INDEX

NAME: Pandya Janardan STD.: SEC.: ROLL NO.: SUB.: IC-121 Lab

S. No.	Date	Title	Page No.	Teacher's Sign / Remarks
1.	13/04/22	Study of Universal Logic Gates	1	
2.	20/04/22	Incrementer & Decrementer Design	6 11	
3.	27/04/22	2's complementer & Incrementer / Decrementer		
4.	02/09/22	BCD to 7-segment LED Display	20	
5.	09/09/22	Comparator circuit & MAX-MIN	25	
6.	27/09/22	Familiarization with logic gates and controlled one's & two's complement operation on breadboard	28	
7.	30/05/22	Familiarization with counter operations & its display mods.	33	
8.	06/06/22	Counter Output through 7 segment display	36	
9.	17/06/22	Synthesis and Simulation of Full Adder Design	39	
10.	20/06/22	Synthesis & Simulation of Barrel Shifter	41	
11.	27/06/22	Synthesis & Simulation of counter & 8:1 Mux using 2:1 Mux.	48	

LAB-1

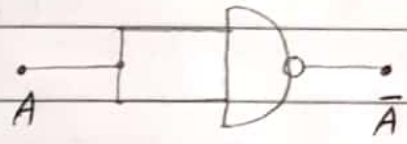
Study of Universal Logic Gates

classmate
Date 13/04/22
Page 1

Q.1 Using 2-Input NAND gates only realize AND, OR, NOT, NOR, XOR, XNOR gates in the most optimized fashion.

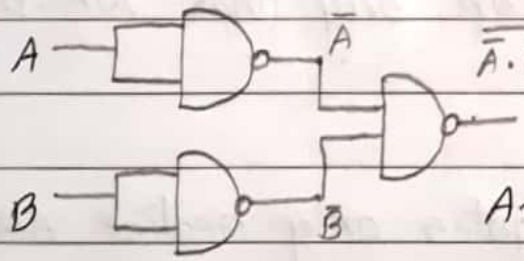
Ans.

NOT Gate

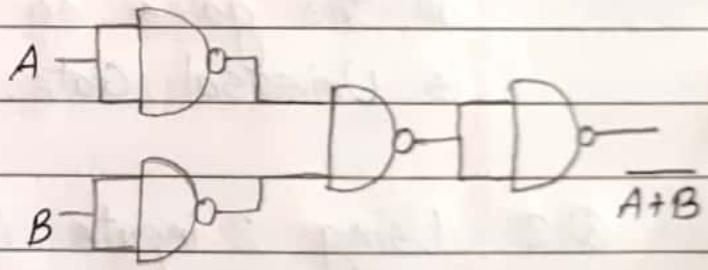


AND Gate

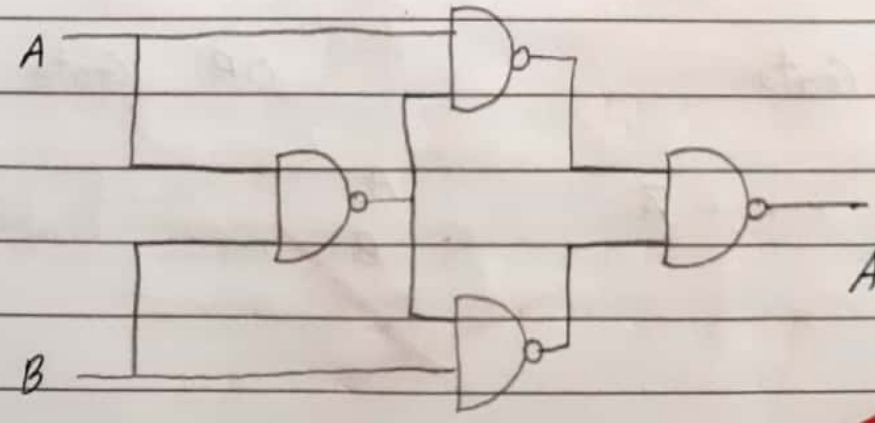
OR Gate



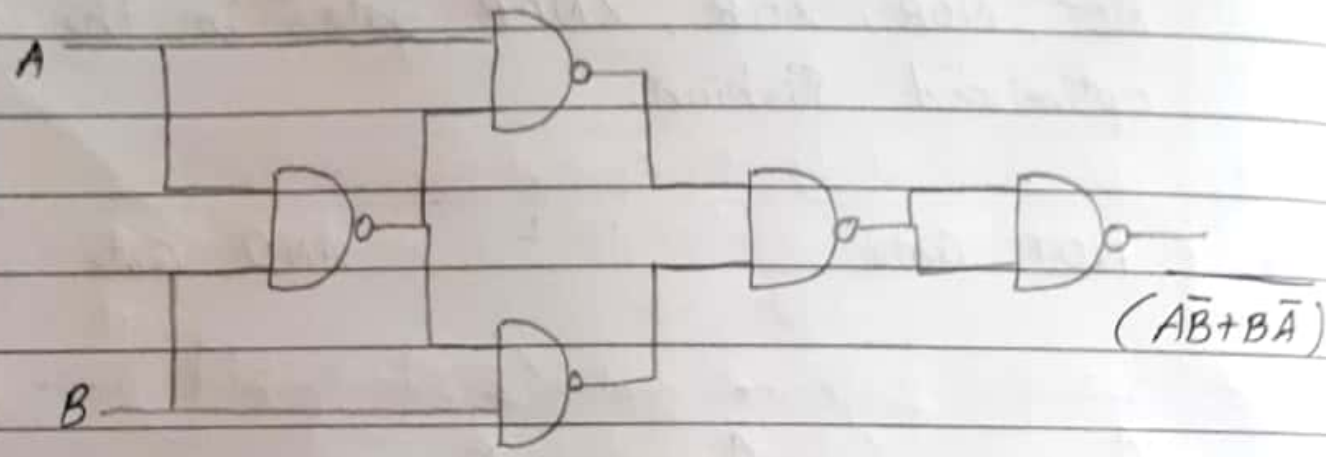
NOR Gate



XOR Gate



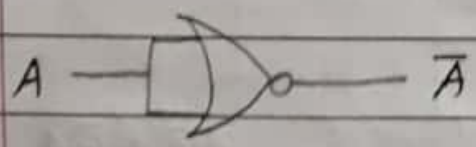
XNOR Gate



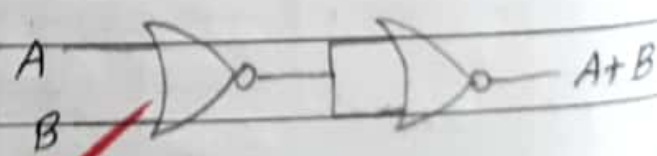
Conclusion:
Here with use of NAND Gate only we have build all the gates so we can say that NAND Gate is Universal Gate.

Q.2 Using 2 inputs NOR gates only realize AND, OR, NOT, NAND, XOR, XNOR gates in the most optimized way.

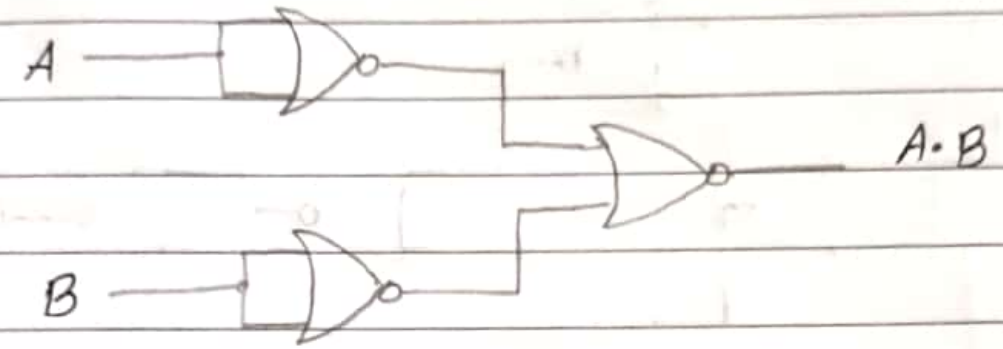
Ans. NOT Gate



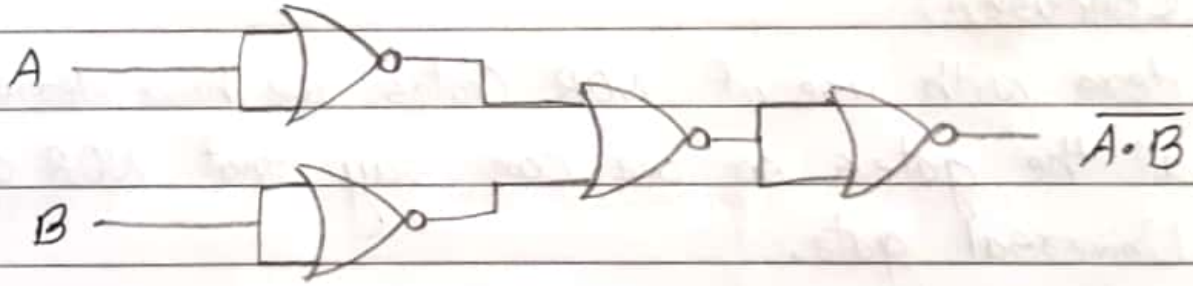
OR Gate



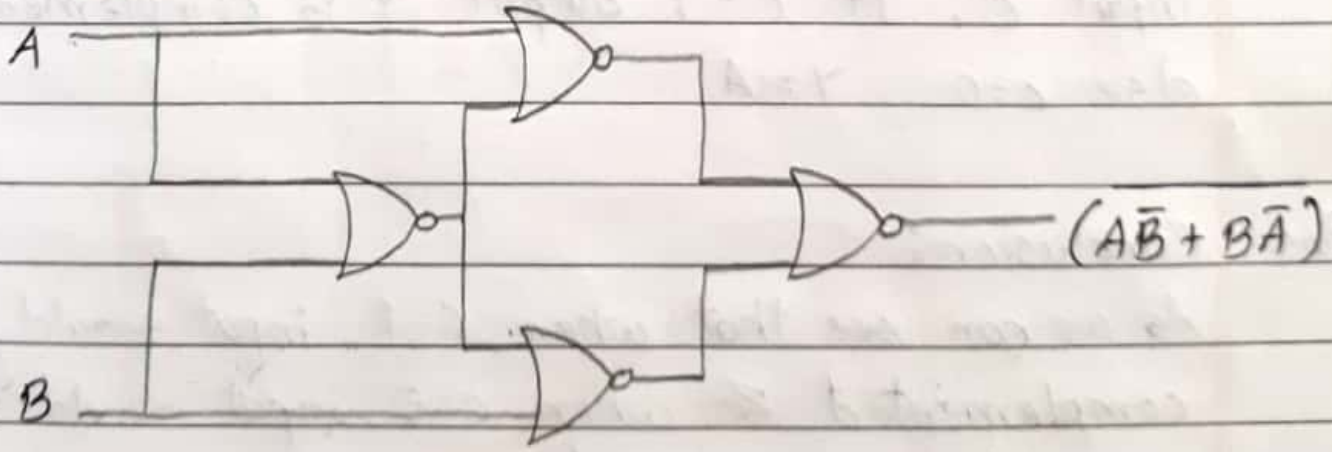
AND Gate



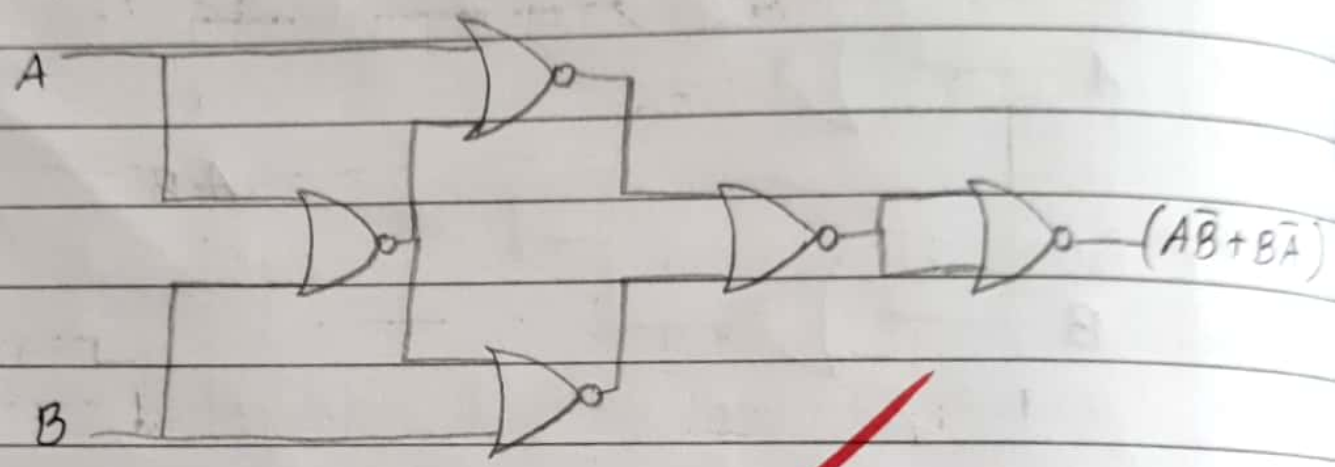
NAND Gate



XNOR Gate



XOR Gate

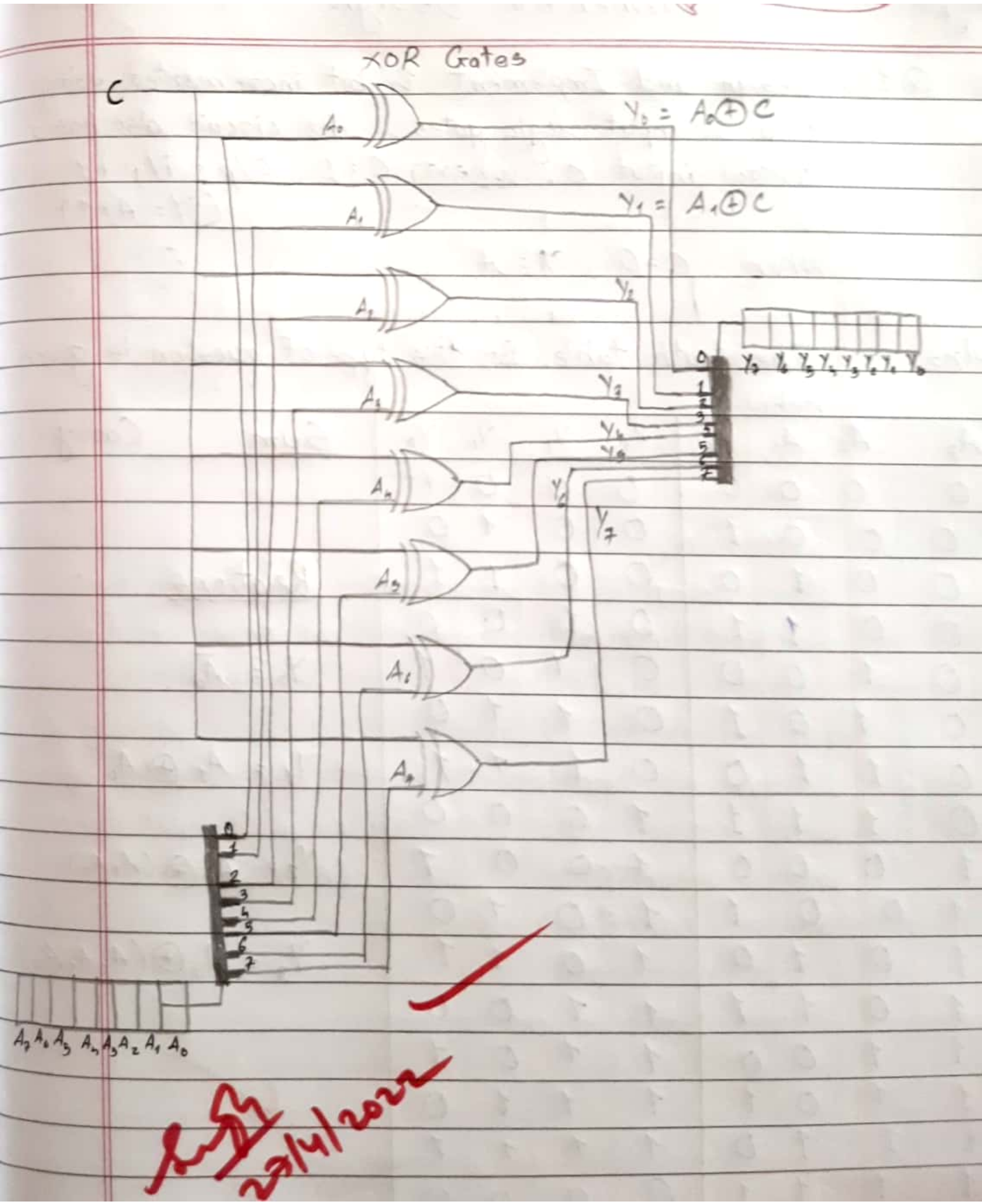


Conclusion:
Here with use of NOR Gates we have derived all the gates so we can say that NOR gate is Universal gate.

Q.3 You are Given an 8 bit input A and a control input C. If C = 1 output Y is complement of A else C=0 , Y = A.

Ans. Conclusion:
As we can see that when C=1, input should get complemented & when C=0, input would remain same. So I have used XOR gates.

XOR Gates



23/4/2022

LAB - 2

Incrementer & Decrementer Design

Q.1 (a) Design and Implement 4 bit incrementer using only 2 input logic gates. The circuit also has a control input C. When C = 1 O/p = i/p + 1 (Y = A + 1)

When C = 0, Y = A

Ans. The truth table for this type of question is given below.

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0	Sum	Carry
0	0	0	0	0	0	0	1		
0	0	0	1	0	0	1	0		
0	0	1	0	0	0	1	1		
0	0	1	1	0	1	0	0		
0	1	0	0	0	1	0	1		
0	1	0	1	0	1	1	0		
0	1	1	0	0	1	1	1		
0	1	1	1	1	0	0	0		
1	0	0	0	1	0	0	1		
1	0	0	1	1	0	1	0		
1	0	1	0	1	0	1	1		
1	0	1	1	1	1	0	0		
1	1	0	0	1	1	0	1		
1	1	0	1	1	1	1	0		
1	1	1	0	1	1	1	1		
1	1	1	1	0	0	0	0		

Relations

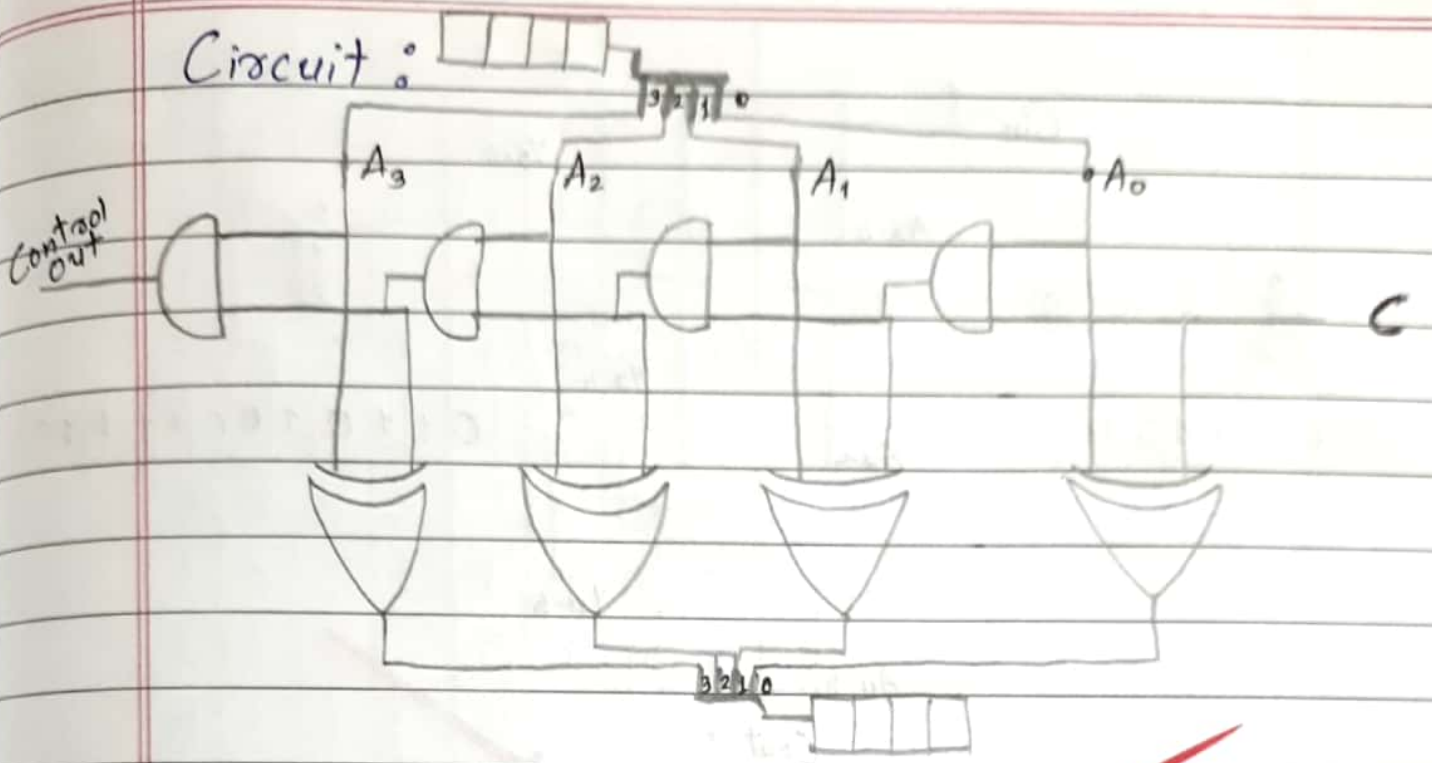
$Y_0 = \overline{A_0}$

$Y_1 = A_1 \oplus A_0$

$Y_2 = A_2 \oplus (A_1 A_0)$

$Y_3 = A_3 \oplus (A_2 A_1 A_0)$

Circuit :

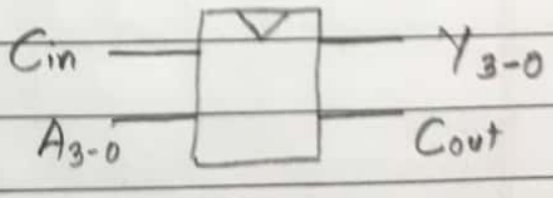


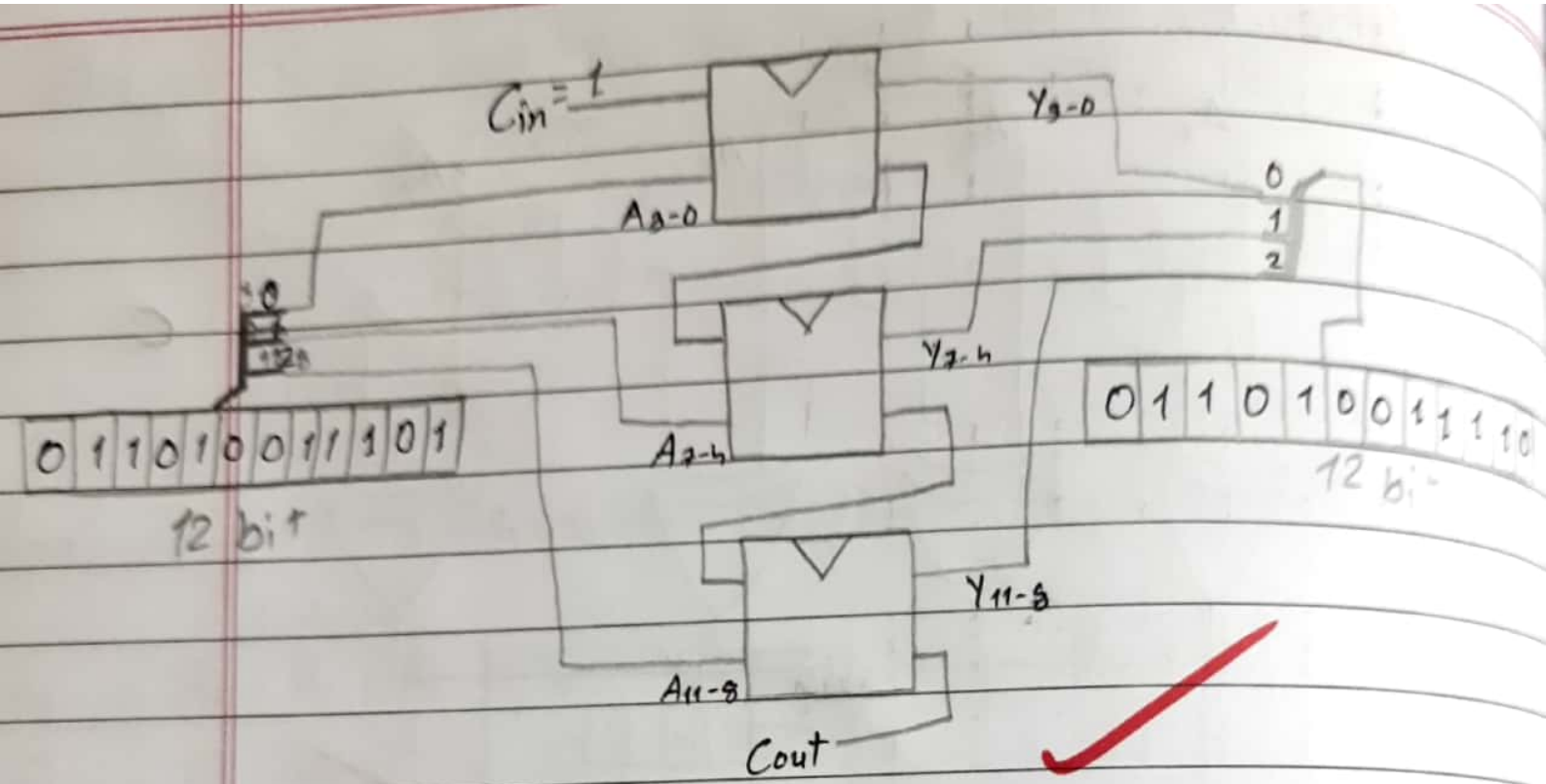
Conclusion :

This is a 4 bit increment circuit, where the C is control input and the output is carry for overall circuit & we can use it as control input in bigger circuits.

(b) Using the 4 bit incrementer as a sub circuit, construct a 12-bit incrementer.

4 bit incrementer





Conclusion :
Here using 4-bit incrementer as a subcircuit we built 12 bit incrementer as shown in figure.

Q. 2 Design & Implement 4-bit decrementer circuit with control input C using only 2 input logic gate. When C=0 O/P = I/P - 1 (Y = A-1)
When C=1, Y = A.

Ans. Truth table is as given

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
1	1	1	1	1	1	1	0
1	1	1	0	1	1	0	1
1	1	0	1	1	1	0	0
1	1	0	0	1	0	1	1
1	0	1	1	1	0	1	0
1	0	1	0	1	0	0	1
1	0	0	1	1	0	0	0
1	0	0	0	0	1	1	1
0	1	1	1	0	1	1	0
0	1	1	0	0	1	0	1
0	1	0	1	0	1	0	0
0	1	0	0	0	0	1	1
0	0	1	1	0	0	1	0
0	0	1	0	0	0	0	1
0	0	0	1	0	0	0	0
0	0	0	0	1	1	1	1

Relations

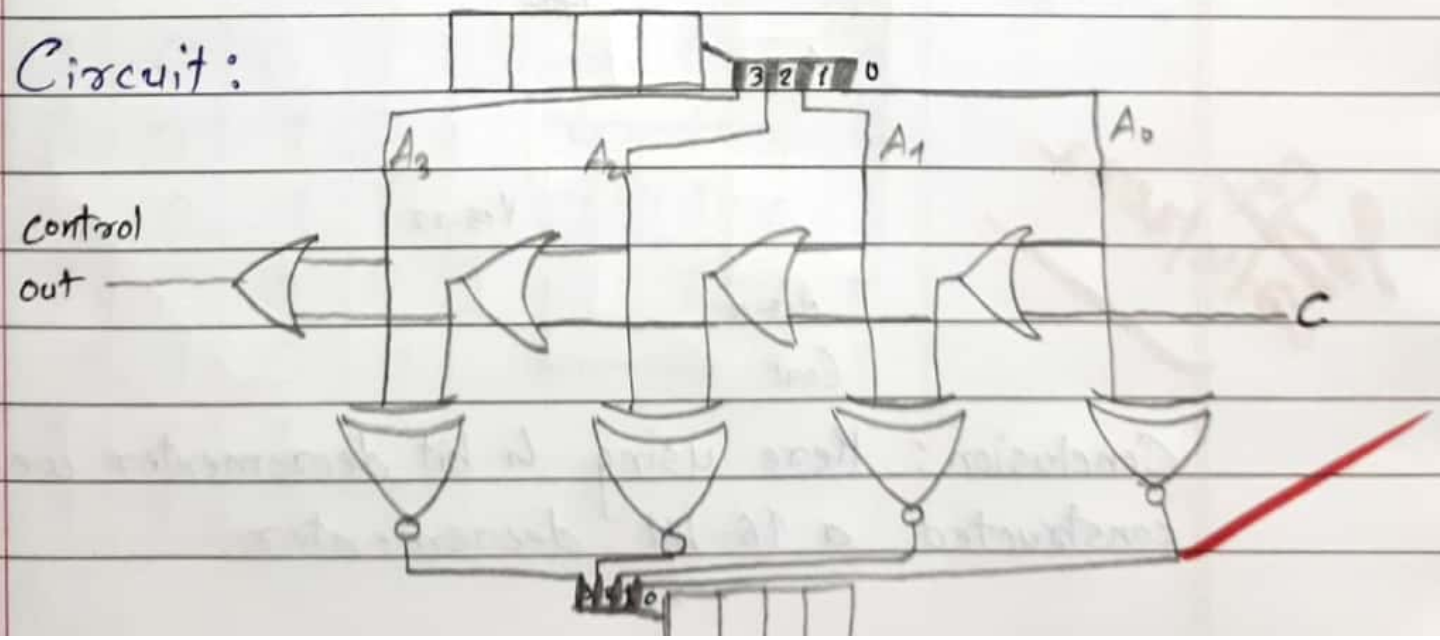
$$Y_0 = \overline{A_0}$$

$$Y_1 = \overline{A_1 \oplus A_0}$$

$$Y_2 = \overline{A_2 \oplus (A_1 + A_0)}$$

$$Y_3 = \overline{A_3 \oplus (A_2 + A_1 + A_0)}$$

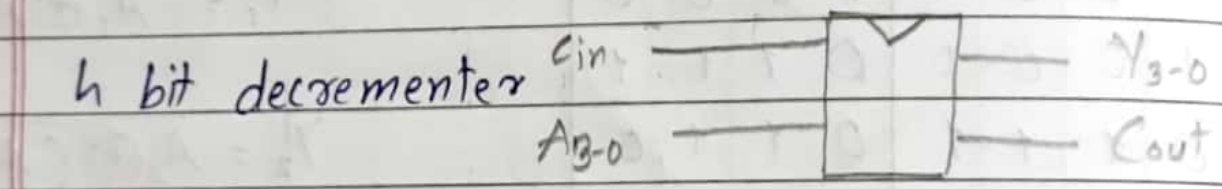
Circuit:



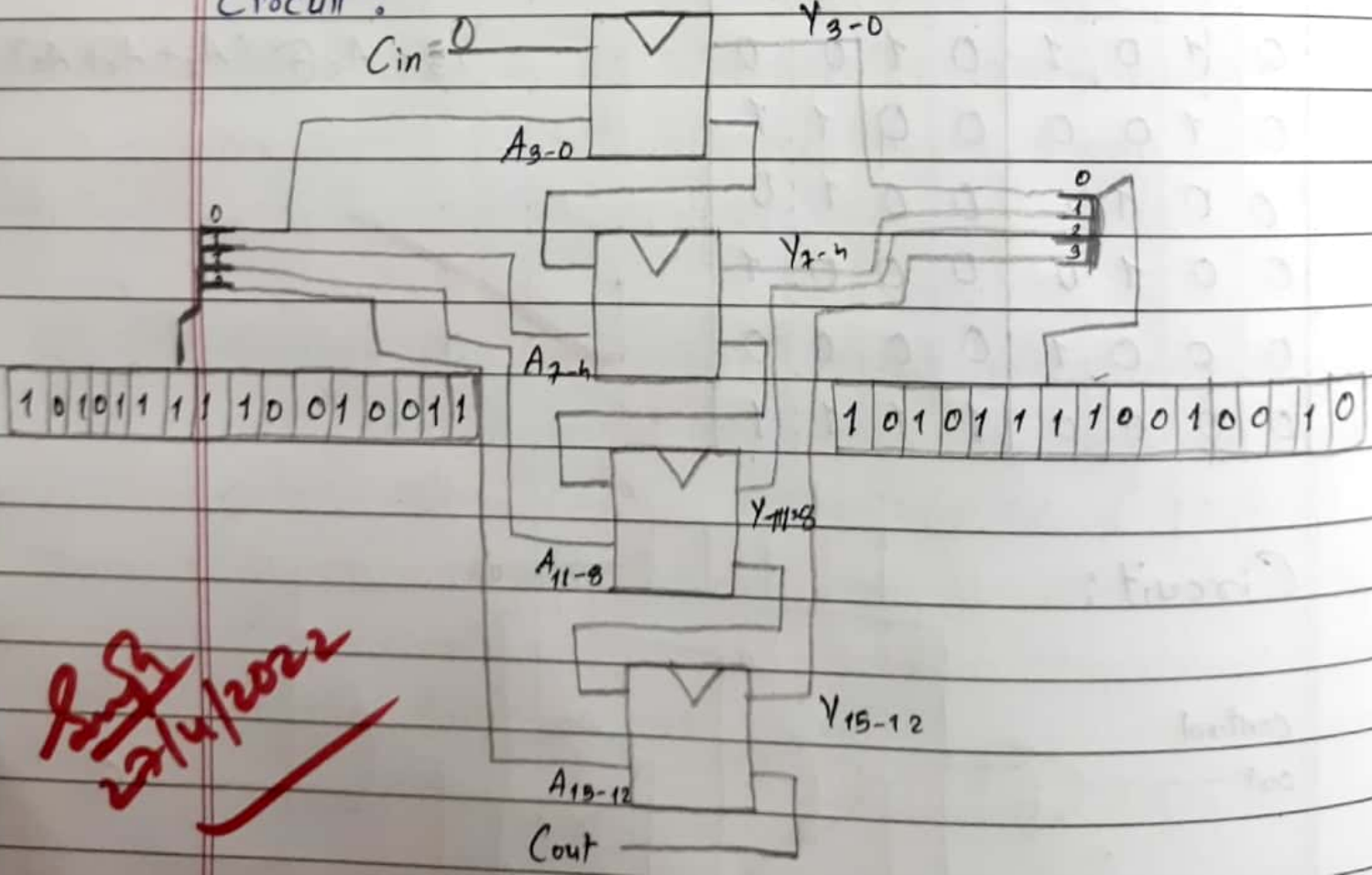
→ So here we draw the circuit of 4 bit decrementer circuit using 2 input XOR gates & OR gates.

→ This is 4 bit decrementer circuit which depends on control switch.

(b) Using the 4 bit decrementer as a subcircuit, constructer, a 16 bit decrementer circuit.



Circuit :



Conclusion : Here using 4 bit decrementer we have constructed a 16-bit decrementer.

LAB-3

classmate
Date
Page 11

Q.1 (a) Using 2-input gates only, design a 4 bit controlled 2's complimenter circuit, which performs 2's complimention when a control input P=1, and leaves the input unchanged otherwise.

Truth table

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1
0	0	1	0	1	1	1	0
0	0	1	1	1	1	0	1
0	1	0	0	1	1	0	0
0	1	0	1	1	0	1	1
0	1	1	0	1	0	1	0
0	1	1	1	1	0	0	1
1	0	0	0	1	0	0	0
1	0	0	1	0	1	1	1
1	0	1	0	0	1	1	0
1	0	1	1	0	1	0	1
1	1	0	0	0	1	0	0
1	1	0	1	0	0	1	1
1	1	1	0	0	0	1	0
1	1	1	1	0	0	0	1

Equation

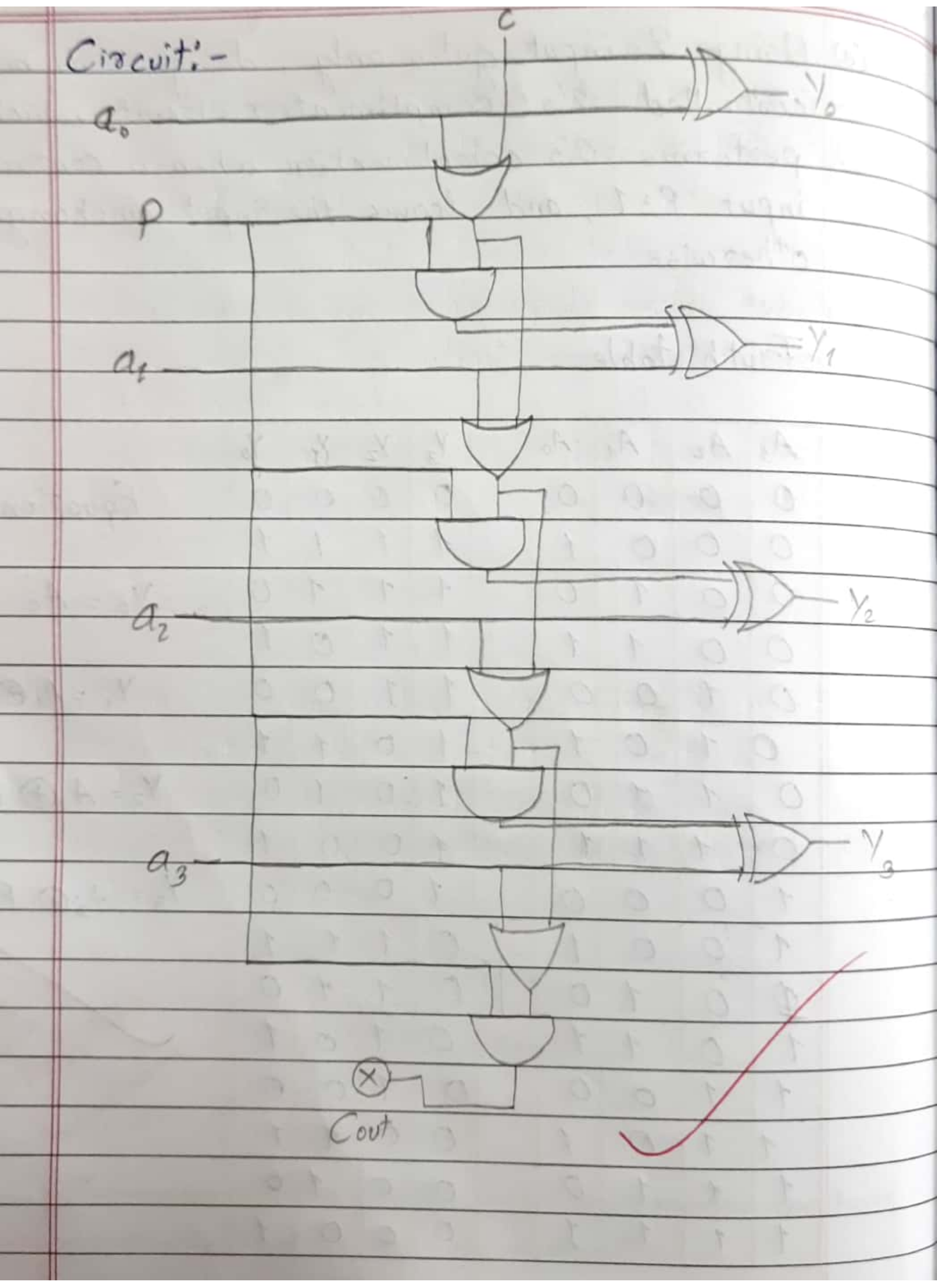
$Y_0 = A_0$

$Y_1 = A_1 \oplus P \cdot A_0$

$Y_2 = A_2 \oplus P \cdot (A_1 + A_0)$

$Y_3 = A_3 \oplus P \cdot (A_2 + A_1 + A_0)$

Circuit:-



Conclusion:-

→ From the truth table we analyze that

$Y_0 = A_0$, $Y_1 = A_1 \oplus P \cdot A_0$, $Y_2 = A_2 \oplus P \cdot (A_1 + A_0)$

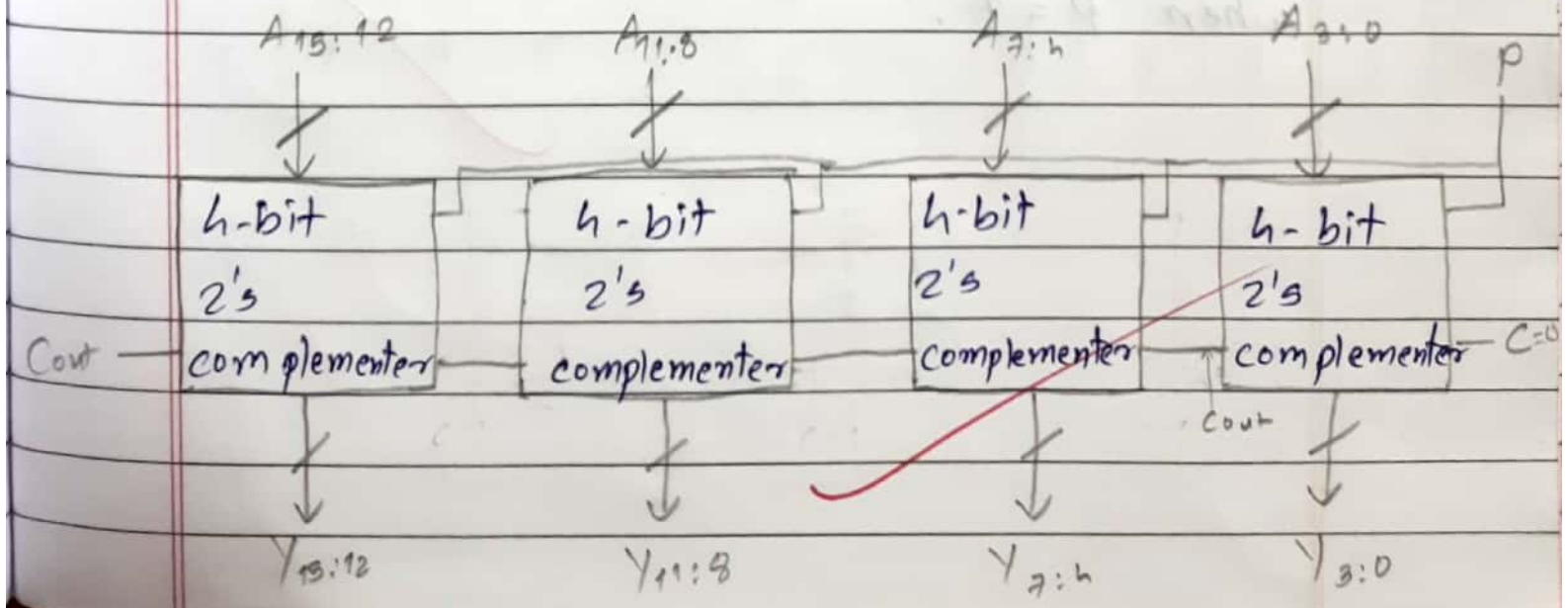
$Y_3 = A_3 \oplus P \cdot (A_2 + A_1 + A_0)$

Here as we can see that I have added P a controling input in equation. so when P = 0 output would be as same as input.

→ Also I have connect a AND gate with XOR gate to include controlling input P.

→ And with the use of XOR, AND and OR gate we have made controlled circuit for 2's compliment.

(b) Using the 4-bit controlled 2's complementer circuit as a sub-circuit design a 16 bit controlled 2's complementer circuit.



Conclusion :-

→ Here, We see that it is a 16 - bit controlled switch, which is built from 4 - bit controlled switch 2's complementer.

→ Here, we see that when switch is a the output is as same as input and when the switch is 1, the output is 2's complement of particular input.

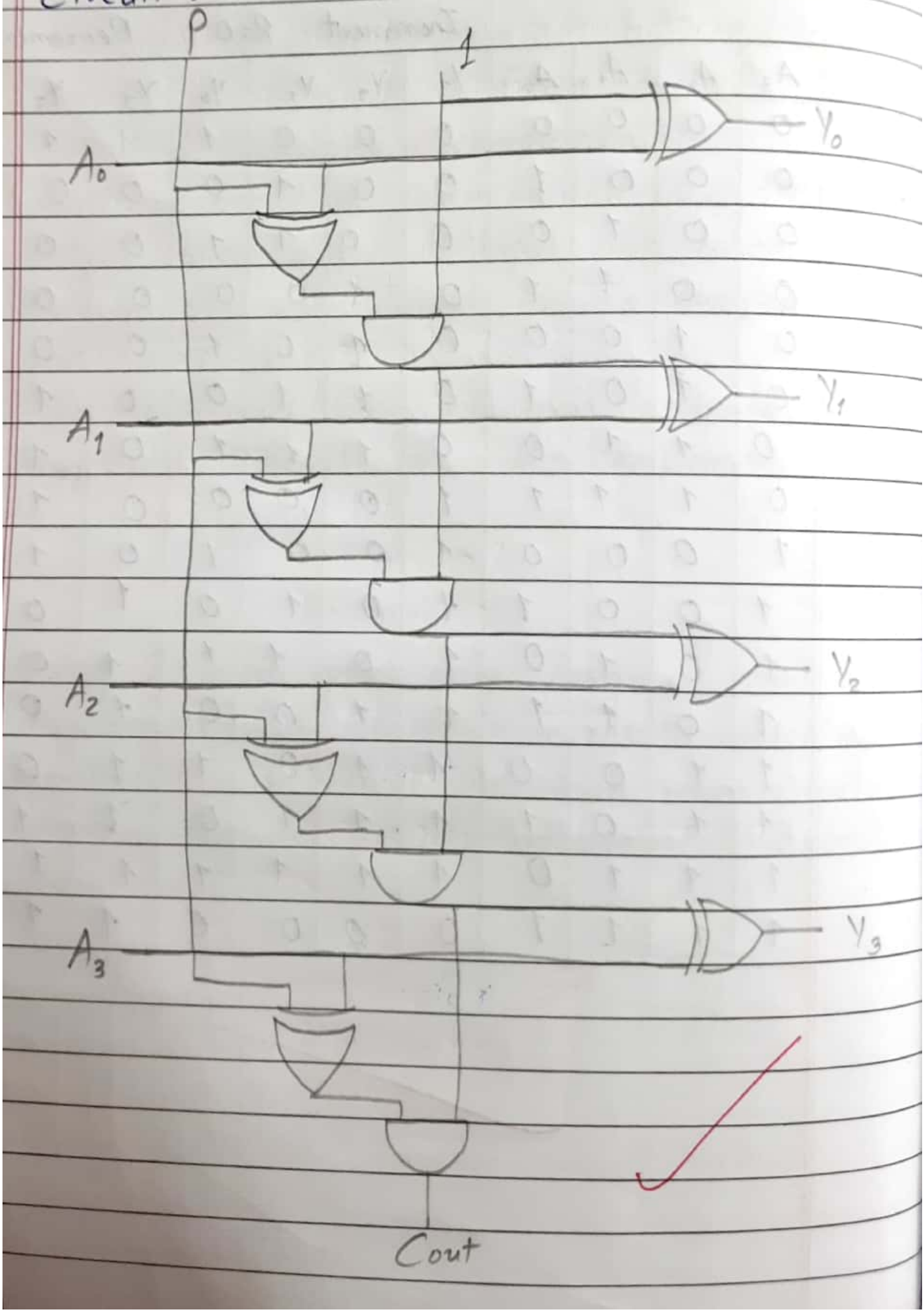
→ In this way we were able to build this type of circuit for 2's complement.

Q. 2(o) Using 2-input gates only, design a 4-bit incrementer / decrementer circuit which performs increment operation when a control input P = 0 & perform decrement operation when P = 1.

Truth table :

				Increment P=0				Decrement P=1			
A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	1	1	1	1	1
0	0	0	1	0	0	1	0	0	0	0	0
0	0	1	0	0	0	1	1	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	0
0	1	0	0	0	1	0	1	0	0	1	1
0	1	0	1	0	1	1	0	0	1	0	0
0	1	1	0	0	1	1	1	0	1	0	1
0	1	1	1	1	0	0	0	0	1	1	0
1	0	0	0	1	0	0	1	0	1	1	1
1	0	0	1	1	0	1	0	1	0	0	0
1	0	1	0	1	0	1	1	1	0	0	1
1	0	1	1	1	1	0	0	1	0	1	0
1	1	0	0	1	1	0	1	1	0	1	1
1	1	0	1	1	1	1	0	1	1	0	0
1	1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	0	0	0	0	1	1	1	0

Circuit :



Conclusion :

→ From the truth table,

For Increment :- $P=0$

$$Y_0 = \overline{A_0}$$

$$Y_1 = A_1 \oplus A_0$$

$$Y_2 = A_2 \oplus A_1 \cdot A_0$$

$$Y_3 = A_3 \oplus A_2 A_1 A_0$$

For Decrement :- $P = 1$

$$Y_0 = \overline{A_0}$$

$$Y_1 = \overline{A_1 \oplus A_0} = A_1 \oplus \overline{A_0}$$

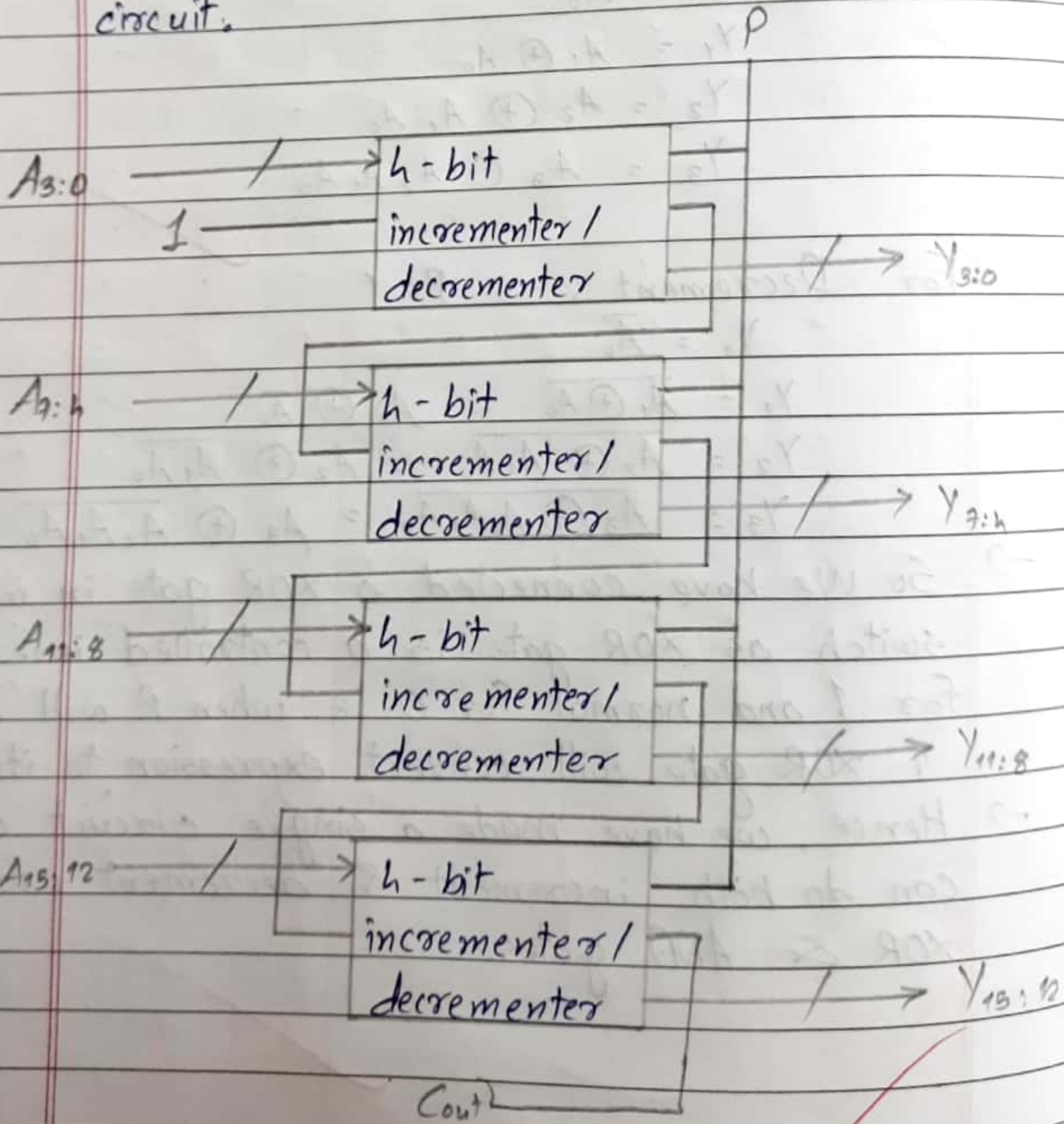
$$Y_2 = \overline{A_2 \oplus A_1 A_0} = A_2 \oplus \overline{A_1 A_0}$$

$$Y_3 = \overline{A_3 \oplus A_2 A_1 A_0} = A_3 \oplus \overline{A_2 A_1 A_0}$$

→ So We have connected a XOR gate in with switch as XOR gate is a controlled inverter for 1 and normal for 0 ie when P will be 1 XOR gate will convert expression to its invers

→ Hence, we have made a single circuit which can do both increment & decrement using XOR & AND gates.

(b) Using 4-bit controlled incrementer and decrementer circuit as a sub circuit design a 16 bit controlled incrementer and decrementer circuit.



Date
Page 19

Conclusion :

→ Here we see that it is a 16-bit incrementer / decrementer circuit which is built from 4-bit incrementer / Decrementer circuit as sub circu

→ Here the cin is fixed as 1

→ And we observe that when p=0 the output is increment by 1 as input & when p = 1 the output is decrement by 1 ~~as~~ of input.

2/5/2022

3.4

BCD to 7-Segment LED display

classmate
Date
Page 20

Que. Design a Roll no. display generator using a BCD to 7-segment display decoder logic.

a) Design a BCD to 7-segment display using logic gates.

b) Design the Roll no. generator circuit whose output feed the BCD to 7 segment [Roll no. 202104---]

★ Truth Table:

A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	x	x	x	x	x	x	x
1	0	1	1	x	x	x	x	x	x	x
1	1	0	0	x	x	x	x	x	x	x
1	1	0	1	x	x	x	x	x	x	x
1	1	1	0	x	x	x	x	x	x	x
1	1	1	1	x	x	x	x	x	x	x

a

AB \ CD	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	X	X	X	X
10	1	1	X	X

$a = A + C + BD + \bar{B}\bar{D}$

b

AB \ CD	00	01	11	10
00	1	1	1	1
01	1	0	1	0
11	X	X	X	X
10	1	1	X	X

$b = \bar{B} + \bar{C}\bar{D} + CD$

c

AB \ CD	00	01	11	10
00	1	1	1	0
01	1	1	1	1
11	X	X	X	X
10	1	1	X	X

$c = B + \bar{C} + D$

d

AB \ CD	00	01	11	10
00	1	0	1	1
01	0	1	0	1
11	X	X	X	X
10	1	1	X	X

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

e

AB \ CD	00	01	11	10
00	1	0	0	1
01	0	0	0	1
11	X	X	X	X
10	1	0	X	X

$e = \bar{B}\bar{D} + C\bar{D}$

f

AB \ CD	00	01	11	10
00	1	0	0	0
01	1	1	0	1
11	X	X	X	X
10	1	1	X	X

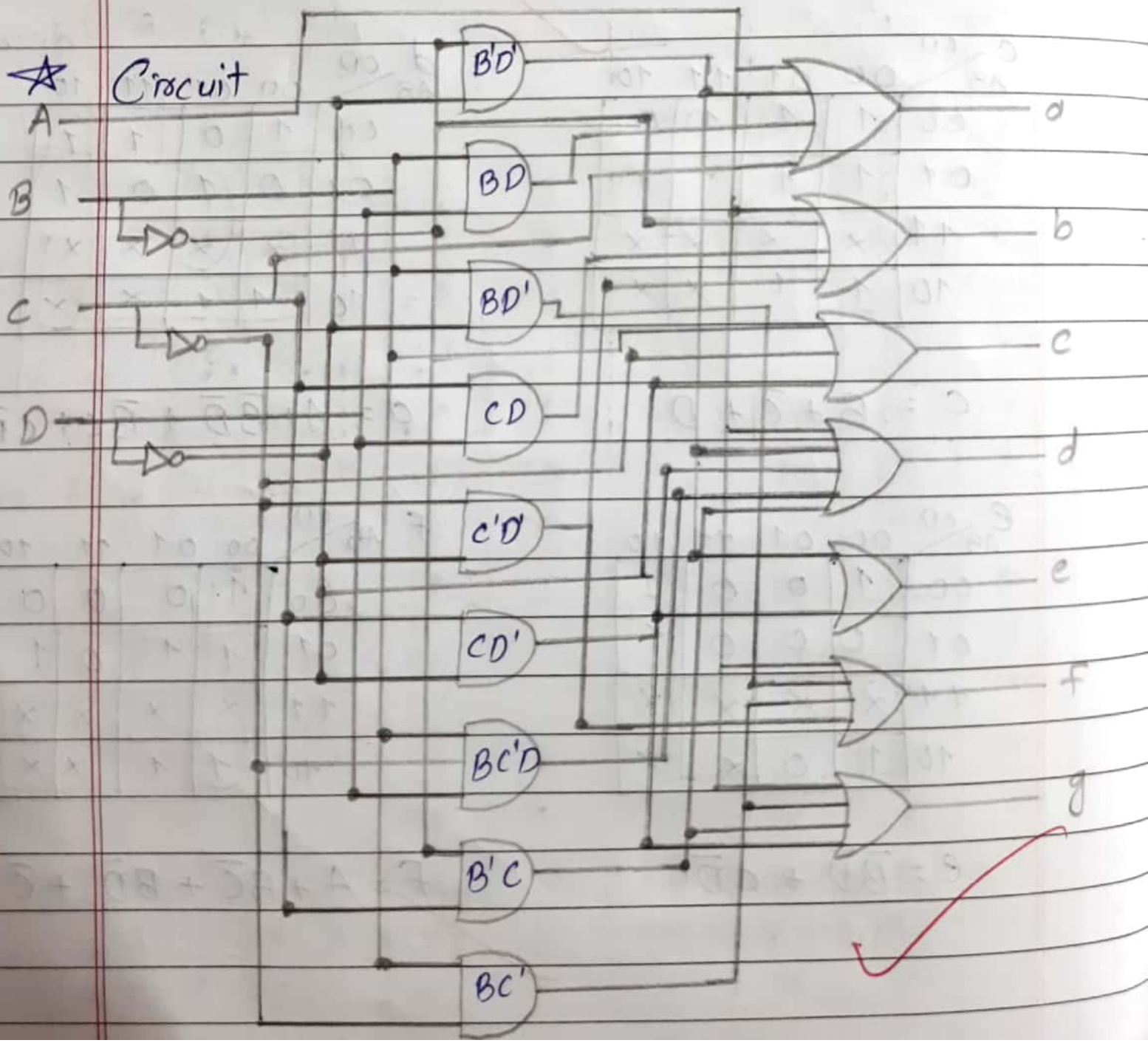
$F = A + B\bar{C} + B\bar{D} + \bar{C}\bar{D}$

Date
Page 22

g_{AB} \ CD	00	01	11	10
00	0	0	1	1
01	1	1	0	1
11	x	x	x	x
10	1	1	x	x

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

★ Circuit



(b) Design Roll no. generator circuit whose o/ps feed the BCD to 7 segment display [2021890817]

☆ Truth Table :

	B_3	B_2	B_1	B_0	P_3	P_2	P_1	P_0
2	0	0	0	0	0	0	1	0
0	0	0	0	1	0	0	0	0
2	0	0	1	0	0	0	1	0
1	0	0	1	1	0	0	0	1
8	0	1	0	0	1	0	0	0
9	0	1	0	1	1	0	0	1
0	0	1	1	0	0	0	0	0
8	0	1	1	1	1	0	0	0
1	1	0	0	0	0	0	0	1

P_0 | B_3B_2 \ B_1B_0

	00	01	11	10
00	0	0	1	0
01	0	1	0	0
11	x	x	x	x
10	1	x	x	x

$P_0 = B_3 + B_2\bar{B_1}B_0 + \bar{B_2}B_1B_0$

P_1 | B_3B_2 \ B_1B_0

	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	x	x	x	x
10	0	x	x	x

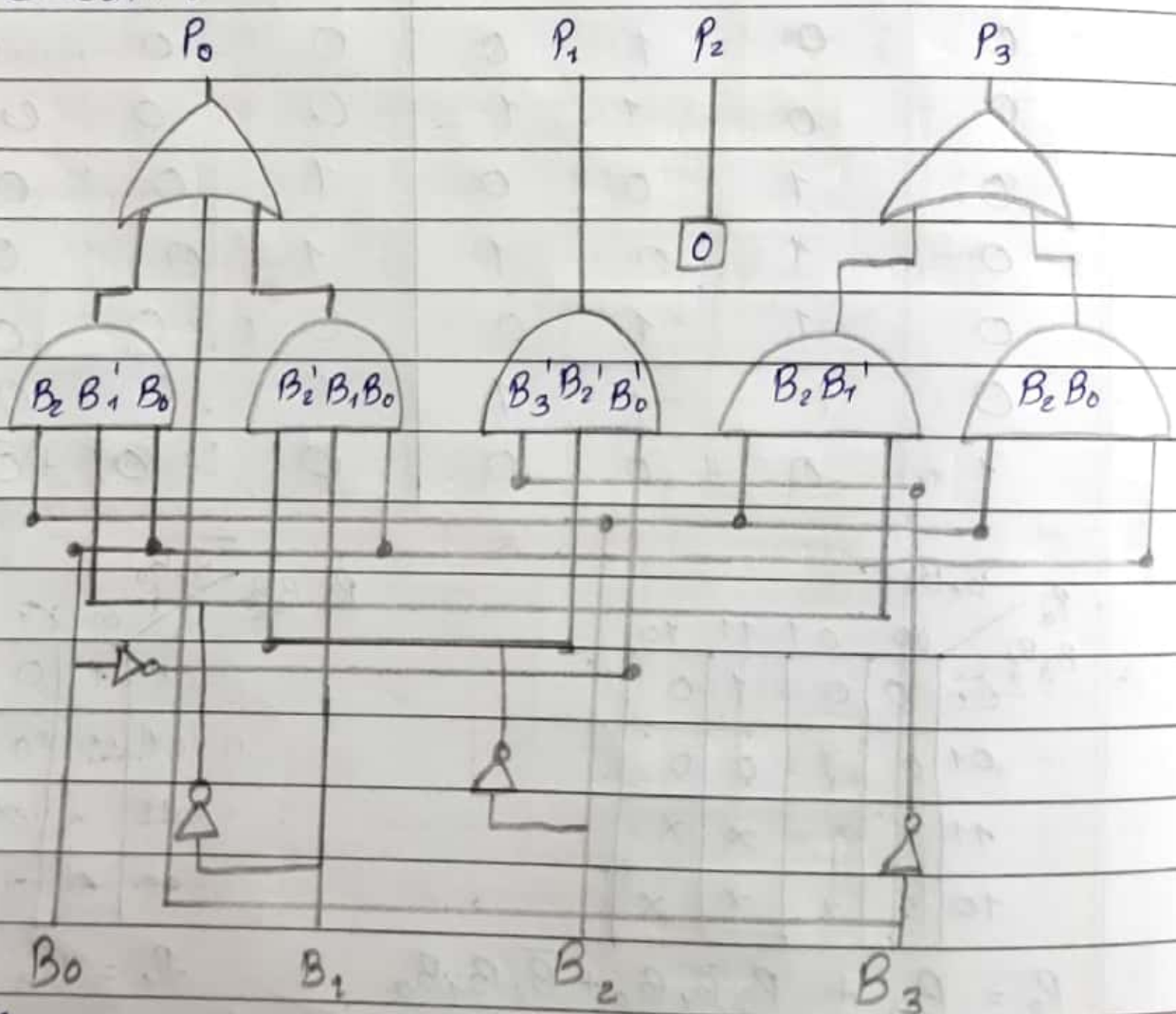
$P_1 = \bar{B_3}\,\bar{B_2}\,\bar{B_0}$

$P_2 = 0$

P_3 B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	0	0
01	1	1	1	0
11	x	x	x	x
10	0	x	x	x

$P_3 = B_2\overline{B_1} + B_2B_0$

Circuit:



So as we change $B_{3:0}$ from 0 to 8 we will get output as 202189081.

Good

Scanned with CamScanner

Lab 5

Comparator Circuit & Max - Min block

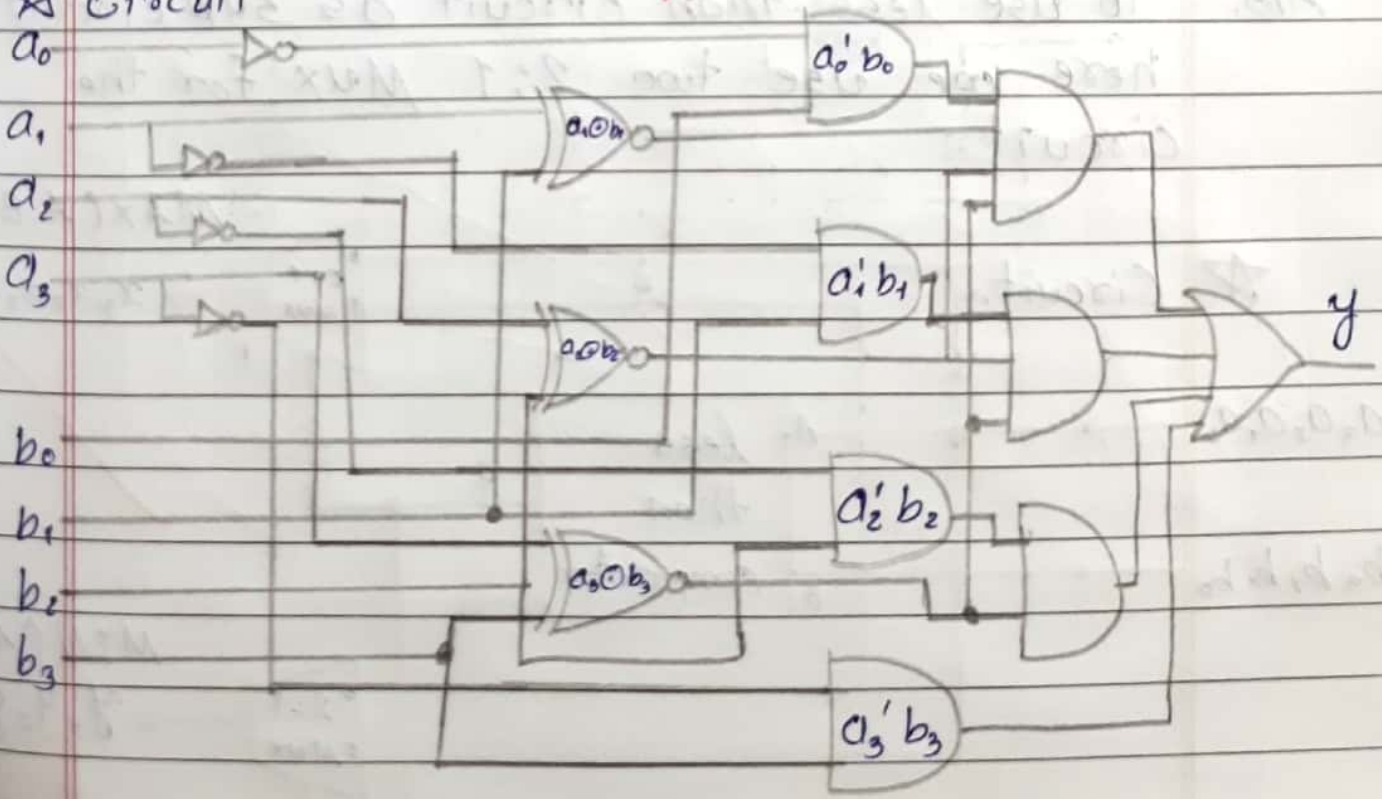
Q.1 a) Design a less than comparator accepting 4-bit unsigned i/p A and B and generating 1-bit output L. If $A < B$, $L = 1$ & else $L = 0$.

Ans. We must compare the 4-bit numbers from MSB to LSB, if $MSB(A) < MSB(B)$ then condition is met & output should be 1.
Or else $MSB(A) = MSB(B)$ then check number upto LSB to get comparison.
Hence eqn for less than operation is:

$$L.T. = \bar{a}_3 b_3 + (a_3 \odot b_3)\bar{a}_2 b_2 + (a_3 \odot b_3)(a_2 \odot b_2) \cdot \bar{a}_1 b_1 + (a_3 \odot b_3)(a_2 \odot b_2)(a_1 \odot b_1)\bar{a}_0 b_0$$

where $A = a_3 a_2 a_1 a_0$ & $B = b_3 b_2 b_1 b_0$

☆ Circuit



☆ Conclusion:

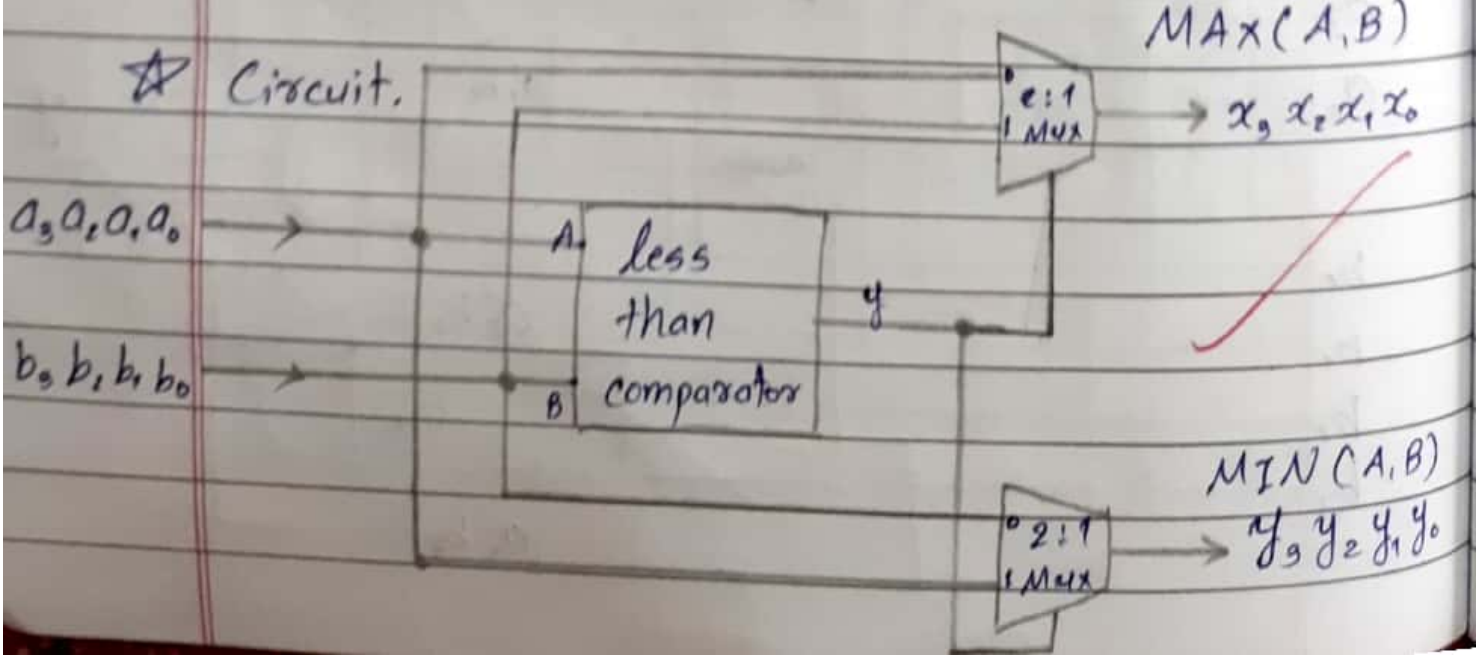
→ Thus we can built circuit for "less than" or "<" operator this way using XNOR, AND, OR & NOT gate.

→ As this circuit is for strictly A < B so if A = B output will be 0.

b) Using the less than comparator as subcircuit, determine MAX(A,B) & MIN(A,B) where MAX(A,B) = Maximum of the unsigned numbers A & B and MIN(A,B) = Minimum of two unsigned numbers A & B.

Ans. To use less than circuit as sub circuit here we use two 2:1 Mux for the circuit.

☆ Circuit.



☆ Conclusion:

Thus we can use less than circuit as sub circuit to build a MAX - MIN Block.

Here when $A < B$, $y = 1$ then input of both 2:1 Mux will be 1 and in that case output of MAX(A,B) will be B & MIN(A,B) will be A. similar for the case when $y = 0$ then in both Muxs Input line 0 will pass as output.

good

30/5/2022

LAB-6

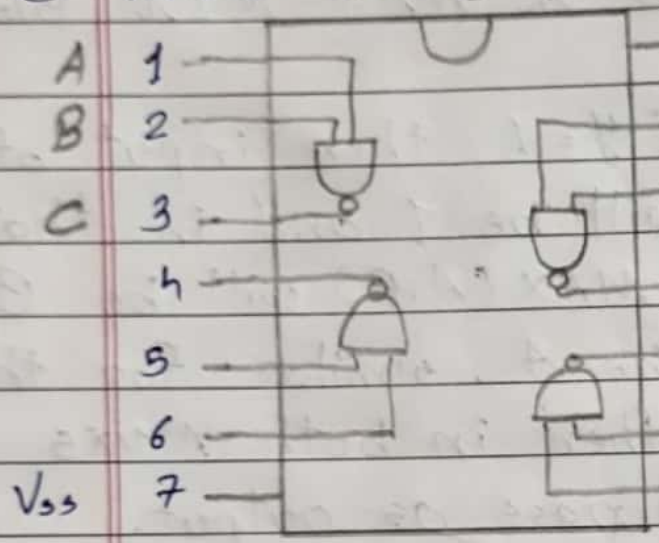
Familiarization with logic gates & controlled one's & two's complement operations on breadboard.

1. All basic & derived logic gates on breadboard

(a) NAND Gate (4011)

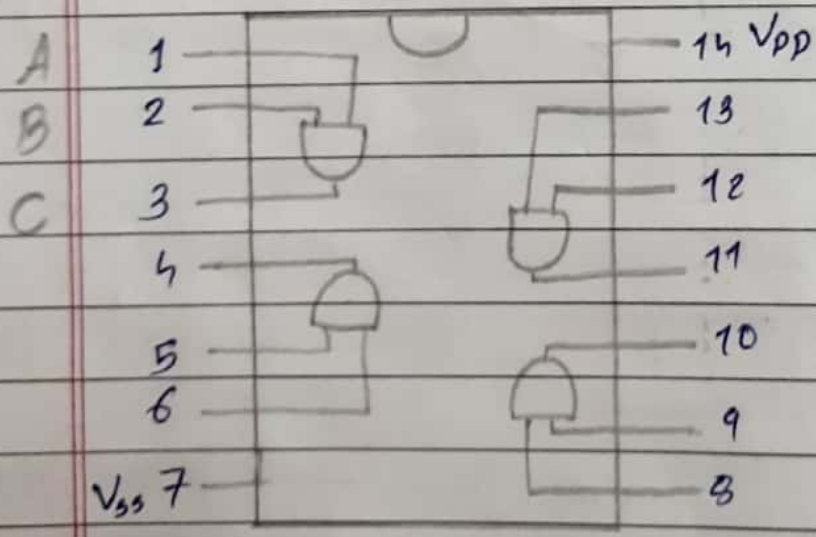
$V_{DD} = +12V$

$V_{SS} = GND$



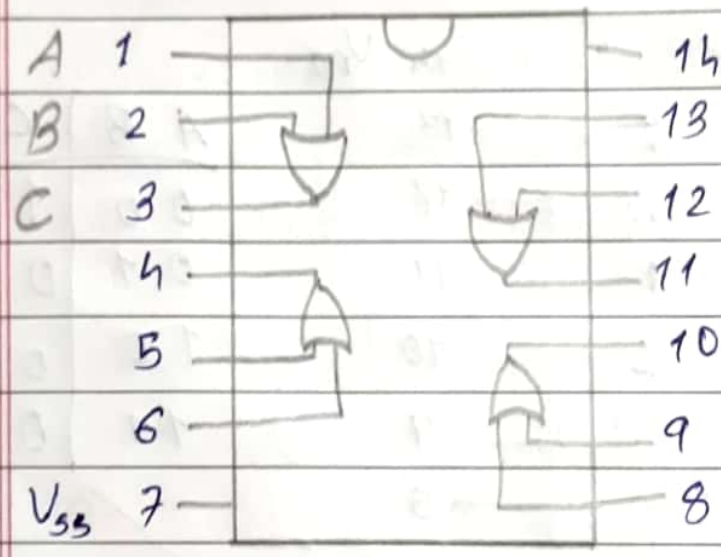
A	B	C
0	0	1
0	1	1
1	0	1
1	1	0

(b) AND Gate (4081)



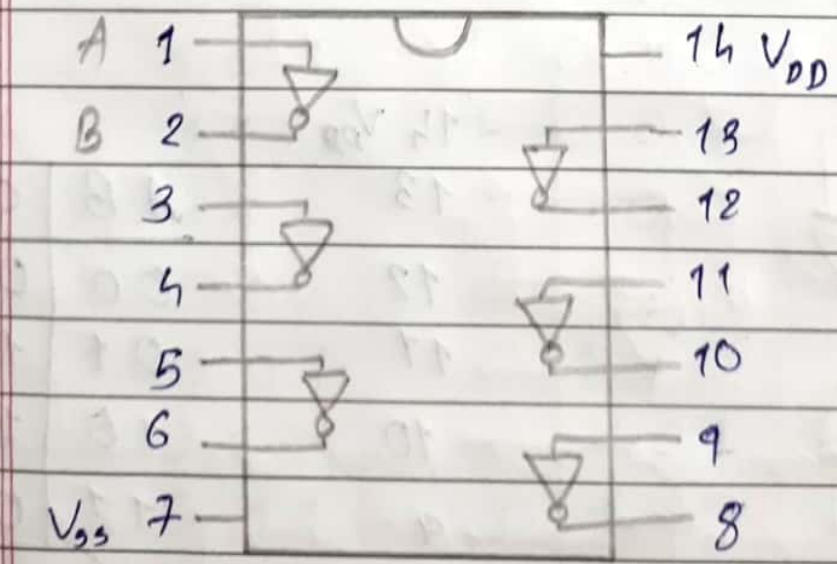
A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

(c) OR Gate (4071)



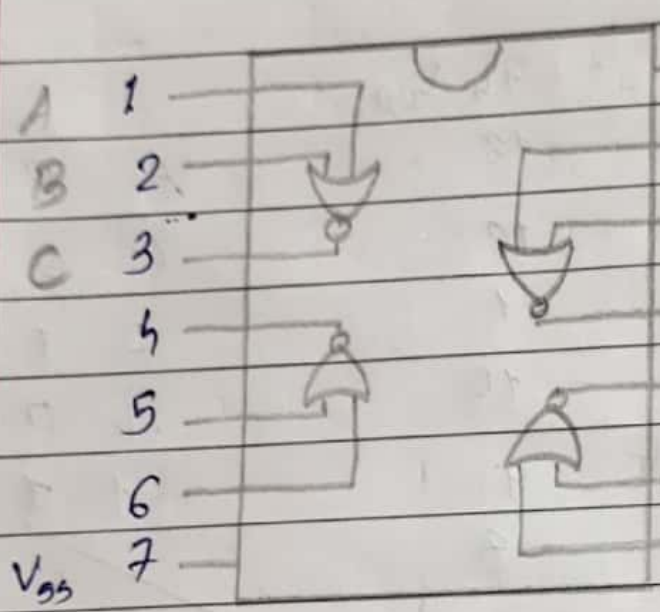
A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

(d) NOT Gate (4069)



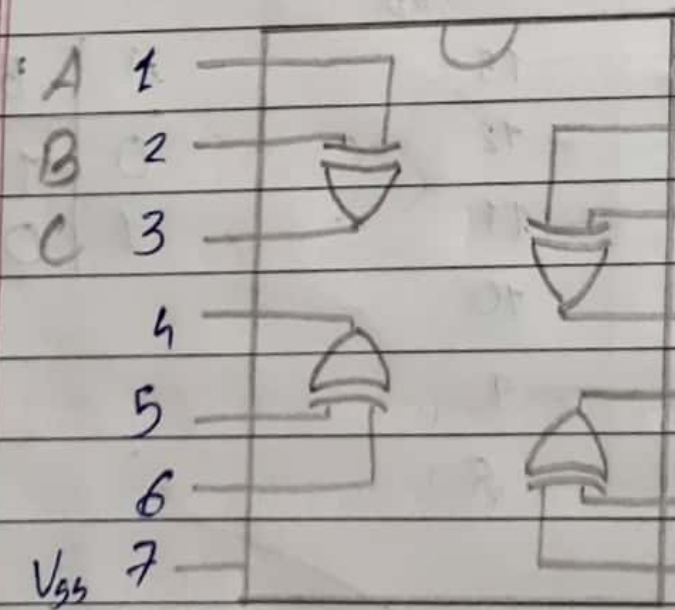
A	B
0	1
1	0

(e) NOR Gate (4001)



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

(f) XOR Gate (4030)



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

2. Design a 4-bit circuit on the breadboard that accepts a 4-bit input & a control c that returns the 1's complement of the input when $C=1$ & 2's complement of the input when $C=0$.

	$C=1$	$C=0$
$a_3\ a_2\ a_1\ a_0$	$b_3\ b_2\ b_1\ b_0$	$b_3\ b_2\ b_1\ b_0$
0 0 0 0	1 1 1 1	0 0 0 0
0 0 0 1	1 1 1 0	1 1 1 1
0 0 1 0	1 1 0 1	1 1 1 0
0 0 1 1	1 1 0 0	1 1 0 1
0 1 0 0	1 0 1 1	1 1 0 0
0 1 0 1	1 0 1 0	1 0 1 1
0 1 1 0	1 0 0 1	1 0 1 0
0 1 1 1	1 0 0 0	1 0 0 1
1 0 0 0	0 1 1 1	1 0 0 0
1 0 0 1	0 1 1 0	0 1 1 1
1 0 1 0	0 1 0 1	0 1 1 0
1 0 1 1	0 1 0 0	0 1 0 1
1 1 0 0	0 0 1 1	0 1 0 0
1 1 0 1	0 0 1 0	0 0 1 1
1 1 1 0	0 0 0 1	0 0 1 0
1 1 1 1	0 0 0 0	0 0 0 1

So expressions,

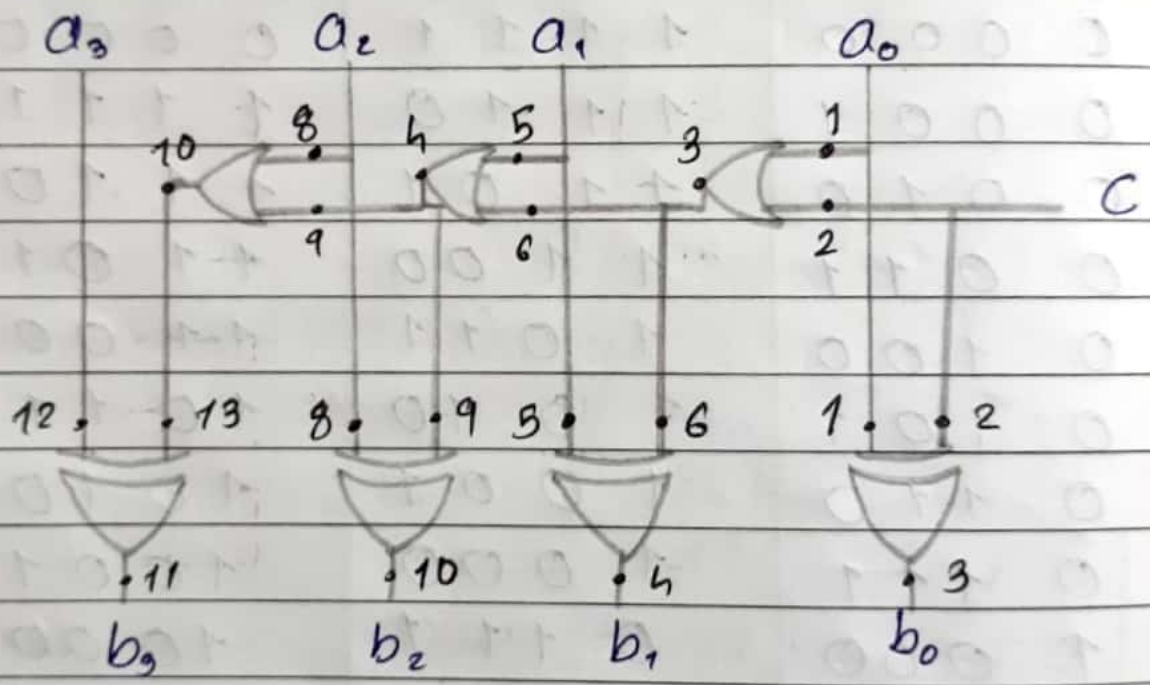
$b_0 = a_0 \oplus C$

$b_1 = a_1 \oplus (a_0 + C)$

$b_2 = a_2 \oplus (a_1 + a_0 + C)$

$b_3 = a_3 \oplus (a_2 + a_1 + C)$

★ Circuit.



So as we can see we have implemented 1's & 2's complementor circuit using XOR & OR gates.

Here we have used IC's 4030 (XOR) & 4071 (OR) to implement circuit where black numbers in diagram shows pin in IC.

LAB. 7 Familiarization with counter operations & its display modes.

Q.1 You shall be given the 4029 IC for 4-bit presettable up/down counter. Study the counter transitions on applications of clock pulses in the up mode & down mode for both binary and decade configurations.

Ans. 4029 IC for Counter

Preset Enable	1	16	VDD
Q_4	2	15	CLOCK
JAM 4	3	14	Q3
JAM 1	4	13	JAM 3
CARRY IN	5	12	JAM 2
Q1	6	11	Q2
CARRY OUT	7	10	UP / DOWN
VSS	8	9	BINARY / DECADE

- In this counter IC,
  there are 4 input nodes - JAM1, JAM2, JAM3, JAM4
  4 output nodes - Q1, Q2, Q3, Q4
- Up/down node where if ip is 1 → up
  ip is 0 → down

If number is in up mode, it will increase output value by 1 on each transition otherwise it will decrease.

→ In binary / decade mode where 0 → binary
1 → decade
IF it is set to binary then counter will return values from 0 to 15 otherwise will return values from 0 to 9.

→ Preset pin is for whether we want to give counter some initial value or not.

→ clock pin would be connected to T2 output of clock for increasing manner.

2. Observe the output of the counter using seven segment display in up and down mode for both binary and decade configurations, by appending BCD correction circuit.

Ans. When we give output of counter circuit Q1, Q2, Q3 & Q4 as input in seven segment display in up mode values increase by 1 as we increase clock & in down mode value decrease by 1 as we top clock.

→ In binary configuration output shows 0-9 & don't care values for 10 - 15 & in decade configuration it shows value from 0-9 & after increasing again it will show value from 0-9.

3. Convert the output of the 4 bit binary counter to gray code.

Truth table.

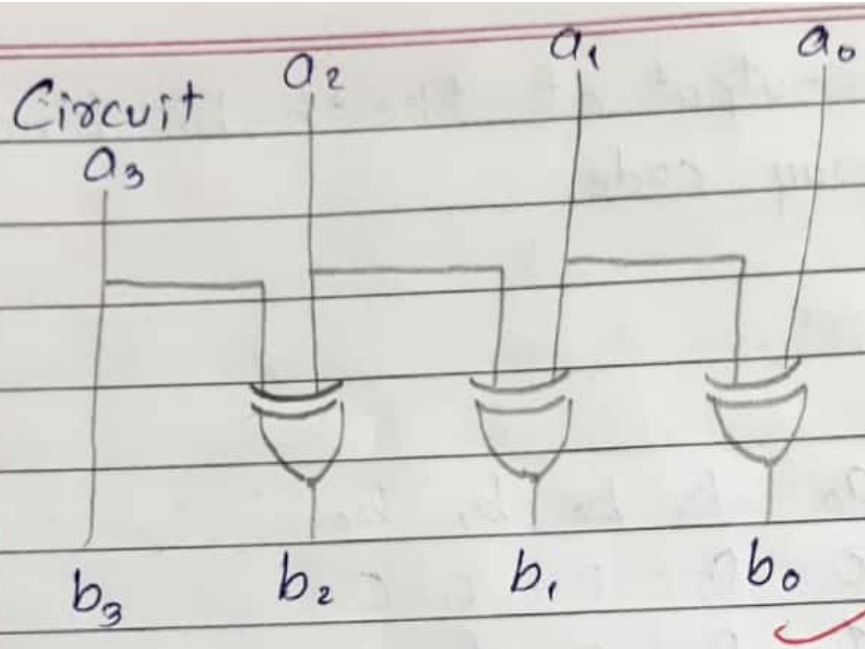
a_3	a_2	a_1	a_0	b_3	b_2	b_1	b_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

$b_3 = a_3$

$b_2 = a_2 \oplus a_3$

$b_1 = a_1 \oplus a_2$

$b_0 = a_0 \oplus a_1$



Good

20/8/20

Here we will give output of counter circuit as input & get output as gray code.

Ans. 2

LAB - 8

Logic expression :-

$$P = Q_3Q_2 + Q_3Q_1$$

when $P = 1$, add 0110 (i.e. 6)
$P = 0$, add 0000 (i.e 0)

Also, $S_0 = Q_1$

$S_1 = P \oplus Q_1$

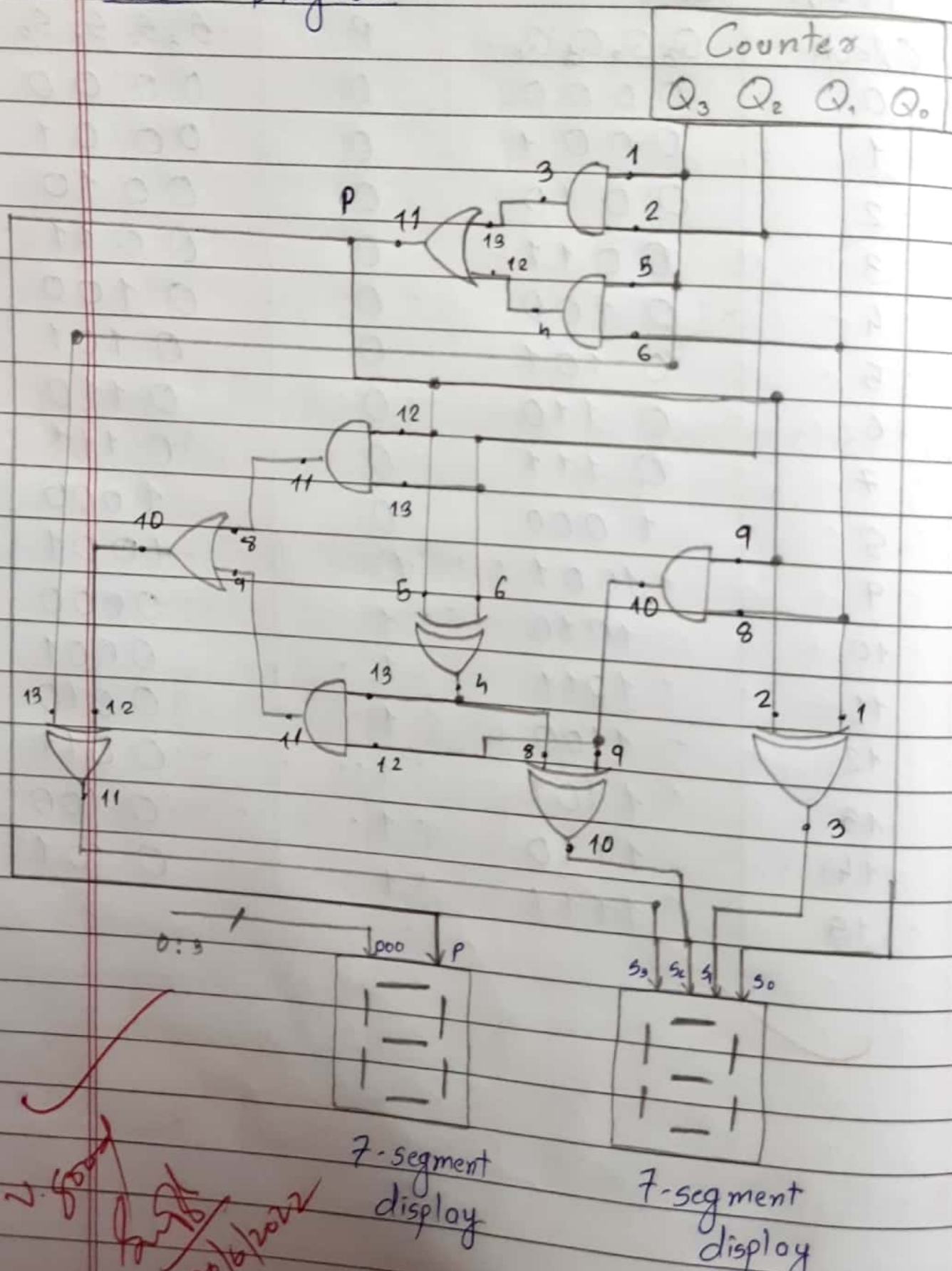
$S_2 = (P \oplus Q_2) \oplus (P \cdot Q_1)$

$S_3 = Q_3 \oplus (((P \oplus Q_2) \cdot (P \cdot Q_1)) + P \cdot Q_2)$

Truth table

Clock	$Q_3 Q_2 Q_1 Q_0$	P	$S_3 S_2 S_1 S_0$
0	0 0 0 0	0	0 0 0 0
1	0 0 0 1	0	0 0 0 1
2	0 0 1 0	0	0 0 1 0
3	0 0 1 1	0	0 0 1 1
4	0 1 0 0	0	0 1 0 0
5	0 1 0 1	0	0 1 0 1
6	0 1 1 0	0	0 1 1 0
7	0 1 1 1	0	0 1 1 1
8	1 0 0 0	0	1 0 0 0
9	1 0 0 1	0	1 0 0 1
10	1 0 1 0	1	0 0 0 0
11	1 0 1 1	1	0 0 0 1
12	1 1 0 0	1	0 0 1 0
13	1 1 0 1	1	0 0 1 1
14	1 1 1 0	1	0 1 0 0
15	1 1 1 1	1	0 1 0 1

☆ Circuit Diagram:



20/6/2022

LAB-9

Synthesis and Simulation of Full Adder Design

☆ Module Adder:

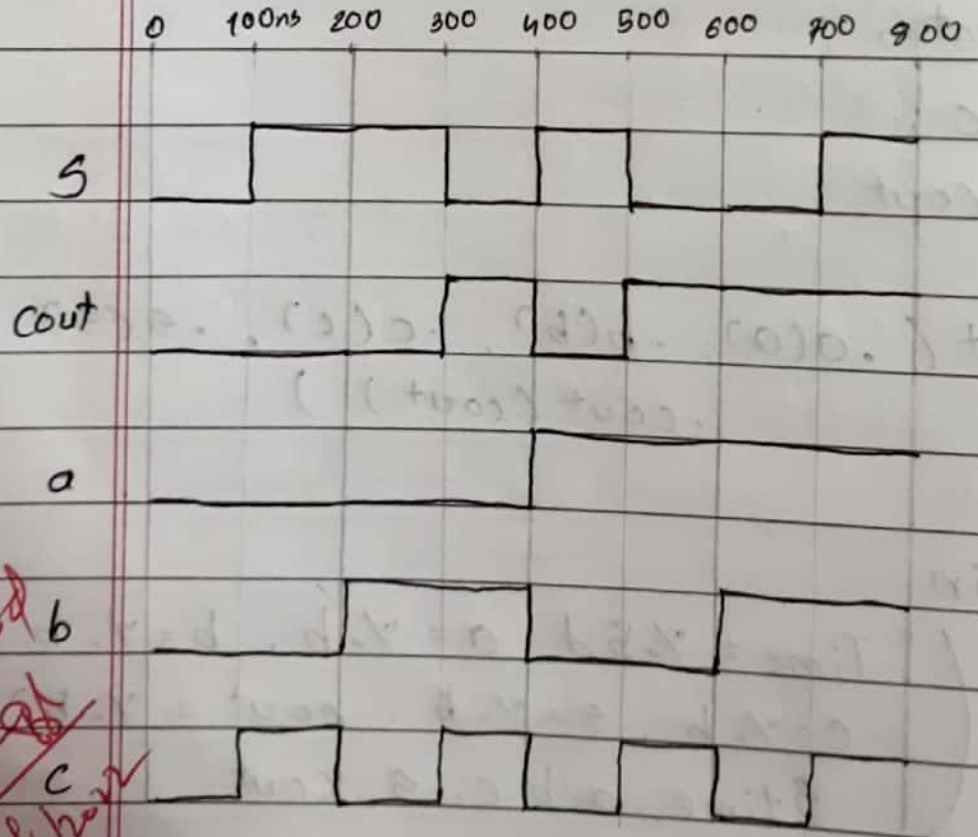
```
Module fulladder(
        input a, b, c,
        output s, cout
        );
assign s = a ^ b ^ c;
assign cout = (a & b) | (b & c) | (a & c);

end module :
```

☆ Verilog Testbench:-

```
module tester;
reg a, b, c;
wire s, cout;

fulladder uut(.a(a), .b(b), .c(c), .s(s),
              .cout(cout))

initial begin
$monitor("Time = %5d, a = %b, b = %b,
          c = %b, s = %b, cout = %b\n",
          $time, a, b, c, s, cout

#100; a=0; b=0; c=0;
```

```
#100; a=0; b=0; c=1;
#100; a=0; b=1; c=0;
#100; a=0; b=1; c=1;
#100; a=1; b=0; c=0;
#100; a=1; b=0; c=1;
#100; a=1; b=1; c=0;
#100; a=1; b=1; c=1;

$finish;
end
```

Graphical Waveform



LAB-10

Synthesis & Simulation of Barrel Shifter

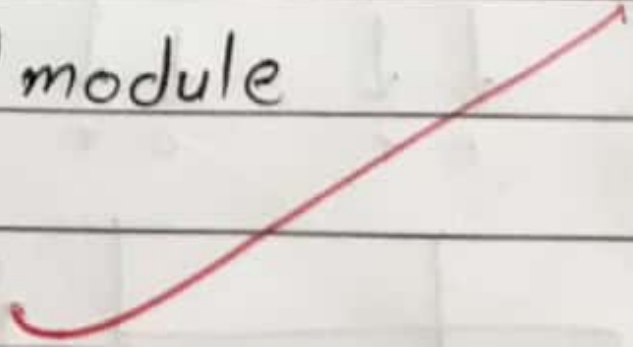
☆ 4 bit full adder

```
module Adder_4bit (
    input [3:0] A,
    input [3:0] B,
    input cin,
    output cout,
    output [3:0] sum
    );

wire [3:1] t;

FullAdder FA0 (.s(sum[0]), .co(t[1]), .a(A[0]),
              .b(B[0]), .c(cin));
FullAdder FA1 (.s(sum[1]), .co(t[2]); .a(A[1]),
              .b(B[1]), .c(t[1]));
FullAdder FA2 (.s(sum[2]), .co(t[3]),
              .a(A[2]), .b(B[2]), .c(t[2]));
FullAdder FA3 (.s(sum[3]), .co(cout), .a(A[3]),
              .b(B[3]), .c(t[3]));

endmodule
```

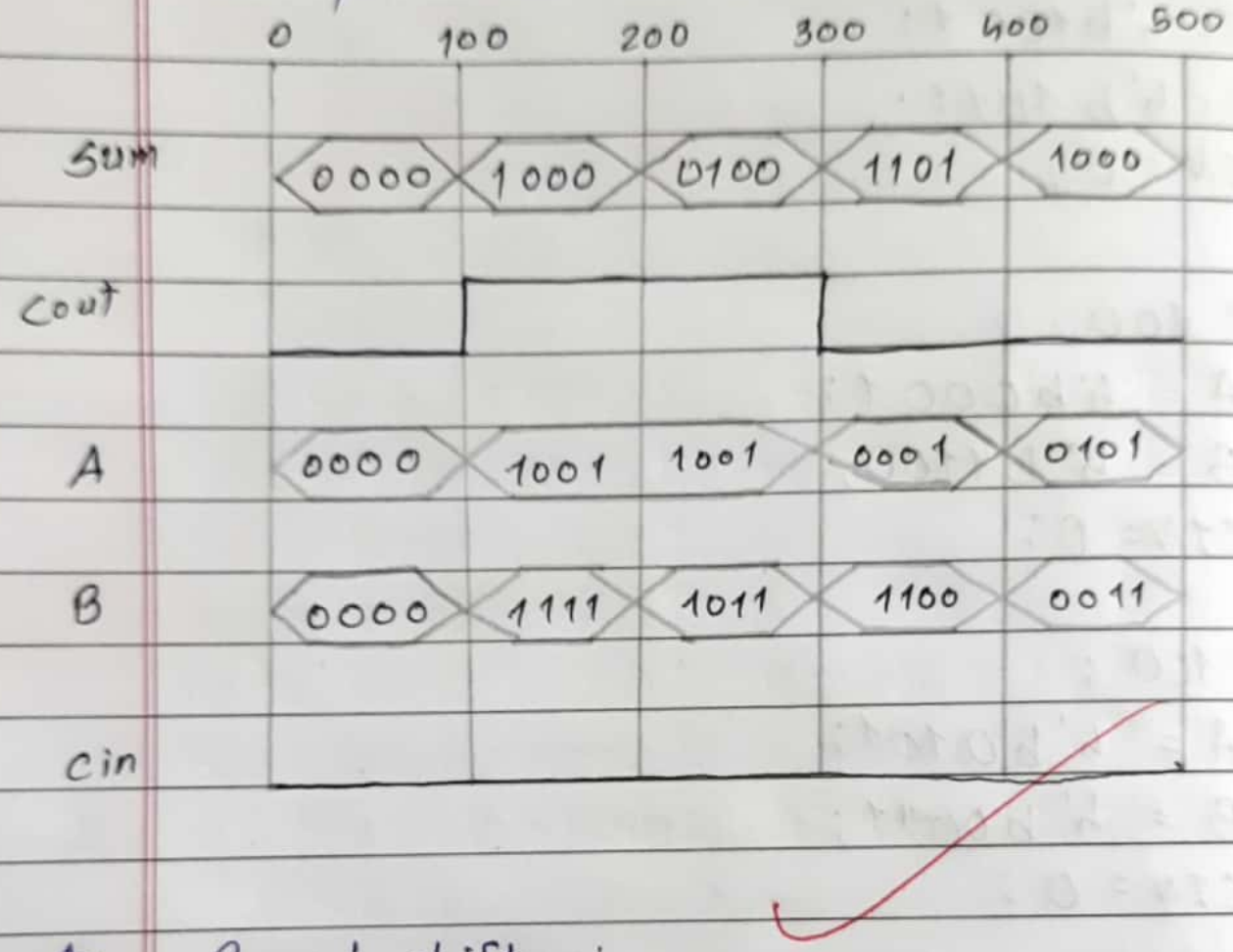
mod Verilog Testbench :-

```
module tester2i
  reg [3:0] A;
  reg [3:0] B;
  reg CIN;

  wire [3:0] sum;
  wire cout;

rca_4bit uut(
          .A(A), .B(B),
          .CIN(CIN),
          .sum(sum),
          .cout(cout)
         );

initial begin
    A = 4'b0000;
    B = 4'b0000;
    CIN = 0;

#100;
 A = 4'b1001;
B = 4'b1111;
CIN = 0;
```

```
#100;
A = 4'b1001;
B = 4'b1011;
CIN = 0;

#100;
A = 4'b0001;
B = 4'b1100;
CIN = 0;

#100;
A = 4'b0101;
B = 4'b0011;
CIN = 0;

end
endmodule
```

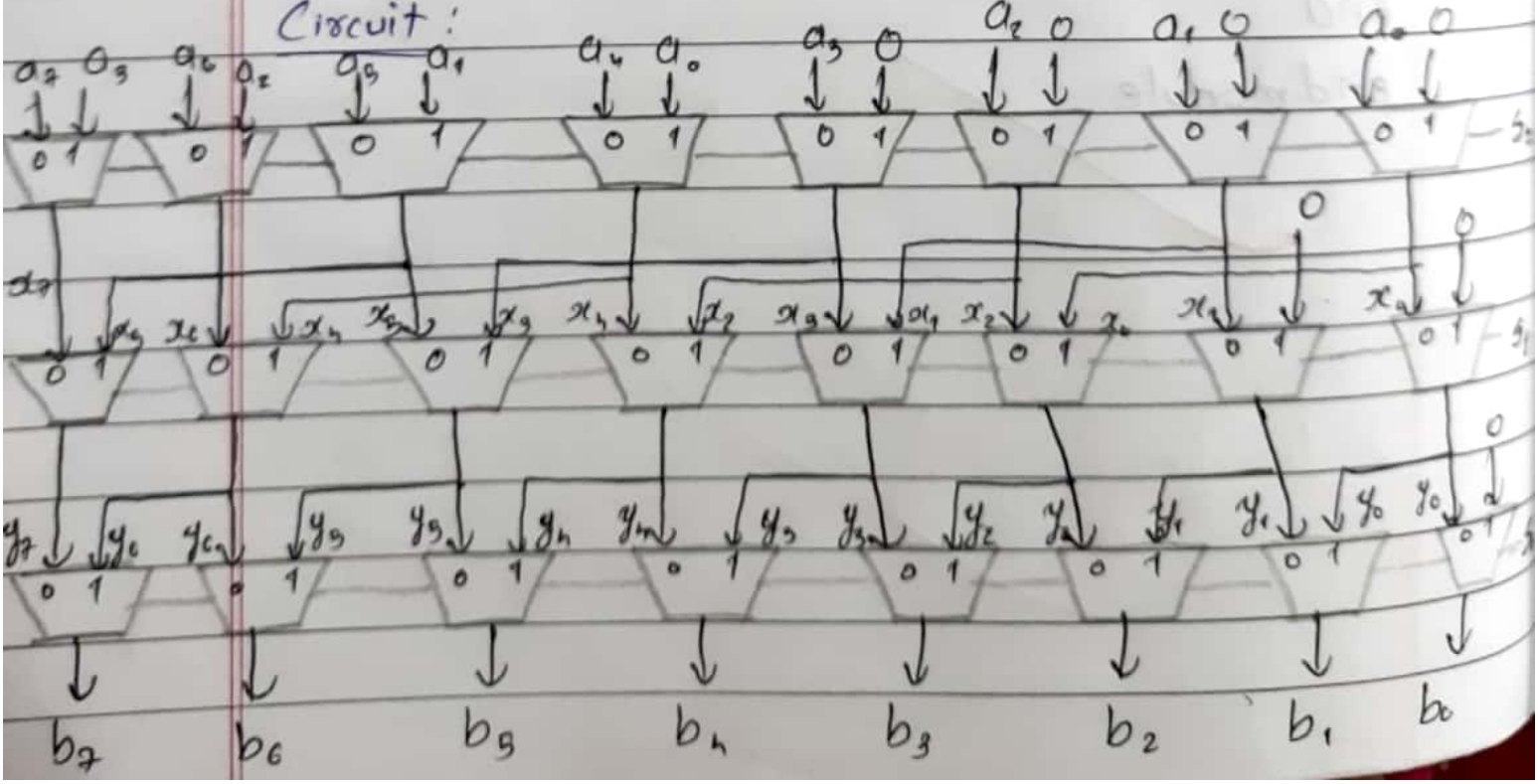


Graphical waveform



☆ Barrel shifter:

Circuit:



Verilog Code

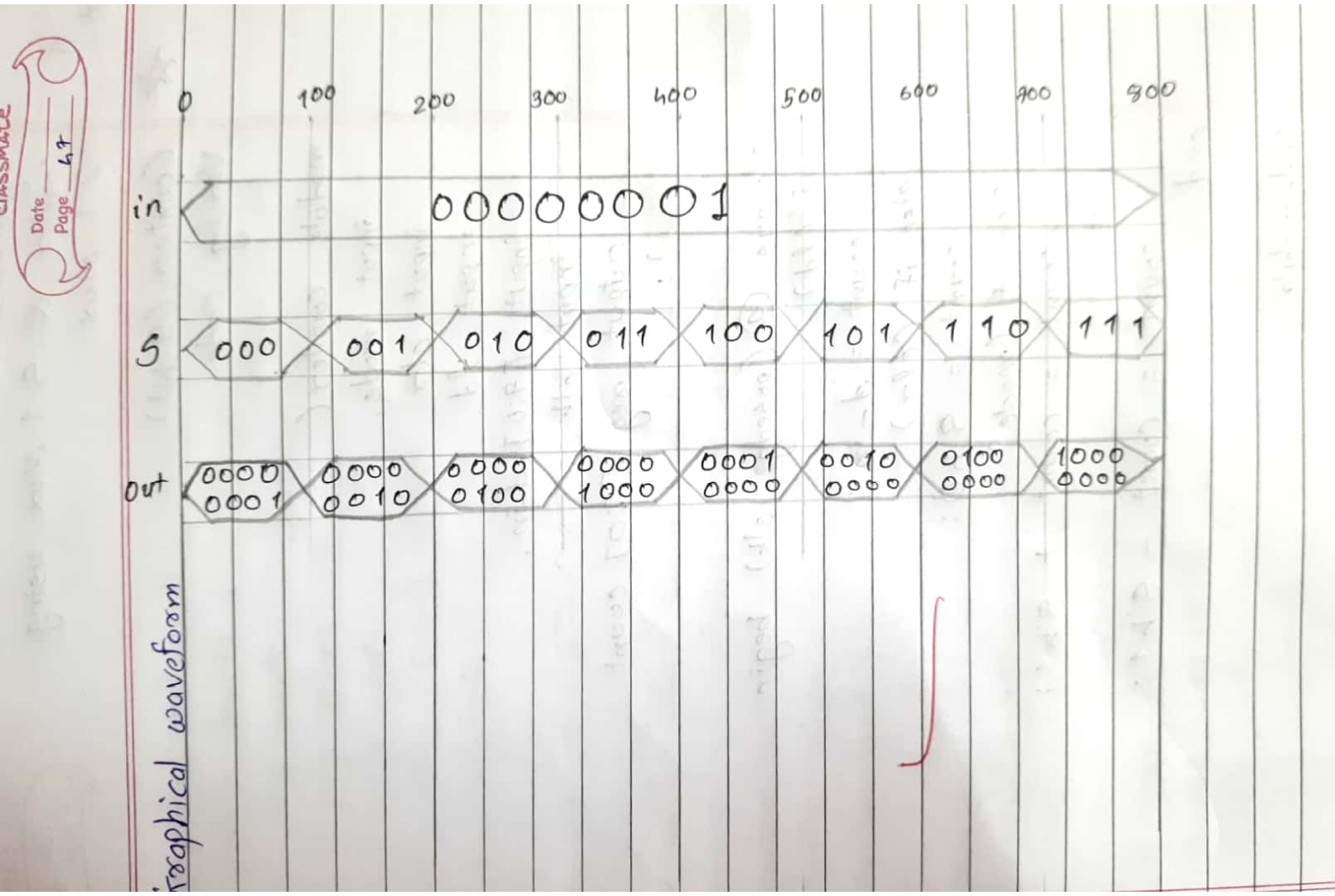
```
module barrel_shifter_8bit (
        input [7:0] in,
        output [7:0] out,
        input [2:0] S,
        );
wire [7:0] x, y;

assign x = (S[2] == 1'b1) ? {in[3:0], {4{1'b0}}} : in;
assign y = (S[1] == 1) ? {x[5:0], {2{1'b0}}} : x;
assign out = (S[0] == 1) ? {y[6:0], {1'b0}} : y;

endmodule
```

Verilog testbench

```
barrel_shifter_8bit uut (
        .in(in),
        .out(out),
        .S(S)
        );
```

```
initial begin
    in = 0;
    s = 0;

 # 100;

in = 8'b00000001; s = 3'b001; #100;
in = 8'b00000001; s = 3'b010; #100;
in = 8'b00000001; s = 3'b011; #100;
in = 8'b00000001; s = 3'b100; #100;
in = 8'b00000001; s = 3'b101; #100;
in = 8'b00000001; s = 3'b110; #100;
in = 8'b00000001; s = 3'b111; #100;
$finish;

end
```

29/6/2022



Graphical waveform

LAB 11

Synthesis & Simulation
Counter & 8:1 Mux using
2:1 Mux

☆ Counter (8 bit)
Verilog code.

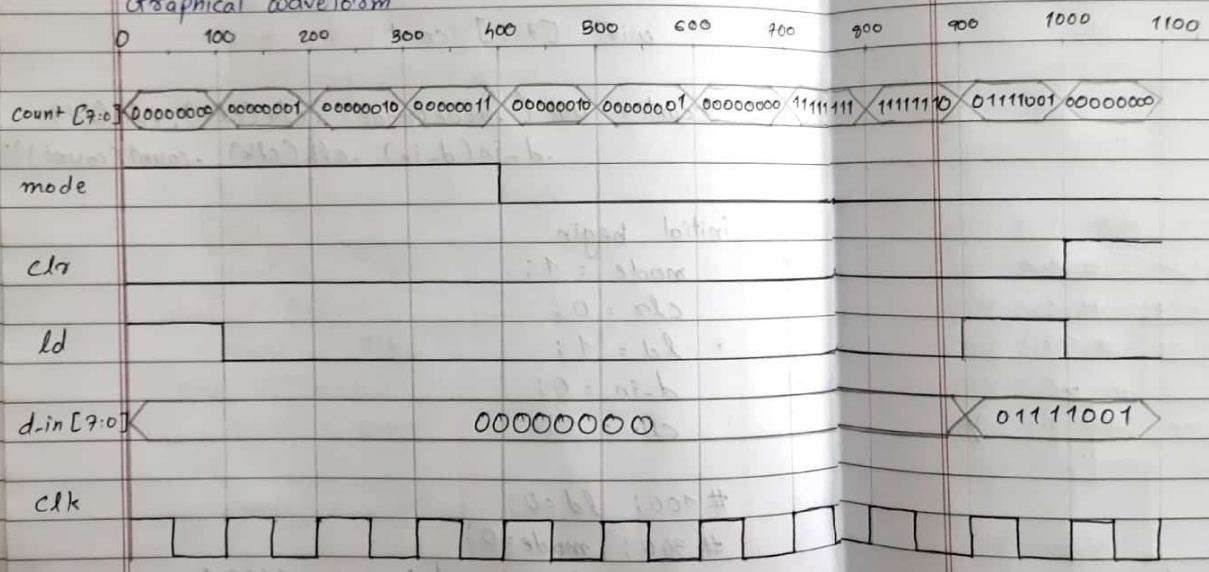
```
module counter(
    input mode,
    input clr,
    input ld,
    input [7:0] d_in,
    input clk,
    output reg [7:0] count
    );

always @(posedge clk) begin
  if(ld)
      count = d_in;
  else if (clr)
      count = 8'b0;
  else if (mode)
      count = count + 8'b1;
  else
      count = count - 8'b1;
end

endmodule
```

Verilog testbench :

```
module tester ;
    reg mode ;
    reg clr ;
    reg ld ;
    reg [7:0] d_in ;
    reg clk ;

    wire [7:0] count ;

counter uut ( .mode(mode), .clr(clr), .ld(ld),
              .d_in(d_in), .clk(clk), .count(count));

initial begin
    mode = 1 ;
    clr = 0 ;
    ld = 1 ;
    d_in = 0 ;
    clk = 1 ;

#100 ; ld = 0 ;
#300 ; mode = 0 ;
#500 ; ld = 1 ; d_in = 10011001 ;
#100 ; ld = 0 ; clr = 1 ;
#100 ; $finish ;
 end
```

```
always begin
#50;
clk = ~clk;
end

endmodule
```

Graphical waveform



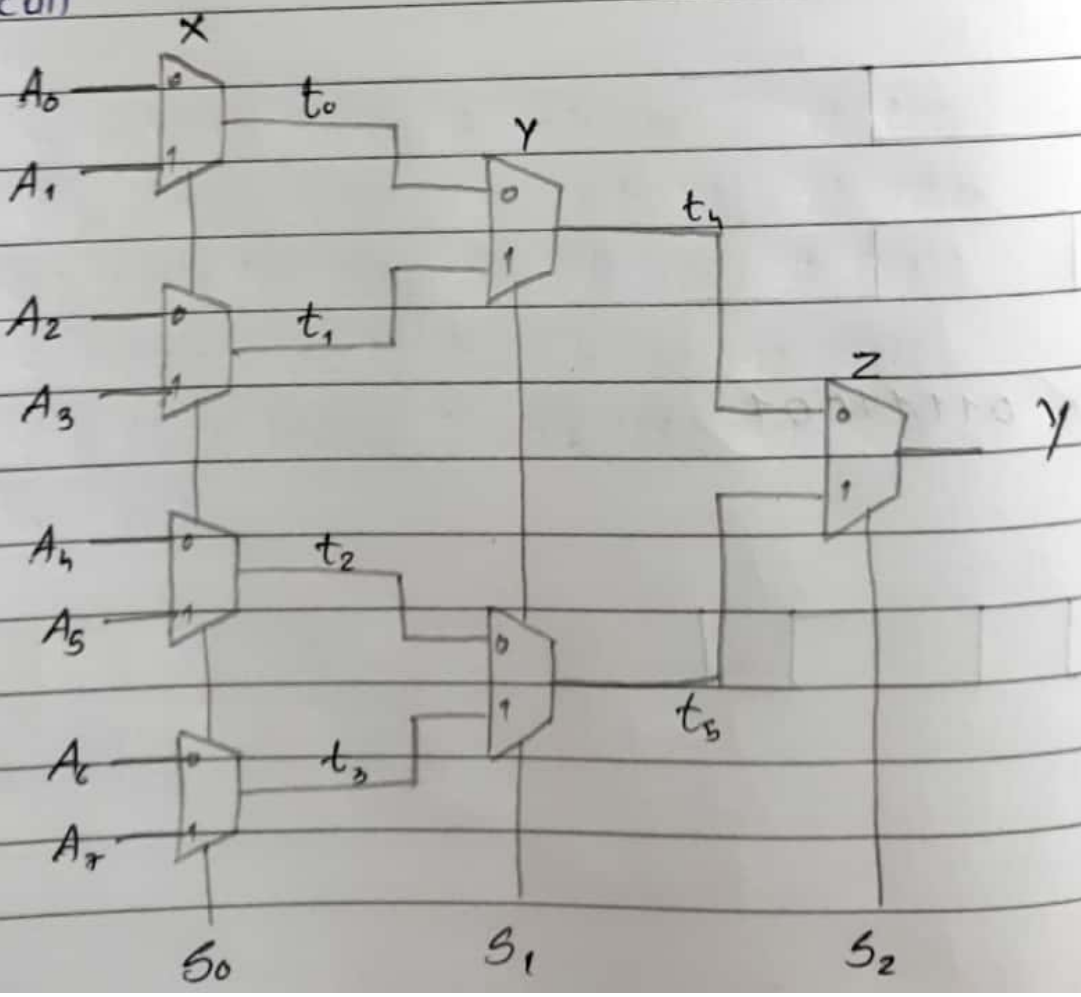
★ 8:1 MUX using 2:1 MUX

⇒ 2:1 MUX

Verilog code.

```
module MUX_2_1(
    input a, input b,
    input s, output y );

assign y = s ? b : a ;

endmodule
```

Circuit



Note: Rather than taking 8 variables I have used 8 bit array & used its bits as input.

=> 8:1 MUX

Verilog code

```
module MUX_8_1(
    input [7:0] A,
    input [2:0] S,
    output Y
    );
wire [5:0] t;

MUX_2_1 MUX_x1(.a(A[0]), .b(A[1]), .s(S[0]), .y(t[0]));
MUX_2_1 MUX_x2(.a(A[2]), .b(A[3]), .s(S[0]), .y(t[1]));
MUX_2_1 MUX_x3(.a(A[4]), .b(A[5]), .s(S[0]), .y(t[2]));
MUX_2_1 MUX_x4(.a(A[6]), .b(A[7]), .s(S[0]), .y(t[3]));
MUX_2_1 MUX_y1(.a(t[0]), .b(t[1]), .s(S[1]), .y(t[4]));
MUX_2_1 MUX_y2(.a(t[2]), .b(t[3]), .s(S[1]), .y(t[5]));
MUX_2_1 MUX_z1(.a(t[4]), .b(t[5]), .s(S[2]), .y(Y));

endmodule
```

Verilog testbench

```
module tester_MUX_8_1;
    reg [9:0] A;
    reg [2:0] S;

    wire Y;

    MUX_8_1 uut (.A(A), .S(S), .Y(Y));

initial begin
    A = 0;
    S = 0;

#100;
A = 8'b0000 1000; S = 3'b000; #100;
A = 8'b 0000 1000; S = 3'b011; #100;
A = 8'b 000 10000; S = 3'b 100; #100;
A = 8'b 00000001; S = 3'b 000; #100;
A = 8'b 1000 1000; S = 3'b 111; #100;
$ finish;

end

endmodule
```

Graphical waveform.