IMPORTANT QUESTIONS WITH ANSWERS

FOC {FONDAMENTAL OF COIMPUTERS}

Chapter -01 basic of logic design

01. Explain different types of number system **10marks**

02.conversion from one number to other number system **5 0r 10 marks**

03. Logic gates symbols with truth table **5 marks**

04.AND and OR GATE FOR 3 INPUTS **10 MARKS**

05.De margan's theorms **5 OR 10 MARKS**

06.List the Boolean laws and rules **5marks**

01. Explain different types of number system 10marks

https://youtube.com/
@Ndiplomainkannada

Four most common types of number system are

1. Binary number system (Base- 2)
2. Octal number system (Base-8)
3. Decimal number system (Base- 10)
4. Hexadecimal number system (Base- 16)

1. Binary number system:

- **It is the type of Number Representation techniques, in which base is 2.**
- **It is most popularly used in digital devices like computers.**
- **Binary system is used for representing binary quantities**
- **Digits used are: 0 & 1.**
- **Eg: 110011**

2. Octal number system:

- **It is the type of Number Representation technique, in which base is 8.**
- **There are only 8 symbols or possible digit values, they are 0, 1, 2, 3, 4, 5, 6, 7.**
- **It requires only 3 bits to represent value of any digit.**

3. Decimal number system:

 It is a type of Number Representation technique, in which base is 10

 It is also called as Hindu-Arabic number system, used since ancient days.

 It is a base 10 number system. Digits used are: 0,1,2,3,4,5,6,7,8,9.

4. Hexadecimal number system

 It is a type of Number Representation technique, in which base is 16.

 Digits used are: 0,1,2,3,4,5,6,7,8,9 Letters used A-F.

 It requires only 4 bits to represent value of any digit.

 Hexadecimal numbers are indicated by the addition of either an 0x prefix or an h suffix

02.conversion from one number to other number system **5 0r 10 marks**

01.Binary to decimal conversion

02.Decimal to binary conversion

03.Octal to decimal conversion

04.Decimal to octal conversion

05.Octal to binary conversion

06.Binary to octal conversion

07.Decimal to hexa-decimal conversion

08.Hexa-decimal to binary conversion

09.Binary to hexa-decimal conversion

Practice below conversion

1. (111011)2 = (59)10:

 1x25 + 1x24 + 1x23 + 0x22 + 1x21 + 1x20

 1x32 + 1x16 + 1x8 + 0x4 + 1x2 + 1x1 =

 32 + 16 + 8 + 0 + 2 + 1

 32 + 16 + 8 + 0 + 2 + 1

 59.

2. (246)10 = (11110110)2

Divide 246 by 2. Use the integer quotient obtained in this step as the dividend for

the next step. Repeat the process until the quotient becomes 0. Write the remainder

from bottom to top i.e. in the reverse chronological order. This will give the

binary equivalent of 246.

Dividend	Remainder
246/2 = 123	0
123/2 = 61	1
61/2 = 30	1
30/2 = 15	0
15/2 = 7	1
7/2 = 3	1
3/2 = 1	1
1/2 = 0	1

1/2 = 0 1

Therefore, the binary equivalent of decimal number 246 is 11110110

3. 999 (10) to Octal:

Divide 999 by 8 as below:

999/8=124 with remainder 7

124 / 8 = 15 with remainder 4

15 / 8 = 1 with remainder 7

1 / 8 = 0 with remainder 1

Rewrite the remainders from bottom to top. Result: 1747(8)

4. 10101111(2) to Hexadecimal

First, convert 101011112 into decimal as below:

10101111(2) = 1 × 27 + 0 × 26 + 1 × 25 + 0 × 24 + 1 × 23 + 1 × 22 + 1 × 21 + 1 × 20

= 17510

Now convert 17510 to hexadecimal as below:

175 / 16 = 10 with remainder 15 (F)

10 / 16 = 0 with remainder 10 (A)

Result is: AF(16)

5.First, convert 123.778 into decimal, by using above steps:

= 1 × 82 +2 ×81 +3 × 8

0 +7 × 8-1 + 7 × 8-2

= 83.98437510

Now, we have to convert 83.98437510 to binary

83 / 2 = 41 with remainder 1

41 / 2 = 20 with remainder 1

20 / 2 = 10 with remainder 0

10 / 2 = 5 with remainder 0

5 / 2 = 2 with remainder 1

2 / 2 = 1 with remainder 0

1 / 2 = 0 with remainder 1

83 = 1010011 ------- (1)

0.984375 × 2 = 1 + 0.96875

0.96875 × 2 = 1 + 0.9375

0.9375 × 2 = 1 + 0.875

0.875 × 2 = 1 + 0.75

0.75 × 2 = 1 + 0.5

0.5 × 2 = 1 + 0

0.984375 = 0.111111------- (2)

Result: 1010011.111111(2)

6. (4BA)16= (1210)10

4x162 + Bx161 + Ax160

4x256 + 11x16 + 10x1 = (1210)10

So, the number 1210 is the decimal equivalent of hexadecimal number

4BA

.

7. (101101101)2 =(16D)16

Write the given binary number in groups of 4 starting from right.

1 0110 1101

 Each group of 4 equates to 0 through 9 and then A BCDEF, where A represents 10 and F is 15

 So 1101 equals 13 which is D

 0110 is 6

 And 1 is 1

 101101101 equates to 16D in hex.

8. (564)8 = (101110100)2

Take each octal number as input

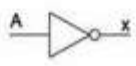
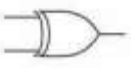
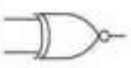
Convert each digit of octal into 3 digit binary number.

Octal Digit	Binary equivalent
4	100
6	110
5	101

 (564)8 equates to 101110100 in binary

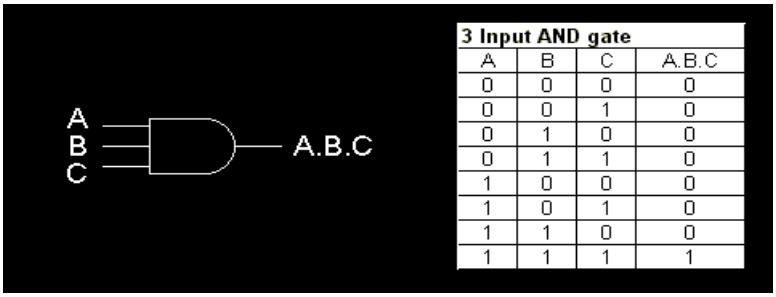
03. Logic gates symbols with truth table **5 marks**

Logic Gates

Name	NOT	AND	NAND	OR	NOR	XOR	XNOR
Alg. Expr.	$\overline{A}$	AB	$\overline{AB}$	$A+B$	$\overline{A+B}$	$A \oplus B$	$\overline{A \oplus B}$
Symbol							
Truth Table	A X 0 1 1 0	B A X 0 0 0 0 1 0 1 0 0 1 1 1	B A X 0 0 1 0 1 1 1 0 1 1 1 0	B A X 0 0 0 0 1 1 1 0 1 1 1 1	B A X 0 0 1 0 1 0 1 0 0 1 1 0	B A X 0 0 0 0 1 1 1 0 1 1 1 0	B A X 0 0 1 0 1 0 1 0 0 1 1 1

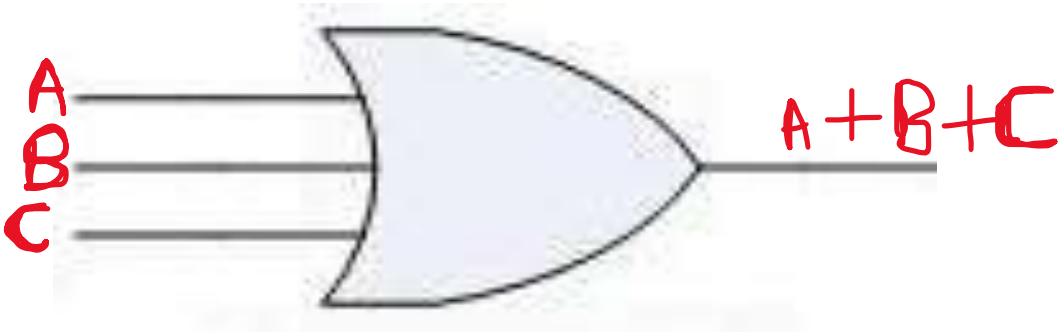
04.AND and OR GATE FOR 3 INPUTS **10 MARKS**

INPUT			OUTOUT
A	B	C	A.B.C
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0



.OR GATE FOR 3 INPUTS

INPUT			OUTOUT
A	B	C	A+B+C
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1



05.De margan's theorms 5 OR 10 MARKS

i) Demorgan's First law: $\overline{A.B} = \bar{A} + \bar{B}$ **(1m)**

Proof using TruthTable: **(4m)**

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

(or)

ii) Demorgan's Second law: $\overline{A + B} = \bar{A}.\bar{B}$ **(1m)**

Proof using Truth Table: **(4m)**

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

06.List the Boolean laws and rules 5marks

List the Boolean laws & rules.

Rules of Boolean algebra: Table lists the Basic rules of Boolean algebra.

1. $A + 0 = A$	7. $A \cdot A = A$
2. $A + 1 = 1$	8. $A \cdot \overline{A} = 0$
3. $A \cdot 0 = 0$	9. $\overline{\overline{A}} = A$
4. $A \cdot 1 = A$	10. $A + AB = A$
5. $A + A = A$	11. $A + \overline{A}B = A + B$
6. $A + \overline{A} = 1$	12. $(A + B)(A + C) = A + BC$

A, B, or *C* can represent a single variable or a combination of variables.

Laws of Boolean Algebra:

The basic laws of Boolean algebra-the commutative laws for addition and multiplication, the associative laws for addition and multiplication, and the distributive law-are the same as in ordinary algebra.

1. Commutative Laws:

1. The commutative law of addition for two variables is written as A+B = B+A

2. The commutative law of multiplication for two variables is written as A.B = B.A

2. Associative Laws:

1. The associative law of addition is written as follows for three variables:

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

2. The associative law of multiplication is written as follows for three variables: A(BC) = (AB)C **3. Distributive Law:**

1. The distributive law is written for three variables as follows:

A(B + C) = AB + AC

Chapter -02 LOGIC CIRCUITS

01.Design full adder with truth table.. **10 marks**

02. design half adder **10 marks**

03.describe half-subtractor with logic diagram and truth table. **10marks**

04.Describe 1 to 4 (1:4) demultiplexer **8 or 10marks**

05. Explain 4:1 multiplexer **10marks**

06.Design comparator circuit with truth table **10marks**

07. When 2 i/p of an SR flip-flop is set to high, the o/p is invalid, explain how this problem is resolved

08. Construct 4-bit SISO shift register

09. design 4:2 Encoder with truth table **10marks**

10. list different types of SHIFT REGISTER: **3marks**

11. list different types of FLIP-FLOPS **3marks**

12.deffrence between combination circuit and sequential circuit **4or 5marks**

13. Construct 4-bit asynchronous counter with logic circuit and truth table

4 bit asynchronous (ripple or serial) counter **10 marks**

01.Design full adder with truth table.. **10 marks**

2.2 Full Adder

A full adder is a combinational circuit that accepts two input binary digits (bits) and an input carry bit. It generates a sum output bit and an output carry bit.

☞ The logic symbol or block diagram of Full adder is shown in the figure below :

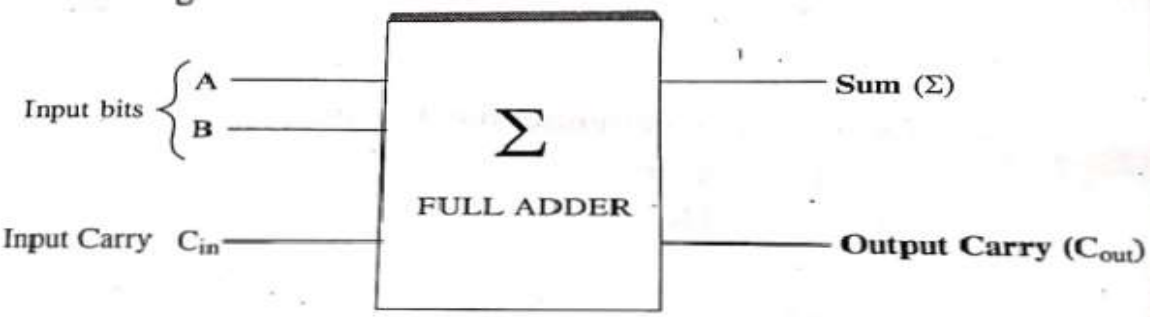


Figure ▲ Logic Symbol of a Full Adder.

☞ The operation of a Full Adder can be realised with the following Truth Table :

Input bits			Output bits	
A	B	C_{in}	Sum(Σ)	Output Carry 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

Figure ▲ Full Adder Truth Table.

The Full Adder operation / output can be expressed as an equation (boolean function):

$$\Sigma = (A \oplus B) \oplus C_{in}$$

☞ From the knowledge of Half adder sum of two input bits A and B is the exclusive-OR operation of A and B. Now the input carry bit C_{in} is to be added to A and B. In other words C_{in} must be exclusive-ORed with A and B for the complete full adder operation i.e., $A \oplus B \oplus C_{in}$ is the equation that generates Σ (sum) and C_{out} (output carry).

To conclude – the Full Adder logic diagram can be illustrated as shown in the figure below :

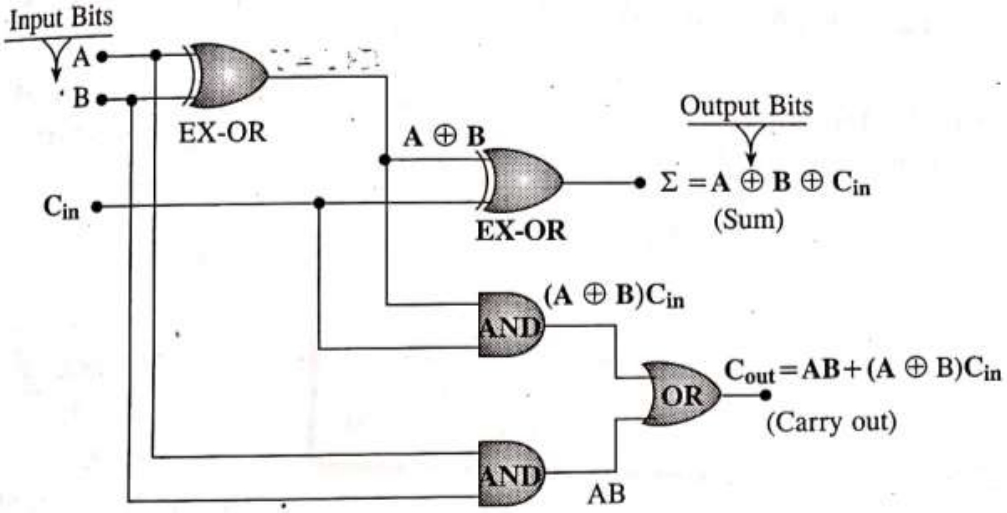
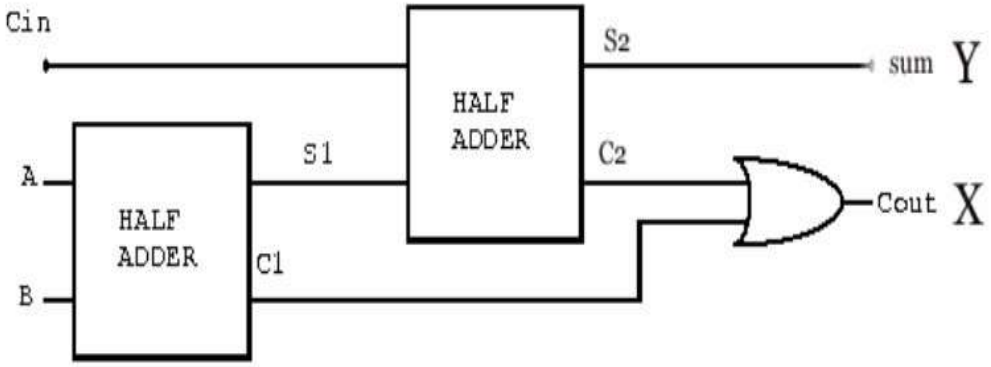


Figure ▲ The Complete logic circuit / logic diagram of a Full Adder.

02. design half adder 10 marks



Sum, X= (A ⊕ B) ⊕C_{in}(1m)

Carry, Y= A.B + C_{in}.(A ⊕ B) (1m)

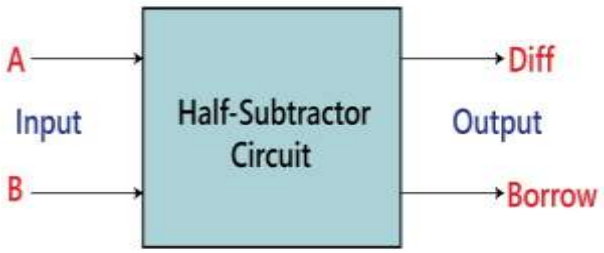
03.describe half-subtractor with logic diagram and truth table. 10marks

HALF SUBTRACTOR

- A Logic circuit used for the subtraction of two binary numbers is known as half subtractor.
- It has two inputs and two outputs.

- **A** and **B** as two **inputs** and **difference** and **borrow** as **outputs**.

Block diagram



Truth Table

Inputs		Outputs	
A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

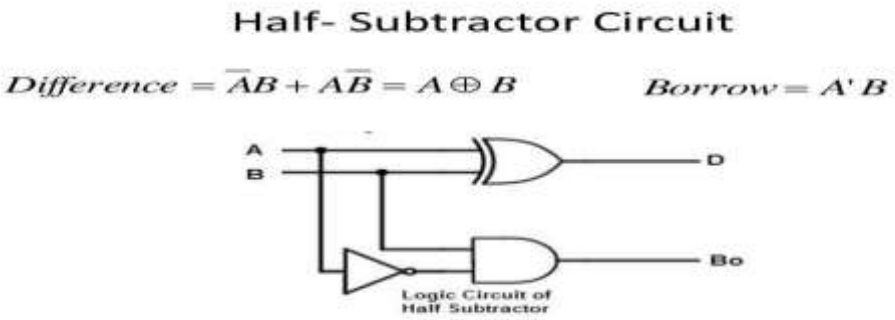
The SOP form of the diff and borrow are as follows:

Difference = A'B+AB' = A⊕B

Borrow = A'B

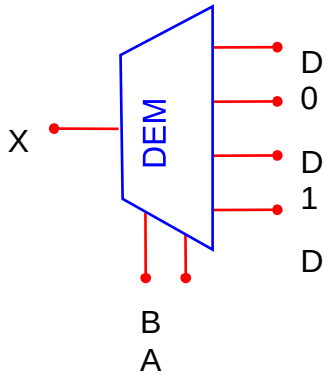
Logical circuit:

The Half Subtractor circuit can be designed and implemented using 'XOR' and 'AND' gates is as follows.



04.Describe 1 to 4 (1:4) demultiplexer 8 or 10marks

One to Four (1 to 4) line DEMUX:



A 1-to-4 demultiplexer has a single input (X), two selection lines (B and A) and four outputs (D0, D1, D2 and D3).

The input data bit is send to the data bit of the output lines depending on the value of the select input.

Block Diagram

TRUTH TABLE:

B	A	D	D	D	D
		o	1	2	4
0	0	X	0	0	0

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

When B=0 A=0, D0=X, similarly

B=0 A=1, D1= X

B=1 A=0, D2=X

B=1 A=1, D3=X

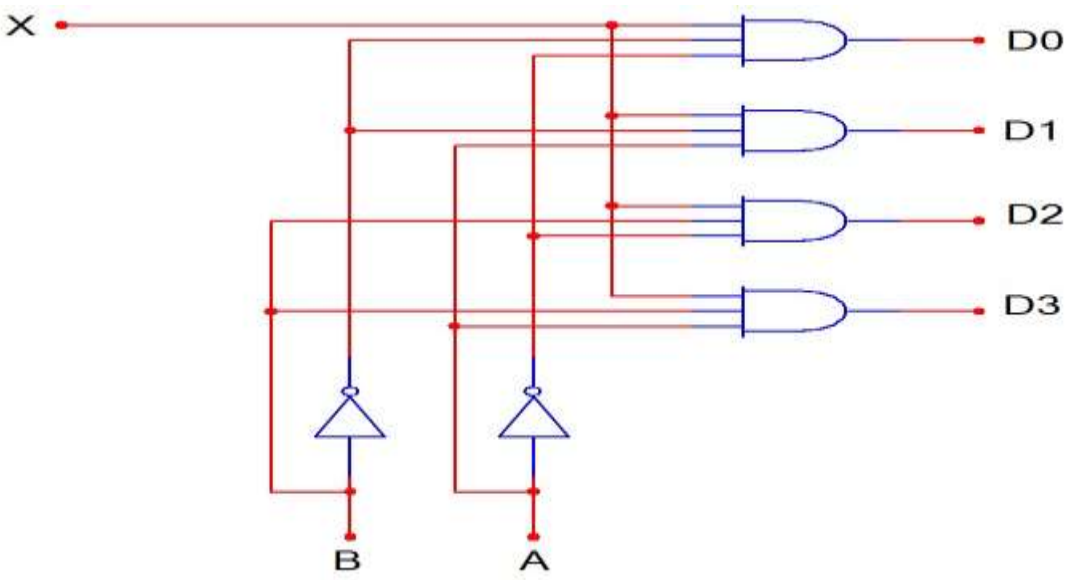
From the above Truth table, we can directly write the **Boolean Expressions** for each output

Do=$\overline{B}\ \overline{A}$ X

D1=$\overline{B}$ AX

D2=B$\overline{A}$ X

D3=BAX



LOGIC CIRCUIT

05. Explain 4:1 multiplexer 10marks

2.5.1 4:1 MUX

A 4:1 Mux is a multiplexer circuit having 4–input data lines and one–output line with two numbers of data–select control lines.

➤ With 2–control bits for data select (2^2) –one–of–4 data inputs can be selected at each time. Accordingly, the logic symbol for 4–input multiplexer (MUX) and the corresponding truth table is illustrated in the figure below :

Data Select/ Control Inputs		Input Selected
S_1	S_0	Y=
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

▲ Truth Table for Data Selection : 1–of–4

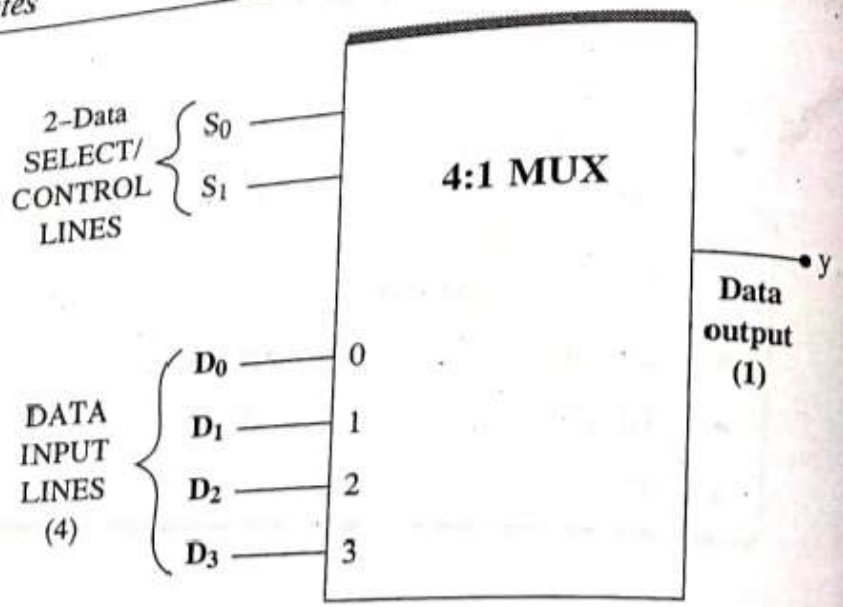


Figure ▲ Logic Symbol for 1:4 Multiplexer

The logic expression for the 4:1 multiplexer output can be derived interms of data input and data select/control input and is given as.

$$Y = D_0 \overline{S_1}\,\overline{S_0} + D_1 \overline{S_1} S_0 + D_2 S_1 \overline{S_0} + D_3 S_1 S_0$$

That is

- The data output $Y = D_0$ if and only if $S_1 = 0,\ S_0 = 0$
  i.e, $Y = D_0 \overline{S_1}\,\overline{S_0}$
- The data output $Y = D_1$ If and only if $S_1 = 0,\ S_0 = 1$
  i.e., $Y = D_1 \overline{S_1} S_0$
- The data output $Y = D_2$ If and only if $S_1 = 1,\ S_0 = 0$
  i.e., $Y = D_2 S_1 \overline{S_0}$
- The data output $Y = D_3$ if and only if $S_1 = 1,\ S_0 = 1$
  i.e,. $Y = D_3 S_1 S_0$.

To conclude implementation of 4:1 MUX logical equation requires 4-number of 3-input AND gates and a single 4-input OR gate and 2-number of NOT gates (Inverters) to generate complements of S_0 and S_1.

➢ The logic diagram for a 4-input (4:1 MUX) Multiplexer circuit is illustrated in the figure below :

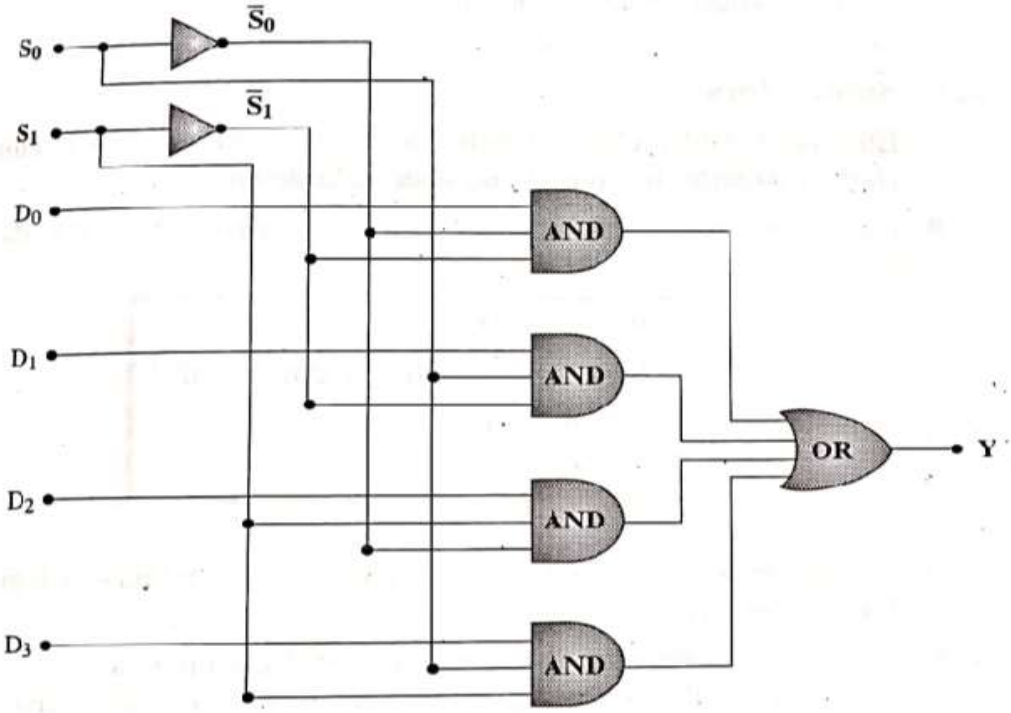
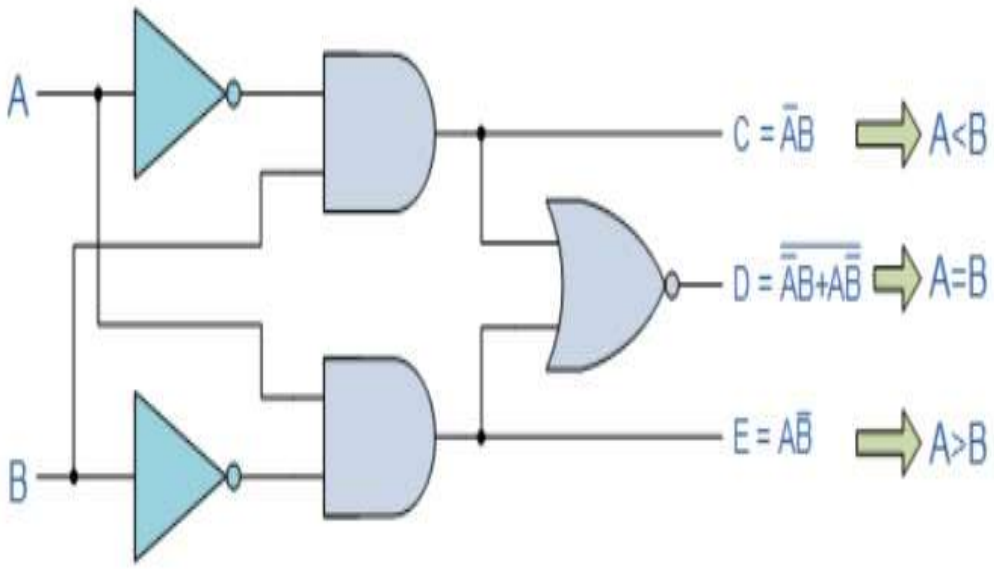


Figure ▲ Logic diagram of 4:1 MUX

☞ Thus Data input can be selected from any one of 4—input lines D_0, D_1, D_2 and D_3 this multiplexer circuit is also termed as *Data selector*.

06.Design comparator circuit with truth table 10marks

Comparator circuit : (3m)

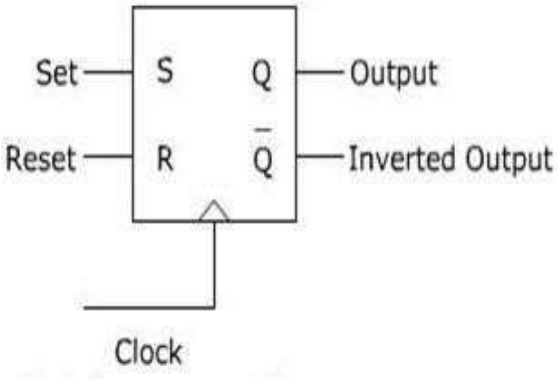


Truth Table: (2m)

Inputs		Outputs		
B	A	A > B	A = B	A < B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

07. When 2 i/p of an SR flip-flop is set to high, the o/p is invalid, explain how this problem is resolved.

Logic Symbol and truth table for S R flip flop



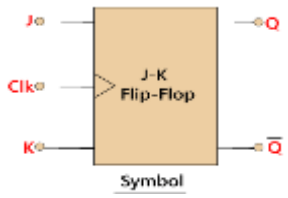
INPUTS			OUTPUT	STATE
CLK	S	R	Q	
X	0	0	No Change	Previous
↑	0	1	0	Reset
↑	1	0	1	Set
↑	1	1	-	Forbidden

- When the clock is not applied and S = 0, R = 0, flip flop gets disabled, so output remains in its previous state.
- When the clock is applied, S = 0, R = 1, then output goes to a Reset state. So Q = 0.

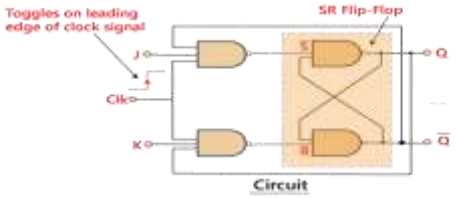
- When the clock is applied, S = 1, R = 0, then output Q = 1 that means flip flop is in Set state.
- **When the clock is applied, S = 1, R = 1, then output goes to a forbidden state / invalid state. This invalid state problem can be solved using JK flip flop.**

JK Flip flop:

- The JK flip flop works in the same way as the SR flip flop work.
- The JK flip flop has 'J' and 'K' inputs instead of 'S' and 'R'.
- The only difference between JK flip flop and SR flip flop is that when both inputs of SR flip flop is set to 1, the circuit produces the invalid states as outputs, but in case of JK flip flop, there are no invalid states even if both 'J' and 'K' flip flops are set to 1.



Symbol



Circuit

Logic symbol and Circuit Diagram of JK flip flop

Truth table of JK flip flop

Clk	J	K	Q	Q'	State
1	0	0	Q	Q'	No change in state
1	0	1	0	1	Resets Q to 0
1	1	0	1	0	Sets Q to 1
1	1	1	-	-	Toggles

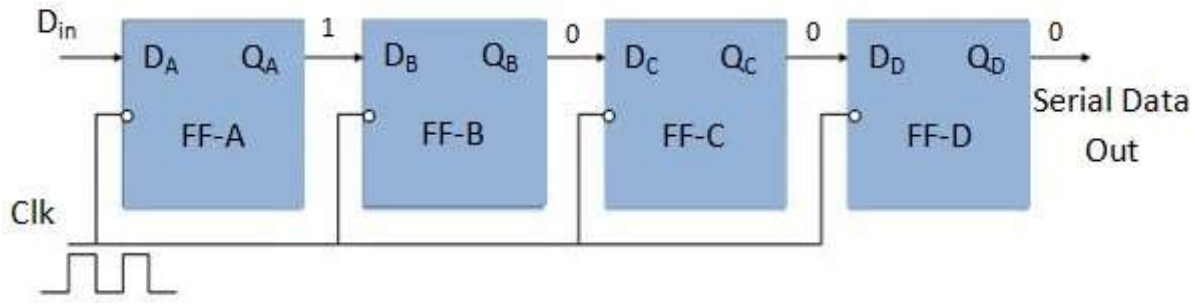
- When J and K inputs are both HIGH and logic "1", then the Q output will change state (Toggle) for as long as the clock input, (CLK) is HIGH.

08. Construct 4-bit SISO shift register

- Serial-in-serial-out (SISO) shift register accepts data serially i,e one bit at a time on a single input line.
- It produces the stored information on its single output line in serial form.
- Data may be shifted left using shift-left register or shifted right by using shift- right register

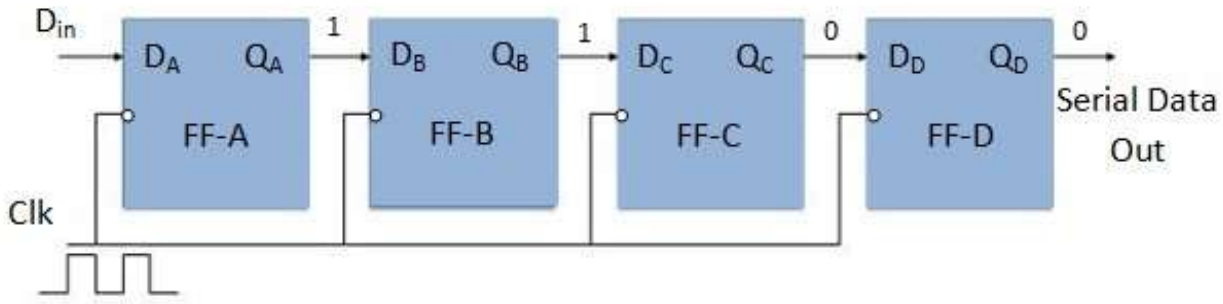
Operation (If in the below sequence, any one occurrence with correct representation is written, full marks for logic diagram can be awarded) Assume a data word 1111 is to be stored. Before application of clock signal, let Q_A

Q_B Q_C Q_D = 0000 and apply LSB bit of the number to be entered to D_{in}. So, D_{in} = D_A= 1. Apply the clock. On the first falling edge of clock, the FF-A is set, and stored word in the register is Q_A Q_B $Q_C$$Q_D$= 1000

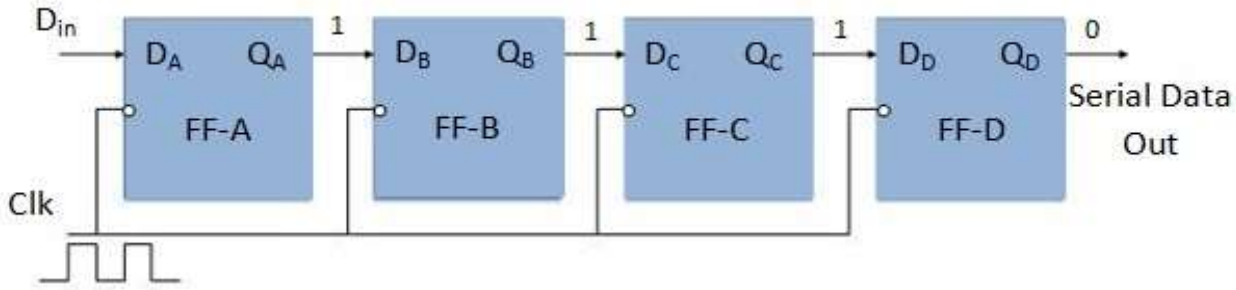


.

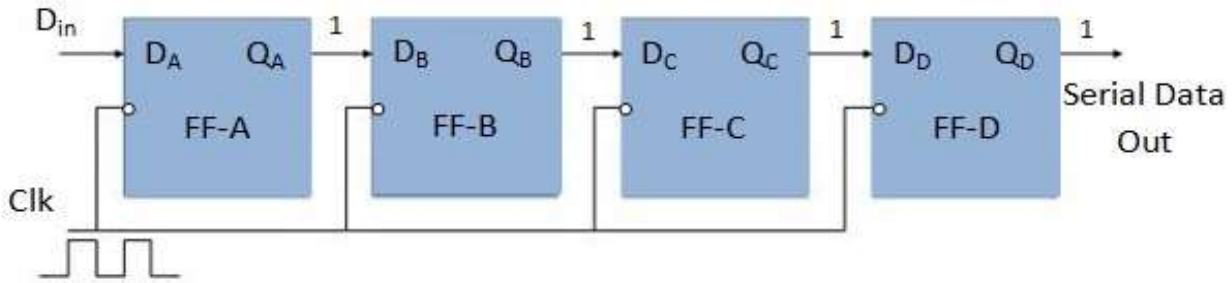
Apply the next bit to D_{in}. So D_{in} = 1. As soon as the next negative edge of the clock hits, FF-B will set and the stored word change to Q_A Q_B Q_C Q_D = 1100



Apply the next bit to be stored i.e. 1 to D_{in}. As soon as the third negative edge of the clock hits, FF-C will be set and output will be modified to$Q_A$$Q_B$ Q_C Q_D = 1110



Similarly with D_{in} = 1 and with the fourth negative clock edge arriving, the stored word in the register is Q_A Q_B Q_C Q_D = 1111.

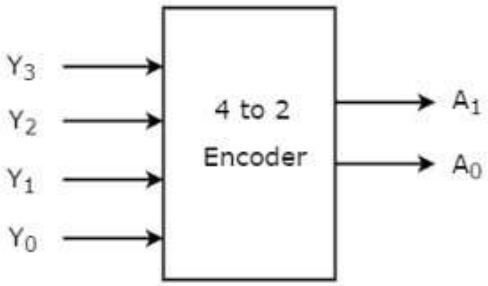


Truth table

	CLK	$D_{in}=Q_A$	$Q_A=D_B$	$Q_B=D_C$	$Q_C=D_D$	Q_D
Initially			0	0	0	0
(i)	↓	1	1	0	0	0
(ii)	↓	1	1	1	0	0
(iii)	↓	1	1	1	1	0
(iv)	↓	1	1	1	1	1

→ Direction of data travel

09. design 4:2 Encoder with truth table



Truth table: (3m)

Inputs				Outputs	
Y_3	Y_2	Y_1	Y_0	A_1	A_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

10. list different types of SHIFT REGISTER: 3marks

A group of flip flops which is used to store multiple bits of data and the data is moved from one flip flop to another is known as **Shift Register**. **(1m)**

Types of Shift registers: (2m) ○ Serial In Serial Out (SISO) ○ Serial In Parallel Out (SIPO) ○ Parallel In Serial Out (PISO) ○ Parallel In Parallel Out

11. list different types of FLIP-FLOPS 3marks

A flip flop is an electronic circuit used to store binary data. The stored data can be changed by applying varying inputs. (1m) Types of flip flops: (2m)

i. RS Flip Flop ii. JK Flip Flop iii. D Flip Flop iv. T Flip Flop

12.deffrence between combination circuit and sequential circuit 4or 5marks

Difference between Combinational and sequential circuits

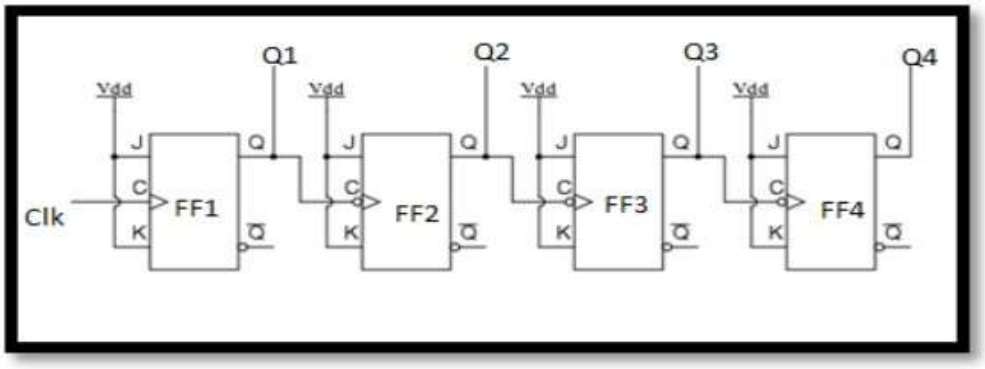
Combinational	Sequential
Output of any instance of time depends only upon the input variables	Output is generated dependent upon the present input variables and also on the basis of past history of these inputs
Memory unit is not required. i.e. it doesn't allocate any memory to the elements.	Memory unit is required. . i.e. it allocates any memory to the elements.
Faster	Slower
Easy to design	Difficult
Parallel adder	Serial adder
Ex- Half and full adder Half and full subtractor MUX , DEMUX	Ex- Flip flops Shift registers Binary counters

13. Construct 4-bit asynchronous counter with logic circuit and truth table

4 bit asynchronous (ripple or serial) counter 10 marks

- The term Asynchronous refers to events that do not have a fixed time relationship with each other.
- It is the one in which the flip flops within the counter do not change the states at exactly the same time because they don't have common clock pulse.
- Asynchronous counters are also called *ripple-counters* because of the way the clock pulse ripples it way through the flip-flops.
- The below figure shows a 4-bit binary ripple counter constructed using JK flip flops.
- The output of FF1 drives FF2, the output of FF2 drives FF3 and the output of FF3 drives FF4. All the J&K inputs are connected to V_{dd}, where each flip-flop toggles on the negative edge of its clock input.

- Consider initially all Flip flops to be in logical 0 state (Q1=Q2=Q3=Q4=0). A negative transition in clock input drives FF1 and causes Q1 to change from 0 to 1.
- FF2 does not change its state since it also requires negative transition at its clock input. With the arrival of the second clock pulse to FF1 Q1 goes from 1 to 0. This change of state creates the negative going edge needed to trigger FF2, and thus Q2 goes from 0 to 1.
- Thus before the arrival of the 16th clock pulse all the FF are in the logical 1 state. Clock pulse 16 causes Q1 Q2 Q3 Q4 to go logic 0 state.



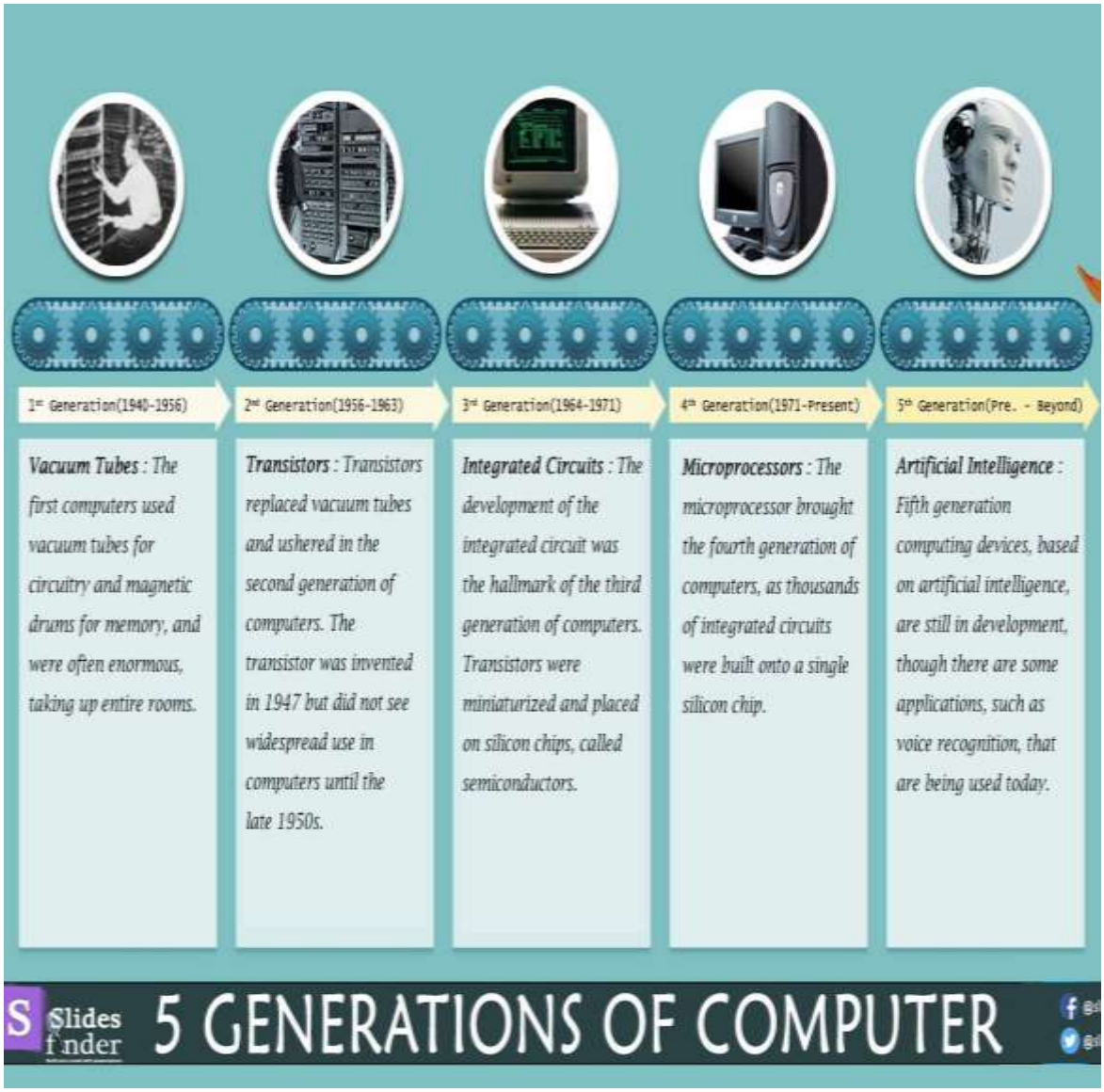
Truth table of 4 bit binary Ripple counter

State	Q4	Q3	Q2	Q1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
0	0	0	0	0

Chapter -03

Introduction to computer concepts

01. explain generations of computer 8 or 10 marks



02.clssify computer according to purpose 5 or 7marks

) Classify computers according to purpose.

According to purpose, computers are either general purpose and specific purpose.

General purpose computers are designed to perform a range of tasks. They have the ability to store numerous programs, but lack in speed and efficiency.

General Purpose Computer have the capability of dealing with variety of different problems, and are able to act in response to programs created to meet different needs.

A general-purpose computer is one that has the ability to store different programs of instruction and thus to perform a variety of operations.

Specific purpose computers are designed to handle a specific problem or to perform a specific task. A set of instructions is built into the machine.

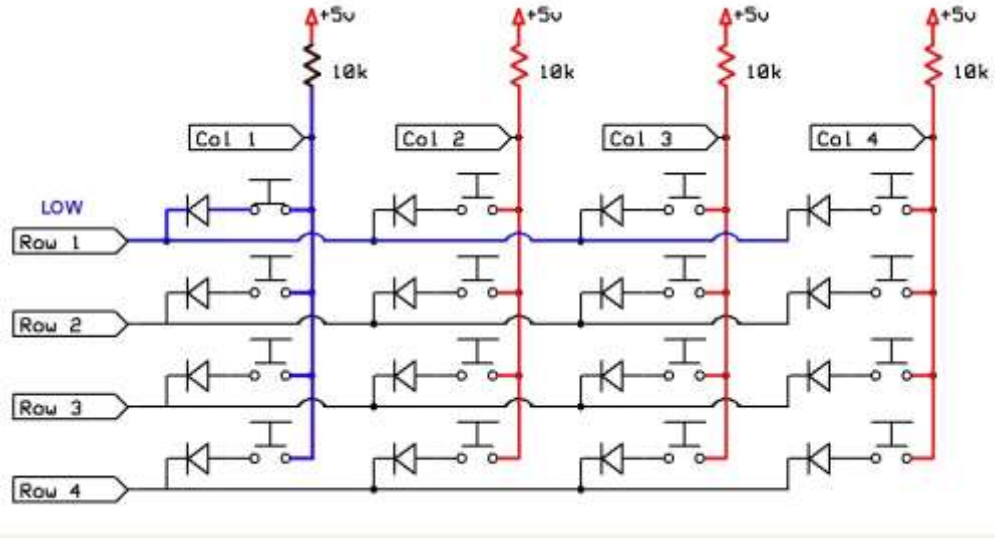
Special Purpose Computer is designed to perform one specific task. The program of instructions is built into, or permanently stored in the machine. Specialization results in the given task being performed very quickly and efficiently.

Most special purpose computers have the capability of performing just one task. They are frequently referred to us "dedicated," because of their limitations to the specific task at hand.

03

04.Explain the working of a keyboard with neat diagram 6 or 8marks

) A keyboard has own processor and circuitry that carries information to and from that processor. A large part of this circuitry makes up the key matrix.A keyboard contains many push-buttons called "keys". When one of these are pushed, an electrical circuit is closed, and the keyboard sends a signal to the computer that tells it what letter, number or symbol it would like to be shown on the screen. **(2m)**



05.list applications of computer 4 or 5 marks

Applications of computer: (any 4)**(4*1=4m)**

- Business
- Education
- Marketing
- Banking
- Insurance
- Communication

- Health Care
- Military
- Engineering Design

06.define computer network explain computer categories 10 marks

COMPUTER NETWORK

- A computer network is an interconnection of two or more devices (computers) that are able to exchange information.

CATEGORIES/TYPES OF COMPUTER NETWORK

A computer network can be categorized by their size. A computer network is mainly of **three types:**

1. LAN(Local Area Network)
2. MAN(Metropolitan Area Network)
3. WAN(Wide Area Network)

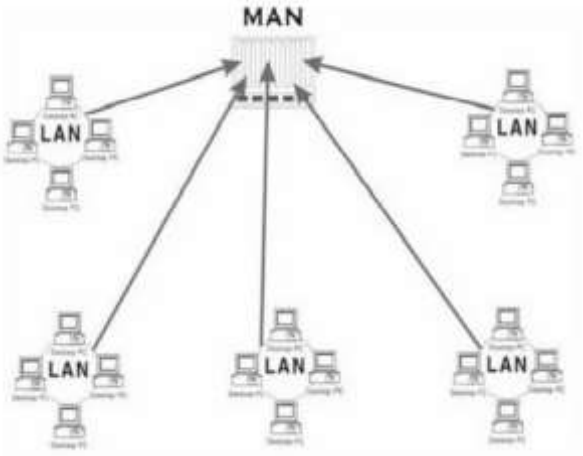
1. Local Area Network (LAN):

- A Local Area Network (LAN) is a group of computer and peripheral devices which are connected in a limited area such as school, laboratory, home, and office building or a campus.

- Network size is limited to a small geographical area, presently to a few kilometers.
- A local area network (LAN) is privately owned.
- Data transfer rate is generally high. They range from 100 Mbps to 1000 Mbps.
- The number of computers connected to a LAN is usually restricted. In other words, LANs are limitedly scalable. • IEEE 802.3 or Ethernet is the most common LAN. Ethernet's speed has a range from 2.9 Mbps to 400 Gbps.
- LAN Topologies: LAN topologies define the manner in which network devices are organized. The following are the topologies.
  1. Bus Topology
  2. Ring Topology
  3. Tree Topology
  4. Star Topology
  5. Mesh Topology
  6. Hybrid Topology
-

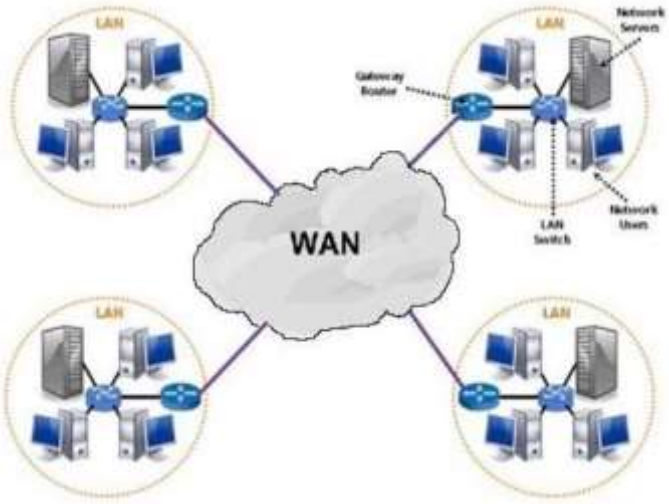
2. METROPOLITAN AREA NETWORK (MAN)

- A metropolitan area network (MAN) is a network with a size between a LAN and a WAN. It normally covers the area inside a town or a city.
- In MAN, various LANs are connected to each other through a telephone exchange line.
- The most widely used protocols in MAN are RS-232, Frame Relay, ATM, ISDN, OC-3, ADSL, etc.
- It has a higher range than Local Area Network.



3. Wide Area Network (WAN)

- A **wide area network (WAN)** is an interconnection of devices over a large geographical area such as a town, a state, a country, or even the world.
- WANs are often used by larger corporations or organizations to facilitate the exchange of data.
- A Wide Area Network is not limited to a single location, but it spans over a large geographical area through a telephone line, fibre optic cable or satellite links.
- The internet is one of the biggest WAN in the world.



07.lan vs man vs wan 5marks

Differences between LAN, MAN and WAN

Basis of Comparison	LAN	MAN	WAN
Full-Form	Local Area Network.	Metropolitan Area Network.	Wide Area Network.
Geographic Span	Operates in small areas such as the same building or campus.	Operates in large areas such as a city.	Operates in larger areas such as country or continent.
Ownership	Private.	Private or Public.	Private or Public
Transmission Speed	High.	Average.	Low.
Propagation delay	Short	Moderate	Long
Congestion	Less Congestion	More Congestion	More Congestion than MAN.
Design & Maintenance	Easy.	Difficult	Difficult
Fault tolerance	More fault tolerant	Less fault tolerant	Less fault tolerant
Bandwidth	High bandwidth	Less bandwidth	Low bandwidth
Examples	College, School, University Hospital	City Building	Internet throughout the country or continent.

08.What are the differences between System Software and Application Software 4 marks

Ans:

System Software	Application Software
• System software are mainly designed for managing system resources.	• Application software are designed to accomplish tasks for specific purposes.
• Programming of system software is complex.	• Programming of application software is comparatively easy.
• A computer cannot run without system software.	• A computer can easily run without application software.
• System software do not depend on application software.	• Application software depend on system software and cannot run without system software.

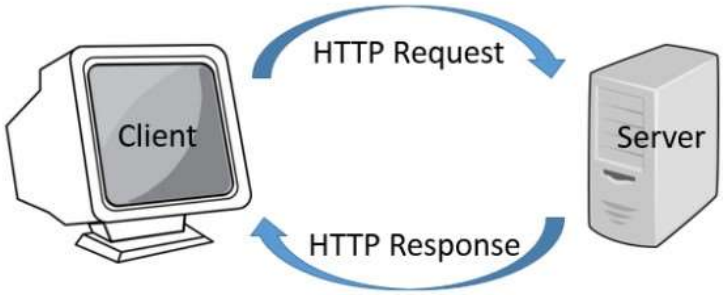
09.explain application layer protocols 7marks

It allows the user to use the services of the network.

1. It provides user services like user login, naming network devices, formatting messages, and emails, transfer of files etc.

2. The commonly used protocols are HTTP, FTP, SMTP, POP/IMAP, DNS, DHCP, TELNET, MIME .

1. **Hyper Text Transfer Protocol (HTTP):** Used to access data i.e., text, audio, video and images on the World Wide Web (WWW).



2. **File Transfer Protocol (FTP):** Used to transfer files from one computer to another on a network.

3. **Simple Mail Transfer Protocol (SMTP):** SMTP is used to transfer email messages from sender's mail server to receiver's mail server.

4. **Post Office Protocol (POP) or Internet Message Access Protocol (IMAP):** POP and IMAP provides end users the ability to fetch and receive email from a remote mail server.

5. **Domain Name System (DNS):** It translates the domain name (such as amazon.in, google.com) into the corresponding IP address. For example, the domain name www.abc.com might translate to 198.105.232.4

6. **TELNET (Terminal Network):** TELNET allows a user to log onto a remote machine and access any application program on a remote computer.

7. **Multipurpose Internet Mail Extension (MIME):** It is an extension of SMTP that allows the transfer of multimedia messages.

8. **Dynamic Host Configuration Protocol (DHCP):** DHCP is a client/server protocol that automatically provides a host (i.e., computer or mobile) with its IP address and other related configuration information such as the subnet mask and default gateway.

10.explain cyber secuirities threats 5 marks

Sources of Threat

The possible sources of a computer threat may be −

- **Internal** −It includes employees, partners, contractors (and vendors).
- **External** −It includes cyber-criminals (professional hackers), spies, nonprofessional hackers, activists, malware (virus/worm/etc.), etc.

Common Terms

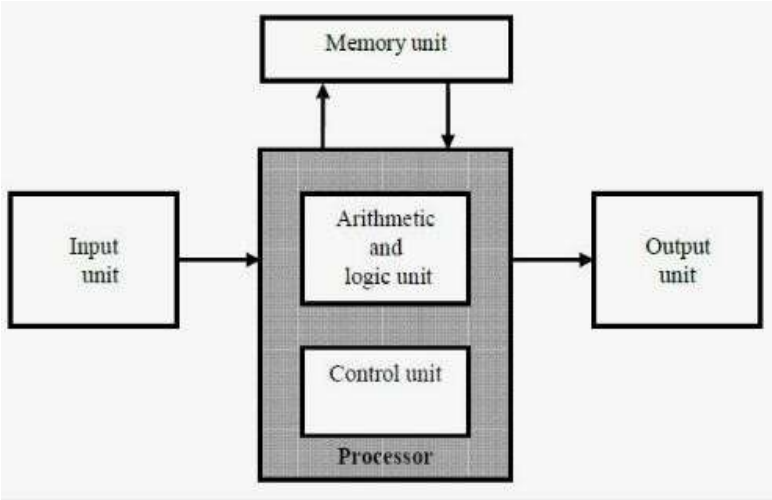
Following are the common terms frequently used to define computer threat −

- **Virus Threats**
  A computer virus is a program designed to disrupt the normal functioning of the computer without the permission of the user.
- **Spyware Threats**
  Spyware is a computer program that monitors user's online activities or installs programs without user's consent for profit or theft of personal information.
- **Hackers**
  Hackers are programmers who put others on threats for their personal gain by breaking into computer systems with the purpose to steal, change or destroy information.

Chapter -04

01. Explain functional units of computer with neat diagram 6 or 10 marks

Diagram of Functional units of a Computer:



Explanation:

Various Functional units of computer are:

i.**Input Unit:** Data or Instructions to a computer is input using any Input device. Commonly used input devices are Keyboard, Mouse, Joystick, scanner etc.

ii.**Central Processing Unit(CPU):** CPU- also known as brain of computer, is constituted of Arithmetic and Logic Unit(ALU) and Control Unit(CU).

ALU is responsible for performing all the Arithmetic like addition, subtraction etc and Logical operations like AND, OR etc. The intermediate results are stored on temporary storage called the Registers.

CU controls and coordinates all the processes being executed on the computer. It generates the timing signal responsible to which all the processes are synchronized.

iii.**Memory Unit**: Memory unit store the programs and the data on which the instructions of programs operate. Instructions are fetch from memory, executed on CPU and result is stored back to memory.

iv.**Output Unit**: This unit renders the output to the users. Commonly used output devices are Monitor, Printer, Speaker etc.

02.Flynn's classification of computers 6marks

Flynn's classification of computers: (**Listing or Table) - (2m)**

	Single Data	Multiple Data
Single Instruction	SISD	SIMD
Multiple Instruction	MISD	MIMD

Features: any two each **(4*1=4m)**

i. **SISD** (Single Instruction Single Data): Instructions are executed sequentially, Von Neumann Architecture based, Single CPU computer,

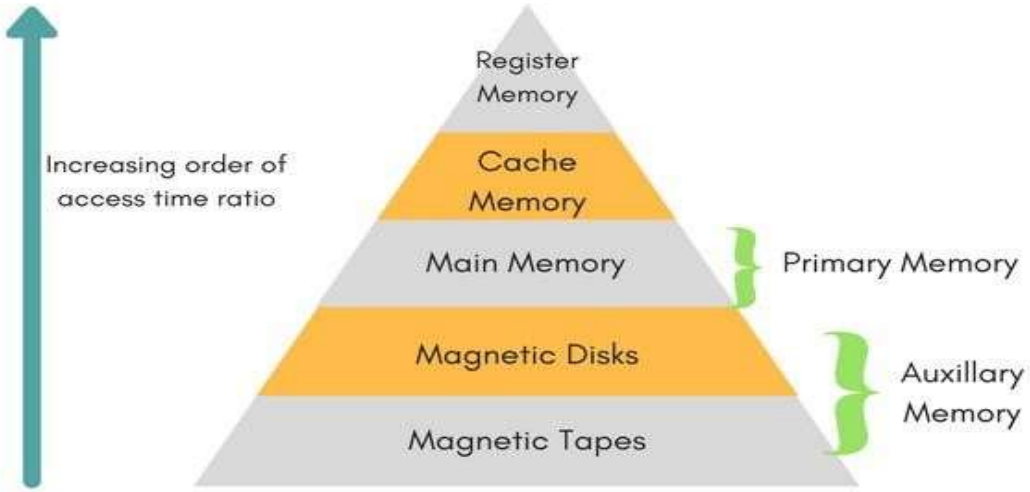
ii. **SIMD** (Single Instruction Multiple Data): includes many processing units under the supervision of a common control unit, Vectors processors, Parallel Processing

iii. **MISD** (Multiple Instruction Single Data): multiple processing units operate on one single-data stream,May be pipelined, Theoretical model only (not practically used)

iv. **MIMD** (Multiple Instruction Multiple Data): Multiple processors in a parallel computer execute different instructions and operate on various data at the same time, Multiprocessing, Ex: Cray T90, Cray T3E, IBMSP2

03.Draw the memory hierarchy and label it 5marks

Computer memory can be said to be organized in a hierarchical structure, where memory with the fastest access speeds and highest costs lies at the top whereas those with lowest speeds and hence lowest costs lie at the bottom.



Applications

Registers: Used in CPU to store intermediate result

Cache memory: used to increase speed of access

Main memory: It is the addressable memory by the CPU. All the instructions and data are stored here before loaded toc cpu.

Disk cache: Part of hard disk, used as cache memory to increase disk access speed

Magnetic disk / tape: Large storage devices, used for permanent/persistent storage of data

04. differentiate between BIOS AND UEFT 5MARKS

BIOS	UEFI
BIOS refers to Basic Input/Output system	UEFI refers to Unified Extensible Firmware Interface
It works on 16bit mode	Supports 32bit and 64bit mode
Uses MBR (Master boot record) to save information about hard drive	Uses GPT (GUID Partition Table)) to save information about hard drive
BIOS doesn't support GUI	UEFI supports GUI
Hard disk drive is limited to maximum size of 2TB	Hard disk Drive may of size larger than 2TB
Require more boot time than UEFI	Booting takes less time than in BIOS
Less secure	More secure than BIOS

05. differentiate between mobail os AND computer os 5MARKS

Any 5 differences- (5*1=5m)

Mobile OS	Computer OS
Mobile OS helps and supports to run application software on mobile devices	Computer OS allows user to run their applications on Computer
Mobile OS is simple and lightweight compared to Computer OS (small in size)	Computer OS is complex and heavy weight compared to Mobile OS (large in size)
Supports less number of functionalities/ services	Supports large number of functionalities/ services
Boot time is lesser than Computer OS	Boot time is more than Mobile OS
Applications are designed to consume less	No such restrictions imposed on the applications
memory and space	
Ex: Android, Bada, Blackberry OS, iOS etc	Ex: Windows 10, Mac, Redhatetc

06.list and explain the services of OS(operating system) 10marks

(b) List and explain the services of OS (Operating system).

(c) List and explain the services of OS (Operating system).

(List -2m & Explanation of any 4 services- 8m)

A typical operating system contains number of management routines and they could vary from one OS make to another. Thus the operating system services can be summarized as follows: (OS management activities summary).

- **Program Execution**: (The runtime environment)
  - Accomplish the task of loading a program into main memory partitions.
  - And initiate program execution.
  - Provide for normal termination of program after successful execution.

- Provide for displaying error messages in case of abnormal termination.

- **I/O Operations:**
  - Accomplish the task of device allocation and control I/O devices.
  - Provide for notifying device errors, device status etc.

File system Manipulation :(File Handling).

- Accomplish the task of opening a file, closing a File
- Provide for creating a file, deleting a file
- Allow file manipulation such as reading a File, writing a File,

  Appending a

  File etc. **Communications:**
- Accomplish the task of inter-process communication either on the same computer system or between different computer systems on a computer networks.
- Provide for message passing and shared memory access in safe mode.

- **Accounting:**
  - Accomplish the task of record keeping the system usage (the resources being used) by how long (the duration) for the billing & accounting purposes.
  - Maintain logs of system activities for performance analysis and error recovery.
  - Keep track of history of user logins.

- **Error Detection:**
  - Accomplish the task of error detection and recovery if any. For instance Paper Jam or Out of Paper in a Printer.
  - Keep track status of CPU, Memory, I/O devices, Storage devices, File system networking etc.
  - About execution in case of fatal errors such as RAM parity errors, power fluctuations, if any.
  - Report and /or deliver error messages in case of arithmetic overflow, divide by zero errors, etc.

- **Resource Allocations:**
  - Accomplish the task of resource allocations to multiple jobs.
  - Reclaim the allocated resources after their use or as and when the job terminates.
  - Provide for resource allocation and scheduling policies (such as round robin, shortest job first, polling & handshaking, CPU bound, I/O bound) for prioritizing jobs for resources allocation and detaining them for execution.

Protecting the system:

- Accomplish the task of protecting the system resources against malicious use.
- Provide for safe computing by employing security scheme against unauthorized access/users.

Authenticate legitimate users with login passwords and registrations

Chapter -05 introduction to computer programming

01.Discuss the characteristics of algorithm 5marks

02.explain generations of programming languages 10marks

O3.explain different types of constants 4marks

04.important algorithms 5 or 10marks

05.important flowcharts 5 or 10 marks

06.rules for naming variables 5marks

Answers

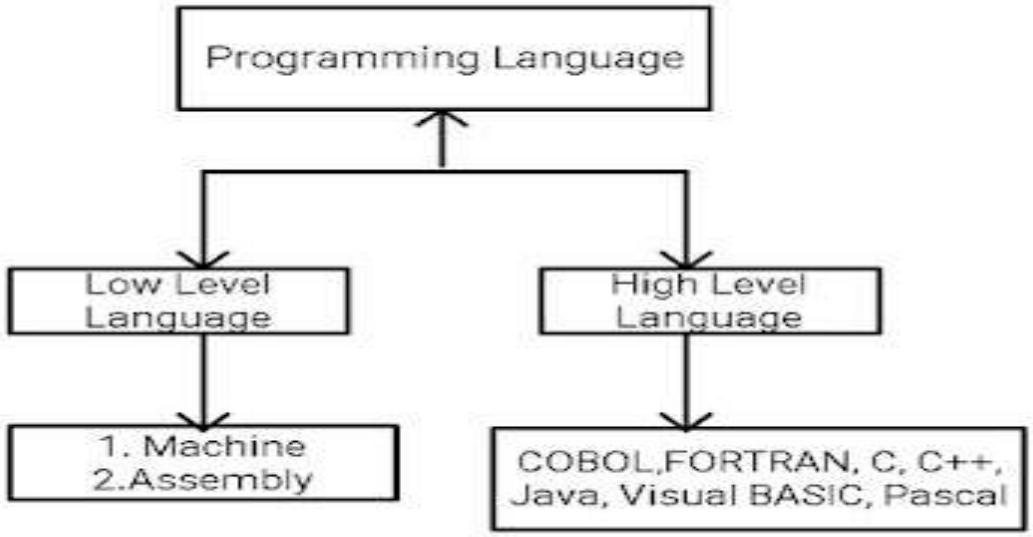
01.Discuss the characteristics of algorithm 5marks

Discuss the characteristics of algorithms.

ALGORITHM

- Algorithms are one of the most basic tools that are used to develop problemsolving logic.
- An algorithm is defined as a finite sequence of explicit instructions that when provided with a set of input values produces an output and then terminates.
- To be an algorithm, the steps must be unambiguous and after a finite number of steps, the solution of the problem should be achieved.
- However, algorithms can have steps that repeat (iterate) or require decisions (logic and comparison) until the task is completed.
- The need of an algorithm can be understood as it provides a logical structure to plan the solution.

02.explain different types of generations of programming languages 10marks



(1)Low Level Programming Languages

Programming languages that can only understand the computer. Low Level LANGUAGE is called Low Level Language Computer. This is the exchange that computers do not directly understand. That is why it does not need to translate the instructions written in. Therefore, the program written in it is a fast task. But it is very difficult to develop the program because it is very important to know the computer level of the computer. The Low Level Language is the following two types

(1) Machine

Machine language is the language of computer. The computer only understands the instructions written in machine language and can execute. Instructions written in machine language are 0,1 in the form of two digits, which is why it is also called binary language. Programs created in machine language are not portable. That is, the computer built into them can be run only in the same computer and not in any other computer.

(2) Assembly

Due to the instructions points written in the machine, it only makes computers. It is only aware of the writing. It has a very difficult task to write in it. That's why a new language was created to write instructions, which is called Assembli language. In it, the instructions are written as words of English, which is called Mnemonics. The later computer is translated into machine language with the help of the assembler, the computer understands it. The program created in Assembli language is not fully portable. They need to be transported to run on another computer.

High level language

- These are high level languages similar to English language.
- High level languages are human readable and easy to understand.
- A high-level language is designed to simplify computer programming.
- High-level source code contains easy-to-read syntax
- But machines cannot understand high level language.

C++

C ++ language development BJARNE STOURTRUP was in the 1980s at the United States of America. In this year, the development of this language was adding a lot of new features in the language. Therefore, the name of C ++ is kept using the C ++. Mainly used to make software software, but all kinds of software can be created. This is the main difference in C and C ++ that supports the procedure oriented programming (POP) technology to make the program while C ++ object supports the Oriented Programming (OOP) technology.

Java

Java language development was made in Sun Microsystem in 1990. It is also an object oriented programming language that is used primarily to create applications related to website. Application created in all devices like-desktop, laptop, tablets, go on smartphone. Therefore it is also called Platform Independent Language.

O3.explain different types of constants 4marks

Explain different types of Constants.

Constants refer to fixed values that the program may not alter during its execution. These fixed values are also called **literals** .Constants can be of any of the basic data types like an integer constant, a floating constant, a character constant, or a string literal.

Integer Literals

An integer literal can be a decimal, octal, or hexadecimal constant. A prefix specifies the base or radix: 0x or 0X for hexadecimal, 0 for octal, and nothing for decimal.

An integer literal can also have a suffix that is a combination of U and L, for unsigned and long, respectively. The suffix can be uppercase or lowercase and can be in any order.

Examples:

212 /* Legal */

215u /* Legal */

0xFeeL /* Legal */

Floating-point Literals

A floating-point literal has an integer part, a decimal point, a fractional part, and an exponent part. You can represent floating point literals either in decimal form or exponential form.

While representing decimal form, you must include the decimal point, the exponent, or both; and while representing exponential form, you must include the integer part, the fractional part, or both. The signed exponent is introduced by e or E.

Examples:

3.14159 /* Legal */

314159E-5L /* Legal */

Character Constants

Character literals are enclosed in single quotes, e.g., 'x' can be stored in a simple variable of **char** type.

A character literal can be a plain character (e.g., 'x'), an escape sequence (e.g., '\t'), or a universal character (e.g., '\u02C0').

There are certain characters in C that represent special meaning when preceded by a backslash for example, newline (\n) or tab (\t).

String Literals

String literals or constants are enclosed in double quotes " ". A string contains characters that are similar to character literals: plain characters, escape sequences, and universal characters.

We can break a long line into multiple lines using string literals and separating them using white spaces.

Examples:

"hello, dear" "hello, \ dear"

"hello, " "d" "ear"

04.important algorithms 5 or 10marks

1. Algorithm for adding two numbers

Step 1: Start

Step 2: Declare variables num1, num2 and sum.

Step 3: Read values for num1, num2.

Step 4: Add num1 and num2 and assign the result to a variable sum.

Step 5: Display sum

Step 6: Stop

Computer Science and 20CS11T 2020-21

Fundamentals of Computer 3

2. Algorithm for subtracting two numbers

Step 1: Start

Step 2: Read two numbers A and B

Step 3: Answer = A - B

Step 4: Display Answer

Step 5: Stop

3. Algorithm: To determine largest of three numbers

1. Start

2. Read three numbers A, B, C

3. Find the larger number between A and B and store it in MAX_AB

4. Find the larger number between MAX_AB and C and store it in MAX

5. Display MAX=4

6. Stop

4. Algorithm: To determine smallest of three numbers

1. Start

2. Read three numbers A, B,C

3. Find the smaller number between A and B and store it in MIN_AB

4. Find the smaller number between MIN _AB and C and store it in MIN

5. Display MIN

6. Stop

Computer Science and 20CS11T 2020-21

Fundamentals of Computer 4

5. Algorithm : Calculate and print sum of 'N' numbers

Step 1 : Start

Step 2 : Read how many numbers are to be considered for sum i.e addition &

store it in variable N

Step3 : Initialize the variable sum=0 & count=0

Step 4: Read the first number & store it in A

Step 5: Compute sum=sum+A

Step 6 :Compute count=count+1

Step 7: Check whether the value of COUNT is less than n i.e “count<N”, if

yes repeat the steps 4,5,6 other go to next step i,e 8

Step 8 : Print sum

Step 9 : Stop

The given number is odd or even

b) ALGORITHM :**(4m)**

STEP 1: START

STEP 2: Read an integer value from user, say N

STEP 3: IF N%2== 0 THEN

Print "N is an Even number"

ELSE

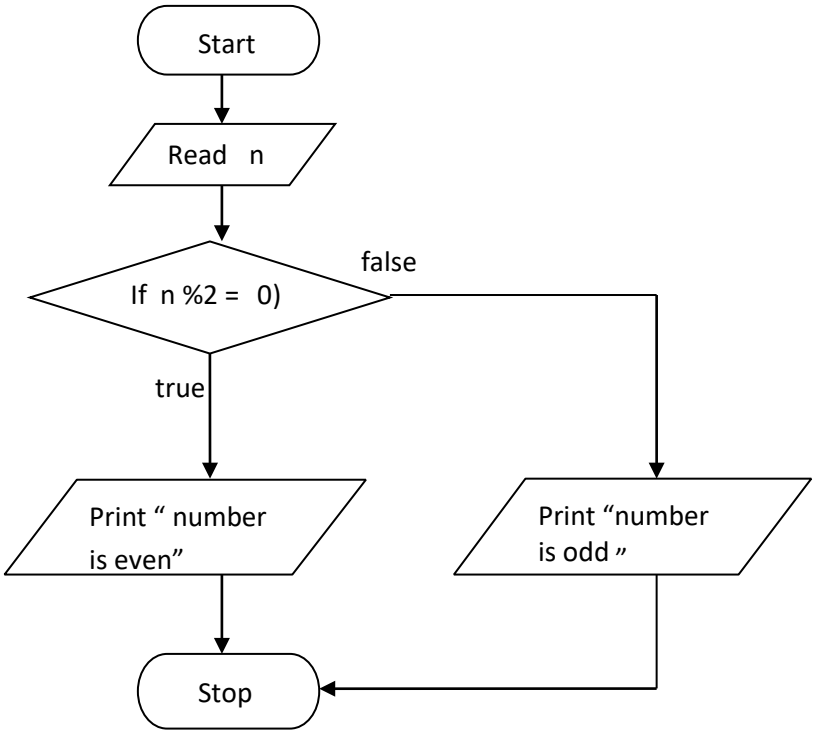
Print "N is an Odd Number"

ENDIF

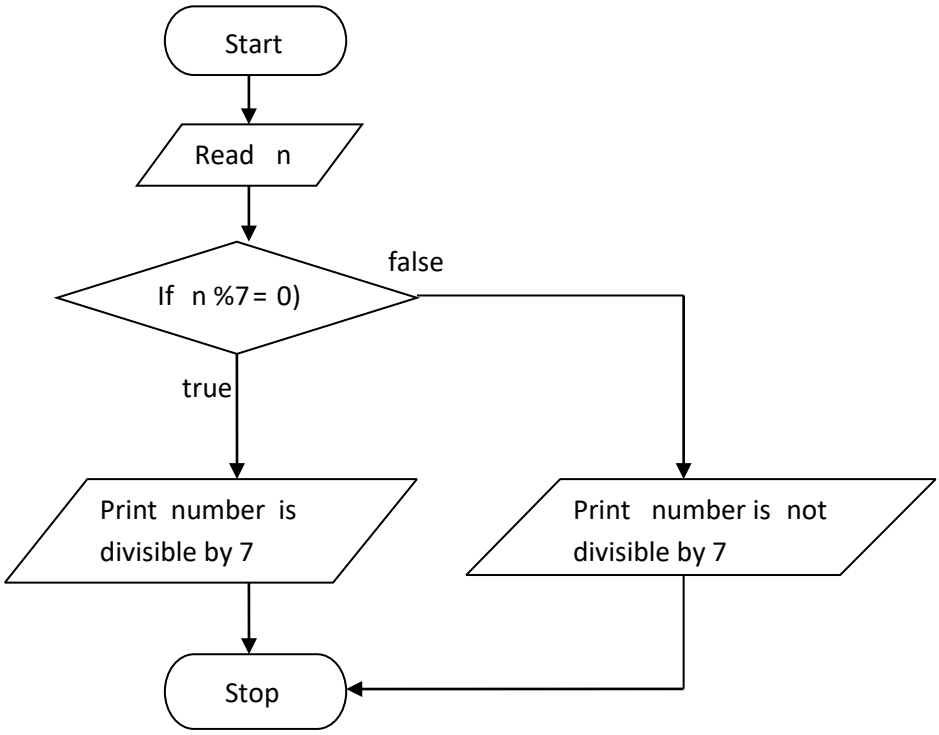
STEP 4: END

05.important flowcharts 5 or 10 marks

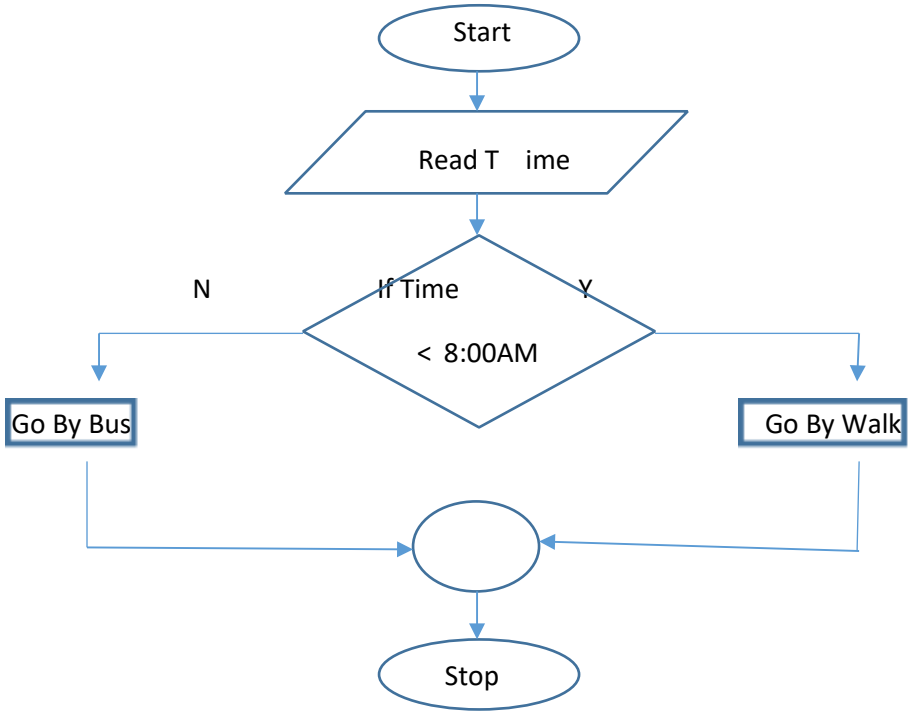
Flowchart to check whether the given number is odd or even.

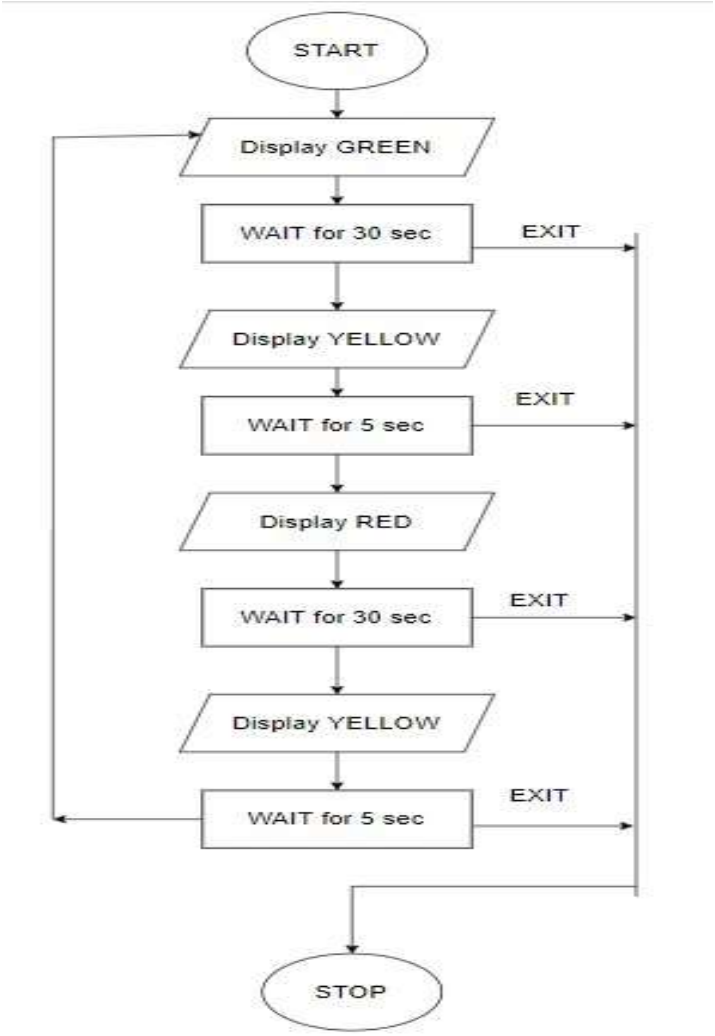


Flowchart to check whether the given number is divisible by 7 or not.



(b) Raj has to reach his college at 9:00am.If he leaves his house before 8:00am, he will go by walk, otherwise he takes a bus. Draw a flow chart for this given scenario, to output how to travel.





https://youtube.com/@Ndiplomainkannada

06.Rules for naming variable 5marks

● The name of a variable can be composed of letters, digits, and the underscore character.

● It must begin with either a letter or an underscore.

● Upper and lowercase letters are distinct.

● A variable name can hold a single type of value. For example, if variable a has been defined int type, then it can store only integer.

● You can use a variable name only once inside your program. For example, if a variable a has been defined to store an integer value, then you cannot define a again to store any other type of value.