Assignment No: 1

Title: Full Adder/Substractor

Objective: To learn and understand design and construction of combinational circuit of Full adder and Substractor.

Outcomes: On completion of this practical, learner will be able to

CO1: Understand the working of Full adder and full Substractor circuits.

CO2: Apply the knowledge to gate IC as per the design specifications.

CO3: Design and implement Combinational digital circuits as per the specifications.

Problem Statement: To Realize Full Adder/ Subtractor using a) Basic Gates and b) Universal/special Gates.

Hardware Requirement:

i) IC 7404(NOT-gate), 7432 (OR-gate), 7408 (AND-gate), 7486 (Ex-or gate)

ii) Digital Trainer Kit -1

iii) Patch cords

Software Requirement: Digital Work 3.0

Theory:

Introduction

Adders are digital circuits that carry out addition of numbers. Adders are a key component of arithmetic logic unit. Adders can be constructed for most of the numerical representations like Binary Coded Decimal (BCD), Excess – 3, Gray code, Binary etc. out of these, binary addition is the most frequently performed task by most common adders. Apart from addition, adders are also used in certain digital applications like table index calculation, address decoding etc.

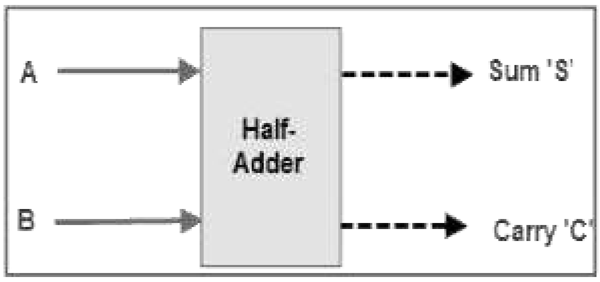
Binary addition is similar to that of decimal addition. Some basic binary additions are shown below.

```
 0       0       1            1
+0      +1      +0           +1
 0       1       1    (carry) 1 0
```

Figure 1. Schematic representation of half adder

1) Half Adder

Half adder is a combinational circuit that performs simple addition of two binary numbers. If we assume A and B as the two bits whose addition is to be performed, the block diagram and a truth table for half adder with A, B as inputs and Sum, Carry as outputs can be tabulated as follows.

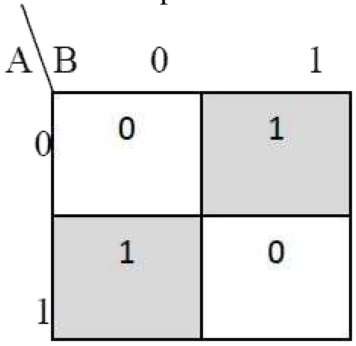


Truth Table			
Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Figure 2. Block diagram and truth table of half adder

The sum output of the binary addition carried out above is similar to that of an Ex-OR operation while the carry output is similar to that of an AND operation. The same can be verified with help of Karnaugh Map. The truth table and K Map simplification and logic diagram for sum output is shown below.

Truth Table		
A	B	SUM
0	0	0
0	1	1
1	0	1
1	1	0



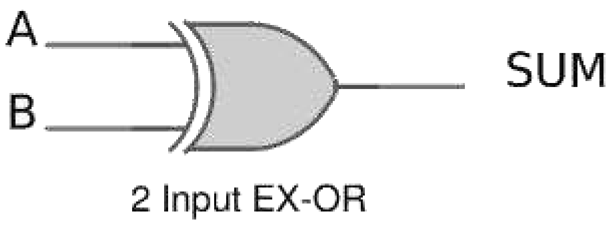
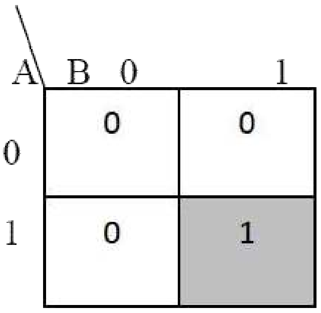


Figure 3. Truth table, K Map simplification and Logic diagram for sum output of half adder

Sum = A B' + A' B

The truth table and K Map simplification and logic diagram for carry is shown below.

Truth Table		
A	B	Carry
0	0	0
0	1	0
1	0	0
1	1	1



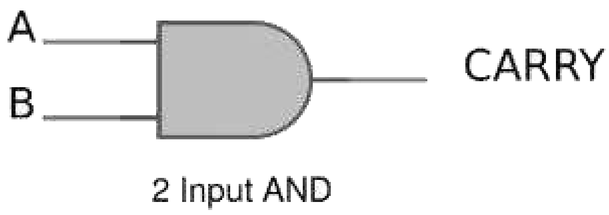


Figure 4. Truth table, K Map simplification and Logic diagram for sum output of half adder

Carry = AB

If A and B are binary inputs to the half adder, then the logic function to calculate sum S is Ex – OR of A and B and logic functionto calculate carry C is AND of A and B. Combining these two, the logical circuit to implement the combinational circuit of half adder is shown below.

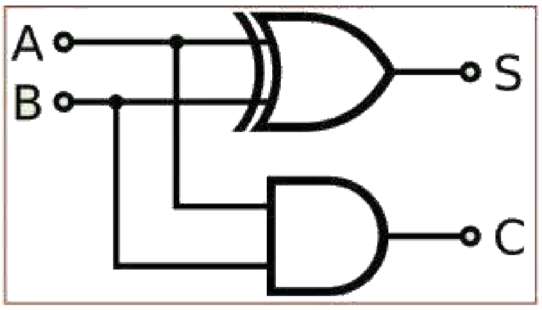


Figure 5. Half Adder Logic Diagram

As we know that NAND and NOR are called universal gates as any logic system can be implemented using these two, the half adder circuit can also be implemented using them. We know that a half adder circuit has one Ex – OR gate and one AND gate.

1.1) Half Adder using NAND gates

Five NAND gates are required in order to design a half adder. The circuit to realize half adder using NAND gates is shown below.

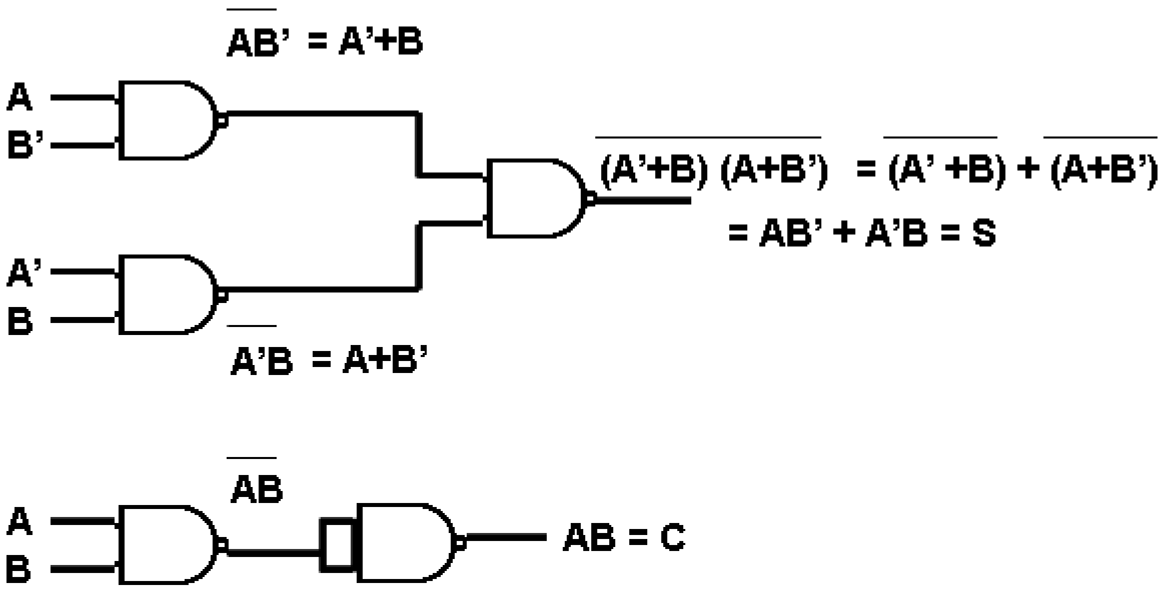
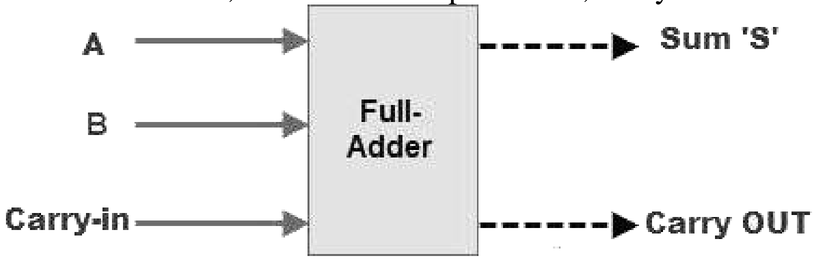


Figure 6. Realization of half adder using NAND gates

2) <u>Full Adder</u>

Full adder is a digital circuit used to calculate the sum of three binary bits. Full adders are complex and difficult to implement when compared to half adders. Two of the three bits are same as before which are A, the augend bit and B, the addend bit. The additional third bit is carry bit from the previous stage and is called 'Carry' – in generally represented by CIN. It calculates the sum of three bits along with the carry. The output carry is called Carry – out and is represented by Carry OUT.
The block diagram of a full adder with A, B and CIN as inputs and S, Carry OUT as outputs is shown below.



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

Figure 8. Full Adder Block Diagram and Truth Table

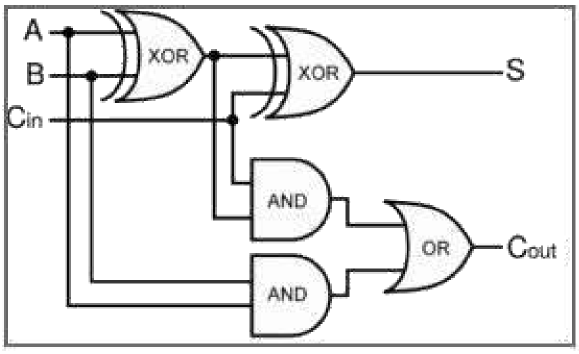


Figure 9. Full Adder Logic Diagram

Based on the truth table, the Boolean functions for Sum (S) and Carry – out (COUT) can be derived using K – Map.

A \ BC_{IN}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Figure 10. The K-Map simplified equation for sum is S = A'B'Cin + A'BCin' + ABCin

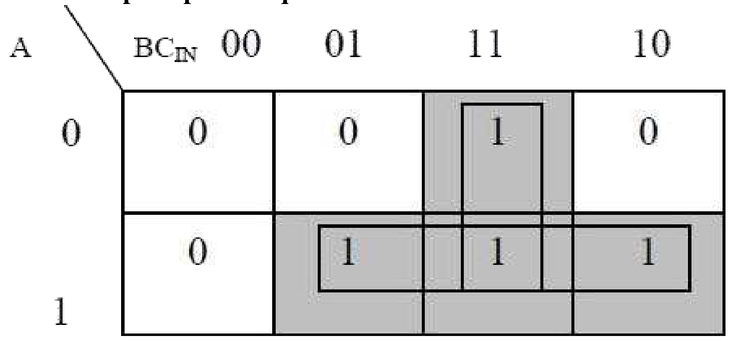


Figure 11. The K-Map simplified equation for COUT is COUT = AB + ACIN + BCIN

In order to implement a combinational circuit for full adder, it is clear from the equations derived above, that we need four 3-input AND gates and one 4-input OR gates for Sum and three 2-input AND gates and one 3-input OR gate for Carry – out.

2.1) Full Adder using NAND gates

As mentioned earlier, a NAND gate is one of the universal gates and can be used to implement any logic design. The circuit of fulladder using only NAND gates is shown below.

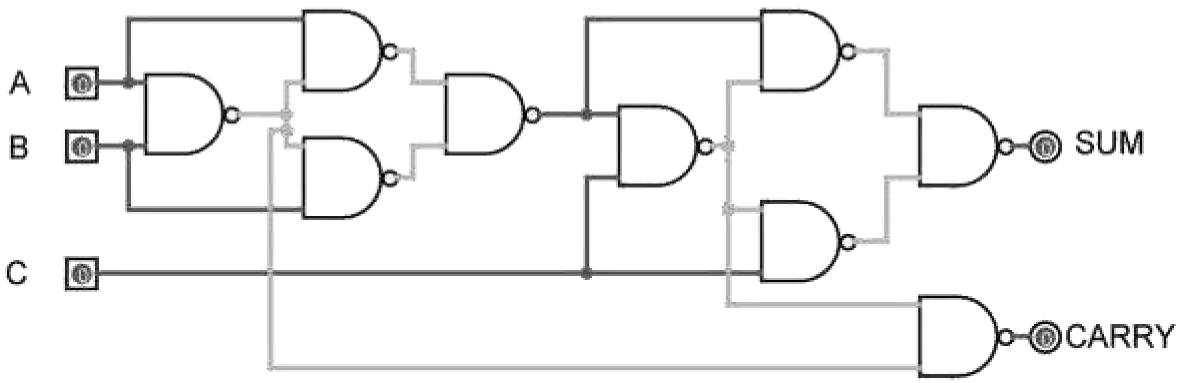


Figure 12. Full Adder using NAND gates

1) Half Subtractor

The half-Subtractor is a combinational circuit which is used to perform subtraction of two bits. It has two inputs, A (minuend) and B (subtrahend) and two outputs Difference and Borrow. The logic symbol and truth table are shown below.

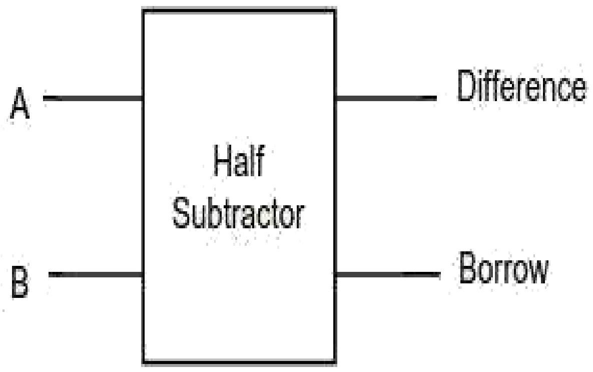
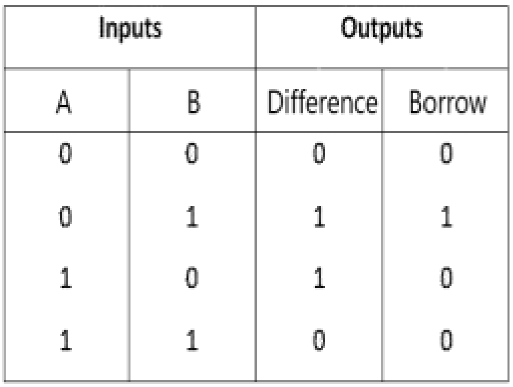
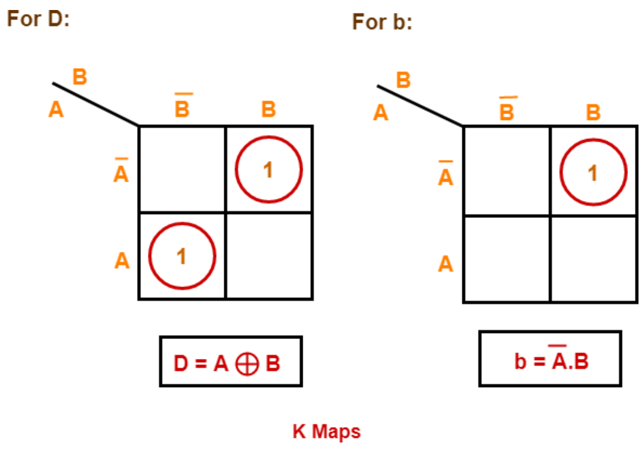


Figure-1: Logic Symbol of Half Subtractor



Inputs		Outputs	
A	B	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Figure-2: Truth Table of Half Subtractor



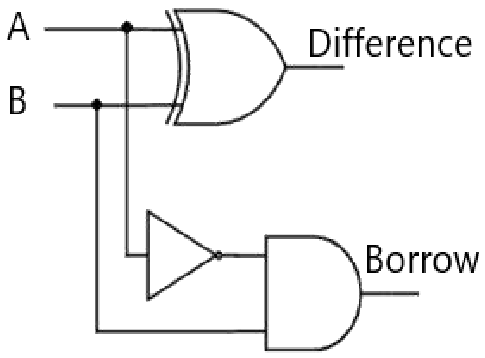


Figure-3: Circuit Diagram of Half Subtractor

From the above truth table we can find the Boolean expression.

Difference = A $\oplus$
BBorrow = A' B

From the equation we can draw the half-Subtractor circuit as shown in the figure 3.

2) Full Subtractor

A full Subtractor is a combinational circuit that performs subtraction involving three bits, namely A (minuend), B (subtrahend), and Bin (borrow-in). It accepts three inputs: A (minuend), B (subtrahend) and a Bin (borrow bit) and it produces two outputs: D (difference) and Bout (borrow out). The logic symbol and truth table are shown below.

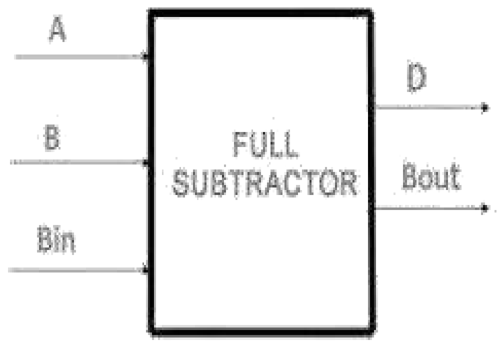
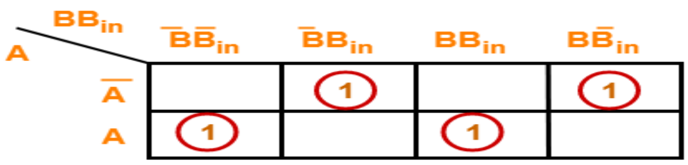


Figure-4:Logic Symbol of Full Subtractor

A	B	B_{in}	D	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

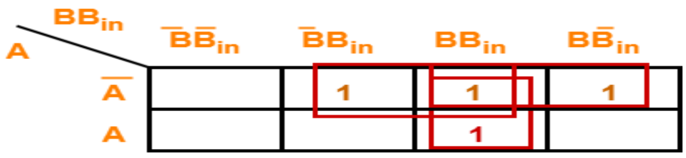
Figure-5:Truth Table of Full Subtractor

For D:



$$\boxed{D = A \oplus B \oplus B_{in}}$$

For B_{in}:



$$\boxed{B_{out} = \bar{A} B + (\bar{A} + B) B_{in}}$$

From the above truth table we can find the Boolean expression.

D = A ⊕ B ⊕ Bin
Bout = A' Bin + A' B + B Bin

From the equation we can draw the Full-Subtractor circuit as shown in the figure 6.

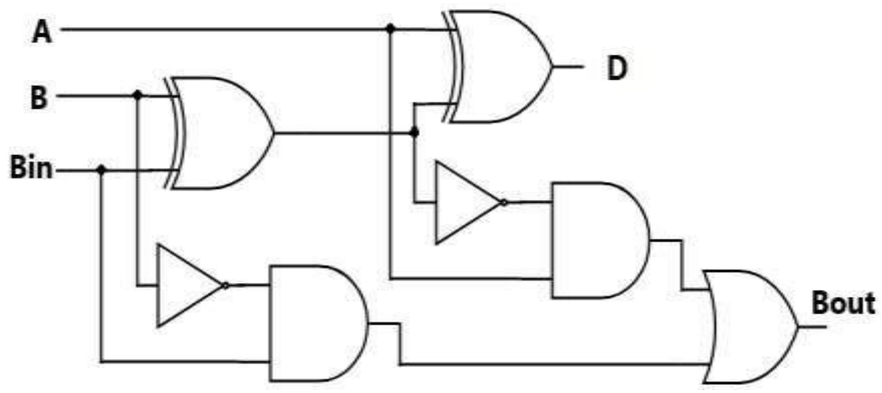


Figure-6: Circuit Diagram of Full Subtractor

Assignment No: 2

Title: Code Convertor

Problem Statement: Design and Implement Code Convertor:- Binary to Gray and BCD to Excess-3 code convertor.

Objective: To learn and understand design and construction of combination circuit Binary to Gray And BCD to Excess-3 code convertor.

Outcomes: On completion of these practical, students will be able to

CO1: Understand the working of code converter circuits.

CO2: Apply the knowledge of gate IC 7486 as per the design specifications.

.

Hardware Requirement:

i) IC 7404(Not-gate), 7432 (OR-gate), 7408 (AND-gate), 7486 (Ex-or gate)

ii) Digital Trainer Kit -1

iii) Patch cords

Software Requirement: Digital Work 3.0 or Virtual lab

Theory:

Logic circuits for digital systems may be combinational or sequential. A combinational circuit consists of input variables, logic gates, and output variables. Example of combinational circuit are Code Convertor, Adder, Subtractor, Multiplexer,Demultiplxer,Decoder etc

1. **Introduction about Binary codes and Code Convertor?**

There is a wide variety of binary codes used in digital systems. Some of these codes are binary-coded-decimal(BCD), Excess-3, Gray, octal, hexadecimal, etc. Often it is required to convert from one code to another.

For example the input to a digital system may be in natural BCD and output may be 7-segment LEDs. The digital system used may be capable of processing the data in straight binary format. A conversion circuit is necessary between two systems if each system uses different codes for the same information.

There is a wide variety of binary codes used in digital systems. Some of these codes are binary-coded-decimal(BCD), Excess-3, Gray, octal, hexadecimal, etc. Often it is required to convert from one code to another. For example the input to a digital system may be in natural BCD and output may be 7-segment LEDs. The digital system used may be capable of processing the data in straight binary format. A conversion circuit is necessary between two systems if each system uses different codes for the same information.

Therefore, the data has to be converted from one type of code to another type for different purpose. The various code converters can be designed using gates.

- ***What is binary code?***

It is straight binary code. The binary number system (with base 2) represents values using two symbols, typically 0 and 1.Computers call these bits as either off (0) or on (1). The binary code are made up of only zeros and ones, and used in computers to stand for letters and digits. It is used to represent numbers using natural or straight binary form.

It is a weighted code since a weight is assigned to every position. Various arithmetic operations can be performed in this form. Binary code is weighted and sequential code.

The digital data is represented, stored and transmitted as group of binary bits.This group is called as binary code.

- ***What is gray code?***

The Gray code also known as reflected binary code Bell Labs researcher Frank Gray introduced the term reflected binary code in his 1947.

It is a modified binary code in which a decimal number is represented in binary form in such a way that each Gray- Code number differs from the preceding and the succeeding number by a single bit. This feature allows a system designer to perform some error checking (i.e. if more than one bit changes, the data must be incorrect). (E.g. for decimal number 5 the equivalent Gray code is 0111 and for 6 it is 0101. These two codes differ by only one bit position i. e. third from the left.) Whereas by using binary code there is a possibility of change of all bits if we move from one number to other in sequence (e.g. binary code for 7 is 0111 and for 8 it is 1000). Therefore it is more useful to use Gray code in some applications than binary code.

The Gray code is a nonweighted code i.e. there are no specific weights assigned to the bit positions.

Like binary numbers, the Gray code can have any no. of bits. It is also known as reflected code.

- ***Applications for Gray code :***

1. Important feature of Gray code is it exhibits only a single bit change from one code word to the next in sequence. This property is important in many applications such as Shaft encoders where error susceptibility increases with number of bit changes between adjacent numbers in sequence.

2. It is sometimes convenient to use the Gray code to represent the digital data converted from the analog data (Outputs of ADC).

3. Gray codes are used in angle-measuring devices in preference to straight forward binary encoding.

4. Gray codes are widely used in K-map

The disadvantage of Gray code is that it is not good for arithmetic operation

- ***Binary To Gray Conversion***

In this conversion, the input straight binary number can easily be converted to its Gray code equivalent.

1. Record the most significant bit as it is.
2. EX-OR this bit to the next position bit, record the resultant bit.
3. Record successive EX-ORed bits until completed.
4. Convert 0011 binary to Gray.

0	0	1	1	Binary code
0	0	1	0	Gray code
(MSB)				(LSB)

Fig. 1 Binary To Gray Conversion

- ***BCD Code:***

Binary Coded Decimal (BCD) is used to represent each of decimal digits (0 to 9) with a 4-bit binary code. For example $(23)_{10}$ is represented by 0010 0011 using BCD code rather than$(10111)_2$ This code is also known as 8-4-2-1 code as 8421 indicates the binary weights of four bits(2^3, 2^2, 2^1, 2^0). It is easy to convert between BCD code numbers and the familiar decimal numbers. It is the main advantage of this code. With four bits, sixteen numbers (0000 to 1111) can be represented, but in BCD code only 10 of these are used. The six code combinations (1010 to 1111) are not used and are invalid.

- ***Applications:*** Some early computers processed BCD numbers. Arithmetic operations can be performed using this code. Input to a digital system may be in natural BCD and output may be 7-segment LEDs.

It is observed that more number of bits are required to code a decimal number using BCD code than using the straight binary code. However in spite of this disadvantage it is very convenient and useful code for input and output operations in digital systems.

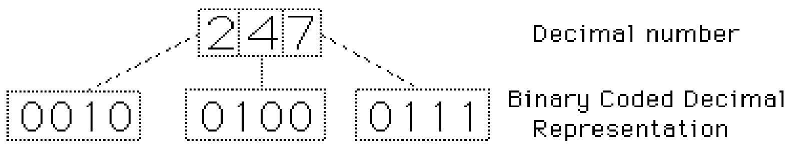


Fig. : BCD Coded Decimal Representation

- ***EXCESS-3 Code:***

Excess-3, also called XS3, is a non-weighted code used to express decimal numbers. It can be used for the representation of multi-digit decimal numbers as can BCD.The code for each decimal number is obtained by adding decimal 3 and then converting it to a 4-bit binary number. For e.g. decimal 2 is coded as 0010 + 0011 = 0101 in Excess-3 code.

This is self-complementing code which means 1's complement of the coded number yields 9's complement of the number itself. Self-complementing property of this helps considerably in performing subtraction operation in digital systems, so this code is used for certain arithmetic operations.

- ***BCD To Excess – 3 Code Conversions:***

Convert BCD 2 i. e. 0010 to Excess – 3 code

For converting 4 bit BCD code to Excess – 3, add 0011 i. e. decimal 3 to the respective code using rules of binary addition.

0010 + 0011 = 0101 – Excess – 3 code for BCD 2

2. **Design of Binary to Gray code Convertor.**

   Steps:

i. Truth table

ii. K-map

iii. Logic diagram

i. Truth table for Binary To Gray Code Conversion

Decimal No	Binary				Gray			
	B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

ii.K-map for B0,B1,B2,B3

B3B2 \ B1B0	00	10	11	01
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

G0=B0⊕B1

B3B2 \ B1B0	00	10	11	01
00	0	0	1	1
01	1	1	0	0
11	1	1	0	0
10	0	0	1	1

G1=B1⊕B2

B3B2 \ B1B0	00	10	11	01
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$G2=B2\oplus B3$

B3B2 \ B1B0	00	10	11	01
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

$G3=B3$

iii. Logic diagram

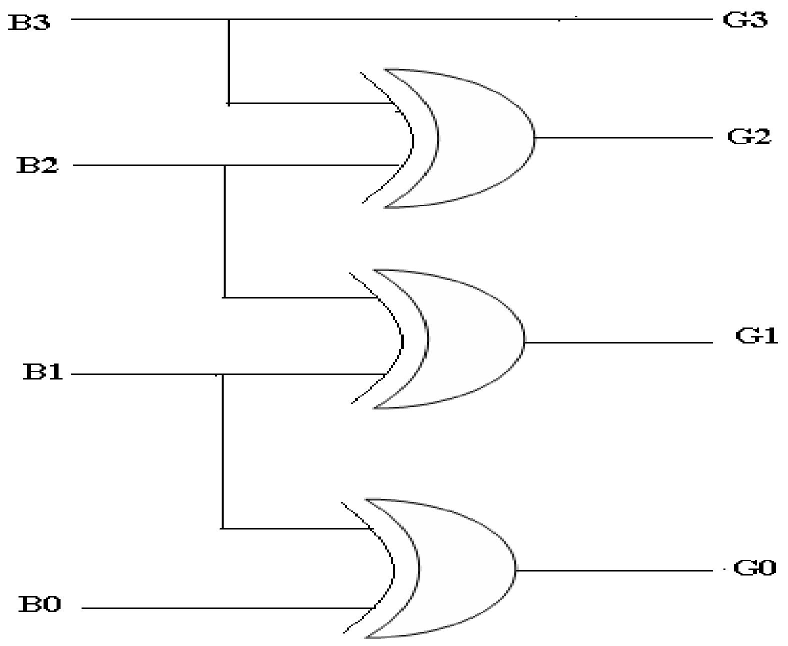


Fig.:- Logic diagram for binary to gray converter

3. Design BCD to Excess-3 Code Convertor.

Steps:

i. Truth table

ii. K-map

iii. Logic diagram

i. Truth Table:

INPUT (BCD CODE)				**OUTPUT (EXCESS-3 CODE)**			
B_3	**B_2**	**B_1**	**B_0**	**E_3**	**E_2**	**E_1**	**E_0**
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

ii. K-Map For Reduced Boolean Expressions Of Each Output of E0,E1,E2,E3

$E3 = B3 + B2 (B1 + B0)$

B1B0 \ B3B2	00	01	11	10
00	0	0	X	1
01	0	1	X	1
11	0	1	X	X
10	0	1	X	X

$E2 = \overline{B2} (B1 + B0) + B2\overline{B1}\overline{B0}$

B1B0 \ B3B2	00	01	11	10
00	0	1	X	0
01	1	0	X	1
11	1	0	X	X
10	1	0	X	X

$E1 = B1B0 + \overline{B1}\overline{B0}$

B1B0 \ B3B2	00	01	11	10
00	1	1	X	1
01	0	0	X	0
11	1	1	X	X
10	0	0	X	X

$E0 = \overline{B0}$

B1B0 \ B3B2	00	01	11	10
00	1	1	X	1
01	0	0	X	0
11	0	0	X	X
10	1	1	X	X

iv. Logic Diagram:

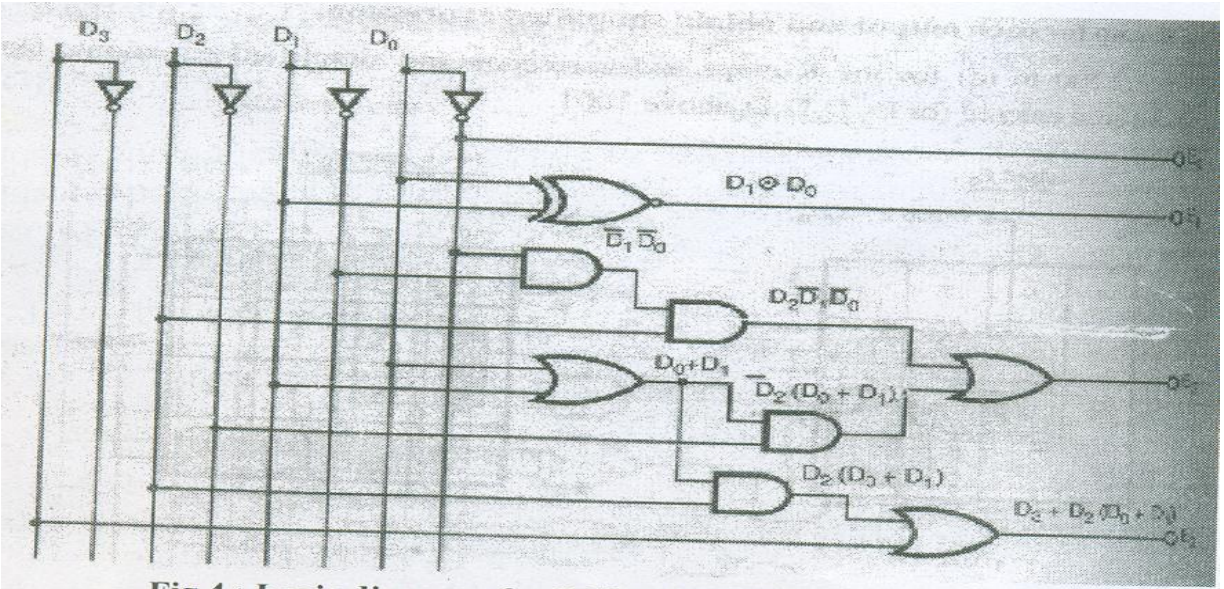

D_3
D_2
D_1
D_0
$D_1 \oplus D_0$
$\overline{D_1}\,\overline{D_0}$
$D_2\overline{D_1}\,\overline{D_0}$
D_0+D_1
$\overline{D_2}(D_0+D_1)$
$D_2(D_0+D_1)$
$D_3+D_2(D_0+D_1)$

Fig.4 : Logic diagram for BCD to excess 3 converter

Hardware Requirements Table:

GATE	**Quantity**	**IC**	**Quantity**
XOR	1	7486	1
NOT	4	7404	1
AND	4	7408	1
OR	3	7432	1

Test the circuit for all possible combinations of input and output codes.

4. Conclusion:

Thus, we studied different codes and their conversions including applications. The truth tables have been verified using IC 7486, 7432, 7408, and 7404.

FAQ's with answers:

Q.1) What is the need of code converters?

There is a wide variety of binary codes used in digital systems. Often it is required to convert from one code to another. For example the input to a digital system may be in natural BCD and output may be 7-segment LEDs. The digital system used may be capable of processing the data in straight binary format. Therefore, the data has to be converted from one type of code to another type for different purpose.

Q.2) What is Gray code?

It is a modified binary code in which a decimal number is represented in binary form in such a way that each Gray- Code number differs from the preceding and the succeeding number by a single bit.

(e.g. for decimal number 5 the equivalent Gray code is 0111 and for 6 it is 0101. These two codes differ by only one bit position i. e. third from the left.) It is non weighted code.

Q.3) What is the significance of Gray code?

Important feature of Gray code is it exhibits only a single bit change from one code word to the next in sequence. Whereas by using binary code there is a possibility of change of all bits if we move from one number to other in sequence (e.g. binary code for 7 is 0111 and for 8 it is 1000). Therefore it is more useful to use Gray code in some applications than binary code.

Q.4) What are applications of Gray code?

1. Important feature of Gray code is it exhibits only a single bit change from one code word to the next in sequence. This property is important in many applications such as Shaft encoders where error susceptibility increases with number of bit changes between adjacent numbers in sequence.

2. It is sometimes convenient to use the Gray code to represent the digital data converted from the analog data (Outputs of ADC).

3. Gray codes are used in angle-measuring devices in preference to straight forward binary encoding.

4. Gray codes are widely used in K-map

Q.5) What are weighted codes and non-weighted codes?

In weighted codes each digit position of number represents a specific weight. The codes 8421, 2421, and 5211 are weighted codes. Non weighted codes are not assigned with any weight to each digit position i.e. each digit position within the number is not assigned a fixed value. Gray code, Excess-3 code are non-weighted code.

Q.6) Why is Excess-3 code called as self-complementing code?

Excess-3 code is called self-complementing code because 9's complement of a coded number can be obtained by just complementing each bit.

Q.7) What is invalid BCD?

With four bits, sixteen numbers (0000 to 1111) can be represented, but in BCD code only 10 of these are used as decimal numbers have only 10 digits fro 0 to 9. The six code combinations (1010 to 1111) are not used and are invalid.

Assignment No: 3

Title: Adder

Objective: To learn and understand design and construction of combination circuit BCD adder.

Problem Statement: Design and Realization of BCD Adder using 4-bit Binary Adder (IC 7483).

Hardware Requirement:

i) IC 7483(4 bit Binary adder),IC 7404(Not-gate), 7432 (OR-gate), 7408 (AND-gate)

ii) Digital Trainer Kit -1

iii) Patch cords

Software Requirement: Digital Work 3.0

Theory:

Logic circuits for digital systems may be combinational or sequential. A combinational circuit consists of input variables, logic gates, and output variables. Example of combinational circuit are Code Convertor, BCD Adder, Subtractor, Multiplexer,Demultiplxer,Decoder etc

BCD-Binary coded decimal:-

- In this code each decimal digit represent by a 4-bit binary no. BCD is a way to express each of the decimal digits with binary code. Positional weights associated to the binary bits in BCD code are (MSB) 8-4-2-1(LSB).
- BCD number sae 0 to 9, but grater than 9 is invalid BCD.
- So Convert in valid BCD using add 6 i.e. (0110).

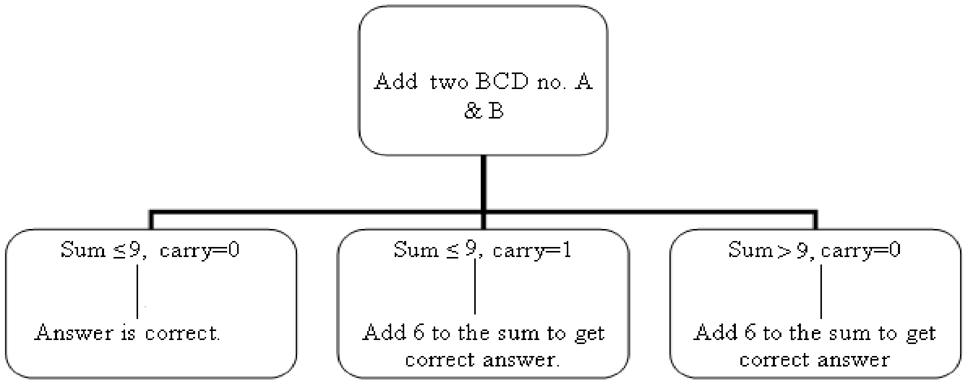
BCD Adder:

BCD adder is a circuit that adds two BCD digits & produces a sum of digits also in BCD.

Rules for BCD addition:

1. Add two numbers using rules of Binary addition.

2. If the 4 bit sum is greater than 9 or if carry is generated then the sum is invalid. To correct the sum add 0110 i. e. (6)10 to sum. If carry is generated from this addition add it to next higher order BCD digit.
3. If the 4 bit sum is less than 9 or equal to 9 then sum is in proper form.



CASE I : Sum <= 9 & carry = 0.

Add BCD digits 3 & 4

```
1.      0 0 1 1
       +0 1 0 0
         ------------
        0 1 1 1
```

Answer is valid BCD number = **(7)BCD**& so 0110 is not added.

CASE II : Sum > 9 & carry = 0.

Add BCD digits 6 & 5

```
1.      0 1 1 0
       +0 1 0 1
        -----------
        10 1 1
```

Invalid BCD (since sum > 9) so 0110 is to be added

```
2.      1 0 1 1
       +0 1 1 0
        -----------
1       0 0 0 1

        |        |
        ________
        (1       1)BCD
```

Valid BCD result = **(11) BCD**

CASE III : Sum < = 9 & carry = 1.

Add BCD digits 9 & 9

```
1.      1 0 0 1

       +1 0 0 1

        -----------

       1 0 0 1 0
```

Invalid BCD (since Carry = 1) so 0110 is to be added

```
2.      1  0 0 1 0
        +  0 1 1 0

        -----------

        1 1 0 0 0

        (1      8)BCD
```

Valid BCD result = **(18) BCD**

Design of BCD adder :

1. To execute first step i. e. binary addition of two 4 bit numbers we will use IC 7483 (withCin = 0), which is 4 bit binary adder.
2. We need to design a digital circuit which will sense sum & carry of IC 7483 & if sum exceeds 9 or carry = 1, this digital circuit will produce high output otherwise its output will be zero.

Circuit to check invalid BCD :

First we will design circuit to check sum & then we will logically OR output of this circuit to carry output of IC 7483

For digital circuit which we are going to design we will have 4 inputs (S3, S2, S1, S0) & only 1 output Y.

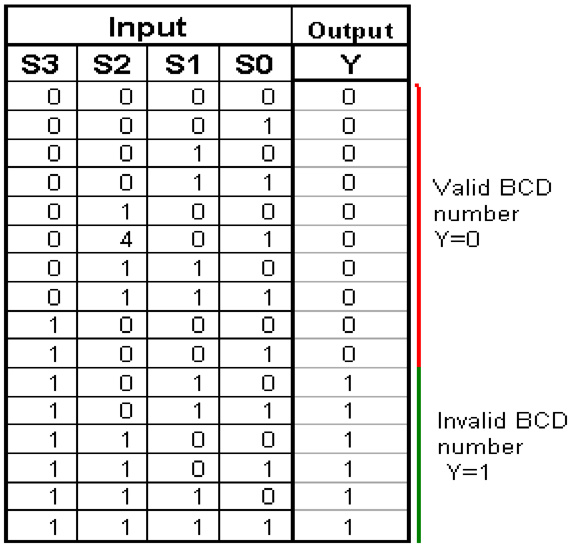
a) Y output of this circuit. Will be ORed with carry output of first adder IC 7483.

b) If BCD result is invalid i. e. sum output of first 7483 we have to add (6)10 i.e. (0110)2 that means we need one more binary adder IC 7483.

c) If BCD result is valid (i.e. final output of the circuit to check validity is 0) we will make an arrangement that second adder IC 7483 adds (0)10 i. e. (0000)2 to the sum of the first adder IC 7483.

The output of the combinational circuit is used as final output carry & carry output of second adder IC is ignored.

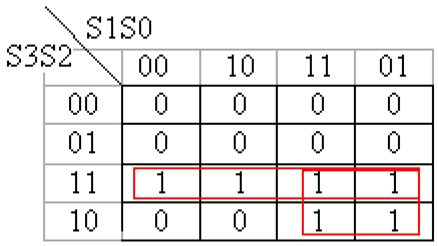
i) Truth Table for design of combinational circuit for BCD adder to check invalid BCD :



Input				Output
S3	S2	S1	S0	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	4	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Valid BCD number Y=0
Invalid BCD number Y=1

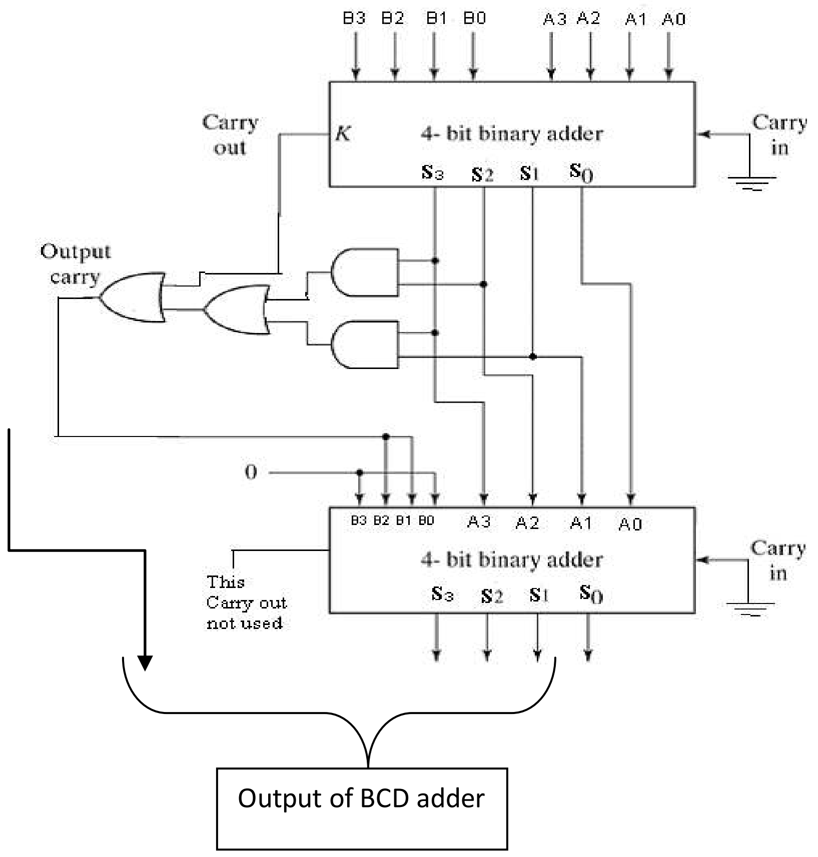
K-map for output:-



S3S2 \ S1S0	00	10	11	01
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	0	0	1	1

Y=S3S2+S3S1

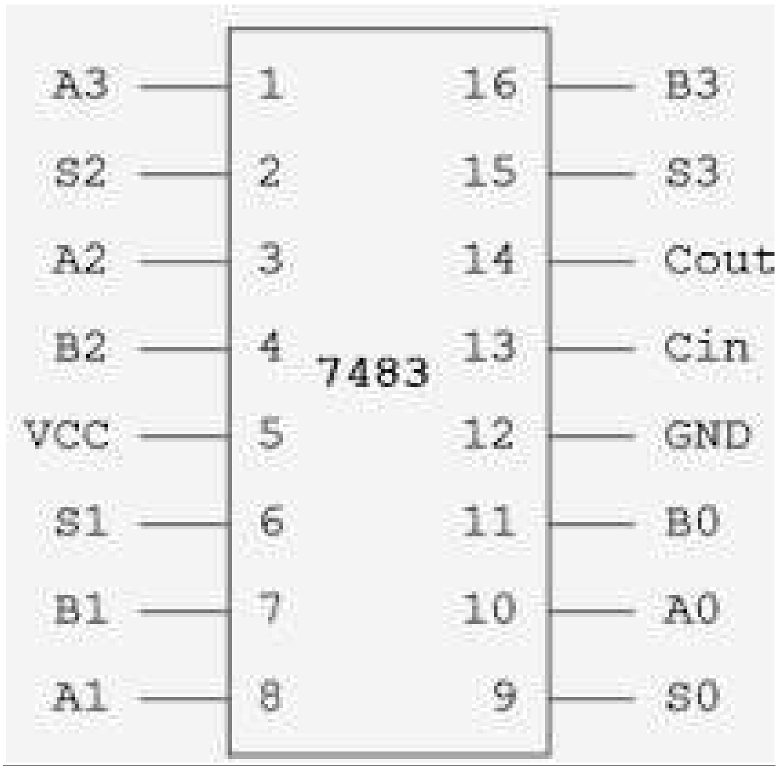
iii) Circuit diagram:



i) Hardware Requirements :

GATE	Quantity	IC	Quantity
Binary adder	2	7483	2
AND	2	7408	1
OR	2	7432	1

<u>IC 7483: 4 bit Binary Adder</u>



Test the circuit for all possible combinations of input and output codes.

Conclusion:

BCD adder is designed & tested for all possible combinations**.**

FAQ's:

1. Explain and Write the significance of. BCD number system

2. Write the applications of BCD & Excess 3 code.

3. What is the difference between BCD and binary codes?

4. What do you mean by unpacked and packed BCD nos.?

Assignment No: 4

Title: Multiplexer and Demultiplexer.

Problem Statement: Realization of Boolean Expression for suitable combination logic using MUX 74151/74153, DMUX 74154/74138.

Objective: To learn and understand design and construction of combination circuit Multiplexer and Demultiplexer.

Outcomes: On completion of this practical, learner will be able to

CO1: Understand the working of Multiplexer and Demultiplexer. .

CO2: Apply the knowledge to multiplexer and decoder IC as per the design specifications.

CO3: Design and implement Multiplexer and Demultiplexer Combinational digital circuits as per the specifications.

Hardware Requirement:

i) IC 74151.(8:1 Mux),IC 74138(3:8 decode),IC 7404(Not-gate), 7432 (OR-gate), 7408 (AND-gate)

ii) Digital Trainer Kit -1

iii) Patch cords

Software Requirement: Digital Work 3.0

Theory:

Logic circuits for digital systems may be combinational or sequential. A combinational circuit consists of input variables, logic gates, and output variables. Example of combinational circuit is Code Convertor, BCD Adder, Subtractor, **Multiplexer**, Demultiplexer, Decoder etc

Introduction of Multiplexer:

- Multiplexer is a digital switch which allows digital information from several sources to be routed onto a single output line. Basic multiplexer has several data inputs and a single output line.

- The selection of a particular input line is controlled by a set of selection line.
- There are 2n input lines & n is the number of selection line whose bit combinations determines which input is selected .It is “Many into One”.
- Multiplxer is special type of combination circuit. A multiplexer performs the function of selecting the input on any one of 'n' input lines and feeding this input to one output line. The selection of a particular input line is controlled by a set of selection line. To select ‘n’ inputs we need ‘m’ select line such that **2^m = n.**

E is called as enable input which is generally active low.

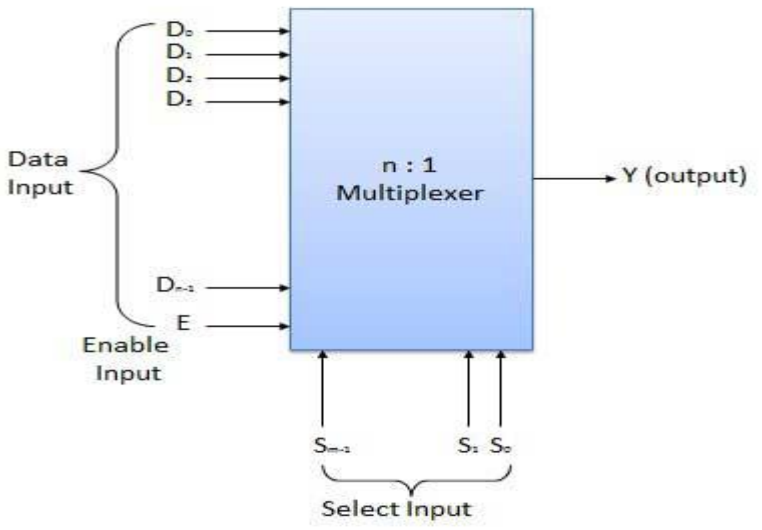
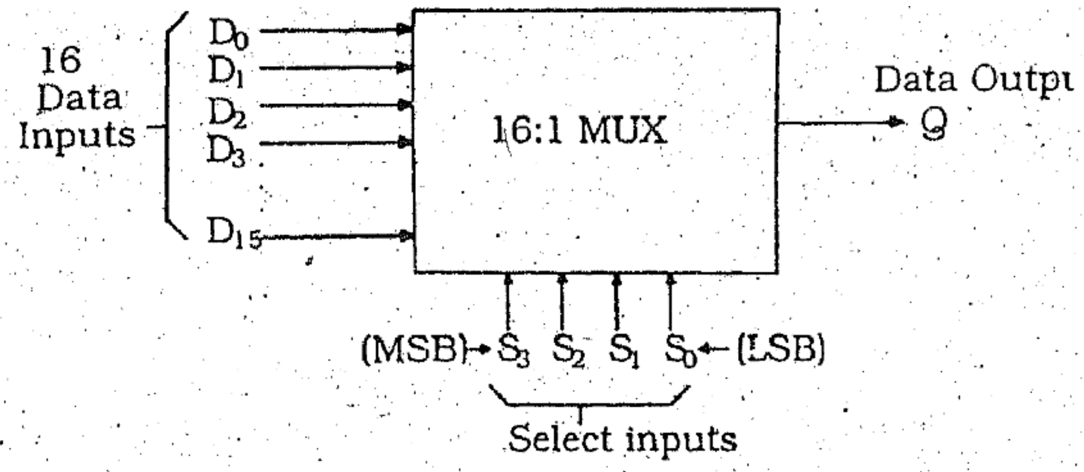


Fig:.-Block diagram for N:1 Multiplxer



Block diagram of 16:1 Multiplexer

Necessity of multiplxer:-

- In most of the electronic system, the digital data is available on more than one line.

It is necessary to route this data over a single line.

- Multiplexers are used as one method of reducing the number of integrated circuit packages required by a particular circuit design. This in turn reduces the cost of the system.

Application of multiplxer :-

- No need to simplify logic expression.
- The IC package count is minimized.
- Logic design is simplified.
- To minimize number of connections in communication system were we need to handle thousands of connections.
  Ex. Telephone exchange

- In designing the combinational circuit

Design Boolean expression using Multiplexer IC 74151

Implement Following function using Multiplxer.

$$F(A,B,C) = \sum m\ (1, 2, 5,7)$$

Solution:-Since there are 3 varible,the multiplxer have 3 select I/P should be used.

Hence one 8:1 mux should be used.

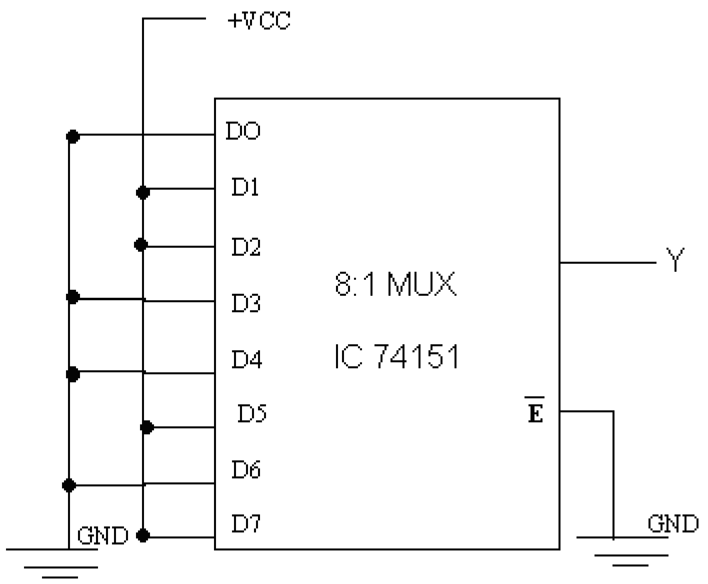
Setp 1:-Identify the number decimal corresponding to each minterm.

Here 1,2,5,7

Step 2:-Connect the data input lines 1,2,5,7 o logic 1(+Vcc) & remaining input line

0,3,4,6 to logic 0(GND)

Step 3:-Connect variables A,B & C to select input.



Truth table:-

Select line			Enable	Output
C	B	A	E	Y
X	X	X	0	0
0	0	0	1	D0
0	0	1	1	D1
0	1	0	1	D2
0	1	1	1	D3
1	0	0	1	D4
1	0	1	1	D5
1	1	0	1	D6
1	1	1	1	D7

Introduction of DeMultiplexer:

- A de-multiplexer is a circuit that receives information on a single line and transmits information on one of the possible output lines. The selection of specific output line is controlled by the values of n-select lines.
- Demultiplexer is a logic used to perform exactly reverse function performed by multiplexer.
- It accepts a single input and distributes among several outputs.
- The selection of a particular output line is controlled by a set of selection line.
- The single input variable Din has a path to all output, but input information is directed to only one of the output lines

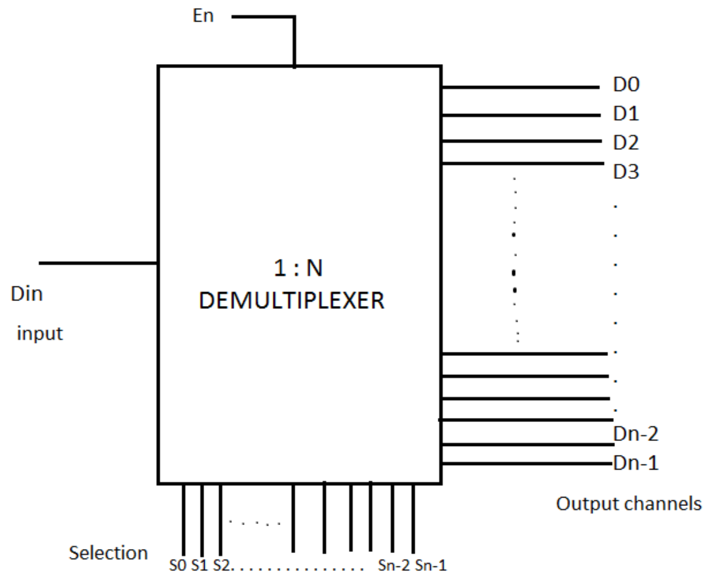


Figure:-Block diagram of 1:N Demultiplxer

Differentiate between Mux / Demux / Decoder.

Point	Multiplexer	Demultiplexer	Decoder
Input	Many input lines	Single input line	Many input line also Acts as select line
Output	Single output line	Many output line	Many output line, Active low output
Select line	2m = n	n = 2m	Enable inputs used

Design Boolean expression using DeMultiplexer/Decoder IC 74138

The procedure for implementation of combinational circuit by means of a decoder and ‘OR’ gates requires that the Boolean function fir the circuit be expressed in Sum of Minterms. These forms can be obtained by expanding the function. A decoder is then chosen which generates all the minterms of n i/p variables. The i/p to each OR gate are selected from the decoder outputs according to the minterms list in each function.

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

Sum S=∑m(1,2,4,7);

Carry S=∑m(3,5,6,7);

Logic diagram:-

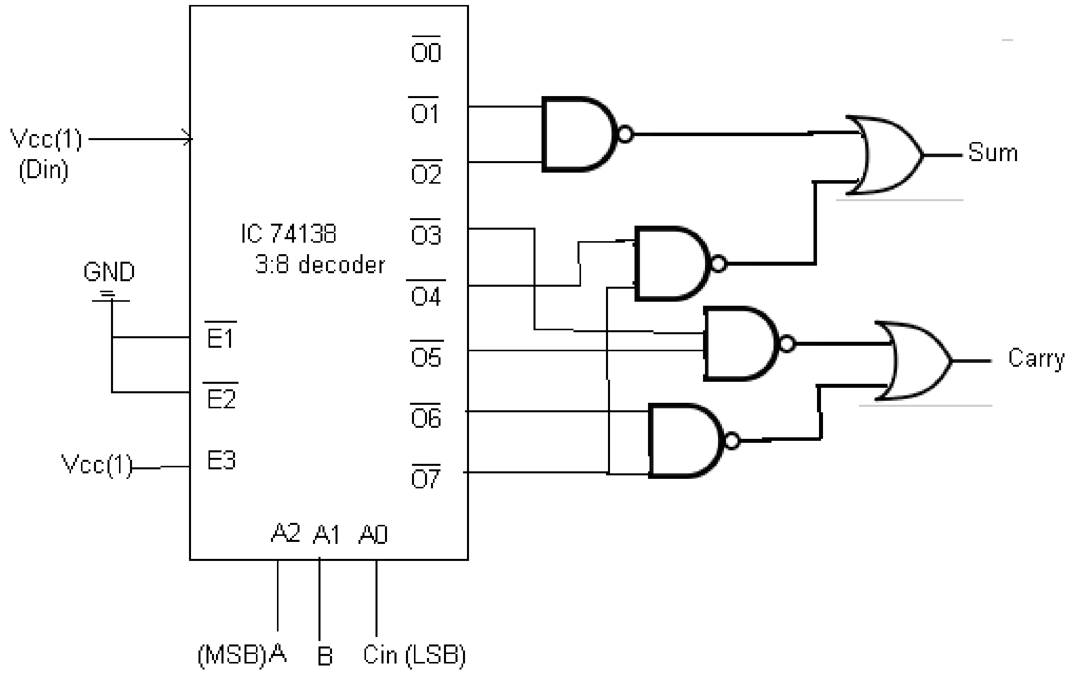
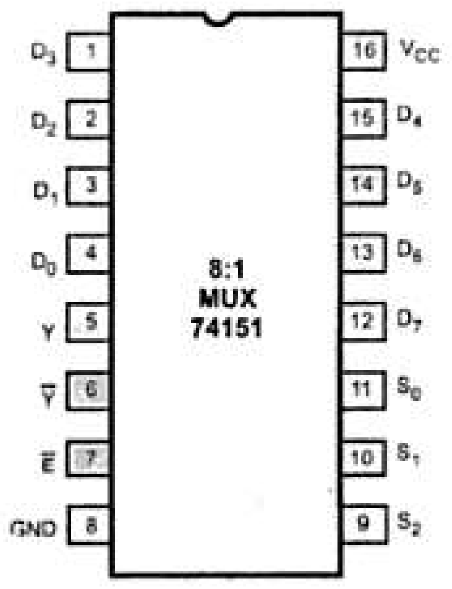


Figure :- Logic diagram for Full adder using 1:8 demux or 3:8 decoder.

Pin Diagrams:

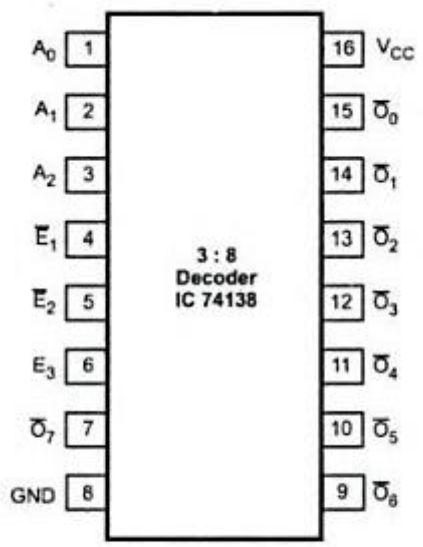
IC 74151 multiplxer 8:1:-



$\overline{Y}$: Complemented output

$\overline{E}$: Active low enable input

- 1.5.1 IC 74138 De multiplxer 1:8 or Decoder 3:8:-
- IC 74138 De-multiplexer:
- This is 1:8 de-multiplexers IC.Also it is 3:8 decoder. It has 3 enable pins out of which 1 is active low enable pine can be used as data line.



Pin description:

The IC 74138 accepts three binary inputs (A_0, A_1, A_2) and when enabled provides eight individual active low outputs (O_0 - O_7). The device has three enable inputs : two active low ($\overline{E}_1$, $\overline{E}_2$) and one active high (E_3).

Conclusion:

In this way multiplexer, Decoder& its applications are studied, implemented & Tested for all possible combinations

FAQ's

1. Difference between multiplxer & demultiplxer?
2. Different type of multiplxer?
3. List of different IC for multiplxer?
4. What you mean by Encoder?
5. What is releation between select line & input line?
6. What is the releation between select lines and output lines?
7. How is IC 74138 activated?
8. How to convert 2:4 decoder to 3:8 decoder?
9. Implement a 3-bit binary to gray code converter using decoder IC 74138?
10. Application of Decoder?

Assignment No: 5

Title: Comparator

Problem Statement: To Verify the truth table of two bit comparators using logic gates.

Objective: To learn and understand design and construction of combinational circuit of Comparator.

Outcomes: On completion of this practical, learner will be able to

CO1: Understand the working of Comparator .

CO2: Apply the knowledge to gate IC as per the design specifications.

CO3: Design and implement Comparator Combinational digital circuits as per the specifications.

Hardware Requirement:

i) IC 7404(Not-gate), 7432 (OR-gate), 7408 (AND-gate), 7486 (Ex-or gate)

ii) Digital Trainer Kit -1

iii) Patch cords

Software Requirement: Digital Work 3.0

Theory:

Logic circuits for digital systems may be combinational or sequential. A combinational circuit consists of input variables, logic gates, and output variables. Example of combinational circuit are Adder, Subtractor, Multiplexer,Demultiplxer,Decoder,**Comparator** etc

1. **Introduction about Comparator?**

   A magnitude digital Comparator is a combinational circuit that **compares two digital or binary numbers** in order to find out whether one binary number is equal, less than or greater than the other binary number i.e 1-bit,2-bit,4 bit comparator we can use.

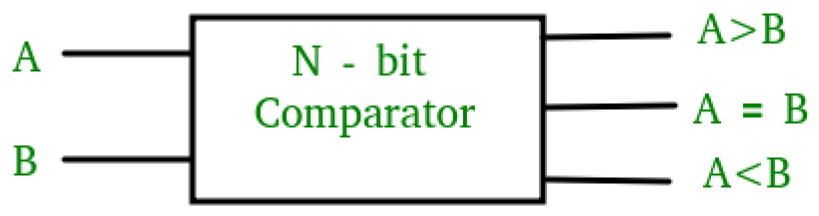
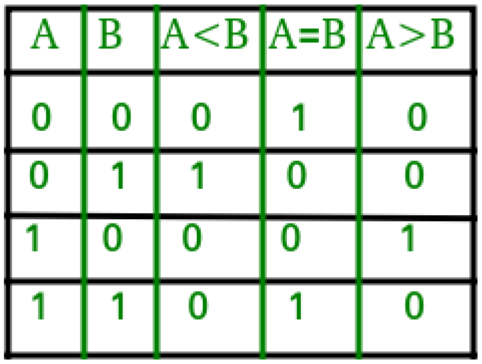


Figure: Block diagram of Comparator

1-Bit Magnitude Comparator –

- A comparator used to compare two bits is called a single bit comparator. It consists of two inputs each for two single bit numbers and three outputs to generate less than, equal to and greater than between two binary numbers.
- The truth table for a 1-bit comparator is given below:

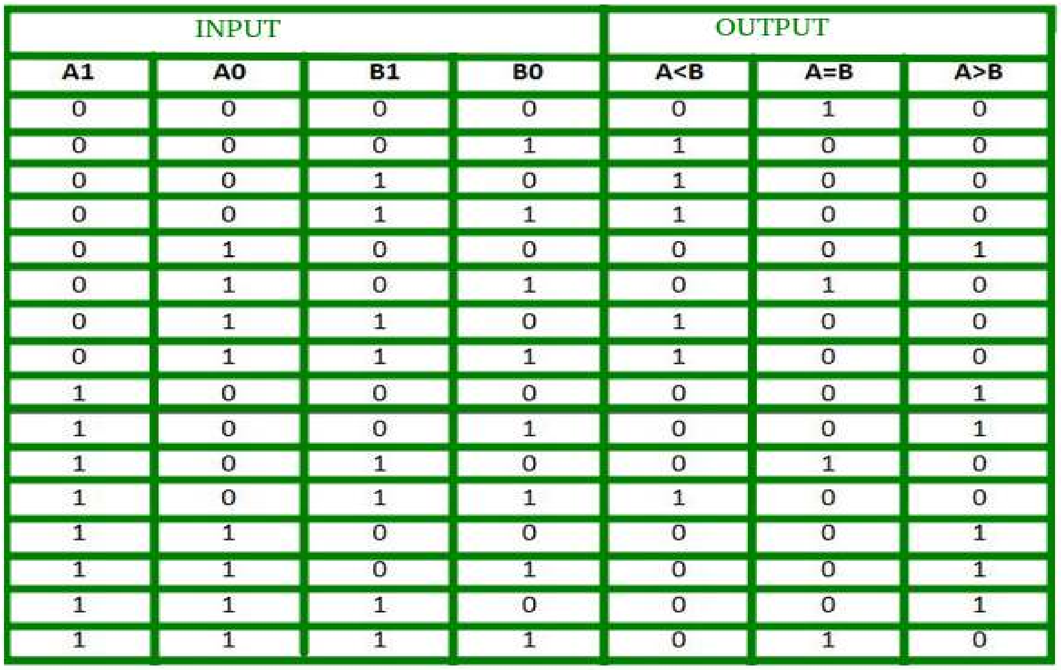


A	B	A<B	A=B	A>B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

2. **Design 2-bit Comparator using gates.**

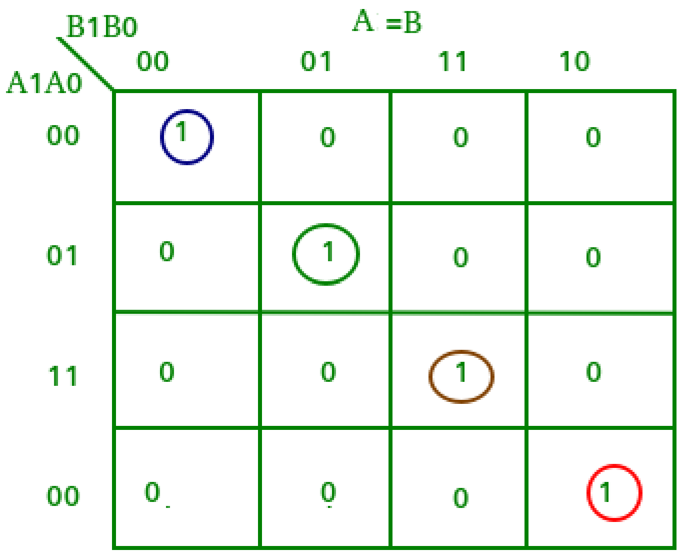
2-Bit Comparator

- A comparator used to compare two binary numbers each of two bits is called a 2-bit Magnitude comparator. It consists of four inputs and three outputs to generate less than, equal to and greater than between two binary numbers.
- ***The truth table*** for a 2-bit comparator is given below:



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

- *K-map for A=B :*

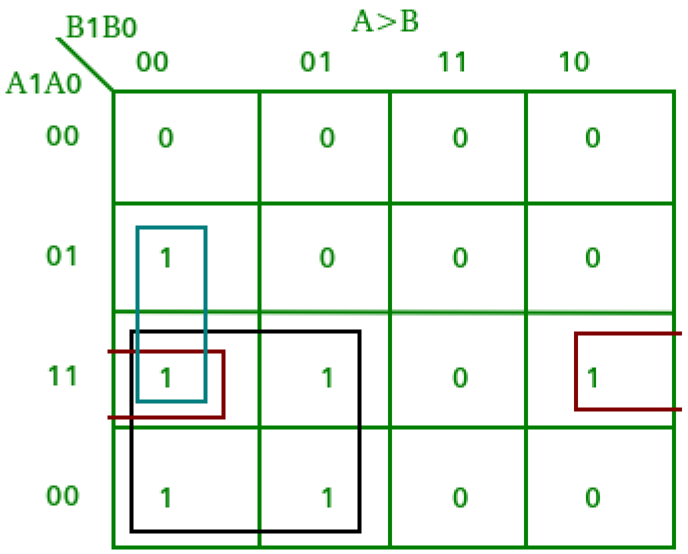


A =B

B1B0 / A1A0	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
00	0	0	0	1

A=B: A1’A0’B1’B0’ + A1’A0B1’B0 + A1A0B1B0 + A1A0’B1B0’
: A1’B1’ (A0’B0’ + A0B0) + A1B1 (A0B0 + A0’B0’)
: (A0B0 + A0’B0’) (A1B1 + A1’B1’)
: (A0 Ex-Nor B0) (A1 Ex-Nor B1)

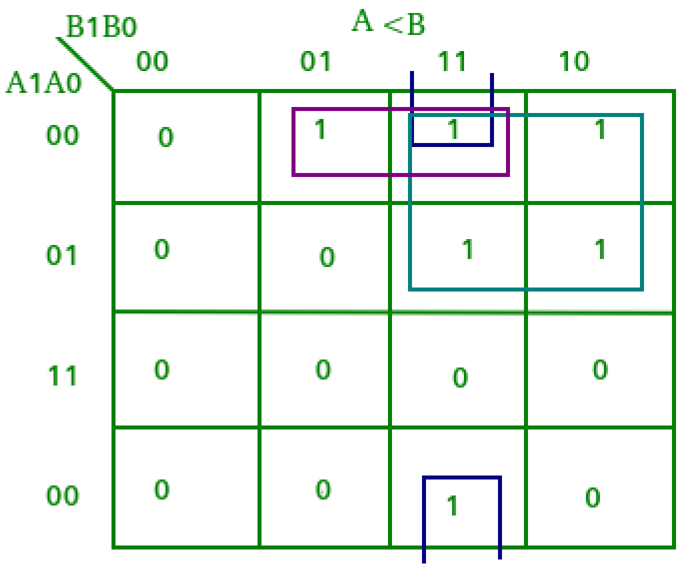
- *K-map for A>B :*



A1A0 \ B1B0	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	1	1	0	1
00	1	1	0	0

A>B = A1B1’ + A0B1’B0’ + A1A0B0’

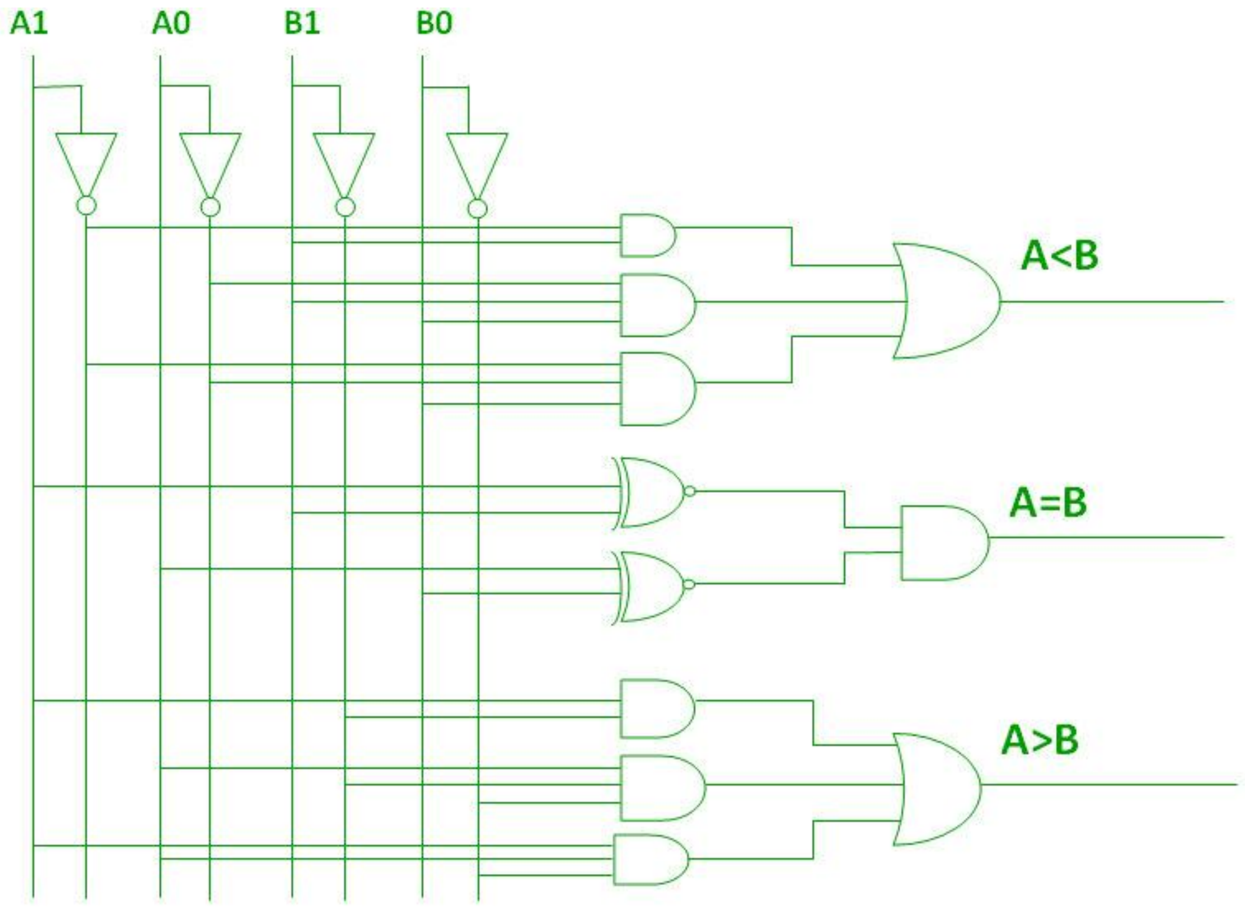
- ***K-map for A<B :***



A1A0 \ B1B0	00	01	11	10
00	0	1	1	1
01	0	0	1	1
11	0	0	0	0
00	0	0	1	0

A<B:A1’B1 + A0’B1B0 + A1’A0’B0

Logic diagram:



Conclusion: Comparator is successfully implemented, the comparator is studied & output are checked. The truth table is verified.

Assignment No: 6

Title: Parity Generator.

Objective: To learn and understand design and construction of combination circuit Parity Generator.

Problem Statement: Design and Implement Parity Generator and checker using EX-OR.

Hardware Requirement:

i) IC 7486(Ex-OR gate), IC 7404(Not-gate), 7432 (OR-gate), 7408 (AND-gate)

ii) Digital Trainer Kit -1

iii) Patch cords

Software Requirement: Digital Work 3.0

Theory:

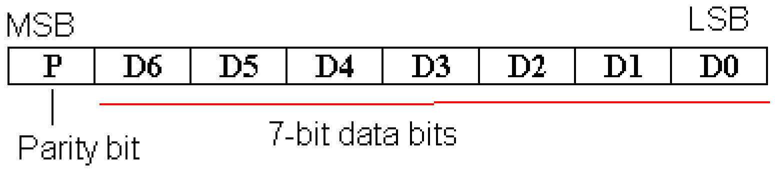
Parity: A term used to specify the number of one's in a digital word as odd or even. There are two types of Parity - even and odd. An even parity generator will produce a logic 1 at its output if the data word contains an odd number of ones. If the data word contains an even number of ones then the output of the parity generator will be low. By concatenating the Parity bit to the data word, a word will be formed which always has an even number of ones i.e. has even parity.

Parity bit: An extra bit attached to a binary word to make the parity of resultant word even or odd. Parity bits are extra signals which are added to a data word to enable error checking.

Parity generator: A logic circuit that generates an additional bit which when appended to a digital word makes its parity as desired (odd or even).Parity generators calculate the parity of data packets and add a parity amount to them. Parity is used on communication links (e.g. Modem lines) and is often included in memory systems. If a data word is sent out with even parity, but has odd parity when it is received then the data has been corrupted and must be resent. As its name implies the operation of an Odd Parity generator is similar but it provides odd parity. The table shows the parity generator outputs for various 8-bit data words.

It is simplest technique for detecting error is to add extra bit known as parity bit to each word being transmitted.

Parity bit is attach to the information to enable error checking.The parity bit can be set to 0 or 1 depending on the type of parity required.



Two types of Parity - even and odd.

1) Even parity generator will produce a logic 1 at its output if the data word contains an odd number of ones.(i.e. number of ones is 0,2,4,6,…)

2) Odd parity generator will produce a logic 1 at its output if the data word contains an even number of ones. (i.e. number of ones is 1,3,5,7,…)

If a dataword is sent out with even parity, but has odd parity when it is received then the data has been corrupted and must be resent.

Parity checker: A logic circuit that checks the parity of binary word. Parity checkers are integrated circuits (ICs) used in digital systems to detect errors when streams of bits are sent from a transmitter to a receiver. The minimum distance between any two code words with parity bit attached is two. The parity bit 1 or 0 is attached to the code to be transmitted at the transmitter end and the parity of the received (n+1)-bit word is checked at the receiving end. If there is only one error, the erroneously code is detected at the receiving end by parity check.

Need of parity bit:-

- Parity is used on communication links (e.g. Modem lines) and is often included in memory systems.
- Errors can occur as digital codes are transferrd from one source to another source within the digital system.
- Errors occur as the bits may change, like the 1 bit in the data change to 0 or 0 bit in the data change to 1,this results in error which can be due to various reasons such as noise or component malfunction.

Truth table

For Odd parity checker :-

0– Error

1– No Error

For Even parity checker :-

0– Error

1– No Error

This table shows the parity generator outputs for various 4-bit data words.

A B C D	Even	Odd
0 0 0 0	0	1
0 0 0 1	1	0
0 0 1 0	1	0
0 0 1 1	0	1
0 1 0 0	1	0
0 1 0 1	0	1
0 1 1 0	0	1
0 1 1 1	1	0
1 0 0 0	1	0
1 0 0 1	0	1
1 0 1 0	0	1
1 0 1 1	1	0
1 1 0 0	0	1
1 1 0 1	1	0

1 1 1 0	1	0
1 1 1 1	0	1

K- Map for Odd:-

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

Odd = $\overline{A \oplus B \oplus C \oplus D}$

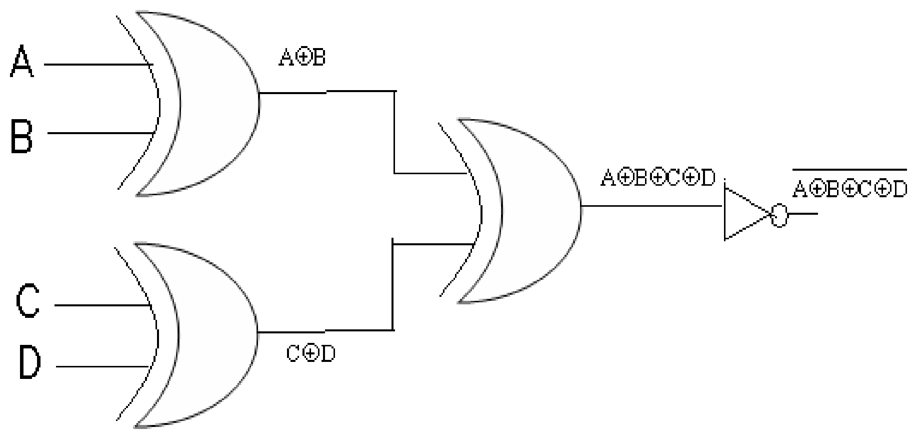


Figure:-Logic diagram for Odd parity generator.

K- Map for Even :-

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

Even = A□B□C□D

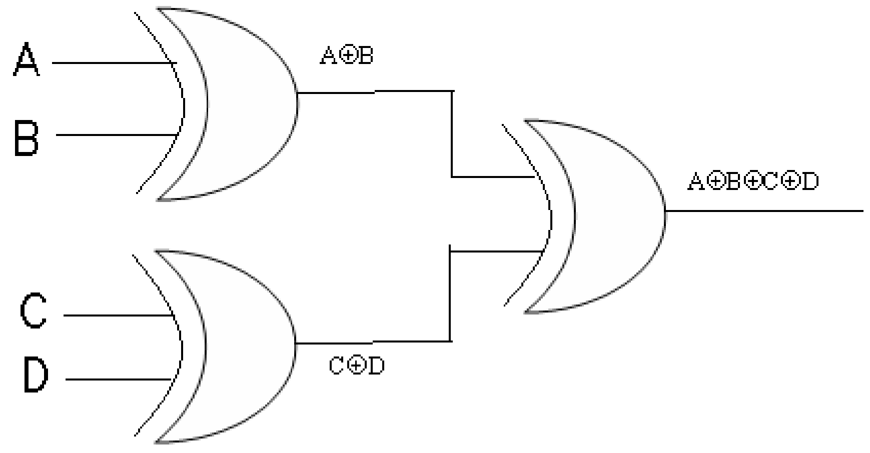


Figure:-Logic diagram for Even parity generator.

Conclusion:

In this way parity generator is studied, implemented & Tested for all possible combinations

FAQ's

Q.1 What is difference between error detecting & error correcting?.

Q.2 Different type of error detecting methods?

Q.3 What is error?

Q.4 Limitation of parity generator?

Advantages of Parity

The advantages of parity are

- Simplicity
- Easy to use

Applications of Parity

The applications of parity are

- In digital systems and many hardware applications, this parity is used

- The parity bit is also used in Small Computer System Interface (SCSI) and also in Peripheral Component Interconnect (PCI) to detect the errors

FAQs

1). What is the difference between the parity generator and parity checker?

The parity generator generates the parity bit in the transmitter and the parity checker checks the parity bit in the receiver.

2). What does no parity mean?

When the parity bits are not used to check for errors then the parity bit is said to be non-parity or no parity or the absence of parity.

3). What is the parity value?

The parity value concept used for both commodities and securities and the term refers to when the value of the two assets is equal.

4). Why do we need a parity checker?

The parity checker is needed to detect the errors in communication and also in the memory storage devices parity checker is used for testing.

5). How can the parity bit detect a damaged data unit?

The redundant bit in this technique is called a parity bit, it detects damaged data unit when an error occurs during the transmission of data.

In this article, how the parity generator and checker generate and check the bit and its types, logic circuits, truth tables, and k-map expressions are discussed briefly. Here is a question for you, how do you calculate even and odd parity?

Assignment No: 7

Title: Flip Flop

Problem Statement: Design and Realization: Flip Flop conversion

---

Objective: To learn and understand design and construction of sequential circuit.

Outcomes: On completion of these practical, students will be able to

CO1: Understand the working of Flip flop.

CO2: Apply the knowledge gate IC and flip flop as per the design specifications.

CO3: Design and implement flip flop conversion circuits.

Hardware Requirement:

i) IC 7404(Not-gate), 7432 (OR-gate), 7408 (AND-gate), 7486 (Ex-or gate)

ii) Filp flop:SR,JK,D,T

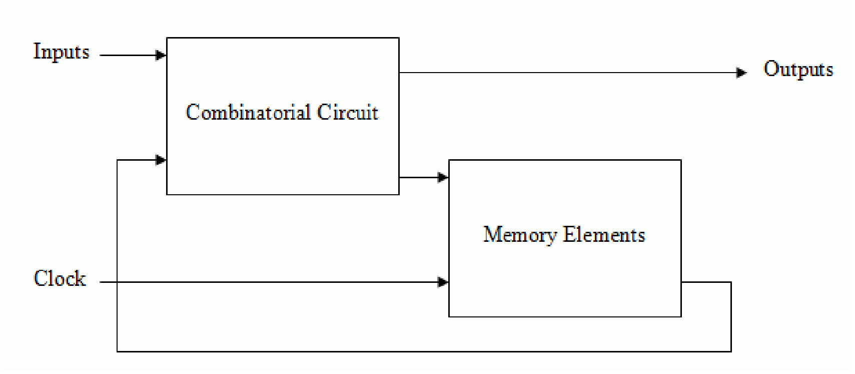
iii) Digital Trainer Kit -1

iv) Patch cords

Software Requirement: Digital Work 3.0,Android app:Logic Circuit SIM

Theory:

- A **sequential circuit** is one whose outputs depend not only on its current inputs, but also on the past sequence of inputs.
- In other words, sequential circuits must be able to "**remember**" (i.e., store) the past history of the inputs in order to produce the present output.
- The two most popular varieties of storage cells used to build sequential circuits are: latches and flip-flops.
- **Latch:** **level sensitive** storage element
- **Flip-Flop:** **edge triggered** storage element
- SR, D latches.
- D, JK, SR & T flip flops.



- **Two tables are there:-**

1.Truth table:- shows operation of circuits i.e. shows changes in output whenever changes in inputs. Mapping between input to output.

2.Excitation table:- Sometimes there is need to find input condition from the given output conditions, we need excitation table. In this case for the desired output, we need to find out input conditions

JK Flip flop truth table

J	K	Qn+1
0	0	Qn
0	1	0
1	0	1
1	1	Qn'

JK Flip flop excitation table

Qn	Qn+1	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

SR Flip flop truth table

SR Flip flop excitation table

S	R	Qn+1
0	0	Qn
0	1	0
1	0	1
1	1	?

Qn	Qn+1	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

D Flip flop truth table

D	Qn+1
0	0
1	1

D Flip flop excitation table

Qn	Qn+1	D
0	0	0
0	1	1
1	0	0
1	1	1

T Flip flop truth table

T	Qn+1
0	Qn
1	Qn’

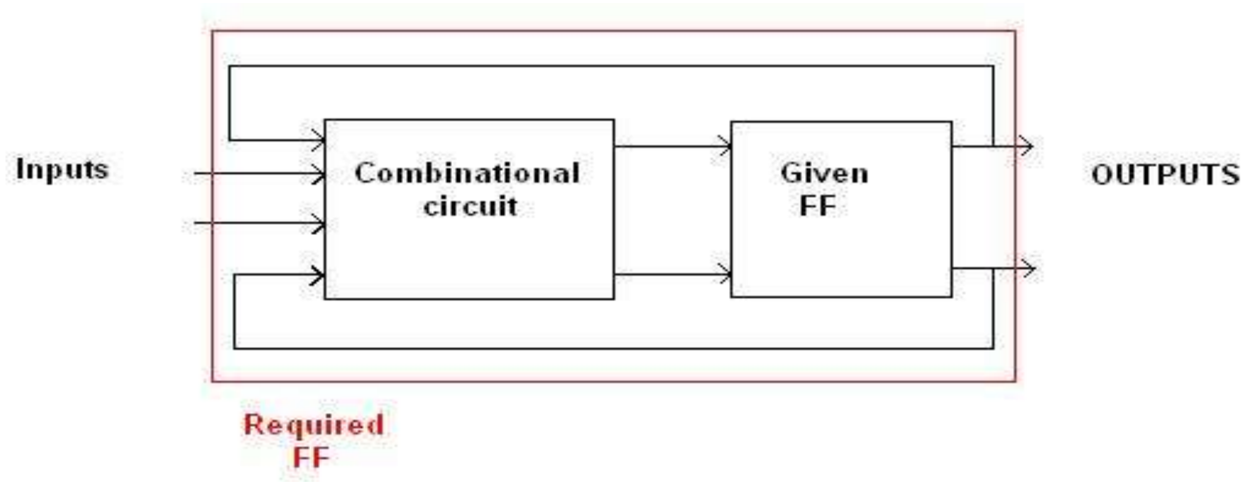
Qn	Qn+1	T
0	0	0

0	1	1
1	0	1
1	1	0

T Flip flop excitation table

Conversion of FF

- We can convert easily one type of flip flop into another flip flop by using some additional gates / combinational circuits.
- In this case, what we need is truth table of required FF and excitation table of given FF.



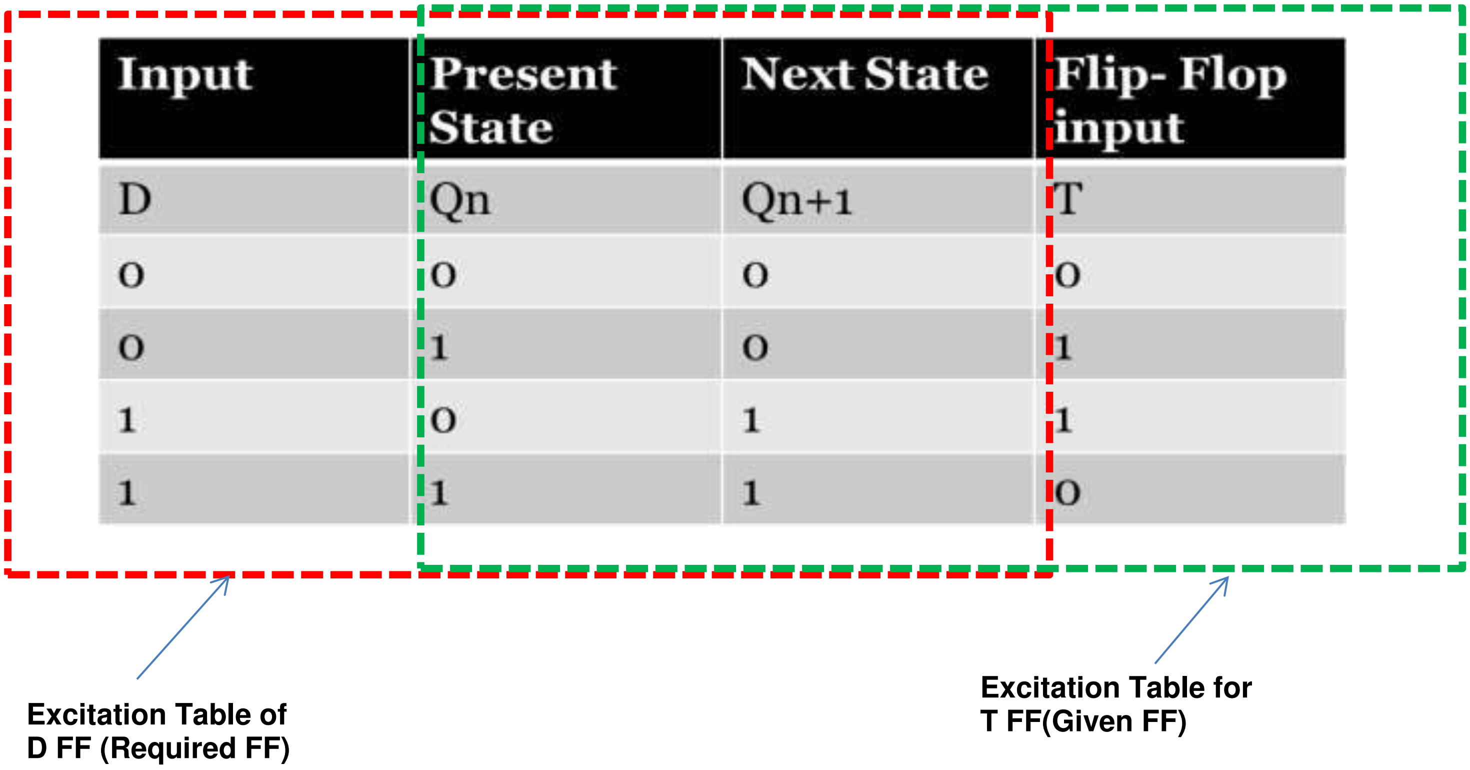
Example Conversion of Flip Flops

- **SR to JK**
- **SR to D**
- **SR to T**
- **JK to D**
- **JK to T**
- **D to T**
- **T to D**
- **JK to SR**
- **T to SR**
- **D to JK**

Steps for FF of Conversion

1. Prepare table(Excitation table of T and D FF)
2. Prepare k-map and simplify logic expression
3. Draw logic diagram of FF conversion.

1. T Flip-Flop to D Flip-Flop

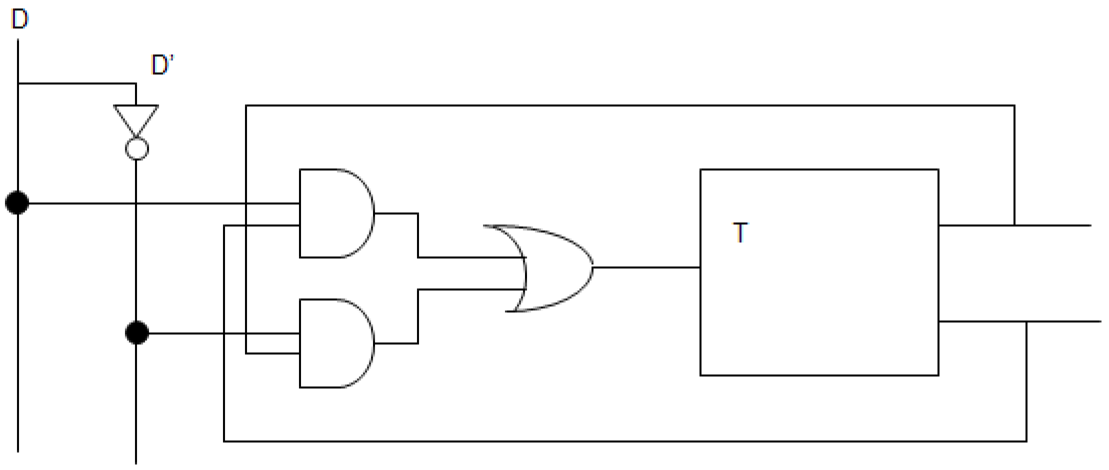


Input	Present State	Next State	Flip- Flop input
D	Qn	Qn+1	T
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Excitation Table of D FF (Required FF)
Excitation Table for T FF(Given FF)

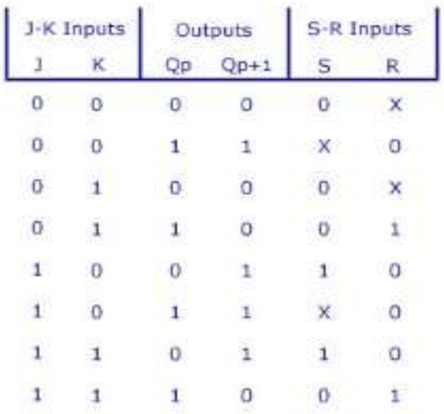
D \ Qn	0	1
0	0	1
1	1	0

T = D Q' + D' Q



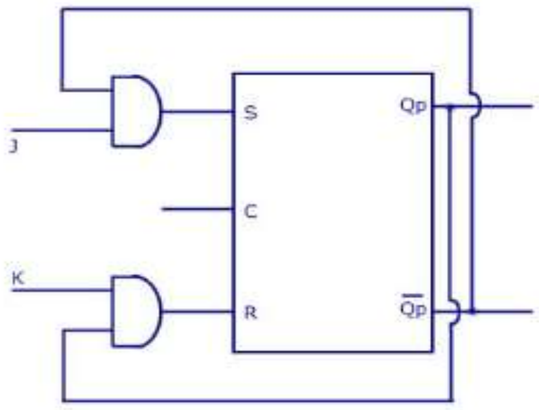
S-R Flip Flop to J-K Flip Flop

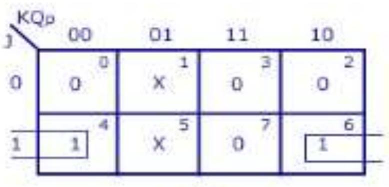
Conversion Table



J-K Inputs		Outputs		S-R Inputs	
J	K	Qp	Qp+1	S	R
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	X	0
1	1	0	1	1	0
1	1	1	0	0	1

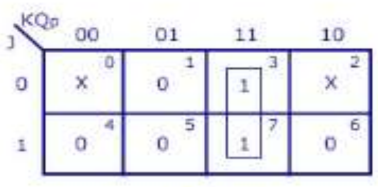
Logic Diagram





J \ KQp	00	01	11	10
0	0	X	0	0
1	1	X	0	1

$S = \overline{K}\,Qp$ wait



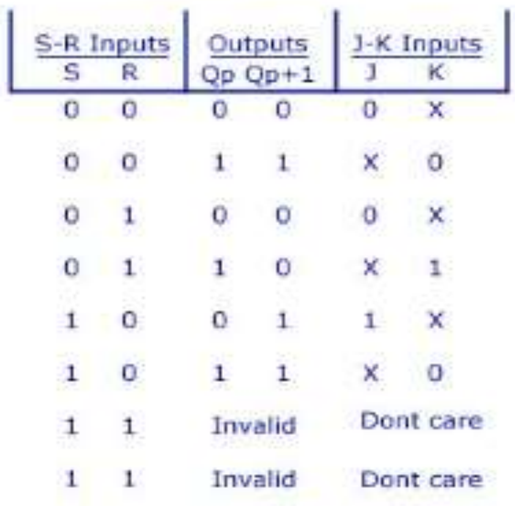
J \ KQp	00	01	11	10
0	X	0	1	X
1	0	0	1	0

R = KQp

K-Map

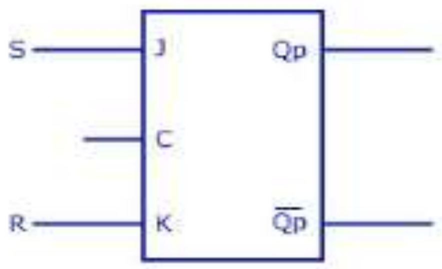
J-K Flip Flop to S-R Flip Flop

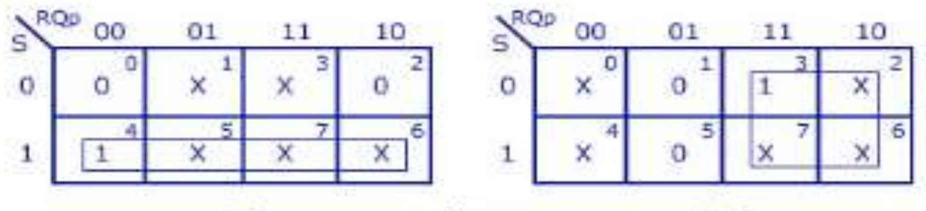
Conversion Table



S-R Inputs		Outputs		J-K Inputs	
S	R	Qp	Qp+1	J	K
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	0	X	1
1	0	0	1	1	X
1	0	1	1	X	0
1	1	Invalid		Dont care	
1	1	Invalid		Dont care	

Logic Diagram





S \ RQp	00	01	11	10
0	0	X	X	0
1	1	X	X	X

S \ RQp	00	01	11	10
0	X	0	1	X
1	X	0	X	X

J=S K-maps K=R

4. **Conclusion:** Flip flop conversion is studied & is successfully implemented, the Flip flop conversion output are checked. The truth table is verified.

Assignment No: 8

Title: Sequential Circuit

Problem Statement: Design of 2 bit and 3 bit Ripple Counter using MS JK flip-flop. ----

----

Objective: To learn and understand design and construction of sequential circuit.

Outcomes: On completion of these practical, students will be able to

CO1: Understand the working of ripple counter.

CO2: Apply the knowledge MS J K flip flop as per the design specifications.

CO3: Design and implement Ripple circuits.

Hardware Requirement:

v) IC 7404(Not-gate), 7432 (OR-gate), 7408 (AND-gate), 7486 (Ex-or gate)

vi) Filp flop: JK

vii) Digital Trainer Kit -1

viii) Patch cords

Software Requirement: Digital Work 3.0, Android app: Logic Circuit SIM

Theory:

Asynchronous counter / Ripple Counter:

A digital counter is a set of flip flop. When state changes in response to pulse applied at i/p to counter. The flip flop are connected such that their combined state at any time is binary equivalent of total no. of pulses that have occurred up to that time. Thus its name implies a counter is used to count pulse. A counter is used as frequency dividers. To obtain waveform with frequency that is specific fraction of clock frequency.

Counter may be Asynchronous or synchronous. The Asynchronous counter is also called as ripple counter .An Asynchronous counter uses T flip flop to perform a counting function. The actual hardware used is usually J-K flip flop connected to logic1.Even D flip flops may be used here.

In asynchronous counter commonly called ripple counter, the first flip-flop is clocked by the external clock pulse & then each successive flip-flop is clocked by the Q or /Q' output the previous flip-flop. Therefore in an asynchronous counter the flip-flop are not clocked simultaneously. The input of MS-JK is connected to VCC because when both inputs are one output is toggled. As MS-JK is negative edge triggered at each high to low transition the next flip-flop is triggered. On this basis the design is done for MOD-8 counter.

6.2 Up Counter:-

Fig1 shows 3 bit Asynchronous Up Counter. Here Flip-flop A act as a MSB Flip-flop and Flip-flop 0 can act as a LSB Flip-flop. Clock pulse is connected to the Clock of Flip-flop 0. Output of Flip-flop 0 (Q0) is connected to clock of next flip-flop (i.e Flip-flop 1) and so on. For 3 bit Up counter Truth table is as shown below.

Table 1: Truth table for 3-bit Up counter

Counter States	Q_2	Q_1	Q_0
0	**0**	**0**	**0**
1	**0**	**0**	**1**
2	**0**	**1**	**0**
3	**0**	**1**	**1**
4	**1**	**0**	**0**
5	**1**	**0**	**1**
6	**1**	**1**	**0**
7	**1**	**1**	**1**

6.2.1 Implementation of 3-bit up counter using IC 7476:-

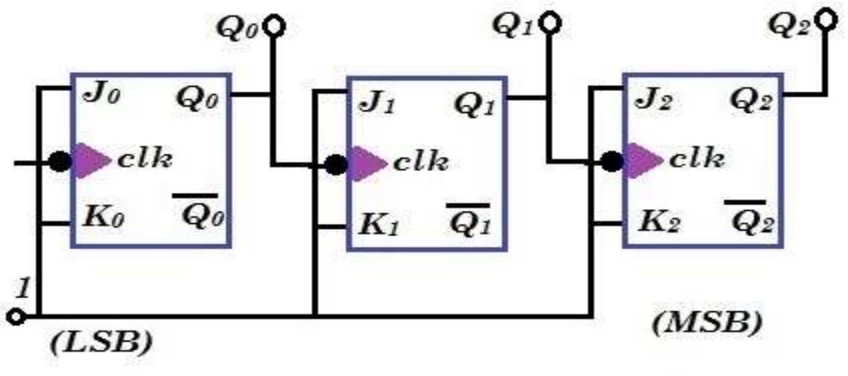


Table 2:Truth table for 2-bit Up counter

Counter States	Q_1	Q_0
0	0	0
1	0	1
2	1	0
3	1	1
4	0	0

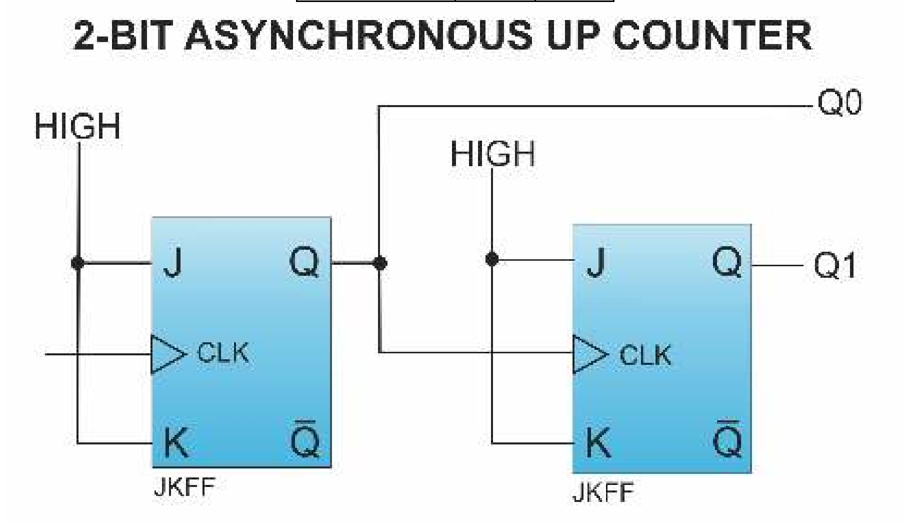


Fig 1: 4 Bit Asynchronous Up Counter

6.3 Down Counter:-

Fig 2 shows 3 bit Asynchronous Down Counter. Here Flip-flop A act as a MSB Flip-flop and Flip-flop 2 can act as a LSB Flip-flop. Clock pulse is connected to the Clock of Flip-flop 2. Output of Flip-flop D (Q2’) is connected to clock of next flip-flop(i.e Flip-flop 1) and so on. As soon as clock pulse changes out put is going to change (at the negative edge of clock pulse) as a Down count sequence. For 3 bit Down counter Truth table is as shown below.

In both the counters Inputs J and K are connected to Vcc, hence J-K Flip flop can work in toggle mode. Preset and Clear both are connected to logic 1.

Table 2: Truth table for 3-bit Down counter

Counter States	**Count**			
	Q_2	**Q_1**	**Q_0**	
8	**0**	**0**	**0**	
7	**1**	**1**	**1**	
6	**1**	**1**	**0**	
5	**1**	**0**	**1**	
4	**1**	**0**	**1**	
3	**0**	**1**	**1**	
2	**0**	**1**	**0**	
1	**0**	**0**	**1**	
0	**0**	**0**	**0**	

6.3.1 Implementation of 3-bit Down counter using IC 7476:-

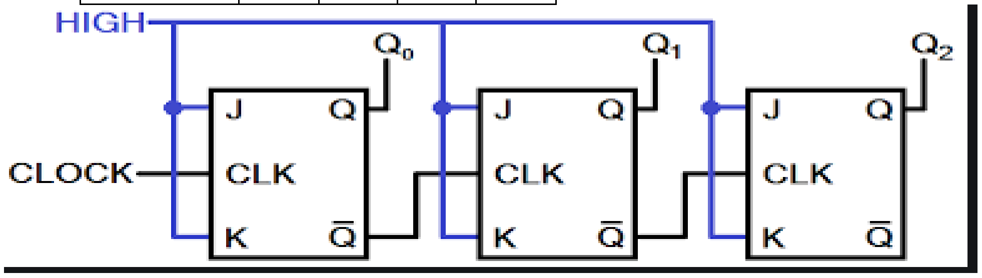
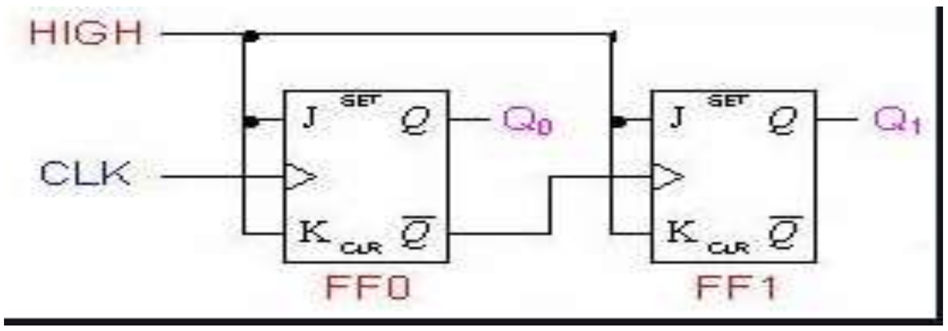


Fig 2: **3 Bit Asynchronous Down Counter**

Table 2: Truth table for 3-bit Down counter

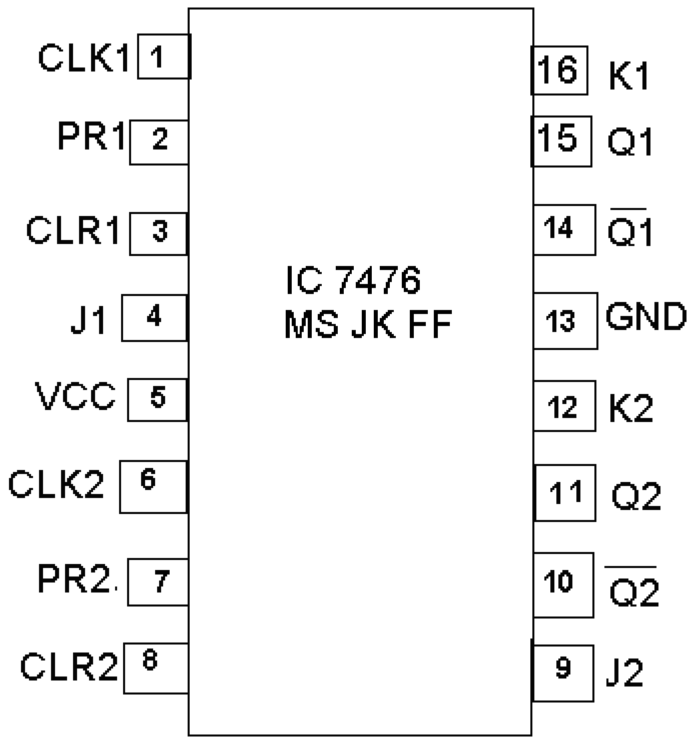
Counter States	**Count**			
	Q_1	**Q_0**		
3	**1**	**1**		
2	**1**	**0**		
1	**0**	**1**		
0	**0**	**0**		



6.5 Uses:-

1) The counters are very important digital circuits.
2) The synchronous counters are specially used as the counting devices.
3) They are also used to count number of pulses applied.
4) It also works for dividing frequency.
5) It helps in counting the number of product coming out of the machinery where product is coming out at equal interval of time.

7. Pin diagram:-



Conclusion:

Thus we checked & verified truth table of counters. And also studied Asynchronous up,down,up/down counter using IC 7476 & basic gates.

Assignment No: 9

Title: Assignment based on Sequential circuit design

Problem Statement: Design of Synchronous 3 bit Up and Down Counter using MSJK Flip Flop / D Flip Flop.

Objective: To learn and understand design and construction of sequential circuit.

Outcomes: On completion of these practical, students will be able to

CO1: Understand the working of synchronous counter.

CO2: Apply the knowledge flip flop as per the design specifications.

CO3: Design and implement synchronous counter circuits.

Hardware Requirement:

i) IC 7404(Not-gate), 7432 (OR-gate), 7408 (AND-gate), 7486 (Ex-or gate)
ii) Filp flop:JK
iii) Digital Trainer Kit -1
iv) Patch cords

Software Requirement: Digital Work 3.0,Android app:Logic Circuit SIM

Theory:

Introduction:-

Counters are logical device or registers capable of counting the number of states or number of clock pulse arriving at its clock input where clock is a timing parameter arriving at regular intervals of time, so counters can be also used to measure time & frequencies. They are made up of flip flops. Where the pulse are counted to be made of it goes up step by step & the o/p of counter in the flip flop is decoded to read the count to its starting step after counting n pulse incase of module & counters.

Types of counter:-

Counter are of two types:

1) Synchronous counter.

2) Asynchronous counter.

Synchronous counter :-

When counter is clocked such that each flip flop in the counter is triggered at the same time, the counter is called as synchronous counter.

Synchronous binary counter have regular & can easily be constructed with complimentary flip flop & gates.

The gates propagation delay at reset time will not be present or we may say will not occur.

Types of synchronous counter:

1) Synchronous up counter.

2) Synchronous down counter.

3) Up down counter.

Synchronous up counter:-

The up counter counts binary form 0 to7 i.e.(000 to 111).for this we are using JK flip flop.

In IC 74LS76,2 J-K flip flops are present. The clock pulse is given at pin 1 to 6 of the 1st IC & pin 1 of 2nd IC. This unable to apply clock at a time to all these flip flop.

Table 1:-Excitation Table for JK FF

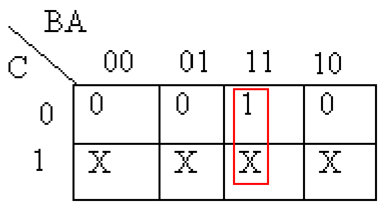
Present State	Next State	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Table 2: State table for synchronous up counter

Present state			Next state			Flip flop 3		Flip flop 2		flip flop 1	
C	B	A	C	B	A	JC	KC	JB	KB	JA	KA
0	0	0	0	0	1	0	x	0	x	1	x
0	0	1	0	1	0	0	x	1	x	x	1
0	1	0	0	1	1	0	x	x	0	1	x
0	1	1	1	0	0	1	x	x	1	x	1
1	0	0	1	0	1	x	0	0	x	1	X
1	0	1	1	1	0	x	0	1	x	x	1
1	1	0	1	1	1	x	0	x	0	1	X
1	1	1	0	0	0	x	1	x	1	x	1

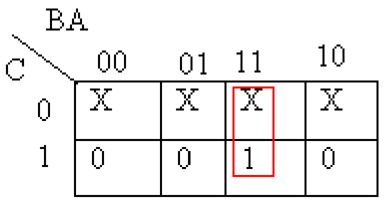
1 K-Map Simplification:-

K-map for JC



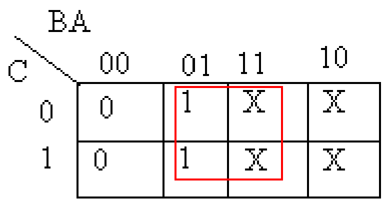
JC = BA

K-map for KC



KC = BA

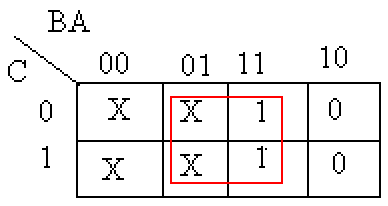
K-map for JB



C \ BA	00	01	11	10
0	0	1	X	X
1	0	1	X	X

JB = A

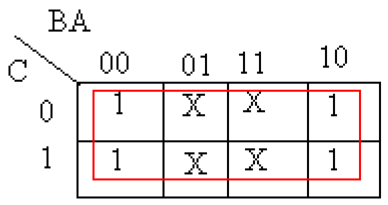
K-map for KB



C \ BA	00	01	11	10
0	X	X	1	0
1	X	X	1	0

KB= A

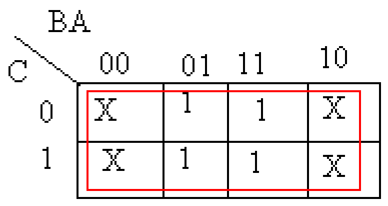
K-map for JA



C \ BA	00	01	11	10
0	1	X	X	1
1	1	X	X	1

JA = 1

K-map for KA



C \ BA	00	01	11	10
0	X	1	1	X
1	X	1	1	X

KA=1

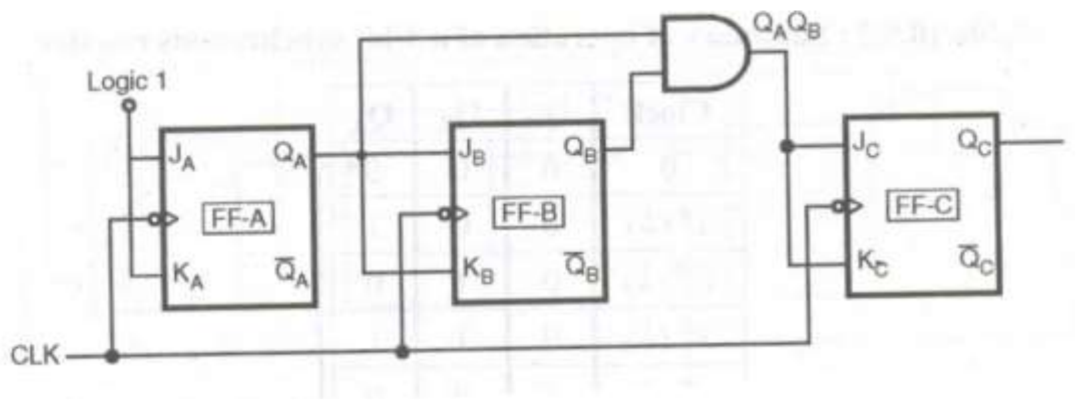


Fig 1: Logic diagram for Synchronous Up Counter

.2 Synchronous Down counter:-

This is used to count pulse from 7-0 i.e.(111-000).for this also 2 IC's of 74LS76 are required & hence we use 3JK flip flops.

Here also clock is given to 1st & 6th pin of 1st IC & 1st pin of 2nd IC enabling to apply clock to all flip flop at a time.

Table 3: State table for synchronous down counter

Present state			Next state			Flip flop 3		Flip flop 2		Flip flop 1	
C	B	A	C	B	A	JC	KC	JB	KB	JA	KA
1	1	1	1	1	0	X	0	X	0	X	1
1	1	0	1	0	1	X	0	X	1	1	X
1	0	1	1	0	0	X	0	0	X	X	1
1	0	0	0	1	1	X	1	1	X	1	X
0	1	1	0	1	0	0	X	X	0	X	1
0	1	0	0	0	1	0	X	X	1	1	X
0	0	1	0	0	0	0	X	0	X	X	1

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

6.3.2.1 K-Map Simplification:-

JC = B’A’

KC = B’A’

JB = A’

KB= A’

JA = 1

KA=1

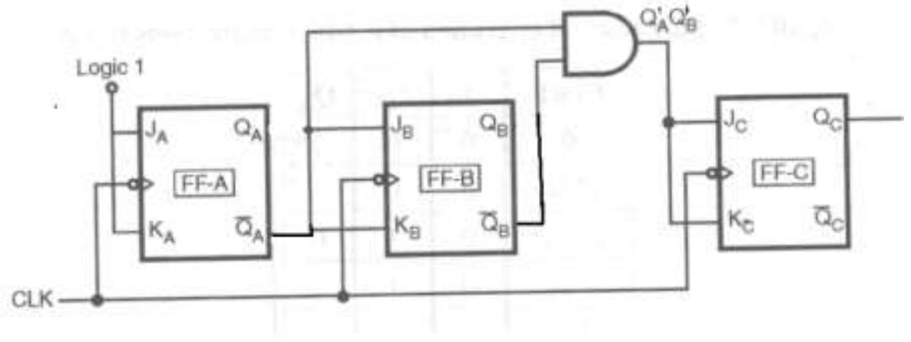


Fig 2: Logic diagram for Synchronous Down Counter

Uses:-

1)The counters are very important digital circuit .

2) The synchronous counter are specially used as the counting devices.

3) They are also used as counter to count the no of clock pulses applied.

4) It also works for counting frequency & is used in frequency divider circuit .

5) It is used in digital voltmeter.

6) It is also used in counter type A to D converter.

7) It is also used for time measurement.

8) It is also used in digital triangular wave generator.

9) It helps in counting the no of product coming out from machinery where product is coming out at equal interval of time.

Conclusion:

Up and down counters are successfully implemented, the counters are studied & o/p are checked. The truth table is verified

Enhancements/modifications:

As the design part is done for the 3 bit Counter, we can implement the same for 4 bit counter.

Assignment No: 10

Title: Assignment based on Sequential circuit design

Problem Statement: Realization of Mod -N counter using (Decade Counter IC 7490) .

Objective: To learn and understand design and construction of Mod n Counter.

Outcomes: On completion of these practical, students will be able to

CO1: Understand the working of mod n counter.

CO2: Apply the knowledge IC 7490 as per the design specifications.

CO3: Design and implement mod n counter circuits.

Hardware Requirement:

i) IC 7404(Not-gate), 7432 (OR-gate), 7408 (AND-gate), 7490(mod n counter)

ii) Digital Trainer Kit -1

iii) Patch cords

Software Requirement: Digital Work 3.0,Android app:Logic Circuit SIM

Theory:

Introduction:-

Counters are logical device or registers capable of counting the number of states or number of clock pulse arriving at its clock input where clock is a timing parameter arriving at regular intervals of time, so counters can be also used to measure time & frequencies. They are made up of flip flops.

Ripple counter IC-7490 (decade counter):IC-7490 is a TTL MSI decade counter. It contains four master slave flip flops and additional gating to provide a divide-by-two counter and a three stage binary counter which provides a divide by 5 counter.

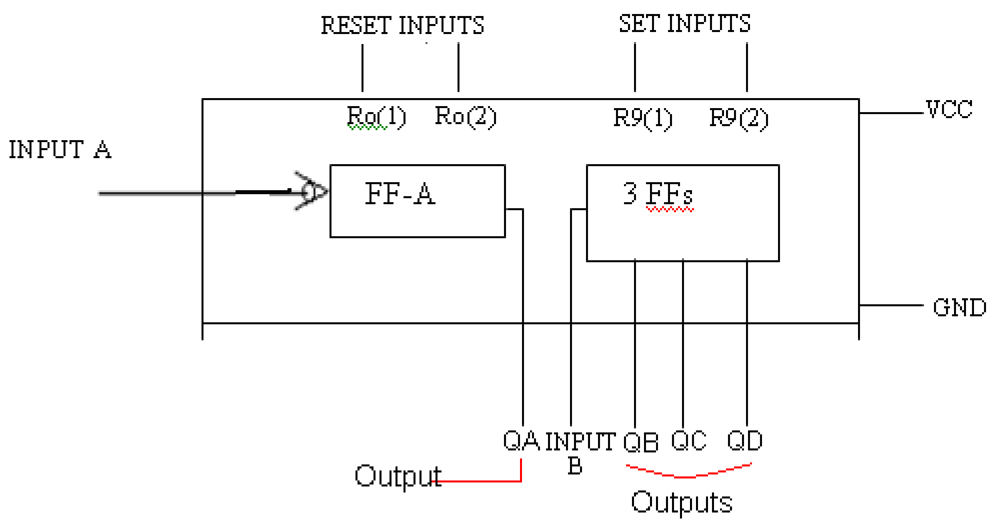


Fig.1 The Basic internal structure of IC 7490

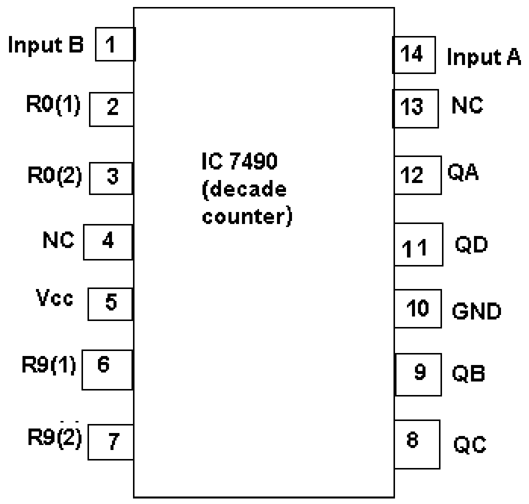


Fig. 2 Pin configuration of IC7490.

Table 1:-pin name and decription of IC7490

Pin name	Description
Input B	This is clock input to the internal MOD-5 ripple counter,which is negative edge triggered.
R0(1),R0(2)	Gated zero reset inputs

R9(1),R9(2)	These are gated set to nine inputs
QD,QC,QB	Output of internal MOD-5 counter with QD as MSB.
QA	Output of internal MOD-2 counter with QA as LSB.
Input A	Clock input to FF-A which is negative edge triggered.

The reset/count function table of IC7490 is shown in table 2.

Table 2:-Reset/count truth table

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

2 MOD-10 OR Decade Counter:-

The output of MOD-2 is externally connected to the input B which is the clock input of the internal MOD-5 counter. Therefore QA toggles on every falling edge of clock input whereas the output QD,QC,QB of the MOD-5 counter will increment from 000 to 100 on low going change of QA output.

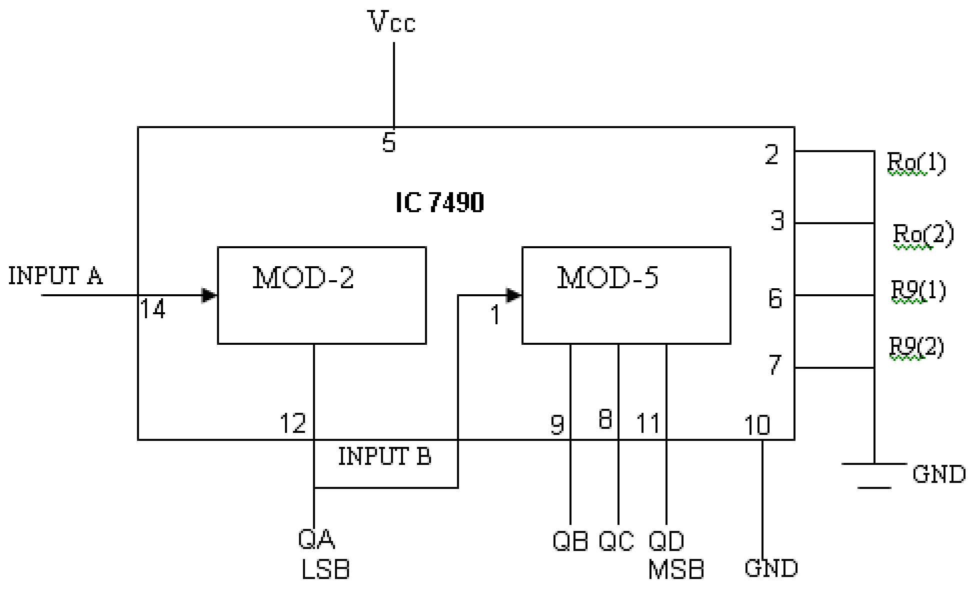
Due to cascading of MOD-2 and MOD-5 counter,the overall configuration becomes a MOD-10 i.e. decade counter.

The reset inputs Ro(1),Ro(2) and preset inputs R9(1),R9(2) are connected to ground so as to make them inactive.

Table 3:-Truth table for mod-10

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

Implimentation:



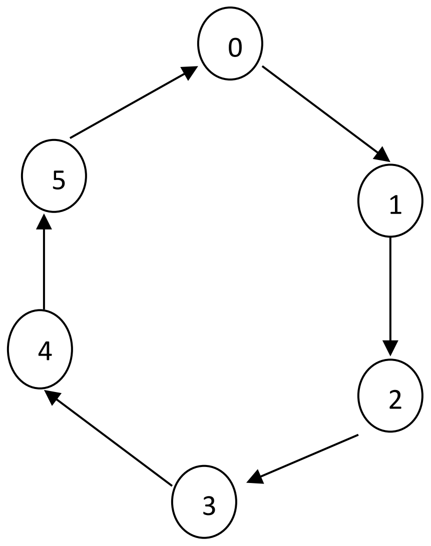
Note:- Pin 4 & 13 For NC

Fig 3.MOD-10 counter using IC 7490

3 MOD-6 Counter :-

It requires one IC 7490. The design of it using decade counter is as follows

State Diagram:



Implimentation:

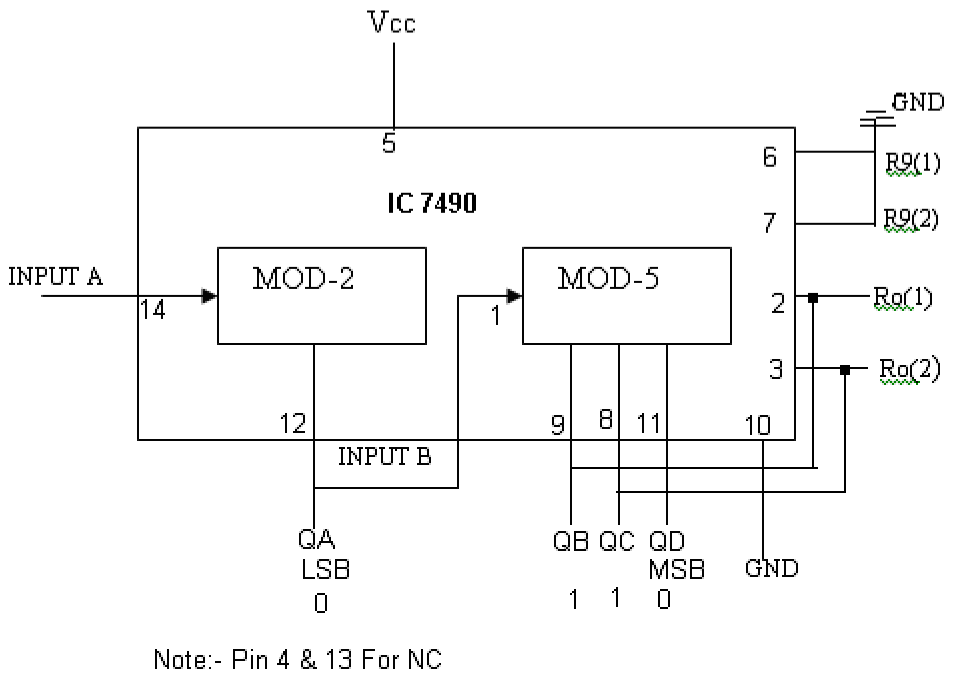


Fig 4. MOD-6 counter using IC 7490

4 MOD-100 Counter:-

MOD-100 counter can be obtained by cascading two decade counter IC's .

Each IC gives divide by 10 count. Hence 2 ICs will give 10*10=100 counts.

Clock is applied to the input A of IC 7490 , the Q1 output is connected to B input etc.

For cascading, the QD output of the first IC is connected to the input A of second IC. Thus QD of IC-(1) acts as clock input to IC-(2).

The Ro(1), Ro(2), R9(1), R9(2) inputs of both ICs are connected to ground so as to make them inactive.

The total count provided by the two counter varies from 00 to 99 decimal that means 100 states. Hence it is a MOD-10 counter.

Implementation:

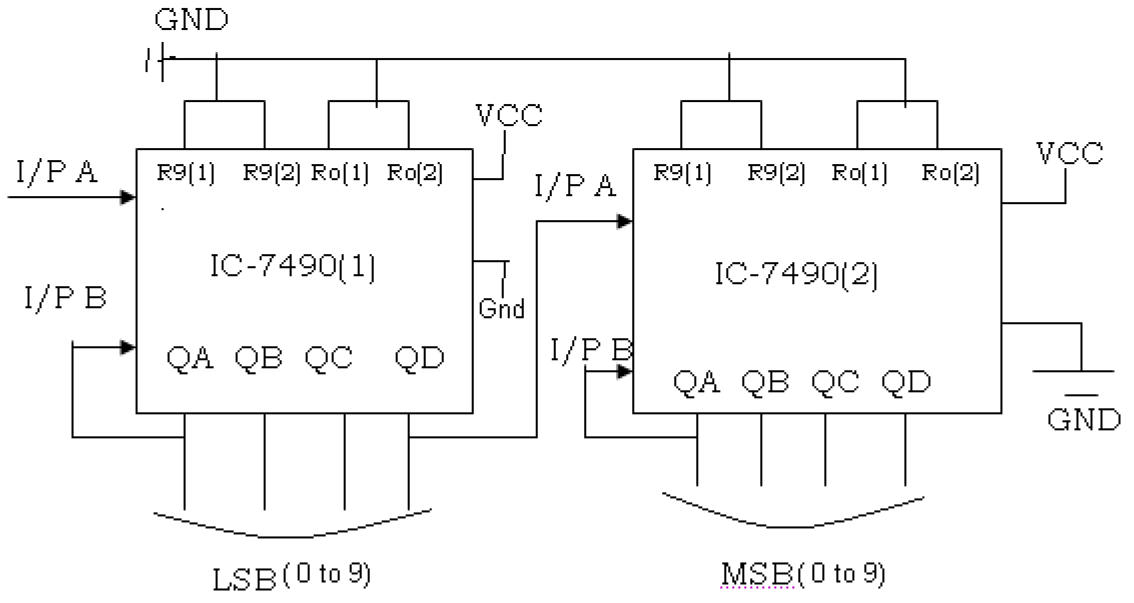


Fig 5.MOD-100 counter using IC 7490

5 MOD-96 Counter :-

It requires two IC-7490. One is considered LSB where the units place digits occur.

The other IC is considered MSB where the tens place digits occur.

The counter counts in the up direction from 0 to 96 and after 96 it returns to 0.

The reset output Y is connected or given to both the reset pins of the two IC's which resets the counter .

The truth table and K-maps are not feasible to make hence it is implemented using a decade counter.

Assignment No: 11

Title: Assignment based on Sequential circuit design

Problem Statement: Design and implement Sequence detector using MS JK flip-flop.

Objective: To learn and understand design and construction of sequential circuit.

Outcomes: On completion of these practical, students will be able to

CO1: Understand the working of Sequence detector.

CO2: Apply the knowledge flip flop as per the design specifications.

CO3: Design and implement Sequence detector circuits.

Hardware Requirement:

i) IC 7404(Not-gate), 7432 (OR-gate), 7408 (AND-gate), 7486 (Ex-or gate)

ii) Filp flop:JK

iii) Digital Trainer Kit -1

iv) Patch cords

Software Requirement: Digital Work 3.0,Android app:Logic Circuit SIM

Theory:

Introduction:-

- To detect the desired sequence.
- It checks the sequence bit by bit, output will be equal to 1 if the complete sequence is detected.
- A Sequence Circuit, Which detects a prescribed sequence of bits in synchronism with a clock, is referred to as sequence detector.

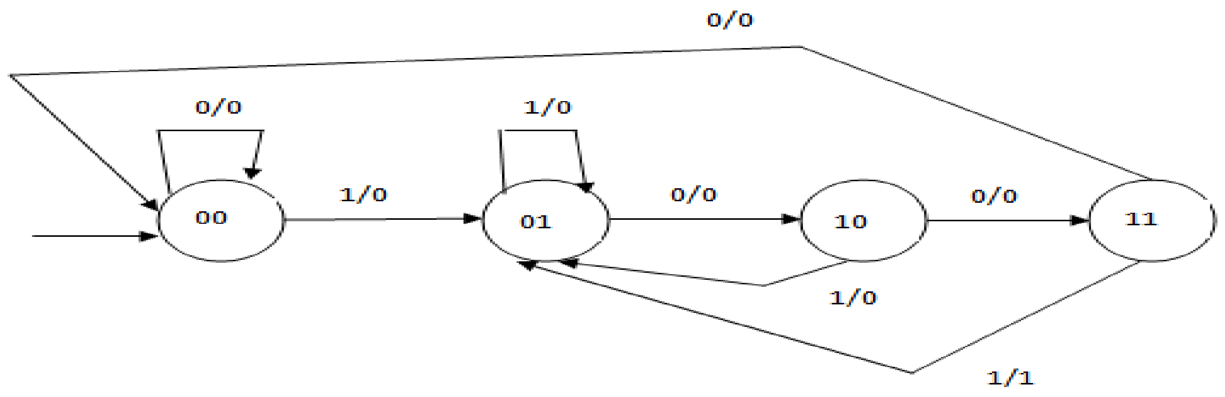
Steps for design the Sequence Detector:

1. A sequence to be detected is given us.
2. Develop the state diagram

3. Write state table and circuit excitation table
4. K-map and obtain Simplified equation
5. Draw Logic diagram

Problem Statement

- Design sequence detector using J K flip-flop to detect the following sequence: 1001
- Solution: State diagram



Excitation table of JK FF

Present state	Next state	Jn	Kn
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

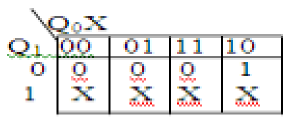
Circuit Excitation table:

Present states	Next state	FF 1	FF 0

Q_1	Q_0	X	Q_{1+1}	Q_{0+1}	Z	J_1	K_1	J_0	K_0
0	0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	0	X	1	X
0	1	0	1	0	0	1	X	X	1
0	1	1	0	1	0	0	X	X	0
1	0	0	1	1	0	X	0	1	X
1	0	1	0	1	0	X	1	1	X
1	1	0	0	0	0	X	1	X	1
1	1	1	0	1	1	X	1	X	0

K-map and Simplification:

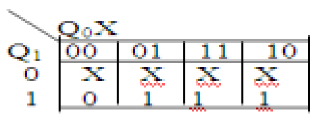
For J1



Q_1 \ Q_0X	00	01	11	10
0	0	0	0	1
1	X	X	X	X

$J_1 = Q_0\overline{X}$

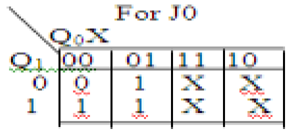
For K1



Q_1 \ Q_0X	00	01	11	10
0	X	X	X	X
1	0	1	1	1

$K_1 = X + Q_0$

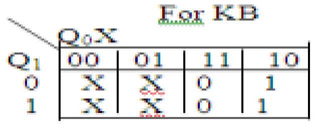
For J0



Q_1 \ Q_0X	00	01	11	10
0	0	1	X	X
1	1	1	X	X

$J_0 = X + Q_1$

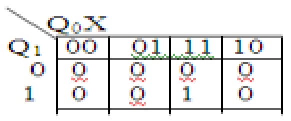
For KB



Q_1 \ Q_0X	00	01	11	10
0	X	X	0	1
1	X	X	0	1

$K_0 = \overline{X}$

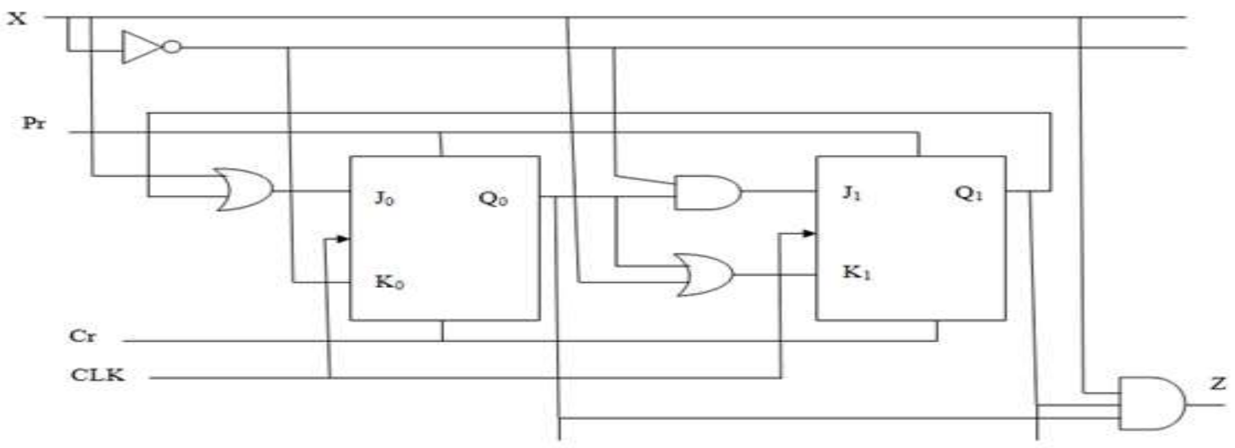
For Z



Q_1 \ Q_0X	00	01	11	10
0	0	0	0	0
1	0	0	1	0

$Z = Q_1Q_0X$

Fig 13.1 Connection Diagram:-



Conclusion:

Sequence detector successfully implemented.

Assignment No: 12

Title: Assignment based on Sequential circuit design

Problem Statement: Study of Shift Registers (SISO,SIPO, PISO, PIPO).

Objective: To learn and understand shift register of sequential circuit.

Outcomes: On completion of these study assignment, students will be able to

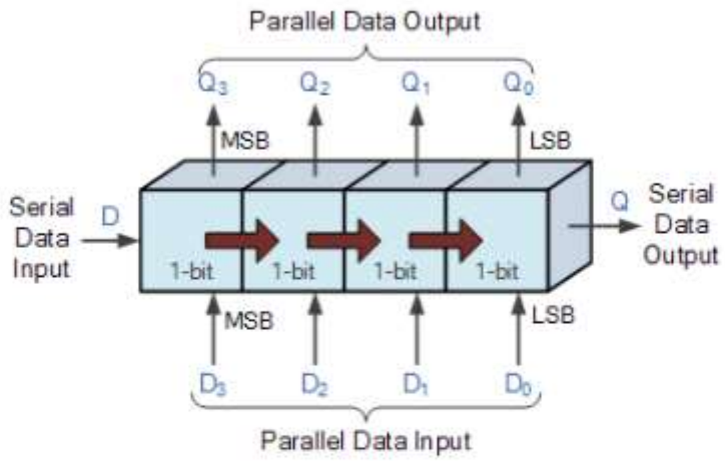
CO1: Understand the shift register.

CO2: Understand the types of shift register

Introduction:-

The Shift Register is another type of sequential logic circuit that can be used for the storage or the transfer of binary data.

This sequential device loads the data present on its inputs and then moves or "shifts" it to its output once every clock cycle, hence the name Shift Register



Mode of Shift Register

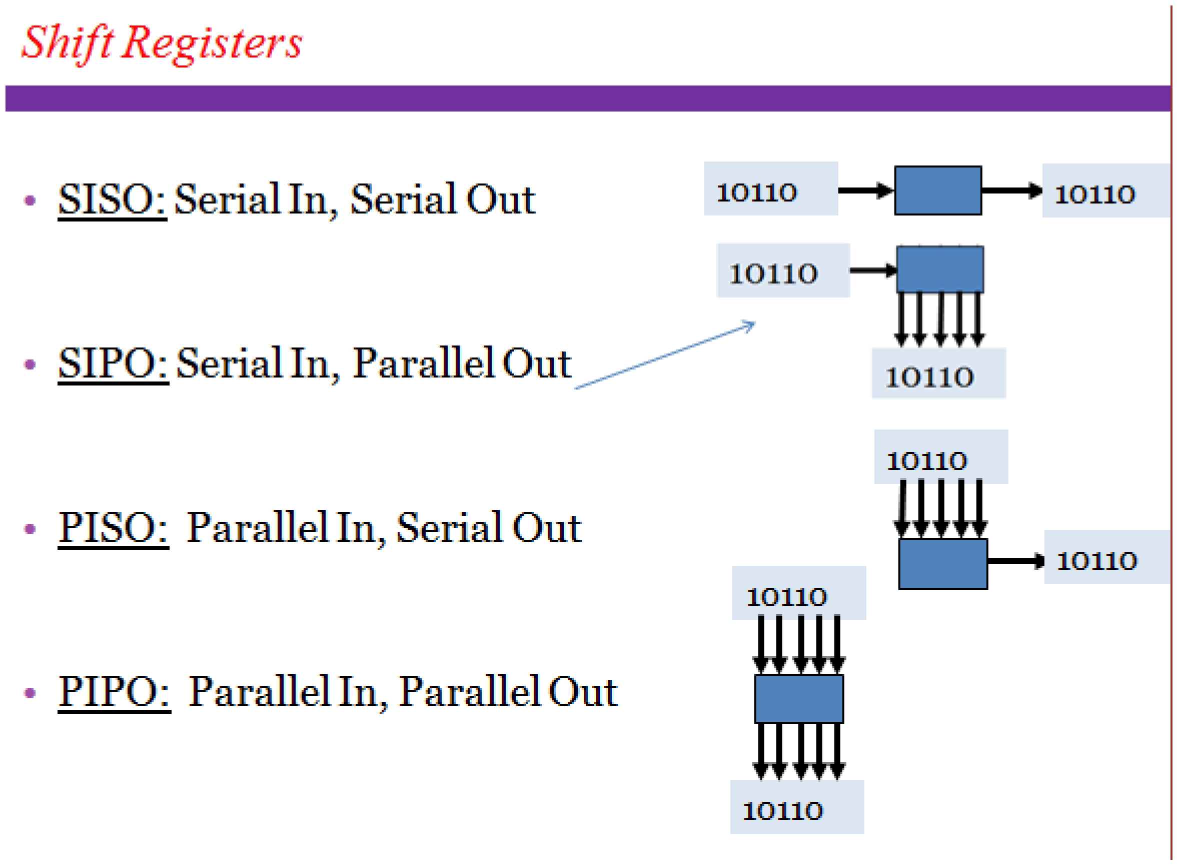
Serial In Serial Out (SISO)

Serial In Parallel Out (SIPO)

Parallel In Serial Out (PISO)

Parallel In Parallel Out (PIPO)

Bidirectional Shift Register



Shift Registers

- SISO: Serial In, Serial Out
- SIPO: Serial In, Parallel Out
- PISO: Parallel In, Serial Out
- PIPO: Parallel In, Parallel Out

shift registers operate in one of four different modes with the basic movement of data through a shift register being:

- Serial-in to Parallel-out (SIPO) - the register is loaded with serial data, one bit at a time, with the stored data being available at the output in parallel form.
- Serial-in to Serial-out (SISO) - the data is shifted serially “IN” and “OUT” of the register, one bit at a time in either a left or right direction under clock control.
- Parallel-in to Serial-out (PISO) - the parallel data is loaded into the register simultaneously and is shifted out of the register serially one bit at a time under clock control.
- Parallel-in to Parallel-out (PIPO) - the parallel data is loaded simultaneously into the register, and transferred together to their respective outputs by the same clock pulse.

Conclusion:

In this way we study about shift register.