Number system

(face value / Place value) weighted — binary, O; octal; Decimal; hexadecimal; BCD

unweighted — Gray code; Excess-3 code

(2)

Any No. such as $8936 = 8$ thousand $+ 900 + 30 + 6$

$= 8\times10^3 + 9\times10^2 + 3\times10^1 + 6\times10^0$

So any No. with decimal can be represented as

$$\ldots a_4 a_3 a_2 a_1 a_0 . a_{-1} a_{-2} a_{-3} a_{-4} \ldots$$

the co-efficient a_j are any of ten digit. (0, 1, ... 9)

and subsript j gives place value and hence power of ten by which the cofficient must be multiplied.

$$\ldots + a_2\times10^2 + a_1\times10^1 + a_0\times10^0 . + a_{-1}\times10^{-1} + a_{-2}\times10^{-2} + \ldots$$

A decimal system is said to be of base or radix 10, because it uses 10 digits and cofficient are multiplied by power of 10

In general a number expressed in a base r- system has cofficient multiplied by base of r.

$$a_n r^n + \ldots + a_2 r^2 + a_1 r^1 + a_0 r^0 + a_1 r^{-1} + a_2 r^{-2} + \ldots a_{-m} r^{-m}$$

The cofficient a_j ranges $\rightarrow$ 0 to $r-1$

for example. 0 base 5 No. 18

$(4021.2)_5 = 4\times5^3 + 0\times5^2 + 2\times5^1 + 1\times5^0 + 2\times5^{-1}$

if we calculate $= (511.4)_{10}$

3

Cofficient value - 0, 1, 2, 3, 4

Number system	base	Diff cofficient.
Binary.	2	0, 1
Octal	8	0, 1, 2, 3, 4, 5, 6, 7.
Decimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9.
hexadecimal.	16	0, 1, 2, ---- 9, A, B, C, D, E, F

In computer

$1K = 2^{10}$ byte

$1M = 2^{20}$ byte

$1G = 2^{30}$ byte.

1 byte = 8 bit

4G Hardis = 2^{32} byte

$(10101)_2 \rightarrow 1\times2^4 + 1\times2^2 + 0\times2^1 + 1\times2^0$

$32 + 4 + 1 = (37)_{10}$

$(B65F)_{16} \rightarrow 11\times16^3 + 6\times16^2 + 5\times16^1 + 15\times16^0 = 46.683.$

Base conversion.

Decimal to others. $\longrightarrow (\quad)_r$

$(x_1x_2x_3 . y_1y_2y_3)_{10}$

$(\underbrace{x_1x_2x_3}_{\text{Integer}} . \underbrace{y_1y_2y_3}_{\text{fraction}})$

Integer.

$$\begin{array}{c|c|} r & x_1x_2x_3 \\ \hline & \\ & 1 \end{array}$$

Fractional.

$.y_1y_2y_3 \times r$

$x . y_4y_5y_6$

$.y_4y_5y_6 \times r$

$z . y ....$

decimal to Binary

$(25.25)_{10} \longrightarrow (\quad)_2$

2	25	1
2	12	0
2	6	0
2	3	1
	1	

$.25 \times 2 = 0.50$
$.50 \times 2 = 1.00$

$(1001.01)_2$

11001.

4

decimal to octal

$(25.25)_{10} \longrightarrow (\quad)_8$

8	25	1
	3	

$.25 \times 8 = 2.00$

$(31.2)_8$

$(37.625)_{10} \longrightarrow (\quad)_2$

$.625 \times 2 = 1.250$
$.250 \times 2 = 0.500$
$.50 \times 2 = 1.00$

2	37	1
2	18	0
2	9	1
2	4	0
2	2	0
	1	

$(100101.101)_2$

decimal to hexadecimal

$(127)_{10} \longrightarrow (\quad)_{16}$

127
112
15

16	127	15
	7	

$(7F)_{16}$

Others to Decimal

$(x_3 x_2 x_1 x_0)_r \longrightarrow (\quad)_{10}$

~~$(x_3 x_2 x_1 x_0 \cdot x_{-1} x_{-2} x_{-3})_r \longrightarrow (\quad)_{10}$~~

~~Binary to decimal~~

$(x_3 x_2 x_1 x_0 \cdot x_{-1} x_{-2} x_{-3})_r \longrightarrow (\quad)_{10}$

$\rightarrow (x_3 r^3 + x_2 r^2 + x_1 r^1 + x_0 \cdot r^0 + x_{-1} r^{-1} + x_{-2} r^{-2} + x_{-3} r^{-3})_{10}$

Binary to decimal

$(11010.11)_2 \longrightarrow (\quad)_{10}$

$(1\times2^4 + 1\times2^3 + 0\times2^2 + 1\times2^1 + 0\times2^0 + 1\times2^{-1} + 1\times2^{-2})_{10}$

$(16 + 8 + 2 + .5 + .25)_{10}$

$(26.75)_{10}$

Octal to decimal

$(127.4)_8 \longrightarrow (\quad)_{10}$

$\downarrow$

$(1\times8^2 + 2\times8^1 + 7\times8^0 + 4\times8^{-1})_{10}$

$(64 + $ [illegible] $)_{10}$

([illegible]$)_{10}$

Note that digit 8 and 9 cannot appear in an octal no.

Hexadecimal to decimal

$(B65F)_{16} \longrightarrow (\quad)_{10}$

$B\times16^3 + 6\times16^2 + 5\times16^1 + F\times16^0$

$11\times16^3 + 6\times16^2 + 5\times16^1 + 15\times16^0 = (46687)_{10}$

Binary to Octal & Hexadecimal

Binary to octal:-

Since $2^3 = 8$, each octal digit correspon to three binary digits.

6

So conversion from binary to octal is easily accomplish by ~~Partion~~ Partitioning the binary no. into groups of three digit each, starting from binary point and proceeding to th left and to the right.

$$(\underbrace{010}\ \underbrace{110}\ \underbrace{001}\ \underbrace{011}\,.\,\underbrace{111}\ \underbrace{100}\ \underbrace{000}\ \underbrace{110})_2 \rightarrow (\quad)_8$$

~~added~~

add zero $\rightarrow \underbrace{010}$

if group not completed

$$(2613.7406)_8$$

Binary to Hexadecimal.

Since $2^4 = 16$ each hexadecimal digit correspon to four binary digits.

So by partitioning binary no. into group of four digit each, starting from decimal point and proceeding ~~from~~ t the left and to the left.

$$(10110001101011.111101)_2 \rightarrow (\quad)_{16}$$

$$\underbrace{0010}_{2}\ \underbrace{1100}_{C}\ \underbrace{0110}_{6}\ \underbrace{1011}_{B}\,.\,\underbrace{1111}_{F}\ \underbrace{0100}_{4}$$

$$(2C6B.F4)_{16}$$

(Octal & Hexadecimal) to Binary

(By Reversing the preceding procedure.)

Octal to Binary: Each octal digit is converted to its three digit binary Equivalent

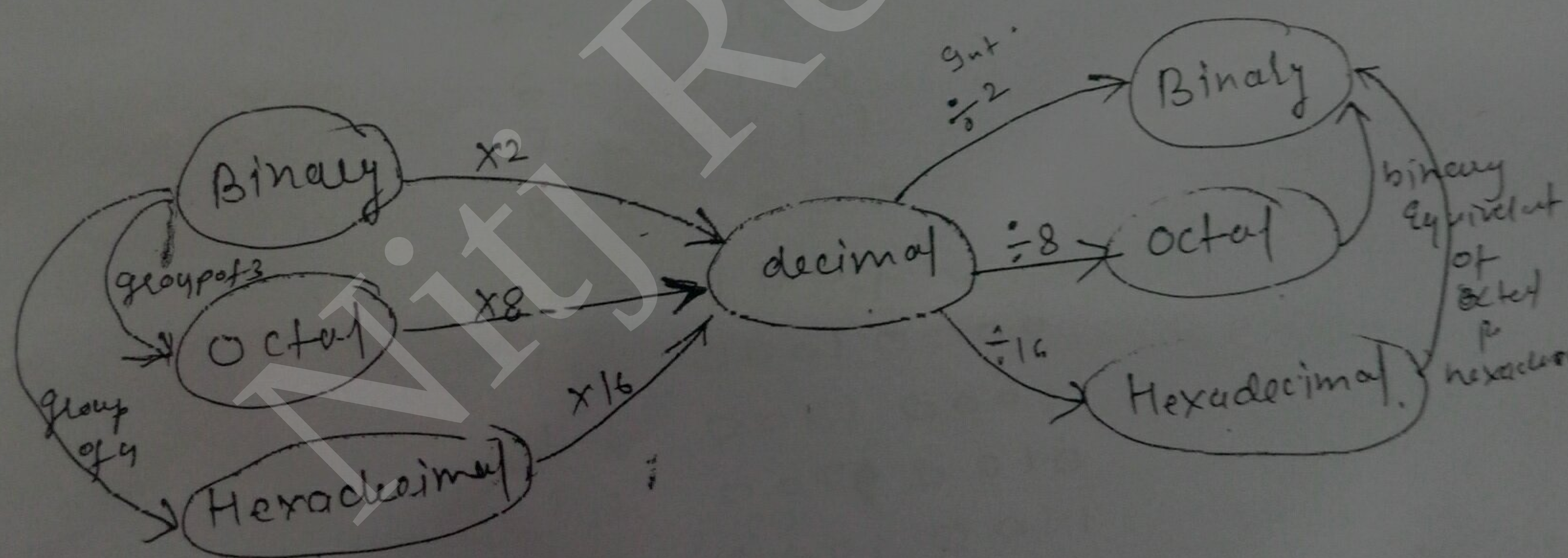
$$(673.124)_8 \rightarrow (\quad)_2$$

$$\underbrace{110}_{6}\ \underbrace{111}_{7}\ \underbrace{011}_{3}\ .\ \underbrace{001}_{1}\ \underbrace{010}_{2}\ \underbrace{100}_{4}$$

Hexadecimal to Binary. Each hexadecimal digit is converted to its four digit binary Equivalent.

$$(306.D)_{16} \rightarrow (\quad)_2$$

$$\overbrace{0011}^{3}\ \overbrace{0000}^{0}\ \overbrace{0110}^{6}\ .\ \overbrace{1101}^{D}$$



BCD Code(Binary Coded Decimal)

Each decimal no. is represented with its 4 bit equ binary equivalent.

such as.

$(396)_{10} \longrightarrow$ BCD $(\underbrace{0011}_{3}\ \underbrace{1001}_{9}\ \underbrace{0110}_{6})_{BCD}$

$(185)_{10} \longrightarrow (\quad)_{BCD}$

$(0001\ 1000\ 0101)_{BCD}$

0 -	0000
1 -	0001
2 -	0010
3 -	0011
4 -	0100
5 -	0101
6 -	0110
7 -	0111
8 -	1000
9 -	1001
10	0001 0000
11	0001 0001
12	0001 0010
13	0001 0011

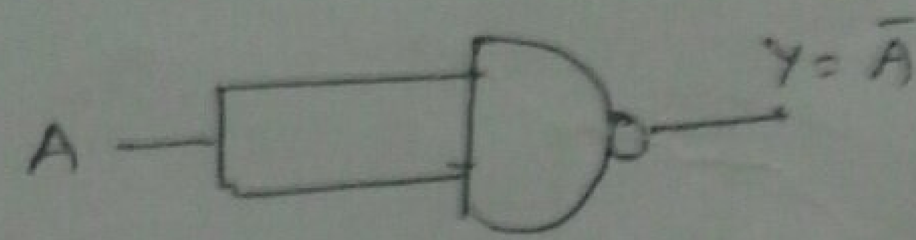
Note

BCD No. are decimal no. are decimal no. and not binary no., although use bits in their representation.

NAND Gate as universal gate

other gate using universal NAND gate.

① NOT $Y = \overline{A}$

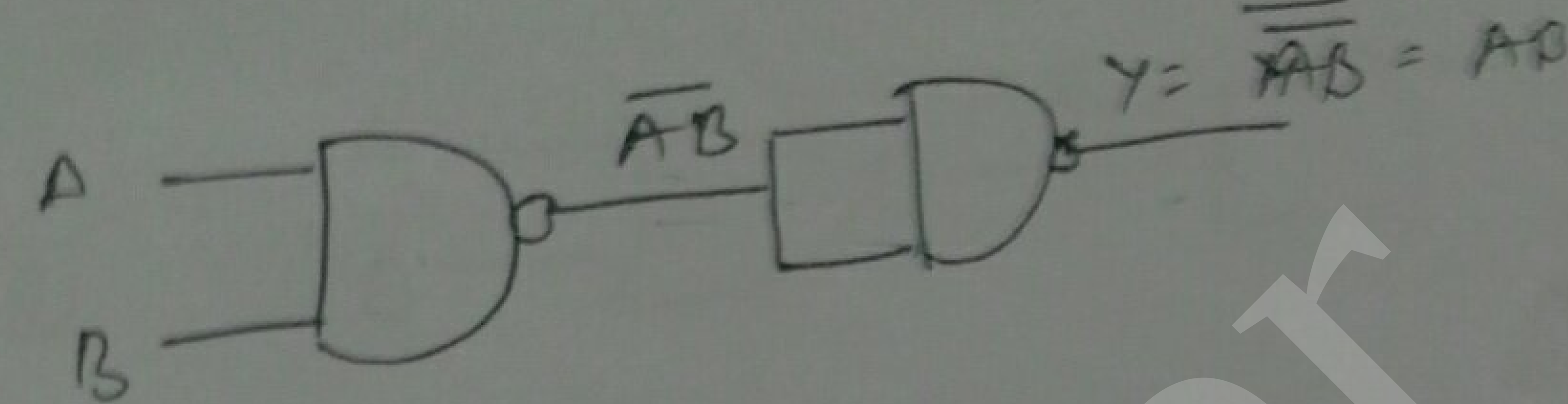


② AND Gate :-

$$Y = AB$$

take bar $\overline{Y} = \overline{AB}$

take once more bar :⇒ $\overline{\overline{Y}} = Y = (\overline{\overline{A \cdot B}})$



③ OR Gate

$$Y = A + B$$

$$\overline{Y} = \overline{A+B}$$

$$\overline{Y} = \overline{A} \cdot \overline{B}$$

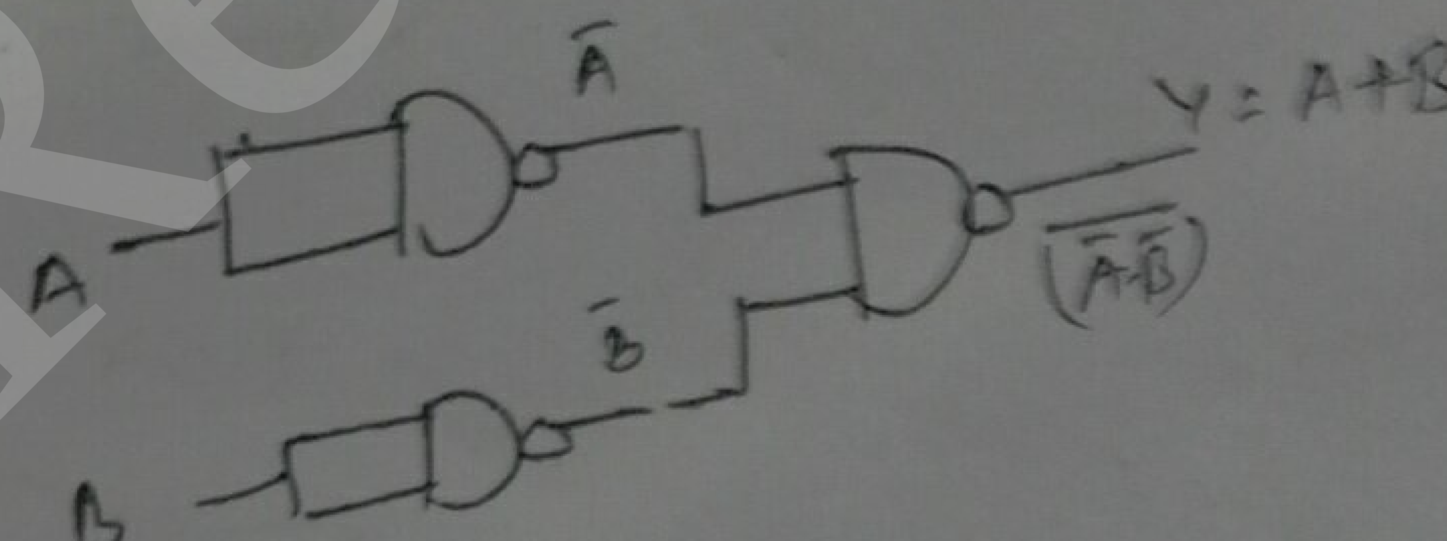
$$\overline{\overline{Y}} = (\overline{\overline{A} \cdot \overline{B}})$$

$$Y = \overline{\overline{A}} + \overline{\overline{B}}$$

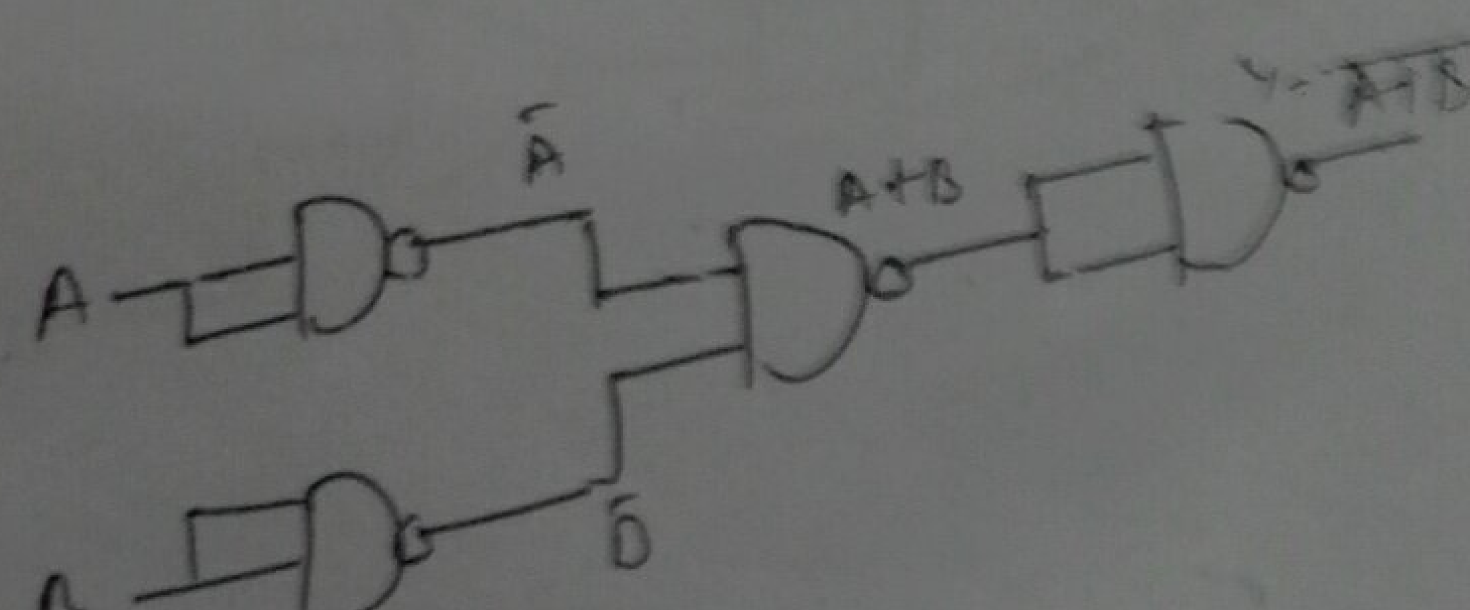
$$(Y = A+B)$$

Use demorgan's

$$\left\{ \begin{array}{l} \overline{A+B} = \overline{A} \cdot \overline{B} \\ \overline{A \cdot B} = \overline{A} + \overline{B} \end{array} \right\}$$



④ NOR Gate $Y = \overline{A+B}$

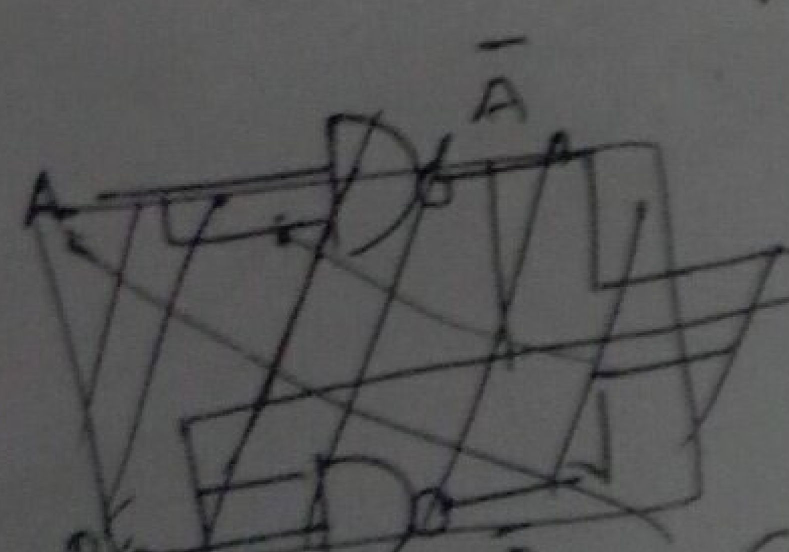


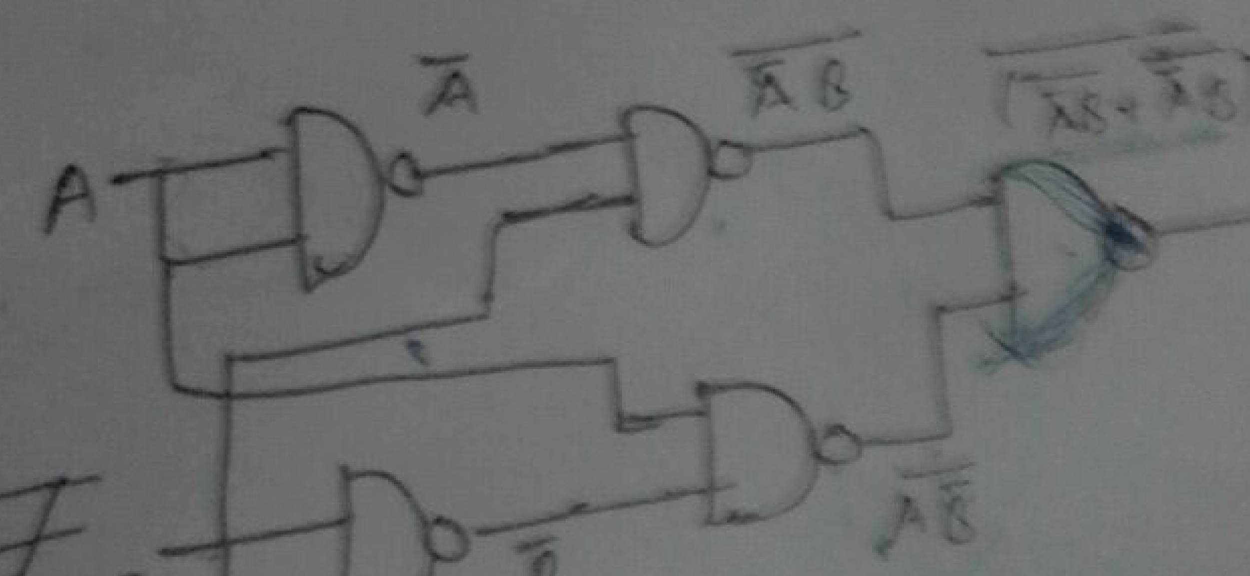
⑤ Ex-OR - $Y = A\overline{B} + \overline{A}B$

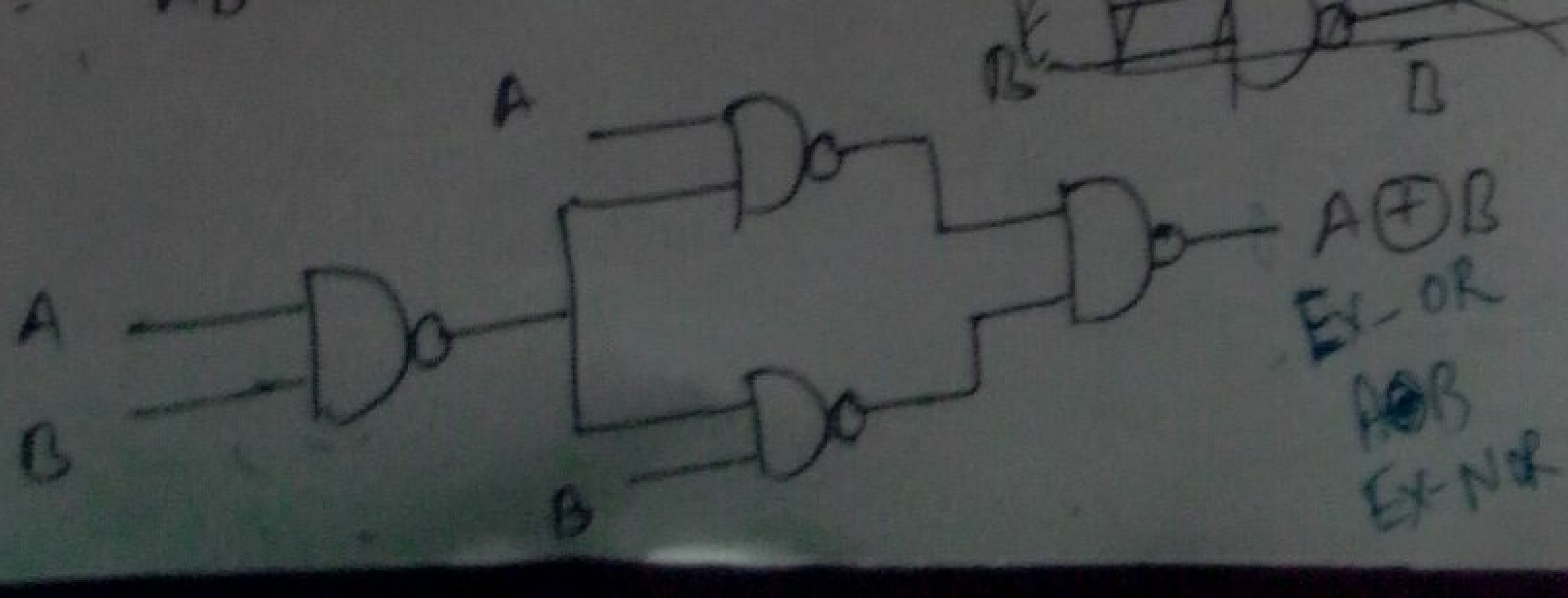
$$\overline{Y} = \overline{A\overline{B} + \overline{A}B}$$

$$\overline{Y} = \overline{A\overline{B}} \cdot \overline{\overline{A}B}$$

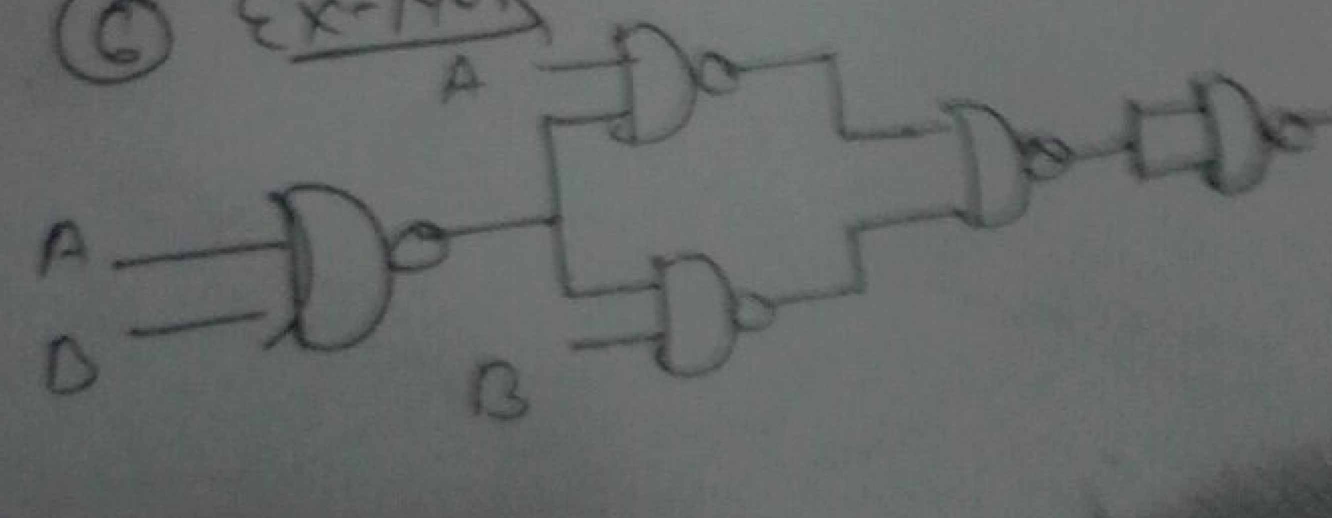
$$\overline{\overline{Y}} = \overline{\overline{A\overline{B}} \cdot \overline{\overline{A}B}}$$





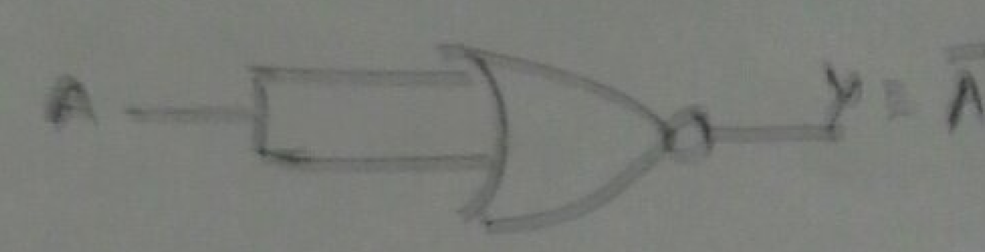


⑥ Ex-NOR



NOR as universal gate:

① NOT



10

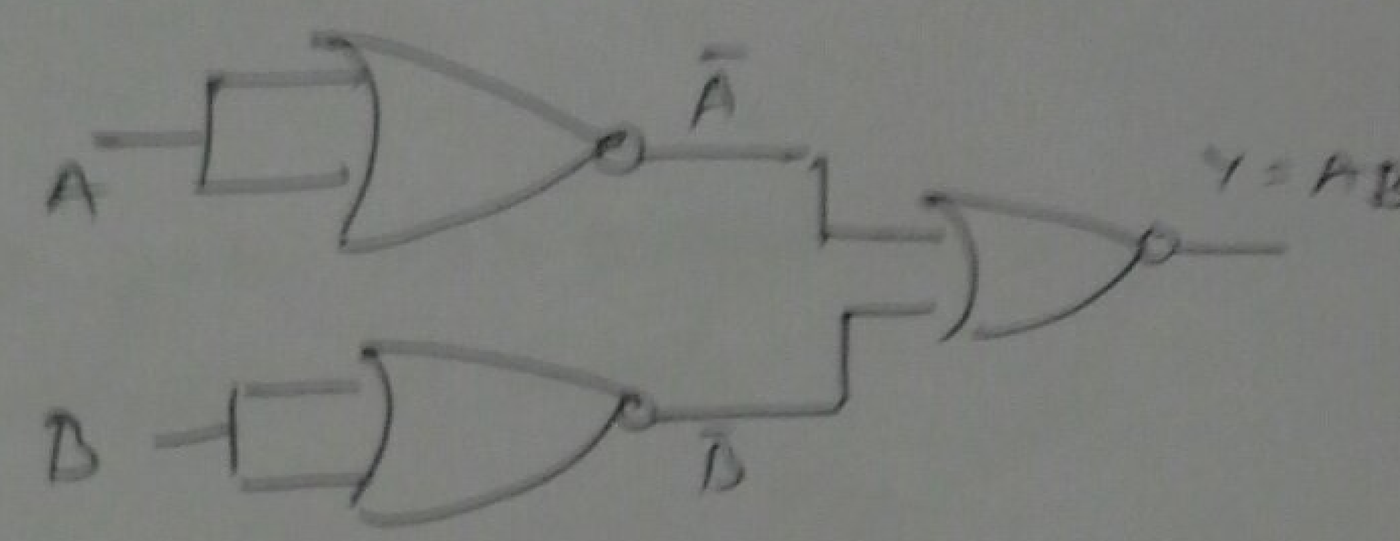
② AND

$Y = AB$

$\overline{Y} = \overline{AB} = \overline{A} + \overline{B}$

$\overline{\overline{Y}} = \overline{(\overline{A} + \overline{B})}$

$Y = AB$

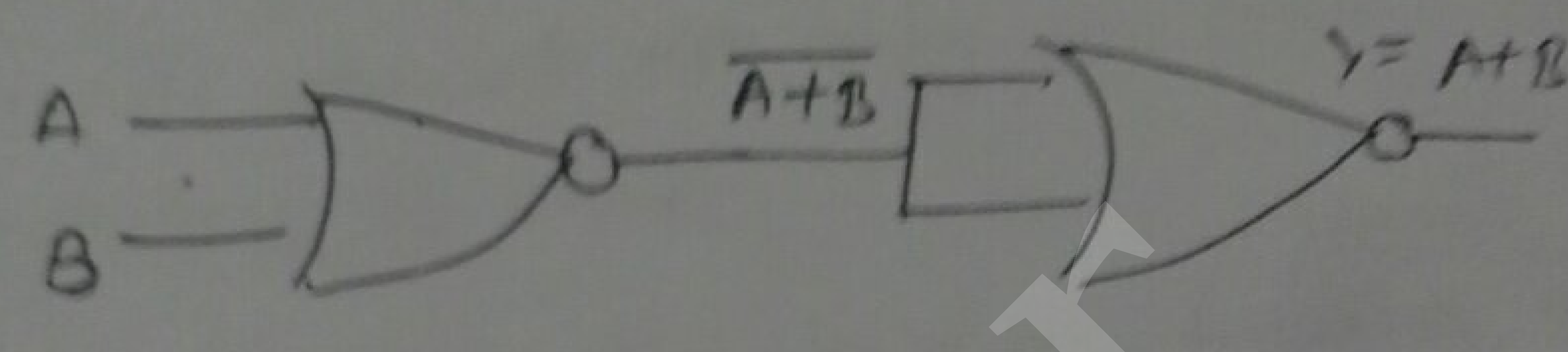


③ OR

$Y = A + B$

$\overline{Y} = \overline{A+B}$

$Y = \overline{\overline{Y}} = \overline{\overline{(A+B)}}$



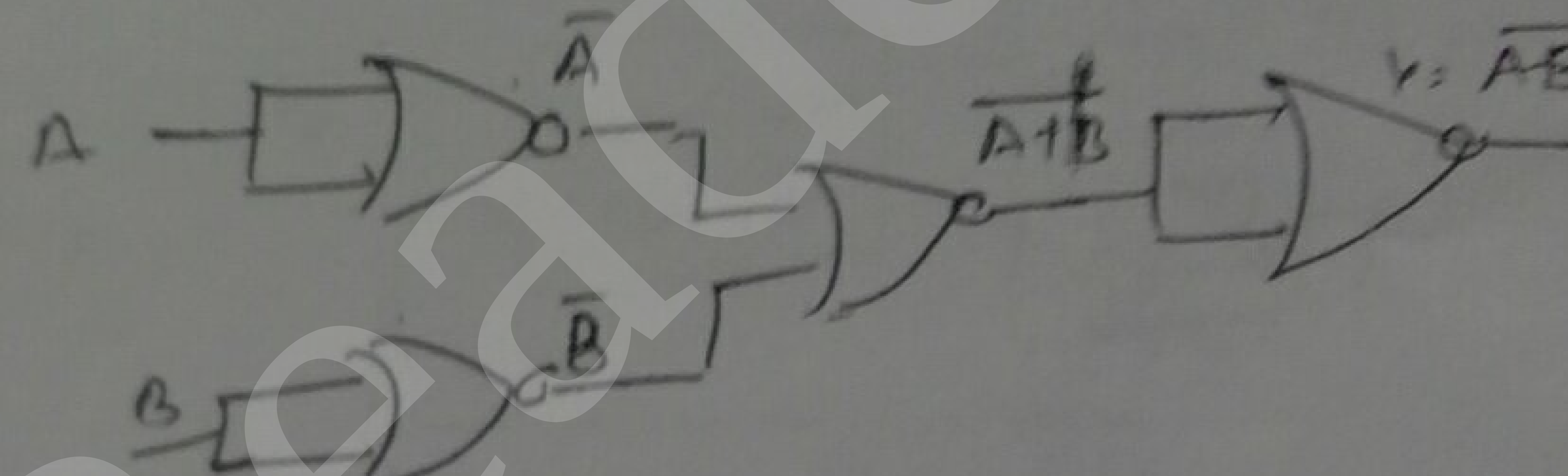
④ NAND

$Y = \overline{AB}$

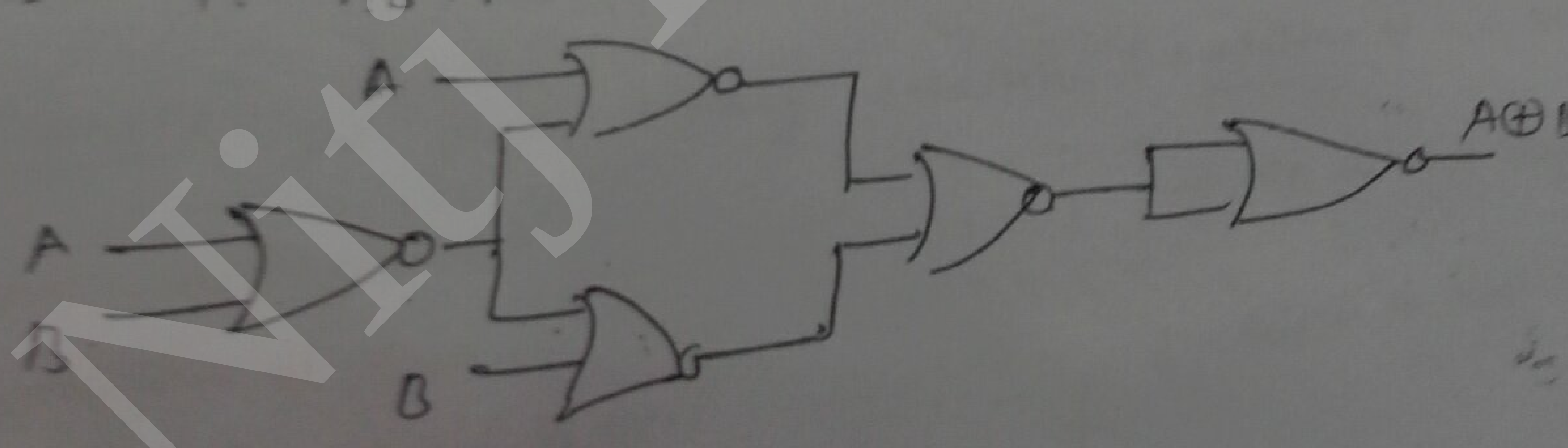
$Y = \overline{A} + \overline{B}$

$\overline{Y} = \overline{\overline{A} + \overline{B}}$

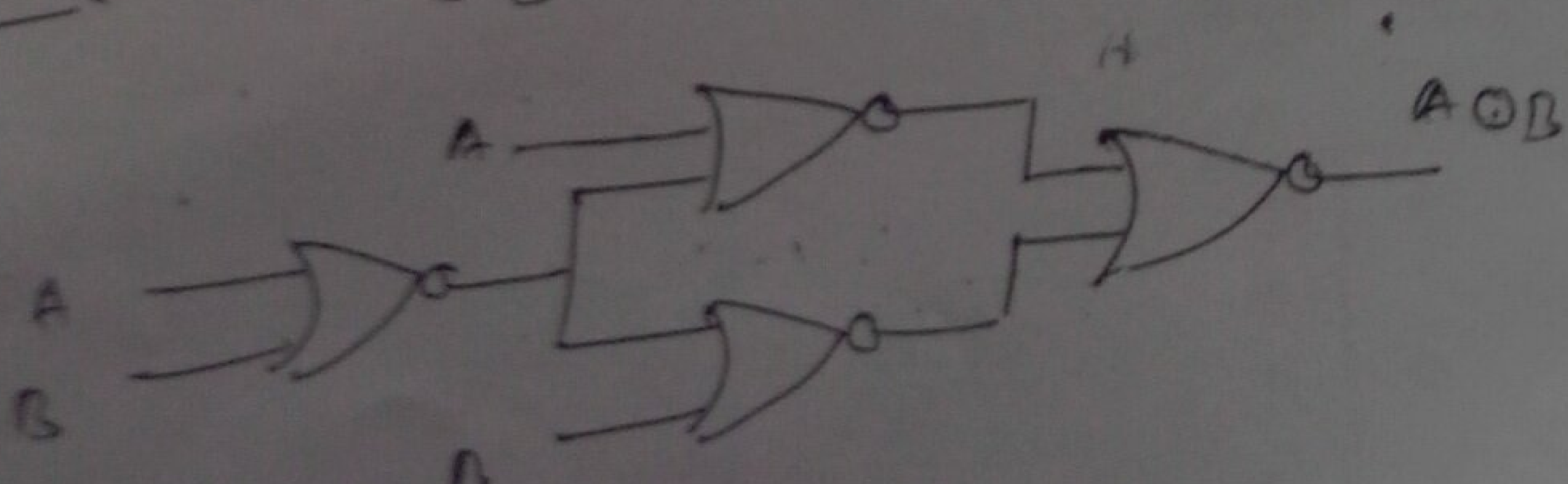
$Y = \overline{\overline{Y}} = \overline{\overline{\overline{A} + \overline{B}}}$



⑤ Ex-OR - $Y = A\overline{B} + \overline{A}B = A \oplus B$



⑥ Ex-NOR $Y = A \odot B$

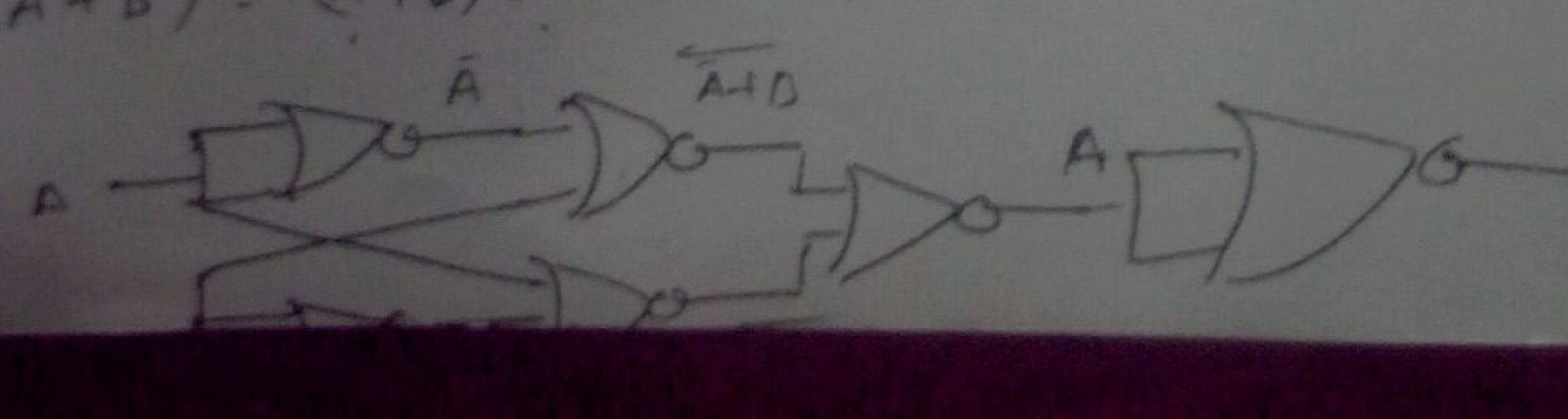


$\overline{Y} = \overline{A\overline{B} + \overline{A}B}$

$\overline{Y} = (\overline{A\overline{B}})(\overline{\overline{A}B}) = (\overline{A} + \overline{\overline{B}})(\overline{\overline{A}} + \overline{B}) = (\overline{A} + B)(A + \overline{B})$

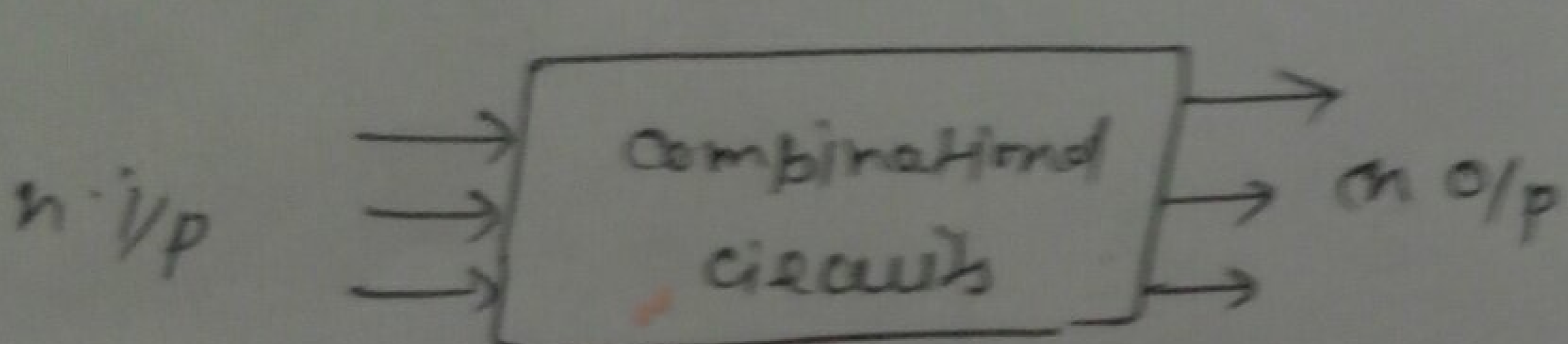
$\overline{\overline{Y}} = \overline{(\overline{A} + B)(A + \overline{B})}$

$Y = \overline{(\overline{A} + B)} + \overline{(A + \overline{B})}$

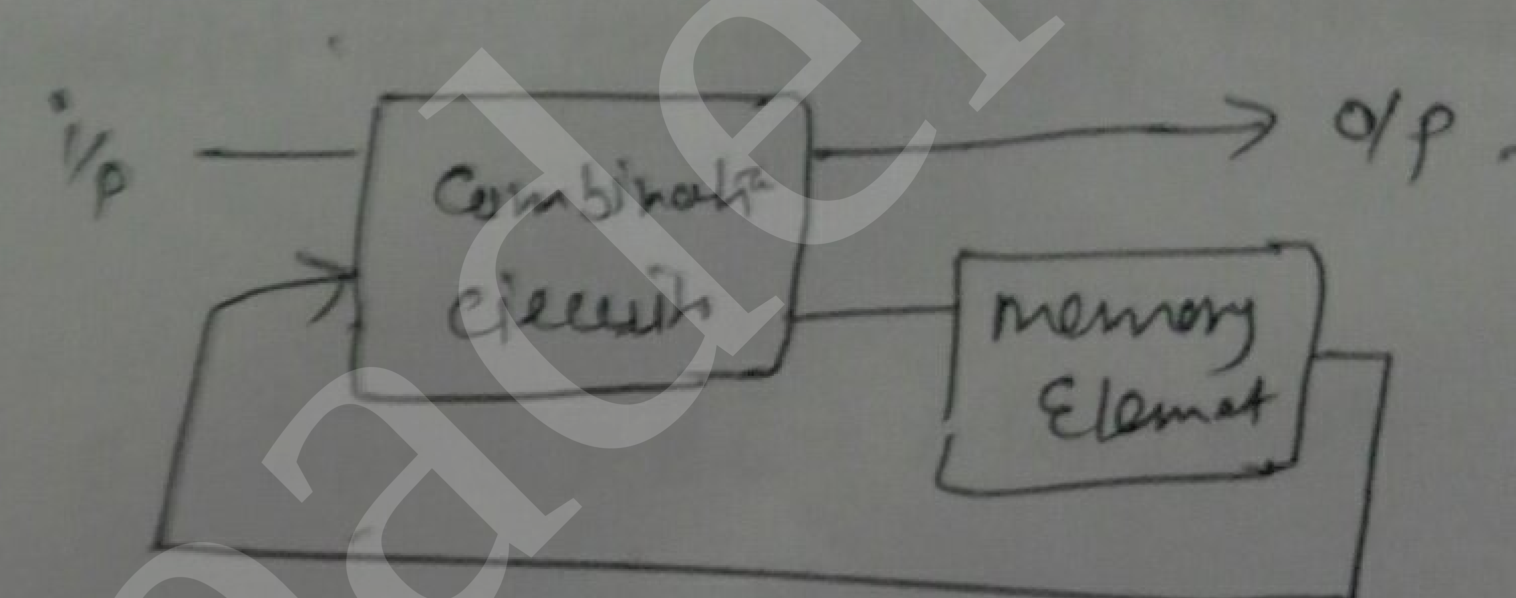


Combinational circuits consist of logic gates whose output at any time are determined from the present combination of inputs
No feedback or memory required.

Sequential circuits - consist of storage element in addition logic gates
The output of sequential circuits depend not only on present values of inputs but also on past values of inputs,
The circuit behaviour must be specified by time sequence of input & internal states.



Combinational circuit

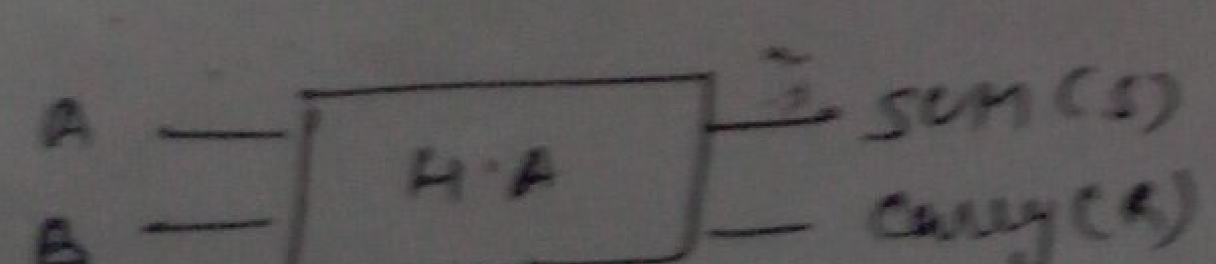


Sequential Ckt

① HALF ADDER we know binary addition (addition of two binary digit)

$0+0=0$
$0+1=1$ → o/p in one digit
$1+0=1$
$1+1=10$ — o/p - in two digit
carry

Combination circuit that perform addition of two bit is called half adder



Truth table.

A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

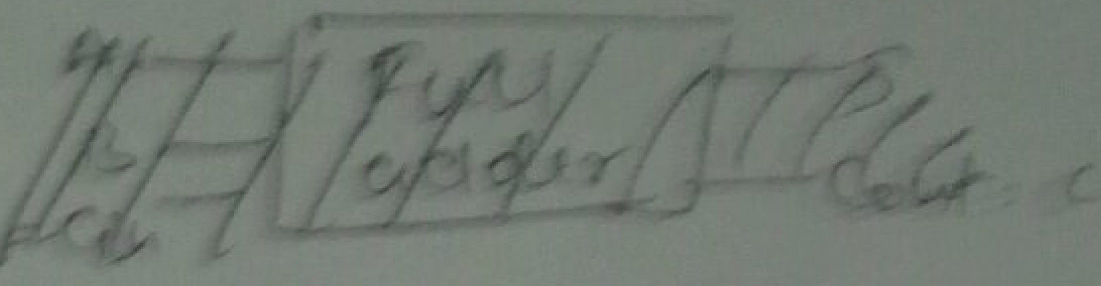
Simplified Boolean function for two output can be obtained directly from truth table.

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

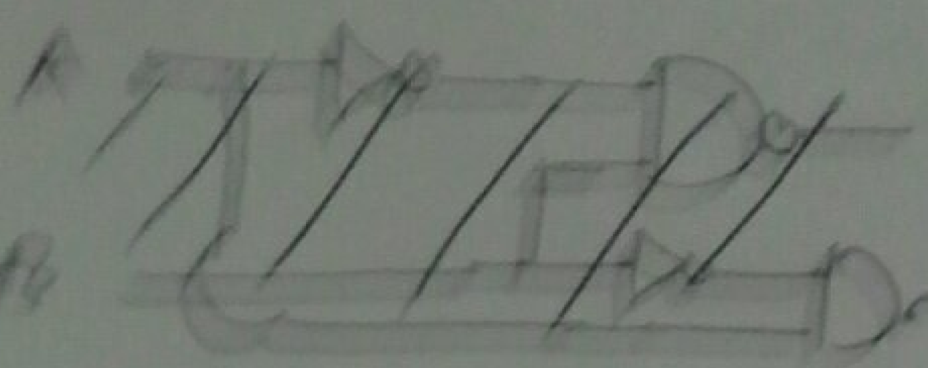
$C = AB$

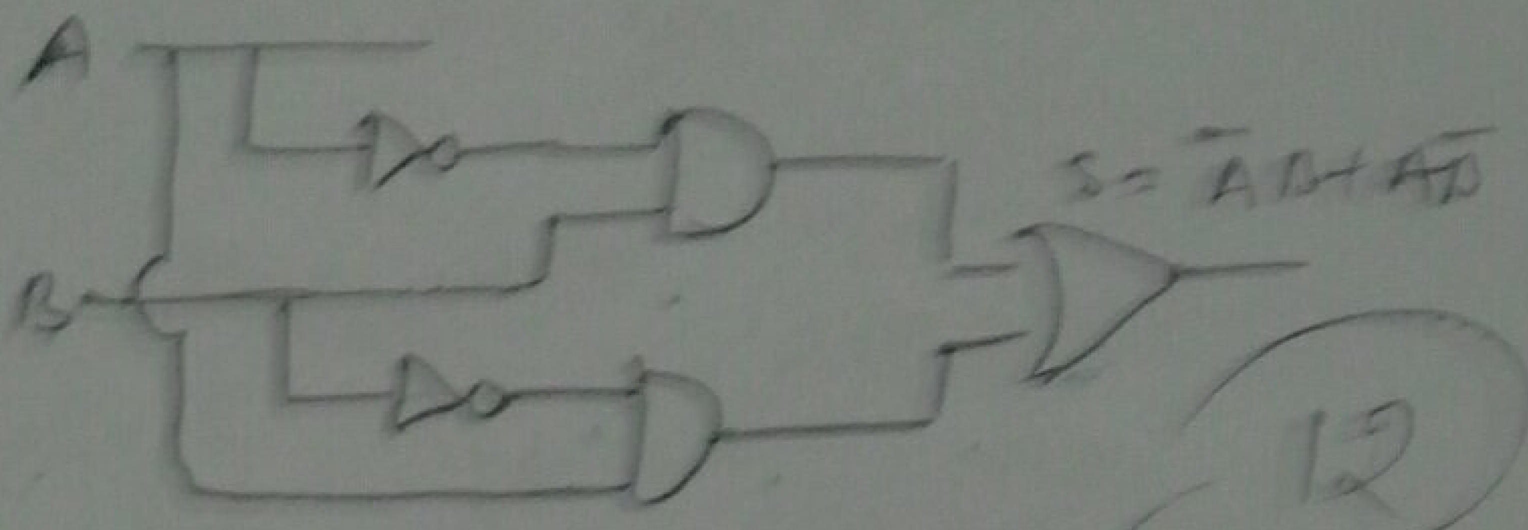
~~Logic diagram~~

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

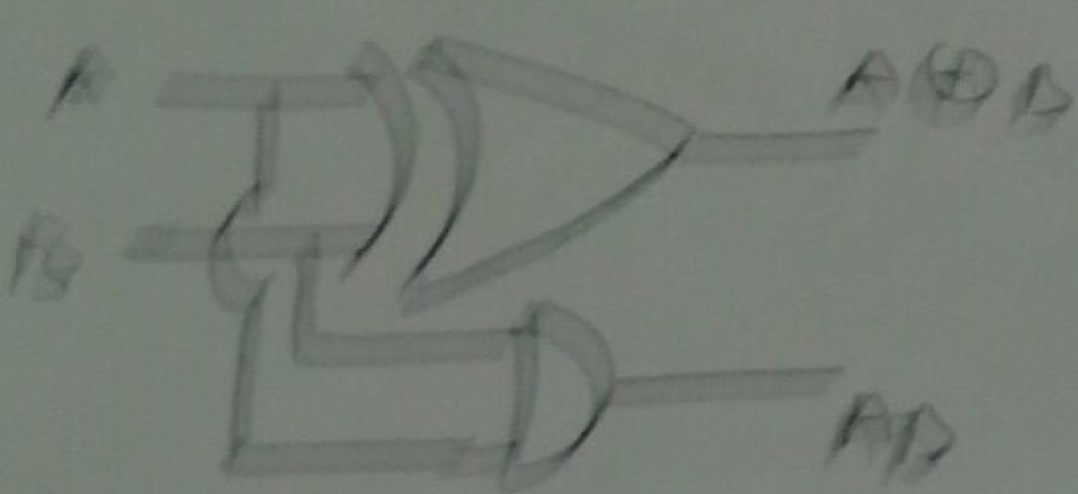


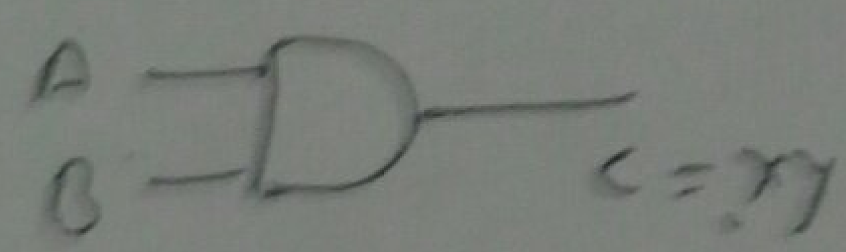
$C = AB$





12



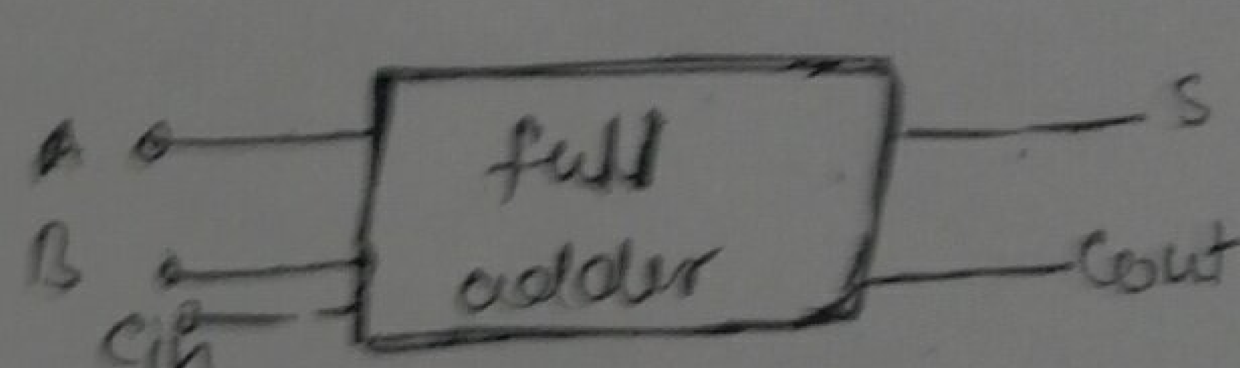


FULL ADDER (A combinational ckt that perform addition of three bits (two significant bits and a previous carry) is a full adder. (It consist of three i/p & two o/p)

A	B	C_{in}	C_{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

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

$C_{out} = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$



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

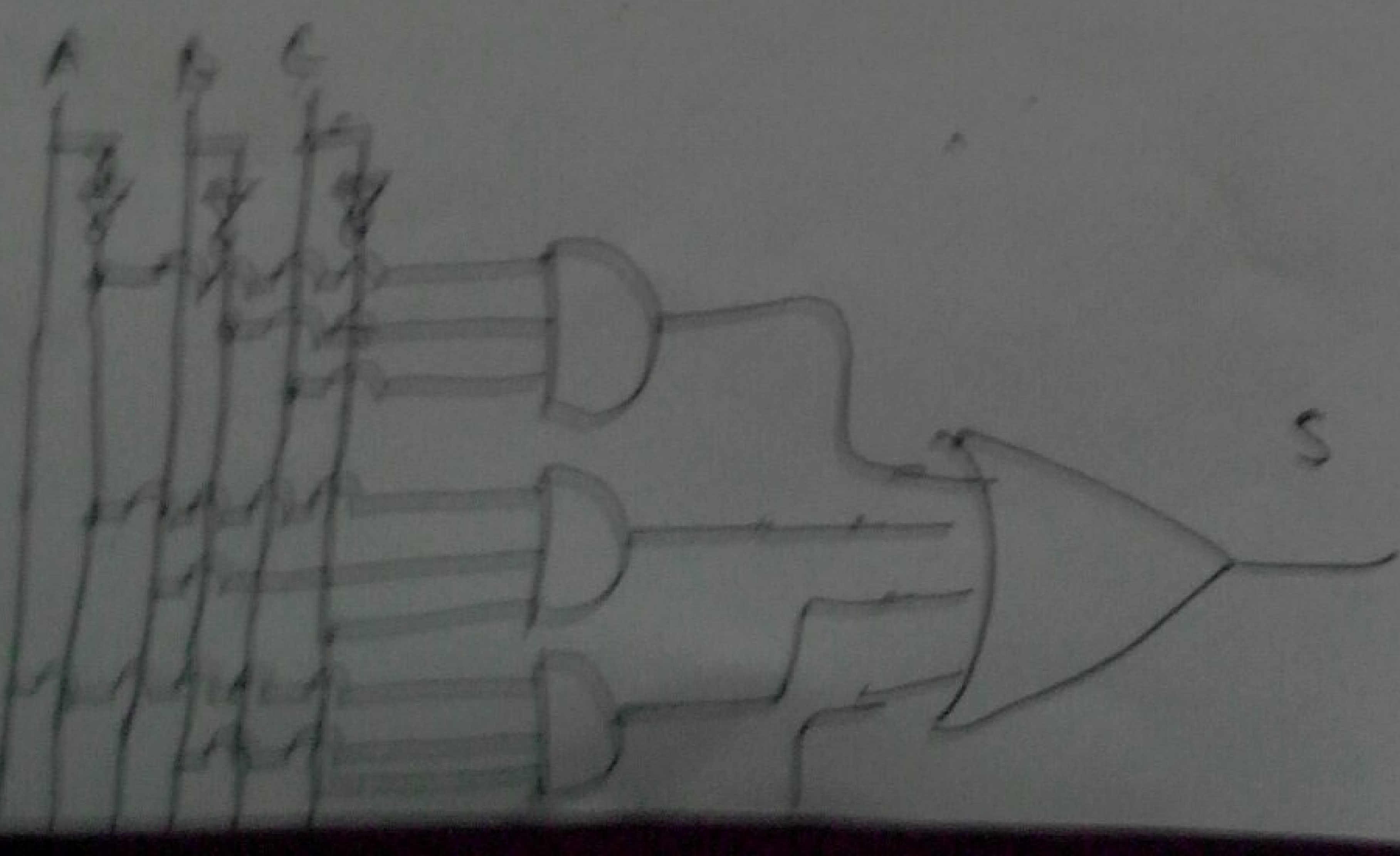
$C_{out} = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$

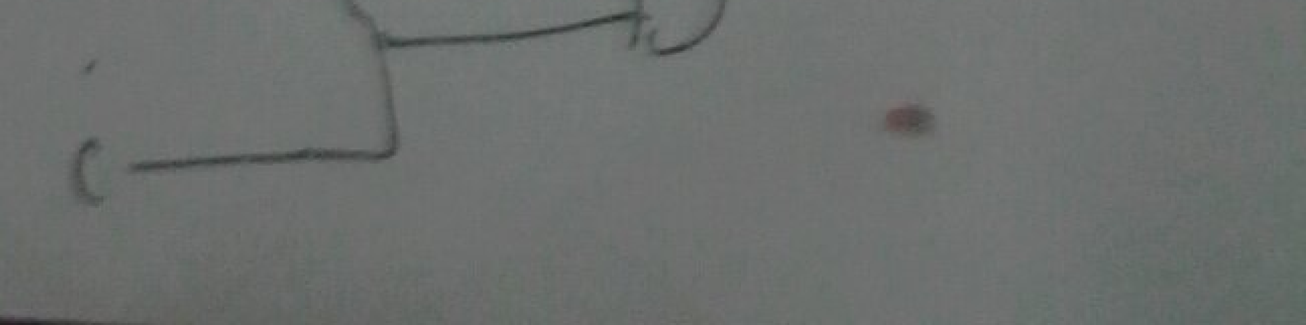
$= \cancel{BC(\bar{A}+A) + A(\bar{B}C + B\bar{C})}$

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

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

$C_{out} = AC + BC + AB$





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

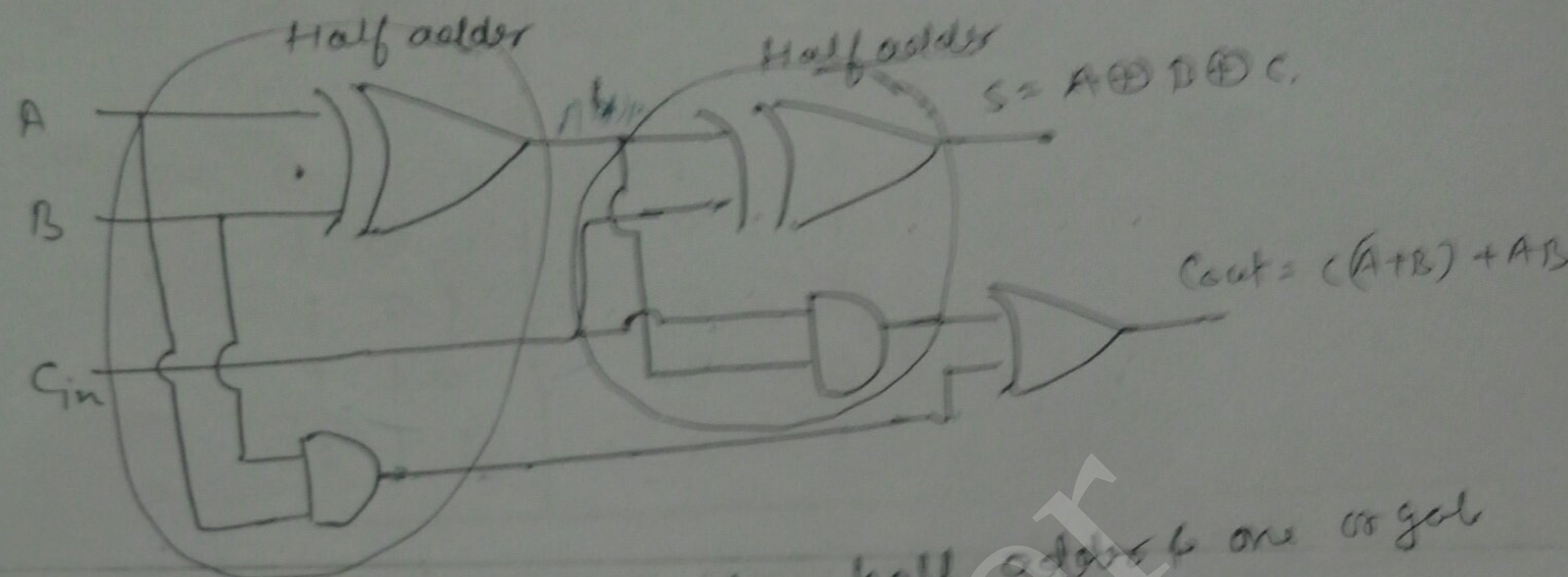
$= \bar{A}(\bar{B}C + B\bar{C}) + A(\bar{B}\bar{C} + BC)$

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

$S = \underline{A \oplus B \oplus C}$

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

$C = AC + BC + AB$

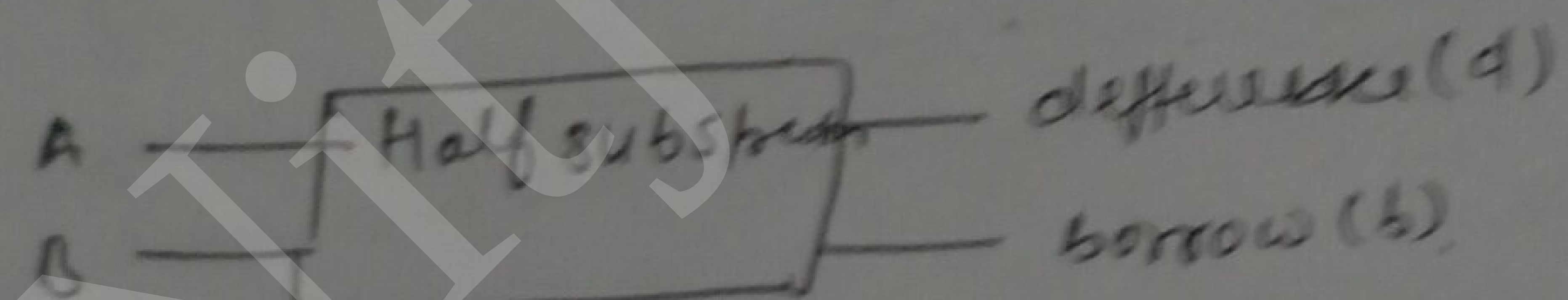


full adder using two half adder & one or gate

The HALF SUBTRACTORS :-

Combinational ckt that subtract one bit from the other and produces the difference. & also has an output to specify if a 1 has borrowed.

minuend
- substrahend

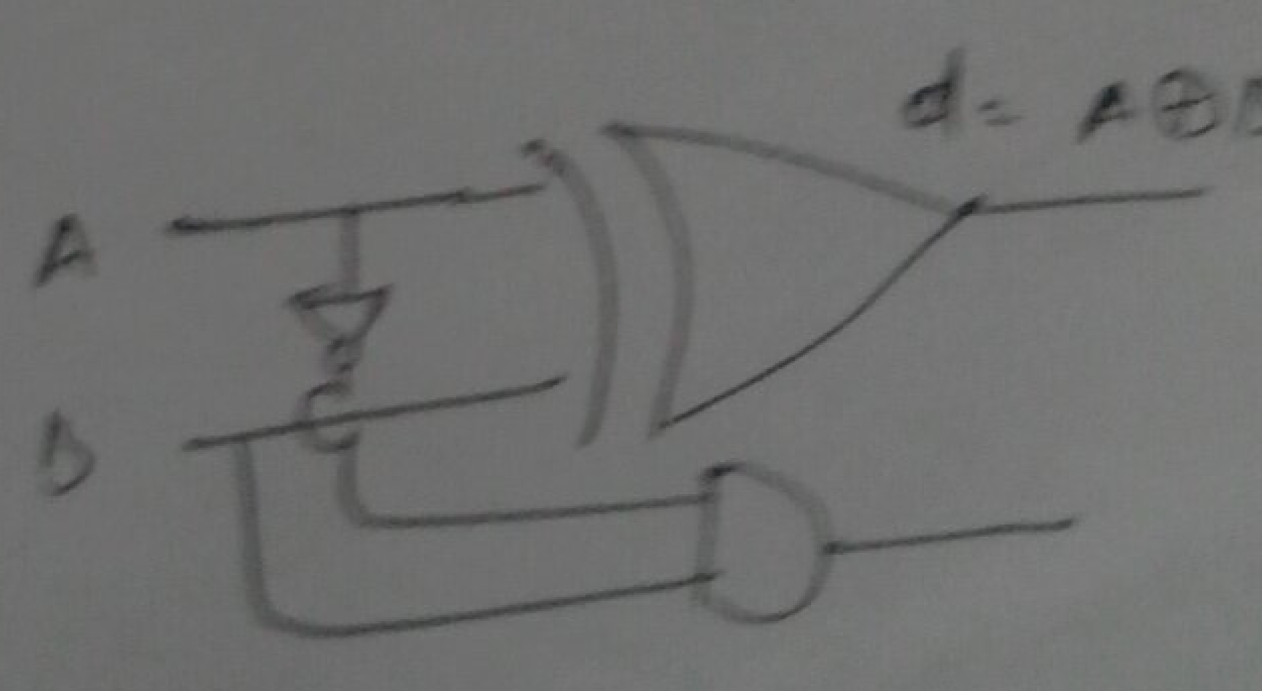


A	B	b	d
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

2 - 1
10 - 1

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

$b = \bar{A}B$



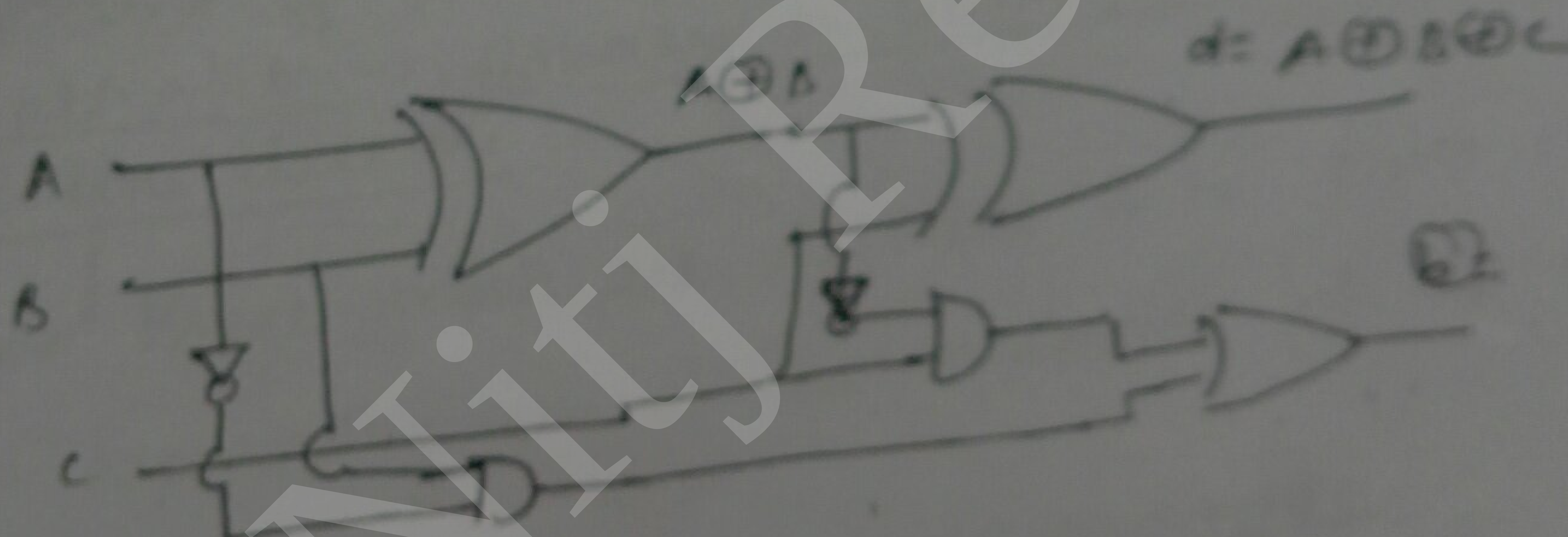
Full substructor

14

A	B	C	b	d
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$d = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC = \boxed{A \oplus B \oplus C}$

$b = \bar{A}\bar{B}C + \bar{A}B\bar{C} + \bar{A}BC + ABC = \boxed{\bar{A}B + (\overline{A \oplus B})C}$



A binary code of <u>n-bits</u> is capable of representing up to 2^n distinct elements of coded information

15

Such as- n=2.

0	0
0	1
1	0
1	1

$\Rightarrow 2^2 = 4$

n=3. $2^n = 8$

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

= 8

A decoder is combinational ckts that convert binary information from <u>n-input lines</u> to a <u>maximum of 2^n output line</u>

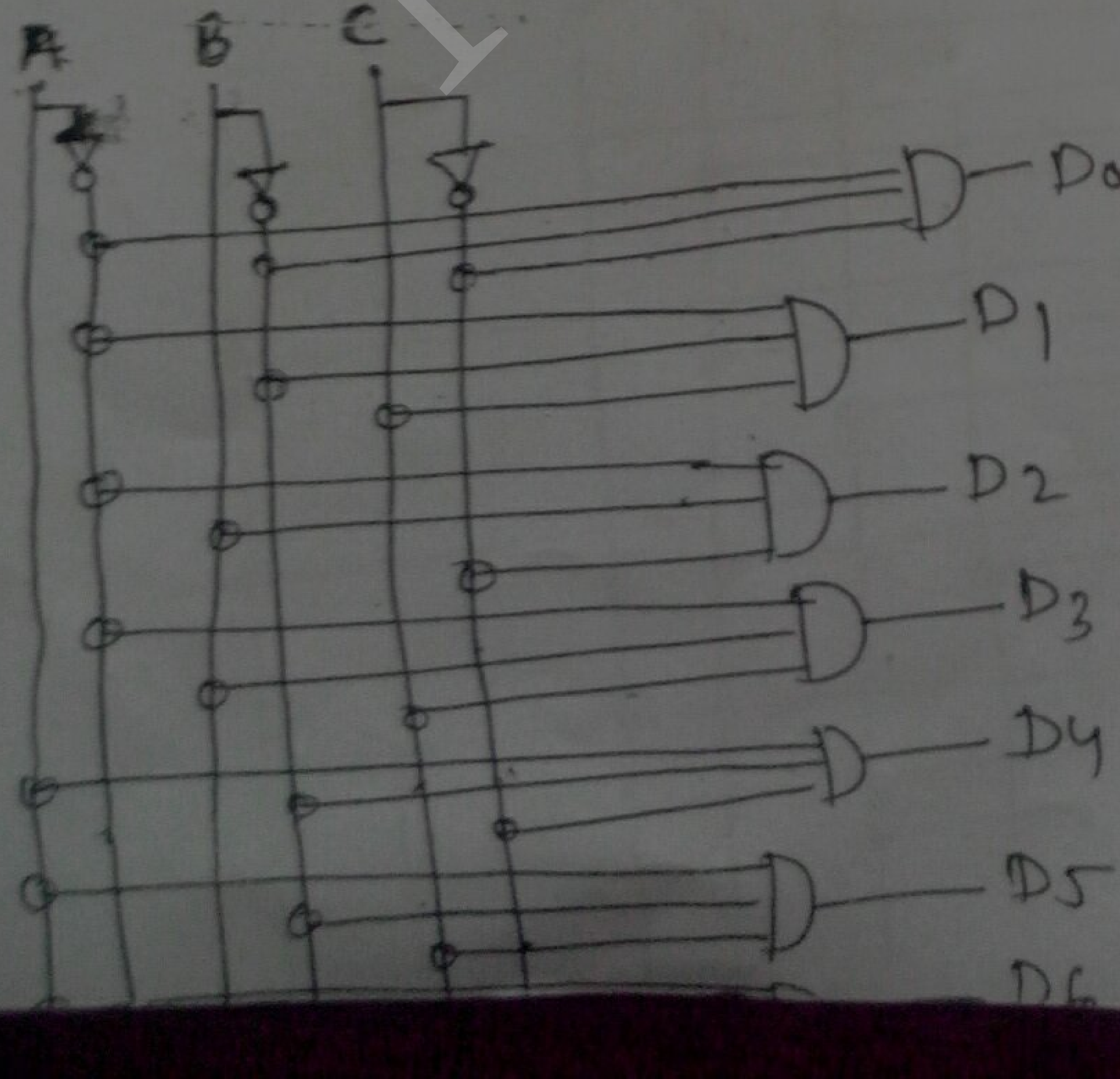
The decoder presented here are called n to m line decoders, where $m \leq 2^n$. Their purpose is to generate the 2^n (or fewer) minterms of n input variables.

<u>Application</u> → <u>Binary to octal converter</u>.

<u>3 to 8 line decoder</u> :-

The three inputs are decoded into eight outputs, each each representing one of the minterms of the three input variables.

~~Application ⇒ Binary to octal conversion.~~



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

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

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

$D_3 = \bar{A}BC$

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

$D_5 = A\bar{B}C$

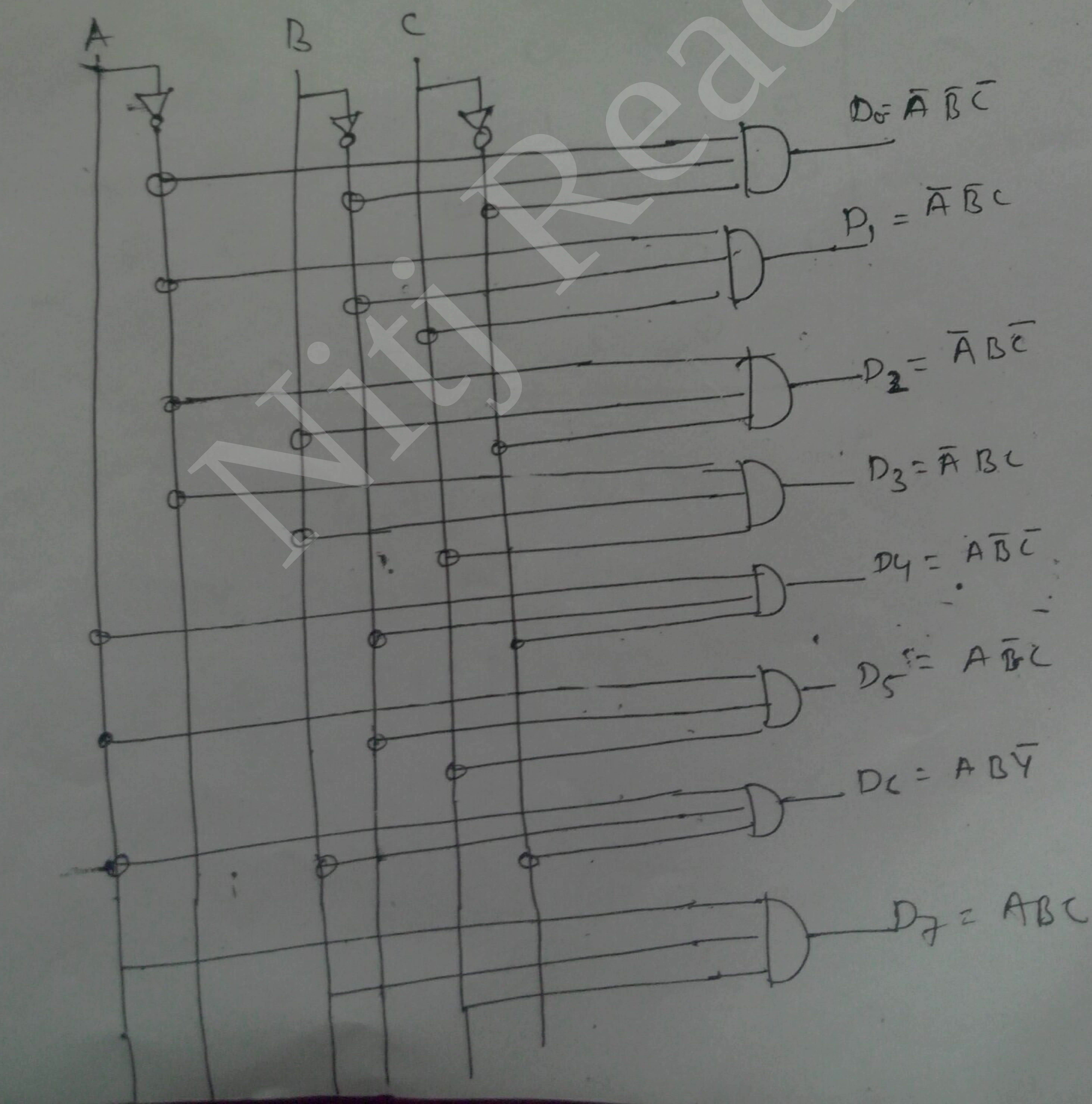
$D_6 = AB\bar{C}$

$D_7 = ABC$

16

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



ENCODERS

An Encoder is a digital circuit that performs the inverse operation of decoder.

17

An Encoder has 2^n (or fewer) input lines and n output lines. The output lines generate the binary code corresponding to the input value. An example of an Encoder is octal to binary Encoder. It has eight i/p's (one for each of the octal digits) and three outputs that generate the corresponding binary no.

It is assumed that only one input has a value of 1 at a given time.

INPUTS								OUTPUTS		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_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

$$A = D_4 + D_5 + D_6 + D_7$$

$$B = D_2 + D_3 + D_6 + D_7$$

$$C = D_1 + D_3 + D_5 + D_7$$

MULTIPLEXERS: (DATA SELECTORS)

A multiplexer is a combinational circuit that selects binary information from one of many inputs lines and directs it to single output line.

Normally there are 2^n inputs lines and n selection li[nes] whose bit combinations determine which input is selected.

2 : 1 MUX

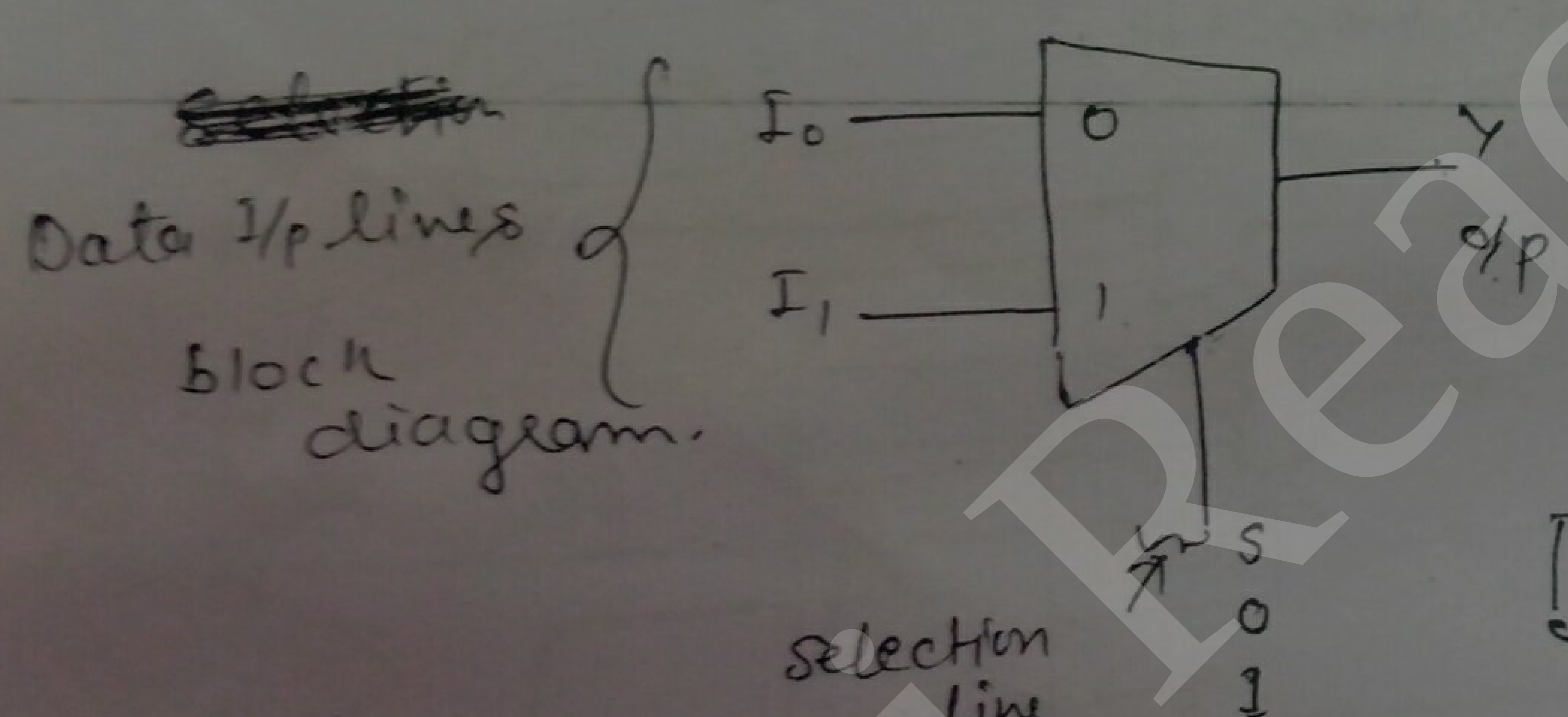
(18)

if $n = 1$

I/p lines $2^1 = 2$
selection line $= n = 1$ → This is called 2 : 1 MUX

Function table

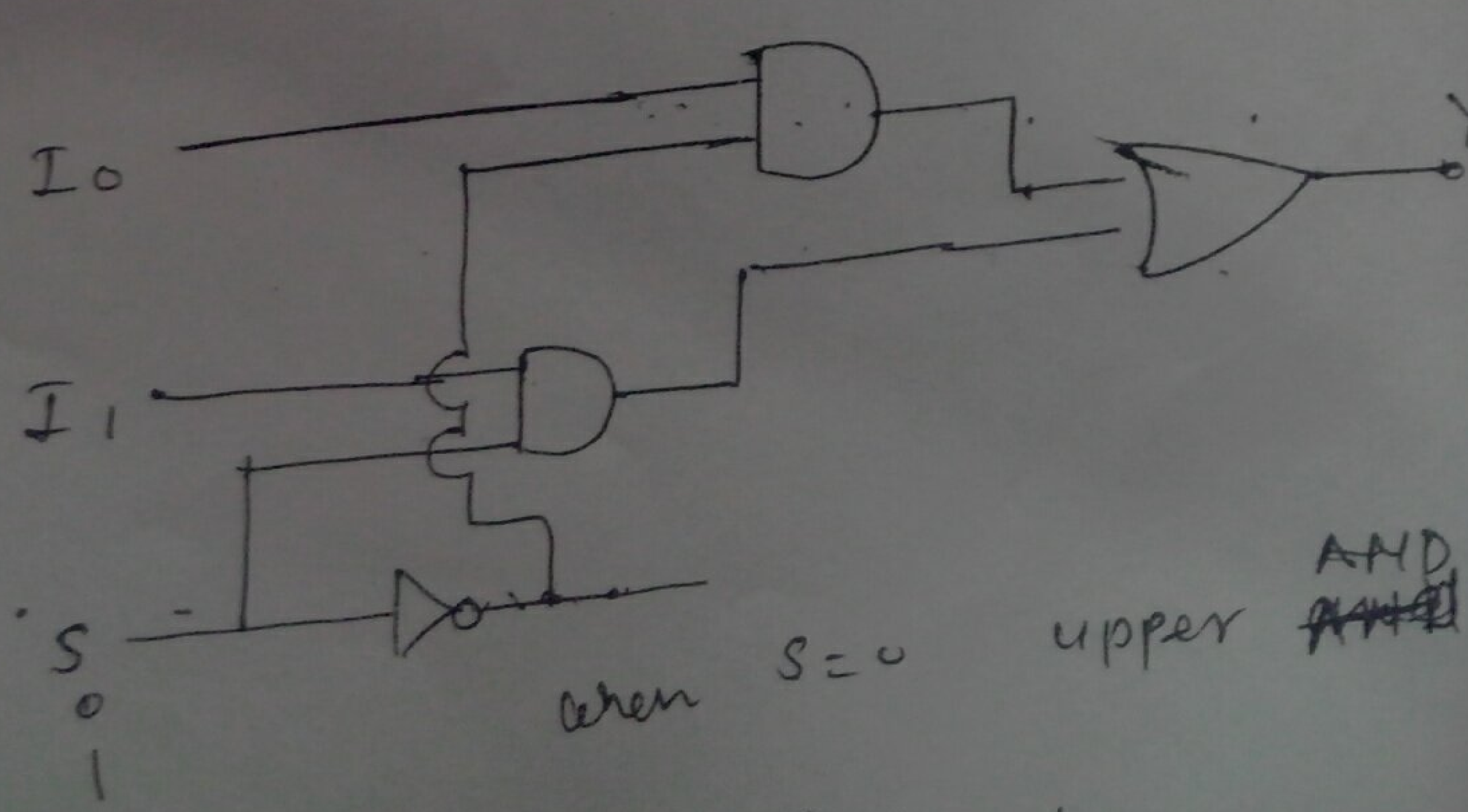
S	Y
0	I_0
1	I_1



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

when $S = 0$, I_0 has a path to output
when $S = 1$, I_1 has a path to output.

The mux act like an electronic switch that selects one of the two sources.



when $S = 0$ upper ~~AND~~ AND gate is enab[led]

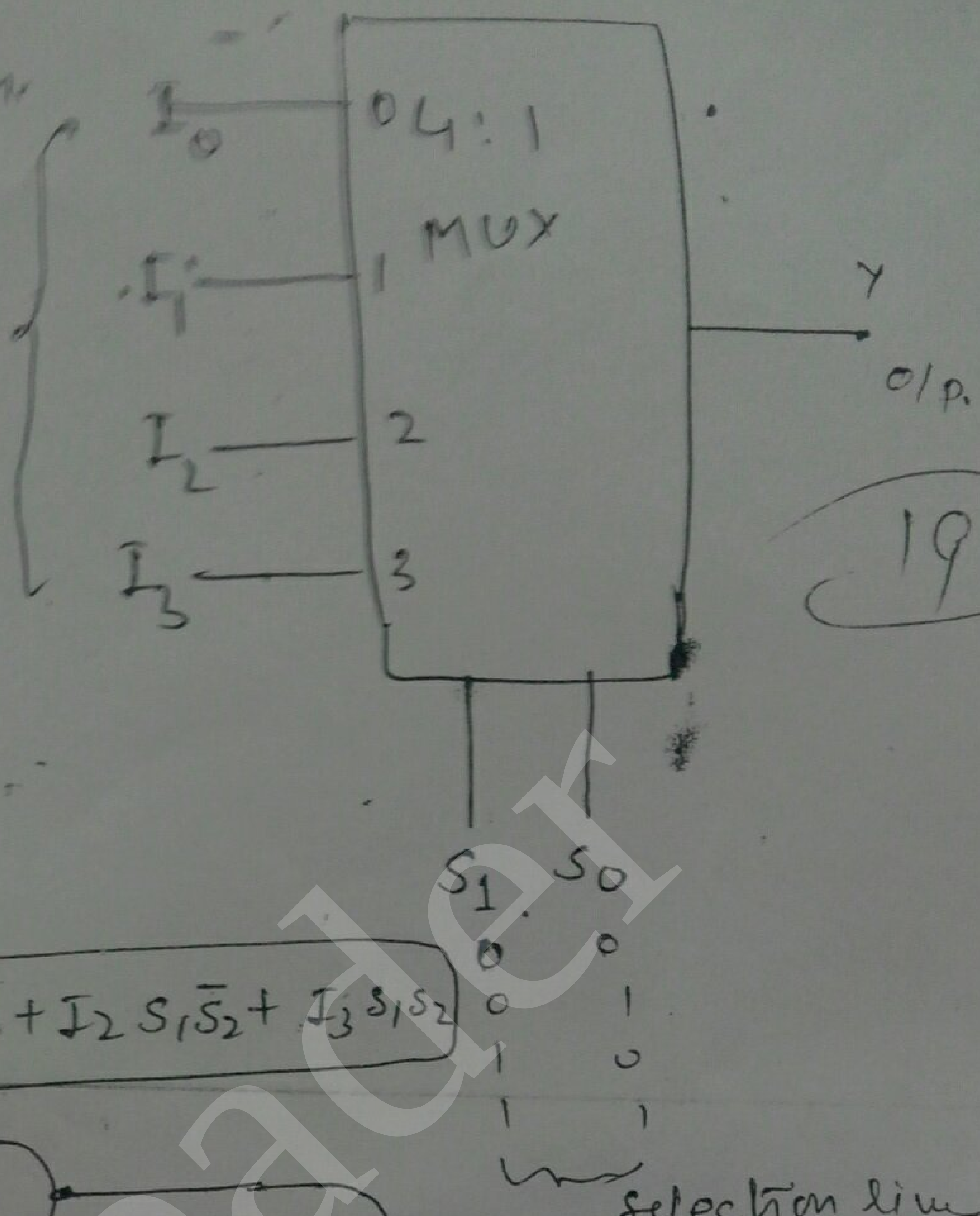
$S = 1$ lower AND gat[e] ...

4 : 1 MUX

n = 2

I/p lines : $2^n = 2^2 = 4$

~~selecte~~ selection line = 2 = n

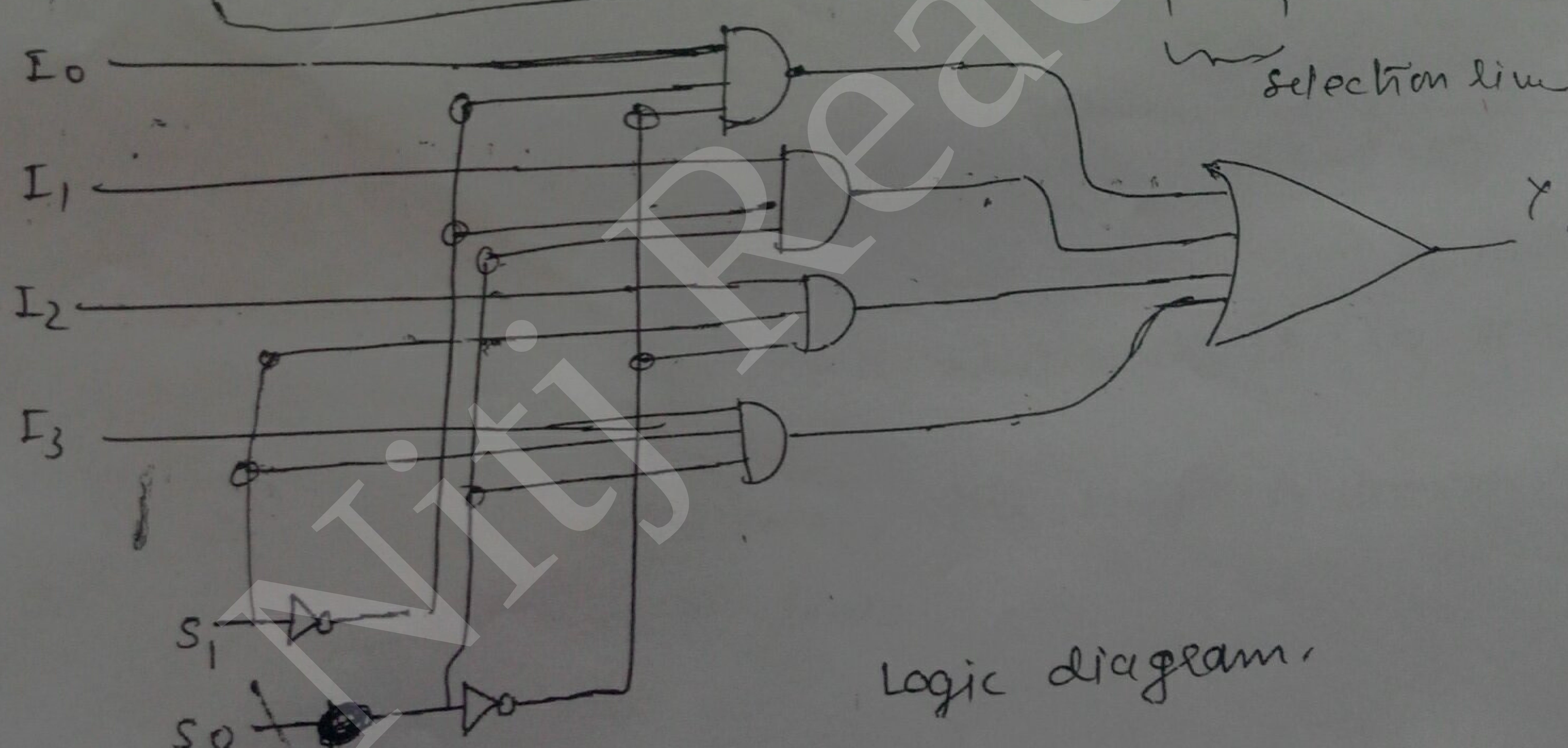


19

Function table.

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

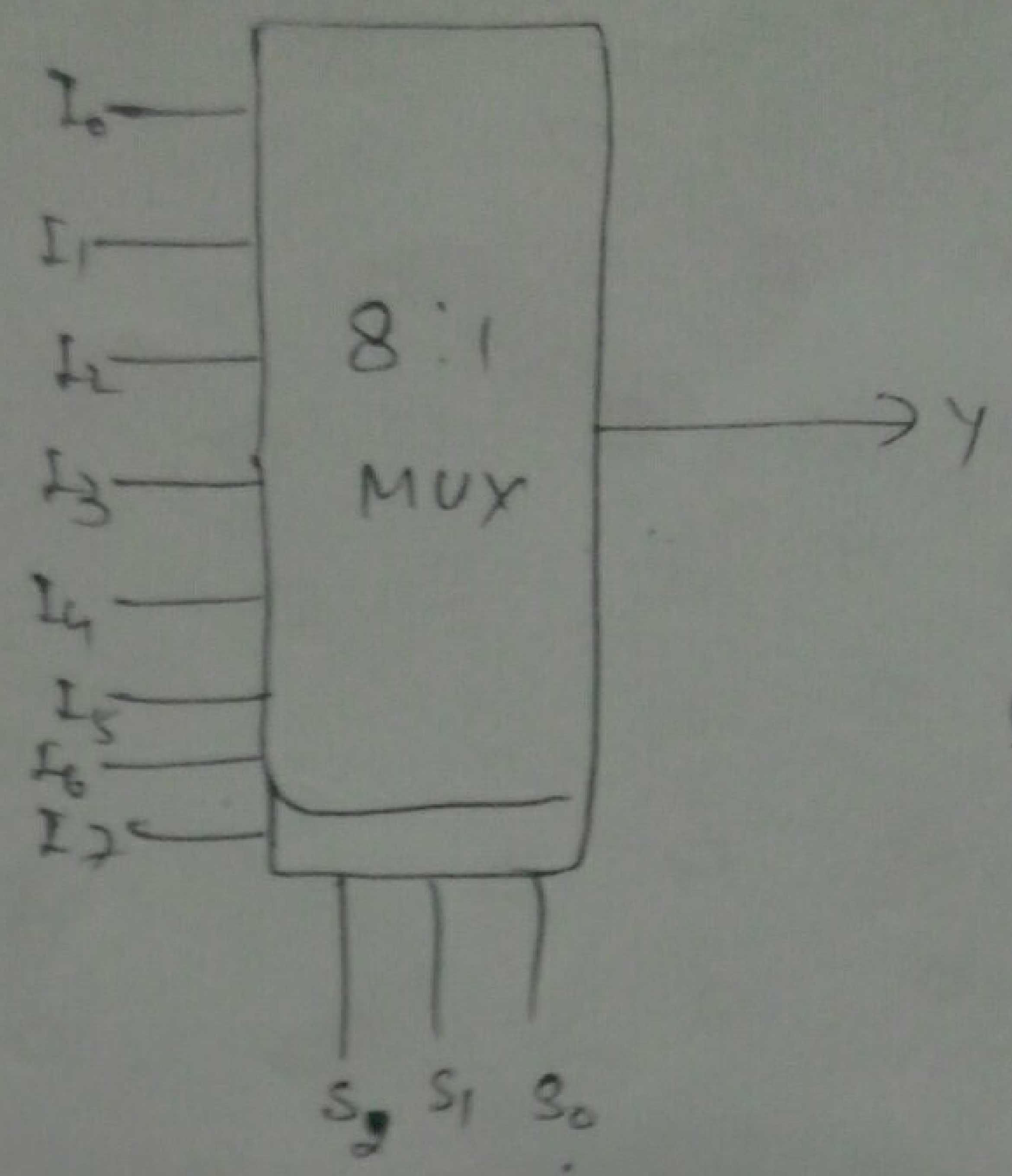
$$\boxed{Y = I_0 \bar{S}_1 \bar{S}_0 + I_1 \bar{S}_1 S_0 + I_2 S_1 \bar{S}_2 + I_3 S_1 S_2}$$



Logic diagram.

(8:1 MUX)

S_2	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	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7



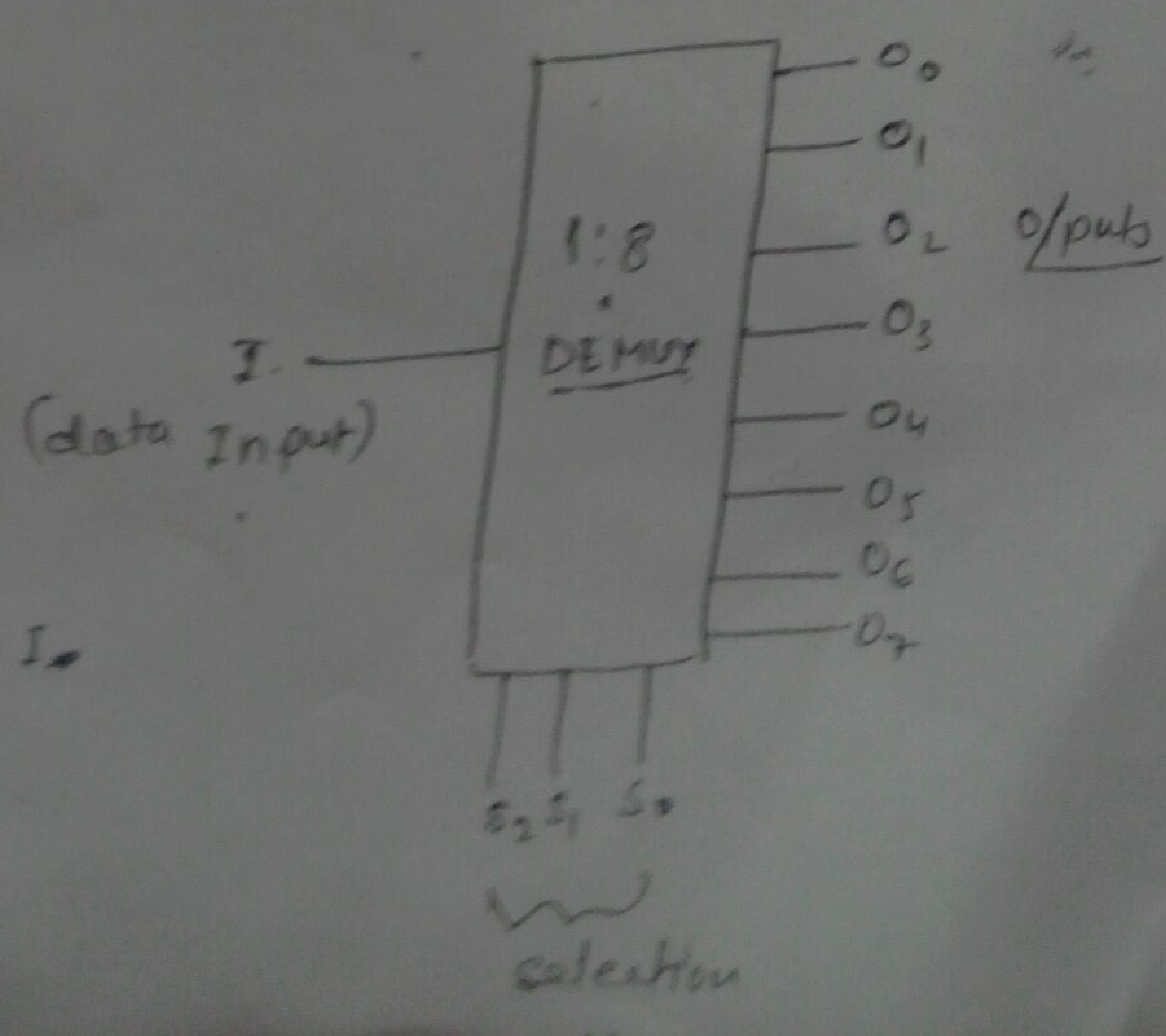
20

$$Y = I_0 \bar{s}_2 \bar{s}_1 \bar{s}_0 + I_1 \bar{s}_2 \bar{s}_1 s_0 + I_2 \bar{s}_2 s_1 \bar{s}_0 + I_3 \bar{s}_2 s_1 s_0 + I_4 s_2 \bar{s}_1 \bar{s}_0 + I_5 s_2 \bar{s}_1 s_0 + I_6 s_2 s_1 \bar{s}_0 + I_7 s_2 s_1 s_0$$

DEMULTIPLEXER (DATA DISTRIBUTORS)

A demultiplexer performs the reverse operation of multiplexer. It takes a single input and distributes it to several outputs.
The select input determines, the input is transmitted to which of the output lines.

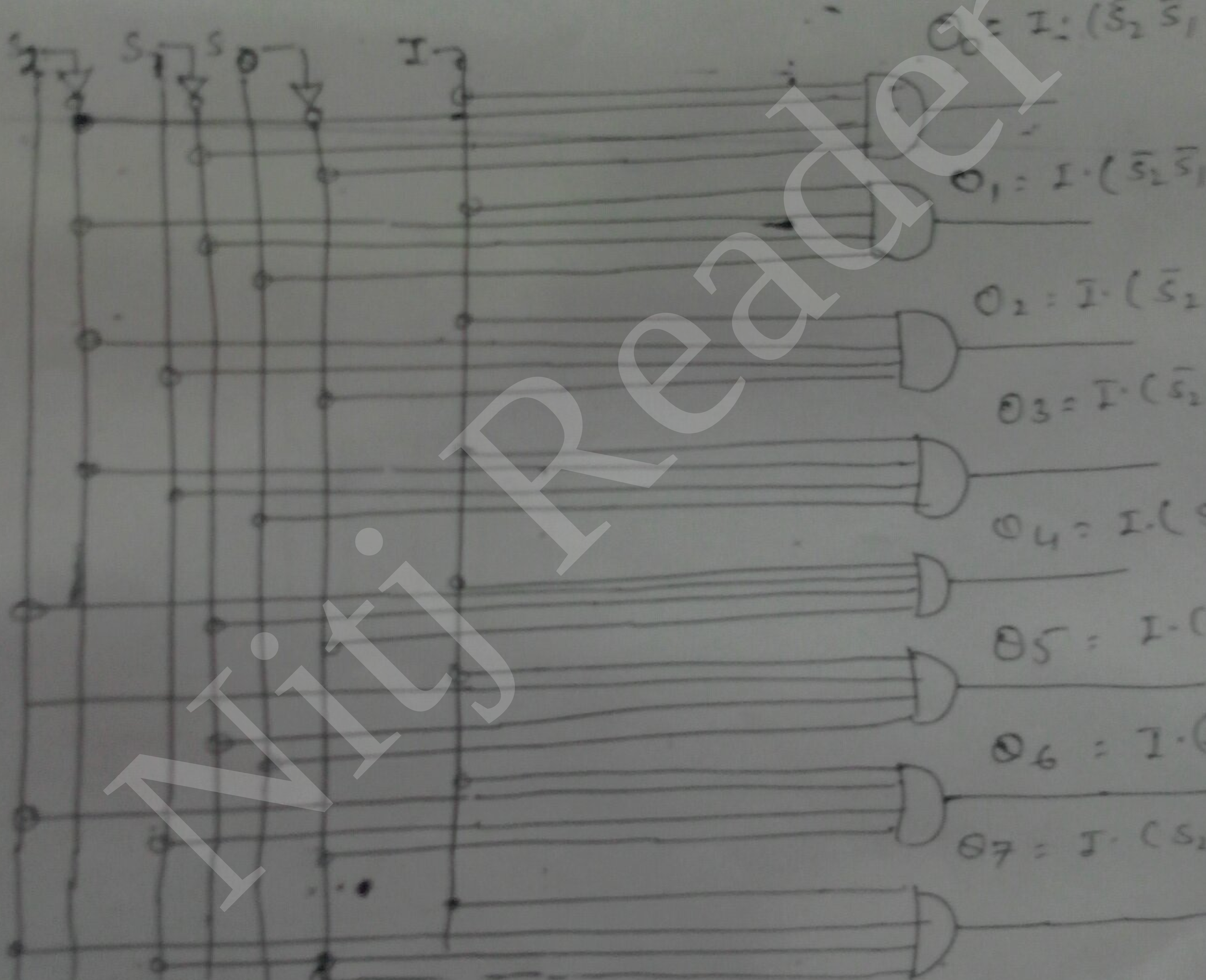
1-8 DEMUX



if $S_2 S_1 S_0 = 0\,0\,0$

the output available at $O_0 = I$

S_2	S_1	S_0	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	I
0	0	1	0	0	0	0	0	0	I	0
0	1	0	0	0	0	0	0	I	0	0
0	1	1	0	0	0	0	I	0	0	0
1	0	0	0	0	0	I	0	0	0	0
1	0	1	0	0	I	0	0	0	0	0
1	1	0	0	I	0	0	0	0	0	0
1	1	1	I	0	0	0	0	0	0	0



$O_0 = I \cdot (\bar{S}_2 \bar{S}_1 \bar{S}_0)$

$O_1 = I \cdot (\bar{S}_2 \bar{S}_1 S_0)$

$O_2 = I \cdot (\bar{S}_2 S_1 \bar{S}_0)$

$O_3 = I \cdot (\bar{S}_2 S_1 S_0)$

$O_4 = I \cdot (S_2 \bar{S}_1 \bar{S}_0)$

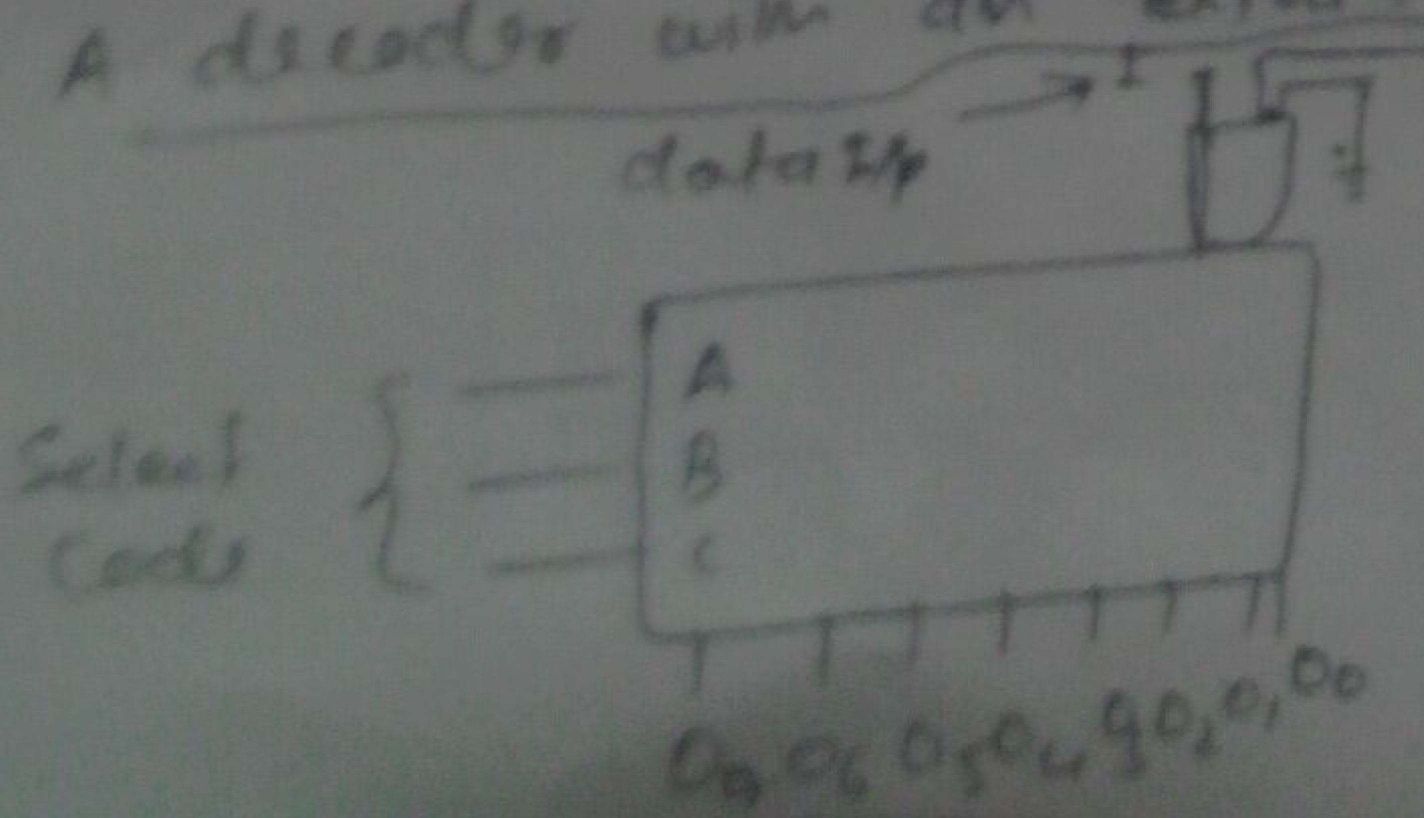
$O_5 = I \cdot (S_2 \bar{S}_1 S_0)$

$O_6 = I \cdot (S_2 S_1 \bar{S}_0)$

$O_7 = I \cdot (S_2 S_1 S_0)$

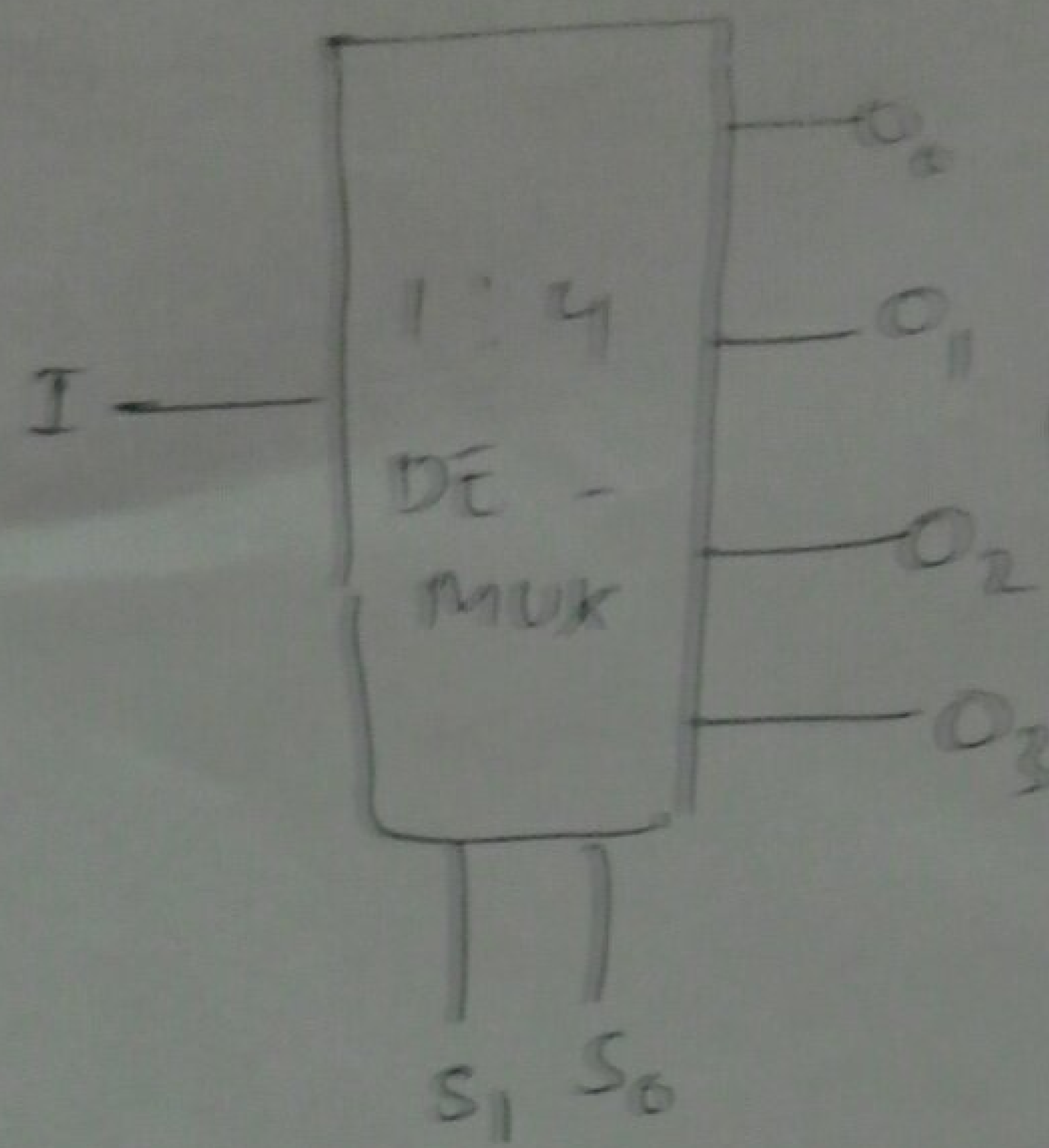
The demux shown above is very similar to the 3:8 decoder except that fourth Input I has been added to each gate.

i.e. A decoder with an extra (enable) input act as demultiplexer.

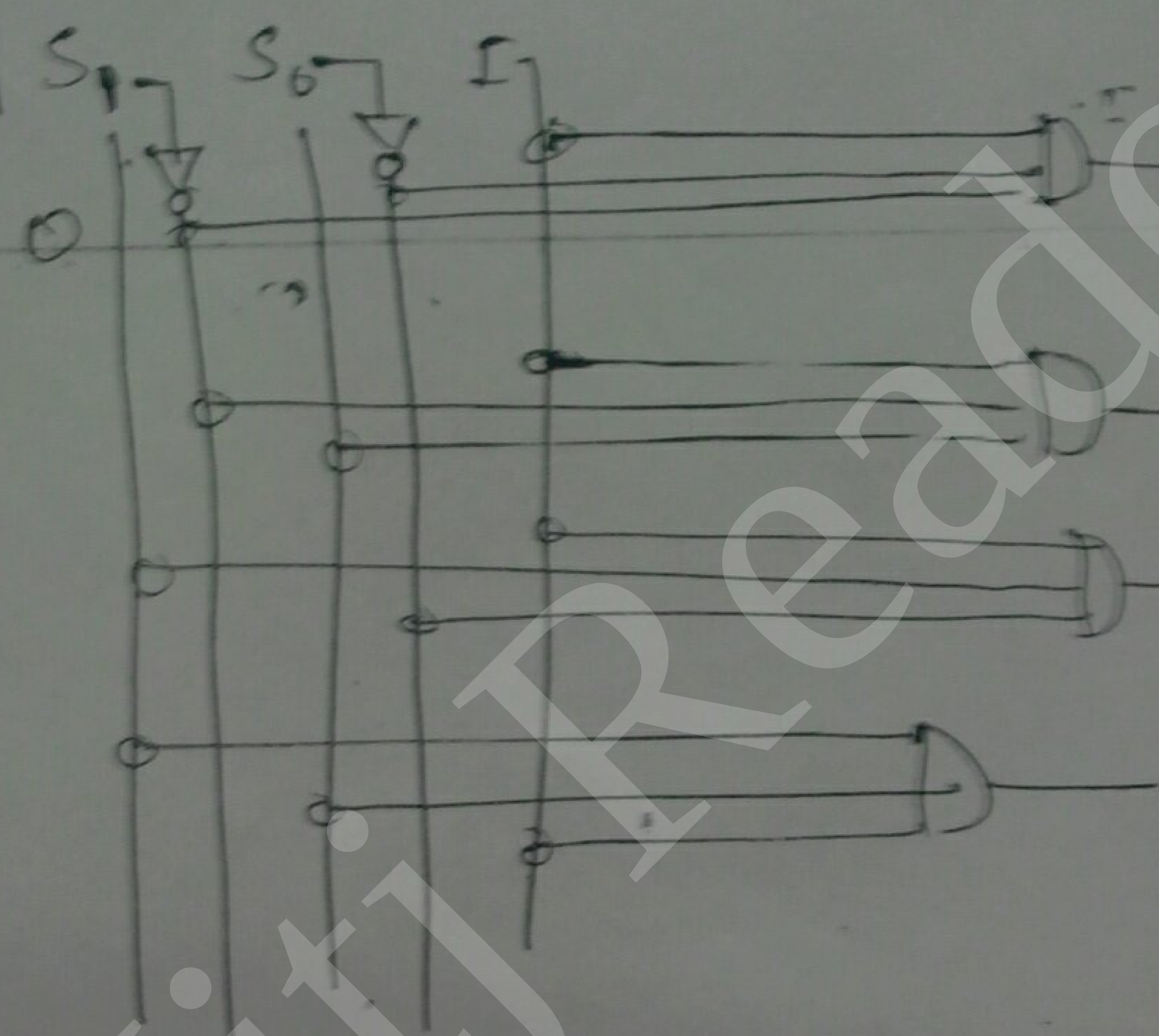


1:4 DEMUX

S_1	S_0	O_3	O_2	O_1	O_0
0	0	0	0	0	I
0	1	0	0	I	0
1	0	0	I	0	0
1	1	I	0	0	0



82



$O_0 = I \cdot (\bar{S_1} \bar{S_0})$

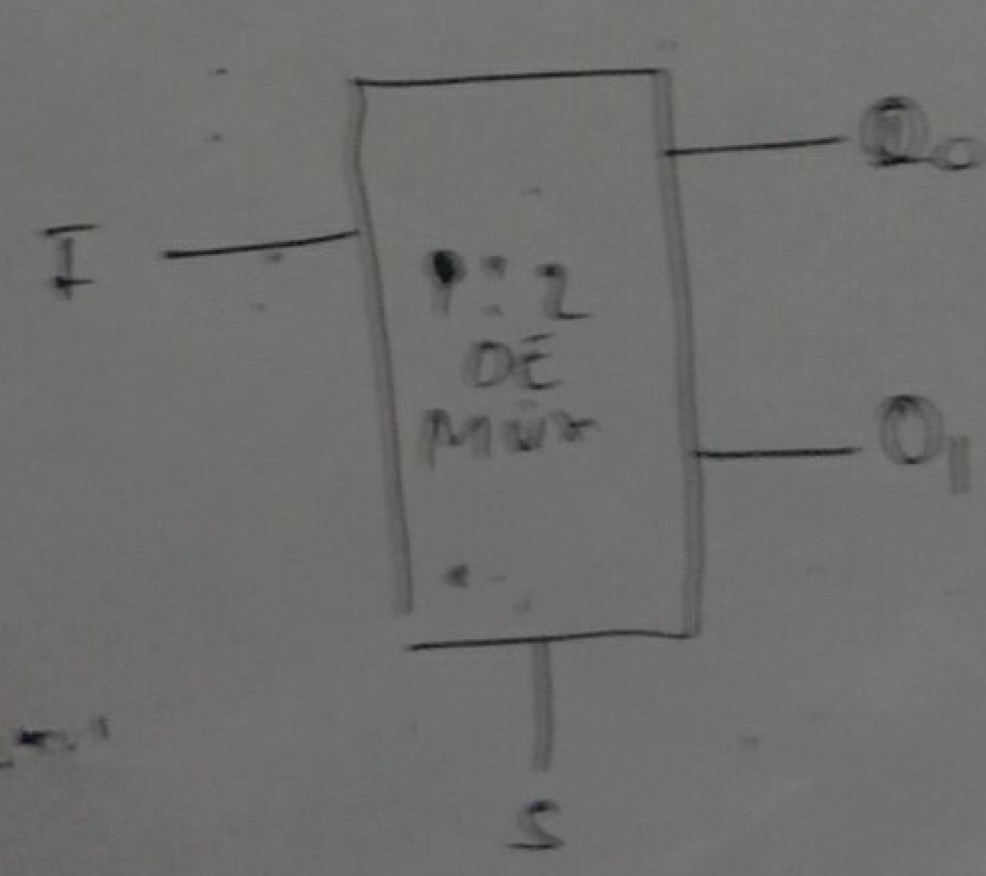
$O_1 = I \cdot (\bar{S_1} \cdot S_0)$

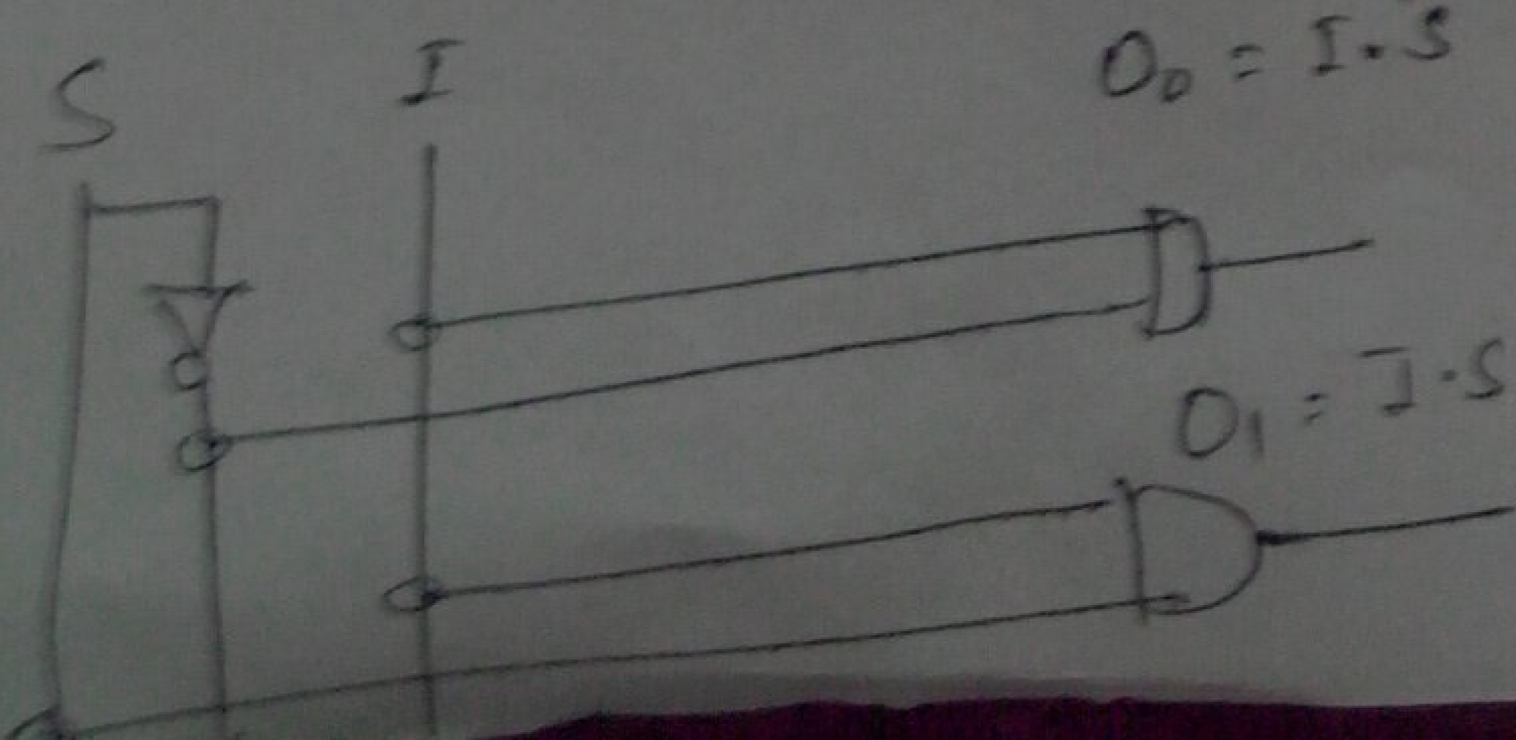
$O_2 = I (S_1 \cdot \bar{S_0})$

$O_3 = I (S_1 S_0)$

1:2 DEMUX:

S	O_1	O_0
0	0	I
1	I	0





$O_0 = I \cdot \bar{S}$

$O_1 = I \cdot S$