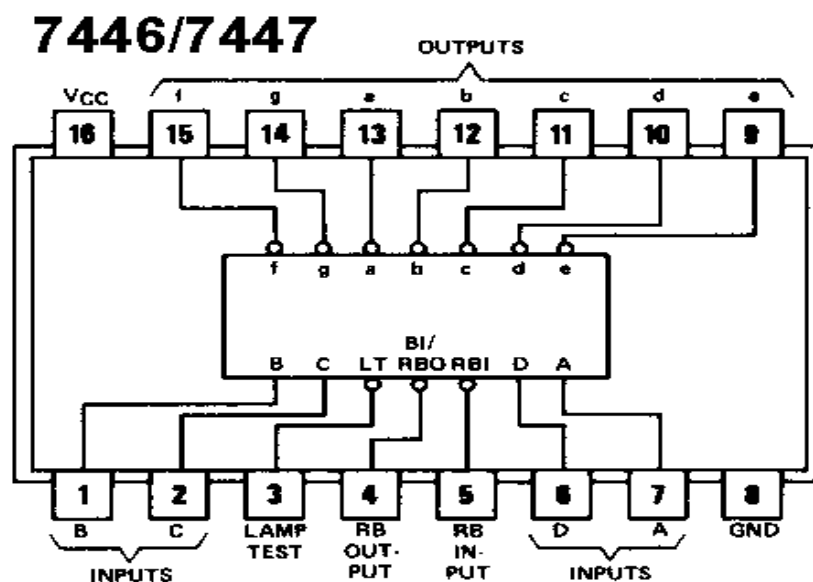
SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		
2.	IC	7447	1
		7490	1
3.	seven segment display	FND 507	1
4.	Resistor	1K	7

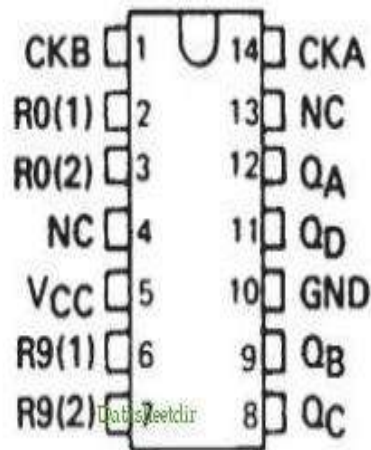
INTRODUCTION:

Digital displays are widely used for the front panels of electronic instruments, display boards, calculators etc. A single digit decimal number display system needs a decade counter, a seven segment indicator and decoder/driver.

7446

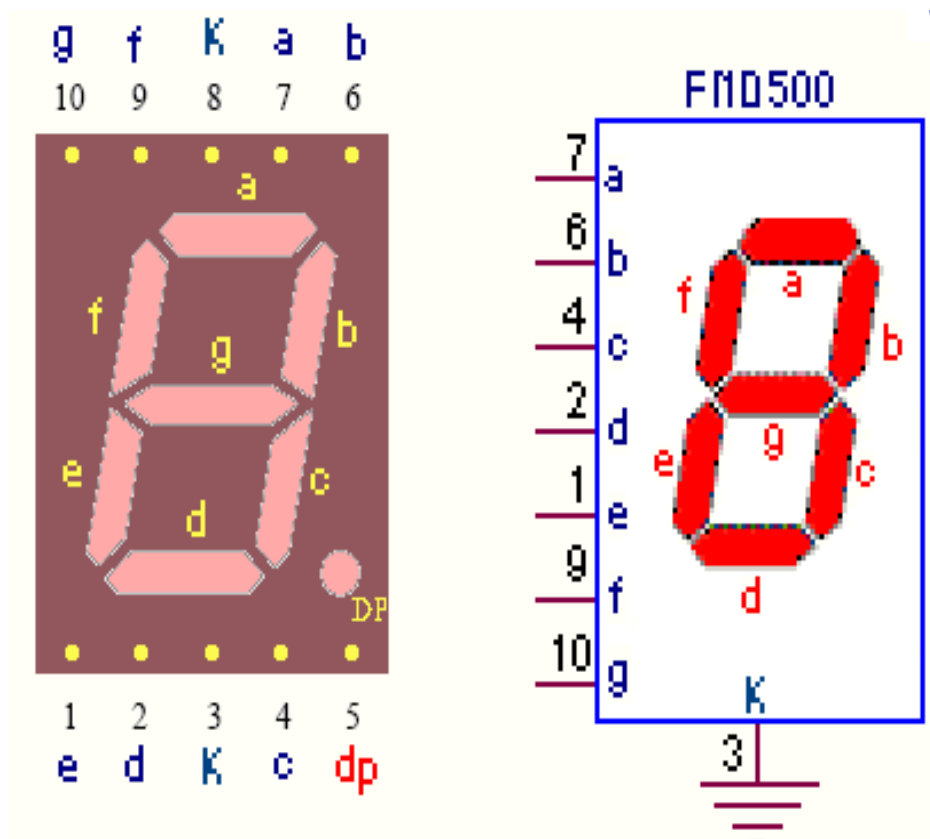
It is a decoder driver IC used to drive a seven segment indicator. 7446 is the decoder used to drive common anode indicator such as FND 507. Logic circuits inside the 7446 convert the 4 bit BCD input to the 7 bit output which are active low.





Seven segment display

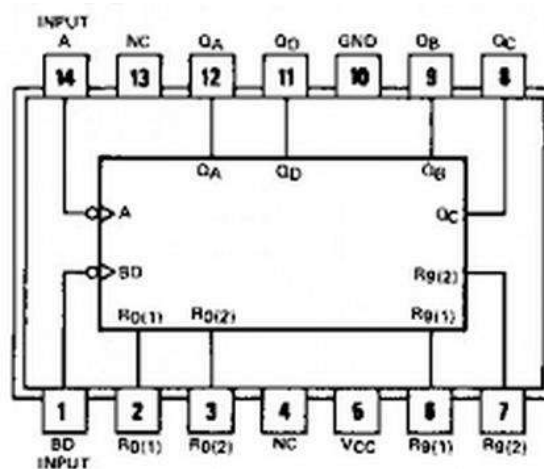
It consists of seven LEDs labeled a through g arranged in a shape of 8. When an led is forward biased that segment glows. Displays of 0 through 9 are possible by selectively forward biasing these LED's. FND 507 is a common anode type display. In common anode type, anodes of all seven LED's are tied together. Resistors are used to limit the current through LED's.



Pin outs of FND 507

7490

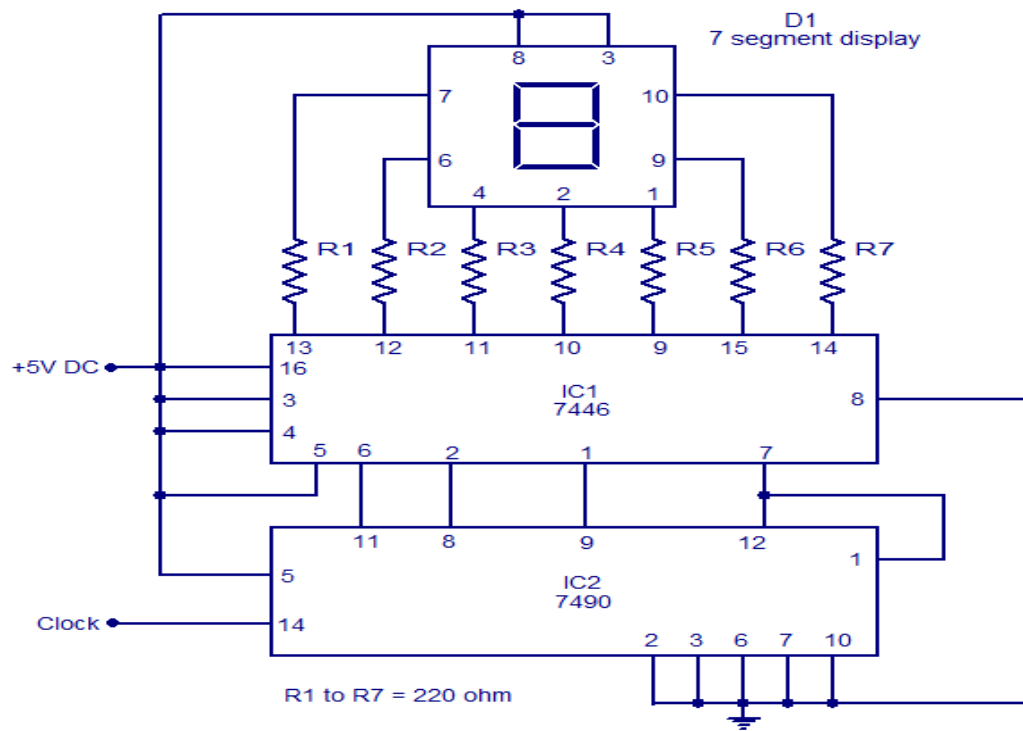
The **74LS90** is a simple counter, i.e. it can count from 0 to 9 cyclically in its natural mode. It counts the input pulses and the output is received as a 4-bit binary number through pins Q_A , Q_B , Q_C and Q_D . The binary output is reset to 0000 at every tenth pulse and count starts from 0 again. A pulse is also generated (probably at pin 9) as it resets its output to 0000. The chip can count up to other maximum numbers and return to zero by changing the modes of 7490. These modes are set by changing the connection of reset pins $R_1 - R_4$. For example, if either R_1 & R_2 are high or R_3 & R_4 are ground, then it will reset Q_A , Q_B , Q_C and Q_D to 0. If resets R_3 & R_4 are high, then the count on Q_A , Q_B , Q_C and Q_D goes to 1001.



Pin outs of 7490

CIRCUIT DIAGRAM :

Static display using common anode display



EXPERIMENT:

Procedure

1. Test all components and IC packages using multimeter.
2. Set up the circuit and feed 1Hz clock to 7490 clock input and observe the decimal display.

DESIGN:

$$R = \frac{V_{CC} - V_D}{I_D} = \frac{5 - 1.5}{20mA} = 175 \text{ ohm}$$

CONCLUSION:

Designed a seven segment LED display.

EXPT NO: 17

DATE: __/__/__

TRANSFER CHARACTERISTICS OF TTL AND CMOS NAND GATE**OBJECTIVE:**

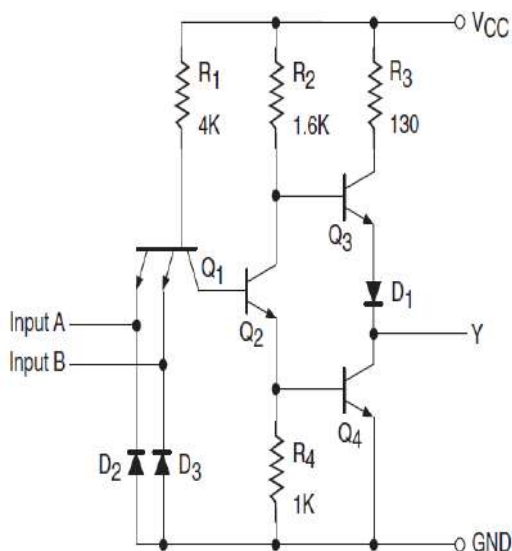
To plot the transfer characteristics of TTL and CMOS

HARDWARE REQUIRED:

SLNo.	Components/Equipments	Specification	Quantity
1.	Digital Trainer Kit		1
2.	IC	CD4011	1
		7400	1
3	Voltmeter	(0-5)V	2

INTRODUCTION:**TTL**

Transistor–transistor logic (TTL) is a class of digital circuits built from bipolar junction transistors and resistors. It is called transistor–transistor logic because both the logic gating function and the amplifying function are performed by transistors. TTL inputs are the emitters of a multiple-emitter transistor.



A	B	Q1	Q2	Q3	Q4	Y
L	L	sat	off	off	Off	H
L	H	sat	off	off	Off	H
H	L	sat	off	off	Off	H
H	H	iam	sat	sat	On	L

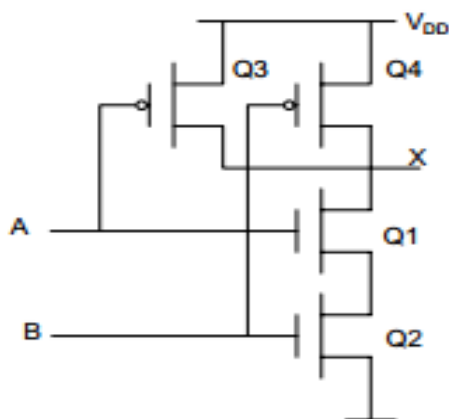
In the TTL gate the conducting state occurs when the base emitter signal is high, and the non conducting (off) when it is low. Signals corresponding to A and B applied to the emitters of the input transistor Q1 where the actual logic is essentially performed. Transistor Q2 plays the role of a phase splitter. When it is off, the collector voltage is at essentially V and the emitter is at 0. When it is conducting current flows through the series resistors. The collector voltage drops to $V - I_2 R$

and the emitter voltage rises to I_2R , where R is the appropriate resistor. The collector and emitter voltages then vary in opposite phase. Moreover each phase leads to only one of the two output transistors being conducting. Note that if either x or y is low (ie $=0$) then Q_1 is on and the base voltage is at its lowest value $V - I_1R$. Here again R is used generically, each element having distinct values in practice. The collector of Q_1 and the base of Q_2 are low so that Q_2 is off. In this situation the base of Q_3 is high turning this transistor on while the base of Q_4 is 0 and this transistor is off.

In this situation the output is connected to the high voltage line V and goes high. When connected to a load it will deliver current. In the case when both x and y are high then Q_1 is off, turning Q_2 on. This reduces the voltage on the base of Q_3 below its emitter voltage so that it becomes non-conducting. The base of Q_4 on the other hand is driven up by an amount I_2R and becomes conducting. In this case the output is connected to ground through Q_4 and the output is low. Note that in this state, when connected to a load, The pair of transistors Q_3 and Q_4 constitute what is known as a totem pole output. They are arranged so that only one of the two is conducting in a steady state condition

CMOS

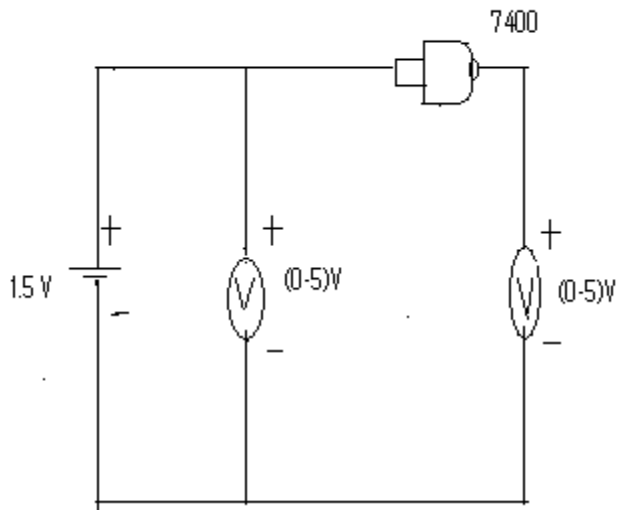
Figure shows a 2-input NAND gate. If either input is Low, the output X is High with low impedance connection to V_{DD} through the corresponding p-channel transistor, and the path to the ground is blocked by the corresponding OFF n-channel MOSFET. If both inputs are High, the two n-channel MOSFETs are ON and the two p-channel MOSFETs are OFF. This is the operation required for the circuit to function as a NAND gate.



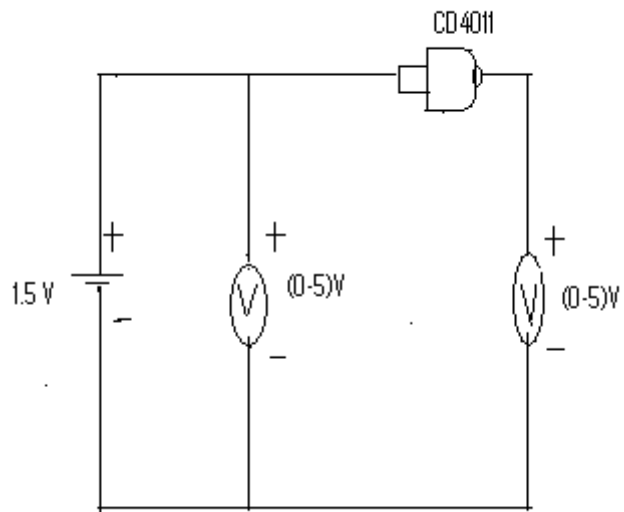
A	B	Q1	Q2	Q3	Q4	X
L	L	OFF	OFF	ON	ON	H
L	H	OFF	ON	ON	OFF	H
H	L	ON	OFF	OFF	ON	H
H	H	ON	ON	OFF	OFF	L

CIRCUIT DIAGRAM :

1) TTL



2) CMOS

**EXPERIMENT:****Procedure**

- 1) Place the IC on IC Trainer Kit.
- 2) Connect V_{CC} and ground to respective pins of IC Trainer Kit.
- 3) Connect the inputs to the input switches provided in the IC Trainer Kit.

- 4) Connect the outputs to the switches
- 5) Apply various combinations of inputs according to the truth table and observe condition of LEDs

DESIGN:

$$R = V_{CC} - V_D / I_D = \frac{5-1.5}{20mA} = 175 \text{ ohm}$$

OBSERVATION :

- 1) TTL

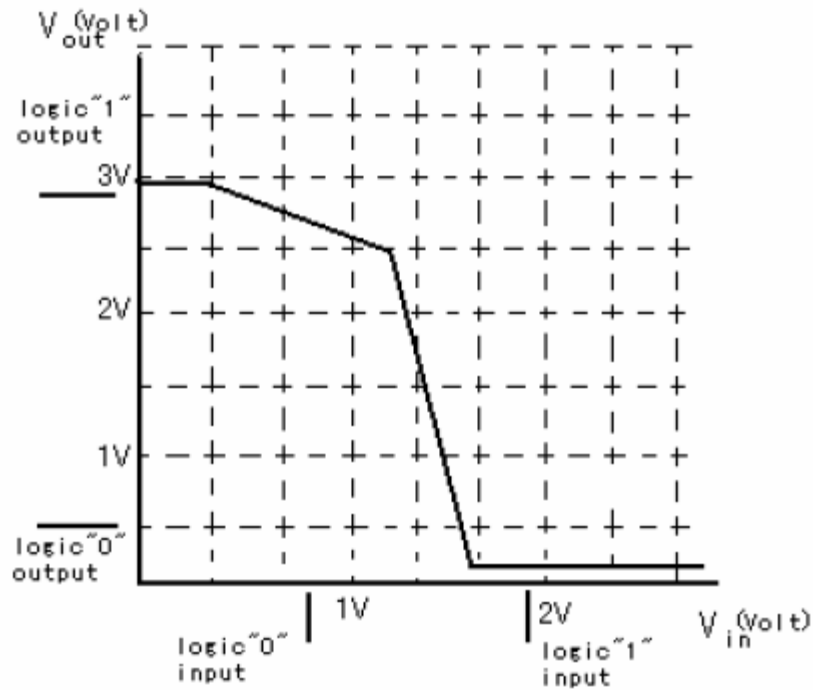
Vin	Vout

- 2) CMOS

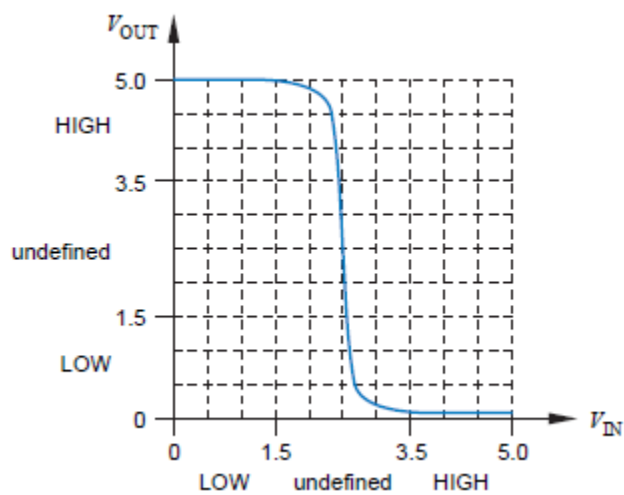
Vin	Vout

MODEL GRAPH:

1) TTL



2) CMOS

**CALCULATION:**TTL\Transition width = $V_{IH} - V_{IL} =$ Logic swing = $V_{OH} - V_{OL} =$ High level DC Noise margin = $V_{OH} - V_{IH} =$

Low level DC Noise margin= $V_{OL}-V_{IL}$ =

CONCLUSION:

Transfer Characteristics of TTL and CMOS are plotted.