NUMBER SYSTEM (contd...)

9's and 10's complement :—

- Subtraction of decimal no. can be obtained by the 9's and 10's complement methods
- The 9's complement of decimal no. obtained by subtracting each digit of that decimal no. from 9.
- 10's complement → adding 1 to its 9's comp.

eg:— 9's complement :—

3465

```
  9 9 9 9
- 3 4 6 5
---------
  6 5 3 4
```

↑ 9's complement of 3465

17 APRIL
WEDNESDAY • APRIL

2019 - MARCH

S	M	T	W	T	F	S
31						
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

10's complement
4069

9999
− 4069
5930 ← 9's comp. of 4069
+ 1
5931 ← 10's comp. of 4069

1056.074

9999.999
1056.074
8943.925
+1
8943.926

9's complement Method of subtraction:-

↳ obtain the 9's complement of the subtrahend and add it to the minuend.

↳ If there is a carry, it indicates the answer is positive. Add the carry to the LSD of the result to get the answer. This is called end around carry.

↳ If there is no carry, the ans is -ve. and the intermediate result is its 9's complement. Take the 9's complement of this result & place a -ve sign in front to get the ans.

eg:-

745.81

− 436.62

⇒

745.81

+ 563.37 (9's comple. of 436.62)

(1) 309.18 (Intermediate result)

↳ +1

309.19 (Ans).

Carry → +ve ans.

∴ Ans:- +309.19

436.62

−745.81

⇒

436.62

+ 254.18 (9's comp. of 745.81)

690.80 (Intermediate result, no carry)

↓

9's complement is 309.19

∴ Ans:- −309.19

10's complement method of Subtraction :—

↳ Obtain 10's complement of the subtrahend & add it to the minuend.

⇒ If carry, ignore

⇒ If carry, ans is +ve; the result obtained itself is the ans.

⇒ If no carry, ans is -ve, the result obtained is its 10's complement.

Representation of Signed Nos:—

① Signed magnitude form.

② Complement form → 1's
→ 2's

7 ——→ 111

+7 or -7 ——→ ?

'+' → 0

'-' → 1.

Signed Magnitude form :-

Syntax :- [Sign bit | Actual Binary]

eg: +7 [0 | 111]

-7 [1 | 111]

Complement form :-

↓

- 1's complement.
- 2's complement.

→ complementing each & every digit of binary decimal no.

9 → 1001

↓ 1's comp.

0110

→ Adding 1 to 1's complement.

9 → 1000

↓

0 110 (1's complement)

+ 1

---

0 1 1 1 (2's complement)

1's complement Representation of signed no:-

Syntax:- | Sign bit | 1's complemat of actual binary |

(-7)

111

| 1 | 0 0 0 |

1's comp.

000

2's comp Representation

Syntax:- | Sign bit | 2's complemat of actual binary |

'-7' →

7 → 111

1's → 000

2's → +1

001

| 1 | 0 0 1 |

2's complement Subtraction:-

↳ Add 2's complement of the subtrahend to the minuend.

↳ If there is carry out, ignore it.

↳ Look at sign bit, i.e MSB of the sum term. If MSB is '0' result is +ve & if it is '1' result is -ve and is in its 2's complement form.

↳ Take its 2's complemant to find its magnitude in binary.

eg:- Subtract 14 from 46 using 8 bit 2's complement arithmatic.

+14 0000 1110

1111 0010

eg:- 48 − 28

48 + (−28)

48 → 110000

28 → 011100

↓

1's comp → 100011

+1

→ 100100

2's complement of 28

+110000

(1) 010100

Discard

BINARY CODES

- weighted
  - Binary
  - BCD
    - Excess-3
    - Gray
    - 5-bit BCD
    - 2421
    - 5211
    - Excess-3
- non weighted
  - 8421
  - Excess 3
- Sequential
- Reflective
- Alpha Numeric
  - ASCII
  - EBCDIC
  - Hollerith
- Error detecting & correcting
  - Parity
  - Hamming

- 8421
- 2421
- 3321
- 4221
- 5211'
- 5311
- 5421
- 6311
- 7421
- 7421
- $8\bar{4}\bar{2}\bar{1}$

Binary Coded Decimal (BCD) code :-

→ In this code each decimal digit is '0-9' represented by a 4-bit binary no.

Decimal no.

143

Decimal digits.

→ Positional weights are '8,4,2,1', so also called 8421 code.

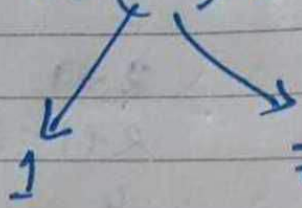
Decimal	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

4 bits — $2^4 = 16$

Invalid BCD as they are decimal no. & not decimal digit	10	xxxx
	11	xxxx
	12	⋮ ⋮
	13	⋮ ⋮
	14	⋮
	15	xxxx

↳ Conversion of Decimal No to BCD.

(i) $(17)_{10}$ ⇒ 00010111



0001 0111

(ii) 156 ⇒ ?

↳ BCD to Decimal :-

(i) 10100 BCD

0001 0100

1 4

$(14)_{10}$

Make group of 4 bits from right.

(ii) 1001001

0100 1001

$(49)_{10}$

↳ Comparision betⁿ Binary & BCD :—

	Binary	BCD
$(10)_{10} \rightarrow$	1010	00010000
$(12)_{10} \rightarrow$	1100	00010010

(147-218) WK 22

27 MONDAY • MAY

BCD Addition :-

1) Sum ≤ 9,	Final carry = 0	Ans ✓
2) Sum ≤ 9,	Final carry = 1	Ans X (To correct it add 0110)
3) Sum > 9,	Final carry = 0	Ans X (")

Perform simple binary addition. ✓

✓ $(2)_{10} + (6)_{10}$

```
  0010
  0110
------
  1000
```

Sum < 9

Final carry = 0.

✓ $(3)_{10} + (7)_{10}$

```
  0011
  0111
------
  1010
```

Sum > 9

Final carry = 0

We have to add 0110.

$2^4 = 16$

0 0.
: :
: :
: :
: :
9 15

$15 - 9 = 6$

Sum = 1010 → Binary

0110

0000 | 0000

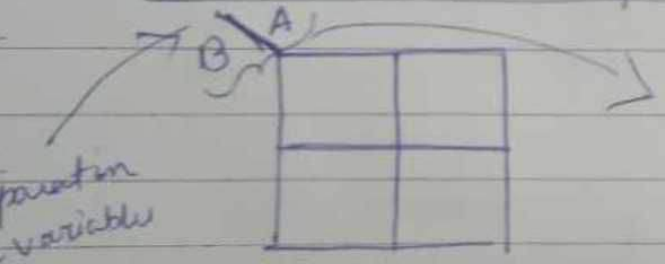
1 0

K-Map :— (Karnaugh Map)

→ Developed by Karnaugh in 1953

→ Used to simplify boolean algebriac expressions without using Boolean laws.

2-variable K-Map.



2^2 cells = 4 cells

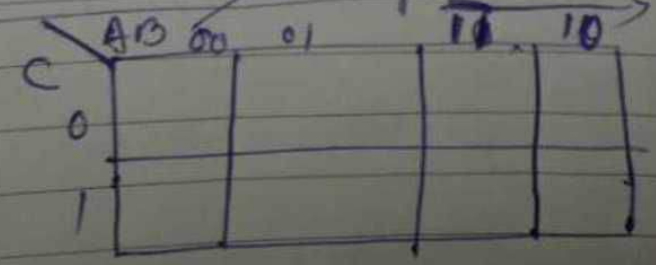
can interchange the positions.

Order " number of cells = 2^n "

n = no. of variable.

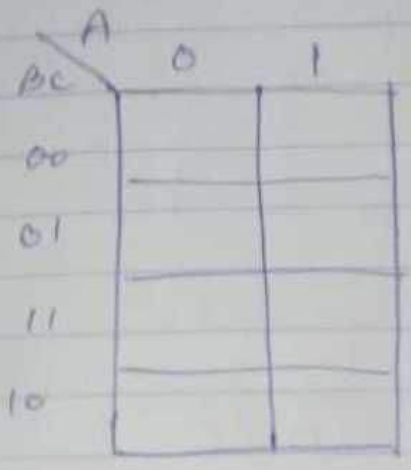
3 variable K-map:—

→ maximum variables at top side or Horizontal



2^3 = 8 cells

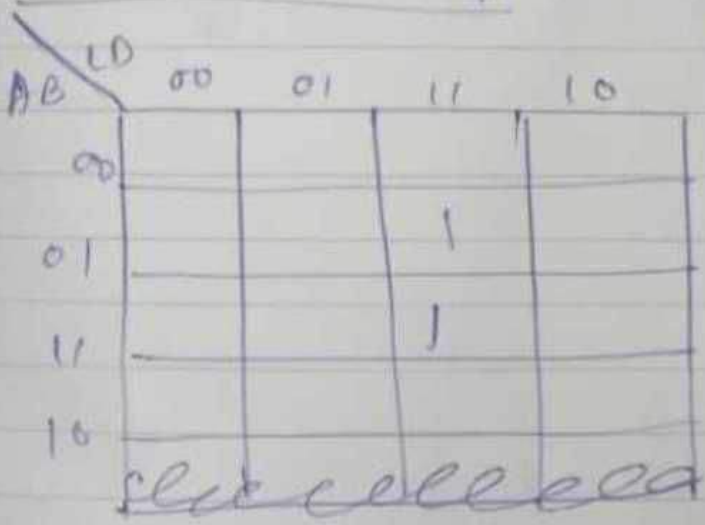
Horizontal representation



Vertical representation.

4-variable K-Map:—

$2^4 = 16$ cells.



ABCD = 1111

ABCD = 0111

Rules for K-Map Simplification :—

1. Groups may not contain zero.

2. We can group 1, 2, 4, & (or) 2^n cells.

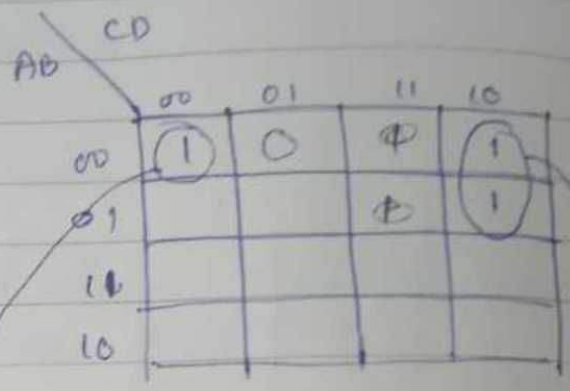
3. Each group should be as large as possible

4. cells containing 1 must be grouped

5. Groups may overlap.

6. Opposite grouping and corner grouping is allowed.

7. There should be as few groups as possible.



→ prime implicant.

$\bar{A}\bar{B}\bar{C}\bar{D}$

A is having same value in both cells so consider only A

$A\bar{C}\bar{D}$

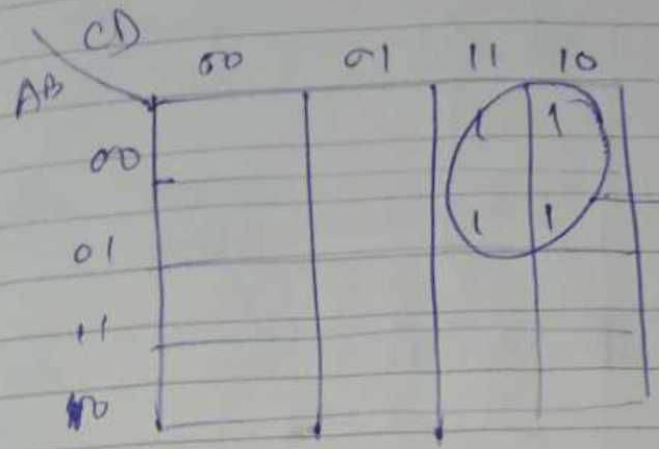
For SOP, groups may not contain 0.

group single variable

2 "

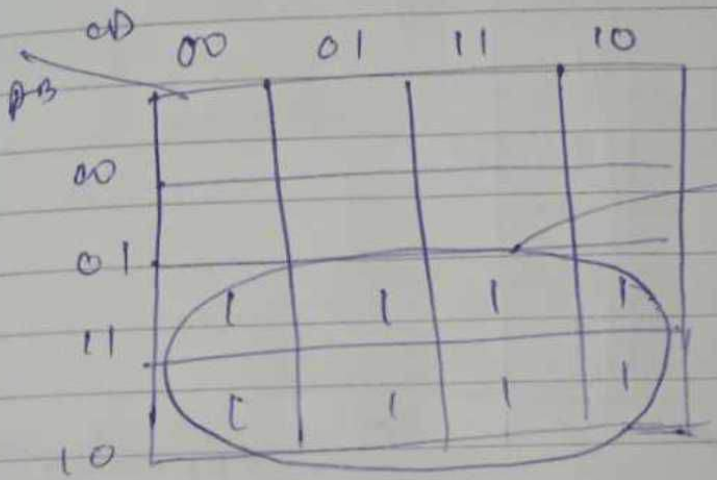
Quarter (4 variables)

Octate (8 variables)

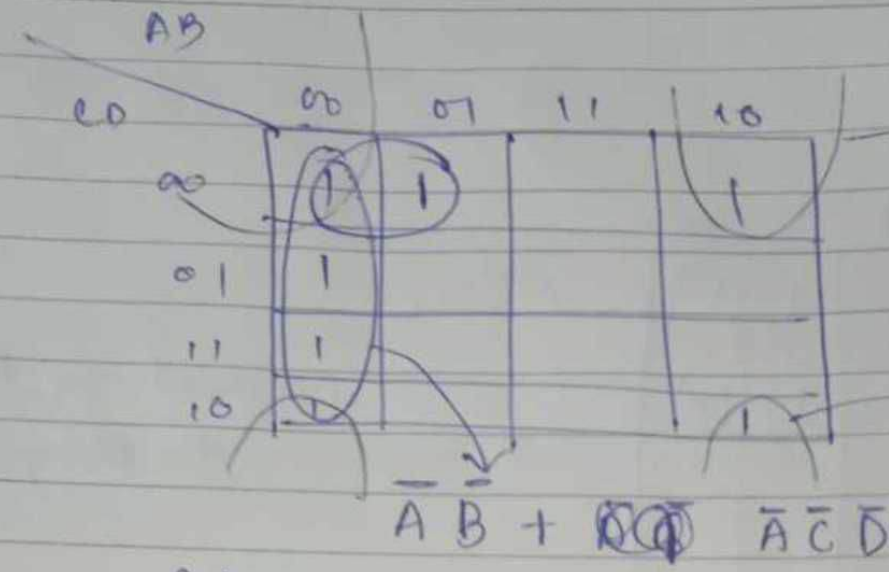


$\bar{A}C$

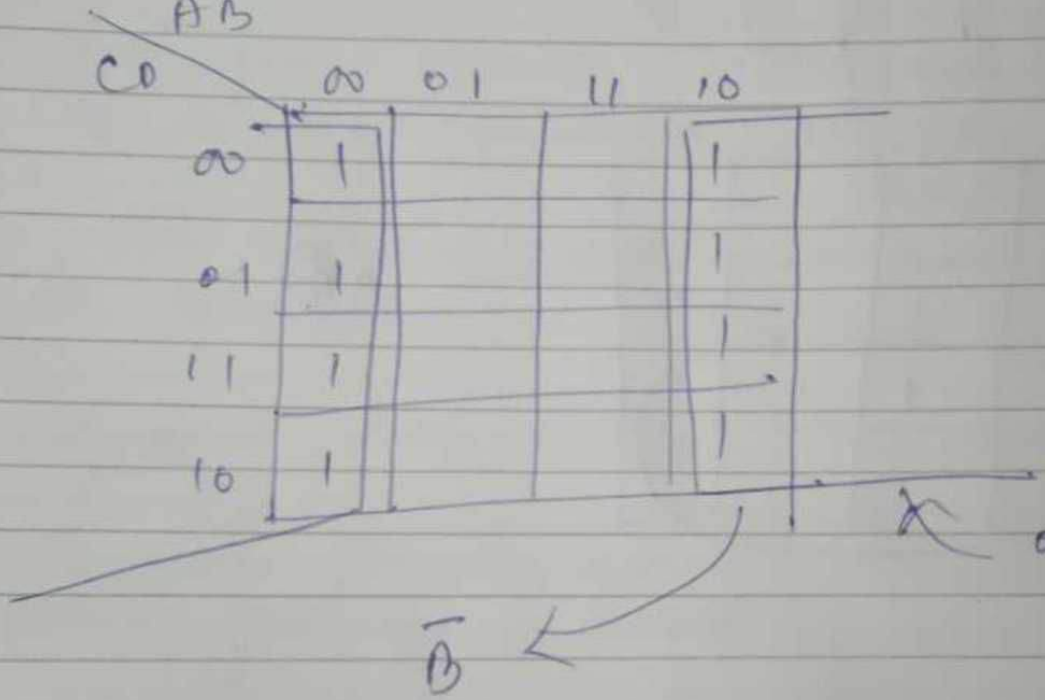
no need to take double grouping.



A'



See first if octate is poss then quadra then



In the sum of product function $f(x, y, z) = \Sigma(2, 3, 4, 5)$

The prime implicants are (Gate 2007)

(a) $\bar{x}y$, $x\bar{y}$

(b) $\bar{x}y$, $x\bar{y}\bar{z}$, $x\bar{y}z$

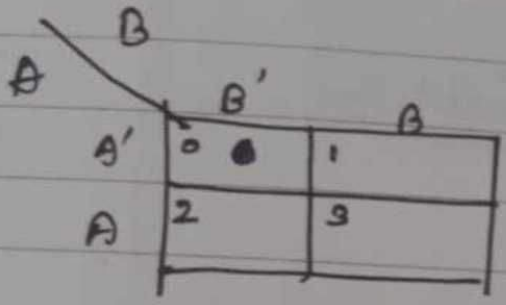
(c) $\bar{x}y\bar{z}$, $\bar{x}yz$, $x\bar{y}$

(d) $\bar{x}y\bar{z}$, $\bar{x}yz$, $x\bar{y}\bar{z}$, $x\bar{y}z$.

K- Map Simplication :-

① Two- variable :-

Decimal	A	B
0	0	0
1	0	1
2	1	0
3	1	1



② Three- variable :-

A \ BC	B'C' (00)	B'C (01)	BC (11)	BC' (10)
A'0	0	1	3	2
A 1	4	5	7	6

D	A BC
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

③ ~~Three~~ Four variable K-Map :-

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

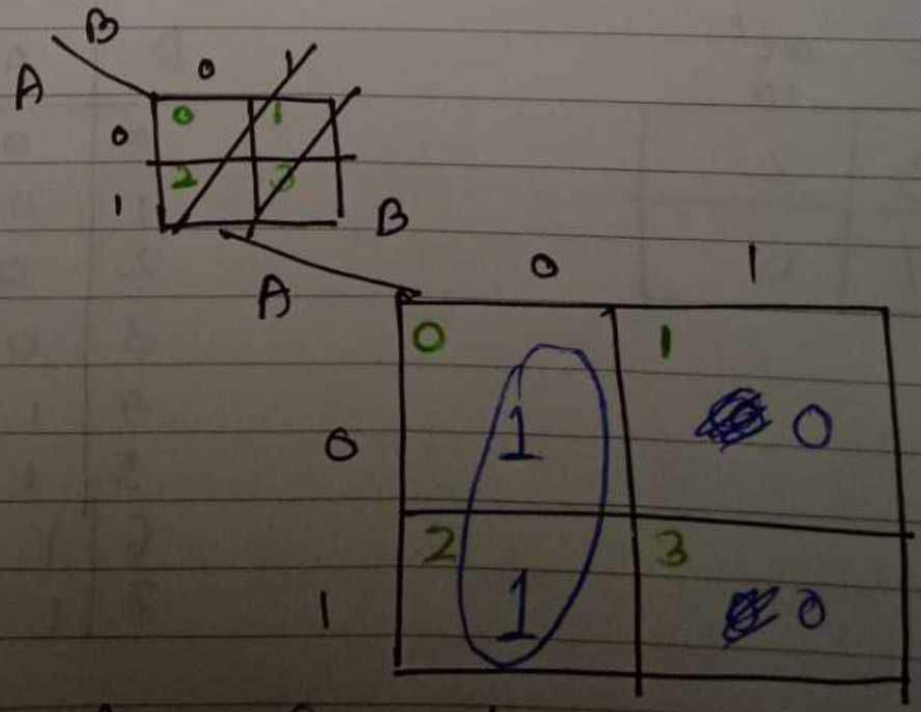
① Example for 2-variable :-

$$f = \Sigma (0, 2)$$

or

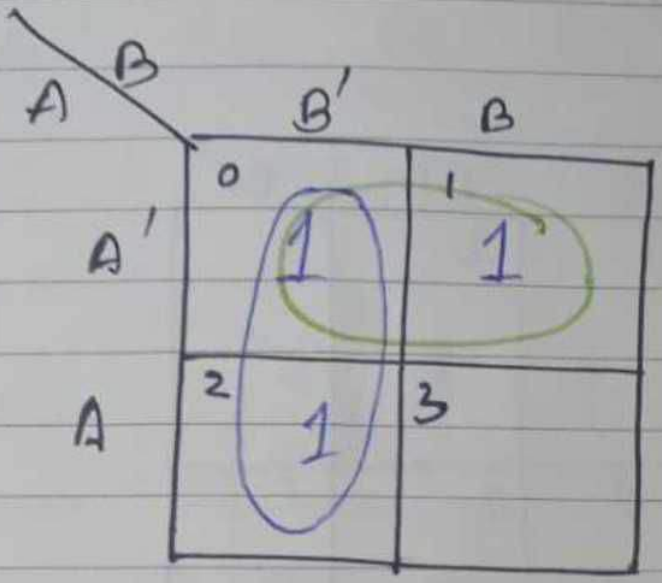
$$f = A'B' + AB'$$

00 10



Ans:- $f = B'$

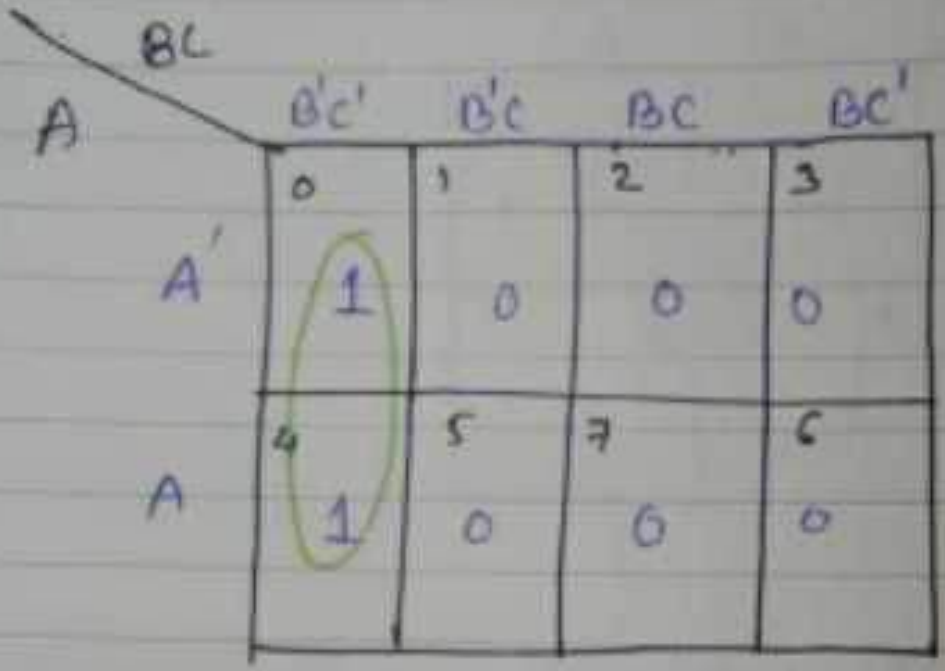
② $f = \Sigma(0, 1, 2)$



$f = B' + A'$

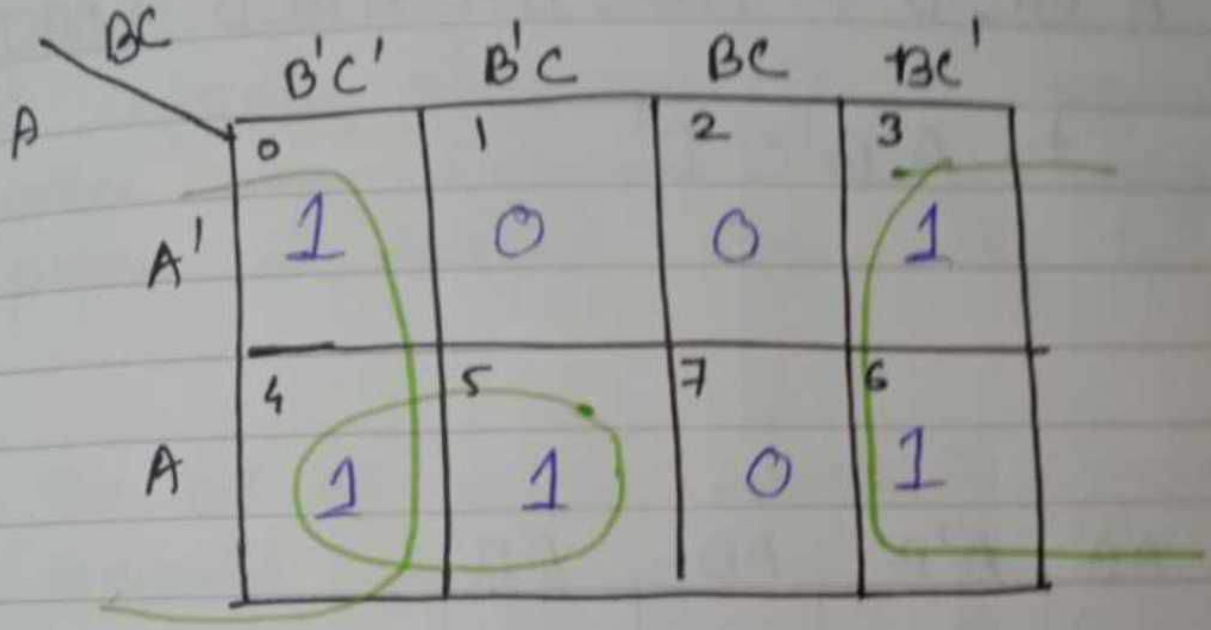
Example for 3-variable :—

① $f = A'B'C' + AB'C'$



$f = B'C'$

② $f(A, B, C) = \Sigma(0, 2, 4, 5, 6)$



$$f = AB' + C'$$

24 WEDNESDAY • JULY

(3) ~~f= A'B'C~~

$f = A'BC'D + ABC'D + A'BCD + ABCD + A'B'C'D'.$

(205-160) WK 30

24 WEDNESDAY • JULY

③ ~~f= A'B'C'~~

$f = A'BC'D + ABC'D + A'BCD + ABCD + A'B'C'D'.$

CD \ AB	A'B'	A'B	AB	AB'
C'D'	1	0	0	0
C'D	0	1	1	0
CD	0	1	1	0
CD'	0	0	0	0

$f = A'B'C'D' + BD$

JULY • SATURDAY 27

K- map simplification for POS form:-

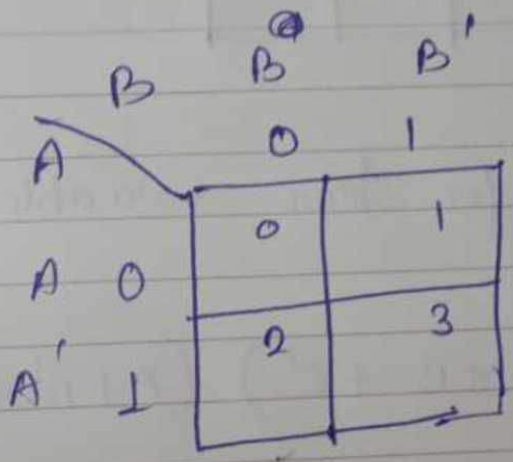
POS → Maxterm

0 → A
1 → A'

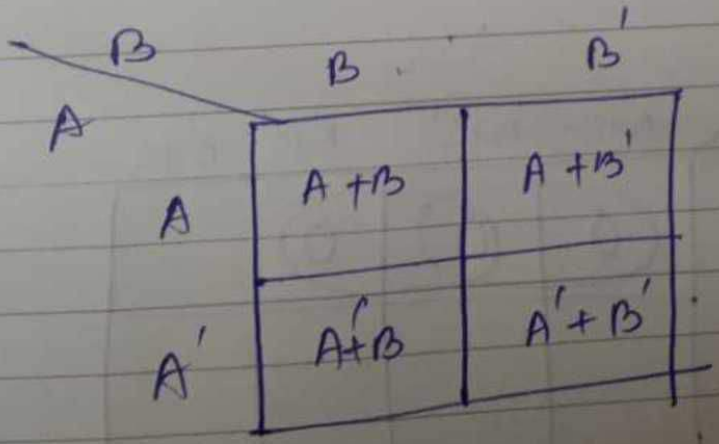
Minterm

0 → A'
1 → A

Two variable



D	A	B	Maxterm
0	0	0	A+B
1	0	1	A+B'
2	1	0	A'+B
3	1	1	A'+B'



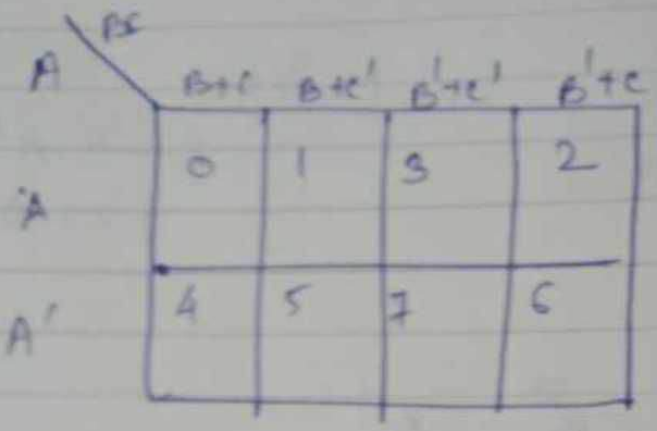
AUGUST

WK 30 (209-156)

SUNDAY 28

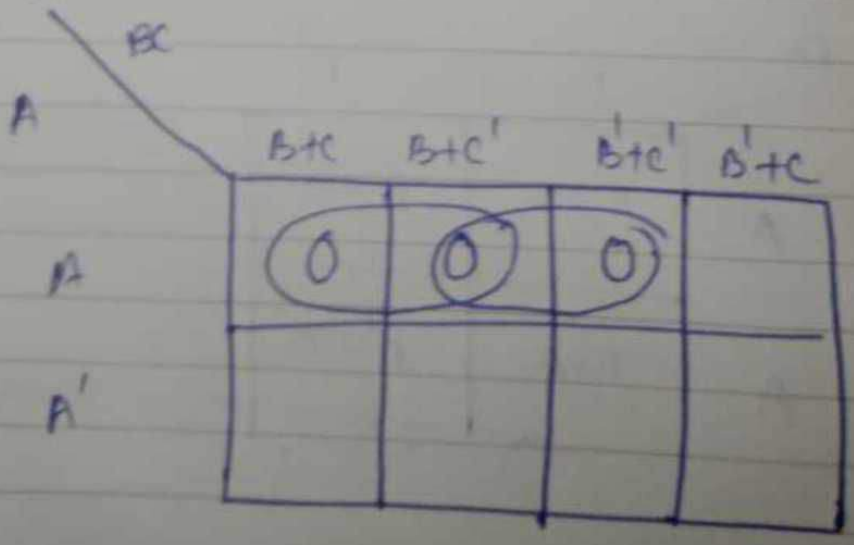
29 MONDAY • JULY

② Three variable



③ Similarly for four variable . . . . same process

Q. $f = (A+B+C')(A+B'+C')(A+B+C)$



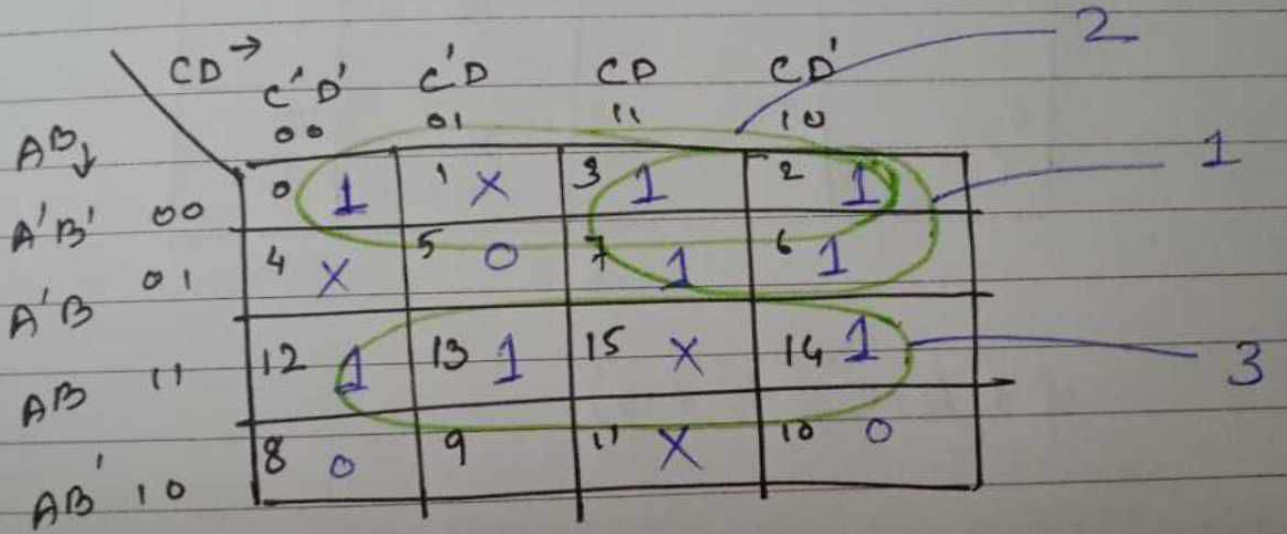
$f = (A+B)(A+C')$

K- Map (Don't Care Condition)

① $f(A, B, C, D) = \sum m (0, 2, 3, 6, 7, 12, 13, 14) + \sum d (1, 4, 11, 15)$

↑ don't care cond^n.

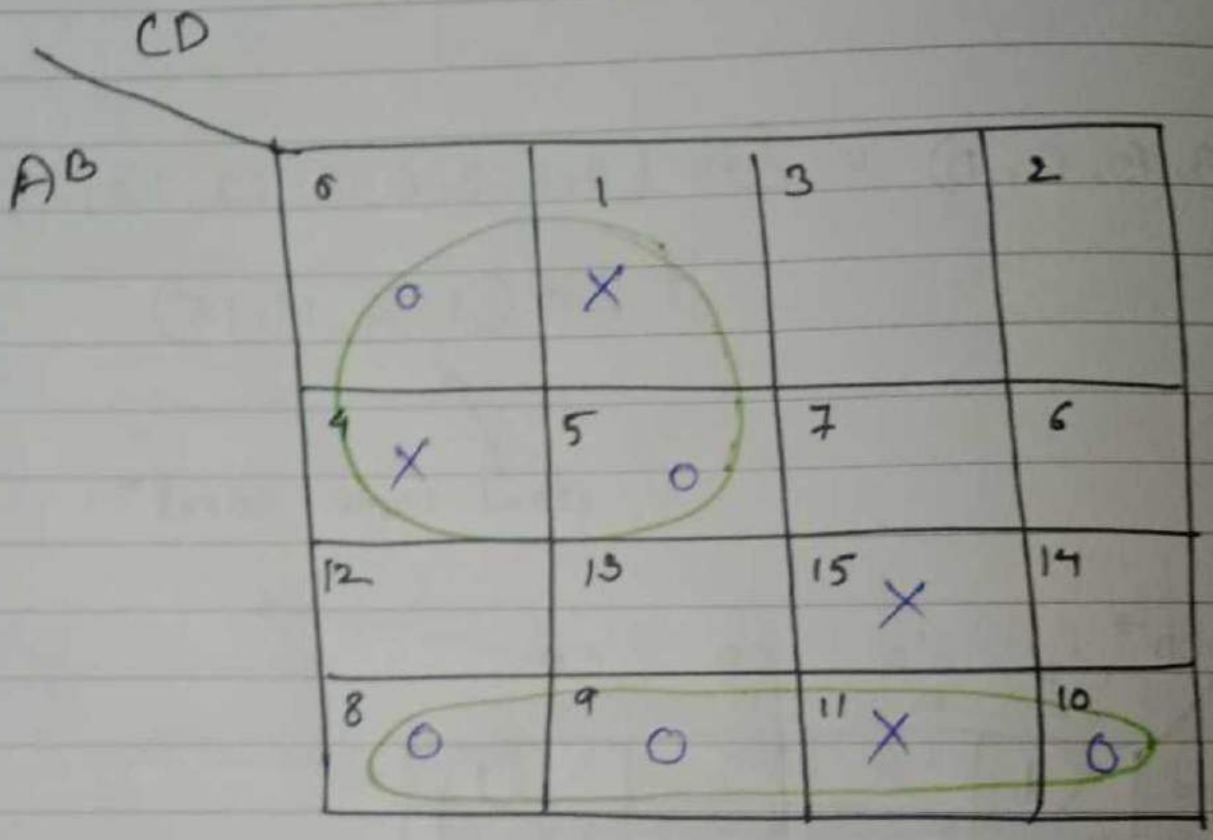
SOP



$f = A'B' + CA' + AB$

10 SATURDAY • AUGUST

② $f(A, B, C, D) = \pi M(5, 8, 9, 10)\ \pi D(1, 4, 11, 15)$



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

Questions

(1) Reduce using mapping the expression

$$f = \Sigma m (2, 3, 6, 7, 8, 10, 11, 13, 14)$$

(2) Reduce the following expression to the simplest possible POS & SOP forms

$$f = \Sigma m (6, 9, 13, 18, 19, 25, 27, 29, 31) + d (2, 3, 11, 15, 17, 24, 28)$$

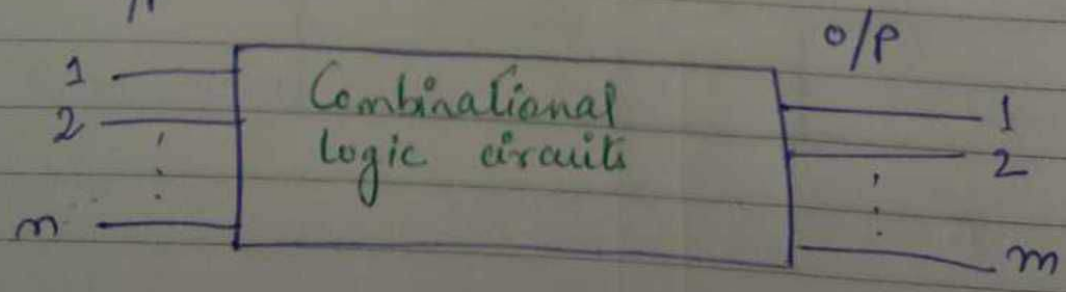
COMBINATIONAL CIRCUITS

↳ output is dependent upon combination of input variables.

↳ Does not use any memory.

↳ It can have 'n' no. of inputs & 'm' no. of outputs.

↳ eg:- Adders & subtractors
Decoders
Comparators
MUX
Code Converters
ROM
PLA
PAL



COMPARISION BETWEEN COMBINATIONAL & SEQUENTIAL CKTS

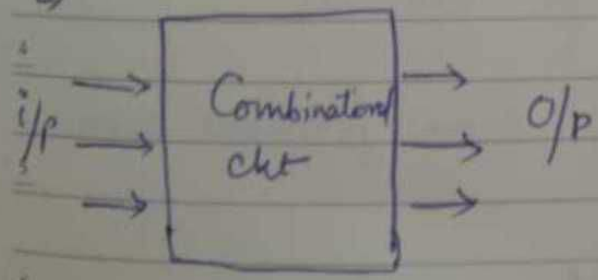
Combinational

↳ O/p is dependent on the present input

↳ ex:- Adder

$$\begin{array}{r} 1 \\ +\,0 \\ \hline 1 \end{array}$$

↳



Sequential

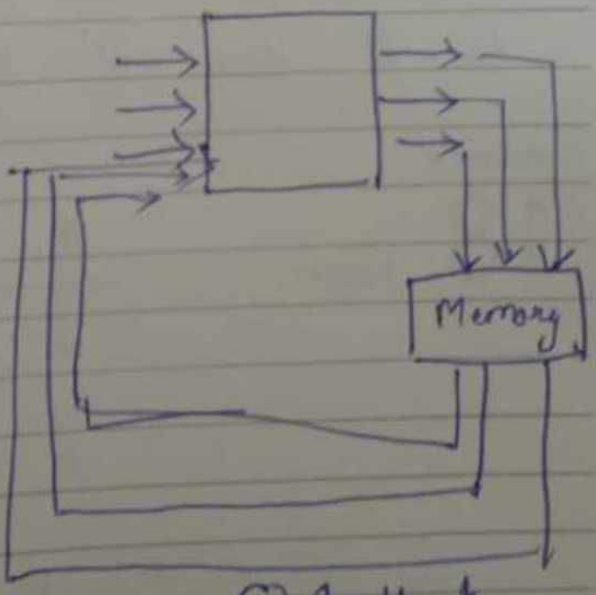
↳ O/p dependents on the present i/p as well as previous O/p or O/ps

↳ Counter (ex)

+1 to the previous O/p

If o/p is 5 that means previous O/p was 4 to which 1 is added

↳

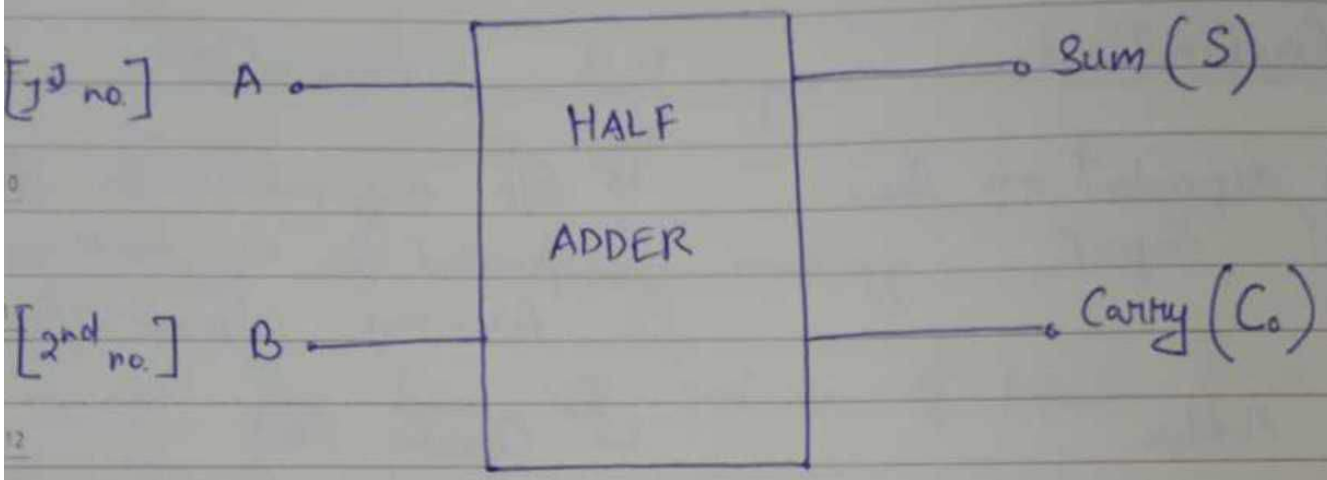


⊕ Feedback.

(229-136) WK 33

17 SATURDAY • AUGUST

HALF ADDER



$S = A \oplus B$

$C_o = A.B$

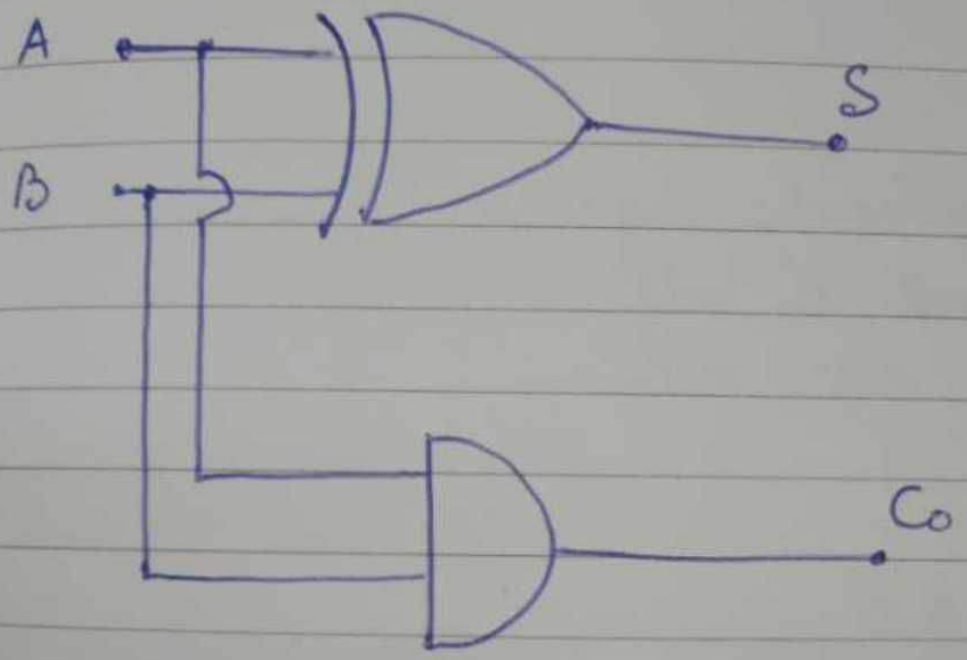
- used to add a single bit nos.
- It does not take carry from previous sum.

(230-135) WK 33

18 SUNDAY

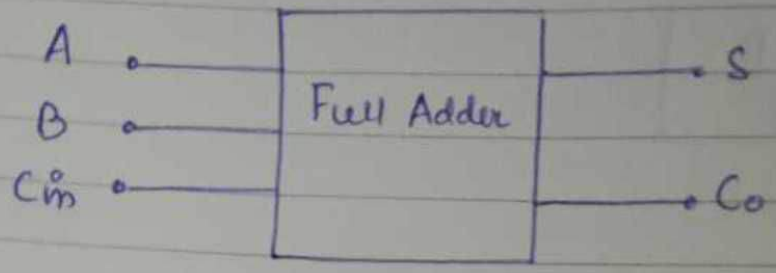
A	B	S	C_o	
0	0	0	0	Binary Addition
0	1	1	0	
1	0	1	0	
1	1	0	1	

$$S = A \oplus B$$
$$C_o = A \cdot B$$



20 TUESDAY • AUGUST

FULL ADDER :-



↳ Full Adder is a arithmatic logic circuit ~~dog~~ designed to add two single bit nos. with a carry.

A 0 0
B 0 1
C 0 1
① 0

T.T.

A	B	Cin	S	Co
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

Sum

A \ BCin	00	01	11	10
0	0	(1)	0	(1)
1	(1)	0	(1)	0

Checkboard configuration

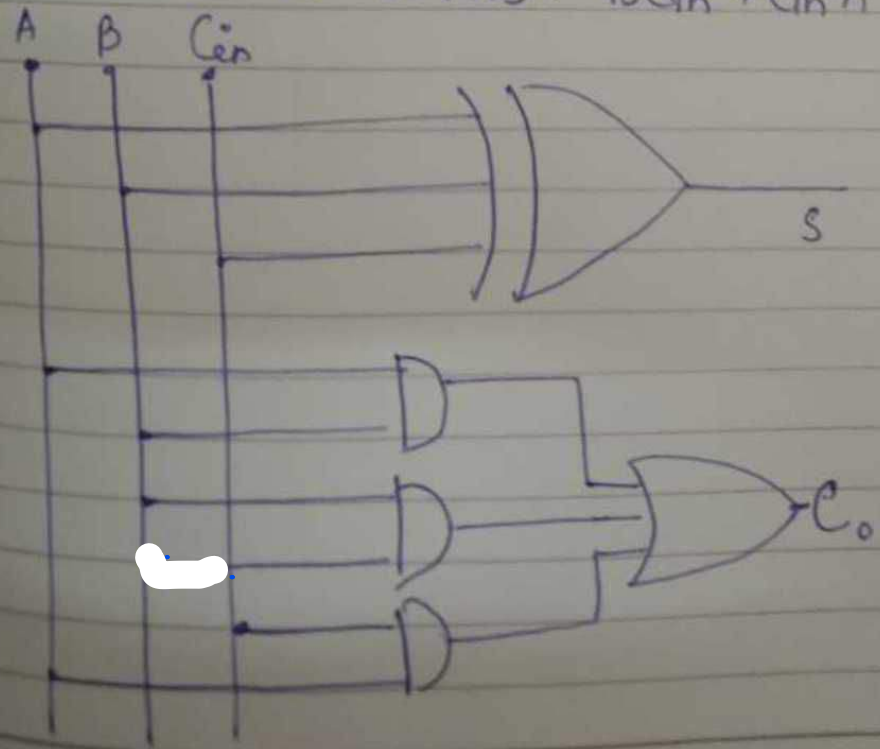
$S = A \oplus B \oplus C_{in}$

Carry out

A \ BCin	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$C_o = BC_{in} + AB + AC_{in}$

$= = AB + BC_{in} + C_{in}A.$

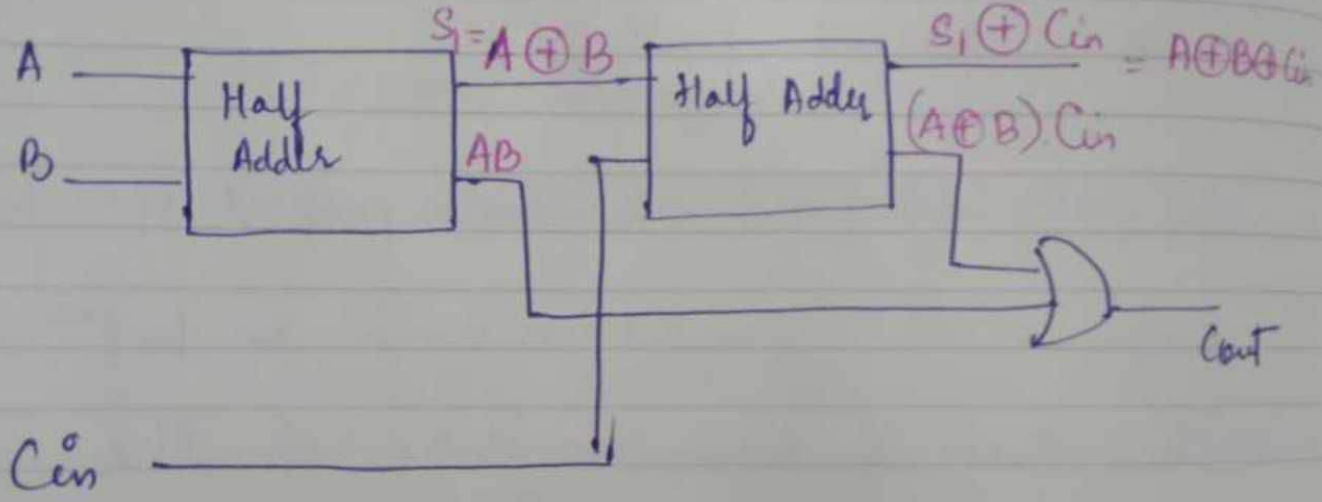


31 SATURDAY • AUGUST

COMBINATIONAL CIRCUITS:

Full Adder using Half Adder:-

* If full Adder not available & we have only half adders. we can design full Adder using two half Adders.



Half Adder $S_1 = A \oplus B$

Carry $= A.B.$

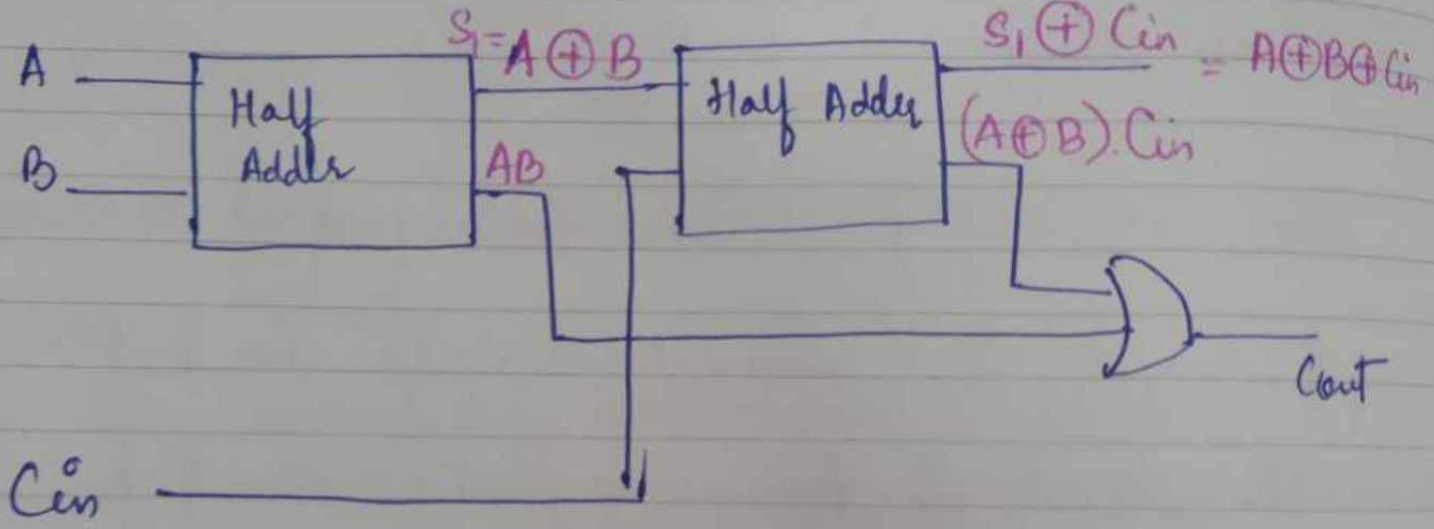
$C_{out} = (A \oplus B) C_{in} + AB$

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

COMBINATIONAL CIRCUITS:

Full Adder using Half Adder:-

* If full Adder not available & we have only half adders. we can design full Adder using two half Adders.



Half Adder

$S_1 = A \oplus B$

$Carry_1 = A.B.$

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

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

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

$= \bar{A}BC_{in} + A\bar{B}C_{in} + AB$

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

$= B(A + C_{in}) + \bar{B}AC_{in}$

(244-121) WK 35

01 SUNDAY • SEPTEMBER

$= AB + BC_{in} + \bar{B}AC_{in}$.

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

$= AB + (A+B)C_{in}$

$= AB + BC_{in} + C_{in}A$.

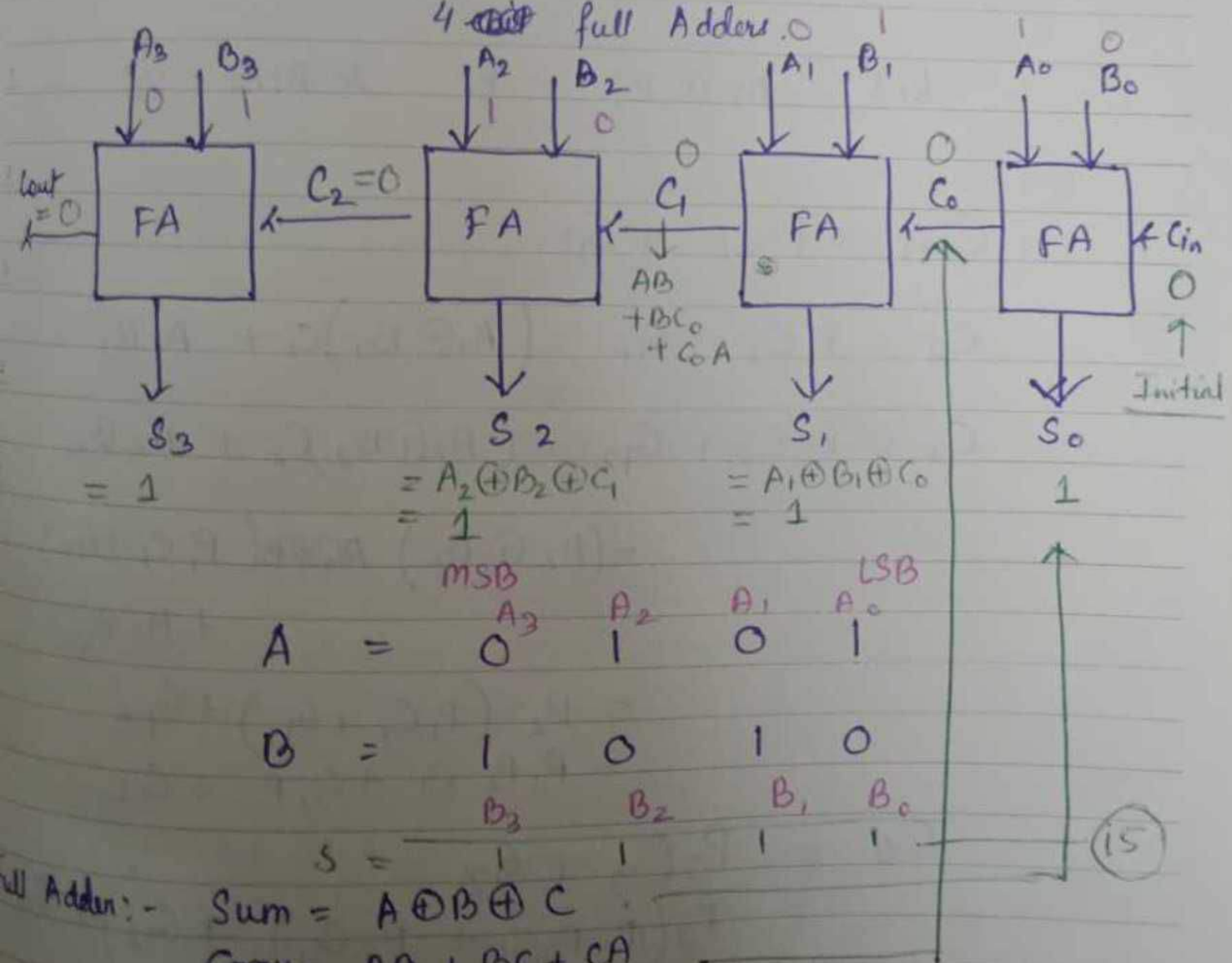
Self study

Subtractors

N Bit parallel Adder :—

- Cascade of Full Adder
- device for adding multi bit nos.

4-bit Parallel Adder :—



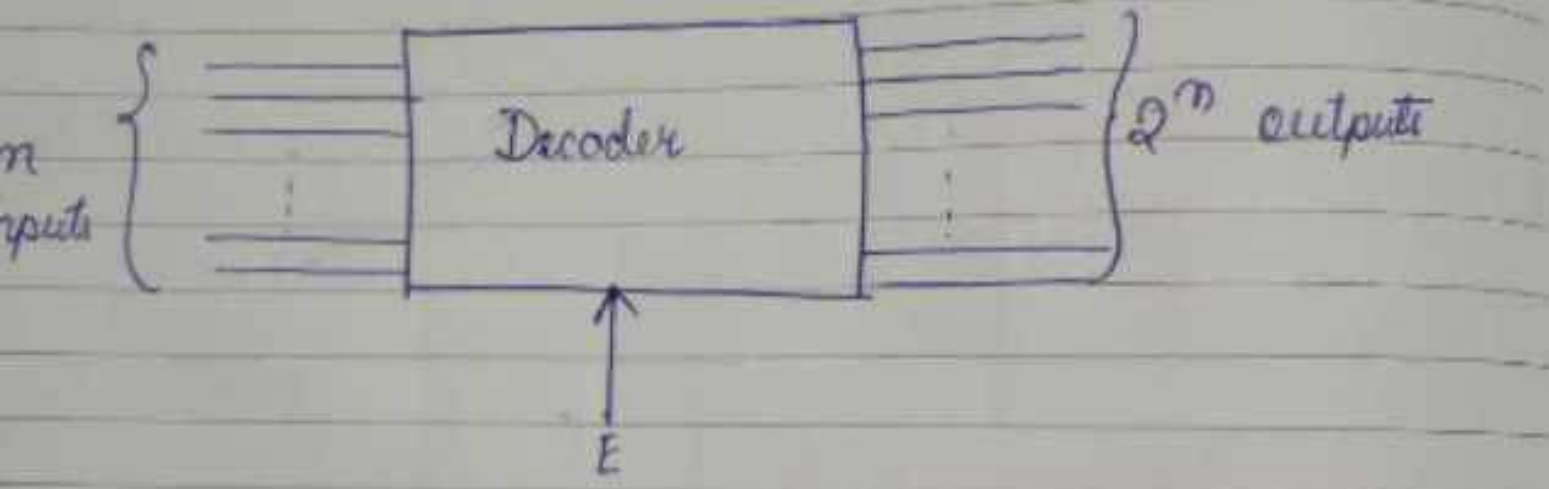
Full Adder :- Sum = A ⊕ B ⊕ C

Carry = AB + BC + CA

07 SATURDAY • SEPTEMBER

DECODERS

* Decoder is a multi-input, multi-output logic circuit which decodes 'n' inputs into 2^n possible outputs.



where E = Enable
- E=0, decoder is disabled. (OFF)
- E=1, decoder is enabled. (ON)

8 SUNDAY

Eg: If 2 inputs are provided we will get 2^n o/p

2 to 4 Decoder Design :-

2 i/p , 4 o/p

$n=3 \rightarrow n=2 \rightarrow n=1 \rightarrow n=0$

$2^3 \quad 2^2 \quad 2^1 \quad 2^0$

E	A	B		Y_3	Y_2	Y_1	Y_0
0	X	X		0	0	0	0
1	0	0	$n=0$	0	0	0	1
1	0	1	$n=1$	0	0	1	0
1	1	0	$n=2$	0	1	0	0
1	1	1	$n=3$	1	0	0	0

If E=0, without knowing i/p, o/ps are '0'.

when $n=0$, 2^0 is enabled
When $n=1$, 2^1 place enabled
so - on.

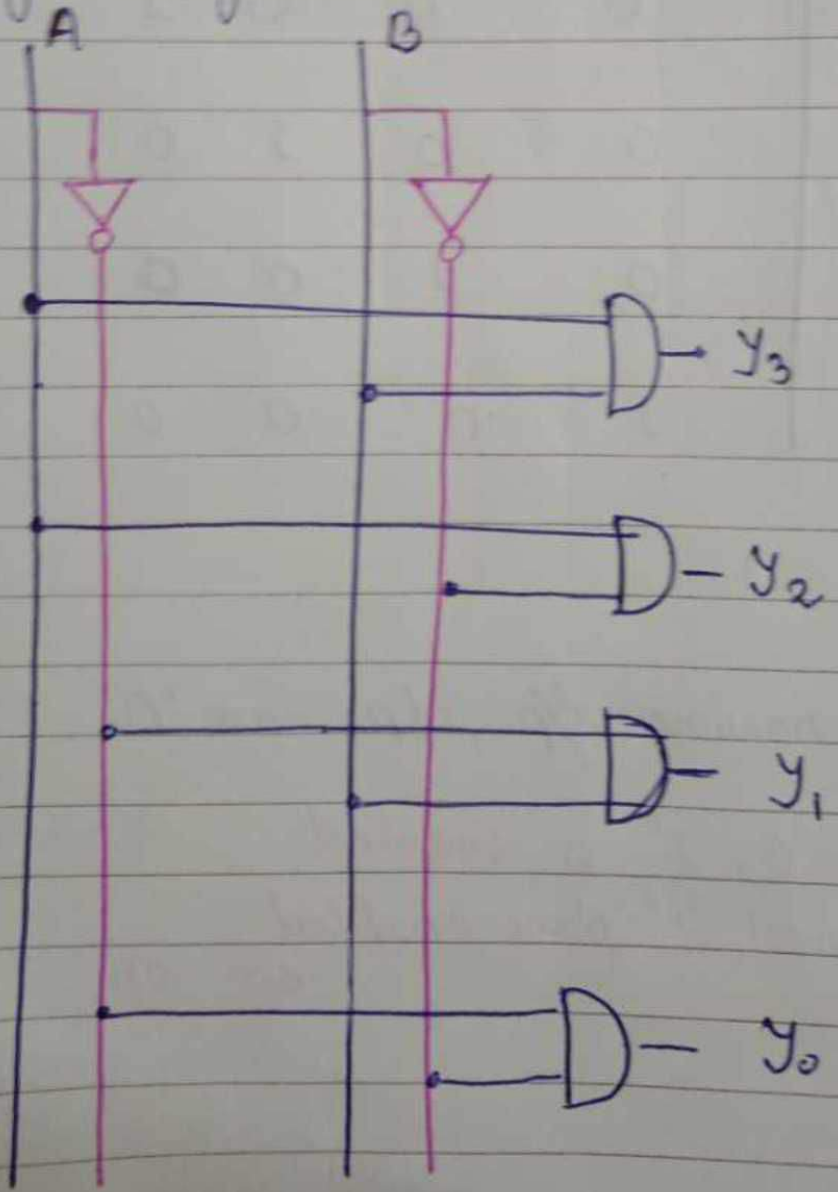
$Y_3 = A.B$

$Y_2 = A.\bar{B}$ (see the table)

$Y_1 = \bar{A}.B$

$Y_0 = \bar{A}.\bar{B}$

Logic Diagram:-

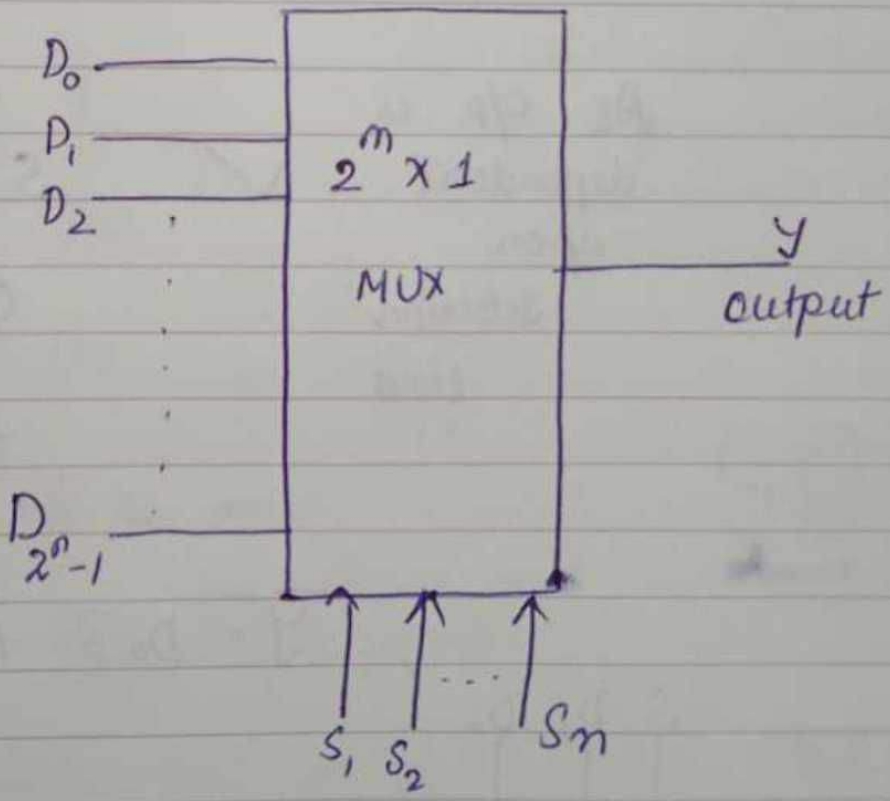


3 to 8 Decoder (Self Study)

$n=3$, $2^3 = 8$ o/ps.

MULTIPLEXER (MUX)

↳ Multiplexer is a combinational logic circuit used to select only one o/p (output) among several inputs based on selection lines.



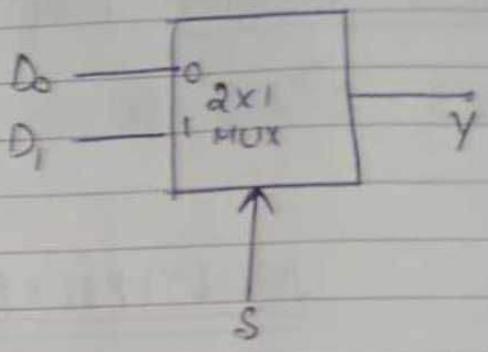
↳ This can act as digital switch.

↳ This is also called as data selector.

↳ For a MUX, there can be 2^n i/ps, 'n' selection lines and only one o/p.

12 THURSDAY • SEPTEMBER

2 x 1 MUX Design :-



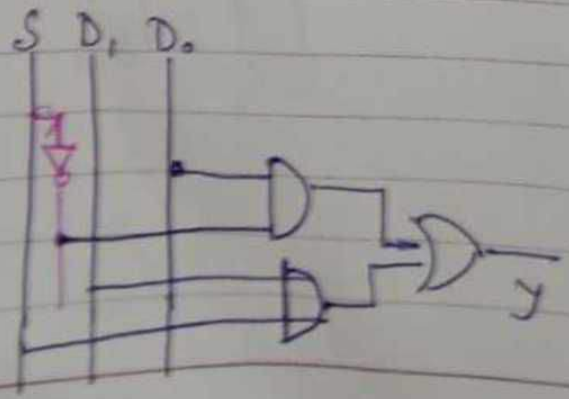
J.J.

D_0	D_1	S	Y
X			

As O/p is dependent upon selection line ✓

S	Y
0	D_0
1	D_1

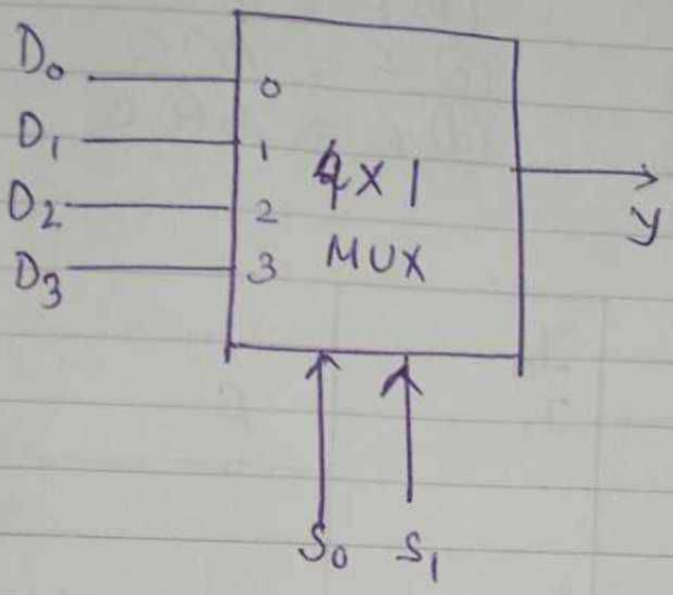
$$Y = D_0 \bar{S} + D_1 S$$



4×1 MUX :—

$2^n = 4$

$\therefore n = 2$



S_0	S_1	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

$$Y = D_0 \bar{S_0} \bar{S_1} + D_1 \bar{S_0} S_1 + D_2 S_0 \bar{S_1} + D_3 S_0 S_1$$

Logic Diagram → ?

(257-108) WK 37

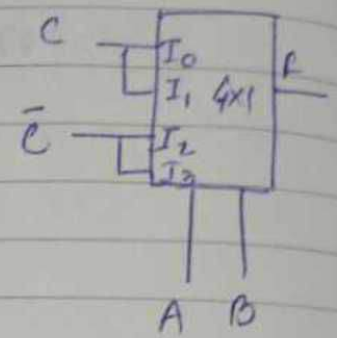
14 SATURDAY • SEPTEMBER

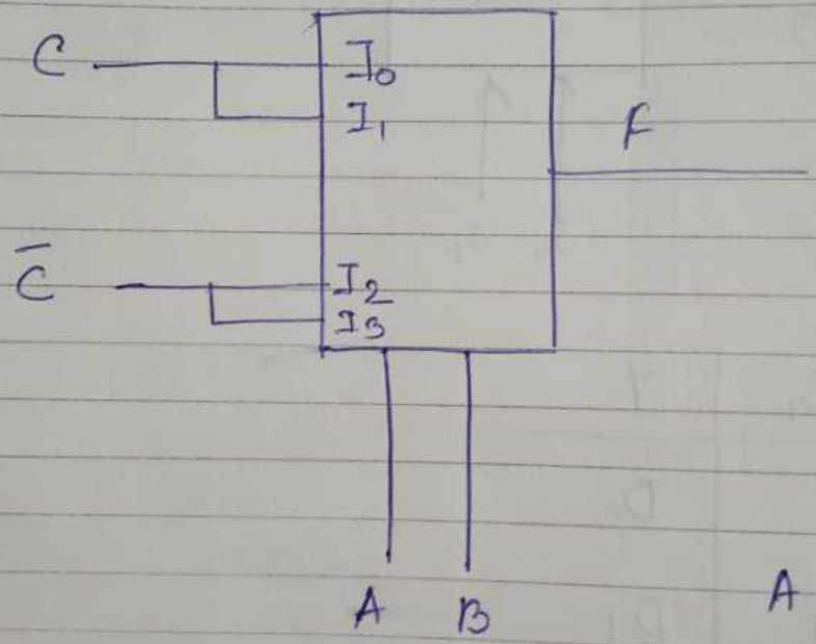
2019 AUGUST

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Q1 The logic realized by the circuit shown in the figure is —

(a) $F = A \odot C$
(b) $F = A \oplus C$
(c) $F = B \odot C$
(d) $F = B \oplus C$





A	B	F
0	0	C
0	1	C
1	0	$C\bar{C}$
1	1	$\bar{C}$

$F = C\bar{A}\bar{B} + C\bar{A}B + \bar{C}A\bar{B} + \bar{C}AB$

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

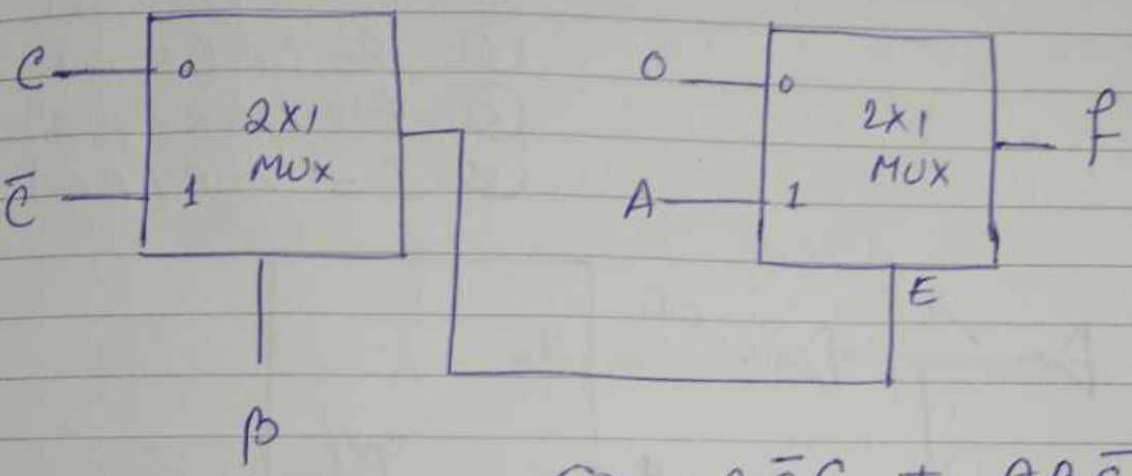
$= C\bar{A} + \bar{C}A$

$= A \oplus B$

(258-107) WK 37

15 SUNDAY

82. The Boolean 'f' implemented in the figure using Two input multiplexers is



(a) $A\bar{B}C + AB\bar{C}$

(b) $ABC + A\bar{B}\bar{C}$

(c) $\bar{A}BC + \bar{A}\bar{B}\bar{C}$

(d) $\bar{A}\bar{B}C + \bar{A}B\bar{C}$

1st MUX

B	E
0	C
1	$\bar{C}$

$E = C\bar{B} + \bar{C}B$

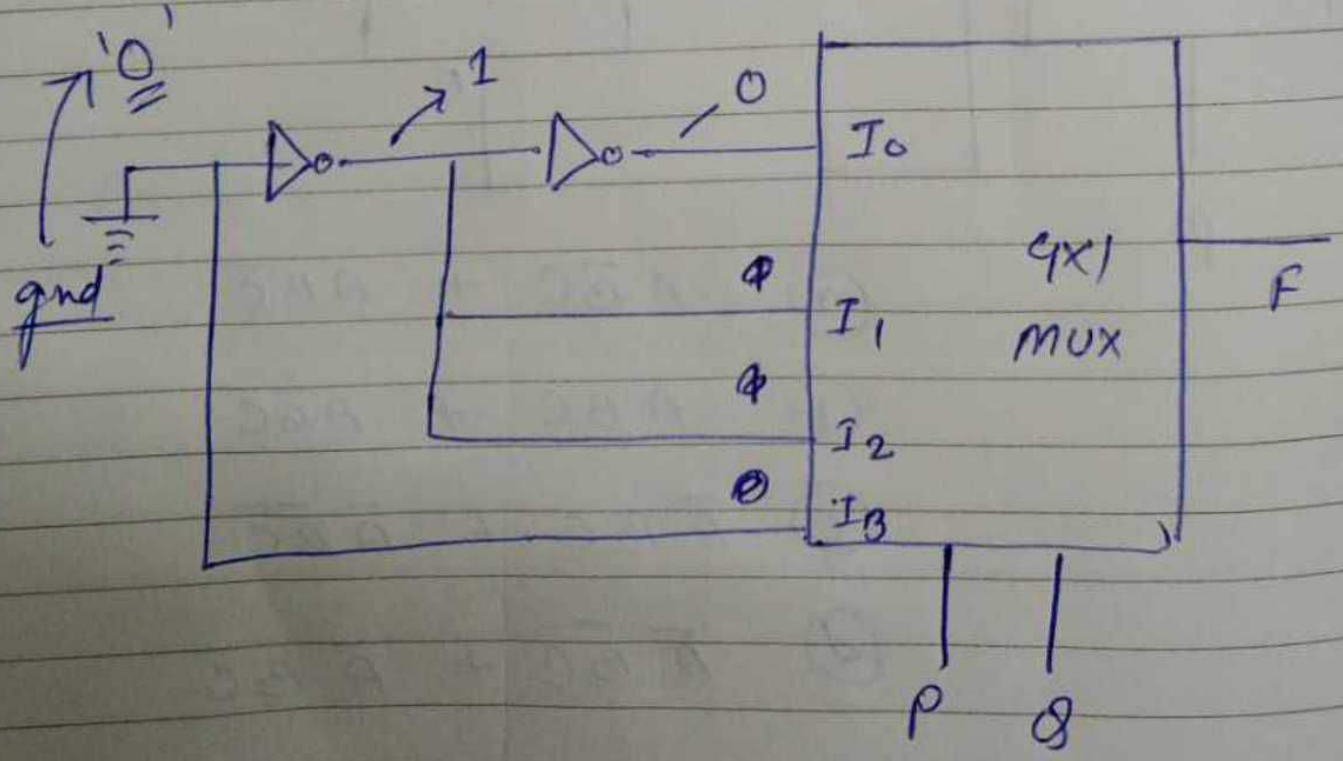
2nd MUX

E	f
0	0
1	A

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

Q3. The logic function implemented by the circuit below is —

(a) F = AND (P, Q)
(b) F = OR (P, Q)
(c) F = XNOR (P, Q)
(d) F = XOR (P, Q)



DEMULTIPLEXERS (DEMUX)

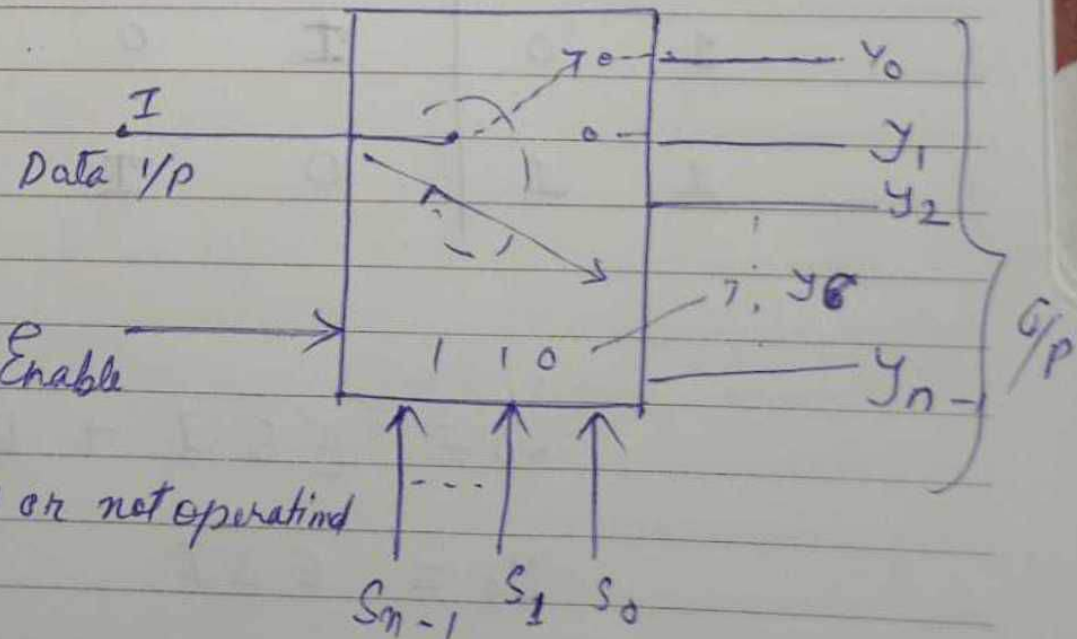
↳ one i/p and many o/ps.

↳ Reverse operation of MUX.

↳ Also called as Data Distributor.

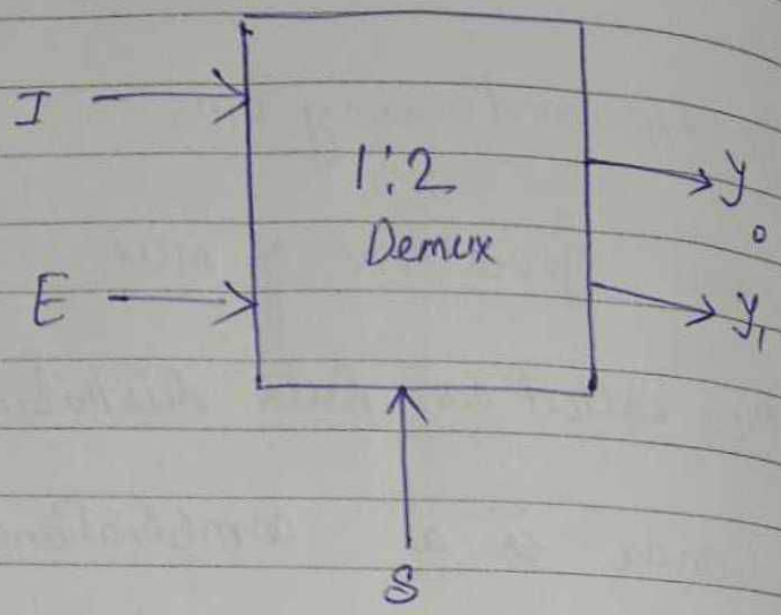
↳ Demux is a combinational ckt.

MUX	Demux
2:1 / 2x1	1:2 / 2x1
8:1 / 8x1	1:8
4:1	1:4



Select line 1 1 0 (6) the data i/p will go to Y6.

1 : 2 Demux



E	S	y_0	y_1
0	0	0	0
0	1	0	0
1	0	I	0
1	1	0	I

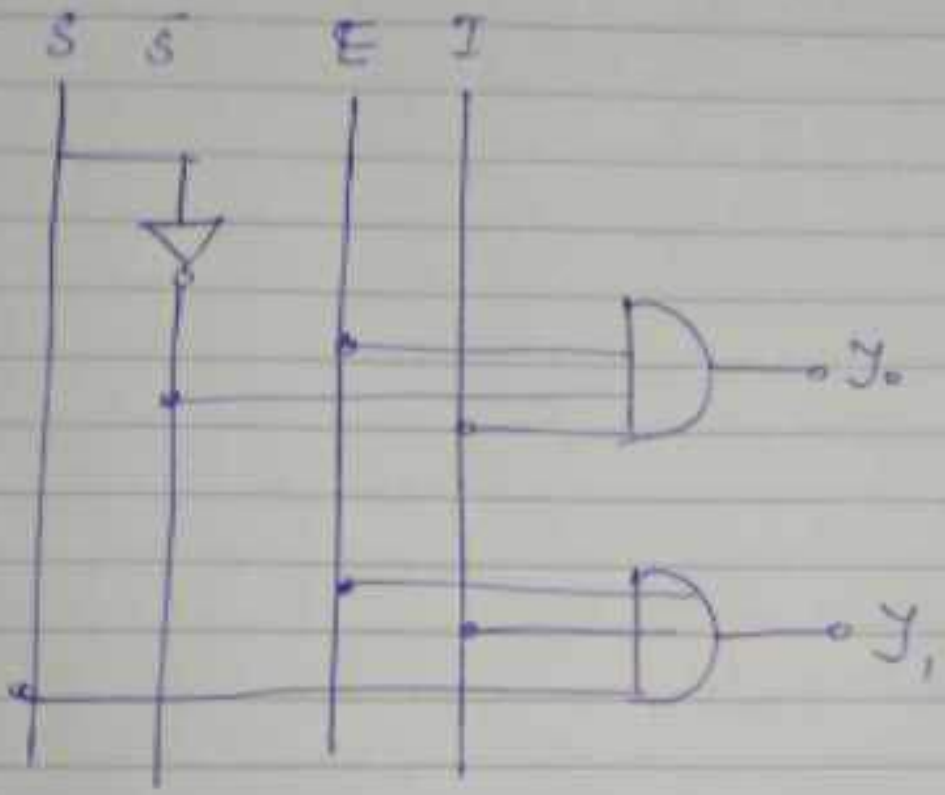
$$y_0 = E\bar{S}\cdot I + ES\cdot 0$$

$$y_0 = E\bar{S}I$$

$$y_1 = ESI$$

SEPTEMBER • FRIDAY 20

logic Diagram

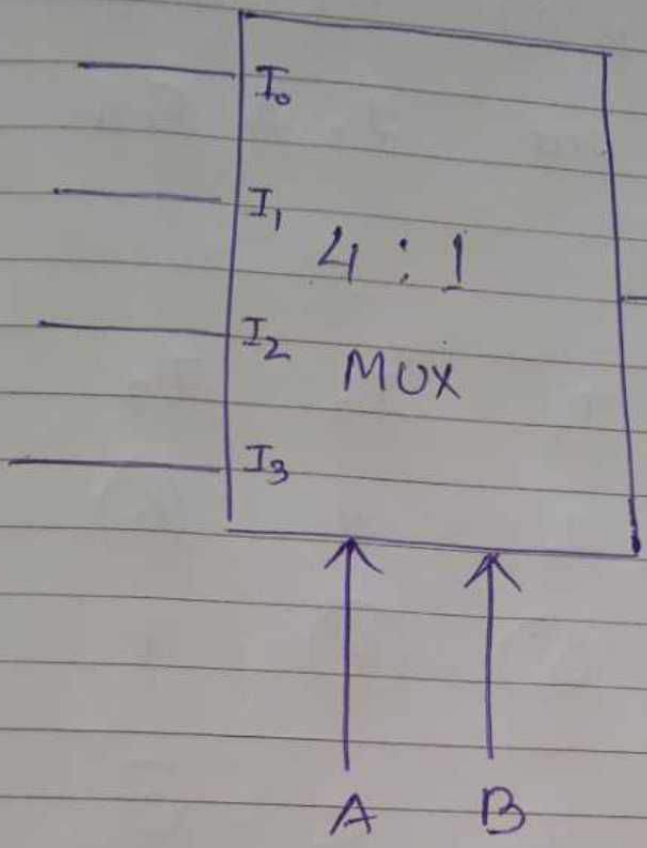


8. Implement the following fn using a MUX:-

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

Soln:-

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



∴ 3 variables given, 2 selection lines should be there.
2^2 i.e. 4:1 MUX

A	B	F
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

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

see where value of C = 0

C=1

Now see the f"

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

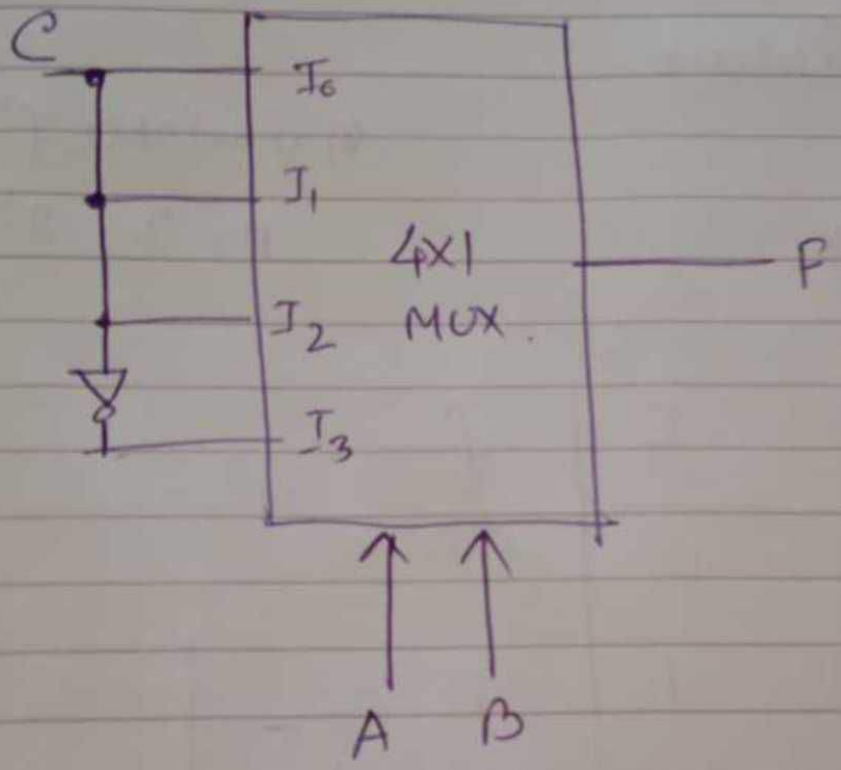
Circle out 1, 3, 5, 6

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

O O - X

O - O X

1 $\bar{C}$ C 0


C
I_0
I_1
4X1
MUX
F
I_2
I_3
A
B

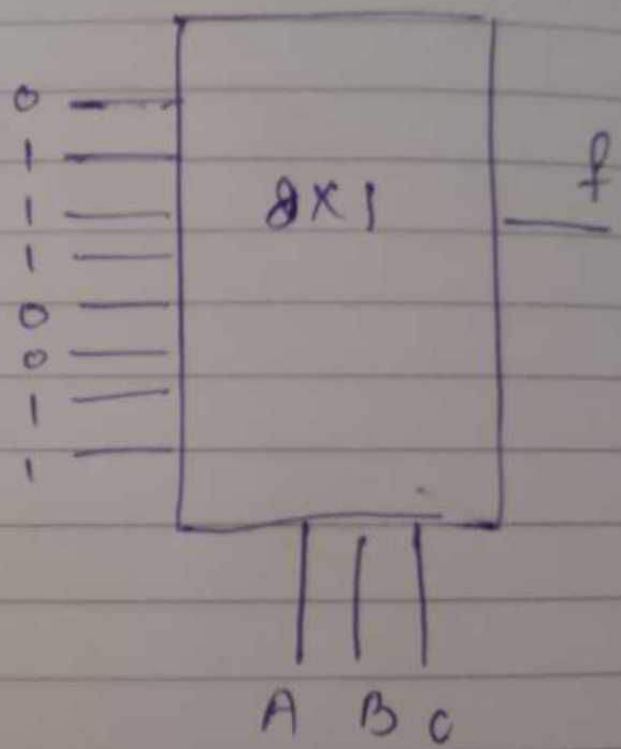
Q. Implement $f = \Sigma m(1,2,3,6,7)$ using multiplexer

If variable n then 2^n minterms

$n = 3$, $2^3 = 8 \rightarrow 0-7$

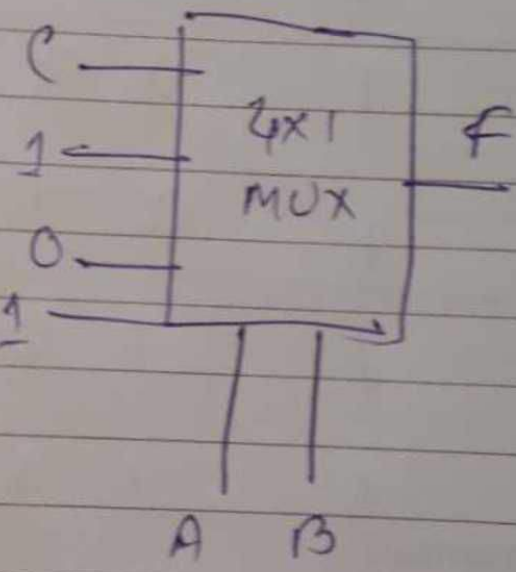
8X1 MUX

A	B	C	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



4x1 MUX

A	B	C	f	
0	0	0	0	C
0	0	1	1	
0	1	0	1	1
0	1	1	1	
1	0	0	0	0
1	0	1	0	
1	1	0	1	1
1	1	1	1	



Q1

A	B	C	f	
0	0	0	0	$F = C$
0	0	1	1	
0	1	0	0	$F = C$
0	1	1	1	
1	0	0	0	$F = C$
1	0	1	1	
1	1	0	0	$F = \bar{C}$
1	1	1	0