Carry Look Ahead Adder : (16m) (1) Q.B

In parallel adder the carry output of each full adder stage is connected to the carry input of the next higher order stage. Therefore the sum and the carry outputs of any stage cannot be produce until the ~~carry~~ input carry occurs. This leads to a time delay in the addition process. This delay is called carry propagation delay.

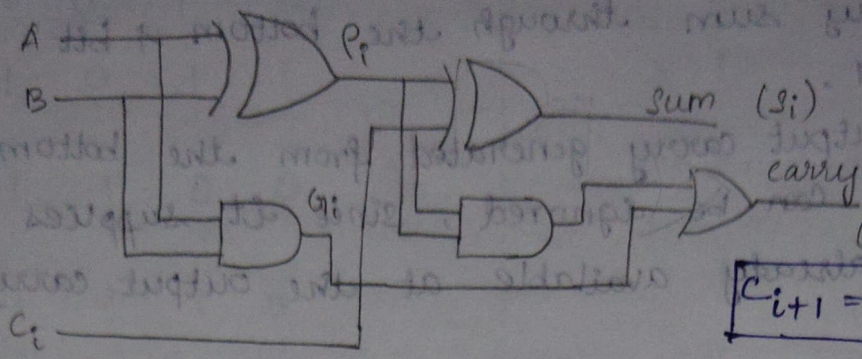
(eg):

```
  ① ① ①
  0 1 0 1
  0 0 1 1
---------
  1 0 0 0
---------
```

Addition LSB (Least significant Bit) produces carry into second and then second to third and so on. from this the sum bit generated in the last position (MSB) Most significant Bit. Depends on the carry that was generated by the addition of the previous position.

This means adder will not produce correct result until LSB has carry propagated through intermediate full adders. One method of speeding up this process by eliminating interstage carry delay is called Look ahead carry addition.

(eg): Implement full adder using 2 half adder :



$$C_{i+1} = G_i + P_i C_i$$

$$P_i = A_i \oplus B_i$$

$$G_i = A_i B_i$$

If $i=1$, $\boxed{C_2 = G_1 + P_1 C_1}$

$i=2$, $C_3 = G_2 + P_2 C_2$

$= G_2 + P_2 (G_1 + P_1 C_1)$

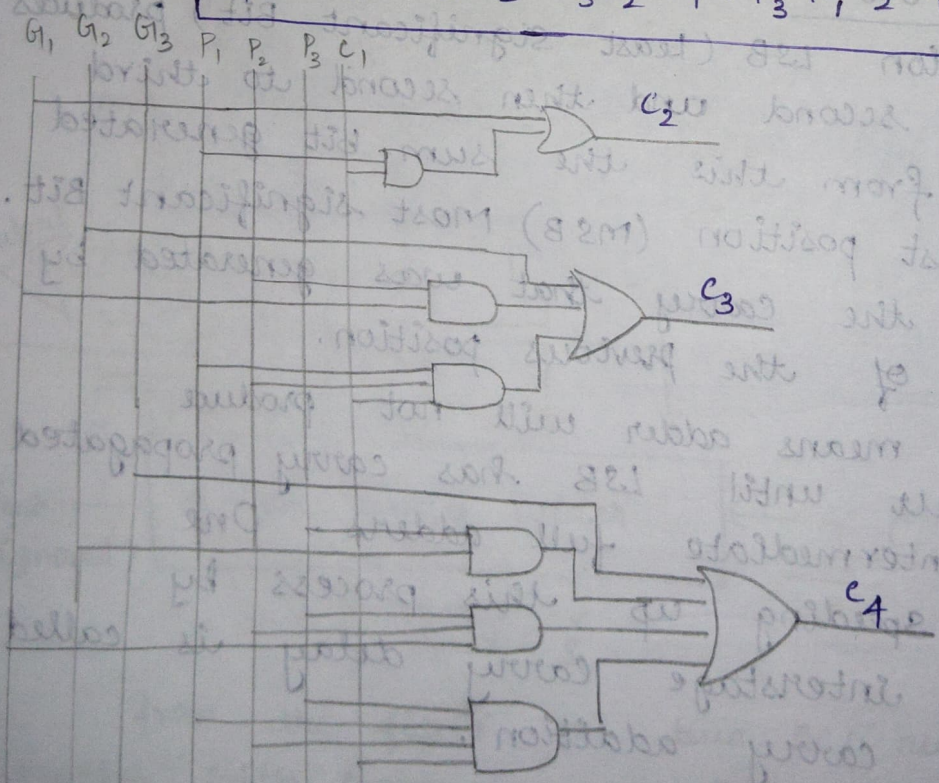
$$\boxed{C_3 = G_2 + P_2 G_1 + P_1 P_2 C_1}$$

$i=3$,

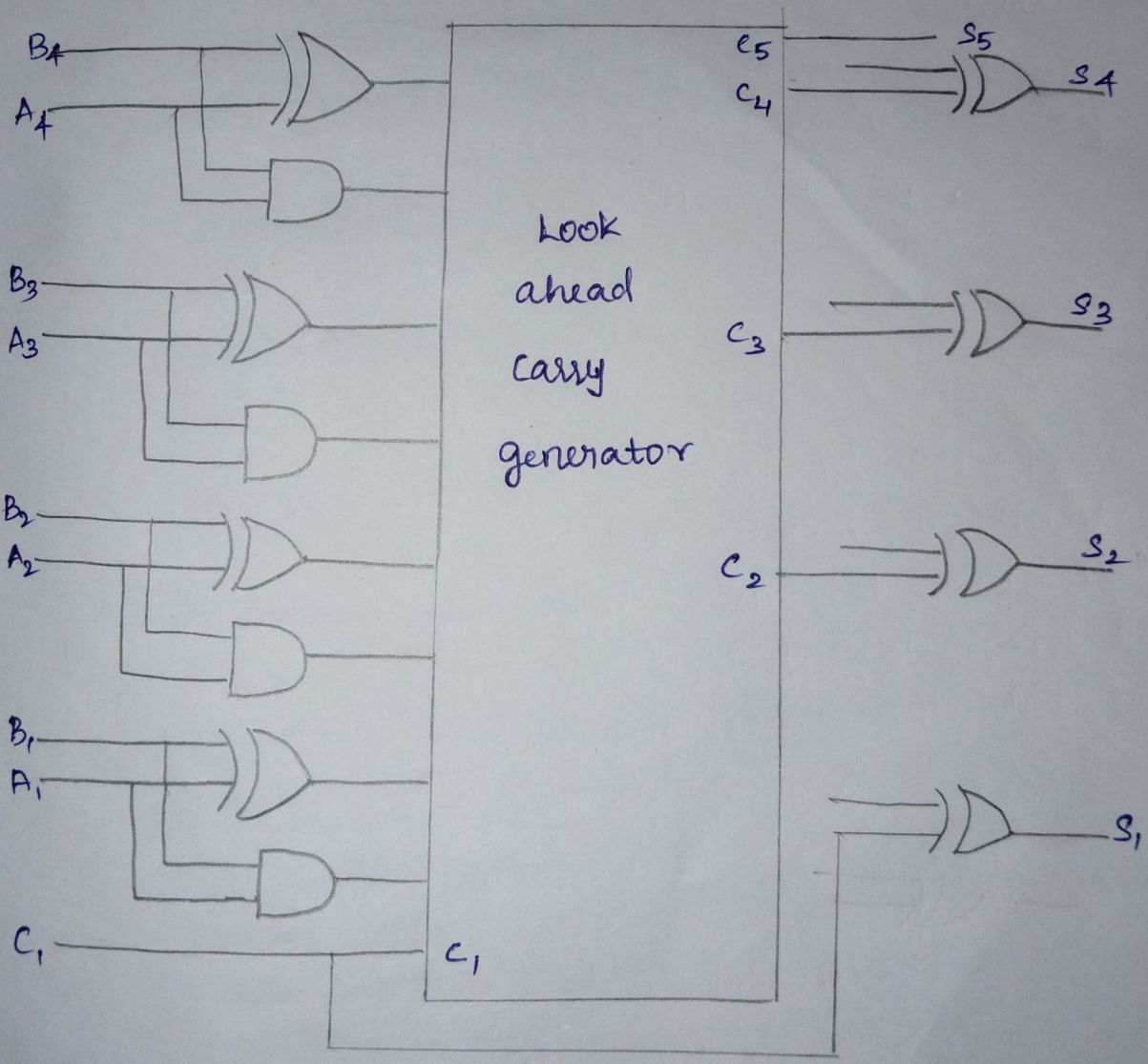
$C_4 = G_3 + P_3 C_3$

$= G_3 + P_3 (G_2 + P_2 G_1 + P_1 P_2 C_1)$

$$\boxed{C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_1 P_2 C_1}$$



Using a carry look ahead generator we can easily construct 4-bit parallel adder with a look ahead carry scheme. In this each sum output requires two ex-OR gate. The output of first EX-OR gate generates P_i and the AND gate generates G_i. The carry that of generator using look ahead carry generator and applied as inputs to the second EX-OR gate and the input to the EX-OR gate is P_i. Thus second EX-OR gate genetrates the sum output and each output generated after a delay of two levels of gate.

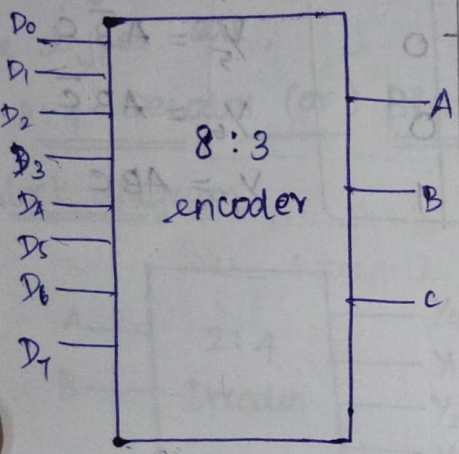


ENCODER: (2) Q.B 8m

An encoder is a digital circuit that performs the inverse operation of a decoder, that is input will be more number when compare to that of output. It is in the order of $2^n : n$

1) 8:3 Encoder (Or) Octal to Binary encoder:

Block diagram



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

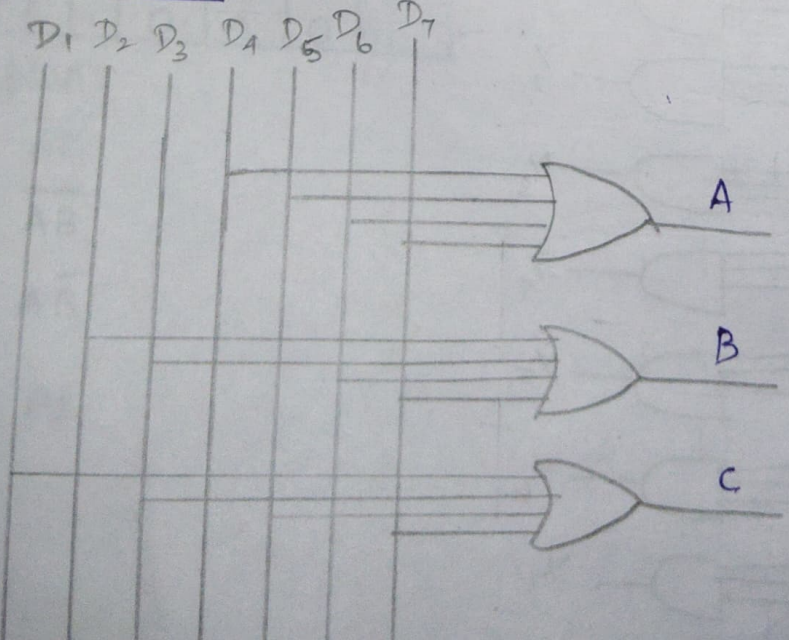
Equation:

$A = D_4 + D_5 + D_6 + D_7$

$B = D_2 + D_3 + D_6 + D_7$

$C = D_1 + D_3 + D_5 + D_7$

Logic circuit



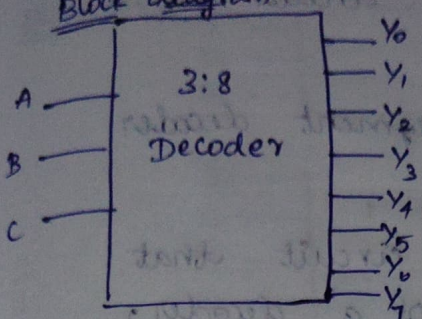
08/08/2024.

Decorder: (8m) 2. Q.B

A decorder is a multiple input and multiple output logic circuit which converts coded inputs into coded outputs were input and output codes are different. Input code generally has fewer codes than output codes.

3:8 Decoder

Block diagram



Truth table

A	B	C	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_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

Equation

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

$Y_1 = \bar{A}\bar{B}C$

$Y_2 = \bar{A}B\bar{C}$

$Y_3 = \bar{A}BC$

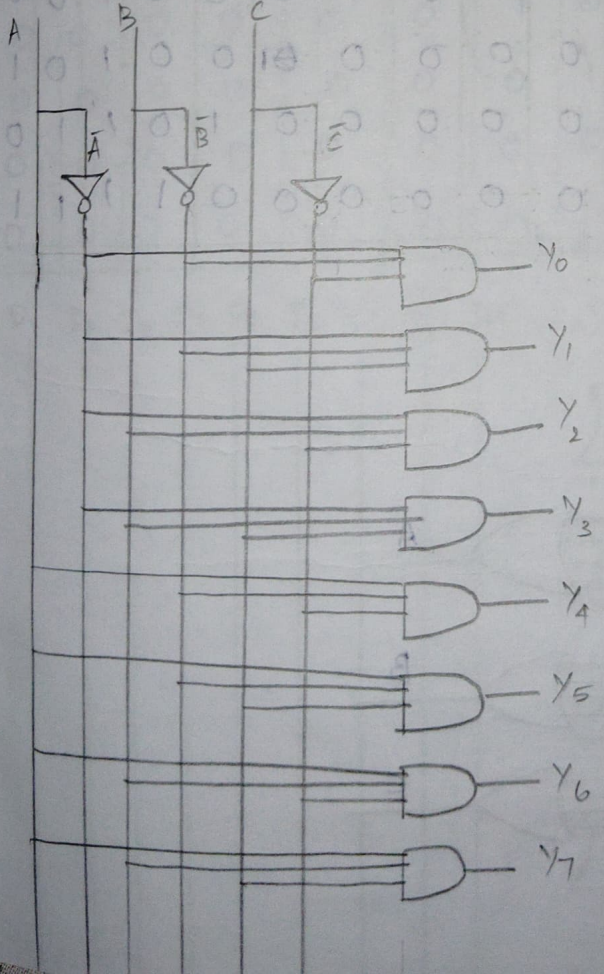
$Y_4 = A\bar{B}\bar{C}$

$Y_5 = A\bar{B}C$

$Y_6 = AB\bar{C}$

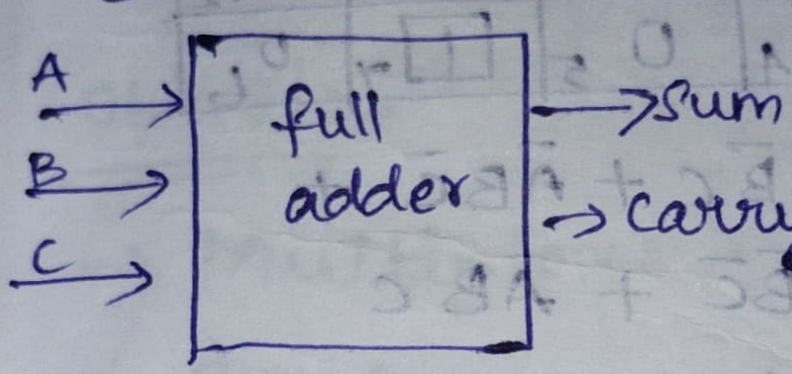
$Y_7 = ABC$

Logic circuit



b) **full adder** : (10m)

Block diagram (3) Q.P



Truth Table

A	B	C	S	C
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

equation:

K-map

Sum

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

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

carry

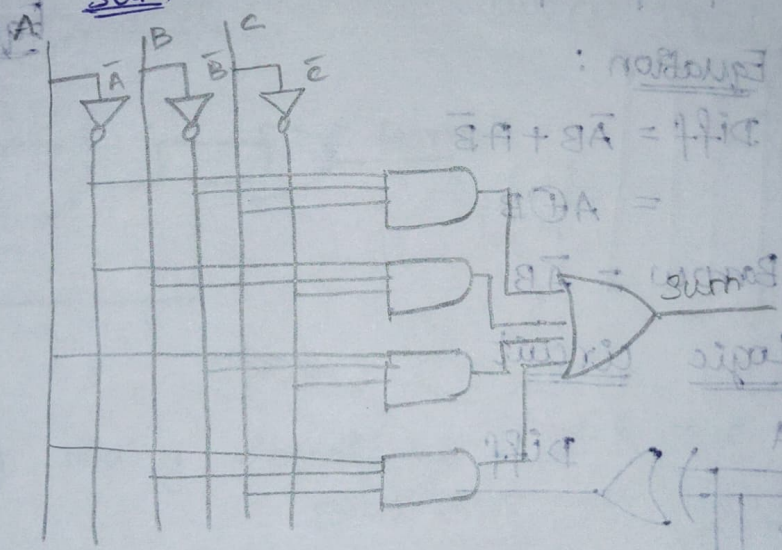
A \ BC	00	01	11	10
0	0 (0)	0 (1)	1 (3)	0 (2)
1	0 (4)	1 (5)	1 (7)	1 (6)

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

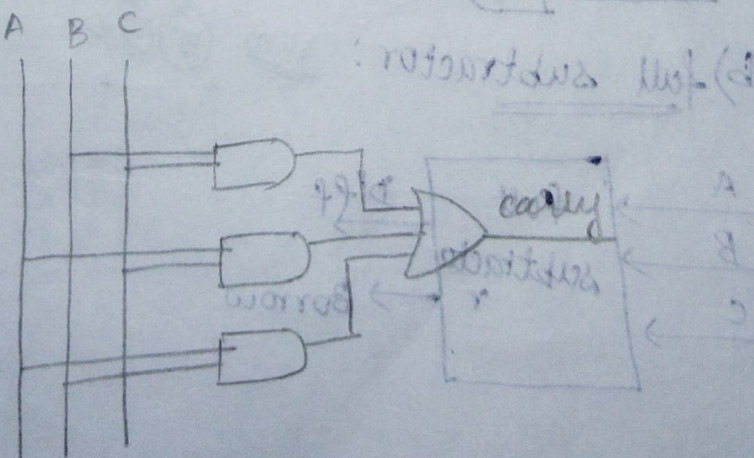
$= BC + AC + AB$ //

Logic circuit:

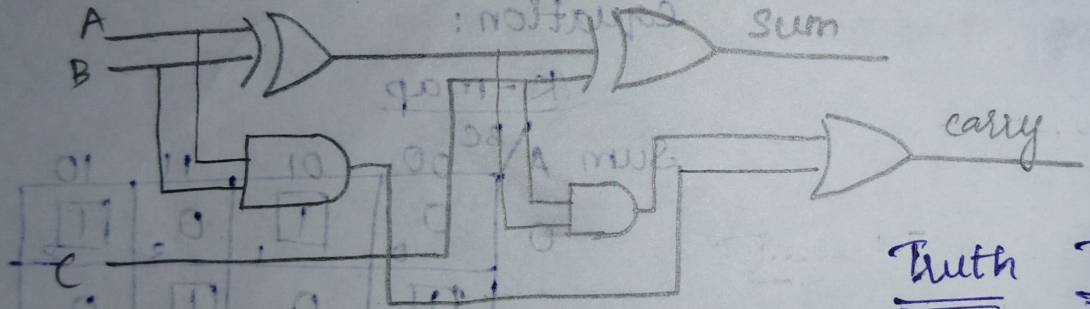
sum



carry:



full adder circuit using 2 half adder.

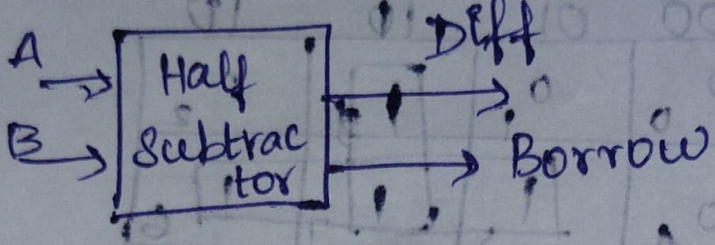


Truth Table

2) Subtractors

a) Half subtractor: (3)

Block diagram:



Truth Table

A	B	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

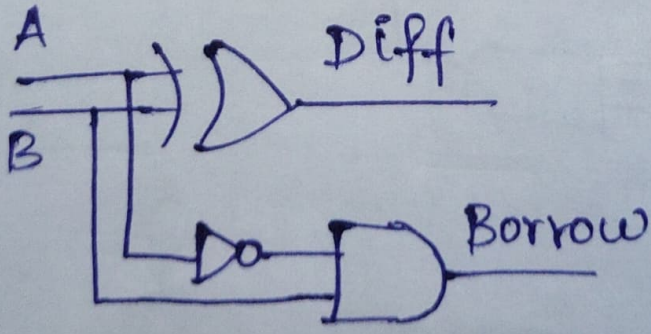
Equation:

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

$$= A \oplus B$$

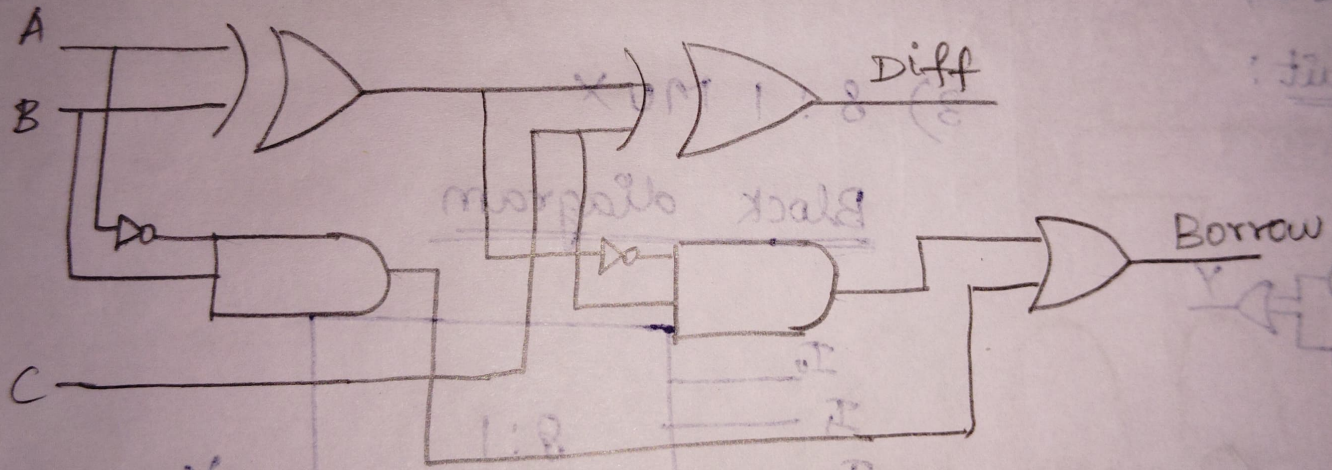
$$Borrow = \bar{A}B$$

Logic circuit



circuit

full subtractor using 2 half subtractor:



12/08/2024

MAGNITUDE COMPARATOR: (4) Q.B (16m)

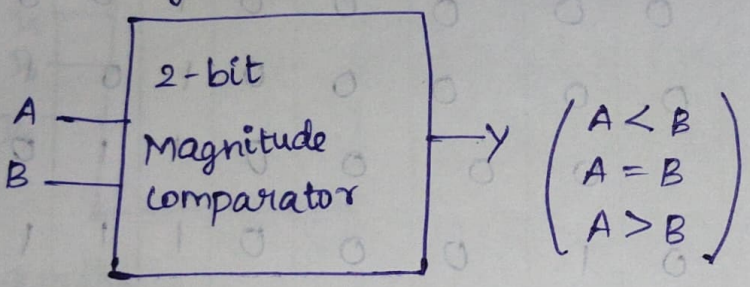
It is a combinational circuit designed to compare the relative magnitude of two binary numbers (A & B) and generates one of the following output

i) $A = B$ ii) $A \gtrless B$ iii) $A < B$

Types: i) Two bit magnitude comparator
four bit magnitude comparator

1) 2 bit Magnitude comparator:

Block diagram



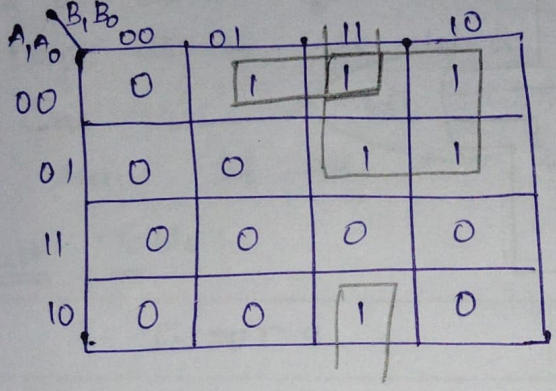
Truth Table

4 2 method

A_1 A_0	B_1 B_0	$A < B$	$A = B$	$A > B$
0 0	0 0	0	1	0
0 0	0 1	1	0	0
0 0	1 0	1	0	0
0 0	1 1	1	0	0
0 1	0 0	0	0	1
1 0	1 0	0	1	0
0 1	1 0	1	0	0
0 1	1 1	1	0	0

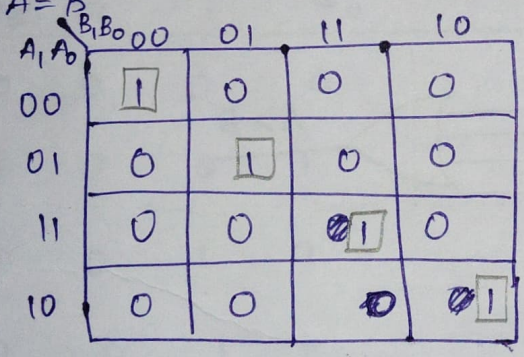
1	0	0	0	0	0	0	1
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	1	0	0	1
1	1	0	0	0	0	1	0
1	1	0	1	0	0	1	0
1	1	1	0	0	0	1	0
1	1	1	1	0	1	0	0

k-map $A < B$



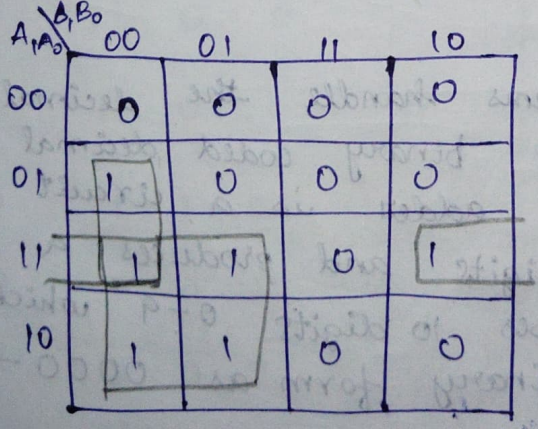
$A<B = \bar{A}_1 B_1 + \bar{A}_1 \bar{A}_0 B_0 + \bar{A}_0 B_1 B_0$

$A = B$



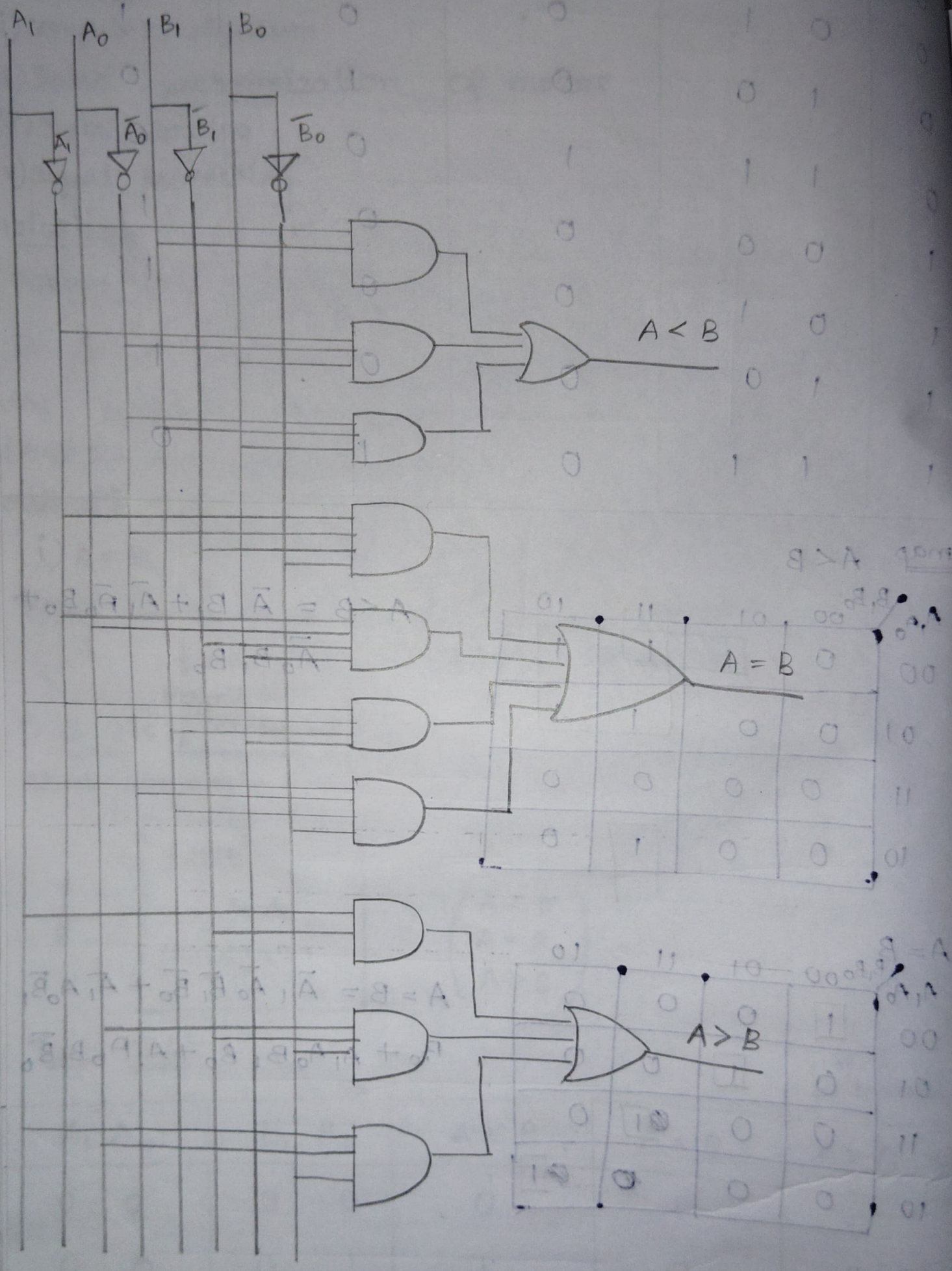
$A = B = \bar{A}_1 \bar{A}_0 \bar{B}_1 \bar{B}_0 + \bar{A}_1 A_0 \bar{B}_1 B_0 + A_1 A_0 B_1 B_0 + A_1 \bar{A}_0 B_1 \bar{B}_0$

$A > B$

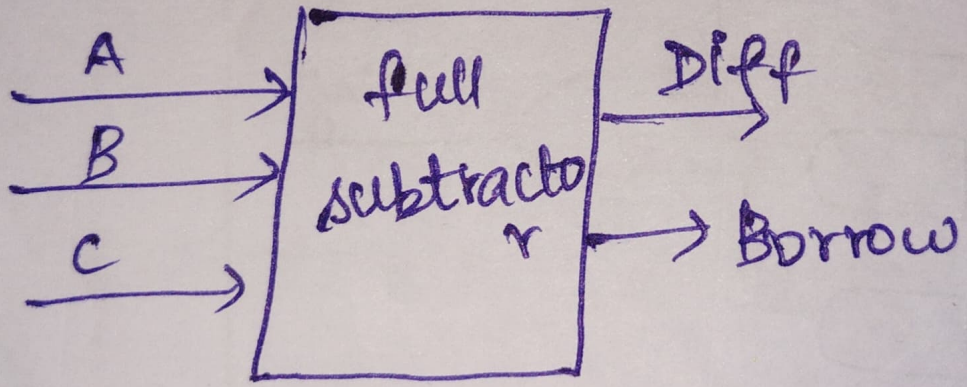


$A > B = A_0 \bar{B}_1 \bar{B}_0 + A_1 \bar{B}_1 + A_1 A_0 \bar{B}_0$

LOGIC CIRCUIT:



b) full subtractor : 1bm (5) Q.5

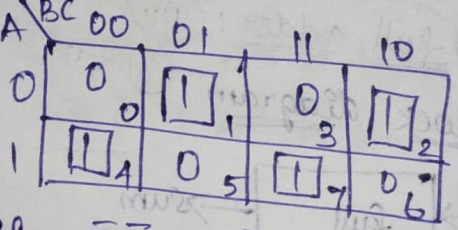


Truth Table

A	B	C	D	B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

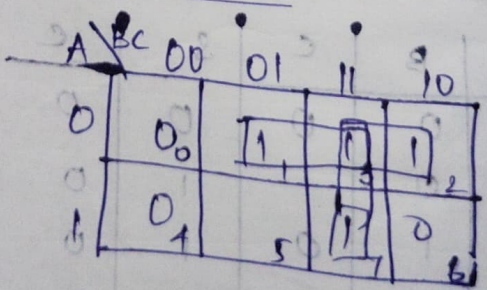
Equation

k-map for Diff



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

k-map for Borrow

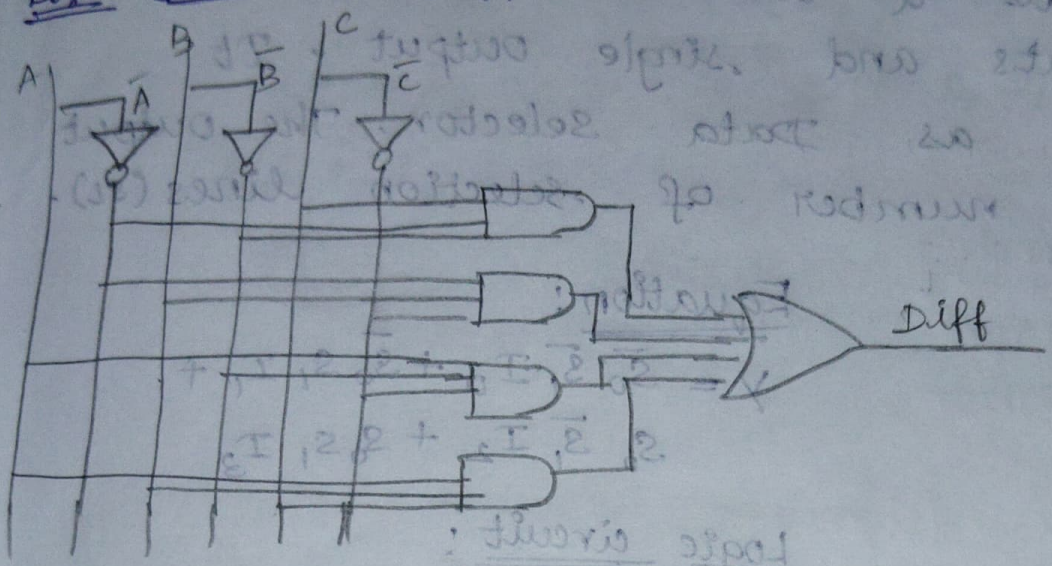


Borrow

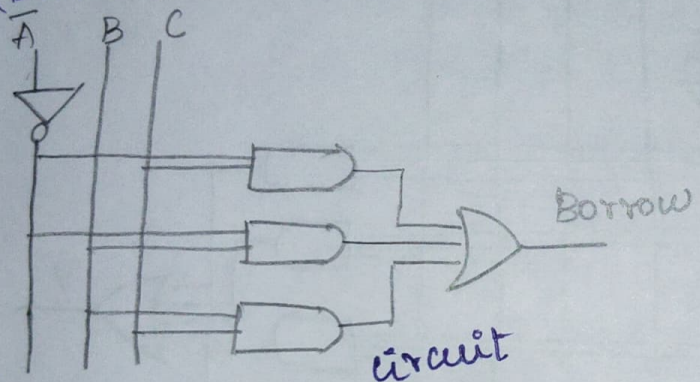
$= \bar{A}\bar{B}C + \bar{A}BC + \bar{A}BC + \bar{A}B\bar{C} + \bar{A}BC + ABC$

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

Logic circuit : (Diff)



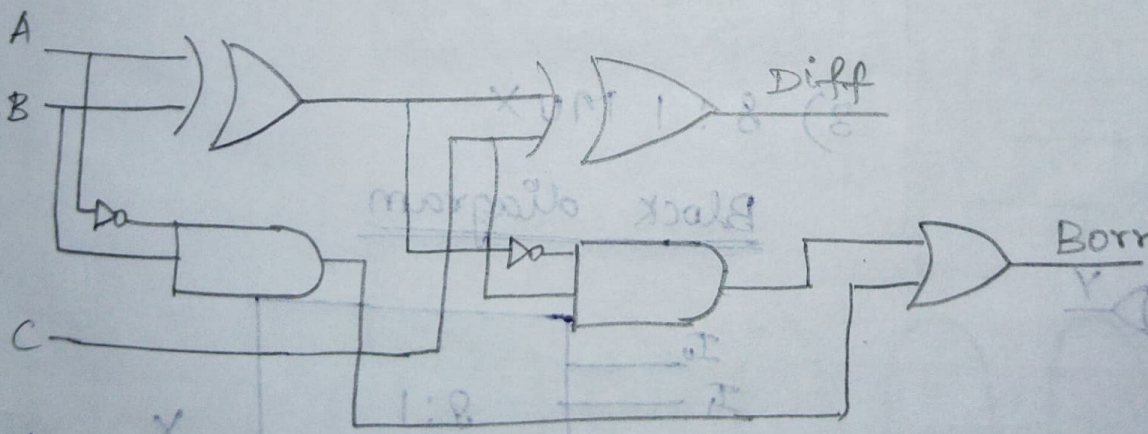
(Borrow) :



circuit

Y	Z
0	0
1	1

full subtractor using 2 half subtractor :



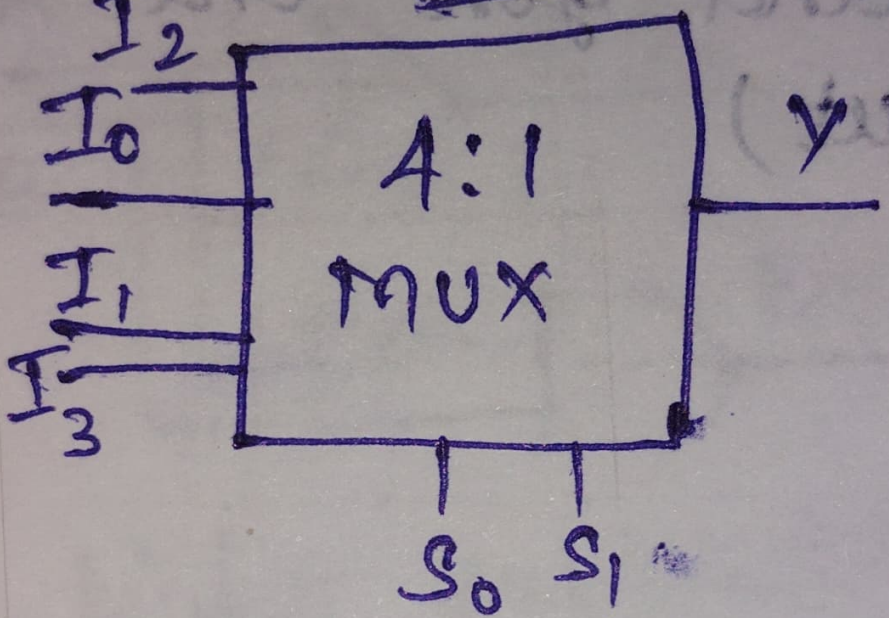
⇒ Multiplexer is a combinational circuit with many inputs and single output. It is also called as Data selector. The output also depends on number of selection lines (SL).

(6) Q.B

2) 4 : 1 MUX

B.D

8m



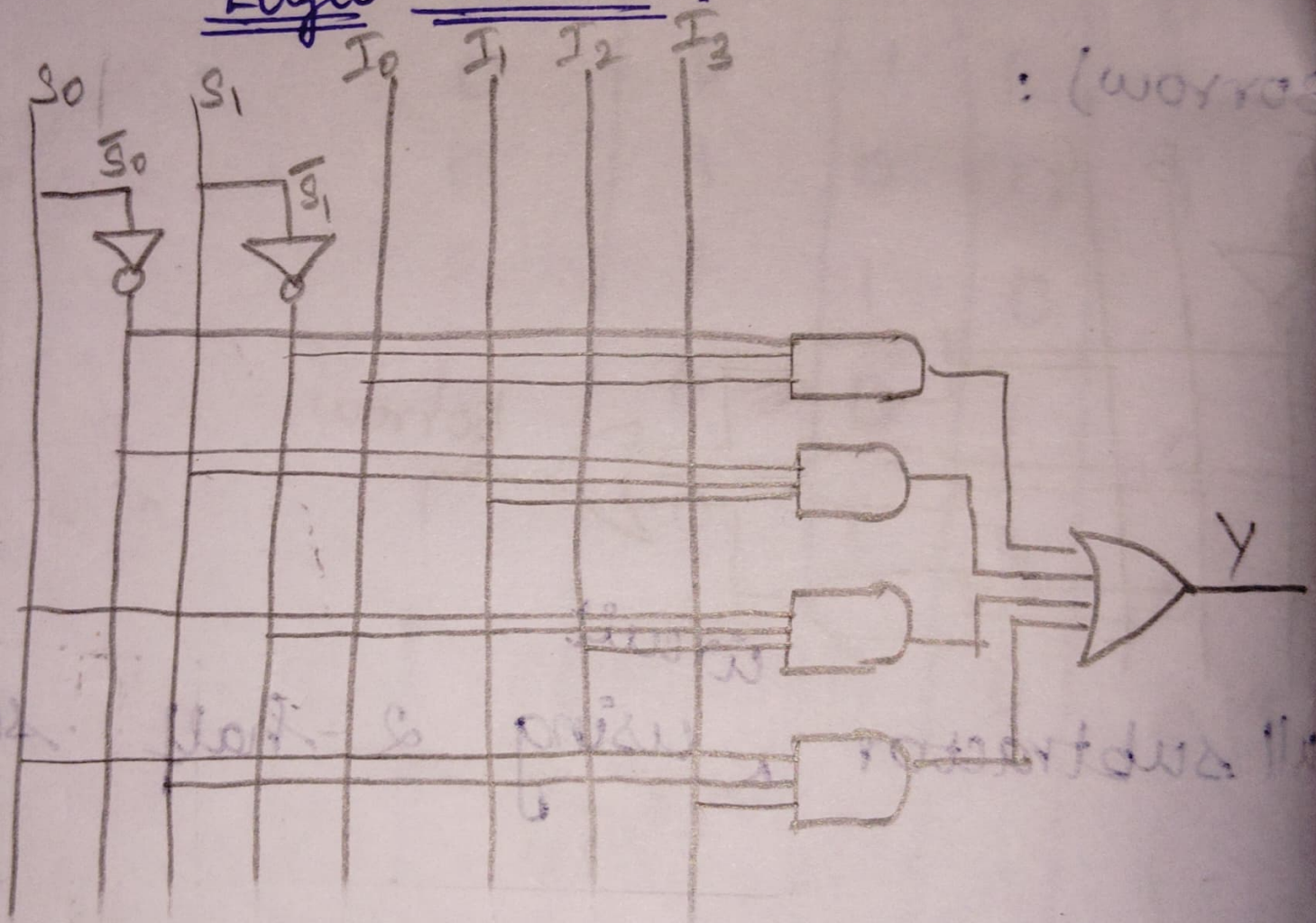
Truth Table

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

Equation:

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

Logic circuit:

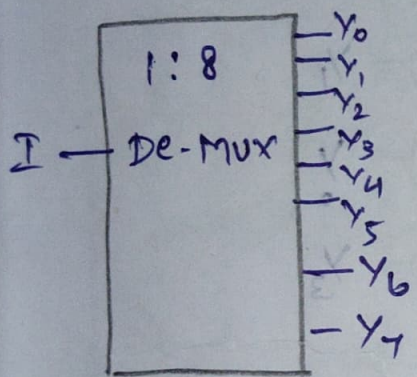


D_7	S_0	S_1	S_2
	1	1	1

De-Multiplexer :

It has one input and Transmit the same over several outputs. The inputs and the outputs depends on the number of selection lines. 1:8 (6) 9.B 8m.

Block diagram



S_0	S_1	S_2	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	I	0	0	0	0	0	0	0
0	0	1	0	I	0	0	0	0	0	0
0	1	0	0	0	I	0	0	0	0	0
0	1	1	0	0	0	I	0	0	0	0
1	0	0	0	0	0	0	I	0	0	0
1	0	1	0	0	0	0	0	I	0	0
1	1	0	0	0	0	0	0	0	I	0
1	1	1	0	0	0	0	0	0	0	I

Equation : $Y_0 = \bar{S}_0 \bar{S}_1 \bar{S}_2 I$, $Y_1 = \bar{S}_0 \bar{S}_1 S_2 I$

$Y_2 = \bar{S}_0 S_1 \bar{S}_2 I$, $Y_3 = \bar{S}_0 S_1 S_2 I$

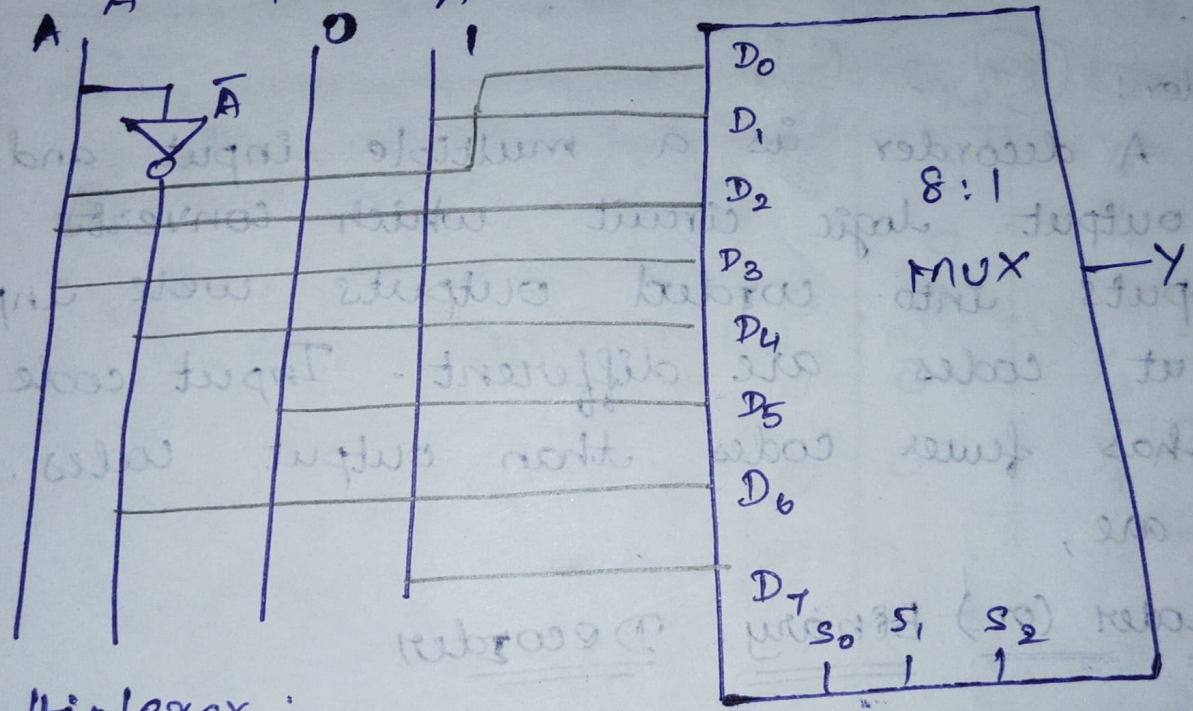
$Y_4 = S_0 \bar{S}_1 \bar{S}_2 I$, $Y_5 = S_0 \bar{S}_1 S_2 I$

$Y_6 = S_0 S_1 \bar{S}_2 I$, $Y_7 = S_0 S_1 S_2 I$

(Draw logic circuit).

2) Simplify the function $f(w, x, y, z) = \Sigma(1, 4, 6, 7, 8, 9, 10, 11, 15)$ using 8:1 multiplexer. (7) Q.8 8m

	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
$\bar{A}$	0	(1)	2	3	(4)	5	(6)	(7)
A	(8)	(9)	(10)	(11)	12	13	14	(15)
	A	1	A	A	$\bar{A}$	0	$\bar{A}$	1



Multiplexer :

BCD adder:

The digital systems handle the decimal number in the form of binary coded decimal numbers (BCD). A BCD adder is a circuit that adds 2 BCD digits and produces a sum in BCD. It uses 10 digits 0-9 which are represented in Binary form as 0000-1001.

BCD Addition Procedure: (8) Q.B 8m

i) Add 2 BCD numbers using ordinary binary addition.

ii) If 4 bit sum is equal to or lesser than 9 then no correction is needed

iii) If 4 bit sum is greater than 9 or If a carry is generated then the sum is invalid.

iv) To correct the invalid sum add 0110 to the 4 bit sum and If a carry results from this addition, add it to next higher order BCD digits.

⇒ To implement BCD adder we require :

i) 4 bit adder for initial addition.

ii) Logic circuit to detect the sum greater than 9.

iii) One more 4 bit adder to add 0110 in the sum If the sum is greater than 9.

Truth Table:

INPUTS				OUTPUT
S_3	S_2	S_1	S_0	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0

this BCD

1 0 1 0

1 0 1 1

1 1 0 0

1 1 1 0

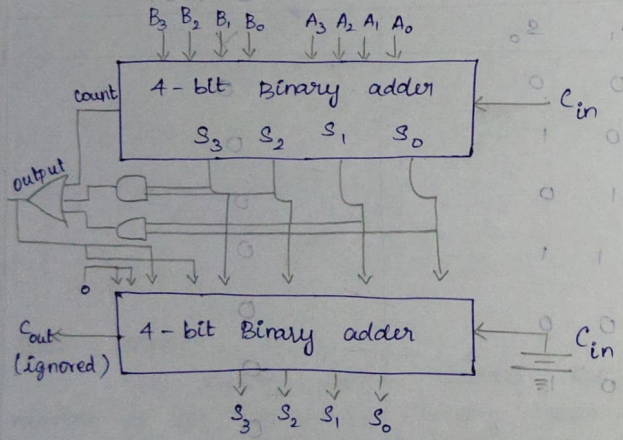
1 1 1 1

k-map:

S_3S_2 \ S_1S_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	0	0	1	1

$Y = S_3S_2 + S_3S_1$

Block diagram of BCD adder:



from the figure the two BCD numbers together with input carry are first adder in the top four bit binary adder to produce binary sum.

When the output carry is equal to zero nothing is added to binary sum.

When it is equal to 0, binary 0110 is added to the binary sum through the bottom 4 bit binary adder.

The output carry generated from the bottom binary adder can be ignored, since it supplies information already available at the output carry terminal.