Final Exam

1). Definition combinational Logic ?

Ans: combinational logic is a type of logic where the output depends only on the current inputs, not on any past inputs.

2) Definition sequential Logic ?

Ans: Sequential logic is a type of logic used in electronics and computers where the output depends not only on the current inputs but also on past inputs.

3) Difference between combinational and sequential logic?

Ans:

Combinational Logic	Sequential Logic
The output depends only on the current inputs.	The output depends on both current and past inputs.
It has no memory of past inputs.	It has memory and remembers what happened before.
Example: A calculator that gives a result based only on the numbers you enter right now.	Example: A light switch that remembers if it was on or off earlier and changes based on that memory.

*

Binary System 1+1 = 10

Boolean Algebra 1+1 = 1

4) Design Procedure:

i) The problem is stated.

ii) The number of available input variables and required output variables is determined.

iii) The input and output variables are assigned letter symbols.

iv) The truth table that defines the required relationships between inputs and outputs is derive

v) The simplified Boolean function for each output is obtained.

vi) The logic diagram is drawn.

* Adders -

1) Half adder : 2 bit (sum and carry)

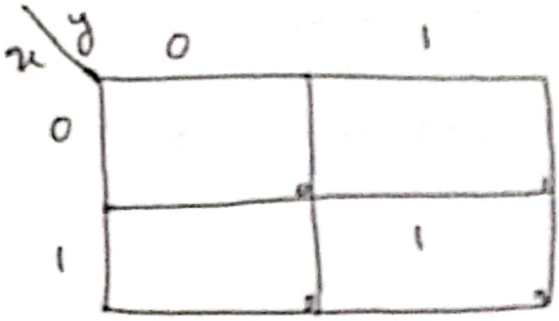
2) Full adder : 3 bit (sum and carry)

* Question : Half adder design ?

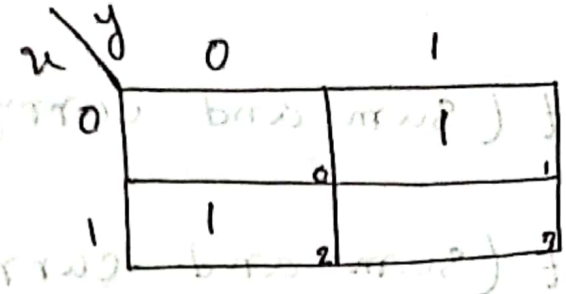
Ans: This truth table is shown below:

	x	y	c	s
0	0	0	0	0
1	0	1	0	1
2	1	0	0	1
3	1	1	1	0

simplify -

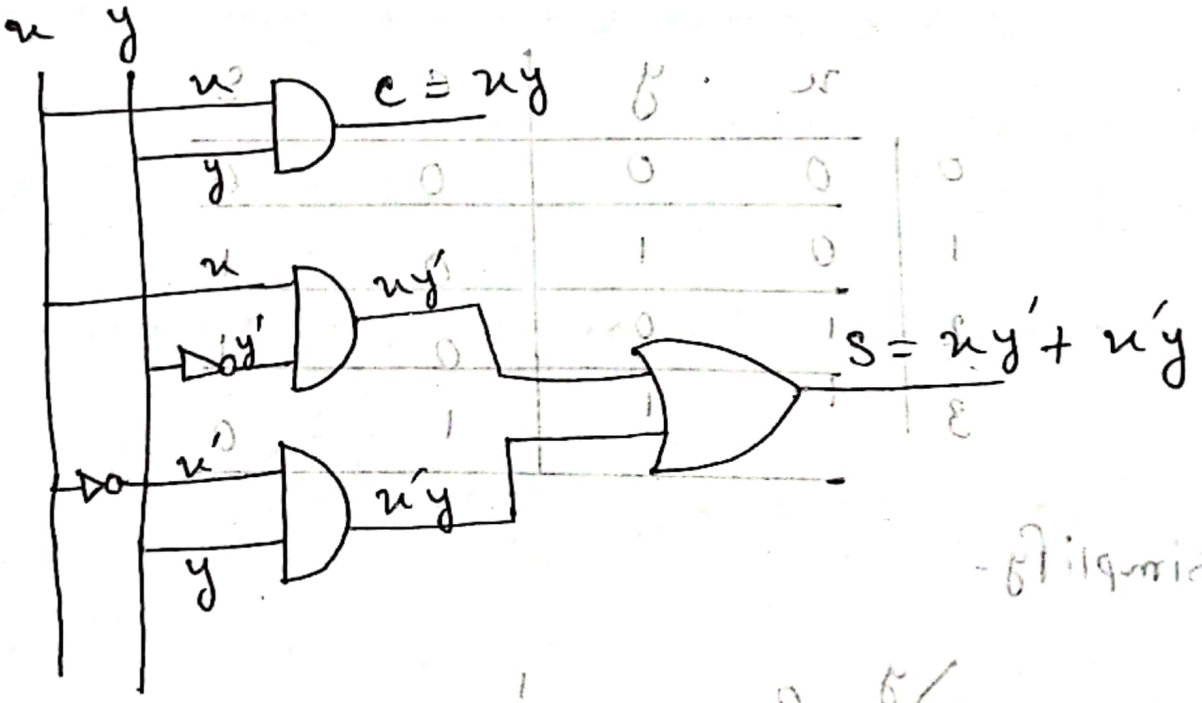


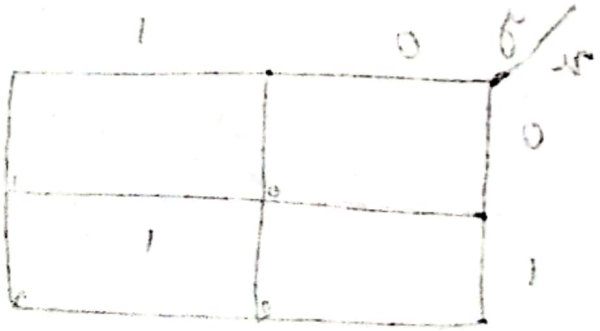
$c = xy$



$$S = xy' + x'y$$

Block diagram -



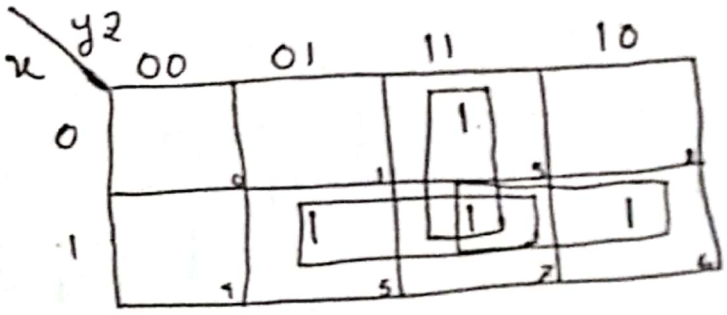


* Question: Full adder design?

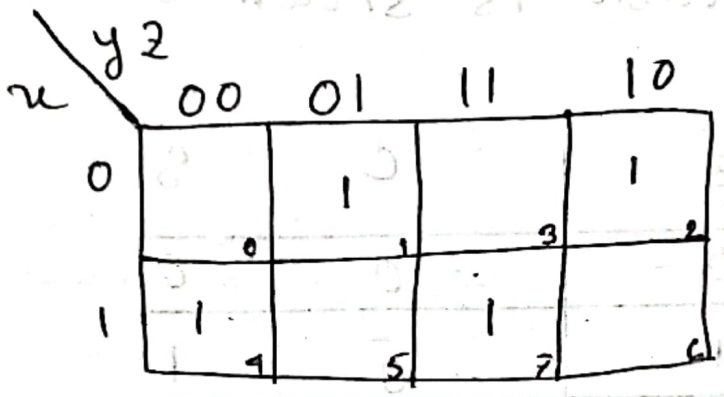
Ans: This truth table is shown below:

	x	y	z	c	s
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	1

simplify -

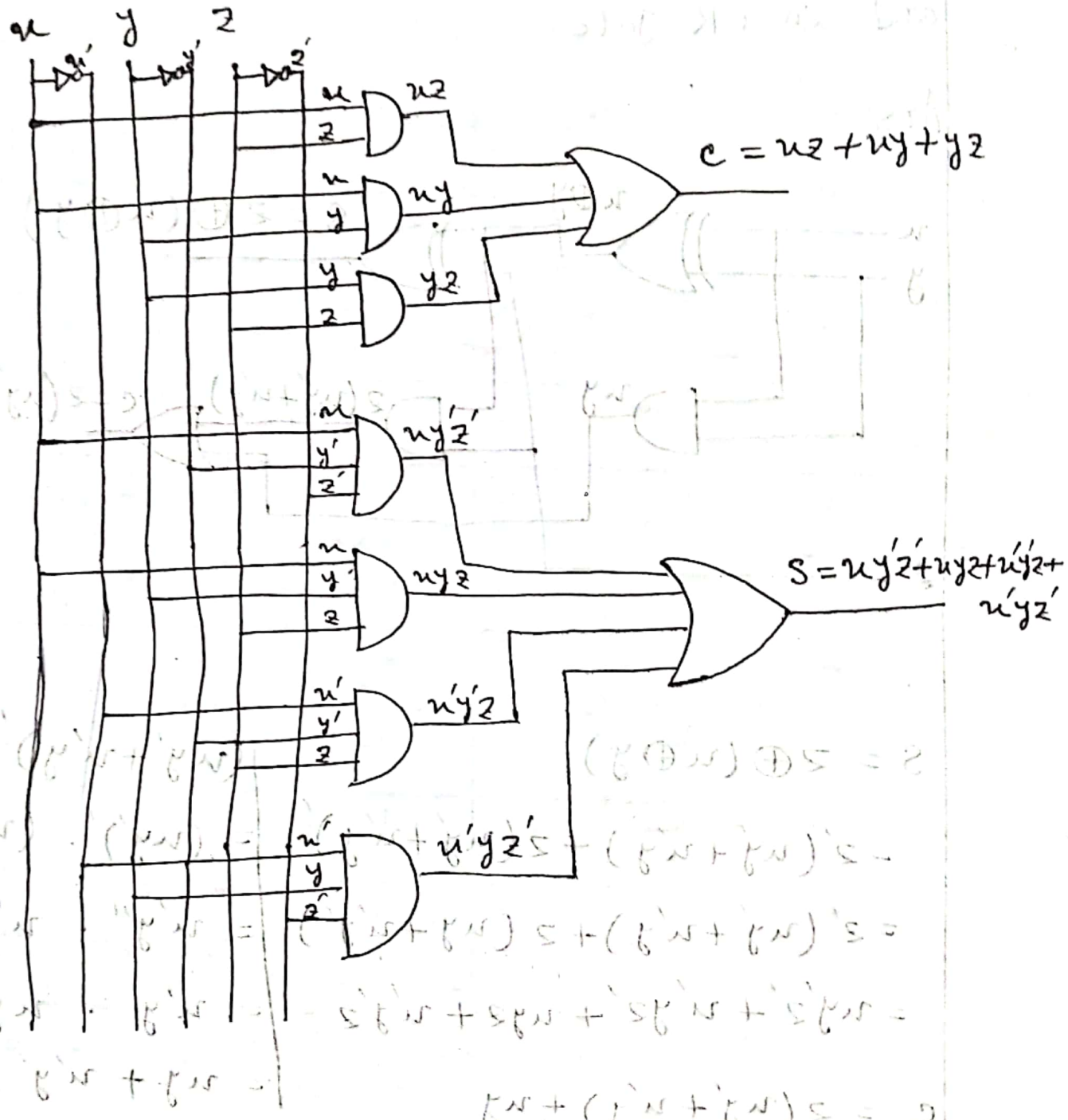


$$c = xz + xy + yz$$



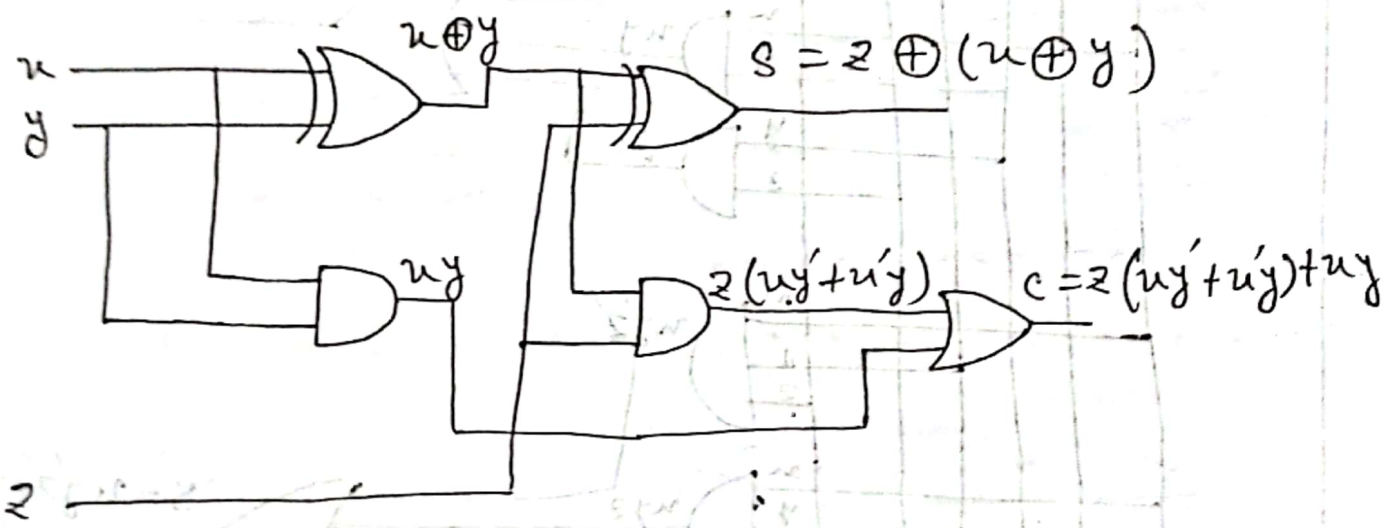
$$S = xy'z' + xyz + x'y'z + x'yz'$$

Block diagram -



* Implementation of a full adder with two half-adder and an OR gate:

Ans:



$S = z \oplus (x \oplus y)$

$= z'(xy' + x'y) + z(xy' + x'y)'$

$= z'(xy' + x'y) + z(xy + x'y')$

$= xy'z' + x'yz' + xyz + x'y'z$

$C = z(xy' + x'y) + xy$

$= xy'z + x'yz + xy$

$(xy' + x'y)'$

$= (xy')' \cdot (x'y)'$

$= x'y'' \cdot x''y'$

$= x'y \cdot xy'$

$= xy + x'y'$

* Code conversion -

Truth table for code-conversion example:

	Input BCD				Output Excess-3 code			
	A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

Ref

8 4 2 1
0 0 0 1

1 + 3 = 4

8 4 2 1
0 0 1 1

0 1 0 0

Simplify -

AB \ CD	00	01	11	10
00	0	1	3	2
01	4	1 (5)	1 (7)	1 (6)
11	X (12)	X (13)	X (15)	X (14)
10	1 (8)	1 (9)	X (11)	X (10)

AB \ CD	00	01	11	10
00				
01		1	1	1
11	X	X	X	X
10	1	1	X	X

$$W = A + BD + Bc$$

$$= A + Bc + BD = A + B(c + D)$$

AB \ CD	00	01	11	10
00	0	1 (1)	1 (3)	1 (2)
01	1 (4)	5	7	6
11	X (12)	X (13)	X (15)	X (14)
10	8	1 (9)	X (11)	X (10)

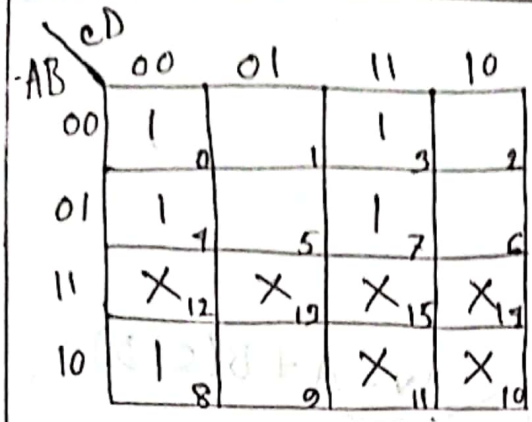
AB \ CD	00	01	11	10
00		1	1	1
01	1			
11	X	X	X	X
10		1	X	X

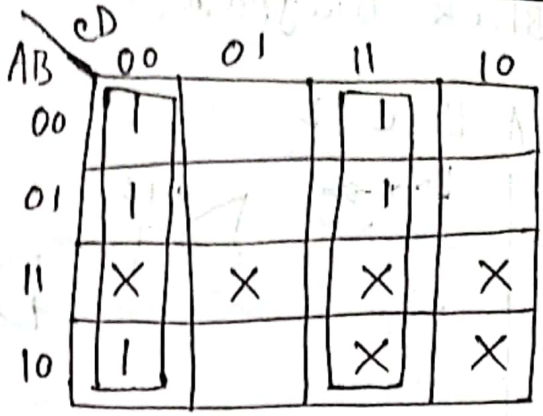
$$X = Bc'D' + B'D + B'c$$

$$= B'c + B'D + Bc'D'$$

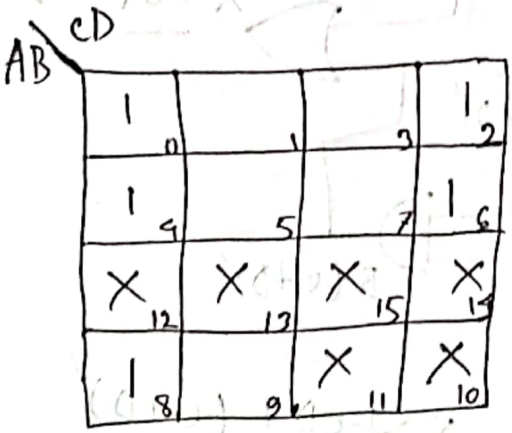
$$= B'(c + D) + Bc'D'$$

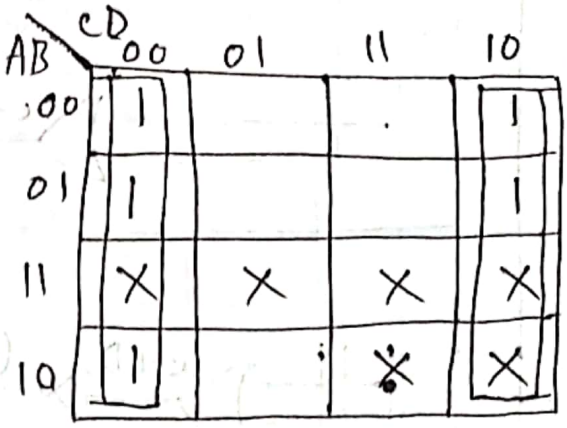
$$= B'(c + D) + B(c + D)'$$





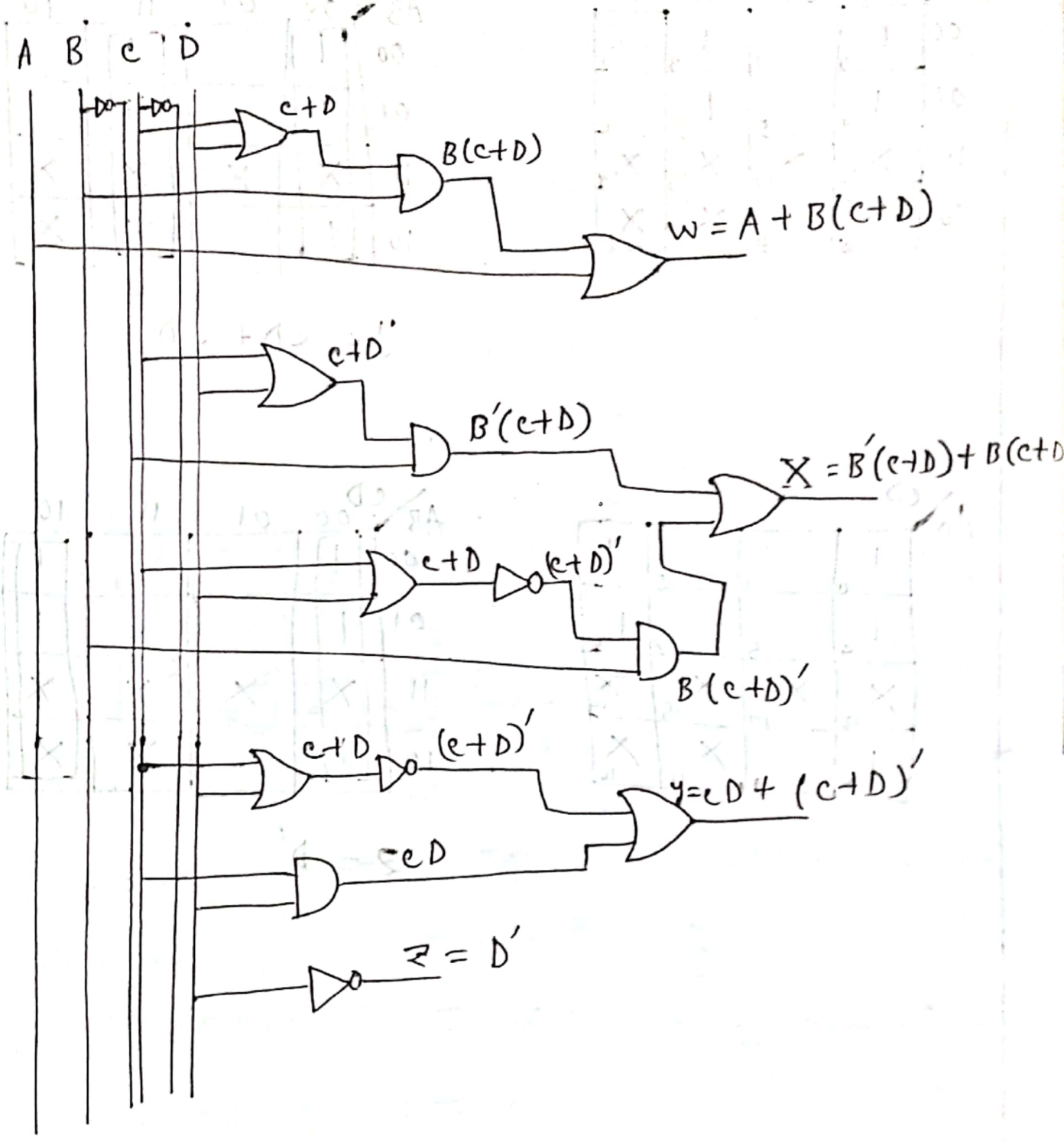
$y = cD + c'D'$





$z = D'$

Block Diagram:-



Combinational Logic with MSI and LSI -

MSI = Medium Scale Integration

LSI = Large Scale Integration

Decoder -

n Input 2^n output

Truth table of a 3-to-8 line decoder -

x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Simplify -

$D_0 = x'y'z'$

$D_1 = x'y'z$

$D_2 = x'yz'$

$D_3 = x'yz$

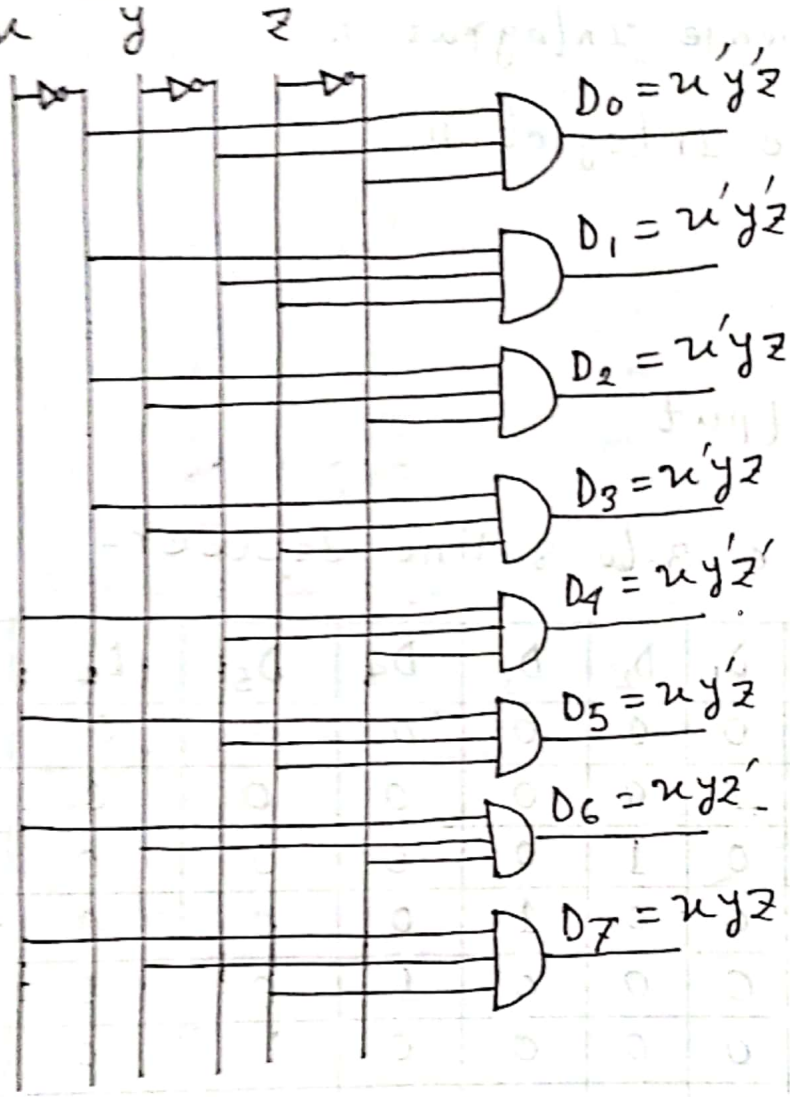
$D_4 = xy'z'$

$D_5 = xy'z$

$D_6 = xyz'$

$D_7 = xyz$

Diagram -



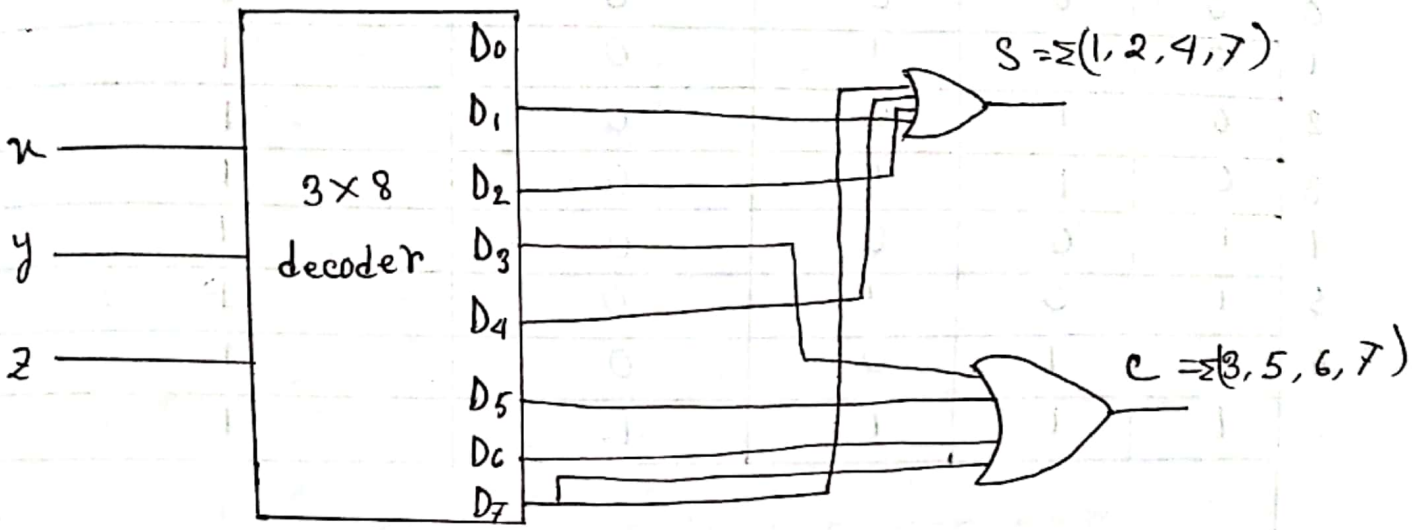
Combinational Logic Implementation -

Example -

$S(x, y, z) = \Sigma(1, 2, 4, 7)$

$c(x, y, z) = \Sigma(3, 5, 6, 7)$

Σ = Sum of minter



Implementation of a full adder with a decoder

* Implement the function $F(x,y,z) = xy + yz + zx$ and $H(x,y,z) = x + y + z + xyz$.

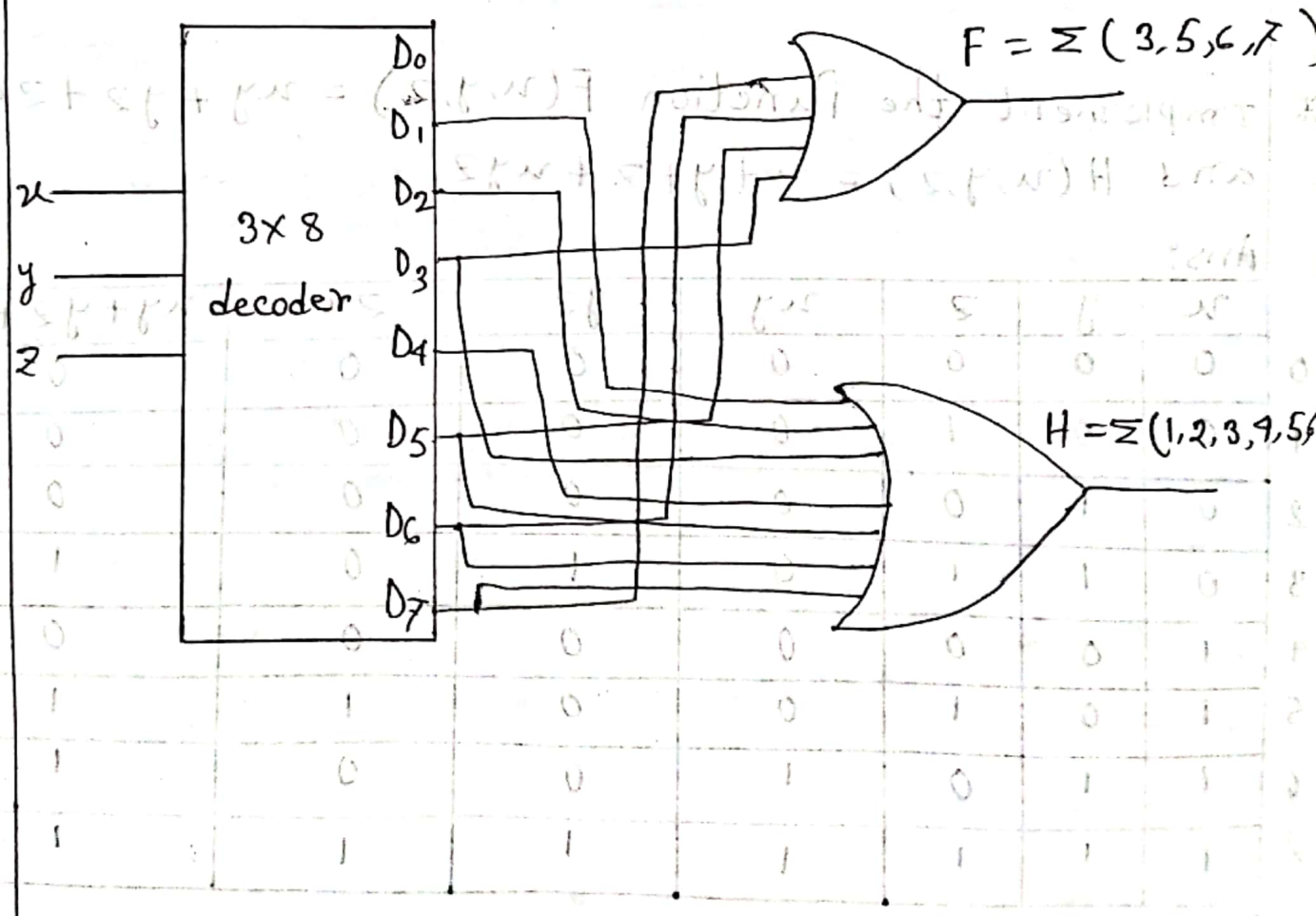
Ans:

	x	y	z	xy	yz	zx	xy+yz+zx
0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0
2	0	1	0	0	0	0	0
3	0	1	1	0	1	0	1
4	1	0	0	0	0	0	0
5	1	0	1	0	0	1	1
6	1	1	0	1	0	0	1
7	1	1	1	1	1	1	1

$F = \Sigma(3, 5, 6, 7)$

	x	y	z	xyz	x+y+z+xyz
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	0	1
4	1	0	0	0	1
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

$H(x,y,z) = \Sigma(1,2,3,4,5,6,7)$



*

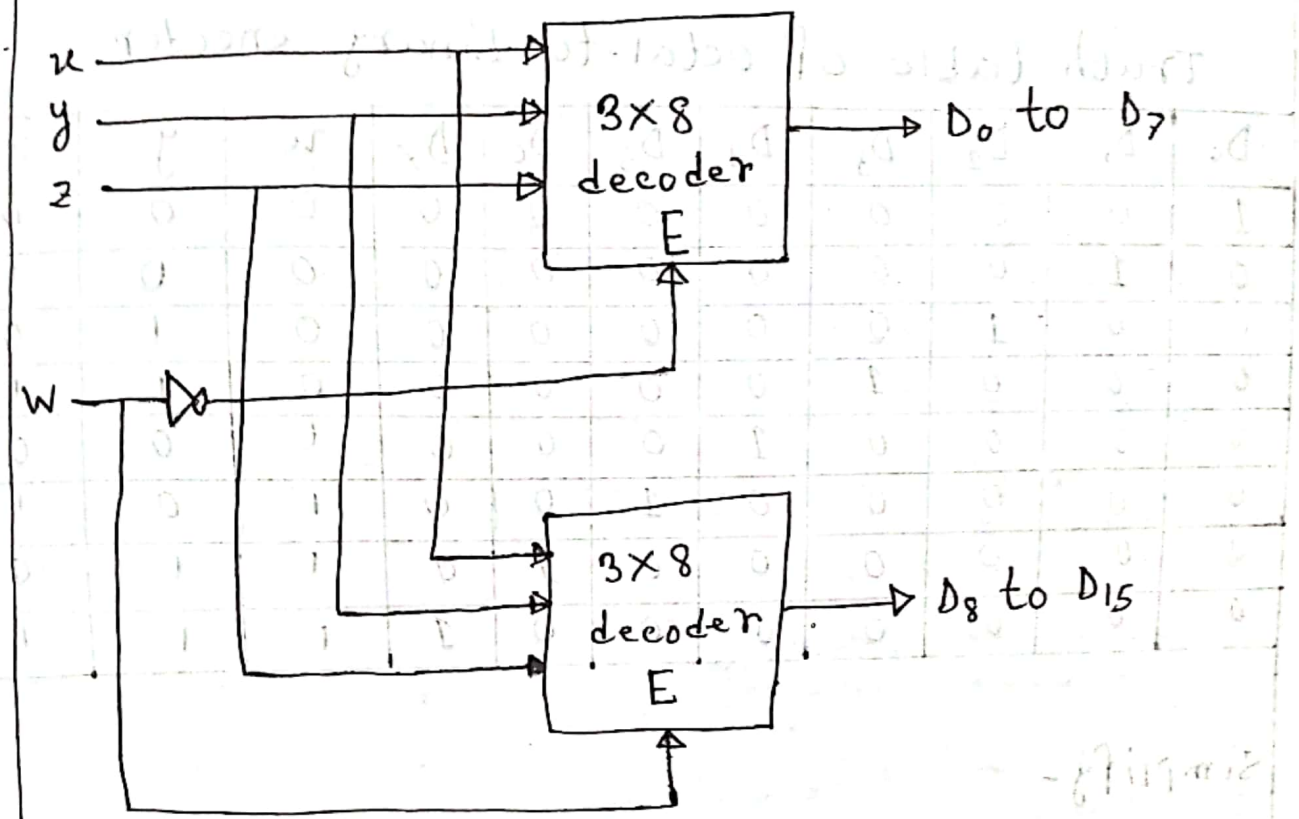


Figure: 4×16 decoder constructed with two 3×8 decoders

* **Encoder:** 2^n Input n output

Truth table of octal-to-binary encoder

D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

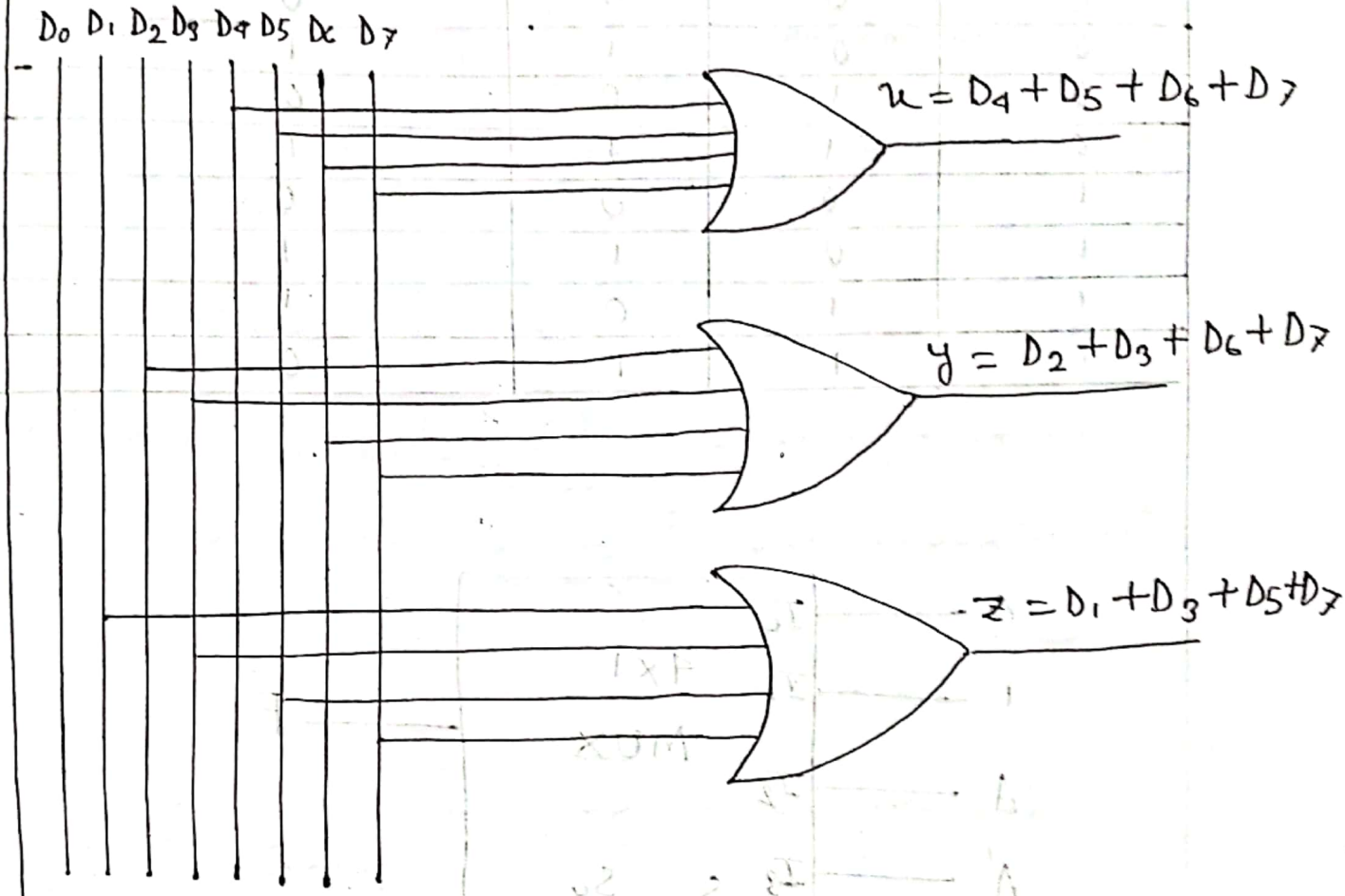
Simplify-

$$x = D_4 + D_5 + D_6 + D_7$$

$$y = D_2 + D_3 + D_6 + D_7$$

$$z = D_1 + D_3 + D_5 + D_7$$

Diagram -

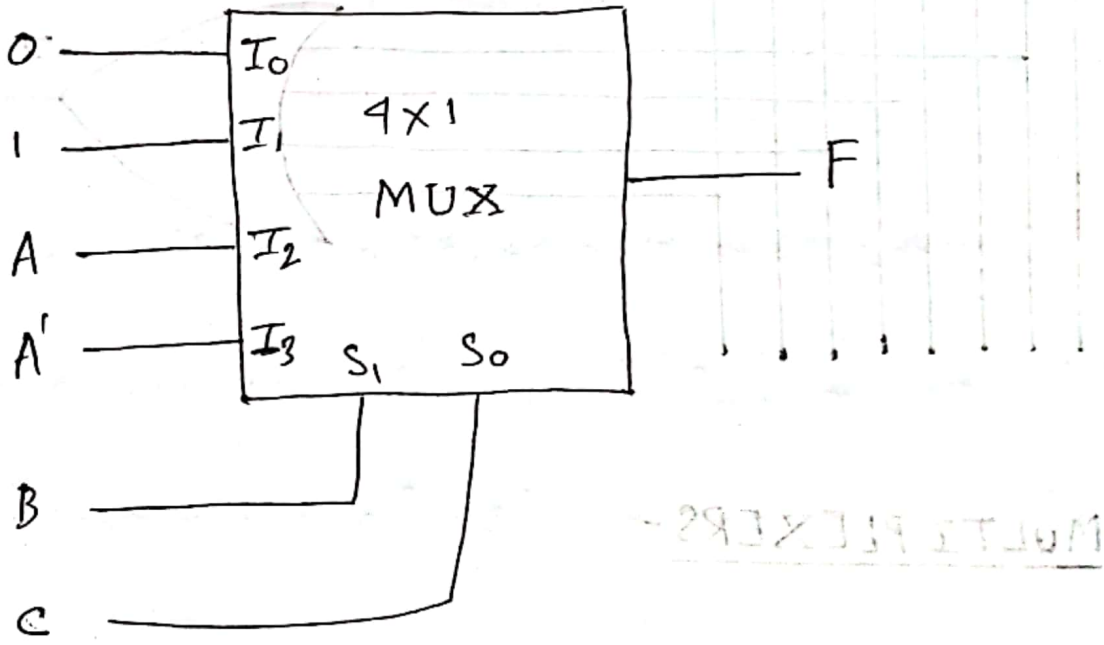


* MULTIPLEXERS -

$F(A, B, C) = \Sigma(1, 3, 5, 6)$

Truth table-

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0



Multiplexer implementation

	I_0	I_1	I_2	I_3
A'	0	(1)	2	(3)
	4	(5)	(6)	7
	0	1	A	A'

Implementation table

Fig - Implementing $F(A, B, C) = \Sigma(1, 3, 5, 6)$ with a multiplexer

* Multiplexing means transmitting a large number of information units over a smaller number of channels or lines.

* An encoder is a digital function that produces a reverse operation from that of a decoder. An encoder has 2^n input lines and n output lines.

* A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2^n unique output lines.