CBCS SCHEME

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

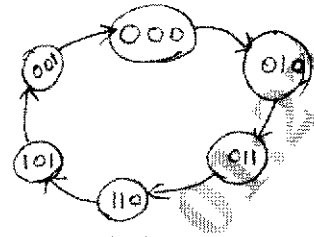
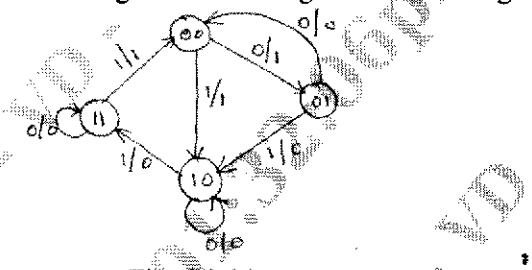
BEE306A

Third Semester B.E./B.Tech. Degree Examination, Dec.2024/Jan.2025
Digital Logic Circuits

Time: 3 hrs. Max. Marks: 100

Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.
2. M : Marks , L: Bloom's level , C: Course outcomes.

		Module – 1	M	L	C
Q.1	a.	Define combinational logic. What are Canonical forms in combinational logic? Explain the difference between SOP and POS canonical forms with suitable examples.	05	L1	CO1
	b.	What are prime implicant and essential prime implicant? Simplify the given Boolean function using KMap and identify its PI and EPI. $f(a,b,c,d) = \sum_m m(0,1,2,5,6,7,8,9,10,13,14,15)$	08	L2	CO1
	c.	Given $f(a,b,c) = \bar{a}b + ac + b\bar{c}$. Express it in canonical SOP and canonical POS.	07	L2	CO1
		OR			
Q.2	a.	Minimize the following expression in POS using K-map. $f(a,b,c,d) = \sum m(1,3,7,11,15) + \sum d(0,2,5)$	05	L3	CO1
	b.	Simplify the Boolean expression using K-map. $f(p,q,r,s,t) = \sum m(0,2,4,6,8,16,18,28,30) + dc(3,7,11,19,27,31)$	05	L3	CO1
	c.	Using Quine McCluskey and PI Reduction Table, determine minimal SOP for the function, $f(w,x,y,z) = \sum m(1,3,4,7,8,9,12,14) + \sum dc(2,13)$	10	L3	CO1
		Module – 2			
Q.3	a.	Write the Truth table for 8 line to 3 line priority Encoder.	04	L2	CO2
	b.	Write short note on look ahead carry adder.	06	L2	CO2
	c.	Design a combinational circuit to convert BCD to Excess 3 code.	10	L3	CO2
		OR			
Q.4	a.	Design full adder using a decoder (74138).	05	L3	CO2
	b.	Implement the following function using 8 : 1 MUX. Treat w, x, y as select lines, $f(w,x,y,z) = \sum m(0,1,5,6,7,9,10,15)$	05	L3	CO2
	c.	Implement 2 bit comparator using 4-16 decoder.	10	L3	CO2
		Module – 3			
Q.5	a.	With the help of logic diagram, explain the working of Master Slave JK flipflop and explain how race around condition is eliminated.	10	L2	CO3
	b.	Explain how SR flip flop/latch act as switch debouncing circuit and draw its waveform.	06	L2	CO3
	c.	Differentiate between latch and flip flop.	04	L2	CO3
		OR			
Q.6	a.	Derive the characteristics equation of SR, JK and T flip flop.	06	L3	CO3
	b.	With a logic diagram, explain the working of SR-flip flop.	07	L2	CO3
	c.	With the help of logic diagram, explain the working of positive edge Triggered D-flip flop.	07	L2	CO3

		Module – 4			
Q.7	**a.**	Design a synchronous counter for the following sequence using JK flip flop.  Fig. Q7 (a)	**10**	**L3**	**CO4**
	b.	Design synchronous MOD-6 counter using clocked T flip flop.	**10**	**L3**	**CO4**
		OR			
Q.8	**a.**	Design 4 bit register using positive edge triggered DFF to operate as below : Mode Select Register operation A_1 A_0 0 0 → Hold 0 1 → Shift Right 1 0 → Shift Left 1 1 → Parallel load	**12**	**L4**	**CO4**
	b.	Explain 4 bit SISO uni-directional shift register using DFF (Consider positive edge triggered).	**08**	**L2**	**CO4**
		Module – 5			
Q.9	**a.**	Design a sequential circuit using TFF for the figure shown in Fig. Q9 (a).  Fig. Q9 (a)	**12**	**L3**	**CO5**
	b.	With a neat diagram, explain mealy and Moore model in a sequential circuit analysis.	**08**	**L2**	**CO4**
		OR			
Q.10	**a.**	Write short note on : (i) Read / write memories (ii) Programmable ROM	**10**	**L2**	**CO6**
	b.	Write short note on : (i) Flash memory (ii) EPROM	**10**	**L2**	**CO6**

* * * * *

Module-1

Q. 1/a) **Definition of Combinational Logic** –

A digital logic circuit where the output at any given time depends solely on the current input values, without any memory of past inputs.
Ex - Half adder, full adder etc.

Canonical forms –

In combinational logic, canonical forms are standardized ways to represent Boolean expressions, making it easier to analyze & design logic circuits.

Difference between SOP & POS canonical form –

SOP & POS are two types of canonical forms used to represent Boolean fns.

SOP (Sum of Products):

→ Represents a Boolean fn. as a sum of product terms (minterms), where the fn. value is '1'.
→ Each minterm contains all input variables, either in their original or complemented form, and corresponds to one input combination that results in a logic '1' output.
→ SOP uses AND operations to form each product term and OR operations to combine the terms.
Ex- $f(a, b, c) = \bar{A}B\bar{C} + A\bar{B}C$

POS (Product of Sums):

→ Represents a Boolean fn. as a product of sum terms (maxterms), where the fn. value is '0'.
→ Each maxterm contains all input variables,

either in original or complemented form, and corresponds to one input combination that results in a '0' output.
→ POS uses 'OR' operations to form each sum term and 'AND' operations to combine the terms.
Ex- $f(A, B, C) = (A+B+C)(A+\bar{B}+\bar{C})$

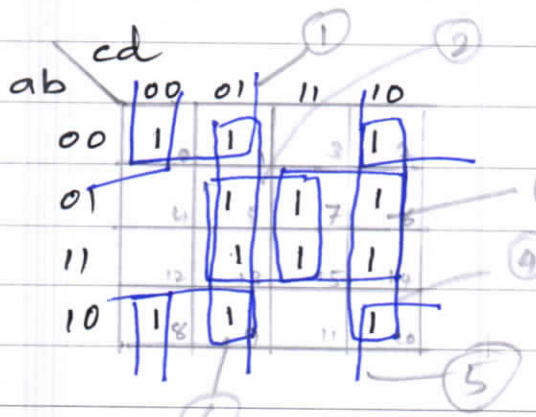
(b) Prime implicant - (PI's)

Prime Implicants (PI's) in digital logic and Boolean algebra are groups of adjacent minterms on a K-map that can not be further grouped to create larger, more simplified terms. They represent the largest possible groupings of 1's (or 0's in some cases) that are allowed by K-map's definition.

Essential Prime Implicant - (EPI's)

Essential Prime Implicants (EPI's) in K-maps are prime implicants that must be included in the simplified Boolean expression to minimize the function. They are characterized by covering at least one minterm that can not be covered by any other prime implicant.

Given fn. $f(a,b,c,d) = \Sigma m(0,1,2,5,6,7,8,9,10,13,14,15)$



The prime implicants are: $\bar{b}\bar{c}$, bd, bc, $c\bar{d}$, $\bar{b}\bar{d}$, $\bar{c}d$.

There are no EPI's.

(c) Given $f(a,b,c) = \bar{a}b + ac + b\bar{c}$ → SOP

$\therefore \bar{a}b = \bar{a}b(c+\bar{c}) = \bar{a}bc + \bar{a}b\bar{c}$

$ac = ac(b+\bar{b}) = abc + a\bar{b}c$

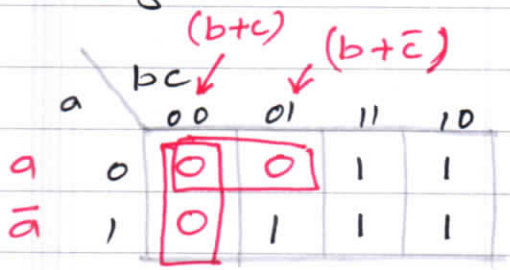
$b\bar{c} = b\bar{c}(a+\bar{a}) = ab\bar{c} + \bar{a}b\bar{c}$

$\therefore f = \bar{a}bc + \bar{a}b\bar{c} + abc + a\bar{b}c + ab\bar{c} + \bar{a}b\bar{c}$

$\therefore$ Canonical SOP is, $f = \bar{a}bc + \bar{a}b\bar{c} + abc + a\bar{b}c + ab\bar{c}$

Next, procedure to obtain in Canonical POS form ⇒

i) Writing SOP in K-map ⇒



ii) Grouping 0's (i.e., max terms) →

$\therefore f(a,b,c) = (a+b)(b+c)$

iii) Expand in canonical POS form → i.e.,

$(a+b) + (c\bar{c}) = (a+b+c)(a+b+\bar{c})$

& $(b+c) + (a\bar{a}) = (a+b+c)(\bar{a}+b+c)$

$\therefore f = (a+b+c)(a+b+\bar{c})(a+b+c)(\bar{a}+b+c)$

$\therefore$ Canonical POS is, $f = (a+b+c)(a+b+\bar{c})(\bar{a}+b+c)$

Q.2) a) Given $f(a,b,c,d) = \sum m(1,3,7,11,15) + \sum d(0,2,5)$

ab \ cd	00	01	11	10
00	X	1	1	X
01	0	X	1	0
11	0	0	1	0
10	0	0	1	0

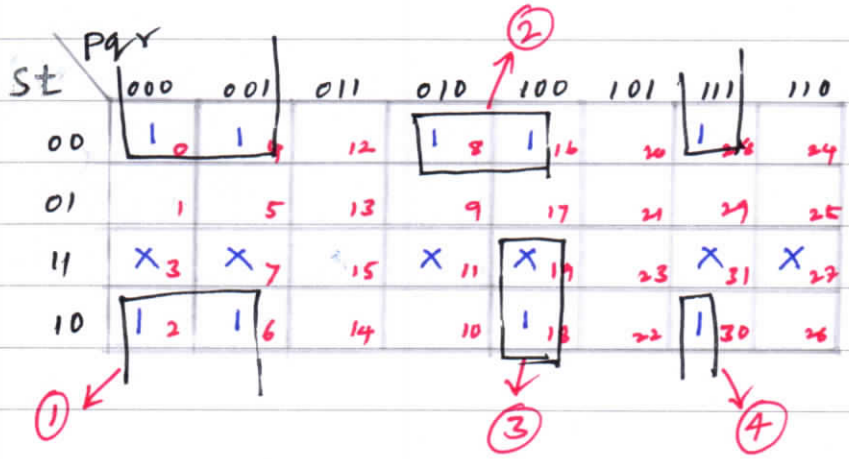
1 → $(c+d)$

2 → $(\bar{a}+c)$

3 → $(\bar{c}+d)$

∴ POS is $f(a,b,c,d) = (c+d)(\bar{a}+c)(\bar{c}+d)$

(b) Given $f(P,q,r,s,t) = \sum m(0,2,4,6,8,16,18,28,30) + dc(3,7,11,19,27,31)$



∴ $f(P,q,r,s,t) = \bar{P}\bar{q}\bar{t} + \bar{r}\bar{s}t + P\bar{q}\bar{r}\bar{s} + pqr\bar{t}$

c) Given, $f(w,x,y,z) = \sum m(1,3,4,7,8,9,12,14) + \sum dc(2,13)$

i) construct a list of minterms & don't care terms classified according to number of 1's (index)

Column-1

minterm	w x y z	index	
1	0 0 0 1	1	✓
2*	0 0 1 0	1	✓
4	0 1 0 0	1	✓
8	1 0 0 0	1	✓
3	0 0 1 1	2	✓
9	1 0 0 1	2	✓
12	1 1 0 0	2	✓
7	0 1 1 1	3	✓
13*	1 1 0 1	3	✓
14	1 1 1 0	3	✓

Column-2

minterm	w x y z	index	
1,3	0 0 − 1	1	✗
1,9	− 0 0 1	1	✗
2*,3	0 0 1 −	1	✗
4,12	− 1 0 0	1	✗
8,9	1 0 0 −	1	✓
8,12	1 − 0 0	1	✓
3,7	0 − 1 1	2	✗
9,13*	1 − 0 1	2	✗
12,13*	1 1 0 −	2	✓
12,14	1 1 − 0	2	✗

Column-3

minterm	w x y z	index	
8,9,12,13*	1 − 0 −	1	✗
8,12,9,13	1 − 0 −	1	→ same as 8,9,12,13 ∴ Neglect.

- The minterms marked with 'x' mark are prime implicants (PI's).
- Replace 0's & 1's by complemented & uncomplemented variables

∴ PI's are → $w\bar{y}$, $wx\bar{z}$, $w\bar{y}z$, $\bar{w}yz$, $x\bar{y}\bar{z}$, $\bar{w}\bar{x}y$, $\bar{x}\bar{y}z$, $\bar{w}\bar{x}z$

All of the above PI's are placed in Prime Implicant table to find EPI's.

PI	Min terms	Minterms														
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$w\bar{y}$	8, 9, 12, 13*									(×)	×			×	×		
$wx\bar{z}$	12, 14													×		(×)	
$w\bar{y}z$	9, 13*										×				×		
$\bar{w}yz$	3, 7				×				(×)								
$x\bar{y}\bar{z}$	4, 12					(×)								×			
$\bar{w}\bar{x}y$	2*, 3			(×)	×												
$x\bar{y}z$	1, 9		×								×						
$\bar{w}xz$	1, 3		×		×												

∴ EPI's are →

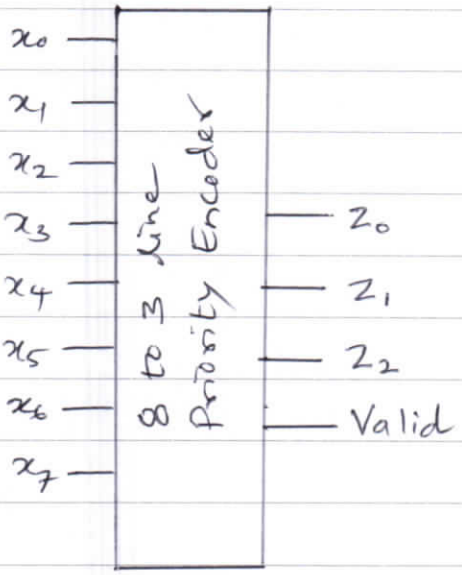
$w\bar{y}$, $wx\bar{z}$, $\bar{w}yz$, $x\bar{y}\bar{z}$, $\bar{w}\bar{x}y$

$\therefore f(w, x, y, z) = w\bar{y} + wx\bar{z} + \bar{w}yz + x\bar{y}\bar{z} + \bar{w}\bar{x}y$

Module - 2

Q. 3) a)

Truth table for 8-line to 3 line priority encoder.



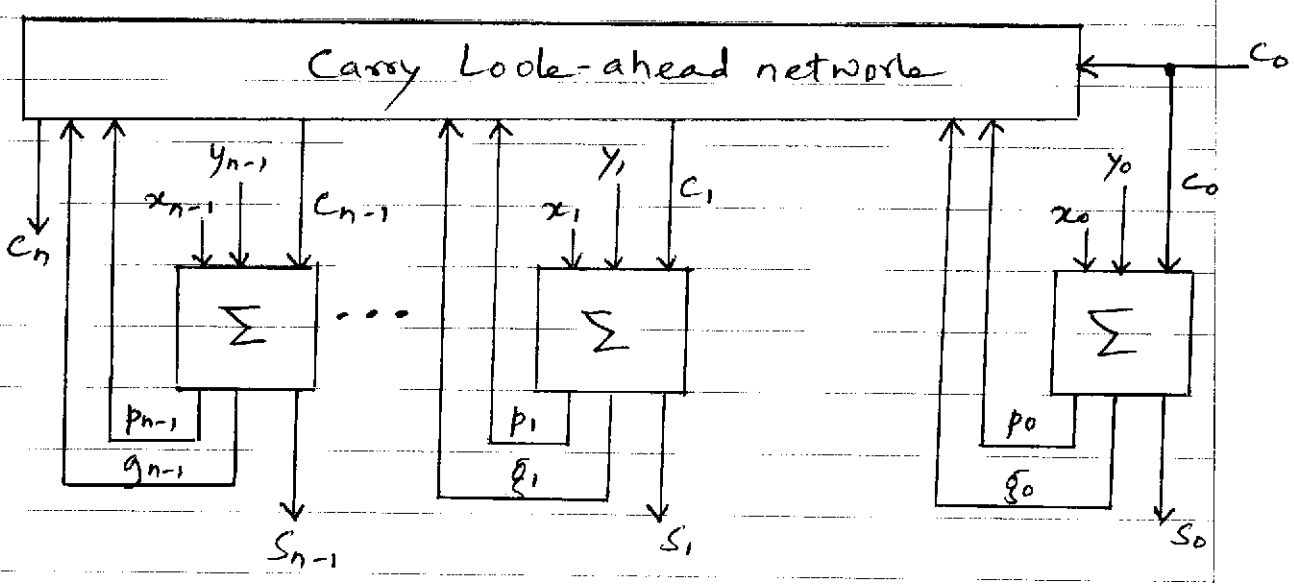
Truth - Table

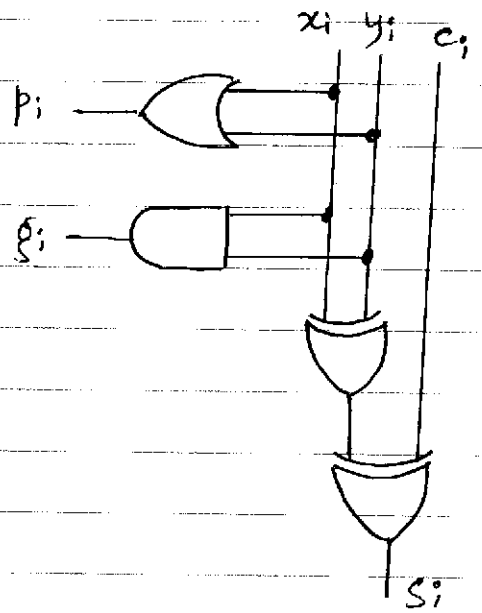
Inputs								Outputs			
x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7	Z_2	Z_1	Z_0	Valid
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
x	1	0	0	0	0	0	0	0	0	1	1
x	x	1	0	0	0	0	0	0	1	0	1
x	x	x	1	0	0	0	0	0	1	1	1
x	x	x	x	1	0	0	0	1	0	0	1
x	x	x	x	x	1	0	0	1	0	1	1
x	x	x	x	x	x	1	0	1	1	0	1
x	x	x	x	x	x	x	x	1	1	1	1

An output labeled "valid" is included in the Truth table to indicate that at least one input line is asserted. This is done so as to distinguish the situation that no input line is asserted from when the x_0 input line is asserted, since in both cases $Z_2Z_1Z_0 = 000$.

(b) Lookahead Carry Adder -

The loole-ahead carry or fast carry technique reduces propagation delay thro' the adder. This is accomplished by generating all the individual carry terms needed by each full-adder in as few levels of logic as possible.





Sigma block.

Equations for sum & carry for the outputs at the i^{th} stage of a binary adder are;

$$S_i = c_i \oplus (x_i \oplus y_i) \quad \text{—— (1)}$$

$$C_{i+1} = x_i y_i + x_i c_i + y_i c_i \quad \text{——} (2)$$

Eqn (2) is the output carry at the i^{th} stage i.e.,

$$C_{i+1} = x_i y_i + x_i c_i + y_i c_i$$
$$= x_i y_i + (x_i + y_i) c_i$$

The first term "$x_i y_i$" is called the carry-generate function since it corresponds to the formation of a carry at the i^{th} stage. The second term "$(x_i + y_i) c_i$" corresponds to a previously generated carry c_i, that must propagate past the i^{th} stage to the next stage. The "$(x_i + y_i)$" part of this term is called the "carry-propage" function.

Let the carry-generate fn be denoted by g_i & the carry propagate fn by p_i i.e.,

$$g_i = x_i y_i \quad \text{——} (3)$$
$$p_i = x_i + y_i \quad \text{——} (4)$$

The o/p carry equation for the i^{th} stage is given by, $C_{i+1} = g_i + p_i c_i$

Using this general result, the output carry at each of the stages can be written in terms of carry-generate functions, the carry-propagate fns, with initial input carry 'c_0' as follows:

$$C_1 = g_0 + p_0 c_0 \quad \text{——} (5)$$
$$C_2 = g_1 + p_1 g_1 + p_1 p_0 c_0 \quad \text{——} (6)$$
$$C_3 = g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 c_0 \quad \text{——} (7)$$
$$\vdots$$
$$C_{i+1} = g_i + p_i g_{i-1} + p_i p_{i-1} g_{i-2} + \ldots + p_i p_{i-1} \cdots p_1 g_0 + p_i p_{i-1} \cdots p_0 c_0$$
$$\text{——} (8)$$

Parallel adders whose realizations are based on the above equations are called "Carry Look-ahead adders"

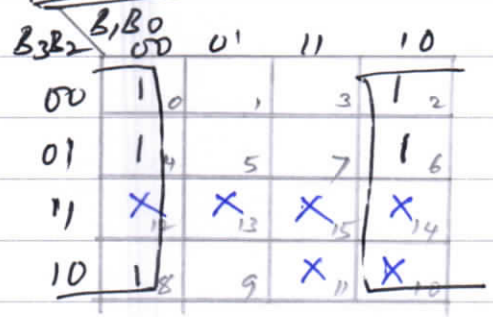
(c) Combinational Ckt. to convert BCD to Excess 3 code-

Truth table

	BCD code				Excess-3 code				
	B_3	B_2	B_1	B_0	E_3	E_2	E_1	E_0	
0	0	0	0	0	0	0	1	1	3
1	0	0	0	1	0	1	0	0	4
2	0	0	1	0	0	1	0	1	5
3	0	0	1	1	0	1	1	0	6
4	0	1	0	0	0	1	1	1	7
5	0	1	0	1	1	0	0	0	8
6	0	1	1	0	1	0	0	1	9
7	0	1	1	1	1	0	1	0	10
8	1	0	0	0	1	0	1	1	11
9	1	0	0	1	1	1	0	0	12

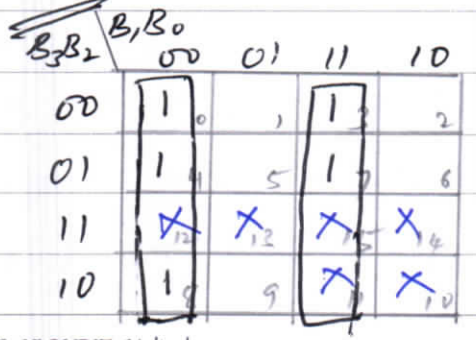
Now eqns for o/p variables can be generated using K-maps:

for E_0



$\therefore E_0 = \overline{B_0}$

for E_1

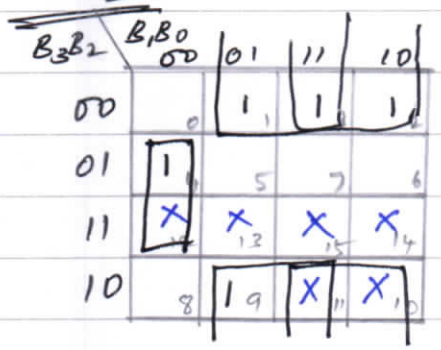


$\therefore E_1 = \overline{B_1}\,\overline{B_0} + B_1 B_0$

$= \overline{B_1 \oplus B_0}$

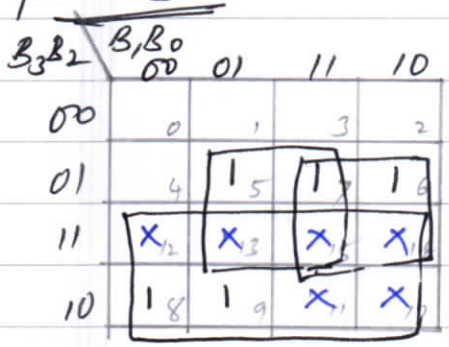
or $E_1 = B_1 \odot B_0$

for E_2



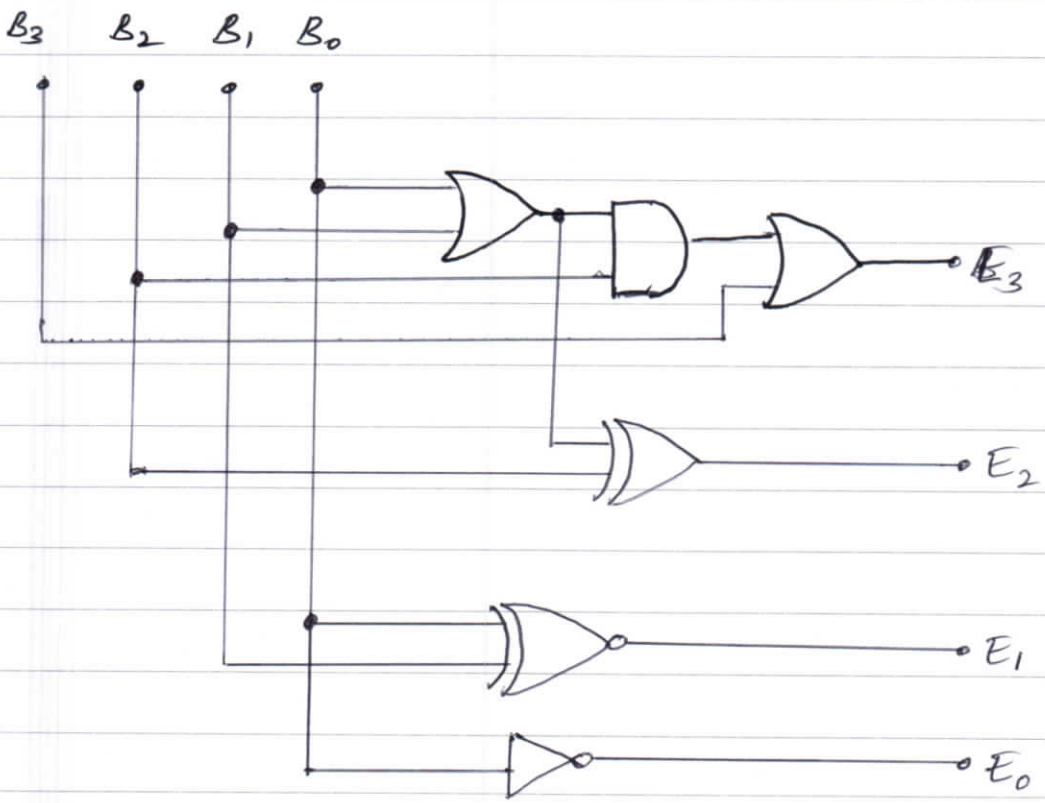
$$\therefore E_2 = B_2\bar{B}_1\bar{B}_0 + \bar{B}_2B_0 + \bar{B}_2B_1$$
$$= B_2\bar{B}_1\bar{B}_0 + \bar{B}_2(B_0 + B_1)$$

for E_3

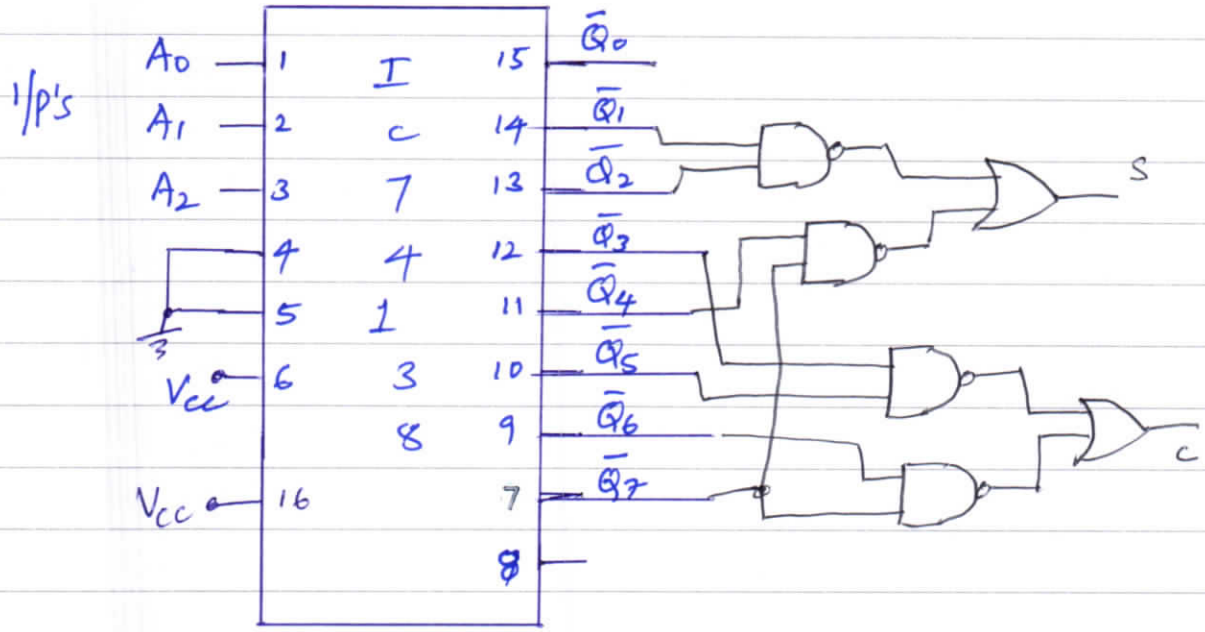


$$\therefore E_3 = B_3 + B_2B_0 + B_2B_1$$
$$= B_3 + B_2(B_0 + B_1)$$

$\therefore$ Next draw the circuit diagram :



Q. 4) a) Full adder using a decoder (74138):



Truth Table of Full adder

Dec. equ.	I/P			O/P S	O/P C
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

∴ Minterms for O/P S & C are

$S = \Sigma m(1, 2, 4, 7)$

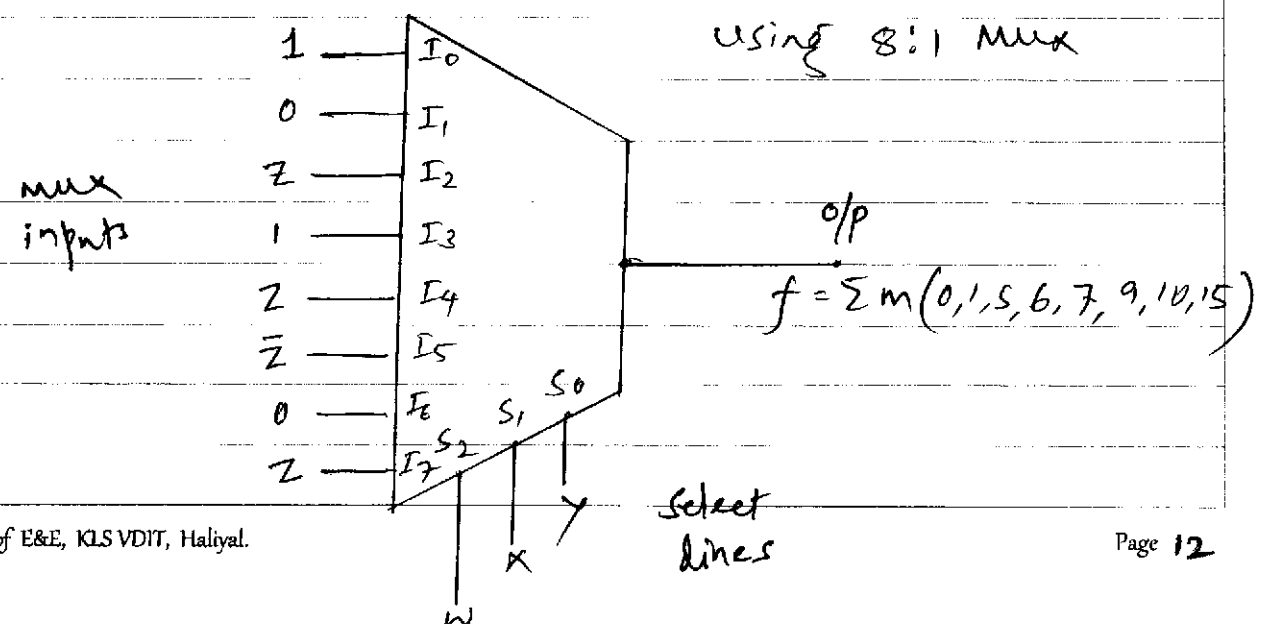
$C = \Sigma m(3, 5, 6, 7)$

(b) Given $f = (W, X, Y, Z) = \Sigma m(0, 1, 5, 6, 7, 9, 10, 15)$.

Using 8:1 mux treating W, X, Y as select lines.

Truth Table

	Data Inputs				Data o/p		
	Select lines			Mux			
	W	X	Y	Z	f	MEV	
0	0	0	0	0	1	1	I_0
1	0	0	0	1	1		
2	0	0	1	0	0	0	I_1
3	0	0	1	1	0		
4	0	1	0	0	0	Z	I_2
5	0	1	0	1	1		
6	0	1	1	0	1	1	I_3
7	0	1	1	1	1		
8	1	0	0	0	0	Z	I_4
9	1	0	0	1	1		
10	1	0	1	0	1	$\bar{Z}$	I_5
11	1	0	1	1	0		
12	1	1	0	0	0	0	I_6
13	1	1	0	1	0		
14	1	1	1	0	0	Z	I_7
15	1	1	1	1	1		



(c) 2-bit comparator using 4:16 decoder -

Let 'A' & 'B' be two bits ie,

$A \rightarrow A_1 A_0$ & $B \rightarrow B_1 B_0$

Truth Table

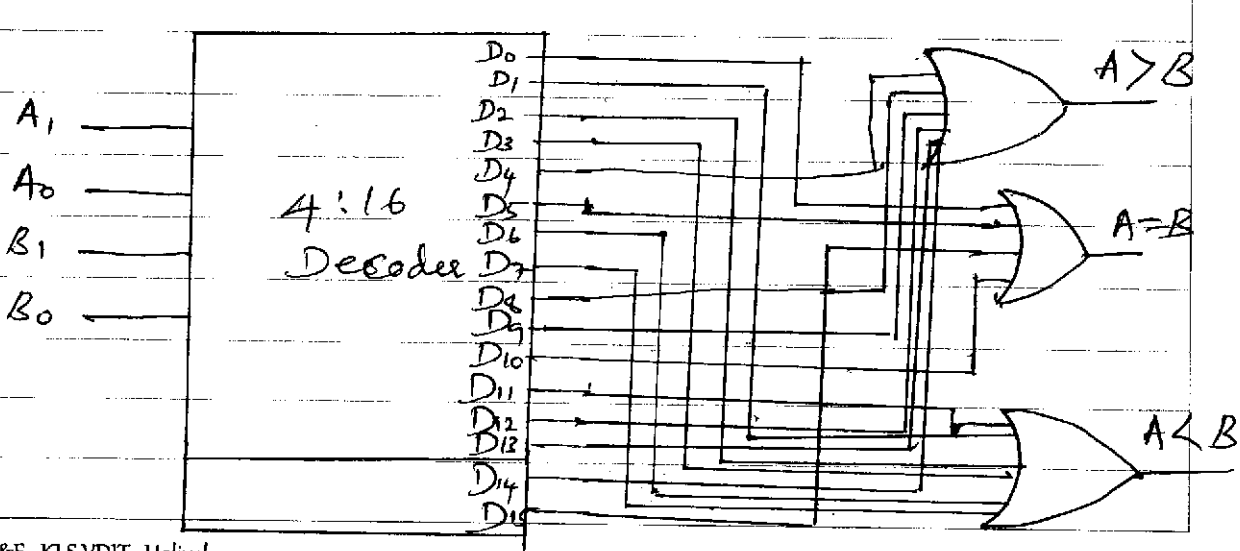
	Inputs				outputs		
	A_1	A_0	B_1	B_0	A>B	A=B	A<B
0	0	0	0	0		1	
1	0	0	0	1			1
2	0	0	1	0			1
3	0	0	1	1			1
4	0	1	0	0	1		
5	0	1	0	1		1	
6	0	1	1	0			1
7	0	1	1	1			1
8	1	0	0	0	1		
9	1	0	0	1	1		
10	1	0	1	0		1	
11	1	0	1	1			1
12	1	1	0	0	1		
13	1	1	0	1	1		
14	1	1	1	0	1		
15	1	1	1	1		1	

∴ Minterms are

$A>B = \Sigma m(4, 8, 9, 12, 13, 14)$

$A=B = \Sigma m(0, 5, 10, 15)$

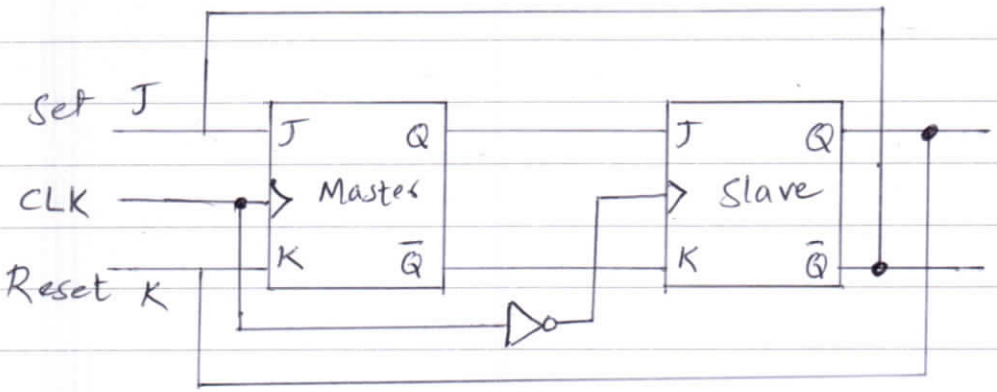
$A<B = \Sigma m(1, 2, 3, 6, 7, 11)$



Module-3

Q5) a) **Master-Slave JK FF** –

Master-slave JK FF is a combination of two JK FF's which are connected in the cascaded manner.



In this combination of two JK FF's one acts as a Master FF & the other acts as a slave FF. In this MS FF, the O/p's of Master JK FF are connected to the inputs of the slave JK FF. The O/p's of the slave FF are fed backe to the inputs of the Master JK FF. An inverted clocke pulse is applied to clocke input of the slave FF.

Truth Table

I/p's			O/p		
J	K	Clk	Q^+	$\bar{Q}^+$	comment
0	0	⊓	Q	$\bar{Q}$	No change or Hold state
0	1	⊓	0	1	Reset
1	0	⊓	1	0	Set
1	1	⊓	$\bar{Q}$	Q	Toggle
X	X	0	Q	$\bar{Q}$	

→ When J=0 & K=0; both JK FF's remains inactive & the output remains unchanged.

→ When J=0 & K=1; the o/p $\bar{Q}$ of the Master FF is high & goes to the i/p 'K' of the slave FF. The clock signal forces the slave FF to reset. Therefore, the slave FF has the same o/p as the master FF ie, high $\bar{Q}$ & low Q. This is called Reset state of MS JK FF.

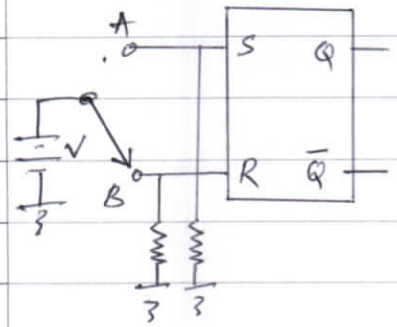
→ When J=1 & K=0; the o/p $\bar{Q}$ of the Master FF is high & goes to the i/p of the slave FF, the negative transition of the clock signal sets the slave FF, and this is called the set state of the MS JK FF.

→ When J=1 & K=1; the master FF toggles on the positive transition of the clock pulse and the slave FF toggles on the negative transition of the clock pulse.

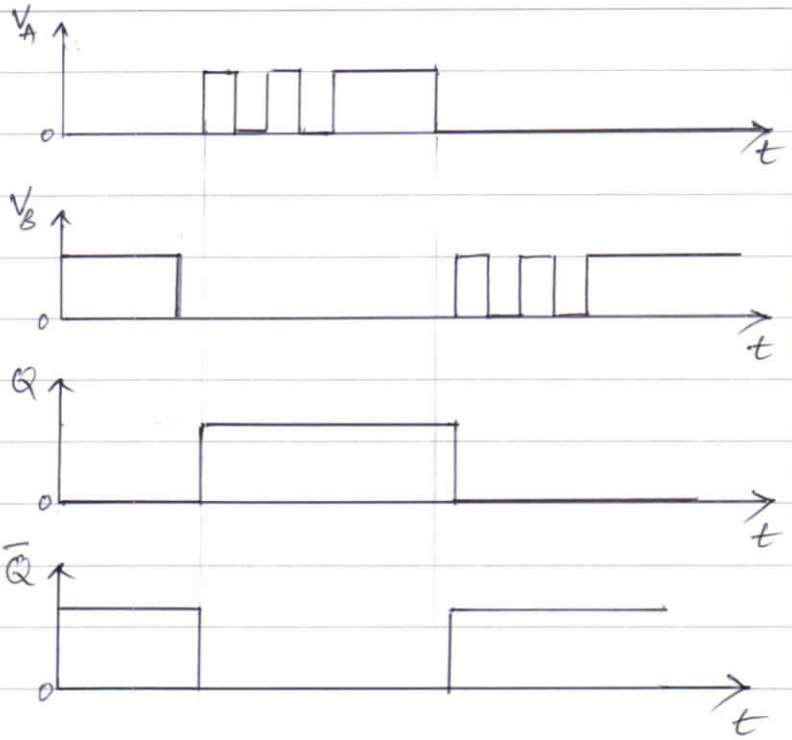
In MS JK FF, the race around condition is eliminated by ensuring the clock pulse to the slave flip-flop is inverted, preventing the slave from changing the state during the master's state change, which resolves the unstable o/p.

(b) **SR FF as a switch debouncing circuit –**

An SR FF can be used as a debouncing ckt. to filter out unwanted noise or bouncing from switch contacts. By connecting the switch to the S & R inputs of the flip-flop, the output will become stable and ignore the brief oscillations caused by switch contact bounce.



(a) Ckt. diagram



(b) A switch debouncer waveforms

(c) **Difference between Latch & Flip-flop :**

i) **Triggering -**

- Latches are level triggered (ie, changes o/p based on the sustained level of the input)
- FF's are edge triggered (ie, changes o/p only on clock signal transitions.

ii) **Clock signal -**

- Latches typically does not require clock signal
- FF's require clock signal

iii) **Operation -**

- Latches are asynchronous (ie, changes o/p immediately when i/p changes)
- FF's are synchronous (ie, changes o/p in synch with clock signal)

iv) Out Change -

- In latches output changes as long as the input is active and the latch is enabled
- In FF's output changes only on the specified clock edge.

v) Application -

- Latches are used in asynchronous sequential ckts where immediate o/p changes are needed.
- FF's are used in synchronous sequential ckts. to control timing and data transfer.

Q.6) a) Characteristics Equations of FF's - SR, JK & T

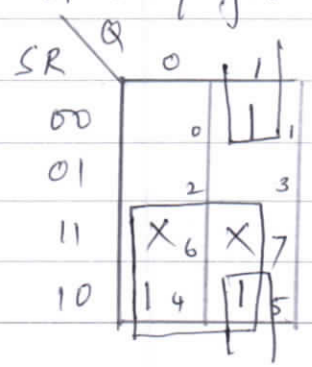
The algebraic description of the next-state table of a FF is called the characteristic eqn.

- Since the next state (Q^+) of FF's depend upon their present state (Q) and present i/p's (SR/JK/T), the functional tables can be rewritten with next state as the o/p and present state as one of the i/p's.

(i) SR FF -

	S	R	Q	Q^+
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	X
7	1	1	1	X

K-Map for Q^+ -

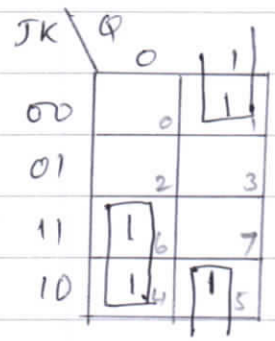


$Q^+ = S + \bar{R}Q$

(ii) JK FF -

	I/P			O/P
	J	K	Q	Q^+
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

K-Map for next state Q^+ →

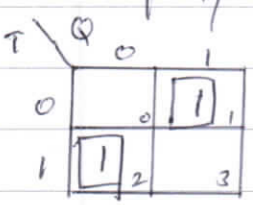


$\therefore Q^+ = J\bar{Q} + \bar{K}Q$

(iii) T FF -

	I/P		O/P
	T	Q	Q^+
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

K-Map for next state, Q^+ →



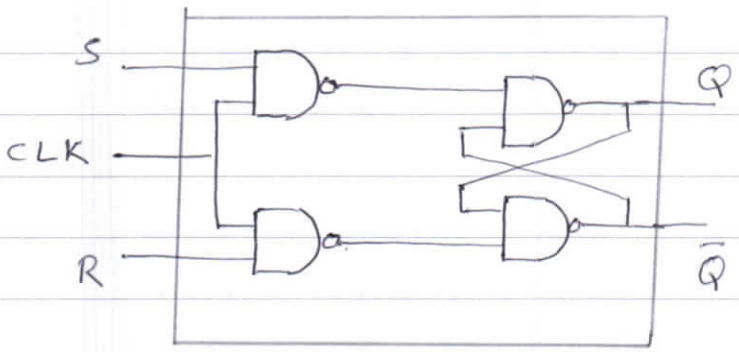
$\therefore Q^+ = \bar{T}Q + T\bar{Q}$

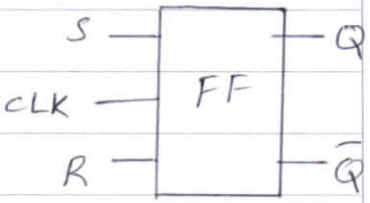
$= T \oplus Q$

Characteristic eqn table for FF's -

	FF Type	Characteristic eqn
1	SR	$Q^+ = S + \bar{R}Q$
2	JK	$Q^+ = J\bar{Q} + \bar{K}Q$
3	T	$Q^+ = T \oplus Q$

(b) **Working of SR Flip-flop -**





Truth Table

S	R	CLK	Q^+	$\bar{Q}^+$	State
0	0	1	Q	$\bar{Q}$	NC
0	1	1	0	1	Set
1	0	1	1	0	Reset
1	1	1	Racing		Racing
x	x	0	Q	$\bar{Q}$	NC

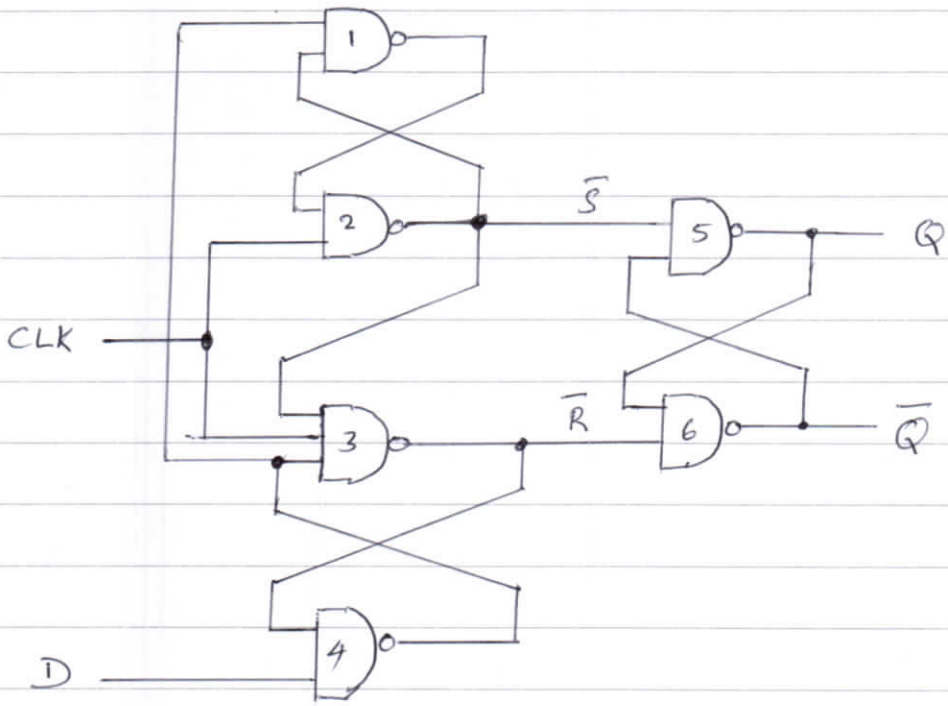
→ Setting (S=1, R=0) : When the Set i/p is high and Reset input (R) is low, the FF's o/p is set to 1

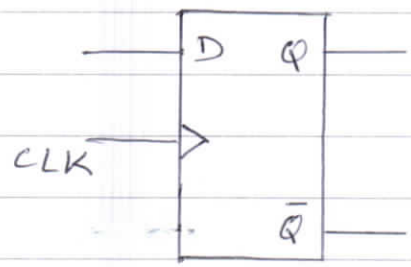
→ Resetting (S=0, R=1) : When the Reset i/p (R) is high and the Set i/p (S) is low, the FF's o/p is reset to zero

→ No change (S=0, R=0) : When both inputs (S & R) are low, the flip-flop retains its current state.

→ Invalid state (S=1, R=1) : A situation where both inputs are high is typically considered invalid or undefined, as it leads to unstable state where both outputs Q & $\bar{Q}$ are high. This is also called as race around condition.

(c) **Working of positive edge triggered D-FF -**





Truth Table

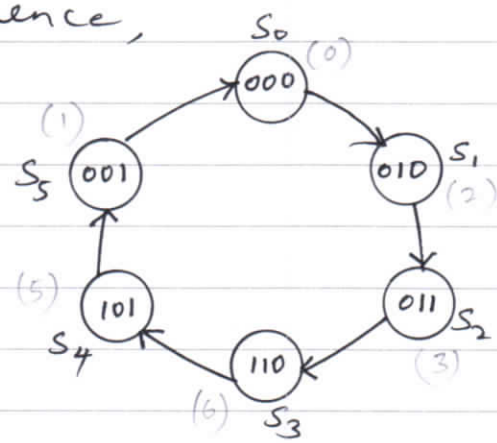
I/P's		O/P's	
D	CLK	Q^+	$\bar{Q}$
0	↑	0	1
1	↑	1	0
×	0	Q	$\bar{Q}$
×	↑	Q	$\bar{Q}$

As shown in the logic diagram, NAND gates 5 & 6 serve as an $\bar{S}\bar{R}$ latch. Thus as long as $\bar{S} = \bar{R} = 1$, the state of the latch cannot change; while whenever $\bar{S}$ or $\bar{R}$ is '0', but not both, the latch sets or resets resply.

Only upon the occurance of the positive edge of the clock signal does the flip-flop respond to the value of the D input. Once the new O/p state is established, changes in the D input while C=1 are ineffectual. When the clock signal returns to '0', both $\bar{S}$ & $\bar{R}$ become 1, & the $\bar{S}\bar{R}$ latch retains the state.

Module - 4

Q.7) (a) Given sequence,



To implement using JK FF, first write the excitation table for JK FF; $Q^+ = J\bar{Q} + \bar{K}Q$

PS	NS	FF I/P's	
Q	Q^+	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Next, write state transition table including PS, NS and FF I/P's for JK FF.

	PS			NS			FF Inputs			
	Q_3	Q_2	Q_1	Q_3^+	Q_2^+	Q_1^+	$J_3 K_3$	$J_2 K_2$	$J_1 K_1$	
0	0	0	0	0	1	0	0 X	1 X	0 X	2
2	0	1	0	0	1	1	0 X	X 0	1 X	3
3	0	1	1	1	1	0	1 X	X 0	X 1	6
6	1	1	0	1	0	1	X 0	X 1	1 X	5
5	1	0	1	0	0	1	X 1	0 X	X 0	1
1	0	0	1	0	0	0	0 X	0 X	X 1	0

Generating eqns for FF i/p's using K-Maps –

for J_3 –

Q_3 \ Q_2Q_1	00	01	11	10
0	0 (0)	0 (1)	1 (3)	0 (2)
1	X (4)	X (5)	X (7)	X (6)

$\therefore J_3 = Q_2Q_1$

for K_3 –

Q_3 \ Q_2Q_1	00	01	11	10
0	X	X	X	X
1	X	1	X	0

$\therefore K_3 = \overline{Q_2}$

for J_2 –

Q_3 \ Q_2Q_1	00	01	11	10
0	1	0	X	X
1	X	0	X	X

$\therefore J_2 = \overline{Q_1}$

for K_2 –

Q_3 \ Q_2Q_1	00	01	11	10
0	X	X	0	0
1	X	X	X	1

$\therefore K_2 = Q_3$

for J_1 –

Q_3 \ Q_2Q_1	00	01	11	10
0	0	X	X	1
1	X	X	X	1

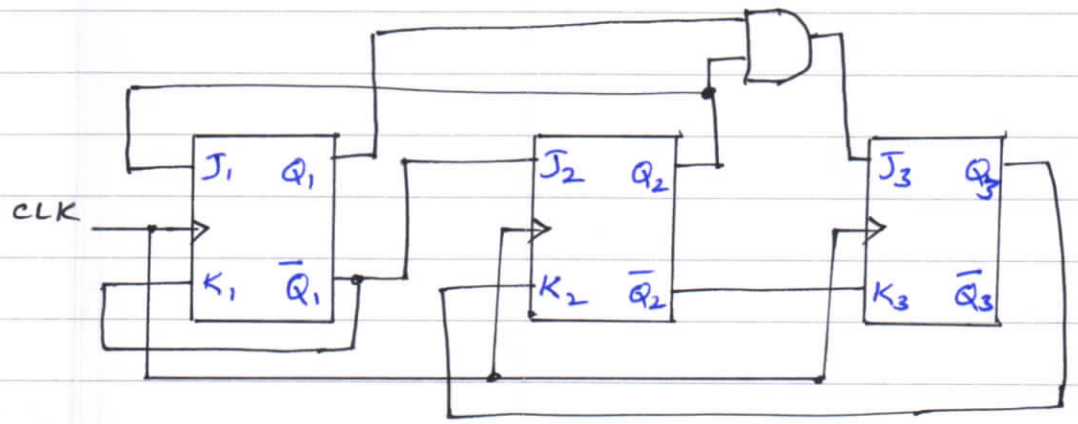
$\therefore J_1 = Q_2$

for K_1 –

Q_3 \ Q_2Q_1	00	01	11	10
0	X	1	1	X
1	X	0	X	X

$\therefore K_1 = \overline{Q_1}$

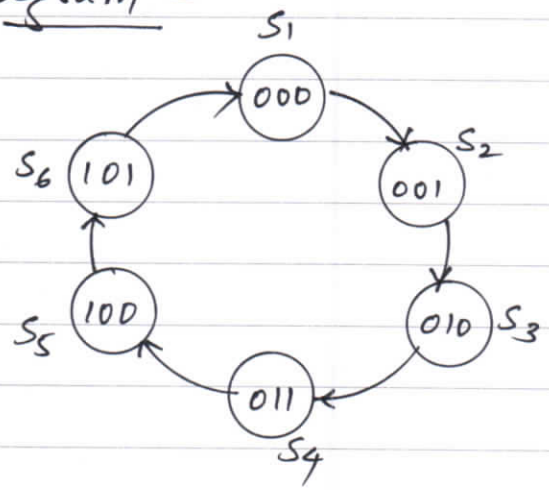
∴ Logic diagram using three JK FF's -



(b) Synchronous mod-6 counter using clocked T-FF

Synchronous mod-6 counter counts the sequence from 000 to 101 and then resets.

State diagram -



Excitation table for T-FF :

$Q^+ = T \oplus Q$

	PS	NS	FF i/p
	Q	Q^+	T
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

Generating eqns for FF i/p's using K-map and state transition table using PS, NS.

	PS			NS			FF i/P's		
	Q_3	Q_2	Q_1	Q_3^+	Q_2^+	Q_1^+	T_3	T_2	T_1
0	0	0	0	0	0	1	0	0	1
1	0	0	1	0	1	0	0	1	1
2	0	1	0	0	1	1	0	1	1
3	0	1	1	1	0	0	1	1	1
4	1	0	0	1	0	1	0	0	1
5	1	0	1	0	0	0	1	0	1

for T_3

Q_3 \ Q_2Q_1	00	01	11	10
0	0 (0)	0 (1)	1 (3)	0 (2)
1	0 (4)	1 (5)	X (7)	X (6)

$\therefore T_3 = Q_3Q_1 + Q_2Q_1$

$= Q_1(Q_3 + Q_2)$

for T_2

Q_3 \ Q_2Q_1	00	01	11	10
0	0 (0)	1 (1)	1 (3)	1 (2)
1	0 (4)	0 (5)	X (7)	X (6)

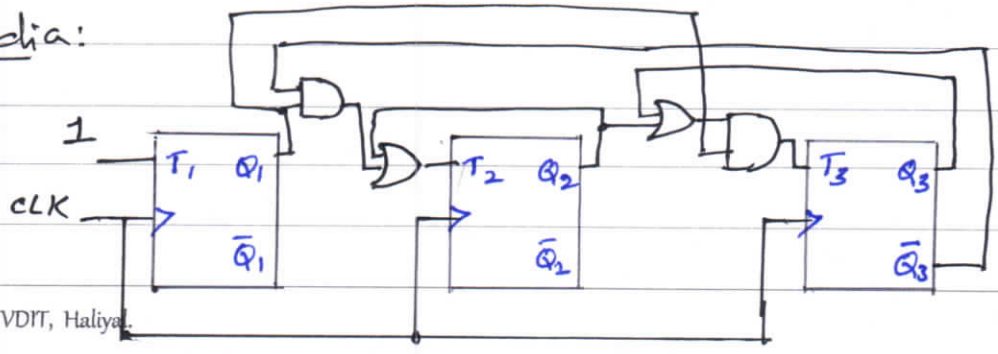
$\therefore T_2 = \bar{Q}_3 Q_1 + Q_2$

for T_1

Q_3 \ Q_2Q_1	00	01	11	10
0	1 (0)	1 (1)	1 (3)	1 (2)
1	1 (4)	1 (5)	X (7)	X (6)

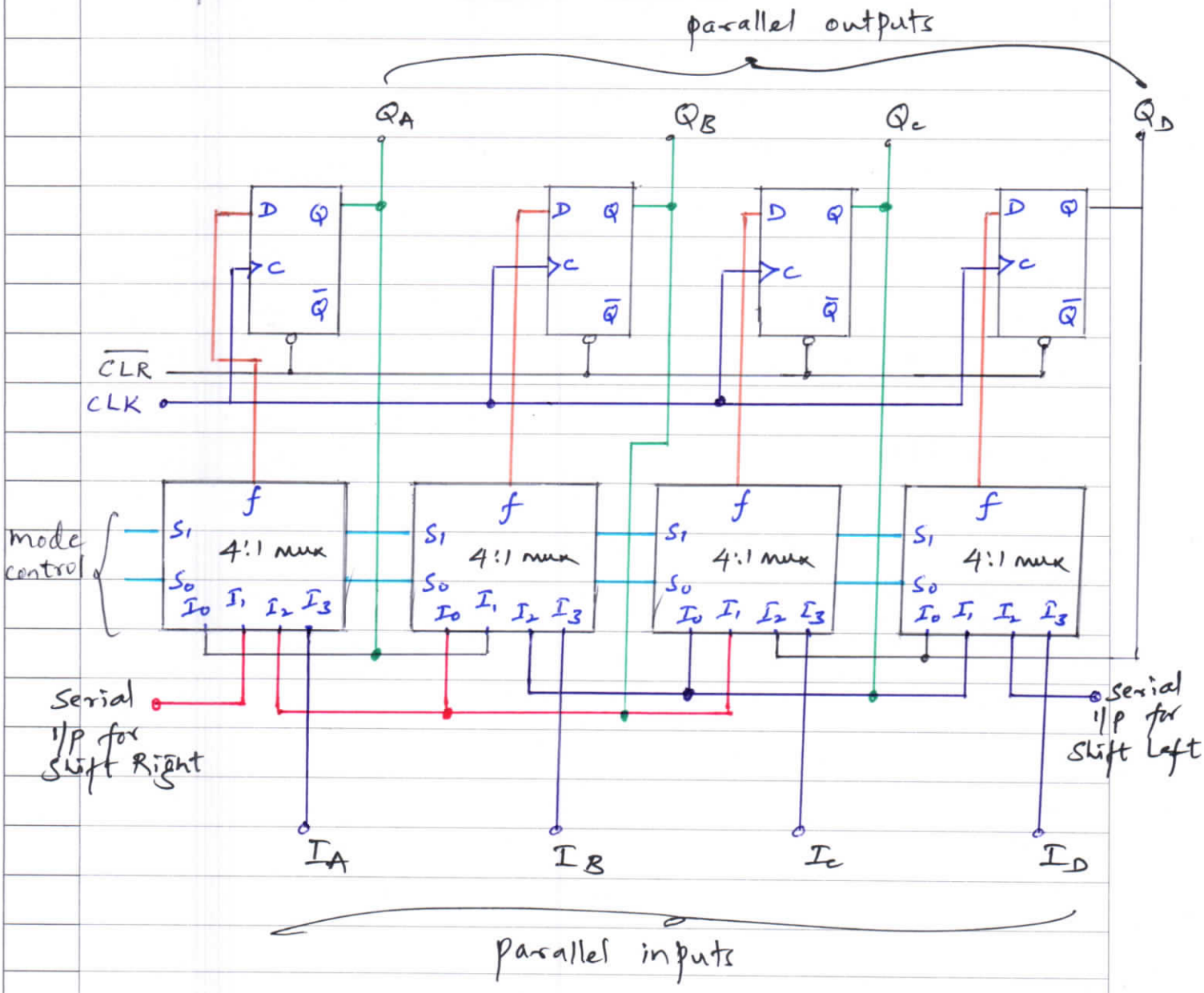
$\therefore T_1 = 1$

Logic dia:

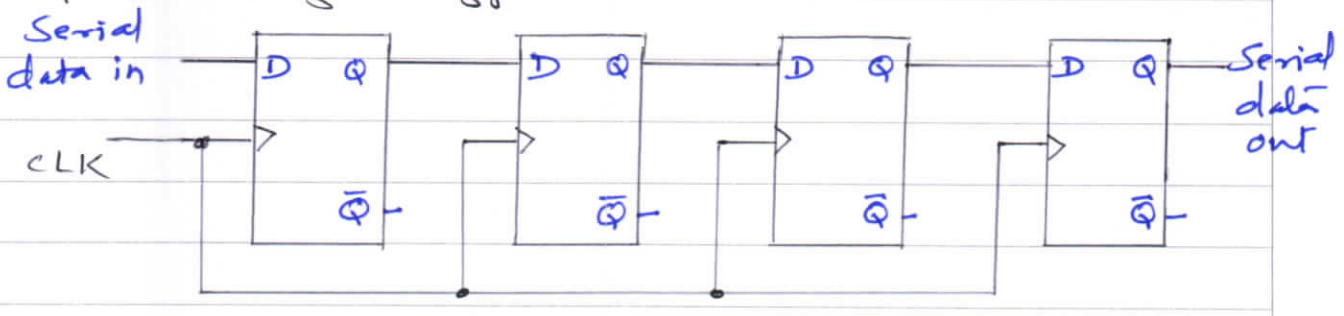


Q.8) (a) 4-bit register using positive edge triggered D-FF to operate as per below truth table.

Select lines A_1	A_0	Register operation
0	0	Hold
0	1	shift right
1	0	shift left
1	1	Parallel load



(b) 4-bit SISO uni-directional shift register using positive edge triggered D-FF —



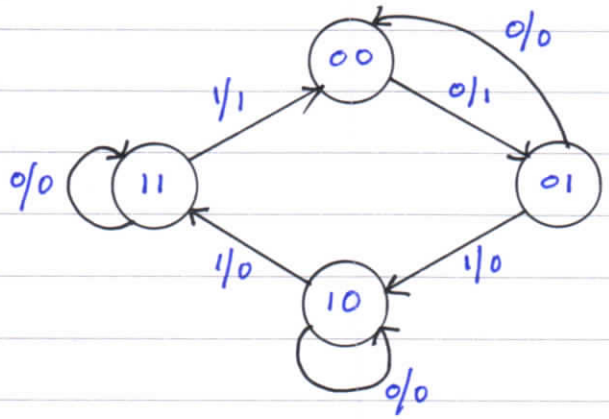
Serial-in Serial-out (SISO) shift register is a sequential logic circuit that allows ~~that~~ data to be shifted in and out one bit at a time in a serial manner. It consists of a cascade of a flip-flops connected in series, forming a chain. The input data is applied to the first flip-flop in the chain, and as the clock pulses, the data propagates thro' the flip-flops, ultimately appearing at the output. The cascaded flip-flops operate synchronously with one another, as they all receive the same clock signal. The synchronous nature of the FF's ensures that the shifting of data occurs in a co-ordinated manner. When the clock rises, the input data is sampled & stored in the first flip-flop. On subsequent clock pulses, the stored data propagates thro' the FF's, moving from one FF to the next.

Each D-FF in the ckt has a data (D) input, a clock (CLK) input, and an o/p (Q). The 'D' input represents the data to be loaded in to the FF, while the clk input is connected to the common clock signal. The o/p 'Q' of each FF is connected to the D i/p of next FF, forming a cascade.

Module - 5

Q.9) a)

Given state diagram,



Excitation Table of T-FF.

PS Q	NS Q^+	FF i/p T
0	0	0
0	1	1
1	0	1
1	1	0

Next generate the state table.

	PS Q_2	PS Q_1	i/p 'x'	NS Q_2^+	NS Q_1^+	o/p 'y'	FF i/p's T_2	FF i/p's T_1
0	0	0	0	0	1	1	0	1
1	0	0	–	—		—	—	
2	0	1	0	0	0	0	0	1
3	0	1	1	1	0	0	1	1
4	1	0	0	1	0	0	0	0
5	1	0	1	1	1	0	0	1
6	1	1	0	1	1	0	0	0
7	1	1	1	0	0	1	1	1

∴ Eqns for FF i/p's and o/p 'Y' are,

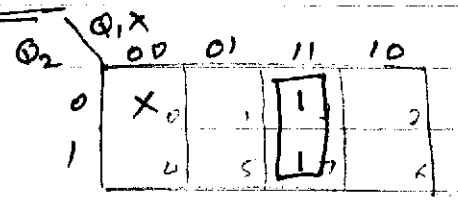
$T_2 = \Sigma m(3,7) + dc(1)$

$T_1 = \Sigma m(0,2,3,7) + dc(1)$

$Y = \Sigma m(0,7) + dc(1)$

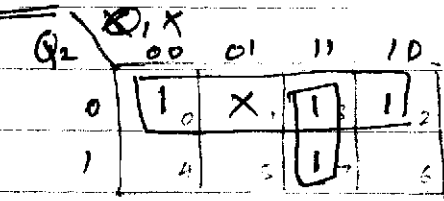
Eqns for FF i/ps & o/p 'Y' are using K-Map -

for T_2 -



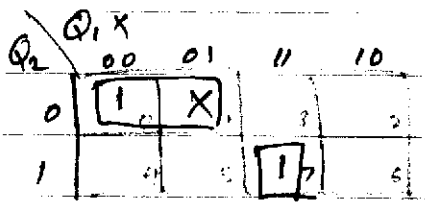
∴ $T_2 = Q_1 X$

for T_1 -



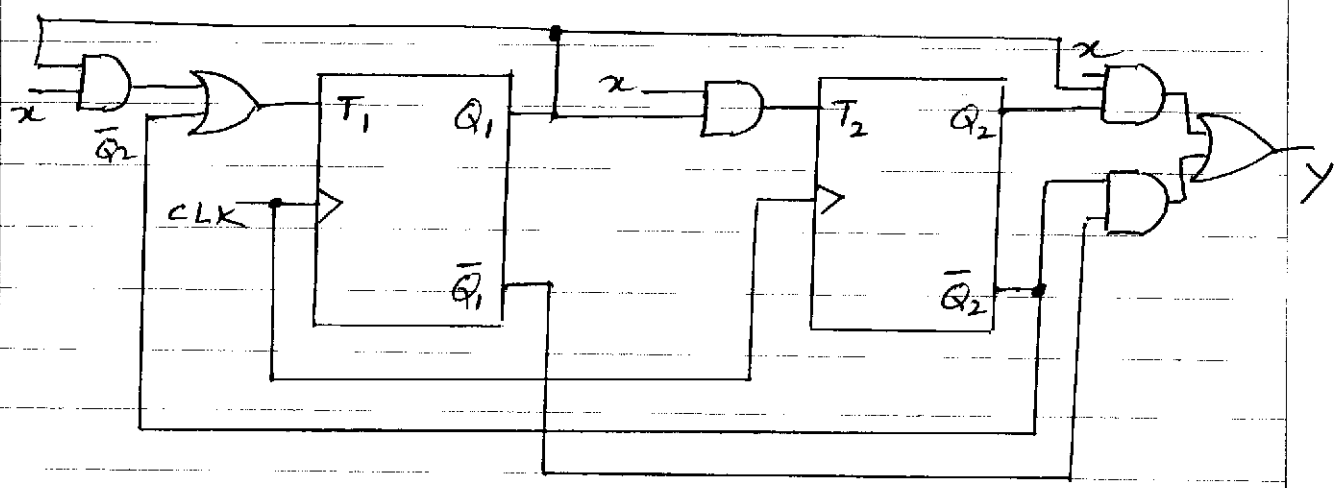
∴ $T_1 = \bar{Q}_2 + Q_1 X$

for Y -



∴ $Y = \bar{Q}_2 \bar{Q}_1 + Q_2 Q_1 X$

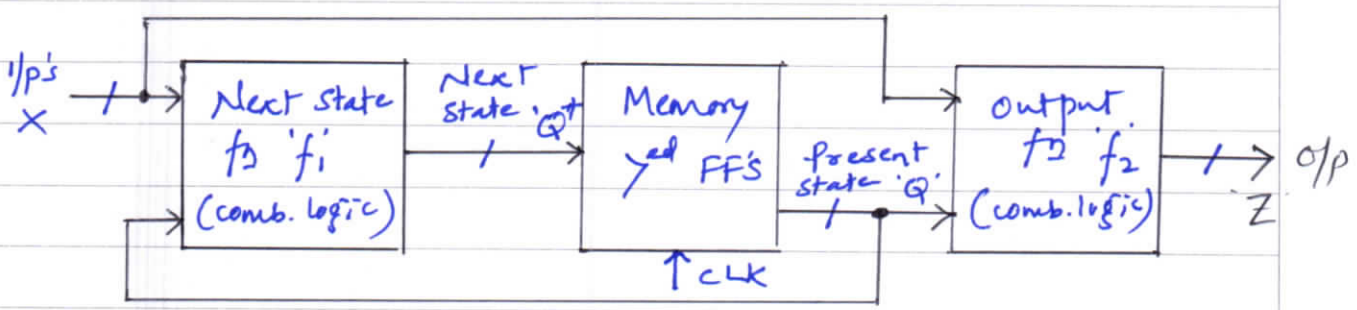
∴ Sequential circuit using T- FF –



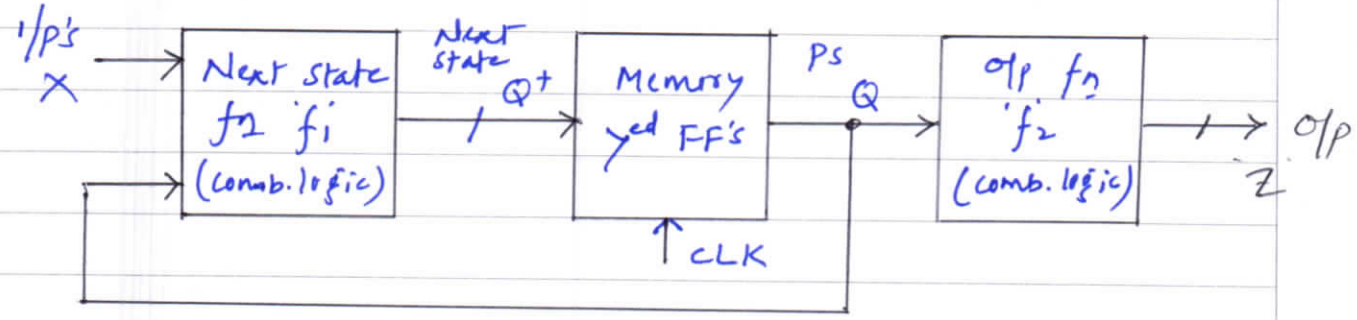
(b) **Melay & Moore Model –**

Melay & Moore models are two types of finite state machines (FSM) used to model sequential logic. The key difference lies in how they generate output; Melay machines o/p depends on both the current state and the input, while Moore machine's o/p depends solely on the current state.

Melay Model –



Moore model –



Let

X → denotes the collective input signals to the n/w

Q → denotes the collective present states of the FF's.

Q^+ → denotes the collective next states of the FF's.

Z → denotes the collective output signals of the n/w

Functionally, $Q^+ = f_1(x, Q)$ —— (1)

and $z = f_2(x, Q)$ —— (2)

Eqns (1) & (2) can be represented in the form of network called "Melay Machine"

A variation to the Melay model occurs when the outputs are only a function of present state and not the external inputs.

ie., $z = f_2(Q)$ —— (3)

Eqn (3) which can be represented in the form of network called "Moore Machine"

Q.10) (a)

(i) Read / Write Memories –

Read / Write memory, often referred to as Random Access Memory (RAM), is a type of computer memory that allows both data retrieval (reading) and data storage (writing). It is used to temporarily hold information that the computer is actively using.

Key characteristics of Read / write memory –

i) Volatile – RAM is volatile ie, data stored in RAM is lost when the power supply is turned off.

ii) Fast access – RAM provide very quick access to data, making it ideal for storing and retrieving information that the processor needs frequently.

iii) Direct access - RAM allows the processor to access any memory location directly without having to sequentially search thro' the entire memory

iv) Read / Write capabilities - The primary feature of RAM is its ability to both read data from and write data to any memory location.

ii) Programmable ROM (PROM) -

PROM stands for Propcrammable Read only Memory is a type of non-volatile memory that can be prog-rammed once by the user. Unlike traditional ROM, which has data permanently written during manufa-cturing, PROM can be programmed after it is manufactured, but can not be erased or reprogrammed later. This makes PROM suitable for applications where data needs to be stored permanently but can be customized after purchase.

Key characteristics -

i) Non-volatile : Data is retained even when power is off.

ii) One-time programming - Data is written once & can not be erased or rewritten

iii) Cost effective - Often more cost effective than EPROM or EEPROM for single use applications.

(b) i) Flash Memory -

Flash memory is a type of non-volatile storage that retains data even when power is off. It is widely used in devices like USB drives, SSD's, and memory cards, offering advantages like fast access times and low power consumption.

Key features of flash memory -

i) Non-volatile - Data is retained even without power.
ii) Electrically erasable & reprogrammable - Data can be erased & re-written
iii) Floating gate transistors - These transistors are the core of flash memory, trapping electrons to represent data.

ii) EPROM -

EPROM short for "Erasable Programmable Read-Only Memory", is a type of non-volatile memory that can be reprogrammed by exposing to UV light, allowing the stored data to be erased & reprogrammed with new data. It's a reusable chip.

Key features of EPROM -

i) Non-volatile - EPROM retains data even when power is off.
ii) Erasable - The stored data can be erased by exposing the EPROM chip to UV light.
iii) Programmable - EPROM's can be programmed with new data after being erased
iv) Reusable - Due to ability to be erased & reprogrammed, EPROM's can be reused for different applications.

HEAD
Dept. of Electrical & Electronics Engg.
KLS's V. D. Institute of Technology
HALIYAL-581 329.