08/07/2022

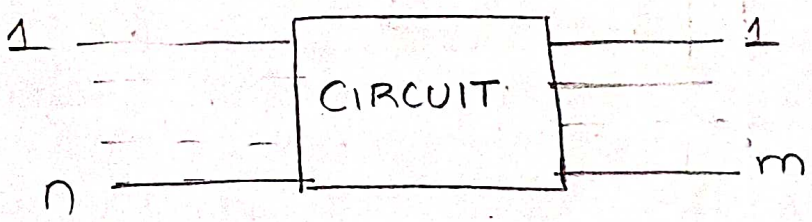
UNIT-3

COMBINATIONAL CIRCUITS:

* Design Procedure:

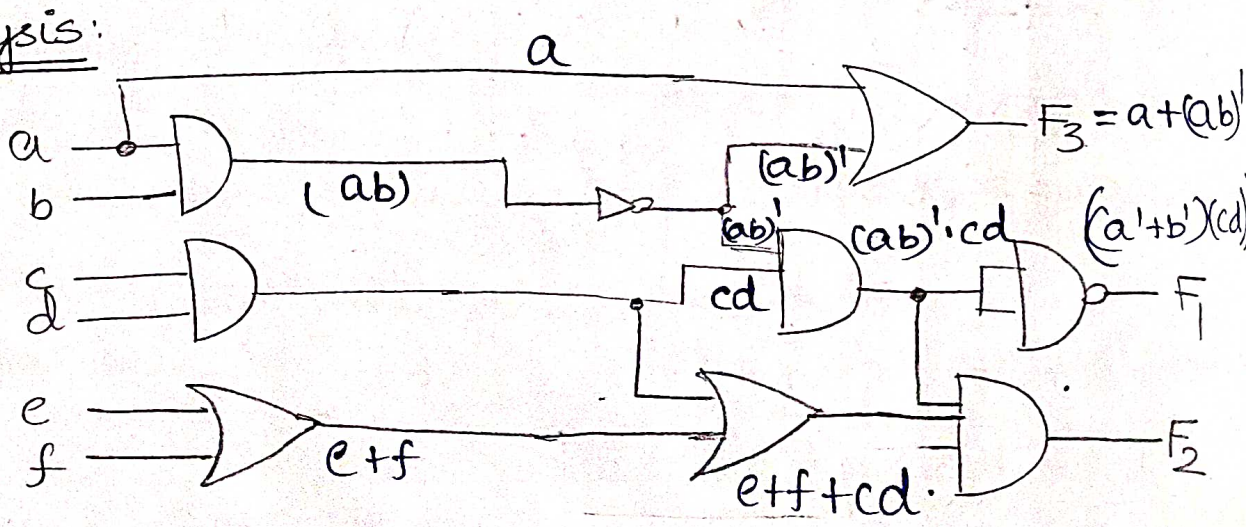
1) Identifying I/Ps and O/Ps & assign them labels
2) Draw the Truth Table.
3) Derive Boolean functions
4) Simplify (Reduce no. of literals) function.
5) Using gates provided; design the circuit diagram.

→ O/P is dependant on given I/P.



no. of inputs need not be equal no. of outputs.

Analysis:



$F_3 = a + a' + b' = b' + 1$ (b) $F_3 = 1$

$F_1 = (a'+b')' + (cd)'$

$\boxed{F_1 = ab + (cd)'}$

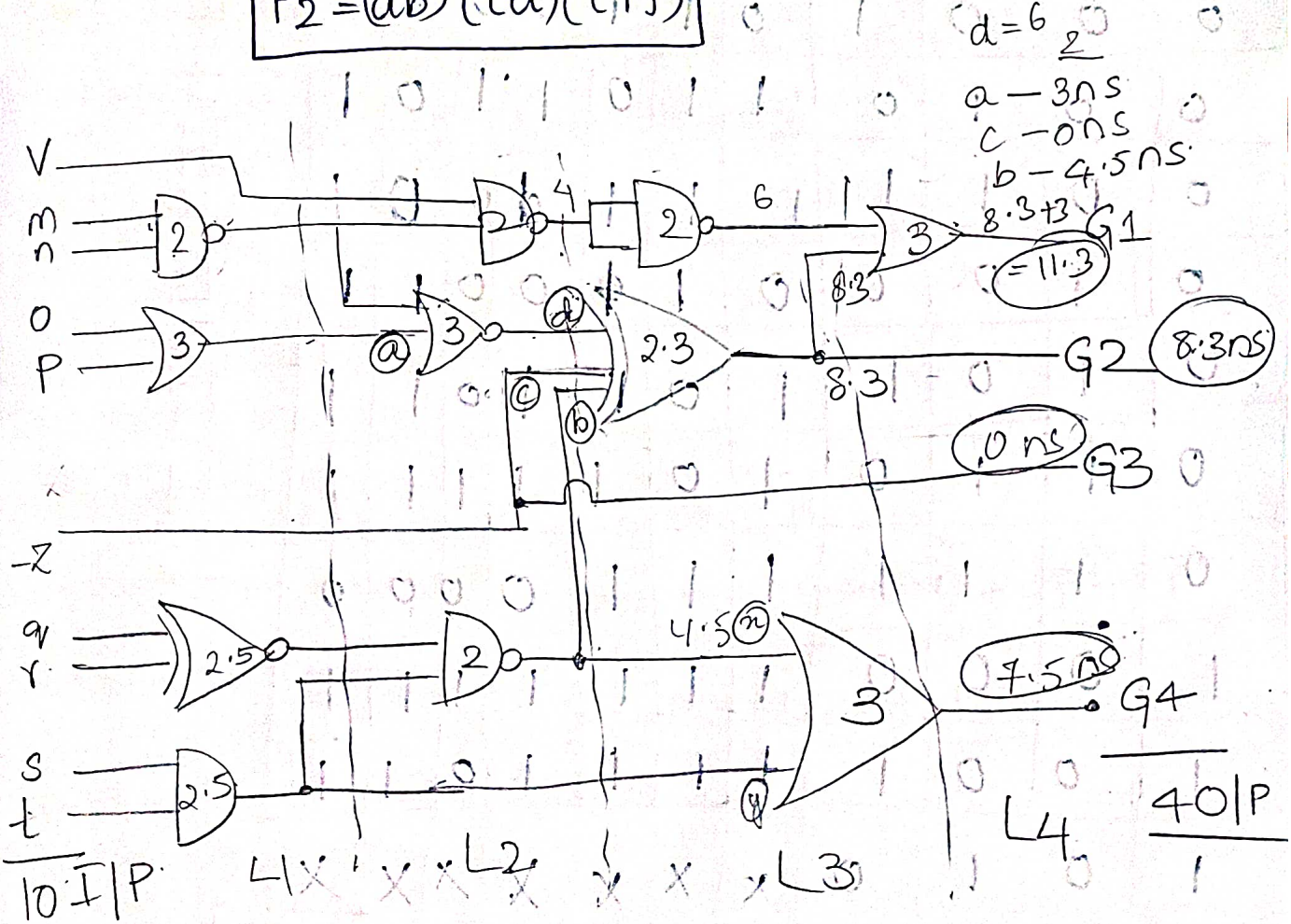
$F_2 = (ab)' \cdot cd \,(e+f+cd)$

$= (ab)'e + (ab)'f + a'(ab)'cd$

$= (ab)'cde + (ab)'cdf + (ab)'cd$

$F_2 = (ab)'cd\,(e+f+1) = (ab)' \cdot (cd)(e+f)$

$\boxed{F_2 = (ab)'(cd)(e+f)}$



Propagation Delay

Nand – 2ns ExNOR – 2.5ns

Nor – 3ns

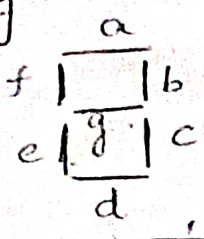
And – 2.5ns

OR – 3ns

EXOR = 2.3Ns

$$F4 = [(q \cdot r)\,st] + st$$

* Design a circuit to lit 7-segment display as 7-segment display 0-9.



I/P size: 4 bits

let m, n, o, p → be I/P

m	n	o	p	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	x	x	x	x	x	x	x
1	0	1	1	x	x	x	x	x	x	x
1	1	0	0	x	x	x	x	x	x	x
1	1	0	1	x	x	x	x	x	x	x
1	1	1	0	x	x	x	x	x	x	x

0 1 2 3 4 5 6 7 8 9

1	1	1	1	X	X	X	X	X	X	X

$a = \Sigma m(0,2,3,5,6,7,8,9) + d(10,11,12,13,14,15)$

$b = \Sigma m(0,1,2,3,4,7,8,9) + d(10,11,12,13,14,15)$

$c = \Sigma m(0,1,3,4,5,6,7,8,9) + d(10,11,12,13,14,15)$

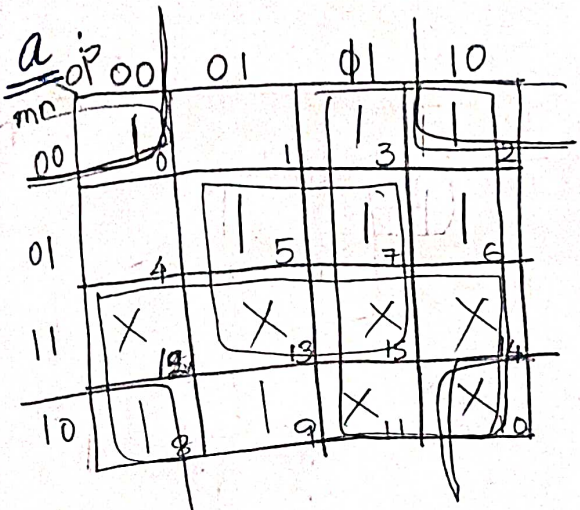
$d = \Sigma m(0,2,3,5,6,8,9) + d(10,11,12,13,14,15)$

$e = \Sigma m(0,2,6,8) + d(10,11,12,13,14,15)$

$f = \Sigma m(0,4,5,6,8,9) + d(10,11,12,13,14,15)$

$g = \Sigma m(2,3,4,5,6,8,9) + d(10,11,12,13,14,15)$

a:

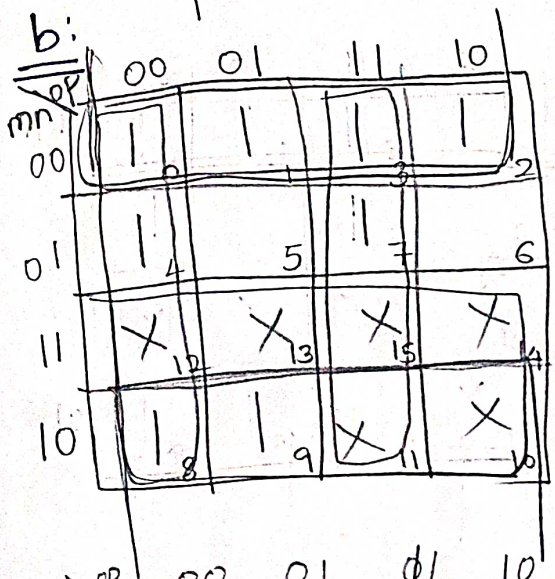


$a = x'z + x'y'w' + xyw + x$

$= x + z + x'y'w' + yw$

$\boxed{a = n'p' + m + o + np}$

b:

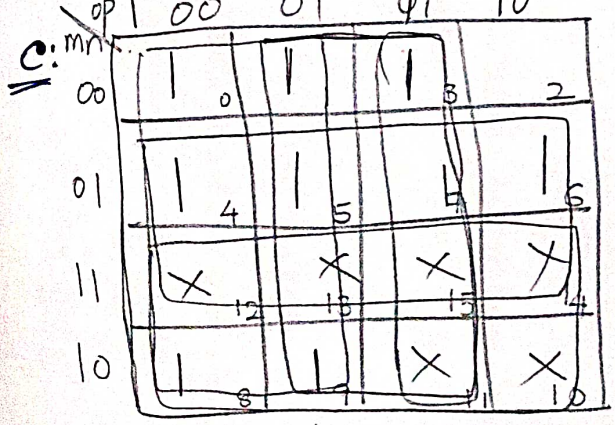


~~$b = o'p'$~~

$b = n' + m + mn + o'p' + op$

$\boxed{b = n' + m + o'p' + op}$

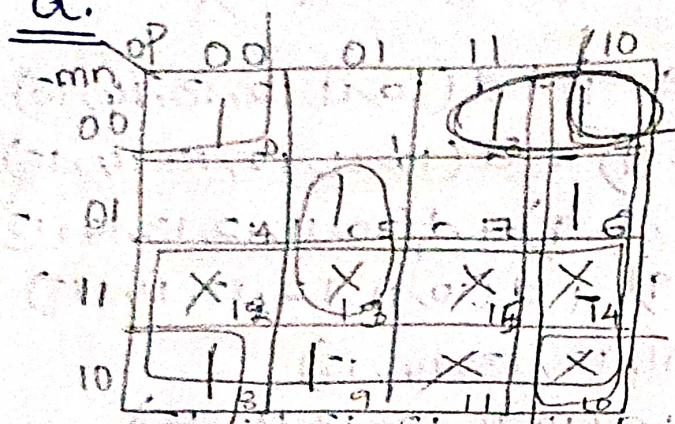
c:



$c = m + n + p + op$

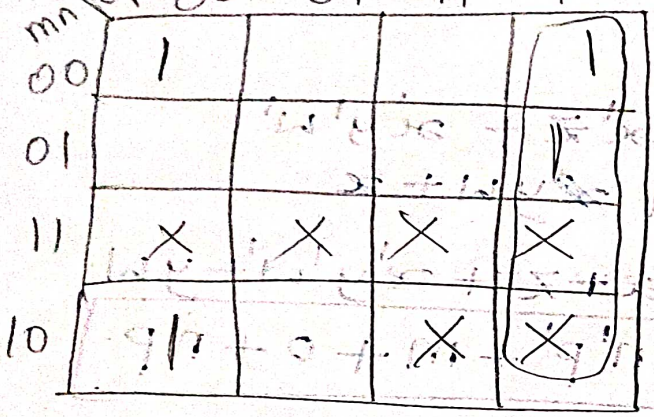
$\boxed{c = m + n + p + o'}$

d:



$$d = n' + m + op' + no'p + m'n'o$$

e:



$a'b + bc$

$F = (a'+b)(b+c')$

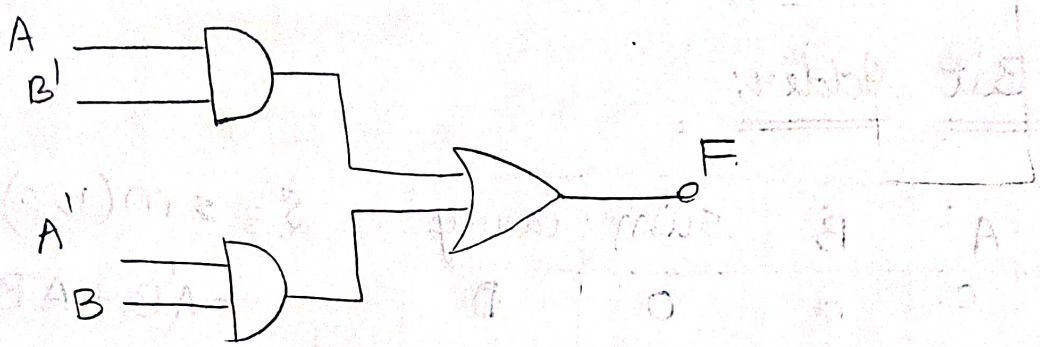
$F' = (a+b')(b'+c)$

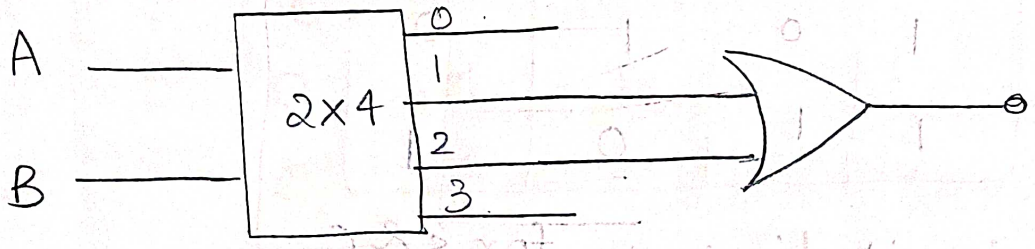
28/07/2022

→ Decoder: for n inputs, it generates 2^n outputs.

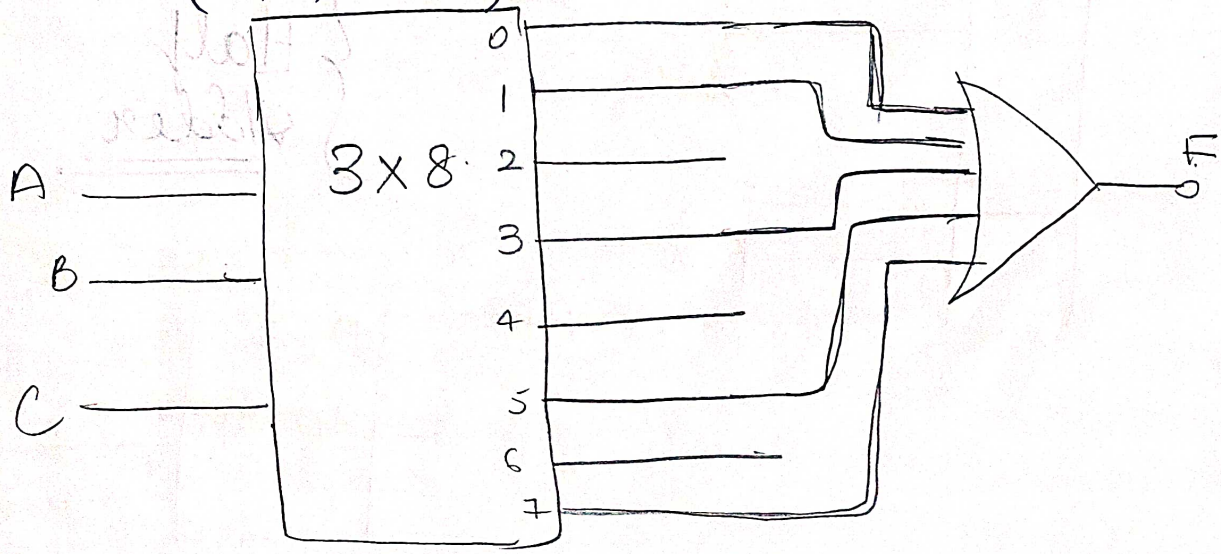
A	B	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

$F = \Sigma m(1,2) = AB' + A'B$.

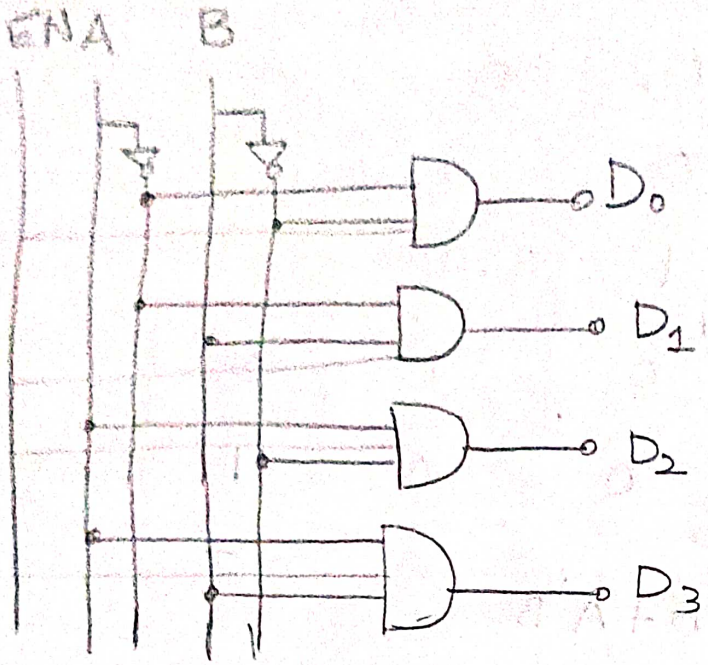




* $F = \Sigma m(0,1,3,5,7)$



* Circuit for 2×4 decoder



* 2-Bit Adder:

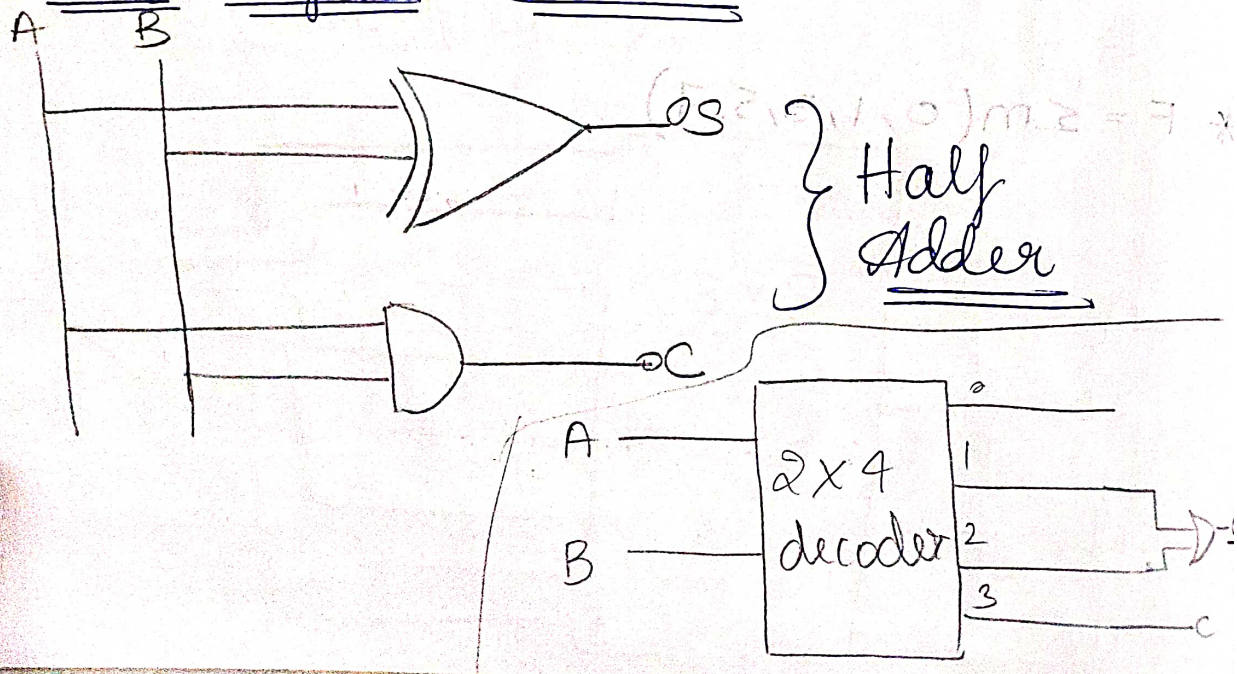
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

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

$= A'B + AB'$

$C = \Sigma m(3) = AB.$

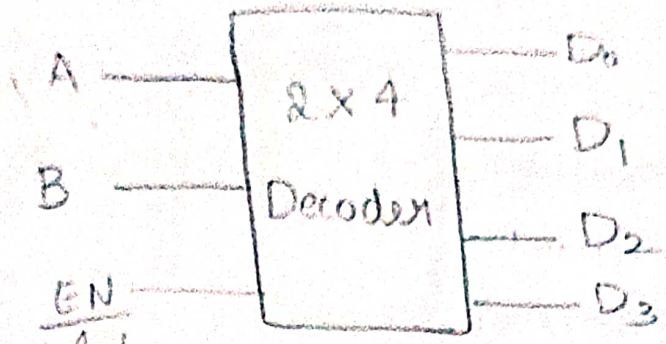
→ circuit diagram: for s & C



* Applications of Decoder:

→ realize a function
→ I/o Devices selection
→ Memory banks selection.

*



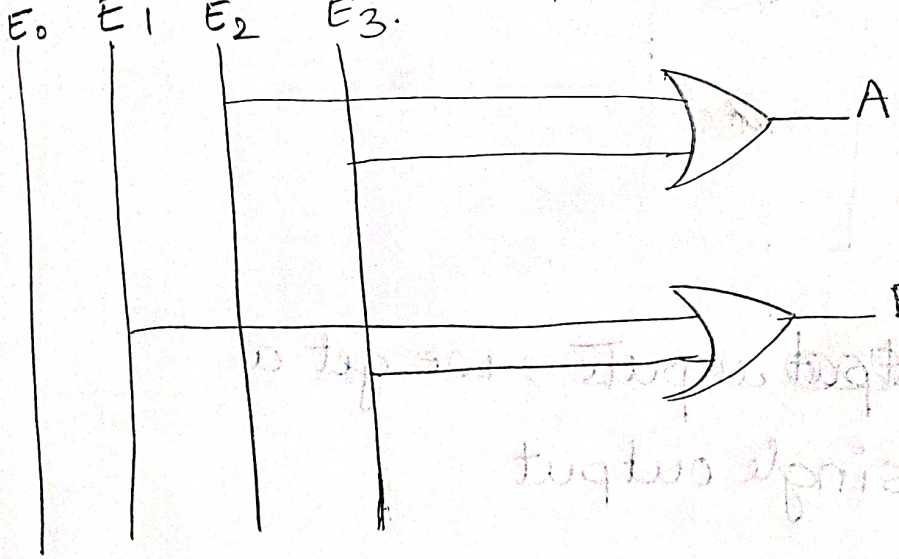
1 → Enables the output
EN=0 → Disables.

DECODER: n I/P ⟶ 2^n O/P
ENCODER: 2^n I/P ⟶ n O/P

E_0	E_1	E_2	E_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

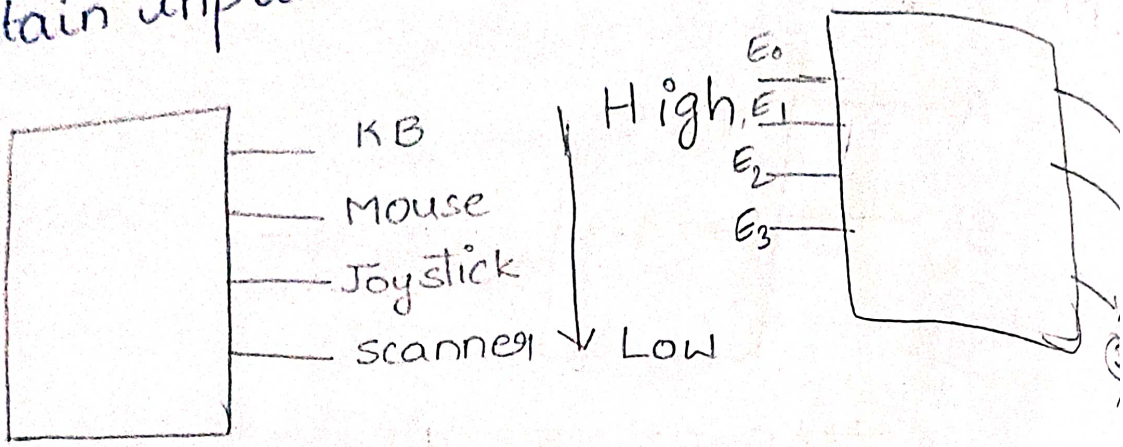
$A = E_2 + E_3$

$B = E_1 + E_3$



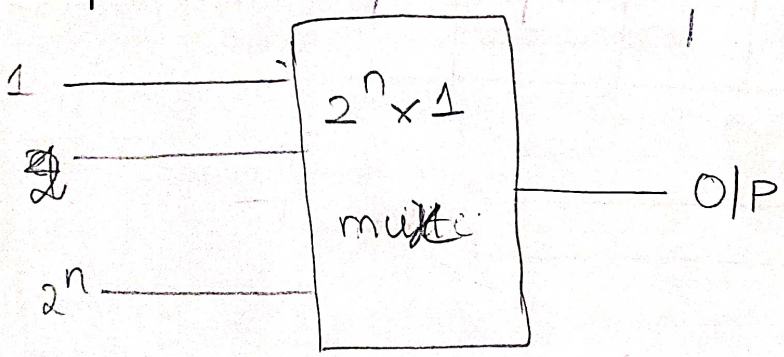
- Communication technology
- Compression.

* Priority Encoder: Giving priority to certain input.

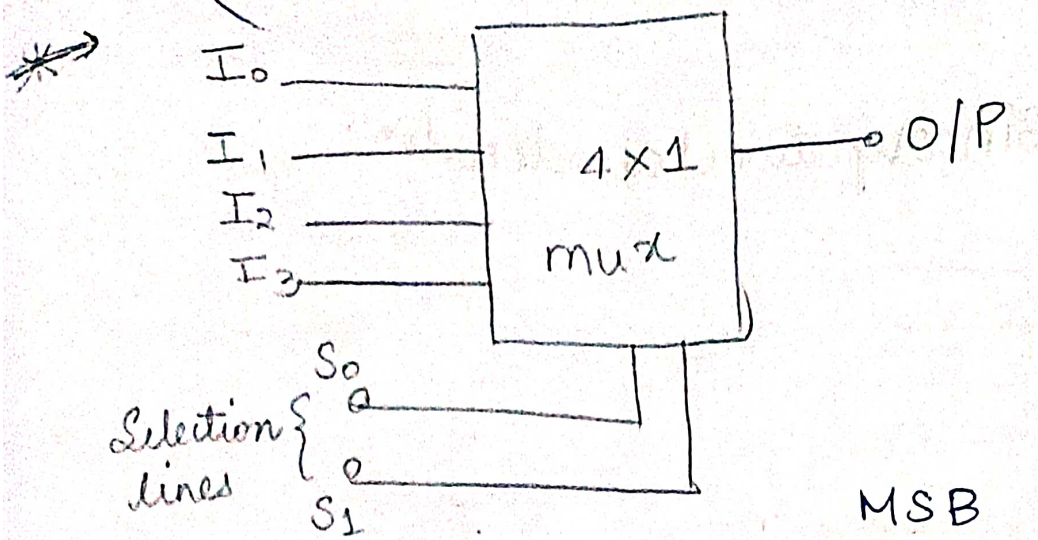


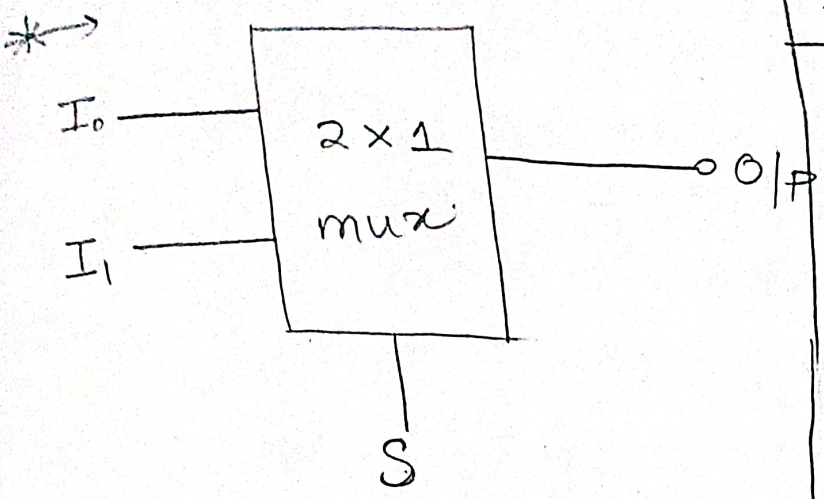
E_0	E_1	E_2	E_3	A	B	Valid
1	x	x	x	0	0	V
0	1	x	x	0	1	V
0	0	1	x	1	0	V
0	0	0	1	1	1	V
0	0	0	0	x	x	Invalid

* Multiplexer:



for 2^n inputs ; we get a single output.



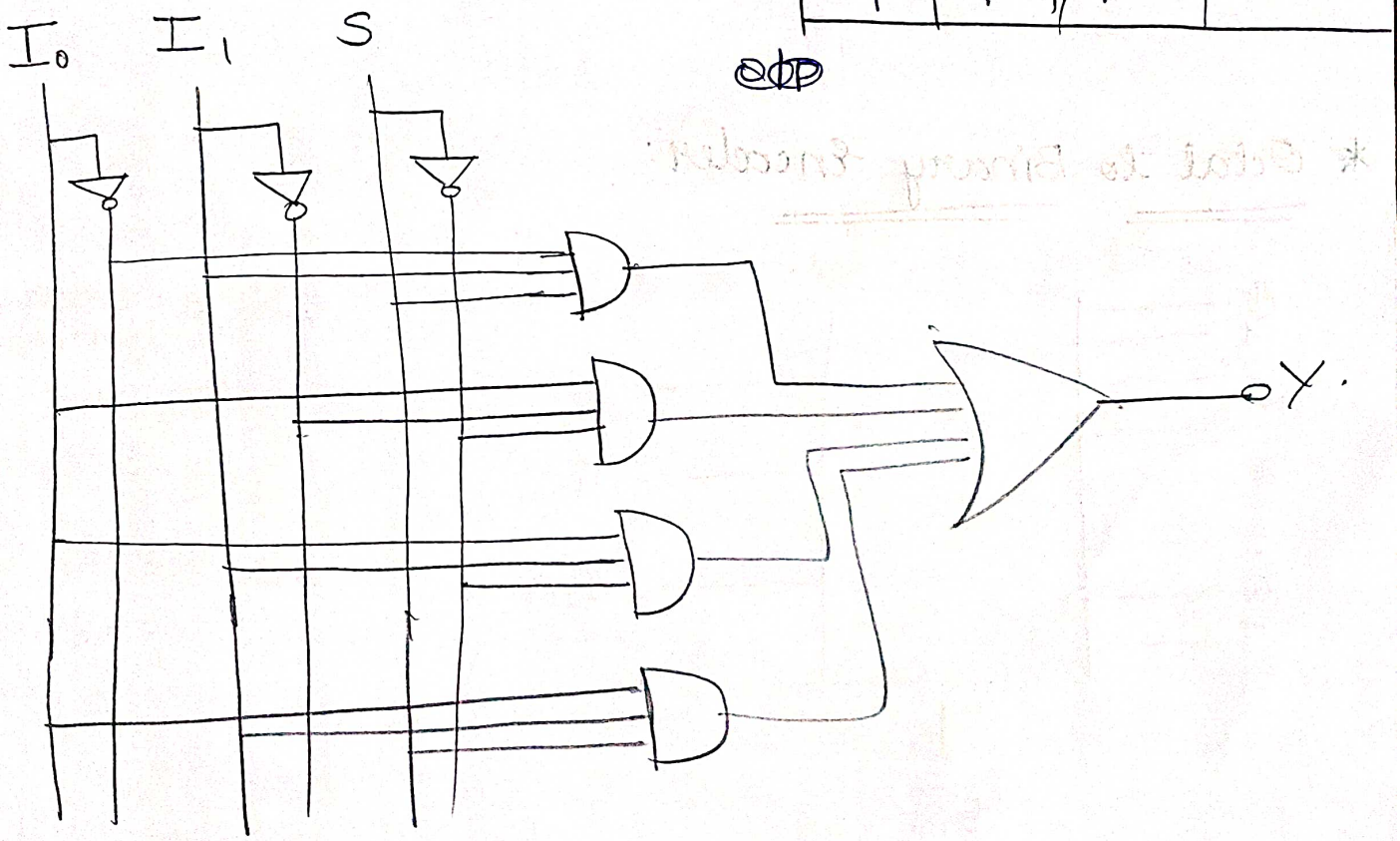


MSB LSB

I_0	I_1	S	O/P
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

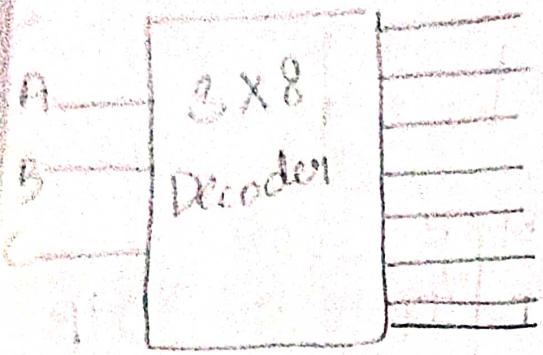
$= \Sigma m(3,4,6,7)$

$= I_0' I_1 S + I_0 I_1' S' + I_0 I_1 S' + I_0 I_1 S$

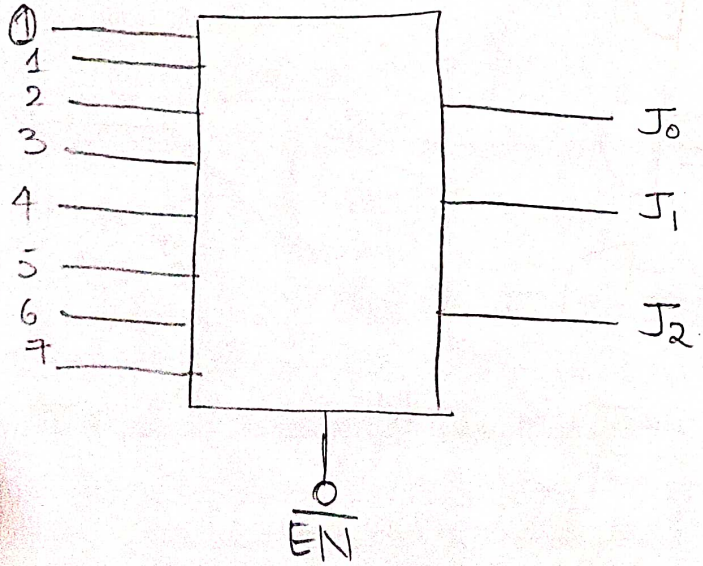


29/07/2022

① Design a Binary to octal decoder:



* Octal to Binary Encoder:

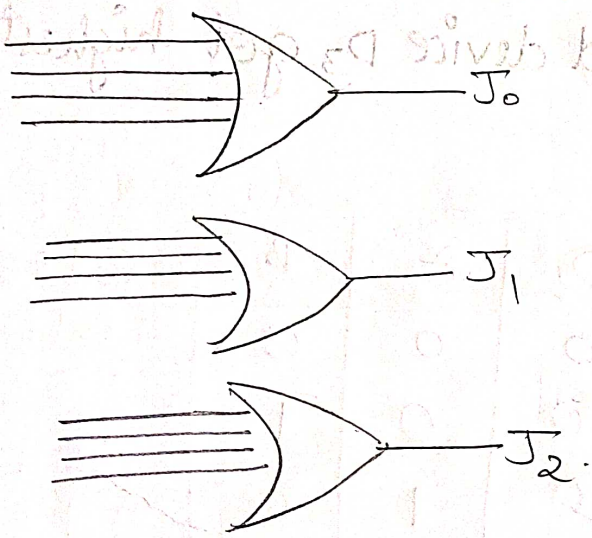


$\overline{EN}$	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	J_0	J_1	J_2
0	1								0	0	0
0		1							0	0	1
0			1						0	1	0
0				1					0	1	1
0					1				1	0	0
φ						1			1	0	1
φ							1		1	1	0
φ								1	1	1	1

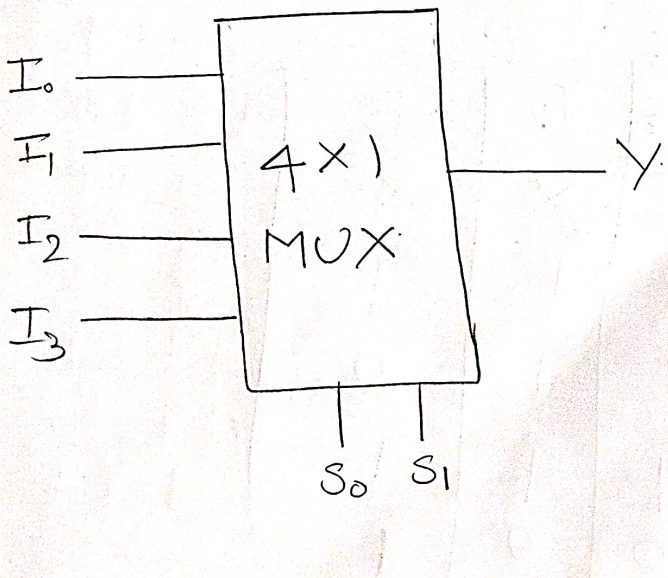
$J_0 = Y_4 + Y_5 + Y_6 + Y_7$

$J_1 = Y_2 + Y_3 + Y_6 + Y_7.$

$J_2 = Y_1 + Y_3 + Y_7 + Y_5.$



*MUX: Data Selector selects on the 1/1 lines.



$\overline{EN}$	S_1	S_0	Y
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	X	X	0

I_0	I_1	I_2	I_3	S_0	S_1	Y
1	0	0	0	0	0	
0	1	0	0	0	1	
0	0	1	0	1	0	
0	0	0	1	1	1	

$$Y = S_0' S_1' I_0 + S_0' S_1 I_1 + S_0 S_1' I_2 + S_0 S_1 I_3$$

8 4×1 Mux, 3 2×1 Mux

De-Mux:

1 → 2?

01/08/2022

* Design a encoder where device D_0 got lowest priority and device D_3 got highest priority.

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

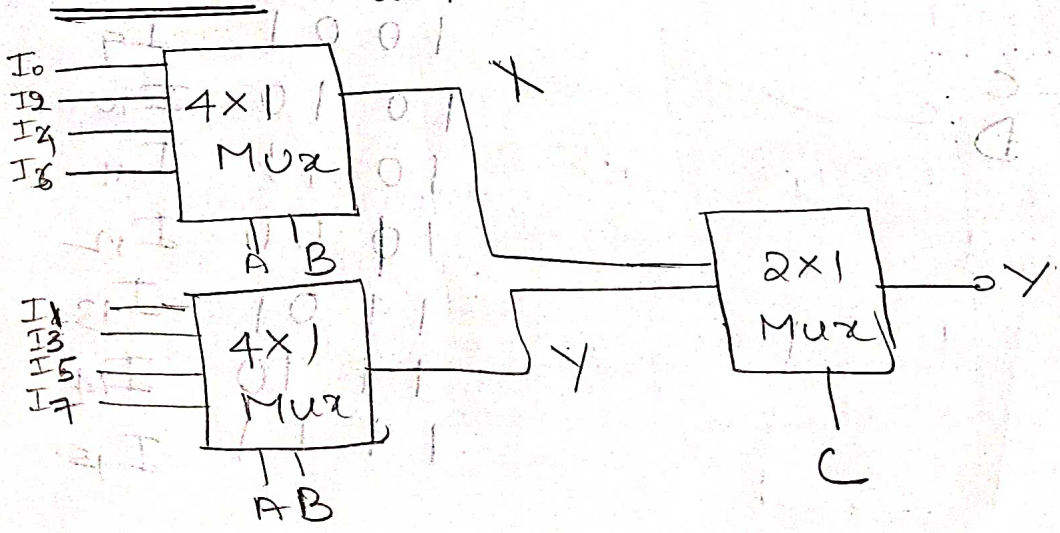
D_0	D_1	D_2	D_3	A	B
0	0	0	0	X	X
0	0	0	1	1	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1

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

$A = \Sigma m(1,2,3,5,6,7,10,9,11,13,14,15) + d(0)$

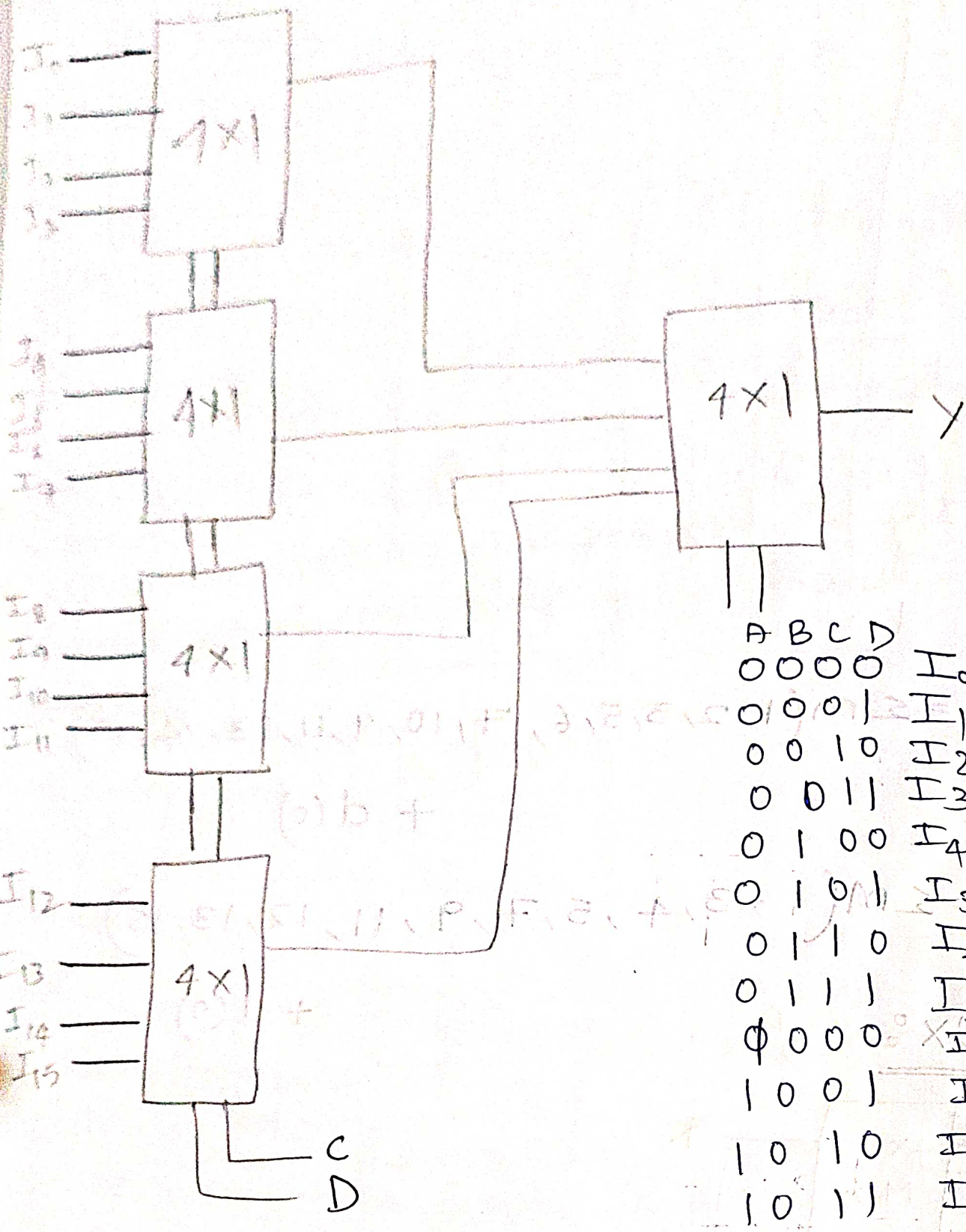
$B = \Sigma m(1,3,4,5,7,9,11,12,13,15) + d(0)$.

* MUX: $2^n \times 1 = 8 \times 1$ $(n=3)$



0 0 0	I_0
0 0 1	I_1
0 1 0	I_2
0 1 1	I_3
1 0 0	I_4
1 0 1	I_5
1 1 0	I_6
1 1 1	I_7

* Design 16 × 1 multiplexer using 4 × 1 Multi...



A	B	C	D	
0	0	0	0	I_0
0	0	0	1	I_1
0	0	1	0	I_2
0	0	1	1	I_3
0	1	0	0	I_4
0	1	0	1	I_5
0	1	1	0	I_6
0	1	1	1	I_7
ϕ	0	0	0	I_8
1	0	0	1	I_9
1	0	1	0	I_{10}
1	0	1	1	I_{11}
1	ϕ	0	0	I_{12}
1	1	0	1	I_{13}
1	1	1	0	I_{14}
1	1	1	1	I_{15}

* Design a circuit using multiplexer for a function

$F = \Sigma m(1, 3, 6, 7)$

(i) Decoder : 3×8

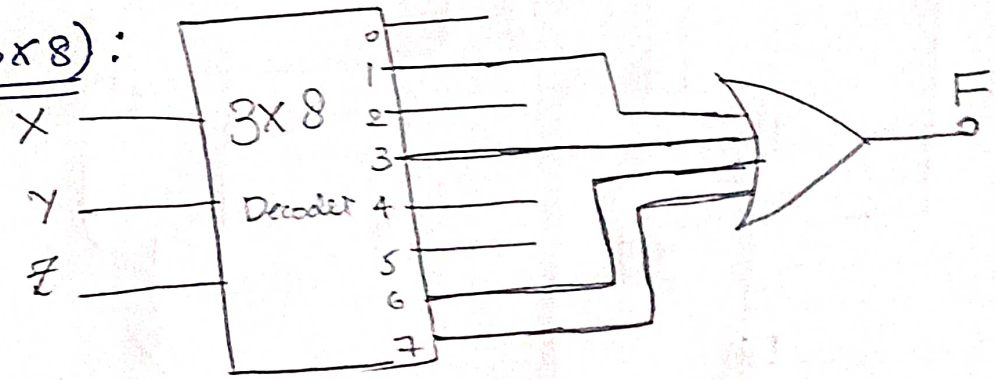
(ii) Mux – 8×1

(iii) Decoder : 2×8

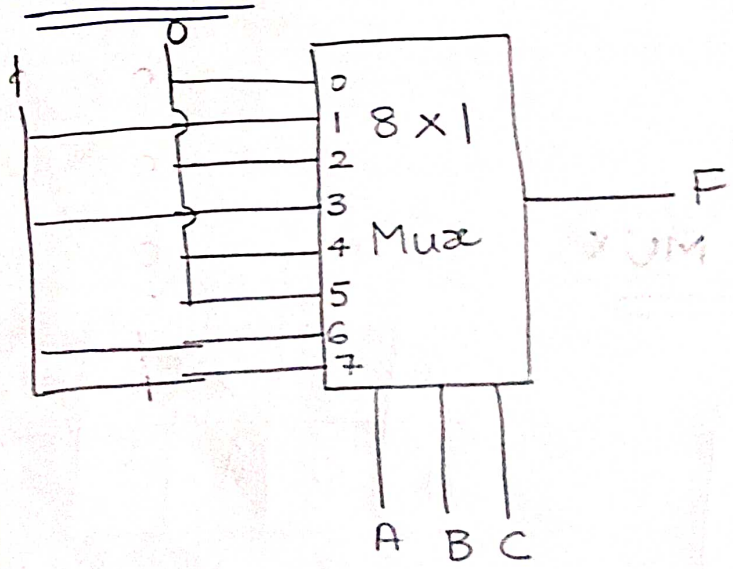
(iv) Mux – 4×1

X	Y	Z	F	
0	0	0	0	I_0
0	0	1	1	I_1
0	1	0	0	I_2
0	1	1	1	I_3
1	0	0	0	I_4
1	0	1	0	I_5
1	1	0	0	I_6
1	1	1	1	I_7

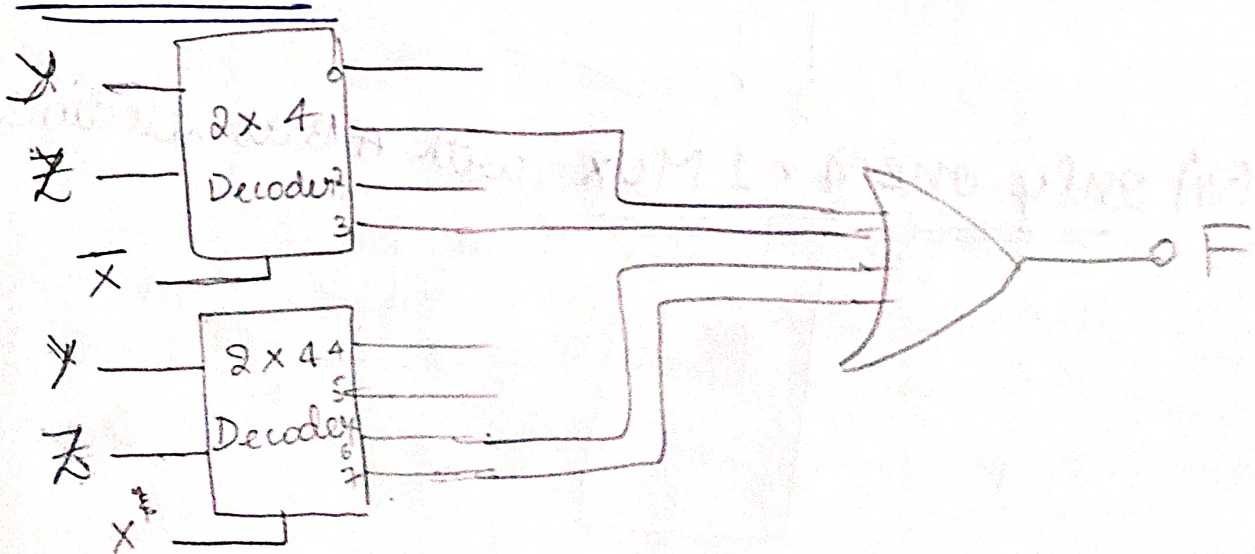
(i) Decoder (3×8):



(ii) 8×1 Mux:



(iii) 2×4 Decoder:



* Design a circuit using multiplexer, for a functio

$F = \Sigma m(1, 3, 6, 7)$

(i) Decoder : 3×8

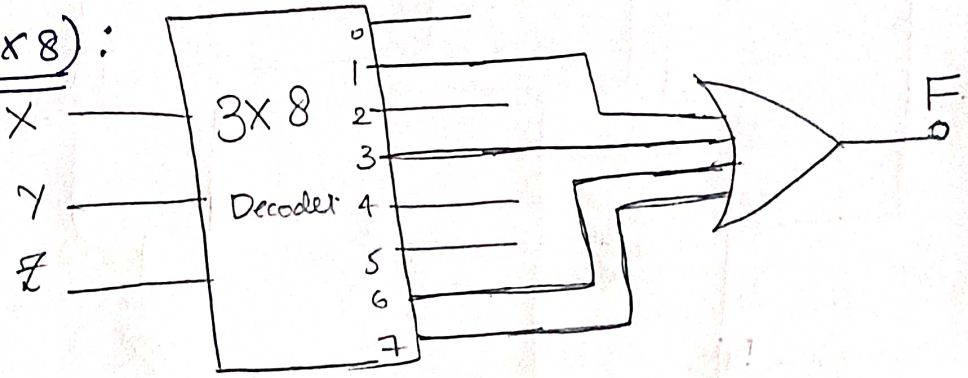
(ii) Mux – 8×1

(iii) Decoder : 2×8

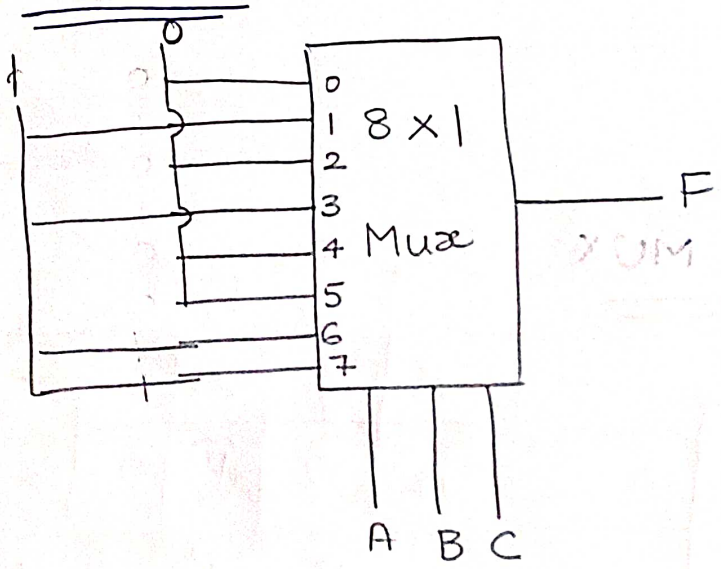
(iv) Mux – 4×1

x	y	z	
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

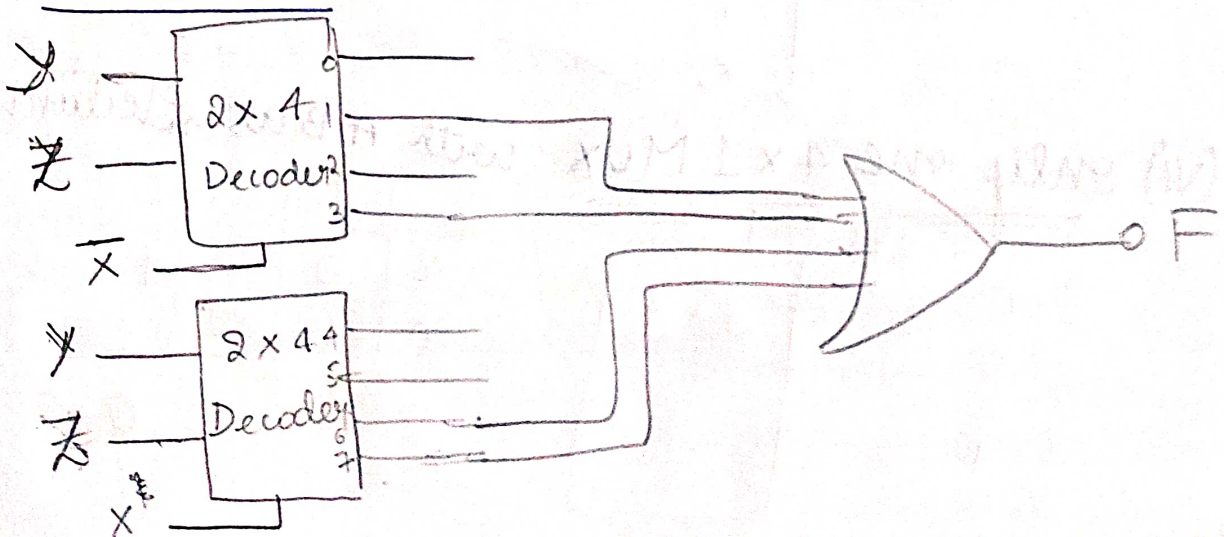
(i) Decoder (3×8):



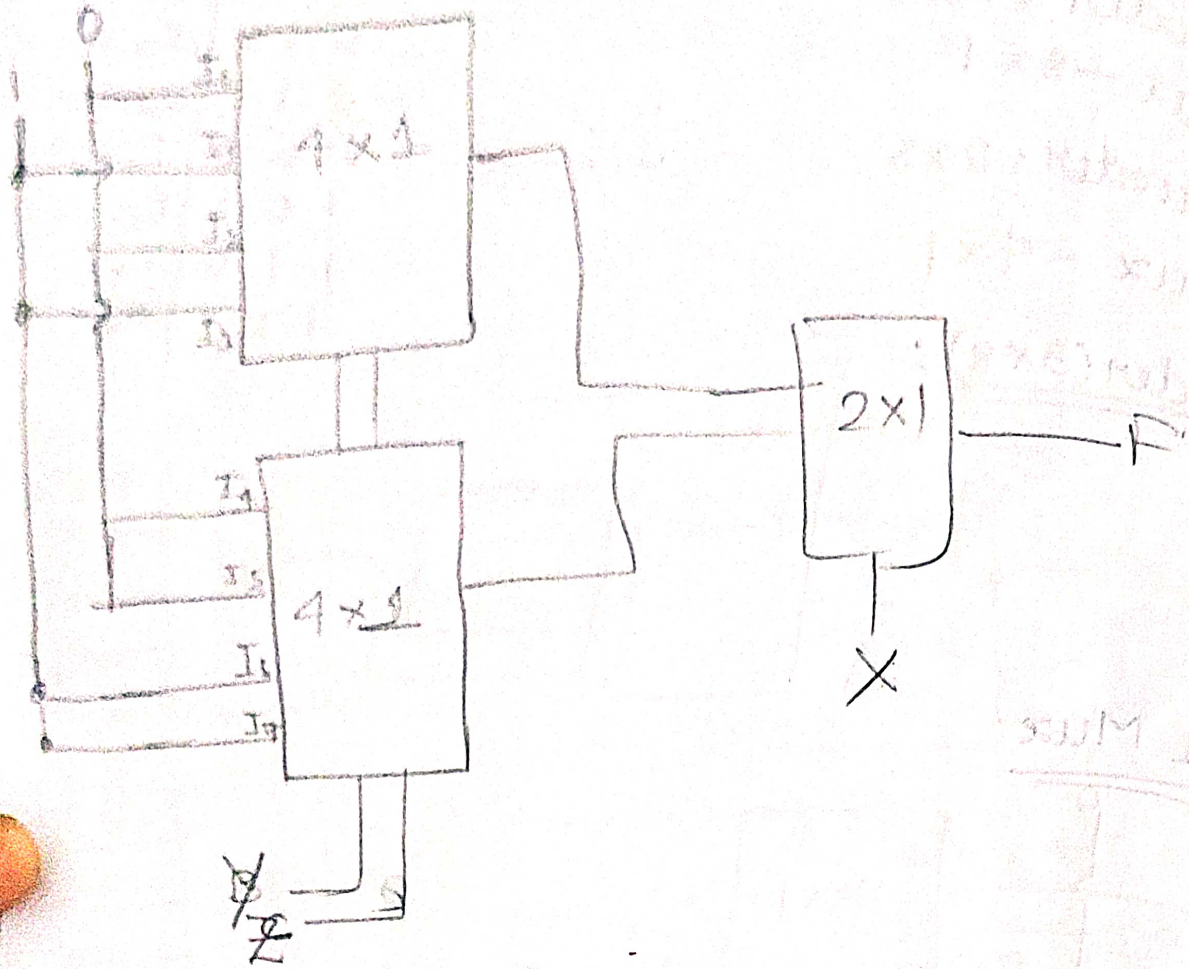
(ii) 8×1 Mux:



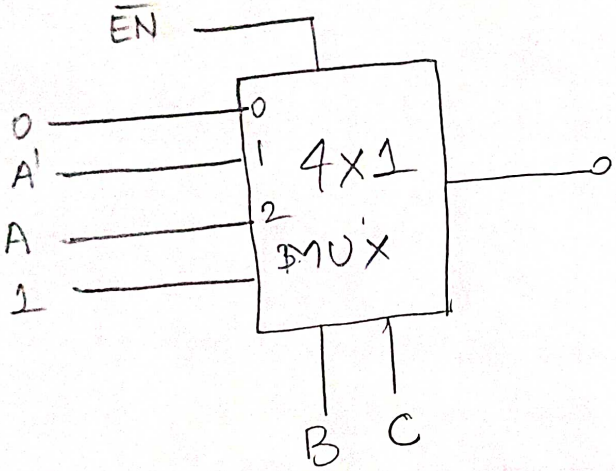
(iii) 2×4 Decoder:



(iv) Mux : 4 × 1

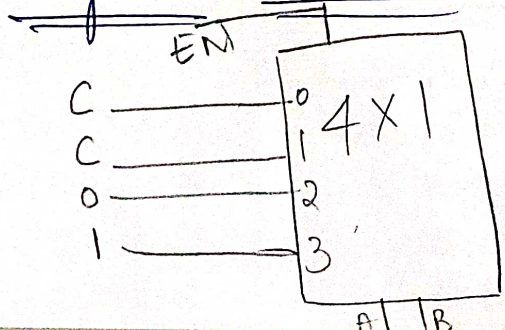


(v) only one 4 × 1 MUX:



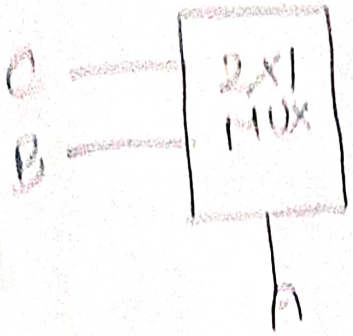
	I_0	I_1	I_2	I_3
A	4	5	(6)	(7)
A'	0	(1)	2	(3)
	0	A'	A	1

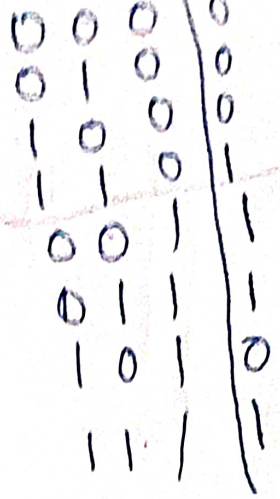
(vi) only one 4 × 1 MUX : with AB as selection line



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

(vii) only one 2×1 MUX:





	$I_0(A')$	$I_1(A)$
$B'C'$	0	4
$B'C$	1	5
BC'	2	6
BC	3	7
	C	B.

* Half Adder:

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$S = A'B + AB' = A \oplus B$

$C = AB$

* Full Adder: (Adder).

A	B	C	Sum	Carry
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

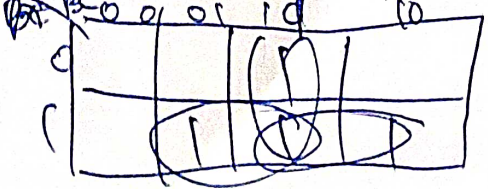
$C = AB + C(A \oplus B)$.

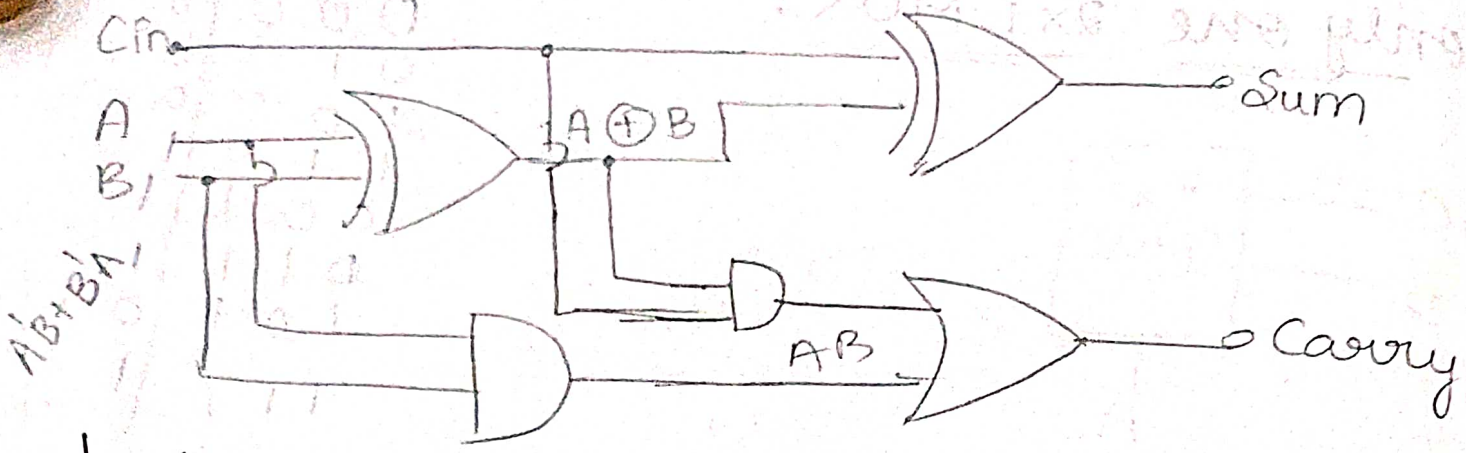
$S = \Sigma(1,2,4,7) = A \oplus B \oplus C$

$S = A \oplus B \oplus C$

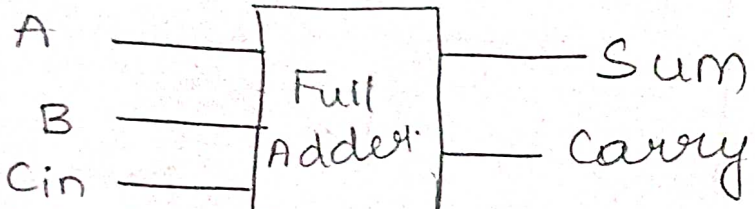
$C = \Sigma(3,5,6,7)$

$C = AB + BC + AC$

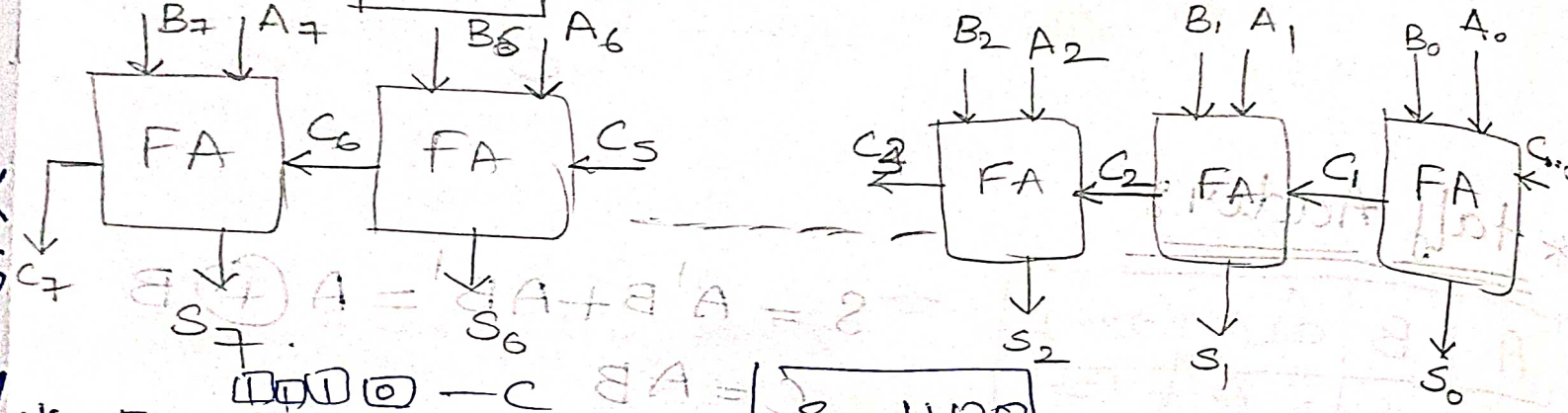




04/08/2022



8bit Adder



* 5 — 0101 — A
7 — 0111 — B.

C=0 1100

S = 1100
C = 0

* 15 — 1111 — A
7 — 0111 — B

1 0110 → 22.

S = 0110
C = 1.

⟹ Carry bit should be at the beginning as well as it is the most significant bit.

also called overflow.

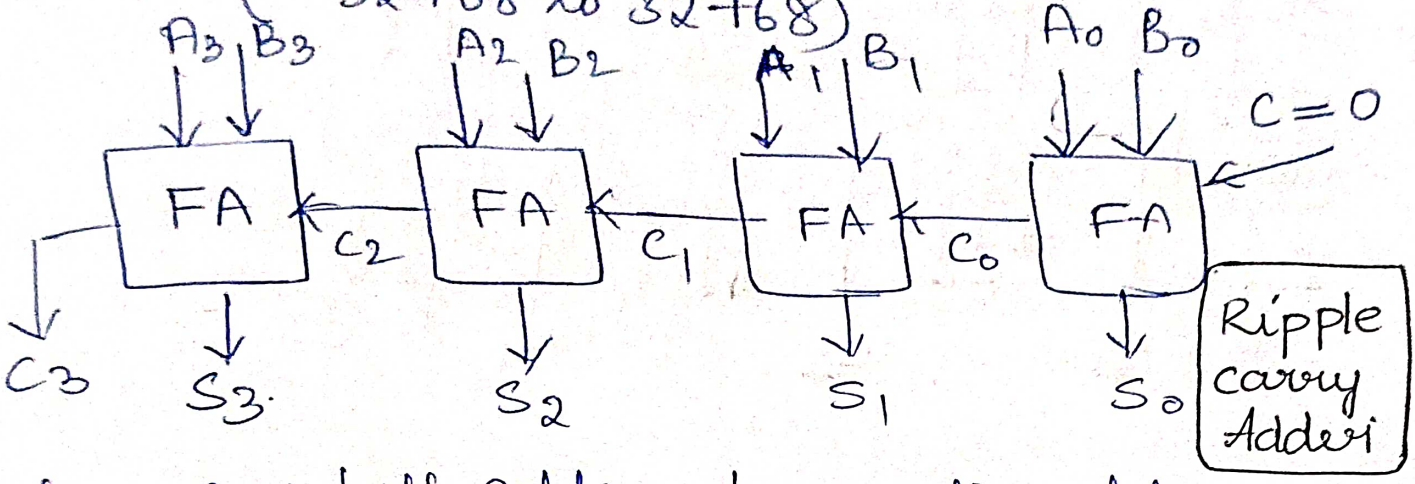
* Overflow: When the result exceeds

the no. of bits.

→ 15 bits
↳ signed bit

0 – 65536

Range
(−32768 to 32768)



* et for one full adder propagation delay = 2ms.

$S_0 = 2ms$
$S_1 = 4ms$
$S_2 = 6ms$
$S_3 = 8ms$

} Not very much optimal.

* Carry Look-Ahead Adder:

$C_{in} = 0$

Propagator: $A_i \oplus B_i$

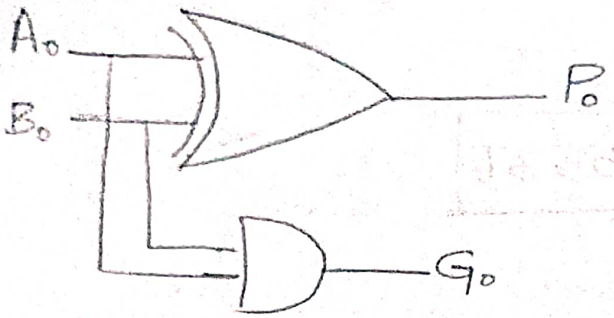
Generator: $A_i B_i$

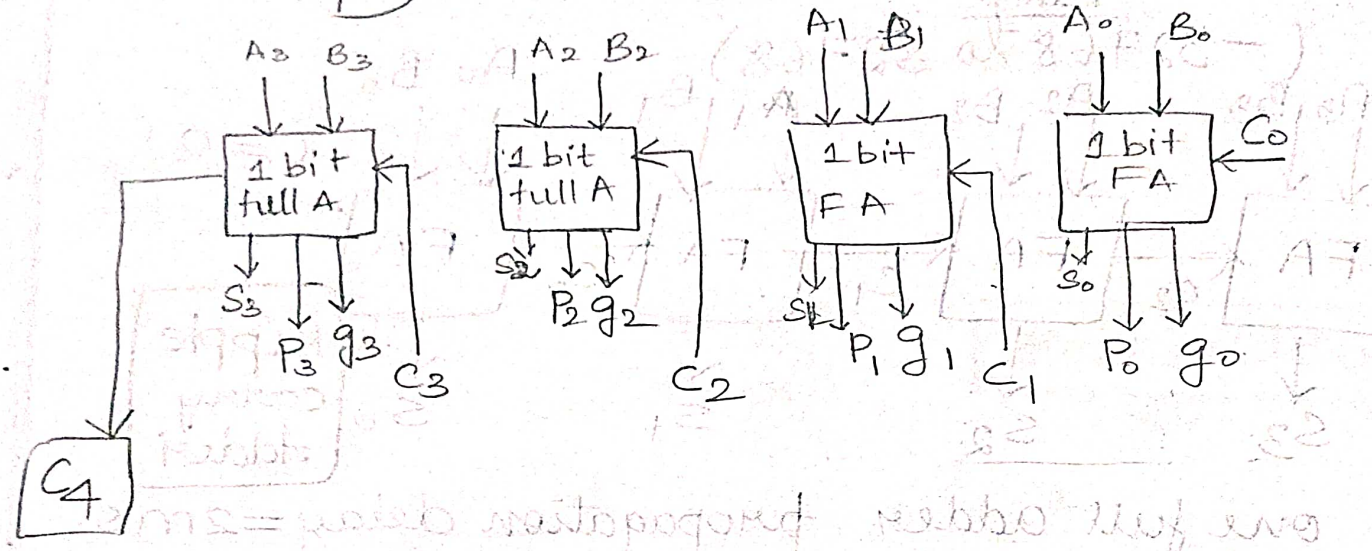
$C_0 = A_0B_0 + C_{in}(A_0 \oplus B_0)$

$C_1 = A_1B_1 + C_0(A_1 \oplus B_1)$

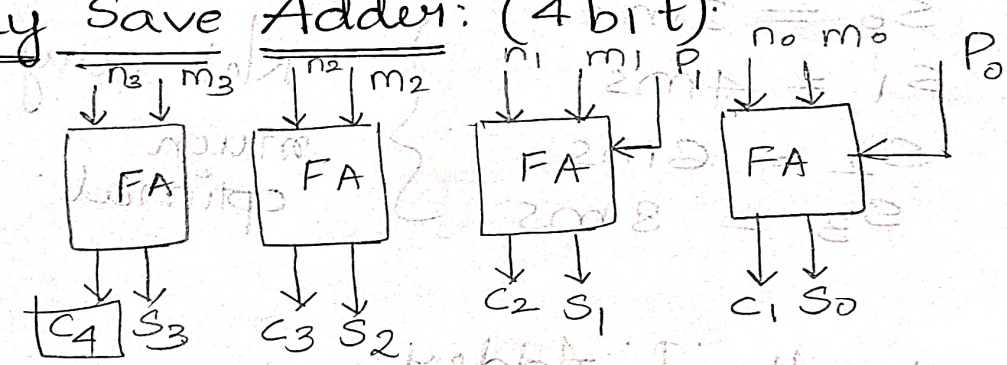
$C_2 = A_2B_2 + C_1(A_2 \oplus B_2)$

$C_3 = A_3B_3 + C_2(A_3 \oplus B_3)$





* Carry Save Adder: (4 bit)



2 – 0 0 1 0
5 – 0 1 0 1
7 – 0 1 1 1
0 0 0 0
0 1 1 1
14 – 0 1 1 1 0

Addition of Sum + carry.

– We can add 3 integer bit numbers. can be added in single go

K. SIVANI LST

FULL SUBTRACTOR:

A	B	C	Difference (D)	Borrow
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

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

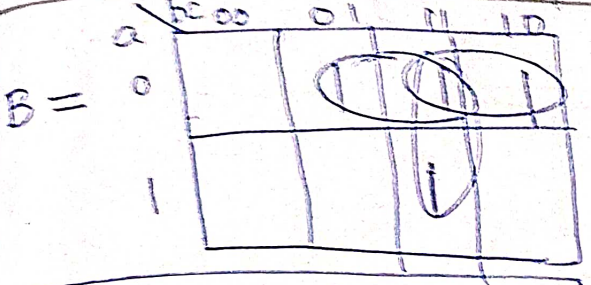
$B = \sum m(1, 2, 3, 7)$

$D = A'B'C + A'BC' + AB'C' + ABC$

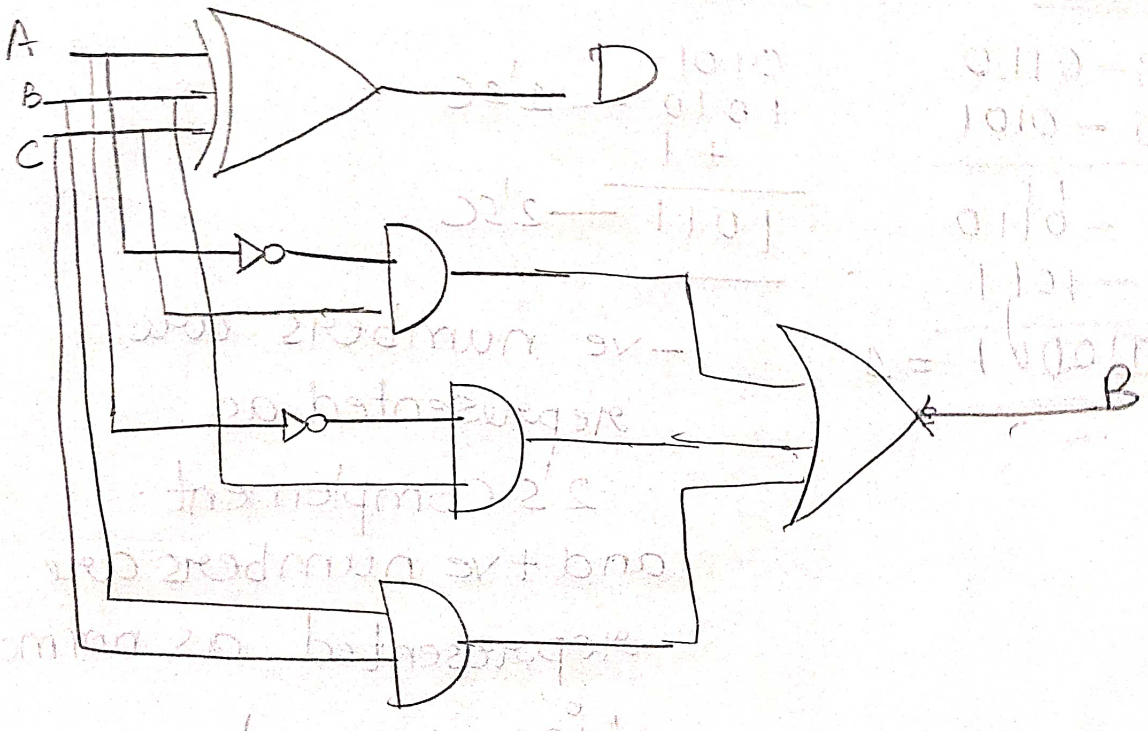
$= C(A'B' + AB) + C'(A'B + AB')$

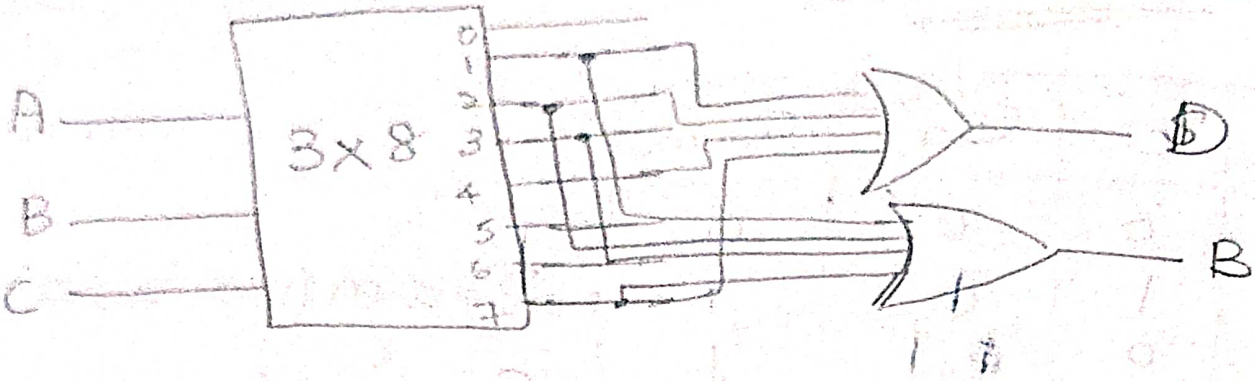
$= C \overline{(A \oplus B)}$

$D = A \oplus B \oplus C$

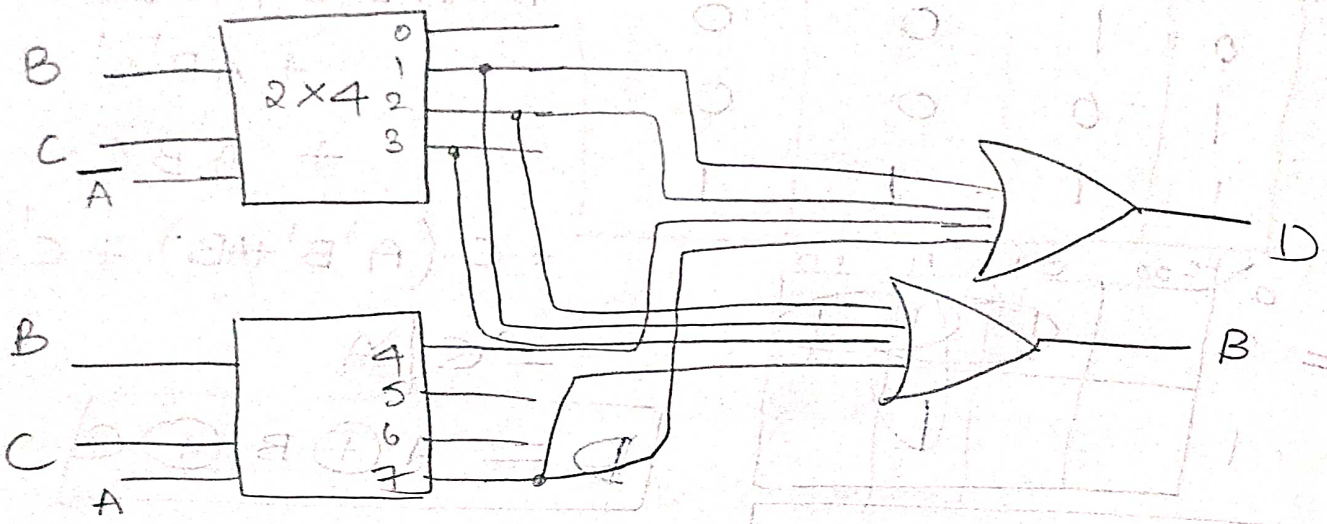


$B = A'C + A'B + BC$





2x4 decoders



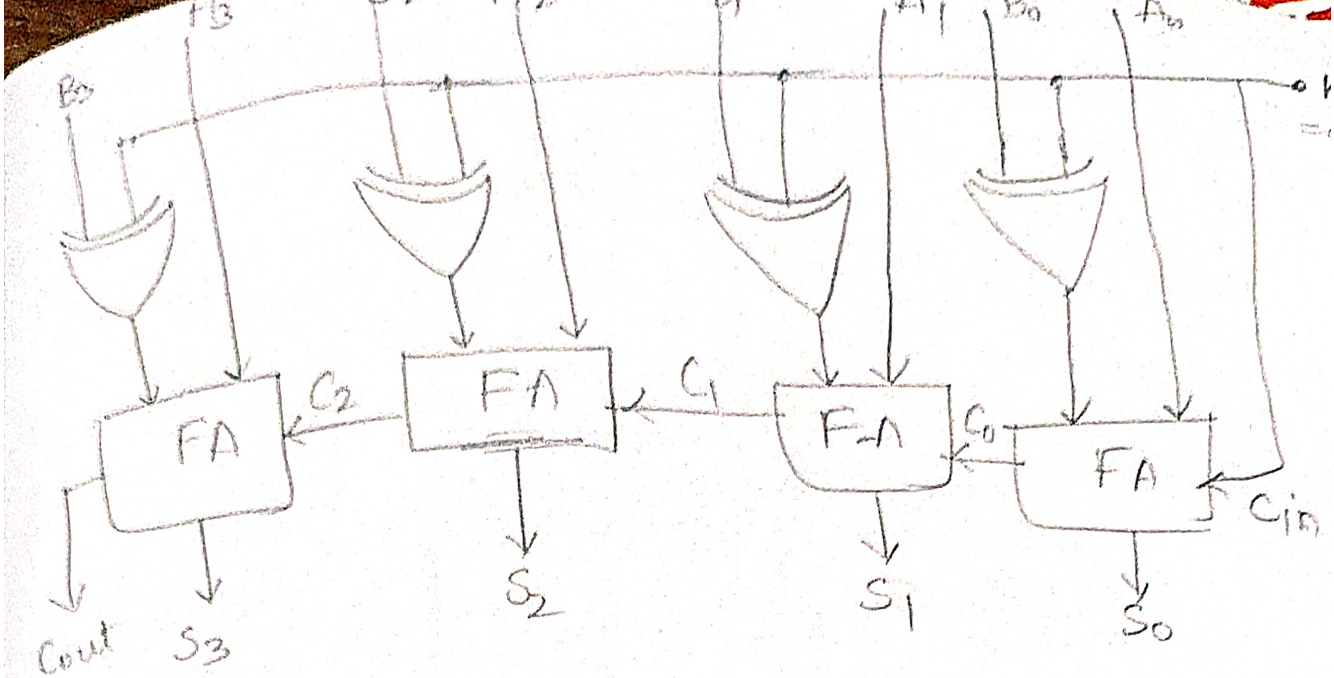
* ADDER & SUBTRACTOR:

```
  6 - 0110
- 5 - 0101
-----------
    1 1
  6 - 0110
- 5 - 1011
-----------
 1 0001 = 1.
```

```
 0101
 1010  — 1'sc
   +1
-------
 1011  — 2'sc
-------
```

-ve numbers are represented as 2's complement. and +ve numbers are represented as normal binary numbers.

K. SIVANI LST



$K = 0 \rightarrow$ adder

$K = 1 \rightarrow$ Subtractor