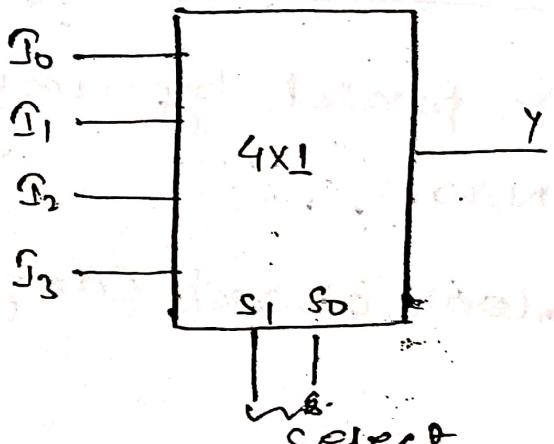
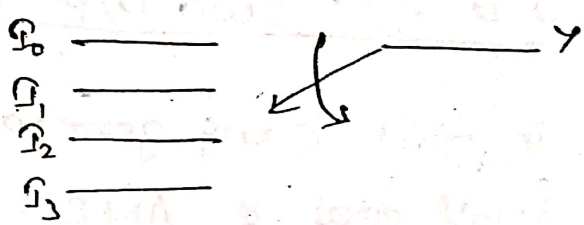
Multiplexes

Multiplexes is a combinational ckt which has many data i/p and one o/p line.



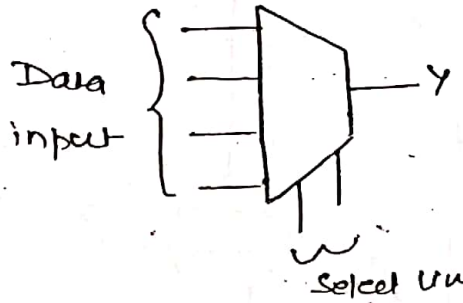


*) Mux is known as many to one ckt.

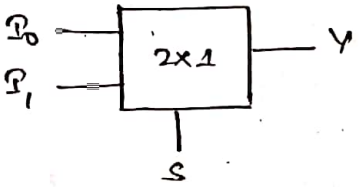
*) Data selector

*) Universal logic ckt

*) Parallel to Serial data converter



2x1 Mux

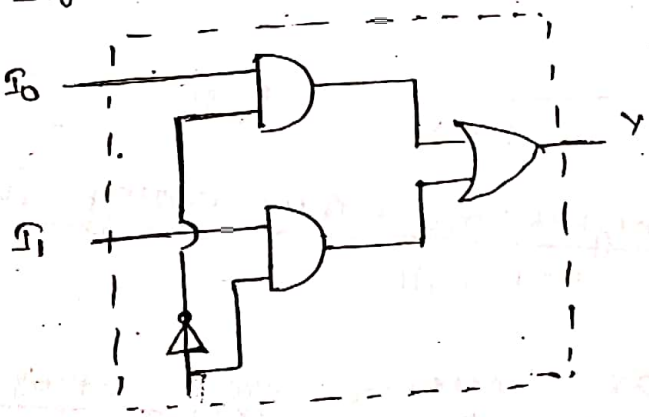


Truth table

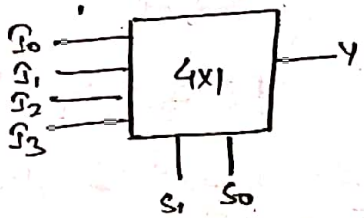
S	Y
0	I_0
1	I_1

$Y = \bar{S} I_0 + S I_1$

Logic ckt

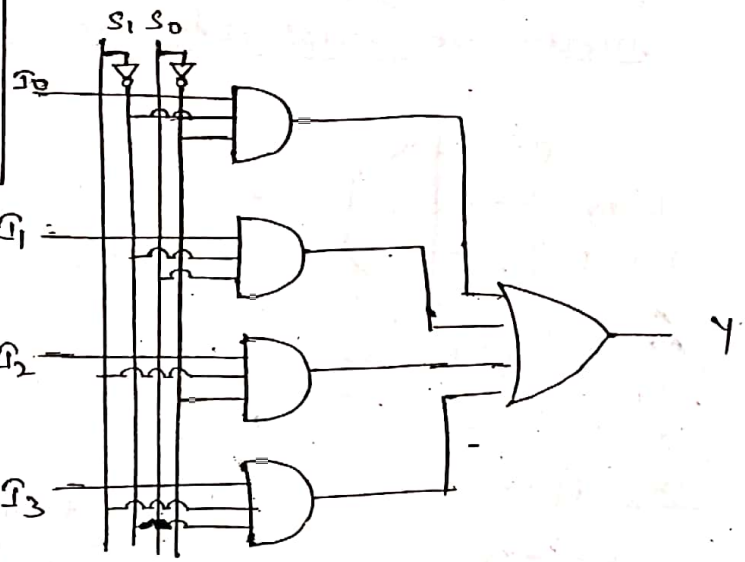


4x1 Mux



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

$$Y = \bar{S_1}\bar{S_0}I_0 + \bar{S_1}S_0I_1 + S_1\bar{S_0}I_2 + S_1S_0I_3$$



Note

Mux is two level AND-OR gate.

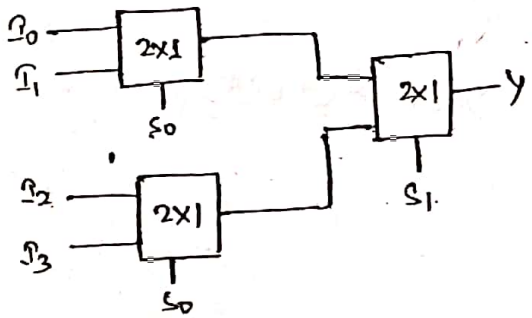
Ex. for 8×1 mux

$$y = \bar{S_2}\bar{S_1}\bar{S_0}I_0 + \bar{S_2}\bar{S_1}S_0I_1$$

1) Implementation of higher order mux using lower order mux

4×1 using 2×1 mux

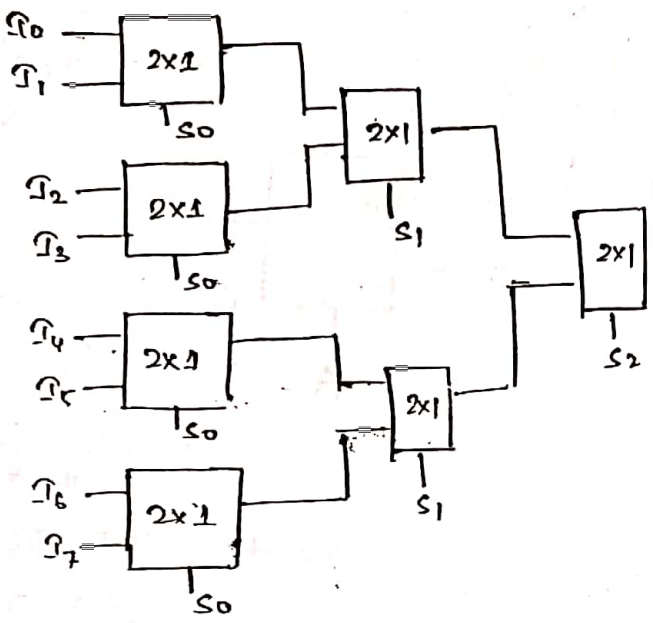
4×1 | S_1 S_0 → 2+1=3 → 2×1



$$y = \bar{S_0}\bar{S_1}I_0 + S_0S_1I_4 + S_0\bar{S_1}I_2 + S_0S_1I_3$$

$$Y = \bar{S_1}\bar{S_0}I_0 + \bar{S_1}S_0I_4 + S_1\bar{S_0}I_2 + S_1S_0I_3$$

11y $8\times1 \xrightarrow{4+2+1} 2\times1$



$y = \bar{S}_2\bar{S}_1\bar{S}_0 I_0 +$

$\vdots$

$S_2 S_1 S_0 I_7$

$8+4+2+1$

b. So

$16\times1 \xrightarrow[8+4+2+1]{15} 2\times1$

$64\times1 \xrightarrow[32+16+8+4+2+1]{63} 2\times1$

$256\times1 \xrightarrow{255} 2\times1$

$2^n\times1 \xrightarrow{2^n-1} 2\times1$

Now

Imp.

$16\times1 \xrightarrow[4+1]{} 4\times1$

$64\times1 \xrightarrow[16+4+1]{} 4\times1$

$64\times1 \xrightarrow[8+1]{9} 8\times1$

$256\times1 \xrightarrow[16+1]{17} 16\times1$

$(1+x+x^2+\cdots x^{n-1})$

$y = x^n$ = big one

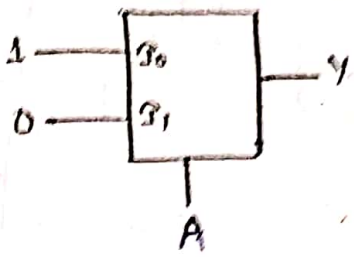
x = small one)

Mux as a Universal logic circuit

Select line → variable
$I_0, I_1, I_2, \ldots$ → variable, I/P, complement form

For NOT gate

0 → 1
1 → 0



'1' → 'Vdd
'0' → gnd

Q: A 2:1 mux having one I/P as 1 & other to get NOT as ? gate.

So for NOT gate only one 2×1 mux is required.

Alternate

For 2×1 mux

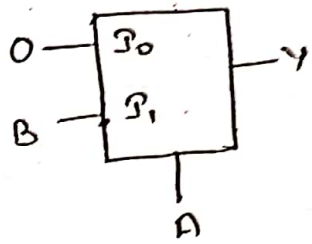
$Y = \bar{S} I_0 + S I_1$

$= \bar{A} \cdot 1 + A \cdot 0$

$= \bar{A}$

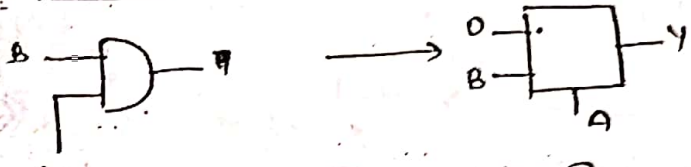
FOR AND gate

A	Y
0	0
1	B



$Y = A \cdot B$

Alternate



If A = 0 then O/P = 0
If A = 1 then O/P = B.

for AND gate only one

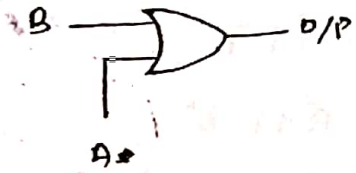
OR

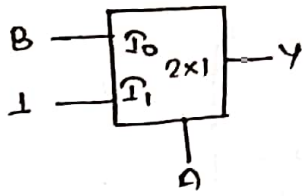
$Y = \bar{S} I_0 + S I_1$

$= A \cdot 0 + A \cdot B$

$= AB$

For OR gate





A	B
0	B
1	1

if A = 0 then O/P is B

A = 1 then O/P is 1

$Y = \bar{S} I_0 + S I_1$

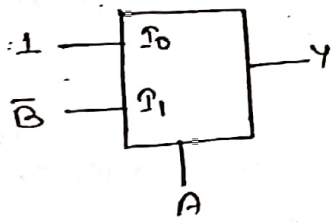
$= \bar{A} B + A \cdot 1$

$= A + B$

p-14

Q 28 -

For NAND gate



$Y = \bar{S} I_0 + S I_1$

$= \bar{A} 1 + A \bar{B}$

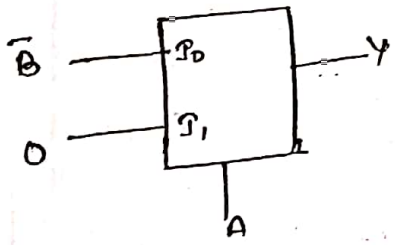
$= \overline{AB} (\bar{A} + A)(\bar{A} + \bar{B})$

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

$= \overline{AB}$

So NAND require 2 – 2×1 MUX (1 for $\bar{B}$)

For NOR gate



$Y = \bar{S} \cdot I_0 + S I_1$

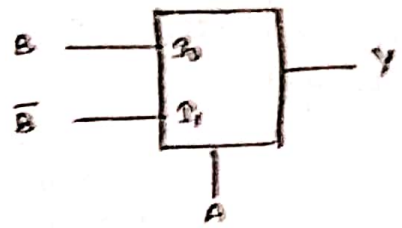
$= \bar{A} \bar{B} + A \cdot 0$

$= \overline{A + B}$

NOR requires – 2 – 2×1 mux (1 for $\bar{B}$)

EX-OR

2 mux required if variable is not available in complement form

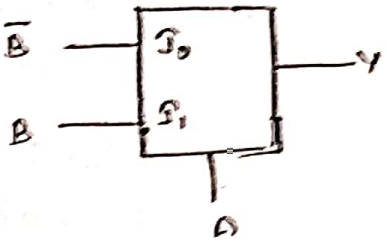


$y = \bar{A} I_0 + A I_1$

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

so 2 two 2×1 mux is require (1 for $\bar{B}$)

EXNOR



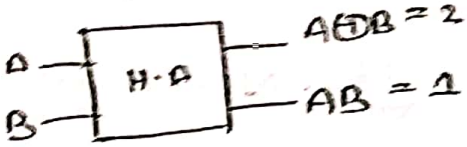
$y = \bar{A} I_0 + A I_1$

$= \bar{A}\bar{B} + AB$ (1 for $\bar{B}$)

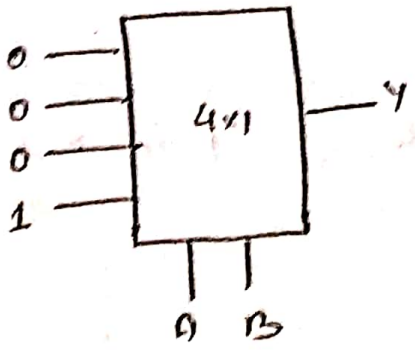
for EXNOR — two 2×1 mux is required

Half adder → 3 → 2×1 (1 2:1 EXOR 2 1 OR)

Half subtractor → 3 → 2×1 (2×1, 1×1)

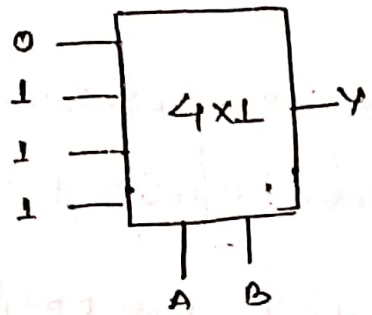


AND using 4-MUX

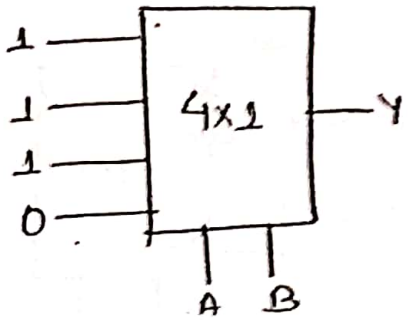


A	B	y
0	0	0
0	1	0
1	0	0
1	1	1

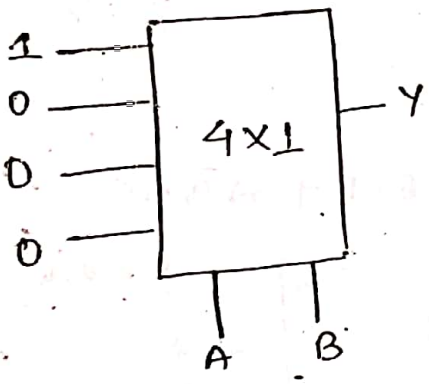
· OR gate using 4x1



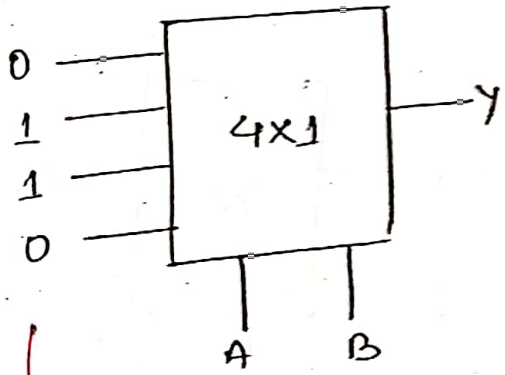
NAND gate using 4x1



· NOR gate using 4x1

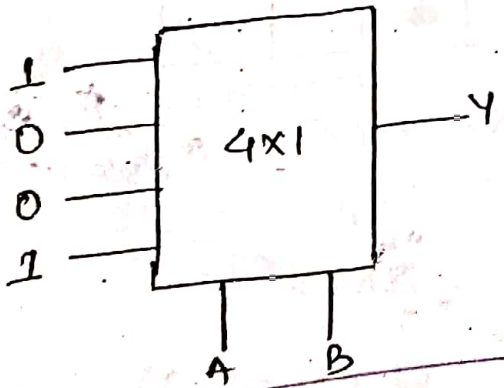


· EX OR



this is the truth table o/p

EX NOR

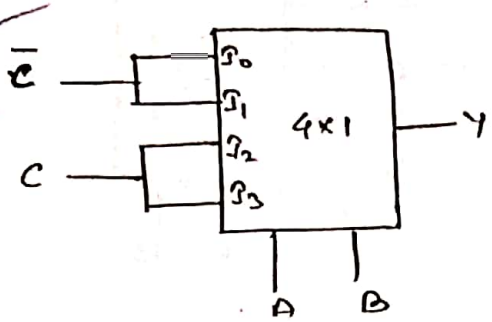


For HA ~~(2)~~ (2) → 4x1

HS (2) → 4x1

Minimized logic expression of MUX:

Q.



$= \bar{S}_1\bar{S}_0 I_0 + \bar{S}_1 S_0 I_1 + S_1\bar{S}_0 I_2 + S_1 S_0 I_3$

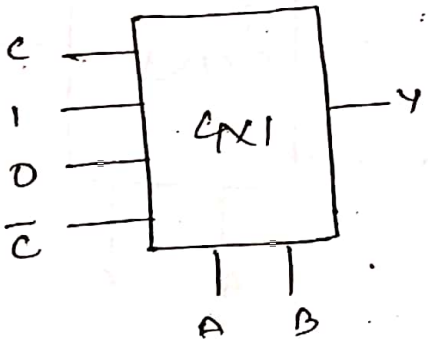
$= \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}C + ABC$

$= \bar{A}\bar{C}[B+\bar{B}] + AC[B+\bar{B}]$

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

$= A \odot C$

Q



$= \bar{A}\bar{B}C + \bar{A}B\cdot 1 + A\bar{B}\cdot 0 + AB\bar{C}$

$=$ [illegible]

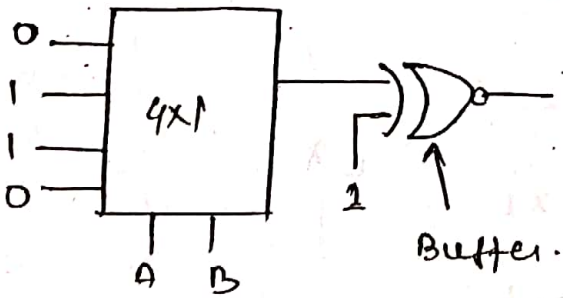
$=$

A \ BC	$\bar{B}\bar{C}$	$\bar{B}C$	$B\bar{C}$	BC
$\bar{A}$		1	1	1
A		[illegible]		1

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

Q.11

Q2



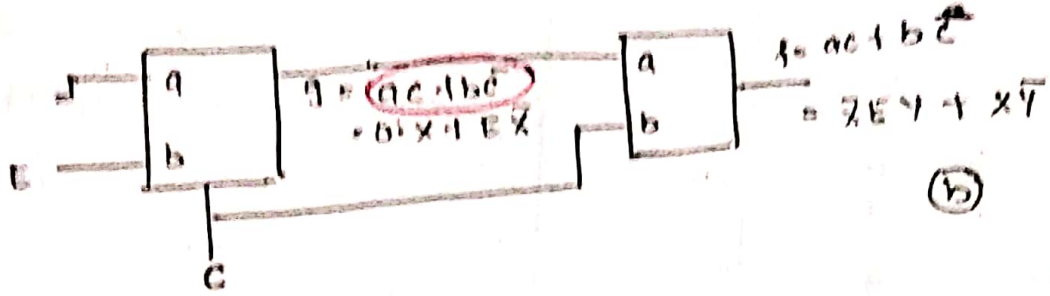
$A \oplus B$ $\Sigma m(1, 2)$

[illegible] $\pi M(0, 3)$

Q10

$f = q\bar{y} + by$

$= (ac + b\bar{c})\bar{y} + \bar{x}c$



(b)

Q13

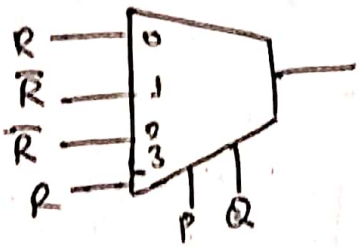
$f(A, B, C) = \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{C}$

$= \bar{A}\bar{B}[C + \bar{C}]$

$= \overline{A + B}$

= NOR → 2 → 2x2

Q15

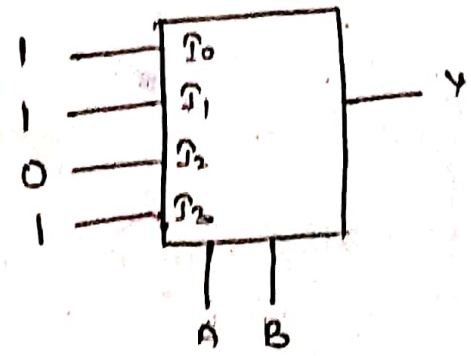


$f = \bar{P}\bar{Q}R + \bar{P}Q\bar{R} + P\bar{Q}\bar{R} + PQR$

(0,0) (1) (1,0) (1,1)

IV Implementation of given logical expression.

Q1. $f(A, B) = \sum m(0, 1, 3)$ using 4x1 mux.

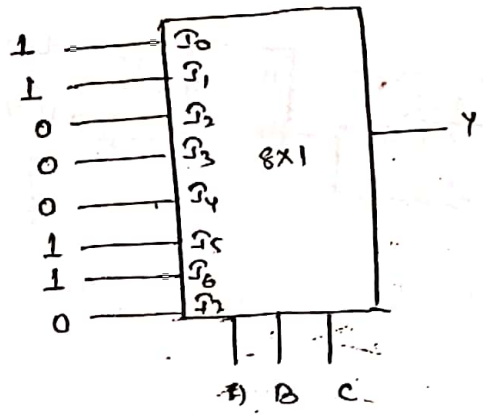


minterm

$\sum m()$

= 1 (remember)

Q Implement $f(A, B, C) = \Sigma m(0, 1, 5, 6)$ using 8×1



→ so any 3 variable can be imple'

Imp { So any n-variable → one $2^n \times 1$ mux can be implemented.

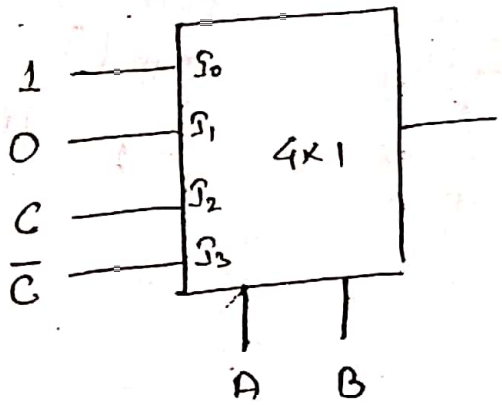
Q Implement $f(A, B, C) = \Sigma m(0, 1, 5, 6)$ using 4×1 mux with AB as select line.

(MSB = A, LSB = C)

Implementation table

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

	A	B	C	
0	0	0	0	$\bar{C}$
1	0	0	1	C
2	0	1	0	$\bar{C}$
3	0	1	1	C
4	1	0	0	$\bar{C}$
5	1	0	1	C
6	1	1	0	$\bar{C}$
7	1	1	1	C



3-variable f^n ——— One 4×1 mux and 1 NOT gate

→ one 8×1 mux

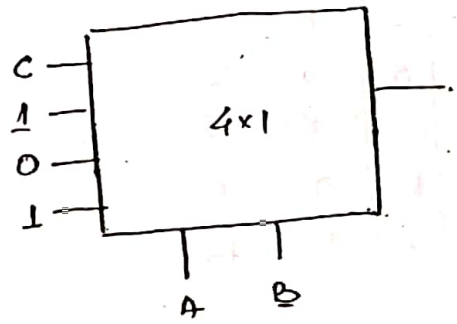
So

n-variable →	$2^n \times 1$ mux
	$2^{n-1} \times 1$ & one NOT gate

Q Implement $f(A, B, C) = \sum m(1, 2, 3, 6, 7)$ using 4×1 mux with 1) AB as select
2) AC as select
3) BC as select.

AB as select

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

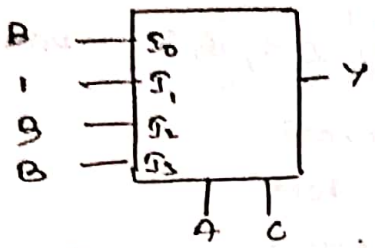


So using one 4×1 mux → any two variable
→ some of three variables

using one 4×1 mux & one NOT gate → any 2 any 3 variables.

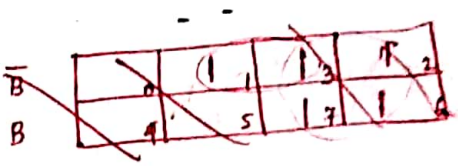
using one 8×1 mux < any 3 / some 1 of 4 i/p variables

ii) AC select line



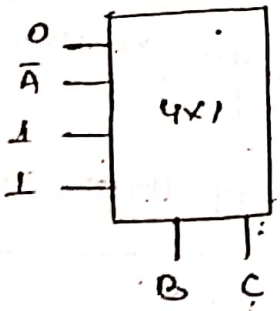
	I_0	I_1	I_2	I_3
$\bar{B}$	0	(1)	4	5
B	(2)	(3)	(6)	(7)
	B	1	B	B

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



got it... :)

iii) BC as select line



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

Q $f(A,B,C) = \bar{A} + BC$ using 4×1 mux

$f(A,B,C) = \pi M(1,2,3,7)$ using 4×1 mux

F.A using 8×1 mux (s) → s means not single mux it's mux.

F.A using 4×1 mux (s)

11

Q3 W.B

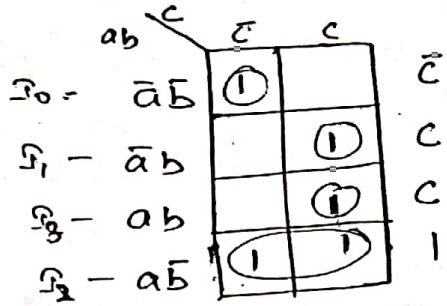
	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
$\bar{B}$	0	2	4	6	(8)	(10)	12	14
D	1	(3)	5	7	(9)	11	13	15
	0	D	0	0	1	$\bar{D}$	0	0

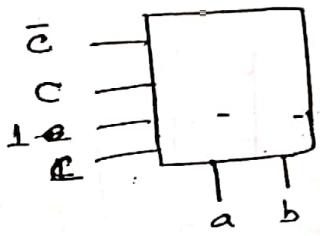
$f = (A, B, C, D) = \Sigma m(3, 8, 9, 10)$

Q5 solve urself

Q8 $F(a, b, c) = \bar{b}\bar{c} + bc + a\bar{b}$

Imp.





↗ By using K-MAP

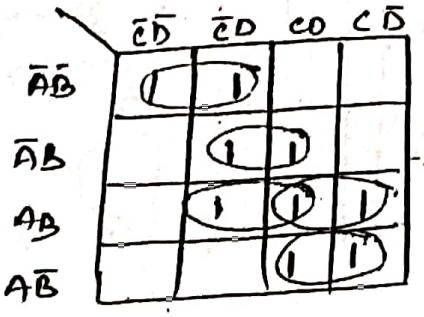
or you do same problem using implementation.

Q12

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

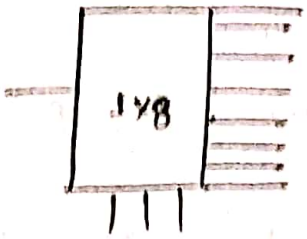
Q 4-variable using 4×1 mux using k-map

$f(A, B, C, D)$



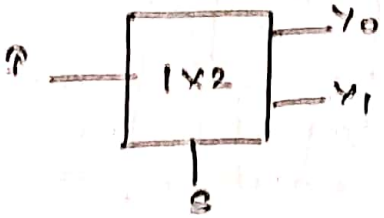
$\bar{C}D + C\bar{D} + CD = C + D$ ✓

DeMux De Mux

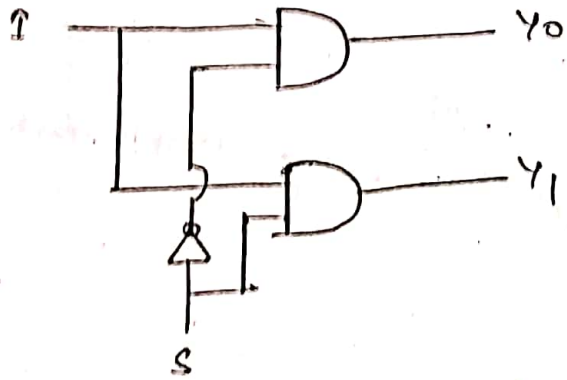


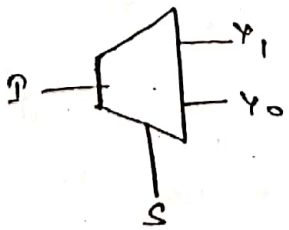
Demux is have 1 i/p & many o/p depending upon the select or control lins Hence it is one to many ckt. or Data distributor

1x2 Demux



S	Y_1	Y_0
0	0	(I)
1	(I)	0

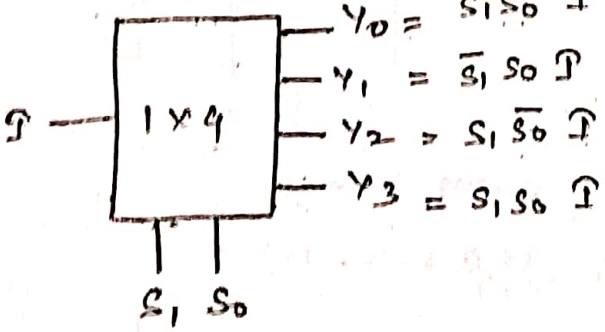




$Y_0 = \bar{S} \cdot I$

$Y_1 = S I$

1x4 Demux

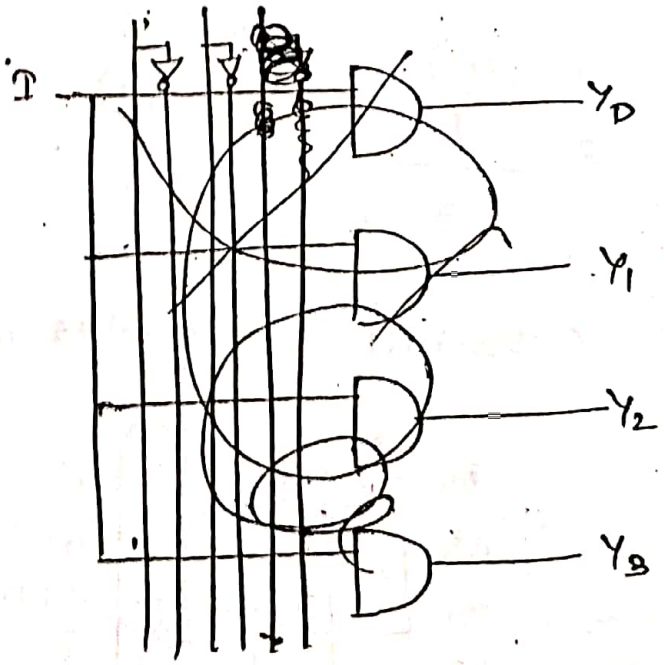


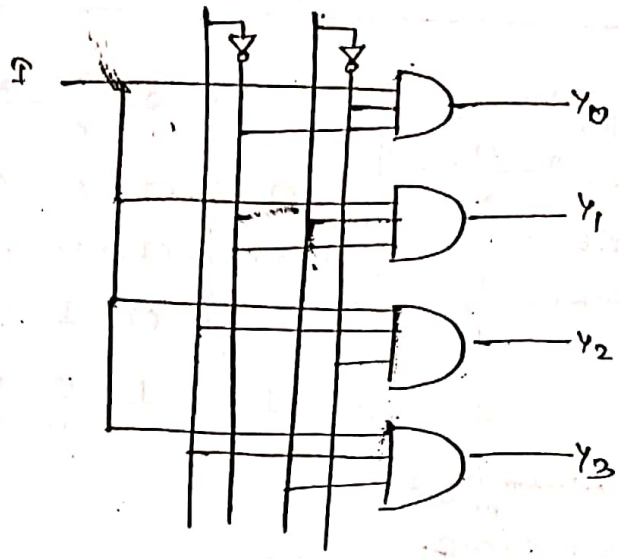
$Y_0 = \bar{S_1}\bar{S_0} I$

$Y_1 = \bar{S_1} S_0 I$

$Y_2 = S_1 \bar{S_0} I$

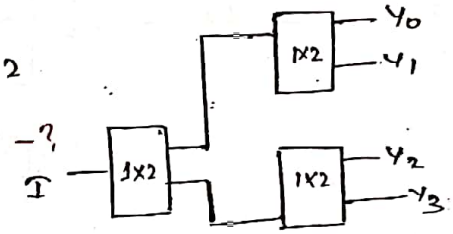
$Y_3 = S_1 S_0 I$





Highes order Demux — lowes order.

1 x 4	1 + 2 = 3	1x2
4 x 1	2+1 = 3	2x1 -?
1 x 8	(7)	1x2
8x1	7	2x1 -?
1x16	15	1x2
1x16	5	1x4
1x64	21	1x4
1x64	9	1x8
1x256	17	1x16



what is 4x1 demux ??

Decodes:

*) Decodes is many to many ckts which is used to decode binary to other codes such as

- binary to octal (3 x 8)
- BCD to decimal (4x10)
- binary to hex (4 x 16)