(4) Simplify the following 5 variable expression using K-map →

1) $T = f(a,b,c,d,e) = \sum(0,2,8,10,16,18,24,26)$

Sol^n

ab \ cde	000	001	011	010	100	101	111	110
00	0 1	1 0	3 0	2 1	4 0	5 0	7 0	6 0
01	8 1	9 0	11 0	10 1	12 0	13 0	15 0	14 0
11	24 1	25 0	27 0	26 1	28 0	29 0	31 0	30 0
10	16 1	17 0	19 0	18 1	20 0	21 0	23 0	22 0

$T = \bar{c}\bar{e}$

2) $R = f(v,w,x,y,z) = \sum(5,7,13,15,21,23,29,31)$

Sol^n

vw \ xyz	000	001	011	010	100	101	111	110
00	0	1 0	3 0	2 0	4 0	5 1	7 1	6 0
01	8 0	9 0	11 0	10 0	12 0	13 1	15 1	14 0
11	24 0	25 0	27 0	26 0	28 0	29 1	31 1	30 0
10	16 0	17 0	19 0	18 0	20 0	21 1	23 1	22 0

$R = xz$

3) $w = f(a,b,c,d,e) = \sum(1,3,4,6,9,11,12,14,17,19,30,28,20,22,25)$

Sol^n

ab \ cde	000	001	011	010	100	101	111	110
00	0	1	1	0	1	0	0	1
01	0	1	1	0	1	0	0	1
11	0	1	1	0	1	0	0	1
10	0	1	1	0	1	0	0	1

$w = \bar{c}e + c\bar{e}$

$= c \oplus e$

4) $f = f(w, x, y, z) = \Sigma(4, 5, 6, 7, 9, 11, 13, 15, 25, 27, 29, 31)$

Soln

vw \ xyz	000	001	011	010	100	101	111	110
00	0	0	0	0	1	1	1	0
01	0	1	1	0	0	1	1	0
11	0	1	1	0	0	1	1	0
10	0	0	0	0	0	0	0	0

$$f = w\bar{x}z + \bar{v}\bar{w}x + wxz$$

$$= wz(\bar{x} + x) + \bar{v}\bar{w}x$$

$$f = wz + \bar{v}\bar{w}x = wz(\bar{x} + x) + \bar{v}\bar{w}x.$$

$$f = wz + \bar{v}\bar{w}x$$

Incompletely specified functions (Don't care terms) →

* When an output value is known for every possible combination of input variables the function is said to be completely specified.
+ However, when an output value is not known for every combination of input variable usually because all combinations cannot occur, the function is said to be incompletely specified.
+ This means that the truth table doesnot generate an output value for every possible input combination.
+ The min terms & max terms that are not used as part of the output function are called dont care terms.

Ex: The truth table for binary to Ex-3 BCD is shown below.

Binary

W	X	Y	Z
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

Ex-3 BCD

A	B	C	D
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
Don't care terms			
1	1	1	
1	1		1
1	1		
1		1	1
1		1	
1			1

$A = f(w,x,y,z) = \Sigma(5,6,7,8,9) + \Sigma d(10,11,12,13,14,15)$

wx \ yz	00	01	11	10
00	0	0	0	0
01	0	1	1	1
11	d	d	d	d
10	1	1	d	d

$A = w + xz + xy$

$B = f(w,x,y,z) = \Sigma(1,2,3,4) + \Sigma(10,11,12,13,14,15)$

wx \ yz	00	01	11	10
00	0	1	1	1
01	1	0	0	0
11	d	d	d	d
10	0	1	d	d

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

$B = \bar{w}\bar{x}\bar{z} + \bar{w}xy + x\bar{y}\bar{z} + wz$

$C = f(w,x,y,z) = \Sigma(0,3,4,7,8) + \Sigma(10,11,12,13,14,15)$

wx \ yz	00	01	11	10
00	1	0	1	0
01	1	0	1	0
11	d	d	d	d
10	1	0	d	d

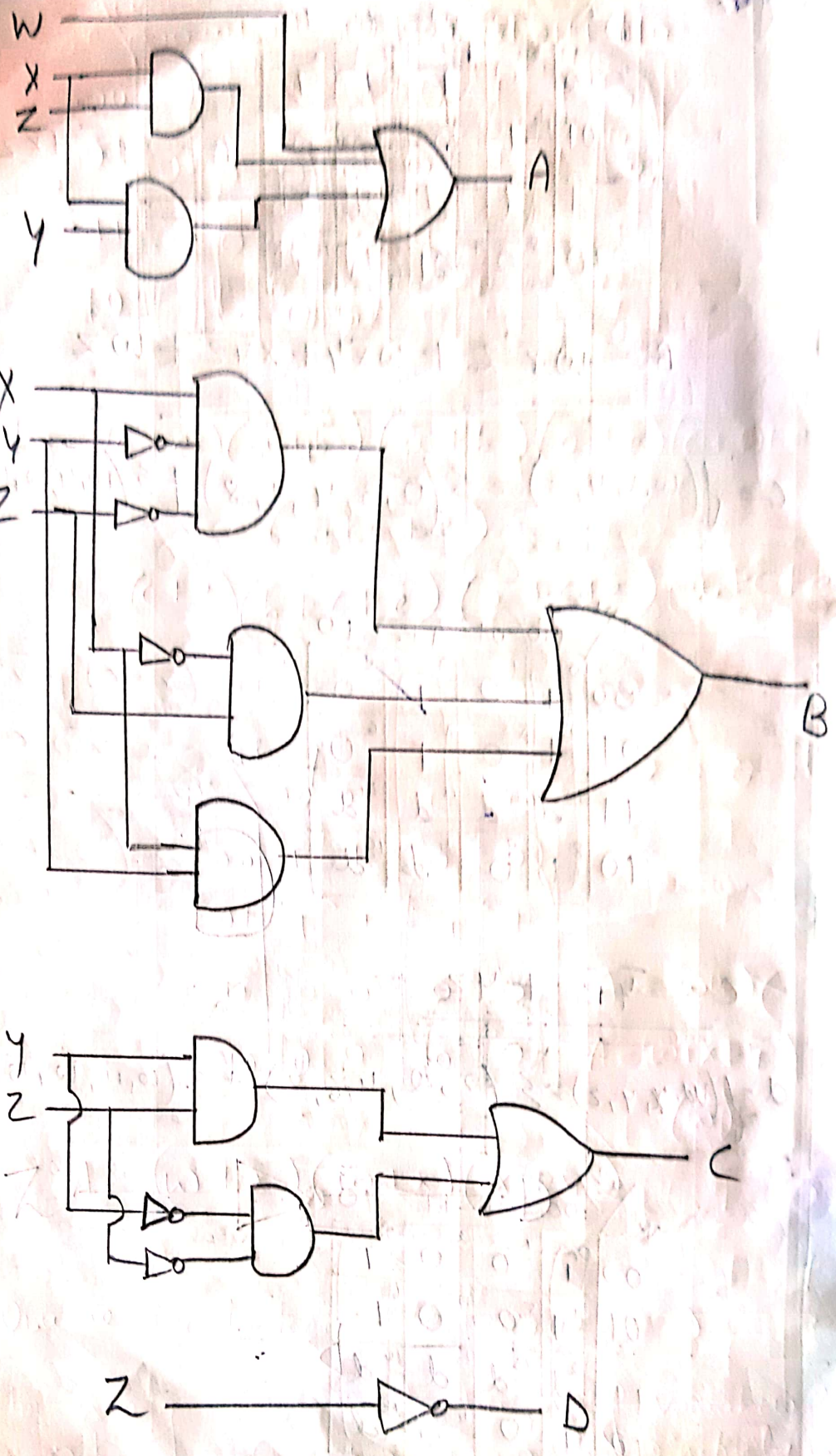
$C = \bar{y}\bar{z} + yz$

$d = f(w,x,y,z) = \Sigma(0,2,4,6,8) + \Sigma(10,11,12,13,14,15)$

wx \ yz	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	d	d	d	d
10	1	0	d	d

$D = \bar{z}$

Circuit diagram →



(4) Simplify the maxterm equation

1) $S = f(x, y, z) = \pi(0, 3, 4, 7)$

x \ yz	00	01	11	10
0	0	1	0	1
1	0	1	0	1

$J = (\bar{y}+\bar{z})(y+z)$

2) $G = f(a,b,c,d) = \pi(0,4,5,7,8,9,11,12,13,15)$

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

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

3) $T = f(w,x,y,z) = \pi(0,4,5,7,8,9,11,12,13,15)$

$\pi(1,3,8,10,12,13,14,15)$

wx \ yz	00	01	11	10
00	1	0	0	1
01	1	1	1	1
11	0	0	0	0
10	0	1	1	0

$T = (w+x+\bar{z})(\bar{w}+z)(\bar{w}+\bar{x})$

~~$T = \bar{w}\bar{x}\bar{y}(w+x)$~~

4) $L = f(a,b,c,d,e) = \pi(0,2,4,6,8,9,10,11,12,14,16,17,18,19,24,25,26,27)$

5) $A = f(w,x,y,z) = \pi(2,3,8,9,10,11,12,13,14,15)$

4)

ab \ cde	000	001	011	010	100	101	111	110
00	0	1	1	0	0	1	1	0
01	0	0	0	0	0	1	1	0
11	0	0	0	0	1	1	1	1
10	0	0	0	0	1	1	1	1

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

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

5)

wx \ yz	00	01	11	10
00	1	1	0	0
01	1	1	1	1
11	0	0	0	0
10	0	0	0	0

$A = (w+x+\bar{y})(\bar{w}+x)(x+\bar{y})$

⑥ Quine Mccluskey Technique ⟹

+ For many applications the no. of variables in the problem is too large to simplify manually using k-maps
+ K-maps although capable of handling any no. of variables become difficult beyond 6

& 7 variables.

+ ∴ Some automatic & computer driven simplification & routine is desirable.

+ The Quine Mccluskey minimization technique is an algorithm that uses the same boolean algebra postulates that were used with k-maps, but in a form suitable for a computer solution.

+ Large k-maps require recognition of groups of terms that may form EPI

+ The larger the map the more difficult this pattern recognition becomes.

+ The Quine Mccluskey approach eliminates the need for such pattern recognition.

---

④ Ex:- Simplify the following using Quine Mccluskey minimization technique.

$$D = f(a,b,c,d) = \Sigma(0,1,2,3,6,7,8,9,14,15)$$

Group	Minterm	variables a	b	c	d
0	0	0	0	0	0 ✓
1	1	0	0	0	1 ✓
	2	0	0	1	0 ✓
	8	1	0	0	0 ✓
2	3	0	0	1	1 ✓
	6	0	1	1	0 ✓
	9	1	0	0	1 ✓

3	7	0	1	1	1
	14	1	1	1	0
4	15	1	1	1	1

Group	minterm	Variables a	b	c	d
0	0,1	0	0	0	—
	0,2	0	0	—	0
	0,8	—	0	0	0
1	1,3	0	0	—	1
	1,9	—	0	0	1
	2,3	0	0	1	—
	2,6	0	—	1	0
	8,9	1	0	0	—
2	3,7	0	—	1	1
	6,7	0	1	1	—
	6,14	—	1	1	0
3	7,15	—	1	1	1
	14,15	1	1	1	—
0	0,1,2,3	0	0	—	—
	0,1,8,9	—	0	0	—
	0,2,1,3	0	0	—	—
	0,8,1,9	—	0	0	—

1	2, 6, 3, 7	0	—	1	—
2	6, ~~7~~, 14, 15	—	1	1	—
	6, 14, 7, 15	—	1	1	—

* All non-checked minterm groups are now considered to be prime Implicants.
+ All of the prime Implecants are formed into a prime Implecant table as shown below.

PI terms	~~Minterms~~ Decimal	Decimal Minterm									
		0	1	2	3	6	7	8	9	14	15
$\bar{a}\bar{b}$	0, 1, 2, 3	x	x	x	x						
$\bar{b}\bar{c}$	0, 1, 8, 9	x	x					(x)	(x)		
$\bar{a}c$	2, 6, 3, 7			x	x	x	x				
bc	6, 7, 14, 15					x	x			(x)	(x)

+ The PI table test each of the minterms contained in the original switching equation across the top of the table.
+ Each PI is listed vertical in 2 forms, PI terms & the decimal list of minterms that makeup the PI.
* Evaluate the PI by circling those minterms that are contained in only one PI. (only one x in a column)
+ These circled minterms represent Essential Prime Implecants (EPI)
+ Minterms $(0, 1, 8, 9) \rightarrow \bar{b}\bar{c}$ & minterms $(6, 7, 14, 15) \rightarrow bc$ are EPI.

$D = \bar{b}\bar{c} + bc + \bar{a}b$ या $D = \bar{b}\bar{c} + bc + \bar{a}c$

19/8/19

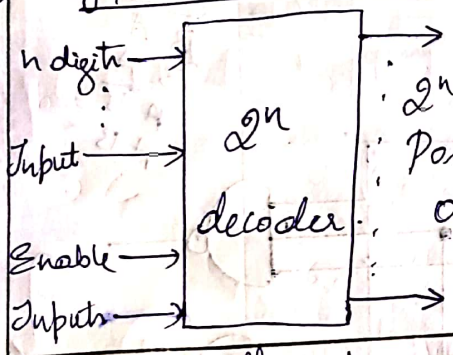
Module - 02

Decoders

(*) **Decoders** → Are a class of combinational logic circuits that convert a set of input variables representing a code into a set of output variables representing a different code.

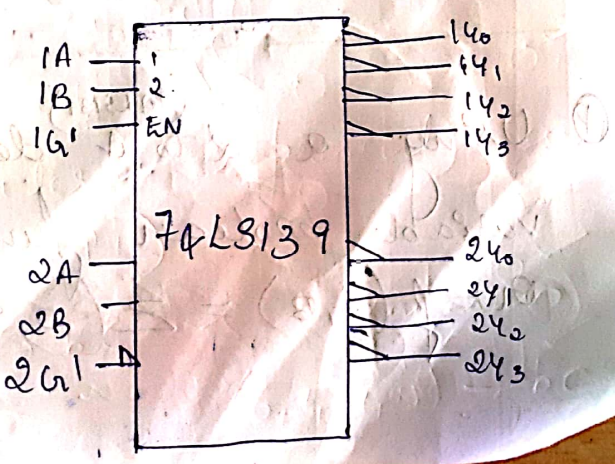
+ Encoded information is presented as 'n' inputs producing 2^n outputs.
+ The 2^n output values can range from 0 to $2^n - 1$.

(*) **Typical decoder** ⇒



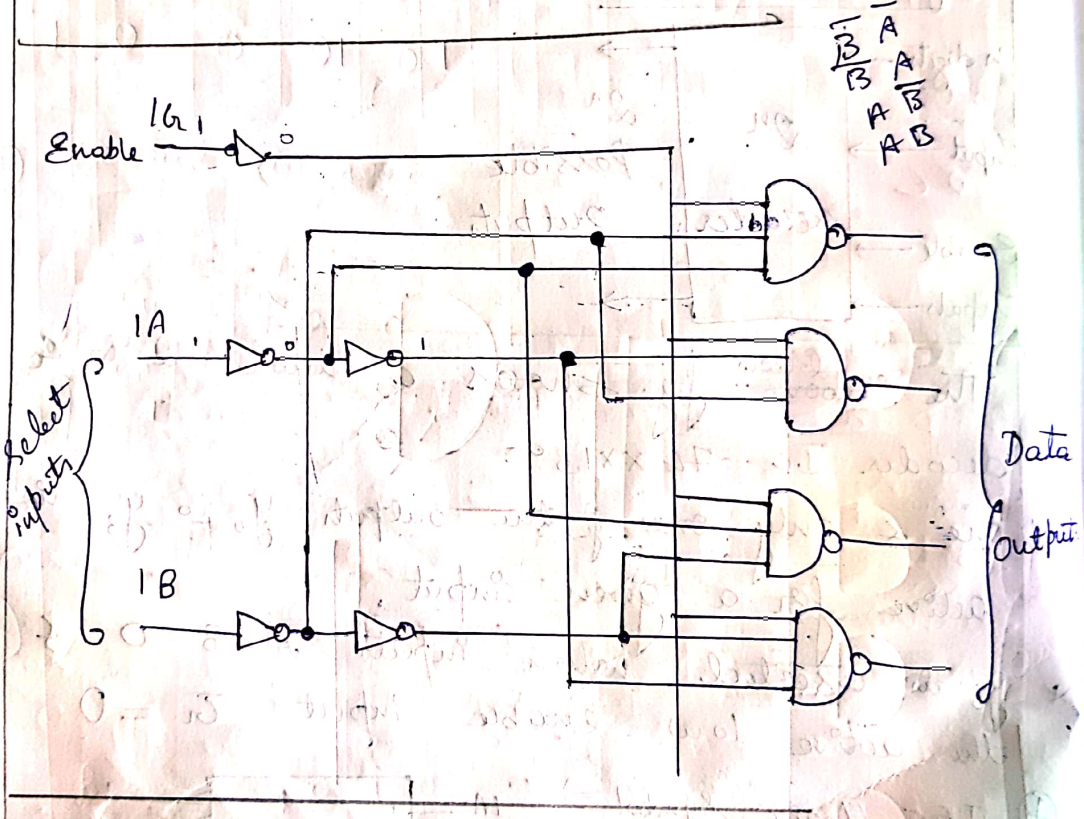
* The ~~above~~ below fig. shows a dual 2 to 4 decoder IC 74xx139.
* one & only one of the outputs, Y_0 to Y_3 is active for a given input.
* Y_0 is asserted when inputs $A = B = 0$ & the active low enable input $G = 0$

(*) **IEEE Logic Symbol** ⇒



Inputs			Outputs			
Enable	Select					
G_1'	B	A	Y_0	Y_1	Y_2	Y_3
H	X	X	H	H	H	H
L	L	L	L	H	H	H
L	L	H	H	L	H	H
L	H	L	H	H	L	H
L	H	H	H	H	H	L

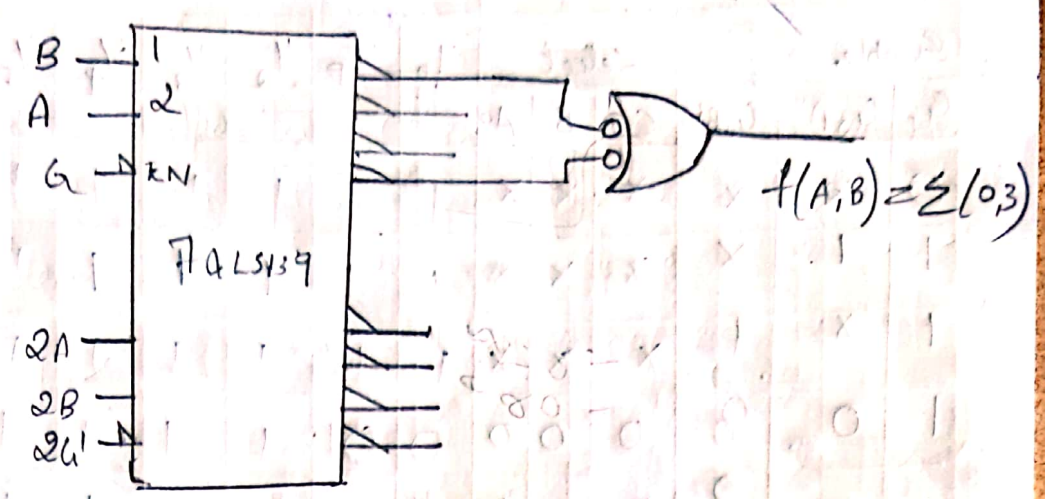
Function Table.



④ Using a 2 to 4 decoder (74LS139) realize a two variable boolean function,

$O_1 = f(A,B) = \Sigma(0,3)$

Soln

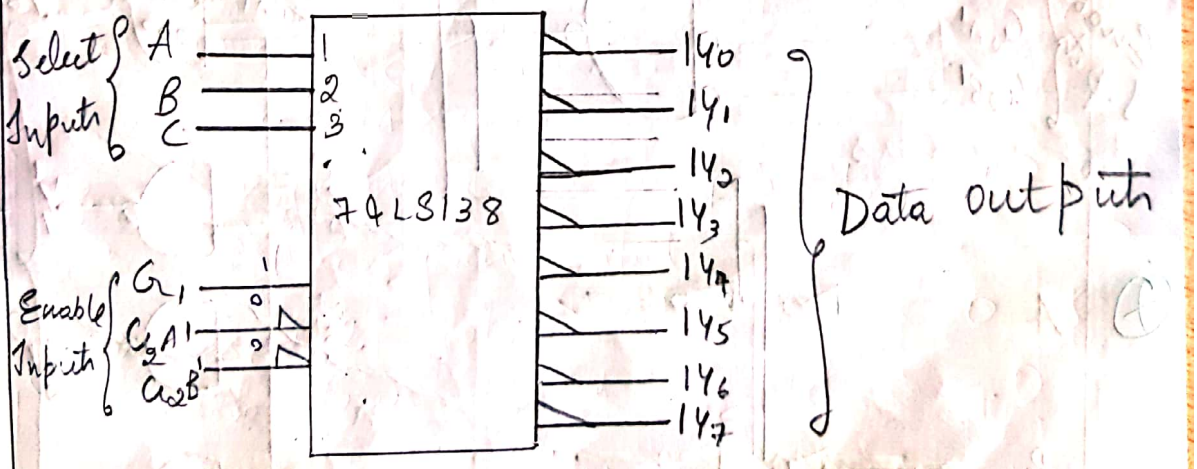


$$Or = f(A,B) = \Sigma(0,3) = \bar{A}\bar{B} + AB$$

(A) The 74xx138 is a 3 to 8 decoder IC. The 3 inputs are decoded to produce one of the 8 outputs.

+ 3 enable inputs are provided all of which must be active before decoding can occur.

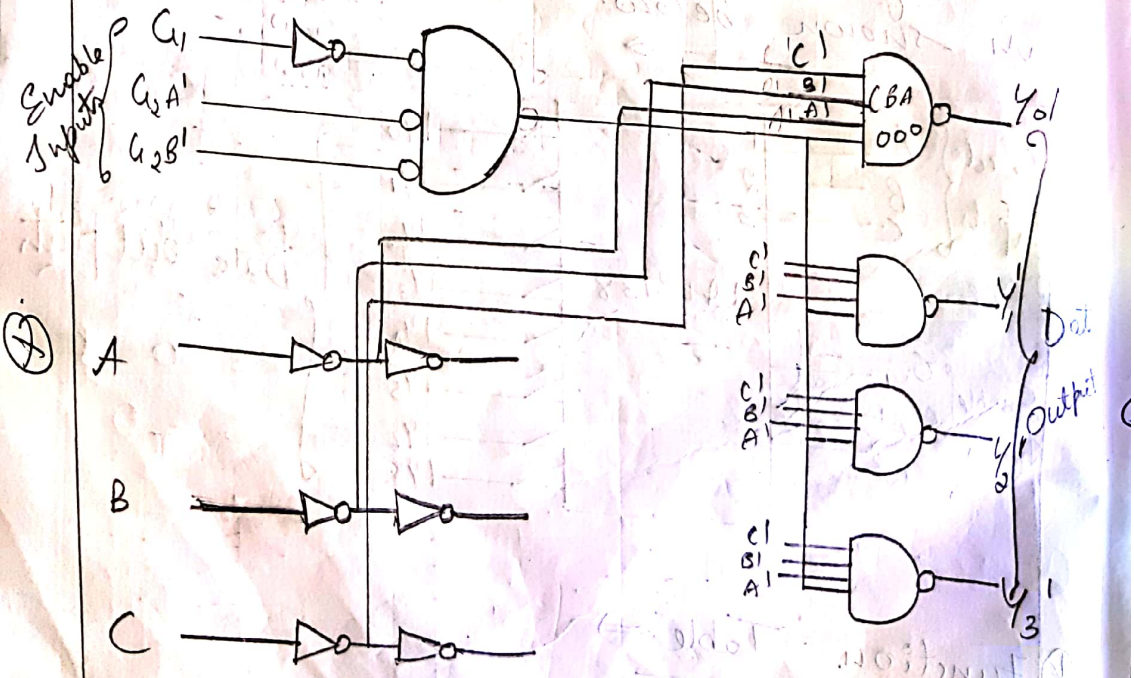
+ The logic circuit, Symbol & function table is shown below.



(*) Function Table ⇒

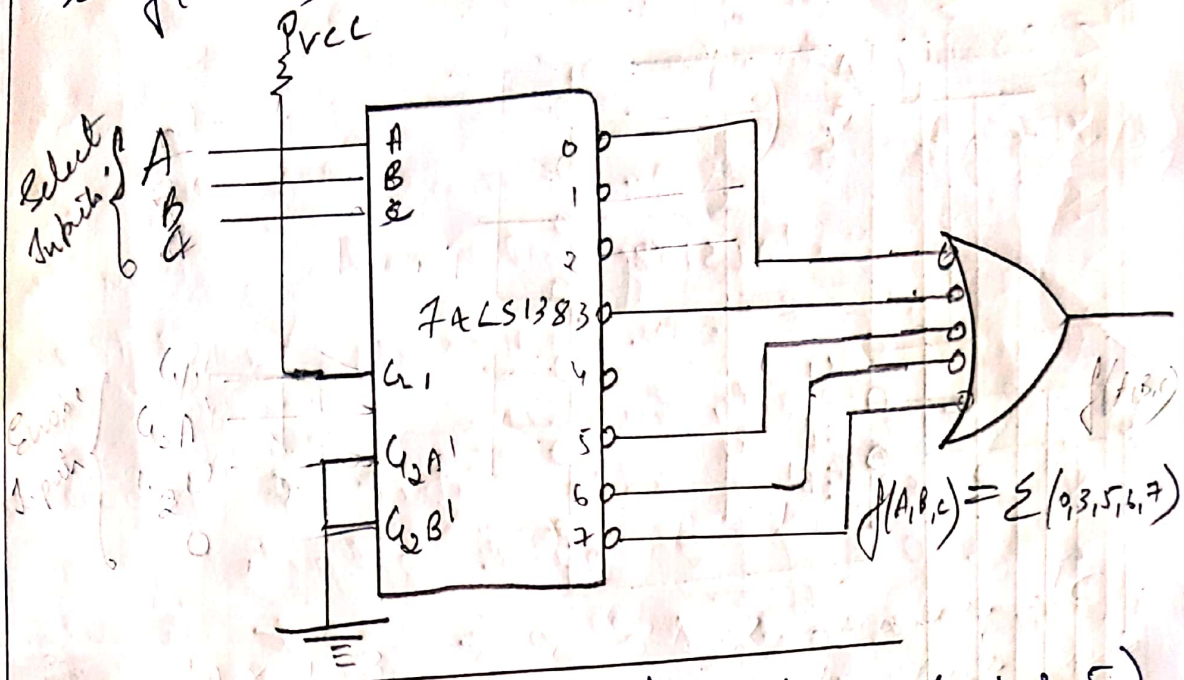
Inputs						Outputs							
Enable			Select			Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
G_1	G_2A'	G_2B'	C	B	A								
0	X	X	X	X	X	1	1	1	1	1	1	1	1
1	1	X	X	X	X	1	1	1	1	1	1	1	1
1	X	1	X	X	X	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	0	0	1	1	1	1	0	1	1	1
1	0	0	1	0	1	1	1	1	1	1	0	1	1

(3) Circuit diagram ⟹

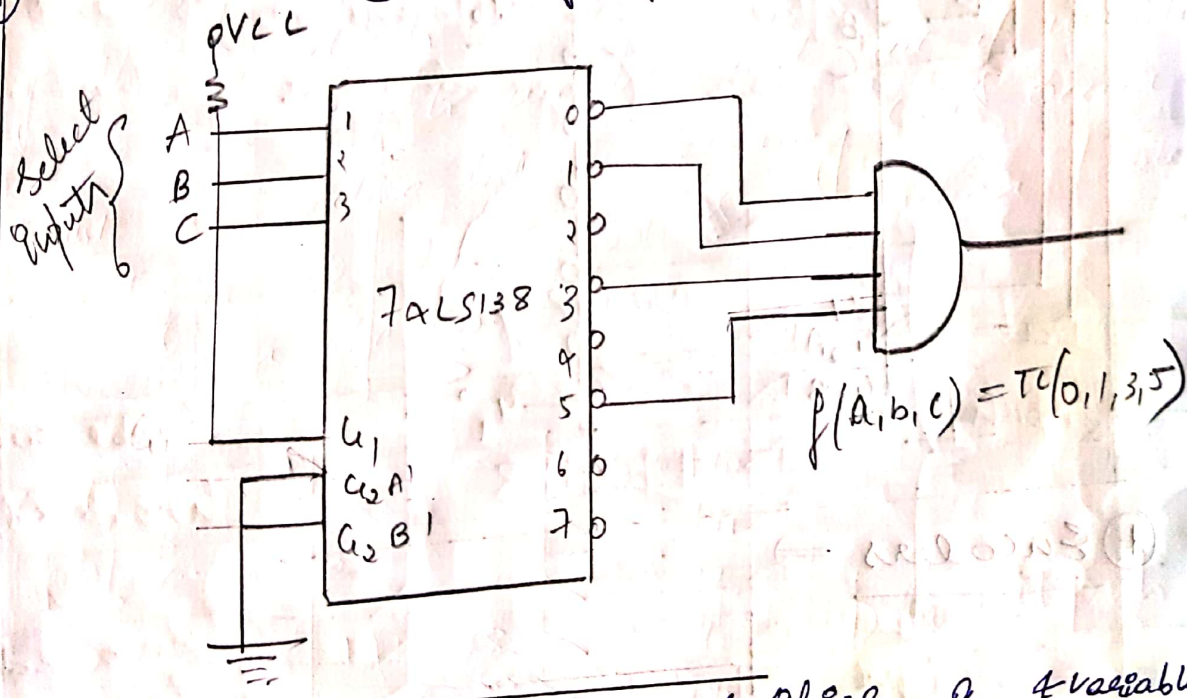


(4) Using a 3 to 8 decoder simplify the following boolean expression

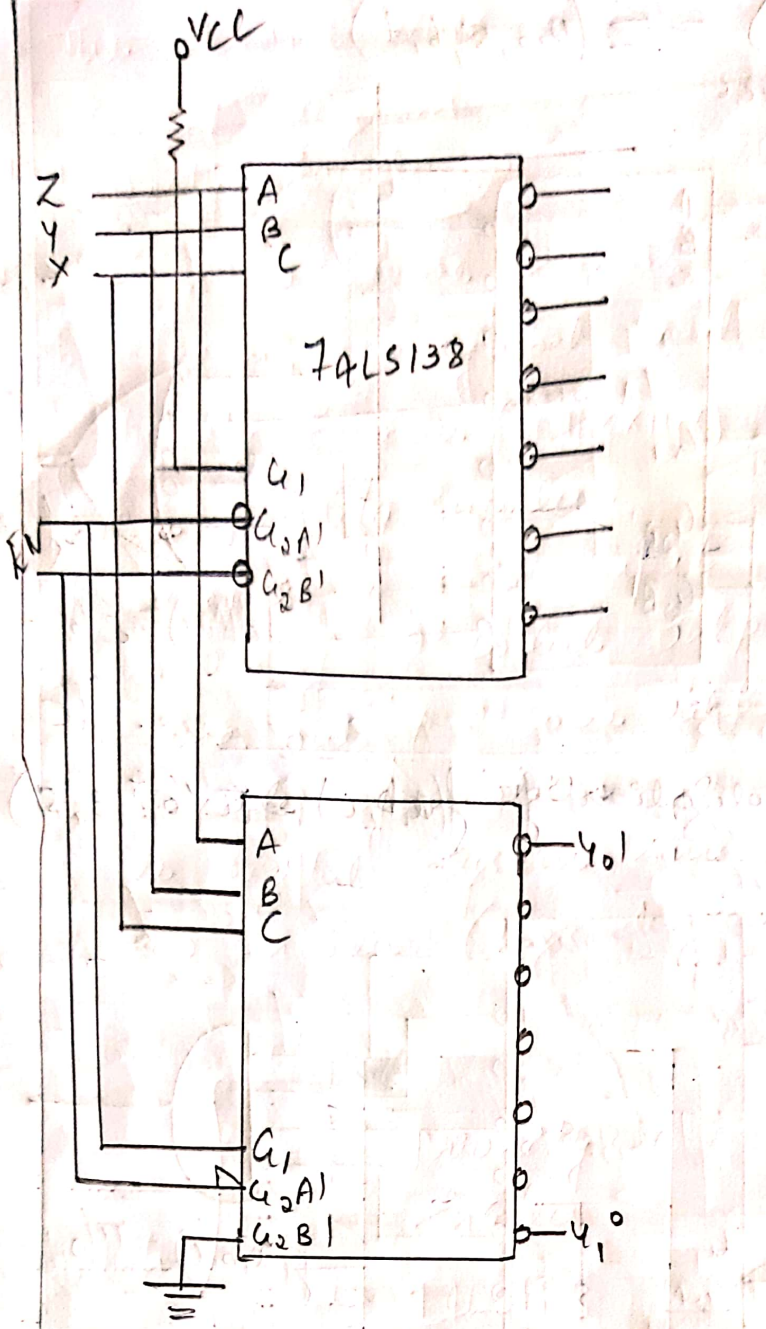
$x = f(a,b,c) = \Sigma(0,3,5,6,7)$



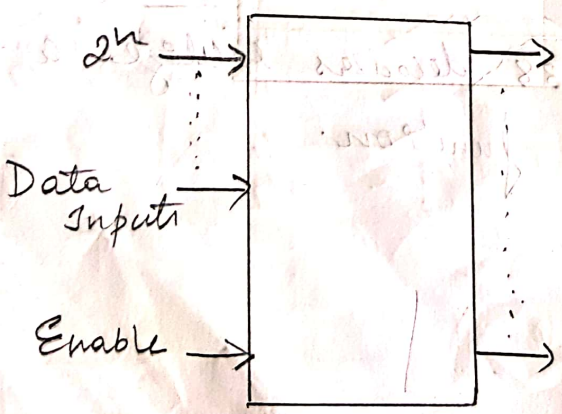
(*) Decoder realization of $f(A,B,C) = \Pi(0,1,3,5)$



(*) Using 2 74XX138 decoders realize a 4 variable multiple output function.



⊛ Encoders ⇒



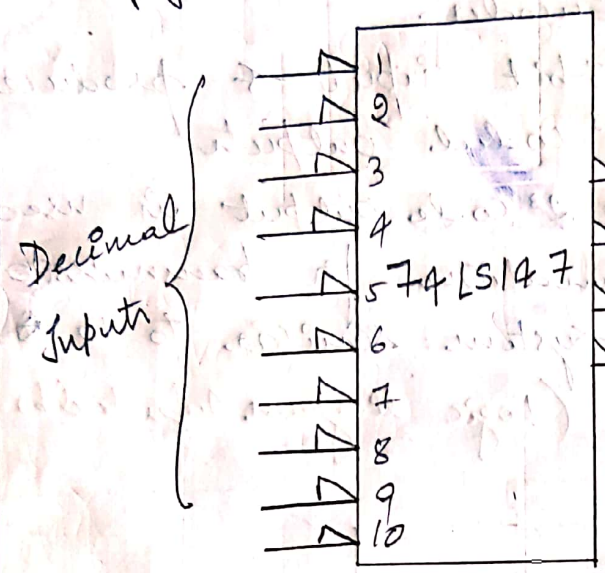
Typical Encoder

Encoders perform a function that is

Inverse of decoders.

+ An Encoder produces 'n' outputs from 2ⁿ inputs as shown in fig below.
+ The logic diagram of 74XX147, IEE symbol & function table is shown in fig below (A 10 line to BCD Encoder)

(Circuit diagram refer text book)



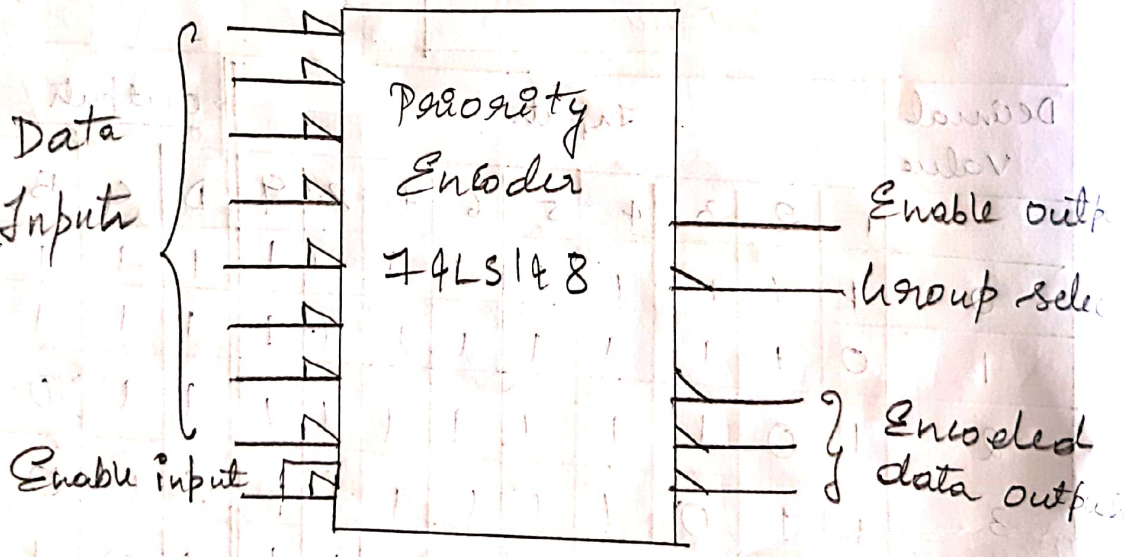
Decimal value	Inputs									Outputs (Complement of real value)			
	1	2	3	4	5	6	7	8	9	D	C	B	A
0	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1	1	1	0
2	1	0	1	1	1	1	1	1	1	1	1	0	1
3	1	1	0	1	1	1	1	1	1	1	1	0	0
4	1	1	1	0	1	1	1	1	1	1	0	1	1
5	1	1	1	1	0	1	1	1	1	1	0	1	0
6	1	1	1	1	1	0	1	1	1	1	0	0	1
7	1	1	1	1	1	1	0	1	1	1	0	0	0
8	1	1	1	1	1	1	1	0	1	0	1	1	1
9	1	1	1	1	1	1	1	1	0	0	1	1	0

9 → 1001
Comp → 0110

* The 9 input lines to the Encoder

⊕ Priority Encoder ⇒

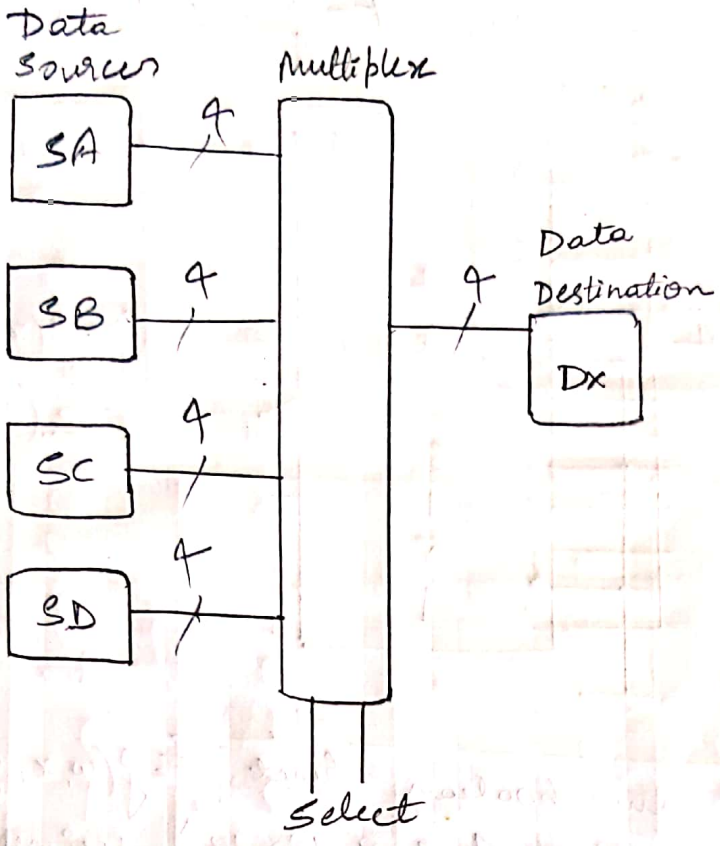
* The below figure shows the logic diagram, logic symbol and function table of priority Encoder (74xx148), a 8 line to 3 line priority Encoder.
+ It accepts a 3-bit input & produces a one out of 2^3 coded outputs
* The one out of 2^3 code output is used to encode information for transmission to a digital system similar to 10 line to BCD Encoder. (Logic diagram refer textbook)



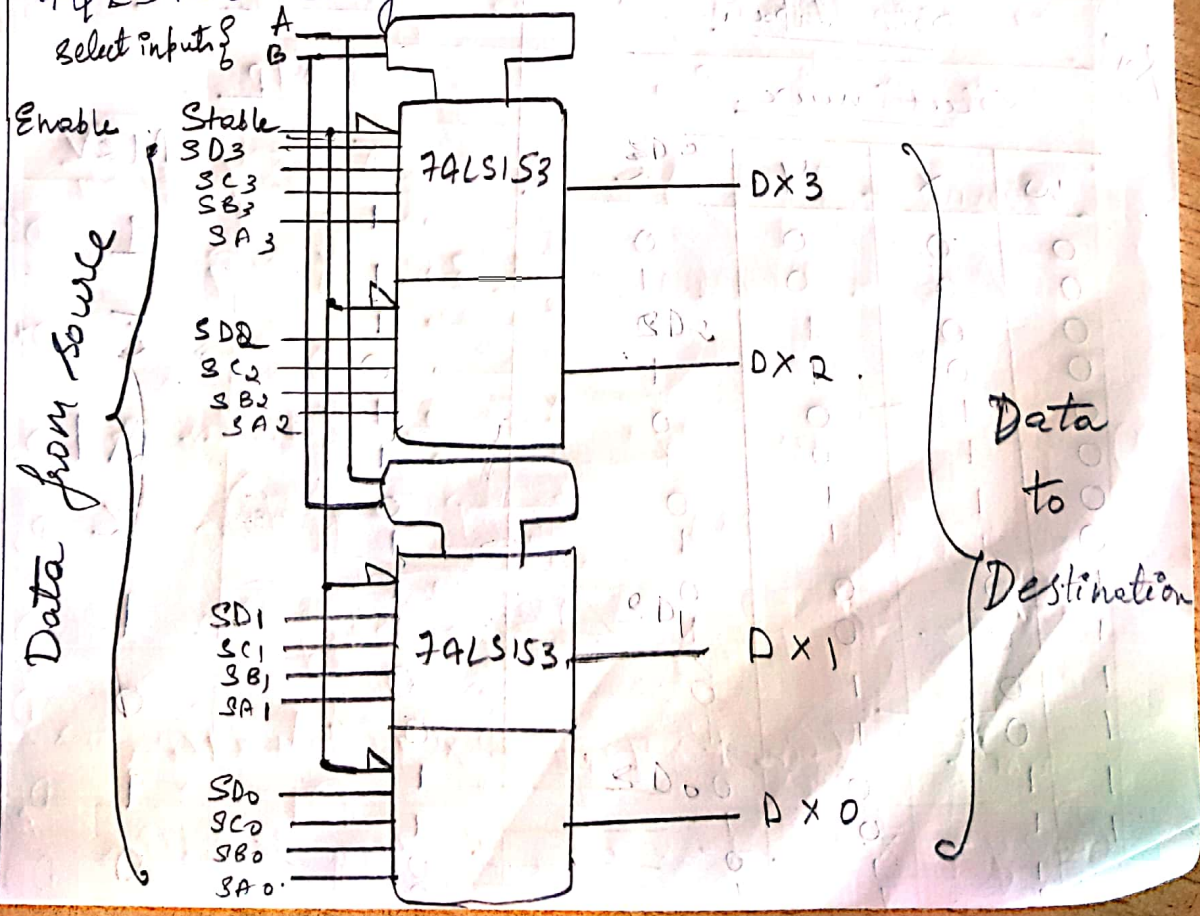
Inputs									Outputs (current)					
E_1	0	1	2	3	4	5	6	7		A_2	A_1	A_0	GS	EO
1	x	x	x	x	x	x	x	x	→	1	1	1	1	1
0	1	1	1	1	1	1	1	1	→	1	1	1	1	0
0	0	1	1	1	1	1	1	1	→	1	1	1	0	1
0	x	0	1	1	1	1	1	1	→	1	1	0	0	1
0	x	x	0	1	1	1	1	1	→	1	0	1	0	1
0	x	x	x	0	1	1	1	1	→	1	0	0	0	1
0	x	x	x	x	0	1	1	1	→	0	1	1	0	1
0	x	x	x	x	x	0	1	1	→	0	1	0	0	1
0	x	x	x	x	x	x	0	1	→	0	0	1	0	1
0	x	x	x	x	x	x	x	0	→	0	0	0	0	1

* The output $\overline{GS}$ goes low when any of the inputs to the encoder is active.
+ The purpose of the signal is to communicate to the control system logic that an event has occured & priority encoded data are present.
+ When EO is active no priority event connected to the IC is present. It is used as enable input to the next lower priority encoder.
* EO is an active low signal that can be used to cascade several devices to form a larger priority Encoding System.

(*) 4-bit data sources multiplex to a single 4-bit destination ⇒

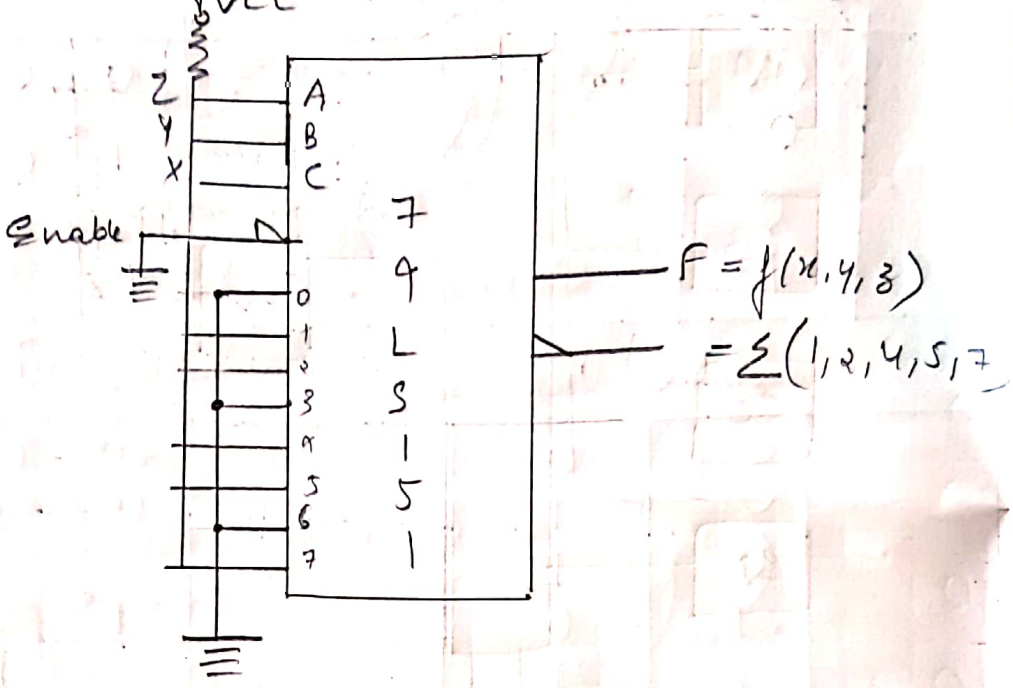


(**) A 4-bit multiplexer system using 2 74LS153 Intigrated circuits.



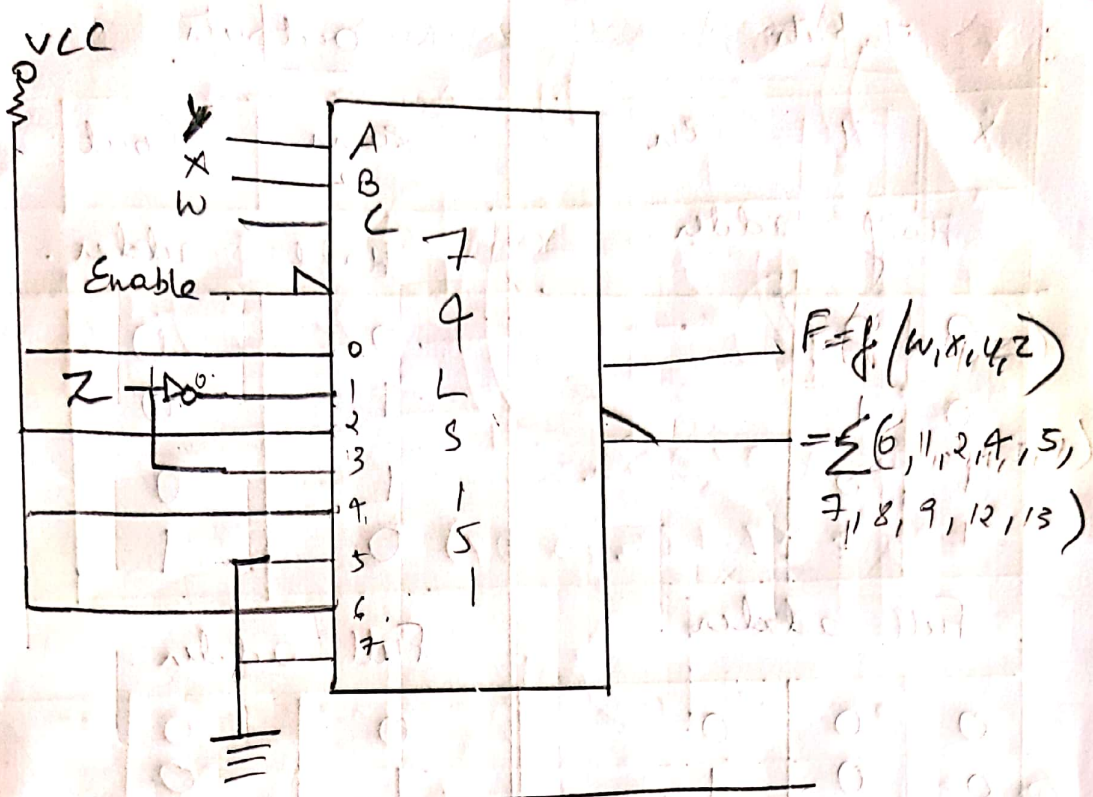
Problem →

1) Using an 8:1 mux 74LS151 realize the boolean function $F = f(x,y,z) = \Sigma(1,2,4,5,7)$



2) Realize the boolean funct $T = f(w,x,y,z) = \Sigma(0,1,2,4,5,7,8,9,12,13)$ using 8:1 mux (74LS151)

	Data Inputs				Data outputs	
	Select mux.				O/P	
	w	x	y	z	T	MEV
0	0	0	0	0	1	D^0
1	0	0	0	1	1	
2	0	0	1	0	1	$\bar{Z}$ D^1
3	0	0	1	1	0	
4	0	1	0	0	1	D_2
5	0	1	0	1	1	
6	0	1	1	0	0	Z D_3
7	0	1	1	1	1	
8	1	0	0	0	1	D_4
9	1	0	0	1	1	
10	1	0	1	0	0	D_5
11	1	0	1	1	0	
12	1	1	0	0	1	D_6
13	1	1	0	1	1	
14	1	1	1	0	0	D_7
15	1	1	1	1	0	



(*) Adders And Subtractors ⇒

* Adding two single bit binary values produces a sum & a carry output.
* This operation is called a Half Add & the circuit that realises the function is called a Half Adder.
+ Adding two single bit binary values, with the inclusion of a carry input produces 2 outputs a sum & a carry. This circuit is called a Full Adder.
+ The truth table for Half & Full Adder is given below.

(P) Truth Table ⇒

Inputs			outputs	
x	y	Cin	Sum	Cout
Half adder.			Half adder.	
0	0		0	0
0	1		1	0
1	0		1	0
1	1		0	1
Full adder.			Full adder.	
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

(*) The boolean equations for the 2 outputs is given below,

$$Sum = f(x,y) = \Sigma(1,2) = \bar{x}y + x\bar{y}$$

$$Cout = f(x,y) = \Sigma 3 = xy$$

(*) The boolean eqn for full adder is,

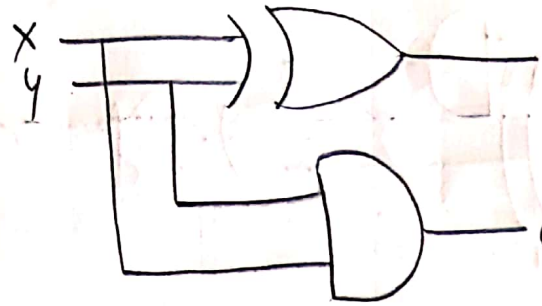
$$Sum = f(x,y,Cin) = \Sigma(1,2,4,7)$$

$$Cout = f(x,y,Cin) = \Sigma(3,5,6,7)$$

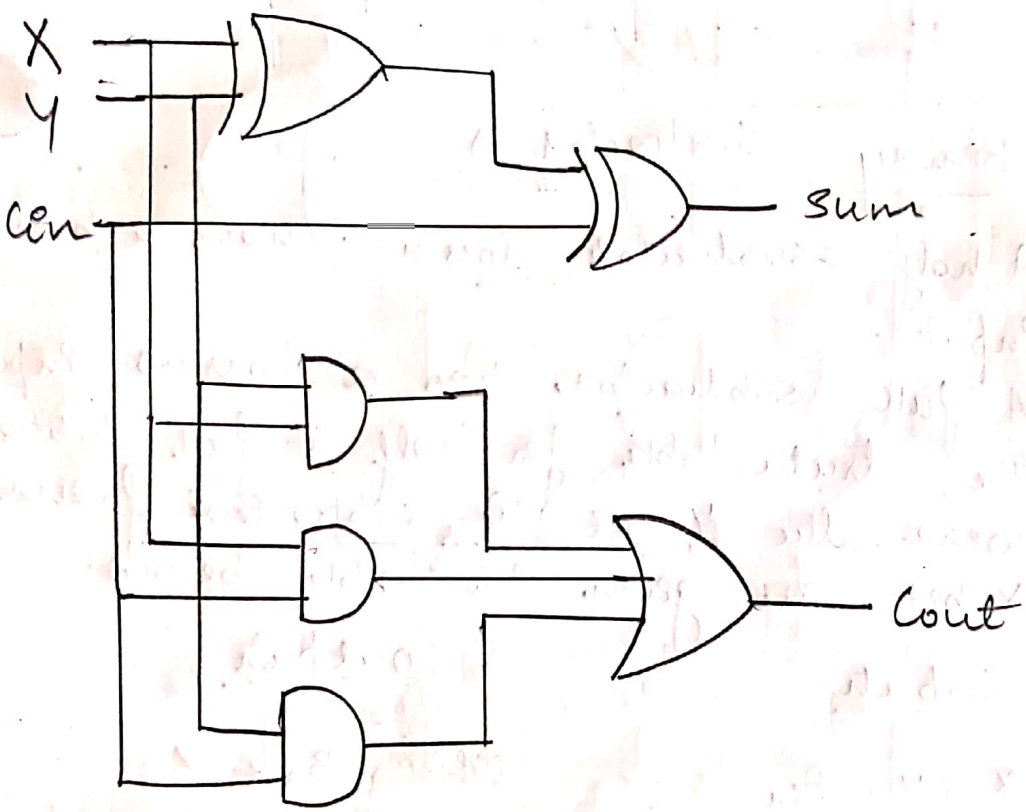
$$Sum = \bar{x}\bar{y}Cin + \bar{x}y\overline{Cin} + x\bar{y}\overline{Cin} + xyCin \text{ or } x \oplus y \oplus Cin$$

$$Carry = \bar{x}yCin + x\bar{y}Cin + xy\overline{Cin} + xyCin$$

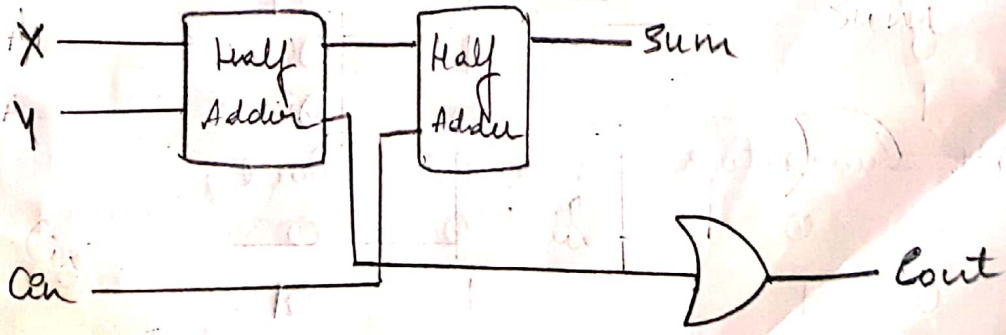
* Half Adder logic diagram →

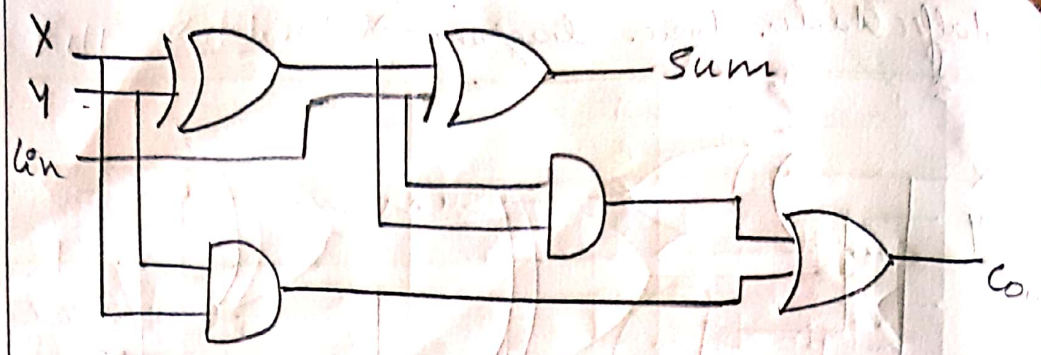


* Full Adder logic diagram →



* Design a Full Adder using Half Adder ⟹





$Sum = X \oplus Y \oplus Cin$

$Carry = XY + Cin(X \oplus Y)$

$XY + Cin(\bar{X}Y + X\bar{Y})$

$XY + CinXY + Cin\bar{X}Y$

⊛ Binary Subtractor ⇒

* A half subtractor does not have a borrow input.
+ A full subtractor has a borrow input
+ The truth table for half & full subtractor were the Y-bit is subtracted from X-bit. are given in table below.

Inputs			outputs	
X	Y	B_{in}	Diff	Bout
Half subtractor:				
0	0		0	0
0	1		1	1
1	0		1	0
1	1		0	0
Full Subtractor			Diff	Bout.
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1

1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

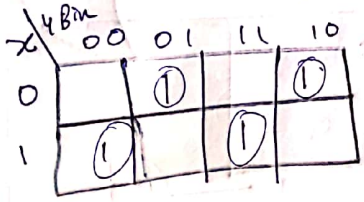
+ The output eqn for half subtractor are.

$$Diff = f(x,y) = \Sigma(1,2) = \bar{x}y + x\bar{y}$$

$$B_{out} = f(x,y) = \Sigma(1) = \bar{x}y.$$

+ The output eqn for full subtractor are

$$B_{in} = \Sigma(1,2,4,7) =$$

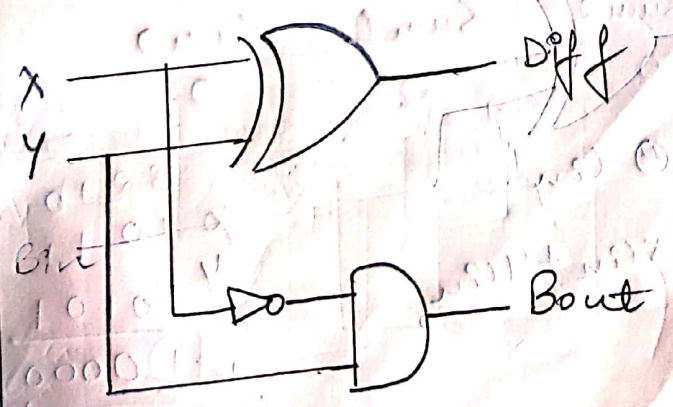


$$Diff = x\bar{y}\bar{B}in + \bar{x}\bar{y}Bin + xyBin + \bar{x}y\bar{B}in$$

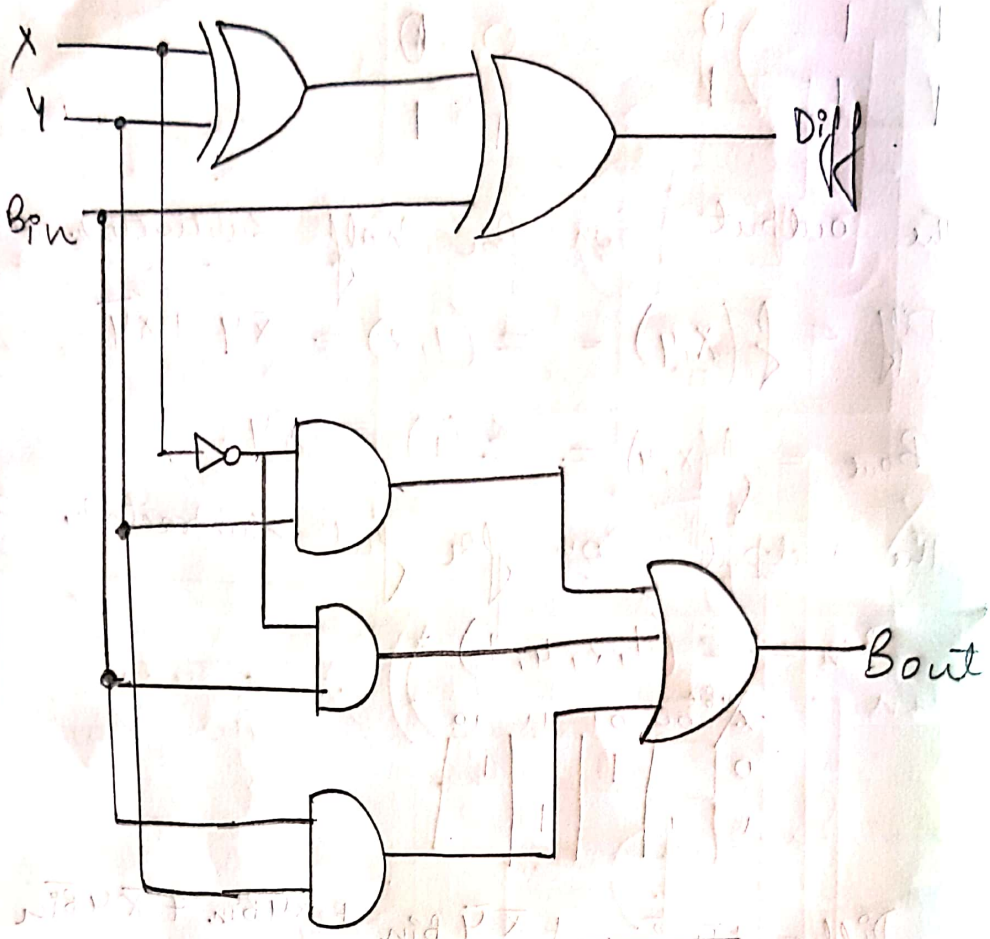
$$Diff. = x \oplus y \oplus Bin$$

$$B_{out} \; \bar{x}y + \bar{x}Bin + yBin.$$

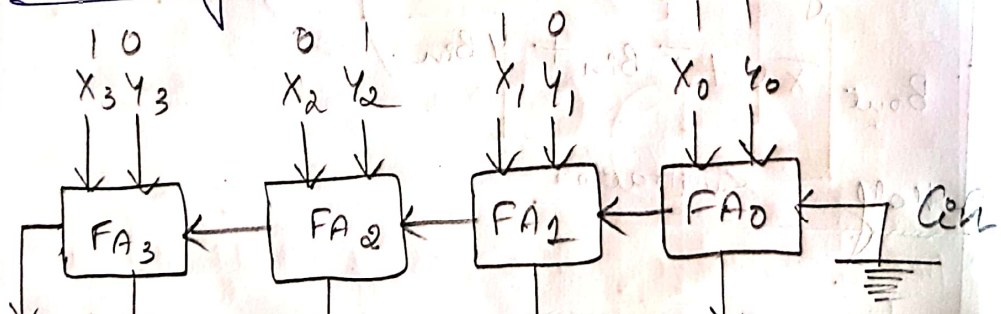
⊕ Half Subtractor ⇒



Full Subtractor ⟹



⊕ Cascading Full Adders ⟹



Sum = X ⊕ Y ⊕ Cin

Cout = XY + XCin + YCin

3 2 1 0
X 1 0 1 1
Y 0 1 0 1
1 0 0 0 0
Cout ⟺ Sum3 : Sum