xoR → pure addition

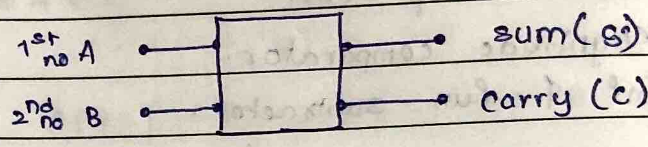
PAGE No.	
DATE	/ /

① check T.T
② identify gates

Half adder (XOR = sum , AND = carry)

→ used to add single bit no.
→ Does not take carry from previous sum.

halfadder →



A	+ B	sum (XOR)	carry (AND)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

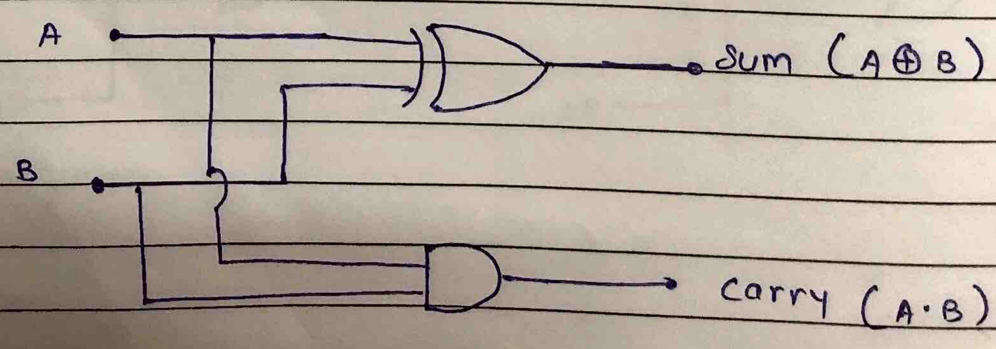
* As we see, truthtable of sum, it is an XOR operator T.T. [0, 1, 1, 0].

∴ $\boxed{\text{sum} = A \oplus B}$.

* As we see carry truthtable, it is a AND operator T.T [0, 0, 0, 1].

$\boxed{\text{Carray} = A \cdot B}$

⇒



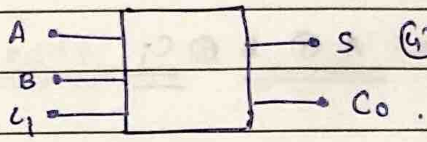
# In kmap when nothing gets grouped, output is
output = xor of all inputs

PAGE No.
DATE / /

Full Adder   sum = (xor of all inputs)

#. ① TT
② Kmap
③ Kmap – sum $A \oplus B \oplus C_i$
④ carry $= AB + (A \oplus B)C_i$

→ 3 inputs, A, B, C_i
2 output, S, C_o



A +	B +	C_i	sum	carry o.
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

$\boxed{sum = A \oplus B \oplus C_i}$

$\boxed{carry = AB + C_i(A \oplus B)}$

Make 3 variable kmap.

* For sum (insert values of sum = 1)

A \ B C_i	00	01	11	10
00	0	1	0	1
01	1	0	1	0

* Nothing can be grouped.
→ $\boxed{S = A \oplus B \oplus C_i}$
output = XOR of all inputs.

* For carry output. (insert values of carry o = 1).

A \ B C_i	00	01	11	10
0			1	
1		1	1	1

→ $AC_i + BC_i + AB = C_o$.
$C_o = AB + C_i(A \oplus B)$

Do not combine the pairs

	00	01	11	10
0			1	
1		1	1	1

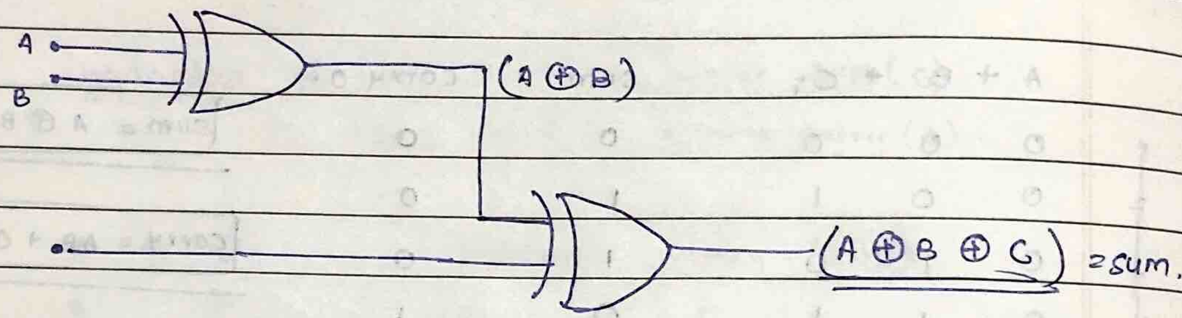
$= AB + A\bar{B}C_i + \bar{A}BC_i$
$= AB + C_i(A\bar{B} + \bar{A}B) = AB + C_i(A \oplus B)$

To draw circuits, get eqn from K maps & then step by step go on drawing.

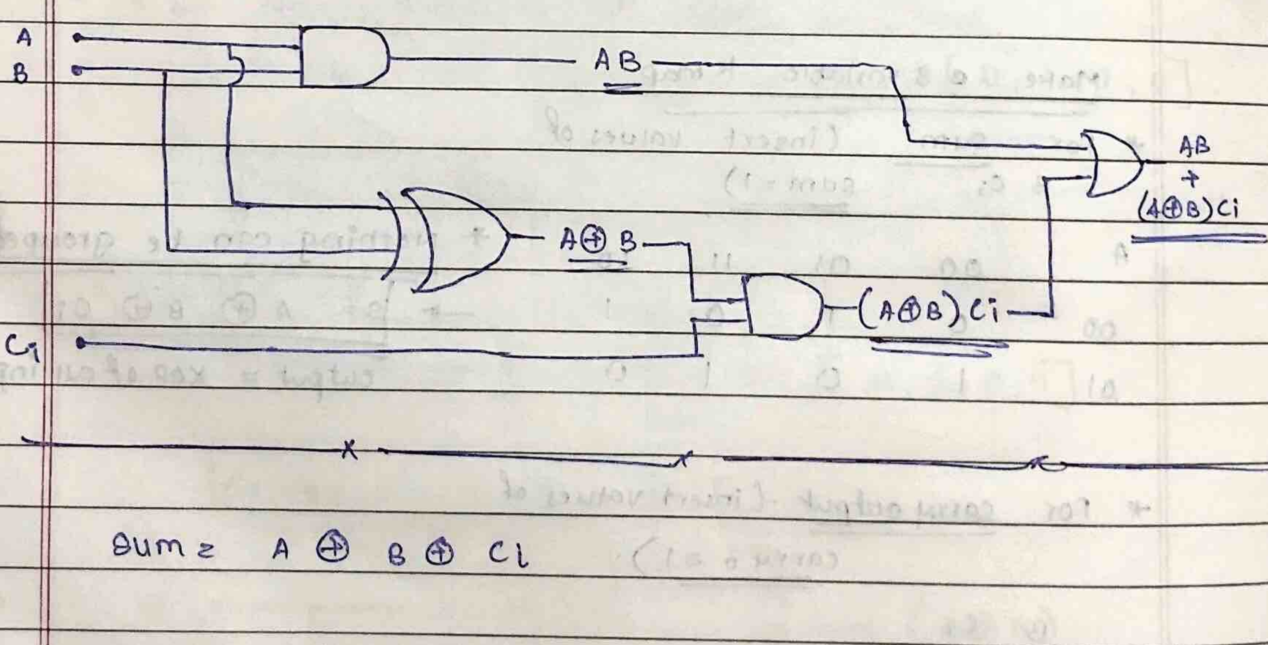
PAGE No.	
DATE	/ /

Circuit

sum = $A \oplus B \oplus C_i$



Carry = $(A \oplus B)\,C_i + AB$.



---

sum = $A \oplus B \oplus C_i$

carry O = $(A \oplus B)C_i + AB \longrightarrow \bar{A}B\,C_{in} + A\bar{B}\,C_{in} + AB\overline{C_{in}} + AB\,C_{in}$

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

$\rightarrow 1$

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

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

Fulladder using Half adder

Halfadder

$S = A \oplus B$.

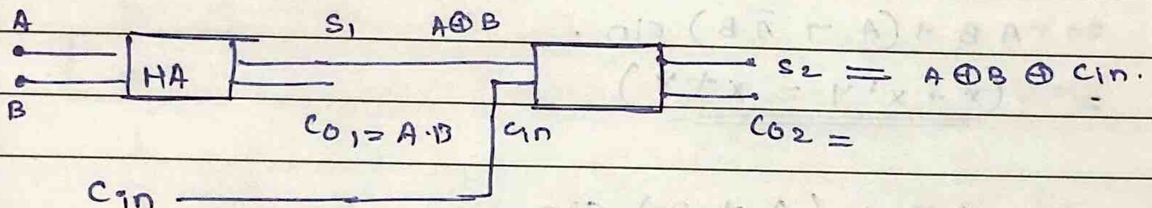
$C = A \cdot B$

Fulladder.

$S = A \oplus B \oplus C_i$

$C_o = (A \oplus B) \cdot C_i + AB$.

* We need 2 halfadder for 1 Fulladder.



$S_1 = A \oplus B$

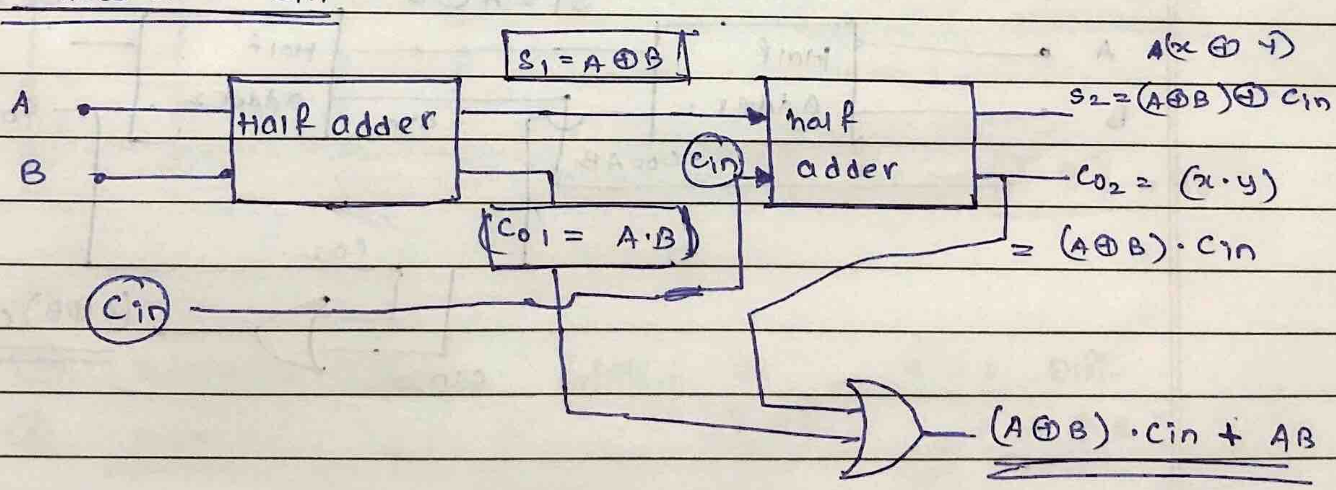
$C_{o1} = A \cdot B$

Cin extra

$S_2 = A \oplus B \oplus C_{in}$

$C_{o2} = (A \oplus B)\, C_{in} + AB$.

we need extra Cin



#

1. take 2 half adders.
2. sum of h_1 → sum (h_2 (input)) $= A \oplus B$
   carry of h_1 → stays out.
3. Cin → (h_2 input) (extra)

✓output = sum = $A \oplus B \oplus Cin$.
= carry = $(A \oplus B) \cdot Cin$

4. OR gate. towards

$h_1[C]$ & $h_2[C]$

$= A \cdot B$ $= Cin(A \oplus B)$

$= AB + Cin(A \oplus B)$

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

$A \cdot B + C_{in}(A \oplus B)$

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

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

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

$(x + x'y = x + y.)$

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

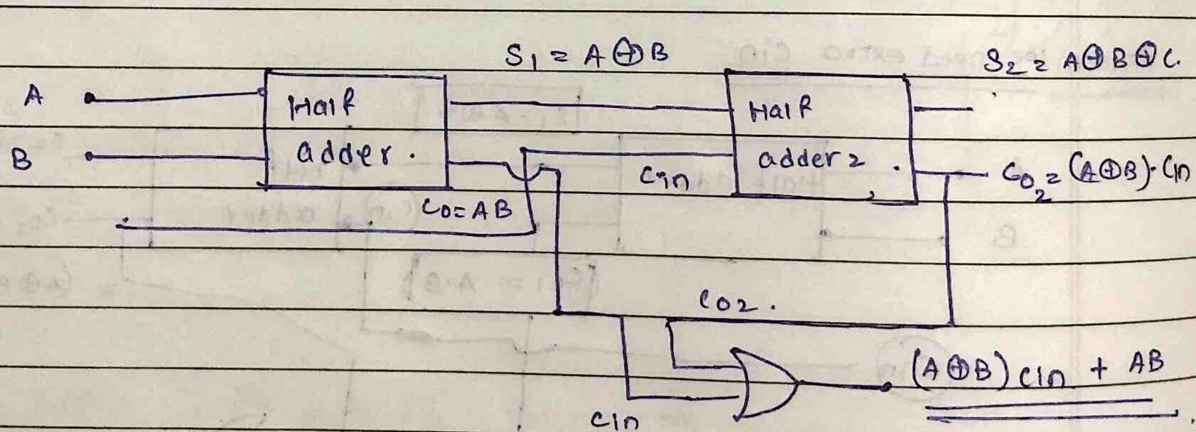
$= AB + AC_{in} + \bar{A}BC_{in}$

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

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

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

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



(XOR = pure addition)

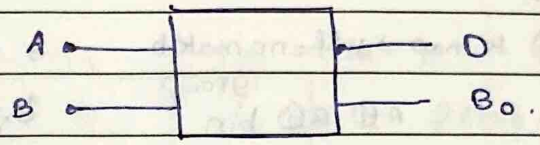
PAGE No.
DATE / /

(AND)

① Truth table
② Identify gates

Half ~~Add~~ Subtractor [D = XOR , $B_0 = \bar{A}B$]

- A & B are single bit.



D = difference

B_0 = burrow output.

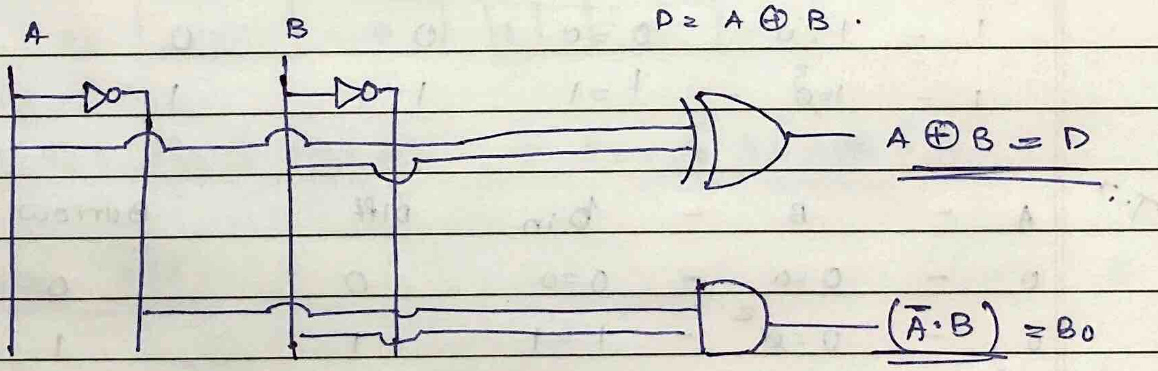
~~Truthtable~~:

A	− B	D (XOR)	B_0.
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Difference (D) = XOR Gate $(A \oplus B)$

$\boxed{B_0 = \bar{A}B}$ (AND)

D = A ⊕ B.



① Truth table.

② Identify Gates. AND = B_0 $(\bar{A} \cdot B)$, XOR = Diff. $(A \oplus B)$.

③ Plot

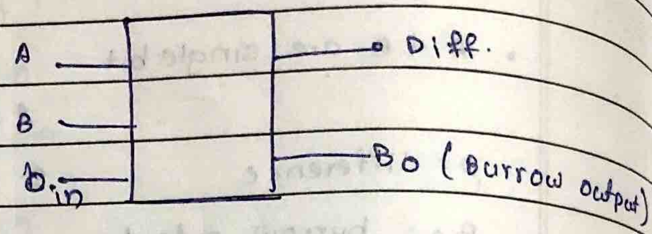
Full subtractor

① T.T

② K map → diff = no match group
$= A \oplus B \oplus b_{in}$

③ K map → B_o

⇒ get ean.



b_{in} = burrow from next stage

A		B		b_{in}	Diff	Bo.
0		0		0	0	0
0	—	0	—	1 = 1	1	1
$\cancel{0}^2$	—	1 = 1	—	0 = 1	1	1
$\cancel{0}$ 2-1		1 = 1		1 = 0	0	1
1	—	0 = 1	—	0 = 1	1	0
1	—	0 = 1	—	1 = 0	0	0
1	—	1 = 0		0 = 0	0	0
1	—	1 = $\cancel{0}^2$	—	1 = 1	1	1

T.T

A		B		b_{in}	Diff	Burrow.
0	—	0 = 0	—	0 = 0	0	0
0	—	0 = $\cancel{0}^2$	—	1 = 1	1	1
$\cancel{0}^2$	—	1 = 1	—	0 = 1	1	1
$\cancel{0}^2$	—	1 = 1	—	1 = 0	0	1
1	—	0 = 1	—	0 = 1	1	0
1	—	0 = 1	—	1 = 0	0	0
1	—	1 = 0	—	0 = 0	0	0
1	—	1 = $\cancel{0}^2$	—	1 = 1	1	1

When no group is formed.

→ XOR ~~is~~ of all input.

PAGE No. DATE / /

$Diff = A \oplus B \oplus B_{in}$ | $B_o = B\,b_{in} + A\,b_{in} + \bar{A}B$

* K map for Diff of 3 variables (Diff = 1 → input values).

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

Same happened in full adder.

No group is formed.

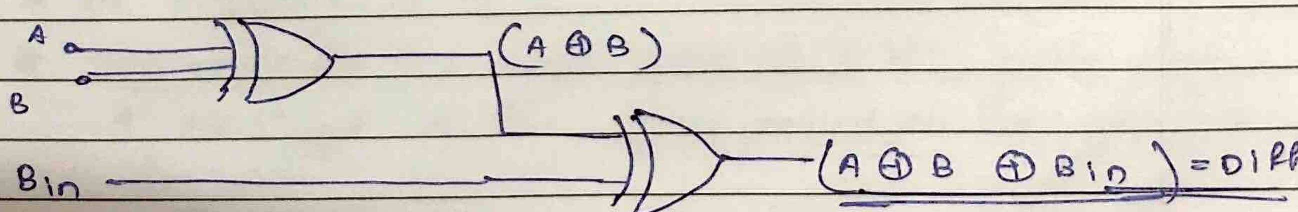
thus $\boxed{D = A \oplus B \oplus b_{in}}$

* K map for B_o of 3 variables (B_o = 1 → input values)

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

$B_o = B\,b_{in} \oplus \bar{A} b_{in} + \bar{A}B \Rightarrow B\,b_{in} + A\,b_{in} + \bar{A}B$.

~~Stem~~ Diff

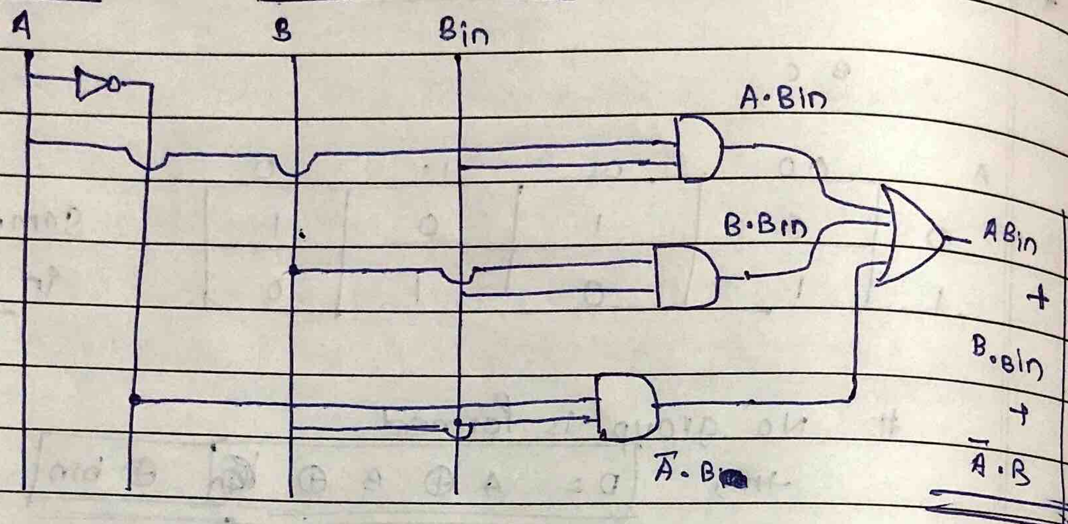


① $(A \oplus B)$

② $(A \oplus B) \oplus B_{in}$

PAGE No.
DATE / /

Burrow → A bin + B bin + $\bar{A}$ B



* similar to fulladder

① Truth table A B Bin | Diff B_r

② K map → diff = $A \oplus B \oplus Bin$ [No group].

K map → get eqn = Abin + B bin + $\bar{A}$ B .

③ Plot

#7 ~~Four bit parallel adder~~

Ripple Carry Adder / Parallel adder

→ Application of full adder, created using full adder.
→ Used to add ~~2 n bit~~ 2, n-bit binary no.

We consider ~~4 full adders~~ 2, 4 bit binary no.

A = 1 1 0 1 = 13

B = 1 0 1 1 = 11 → 2, 4 bit binary numbers.

* we require 4 full adders.

		1	1	1	
∴		1	1	0	1 = 13
+		1	0	1	1 = 11
	1	1	0	0	0 = 24

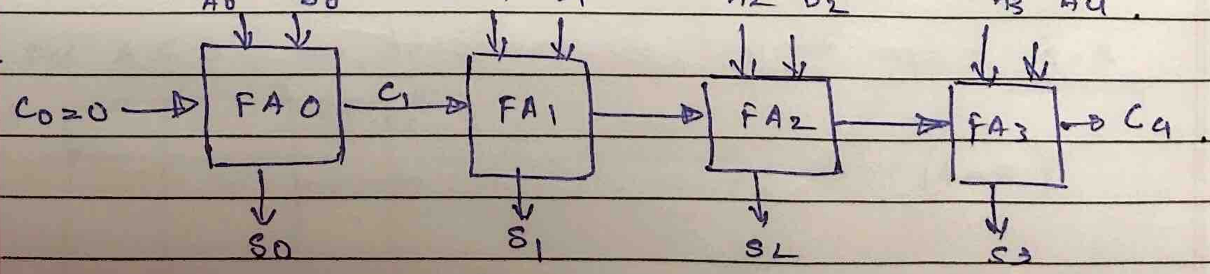
A_0	A_1	A_2	A_3
1	1	0	1
B_0	B_1	B_2	B_3
1	0	1	1

(onebits)

Basically a full adder, adds the two bit, produces a sum & stores it & passes on the carry to next full adder as a input.

at beginning $C_0 = 0$.
two inputs are the single digit of 4 bit binary no(s) & third input is the carry passed on from previous full adder

4 bit adder



2 outputs are ① carry that will be passed to next FA.
② sum that will get stored.

Just add FA, to increase size as much we want.

n bit adder → parallel adder.

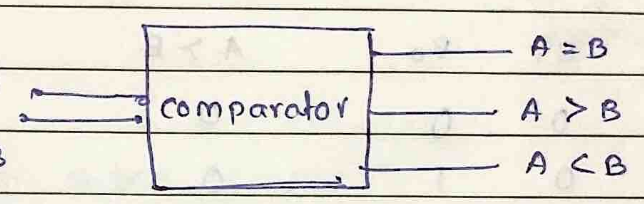
* By connecting 2,4 bit adders. we get a n bit adder or a 8 bit adder.

This is called a parallel adder because, additⁿ of all bits is happenning parallely.

$X\,NOR = (XOR)'$

Magnitude Comparator:

→ Compares 2 no, determines ~~the~~ whether =, <, or >.
→ output is in the form of 3 binary variables,
A = B or A > B or A < B.



* According to input & logic, at a time only one output is high.

* 2 inputs → 1 bit.

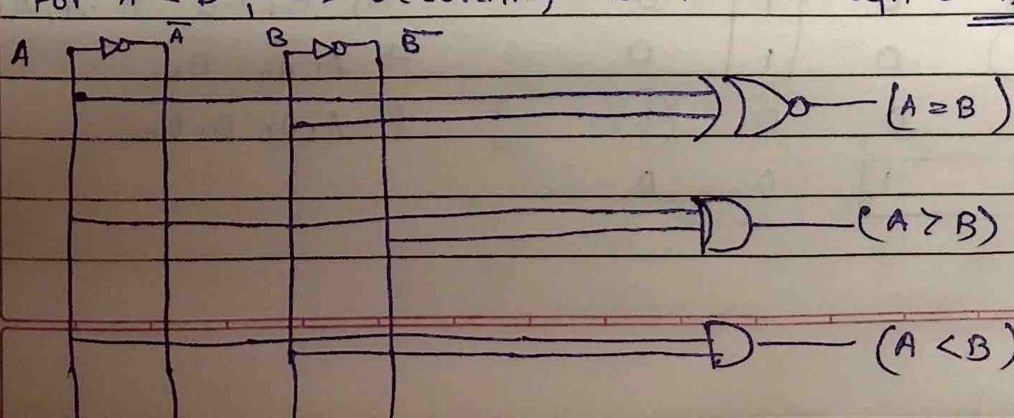
A	B	when, A = B	when, A > B	A < B
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

For A = B, → according to T.T. → X NOR Gate.

For A > B, → according to T.T. → eqn = $A \cdot \bar{B}$

For A < B, → according to T.T. → eqn = $\bar{A} \cdot B$

1 bit comparator



① Truth table for =, >, <.

② K map for =, >, <.   (<, >) AND

→ eqn. → plot.   (=) → XNOR

PAGE No.
DATE / /

For 2 bit comparator: we have 4 bits

i.e. A_1, A_0, B_1, B_0;

A>B & A<B & A=B.

$A = A_1 A_0$ $A_1 A_0 > B_1 B_0$ $A_1 A_0 < B_1 B_0$

$B = B_1 B_0$.   A > B   A < B.

	A_1	A_0	B_1	B_0	A > B	A < B	A = B.
	0	0	0	0	0	0	1
	0	0	0	1	0	1	0
	0	0	1	0	0	1	0
	0	0	1	1	0	1	0
	0	1	0	0	1	0	0
01 = 01	0	1	0	1	0	0	1
01 < 10	0	1	1	0	0	1	0.
	0	1	1	1	0	1	0
	1	0	0	0	1	0	0
10 < 01	1	0	0	1	1	0	0
10 = 10	1	0	1	0	0	0	1
	1	0	1	1	0	1	0
	1	1	0	0	1	0	0
	1	1	0	1	1	0	0
	1	1	1	0	1	0	0
	1	1	1	1	0	0	1

Kmap.   K map → A>B (0100, 1000, 1001, 1100, 1101, 1110)

$A_1 A_0$ \ $B_1 B_0$	00	01	11	10
00	0	0	1	0
01	1	0	1	0
11	0	0	0	1
10	1	1	0	0

$\overline{A_1}\, B_1 B_0 + \overline{A_1} A_0 \overline{B_1}\, \overline{B_0}$

$+ A_1 \overline{A_0}\, \overline{B_0}$

$+ A_1 A_0 B_1 \overline{B_0}$.

2^n

PAGE No.
DATE / /

As same, Kmaps for $A < B$ & $A = B$.

$\therefore$ 1 bit comparator $\rightarrow$ 2 variable $\rightarrow$ 4 rows

2 bit comparator $\rightarrow$ 4 variable $\rightarrow$ 16 rows.

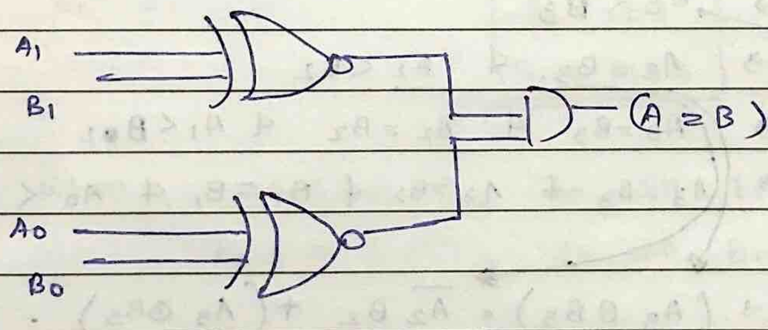
n bit comparator $\rightarrow$ 2n variable $\rightarrow$ $\boxed{2^{2n} \text{ rows}}$.

for (A=B) in 2 bit.

$= (A_1 = B_1)$ & $(A_0 = B_0)$

$= (A_1 \text{ xnor } B_1)$ and $(A_0 \text{ xNor } B_0)$.

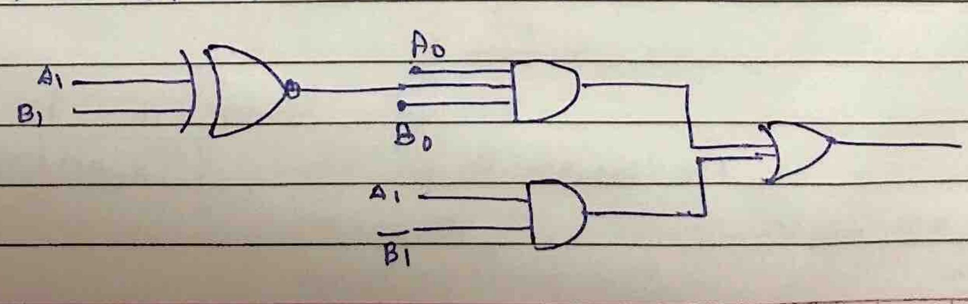
$= (A_1 \odot B_1) \cdot (A_0 \odot B_0)$.



for (A>B) $\rightarrow$ $A_1 > B_1$ (start comparing from MSB)

if $A_1 = B_1$ + $A_0 > B_0$.

$(A>B) = (A_1 \overline{B_1}) + (A_1 \odot B_1)(A_0 \overline{B_0})$.

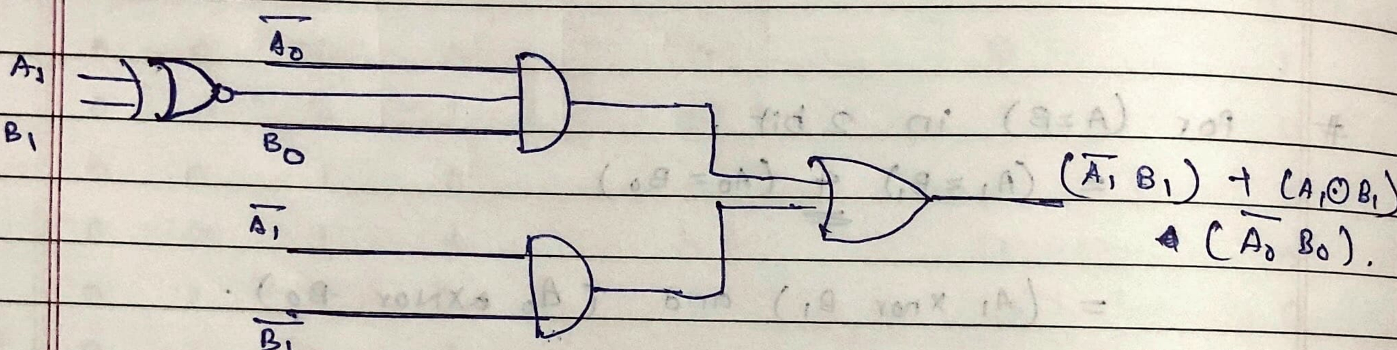


For $A < B$: , start comparing from MSB -

→ $A_1 < B_1$

(or) $A_1 = B_1$ & $A_0 < B_0$

$$A<B \rightarrow (\overline{A_1}\, B_1) + (A_1 \odot B_1)(\overline{A_0}\, B_0)$$



For 4 bit

$A<B$ → $A_3 < B_3$

→ $A_3 = B_3$ & $A_2 < B_2$

→ $A_3 = B_3$ & $A_2 = B_2$ & $A_1 < B_1$.

→ $A_3 = B_3$ & $A_2 = B_2$ & $A_1 = B_1$ & $A_0 < B_0$.

$$(A<B) = \overline{A_3}\, B_3 + (A_3 \odot B_3)\cdot \overline{A_2}\, B_2 + (A_3 \odot B_3)\cdot ---$$

① Truth-table of ⑤

② write eqn of y including I & S

③ plot eqn

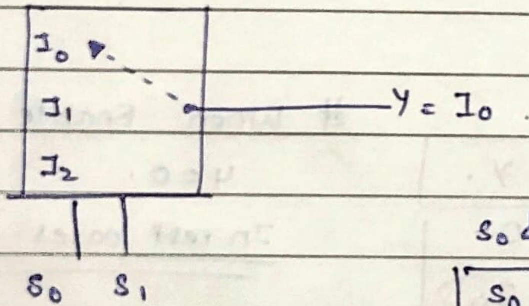
Multiplexer : * always 1 output . (Many to one device)

E, S, I are inputs

- Combinational circuit → that ~~takes~~ selects binary inforamation from one of many input lines and directs it to ~~one~~ output line.

eqn for y includes I

- Simply a data selector .



Advantages

① Reduces wires.

② Reduces complexity

③ Implement various circuits.

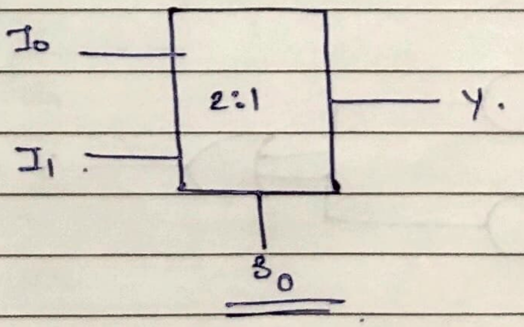
S_0 & S_1 = selector line.

$\boxed{S_0 = LSB \quad S_1 = MSB}$

when , S_0 & S_1 = 00 , I_0 will be selected

S_0 & S_1 = 11 , I_3 will be selected.

Representation



If n = no. of inputes.

$\boxed{n = 2^m}$, m = no. of selectlines.

$m = \log_2 n$

$\therefore n = 4$, $m = \log_2 2^2$
$= 2$

E	S	Y
0	x	0.
1	0 →	I_0
1	1 →	I_1

y me I ata hai
s ka logic banta hai
S_1 + S_0 LHS (input) / Y output

TT

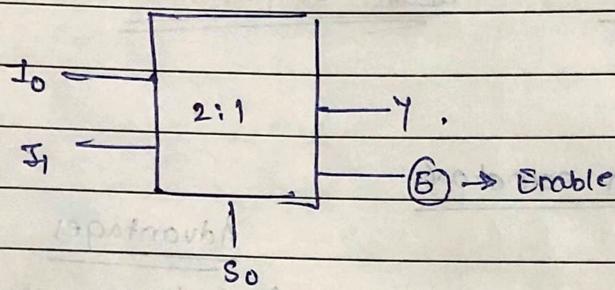
2:1, 4:1, 8:1, 16:1 & 32:1 Mux.

1 2 3 4 5

selector lines.

(TT→S) ① Truth table (ratta).
② Eqn acc. to tt of y
③ Plot eqn.

2:1 Mux



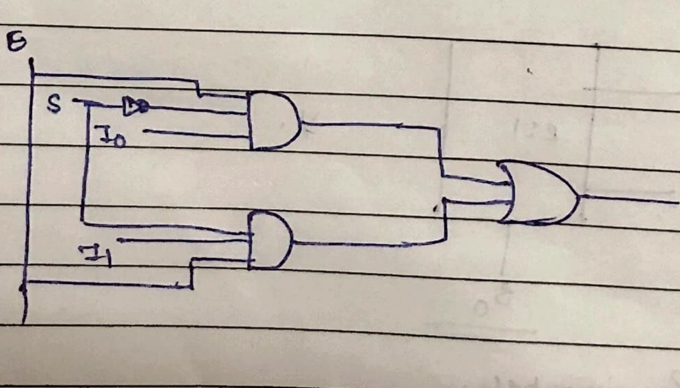
Truth table.

E	S	Y.
0	x	0
1	0	I_0.
1	1	I_1.

# When Enable = 0,
y = 0.
In rest cases E = 1

① TT of S
② Eqn of y including I & S

$$Y = E \cdot \bar{S} I_0 + E \cdot S \cdot I_1$$
$$= E(\bar{S} \cdot I_0 + S \cdot I_{01})$$



$Y = (E \cdot \bar{S} I_0 + ES \cdot I_1)$

$n = 2^m$

n = no. of inputs

m = no. of selectlines.

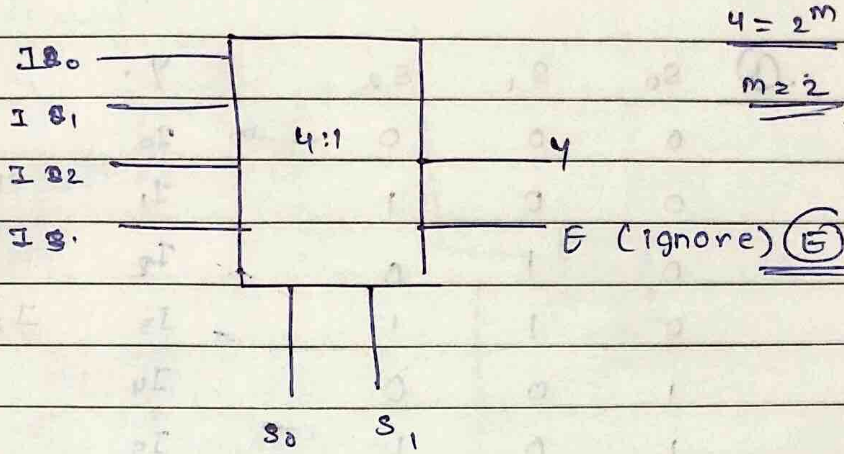
PAGE No. | DATE / /

4:1 Multiplexer

4 inputs 1 output, 2 selector lines. $n = 2^m$

$4 = 2^m$

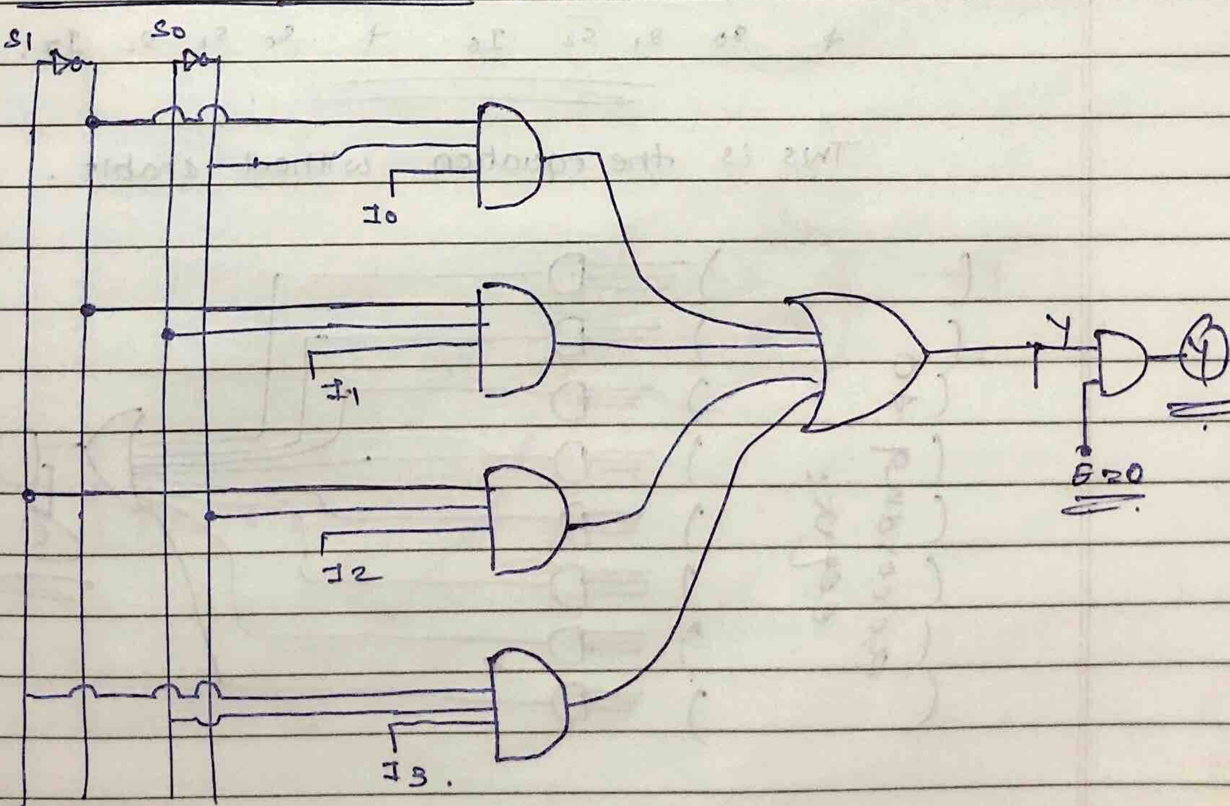
$m = 2$



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

$Y = \overline{S_1}\,\overline{S_0}\,I_0 + \overline{S_1}\,S_0\,I_1 + S_1\,\overline{S_0}\,I_3 + S_1\,S_0\,I_4$.

plot:



① Truthtable of s
② Eqn of y including I & S
③ Plot eqn

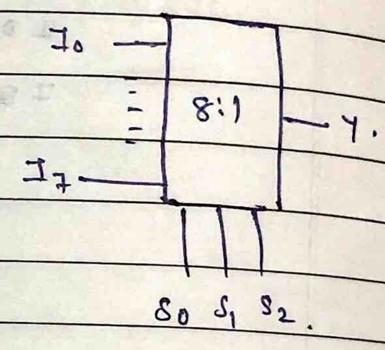
PAGE No.
DATE / /

8:1 Multiplexer → 3 selector lines $2^m = n$

I_0 --- I_7 & 3 selector lines & 1 y Output.

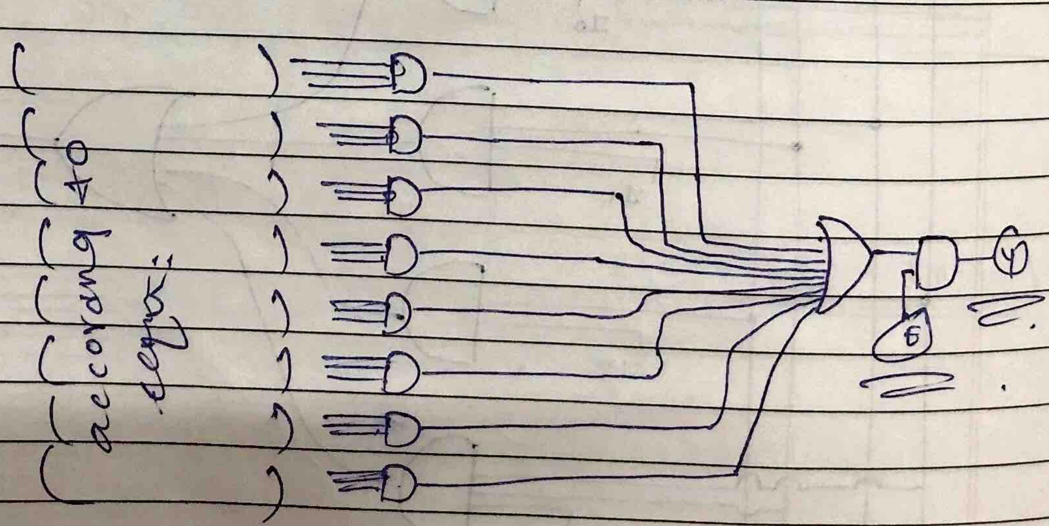
∴ ①

S_0	S_1	S_2	Y
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

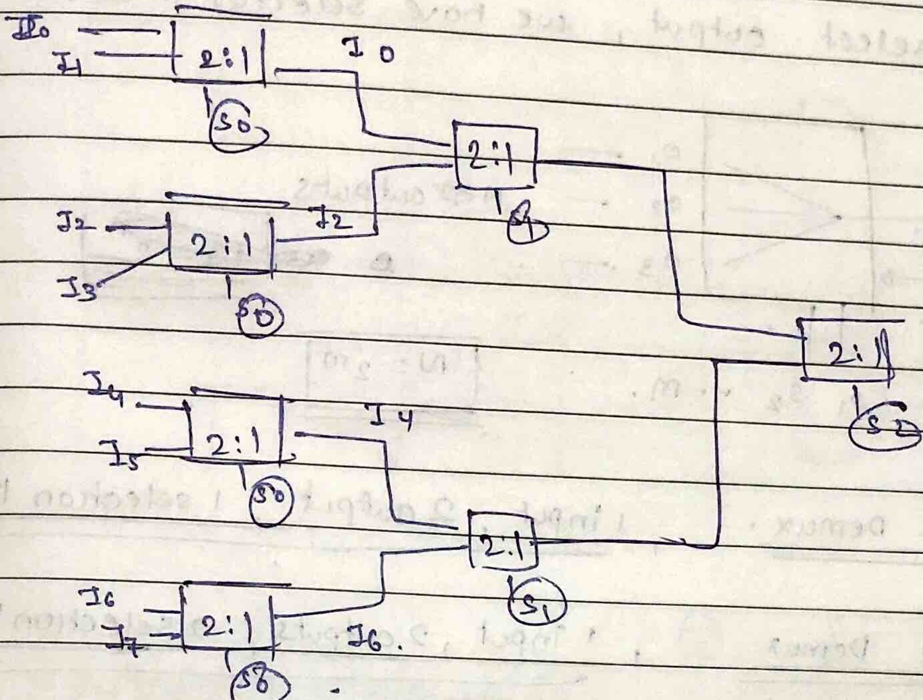


② Eqn ⇒
$$Y = \bar{S_0}\,\bar{S_1}\,\bar{S_2}\,I_0 + \bar{S_0}\,\bar{S_1}\,S_2\,I_1 + \bar{S_0}\,S_1\,\bar{S_2}\,I_2 + \bar{S_0}\,S_1\,S_2\,I_3 + S_0\,\bar{S_1}\,\bar{S_2}\,I_4 + S_0\,\bar{S_1}\,S_2\,I_5 + S_0\,S_1\,\bar{S_2}\,I_6 + S_0\,S_1\,S_2\,I_7.$$

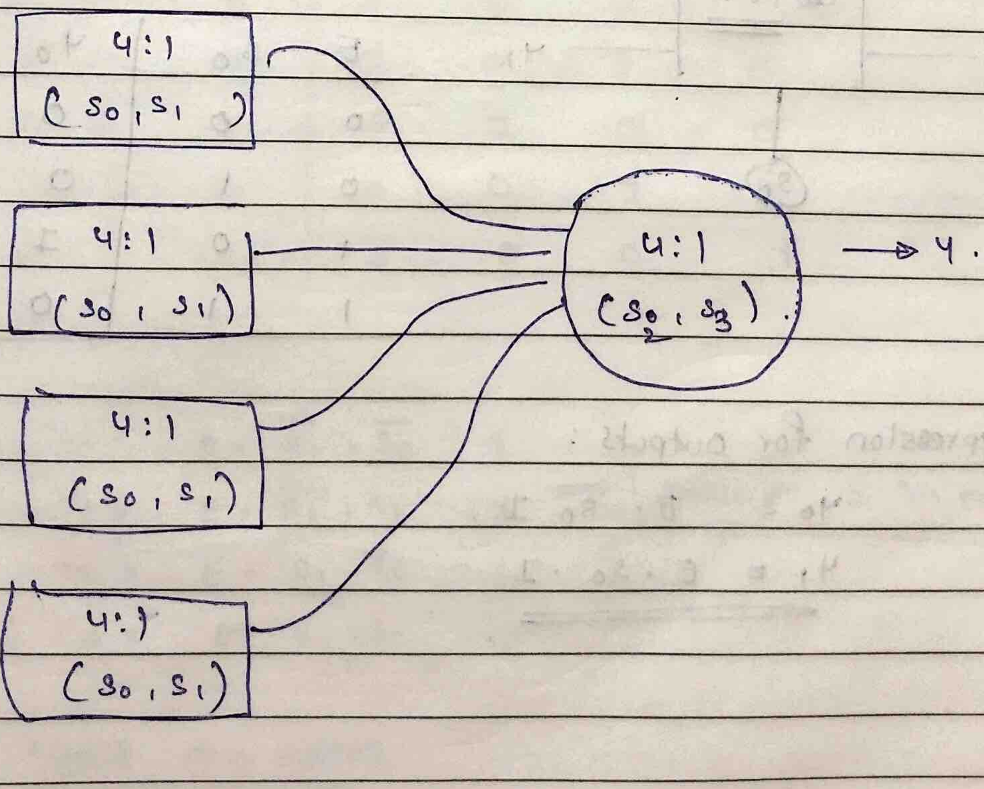
This is the equation without enable.



8:1 multiplexer using 2:1



similarly (16:1)

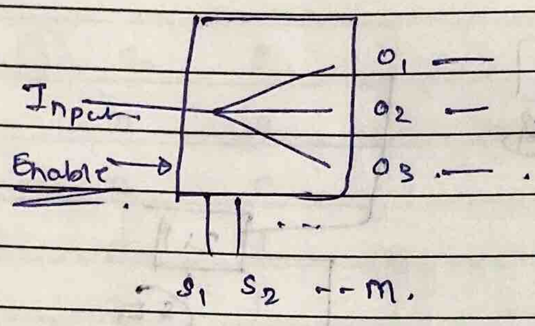


When E = 0, multiplexer & demux don't operate.

* ① T.T according to no. of outputs.
② Eqn for each output.

Demultiplexer (One to Many) (only one input).

* to select output, we have selector lines.

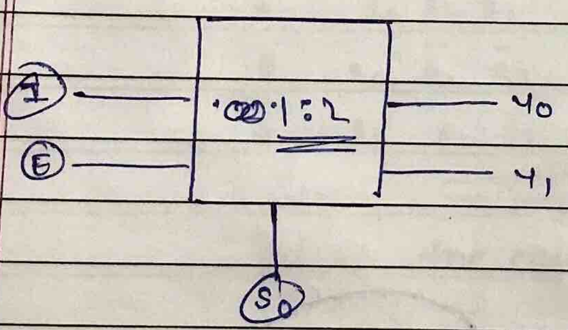


$\boxed{N = 2^m}$

1:2 Demux. , 1 input, 2 output, 1 selection line

* 1:4 Demux , 1 input, 2 outputs, 2 selection lines

1:2 Demux : (E, I & s_0 are inputs)



2 outputs, 1 selectline.

E	s_0	y_0	y_1
0	0	0	0
0	1	0	0
1	0	1	0
1	1	0	1

expression for outputs :

$y_0 = E \cdot \overline{s_0}\, I$.

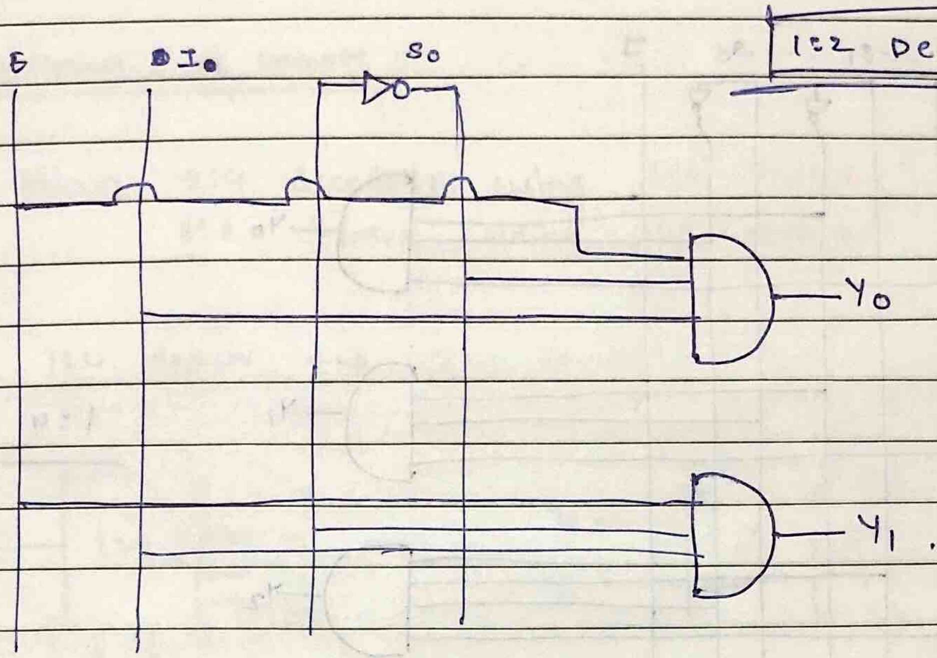
$y_1 = E \cdot s_0 \cdot I$

① Truth table. Include E, S, I. inputs Y output.

② . Eqn of Y including I, S & E.

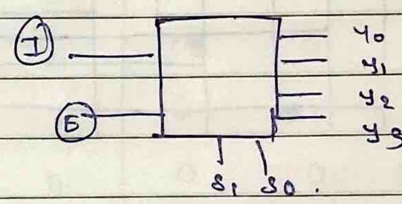
③ Plot eqn.

1:2 Demux.



1:4 Demux.

4 outputs, 2 selector lines.



① Truth tables.

E	S_1	S_0	Y_0	Y_1	Y_2	Y_3
0	X	X	0	0	0	0
1	0	0	I	0	0	0
1	0	1	0	I	0	0
1	1	0	0	0	I	0
1	1	1	0	0	0	I

② Eqn

$Y_0 = E \cdot \overline{S_1} \cdot \overline{S_0} \cdot I$

$Y_1 = E \cdot \overline{S_1} \cdot S_1 \cdot I$

$Y_2 = E \cdot S_1 \cdot \overline{S_0} \cdot I$

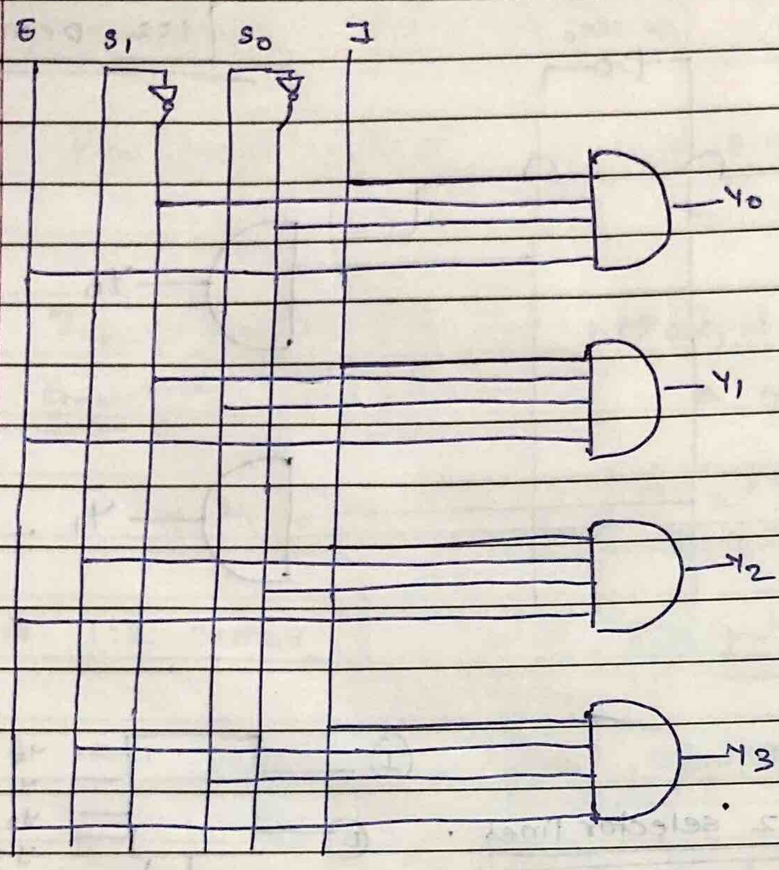
$Y_3 = E \cdot S_1 \cdot S_0 \cdot I$

Include I in eqn.

4 input and gates.

Truth table , LHS = input
RHS = output .

Eqn(s) must contain input



1:4 demux

\# 1:8 using 2, 1:4 & 1 1:2 (because to take input & 2 1:4)

\# 1:16 using 4, 1:4 , & 1 1:2 (" " " " of 4 1:4)

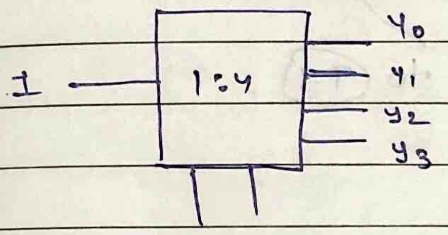
\# ~~NOT Gate using 2×1 multiplexer~~:

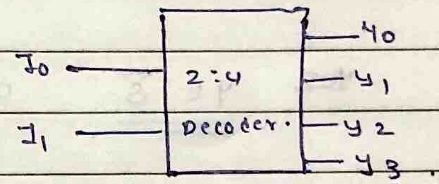
map S_0 & S_1 as input in decoder

Demultiplexer as Decoder :

- Obtain 2:4 decodedr using 1:4 demux.
  3:8 decoder using 1:8 demux.

1:4 demux → 2:4 decoder





I_1	I_0	Y_0	Y_1	Y_2	Y_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

* we don't have to do any thing with outputs in decoder & demux.

* Consider S_1 & S_0 in demux as inputs and I as a 5 volt power supply.
→ thus demux becomes decoder.

∴ $S_0 \rightarrow I_0$
$S_1 \rightarrow I_1$

→ for decoded inputs.
1:4 demux → 2:4 decoder.

* for 3:8 decoder & 1:8 demux

$S_0 \rightarrow I_0$
$S_1 \rightarrow I_1$
$S_2 \rightarrow I_2$

* Consider S_0, S_1 & S_2 as inputs & I as 5 volt power in demux = ~~~~ Decoder.

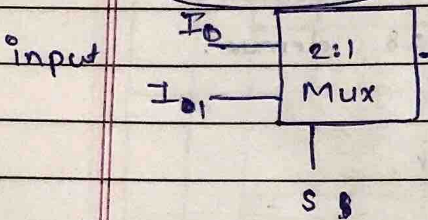
TT
LHS → S
RHS → Y(I)

① T.T
② Eqn of y interms of S.

① TT of 2:1.
② Eqn of 2:1
③ apply values to S, I_1 & I_0 ($S = A$) such that we get required Gate.

Basic logic gates using Mux: (2:1).

* NOT Gate

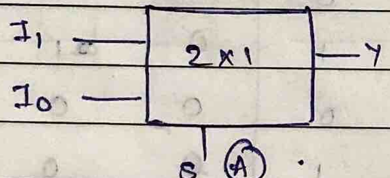


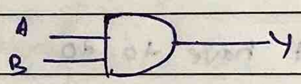
Truth table

S	Y
0	1
1	0

$Y = \overline{S}$ according to (T.T) ∴

* AND Gate using (2:1):





$Y = A \cdot B$

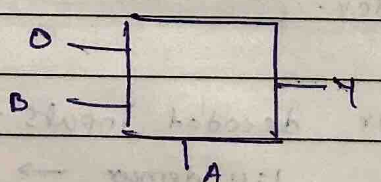
y =

E	S	Y
0	x	0
1	0	I_0
1	1	I_1

∴ TT → $Y = \overline{E} \cdot \overline{S} I_0 + E \cdot S \cdot I_1$

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

put $S = A$, $I_1 = B$, $I_0 = 0$



$y = \overline{A} \cdot 0 + A \cdot B$

$y = AB$ ⇒ AND.

* OR Gate ⇒ (A + B)

$Y = \overline{S} \cdot I_0 + S I_1$.

put $I_0 = B$, $S = A$.

$I_1 = 1$.

E	S	Y
1	x	
1	0	I_0
1	1	I_1

either I_0 or I_1 = value & other = B

① Truth table of 2:1
② Gen of 2:1
③ S = A

I_0 + I_1 (value + B)

④ Truth table of Gate
⑤ Determine value of I_0 & I_1. (by grouping T.T. in pairs of 2)

PAGE No.
DATE / /

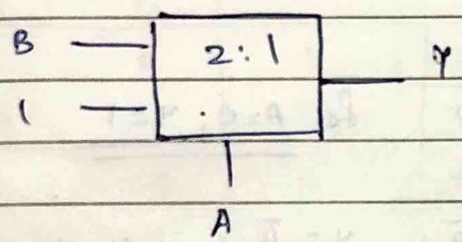
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Y = B. ~~despite~~.

Y = 1

Despite B = 1, Y = A = 1.

$\therefore I_0 = B, \quad I_1 = 1$



NOR Gate using 2:1

S = A.

S	Y
0	I_0
1	I_1 .

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

NOR →

A	B	Y	
0	0	1	$Y = \bar{B}$
0	1	0	
1	0	0	$Y = 0$
1	1	0	

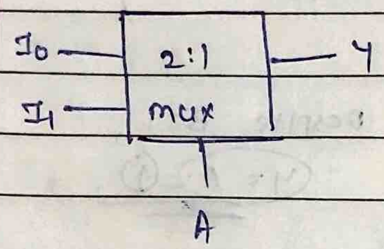
Y = complement of B $= \bar{B} = I_0$.

Y = 0, irrespective of A & B. $= I_1$.

$\boxed{\bar{A} \cdot \bar{B}} + A \cdot 0 \quad =$ Norgate.

$S = A$

* Nand Gate (2:1 mux)



S	Y
0	I_0
1	I_1.

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

NAND

A	B	Y	
0	0	1	$A=0$
0	1	1	$Y=1$
1	0	1	$Y = \bar{B}$
1	1	0	

for $A=0$, $Y=1$

$Y = \bar{B}$, y is always complement of B.

$I_0 = 1$, $I_1 = B$.

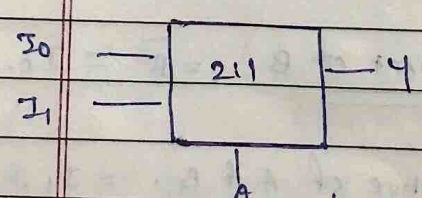
$= \bar{S} I_0 + S I_1$

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

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

$= \cancel{(\bar{A} + A)} \cdot (\bar{A} + \bar{B}) \quad = \bar{A} + \bar{B} \quad = \text{NAND}$

* XOR (2:1)



$\rightarrow \bar{S} I_0 + S I_1.$

XOR

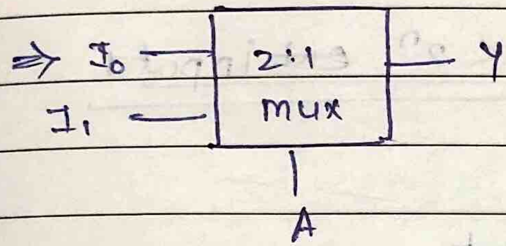
A	B	Y	
0	0	0	$Y = B$
0	1	1	
1	0	1	$Y = \bar{B}$
1	1	0	

$\therefore I_0 = B$, $I_1 = \bar{B}$

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

$= A \oplus B$.

* XNOR Gate using mux 2:1



S	Y
0	I_0
1	I_1

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

XNOR

A	B	Y	
0	0	1	$Y = \bar{B}$
0	1	0	
1	0	0	$Y = B$
1	1	1	

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

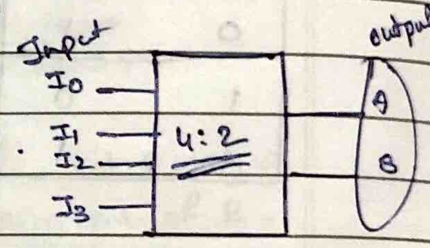
At one time only one input is high rest all law.

Encoder :

- has n outputs. & $M < 2^n$ ~~out~~ inputs

More inputs less output

* * * It has been assumed that at a given time only one input is high & depending on that we get the output



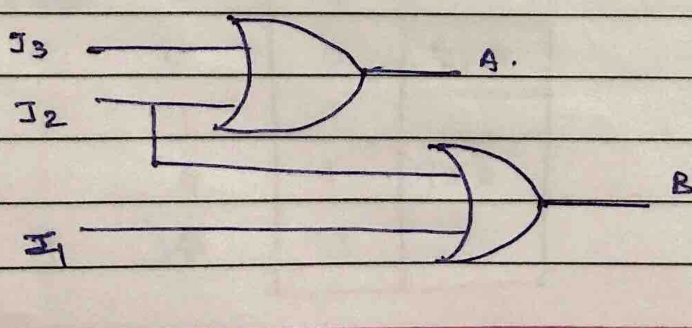
4:2 encoder , 4 inputs 2 outputs.

Truth table of 4:2 Encoder.

I_0	I_1	I_2	I_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 = eqn = $D_2 + D_3$ = A~~ ~~A is high for I_2 + I_3~~
$D_1 + D_3$ = B

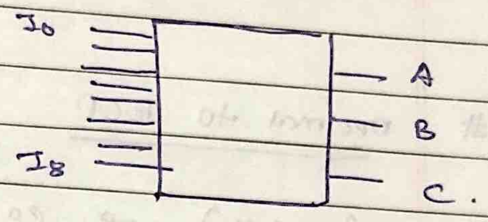
Eqn of output $A = I_3 + I_2$. $B = I_1 + I_2$. } Based on, output should include 1.



Only single input is high at a time.

PAGE No.	
DATE	/ /

8 : 3 encoder. 8 inputs & 3 outputs.



Truthtabele

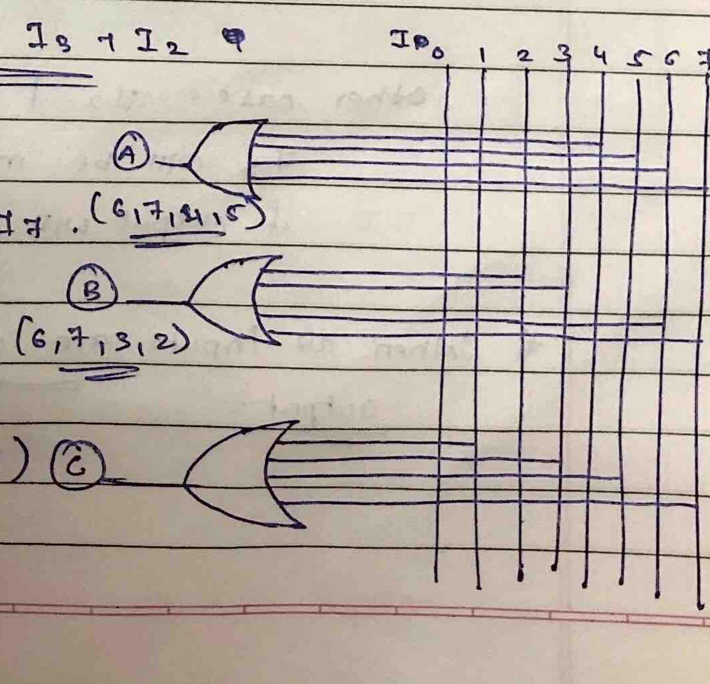
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	A	B	C
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	A	B	C

Ean of A → $I_4 + I_5 + I_6 + I_7$.
i.e. (A is high at)

Ean of B → $I_6 + I_7 + I_3 + I_2$
i.e (B is high for input).

Ean of C → $I_1 + I_3 + I_5 + I_7$.

Diagram



(1,3,5,7)

At one pt only one input pin is high.

PAGE No.
DATE / /

Applicaion of Encoder : saving i/o pins

Decimal to BCD (10:4)

(10:4) → each input = decimal.
each output = BCD.

① T.T. → 10 inputs - (LHS)
→ 4 outputs of BCD at RHS.

② Eqn
③ Plot.

Priority Encoder:

To set priority to the inputs.

if I_1 & I_2 are high simultaneously,
I_2 will be more priority.
& output will be from I_2.

Other case. I_2 & I_5 are high simultaneously
I_5 will be more priority.
& output will be from I_5.

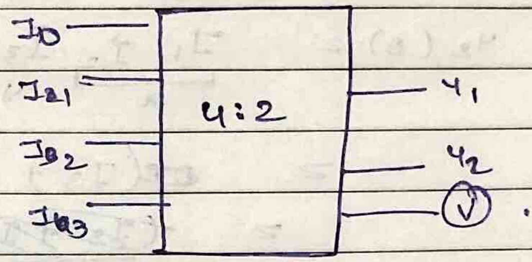
When all inputs are zero → we have a V pin at output.

only one input high at a time.

① TT of priority Encoder.
② always check the value of highest priority
③ Eqn.

Priority Encoder

* assume I_0 = low priority
& I_3 = max priority



If I_3 = high, no need to check rest.

Truth table

← priority

I_3	I_2	I_1	I_0	Y_1	Y_2
0	0	0	0	x	x
0	0	0	1	0	0
0	0	1	x	0	1
0	1	x	x	1	0
1	x	x	x	1	1

Truth table → always check the highest priority

I_0	I_1	I_2	I_3	Y_1 (A)	Y_2 (B)	V
1	0	0	0	0	0	1
x	1	0	0	0	1	1
x	x	1	0	1	0	1
x	x	x	1	1	1	1

$Y_1 \Rightarrow (I_2 + I_3)$ $\quad$ $Y_2 \Rightarrow (I_1 + I_3)$

high at $(I_2 + I_3)$ $\quad$ high at $(I_1 + I_3)$

$x + (\bar{x} \cdot y) = \cancel{(x+\bar{x})}^{1} \cdot (x+y)$

$= (x+y)$

PAGE No.
DATE / /

$V = I_0 + I_1 + I_2 + I_3$.

$y_1(A) = I_3 + \overline{I_3} \cdot I_2 = (I_3 + I_2)$

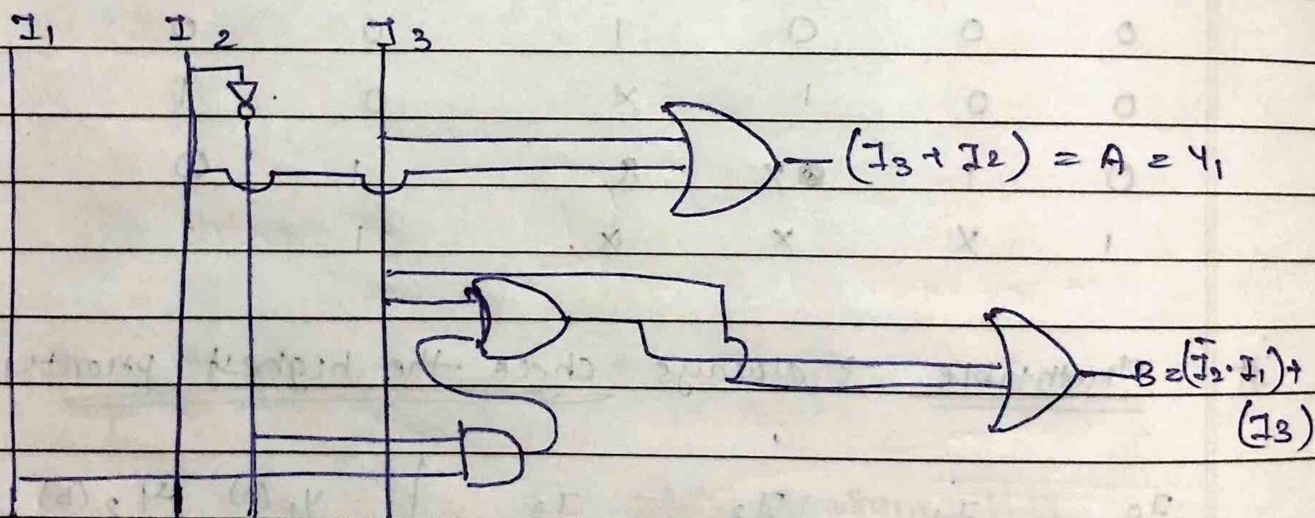
$y_2(B) = \underbrace{I_1 \overline{I_2}}_{x} \underbrace{\overline{I_3}}_{y} + \underbrace{I_3}_{y}$.

$= (I_3 + \overline{I_3}) \cdot I_2)$

$= \cancel{(I_3 + I_2)} (I_3 + \overline{I_2} \cdot I_1)$

$y_1(A) = (I_3 + I_2)$

$y_1(B) = (I_3 + (\overline{I_2} \cdot I_1))$



(8:3) → same working as 4:2 $I_0 -- I_7$ f(A B C) V.

(8:3) using (4:2).

study during exams

only one of the output is high at a time.

PAGE No.	
DATE	/ /

Decoder : (4:10) (3:8)

* More outputs less inputs.

* used for → BCD to Decimal.

3:8 decoder , 3 inputs, 8 outputs.

A	B	C	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

$D_0 = \bar{A}\,\bar{B}\,\bar{C}$

$D_1 = \bar{A}\,\bar{B}\,C$

$D_2 = \bar{A}\,B\,\bar{C}$

$D_3 = \bar{A}\,B\,C$

$D_4 = A\,\bar{B}\,\bar{C}$

$D_5 = A\,\bar{B}\,C$

$D_6 = A\,B\,\bar{C}$

$D_7 = A\,B\,C$

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

$(\bar{A}\,\bar{B}\,\bar{C})$ —— D — D_0

$(\bar{A}\,\bar{B}\,C)$ —— D — D_1

$(\bar{A}\,B\,\bar{C})$ —— D → D_2

$(\bar{A}\,B\,C)$ —— D → D_3

$(A\,\bar{B}\,\bar{C})$ —— D → D_4

$(A\,\bar{B}\,C)$ —— D → D_5

$(A\,B\,\bar{C})$ —— D → D_6

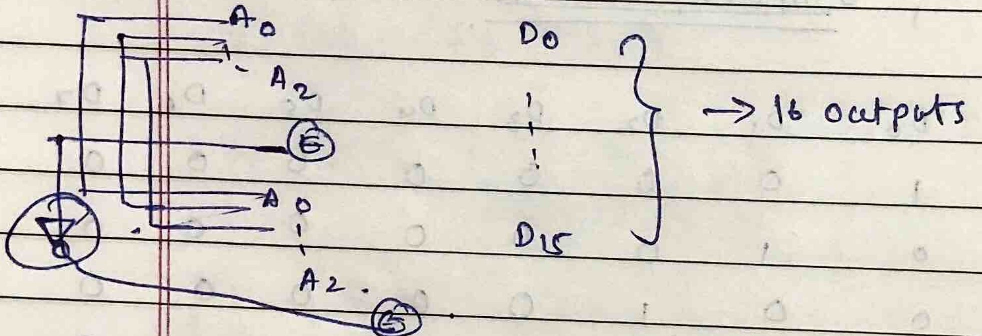
$(A\,B\,C)$ —— D → D_7

for BCD to Decimal. (~~10~~ 4:10).

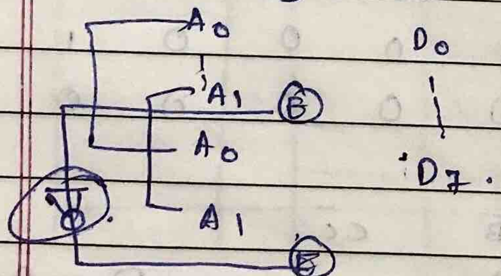
BCD → input (A, B, C, D).

output → Decimal (value of decimal).

4:16 using 3:8 decoders.



3:8 using 2:4.



#

1. Outputs remains as it is
2. combine inputs as single input
3. Enable (use as switch NOT Gate)