IAT 1 QP :

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

Internal Assessment Test 1 – Dec. 2021

Sub:	DIGITAL SYSTEM DESIGN				
Date :	17-12-2021	Duration :	90 Minutes	Max Marks:	50

Answer any FIVE FULL Questions

1 (a)	Define a) literal b) canonical SOP c) PrimeImplicants& Essential Prime Implicants d) Maxterm e) Combi examples for each.
1 (b)	Design a combinational circuit to output 2s complement of BCD number. Show the circuit can be implen gates?
2 (a)	Express the following functions into canonical form: i) f = ab'+bc ii) f=(a+b')(b'+c)
2 (b)	Transform each of the following canonical expression into other canonical form in decimal notation: $f(x,y,z) = \Sigma m(0,1,3,4,6,7)$ $f(w,x,y,z) = \pi(0,1,2,3,4,6,12)$
3(a)	Identify all PIs and EPIs of the following using kmap.

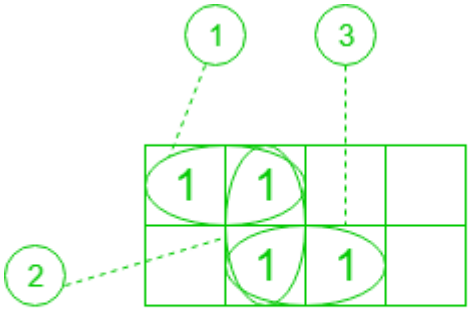
	f(a,b,c,d)= Σm(0,2,4,5,6,7,8,10,13,15)
3(b)	Simplify the following expression using Kmap. Implement the simplified expression using NA =Σm(0,1,2,5,6,7,8,9,10,13,14,15).
4	Simplify the following using QM method. f(a,b,c,d) = Σm(7,9,12,13,14,15)+dc(4,11)

5	Explain the working principle of 4 bit Carry Look Ahead adder with relevant equations & block diagram.	[10]	CO2	L2
6	Explain a BCD to Seven Segment Display with relevant equations and block diagram.	[10]	CO2	L2
7	What is magnitude comparator? Design and implement 4-bit magnitude comparator using suitable logic gates.	[10]	CO2	L2
8	Implement f(a,b,c,d)=∑m(1,4,5,7,9,12,13) using 4:1 mux. Describe how a Demux will work as a Decoder using suitable block diagrams & truth table.	[10]	CO2	L2

SOLUTION FOR IAT 1

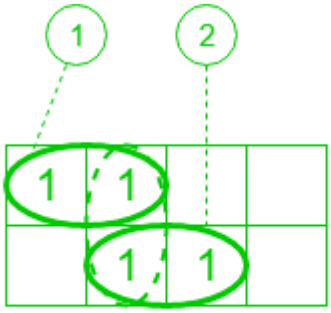
1a) Define a) literal b) canonical SOP c) PrimeImplicants& Essential Prime Implicants d) Maxterm e) Combinational circuit.Give examples for each.

A group of square or rectangle made up of bunch of adjacent minterms which is allowed by definition of K-Map are called **prime implicants(PI)** i.e. all possible groups formed in K-Map.



No. of Prime Implicants = 3

Essential Prime Implicants –
These are those subcubes(groups) which cover atleast one minterm that can't be covered by any other prime implicant. **Essential prime implicants(EPI)** are those prime implicants which always appear in final solution.



No. of Essential Prime Implicants = 2

Canonical SoP form

Canonical SoP form means Canonical Sum of Products form. In this form, each product term contains all literals. So, these product terms are nothing but the min terms. Hence, canonical SoP form is also called as **sum of min terms** form.

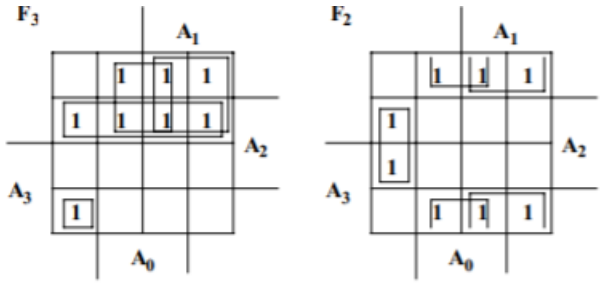
Therefore, the Boolean function of output is, f = p'qr + pq'r + pqr' + pqr. This is the **canonical SoP form** of output, f. We can also represent this function in following two notations.

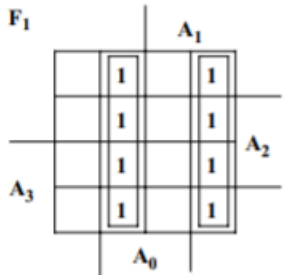
$$f = m_3 + m_5 + m_6 + m_7$$

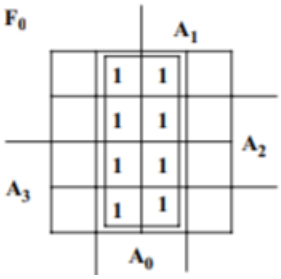
$$f = \sum m\,(3, 5, 6, 7)$$

1b) Design a combinational circuit to output 2s complement of BCD number. Show the circuit can be implemented using EX-OR gates?

$A_3A_2A_1A_0$	$F_3F_2F_1F_0$
0 0 0 0	0 0 0 0
0 0 0 1	1 1 1 1
0 0 1 0	1 1 1 0
0 0 1 1	1 1 0 1
0 1 0 0	1 1 0 0
0 1 0 1	1 0 1 1
0 1 1 0	1 0 1 0
0 1 1 1	1 0 0 1
1 0 0 0	1 0 0 0
1 0 0 1	0 1 1 1
1 0 1 0	0 1 1 0
1 0 1 1	0 1 0 1
1 1 0 0	0 1 0 0
1 1 0 1	0 0 1 1
1 1 1 0	0 0 1 0
1 1 1 1	0 0 0 1







$F_3 = \overline{A_3}A_2' + \overline{A_3}A_1 + \overline{A_3}A_0 + A_3\overline{A_2}\,\overline{A_1}\,\overline{A_0}$
$F_2 = \overline{A_2}A_0 + \overline{A_2}A_1 + A_2\overline{A_1}\,\overline{A_0}$
$F_1 = A_1\overline{A_0} + \overline{A_1}A_0$
$F_1 = A_0$

2a) Express the following functions into canonical form:

i) f = ab'+bc ii) f=(a+b')(b'+c)

1) f= a'bc+ab'c'+ab'c+abc ii) f= (a+b'+c)(a+b'+c')(a'+b'+c)

2b) Transform each of the following canonical expression into other canonical form in decimal notation:

f(x,y,z) = Σm(0,1,3,4,6,7)

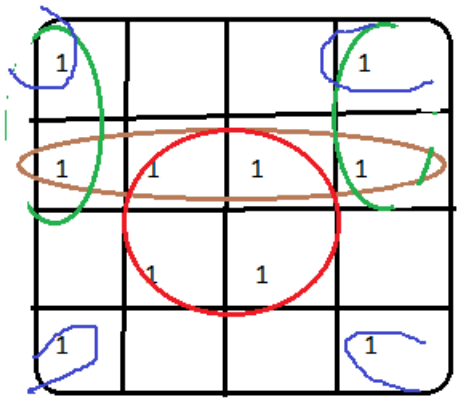
f= ∏(2,5)

f(w,x,y,z) = π(0,1,2,3,4,6,12)

f= Σ(5,7,8,9,10,11,13,14,15)

3a)Identify all PIs and EPIs of the following using kmap.

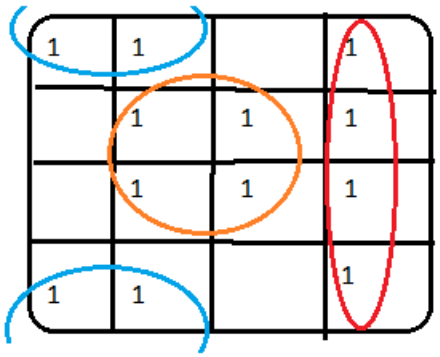
f(a,b,c,d)= Σm(0,2,4,5,6,7,8,10,13,15)



PIs = a'b + bd +a'd+b'd'

EPIs = bd+b'd'

Simplify the following expression using Kmap. Implement the simplified expression using NAND gates only. f =Σm(0,1,2,5,6,7,8,9,10,13,14,15).



f= cd'+bd+b'c'

$$\overline{\overline{cd'+bd+b'c'}}$$

$$= \overline{\overline{cd'}.\overline{bd}.\overline{b'c'}}$$

Simplify the following using QM method.

f(a,b,c,d) = Σm(7,9,12,13,14,15)+dc(4,11)

m_7 0111
m_9 1001
m_{12} 1100
m_{13} 1101
m_{14} 1110
m_{15} 1111
d_4 0100
d_{11} 1011

Step 1: Arrange the given terms in an ascending order of no. of 1's present in them

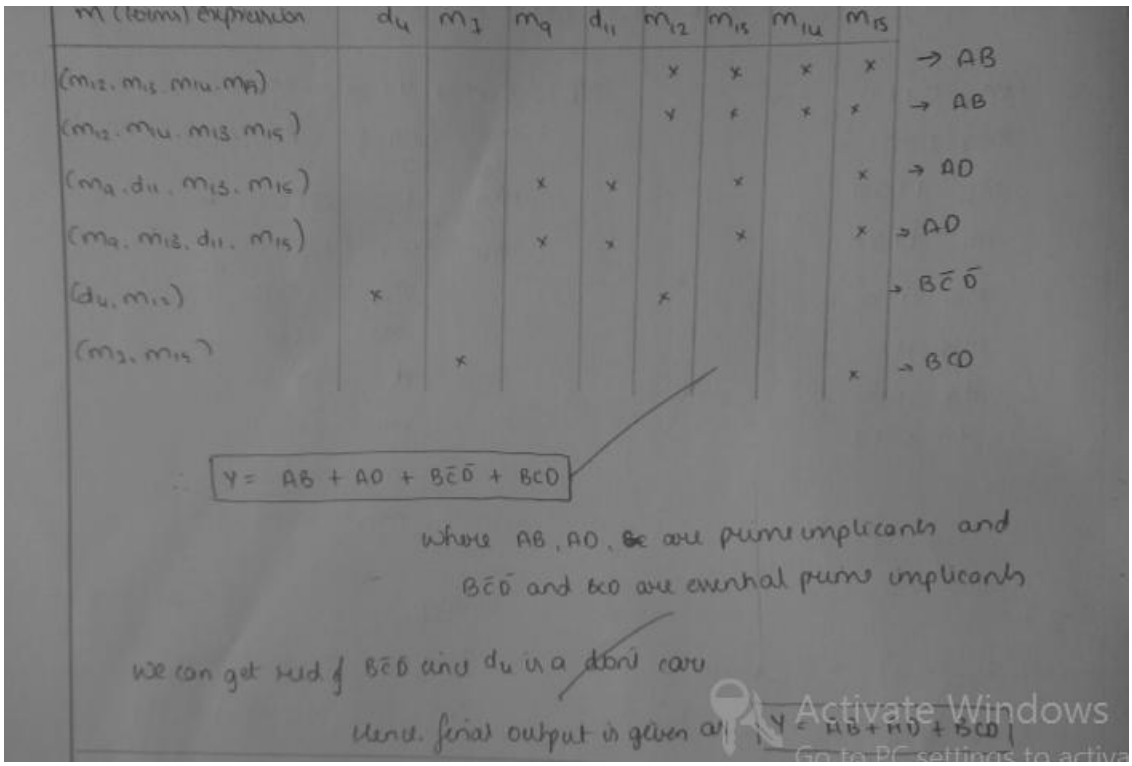
~~d_4~~			
~~m_9~~	d_4	0100	①
~~m_{11}~~	m_9	1001	②
	m_{12}	1100	
	m_7	0111	
	~~d_{11}~~	1011	③
	m_{13}	1101	
	m_{14}	1110	
	m_{15}	1111	④

Step 2:

$(d_4, m_{12}) \Rightarrow -100$
$(m_{12}, m_{14}) \Rightarrow 11{-}0$
$(m_{12}, m_{14}) \Rightarrow 11{-}0$
$(m_9, d_{11}) \Rightarrow 10{-}1$
$(m_9, m_{13}) \Rightarrow 1{-}01$
$(m_7, m_{15}) \Rightarrow -111$
$(d_{11}, m_{15}) \Rightarrow 1{-}11$
$(m_{13}, m_{15}) \Rightarrow 11{-}1$
$(m_{14}, m_{15}) \Rightarrow 111{-}$

Step 3

$(m_{12}, m_{15}, m_{14}, m_{15}) \Rightarrow 11{-}{-}$
$(m_{12}, m_{14}, m_{13}, m_{15}) \Rightarrow 11{-}{-}$
$(m_9, d_{11}, m_{13}, m_{15}) \Rightarrow 1{-}{-}1$
$(m_9, m_{13}, d_{11}, m_{15}) \Rightarrow 1{-}{-}1$
$(d_4, m_{12}) \Rightarrow -100$
$(m_7, m_{15}) \Rightarrow -111$



M (terms) Expression	d_4	m_1	m_9	d_{11}	m_{12}	m_{13}	m_{14}	m_{15}	
$(m_{12}, m_{13}, m_{14}, m_{15})$					x	x	x	x	→ AB
$(m_{12}, m_{14}, m_{13}, m_{15})$					x	x	x	x	→ AB
$(m_9, d_{11}, m_{13}, m_{15})$			x	x		x		x	→ AD
$(m_9, m_{13}, d_{11}, m_{15})$			x	x		x		x	→ AD
(d_4, m_{12})	x				x				→ $B\bar{C}\bar{D}$
(m_1, m_{15})		x						x	→ BCD

$$\therefore Y = AB + AD + B\bar{C}\bar{D} + BCD$$

where AB, AD, Be are prime implicants and

$B\bar{C}\bar{D}$ and BCD are essential prime implicants

We can get rid of $B\bar{C}\bar{D}$ since d_4 is a don't care

Hence, final output is given as $Y = AB + AD + BCD$

PROF. SUCHISMITA SENGUPTA.

CARRY LOOK AHEAD ADDER (CLA ADDER)

This adder is the fastest adder configuration.

A	B	C_{in}	Cout	
0	0	0	0	
0	0	1	0	
0	1	0	0	C_{in} propagates to Cout
0	1	1	1	
1	0	0	0	
1	0	1	1	
1	1	0	1	Carry Generated.
1	1	1	1	

$A \cdot B$

$(A \oplus B) \cdot C_{in}$

Computing Sum takes time and based on the Sum output Carry bit gets calculated.

Again, the carry bit so computed needs to be fed to the next stage for computation.

Also remember that the 1 bit Full Adder is made of logic gates. Therefore the signals take time to propagate.

This time required to propagate through the device by the signals (both output & input) is called propagation delay.

Carry Look Ahead Adders "predict" the carry bit for the computation required in the next stage.

"Predict" now means "compute".

$$\therefore Cout = \underbrace{(A \cdot B)}_{\text{Carry Generator}} + \underbrace{(A \oplus B) \cdot C_{in}}_{\text{Carry Propagate}}$$

↓ doesn't depend on C_{in}

↓ Depends on C_{in}.

Q5 **Explain the working principle of 4 bit Carry Look Ahead adder with relevant equations & block diagram**

∴ Cout becomes :-

$\boxed{Cout = G + P\,Cin}$ where G → Generate signal

& $\boxed{G = A \cdot B}$

& P → Propagate signal

& $\boxed{P = A \oplus B}$

Cin → Carry in

Now, let, Ci = Cout & C_{i-1} = Cin

∴ $\boxed{C_i = G_i + P_i \cdot C_{i-1}}$ ✓ — (1)

Let $i = 0$

⇒ $C_0 = G_0 + P_0 C_{-1}$ —— (2)

$i = 1$

$C_1 = G_1 + P_1 C_0$

⇒ $C_1 = G_1 + P_1 (G_0 + P_0 C_{-1})$

$\boxed{C_1 = G_1 + P_1 G_0 + P_1 P_0 C_{-1}}$ —— (3)

$i = 2$

$C_2 = G_2 + P_2 C_1$

⇒ $C_2 = G_2 + P_2 (G_1 + P_1 G_0 + P_1 P_0 C_{-1})$

$\boxed{C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_{-1}}$ — (4)

$i = 3$ $C_3 = G_3 + P_3 C_2$

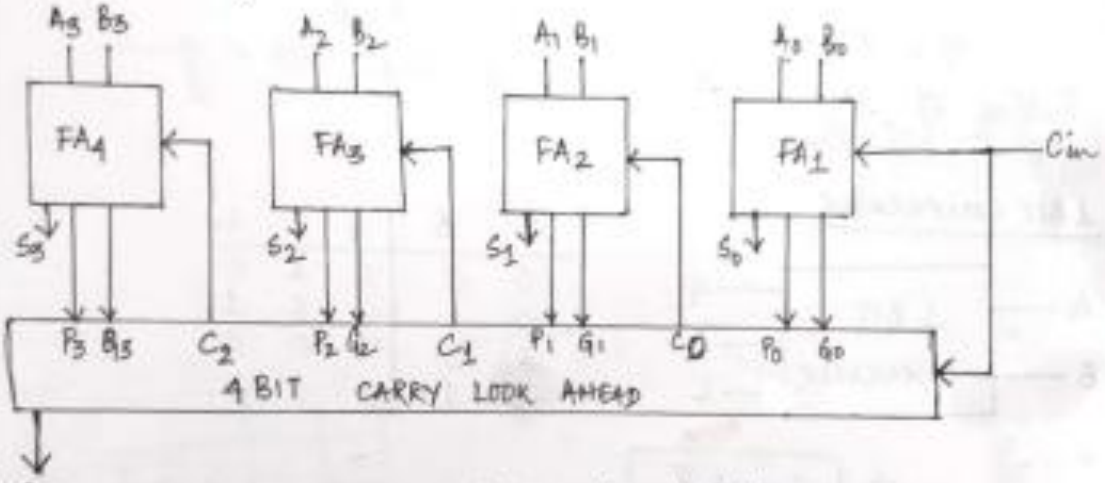
⇒ $C_3 = G_3 + P_3 (G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_{-1})$

$\boxed{C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_{-1}}$

(5)

Now when we look at eq^n ②, ③, ④, ⑤ we can observe that the next carry bit doesn't depend on the previous carry but it solely depends on the initial carry C_{-1} i.e. C_{in}.

Therefore, the delay caused by rippling effect of carry bit is eradicated.
Also note that these eq's depend on Propagate & Generate signal.
Hence Carry Look Ahead Adders are the fastest adders



4 bit Carry Look Ahead Adder

Note :-

$(S_0 - S_3)$ → Sum output
C_3 → Carry output
$(A_0 - A_3)$ → Input A
$(B_0 - B_3)$ → Input B
C_{in} → Carry input

Advantages

* Propagation delay is reduced
* Provides fastest addition logic

Disadvantages

* Complexity of the circuit increases
* CLA adders are costly

BCD Decoders

BCD to Seven Segment Display.

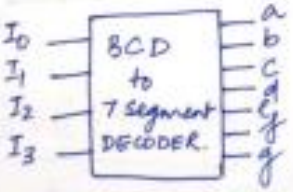
a, b, c, d, e, f, g, h → leds

h → is the decimal point.

Binary data will be converted to decimal.

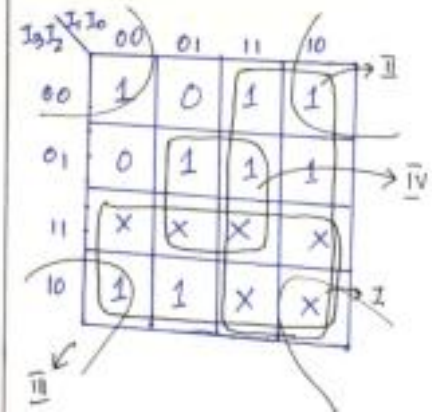
7 segment display

I_3	I_2	I_1	I_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	DONTCARE.						
⋮	⋮	⋮	⋮							
1	1	1	1							



Page 8

for a :-



$\Rightarrow a = I + \bar{II} + \bar{III} + \bar{IV}$.

$\therefore a = I_3 + I_1 + \bar{I}_2\bar{I}_0 + I_2I_0$

$\boxed{a = I_3 + I_1 + I_2 \odot I_0}$

BINARY COMPARATORS

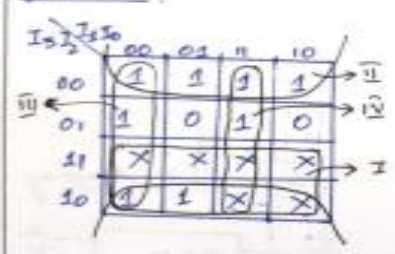
They are also called as Magnitude Comparator. It is a combinational circuit that compares two input data and indicates whether one of the inputs is greater than or lesser than or whether they are equal.

- ie. if inputs are A and B then there will be three outputs namely:- Greater than (G)
  Lesser than (L)
  Equals to (E)

$\therefore$ G is $A > B$, L is $A < B$, E is $A = B$.

Either of the outputs will be high at a given point of time.

for b :-

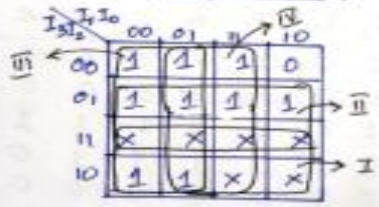


$\Rightarrow b = \bar{I} + \bar{II} + \bar{III} + \bar{IV}$

$\therefore b = I_3 + \bar{I}_2 + \bar{I}_1\bar{I}_0 + I_1 I_0$

$$b = I_3 + \bar{I}_2 + I_1 \odot I_0$$

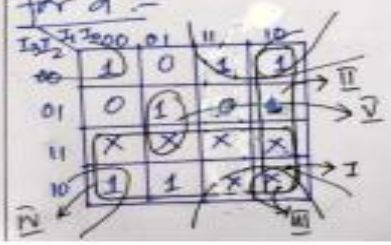
for C :-



$\Rightarrow C = \bar{I} + \bar{II} + \bar{III} + \bar{IV}$

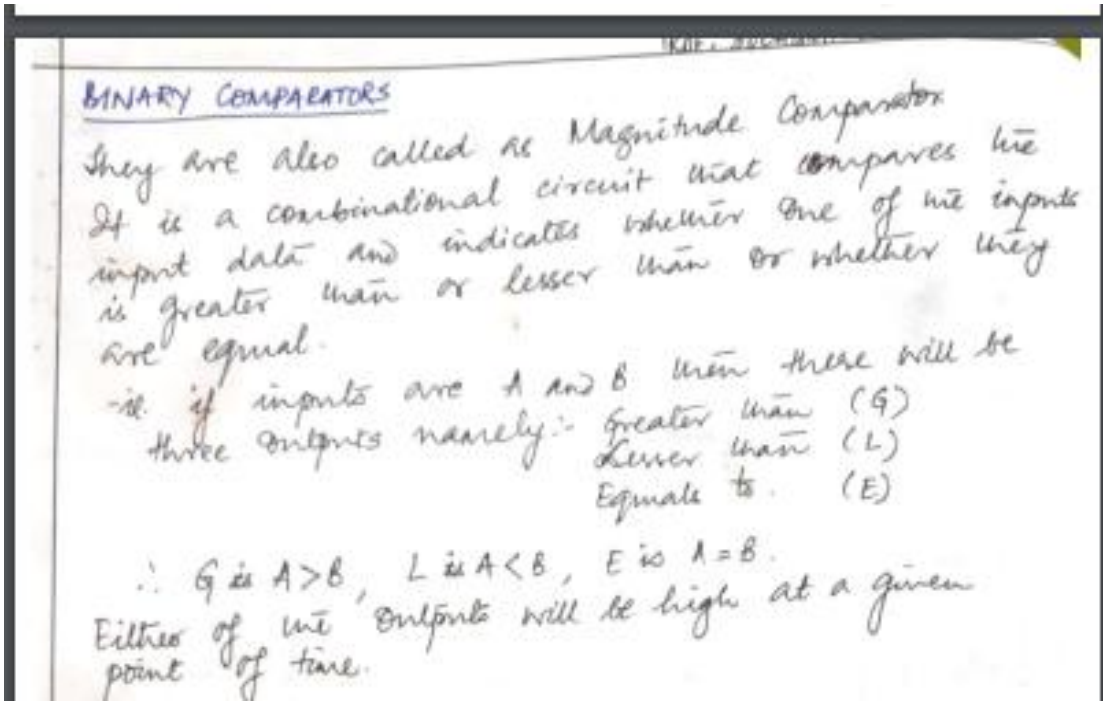
$\therefore C = I_3 + I_2 + \bar{I}_1 + I_0$

for d :-



$\Rightarrow d = \bar{I} + \bar{II} + \bar{III} + \bar{IV} + \bar{V}$

$$d = I_3 + I_1\bar{I}_0 + \bar{I}_2 I_1 + \bar{I}_2\bar{I}_0 + I_2\bar{I}_1 I_0$$



BINARY COMPARATORS

They are also called as Magnitude Comparator
It is a combinational circuit that compares two
input data and indicates whether one of the inputs
is greater than or lesser than or whether they
are equal.
-ie. if inputs are A and B then there will be
three outputs namely:- Greater than (G)
Lesser than (L)
Equals to (E)

∴ G is A>B, L is A<B, E is A=B.
Either of the outputs will be high at a given
point of time.

Q7 **What is magnitude comparator? Design and implement 4-bit magnitude comparator using suitable logic gates.**

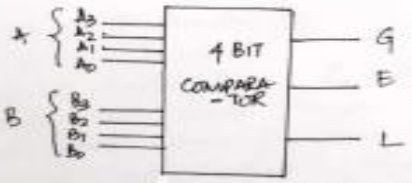
4-BIT COMPARATOR

PROF. SUCHISMITA SENGUPTA.

$A \Rightarrow (A_3 - A_0)$ $B \Rightarrow (B_3 - B_0)$

Where A_3 & B_3 are the MSB.

A_3, B_3	A_2, B_2	A_1, B_1	A_0, B_0	G ($A > B$)	E ($A = B$)	L ($A < B$)
$A_3 > B_3$	×	×	×	1	0	0
$A_3 < B_3$	×	×	×	0	0	1
$A_3 = B_3$	$A_2 > B_2$	×	×	1	0	0
$A_3 = B_3$	$A_2 < B_2$	×	×	0	0	1
$A_3 = B_3$	$A_2 = B_2$	$A_1 > B_1$	×	1	0	0
$A_3 = B_3$	$A_2 = B_2$	$A_1 < B_1$	×	0	0	1
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 > B_0$	1	0	0
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 < B_0$	0	0	1
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	0	1	0



BLOCK DIAGRAM OF 4-BIT COMPARATOR