CBCS SCHEME

USN | | | | | | | | | | |

21EE42

Fourth Semester B.E. Degree Examination, June/July 2023
Digital System Design

Time: 3 hrs. Max. Marks: 100

Note: ***Answer any FIVE full questions, choosing ONE full question from each module.***

Module-1

1 a. Design a circuit to cube a 2-bit number. Implement using minimum number of only NAND gates. **(07 Marks)**

b. Explain Maxterm canonical form with an example. Express the given Boolean function in proper canonical form with decimal notation.

$f(w,x,y,z) = (\bar{w}+x)(y+\bar{z})$. **(06 Marks)**

c. Convert the given Boolean expression into :
i) Minterm canonical form

ii) Maxterm canonical form $f(p,q,r) = P(\bar{q}+\bar{r})$. **(07 Marks)**

OR

2 a. Simplify the following function using Quine -Mc Cluskey technique.
$f(a, b, c, d) = \Sigma m(0, 4, 5, 9) + \Sigma dc(1, 7, 13)$.
Also obtain the minimal SOP form using Karnaugh map and verify the result. **(12 Marks)**

b. List all the prime implicants of the given function and obtain the minimal SOP form using Karnaugh Map.
$Y(a, b, c, d) = \Sigma m(0, 1, 2, 3, 4, 7, 8, 9) + \Sigma dc(10, 11, 12, 13, 14, 15)$. **(08 Marks)**

Module-2

3 a. Design a combinational logic circuit that will convert BCD digit to Excess-3 BCD digit using gates. Construct a truth table and simplify each output equation using Karnaugh maps. **(08 Marks)**

b. Implement the following function pairs using 74138IC and gates with minimum number of inputs.
$f_1(a, b, c) = \Sigma m(0, 2, 4)$ $f_2(a, b, c) = \Sigma m(1, 2, 4, 5, 7)$. **(06 Marks)**

c. Implement a 1-bit comparator using a decoder. **(06 Marks)**

OR

4 a. Implement the function : $f(a, b, c, d) = \Sigma m(1, 1, 5, 6, 7, 9, 10, 15)$ using a 4 : 1 Mulitplexer with a, b as select inputs. **(08 Marks)**

b. Implement a 4-bit carry look Ahead adder and explain how carry propagation delay is eliminated in a carry look ahead adder. **(12 Marks)**

Module-3

5 a. Explain how the switch bounce effect is eliminated by the use of an SR latch with the help of timing diagram. **(08 Marks)**

b. Explain the working of a Master-solve JK flip-flop with timing diagram. **(12 Marks)**

OR

6 a. Obtain the characteristic equations for T, D SR and JK flip-flop. **(10 Marks)**

b. Explain the working of a Master Slave SR flip-flop with timing diagram. **(10 Marks)**

Module-4

7 a. Explain the working of a universal shift Register with neat circuit diagram. **(09 Marks)**

b. Explain the operation of a 4-bit binary ripple counter using –ve edge triggered JK flip-flops giving the timing diagram. **(06 Marks)**

c. Draw the circuit diagram and timing diagram for MOD-12 ripple UP-counter using T flip-flops. **(05 Marks)**

OR

8 a. Design a MOD-8 twisted Ring counter using positive edge triggered D flip-flops and give the count sequence and timing diagram. **(08 Marks)**

b. Design a synchronous counter using positive edge triggered JK flip-flops for the count sequence 0, 1, 4, 6, 7, 5 ...... **(12 Marks)**

Module-5

9 a. Explain Mealy and Moore models in sequential circuits with block diagrams and examples. **(08 Marks)**

b. Design a synchronous circuit using positive edge triggered JK flip-flops to generate the flowing sequence :

$0 \rightarrow 1 \rightarrow 2 \rightarrow 0$ if Input X = 0 and

$0 \rightarrow 2 \rightarrow 1 \rightarrow 0$ if Input X = 1

Provide an output which becomes equal to '1' to indicate non-zero present state when X = 0. **(12 Marks)**

OR

10 a. Analyze the following sequential circuit and obtain :

i) Flip-flop input and output equations

ii) Transition table

iii) State table

iv) State diagram.

(Refer Fig.Q10(a)).

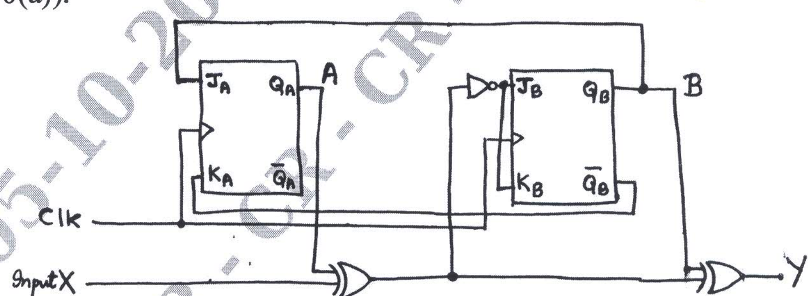


Fig.Q10(a) **(12 Marks)**

b. Write short notes on :

i) Read only memory

ii) Programmable ROM

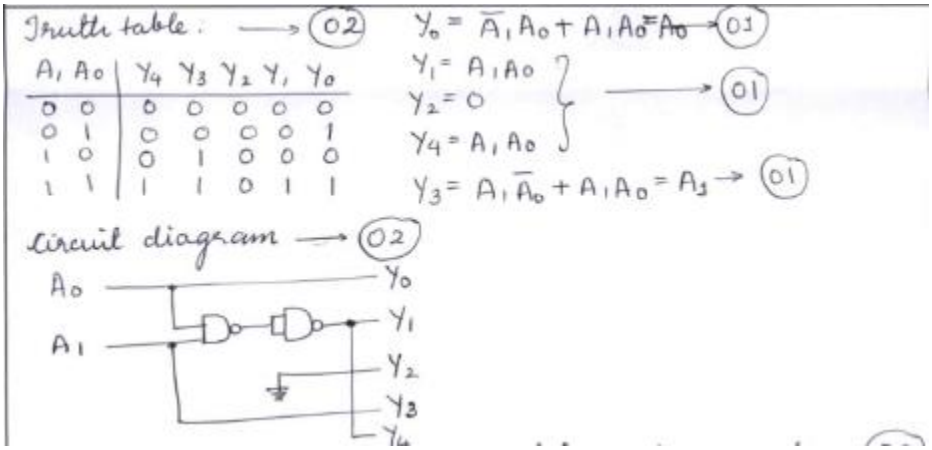
iii) EPROM

iv) Flash memory. **(08 Marks)**

* * * * *

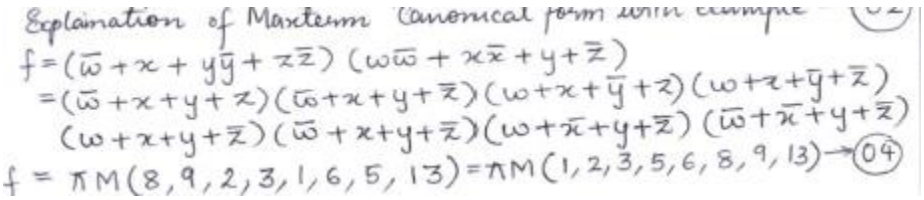
Module 1

1a.

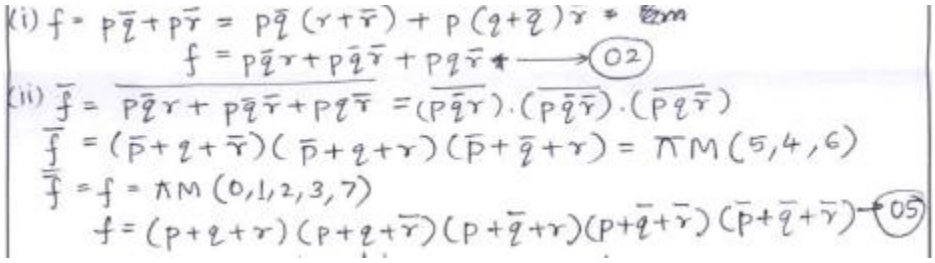


A_1 A_0	Y_4	Y_3	Y_2	Y_1	Y_0
0 0	0	0	0	0	0
0 1	0	0	0	0	1
1 0	0	1	0	0	0
1 1	1	1	0	1	1

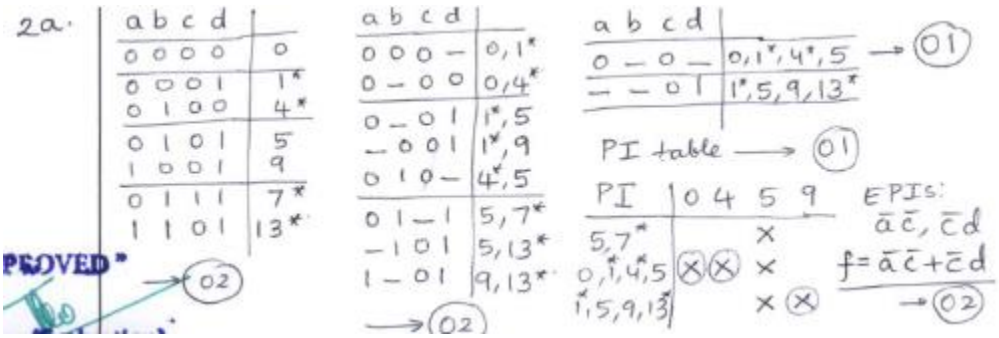
b.

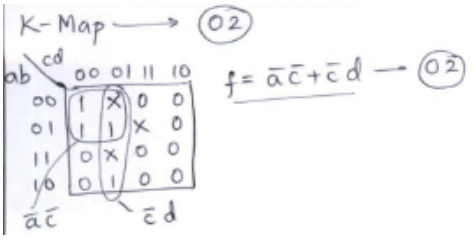


c.

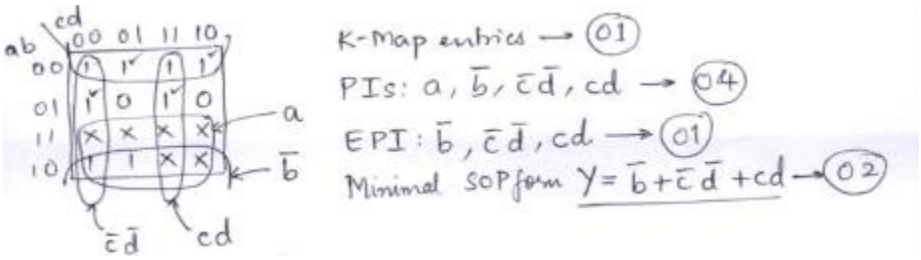


2a.



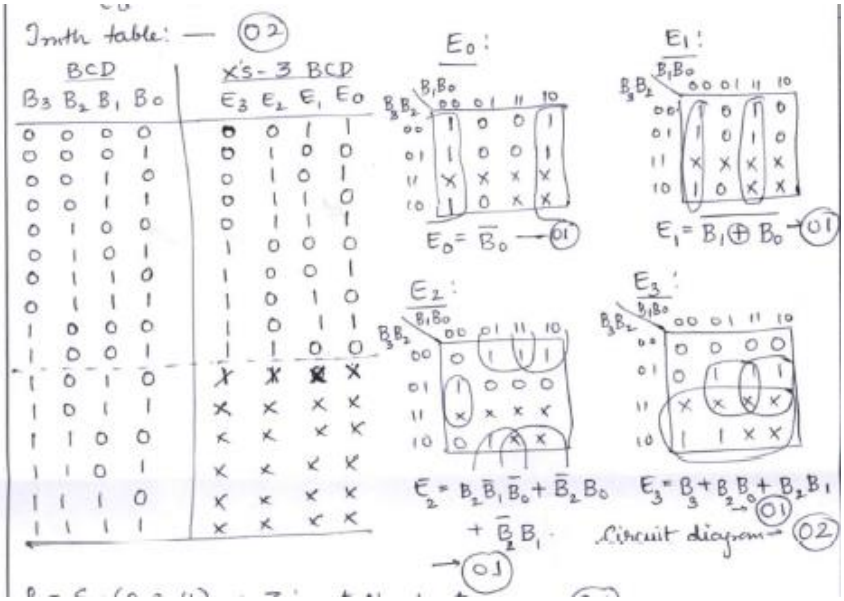


b.

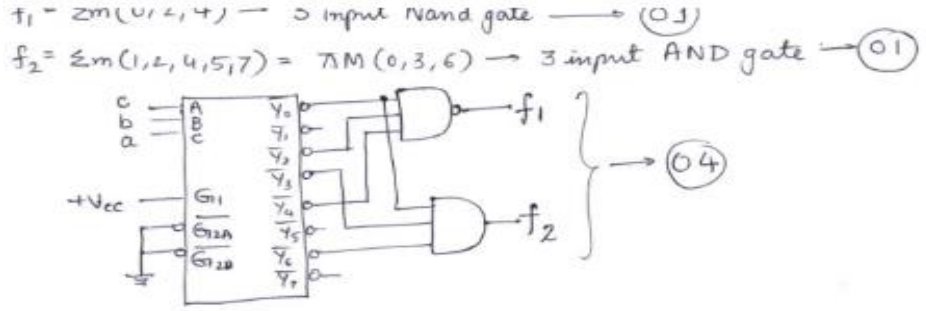


Module 2

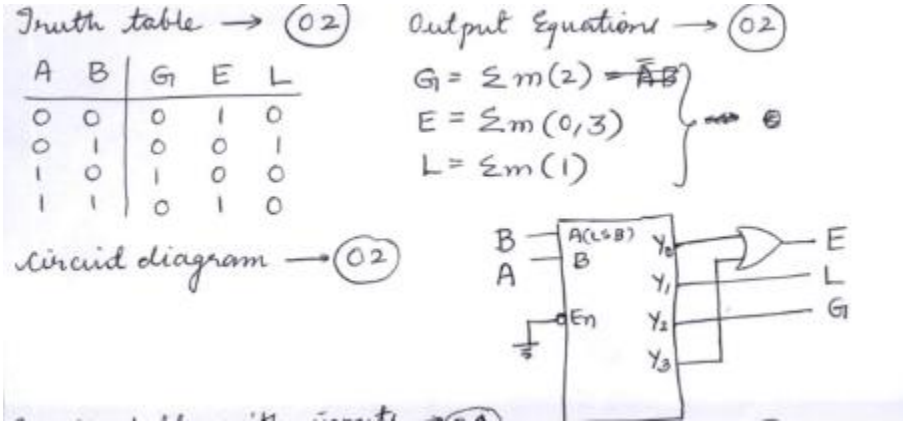
3a.



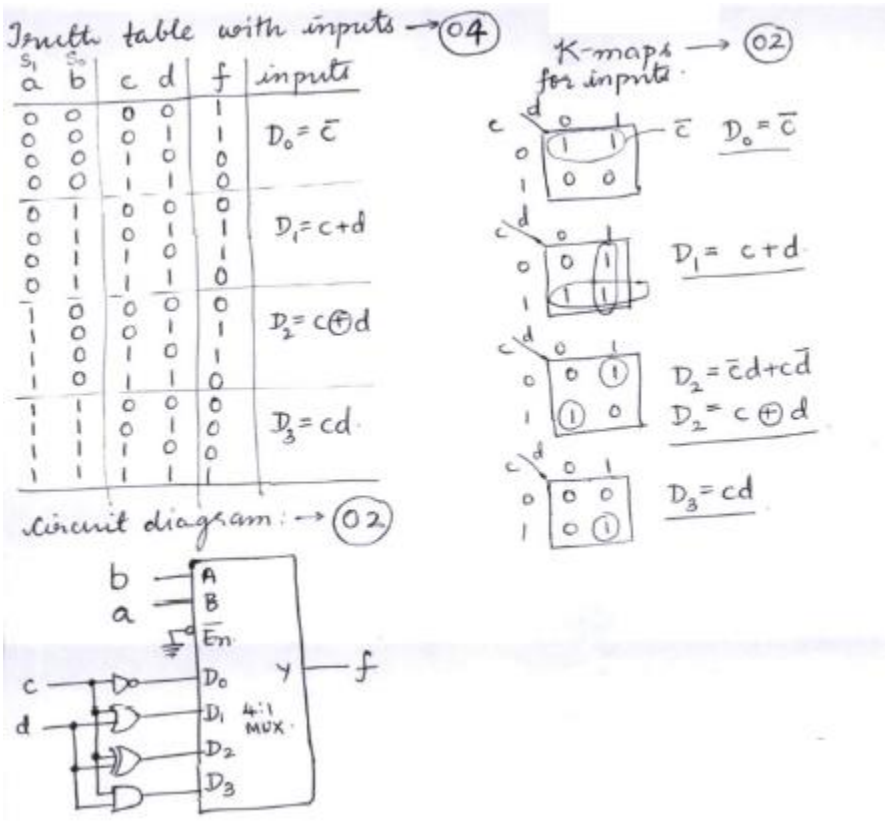
b.



c.



4a.



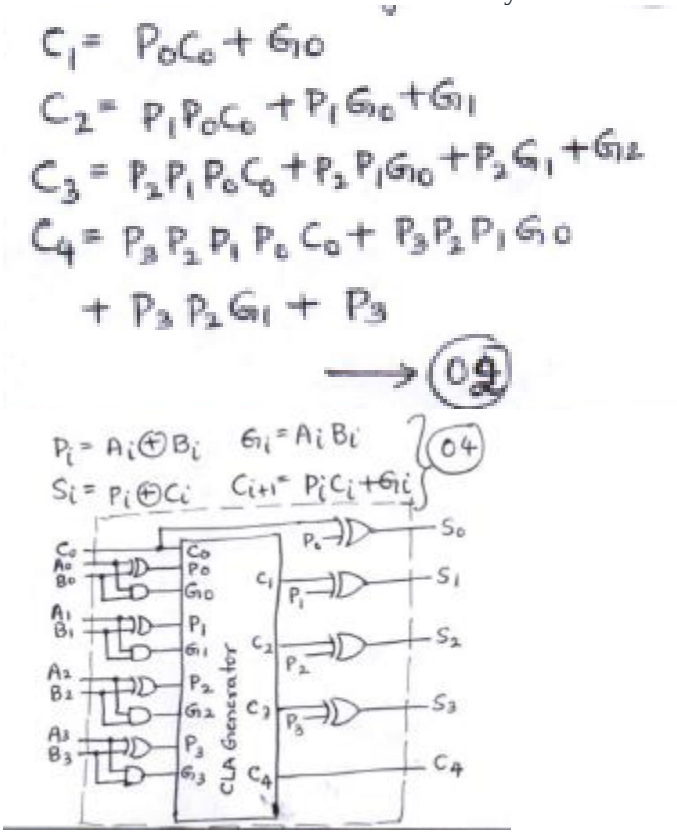
b.

The adder produces carry propagation delay while performing other arithmetic operations like multiplication and divisions as it uses several additions or subtraction steps. This is a major problem for the adder and hence improving the speed of addition will improve the speed of all other arithmetic operations. Hence reducing the carry propagation delay of adders is of great importance. There are different logic design approaches that have been employed to overcome the carry propagation problem. One widely used approach is to employ a carry look-ahead which solves this problem by calculating the

carry signals in advance, based on the input signals. This type of adder circuit is called a carry look-ahead adder.

Here a carry signal will be generated in two cases:

1. Input bits A and B are 1
2. When one of the two bits is 1 and the carry-in is 1.

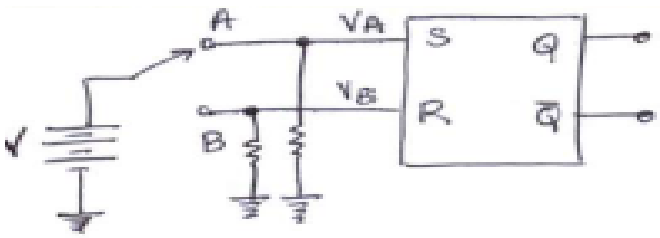


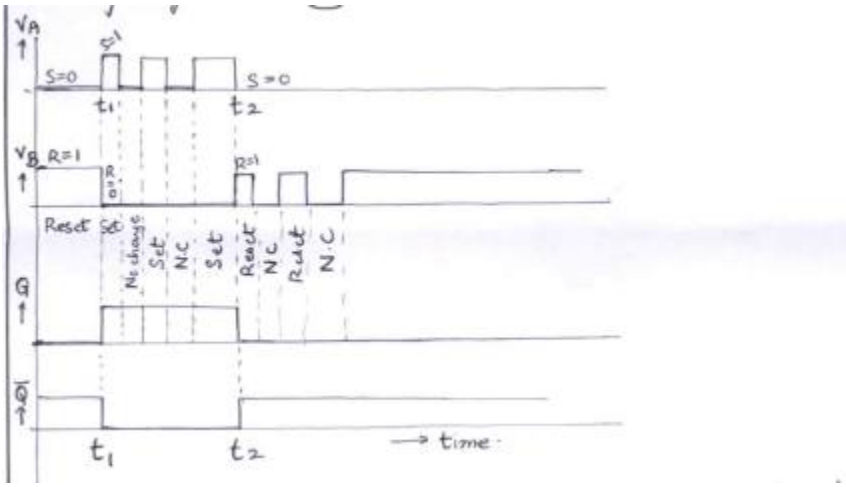
Module 3

5a.

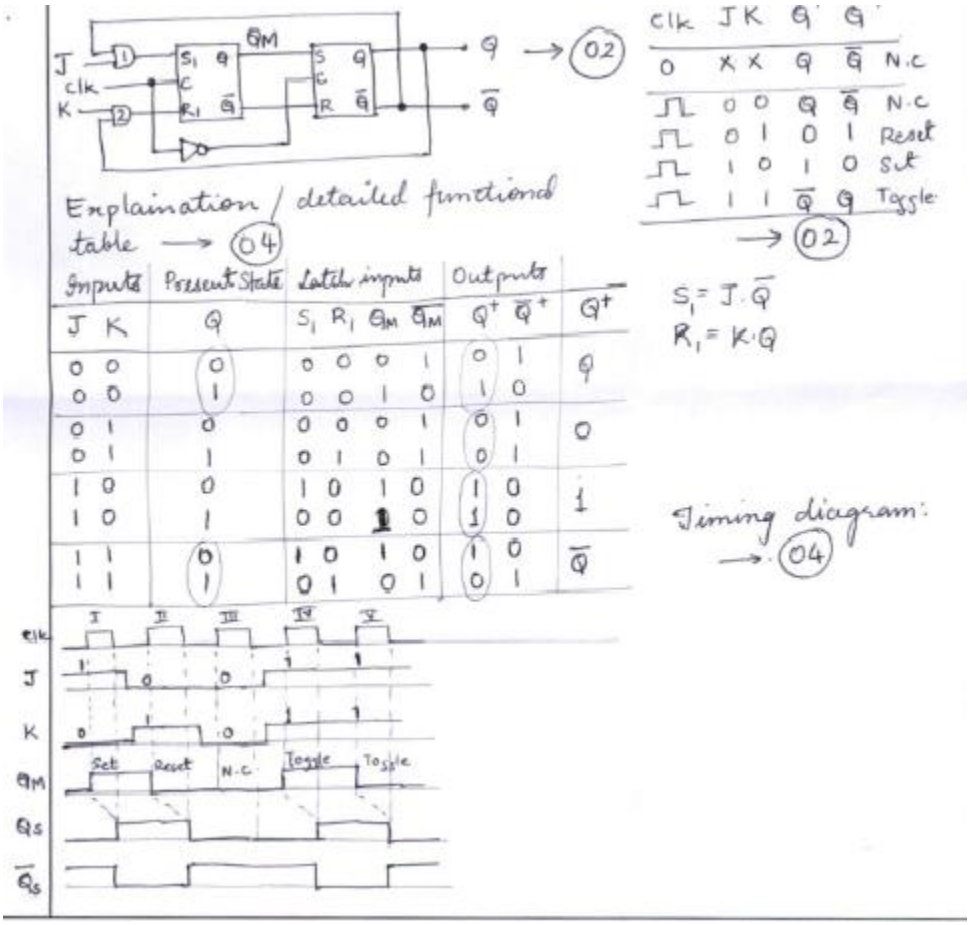
Depending upon the current state of the output, if the set or reset buttons are depressed the output will change over in the manner described above and any additional unwanted inputs (bounces) from the mechanical action of the switch will have no effect on the output at Q.

When the other button is pressed, the very first contact will cause the latch to change state, but any additional mechanical switch bounces will also have no effect. The SR flip-flop can then be RESET automatically after a short period of time

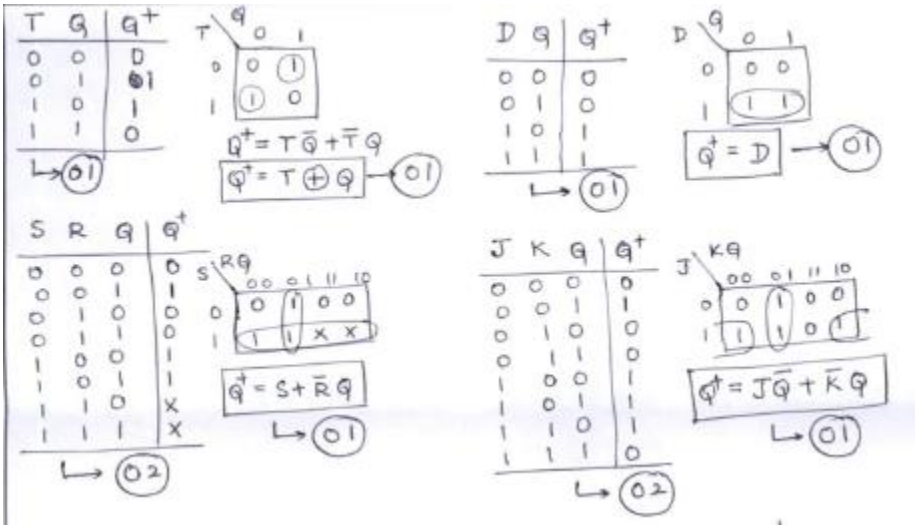



VA
S=0
S=1
S=0
t1
t2
VB
R=1
R=0
R=1
Reset
Set
No change
Set
N.C
Set
Reset
N.C
Reset
N.C
Q
Q̄
t1
t2
time

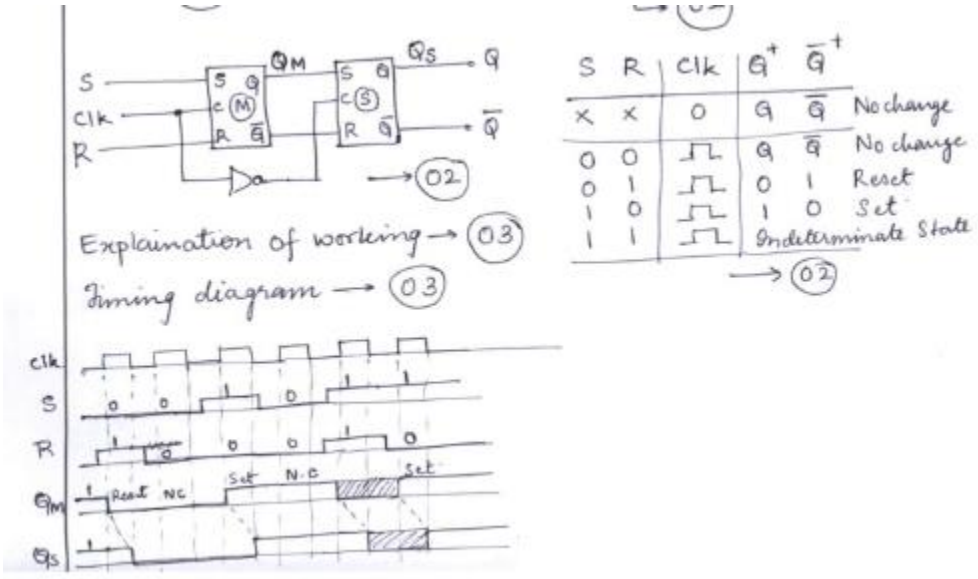
b.


J
clk
K
S1
C
R1
Q
Q̄M
S
C
R
Q
Q̄
Q
Q̄
02
clk J K Q Q̄
0 X X Q Q̄ N.C
0 0 Q Q̄ N.C
0 1 0 1 Reset
1 0 1 0 Set
1 1 Q̄ Q Toggle
02
Explaination / detailed functional table → 04
Inputs | Present State | Latch inputs | Outputs
J K | Q | S1 R1 Q̄M QM | Q+ Q̄+ | Q+
0 0 | 0 | 0 0 0 1 | 0 1 | Q
0 0 | 1 | 0 0 1 0 | 1 0
0 1 | 0 | 0 0 0 1 | 0 1 | 0
0 1 | 1 | 0 1 0 1 | 0 1
1 0 | 0 | 1 0 1 0 | 1 0 | 1
1 0 | 1 | 0 0 1 0 | 1 0
1 1 | 0 | 1 0 1 0 | 1 0 | Q̄
1 1 | 1 | 0 1 0 1 | 0 1
S1 = J·Q̄
R1 = K·Q
Timing diagram: → 04
clk
J
K
QM
Qs
Q̄s
I
II
III
IV
V
Set
Reset
N.C
Toggle
Toggle

6a.



b.



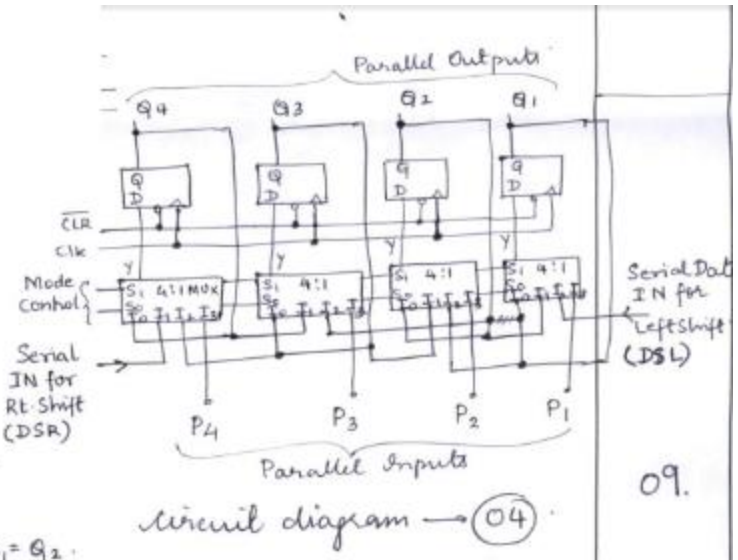
Module 4

7a.

7a.

S_1	S_0	Operation
0	0	HOLD
0	1	Shift Right
1	0	Shift Left
1	1	Parallel Load

→ (01)



Circuit diagram → (04)

09.

Explanation: (04)

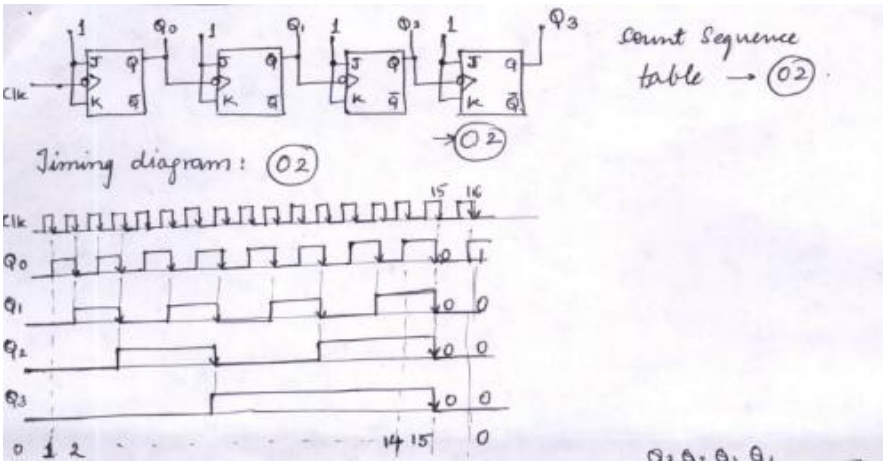
1) HOLD: $S_1=0$ $S_0=0$
I_0 s selected.
$D_4=Q_4$, $D_3=Q_3$, $D_2=Q_2$, $D_1=Q_1$.

2) Shift Right: $S_1=0$ $S_0=1$
I_1 s selected.
$D_4=DSR$, $D_3=Q_4$, $D_2=Q_3$, $D_1=Q_2$.

3) Shift Left: $S_1=1$ $S_0=0$
I_2 s selected. $D_1=DSL$, $D_2=Q_1$, $D_3=Q_2$, $D_4=Q_3$.

3) Parallel load $S_1=1$, $S_0=1$ I_3 s selected $D_4=P_4$, $D_3=P_3$, $D_2=P_2$, $D_1=P_1$

b.

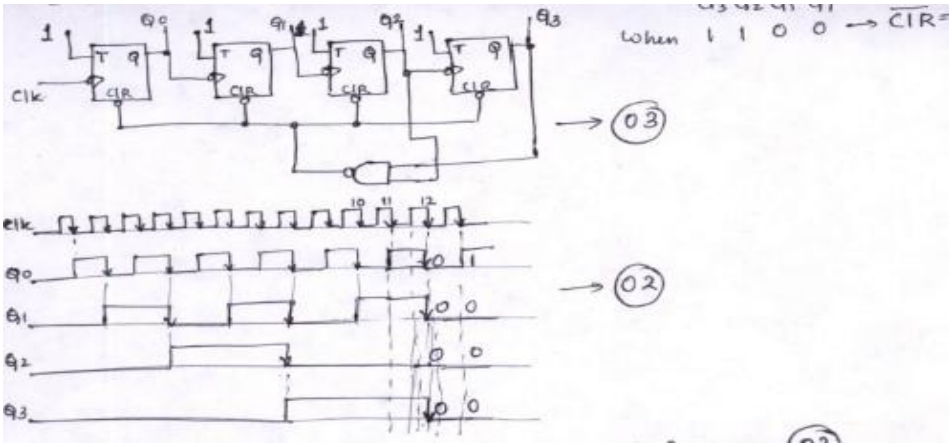


→ (02)

Count Sequence table → (02)

Timing diagram: (02)

c.

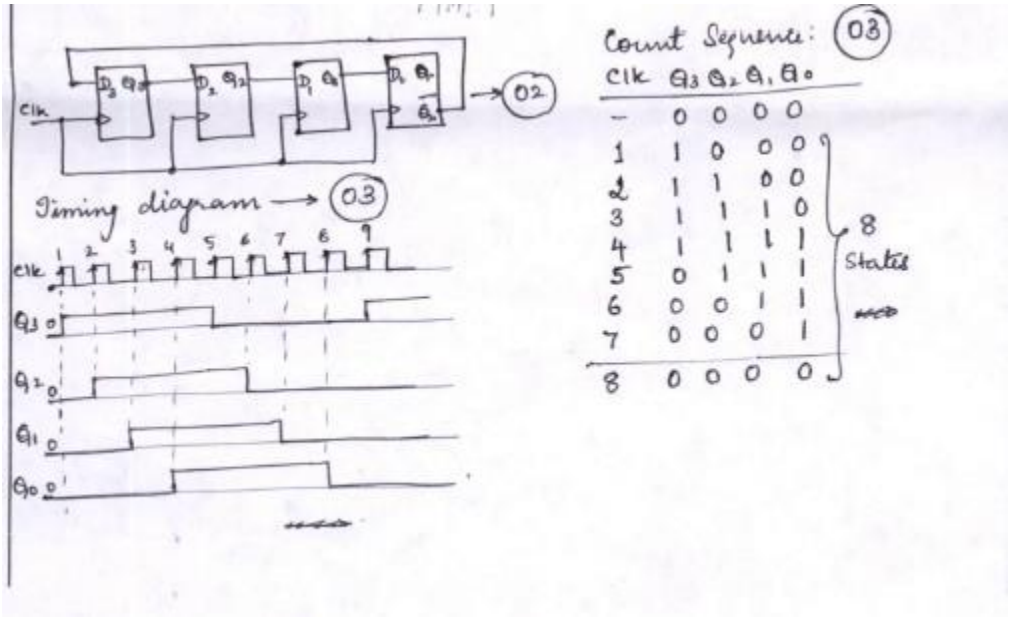


When 1 1 0 0 → $\overline{CLR}$=

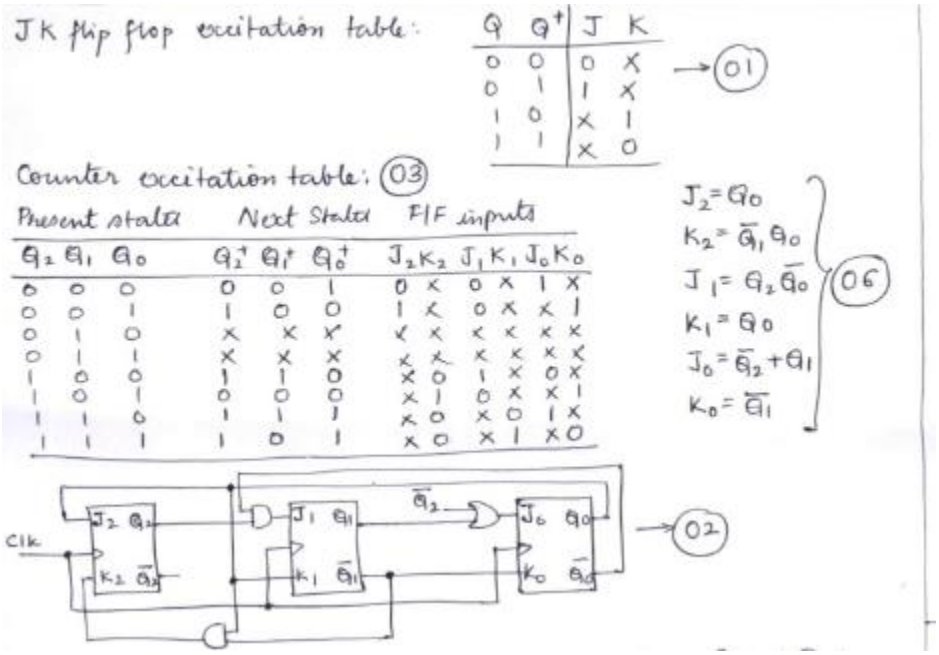
→ (03)

→ (02)

8a.



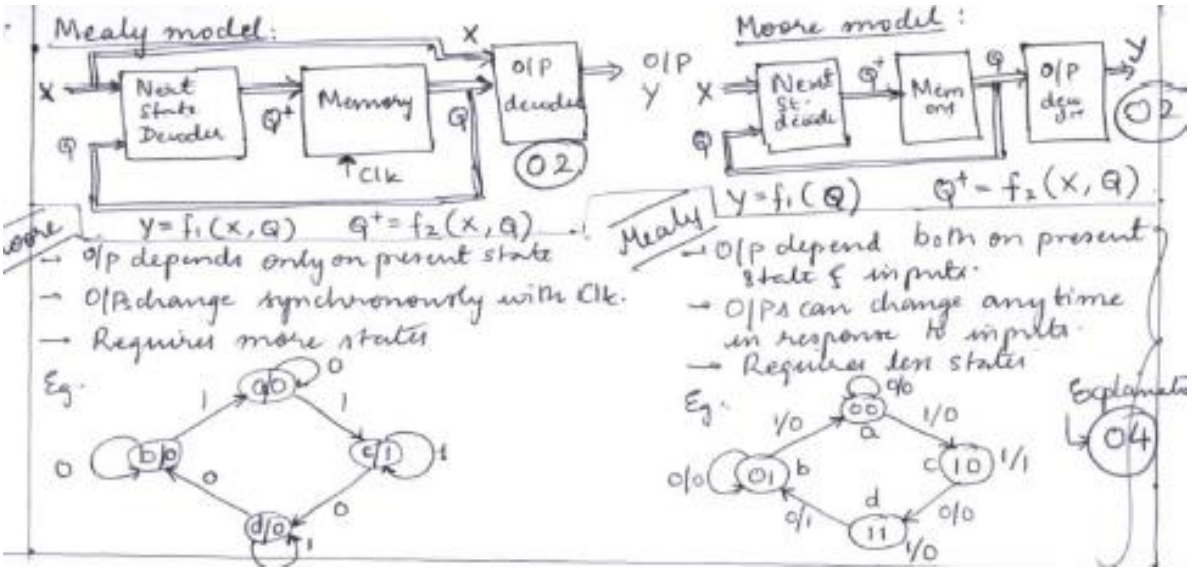
b.



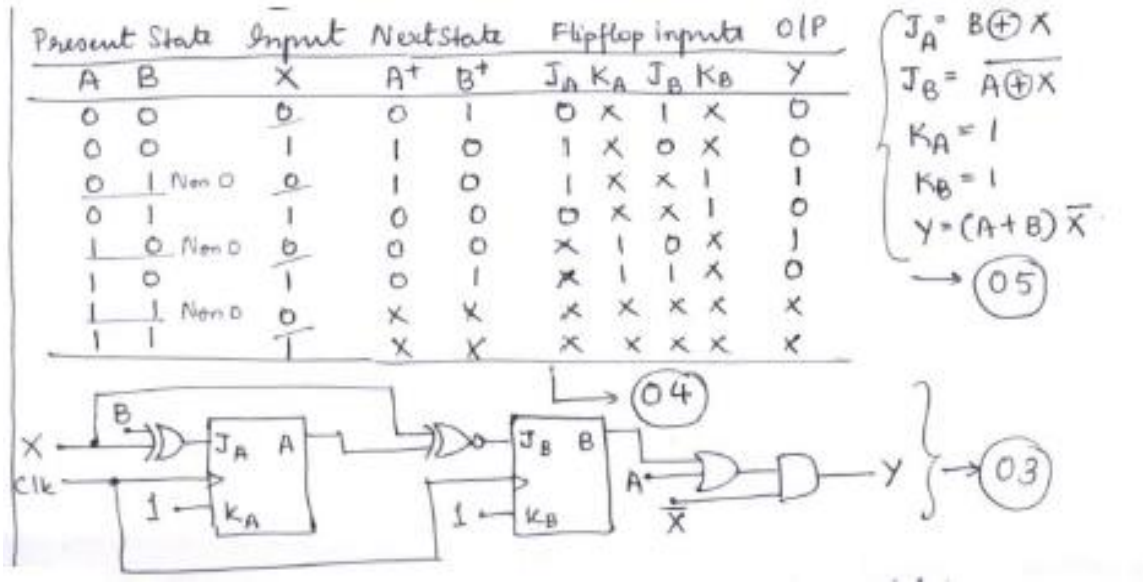
Module 5

9a.

In the theory of computation, a Mealy machine is a finite-state machine whose output values are determined both by its current state and the current inputs. This is in contrast to a Moore machine, whose output values are determined solely by its current state



b.



Present State		Input	Next State		Flip flop inputs				O/P
A	B	X	A^+	B^+	J_A	K_A	J_B	K_B	Y
0	0	0	0	1	0	X	1	X	0
0	0	1	1	0	1	X	0	X	0
0	1 Non 0	0	1	0	1	X	X	1	1
0	1	1	0	0	0	X	X	1	0
1	0 Non 0	0	0	0	X	1	0	X	1
1	0	1	0	1	X	1	1	X	0
1	1 Non 0	0	X	X	X	X	X	X	X
1	1	1	X	X	X	X	X	X	X

$J_A = B \oplus X$

$J_B = \overline{A \oplus X}$

$K_A = 1$

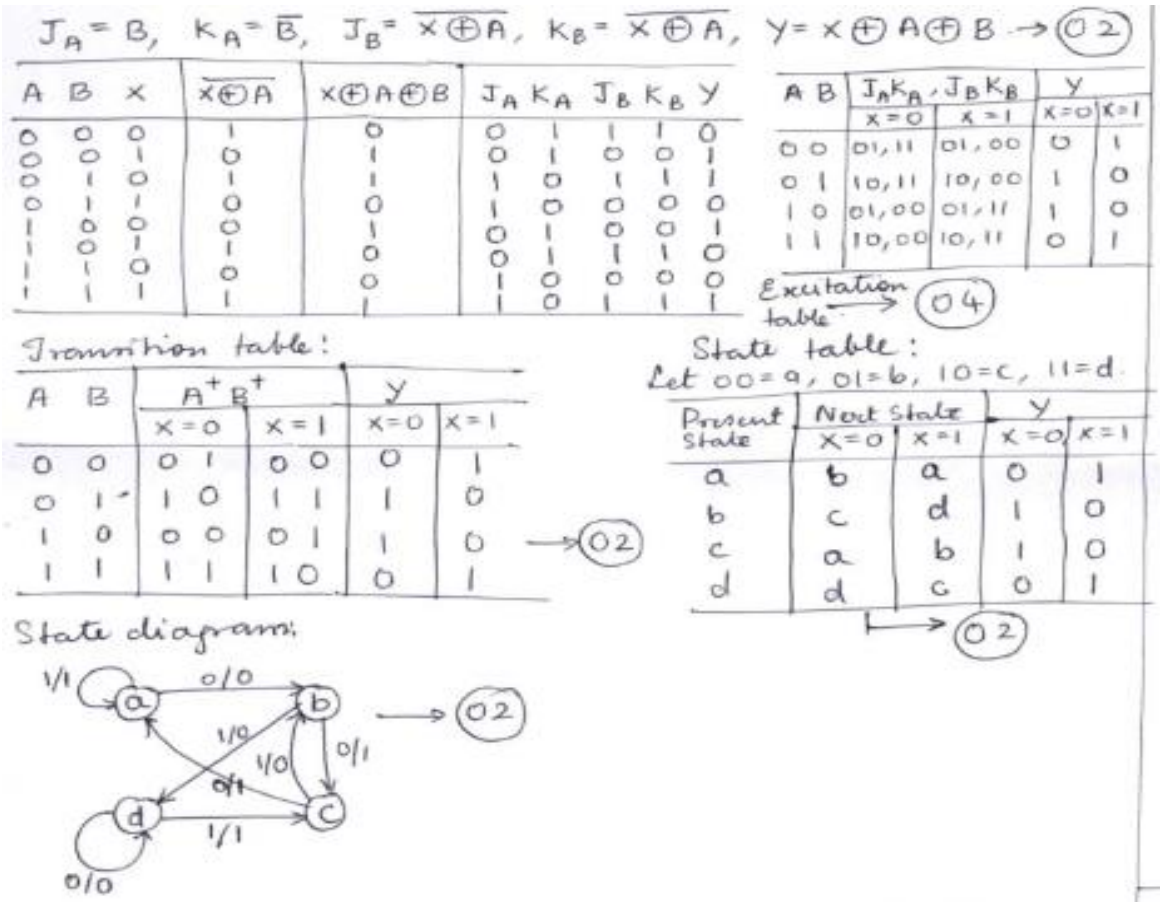
$K_B = 1$

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

→ O5

→ O4

10a.



b.

ROM stands for Read-Only Memory. It is a non-volatile memory that is used to store important information which is used to operate the system. As its name refers to read-only memory, we can only read the programs and data stored on it. It is also a primary memory unit of the computer system. It contains some electronic fuses that can be programmed for a piece of specific information. The information stored in the ROM in binary format. It is also known as permanent memory.

PROM (Programmable read-only memory): PROM is a form of digital memory. In this type of ROM, each bit is locked by a fuse or anti-fuse. The data stored in it are permanently stored and can not be changed or erasable. It is used in low-level programs such as firmware or microcode.

EPROM (Erasable programmable read-only memory): EPROM also called EROM, is a type of PROM but it can be reprogrammed. The data stored in EPROM can be erased and reprogrammed again by ultraviolet light. Reprogrammed of it is limited. Before the era of EEPROM and flash memory, EPROM was used in microcontrollers.

Flash memory is a long-life and non-volatile storage chip that is widely used in embedded systems. It can keep stored data and information even when the power is off. It can be

electrically erased and reprogrammed. Flash memory was developed from EEPROM (electronically erasable programmable read-only memory).