



Techno India NJR Institute of Technology

Department of Electronics & Communication Engineering

B.Tech. III Semester

Lab: Digital System Design (3EC4-22)



RAJASTHAN TECHNICAL UNIVERSITY, KOTA

SYLLABUS

II Year - III Semester: B.Tech. (Electronics & Communication Engineering)

3EC4-22: Digital System Design Lab

1 Credit

Max. Marks: 50 (IA:30, ETE:20)

OL:OT:2P

List of Experiments

S.No.	Name of Experiment
Part A: Combinational Circuits	
1.	To verify the truth tables of logic gates: AND, OR, NOR, NAND, NOR, Ex-OR and Ex-NOR
2.	To verify the truth table of OR, AND, NOR, Ex-OR, Ex-NOR logic gates realized using NAND & NOR gates.
3.	To realize an SOP and POS expression.
4.	To realize Half adder/ Subtractor & Full Adder/ Subtractor using NAND & NOR gates and to verify their truth tables
5.	To realize a 4-bit ripple adder/ Subtractor using basic Half adder/ Subtractor & basic Full Adder/ Subtractor.
6.	To design 4-to-1 multiplexer using basic gates and verify the truth table. Also verify the truth table of 8-to-1 multiplexer using IC
7.	To design 1-to-4 demultiplexer using basic gates and verify the truth table. Also to construct 1-to-8 demultiplexer using blocks of 1-to-4 demultiplexer
8.	To design 2x4 decoder using basic gates and verify the truth table. Also verify the truth table of 3x8 decoder using IC
9.	Design & Realize a combinational circuit that will accept a 2421 BCD code and drive a TIL -312 seven-segment display
Part B: Sequential Circuits	
10.	Using basic logic gates, realize the R-S, J-K and D-flip flops with and without clock signal and verify their truth table.
11.	Construct a divide by 2, 4 & 8 asynchronous counter. Construct a 4-bit binary counter and ring counter for a particular output pattern using D flip flop.
12.	Design and construct unidirectional shift register and verify the
13.	Design and construct BCD ripple counter and verify the function.
14.	Design and construct a 4 Bit Ring counter and verify the function
15.	Perform input/output operations on parallel in/Parallel out and Serial in/Serial out registers using clock. Also exercise loading only one of multiple values into the register using multiplexer.

Note: Minimum 6 experiments to be conducted from **Part-A** & 4 experiments to be conducted from **Part-B**.

Office of Dean Academic Affairs
Rajasthan Technical University, Kota



RAJASTHAN TECHNICAL UNIVERSITY, KOTA

SYLLABUS

II Year - III Semester: B.Tech. (Electronics & Communication Engineering)

Course Outcome:

Course Code	Course Name	Course Outcome	Details
3EC4-22	Digital System Design Lab	CO 1	To verify TTL IC's functionality with the data sheets.
		CO 2	To minimize the complexity of digital logic circuits.
		CO 3	To design and analyse combinational logic circuits.
		CO 4	To design and analyse sequential logic circuits.
		CO 5	Able to implement applications of combinational & sequential logic circuits.

CO-PO Mapping:

Subject	Course Outcomes	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12
3EC4-22 Digital System Design Lab	CO 1	3	3	1									1
	CO 2	3	3	2	1	1							1
	CO 3	3	3	3	2	3	1						2
	CO 4	3	3	3	2	3	1						2
	CO 5	3	3	3	3	3	3						3

3: Strongly

2: Moderate

1: Weak

Office of Dean Academic Affairs
Rajasthan Technical University, Kota

Course Outcomes:

CO1:	Knowledge	To Verify the TTL ICs functionality with the datasheet.
CO2:	Comprehension	To classify the responses of different logic family.
CO3:	Analysis	To calculate and analyse delays combinational and sequential logic circuits.
CO4:	Synthesis	To design and develop Moore and Melay FSM .
CO5:	Evaluation	Able to Evaluate and predict the output of combinational & sequential logic circuits.

Course Outcome Mapping with Program Outcome:

CO. NO.	Domain Specific					Domain Independent						
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	1	1	2	1	1	2	0	0	0	0	1	1
CO2	1	2	1	1	2	1	0	0	0	0	2	2
CO3	2	2	1	2	2	2	2	0	0	0	2	1
CO4	2	2	2	1	2	1	2	0	0	0	2	1
CO5	1	1	1	1	2	1	0	0	0	0	1	2

1: Slight (Low) . 2: Moderate (Medium). 3: Substantial (High)

Instructions for the Lab

DO'S

1. Student should get the record of previous experiment checked before starting the new experiment.
2. Read the manual carefully before starting the experiment.
3. Before starting the experiment, get circuit diagram checked by the teacher.
4. Before switching on the power supply, get the circuit connections checked.
5. Get your readings checked by the teacher.
6. Apparatus must be handled carefully.
7. Maintain strict discipline.
8. Keep your mobile phone switched off or in vibration mode.
9. **Students should get the experiment allotted for next turn, before leaving the lab.**

DON'TS

1. Do not touch or attempt to touch the mains power supply wire with bare hands.
2. Do not overcrowd the tables.
3. Do not tamper with equipment's.
4. Do not leave the lab without permission from the teacher.

Introduction to the DSD – Lab

Digital System Design is the Lab of electronics and computer science that deals with the digital inputs and outputs to perform various tasks. It is based upon the digital design methodologies and consists of digital circuits, IC's and logic gates. It uses only binary digits, i. e. either '0' or '1'.

We know there are two types of signals, one is analog or continuous signal and the second one is Digital or discrete signal. Now coming to the area of Digital system design, it is essential to understand wide range of applications from industrial electronics to the fields of communication, from micro embedded systems to military equipment. The main and perhaps the most revolutionary advantage of digital system design is the decrease in size and the improvement in technology.

The world of electronics was initially dominated by analogue signals—that is, signals representing a continuous range of values. In digital circuitry, however, there are only two states: on and off, also referred to as 1 and 0, respectively. Digital information has its roots back in the Victorian era thanks to George Boole, who developed the idea of Boolean algebra. Every aspect of our lives is increasingly becoming integrated and connected by the Internet of Things (IoT), which consists of computers and embedded systems. These devices are controlled by software which at its core is Boolean logic in conjunction with digital information. The world around us is analogue, but with every passing day our interaction with the world is becoming more digital and more integrated.

Some general things to know in this lab are:

- **Bread Board/ Proto Board**

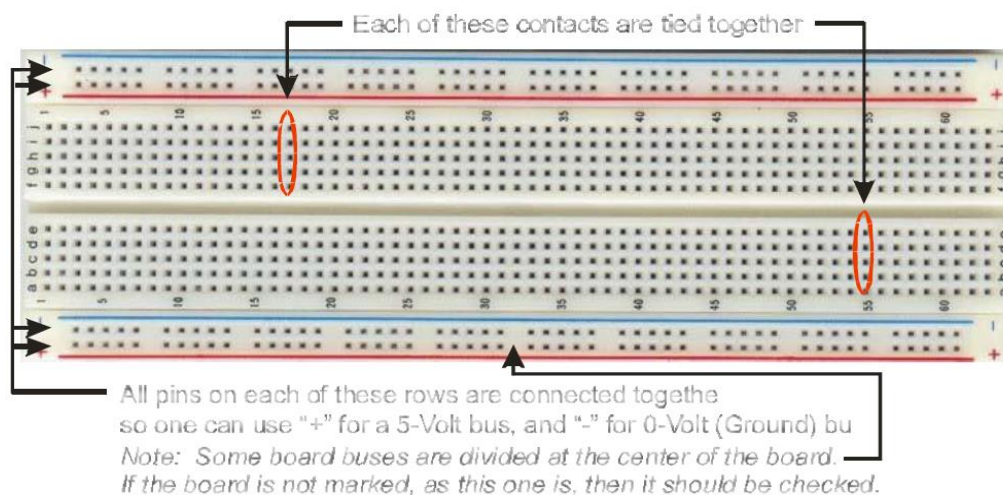


Figure 1: Bread Board and internal connections

Your kit includes a plastic board used to wire together electric circuits. This is called a breadboard or a proto board, since it is used to prototype circuits. Figure 1 above shows how the terminals are connected internally inside the board. The horizontal connections X and Y are called buses and are usually for power and ground.

Usually, the top row is connected to the +5V power supply and is called the power bus. The bottom row is connected to an external ground and is called the ground bus.

Figure 2 shows how the power and ground pins of IC1 and IC2 can be connected together by the red wires and black wires. It also shows how the output of pin 3 on IC1 can be connected to the input pin 1 of IC2 with a green wire.

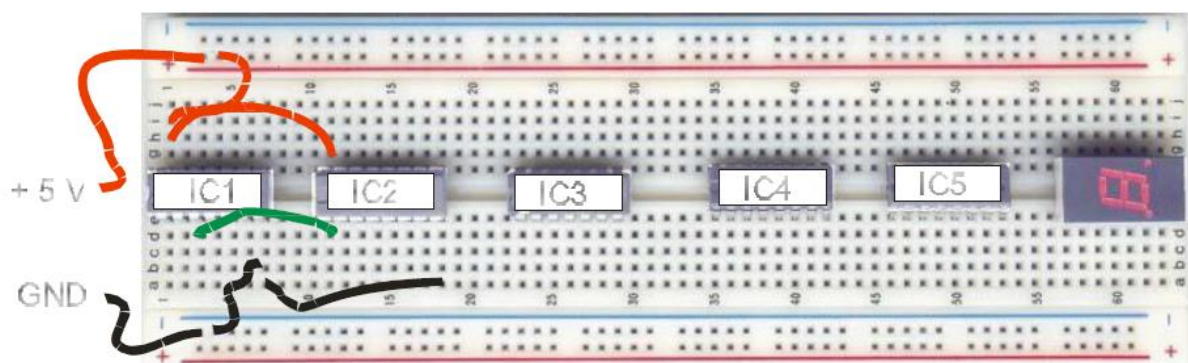


Figure 2: Connecting Wires

Special care is needed when using the protoboard. If a wire which is too large is forced into one of the holes, that particular contact point most probably will be damaged. As a result, the next time the protoboard is used and a wire is inserted into that hole, the wire may not make contact with the internal connection strip and the circuit will not operate properly.

An even more aggravating situation is when the wire makes contact only part of the time. This is called an “intermittent fault.” Since the circuit will operate correctly part of the time, and then mysteriously fail, this type of fault is very hard to find. A good rule to follow is: if the wire doesn't go in easily, find another wire. Pay particular attention to components such as resistors, capacitors - etc., since many have lead with diameters which allow insertion into the protoboard, but still cause damage to the contacts.

To avoid damage, do not insert wires too far.

You will have to strip the wire leads before using them to interconnect circuits on your protoboards. The proper way to do this is to use a pair of wire cutters to carefully strip $\frac{1}{4}$ inch of insulation off of each end of the wire, taking care not to nick the copper wire. If you take more than a quarter inch off, you risk having wire exposed above the protoboard, which will cause a short if it touches another lead or IC pin. If you cut less than a quarter off, the wire may not make a good connection within the protoboard hole. It also helps to cut the ends of

the wire at an angle, which produces a point on the end of the wire. The wire will then slide into the protoboard easier.

- **Integrated Circuits (IC's)**

The IC's are main components for DSD lab. Each looks similar to the one shown below:

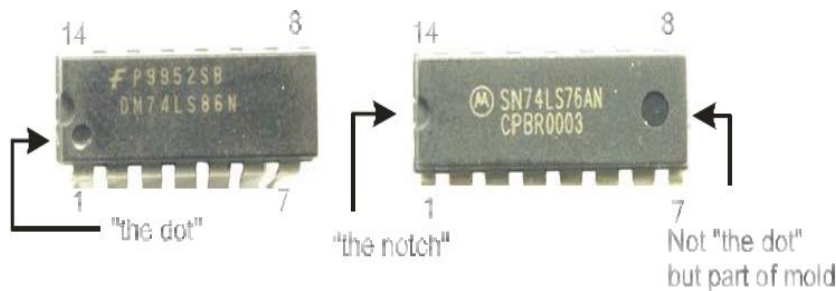


Figure 3: Physical view of IC

Note the numbering system, which progresses counter-clockwise from pin 1. Pin 1 is designated by a dot or a notch. Usually on TTL logic, the last pin is to be connected to the +5 V supply, and the pin diagonally opposite is connected to ground. These are pins 14 and 7, respectively, for 14-pin IC's and 16 and 8 for 16-pin IC's.

There are some exceptions to this rule, so always check!

Special caution should be taken when inserting and removing IC's from the protoboard.

When shipped from the factory, the leads (legs) of the IC's are slightly bent apart to aid in machine insertion. It is necessary to straighten the legs prior to insertion into the protoboards.

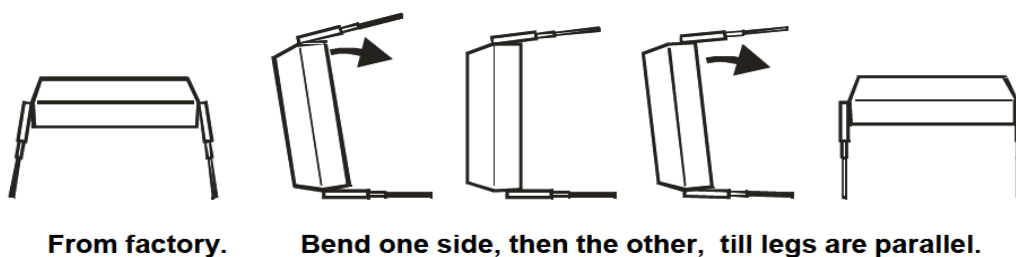


Figure 4: Bending in ICs

When you remove a chip (IC) from the protoboard, it is very easy to bend the legs by not exercising caution. You should not simply pull on the chip with two fingers to remove it. One end will invariably rise before the other causing the legs on the other end to bend. It may even result in a puncture wound to one of your fingers because the legs of the IC's are very

sharp. Be careful! If an IC extractor is not available, you should use the tip of a pen or pencil to gently pry up the legs on one end of the chip, and then the other, as shown below. Usually, if a leg is bent twice, it will break off easily and the IC is virtually useless so.

GOOD LAB PROCEDURES

In order to make your experiments go easier, there are some procedures which should be followed. Most are generally common sense.

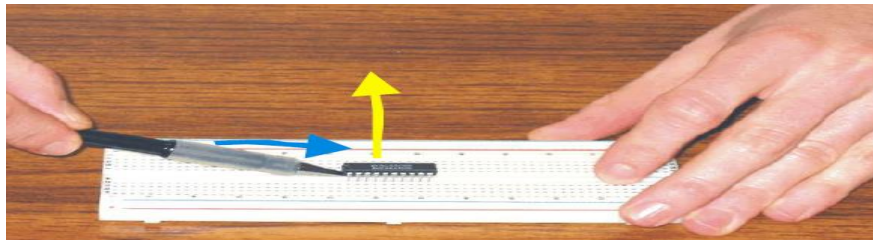


Figure 5: Correct way of removing any IC from breadboard

- Before wiring a circuit, a circuit diagram should be drawn and simulated. This diagram should include pin numbers as shown below.

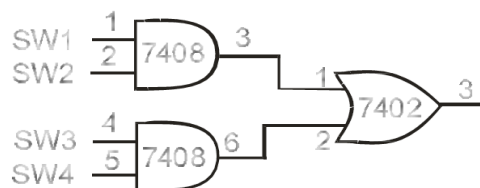


Figure 6: Circuit connection diagram including IC pin numbers

You can then check off each connection as it is wired. For example, after the top AND gate is connected to the OR gate on the right, the line connecting them in the diagram should be checked off as having been wired. Without some system like this, it is very easy to forget which gate is which within a system.

- While wiring, rewiring, etc. *turn off the power*. This prevents the application of power to the circuit in unwanted places.
- Try to avoid messy "rat's nest" wiring. It is almost impossible to trouble-shoot a messy wiring job. It is also hard to make changes to a disorganized board. Keep lead wires as short as possible, and make neat, flat bends. It is also smart not to wire across the top of the IC's.

Wire the power leads of chips first, using a color scheme if possible.

Traditionally, **RED** = +5 Volts and **BLACK** = Ground

Experiment No. – 1

Aim: To verify the truth tables of logic gates: AND, OR, NOT, NAND, NOR, Ex-OR and Ex-NOR

Apparatus / Component Required: IC tester, Digital trainer kit and Digital ICs

S.No.	COMPONENT	SPECIFICATION
1.	2 i/p AND GATE	IC 7408
2.	2 i/p OR GATE	IC 7432
3.	NOT GATE	IC 7404
4.	2 i/p XOR GATE	IC 7486
5.	2 i/p NAND GATE	IC 7400
6.	2 i/p NOR GATE	IC 7402
7.		IC 7410
8.		IC 7411
9.	3 i/p NAND GATE	IC 7427
	3 i/p AND GATE	
	3 i/p NOR GATE	
10.	4 i/p NAND GATE	IC 7420
11.		IC 7421
12.	4 i/p AND GATE	IC 7425
	4 i/p NOR GATE	
13.	2 i/p Ex-NOR GATE	IC74266

Theory:

BASIC GATES:

The Digital Logic "AND" Gate

A Logic AND Gate is a type of digital logic gate that has an output which is normally at logic level "0" and only goes "HIGH" to a logic level "1" when ALL of its inputs are at logic level "1". The output of a Logic AND Gate only returns "LOW" again when ANY of its inputs are at a logic level "0". The logic or Boolean expression given for a logic AND gate is that for **Logical Multiplication** which is denoted by a single dot or full stop symbol, (.) giving us the Boolean expression of: $A.B = Q$.

Then we can define the operation of a 2-input logic AND gate as being:

"If both A and B are true, then Q is true"

2-input AND Gate


Symbol	Truth Table		
 <p style="text-align: center;">2-input AND Gate</p>	B	A	Q
	0	0	0
	0	1	0
	1	0	0
	1	1	1
Boolean Expression $Q = A.B$	Read as A AND B gives Q		

Figure 1.1: Logic symbol and truth table of AND gate

Commonly available digital logic AND gate IC's included:

TTL Logic Types

- 74LS08 Quad 2-input
- 74LS11 Triple 3-input
- 74LS21 Dual 4-input

CMOS Logic Types

- CD4081 Quad 2-input
- CD4073 Triple 3-input
- CD4082 Dual 4-input

Quad 2-Input AND Gate IC 7408

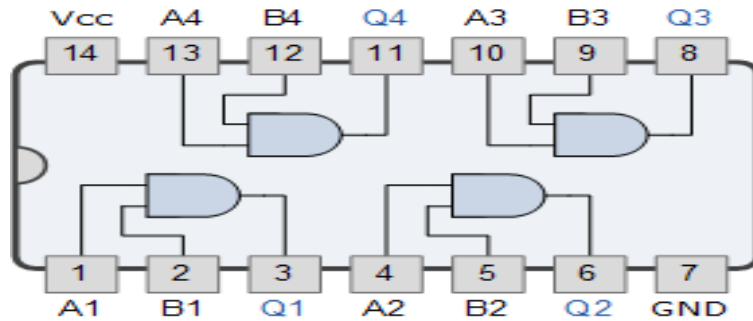


Figure 1.2: Pin diagram of IC 7408

Triple 3-input AND Gate IC 7411

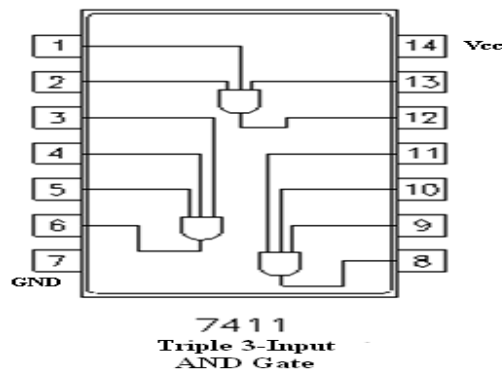


Figure 1.3: Pin diagram of IC 7411

Dual 4-input AND Gate IC 7421

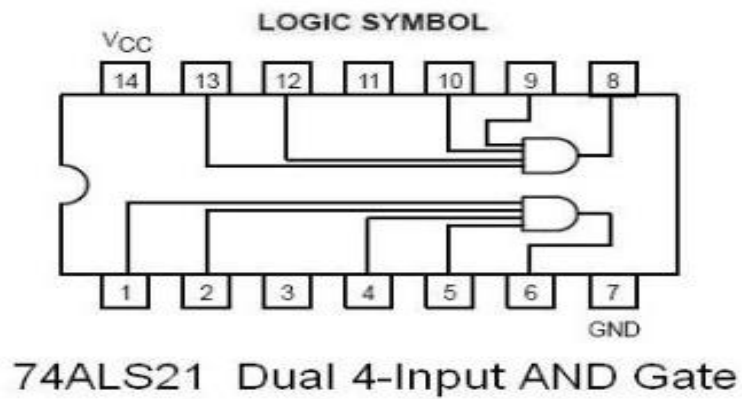


Figure 1.4: Pin diagram of IC 7421

The Logic "OR" Gate

A Logic OR Gate or Inclusive-OR gate is a type of digital logic gate that has an output which is normally at logic level "0" and only goes "HIGH" to a logic level "1" when ANY of its inputs are at logic level "1". The output of a Logic OR Gate only returns "LOW" again when ALL of its inputs are at a logic level "0". The logic or Boolean expression given for a logic OR gate is that for *Logical Addition* which is denoted by a plus sign, (+) giving us the Boolean expression of: $A+B = Q$.

Then we can define the operation of a 2-input logic OR gate as being:

"If either A or B is true, then Q is true"

2-input OR Gate

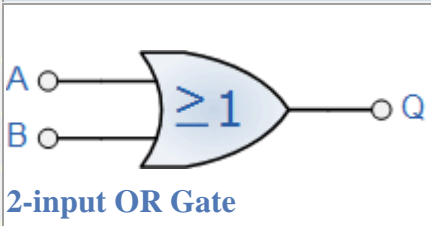
Symbol	Truth Table		
 <p>2-input OR Gate</p>	B	A	Q
	0	0	0
	0	1	1
	1	0	1
	1	1	1
Boolean Expression $Q = A+B$	Read as A OR B gives Q		

Figure 1.5: Logic symbol and truth table of OR Gate

Commonly available OR gate IC's included:

TTL Logic Types

- 74LS32 Quad 2-input

CMOS Logic Types

- CD4071 Quad 2-input
- CD4075 Triple 3-input
- CD4072 Dual 4-input

Quad 2-Input OR Gate IC 7432

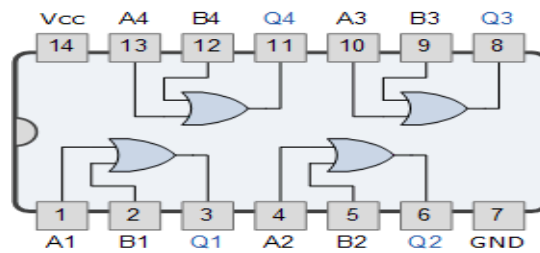


Figure 1.6: Pin diagram of IC 7432

The Digital Logic "NOT" Gate (Digital Inverter)

The digital Logic NOT Gate is the most basic of all the logical gates and is sometimes referred to as an Inverting Buffer or simply a Digital Inverter. It is a single input device which has an output level that is normally at logic level "1" and goes "LOW" to a logic level "0" when its single input is at logic level "1", in other words it "inverts" (complements) its input signal. The output from a NOT gate only returns "HIGH" again when its input is at logic level "0" giving us the Boolean expression of: $A = Q$.

Then we can define the operation of a single input logic NOT gate as being:

"If A is NOT true, then Q is true"

Symbol	Truth Table	
<p style="text-align: center;">Inverter or NOT Gate</p>	A	Q
	0	1
	1	0
Boolean Expression $Q = \text{not } A$ or A	Read as inverse of A gives Q	

Figure 1.7: Logic symbol and truth table of OR gate

Then, with an input voltage at "A" HIGH, the output at "Q" will be LOW and an input voltage at "A" LOW the Resulting output voltage at "Q" is HIGH producing the complement of the input signal. Commonly available logic NOT gate and Inverter IC's include

TTL Logic Types

- 74LS04 Hex Inverting NOT Gate
- 74LS04 Hex Inverting NOT Gate
- 74LS14 Hex Schmitt Inverting NOT Gate
- 74LS1004 Hex Inverting Drivers

CMOS Logic Types

- CD4009 Hex Inverting NOT Gate
- CD4069 Hex Inverting NOT Gate

Hex Inverter Gate IC 7404

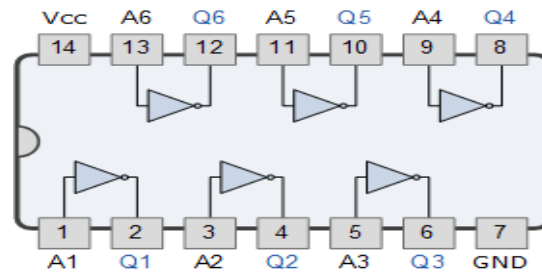


Figure 1.8: Pin diagram of IC 7404

UNIVERSAL GATES:

The Logic "NAND" Gate

The Logic NAND Gate is a combination of the digital logic AND gate with that of an inverter or NOT gate connected together in series. The NAND (Not - AND) gate has an output that is normally at logic level "1" and only goes "LOW" to logic level "0" when ALL of its inputs are at logic level "1". The Logic NAND Gate is the reverse or "Complementary" form of the AND gate we have seen previously.

Logic NAND Gate Equivalence

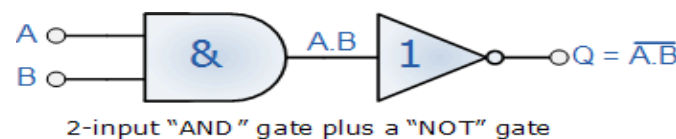


Figure 1.9: Equivalent circuit of NAND gate using AND and NOT gate

The logic or Boolean expression given for a logic NAND gate is that for Logical Addition, which is the opposite to the AND gate, and which it performs on the complements of the inputs. The Boolean expression for a logic NAND gate is denoted by a single dot or full stop symbol, (.) with a line or Overline, (¯) over the expression to signify the NOT or logical negation of the NAND gate giving us the Boolean expression of: $A.B = Q$. Then we can define the operation of a 2-input logic NAND gate as being:

"If either A or B are NOT true, then Q is true"

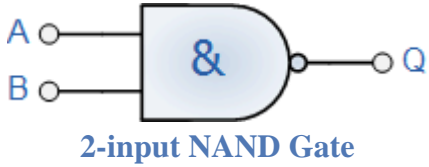
Symbol	Truth Table		
 <p>2-input NAND Gate</p>	B	A	Q
	0	0	1
	0	1	1
	1	0	1
	1	1	0
Boolean Expression $Q = A \cdot B$	Read as A AND B gives NOT Q		

Figure 1.10: Logic symbol and truth table of NAND gate

Commonly available logic NAND gate IC's including:

TTL Logic Types

- 74LS00 Quad 2-input
- 74LS10 Triple 3-input
- 74LS20 Dual 4-input
- 74LS30 Single 8-input

CMOS Logic Types

- CD4011 Quad 2-input
- CD4023 Triple 3-input
- CD4012 Dual 4-input

Quad 2-Input NAND Gate IC 7400

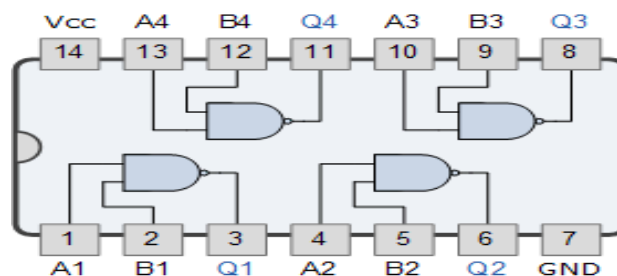


Figure 1.11: Pin diagram of IC 7400

Triple 3-Input NAND Gate IC 7410

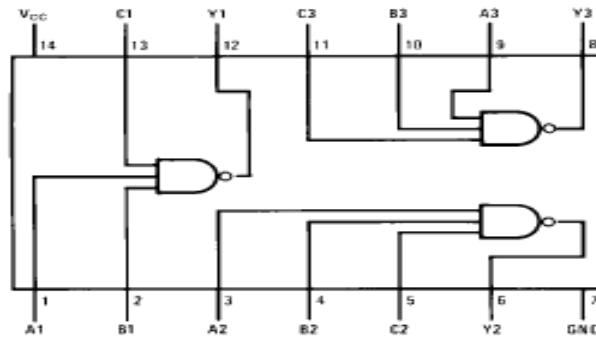


Figure 1.12: Pin diagram of IC 7410

Dual 4-Input NAND Gate IC 7420

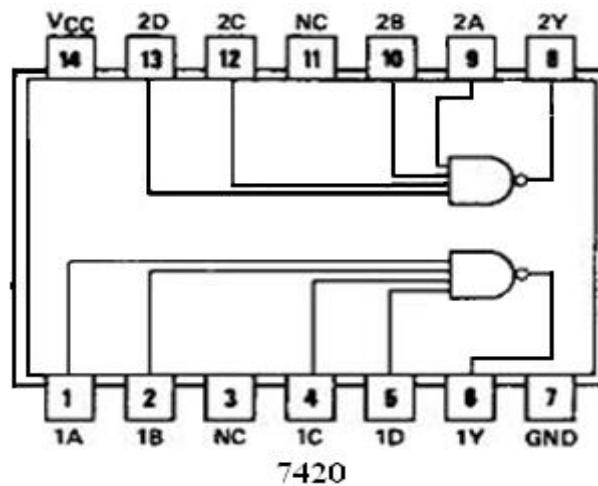
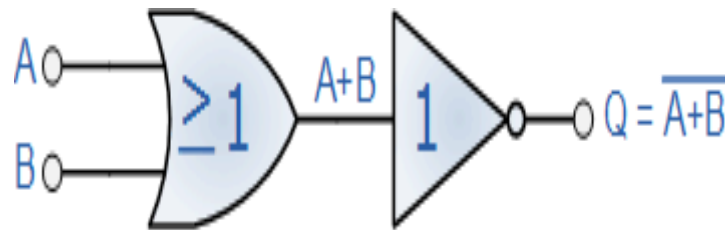


Figure 1.13: Pin diagram of IC 7420

The Logic "NOR" Gate

The Logic NOR Gate or Inclusive-NOR gate is a combination of the digital logic OR gate with that of an inverter or NOT gate connected together in series. The NOR (Not - OR) gate has an output that is normally at logic level "1" and only goes "LOW" to logic level "0" when ANY of its inputs are at logic level "1". The Logic NOR Gate is the reverse or "Complementary" form of the OR gate we have seen previously.

NOR Gate Equivalent



2-input "OR" gate plus a "NOT" gate

Figure 1.14: Equivalent circuit of NOR gate using OR and NOT gate

The logic or Boolean expression given for a logic NOR gate is that for Logical Multiplication which it performs on the complements of the inputs. The Boolean expression for a logic NOR gate is denoted by a plus sign, (+) with a line or Over line, (¯) over the expression to signify the NOT or logical negation of the NOR gate giving us the Boolean expression of: $Q = \overline{A+B}$. Then we can define the operation of a 2-input logic NOR gate as being:

"If both A and B are NOT true, then Q is true"

Symbol	Truth Table		
<p>2-input NOR Gate</p>	B	A	Q
	0	0	1
	0	1	0
	1	0	0
	1	1	0
Boolean Expression $Q = \overline{A+B}$	Read as A OR B gives NOT Q		

Figure 1.15: Logic symbol and truth table of NOR gate

Commonly available NOR gate IC's include:

TTL Logic Types

- 74LS02 Quad 2-input
- 74LS27 Triple 3-input
- 74LS260 Dual 4-input

CMOS Logic Types

- CD4001 Quad 2-input
- CD4025 Triple 3-input
- CD4002 Dual 4-input

Quad 2-Input NOR Gate 7402

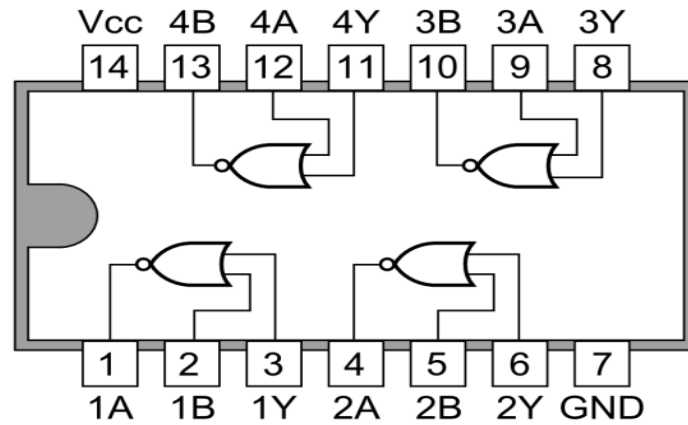


Figure 1.16: Pin diagram of IC 7402

Triple 3-Input NOR Gate IC 7427

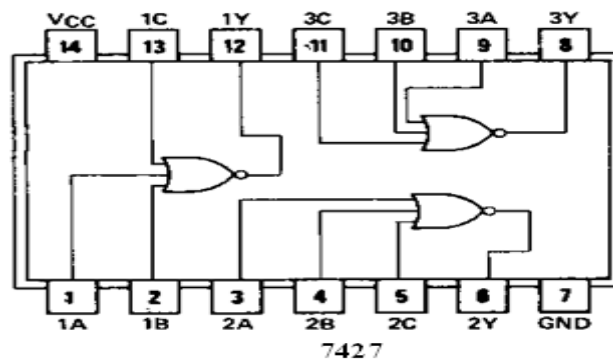


Figure 1.17: Pin diagram of IC 7427

Dual 4-Input NOR Gate IC 7425

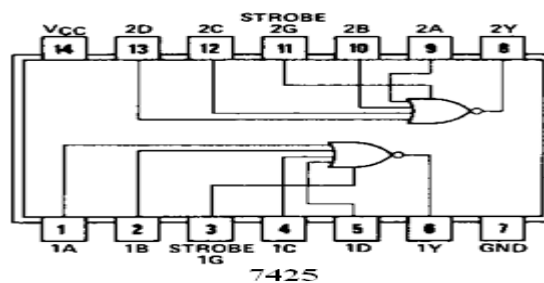


Figure 1.18: Pin diagram of IC 7425

The Exclusive-OR Gate (Ex-OR Gate)

The output of an Exclusive-OR gate ONLY goes "HIGH" when its two input terminals are at "DIFFERENT" logic levels with respect to each other and they can both be at logic level "1" or both at logic level "0" giving us the Boolean expression of: $Q = A'B + AB'$. The Exclusive-OR Gate function is achieved by combining standard gates together to form more complex gate functions. An example of a 2-input Exclusive-OR gate is given below.

2-input Ex-OR Gate

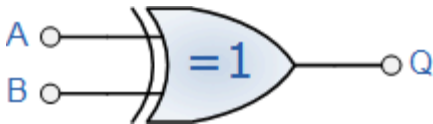
Symbol	Truth Table	
 <p>2-input Ex-OR Gate</p>	B	A Q
	0	0 0
	0	1 1
	1	0 1
	1	1 0
Boolean Expression $Q = A \oplus B$	Read as A OR B but NOT BOTH gives Q	
<u>TTL Logic Types</u>		<u>CMOS Logic Types</u>
<ul style="list-style-type: none"> • 74LS86 Quad 2-input 		<ul style="list-style-type: none"> • CD4030 Quad 2-input

Figure 1.19: Logic symbol and truth table of XOR gate

Quad 2-Input Ex-OR Gate IC 7486

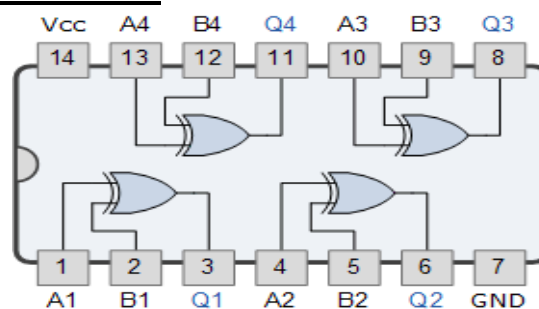


Figure 1.20: Pin diagram of IC 7486

The Exclusive-NOR Gate

The output of an Exclusive-NOR gate ONLY goes "HIGH" when its two input terminals, A and B are at the "SAME" logic level which can be either at a logic level "1" or at a logic level "0". Then this type of gate gives an output "1" when its inputs are "logically equal" or "equivalent" to each other, which is why an Exclusive-NOR gate is sometimes called an Equivalence Gate.

Ex-NOR Gate Equivalent



2-input "Ex-OR" gate plus a "NOT" gate

Figure 1.21: Equivalent circuit of XNOR gate using XOR – NOT gate

Symbol	Truth Table		
<p>2-input Ex-NOR Gate</p>	B	A	Q
	0	0	1
	0	1	0
	1	0	0
	1	1	1
Boolean Expression $Q = A \text{ XNOR } B$	Read if A AND B the SAME gives Q		

Figure 1.22: Logic symbol and truth table of XNOR gate

In general, an Exclusive-NOR gate will give an output value of logic "1" ONLY when there are an EVEN number of 1's on the inputs to the gate (the inverse of the Ex-OR gate) except when all its inputs are "LOW". Commonly available Exclusive-NOR gate IC's include:

TTL Logic Types

74LS266 Quad 2-input

CMOS

Logic Types

CD4077 Quad 2-input

Quad 2-Input Ex-NOR Gate IC 74267

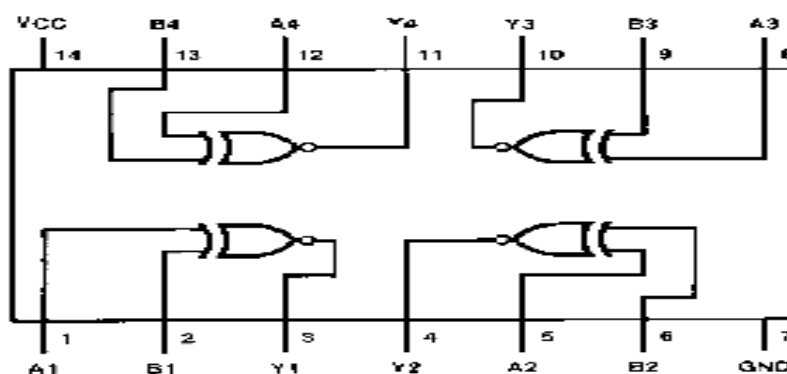


Figure 1.23: Pin diagram of IC 74267

Procedure:

- Test the digital ICs with the help of IC tester.
- Make the connections on the breadboard as per the pin diagram. Connect the inputs of any logic gate to the logic sources and its output to the logic indicator.
- Apply various input combinations and observe output for each one.
- Verify the truth table for each input/ output combination.
- Repeat the process for all other logic gates.

Output: The truth tables of logic gates: AND, OR, NOT, NAND, NOR Ex-OR and Ex-NOR are verified using their ICs.

Result: In conclusion, each basic gate works in unique way, which is proved during this experiment. We used the truth table to examine the operation of the basic logic gate. It is proved from experiment that logic gates work in basis of Boolean Algebra. AND Gate, OR Gate and NOT Gate are the basic gates. All the combinational logic gates are made of these three basic gates. Output from one logic gate can be used as input for another logic gate to form combinational logic gate. So, we have studied how logic gates work on the basis of Boolean algebra.

Discussions:

Q1. Define the term digital and analog.

Q2. What is IC?

Q3. Define the Basic and universal gates.

Q4. If a 3-input NOR gate has eight input possibilities, how many of those possibilities will Result in a HIGH output?

Q5. Which of the logical operations is represented by the + sign in Boolean algebra?

Q5. How many no. of input variables can a NOT Gate have?

Experiment No – 2

Aim: To verify the truth table of OR, AND, NOR, NAND, NOT, Ex-OR and Ex-NOR logic gates realized using NAND & NOR gates.

Apparatus / Component Required: IC tester, Digital trainer kit, Digital ICs (7400, 7402)

Theory:

Universal Gates (NAND & NOR Gates):

A universal gate is a gate which can implement any Boolean function without need to use any other gate type. The NAND and NOR gates are universal gates. This is advantageous since NAND and NOR gates are economical and easier to fabricate.

NAND Gate is a Universal Gate:

To prove that any Boolean function can be implemented using only NAND gates, we will show that the AND, OR, and NOT operations can be performed using only these gates.

Implementing an Inverter Using only NAND Gate

The figure shows two ways in which a NAND gate can be used as an inverter (NOT gate).

1. All NAND input pins connect to the input signal A gives an output A'.

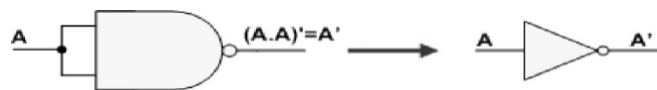


Figure 2.1

2. One NAND input pin is connected to the input signal A while all other input pins are connected to logic 1. The output will be A'.

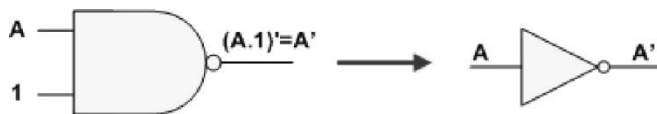


Figure 2.2

Implementing AND Using only NAND Gates

3. An AND gate can be replaced by NAND gates as shown in the figure (The AND is replaced by a NAND gate with its output complemented by a NAND gate inverter).

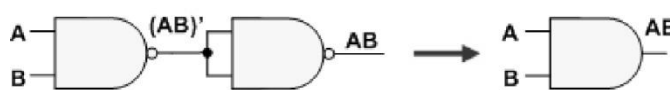


Figure 2.3

Implementing OR Using only NAND Gates

4. An OR gate can be replaced by NAND gates as shown in the figure (The OR gate is replaced by a NAND gate with all its inputs complemented by NAND gate inverters).

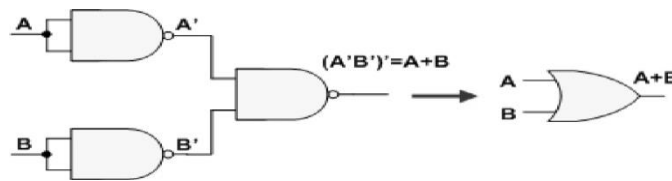


Figure 2.4

Thus, the NAND gate is a universal gate since it can implement the AND, OR and NOT functions.

Implementing NOR Using only NAND Gates

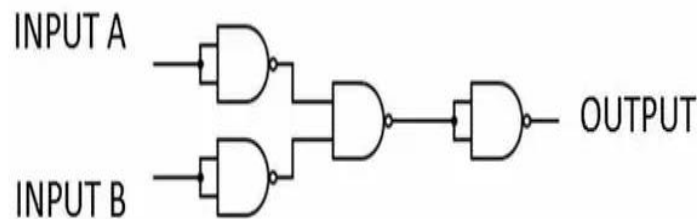
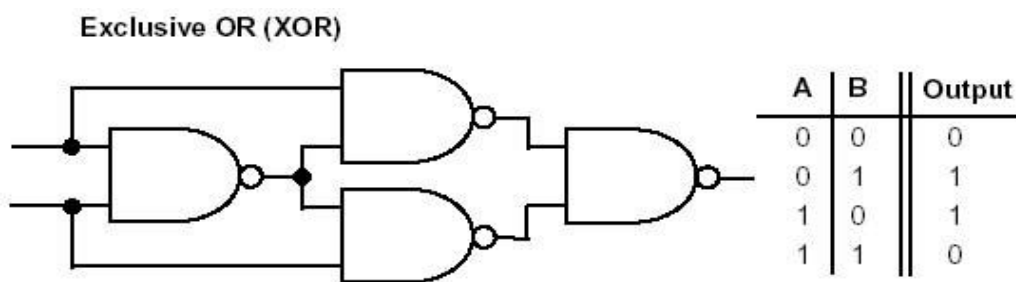


Figure 2.5

Implementing EX-OR Using only NAND Gates



A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

Figure 2.6

Implementing EXNOR Using only NAND Gates

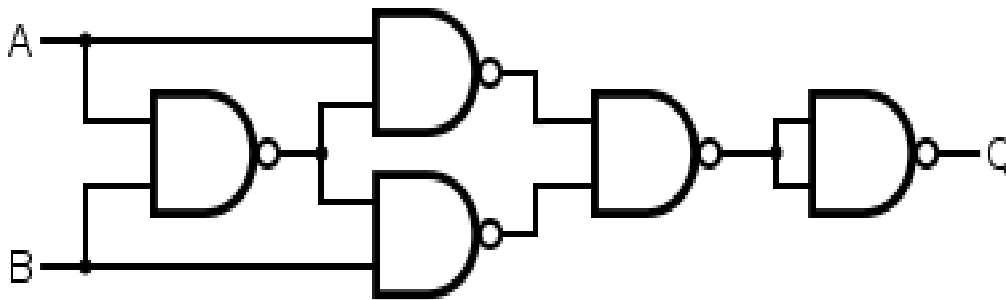


Figure 2.7

NOR Gate is a Universal Gate:

To prove that any Boolean function can be implemented using only NOR gates, we will show that the AND, OR, and NOT operations can be performed using only these gates.

Implementing an Inverter Using only NOR Gate

The figure shows two ways in which a NOR gate can be used as an inverter (NOT gate).

1. All NOR input pins connect to the input signal **A** gives an output **A'**.

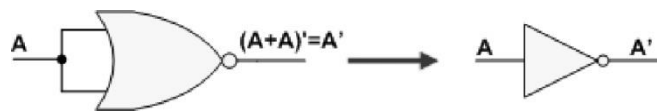


Figure 2.8

2. One NOR input pin is connected to the input signal **A** while all other input pins are connected to logic **0**. The output will be **A'**.

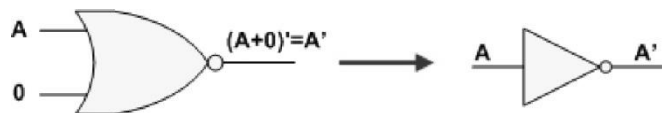


Figure 2.9

Implementing OR Using only NOR Gates

An OR gate can be replaced by NOR gates as shown in the figure (The OR is replaced by a NOR gate with its output complemented by a NOR gate inverter)

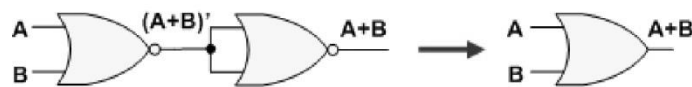


Figure 2.10

Implementing AND Using only NOR Gates

An AND gate can be replaced by NOR gates as shown in the figure (The AND gate is replaced by a NOR gate with all its inputs complemented by NOR gate inverters)

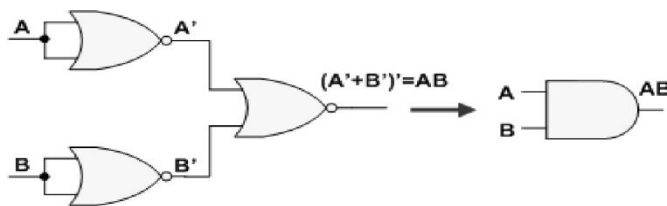


Figure 2.11

Thus, the NOR gate is a universal gate since it can implement the AND, OR and NOT functions.

Implementing EX-NOR Using only NOR Gates

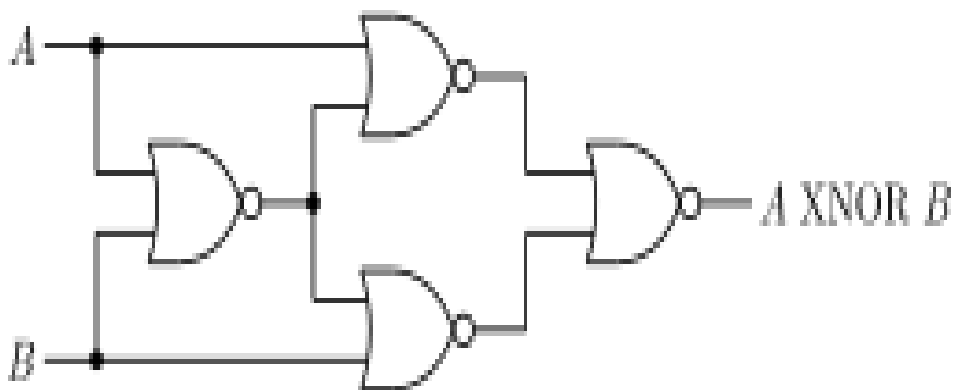


Figure 2.12

Implementing EXOR Using only NOR Gates

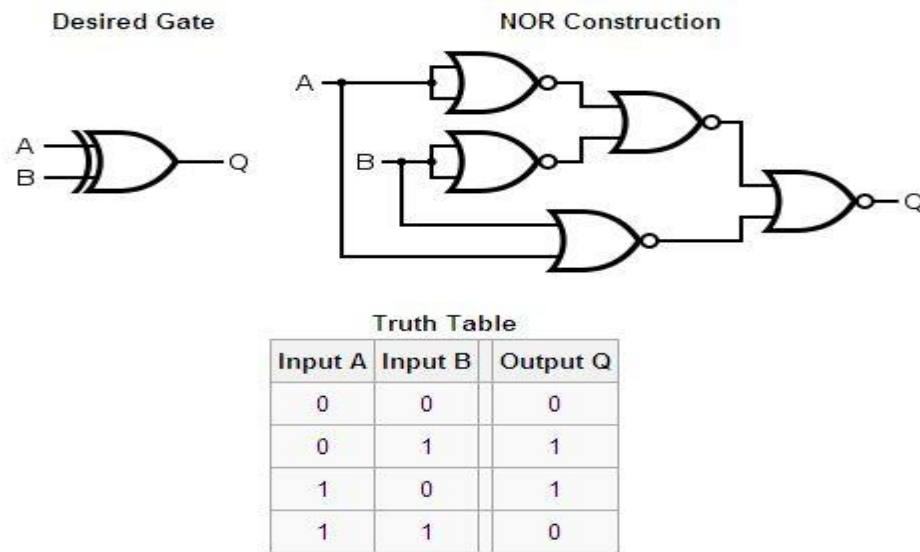


Figure 2.13: Logic Diagram & truth table of XOR gate

Implementing NAND Using only NOR Gates

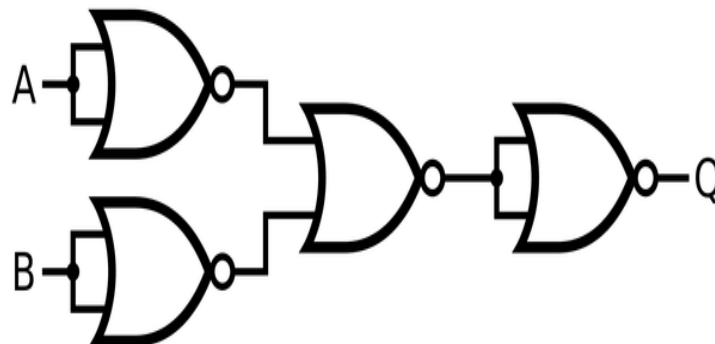


Figure 2.14

Procedure:

- Test the digital ICs (7400, 7402) with the help of IC tester.
- Make the connections on breadboard as per the pin diagram. Connect the inputs of the logic gate to the logic sources and its output to the logic indicator.
- Apply various input combinations and observe output for each one.
- Verify the truth table for each input/ output combination.

Output: The truth tables of logic gates: AND, OR, NOT, NAND, NOR Ex-OR and Ex-NOR are verified using the universal gate ICs 7400 and IC 7402.

Result: The universal gates are the gates which can implement any Boolean function without need to use any other gate type. Hence all the gates are realized using Universal gates. In practice, this is advantageous since NAND and NOR gates are economical and easier to fabricate and are the mostly used gates in digital logic families.

Discussion:

Q1. Why NAND & NOR gates are called universal gates?

Q2. Which gate is equal to AND-inverter Gate?

Q3. Which gate is equal to OR-inverter Gate?

Q4. Draw $X = ab + b'a'$ by NAND gate?

Q5. What is Demorgan's theorem?

Q6. Solve following example by Demorgan's theorem

1) $(A+B+C)'$ 2) $(ABC)'$ $(AB)'$ 3) $(A+A) A = ?$

Experiment No. – 3

Aim: To realize SOP and POS expressions.

Apparatus / Component Required: IC tester, Digital trainer kit, Digital ICs (7400, 7402, 7408, 7432).

Theory:

Sum of Product (SOP)

The sum-of-products (SOP) form is a method (or form) of simplifying the Boolean expressions of logic gates. In this SOP form of Boolean function representation, the variables are operated by AND (product) to form a product term and all these product terms are ORed (summed or added) together to get the final function. A sum-of-products form can be formed by adding (or summing) two or more product terms using a Boolean addition operation. Here the product terms are defined by using the AND operation and the sum term is defined by using OR operation.

The sum-of-products form is also called as Disjunctive Normal Form as the product terms are ORed together and Disjunction operation is logical OR. Sum-of-products form is also called as Standard SOP.

SOP form representation is most suitable to use them in FPGA (Field Programmable Gate Arrays).

Product of Sums (POS)

The product of sums form is a method (or form) of simplifying the Boolean expressions of logic gates. In this POS form, all the variables are ORed, i.e. written as sums to form sum terms. All these sum terms are ANDED (multiplied) together to get the product-of-sum form. This form is exactly opposite to the SOP form. So this can also be said as “Dual of SOP form”.

Here the sum terms are defined by using the OR operation and the product term is defined by using AND operation. When two or more sum terms are multiplied by a Boolean OR operation, the resultant output expression will be in the form of product-of-sums form or POS form.

The product-of-sums form is also called as Conjunctive Normal Form as the sum terms are ANDED together and Conjunction operation is logical AND. Product-of-sums form is also called as Standard POS.

Implement the following SOP function

$$F = XZ + \bar{Y}Z + \bar{X}YZ$$

Being an SOP expression, it is implemented in 2-levels as shown in the figure. Here the expression is realized using NAND Gate.

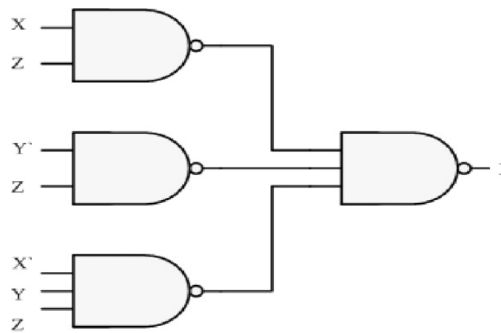


Figure 3.1: SOP using NAND gate

Implement the following POS function

$$F = (X + Z)(\bar{Y} + Z)(\bar{X} + Y + Z)$$

Being a POS expression, it is implemented in 2-levels as shown in the figure.

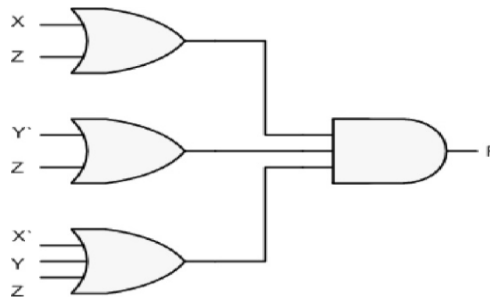


Figure 3.2: POS using OR & AND gate

Introducing two successive inverters at the inputs of the AND gate Results in the shown equivalent implementation. Since two successive inverters on the same line will not have an overall effect on the logic as it is shown before.

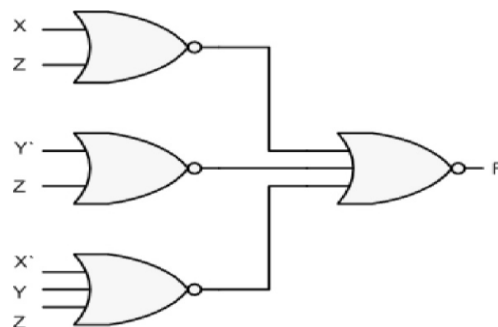


Figure 3.3: POS using NOR gate

Procedure:

- Test the digital ICs with the help of IC tester.
- Make the connections on the breadboard as per the pin diagram and circuit. Connect the inputs of the logic gate to the logic sources and its output to the logic indicator.
- Apply various input combinations and observe output for each one.
- Verify the circuit for each input/ output combination.

Output: We have implemented the given Boolean function using logic gates in both SOP and POS forms.

Result: In the experiment, De-Morgan's theorem and postulate of Boolean algebra are verified. Also the Sum of products and product of sum expressions are realized using the basic gates and the universal gates.

Discussion:

Q1. Express the function $f(x, y, z) = 1$ in the form of sum of minterms and a product of maxterms.

Q2. What is D'morgans theorem?

Q3. Solve following example by using D'morgans theorem.

$$(ABC)' (AB)'$$

Q4. Examine this truth table and then write both SOP and POS Boolean expressions describing the Output:

A	B	C	output
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Q5. Convert the following SOP expression to an equivalent POS expression.

$$ABC + A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C} + \bar{A}\bar{B}C$$

Q6. Express the function $f(x, y, z) = 1$ in the sum of minterms and a product of maxterms?

Minterms = (0,1,2,3,4,5,6,7), Maxterms = No maxterms.

Q7. Make the truth table of $\pi M(1,2,6,7,13,14,15) + d(0,3,5)$?

Experiment No: 4

Aim: To realize half adder/ Subtractor & Full Adder/ Subtractor using NAND & NOR gates and to verify their truth table.

Apparatus/ Component Required: IC tester, Digital trainer kit, Digital ICs (7400, 7402)

Theory: A very useful combinational logic circuit which can be constructed using just a few basic logic gates and adds together binary numbers is the **Binary Adder** circuit. The Binary Adder is made up from standard AND and Ex-OR gates and allow us to "add" together single bit binary numbers, a and b to produce two outputs, the SUM of the addition and a CARRY called the Carry-out, (**Cout**) bit. One of the main uses for the **Binary Adder** is in arithmetic and counting circuits.

The Half Adder Circuit

1-bit Adder with Carry-Out

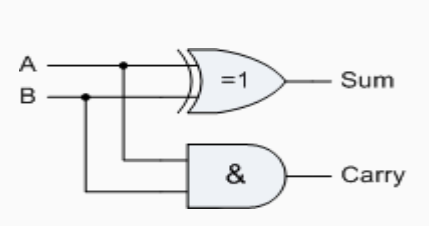
Half Adder		Truth Table			
	A	B	SUM	CARRY	
	0	0	0	0	
	0	1	1	0	
	1	0	1	0	
	1	1	0	1	
Boolean Expression: $Sum = A \oplus B$ $Carry = A \cdot B$					

Figure 4.1: Logic diagram & truth table of half adder

From the truth table we can see that the SUM (S) output is the Result of the Ex-OR gate and the Carry-out (Cout) is the Result of the AND gate. One major disadvantage of the Half Adder circuit when used as a binary adder, is that there is no provision for a "Carry-in" from the previous circuit when adding together multiple data bits. For example, suppose we want to add together two 8-bit bytes of data, any Resulting carry bit would need to be able to "ripple" or move across the bit patterns starting from the least significant bit (LSB).

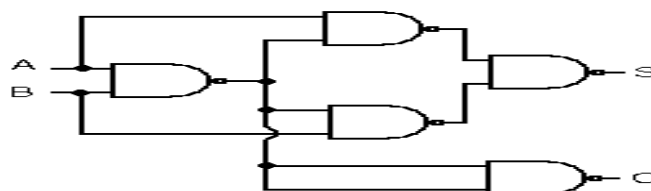


Figure 4.2: Circuit diagram for half-adder using NAND gates

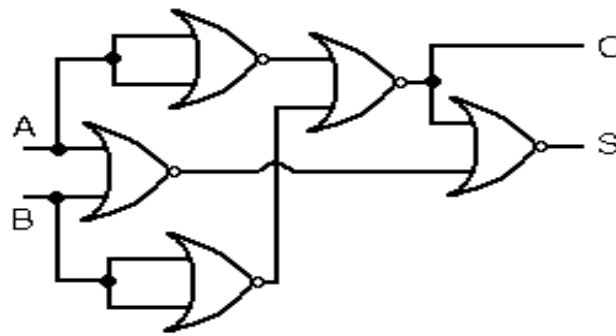


Figure 4.3: Circuit diagram for half-adder using NOR gates

The Full Adder Circuit: The main difference between the **Full Adder** and the previous seen **Half Adder** is that a full adder has three inputs, the same two single bit binary inputs A and B as before plus an additional *Carry-In* (C-in) input as shown below. The 1-bit **Full Adder** circuit above is basically two half adders connected together and consists of three Ex-OR gates, two AND gates and an OR gate, six logic gates in total.

Full Adder	Truth Table				
	A	B	C-in	Sum	C-out
	0	0	0	0	0
	0	1	0	1	0
	1	0	0	1	0
	1	1	0	0	1
	0	0	1	1	0
	0	1	1	0	1
	1	0	1	0	1
	1	1	1	1	1
Boolean Expression: $Sum = A \oplus B \oplus C-in$					

Figure 4.4: Logic diagram & truth table of half adder

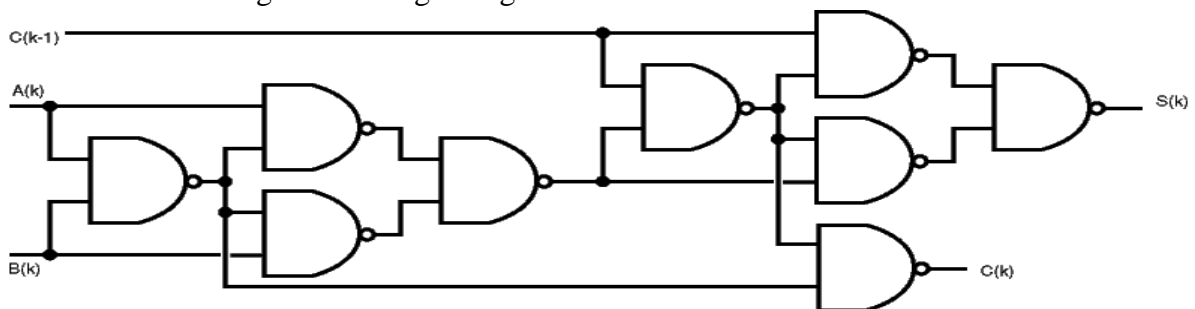


Figure 4.5: Full Adder implementation using NAND Gate

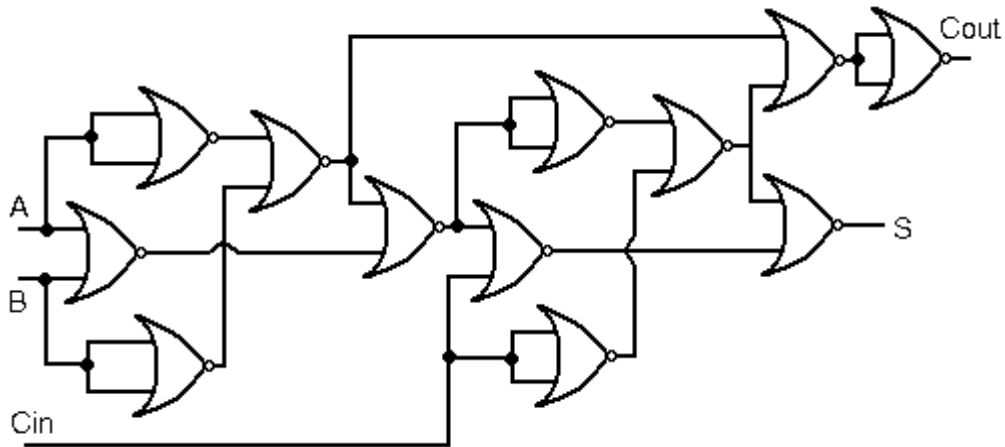


Figure 4.6: Full Adder implementation using NOR Gate

The Half Subtractor Circuit

A half-subtractor has two inputs and two outputs. Let the input variables minuend and subtrahend be designated as A and B respectively, and output functions be designated as DIFF for difference and BORROW for borrow. The truth table of the functions is as follows.

Half Subtractor	Truth Table			
	A	B	DIFF	BORROW
	0	0	0	0
	0	1	1	1
	1	0	0	1
	1	1	0	0
Boolean Expression: $DIFF = A \oplus B$ $BORROW = A \cdot B$				

Figure 4.7: Circuit diagram & truth table of half subtractor

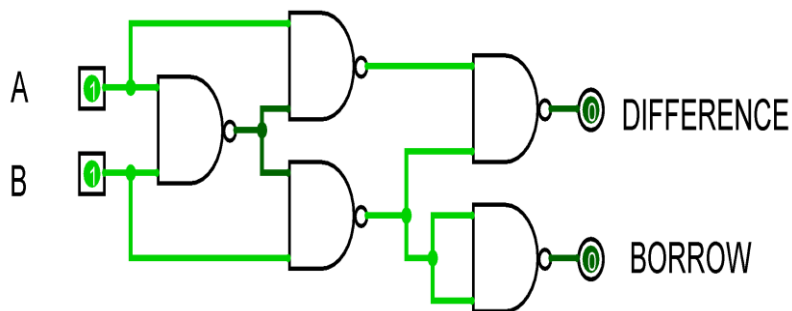


Figure 4.8: Circuit diagram for half-Subtractor using NAND gates.

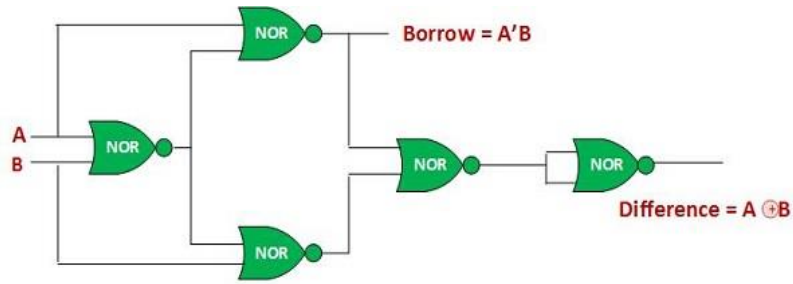


Figure 4.9: Circuit diagram for half-Subtractor using NOR gates.

The Full Subtractor Circuit

A combinational circuit of full-Subtractor performs the operation of subtraction of three bits—the minuend, subtrahend, and borrow generated from the subtraction operation of previous significant digits and produces the outputs difference and borrow. Let us designate the input variables minuend as A, subtrahend as B, and previous borrow as C, and outputs difference as D and B as Borrow. Eight different input combinations are possible for three input variables.

Full Subtractor		Truth Table				
		a	b	C	D	B
		0	0	0	0	0
		0	1	0	1	1
		1	0	0	1	1
		1	1	0	0	1
		0	0	1	1	0
		0	1	1	0	0
		1	0	1	0	0
		1	1	1	1	1
Boolean Expression: $D = a \oplus b \oplus c$; $B = a'b + bc + b'c$						

Figure 4.10: Logic diagram & truth table of Full subtractor

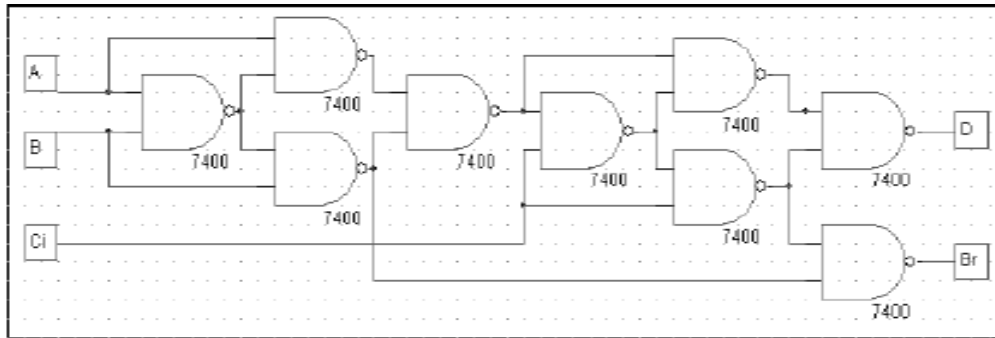


Figure 4.11: Full Subtractor implementation using NAND Gate

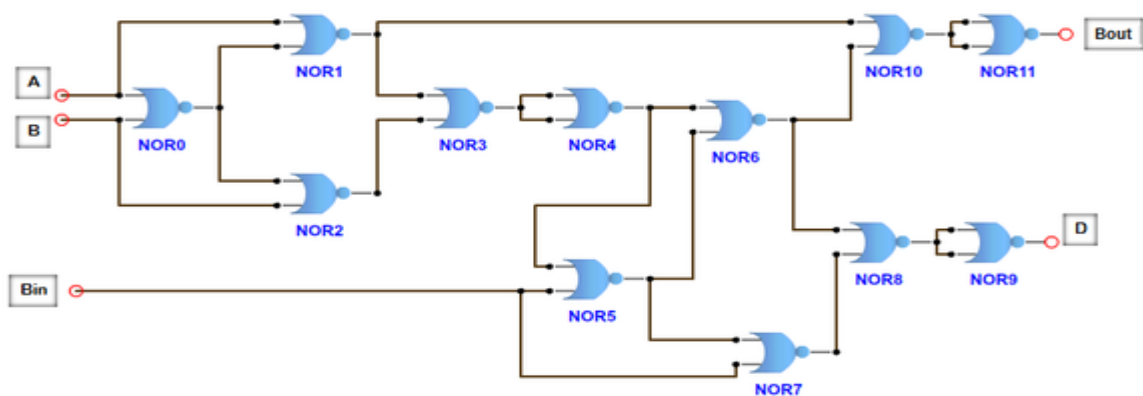


Figure 4.12: Full Subtractor implementation using NOR Gate

Procedure:

1. Insert the IC chips into the breadboard. Point all the chips in the same direction with pin 1 at the upper-left corner. (Pin 1 is often identified by a dot or a notch next to it on the chip package).
2. Connect +5V and GND pins of each IC chips to the power and ground bus strips on the breadboard.
3. Make the connections as per the circuit diagram.
4. Switch on V_{CC} and apply various combinations of input according to truth table.
5. Note down the output readings for half/full adder and sum and the carry bit for different combinations of inputs. Where 5V indicating logic 1 and 0V indicating logic 0.

Output: The truth table for Half adder/Subtractor and Full Adder/Subtractor has been verified.

Result: By using various logic gate ICs we can design full or half adder and subtractor. Half adders/subtractors perform their operation on two bits and gives a two bit output. Full adder/subtractor perform their operation on three bit inputs and gives a two bit output.

Discussion:

- Q1. Define Combinational Circuits.
- Q2. What is the difference between half adder and a full adder?
- Q3. What is meant by two and three variable map?
- Q4. Realize half -subtractor and full-subtractor using NOR gates.
- Q5. Realize a full adder using two half adders.
- Q6. Realize a full subtractors using two half subtractors.

Experiment No: 5

Aim: To design 4-to-1 multiplexer using basic gates and verify the truth table. Also verify the truth table of 8-to-1 multiplexer using IC.

Apparatus / Component Required: IC tester, Digital trainer kit, Digital ICs (74153, 74151, 7408, 7432, 7404, 7411)

Theory:

MULTIPLEXER:

Combinational logic switching devices that operate like a very fast acting multiple position rotary A data selector, more commonly called a Multiplexer, shortened to "Mux" or "MPX", are switch. They connect or control, multiple input lines called "channels" consisting of 2, 4, 8 or 16 individual inputs, one at a time to an output. Then the job of a multiplexer is to allow multiple signals to share a single common output.

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.

BLOCK DIAGRAM FOR 4:1 MULTIPLEXER:

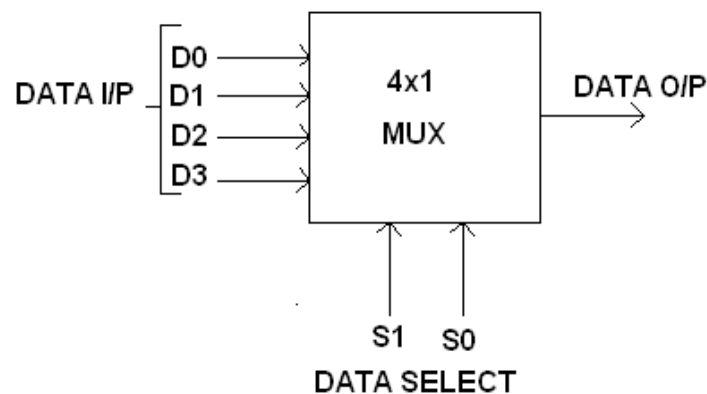


Figure 5.1

TRUTH TABLE:

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

CIRCUIT DIAGRAM FOR MULTIPLEXER:

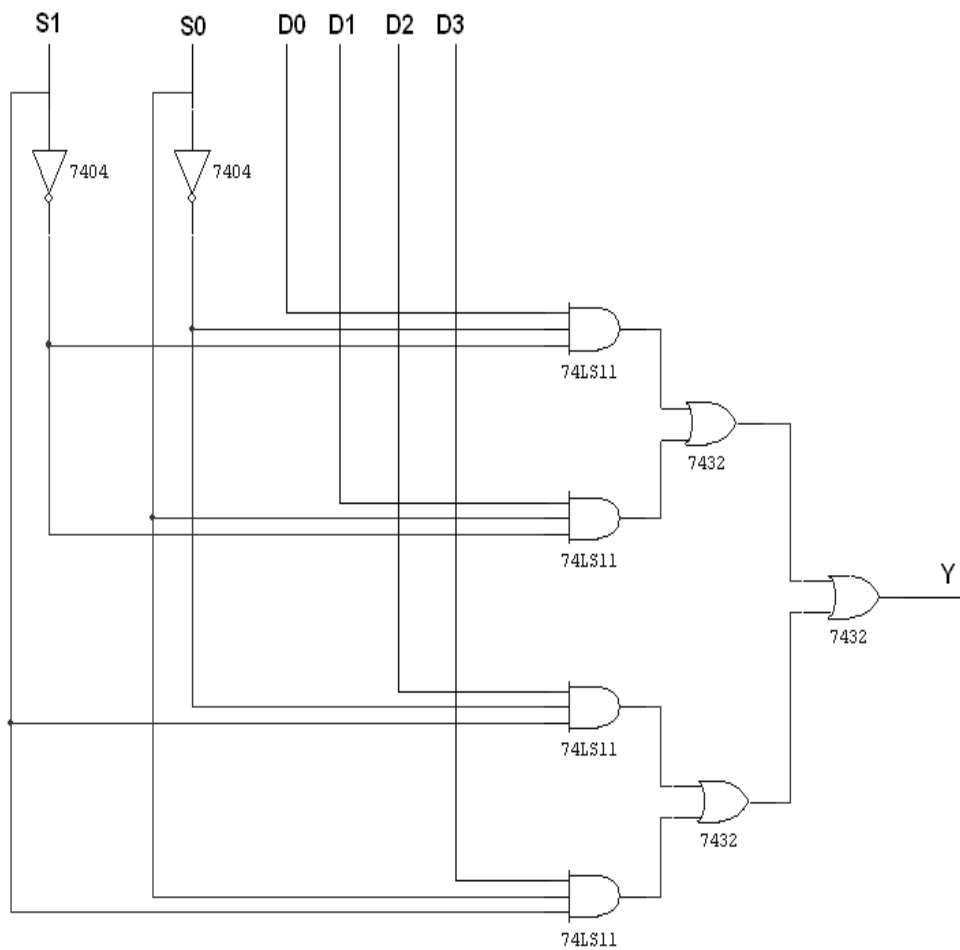


Figure 5.2

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	$X \rightarrow D3 = X S1 S0$

$$Y = X S_1' S_0' + X S_1' S_0 + X S_1 S_0' + X S_1 S_0$$

PIN DIAGRAM FOR IC 74153

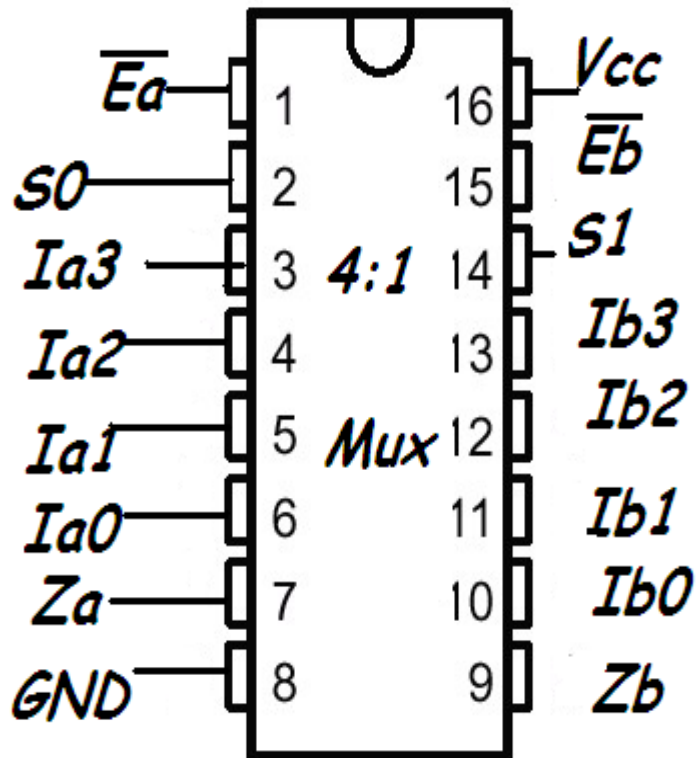


Figure 5.3

TRUTH TABLE OF 4:1 MUX(IC 74153) (CHANNEL A) WITH ACTIVE LOW MODE:

Inputs(Channel A)					Select lines		Output
E_a	I_{a0}	I_{a1}	I_{a2}	I_{a3}	S_0	S_1	Z_a
1	×	×	×	×	×	×	0
0	0	×	×	×	0	0	0
0	1	×	×	×	0	0	1
0	×	0	×	×	0	1	0
0	×	1	×	×	0	1	1
0	×	×	0	×	1	0	0
0	×	×	1	×	1	0	1
0	×	×	×	0	1	1	0
0	×	×	×	1	1	1	1

IMPLEMENTATION OF 8:1 MULTIPLEXER USING 4:1 MUX:

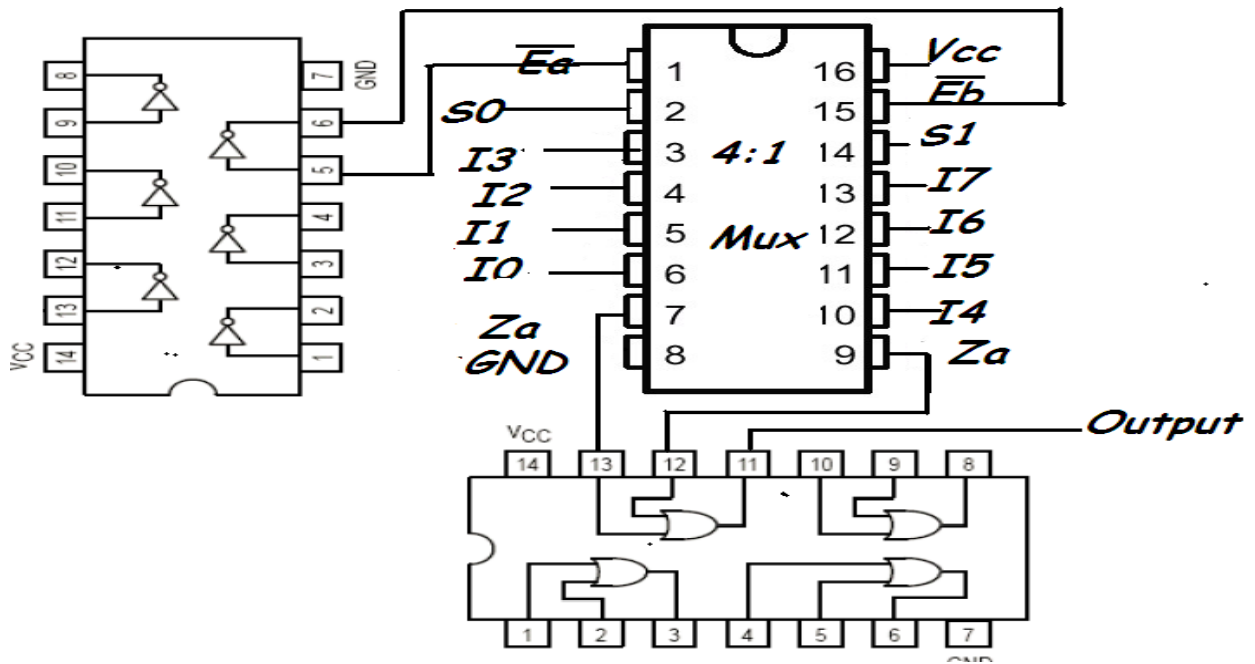


Figure 5.4

CIRCUIT OF 8:1 MUX USING DUAL 4:1 MUX

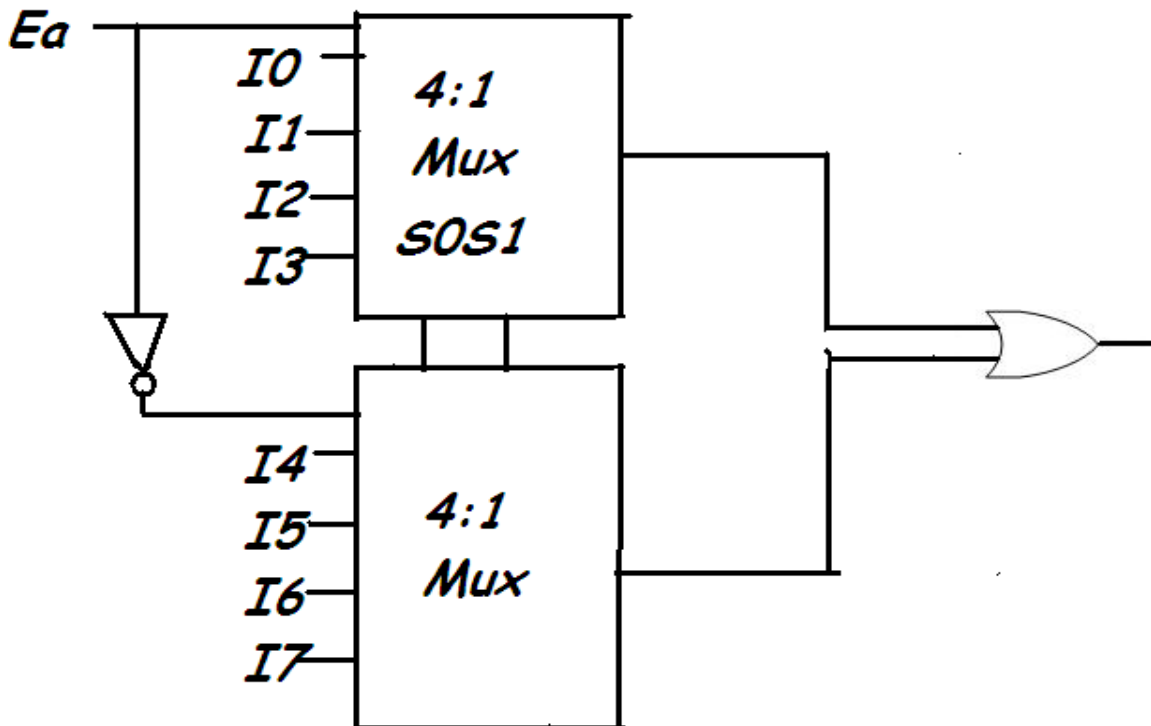


Figure 5.5

TRUTH TABLE OF 8:1 MUX

Input				Outputs	
Select			Enable		
C	B	A	\overline{EN}	Y	\overline{Y}
x	x	x	1	0	1
0	0	0	0	D ₀	$\overline{D_0}$
0	0	1	0	D ₁	$\overline{D_1}$
0	1	0	0	D ₂	$\overline{D_2}$
0	1	1	0	D ₃	$\overline{D_3}$
1	0	0	0	D ₄	$\overline{D_4}$
1	0	1	0	D ₅	$\overline{D_5}$
1	1	0	0	D ₆	$\overline{D_6}$
1	1	1	0	D ₇	$\overline{D_7}$

Table: 5.1: Truth table of 8:1 MUX

IMPLEMENTATION OF 8:1 MULTIPLEXER (74151)

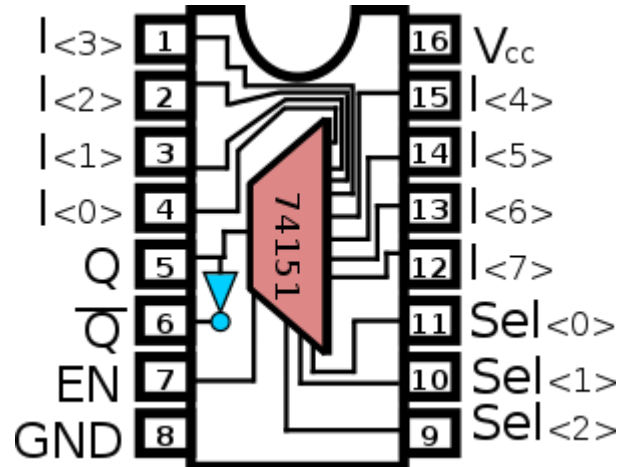


Figure 5.6

Observations for IC 74153

Symbol	Parameter	Min	Nom	Max	Units
V _{CC}	Supply Voltage	4.75	5	5.25	V
V _{IH}	HIGH Level Input Voltage	2			V
V _{IL}	LOW Level Input Voltage			0.8	V
I _{OH}	HIGH Level Output Current			-0.4	mA
I _{OL}	LOW Level Output Current			8	mA
T _A	Free Air Operating Temperature	0		70	°C

Procedure:

1. Test the digital ICs with the help of IC tester.
2. Make the connections as per the circuit diagram.
3. Connect the output pin on LED through Resistor.
4. Switch on V_{CC} and apply various combinations of input according to truth table.
5. Observe the logical output and verify with the truth tables.

Output: Hence the 4:1 mux is designed with basic gate as given in the figure and verified with the truth table. 8:1 mux are designed with 4:1 mux as given in circuit.

Result: 4:1 multiplexer and 8:1 multiplexer can be realized using basic logic gates, they can also be designed by using IC 74153 (multiplexer) and IC 74151 (multiplexer).

Discussion:

- Q1. What is the function of the enable input in a Multiplexer?
- Q2. What is the difference between multiplexer and decoder?
- Q3. List out the applications of multiplexer?
- Q4. Realize the following Boolean expression using 4:1 MUXs only

$$Z = \overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}BC\overline{D} + A\overline{B}C\overline{D} + A\overline{B}C\overline{D} + \overline{A}\overline{B}C\overline{D} + ABCD$$

Experiment No: 6

Aim: Design and realize a combinational circuit that will accept the 2421 BCD code and drive a TIL-312 to 7-Segment Display

Apparatus / Component Required: IC tester, Digital trainer kit, Digital ICs (7447), register (200 Ω / 330 Ω), seven segment display

Theory:

The Binary Coded Decimal (BCD) to 7-Segment Display Decoder. 7-segment **LED** (Light Emitting Diode) or **LCD** (Liquid Crystal) displays, provide a very convenient way of displaying information or digital data in the form of numbers, letters or even alpha-numerical characters and they consist of 7 individual LED's (the segments), within from 0 to 9 and A to F respectively, on the display the correct combination of LED segments need to be illuminated and **BCD to 7-segment Display Decoders** such as the 74LS47 do just that. A standard 7-segment LED display generally has 8 input connections, one for each LED segment and one that acts as a common terminal or connection for all the internal segments.

There are two important types of 7-segment LED digital display.

- The Common Cathode Display (CCD) - In the common cathode display, all the cathode connections of the LED's are joined together to logic "0" and the individual segments are illuminated by application of a "HIGH", logic "1" signal to the individual Anode terminals.
- The Common Anode Display (CAD) - In the common anode display, all the anode connections of the LED's are joined together to logic "1" and the individual segments are illuminated by connecting the individual Cathode terminals to a "LOW", logic "0" signal.

7-Segment Display Format

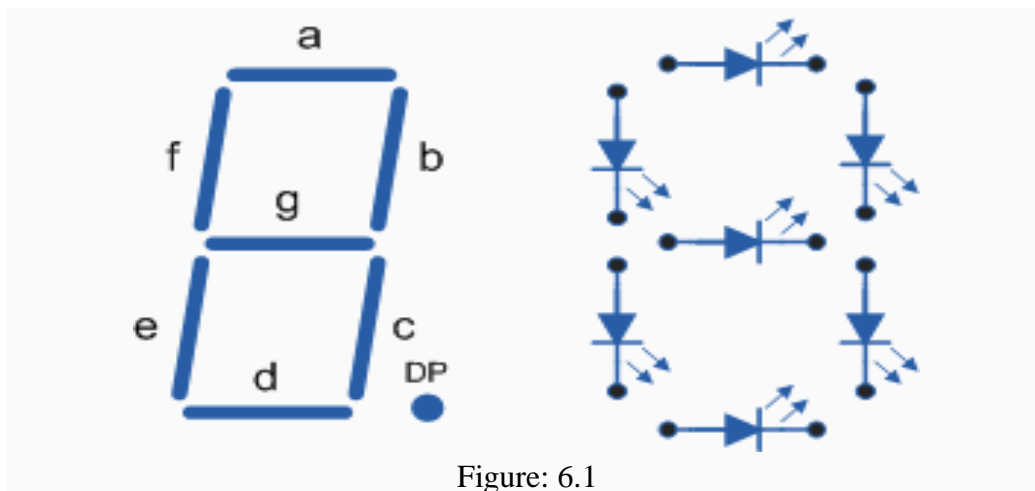


Figure: 6.1

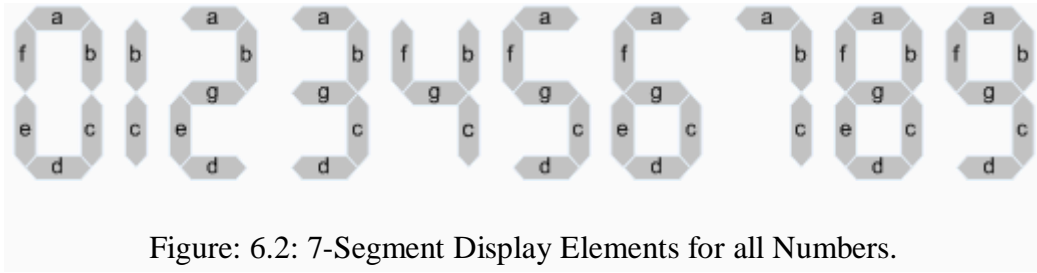


Figure: 6.2: 7-Segment Display Elements for all Numbers.

It can be seen that to display any single digit number from 0 to 9 or letter from A to F, we would need 7 separate segment connections plus one additional connection for the LED's "common" connection. Also as the segments are basically a standard light emitting diode, the driving circuit would need to produce up to 20mA of current to illuminate each individual segment and to display the number 8, all 7 segments would need to be lit resulting a total current of nearly 140mA, (8 x 20mA). Obviously, the use of so many connections and power consumption is impractical for some electronic or microprocessor based circuits and so in order to reduce the number of signal lines required to drive just one single display, display decoders such as the BCD to 7-Segment Display Decoder and Driver IC's are used instead.

Combinational circuit for converting 2421 BCD code to seven segment number

Both the 2421 code and BCD code are 4-bit codes and represent the decimal equivalents 0 to 9. To design the converter circuit for the above, first the truth table is prepared with the input variables W, X, Y, and Z of 2421 code, and the output variables A, B, C, and D. Karnaugh maps to obtain the simplified expressions of the output functions are shown in Figure. Unused combinations are considered as don't-care condition.

Table 6.1: Truth Table for BCD to Seven Segment

Decimal Digit	Input lines				Output lines							Display pattern
	A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	0	1	1	1	1	1	1	0	
1	0	0	0	1	0	1	1	0	0	0	0	
2	0	0	1	0	1	1	0	1	1	0	1	
3	0	0	1	1	1	1	1	1	0	0	1	
4	0	1	0	0	0	1	1	0	0	1	1	
5	0	1	0	1	1	0	1	1	0	1	1	
6	0	1	1	0	1	0	1	1	1	1	1	
7	0	1	1	1	1	1	1	0	0	0	0	
8	1	0	0	0	1	1	1	1	1	1	1	
9	1	0	0	1	1	1	1	1	0	1	1	

Decoder

The output expressions are:

$$a = F1(A, B, C, D) = \sum m(0, 2, 3, 5, 7, 8, 9)$$

$$b = F2(A, B, C, D) = \sum m(0, 1, 2, 3, 4, 7, 8, 9)$$

$$c = F3(A, B, C, D) = \sum m(0, 1, 3, 4, 5, 6, 7, 8, 9)$$

$$d = F4(A, B, C, D) = \sum m(0, 2, 3, 5, 6, 8)$$

$$e = F5(A, B, C, D) = \sum m(0, 2, 6, 8)$$

$$f = F6(A, B, C, D) = \sum m(0, 4, 5, 6, 8, 9)$$

$$g = F7(A, B, C, D) = \sum m(2, 3, 4, 5, 6, 8, 9)$$

The K-Map of the above expressions are:

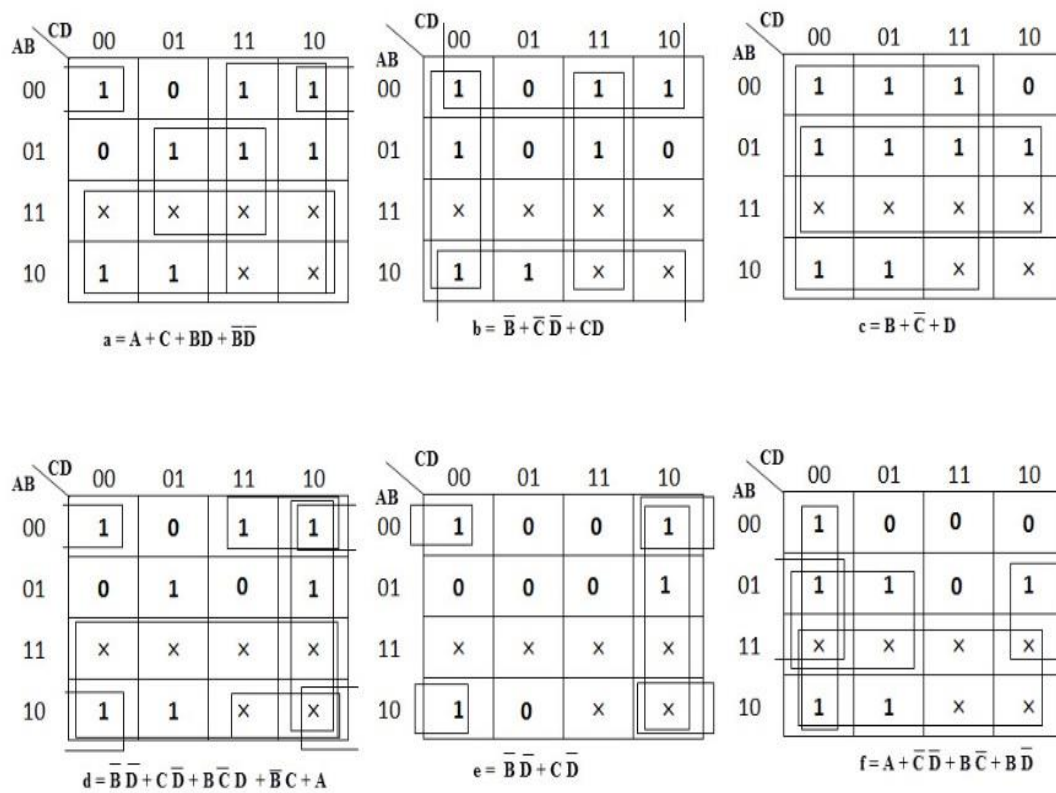


Figure: 6.3

AB \ CD	00	01	11	10
00	0	0	1	1
01	1	1	0	1
11	x	x	x	x
10	1	1	x	x

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

Figure: 6.4

From the above simplification, we get the output values as

$$a = A + C + BD + \bar{B}\bar{D}$$

$$b = \bar{B} + \bar{C}\bar{D} + CD$$

$$c = B + \bar{C} + D$$

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

$$e = \bar{B}\bar{D} + C\bar{D}$$

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

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

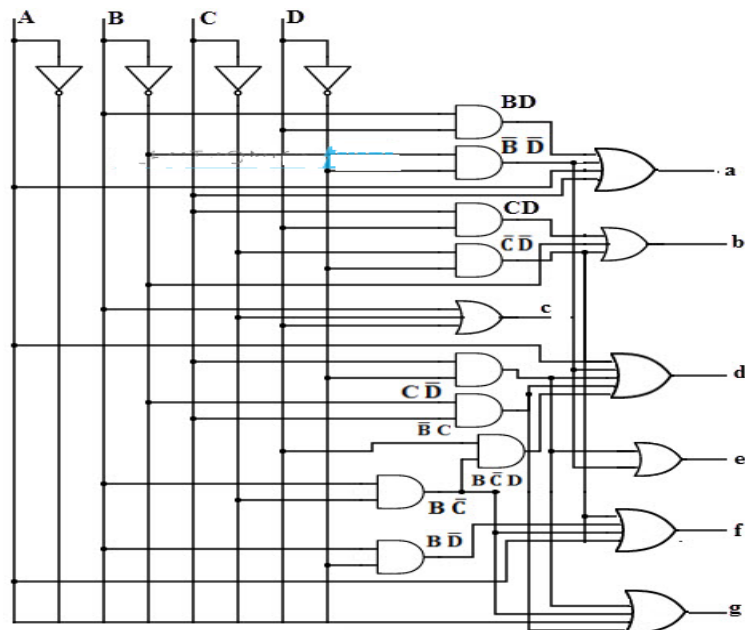


Figure 6.5: Logic diagram of BCD to Seven segment decoder

Procedure:

1. Test the digital ICs with the help of IC tester.
2. Make the connections as per the circuit diagram on the digital trainer kit.
3. Connect the output pins to the seven segment pins.
4. Switch on V_{CC} and apply various combinations of input according to the truth table.
5. Observe the logical output and verify with the truth tables.

BCD to 7-Segment Display Decoders using TTL 74LS47

A binary coded decimal (BCD) to 7-segment display decoder such as the TTL 74LS47 or 74LS48, have 4 BCD inputs and 7 output lines, one for each LED segment. This allows a smaller 4-bit binary number (half a byte) to be used to display all the denary numbers from 0 to 9 and by adding two displays together, a full range of numbers from 00 to 99 can be displayed with just a single byte of 8 data bits.

PIN diagram for 7447 IC:

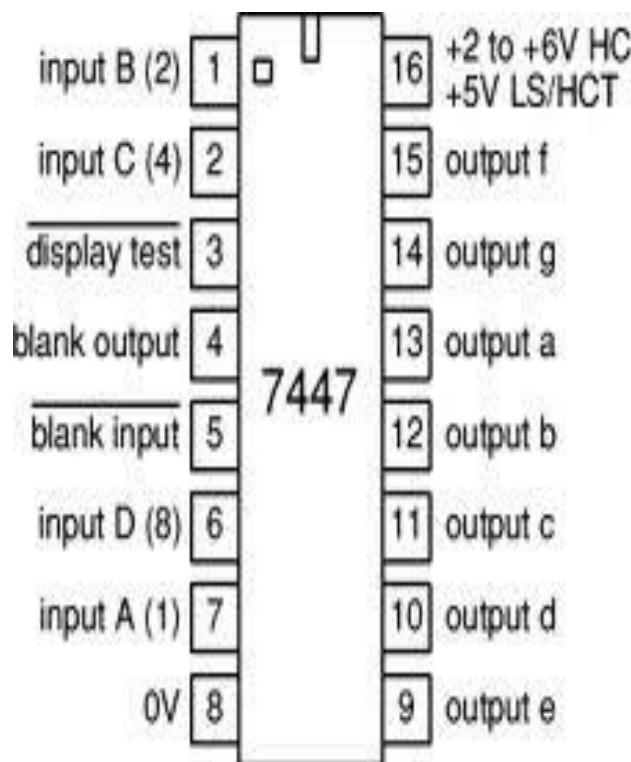


Figure 6.6: PIN diagram for 7447 IC

The use of **packed** BCD allows two BCD digits to be stored within a single byte (8-bits) of data, allowing a single data byte to hold a BCD number in the range of 00 to 99.

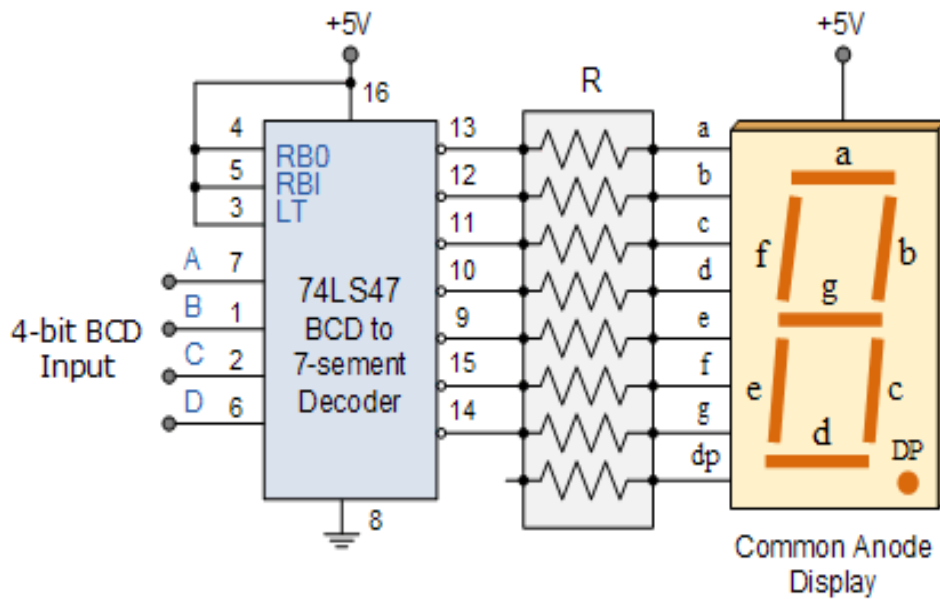


Figure 6.7: 4 BCD to 7-Segment Display Decoders using TTL 74LS47

Observations: For DM7447A:

Symbol	Parameter	Min	Typical	Max	Unit
V _{CC}	Supply Voltage	4.75	5	5.25	V
V _{IH}	High Level Input Voltage	2	-	-	V
V _{IL}	Low Level Input Voltage	-	-	0.8	V
V _{OH}	High level output voltage(a to g)	-	-	30	
I _{OH}	High Level Output Current(BI/RBO)	-	-	-0.2	μA
I _{OL}	Low Level Output Current(a to g)	-	-	40	mA

Output: Hence the BCD to seven segment display is designed using basic gates and with a BCD to seven segment display IC as given in the figure and verified with the truth table.

Result: It is possible to display any single digit number on a 7-segment display by sending a high digital signal to the specific segments that make up the number. It is possible to display the decimal value of a binary number on a 7-segment display using a BCD decoder. However, this method will allow displaying only digits from 0 to 9. In the case of the decoder circuit, any binary number between 1010 through 1111 (A to F) is an invalid input and would provide distorted shapes on the LCD display. The usage of a 7-segment display paired with a BCD decoder is opening the door for an application using digital computation requiring a human-readable. That application can be for instance: “a clock, a timer, a calculator, counter...” Thus we had concluded that 7 segment display can be implemented using IC,s and we had seen different numbers 0-9.

Discussion:

- Q1. What is code conversion?
- Q2. Can a decoder function as a Demultiplexer?
- Q3. Give the applications of seven segment display?
- Q4. What are happened if the input given in between A to F.

Experiment No: 7

Aim: Using basic logic gates realize the R-S, J-K and D flip flops with and without clock pulse and verify truth table.

Apparatus / Component Required: IC tester, Digital trainer kit, connecting wires, Digital ICs (7400, 7402, 7404, 7408, 7432)

Theory:

Sequential Logic circuits have some form of inherent "Memory" built in to them as they are able to take into account their previous input state as well as those actually present, a sort of "before" and "after" is involved with sequential circuits.

In other words, the output state of a sequential logic circuit is a function of the following three states, the "present input", the "past input" and/or the "past output". Sequential Logic circuits remember these conditions and stay fixed in their current state until the next clock signal changes one of the states, giving sequential logic circuits "Memory".

Sequential logic circuits are generally termed as two state or Bistable devices which can have their output or outputs set in one of two basic states, a logic level "1" or a logic level "0" and will remain "latched" (hence the name latch) indefinitely in this current state or condition until some other input trigger pulse or signal is applied which will cause the bistable to change its state once again.

Sequential Logic Representation

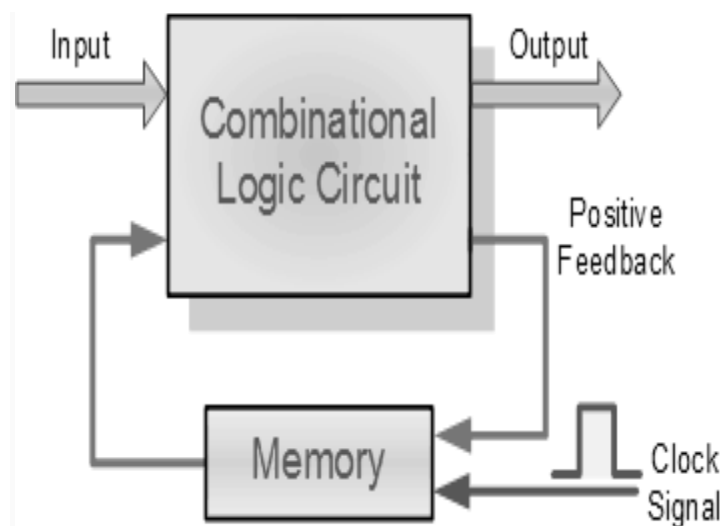


Figure 7.1: Block diagram of Sequential circuit

The word "Sequential" means that things happen in a "sequence", one after another and in Sequential Logic circuits, the actual clock signal determines when things will happen next.

Simple sequential logic circuits can be constructed from standard Bistable circuits such as Flip-flops, Latches and Counters and which themselves can be made by simply connecting together universal NAND Gates and/or NOR Gates in a particular combinational way to produce the required sequential circuit.

FLIP-FLOP:-

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

S-R Flip Flop

The SR flip-flop can be considered as a 1-bit memory, since it stores the input pulse even after it has passed. Flip-flops (or bi-stables) of different types can be made from logic gates and, as with other combinations of logic gates, the NAND and NOR gates are the most versatile, the NAND being most widely used. This is because, as well as being universal, i.e. it can be made to mimic any of the other standard logic functions, it is also cheaper to construct. The SET-RESET flip flop is designed with the help of two NOR gates and also two NAND gates. These flip flops are also called S-R Latch.

- **S-R Flip Flop using NOR Gate(Without Clock)**

The design of such a flip flop includes two inputs, called the SET [S] and RESET [R]. There are also two outputs, Q and Q'. The diagram and truth table is shown below.

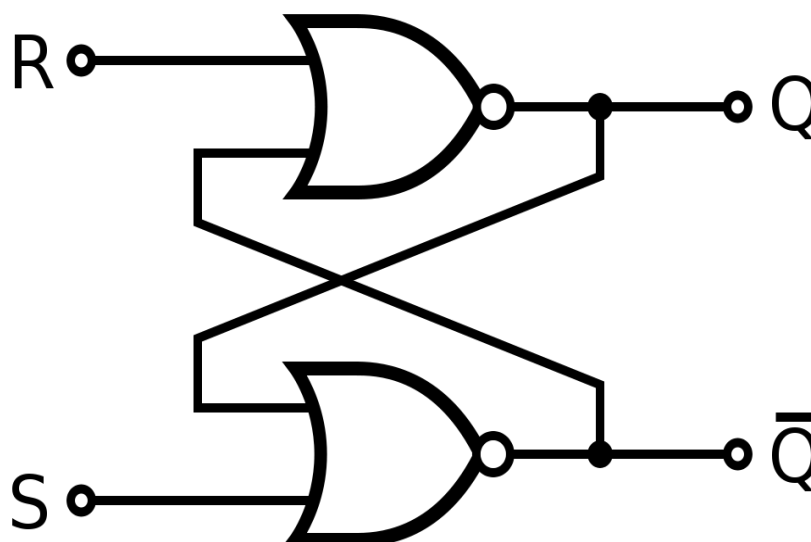


Figure 7.2: Logic Diagram of NOR based S-R Flip flop

Truth Table – 7.1: S-R Latch using NOR Gate

Inputs		Outputs		Action
S	R	Q_{n+1}	$\overline{Q_{n+1}}$	
0	0	Q_n	$\overline{Q_n}$	No change
0	1	0	1	Reset
1	0	1	0	Set
1	1	?	?	Forbidden

- **S-R Flip Flop using NAND Gate(Without Clock)**

The circuit of the S-R flip flop using NAND Gate and its truth table is shown below.

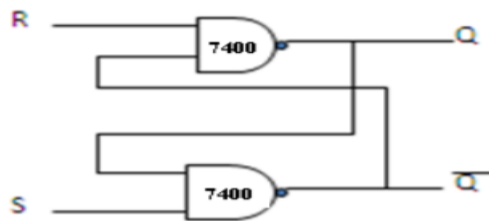


Figure 7.3: Logic Diagram of NAND based S-R Latch

Truth Table – 7.2: S-R Latch using NAND Gate

Inputs		Outputs		Action
S	R	Q_{n+1}	$\overline{Q_{n+1}}$	
0	0	?	?	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	Q_n	$\overline{Q_n}$	No change

- **Clocked S-R Flip Flop**

It is also called a Gated S-R flip flop.

The problems with S-R flip flops using NOR and NAND gate is the invalid state. This problem can be overcome by using a bistable SR flip-flop that can change outputs when certain invalid states are met, regardless of the condition of either the Set or the Reset inputs. For this, a clocked S-R flip flop is designed by adding two NAND gates to a basic NAND Gate flip flop. The circuit diagram and truth table is shown below.

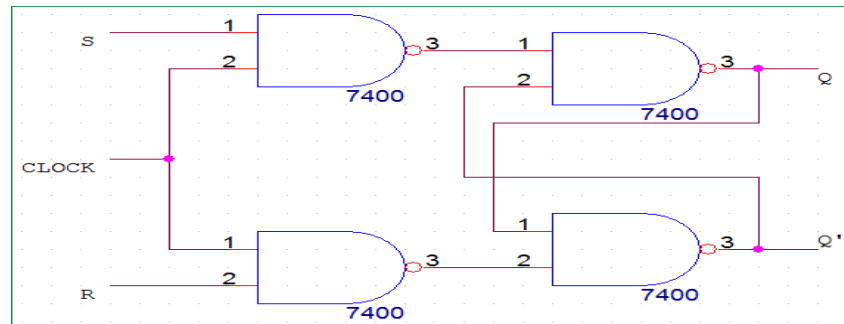


Figure 7.4: Logic Diagram of NAND based Clocked S-R Flip flop

Truth Table – 7.3 Clocked Based S-R Flip Flop

Clock	Inputs		Outputs		Action
	CLK	S	R	Q_{n+1}	
0	X	X	Q_n	$\overline{Q_n}$	No change
1	0	0	Q_n	$\overline{Q_n}$	No change
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	?	?	Forbidden

D Flip Flop

D flip flop is actually a slight modification of the above explained clocked SR flip-flop. From the figure you can see that the D input is connected to the S input and the complement of the D input is connected to the R input.

- **D Flip Flop without clock**

The circuit diagram of D latch with NAND gates:

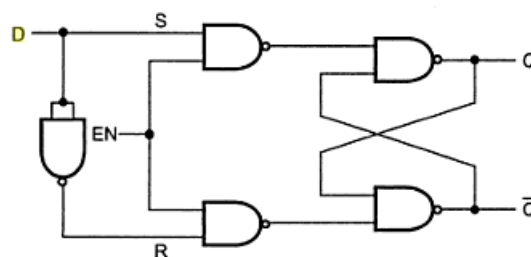


Figure 7.5: Logic Diagram of NAND Based D Latch

Truth Table – 7.4: D Latch

Enable	Input	Output
En	D	Q_{n+1}
0	X	Q _n
1	0	0
1	1	1

- Clocked D Flip Flop**

The D flip-flop is the modification of the SR flip flop which is shown in the figure 5. The i/p D goes directly into the input S and the complement of the input D goes to the input R. The D input is sampled during the existence of a clock pulse. If it is 1, then the flip flop is switched to the set state. If it is 0, then the flip-flop switches to the clear state.

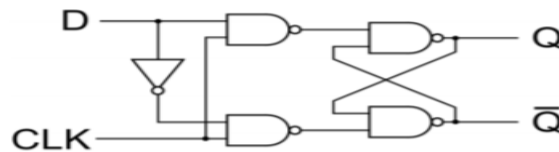


Figure 7.6: Logic Diagram of Clocked based D Flip Flop

Truth Table 7.5: D Flip Flop

Clock	Input	Output
CLK	D	Q_{n+1}
0	X	Q _n
1	0	0
1	1	1

- IC 7474 (D FF)**

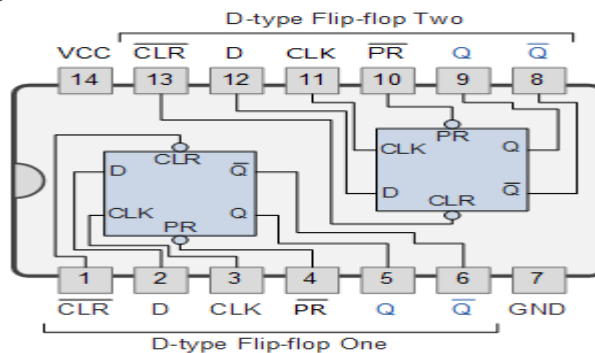


Figure 7.7

J-K Flip Flop

JK flip-flop is basically an SR flip flop with feedback which enables only one of its two input terminals, either SET or RESET to be active at any one time thereby eliminating the invalid condition seen previously in the SR flip flop circuit. The only difference is that the intermediate state is more refined and precise than that of an S-R flip flop.

- **JK Flip Flop without clock**

JK latch is similar to RS latch. This latch consists of 2 inputs J and K as shown in the below figure 6. The ambiguous state has been eliminated here: when the inputs of J-K latch are high, then output toggles. The output feedback to inputs is the only difference, which is not there in the RS latch.

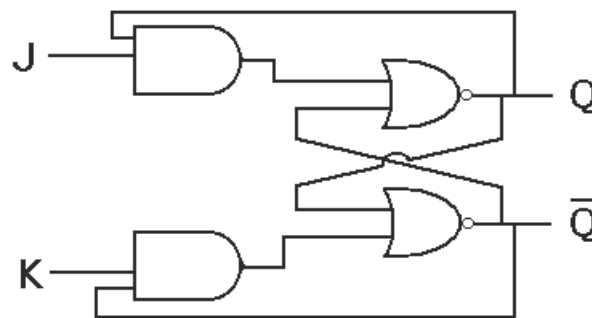


Figure 7.8: Logic Diagram of J-K Latch

Truth Table 7.6: J-K Latch

Inputs		Outputs		Action
J	K	Q_{n+1}	$\overline{Q_{n+1}}$	
0	0	Q_n	$\overline{Q_n}$	No change
0	1	0	1	Reset
1	0	1	0	Set
1	1	$\overline{Q_n}$	Q_n	Toggle

When both the inputs J and K have a HIGH state, the flip-flop switches to the complement state. So, for a value of $Q = 1$, it switches to $Q=0$ and for a value of $Q = 0$, it switches to $Q=1$.

- **JK Flip Flop with clock**

A JK flip-flop is a refinement of the SR flip-flop in that the indeterminate state of the SR type is defined in the JK type. Inputs J and K behave like inputs S and R to set and clear the flipflop (note that in a JK flip-flop, the letter J is for set and the letter K is for clear).

When logic 1 inputs are applied to both J and K simultaneously, the flip-flop switches to its complement state, ie., if $Q=1$, it switches to $Q=0$ and vice versa

The circuit diagram and truth-table of a J-K flip flop is shown below.

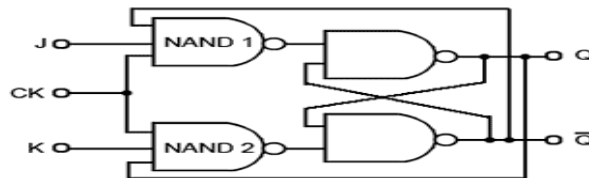


Figure 7.9: Logic Diagram of J-K Clocked Flip Flop

Truth Table 7.6: J-K Flip Flop

Clock	Inputs		Outputs		Action
	J	K	Q_{n+1}	$\overline{Q_{n+1}}$	
0	X	X	Q_n	$\overline{Q_n}$	No change
1	0	0	Q_n	$\overline{Q_n}$	No change
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	$\overline{Q_n}$	Q_n	Toggle

The output may be repeated in transitions once they have been complimented for $J=K=1$ because of the feedback connection in the JK flip-flop. This can be avoided by setting a time duration lesser than the propagation delay through the flip-flop. The restriction on the pulse width can be eliminated with a master-slave or edge-triggered construction.

- IC 7476 (JK FF)

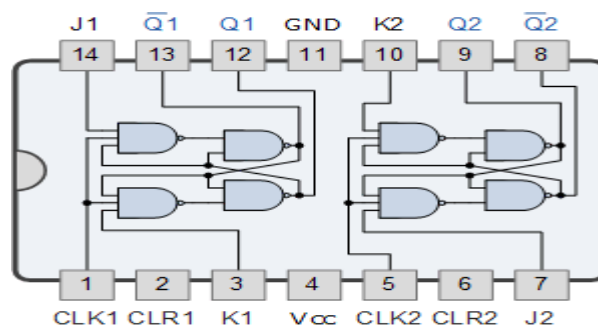


Figure 7.10: Pin diagram of IC 7476

Procedure:

1. Test the digital ICs with the help of IC tester.
2. Make the connections as per the circuit diagram.
3. Connect the output pin of LED on Digital Trainer Kit.
4. Switch on V_{CC} and apply various combinations of input according to the truth table.
5. Observe the logical output and verify with the truth tables.

Observtions:

Symbol	Parameter	Min	Typical	Max	Unit
V_{CC}	Supply Voltage	4.75	5	5.25	V
V_{IH}	High Level Input Voltage	2	-	-	V
V_{IL}	Low Level Input Voltage	-	-	0.8	V
I_{OH}	High Level Output Current	-	-	-0.4	mA
I_{OL}	Low Level Output Current	-	-	16	mA

Output: Hence the S-R, J-K, D flip flops with and without clocks are designed with basic gates IC and verified with the truth table.

Result:

- The function of the S – R flip-flop is to store a bit value of either 0 or 1 for later use based on the S and R input values. When S is high, the flip-flop stores a logic value of 1 and stores a logic 0 when input R is high.
- The D – Flip-flop operates by propagating the logic level placed on the input D to the Output Q on the high edge of the clock. When the clock level is low, the state of the flip-flop latches and output does not change
- The J-K flip-flop sequential operation is the same as the S-R flip-flop with set and reset inputs. But it has no forbidden or invalid input states of the S-R Latch, when both inputs, S and R, are both equal to logic 1.

Discussion:

- Q1. What is sequential circuit?
- Q2. What is Synchronous sequential circuit?
- Q3. What is an excitation table?
- Q4. What do you mean by triggering of flip-flop?
- Q5. What is called latch?
- Q6. What advantage does a J-K Flip-flop have over an S-R?
- Q7. What is meant by Race around condition?

Experiment No: 8

Aim: Construct a divide by 2, 4 & 8 asynchronous counter. Construct a 4-bit binary counter and ring counter for a particular output pattern using D flip flop.

Apparatus / Component Required: IC tester, Digital trainer kit, Digital IC 7474

Theory: 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.

Asynchronous or ripple counters are arranged in such a way that the output of one flip flop changes the state of the next. In a long chain of ripple counter stages, the last flip flop changes its state considerably later than the first FF due to propagation delays in each stage. If the output lines are connected to logic which can respond to these in-between states, glitches can occur that are so fast the error may be difficult to track down.

The 2-bit ripple counter circuit has four different states, each one corresponding to a count value. Similarly, a counter with n flip-flops can have 2 to the power n states. The number of states in a counter is known as its mod (modulo) number. Thus a 2-bit counter is a mod-4 counter.

A mod- n counter may also describe as a divide-by- n counter. This is because the most significant flip-flop (the furthest flip-flop from the original clock pulse) produces one pulse for every n pulses at the clock input of the least significant flip-flop (the one triggers by the clock pulse). Thus, the above counter is an example of a divide-by-4 counter.

Divide by Two Counter

The edge-triggered D-type ip-ops which we introduced in the previous Section are quite useful and versatile building blocks of sequential logic. A simple application is the divide-by-2 counter shown in Fig. 1, along with the corresponding timing diagram.

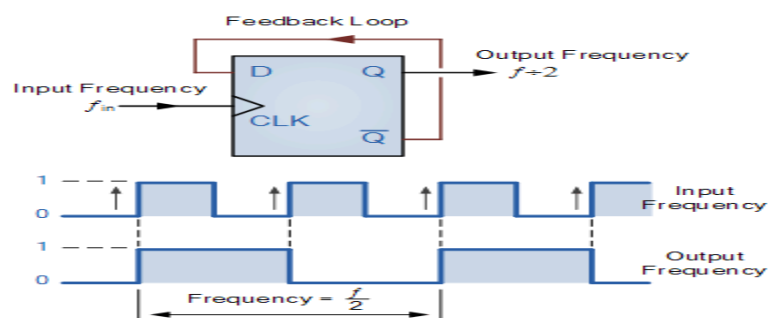


Figure 8.1: Block diagram & Timing wave-forms of divide by 2 counter

Divide by four Counter

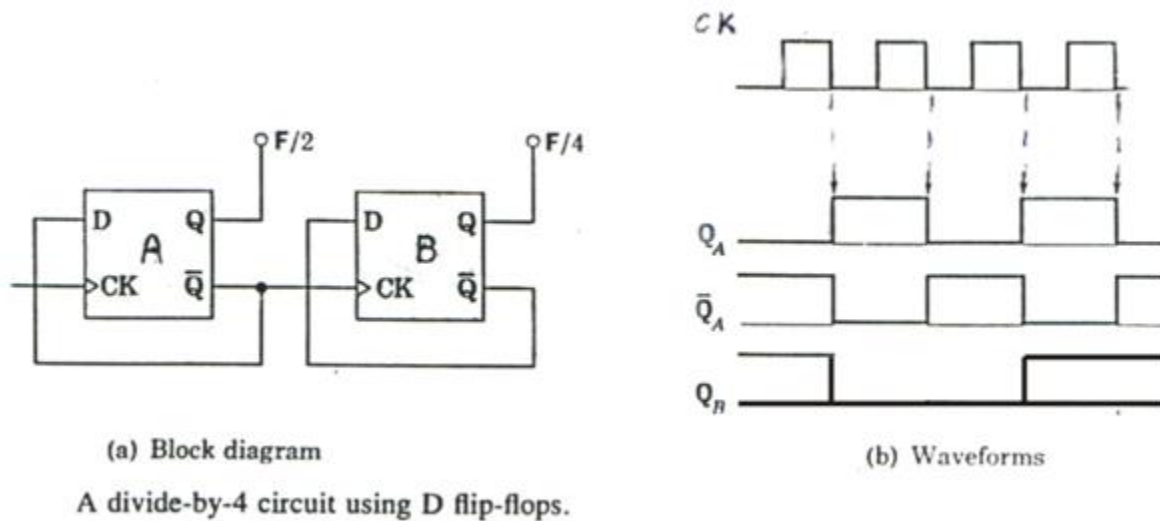


Figure 8.2: Block diagram & Timing wave-forms of divide by 4 counter

Since the Q output of the first flip flop changes every time the clock input goes low, it effectively divides the input frequency by 2. The other output of the flip flop becomes the clock input to the next stage, so the output of the second flip flop divides the original input frequency by 4. Hence the term "divide-by-4" or "modulo-4".

Divide by Eight Counter

We can chain as many ripple counters together as we like. A three bit ripple counter will count $2^3=8$ numbers, and an n-bit ripple counter will count 2^n numbers.

The problem with ripple counters is that each new stage put on the counter adds a delay. This propagation delay is seen when we look at a less idealized timing diagram:

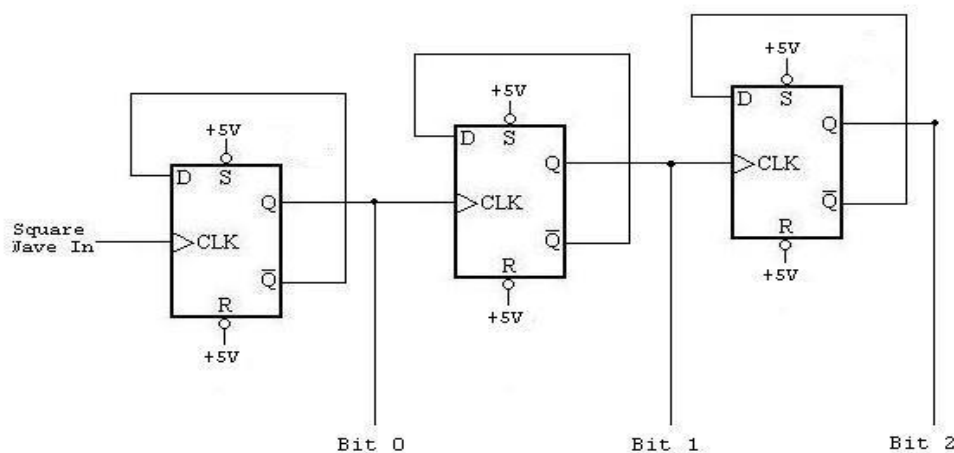


Figure 8.3: Block diagram of divide by 8 counter

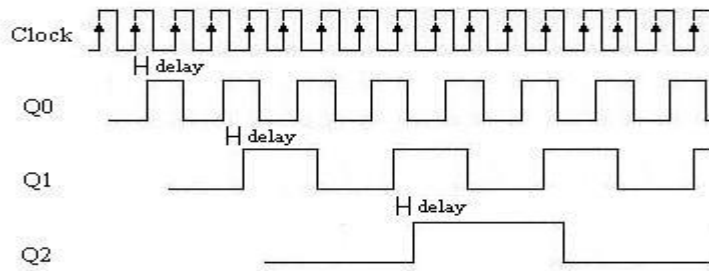


Figure 8.4: Timing wave-forms of divide by 8 counter

Binary 4 bit UP-COUNTER

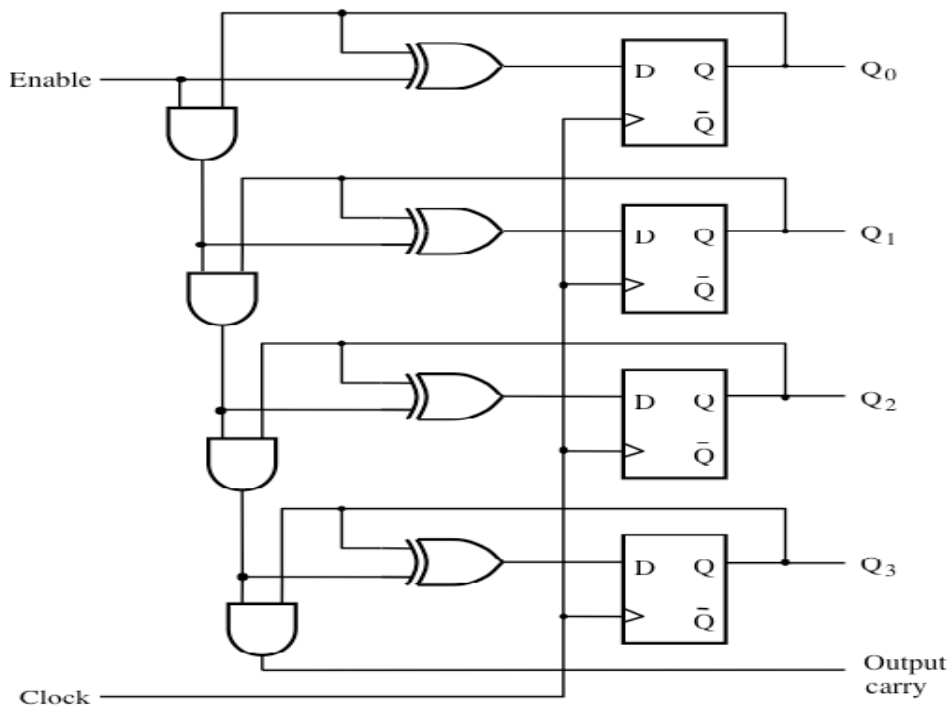


Figure 8.5: Block diagram of 4-bit binary-up counter

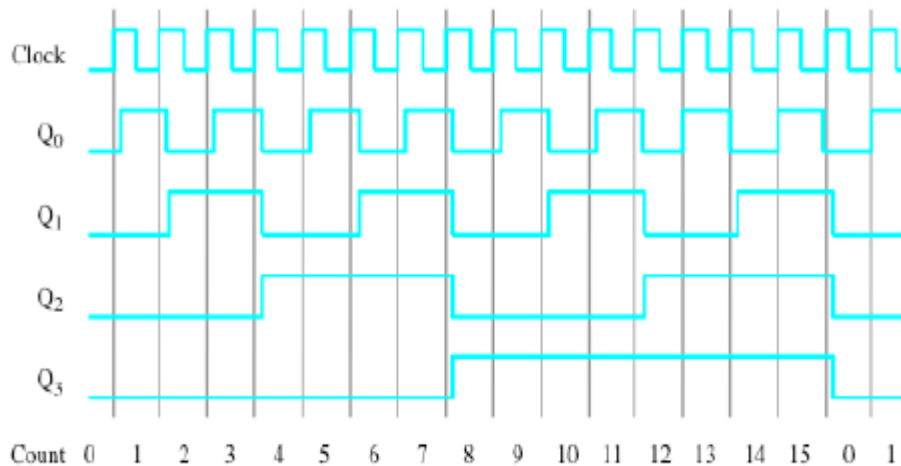


Figure 8.5: Timing waveform of 4-bit binary-up counter

Ring Counter

A ring counter is a Shift Register (a cascade connection of flip-flops) with the output of the last flip flop connected to the input of the first. It is initialized such that only one of the flip flop output is 1 while the remainder is 0. The 1 bit is circulated so the state repeats every n clock cycles if n flip-flops are used. The "MOD" or "MODULUS" of a counter is the number of unique states. The MOD of the n flip flop ring counter is n .

The following is a 4-bit ring counter constructed from D flip-flops. The output of each stage is shifted into the next stage on the positive edge of a clock pulse. If the CLEAR signal is high, all the flip-flops except the first one FF0 are reset to 0. FF0 is preset to 1 instead.

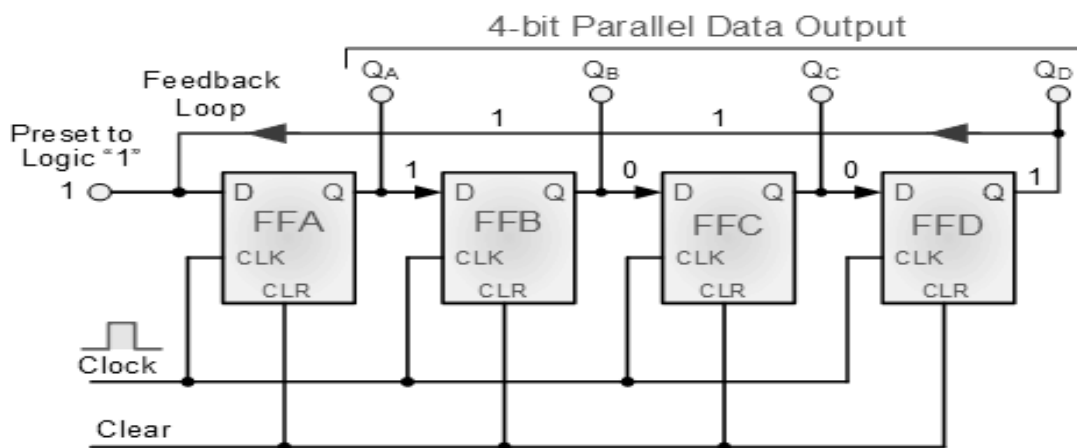


Figure 8.6: Block diagram of ring counter

Clock Pulse	Q3	Q2	Q1	Q0
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	1	0	0	0

CLEAR	FF0	FF1	FF2	FF3
	1	0	0	0

Figure 8.7: Timing states of ring counter

Since the count sequence has 4 distinct states, the counter can be considered as a mod-4 counter. Only 4 of the maximum 16 states are used, making ring counters very inefficient in terms of state usage. But the major advantage of a ring counter over a binary counter is that it is self-decoding. No extra decoding circuit is needed to determine what state the counter is in.

IC used for realizing above counters:

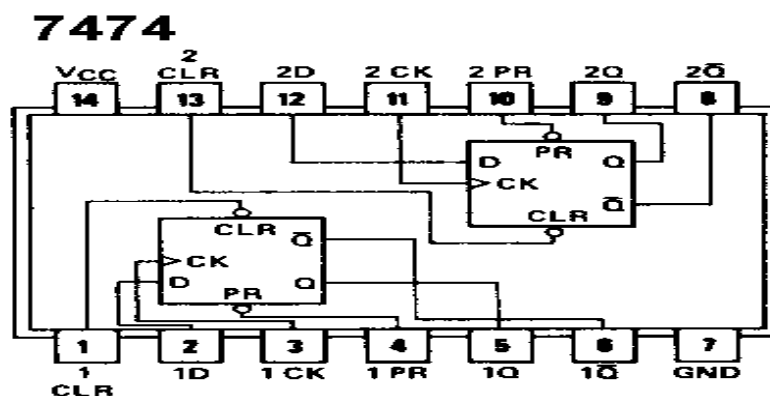


Figure 8.8: Pin-diagram of IC 7474

Procedure:

1. Make the connections as per the circuit diagram.
2. Connect the output pin on LED through Resistor.
3. Switch on V_{cc} and apply various combinations of input according to truth table.
4. Observe the logical output and verify with the truth tables.

Output: counters are designed with the help of trainer kit and verified with the truth table.

Result: Divide by 2, 4 & 8 asynchronous counter, 4-bit binary counter and ring counter for a particular output pattern using D flip flop are designed and their truth table are verified.

When the value of output voltage is greater than 2.4V then it will be represented by logic 1 and if output voltage is less than 0.4 volt it will be represented by logic 0.

Discussion:

- Q1. What is the difference between synchronous and asynchronous counter?
- Q2. What is up and down counter?
- Q3. How many flip flops are required for designing the decade counter?
- Q4. How the synchronous counters eliminate the delay problems encountered in asynchronous counters.
- Q5. Design Mod 5 counter in synchronous mode.

Experiment No: 9

Aim: Design and construct unidirectional shift register and verify the function.

Apparatus / Component Required: IC tester, Digital trainer kit, Digital ICs 7474

Theory:

A flip-flop stores 1-bit of digital information. It is also referred to as 1-bit register. An array of flip-flops is required to store the no. of bits. This is called register. The data can be entered into or retrieved from the register. A register is capable of shifting its binary information in one or both directions is known as shift register. The logical configuration of shift register consists of a D-Flip flop cascaded with output of one flip flop connected to input of next flip flop. All flip flops receive common clock pulses which causes the shift in the output of the flip flop. The simplest possible shift register is one that uses only flip-flops, as shown in figure. The Q output of a given flip-flop is connected to the D input of the flip-flop at its right. Each clock pulse shifts the contents of the register one bit position to the right. The serial input determines what goes into the leftmost flip-flop during the shift. The serial output is taken from the output of the rightmost flip-flop prior to the application of a pulse. Although this register shifts its contents to the right, if we turn the page upside down; we find that the register shifts its contents to the left. Thus, a unidirectional shift register can function either as a shift-right or as shift-left register.

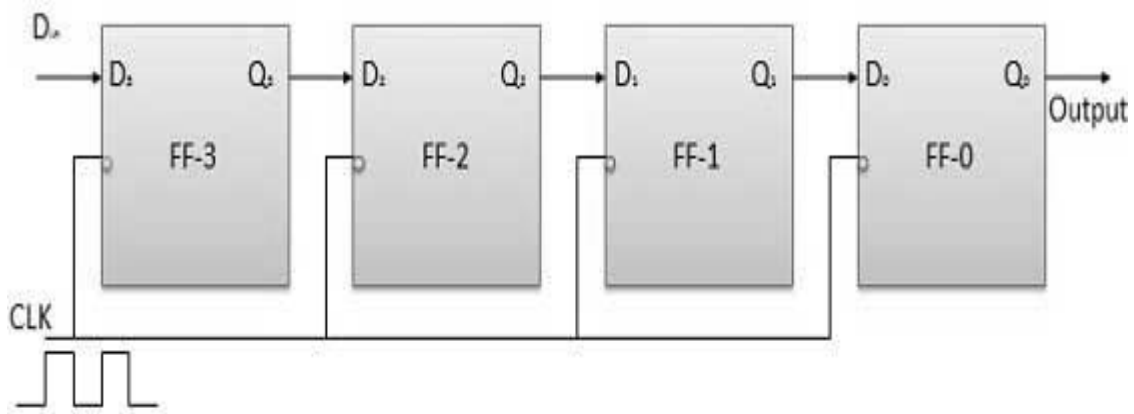


Figure 9.1: Block diagram of unidirectional shift register

The register in figure shifts its contents with every clock pulse during the positive edge of the pulse transition. If we want to control the shift so that it occurs only with certain pulses but not with others, we must control CLK input of the register.

Shift registers can be used for converting serial data to parallel, and vice versa. If we have access to all the flip-flop outputs of a shift register, then information entered serially by

shifting can be taken out in parallel from the outputs of the flip-flops. If a parallel-load capability is added to a shift register, then data entered in parallel can be taken out in serial fashion by shifting the data stored in the register.

There are five basic types of shift registers:

- 1) Serial in serial out (SISO)
- 2) Serial in parallel out (SIPO)
- 3) Parallel in serial out (PISO)
- 4) Parallel in parallel out (PIPO)
- 5) Bidirectional shift registers

Unidirectional shift register - Serial in serial out (SISO)

A Serial-in Serial-out shift register can be implemented using D-type flip-flops joined together, the output of one flip-flop used as the input to the next flip-flop. The circuit for a 4-bit Serial-in Serial-out shift register is shown below.

The operation of the serial-in Serial-out shift register can be easily explained. Consider the circuit shown. On each clock edge (rising in this case) we can say the following about the outputs of each stage of the register (i.e. each D-type flip-flop in the register):

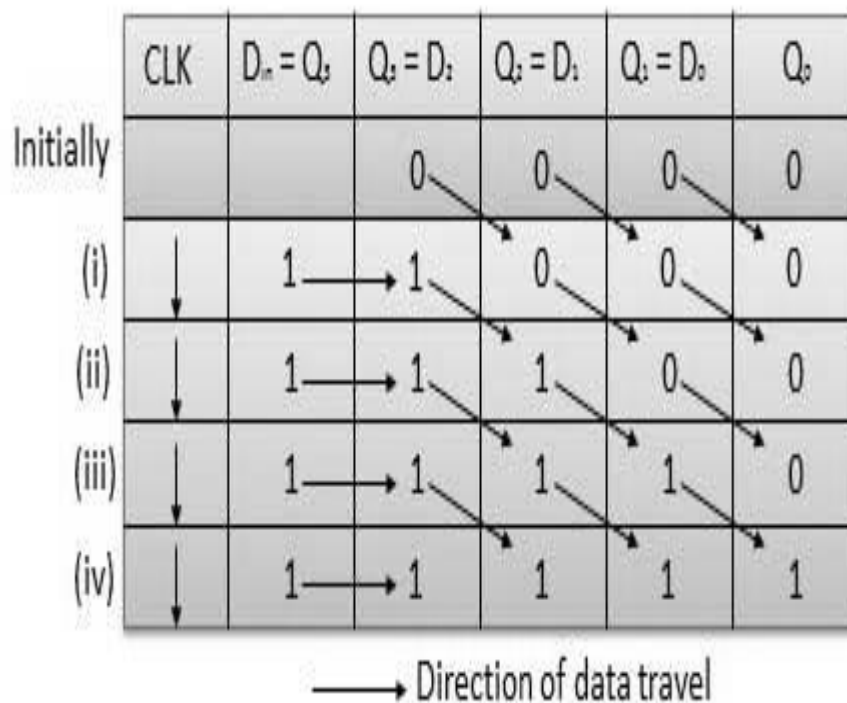


Figure 9.2: Showing shifting operation in SISO

Waveforms

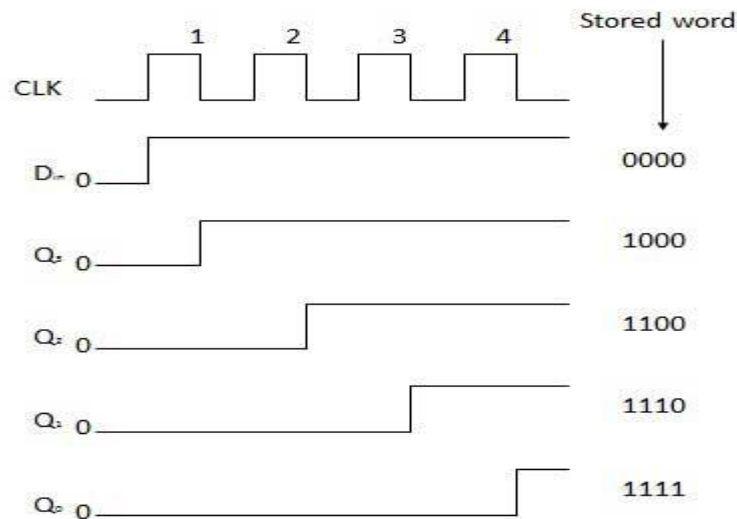


Figure 9.3: Timing Waveforms of SISO

Serial in parallel out (SIPO)

- 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.

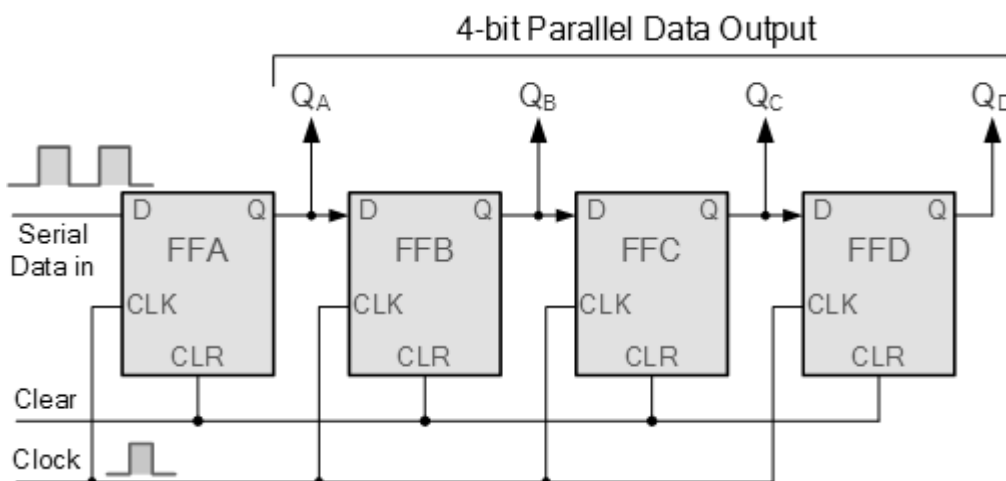


Figure 9.4: Block diagram of SIPO

Truth Table

Clock Pulse No	QA	QB	QC	QD
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	0	0	0	0

Parallel-in to Serial-out (PISO)

The Parallel-in to Serial-out shift register acts in the opposite way to the serial-in to parallel-out one above. The data is loaded into the register in a parallel format i.e. all the data bits enter their inputs simultaneously, to the parallel input pins P_A to P_D of the register. The data is then read out sequentially in the normal shift-right mode from the register at Q representing the data present at P_A to P_D . This data is outputted one bit at a time on each clock cycle in a serial format. It is important to note that with this system a clock pulse is not required to parallel load the register as it is already present, but four clock pulses are required to unload the data.

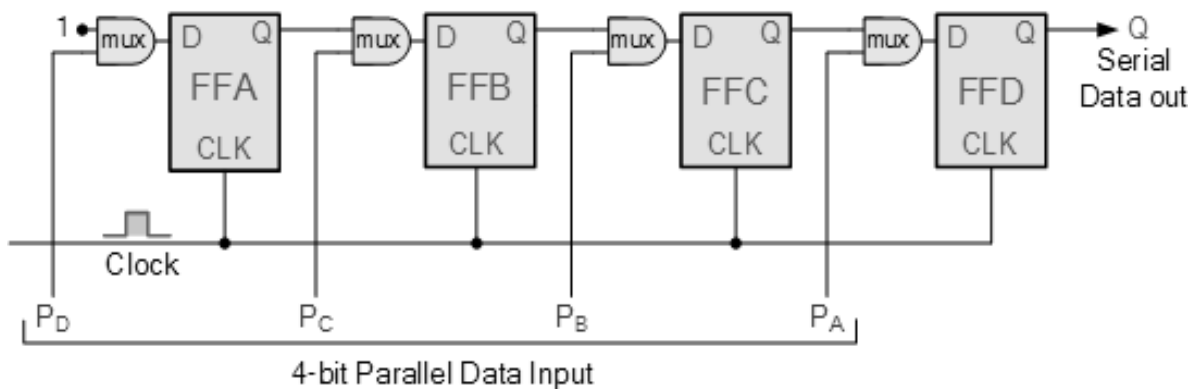


Figure 9.5: Block diagram of unidirectional shift register

Truth Table

CLK	Q3	Q2	Q1	Q0	O/P
0	1	0	0	1	1
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	1

As this type of shift register converts parallel data, such as an 8-bit data word into serial format, it can be used to multiplex many different input lines into a single serial DATA stream which can be sent directly to a computer or transmitted over a communications line.

Parallel-in to Parallel-out (PIPO)

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

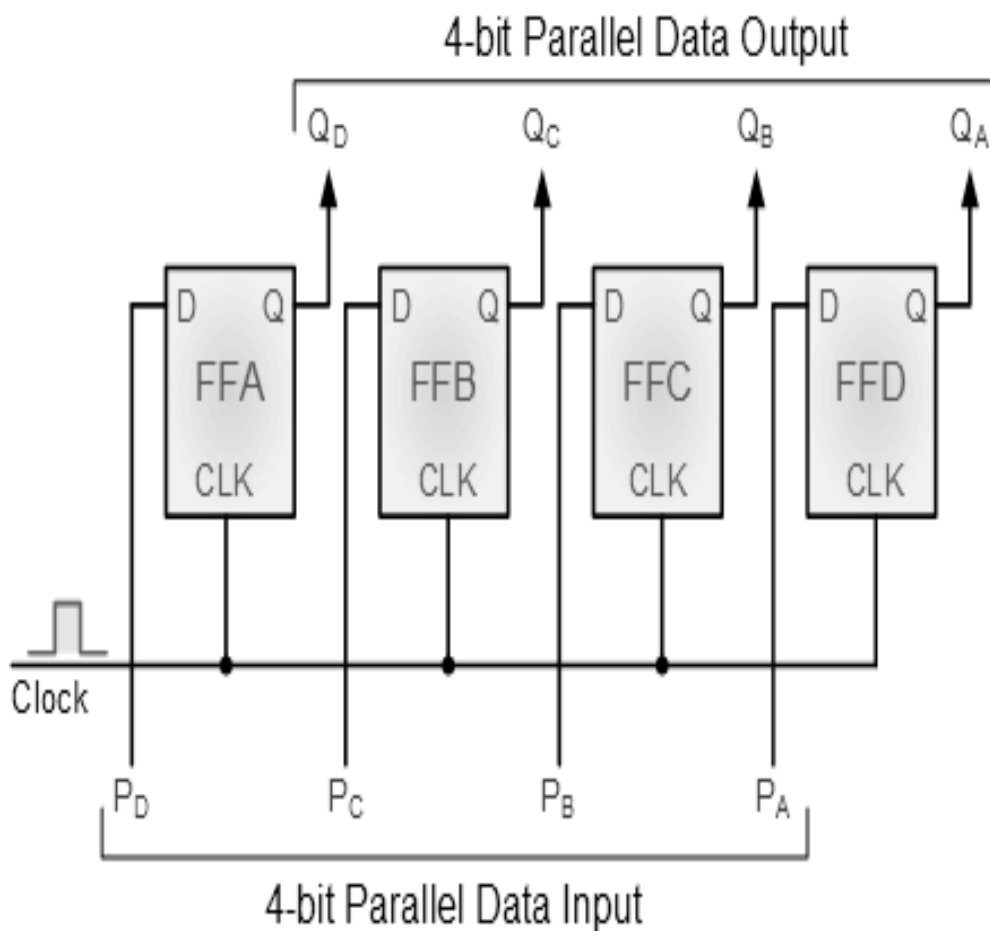


Figure 9.6: Block diagram of unidirectional shift register

TRUTH TABLE:

CLK	DATA INPUT				OUTPUT			
	DA	DB	DC	DD	QA	QB	QC	QD
1	1	0	0	1	1	0	0	1
2	1	0	1	0	1	0	1	0

Pin Diagram for IC 7495:

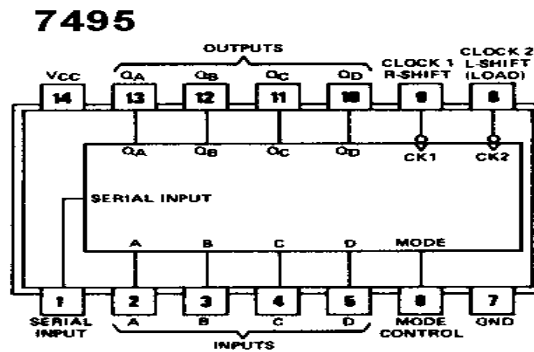
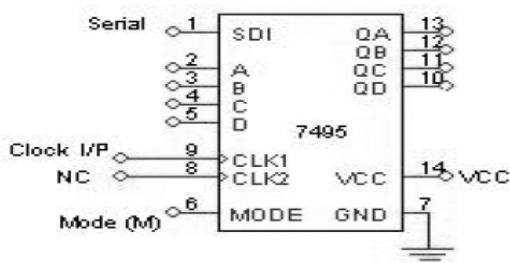


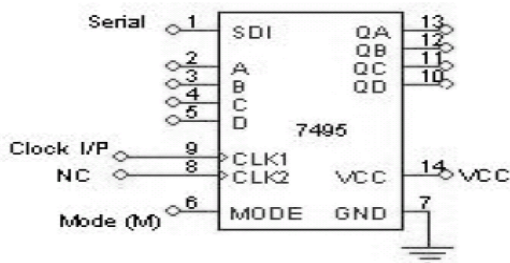
Figure 9.7: Pin-diagram of IC 7495

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

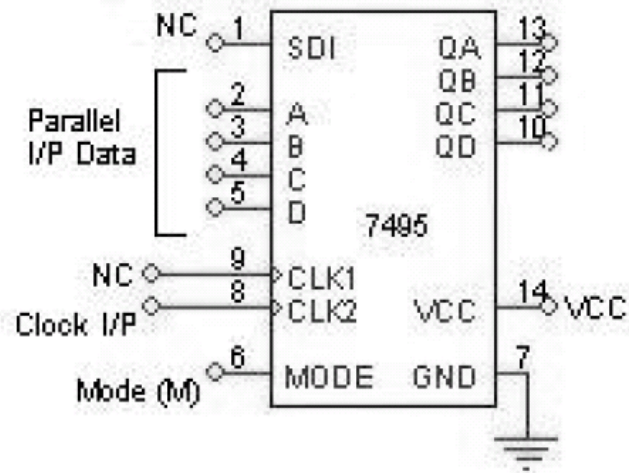
SISO:-



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

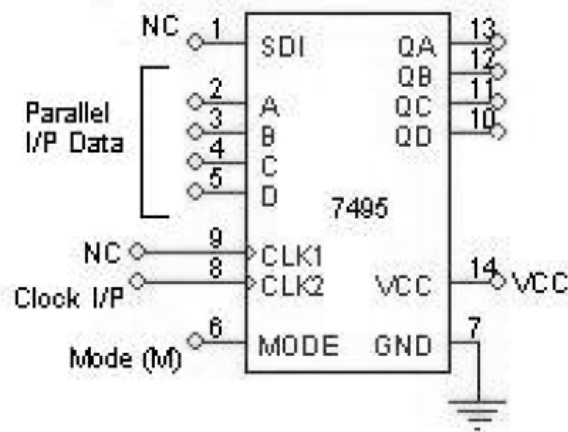
Figure 9.8: Pin-diagram and truth-table of SISO & SIPO register IC 7495

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

Figure 9.9: Pin-diagram and truth-table of PISO & PIPO register IC 7495

PROCEDURE:

Serial In Parallel Out (SIPO):

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

Serial In Serial Out (SISO):

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

Parallel In Serial Out (PISO):

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

Parallel In Parallel Out (PIPO):

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

Output: Thus, the Serial in serial out, Serial in parallel out, Parallel in serial out and Parallel in parallel out shift registers were implemented using IC 7495.

Result: Shift registers using IC 7495 in all its modes i.e.SIPO/SISO, PISO/PIPO are verified by their truth table. When the value of output voltage is greater than 2.4V then it will be represented by logic 1 and if output voltage is less than 0.4 volt it will be represented by logic 0.

Discussion:

- Q1. What is bi-directional shift register and unidirectional shift register?
- Q2. Write the uses of a shift register?
- Q3. Shifting a register content to left by one bit position is equivalent to
(A) division by two. (B) addition by two.
(B) multiplication by two. (D) subtraction by two.
- Q4. Can a shift register be used as a counter? If yes, explain how?

Experiment No: 10

Aim: Design and construct BCD ripple counter and verify the function.

Apparatus / Component required: - IC tester, Digital trainer kit, Digital ICs 7476

Theory: A decimal counter follows a sequence of ten states and returns to 0 after the count of 9. Such counter must have at least four flip-flops to represent each decimal digit, since a decimal digit is represented by a binary code with at least four bits. The sequence of states in a decimal counter is indicated by the binary code used to represent a decimal digit. If BCD is used, the sequence of states is as shown in the state diagram of figure fig.1. This is similar to a binary counter, except that the state after 1001 (code for decimal 9) is 0000 (code for decimal 0).

State Diagram of decimal BCD counter

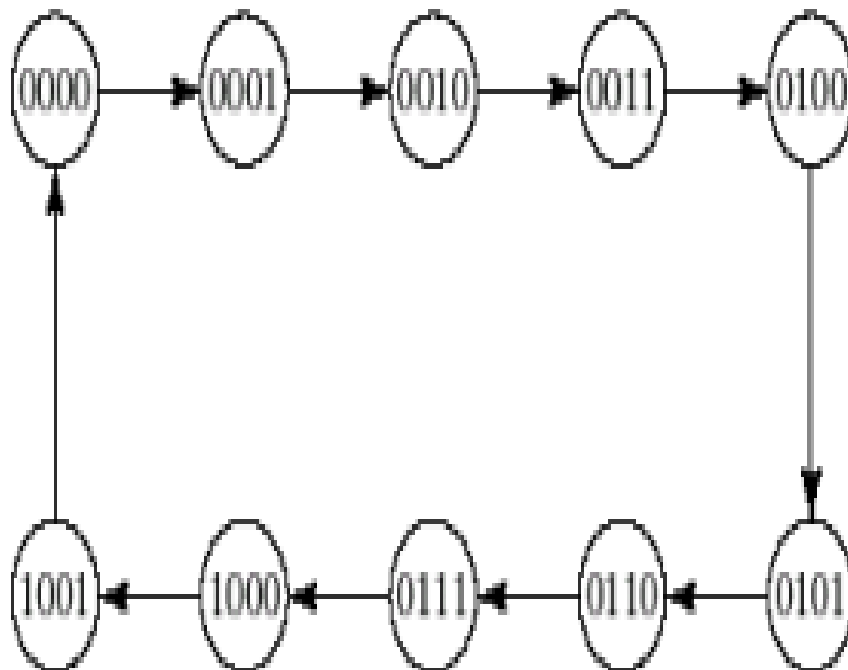


Figure 10.1: State Diagram

The logic diagram of a BCD ripple counter is shown in fig.2. The four outputs are designed by the letter symbol Q with a numeric subscript equal to the binary weight of the corresponding bit in the BCD code. The flip-flops trigger on the negative edge. Note that the output Q1 is applied to the Clk inputs of both Q2 and Q8 and the output of Q2 is applied to

the Clk input of Q4. The J and K inputs are connected either to a permanent 1 signal or to outputs of flip-flops, as shown in the diagram.

Logic Diagram of decimal BCD counter

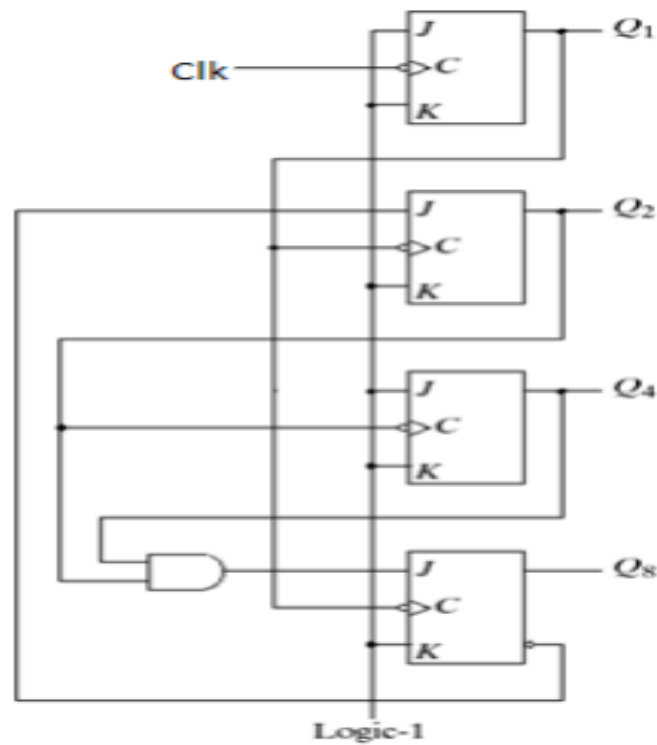


Figure 10.2: Logic Diagram

IC 7476 (JK FLIP FLOP)

CLK1	1		16	K1
$\overline{\text{PRE1}}$	2	I	15	Q1
$\overline{\text{CLR1}}$	3	C	14	$\overline{\text{Q1}}$
J1	4	7	13	GND
VCC	5	4	12	K2
CLK2	6	7	11	Q2
$\overline{\text{PRE2}}$	7	6	10	$\overline{\text{Q2}}$
$\overline{\text{CLR2}}$	8		9	J2

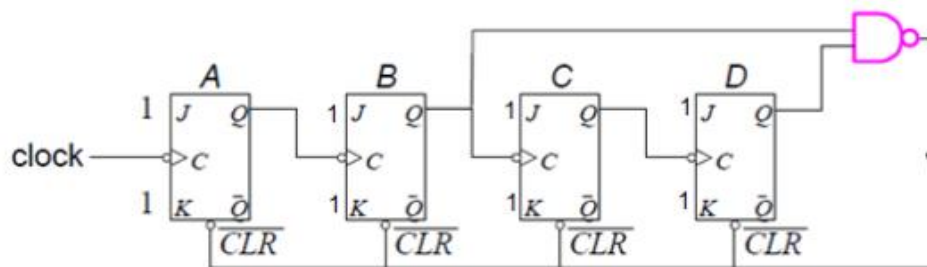
Figure 10.3: Pin-diagram of IC 7476

The following are the conditions for each flip-flop state transition: -

1. Q1 is complemented on the negative edge of every count pulse.
2. Q2 is complemented if Q8 = 0 and Q1 goes from 1 to 0.
Q2 is cleared if Q8 = 1 and Q1 goes from 1 to 0.
3. Q4 is complemented when Q2 goes from 1 to 0.
4. Q8 is complemented when Q4Q2 = 11 and Q1 goes from 1 to 0.
Q8 is cleared if either Q4 or Q2 is 0 and Q1 goes from 1 to 0.

Other presentation of BCD ripple counter will be as follows:

1. We need four flip-flops, for example JK flip flops where J and K are high.
2. Count from 0-9, so when counter reaches 9 it will make the 10 to be 0.
3. Decimal 10 = (1010)₂, clear = D C' B A'
4. Take ones as inputs of NAND and take output of NAND to CLR.



TRUTH TABLE

Truth Table				
count	Q _D	Q _C	Q _B	Q _A
0 [start]	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 [new cycle]	0	0	0	0

Output: We had designed a BCD ripple counter that counts from 0 to 9 and then return to 0.

Result: as we had studied in course that BCD is valid up to 9 so we designed the circuit for that and got same response as studied.

Discussion:

Q1. What do you mean by BCD?

Q2. Design a BCD combinational circuit?

Q3. Differentiate between: a) Synchronous and Asynchronous 2) BCD and decade counter

Q4. How does architecture of asynchronous UP counter differ from that of DOWN counter?

Experiment No: 11 (B1)

Aim: To realize a 4-bit ripple adder/Subtractor using basic half adder/ Subtractor and basic full adder/ Subtractor.

Apparatus / Component Required: IC tester, Digital trainer kit, Digital ICs (7483, 7486, 7404, 7408, and 7432)

Theory:

In digital circuits, an adder–Subtractor is a circuit that is capable of adding or subtracting numbers (in particular, binary). Adder-subtractor is a circuit that does adding or subtracting depending on a control signal. It is also possible to construct a circuit that performs both addition and subtraction at the same time.

4 Bit Binary Adder

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 .

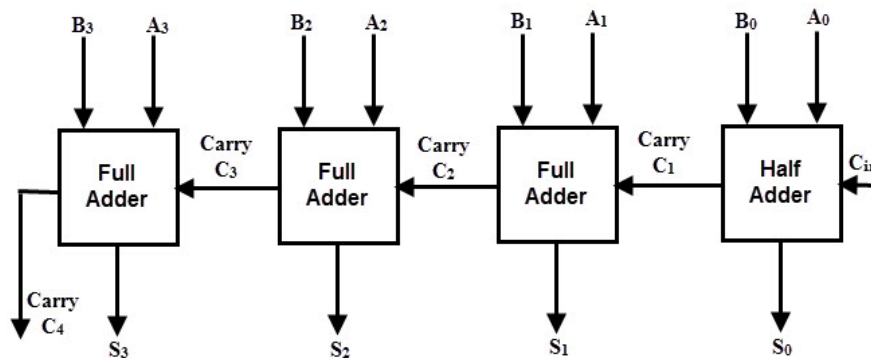


Figure 11.1: Block diagram of 4-bit parallel adder using full-adder

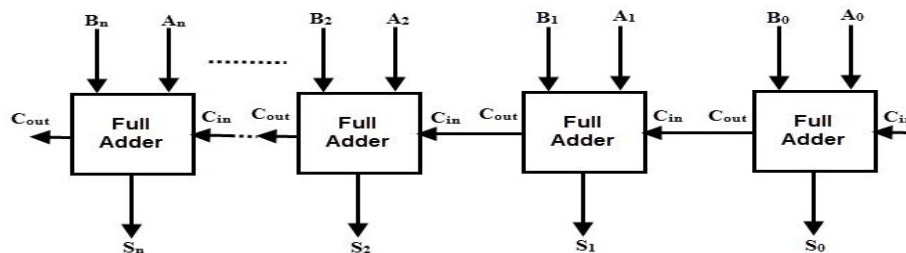


Figure 11.2: Block diagram of n-bit parallel adder using full-adder

4 Bit Binary Subtractor

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.

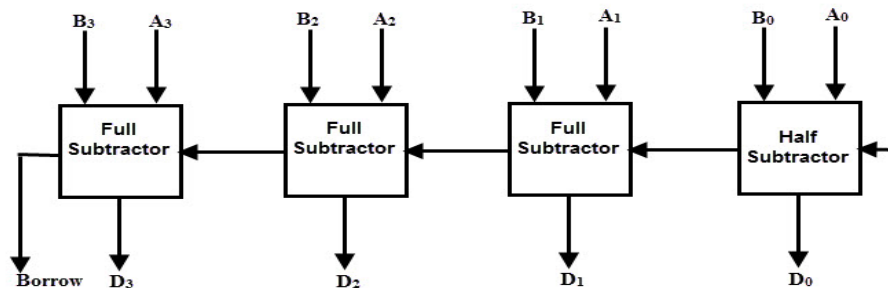


Figure 11.3: Block diagram of 4-bit binary subtractor using full-subtractor

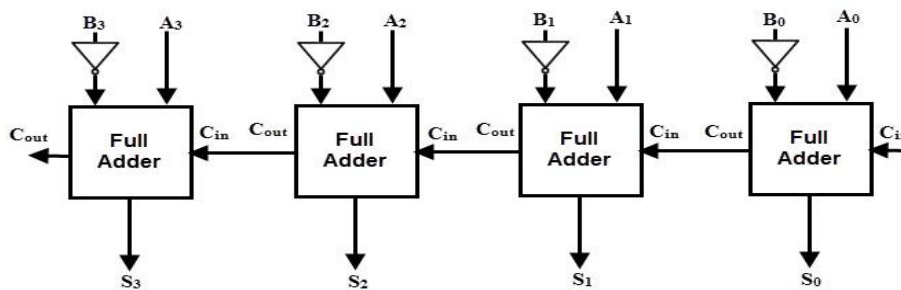


Figure 11.4: Block diagram of 4-bit binary subtractor using full-adder

4 Bit Binary Adder/Subtractor

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.

Pin Diagram for IC 7483

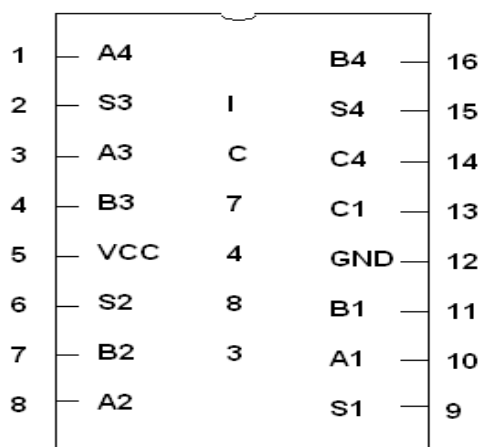


Figure 11.5: Pin-diagram of IC7483

LOGIC DIAGRAM:

Logic Diagram for 4-Bit Binary Adder

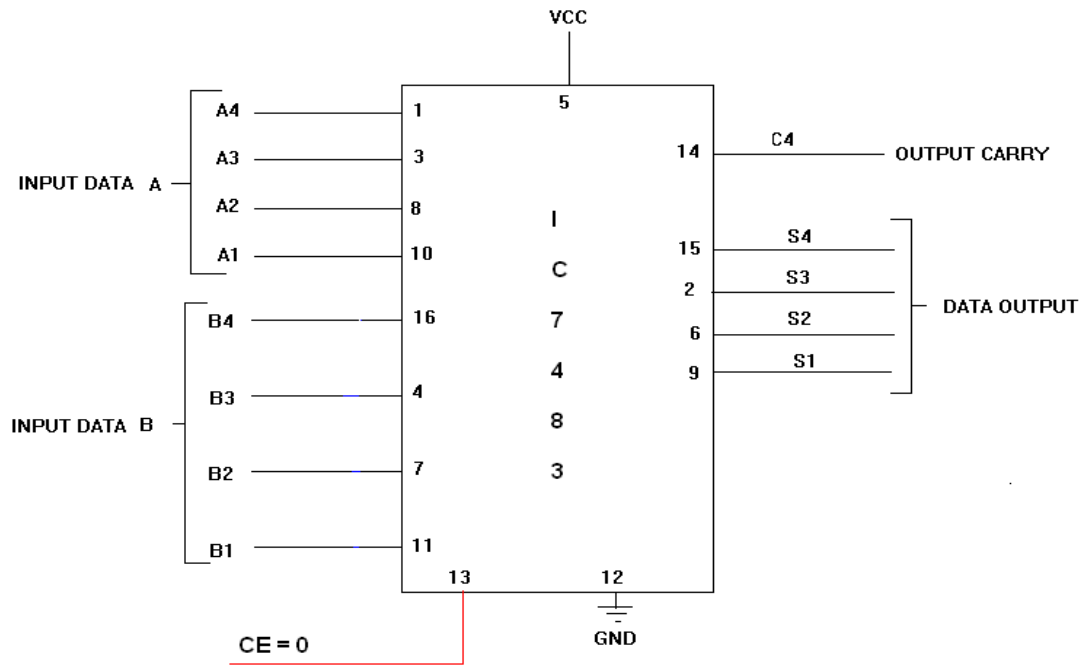


Figure 11.6

Logic Diagram for 4-Bit Binary Subtractor:

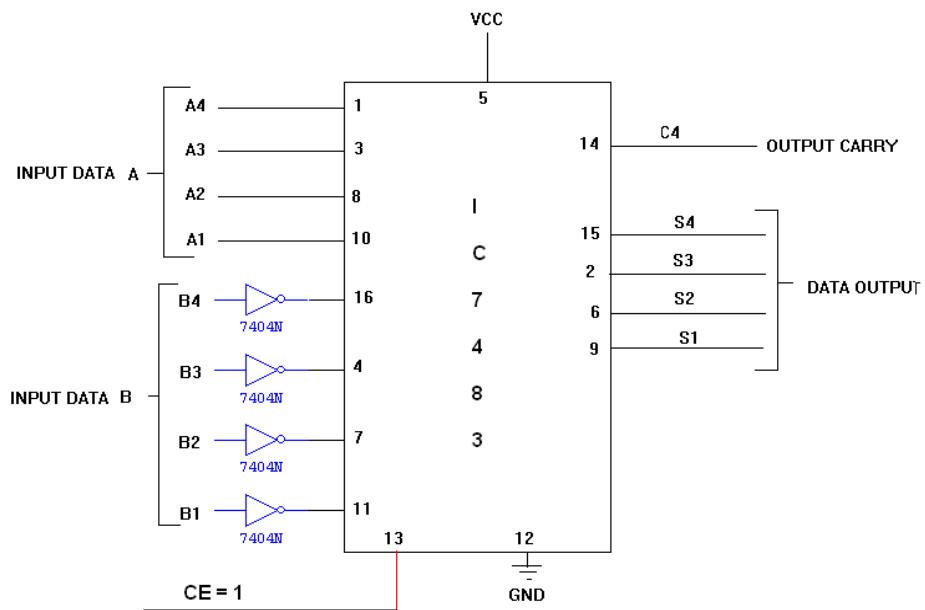


Figure 11.7

Logic Diagram for 4-Bit Binary Adder/Subtractor:

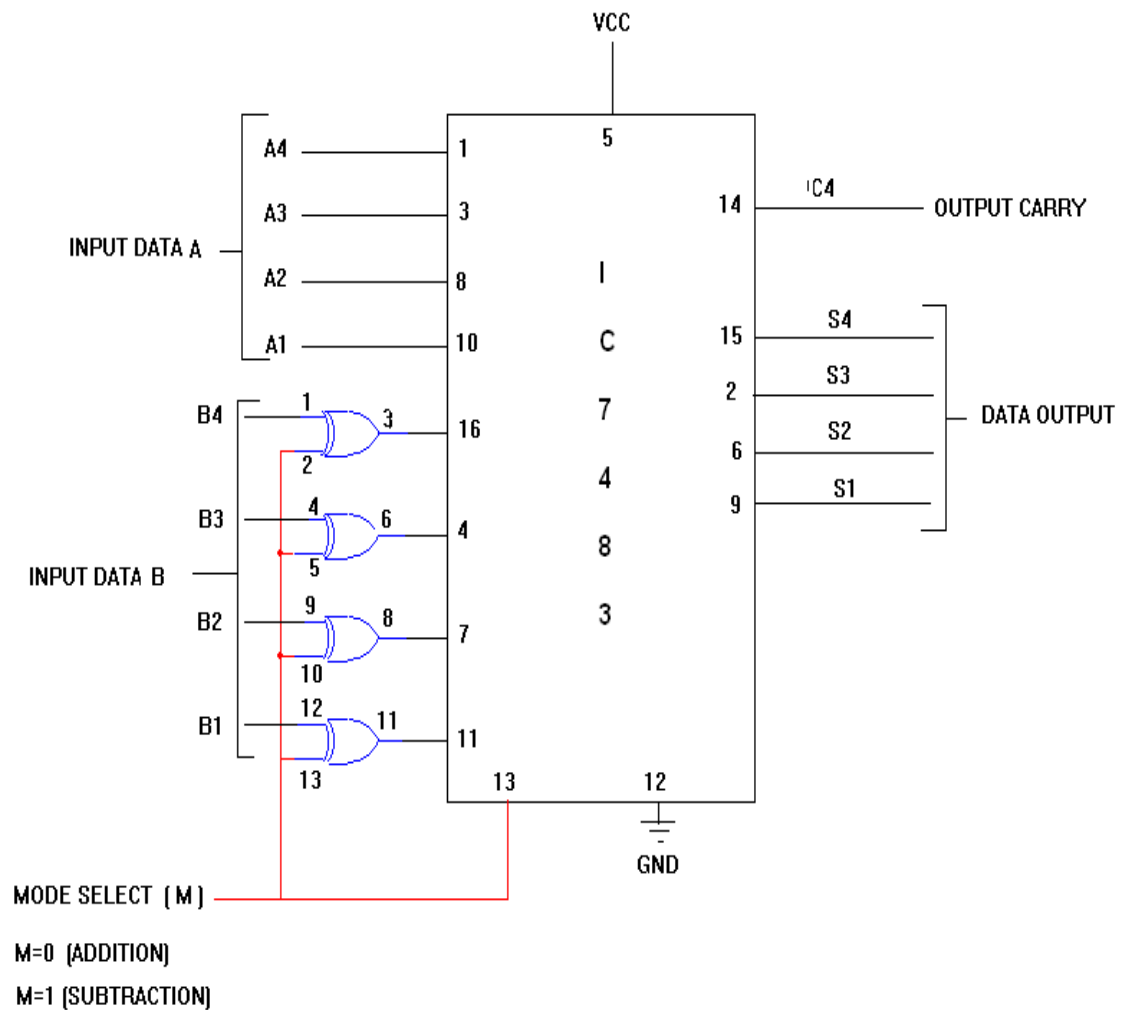


Figure 11.8

Observations for IC 7483

Symbol	Parameter		Min	Typ	Max	Unit
VCC	Supply Voltage	54 74	4.5 4.75	5.0 5.0	5.5 5.25	V
T _A	Operating Ambient Temperature Range	54 74	-55 0	25 25	125 70	°C
I _{OH}	Output Current — High	54, 74			-0.4	mA
I _{OL}	Output Current — Low	54 74			4.0 8.0	mA

Procedure:

1. Test the digital ICs with the help of IC tester.
2. Make the connections as per the circuit diagram.
3. Connect the output pin on LED through Resistor.
4. Switch on V_{CC} and apply various combinations of input according to truth table.
5. Observe the logical output and verify with the truth tables.

Output: Thus the 4-bit adder and subtractor using IC 7483 was designed and implemented.

Result: A four-bit ripple carry adder/subtractor can be designed by using four full adders/subtractors. They can also be designed by using IC 7483 by using appropriate selection mode (M).

Discussion:

- Q1. What is BCD adder?
- Q2. What is the difference between adder and parallel adder?
- Q3. How many full adders are required to construct an m-bit parallel adder?
- Q4. What are the disadvantages of the ripple-carry adder/Subtractor?

Experiment No: 12 (B2)

Aim: To design 1-to-4 demultiplexer using basic gates and verify the truth table. Also, to construct 1-to-8 demultiplexer using blocks of 1-to-4 demultiplexer

Apparatus / Component Required: IC tester, Digital trainer kit, connecting wires, Digital ICs (74155, 7408, 7432, 7404, 7411)

Theory:

DEMULTIPLEXER:

The function of De-multiplexer 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 de-multiplexer. In the 1: 4 de-multiplexer 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.

BLOCK DIAGRAM FOR 1:4 DEMULTIPLEXER:

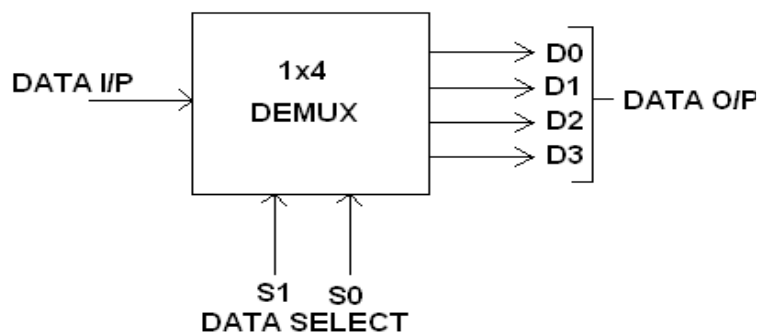


Figure: 12.1: Block diagram of 1:4 Demux

TRUTH TABLE:			OUTPUT			
INPUT			D0	D1	D2	D3
S1	S0	I/P				
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

LOGIC DIAGRAM FOR DEMULTIPLEXER:

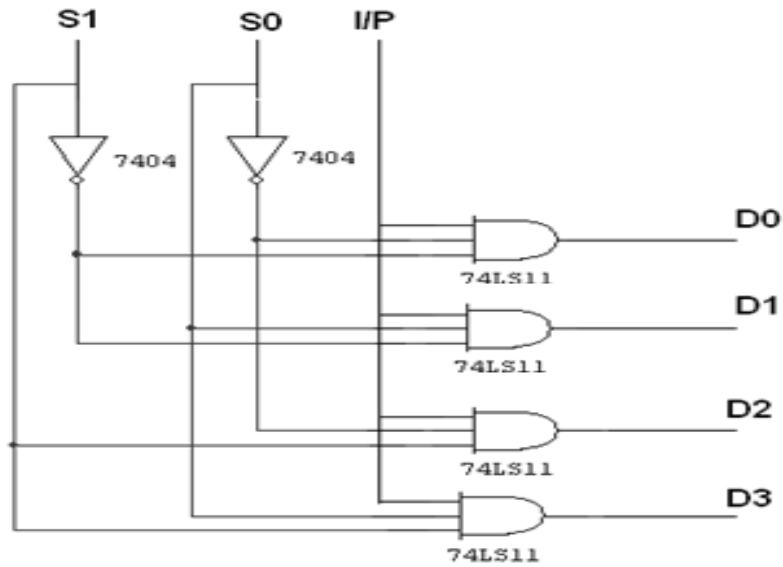


Figure: 12.2: Logic diagram of 1:4 Demux

PIN DIAGRAM OF DUAL 1:4 DEMUX (IC74155)

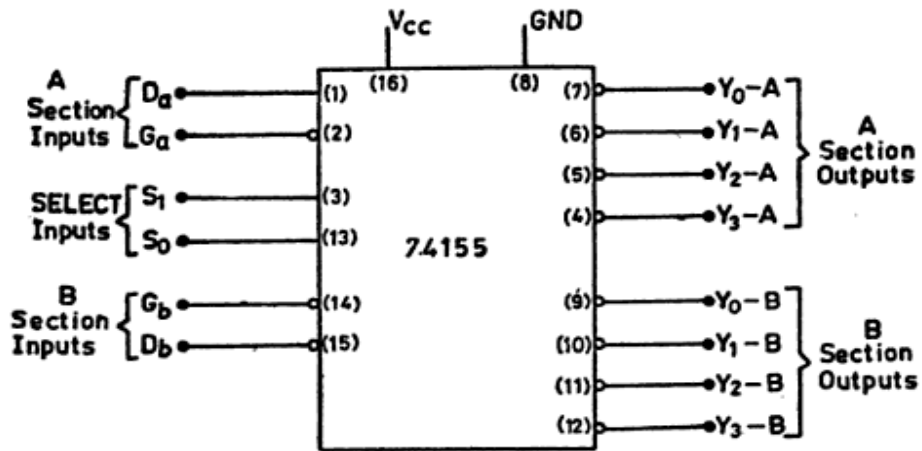


Figure: 12.3: Pin-diagram of 1:4 IC74155

TRUTH TABLE OF 1:4 DEMUX (IC 74155) WITH ACTIVE LOW MODE:

Input(Channel A)			Output			
E _a	S ₀	S ₁	D ₀	D ₁	D ₂	D ₃
1	×	×	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

Observations for IC 74155

Symbol	Parameter	Min	Nom	Max	Units
V_{CC}	Supply Voltage	4.75	5	5.25	V
V_{IH}	HIGH Level Input Voltage	2			V
V_{IL}	LOW Level Input Voltage			0.8	V
V_{OH}	HIGH Level Output Current			-0.4	mA
I_{OL}	LOW Level Output Current			8	mA
T_A	Free Air Operating Temperature	0		70	°C

Procedure:

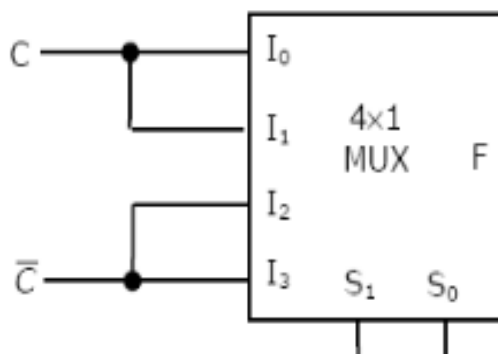
1. Test the digital ICs with the help of IC tester.
2. Make the connections as per the circuit diagram.
3. Connect the output pin on LED through Resistor.
4. Switch on V_{CC} and apply various combinations of input according to truth table.
5. Observe the logical output and verify with the truth tables.

Output: Hence the 1:4 de-mux is designed with basic gate as given in the figure and verified with the truth table. 1:8 de-mux is designed with 1:4 de-mux as given in circuit.

Result: 1:8 Demultiplexer is designed by using two 1:4 demultiplexer (IC 74155) by choosing appropriate selection lines. The demultiplexer was also designed USING IC 74155 and verified the truth table.

Discussion:

- Q1. How many select lines will a 32:1 multiplexer will have.
 Q2. Give the applications of Demultiplexer.
 Q3. The logic realized by the circuit shown in figure is



- (a) $F = A.C$
 (b) $F = A + C$
 (c) $F = B.C$
 (d) $F = B + C$

Experiment No: 13 (B3)

Aim: To design 2x4 decoder using basic gates and verify the truth table. Also verify the truth table of 3x8 decoder using IC.

Apparatus / Component required: - IC tester, Digital trainer kit, Digital ICs 74138, 7404, 7408

Theory:**Decoder: -**

In a digital system, discrete quantities of information are represented with binary codes. A binary code of n bits can represent up to 2^n distinct elements of the coded information. A *decoder* is a combinational circuit that converts n bits of binary information of input lines to a maximum of 2^n unique output lines. Usually, decoders are designated as an n to m lines decoder, where n is the number of input lines and $m (=2^n)$ is the number of output lines.

The 2-to-4 line decoder consists of two input variables and eight output lines. Note that each of the output lines represents one of the minterms generated from two variables. The internal combinational circuit is realized with the help of INVERTER gates and AND gates. Block diagram of 2-to-4 line decoder is shown below.

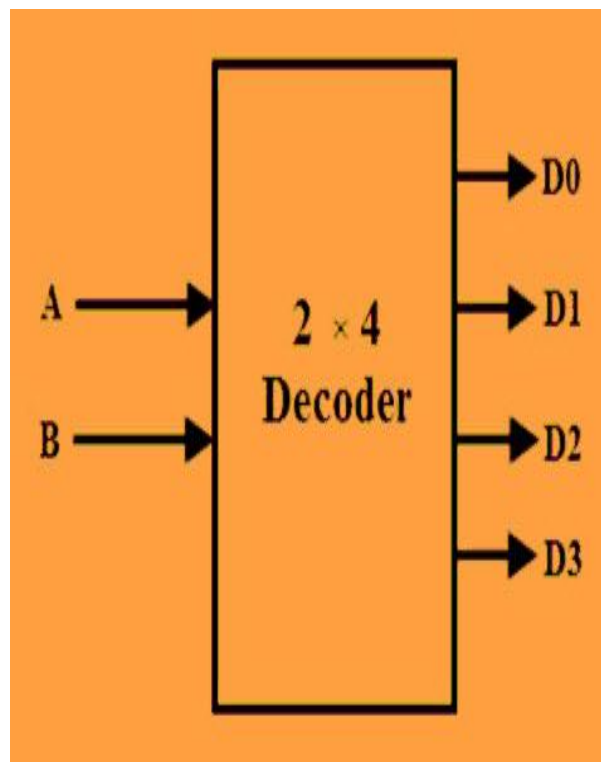


Figure 13.1: Block diagram of 2:4 Decoder

TRUTH TABLE OF 2-TO-4 LINE DECODER

TRUTH TABLE						
INPUTS			OUTPUTS			
B	A	IN	OUT ₀	OUT ₁	OUT ₂	OUT ₃
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

IMPLEMENTATION OF 2-TO-4 LINE DECODER

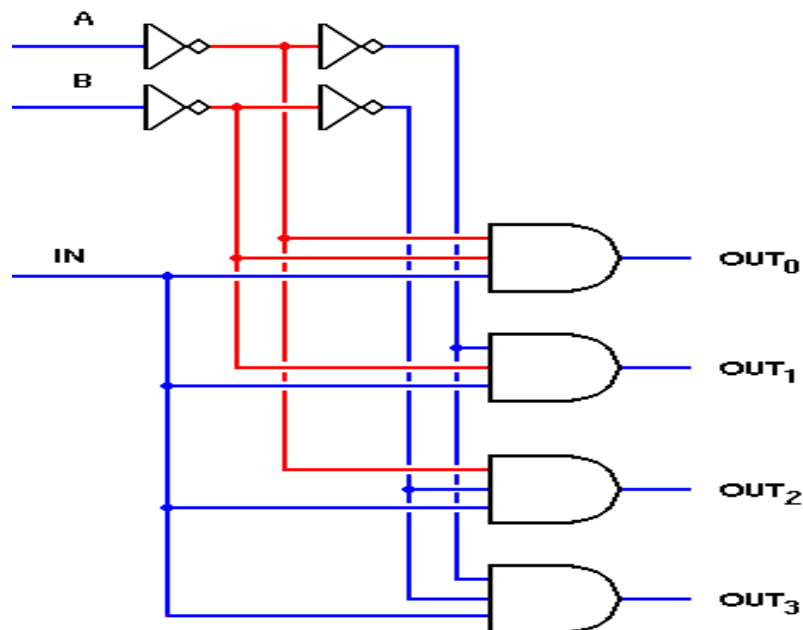


Figure 13.2: Logic-diagram of 2:4 Decoder

3-TO-8 LINE DECODER

The 3-to-8 line decoder consists of three input variables and eight output lines. Note that each of the output lines represents one of the minterms generated from three variables. The internal combinational circuit is realized with the help of INVERTER gates and AND gates. By using IC-74LS138 we can perform the 3X8 Decoder and by using 2 3X8 Decoder and also the NOT Gate we can perform 4X16 Decoder. We can also check the truth table.

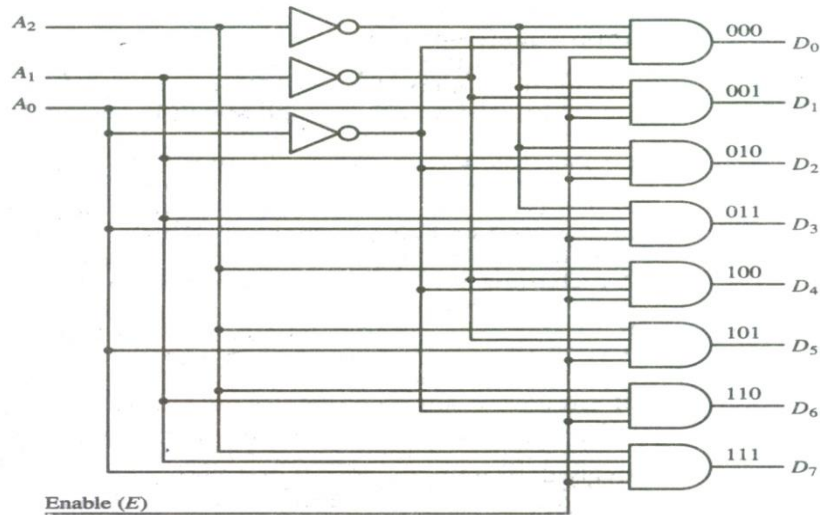


Figure 13.3: Logic Diagram for 3-to-8 Decoder

Truth Table for 3-to-8 Decoder

TRUTH TABLE

INPUTS						OUTPUTS							
E ₁	E ₂	E ₃	A ₀	A ₁	A ₂	O ₀	O ₁	O ₂	O ₃	O ₄	O ₅	O ₆	O ₇
H	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	L	X	X	X	H	H	H	H	H	H	H	H
L	L	H	L	L	L	L	H	H	H	H	H	H	H
L	L	H	H	L	L	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	L	H	H	H	H	H
L	L	H	H	H	L	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H
L	L	H	L	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	L

H = HIGH Voltage Level
 L = LOW Voltage Level
 X = Don't Care

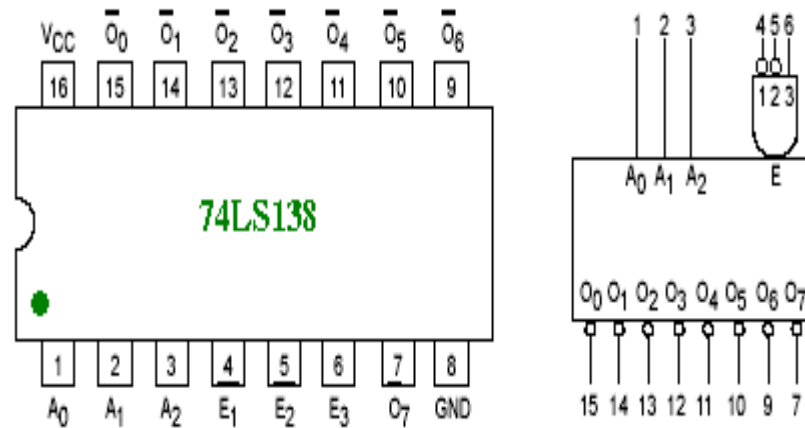


Figure 13.4: Pin-diagram of IC 74138

Procedure:

- Using basic gates, make circuit of 2:4 decoder on breadboard and check out as done earlier.
- Draw block diagram, truth table and pin diagram for 74138 IC and 7404 IC.
- According to pin assignment, connect the circuit as shown in fig. by using connecting wires.
- Switch 'ON' the power supply.
- Apply the corresponding inputs and verify the truth table.
- Note down the output reading according to the truth tables for 3 X 8 Decoders.

Output: By using basic gates and IC-74LS138 we can perform the different line decoder and verify the truth table.

Result: Thus, we have concluded that the inputs to a decoder are the bits 1, 0 and their combinations. The output is the corresponding decimal number. It converts binary information from n input lines to a maximum of 2^n unique output lines.

Discussion:

- Q1. What is the difference between Encoder and Decoder?
- Q2. What is the difference between combinational and sequential circuits?
- Q3. What is the difference between decoder and demultiplexer?
- Q4. Design a Full adder using two half adders.

Experiment No: 14 (B4)

Aim: Design and construct a 4 Bit Ring counter and verify the function.

Apparatus / Component required: - IC tester, Digital trainer kit, Digital ICs 7474

Theory: The ring counter is basically a shift register and is therefore also called ring shift register. In a ring counter the output of the last flip-flop is connected to the input of the first flip-flop. In a ring counter, the information is shifted at each clock signal Clk by one position or flip-flop. Once the information reaches the last flip-flop, at the next clock signal it is pushed back to the first flip-flop...

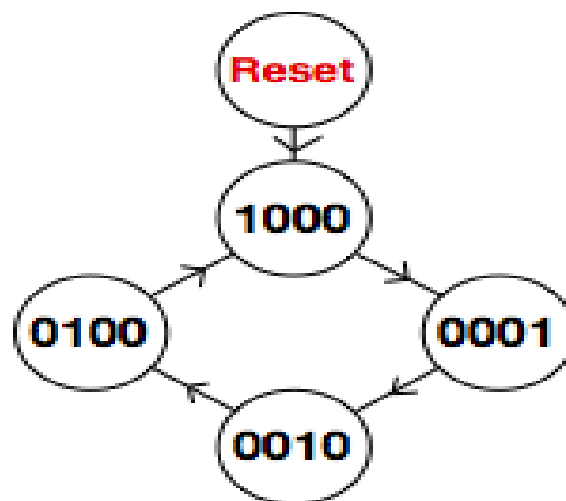
State Diagram of ring counter

Figure 14.1: State Diagram of ring counter

Circuit Diagram of Ring Counter

Take a look at the circuit diagram. The shown ring counter consists of four D flip-flops. With Reset you set the counter to (1000). Thereafter, the ring counter at each clock signal passes through four states (mod-4 counter).

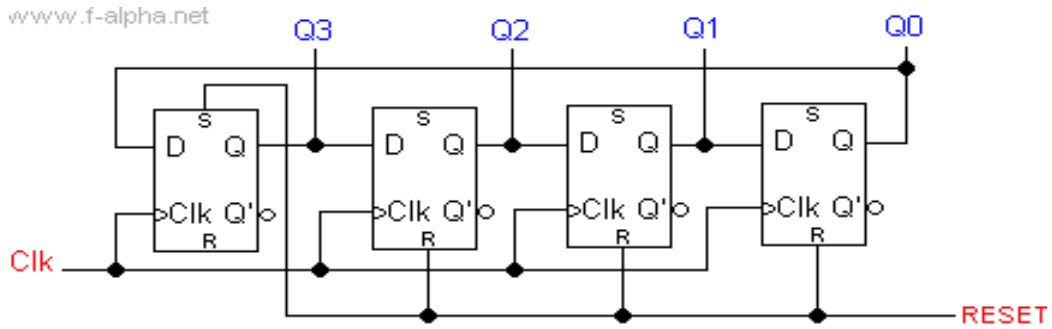


Figure 14.2: Logic Diagram of ring counter

Waveforms for ring counter

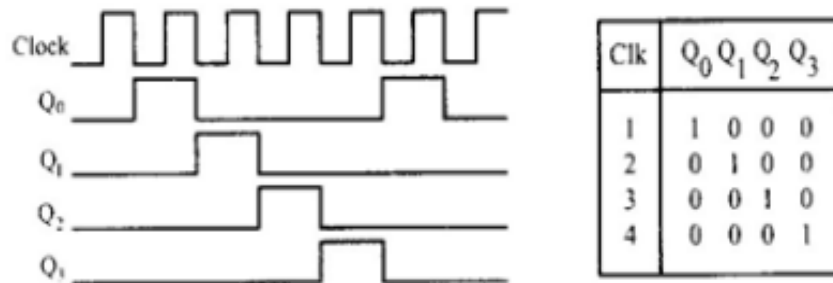


Figure 14.3: Timing waveforms of ring counter

As it can be seen from the truth table there are four unique output stages for this counter. The modulus value of a ring counter is n, where n is the number of flip flops. Ring counter is called divided by N counter where N is the number of FF.

IC 7474 (D FLIP FLOP)

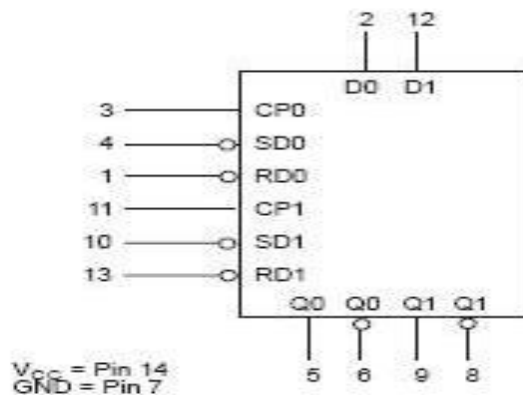


Figure 14.4: Pin-diagram of IC 7474

Output: We had designed a 4-bit ring counter using DFF.

Result: as we had studied in course that Ring counter make ring of different sequences and repeat it, so we designed the circuit for that and got same response as studied.

Discussion:

- Q1. Draw the circuit diagram of a basic ring counter?
- Q2. When is a counter said to suffer from lockout?
- Q3. What is the minimum number of flip-flops needed to build a counter of modulus 60?
- Q4. What is a state diagram?

Experiment No: 15 (B5)

Aim: Perform input/output operations on parallel in/Parallel out and Serial in/Serial out registers using clock. Also exercise loading only one of multiple values into the register using multiplexer.

Apparatus / Component Required: IC tester, Digital trainer kit, Digital ICs 7474

Theory:

A shift register is an n-bit register with provision for shifting its stored data by one position at each clock pulse. The logical configuration of a shift register consists of a chain of flip-flops connected in cascade, with the output of one flip-flop connected to the input of the next flip-flop. All flip-flops receive a common clock pulse which causes the shift from one stage to the next. Fig. 1 shows a simple shift register configuration. The new bit to be shifted into one end must be specified, and the bit shifted off the other end is lost unless it is saved externally. Although Fig. 1 shows a right-shift register, the same register can obviously be used for left shifts simply by reversing the sense of the bits. Most shift registers have provision for shifting only in one direction, but some have a control input that allows either left or right shifting to be specified at each clock.

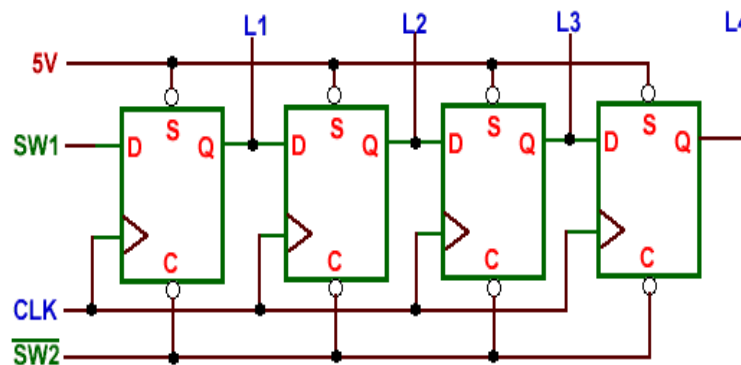


Figure 15.1: Simple Shift Register

Depending on the way how the data can be entered or retrieved there are four possible modes of operation.

- 1) Serial in serial out (SISO)
- 2) Serial in parallel out (SIPO)
- 3) Parallel in serial out (PISO)
- 4) Parallel in parallel out (PIPO)

Serial in serial out (SISO)

A Serial-in Serial-out shift register can be implemented using D-type flip-flops joined together, the output of one flip-flop used as the input to the next flip-flop. The circuit for a 4-bit Serial-in Serial-out shift register is shown below.

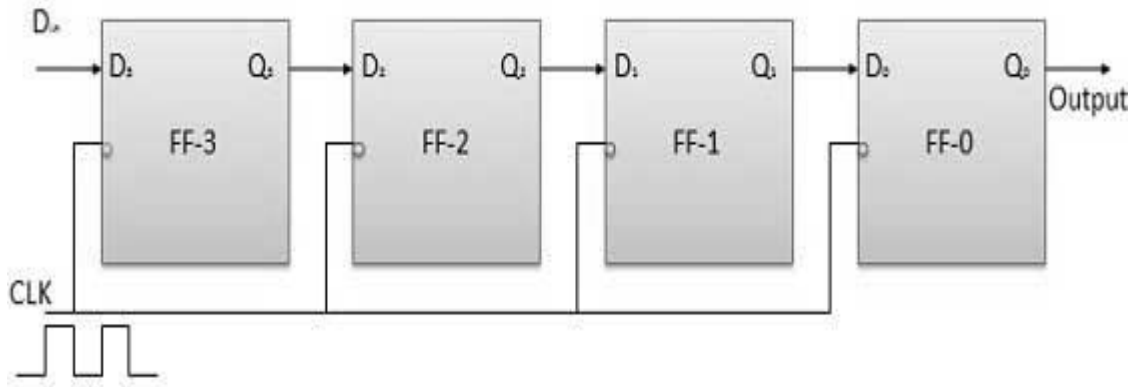


Figure 15.1: Logic – diagram of SISO

The operation of the serial-in Serial-out shift register can be easily explained. Consider the circuit shown. On each clock edge (rising in this case) we can say the following about the outputs of each stage of the register (i.e. each D-type flip-flop in the register):

	CLK	$D_{in} = Q_3$	$Q_3 = D_2$	$Q_2 = D_1$	$Q_1 = D_0$	Q_0
Initially			0	0	0	0
(i)	↓	1	1	0	0	0
(ii)	↓	1	1	1	0	0
(iii)	↓	1	1	1	1	0
(iv)	↓	1	1	1	1	1

→ Direction of data travel

Figure 15.2: Timing states of SISO

Waveforms

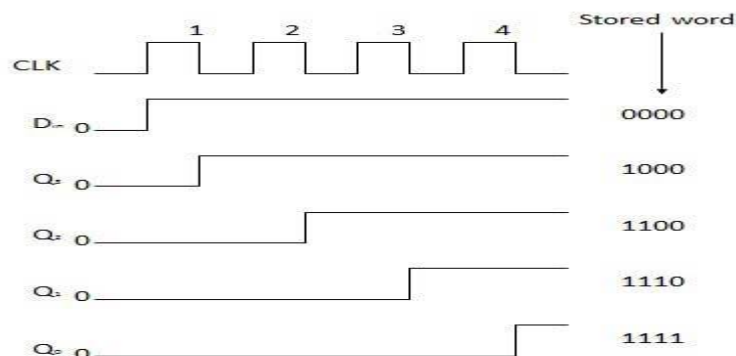


Figure 15.3: Timing waveforms of SISO

Parallel-in to Parallel-out (PIPO)

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

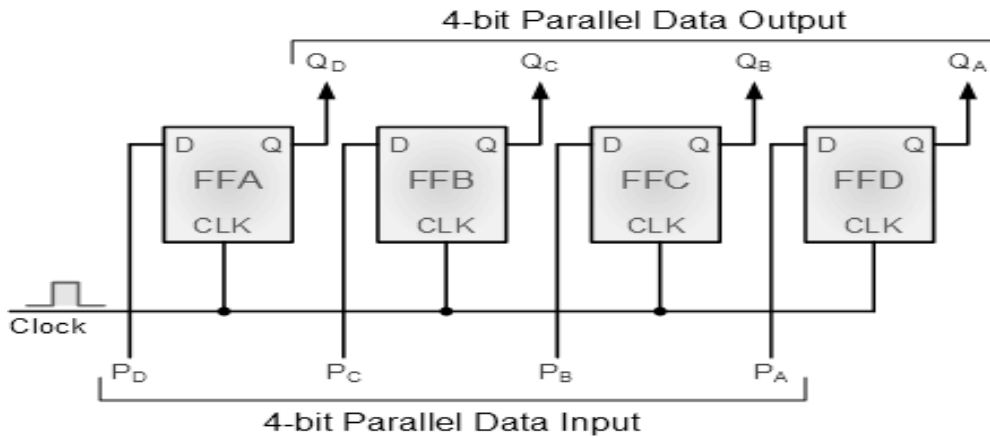


Figure 15.4: Logic – diagram of PIPO

TRUTH TABLE:

CLK	DATA INPUT				OUTPUT			
	DA	DB	DC	DD	QA	QB	QC	QD
1	1	0	0	1	1	0	0	1
2	1	0	1	0	1	0	1	0

Pin Diagram for IC 7495:

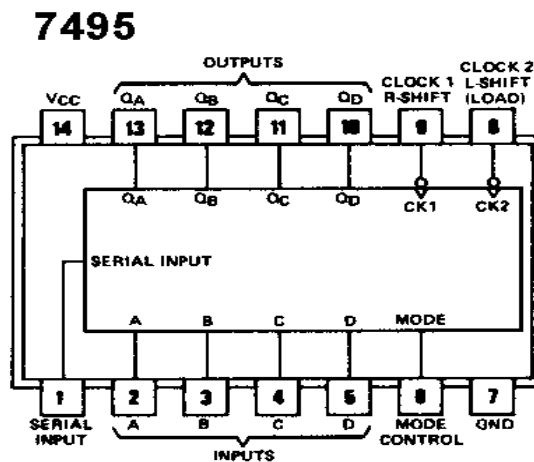


Figure 15.5: Pin-diagram of IC 7495

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

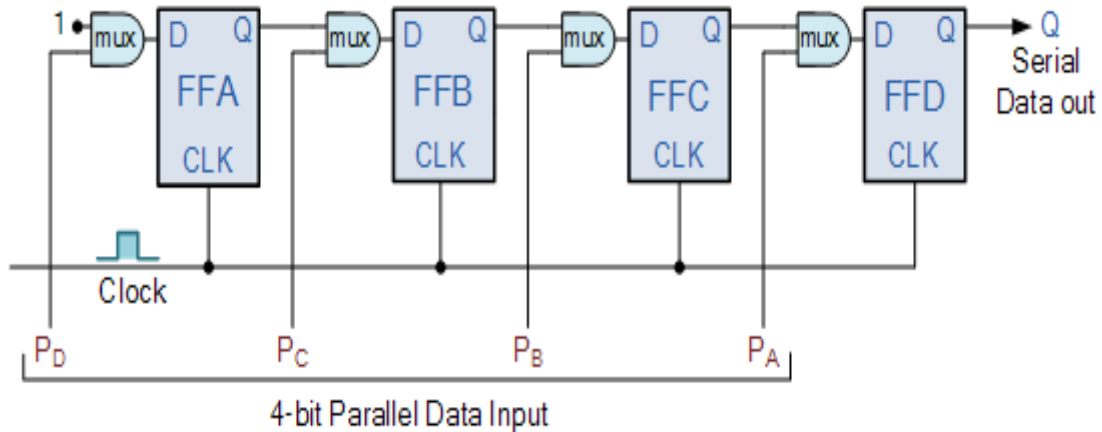


Figure 15.6: Logic – diagram of PISO

As this type of shift register converts parallel data, such as an 8-bit data word into serial format, it can be used to multiplex many different input lines into a single serial DATA stream which can be sent directly to a computer or transmitted over a communications line. Commonly available IC's include the 74HC166 8-bit Parallel-in/Serial-out Shift Registers.

CLK	Q3	Q2	Q1	Q0	O/P
0	1	0	0	1	1
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	1

PROCEDURE:

Serial In Serial Out (SISO):

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

Parallel In Serial Out (PISO):

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

Parallel In Parallel Out (PIPO):

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

Output: Thus, the Serial in serial out, Parallel in serial out and Parallel in parallel out shift registers were implemented using IC 7495.

Result: shift registers using IC 7495 in all its modes i.e. SISO, PISO/PIPO are verified and outputs are same as studied.

Discussion:

- Q1. What is bi-directional shift register and unidirectional shift register?
- Q2. Write the uses of a shift register?
- Q3. Explain the function of SHIFT/LOAD input in PISO shift register?
- Q4. Draw the Logic diagram of Universal shift register.
- Q5. Write the applications of Shift registers.
- Q6. Draw the State Table and Block diagram of a Serial Adder.

Experiment No: 16 (Design)

Aim: Design a circuit that will do the assigned work as mentioned below:

A lawn-sprinkling system is controlled automatically by certain combinations of the following variables.

Season (S=1, if summer; 0, otherwise)

Moisture content of soil (M=1, if high; 0, if low)

Outside temperature (T=1, if high; 0, if low)

Outside humidity (H=1, if high; 0, if low)

The sprinkler is turned on under any of the following circumstances.

1. The moisture content is low in winter.
2. The temperature is high and the moisture content is low in summer.
3. The temperature is high and the humidity is high in summer.
4. The temperature is low and the moisture content is low in summer.
5. The temperature is high and the humidity is low.

Apparatus / Component Required: Digital trainer kit, connecting wires, Digital ICs 7400

Theory:

The given circumstances in 1,2,3,4,5 are expressed in terms of the defined variables S, M, T and H as $M'S'$, $TM'S$, THS , $T'M'S$, and TH' , respectively.

The Boolean expression is

$$\begin{aligned} F &= S'M'+SM'T+STH+SM'T'+TH' \\ &= 00XX+101X+1X11+100X+XX10 \end{aligned}$$

The expression in terms of minterms and maxterms are

$$F = \sum m (0,1,2,3,6,9,10,11,14,15)$$

$$= \Pi M (4,5,7,12,13)$$

The K-Map in SOP forms:

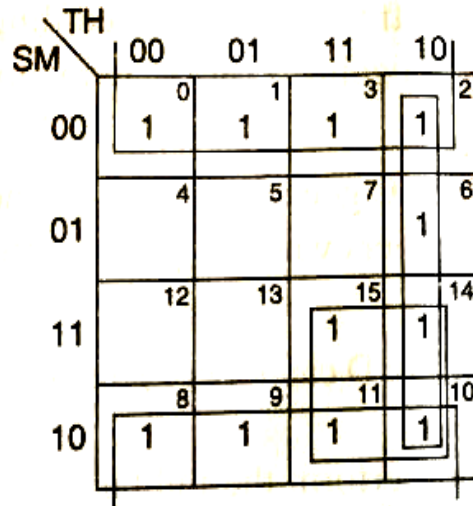


Figure 16.1

Minimal SOP expression = $M' + ST + TH'$

The K-Map in POS forms:

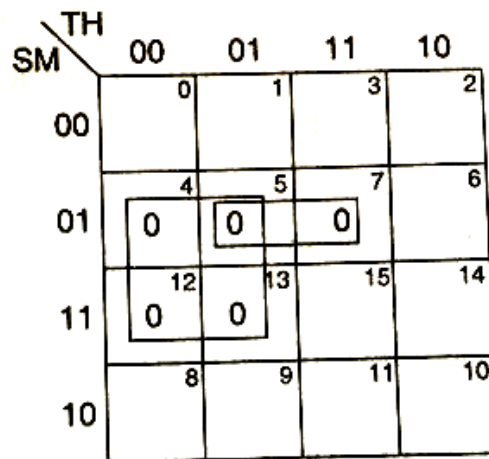


Figure 16.2

Minimal POS expression = $(M' + T)(S + M' + H')$

Implementation:

Logic diagram for SOP form is made with help of NAND gate:

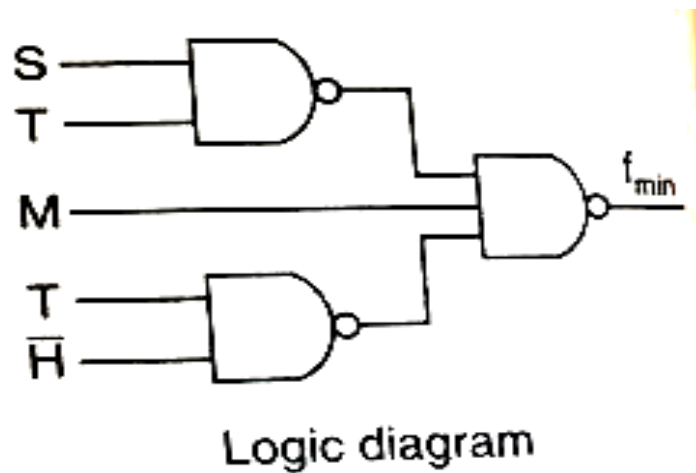


Figure 16.3: Final logic diagram of required design

Output: Thus, the automated lawn sprinkling system is designed by using digital IC which will work according to the specified conditions.

Result: Digital IC's can be used to design any kind of combinational circuits as well as sequential circuits like air-conditioning unit, room sensing system etc.

Discussion:

- Q1. What is arithmetic circuits?
- Q2. What is disadvantage of realizing a full-adder using two half-adders?
- Q3. What is parity bit generator?
- Q4. What is the type of displays used in calculators?