ANNA UNIVERSITY QB

UNIT I - BOOLEAN ALGEBRA AND LOGIC GATES

PART A – 2 MARKS

1. Define binary logic

Binary logic consists of binary variables and logical operations. The variables are designated by the alphabets such as A, B, C, x, y, z, etc., with each variable having only two distinct values: 1 and 0.

2. What is meant by Digital Systems?

A digital system is a system that manipulates discrete elements of information that is represented by binary form i.e. 0's and 1's.

3. List the number systems

i) Decimal Number system
ii) Binary Number system
iii) Octal Number system
iv) Hexadecimal Number system

4. Convert 736_8 into an equivalent binary number.

The binary equivalents of 7, 3 and 6 are 111,011 & 110 respectively.
Therefore $736_8 = 111011110_2$

5. Which gates are called as the universal gates? What are its advantages?

The NAND and NOR gates are called as the universal gates. These gates are used to perform any type of logic application.

6. State the sequence of operator precedence in Boolean expression?

- Parenthesis
- AND
- OR

7. What are the different types of number complements?

- Radix Complement
- Diminished Radix Complement

8. Why complementing a number representation is needed.

Complementing a number becomes as in digital computer for simplifying the subtraction operation and for logical manipulation complements are used.

9. How to represent a positive and negative sign in computers

- Positive (+) sign by 0
- Negative (-) sign by 1.

10. What is meant by Map method?

The map method provides a simple straightforward procedure for minimizing Boolean function.

11. What is meant by two variable map?

Two variable map have four minterms for two variables, hence the map consists of four squares, one for each minterm.

12. State Duality principle.

The dual of any Boolean function can be obtained by changing each OR sign to an AND sign and vice versa and complementing any 0 or 1 appearing in the expression.

13. Why parity checker is needed?

Parity checker is required at the receiver side to check whether the expected parity is equal to the calculated parity or not. If they are not equal then it is found that the received data has error.

14. What is meant by parity bit?

Parity bit is an extra bit included with a binary message to make the number of 1'seither odd or even. The message, including the parity bit is transmitted and then checked at the receiving and for errors.

15. What are the needs for binary codes?

- Code is used to represent letters, numbers and punctuation marks.
- Coding is required for maximum efficiency in single transmission.
- Binary codes are the major components in the synthesis (artificial generation) of speech and video signals.
- By using error detecting codes, errors generated in signal transmission can be detected.
- Codes are used for data compression by which large amounts of data are transmitted in very short duration of time.

16. Mention the different type of binary codes?

- Binary weighted code
- Binary non - weighted code
- Sequential code
- Alphanumeric code
- Error-detecting and error-correcting code

17. List the advantages and disadvantages of BCD code?

Advantages of BCD code:

a. Any large decimal number can be easily converted into corresponding binary number
b. A person needs to remember only the binary equivalents of decimal number from 0 to 9.
c. Conversion from BCD into decimal is also very easy.

Disadvantages of BCD code:

a. The code is least efficient. It requires several symbols to represent even small numbers.
b. Binary addition and subtraction can lead to wrong answer.
c. c. Special codes are required for arithmetic operations.
d. This is not a self-complementing code.
e. Conversion into other coding schemes requires special methods.

18. What is meant by self-complementing code?

A self-complementing code is the one in which the members of the number system complement on themselves. This requires the following two conditions to be satisfied.

- The complement of the number should be obtained from that number by replacing 1s with0s and 0s with 1s.
- The sum of the number and its complement should be equal to decimal 9.

Example of a self-complementing code is

i) 2-4-2-1 code. ii). Excess-3 code

19. Mention the advantages of ASCII code

- There are 27 =128 possible combinations. Hence, a large number of symbols, alphabets etc.., can be easily represented.
- There is a definite order in which the alphabets, etc.., are assigned to each code word.
- The parity bits can be added for error-detection and correction.

20. What are the disadvantages of ASCII code?

- The length of the code is larger and hence more bandwidth is required for transmission.
- With more characters and symbols to represent, this is not sufficient.

21. Application of octal number system:

It is highly in convenient to handle long strings of binary numbers while entering into the digital systems. It may cause errors also. Therefore, octal numbers are used for entering binary data and displaying certain information.

22. What is a Logic gate?

Logic gates are the basic elements that make up a digital system. The electronic gate is a circuit that is able to operate on a number of binary inputs in order to perform a particular logical function.

23. List out the advantages and disadvantages of K-map method?

The advantages of the K-map method are

- It is a fast method for simplifying expression up to four variables.
- It gives a visual method of logic simplification.
- Prime implicants and essential prime implicants are identified fast.
- Suitable for both SOP and POS forms of reduction.
- It is more suitable for class room teachings on logic simplification.

The disadvantages of the K-map method are

- It is not suitable for computer reduction.
- K-maps are not suitable when the number of variables involved exceed four.
- Care must be taken to fill in every cell with the relevant entry, such as a 0, 1 (or) don't care terms.

24. Obtain truth table and name operation performed for A'B+AB'

PART B – 16 MARKS

1. a. Explain the various types of K-Map with Examples

(12)

b. Prove that x + 1 = 1

c. Prove that x + xy = x (2)

2. a. Simply the Boolean Function Using Three Variable K-Map

F(X, Y, Z) = ∑ (3, 4, 6, 7) (8)

b. Simply the Boolean Function Using Four Variable K-Maps

F(W,X,Y,Z) = ∑ (0,1,2,4,5,6,8,9,12,13,14) (8)

34. a. Explain logic operations with NAND Gates? (8)

b. Explain Multilevel NAND Gates? (8)

4. a. Explain Implementation of NOR Gates ? (8)

b. Explain AND- OR Invert Implementation (8)

5. a. Explain BCD Code with Examples (6)

b. Explain Excess 3 Code with Examples? (6)

c. Convert the number (28) Decimal to Excess 3 Code (4)

6. a. List out the Procedure for converting Binary to Gray Code (4)

b. Convert the number (1011) binary to gray? (4)

c. Explain 7 - Bit ASCII Code ? (8)

7. simplify the Boolean function using tabulation method (16)

F= sum (0,1,4,11,15,32,29,28)

UNIT II - COMBINATIONAL LOGIC

Part A – 2 Marks

.

1. Define Combinational circuit

A combinational circuit consist of logic gates whose outputs at anytime are determined directly from the present combination of inputs without regard to previous inputs.

5. Explain the design procedure for combinational circuits

x Determine the number of available input variables & required O/P variables. x Assigning letter symbols to I/O variables

x Obtain simplified Boolean expression for each O/P. x

Obtain the logic diagram.

6. What is code conversion?

If two systems working with different binary codes are to be synchronized in operation, then we need digital circuit, which converts one system of codes to the other. The process of conversion is referred to as code conversion.

4. 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. One example is a BCD to Xs3 converter.

8. Analysis procedure for combinational circuits

- x Find the given circuit is combinational or sequential.
- x Combinational circuit has a logic gate with no feedback paths or memory elements.
- x A feedback path is a connection from the output of one gate to the input of second gate that forms part of the input to the first gate

9. Design procedure for combinational circuits

Determine the required number of inputs and outputs and assign a symbol to each.

Derive the truth table that defines the required relationship between inputs and outputs.

Obtain the simplified Boolean functions for each output as a function of the input variables.

Draw the logic diagram and verify the correctness of the design.

10. What is a half-adder?

The combinational circuit that performs the addition of two bits is called a half-adder.

8. What is a full-adder?

The combinational circuit that performs the addition of three bits is called a full-adder.

The combinational circuit that performs the subtraction of two bits is called a half-subtractor.

10. What is a full-subtractor?

The combinational circuit that performs the subtraction of three bits is called a half-subtractor.

11. What is Binary parallel adder?

A binary parallel adder is a digital function that produces the arithmetic sum of two binary numbers in parallel.

11. Logic equation for half adder

S=X † Y

C=X.Y

12. Limitations of Half-adder

In multidigit addition, add two bits along with the carry of previous digit addition. Effectively such addition requires addition of three bits. This is not possible with half adder. Hence, half-adders are not used in practice.

16. Limitations of Half-adder

In multidigit subtraction, subtract two bits along with the borrow of previous digit subtraction. Effectively such subtraction requires subtraction of three bits. This is not possible with half subtractor.

17. Define Hardware Description Language (HDL)

The size and complexity of the digital systems increases, they cannot be designed manually; their design is highly complex. At the most detailed level, they may consists of millions of elements i.e.) transistor or logic gates. So the computer-aided tools are used to design the Hardware Description Language.

18. Structure of Verilog module

module <module name> <port list>;

<declares>

<module items>

endmodule

f. Operators in Verilog HDL

x Boolean logical

Unary reduction logical

Bitwise logical

Relational

Binary arithmetic

Unary arithmetic

x **What are the Verilog data types?** reg and wire

reg variables store the last value that was procedurally assigned to them whereas the wire variables represent physical connections between structural entities such as gates.

21. What is BCD adder?

A BCD adder is a circuit that adds two BCD digits in parallel and produces a sum digit also in BCD.

22. What is Magnitude Comparator?

A Magnitude Comparator is a combinational circuit that compares two numbers, A and B and determines their relative magnitudes.

23. What is decoder?

A decoder is a combinational circuit that converts binary information from 'n' input lines to a maximum of 2^n unique output lines.

24. What is encoder?

An encoder is a combinational circuit that converts binary information from 2^n Input lines to a maximum of 'n' unique output lines.

25. Define Multiplexing

Multiplexing means transmitting a large number of information units over a smaller number of channels or lines.

26. What is Demultiplexer?

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

27. What is the function of the enable input in a Multiplexer?

The function of the enable input in a MUX is to control the operation of the unit.

28. Give the applications of Demultiplexer.

Multiplexing means transmitting a large number of information units over a smaller number of channels or lines.

29. What is priority encoder?

A priority encoder is an encoder that includes the priority function. The operation of the priority encoder is such that if two or more inputs are equal to 1 at the same time, the input having the highest priority will take precedence.

2 Can a decoder function as a Demultiplexer?

i It finds its application in Data transmission system with error detection.

ii One simple application is binary to Decimal decoder.

3 Mention the uses of Demultiplexer

Demultiplexer is used in computers when a same message has to be sent to different receivers. Not only in computers, but any time information from one source can be fed to several

20. List basic types of programmable logic devices.

. Read only memory
. Programmable logic Array
. Programmable Array Logic

21. List out the applications of multiplexer

The various applications of multiplexer are
a. Data routing.
b. Logic function generator.
c. Control sequencer.
Parallel-to-serial converter.

List out the applications of decoder

The applications of decoder are
Decoders are used in counter system.
They are used in analog to digital converter.
Decoder outputs can be used to drive a display system.

25. Give other name for Multiplexer and Demultiplexer.

- Multiplexer is otherwise called as Data selector.
- Demultiplexer is otherwise called as Data distributor.

26. What is logic synthesis? (May/June 2011)

27. What are the modeling techniques in HDL? May/June 2013,2012

28. Give the need for using carry look ahead adder (nov/dec 2011)

- Ans: To reduce the carry propagation delay and to reduce the complexity in designing combimnational circuis

39. Construct 4x16 decoder using 3x8 decoders. (Nov/dec2012)

40. Implement full adder using 2 half adders (Nov/dec2012)

41. Draw the truthtable for BCD to excess 3 code (Nov/Dec2013)

Part B - 16 Marks

1. a. Explain the Design procedure for Combination Logic Circuits (6)

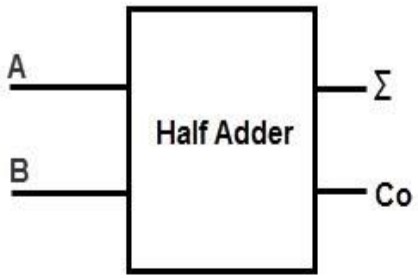
Design procedure

The design procedure for combinational logic circuits starts with the problem specification and comprises the following steps:

1. Determine required number of inputs and outputs from the specifications.
2. Derive the truth table for each of the outputs based on their relationships to the input.
3. Simplify the boolean expression for each output. Use Karnaugh Maps or Boolean algebra.
4. Draw a logic diagram that represents the simplified Boolean expression. Verify the design by analysing or simulating the circuit.

b. Explain the Logic implementation of half–adder and half-subtractor (10)

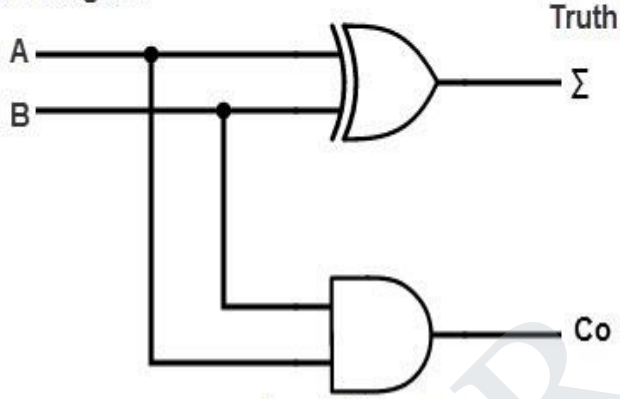
Half Adder



Block Diagram

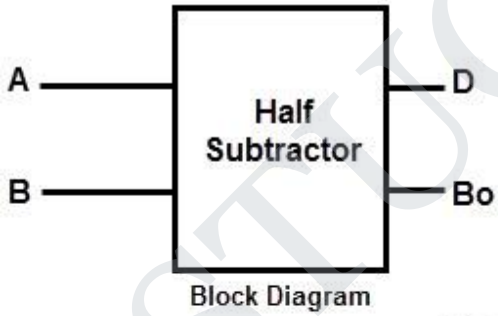
A	B	Sum (∑)	Carry Out (Co)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Truth Table



Logic Diagram

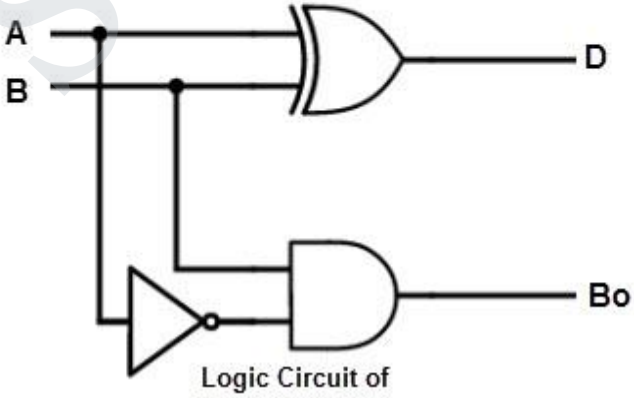
Half Subtractors



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



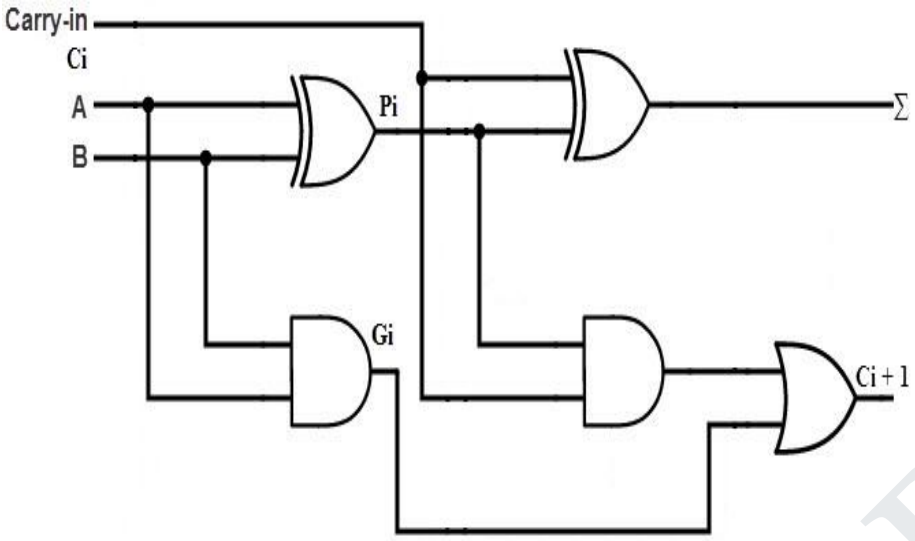
Logic Circuit of
Half Subtractor

2. a. Explain

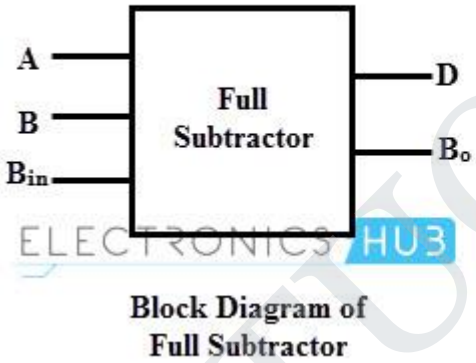
Logical Implementation of Full – adder and Full – Subtractor

(6)

Full Adder



Full Subtractor



Block Diagram of Full Subtractor

A	B	B_{in}	D	B_o
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

Truth Table

Full adder can be formed by combining two half adders and an OR gate as shown in above where output and carry-in of the first adder becomes the input to the second half adder that produce the total sum output. The total carry out is produced by ORing the two half adder carry outs as shown in figure. The full adder block diagram and truth table is shown below.

Block Diagram of Full Adder

C_{in}	B	A	$\sum$	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Truth Table

b. Draw the Logic Diagram for BCD to Excess 3 code Converter with Explain

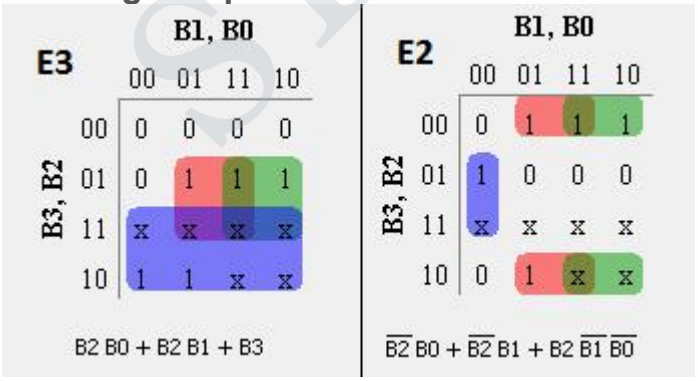
Truth Table relating BCD and Excess-3 codes

Decimal	BCD inputs				Excess-3 Code outputs			
	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

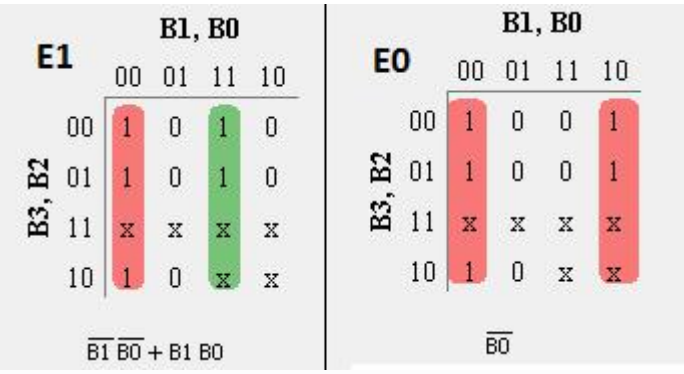
Boolean expression for each Excess-3 code bits

```
E3= m(5, 6, 7, 8, 9)
E2= m(1, 2, 3, 4, 9)
E1= m(0, 3, 4, 6, 7, 8)
E0= m(0, 2, 4, 6, 8)
```

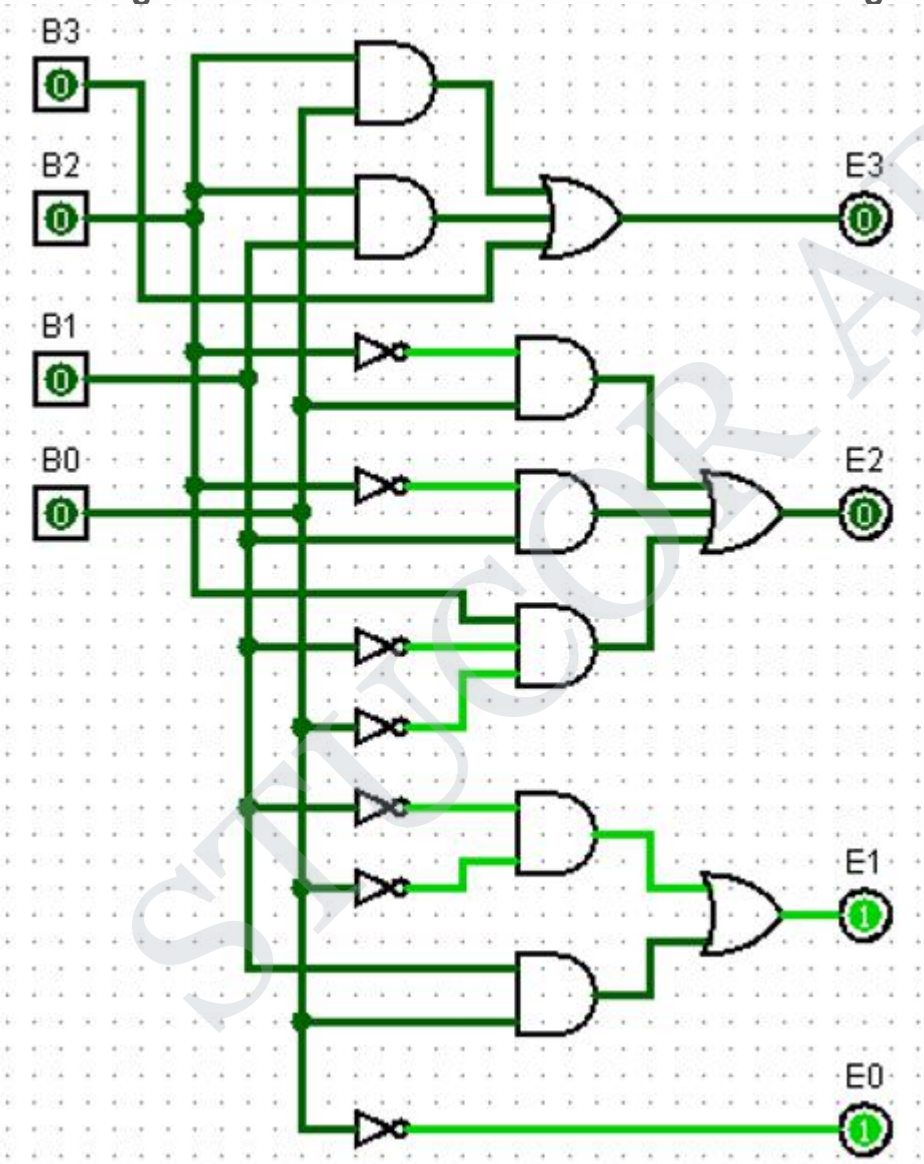
Karnaugh-Map solution for each output



(6)



Realizing code conversion using Logic Gates

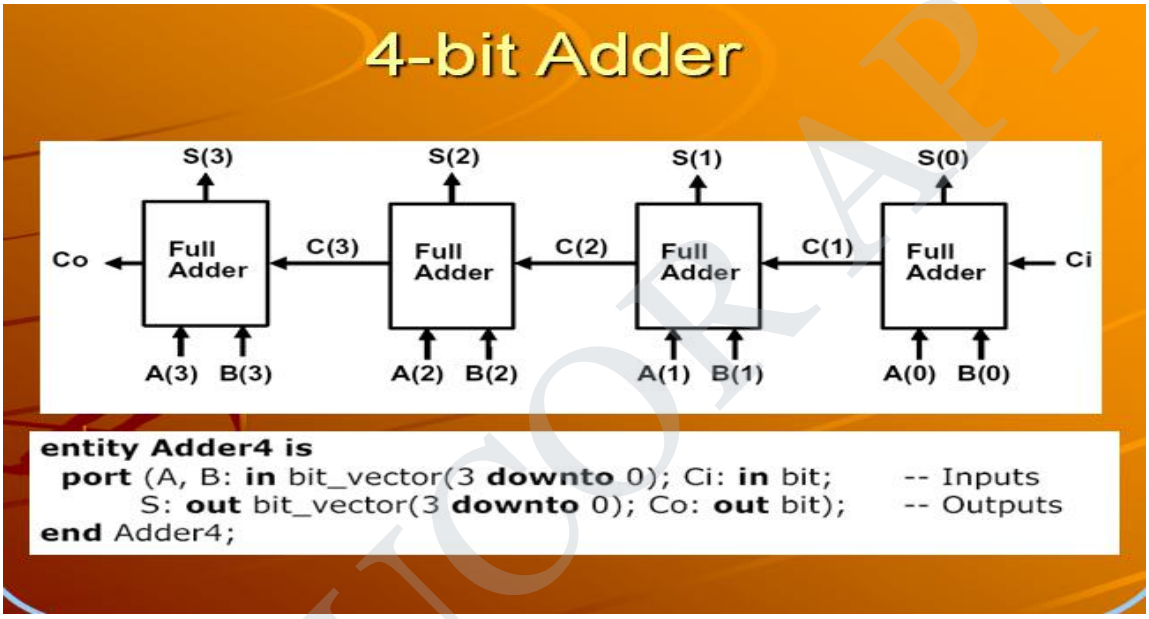


3. a. Explain the analysis procedure for combinational circuit (4)

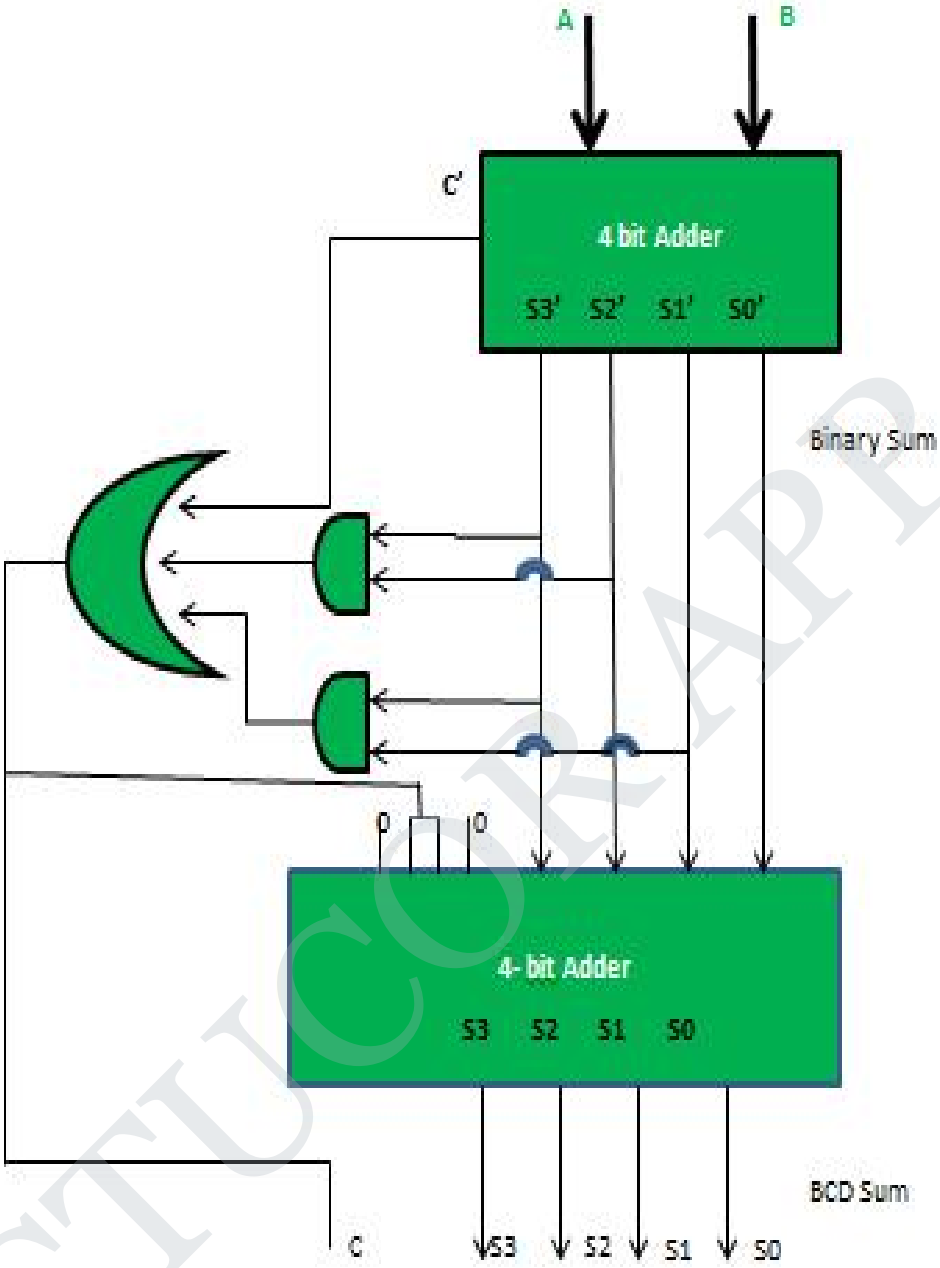
ANALYSIS PROCEDURE OF COMBINATIONAL CIRCUIT

- TO DETERMINE THE OUTPUT FUNCTIONS AS ALGEBRAIC EXPRESSIONS.
- It is the reverse process of design procedure.
- Logic diagram of the circuit is given.
- Obtain the truth table from the diagram.
- Obtain Boolean function from the Truth Table for output.

b. Explain the 4- bit Full adder



c. Explain the Block Diagram of BCD Adder (6)



4. a. Explain the 4 – Bit Magnitude Comparator (10)

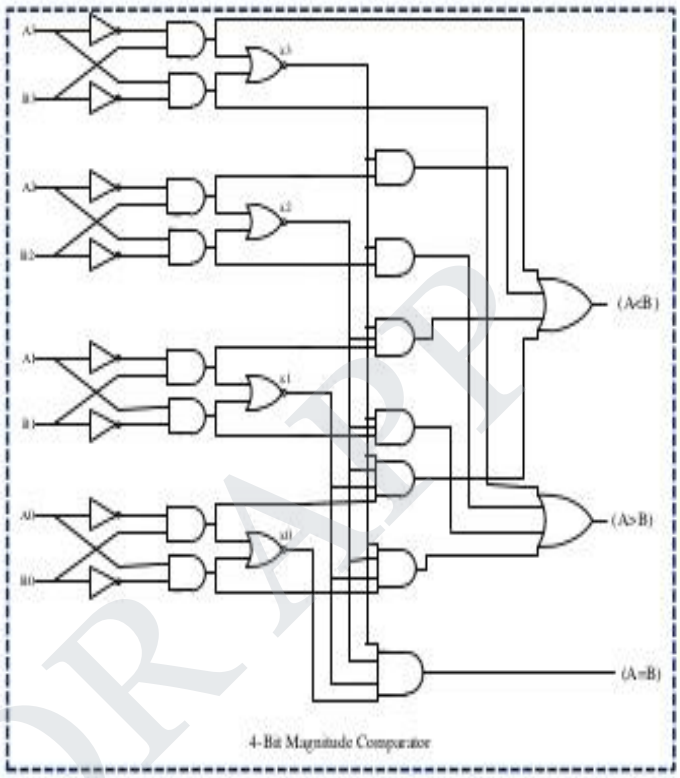
(6)

4-BIT COMPARATOR

The procedure for binary numbers with more than 2 bits can also be found in the similar way. Figure shows the 4-bit magnitude comparator.

Input: $A=A_3A_2A_1A_0$

$B=B_3B_2B_1B_0$



4-Bit Magnitude Comparator

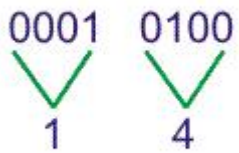
6. a. Explain the Binary to BCD Convertor (10)

Binary to BCD Code Converter

August 9, 2018 by Electrical4U

BCD is binary coded decimal number, where each digit of a decimal number is respected by its equivalent binary number. That means, LSB of a decimal number is represented by its equivalent binary number and similarly other higher significant bits of decimal number are also represented by their equivalent binary numbers.

For example, BCD Code of 14 is-

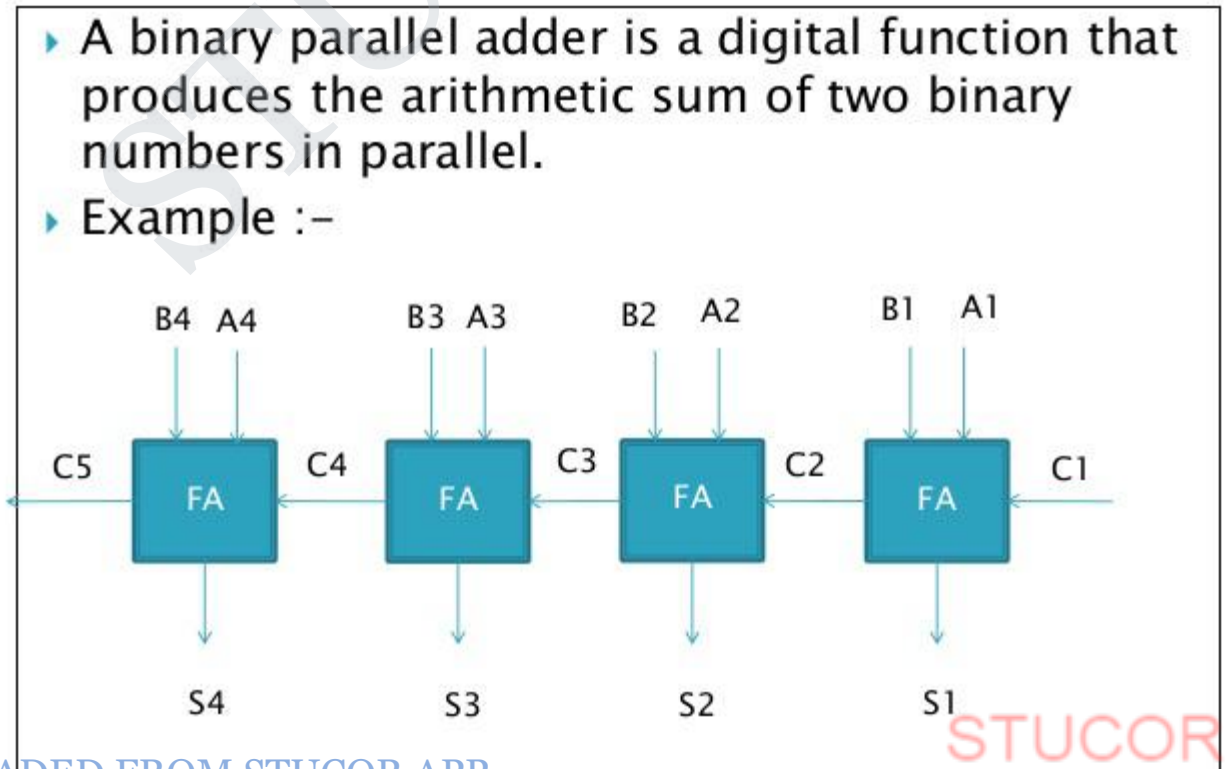


Let us design a 4bit **binary to BCD code converter**. As the 4 bit can represent 0 to 15, we can draw the conversion table as follows, (6)

Binary Code A B C D	Decimal Number	BCD Code B_5 B_4 B_3 B_2 B_1
0 0 0 0	0	0 0 0 0 0
0 0 0 1	1	0 0 0 0 1
0 0 1 0	2	0 0 0 1 0
0 0 1 1	3	0 0 0 1 1
0 1 0 0	4	0 0 1 0 0
0 1 0 1	5	0 0 1 0 1
0 1 1 0	6	0 0 1 1 0
0 1 1 1	7	0 0 1 1 1
1 0 0 0	8	0 1 0 0 0
1 0 0 1	9	0 1 0 0 1
1 0 1 0	10	1 0 0 0 0
1 0 1 1	11	1 0 0 0 1
1 1 0 0	12	1 0 0 1 0
1 1 0 1	13	1 0 0 1 1
1 1 1 0	14	1 0 1 0 0
1 1 1 1	15	1 0 1 0 1

Here, B_5 bit represents MSB of decimal number and B_4, B_3, B_2, B_1 represents 4 bit binary equivalent of LSB of decimal number.

b. Explain the Binary Parallel adder



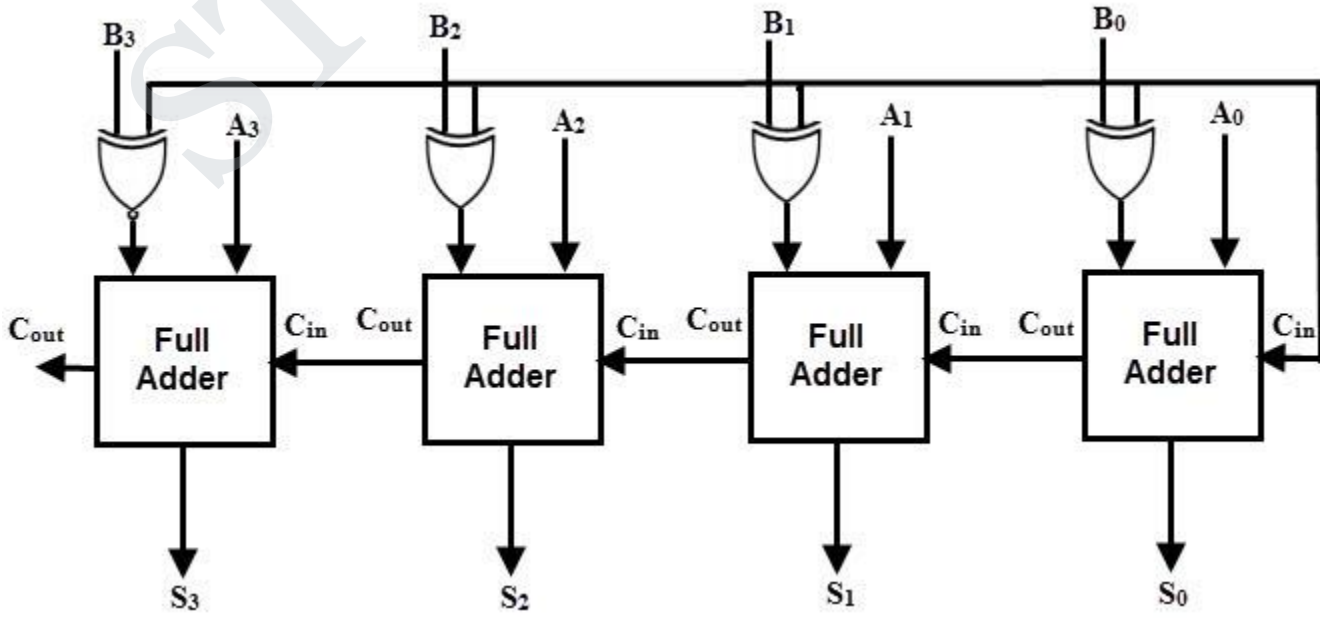
7. a. Explain the excess 3 to BCD Code Converter

THE BCD TO EXCESS 3 CODE CONVERTER

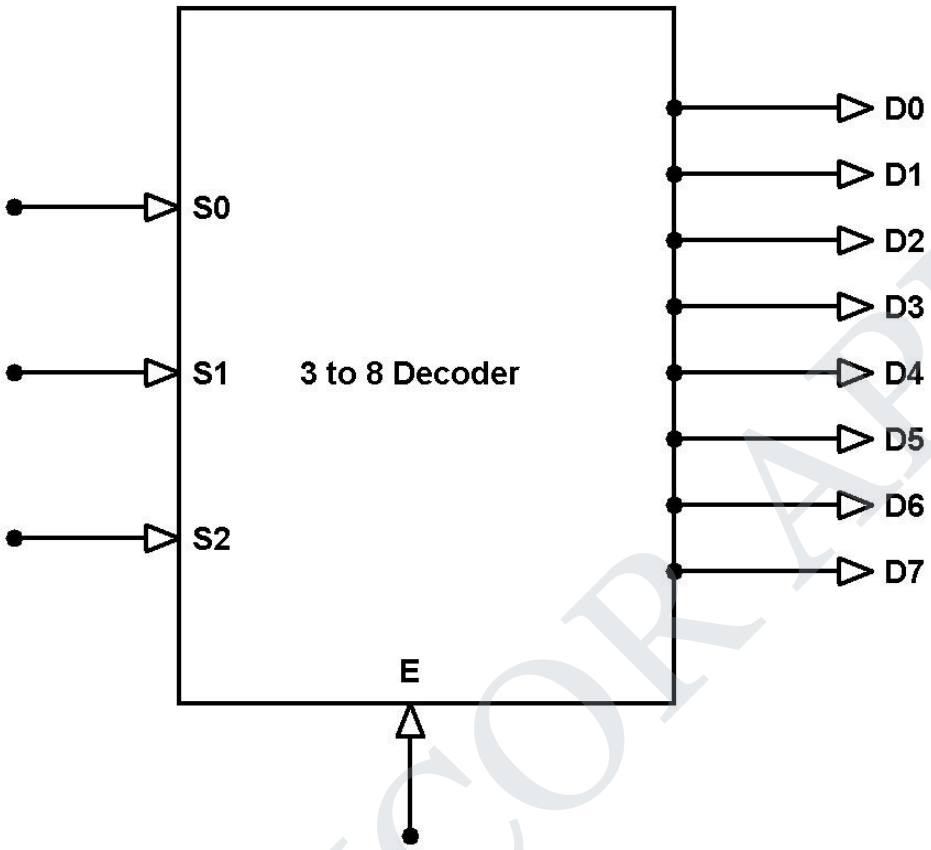
- BCD Excess-3 circuit will convert numbers from their binary representation to their excess-3 representation. Hence our truth table is as below:

B3	B2	B1	B0		E3	E2	E1	E0
0	0	0	0		0	0	1	1
0	0	0	1		0	1	0	0
0	0	1	0		0	1	0	1
0	0	1	1		0	1	1	0
0	1	0	0		0	1	1	1
0	1	0	1		1	0	0	0
0	1	1	0		1	0	0	1
0	1	1	1		1	0	1	0
1	0	0	0		1	0	1	1
1	0	0	1		1	1	0	0

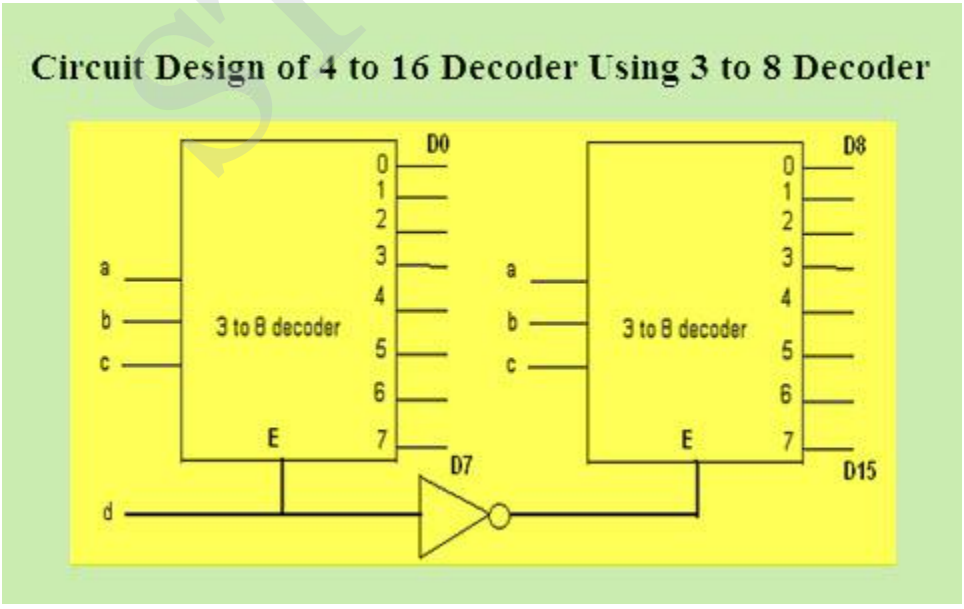
b. Explain the Binary Adder- Subtractor (6)

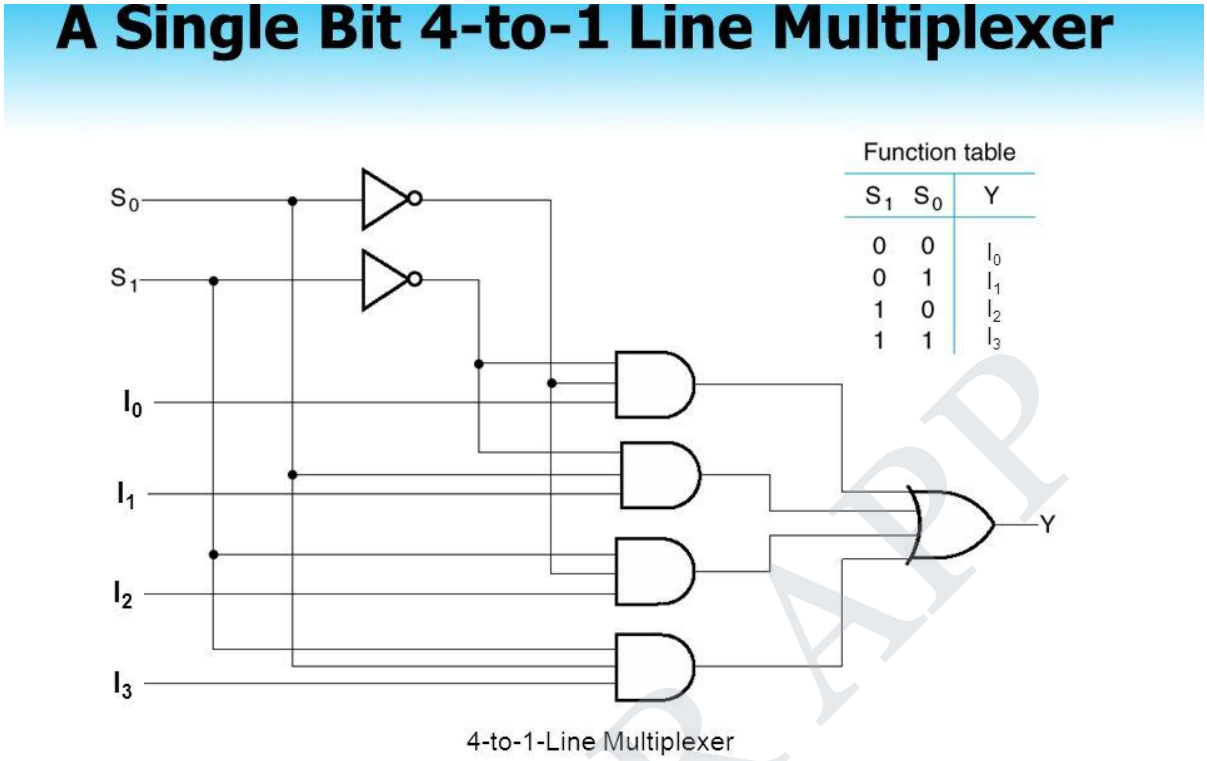


8. a. Explain the Logic Diagram of 3 to 8 line Decoder



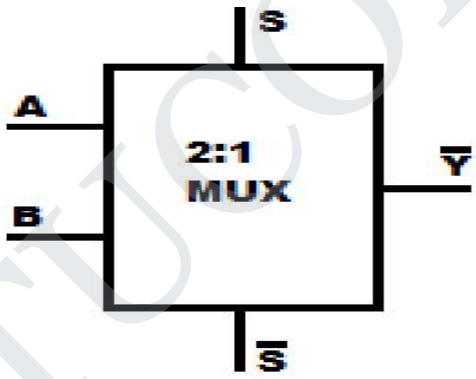
b. How to Construct the 4 x 16 Decoder with two 3 x 8 Decoder (8)





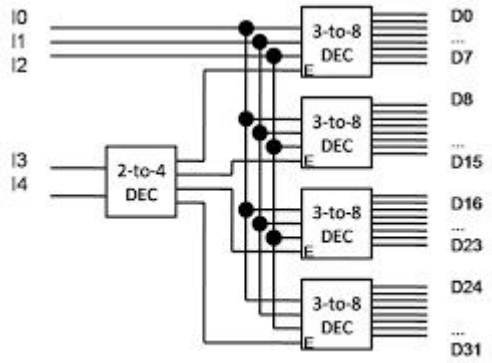
4-to-1-Line Multiplexer

b. Explain the 2 to 1 line Multiplexer (8)



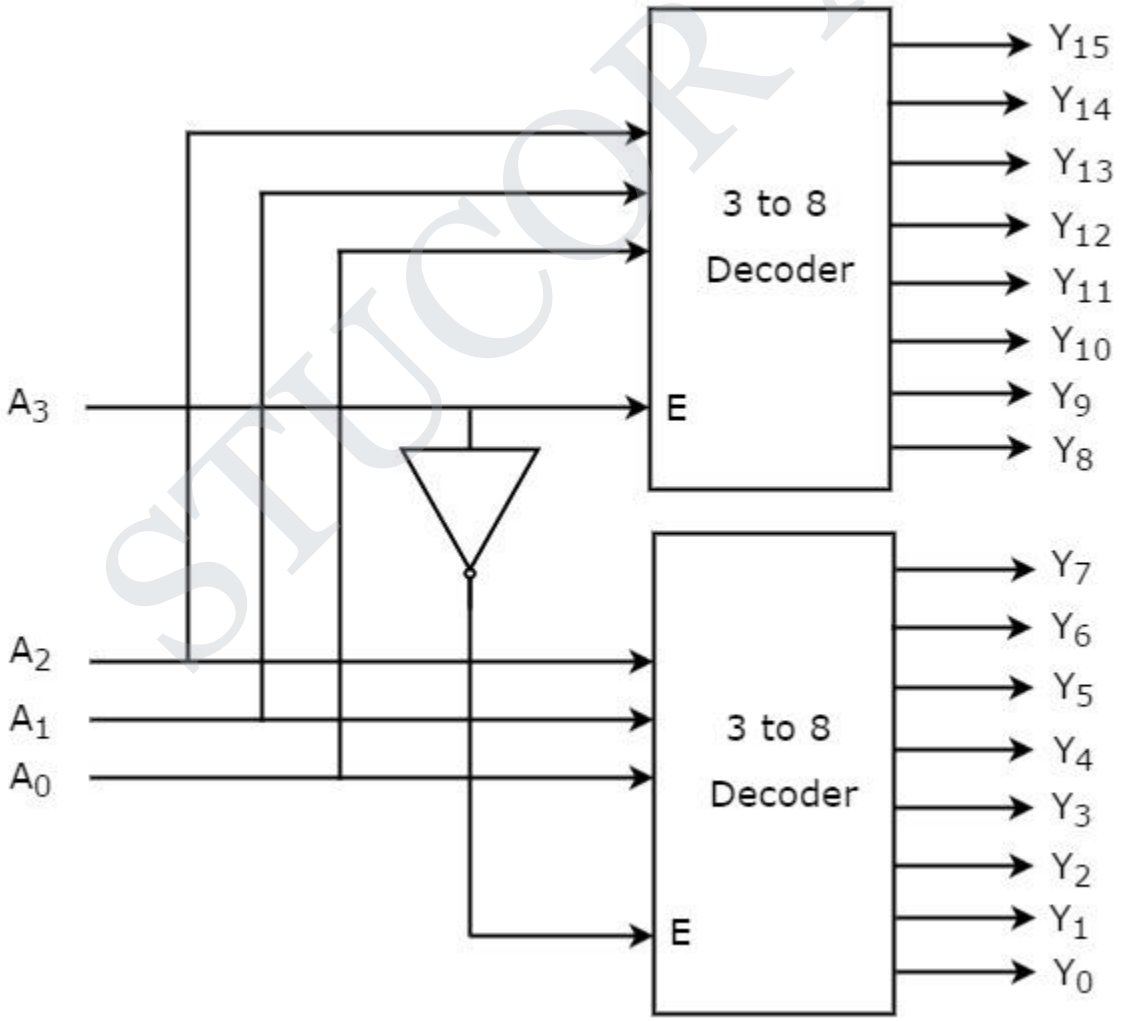
10. Implement Boolean function using mux (8)
F=sum (1,2,5,6,8,9,10,15,14)

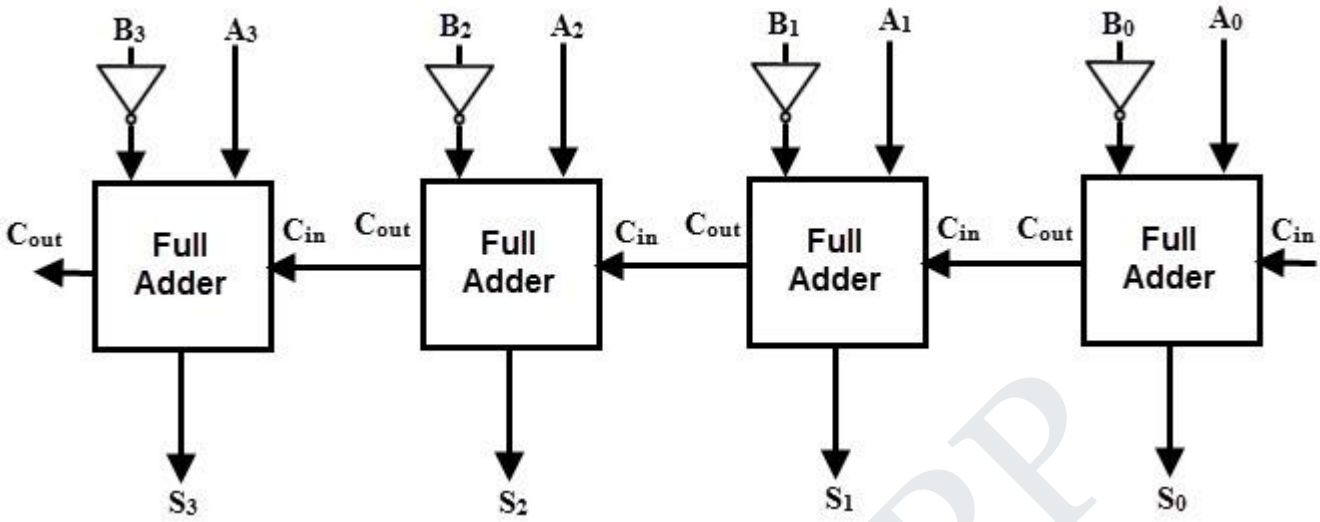
11. Construct 5 to 32 decoder using one 2 to 4decoder and four 3 to 8 decoder (8)



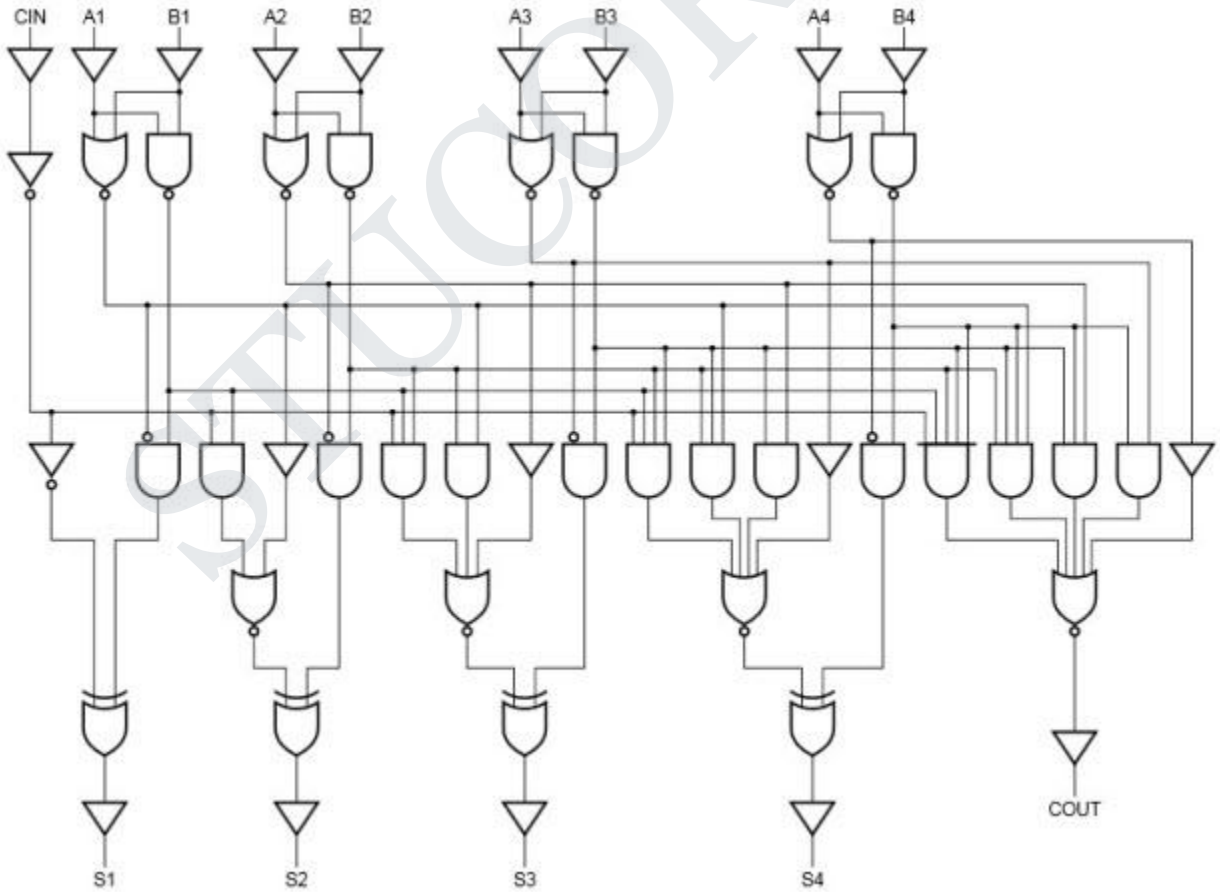
5-to-32 line decoder

12. Construct 4 x16 decoder using 2 3x8 decoders with enable input (nov/dec2013)





15.Discuss the carry look ahead adder generation (April/may 2010)



16.Design binary multiplier circuit (nov/dec2010)

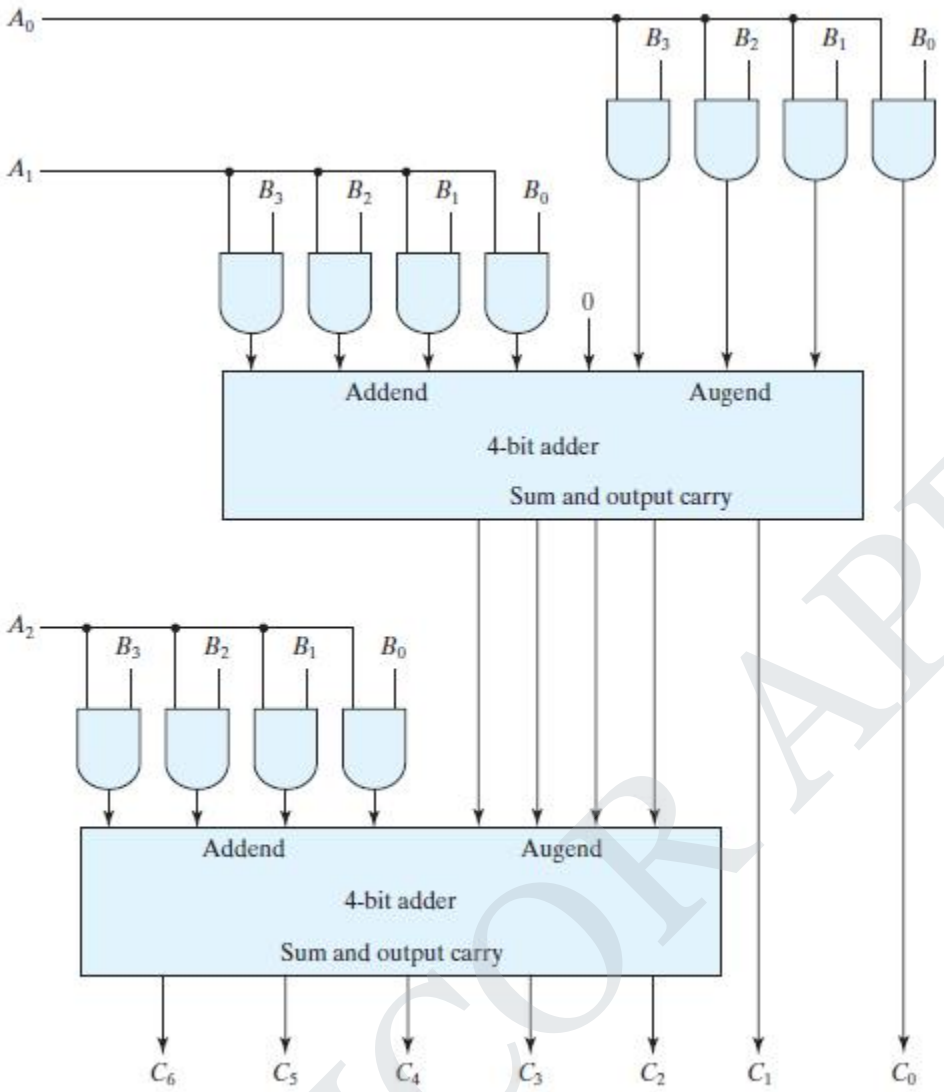


FIGURE 4.16
Four-bit by three-bit binary multiplier

SYNCHRONOUS SEQUENTIAL LOGIC

Part A – 2 Marks

1. What is sequential circuit?

Sequential circuit is a broad category of digital circuit whose logic states depend on a specified time sequence. A sequential circuit consists of a combinational circuit to which memory elements are connected to form a feedback path.

7. List the classifications of sequential circuit.

i) Synchronous sequential circuit.
ii) Asynchronous sequential circuit.

8. What is Synchronous sequential circuit?

A Synchronous sequential circuit is a system whose behavior can be defined from the knowledge of its signal at discrete instants of time.

4. What is a clocked sequential circuit?

Synchronous sequential circuit that use clock pulses in the inputs of memory elements are called clocked sequential circuit. One advantage as that they don't cause instability problems.

5. What is called latch?

Latch is a simple memory element, which consists of a pair of logic gates with their inputs and outputs inter connected in a feedback arrangement, which permits a single bit to be stored.

9. List different types of flip-flops.

x SR flip-flop
i Clocked RS flip-flop
ii D flip-flop
iii T flip-flop
iv JK flip-flop
v JK master slave flip-flop

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

The state of a flip-flop is switched by a momentary change in the input signal. This momentary change is called a trigger and the transition it causes is said to trigger the flip-flop.

8. What is an excitation table?

During the design process we usually know the transition from present state to next state and wish to find the flip-flop input conditions that will cause the required transition. A table which lists the required inputs for a given chance of state is called an excitation table.

9. Give the excitation table of JK-flip flop?

Present state	Next state	Flip-flop Inputs	
Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Present state	Next state	Flip-flop Inputs	
Q_n	Q_{n+1}	R	S
0	0	X	0
0	1	0	1
1	0	1	0
1	1	0	X

11. What is counter?

A counter is used to count pulse and give the output in binary form.

12. What is synchronous counter?

In a synchronous counter, the clock pulse is applied simultaneously to all flip-flops. The output of the flip-flops change state at the same instant. The speed of operation is high compared to an asynchronous counter

13. What is Asynchronous counter?

In an Asynchronous counter, the clock pulse is applied to the first flip-flops. The change of state in the output of this flip-flop serves as a clock pulse to the next flip-flop and so on. Here all the flip-flops do not change state at the same instant and hence speed is less.

11. What is the difference between synchronous and asynchronous counter? Synchronous counter:

13. Clock pulse is applied simultaneously Clock pulse is applied to the first flip-flop, the change of output is given as clock to next flip-flop

Asynchronous counter:

1. Speed of operation is high Speed of operation is low.

19. Name the different types of counter.

g. Synchronous counter
h. Asynchronous counter
Up counter
Down counter
Modulo – N counter
Up/Down counter

16. What is up counter?

A counter that increments the output by one binary number each time a clock pulse is applied.

17. What is down counter?

A counter that decrements the output by one binary number each time a clock pulse is applied.

18. What is up/down counter?

A counter, which is capable of operating as an up counter or down counter, depending on a control lead.

19. What is a ripple counter?

A ripple counter is nothing but an asynchronous counter, in which the output of the flip-flop changes state like a ripple in water.

- **What are the uses of a counter?**

The digital clock
Auto parking control
Parallel to serial data conversion.

- **What is Johnson counter?**

It is a ring counter in which the inverted output is fed into the input. It is also known as a twisted ring counter.

22. Define Flip flop.

The basic unit for storage is flip flop. A flip-flop maintains its output state either at1 or 0 until directed by an input signal to change its state.

23. Give the comparison between combinational circuits and sequential circuits

Combinational circuits	Sequential circuits
Memory unit is not required	Memory unit is required
Parallel adder is a combinational circuit	Serial adder is a sequenti

PART – B

1. a. Write the verilog code generate for paralled load up / down counter

Verilog Up/Down Counter (cont.)

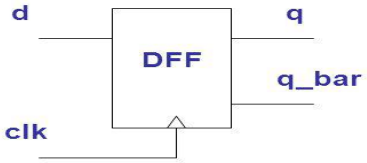
```
module up_down_counter (clk, reset, load, count_up, counter_on, Data_in, Count);
  input clk, reset, load, count_up, counter_on;
  input [2:0] Data_in;
  output [2:0] Count;
  reg [2:0] Count;

  always @ (posedge reset or posedge clk)
    if (reset == 1'b1)
      Count = 3'b0;
    else if (load == 1'b1)
      Count = Data_in;
    else if (counter_on == 1'b1) begin
      if (count_up == 1'b1)
        Count = Count + 1;
      else Count = Count - 1;
    end
endmodule
```

b. Write a verilog code for D Flip Flop

D Flip-Flop

```
// D flip-flop Code
module d_ff ( d, clk, q, q_bar);
    input d ,clk;
    output q, q_bar;
    wire d ,clk;
    reg q, q_bar;
    always @ (posedge clk)
begin
    q <= d;
    q_bar <= !d;
End
endmodule
```



3 .a. Explain S-R Flip Flop

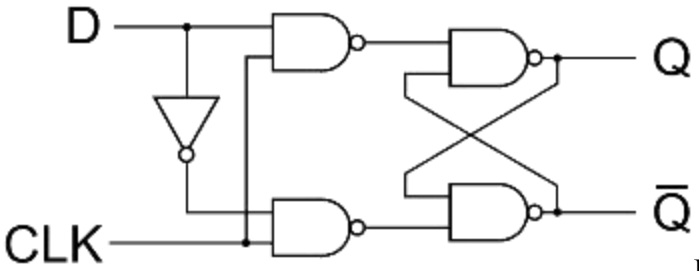
SR Flip Flop

There are majorly 4 types of flip flops, with the most common one being SR flip flop. As shown above, it is the simplest and the easiest to understand. The two outputs as shown above are the inverse of each other. The outputs of an SR flip flop are highlighted in the table below.

S	**R**	**Q**
0	0	0
0	1	0
1	0	1
1	1	∞

b. Explain D Flip Flop

D flip flop is a better alternative that is very popular with digital electronics. They are commonly used for counters and shift-registers and input synchronisation.



D Flip-Flop

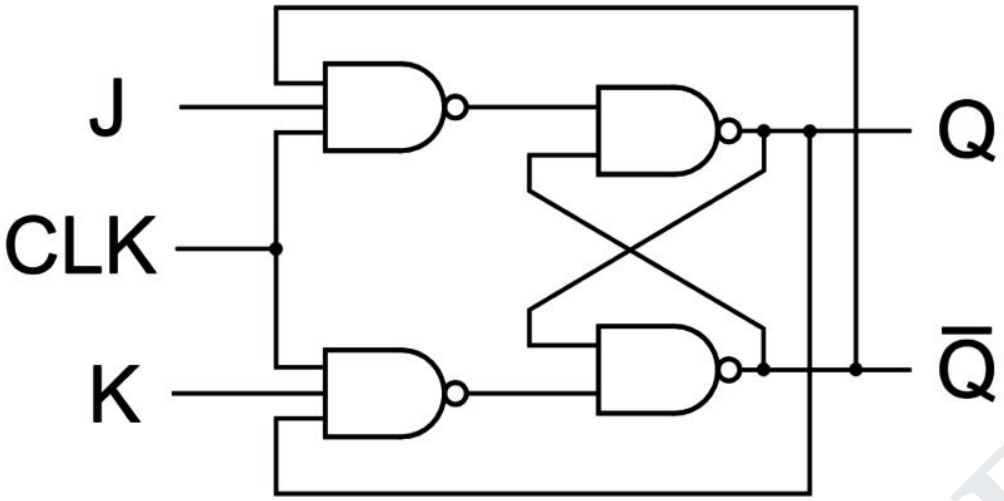
In a D flip flop, the output can be only changed at the clock edge, and if the input changes at other times, the output will be unaffected.

Clock	**D**	**Q**
↓ » 0	0	0
↑ » 1	0	0
↓ » 0	1	0
↑ » 1	1	1

The change of state of the output is dependent on the rising edge of the clock. The output (Q) is same as the input and can only change at the rising edge of the clock.

4. a. Explain JK Flip Flop

Due to the undefined state in the SR flip flop, another is required in electronics. The JK flip flop is an improvement on the SR flip flop where S=R=1 is not a problem.



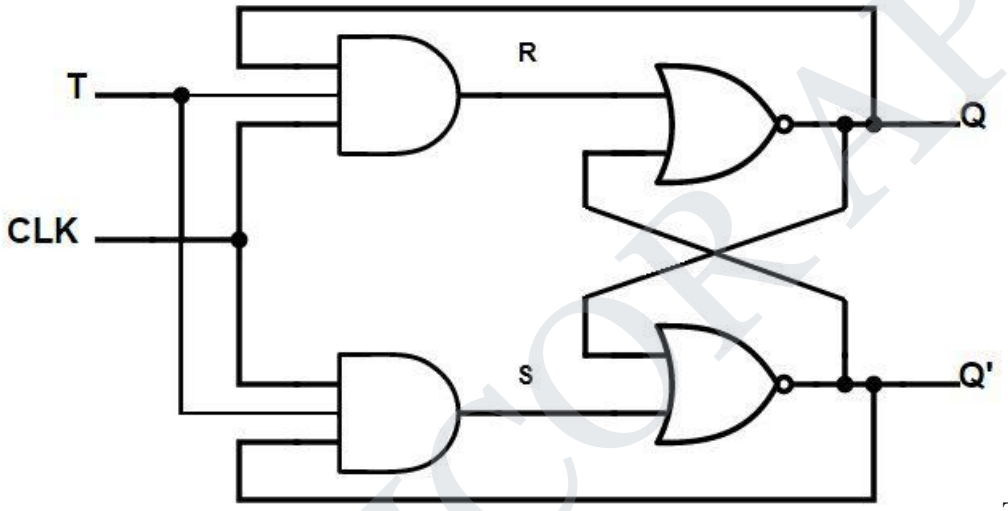
JK Flip-Flop

The input condition of J=K=1, gives an output inverting the output state. However, the outputs are the same when one tests the circuit practically.

J	K	Q
0	0	0
0	1	0
1	0	0
1	1	0
0	0	1
0	1	1
1	0	1
1	1	1

b. Explain T Flip Flop

A T flip flop is like JK flip-flop. These are basically a single input version of JK flip flop. This modified form of JK flip-flop is obtained by connecting both inputs J and K together. This flip-flop has only one input along with the clock input.



T Flip-Flop

These flip-flops are called T flip-flops because of their ability to complement its state (i.e.) Toggle, hence the name Toggle flip-flop.

T	Q	Q (
0	0	0
1	0	1
0	1	1

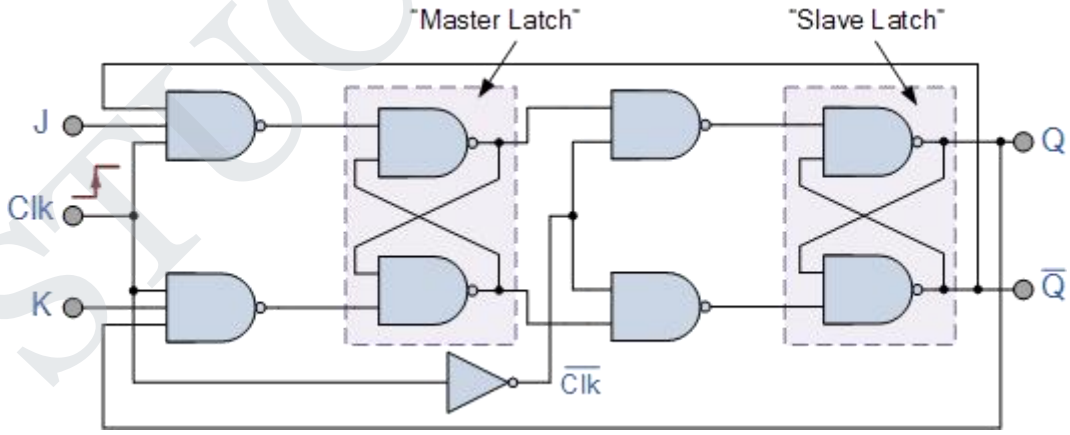
1	1	0

5. a. Explain Master Slave Flip Flop

The Master-Slave JK Flip-flop

The **Master-Slave Flip-Flop** is basically two gated SR flip-flops connected together in a series configuration with the slave having an inverted clock pulse. The outputs from Q and Q from the “Slave” flip-flop are fed back to the inputs of the “Master” with the outputs of the “Master” flip flop being connected to the two inputs of the “Slave” flip flop. This feedback configuration from the slave’s output to the master’s input gives the characteristic toggle of the JK flip flop as shown below.

The Master-Slave JK Flip Flop



The input signals J and K are connected to the gated “master” SR flip flop which “locks” the input condition while the clock (Clk) input is “HIGH” at logic level “1”. As the clock input of the “slave” flip flop is the inverse (complement) of the “master” clock input, the “slave” SR flip flop does not toggle. The outputs from the “master” flip flop are only “seen” by the gated “slave” flip flop when the clock input goes “LOW” to logic level “0”.

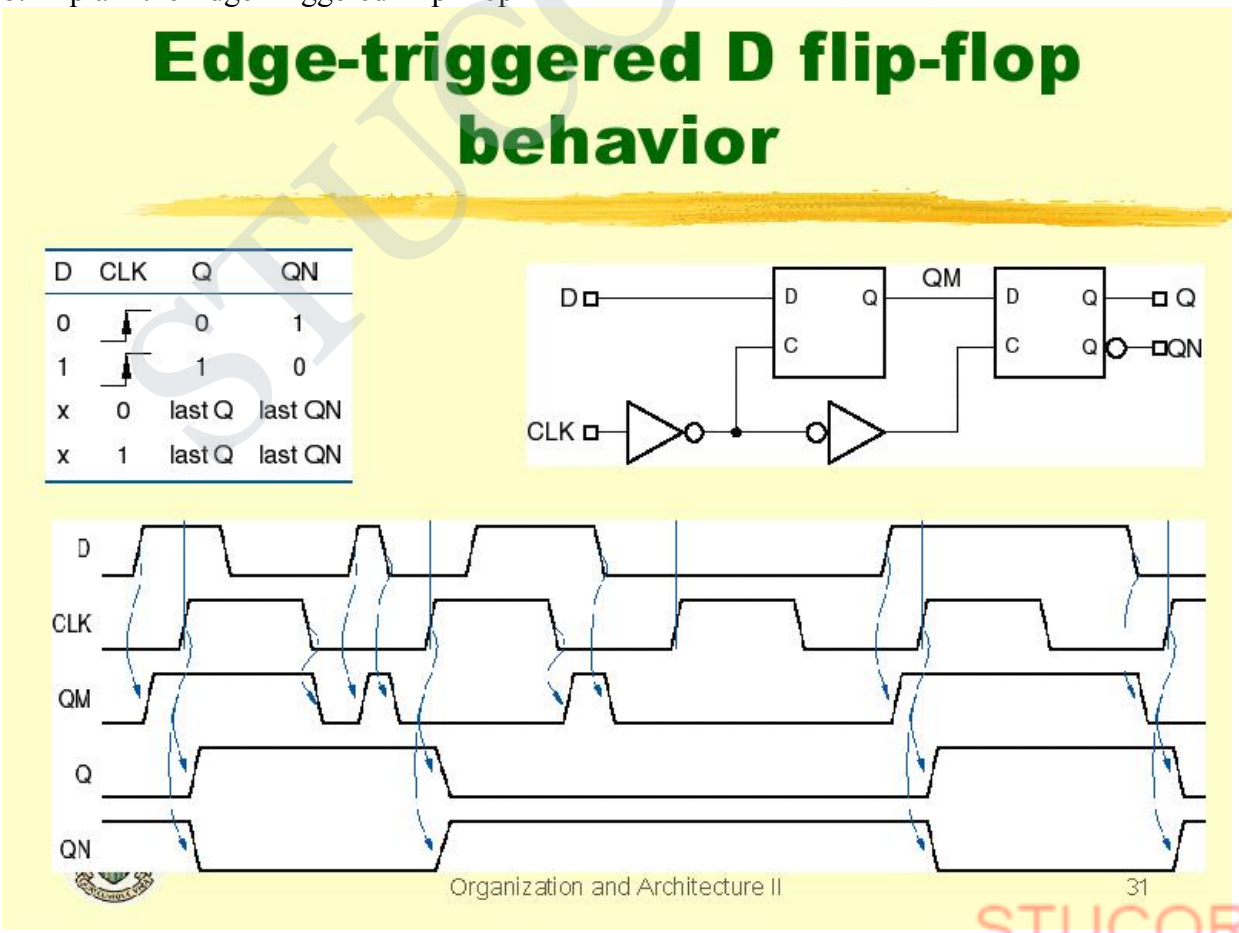
latched and any additional changes to its inputs are ignored. The gated “slave” flip flop now responds to the state of its inputs passed over by the “master” section.

Then on the “Low-to-High” transition of the clock pulse the inputs of the “master” flip flop are fed through to the gated inputs of the “slave” flip flop and on the “High-to-Low” transition the same inputs are reflected on the output of the “slave” making this type of flip flop edge or pulse-triggered.

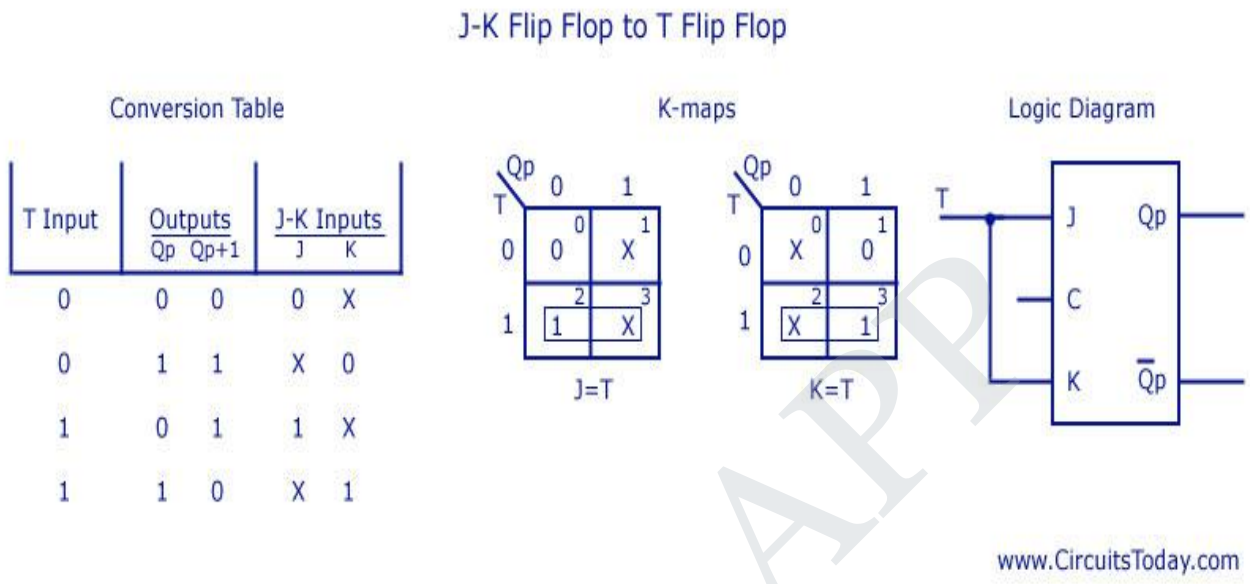
Then, the circuit accepts input data when the clock signal is “HIGH”, and passes the data to the output on the falling-edge of the clock signal. In other words, the **Master-Slave JK Flip flop** is a “Synchronous” device as it only passes data with the timing of the clock signal.

In the next tutorial about **Sequential Logic Circuits**, we will look at *Multivibrators* that are used as waveform generators to produce the clock signals to switch sequential circuits.

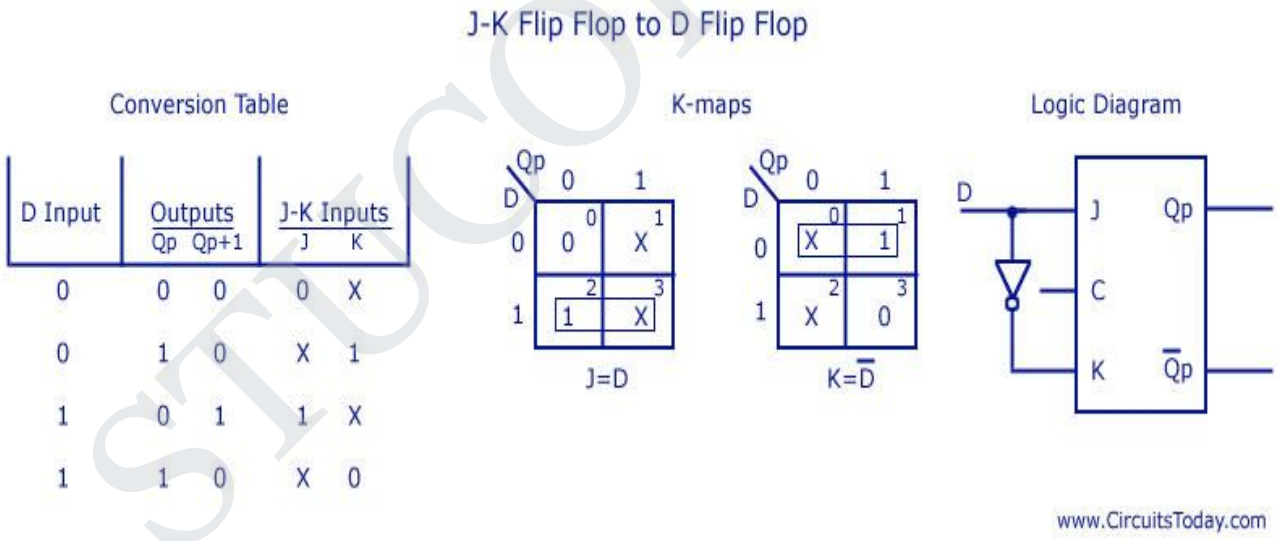
b. Explain the Edge Triggered Flip Flop



6. a. Convert it JK Flip Flop in to T Flip Flop



b. Convert it JK Flip Flop in to D Flip Flop



8. a. Explain Serial in Serial out Shift Register

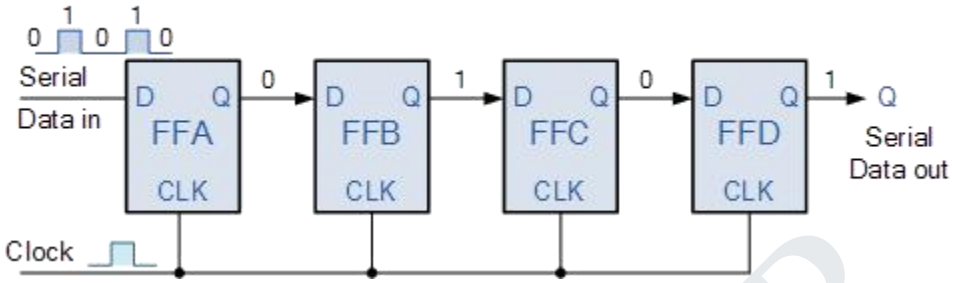
Serial-in to Serial-out (SISO) Shift Register

This **shift register** is very similar to the SIPO above, except were before the data was read directly in a parallel form from the outputs Q_A to Q_D, this time the data is allowed to flow straight through the register and out of the other end. Since there is only one output, the DATA leaves the shift register one bit at a time in a serial pattern, hence the name **Serial-in to Serial-Out Shift Register** or **SISO**.

serial input (SI) which determines what enters the left hand flip-flop, the serial output (SO) which is taken

from the output of the right hand flip-flop and the sequencing clock signal (Clk). The logic circuit diagram below shows a generalized serial-in serial-out shift register.

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

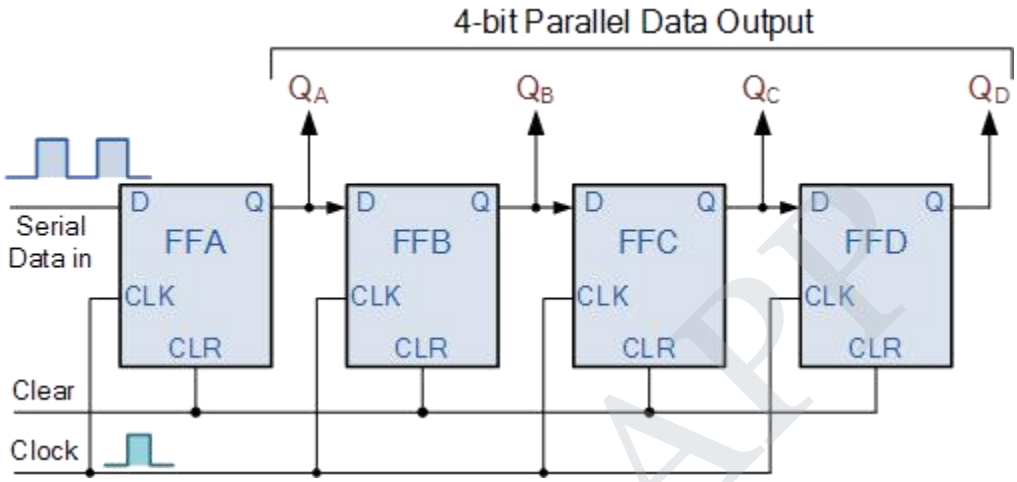


You may think what's the point of a SISO shift register if the output data is exactly the same as the input data. Well this type of **Shift Register** also acts as a temporary storage device or it can act as a time delay device for the data, with the amount of time delay being controlled by the number of stages in the register, 4, 8, 16 etc or by varying the application of the clock pulses. Commonly available IC's include the 74HC595 8-bit Serial-in to Serial-out Shift Register all with 3-state outputs.

b. Explain Serial in parallel out Shift Register

Serial-in to Parallel-out (SIPO) Shift Register

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



The operation is as follows. Lets assume that all the flip-flops (FFA to FFD) have just been RESET (CLEAR input) and that all the outputs Q_A to Q_D are at logic level "0" ie, no parallel data output.

If a logic "1" is connected to the DATA input pin of FFA then on the first clock pulse the output of FFA and therefore the resulting Q_A will be set HIGH to logic "1" with all the other outputs still remaining LOW at logic "0". Assume now that the DATA input pin of FFA has returned LOW again to logic "0" giving us one data pulse or 0-1-0.

The second clock pulse will change the output of FFA to logic "0" and the output of FFBand Q_B HIGH to logic "1" as its input D has the logic "1" level on it from Q_A. The logic "1" has now moved or been "shifted" one place along the register to the right as it is now at Q_A.

When the third clock pulse arrives this logic "1" value moves to the output of FFC (Q_C) and so on until the arrival of the fifth clock pulse which sets all the outputs Q_A to Q_D back again to logic level "0" because the input to FFA has remained constant at logic level "0".

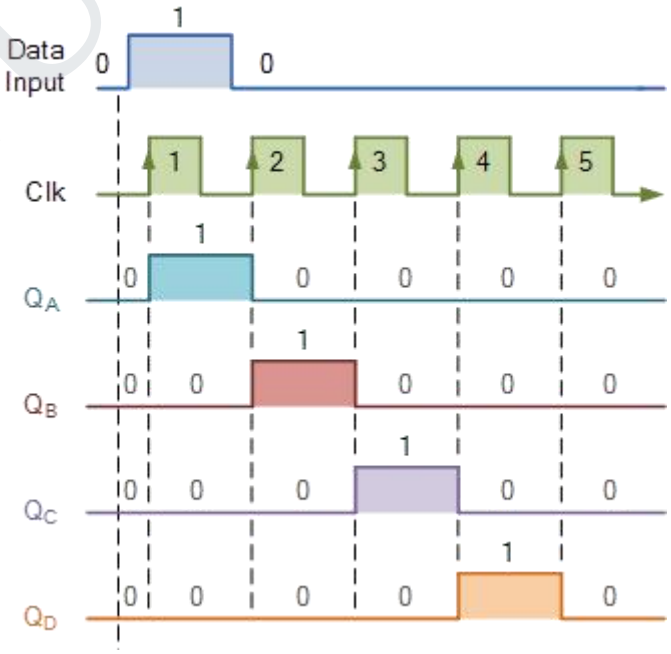
The effect of each clock pulse is to shift the data contents of each stage one place to the right, and this is shown in the following table until the complete data value of 0-0-0-1 is stored in the register. This data value can now be read directly from the outputs of Q_A to Q_D.

Then the data has been converted from a serial data input signal to a parallel data output. The truth table and following waveforms show the propagation of the logic "1" through the register from left to right as follows.

Basic Data Movement Through A Shift Register

Clock Pulse No	QA	QB	QC	QD

0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	0	0	0	0



Note that after the fourth clock pulse has ended the 4-bits of data (0-0-0-1) are stored in the register and consist of various combinations of logic "1" and "0". Commonly available SIPO IC's include the standard 8-

bit 74LS164 or the 74LS594.

Serial-in to Serial-out (SISO) Shift Register

This **shift register** is very similar to the SIPO above, except were before the data was read directly in a parallel form from the outputs Q_A to Q_D, this time the data is allowed to flow straight through the register and out of the other end. Since there is only one output, the DATA leaves the shift register one bit at a time in a serial pattern, hence the name **Serial-in to Serial-Out Shift Register** or **SISO**.

The SISO shift register is one of the simplest of the four configurations as it has only three connections, the serial input (SI) which determines what enters the left hand flip-flop, the serial output (SO) which is taken from the output of the right hand flip-flop and the sequencing clock signal (Clk). The logic circuit diagram below shows a generalized serial-in serial-out shift register.

9. a. Explain parallel in parallel out Shift Register

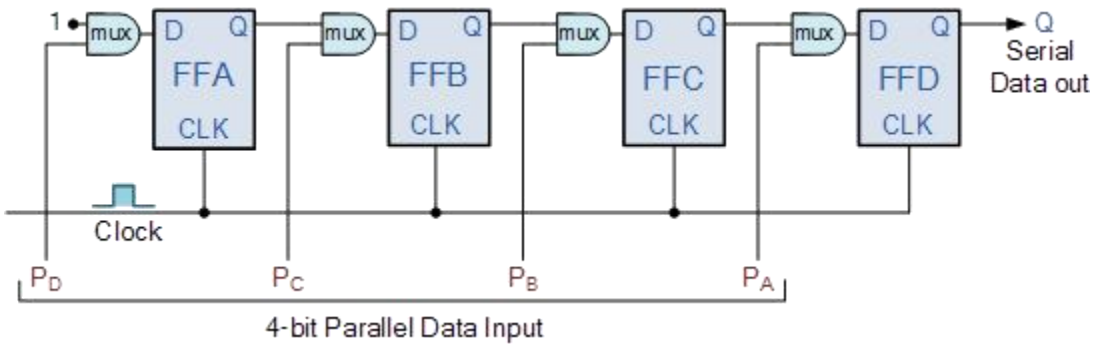
Parallel-in to Serial-out (PISO) Shift Register

The Parallel-in to Serial-out shift register acts in the opposite way to the serial-in to parallel-out one above. The data is loaded into the register in a parallel format in which all the data bits enter their inputs simultaneously, to the parallel input pins P_A to P_D of the register. The data is then read out sequentially in the normal shift-right mode from the register at Q representing the data present at P_A to P_D.

This data is outputted one bit at a time on each clock cycle in a serial format. It is important to note that with this type of data register a clock pulse is not required to parallel load the register as it is already present, but four clock pulses are required to unload the data.

b. Explain parallel in Serial out Shift Register

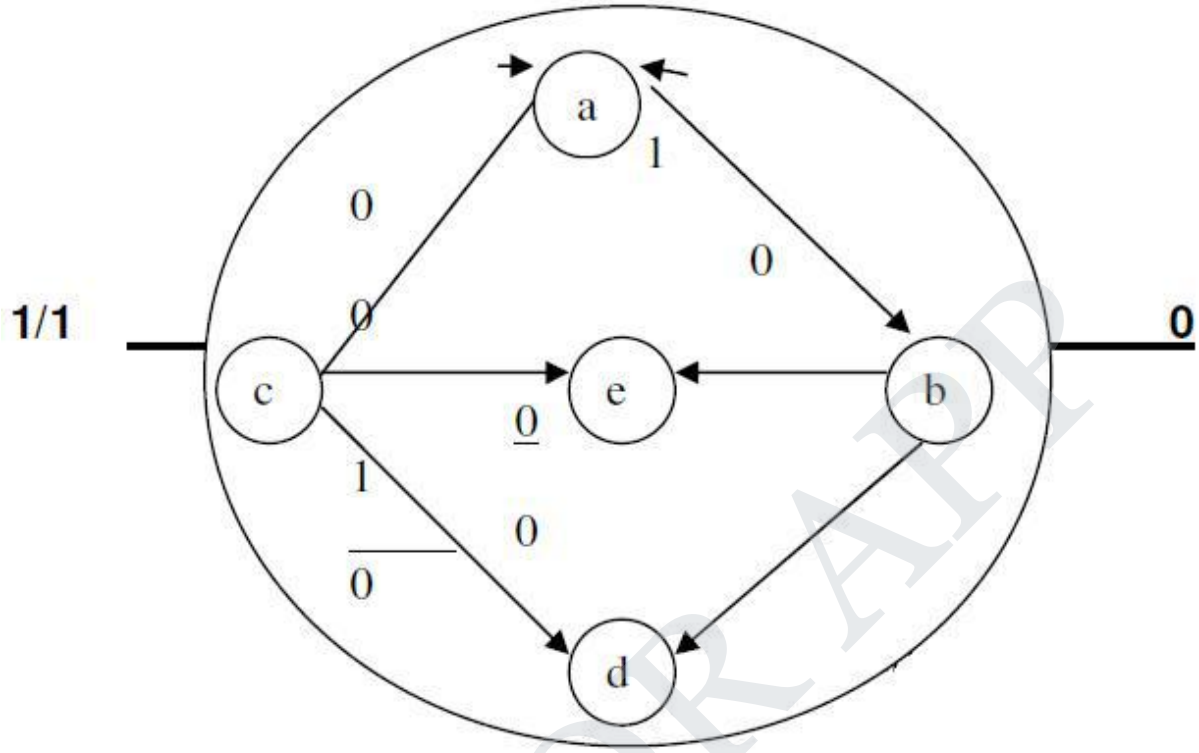
4-bit Parallel-in to Serial-out Shift Register



As this type of shift register converts parallel data, such as an 8-bit data word into serial format, it can be

computer or transmitted over a communications line. Commonly available IC's include the 74HC166 8-bit Parallel-in/Serial-out Shift Registers.

10. Design sequential circuit for a state diagram?



UNIT IV

ASYNCHRONOUS SEQUENTIAL LOGIC

Part A – 2 Marks

1. Define Asynchronous sequential circuit?

In asynchronous sequential circuits change in input signals can affect memory element at any instant of time.

2. Comparison between synchronous & Asynchronous sequential circuits?

Synchronous sequential circuits	Asynchronous sequential circuits
Memory elements are clocked flip-flops	Memory elements are either unlocked flip - flops or time delay elements.
Easier to design.	More difficult to design.

9. What is fundamental mode sequential circuit?

-input variables changes if the circuit is
stable -inputs are levels, not pulses
-only one input can change at a given time.

10. What is the significance of state assignment?

In synchronous circuits-state assignments are made with the objective of circuit reduction. Asynchronous circuits-its objective is to avoid critical races.

5. When do race conditions occur?

Two or more binary state variables change their value in response to the change in input variable.

6. Write short note on shared row state assignment.

Races can be avoided by making a proper binary assignment to the state variables. Here, the state variables are assigned with binary numbers in such a way that only one state variable can change at any one state variable can change at any one time when a state transition occurs. To accomplish this, it is necessary that states between which transitions occur be given adjacent assignments. Two binary are said to be adjacent if they differ in only one variable.

7. Write short note on one hot state assignment.

The one hot state assignment is another method for finding a race free state assignment. In this method, only one variable is active or hot for each row in the original flow table, i.e., it requires one state variable for each row of the flow table. Additional row are introduced to provide single variable changes between internal state transitions.

11. What are the different techniques used in state assignment?

- shared row state assignment
- one hot state assignment

12. What are the steps for the design of asynchronous sequential circuit?

-construction of primitive flow table

on of flow table

-realization of primitive flow table

13. What is hazard?

Hazard is an unwanted switching transient.

14. What are the steps for the design of asynchronous sequential circuit?

Construction of a primitive flow table from the problem statement.

Primitive flow table is reduced by eliminating redundant states using the state reduction

State assignment is made

x The primitive flow table is realized using appropriate logic elements.

15. Give the comparison between state Assignment Synchronous circuit and state assignment asynchronous circuit.

In synchronous circuit, the state assignments are made with the objective of circuit reduction. In asynchronous circuits, the objective of state assignment is to avoid critical races.

16. What are races?

When 2 or more binary state variables change their value in response to a change in an input variable, race condition occurs in an asynchronous sequential circuit. In case of unequal delays, a race condition may cause the state variables to change in an unpredictable manner.

14. Define non critical race.

If the final stable state that the circuit reaches does not depend on the order in which the state variable changes, the race condition is not harmful and it is called a non critical race.

15. Define critical race?

If the final stable state depends on the order in which the state variable changes, the race condition is harmful and it is called a critical race.

16. Define flow table in asynchronous sequential circuit.

In asynchronous sequential circuit state table is known as flow table because of the behavior of the asynchronous sequential circuit. The stage changes occur in independent of a clock, based on the logic propagation delay, and cause the states to flow from one to another.

17. Define merger graph.

The merger graph is defined as follows. It contains the same number of vertices as the state table contains states. A line drawn between the two state vertices indicates each compatible state pair. It two states are incompatible no connecting line is drawn.

18. What is fundamental mode?

A transition from one stable state to another occurs only in response to a change in the input state. After a change in one input has occurred, no other change in any input occurs until the circuit enters a stable state. Such a mode of operation is referred to as a fundamental mode.

PART – B

1. Explain the classification of Race- Free State Algorithm? (16)

RACE -FREE STATE ASSIGNMENT

Once a reduced flow table has been derived for an asynchronous sequential circuit, the next step in the design is to assign binary variables to each stable state. This assignment results in the transformation of the flow table into its equivalent transition table. The primary objective in choosing a proper binary state assignment is the prevention of critical races. Critical races can be avoided by making a binary state assignment in such a way that only one variable changes at any given time when a state transition occurs in the flow table.

✓·**Three-Row Flow-Table Example**

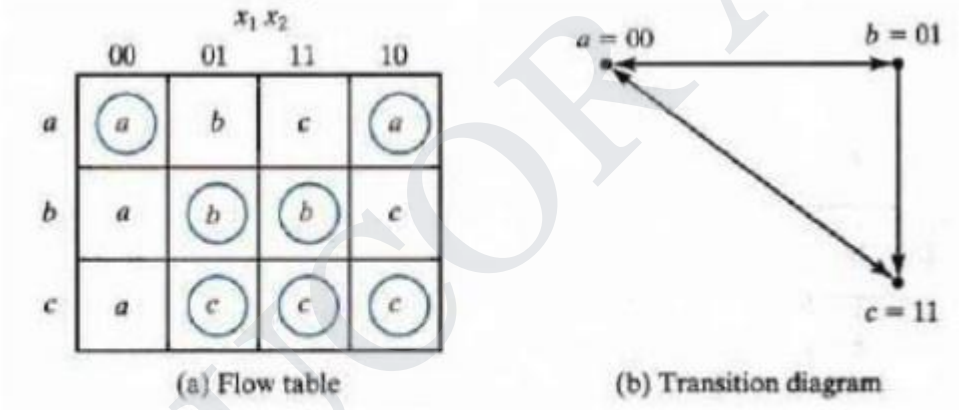
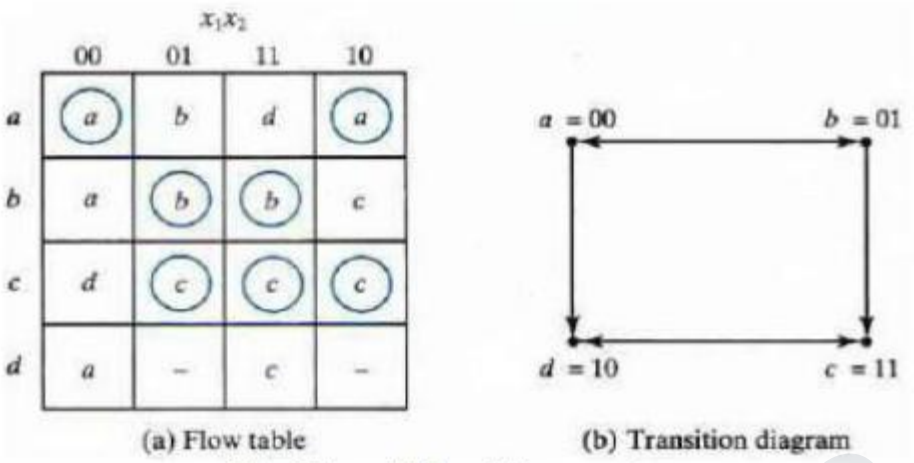


Fig: Three row flow table example

Fig: Three row flow table example

To avoid critical races, we must find a binary state assignment such that only one binary variable changes during each state transition. An attempt to find such an assignment is shown in the transition diagram. State a is assigned binary 00, and state c is assigned binary 11. This assignment will ca use a critical race during the transition from a to c because there are two changes in the binary state variables and the transition from a to c may occur directly or pass through b. Note that the transition from c to a also ca uses a race condition, but it is noncritical because the transition does not pass through other states.



(a) Flow table (b) Transition diagram

Fig: Flow table with an extra row

A race-free assignment can be obtained if we add an extra row to the flow table. The use of a fourth row does not increase the number of binary state variables, but it allows the formation of cycles between two stable states.

The transition table corresponding to the flow table with the indicated binary state assignment is shown in Fig. The two dashes in row d represent unspecified states that can be considered don't-care conditions. However, care must be taken not to assign 10 to these squares, in order to avoid the possibility of an unwanted stable state being established in the fourth row.

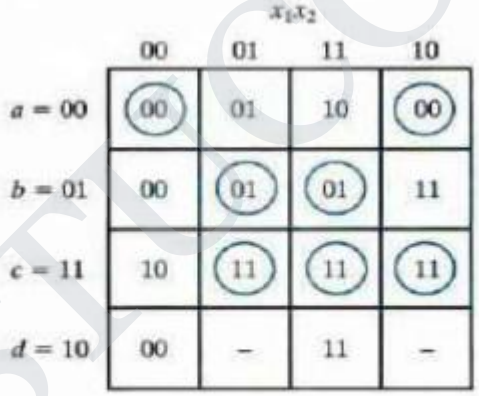


Fig: Transition table

✓ **Four-Row Flow-Table Example**

A flow table with four rows requires a minimum of two state variables. Although a race-free assignment is sometimes possible with only two binary state variables, in many cases the requirement of extra rows to avoid critical races will dictate the use of three binary state variables

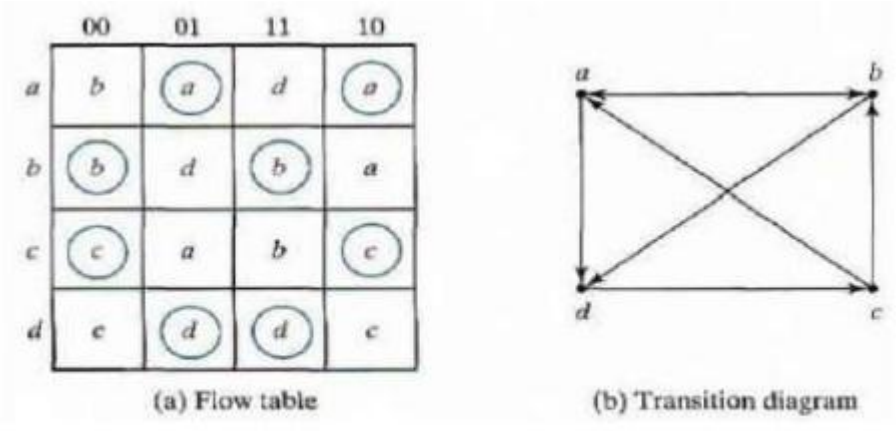


Fig: Four-row flow-table example

Fig: Four-row flow-table example

The following figure shows a state assignment map that is suitable for any four-row flow table. States a, b, c and d are the original states and e, f and g are extra states. The transition from a to d must be directed through the extra state e to produce a cycle so that only one binary variable changes at a time. Similarly, the transition from c to a is directed through g and the transition from d to c goes through f. By using the assignment given by the map, the four-row table can be expanded to a seven-row table that is free of critical races.

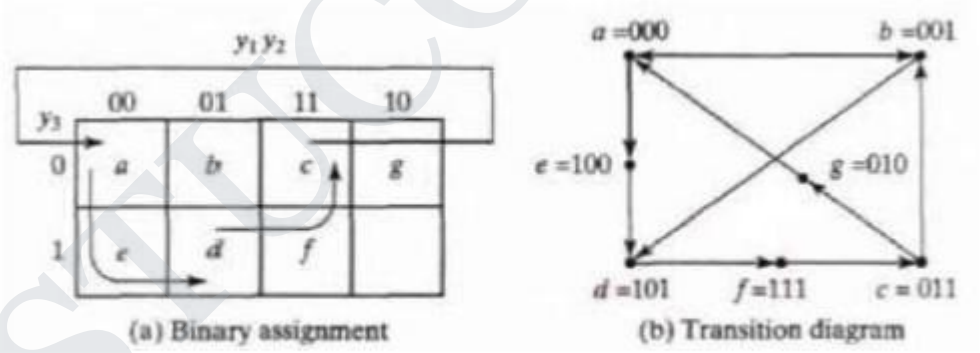


Fig: Choosing extra rows for the flow table

Fig: Choosing extra rows for the flow table

Note that although the flow table has seven rows there are only four stable states. The uncircled states in the three extra rows are there merely to provide a race-free transition between the stable states.

	00	01	11	10
000 = a	b	(a)	c	(a)
001 = b	(b)	d	(b)	a
011 = c	(c)	g	b	(c)
010 = g	–	a	–	–
110 –	–	–	–	–
111 = f	c	–	–	c
101 = d	f	(d)	(d)	f
100 = e	–	–	d	–

Fig: State assignment to modified flow table

Fig: State assignment to modified flow table

✓ **Multiple-Row Method**

The method for making race-free stale assignments by adding extra rows in the flow table is referred to as the *shared-row* method. A second method called the *multiple-row* method is not as efficient, but is easier to apply. In multiple-row assignment each state in the original row table is replaced by two or more combinations of state variables.

	y_2y_3 00	01	11	10
y_1 0	a_1	b_1	c_1	d_1
1	c_2	d_2	a_2	b_2

(a) Binary assignment

	00	01	11	10
000 = a_1	b_1	(a_1)	d_1	(a_1)
111 = a_2	b_2	(a_2)	d_2	(a_2)
001 = b_1	(b_1)	d_2	(b_1)	a_1
110 = b_2	(b_2)	d_1	(b_2)	a_2
011 = c_1	(c_1)	a_2	b_1	(c_1)
100 = c_2	(c_2)	a_1	b_2	(c_2)
010 = d_1	c_1	(d_1)	(d_1)	c_1
101 = d_2	c_2	(d_2)	(d_2)	c_2

(b) Flow table

Fig: Multiple row assignment

the logical complement of the other. For example, the original state a is replaced with two equivalent states a_1 =000 and a_2 = 111. The output values, not shown here must be the same in a_1 and a_2. Note that a_1 is adjacent to b_1, c_2 and d_1, and $a_2$is adjacent to c_1, b_2 and d_2, and similarly each state is adjacent to three states with different letter designations.

The expanded table is formed by replacing each row of the original table with two rows. In the multiple-row assignment, the change from one stable state 10 another will always cause a change of only one binary state variable. Each stable stale has two binary assignments with exactly the same output.

2. Explain the Hazards in combinational circuits? (6)

Hazards In Combinational Circuits

A hazard is a condition in which a change in a single variable produces a momentary change in output when no change in output should occur.

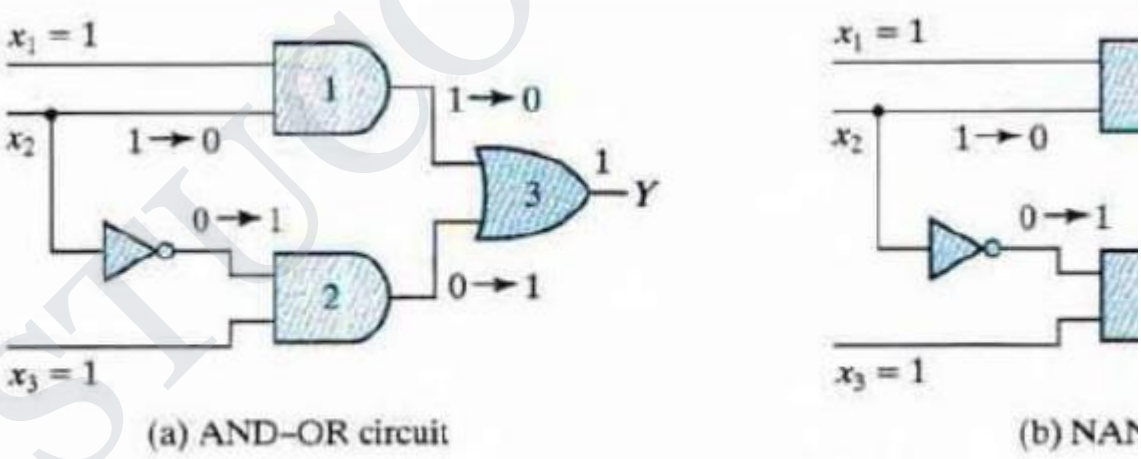


Fig: Circuits with Hazards

Fig: Circuits with Hazards

Assume that all three inputs are initially equal to 1. This causes the output of gate 1 10 be 1, that of gate 2 to be 0 and that of the circuit to be 1. Now consider a change in x 2 from 1 to 0. Then the output of gate 1 changes to 0 and that of gate 2 changes to 1, leaving the output at 1. However, the output may momentarily go to 0 if the propagation delay through the inverter is taken into

e delay in the inverter may cause the output of gate 1 to

The two circuits shown in Fig implement the Boolean function in sum-of-products form:

$$Y = x_1x_2 + \overline{x_2}x_3$$

This type of implementation may cause the output to go to 0 when it should remain a 1. If however, the circuit is implemented instead in product-of-sums form namely,

$$Y = (x_1 + \overline{x_2})(x_2 + x_3)$$

then the output may momentarily go to 1 when it should remain 0. The first case is referred to as **static 1-hazard** and the second case as **static 0-hazard**.

A third type of hazard, known as **dynamic hazard**, causes the output to change three or more times when it should change from 1 to 0 or from 0 to 1.

Fig: Types of hazards

Fig: Types of hazards

The change in x2 from 1 to 0 moves the circuit from minterm 111 to minterm 101. The hazard exists because the change in input results in a different product term covering the two minterm.

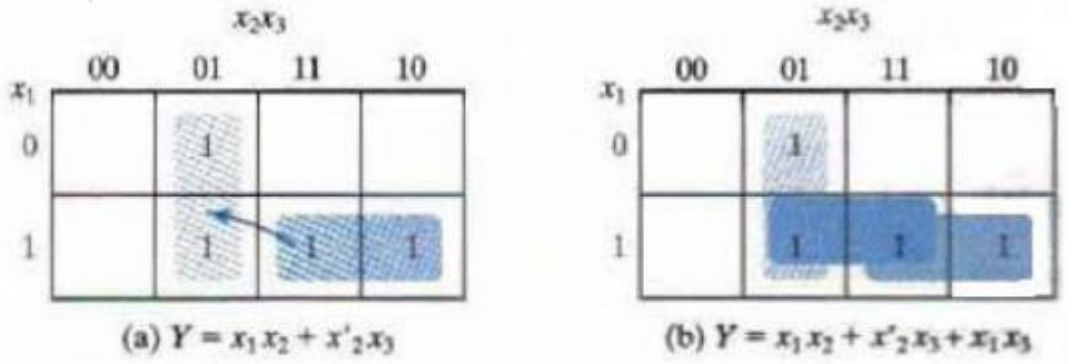


Fig: Illustrates hazard and its removal

Minterm 111 is covered by the product term implemented in gate 1 and minterm 101 is covered by the product term implemented in gate 2. The remedy for eliminating a hazard is to enclose the two minterms with another product term that overlaps both groupings. The hazard-free circuit obtained by such a configuration is shown in figure below. The extra gate in the circuit generates the product term x_1x_3. In general, hazard s in combinational circuits can be removed by cove ring any two minterms that may produce a hazard with a product term common to both. The removal of hazards requires the addition of redundant gates to the circuit.

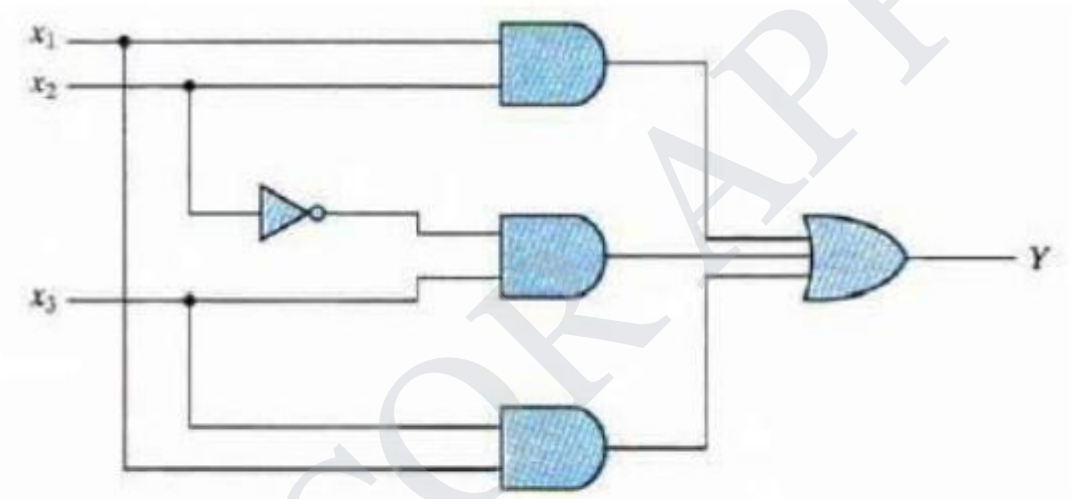


Fig: Hazard free circuit

Fig: Hazard free circuit

✓ **Hazards in Sequential Circuits**

In normal combinational-circuit design associated with synchronous sequential circuits, hazards are of no concern, since momentary erroneous signals are not generally troublesome. However, if a momentary incorrect signal is fed back in an asynchronous sequential circuit, it may cause the circuit to go to the wrong stable state.

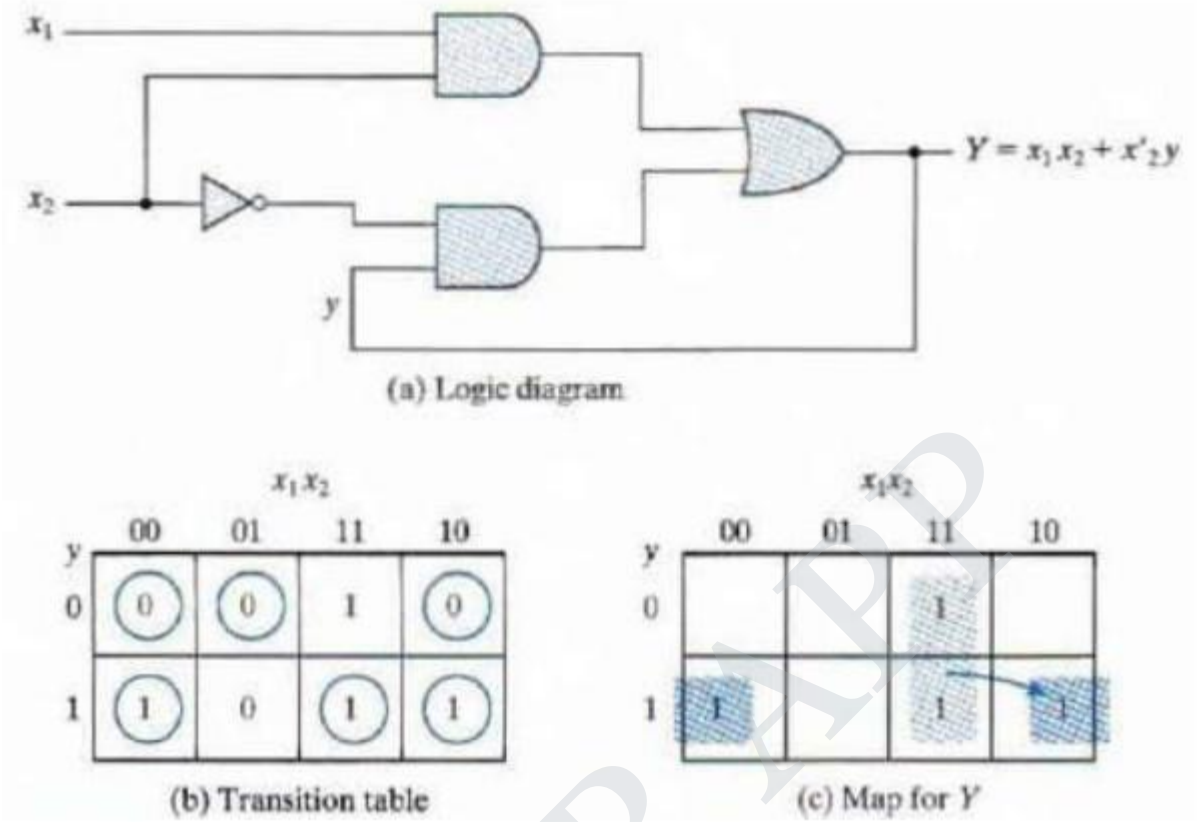


Fig: Hazard in an Asynchronous sequential circuit

Fig: Hazard in an Asynchronous sequential circuit

If the circuit is in total stable state yx_1x_2 =111 and input x2 changes from I to 0, the next total stable state should be 110. However, because of the hazard, output Y may go to 0 momentarily. If this false signal feeds back into gate 2 before the output of the inverter goes to 1, the output of gate 2 will remain at 0 and the circuit will switch to the incorrect total stable state 010. This malfunction can be eliminated by adding an extra gate.

✓ **Essential Hazards**

Another type of hazard that may occur in asynchronous sequential circuits is called an **essential hazard**.This type of hazard is caused by unequal delays along two or more paths that originate from the same input. An excessive delay through an inverter circuit in comparison to the delay associated with the feedback path may cause such a hazard.

cannot be corrected by adding redundant gates as in static hazards. The problem that they impose can be corrected by adjusting the

amount of delay in the affected path. To avoid essential hazards, each feedback loop must be handled with individual care to ensure that the delay in the feedback path is long enough compare d with delays of other signals that originate from the input terminals.

3.Explain the Hazards in sequential circuits? (10)

Hazards in Sequential Circuits

In normal combinational-circuit design associated with synchronous sequential circuits, hazards are of no concern, since momentary erroneous signals are not generally troublesome. However, if a momentary incorrect signal is fed back in an asynchronous sequential circuit, it may cause the circuit to go to the wrong stable state.

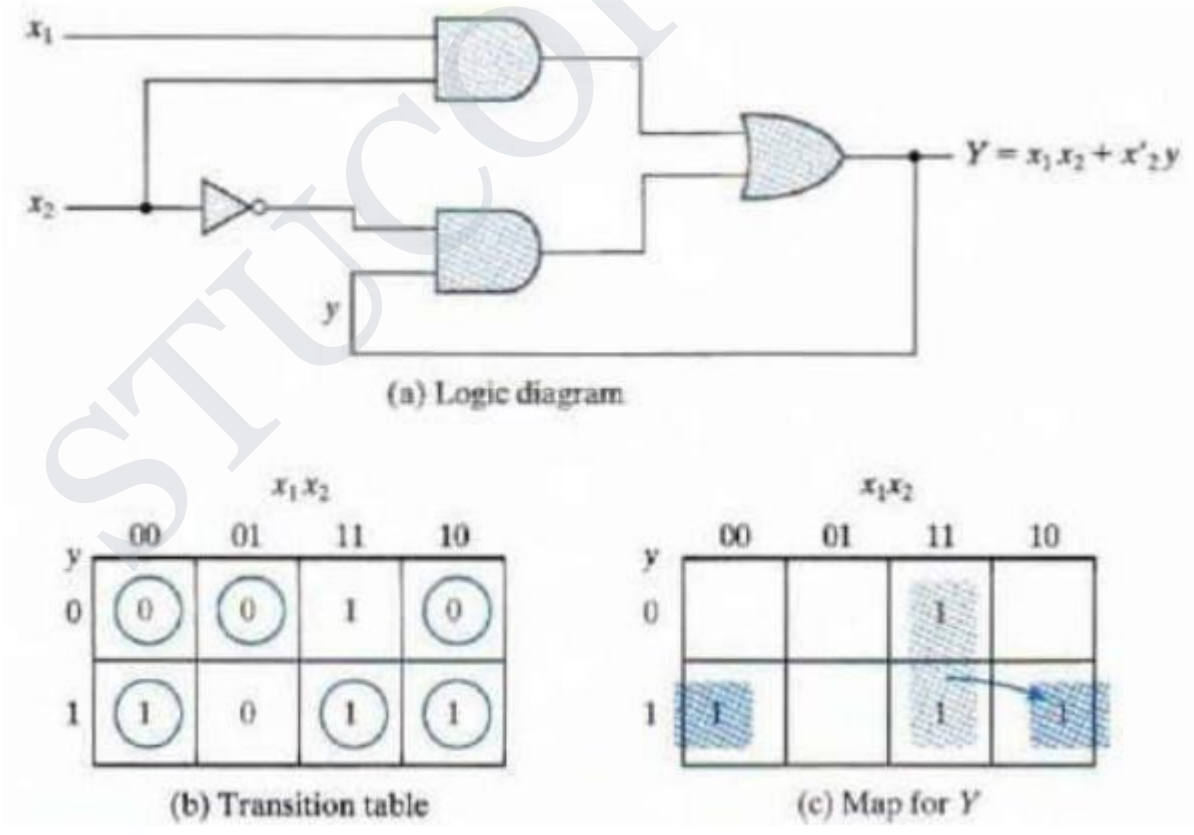


Fig: Hazard in an Asynchronous sequential circuit

Fig: Hazard in an Asynchronous sequential circuit

total stable state yx x =111 and input x2 changes from I to 0,

the next total stable state should be 110. However, because of the hazard, output Y may go to 0 momentarily. If this false signal feeds back into gate 2 before the output of the inverter goes to 1, the output of gate 2 will remain at 0 and the circuit will switch to the incorrect total stable state 010. This malfunction can be eliminated by adding an extra gate.

4. Explain the analysis and design procedures of synchronous sequential circuits. (8)

Analysis of Sequential Circuits

- Analysis is describing what a given circuit will do
- The behavior of a clocked (synchronous) sequential circuit is determined from the inputs, the output, and the states of FF

Steps:

- Obtain state equations
  - FF input equations
  - Output equations
- Fill the state table
  - Put all combinations of inputs and current states
  - Fill the next state and output
- Draw the state diagram

KFUPM

5.

5-7 Design Procedure

The procedure for designing synchronous sequential circuits can be summarized by a list of recommended steps.

1. From the word description and specifications of the desired operation, derive a state diagram for the circuit.
2. Reduce the number of states if necessary.
3. Assign binary values to the states.
4. Obtain the binary-coded state table.
5. Choose the type of flip-flops to be used.
6. Derive the simplified flip-flop input equations and output equations.
7. Draw the logic diagram.

62

6. With necessary example and diagram explain the concept of reduction of state and flow tables.

REDUCTION OF STATE AND FLOW TABLES

The procedure for reducing the number of internal states in an asynchronous sequential circuit resembles the procedure that is used for synchronous circuits.

✓·**Implication Table and Implied State**

The state-reduction procedure for completely specified state tables is based on an algorithm that combines two slates in a slate table into one as long as they can be shown to be equivalent. Two states are equivalent if, for each possible input, they give exactly the same output and go to the same next states or to

tes.

State Table to Demonstrate Equivalent States

Present State	Next State		Output	
	x = 0	x = 1	x = 0	x = 1
a	c	b	0	1
b	d	a	0	1
c	a	d	1	0
d	b	d	1	0

Consider for example the state table shown in above table. The present states a and b have the same output for the same input. Their next states are c and d for x = 0 and b and a for x = 1. If we can show that the pair of states (c, d) are equivalent, then the pair of states (a , b) will also be equivalent, because they will have the same or equivalent next states. When this relationship exists, we say that (a. b) imply (c, d) in the sense that if a and b are equivalent then r and d have to be equivalent. Similarly, from the last two rows of above table, we find that the pair of stales (c, d) implies the pair of states (a, b). The characteristic of equivalent states is that if (a, b) imply (c, d) and (c, d) imply (a, b), then both pairs of states are equivalent that is, a and b are equivalent, and so are c and d. As a consequence, the four rows of table can be reduced to two rows by combining a and b into one state and c and d into a second state.

State Table to Be Reduced

Present State	Next State		Output	
	x = 0	x = 1	x = 0	x = 1
a	d	b	0	0
b	e	a	0	0
c	g	f	0	1
d	a	d	1	0
e	a	d	1	0
f	c	b	0	0
g	a	e	1	0

The implication table is shown in Fig. On the left side along the vertical are listed all the states defined in the state table except the first and across the bottom horizontally are listed all the states except the last. The result is a display of all possible combinations of two stares with a square placed in the

equivalence. Two states having different outputs for the same input are not equivalent.

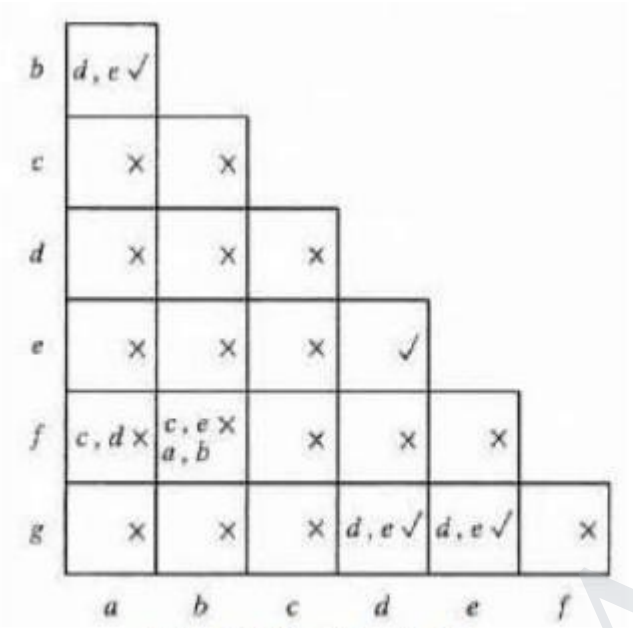


Fig: Implication table

Two states that are not equivalent are marked with a cross [X] in the corresponding square whereas their equivalence is recorded with a check mark('Dik'). Some of the squares have entries of implied states that must be investigated further to determine whether they are equivalent. Thus table can be reduced from seven states to four one for each member of the preceding partition. The reduced state table is obtained by replacing state b by a and states e and g by d and it is shown below,

Reduced State Table

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	d	a	0	0
c	d	f	0	1
d	a	d	1	0
f	c	a	0	0

✓ Merging of the Flow Table

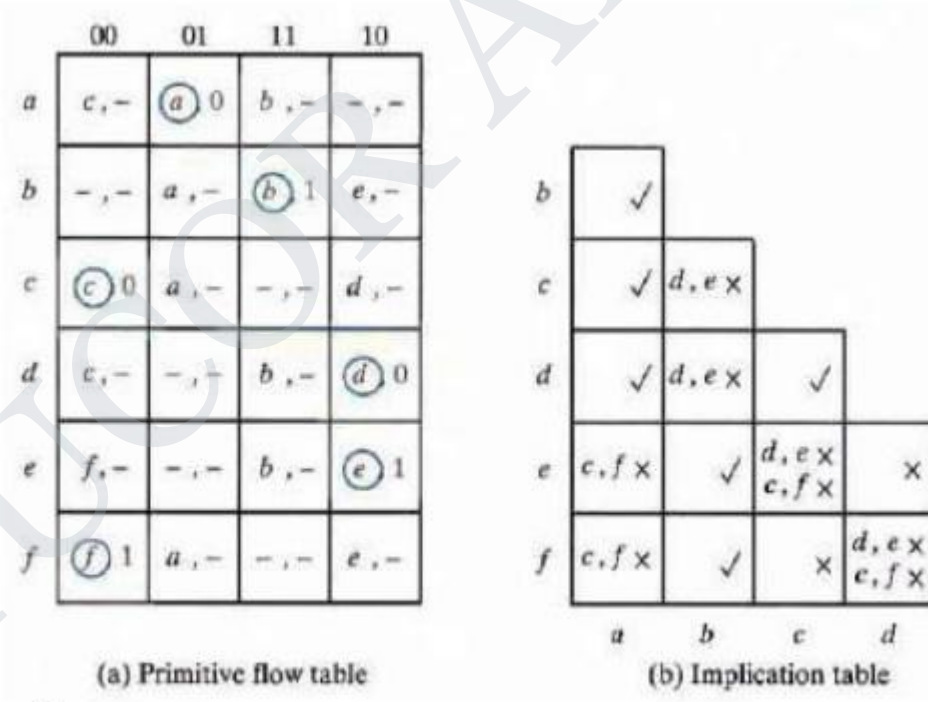
Incompletely specified states can be combined to reduce the number of state in the flow table. Such stares cannot be called equivalent because the formal definition of equivalence requires that all outputs and next states be specified for all inputs. Instead, two incompletely specified states that can be combined

patible. The process that must be applied in order to find a suitable group of compatibles for the purpose of merging a flow table can be

divided into three steps:

1. Determine all compatible pairs by using the implication table.
2. Find the maximal compatibles with the use of a merge r diagram.
3. Find a minimal collection of compatibles that covers all the states and is closed

✓**Compatible Pairs**-The entries in each square of primitive flow table represent the next state and output The dashes represent the unspecified states or outputs. The implication table is used to fmd compatible states just as it is used to find equivalent stales in the completely specified case. The only difference is that, when comparing rows, we are at liberty to adjust the dashes to fit any desired condition.



	00	01	11	10
a	c, –	(a) 0	b, –	–, –
b	–, –	a, –	(b) 1	e, –
c	(c) 0	a, –	–, –	d, –
d	c, –	–, –	b, –	(d) 0
e	f, –	–, –	b, –	(e) 1
f	(f) 1	a, –	–, –	e, –

(a) Primitive flow table

	a	b	c	d
b	✓			
c	✓	d, e ×		
d	✓	d, e ×	✓	
e	c, f ×	✓	d, e × c, f ×	×
f	c, f ×	✓	×	d, e × c, f ×

(b) Implication table

The compatible pairs are,
$(a,b)(a,c)(a,d)(b,e)(b,f)(c,d)(e,f)$

✓ **Maximal Compatibles**

The *maximal compatible* is a group of compatibles that contains all the possible combinations of compatible states. The maximal compatible can be obtained from a merger diagram. The merger diagram is a graph in which each state is represented by a dot placed along the circumference of a circle. Lines are drawn between any two corresponding dots that form a compatible pair. All possible compatibles can be obtained from the merger diagram by observing the geometrical patterns in which states are connected to each other. An isolated

represents a compatible pair. A triangle constitutes a compatible with three states.

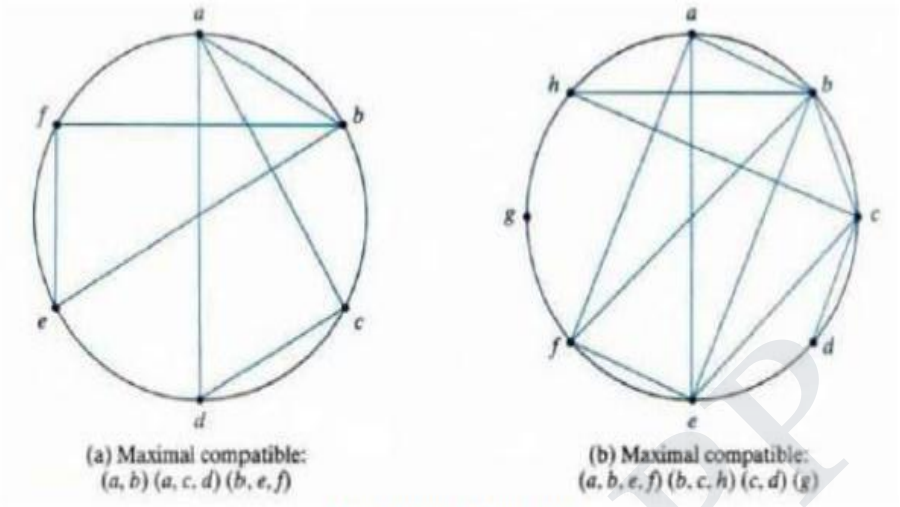


Fig: Merger diagrams

The maximal compatibles of fig (a) are$(a,b)(a,c,d)(b,e,f)$

The maximal compatibles of fig (b) are$(a,b,e,f)(b,c,h)(c,d)(g)$

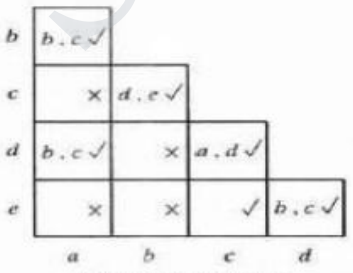
The maximal compatibles of fig (a) are

The maximal compatibles of fig (b) are

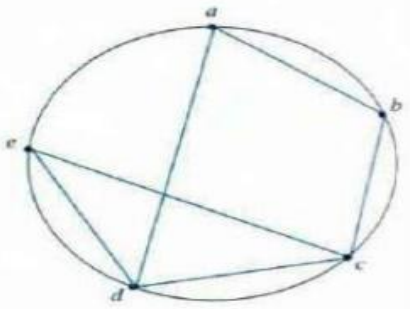
Closed-Covering Condition

The condition that must be satisfied for merging rows is that the set of chosen compatibles must *cover* all the states and must be *closed.* The set will cover all the states if it includes all the states of the original state table. The closure condition is satisfied if there are no implied states or if the implied states are included within the set. A closed set of compatibles that covers all the states is called a *closed covering.*

Example:



(a) Implication table



(b) Merger diagram

Compatibles	(a, b)	(a, d)	(b, c)	(c, d, e)
Implied states	(b, c)	(b, c)	(d, e)	(a, d) (b, c)

(c) Closure table

UNIT V

MEMORY AND PROGRAMMABLE LOGIC

Part A – 2 Marks

11. List basic types of programmable logic devices.

. Read only memory
. Programmable logic Array
. Programmable Array Logic

2 Explain ROM

A read only memory (ROM) is a device that includes both the decoder and the OR gates within a single IC package. It consists of n input lines and m output lines. Each bit combination of the input variables is called an address. Each bit combination that comes out of the output lines is called a word. The number of distinct addresses possible with n input variables is 2^n.

3. Define address and word

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.

4. What is programmable logic array? How it differs from ROM?

In some cases the number of don't care conditions is excessive, it is more economical to use a second type of LSI component called a PLA. A PLA is similar to a ROM in concept; however it does not provide full decoding of the variables and does not generates all the minterms as in the ROM.

12. What is memory and draw the block diagram of memory cell. ***Refer note***

13. What is the difference between programmable array logic (PAL) and programmable logic array (PLA) **refer note**

7.. State the types of ROM

- Masked ROM.
- Programmable Read only Memory
- Erasable Programmable Read only memory.
- Electrically Erasable Programmable Read only Memory.

14. How error is detected and corrected in digital system . refer note

9.Draw the timing diagram of Write operation. Refer note

10.What is the significance of FPGA . ***refer note***

Part B

17. Write notes on RAM, its operation and its types

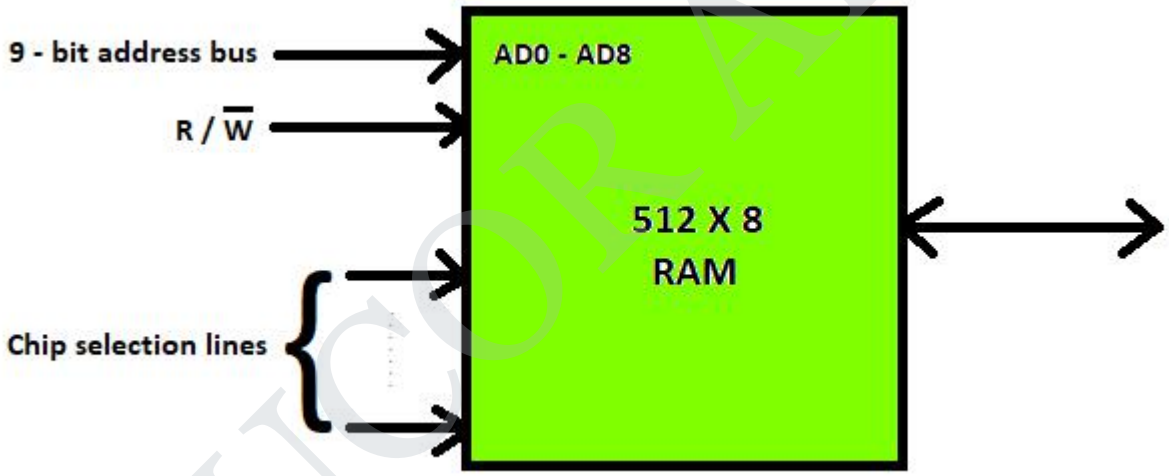
Memory)

RAM(Random Access Memory) is a part of computer's Main Memory which is directly accessible by CPU. RAM is used to Read and Write data into it which is accessed by CPU randomly. RAM is volatile in nature, it means if the power goes off, the stored information is lost. RAM is used to store the data that is currently processed by the CPU. Most of the programs and data that are modifiable are stored in RAM.

Integrated RAM chips are available in two form:

1. SRAM(Static RAM)
2. DRAM(Dynamic RAM)

The block diagram of RAM chip is given below.



SRA

Types of DRAM

There are mainly 5 types of DRAM:

1. **Asynchronous DRAM (ADRAM):** The DRAM described above is the asynchronous type DRAM. The timing of the memory device is controlled asynchronously. A specialized memory controller circuit generates the necessary control signals to control the timing. The CPU must take into account the delay in the response of the memory.
2. **Synchronous DRAM (SDRAM):** These RAM chips' access speed is directly synchronized with the CPU's clock. For this, the memory chips remain ready for operation when the CPU expects them to be ready. These memories operate at the CPU-memory bus without imposing wait states. SDRAM is commercially available as modules g multiple SDRAM chips and forming the required

3. **Double-Data-Rate SDRAM (DDR SDRAM):** This faster version of SDRAM performs its operations on both edges of the clock signal; whereas a standard SDRAM performs its operations on the rising edge of the clock signal. Since they transfer data on both edges of the clock, the data transfer rate is doubled. To access the data at high rate, the memory cells are organized into two groups. Each group is accessed separately.
4. **Rambus DRAM (RDRAM):** The RDRAM provides a very high data transfer rate over a narrow CPU-memory bus. It uses various speedup mechanisms, like synchronous memory interface, caching inside the DRAM chips and very fast signal timing. The Rambus data bus width is 8 or 9 bits.
5. **Cache DRAM (CDRAM):** This memory is a special type DRAM memory with an on-chip cache memory (SRAM) that acts as a high-speed buffer for the main DRAM.

Difference between SRAM and DRAM

Below table lists some of the differences between SRAM and DRAM:

SRAM	DRAM
1. SRAM has lower access time, so it is faster compared to DRAM.	1. DRAM has higher access time, so it is slower than SRAM.
2. SRAM is costlier than DRAM.	2. DRAM costs less compared to SRAM.
3. SRAM requires constant power supply, which means this type of memory consumes more power.	3. DRAM offers reduced power consumption, due to the fact that the information is stored in the capacitor.
4. Due to complex internal circuitry, less storage capacity is available compared to the same physical size of DRAM memory chip.	4. Due to the small internal circuitry in the one-bit memory cell of DRAM, the large storage capacity is available.
5. SRAM has low packaging density.	5. DRAM has high packaging density.

18. Discuss the operation of memory decoding and elaborate its application as address multiplexing and coincident decoding circuits

MEMORY DECODING

In addition to requiring storage components in a memory unit, there is a need for decoding circuits to select the memory word specified by the input address.

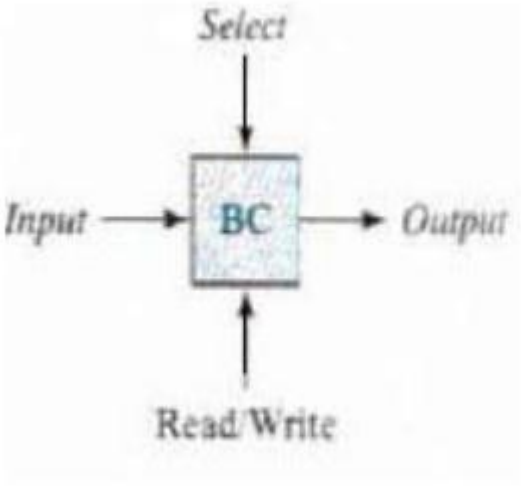


Fig: Block diagram of a memory cell

The storage part of the cell is modeled by an SR latch with associated gate s to form a D latch. Actually, the cell is an electronic circuit with four to six transistors. The select input enables the cell for reading or writing and the read/write input determines the operation of the cell when it is selected. A 1 in the read/write input provides the read operation by fanning a path from the latch to the output terminal. A 0 in the read/write input provides the write operation by forming a path from the input terminal to the latch.

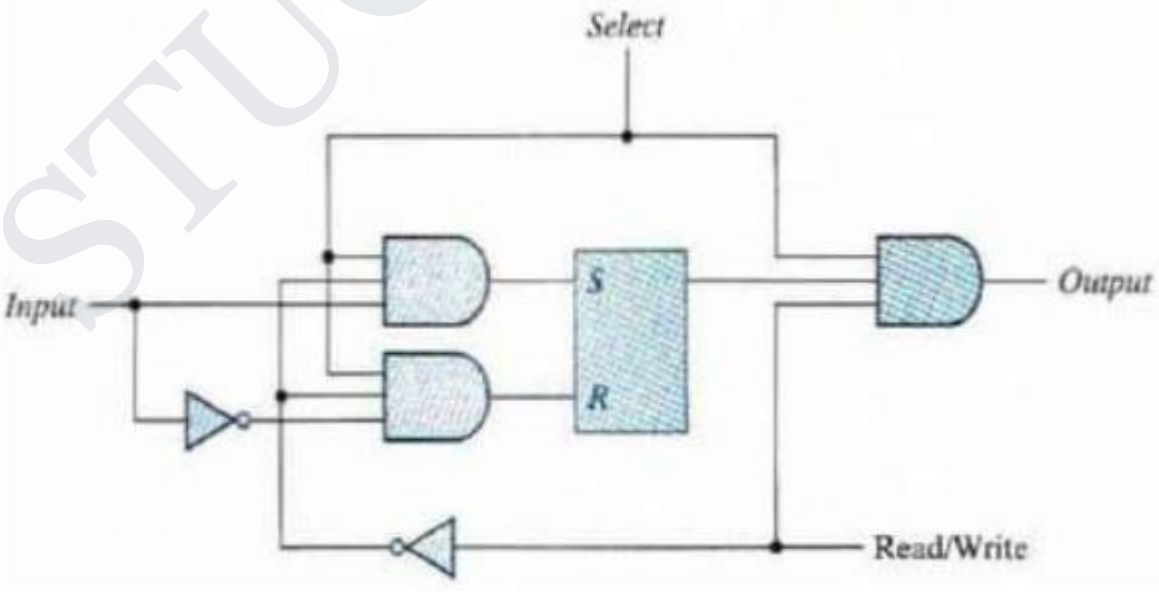


Fig: Logic diagram of a memory cell

The logical construction of a small RAM consists of four words of four bits each and bas a total of 16 binary cells. The small blocks labeled BC represent the binary cell with its three inputs and one output. A memory with four words

needs two address lines. The two address inputs go through a 2 4 decoder to select one of the four words. The decoder is enabled with the memory-enable input.

When the memory enable is 0, all outputs of the decoder are 0 and none of the memory words are selected. With the memory select at 1, one of the four words is selected, dictated by the value in the two address lines.

Once a word has been selected, the read/write input determines the operation. During the read operation the four bits of the selected word go through OR gates to the output terminals.

During the write operation, the data available in the input lines arc transferred into the four binary cells of the selected word. The binary cells that are not selected are disabled and their previous binary values remain unchanged.

When the memory select input that goes into the decoder is equal to 0 none of the words are selected and the contents of all cells remain unchanged regardless of the value of the read/write input.

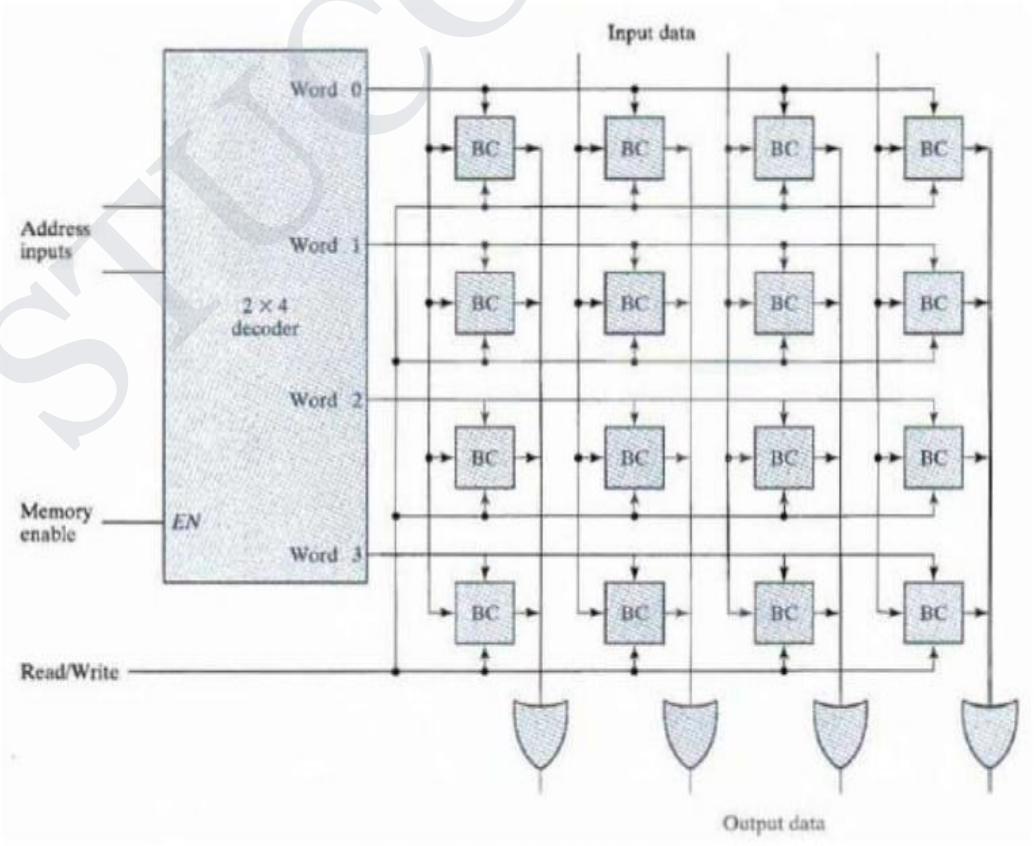


Fig: Diagram of a 4×4 RAM

Coincident Decoding

A decoder with *k* inputs and 2^k outputs requires 2^k AND gates with *k* inputs per gate. The total number of gates and the number of inputs per gate can be reduced by employing two decoders in a two - dimensional selection scheme.

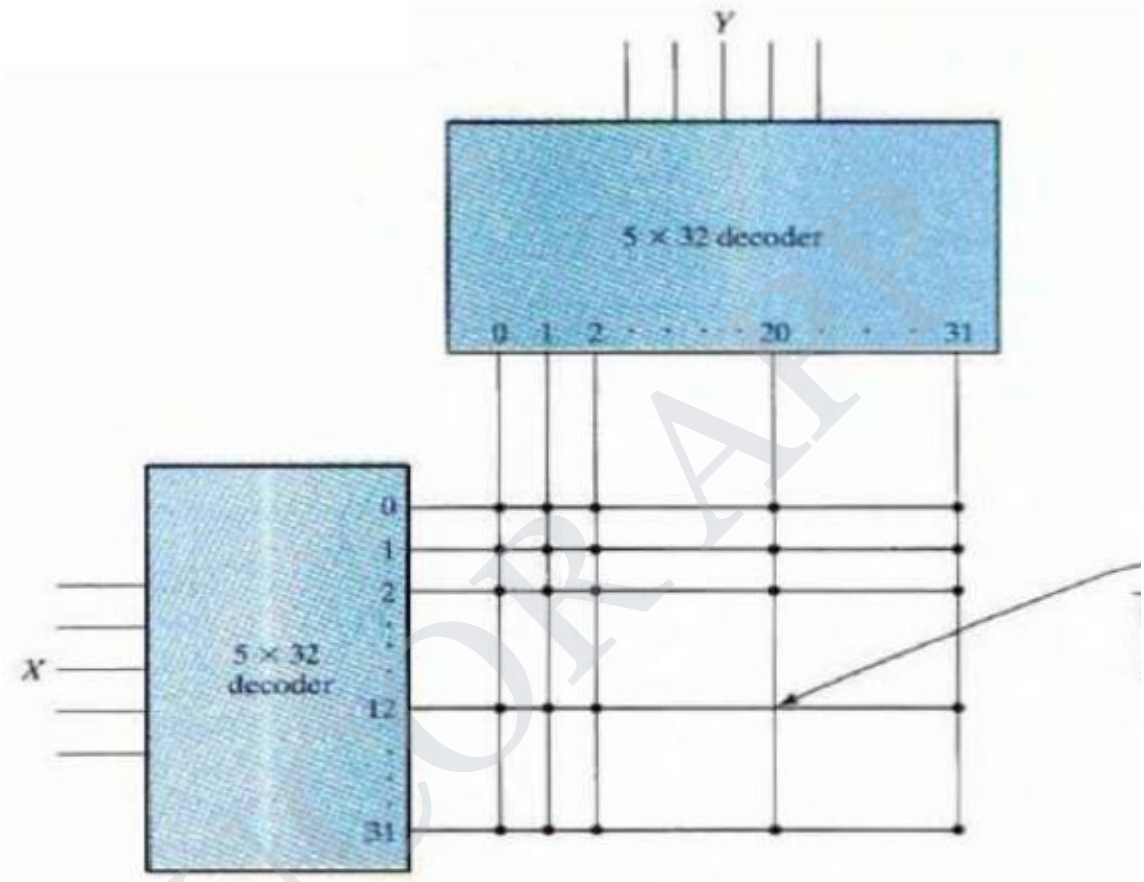


Fig: Two dimensional decoding structure for a 1K-word mei

The basic idea in two -dimensional decoding is to arrange the memory cells in an array, (i.e.) two $k/2$-

input decoders are used instead of one k-input decoder. One decoder performs the row selection and the other the column selection in a two-dimensional matrix configuration.

For example, instead of using a single 10 x 1,024 decoder, we use two 5 x 32 decoders. With the single decoder, we would need 1,024 AND gates with 10 inputs in each. The five most significant bits of the address go to input X and

array is selected by the coincidence of one X line and one Y line. Thus each

word in memory is selected by the coincidence between 1 of 32 rows and 1 of 32 columns, for a total of 1,024 words.

Address Multiplexing

Because of large capacity, the address decoding of DRAM is arranged in a two dimensional array and larger memories often have multiple arrays. To reduce the number of pins in the IC package, designers utilize address multiplexing whereby one set of address input pins accommodates the address components.

In a two-dimensional array, the address is applied in two parts at different times, with the row address first and the column address second. Since the same set of pins is used for both parts of the address, the size of the package is decreased significantly.

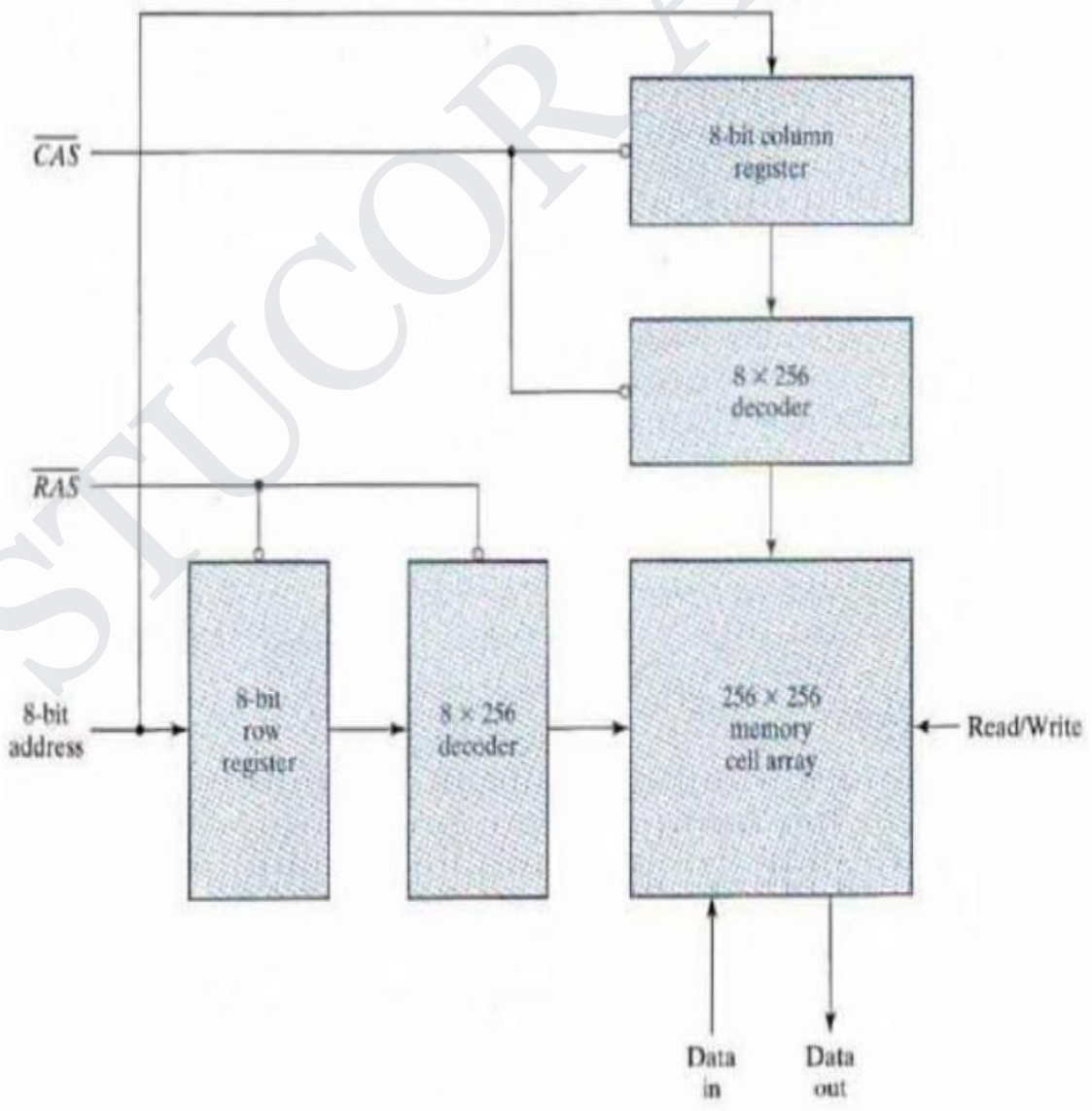


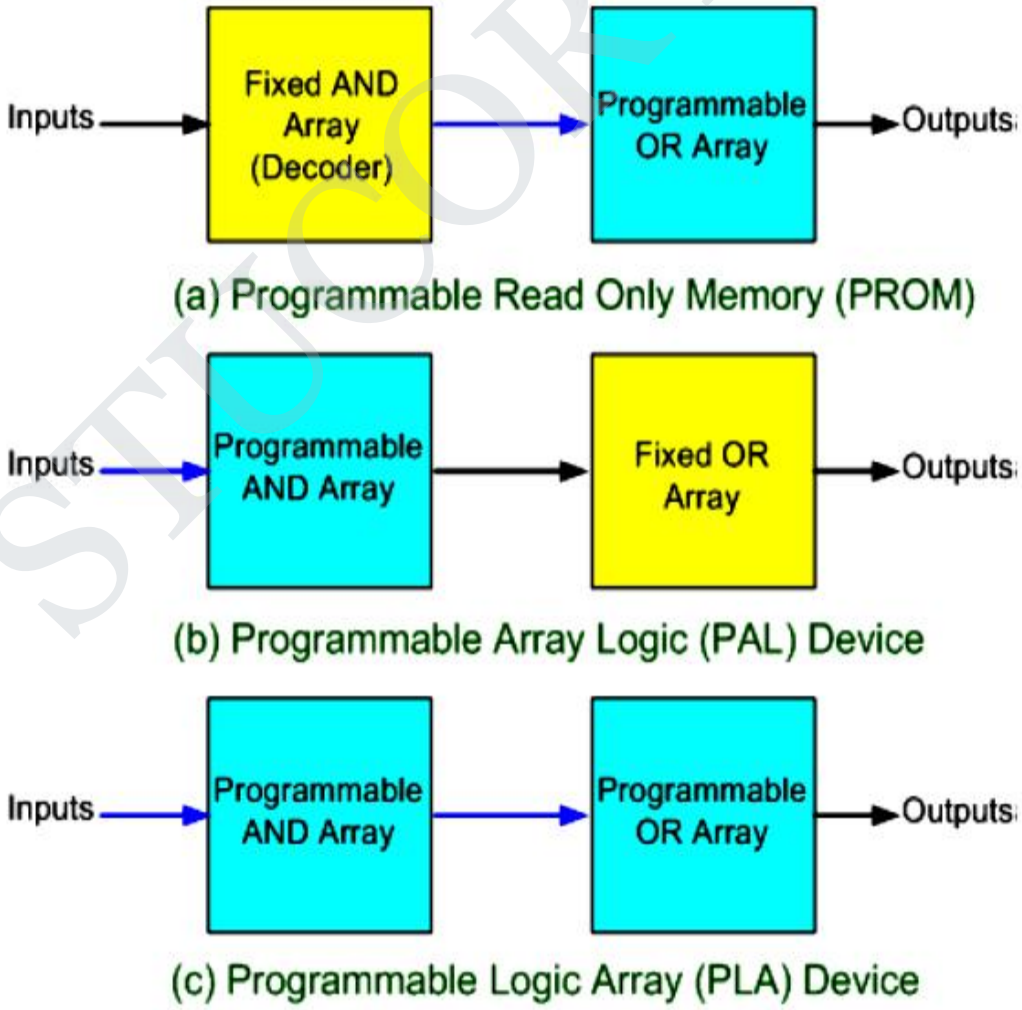
Fig: Address multiplexing for a 64K DRAM

The memory consists of a two-dimensional array of cells arranged into 256 rows by 256 columns, for a total of $2^8 \times 2^8 = 2^{16} = 64K$ words. There is a single data input line; a single data output line, and a read/write control as well as an eight-bit address input and two address *strobes,* the latter included for-enabling the row and column address into their respective registers. The row address strobe (RAS) enables the eight-bit row register and the column address strobe (CAS) enables the eight-bit column register. The bar on top of the name of the strobe symbol indicates that the registers are enabled on the zero level of the signal.

4. a. Comparison between PROM, PLA and PAL (6)

Three Fundamental Types of PLDs:

The three fundamental types of PLDs differ in the placement of programmable connections in the AND-OR arrays. Figure shows the locations of the programmable connections for the three types.



The PROM (Programmable Read Only Memory) has a fixed AND array

OM implements Boolean functions in sum-of-min terms (10)

form.

The PAL (Programmable Array Logic) device has a programmable AND array and fixed connections for the OR array.

The PLA (Programmable Logic Array) has programmable connections for both AND and OR arrays. So it is the most flexible type of PLD.

The ROM (Read Only Memory) or PROM (Programmable Read Only Memory):

The input lines to the AND array are hard-wired and the output lines to the OR array are programmable.

Each AND gate generates one of the possible AD products (.e., i minterms).

In the previous lesson, you have learnt how to implement a digital circuit using ROM.

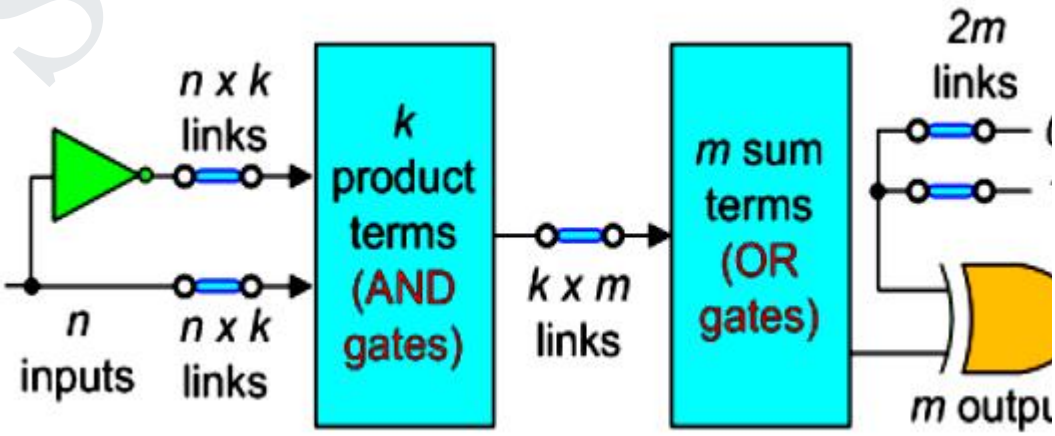
The PLA (Programmable Logic Array):

In PLAs, instead of using a decoder as in PROMs, a number (**k**) of AND gates is used where **k < 2n,** (**n** is the number of inputs).

Each of the AND gates can be programmed to generate a product term of the input variables and does not generate all the minterms as in the ROM.
The AND and OR gates inside the PLA are initially fabricated with the links (fuses) among them.

The specific Boolean functions are implemented in sum of products form by opening appropriate links and leaving the desired connections.

A block diagram of the PLA is shown in the figure. It consists of **n** inputs, **m** outputs, and **k** product terms.



The product terms constitute a group of **k** AND gates each of 2n inputs.

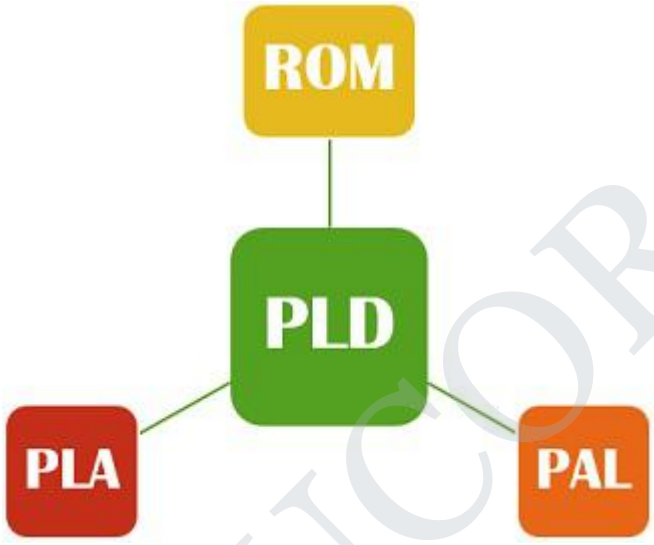
b. Realise the function gives using a PLA with 6 Input, 4 Outputs and 10 AND gates
F1(A,B,C,D,E,F) = ∑(0,1,7,8,9,10,11,15,19,23,27,31,32,33,35,39,40,41,47,63)

20. F2(A,B,C,D,E,F) = ∑(8,9,10,11,12,14,21,25,27,40,41,42,43,44,46,57,59) using PAL

21. Write notes on PLA and PAL

Difference Between PLA and PAL

October 23, 2018 Leave a Comment



PLA and PAL are types of Programmable Logic Devices (PLD) which are used to design combination logic together with sequential logic. The significant difference between the PLA and PAL is that the PLA consists of the programmable array of AND and OR gates while PAL has the programmable array of AND but a fixed array of OR gate. PLD's provides a more simple and flexible way of designing the logic circuits where the number of functions can also be increased. These are also implemented in IC.

Before PLD's, multiplexers were used for designing a combinational logic circuit, these circuits were highly complex and rigid.
Then **Programmable logic devices** (PLD) are developed, and the first PLD was ROM. ROM design was not very successful as it emerged the issue of hardware wastage and increasing exponential growth in the hardware for every large application. To overcome the limitations of ROM, PLA and PAL were devised. PLA and PAL are programmable and

effectively utilizes the hardware.

Content: PLA Vs PAL

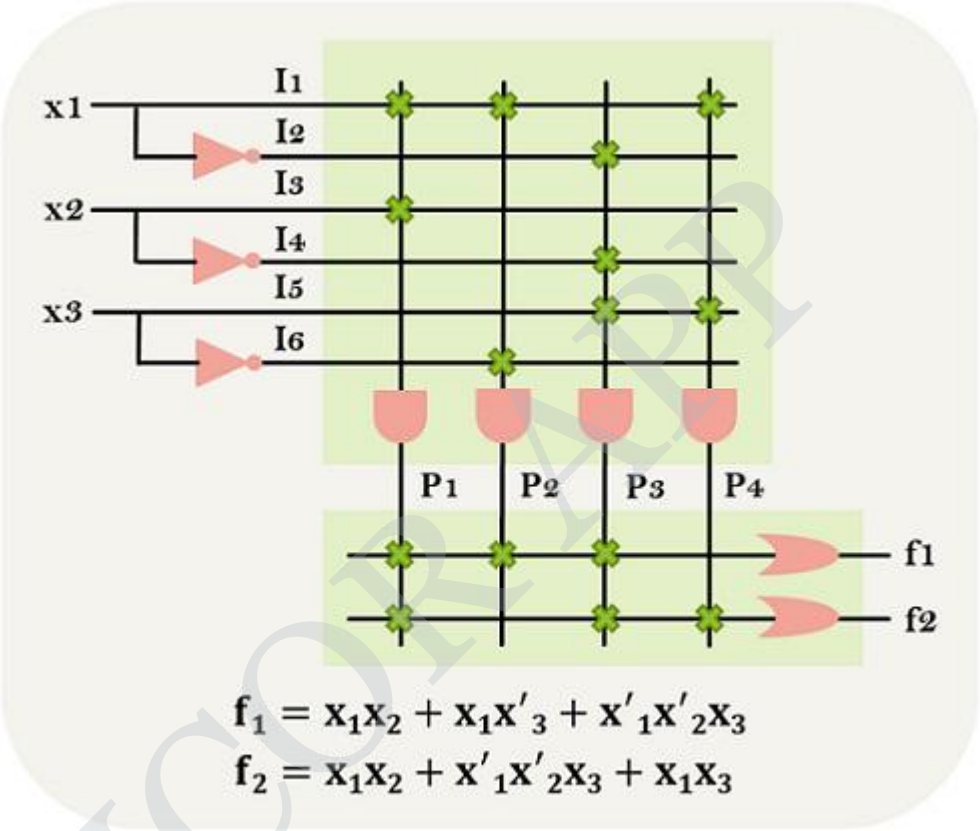
1.
    1. Comparison Chart
    2. Definition
    3. Key Differences
    4. Conclusion

Comparison Chart

BASIS FOR COMPARISON	PLA	PAL
Stands for	Programmable Logic Array	Programmable Array Logi
Construction	Programmable array of AND and OR gates.	Programmable array of AN fixed array of OR gates.
Availability	Less prolific	More readily available
Flexibility	Provides more programming flexibility.	Offers less flexibility, but used.
Cost	Expensive	Intermediate cost
Number of functions	Large number of functions can be implemented.	Provides the limited numb
Speed	Slow	High

Definition of PLA

boolean function in the SOP (Sum of Products) form. The PLA contains NOT, AND and OR gates fabricated on the chip. It passes every input by a NOT gate which makes each input and its complement available to every AND gate. The output of each AND gate is given to the each OR gate. At last, the OR gate output produces chip output. So, this is how suitable connections are made to employ SOP



Programmable Logic Array

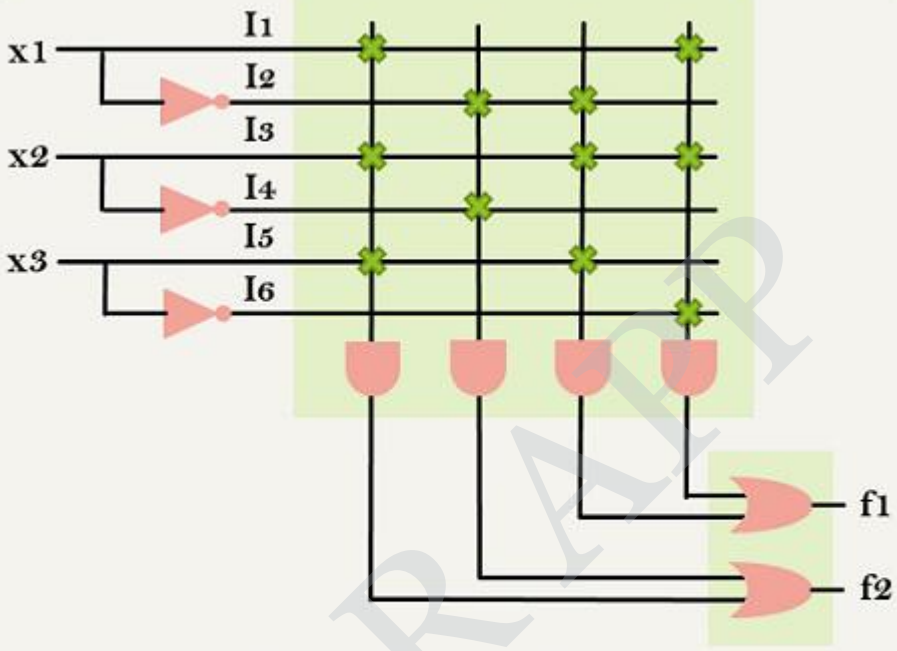
expressions.

In PLA the connections to both AND and OR arrays are programmable. PLA is considered more expensive and complex as compared to the PAL. The two different manufacturing techniques can be used for PLA to increase the ease of programming. In this technique, each connection is built through a fuse at every intersection point where the unwanted connections can be removed by blowing the fuses. The latter technique involves the connection making at the time of the fabrication process with the help of the proper mask provided for the specific interconnection pattern.

Definition of PAL

PAL **(Programmable Array Logic)** is also a PLD (Programmable Logic Device) circuit which works similar to the PLA. PAL employs the ND gates but fixed OR gates, unlike PLA. It implements

gate specifies the maximum number of product terms that can be generated in a sum-of-products representation of the particular function. While the AND gates are perpetually connected to the OR gates, which signifies that the produced product term is not shareable with the output



$$f_1 = x_1x_2x'_3 + x'_1x_2x_3$$
$$f_2 = x'_1x'_2 + x_1x_2x_3$$

Programmable Array Logic

functions.

The main concept behind developing PLD's is to embed a complex boolean logic into a single chip. Therefore, eliminating the unreliable wiring, preventing the logic design and minimizing power consumption.

Key Differences Between PLA and PAL

1. The PLA is PLD, comprised of two levels of programmable logic AND plane and OR plane. On the other hand, PAL contains only programmable AND plane and fixed OR plane.
2. When it comes to availability, the PAL is more readily available along with easy production. In contrast, the PLA is not easily available.
3. The PLA is more flexible than a PAL.
4. PLA is costlier as compared to the PAL.

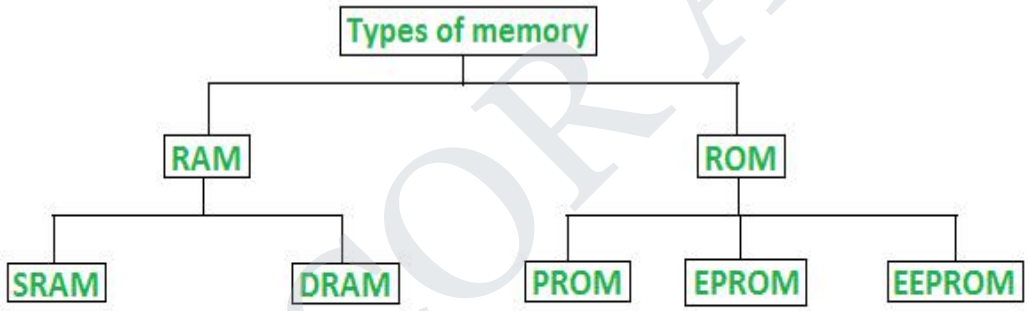
it enables the programming of the OR plane also.

6. PAL works faster while PLA is slower comparatively.

22. Define Memory and discuss the operation & types of RAM and ROM

Types of computer memory (RAM and ROM)

Memory is the most essential element of a computing system because without it computer can't perform simple tasks. Computer memory is of two basic type – Primary memory / Volatile memory and Secondary memory / non-volatile memory. Random Access Memory (RAM) is volatile memory and Read Only Memory (ROM) is non-volatile memory.



Classification of computer memory

1. Random Access Memory (RAM) –

- It is also called as *read write memory* or the *main memory* or the *primary memory*.
- The programs and data that the CPU requires during execution of a program are stored in this memory.
- It is a volatile memory as the data loses when the power is turned off.
- RAM is further classified into two types- *SRAM (Static Random Access Memory)* and *DRAM (Dynamic Random Access Memory)*.

DRAM	SRAM
1. Constructed of tiny capacitors that leak electricity.	1.Constructed of circuits similar to D flip-flops.
2.Requires a recharge every few milliseconds to maintain its data.	2.Holds its contents as long as power is available.
3.Inexpensive.	3.Expensive.
4. Slower than SRAM.	4. Faster than DRAM.
5. Can store many bits per chip.	5. Can not store many bits per chip.
6. Uses less power.	6.Uses more power.
7.Generates less heat.	7.Generates more heat.
8. Used for main memory.	8. Used for cache.

Difference between SRAM and DRAM

2. Read Only Memory (ROM) –

- Stores crucial information essential to operate the system, like the program essential to boot the computer.
- It is not volatile.
- Always retains its data.
- Used in embedded systems or where the programming needs no change.
- Used in calculators and peripheral devices.
- ROM is further classified into 4 types- *ROM*, *PROM*, *EPROM*, and *EEPROM*.

Types of Read Only Memory (ROM) –

1. **PROM (Programmable read-only memory)** – It can be programmed by user. Once programmed, the data and instructions in it cannot be changed.
2. **EPROM (Erasable Programmable read only memory)** – It can be reprogrammed. To erase data from it, expose it to ultra violet light. To reprogram it, erase all the previous data.
3. **EEPROM (Electrically erasable programmable read only memory)** – The data can be erased by applying electric field, no need of ultra violet light. We can erase only portions of the chip.

RAM	ROM
1. Temporary Storage.	1. Permanent storage.
2. Store data in MBs.	2. Store data in GBs.
3. Volatile.	3. Non-volatile.
4.Used in normal operations.	4. Used for startup process of computer.
5. Writing data is faster.	5. Writing data is slower.

Difference between RAM and ROM

23. Elaborate the construction of sequential programmable devices in detail

SEQUENTIAL PROGRAMMABLE DEVICES Digital systems are designed with flip-flops and gates. Since the combinational PLD consists of only gates, it is necessary to include external flip-flops when they are used in the design. Sequential programmable devices include both gates and flip-flops. In this way, the device can be programmed to perform a variety of sequential-circuit functions. There are several types of sequential programmable devices available commercially, and each device has vendor-specific variants within each type. The internal logic of these devices is too complex to be shown here. Therefore, we will describe three major types without going into their detailed construction: 1. Sequential (or simple) programmable logic device (SPLD) 2. Complex programmable logic device (CPLD) 3. Field-programmable gate array (FPGA) 1. Sequential (or simple) programmable logic device (SPLD) The sequential PLD is sometimes referred to as a simple PLD to differentiate it from the complex PLD. The SPLD includes flip-flops, in addition to the AND–OR array, within the integrated circuit chip. The result is a sequential circuit as shown in Fig. 1. A PAL or PLA is modified by including a number of flip-flops connected to form a register. The circuit outputs can be taken from the OR gates or from the outputs of the flip-flops. Fig 1. Sequential Programmable logic device Additional programmable connections are available to include the flip-flop outputs in the product terms formed with the AND array. The flip-flops may be of the D or the JK type. The configuration mostly used in an SPLD is the combinational PAL together with D flip-flops. A PAL that includes flip-flops is referred to as a registered PAL, to signify that the device contains flip-flops in addition to the AND–OR array. Each section of an SPLD is called a macrocell, which is a circuit that contains a sum-of-products combinational logic function and an optional flip-flop. Fig. 2 shows the logic of a basic macrocell. The AND–OR array is the same as in the combinational PAL. The output is driven by an edge-triggered D flip-flop connected to a common clock input and changes state on a clock edge. The output of the flip-flop is connected to a three-state buffer (or inverter) controlled by an output-enable signal marked in the diagram as OE. The output of the flip-flop is fed back into one of the inputs of the programmable AND gates to provide the present-state condition for the sequential circuit. A typical SPLD has from 8 to 10 macrocells within one IC package. All the flip-flops are connected to the common CLK input, and all three-state buffers are controlled by the OE input. Fig 2. Basic Macrocell 2. Complex programmable logic device (CPLD) The design of a digital system using PLDs often requires the connection of several devices to produce the complete specification. For this type of application, it is more economical to use a complex programmable logic device (CPLD), which is a collection of individual PLDs on a single integrated circuit. A programmable interconnection structure allows the PLDs to be connected to each other in the same way that can be done with individual PLDs. Fig. 3 shows the general configuration of a CPLD. The device consists of multiple PLDs interconnected through a programmable switch matrix. The input–output (I/O) blocks provide the connections to the IC pins. Each I/O pin is driven by a three state buffer and can be programmed to act as input or output. The switch matrix receives inputs from the I/O block and directs them to the individual macrocells. Similarly, selected outputs from macrocells are sent to the outputs as needed. Each PLD typically contains from 8 to 16 macrocells, usually fully connected. If a macrocell has unused product terms, they can be used by other nearby macrocells. In some cases the macrocell flip-flop is programmed to act

Fig3. General CPLD configuration 3. Field-programmable gate array (FPGA) A

typical FPGA consists of an array of millions of logic blocks, surrounded by programmable input and output blocks and connected together via programmable interconnections. A typical FPGA logic block consists of lookup tables, multiplexers, gates, and flip-flops. A lookup table is a truth table stored in an SRAM and provides the combinational circuit functions for the logic block. The combinational logic section, along with a number of programmable multiplexers, is used to configure the input equations for the flip-flop and the output of the logic block. The program can be downloaded either from a host computer or from an onboard PROM. The program remains in SRAM until the FPGA is reprogrammed or the power is turned off. The device must be reprogrammed every time power is turned on. The ability to reprogram the FPGA can serve a variety of applications by using different logic implementations in the program. The design with PLD, CPLD, or FPGA requires extensive computer-aided design (CAD) tools to facilitate the synthesis procedure. Among the tools that are available are schematic entry packages and hardware description languages (HDLs), such as ABEL, VHDL, and Verilog Synthesis tools

24. Explain ASIC in detail

10. Implement following function using PLA (8)

Z1= ab‘d’e+a’b’c’d’e’+bc+de, Z2=a’c’e, Z3=bc+de+c’d’e+bd, Z4=a’c’e+ce us
5x8x4 PLA

11. Implement the two following Boolean function using 8x2 PROM. (8)

F (x,y,z) = 13,5,6,7
F (x,y,z) = 1,2,3,4