DOWNLOADED FROM STUCOR APP

PART A

1. Determine $(377)_{10}$ in Octal and Hexa-Decimal equivalent. *[Nov'14]*

8	377	
8	47	1
	5	7

$(377)_{10} = (571)_8$

16	377	
16	23	9
	1	7

$(377)_{10} = (179)_{16}$

2. Convert $(115)_{10}$ *and* $(235)_{10}$ to hexadecimal numbers. *(Nov'17)*

115_{10}

16	115	
	7	3

$115_{10} - 73_{16}$

235_{10}

16	235	
	14	11

$235_{10} - EB_{16}$

3. Convert:

a) $(475.25)_8$ to its decimal equivalent.

b) $(549.B4)_{16}$ to its binary equivalent. *[Apr'15]*

a). $(475.25)_8$

$$=4 \times 8^2 + 7 \times 8^1 + 5 \times 8^0 + 2 \times 8^{-1} + 5 \times 8^{-2}$$
$$= 256+56+5+0.25+0.078125$$
$$= [317.32825]_{10}$$

b). $(549.B4)_{16}$

5 4 9 . B 4

0101 0100 1001 . 1011 0100

$= [10101001001.10110100]_2$

4. Convert each of the following decimal numbers to excess – 3 code. i). [18]10 ii). [56]10

i) $[18]_{10}$

1	8
0001	1000
+ 0011	+ 0011
0100	1011

$[18]_{10} = (0100\ 1011)$

ii) $[56]_{10}$

5	6
0101	0110
+ 0011	+ 0011
1000	1001

$[56]_{10} = (1000\ 1001)$

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

5. Convert 143_{10} into its binary and binary coded decimal equivalent. *[May'17]*

Decimal to binary:

2	143	
2	71	1
2	35	1
2	17	1
2	8	1
2	4	0
2	2	0
	1	0

$143_{10} \longrightarrow 1001111_2$

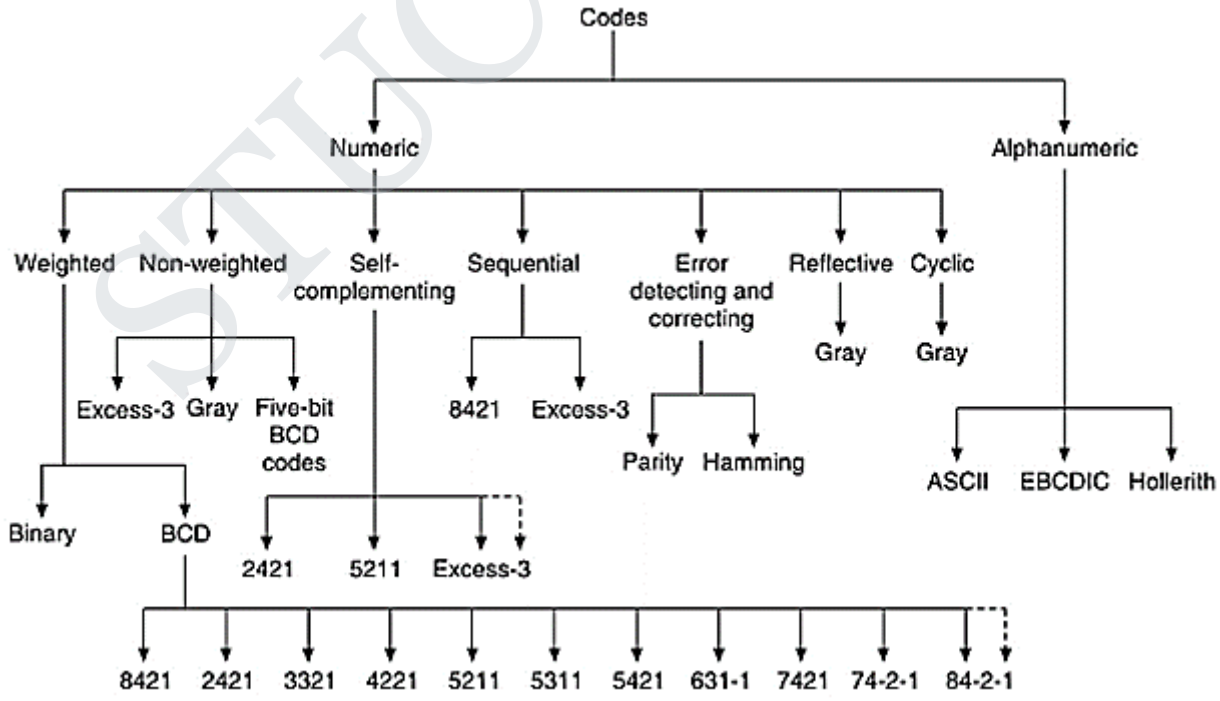
Decimal to BCD:

1 4 3

0001 0100 0011

$143_{10} \longrightarrow$ 0001 0100 0011

6. Give the classification of binary codes.



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

7. What are weighted codes? Give example.

- In weighted codes, each digit is assigned a specific weight according to its position.
- For example, in 8421 BCD code, 1001 the weights of 1, 0, 0, 1 (from left to right) are 8, 4, 2 and 1 respectively.
- The codes 8421 BCD, 2421 BCD, 5211 BCD are all weighted codes.

8. What are non-weighted codes? Give example. *[Nov'13]*

- The non-weighted codes are not positional weighted.
- In other words, each digit position within the number is not assigned a fixed value (or weight).
- Excess-3 and gray code are non-weighted codes.

9. What is a unit distance code? Give an example. *[Nov'15]*

- Unit distance code is a non – weighted code in which next increment or decrement causes the bit – transition only at one place.
- Example: Gray Code.

10. What are reflective codes? Give example.

- A code is reflective when the code is self-complementing.
- In other words, when the code for 9 is the complement of 0, 8 for 1, 7 for 2, 6 for 3 and 5 for 4.
- 2421BCD, 5421BCD, 5211BCD and Excess-3 code are reflective codes.

11. What are sequential codes? Give example.

- In sequential codes, each succeeding code is one binary number greater than its preceding code.
- This property helps in manipulation of data.
- 8421 BCD and Excess-3 are sequential codes.

12. What are alphanumeric codes? Give example.

- Codes used to represent numbers, alphabetic characters, symbols and various instructions necessary for conveying intelligible information.
- ASCII, EBCDIC, UNICODE are the most-commonly used alphanumeric codes.

13. What are Error detecting and correcting codes? Give example.

- Codes which allow error detection and correction are called error detecting and correcting codes.
- Hamming code is the mostly commonly used error detecting and correcting code.

14. What is Error detection and Error correction?

- Error detection is the ability to detect errors.
- Error correction has an additional feature that enables identification and correction of the errors.
- Error detection always precedes error correction.
- Both can be achieved by having extra/redundant/check bits in addition to data deduce that there is anerror.

15. What is self-Complement Codes?

- Self-complement codes are whose arithmetic and logic complements are same.
- Example: 2421, 631-1.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

16. What is Excess-3 code?

- Excess-3, also called XS3, is a non-weighted code used to express decimal numbers.
- It is particularly significant for arithmetic operations as it overcomes the shortcomings encountered while using the 8421 BCD code to add two decimal digits whose sum exceeds 9.
- This code is used in some old computers.

17. What is Gray code or cyclic code? [OR] What is a gray code and mention its advantages. *(Nov'17)*

- Gray code is a type of unit distance code.
- In this code bit patterns for two consecutive number differ in only one bit position. This property makes it very useful in Karnaugh Map.

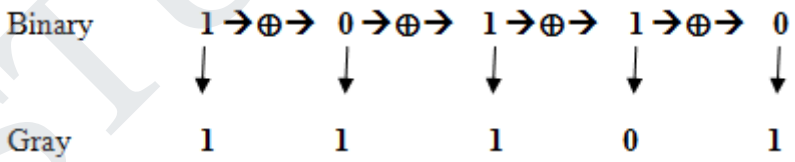
Advantages:

- In Gray code, if we go from one decimal number to next, only one bit of the gray code changes.
- Because of this feature, an amount of switching is minimized and the reliability of the switching systems is improved.

18. What is Binary Coded Decimal (BCD)?

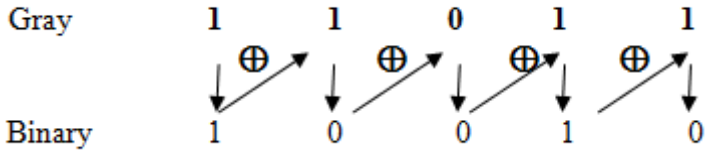
- The binary coded decimal (BCD) is a type of binary code used to represent a given decimal number in an equivalent binary form.
- Its main advantage is that it allows easy conversion to decimal digits for printing or display and faster calculations.
- The most common BCD code is the 8421 BCD code.
- In this, the BCD equivalent of a decimal number is written by replacing each decimal digit in integer and fractional parts with its four-bit binary equivalent (or nibble).
- Here 8, 4, 2 and 1 represent the weights of different bits in the four-bit groups, starting from the (MSB) most significant bit (to extreme left) and proceeding towards the least significant (LSB) bit.

19. Convert binary number 10110 to gray code.



The gray code is 11101

20. Convert gray code word 11011 to binary.



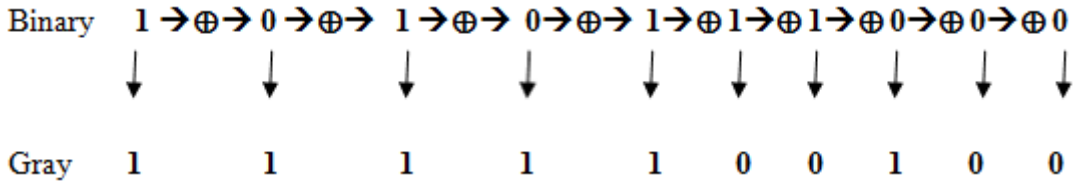
The binary number is 10010_2

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

21. Convert the following binary code into a Gray code, 1010111000_2. *[May'16]*

Solution:

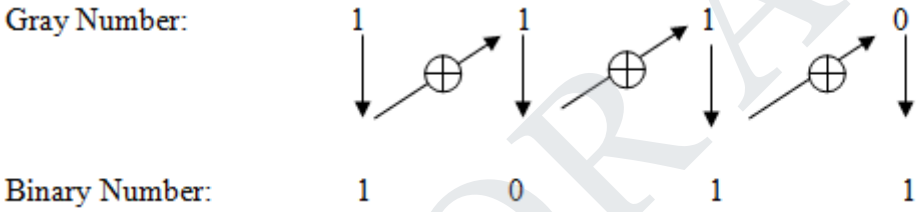


The gray code is 1111100100.

22. Give the binary, BCD, Excess–3, gray code representation of decimal numbers: 5, 8, 14.

Decimal Numbers	Binary Code	BCD Code	Excess – 3	Gray code
5	0101	0000 0101	1000	0111
8	1000	0000 1000	1011	1100
14	1110	0001 0100	0100 0111	1001

23. Convert gray number 1110 to its BCD equivalent



Therefore the it's equivalent Binary number is 10112

For BCD: $1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 8 + 0 + 2 + 1 = 1110$

1110 = 0001 0001 (BCD)

24. Convert the following Excess-3 numbers into decimal numbers. *[Nov'16]*

(a)1011

Step 1: Convert excess -3 into binary

1011
-0011
1000

Step 2: Binary to decimal

1000

8

Answer: $1011 = 8_{10}$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

(b) 1001 0011 0111

Step 1: Convert excess -3 into binary

```
  1001  0011  0111
- 0011  0011  0011
  0110  0000  0100
```

Step 2: Binary to decimal

```
 0110  0000  0100
  6     0     4
```

Answer: $1001\ 0011\ 0111 = 604_{10}$

25. What is Parity codes?

- Some of the codes are used for detecting errors while transferring binary data from one digital system to other.
- The data transferred from transmitter and the data received by the receiver should match.
- Parity checking is a detection of error detecting code.
- Parity is counting the number of ones in a binary data before the transmission.

26. What is even parity?

- An error-checking system that requires a binary number to have an even number of 1's. Generating even parity bit is just an XOR function.

Data Received Examples:

0111 1111 – incorrect
1000 0000 - incorrect
1000 0001 - valid

27. What is odd parity?

- An error-checking system that requires a binary number to have an odd number of 1's.
- Odd parity is generated using a XNOR function.
- Example:

000 0001 →**0**correct
000 0011 →**1**incorrect

28. What is hamming code?

- Sender adds the parity bit to existing data bits before transmission and the Receiver checks for the expected parity.
- If wrong parity is found, the received data is discarded and retransmission is requested.
- Hamming code is a code to detect error and also to correct it.
- It detects a single bit error and also identifies the bit that is in error.
- This code uses a number of parity bits located at certain positions in the code group.

29. List the characteristics of digital ICs.

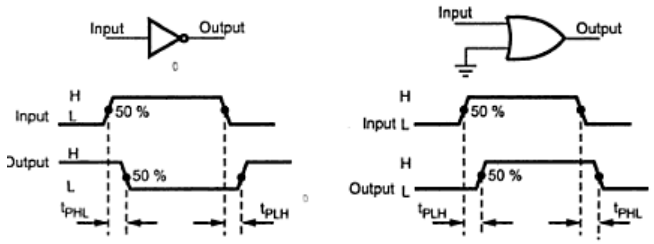
- Propagation delay
- Power dissipation
- Fan- in
- Fan-out
- Noise margin

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

30. What is propagation delay? *[Apr'15]*

- It is defined as the average transition time and delay time for the signal to propagate from input to output when signal change in value.
- Propagation delay times are defined as follows:
  - It is the propagation delay time in going from low level (0) to high level (1).
  - It is the propagation delay time in going from High level (1) to high level (0).



31. What is power dissipation?

- This is the power consumed by the gate which must be availed from power supply.
- Expresses inmilliwatts or nanowatts.
- Power dissipation = Supply voltage x meant current taken from that supply.

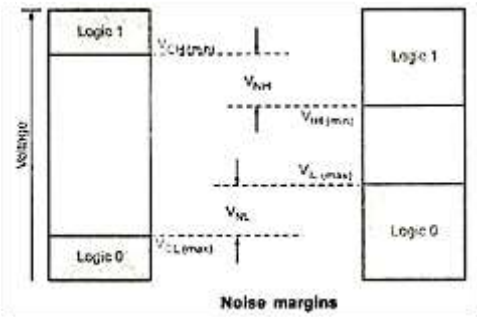
$$= V_{CC} \times I_{CC_{avg}}$$

32. Define fan in and fan out characteristics of digital logic families. *[Apr'11, Nov'15,May'16]*

- **Fan-in:** The fan-in of a gate is the number of inputs connected to the gate without any degradation in the voltage levels.
- **Fan-out**: It is defined as the maximum number of inputs of the same IC family that a gate can drive maintaining its output levels within the specified limits.

33. What is Noise margin?

- Noise margin is the maximum external noise voltage added to an input signal that does not cause an undesirable change in the circuit output.



34. Define Figure of Merit (SPP).

- Figure of merit is defined as the product of speed and power.
- The speed is specified in terms of propagation delay time expressed in nanoseconds.
- Figure of merit = Propagation delay time (ns) x Power (mW).

35. Define Noise Immunity.

- The ability of a logic circuit to tolerate the noise without causing any unwanted changes in the output.

36. Mention the characteristics of MOS transistor?

- The n- channel MOS conducts when its gate – to- source voltage is positive.
- The p- channel MOS conducts when its gate – to- source voltage is negative
- Either type of device is turned off if its gate – to- source voltage is zero.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

37. Why totem pole outputs cannot be connected together.

- Totem pole outputs cannot be connected together because such a connection might produce excessive current and may result in damage to the devices.

38. State advantages and disadvantages of TTL.

Advantages:

- Easily compatible with other ICs
- Low output impedance

Disadvantages:

- Wired output capability is possible only with tri-state and open collector types
- Special circuits in Circuit layout and system design are required.

39. What is speed-power product or Figure of Merit?

- It is a common means of measuring and comparing the overall performance of an IC Family
- Speed-power product=Propagation delay X Power dissipation

40. State the advantages of CMOS Technologies. *[May'13]*

- Consumes low power
- Can be operated at high voltages resulting in improved noise immunity
- Better noise margin
- High packaging density.

41. State the advantages of QM Method.

- QM Method is more effective for increased input variable.
- A digital computer can be used to simplify the given Boolean switching function.

42. State the advantages and disadvantages of Totem pole output.

Advantages

- External pull-up resistor is not required
- Operating speed is high

Disadvantages

- Output of two gates cannot be tied together

43. Compare the totem pole output with open collector output?*[Nov'14]*

- The open collector TTL gate needs an external resistor that must be connected between the collector of a pull-down transistor and the supply voltage for proper operation.
- The totem pole output is the standard output of a TTL gates and is specifically designed to reduce the propagation delay in the circuit and to provide sufficient output power for a high fan-out.

44. State the important characteristics of TTL family.*[Nov'10]*

- Supply voltage
- Temperature range
- Voltage levels
- Noise margin
- Power dissipation
- Propagation delay
- Fan-out

45. In which type of TTL gate wired AND logic is possible? *[Nov'10]*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- When the collector terminal of a transistor is kept open without any pull up transistor the arrangement is called open collector output.
- The output is taken directly from the open collector terminal of a transistor at the output.
- But, a gate with open collector will not work properly until an external resistor is connected.

46. Which IC family offers (a) low propagation delay, and (b) low power dissipation? *[Apr'10]*

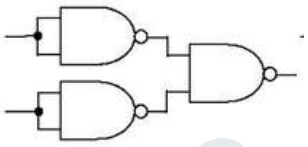
- Low propagation delay–ECL (Emitter-coupled logic)
- Low power dissipation–CMOS (Complementary MOS)

47. Why should we take care while using CMOS devices? *[Nov'11]*

- We should take care while using CMOS devices because these devices are highly prone to damage by electrostatic discharge.

48. Construct OR gate and AND gate using NAND gates.*[Nov'16]*

OR gate:



AND gate:

49. Reduce a(b+bc')+ab'.*[May'17]*

a(b+bc')+ab'= ab+abc'+ab'
= a(b+b')+abc'
= a+abc'
= a(1+bc')
= a.

50. Reduce A(A+B) *(Apr'18)*

A(A+B) = AA+AB
=A(1+B) [1+B=1]
=A

51. State the associative property of Boolean algebra. *(Apr'18)*

- The associative property of Boolean algebra states that the ORing of several variables results in the same regardless of the grouping of the variables. The associative property is stated as follows:

$$A + (B + C) = (A + B) + C$$

PART B & C

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

1. **Define number system. And explain it in detail.**

Number System:

- A numeral system (or system of numeration) is a writing system for expressing numbers, that is, a mathematical notation for representing numbers of a given set, using digits or other symbols in a consistent manner.
- The values of each digit is determined by the digit, the position of the digit in the number and the base or radix of the number system.
- The number system will be of 4 types.
    - Decimal
    - Binary
    - Octal
    - Hexadecimal

Decimal:

- Decimal is said to be Base Ten.
- When we count in Base Ten, we count starting with zero and going to nine in the order.
- It is a human language one.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

- Each additional digit represents a higher and higher quantity.
- This is applicable for base 10 as well as to other bases.

Binary:

- In a Binary number system, there are only two symbols used to represent numbers: 0 and 1.
- When we count up from zero in binary, we run out of symbols much more frequently.
- It is a circuit language one.

0, 1

Applications:

- The binary system is useful in computer science and electrical engineering.
- Transistors operate from the binary system and transistors are found in practically all electronic devices.

Octal:

- Octal is Base Eight meaning eight symbols are used to represent all the quantities.
- When we count up one from the 7, we need a new placement to represent what we call 8 since an 8 does not exist in Octal. So after 7 is 10.
- It is a machine language one.

0, 1, 2, 3, 4, 5, 6, 7

Hexadecimal:

- The hexadecimal system is Base Sixteen; this number system uses sixteen symbols to represent numbers.
- Unlike binary and octal, hexadecimal has six additional symbols that it uses beyond the conventional ones found in decimal.
- It is a machine language one.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Decimal (10)	Binary (2)	Octal (8)	Hexadecimal (16)
0	0 0 0 0	0	0
1	0 0 0 1	1	1
2	0 0 1 0	2	2
3	0 0 1 1	3	3
4	0 1 0 0	4	4
5	0 1 0 1	5	5
6	0 1 1 0	6	6
7	0 1 1 1	7	7
8	1 0 0 0	10	8
9	1 0 0 1	11	9
10	1 0 1 0	12	A
11	1 0 1 1	13	B
12	1 1 0 0	14	C
13	1 1 0 1	15	D
14	1 1 1 0	16	E
15	1 1 1 1	17	F

2. Convert the following decimal to binary/ Octal/ Hexadecimal.

i). 267_{10} ii).14768_{10} iii). 34.45_{10}

i). $(267)_{10}$

Binary:

2	267	
2	133	1
2	66	1
2	33	0
2	16	1
2	8	0
2	4	0
2	2	0
	1	

Octal:

8	267	
8	33	3
	4	1

Hexadecimal:

16	267	
16	16	11
	1	0

$(267)_{10} = (10001011)_2 (267)_{10} = (413)_8 (267)_{10} = 1011 = (10B)_{16}$

ii). $(14768)_{10}$

Binary:

2	14768	
2	7384	0
2	3692	0
2	1846	0
2	923	0
2	451	1
2	23	1
2	11	1
2	5	1
2	2	1
	1	

Octal:

8	14768	
8	1846	0
8	23	6
	2	7

Hexadecimal:

16	14768	
16	923	0
16	57	B
	3	9

$(14768)_{10} = (1111110000)_2 (14768)_{10} = (2760)_8 (14768)_{10} = (39B0)_{16}$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

iii). $(34.45)_{10}$

Binary:

2	34	
2	17	0
2	8	1
2	4	0
2	2	0
	1	0

Octal:

8	34	
	4	2

Hexadecimal:

16	34	
	2	2

$34_{10} = 100010_2 34_{10} = 42_8 34_{10} = 22_{16}$

0.45 x 2 = 0.90	0 (LSB)	0.45 x 8 = 3.6	3	0.45 x 16 = 7.2	7
0.90 x 2 = 1.80	1	0.60 x 8 = 4.8	4	0.2 x 16 = 3.2	3
0.80 x 2 = 1.60	1	0.80 x 8 = 6.4	6	0.2 x 16 = 3.2	3
0.60 x 2 = 1.20	1	0.40 x 8 = 3.2	3		
0.20 x 2 = 0.40	0 (MSB)	0.20 x 8 = 1.6	1		

$(0.45)_{10} = (0.\ 01110)_2 (0.45)_{10} = (0.34631)_8 (0.45)_{10} = (0.733)_{16}$

$(34.45)_{10} = (100010.01110)_2 (34.45)_{10} = (42.34631)_8 (34.45)_{10} = (22.733)_{16}$

3. Convert the Binary number 11100101_2 to its decimal, octal and hexadecimal equivalent.

Decimal:

$$1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$= 128 + 64 + 32 + 0 + 0 + 4 + 0 + 1$

$= [229]_{10}$

Octal: 011 100 101

3 4 5 $= [345]_8$

Hexadecimal: 1110 0101

E 5 $= [E5]_{16}$

4. Convert the Octal number 2436_8 to its decimal, binary and hexadecimal equivalent.

Decimal: $2 \times 8^3 + 4 \times 8^2 + 3 \times 8^1 + 6 \times 8^0$

$= 1024 + 256 + 24 + 6$

$= [1310]_{10}$

Binary: 2 4 3 6

010 100 011 110

$= [10100011110]_2$

Hexadecimal: 2 4 3 6

010 100 011 110

$= 0101$ 0001 1110

5 1 E $= [51E]_{16}$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

5. Convert the hexadecimal number $28D_{16}$ to its decimal, binary and octal equivalent.

Decimal: 2 8 D

= 2 8 13

$= 2 \times 16^2 + 8 \times 16^1 + 13 \times 16^0$

= 512 + 128+ 13

$= [653]_{10}$

Binary: 2 8 D

= 2 8 13

= 0010 1000 1101

$= [1010001101]_2$

Octal: 2 8 D

= 2 8 13

= 0010 1000 1101

= 001 010 001 101

1 2 1 5

$= [1215]_8$

6. Convert $FACE_{16}$ into its binary, octal and decimal equivalent. *[May'17]*

(i) Hexadecimal to binary:

$FACE_{16}$ = F A C E

15 10 12 14

1111 1010 1100 1110

$= (1111101011001110)_2$

(ii) Hexadecimal to octal:

$FACE_{16}$ = F A C E

15 10 12 14

1111 1010 1100 1110

001 111 101 011 001 110

1 7 5 3 1 6

$= 17\ 5\ 316_8$

(iii) Hexadecimal to decimal

$FACE_{16}$ = F A C E

15 10 12 14

1111 1010 1100 1110

$= 1111101011001110_2$

$= 1x2^{15}+1x2^{14}+1x2^{13}+1x2^{12}+1x2^{11}+0x2^{10}+1x2^9+0x2^8+1x2^7+1x2^6+0x2^5+0x2^4+1x2^3 +1x2^2+1x2^1+0x2^0$

$= 64206_{10}$

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

7. Convert 23.625_{10} to octal (base 8). *[Nov'16]*

8 | 23
2 7

$23_{10} = 27_8$

0.625 x 8 = 5.0 5 (LSB)
5.0 x 8 = 40.0 40 (MSB)

$(0.625)_{10} = (0.540)_8$
$(23.625)_{10} = (27.540)_8$

8. Convert 1010111011101100_2 into its octal, decimal and hexadecimal equivalent. *[Apr'16]*

(i) Binary to octal:

$1\ 010\ 111\ 011\ 101\ 100_2$
001 010 111 011 101 100
1 2 7 3 5 4

$=127354_8$

(ii) Binary to decimal:

1010111011101100_2
$= 1x2^{15} + 0x2^{14} + 1x2^{13} + 0x2^{12} + 1x2^{11} + 1x2^{10} + 1x2^9 + 0x2^8 + 1x2^7 + 1x2^6 + 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 0x2^0$
$= 44780_{10}$

(iii)Binary to hexadecimal:

1010 1110 1110 1100
1010 1110 1110 1100
10 14 14 12
A E E C = $AEEC_{16}$

52. Perform the following addition using BCD and Excess-3 addition (205+569) *[Apr'15]*

BCD:

205 - 0 0 1 0 0 0 0 0 0 1 0 1
569 - 0 1 0 1 0 1 1 0 1 0 0 1
0 1 1 1 0 1 1 0 1 1 1 0
+ 0 1 1 0
0 1 1 1 0 1 1 1 0 1 0 0

(205+569)= $(0\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0)_2$

Excess-3:

205 - 0 0 1 0 0 0 0 0 0 1 0 1
0 0 1 1 0 0 1 1 0 0 1 1
Excess-3 of 205 - 0 1 0 1 0 0 1 1 1 0 0 0

569 - 0 1 0 1 0 1 1 0 1 0 0 1
0 0 1 1 0 0 1 1 0 0 1 1
Excess-3 of 569 - 1 0 0 0 1 0 0 1 1 1 0 0

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Excess-3 addition:

0 1 0 1	0 0 1 1	1 0 0 0
1 0 0 0	1 0 0 1	1 1 0 0
1 1 0 1	1 1 0 1	0 1 0 0
- 0 0 1 1	- 0 0 1 1	+0 0 1 1
1 0 1 0	1 0 1 0	0 1 1 1

$(205+569) = (1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 1)_2$

53. Perform the following operation $(756)_8 - (437)_8 + (725)_{16}$. Express the answer in octal form. *[Nov'15]*

$(756)_8$ - 1 1 1 1 0 1 1 1 0

$(437)_8$ - 1 0 0 0 1 1 1 1 1

$(725)_{16}$ - 0 1 1 1 0 0 1 0 0 1 0 1

Since $(437)_8$ is in negative, taking 2's complement,

$(437)_8$ - 1 0 0 0 1 1 1 1 1

0 1 1 1 0 0 0 0 1

Adding: $(756)_8$ - 1 1 1 1 0 1 1 1 0

0 1 1 1 0 0 0 0 1

1 0 1 1 0 0 1 1 1

Converting octal value to hexa value:

1 0 1 1 0 0 1 1 1 1

Adding: 1 0 1 1 0 0 1 1 1 1

$(725)_{16}$ - 0 1 1 1 0 0 1 0 0 1 0 1

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

Hexadecimal to Octal conversion:

100	111	110	100
4	7	6	4

$(756)_8 - (437)_8 + (725)_{16} = (4764)_8$

11. Given the two binary numbers X=1010100 and Y= 1000011, perform the subtraction Y-X by using 2's complements. *[Nov'16]*[OR]**Given the two binary numbers X = 1010100 and Y = 1000011, perform the subtraction (a) X -Y and (b) Y -X using 2's complements.**

a) X = 1010100

2's complement of Y = + 0111101

Sum = 10010001

Discard end carry

X -Y = 0010001

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

b) Y = 1000011
2's complement of X = + 0101100
Sum = 1101111
There is no end carry,
Therefore the answer is Y-X = - (2's complement of 1101111)
= -0010001

12. Explain in detail about the classification of binary codes. *[Nov'15] or* **Explain in detail about error detecting and error correcting coding.** *[Nov'17]*

Weighted codes:

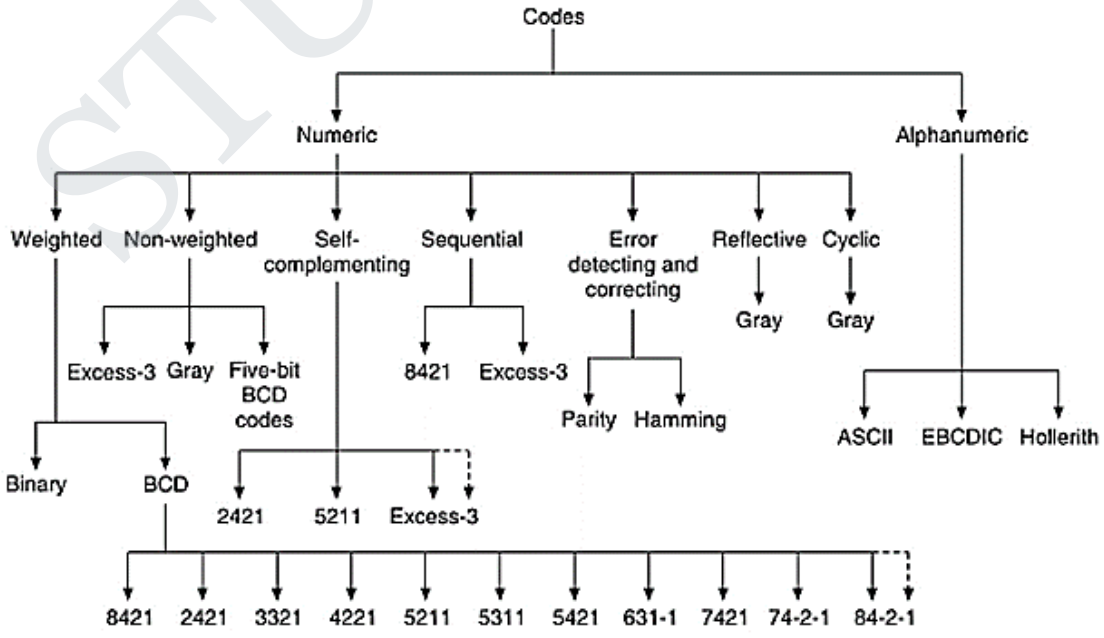
- In weighted codes, each digit is assigned a specific weight according to its position.
- For example, in 8421 BCD code, 1001 the weights of 1, 0, 0, 1 (from left to right) are 8, 4, 2 and 1 respectively.
- Suppose W_1 W_2 W_3 and W_4 are the weights of binary digits and X_1 X_2 X_3 and X_4 are the corresponding digit values then decimal digit.
- $N = W_1 X_1 + W_2 X_2 + W_3 X_3 + W_4 X_4$ is represented by binary sequence X_4 X_3 X_2 X_4.
- The codes 8421 BCD, 2421 BCD, 5211 BCD are all weighted codes.

Non-weighted codes:

- The non-weighted codes are not positional weighted.
- In other words, each digit position within the number is not assigned a fixed value (or weight).
- Excess-3 and gray code are non-weighted codes.

Reflective codes:

- A code is reflective when the code is self-complementing.
- In other words, when the code for 9 is the complement of 0, 8 for 1, 7 for 2, 6 for 3 and 5 for 4.
- 2421BCD, 5421BCD and Excess-3 code are reflective codes.



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Sequential codes:

- In sequential codes, each succeeding code is one binary number greater than its preceding code.
- This property helps in manipulation of data.
- 8421 BCD and Excess-3 are sequential codes.

Alphanumeric codes:

- Codes used to represent numbers, alphabetic characters, symbols and various instructions necessary for conveying intelligible information.
- ASCII, EBCDIC, UNICODE are the most-commonly used alphanumeric codes.

Error detecting and correcting codes:

- Codes which allow error detection and correction are called error detecting and correcting codes.
- Hamming code is the mostly commonly used error detecting and correcting code.

Error detection and Error correction:

- Error detection is the ability to detect errors.
- Error correction has an additional feature that enables identification and correction of the errors.
- Error detection always precedes error correction. Both can be achieved by having extra/redundant/check bits in addition to data deduce that there is anerror.

Decimal	Binary 8 4 2 1	Excess - 3	Gray Code
0	0 0 0 0	0 0 1 1	0 0 0 0
1	0 0 0 1	0 1 0 0	0 0 0 1
2	0 0 1 0	0 1 0 1	0 0 1 1
3	0 0 1 1	0 1 1 0	0 0 1 0
4	0 1 0 0	0 1 1 1	0 1 1 0
5	0 1 0 1	1 0 0 0	0 1 1 1
6	0 1 1 0	1 0 0 1	1 0 1 0
7	0 1 1 1	1 0 1 0	0 1 0 0
8	1 0 0 0	1 0 1 1	1 1 0 0
9	1 0 0 1	1 1 0 0	1 1 0 1
10	1 0 1 0	1 1 0 1	1 1 1 1
11	1 0 1 1	1 1 1 0	
12	1 1 0 0	1 1 1 1	
13	1 1 0 1		
14	1 1 1 0		
15	1 1 1 1		

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Decimal	Binary 8 4 2 1	BCD	2 4 2 1	3 3 2 1	8 4 -2 -1	5 4 2 1
0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1	0 0 0 1	0 0 0 1	0 1 1 1	0 0 0 1
2	0 0 1 0	0 0 1 0	0 0 1 0	0 0 1 0	0 1 1 0	0 0 1 0
3	0 0 1 1	0 0 1 1	0 0 1 1	0 1 0 0	0 1 0 1	0 0 1 1
4	0 1 0 0	0 1 0 0	0 1 0 0	0 1 0 1	0 1 0 0	0 1 0 0
5	0 1 0 1	0 1 0 1	1 0 1 1	0 1 1 0	1 0 1 1	1 0 0 0
6	0 1 1 0	0 1 1 0	1 1 0 0	0 1 1 1	1 0 1 0	1 0 0 1
7	0 1 1 1	0 1 1 1	1 1 0 1	1 1 0 1	1 0 0 1	1 0 1 0
8	1 0 0 0	1 0 0 0	1 1 1 0	1 1 1 0	1 0 0 0	1 0 1 1
9	1 0 0 1	1 0 0 1	1 1 1 1	1 1 1 1	1 1 1 1	1 1 0 0
10	1 0 1 0	0 0 0 10 0 0 0				
11	1 0 1 1	0 0 0 1 0 0 0 1				
12	1 1 0 0	0 0 0 1 0 0 1 0				
13	1 1 0 1	0 0 0 1 0 0 1 1				
14	1 1 1 0	0 0 0 1 0 1 0 0				
15	1 1 1 1	0 0 0 1 0 1 0 1				

13. Explain Hamming code with an example. State its advantages over parity codes. *[Nov'14]*

Hamming code:

- Hamming code is a code to detect error and also to correct it.
- It detects a single bit error and also identifies the bit that is in error.
- This code uses a number of parity bits located at certain positions in the code group.
- Let us consider an example for Hamming Code, Determine which bit, is in error in the odd parity, Hamming coded information 0011010_2 and decode the correct message.

Step1: Construct the bit location table

Bit destination	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	7	6	5	4	3	2	1
Bit location number	111	110	101	100	011	010	001
Received Code	0	0	1	1	0	1	0

Step 2: Check for parity bits

For P1: P1 checks locations 1, 3, 5 and 7

There is only one in the group

Hence Parity check for odd parity is correct……0 (LSB)

For P2: P2 checks locations 2, 3, 6 and 7

There is only one in the group

Hence Parity check for odd parity is correct……0

For P4: P4 checks locations 4, 5, 6 and 7

There are two 1s in the group

Hence Parity check for odd parity is wrong ……1 (MSB)

- The resultant word is C = **1 0 0.** This says that the bit in the number 4 location is in error.
- It is 1 and should be 0. Therefore, the correct code is **0 0 1 0 0 1 0**$_2$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Advantages:

- By using Hamming code we can detect and correct the errors in the information in single or double bit information.
- But in parity codes, it can only use to detect the error in the information.

14. Determine which bit, if any, is in error in the even parity, Hamming coded information 1100111_2. Decode the correct message.

Step1: Construct the bit location table

Bit destination	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	7	6	5	4	3	2	1
Bit location number	111	110	101	100	011	010	001
Received Code	1	1	0	0	1	1	1

Step 2: Check for parity bits

For P1: P1 checks locations 1, 3, 5 and 7

There are three 1's in the group

Hence Parity check for even parity is wrong……1 (LSB)

For P2: P2 checks locations 2, 3, 6 and 7

There are four 1s in the group

Hence Parity check for even parity is correct……0

For P4: P4 checks locations 4, 5, 6 and 7

There are two 1s in the group

Hence Parity check for even parity is correct……0 (MSB)

- The resultant word is C =**0 0 1.** This says that the bit in the number 1 location is in error.
- It is 1 and should be 0. Therefore, the correct code is **1 1 0 0 1 1 0**$_2$ and the message is **1 1 0 1**$_2$

15. Assume that the even parity Hamming code in example (0 1 1 0 0 1 1)$_2$ is transmitted and that (0 1 0 0 0 1 1)$_2$ is received. The receiver does not know what was transmitted. Determine bit location where error has occurred using received code.

Step1: Construct the bit location table

Bit destination	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	7	6	5	4	3	2	1
Bit location number	111	110	101	100	011	010	001
Received Code	0	1	0	0	0	1	1

Step 2: Check for parity bits

For P1: P1 checks locations 1, 3, 5 and 7

There is only one in the group

Hence Parity check for even parity is wrong……1 (LSB)

For P2: P2 checks locations 2, 3, 6 and 7

There are two 1s in the group

Hence Parity check for even parity is correct……0

For P4: P4 checks locations 4, 5, 6 and 7

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

There is only one in the group

Hence Parity check for even parity is wrong ……1 (MSB)

- The resultant word is C = **1 0 1**. This says that the bit in the number 5 location is in error.
- It is 0 and should a 1. Therefore, the correct code is **(0 1 1 0 0 1 1)$_2$**, which agrees with the transmitted code.

16. Given that a frame with bit sequence 1101011011$_2$ is transmitted, it has been received as 1101011010$_2$. Determine the method of detecting the error using any one error detecting code. *[Nov'14]*

Step1: Construct the bit location table

Bit destination	D_{10}	D_9	P_8	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	10	9	8	7	6	5	4	3	2	1
Bit location number	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Received Code	1	1	0	1	0	1	1	0	1	0

Step 2: Check for parity bits

For P1: P1 checks locations 1, 3, 5, 7 and 9

There arethree ones in the group

Hence Parity check for odd parity is correct……0 (LSB)

For P2: P2 checks locations 2, 3, 6, 7 and 10

There arethree ones in the group

Hence Parity check for odd parity is correct……0

For P4: P4 checks locations 4, 5, 6 and 7

There arethree ones in the group

Hence Parity check for odd parity is correct……0

For P8: P8 checks locations 8, 9 and 10

There aretwo ones in the group

Hence Parity check for odd parity is wrong……1 (MSB)

- The resultant word is C = **1 0 0 0**. This says that the bit in the number 8 location is in error.
- It is 0 and should be 1. Therefore, the correct code is **(**1111011010**)$_2$**, which agrees with the transmitted code.

17. The Hamming code 1 0 1 1 0 1 1 0 1 is received. Correct it if any errors. There are four parity bits and odd parity is used.

Step1: Construct the bit location table

Bit destination	D_9	p_8	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	9	8	7	6	5	4	3	2	1
Bit location number	1001	1000	0111	0110	0101	0100	0011	0010	0001
Received Code	1	0	1	1	0	1	1	0	1

Step 2: Check for parity bits

For P1: P1 checks locations 1, 3, 5, 7 and 9

There are four 1s in the group

Hence Parity check for odd parity is wrong……1 (LSB)

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

For P2: P2 checks locations 2, 3, 6 and 7
There are three 1s in the group
Hence Parity check for odd parity is correct……0
For P4: P4 checks locations 4, 5, 6 and 7
There are three 1s in the group
Hence Parity check for odd parity is correct ……0
For P8: P8 checks locations 8 and 9
There is only one in the group
Hence Parity check for odd parity is correct ……0 (MSB)

- The resultant word is C = **0 0 0 1**. This says that the bit in the number 1 location is in error.
- It is 1 and should a 0. Therefore, the correct code is **101101100_2**.

18. A 12-bit Hamming code word containing 8 bits of data and 4 parity bits is read from memory. What was the original 8-bit data word that was written into memory if the 12-bit word read out is as (1) 101110010100 and (2) 111111110100? *[Nov'15]*

(1) 101110010100 (using Even parity)

Step1: Construct the bit location table

Bit destination	D_{12}	D_{11}	D_{10}	D_9	P_8	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	12	11	10	9	8	7	6	5	4	3	2	1
Bit location number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Received Code	1	0	1	1	1	0	0	1	0	1	0	0

Step 2: Check for parity bits
For P1: P1 checks locations 1, 3, 5, 7, 9, 11
There are three 1s in the group
Hence Parity check for even parity is wrong……1 (LSB)
For P2: P2 checks locations 2, 3, 6, 7, 10, 11
There are two 1s in the group
Hence Parity check for even parity is correct……0
For P4: P4 checks locations 4, 5, 6, 7, 12
There are two 1s in the group
Hence Parity check for even parity is correct……0
For P8: P8 checks locations 8, 9, 10, 11, 12
There are four 1sin the group
Hence Parity check for even parity is correct ……0 (MSB)

Check bits: C = **1 0 0 0**, i.e. Erroris in bit 8= 101100010100_2

Actual 8 bit data stored in memory is **10110011_2**

(2) 111111110100 (using odd parity)
Step1: Construct the bit location table

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Bit destination	D_{12}	D_{11}	D_{10}	D_9	P_8	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	12	11	10	9	8	7	6	5	4	3	2	1
Bit location number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Received Code	1	1	1	1	1	1	1	1	0	1	0	0

Step 2: Check for parity bits

For P1: P1 checks locations 1, 3, 5, 7, 9, 11

There are five 1s in the group

Hence Parity check for odd parity is correct……0 (LSB)

For P2: P2 checks locations 2, 3, 6, 7, 10, 11

There are five 1s in the group

Hence Parity check for odd parity is correct ……0

For P4: P4 checks locations 4, 5, 6, 7, 12

There are four 1s in the group

Hence Parity check for odd parity is wrong……1

For P8: P8 checks locations 8, 9, 10, 11, 12

There are five 1s in the group

Hence Parity check for odd parity is correct……0 (MSB)

- Check bits: C = **0100**, i.e. Error is in bit 4 = 111111111100_2
- Actual 8 bit data stored in memory is **11111111_2**

19. The message below has been coded in the even parity Hamming code and transmitted through a noisy channel. Decode the message that at most a single error has occurred in each word code. i). 1001001 ii) 0111001 iii) 1110110 iv) 00110011

Bit destination	D_7	D_6	D_5	P_4	D_3	P_2	P_1	Error code
Bit location	7	6	5	4	3	2	1	
Hamming coded message	1	0	0	1	0	0	1	
1, 3, 5 and 7 check for P_1	1		0		0		1	0
2, 3, 6 and 7 check for P_2	1	0			0	0		1
4, 5, 6 and 7 check for P_4	1	0	0	1				0

Error in bit position 2. Therefore, hamming code should be 1001011_2 and message is 1000_2.

ii)

Bit destination	D_7	D_6	D_5	P_4	D_3	P_2	P_1	Error code
Bit location	7	6	5	4	3	2	1	
Hamming coded message	0	1	1	1	0	0	1	
1, 3, 5 and 7 check for P_1	0		1		0		1	0
2, 3, 6 and 7 check for P_2	0	1			0	0		1
4, 5, 6 and 7 check for P_4	0	1	1	1				1

Error in bit position 6. Therefore, hamming code should be 0011001_2 and message is 0010_2.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

iii)

Bit destination	D_7	D_6	D_5	P_4	D_3	P_2	P_1	Error code
Bit location	7	6	5	4	3	2	1	
Hamming coded message	1	1	1	0	1	1	0	
1, 3, 5 and 7 check for P_1	1		1		1		0	1
2, 3, 6 and 7 check for P_2	1	1			1	1		0
4, 5, 6 and 7 check for P_4	1	1	1	0				1

Error in bit position 5. Therefore, hamming code should be 1100110_2 and message is 1101_2.

iv)

Bit destination	D_7	D_6	D_5	P_4	D_3	P_2	P_1	Error code
Bit location	7	6	5	4	3	2	1	
Hamming coded message	0	0	1	1	0	1	1	
1, 3, 5 and 7 check for P_1	0		1		0		1	0
2, 3, 6 and 7 check for P_2	0	0			0	1		1
4, 5, 6 and 7 check for P_4	0	0	1	1			1	0

Error in bit position 2. Therefore, hamming code should be 0011001_2 and message is 0010_2.

20. Encode the information character 01101110101 according to the 15-bit hamming code.

- In this the message bits are 11 and parity bits are 4.
- We know that the parity bits are located in the positions that are numbered corresponding to ascending powers of two (1, 2, 4, 8…).
- Therefore, we have following, format for 15-bit hamming code. We also know that parity bits are assigned by checking message bits having 1 in the same location as parity bit in their binary location numbers.

Bit destination	D_{15}	D_{14}	D_{13}	D_{12}	D_{11}	D_{10}	D_9	P_8	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Bit location number	1111	1110	1101	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Information bits	0	1	1	0	1	1	1		0	1	0		1		
Check for P_1: 3, 5, 7, 9, 11, 13, 15	0		1		1		1		0		0		1		0
Check for P_2: 3, 6, 7, 10, 11, 14, 15	0	1			1	1			0	1			1	1	

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Check for P_4: 5, 6, 7, 12, 13, 14, 15	0	1	1	0					0	1	0	1			
Check for P_8: 9, 10, 11, 12, 13, 14, 15	0	1	1	0	1	1	1	1							
Hamming coded message	0	1	1	0	1	1	1	1	0	1	0	1	1	1	0

The 15 – bit hamming code is 011011110101110_2

21. Encode the binary word 1011 into seven bit even parity Hamming code. *[Apr'15]*

Bit destination	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	7	6	5	4	3	2	1
Bit location number	111	110	101	100	011	010	001
Information Bit	1	0	1		1		
3, 5 and 7 check for P_1	1		1		1		1
3, 6 and 7 check for P_2	1	0			1	0	
4, 5, 6 and 7 check for P_4	1	0	1	0			
Hamming coded message	1	0	1	0	1	0	1

Thus the encoded seven bit parity Hamming code is 1010101_2

22. Explain in detail the usage of Hamming codes for error detection and error correction with an example considering the data bits as 0101. *[Nov'16]*

Let us assume a odd parity

Bit destination	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	7	6	5	4	3	2	1
Bit location number	111	110	101	100	011	010	001
Information Bit	0	1	0		1		
3, 5 and 7 check for P_1	0		0		1		0
3, 6 and 7 check for P_2	0	1			1	1	
4, 5, 6 and 7 check for P_4	0	1	0	0			
Hamming coded message	0	1	0	0	1	1	0

The 7 – bit hamming code is 0100110_2

23. Deduce the odd parity hamming code for the data: 1010_2. Introduce an error in the LSB of the hamming code and deduce the steps to detect the error. *[Apr'16]* [OR] **Design a odd parity hamming code generator and detector for 4 bit data and explain their logic.** *[May'17]*

Bit destination	D_7	D_6	D_5	P_4	D_3	P_2	P_1
Bit location	7	6	5	4	3	2	1
Bit location number	111	110	101	100	011	010	001
Information Bit	1	0	1		0		

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

3, 5 and 7 check for P_1	1		1		0		1
3, 6 and 7 check for P_2	1	0			0	0	
4, 5, 6 and 7 check for P_4	1	0	1	1			
Hamming coded message	1	0	1	1	0	0	1

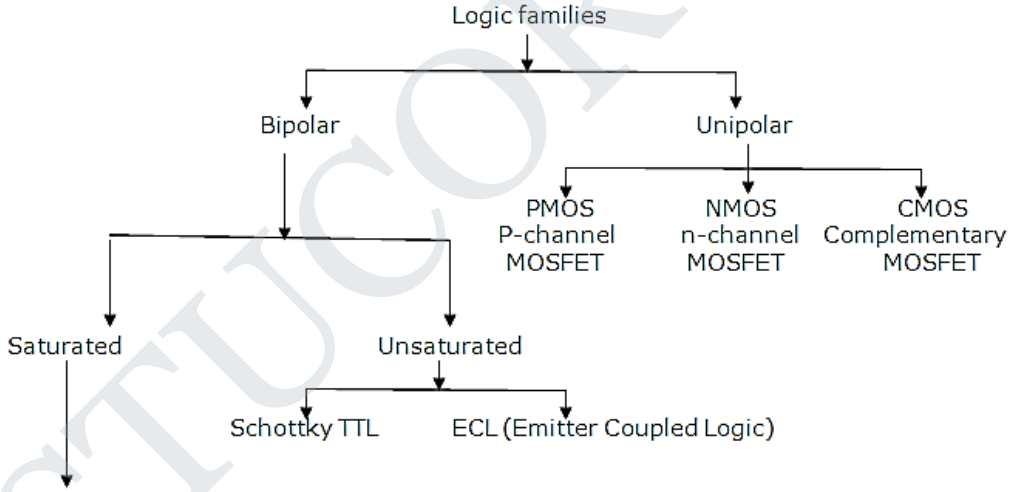
24. Write notes on digital logic families.

A digital logic family is a group of compatible devices with the same logic levels and supply voltages. According to components used in the logic family, digital logic families are classified as shown in the figure.

Of the above the most widely used Logic families are TTL, CMOS and ECL, due to their characteristics matching the hardware requirements.

Transistor Transistor Logic (TTL)

- Transistor Transistor logic, TTL, is named for its dependence on transistor alone to basic operations.
- The first version, which is now known as standard TTL, was developed in 1965 and is rarely used in today's system.
- Through the years, the basic design has been modified to improve its performance in several aspects and as a consequence, a number of subfamilies have evolved.



CMOS Logic:

Digital circuit with MOSFETs can be grouped into three categories:

- PMOS - Uses only P-channel enhancement MOSFETs,
- NMOS - Uses only N-channel enhancement MOSFETs, and
- CMOS (Complementary MOS) – Uses both P and N-channel devices.

- PMOS and NMOS digital ICs are economical than CMOS ICs because they have greater packing density than CMOS.
- NMOS has twice the packing density than PMOS.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- NMOS can operate at about three times faster than their PMOS counterparts. This is because NMOS has faster moving current carriers (holes).
- CMOS has the greatest complexity and lowest packaging density
- CMOS has advantages of high speed and much lower dissipation.
- NMOS and CMOS are widely used in the digital ICs, but PMOS are no longer part of new designs.
- CMOS circuit contains both NMOS and PMOS devices to speed the switching of capacitive loads. It consumes low power and can operate at high voltages, resulting in improved noise immunity.

ECL Family

- The TTL family uses transistors operating in the saturation mode.
- As a result, their switching speed is limited by the storage delay time associated with transistor that is driven into saturation.
- Another logic family has been developed that prevents transistor saturation, thereby increasing overall switching speed by using radically different circuit structure, called current mode logic (CML).
- This logic family is also called emitter-coupled logic (ECL).
- Unlike TTL and CMOS families, ECL does not produce a large voltage swing between the LOW and HIGH levels.
- It has a small voltage swing, less than a volt, and it internally switches current between two possible paths, depending on the output state

25. Explain the concept of working of RTL logic families. *[Nov'17]*

- In this logic family of ICs, the series of resistors are added to each transistor.
- By reducing the current – hogging effect with resistors, a larger fan-out is achieved.
- But due to the resistor's presence, the speed of the circuit will be always slow.

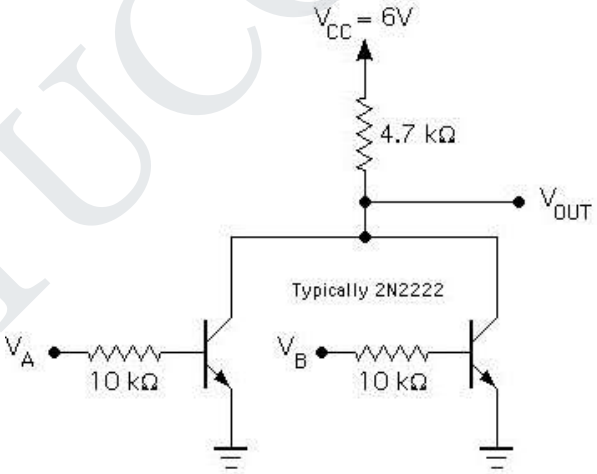


Fig. RTL NOR gate circuit

RTL working:

- When inputs A, B are '0', the transistors Q_1 and Q_2 are OFF. Thus the node C is not connected to ground and the V_{cc} will appear at node C s output which is logic '1'.
- When any one inputs either A or B is '1' or if both A and B are '1' Q1 or Q2 or both the transistors will be in saturated mode. Thus the node C will be connected to ground making the output C as 0V or Logic LOW for all the remaining three conditions.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- If more number of resistors are included in the logic circuit, then the input resistance gets increased and switching speed will decrease. An alternate approach to increase the switching speed in RTL is to add a capacitor parallel to the resistor in the input of the transistor's base.
- Another problem is the transistors go to saturation causing longer turn off delay (i.e.,) it takes more time for the output to become 1 to 0. Integrated Injection Logic (IIL) can eliminate all the problems of the RTL circuit.

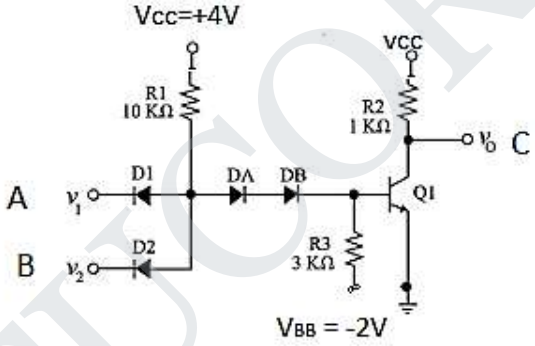
Characteristics of RTL logic circuit:

1. Speed of operation is low. The propagation delay is in the order of 500ns. It cannot operate at speeds above 4MHz.
2. Fan out is 4 or 5 with a switching delay of 50ns and fan in is 4.
3. Poor noise immunity.
4. High average power dissipation due to resistors.
5. The noise margin from zero to the threshold voltage is about 0.5V and from one to the threshold voltage is 0.2V.
6. Sensitive to temperature.

The RTL family of ICs includes NOR gates, flip flops and 4 bit shift registers.

26. Explain the working of DTL logic families. *[Nov'17]*

- The formation of NOT gate using Diode Logic is difficult and requires two voltage levels to represent logic HIGH and LOW. To avoid this, transistor inverter is combined with diodes to form NAND and NOR gates.



- The DTL circuit combines the diode AND gate and the bipolar transistor inverter into a NAND gate. The AND function is performed by two diodes with a resistor for pull up and NOT function is formed by the transistor inverter circuit.
- When A=0, B=0, the node X has 0V. This 0V is given as an input to the transistor Q_1. The transistor will be in cut off condition only. Node C will have +5V (HIGH).
- Similarly if any one input is 0, A=0, B=0 or both A and B are 0, then the node X will be grounded. Thus there is no base current. The transistor will be in cut off condition. Therefore the node C will have +5V (HIGH).
- But for the inputs A=1, B=1, A and B are give +5V. Now the node X will have +5V (since both diodes do not conduct). This voltage is given to the transistor's base with a drop by R_2.
- Now the transistor conducts. The output of the NAND gate is LOW. The node C is grounded. In this way we can generate NOR using DTL circuits.

27. Explain the concept, working and characteristics of TTL logic families. *[Nov'10]* [OR] **Design a TTL logic circuit for a 3 input NAND gate.** *[Nov'14]* [OR] **With circuit schematic explain the working of a two-input TTL NAND gate.** *[May'17]* [OR] **With circuit schematic explain the operation of a two input TTL NAND gate.** *[Apr'16]*

STUCOR APP
DOWNLOADED FROM STUCOR APP

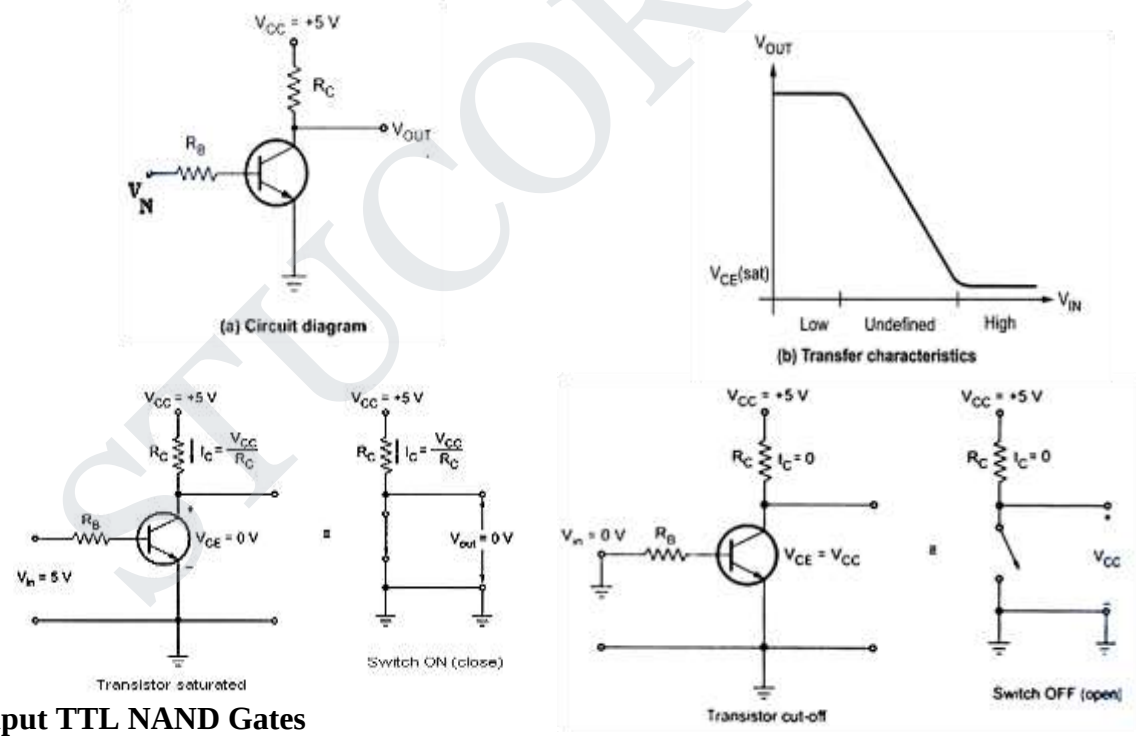
DOWNLOADED FROM STUCOR APP

Transistor -Transistor Logic (TTL):

- Transistor Transistor logic, TTL, is named for its dependence on transistor alone to basic operations.
- The first version, which is now known as standard TTL, was developed in 1965 and is rarely used in today's system.
- The subfamily circuits along with their characteristics of TTL are discussed below. They are
    - TTL inverter
    - TTL 2-input NAND gate
    - TTL 3-input NAND gate
    - Totem-pole output
    - Open collector output

TTL Inverter

- We have seen that when the input voltage is low, the output voltage is HIGH and vice versa.
- Therefore, we can make a logic inverter from an NPN transistor in the common emitter configuration.
- The operation of transistor inverter for both the input (HIGH and LOW) using switching analogy is shown below.

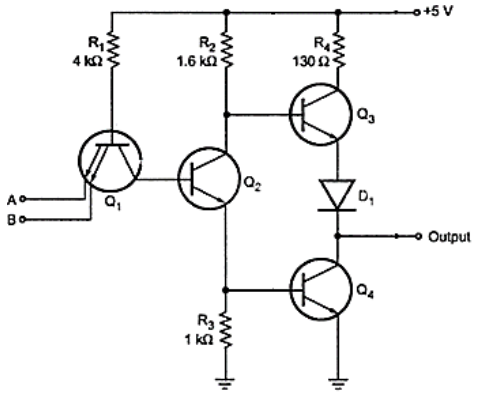


2-Input TTL NAND Gates

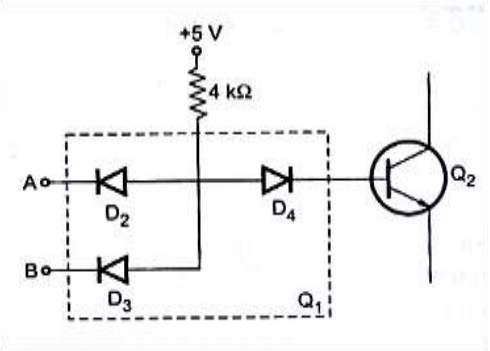
- The circuit diagram of 2-input TTL NAND Gate is as shown in figure.
- Its input structure consists of multiple-emitter transistor and output structure consists of totem-pole output.
- Here, Q_1 is an NPN transistor having two emitters, one for each input to the gate.
- Although this circuits looks complex, we can simplify its analysis by using the diode equivalent of the multiple-emitter transistor Q_1.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



2-input TTL NAND gate



Diode equivalent for Q_1

- The diodes D_2 and D_3 represents the two E-B junction of Q_1 and D_4 is the collector-base(C-B) junction.
- The input voltages A and B are either LOW (ideally grounded) or HIGH (ideally +5 volts).
- If either A and B or both are low, the corresponding diode conducts and the base of Q_1 is pulled to approximately 0.7V. This reduces the base voltage of Q_2 to atmost zero. Therefore, Q_2 cuts off. With Q_2 open, Q_4 goes into cut-off and the Q_3 Base is pulled HIGH. Since Q_3 acts as an emitter follower, the Y output is pulled up to a HIGH voltage.
- On the other hand, when A and B both are HIGH, the emitter diode of Q_1 is reverse biased making them off. This causes the collector diode D_4 to get in to forward conduction. This forces Q_2 base to go HIGH. In turn, Q_4 goes into saturation, producing a low output in all input and output conditions.
- Without diode D_1 in the circuit, Q_3 will conduct slightly when the output is low.
- To prevent this, the diode is inserted. Its voltage drops keeps the base-emitter diode of Q_3 reverse biased.
- In this way, only Q_4 conducts when the output is low.

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

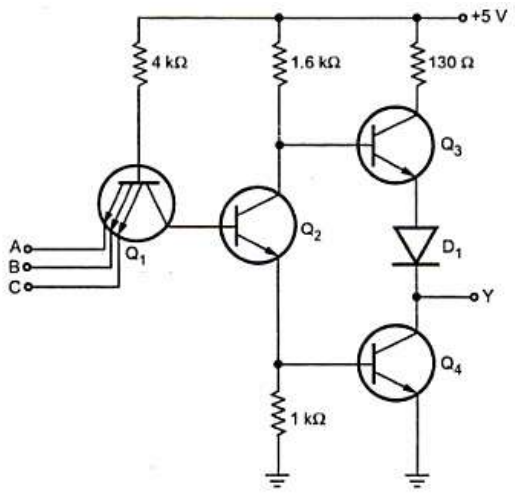
Truth table for 2-input NAND gate

3-Input TTL NAND Gate:

- The three inputs TTL NAND Gate is same as that of two input TTL NAND Gate except that its Q_1 (NPN) transistor has three emitters instead of two. Rest of the circuit is same.
- For three input NAND gate if all the inputs are logic 1 then only output is logic 0; otherwise output is logic 1. The operation is similar to the 2-input NAND gate.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

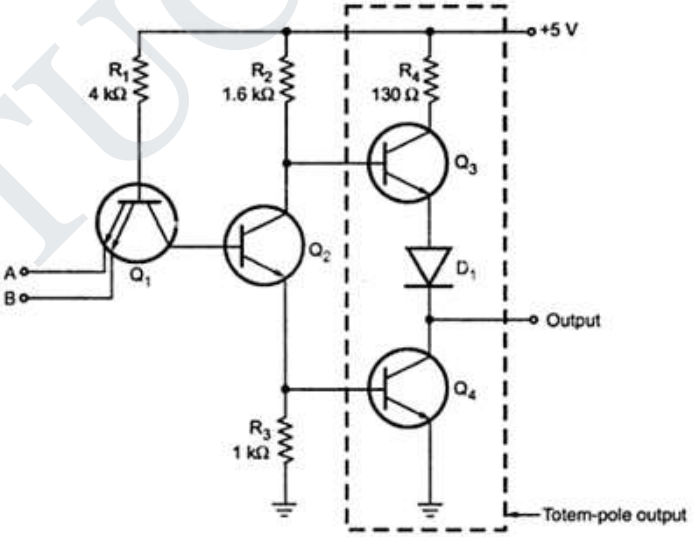


A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Three input TTL NAND gate **Truth table of 3 input NAND gate**

28. With circuit Schematic, Explain the operation of a two input TTL NAND gate with totem-pole output. *[Apr'15]*

- In the TTL circuit, transistors Q_3 and Q_4 form a totem-pole. Such a configuration is known as active pull-up or totem pole output.
- The active pull-up formed by Q_3 and Q_4 has a specific advantage. Totem-pole transistors are used because they produce LOW output impedance.
- Either Q_3 acts an emitter follower (HIGH output) or Q_4 is saturated (LOW output).
- When Q_3 is conducting, the output impedance is approximately 70Ω. When Q4 is saturated, the output impedance is only 12Ω. Either way, the output impedance is low.
- This means that the output voltage can change quickly from one state to another because any stray output capacitance is rapidly charged or discharged through the low output impedance.
- Thus the propagation delay is low in totem-pole TTL logic.



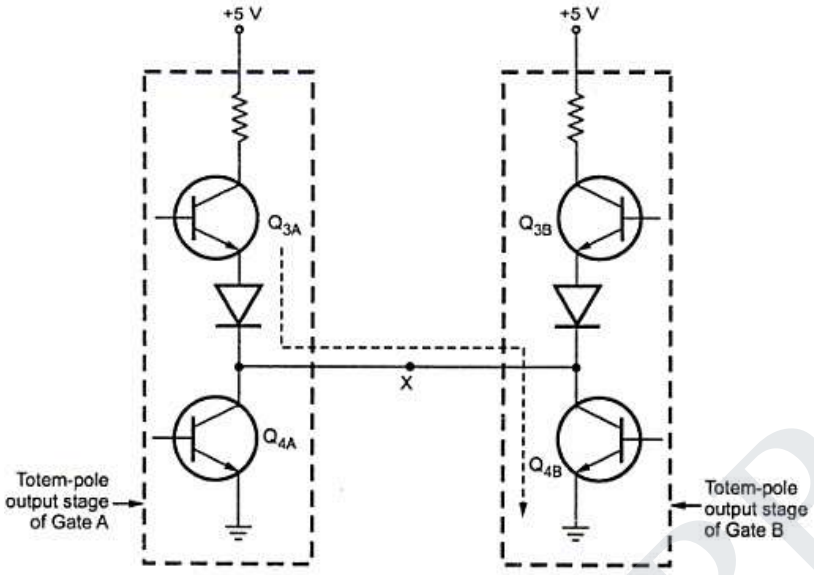
Two input NAND gate with totem-pole output

29. Explain in detail about TTL with open collector output configuration. *[Nov'13, Nov'17]*

- One problem with totem-pole output is that two outputs cannot be tied together, as shown in below figure, where the totem pole outputs of two separate gates are connected together at point X.

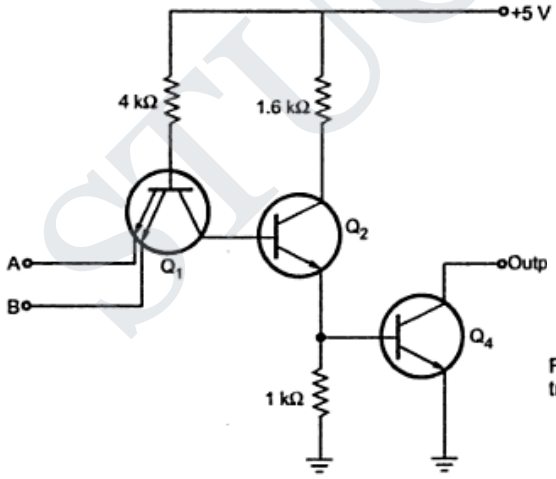
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

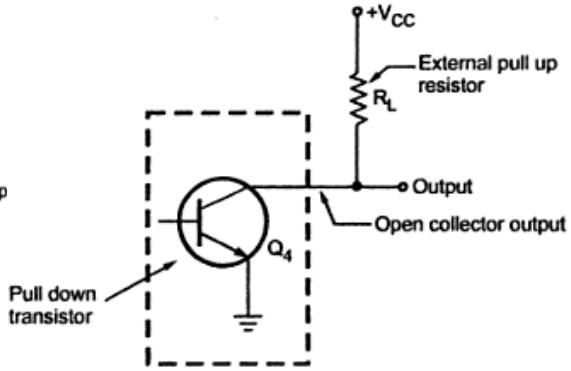


Totem-pole outputs tied together can produce harmful current

- Suppose that the output of gate A is high (Q_{3A} ON and OFF) and the output of gate B is LOW (Q_{3B} OFF and Q_{4B} ON). In this situation transistor Q_{4B} act as a load for Q_{3A}.
- Since Q_{4B} is a low resistance load, it draws high current around 55mA.
- This current might not damage Q_{3A} or Q_{4B} immediately, but over a period of time can cause overheating and deterioration in performance and eventually device failure.
- Some TTL devices provide another type of output called open collector output.
- The output of two different gates with open collector output can be tied together.
- This is known as wired logic.
- A 2-input NAND gate with an open-collector output eliminates the pull-up transistor Q_3, D_1 and R_4.
- The output is taken from the open collector terminal of transistor Q_4.
- Totem pole o/p tied together can produce harmful current.



Open collector 2-input TTL NAND ga



Open collector output with pull-up resistor

- Because the collector of Q_4 is open, a gate like this will not work properly until you connect an pull-up resistor.
- When Q_4 is OFF output is tied to V_{cc} through an external pull up resistor.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- As mentioned earlier, the open collector output of two or more gates can be connected together, as connection is called a wired-AND and represented schematically by the special AND gate symbol.

30. Compare the Totem pole and open collector outputs. *[Apr'15][May'17]*

Sl.No.	Totem Pole	Open Collector
1.	Output stage consists of pull up transistor (Q_4), diode resistor and pull down transistor (Q_5).	Output stage consists of only pull down transistor.
2.	External pull up resistor is not required.	External pull up resistor is not required for proper operation of gate.
3.	Output of two gates cannot be tied together.	Output of two gates can be tied together using Wired AND technique.
4.	Operating speed is high.	Operating speed is low.

31. Demonstrate the CMOS logic circuit configuration and characteristics in details. *[Nov'13]* (OR) **Draw a CMOS two neither input NOR gate and NAND gate.** (OR) **Explain the characteristics of CMOS.** *[Nov'11]* (OR) **Draw the MOS logic circuit for NOT gate and explain its operation.** *[Nov'14]* (OR) **Draw the CMOS logic circuit for NOR gate and explain its operation.** *[Nov'15]* (OR) **Explain with an aid of circuit diagram the operation of 2 input CMOS NAND gate and list out its advantages over other logic families**. *[Nov'16, Apr'18]*

CMOS Logic:

Digital circuit with MOSFETs can be grouped into three categories:

- PMOS - Uses only P-channel enhancement MOSFETs,
- NMOS - Uses only N-channel enhancement MOSFETs, and
- CMOS (Complementary MOS) – Uses both P and N-channel devices.

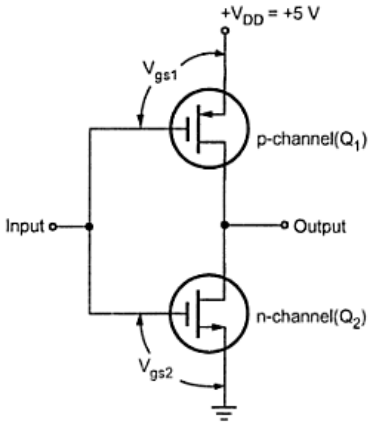
- PMOS and NMOS digital ICs are economical than CMOS ICs because they have greater packing density than CMOS.

- NMOS has twice the packing density than PMOS. Furthermore, NMOS can operate at about three times faster than their PMOS counterparts.

- This is because NMOS has faster moving current carriers (holes). CMOS has the greatest complexity and lowest packaging density.

- However, it has advantages of high speed and much lower dissipation. NMOS and CMOS are widely used in the digital ICs, but PMOS are no longer part of new designs.

- CMOS circuit contains both NMOS and PMOS devices to speed the switching of capacitive loads. It consumes low power and can operate at high voltages, resulting in improved noise immunity.



CMOS inverter circuit

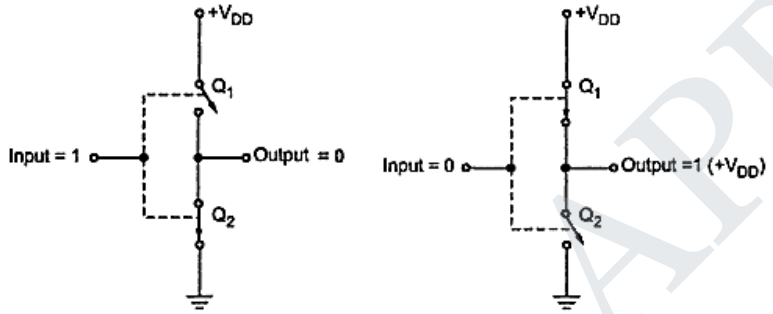
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

CMOS Inverter:

- It consists of two MOSFET's in series in such a way that the p-channel device has its source connected to $+V_{DD}$ and the n-channel device has its source connected to ground.
- The gates of the two devices are connected together as the common input and the drains are connected together as the common output.

1. When input is HIGH, the gate of Q_1 (p=channel) is at 0 V relative to the source of Q_1 i.e. $V_{gs1} = 0$ V. Thus Q_1 is OFF. On the other hand, the gate of Q_2 (n-channel) as at $+V_{DD}$ relative to its source i.e. $V_{gs2} = +V_{DD}$. Thus, Q_2 is ON. This will produce $V_{OUT} = 0$ V as in figure.
2. When input is LOW, the gate of Q_1 (p=channel) is at negative potential relative to its source while Q_2 has $V_{gs} = 0$ V. Thus Q_1 is ON and Q_2 is OFF. This produces output voltage approximately $+V_{DD}$ in figure.



(a) Input = 1 (b) Input = 0

Operation of CMOS inverter for both input conditions

Truth Table

A	Q1	Q2	Output
0	ON	OFF	1
1	OFF	ON	0

CMOS NAND Gate :

- It consists of two p-channel MOSFET's Q_1 and Q_2, connected in parallel and two n-channel MOSFET's Q_3 and Q_4 connected in series.

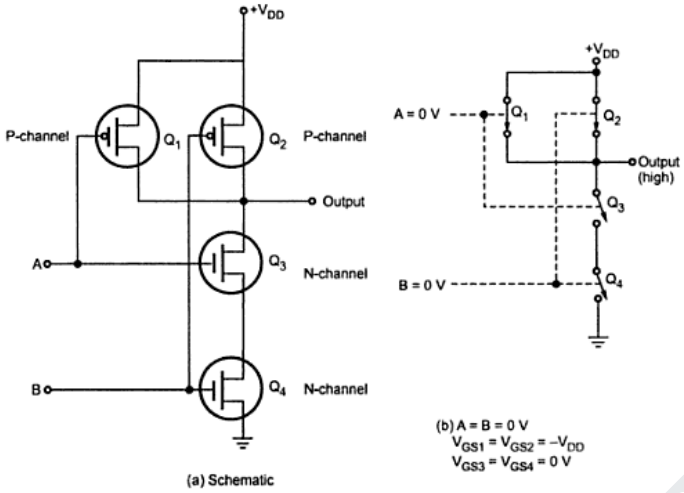
1. When both the inputs are low, the gates of both p-channel MOSFET's are negative with respect to their source, since the sources are connected to $+V_{DD}$. Thus Q_1 and Q_2 are both ON. Since the gate-to-source voltages of Q_3 and Q_4 (n-channel MOSFETs) are both 0 V, those MOSFET's are OFF. The output is therefore connected to $+V_{DD}$ (HIGH) through Q_1 and Q_2 and is disconnected from ground, as shown in fig(b).
2. When A=0 and B=$+V_{DD}$, Q_1 is ON because V_{gs1}=$-V_{DD}$ and Q_4 is ON because V_{gs4}=$+V_{DD}$. MOSFET's Q_2 and Q_3 are OFF because their gate-to-source voltages are 0 V. Since Q_1 is ON and Q_3 is OFF, the output is connected to $+V_{DD}$ and it is disconnected from ground. Output is HIGH.
3. When A=$+V_{DD}$ and B=0, Q_1 is OFF because V_{gs1}=$+V_{DD}$ and Q_4 is OFF because V_{gs4}=$-V_{DD}$. MOSFET's Q_2 and Q_3 are ON because their gate-to-source voltage is $+V_{DD}$. Since Q_2 and Q_3 are ON, the output is connected to $+V_{DD}$ and it is disconnected from ground. Output is HIGH.
4. Finally, when both inputs are HIGH, Q_1 and Q_2 are both OFF and Q_3 and Q_4 are both ON, therefore the output is connected to ground and is LOW.

Note:

- P-channel MOSFET is ON when its gate voltage is negative with respect to its source
- N-channel MOSFET is ON when its gate voltage is positive with respect to its source.

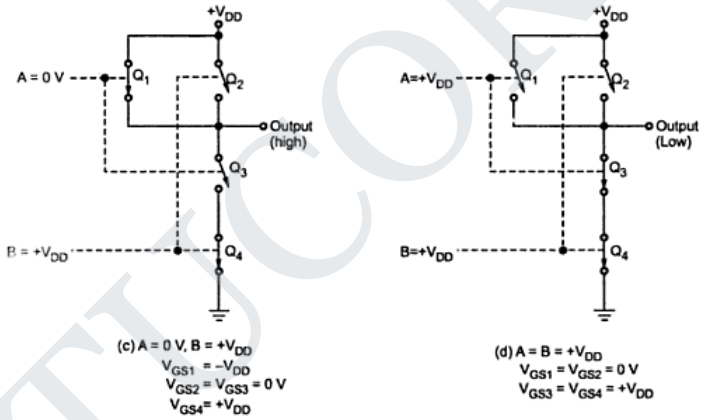
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



A	B	Q_1	Q_2	Q_3	Q_4	Output
0	0	ON	ON	OFF	OFF	1
0	1	ON	OFF	OFF	ON	1
1	0	OFF	ON	ON	OFF	1
1	1	OFF	OFF	ON	ON	0

Truth table for CMOS NAND gate



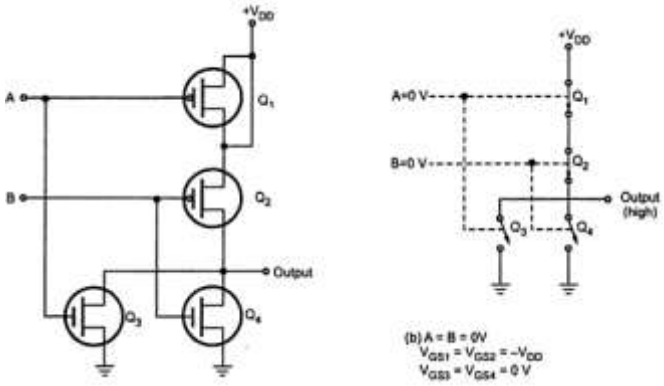
CMOS NAND gate

CMOS NOR Gate:

- Below figure shows 2-input CMOS NOR gate. Here, p-channel MOSFET's Q_1 and Q_2 are connected in series and n-channel MOSFET's Q3 and Q_4 are connected in parallel.
- Like NAND circuit, this circuit can be analyzed by realizing that a LOW at any input turns ON its corresponding p-channel MOSFET and turns OFF its corresponding n-channel MOSFET, and vice versa for a HIGH input.

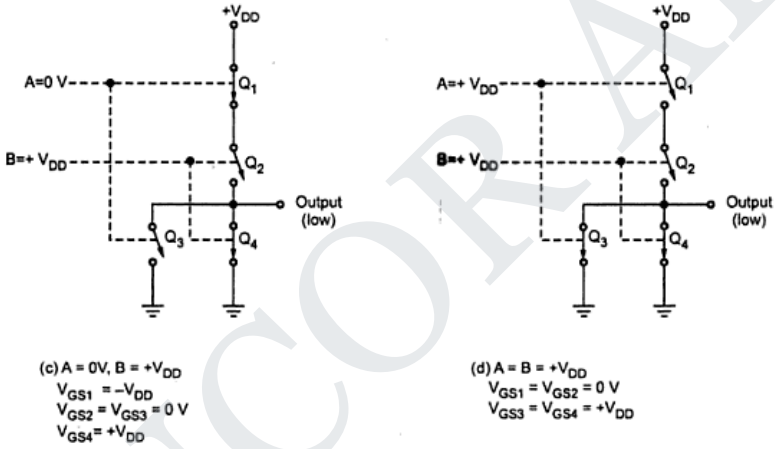
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



A	B	Q_1	Q_2	Q_3	Q_4	Output
0	0	ON	ON	OFF	OFF	1
0	1	ON	OFF	OFF	ON	0
1	0	OFF	ON	ON	OFF	0
1	1	OFF	OFF	ON	ON	0

Truth table for CMOS NOR gate



CMOS NOR gate

32. Explain the characteristics of CMOS family.

Characteristics of CMOS family:

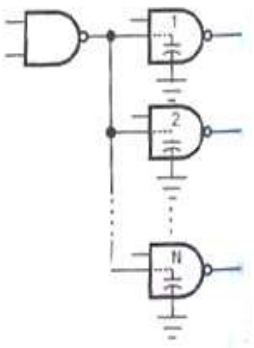
- **Operating Speed:** Slower than TTL series. Approximately 25 to 100ns depending on the subfamily of CMOS. It also depends on the power supply voltage.

- **Voltage levels and noise margins:** The voltage level for CMOS varies according to their subfamilies. Noise margin are calculated as follow.

  $V_{NH} = V_{OH\,(MIN)} - V_{IH\,(MIN)}$

  $V_{NL} = V_{IL\,(MAX)} - V_{OL\,(MAX)}$

- **Fan-out**: The CMOS inputs have an extremely large resistance ($10^{12}\Omega$) that draws essentially no current from the signal source. Each CMOS input, however, typically present a 5 pF load to ground as shown in the fig. This input capacitance limits the number of CMOS inputs that one CMOS output can drive.



  - The CMOS output has to charge and discharge the parallel combination of all the input capacitances.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- This charging and discharging time increases as we increase number of loads.
- Typically, each CMOS load increases the driving circuit's propagation delay by 3ns.
- Thus, fan-out for CMOS depends on the permissible maximum propagation delay.
- Typically, CMOS outputs are limited to a fan-out of 50 for low-frequency operation ($\leq$1 MHz).
- Of course, for high-frequency operation the fan-out would have to be less.

One CMOS O/P driving Several CMOS I/P

- **Power Dissipation (P_D):**

The power dissipation of a CMOS IC is very low as long as it is in a d.c condition. Unfortunately, power dissipation of CMOS IC increases in propagation to the frequency at which the circuits are switching states. For example, a CMOS NAND gate that has P_D=10 nW under d.c conditions will have P_D=0.1 mW at a frequency of 100 kHz and 1 mW at 1 MHz.

When CMOS output switches from LOW to HIGH, a transient charging current has to be supplied to the load capacitance. Therefore, as the switching frequency increases, the average current drawn from V_{DD} also increases, resulting increase in power dissipation.

- **Propagation Delay:**
  - The propagation delay in CMOS is the sum of delay due to internal capacitance and due to load capacitance. The delay due to internal capacitance is called the intrinsic propagation delay.
  - The delay due to load capacitance can be approximated as follows.
    - **$T_p(C_L)=0.5\ R_O\ C_L$ seconds**
    - Where $T_p(C_L)$ is either t_{pLH} or t_{pHL}.
  - R_o is the output resistance of the gate and $C_L$is the total load capacitance. The R_o depends on the supply voltage and it can be approximated as
    - **$R_o=V_{CC}/I_{OS}$**
    - Where I_{OS} is the short circuit output current.

- **Unused Inputs:** CMOS inputs should never be left disconnected. All CMOS inputs have to be tied either to a fixed voltage level (0 V or V_{DD}) or to another input. This rule applies even to the inputs of extra unused logic gates on a chip. An unused CMOS input is susceptible to noise and static charges that could easily bias both the P and N-channel MOSFETs in the conductive state, resulting in increased power dissipation and possible overheating.

- **Static- charge susceptibility (CMOS Hazards):** Every CMOS device is vulnerable to the building up of electrical charge on its insulated gate. Recall that the relationship between charge Q and voltage V on a capacitor having capacitance C is

$$V=Q/C$$

Since the input capacitance at the gate is usually quite small (a few picofarads), a relatively small amount of charge can create a large voltage which may be greater than the breakdown voltage of a MOS gate (typically 100 V).

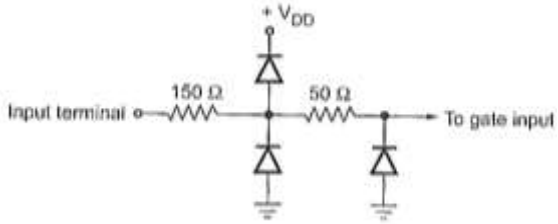
The primary source of charge is "static" electricity, usually produced by handling and the plastics and textiles. The CMOS devices are protected against this static charge by on chip diode-resistor network, as shown in the fig. these diodes are designed to turn ON and limit the size of the input voltages to well below any damaged value.

- **Latch-up:** CMOS integrated circuits contain parasitic PNP and NPN transistors: transistors that exist because of the proximity of P and N materials embedded in the substrate. Their existence is not intentional but is unavoidable. Because of conducting paths between a pair of such transistors, a device can be triggered into a heavy conducting mode, known as latch-up. This

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

heavy conduction mode, results large current flow which can destroy IC. Most CMOS circuits contain protective measures to prevent latch-up, but it can still occur if the manufactures specified maximum ratings are exceeded.

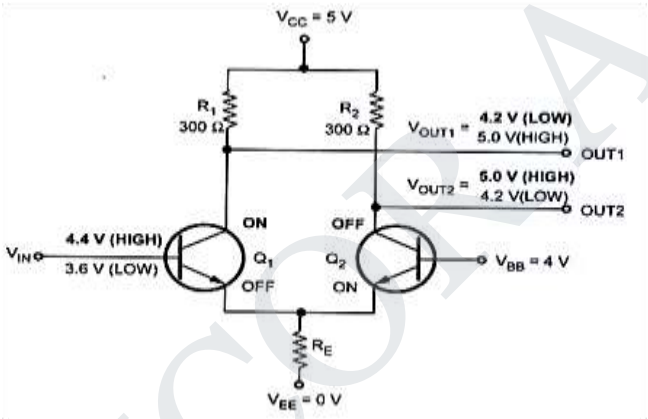


Typical network used to protect CMOS from static charges

33. Explain in detail about ECL family. OR With circuit schematic and explain the operation and characteristics of a ECL gate. *[Apr'16, Nov'17]*

Basic ECL Circuit:

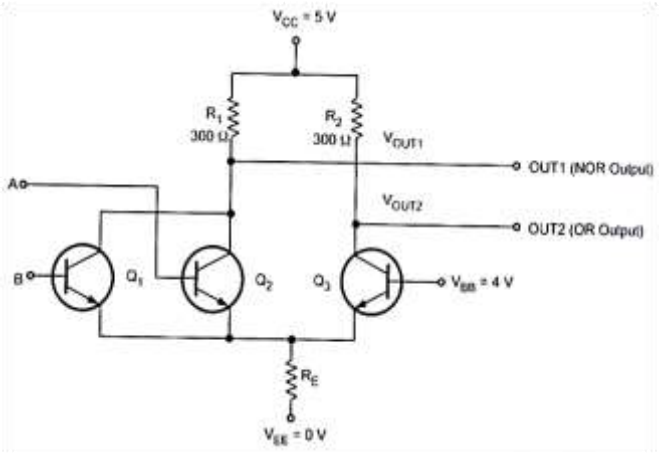
- The basic inverter/buffer circuit in ECL family consists of two transistor connected in differential single ended input mode with a common emitter resistance.



- The circuit has two outputs: inverting output (OUT1) and non-inverting output (OUT2). For this circuit, the input LOW and HIGH voltage levels are defined as 3.6 V and 4.4 V, and it produces output LOW and HIGH levels as 4.2 V and 5.0 V.
- When V_{IN} is HIGH (4.4V), transistor Q_1 is ON, but not saturated and transistor Q_2 is OFF. Thus V_{OUT2} is pulled to 5.0V (HIGH) through R_2 and drop across R_1 is 0.8 V so that V_{OUT1}.
- When V_{IN} is LOW (3.6V), transistor Q_2 is ON, but not saturated and transistor Q_1 is OFF. Thus, V_{OUT1} is pulled to 5.0V (HIGH) through R_1 and drop across R_2 is 0.8 V so that V_{OUT2} is 4.2 V (LOW).

ECL OR/NOR Gate

- The 2-input ECL OR/NOR gate and it logic symbol. There has an additional transistor in parallel with Q_1 as compared to ECL inverter.
- If any input is HIGH corresponding transistor is active, and V_{OUT1} is LOW (NOR output).At the same time Q_3 is off producing V_{OUT2} HIGH (OR output).



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

34. Compare the characteristics of TTL, ECL and CMOS logic families. *[Apr'10]*

S.No:	Parameter	CMOS	TTL	ECL
1	Device used	n-channel and p-channel MOSFET	Bipolar junction transistor	Bipolar junction transistor
2	$V_{IH(min)}$	3.5 V	2 V	-1.2 V
3	$V_{Il(max)}$	1.5 V	0.8 V	-1.4 V
4	$V_{OH(min)}$	4.95 V	2.7 V	-0.9 V
5	$V_{OL(max)}$	0.005 V	0.4 V	-1.7 V
6	High level noise margin	V_{NH}=1.45 V	0.4 V	0.3 V
7	Low level noise margin	V_{NL}=1.45 V	0.4 V	0.3 V
8	Noise immunity	Better than TTL	Less than CMOS	More vulnerable to noise
9	Propagation delay	70 ns	10 ns	500 ps
10	Switching speed	Less than TTL	Faster than CMOS	Fastest
11	Power dissipation per gate	0.1 mW	10 mW	25 mW
12	Speed power product	0.7 pJ	100 pJ	0.5 pJ
13	Fan-out	50	10	25
14	Power supply voltage	3-15 V	Fixed 5 V	-4.5 to 5.2 V
15	Power dissipation	Increase with frequency	Increase with frequency	Constant with frequency
16	Application	Portable instrument where battery supply is used.	Laboratory instruments	High speed instruments.

35. Prove that ABC+ABC'+AB'C+A'BC=AB+AC+BC. *(Apr'18)*

ABC+ABC'+AB'C+A'BC = AB (C+C') + AB'C+A'BC
= AB+AB'C + A'BC
= A (B+B'C) + A'BC
=A (B+C) + A'BC
= AB+AC +A'BC
=B (A+C) + AC
=AB+BC+AC
=AB +AC+ BC ….Proved

36. Convert the given expression in canonical SOP form Y=AC+AB+BC. *(Apr'18)*

Y = AC+ AB +BC
=AC (B+B') + AB (C+C') + (A + A') BC
=ABC+ABC'+AB'C+AB'C'+ABC+ABC'+ABC
=ABC+ABC'+AB'C+AB'C' [A+A=1]

37. Designing a 4 bit Adder subtractor circuit. *(Apr'18)*

The instruction I was given for the design portion are as follows:

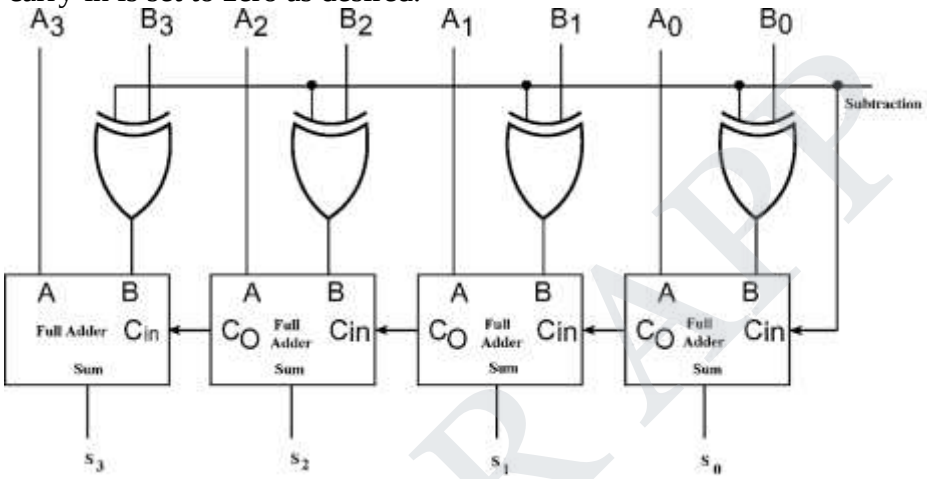
- Given two 4 bit positive binary numbers A and B, you are to design an adder/subtractor circuit

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

to compute (A+B) or (A-B), depending upon mode input which controls the operation.

- You may use one's or two's compliment of B to perform subtraction.
- The result with the proper sign is displayed in un-complemented binary form.
- My approach is to use four full-adders with a 4-bit input A, and a 4-bit input B whose bits may be XOR'd based on the mode chosen.
- The mode will be decided by bit M in the circuit below.
- For subtraction M = 1. 1 is chosen because M acts as the carry-in.
- Therefore, all bits of B will be inverted and 1 will be added to the LSB to find the 2's complement. For addition, M = 0.
- Therefore, carry-in is set to zero as desired.



- If the inputs A and B are unsigned, the answer will give A - B if A >= B OR the 2's complement of (B-A) if A < B.
- In this case, I would have to compare the inputs to see which one is larger and in cases where B is larger than A, take the 2's complement of the answer to show the positive value and turn on a bit to show that it is negative.
- If the inputs A and B are signed, the range of values I could use are from 0 - 7 and the result will give signed A - B as long as there is no overflow.
  0101 - 5 0111- 7
  1010- 6 1101- 3
  1111- 1 10100→ 0100 4
- The problem with using signed inputs and signed outputs are that the instructions ask for the answer to have the proper sign (indicated here by the MSB) but UNCOMPLEMENITED.
- My interpretation of uncomplemented is for the answer to be unsigned.
- I would have to therefore design a method, as in the case above to change 1111 (-1) to unsigned.
- Here is where I am stumped, as then the answer would have to appear as 10001. The MSB indicating it's negative and the other 4 bit's indicating the value.

ANNA UNIVERSITY QUESTIONS

PART A

1. Determine $(377)_{10}$ in Octal and Hexa-Decimal equivalent. *[Nov'14]*
2. Convert $(115)_{10}$ *and* $(235)_{10}$ to hexadecimal numbers. *(Nov'17)*
3. Convert:
   $(475.25)_8$ to its decimal equivalent.
   $(549.B4)_{16}$ to its binary equivalent. *[Apr'15]*
4. Convert 143_{10} into its binary and binary coded decimal equivalent. *[May'17]*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

5. What are non-weighted codes? Give example.*[Nov'13]*
6. What is a unit distance code? Give an example.*[Nov'15]*
7. What is a gray code and mention its advantages. *(Nov'17)*
8. Convert the following binary code into a Gray code, 1010111000_2. *[May'16]*
9. Convert the following Excess-3 numbers into decimal numbers. *[Nov'16]*
10. What is propagation delay? *[Apr'15]*
11. State the advantages of CMOS Technologies. *[May'13]*
12. Compare the totem pole output with open collector output?*[Nov'14]*
13. State the important characteristics of TTL family.*[Nov'10]*
14. In which type of TTL gate wired AND logic is possible? *[Nov'10]*
15. Which IC family offers (a) low propagation delay, and (b) low power dissipation? *[Apr'10]*
16. Why should we take care while using CMOS devices? *[Nov'11]*
17. Construct OR gate and AND gate using NAND gates.*[Nov'16]*
18. Reduce a(b+bc')+ab'.*[May'17]*
19. Reduce A(A+B) *(Apr'18)*
20. State the associative property of Boolean algebra. *(Apr'18)*

PART B & C

1. Convert $FACE_{16}$ into its binary, octal and decimal equivalent. *[May'17]*
2. Convert 23.625_{10} to octal (base 8). *[Nov'16]*
3. Convert 1010111011101100_2 into its octal, decimal and hexadecimal equivalent. *[Apr'16]*
4. Perform the following addition using BCD and Excess-3 addition (205+569)*[Apr'15]*
5. Perform the following operation $(756)_8 - (437)_8 + (725)_{16}$. Express the answer in octal form.*[Nov'15]*
6. Given the two binary numbers X=1010100 and Y= 1000011, perform the subtraction Y-X by using 2's complements. *[Nov'16]*
7. Explain in detail about the classification of binary codes.*[Nov'15]*
8. Explain in detail about error detecting and error correcting coding. *[Nov'17]*
9. Explain Hamming code with an example. State its advantages over parity codes. *[Nov'14]*
10. Given that a frame with bit sequence 1101011011_2 is transmitted, it has been received as 1101011010_2. Determine the method of detecting the error using any one error detecting code. *[Nov'14]*
11. A 12-bit Hamming code word containing 8 bits of data and 4 parity bits is read from memory. What was the original 8-bit data word that was written into memory if the 12-bit word read out is as (1) 101110010100 and (2) 111111110100? *[Nov'15]*
12. Encode the binary word 1011 into seven bit even parity Hamming code.*[Apr'15]*
13. Explain in detail the usage of Hamming codes for error detection and error correction with an example considering the data bits as 0101_2. *[Nov'16]*
14. Deduce the odd parity hamming code for the data: 1010_2. Introduce an error in the LSB of the hamming code and deduce the steps to detect the error. *[Apr'16]*
15. Design a odd parity hamming code generator and detector for 4 bit data and explain their logic. *[May'17]*
16. Explain the concept of working of RTL logic families. *[Nov'17]*
17. Explain the working of DTL logic families. *[Nov'17]*
18. Explain the concept, working and characteristics of TTL logic families. *[Nov'10]*
19. Design a TTL logic circuit for a 3 input NAND gate. *[Nov'14]*
20. With circuit schematic explain the working of a two-input TTL NAND gate. *[May'17]*
21. With circuit schematic explain the operation of a two input TTL NAND gate. *[Apr'16]*

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

22. With circuit Schematic, Explain the operation of a two input TTL NAND gate with totem-pole output. *[Apr'15]*
23. Explain in detail about TTL with open collector output configuration. *[Nov'13, Nov'17]*
24. Compare the Totem pole and open collector outputs. *[Apr'15][May'17]*
25. Demonstrate the CMOS logic circuit configuration and characteristics in details. *[Nov'13]*
26. Explain the characteristics of CMOS. *[Nov'11]*
27. Draw the MOS logic circuit for NOT gate and explain its operation. *[Nov'14]*
28. Draw the CMOS logic circuit for NOR gate and explain its operation. *[Nov'15]*
29. Explain with an aid of circuit diagram the operation of 2 input CMOS NAND gate and list out its advantages over other logic families. *[Nov'16, Apr'18]*
30. Explain in detail about ECL family. OR With circuit schematic and explain the operation and characteristics of a ECL gate. *[Apr'16, Nov'17]*
31. Compare the characteristics of TTL, ECL and CMOS logic families.*[Apr'10]*
32. Prove that ABC+ABC$^{'}$+AB$^{'}$C+A$^{'}$BC=AB+AC+BC. *(Apr'18)*
33. Convert the given expression in canonical SOP form Y=AC+AB+BC. *(Apr'18)*
34. Designing a 4 bit Adder subtractor circuit. *(Apr'18)*

STUCOR APP

STUCOR APP
DOWNLOADED FROM STUCOR APP

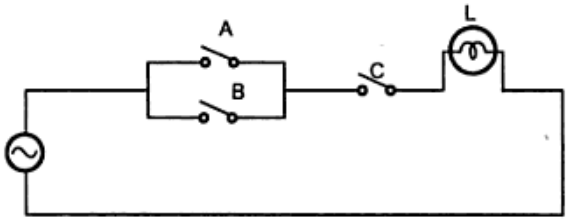
DOWNLOADED FROM STUCOR APP

UNIT II PART A

1. What are combinational Logic Circuits?

- Combinational Logic Circuits are made up from basic logic NAND, NOR or NOT gates that are "combined" or connected together to produce more complicated switching circuits.
- These logic gates are the building blocks of Combinational Logic Circuits.

2. Explain the following switching circuits in binary logic notation. *[May' 12]*



Ans: Y = (A+B).C

3. Which gates are called as the Universal gates? What are it advantages? (OR) **Why NAND and NOR are called Universal Gates?***[Nov'13, Nov'10]*

- TheNANDandNORgatesarecalledastheUniversalgates.Thesegatesareusedtoperform any type of logic application.

4. What do you mean by literal?

- In Boolean function, the total no of variables in complemented or un-complemented form are called literal.

  Eg. A+BC' contains 3 literals

5. Write the Boolean function of an XOR gate give its truth table?

Input		Output
A	B	Y= A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

6. State the absorption law of Boolean algebra.

- The absorption law of Boolean algebra is given by

  X+XY=X and X (X+Y) = X

7. Explain consensus theorem in Boolean algebra.

- In simplification of Boolean expression the redundant term in an expression can be eliminated to form the equivalent expression.
- This theorem used for simplification is called consensus theorem.

AB+A'C+BC = AB+A'C

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

8. Simplify A+AB+A+B

A+AB+A'+B =A (1+B) +A'+B [1+B=1]
= A+A'+B [A+A'=1]
= 1+B
A+AB+A'+B = 1+B

9. State De Morgan's theorem. *[Nov'09, May'10, Nov'10, Apr'11, Nov'12, May'13]*

De Morgan suggested two theorems that form important part of Boolean algebra. They are,

(1) The complement of a product is equal to the sum of the complements.

(A.B)'=A'+B'

(2) The complement of a sum term is equal to the product of the complements.

(A+B)'=A'B'

10. Apply De Morgan's theorem to the following expression (A+B+C+D).

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

11. What are the applications of De Morgan's Theorem?

- Convert max term-to-min term or min term-to-max term form.
- Examine two different logic diagrams that perform same logic function.

12. Simplify the following using De Morgan's theorem [((AB)'C)'' D]'

$\overline{[(\overline{\overline{(\overline{AB})C}})\,D]} = (\overline{\overline{(\overline{AB})C}}) + \bar{D}$ Since $[(\overline{AB}) = \bar{A}+\bar{B}]$
$= (\overline{(\overline{AB})C}) + \bar{D}$
$= (\overline{AB}) + \bar{C} + \bar{D}$
$= \bar{A} + \bar{B} + \bar{C} + \bar{D}$

13. What are the methods adopted to reduce Boolean function?

- Algebraic Method
- Karnaugh map
- Tabulation method or Quine Mc-Cluskey method

14. Reduce A.A'C

A.A'C = 0.
C = 0 [A.A' = 0]

15. Reduce A (A + B)

A (A + B) = AA + AB
= A (1 + B) [1 + B = 1]
= A

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

16. Reduce $\overline{AB\,C} + \overline{A}\,B\,\overline{C} + \overline{A}\,BC$

$$\overline{AB}\,\overline{C} + \overline{A}\,B\,\overline{C} + \overline{A}\,BC = \overline{AC}\left(B + \overline{B}\right) + \overline{A}\,BC$$
$$= \overline{AC} + \overline{A}BC$$
$$= \overline{A}\left[\overline{C} + BC\right]$$
$$= \overline{A}\left[\overline{C} + B\right] \qquad [A + A'B = A + B]$$

17. Find the complement of the functions F1 = x'yz' + x'y'z and F2 = x (y'z' + yz).

By applying De-Morgan's theorem.

F1' = (x'yz' + x'y'z)' = (x'yz')'(x'y'z)'
= (x + y' + z) (x + y +z')

F2' = [x(y'z' + yz)]'
= x' + (y'z' + yz)'
= x' + (y'z')'(yz)'
= x' + (y + z) (y' + z')

18. Simplify the following expression Y = (A+B) (A+C) (B+C)

(A+B) (A+C) (B+C) = (AA+AC+AB+BC) (B+C)
= (A+AC+AB+BC) (B+C)
= (AB+ABC+ABB+BBC) C
= (AB+ABC+AB+BC) C
= (ABC+ABCC+ABC+BCC)
= (ABC+ABC+ABC+BC)
= ABC +BC
= BC [1+A]
= BC

19. If A and B are the Boolean variables & if A=1 & (A+B)'=0. Find B.

(A+B)' = 0
(A+B)' = A'B'
A'B'= 0 we know that A' =0
B = 0 or 1

20. Simplify (A+AB+A'+B).

A+AB+A'+B = A (1+B) + A'+B
= A+A'+B
= 1+B
=1

21. Simplify the expression Z = AB + AB'. (A'C')'. *[Apr'15]*

Z = AB + AB'. (A'C')'
= AB + AB'. (A''C'') [De Morgan's Law]
= AB + AB'. (A +C)
= AB + (A.A) B'+ AB'C
= AB + AB'+ AB'C [A.A =A]

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

= AB + AB' [1+C]
= AB + AB' [A+1= 1]
= A [B+B']
= A [A+A'=1]

22. Find the compliment of F = x (y'z'+yz). *[Nov'12]*

F = [x (y'z'+yz)]'
= x'+ [y'z'+yz]'
= x'+ [(y'z')' (yz)']
= x'+ [(y'' + z'') (y' + z')]
=x'+ [(y + z) (y' + z')]
= x' + [yy' + yz' + zy' + zz']
= x' + [y⊕ z]

23. Show that a) A+A'B=A+B and b) X'Y'Z+X'YZ+XY' = X'Z+XY' *[Apr'10]*

a) A+A'B = A+AB+A'B since A+AB=A
= A+B (A + A')
= A+B

b) X'Y'Z+X'YZ+XY' = X'Z (Y'+Y) +XY'
= X'Z (1) + XY'
= X'Z+XY'

24. Define Minterm & Maxterm.

- The products of Boolean expression where all possible variables appear once in complement to run-complement variables are called **Minterm.** eg. A'B'C+AB'C
- A sum terms in a Boolean expression where all possible variables appearance, in complement or un-complement form are called **Maxterm.** eg. (A+B+C')(A+B+C)

25. What is meant by karnaugh map or K-Map method? *(Nov'17, Apr'18)*

- A karnaugh map or k-map is a Pictorial form of truth table, in which the map diagram is made up of cells, with each cell representing one minterm or maxterm of the function.
- This method provides a simple straight forward procedure for minimizing Boolean function.

26. What are don'tcare conditions? How does it help K-map for circuit simplification? *[Nov'11]*

- In some logic circuits certain input conditions never occur, therefore the corresponding output never appears.
- In such cases the output level is not defined, it can be either high or low.
- These output levels are indicated by 'X' or 'd' in the truth tables and are called **don't care conditions** or incompletely specified functions

27. What is tabulation method?

- A method involving an exhaustive tabular search method for the minimum expression to solve a Boolean equation for more variables is called as a tabulation method.

28. What is a prime implicant?

- A prime implicant is a product term obtained by combining the maximum possible number of adjacent squares in the map.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- They cannot be reduced further.
- A prime implicant is a group of minterms which cannot be combined with any other minterm or groups.

29. What is an essential prime implicant?

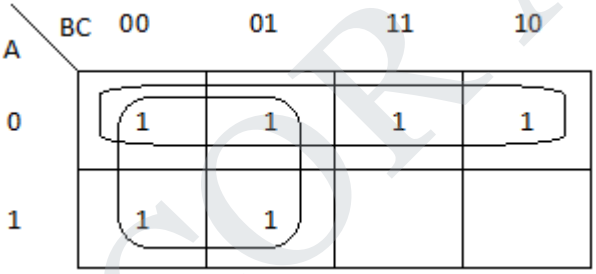
- The Essential Prime Implicant is a prime implicant in which one or more minterms are unique.
- It contains atleast one minterm which is not contained in any other prime implicant.

30. Define Duality Theorem. [OR] **Define Duality Property.** *(Apr'18)*

- The Duality theorem states that starting with a Boolean relation we can derive another Boolean relation by:
  i).Changing OR (operation) i.e., + (Plus) sign to an AND (operation) i.e., · (dot) and Vice-versa.
  ii).Complement any 0or 1 appearing in the expression i.e. replacing contains 0 and 1 by 1 and 0 respectively.

31. Given F = B' + A'B +A'C': Identify the redundant term using K-Map. *[Nov'14]*

F = B' + A'B +A'C'
= (A+A') B' (C+C') + A'B (C+C') + A' (B+B') C'
= AB'C' + A'B'C' + AB'C + A'B'C+ A'BC+A'BC'



F = A'+B'

32. Express F=A+BC' as sum of minterms.

A+BC' = A (B+B') (C+C') + (A+A') (B'C)
= (AB+AB') (C+C') +AB'C+A'B'C
= ABC+AB'C+ABC'+AB'C'+AB'C+A'B'C
F= ∑m (1, 4, 5, 6, 7)

33. Convert the given expression in canonical SOP form Y = AC + AB + BC. *[Apr'15]*

Y = AC + AB + BC = AC (B + B') + AB (C + C') + (A + A') BC
= ABC + ABC' + AB'C + AB'C' + ABC + ABC' + ABC
= ABC + ABC' +AB'C + AB'C'

34. Convert the given expression in canonical SOP form Y = AB + A'C + BC'. *[Nov'15]*

Y = AB + A'C + BC'
= AB (C+C') + A' (B + B') C + (A + A') BC'
=ABC+ ABC' + A'BC+ A'B'C + ABC' + A'BC'
= ABC + ABC' + A'BC + A'B'C + A'BC'
= 1 1 1 + 1 1 0 + 0 1 1 + 0 0 1 + 0 1 0
= (7, 6, 3, 1, 2)
= ∑m (1, 2, 3, 6, 7)

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

35. Convert the given expression in canonical SOP form $Y = AB + \bar{A}\,C + B\,\bar{C}$ *(Nov'16)*

$Y = AB + \bar{A}\,C + B\,\bar{C}$

$= AB(C + \bar{C}) + \bar{A}(B + \bar{B})C + (A + \bar{A})B\,\bar{C}$

$= ABC + AB\bar{C} + \bar{A}\,BC + \bar{A}\bar{B}C + AB\,\bar{C} + \bar{A}B\,\bar{C}$

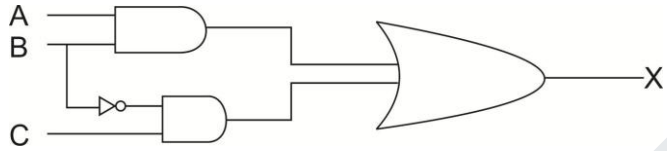
$= ABC + AB\bar{C} + \bar{A}\,BC + \bar{A}\bar{B}C + \bar{A}B\,\bar{C}$

= 111, 110, 011, 001, 010

= 7, 6, 3, 1, 2

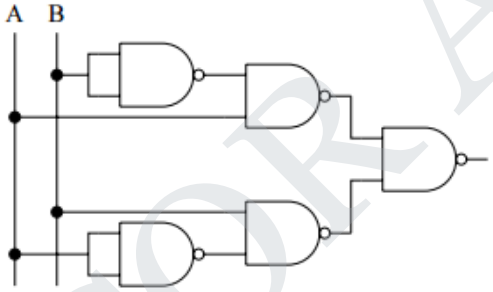
Y (A, B, C) = ∑m (1, 2, 3, 6, 7)

36. Draw the logic diagram for X = AB+BC



37. Draw the logical diagram of EX – OR gate using NAND gates. *[Nov'15]*

EX – OR $= A \oplus B = A\bar{B} + \bar{A}\,B$



38. Implement F = (AB'+A'B) (C+D') with only NOR gate.

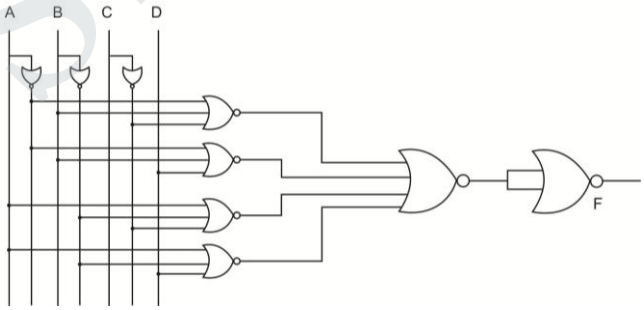
$F = (A\overline{B} + \overline{A}B)(C + \overline{D})$

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

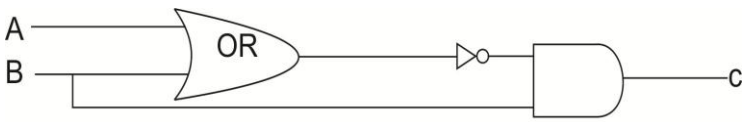
$\bar{\bar{F}} = \overline{\overline{A\overline{B}C + A\overline{B}\overline{D} + \overline{A}BC + \overline{A}B\overline{D}}}$

$= \overline{A\overline{B}C.A\overline{B}\overline{D}.\overline{A}BC.\overline{A}B\overline{D}}$

$= \overline{(\overline{A} + B + \overline{C})(\overline{A} + B + D)(A + \overline{B} + \overline{C})(A + \overline{B} + D)}$



39. Write the Boolean expression for the output of the system shown.



STUCOR APP
DOWNLOADED FROM STUCOR APP

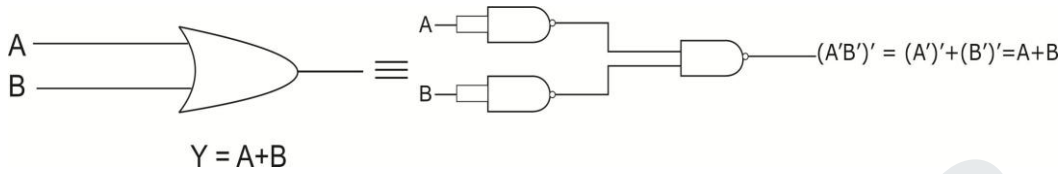
DOWNLOADED FROM STUCOR APP

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

$C = \overline{A} \cdot \overline{B} \cdot B$

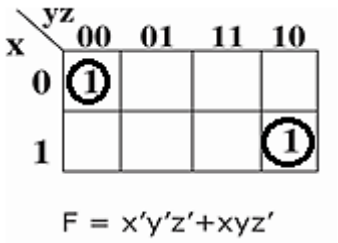
$C = 0$

40. Realize an OR gates using NAND gates? *[Apr'10, Nov'12, May'13]*

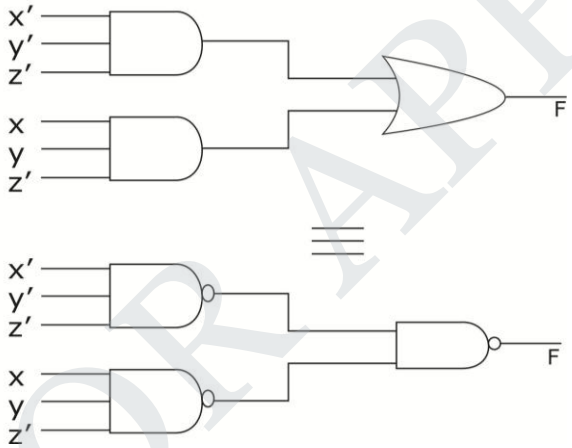


Y = A+B

41. Implement the given function using NAND gates F(x, y,z) =∑m (0, 6).



F = x'y'z'+xyz'



42.Write the POS representation of the following SOP function.*(May'16)*

F(x,y,z)=∑m (0,1,3,5,7)

Let the SOP functions will ∑m (0, 1, 3, 5, 7)

Now taking left out values πM (2, 4, 6)

F(x,y,z) = πM (2, 4, 6)

= (1 + 0 + 1) (0 + 1 + 1) (0 + 0 + 1)

$= (x + \bar{y} + z)(\bar{x} + y + z)(\bar{x} + \bar{y} + z)$

43. Write the POS representation of the following SOP function$F(x, y, z) = \bar{x}yz + xy\bar{z} + x\bar{y}z$

(May'17)

$F(x, y, z) = \bar{x}yz + xy\bar{z} + x\bar{y}z$

= 011, 110, 101

= 3, 6, 5

F(x, y, z) = ∑m (3, 5, 6)

Let the remaining term for POS,

F(x, y, z) = πM (0, 1, 2, 4, 7)

= (1+1+1) (1+1+0) (1+0+1) (0+1+1) (0+0+0)

$F(x, y, z) = (x + y + z)(x + y + \bar{z})(x + \bar{y} + z)(\bar{x} + y + z)(\bar{x} + \bar{y} + \bar{z})$

STUCOR APP

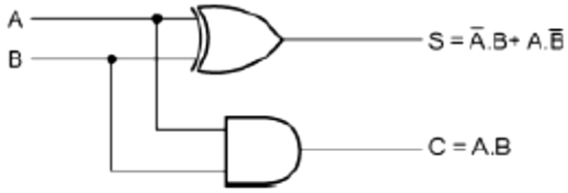
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

44. Define half adder and full adder

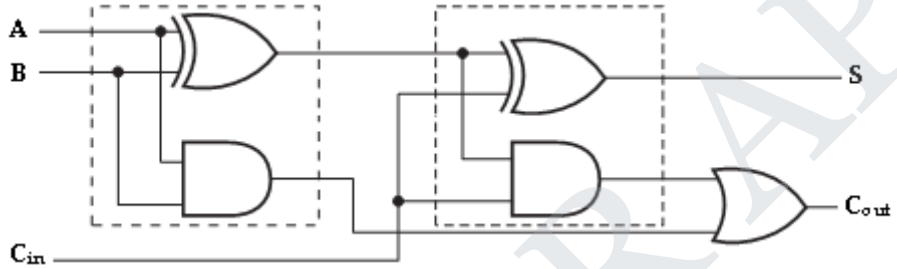
Half Adder: The logic circuit that performs the addition of two bits is a half adder.
Full Adder: The circuit that performs the addition of three bits is a full adder.

45. Sketch a half-adder using logic gates. Draw the truth table. *[Apr'10, Nov'12]*

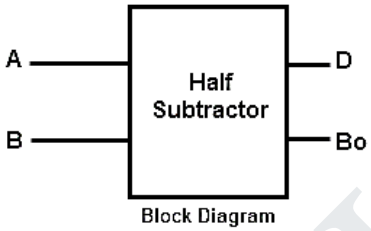


A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

46. Implement full adder with two half adders. *[Nov'12]*



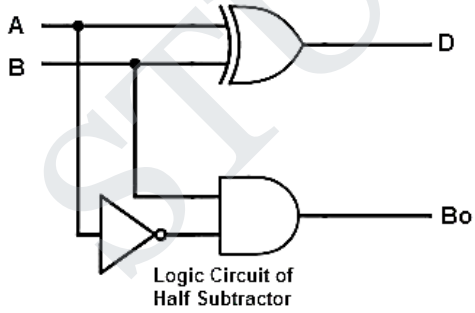
47. Design a half subtractor. *[May'16, May'17]*



Block Diagram

A	B	D	B_o
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Truth Table



Logic Circuit of Half Subtractor

48. What is code converter?

- It is a circuit that makes the two systems compatible even though each uses a different binary code.
- It is a device that converts binary signals from a source code to its output code. Example is a BCD to Ex-3 converter.

49. What do you mean by comparator?

- It is a special combinational circuit designed primarily to compare the relative magnitude of two binary numbers.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- An n bit operator receives two n bit numbers A and B outputs, A>B, A=B, A<B.
- As per the magnitude of two numbers, one of the outputs will be high.

50. Define Multiplexer. *[May'13]*

- A multiplexer or MUX is a device that performs multiplexing.
- It selects one of many analog or digital input signals and forwards the selected input into a single line.
- A multiplexer of 2n inputs has n select lines, which are used to select which input line to send to the output.

51. Why MUX is called as Data Selector? *[Apr'11]*

- Multiplexer is a digital switch.
- If allows digital information from several sources to be routed onto a single output line.
- Multiplexer is otherwise called as Data selector.

52. Draw the truth table of 2:1 MUX. *(Nov'16)*

Select	Inputs		Output
0	0	0	0
0	0	1	1
1	1	0	1
1	1	1	1

53. What is Demultiplexer?

- A Demultiplexer is a circuit that receives information on a single line and transmits this information on one of 2" possible output lines

54. Define Encoder?

- An encoder has 2" input lines and n output lines.
- In encoder the output lines generate the binary code corresponding to the input value.

55. What is priority encoder? *[May'12]*

- A priority encoder is an encoder circuit that includes the priority function.
- In priority encoder, if 2 or more inputs are equal to 1 at the same time, the input having the highest priority will take precedence.

56. Define Decoder.

- A decoder is a multiple - input multiple output logic circuit that converts coded inputs into coded outputs where the input and output codes are different.

57. What is binary decoder?

- A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2" outputs lines.

.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

58. Difference between Decoder & Demux. *[Nov'11] [OR]* **Compare decoder and demultiplexer.** *(Nov'17)*

S.No	Decoder	Demux
1	Decoder is a many inputs to many outputs device.	Demux is a single input to many outputs.
2	There are no selection lines.	The selection of specific output line is controlled by the value of selection lines.

59. Give one application each for Multiplexer and Decoder. *[Nov'14]*

Multiplexer:

- Used as a Data selector.
- Used in telephone communication

Decoder:

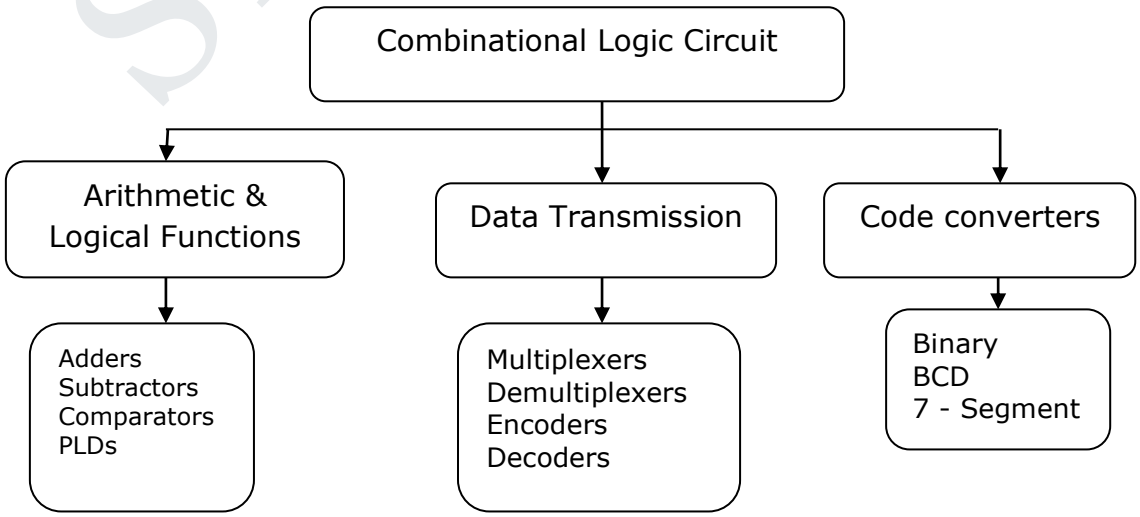
- Used to convert more data lines to less data lines.

PART B& C [8/16 Marks]

1. Write notes on Combinational Logic circuit and its classification?

- Combinational Logic Circuits are made up from basic logic NAND, NOR or NOT gates that are "combined" or connected together to produce more complicated switching circuits.
- These logic gates are the building blocks of Combinational Logic Circuits.
- Combinational logic circuits can be very simple or very complicated and any combinational circuit can be implemented with only NAND and NOR gates as these are classed as "universal" gates.
- The three main ways of specifying the function of a combinational logic circuit are:

  - Boolean algebra – This forms the algebraic expression showing the operation of the logic circuit for each input variable either True or False that result in a logic "1″ output.
  - Truth Table – A truth table defines the function of a logic gate by providing a concise list that shows all the output states in tabular form for each possible combination of input variable that the gate could encounter.
  - Logic Diagram – This is a graphical representation of a logic circuit that shows the wiring and connections of each individual logic gate, represented by a specific graphical symbol that implements the logic circuit.



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

2. Write short notes on Boolean algebra.

Boolean algebra:

- In working with logic relations in digital form, we need a set of rules for symbolic manipulation which will enable us to simplify complex expressions and solve for unknowns.
- In computer work it is used in addition to describe circuits whose state can be either 1 *(true)* or 0 *(false)*.
- Using the relations defined in the AND, OR and NOT operation, a number of

Postulates:

P1 : X = 0 or X = 1
P2 : 0 . 0 = 0
P3 : 1 + 1 = 1
P4 : 0 + 0 = 0
P5 : 1 . 1 = 1
P6 : 1 . 0 = 0 . 1 = 0
P7 : 1 + 0 = 0 + 1 = 1

Theorems

T1 :Commutative Law
(a) $A + B = B + A$
(b) $A \cdot B = B \cdot A$

T2 : Associative Law
(a) $(A + B) + C = A + (B + C)$
(b) $(A \cdot B) C = A (B \cdot C)$

T3 :Distributive Law
(a) $A (B + C) = A \cdot B + A \cdot C$
(b) $A + (B \cdot C) = (A + B) (A + C)$

T4 :Identity Law
(a) $A + A = A$
(b) $A \cdot A = A$

T5 :Negation Law
(a) $(\overline{A}) = \overline{A}$
(b) $(\overline{\overline{A}}) = A$

T6 :Redundance Law
(a) $A + A \cdot B = A$
(b) $A (A + B) = A$

T7 : (a) $0 + A = A$
(b) $1 \cdot A = A$
(c) $1 + A = 1$
(d) $0 \cdot A = 0$

T8 : (a) $\overline{A} + A = 1$
(b) $\overline{A} A = 0$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

T9 : (a) $A + \overline{A}\,B = A + B$

(b) $A\,(\overline{A} + B) = A\,B$

T10: De Morgan's Theorem

(a) $\overline{(A + B)} = \overline{A}\ \overline{B}$

(b) $\overline{(A\,B)} = \overline{A} + \overline{B}$

3. Show that $A + \overline{A}B = A + B$

$A + \overline{A}B = A.1 + \overline{A}B$ T7(a)

$= A\,(1 + B) + \overline{A}B$ T7(c)

$= A + AB + \overline{A}B$ T3(a)

$= A + B\,(A + \overline{A})$ T3(a)

$= A + B$ T8

Using the truth Table,

A	B	A+B	$\overline{A}\,B$	$A + \overline{A}\,B$
0	0	0	0	0
0	1	1	1	1
1	0	1	0	1
1	1	1	0	1

4. Prove that F= $\overline{A}$.B + A.$\overline{B}$is exclusive OR operation and it equals $\overline{\overline{(A.B)}.A}.\overline{\overline{(A.B)}.B}$ *[May'13]*

$\overline{\overline{(A.B)}.A}.\overline{\overline{(A.B)}.B} = \overline{(\overline{A} + \overline{B}).\ A}.\overline{(\overline{A} + \overline{B}) + B}$ By De Morgan's Theorem

$= \overline{(\overline{\overline{A} + \overline{B}) + \overline{A})}\,.\,(\overline{\overline{A} + \overline{B}}\ ) + \overline{B})}$ By De Morgan's Theorem

$= \overline{((\overline{\overline{A}.\overline{B}}) + \overline{A})\,.\,\left(\overline{\overline{A}.\overline{B}}\right) + \overline{B})}$ By De Morgan's Theorem

$= \overline{(AB + \overline{A})\,.\,(AB + \overline{B})}$ By De Morgan's Theorem

$= \overline{(AB + \overline{A})} + \overline{(AB + \overline{B})}$ By De Morgan's Theorem

$= (\overline{A} + B) + (A + \overline{B})$ By Theorem 5

$= (A + \overline{A}B = A + B)$

$= (\overline{\overline{A}.\overline{B}}) + (\overline{A}\,.\,\overline{\overline{B}}) = A\overline{B} + \overline{A}B$

F = $\overline{A}$.B + A.$\overline{B}$ = LHS. Hence the proof

5. Reduce the given expressions using Boolean algebra:

i) x'y'z'+x'y'z+x'yz+xy'z+xyz

ii) abc'+ab'c+a'bc+abc

iii) p'q'r+p'qr'+p'qr+pqr'+pq'r' *[May'11]*

i) x'y'z' + x'y'z + x'yz + xy'z + xyz

= x'y'(z' + z) + x'yz + x (y' + y)z

= x'y' + x'yz + xz

= x'(y' + yz) + xz

= x'(y' + z) + xz By ConsensesTheorem

= x'y' + x'z + xz

= x'y' +z (x'+ x)

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

= x'y' + z

ii) abc' + ab'c + a'bc + abc
= a(bc' + b'c + bc) + a'bc
= a[b(c'+c) + b'c] + a'bc
= a[b + b'c] + a'bc
= a[b + c] + a'bc ------- since [A+A'B = A+B]
= ab + ac + a'bc
= b[a + a'c] + ac
= b[a + c] +ac------- since [A+A'B = A+B]
= ab + bc + ac

iii) p'q'r + p'qr' + p'qr + pqr' + pq'r'
= p'q'r + p'q(r'+ r) + pr'(q + q')
= p'q'r + p'q + pr'
= p'(q'r + q) + pr'
= p'(q + r) + pr'-------- since [A+A'B = A+B]
= p'q + p' r + pr'

6. Simplify the following expression

i). AB + (AC)' + AB'C (AB + C)

ii). Y = (A + B)(A + C') (B' + C')

iii). (X + Y' + XY)(X + Y') (X'Y) = 0

i).AB + (AC)' + AB'C(AB + C) = AB + (AC)' + AAB'BC + AB'CC
= AB + (AC)' + AB'CC [A.A' = 0]
= AB + (AC)' + AB'C [A.A = A]
= AB + A' + C' +AB'C [(AB)' = A' + B']
= A' + B + C' + AB'C [A + AB' = A + B]
= A' + B'C + B + C' [A + A'B = A + B]
= A' + B + C' + B'C
= A' + B + C' + B'
= A' + C' + 1
= 1

ii). Y = (A + B) (A + C') (B' + C')
= (A + B) (A + C') (B' + C')
= (AA + AC' +AB +BC') (B' + C')
= (A+ AC' +AB +BC') (B' + C')
= (A (1+ C' + B) + BC') (B' + C')
= (A (1) + BC') (B' + C') [1 + A = 1]
= (A + BC') (B' + C')
= AB' + AC' + BB'C' + BC'C'
= AB' + AC' + BC' [A.A' = 0, A.A = A]

iii). (X + Y' + XY) (X + Y') (X'Y) = 0
LHS = (X + Y' + XY) (X + Y') (X' Y)
= (X + Y') (XX'Y+ X'YY') [A + A'B = A + B]

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

= (X + Y') (0 + 0) [A.A' = 0]
= 0
= RHS

7. Prove that the expression ABC + ABC' + AB'C + A'BC = AB + AC + BC

ABC + ABC' + AB'C + A'BC
= AB(C + C') + AB'C + A'BC
= AB + AB'C + A'BC
= A (B + B'C) + A'BC
= A (B + C) + A'BC [A + A'B = A + B]
= AB + AC + A'BC
= AB + A'BC + AC
= B (A + A'C) + AC
= B (A + C) + AC [A + A'B = A + B]
= AB + BC + AC……..Proved

8. Explain the functions of logic circuits. [OR] **Write down the steps in implementing a Boolean function with levels of AND gates.** *(Apr'18)*

- Any decision that can be answered yes/no or true/false can be mathematically represented as a combination of logic functions.
- The 3 basic logic functions, which can be used to solve any Boolean equation, are: NOT, AND, OR
- Other common logic functions, that are combinations of the basic 3, are: NAND, NOR, XOR

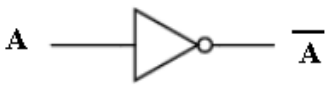
NOT FUNCTION:

- The NOT gate is an electronic circuit that produces an inverted version of the input at its output.
- It is also known as an inverter.
- If the input variable is A, the inverted output is known as $\overline{A}$.
- Boolean Logic Equation will be $X = \overline{A}$ or $\overline{X} = A$

Truth Table:

Input A	Output $\overline{A}$
0	1
1	0

Logic Symbol:



AND FUNCTION:

- The AND gate is an electronic circuit that gives a high output (1) only if all its inputs are high.
- A dot (.) is used to show the AND operation i.e. A.B or AB.
- Boolean Logic Equation will be X = AB or X = A · B

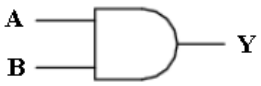
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Truth Table:

Input A	Input B	Output Y
0	0	0
0	1	0
1	0	0
1	1	1

Logic Symbols:



OR FUNCTION:

- The OR gate is an electronic circuit that gives a high output (1) if one or more of its inputs are high.
- A plus (+) is used to show the OR operation.
- Boolean Logic Equation X = A + B

Truth Table:

Input A	Input B	Output Y
0	0	0
0	1	1
1	0	1
1	1	1

Logic Symbols:

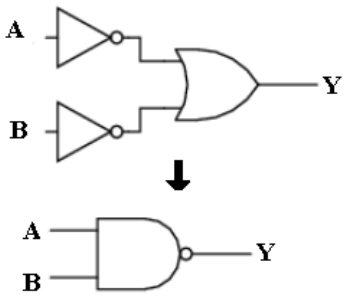
NAND FUNCTION:

- This is a NOT-AND gate which is equal to an AND gate followed by a NOT gate.
- The outputs of all NAND gates are high if any of the inputs are low.
- The symbol is an AND gate with a small circle on the output.
- The small circle represents inversion.
- Boolean Logic Equation $Y = \overline{AB}$ *or* $Y = (AB)'$

Truth Table:

Input A	Input B	Output Y
0	**0**	1
0	1	1
1	0	1
1	1	0

Logic Symbols:



NOR FUNCTION:

- This is a NOT-OR gate which is equal to an OR gate followed by a NOT gate.
- The outputs of all NOR gates are low if any of the inputs are high.
- The symbol is an OR gate with a small circle on the output.
- The small circle represents inversion.
- Boolean Logic Equation $Y = \overline{A + B}$ *or* $Y = (A + B)'$

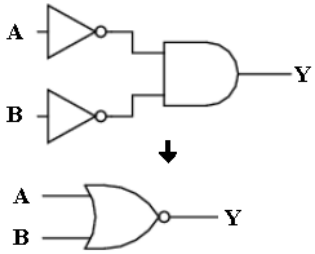
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Truth Table:

Input A	Input B	Output Y
0	**0**	1
0	1	0
1	0	0
1	1	0

Logic Symbols:



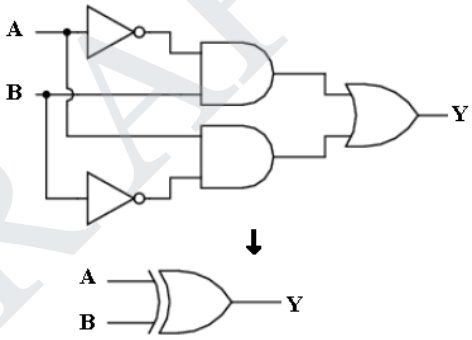
XOR FUNCTION:

- The 'Exclusive-OR' gate is a circuit which will give a high output if either, but not both, of its two inputs are high.
- An encircled plus sign (⊕) is used to show the EOR operation.
- Boolean Logic Equation $Y = A\,.\,\overline{B} + \overline{A}\,.\,B$ *or* $(A + B)\overline{(AB)}$ *or* $A \oplus B$

Truth Table:

Input A	Input B	Output Y
0	0	0
0	1	1
1	0	1
1	1	0

Logic Symbols:



XNOR FUNCTION:

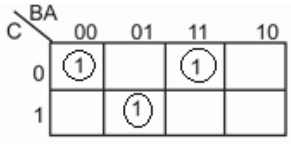
- The '**Exclusive-NOR**' gate circuit does the opposite to the EOR gate.
- It will give a low output if either, but not both, of its two inputs are high.
- The symbol is an EXOR gate with a small circle on the output.
- The small circle represents inversion.
- Boolean Logic Equation $Y = AB + \overline{AB}$ *or* $\overline{(A \oplus B)}$

Truth Table:

Input A	Input B	Output Y
0	**0**	1
0	1	0
1	0	0
1	1	1

Logic Symbols:

9. Simplify and draw the logic diagram for the expression shown below:
Y=C'B'A'+C'BA+CB'A

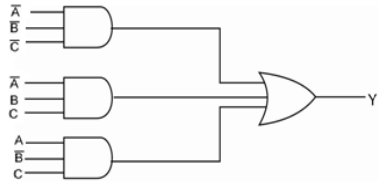


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Simplification is not possible. So $Y = \overline{A}\overline{B}\overline{C} + \overline{A}BC + A\overline{B}C$

Logic diagram:



10. Reduce the following expression using K-map f = x'y'z + w'xz + wxyz' + wxz + w'xyz.

[May'11]

f = x'y'z+w'xz+wxyz'+wxz+w'xyz

= (w+w')x'y'z + w'x(y+y')z + wxyz' + wx(y+y')z + w'xyz

= wx'y'z + w'x'y'z + w'xyz + w'xy'z + wxyz' + wxyz + wxy'z + w'xyz

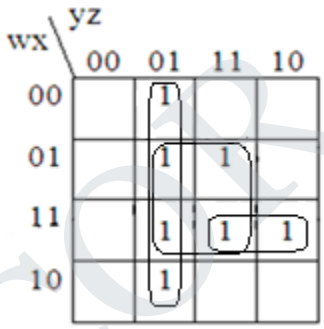
= wx'y'z + w'x'y'z + w'xyz + w'xy'z + wxyz' + wxyz + wxy'z

= 1001 + 0001 + 0111 + 0101 + 1110 + 1111 + 1101

= $m_9 + m_1 + m_7 + m_5 + m_{14} + m_{15} + m_{13}$

= Σm(1, 5, 7, 9, 13, 14, 15)

K-map:

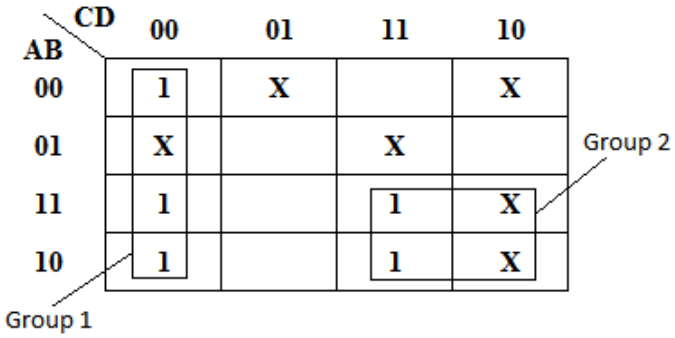


$f = wzy + xz + \overline{y}z$

11. Simplify the Boolean function using K – map and implement using only NAND gates. F (A, B, C, D) = ∑m (0, 8, 11, 12, 15) + ∑d (1, 2, 4, 7, 10, 14). Mark the essential and non – essential prime implicants. *[Nov'15]* [OR] **Give the general procedure for converting a Boolean expression in to multilevel NAND diagram.** *(Apr'18)*

F (A, B, C, D) = ∑m (0, 8, 11, 12, 15) + ∑d (1, 2, 4, 7, 10, 14)

K- Map:



Group 1: $\overline{C}\,\overline{D}$

Group 2: A C

F (A, B, C, D) = $\overline{C}\,\overline{D} + AC$

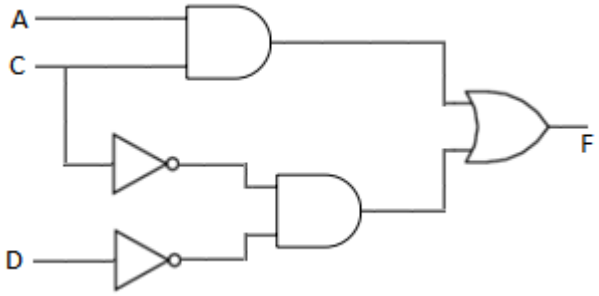
Both the values are essential prime implicants.

STUCOR APP
DOWNLOADED FROM STUCOR APP

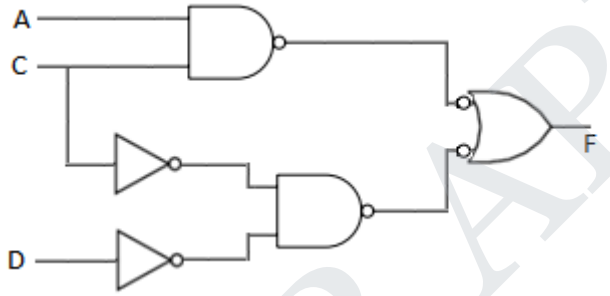
DOWNLOADED FROM STUCOR APP

Implementation:

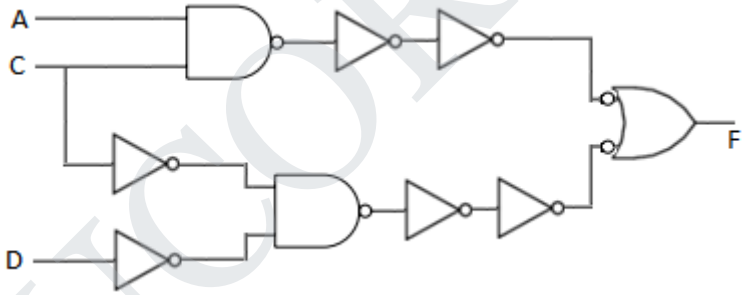
Step 1: Implementation of circuits using basic elements



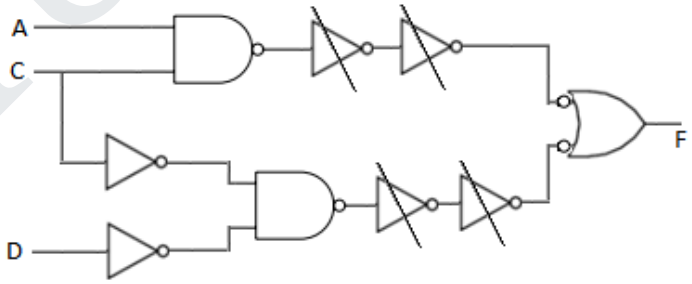
Step 2: Adding bubble after the AND gate and before the OR gate.



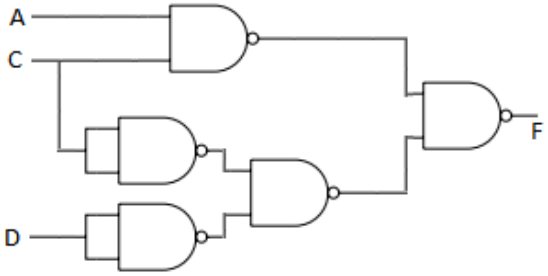
Step 3: Adding NOT gate where bubble is inserted.



Step 4: Cancel NOT gates if two NOT gates present in a same line.



Step 5: Convert all the gates to NAND gates



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

12. Simplify and implement the following SOP function using NOR gates
f(A,B,C,D) = ∑ m(0,1,4,5,10,11,14,15) *[May'12]*

Step 1: Covert SOP function into its equivalent POS function.

∑ m(0,1,4,5,10,11,14,15) = ΠM (2,3,6,7,8,9,12,13)

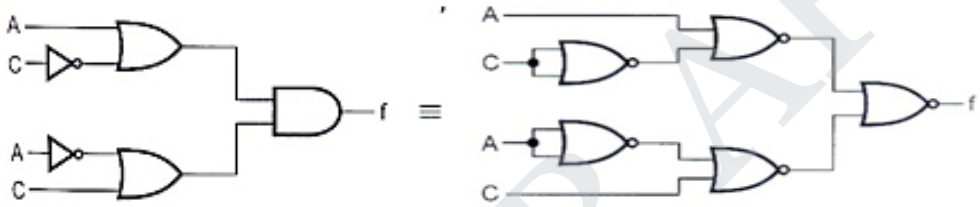
Step 2: K-map simplification:

AB \ CD	C+D	C+D	C+D	C+D
A+B			0	0
A+$\bar{B}$			0	0
$\bar{A}$+$\bar{B}$	0	0		
$\bar{A}$+B	0	0		

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

Step 3: Implementation:

Using basic gates ≡ Using NOR gates



13. Convert SOP to Equivalent POS:

i) A'B'C+A'B'C+A'BC+AB'C+ABC.
ii). (A+B) (A+C) (B+C')
iii) F=XY+X'Z

i) A'B'C+A'B'C+A'BC+AB'C+ABC
= A'B'C+A'B'C+A'BC+AB'C+ABC
= A'B'C+A'BC+AB'C+ABC
= (A+B+C') (A+B'+C') (A'+B+C') (A'+B'+C')

ii). (A+B) (A+C) (B+C')
= [(A+B) + (C.C')] [(A+ (B.B') + C)] [(A.A') + (B+C')]
= [(A+B+C) (A+B+C') (A+B +C) (A+B'+C) (A+B+C') (A'+B+C')]
= [(A+B+C) (A+B+C') (A+B'+C) (A'+B+C')]

iii). F = XY+X'Z
= XY (Z+Z') + X' (Y+Y') Z
= XYZ+XYZ'+X'YZ+X'Y'Z
= 1 1 1 + 1 1 0 + 0 1 1 + 0 0 1
= 7, 6, 3, 1
= ∑m (1, 3, 6, 7)
= ∏ M (0, 2, 4, 5)
= (1 + 1 + 1) (1 + 0 +1) (1 + 0 + 0) (1 + 0 + 1)
= (X+Y+Z) (X+Y'+Z) (X+Y'+Z') (X'+Y+Z')

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

14. Obtain the canonical POS for F (A, B, C) = (A+B') (B+C) (A+C') ii) Apply De Morgan theorem for the function [(A+B+C) D]' iii) Find the compliment of A+BC+AB.

i) F(A,B,C) = (A+B'+(C.C'))((A.A')+B+C)(A+(B.B')+C')
= (A+B'+C) (A+B'+C') (A+B+C) + (A'+B+C) (A+B+C') (A+B'+C')

ii) [(A+B+C) D]'
= (A+B+C)'+D'
= A'B'C'+D'

iii) (A+BC+AB)' = A' (BC)' (AB)'
= A' (B'+C') (A'+B')
= (A'B'+A'C') (A'+B')
= A'A'B' + A'A'C' + A'B'B' + A'B'C'
= A'B'+A'C'+A'B'+A'B'C'
= A'B'+A'C'+A'B'C'

15. Use K-map to obtain the POS forms for F=A'B'D'+A'CD+A'BC'D+AB'D'.

F = A'B'D'+A'CD+A'BC'D+AB'D'

= [A'B' (C+C') D'] + [A' (B+B') CD] + [A'BC'D] + [AB' (C+C') D']

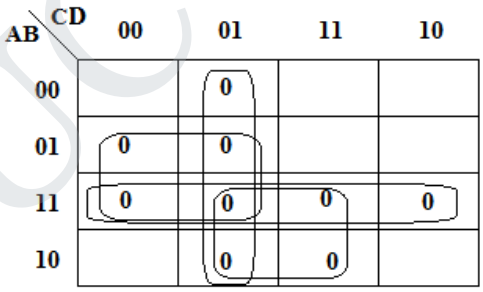
= A'B'CD'+A'B'C'D'+A'BCD+A'B'CD+A'BCD'+AB'CD'+AB'C'D'

= 0 0 1 0 + 0 0 0 0 + 0 1 1 1 + 0 0 1 1 + 0 1 1 0 + 1 0 1 0 + 1 0 0 0

= 2, 0, 7, 3, 6, 10, 8

= Σm (0, 2, 3, 6, 7, 8, 10)

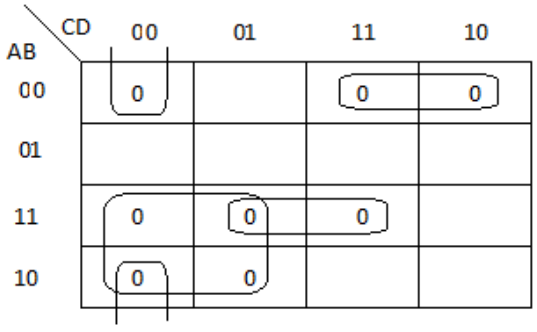
= πM (1, 4, 5, 9, 11, 12, 13, 14, 15)



F = (C + D') (A' + B') (B' + C) (A' + D')

16. Reduce the following function using K-map, f(A,B,C,D) = π M(0,2,3,8,9,12,13,15) *[Apr'15]*

F(A,B,C,D) = π M(0,2,3,8,9,12,13,15)



= (A'+C) (A+B+C') (A'+B'+D') (B+C+D)

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

17. Express the function F=A+B'C in Canonical SOP form and Canonical POS form [*Nov'13, Nov'17]*

Canonical SOP form

F = A + B'C

= A (B + B') (C+C') + (A+A') B'C

= (AB + AB') (C+C') + AB'C + A'B'C

= ABC +ABC' +AB'C + AB'C' +AB'C + A'B'C

= $m_7 + m_6 + m_5 + m_4 + m_5 + m_1$

F = ∑m (1, 4, 5, 6, 7)

Canonical POS form

F= ∑m (1, 4, 5, 6, 7)

F = πM (0, 2, 3)

= $M_0 \cdot M_2 \cdot M_3$

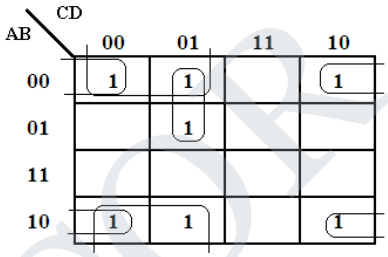
= (A + B + C) (A + B' + C) (A + B' + C')

18. Simplify the given Boolean function F (A, B, C, D) = ∑ (0, 1, 2, 5, 8, 9, 10) into

(i) Sum of products form

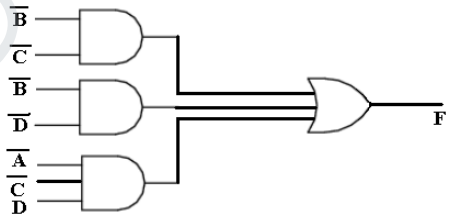
(ii) Product of sum form and implement if using basic gates

(i) Sum of products form: F (A, B, C, D) = ∑m (0, 1, 2, 5, 8, 9, 10)



$$F = \overline{BD} + \overline{BC} + \overline{AC}D$$

Implementation:

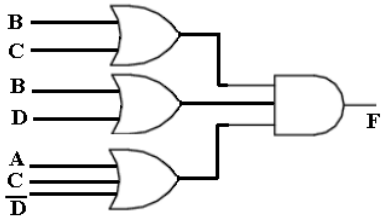


(ii) Product of sum form: $F = \overline{BD} + \overline{BC} + \overline{AC}D$

$$POS = \overline{F} = \overline{\left[\overline{BD} + \overline{BC} + \overline{AC}D\right]}$$

$$\overline{F} = (B + D)(B + C)\left(A + C + \overline{D}\right)$$

Implementation:

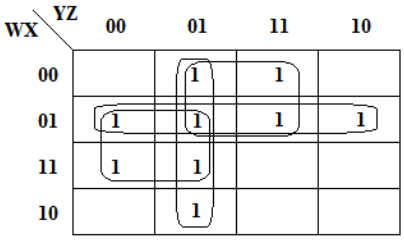


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

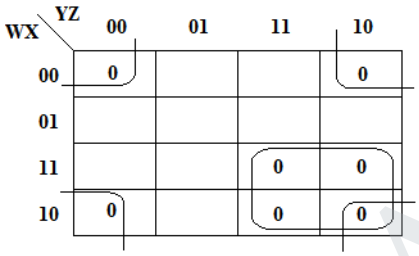
19. Using K map method obtain the minimal SOP & POS expression for the function.
F(W, X,Y,Z) =∑ m(1,3,4,5,6,7,9,12,13)

(i). minimal SOP: F(X,Y,Z,W) =∑ m(1,3,4,5,6,7,9,12,13)



F (W, X, Y, Z) = Y'Z + W'X + W'Z' + XY'

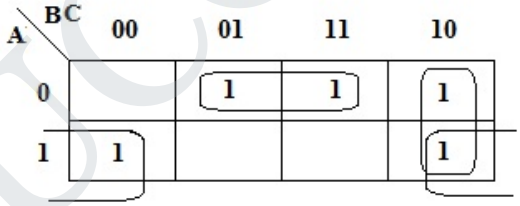
(ii).minimal POS:



F (W, X, Y, Z) = (W'+Y') (X+Z)

20. Simplify the following using K map. X=A'B+A'B'C+ABC'+AB'C'

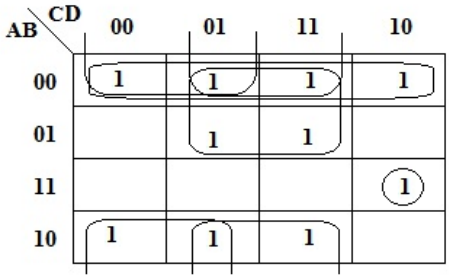
X= A'B+A'B'C+ABC'+AB'C'
= A'B (C+C') +A'B'C+ABC'+AB'C'
=A'BC+A'BC'+A'B'C+ABC'+AB'C'
= 0 1 1 + 0 1 0 + 0 0 1 + 1 1 0 + 1 0 0
= 3, 2, 1, 6, 4
= ∑m (1, 2, 3, 4, 6)



X = A'C + BC' + AC'

21. Minimize the four variable logic function using k-map.
F(A,B,C,D)=∑m(0,1,2,3,5,7,8,9,11,14)

F (A, B, C, D) = ∑ m (0, 1, 2, 3, 5, 7, 8, 9, 11, 14)

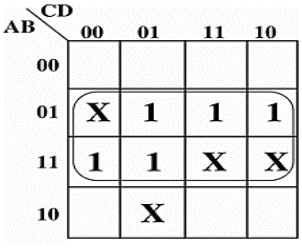


F (A, B, C, D) =AB + A'D + B'C' + B'D + ABCD'

STUCOR APP
DOWNLOADED FROM STUCOR APP

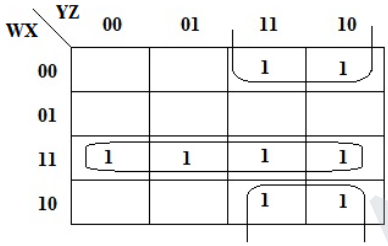
DOWNLOADED FROM STUCOR APP

22. Reduce the following function using Karnaugh map technique.
F(A,B,C,D) = ∑m(5,6,7,12,13)+∑d(4,9,14,15).



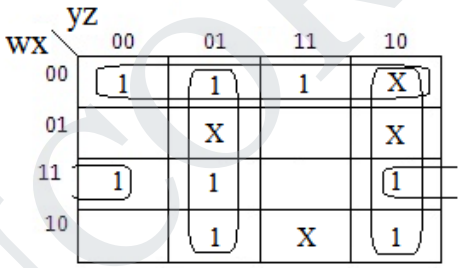
=> F = B

23. Reduce the Following Boolean Function Using 4 Variable K Map.
F(W,X,Y,Z)=∑(2,3,10,11,12,13,14,15)



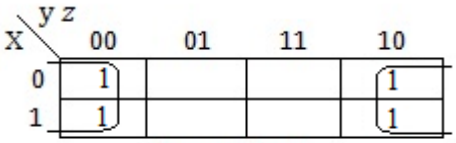
F (W, X, Y, Z) = WX+X'Y

24. Simplify the following function using karnaugh map.
F(w,x,y,z)=∑m(0,1,3,9,10,12,13,14)+∑d(2,5,6,11)*(May'16)*



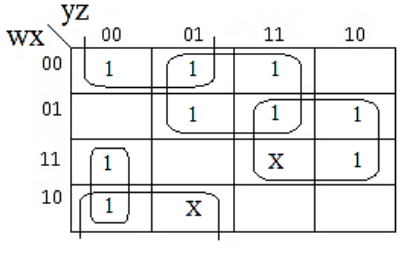
$F = \overline{w}\overline{x} + \overline{y}z + y\overline{z} + wx\overline{z}$

25. Implement the following function using only NAND gates .F(x, y, z) = ∑m (0, 2, 4, 6) *(May'16)*



f = z

26. Reduce the following minterms using k-map. F(w,x,y,z) =∑ m(0,1,3,5,6,7,8,12,14)+∑d(9,15) *[May'17]*

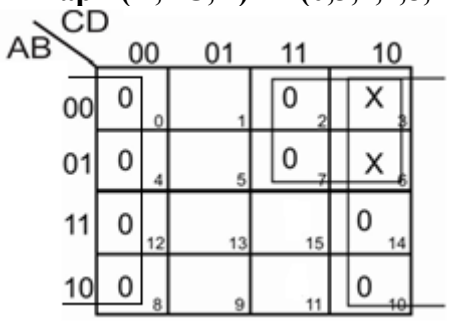


$F = \overline{W}Z + XY + \overline{X}\overline{Y} + W\overline{Y}\overline{Z}$

STUCOR APP
DOWNLOADED FROM STUCOR APP

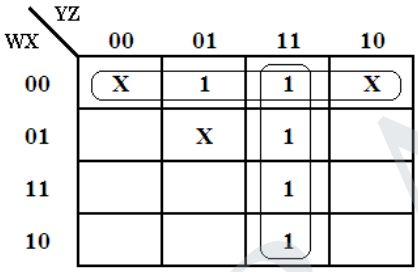
DOWNLOADED FROM STUCOR APP

27. Reduce the following using k map F(A,BC,D)= Π(0,3,4,7,8,10,12,14) + d (2,6)



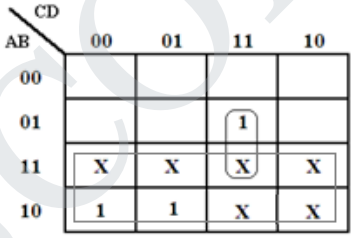
F (A, B, C, D) =D. (A+C')

28. Simplify the Boolean function using K-map F (W,X,Y,Z) =∑ (1,3,7,11,15) which has the don't care conditions d (W,X,Y,Z)= ∑(0,2,5). *[May'13]*



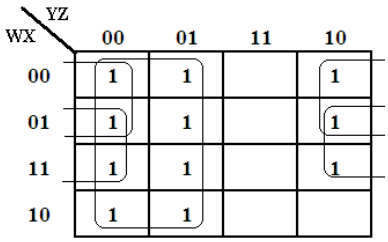
$\therefore F = \overline{wx} + yz$

29. Simplify using k-map F(A,B,C,D) = ∑ m(7,8,9) + d(10,11,12,13,14,15) *[Nov'13]*



F = A + BCD

30. Simplify using K map F(W,X,Y,Z) = ∑(0,1,2,4,5,6,8,9,12,13,14) *[Apr'10]*



$f(w, x, y, z) = \overline{y} + x\overline{z} + \overline{wz}$

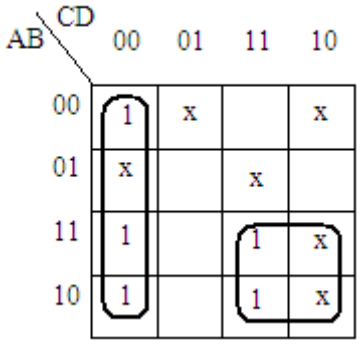
31. Give the simplified expression for the following logic equation where d represents don't care condition. F(A,B,C,D)=∑m(0,8,11,12,15) + d(1,2,4,7,10,14). Represent the simplified expression using logic gates. *[Nov'11]*

F(A,B,C,D)=∑m(0,8,11,12,15)+d(1,2,4,7,10,14)

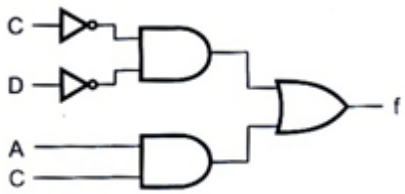
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

K-map simplification:



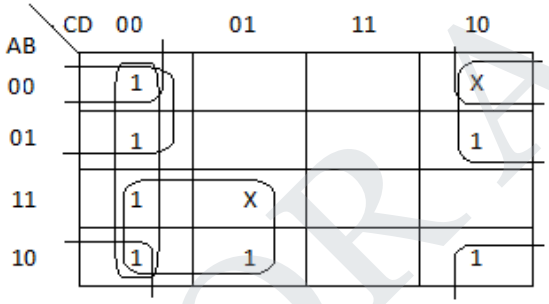
Implementation:



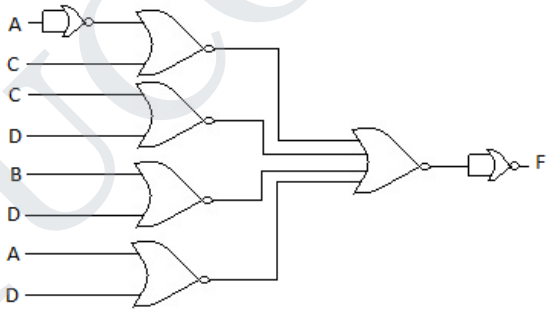
F (A, B, C, D) = $\bar{C}\bar{D}$+AC

32. Minimize the function F (a, b, c, d) = Σ(0, 4, 6, 8, 9, 10, 12) with d = Σ (2, 13). Implement the function using only NOR gates. *[Nov'14]*

F (a, b, c, d) = Σm (0, 4, 6, 8, 9, 10, 12), d = Σ(2, 13)

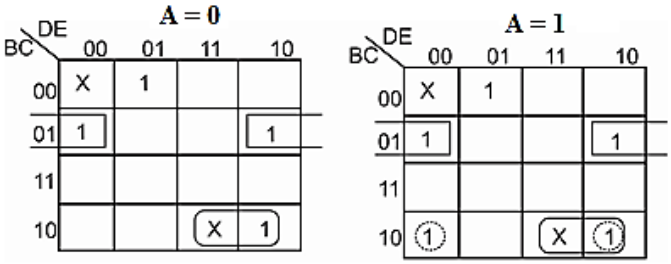


F = AC' + C'D'+B'D'+A'D'



33. Find a minimal sum of products representation for f(A,B,C,D,E) = ∑m (1,4,6,10,20,22,24,26) + d(0,11,16,27) using k map. Draw the circuit of the minimal expression using only NAND gates.

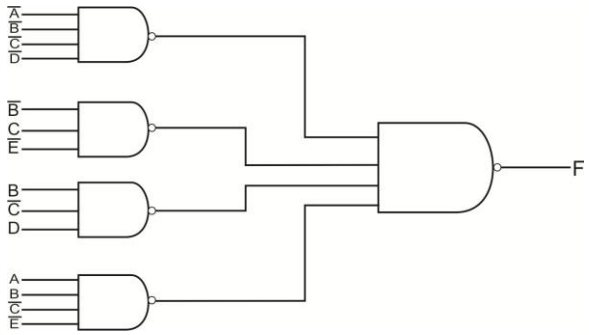
5 Variable K Map



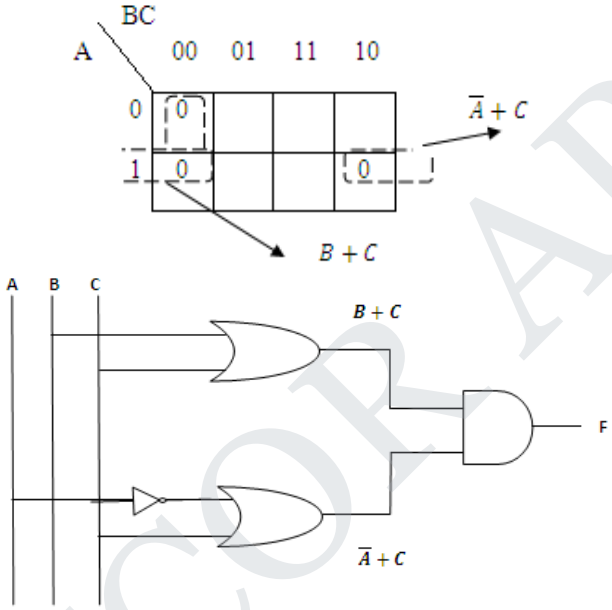
f (A, B, C, D, E) =A'B'C'D'+B'CE'+BC'D+ABC'E'

STUCOR APP
DOWNLOADED FROM STUCOR APP

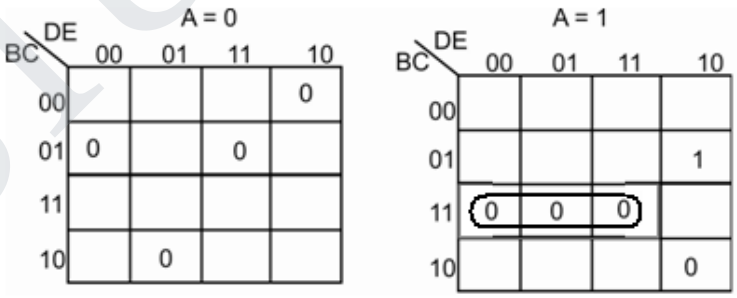
DOWNLOADED FROM STUCOR APP



34. Using K-map simplify the following function and implement the function using logic gates f(A, B, C) = πM (0, 4, 6) *[Nov'12]*



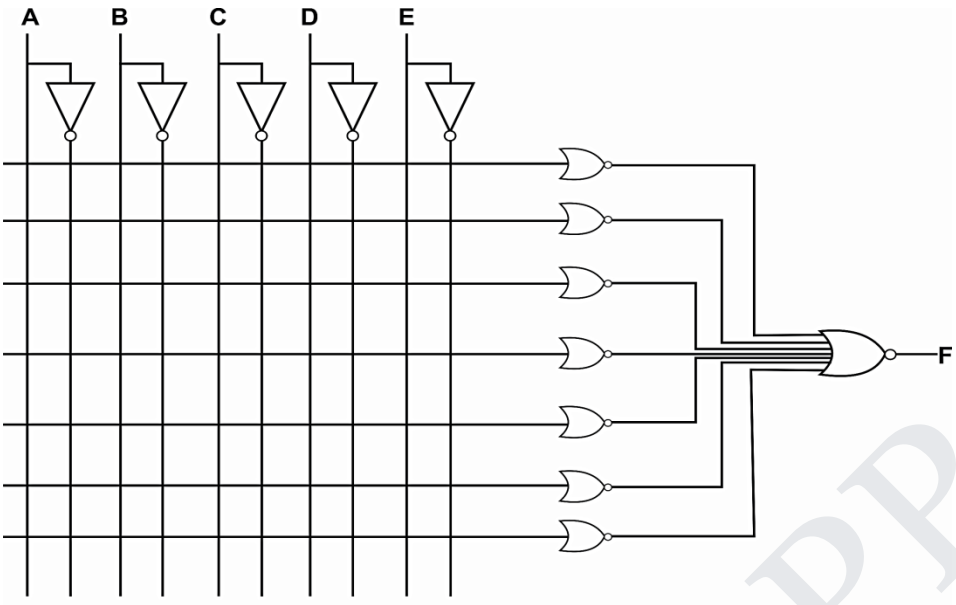
35. Minimize the following using k map. Implement the resultant function using NOR gates only. f(A,B,C,D,E)=πM(2,4,7,9,26,28,29,31)



f(A,B,C,D,E) = (A+B+C'+D+E)(A+B+C'+D'+E') (A+B+C+D'+E) (A+B'+C+D+E) (A'+B'+C'+D) (A'+B'+C'+E')(A'+B'+C+D'+E)

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



36. Implement the following function using a quad 2 input NOR gates. F= (A'B+C) D

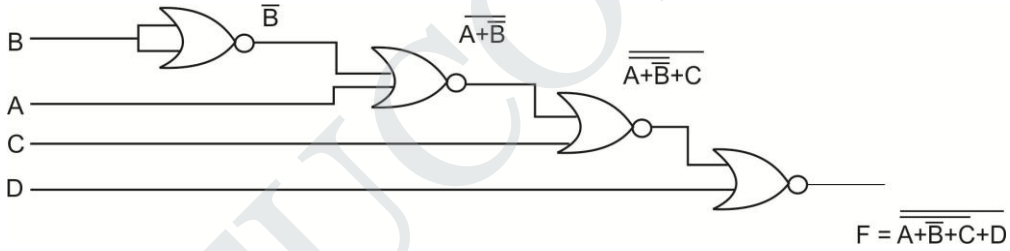
F= (A'B+C) D

= (((A'B+C) D)')'

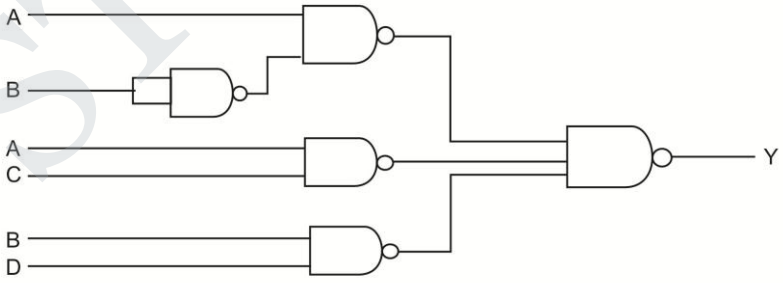
= ((A'B+C) D)'+ (D')')'

= ((A'B+C)'+D)'

F = (((A+B')'+C)'+D)'



37. Sketch a NAND-NAND logic circuit for the Boolean expression. Y=AB'+AC+BD

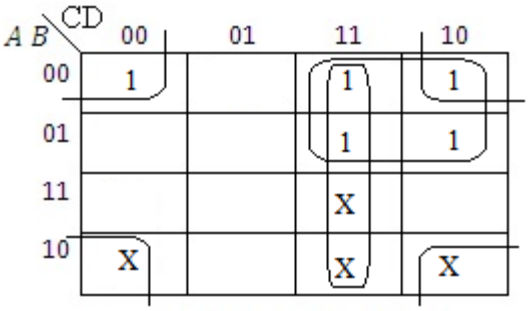


38. Simplify the logical expression using K-map in SOP and POS form. F(A,B,C,D)=∑m(0,2,3,6,7)+d(8,10,11,15). *(Nov'16)*

STUCOR APP
DOWNLOADED FROM STUCOR APP

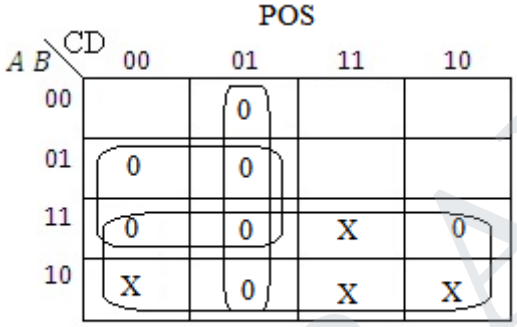
DOWNLOADED FROM STUCOR APP

SOP



$$F = CD + \overline{A}C + \overline{B}\,\overline{D}$$

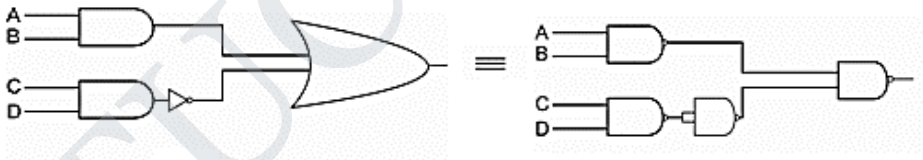
POS



$$F = [\overline{B} + C]\,[C + \overline{D}]\,\overline{A}$$

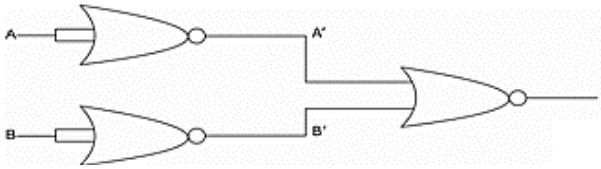
39. a) Implement the function F=AB + (CD)' using NAND gate only.
b) Implement 2 input AND gate using NOR gate only.

a) Implementing F=AB+ (CD)' using NAND gate.



$$F = \overline{\overline{AB}\,.\,CD} = \overline{\overline{AB}} + \overline{\overline{CD}} = AB + \overline{CD}$$

b) Implementing 2 input AND gate using NOR gate.



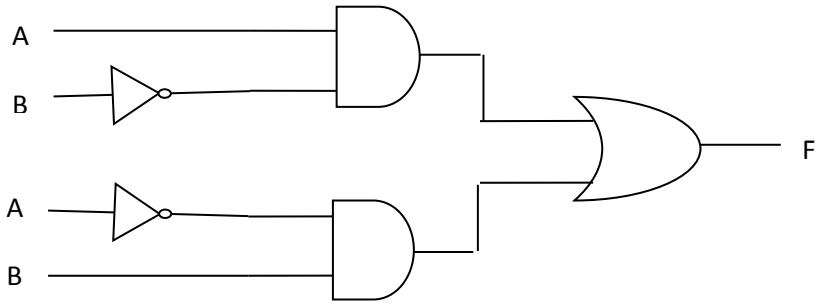
$$F = \overline{(\overline{A} + \overline{B})} = \overline{(\overline{A})}.\overline{(\overline{B})} = A.B$$

STUCOR APP
DOWNLOADED FROM STUCOR APP

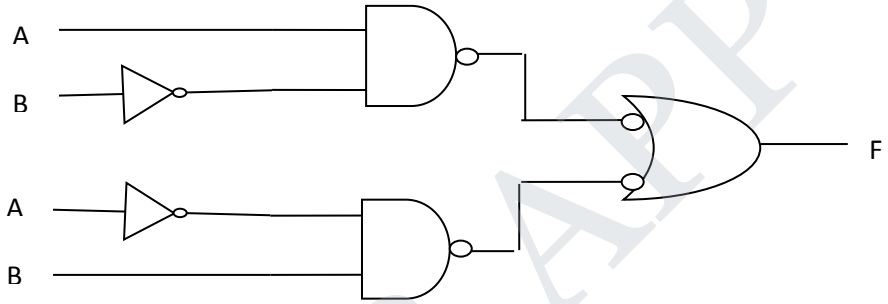
DOWNLOADED FROM STUCOR APP

40. Prove that for constructing XOR from NAND's we need 4 NAND gates. *[May'13]*

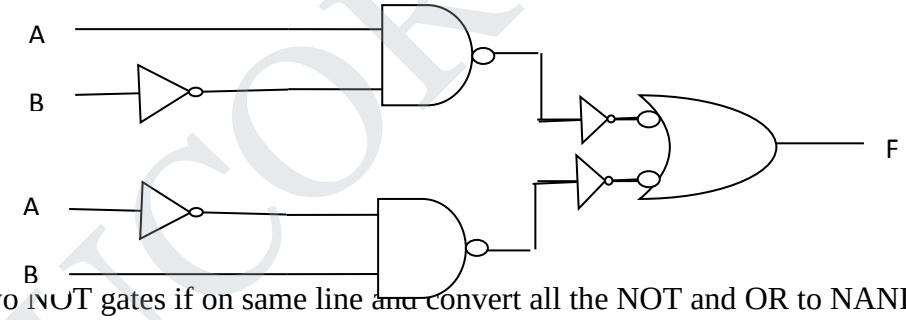
Step 1: Draw original OR / AND / NOT circuit



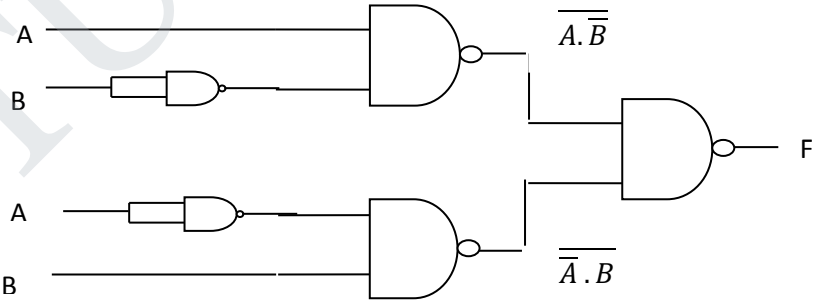
Step 2: Place bubbles on outer of AND gates and inner of OR gates



Step 3: Place NOT gate near all bubbles placed in step 2.



Step 4: Cancel two NOT gates if on same line and convert all the NOT and OR to NAND gates.



41. Implement the following Boolean function with NAND-NAND logic.
Y=AC+ABC+A'BC+AB+D *[May'12]*

Step 1: Simplify the given Boolean function.

$Y = AC + ABC + \bar{A}BC + AB + D$

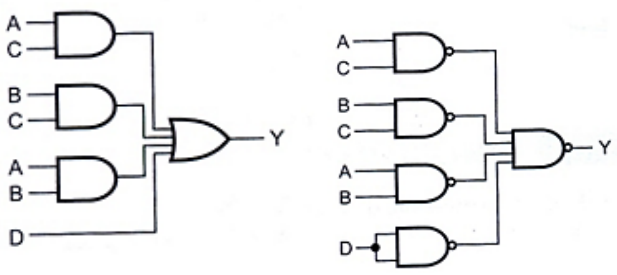
$= AC + BC(A + \bar{A}) + AB + D = AC + BC + AB + D$

Step 2: Implement using AND-OR logic.

Step3: Convert AND-OR logic to NAND-NAND logic.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



42. Write brief notes on the following: i) Demorgan's theorem ii) Comparators iii) Binary to gray code converter. *[Nov'11]*

(i) Demorgan's theorem

- De Morgan suggested two thermos that form an important part of Boolean algebra In the equation form, they are

  1.$\overline{AB} = \bar{A} + \bar{B}$

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

1. $\overline{AB} = \bar{A} + \bar{B}$:

The complement of a product is equal to the sum of the complements.

A	B	$\overline{AB}$	$\bar{A} + \bar{B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

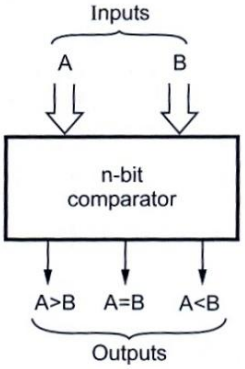
2. $\overline{A + B} = \bar{A}.\bar{B}$:

The complement of a sum is equal to the product of the complements.

A	B	$\overline{A + B}$	$\bar{A}.\bar{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

(ii) Comparators

- A comparator is a special combinational circuit designed primarily to compare the relative magnitude of two binary numbers.
- Fig shows the block diagram of an n-bit comparator.
- It receives two n-bit numbers A and B as inputs and the outputs are **A > B, A=B and A< B**.
- Depending upon the relative magnitudes of the two number, one of the outputs will be high.



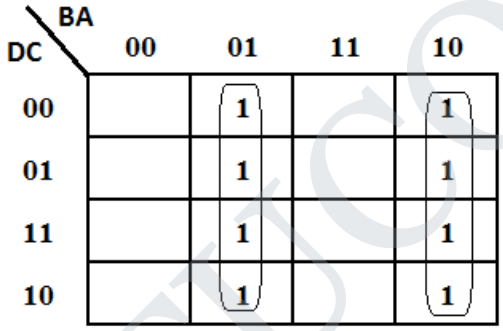
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

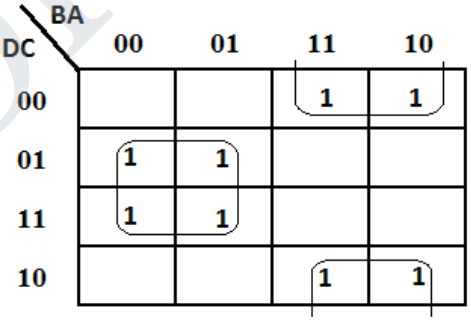
(iii) Binary to gray code converter

- Let us represent binary number as A,B,C, D and its equivalent gray code as G_0G_1 G_2 G_3 with this representation gray code bits are obtained from the binary bits as follows:

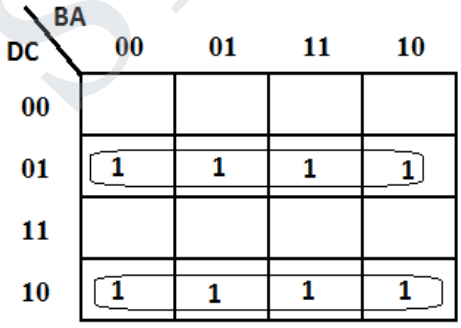
Binary code					Gray code			
Decimal	**D**	**C**	**B**	**A**	**G3**	**G2**	**G1**	**G0**
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	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



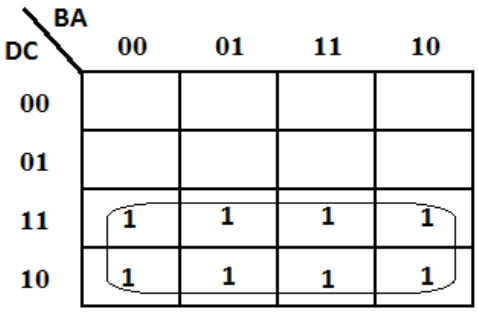
$G_0 = \overline{B}A + \overline{B}A = A \oplus B$



$G_1 = \overline{B}C + \overline{C}B = B \oplus C$



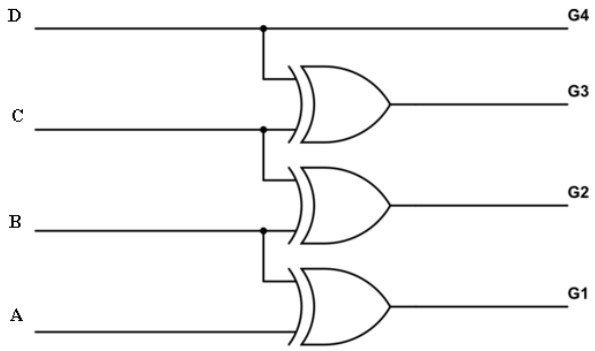
$G_2 = \overline{D}C + \overline{C}D = C \oplus D$



$G_3 = D$

STUCOR APP
DOWNLOADED FROM STUCOR APP

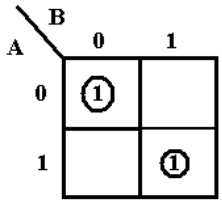
DOWNLOADED FROM STUCOR APP

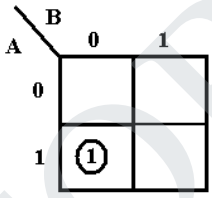


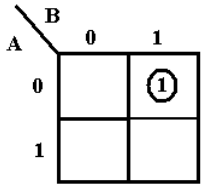
43. Design a 1 – bit comparator using basic gates.

Consider two one bit numbers A and B. The truth table is

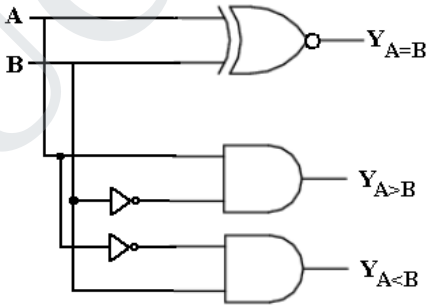
Inputs		Outputs		
A	**B**	$Y_{A=B}$	$Y_{A>B}$	$Y_{A<B}$
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0







$$Y_{A=B} = \overline{AB} + AB = \overline{A \oplus B}\; Y_{A>B} = A\overline{B} Y_{A<B} = \overline{A}B$$



44. Design a 2 – bit comparator using basic gates.

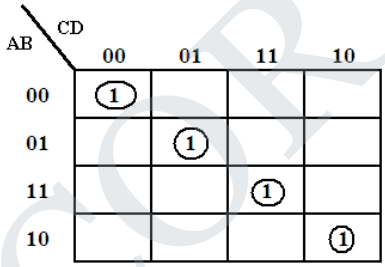
- A two bit comparator compares the magnitude of two bits
- Number of required inputs = 4 (A_0,A_1,B_0, B_1)
- Number of required outputs = 3 (X, Y, Z)

Where X = A_0 A_1 = B0 B_1 Y = A_0 A_1 > B_0 B_1 Z = A_0 A_1 < B_0 B_1

STUCOR APP
DOWNLOADED FROM STUCOR APP

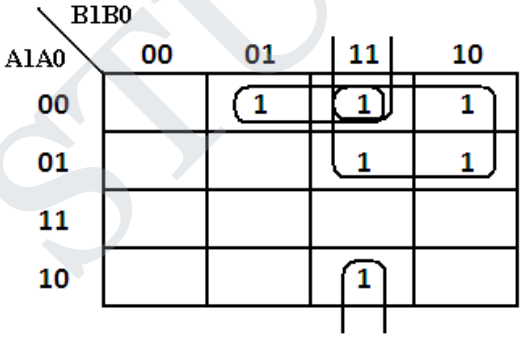
DOWNLOADED FROM STUCOR APP

Inputs				Outputs		
A_1	A_0	B_1	B_0	X	Y	Z
0	0	0	0	1	0	0
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0
0	1	0	0	0	0	1
0	1	0	1	1	0	0
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	1	0	0
1	0	1	1	0	1	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	1	0	0

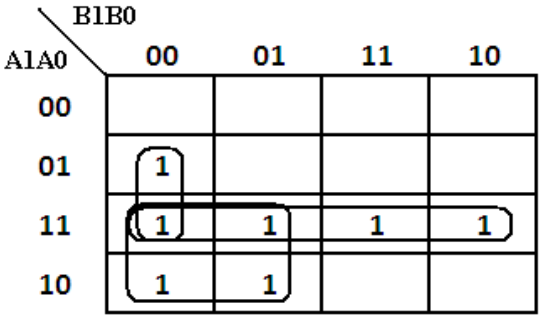


$$X = \overline{A_1 A_0 B_1 B_0} + \overline{A_1} A_0 \overline{B_1} B_0 + A_1 A_0 B_1 B_0 + A_1 \overline{A_0} B_1 \overline{B_0}$$

$$X = (A_1 \odot B_1)(A_0 \odot B_0)$$



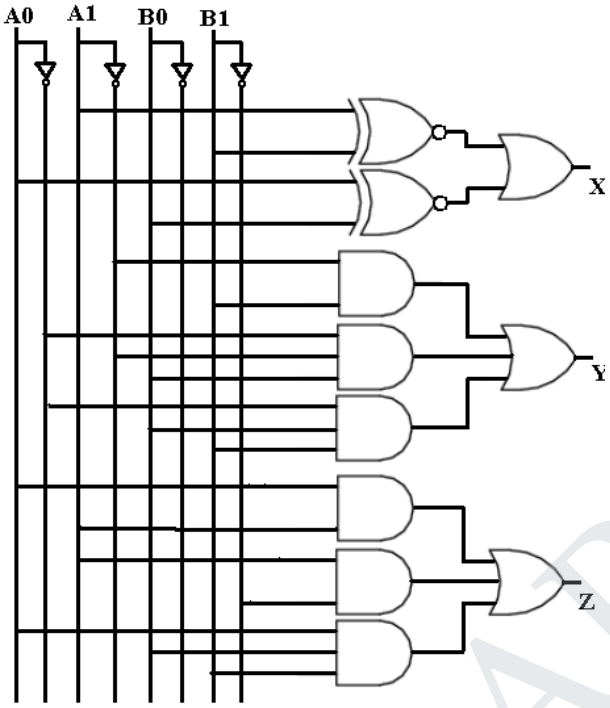
$$Y = \overline{A_1 A_0} B_0 + \overline{A_1} B_1 + \overline{A_0} B_1 B_0$$



$$Z = A_1 A_0 + A_1 \overline{B_1} + A_0 B_1 B_0$$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

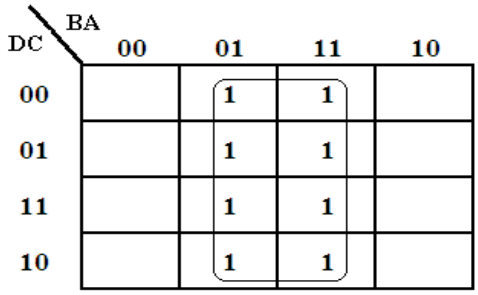


45. Convert Binary numbers to BCD code numbers.

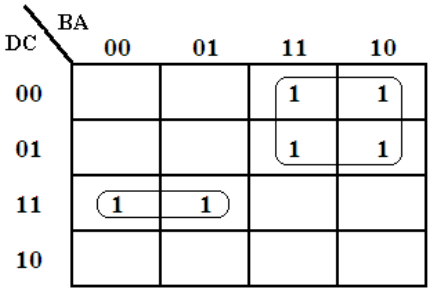
Binary Code				BCD Code				
D	**C**	**B**	**A**	**B4**	**B3**	**B2**	**B1**	**B0**
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1	0
0	0	1	1	0	0	0	1	1
0	1	0	0	0	0	1	0	0
0	1	0	1	0	0	1	0	1
0	1	1	0	0	0	1	1	0
0	1	1	1	0	0	1	1	1
1	0	0	0	0	1	0	0	0
1	0	0	1	0	1	0	0	1
1	0	1	0	1	0	0	0	0
1	0	1	1	1	0	0	0	1
1	1	0	0	1	0	0	1	0
1	1	0	1	1	0	0	1	1
1	1	1	0	1	0	1	0	0
1	1	1	1	1	0	1	0	1

STUCOR APP
DOWNLOADED FROM STUCOR APP

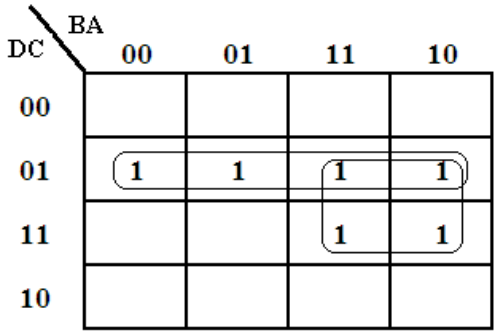
DOWNLOADED FROM STUCOR APP



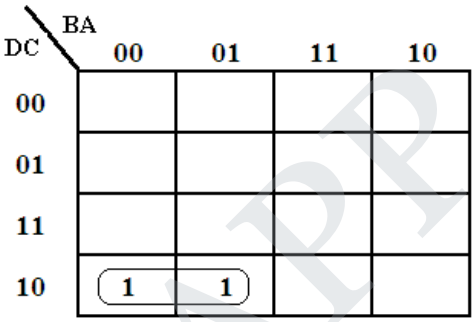
$B_4 = A$



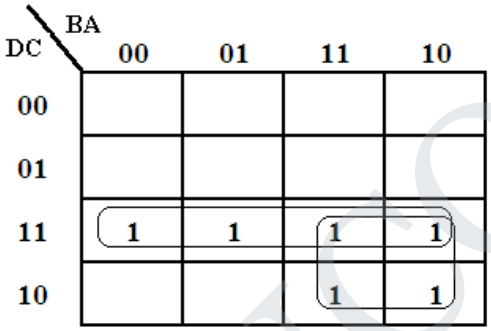
$B_3 = DC\overline{B} + \overline{D}B$



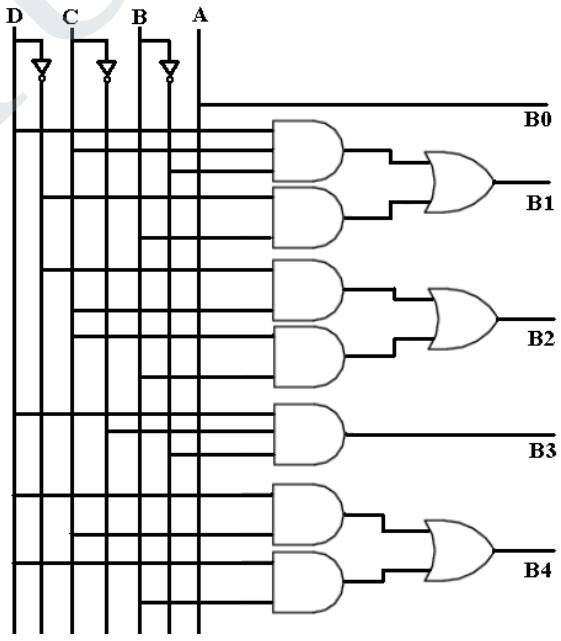
$B_2 = \overline{D}C + CB$



$B_1 = D\bar{C}\bar{B}$



$B_0 = DC + DB$

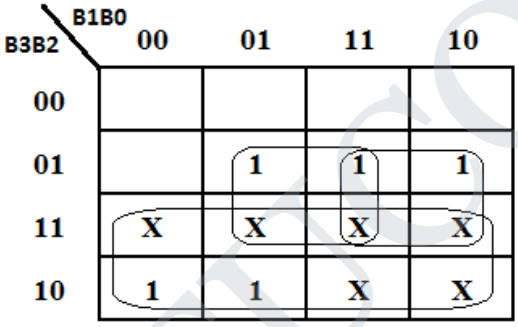


STUCOR APP
DOWNLOADED FROM STUCOR APP

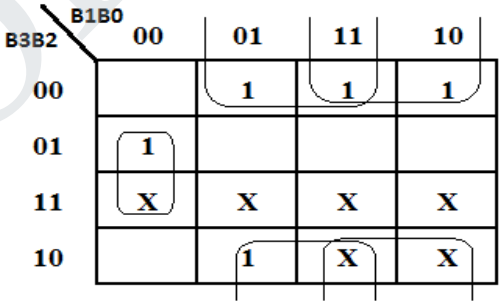
DOWNLOADED FROM STUCOR APP

46. Explain how to convert Binary Coded Decimal (BCD) to Excess – 3 codes? *[Apr'15]*(OR) **Design a 4 bit BCD to Excess 3 code converter and implement using logic gates.** *[Nov'15, May'16]*

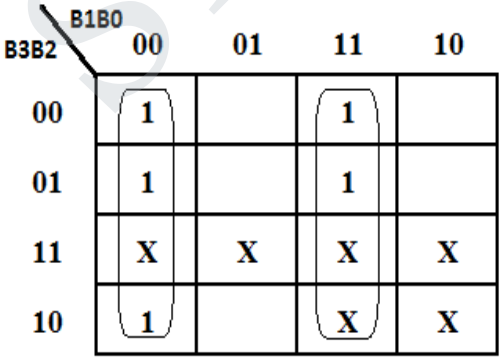
	BCD Code				Excess – 3 code			
Decimal	B3	B2	B1	B0	E3	E2	E1	E0
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0
10	X	X	X	X	X	X	X	X
11	X	X	X	X	X	X	X	X
12	X	X	X	X	X	X	X	X
13	X	X	X	X	X	X	X	X
14	X	X	X	X	X	X	X	X
15	X	X	X	X	X	X	X	X



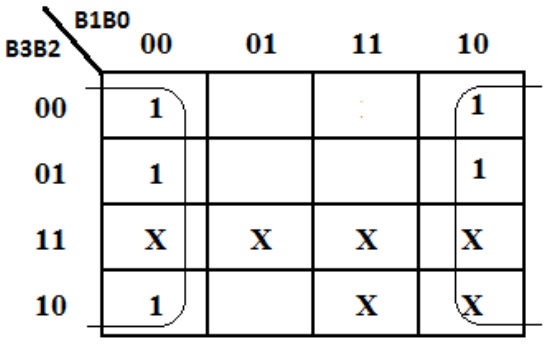
$E_3 = B_3 + B_2B_0 + B_2B_1$



$E_2 = B_2\overline{B_1B_0} + \overline{B_2}B_0 + \overline{B_2}B_1$



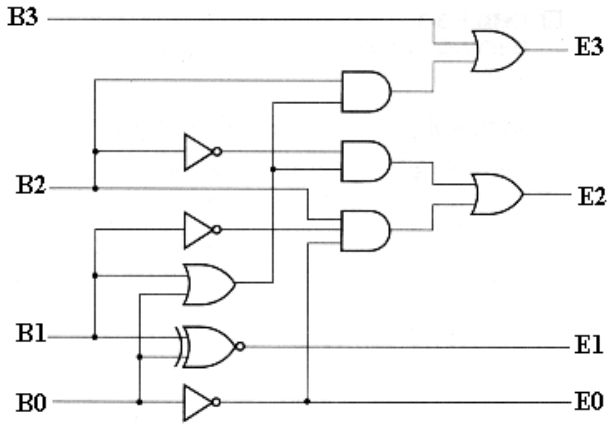
$E_1 = \overline{B_1B_0} + B_1B_0 = B_1 \odot B_0$



$B_0 = \overline{B_0}$

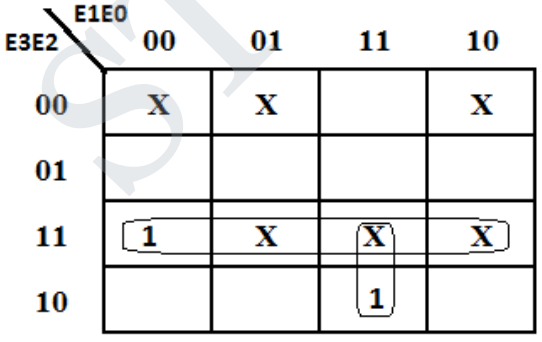
STUCOR APP
DOWNLOADED FROM STUCOR APP

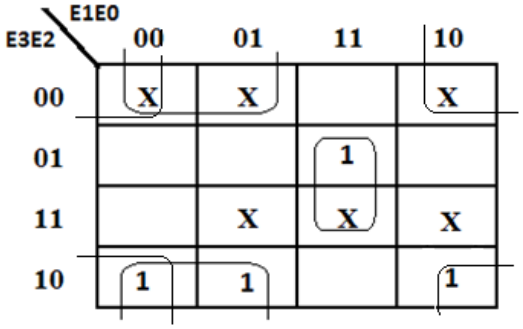
DOWNLOADED FROM STUCOR APP



47. Explain how to convert Excess – 3 codes to Binary Coded Decimal (BCD)?

Excess – 3 code				BCD code			
E3	**E2**	**E1**	**E0**	**B3**	**B2**	**B1**	**B0**
0	0	0	0	X	X	X	X
0	0	0	1	X	X	X	X
0	0	1	0	X	X	X	X
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	1
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

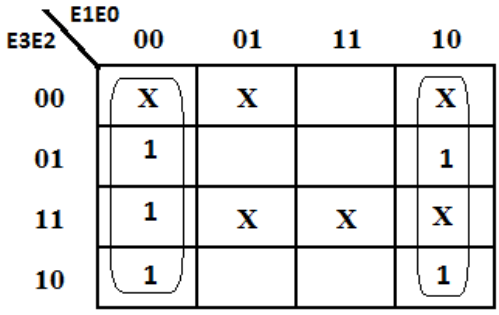


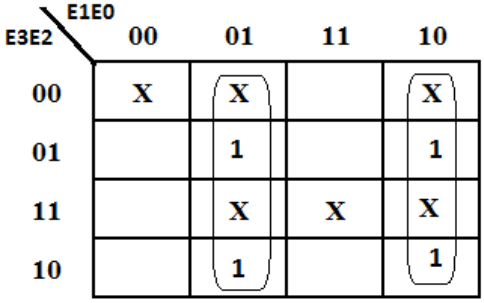


$$B_3 = E_3E_2 + E_3E_1E_0 = E_3(E_2 + E_1E_0) \qquad B_2 = \overline{E_2E_1} + \overline{E_2E_0} + E_2E_1E_0$$

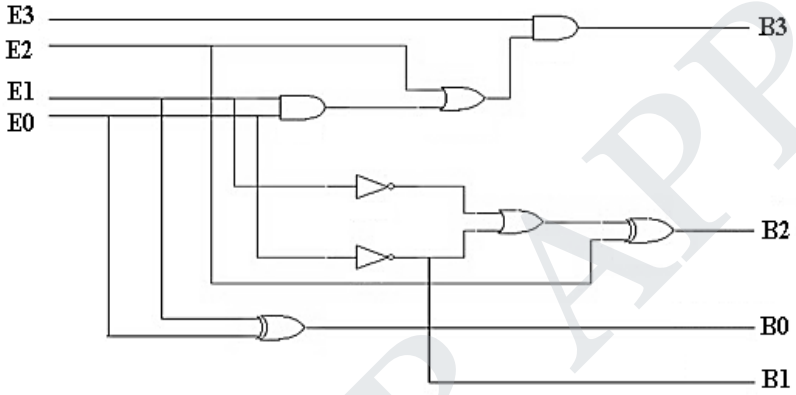
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP





$B_1 = \overline{E_1}E_0 + E_1\overline{E_0} = \overline{E_0}$ $B_0 = \overline{E_1}E_0 + E_1\overline{E_0} = E_1 \oplus E_0$

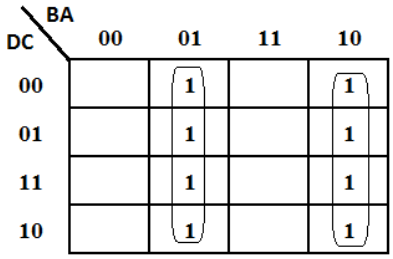


48. Design a 4-bit Binary to gray code converter and implement it using logic gates? *[Nov'14]*

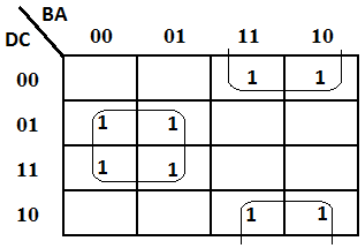
Binary code					Gray code			
Decimal	**D**	**C**	**B**	**A**	**G3**	**G2**	**G1**	**G0**
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	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

STUCOR APP
DOWNLOADED FROM STUCOR APP

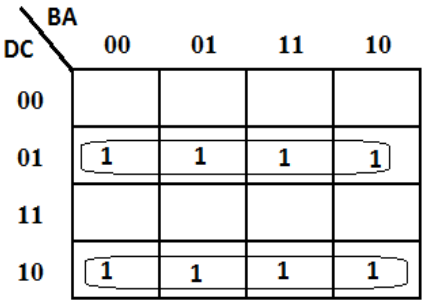
DOWNLOADED FROM STUCOR APP



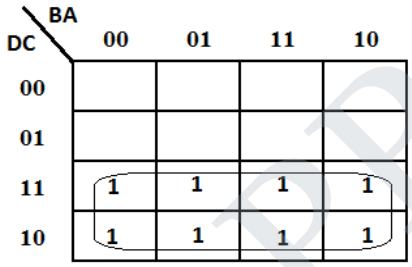
$G_0 = \overline{B}A + \overline{B}A = A \oplus B$



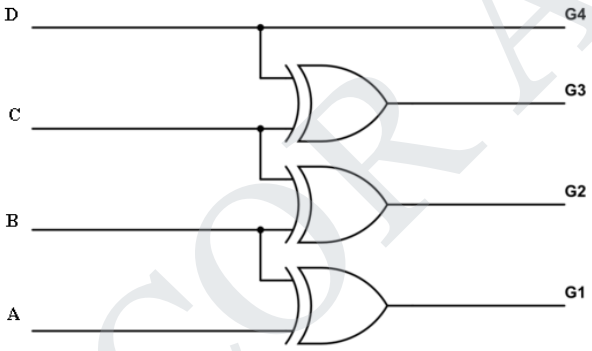
$G_1 = \overline{B}C + \overline{C}B = B \oplus C$



$G_2 = \overline{D}C + \overline{C}D = C \oplus D$



$G_3 = D$

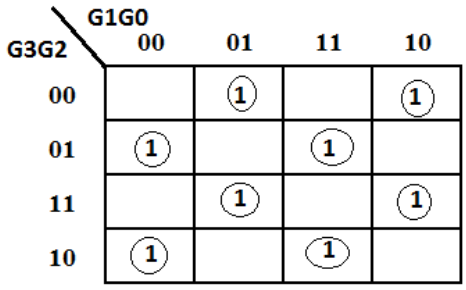


49. Explain how to convert Gray codes to Binary Codes? [OR] **Design a 4 bit gray code to binary converter and express using logic gates.** *(Nov'17)*

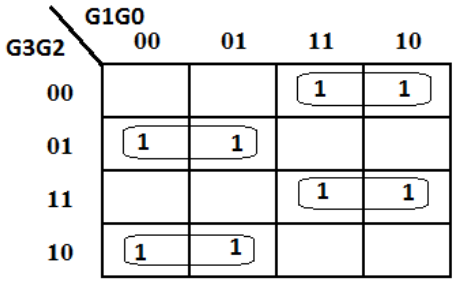
Gray code					Binary code			
Decimal	**G3**	**G2**	**G1**	**G0**	**D**	**C**	**B**	**A**
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	0
3	0	0	1	1	0	0	1	1
4	0	1	0	0	0	1	0	0
5	0	1	0	1	0	1	0	1
6	0	1	1	0	0	1	1	0
7	0	1	1	1	0	1	1	1
8	1	0	0	0	1	0	0	0
9	1	0	0	1	1	0	0	1
10	1	0	1	0	1	0	1	0
11	1	0	1	1	1	0	1	1
12	1	1	0	0	1	1	0	0
13	1	1	0	1	1	1	0	1
14	1	1	1	0	1	1	1	0
15	1	1	1	1	1	1	1	1

STUCOR APP
DOWNLOADED FROM STUCOR APP

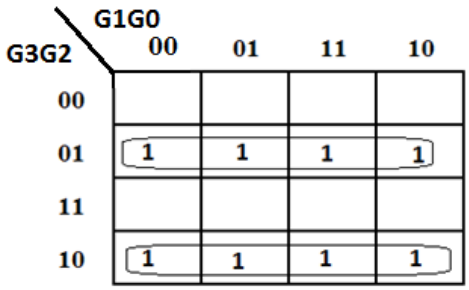
DOWNLOADED FROM STUCOR APP



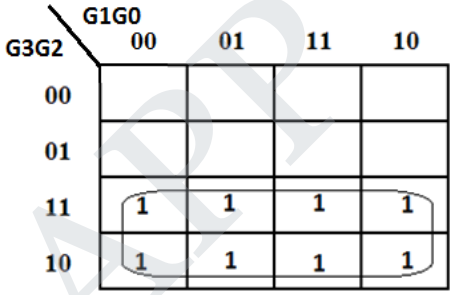
$A = (G_3 \oplus G_2) \oplus (G_1 \oplus G_0)$



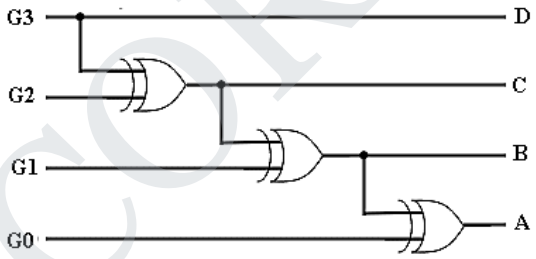
$B = G_3 \oplus G_2 \oplus G_1$



$C = G_3 \oplus G_2$



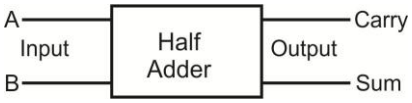
$D = G_3$



50. Explain Half Adder & Full adder circuit. [OR] **Design a decimal adder to add two decimal digits** *[Apr'10]* **Design a full adder using two half-adders and an or gate**. *[Apr'15]* [OR] **Design a Full adder and implement it using suitable multiplexer.** *[May'16]* [OR] **Design a full adder using only NOR gate.** *[May'17]*

Half Adder:

- For adding two binary bits are called Half Adder.



Truth Table

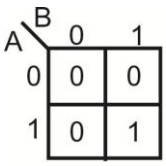
Inputs		Outputs	
A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

STUCOR APP
DOWNLOADED FROM STUCOR APP

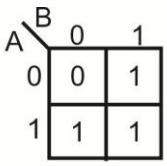
DOWNLOADED FROM STUCOR APP

K-map:-

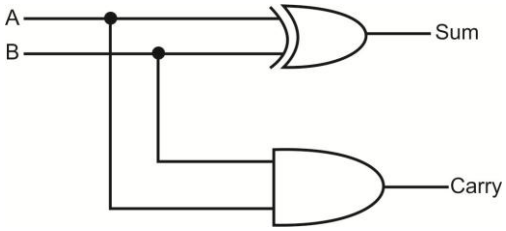
For Carry



For Sum

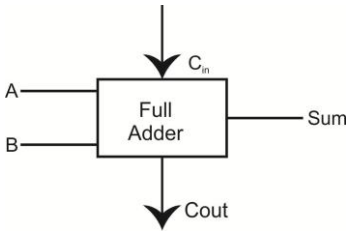


Logic Diagram

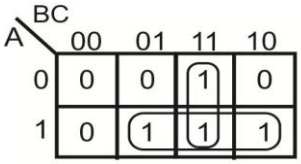


Full Adder:

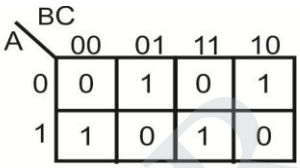
It adds more than 2 bits.



For Carry



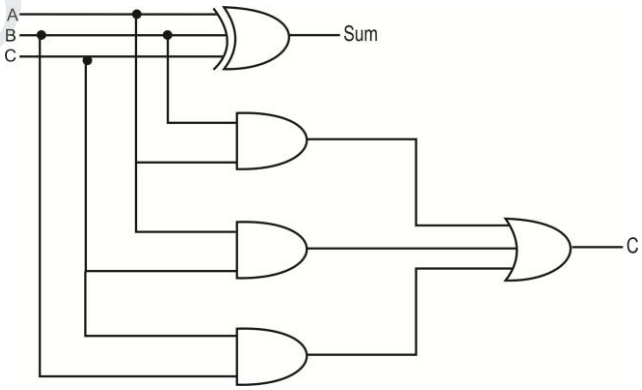
For Sum



Truth Table

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

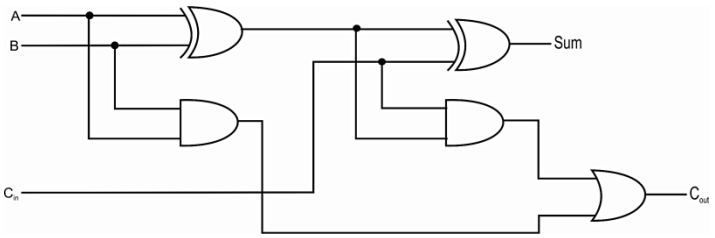
Logic Diagram



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

Carry = AB+BC+AC

Full Adder Using Two Half Adder:



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

51. Explain Half Subtractor & Full Subtractor Circuit. *[Nov'13, Nov'14]*(OR) **Design a full subtractor and implement using logic gates.** *[Nov'15]* (OR**) Design a full subtractor and realize using logic gates. Also implement the same using half subtractors.** *(Nov'16)*

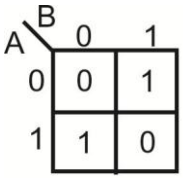
Half Subtractor: It subtracts two bits

Truth table:

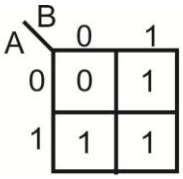
Inputs		Outputs	
A	B	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

K-map:

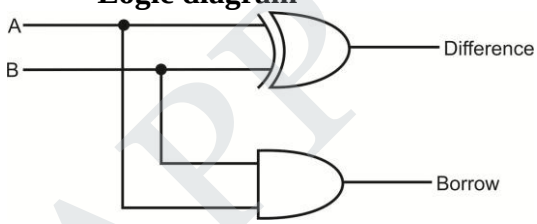
For Difference



For Borrow



Logic diagram

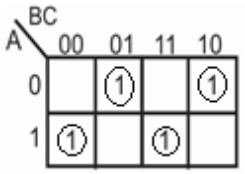


Full Subtractor

Inputs			**Outputs**	
A	**B**	**C_{in}**	**D**	**Bout**
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

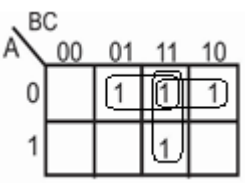
K-map:

Difference:



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

Borrow:

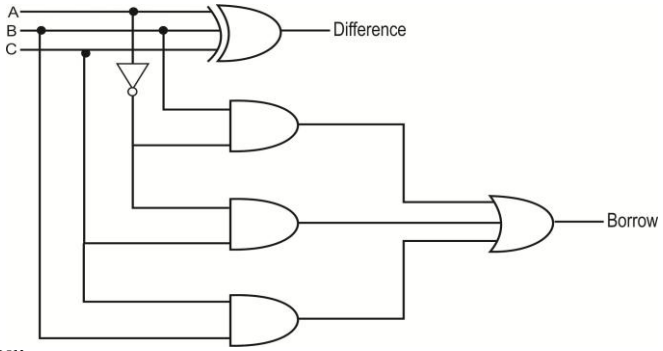


$$\text{Borrow} = \bar{A}C + \bar{A}\,B + BC$$

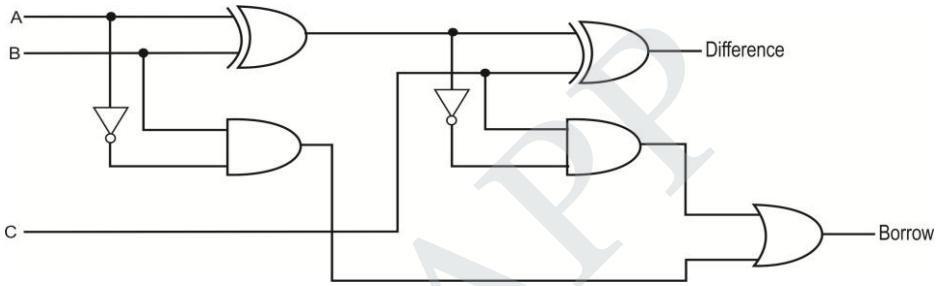
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Logic Diagram

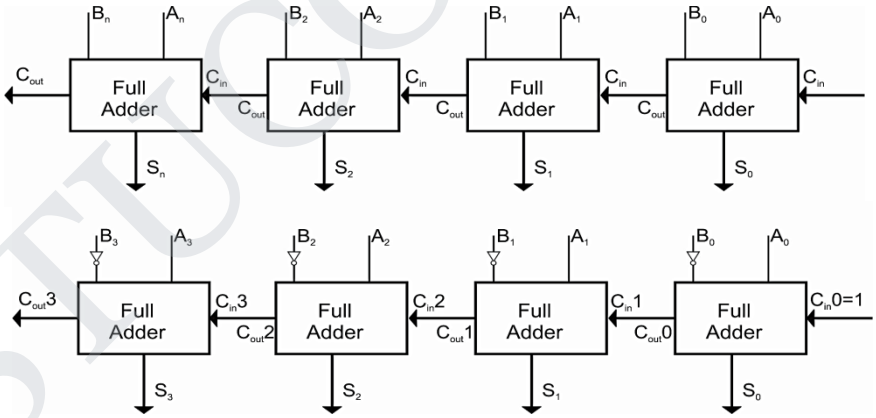


Using two half – subtractor:



52. Explain Parallel adder & Parallel subtractor.

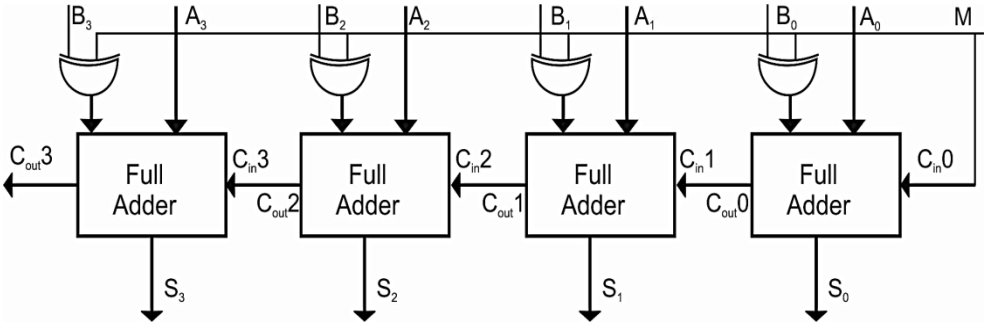
- A single full-adder is capable of adding two one-bit number and an input carry.
- In order to add binary numbers with more than one bit, additional full-adder must be employed.
- An-bit, parallel adder can be constructed using number of full adder circuits connected in parallel.



- The subtraction of binary numbers can be done most conveniently by means of complements.
- Remember that the subtraction A-B can be by taking the 2's complement of B and adding it to A.
- The 2's compliment can be obtained by taking the 1's complement and adding one to the least signification pair of bits.
- The 1's complement can be implemented with inverters and a one can be added to the sum through the input carry.

STUCOR APP
DOWNLOADED FROM STUCOR APP

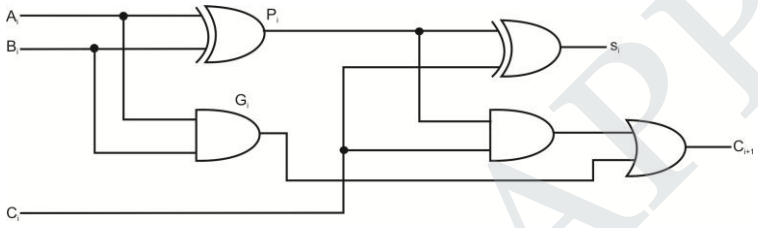
DOWNLOADED FROM STUCOR APP



53. Explain Look ahead-carry generator (Fast Adder)

- Since due to time delay we go for fast adder. This delay is known as carry propagation delay.

Full-adder circuit



- Define two functions: carry generator and carry propagate.

$$P_i = A_i + B_i$$

$$G_i = A_i B_i$$

- The output sum and carry can be expressed as

$$S_i = P_i + C_i$$

$$C_{i+1} = G_i + P_i C_i$$

- The Boolean function for the carry output of each stage can be written as follows.

$$C_2 = G_1 + P_1 C_3$$

$$C_3 = G_2 + P_2 C_2$$

$$= G_2 + P_2 (G_1 + P_1 C_1)$$

$$= G_2 + P_2 G_1 + P_2 P_1 C_1$$

$$C_4 = G_3 + P_3 C_3$$

$$= G_3 + P_3 (G_2 + P_2 G_1 + P_2 P_1 C_1)$$

$$= G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_1$$

54. Draw and explain the BCD adder circuit.

- The digital system handles the decimal number in the form of binary coded decimal numbers (BCD).
- A BCD adder is a circuit that adds two BCD digits and produces the sum digits also in BCD.
- BCD numbers use 10 digits, 0 to 9 which are represented in the binary form 0000 to 1001.

BCD addition procedure:

- Add two BCD numbers using ordinary binary addition.
- If four-bit sum is equal to or less that q, no correction is needed. The sum is in proper BCD form.

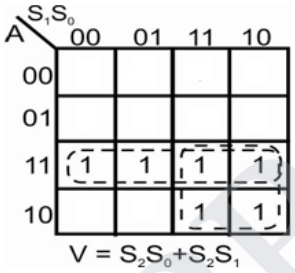
STUCOR APP
DOWNLOADED FROM STUCOR APP

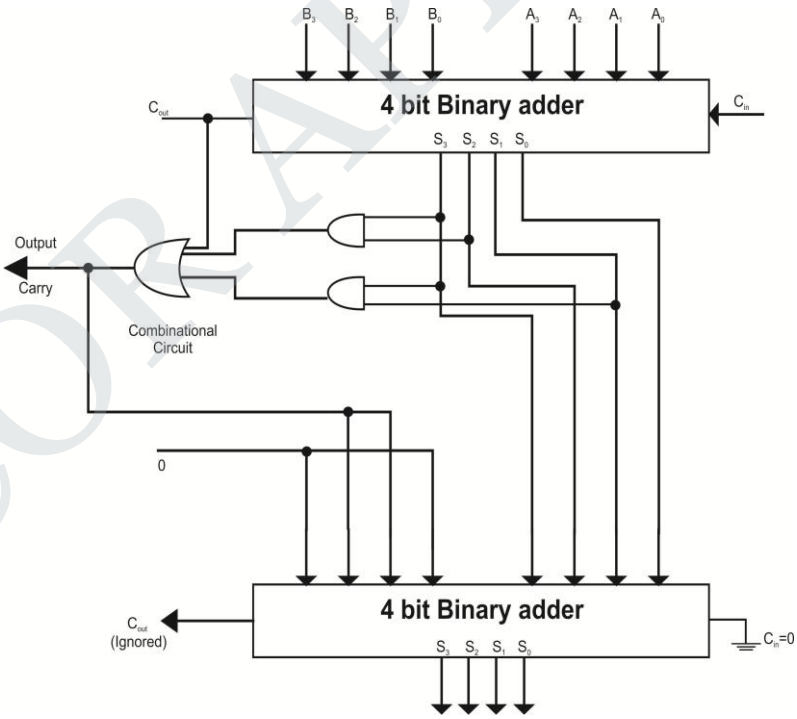
DOWNLOADED FROM STUCOR APP

- If the four-bit sum is greater than 9 or if a carry is generated from the four-bit sum, the sum is invalid.
- To correct the invalid sum, add 01101_2 to four-bit sum. If a carry results from this addition, add it to the next higher order BCD digit.
- Thus to implement BCD adder we require. 4-bit binary adder for initial addition.
- One more 4-bit adder to add 0110_2 in the sum if sum is greater than 9 or carry is 1.

Truth Table

Inputs				Outputs
S_3	S_2	S_1	S_0	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	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1





55. Define Encoder and Design the Octal to Binary Encoder (OR) 8:3 Encoder.

- Encoder is a digital circuit that performs the inverse operation of a decoder.
- An encoder has 2n input line.
- It is just opposite to decoder.
- It converts the useful information into binary digit.
- Based on the input, the binary code is generated.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Design of the Octal to Binary Encoder

Step 1: Truth Table

D0	D1	D2	D3	D4	D5	D6	D7	A	B	C
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Step 2: Boolean Expression when A=1, inputs are D_4, D_5, D_6, D_7

B=1, inputs are D_2, D_3, D_6, D_7

C=1, inputs are D_1, D_3, D5,D7

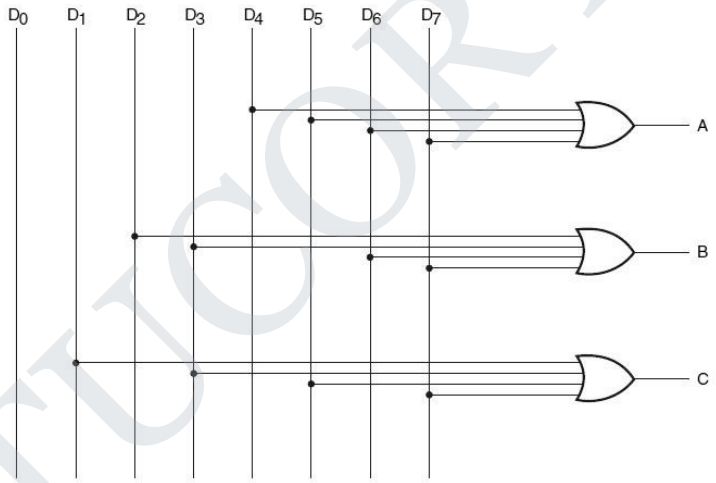
Therefore the Boolean Expression is

$A = D_4+D_5+D_6+D_7$

$B = D_2+D_3+D_6+D_7$

$C = D_1+D_3+D_5+D_7$

Step 3: Logic circuit



56. Design the decimal to BCD Encoder.

Step 1: Truth Table

D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	A	B	C	D
1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0	0	0	0	1	0	1
0	0	0	0	0	0	1	0	0	0	0	1	1	0
0	0	0	0	0	0	0	1	0	0	0	1	1	1
0	0	0	0	0	0	0	0	1	0	1	0	0	0
0	0	0	0	0	0	0	0	0	1	1	0	0	1

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Step 2: Boolean Expression

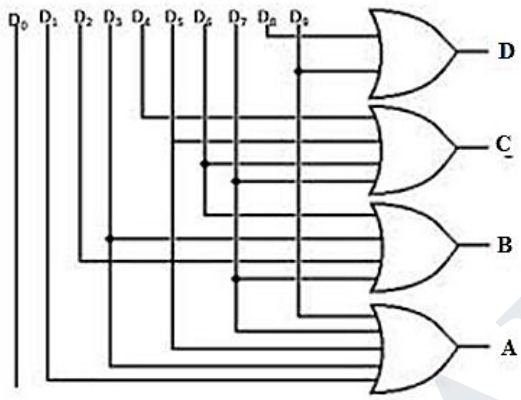
$A=D_8+D_9$

$B=D_4+D_5+D_6+D_7$

$C=D_2+D_3+D_6+D_7$

$D=D_1+D_3+D_5+D_7+D_9$

Step 3: Logic circuit



57. Define Decoder and Design the 2 to 4 decoder.

- Decoder is acronym which decodes the given binary input into information in digital form.
- A binary code of n bits is capable of representing up to 2^noutput.
- A decoder is a combination circuit which converts binary information from n input lines to 2^n output lines.
- The decoder is generally represented as n to m lines to 2^n output lines.

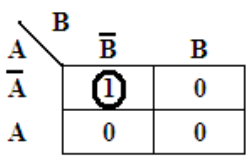
Design of the 2 to 4 decoder:

Step 1: Truth Table

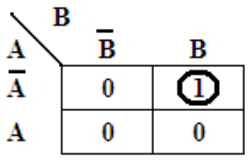
- In 2X4 decoder, the two inputs are decoded into 4 output. Each output is representing one of the minterm of the two input variable.

INPUT			OUTPUT			
E	**A**	**B**	**D0**	**D1**	**D2**	**D3**
0	**0**	**0**	**1**	**0**	**0**	**0**
0	**0**	**1**	**0**	**1**	**0**	**0**
0	**1**	**0**	**0**	**0**	**1**	**0**
1	**X**	**X**	**0**	**0**	**0**	**1**

Step 2: Boolean Expression using K-map



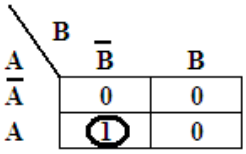
$D0 = \bar{A}\,\bar{B}$



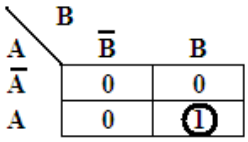
$D1 = \bar{A}.B$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

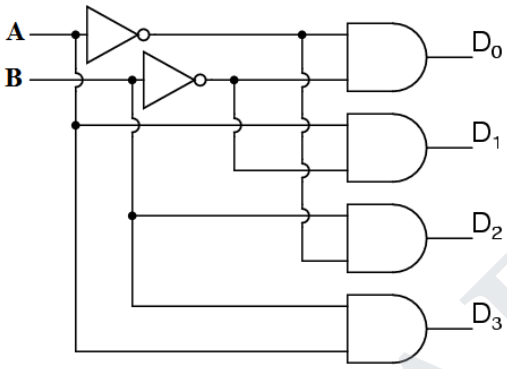


D2 = A $\overline{B}$



D3 = A B

Logic diagram:



58. Design the 3 to 8 decoder. [OR] **Design a 3x8 decoder and explain its operation as a minterm generator.** *[May'17]*

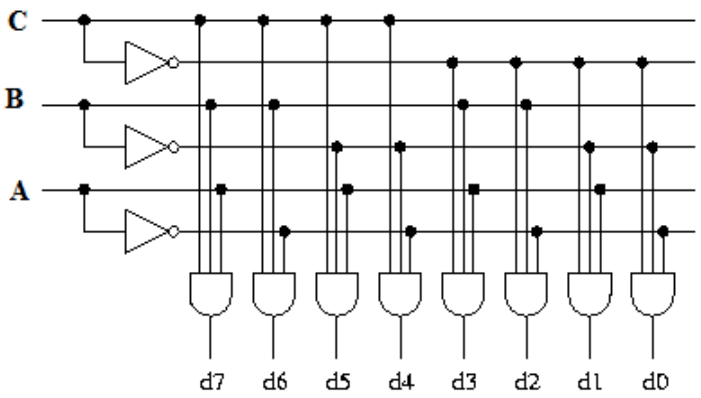
Step 1: Truth Table

INPUTS			OUTPUTS							
A	**B**	**C**	**D0**	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Step 2: Boolean Expression

$$Y = \overline{ABC} + \overline{AB}\,C + \overline{A}\,B\,\overline{C} + \overline{A}\,B\,C + A\,\overline{BC} + A\,\overline{B}\,C + A\,B\,\overline{C} + A\,B\,C$$

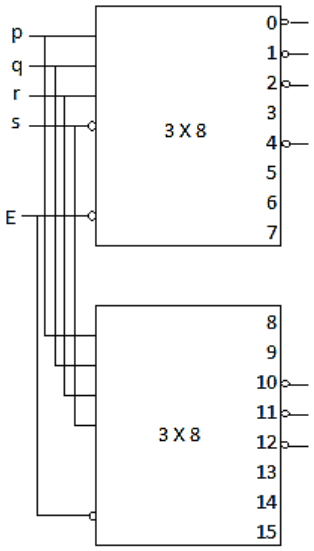
Step 3: Logic Circuit



STUCOR APP
DOWNLOADED FROM STUCOR APP

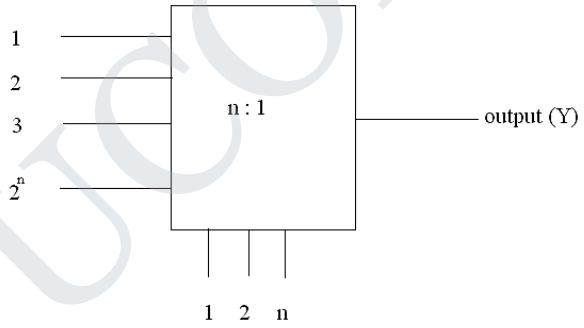
DOWNLOADED FROM STUCOR APP

59. Implement the function F (p, q, r, s) =Σ (0, 1, 2, 4, 7, 10, 11, 12) using Decoder. *[Nov'14]*



60. Define Multiplexer and design the 2:1 Multiplexer.

- Multiplexer is one which gives the output based on the selection of input.
- Multiplexer has many input and one output.
- It gives the output based on select line.
- A digital multiplexer is combinational circuit which select one line from many input lines and direct to output, the selection of input is controlled by select lines or address lines.
- For multiplexer there are 2^n input lines and n select lines.



Block Diagram

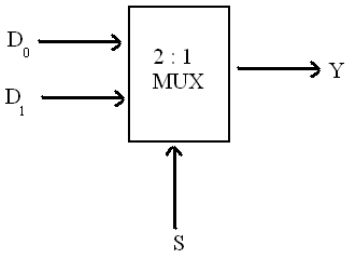
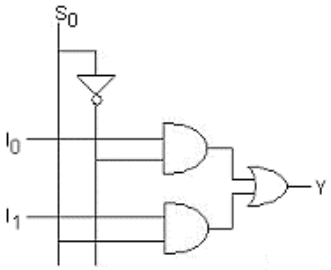


Table:

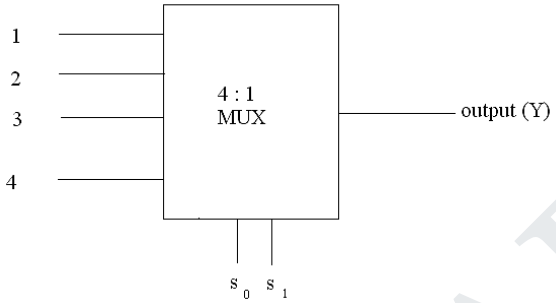
	Select	Output
EN	**S0**	**Y**
0	X	0
1	0	I_0
1	1	I_1

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



61. Design the 4:1 Multiplexer

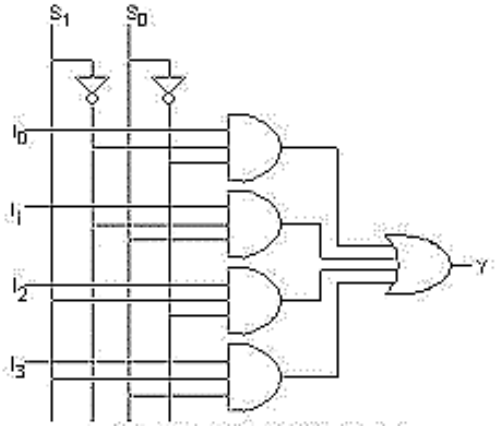


	Select		Output
EN	**S0**	**S1**	**Y**
0	X	X	0
1	0	0	D0
1	0	1	D1
1	1	0	D2
1	1	1	D3

Truth Table:

A	**B**	**C**	**Y**	**Output**
0	0	0	0	Y = C
0	0	1	1	
0	1	0	0	Y =0
0	1	1	0	
1	0	0	1	Y = 1
1	0	1	1	
1	1	0	0	Y = C
1	**1**	**1**	**1**	

Logic diagram:



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

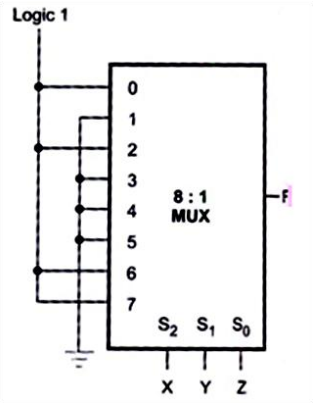
62. Implement the given function using Multiplexer F(x,y,z) = ∑(0,2, 6,7) [*May'12]*

Step 1: Select the multiplexer. Here, Boolean expression has 3variables, thus we require 23 = 8:1 multiplexer.

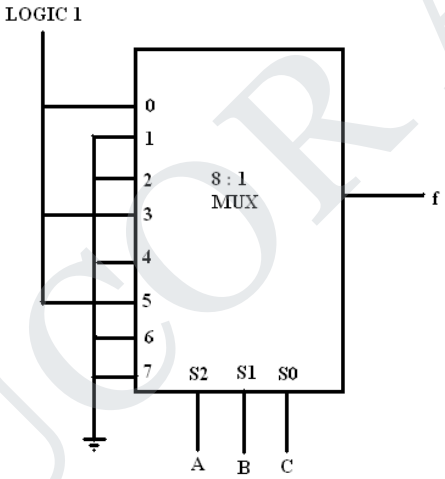
Step 2: Connect inputs corresponding to the present minterms to logic 1.

Step 3: Connect remaining inputs to logic 0.

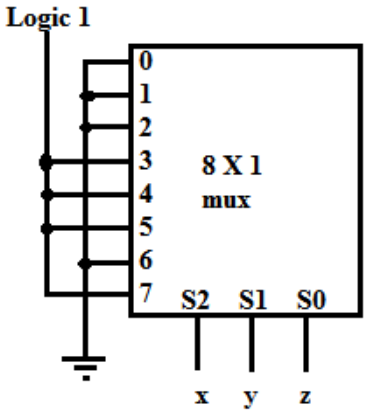
Step 4: Connect input variables to select lines of MUX.



63. Implement the following function f (A,B,C) = ∑ m(0,3,5) using 8:1 MUX. *[Nov'12]*



64. Implement the following function using a suitable multiplexer. F (a,b,c) = ∑m(3,7,4,5) *[May'17]*

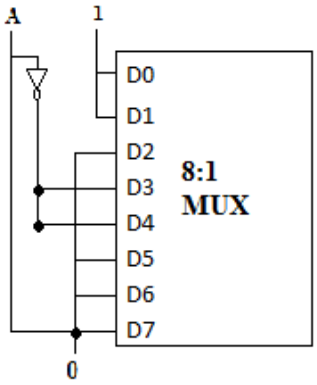


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

65. Implement the following Boolean function using 8:1 MUX: F(A,B,C,D) = Σm(0,1,3,4,8,9,15). *[Apr'15]*

	D0	D1	D2	D3	D4	D5	D6	D7
$\overline{A}$	(0)	(1)	2	(3)	(4)	5	6	7
A	(8)	(9)	10	11	12	13	14	(15)
	1	1	0	$\overline{A}$	$\overline{A}$	0	0	A



66. What is a multiplexer? Implement the following Boolean function with 8 x 1 MUX and external gates. F (A, B, C, D) = ∑m (1, 3, 4, 11, 12, 13, 14, 15). *[Nov'15]*

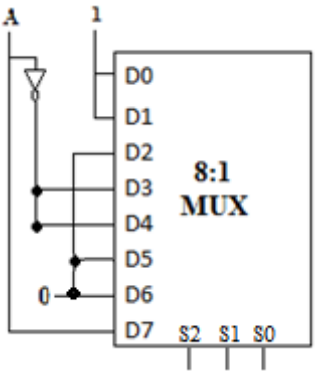
Multiplexer:

- A multiplexer or MUX is a device that performs multiplexing; it selects one of many analog or digital input signals and forwards the selected input into a single line.
- A multiplexer of 2n inputs has n select lines, which are used to select which input line to send to the output.

Implementation:

	D0	D1	D2	D3	D4	D5	D6	D7
$\overline{A}$	0	(1)	2	(3)	(4)	5	6	7
A	8	9	10	(11)	(12)	(13)	(14)	(15)
	0	$\overline{A}$	0	1	1	A	A	A

Logic circuits:



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

67. Implement full subtractor using demultiplexer. *[May'12]*

Step 1: Write the truth table of full subtractor.

Step 2: Represent output of subtractors in minterm form.

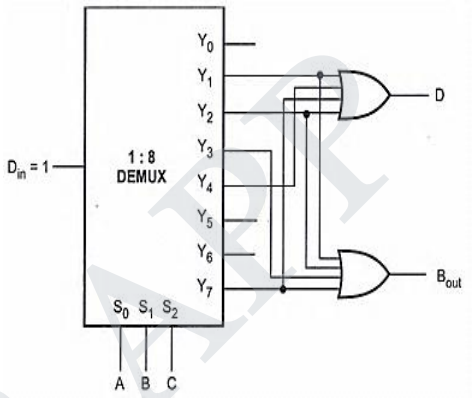
$D = f(A,B,C) = \sum m(1,2,4, 7)$

$B_{out} = f(A,B,C) = \sum m(1,2,3,7)$

Step 3: Logically OR the outputs corresponding to minterms

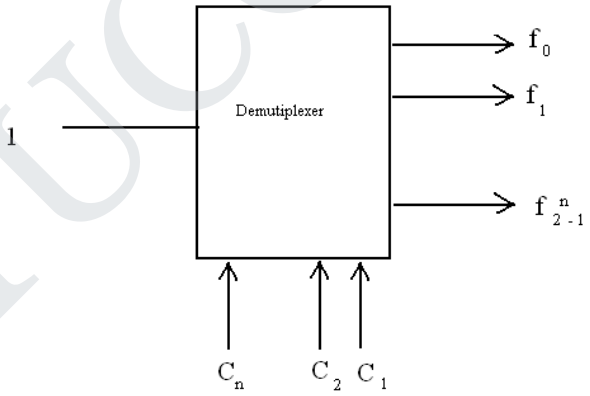
- With D_{in} input 1, demultiplexer gives minterms at the output so minterms at the output by logically ORing required minterms

A	B	BIN	D	BOUT
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



68. What is a Demultiplexer and Design the 1 to 4 Demultiplexer

- Demultiplexer is one which gives many outputs for single input it says to which output the input is transmitted.
- The selection of output is done by address lines or selection lines.



1 to 4 De-Multiplexer

- A 1 to 4 De Multiplexer is having '1' input and 4 output lines.
- The address line for 1 X 4 De Multiplexer is 2 Select lines.

Table

Data Input	Select Inputs		Outputs			
	S1	S2	Y3	Y2	Y1	Y0
I	0	0	0	0	0	1
I	0	1	0	0	1	0
I	1	0	0	1	0	0
I	1	1	1	0	0	0

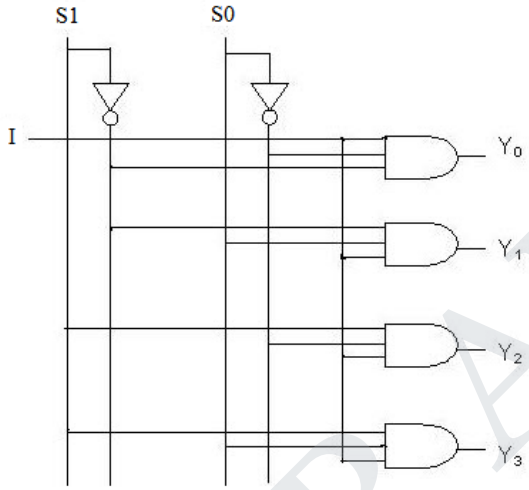
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- From the truth table in order to connect or transmit the input to first line (i.e.,) Y_0 line the necessary address line is $S_1 = 0$ and $S_0 = 0$.
- Similarly for all other output, if we want to transmit the data to second (i.e.,) Y_1 line $S_1 = 0$ and $S_0 = 1$.
- The outputs are

$$Y_0 = I\,\overline{S}_1\overline{S}_0 Y_2 = I\,S_1\overline{S}_0$$
$$Y_1 = I\,\overline{S}_1 S_0 Y_3 = I\,S_1 S_0$$

Logic Circuit



69. Design the 1 to 8 De Multiplexer.

Data Input	Select Inputs			Outputs							
	S0	S1	S2	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
I	0	0	0	0	0	0	0	0	0	0	1
I	0	0	1	0	0	0	0	0	0	1	0
I	0	1	0	0	0	0	0	0	1	0	0
I	0	1	1	0	0	0	0	1	0	0	0
I	1	0	0	0	0	0	1	0	0	0	0
I	1	0	1	0	0	1	0	0	0	0	0
I	1	1	0	0	1	0	0	0	0	0	0
I	1	1	1	1	0	0	0	0	0	0	0

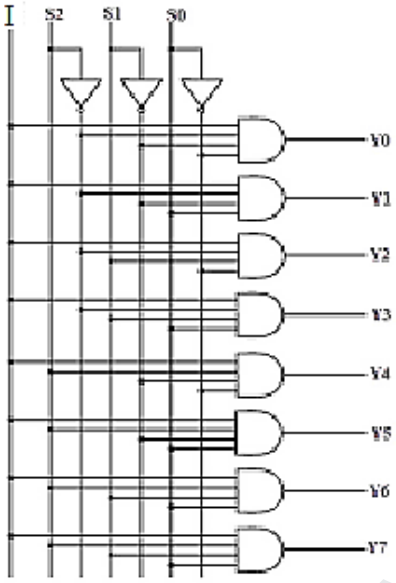
Boolean Expression

$$Y_0 = I\,\overline{S}_2\overline{S}_1\overline{S}_0$$
$$Y_1 = I\,\overline{S}_2\overline{S}_1 S_0 Y_5 = I\,S_2\overline{S}_1 S_0$$
$$Y_2 = I\,\overline{S}_2 S_1\overline{S}_0 Y_6 = I\,S_2 S_1\overline{S}_0$$
$$Y_3 = I\,\overline{S}_2 S_1 S_0 Y_7 = I\,S_2 S_1 S_0$$
$$Y_4 = I\,S_2\overline{S}_1\overline{S}_0$$

STUCOR APP
DOWNLOADED FROM STUCOR APP

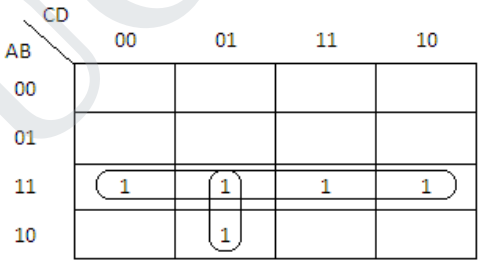
DOWNLOADED FROM STUCOR APP

Logic circuit



70. Plot the logical expression ABCD+A$\overline{BCD}$ +A$\overline{B}$C+AB on a 4-variable k-map, obtain the simplified expression from the map. *(Nov'17)*

F= ABCD+A$\overline{BCD}$ +A$\overline{B}$C+AB
= ABCD+A$\overline{BCD}$ +A$\overline{B}$C [D+$\overline{D}$] + AB [C+$\overline{C}$] [D+$\overline{D}$]
= ABCD+A$\overline{BCD}$ + A$\overline{B}$CD+ A$\overline{B}$C$\overline{D}$ +ABC [D+$\overline{D}$]+ AB$\overline{C}$[D+$\overline{D}$]
= ABCD+A$\overline{BCD}$ + A$\overline{B}$CD+ A$\overline{B}$C$\overline{D}$ + ABCD+ABC$\overline{D}$+ AB$\overline{C}$D+ AB$\overline{CD}$
=1111+1000+1011+1010+1111+1110+1101+1100
= [15,8,11,15,14,13,12]
= [8,11,12,13,14,15]

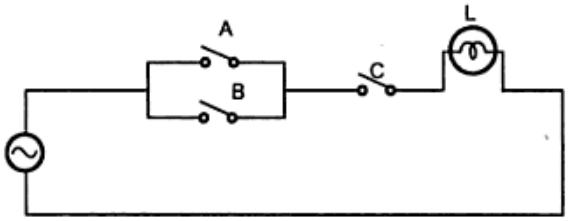


$F = AB + A\bar{C}D$

ANNA UNIVERSITY QUESTIONS

PART A

1. Explain the following switching circuits in binary logic notation. *[May' 12]*



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

2. Why NAND and NOR are called Universal Gates?*[Nov'13, Nov'10]*
3. State De Morgan's theorem. *[Nov'09, May'10, Nov'10, Apr'11, Nov'12, May'13]*
4. Simplify the expression Z = AB + AB'. (A'C')'. *[Apr'15]*
5. Find the compliment of F = x (y'z'+yz). *[Nov'12]*
6. Show that a) A+A'B=A+B and b) X'Y'Z+X'YZ+XY' = X'Z+XY' *[Apr'10]*
7. What is meant by karnaugh map or K-Map method? *(Nov'17, Apr'18)*
8. What are don'tcare conditions? How does it help K-map for circuit simplification? *[Nov'11]*
9. Define Duality Theorem. [OR] Define Duality Property. *(Apr'18)*
10. Given F = B' + A'B +A'C': Identify the redundant term using K-Map. *[Nov'14]*
11. Convert the given expression in canonical SOP form Y = AC + AB + BC. *[Apr'15]*
12. Convert the given expression in canonical SOP form Y = AB + A'C + BC'. *[Nov'15]*
13. Convert the given expression in canonical SOP form $Y = AB + \bar{A}C + B\bar{C}$ *(Nov'16)*
14. Draw the logical diagram of EX – OR gate using NAND gates. *[Nov'15]*
15. Realize an OR gates using NAND gates? *[Apr'10, Nov'12, May'13]*
16. Write the POS representation of the following SOP function.*(May'16)*
17. Write the POS representation of the following SOP function$F(x, y, z) = \bar{x}yz + xy\bar{z} + x\bar{y}z$ *(May'17)*
18. Sketch a half-adder using logic gates. Draw the truth table. *[Apr'10, Nov'12]*
19. Implement full adder with two half adders. *[Nov'12]*
20. Design a half subtractor. *[May'16, May'17]*
21. Define Multiplexer. *[May'13]*
22. Why MUX is called as Data Selector? *[Apr'11]*
23. Draw the truth table of 2:1 MUX. *(Nov'16)*
24. What is priority encoder? *[May'12]*
25. Difference between Decoder & Demux. *[Nov'11] [OR]* Compare decoder and demultiplexer. *(Nov'17)*
26. Give one application each for Multiplexer and Decoder. *[Nov'14]*

PART B & C [8/16 Marks]

1. Prove that F= $\overline{A}$.B + A.$\overline{B}$is exclusive OR operation and it equals $\overline{\overline{(A.B)}.A}.\overline{\overline{(A.B)}.B}$*[May'13]*
2. Write down the steps in implementing a Boolean function with levels of AND gates. *(Apr'18)*
3. Reduce the following expression using K-map f = x'y'z + w'xz + wxyz' + wxz + w'xyz. *[May'11]*
4. Simplify the Boolean function using K – map and implement using only NAND gates.
5. F (A, B, C, D) = ∑m (0, 8, 11, 12, 15) + ∑d (1, 2, 4, 7, 10, 14). Mark the essential and non – essential prime implicants. *[Nov'15]*
6. Give the general procedure for converting a Boolean expression in to multilevel NAND diagram. *(Apr'18)*
7. Simplify and implement the following SOP function using NOR gates f(A,B,C,D) = ∑ m(0,1,4,5,10,11,14,15) *[May'12]*
8. Reduce the following function using K-map, f(A,B,C,D) = π M(0,2,3,8,9,12,13,15) *[Apr'15]*
9. 17. Express the function F=A+B'C in Canonical SOP form and Canonical POS form [*Nov'13, Nov'17]*
10. Simplify the following function using karnaugh map. F(w,x,y,z) = ∑m(0,1,3,9,10,12,13,14) + ∑d(2,5,6,11) *(May'16)*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

11. Implement the following function using only NAND gates .F(x, y, z) = ∑m (0, 2, 4, 6) *(May'16)*
12. Reduce the following minterms using k-map. F(w,x,y,z) =∑ m(0,1,3,5,6,7,8,12,14)+∑d(9,15) *[May'17]*
13. Simplify the Boolean function using K-map F (W,X,Y,Z) =∑ (1,3,7,11,15) which has the don't care conditions d (W,X,Y,Z)= ∑(0,2,5). *[May'13]*
14. Simplify using k-map F(A,B,C,D) = ∑ m(7,8,9) + d(10,11,12,13,14,15) *[Nov'13]*
15. Simplify using K map F(W,X,Y,Z) = ∑(0,1,2,4,5,6,8,9,12,13,14) *[Apr'10]*
16. Give the simplified expression for the following logic equation where d represents don't care condition. F(A,B,C,D)=∑m(0,8,11,12,15) + d(1,2,4,7,10,14). Represent the simplified expression using logic gates. *[Nov'11]*
17. Minimize the function F (a, b, c, d) = Σ (0, 4, 6, 8, 9, 10, 12) with d = Σ (2, 13). Implement the function using only NOR gates. *[Nov'14]*
18. Using K-map simplify the following function and implement the function using logic gates f(A, B, C) = πM (0, 4, 6) *[Nov'12]*
19. Simplify the logical expression using K-map in SOP and POS form. F(A,B,C,D)=∑m(0,2,3,6,7)+d(8,10,11,15). *(Nov'16)*
20. Prove that for constructing XOR from NAND's we need 4 NAND gates. *[May'13]*
21. Implement the following Boolean function with NAND-NAND logic. Y = AC + ABC + A'BC + AB + D *[May'12]*
22. Write brief notes on the following: i) Demorgan's theorem ii) Comparators iii) Binary to gray code converter. *[Nov'11]*
23. Explain how to convert Binary Coded Decimal (BCD) to Excess – 3 codes?*[Apr'15]*
24. Design a 4 bit BCD to Excess 3 code converter and implement using logic gates. *[Nov'15, May'16]*
25. Design a 4-bit Binary to gray code converter and implement it using logic gates? *[Nov'14]*
26. Design a 4 bit gray code to binary converter and express using logic gates. *(Nov'17)*
27. Design a decimal adder to add two decimal digits *[Apr'10]*
28. Design a full adder using two half-adders and an or gate. *[Apr'15]*
29. Design a Full adder and implement it using suitable multiplexer. *[May'16]*
30. Design a full adder using only NOR gate. *[May'17]*
31. Explain Half Subtractor & Full Subtractor Circuit. *[Nov'13, Nov'14]*
32. Design a full subtractor and implement using logic gates. *[Nov'15]*
33. Design a full subtractor and realize using logic gates. Also implement the same using half subtractors. *(Nov'16)*
34. Design a 3x8 decoder and explain its operation as a minterm generator. *[May'17]*
35. Implement the function F (p, q, r, s) =Σ (0, 1, 2, 4, 7, 10, 11, 12) using Decoder. *[Nov'14]*
36. Implement the given function using Multiplexer F(x,y,z) = ∑(0,2, 6,7) [*May'12]*
37. Implement the following function f (A,B,C) = ∑ m(0,3,5) using 8:1 MUX. *[Nov'12]*
38. Implement the following function using a suitable multiplexer. F (a,b,c) = ∑m(3,7,4,5) *[May'17]*
39. Implement the following Boolean function using 8:1 MUX: F(A,B,C,D) = Σm(0,1,3,4,8,9,15). *[Apr'15]*
40. What is a multiplexer? Implement the following Boolean function with 8 x 1 MUX and external gates. F (A, B, C, D) = ∑m (1, 3, 4, 11, 12, 13, 14, 15). *[Nov'15]*
41. Implement full subtractor using demultiplexer. *[May'12]*
42. Plot the logical expression ABCD+A$\overline{BCD}$ +A$\overline{B}$C+AB on a 4-variable k-map, obtain the simplified expression from the map. *(Nov'17)*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

UNIT – III PART A

1. Difference between Combinational & Sequential Circuits. *[Nov'13]*

Sl.No.	Combinational Circuits	Sequential Circuits
1	The output at all times depends only on the present combination of input variables.	The output not only depends on the present input but also depends on the past history of input variables.
2	Memory unit is not required.	Memory unit is required to store the pastinput history of input variable.
3	Clock input is not needed.	Clock input is needed.
4	Process is faster.	Process is slower.
5	Easy to design.	Difficult to design.
6	Eg: Mux, Demux, Encoder, Decoder, Adders, Subtractors.	Eg: Shift Registers, Counters.

2. Define Flip flop.

- The basic unit for storage is flip flop.
- A flip-flop maintains its output state either at 1 or 0until directed by an input signal to change its state.

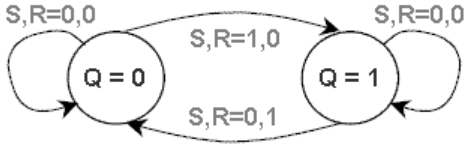
3. What are the different types of flip-flop?

There are various types of flip flops. Some of them are mentioned below they are:

1. SR flip-flop,
2. D flip-flop,
3. JK flip-flop,
4. T flip-flop

4. Draw the truth table and state diagram of SR flip flop. *[Nov'15]*

CLK	S	R	Q_n	Q_{n+1}	State
1	0	0	0	0	No Change (NC)
1	0	0	1	1	
1	0	1	0	0	Reset
1	0	1	1	0	
1	1	0	0	1	Set
1	1	0	1	1	
1	1	1	0	x	Indeterminate (*)
1	1	1	1	x	



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

5. What is the drawback of SR flipflop? *[May'14]*

- When we provide two input as 1 or true or high output of q and q' become same that violate the complement law.

6. Write the truth table of D flip-flop.

EN	D	Q_n	Q_{n+1}	State
1	0	x	0	Reset
1	1	x	1	Set
0	X	x	Q_n	No Change (NC)

7. Write the truth table of JK flip-flop.

The truth table of JK Flip-Flop is given below.

CLK	Inputs		Output	State
	J	K	Q_{n+1}	
1	0	0	Q_n	No Change
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	Q_n'	Toggle

8. Write the truth table of T flip-flop.

The truth table of T Flip-Flop is given below.

T	Q_{n+1}	State
0	Q_n	No Change
1	Q_n'	Toggle

9. Define race around condition. *[Nov'10, Nov'11]* [OR] **What do you mean by race around condition in a flip flop?** *(Nov'17)*

- In JK flip-flop output is fed back to the input.
- Therefore change in the output results change in the input.
- Due to this in the positive half of the clock pulse if both J and K are high then output toggles continuously.
- This condition is called 'race around condition'.

10. What are the various types of triggering of flip-flop?

- Level triggering
  - High level
  - Low level
- Edge triggering
  - Positive edge
  - Negative edge

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

11. What is edge-triggered flip-flop? *[Nov'15]*

- The problem of race around condition can solved by edge triggering flip flop.
- The term edge triggering means that the flip-flop changes state either at the positive edge or negative edge of the clock pulse and it is sensitive to its inputs only at this transition of the clock.

12. Write the characteristic table for SR flip-flop. *[Apr'11]* [OR] **Give characteristics equation and characteristics table of SR flip flop.** *[May 16]*

Q_n	Q_{n+1}	R	S
0	0	X	0
1	1	0	1
0	1	1	0
	0	0	X

$$Q_{n+1} = S + \bar{R}Q_n$$

13. Derive the characteristic equation of D FF.

Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

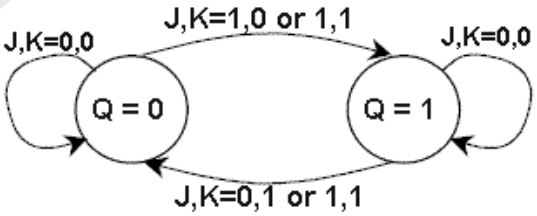
$Q_{n+1} = D$

14. Write down the characteristic table of JK FF and state diagram. *[Apr'11, May'12, May'13]* [OR] **Draw the state diagram of JK flip flop**. *[Nov'16]*

Characteristic table:

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

State diagram:



15. The JK flip-flop is a universal flip-flop. Justify. *[Nov'12]*

- Using JK flip-flop we can construct other flip-flop like SR, D and T.
- So it is called as universal flip-flop.

16. What is a master-slave flip-flop? Mention the major applications of Master Slave FF. *[Nov'10]*

- A master-slave flip-flop consists of two flip-flops, where first circuit serves as a master flip flop and other flip flops acts as a slave.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

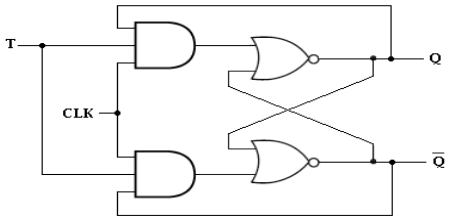
- The input is given to the first flip flop (Master) and the second flip flop (slave) onwards it must get the input from corresponding first one only.
- The master slave flip flop is used as a solution to the race around problem in flip flops.

17. Give the characteristics equation and characteristic table of a T flip flop. *[May'17]*

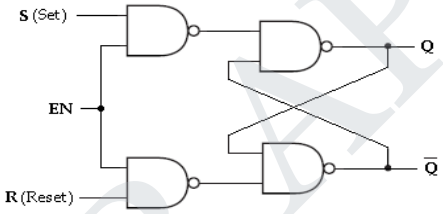
T	$Q_{(NEXT)}$
0	Q
1	Q'

Q (next) = TQ' + T'Q

18. Draw the logic diagram for T flip flop and SR flip-flop. *[Apr'10]*



T Flip flop



SR Flip flop

19. How will you convert JK FF into DFF? *[Nov'14]*

Input	Present state	Next state	Flip-Flop Inputs	
D	**Q_n**	**Q_{n+1}**	**J**	**K**
0	0	0	0	x
0	1	0	x	1
1	0	1	1	x
1	1	1	x	0

K-map simplification

For J

D \ Q_n	0	1
0	0	x
1	1	x

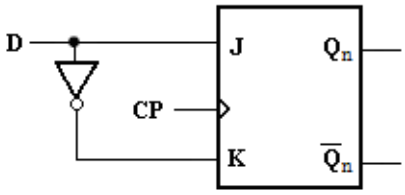
J = D

For K

D \ Q_n	0	1
0	x	1
1	x	0

K = $\overline{D}$

Logic diagram



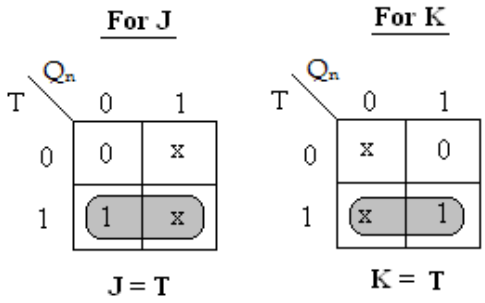
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

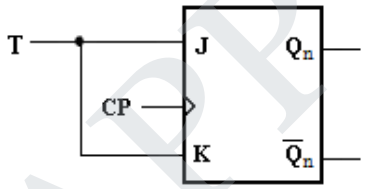
20. Convert JK flip-flop to T flip-flop. *[Nov'10, Nov'14]*

Input	Present state	Next state	Flip-flop inputs	
T	Q_n	Q_{n+1}	J_A	K_A
0	0	0	0	X
0	1	1	X	0
1	0	1	1	X
1	1	0	X	1

K-map simplification



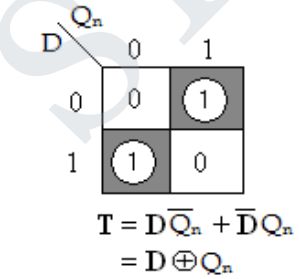
Logic diagram



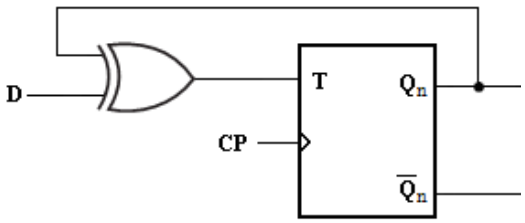
21. Convert T Flip Flop to D Flip Flop. *[Apr'15]*

Input	Present state	Next state	Flip-Flop Input
D	Q_n	Q_{n+1}	T
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

K-map simplification



Logic diagram



22. What are the applications of Flip-Flop?

- Used as a memory Element.
- Used as a Delay Element.
- Used as a basic building block in sequential circuits such as counters and registers.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- Data Transfer.
- Frequency Division &Counting.

23. Define registers.

- A register is a group of flip-flops; flip-flop can store one bit information.
- So an n-bit register has a group of n flip-flops and is capable of storing any binary information/number containing n-bits.

24. Define shift registers.

- The binary information in a register can be moved from stage to stage within the register or into or out of the register upon application of clock pulses.
- This type of bit movement or shifting is essential for certain arithmetic and logic operations used in microprocessors.
- This gives rise to group of registers called shift registers.

25. What are the different types of shift registers?

There are five types.

1. Serial In Serial Out Shift Register
2. Serial In Parallel Out Shift Register
3. Parallel In Serial Out Shift Register
4. Parallel In Parallel Out Shift Register
5. Bidirectional Shift Register

26. Define Shift Register Counter.

- A shift register can also be used as a counter.
- A shift register with the serial output connection back to the serial input is called Shift register counter.

27. What are the two types of shift register counters?

There are 2 types of shift register counters:

i) Ring counter:

A ring counter is a circular shift register with only one flip flop being set, at any particular time, all others are cleared.

ii) Johnson counter:

The Johnson counter is a K-bit switch-tail ring counter with 2^k decoding gates to provide outputs for 2^ktiming signals.

28. What are the applications of shift register?

- Shift register act as sequence generator
- Used in delay lines.
- Used in counters.
- Used as a memory element.

29. What is bi-directional shift register and unidirectional shift register?

- A register capable of shifting both right and left directions are called as bi-directional shiftregister.
- A register capable of shifting only in one direction left or right directions are called unidirectional shift register.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

30. Define sequential circuit? *[Apr'10]*

- In sequential circuits the output variables dependent not only on the present input variables but they also depend up on the past output of these input variables.

31. What are the classifications of sequential circuits?

- The sequential circuits are classified on the basis of timing of their signals into two types.
  1) Synchronous sequential circuit.
  2) Asynchronous sequential circuit.

32. What do you mean by present state?

- The information stored in the memory elements at any given time defines the present state of the sequential circuit.

33. What do you mean by next state?

- The present state and the external inputs determine the outputs and the next state of the sequential circuit.

34. Define synchronous sequential circuit. *[May'14]*

- In synchronous sequential circuits, signals can affect the memory elements only atdiscrete instant of time.

35. Give the comparison between synchronous & Asynchronous counters. *(Apr'18)*

S.No	Asynchronous counters	Synchronous counters
1	Counter flip-flops are connected in such a way that output of 1st flip-flop drives the clock for the next flip - flop.	No connection between output of first flip-flop and clock input of the next flip – flop.
2	All the flip-flops are Not clocked simultaneously	All the flip-flops are clocked Simultaneously
3	Logic circuit is very simple even for more number of states.	Design involves complex logic circuit as number of states increases.
4	Counter are low speed	Counter are high speed

36. What are Steps or Design procedure for Synchronous Counter?

- State Diagram.
- State Table.
- State Assignment.
- Excitation Table (Consider which Memory Unit Using K-Map)
- Circuit Diagram.

37. How many flip flops are required for designing synchronous mod 25 counter? *[May'13]*

$2^n > 25 \Longrightarrow n=5.$

- Therefore 5flip-flops are required for designing synchronous mod 25 counter.

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

38. What are synchronous sequential circuits?

- In synchronous circuit the designer has to consider the time delays involved to determine the maximum operating speed of the clock.

39. What is a state equation?

- A state equation also called, as an application equation is an algebraic expression that specifies the condition for a flip-flop state transition.
- The left side of the equation denotes the next state of the flip-flop and the right side; a Boolean function specifies the present state.

40. What are the state table and state diagram as applicable to sequential logical circuits? *[Nov'12]*

State table:

- A table, which consists of time sequence of inputs, outputs and flip-flop states, is called state table.
- Generally it consists of three sections: present state, next state and output.

State diagram:

- A graphical representation of a state table is called a state diagram.

Uses: i) Behavior of a state machine can be analyzed rapidly.
ii) It can be used to design a machine from a set of specification.

41. Differentiate between Mealy and Moore models. *[Nov'14, May'16, Nov'16, May'17]*

Sl.No:	Moore Circuit	Mealy Circuit
1	Its output is a function of Present state only.	Its output is a function of present state as well as present input.
2	Input changes does not affect the output	Input changes may affect the output of the circuit
3	Moore circuit requires more number of states for implementing same function	It requires less number of states for implementing same function

42. How does the state transition diagram of a Moore model differ from Mealy model? *[Nov'11]*

- A Moore model requires more number of states for implementing same functions in mealy model.

43. What is lockout? How it is avoided? *[May'12]*

- In a counter if the next state of some unused state is again a used state.
- If by chance the counter happens to find itself in the unused states and never arrived at a used state then the counter is said to be in the lockout condition.
- To avoid lockout the counter should be provided with an additional logic circuitry which will force the counter from an unused state to next state as initial state

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

44. What is meant by State assignment? *[Nov'13]*

- In sequential circuits, the behavior of the circuit is defined in terms of its inputs, present states, next states and outputs.
- To generate desired next state at particular present state and inputs, it is necessary to have specific flip-flop inputs.
- These flip-flop inputs are described by a set of Boolean functions called flip-flop input functions.
- To determine the flip-flop input functions, it is necessary to represent states in the state diagram using binary values instead of alphabets.
- This procedure is known as State assignment.

45. State the rules for state assignment. *[Apr'15]*

Rule 1: States having the same NEXT STATES for a given input condition should have assignments which can be grouped into logically adjacent cells in a K-map.

Rule 2: States that are the NEXT STATES of a single state should have assignment which can be grouped into logically adjacent cells in a K-map.

46. Define rise time.

- The time required to change the voltage level from 10% to 90% is known as rise time (t_r).

47. Define fall time.

- The time required to change the voltage level from 90% to 10% is known as fall time (t_f).

48. Define skew and clock skew.

- The phase shift between the rectangular clock waveforms is referred to as skew and the time delay between the two clock pulses is called clock skew.

49. Define setup time.

- The setup time is the minimum time required to maintain a constant voltage levels at the excitation inputs of the flip-flop device prior to the triggering edge of the clock pulse in order for the levels to be reliably clocked into the flip flop.
- It is denoted as $\mathbf{t}_{setup}$.

50. Define hold time.

- The hold time is the minimum time for which the voltage levels at the excitation inputs must remain constant after the triggering edge of the clock pulse in order for the levels to be reliably clocked into the flip flop.
- It is denoted as hold.

51. Define propagation delay.

- A propagation delay is the time required to change the output after the application of the input.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

52. Define address and word. *(Apr'18)*

- In a ROM, each bit combination of the input variables is called address.
- Each bit combination that comes out of the output lines is called a word.

54. What is a preset table counter and ripple counter? *(Nov'17)*

Pressettable Counter:

- Presettable synchronous 4-bit binary counter; synchronous reset.
- Synchronous presettable binary counter with an internal look-head carry.
- Synchronous operation is provided by having all flip-flops clocked simultaneously on the positive-going edge of the clock (CP).

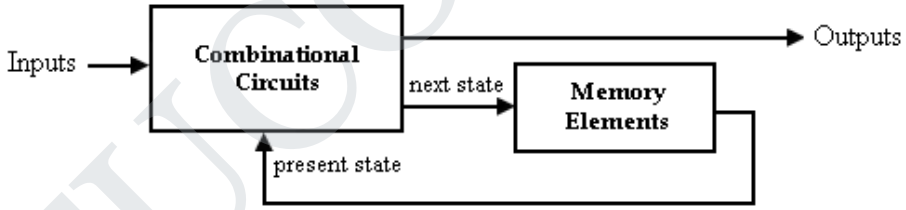
Ripple Counter:

- A ripple counter is an asynchronous counter where only the first flip-flop is clocked by an external clock.
- All subsequent flip-flops are clocked by the output of the preceding flip-flop.

PART B & C [8 /16 Marks]

1. What is sequential circuit? Give an example.

- In sequential logic circuits, it consists of combinational circuits to which storage elements are connected to form a feedback path.
- The storage elements are devices capable of storing binary information either 1 or 0.
- The information stored in the memory elements at any given time defines the present state of the sequential circuit.
- The present state and the external circuit determine the output and the next state of sequential circuits.



Sequential Circuit- Block Diagram

- Thus in sequential circuits, the output variables depend not only on the present input variables but also on the past history of input variables.
- The rotary channel selected knob on an old-fashioned TV is like a combinational. Its output selects a channel based only on its current input – the position of the knob.
- The channel-up and channel-down push buttons on a TV is like a sequential circuit. The channel selection depends on the past sequence of up/down pushes.

The comparison between combinational and sequential circuits is given in table below.

Sl.No:	Combinational logic	Sequential logic
1	The output variable, at all times depends on the combination of input variables.	The output variable depends not only on the present input but also depend upon the past history of inputs.
2	Memory unit is not required	Memory unit is required to store the past history of input variables.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

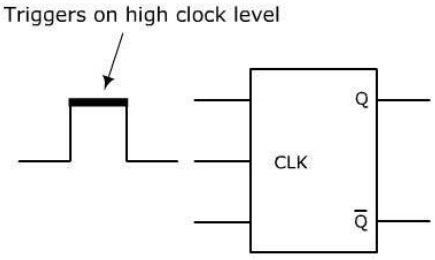
3	Faster in speed	Slower than combinational circuits.
4	Easy to design	Comparatively harder to design.
5	Eg. Adder, Parity generator, Magnitude comparator	Eg. Shift registers, Counters

2. Explain the various types of triggering with suitable diagrams. Compare their merits and demerits. *[Nov'14, Nov'17]*

- 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.
- Flip flops are applicable in designing counters or registers which stores data in the form of multi-bit numbers.
- But such registers need a group of flip flops connected to each other as sequential circuits. And these sequential circuits require trigger pulses.
- The number of trigger pulses that is applied to the input of the circuit determines the number in a counter.
- A single pulse makes the bit move one position, when it is applied onto a register that stores multi-bit data.
- In the case of SR Flip Flops, the change in signal level decides the type of trigger that is to be given to the input.
- But the original level must be regained before giving a second pulse to the circuit.
- There are mainly four types of pulse-triggering methods.

1. High Level Triggering

- When a flip flop is required to respond at its HIGH state, a HIGH level triggering method is used.
- It is mainly identified from the straight lead from the clock input.

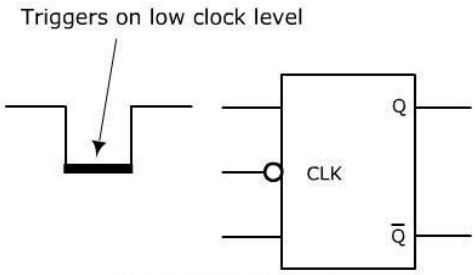


2. Low Level Triggering

- When a flip flop is required to respond at its LOW state, a LOW level triggering method is used.
- It is mainly identified from the clock input lead along with a low state indicator bubble.

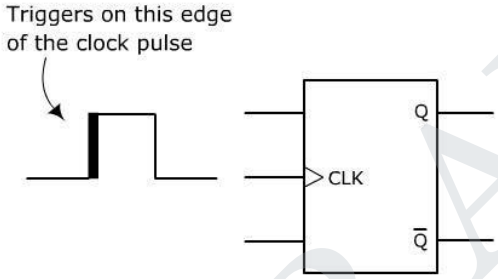
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



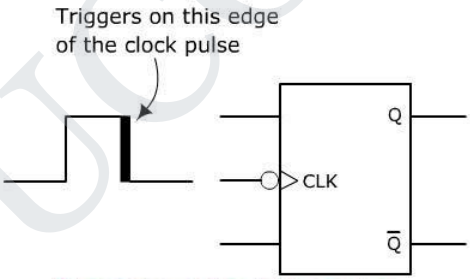
3. Positive Edge Triggering

- When a flip flop is required to respond at a LOW to HIGH transition state, POSITIVE edge triggering method is used.
- It is mainly identified from the clock input lead along with a triangle.



4. Negative Edge Triggering

- When a flip flop is required to respond during the HIGH to LOW transition state, a NEGATIVE edge triggering method is use.
- It is mainly identified from the clock input lead along with a low-state indicator and a triangle.



Merits:

- By using level triggering, toggle state is obtain (unable to identify the state either 0 or 1), causes races.
- This can be overcome by using edge level triggering.

Demerits:

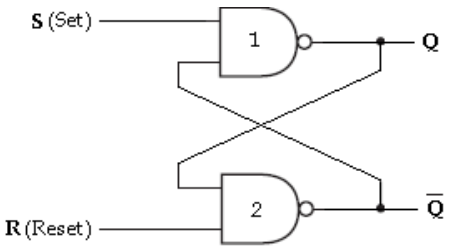
- If a clock pulse is given to the input of the flip flop at the same time when the output of the flip flop is changing, it may cause instability to the circuit.
- The instability is the feedback is given from the output combinational circuit to the memory elements.

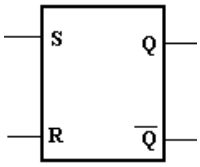
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

3. Explain SR latch using NAND gates.

- The SR latch can also be implemented using NAND gates.
- The input of this Latch is S and R. To understand how this circuit functions, recall that a low on any input to a NAND gate forces its output high.





SR latch using NAND gates **Logic Symbol**

We can summarize the operation of SR latch as follows:

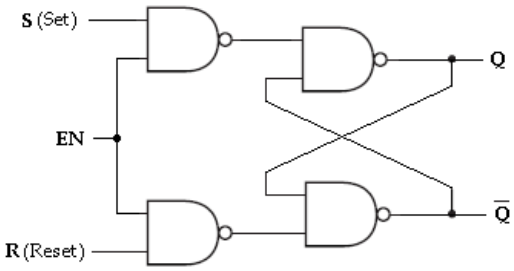
- When S= 0 and R= 0, the output of both gates will produce 0.
- When S= 0 and R= 1, the latch is reset to 0.
- When S= 1 and R= 0, the latch is set to 1.
- When S= 1 and R= 1, the output, Q_{n+1} remains in its present state, Q_n.

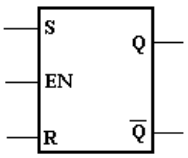
The truth table of NAND based SR latch is shown below

S	R	Q_n	Q_{n+1}	State
0	0	0	x	Indeterminate (*)
0	0	1	x	
0	1	0	1	Set
0	1	1	1	
1	0	0	0	Reset
1	0	1	0	
1	1	0	0	No Change (NC)
1	1	1	1	

Gated SR Latch:

- In the SR latch, the output changes occur immediately after the input changes i.e, the latch is sensitive to its S and R inputs all the time.
- A latch that is sensitive to the inputs only when an enable input is active.
- Such a latch with enable input is known as gated SR latch.
- The circuit behaves like SR latch when EN= 1.
- It retains its previous state when EN= 0





SR Latch with enable input using NAND gates Logic Symbol

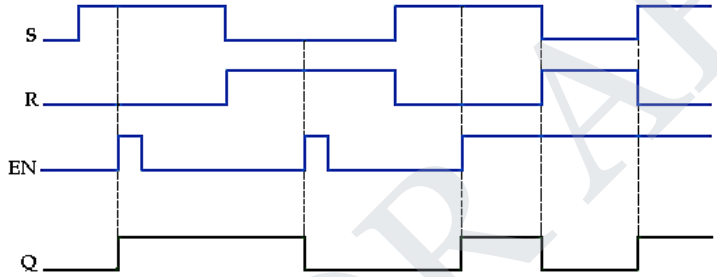
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

The truth table of gated SR latch is shown below.

EN	S	R	Q_n	Q_{n+1}	State
1 1	0 0	0 0	0 1	0 1	No Change (NC)
1 1	0 0	1 1	0 1	0 0	Reset
1 1	1 1	0 0	0 1	1 1	Set
1 1	1 1	1 1	0 1	x x	Indeterminate *
0 0	x x	x x	0 1	0 1	No Change (NC)

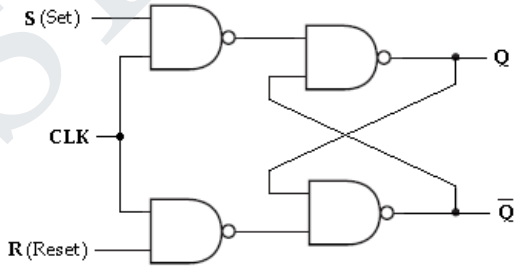
- When S is HIGH and R is LOW, a HIGH on the EN input sets the latch.
- When S is LOW and R is HIGH, a HIGH on the EN input resets the latch.

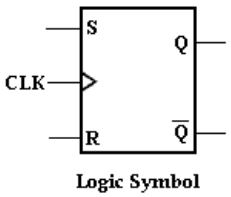


Input and output waveforms of Gated SR Latch

4. Draw the clocked S-R Flip-Flop and explain with truth table. [OR] **Explain the circuit of a SR flip-flop and explain its operation.** *[Nov'14]* [OR] **Explain the operation of SR flip flop.** *(Apr'18)*

- The S and R inputs of the S-R Flip-Flop are called synchronous inputs because data on these inputs are transferred to the Flip-Flop's output only on the triggering edge of the clock pulse.
- The circuit is similar to SR latch except enable signal is replaced by clock pulse (CLK).
- On the positive edge of the clock pulse, the circuit responds to the S and R inputs.





Logic Symbol

SR Flip-Flop

- When S is HIGH and R is LOW, the Q output goes HIGH on the triggering edge of the clock pulse, and the Flip-Flop is SET.
- When S is LOW and R is HIGH, the Q output goes LOW on the triggering edge of the clock pulse, and the Flip-Flop is RESET.

STUCOR APP
DOWNLOADED FROM STUCOR APP

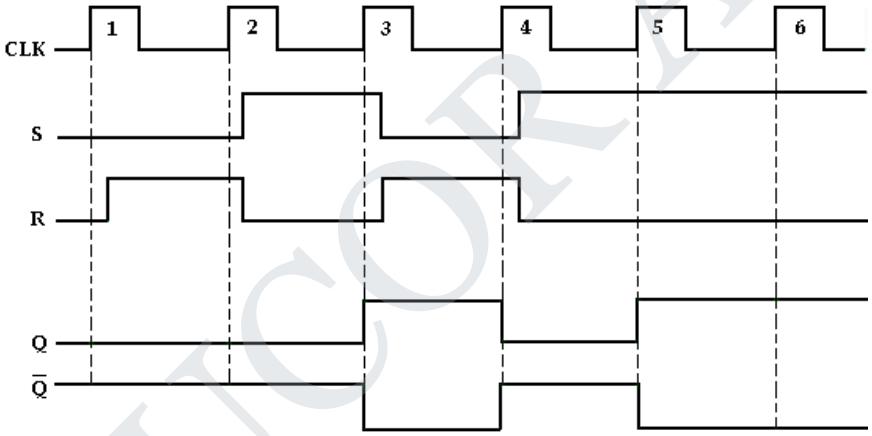
DOWNLOADED FROM STUCOR APP

- When both S and R are LOW, the output does not change from its prior state. An invalid condition exists when both S and R are HIGH.

CLK	S	R	Q_n	Q_{n+1}	State
1 1	0 0	0 0	0 1	0 1	No Change (NC)
1 1	0 0	1 1	0 1	0 0	Reset
1 1	1 1	0 0	0 1	1 1	Set
1 1	1 1	1 1	0 1	x x	Indeterminate *

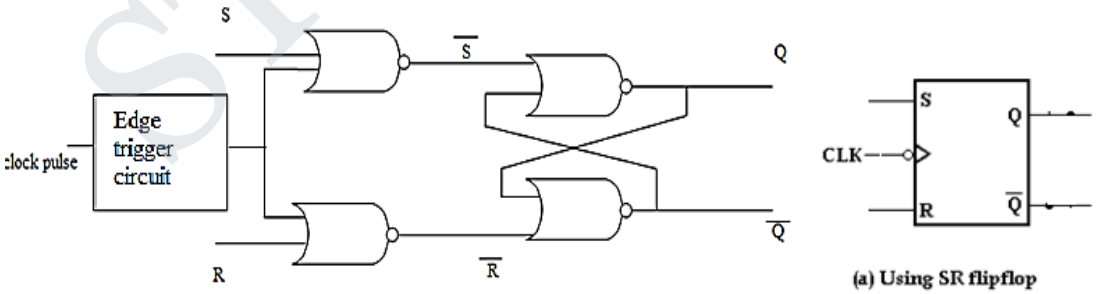
Truth table for SR Flip-Flop

The timing diagram of positive edge triggered SR flip-flop is shown below.



Input and output waveforms of SR Flip-Flop

5. Realize SR FF using NAND gates and explain its operations. *[May'12]*



(a) Using SR flipflop

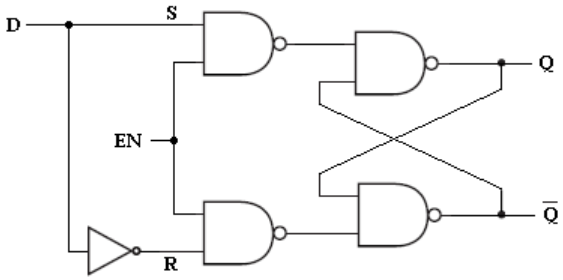
- In the negative edge triggered SR flip flop, the negative edge detector circuit is used and the circuit output responds at the negative edges of the clock pulse.
- Fig shows the logic symbol for negative edge triggered SR FF.
- The bubble at the clock input indicates that the flip-flop in negative edge triggered

STUCOR APP
DOWNLOADED FROM STUCOR APP

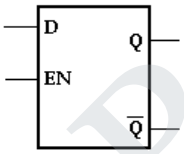
DOWNLOADED FROM STUCOR APP

6. Explain D Latch and draw the waveform.

- In SR latch, when both inputs are same (00 or 11), the output either does not change or it is invalid.
- In many practical applications, these input conditions are not required.
- These input conditions can be avoided by making them complement of each other.
- This modified SR latch is known as **D latch or DELAY latch.**



D Latch



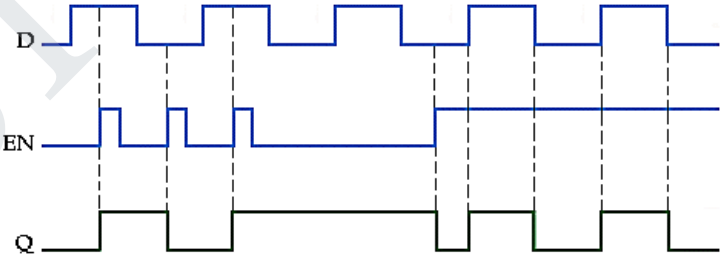
Logic Symbol

- As shown in the figure, D input goes directly to the S input, and its complement is applied to the R input.
- Therefore, only two input conditions exists, either S=0 and R=1 or S=1 and R=0.

The truth table for D latch is shown below.

EN	D	Q_n	Q_{n+1}	State
1	0	x	0	Reset
1	1	x	1	Set
0	x	x	Q_n	No Change (NC)

- As shown in the truth table, the Q output follows the D input. For this reason, D latch is called **transparent latch**.
- When D is HIGH and EN is HIGH. Q goes HIGH. When D is LOW and EN is HIGH, Q goes LOW.
- When EN is LOW, the state of the latch is not affected by the D input.

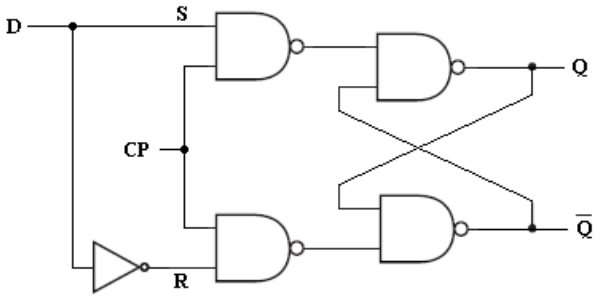


7. Draw the logic diagram of a D Flip-Flop using NAND gates and explain. (OR) **Sketch the state diagram and state table for 'D' flip-flops.** *[Nov'12]*

- Like in D latch, in D Flip-Flop the basic SR Flip-Flop is used with complemented inputs.
- The D Flip-Flop is similar to D-latch except clock pulse is used instead of enable input.

STUCOR APP
DOWNLOADED FROM STUCOR APP

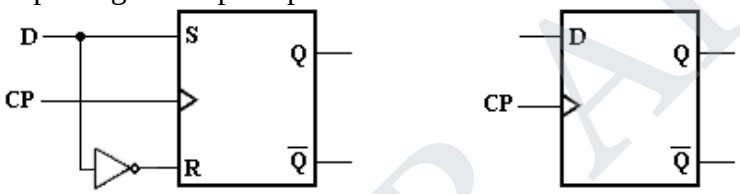
DOWNLOADED FROM STUCOR APP



D Flip-Flop

- To eliminate the undesirable condition of the indeterminate state in the RS Flip-Flop is to ensure that inputs S and R are never equal to 1 at the same time.
- This is done by D Flip-Flop. The D (***delay***) Flip-Flop has one input called delay input and clock pulse input.

The D Flip-Flop using SR Flip-Flop is shown below.

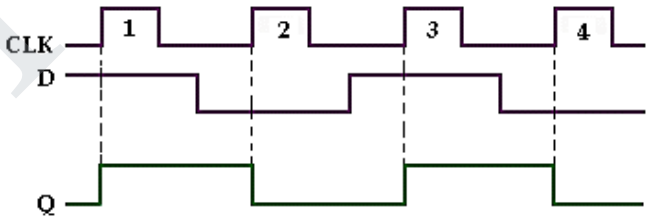


(a) Using SR flipflop (b) Graphic symbol

Truth Table: The truth table of D Flip-Flop is given below.

Clock	D	Q_{n+1}	State
1	0	0	Reset
1	1	1	Set
0	x	Q_n	No Change

The timing diagram of positive edge triggered D flip-flop is shown below.



Input and output waveforms of clocked D Flip-Flop

- Looking at the truth table for D Flip-Flop we can realize that Q_{n+1} function follows the D input at the positive going edges of the clock pulses.

8. Provide the Characteristic table and Characteristic equation of D flip-flop.

- The characteristic table for D Flip-Flop shows that the next state of the Flip-Flop is independent of the present state since Q_{n+1} is equal to D.

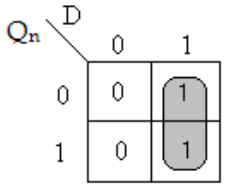
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- This means that an input pulse will transfer the value of input D into the output of the Flip-Flop independent of the value of the output before the pulse was applied.

The characteristic equation is derived from K-map.

Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

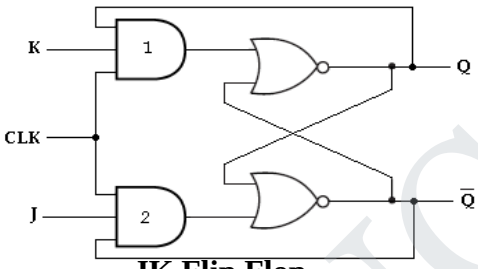


Q_{n+1}= D

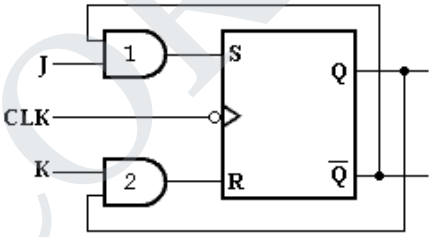
Characteristic table k-map Simplification Characteristic equation:

9. Construct a clocked J-K Flip-Flop which is triggered at positive edge. (OR) **Sketch the state diagram and state table for JK' flip-flops.** *[Nov'12]*. (OR) **Explain the operation of JK flip flop** *(Apr'18)*

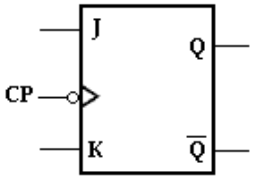
- JK Flip-Flop has two inputs J(set) and K (reset).
- A JK Flip-Flop can be obtained from the clocked SR Flip-Flop by augmenting two AND gates as shown below.



JK Flip Flop



(a) Using SR flipflop



(b) Graphic symbol

- The data input J and the output Q' are applied o the first AND gate and its output (JQ') is applied to the S input of SR Flip-Flop.
- Similarly, the data input K and the output Q are applied to the second AND gate and its output (KQ) is applied to the R input of SR Flip-Flop.

J= K= 0

When J=K= 0, both AND gates are disabled. Therefore clock pulse have no effect, hence the Flip-Flop output is same as the previous output.

J= 0, K= 1

When J= 0 and K= 1, AND gate 1 is disabled i.e., S= 0 and R= 1. This condition will reset the Flip-Flop to 0.

J= 1, K= 0

When J= 1 and K= 0, AND gate 2 is disabled i.e., S= 1 and R= 0. Therefore the Flip-Flop will set on the application of a clock pulse.

J= K= 0

When J=K= 1, it is possible to set or reset the Flip-Flop.

- If Q is High, AND gate 2 passes on a reset pulse to the next clock.

STUCOR APP
DOWNLOADED FROM STUCOR APP

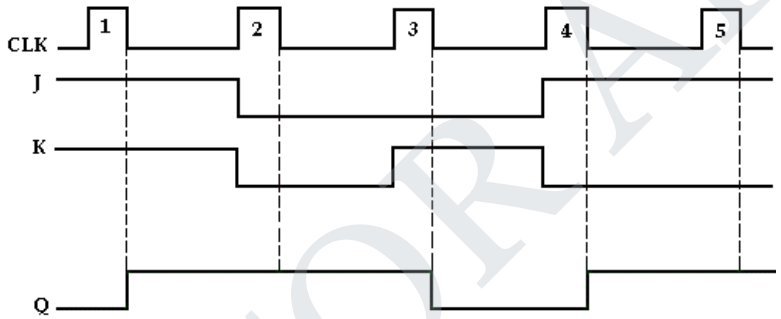
DOWNLOADED FROM STUCOR APP

- When Q is low, AND gate 1 passes on a set pulse to the next clock.
- Either way, Q changes to the complement of the last state i.e., toggle. Toggle means to switch to the opposite state.

Truth table: The truth table of JK Flip-Flop is given below.

CLK	Inputs		Output	State
	J	K	Q_{n+1}	
1	0	0	Q_n	No Change
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	Q_n'	Toggle

The timing diagram of negative edge triggered JK flip-flop is shown below.



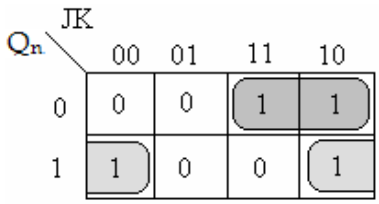
Input and output waveforms of JK Flip-Flop

10. Write down Characteristic table and Characteristic equation for JK flip-flop.

- The characteristic table for JK Flip-Flop is shown in the table below.
- From the table, K-map for the next state transition (Q_{n+1}) can be drawn and the simplified logic expression which represents the characteristic equation of JK Flip-Flop can be found.

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

Characteristic table



K-map Simplification

Q_{n+1}= JQ'+ K'Q

Characteristic equation

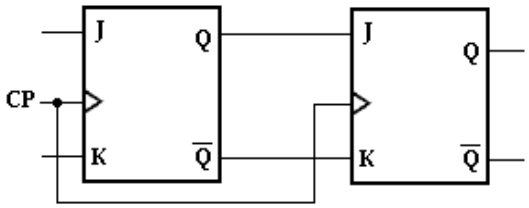
11. Explain the operation of a Master-Slave JK Flip-flop. *[Apr'15, May'16, May'17, Nov'17]*

- A master-slave Flip-Flop is constructed using two separate JK Flip-Flops.
- The first Flip-Flop is called the master.

STUCOR APP
DOWNLOADED FROM STUCOR APP

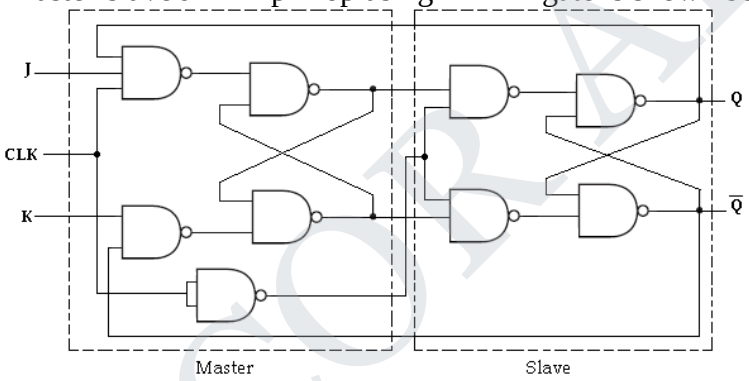
DOWNLOADED FROM STUCOR APP

- It is driven by the positive edge of the clock pulse.
- The second Flip-Flop is called the slave.
- It is driven by the negative edge of the clock pulse.
- The logic diagram of a master-slave JK Flip-Flop is shown below.



Logic diagram

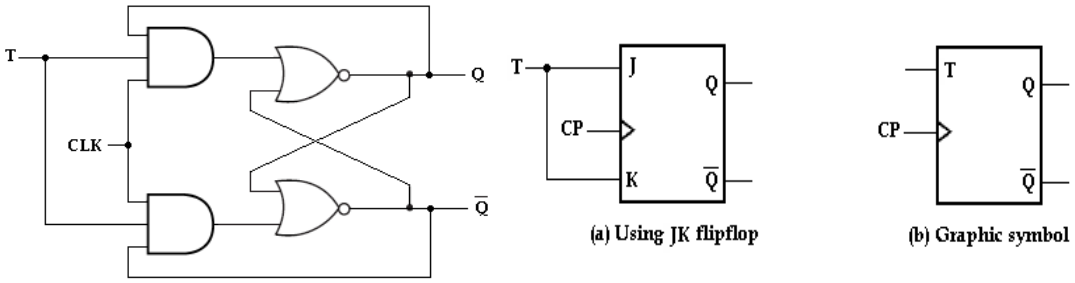
- When the clock pulse has a positive edge, the master acts according to its J-K inputs, but the slave does not respond, since it requires a negative edge at the clock input.
- When the clock input has a negative edge, the slave Flip-Flop copies the master outputs.
- But the master does not respond since it requires a positive edge at its clock input.
- The clocked master-slave J-K Flip-Flop using NAND gate is shown below.



Master-Slave JK Flip-Flop

12. Provide the Characteristic table and Characteristic equation of T Flip-Flop. (OR) **Explain the working principle of T FF.** (OR) **Explain the operation of T flip flop** *(Apr'18)* (OR) **Explain the operation, state diagram and characteristics of T flip flop.** *(Nov'17)*

- The T (***Toggle***) Flip-Flop is a modification of the JK Flip-Flop.
- It is obtained from JK Flip-Flop by connecting both inputs J and K together, i.e., single input.
- Regardless of the present state, the Flip-Flop complements its output when the clock pulse occurs while input T= 1.



(a) Using JK flipflop (b) Graphic symbol

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

T Flip-Flop

- When T= 0, $Q_{n+1}= Q_n$, ie., the next state is the same as the present state and no change occurs.
- When T= 1, $Q_{n+1}= Q_n'$,ie., the next state is the complement of the present state.

Truth Table:

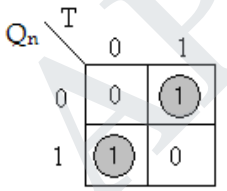
T	Q_{n+1}	State
0	Q_n	No Change
1	Q_n'	Toggle

Characteristic table and Characteristic equation:

- The characteristic table for T Flip-Flop is shown below and characteristic equation is derived using K-map.

Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

K-map Simplification



$Q_{n+1}= TQ_n'+ T'Q_n$

Characteristic equation

13. Write characteristic table and excitation table of D Flip-Flop and T flip-flop.

D Flip-Flop:

Present State	Input	Next State
Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

Characteristic Table

Present State	Next State	Input
Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

Excitation table

T flip-flop:

Present State	Input	Next State
Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

Characteristic Table

Present State	Next State	Input
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Excitation table

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

14. Write the excitation table for RS, JK, T and D flip flops. (OR) **Explain the flip flop excitation tables for JK and RS flipflop.** *(Apr'18)*

Present State	Next State	Inputs	
Q_n	Q_{n+1}	S	R
0	0	0	x
0	1	1	0
1	0	0	1
1	1	X	0

Excitation table for SR Flip-Flop

Present State	Next State	Inputs	
Q_n	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Excitation table for JK Flip-Flop

Present State	Next State	Input
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Excitation table for T Flip-Flop

Present State	Next State	Input
Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

Excitation table for D Flip-Flop

15. Write the procedure to convert SR Flip-Flop to D Flip-Flop.

- Write the characteristic table for required Flip-Flop (D Flip-Flop).
- Write the excitation table for given Flip-Flop (SR Flip-Flop).
- Determine the expression for the given Flip-Flop inputs (S and R) by using K- Map.
- Draw the Flip-Flop conversion logic diagram to obtain the required flip-flop (D Flip-Flop) by using the above obtained expression.

The excitation table for the above conversion is

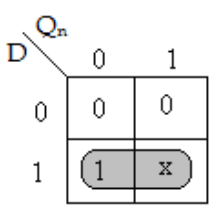
Required Flip-Flop (D)			Given Flip-Flop (SR)	
Input	Present state	Next state	Flip-Flop Inputs	
D	Q_n	Q_{n+1}	S	R
0	0	0	0	x
0	1	0	0	1
1	0	1	1	0
1	1	1	x	0

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

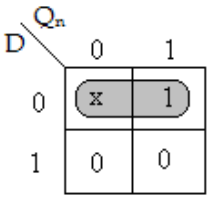
K-map simplification

For S



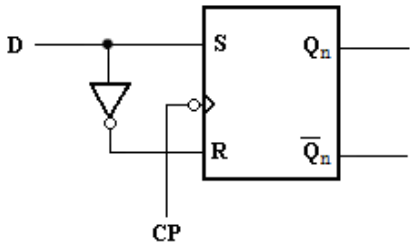
S = D

For R



R = $\overline{D}$

Logic diagram



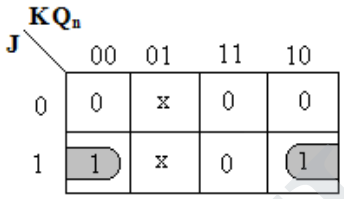
16. How will you convert SR Flip-Flop to JK Flip-Flop? *[May '12]*

The excitation table for the above conversion is,

Inputs		Present state	Next state	Flip-Flop Inputs	
J	**K**	**Q_n**	**Q_{n+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

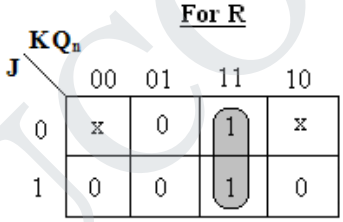
K-map simplification

For S



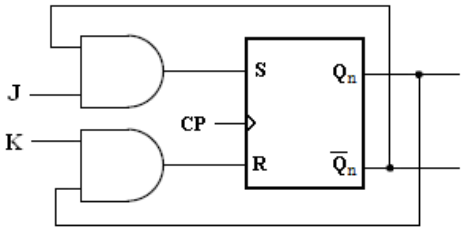
S = $J\overline{Q}_n$

For R



R = KQ_n

Logic diagram



17. How will you convert SR Flip-Flop to T Flip-Flop?

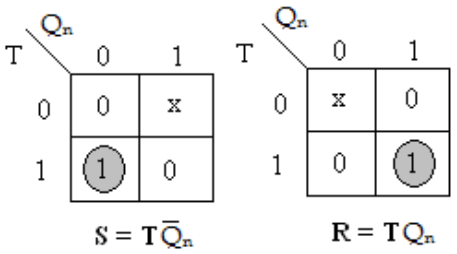
The excitation table for the above conversion is

Input	Present state	Next state	Flip-Flop Inputs	
T	**Q_n**	**Q_{n+1}**	**S**	**R**
0	0	0	0	x
0	1	1	x	0
1	0	1	1	0
1	1	0	0	1

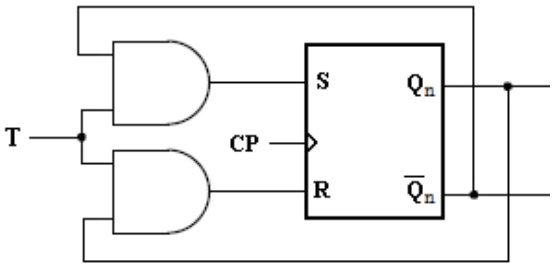
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

K-map simplification



Logic diagram

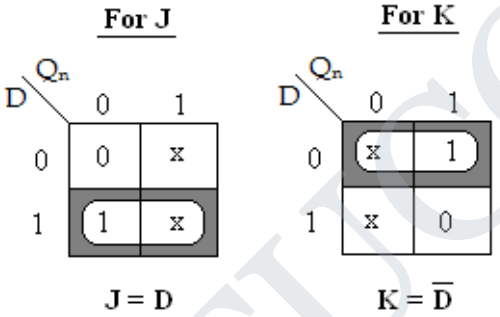


18. How will you convert JK Flip-Flop to D Flip-Flop?

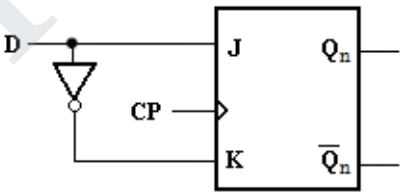
The excitation table for the above conversion is

Input	Present state	Next state	Flip-Flop Inputs	
D	**Q_n**	**Q_{n+1}**	**J**	**K**
0	0	0	0	x
0	1	0	x	1
1	0	1	1	x
1	1	1	x	0

K-map simplification



Logic diagram



19. How will you convert JK Flip-Flop to T Flip-Flop? [OR] Realize T flip flop using JK flip flop. *[Nov'15]* [OR] **Explain the realization of JK flipflop from T flip flop.** *[Nov'16]*

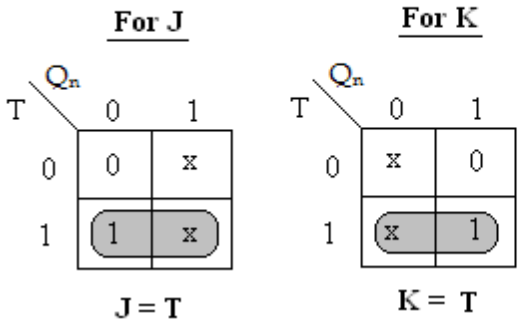
The excitation table for the above conversion is

Input	Present state	Next state	Flip-Flop Inputs	
T	**Q_n**	**Q_{n+1}**	**J**	**K**
0	0	0	0	x
0	1	1	x	0
1	0	1	1	x
1	1	0	x	1

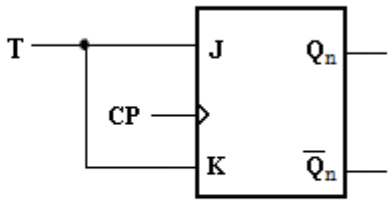
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

K-map simplification



Logic diagram

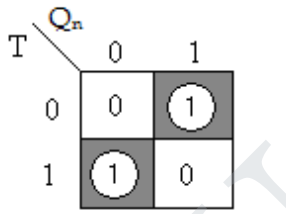


20. How will you convert D Flip-Flop to T Flip-Flop?

The excitation table for the above conversion is

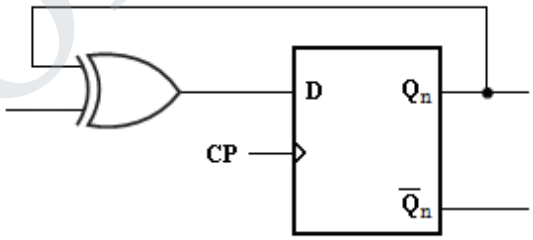
Input	Present state	Next state	Flip-Flop Input
T	**Q_n**	**Q_{n+1}**	**D**
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

K-map simplification



$$D = \overline{T}Q_n + T\overline{Q}_n$$
$$= T \oplus Q_n$$

Logic diagram



- The type of Flip-Flop to be used may be included in the design specifications or may depend what is available to the designer.
- Many digital systems are constructed with JK Flip-Flops because they are the most versatile available. The selection of inputs is given as follows.

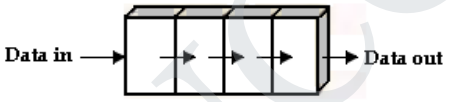
Flip-Flop	Application
JK	General Applications
D	Applications requiring transfer of data (Ex: Shift Registers)
T	Application involving complementation (Ex: Binary Counters)

STUCOR APP
DOWNLOADED FROM STUCOR APP

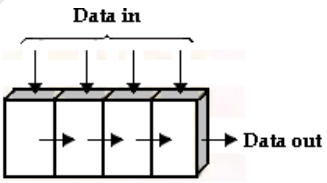
DOWNLOADED FROM STUCOR APP

21. Write short notes about Shift Registers. Or Explain in detail about different shift registers. *(Nov'17)*

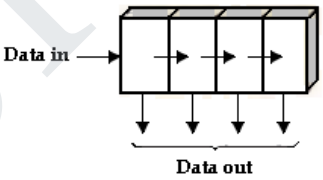
- A register is simply a group of Flip-Flops that can be used to store a binary number. There must be one Flip-Flop for each bit in the binary number.
- For instance, a register used to store an 8-bit binary number must have 8 Flip-Flops.
- The Flip-Flops must be connected such that the binary number can be entered (shifted) into the register and possibly shifted out.
- A group of Flip-Flops connected to provide either or both of these functions is called a **shift register**.
- The bits in a binary number (data) can be removed from one place to another in either of two ways.
- The first method involves shifting the data one bit at a time in a serial fashion, beginning with either the most significant bit (MSB) or the least significant bit (LSB).
- This technique is referred to as serial shifting.
- The second method involves shifting all the data bits simultaneously and is referred to as parallel shifting.
- There are two ways to shift into a register (serial or parallel) and similarly two ways to shift the data out of the register.
- This leads to the construction of four basic register types
  i. Serial in- serial out
  ii. Serial in- parallel out
  iii. Parallel in- serial out
  iv. Parallel in- parallel out



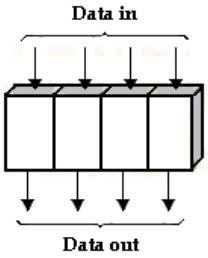
(i) Serial in- serial out



(iii) Parallel in- serial out



(ii) Serial in- parallel out



(iv) Parallel in- parallel out

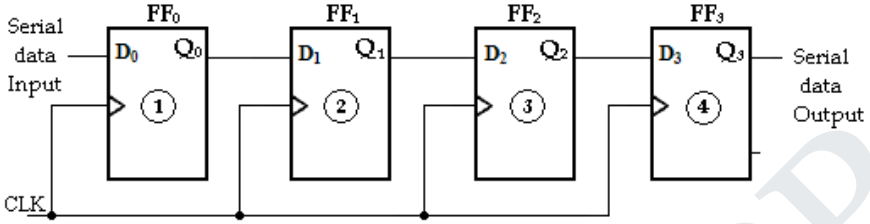
22. Draw a 4 bit Serial-In Serial-Out Shift Register.

- The serial in/serial out shift register accepts data serially, i.e., one bit at a time on a single line.
- It produces the stored information on its output also in serial form.
- The entry of the four bits 1010 into the register is illustrated below, beginning with the right-most bit.
- The register is initially clear.

STUCOR APP
DOWNLOADED FROM STUCOR APP

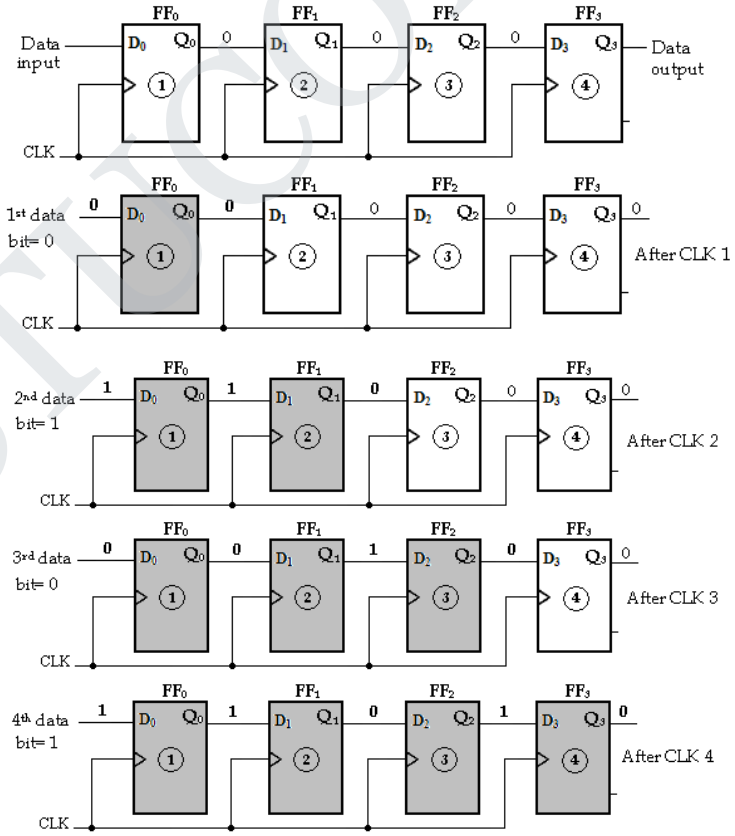
DOWNLOADED FROM STUCOR APP

- The 0 is put onto the data input line, making D=0 for FF_0.
- When the first clock pulse is applied, FF_0 is reset, thus storing the 0.
- Next the second bit, which is a 1, is applied to the data input, making D=1 for FF_0 and D=0 for FF_1 because the D input of FF_1 is connected to the Q_0 output.
- When the second clock pulse occurs, the 1 on the data input is shifted into FF0, causing FF0 to set; and the 0 that was in FF0 is shifted into FFl.



Serial-In Serial-Out Shift Register

- The third bit, a 0, is now put onto the data-input line, and a clock pulse is applied.
- The 0 is entered into FF_0, the 1 stored in FF_0 is shifted into FF_l, and the 0 stored in FF_1 is shifted into FF_2.
- The last bit, a 1, is now applied to the data input, and a clock pulse is applied.
- This time the 1 is entered into FF0, the 0 stored in FF0 is shifted into FFl, the 1 stored in FF1 is shifted into FF2, and the 0 stored in FF2 is shifted into FF3.
- This completes the serial entry of the four bits into the shift register, where they can be stored for any length of time as long as the Flip-Flops have dc power.

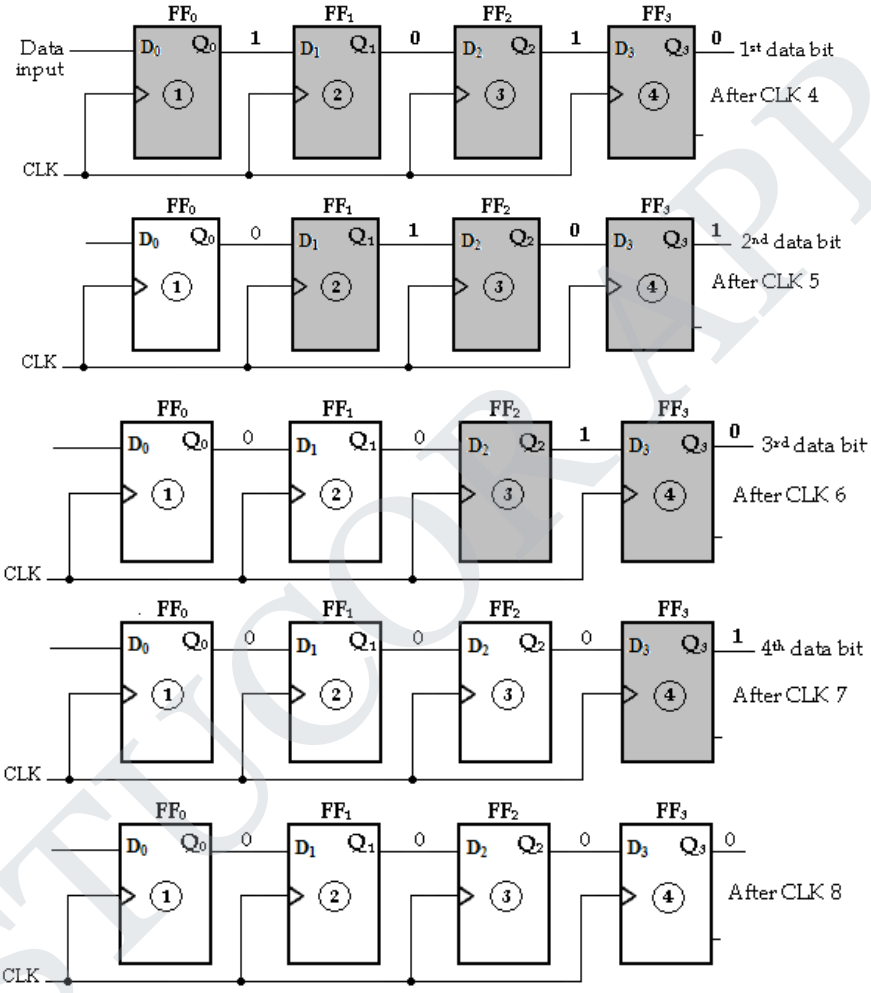


Four bits (1010) being entered serially into the register

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- To get the data out of the register, the bits must be shifted out serially and taken off the Q3 output.
- After CLK4, the right-most bit, 0, appears on the Q3 output.
- When clock pulse CLK5 is applied, the second bit appears on the Q3 output.
- Clock pulse CLK6 shifts the third bit to the output, and CLK7 shifts the fourth bit to the output.
- While the original four bits are being shifted out, more bits can be shifted in. All zeros are shown being shifted out, more bits can be shifted in.



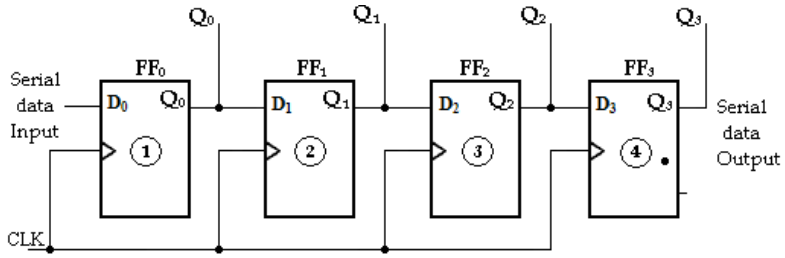
- Four bits (1010) being entered serially-shifted out of the register and replaced by all zeros

23. Draw the Serial-In Parallel-Out Shift Register and explain its operation. [OR] **Write short note on SIPO and draw the output waveforms.** *[Nov'16]*

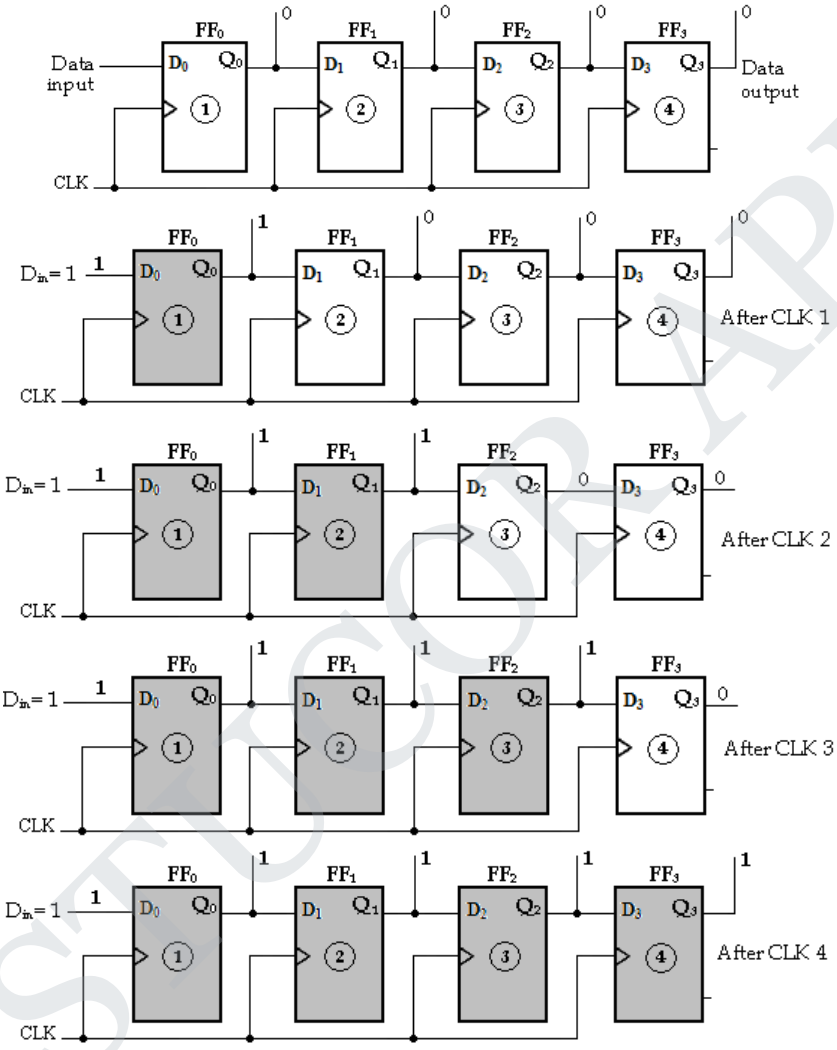
- In this shift register, data bits are entered into the register in the same as serial-in serial-out shift register.
- But the output is taken in parallel.
- Once the data are stored, each bit appears on its respective output line and all bits are available simultaneously instead of on a bit-by-bit.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



Serial-In parallel-Out Shift Register



Four bits (1111) being serially entered into the register

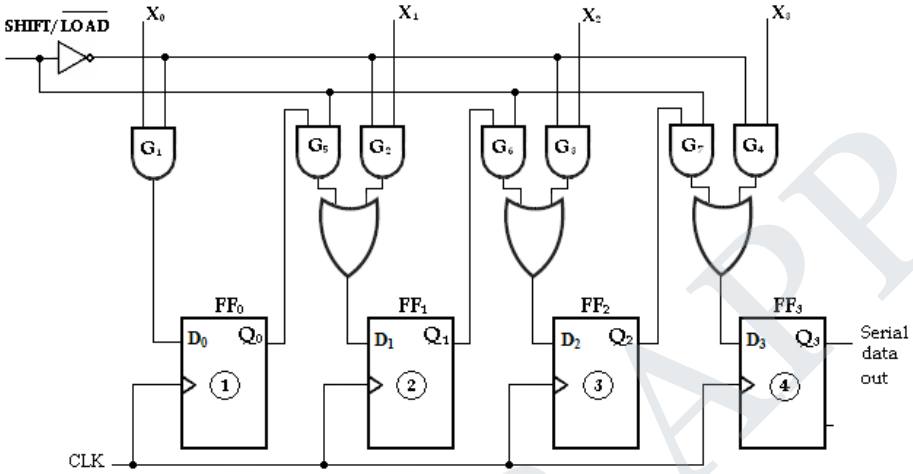
24. Draw a 4 bit Parallel-In Serial-Out Shift Register and briefly explain. [OR] **Design a 4 bit parallel –in serial out shift register using D Flip Flops.** *[May'17]*

- In this type, the bits are entered in parallel i.e., simultaneously into their respective stages on parallel lines.
- A 4-bit parallel-in serial-out shift register is illustrated below. There are four data input lines, X_0, X_1, X_2 and X_3 for entering data in parallel into the register.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- SHIFT/ LOAD input is the control input, which allows four bits of data to **load** in parallel into the register.
- When SHIFT/LOAD is LOW, gates G_1, G_2, G_3 and G_4 are enabled, allowing each data bit to be applied to the D input of its respective Flip-Flop.
- When a clock pulse is applied, the Flip-Flops with D = 1 will **set** and those with D = 0 will **reset,** thereby storing all four bits simultaneously.

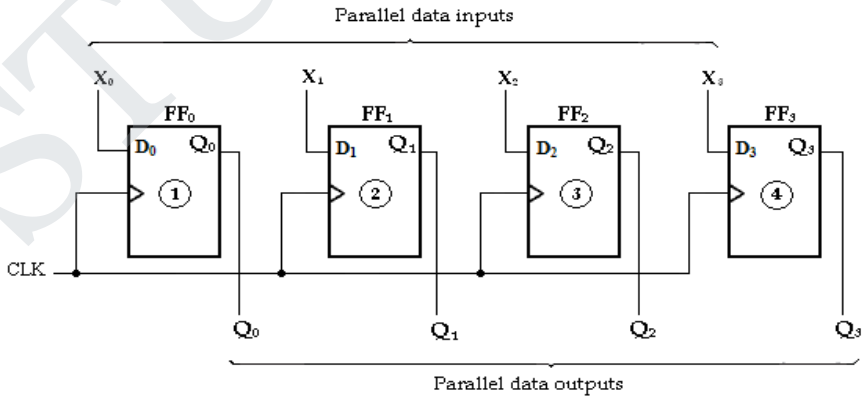


Parallel-In Serial-Out Shift Register

- When SHIFT/LOAD is HIGH, gates G_1, G_2, G_3 and G_4 are disabled and gates G_5, G_6 and G_7 are enabled, allowing the data bits to shift right from one stage to the next.
- The OR gates allow either the normal shifting operation or the parallel data-entry operation, depending on which AND gates are enabled by the level on the SHIFT/LOAD input.

25. Design a Parallel-In Parallel-Out Shift Register.

- In this type, there is simultaneous entry of all data bits and the bits appear on parallel outputs simultaneously.



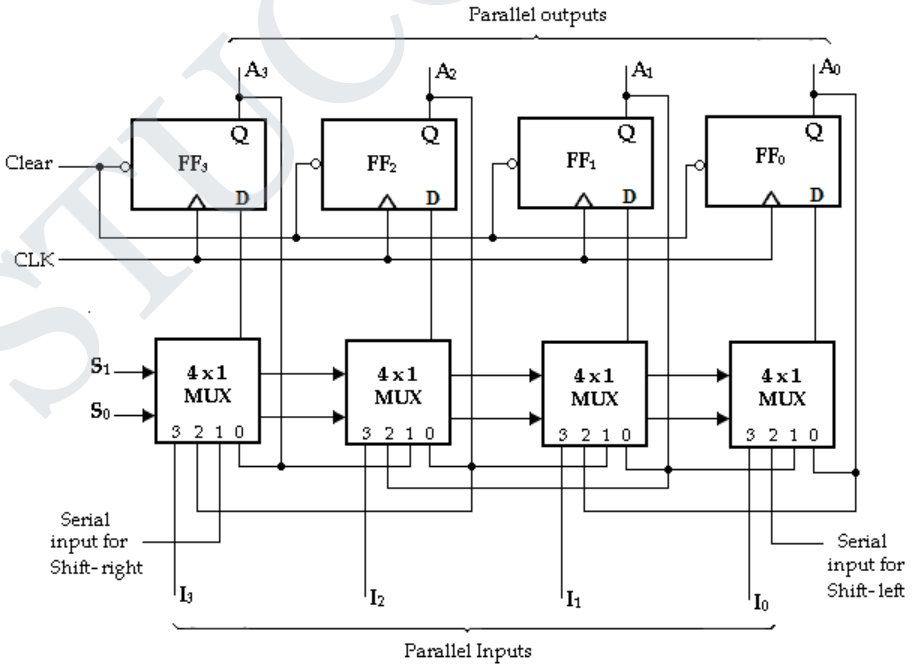
Parallel-In Parallel-Out Shift Register

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

26. Design 4 bit Universal Shift Registers and explain its operation.

- If the register has shift and parallel load capabilities, then it is called a shift register with parallel load or universal shift register.
- Shift register can be used for converting serial data to parallel data, and vice-versa.
- If a parallel load capability is added to a shift register, the data entered in parallel can be taken out in serial fashion by shifting the data stored in the register.
- The functions of universal shift register are:
  1. A clear control to clear the register to 0.
  2. A clock input to synchronize the operations.
  3. A shift-right control to enable the shift right operation and the serial input and output lines associated with the shift right.
  4. A shift-left control to enable the shift left operation and the serial input and output lines associated with the shift left.
  5. A parallel-load control to enable a parallel transfer and the n input lines associated with the parallel transfer.
  6. 'n' parallel output lines.
  7. A control line that leaves the information in the register unchanged even though the clock pulses re continuously applied.
- It consists of four D-Flip-Flops and four 4 input multiplexers (MUX). S_0 and S_1 are the two selection inputs connected to all the four multiplexers.
- These two selection inputs are used to select one of the four inputs of each multiplexer.
- The input 0 in each MUX is selected when S_1S_0= 00 and input 1 is selected when S_1S_0= 01.
- Similarly inputs 2 and 3 are selected when S_1S_0= 10 and S_1S_0= 11 respectively.
- The inputs S1 and S0 control the mode of the operation of the register.



4-Bit Universal Shift Register

STUCOR APP
DOWNLOADED FROM STUCOR APP

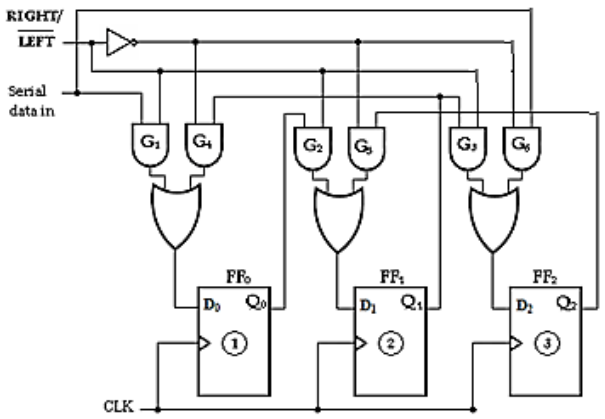
DOWNLOADED FROM STUCOR APP

- When S_1S_0= 00, the present value of the register is applied to the D-inputs of the Flip-Flops.
- This is done by connecting the output of each Flip-Flop to the 0 input of the respective multiplexer.
- The next clock pulse transfers into each Flip-Flop, the binary value is held previously, and hence no change of state occurs.
- When S_1S_0= 01, terminal 1 of the multiplexer inputs has a path to the D inputs of the Flip-Flops.
- This causes a shift-right operation with the left serial input transferred into Flip-Flop FF_3.
- When S_1S_0= 10, a shift-left operation results with the right serial input going into Flip-Flop FF_1.
- Finally when S_1S_0= 11, the binary information on the parallel input lines (I_1, I_2, I_3 and I_4) are transferred into the register simultaneously during the next clock pulse.
- The function table of bi-directional shift register with parallel inputs and parallel outputs is shown below.

Mode Control		Operation
S_1	S_0	
0	0	No change
0	1	Shift-right
1	0	Shift-left
1	1	Parallel load

27. Design a 3-bit bi-direction shift register. *[Apr'15]*

- A 3-bit bidirectional shift register is shown below.
- A HIGH on the RIGHT/LEFT control input allows data bits inside the register to be shifted to the right, and a LOW enables data bits inside the register to be shifted to the left.
- When the RIGHT/LEFT control input is **HIGH**, gates G_1, $G_2$and G_3 are enabled and the state of the Q output of each Flip-Flop is passed through to the D input of the following Flip-Flop.
- When a clock pulse occurs, the data bits are shifted one place to the right.
- When the RIGHT/LEFT control input is **LOW**, gates G_4, G_5, and G_6 are enabled, and the Q output of each Flip-Flop is passed through to the D input of the preceding Flip-Flop.
- When a clock pulse occurs, the data bits are then shifted one place to the left.

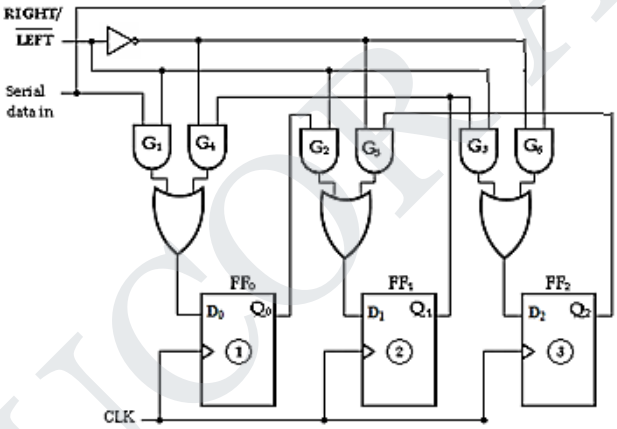


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

28. Design a 4-bit Bi-direction Shift Registers

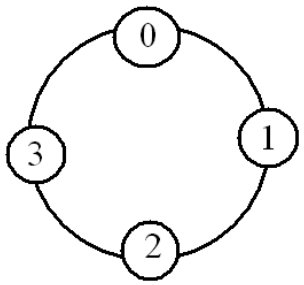
- A bidirectional shift register is one in which the data can be shifted either left or right.
- It can be implemented by using gating logic that enables the transfer of a data bit from one stage to the next stage to the right or to the left depending on the level of a control line.
- A 4-bit bidirectional shift register is shown below.
- A HIGH on the RIGHT/LEFT control input allows data bits inside the register to be shifted to the right, and a LOW enables data bits inside the register to be shifted to the left.
- When the RIGHT/LEFT control input is **HIGH**, gates G_1, G_2, G_3 and G_4 are enabled, and the state of the Q output of each Flip-Flop is passed through to the D input of the following Flip-Flop.
- When a clock pulse occurs, the data bits are shifted one place to the right.
- When the RIGHT/LEFT control input is **LOW**, gates G_5, G_6, G_7 and G_8 are enabled, and the Q output of each Flip-Flop is passed through to the D input of the preceding Flip-Flop.
- When a clock pulse occurs, the data bits are then shifted one place to the left.



4-bit bi-directional shift register

29. Design 2-Bit Synchronous Binary Counter.

Step 1: State Diagram



Step 2: Number of Flip Flops = 2; Type: T Flip Flop

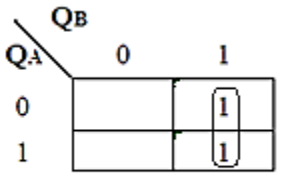
Step 3: State Transition Table

STUCOR APP
DOWNLOADED FROM STUCOR APP

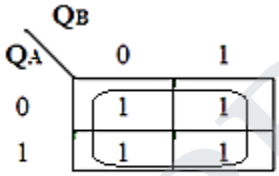
DOWNLOADED FROM STUCOR APP

Present State		Next State		Flip Flop Inputs	
QA	QB	QA+1	QB+1	TA	TB
0	0	0	1	0	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	0	0	1	1

Step 4: K-map

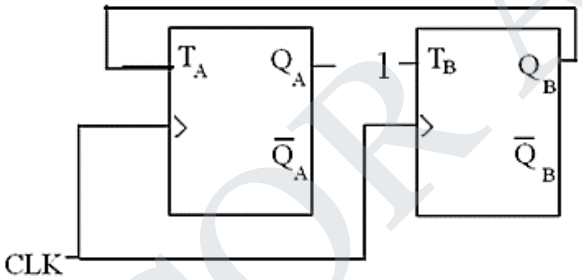


$T_A = Q_B$



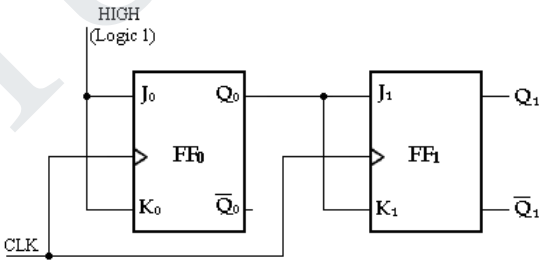
$T_B = 1$

Step 5: Logic Circuit



Explanation:

- In this counter the clock signal is connected in parallel to clock inputs of both the Flip-Flops (FF_0 and FF_1).
- The output of FF_0 is connected to J_1 and K_1 inputs of the second Flip-Flop (FF_1).

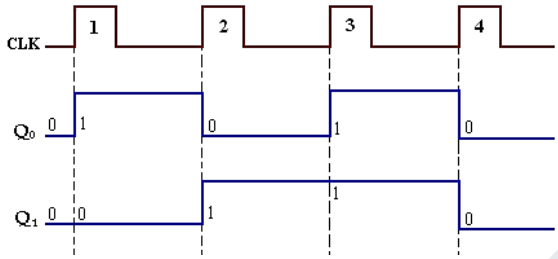


- Assume that the counter is initially in the binary 0 state: i.e., both Flip-Flops are RESET.
- When the positive edge of the first clock pulse is applied, FF_0 will toggle because $J_0 = k_0 = 1$, whereas FF_1 output will remain 0 because $J_1 = k_1 = 0$. After the first clock pulse $Q_0 = 1$ and $Q_1 = 0$.
- When the leading edge of CLK2 occurs, FF_0 will toggle and Q_0 will go LOW. Since FF_1 has a HIGH ($Q_0 = 1$) on its J_1 and K_1 inputs at the triggering edge of this clock pulse, the Flip-Flop toggles and Q_1 goes HIGH. Thus, after CLK2, $Q_0 = 0$ and $Q_1 = 1$.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

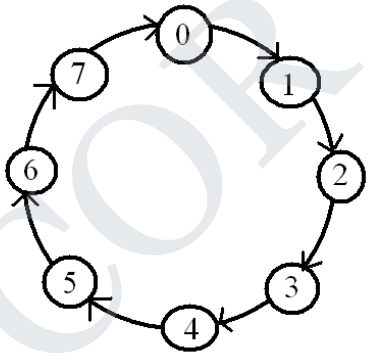
- When the leading edge of CLK3 occurs, FF_0 again toggles to the SET state ($Q_0 = 1$), and FF_1 remains SET ($Q_1 = 1$) because its J_1 and K_1 inputs are both LOW ($Q_0 = 0$). After this triggering edge, $Q_0 = 1$ and $Q_1 = 1$.
- Finally, at the leading edge of CLK4, Q_0 and Q_1 go LOW because they both have a toggle condition on their J_1 and K_1 inputs. The counter has now recycled to its original state, $Q_0 = Q_1 = 0$.



Timing diagram

30. Design a 3-Bit Synchronous Binary Counter using JK flip-flop. (OR) **Design a Sequential logic circuit for a 3 bit binary counter.** *[Nov'12]*

Step 1: State Diagram



Step 2: Number of Flip Flops = 3; Type: T Flip Flop

Step 3: State Transition Table

Present State			Next State			Flip Flop Inputs		
QA	QB	QC	QA+1	QB+1	QC+1	TA	TB	TC
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Step 4: K-map

For T_A

Q_A \ Q_BQ_C	00	01	11	10
0			1	
1			1	

$T_A = Q_BQ_C$

For T_B

Q_A \ Q_BQ_C	00	01	11	10
0		1	1	
1		1	1	

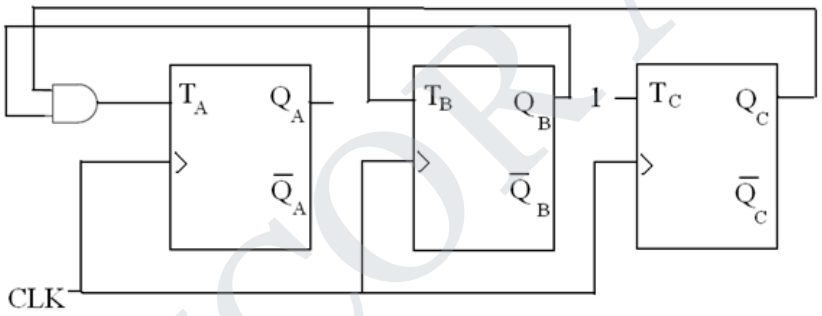
$T_B = Q_C$

For T_C

Q_A \ Q_BQ_C	00	01	11	10
0	1	1	1	1
1	1	1	1	1

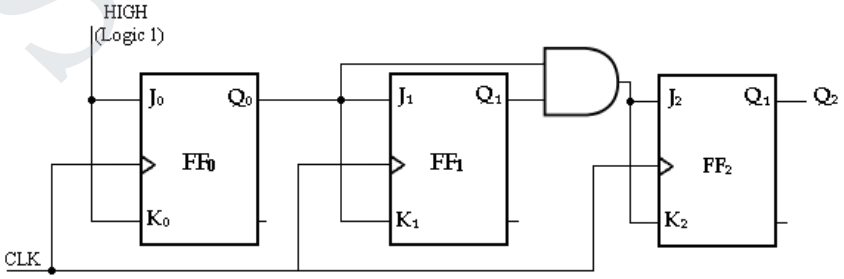
$T_C = 1$

Step 5: Logic Circuit



Explanation:

- A 3 bit synchronous binary counter is constructed with three JK Flip-Flops and an AND gate.
- The output of FF_0 (Q_0) changes on each clock pulse as the counter progresses from its original state to its final state and then back to its original state.
- To produce this operation, FF_0 must be held in the toggle mode by constant HIGH, on its J_0 and K_0 inputs.



3-Bit Synchronous Binary Counter

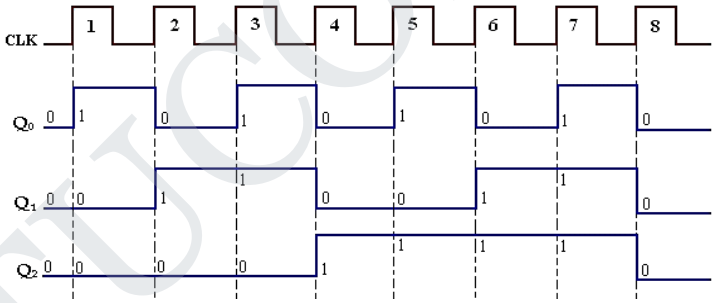
- The output of FF_1 (Q_1) goes to the opposite state following each time $Q_0 = 1$.
- This change occurs at CLK2, CLK4, CLK6, and CLK8.
- The CLK8 pulse causes the counter to recycle.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

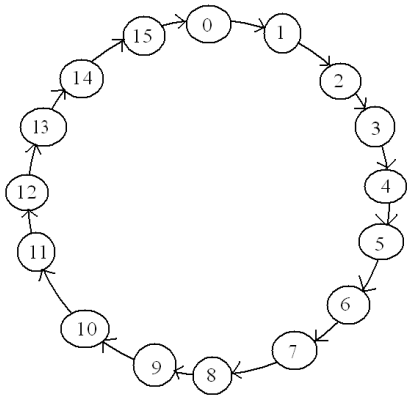
- To produce this operation, Q_0 is connected to the J_1 and K_1 inputs of FF_1.
- When Q_0= 1 and a clock pulse occurs, FF_1 is in the toggle mode and therefore changes state.
- When Q_0= 0, FF_1 is in the no-change mode and remains in its present state.
- The output of FF_2 (Q_2) changes state both times; it is preceded by the unique condition in which both Q_0 and Q_1 are HIGH.
- This condition is detected by the AND gate and applied to the J_2 and K_2 inputs of FF_2.
- Whenever both outputs Q_0= Q_1= 1, the output of the AND gate makes the J_2= K_2= 1 and FF_2 toggles on the following clock pulse.
- Otherwise, the J_2 and K_2 inputs of FF2 are held LOW by the AND gate output, FF_2 does not change state.

CLOCK Pulse	Q_2	Q_1	Q_0
Initially	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
8 (recycles)	0	0	0



31. Design a 4-Bit Synchronous Binary Counter.

Step 1: State Diagram



Step 2: Number of Flip Flops = 4; Type: JK Flip Flop

Step 3: State Transition Table

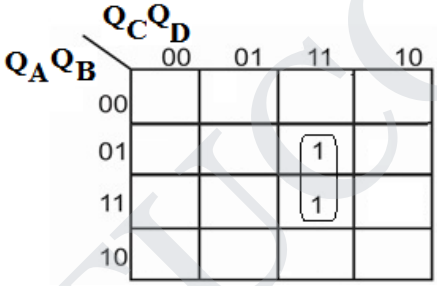
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Present State				Next State				Flip Flop Inputs			
Q_A	Q_B	Q_C	Q_D	Q_{A+1}	Q_{B+1}	Q_{C+1}	Q_{D+1}	T_A	T_B	T_C	T_D
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	1
0	0	1	1	0	1	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	1
0	1	0	1	0	1	1	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	1
0	1	1	1	1	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	1
1	0	0	1	1	0	1	0	0	0	1	1
1	0	1	0	1	0	1	1	0	0	0	1
1	0	1	1	1	1	0	0	0	1	1	1
1	1	0	0	1	1	0	1	0	0	0	1
1	1	0	1	1	1	1	0	0	0	1	1
1	1	1	0	1	1	1	1	0	0	0	1
1	1	1	1	0	0	0	0	1	1	1	1

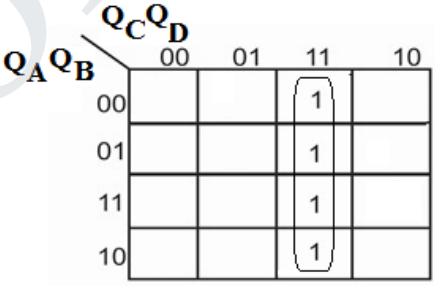
Step 4: K-map

For T_A



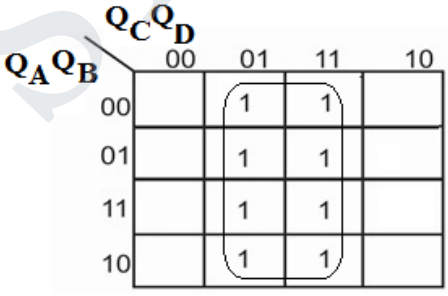
$T_A = Q_B Q_C Q_D$

For T_B



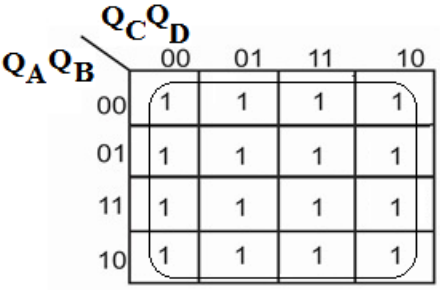
$T_B = Q_C Q_D$

For T_C



$T_C = Q_D$

For T_D

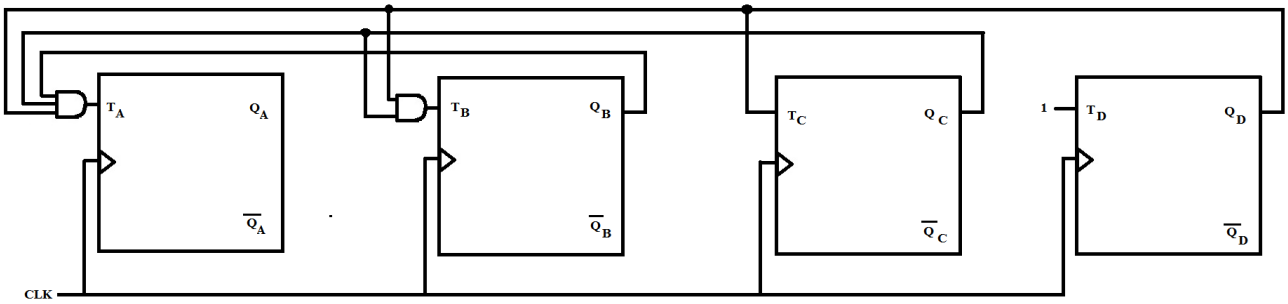


$T_D = 1$

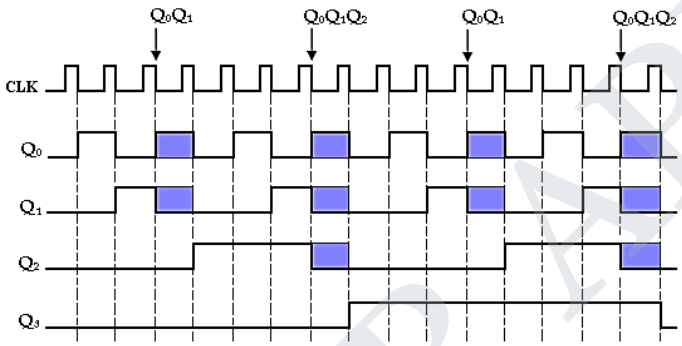
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Logic Diagram:

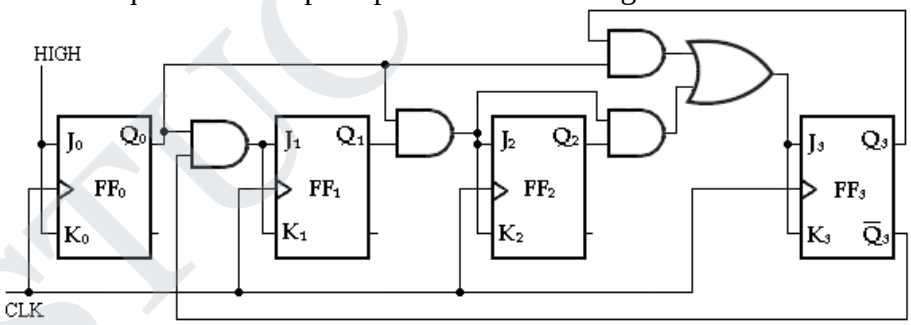


4-Bit Synchronous Binary Counter



32. Design a 4-Bit Synchronous Decade Counter. *[Nov'12]* **(BCD Counter).**

- BCD decade counter has a sequence from 0000 to 1001 (9).
- After 1001 state it must recycle back to 0000 state.
- This counter requires four Flip-Flops and AND/OR logic as shown below.



4-Bit Synchronous Decade Counter

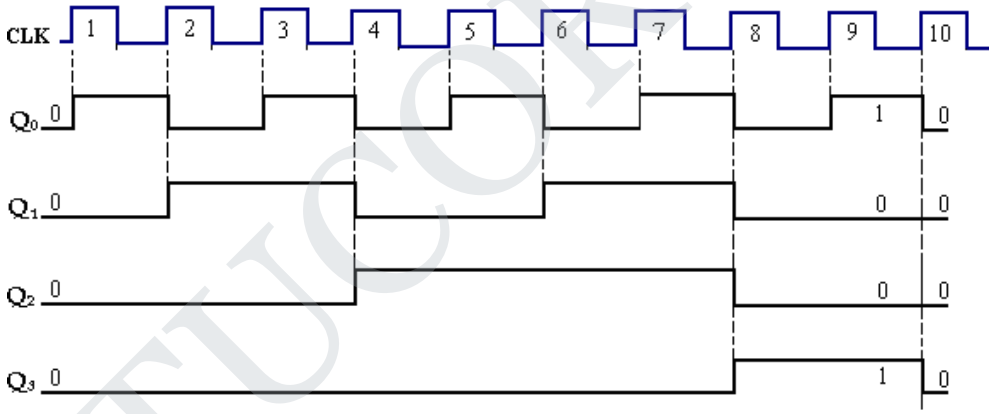
CLOCK Pulse	Q_3	Q_2	Q_1	Q_0
Initially	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

8	1	0	0	0
9	1	0	0	1
10(recycles)	0	0	0	0

- First, notice that FF_0 (Q_0) toggles on each clock pulse, so the logic equation for its J_0 and K_0 inputs is **$J_0 = K_0 = 1$.**
- This equation is implemented by connecting J_0 and K_0 to a constant HIGH level.
- Next, notice from table, that FF_1 (Q_1) changes on the next clock pulse each time Q0 = 1 and Q3 = 0, so the logic equation for the J_1 and K_1 inputs is **$J_1 = K_1 = Q_0Q_3'$.**
- This equation is implemented by ANDing Q_0 and Q_3 and connecting the gate output to the J_1 and K_1 inputs of FF_1.
- Flip-Flop 2 (Q_2) changes on the next clock pulse each time both $Q_0 = Q_1 = 1$.
- This requires an input logic equation as follows: **$J_2 = K_2 = Q_0Q_1$.**
- This equation is implemented by ANDing Q_0 and Q_1 and connecting the gate output to the J_2 and K_2 inputs of FF_2.
- Finally, FF_3 (Q_3) changes to the opposite state on the next clock pulse each time $Q_0 = 1$, $Q_1 = 1$, and $Q_2 = 1$ (state 7), or when $Q_0 = 1$ and $Q_1 = 1$ (state 9).
- The equation for this is as follows: **$J_3 = K_3 = Q_0Q_1Q_2 + Q_0Q_3$**. This function is implemented with the AND/OR logic connected to the J_3 and K_3 inputs of FF_3.



Timing diagram

33. Design a Synchronous UP/DOWN Counter. Explain the working principle.

- An up/down counter is a bidirectional counter, capable of progressing in either direction through a certain sequence.
- A 3-bit binary counter that advances upward through its sequence (0, 1, 2, 3, 4, 5, 6, 7) and then can be reversed so that it goes through the sequence in the opposite direction (7, 6, 5, 4, 3, 2, 1,0) is an illustration of up/down sequential operation.
- The complete up/down sequence for a 3-bit binary counter is shown in table below. The arrows indicate the state-to-state movement of the counter for both its UP and its DOWN modes of operation.
- An examination of Q_0 for both the up and down sequences shows that FF_0 toggles on each clock pulse.
- Thus, the J_0 and K_0 inputs of FF_0 are, **$J_0 = K_0 = 1$**

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

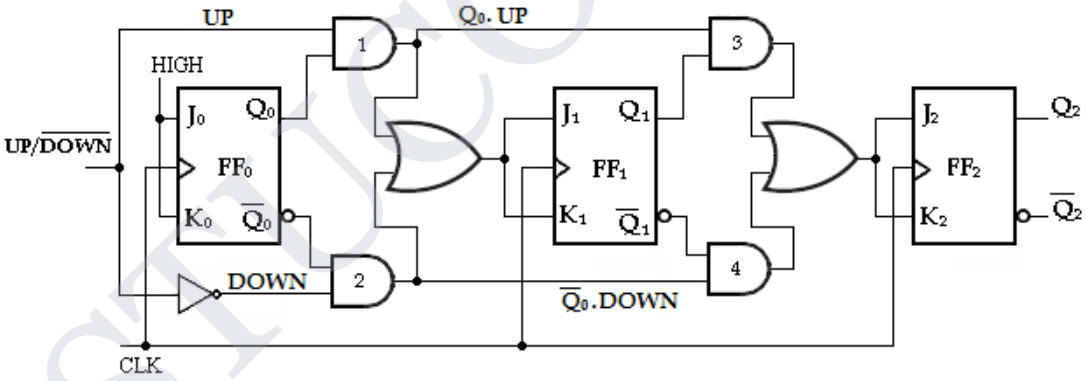
- To form a synchronous UP/DOWN counter, the control input (UP/DOWN) is used to allow either the normal output or the inverted output of one Flip-Flop to the J and K inputs of the next Flip-Flop.
- When UP/DOWN= 1, the MOD 8 counter will count from 000 to 111 and UP/DOWN= 0, it will count from 111 to 000.
- When UP/DOWN= 1, it will enable AND gates 1 and 3 and disable AND gates 2 and 4.
- This allows the Q0 and Q1 outputs through the AND gates to the J and K inputs of the following Flip-Flops, so the counter counts up as pulses are applied.

CLOCK PULSE	UP	Q_2	Q_1	Q_0	DOWN
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	

When UP/DOWN= 0, the reverse action takes place.

$J_1 = K_1 = (Q_0.UP) + (Q_0'.DOWN)$

$J_2 = K_2 = (Q_0.\ Q_1.UP) + (Q_0'.Q_1'.DOWN)$

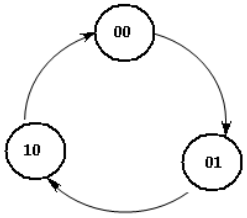


3-bit UP/DOWN Synchronous Counter

34. Design and explain the working of a synchronous MOD-3 counter.

$2^n \geq N = 3$ $\Rightarrow 2^2 > 3$. Therefore, 2 Flip-Flops are required.

State Diagram:



State Table:

Present State	Next State
00	01
01	10
10	00

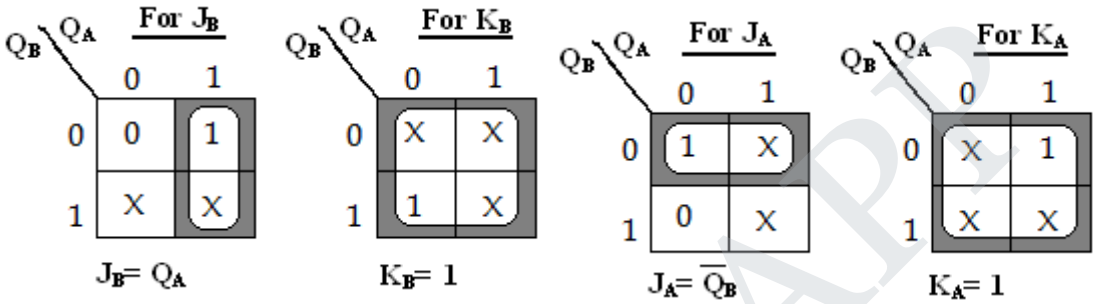
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

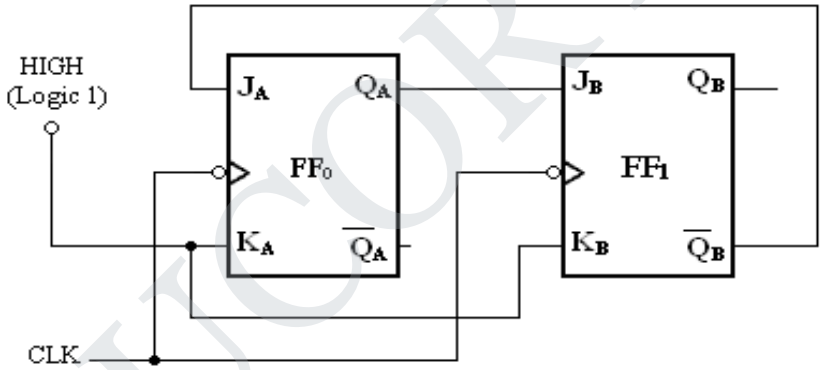
Excitation Table for Counter:

Present State		Next State		Flip-Flop Inputs			
Q_B	Q_A	Q_{B+1}	Q_{A+1}	J_B	K_B	J_A	K_A
0	0	0	1	0	X	1	x
0	1	1	0	1	X	x	1
1	0	0	0	x	1	0	x

K-map Simplification:



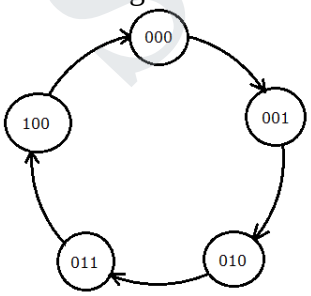
Logic Diagram:



35. Design a MOD-5 synchronous counter using JK Flip-Flops. *[Apr'15]* [OR] **Design a MOD5 counter using T flip flop.** *[May '16]* [OR] **Design a 5 bit ring counter and mention its application.** *[May '17]*

$2^n \geq N = 5$ $\Rightarrow 2^3 > 8$. Therefore, 3 Flip-Flops are required.

State Diagram:



State Table:

Present State	Next State
000	001
001	010
010	011
011	100
100	000

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Excitation Table for JK Flip-Flop:

Present State	Next State	Inputs	
Q_n	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Excitation Table for Counter:

Present State			Next State			Flip-Flop Inputs					
Q_C	Q_B	Q_A	Q_{C+1}	Q_{B+1}	Q_{A+1}	J_C	K_C	J_B	K_B	J_A	K_A
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	0	0	0	x	1	0	x	0	x
1	0	1	x	x	x	x	x	x	x	x	x
1	1	0	x	x	x	x	x	x	x	x	x
1	1	1	x	x	x	x	x	x	x	x	x

K-Map Simplification:

For J_C

Q_C \ $Q_B Q_A$	00	01	11	10
0	0	0	1	0
1	X	X	X	X

$J_C = Q_B Q_A$

For K_C

Q_C \ $Q_B Q_A$	00	01	11	10
0	X	X	X	X
1	1	X	X	X

$K_C = 1$

For J_B

Q_C \ $Q_B Q_A$	00	01	11	10
0	0	1	X	X
1	0	X	X	X

$J_B = Q_A$

For K_B

Q_C \ $Q_B Q_A$	00	01	11	10
0	X	X	1	0
1	X	X	X	X

$K_B = Q_A$

For J_A

Q_C \ $Q_B Q_A$	00	01	11	10
0	1	X	X	1
1	0	X	X	X

$J_A = Q_C$

For K_A

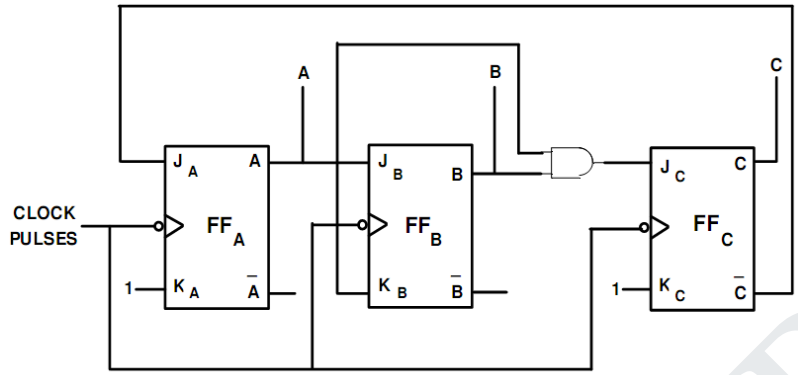
Q_C \ $Q_B Q_A$	00	01	11	10
0	X	1	1	X
1	X	X	X	X

$K_A = 1$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

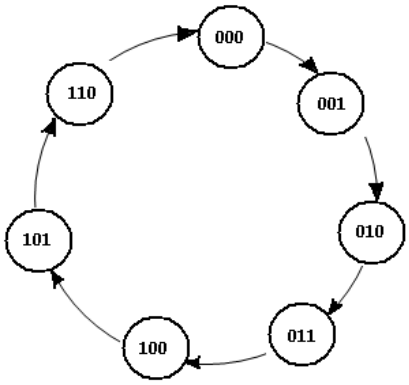
Logic Diagram:



36. Design MOD 7 counter using D flip flop. *[May'14]*

$2^n \geq N = 7 \Rightarrow 2^3 > 8$. Therefore, 3 Flip-Flops are required.

State Diagram:



State Table:

Present State	Next State
000	001
001	010
010	011
011	100
100	101
101	110
110	000

Excitation Table for JK Flip-Flop:

Present State	Next State	Inputs
Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

Excitation Table for Counter:

Present State			Next State			Flip-Flop Inputs		
Q_2	Q_1	Q_0	q_2	q_1	q_0	D_2	D_1	D_0
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	1	1	0	1	1	0
1	1	0	0	0	0	0	0	0

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

For D_2

Q_2 \ Q_1Q_0	00	01	11	10
0	0	0	(1)	0
1	[1]	[1]		0

$D_2 = Q_2\overline{Q_1} + \overline{Q_2}Q_1Q_0$

For D_1

Q_2 \ Q_1Q_0	00	01	11	10
0	0	[1]	0	(1)
1	0	[1]		0

$D_1 = \overline{Q_1}Q_0 + \overline{Q_2}Q_1\overline{Q_0}$

For D_1

Q_2 \ Q_1Q_0	00	01	11	10
0	0	[1]	0	(1)
1	0	[1]		0

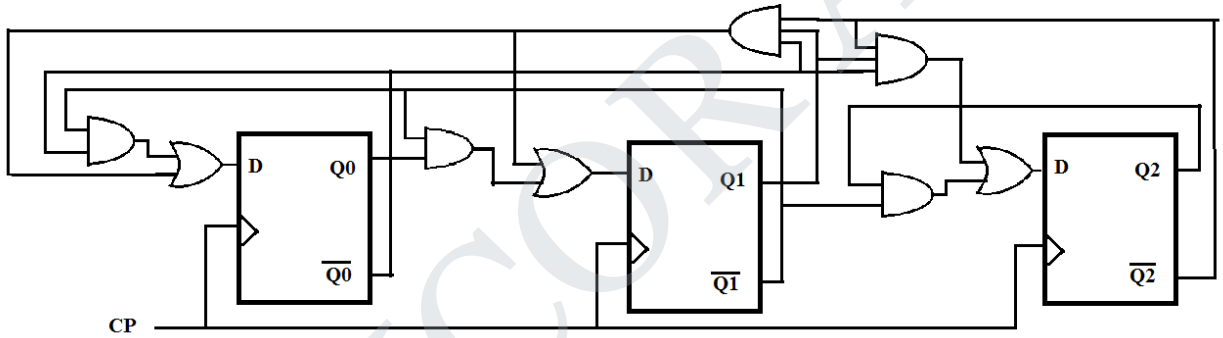
$D_1 = \overline{Q_1}Q_0 + \overline{Q_2}Q_1\overline{Q_0}$

For D_0

Q_2 \ Q_1Q_0	00	01	10	11
0	[1]	0	0	(1)
1	[1]	0		0

$D_0 = \overline{Q_1}\,\overline{Q_0} + \overline{Q_2}Q_1\overline{Q_0}$

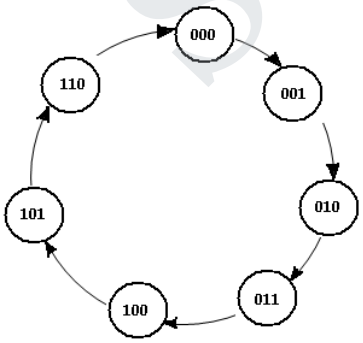
Then the circuit diagram will be,



37. Design a MOD-7 synchronous counter using JK Flip-Flops. Write excitation table and state table. *[May'14]*

$2^n \geq N = 7$ =>2^3> 8.Therefore, 3 Flip-Flops are required.

State Diagram:



State Table:

Present State	Next State
000	001
001	010
010	011
011	100
100	101
101	110
110	000

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

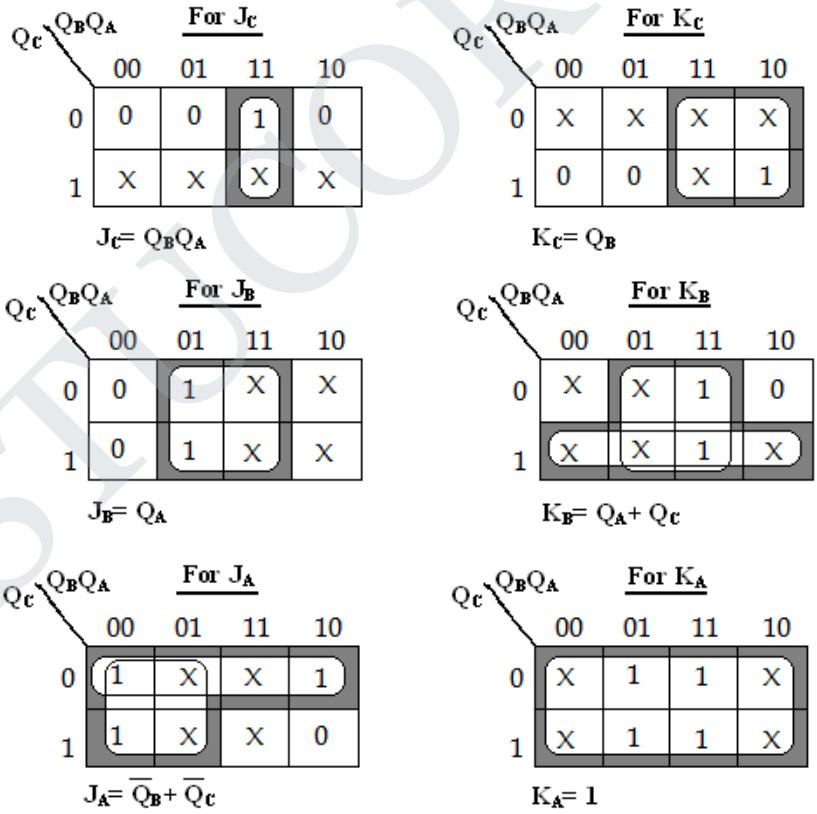
Excitation Table for JK Flip-Flop:

Present State	Next State	Inputs	
Q_n	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Excitation Table for Counter:

Present State			Next State			Flip-Flop Inputs					
Q_C	Q_B	Q_A	Q_{C+1}	Q_{B+1}	Q_{A+1}	J_C	K_C	J_B	K_B	J_A	K_A
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	0	0	0	x	1	x	1	0	X

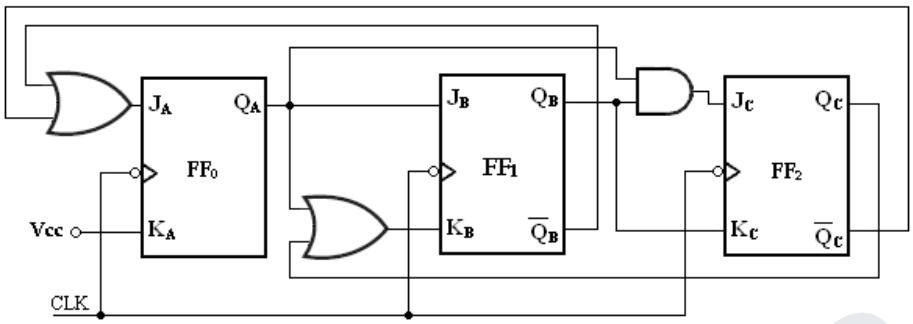
K-map Simplification:



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Logic Diagram:



38. Design a Modulus-10 –Counters (or) Decade counter.

- The counter with 'n' Flip-Flops has maximum MOD number 2^n.
- Find the number of Flip-Flops (n) required for the desired MOD number (N) using the equation, $\mathbf{2^n \geq N.}$
- For example, a 3 bit binary counter is a MOD 8 counter.
- The basic counter can be modified to produce MOD numbers less than 2^n by allowing the counter to skin those are normally part of counting sequence.

$n= 3, N= 8$

$2^n = 2^3 = 8 = N$

(i) MOD 5 Counter:

$2^n = N$

$2^n = 5$

$2^2 = 4$ less than N.

$2^3 = 8 > N$ (5)

Therefore, 3 Flip-Flops are required.

(ii) MOD 10 Counter:

$2^n = N = 10$

$2^3 = 8$ less than N.

$2^4 = 16 > N$ (10).

Therefore, 4 Flip-Flops are required.

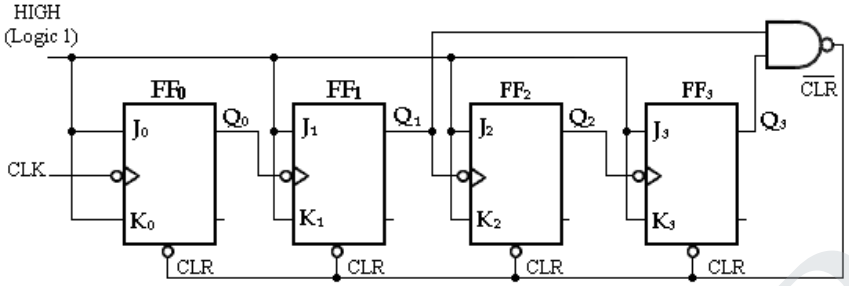
- To construct any MOD-N counter, the following methods can be used.
  1. Find the number of Flip-Flops (n) required for the desired MOD number (N) using the equation, $\mathbf{2^n \geq N.}$
  2. Connect all the Flip-Flops as a required counter.
  3. Find the binary number for N.
  4. Connect all Flip-Flop outputs for which Q= 1 when the count is N, as inputs to NAND gate.
  5. Connect the NAND gate output to the CLR input of each Flip-Flop.

- When the counter reaches N^{th} state, the output of the NAND gate goes LOW, resetting all Flip-Flops to 0.
- Therefore the counter counts from 0 through N-1.
- For example, MOD-10 counter reaches state 10 (1010). i.e., $Q_3Q_2Q_1Q_0 = 1\ 0\ 1\ 0$.
- The outputs Q_3 and Q_1 are connected to the NAND gate and the output of the NAND gate goes LOW and resetting all Flip-Flops to zero.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

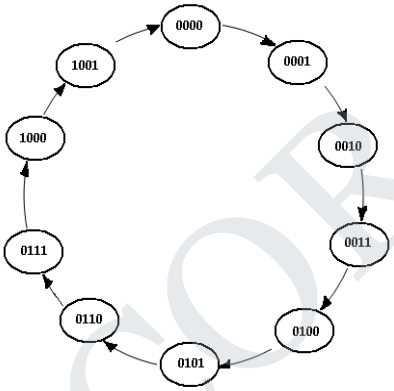
- Therefore MOD-10 counter counts from 0000 to 1001. And then recycles to the zero value.
- The MOD-10 counter circuit is shown below.



MOD-10 (Decade) Counter

39. Design a MOD-10 synchronous counter using JK Flip-Flops. Write excitation table and state table.

$2^n \geq N = 10 \Rightarrow 2^4 > 10$. Therefore, 4 Flip-Flops are required.



State Diagram

State Table:

Present State	Next State
0000	0001
0001	0010
0010	0011
0011	0100
0100	0101
0101	0110
0110	0111
0111	1000
1000	1001
1001	0000

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

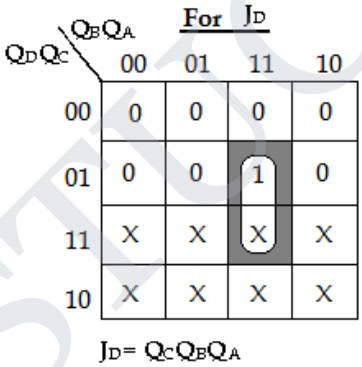
Excitation Table for JK Flip-Flop:

Present State	Next State	Inputs	
Q_n	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

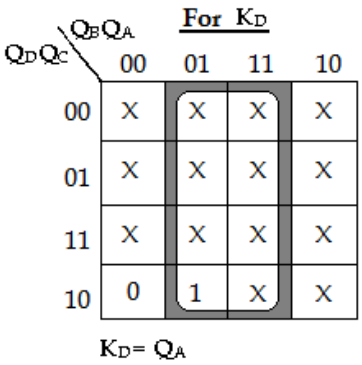
Excitation Table for Counter:

Present State				Next State				Flip-Flop Inputs							
Q_D	Q_C	Q_B	Q_A	Q_{D+1}	Q_{C+1}	Q_{B+1}	Q_{A+1}	J_D	K_D	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	0	0	1	0	x	0	x	0	x	1	x
0	0	0	1	0	0	1	0	0	x	0	x	1	x	x	1
0	0	1	0	0	0	1	1	0	x	0	x	x	0	1	x
0	0	1	1	0	1	0	0	0	x	1	x	x	1	x	1
0	1	0	0	0	1	0	1	0	x	x	0	0	x	1	x
0	1	0	1	0	1	1	0	0	x	x	0	1	x	x	1
0	1	1	0	0	1	1	1	0	x	x	0	x	0	1	x
0	1	1	1	1	0	0	0	1	x	x	1	x	1	x	1
1	0	0	0	1	0	0	1	x	0	0	x	0	x	1	x
1	0	0	1	0	0	0	0	x	1	0	x	0	x	x	1

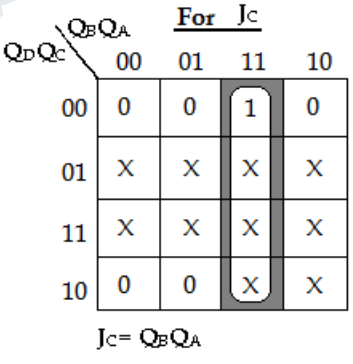
K-map Simplification:



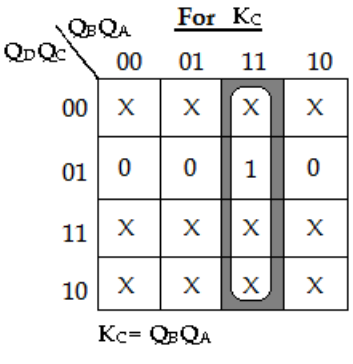
$J_D = Q_CQ_BQ_A$



$K_D = Q_A$



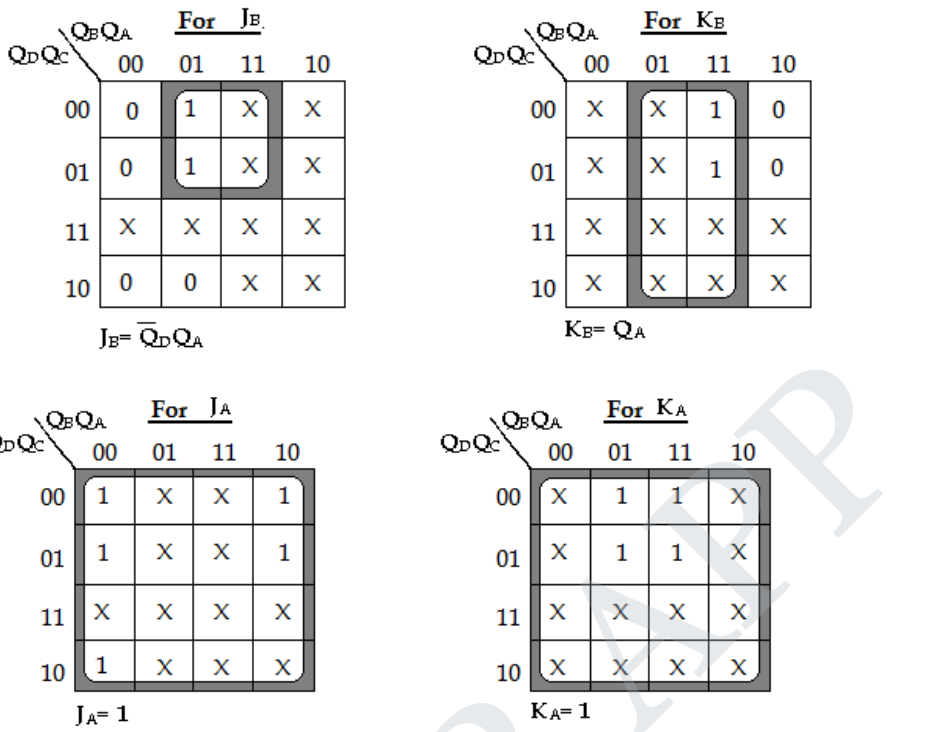
$J_C = Q_BQ_A$



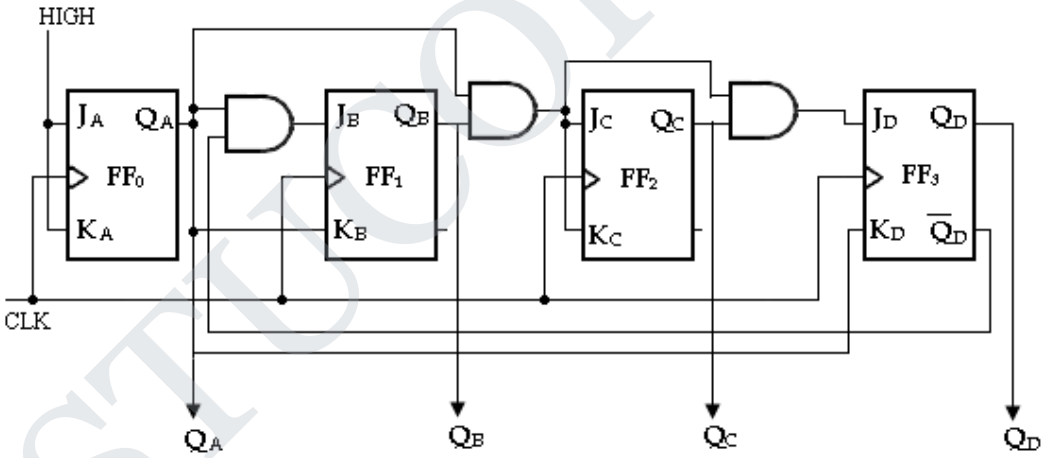
$K_C = Q_BQ_A$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



Logic Diagram:



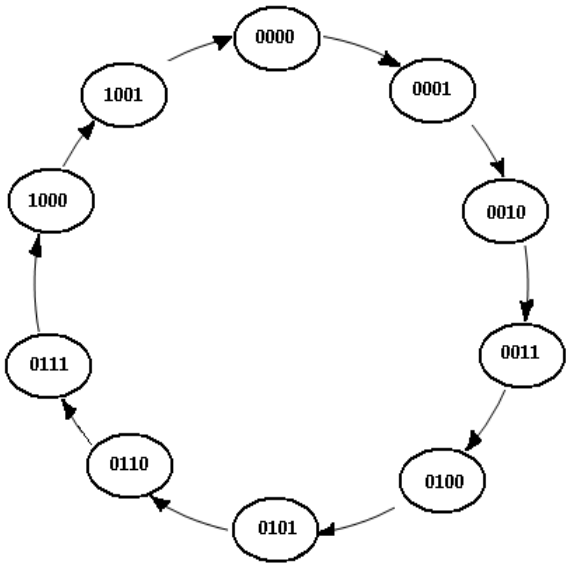
40. Design a synchronous decade counter using T flip flop and construct the timing diagram. *[Nov'15]*

$2^n \geq N = 10 \Rightarrow 2^4 > 10$. Therefore, 4 Flip-Flops are required.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

State Diagram:



State Table:

Present State	Next State
0000	0001
0001	0010
0010	0011
0011	0100
0100	0101
0101	0110
0110	0111
0111	1000
1000	1001
1001	0000

Excitation Table for JK Flip-Flop:

Present State	Next State	Inputs
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Excitation Table for Counter:

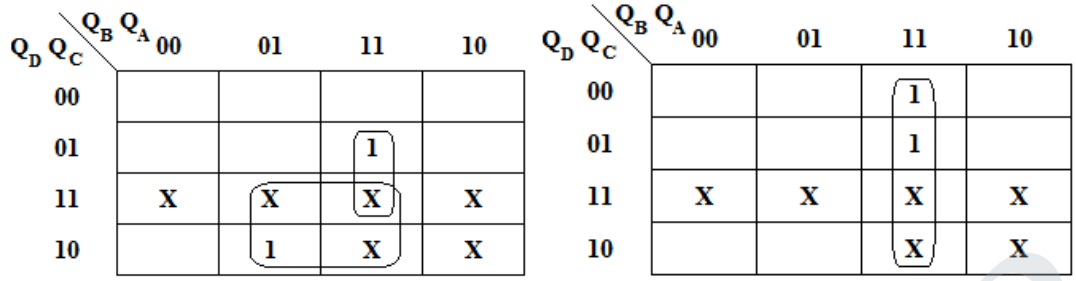
Present State				Next State				Flip-Flop Inputs			
Q_D	Q_C	Q_B	Q_A	Q_{D+1}	Q_{C+1}	Q_{B+1}	Q_{A+1}	T_D	T_C	T_B	T_A
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	1
0	0	1	1	0	1	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	1
0	1	0	1	0	1	1	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	1
0	1	1	1	1	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	1
1	0	0	1	0	0	0	0	1	0	0	1

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

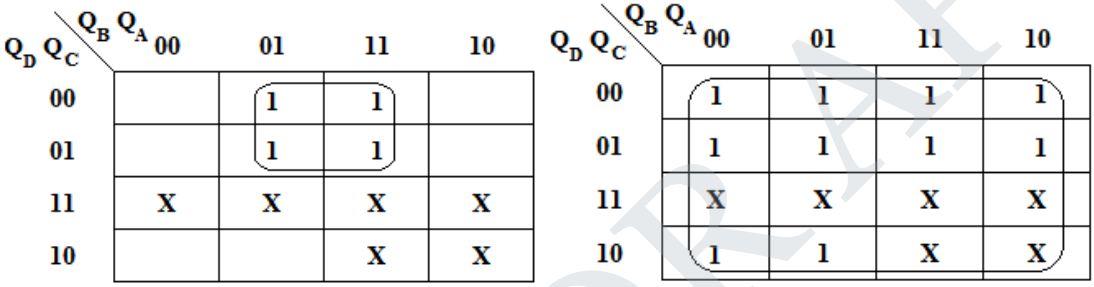
K-map Simplification:

For T_D:For T_C:



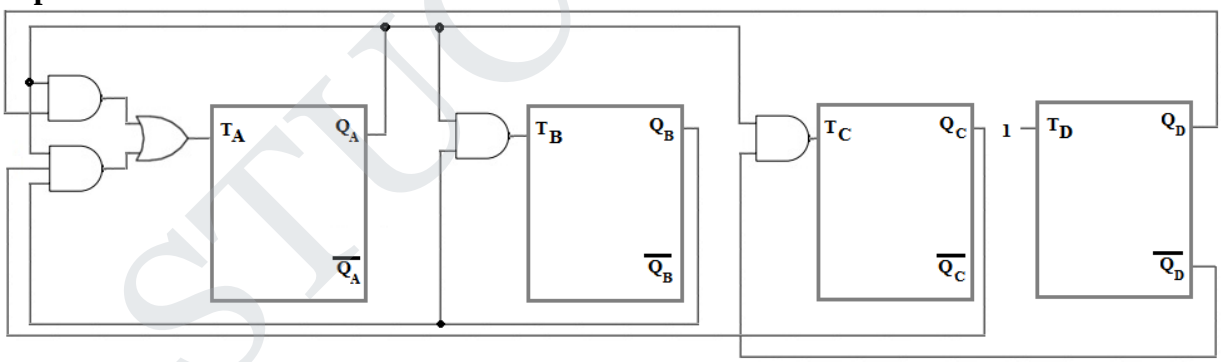
$$T_D = Q_A Q_D + Q_A Q_B Q_C \qquad T_C = Q_A Q_B$$

For T_B:For T_A:



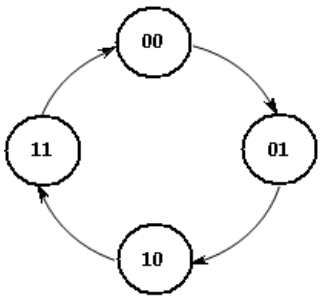
$$T_B = Q_A \overline{Q_D} \quad T_A = 1$$

Implementation:



41. Design a synchronous counter with states 0, 1, 2, 3, 0, 1 …using JK FF.

State Diagram:



State Table:

Present State	Next State
00	01
01	10
10	11
11	00

Excitation Table for

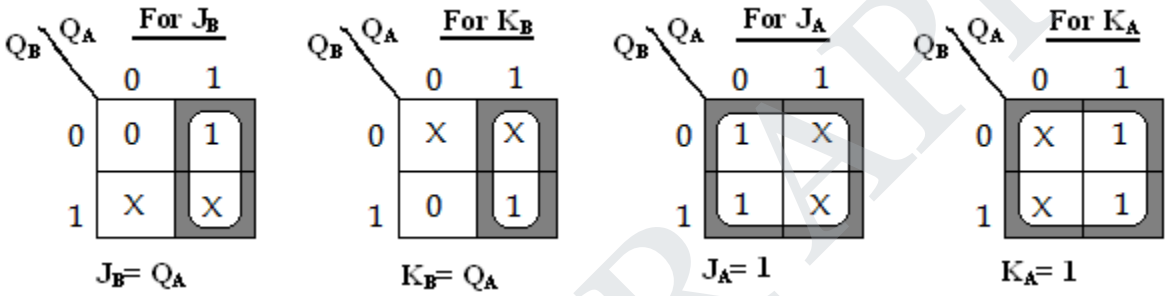
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

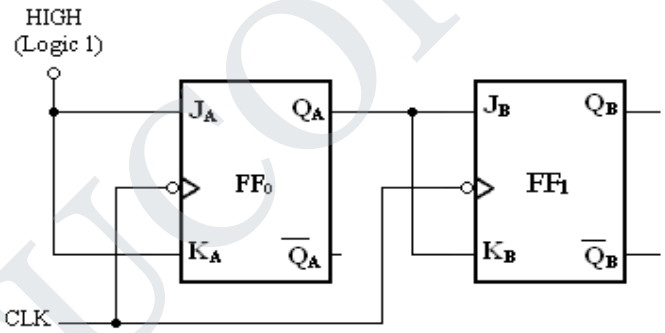
Counter:

Present State		Next State		Flip-Flop Inputs			
Q_B	Q_A	Q_{B+1}	Q_{A+1}	J_B	K_B	J_A	K_A
0	0	0	1	0	X	1	x
0	1	1	0	1	X	x	1
1	0	1	1	x	0	1	x
1	1	0	0	x	1	x	1

K-map Simplification: Logic Diagram:

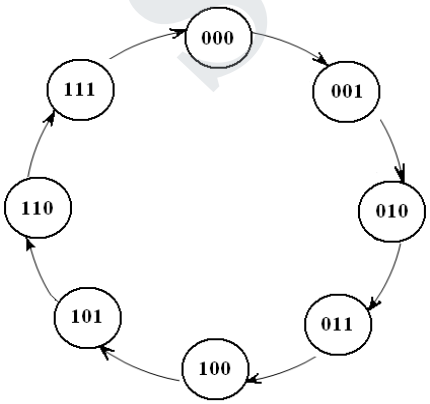


Logic Diagram:



42. Using JK Flip-Flops, design a synchronous counter which counts in the sequence 000,001, 010,011, 100, 101, 110, 111, 000. *[Nov'13]*

State Diagram



State Table

Present State	Next State
000	001
001	010
010	011
011	100
100	101
101	110
110	111
111	000

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

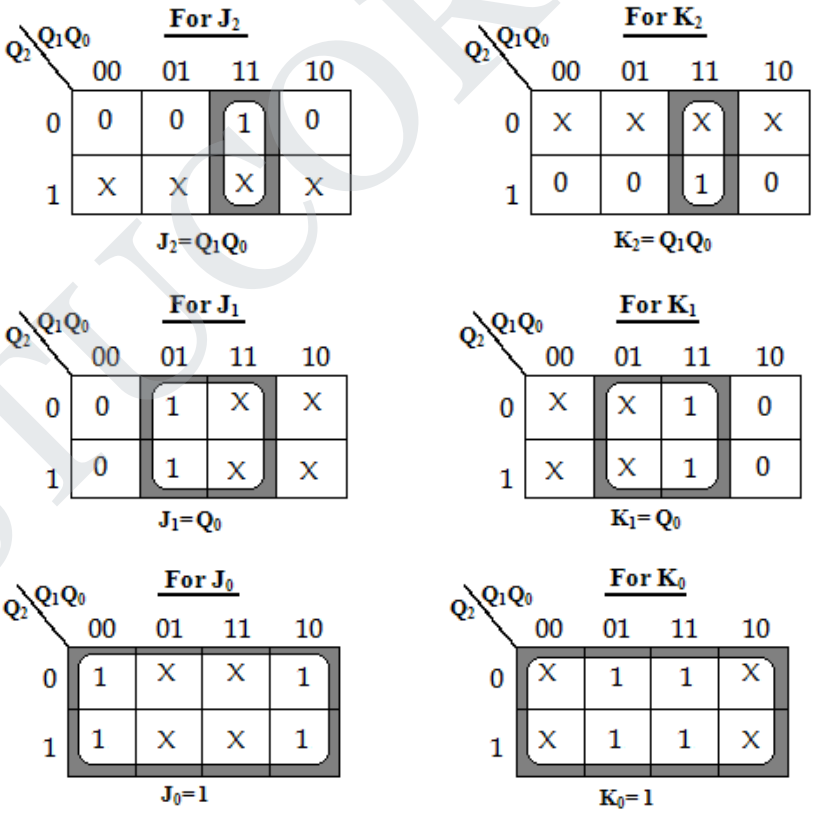
Excitation Table for JK Flip-Flop:

Present State	Next State	Inputs	
Q_n	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Excitation Table for Counter:

Present State			Next State			Flip-Flop Inputs					
Q_2	Q_1	Q_0	q_2	q_1	q_0	J_2	K_2	J_1	K_1	J_0	K_0
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

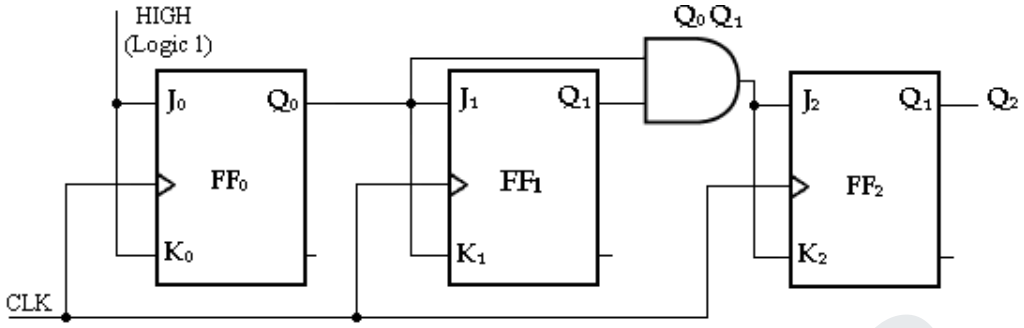
K-map Simplification



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

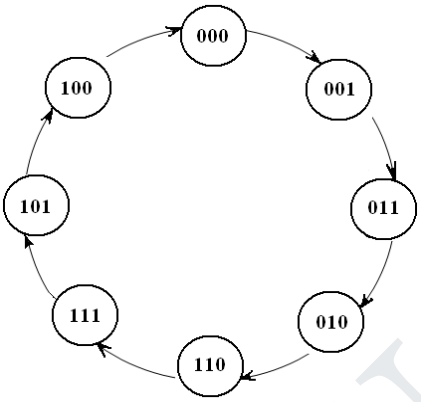
Logic Diagram



43. Design a synchronous 3-bit gray code up counter with the help of excitation table.

Gray code sequence: 000, 001, 011, 010, 110, 111, 101, 100.

State Diagram: State Table:



Present State	Next State
000	001
001	011
011	010
010	110
110	111
111	101
101	100
100	000

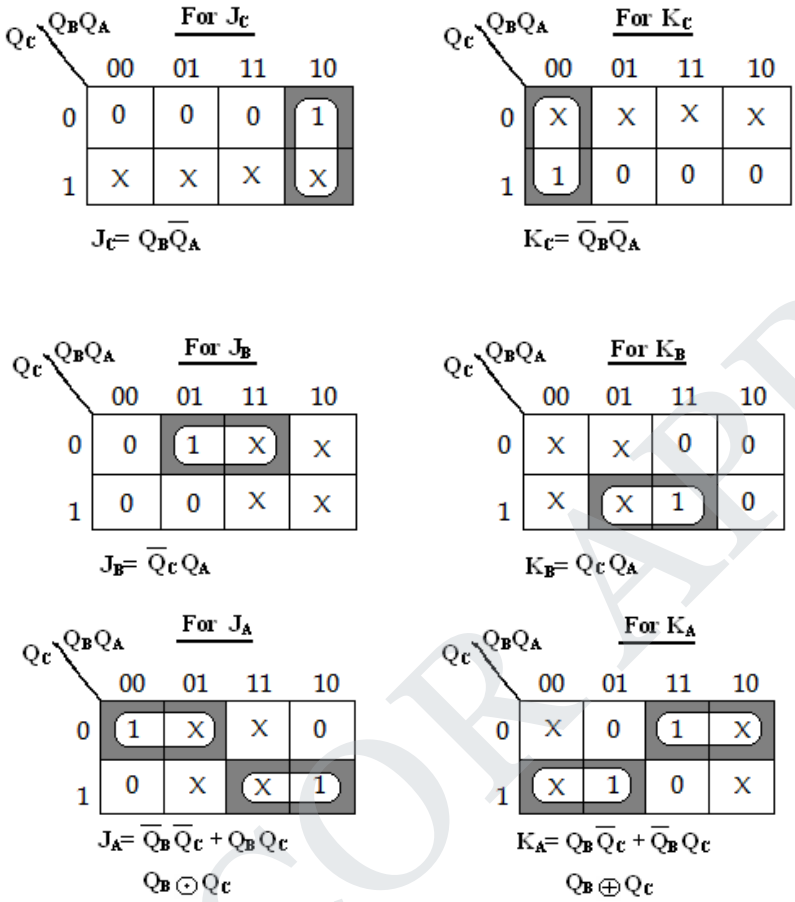
Excitation Table for counter:

Present State			Next State			Flip-Flop Inputs					
Q_C	Q_B	Q_A	Q_{C+1}	Q_{B+1}	Q_{A+1}	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	1	0	x	0	x	1	x
0	0	1	0	1	1	0	x	1	x	x	0
0	1	1	0	1	0	0	x	x	0	x	1
0	1	0	1	1	0	1	x	x	0	0	x
1	1	0	1	1	1	x	0	x	0	1	x
1	1	1	1	0	1	x	0	x	1	x	0
1	0	1	1	0	0	x	0	0	x	x	1
1	0	0	0	0	0	x	1	0	x	0	x

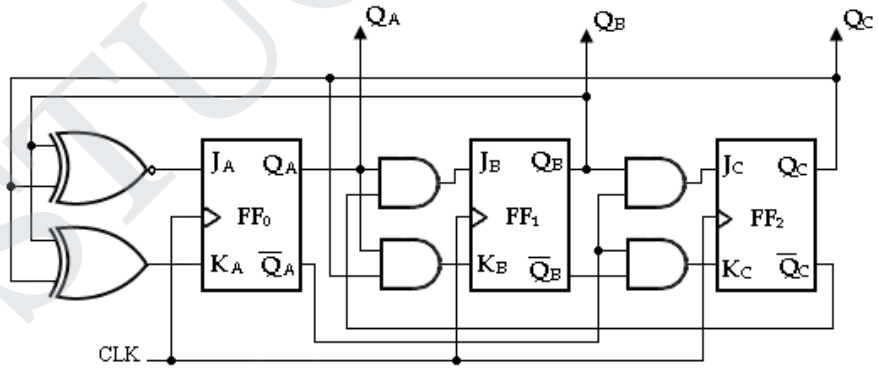
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

K-map Simplification:



Logic Diagram:



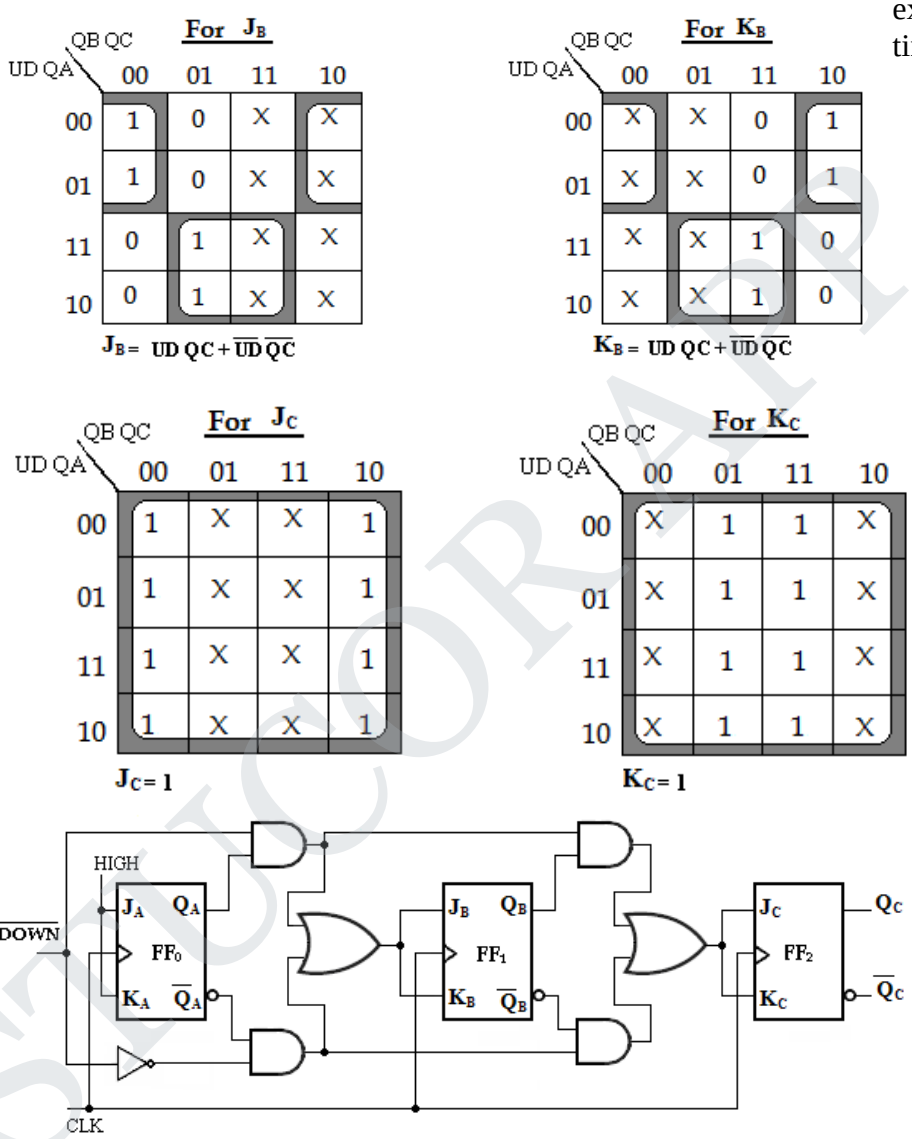
44. Design a 2-Bit Asynchronous Binary Counter.

- n 2-bit asynchronous binary counter, the clock (CLK) is applied to the clock input of first Flop-Flop (FF_0) which is always the least significant bit (LSB).
- The second Flip-Flop (FF_1) is triggered by the Q_0 output of FF_0. FF_0 changes state at the negative-going edge of each clock pulse.

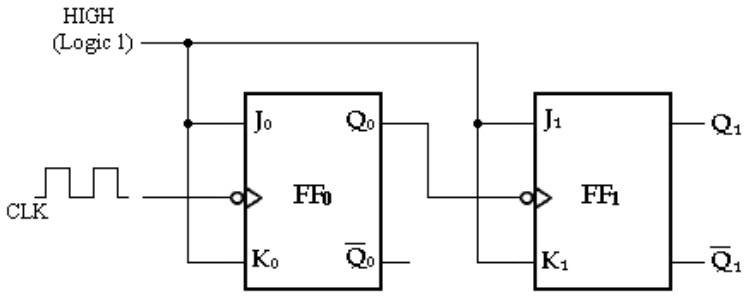
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- But FF_1 changes only when triggered by a negative-going transition of the Q_0 output of FF_0.
- Because of the inherent propagation delay time through a Flip-Flop, a transition of the input clock pulse (CLK) and a transition of the Q_0 output of FF_0 can never occur at exactly the same time. +



- Therefore, the two Flip-Flops are never simultaneously triggered, so the counter operation is ***asynchronous***.



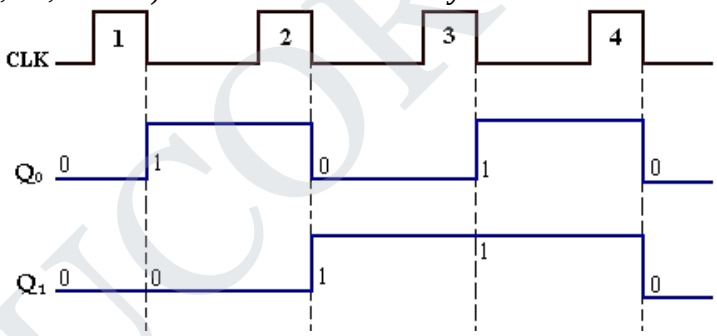
2-Bit Asynchronous Binary Counter

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

The Timing Diagram:

- Let's examine the basic operation of the asynchronous counter by applying four clock pulses to FF_0 and observing the Q output of each Flip-Flop.
- It illustrates the changes in the state of the Flip-Flop outputs in response to the clock pulses.
- Both Flip-Flops are connected for toggle operation (J =1, K = I) and are assumed to be initially RESET (Q LOW).
- The negative-going edge of CLK 1 causes Q_0 (output of FF_0) to go HIGH.
- It has no effect on FF_1 because a negative-going transition must occur to trigger the Flip-Flop. After the leading edge of CLK 1, **Q_0= 1** and **Q_1= 0**.
- The negative-going edge of CLK 2 causes Q_0 to go LOW, and it triggers FF_1, causes Q_1 to go HIGH.
- After the leading edge of CLK 2, **Q_0= 0** and **Q_1= 1**.
- The negative-going edge of CLK 3 causes Q_0 to go HIGH again and it has no effect on FF_1.
- Thus after the leading edge of CLK 3, **Q_0= 1** and **Q_1= 1**.
- The negative-going edge of CLK 4 causes Q_0 to go LOW, causing Q_1 to go LOW.
- After the leading edge of CLK 4, **Q_0= 0** and **Q_1= 0**.
- The counter has now recycled to its original state. Since it goes through a binary sequence (00, 01, 10, and 11) this counter is a ***binary counter***.



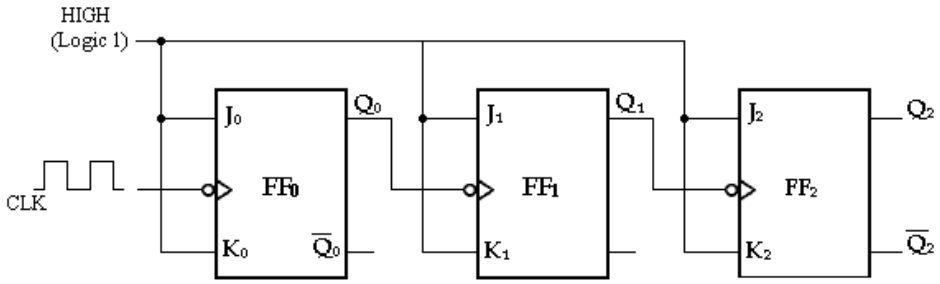
Timing diagram for 2-bit counter

45. Explain in detail the operation of a 3-Bit Asynchronous Binary Counter (OR) **Binary ripple counter.** *[Nov'12]*

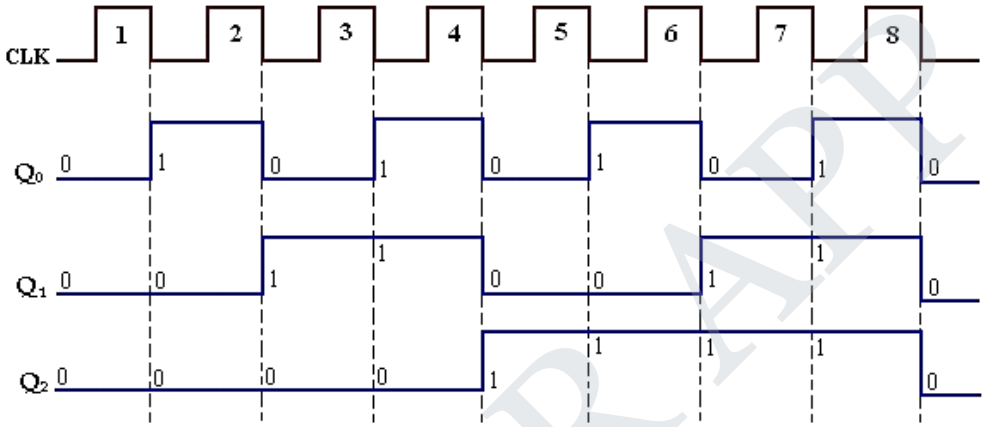
- The basic operation is the same as that of the 2-bit counter except that the 3-bit counter has eight states, due to its three Flip-Flops.
- A timing diagram is shown below for eight clock pulses.
- Notice that the counter progresses through a binary count of zero through seven and then recycles to the zero state.
- This counter can be easily expanded for higher count, by connecting additional toggle Flip-Flops.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



3-Bit Asynchronous Binary Counter



Timing diagram

Propagation Delay

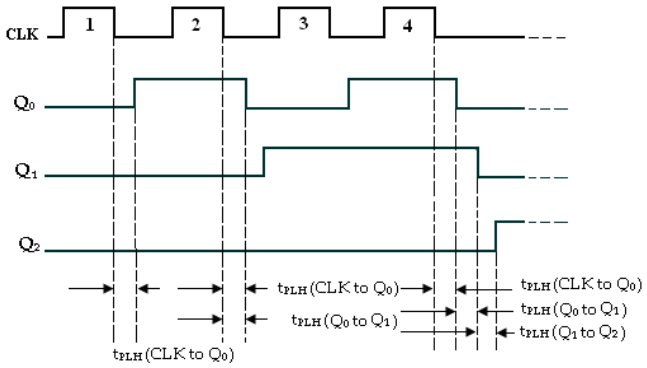
- Asynchronous counters are commonly referred to as "**ripple counters"** for the following reason:
- The effect of the input clock pulse is first "'felt" by FF_0.
- This effect cannot get to FF_1 immediately because of the propagation delay through FF_0.
- Then there is the propagation delay through FF_1 before FF_2 can be triggered.
- Thus, the effect of an input clock pulse "ripples" through the counter taking some time, due to propagation delays, to reach the last Flip-Flop.
- The figure below shows the ripple clocking effect for the first four clock pulses, with the propagation delay indicated.
- The propagation delay of the first stage is added in the propagation delay of second stage to decide the transition time for the third stage.
- The cumulative delay of an asynchronous counter is a major disadvantage in many applications because it limits the rate at which the counter can be clocked and creates decoding problems.
- For example, each Flip-Flop has a propagation delay for 10nsec,

  Total delay time, t_p= 3 x 10= 30ns
- The maximum clock frequency,

  f_{max}= $1/t_p$= $1/(30x\ 10^{-9})$= 33.33 MHz.

STUCOR APP
DOWNLOADED FROM STUCOR APP

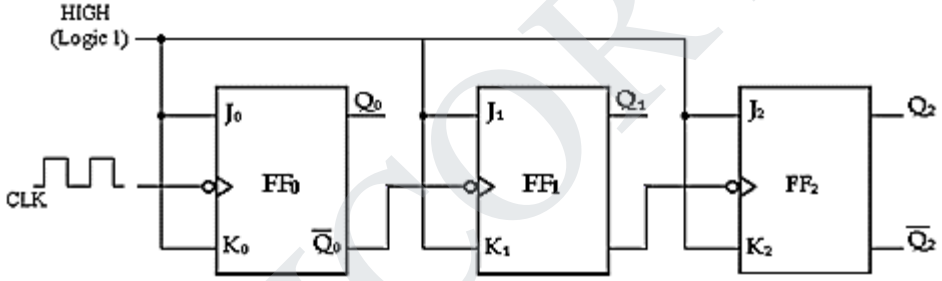
DOWNLOADED FROM STUCOR APP



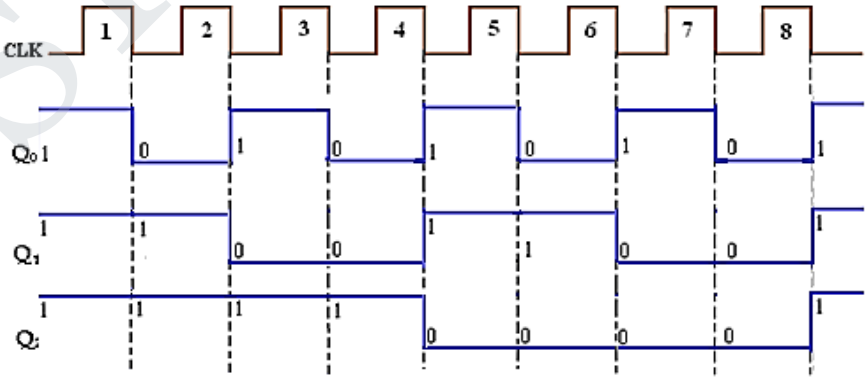
Propagation delays

46. Design an asynchronous modulo – 8 down counter using JK flip flop. *[Nov'14]*

- The basic operation is that when the clock (CLK) is applied to the clock input of first Flop-Flop (FF_0) which is always the least significant bit (LSB).
- The second Flip-Flop (FF_1) is triggered by the Q_0' output of FF_0.
- Similarly to next flip flops. FF_0 changes state at the negative-going edge of each clock pulse.
- But FF_1 changes only when triggered by a negative-going transition of the Q_0' output of FF_0.



- A timing diagram is shown below for eight clock pulses.
- Notice that the counter progresses through a binary count of seven through zero and then recycles to the seven states.
- This counter can be easily expanded for higher count, by connecting additional toggle Flip-Flops.

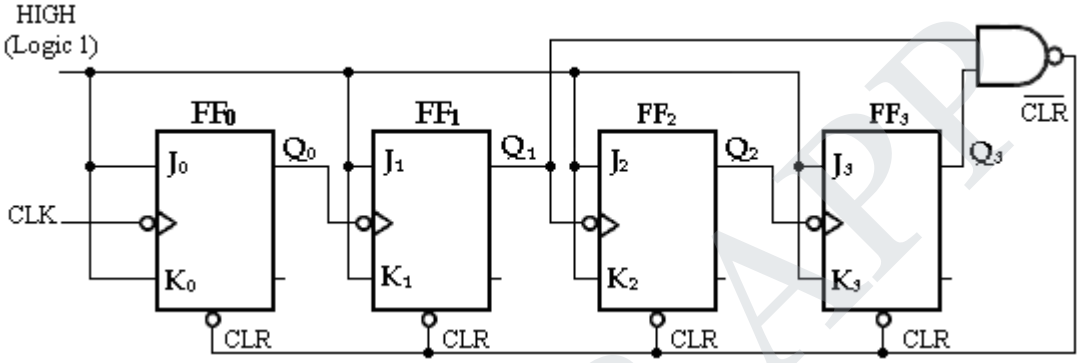


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

47. Design an Asynchronous Decade (MOD 10) Counters. *[Nov '12]*

- The modulus (N) of a counter is the number of unique states that the counter will sequence through.
- The maximum possible states (maximum modulus) of a counter is 2^n; where 'n' is the number of Flip-Flops in the counter.
- Counters can also be designed to have a number of states in their sequence that is less than the maximum of 2^n.
- The resulting sequence is called a ***truncated sequence***.
- One common modulus for counters with truncated sequences is ten, called **MOD10**.



Asynchronous Decade Counters

$2^n \geq N$

$2^4 \geq 10$

N= 4= Number of Flip-Flops

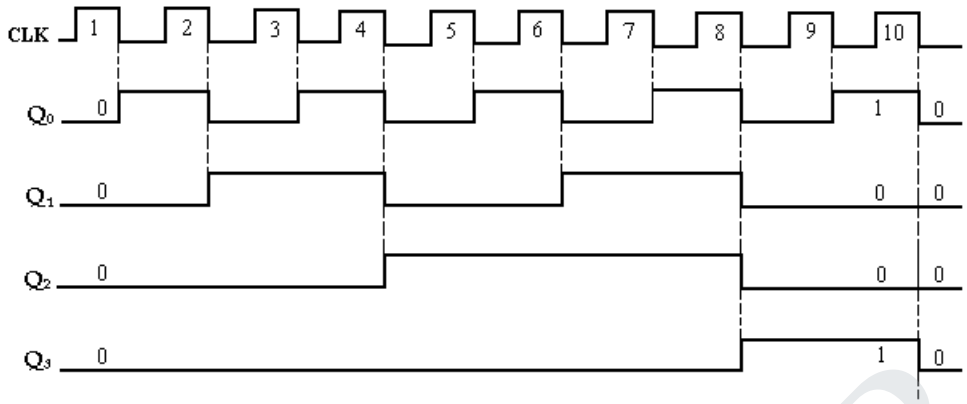
- Counters with 10 states in their sequence are called decade counters.
- A decade counter with a count sequence of zero (0000) through nine (1001) is a BCD decade counter because its ten-state sequence produces the BCD code.
- This type of counter is useful in display applications in which BCD is required for conversion to a decimal readout.
- To obtain a truncated sequence, it is necessary to force the counter to recycle before going through all of its possible states.
- For example, the BCD decade counter must recycle back to the 0000 state after the 100l state.
- A decade counter requires four Flip-Flops (three Flip-Flops are insufficient because 2^3 = 8).
- One way to make the counter recycle after the count of nine (1001) is to decode count ten (l010) with a NAND gate and connect the output of the NAND gate to the clear (CLR) inputs of the Flip-Flops.

Partial Decoding

- Notice in Figure that only Q_1 and Q_3 are connected to the NAND gate inputs.
- This arrangement is an example of partial decoding, in which the two unique states (Q_1 = 1 and Q_3 = 1) are sufficient to decode the count of ten because none of the other states (zero through nine) have both Q_1 and Q_3, HIGH at the same time.
- When the counter goes into count ten (1010), the decoding gate output goes LOW and asynchronously resets all the Flip-Flops.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



Timing Diagram

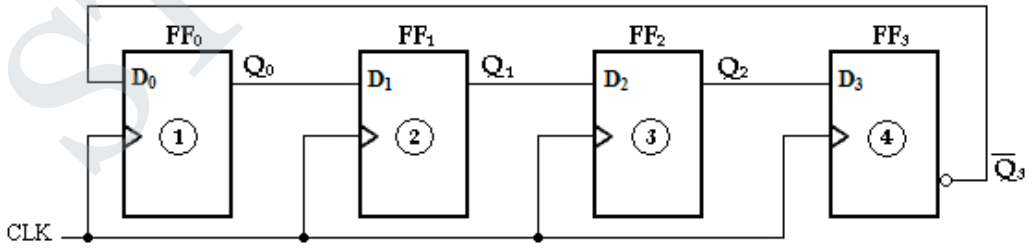
48. Draw a 4 bit Johnson counter (Shift Counter) and explain the operation.

SHIFT REGISTER COUNTERS:

- A shift register counter is basically a shift register with the serial output connected back to the serial input to produce special sequences.
- Two of the most common types of shift register counters are:
  - Johnson counter (Shift Counter),
  - Ring counter,

Johnson counters (Shift Counter):

- In a Johnson counter the complement of the output of the last Flip-Flop is connected back to the D input of the first Flip-Flop.
- This feedback arrangement produces a characteristic sequence of states as shown in table below.
- The 4-bit sequence has a total of eight states, and that the 5-bit sequence has a total of ten states.
- In general, a Johnson counter will produce a modulus of **2n**, where 'n' is the number of stages in the counter.

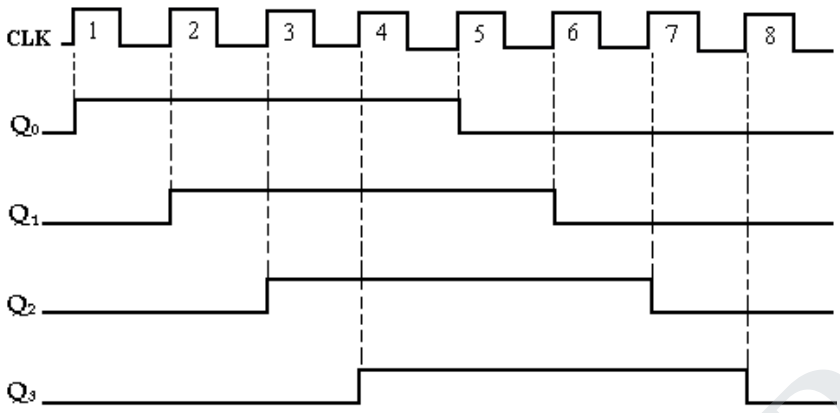


4-Bit Johnson Counter

- The Q output of each stage is connected to the D input of the next stage (assuming that D Flip-Flops are used).
- The complement output of the last stage is connected back to the D input of the first stage.

STUCOR APP
DOWNLOADED FROM STUCOR APP

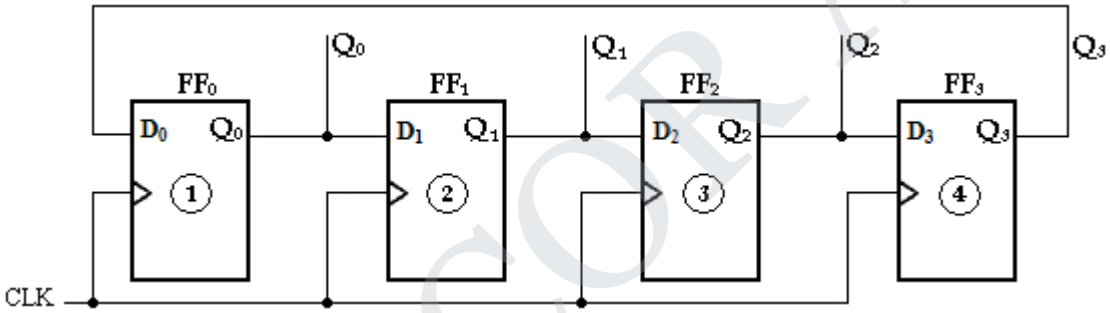
DOWNLOADED FROM STUCOR APP



Time sequence for a 4-bit Johnson counter

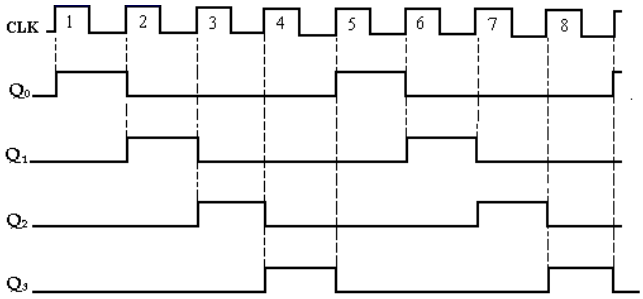
49. Draw a Ring Counters and explain its operation.

- The ring counter utilizes one Flip-Flop for each state in its sequence. It has the advantage that decoding gates are not required.
- In the case of a l0-bit ring counter, there is a unique output for each decimal digit.



Ring counter

- The output Q_0 sets D_1 input, Q_1 sets D_2, Q_2 sets D_3 and Q_3 is fed back to D_0.
- Because of these conditions, bits are shifted left one position per positive clock edge and fed back to the input.
- All the Flip-Flops are clocked together. When CLR goes low then back to high, the output is 0000.



Time sequence for a Ring counter

- The first positive clock edge shifts MSB to LSB position and other bits to one position left so that the output becomes Q= 0010.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- This process continues on second and third clock edge so that successive outputs are 0100 and 1000.
- The fourth positive clock edge starts the cycle all over again and the output is 0001.
- Thus the stored 1 bit follows a circular path (i.e., the stored 1 bits move left through all Flip-Flops and the final Flip-Flop sends it back to the first Flip-Flop).
- This action has given the name of ring counter.

50. Design a counter with the sequence 0, 1, 3, 7, 6, 4, 0. *[Nov'10]*

Step 1: Determine the number of FF needed.
Here counter should count maximum count =7= $(111)_2$ which is 3 bit.
Thus we need 3 FF
Step 2: FF to be used JK
Step 3: Determine the excitation table for counter. Here the next state of each present state is written according to given sequence. For example the next state for the present state 3 (011) is 7 (111). The count which are not in sequence are treated as don't care.

Excitation table of JK 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

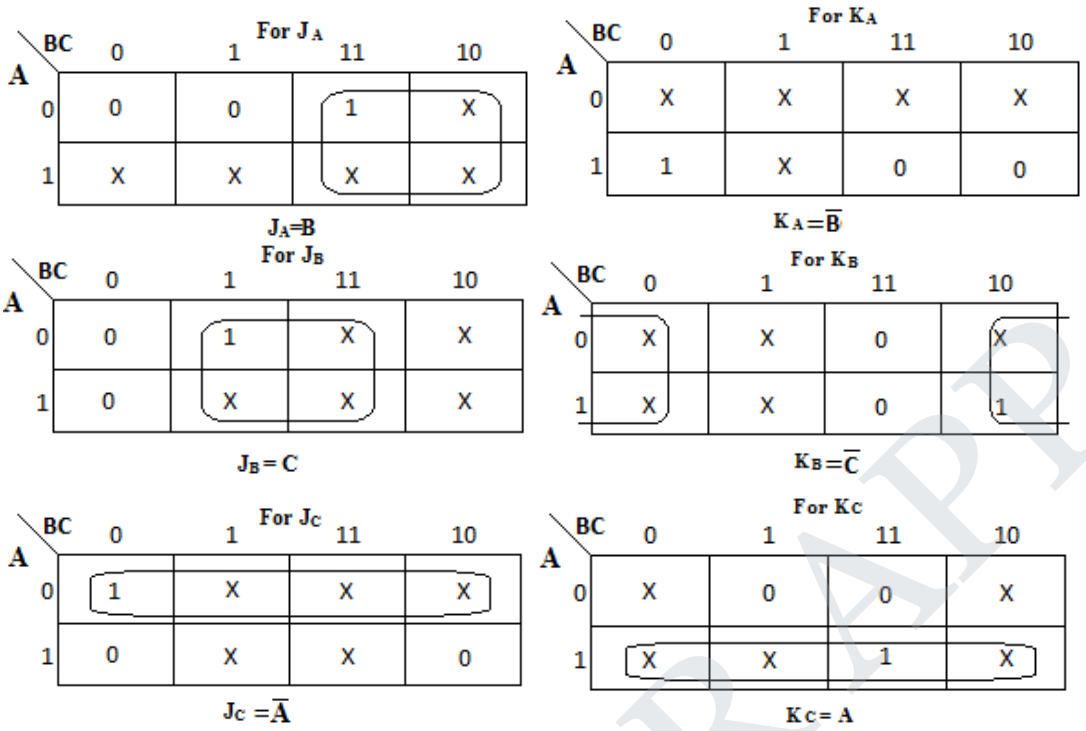
Excitation table for counter:

Present state			Next state			Flip flop inputs					
A	B	C	A^+	B^+	C^+	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	1	0	X	1	X	X	0
0	1	0	X	X	X	X	X	X	X	X	X
0	1	1	1	1	1	1	X	X	0	X	0
1	0	0	0	0	0	X	1	0	X	0	X
1	0	1	X	X	X	X	X	X	X	X	X
1	1	0	1	0	0	X	0	X	1	0	X
1	1	1	1	1	0	X	0	X	0	X	1

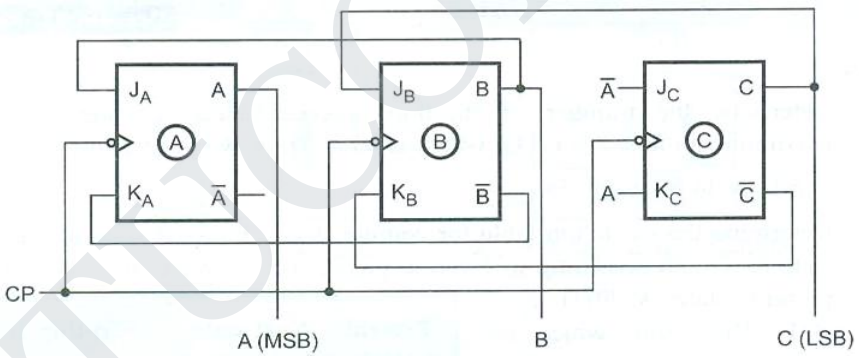
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

K-Map:



Logic Diagram:



51. Design synchronous sequential circuit that goes through the count sequence 1, 3, 4, 5 repeatedly. Use T flip-flops for your design. *[Nov'14]*

Step 1: Determine the number of FF needed.

Here counter should count maximum count =5 = $(101)_2$ which is 3 bit.

Thus we need 3 FF

Step 2: FF to be used isT

Step 3: Determine the excitation table for counter. Here the next state of each present state is written according to given sequence. The count which are not in sequence are treated as don't care.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

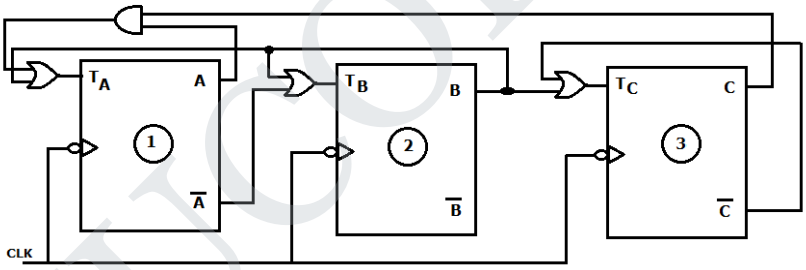
Excitation table of T flip flop:

Present State	Next State	Input
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Excitation table for counter:

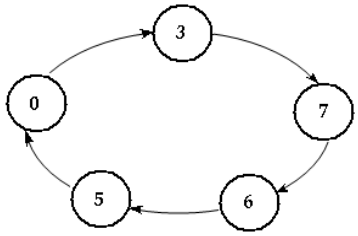
Present state			Next state			Flip flop inputs		
A	B	C	A^+	B^+	C^+	T_A	T_B	T_C
0	0	0	X	X	X	X	X	X
0	0	1	0	1	1	0	1	0
0	1	0	X	X	X	X	X	X
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	0	0	1	1	0	0
1	1	0	X	X	X	X	X	X
1	1	1	X	X	X	X	X	X

Logic Diagram:



52. Explain Lockout Condition. Explain how to avoid this problem

- In a counter if the next state of some unused state is again a used state and if by chance the counter happens to find itself in the unused states and never arrived at a used state then the counter is said to be in the lockout condition.



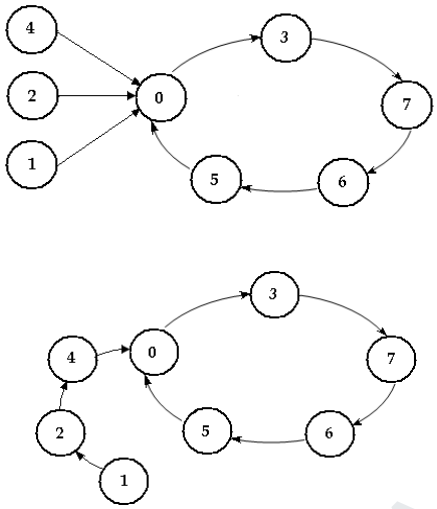
Desired Sequence

- The circuit that goes in lockout condition is called ***brushless*** circuit. To make sure that the counter will come to the initial state from any unused state, the additional logic circuit is necessary.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

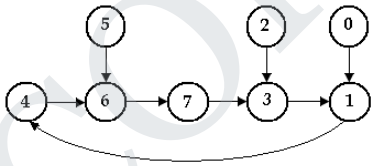
- To ensure that the lockout does not occur, the counter should be designed by forcing the next state to be the initial state from the unused states as shown below.



State diagram for removing lockout

53. Design a synchronous counter for 4 → 6 → 7 → 3 → 1 → 4 … Avoid lockout condition. Use JK type design.

State diagram



- Here, states 5, 2 and 0 are forced are forced to go into 6, 3 and 1state, respectively to avoid lockout condition.

State Excitation table:

Present State			Next State			Flip-Flop Inputs					
Q_A	Q_B	Q_C	Q_{A+1}	Q_{B+1}	Q_{C+1}	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	x	0	x	1	x
0	0	1	1	0	0	1	x	0	x	x	1
0	1	0	0	1	1	0	x	x	0	1	x
0	1	1	0	0	1	0	x	x	1	x	0
1	0	0	1	1	0	x	0	1	x	0	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	1	1	x	1	x	0	x	0

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

K-map Simplification:

For J_A

Q_A \ Q_BQ_C	00	01	11	10
0	0	1	0	0
1	X	X	X	X

$J_A = \overline{Q}_B Q_C$

For K_A

Q_A \ Q_BQ_C	00	01	11	10
0	X	X	X	X
1	0	0	1	0

$K_A = Q_B Q_C$

For J_B

Q_A \ Q_BQ_C	00	01	11	10
0	0	0	X	X
1	1	1	X	X

$J_B = Q_A$

For K_B

Q_A \ Q_BQ_C	00	01	11	10
0	X	X	1	0
1	X	X	0	0

$K_B = \overline{Q}_A Q_C$

For J_C

Q_A \ Q_BQ_C	00	01	11	10
0	1	X	X	1
1	0	X	X	1

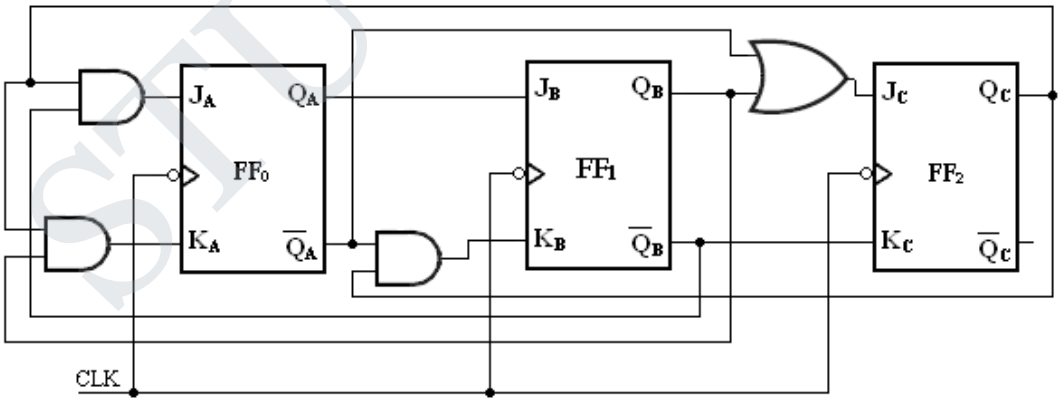
$J_C = \overline{Q}_A + Q_B$

For K_C

Q_A \ Q_BQ_C	00	01	11	10
0	X	1	0	X
1	X	1	0	X

$K_C = \overline{Q}_B$

Logic Diagram:



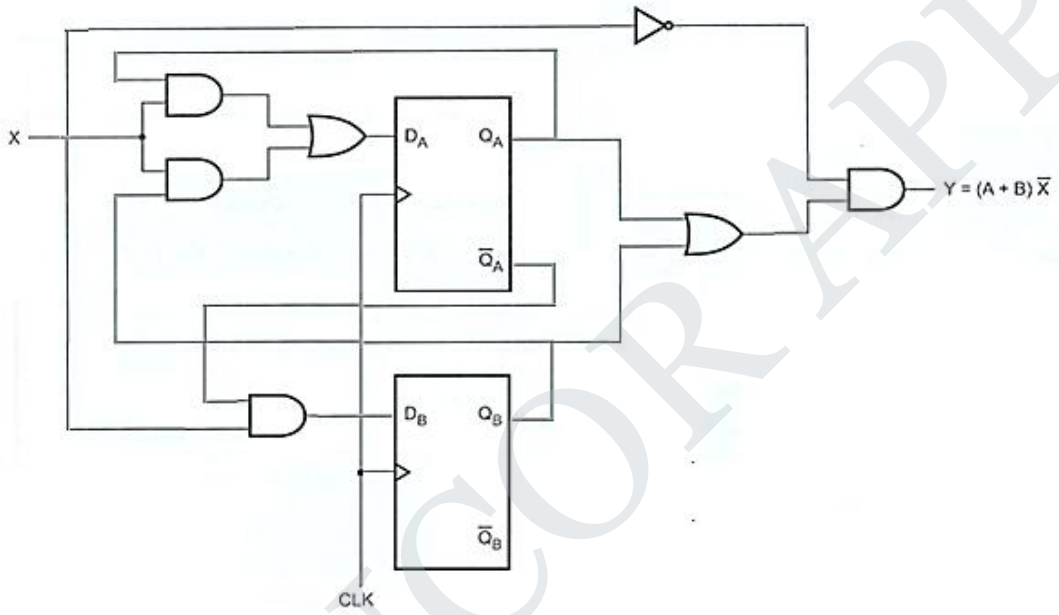
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

54. A sequential circuit with 2D a FF A and B and input X and output Y is specified by the following next state and output equation

A (t+1) = AX+BX
B (t+1) = A'X
Y = (A+B) X'
(i) Draw the logic diagram of the circuit
(ii) Derive the state table
(iii) Draw the state diagram.*[May'12, May'14, Nov'15]*

(i) Logic diagram of the circuit



(ii) Transition Table

Step 1: Plot the next state map for each flip flop

Input	Present state		D Flip Flop Input		Next state		Output
x	A	B	D_A = AX+BX	D_B= A'X	A (t+1)	B (t+1)	Y
0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	1
0	1	0	0	0	0	0	1
0	1	1	0	0	0	0	1
1	0	0	0	1	0	1	0
1	0	1	1	1	1	1	0
1	1	0	1	0	1	0	0
1	1	1	1	0	1	0	0

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

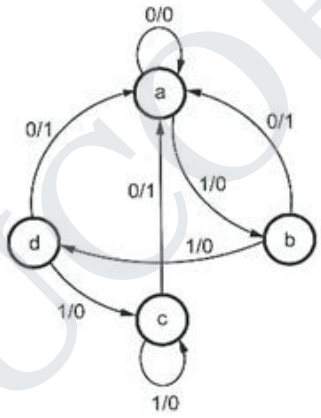
Step 2: Plot the transition table

present state		Next state				Output	
		X=0		X=1			
A	B	A^+	B^+	A^+	B^+	X=0	X=1
0	0	0	0	0	1	0	0
0	1	0	0	1	1	1	0
1	0	0	0	1	0	1	0
1	1	0	0	1	0	1	0

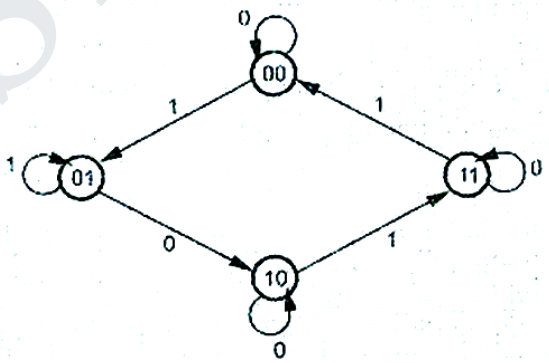
Step 3: Draw the state table
By assigning a=00, b=01, c=10 and d=11 we can write state table from the transition table

Present state	Next state		Output Y	
	X=0	X=1	X=0	X=1
	A^+ B^+	A^+ B^+		
a	a	b	0	0
b	a	d	1	0
c	a	c	1	0
d	a	c	1	0

(iii) State diagram



55. Design a synchronous sequential circuit using JK FF for given state diagram. *[Apr'10, Apr'11]*



Step 1: Since N=4. Number of flip flop needed=2
Step 2: Flip flop to be used isJK

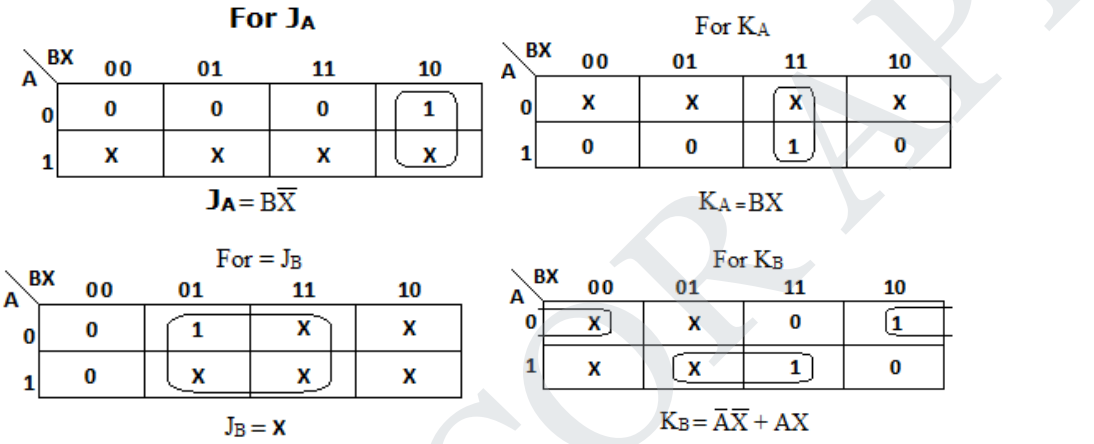
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

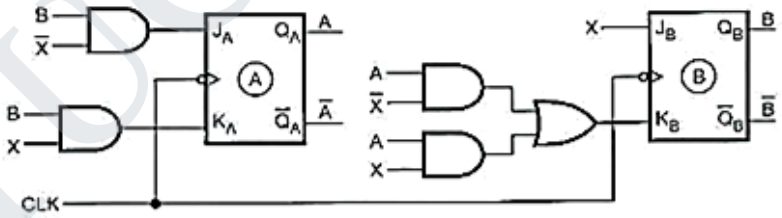
Step 3: Determine the excitation table.

Present state		Input	Next state		Flip flop inputs			
A	B	X	A^+	B^+	J_A	K_A	J_B	K_B
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	1	0	X	X	X	X
0	1	1	0	1	1	X	X	0
1	0	0	1	0	X	1	0	X
1	0	1	1	1	X	X	X	X
1	1	0	1	1	X	0	X	1
1	1	1	0	0	X	0	X	0

`Step 4: K- map simplification



Step 5: **L**ogic diagram:



56. Design a BCD counter using T flip-flop. [OR]Design BCD counter using T flip-flop inputs are TQ_1, TQ_2, TQ_4 and TQ_8. *[Apr'10, Apr'11]*

Step 1: Since N = 10, n = a (i.e.) flip – flop needed = 4

Step 2: Flip – flop to be used: T

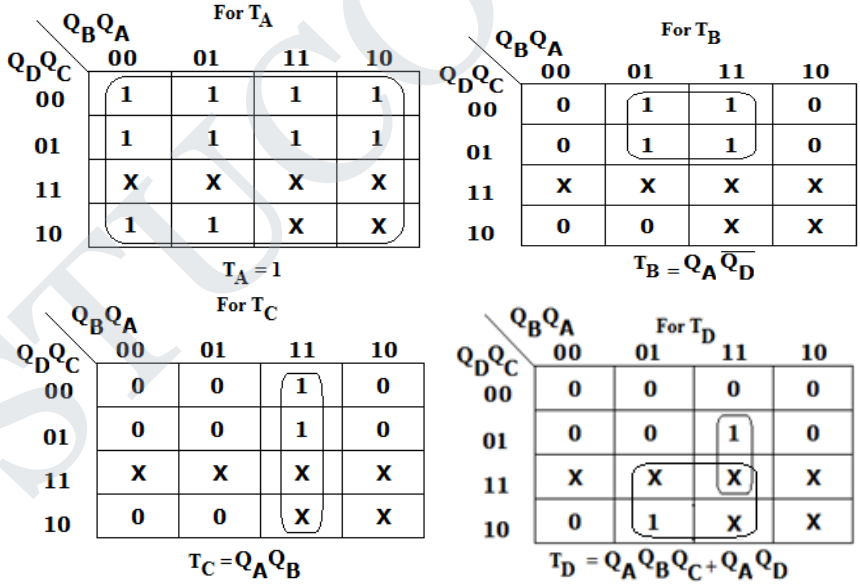
Step 3: Determine excitation table for counter

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Present State				Next State				Flip – Flop Inputs			
Q_D	Q_C	Q_B	Q_A	Q_{D+1}	Q_{C+1}	Q_{B+1}	Q_{A+1}	T_D	T_C	T_B	T_A
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	1
0	0	1	1	0	1	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	1
0	1	0	1	0	1	1	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	1
0	1	1	1	1	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	1
1	0	0	1	0	0	0	0	1	0	0	1
1	0	1	0	X	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X

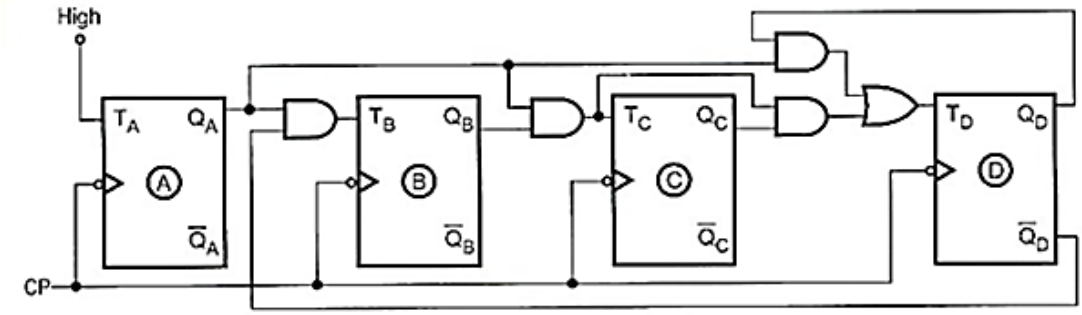
Step 4: K – map simplification



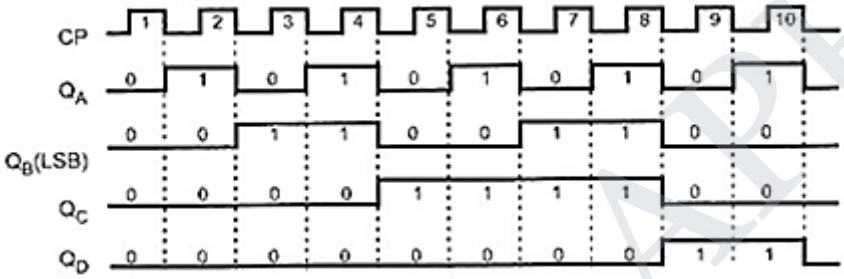
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Step 5: Logic diagram



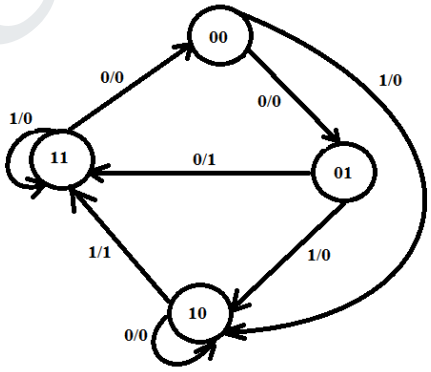
Step 6: Timing diagram (Synchronous decade counter)



57. Draw the state diagram. Derive the state equation and draw the clocked sequential circuit for the following state table. *[Apr'11]*

Present State	Next state		Output	
	X = 0	X = 1	X = 0	X = 1
00	01	10	0	0
01	11	10	1	0
10	10	11	0	1
11	00	11	0	0

State Diagram:



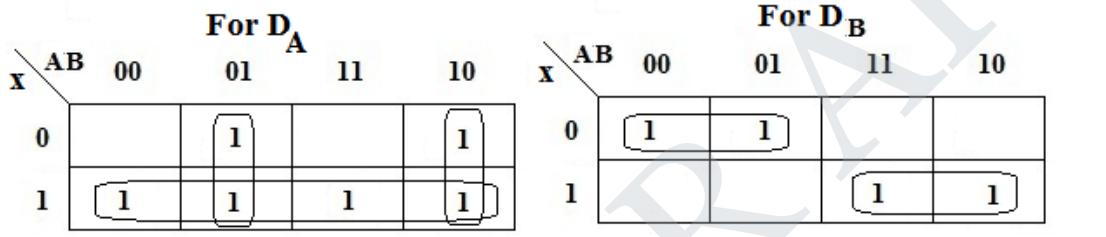
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

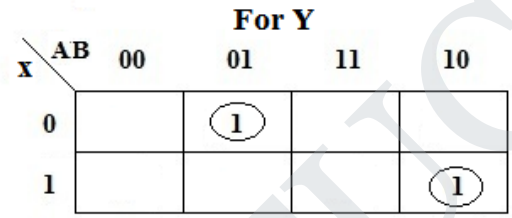
State Table:

Input	Presentstate		Next state		D Flip Flop Input		Output
x	A	B	A (t+1)	B (t+1)	D_A	D_B	Y
0	0	0	0	1	0	1	0
0	0	1	1	1	1	1	1
0	1	0	1	0	1	0	0
0	1	1	0	0	0	0	0
1	0	0	1	0	1	0	0
1	0	1	1	0	1	0	0
1	1	0	1	1	1	1	1
1	1	1	1	1	1	1	0

K-Map for state equations:

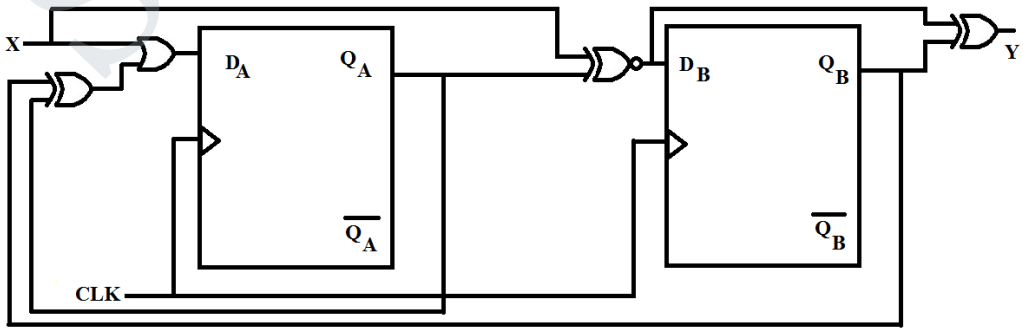


$$D_A = X + \overline{A}\,B + A\,\overline{B} \quad = X + (A \oplus B) D_B = \overline{XA} + XA = X \odot A$$



$$Y = \overline{XA}\,B + XA\,\overline{B} = X \odot A \oplus B$$

Clocked sequential circuit:

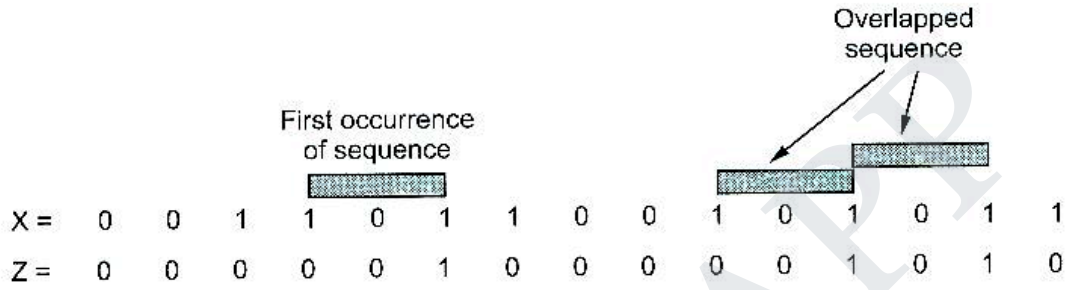


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

58. Draw the state transition diagram of a sequence detector circuit that detects'1010' from input data stream using Moore model Mealy model. *[Dec'11]*

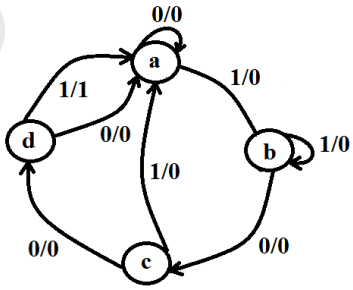
- The specified input sequence can be detected using a sequential machine called \sequence detector.
- In this circuit output goes high when a prescribed input sequence occurs.
- A typical input sequence and the corresponding output sequence for desired input sequence 101 are:



- As shown above the detection of required input sequence can occur in a longer data string and the desired input sequence can overlap with another input sequence.
- It is assumed that input can change only between clock pulses.
- Once we know the sequence which is to be detected, we can draw the state diagram for it.
- Then from the state diagram we can design the circuit.
- It is possible to implement sequence detector using both types of sequential machines: Mealy machine and Moore machine.
- The following examples illustrate how to determine the state diagram from the given input sequence and then implement the sequence detector.

59. Design a mealy model of sequence detector to detect the pattern 1001. *[Nov'15]*

State diagram:



Transition Table:

Present State		Next State				Output	
A	B	x = 0		x = 1		x = 0	x = 1
		A	B	A	B		
0	0	0	0	0	1	0	0
0	1	1	0	0	1	0	0
1	0	1	1	0	0	0	0
1	1	0	0	0	0	0	1

STUCOR APP
DOWNLOADED FROM STUCOR APP

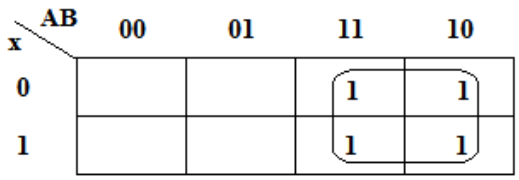
DOWNLOADED FROM STUCOR APP

State Transition Table:

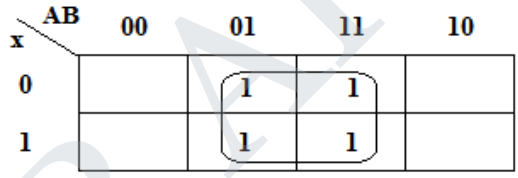
Input	Present State		Next State		Flip Flop inputs		Output
X	A	B	A	B	D_A	D_B	Z
0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	0
0	1	0	1	1	1	0	0
0	1	1	0	0	1	1	0
1	0	0	0	1	0	0	0
1	0	1	0	1	0	1	0
1	1	0	0	0	1	0	0
1	1	1	0	0	1	1	1

K-Map:

For D_A:

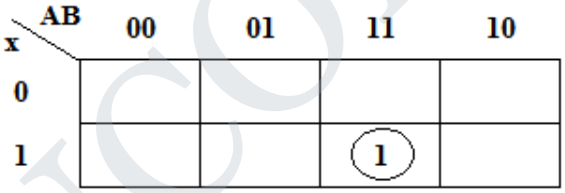


For D_B:



$$D_A = AD_B = B$$

For Z:



$$z = xAB$$

60. A sequential circuit has to JK FF A and B. The FF input functions are

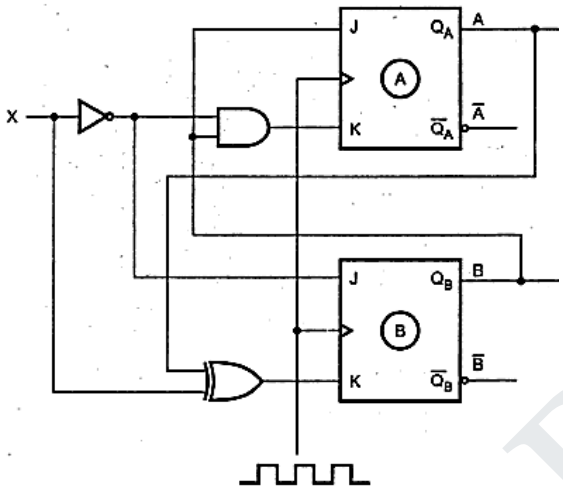
J_A= B
K_A = BX'
J_B = X'
K_B = A⊕X

(i) Draw the logic diagram of the circuit
(ii) Tabulate the state table
(iii) Draw the state diagram. *[Nov '13]*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Logic diagram:



State Table: To obtain the next – state values of a sequential circuit with JK flip flops, use the JK flip flop characteristic table.

Q	J	K	Q_{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Transition table:

Present State		Input	Flip flop Inputs				Next state	
A	B	x	JA = B	KA = $B\overline{X}$	JB = $\overline{X}$	KB = A ⊕ x	A(t+1)	B(t+1)
0	0	0	0	0	1	0	0	1
0	0	1	0	0	0	1	0	0
0	1	0	1	1	1	0	1	1
0	1	1	1	0	0	1	1	0
1	0	0	0	0	1	1	1	1
1	0	1	0	0	0	0	1	0
1	1	0	1	1	1	1	0	0
1	1	1	1	0	0	0	1	1

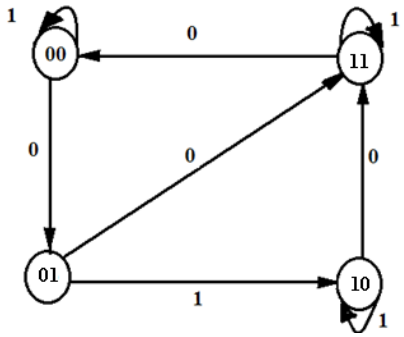
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Second form of state table:

Present State		Next state			
		x = 0		x = 1	
A	B	A	B	A	B
0	0	0	0	0	0
0	1	1	1	1	0
1	0	1	1	1	0
1	1	0	0	1	1

State diagram:



61. Compare Moore and Mealy circuits.

Synchronous or Clocked Sequential Circuits:

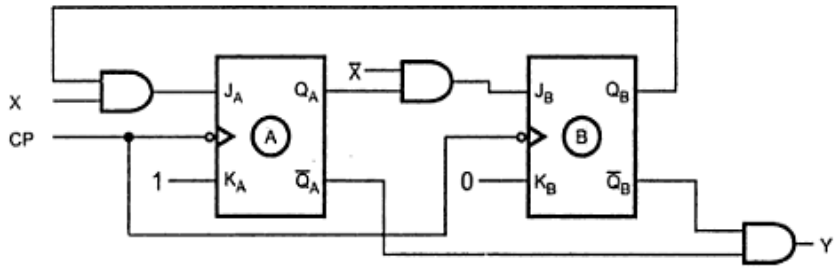
- In synchronous or clocked sequential circuits, clocked flip-flops are used as memory elements, which change their individual states in synchronism with the periodic clock signal.
- Therefore, the change in states of flip-flops and change in state of the entire circuit occurs at the transition of the clock signal.
- The synchronous or clocked sequential circuits are represented by two models.
  - Moore Circuit: The output depends on the present state of the flip-flops
  - Mealy circuit : The output depends on both the present state of the flip-flops and on the inputs.
- Sequential circuits are also called finite state machines (FSMz), the name derives from the fact that the functional behaviour of these circuits can be represented using a finite number of states.

Moore Circuit:

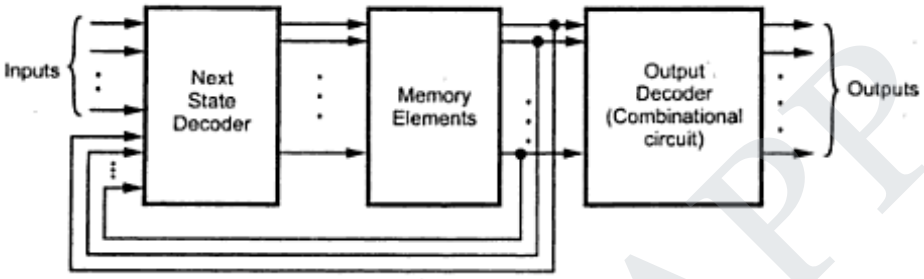
- When the output of the sequential circuit depends only on the present state of the flip-flop, the sequential circuit depends only on the present state of the flip-flop, the sequential circuit is referred to as Moore Circuit.
- The figure shows a sequential circuit which consists of two JK flip-flops and AND gate.
- The circuit has one input X and one output Y.
- The figure shows the input is used to determine the inputs of the flip-flops.
- It is not used to determine the output.
- The output is derived using only present states of the flip-flops or combination of it. $Y = Q_A Q_B$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



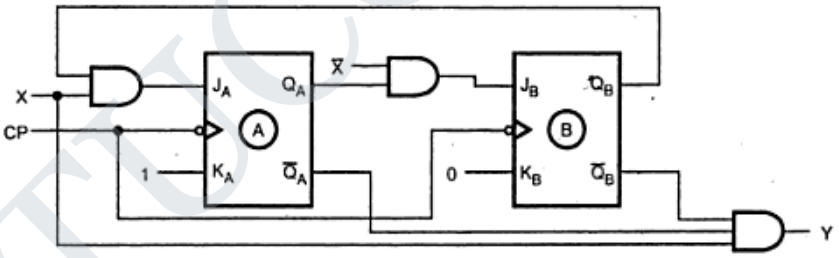
- In general form the Moore circuit can be represented with its block schematic as shown below.



- In the Moore circuit, as output depends only on the present state of the flip-flops, it appears only after the clock pulse is applied.
- It varies in synchronism with the clock input.

Mealy Circuit:

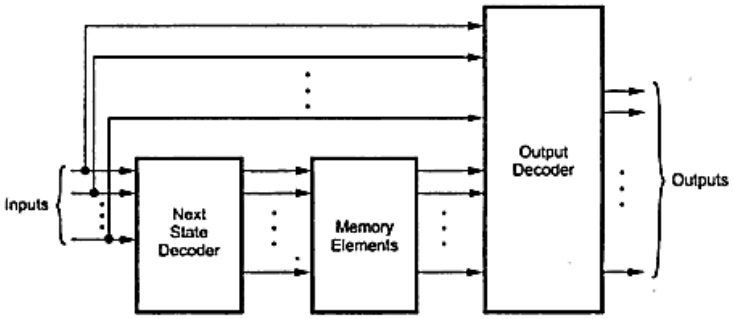
- When the output of the sequential circuit depends on both the present state of flip-flops and on the inputs, the sequential circuit is referred to as Mealy circuit.
- The figure shown below, the output of the circuit is derived from the combination of present state of flip-flops and inputs of the circuit.



- From the figure, changes in the input within the clock pulses cannot affect the state of the flip-flop.
- However, they can affect the output of the circuit.
- Due to this, if the input variations are not synchronized with the clock, the derived output will also not be synchronized with the clock and we get false output.
- The false outputs can be eliminated by allowing input to change only at the active transition of the clock (HIGH – to – LOW).
- In general form the mealy circuit can be represented with its block schematic as shown below

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



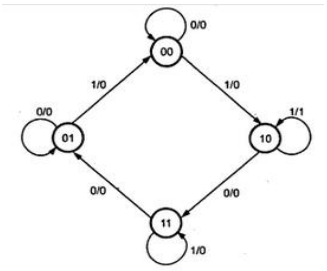
Moore Vs Mealy circuit Models:

S.No:	Moore Circuit	Mealy Circuit
1	Its output is a function of Present state only.	Its output is a function of present state as well as present input.
2	Input changes does not affect the output	Input changes may affect the output of the circuit
3	Moore circuit requires more number of states for implementing same function	It requires less number of states for implementing same function

Mealy Circuit:

State diagram:

- State diagram is a pictorial representation of a behavior of a sequential circuit.
- The figure shows a state diagram.
- The state is represented by the circle and the transition between states is indicated by directed lines connecting a circles.
- A directed line connecting a circle with itself indicates the next state is same as present state.
- The binary number inside each circle identifies the state represented by the circle.
- The directed lines are labeled with two binary numbers separated by a symbol '/'.
- The input vale that causes the state transition is labeled first and the output value during the present state is labeled after the symbol '/'.

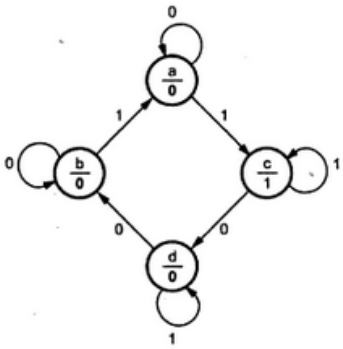


Moore Circuit:

- The directed lines are labeled with only one binary number representing the state of the input that causes the state transition.
- The output state is indicted with in circle, below the present state because output state depends only on present state and not on the input.
- The figure shows the state diagram for Moore circuit.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



State Diagram:

- The state diagram of the Moore Model sequential circuit is shown.
- Since the circuit has no outputs, the directed lines out of the circles are marked with one binary number only to designate the value of input x.

62. Reduce the number of states in the following state table and tabulate the reduced state table.

Present State	Next State		Output	
	X = 0	X = 1	X = 0	X = 1
A	f	b	0	0
B	d	c	0	0
C	f	e	0	0
D	g	a	1	0
E	d	c	0	0
F	f	b	1	1
G	g	h	0	1
H	g	a	1	0

Starting from a, and input sequence 01110010011, determine the output sequence for the given and reduced state table. [OR] **Explain the state minimization using partitioning procedure with a suitable example.** *[May'16]*

Solution:

According to given stable table we have state b and e, states d and h are equivalent.

Present State	Next State		Output	
	X = 0	X = 1	X = 0	X = 1
A	f	b	0	0
B	d	c	0	0
C	f	e	0	0
D	g	a	1	0
E	d	c	0	0
F	f	b	1	1
G	g	h	0	1
H	g	a	1	0

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

From the above stable table we have states A and C are equivalent. Reduced state table is,

Present State	Next State		Output	
	X = 0	X = 1	X = 0	X = 1
A	f	b	0	0
B	d	c	0	0
D	g	a	1	0
F	f	b	1	1
G	g	h	0	1

ii) a)Starting from "A", with the input sequence 01110010011, the output sequence using reduced state table is as follows

Input	State Transition	Output
0	a →f	0
1	f→b	1
1	b→a	0
1	a→b	0
0	b→d	0
0	d→g	1
1	g→d	1
0	d→g	1
0	g→g	0
1	g→d	1
1	d→a	0

The output sequence for reduced state table is 01000111010. Starting from "A", with the input sequence 01110010011, the output sequence using given state table is as follows

Input	State Transition	Output
0	a→f	0
1	f→b	1
1	b→c	0
1	c→e	0
0	e→d	0
0	d→g	1
1	g→h	1
0	h→g	1
0	g→g	0
1	g→h	1
1	h→a	0

The output sequence for given state table is also 01000111010

STUCOR APP
DOWNLOADED FROM STUCOR APP

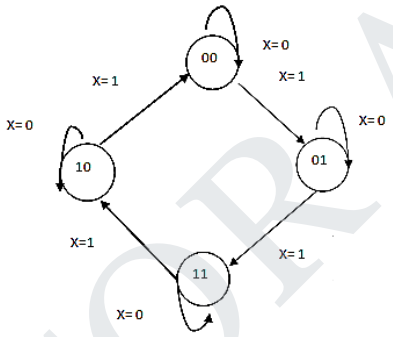
DOWNLOADED FROM STUCOR APP

63. Using partitioning minimization procedure reduce the following state table:

Present State	Next State		Output
	w = 0	w = 1	Z
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

[OR]**Design a sequential circuit with two D – flip flops A and B and one input x. When x = 0, the state of the circuit remains the same. When x = 1, the circuit goes through the state transitions from 00 → 01 → 11→ 10 → 00 and repeats.***[May'14]*

State Diagram:



Reduced state Table:

Present State		Next State			
A_n	B_n	A_{n+1}	B_{n+1}	A_{n+1}	B_{n+1}
		X = 0		X = 1	
0	0	0	0	0	1
0	1	0	1	1	1
1	0	1	0	0	0
1	1	1	1	1	0

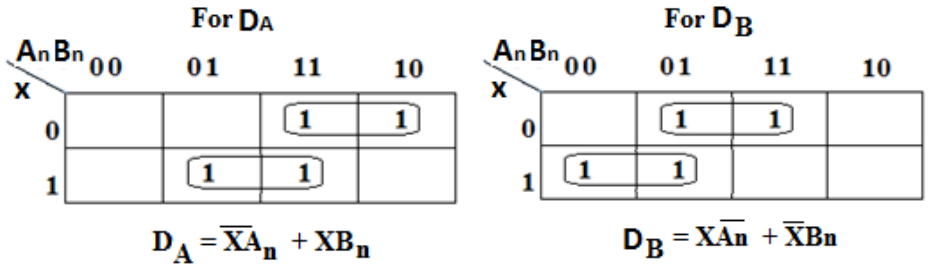
State Transition Table:

Input	Present State		Next State		Flip Flop inputs	
X	A_n	B_n	A_{n+1}	B_{n+1}	D_A	D_B
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	1	0	1	0
0	1	1	1	1	1	1
1	0	0	0	1	0	1
1	0	1	1	1	1	1
1	1	0	0	0	0	0
1	1	1	1	0	1	0

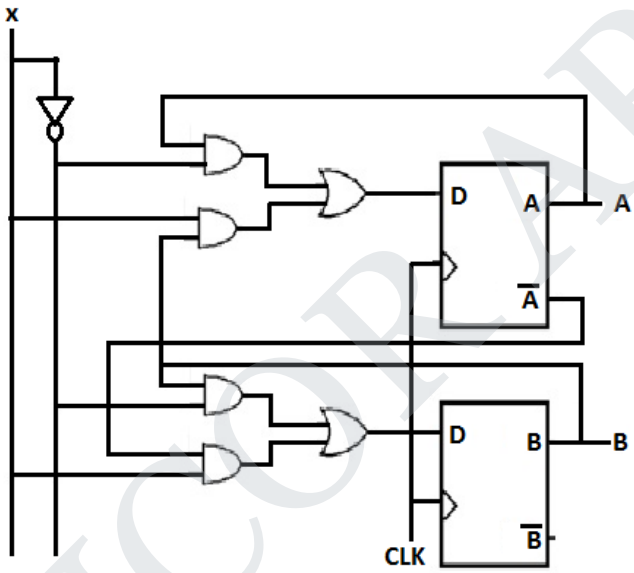
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

K- Maps:



Logic Diagram:

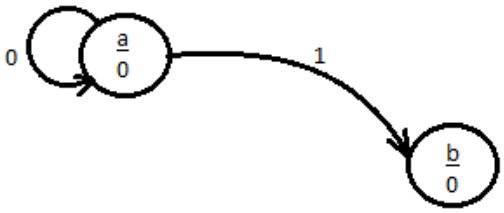


64. Design a sequence detector to detect the sequence 101 using JK flip flop. *[Apr'15]*

- The given sequence has 3-bit, so we require 3 states in the state diagram.
- Let us assuming state 'a' is an initial state.

State a:

- When input = 1, we have detected the first bit in the sequence, hence the next state to detect the next bit in the sequence.
- When input = 0, we have to remain in the same state 'a' because bit 0 is not the first bit in the sequence.
- In both the cases output is 0 [Not yet detected all the bits in the sequence]

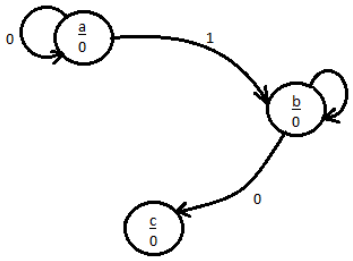


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

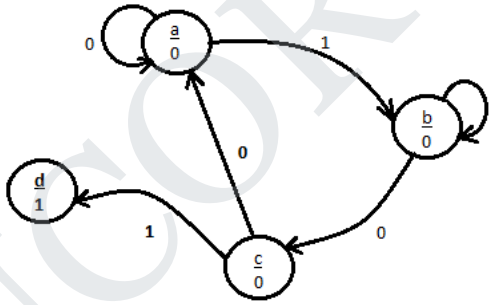
State b:

- When input = 0, we have detected the second bit in the sequence, hence we have to go to the next state to detect the next bit in the sequence.
- When the input= 1, we have to remain in state b because 1 which we have detected may start the sequence. Output is still zero.



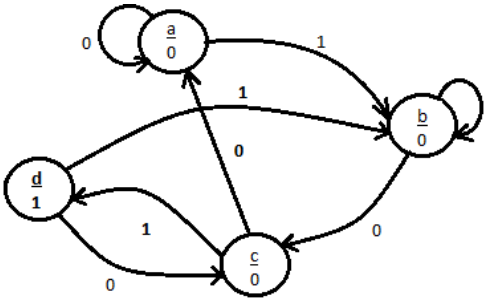
State c:

- When input = 1, the 101 sequence is completed and output must be equal to 1.
- Here, we cannot go back to state b (since output in state b is zero) and hence we have to create new state'd' with an output 1.
- If the input = 0, we have to restart checking of input sequence and hence we have to return to state a.



State d:

- Since the sequence is detected, this is the last state.
- When input = 1, we have to detected the first bit in the next sequence, so we go to state b.
- When input = 0, we have detected the second bit in the overlapped sequence, so we have to go to state c.



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

65. Design a serial adder using mealy state model. *[May'16]*

Let us considering two inputs A and B, one output S

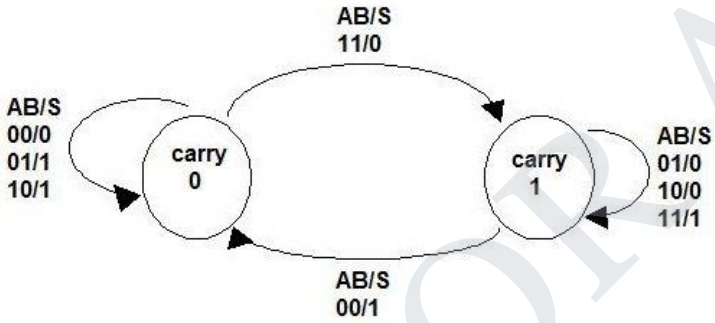
When carry = 0,

AB = 00, S = 0;
AB = 01, S = 1;
AB = 10, S = 1;
AB = 11, S = 0.

When carry = 1,

AB = 00, S = 1;
AB = 01, S = 0;
AB = 10, S = 0;
AB = 11, S = 1.

State Diagram:



State Table:

Given state of door q	AB = 00	AB = 01	AB = 10	AB = 11	AB = 00	AB = 01	AB = 10	AB = 11
	Next state				Output			
z	Z				S			
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

Transition Table:

K-map For Z				
z/AB	00	01	10	11
0	0	0	0	1
1	0	1	1	1

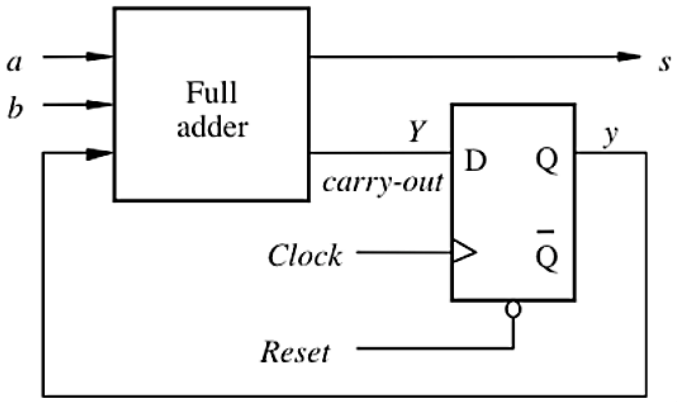
K-map For S				
z/AB	00	01	10	11
0	0	1	1	0
1	1	0	0	1

$$S = A \oplus B \oplus z$$

$$Z = AB + Az + Bz$$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



ANNA UNIVERSITY QUESTIONS

PART A

1. Difference between Combinational & Sequential Circuits.*[Nov'13]*
2. Draw the truth table and state diagram of SR flip flop. *[Nov'15]*
3. What is the drawback of SR flipflop? *[May'14]*
4. Define race around condition. *[Nov'10, Nov'11]*
5. What do you mean by race around condition in a flip flop? *(Nov'17)*
6. What is edge-triggered flip-flop? *[Nov'15]*
7. Write the characteristic table for SR flip-flop. *[Apr'11]*
8. Give characteristics equation and characteristics table of SR flip flop. *[May 16]*
9. Write down the characteristic table of JK FF and state diagram. *[Apr'11, May'12, May'13]*
10. Draw the state diagram of JK flip flop. *[Nov'16]*
11. The JK flip-flop is a universal flip-flop. Justify. *[Nov'12]*
12. What is a master-slave flip-flop? Mention the major applications of Master Slave FF. *[Nov'10]*
13. Give the characteristics equation and characteristic table of a T flip flop.*[May'17]*
14. Draw the logic diagram for T flip flop and SR flip-flop. *[Apr'10]*
15. How will you convert JK FF into DFF? *[Nov'14]*
16. Convert JK flip-flop to T flip-flop. *[Nov'10, Nov'14]*
17. Convert T Flip Flop to D Flip Flop. *[Apr'15]*
18. Define sequential circuit? *[Apr'10]*
19. Define synchronous sequential circuit. *[May'14]*
20. Give the comparison between synchronous & Asynchronous counters. *(Apr'18)*
21. How many flip flops are required for designing synchronous mod 25 counter? *[May'13]*
22. What are the state table and state diagram as applicable to sequential logical circuits? *[Nov'12]*
23. Differentiate between Mealy and Moore models. *[Nov'14, May'16, Nov'16, May'17]*
24. How does the state transition diagram of a Moore model differ from Mealy model? *[Nov'11]*
25. What is lockout? How it is avoided? *[May'12]*
26. What is meant by State assignment? *[Nov'13]*
27. State the rules for state assignment. *[Apr'15]*
28. Define address and word. *(Apr'18)*
29. What is a preset table counter and ripple counter? *(Nov'17)*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

PART B & C [8 /16 Marks]

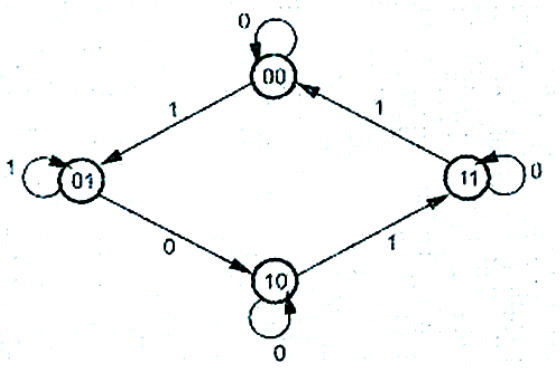
1. Explain the various types of triggering with suitable diagrams. Compare their merits and demerits. *[Nov'14, Nov'17]*
2. Explain the circuit of a SR flip-flop and explain its operation. *[Nov'14]*
3. Explain the operation of SR flip flop. *(Apr'18)*
4. Realize SR FF using NAND gates and explain its operations. *[May'12]*
5. Sketch the state diagram and state table for 'D' flip-flops. *[Nov'12]*
6. Sketch the state diagram and state table for JK' flip-flops. *[Nov'12]*.
7. Explain the operation of JK flip flop *(Apr'18)*
8. Explain the operation of a Master-Slave JK Flip-flop. *[Apr'15, May'16, May'17, Nov'17]*
9. Explain the working principle of T FF. (OR) Explain the operation of T flip flop *(Apr'18)*
10. Explain the operation, state diagram and characteristics of T flip flop. *(Nov'17)*
11. Explain the flip flop excitation tables for JK and RS flipflop. *(Apr'18)*
12. How will you convert SR Flip-Flop to JK Flip-Flop? *[May '12]*
13. Realize T flip flop using JK flip flop. [Nov'15]
14. Explain the realization of JK flipflop from T flip flop. [Nov'16]
15. Write short notes about Shift Registers. Or Explain in detail about different shift registers. (Nov'17)
16. Write short note on SIPO and draw the output waveforms. *[Nov'16]*
17. Design a 4 bit parallel –in serial out shift register using D Flip Flops. *[May'17]*
18. Design a 3-bit bi-direction shift register. *[Apr'15]*
19. Design a Sequential logic circuit for a 3 bit binary counter. *[Nov'12]*
20. Design a 4-Bit Synchronous Decade Counter. *[Nov'12]* (BCD Counter).
21. Design a MOD-5 synchronous counter using JK Flip-Flops. *[Apr'15]*
22. Design a MOD5 counter using T flip flop. *[May '16]*
23. Design a 5 bit ring counter and mention its application. *[May '17]*
24. Design MOD 7 counter using D flip flop. *[May'14]*
25. Design a MOD-7 synchronous counter using JK Flip-Flops. Write excitation table and state table.*[May'14]*
26. Design a synchronous decade counter using T flip flop and construct the timing diagram. *[Nov'15]*
27. Using JK Flip-Flops, design a synchronous counter which counts in the sequence 000,001, 010,011, 100, 101, 110, 111, 000. *[Nov'13]*
28. Explain in detail the operation of a 3-Bit Asynchronous Binary Counter (OR) Binary ripple counter. *[Nov'12]*
29. Design an asynchronous modulo – 8 down counter using JK flip flop. *[Nov'14]*
30. Design an Asynchronous Decade (MOD 10) Counters. *[Nov '12]*
31. Design a counter with the sequence 0, 1, 3, 7, 6, 4, 0.*[Nov'10]*
32. Design synchronous sequential circuit that goes through the count sequence 1, 3, 4, 5 repeatedly. Use T flip-flops for your design. *[Nov'14]*
33. A sequential circuit with 2D a FF A and B and input X and output Y is specified by the following next state and output equation
    A (t+1) = AX+BX
    B (t+1) = A'X
    Y = (A+B) X'
    a. Draw the logic diagram of the circuit
    b. Derive the state table

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

c. Draw the state diagram.*[May'12, May'14, Nov'15]*

34. Design a synchronous sequential circuit using JK FF for given state diagram.*[Apr'10, Apr'11]*



35. Design BCD counter using T flip-flop inputs are TQ_1, TQ_2, TQ_4 and TQ_8. *[Apr'10, Apr'11]*
36. Draw the state diagram. Derive the state equation and draw the clocked sequential circuit for the following state table. *[Apr'11]*

Present State	Next state		Output	
	X = 0	X = 1	X = 0	X = 1
00	01	10	0	0
01	11	10	1	0
10	10	11	0	1
11	00	11	0	0

Draw the state transition diagram of a sequence detector circuit that detects'1010' from input data stream using Moore model Mealy model. *[Dec'11]*

37. Design a mealy model of sequence detector to detect the pattern 1001.*[Nov'15]*
38. A sequential circuit has to JK FF A and B. The FF input functions are

$J_A = B$
$K_A = BX'$
$J_B = X'$
$K_B = A \oplus X$

a. Draw the logic diagram of the circuit
b. Tabulate the state table
c. Draw the state diagram. *[Nov '13]*

39. Design a sequential circuit with two D – flip flops A and B and one input x. When x = 0, the state of the circuit remains the same. When x = 1, the circuit goes through the state transitions from 00 → 01 → 11→ 10 → 00 and repeats. *[May'14]*
40. Design a sequence detector to detect the sequence 101 using JK flip flop. *[Apr'15]*
41. Design a serial adder using mealy state model.*[May'16]*

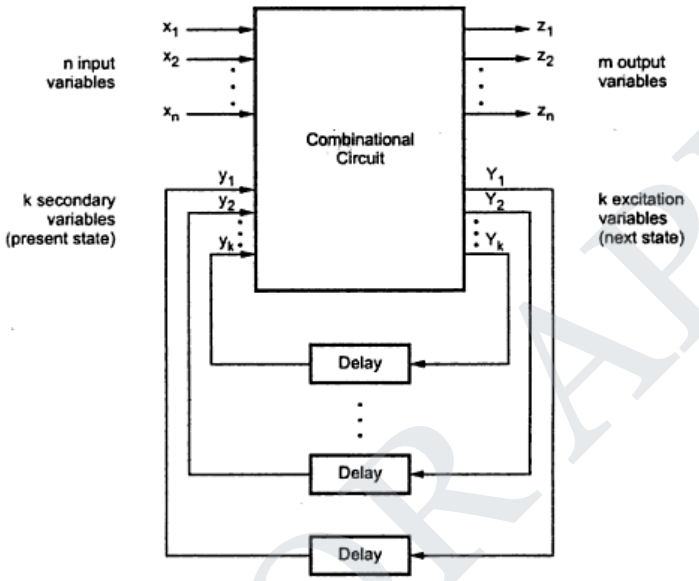
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

UNIT IV PART A

1. **Define asynchronous sequential machine.** *[Nov'10]*
   - In asynchronous sequential circuits, no clock pulses and only the input variables are responsible for state change.

2. **Sketch the block diagram of an asynchronous sequential circuit?** *[Nov'12]*



3. **How does the operation of an asynchronous input differ from that of a synchronous input?** *[May'12]*
   - The state assignment step in asynchronous circuits is essentially the same as it is for synchronous circuit, except for one difference in synchronous circuit, the state assignment are made with the objective of circuit reduction.
   - In asynchronous circuit, the objective of the state assignment is to avoid critical races.

4. **What is the difference between asynchronous and synchronous sequential circuits?** *[Apr'11]*

Sl. No.	Synchronous sequential circuits	Asynchronous sequential circuits
1.	Same clock enables all flip flops for every state change, along with inputs	No clock pulse. Only the input variables are responsible for state change.
2.	No propagation delay accumulation so faster.	Cumulative propagation delay. So slower.
3.	Logic Race problem is absent.	Prone to logic race problem.

5. **What are the drawbacks in designing asynchronous sequential machines?** *[May'14]*
   - Difficult to design
   - Difficult to predict the state
   - Timing problem

STUCOR APP
DOWNLOADED FROM STUCOR APP

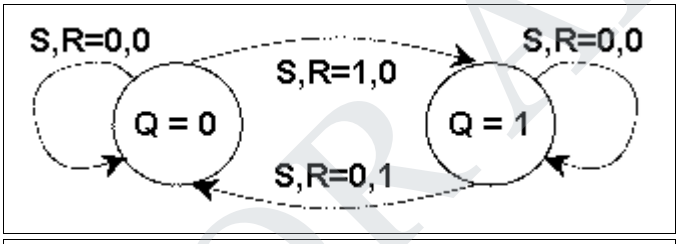
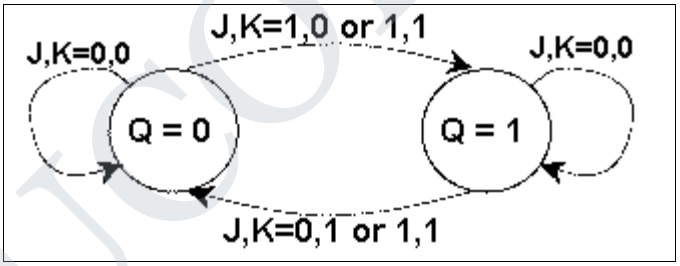
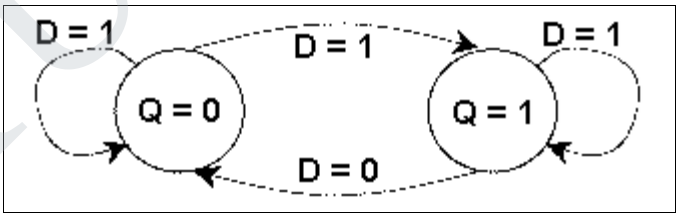
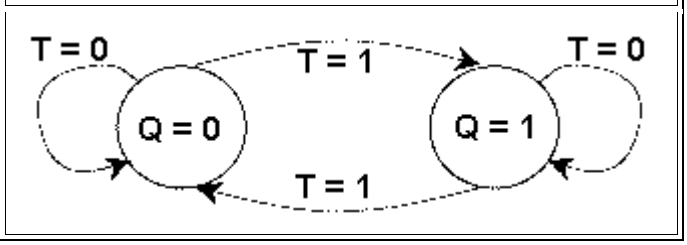
DOWNLOADED FROM STUCOR APP

6. **What is fundamental mode of operation in asynchronous sequential circuits?** *[Nov'11]*
   - The input variables changes only when the circuit is state
   - Only one input variable can change at a given time
   - Inputs are levels and not pulses

7. **Compare Pulsed mode and fundamental mode asynchronous circuit.** *[Nov'15]*

Pulsed mode	Fundamental mode
Input signals change only one at a time and only when the clock is in stable condition.	Inputs are pulse triggered and pulse width is long enough for the circuit to respond to inputs.
Inputs are given as levels.	Inputs are given as pulses.

8. **Draw the state diagram of SR flip-flop.***[Nov'10]*[OR] **Give the characteristic equation and state diagram of JK flip-flop.** *[Apr'10]*

NAME	STATE DIAGRAM
SR	
JK	
D	
T	

State diagrams of the four types of flip-flops.

- You can see from the table that all four flip-flops have the same number of states and transitions.
- Each flip-flop is in the set state when Q=1 and in the reset state when Q=0.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- Also, each flip-flop can move from one state to another, or it can re-enter the same state.
- The only difference between the four types lies in the values of input signals that cause these transitions.

9. What is meant by transition table? *[Nov' 13]*

- A state table with binary assignment is called transition table.
- Transition table of asynchronous sequential circuit is similar to table used for synchronous sequential circuits.
- It is constructed which shows the next states of flip-flops as a function of the present state and inputs.

10. Define flow table in asynchronous sequential circuit. *[May'12]*[OR]**What is flow table? Give example.***[May'17]*

- In asynchronous sequential circuit state table is known as flow table because of the behavior of the asynchronous sequential circuit.
- The stage changes occur independent of a clock, based on the logic propagation delay, and cause the state to flow from one to another.

11. What is the difference between flow table and transition table? *[May'13]*

- A state table with binary assignment is called transition table.
- It is constructed which shows the next states of flip-flops as a function of the present state and inputs.
- During the design of asynchronous sequential circuits, it is more convenient to name the states by letter symbols without making specific reference to their binary values.
- Such a table is called flow table.

12. Define Race condition in Asynchronous sequential circuit? *[May'13]*[OR]**Define races in asynchronous sequential circuits.** *[Nov'16]*

- When two or more binary state variables change their value in response to a change in input variables, race condition occurs in asynchronous sequential circuit.
- In case of unequal delays, a race condition may cause the state variables to change in an unpredictable manner.

13. What is a deadlock condition? *[Nov'14]*

- In a counter, if the next state of some unused state is again in some unused state, it may happen that the counter remains in unused state, never to arrive at a used state.
- Such condition is called a lockout condition or dead lock condition.

14. How can we avoid the lockout condition?

- To avoid the lockout, the counter should be provided with an additional logic circuitry which will force the counter from an unused state to the next state as initial state.

15. What is Self-starting counter? *[Apr'10]*

- The counter which never goes in lockout condition is called Self-starting counter.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

16. What is write cycle time?

- Two write cycle time is the minimum time for which an address must be held stable on the address bus, in write cycle.

17. Name types of Hazards. *[Nov'13]*[OR] **What are hazards in asynchronous sequential circuits?** *[Nov'11]*[OR] **State the hazards in asynchronous sequential circuits?** *[Apr'11]*

- A Hazard is a status where the output of the system is not what it should be i.e., it is temporarily false.
- For example, an output which is supposed to be a '0' at an instant, but may occur as a '1'.
- If this output is an input to another system, it may cause malfunction of the system.
  - Static hazard
    - Static-0 hazard
    - Static-1 hazard
  - Dynamic hazard

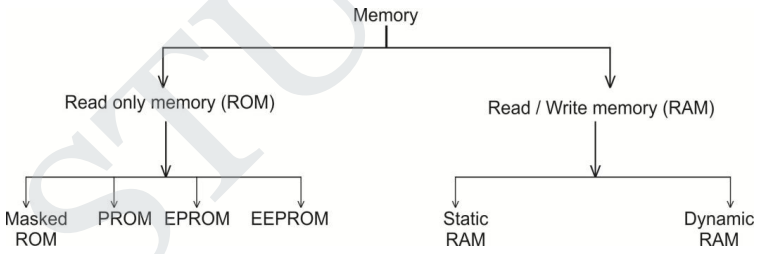
18. State the difference between static 0 and static 1 hazard. *[Apr'15]*

STATIC-0 HAZARD	STATIC-1 HAZARD
When the output is to remains at the value 0 and a momentary 1 output is possible, during the transition between the two input states.	Two input states both produce a 1 output steady state and a momentary 0 output is possible, during the transition between the two input states

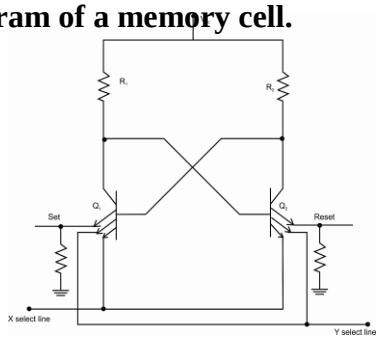
19. What is memory cycle?

- Memory cycle is represented as read and write memory cycle with their timing parameters.

20. What are the different classifications of memory? *[Nov'13]*



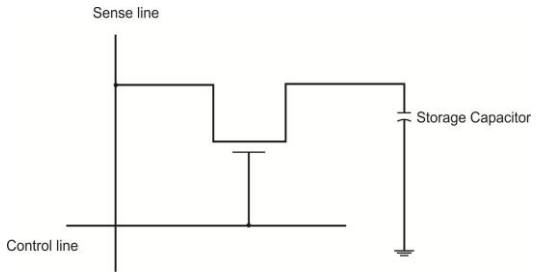
21. Draw the logic diagram of a memory cell.



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

22. Draw the basic dynamic memory cell?



23. What is meant by memory expansion? Mention its limits?

- The memory expansion can be achieved in two ways:
  - By expanding word size
  - By expanding memory capacity.

Limits: (i) The expansion of word size is limited by the data bus width.
(ii) The expansion of memory capacity is limited by address bus width.

24. What is meant by static and dynamic memories?

Static memory:

- Static RAM contains less memory cells per unit area.
- Its access time is less hence faster memories. Static RAM consists of number of flip-flops.
- Each flip-flop stores one - bit.
- Refreshing Circuitry is not required. Cost is more.

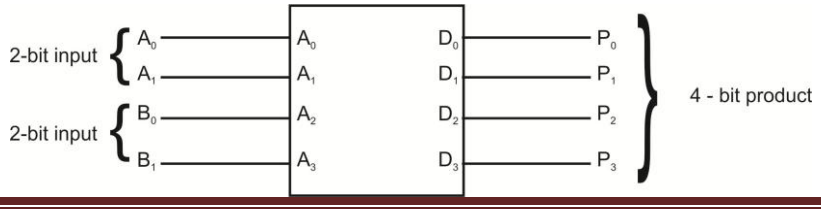
Dynamic memory:

- Dynamic RAM contains more memory cells as compared to static RAM per unit area.
- Its access time is greater than static RAMs. Dynamic RAM stores the data as a change on the capacitor.
- It consists of MOSFET and the capacitor for each cell. Refreshing Circuitry is required.
- Cost is less.

25. What are the advantages of static RAM compared to Dynamic RAM?

S. No.	Static RAM	Dynamic RAM
1.	Its access time is less hence faster memories.	Dynamic RAM contains more memory cells as compared to static RAM per unit area.
2.	Refreshing circuit is not required	Cost is less

26. Implement a 2-bit multiplier using ROM?



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

27. Name the types of ROM. *[Apr'11]*

There are four types of ROM:

- Masked ROM
- PROM
- EPROM
- EEPROM or E^2PROM.

28. Whether ROM is classified as nonvolatile storage device? Why?

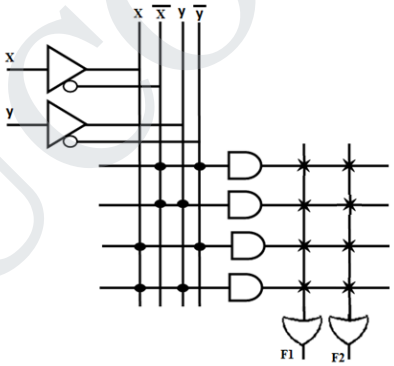
- It is a read only memory.
- We cannot write data in this memory.
- It is non-volatile memory. (i.e.) it can hold data even if power is tuned off generally, ROM is used to store the binary codes for the sequence of instruction you want the computer to carry out and data such as look up tables.
- This is because this type of information does not change.

29. What is a PROM? *[Apr'15],[Nov'15]*

- PROM is Programmable Read Only Memory.
- It consists of a set of fixed AND gates connected to a decoder and a programmable OR array.

30. Draw 4 X 2 ROM with AND –OR gates. *[Nov'13]*

- For a 4x2 ROM, there are four number of fixed AND gates and two number of programmable OR gates as shown in the figure below.



31. What is an EAPROM?

- An EAPROM is otherwise called as EEPROM – (Electrically Erasable Programmable Read only Memory) and EAPROM – (Electrically Alterable Programmable read only Memory).

32. How is individual location in an EEPROM Programmed or erased?

- Since it is electrically erasable memory by activating particular row and column it is possible that individual can be programmed or erased.

33. Write the advantages of EPROM over PROM.

- The important point is that we can erase the stored data in the EPROM by exposing the chip to ultraviolet light through its quartz window for 15 to 20 minutes.

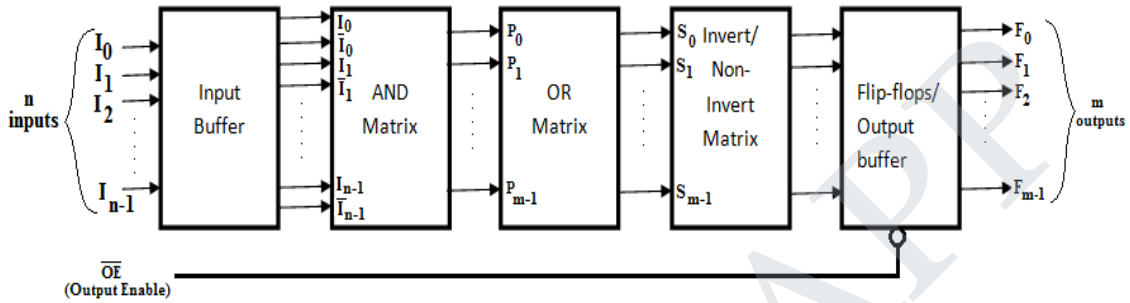
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

34. What is PLA? *[May'12](Nov'17)*

- PLA stands for Programmable Logic Array, which is a LSI component.
- In PLA, both AND & OR gates have fuses at the input, therefore in PLA both AND & OR gates are programmable.
- The output from OR gates go through fuses as inputs to output inverters so that the final output can be programmed as either AND-OR or AND-OR-INVERT.

35. Draw the block diagram of PLA. *[Nov'14]*



36. What are the advantages of PLAs? *[Nov'12]*

- Both AND and OR array can be programmed
- The circuit design is therefore flexible in nature.

37. What is the advantage of PLA over ROM? *[Apr'10]*

- Both AND and OR arrays are programmable.
- Costliest and complex than PAL and PROMs.
- AND arrays can be programmed to get desired minterms.
- Any Boolean functions in SOP form can be implemented using PLA.

38. What is the difference between PROM and EPROM?*[May'13]*

Sl.No.	PROM	EPROM
1	Programmable Read Only Memory is Programmable in nature.	Erasable Programmable Read Only Memory is Erasable in nature.
2	PROM is Non-volatile.	EPROM is Volatile

39. Compare ROM, PROM and EPROM memory devices. *[Nov'12]*

Sl.No.	ROM	PROM	EPROM
1	It is a Read Only Memory	It is a Programmable Read Only Memory	It is an Erasable Programmable Read Only Memory
2	It cannot used to write.	One time program is possible	It can be reprogrammed

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

40. What is FGPA? *[Nov'11, May'14]*

FGPA stands for field programmable gate array, which is the next generation in the programmable logic devices. The word field refers to the ability of the gate arrays to be programmed for a specific function by the end user. The word array indicates a series of columns and rows of gates that can be programmed by the end user.

41. List the configurable elements in the FGPA architecture. *[May'12]*

The FGPA architecture consists of three types of configurable elements:

- A perimeter of input/output blocks (IOBs).
- A core array of configurable logic blocks (CLBs).
- Resources for interconnection.

42. What is static hazards and dynamic hazard? *[Nov'16]*

Static hazard:

- A static hazard is the situation where, when one input variable changes, the output changes momentarily before stabilizing to the correct value.
- There are two types of static hazards:
  - Static-1 Hazard: the output is currently 1 and after the inputs change, the output momentarily changes to 0,1 before settling on 1
  - Static-0 Hazard: the output is currently 0 and after the inputs change, the output momentarily changes to 1,0 before settling on 0

Dynamic hazard:

- A dynamic hazard is the possibility of an output changing more than once as a result of a single input change.
- Dynamic hazards often occur in larger logic circuits where there are different routes to the output (from the input)

43. What are the two types of asynchronuous sequential circuits? *[May'16]*

- Fundamental mode asynchronuous sequential circuits
- Pulse mode asynchronuous sequential circuits

44. State the difference between PROM, PLA and PAL. *[May'16, May'17]*

Sl.No.	PROM	PAL	PLA
1.	PROM AND array is Programmable.	AND array is Programmable and OR array is fixed.	Both AND & OR array are Programmable.
2.	Only Boolean functions in canonical SOP form can be implemented using PROM.	Any Boolean function is standard SOP form can be implemented using PAL.	Any Boolean function is standard SOP form can be implemented.

45. Define address and word. *(Apr'18)*

- In a ROM, each bit combination of the input variable is called on address. Each bit combination that comes out of the output lines is called a word.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

46. Why was PAL developed? *(Apr'18)*

- It is a PAL that was developed to overcome certain disadvantage of PLA such as longer delays due to additional fusible links that result from using two programmable arrays and more circuits complexity.
- The development of a PAL involves research about how storage information in different formats and multiple purpose and it is priority to know how package and transfer process knowledge as corporate assets.

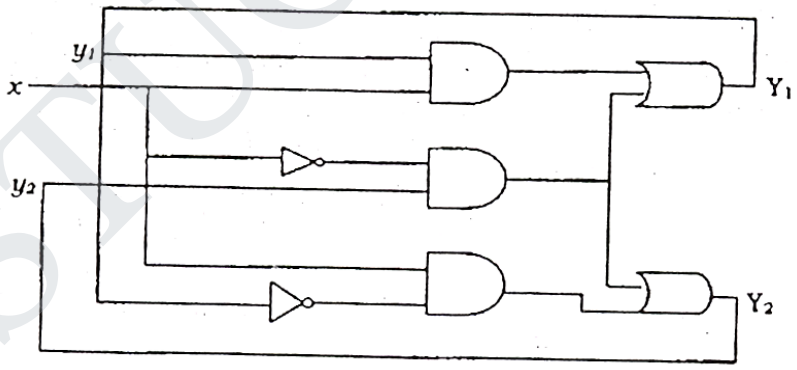
47. What happens to the information stored in a memory location after it has been read and write operation? *(Nov'17)*

- Memory enable and Read/Write signals must be activated after the signals in the address lines are stable to avoid destroying data in other memory words
- The two control signals must stay active for at least 50 ns
- The address and data signals must remain stable for a short time after the control signals are deactivated
- At the completion of the third clock cycle, the CPU can access the memory again with the next T1 cycle

PART B& C [8/16 MARKS]

1. Derive the transition table and primitive flow table for the functional mode asynchronous sequential circuit. *[Nov' 2013]* (OR) **Consider the following asynchronous sequential circuit and draw maps and transition table and state table.** *[May' 2013]* (AND) **Illustrate the analysis procedure of asynchronous sequential circuit with an example.** *[May'13]* (OR) **what are transition table and flow table? Give Suitable examples**. *[May'16]*

Sequential Circuits:



- The analysis of asynchronous sequential circuits consists of obtaining a table or a diagram that describes the sequence of internal states and output has a function of changes in the input variables.

Analysis Procedure:

The procedure for obtaining a transition table for the given circuit diagram is as follows:

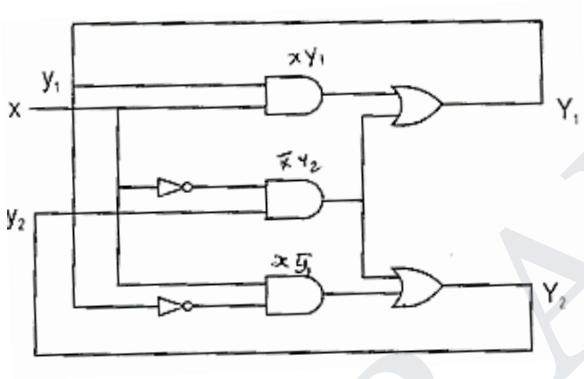
1. Determine all feedback loops in the circuit.
2. Designate the output of each feedback loop with variable Y1 and its corresponding inputs y_1, $y_2 \ldots, y_k$, where k is the number feedback loops in the circuit.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

3. Derive the Boolean functions of all Y's as a function of the external inputs and the y's.
4. Plot each Y function in a map, using y variables for the rows and the external inputs for the columns.
5. Combine all the maps into one table showing the table of $Y=Y_1, Y2 \ldots Y_k$ inside each square.
6. Circle all stable states where Y=y. The resulting map is then the transition table.

2. Derive the transition table and primitive flow table for the fundamental mode Asynchronous sequential circuits



(i) Boolean Expressions: Boolean expressions for the logic diagram are:

$Y_1 = xy_1 + \bar{x}y_2$

$Y_2 = x\bar{y}_1 + \bar{x}y_2$

Where Y_1, Y_2 = Excitation Variables = Next State

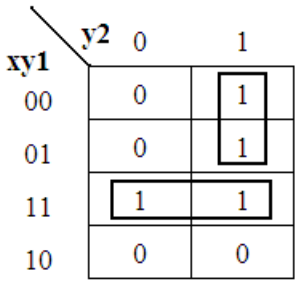
Y_1, Y_2 = Secondary Variables = Present state

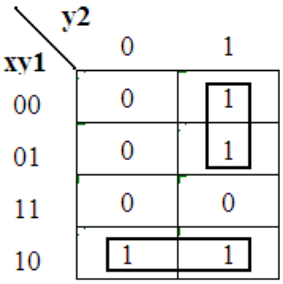
(ii) Next state table:

x	y1	y2	xy1	$\bar{x}$y2	$\bar{x}$y1	$Y_1 = x\,y1 + \bar{x}\,y2$	$Y_2 = \bar{x}\,y1 + \bar{x}\,y2$
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	1
0	1	0	0	0	0	0	0
0	1	1	0	1	0	1	1
1	0	0	0	0	1	0	1
1	0	1	0	0	1	0	1
1	1	0	1	0	0	1	0
1	1	1	1	0	0	1	0

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

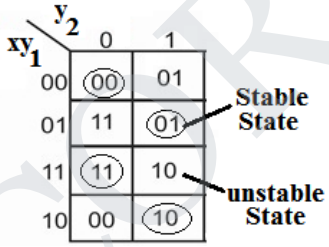




$Y_2 = x\bar{y}_1 + \bar{x}y_2$ $Y_1 = xy_1 + \bar{x}y_2$

(iii) Transition Table:

- The transition table obtained from the maps by combined the binary values in corresponding squares. i.e., $Y=Y_1Y_2$.
- The first bit is obtained from the values of Y_1 and the second bit of Y is obtained from the value of Y_2 in the same square position.
- For a state to be stable, the value of Y must as that $Y=Y_1Y_2$.
- Those entries in the transition table where Y=y are circled to indicate a stable condition.
- An un-circled entry represents an unstable state.



(iv)Primitive flow table

- A flow table is similar to a transition table expect that the internal states are symbolized with letters like a,b,c,S_1, S_2, etc. rather than binary numbers.
- In this example, we assign the following binary values to the states:

(A)	B
C	(B)
(C)	D
A	(D)

a=00 b=01 c=11 d=10

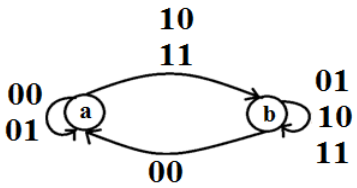
- A primitive flow table is shown in Table; it has only one stable state in each row.

3. Discuss the steps involved in the design of an asynchronous sequential circuit with a suitable example. *[Nov'12]* (AND) **Describe the steps involved in design of asynchronous sequential circuit in detail with an example.** *[Apr'11]* (AND) **Design an asynchronous sequential logic circuit for the state transition diagram shown in figure.** *[Nov'11]*

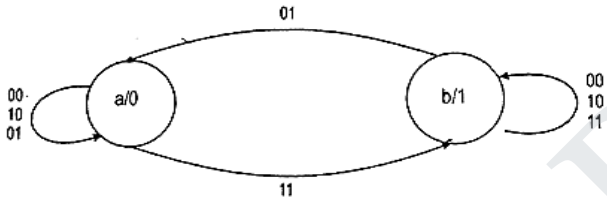
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Design of Fundamental Mode Asynchronous Sequential Circuits



Design of fundamental mode asynchronous sequential circuit with the following behavior. The circuit has two external inputs. A and B and one output Z. The state diagram is shown in figure.



For this design one does not need to obtain a primitive flow table, merger diagram, reduced primitive flow table, transition diagram since Figure cannot be reduced to less than two states.

- For two states, one state variable is required which will be called y. assigning the state code 0 to state 'a' and state code 1 to `b' and output Z=y.
- The K-map for the next state and external outputs can be drawn using the state diagram.
- The logic diagram is shown in above figure.

Present state	AB inputs / Next State				Output Z
	00	01	11	10	
a	a	a	b	a	0
b	b	a	b	b	1

For a⇒ 0, b ⇒ 1

Present state	AB inputs / Next State				Output Z
	00	01	11	10	
0	0	0	1	0	0
1	1	0	1	1	1

K – map:

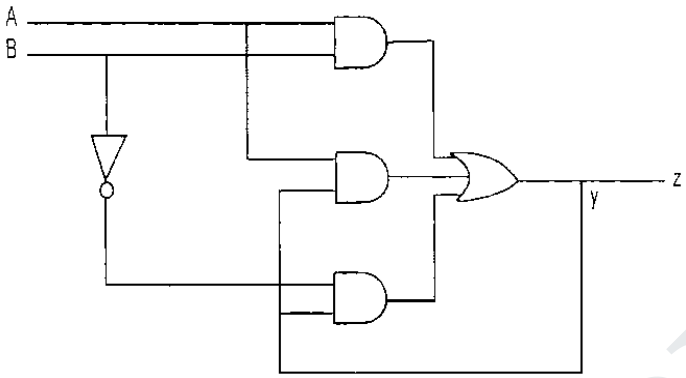
y \ AB	00	01	11	10
0	0	0	1	0
1	1	0	1	1

=> Y = AB + yA + y$\bar{B}$

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Logic Diagram:



4. **Describe the procedure to get state table from excitation table in an asynchronous sequential circuit. How does it differ from asynchronous sequential circuit?** *[Apr'10]* (OR) **How do you get output specification from a flow table in asynchronous sequential circuit operating in fundamental mode?**

- Fundamental mode asynchronous sequential circuit analysis requires careful attention because these utilize unlocked memory and level inputs.
- The procedure to analyze these circuits is as follows:

  Determine the next-secondary state and output equation from given sequential circuits

  - Construct the state table
  - Construct the transition table
  - Construct output map
- The given circuit has two input variables I_1 and I_2 and one output variable Z.
- The circuit has two feedback paths which provide input to gates, creating latching operation necessary to produce a sequential circuit.
- The feedback path also generates the state variables X_0 and X_1.
- The next state for the circuit is determined by both, the state of input variables and the state variables.

Step1: Determine next-secondary state and output equations.

From the sequential circuit we can have next-secondary state and output equations as follows

Step2: Construct state table.

From these next-secondary state and output equations we can construct the state table indicating present-total state, next-total state, stability of next secondary state and the output. The next –secondary state values are found by assigning present-total state values to the Boolean variables in the next- secondary state equation to determine X^+_1 and X^+_0.

For the given input and secondary state if next-secondary state does change then the state is said to be stable.

5. **Obtain the primitive flow table for an asynchronous sequential circuit that has two inputs x,y and 1 output z. The inputs x and y never change or are 1 simultaneously. An**

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

output z=1 is to occur only during the input state xy=01 and then if and only if the input state xy=01 is processed by the input sequence xy=01, 00, 10, 00, 10, 00.

Input sequence	stable state
01	A
00	B
10	C
00	D
10	E

- Stable State A is placed in the first row, second column of the next section of the primitive flow table for the xy=01, assumed that the output z =0.
- When the input state is changed from xy=01 to xy=00, a second row must be added for stable state B. an un-circled B is placed in the first row, first column, and a circled B is placed in the second row, first column.
- The input states xy=10 and xy=11 cannot flow the input state xy=01, since the inputs xy never change or are 1 simultaneously.
- Next in input state of xy=10 result in the circuit entering stable state C this state signifies that the input sequence xy=01, 00, 10has been applied.
- Next an input state xy=00 result on the circuit entering stable state D this state signifies that in the input sequence is xy=01, 00, 10, 00.
- Similarly, stable state E and F are placed in the primitive flow table as shown in table stable state G is introduce with an associated one output.

Present state	Next State				Output (z)			
	Input state (xy)				Input state (xy)			
	00	00	10	11	00	00	11	10
A	B	(A)	-	-	-	0	-	
B	(B)		C	-	0	-	-	-
C	D	-	(C)	-	-	-	0	-
D	(D)		E	-	0	-	-	-
E	F	-	(E)	-	-	-	0	-
F	(F)	G	-	-	-	-	-	-
G		G	-	-	-	1	-	-

- This asynchronous sequential circuit as the following requirements:
  1. The inputs x and y never change simultaneously i.e, only one input variable is allowed to change value at a time
  2. The inputs x and y are never 1 simultaneously.

- These requirements imply that only dashes (-) can appear in the fourth column and other some entries in the next state section the primitive flow as shown in the table.
  - The unstable state A are a entered in the second and fourth rows of the second column of the next section of the primitive flow table since the sequence to be recognized is broken by the input state xy=01 and hence a recognition process must restarted.

STUCOR APP
DOWNLOADED FROM STUCOR APP

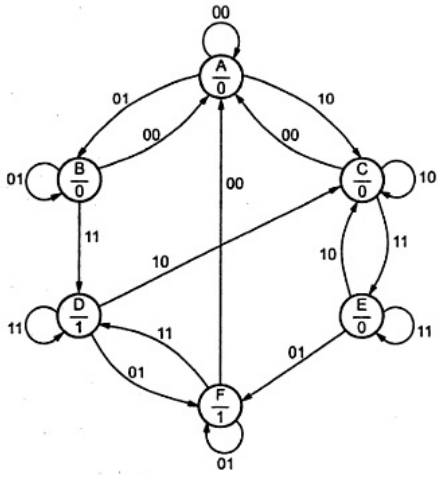
DOWNLOADED FROM STUCOR APP

- The unstable state B is entered in the seventh row, first column, since the input state for the stable state in that row can serve both as first xy=01 input of a new sequence as well as the last input of the recognized sequence.
- Finally two additional stable state H and I must be added to handle the situation that the input state xy=01, which defines the beginning and end of the input sequence to be recognized.
- In the last row an unstable state A indicates the first input state of the sequence to be recognized is applied and an unstable state A indicates the first input state of sequence to be recognized is applied and an unstable state H provides for the network to wait for the beginning of the specified input sequence that is to recognized.
- The complete primitive flow table is shown in table.

Present state	Next State				Output (z)			
	Input state (xy)				Input state (xy)			
	00	00	10	11	00	00	11	10
A	B	(A)	-	-	-	0	-	
B	(B)	A	C	-	0	-	-	-
C	D	-	(C)	-	-	-	0	-
D	(D)	A	E	-	0	-	-	-
E	F	-	(E)	-	-	-	0	-
F	(F)	G	-	-	0	-	-	-
G	B	(G)	-	-	-	1	-	-
H	I	-	(H)	-	-	-	0	-
I	(I)	A	H	-	0	-	-	-

7. Design an asynchronous sequential circuit that has two inputs X_2 and X_1 and one output Z. When $X_1 = 0$, the output Z is 0. The first change in X_2 that occurs while X_1 is 1 will cause output Z to be 1. The output Z will remain 1 until X_1 returns to 0. *[Apr'15]*

State diagram:



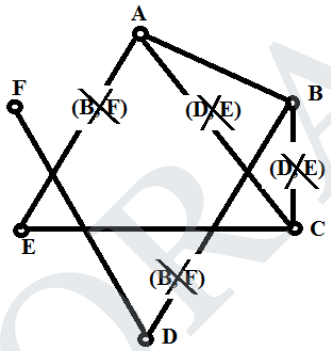
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Primitive flow Table:

Present state	Next state ,Output Z for X_2X_1 Inputs			
	00	01	11	10
A	(A), 0	B , –	– , –	C , –
B	A , –	(B), 0	D , –	– , –
C	A , –	– , –	E , –	(C), 0
D	– , –	F , –	(D), 1	C , –
E	– , –	F , –	(E), 0	C , –
F	A , –	(F), 1	D , –	– , –

Merger Graph:



The merger graph gives the two compatible pairs as a set of maximal compatibles.

(A, B) → S_0, (C, E) → S_1, (D, F) → S_2

Reduced flow table:

Present state	Next state, Output Z for X_2X_1 Inputs			
	00	01	11	10
S_0	(S_0), 0	(S_0), 0	S_2 , –	S_1 , –
S_1	S_0 , –	S_2 , –	(S_1), 0	(S_1), 0
S_2	S_0 , –	(S_2), 1	(S_2), 1	S_1 , –

Assign S_0= 00, S_1= 01 and S_2= 10 then we need one more state S_3 = 11 to prevent critical race during transition of S1 → $S_2$and S_2→ S_1 through S_3.

Flow table:

Present state F_2 F_1	Next state, Output Z for X_2X_1 Inputs			
	00	01	11	10
S_0 → 0 0	(S_0), 0	(S_0), 0	S_2 , –	S_1 , –
S_1 → 0 1	S_0 , –	S_3 , –	(S_1), 0	(S_1), 0
S_2 → 1 0	S_0 , –	(S_2), 1	(S_2), 1	S_3 , –
S_3 → 1 1	– , –	S_2 , –	– , –	S_1 , –

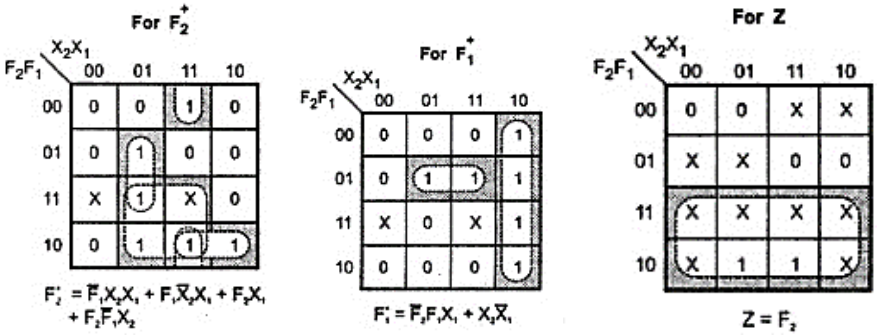
Transition Table:

Present state		Next state, Output Z for X_2X_1 Inputs			
F_2	F_1	00	01	11	10
0	0	00 , 0	00 , 0	10 , –	01 , –
0	1	00 , –	11 , –	01 , 0	01 , 0
1	0	00 , –	10 , 1	10 , 1	11 , –
1	1	– , –	10 , –	– , –	01 , –

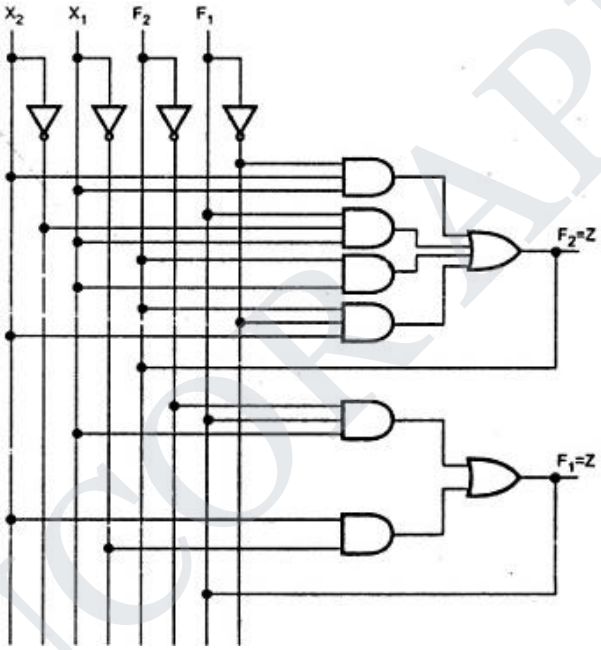
STUCOR APP
DOWNLOADED FROM STUCOR APP

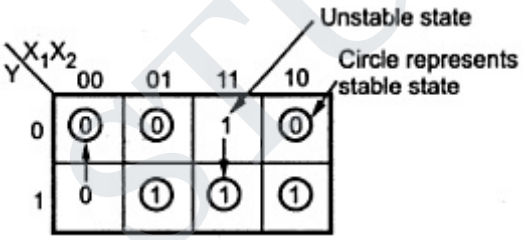
DOWNLOADED FROM STUCOR APP

K-Map:



Logic Diagram:





Y \ X_1X_2	00	01	11	10
0	0	0	–	0
1	–	1	1	1

Output is mapped for all stable states. For unstable states output is mapped unspecified

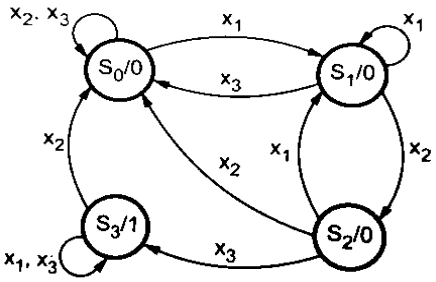
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

8. Design a pulse mode circuit with inputs x_1, x_2, x_3 and output Z as shown in figure.

Step 1: Draw state diagram and state table

Since the output must remain high between input files, a Moore-type circuit is required to realize the circuit. The state diagram and stable table in fig. and table respectively, satisfy the stated requirements.



Present state	Next state			Output Z
	X_1	X_2	X_3	
S_0	S_1	S_0	S_0	0
S_0	S_1	S_2	S_0	0
S_0	S_1	S_0	S_3	0
S_0	S_3	S_0	S_3	1

Step 2: Minimum state table: state table is minimum.

Step 3: Assign states: A state assignment of S_0=00, S_1=01, S_2=11, S_3=10

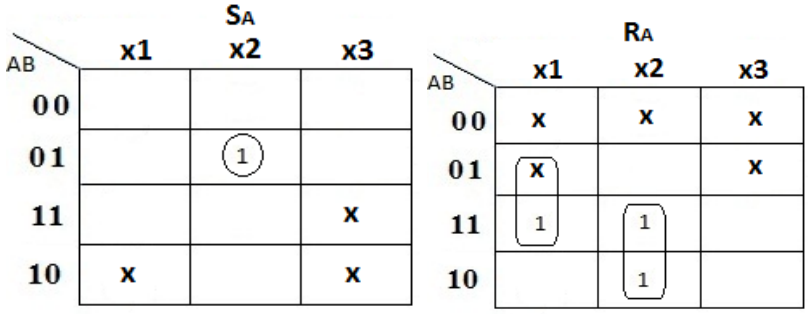
Step 4: Flip-flop to be used:SR

Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	1	1
1	1	x	0

Step 5: Construct excitation table for the circuit

Present state		Next state						Z	Flip-flop inputs											
		X_1		X_2		X_3			S_A			R_A			S_B			R_B		
A	B	A^+	B^+	A^+	B^+	A^+	B^+		X_1	X_2	X_3	X_1	X_2	X_3	X_1	X_2	X_3	X_1	X_2	X_3
0	0	0	1	0	0	0	0	0	0	0	0	X	X	X	1	0	0	0	X	X
0	1	0	1	1	1	0	0	0	0	1	0	X	0	X	X	X	0	0	0	1
1	1	0	1	0	0	1	0	0	0	0	X	1	1	0	X	0	0	0	1	1
1	0	1	0	0	0	1	0	1	X	0	X	0	1	0	0	0	0	X	X	X

Step 6: K-map simplification for SR inputs and Z output



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

$S_A=\bar{A}Bx_2$ $R_A=Bx_1+Ax_2$

S_B

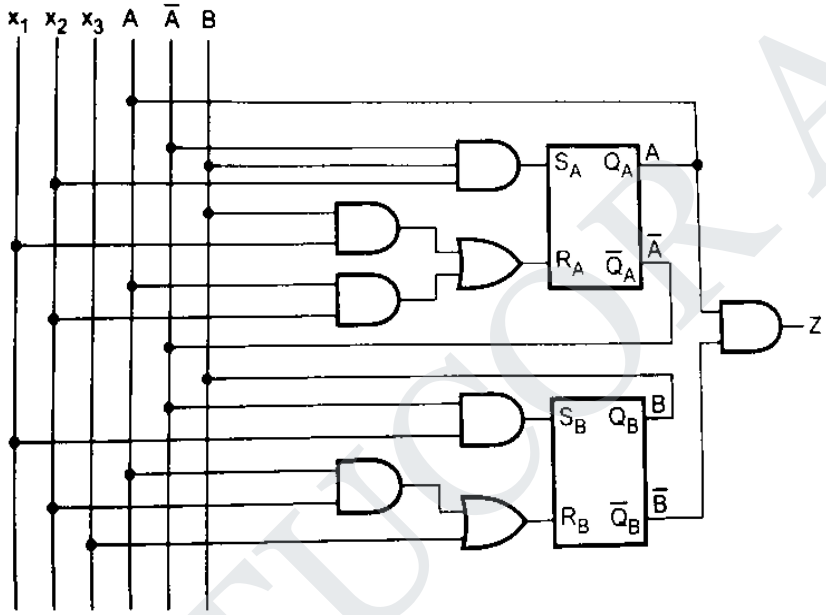
AB	x1	x2	x3
00	1		
01	x	x	
11	x		
10			

R_B

AB	x1	x2	x3
00		x	x
01			1
11		1	1
10	x	x	x

$S_B=\bar{A}x_1$ $R_B=Ax_2+x_3$

Step 7: Draw the logic diagram



9. List and explain the steps used for analyzing an asynchronous sequential circuit. *[May'12]*

Sequential Circuits:

The analysis of asynchronous sequential circuits consists of obtaining a table or a diagram that describes the sequence of internal states and output has a function of changes in the input variables.

Analysis Procedure:

The procedure for obtaining a transition table for the given circuit diagram is as follows:

1. Determine all feedback loops in the circuit.
2. Designate the output of each feedback loop with variable Y1 and its corresponding inputs y_1, $y_2 \ldots, y_k$, where k is the number feedback loops in the circuit.
3. Derive the Boolean functions of all Y's as a function of the external inputs and the y's.
4. Plot each Y function in a map, using y variables for the rows and the external inputs for the columns.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

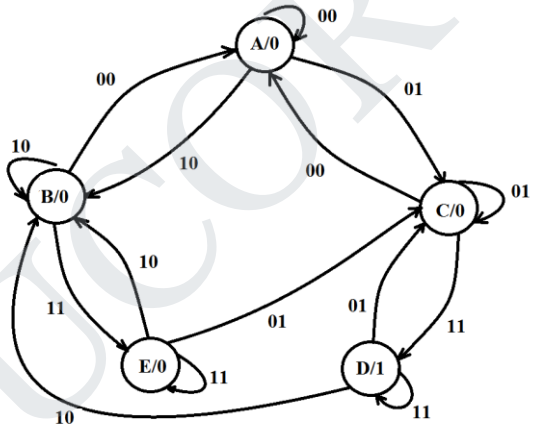
5. Combine all the maps into one table showing the table of $Y=Y_1,Y_2,\ldots..Y_k$ inside each square.
6. Circle all stable states where Y=y. The resulting map is then the transition table.

10. Design an asynchronous sequential circuit that has two inputs X_2 and X_1 and one output Z. When $X_1 = 0$, the output Z is 0. The first change in X_2 that occurs while X_1 is 1 will cause output Z to be 1. The output Z will remain 1 until X_1 returns to 0. *[Nov'15, Nov'17]*

Step 1: Primitive Flow Table

STATES \ X_1 X2	00	01	11	10
A	Ⓐ, 0	C, -	-, -	B, -
B	A, -	-, -	E, -	Ⓑ, 0
C	A, -	Ⓒ, 0	D, -	-, -
D	-, -	C, -	Ⓓ, 1	B, -
E	-, -	C, -	Ⓔ, 0	B, -

Step 2: State Diagram



Step 3: Transition Table

	00	01	11	10
0	0, 0	0, 0	0, 1	1, -
1	0, -	0, -	1, 0	1, 0

Step 4: Function of Y & Z

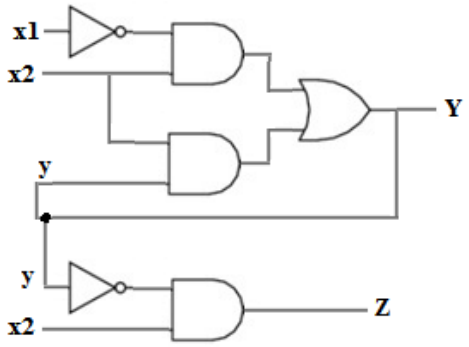
$Y = X_2X_1' + yX_2$

$Z = y'X_2$

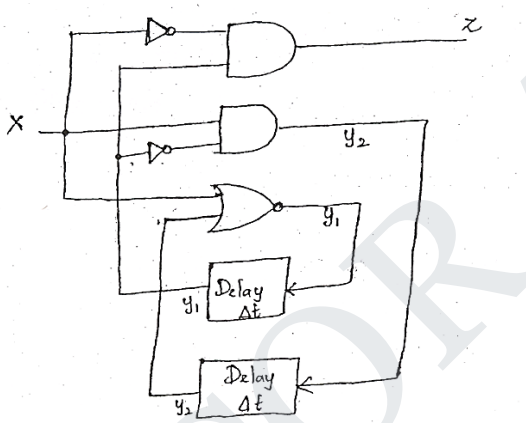
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Step 6: Logical Diagram



11. Derive the flow table for the circuit given in the figure. *[May'12]*



Step 1: The excitation and output equations for the given circuits are:

$Y_1=\bar{X}\bar{Y}_2$

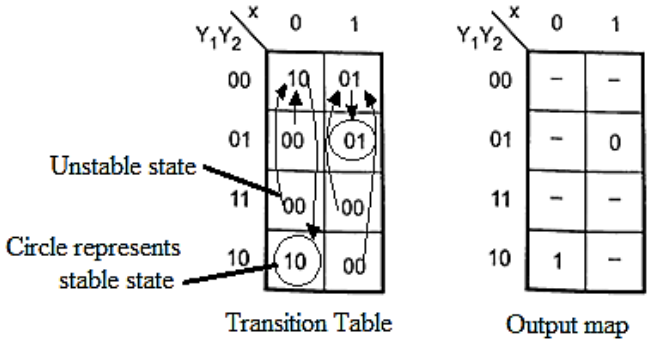
$Y_2=X\bar{Y}_1$

$Z=\bar{X}Y_1$

Present total state			Next total state			Stable total state	Z
NS state		Input	NS state		Input		
Y_1	Y_2	x	Y_1	Y_2	x	Yes/No	
0	0	0	1	0	0	No	0
0	0	1	0	1	1	No	0
0	1	0	0	0	0	No	0
0	1	1	0	1	1	Yes	0
1	0	0	0	0	0	No	1
1	0	1	0	0	1	No	0
1	1	0	1	0	0	Yes	1
1	1	1	0	0	1	No	0

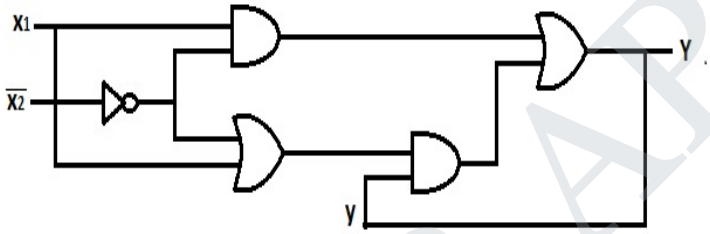
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



12. An Asynchronous sequential circuit is described by $Y = x_1x_2' + (x_1 + x_2')y$; $z = y$. Draw the logic diagram, transition table and outline map. *[May'14]*

Logic Diagram:



Next State table:

y	X1	X2	X1x2'	X1 + x2'	Y	Z
0	0	0	0	1	0	0
0	0	1	0	0	0	0
0	1	0	1	1	1	0
0	1	1	0	1	0	0
1	0	0	0	1	1	1
1	0	1	0	0	0	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1

Maps for Y and Z:

For Y

y \ X1X2	00	01	11	10
0	0	0	0	1
1	1	0	1	1

For Z

y \ X1X2	00	01	11	10
0	0	0	0	0
1	1	1	1	1

Transition Table:

y \ X1X2	00	01	11	10
0	(00)	00	00	(10)
1	11	(01)	(11)	11

Primitive Flow Table:

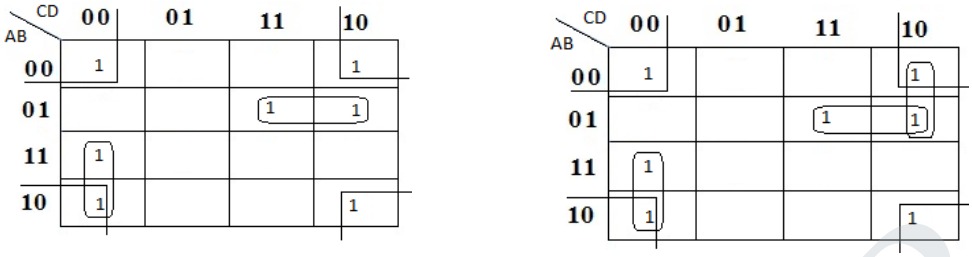
y \ X1X2	a	b	c	d
0	(a)	a	a	(d)
1	c	(b)	(c)	c

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

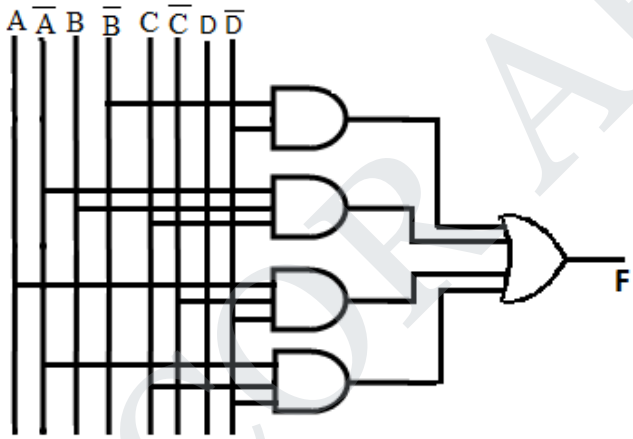
13. Find a circuit that has no static hazards and implements Boolean function F(A,B,C,D)=Σ(0,2,6,7,8,10,12) *[May'14]*

Circuit with hazard: Circuit without hazard:



$$F = \overline{BD} + \overline{A}BC + A\overline{C}\overline{D}F = \overline{BD} + \overline{A}BC + A\overline{C}\overline{D} + \overline{A}C\overline{D}$$

Circuit Diagram:



14. Explain the various types of hazards in sequential circuit design and the methods to eliminate them. Give suitable examples. *[Nov'14] [OR]* **What is hazards ?Explain hazards in digital circuits? .(Nov'17)**

There are three type of hazards occur in sequential circuit.

1) Static hazard
2) Dynamic hazard
3) Essential hazard

Design of Hazard Free Switching Circuits

- To design a hazard- free switching circuits, all adjacent input combinations having same output within some sub cube of the corresponding function.
- In other words, every pair of adjacent 1 cells or 0 cells in the K-map of a switched function should be covered by at least one sub cube.

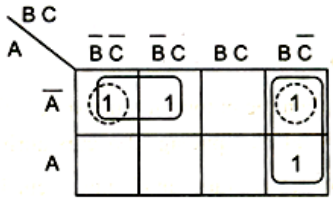
Static Hazards Elimination

- A static hazard can be removed by covering the adjacent cells with a redundant grouping that overlap both groupings. In the K-Map shown in fig.

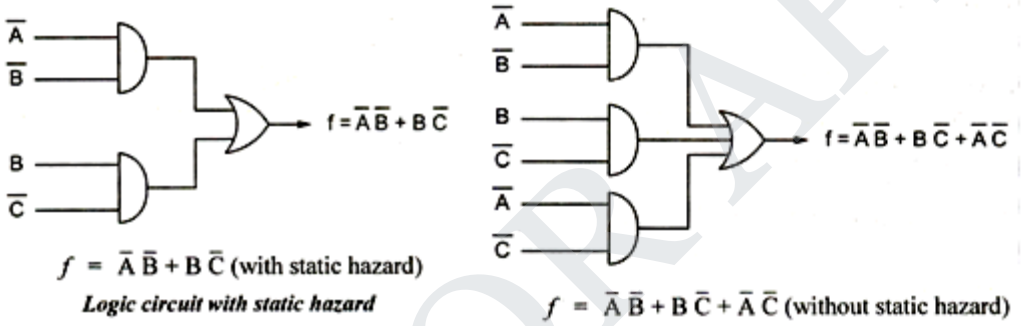
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- Static hazard can be eliminated by covering adjacent cells corresponding to 000 and 010 as shown by dotted sub cube.
- The leads to redundant grouping that overlap both $\bar{A}\bar{B}$ and $\bar{B}\bar{C}$ groupings. The redundant term is $\bar{A}\bar{C}$and the modified circuits is shown in fig.
- Now, the input (ABC) changes from 000 to 010, the output will remain at "1" state (for both 010 and 000 inputs) because of high output at the newly added lower AND gate.



$f = \Sigma(0, 1, 2, 6)$



$f = \bar{A}\bar{B} + B\bar{C}$ (with static hazard)

Logic circuit with static hazard

$f = \bar{A}\bar{B} + B\bar{C} + \bar{A}\bar{C}$ (without static hazard)

Dynamic Hazards Elimination

- Dynamic hazard can also be eliminated I a similar manner as that of static hazard elimination by covering every pair of 1 cells every pair of '0' cells in the K-map by atleast one sub cube.

Essential Hazards Eliminati0n

- Essential hazard can be eliminated by adding redundant gates as in static hazards.
- They can be eliminated by adjusting the amount of delay in the affected path.
- For this, each feedback loop must be designed with extra care to ensure that the delay in the feedback path is long enough compared to the delay other signals that originate from the input terminals.

15. When does oscillation occur in an asynchronous sequential logic circuit? [OR] **Describe with reason, the effect of races in asynchronous sequential circuit design. Explain its types with illustrations. Show the method of race – Free State assignments with examples.** *[Nov'14]* [OR] **Explain cycles and races in asynchronous sequential circuits.** *[May '16]* [OR] **What are static -0 and static -1 hazards? Explain the removal of hazards using hazard covers in k-map?** *(May '16)* **Discuss about hazards in asynchronous sequential circuit and the ways to eliminate them.** *(Nov'17)*

- The asynchronous sequential circuits have oscillation during three problems namely,
  1) Races
  2) Cycles
  3) Hazards

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

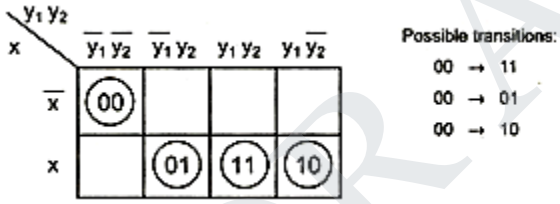
RACES

- When two or more binary state variables change value in response to a change in an input variable.
- Then a RACE condition is exist an asynchronous sequential circuit.

- Races are classified as:
  - i. Critical races and
  - ii. Non-Critical races.

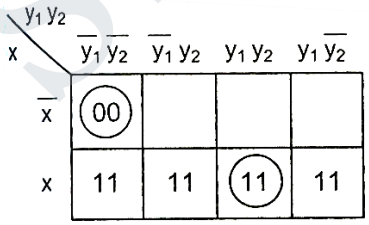
1. Critical Races

- Assume a circuit is in a stable state y_1 y2 x=000 and there is a change in input from 0 to 1. If state variables change simultaneously, the final stable state is y1 y2 x=111.
- If y2 changes to 1 before y1 because of unequal propagation delay, then the circuit goes to the stable state 011 and remain there on the other hand, if y1 changes faster than y2,then the circuit goes to the stable state 101 and remain there.
- Hence, race is critical because the circuit goes to different stable states depending on the order in which the state variable change.



2. Non-Critical Race

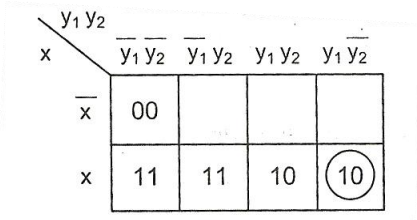
- Figure shows the non-critical races.
- It shows transition tables in which X is an input variables and y1 y2 are the state variables.
- Consider a circuit is in a stable state x y1 y2 =000 and there is a change in input from 0 to 1.
- With this change in the input there are three possibilities that the state variables may change.
- They can either change simultaneously from 00 to 11, or they may change in sequence from 00 to 10 and then 11.
- In all cases, the final stable state is 11, which results in a non- critical race condition.
- In fig. final stable state is x y1 y2 =101.



(a)

Possible transition:

00 → 11
00 → 01 → 11
00 → 10 → 11



(b)

Possible transitions:

00 11 → 10 →
00 → 10
00 → 01 → 11 → 10

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Cycles

- If an input change induces a feedback transition through more than one unstable state, then such a situation is called cycle.

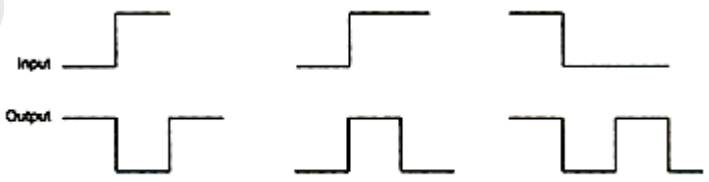
Present State	Next state (x1, x2)			
	00	01	11	10
A	(A)			
B	A			
C	A			
D	A	(D)		

Present State		Next state (x1, x2)			
y1	y2	00	01	11	10
0	0	(00)			
0	1	11			
1	1	10			
1	0	(10)			

- In the table, x1 x2 are input variables, y1 y2 are present state variables and y1 y2 are the next state variables. The circled and uncircled entries represent stable and unstable states respectively.
- Assume that the circuit related to table shown. Is initially in stable state B (i.e., corresponding to input state x1 x2= 01 and secondary state y1 y2 =10).
- Now, when the input x2 changes to 0 (i.e., x1 x2=00 and y1 y2 =10), the next state (y1y2) becomes 11.
- Now, for the present state y1 y2 =11 and x1 x2=00, the next state (y1y2) will be 00 which is aa stable state.
- Thus the circuit goes through a unique sequence of unstable states because of an input change.
- Such a situation is called a cycle.
- When cycle exists in a state table of an asynchronous circuit, care must be taken to ensure that the circuit terminates in a stable state otherwise, the circuit goes from one unstable state to another until a new change in the input occurs.

Hazards

- Hazard is a transient, i.e., spike or glitch that occurs due to unequal path or unequal propagation delays through combinational circuits.
- There are two types of hazards,
  i. Static hazard
  ii. Dynamic hazard as shown in the fig.



(a) Static-1 hazard *(b) Static-0 hazard* *(c) Dynamic hazard*

- Similar to static and dynamic hazards caused by delay in combinational circuits, essential hazards occurs in sequential circuits.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Static Hazards

- Static hazard is a condition which results in a single momentary incorrect output due to change in a single input variable when the output is expected to remain in the same state.

- Two types of static hazards are:
  1. Static- 0 hazard
  2. Static-1 hazard

Static-0 hazard

- When the output is to remain at the value 0 and a momentary 1 output is possible during the transition between the two input states.
- Then the hazard is called a static-0 hazard.

Static-1 Hazard

- If the two input states both produce a 1 output steady state and a momentary 0 output is possible during the transition between the two input states.
- Then the hazards is called a static-1 hazard.

Dynamic Hazard

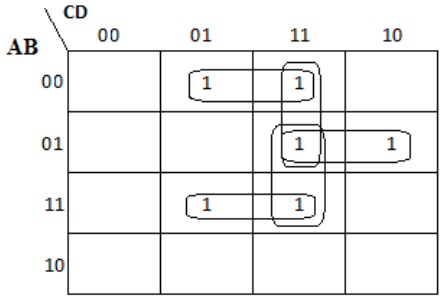
- Dynamic hazards occur when the output of a network is to change between its two logic states, but a momentary false output signal occurs during the transient behavior.
- A dynamic hazard is defined as a transient change occurring three or more times at an output terminal of a logic network when the output is supposed to change only once during a transition between two inputs states differing in the value of one variable.

Essential Hazards

- Essential hazard is a type of hazard that exists only in asynchronous sequential circuits with two or more feedbacks.
- An essential hazard is caused by unequal delays along two or more paths that originate from the same input.
- An excessive delay through an inverter circuits in comparison to the delay associated with the feedback path may cause essential hazard.

16. For the given Boolean function, obtain the hazard – free circuit. F(A,B,C,D) = Σm (1,3,6,7,13,15). *[Apr'15]*

K-map:



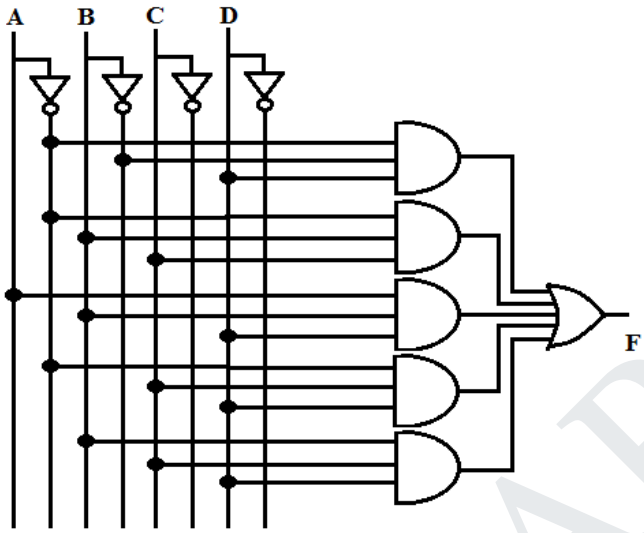
In normal reduction using K-map, we get F = A'B'D + ABC + ABD.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

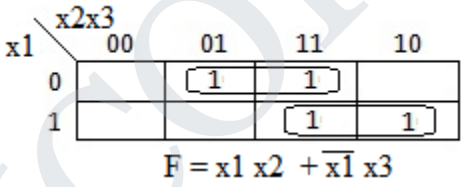
But the hazard – free equation is F = A'B'D + ABC + ABD + A'CD + BCD

Logic Circuit:



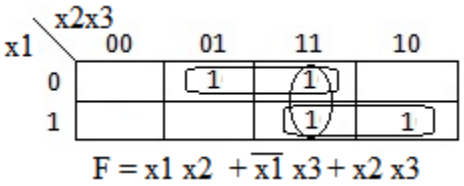
17. Implement the following logic and analyse for the pressure of any hazard F=$x_1x_2 + \overline{x_1}x_3$.If hazard is present briefly explain the type of hazard and design a hazard –free circuit. *(May'17)*

K – Map:



$F = x1\ x2 + \overline{x1}\ x3$

- In the K-Map shown in fig. static hazard can be eliminated by covering adjacent cells corresponding to 011 and 111 as shown by dotted sub cube.
- The leads to redundant grouping that overlap both x1 x2 and $\overline{x1}$ x3 groupings.
- The redundant term is x2 x3 and the modified circuits is shown in fig.



$F = x1\ x2 + \overline{x1}\ x3 + x2\ x3$

18. When do you get the critical and non-critical races? How will you obtain race free condition? *[Apr'10]*

Race free condition:

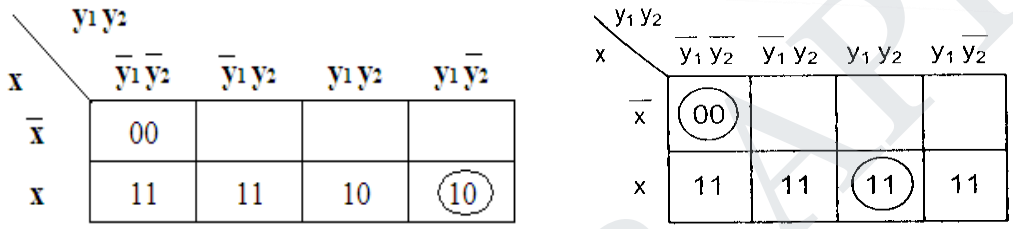
- The state assignment step in asynchronous circuits is essentially the same as it is for synchronous circuit, except for one difference in synchronous circuit, the state assignment are made with the objective of circuit reduction.
- In asynchronous circuit, the objective of the state assignment is to avoid critical races.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

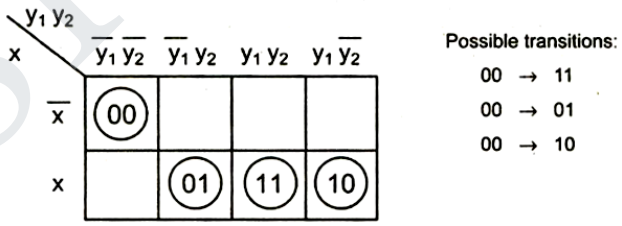
Non-Critical Races

- It shows transition tables in which X is a input variable and y_1 and y_2 are the state variables.
- Consider a circuit is in a stable state y_1 y_2 x=000 and there is a change in input from 0 to 1.
- With this change in the input there are three possibilities that the state variables may change.
- They can either change simultaneously from 00 to 11, or they may change in sequence from 00 to 01 and then to 11, or they may change in sequence from 00 to 10 and then to 11.
- In all case, the final stable state is 11, which results in a non-critical race condition.
- Final stable state is y_1 y_2 x=101.



Critical Race

- Consider a circuit is in a stable state y_1 y_2 x=000 and there is a change in input from 0 to 1.
- If state variables change simultaneously, the final stable state is y_1 y_2 x=111.
- If y_2 changes to 1 before y_1 because of unequal propagation delay, then the circuit goes to the stable state 011 and remains there.
- On the other hand, if y_1 changes faster than y_2, then the circuit goes to stable state 010 and remain there.
- Hence, the race is critical because the circuit goes to different stable state depending on the order in which the state Variable change.



19. Design an asynchronous BCD counter. *[Apr'11, May'14]*

Step 1: Determine the number of flip-flops needed. The BCD counter goes through states 0-9, i.e. total 10 states. Thus, N=10 and for 2n>N, we need n=4, i.e. 4 flip flops required.

Step 2: Type of flip –flops to be used: JK

Step 3: Write the truth table for the counter.

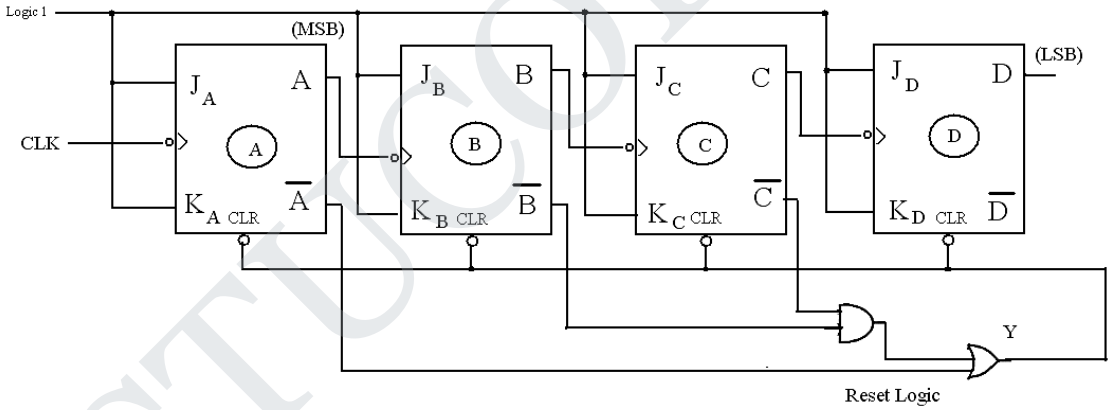
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

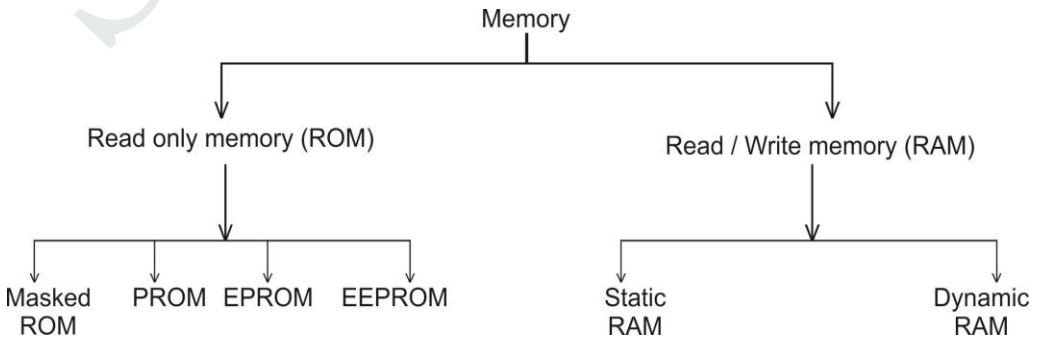
Step 4: Derive reset logic

CLK	A	B	C	D	Output of reset logic Y
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1
-	1	0	1	0	0
-	1	0	1	1	0
-	1	1	0	0	0
-	1	1	0	1	0
-	1	1	1	0	0
-	1	1	1	1	0

Step 5: Draw logic diagram.



20. Describe the characteristics of all types of memories. *[Apr'11]* (AND) **Describe the different types of memories.** *[Apr'10, May'14]*



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Read only memory (ROM):

- We cannot read from or write into memory. It is a read only memory we cannot write data in this memory.
- It is a non-volatile memory (i.e.) it can hold data even if power is turned off.

Read / write memory (RMA):

- Random access memory it is a volatile memory it cannot hold data when the power is turned off.
- We can read from or write into the RAM, so it is called read / write memory.

Masked RAM:

- In integrated circuits, a their metallized layer connects the gates of some transistors to the row select lines the gate connections of MOS transistors depend on the data to be stored in the ROM.
- Once the pattern mask is it is possible to make thousands of such ROMS.
- Such ROMs are called Mask programmed ROMs.

PROM:

- It is a programmable read only memory.
- PROMs are programmed by used to provide the programming facility, each address select and data line intersection has its own fused MOSFET of transistor.
- The PROM are one time programmable. Once programmed the information stored is permanent.

EPROM:

- Erasable programmable read only memory EPROM use MOS circuitry.
- They store 1_s and 0_s as a pocket of charge in a buried layer of the 1c chip.
- EPROMS can be programmed by the user with a special EPROM programmer we can lease the stored data in the EPROM by exposing the chip to ultra violet light through its Quartz window.

Static RAM:

- Memories that consist of circuits capable of retaining their state as long as power is applied are known as static memories.
- These are random access memory (RMA) and hence combine called static RMA memories.

Dynamic RAM:

- Dynamic RAM stores the data as a charge on the capacitor dynamic RAM contains more memory cell as compared to static RAM per unit area.

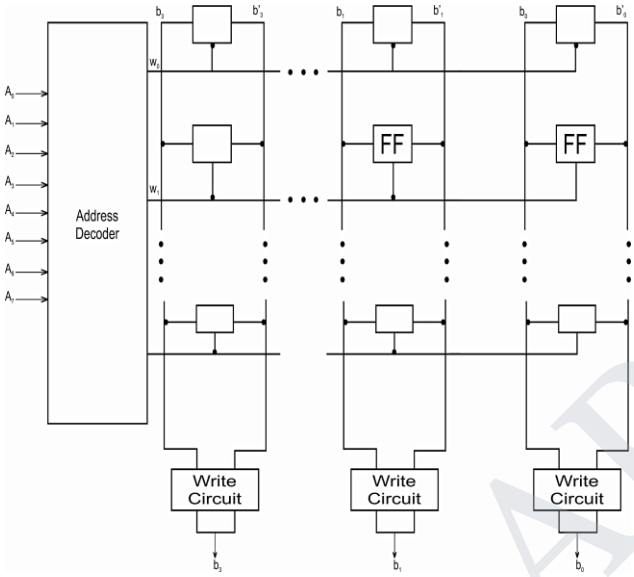
21. Explain the basic structure of a 256x4 static RAM with neat diagram? [AND] **Write a note on dynamic RAM cell?** [OR] **Explain the operation of bipolar RAM cell with suitable diagrams.** *(Apr'18)*

- Unlike ROM, we can read from write into the RAM, so it is often called read / Write memory.
- The numerical and character data that are to be processed by the computer change frequently.
- These data must be stored in type of memory from which they can be read by the microprocessor, modified through processing and written back got storage.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- For this reason they are stored in RAM instead of ROM But it is a volatile memory i.e. it cannot hold data when power is turned off



(b) Compare static RAMs and dynamic RAMs?

Static RAM	Dynamic RAM
Static RAM contains less memory cells per unit area	Dynamic RAM contains more memory cells as compared to static RMA per unit area
Its access time is les hence faster memories	Its access time is greater than static RAMs.
Static RAM contains of number of Flip-Flops-Each flip-flips stores one-bit	Dynamic RAM stores the data as a charge on the capacitor it consists of MOSFET & the capacitor for each cell.
Refreshing circuitry is not required	Refreshing circuitry is required to maintain the charge on the capacitor after every few milliseconds. Extra hardware is required to control refreshing. This makes system design complicated.
Cost is more	Cost is less

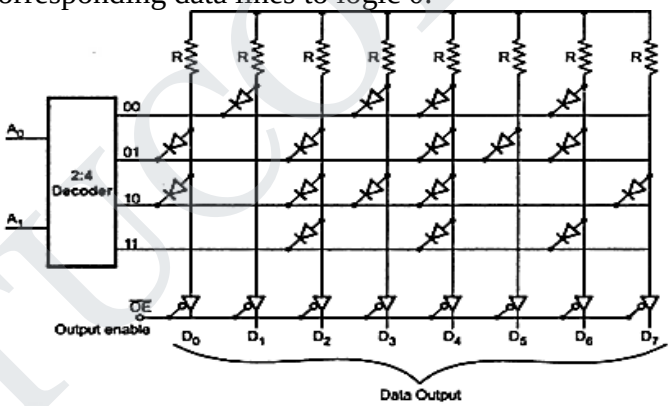
22. Write notes on ROM and its types. *[May'12]* [OR] **Elaborate the concept of PROM, EPROM, EEPROM in detail.** *(Apr'18)*

- It is a read only memory. We can't write data in this memory.
- It is a non-volatile memory i.e. it can hold data even if power is turned off.
- Generally, ROM is used to store the binary codes for the sequence of instruction you want the computer to carry out and data such as look up tables.
- This is because this type of information does not change.
- It is important to note that although we give the name RAM to static and dynamic read/write memory devices that does not mean that the ROMs that we are using are also not random access devices.
- In fact, most ROMs are accessed randomly with unique addresses.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- There four types of ROM: Masked ROM, PROM, EPROM and EEPROM or E^2PROM.
- First we will see simple PROM.
- It is a very simple four byte diode ROM. Diode ROM consists of only diodes and decoder.
- The address lines A_0 and A_1 are decoded by 2:4 decoder used to select one of the four rows.
- As decoder output is active low, it places logic 0 on the selected row.
- Each output data line goes to logic 0 if a diode connects the output data column to the selected row.
- Data is available on the output data lines only when enable (OE) signal is low.
- Now days ROMs use MOS technology instead of diode.
- The nibble (half-byte) ROM using MOS transistors.
- Here, diodes and pull up resistor are replaced by MOS transistors.
- The address on the address lines (A_0 and A_1) is decoded by 2:4 decoder.
- Decoder selects one of the four rows making it logic 0.
- The inverter connected at the output of decoder inverts the state of selected row (i.e. logic 1).
- Therefore, each output data line goes to logic 0 if a gate of MOS transistor is connected to row select lines.
- When gate of the MOS transistor is connected to the selected row, MOS transistor is turned on.
- This pulls the corresponding data lines to logic 0.



Simple four byte diode ROM

- In integrated circuits, a thin metalized layer connects the gates of some transistor to the row select lines.
- The gate connection of MOS transistor depends on the data to be stored in the ROM. Therefore, according to the user truth table, manufacturer can deposit thin layer of metal to connect gates of the transistor.
- Once the pattern/mask is decided, it is possible to make thousands of such ROMs.
- Such ROMs are called mask-programmed ROMs.
- Masked ROMs are used in the microprocessor based toys, TV games, Home computers and other such high volume consumer products.

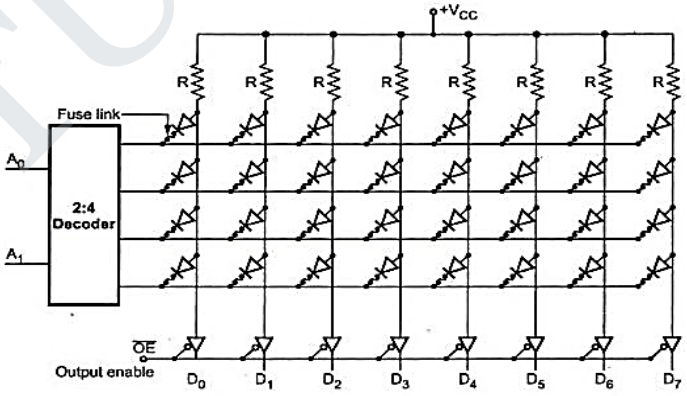
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Address in binary	Binary Data								Data in Hex
	D0	D1	D2	D3	D4	D5	D6	D7	
00	1	0	1	0	0	1	0	1	A5
01	0	1	0	1	0	0	0	1	51
10	0	1	0	0	0	1	1	0	46

Programmable Read Only Memory (PROM)

- PROMs are programmed by user.
- To provide the programming facility, each address select and data line intersection has its own fused MOSFET or transistor.
- When the fuse is intact, the memory cell is configured as logic 1 and when fuse is blown (open circuit), the memory cell is logical 0.
- Logical 0s are programmed by selecting appropriate select line and then driving the vertical data line with a pulse of high current.
- The PROM fused MOSFET memory cell.
- The four byte PROM, it has diode in every bit position; therefore the output is initially all 0s.
- Each diode however has a fusible linking series with it.
- By addressing and applying proper current pulse at the corresponding output, we can blow out the fuse, storing logic1 at the bit position.
- The fuses use material like nichrome and polycrystalline.
- For blowing the fuse it is necessary to pass around 20 to 50 mA of current for period 5 to 20 µs.
- The user can program PROMs with special PROM programmer.
- The PROM programmer selectively burns the fuses according to the bit pattern to be stored.
- This process is also known as burning of PROM; The PROMs are one time programmable.
- Once programmed, the information stored is permanent.



Four Byte PROM

EPROM (Erasable Programmable Read Only Memory)

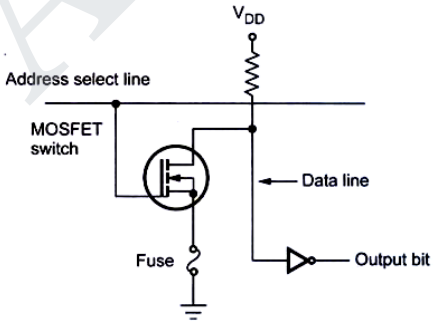
- Erasable Programmable ROMs use MOS circuitry.
- They store 1s and 0s as a packet of charge in a buried layer of the IC chip.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- EPROMs can be programmed by the user with a special EPROM programmer.
- The important point is that we can erase the stored data in the EPROMs by exposing the chip to ultraviolet light through its quartz window for 15 to 20 minutes.
- It is not possible to erase selective information, When erased the entire information is lost. The chip can be reprogrammed.
- This memory is ideally suitable for product development, experimental projects and college laboratories, since the chip can be reused many times, over.
- We can observe that the input and output LOW and HIGH voltage levels for basic ECL family are not same, it has 0.6 V difference.
- This problem cannot be solved by connecting diode in series with output to lower its voltage by 0.6 V because if we does this, it results poor fan-out.
- Another problem occurs when output is HIGH and it drives another ECL input.
- This HIGH output has to drive base current of another ECL, input, resulting drop across R_1 or R_2, and reducing the output voltage. These problems of basic ECL are solved by 10K ECL family.

EEPROM:



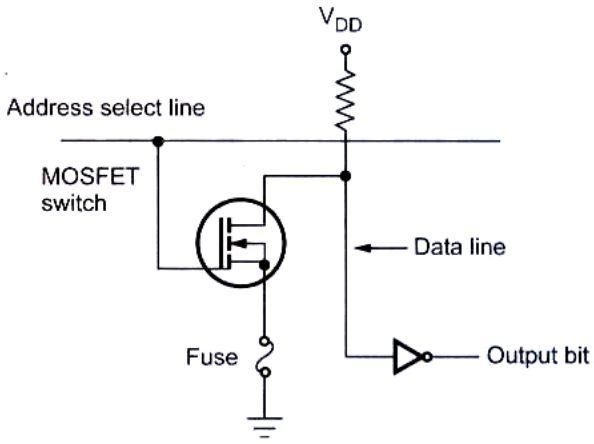
- EEPROM (also E^2PROM) stands for *E*lectrically *E*rasable *P*rogrammable *R*ead-*O*nly *M*emory and is a type of non-volatile memory used in computers, integrated in microcontrollers for smart cards and remote keyless system, and other electronic devices to store relatively small amounts of data but allowing individual bytes to be erased and reprogrammed.
- EEPROMs are organized as arrays of floating-gate transistors. EEPROMs can be programmed and erased in-circuit, by applying special programming signals.
- Originally, EEPROMs were limited to single byte operations which made them slower, but modern EEPROMs allow multi-byte page operations.
- It also has a limited life for erasing and reprogramming, now reaching a million operations in modern EEPROMs. In an EEPROM that is frequently reprogrammed while the computer is in use, the life of the EEPROM is an important design consideration.

23. Elaborate the single fused PROM cell with clear sketch?

- PROMS are programmed by user.
- To provide the programming facility, each address select and data line intersection has its own fused MOSFET or Transistor when the fuse is intact, the memory cell is configured as a logic 1 and when fuse is blown (open circuit) the memory cell is logical 0.
- Logical OS are programmed by selecting the appropriate select line and then driving the vertical data line with a pulse of high current shows a PROM fused MOSFET memory cell.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



24. Explain memory decoding.

- We have seen that in the organization of memory there is a need for decoding circuit to select the memory word specified by the input address. In this section we study the operation of the decoder. There are 16 words of 8-bits each. A memory with 16 words needs four address lines.
- The four address inputs go through 4x16 decoder to select one of the sixteen words. The decoder is enabled with a memory enable input. When the memory enable is 0, all output of the decoder is 0 and none of the memory words are selected. With the memory select at 1, one of the sixteen words is selected, dictated by the value in the four address lines.
- Once a word has been selected, the read / write input determine the operation. During write operation, the data available in the input lines are transferred into the eight memory cells of the selected word. The memory cells that are not selected are disabled and their previous binary values remain unchanged.

25. Explain EPROM and PLA? (OR) **Write briefly about the programmable logic array and EPROM.** *[Nov'11]* (OR) **Describe the working of EPROM. List the application EPROM.**

EPROM – Erasable Programmable Read only memory

- Erasable programmable ROM's use MOS circuitry.
- They stores 1^s and 0^s as a packet EPROM can be programmed by the user with a special EPROM programmer.
- The important point is that we can erase the stored data in the EPROM by exposing the chip to ultraviolet light through its quartz window for 15 to 20 minutes.
- It is not possible to erase selective information, when erased the entire information is lost.
- The chip can be reprogrammed.
- This memory is ideally suitable for product development, experimental project and college laboratories, since this chip can be reduced many times, over EPROM programming.
- When erased each cell in the EPRROM contains 1 Data is introduced by selectively programming 0^s into the desired bit locations although only 0^s will be programmed, both 1^s and 0^s can be presented in the data.
- During programming address and data are applied to address and data pins of the EPRO.
- When the address and data are stable, program pulse is applied to the program input of the EPROM.

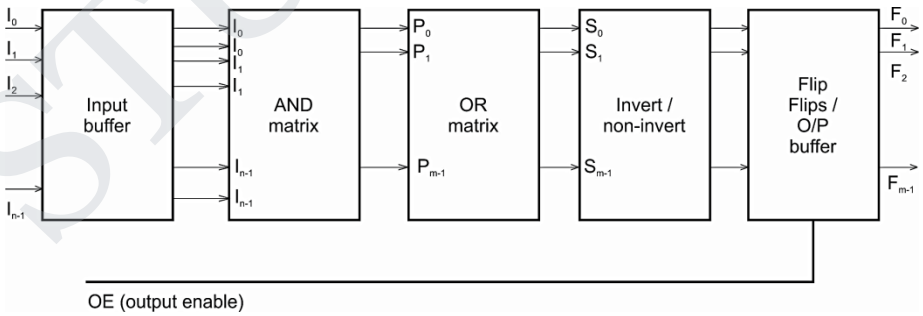
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- The program pulse duration is around 50 ms and its amplitude depends on EPROM IC.
- It is typically 11.5v to 25v in EPROM, it is possible to program any location at any time-either individually, sequentially or at random.

PLA (Programmable logic array):

- The combinational circuit does not use all the minterms every time.
- Occasionally, they have don't card conditions.
- Don't card condition when implemented with PROM becomes an address input that will become an address input that will never occur the result is that not all the bit patterns available in the PROM are used, which may be considered a waste of available equivalent.
- For cases where the number of don't care conditions is excessive, it is more economical to use a second type of LS1 component called a programmable logic array.
- A PLA is similar to a PROM in concept however it does not provide full decoding of the variable and does not generates all the minterms as in the PROM.
- The PAL replaces decoder by group of AND gates, each of which can be programmed to generate a product term of the input variable.
- In PLA both AND and OR gates have fuses at the input diagram of PLA.
- It consists of n-inputs output buffer with m output, m product terms, m sum term, input and output buffers.
- The product terms constitute a group of m AND gates and the sum terms constitute a group of m OR gates, called or matrix.
- Fuses are inserted between all n-inputs and their complement value to each of the AND gates.
- Fuse are also provided between the outputs of the AND gates and the input of the OR gates.
- Their set of fuses in the output inverters allows the output function to be generated either in the AND – OT form or in the AND – OR INVERT form when inverter is by passed by link we get AND – OR implementation.
- To get AND – OR-INVETER implementation inverter link has to be disconnected.



- Erasable Programmable ROMs use MOS circuitry.
- They store 1s and 0s as a packet of charge in a buried layer of the IC chip.
- EPROMs can be programmed by the user with a special EPROM programmer.
- The important point is that we can erase the stored data in the EPROMs by exposing the chip to ultraviolet light through its quartz window for 15 to 20 minutes.
    - It is not possible to erase selective information, When erased the entire information is lost.

STUCOR APP
DOWNLOADED FROM STUCOR APP

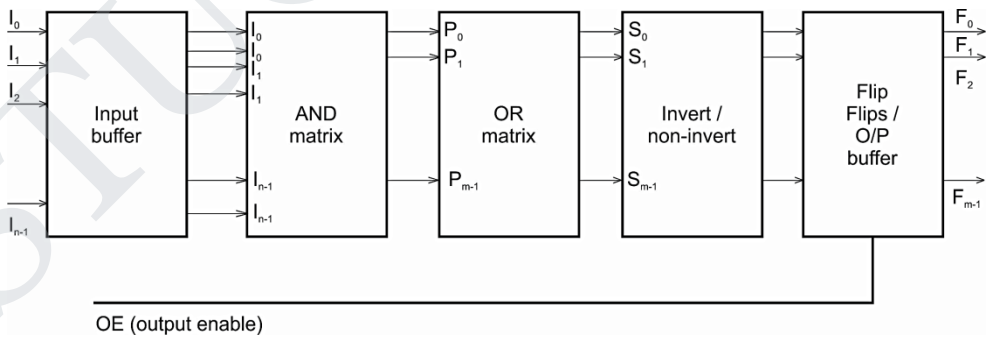
DOWNLOADED FROM STUCOR APP

- The chip can be reprogrammed.
- This memory is ideally suitable for product development, experimental projects and college laboratories, since the chip can be reused many times, over.

26. Illustrate the basic principles of PLA and FPGA. *[May'13]* (OR) **Write short note on field programmable gate Array (FPGA).** (OR) **Describe the concept and working of FPGA.** *[May'12]* (OR) **What do you understand by FPGA? Explain the operation and applications?** *[Nov'10]* (OR) **Write short notes on PLA and PAL.** *(Nov'17)*

PLA (Programmable Logic Array):

- The combinational circuit does not use all the minterms every time.
- Occasionally, they have don't card conditions.
- Don't card condition when implemented with PROM becomes an address input that will become an address input that will never occur the result is that not all the bit patterns available in the PROM are used, which may be considered a waste of available equivalent.
- For cases where the number of don't care conditions is excessive, it is more economical to use a second type of LS1 component called a programmable logic array.
- A PLA is similar to a PROM in concept however it does not provide full decoding of the variable and does not generates all the minterms as in the PROM.
- The PAL replaces decoder by group of AND gates, each of which can be programmed to generate a product term of the input variable.
- In PLA both AND and OR gates have fuses at the input diagram of PLA.
- It consists of n-inputs output buffer with m output, m product terms, m sum term, input and output buffers.
- The product terms constitute a group of m AND gates and the sum terms constitute a group of m OR gates, called or matrix.



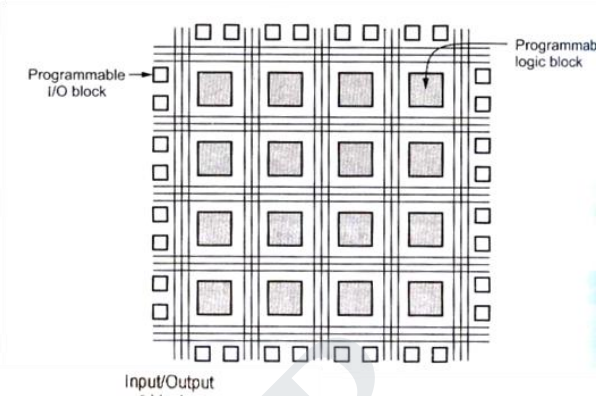
- Fuses are inserted between all n-inputs and their complement value to each of the AND gates.
- Fuse are also provided between the outputs of the AND gates and the input of the OR gates.
- Their set of fuses in the output inverters allows the output function to be generated either in the AND – OT form or in the AND – OR INVERT form when inverter is by passed by link we get AND – OR implementation.
- To get AND – OR-INVETER implementation inverter link has to be disconnected.

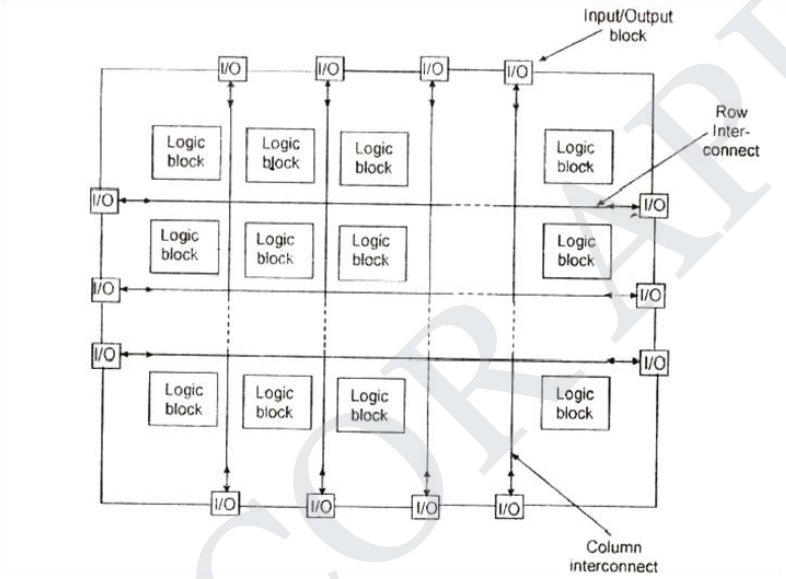
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Programmable Array Logic (PALs)

- PLA is a device with a programmable AND array and programmable OR array.
- PAL programmable array logic is a programmable logic devices with a fixed OR array and a programmable AND array.
- The PAL is easier to program, but it is not flexible as the PLA.





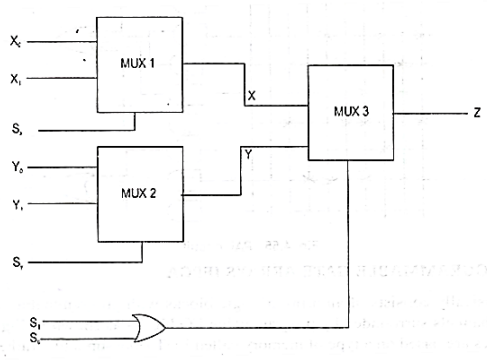
FGPA

- Field programmable date arrays provide the next generation I the programmable logic devices.
- The word field in the name refers to the ability of the gate array to be programmed for a specific function by the user instead of by the manufacture of the device.
- The word array is used of gates that can be programmed by the end user.
- As compared to standard gate array the field programmable gate arrays are larger devices.
- The basic cell structure for FPGA is somewhat complicated then the basic cell structure of standard gate array.
  The programmable logic blocks of FPGAS are called configurable logic blocks (CLBs).
- The FPGA architecture consists of three types of configurable elements – a perimeter of input / output blocks (IOBs), a core array of configurable logic blocks and resources for interconnection.
- The IOBs provide a programmable interface between the internal array of logic blocks and devices external package pins CLBs perform user – specified logic function and the interconnect resource carry signals among the blocks.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- A configuration program stored in internal static memory cells determines the logic functions and the interconnections.



- The configuration data is loaded into the device during power up reprogramming functions.
- FPGA device are customized by loading configuration data into internal memory cells.
- The FPGA device can either actively read its configuration data out of an external serial or bytes – wide parallel PROM (master modes) or the configuration data can be written to the FPGA device (slave and peripheral modes).

Application of FPGA

FPGA have gained rapid acceptance and growth over the past decade because they can be applied to a very wide range of applications. A list of typical applications includes:

- They are used to build random logic, integrating multiple SPLDs, device controllers, communication encoding and filtering, small to medium sized system with SRAM blocks, and many more.
- They are used to build higher level embedded functions such as adders and multipliers.
- FPGAs being parallel processing devices find use in application like brute force attacks used in breaking cryptographic algorithms, in convolutions and FFT computations.
- Other interesting application of FPGAs is prototyping of design later to be implemented in gate arrays.
- They are used in medical imaging, reconfigurable computing, speech recognition, cryptography and bioinformatics and so on.

27. Discuss about the programmable logic devices.

Programmable Logic Devices (PLDs)

- PLD is a Programmable Logic Devices consists of a large array of AND gates and OR gates that can be programmed to achieve specific logic functions.
- The main advantage of PLD approach is that PLDs can be easily configurable by the individual users for specific applications.
- PLDs can be reprogrammed in few seconds and hence gives more flexibility to experiment with designs.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- Reprogramming feature of PLDs also make it possible accept changes /modifications in the previous design circuits.
- According to architecture and flexibility in programming PLDs are classified as
  - PROMs : Programmable Read Only Memory
  - PLAs : Programmable Logic Arrays
  - PAL : Programmable Array Logic
  - FGPAs : Field Programmable Gate Arrays
  - CPLDs : Complex Programmable Logic Devices

Read only memory (ROM)

- We cannot read from or write into memory.
- It is a read only memory we cannot write data in this memory.
- It is a non-volatile memory (i.e.) it can hold data even if power is turned off.

PROMs

- It is a programmable read only memory.
- PROMs are programmed by used to provide the programming facility, each address select and data line intersection has its own fused MOSFET dry transistor.
- The PROM are one time programmable.
- Once programmed the information stored is permanent.

PLAs

- PLA stands for Programmable Logic Array, which is a LSI component.
- In PLA, both AND and OR gates have fuses at the input, therefore in PLA both AND and OR gates are programmable.
- The output from OR gates go through fuses as inputs to output inverters so that the final output can be programmed as either AND-OR or AND-OR-INVERT.

Programmable Array Logic (PALs)

- PLA is a device with a programmable AND array and programmable OR array.
- PAL programmable array logic is a programmable logic devices with a fixed OR array and a programmable AND array.
- The PAL is easier to program, but it is not flexible as the PLA.

FGPAs

- FGPA stands for field programmable gate array, which is the next generation in the programmable logic devices.
- The word field refers to the ability of the gate arrays to be programmed for a specific function by the end user.
- The word array indicates a series of columns and rows of gates that can be programmed by the end user.

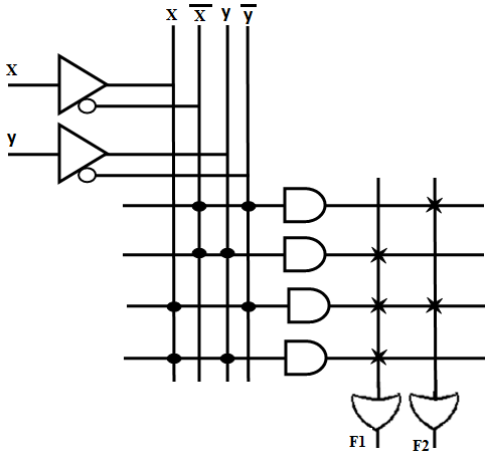
28. Design ROM for the following functions.$F_1 = \sum(1,2,3)$; $F_2 = \sum(0,2)$ *[Apr'11]*

$F_1 (x, y) = \Sigma (1, 2, 3)$ and $F_2 (x, y) = \Sigma (0, 2)$

Inputs		Outputs	
x	**y**	**F1**	**F2**
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	0

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



29. A combinational circuit is defined as the functions.F1 = AB'C+AB'C+ABC and F2 = A'BC+AB'C+ABC. Implement the digital circuit with a PLA having 3 inputs, 3 product terms and 2 outputs.

Let us simplify the function

- There are three distinct product terms: AC, BC and AB' and two and sum terms. The PLA program table consists of three Columns specifying product terms input and outputs.
- The first column gives the list of product term numerically.
- The second column specifies the required path between the input and AND gates.
- The third column specifies the required path b/w the AND gates and OR gates.
- In output variable, we write a 'T' for 'true' if the o/p inverter is to be bypassed, and 'C' for 'complement' if the function is to be a complement with o/p inverter. The product terms listed on the left of first column are not the part of PLA program table they are included for reference only.

PLA table:

Product Term	Input			Output		
	A	B	C	F_1	F_2	
AC	1	-	1	1	1	
BC	-	1	1	1	-	
AB'	1	0	-	-	1	
				T	T	T/C

30. Implement the following two Boolean functions with a PLA?

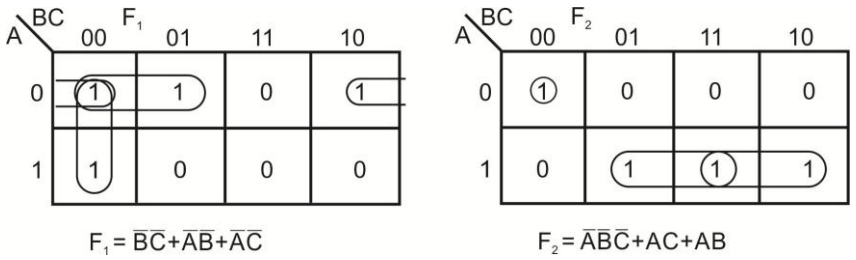
F_1 (A,B,C) = $\sum$ (0,1,2,4)

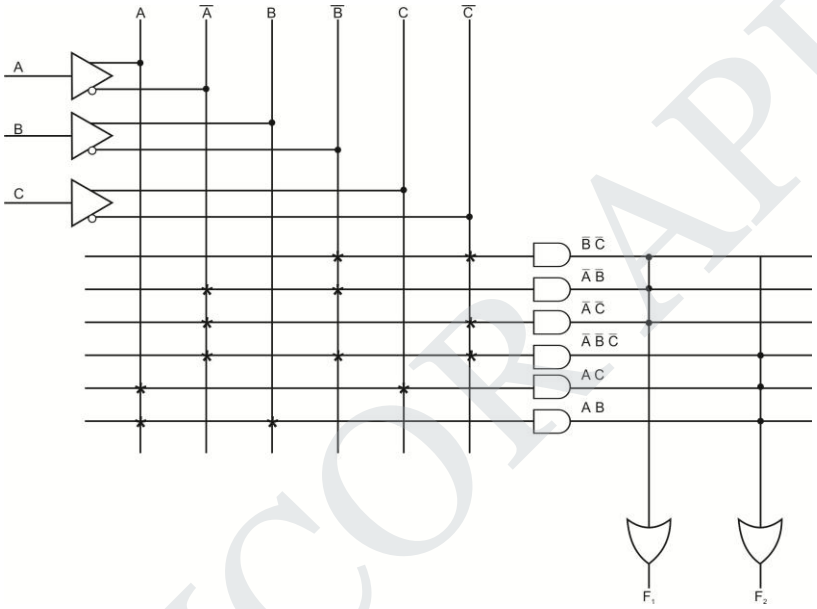
F_2 (A,B,C) = $\sum$ (0,5,6,7)*[Apr'11]*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

K – Map simplification





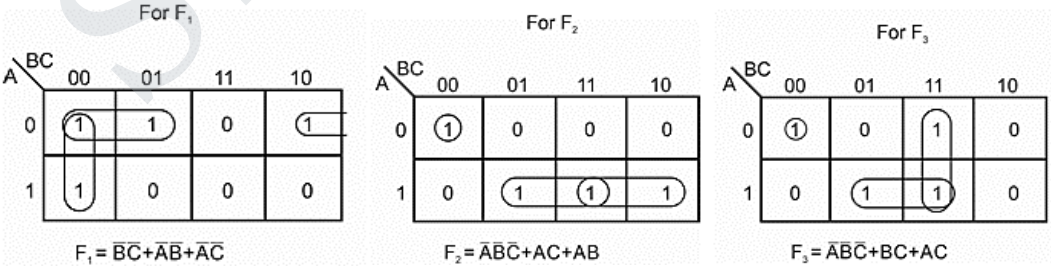
31. Implement the following Boolean function with a PLA?

$F_1(A, B, C) = \sum (0, 1, 2, 4)$

$F_2(A, B, C) = \sum (0, 5, 6, 7)$

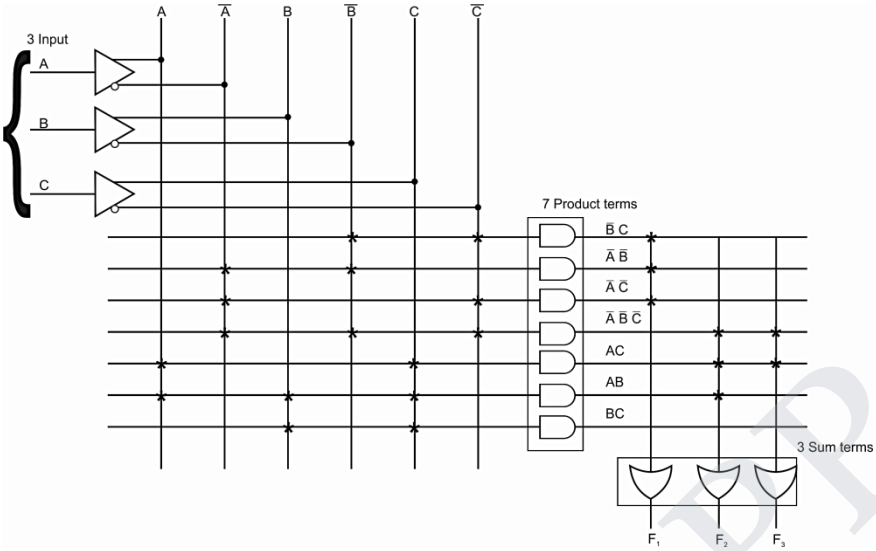
$F_3(A, B, C) = \sum (0, 3, 5, 7)$

K – Map Simplification



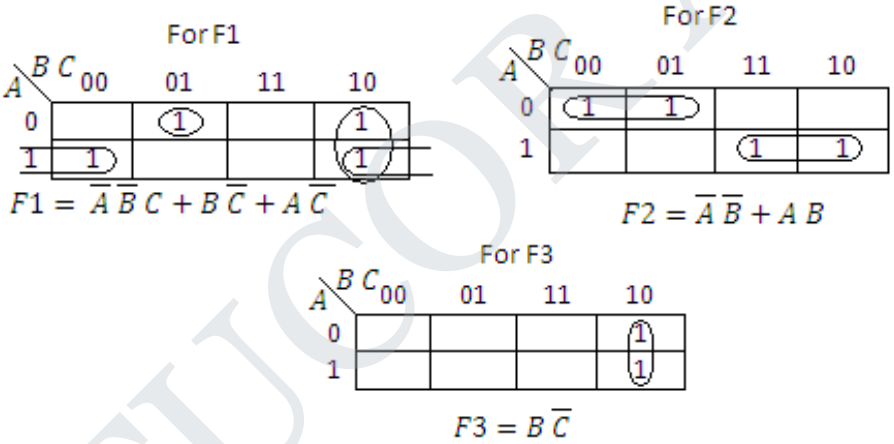
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



32. Implement the following function using PLA:$F_1 = \sum m(1,2,4,6)$; $F_2 = \sum m(0,1,6,7)$; $F_3 = \sum m(2, 6)$.

K – Map:

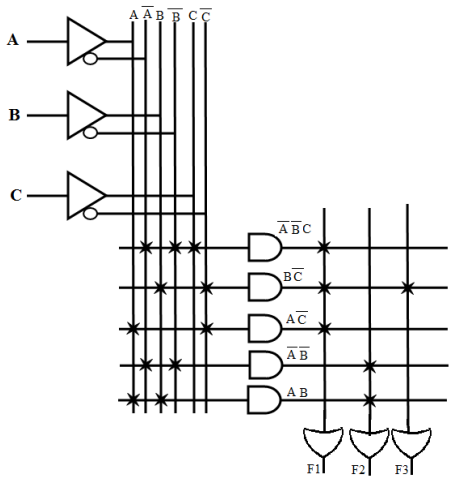


PLA table:

Product Term	Inputs			Outputs			
	A	B	C	F1	F2	F3	
$\overline{A}\overline{B}C$	0	0	1	1	-	-	
$B\overline{C}$	-	1	0	1	-	-	
$A\overline{C}$	1	-	0	1	-	1	
$\overline{AB}$	0	0	-	-	1	-	
AB	1	1	-	-	1	-	
				T	T	T	T/C

STUCOR APP
DOWNLOADED FROM STUCOR APP

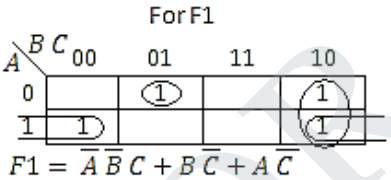
DOWNLOADED FROM STUCOR APP



33. Implement the following function using PLA: F(x,y,z) = ∑m (1, 2, 4, 6). *[Apr'15]*

F(x, y, z) = ∑m (1, 2, 4, 6)

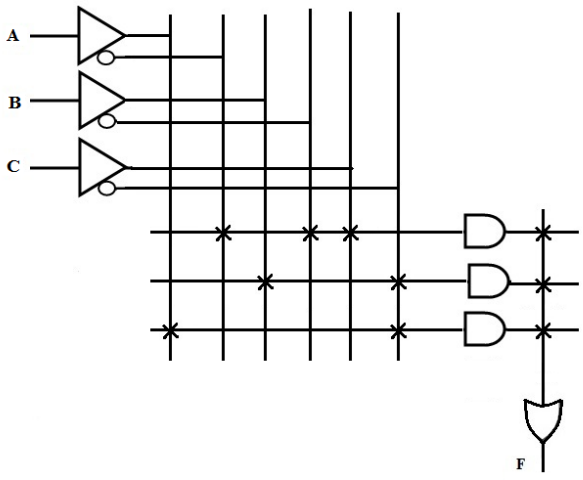
K – Map:



PLA table:

Product Term	Inputs				
	A	B	C	F	
$\overline{A}\overline{B}C$	0	0	1	1	
$B\overline{C}$	-	1	0	1	
$A\overline{C}$	1	-	0	1	
				T	T/C

Logic Diagram:

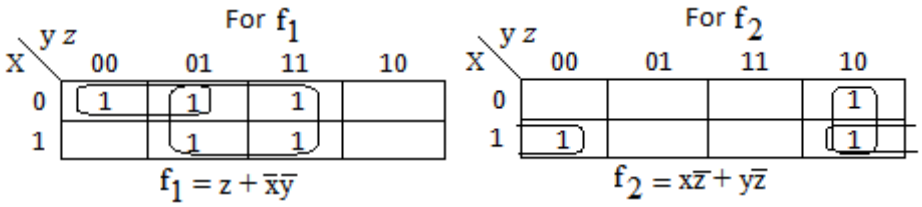


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

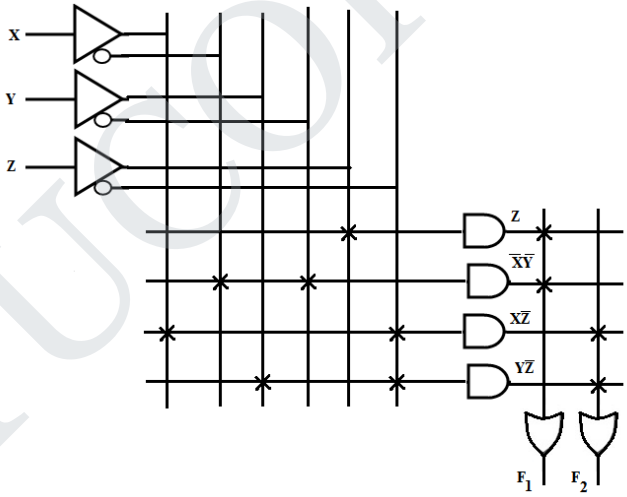
34. Implement the following function using PLA. $F_1(x,y,z)=\sum m(0,1,3,5,7)$, $F_2(x,y,z)=\sum m(2,4,6)$. *(May'17)*

K – Map: PLA table:



Product Term	Inputs			Outputs		
	x	y	z	F1	F2	
z	-	-	1	1		
$\bar{x}\bar{y}$	0	0	-	1		
$x\bar{z}$	1	-	0		1	
$y\bar{z}$	-	1	0		1	
				T	T	T/C

Logic Diagram:



35. Design and implement a 4-bit binary to gray code converter using a PLA. *[Nov'12]*

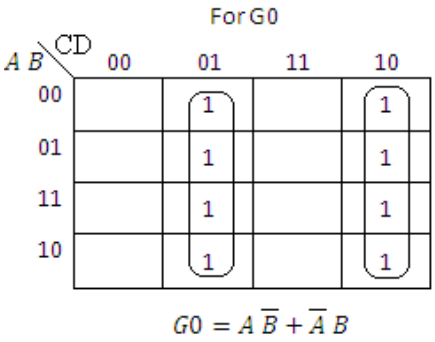
	Binary code				Gray code			
	D	C	B	A	G3	G2	G1	G0
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1

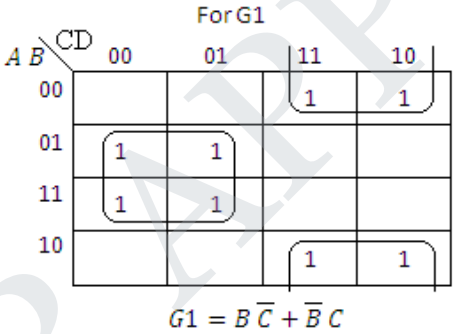
STUCOR APP
DOWNLOADED FROM STUCOR APP

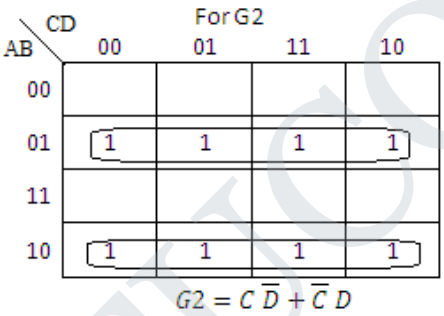
DOWNLOADED FROM STUCOR APP

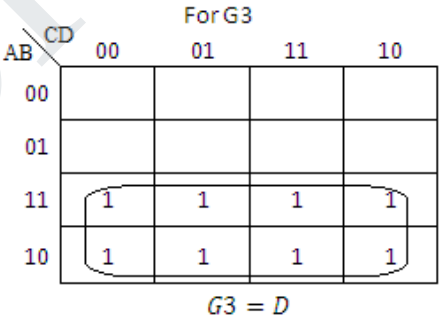
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

K – Map:





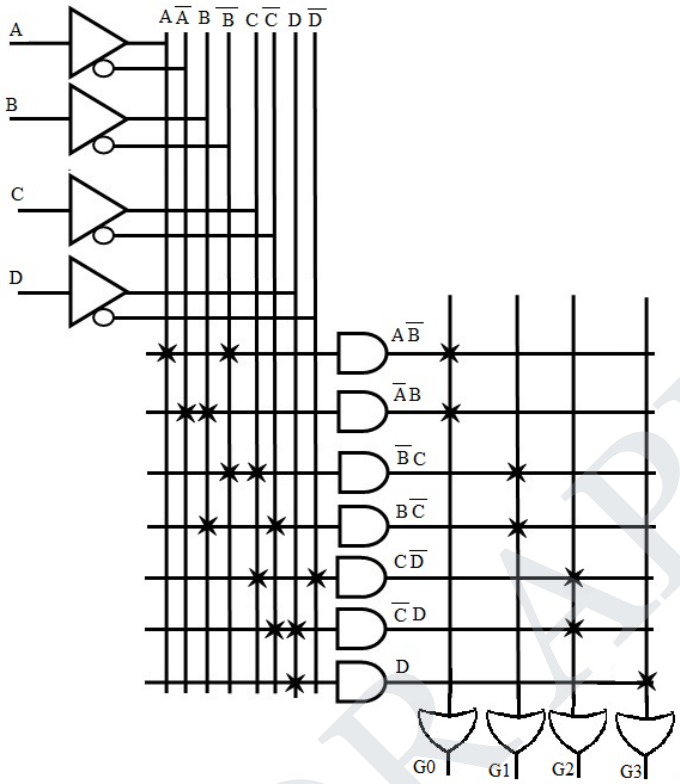




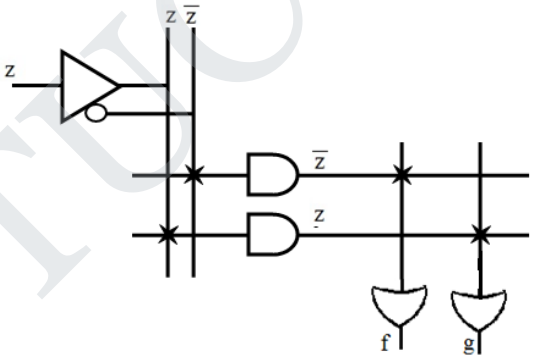
Product Term	Inputs				Outputs				
	D	C	B	A	G3	G2	G1	G0	
$A\overline{B}$	-	-	0	1	-	-	-	1	
$\overline{A}B$	-	-	1	0	-	-	-	1	
$B\overline{C}$	-	0	1	-	-	-	1	-	
$\overline{B}C$	-	1	0	-	-	-	1	-	
$C\overline{D}$	0	1	-	-	-	1	-	-	
$\overline{C}D$	1	0	-	-	-	1	-	-	
D	1	-	-	-	1	-	-	-	
					T	T	T	T	T/C

STUCOR APP
DOWNLOADED FROM STUCOR APP

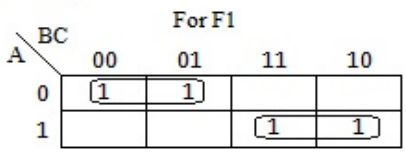
DOWNLOADED FROM STUCOR APP



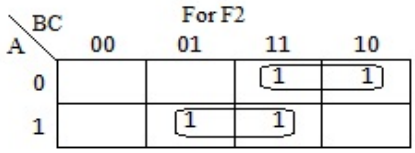
36. Design an AND-OR-PLA that implements the functions. f (x, y, z) = ∑m(0,2,4,6) g (x, y, z) = ∑m(1,3,5,7) *[Nov'12]*



37. A combinational circuit is defined by the following function. $F_1(a,b,c) = \sum (0,1,6,7)$, $F_2(a,b,c) = \sum (2,3,5,7)$. Implement the circuit with a PAL having three inputs, three product terms and two outputs.



$F1 = \overline{A}\,\overline{B} + A\,B$

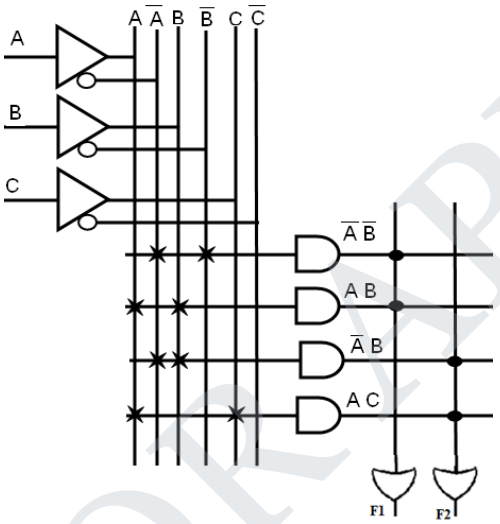


$F2 = \overline{A}\,B + A\,C$

STUCOR APP
DOWNLOADED FROM STUCOR APP

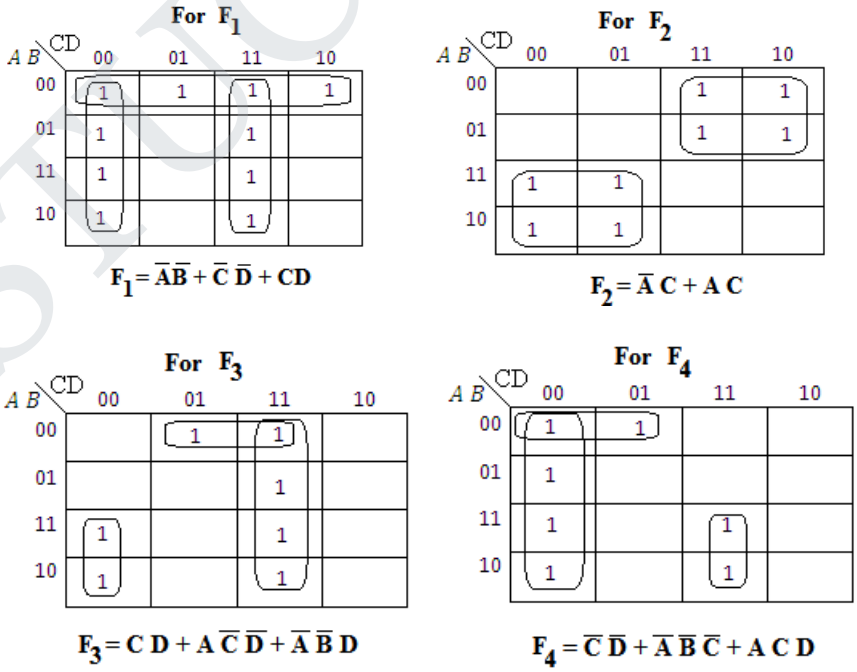
DOWNLOADED FROM STUCOR APP

Product Term	Inputs			Outputs
	A	B	C	
$\overline{A}\overline{B}$	0	0	-	F1
AB	1	1	-	
$\overline{A}$ B	0	1	-	F2
AC	1	-	1	



38. Design a PLA structure using AND and OR logic for the following functions. F1=∑m(0,1,2,3,4,7,8,11,12,15), F2=∑m(2,3,6,7,8,9,12,13), F3=∑m(1,3,7,8,11,12,15), F4=∑m(0,1,4,8,11,12,15) *(Nov'16)*

K – MAP:



$F_1 = \overline{A}\overline{B} + \overline{C}\,\overline{D} + CD$

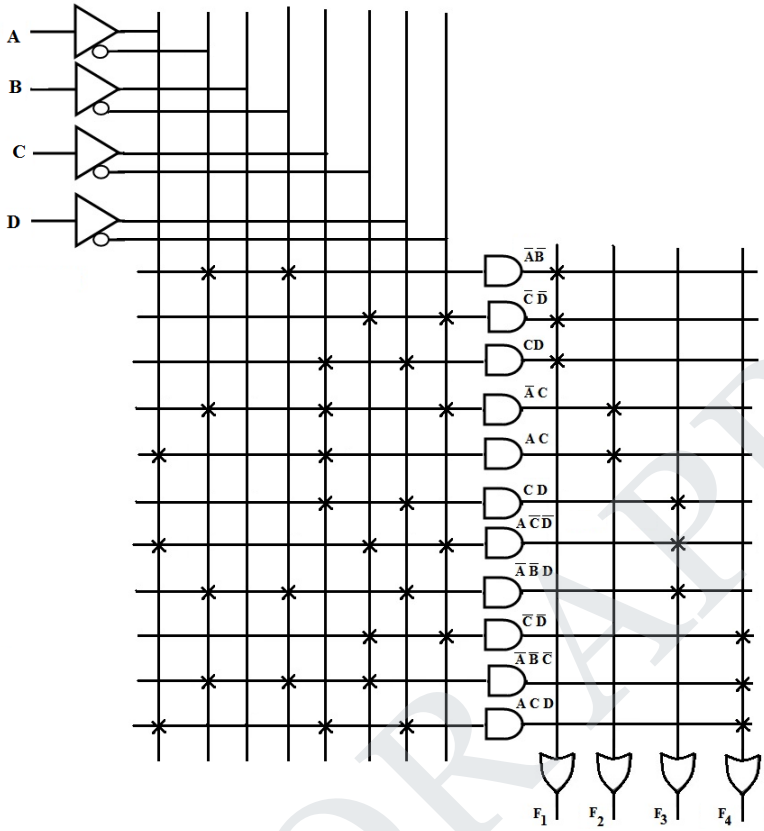
$F_2 = \overline{A}\,C + A\,C$

$F_3 = C\,D + A\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,D$

$F_4 = \overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C} + A\,C\,D$

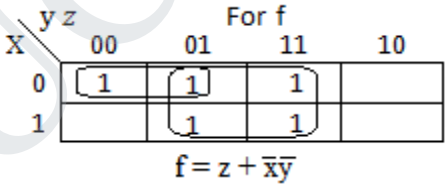
STUCOR APP
DOWNLOADED FROM STUCOR APP

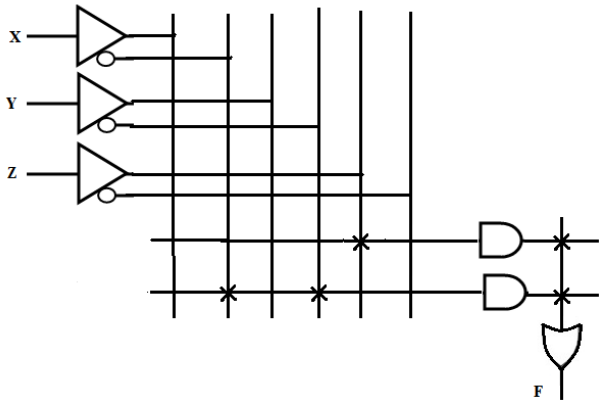
DOWNLOADED FROM STUCOR APP



39. Implement the following function using PLA and PAL. F(x,y,z) = ∑m(0,1,3,5,7). *(May '16)*

K – Map: PLA table:



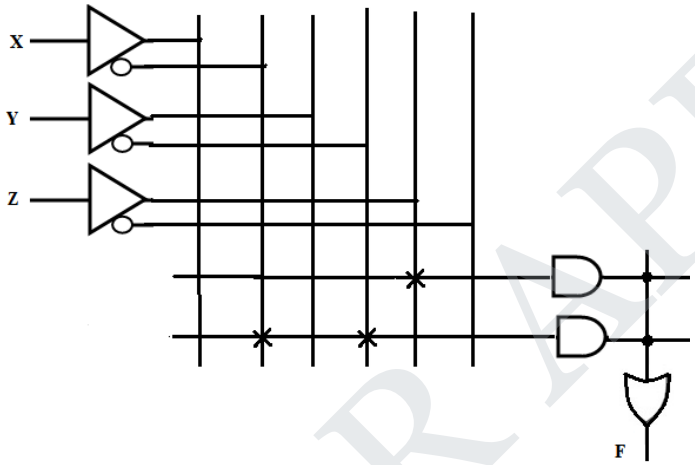


STUCOR APP
DOWNLOADED FROM STUCOR APP

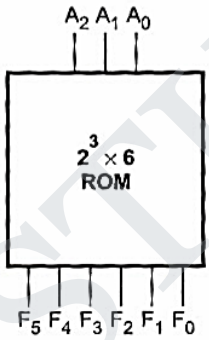
DOWNLOADED FROM STUCOR APP

PAL Table:

Product Term	Inputs			Outputs	
	x	y	z	f	
z	-	-	1	1	
$\bar{x}\bar{y}$	0	0	-	1	
				T	T/C



40. (a) Design a combinational circuit using ROM. The circuit accepts 3-bit number and generates an output binary number equal to square of input number. *[Apr'10]*



A_2	A_1	A_0	F_5	F_4	F_3	F_2	F_1	F_0
0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1
0	1	0	0	0	0	1	0	0
0	1	1	0	0	1	0	0	1
1	0	0	0	1	0	0	0	0
1	0	1	0	1	1	0	0	1
1	1	0	1	0	0	1	0	0
1	1	1	1	1	0	0	0	1

ROM truth table

(b) Repeat the above problem using PLA. *[Apr'10]*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

For F_5

A_2 \ A_1A_0	00	01	11	10
0	0	0	0	0
1	0	0	1	1

$F_5 = A_2 A_1$

For F_4

A_2 \ A_1A_0	00	01	11	10
0	0	0	0	0
1	1	1	1	0

$F_4 = A_2 \bar{A}_1 + A_2 A_0$

For F_3

A_2 \ A_1A_0	00	01	11	10
0	0	0	1	0
1	0	1	0	0

$F_3 = \bar{A}_2 A_1 A_0 + A_2 \bar{A}_1 A_0$

For F_2

A_2 \ A_1A_0	00	01	11	10
0	0	0	0	1
1	0	0	0	1

$F_2 = A_1 \bar{A}_0$

For F_1

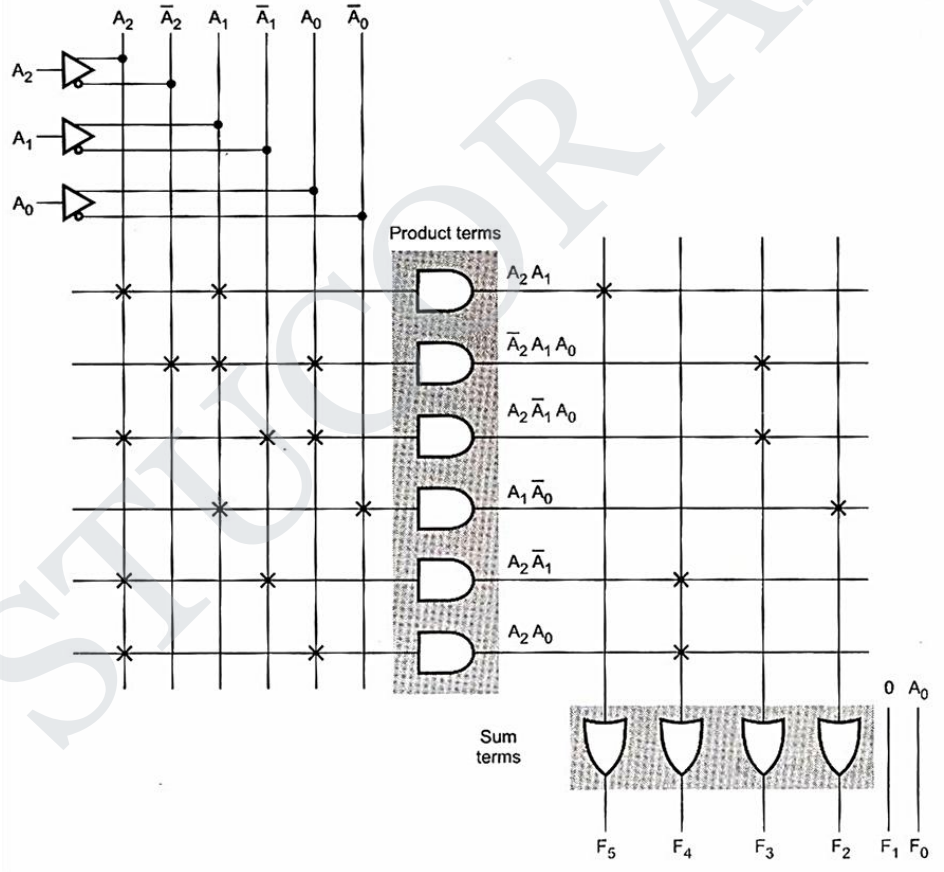
A_2 \ A_1A_0	00	01	11	10
0	0	0	0	0
1	0	0	0	0

$F_1 = 0$

For F_0

A_2 \ A_1A_0	00	01	11	10
0	0	1	1	0
1	0	1	1	0

$F_0 = A_0$



STUCOR APP
DOWNLOADED FROM STUCOR APP

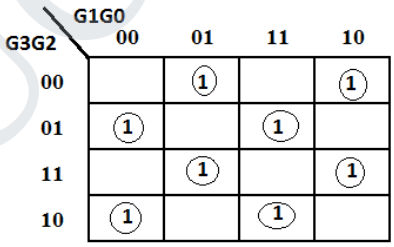
DOWNLOADED FROM STUCOR APP

41. Show how to program the fusible links to get a 4 bit Gray code from the Binary inputs using PLA and PAL and compare the design requirements with PROM. *[Nov'15]*

Step 1: Truth Table for 4 bit gray to Binary

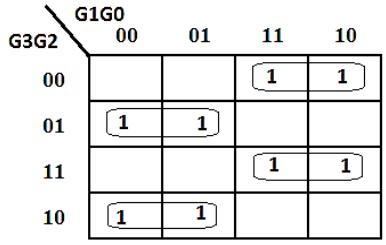
Gray code					Binary code			
Decimal	**G3**	**G2**	**G1**	**G0**	**D**	**C**	**B**	**A**
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	0
3	0	0	1	1	0	0	1	1
4	0	1	0	0	0	1	0	0
5	0	1	0	1	0	1	0	1
6	0	1	1	0	0	1	1	0
7	0	1	1	1	0	1	1	1
8	1	0	0	0	1	0	0	0
9	1	0	0	1	1	0	0	1
10	1	0	1	0	1	0	1	0
11	1	0	1	1	1	0	1	1
12	1	1	0	0	1	1	0	0
13	1	1	0	1	1	1	0	1
14	1	1	1	0	1	1	1	0
15	1	1	1	1	1	1	1	1

Step 2: K-map



$$A = \bar{G}3\bar{G}2\bar{G}1G0 + \bar{G}3\bar{G}2G1\bar{G}0 + \bar{G}3G2\bar{G}1\bar{G}0 + \bar{G}3G2G1G0$$
$$G3G2\bar{G}1G0 + G3G2G1\bar{G}0 + G3\bar{G}2\bar{G}1\bar{G}0 + G3\bar{G}2G1G0$$

A = (G3 ⊕ G2) ⊕ (G1⊕ G0)

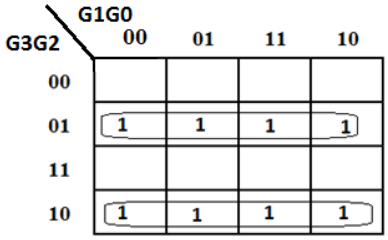


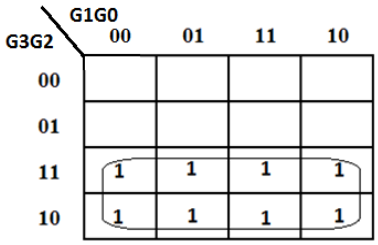
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

$B = \bar{G}3\bar{G}2G1 + \bar{G}3G2\bar{G}1 + G3G2G1 + G3\bar{G}2\bar{G}1$

B = G3⊕G2⊕G1





$C = \bar{G}3G2 + G3\bar{G}2$ **D = G3**

C = G3⊕G2

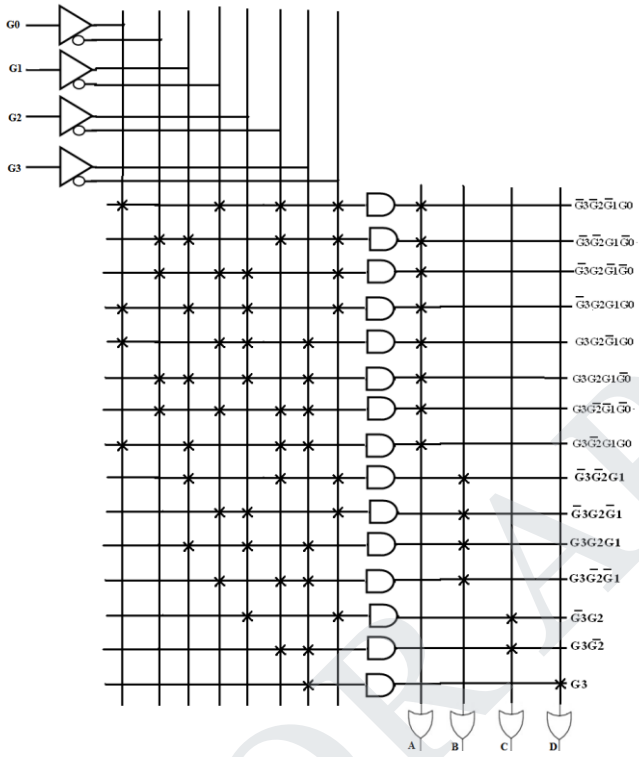
Step 3: PLA Table

Product Term	Inputs				Outputs				
	G3	G2	G1	G0	D	C	B	A	
$\bar{G}3\bar{G}2\bar{G}1G0$	0	0	0	1	-	-	-	1	
$\bar{G}3\bar{G}2G1\bar{G}0$	0	0	1	0	-	-	-	1	
$\bar{G}3G2\bar{G}1\bar{G}0$	0	1	0	0	-	-	-	1	
$\bar{G}3G2G1G0$	0	1	1	1	-	-	-	1	
$G3G2\bar{G}1G0$	1	1	0	1	-	-	-	1	
$G3G2G1\bar{G}0$	1	1	1	0	-	-	-	1	
$G3\bar{G}2\bar{G}1\bar{G}0$	1	0	0	0	-	-	-	1	
$G3\bar{G}2G1G0$	1	0	1	1	-	-	-	1	
$\bar{G}3\bar{G}2G1$	0	0	1	-	-	-	1	-	
$\bar{G}3G2\bar{G}1$	0	1	0	-	-	-	1	-	
$G3G2G1$	1	1	1	-	-	-	1	-	
$G3\bar{G}2\bar{G}1$	1	0	0	-	-	-	1	-	
$\bar{G}3G2$	0	1	-	-	-	1	-	-	
$G3\bar{G}2$	1	0	-	-	-	1	-	-	
G3	1	-	-	-	1	-	-	-	
					T	T	T	T	T/C

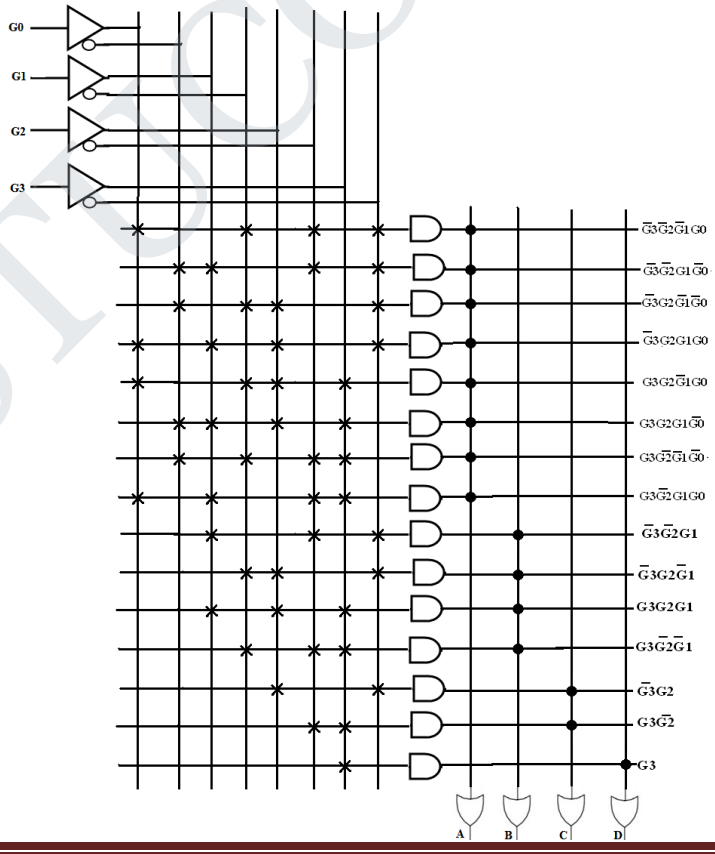
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Step 4 : Implementation Diagram



Step 5: PAL Implementation



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

41. Comparison of design requirements with PROM [OR] **Compare PLA and PAL circuits.** *(Nov'16)*

S.No.	PROM	PAL	PLA
1.	PROM AND array is Programmable.	AND array is Programmable and OR array is fixed.	Both AND & OR array are Programmable.
2.	Only Boolean functions in canonical SOP form can be implemented using PROM.	Any Boolean function is standard SOP form can be implemented using PAL.	Any Boolean function is standard SOP form can be implemented.

ANNA UNIVERSITY QUESTIONS
PART A

1. Define asynchronous sequential machine. *[Nov'10]*
2. Sketch the block diagram of an asynchronous sequential circuit? *[Nov'12]*
3. How does the operation of an asynchronous input differ from that of a synchronous input? *[May'12]*
4. What is the difference between asynchronous and synchronous sequential circuits? *[Apr'11]*
5. What are the drawbacks in designing asynchronous sequential machines? *[May'14]*
6. What is fundamental mode of operation in asynchronous sequential circuits? *[Nov'11]*
7. Compare Pulsed mode and fundamental mode asynchronous circuit. *[Nov'15]*
8. Draw the state diagram of SR flip-flop. *[Nov'10]*
9. Give the characteristic equation and state diagram of JK flip-flop. *[Apr'10]*
10. What is meant by transition table? *[Nov' 13]*
11. Define flow table in asynchronous sequential circuit. *[May'12]*
12. What is flow table? Give example.*[May'17]*
13. What is the difference between flow table and transition table? *[May'13]*
14. Define Race condition in Asynchronous sequential circuit? *[May'13]*
15. Define races in asynchronous sequential circuits. *[Nov'16]*
16. What is a deadlock condition? *[Nov'14]*
17. What is Self-starting counter? *[Apr'10]*
18. Name types of Hazards. *[Nov'13]*
19. What are hazards in asynchronous sequential circuits? *[Nov'11]*
20. State the hazards in asynchronous sequential circuits? *[Apr'11]*
21. State the difference between static 0 and static 1 hazard. *[Apr'15]*
22. What are the different classifications of memory? *[Nov'13]*
23. Name the types of ROM.*[Apr'11]*
24. What is a PROM? *[Apr'15],[Nov'15]*
25. Draw 4 X 2 ROM with AND –OR gates. *[Nov'13]*
26. What is PLA? *[May'12] (Nov'17)*
27. Draw the block diagram of PLA. *[Nov'14]*
28. What are the advantages of PLAs? *[Nov'12]*
29. What is the advantage of PLA over ROM? *[Apr'10]*
30. What is the difference between PROM and EPROM?*[May'13]*
31. Compare ROM, PROM and EPROM memory devices. *[Nov'12]*
32. What is FGPA? *[Nov'11, May'14]*

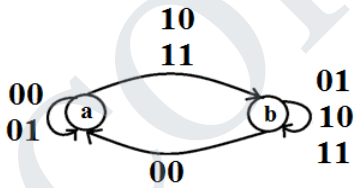
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

33. List the configurable elements in the FGPA architecture. *[May'12]*
34. What is static hazards and dynamic hazard?*[Nov'16]*
35. What are the two types of asynchronuous sequential circuits? *[May'16]*
36. State the difference between PROM, PLA and PAL. *[May'16, May'17]*
37. Define address and word. *(Apr'18)*
38. Why was PAL developed? *(Apr'18)*
39. What happens to the information stored in a memory location after it has been read and write operation? *(Nov'17)*

PART B & C [8/16 MARKS]

1. Derive the transition table and primitive flow table for the functional mode asynchronous sequential circuit. *[Nov' 2013]*
2. Consider the following asynchronous sequential circuit and draw maps and transition table and state table. *[May' 2013]*
3. Illustrate the analysis procedure of asynchronous sequential circuit with an example. *[May'13]*
4. What are transition table and flow table? Give Suitable examples. *[May'16]*
5. Discuss the steps involved in the design of an asynchronous sequential circuit with a suitable example. *[Nov'12]*
6. Describe the steps involved in design of asynchronous sequential circuit in detail with an example. *[Apr'11]*
7. Design an asynchronous sequential logic circuit for the state transition diagram shown in figure. *[Nov'11]*



8. Describe the procedure to get state table from excitation table in an asynchronous sequential circuit. How does it differ from asynchronous sequential circuit? *[Apr'10]*
9. Design an asynchronous sequential circuit that has two inputs X_2 and X_1 and one output Z. When $X_1 = 0$, the output Z is 0. The first change in X_2 that occurs while X_1 is 1 will cause output Z to be 1. The output Z will remain 1 until X_1 returns to 0. *[Apr'15]*
10. List and explain the steps used for analyzing an asynchronous sequential circuit.*[May'12]*
11. Design an asynchronous sequential circuit that has two inputs X_2 and X_1 and one output Z. When $X_1 = 0$, the output Z is 0. The first change in X_2 that occurs while X_1 is 1 will cause output Z to be 1. The output Z will remain 1 until X_1 returns to 0. *[Nov'15, Nov'17]*
12. Derive the flow table for the circuit given in the figure. *[May'12]*
13. An Asynchronous sequential circuit is described by$Y = x_1x_2' + (x_1 + x_2')\,y;\ z = y$. Draw the logic diagram, transition table and outline map. *[May'14]*
14. Find a circuit that has no static hazards and implements Boolean function F(A,B,C,D) = Σ(0,2,6,7,8,10,12) *[May'14]*
15. Explain the various types of hazards in sequential circuit design and the methods to eliminate them. Give suitable examples. *[Nov'14]*
16. What is hazards? Explain hazards in digital circuits? *(Nov'17)*
17. Describe with reason, the effect of races in asynchronous sequential circuit design. Explain its types with illustrations. Show the method of race – Free State assignments with examples. *[Nov'14]*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

18. Explain cycles and races in asynchronous sequential circuits. *[May '16]*
19. What are static -0 and static -1 hazards? Explain the removal of hazards using hazard covers in k-map? *(May '16)*
20. Discuss about hazards in asynchronous sequential circuit and the ways to eliminate them. *(Nov'17)*
21. For the given Boolean function, obtain the hazard – free circuit. F (A,B,C,D) = Σm (1,3,6,7,13,15). *[Apr'15]*
22. Implement the following logic and analyse for the pressure of any hazard F=$x_1x_2 + \overline{x_1}x_3$.If hazard is present briefly explain the type of hazard and design a hazard –free circuit. *(May'17)*
23. When do you get the critical and non-critical races? How will you obtain race free condition? *[Apr'10]*
24. Design an asynchronous BCD counter. *[Apr'11, May'14]*
25. Describe the characteristics of all types of memories. *[Apr'11]*
26. Describe the different types of memories. *[Apr'10, May'14]*
27. Explain the operation of bipolar RAM cell with suitable diagrams. *(Apr'18)*
28. Write notes on ROM and its types. *[May'12]*
29. Elaborate the concept of PROM, EPROM, EEPROM in detail. *(Apr'18)*
30. Write briefly about the programmable logic array and EPROM. *[Nov'11]*
31. Describe the working of EPROM. List the application EPROM.
32. Illustrate the basic principles of PLA and FPGA. *[May'13]*
33. Write short note on field programmable gate Array (FPGA).
34. Describe the concept and working of FPGA. *[May'12]*
35. What do you understand by FPGA? Explain the operation and applications? *[Nov'10]*
36. Write short notes on PLA and PAL. *(Nov'17)*
37. Design ROM for the following functions.$F_1 = \sum(1,2,3)$; $F_2 = \sum(0,2)$ *[Apr'11]*
38. Implement the following two Boolean functions with a PLA?

a. $F_1(A,B,C) = \sum(0,1,2,4)$

b. $F_2(A,B,C) = \sum(0,5,6,7)$*[Apr'11]*

39. Implement the following function using PLA: $F(x,y,z) = \sum m(1, 2, 4, 6)$. *[Apr'15]*
40. Implement the following function using PLA. $F_1(x,y,z)=\sum m(0,1,3,5,7)$, $F_2(x,y,z)=\sum m(2,4,6)$. *(May'17)*
41. Design and implement a 4-bit binary to gray code converter using a PLA. *[Nov'12]*
42. Design an AND-OR-PLA that implements the functions. $f(x, y, z) = \sum m(0,2,4,6)$
43. $g(x, y, z) = \sum m(1,3,5,7)$ *[Nov'12]*
44. Design a PLA structure using AND and OR logic for the following functions. $F1=\sum m(0,1,2,3,4,7,8,11,12,15)$, $F2=\sum m(2,3,6,7,8,9,12,13)$, $F3=\sum m(1,3,7,8,11,12,15)$, $F4=\sum m(0,1,4,8,11,12,15)$*(Nov'16)*
45. Implement the following function using PLA and PAL. $F(x,y,z) = \sum m(0,1,3,5,7)$. *(May '16)*
46. Design a combinational circuit using ROM. The circuit accepts 3-bit number and generates an output binary number equal to square of input number. *[Apr'10]*
47. Show how to program the fusible links to get a 4 bit Gray code from the Binary inputs using PLA and PAL and compare the design requirements with PROM. *[Nov'15]*
48. Compare PLA and PAL circuits. *(Nov'16)*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

UNIT V PART A

1. **What is meant by RTL design?**
   - In digital circuit design, **register-transfer level** (**RTL**) is a design abstraction which models a synchronous digital circuit in terms of the flow of digital signals (data) between hardware registers, and the logical operations performed on those signals.
2. **Where the RTL design is used?**
   - Register-transfer-level abstraction is used in hardware description languages (HDLs) like Verilog and VHDL to create high-level representations of a circuit, from which lower-level representations and ultimately actual wiring can be derived.
   - Design at the RTL level is typical practice in modern digital design.
3. **When can RTL be used to represent digital systems?**
   - When digital systems are composed of registers and combinational function blocks, the RTL can be used to represent digital systems.
4. **What is the meaning of the following RTL statement: T_1:ACC←ACC and MDR.** *(Dec'11)*
   - The contents of register ACC are bit wised ANDed with the contents of MDR register and the result is stored in ACC register when control signal T_1 is activated.
5. **What are ASM?** *(Apr'11)*
   - ASM stands for algorithmic state machine. The Algorithmic State Machine (ASM) method is a method for designing finite state machines. It is used to represent diagrams of digital integrated circuits.
   - The ASM diagram is like a state diagram but less formal and thus easier to understand. An ASM chart is a method of describing the sequential operations of a digital system.
6. **What is HDL?**
   - In electronics, a hardware description language (HDL) is a specialized computer language used to program the structure, design and operation of electronic circuits, and most commonly, digital logic circuits.
   - A hardware description language looks much like a programming language such as C; it is a textual description consisting of expressions, statements and control structures.
   - One important difference between most programming languages and HDLs is that HDLs explicitly include the notion of time.
   - HDLs form an integral part of electronic design automation (EDA) systems, especially for complex circuits, such as microprocessors.

STUCOR APP
DOWNLOADED FROM STUCOR APP

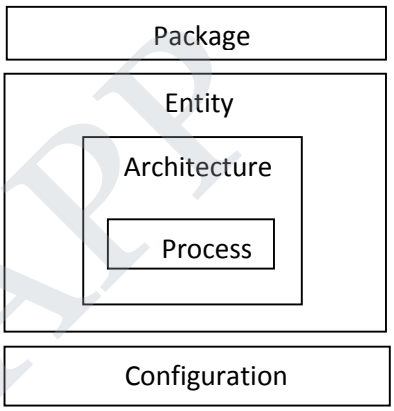
DOWNLOADED FROM STUCOR APP

7. What is the need for VHDL? *(May'13, May'14)*

1. VHDL represents' digital systems in the form of documentation which can be understood by human as well as computers.
2. It makes easy to exchange the idea between the designers.
3. VHDLs are used to describe hardware for the purpose of simulation, modeling, testing, and documentation.
4. It allows the designer to express their design with behavioral constructs.

8. What are the main components of a VHDL description?

- The main components of a VHDL description consists of following kinds of declarations
  - Package (optional)
  - Entity
  - Architecture
  - Configuration (optional)



9. What is entity?

- Entity gives the specification of input/output signal to external circuitry. It gives interfacing between device and the other peripherals. An entity usually has one or more ports, which are analogous to the pins on a schematic symbol.
- All information must flow into and out of the entity through the ports. Each port must contain name, data flow direction and type.

10. What is architecture?

- Architecture specifies behavior, functionality, interconnections or relationship between inputs and outputs.
- It is the actual description of the design. Architecture consists of two portions: architecture declaration and architecture body.

11. How is package represented?

A package is represented by:

- Package declaration
- Package body (optional)

12. What is a package in VHDL? *[Apr'15]*

- There are some declarations which are common across many design units.
- A package is a convenient mechanism to store and share such declarations.
- A set of declarations contained in a package declaration may be shared by many design units.
- It defines items that can be made visible to other design units.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

13. State the advantages of package declaration in VHDL over component declaration. *[Nov'14]*

- Component declarations are in the package.
- Do not have to be declared in the architecture.
- They occur in the package header
- Necessary to compile the package components prior to compiling the entity comparator since the use statement refers to something that already exists in the library.

14. What are the various modeling techniques in HDL? *(Apr'10, May'12, Nov'13)*

There are three modeling techniques in HDL for describing module
1. Gate level modeling /structural modeling
2. Dataflow modeling
3. Behavioral modeling

15. The module is the basic building blocks of VHDL. What are the different modeling techniques used to describe a module? *(Nov'12)*

1. Behavioral
2. Dataflow
3. Structural
4. Switch level
5. Mixed type
6. Mixed language

16. What is behavior modeling?

- The modeling style which directly describes the behavior of the functionality of a circuit is called behavioral modeling.
- It is very similar in syntax and semantics to that of a high-level programming language (for example: C, Pascal). A behavioral description models the system as to how the outputs behave with the inputs.

17. What is data flow modeling?

- Data flow describes how the circuit signals flow from the inputs to the outputs.
- There are some concurrent statements which allow to describe the circuit I terms of operations on signals and flow of signals in the circuit.
- When such concurrent statements are used in a program, the style is called a dataflow modeling.

18. What is structural modeling?

- The modeling style which uses components or gates to model the system is called structural modeling.

19. List the different type of operators supported by VHDL. *[Nov'12, Nov'13, Nov'15]*

1. Logical Operator → NOT, AND, OR, NAND, NOR & XOR
2. Arithmetic Operator →+, –, *, &, /

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

3. Relational Operators → =, /=, <, <=, >, >=
4. Concatenation Operator

20. Give the classification of data types supported by VHDL.

The VHDL data types can be broadly classified into following five data types:

i. **Scalar types**: the scalar types include numeric data types and enumerated data types. The numeric types consist of integer, floating point (real) and physical types. Bit, Boolean and character are all enumerated types.
ii. **Composite types**: array and record types are composite data types. The values of these types are collection of heir elements.
iii. **Access types**: they are pointers; they provide access to objects of a given data type.
iv. **File type**: they provide access to object that contain a sequence of values of a given type.
v. **Other types**: they include the data types provided by the several external libraries.

21. Give the syntax for VHDL entity declaration. [OR] **Give the syntax for package declaration and package body in VHDL.** *[May'17]*

The syntax of a VHDL entity declaration is as shown below:

```
Entity entity_name is

Port (          signal_names : mode signal_type;

signal_names : mode signal_type;

                :

Signal_names : mode signal_type);

End entity_name;
```

22. What is test bench?

- Before processing a design by synthesis tool, the designer usually wants to verify that the design performs according to the specification. This is almost always done by running a simulation.
- Simulating a design requires generation of test data and observation of simulation results. This process is done by used of a VHDL module that is referred to as test bench.

23. List the different types of test benches?

The different type of test benches are:

1. Stimulus only
2. Full test bench
3. Simulator specific
4. Hybrid test-bench
5. Fast test bench

24. What is the meaning of the following RTL statement?

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- The contents of register ACC are bitwise ANDed with the contents of MDR register and the result is stored in the ACC register when control signal T1 is activated.

25. Write the VHDL code for AND gate? [OR] **Write the VHDL code for AND gate?** [OR] **Write the VHDL code for a logical gate which gives high output only when both the input are high.** *(Nov'16)*

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity and gate is
port (A, B: in std_logic);
        Y: out std_logic);
end and gate;
architecture arch_and of and gate is
begin
        Y <= A and B;
end;
```

26. Write VHDL code for half adder in data flow model. *(Apr'14)* [OR] **What is the data flow modeling in VHDL? Give its basic mechanism.** *(May'16)*

```
//data flow model of half adder
entity half_adderis
Port  (     A, B : in bit;
            Sum, Cout : out bit);
end half_adder;
architecture adder of half_adderis
begin
              Sum < = A xor B;                -- signal assignment statement.
Cout< = A and B;                              -- signal assignment statement
end adder;
```

27. Write HDL for half adder?*(Apr'10, May'12, May'14)*

```
//Verilog behavioral model of half adder
    module half adder (A, B, sum, cout);
    input A, B;
    output sum, cout;
    reg sum, cout;
    always @ (A, B)
    begin
            #10 sum = A^B;
            #10 cout = A&B;
    end
    end module
// Verilog data flow model of half adder
    module half adder (A, B, sum, cout);
    input A, B;
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
output sum, cout;
        assign #10 sum = A^B;
        assign #10 cout = A&B;
endmodule
```

28. Write the HDL behavioral model of D flip flop? *(May'13)* [OR] **Write the behavioral modeling code for a D flip flop.** *[Apr'15, Nov'15, Nov'16]*

```
//Verilog behavioral model of D flip-flop
Module D_ff (D, CLK, Q);
Input D, CLK;
output Q;
reg Q;
always @(posedge CLK)
        Q = D;
End module
```

29. Write a VHDL code for a 2 x 1 Multiplexer using behavioural modeling. *[Nov'14, May'16, May'17]*

```
library ieee;
use ieee.std_logic_1164.all;
entity mux 2 x 1 is
port (D0, D1, S, Enbar : instd_ logic;
Y :outstd_logic);
end mux 2 x 1;
architecture MUX of mux 2 x 1 is
begin
process (S, D0, D1, Enbar)
variable temp : std_logic;
begin
        ifEnbar = ' 0' then
        if S = '1' then temp := D1;
elsetemp := D0;
end if;
        Y < = temp;
        else
        Y < = '0';
end if;
end process;
end MUX;
```

30. Define Cache memory? *(Apr'18)*

- It is a relatively small, high speed memory that can store the most recently used instruction or data from larger but slower main memory.

31. Infer the concept of switch –level modeling. *(Apr'18)*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- Verilog allows switch level modeling that is based on the behavior of MOSFETs. Digital circuit at the MOS-transistor level are described using the MOSFET switches.

32. Define modularity. *(Nov'17)*

- Modularity refers to the extent to which a software/Web application may be divided into smaller modules.
- Software modularity indicates that the number of application modules are capable of serving a specified business domain.
- Modularity is successful because developers use prewritten code, which saves resources. Overall, modularity provides greater software development manageability.

33. What are the languages that are combined together to get VHDL language? *(Nov'17)*

- VHDL is a hardware description language used in electronic design automation to describe digital and mixed-signal systems such as field-programmable gate arrays and integrated circuits.
- VHDL can also be used as a general purpose parallel programming language.

PART – B & C [8 / 16 Marks]

1.Explain the digital system design flow sequence with the help of a flow chart. *[Nov'14]*

- Figure presents a conceptual design show from specifications to final product. The figure shows a top-down approach that is very simplified: the reality of an industrial development is much more complex, involving many iterations through various portions of this show, until the final design converges to a form that meets the specification requirements of functionality, area, timing, power and cost.

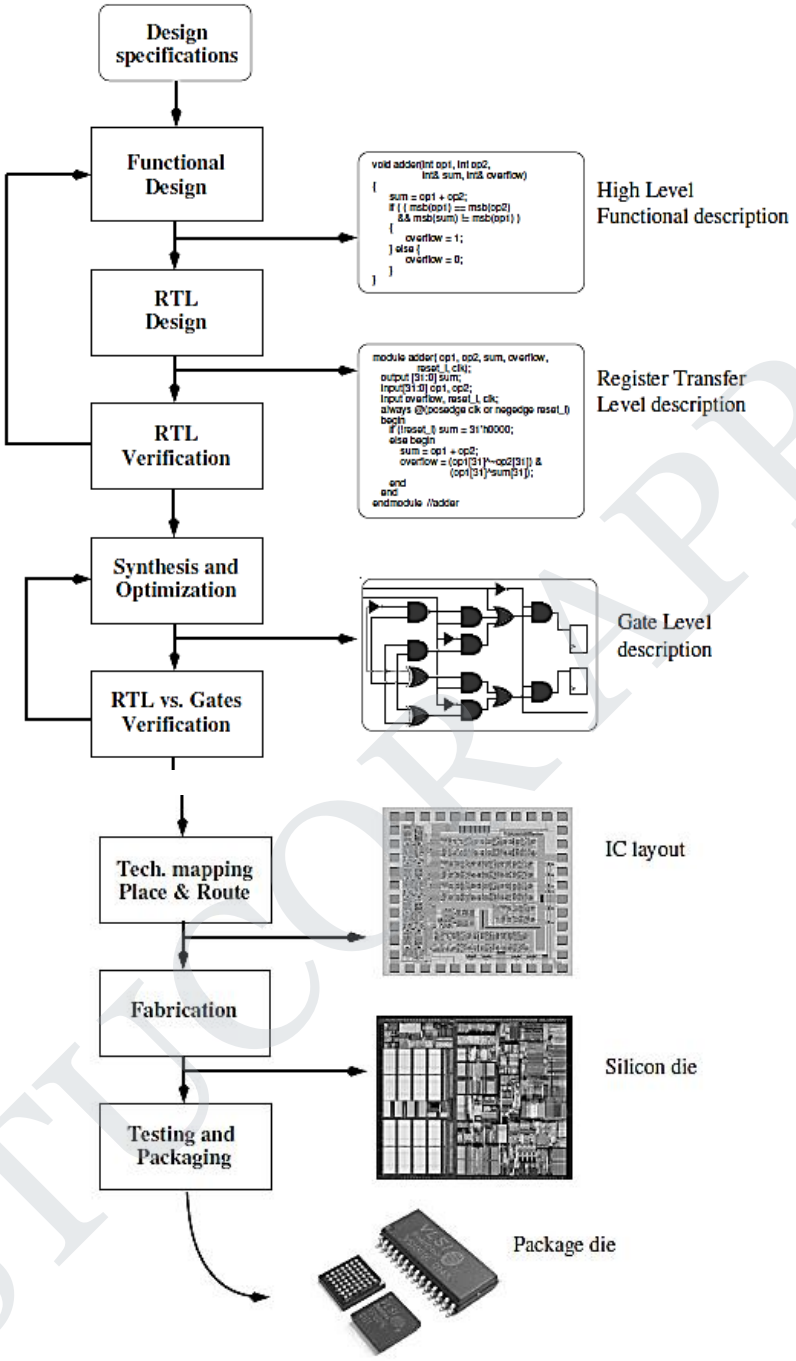
- The design specifications are generally presented as a document describing a set of functionalities that the final solution will have to provide and set constraints that it must satisfy. In this context, the functional design is the initial process of deriving a potential and realizable solution from this design specifications and requirements.

- This is sometimes referred to as modeling and includes such activities as hardware/ software tradeoffs and a micro-architecture design.

- Because of the large scale of the problem, the development of a functional design is usually carried out using a hierarchical approach, so that a single designer can concentrate on a portion of the model at any given time.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



- Thus, the architectural description provides a partition of the design in distinct modules, each of which contributes a specific functionality to the overall design.
- These modules have well defined input/output interfaces and protocols for communicating with the other components of the design.
- Among the results of this design phase is a high level functional description, often a software program in C or similar programming language that simulates the behavior of the design with the accuracy of one clock cycle and rejects the module partition.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- It is used for performance analysis and also as a reference model to verify the behavior of the more detailed designs developed in the following stages. From the functional design model, the hardware design team proceeds to the Register Transfer Level (RTL) design phase.
- During this phase, the architectural description is further refined: memory element and functional components of each model are designed using an Hardware Description Languages (HDL).
- This phase also sees the development of the clocking system of the design and architectural trade-offs such as speed/power.
- With the RTL design, the functional design of our digital system ends and its verification begins. *RTL verification* consists of acquiring a reasonable confidence that a circuit will function correctly, under the assumption that no manufacturing fault is present.
- The underlying motivation is to remove all possible design errors before proceeding to the expensive chip manufacturing. Each time functional errors are found the model needs to be modified to reject the proper behavior.
- During RTL verification, the verification team develops various techniques and numerous suites of tests to check that the design behavior corresponds to the initial specifications.
- When that is not the case, the functional design model needs to be modified to provide the correct behavior specified and the RTL design updated consequently.
- It is also possible that the RTL verification phase reveals overlooked aspects in the original set of specifications and this latter one needs to be updated instead. In the diagram of Figure, RTL verification appears as one isolated phase of the design show.
- However, in practical designs, the verification of the RTL model is carried on in parallel with the other design activities and it often lasts until chip layout.
- An overview of the verification methodologies that are common in today's industrial developments is presented in the next section.

2. **Explain the design procedure of RTL using VHDL.** *(Nov'10)* (OR) **Describe the RTL in VHDL.** *(Apr'10, May'12)* [OR] **Explain in detail the design procedure of RTL.** *(Dec'11, Nov'12, Nov'15)* [OR] **Explain in detail about the principal of operation of RTL design.** *(Apr'18)*
    - Register transfer level, or RTL design lies between a purely behavioral description of the desired circuit and a purely structural one.
    - An RTL description describes A circuit registers and the sequence of transfers between these registers but does not describe hardware used to carry out these operations.

    The steps in RTL design are:

STUCOR APP
DOWNLOADED FROM STUCOR APP

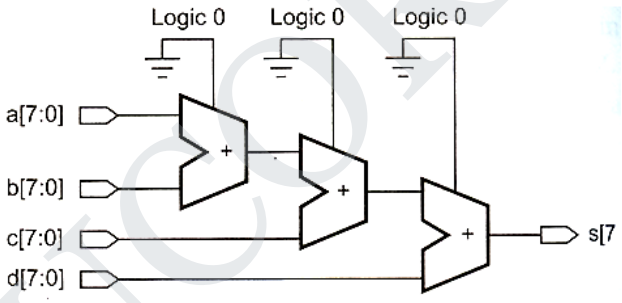
DOWNLOADED FROM STUCOR APP

1. Determine the number and size of the registers needed to hold the data used by the device
2. Determine the logic and arithmetic operations that need to be performed on these registers and contents
3. Design a sequential circuit whose outputs control how the register contents are updated in order to obtain the desired results

- Producing an RTL design is similar to writing a computer program in a conventional programming language.
- Choosing registers is the same as choosing variables.
- Designing the flow of data in the "data path" is analogous to writing expressions involving the variables and operators.
- Designing the controller sequential circuit is similar to deciding on flow control within the program.
- As a simple example, consider a device that needs to add four numbers.
- In VHDL given signals of the correct type, we can simply write:

S <= ((a+b) +c) +d;

- The particular description is simple enough that it can be synthesized.
- However, the resulting circuit will be fairly a large combinational circuit comprising three combinational circuits as shown in figure.



- A behavioral description, would be complete at this point however, if we were concerned about the cost of the implementation we might decide we break down the computation into sequence of steps, each only involving a single addition, where each operation is executed sequentially.

S=0;
S=s+a;
S=s+b;
S=s+c;
S=s+d;

- The logic circuit required is now one adder, a register to hold the value of 's' in between operations, a multiplexer to select the input to be added, and the circuit to clear s at the start of computation.
- Although the approach only needs one adder, the process requires more steps and takes longer time.
- Circuit that divide up into a sequence of arithmetic and logic operations require common and this type of design is called register transfer level or dataflow design.

STUCOR APP
DOWNLOADED FROM STUCOR APP

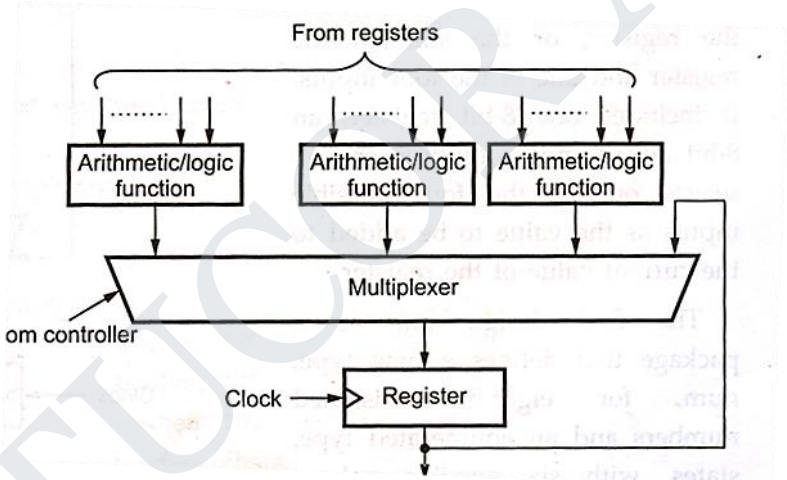
DOWNLOADED FROM STUCOR APP

RTL design is mainly composed of

1. Registers and combinational function blocks (e.g. adders and multipliers) called the datapath and

2. A finite sequential circuit, called the controller that controls the controls the transfer of data through the function blocks and between the registers.

- In VHDL RTL design, the gate level design and optimization of the datapath is done by the synthesizer.
- However, the designer must design the sequential circuit and decide which registers transfers are performed in which state.
- The RTL designer can trade off datapath complexity against speed.
- RTL design is well suited for CPUs and special purpose processors such as disk drive controllers, video display cards, network adapter cards; etc.
- It gives the designer great flexibility in choosing between processing speed and circuit complexity.



- The figure shows a generic component in the datapath. The structure allows the contents of each registers to be updated at the end of each clock period with a value selected by the controller.

- The widths of the registers, the types of combinational functions and their inputs will be determined by the application, a typical design will include many of these components.

Design procedure for Register Transfer Language:

- The basis of RTL design is that circuits can be thought of as a set of register and a set of transfer function defining the data paths between registers.
- The first stage of the design is to specify at a system level what is to be achieved, and logic operations on data coming in at the primary inputs of the circuits.
- At this stage there is no hardware implementations that can be used as the formal specification of the design.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- The second stage of the design is to transform the system level design in to and RTL design.
- It is rare for a design to be directly implemented in exactly the same form as the system level model.

The basics steps involved in RTL design are:

1. Identify the data operations
2. Determine the type and precision of the operations
3. Decide what data processing resources to provide
4. Allocate operations to resource
5. Design the controller
6. Design the reset mechanism

- The third stage of the design is to synthesize the RTL design.
- The resulting gate level net list of schematic can be simulated against the RTC design to confirm that the synthesized circuit has the same behavior.
- Finally the netlist or schematic produced by synthesis is supplied to the placement and routing tools for circuit layout.

3. Express how the arithmetic and logic operations are expressed using RTL? *(Dec'11)* [OR] **Explain the various operators supported by VHDL.** *[May'16]* [OR] **Give the different arithmetic operators and bitwise operators.** *(Apr'18)*

The basic arithmetic operations are

1. Add
2. Subtract
3. Increment
4. Decrement

5 .Arithmetic shift

Operation	Representation	Description
add	$R_3 \leftarrow R_1+R_2$	Contents of R_1 and R_2 are added and the result is transferred to R_3
subtract	$R_3 \leftarrow R_1-R_2$	Contents of R_2 are subtracted from the content R_1 of and the result is transferred to R_3
1's complement	$\overline{R_1}$	Complement the content of R_1
2's complement	$\overline{R_1}+1$	Complement the content of R_1 and add 1 in it
2's complement subtraction	$R_3 \leftarrow R_1+\overline{R_2}+1$	Add R_1 and the 2's complement of R_2
increment	$R_1 \leftarrow R_1+1$	Increment the contents of R_1 by one
decrement	$R_1 \leftarrow R_1-1$	Decrement the contents of R_1 by one

Logic micro operations:

- Logic micro operations perform logic operations such as AND, OR, complement and XOR on the strings of bits stored in registers.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Logic micro operations	Symbol
OR	∨
AND	∧
complement	Bar(¯)
EX-OR	⊕

Bitwise operartors:

Operator	Operation
~	Bitwise negation one
&	Bitwise AND two
|	Bitwise OR Two
^	Bitwise XOR Two
^- or -^	Bitwise XNOR Two
~&	Bitwise NAND two
~|	Bitwise NOR two

4. Express how arithmetic and logic operations are expressed using RTL. *(Nov'11)*

Logical operators

Operator	Operation
AND	A•B
OR	A+B
NAND	$\overline{AB}$
NOR	$\overline{A+B}$
XOR	A⊕B
XNOR	A⊙B
NOT	A•B

Arithmetic operator

Operator	Operation
+	Addition A+B
-	Subtraction A-B
*	Multiplication AxB
/	Division A÷B
mod	Modulus A mod B
rem	Remainder A rem B
abs	Absolute Abs A
&	Concatenation (A & B)
**	Exponent A**B

Shift and Rotate Operators:

Operator	Operation
Sll	A sll 1

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Sll	A sll 2
Srl	A srl 1
Srl	A srl 2
Sla	A sla 1
Sra	A sra 1
Rol	A rol 1
Ror	A ror 1

5. Write a note on VHDL test benches. *(Nov'10)*

- Before processing a design by synthesis tool, the designer usually wants to verify that the designer forms according to the specification.
- This is always done by running a simulation. Simulating a design requires generation of test data and observation of simulation results. This process is done by use of VHDL module that is referred as test bench.
- Test benches can be very simple, for example, just providing a sequence of test inputs to the circuit overtime.
- Sometimes this sequence represents a simple periodic waveform used to describe a system clock. However, sometimes it can be very complex one and stored in file.
- The test bench has to provide a reading of text vectors from the file, and also the writing of results to a report file.

- The figure shows typical structure of test bench. The test bench initiates the design under test/unit under test. The test bench provides the necessary input stimulates to the DUT and examines the output from the DUT.

The objective of test bench

1. Instantiate the DUT/UUT
2. Generate stimuli waveforms and apply them to the DUT/UUT in the form of test vectors during simulation.
3. Generate reference waveforms and compare them with the output from the DUT/UUT model during simulation
4. Provides pass/fail indications

Types of test benches

i. Stimulus only: this test bench contains only the stimulus driver and DUT; does not contain any result verification
ii. Full test bench: this test bench contains stimulus driver, known good results and result comparison
iii. Stimulator specific: this test bench is written in a simulator specific format
iv. Hybrid test bench: this test bench combines techniques from more than one test bench cycle
v. Fast test bench: these test benches are written to get ultimate speed from simulation

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Writing test bench

- When writing test bench, the DUT is included as a component in test bench model.
- Test bench is written as an entity just like any other VHDL model with an architecture body which include DUT as a component.

Syntax of test bench

```
entity b is
end tb;
architecture arch tb of tb is
        component DUT
        port (port name: mode type)
        end component;
        local signals;
        begin
                M1: DUT port map
        process begin
                signal assignments to create stimulus
                applying test vectors to DUT;
        end process;
end arch tb
```

The executable part includes the statement to generate test_ vectors.

With VHDL there are three ways to generate test vectors

i. Generation by algorithm :it includes
ii. Simple assignments of signals,
iii. Continuous loops for periodic signals
iv. Procedures for complex signal patterns
v. Generations from constant stored in array
vi. Generations from constant stored in separate file.

- Once the vectors are available, they are applied to DUT.
- Finally output is verified or written to the result file or error is reported.

Advantages of test benches

i. Test bench provide the user with the capability to test the DUT thoroughly through simulation.
ii. Test bench allows simulating DUT under a variety of test conditions including correct and faulty test inputs.
iii. Test bench makes it easier to create a wide range of stimulus inputs, particularly with respect to the timing of input signals.
iv. Its structure is well defined and flexible to accept the changes.
v. The output generated by test bench can be observed as waveforms.
vi. It gives idea about the timing requirements of the DUT.

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

vii. The test bench approach is useful for post fit verification.

6. **Write HDL program for full adder and 4 bit comparator?** *(Nov'10, May'14, Nov'14, Apr'15, Nov'15)* [OR] **Write the VHDL code to realize a 4 bit parallel binary adder with structural modeling and write the test bench to verify its functionality.** *(May'16)* [OR] **Design a 3 bit magnitude comparator and write the VHDL code to realize it using structural modeling .***[May'17]* [OR] **Explain in detail the concept of structural modeling in VHDL with an example of full adder.** *[Nov'16]* [OR] **Write a VHDL code to realize a full adder using behavioral modeling and structural modeling**. *(Nov'17)*

//The following gives both Verilog and VHDL model for both
//(i)VHDL behavioral description of full adder

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entityfull_addis
        port (A, B, Cin: in bit;
        sum, cout: out bit);
endfull_add;
architecture adder offull_addis
begin
        process (A, B, Cin)
        begin
        sum<= A xor B xorcin;
        cout<= (A and B) or (Cinand A) or (cinand B);
        end process;
end adder;
```

//(ii)VHDL data flow description of full adder

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entityfull_addis
        port (A, B, Cin: in bit;
        sum, cout: out bit);
endfull_add;
architecture adder offull_addis
begin
        sum<= A xor B xorCin;
        cout<= (A and B) or (Cinand A) or (Cinand B);
end adder;
```

//(iii)VHDL structural description of full adder

```
library IEEE;
use ieee.std _logic_1164.all;
entiy full_ adder is
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
            port (A,B, Cin: in std_logic;
                  sum,cout:outstd_ logic);
      end full_ adder;
      architecture adder offull_adderis                                    --
      component xor3
                  port (I1, I2, I3: in bit;
                        O1: out bit);
            end component
            component and2
                  port (I1, I2: in bit;
                        O1: out bit);
            end component
            component or3
                  port (I1, I2, I3: in bit;
                        O1: out bit);
            end component
            signal S1, S2, S3:bit;
            begin
                  Y1: xor3 port map (A, B, Cin, Sum);
                  X1: and2 port map (A, B, S1);
                  X1: and2 port map (A, Cin, S2);
                  X1: and2 port map (B, Cin, S3);
                  Y2: or3 port map (S1, S2, S3, cout);
      end adder;

// (iv) Verilog behavioral description of full adder

      module full adder (A, B, Cin, sum, cout);
      inputA, B, Cin;
      output sum, cout;
      reg sum, cout;
      always @ (A, B, Cin)
      begin
            #10 sum = (A ^ B) ^Cin;
            #10 cout = (A & B) l (Cin& A) l (Cin& B);
      end
      endmodule
//(vi)Verilog data flow description of full adder

      module full adder (A, B, Cin, sum, cout);
      inputA, B, Cin;
      output sum, cout;
            assign #10 sum = (A ^ B) ^Cin;
            assign #10 cout = (A & B) l (Cin& A) l (Cin& B);
      endmodule
// 4 bit magnitude comparator
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
//(i) VHDL code of a 4×4 magnitude comparator
library  IEEE;
use IEEE.STD_LOGIC_1164.all;
entity COMP_4 is
port (A,B : instd_logic_vector(3 downto 0);
AgtB, AltB, AeqB :outstd_logic);
end COMP_4;
architecture COMP of COMP_4 is
begin
        x3<=A(3) xnor B(3);
        x2<=A(2) xnor B(2);
        x1<=A(1) xnor B(1);
        x0<=A(0) xnor B(0);
AeqB<=x3 and x2 and x1 and x0;
AgtB<= (A(3) and notB(3))or
  (x3and A(2) and notB(2))or
        (x3and x2 and A(1) and notB(1)) or
                (x3and x2 and x1 and A(0) and notB(0))
AltB<= (notA(3) and B(3))or
  (x3and notA(2) and B(2))or
        (x3and x2 and notA(1) and B(1)) or
                (x3and x2 and x1 and notA(0) and B(0))
end COMP;
```

//(ii)Verilog behavioral description of 4x4 magnitude comparator

```
module 4×4 magnitude comparator(A3,A2,A1,A0,B3,B2,B1,B0,AgtB,AeqB,AltB);
input A3,A2,A1,A0,B3,B2,B1,B0;
outputAgtB, AeqB, AltB;
regX3, X2, X1, X0;

always@( A3,A2,A1,A0,B3,B2,B1,B0)

begin

 #10 X3 = A3~^B3;
 #10 X2 = A2~^B2;
 #10 X1 = A1~^B1;
 #10 X0 = A0~^B0;
#10 AeqB = X3&X2&X1&X0;

 #10 AgtB = (A3&(~B3))|(X3&(A2&(~B2)))|
            (X3&X2&(A1&(~B1)))|(X3&X2&X1&(A0&(~B0)));

 #10 AltB = ((~A3)&B3)|(X3&((~A2)&B2))|
           (X3&X2&((~A1)&B1))|(X3&X2&X1&((~A0)&B0));

end
endmodule
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

7. **Construct a VHDL module listing for a 16:1 MUX that is based on the design statement. Use a 4 bit select word S_3 S_2 S_1 S_0 to map the selected input p_1 (i=0, 15) to the output).** *[Nov'10]*

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
useIEEE.STD_LOGIC_VECTOR.all;

    entityMUX16_1 is
    port ( SEL: in STD_LOGIC_VECTOR(3 downto 0);
           P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15: in
    STD_LOGIC;
           MUX_OUT: out STD_LOGIC);
    endMUX16_1;

    architectureBEHAVIORAL of MUX16_1 is
    begin
    process (SEL, P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15)
    begin
    case SEL is
when “0000” = > MUX_OUT < = P0;
when “0001” = > MUX_OUT < = P1;
when “0010” = > MUX_OUT < = P2;
when “0011” = > MUX_OUT < = P3;
when “0100” = > MUX_OUT < = P4;
when “0101” = > MUX_OUT < = P5;
when “0110” = > MUX_OUT < = P6;
when “0111” = > MUX_OUT < = P7;
when “1000” = > MUX_OUT < = P8;
when “1001” = > MUX_OUT < = P9;
when “1010” = > MUX_OUT < = P10;
when “1011” = > MUX_OUT < = P11;
when “1100” = > MUX_OUT < = P12;
when “1101” = > MUX_OUT < = P13;
when “1110” = > MUX_OUT < = P14;
when “1111” = > MUX_OUT < = P15;
when others = > null;
end case;
end process;
endBEHAVIORAL;
```

8. **Construct VHDL module for a JK flip flop?** *(Dec'11, May'14)* **[**AND] **Write an HDL behavioral of JK flip flop using if else statement based on the value of present statement?** *(Apr'10, May'14)*

//(i) VHDL code for JK flip flop

```
Library ieee;
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
Use ieee.std_logic_1164.all;
entity JK_FF is
port (JK: in bit _vector (1downto 0);
clk: in _stdlogic;

      Q, Qbar: out bit);

end JK _FF;
architecture flip_ flop of JK_FF is
begin
P1: process (clk)
variable temp1, temp2: bit;
        begin
                ifrising_ edge (clk) then
                if (J='0') AND (K='1') then temp 1:="0";
                elsif (J='1') and (K='0') then temp1:='1';
                elsif (J='0') and (K='0') then temp1:=temp1;
                else temp1<= nottemp1;
                end if
        Q<=temp1;
        temp2:=nottemp1;
        Q bar<=temp2;
        end if;
end process P1;

end flip _flop;
```

9. Write the VHDL code for mod 6 counters? *(Apr'10, Apr'11, May'12)*

Let us see the VHDL code for the synchronous mod-6 counters with active low rest and set inputs.

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_unsigned.ALL;
entity counter6 is
port(
        CLK         : instd_logic;
Resetn     : instd_logic;
          SETN       : in std_logic;
          Q           : out std_logic_vector(5 downto 0));
end counter6;
architecturesynch_cntrof counter6 is
begin
process(CLK,  Resetn,  Setn)
        variableQtemp: std _logic _vector( 5 downto 0);
        begin
        ifResetn = '0'
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

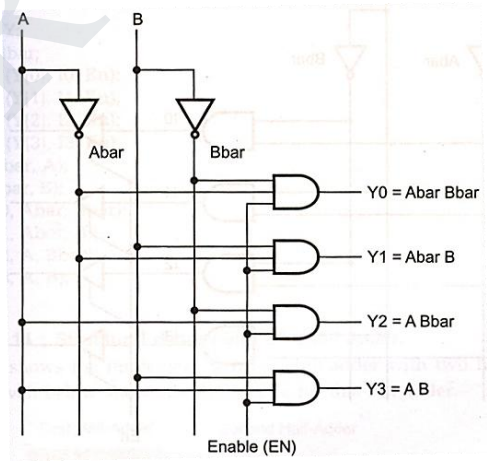
DOWNLOADED FROM STUCOR APP


```
                then Qtemp : = ”000”;
            elsif Setn = ‘0’
                then Qtemp  : = ”111”;
            elsif CLK =’1’ and CLK’event
            then
            if Qtemp< 6
                then Qtemp  : = Qtemp + 1;
            else
                Qtemp  : = ”000”;
            end if;
            end if;
            end if;
            end if;
                Q < = Qtemp;
        end process;
        endsynch_cntr;
```

10. Explain the structural VHDL description for a 2 to 4 decoder in details. *(Nov’13)*

//Verilog code for 2 to 4 line decoder with enable input

```
module decoder2×4 (A, B, En, Y)
input A,B;
input En;
output [3:0] Y;
wireAbar, Bbar;
not (Abar, A);
not (Bbar, B);
and (Y0, Abar, Bbar, En);
and (Y1, Abar, Bbar, En);
and (Y2, A, Bbar, En);
and (Y3, A, B, En);
```



//VHDL structural code for 2 to 4 line decoder

```
library IEEE;
use ieee.std _logic_1164.all;
entiy 2to4_decoder is
        port (A, B, En: in std_logic;
            Y0, Y1, Y2, Y3:out std_ logic);
end 2to4_decoder;
architecturestructof 2to4_decoder is                                    --
component not
            port (I1: in bit;
                O1: out bit);
        end component
        component and3
            port (I1, I2, I3: in bit;
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
                O1: out bit);
        end component
        signalAnot, Bnot, Abar, Bbar, y0, y1, y2, y3 :bit;
        begin
                Abar: not port map (A, Anot);
                Bbar: not port map (B, Bnot);
                Y0: and3 port map (Abar, Bbar, En, y0);
                Y1: and3 port map (Abar, B, En, y1);
                Y2: and3 port map (A, Bbar, En, y2);
                Y3: and3 port map (A, B, En, y3);
endstruct;
```

11. Write HDL for four bit binary counter with parallel load and explain. Description of a 4 bit synchronous counter with parallel load. *(May'13)*

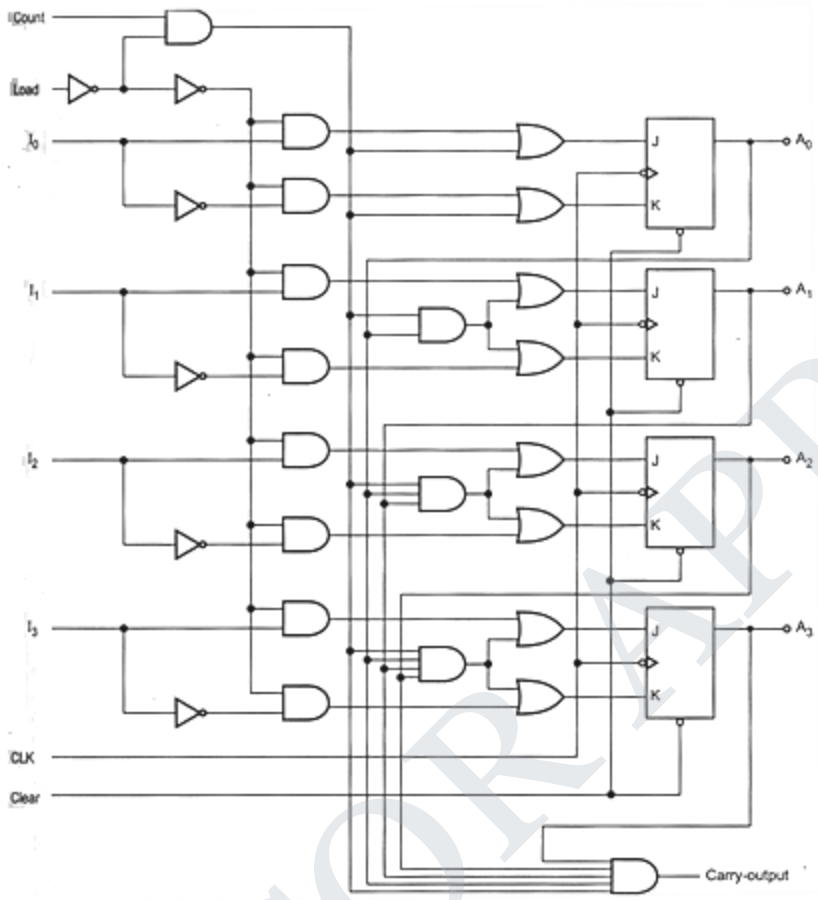
- The circuit of 4 bit binary counters with parallel load shown in figure. There are four inputs, count, load, and clock and clear which determines the operation of the counter.
- The counter has four data outputs (A_3-A_0) and four input data (i_0-i_3) and a carry output (co).the co is generated with an AND gate.so we can use '&' operator and an assign statement for this combinational circuit.
- If count=1, load=1.and A'=111, then co=1 otherwise co=0.when positive edge clock or negative edge clears, the operations which get performed are included in the always block.
- The CLR input is active low. So negative signal on CLR clears A .i.e A=0000.if CLR =1 and positive edge is present on clk, then one of the following operations take place.
  1. If load =1, input (i_3-i_0) is loaded into A
  2. If load =0 and count =1, A is incremented by 1
  3. If load =1 and count =0, the A remains changed

Clear	CLK	Load	Count	Function
0	x	x	x	Clear to 0
1	↑	1	x	Load inputs
1	↑	0	1	Count next binary state
1	↑	0	0	No change

Logic Diagram:

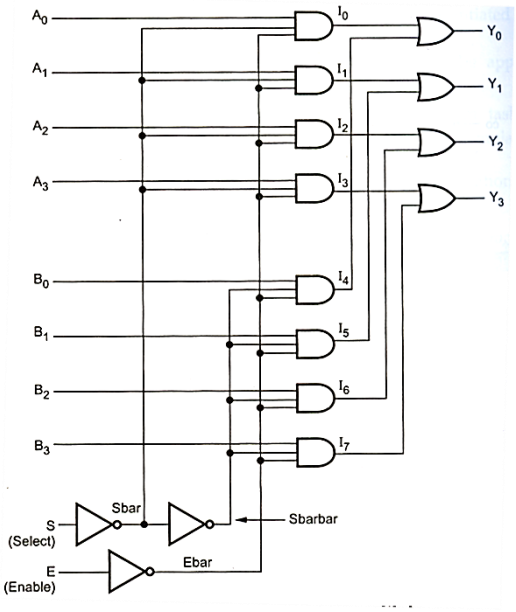
STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



// Description of a 4 bit binary counter with parallel load

```
module bin_ctr (count, load, I, CLK, CLR, A, CO)
input count, load;
input [3:0] I
output [3:0] A;
output co;
reg [3:0] A;
assign co=count &load & (A==4'a1111);
always @ (posedge CLK or negedge CLR)
if (CLR) A=4'b0000;
else if (load) A=1;
else if (count) A=A+1'b1;
else A=A'
end module
```



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

12. Write HDL for two to one quadruple multiplexer with dataflow description and behavioral description. *(May'13)*

```
module quadruplemux2_1 (A, B, S, E, Y)
input [3:0] A, B;
input S, E;
output [3:0] Y;
wire I0, I1, I2, 13, I4, IS, I6, I7, Sbar, Sbarbar, Ebar;
assign #10 I0 = A0 &Sbar&Ebar;
assign #10 I1 = A1 &Sbar&Ebar;
assign #10 I2 = A2 &Sbar&Ebar;
assign #10 I3= A3 &Sbar&Ebar;
assign #10 I4= B0 &Sbarbar&Ebar;
assign #10 I5 = B1 &Sbarbar&Ebar;
assign #10 I6 = B2 &Sbarbar&Ebar;
assign #10 I7= B3 &Sbarbar&Ebar;
assign # 10 Y0 = I0 | I4;
assign # 10 Y1 = I1 | I5;
assign # 10 Y2 = I2 | I6;
assign # 10 Y3 = I3 | I7;
end module
```

13. Briefly discuss the different data types supported in VHDL. *(Nov'12)*

- VHDL supports a variety of data types.
- The type of variable, signal or constant determines the operators that are predefined for object as well as the range of values taken on.
- The VHDL datatypes can be broadly classified in to following five data types.

1. **Scalar types**: the scalar types include numeric data types and enumerated data types. The numeric types consist of integer, floating point (real) and physical types. Bit, Boolean and character are all enumerated types.
2. **Composite types**: array and record types are composite data types. The values of these types are collection of heir elements.
3. **Access types**: they are pointers; they provide access to objects of a given data type.
4. **File type**: they provide access to object that contain a sequence of values of a given type.
5. **Other types**: they include the data types provided by the several external libraries.

14. Write an HDL code that implements an 8:1 multiplexer *(Nov'12, May'14)* [OR] **Design a 4x4 array multipler and write the VHDL code to realize it using structural modeling.** *[May'17]*

```
module multiplexer (D, S0, S1, S2, Y);
input [0 : 7]D, S0, S1, S2;
output  Y;
wire S0not, S1not, e, f, g, h, i, j, k, l;
not
N1 (S0not, S0);
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
N2 (S1not, S1);
N3 (S2not, S2);
and
A1 (e, D[0], S2not, S1not, S0not);
A2 (f, D[1], S2not, S1not, S0);
A3 (g, D[2], S2not, S1, S0not);
A4 (h, D[3], S2not, S1, S0);
A5 (i, D[4], S2, S1not, S0not);
A6 (j, D[5], S2, S1not, S0);
A7 (k, D[6], S2, S1, S0not);
A8 (l, D[7], S2, S1, S0);
or
O (Y, e, f, g, h, i, j, k, l)
end module
```

15. Briefly discuss the use of 'packages' in VHDL. *(Nov'12)* [OR] **Discuss briefly the packages in VHDL.** *(Nov'17)*

- There are some declarations which are common across many design units. A package is a convenient mechanism to store and share such declarations. It is an optional design units.
- A set of declarations contained in a package declarations may be shared by many design units. It defines items that can be made visible to other design units.
- A packages is represent the by:
  - Package declaration
  - Package body

- Package declaration define the interface to the package. The items declared in a package declaration can be accessed by other design units by using "Library" and "Use" clauses.

- Package body contains the details of a package that is behaviour of the subprograms and the values of the deferred constants which are declared in a package declaration.

- The package body may contain other declarations.

Package Declaration

```
Package MUX 4-to-1_package is
Component MUX 4-to-1
Port (M0, M1, M2, M3    :      IN STD_LOGIC;
                  S     :      IN STD_LOGIC_VECTOR (1 downto 0)
                  F     :      OUT STD_LOGIC;
End component;
End MUX 4-to-1_package;
```

16. Write a VHDL program and explain the design procedure of 8 bit comparator. *(Nov'13)*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
Library ieee;
Use ieee .std_logic_1164.all;
entity comparator is
port(A : instd_logic_vector (7 downto 0);
B :instd_logic_ vector (7 downto 0);
Less :outstd_logic;
Equal :outstd_logic;
Greater :out std_logic;
end comparator
architecture behavioral of comparator is
begin
      process (A, B)
      begin
      if (A<B)then
      Less <= '1';
      Equal <= '0';
      Greater <= '0';
      elsif (A=B) then
      Less <= '0';
      Equal <= '1';
      Greater <= '0';
      else
      Less <= '0';
      Equal <= '0';
      Greater <= '1';
      end if ;
      end process ;
endbehavioral;
```

17. Write a VHDL code for a 4-bit universal shift register. *[Nov'14]*

```
libraryieee;
use ieee.std_logic_1164.all;
entityshiftreg is
      Port (CLK, load, W, sh : in Std_logic;
      Data: in Std_logic_vector (3 downto 0);
       Q: inoutStd_logic_vector (3 downto 0));
End  shiftreg
architecturebehavioralofshiftregis
begin
  process (CLK, load)
  begin
         ifload = ' 1' then
         q <= data;
        end ifCLK = '1' and CLK event then
        ifSh = '1' then - - right shift
              q(0) <= q(1);
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

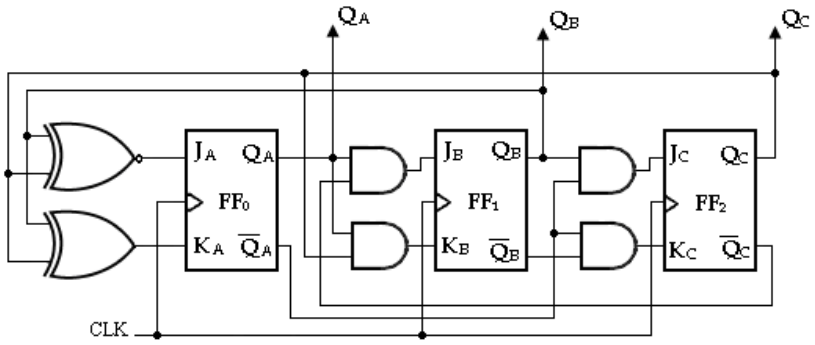

```
                    q(1) <= q(2);
                    q(2) <= q(3);
                    q(3) <=W;
elseifSh = '0' then - - left shift
                    q(3) <= q(2);
                    q(2) <= q(1);
                    q(1) <= q(0);
                    q(0) <=W;
end if;
end if;
end process;
endbehavioral;
```

18. Write the VHDL code to realize a 3-bit Gray code counter using case statement. *[Apr'15]*

```
library ieee;
use ieee.std_logic_1164.aii;
entity JK_FF is
port(JK : inbit_vector (1 downto 0);
        Click :instd_logic;
         Q, Qbar :out bit);
        endJK_FF;
architecture Flip-Flop of JK_FF is
begin
process (clk)
variable temp1, temp2 : bit;
 begin
 ifrising_edge (clk) then
 caseJK is
     when "01" = > temp1 := '0';
     when "10" = > temp1 := '1';
     when "00" = > temp1 := temp1;
     when "11" = > temp1 := not temp1;
 end case;
        Q < = temp1;
 temp2 := not temp1;
 Qbar< = temp2;
 end if;
 end process;
 endFlip_Flop;
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP



19. Write a VHDL Program for 1 to 4 Demux using dataflow modeling. *[Nov'15]*

```
library ieee;
use ieee.std_logic_1146.all;
entity1x4 Demuxis
port(D : inbit);
 S0S1 :inbit;
        Y0, Y1, Y2, Y3: out bit);
end1x4 DEMUX;
architecturearch DEMUX of Flip-Flop of1x4Demuliplier is
begin
Signal (y0, y1, y2, y3 );
   y0 = D and (not S0) and (not S1);
   y1= D and (not S0) and S1;
   y2= D and S0 and (not S1);
   y3= D and S0 and S1;
end arch 1x4 DEMUX;
```

20. VHDL Behavioral description of 2 x 1 multiplexer. *[May'16]*

```
libraryieee;
use ieee.std_logic_1164.all;
entity mux 2 x 1 is
port (D0, D1, S, Enbar : instd_ logic;
Y :outstd_logic);
end mux 2 x1;
architecture MUX of mux2 × 1 is
begin
process (S, D0, D1, Enbar)
variable temp : std_logic;
begin
        ifEnbar = ' 0' then
        if S = '1' then temp := D1;
elsetemp := D0;
end if;
        Y < = temp;
        else
        Y < = '0';
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
 end if;
      end process;
      end MUX;
```

21. VHDL code for behavioral description of D-Latch

```
libraryieee;
use ieee.std_logic_1164.all;
entity DLatch_varis
port (D, En, : in bit;
        Q, Qbar :out bit);
endDLatch_var;
architectureDL_Var of DLatch_varis
begin
process (D, En)
variabletemp1, temp2 : bit;
begin
ifEn = '1' then
temp1 := D;
temp2 := not temp1;
end if;
        Q < = temp1;
Qbar< = temp2;
end process;
endDL_Var;
```

22. VHDL positive Edge-Triggered JK Flip-Flop Using Case

```
libraryieee;
use ieee.std_logic_1164.aii;
entity JK_FF is
port(JK : inbit_vector (1 downto 0);
Click :instd_logic;
                Q, Qbar :out bit);
endJK_FF;
architecture Flip-Flop of JK_FF is
begin
process (clk)
variable temp1, temp2 : bit;
begin
ifrising_edge (clk) then
caseJK is
     when "01" = > temp1 := '0';
     when "10" = > temp1 := '1';
     when "00" = > temp1 := temp1;
     when "11" = > temp1 := not temp1;
end case;
            Q < = temp1;
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
temp2 := not temp1;
Qbar< = temp2;
end if;
end process;
endFlip_Flop;
```

23. VHDL 3-Bit Binary Counter Case Statement Description

```
Library ieee;
use ieee.std_logic_1164.all;
entity CT_Caseis
port ( clk, Reset : instd_logic;
Q :bufferstd_logic_vector  (2 dowento 0));
EndCT_Case;
architecture Counter_3b ofCT_caseis
begin
process(clk)
variable temp : std_logic_vector (2 downto 0);
begin
ifrising_edge (clk) then
if Reset = '0' then
case temp is
    when "000" = > temp := "001";
    when "001" = > temp :=  "010";
    when "010" = > temp := "011";
    when "011" = > temp := "100";
    when "100" = > temp := "101";
    when "101" = > temp := "110";
    when "110" = > temp:= "111";
    when "111" = > temp := "000";
    when others = > temp : ="000";
end case;
else
temp : = "000";
end if;
end if;
                    Q < = temp;
end process;
endcounter _3b;
```

24. VHDL Code for a Four-bit Up Counter

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity upctr is
port (Clock, Resetn, EN  : instd_logic;
Q :out std_logic_vector (3 downto 0));
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
end upctr;
architecture behavior of upctr is
signal count : std_logic_vector (3 downto 0));
begin
process (Clock, Resetn)
begin
ifResetn = '0' then
                    Count < = "0000";
              elsif(Clock'EVENTand Clock = '1') then
if EN = '1' then
                    Count < = Count +1;
else
                    Count < = Count;
end if;
end if;
end process;
      Q < = Count;
endbehaviour;
```

25. Write the VHDL code to realize a decade counter with behavioural modeling. *[May'16]*

VHDL Code for a 4-bit Down Counter

```
library  ieee;
use ieee.std_logic_1164.all;
entity downctr is
port (clock, load,  en  : instd_logic;
Q :outstd_logic_vector (3 downto 0));
End downctr;
architecture behavior ofdownctris
signal count : std_logic_vector (3 downto 0));
begin
process
process (Clock, Resetn)
begin
if Resetn = '0' then
   Count < = "1111";
   elsif (Clock'EVENTand Clock = '1') then
if EN = '1' then
                    Count < = Count -1;
else
                    Count < = Count;
end if;
end if;
end process;
      Q < = Count;
end behaviour;
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

26. Write VHDL coding for 4x1 Multiplexer. *[Nov'16, Nov'17]*

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity mux_4to1 is
 port(
A,B,C,D : in STD_LOGIC;
S0,S1: in STD_LOGIC;
Z: out STD_LOGIC
);
end mux_4to1;
 architecture bhv of mux_4to1 is
begin
process (A,B,C,D,S0,S1) is
begin
  if (S0 ='0' and S1 = '0') then
     Z <= A;
  elsif (S0 ='1' and S1 = '0') then
     Z <= B;
  elsif (S0 ='0' and S1 = '1') then
     Z <= C;
  else
     Z <= D;
  end if;
end process;
endbhv;
```

27. Write short note on built in operators used in VHDL programming. *[Nov'16]*

VHDL includes the following kinds of operators:

- Logical
- Relational
- Arithmetic

Logical Operators

- Logical operators, when combined with signals and/or variables, are used to create combinational logic. VHDL provides the following logical operators: AND, OR, NAND, NOR, XOR, NOT.
- These operators are defined for the types bit, std_ulogic (which is the base type of std_logic) and Boolean, and for one-dimensional arrays of these types (for example, an array of type bit vector or std_logic_vector)

Relational Operators

- Relational operators are used to create equality or magnitude comparison functions. VHDL provides the following relational operators:

  = Equal to
  /= Not equal to
  > Greater than
  < Less than

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

>= Greater than or equal to
<= Less than or equal to

- The equality operators (= and /=) are defined for all VHDL data types. The magnitude operators (>=, <=, >, <) are defined for numeric types, enumerated types, and some arrays. The resulting type for all these operators is Boolean.

Arithmetic Operators:

- Arithmetic operators are used to create arithmetic functions. VHDL provides the following arithmetic operators:

+Addition
-Subtraction
*Multiplication
/Division
mod Modulus
rem Remainder
abs Absolute Value
**Exponentiation

These operators are defined for numeric types such as integer and real.

Overloading Operators

- In addition to the predefined operators, VHDL allows you to create new operators, or to overload existing operators to support alternate types of arguments or to give them new meanings.
- For example, Synario supplies overloaded functions defining the relational operators listed in the previous section for type bit_vector as part of the package bit_ops contained in the file \synario\lib5\dataio.vhd.

28. Explain functions and subprograms with suitable examples. *[May'16]*

Function: A function call is a subprogram of the form of an expression that returns a value.

Syntax:

function function_name (parameters) **return** type;
function function_name (parameters) **return** type **is**
declarations
begin
sequential statements
end functionfunction_name;

Description:

The function definition consists of two parts:

- function declaration, which consists of name, parameter list and type of the values returned by the function;

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

- Function body, which contains local declarations of nested subprograms, types, constants, variables, files, aliases, attributes and groups, as well as sequence of statements specifying the algorithm performed by the function.

Example:

```
type Int_Data is file of NATURAL;
function Func_1 (A,B,X: REAL) return REAL;
function "*" (a,b: Integer_new) return Integer_new;
function Add_Signals (signal In1,In2: REAL) return REAL;
function End_Of_File (file File_name: Int_Data) return BOOLEAN;
```

Subprogram:

- The term subprogram is used as collective name for functions, procedures and operators.
- Operator definitions are treated as a special case of function definitions where the name is replaced by the operator symbol, enclosed by quotation marks (“).
- It is not permitted to declare new operators, i.e. it is just possible to provide a function with a different set of input parameters.
- This feature is called overloading (different subprograms differ by their parameters, only) and may be applied to all subprograms.
- Subprogram definitions consist of the subprogram declaration, where the identifier and parameter list are defined, and the subprogram body, defining the behavior.
- The statements in the subprogram body are executed sequentially. A function call is an expression (like ’a + b’) that can exist within a statement, only.
- A procedure call, on the other hand, is a statement (like ’c := a + b;’) and therefore it can be placed inside a process where it is executed sequentially or inside an architecture where it acts like any other concurrent statement.
- The return value is given after the keyword ’**return**’ which may be placed several times within a subprogram body. Please note that procedures do not have a return value!

29. Write a VHDL coding for realization of clocked SR flip flop. *(Nov’17)*

```
library ieee;
use ieee. std_logic_1164.all;
use ieee. std_logic_arith.all;
use ieee. std_logic_unsigned.all;

entity SR_FF is
PORT( S,R,CLOCK: in std_logic;
Q, QBAR: out std_logic);
end SR_FF;

Architecture behavioral of SR_FF is
begin
PROCESS(CLOCK)
```


STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP


```
variable tmp: std_logic;
begin
if(CLOCK='1' and CLOCK'EVENT) then
if(S='0' and R='0')then
tmp:=tmp;
elsif(S='1' and R='1')then
tmp:='Z';
elsif(S='0' and R='1')then
tmp:='0';
else
tmp:='1';
end if;
end if;
Q <= tmp;
QBAR <= not tmp;
end PROCESS;
end behavioral;
```

ANNA UNIVERSITY QUESTIONS

PART A

1. What is the meaning of the following RTL statement: T_1:ACC←ACC and MDR. *(Dec'11)*
2. What are ASM? *(Apr'11)*
3. What is the need for VHDL? *(May'13, May'14)*
4. What is a package in VHDL? *[Apr'15]*
5. State the advantages of package declaration in VHDL over component declaration. [Nov'14]
6. What are the various modeling techniques in HDL? (Apr'10, May'12, Nov'13)
7. The module is the basic building blocks of VHDL. What are the different modeling techniques used to describe a module? *(Nov'12)*
8. List the different type of operators supported by VHDL. *[Nov'12, Nov'13, Nov'15]*
9. Give the syntax for package declaration and package body in VHDL. *[May'17]*
10. Write the VHDL code for a logical gate which gives high output only when both the input are high. *(Nov'16)*
11. Write VHDL code for half adder in data flow model. *(Apr'14)*
12. What is the data flow modeling in VHDL? Give its basic mechanism. *(May'16)*
13. Write HDL for half adder?*(Apr'10, May'12, May'14)*
14. Write the HDL behavioral model of D flip flop? *(May'13)*
15. Write the behavioral modeling code for a D flip flop. *[Apr'15, Nov'15, Nov'16]*
16. Write a VHDL code for a 2 x 1 Multiplexer using behavioural modeling. *[Nov'14, May'16, May'17]*
17. Define Cache memory? *(Apr'18)*
18. Infer the concept of switch –level modeling. *(Apr'18)*
19. Define modularity. *(Nov'17)*
20. What are the languages that are combined together to get VHDL language? *(Nov'17)*

PART – B [8 / 16 Marks]

1. Explain the digital system design flow sequence with the help of a flow chart. *[Nov'14]*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

2. Explain the design procedure of RTL using VHDL. *(Nov'10)*
3. Describe the RTL in VHDL. *(Apr'10, May'12)*
4. Explain in detail the design procedure of RTL. *(Dec'11, Nov'12, Nov'15)*
5. Explain in detail about the principal of operation of RTL design. *(Apr'18)*
6. Express how the arithmetic and logic operations are expressed using RTL? *(Dec'11)*
7. Explain the various operators supported by VHDL. *[May'16]*
8. Give the different arithmetic operators and bitwise operators. *(Apr'18)*
9. Write a note on VHDL test benches. *(Nov'10)*
10. Write HDL program for full adder and 4 bit comparator? *(Nov'10, May'14, Nov'14, Apr'15, Nov'15)*
11. Write the VHDL code to realize a 4 bit parallel binary adder with structural modeling and write the test bench to verify its functionality. *(May'16)*
12. Design a 3 bit magnitude comparator and write the VHDL code to realize it using structural modeling .*[May'17]*
13. Explain in detail the concept of structural modeling in VHDL with an example of full adder. *[Nov'16]*
14. Write a VHDL code to realize a full adder using behavioral modeling and structural modeling. *(Nov'17)*
15. Construct a VHDL module listing for a 16:1 MUX that is based on the design statement. Use a 4 bit select word $S_3\,S_2\,S_1\,S_0$ to map the selected input p_1 (i=0, 15) to the output). *[Nov'10]*
16. Construct VHDL module for a JK flip flop? *(Dec'11, May'14)*
17. Write an HDL behavioral of JK flip flop using if else statement based on the value of present statement? *(Apr'10, May'14)*
18. Write the VHDL code for mod 6 counters? *(Apr'10, Apr'11, May'12)*
19. Explain the structural VHDL description for a 2 to 4 decoder in details.*(Nov'13)*
20. Write HDL for four bit binary counter with parallel load and explain. Description of a 4 bit synchronous counter with parallel load. *(May'13)*
21. Write HDL for two to one quadruple multiplexer with dataflow description and behavioral description.*(May'13)*
21. Briefly discuss the different data types supported in VHDL. *(Nov'12)*
22. 14. Write an HDL code that implements an 8:1 multiplexer *(Nov'12, May'14)*
23. Design a 4x4 array multipler and write the VHDL code to realize it using structural modeling. *[May'17]*
24. Briefly discuss the use of 'packages' in VHDL. *(Nov'12)*
25. Discuss briefly the packages in VHDL. *(Nov'17)*
26. Write a VHDL program and explain the design procedure of 8 bit comparator. *(Nov'13)*
27. Write a VHDL code for a 4-bit universal shift register. *[Nov'14]*
28. Write the VHDL code to realize a 3-bit Gray code counter using case statement. *[Apr'15]*
29. Write a VHDL Program for 1 to 4 Demux using dataflow modeling. *[Nov'15]*
30. VHDL Behavioral description of 2 x 1 multiplexer. *[May'16]*
31. Write the VHDL code to realize a decade counter with behavioural modeling.*[May'16]*
32. Write VHDL coding for 4x1 Multiplexer. *[Nov'16, Nov'17]*
33. Write short note on built in operators used in VHDL programming. *[Nov'16]*
34. Explain functions and subprograms with suitable examples. *[May'16]*
35. Write a VHDL coding for realization of clocked SR flip flop. *(Nov'17)*

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Reg. No. :

Question Paper Code : 40991

B.E./B.Tech. DEGREE EXAMINATION, APRIL/MAY 2018

Third Semester

Electrical and Electronics Engineering

EE 6301 – DIGITAL LOGIC CIRCUITS

(Common to Electronics and Instrumentation Engineering/Instrumentation and Control Engineering)

(Regulations 2013)

Time : Three Hours

Maximum : 100 Marks

Answer ALL questions

PART – A (10×2=20 Marks)

1. State the associative property of Boolean algebra.
2. Reduce $A(A + B)$.
3. Define duality property.
4. What is a karnaugh map ?
5. What is a master-slave flip-flop ?
6. Give the comparison between synchronous and asynchronous counters.
7. Define address and word.
8. Why was PAL developed ?
9. Define Cache memory.
10. Infer the concept of switch-level modeling.

PART – B (5×13=65 Marks)

11. a) i) Prove that $ABC + ABC' + AB'C + A'BC = AB + AC + BC$. (8)

ii) Convert the given expression in canonical SOP form $Y = AC + AB + BC$. (5)

(OR)

b) Designing a 4-bit Adder-Subtractor circuit. (13)

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

12. a) Write down the steps in implementing a Boolean function with levels of AND gates. (13)

(OR)

b) Give the general procedure for converting a Boolean expression in to multilevel NAND diagram. (13)

13. a) Explain the operation of SR flip-flop, T flip-flop and JK flip-flop. (13)

(OR)

b) Explain the flip-flop excitation tables for JK flip-flop and RS flip-flop. (13)

14. a) Elaborate the concept of PROM, EPROM, EEPROM in detail. (13)

(OR)

b) Explain the operation of bipolar RAM cell with suitable diagram. (13)

15. a) Give the different arithmetic operators and bitwise operators. (13)

(OR)

b) Explain in detail about the principal of operation of RTL design. (13)

PART – C (1×15=15 Marks)

16. a) Draw the circuit of CMOS AND gate and explain its operation. Also implement using PHDL. (15)

(OR)

b) Design and explain and bit shift register. Also give its truth table with its input and output waveform. (15)

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Reg. No. : | | | | | | | | | | | | |

Question Paper Code : 50473

B.E./B.Tech. DEGREE EXAMINATION, NOVEMBER/DECEMBER 2017

Third Semester

Electrical and Electronics Engineering

EE 6301 – DIGITAL LOGIC CIRCUITS

(Common to Electronics and Instrumentation Engineering/Instrumentation and Control Engineering)

(Regulations 2013)

Time : Three Hours | Maximum : 100 Marks

Answer ALL questions

PART – A (10×2=20 Marks)

1. Convert $(115)_{10}$ and $(235)_{10}$ to hexadecimal numbers.
2. What is a gray code and mention its advantages.
3. What is a K-map ?
4. Compare decoder and demultiplexer.
5. What do you mean by race around condition in a flip-flop ?
6. What is a preset table counter and ripple counter ?
7. What happens to the information stored in a memory location after it has been read and write operation ?
8. What is Programmable Logic Array ?
9. Define modularity.
10. What are the languages that are combined together to get VHDL language ?

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

50473

PART – B (5×13=65 Marks)

11. a) Explain in detail about error detecting and error correcting code. (13)

(OR)

b) Write short notes on following : (13)
i) RTL ii) DTL iii) TTL and iv) ECL

12. a) I) Plot the logical expression $ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}C + AB$ on a 4-variable K-map; obtain the simplified expression from the map. (7)

II) Express the function $Y = A + \bar{B}C$ in canonical SOP and canonical POS form. (6)

(OR)

b) Design a 4-bit gray code to binary converter and express using logic gates. (13)

13. a) Explain the operation, state diagram and characteristics of T-flip-flop and master-slave JK flip-flop. (13)

(OR)

b) Explain in detail about different shift registers. (13)

14. a) Discuss about the hazards in asynchronous sequential circuit and the ways to eliminate them. (13)

(OR)

b) I) Write short notes on PLA and PAL. (7)

II) What is hazards ? Explain hazards in digital circuits. (6)

15. a) Write a VHDL code to realize a full adder using behavioural modeling and structural modeling. (13)

(OR)

b) I) Discuss briefly the packages in VHDL. (6)

II) Write an VHDL coding for realization of clocked SR flip-flop. (7)

PART – C (1×15=15 Marks)

16. a) Design an asynchronous sequential circuit with two inputs x_1 and x_2 and one output Z. Initially, both inputs are equal to zero. When x_1 or x_2 becomes 1, the output Z becomes 1. When the second input also becomes 1, the output changes to 0. The output stays at 0 until the circuit goes back to the initial state. (15)

(OR)

b) I) Design a full adder using 4 × 1 multiplexer, also write its truth table and draw the logical diagram. (8)

II) Describe level triggering and edge triggering. (7)

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

B.E./B.Tech. DEGREE EXAMINATION, APRIL/MAY 2017.

Third Semester

Electrical and Electronics Engineering

EE 6301 — DIGITAL LOGIC CIRCUITS

(Common to Electronics and Instrumentation Engineering, Instrumentation and Control Engineering)

(Regulations 2013)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. Reduce $a(b+bc')+ab'$.

2. Convert 143_{10} into its binary and binary coded decimal equivalent.

3. Write the *POS* form of the *SOP* expression $f(x,y,z) = x'yz + xyz' + xy'z$.

4. Design a Half Subtractor.

5. Give the characteristic equation and characteristic table of a T Flip Flop.

6. State the differences between Moore and Melay state machines.

7. What is a flow table? Give example.

8. State the difference between PROM, PAL and PLA.

9. Give the syntax for package declaration and package body in VHDL.

10. Write the VHDL code for a 2 × 1 multiplexer using behavioral modeling.

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

PART B — (5 × 13 = 65 marks)

11. (a) (i) Design a odd-parity hamming code generator and detector for 4-bit data and explain their logic.

(ii) Convert $FACE_{16}$ into its binary, octal and decimal equivalent.

Or

(b) (i) With circuit schematic explain the working of a two-input TTL NAND gate.

(ii) Compare Totem Pole and open collector outputs.

12. (a) (i) Reduce the following minterms using Karnaugh – Map
$f(w,x,y,z) = \sum m(0,1,3,5,6,7,8,12,14) + \sum d(9,15)$. (7)

(ii) Implement the following function using a suitable multiplexer
$f(a,b,c) = \sum m(3,7,4,5)$. (6)

Or

(b) (i) Design a 3 × 8 decoder and explain its operation as a minterm generator. (7)

(ii) Design a full adder using only NOR gates. (6)

13. (a) (i) Draw and explain the operation of a Master – Slave JK Flip Flop. (7)

(ii) Design a 5-bit ring counter and mention its applications. (6)

Or

(b) (i) Design a 4-bit parallel-in serial-out shift register using D Flip Flops. (7)

(ii) Using partitioning minimization procedure reduce the following state table : (6)

Present state	Next state		Output
	$w = 0$	$w = 1$	Z
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

14. (a) A control mechanism for a vending machine accepts nickels and dimes. It dispense merchandise when 20 cents is deposited ; it does not give change if 25 cents is deposited. Design the FSM that implements the required control, using as few states as possible. Find a suitable assignment and derive next-state and output expressions. (13)

Or

(b) (i) Implement the following logic and analyse for the pressure of any hazard $f = x_1x_2 + \bar{x}_1x_3$. If hazard is present briefly explain the type of hazard and design a hazard-free circuit. (7)

(ii) Implement the following functions using programmable logic array :

$f_1(x, y, z) = \Sigma m(0, 1, 3, 5, 7)$

$f_2(x, y, z) = \Sigma m(2, 4, 6)$. (6)

15. (a) Design a 3 –bit magnitude comparator and write the VHDL code to realize it using structural modeling. (13)

Or

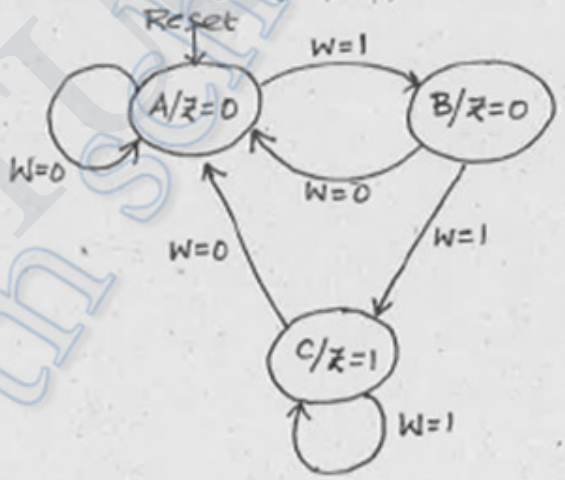
(b) Design a 4 × 4 array multiplier and write the VHDL code to realize it using structural modeling. (13)

PART C — (1 × 15 = 15 marks)

16. (a) Design a CMOS inverter and explain its operation. Comment on its characteristics such as Fan-in, Fan-out power dissipation, propagation delay and noise margin. Compare its advantages over other logic families. (15)

Or

(b) Write the VHDL code for the given state diagram, using behavioral modeling. Design it using one-hot state assignment and implement it using Programmable Array Logic (PAL). (15)



STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Reg. No. AUHIPPO.COM*

Question Paper Code : 80366

B.E./B.Tech. DEGREE EXAMINATION, NOVEMBER/DECEMBER 2016.

Third Semester

Electrical and Electronics Engineering

EE 6301 — DIGITAL LOGIC CIRCUITS

(Common to Electronics and Instrumentation Engineering and Instrumentation and Control Engineering)

(Regulations 2013)

Time : Three hours Maximum : 100 marks

auhippo.com

PART A — (10 × 2 = 20 marks)

1. Construct OR gate and AND gate using NAND gates.
2. Convert the following Excess - 3 numbers into decimal numbers.
   (a) 1011
   (b) 1001 0011 0111
3. Convert the given expression in canonical SOP form
   Y = AB + A'C + BC'
4. Draw the truth table of 2 :1 MUX.
5. Differentiate Mealy and Moore model.
6. Draw the state diagram of JK flip flop.
7. What is static hazard and dynamic hazard?
8. Define races in asynchronous sequential circuits.
9. Write VHDL behavioral model for D flip flop.
10. Write the VHDL code for a logical gate which gives high output only when both the inputs are high.

PART B — (5 × 13 = 65 marks)

11. (a) (i) Explain with an aid of circuit diagram the operation of 2 input CMOS NAND gate and list out its advantages over other logic families. (10)

    (ii) Given the two binary numbers X = 1010100 and Y = 1000011, perform the subtraction $Y - X$ by using 2's complements. (3)

Or

(b) (i) Explain in detail the usage of Hamming codes for error detection and error correction with an example considering the data bits as 0101. (10)

    (ii) Convert 23.625_{10} to octal (base 8). (3)

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

auhippo.com

12. (a) Simplify the logical expression using K-map in SOP and POS form $F(A,B,C,D) = \Sigma m(0,2,3,6,7) + d(8,10,11,15)$. (13)

Or

(b) Design a full subtractor and realise using logic gates. Also, implement the same using half subtractors (13)

13. (a) Design a sequence detector that produces an output '1' whenever the non-overlapping sequence 101101 is detected. (13)

Or

(b) (i) Explain the realization of JK flip flop from T flip flop. (7)

(ii) Write short notes on SIPO and draw the output waveforms. (6)

14. (a) Design an asynchronous circuit that has two inputs $x1$ and $x2$ and one output z. The circuit is required to give an output whenever the input sequence (0,0), (0,1) and (1, 1) received but only in that order (13)

Or

(b) (i) Design a PLA structure using AND and OR logic for the following functions. (10)

$F1 = \Sigma m(0, 1, 2, 3, 4, 7, 8, 11, 12, 15)$

$F2 = \Sigma m(2, 3, 6, 7, 8, 9, 12, 13)$

$F3 = \Sigma m(1, 3, 7, 8, 11, 12, 15)$

$F4 = \Sigma m(0, 1, 4, 8, 11, 12, 15)$

(ii) Compare PLA and PAL circuits. (3)

15. (a) Explain in detail the concept of structural modeling in VHDL with an example of full adder. (13)

Or

(b) (i) Write short notes on built- in operators used in VHDL programming. (6)

(ii) Write VHDL coding for 4 × 1 Multiplexer. (7)

PART C — (1 × 15 = 15 marks)

16. (a) Assume that there is a parking area in a shop whose capacity is 10. No more than 10 cars are allowed inside the parking area and the gate is closed as soon as the capacity is reached. There is a gate sensor to detect the entry of car which is to be synchronized with the clock pulse. Design and implement a suitable counter using JK flip flops. Also, determine the number of flip flops to be used if the capacity is increased to 50. (15)

Or

(b) Design a 4 bit code converter which converts given binary code into a code in which the adjacent number differs by only 1 by the preceding number. Also, develop VHDL coding for the above mentioned code converter. (15)

auhippo.com

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Question Paper Code : 57308

B.E./B.Tech. DEGREE EXAMINATION, MAY/JUNE 2016

Third Semester

Electrical and Electronics Engineering

EE 6301 – DIGITAL LOGIC CIRCUITS

(Common to Electronics and Instrumentation Engineering and Instrumen Control Engineering)

(Regulation 2013)

Time : Three Hours **Maximum**

Answer ALL questions.

PART – A (10 × 2 = 20 Marks)

1. Convert the following binary code into a Gray Code :

   1010111000_2

2. Define fan-in and fan-out.

3. Write the POS representation of the following SOP function :

   $f(x, y, z) = \Sigma m(0, 1, 3, 5, 7)$

4. Design a half subtractor.

5. Give the characteristic equation and characteristic table of SR flip-flop.

6. State any two differences between Moore and Mealy state machines.

7. What are the two types of asynchronous sequential circuits ?

8. State the difference between PROM, PLA and PAL.

9. What is data flow modelling in VHDL ? Give its basic mechanism.

10. Write the VHDL code to realize a 2 × 1 multiplexer.

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

PART – B (5 × 16 = 80 Marks)

11. (a) (i) Convert 10101110111011100_2 into its octal, decimal and hexadecimal equivalent. (6)

(ii) Deduce the odd parity hamming code for the data : 1010. Introduce an error in the LSB of the hamming code and deduce the steps to detect the error. (10)

OR

(b) (i) With circuit schematic explain the operation of a two input TTL NAND gate. (8)

(ii) With circuit schematic and explain the operation and characteristics of a ECL gate. (8)

12. (a) (i) Simplify the following function using Karnaugh Map :
$f(w, x, y, z) = \sum m(0, 1, 3, 9, 10, 12, 13, 14) + \sum d(2, 5, 6, 11)$ (8)

(ii) Implement the following function using only NAND gates :
$f(x, y, z) = \sum m(0, 2, 4, 6)$ (8)

OR

(b) (i) Design a BCD to Excess-3 code converter. (8)

(ii) Design a full adder and implement it using suitable multiplexer. (8)

13. (a) (i) Explain the operation of a JK master slave flip flop. (8)

(ii) Design a MOD-5 counter using T Flip Flops. (8)

OR

(b) (i) Design a serial adder using Mealy state model. (8)

(ii) Explain the state minimization using partitioning procedure with a suitable example. (8)

14. (a) (i) What are Static-0 and Static-1 hazards ? Explain the removal of hazards using hazard covers in K-map. (8)

(ii) Explain cycles and races in asynchronous sequential circuits. (8)

OR

(b) (i) What are transition table and flow table ? Give suitable examples. (6)

(ii) Implement the following function using PLA and PAL : (10)
$f(x, y, z) = \sum m(0, 1, 3, 5, 7)$

15. (a) (i) Explain the various operators supported by VHDL. (8)

(ii) Write the VHDL code to realize a decade counter with behavioural modelling. (8)

OR

(b) (i) Explain functions and subprograms with suitable examples. (6)

(ii) Write the VHDL code to realize a 4-bit parallel binary adder with structural modelling and write the test bench to verify its functionality. (10)

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Reg. No. | | | | | | | | | | | | |

Question Paper Code : 27206

B.E./B.Tech. DEGREE EXAMINATION, NOVEMBER/DECEMBER 2015.

Third Semester

Electrical and Electronics Engineering

EE 6301 — DIGITAL LOGIC CIRCUITS

(Common to Electronics and Instrumentation Engineering and Instrumentation and Control Engineering)

(Regulations 2013)

Time : Three hours | Maximum : 100 marks

Answer ALL questions.

PART A — ($10 \times 2 = 20$ marks)

1. What is an unit distance code? Give an example.
2. Define Fan-out.
3. Convert the given expression in canonical SOP form $Y = AB + A'C + BC'$.
4. Draw the logical diagram of EX-OR gate using NAND gates.
5. Draw the truth table and state diagram of SR flip-flop.
6. What is edge triggered flip flops?
7. What is PROM?
8. Compare pulsed mode and fundamental mode asynchronous circuit.
9. Write the behavioral model of D flip flop.
10. List out the operators present in VHDL.

PART B — ($5 \times 16 = 80$ marks)

11. (a) (i) Draw the CMOS logic circuit for NOR gate and explain its operation. (8)

(ii) Perform the following operation $(756)_8 - (437)_8 + (725)_{16}$. Express the answer in octal form. (8)

Or

STUCOR AP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

(b) (i) A 12 bit Hamming code word containing 8 bits of data and 4 parity bits is read from memory. What was the original 8 bit data word that was written into memory if the 12 bit word read out is as (1) 101110010100 and (2) 111111110100. (12)

(ii) Briefly discuss weighted Binary code. (4)

12. (a) (i) Simplify the boolean function using K-map and implement using only NAND gates.

$F(A,B,C,D) = \sum m(0,8,11,12,15) + \sum d(1,2,4,7,10,14)$.

Mark the essential and non-essential prime implicants. (8)

(ii) Design a full subtractor and implement using logic gates. (8)

Or

(b) (i) Design a 4 bit BCD to excess 3 code converter and implement using logic gates. (8)

(ii) What is a multiplexer? Implement the following Boolean function with 8 × 1 MUX and external gates

$F(A,B,C,D) = \sum m(1,3,4,11,12,13,14,15)$. (8)

13. (a) (i) A sequential circuit with two D flip flops A and B, input X and output Y is specified by the following next state and output equations

$A(t+1) = AX + BX$,

$B(t+1) = A'X$

$Y = (A+B)X'$.

Draw the logic diagram, derive state table and state diagram. (12)

(ii) Realize T flip-flop using JK flip-flop. (4)

Or

(b) (i) Design a synchronous decade counter using T flip flop and construct the timing diagram (8)

(ii) Design a mealy model of sequence detector to detect the pattern 1001. (8)

14. (a) Design an asynchronous sequential circuit (with detailed steps involved) that has 2 inputs x_1 and x_2 and one output z. The circuit is required to give an output z = 1 when x_1 =1, x_2 = 1 and x_1 = 1 being first. (16)

Or

(b) Show how to program the fusible links to get a 4 bit Gray code from the binary inputs using PLA and PAL and compare the design requirements with PROM. (16)

15. (a) (i) Write a VHDL program for 1 to 4 Demux using dataflow modelling. (8)

(ii) Write a VHDL program for Full adder using structural modelling. (8)

Or

(b) Explain in detail the RTL design procedure. (16)

STUCOR AP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Reg. No. : | | | | | | | | | | | | |

Question Paper Code : 77123

B.E./B.Tech. DEGREE EXAMINATION, APRIL/MAY 2015.

Third Semester

Electrical and Electronics Engineering

EE 6301 — DIGITAL LOGIC CIRCUITS

(Common to Electronics and Instrumentation Engineering and Instrumentation and Control Engineering)

(Regulation 2013)

Time : Three hours | Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. Convert :
   (a) $(475.25)_8$ to its decimal equivalent
   (b) $(549.B4)_{16}$ to its binary equivalent.

2. Define propagation delay.

3. Convert the given expression in canonical SOP form
   $Y = AC + AB + BC$.

4. Simplify the expression $Z = AB + A\overline{B}.\left(\overline{\overline{A}.\overline{C}}\right)$.

5. Convert T Flip Flop to D Flip Flop.

6. State the rules for state assignment.

7. State the difference between static 0 and static 1 hazard.

8. What is a PROM?

9. What is a package in VHDL?

10. Write the behavioral modeling code for a D Flip Flop.

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

PART B — (5 × 16 = 80 marks)

11. (a) (i) Perform the following addition using BCD and Excess-3 addition (205 + 569). (8)

(ii) Encode the binary word 1011 into seven bit even parity Hamming code. (8)

Or

(b) (i) With circuit schematic, explain the operation of a two input TTL NAND gate with totem-pole output. (10)

(ii) Compare totem pole and open collector outputs. (6)

12. (a) (i) Reduce the following function using K-map.

$f(A,B,C,D) = \pi M(0,2,3,8,9,12,13,15)$. (8)

(ii) Design a full adder using two half-adders and an OR gate. (8)

Or

(b) (i) Design a BCD to Excess 3 code converter. (8)

(ii) Implement the following Boolean function using 8:1 Mux :

$F(A,B,C,D) = \Sigma m(0,1,3,4,8,9,15)$. (8)

13. (a) (i) Explain the operation of a master slave JK flip flop. (8)

(ii) Design a 3-bit bidirectional shift register. (8)

Or

(b) (i) Design a MOD-5 synchronous counter using JK flip-flops. (8)

(ii) Design a sequence detector to detect the sequence 101 using JK flip flop. (8)

14. (a) Design an asynchronous sequential circuit that has two inputs X_2 and X_1 and one output Z. When $X_1 = 0$, the output Z is 0. The first change in X_2 that occurs while X_1 is 1 will cause output Z to be 1. The output Z will remain 1 until X_1 returns to 0.

Or

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

(b) (i) Implement the following function using PLA :

$F(x,y,z)=\Sigma m(1,2,4,6)$

(ii) For the given boolean function, obtain the hazard-free circuit

$F(A,B,C,D)=\Sigma m(1,3,6,7,13,15)$.

15. (a) Write the VHDL code to realize a full adder using

(i) Behavioral modeling

(ii) Structural modeling. (8 + 8)

Or

(b) Write the VHDL code to realize a 3-bit Gray code counter using case statement. (16)

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Reg. No. : 4 2 1 6 1 3 1 0 5 0 8 8

Question Paper Code : 97064

B.E./B.Tech. DEGREE EXAMINATION, NOVEMBER/DECEMBER 2014.

Third Semester

Electrical and Electronics Engineering

EE 6301 — DIGITAL LOGIC CIRCUITS

(Common to Electronics and Instrumentation Engineering and Instrumentation and Control Engineering)

(Regulation 2013)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. Determine $(377)_{10}$ in Octal and Hexa-Decimal equivalent.

2. Compare the totem-pole output with open-collector output?

3. Given F = B' + A' B + A' C': Identify the redundant term using K-Map.

4. Give one application each for Multiplexer and Decoder.

5. Show how the JK flip flop can be modified into a D flip flop or a T flipflop.

6. Differentiate between Mealy and Moore models.

7. What is a deadlock condition?

8. Draw the block diagram of PLA.

9. Write a VHDL code for 2×1 MUX.

10. State the advantage of package declaration over component declaration.

STUCOR
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

PART B — (5 × 16 = 80 marks)

11. (a) (i) Given that a frame with bit sequence 1101011011 is transmitted, it has been received as 1101011010. Determine the method of detecting the error using any one error detecting code. (8)

(ii) Draw the MOS logic circuit for NOT gate and explain its operation. (8)

Or

(b) (i) Explain Hamming code with an example. State its advantages over parity codes. (8)

(ii) Design a TTL logic circuit for a 3-input NAND gate. (8)

12. (a) (i) Minimize the function $F(a, b, c, d) = \Sigma(0, 4, 6, 8, 9, 10, 12)$ with $d = \Sigma(2, 13)$. Implement the function using only NOR gates. (8)

(ii) Design a Full Subtractor and implement it using logic gates. (8)

Or

(b) (i) Implement the function $F(p, q, r, s) = \Sigma(0, 1, 2, 4, 7, 10, 11, 12)$ using Decoder. (8)

(ii) Design a 4-bit Binary to gray code converter and implement it using logic gates. (8)

13. (a) (i) Design an asynchronous Modulo-8 Down counter using JK flipflops. (8)

(ii) Explain the circuit of a SR flip-flop and explain its operation. (8)

Or

(b) (i) Design synchronous sequential circuit that goes through the count sequence 1,3,4,5 repeatedly. Use T flip-flops for your design. (8)

(ii) Explain the various types of triggering with suitable diagrams. Compare their merits and demerits. (8)

14. (a) Explain the various types of hazards in sequential circuit design and the methods to eliminate them. Give suitable examples. (16)

Or

(b) Describe with reasons, the effect of races in asynchronous sequential circuit design. Explain its types with illustrations. Show the method of race-free state assignments with examples. (16)

STUCOR
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

15. (a) (i) Explain the digital system design flow sequence with the help of a flow chart. (8)

(ii) Write a VHDL code for a 4-bit universal shift register. (8)

Or

(b) Explain the concept of Behavioural modeling and Structural modeling in VHDL. Take the example of Full Adder design for both and write the coding. (16)

STUCOR APP

DOWNLOADED FROM STUCOR APP

STUCOR

DOWNLOADED FROM STUCOR APP

Reg. No. : 421611105102

Question Paper Code : 31399

B.E./B.Tech. DEGREE EXAMINATION, NOVEMBER/DECEMBER 2013.

Fourth Semester

Electrical and Electronics Engineering

EE 2255/EE 46/EC 1261 A/10133 EE 406 A/080280029 — DIGITAL LOGIC CIRCUITS

(Regulation 2008/2010)

(Common to PTEE 2255 – Digital Logic Circuits for B.E. (Part-Time) Third Semester Electrical and Electronics Engineering – Regulation 2009)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. What is meant by non weighted codes?
2. List the names of universal gates.
3. Differentiate sequential circuits and combinational circuits.
4. What is meant by state assignment?
5. What is meant by transition table?
6. Name the types of hazards.
7. What are the different classifications of memory?
8. Draw 4 × 2 ROM with AND-OR gates.
9. What is the function of wait statement in VHDL package?
10. Write the explanation of T'Base and T'Low predefined attributes.

39

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

PART B — (5 × 16 = 80 marks)

11. (a) (i) Express the function $F = A + \overline{B}C$ in
(1) Canonical SOP form and
(2) Canonical POS form. (8)

(ii) Design BCD to Excess 3 code converter. (8)

Or

(b) (i) Simplify using K-map

$F(A,B,C,D) = \sum m(7,8,9) + d(10,11,12,13,14,15)$ (8)

(ii) Design a full subtractor using half subtractors. (8)

12. (a) A sequential circuit has two JK flip flops A and B. The flip flop input functions are :

$J_A = B \quad J_B = \overline{x}$

$K_A = B\overline{x} \quad K_B = A \oplus x$

(i) Draw the logic diagram of the circuit (5)

(ii) Tabulate the state table (6)

(iii) Draw the state diagram. (5)

Or

(b) Using JK flip flops, design a synchronous counter which counts in the sequence 000, 001, 010, 011, 100, 101, 110, 111, 000.

13. (a) Derive the transition table and primitive flow table for the functional mode asynchronous sequential circuit shown in fig.13(a).

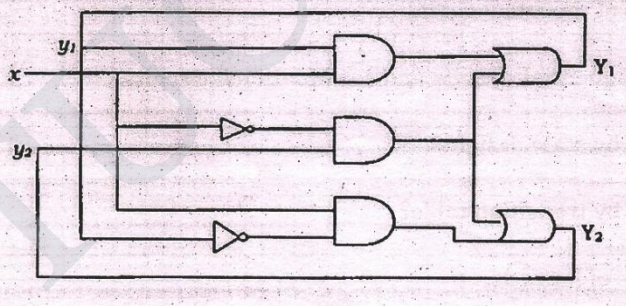


Fig. 13(a)

Or

(b) Design a pulse mode asynchronous sequential circuit, that has two inputs X_1 and X_2, one Mealy output Z_1 and one Moore output Z_2. The Mealy output is coincident with the third consecutive pulse on input X_2 and the Moore output occurs after the third consecutive pulse on input X_2 or after any pulse that occurs on input X_1.

40

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

14. (a) (i) Implement a $1M \times 4$ RAM using $512K \times 4$ RAM. (8)

(ii) Explain in details about TTL with open Collector output configuration. (8)

Or

(b) (i) Implement the following functions using PLA

$F_1 = \sum m(1,2,4,6)$; $F_2 = \sum m(0,1,6,7)$; $F_3 = \sum m(2,6)$. (8)

(ii) Demonstrate the CMOS logic circuit configuration and characteristics in details. (8)

15. (a) Explain the structural VHDL description for a 2 to 4 decoder in details.

Or

(b) Write a VHDL program and explain the design procedure of 8 bit comparator.

———————

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Reg. No. : 0 4 1 2 0 8 1 0 5 3 3 4

Question Paper Code : 21399

B.E./B.Tech. DEGREE EXAMINATION, MAY/JUNE 2013.

Fourth Semester

Electrical and Electronics Engineering

EE 2255/EE 46/EC 1261 A/10133 EE 406 A/080280029 — DIGITAL LOGIC CIRCUITS

(Regulation 2008/2010)

(Common to PTEE 2255 – Digital Logic Circuits for B.E. (Part-Time) Third Semester Electrical and Electronics Engineering – Regulation 2009)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. Construct OR gate using only NAND gates.
2. Define multiplexer.
3. Draw truth table for JK flipflop.
4. How many flip flops are required to design mod 25 counter?
5. What is the difference between flow table and transition table?
6. Define Race conditions in Asynchronous sequential circuit.
7. What is the difference between PROM and EPROM?
8. What are the advantages of CMOS?
9. Write HDL behavioural model of D flip flop.
10. What is the need for VHDL?

42

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

PART B — (5 × 16 = 80 marks)

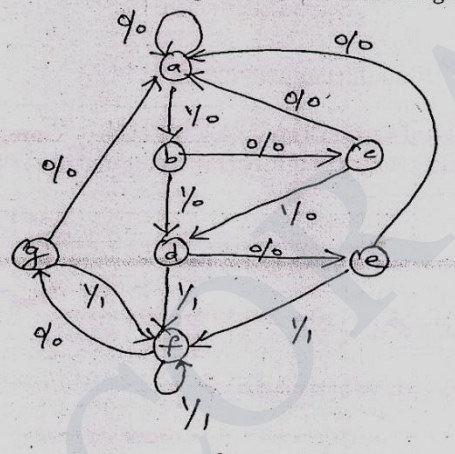
11. (a) (i) Prove that $F=\overline{A}\cdot B+A\cdot\overline{B}$ is exclusive OR operation and it equals $=\overline{\overline{(A\cdot B)}\cdot A}\cdot\overline{\overline{(A\cdot B)}\cdot B}$. (8)

(ii) Prove that for constructing XOR from NANDs we need four NAND gates. (8)

Or

(b) Simplify the Boolean function using Kmap $F(w,x,y,z)=\Sigma(1,3,7,11,15)$. which has the don't care conditions $d(w,x,y,z)=\Sigma(0,2,5)$.

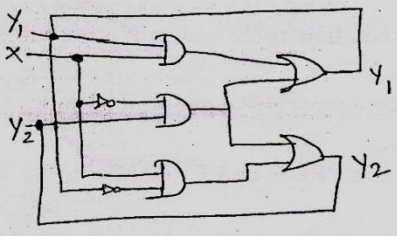
12. (a) Construct reduced state diagram for the following state diagram.



Or

(b) Design a 3 bit binary counter using T flip flop.

13. (a) Consider the following asynchronous sequential circuit and draw maps and transition table, and state table.



Or

(b) Illustrate the analysis procedure of asynchronous sequential circuit with an example.

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

14. (a) Illustrate the basic principles of PLA and FPGA.

Or

(b) Explain the basic working principles of TTL and ECL logic families.

15. (a) Write HDL for four bit binary counter with parallel load and explain.

Or

(b) (i) Write HDL for two to one line multiplexer with data flow description and behavioural description. (8)

(ii) Write HDL for four bit adder. (8)

STUCOR APP

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

Reg. No. : 4 1 2 0 8 1 0 5 3 3 4

Question Paper Code : 11367

B.E./B.Tech. DEGREE EXAMINATION, NOVEMBER/DECEMBER 2012.

Fourth Semester

Electrical and Electronics Engineering

EE 2255/131405/EE 46/EC 1261 A/10133 EE 406 A/080280029 — DIGITAL LOGIC CIRCUITS

(Regulation 2008)

(Common to PTEE 2255 — Digital Logic Circuits for B.E. (Part-Time) Third Semester Electrical and Electronics Engineering – Regulation 2009)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. State De Morgan's theorems.
2. Sketch a half-adder using logic gates.
3. The JK flip-flop is an universal flip-flop. Justify.
4. What are state table and state diagram as applicable to sequential logic circuits?
5. Sketch the block diagram of an asynchronous sequential circuit.
6. What is a 'race condition' in an asynchronous sequential circuit?
7. Compare ROM, PROM and EPROM memory devices.
8. What are the advantages of PLAs?
9. List the different types of 'operators' supported by VHDL.
10. The 'module' is the basic building block of VHDL. What are the different modelling techniques used to describe a module?

PART B — (5 × 16 = 80 marks)

11. (a) (i) Show that a function expressed as a sum of its min terms is equivalent to a function expressed as a product of its max terms. (8)

(ii) Using K-map simplify the following function and implement the function using logic gates $f(A, B, C) = \Pi(0, 4, 6)$. (8)

Or

(b) (i) Design a 4-bit binary ripple carry adder with full adders and discuss its operation. (8)

(ii) Implement the following function $f(A, B, C) = \Sigma m(0, 3, 5)$ using and 8:1 multiplexer. (8)

45

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

12. (a) (i) Design a 4-bit decade counter. (10)

(ii) Sketch the state diagram and the state table for 'D' and 'JK' flip flops. (6)

Or

(b) (i) Differentiate between analysis and design of sequential logic circuits. (4)

(ii) Design a sequential logic circuit for a 3-bit binary counter. (12)

13. (a) (i) Sketch the transition table and state table for an asynchronous sequential circuit described by the following Boolean expressions : (8)

$Y_1 = x\,y_1 + x'y_2$

$Y_2 = x\,y_1' + x'y_2$

(ii) Discuss the steps involved in the design of an asynchronous sequential circuit with a suitable example. (8)

Or

(b) (i) What are hazards in sequential circuits? How can they be eliminated? (6)

(ii) Draw the logic diagram for the product of sums expression given by $Y = (x_1 + x_2')(x_2 + x_3)$. Show that there is a static 0-hazard when x_1 and x_3 are equal to 0 and x_2 goes from 0 to 1. Find a way to remove the hazard by adding one more OR gate. (10)

14. (a) (i) How is memory modelled in VHDL? Write a VHDL code that illustrates the read and write operations of memory. (8)

(ii) Design and implement a 4-bit binary to Gray code converter using a PLA. (8)

Or

(b) (i) Compare the characteristics of TTL, ECL and CMOS logic families. (6)

(ii) Design an AND-OR-PLA that implements the functions (10)

$f(x, y, z) = \Sigma m\,(0, 2, 4, 6)$

$g(x, y, z) = \Sigma m\,(1, 3, 5, 7)$

15. (a) (i) Briefly discuss the different data types supported in VHDL. (6)

(ii) Write the HDL description of the circuit specified by the following Boolean functions :

$x = A + BC + B'D$

$y = B'C + BC'D'$

Use continuous assignment statements. (10)

Or

(b) (i) Briefly discuss the use of 'Packages' in VHDL. (6)

(ii) Write an HDL code that implements an 8:1 multiplexer. (10)

———————

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

15/5/12 FN

Reg. No. : 4 1 2 0 9 1 0 5 0 4 4 X

Question Paper Code : 10323

B.E./B.Tech. DEGREE EXAMINATION, MAY/JUNE 2012.

Fourth Semester

Electrical and Electronics Engineering

EE 2255/131405/EE 46/EC 1261 A/10133 EE 406/080280029 – DIGITAL LOGIC CIRCUITS

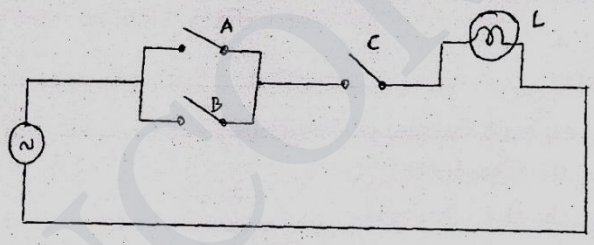
(Regulation 2008)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. Express the following switching circuit in binary logic notation.



2. What is priority encoder?
3. Give the characteristic equation and state diagram of JK flip-flop.
4. What is lockout? How it is avoided?
5. How does the operation of an asynchronous input differ from that of a synchronous input?
6. Define flow table in asynchronous sequential circuit.
7. What is a PLA?
8. List the configurable elements in the FPGA architecture.
9. What are the various modeling techniques in HDL?
10. Write HDL for half adder.

47

STUCOR APP
DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

PART B — (5 × 16 = 80 marks)

11. (a) (i) Implement the following Boolean function with NAND - NAND logic.

$Y = AC + ABC + \overline{A}BC + AB + D$ (6)

(ii) Simplify and implement the following sop function using NOR gates.

$f(A,B,C,D) = \sum m(0, 1, 4, 5, 10, 11, 14, 15)$. (10)

Or

(b) (i) Implement the given function using multiplexer $F(x, y, z) = \sum(0, 2, 6, 7)$. (8)

(ii) Implement full subtractor using demultiplexer. (8)

12. (a) (i) Realize SR flip-flop using NOR gates and explain its operation. (8)

(ii) Convert a SR flip-flop into JK flip-flop. (8)

Or

(b) A sequential circuit with 2D FFs A and B and input X and output Y is specified by the following next state and output equations.

$A(t+1) = AX + BX$

$B(t+1) = A'X$

$Y = (A + B)X'$

(i) Draw the logic diagram of the circuit.

(ii) Derive the state table.

(iii) Derive the state diagram. (16)

13. (a) (i) Design a pulse mode circuit with inputs x_1, x_2, x_3 and output Z as shown in figure 1.

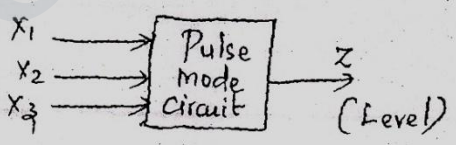


Figure 1

(ii) The output should change from 0 to 1, only for input sequence $x_1 - x_2 - x3$ occurs while $z = 0$. Also the output z should remain in 1 until x_2 occurs. Use SR flip-flops for the design. (16)

Or

STUCOR APP

DOWNLOADED FROM STUCOR APP

DOWNLOADED FROM STUCOR APP

(b) (i) List and explain the steps used for analyzing an asynchronous sequential circuit. (8)

(ii) Derive the flow table for the circuit given in the figure 2. (8)

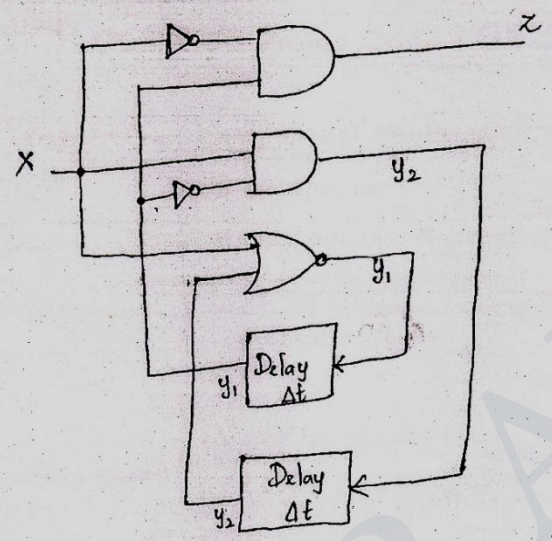


Figure 2

14. (a) Write notes on ROM and its types. (16)

Or

(b) (i) A combinational logic circuit is defined by the following function.

$f_1(a,b,c) = \Sigma(0, 1, 6, 7)$, $f_2(a, b, c) = \Sigma(2, 3, 5,7)$

Implement the circuit with a PAL having three inputs, three product terms and two outputs. (10)

(ii) Describe the concept and working of FPGA. (6)

15. (a) Explain RTL design using VHDL with the help of example. (16)

Or

(b) Write the VHDL code for mod 6 counter. (16)

———————

STUCOR APP

DOWNLOADED FROM STUCOR APP