DIGITAL LOGIC
(Assignment-1)

1. Differentiate between Analog and Digital system.

⇒

Analog system	Digital system
- It uses electronic pulses with varying magnitude to send data.	- It uses binary format 0 and 1.
- In analog system, hardwares are not flexible.	- In digital system, hardwares can be easily modified as per requirement.
- It is cheaper.	- It is expensive.
- An example of analog system is galvanometer.	- An example of digital system is calculator.

2. Convert the following decimal numbers into hexadecimal and octal number.

a) ~~170.46~~ 304

Here,

For hexadecimal,

16 | 304 → 0
16 | 19 → 3
1

$\therefore (304)_{10} = (130)_{16}$

For octal,

8 | 304 → 0
8 | 38 → 6
4

$\therefore (304)_{10} = (460)_8$

b) 224

Here,

For hexadecimal,

16 | 224 → 0
14

$\therefore (224)_{10} = (E0)_{16}$

For Octal,

8 | 224 → 0
8 | 28 → 4
3

$\therefore (224)_{10} = (340)_8$

3. Convert the following octal numbers to hexadecimal.

a) 1760.46

Here,

1 7 6 0 . 4 6 → Octal bits

001 111 110 000 . 100 110 → Equivalent binary bits

0011 1111 0000 . 1001 1000 → Grouping into 4 binary bits

3 F 0 . 9 8

$\therefore (1760.46)_8 = (3F0.98)_{16}$

b) 6055.263

Here,

6 0 5 5 . 2 6 3 → Octal bits

110 000 101 101 . 010 110 011 → Equivalent binary bits

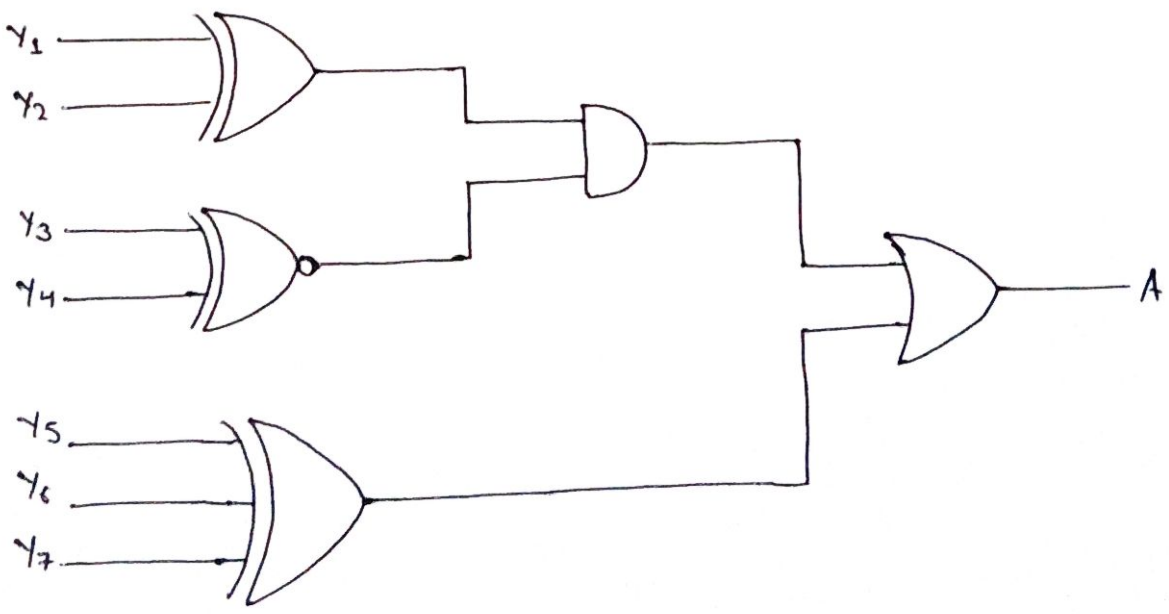
1100 0010 1101 . 0101 1001 1000 → Grouping into 4 bits

C 2 D . 5 9 8 → Equivalent hexadecimal digit

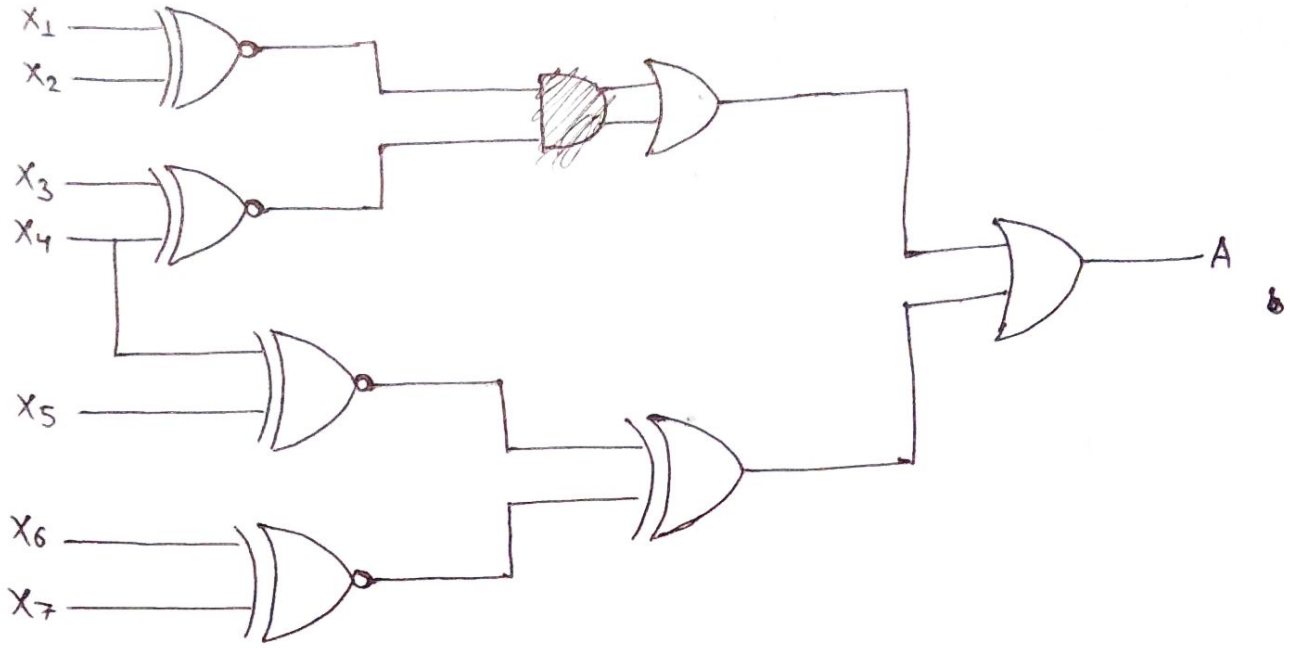
$\therefore (6055.263)_8 = (C2D.598)_{16}$

4. Draw a logic gate that implements the following:

a) $A = (Y_1 \oplus Y_2)(Y_3 \odot Y_4) + (Y_5 \oplus Y_6 \oplus Y_7)$



b) $A = (X_1 \odot X_2) + (X_3 \odot X_4) + (X_4 \odot X_5) \oplus (X_6 \odot X_7)$



5. State and prove De-Morgan's theorem 1st and 2nd with logic gates and truth table.

⇒ De-Morgan's theorem is used to convert OR type operation into AND type and vice-versa.

1st law: It states that "the total complement of sum is equal to the product of individual complements.

For example:

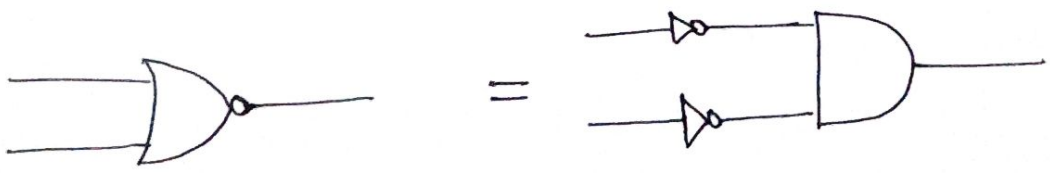
$(A+B)' = A' \cdot B'$

2nd law: It states that "the total complement of product is equal to the sum of individual complements.

For example:

$(A \cdot B)' = A' + B'$

~~A B A'~~ ~~B A+B (A+B)'~~



1st law

2nd law

A	B	$(A+B)'$	$A'.B'$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

1st law

A	B	$(A.B)'$	$A'+B'$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

2nd law

6. Which gates can be used as inverters in addition to the NOT gate and How?

⇒ Buffer invert, AND-invert and OR-invert can be used as inverters in addition to the NOT gate because they convert 0 to 1 and 1 to 0.

(Buffer invert)

(AND-invert)

(OR-invert)

7. Represent the decimal number 8620

a) in BCD

8 6 2 0

1000 0110 0010 0000

$\therefore (8620)_{10} = (1000011000100000)_{BCD}$

b) in excess-3 code

8	6	2	0
+3	+3	+3	+3
11	9	5	3
1011	1001	0101	0011

$\therefore (8620)_{10} = (1011100101010011)_{excess-3}$

c) in 2,4,2,1 code

8	6	2	0
1110	1100	0010	0000

$\therefore (8620)_{10} = (1110110000100000)_{2421}$

d) in binary number

2	8620	→ 0
2	4310	→ 0
2	2155	→ 1
2	1077	→ 1
2	538	→ 0
2	269	→ 1
2	134	→ 0
2	67	→ 1
2	33	→ 1
2	16	→ 0
2	8	→ 0
2	4	→ 0
2	2	→ 0
	1	

$\therefore (8620)_{10} = (10000110101100)_2$

8. Convert the following hexadecimal numbers into decimal and octal numbers.

a) 0FFF

Here,

$(0FFF)_{16}$

$= 0 \times 16^3 + F \times 16^2 + F \times 16^1 + F \times 16^0$

$= 0 + 15 \times 16^2 + 15 \times 16 + 15 \times 1$

$= (4095)_{10}$

$\therefore (0FFF)_{16} = (4095)_{10}$

And,

0 F F F

0000 1111 1111 1111

000 111 111 111 111

0 7 7 7 7

$\therefore (0FFF)_{16} = (7777)_8$

b) 3FFF

Here,

$(3FFF)_{16}$

$= 3\times16^3 + F\times16^2 + F\times16^1 + F\times16^0$

$= 3\times16^3 + 15\times16^2 + 15\times16 + 15\times1$

$= 12288 + 3840 + 240 + 15$

$= (16383)_{10}$

$\therefore (3FFF)_{16} = (16383)_{10}$

And,

3 F F F

0011 1111 1111 1111

011 111 111 111 111

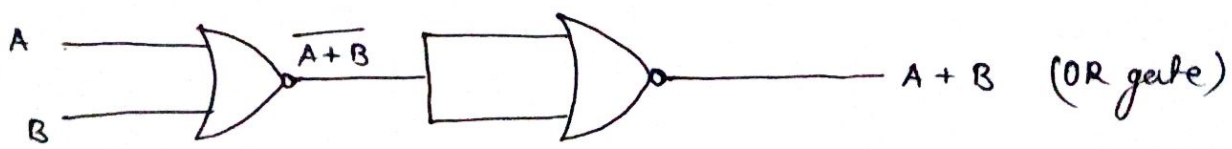
3 7 7 7 7

$\therefore (3FFF)_{16} = (37777)_8$

9. What do you mean by universal gate? Realize the following logic gates using NOR gates.
a) OR gate b) AND gate

⇒ Universal gates are those gates from which basic gates: OR gate, AND gate and ~~NOR~~ NOT gate can be obtained. NOR gate and NAND gates are two universal gates.

a) Using NOR gate to obtain OR gate:



10. What do you mean by Gray code? What are its applications?

⇒ Gray code is a code used to represent digits generated from a mechanical sensor that may be prone to error.
In Gray code, there is only one bit location different between two successive values which makes mechanical transitions from one digit to the next less prone error.
Gray code is used in Kmap, analog-digital computer etc.

11. Write the first 20 decimal digits in base 3.

⇒

Decimal	Base 3
0	0
1	1
2	2
3	10
4	11
5	12
6	20
7	21
8	22
9	100
10	101
11	102
12	110
13	111
14	112
15	120
16	121
17	122
18	200
19	201
~~20~~	

12. Convert the decimal number 250.5 to base 3, base 4, base 7, base 8 and base 16.

⇒ a) $(250.5)_{10} = (?)_3$

3 | 250 → 1
3 | 83 → 2
3 | 27 → 0
3 | 9 → 0
3 | 3 → 0
1

$0.5 \times 3 = 1.5 \longrightarrow 1$
$0.5 \times 3 = 1.5 \longrightarrow 1$
- - - - - - -

$\therefore (250.5)_{10} = (100021.1111\ldots)_3$

b) $(250.5)_{10} = (?)_4$

$0.5 \times 4 = 2.0 \longrightarrow 2$

4	250	→ 2
4	62	→ 2
4	15	→ 3
	3	

$\therefore (250.5)_{10} = (3322.2)_4$

c) $(250.5)_{10} = (?)_7$

7	250	→ 5
7	35	→ 0
	5	

$0.5 \times 7 = 3.5 \longrightarrow 3$

$0.5 \times 7 = 3.5 \longrightarrow 3$

$\therefore (250.5)_{10} = (505.333\ldots)_7$

d) $(250.5)_{10} = (?)_8$

8	250	→ 2
8	31	→ 7
	3	

$0.5 \times 8 = 4.0 \longrightarrow 4$

$\therefore (250.5)_{10} = (372.4)_8$

e) $(250.5)_{10} = (?)_{16}$

16	250	→ A
	15	

$0.5 \times 16 = 8$

$\therefore (250.5)_{10} = (FA.8)_{16}$

14. Perform subtraction with the following binary numbers using both 1's complement and 2's complement.

a) 11010 − 1101

Here,

Using 1's complement,

$= 11010 + (-(01101))$

$= 11010 + 10010$

$= (1)\ 01100$

$= 1101$

Using 2's complement,

$11010 + (-01101)$

$\rightarrow 11010 + 10011$

$\rightarrow (1)01101$

$\rightarrow 01101$

$\therefore 11010 - 01101 = 01101$

b) $11010 - 10000$

Here,

Using 1's complement,

$11010 + (-10000)$

$\rightarrow 11010 + 01111$

$\rightarrow (1)01001$

$\rightarrow 01001$

Using 2's complement,

$11010 + (-10000)$

$\rightarrow 11010 + 10000$

$\rightarrow (1)01010$

$\rightarrow 01010$

Hence,

$11010 - 10000 = 01010$

c) $10010 - 10011$

Here,

Using 1's complement,

$10010 + (-10011)$

$\rightarrow 10010 + 01100$ [Taking 1's complement]

$\rightarrow 11110$ [Adding]

$\rightarrow 00001$ [Taking 1's complement]

$\rightarrow -1$

Using 2's complement,

$10010 + (-10011)$

$\rightarrow 10010 + 01101$ (Taking 2's complement)

$\rightarrow 11111$

$\rightarrow 00001$ (Taking 2's complement)

$\rightarrow 1$

Hence,

$10010 - 10011 = -1$

d) 100 - 110000

Here,

Using 1's complement,

000100 + (- 110000)

000100 + 001111 → 1's complement

010011

- 101100 → 1's complement

Using 2's complement,

000100 + (-110000)

000100 + 010000 → 2's complement

010100

- 101100 → 2's complement

Hence,

100 - 110000 = -101100

13. Perform subtraction with the following decimal numbers using both 9's complement and 10's complement.

a) 5250 - 321

Here,

Using 9's complement,

5250 + (-0321)

5250 + 9678 → 9's complement

(1) 4928

4929 → Adding carry

Using 10's complement,

5250 + (-0321)

5250 + 9679 → 10's complement

(1) 4929

4929 → Ignoring carry

Hence,

5250 - 321 = 4929

b) 3570 - 2100

Here,

Using 9's complement,

3570 + (-2100)

3570 + 7899 → 9's complement

(1) 1469

1470 → Adding carry

Using 10's complement,

3570 + (-2100)

3570 + 7900 → 10's complement

(1) 1470

1470 → Ignoring carry

Hence,

3570 - 2100 = 1470

c) 753 - 864

Here,

Using 9's complement,

753 + (-864)

753 + 135 → 9's complement

888

-111 → 9's complement, negative answer

Using 10's complement,

753 + (-864)

753 + 136 → 10's complement

889

-111 → 10's complement

Hence,

753 - 864 = -111

d) 20 - 1000

Here,

Using 9's complement,

0020 + (-1000)

0020 + 8999 → 9's complement

9019

-0980 → 9's complement

-980

Using 10's complement,

0020 + (−1000)

0020 + 89000 ⟶ 10's complement

89020

−0980 ⟶ 10's complement

−980

Hence,

20 − 1000 = −980

15. Convert the following:

a) $(A08E.FA)_{16} = (?)_{10}$

Here,

$(A08E.FA)_{16}$

$= A \times 16^3 + 0 \times 16^2 + 8 \times 16^1 + E \times 16^0 + F \times 16^{-1} + A \times 16^{-2}$

$= 10 \times 16^3 + 0 + 8 \times 16 + 14 \times 1 + 15 \times 16^{-1} + 10 \times 16^{-2}$

$= 40960 + 128 + 14 + 0.9375 + 0.039$

$= (41102.976)_{10}$

b) $(AE9.B0E)_{16} = (?)_2$

Here,

A E 9. B 0 E

1010 1110 1001. 1011 0000 1110

1010 1110 1001. 101100001110

$\therefore (AE9.B0E)_{16} = (1010111010011011000011110)_2$

16. Perform $(-44)_{10} - (67)_{10}$ using 10's complement method.

Here,

$(-44)_{10} - (67)_{10} = (?)_{10}$

10's complement of 044 = 955 + 1 = 956

10's complement of 067 = 932 + 1 = 933

Now,

(−44) + (−67)

= 956 + 933

= 1889

10's complement of 889 = 110 + 1 = 111

$\therefore (-44)_{10} - (67)_{10} = (-111)_{10}$

17. Differentiate between weighted code and non-weighted code with examples.

⇒

Weighted Code	Non-Weighted Code
- It is positionally weighted.	- It is not positionally weighted
- In this code, ~~each~~ the position of each digit represents a specific weight.	- In this code, no weight is represented by position of digits.
- For eg: 8421, BCD, 2421 are weighted codes.	- For eg: excess-3, and gray code non-weighted codes.

18. Explain why Excess-3 code is self-complementary code with an example.

⇒ Excess-3 code is self-complementary code because excess-3 code is the complement form of given number.
In other words,
One's complement for excess-3 is the excess-3 code for 9's complement of given decimal number.

For example:
Excess-3 for 5 is 8 (1000) and 1's complement of 1000 is 0111 which is excess-3 code for 9's complement of 5.

19. What is Parity bit? Explain how it is used as error detection code.

⇒ Parity bit is a bit which is used for error detection.
There are two types of parity bit:
i) Even parity
ii) Odd parity

A parity bit is an extra 0 or 1 bit that is attached to the original signal and used to detect errors. In the even parity method, the value of the bit is chosen so that the total number of 1's in the transmitted signal, including the parity bit, is even.
Similarly, in the odd parity method, the total number of 1's in the transmitted signal, including the parity bit, is odd.

20. Explain IC Digital Logic Families with their characteristics.

⇒ In digital electronics, a logic family refers to digital integrated circuit devices which are constructed with combination of electronic gates.
A family has its own power supply voltage and distinct logic levels with their own characteristics, advantages and disadvantages.
Following are the IC Digital Logic Families with their characteristics:

i) Diode Logic (DL)

In Diode logic, all the logic is implemented with the use of resistors and diodes. The purpose of the diodes is to perform OR and AND operations. Diodes can also be used as a logical switch. The disadvantage of diodes is that they tend to degrade the signals quickly and cannot perform the NOT operation which limits their functionality.

ii) Resistor-Transistor Logic (RTL)

In RTL family, all the logic is implemented with the use of transistors and resistors. Input signals are combined with the use of transistors, then inverted and amplified. RTL gates are not very expensive and very simple to construct. The drawback in using RTL gates is that they draw a great amount of current from the power supply. They are used in slower applications, but cannot be used in today's computers as they cannot switch at high speeds.

iii) Diode Transistor Logic (DTL)

In DTL family, all the logic is implemented with the use of diodes and transistors. DTL has some advantages over DL and RTL. In DTL, signal can be restored to full logic levels if we add a transistor at the output of the logic gates. This results in logic inversion. In DTL, the OR operation can be performed by the diodes instead of resistors which removes the interaction between input signals. DTL was used in early vacuum tube computers.

iv) Transistor - Transistor Logic (TTL)

In TTL, the logic gates are constructed around the transistors. TTL uses bipolar transistors to construct its integrated circuits. TTL has become the standard logic circuit in many application for a number of years. TTL greatly decreases the manufacturing costs because multiple emitters can be added in the input so no extra space is needed and a multiple input gate can be constructed easily.

v) Emitter Coupled Logic (ECL)

In ECL family, the transistors are prevented from going into deep saturation so that there are no storage delays. This logic is used in applications with high speed environment. ECL is the fastest bi-polar circuit available today and it bypasses TTL in terms of speed. ECL is also a non-saturated logic. The design of ECL consists of termination resistors which allows the signals to propagate with very low reflection.

vi) Complementary Metal Oxide Semiconductor Logic (CMOS)

CMOS is known for its low power consumption and high fan-out. The transistors inside the CMOS are made from an NMOS transistor and PMOS transistor. To realize the logical functions, both P-type and N-type transistors are used. There is no power dissipation in CMOS. It is also considered to be one of the most reliable logic family today.

21. Write short notes on:

a) EBCDIC

EBCDIC is Extended Binary Coded Decimal Interchange Code data encoding system developed by IBM, that uses 8-bit binary code for each number and alphabet characters similar as ASCII and used in old computers. To convert into EBCDIC, we add 1 in front of ASCII format.

b) Integrated Circuit

An Integrated Circuit (IC) is a small chip of semiconductor material that mounts an entire circuit on itself. It is very small when compared to standard circuits.

c) Alphanumeric codes

Alphanumeric codes contain alphabets and numbers and there are three types of alphanumeric codes:

i. ASCII
ii. EBCDIC
iii. UNICODE UNIcode

"Assignment - 2"

Date:
Page:

Q.22 Describe the three Variable k-Map with examples.

Ans: The three Variable k-map consists of three binary variables as its name suggests and has eight min-terms for three variable. Therefore, the map contains eight squares. The three variables and their min-terms in k-map are arranged in a reflected code and not in a binary sequence.

For e.g.:

A \ BC	00	01	11	10
0	M_0	M_1	M_3	M_2
1	M_4	M_5	M_7	M_6

A \ BC	00	01	11	10
0	$\bar{A}\bar{B}\bar{C}$	$\bar{A}\bar{B}C$	$\bar{A}BC$	$\bar{A}B\bar{C}$
1	$A\bar{B}\bar{C}$	$A\bar{B}C$	ABC	$AB\bar{C}$

(SOP)

A \ BC	00	01	11	10
0	$A+B+C$	$A+B+\bar{C}$	$A+\bar{B}+\bar{C}$	$A+\bar{B}+C$
1	$\bar{A}+B+C$	$\bar{A}+B+\bar{C}$	$\bar{A}+\bar{B}+\bar{C}$	$\bar{A}+\bar{B}+C$

(POS)

Following are the rules to be followed:

(i) One square represents one min-term giving a term of three variables.

(ii) Two adjacent squares represent a term of two variables.

(iii) Four adjacent squares represent a term of one variable.

(iv) All eight or group of eight adjacent squares combine to give a constant one.

(i)

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

$F = \bar{A}$ (In SOP)

(ii)

A \ BC	00	01	11	10
0				
1			1	1

$F = A\bar{B}$ (In SOP)

(iii)

A \ BC	00	01	11	10
0		1		
1				

$F = \bar{A}\bar{B}C$ (In SOP)

(iv)

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

$F = 1$

Now, example of simplication of Boolean function with K-map,

$F = (A, B, C) = \sum m\,(2, 3, 5, 6, 7)$

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

$F = B + AC$

The equations are made in the terms of power of 2.

Q.23 What is combinational logic ? What are its important features?

Ans: Combinational logic is a type of digital logic which is implemented by Boolean circuits, where the output is a pure function of the present input only. Combinational logic is used in computer circuits to perform Boolean algebra on input

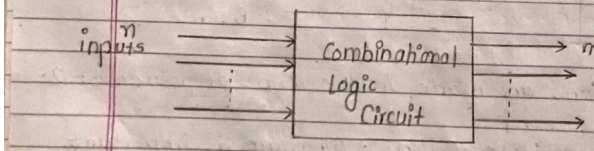
signal and on stored data. For e.g. the part of ALU that does mathematical calculation is done by or constructed by using combinational circuit. Other circuits used in computer such as half adder, full adder, half subtractor, multiplexer, demultiplexer, encoders and decoders are also made by using combinational circuit.

Following are the features of combinational circuits:

(i) At any instant of time, the output of the combinational circuits depends only on the present output terminals.

(ii) The combinational circuit doesnot have any backup or previous memory. The present state of circuit isn't affected by the previous state of the input.

(iii) The n number of inputs and m number of outputs are possible in combinational logic circuits.



Q.25 Draw a block diagram, truth table and logic circuit of a 16 x 1 multiplexer and explain its working principle.

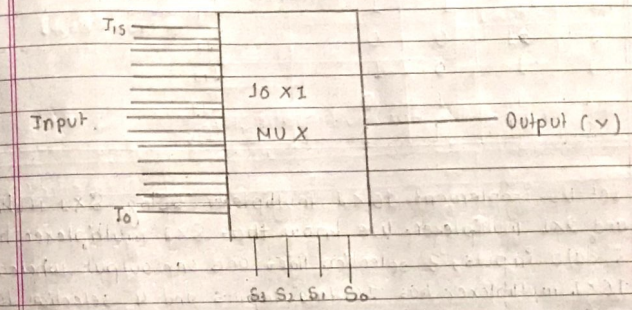
Soln: In 16 x 1 multiplexer,

Input lines, $16 = 2^4$ i.e. 4 selection lines

Input lines will be $I_0 - I_{15}$.

Selection lines will be $S_0 - S_3$.

* Block diagram:



* Truth Table

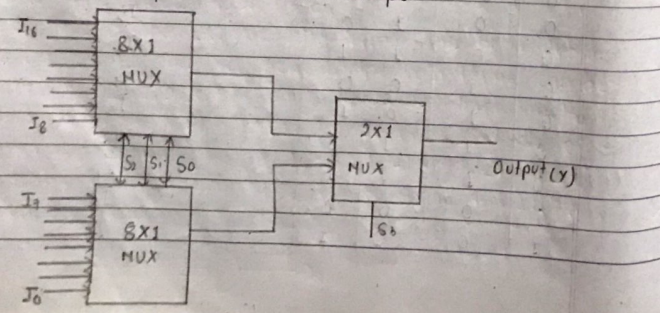
Selection inputs				Outputs
S_3	S_2	S_1	S_0	Y
0	0	0	0	I_0
0	0	0	1	I_1
0	0	1	0	I_2
0	0	1	1	I_3
0	1	0	0	I_4
0	1	0	1	I_5
0	1	1	0	I_6
0	1	1	1	I_7
1	0	0	0	I_8
1	0	0	1	I_9

Date:
Page:

Input set or Stored data

1	0	1	0	I_{10}
1	0	1	1	I_{11}
1	1	0	0	I_{12}
1	1	0	1	I_{13}
1	1	1	0	I_{14}
1	1	1	1	I_{15}

Let us implement 16x1 multiplexer using 8x1 Multiplexer and 2x1 multiplexer. We know that 8x1 multiplexer has 8 data inputs, 3 selection lines and one output whereas 16x1 multiplexer has 16 data inputs and 4 selection lines so, we require two 8x1 multiplexers in first stage in order to get the 16 data inputs. Since, each 8x1 multiplexer produces one output, we require a 2x1 multiplexer in second stage by considering the outputs of first stage as inputs and to produce the final output.



Date:
Page:

The same selection lines S_2, S_1 & S_0 are applied to both 8x1 Multiplexer. The data inputs of upper multiplexer are I_{15} to I_8 to the data inputs of lower 8x1 multiplexer are I_7 to I_0. Therefore, each 8x1 Multiplexer produces an output based on the values of selection lines, S_2, S_1 and S_0.
The outputs of first stage 8x1 MUX are applied as inputs of 2x1 MUX that is present in second stage. The other selection lines, S_3 is applied to 2x1 Multiplexer.
Rules:

(i) If S_3 is zero, then the output of 2x1 Multiplexer will be one of the 8 inputs i.e. I_7 to I_0 based on the values of selection lines S_2, S_1 and S_0.

(ii) If S_3 is one, then the output of 2x1 Multiplexer will be one of the 8 inputs i.e. I_{15} to I_8 based on the values of selection lines S_2, S_1 and S_0.

Therefore, the overall combination of two 8x1 multiplexer and one 2x1 multiplexer performs as one 16x1 multiplexer.

Q.26 Design the full subtractor circuit with using Decoder and explain the working principle:

Ans: The full subtractor is used to subtract three 1-bit numbers which are minuend, subtrahend and borrow respectively. It has so three input states and two output states, i.e. difference and
borrow. i.e.

Date:
Page:

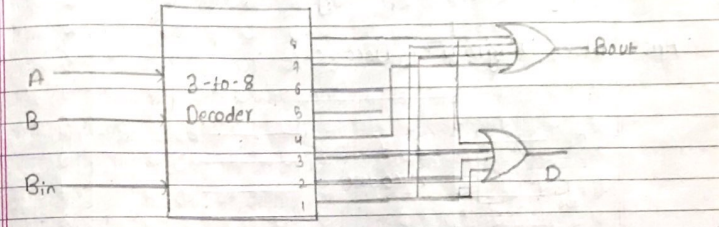
Minuend (A)	Full - Subtractor	Difference (D)
Subtrahend (B)		
Borrow (Bin)		Borrow (Bout)

The truth table for full subtractor :-

Inputs			Outputs	
A	B	Bin	Bout (Borrow)	Difference (D)
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

From the above truth table,
for the difference, the min-terms can be written as Σ (1,2,3,7) and for borrow, the min-terms can be written as Σ(1,2,4,7). Since, there are three inputs, there is a total of eight ($2^3 = 8$) min-terms and so eight outputs. So, we need 3-to-8 line decoder. The decoder generates the eight min-terms for A, B and Bin which is shown as follows:

Date:
Page:



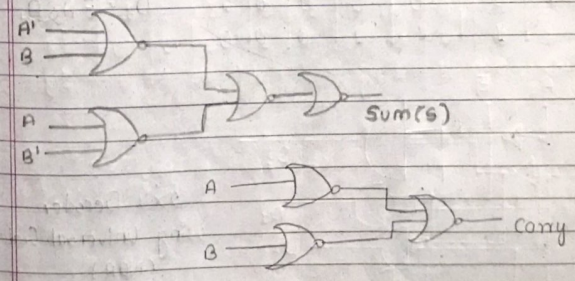
Q.24 Design a half adder logic using only NOR gate.

Ans: A half adder is a combinational logic circuit with two inputs and two outputs i.e. sum 's' and carry 'c'.
From the truth table, the respective equation for sum 's' and carry 'c' is,

$$\text{Sum (S)} = A'B + AB' = A \oplus B$$

$$\text{carry (c)} = AB$$

Using only NOR gate, we have, half adder logic circuit,



Date:
Page:

Q.27 Design a Decoder using Universal Gate.

Ans: Block Diagram of Decoder:

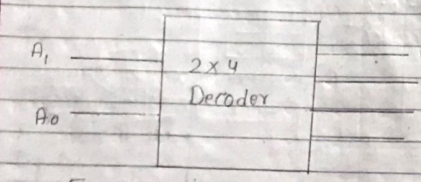


Fig: 2x4 Decoder

Truth Table:

Enable	Inputs		Outputs			
E	A_1	A_0	D_3	D_2	D_1	D_0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

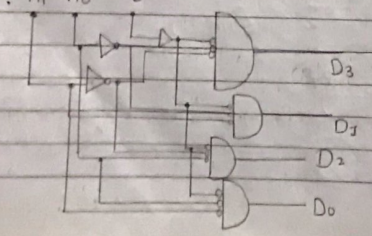
Equations:

$D_0 = EA_0'A_1'$

$D_1 = EA_0A_1'$

$D_2 = EA_0'A_1$

$D_3 = EA_0A_1$



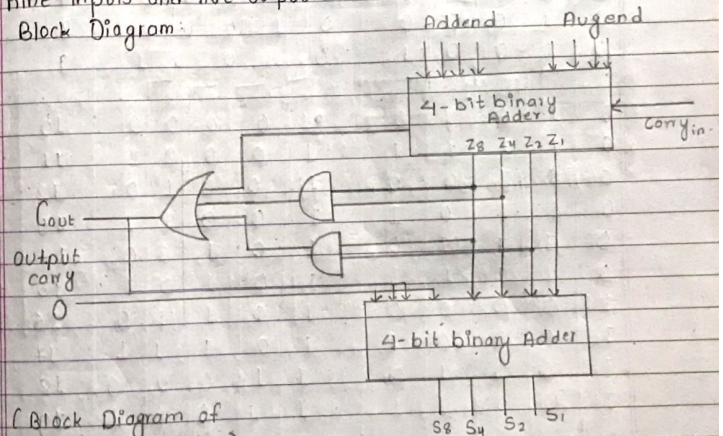
2x4 Decoder using Universal Gate (NOR)

Date:
Page:

Q.28 Design a decimal adder with logic diagram and truth table.

Ans: A Decimal-adder is used in the computers and the calculators that perform arithmetic operation directly in the decimal number system. The decimal-adder accepts the binary-coded form of decimal numbers. The decimal adder requires a minimum of nine inputs and five output.

* Block Diagram:



(Block Diagram of a BCD Adder)

* Truth Table

Binary sum					BCD sum					Decimal
K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1	
0	0	0	0	0	0	0	0	0	0	0

0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

Q. 29 Differentiate between MUX and DEMUX.

Ans: Following are the differences between MUX and DEMUX:

MUX	DEMUX
(i) It is the short form of multiplexer.	(i) It is the short form of demultiplexer.
(ii) It processes the digital information from various source into a single source.	(ii) It receives digital information from a single source and convert into several sources.
(iii) Known as data selector.	(iii) Known as data distributor.
(iv) It is a digital switch.	(iv) It is a digital circuit.
(v) It has n data inputs and single data output.	(v) It has single data input and n data output.
(vi) In time division multiplexing, it is used as transmitter ends.	(vi) In time division multiplexing, it is used as receiver ends.

Q. 30 What is magnitude comparator? Design a logic circuit for a 4-bit magnitude comparator and explain it.

Ans: A magnitude 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.

A 4-bit magnitude comparator is a combinational logic circuit that compares two binary numbers each of 4-bits. Let's consider two numbers A & B with four digits each.

i.e. $A = A_3A_2A_1A_0$

$B = B_3B_2B_1B_0$

* Verification of $(A=B)=M$

- The equality of each pair of bits can be expressed:

$x_i = A_iB_i + \bar{A_i}\bar{B_i}$; $i = 0, 1, 2, 3$

where $x_i = 1$ only if $A_i = B_i$ and $x_i = 0$ only if $A_i \neq B_i$.

- For equality condition to exist, all x_i variables must be equal to 1. A and B will be equal if $x_3x_2x_1x_0 = 1$

$\therefore (A=B) = x_3x_2x_1x_0$

* Verification of $(A>B)=N$

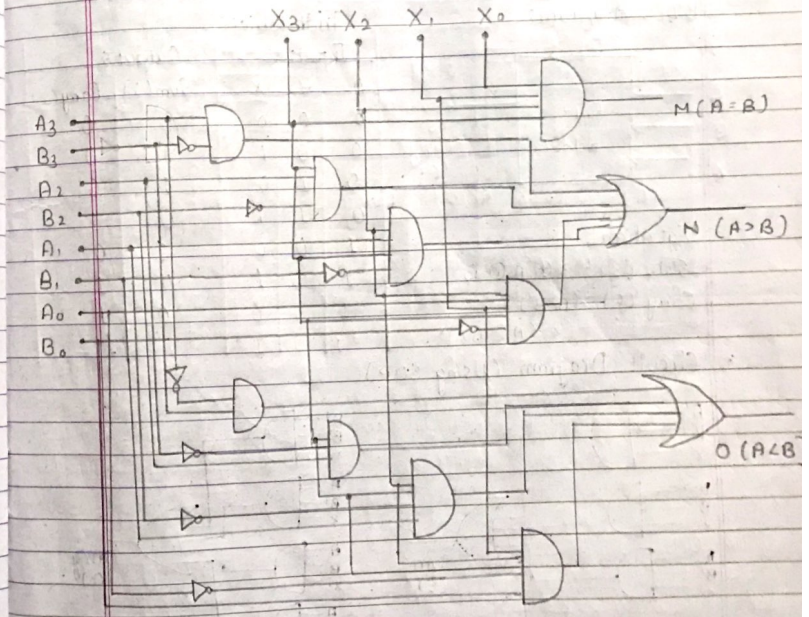
- If $A_3 > B_3$ then $A > B$, it means $A_3 = 1$ and $B_3 = 0$. Therefore, A is greater than B if $A_3\bar{B_3} = 1$
- If $A_3 = B_3$ (i.e. $x_3 = 1$) and $A_2 > B_2$ then $A > B$. Therefore, A is greater than B if $x_3A_2\bar{B_2} = 1$
- If $A_3 = B_3$ (i.e. $x_3 = 1$) and $A_2 = B_2$ (i.e. $(x_2 = 1)$ and $A_1 > B_1$ then $A > B$. Therefore, A is greater than B if $x_3x_2A_1\bar{B_1} = 1$.
- If $A_3 = B_3$ (i.e. $x_3 = 1$) and $A_2 = B_2$ (i.e. $x_2 = 1$) and $A_1 = B_1$ (i.e. $x_1 = 1$) and $A_0 > B_0$ then $A > B$. Therefore, A is greater than B if $x_3x_2x_1A_0\bar{B_0} = 1$.

$\therefore (A>B) = A_3\bar{B_3} + x_3A_2\bar{B_2} + x_3x_2A_1\bar{B_1} + x_3x_2x_1A_0\bar{B_0}$

In some manner, we can derive the expression for $(A<B)=O$

$\therefore A < B = \bar{A_3}B_3 + x_3\bar{A_2}B_2 + x_3x_2\bar{A_1}B_1 + x_3x_2x_1\bar{A_0}B_0$

Logic Diagram:



Q. 31 What do you mean by full adder and full subtractor? Explain.

Ans: A combinational circuit that performs the addition of three bits at a time is called full adder. It consists of three inputs A, B and input carry x and the outputs are carry and the sum.

Block diagram:

Truth table:

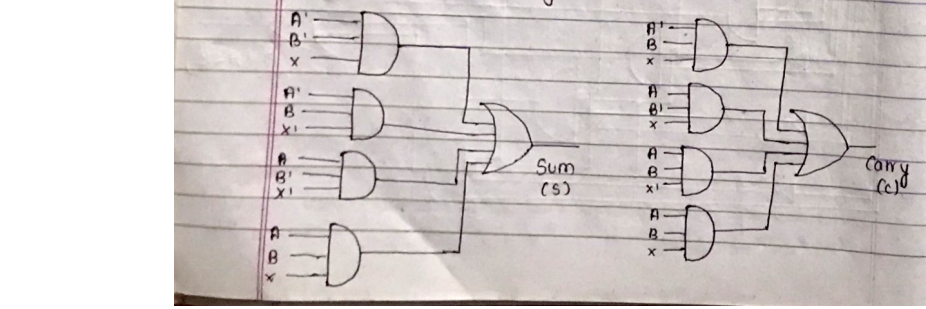
Inputs			Outputs	
A	B	X	Sum(S)	Carry(C)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Equation,

Sum (S) = $X \oplus A \oplus B$

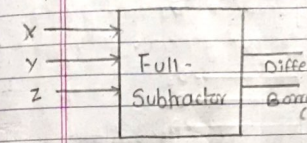
Carry (C) = $(A \cdot B) + (X \cdot (A \oplus B))$

* Circuit Diagram (Using SOP)



A combinational circuit that performs the subtraction of three 1-bit numbers X, Y, and Z which are minuend, subtrahend, and borrow is called full subtractor. It consists of three inputs and two outputs i.e difference and borrow.

* Block diagram



* Truth table:

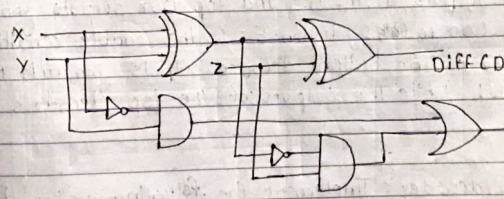
Inputs			Outputs	
			(D)	(B)
X	Y	Z	Difference	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Equations:

$D = X'Y'Z + X'YZ' + XY'Z' + XYZ = X \oplus Y \oplus Z$

$B = X'Y'Z + X'YZ' + X'YZ + XYZ = X'Y + (X \oplus Y)'Z$

* Circuit Diagram (Using SOP)



Q. 32 Design a 3-to-8 line decoder using two 2 to 4 line Decode

and Explain it.

Ans:

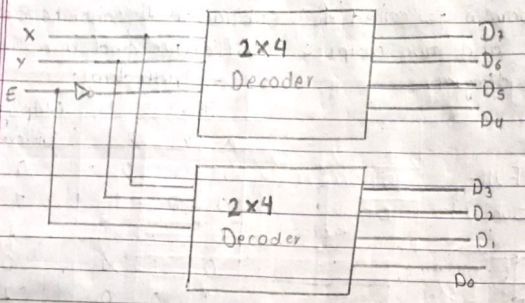


fig: 3-to-8 decoder using 2-to-4 decoder

The figure shows two 2×4 decoder with enable input (E) connected to form a 3×8 decoder. When E = 0, the top decoder is enabled and the other is disabled. The bottom decoder outputs are all 0's and top four outputs generate minterms 000 to 011. When E = 1, the enable conditions are reversed. The bottom decoder outputs generate minterms 100 to 111 while the outputs of the top decoders are all 0's.

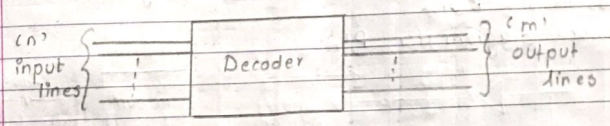
Q.33 What is decoder? Implement the following using decoder.

a, $f(wxyz) = \sum(0,1,2,4,8,9,10)$

b, $f(wxyz) = \sum(1,3,5,6,11,13,14)$.

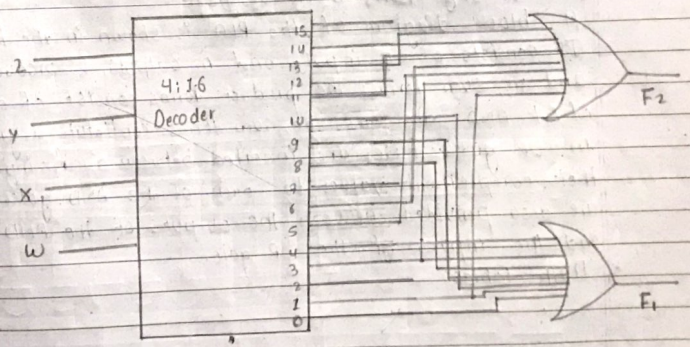
Ans: Decoder is a combinational circuit that has n input line and maximum of 2^n unique output lines. If n-bit decoded information has unused or don't care combinations, the decoder output will have less than 2^n outputs.

The decoders presented here are called n-to-m line decoders where $m \leq 2^n$. Their purpose is to generate the 2^n (or less) minterms of n input variables.



a, $F_1(wxyz) = \sum(0,1,3,4,8,9,10)$

b, $F_2(wxyz) = \sum(1,3,5,6,11,13,14)$.



Q.34 Explain PLA and PAL.

Ans:

① Programmable logic Array (PLA):

PLA is a fixed architecture logic device with programable AND gates followed by programable OR gates. PLA is similar to ROM in concept, however it doesn't provide full decoding of variable and doesnot generate all minterms as in the ROM.

* Block diagram for PLA.

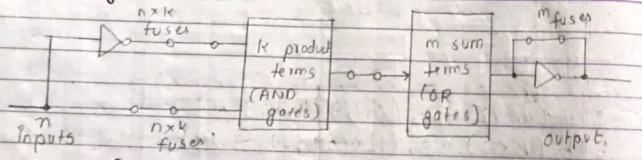


fig: PLA block diagram

A block diagram of the PLA is shown in the figure below. It consists of n inputs and m outputs, k product terms and m sum terms. The product terms consist of a group of k AND gates and the sum terms constitute a group of m OR gates. Fuses are inserted between all n-inputs and their complement values to each of the AND gates. Fuses are also provided between the outputs of the AND gates and the inputs of the OR gate.

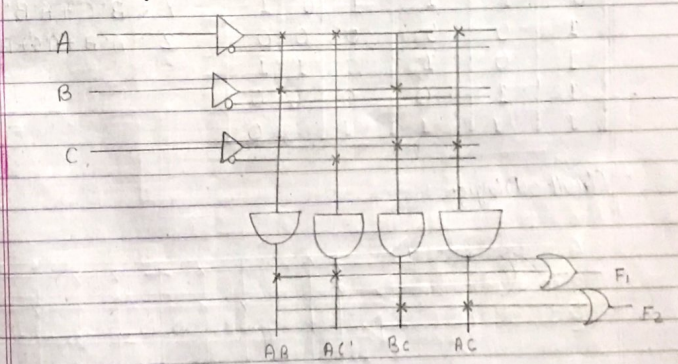
* Truth tables:

A	B	C	F_1	F_2
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

$F_1 = AB + AC'$

$F_2 = BC + AC$

* Circuit Diagram



② Programmable Array logic (PAL):

PAL is a commonly used programmable logic device (PLD). It has programmable AND array and fixed OR array.

Because only AND array is programmable, it's easier to use but not flexible as compared to PLA. PAL consists of small programmable read only memory (PROM) and additional logic output to implement a particular desired logic function with limited components.

* Truth table

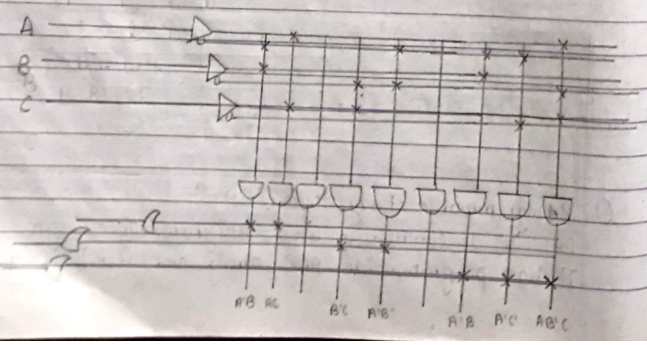
A	B	C	X	Y	Z
0	0	0	0	1	1
0	0	1	0	1	0
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	1	0	0

$X = A'B + AC$

$Y = B'C + A'B'$

$Z = A'B + AC' + AB'C$

* Circuit Diagram



Q.35 Design a multiplexer 4×1 using universal gates only.

Ans: 4×1 multiplexer has 4 data inputs I_0, I_1, I_2, I_3, two selection lines S_1, S_0 and one output Y.

From the truth table, we have,

equation = $S_1'S_0'I_0 + S_1'S_0I_1 + S_1S_0'I_2 + S_1S_0I_3$

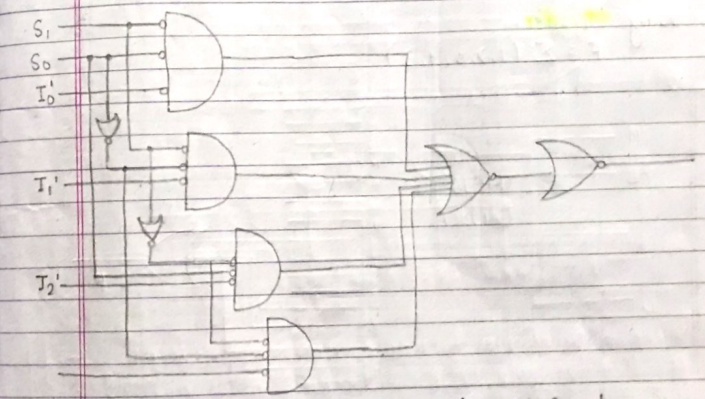


fig: 4×1 multiplexer using NOR gat

Q.36 Design a half adder logic circuit using NOR gates only.

Ans: From the truth table, the equations of half adder,

Sum (S) = $A'B + AB'$

Carry (c) = AB

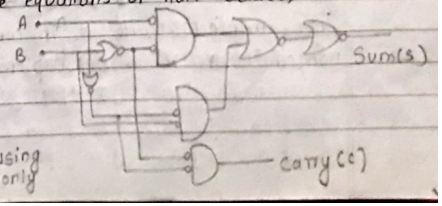
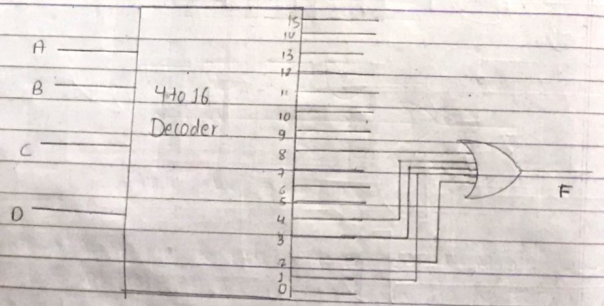


fig: Half adder using NOR gates only

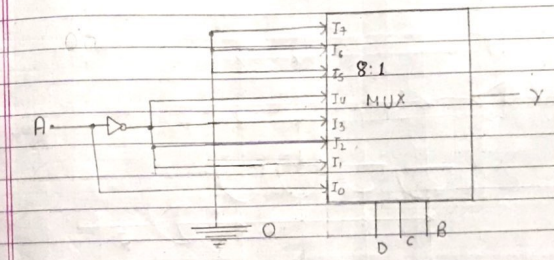
Q.37 Implement the following function.
$F = \sum(1,2,3,4,8)$ using
a, Decoder
b, Multiplexer
c, PLA

a, Using **Decoder**,
$F = \sum(1,2,3,4,8)$

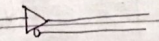


b, Using **Multiplexer**:
Take variable A for input and B, C, D for selection lines.
So, $n = 4$ or $2^{n-1} = 8$ inputs.
Thus, minterms with A in complement form are 0-7 and minterm with A in uncomplemented form are 8-15. So,

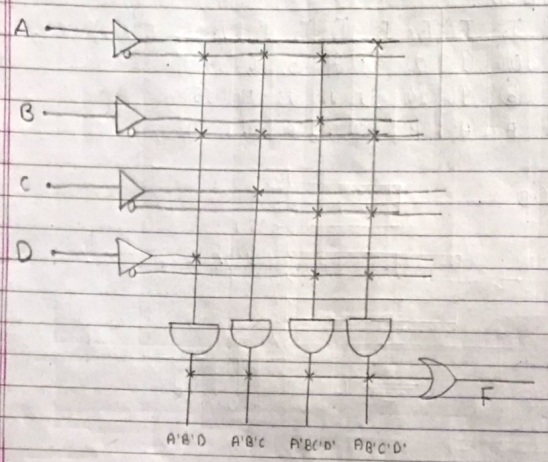
A \ BCD	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
A'	0	①	②	③	④	5	6	7
A	⑧	9	10	11	12	13	14	15
	A	A'	A'	A'	A'	0	0	0



c, **PLA**:
(M) Given function = $\sum(1,2,3,4,8)$
(i) Using k-Map, the required equation is,
$F = A'B'D + A'B'C + A'BC'D' + AB'C'D'$
(ii) Input Buffer:



(iii) No. of P. AND gate = 4 × 1 (Input max. combination × total no. of outputs)
No. of P. OR gate = 4



Q.38 Write a procedure to reduce 4 variable K-Map.

Ans: A four variable (A,B,C,D) expression can have $2^4 = 16$ combination of input variable.

A'B'C'D' ---- ABCD with minterm designation m_0, m_1, m_2, m_3 ---- m_{15} respectively.

AB \ CD	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

AB \ CD	00	01	11	10
00	A'B'C'D'	A'B'C'D	A'B'CD	A'B'CD'
01	A'BC'D'	A'BC'D	A'BCD	A'BCD'
11	ABC'D'	ABC'D	ABCD	ABCD'
10	AB'C'D'	AB'C'D	AB'CD	AB'CD'

(In SOP)

AB \ CD	00	01	11	10
00	A+B+C+D	A+B+C+D'	A+B+C'+D'	A+B+C'+D
01	A+B'+C+D	A+B'+C+D'	A+B'+C'+D'	A+B'+C'+D
11	A'+B'+C+D	A'+B'+C+D'	A'+B'+C'+D'	A'+B'+C'+D
10	A'+B+C+D	A'+B+C+D'	A'+B+C'+D'	A'+B+C'+D

(In POS)

* Rules

(i) One square represents one minterm giving a term of four variables.

(ii) Two adjacent squares represent a term of three variables.

(iii) Four adjacent squares represent a term of two variable.

(iv) Eight adjacent squares represent a term of one variable.

ASSIGNMENT- 3

Date: .....

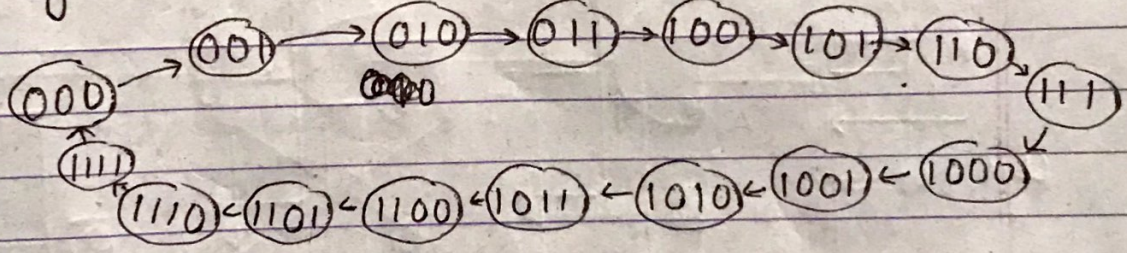
Page: .....

Q.3) Explain the 4-bit ripple counter and also draw a timing diagram.

Ans: The counter in which the external clock is only given to the first flip flop and the succeeding flip-flops are clocked by the output of the preceding flip-flip is called a ripple counter.

A 4-bit ripple counter has 16 states, due to it's 4 flip flops, i.e. $2^4 = 16$. For 16 clock pulses, the counter progress through a binary count of zero through fifteen and then recycles to the zero state. Only the first ff is externally clocked, the succeeding rest depends on the output of the first flip-flop.

State diagram of 4-bit:



* State transition table:

Clock pulse	Flip-flop State transition state.			
	Q_3	Q_2	Q_1	Q_0
0 (Initially)	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0

5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0 (Reset)

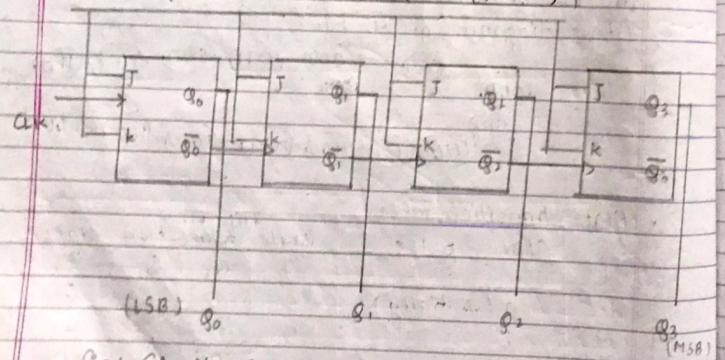


fig: Circuit diagram of 4-bit Ripple counter

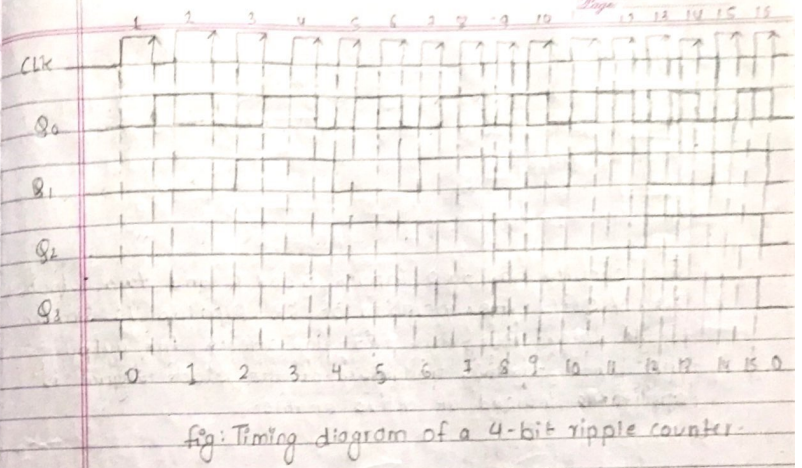
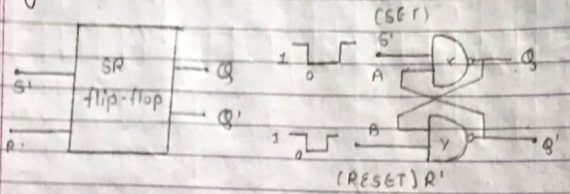


fig: Timing diagram of a 4-bit ripple counter.

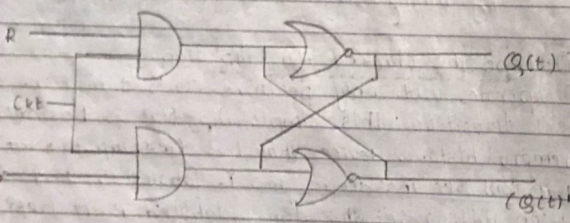
Q. 40 Describe the clocked RS flip-flop.

Ans: It operates with only positive clock transitions or negative clock transitions. This simple flip-flop is basically a one-bit memory bistable device that has two inputs, ~~the~~ one which ~~the~~ will set the device and is labeled S and the other which will reset the device, labeled R. A basic SR flip flop circuit provides feedback from both of it's outputs back to it's opposing inputs and is commonly used in memory circuits to store a single data bit. Then, the SR flip-flop actually has three inputs, Set, Reset and it's current output Q relating to it's current state or ~~elist~~ history. The circuit diagram of basic SR flip-flop is shown in th

figure below:



The circuits has two inputs S and R and two output Q(t) & Q(t'). The operation of SR flipflop is similar to SR latch. But, this flip-flop affects the output only when positive transition of the clock signal is applied instead of active enable.



* The characteristic table of RS flip flop:

Present inputs		Present state	Next state
S	R	Q(t)	Q(t+1)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	x
1	1	1	x

* For Equation,
  Using k-Map, $Q(t+1) = S + R'Q(t)$

* For state table,

S	R	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	x

Q.41. What do you mean by triggering of flip-flop?

Ans: The output of a flip-flop can be changed by bring a small change in the input signal. This small change can be brought with the help of a clock pulse or commonly known as a trigger pulse. When such a trigger pulse is applied to the input, the output changes and thus the flip-flop is said to be triggered. There are two types of triggering of flip-flop which are as follows:

1. level triggering
2. Edge triggering

1. Level Triggering:
In the level triggering, the output state is allowed to change according to input(s) when active level is maintained at the enable input. There are two types of level triggered latches:

a. Positive level triggered:
The output of latch responds to the input change only when its enable input is 1 (HIGH).

b. Negative level triggered:
The output of latch responds to the input changes only when its enable input is 0 (LOW).

2. Edge Triggering:
In the edge triggering, the output responds to the changes in the input only at the positive or negative edge of the clock pulse at the clock input. There are two type of edge triggering:

a. Positive edge triggering:
Here, the output responds to the changes in the input only at the positive edge of the clock pulse at the clock input.

b. Negative edge triggering:
Here, the output respond to the changes in the inputs only at the negative edge of the clock pulse at the clock input.

Q.42 What are the various types of Shift Register operations?

Ans: A shift register is a type of digital circuit using a cascade of flip-flops where the output of one flip-flop is connected to the input of the next. They share a single clock signal, which causes the data stored in the system to shift from one location to the next.
Shift register IC's are generally provided with a clear or reset connection so that they can be "SET" or "RESET" as required. Generally, shift registers operate in one of four different modes with the basic movement of data through a shift register being:

(i) Serial-In-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.

(ii) Serial-In-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.

(iii) Parallel-In-Parallel-Out (PIPO):
The parallel data is loaded simultaneously into the register, and transfered together to their respective outputs by the same clock pulse

(iv) Parallel-In-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.

Q.43 Describe the Ripple Counter.

Ans: The counter in which external clock is only given to the first Flip-Flop and the succeeding Flip-flops are clocked by the output of the preceding flip-flop is called asynchronous counter or ripple counter. The name ripple counter is because the clock signal ripples it's way from the first stage of Flop-flop to be last stage.

In the asynchronous counter, the clock signal is only used to clock the first FF. Each FF (except the first FF) is clocked by the preceding FF. Different clock is applied to different flip flop. It is slower relatively to synchronous counter.

* Features of Asynchronous Counter:

(i) Another name for Asynchronous counters is "Ripple counters"

(ii) These are very simple in design. As its design is simple, they use less number of logic gates to construct an asynchronous counter.

* Types of Ripple counters:

It has many types. Some of them are mentioned as follows:

(i) Up Counter
(ii) Down Counter
(iii) Ripple Up/Down Counter
(iv) Ripple BCD Counter.

Q.46 What are the shift register operations?

Ans: 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 it's inputs and then moves or "shifts" it to it's output once every clock cycle, hence the name shift register. A shift register is basically contains several single bit "D-type Data latches" one for each data bit, either a logic "0" Or a "1" connected together in a serial type daisy-chain arrangement so that the output from one data latch becomes the input of the next latch and so on.

Shift registers are used for data storage or for the movement of data and are therefore commonly used inside calculators or computers to store data such as two binary numbers before they are added together, or to convert the data from either a serial to parallel or parallel to serial or same to same format. The individual data latches that make up a single shift register are all driven by a common clock (Clk) signal making them synchronous devices.

Q.44 Design the 4 bit Synchronous up/down counter with timing diagram and truth table.

Ans: Here,

Step 1: Number of flip flop = 4

i.e. $2^n = 2^4 = 16$

Maximum number of count = 16 - 1 = 15

Step 2: We use J-k flip-flop to design the counter.

Step 3: Excitation table for J-k flip-flop.

Q_n	Q_{n+1}	J	k
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Step 4: Transition Table

~~Present state — Next state — J-k flip-flop as input~~

~~A B — A+ B+ — J_3 k_3 J_2 k_2 J_1 k_1 J_0 k_0~~

Present state				Next stage				J-k flip-flop as input							
A	B	C	D	A_t	B_t	C_t	D_t	J_A	k_A	J_B	k_B	J_C	k_C	J_D	k_D
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	X	0	0	X	0	X	1	X
1	0	0	1	1	0	1	0	X	0	0	X	1	X	X	1
1	0	1	0	1	0	1	1	X	0	0	X	X	0	1	X
1	0	1	1	1	1	0	0	X	0	1	X	X	1	X	1
1	1	0	0	1	1	0	1	X	0	X	0	0	X	1	X
1	1	0	1	1	1	1	0	X	0	X	0	1	X	X	1
1	1	1	0	1	1	1	1	X	0	X	0	X	0	1	X
1	1	1	1	0	0	0	0	X	1	X	1	X	1	X	1

Simplified Boolean Expressions.

$J_A = BC*D$ $J_B = CD$ $J_C = D$ $J_D = 1$

$K_A = BCD$ $K_B = CD$ $K_C = D$ $K_D = 1$

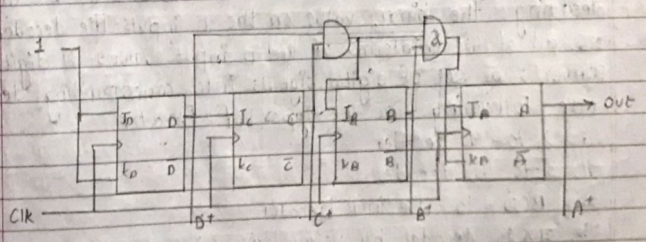
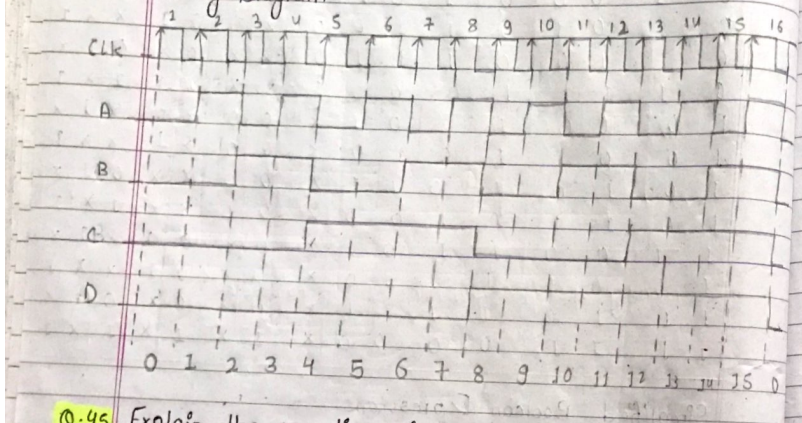


fig: 4-bit synchronous up counter

* Timing Diagram

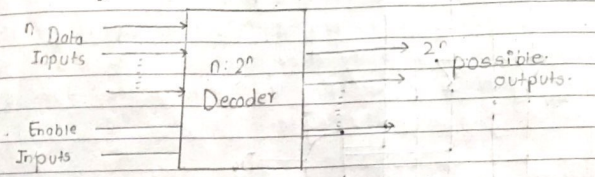


Q.45 Explain the operation of Decoder.

Ans: Decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2^n unique output lines. A decoder selects one of 2^n outputs by decoding the binary value on the n inputs. The decoder generates all of the minterms of the n input values. A digital decoder converts a set of digital signals into corresponding decimal code. Following are the types of Decoder

i. 2-to-4 decoder
ii. 3-to-8 decoder
iii. BCD-to-Decimal Decoder
iv. 4x16 decoder by using two 3x8 decoder.

* Block Diagram of a decoder



To see how the decoder works, let's explain the 2-to-4 decoder.

2-to-4 line decoder contains two inputs X_0, X_1 and four outputs represented by D_0, D_1, D_2 and D_3. The truth table for 2-to-4 decoder is as shown below. For each input combination, one input line is activated.

Truth table for 2-to-4 decoder

Enable	Inputs		Outputs			
E	X_1	X_0	D_3	D_2	D_1	D_0
0	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

The circuit diagram is implemented as shown below:

Equations: $D_0 = E X_1' X_0'$

$D_1 = E X_1' X_0$ $\quad D_3 = E X_1 X_0$

$D_2 = E X_1 X_0'$

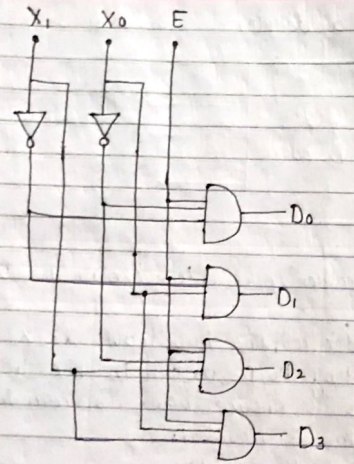


fig: 2-to-4 decoder

Similarly, the other decoders ~~have~~ operates the same way as the above 2-to-4 decoder.

Q. 47 What do you mean by synchronous counter?

Ans: The counters which use clock signal to change their transitions are called "Synchronous Counter". This means the synchronous counters depends on their clock input to change state values. Unlike the asynchronous counter, synchronous counter has one global ~~clock~~ Clock, which drives each flip flop so output changes in parallel. The one advantage of synchronous counter over asynchronous counter is, it can operate on higher frequency than asynchronous counter as it does not have cumulative delay because of same clock is given to each flip-flop.

Following are the features of Synchronous Counter:

(i) Their construction is very simple in design.
(ii) The state of output of the previous ff determines the state change of the present flip-flop.
(iii) As all the flip flops will work synchronously, the synchronous counters don't require settling.
(iv) We require number of logic gates to implement the synchronous counters.
(v) Their operation is fast.

Q. 48 What is Jk master slave flip-flop? Design its logic circuit, truth table and explain the working principle.

Ans: The Master-Slave Flip-flop is basically a combination of two Jk flip-flops connected together in a series configuration. Here, one acts as the 'master' and the other as a 'slave'. The output from the master flip flop is connected to the two inputs of the slave flip flop whose output is fed back to inputs of the master flip flop.

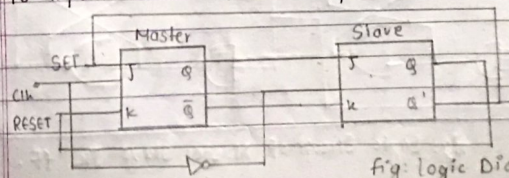


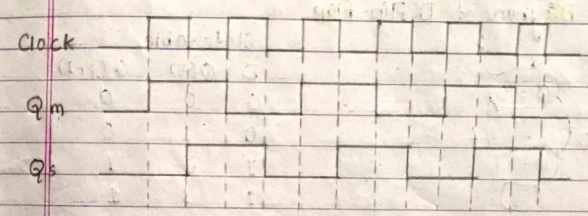
fig: logic Diagram of Master Slave Jk Flip-flop

In addition to these two flip-flops, the circuit also includes an inverter. The inverter is connected to clock pulse in such a way that the inverted clock pulse is given to the slave flip flop. In other words, if CP=0 for a master flip-flop, then CP=1 for a slave flip-flop and if CP=1 for master flip-flop, then it becomes 0 for slave flip-flop.

Working of a master slave flip-flop:

1. When the clock pulse goes to 1, the slave is isolated; J and k inputs may affect the state of the system. The slave flip-flop is isolated until the clock pulse (CP) goes to 0. When the CP goes to 0, information is passed from the master flip-flop to the slave and output is obtained.
2. Firstly, the master flip flop is positively level triggered and the slave flip flop is negative level triggered, so the master responds before the slave.
3. If J=0 and k=1, the high Q' output of the master goes to the k input of the slave and the clock forces the slave to reset, thus the slave copies the master.
4. If J=1 and k=0, the high Q output of the master goes to the J input of the slave and the Negative transition of the clock sets the slave, copying the master.
5. If J=1 and k=1, it toggles on the positive transition of the clock and thus the slave toggles on the negative transition of the clock.
6. If J=0 and k=0, the flip flop is disabled and Q remains unchanged.

Timing diagram of a Master flip-flop:



This makes the Master Slave J-K Flip Flop a Synchronous device as it only passes data with the timing of the clock signal.

1. When the clock pulse is high, the output of master is high and remains high till the clock is low because the state is stored.
2. Now, the output of master becomes low when the clock pulse became high again and remains low until the clock becomes high again.
3. Thus, toggling takes place for a clock cycle.
4. When the output pulse is high, the master is operational but

not the slave thus the output of the slave remains low till the clock remains high.

5. When the clock is low, the slave becomes operational and remains high until the clock again becomes low.
6. Toggling takes place during the whole process since the output is changing once in a cycle.

* Truth table for Master Slave Jk Flip-Flop

S(J)	R(K)	Qn	Qn+1
Present	stateinputs	Present state	Next state
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

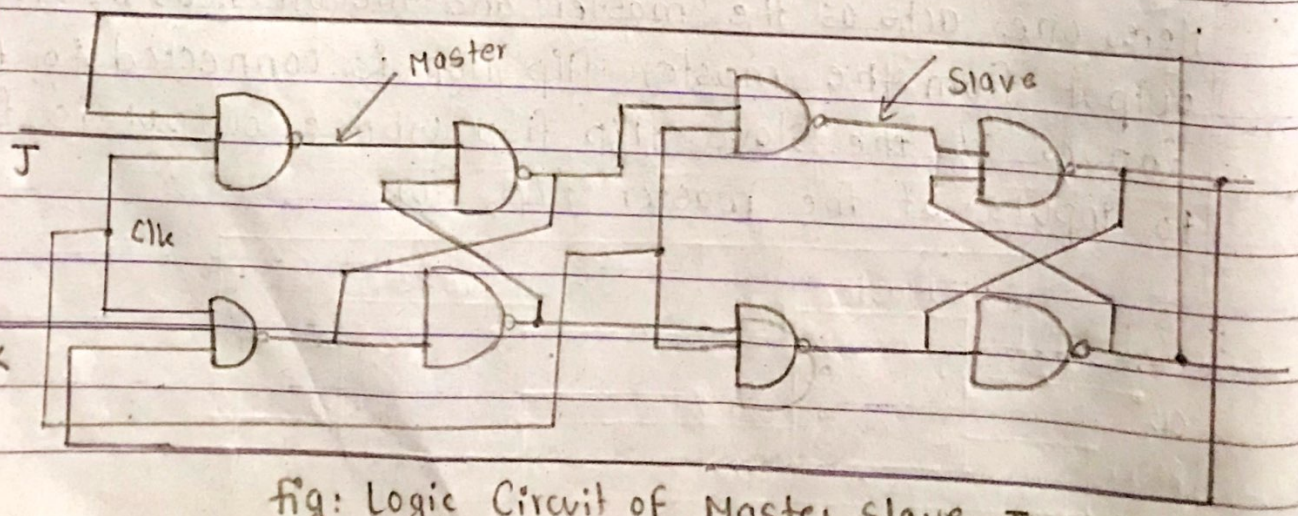
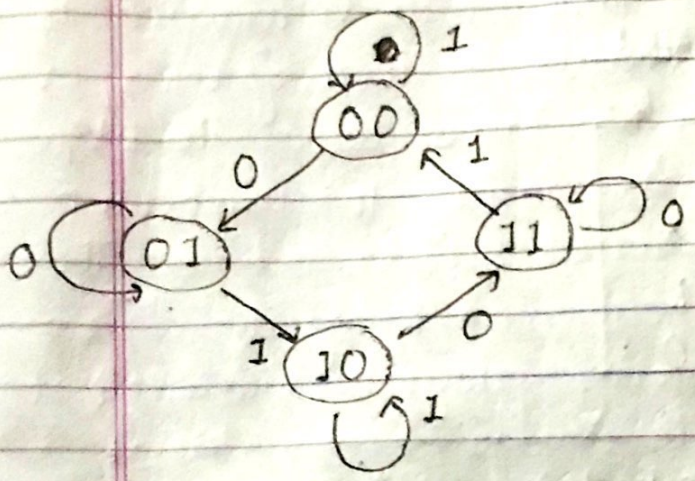


fig: Logic Circuit of Master Slave Jk ff.

Date: .....
Page: .....

Q.49 Design a clocked sequential circuit of the following state diagram by using JK flip flop.



Ans: Here,

State table for the above state diagram:

Present State			Next State		Output	Output
Q_A	Q_B	X	Q_A^+	Q_B^+	Y	Y
0	0		0	0	0	
0	0		0	1	0	

Present state		Next state				Output	
		x = 0		x = 1		x = 0	x = 1
Q_A	Q_B	Q_A^+	Q_B^+	Q_A^+	Q_B^+		
0	0	0	1	0	0	-	-
0	1	0	1	1	0	-	-
1	0	1	1	1	0	-	-
1	1	1	1	0	0	-	-

* Excitation table, for JK flipflop,

Present state Next state

Q_A Q_A x Q_A^+ Q_A^+

Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

* Characteristic Table,

Present state		Input	Next state		Flip flops			
Q_A	Q_B	X	Q_A^+	Q_B^+	J_A	K_A	J_B	K_B
0	0	0	0	1	0	X	1	X
0	0	1	0	0	0	X	0	X
0	1	0	0	1	0	X	X	0
0	1	1	1	0	1	X	X	1
1	0	0	1	1	X	0	1	X
1	0	1	1	0	X	0	0	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1

(Using K-Map,)

Equations,

$J_A = Q_B X$

$K_A = Q_B X$

$J_B = X$

$K_B = X$

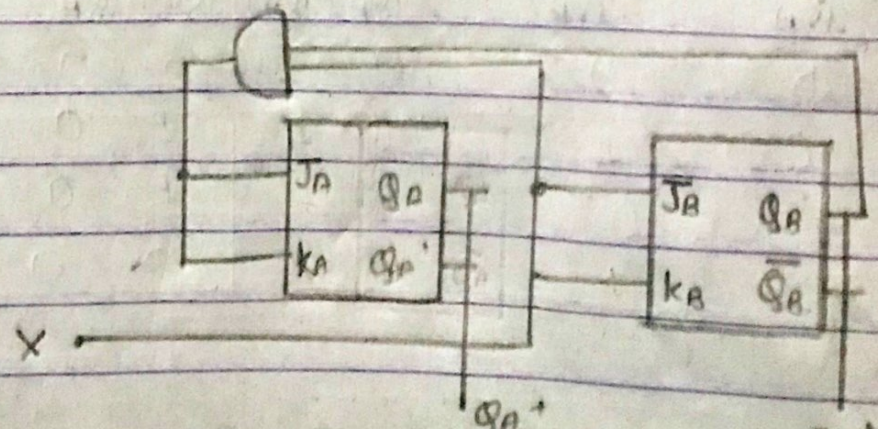


fig: Circuit diagram

Date: .....
Page: .....

Q.50 Write short notes on:

(i) Programming logic array (PLA)
It is a fixed architecture logic device with programmable AND gates followed by programmable OR gates. It is basically a type of programmable logic device used to build a reconfigurable digit circuit. PLDs have an undefined function at the time of manufacturing, but they are programmed before made into use. PLA is a combination of memory and logic. PLA is similar to a ROM in concept, however it doesn't provide full decoding of variables and doesn't generate all minterms as in the ROM. The "programmable" in PLA doesn't require ~~only~~ + any type of programming like in C and C++.

(ii) Triggering at flip-flop:
The output of a flip flop can be changed by bringing a small change in the input signal. This small change can be brought with the help of a clock pulse or commonly known as a trigger pulse. When such a trigger pulse is applied to the input, the output changes and thus the flip flop is said to be triggered. There are two types of triggering which is also classified into two types,

i, Edge triggering
- Positive edge triggering
- Negative edge triggering

ii, level triggering
- Positive level triggered
- Negative level triggered.

Date: .....
Page: .....

(iii) Memory unit:

A memory unit is a device to which binary information is transfered for storage and from which information is retrieved when needed for processing. A memory unit stores binary information in groups of bits called words. The internal structure of memory unit is specified by the number of words it contains and the number of bits in each word. The memory unit is an essential component in any digital computer since it is needed for storing programs and data. Not all accumulated information is needed by the CPU at the same time. Therefore, it is more economical to use low-cost storage devices to serve as a backup for storing the information that is not currently used by CPU. ROM is a primary memory unit in any computer system, along with RAM, but unlike RAM, the binary information is stored permanently in ROM.