SBG STUDY

27/8/19

Unit-2

Date:
Page:

Diff. comb. Seq

Combinational Circuit Design:

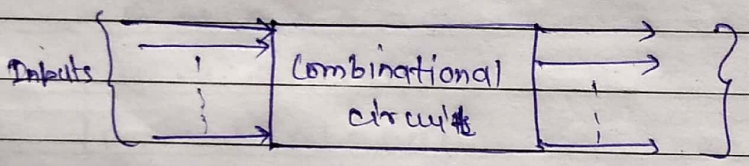
Digital Circuits:

The digital circuits are classified into two catogries ~~data~~ ~~two~~ ~~Catogories~~

i) Combinational
ii) Sequential

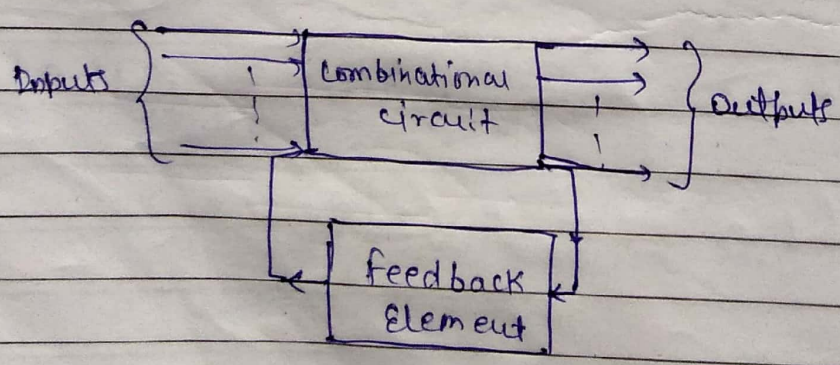
i) Combinational circuit:

These are the type of digital circuit in which output at any instant of time depends upon the combination of inputs applied ~~at~~ ~~third~~ That instance.



ii) Sequential Circuits:

These are the type of digital circuit in which output at any instant of time depends upon the Combination of inputs applied at that ~~of~~ instance as well as the Previous Output.



Dated :
Page :

Diff Difference b/w Combinational and Sequential Circuits

Combinational	Sequential
(1) these are the type of digital circuits in which output of any instance of time depends upon the combination of inputs applied that instance	1) These are the type of digital circuits in which output of any at any instance of time depends upon the combination of inputs applied at that instance as well as the previous output
(2) Combinational circuit does not required any feedback element.	2) Sequential circuits required of feedback element.
(3) The combinational circuits do not required any storing or memory element	3) Since the current output depends upon the previous output therefore a memory element is required to store the previous output.
4) these are cheaper in cost	4) these are more expensive as compared to combinational circuits
5) Less accurate.	5) More accurate.
6) Less complexicity	6) more complex.

Dated:
Page:

6 Aug

Types of Combinational Circuits:

1) Adder
2) Subtractor
3) Multiplexer
4) De-multiplexer
5) Decoder
6) Encoder
7) Parity Generator
8) Magnitude comparators.
9) Code Converters.

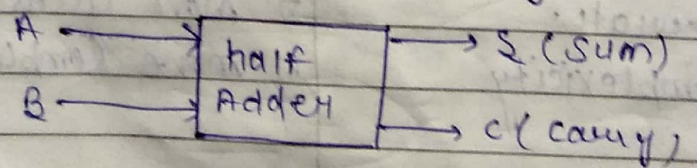
(1) Adder:

A Binary adder is a type of Combinational Circuit which is used to add the inputs applied to it. It has multiple inputs and two outputs one for Sum and another for Carry.
Adders are classified into following Categories:

i) Half Adder
ii) Full Adder
iii) Parallel Adder

i) Half Adder:

It is used to add two binary bits. It has two outputs Sum and Carry



Block diagram

Dated :
Page :

Truth table:

	I/P		O/P	
	A	B	S	C
1	0	0	0	0
2	0	1	1	0
3	1	0	1	0
4	1	1	0	1

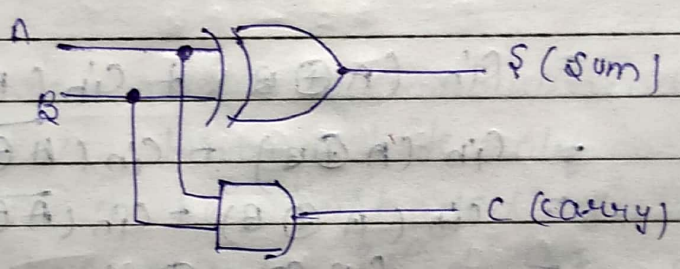
A XOR B

Boolean Expression:

$S = \bar{A}B + A\bar{B} = (A \oplus B)$

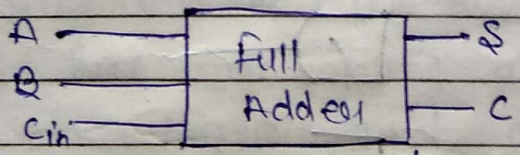
$C = AB$

Logic diagram:



iii) Full Adder:

A full Adder is used to add 3 Binary Bits. A full Adder overcomes the limitations of half adder intermediate carry in half adder. it has three inputs and two outputs



Truth table:

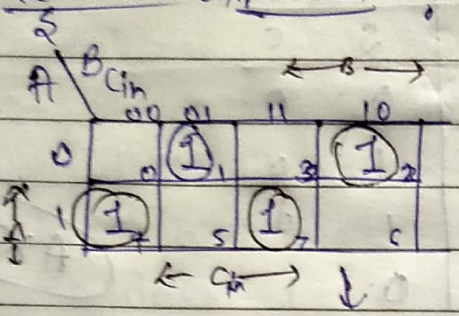
	I/P			O/P	
	A	B	Cin	S	C
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	1	0
6	1	1	0	0	1
7	1	1	1	1	1

Dated :

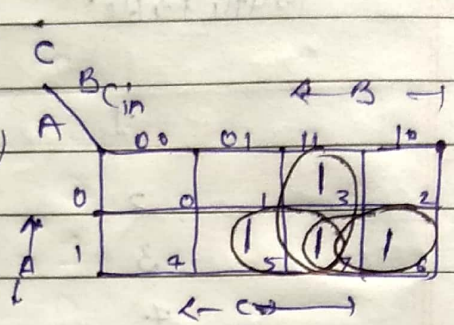
Page :

Boolean Expression:

(i)



(ii)



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

$= A\bar{B}\bar{C}_{in} + \bar{A}\bar{B}C_{in} + ABC_{in} + \bar{A}B\bar{C}_{in}$

$= \bar{C}_{in}(A\bar{B} + \bar{A}B) + C(\bar{A}\bar{B} + AB)$

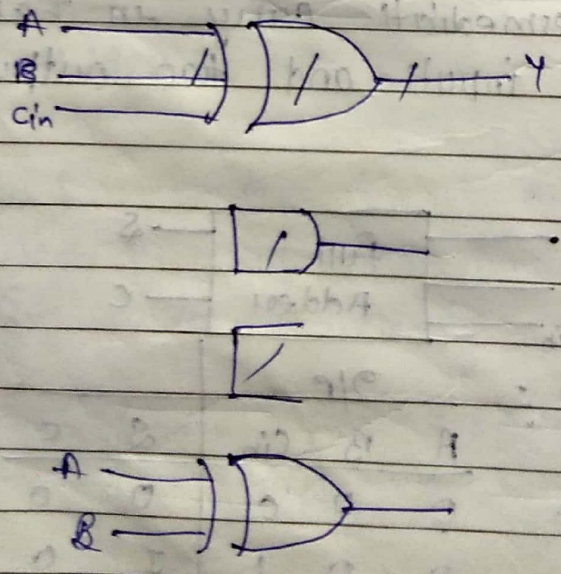
$= C_{in}(A \oplus B) + C_{in}(A \odot B)$

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

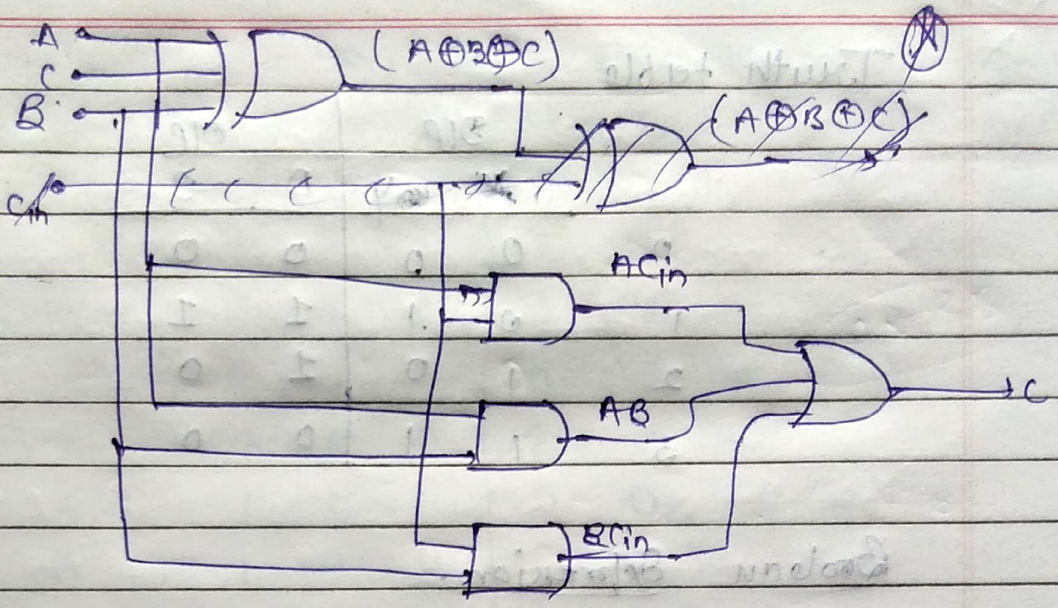
$= C_{in}(A \oplus B) + C_{in}(\overline{A \oplus B})$

$= (A \oplus B \oplus C)$

(ii) $\therefore AC_{in} + AB + BC_{in}$



Dated :
Page :



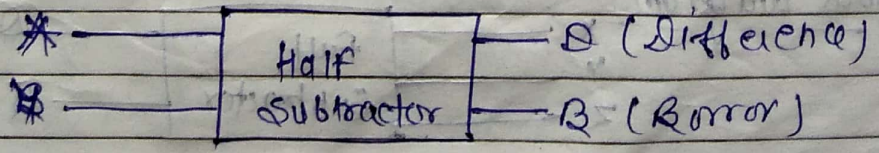
(2) Subtractor:

A subtracter is a type of combinational circuit used to subtract binary bits depending upon the no. of bits to be subtracted. The ~~Sub~~ Subtractor are classified into following Catogries

i) Half Subtractor
ii) Full Subtractor
iii) Parallel Subtractor

(i) Half Subtractor:

A half subtractor is used to subtract two binary bits. It has two inputs and two output ~~and~~ one for borrow and another for difference



Dated :
Page :

Truth table

	I/P		O/P	
	X	Y	D	B
0	0	0	0	0
1	0	1	1	1
2	1	0	1	0
3	1	1	0	0

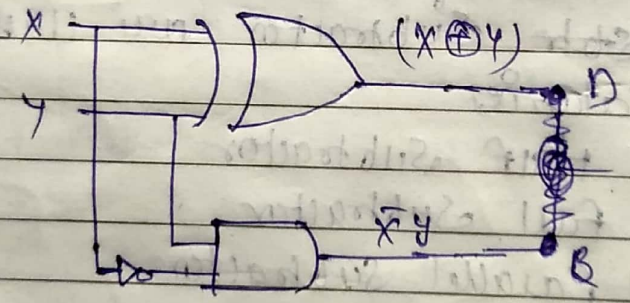
Boolean Expression.

$D = \bar{A}B + A\bar{B}$ $\qquad B = \bar{x}y$

$= \bar{x}y + x\bar{y}$

$= (x \oplus y)$

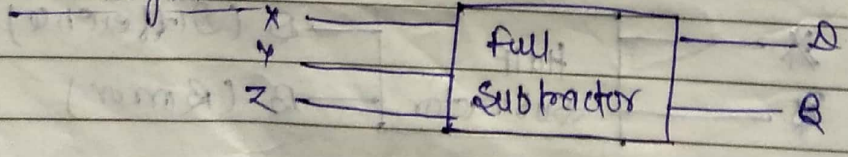
Logic



(ii) full subtractor :-

It is used to subtract to three bits (Binary) & generate two o/p difference and Borrow.

Block diagram :



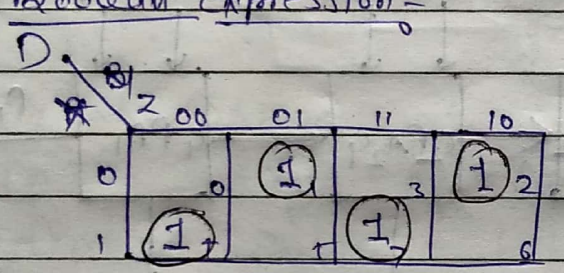
Dated :

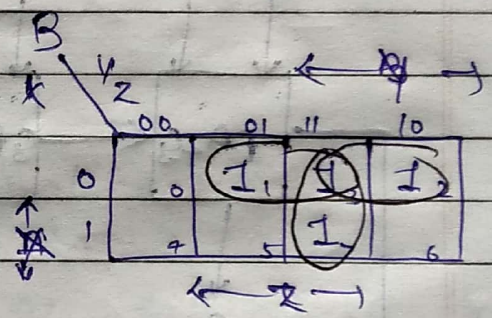
Page :

Truth table.

	I/P			O/P	
	X	Y	Z	D	B
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	1	1
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	0
6	1	1	0	0	0
7	1	1	1	1	1

Boolean Expression :

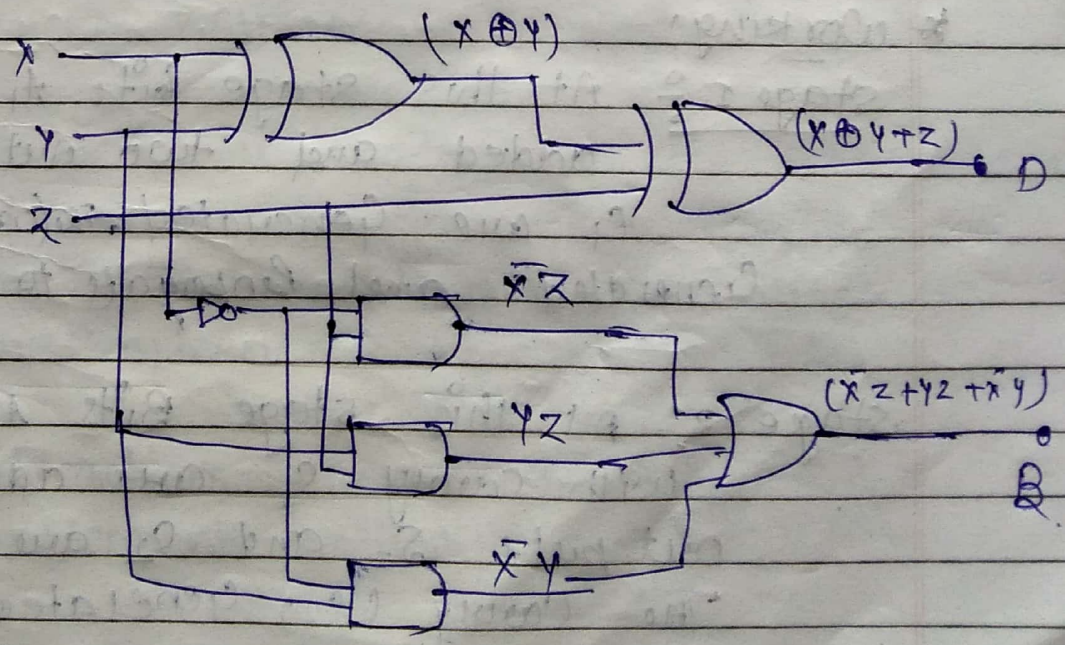




$D = X \oplus Y \oplus Z$

$B = \bar{X}Z + YZ + \bar{Y}Y$

Logic :



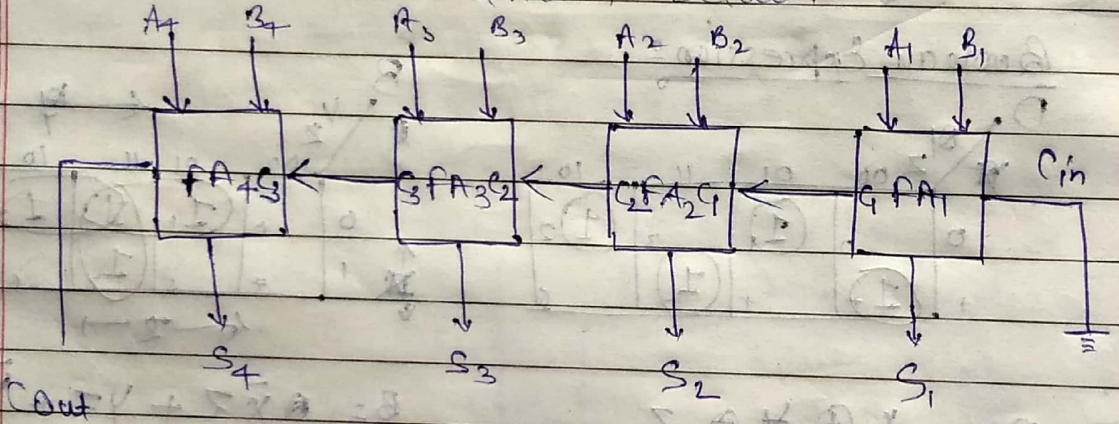
Dated :
Page :

5 marks

Parallel Adder :

It is a type of combinational circuit that parally adds Binary bits. It is constructed by connecting a no. of full adders in parallel. The no. of full adder depends upon the no. of bits to be added

Let the nos to be added are A ($A_4, A_3 A_2 A_1$) and B ($B_4 B_3 B_2 B_1$) then the Parallel adder fully addition of A and B can be constructed as shown below.



Working :

Stage 1 : At this stage bits A_1 and B_1 are added and two outputs S_1 and C_1 are Generated. The (with carry C_1) Generated and Propogate to the Next stage

Stage 2 : At this stage Bits A_2 and B_2 along with Carry C_1 are added and two out puts S_2 and C_2 are generated The Carry (C_2) Generated at this stage Propogates to the Next stage

Dated :
Page :

Stage 3: At this stage Bits A_3 and B_3 along with carry C_2 are added and two output S_3 and C_3 are generated. The carry (C_3) Generated at this stage propagates to the next stage.

Stage 4: At this stage Bits A_4 and B_4 along with carry C_3 are added and two output S_4 and C_{out} are generated. The carry C_{out} Generated at this stage.

Parallel Subtractor :

It is a type of combinational circuit that parally Subtract bits. It is constructed by connecting a no. of full Subtractor in Parallel.

The no. of full subtracter depends upon the no. of bits to be added Subtracted

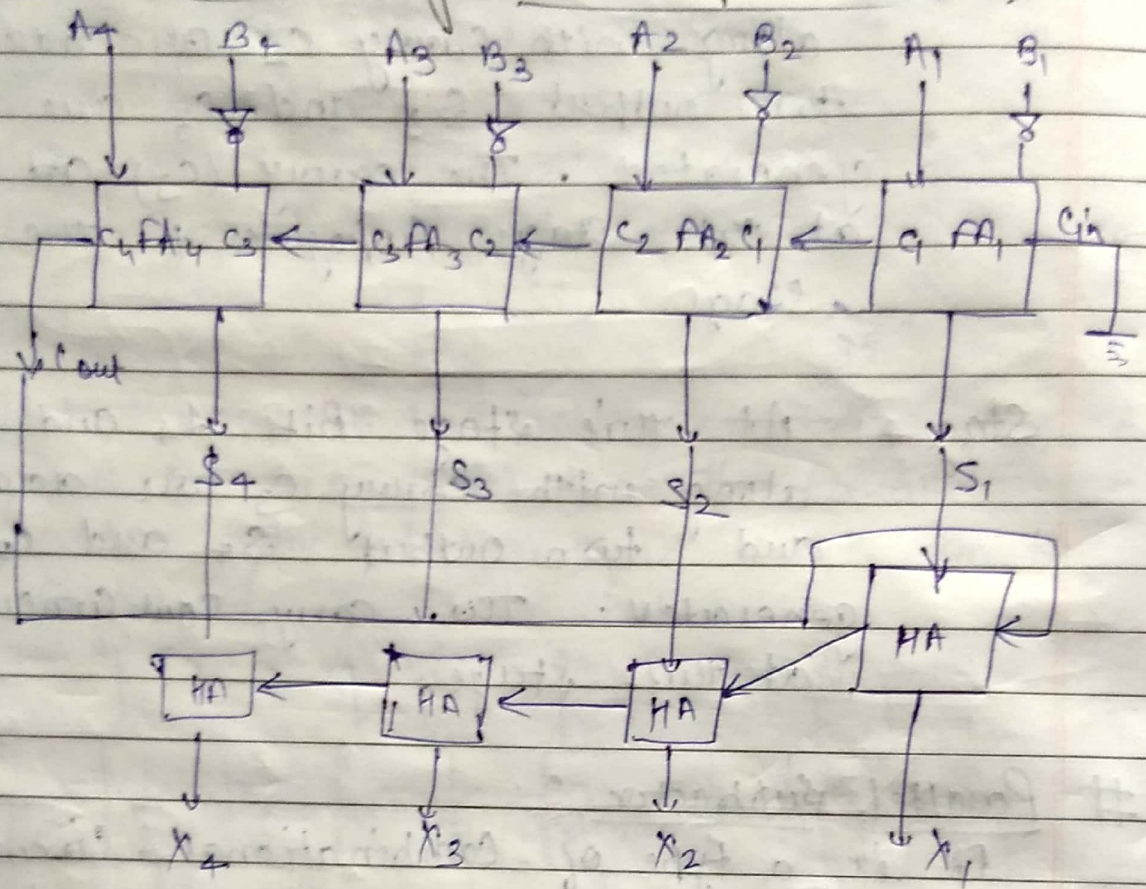
Let the no. to be subtracted are A (A_4, A_3, A_2, A_1) and B ($B_4 B_3 B_2 B_1$) Then the Parallel Subtracter full Adder subtract of A and B can be constructed as shown below

The Subtraction can be done by using the complement methods:

A + (-B)
A - B

Dated :
Page :

* Subtraction using 1's Comp. method:



* Working:

Stage 1: At this stage A_1 and $\bar{B_1}$ are added and two outputs S_1 and C_1 are generated. The carry (C_1) propogate to the Next stage

Stage: 2 At this Stage Bit A_2 and $\bar{B_2}$ along with ~~add~~ carry C_1 are added and two outputs S_2 and C_2 are generated, the carry C_2 generated at this stage propogates to the Next Stage

Stage: 3 At this Stage Bit A_3 and $\bar{B_3}$ along with carry C_2 are added and two outputs

Dated :
Page :

S_2 and C_3 are ~~added~~ and generated, the carry C_2 generated at this stage propagates to the next stage.

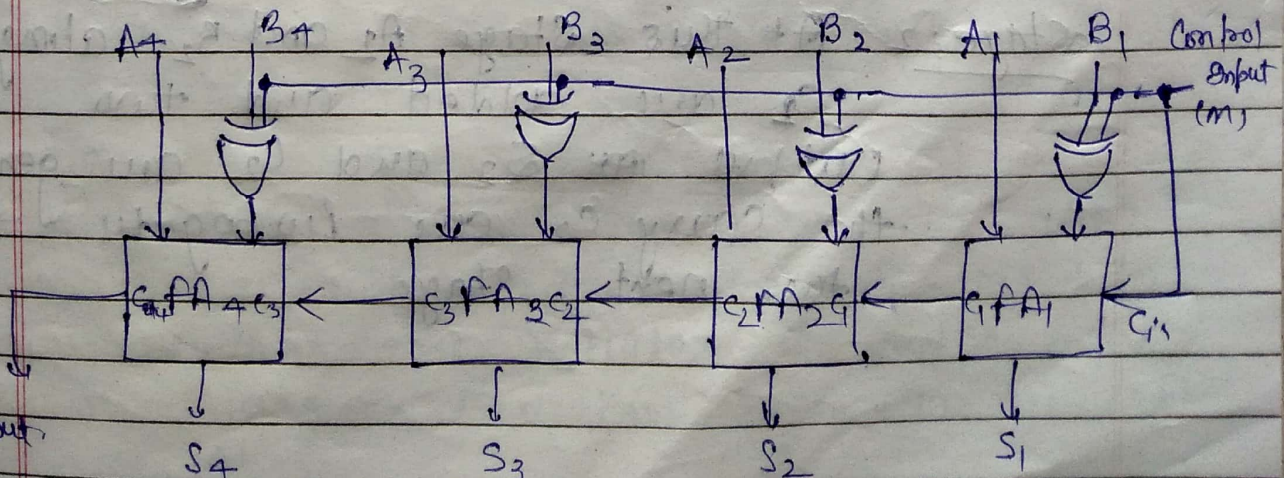
Stage - 4 :- At this stage Bits A_4 and $\bar{B}_4$ along with carry C_3 are added and two outputs S_4 and Cout are generated.

V. Imp

Parallel Adder / Subtractor Circuit

90%

It is type of circuit which is used to Perform both the Parallel adder as well as Parallel Subtractor. It is constructed by connecting a no. of full adder in Parallel. It has a control Input that Enables the circuits to be used either as a parallel Adder or as a Parallel Subtractor



XOR Gate

Dated :
Page :

Case-1 (working as a P. Adder)
M=0
B ⊕ 0
0 ⊕ 0 = 0
1 ⊕ 0 = 1

Case-2 (working as a 1's Subtractor)
M=1
B ⊕ 1
0 ⊕ 1 = 1
1 ⊕ 1 = 0

* Working :
This circuit works as a Parallel Adder at M=0 and as a Parallel Sub. at M=1.

Case-1 (when m=0)
In this case the XOR Gates do not change the value of the number B and therefore Both the numbers A and B are applied to any change

Stage-1 : At this stage A_1 and B_1 are added and two outputs S_1 and C_1 are generated. The carry C_1 are propagated to the next stage

Stage:2 At this stage A_2 and B_2 along with C_2 are added and two Outputs are S_2 and C_2 are generated the carry C_2 are propagates to the next stage

Dated :
Page :

Stage :- 3 At this stage A_3 and $\bar{B}_3$ along with carry c_2 are added. and two outputs are generated S_3 and c_3 are generated. The carry c_3 are propogate to next stage.

Stage :- 4 At this stage A_4 and $\bar{B}_4$ along with carry c_3 are added and two outputs S_4 and C_{out} are generated.

Case - 2 (when m = 1)

In this case the xor Gates produces the complement of the bits of the number B. and therefore no. A is connected directly whereas the no. B is connected in the complement form.

Stage - 1 : At this stage A_1 and $\bar{B}_1$ along with carry input (C_{in}) are added. and two output are S_1 and c_1 are generated. the carry c_1 are propogates to the next stage.

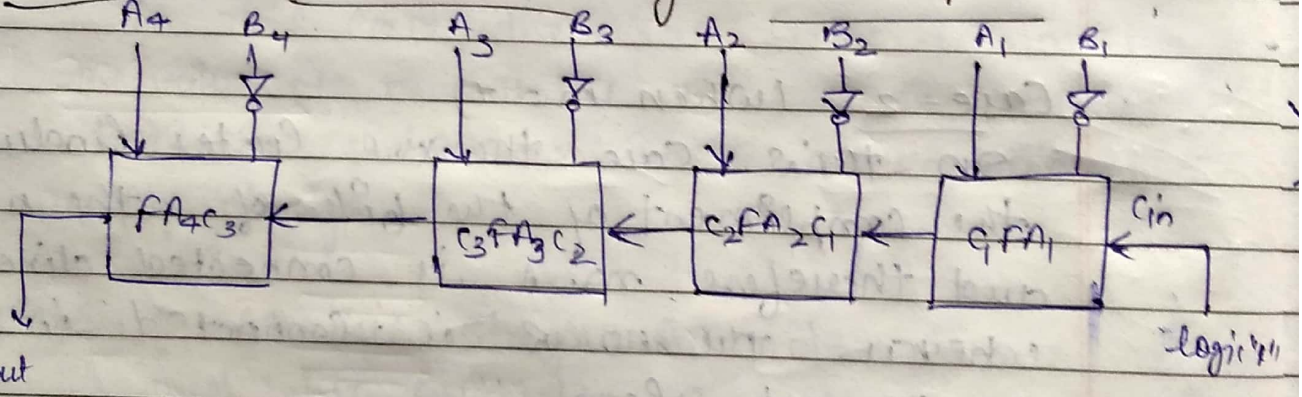
Stage :- 2 At this stage A_2 and $\bar{B}_2$ along with carry c_1 are added. the two output are generated S_2 and c_2. the carry c_2 are propogates to the next stage

Dated :
Page :

Stage :- 3 :- At this stage A_3 and $\bar{B}_3$ along with carry C_2 are generated. and two outputs S_3 and C_3 are generated and the carry C_3 are propogate to the next stage.

Stage :- 4 :- At this stage A_4 and $\bar{B}_4$ along with carry C_3 are generated and two output S_4 and Cout are generated.

Parallel subtraction using 2's Compliment:



Worked!

Stage 1 :- At this stage bits A_1 and $\bar{B}_1$ along with Cin are added and two output S_1 and C_1 are generated. The carry (C_1) generated this stage. Propagate.

Stage :- 2 At this stage bits A_2 and $\bar{B}_2$ along with carry C_1 are are added and two o/p S_2 and C_2 are generated The carry (C_2) generated at this stage propagated to the next stage

Dated :
Page :

Stage :3 At this bit A_3 and $\bar{B}_3$ along with carry C_2 are added and two o/p S_3 and C_3 are generated. The carry (C_3) generated at this stage propogates to next stage.

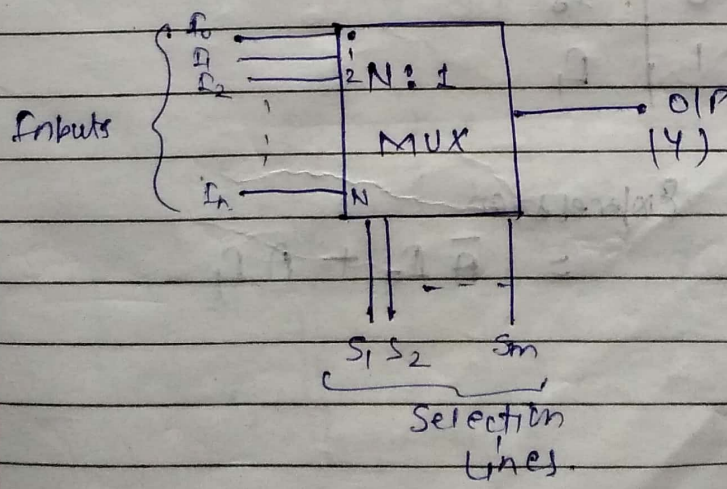
Stage :4 At this st A_4 and $\bar{B}_4$ along with carry C_3 are added and two output S_4 and Cout are genciated.

Imp # Multiplexer :

15-20 marks

V. Imp

Multiplexer is a type of combinational circuit it is used for multiplexing. It has several inputs, single output and several selection lines. Out of several inputs only one input is selected at a perticular instant of time. The selection of a perticular input is done with the help of the combination of selection lines. multiplexer is also known as many into one device



Dated :
Page :

The no. of selection lines is calculated by the relation

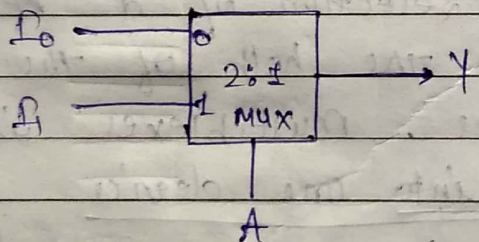
$$N = 2^m$$

where N = no. of I/p
M = no. of selection lines

small & comfy

* Types of multiplexers :-
multiplexers are of following types
i) 2:1 MUX
ii) 4:1 MUX
iii) 8:1 MUX
iv) 16:1 MUX

i) 2:1 Mux :- It has two I/p single O/p and single selection line



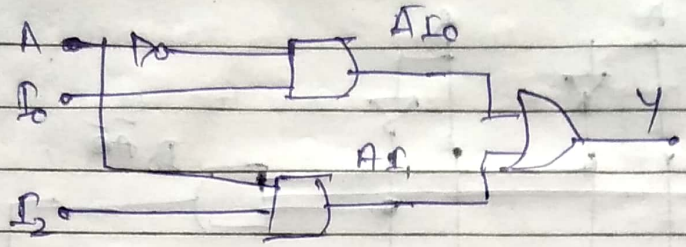
Truth table

A	Y
0	I_0
1	I_1

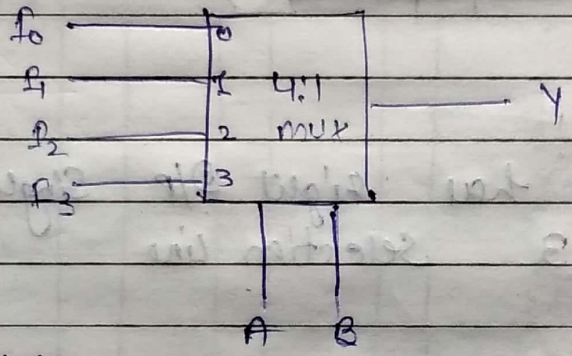
Boolean Expression
$= \bar{A} I_0 + A I_1$

Dated :

Page :



* ii) 4:1 mux :- It has four inputs a single output and 2 selection lines

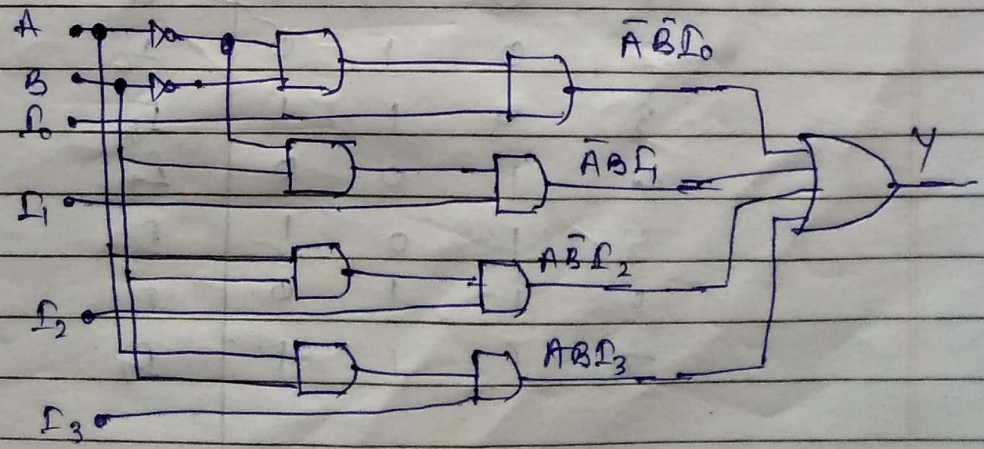


Truth table

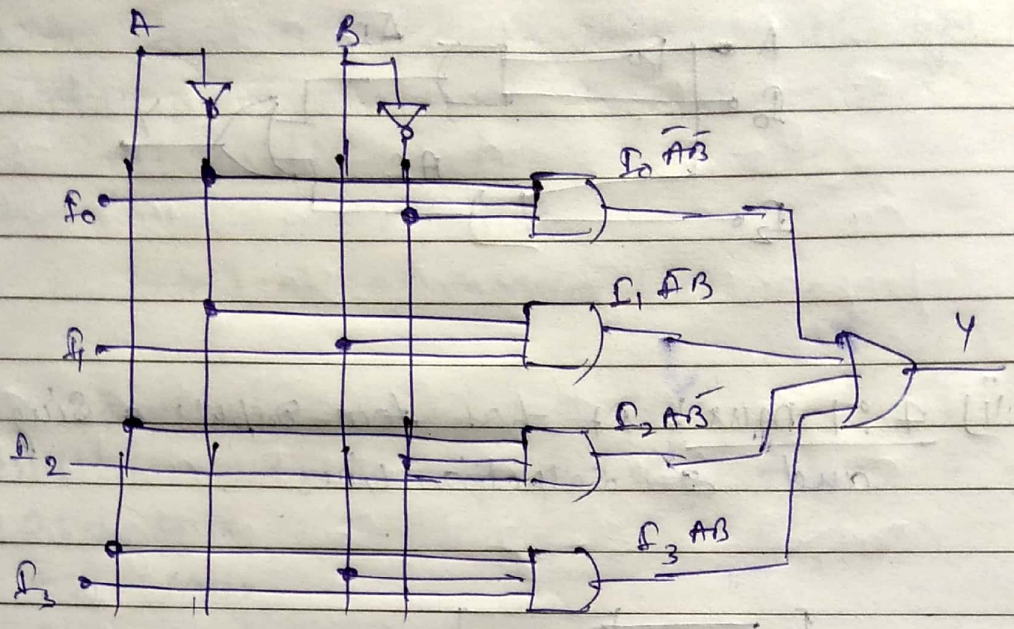
	A	B	Y
0	0	0	I_0
1	0	1	I_1
2	1	0	I_2
3	1	1	I_3

Boolean Expression :

$$Y = \bar{A}\bar{B}I_0 + \bar{A}BI_1 + A\bar{B}I_2 + ABI_3$$

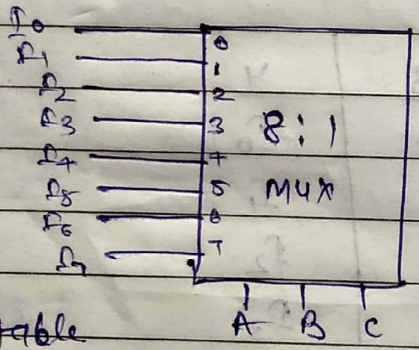


Dated :
Page :



(iii) 8:1 MUX:

It has eight I/P single output and 3 selection line



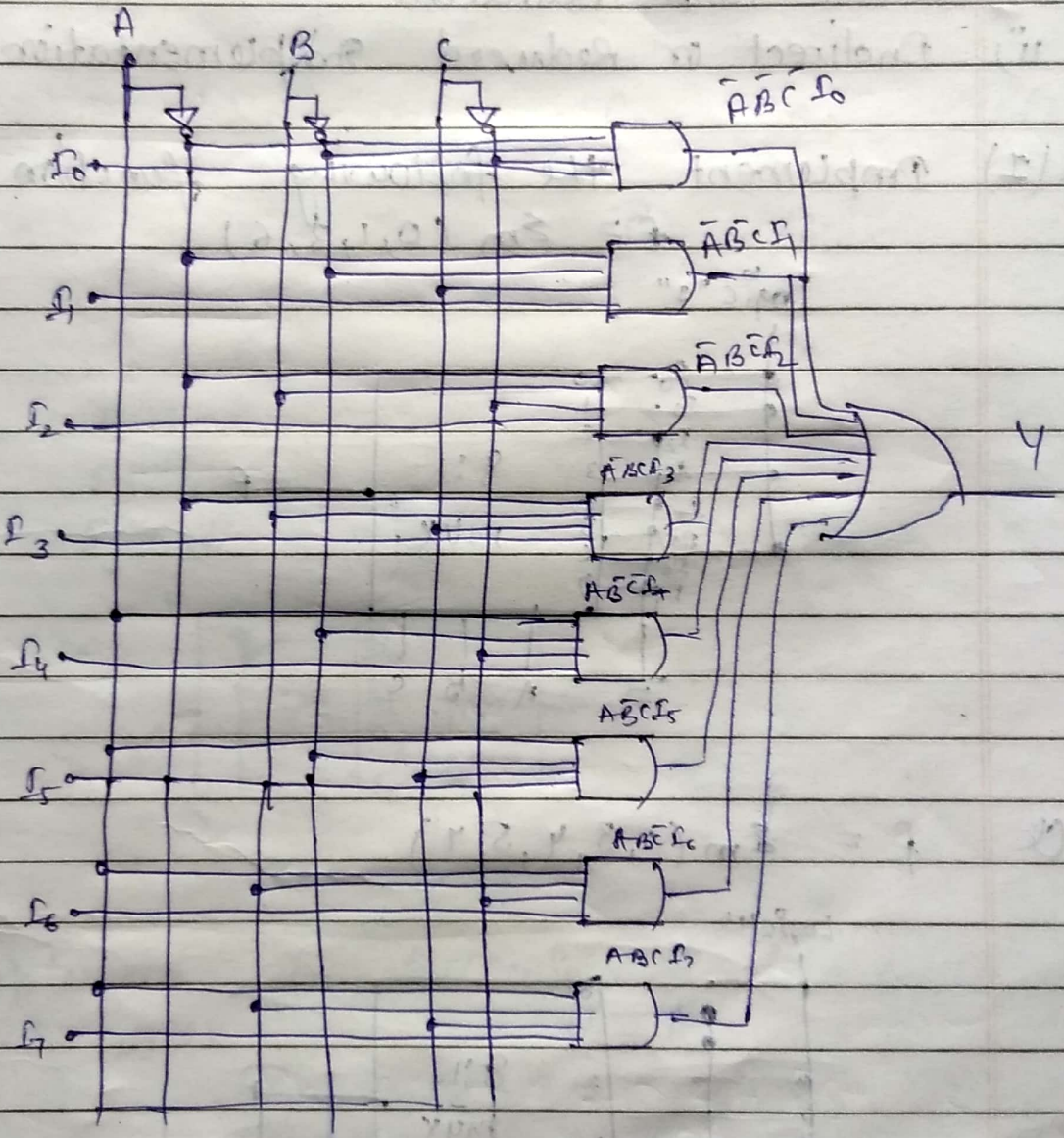
Truth table

	A	B	C	Y
0	0	0	0	I_0
1	0	0	1	I_1
2	0	1	0	I_2
3	0	1	1	I_3
4	1	0	0	I_4
5	1	0	1	I_5
6	1	1	0	I_6
7	1	1	1	I_7

Dated :
Page :

Boolean Expression:

$\Rightarrow \bar{A}\bar{B}\bar{C}I_0 + \bar{A}\bar{B}CI_1 + \bar{A}B\bar{C}I_2 + \bar{A}BCI_3 + A\bar{B}\bar{C}I_4 + A\bar{B}CI_5 + AB\bar{C}I_6 + ABCI_7$



Dated :

Page :

* Implementation using multiplexer:

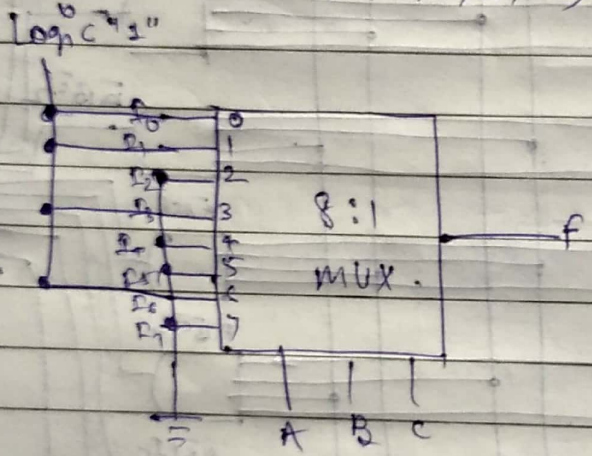
A multiplexer is used to implement various logic functions. Two types of implementations are used

i) Direct implementation

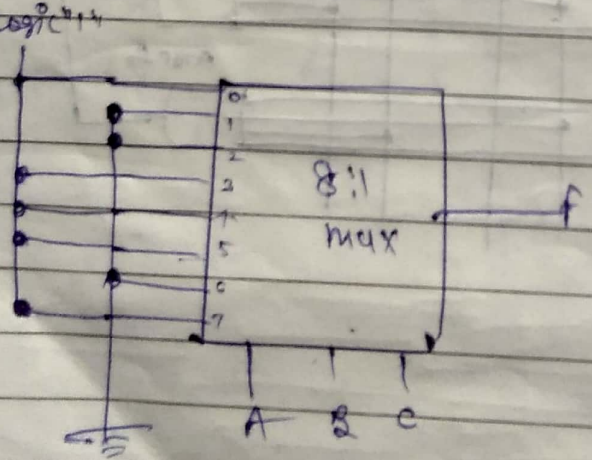
ii) Indirect or Reduced implementation.

Q(1) Implement the following function by using mux

$f = \Sigma m(0,1,3,6)$



Q $f = \Sigma m(0,3,4,5,7)$



Dated :
Page :

Indirect or Reduction Method :

Q. Implement the following function by using 4:1 mux

$$f = \Sigma m(0,1,3,6)$$

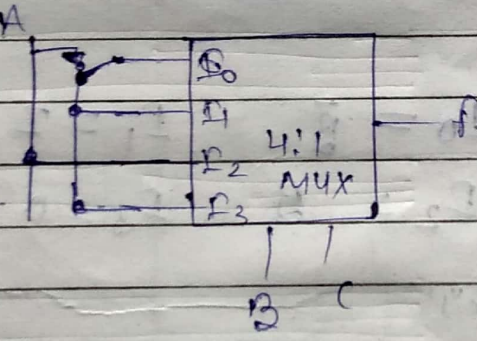
		I_0	I_1	I_2	I_3
$\bar{A}$	0	(0)	(1)	2	(3)
A	1	4	5	(6)	7

$I_0 = \bar{A}$

$I_1 = \bar{A}$

$I_2 = A$

$I_3 = \bar{A}$



Que. Implement the following function by using 2:1 mux

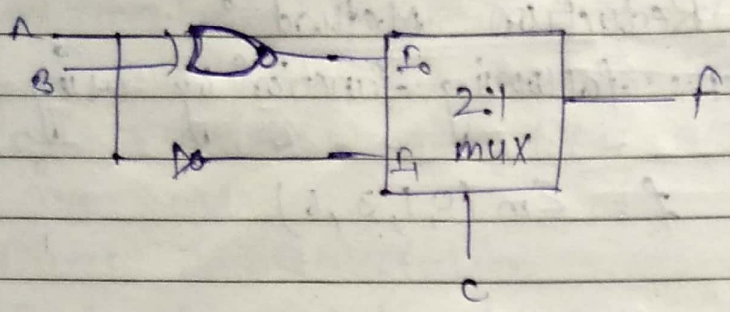
$$f = \Sigma m(0,1,3,6)$$

			I_0	I_1
$\bar{A}\bar{B}$	0	0	(0)	(1)
$\bar{A}B$	0	1	2	(3)
$A\bar{B}$	1	0	4	5
AB	1	1	(6)	7

$I_0 = \bar{A}\bar{B} + AB = (A \odot B)$

$I_1 = \bar{A}B + \bar{A}B = \bar{A}(B+\bar{B}) = \bar{A}$

Dated :
Page :



Que Implement the following function by using 4:1 mux with B as a control variable.

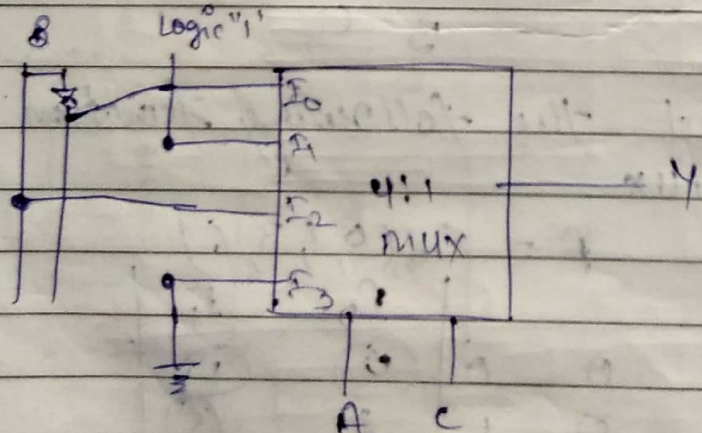
$$f = \Sigma m(0,1,3,6)$$

	A	B	C
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

	I_0	I_1	I_2	I_3
B=0	(0)	(1)	4	5
B=1	2	(3)	(6)	7

$I_0 = \bar{B}$ $\quad I_1 = 1$

$I_2 = B$ $\quad I_3 = 0$



Dated :

Page :

Que: Implement the following mux by using 2:1 with A and C as a control variable

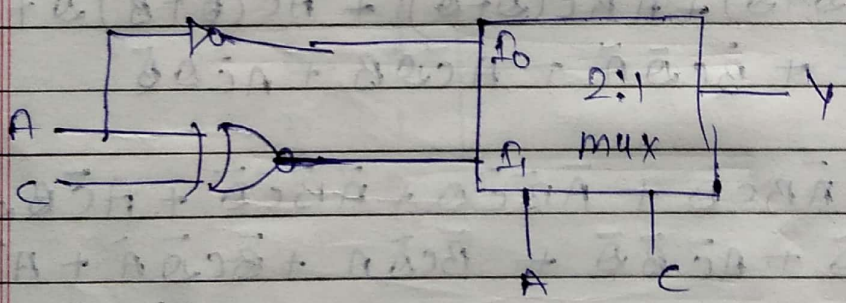
$$f = \Sigma m(0,1,3,6)$$

		I_0	I_1
$\bar{A}\bar{C}$	0 0	(0)	2
$\bar{A}C$	0 1	(1)	(3)
$A\bar{C}$	1 0	4	(6)
AC	1 1	5	7

$$I_0 = \bar{A}\bar{C} + \bar{A}C$$

$$\Rightarrow \bar{A}(\bar{C}+C) = \bar{A}$$

$$I_1 = \bar{A}C + A\bar{C} \Rightarrow A \oplus C$$



8 + 21

Dated :

Page :

Q Implement the following function

$f = \bar{A}B + A\bar{C} + \bar{B}C\bar{D} + A\bar{C}D$

by using :

i) 16 : 1 mux
ii) 8 : 1 mux
iii) 8 : 1 mux with C as a control Variable
iv) 8 : 1 mux " B " " " "
v) 4 : 1 mux " A and C as a cont. var.
vi) 4 : 1 " with D as a cont. var.
vii) 2 : 1 mux
viii) 2 : 1 mux with A, B and D as a cont. var.

$f = \bar{A}B(C+\bar{C})(D+\bar{D}) + A\bar{C}(B+\bar{B})(D+\bar{D}) + \bar{B}C\bar{D}(A+\bar{A}) + A\bar{C}D(B+\bar{B})$

$= \bar{A}BC(D+\bar{D}) + \bar{A}B\bar{C}(D+\bar{D}) + A\bar{C}(B+\bar{B})D + A\bar{C}(B+\bar{B})\bar{D} + \bar{B}C\bar{D}A + \bar{B}C\bar{D}\bar{A} + A\bar{C}DB + A\bar{C}D\bar{B}$

$= \bar{A}BCD + \bar{A}BC\bar{D} + \bar{A}B\bar{C}D + \bar{A}B\bar{C}\bar{D} + A\bar{C}BD + A\bar{C}B\bar{D} + A\bar{C}\bar{B}D + A\bar{C}\bar{B}\bar{D} + \bar{B}C\bar{D}A + \bar{B}C\bar{D}\bar{A} + A\bar{C}DB + A\bar{C}D\bar{B}$

$= \bar{A}BCD + \bar{A}BC\bar{D} + \bar{A}B\bar{C}D + \bar{A}B\bar{C}\bar{D} + AB\bar{C}D + AB\bar{C}\bar{D} + A\bar{C}B\bar{D} + A\bar{B}C\bar{D} + \bar{A}\bar{B}C\bar{D} + AB\bar{C}D + ABCD$

$= \Sigma m(7, 6, 1, 0, 13, 9, 12, 8, 11, 2, 13, 9)$

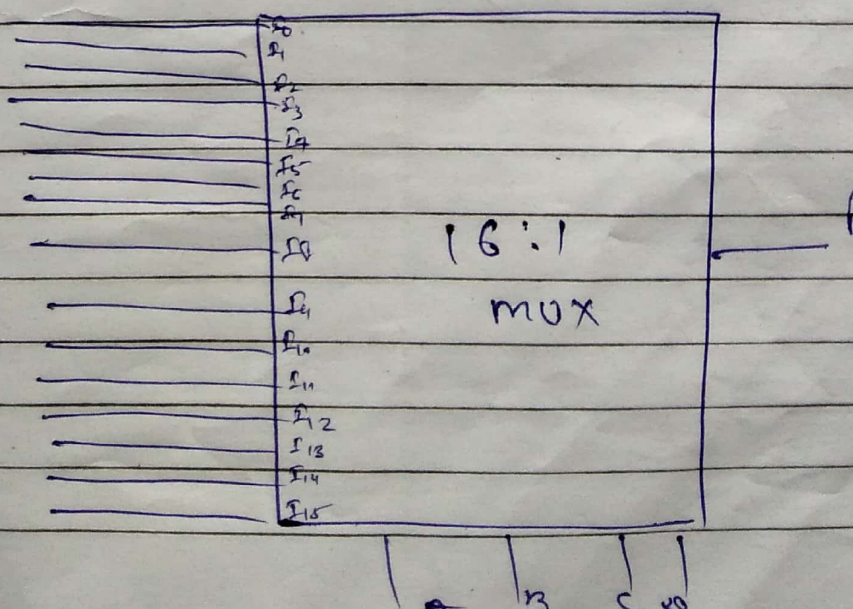
Dated :

Page :

$f = (0,$

$f = \Sigma m\,(0, 1, 2, 6, 7, 8, 9, 11, 12, 13,)$

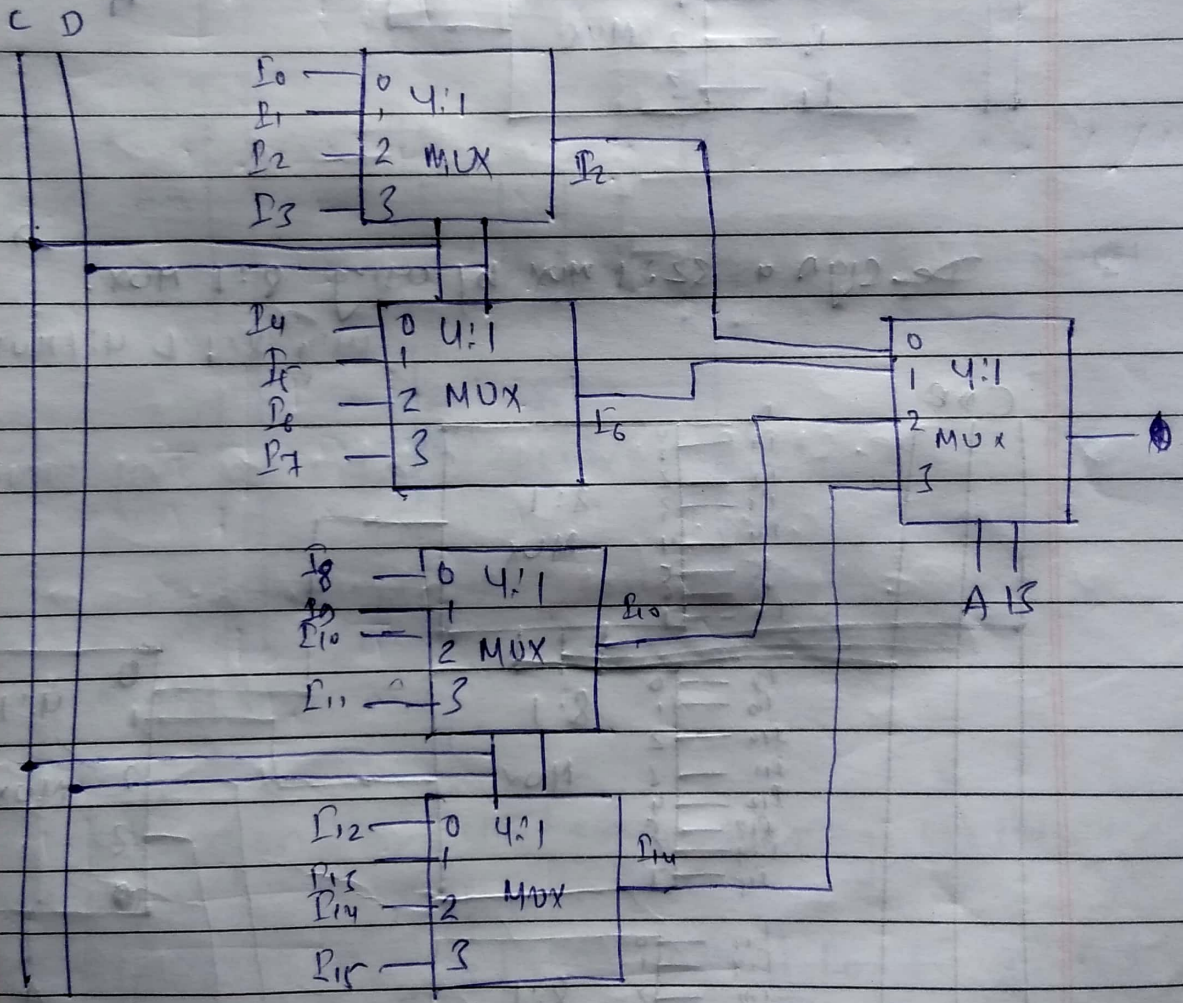
	A	B	C	D	I_0	I_1	I_2	I_3
$\bar{A}\bar{B}\bar{C}\bar{D}$	0	0	0	0				
$\bar{A}\bar{B}\bar{C}D$	0	0	0	1				
$\bar{A}\bar{B}C\bar{D}$	0	0	1	0				
$\bar{A}\bar{B}CD$	0	0	1	1				
$\bar{A}B\bar{C}\bar{D}$	0	1	0	0				
$\bar{A}B\bar{C}D$	0	1	0	1				
$\bar{A}BC\bar{D}$	0	1	1	0				
$\bar{A}BCD$	0	1	1	1				
$A\bar{B}\bar{C}\bar{D}$	1	0	0	0				
$A\bar{B}\bar{C}D$	1	0	0	1				
$A\bar{B}C\bar{D}$	1	0	1	0				
$A\bar{B}CD$	1	0	1	1				
$AB\bar{C}\bar{D}$	1	1	0	0				
$AB\bar{C}D$	1	1	0	1				
$ABC\bar{D}$	1	1	1	0				
$ABCD$	1	1	1	1				



MUX TREE:-

- In this a number of smaller multiplexes are connected together to form a larger multiplexes.

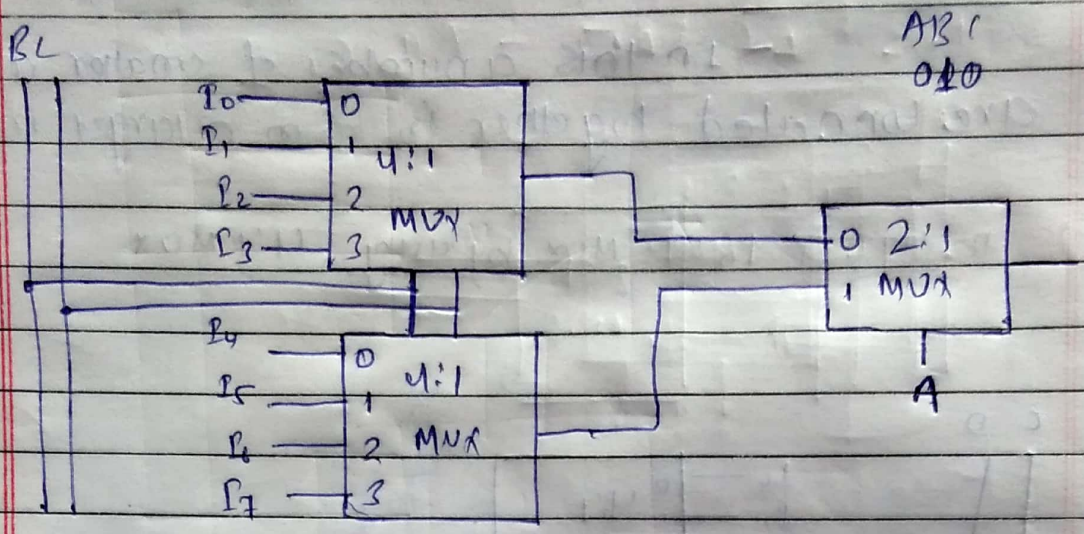
(a) Design a 16:1 Mux by using 4:1 Mux



(b) Design 8:1 Mux by using 4:1 Mux and 2:1 Mux.

or,

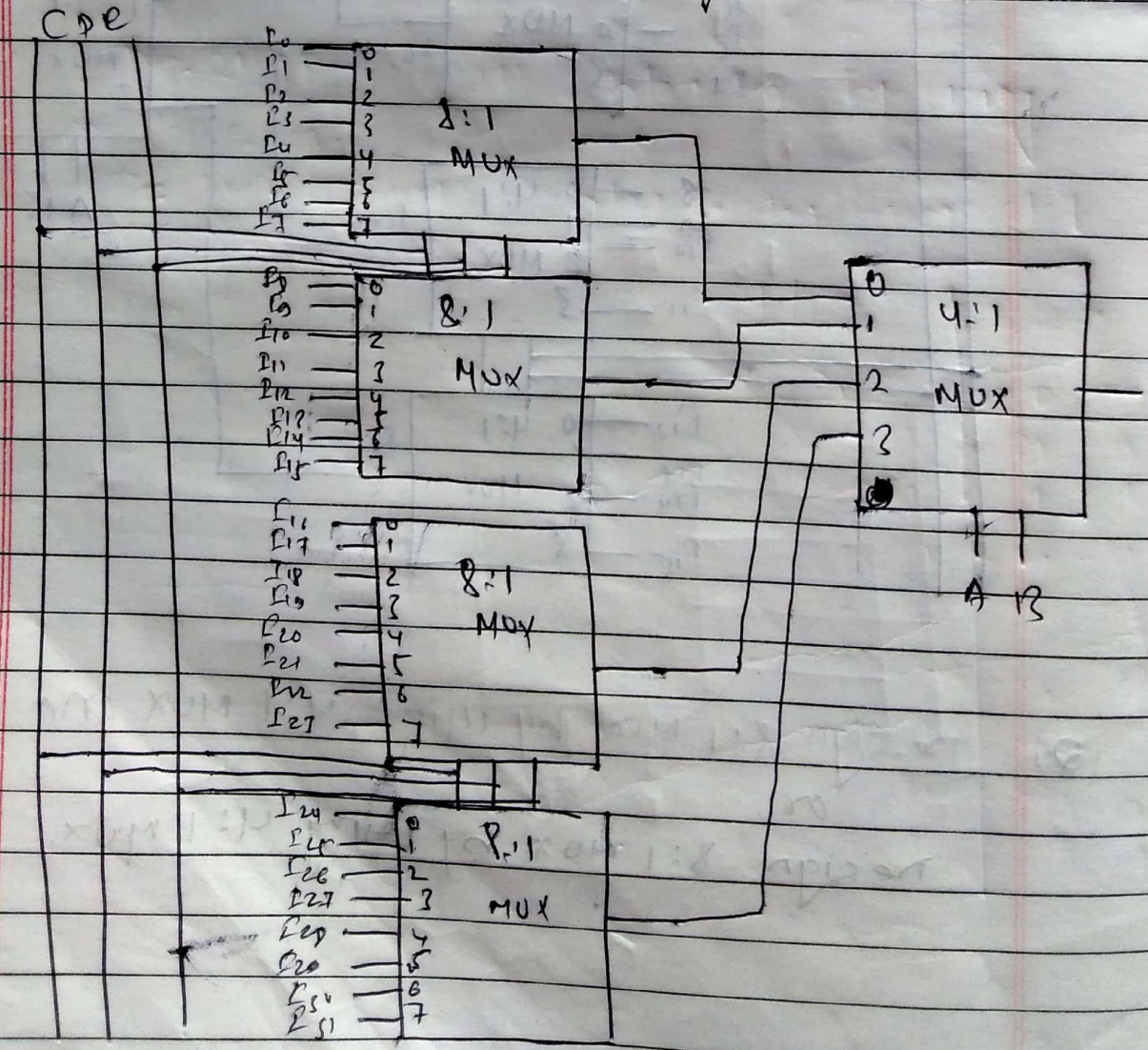
Design 8:1 Mux by using 4:1 Mux.



(2.) Design a 32:1 MUX by using 8:1 MUX.

→ M.Tech

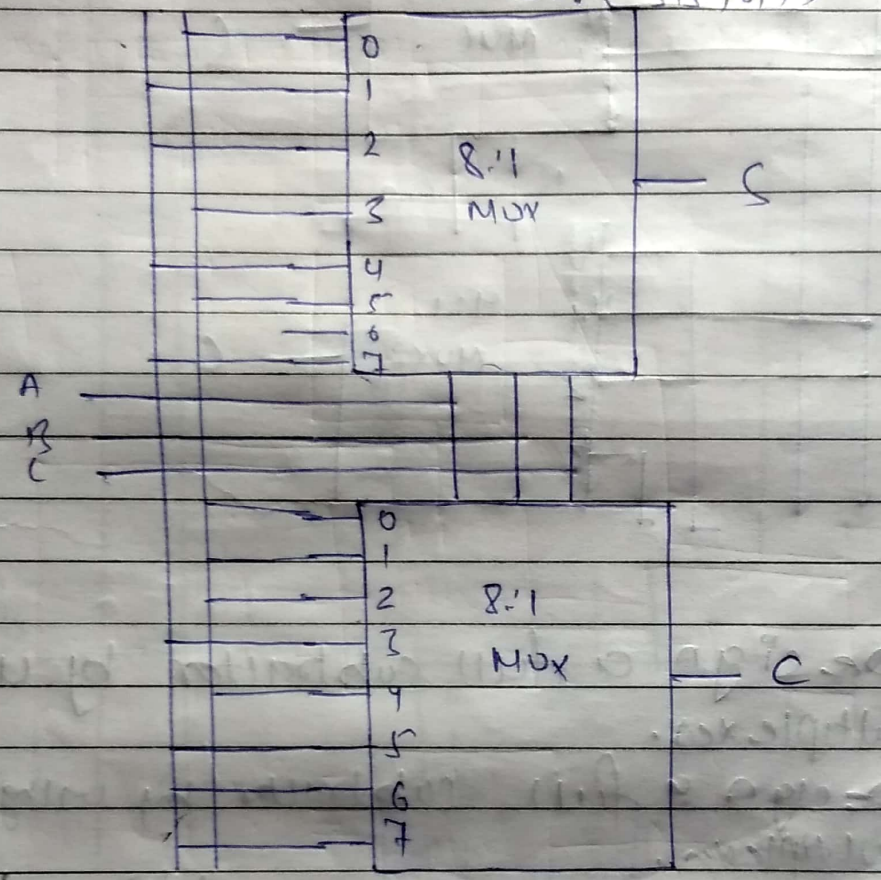
using 8:1 & 4:1 MUX



Q.) Design a full header by using Multiplexers.

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

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



Q.) Design a full header using 4:1 MUX.

S	I_0	I_1	I_2	I_3
$\bar{A}$ 0	0	(1)	(2)	3
A 1	(4)	5	6	(7)

C	I_0	I_1	I_2	I_3
$\bar{A}$ 0	0	1	2	(3)
A 1	4	(5)	(6)	(7)

$I_0 = A$

$I_1 = \bar{A}$

$I_2 = \bar{A}$

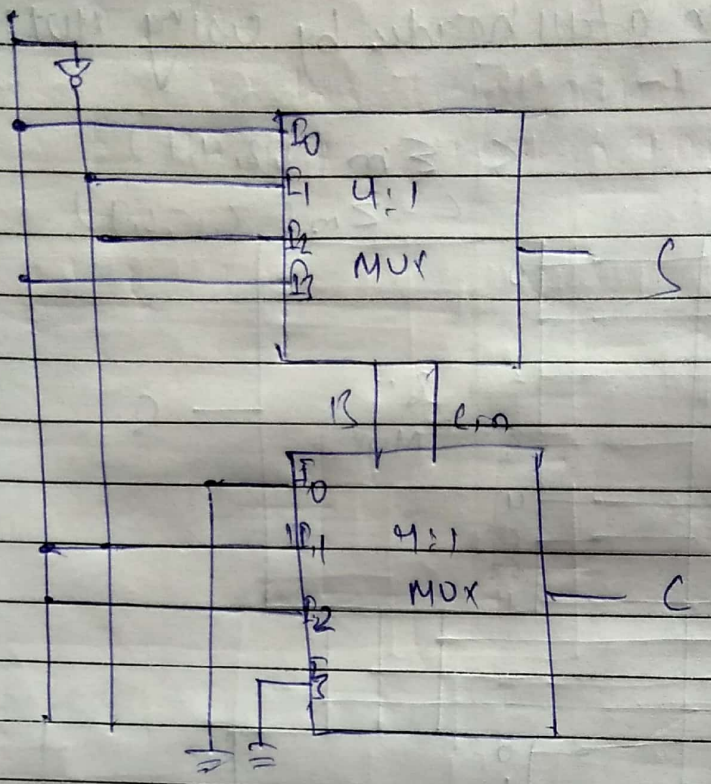
$I_3 = A$

$I_0 = 0$

$I_1 = A$

$I_2 = A$

$I_3 = 1$



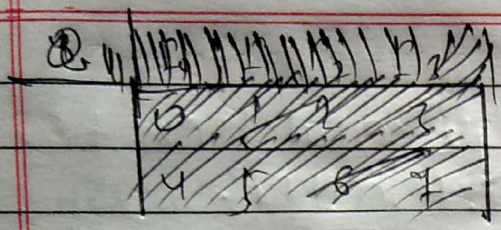
(Q) Design a full subtractor by using multiplexers.

(Q) Design a full subtractor by using 4:1 Multiplexer.

I/P			O/P	
x	y	z	D	B
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	0
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

A

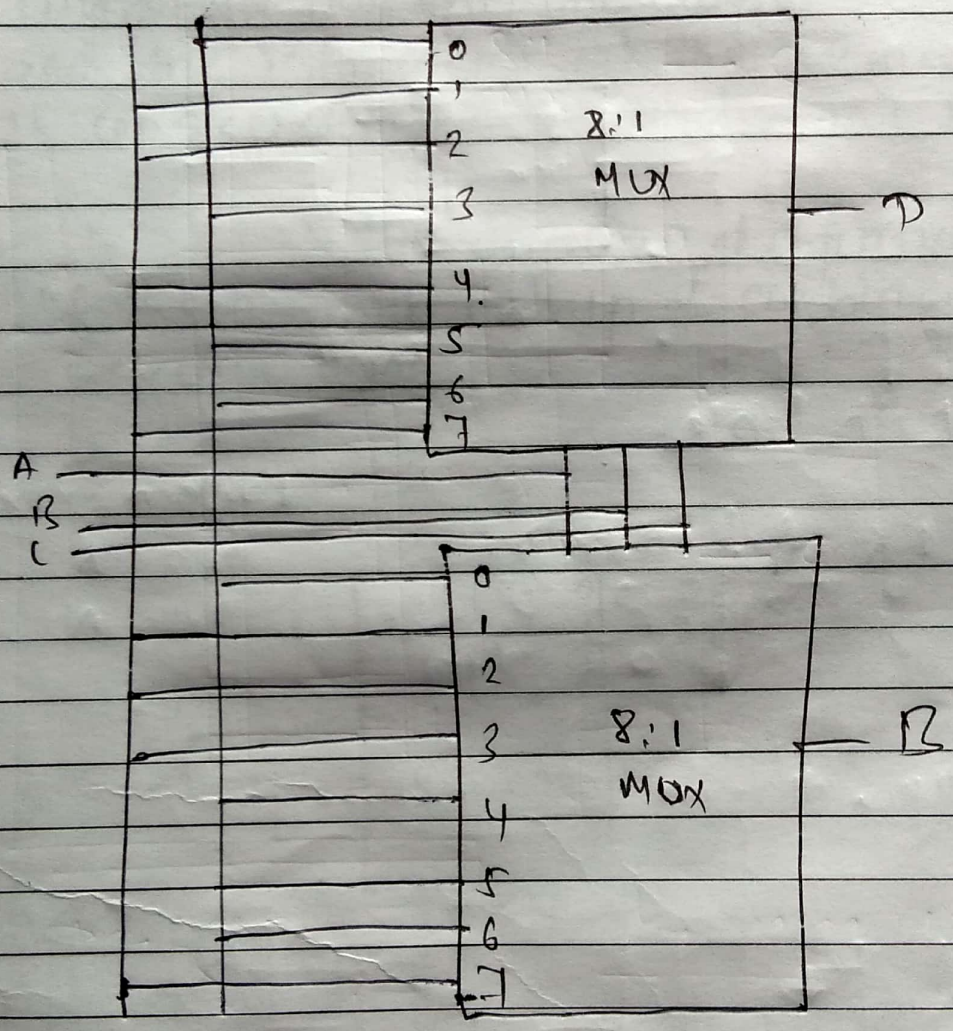
Q.1



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

$B = \Sigma m (1,2,3,7)$

Q.1

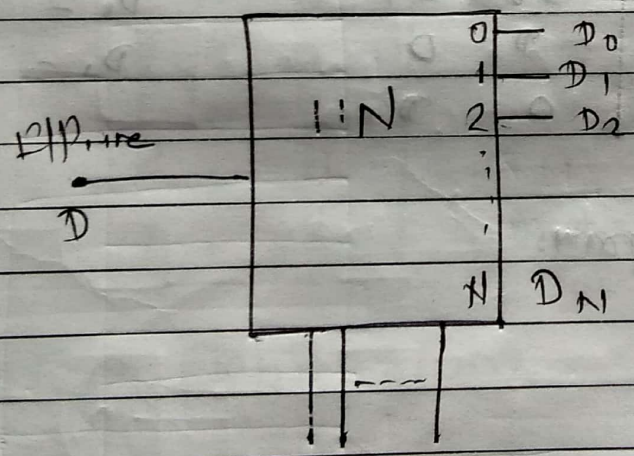


Q2 Soln

De-Multiplexer:-

→ De-Multiplexer is a one into many device it performs the reverse operation of a multi-plexer. A de-multiplexer has single data line multiple output lines. at a particular instant of time only one output line is selected and the information at the single input line is connected to the selected output line.

- the selection of a particular output line is done with the help of combination of selection lines.



Types:-

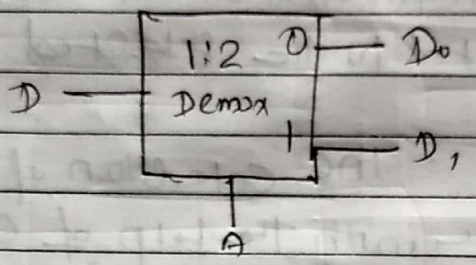
Depending upon the number of outputs de-multiplexers are of following types:-

(i) 1:2 Demux
(ii) 1:4 Demux
(iii) 1:8 Demux
(iv) 1:16 Demux

(i) 1:2 De-MUX :-

- It has single input lines two output lines at single selection line.

Block diagram:-



truth table:-

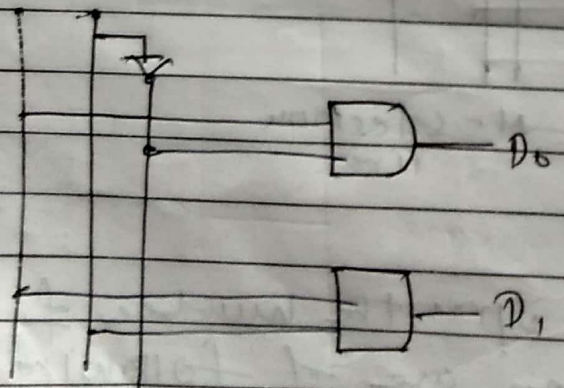
	A	D_0	D_1
0	0	D	0
1	1	0	D

Boolean Expression.

$D_0 = \overline{A}$

$D_1 = A$

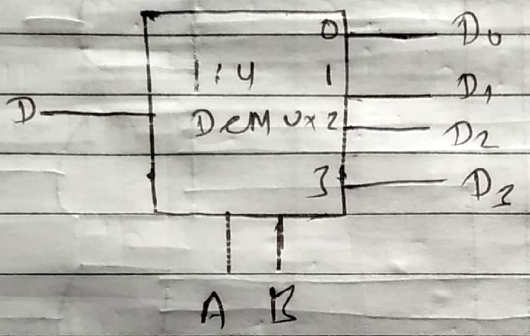
logic diagram:



(iii) 1:4 DEMUX:-

It has single input lines 4 output lines and 2 selection lines.

Block diagram:-



Truth table:-

A	B	D_0	D_1	D_2	D_3
0	0	D	0	0	0
1	0	0	D	0	0
2	1	0	0	D	0
3	1	0	0	0	D

Boolean expressions:-

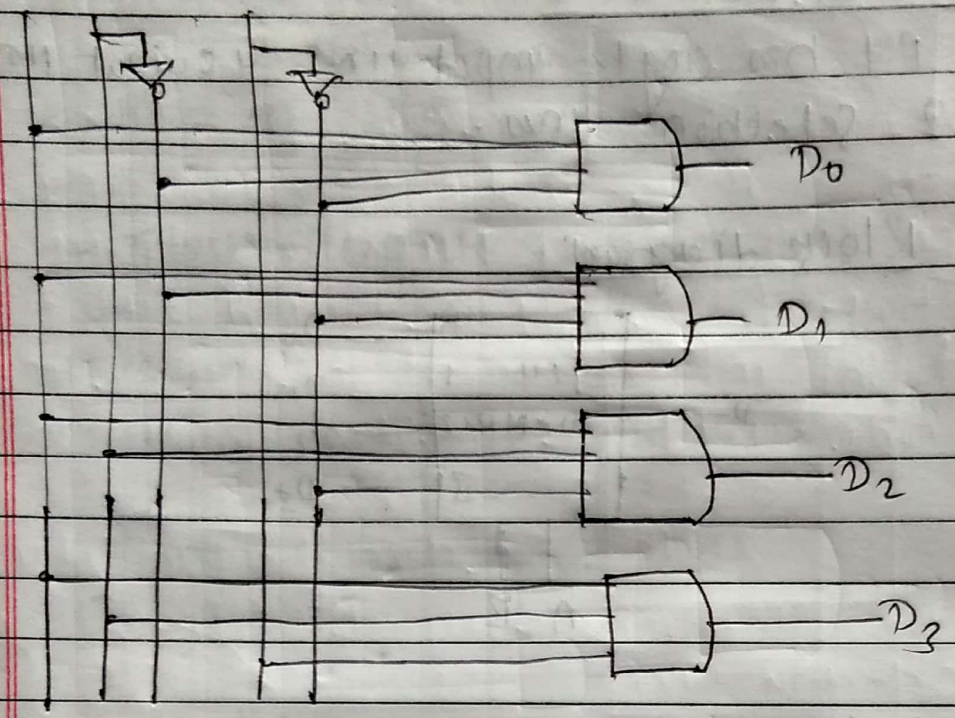
$D_0 = \bar{A}\bar{B}$

$D_1 = \bar{A}B$

$D_2 = A\bar{B}$

$D_3 = AB$

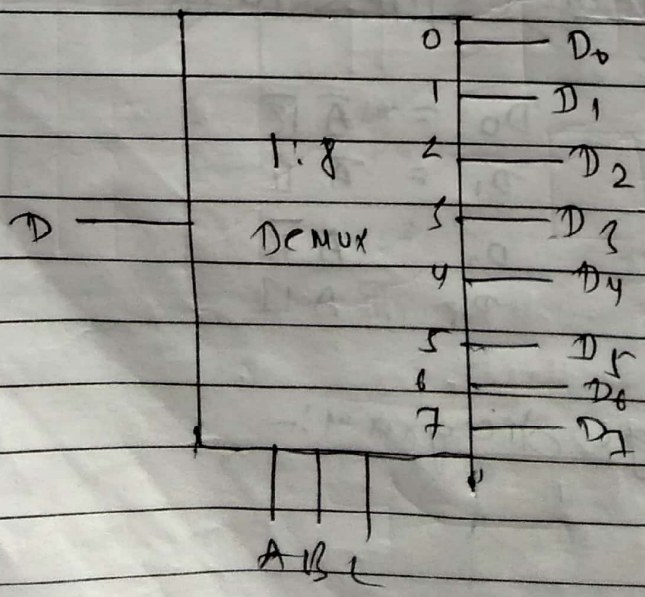
logic diagram:-



(iii) 1:8 De-multiplexer :-

It has single input and 8 output lines and 3 selection lines.

Block diagram



Truth table:

	A	B	C	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0
2	0	1	0	0	0	1	0	0	0	0	0
3	0	1	1	0	0	0	1	0	0	0	0
4	1	0	0	0	0	0	0	1	0	0	0
5	1	0	1	0	0	0	0	0	1	0	0
6	1	1	0	0	0	0	0	0	0	1	0
7	1	1	1	0	0	0	0	0	0	0	1

Boolean expression:

$D_0 = \bar{A}\bar{B}\bar{C}$

$D_1 = \bar{A}\bar{B}C$

$D_2 = \bar{A}B\bar{C}$

$D_3 = \bar{A}BC$

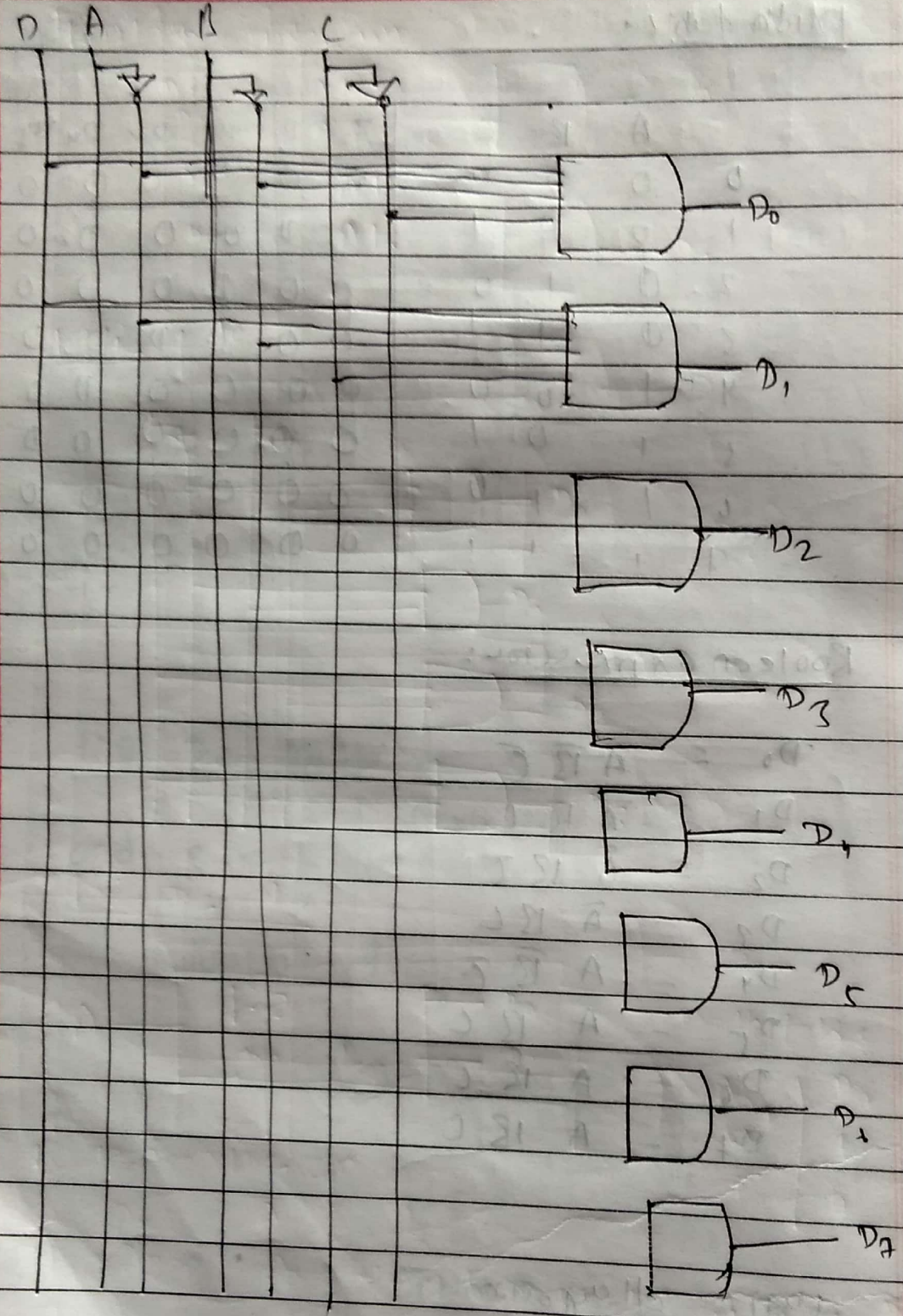
$D_4 = A\bar{B}\bar{C}$

$D_5 = A\bar{B}C$

$D_6 = AB\bar{C}$

$D_7 = ABC$

logic diagram


D
A
B
C
D_0
D_1
D_2
D_3
D_4
D_5
D_6
D_7

De-coder:-

- It is a type of combinational circuit which is used to decode the binary information received at the receiver. In this data is retrieved from the binary code received.

TYPES:-

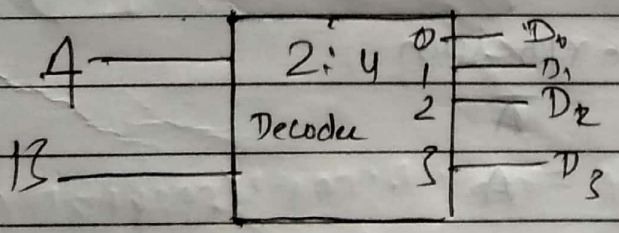
Depending upon the number of inputs and outputs decoders are classified as follows:-

(i) 1:2 Decoder
(ii) 2:4 "
(iii) 3:8 "
(iv) 4:16 "

N:2^N
where N is no. of inputs

(ii) 2:4 De-coder:-

- It receives binary codes each having two bits and activates the corresponding output line.



Truth table:-

	A	B	D_0	D_1	D_2	D_3
0	0	0	1	0	0	0
1	0	1	0	1	0	0
2	1	0	0	0	1	0
3	1	1	0	0	0	1

Boolean expression

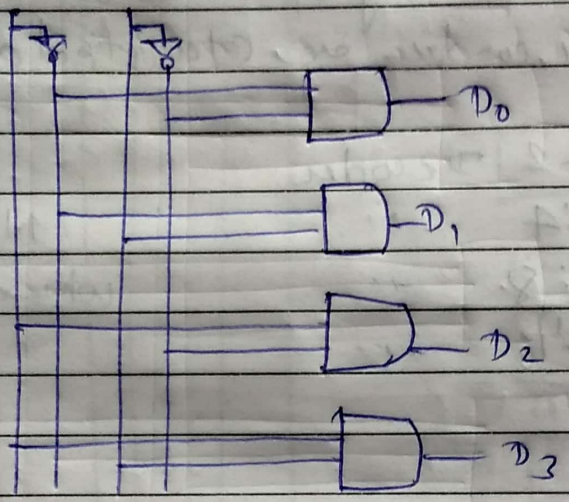
$D_0 = \bar{A}\bar{B}$

$D_1 = \bar{A}B$

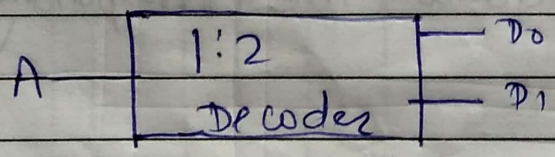
$D_2 = A\bar{B}$

$D_3 = AB$

Logic gate:-



(i) 1:2 Decoder:-



Truth table

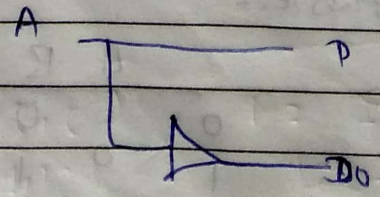
A	D_0	D_1
0	1	0
1	0	1

Boolean expression

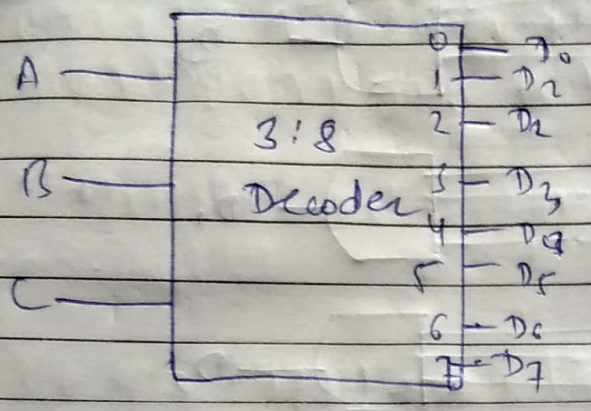
$D_0 = \bar{A}$

$D_1 = A$

logic gate



(iii) 3:8 Decoder:-



Truth table:-

	A	B	C	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0
2	0	1	0	0	0	1	0	0	0	0	0
3	0	1	1	0	0	0	1	0	0	0	0
4	1	0	0	0	0	0	0	1	0	0	0
5	1	0	1	0	0	0	0	0	1	0	0
6	1	1	0	0	0	0	0	0	0	1	0
7	1	1	1	0	0	0	0	0	0	0	1

Boolean expression

$D_0 = \bar{A}\bar{B}\bar{C}$

$D_1 = \bar{A}\bar{B}C$

$D_2 = \bar{A}B\bar{C}$

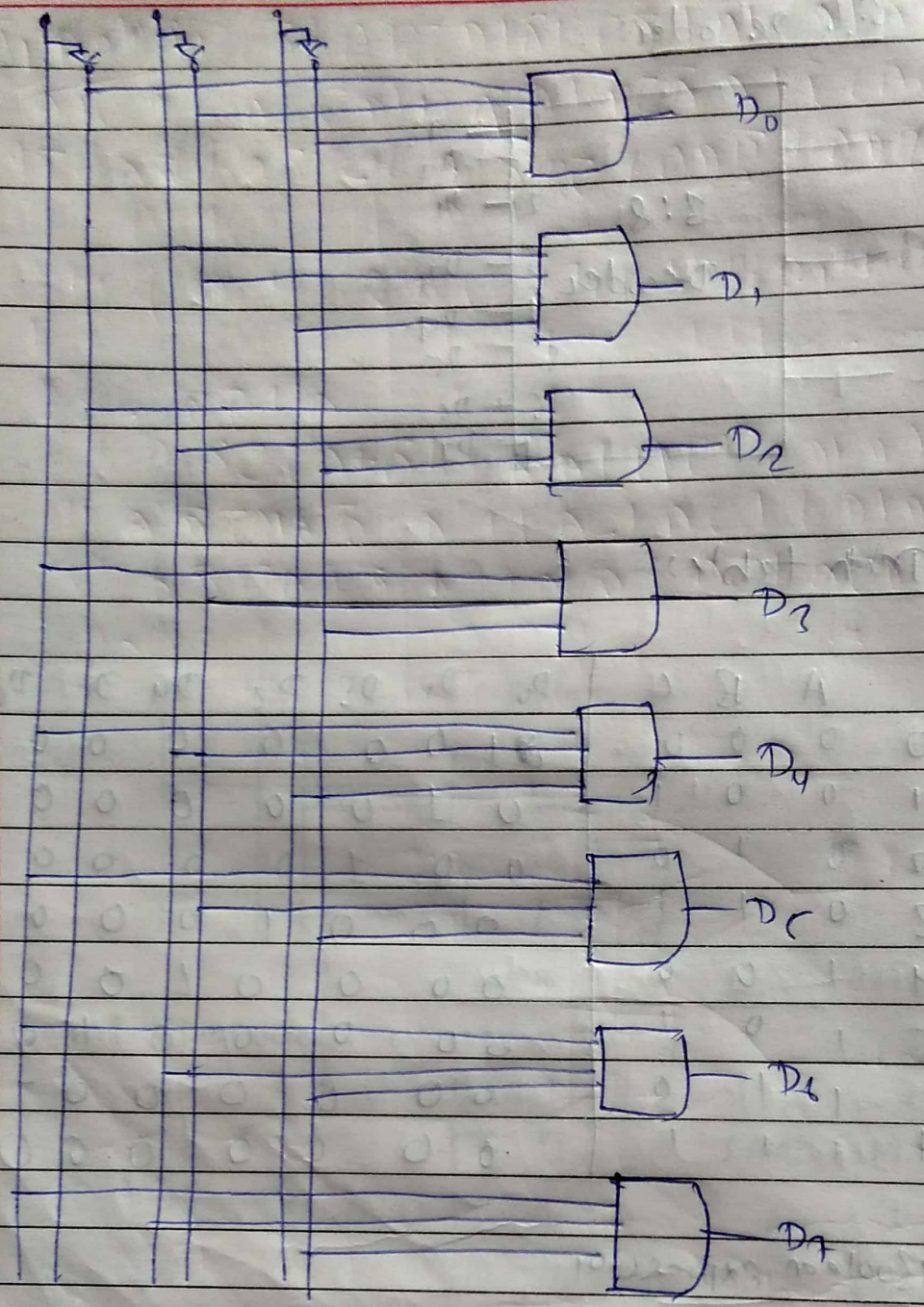
$D_3 = \bar{A}BC$

$D_4 = A\bar{B}\bar{C}$

$D_5 = A\bar{B}C$

$D_6 = AB\bar{C}$

$D_7 = ABC$

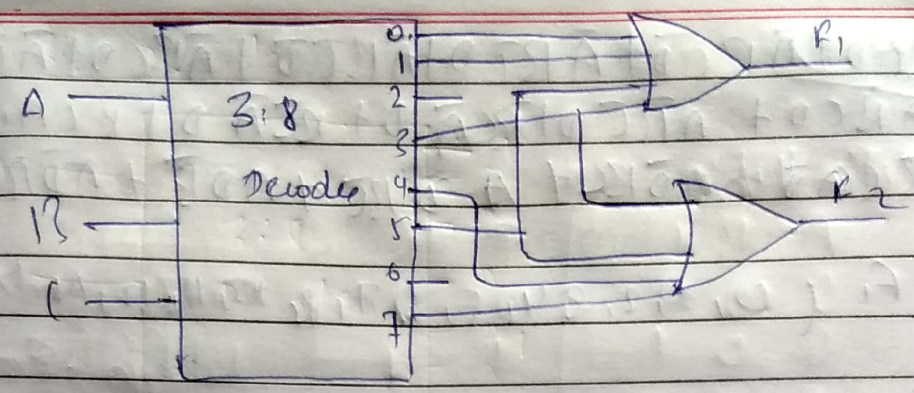


↓ (a) Boolean function implementation by using De-coder

(b) Implement the following function by using De-coder.

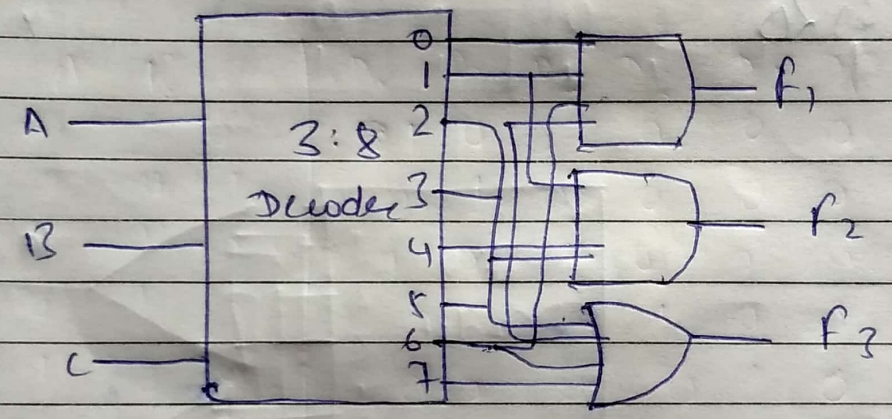
$F_1 = \Sigma m(0,1,3,5)$. $F_2 = \Sigma m(3,4,5,7)$

$[0,1,5,6,9,11,13)$



Q.1) Implement the following fn in by using Decode.

$F_1 = \Pi m\,(0,1,7,6)$ $F_2 = \Pi m(1,2,3)$ $F_3 = \Sigma m\,(2,3,6,7)$



Q.2) Implemat the function using decodes

$F_1 = \bar{A}\bar{B}C + A\bar{D}$

$F_2 = \bar{D}\bar{C} + A$

Soln

$F_1 = \bar{A} + \bar{B}C + A\bar{D}$

$= \bar{A}(B+\bar{B})(1+\bar{C})(D+\bar{D}) + \bar{B}C(\bar{A}+A)(D+\bar{D}) + A\bar{D}(B+\bar{B})(1+\bar{C})$

$\Rightarrow (\bar{A}B + \bar{A}\bar{B})(CD + C\bar{D} + \bar{C}D + \bar{C}\bar{D}) + (\bar{B}C\bar{A} + \bar{B}CA)(D+\bar{D}) + (A\bar{D}B + A\bar{D}\bar{B})(1+\bar{C})$

$\Rightarrow \bar{A}B\bar{C}D + \bar{A}BC\bar{D} + \bar{A}BCD + \bar{A}\bar{B}CD + \bar{A}\bar{B}C\bar{D} + A\bar{B}\bar{C}D + \bar{A}B\bar{C}\bar{D} + \bar{A}BC\bar{D} + \bar{A}\bar{B}\bar{C}D + B\bar{C}\bar{A}D + B\bar{C}AD + B\bar{C}A\bar{D} + A\bar{D}B\bar{C} + AD\bar{B}\bar{C} + AD\bar{B}C$

$= F_1(0,1,2,3,4,5,6,7,8,10,11,12,14)$

$F_2 = \bar{D}\bar{C} + A$

$\Rightarrow \bar{A}\bar{C}(A+\bar{A}) + A(1+\bar{C})(D+\bar{D})$

$\Rightarrow A\bar{D}\bar{C}A + \bar{D}\bar{C}\bar{A} + (AC + A\bar{C})(D+\bar{D})$

$\Rightarrow \bar{D}\bar{C}A + \bar{D}\bar{C}\bar{A} + DC + AC + A\bar{C}D + A\bar{C}\bar{D}$

$\Rightarrow 10$

(Q.1) Design a full Adder by using decoder.

(Q.2) " " " Subtractor " " "

(Q.3) Design a binary to octal decoder

(Q.4) Design a bcd to decimal decode.

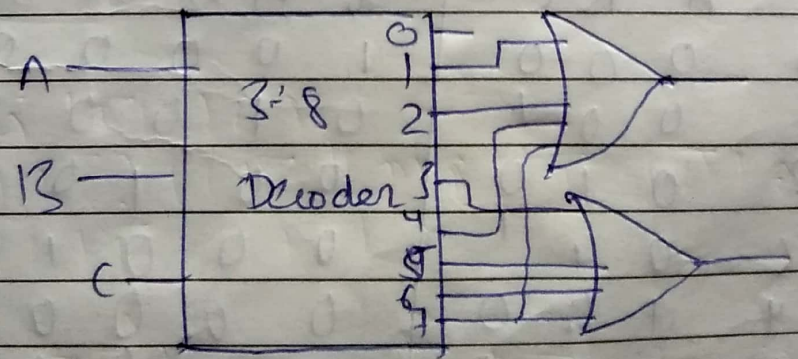
Soln (1)

Truth table

	A	B	C	S	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

$S = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$ or $S, F_1 = \sum(1,2,4,7)$

$C = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$ or $C, F_2 = \sum(3,5,6,7)$



Date
Page

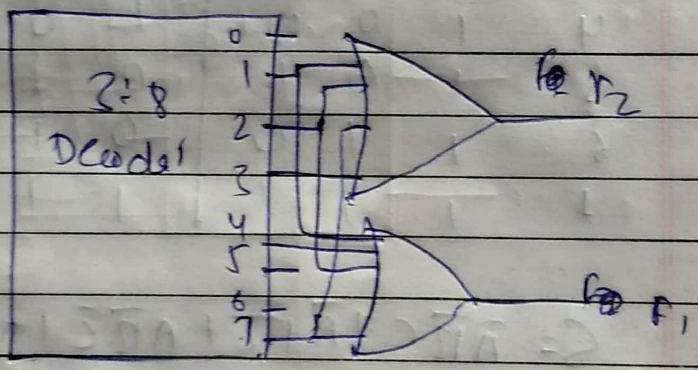
Com (II)

Truth Table:-

	A	B	C	D	B
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	1	1
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	0
6	1	1	0	0	0
7	1	1	1	1	1

$D = F_2 = \Sigma m(1,2,4,7)$

$B = F_1 = \Sigma m(1,2,3,7)$



Com (III)

	A	B	C	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0
2	0	1	0	0	0	1	0	0	0	0	0
3	0	1	1	0	0	0	1	0	0	0	0
4	1	0	0	0	0	0	0	1	0	0	0
5	1	0	1	0	0	0	0	0	1	0	0
6	1	1	0	0	0	0	0	0	0	1	0
7	1	1	1	0	0	0	0	0	0	0	1

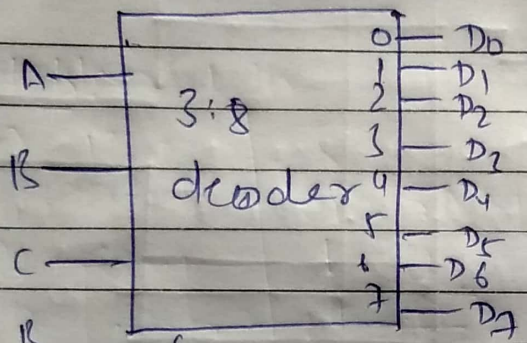
Expression

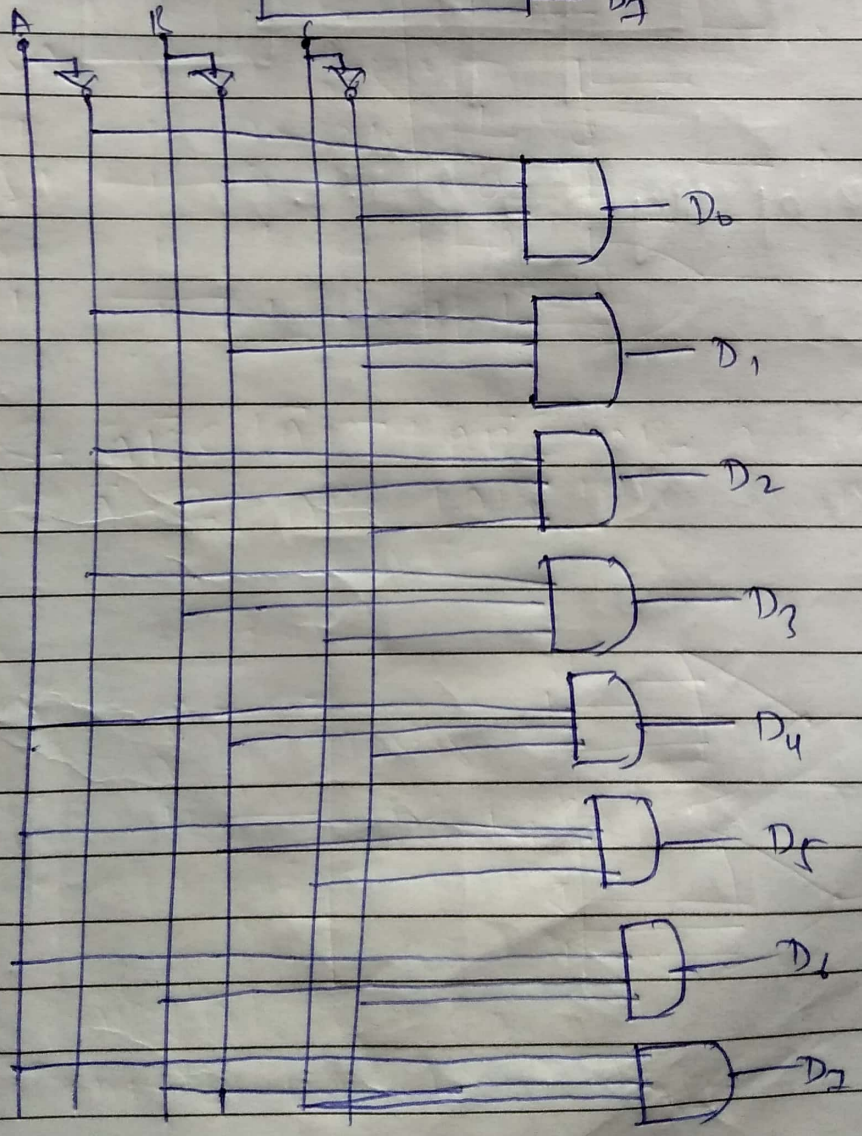
$D_0 = \bar{A}\bar{B}\bar{C}$ $\quad D_4 = A\bar{B}\bar{C}$

$D_1 = \bar{A}\bar{B}C$ $\quad D_5 = A\bar{B}C$

$D_2 = \bar{A}B\bar{C}$ $\quad D_6 = AB\bar{C}$

$D_3 = \bar{A}BC$ $\quad D_7 = ABC$





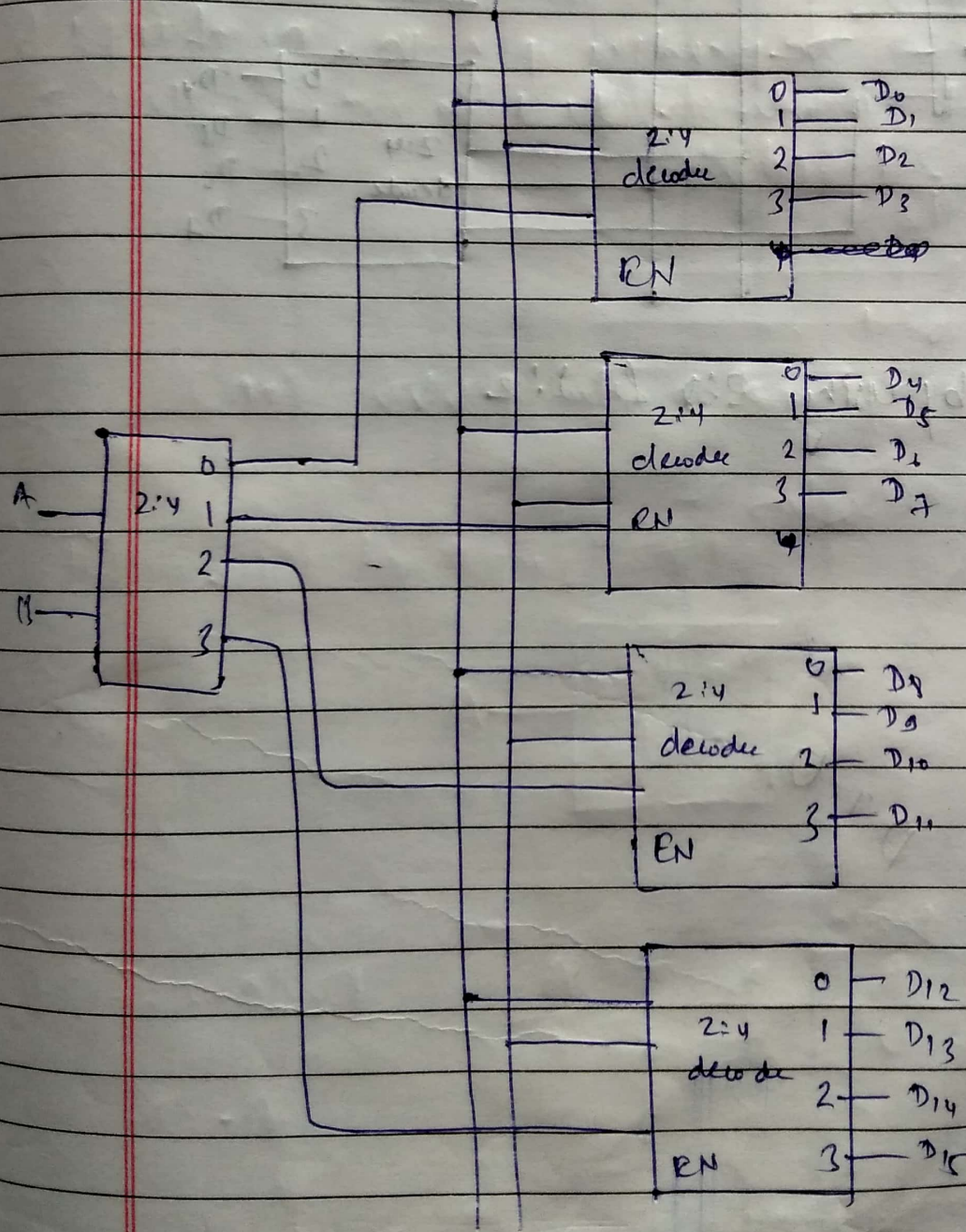
(9) truth table :-

A	B	C	D

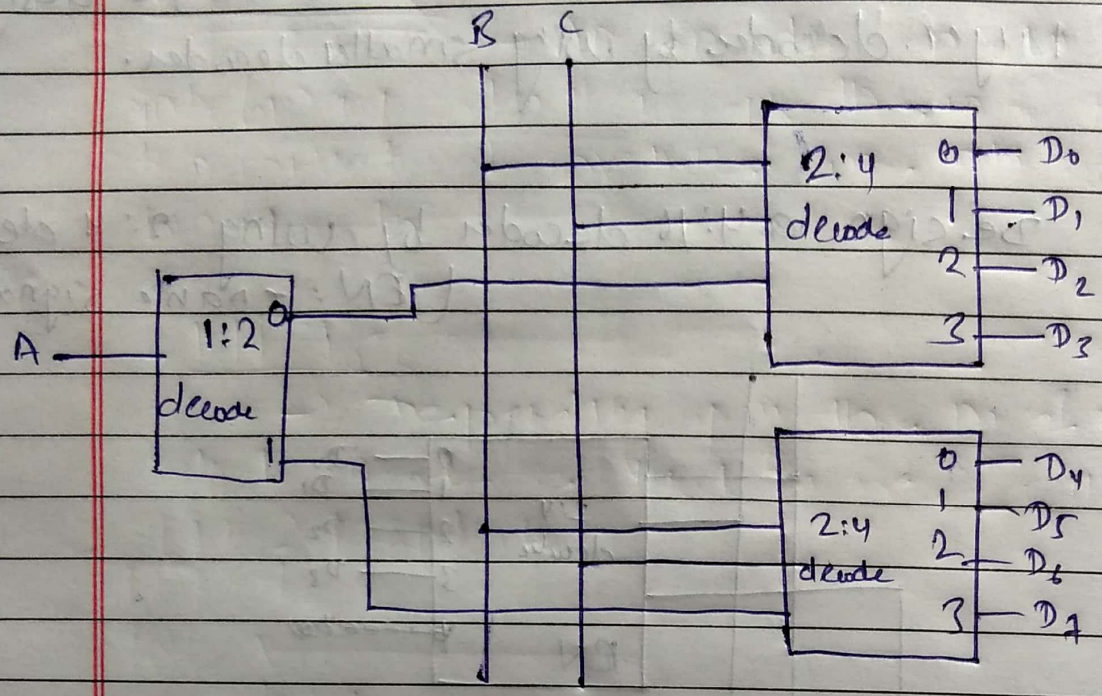
Decoder Tree :- (1 question for sure)

→ A Decoder Tree is used to design larger decoder by using smaller decoder.

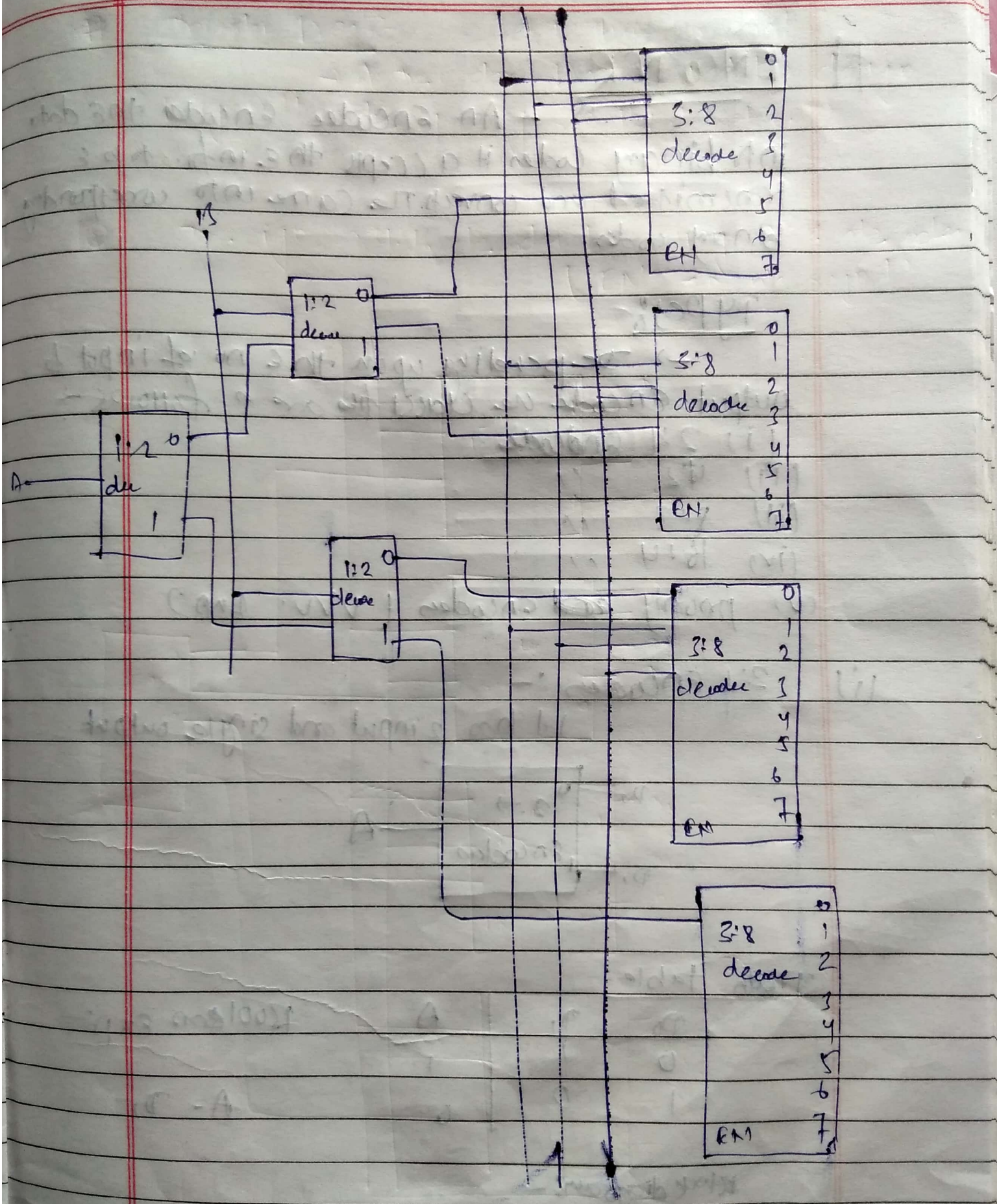
(Q.) Design a 4:16 decoder by using 2:4 decoder. (EN = enable signal)



(9) Design a 3:8 decoder by using 2:4 decoder.



(10) 5:32 by using 3:8 & 1:2


3:8
decode
EN
1:2
dec
A

ENCODER

- An encoder encodes the data into binary codes it accepts the info. to be transmitted and converts the same into corresponding binary codes.

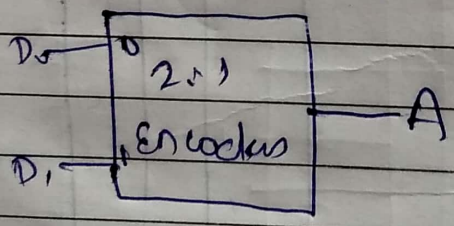
Types:-

→ Depending upon the no of input & output encoder are classified as follows :-

(i) 2:1 Encoder
(ii) 4:2 "
(iii) 8:3 "
(iv) 16:4 "
(v) priority encoders (VVV. Imp)

(i) 2:1 Encoder :-

It has 2 input and single output.



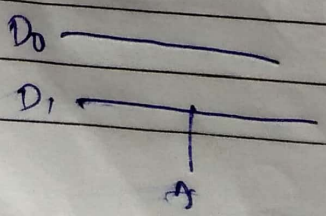
Truth table :-

D_0	D_1	A
0	1	1
1	0	0

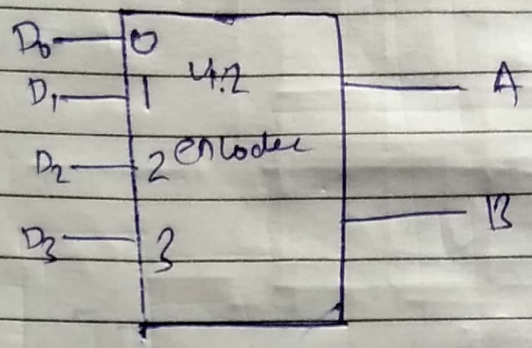
Boolean exp :-

$A = D_1$

Block diagram :-



(IV) 4:2 :- → It has 4 input and 2 output



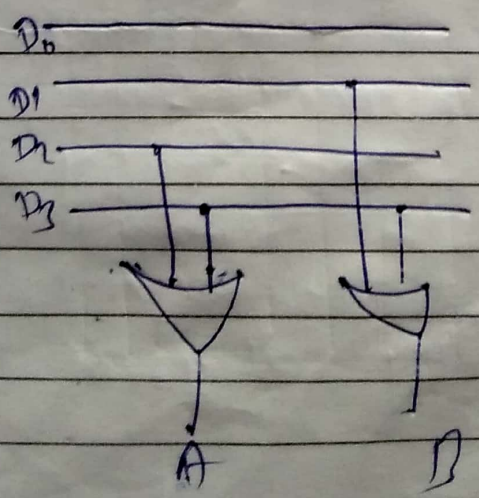
Truth table:-

D_0	D_1	D_2	D_3	A	B
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

Boolean Expression:-

$$A = D_2 + D_3$$
$$B = D_1 + D_3$$

Block diagram:-



PRIORITY ENCODER :-

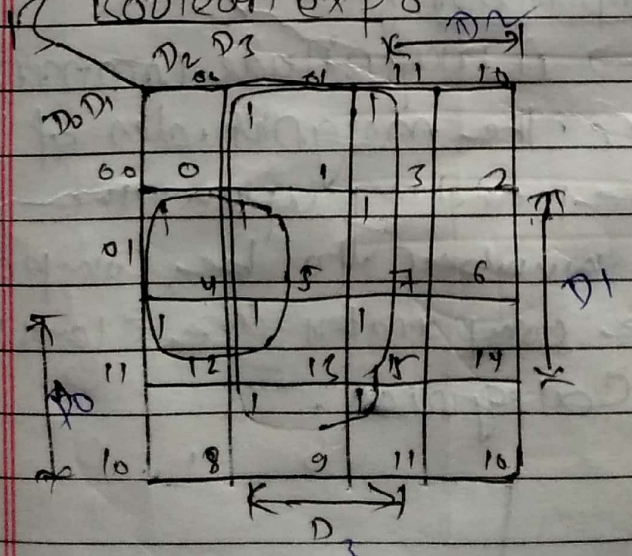
→ priority encoder is used to increase the security of binary data. In this more than one data lines are activated at the same time. It uses priority function that decides the input that will be transmitted first.

In priority encoder, highest priority is given to the input that has maximum decimal digit attached to it and the lowest priority is given to the input having lowest decimal digit. For ex. consider an encoder having 4 inputs

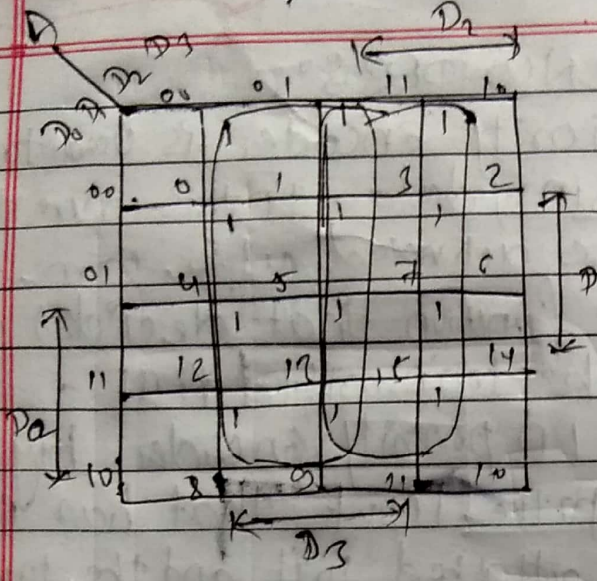
Truth table:-

D_0	D_1	D_2	D_3	A	B
x	x	x	1	1	1
x	x	1	0	1	0
x	1	0	0	0	1
1	0	0	0	0	0
0	0	0	0	x	x

Boolean exp :-

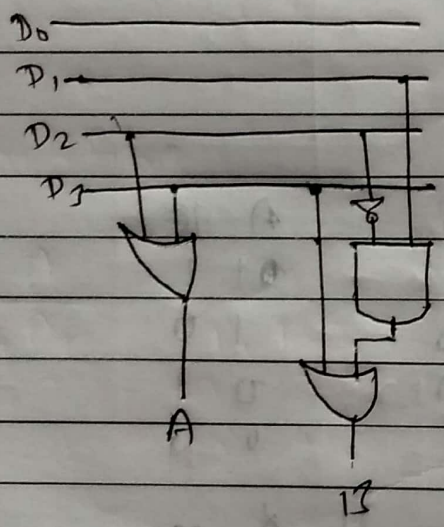


$B = D_3 + \overline{D_2} D_1$



$A = D_3 + D_2$

logic diagram:



Magnitude comparator :-

→ A magnitude comparator is used to compare the magnitudes of two numbers. Depending upon the number of bits in the numbers to be compared, the magnitude comparators are classified into following categories.

(i) 1-bit magnitude comp.
(ii) 2-bit " "
(iii) 3-bit " "
(iv) 4-bit " "

(P) 1-bit magnitude comparator:-
— It compares two numbers each having 1 binary bit.

Truth table:-

A (A_0)	B (B_0)	A=B	A>B	A<B
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

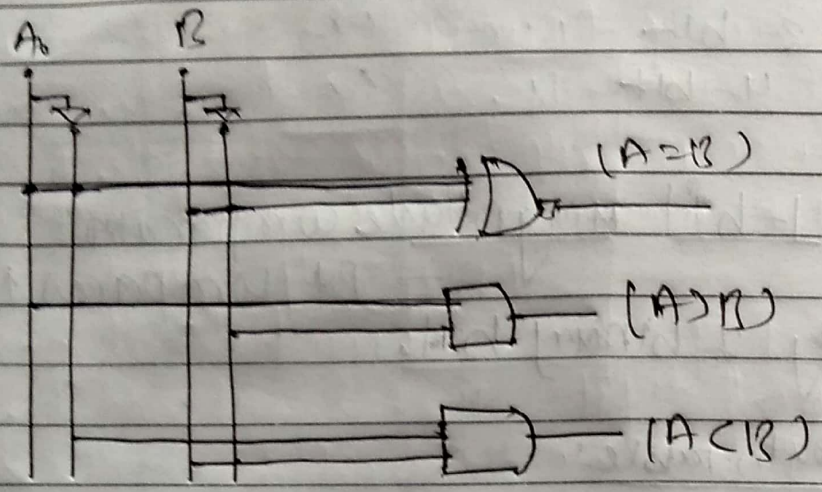
Boolean Expression:-

For (A=B)
$= \overline{A_0}\,\overline{B_0} + A_0 B_0$
$= (A_0 \odot B_0)$

For (A>B)
$= A_0 \overline{B_0}$

For A<B
$= \overline{A_0} B_0$

Logic diagram



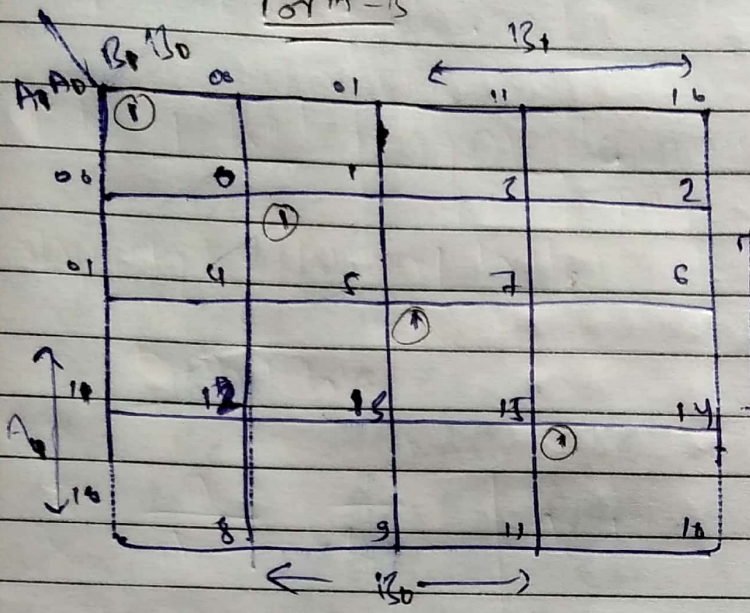
(ii) 2-bit magnitude comparator:-

Truth table:-

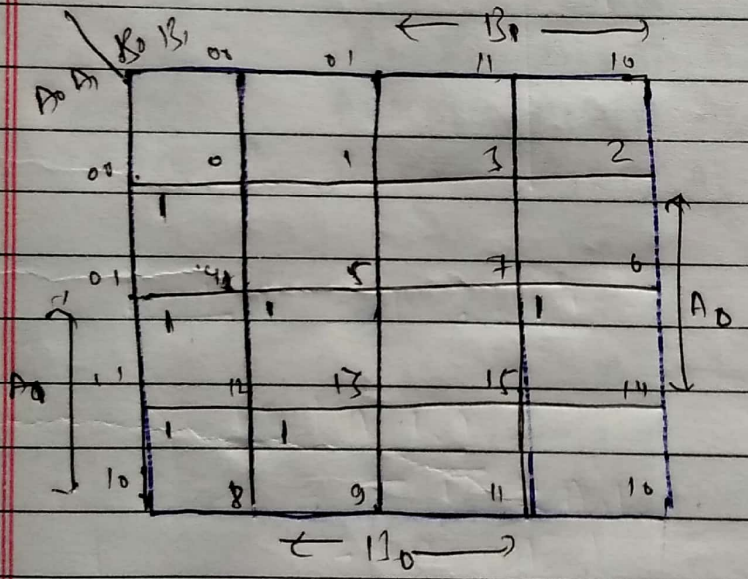
	A		B		A=B	A>B	A<B
	A_1	A_0	B_1	B_0			
0	0	0	0	0	1	0	0
1	0	0	0	1	0	0	1
2	0	0	1	0	0	0	1
3	0	0	1	1	0	0	1
4	0	1	0	0	0	1	0
5	0	1	0	1	1	0	0
6	0	1	1	0	0	0	1
7	0	1	1	1	0	0	1
8	1	0	0	0	0	1	0
9	1	0	0	1	0	1	0
10	1	0	1	0	1	0	0
11	1	0	1	1	0	0	1
12	1	1	0	0	0	1	0

13	1	1	0	1	0	1	0
14	1	1	1	0	0	1	0
15	1	1	1	1	1	0	0

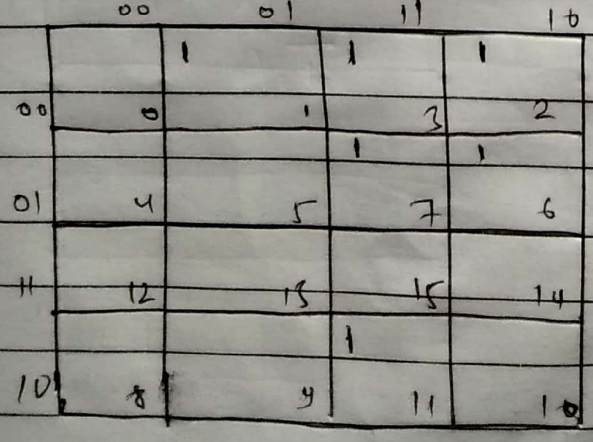
For A = B



For A > B



For A < B



(iii) 4- Bit magnitude comparator :- (M.P)

Truth table :-

A_3	B_3	A_2	B_2	A_1	B_1	A_0	B_0	A=B	A>B	A<B
$A_3 > B_3$		x		x		x		0	1	0
$A_3 < B_3$		x		x		x		0	0	1
$A_3 = B_3$		$A_2 > B_2$		x		x		0	1	
$A_3 = B_3$		$A_2 < B_2$						0	0	1
$A_3 = B_3$		$A_2 = B_2$		$A_1 > B_1$		x		0	1	
$A_3 = B_3$		$A_2 = B_2$		$A_1 < B_1$		x		0	0	
$A_3 = B_3$		$A_2 = B_2$		$A_1 = B_1$		x		0		
$A_3 = B_3$		$A_2 = B_2$		$A_1 = B_1$		$A_0 > B_0$		0		
$A_3 = B_3$		$A_2 = B_2$		$A_1 = B_1$		$A_0 < B_0$		0		
$A_3 = B_3$		$A_2 = B_2$		$A_1 = B_1$		$A_0 = B_0$		1		

(iii) 4-bit magnitude comparator (M.Dup)

Truth table

A_3B_3	A_2B_2	A_1B_1	A_0B_0	A=B	A>B	A<B
$A_3>B_3$	x	x	x	0	1	0
$A_3<B_3$	x	x	x	0	0	1
				0	1	0
				0	0	1
				0	1	0
				0	0	1
				0	1	0
				0	0	1
				1	0	0

Case - I

when the two most significant bits are equal
(i.e $A_3=B_3=1$ or $A_3=B_3=0$)

$$E_3 = A_3B_3 + \overline{A}\overline{B}_3$$
$$E_3 = (A_3 \odot B_3)$$

Case - II

when the next two significant bits are equal
($A_2=B_2=1$ or $A_2=B_2=0$)

$$E_2 = A_2B_2 + \overline{A}\overline{B}_2$$
$$E_2 = (A_2 \odot B_2)$$

Case III

when the next three bit are equal

$(A_1 = B_1 = 1$ or $A_1 = B_1 = 0)$

$$E_1 = A_1 B_1 + \overline{A} \overline{B}_1$$

$$E_1 = (A_1 \odot B_1)$$

Case IV

when least significant bit are equal

$(A_0 = B_0 = 1$ or $A_0 = B_0 = 0)$

$$E_0 = A_0 B_0 + \overline{A} \overline{B}_0$$

$$E_0 = (A_0 \odot B_0)$$

∴ Boolean Expression $E_0 (A = B)$

$$E = E_3 E_2 E_1 E_0$$

∴ Boolean Expression for $(A > B)$

$$= A_3 \overline{B}_3 + E_3 A_2 \overline{B}_2 + E_3 E_2 A_1 \overline{B}_1 + E_3 E_2 E_1 A_0 \overline{B}_0$$

∴ Boolean Expression for $(A < B)$:-

$$= \overline{A}_3 B_3 + E_3 \overline{A}_2 B_2 + E_3 E_2 \overline{A}_1 B_1 + E_3 E_2 E_1 \overline{A}_0 B_0$$