# Lab 8

## To Practice Binary to Gray Code & BCD to Excess-3 Conversion

***Note: You may draw all the logic diagrams with hand and paste the pictures here. Also, the conversions in the tasks can be done by hand if doing it in soft seems difficult to you.***

***Use Logicly software with your name, roll number & section mentioned in your workspace. Make sure that all of your connections are clearly visible and distinguishable. In logicly, use "text" label to point out/show all your inputs & outputs.***

### Task 1

a) **Convert the given binary codes into respective gray codes. Mention each step while converting the codes.**

1111, 1101,1001, 010, 011, 110

Task # 1

(a) Convert given binary Codes into gray codes.

(i) 1111

Soln:

⇒ 1 1 1 1 ⇒ "Put Same element for gray code (MSB) and do Ex-OR with another element"

1000 Ans

∴ $g_3 = b_3$
$g_2 = b_3 \oplus b_2$
$g_1 = b_2 \oplus b_1$
$g_0 = b_1 \oplus b_0$ → Steps

(ii) 1101

Soln:

1 1 0 1

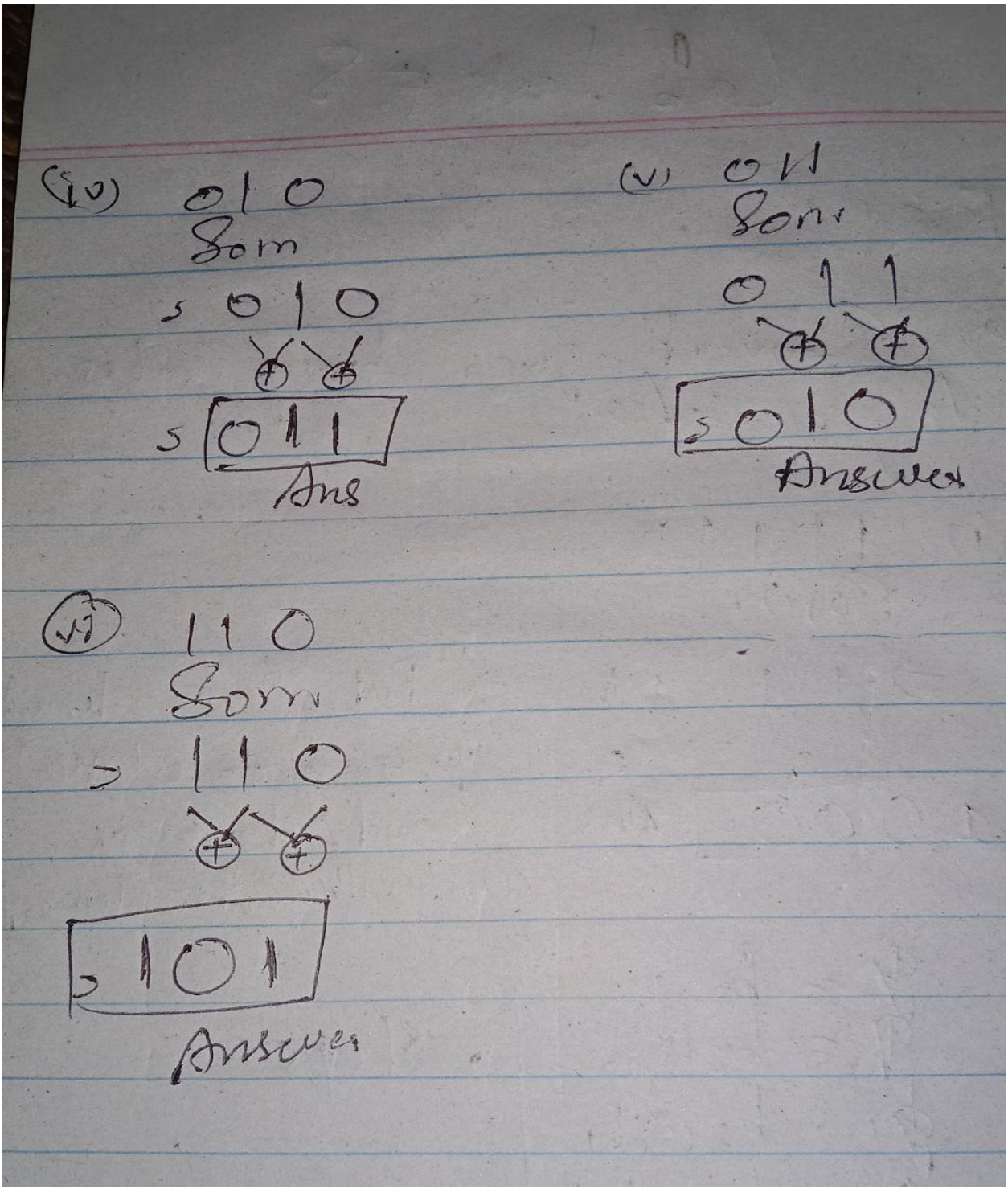
= 1011 → Ans

(iii) 1001

Soln

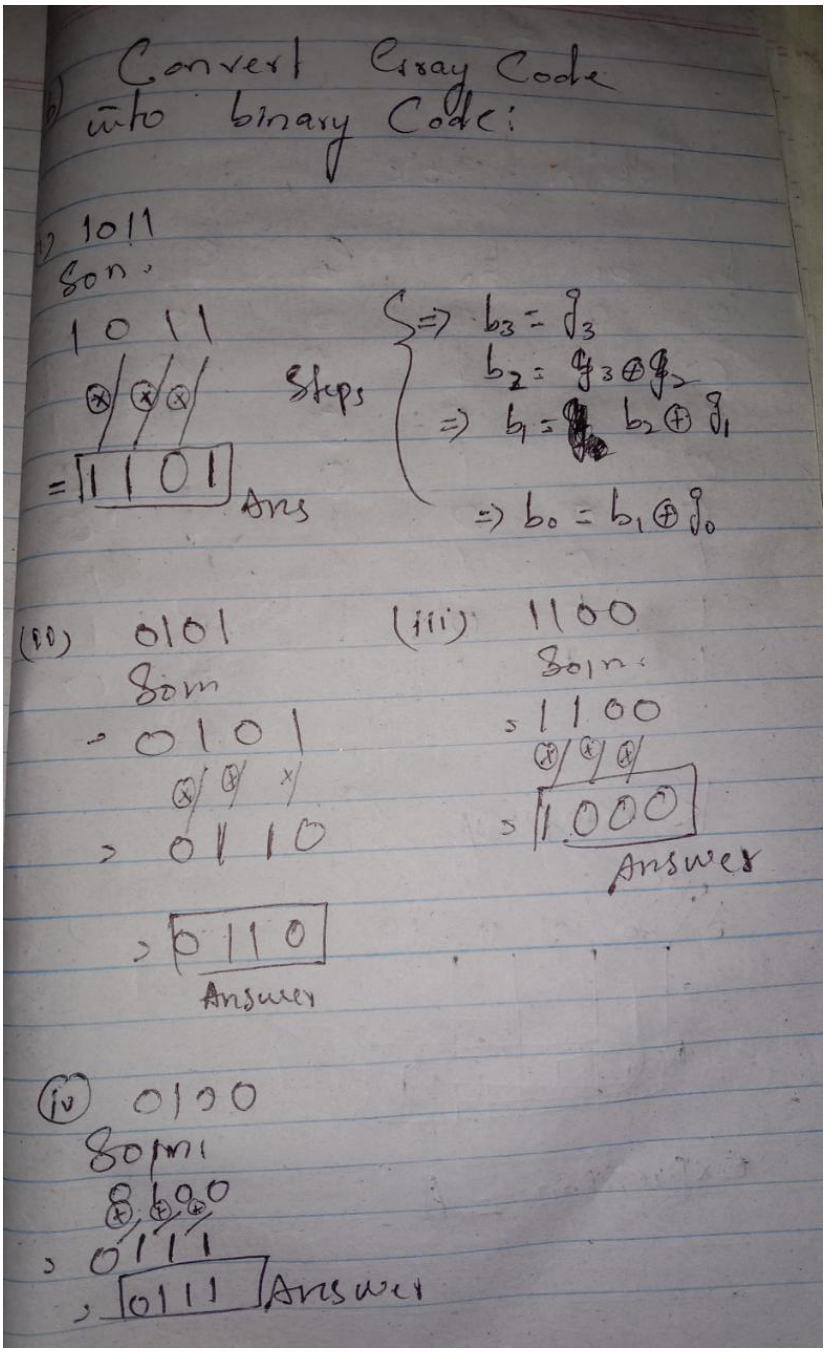
> 1001

> 1101 → Ans




**b) Convert the given gray codes into respective binary codes. Mention each step while converting the codes.**

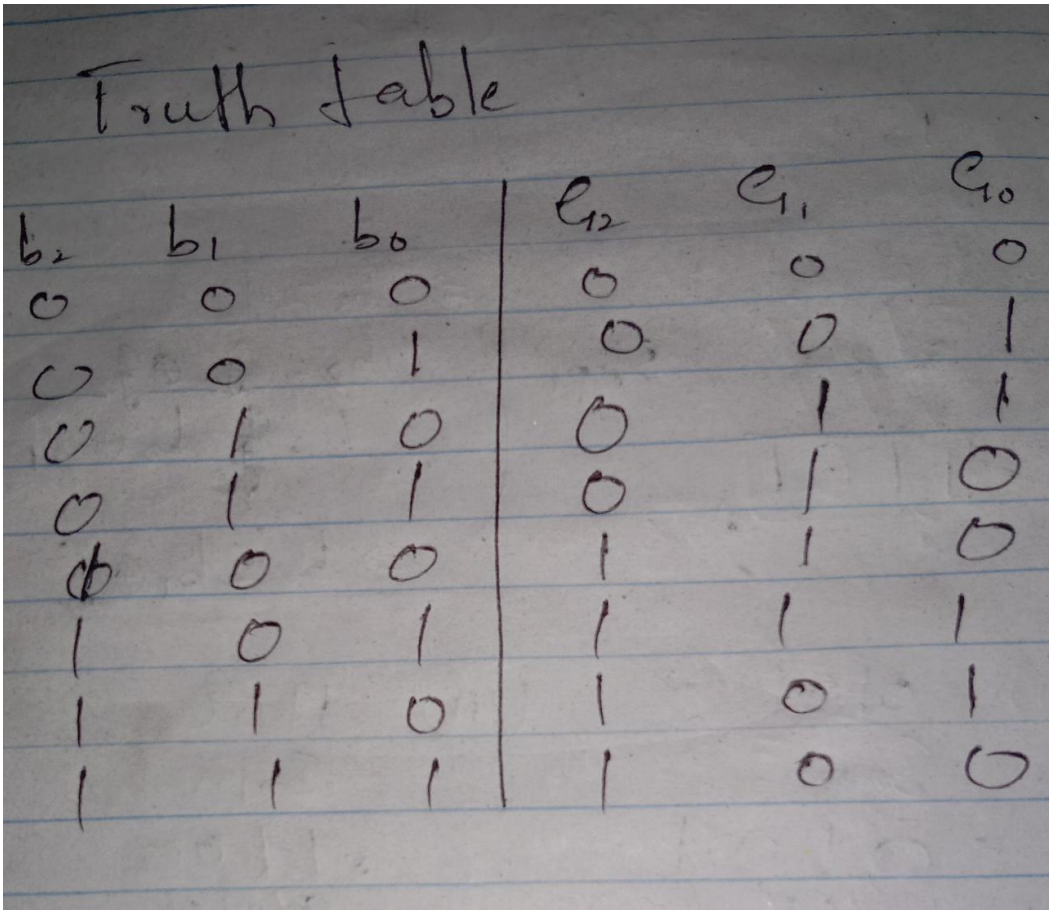
1011, 0101, 1100, 0100


Convert Gray Code into binary Code:
(i) 1011
Soln:
1 0 1 1
Steps
= 1 1 0 1
Ans
⇒ b3 = g3
b2 = g3 ⊕ g2
⇒ b1 = b2 ⊕ g1
⇒ b0 = b1 ⊕ g0
(ii) 0101
Soln
= 0 1 0 1
= 0 1 1 0
= 0 1 1 0
Answer
(iii) 1100
Soln:
= 1 1 0 0
= 1 0 0 0
Answer
(iv) 0100
Soln:
= 0 1 1 1
= 0 1 1 1 Answer

**Task 2**

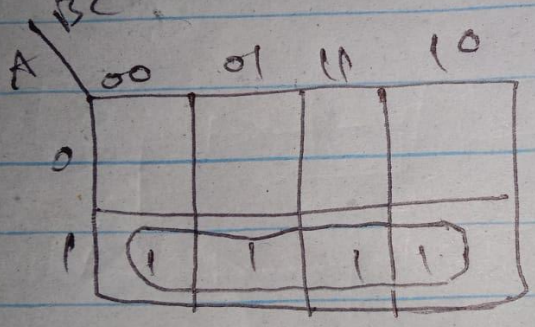
**Devise a truth table for 3-bit Binary to Gray code converter. Write simplified logical expressions and simulate the logic diagram in Logicly software to verify your results. Show KMap simplification as well.**

a) Truth Table



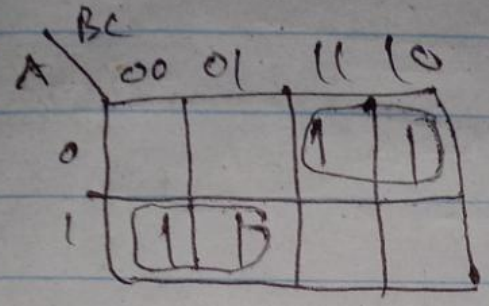

b) Boolean Expression (Simplified using KMap)

K-map of G1



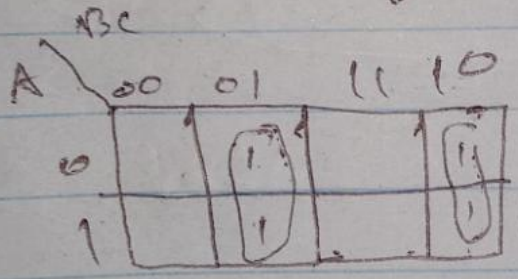

Expression = A

K-map of $C_1$




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

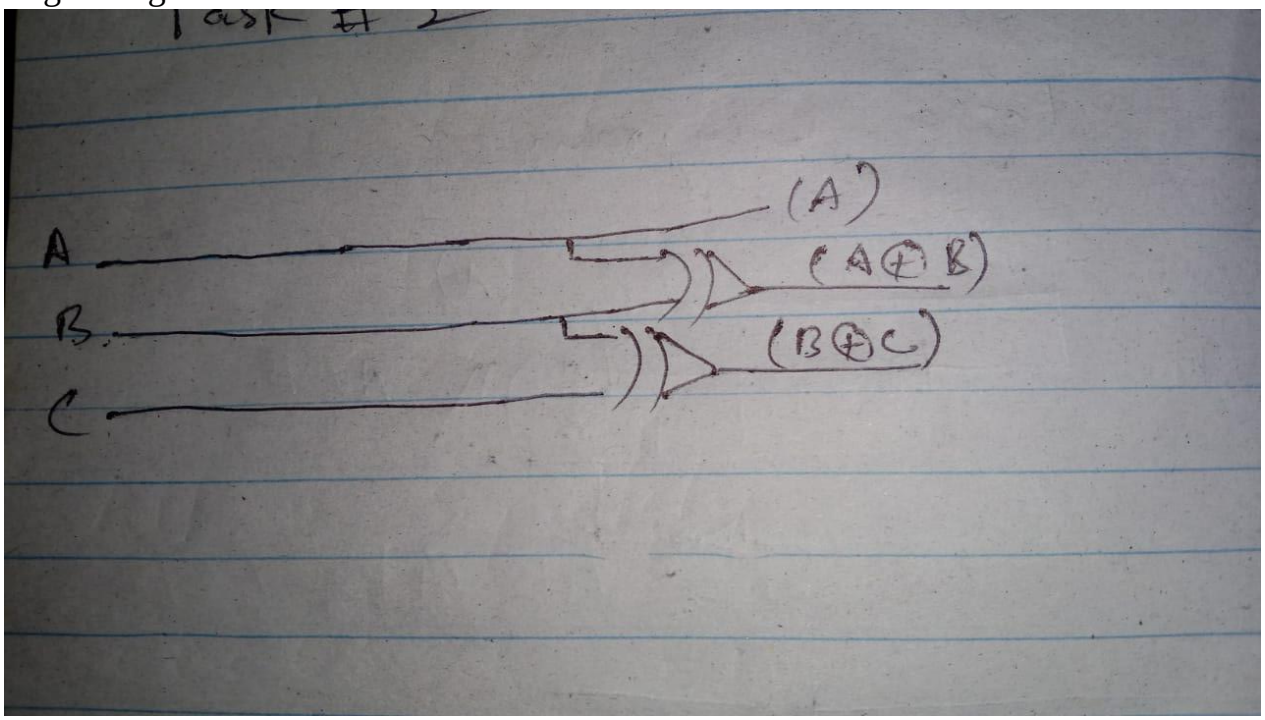
K-map of $C_0$




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

$= B \oplus C$

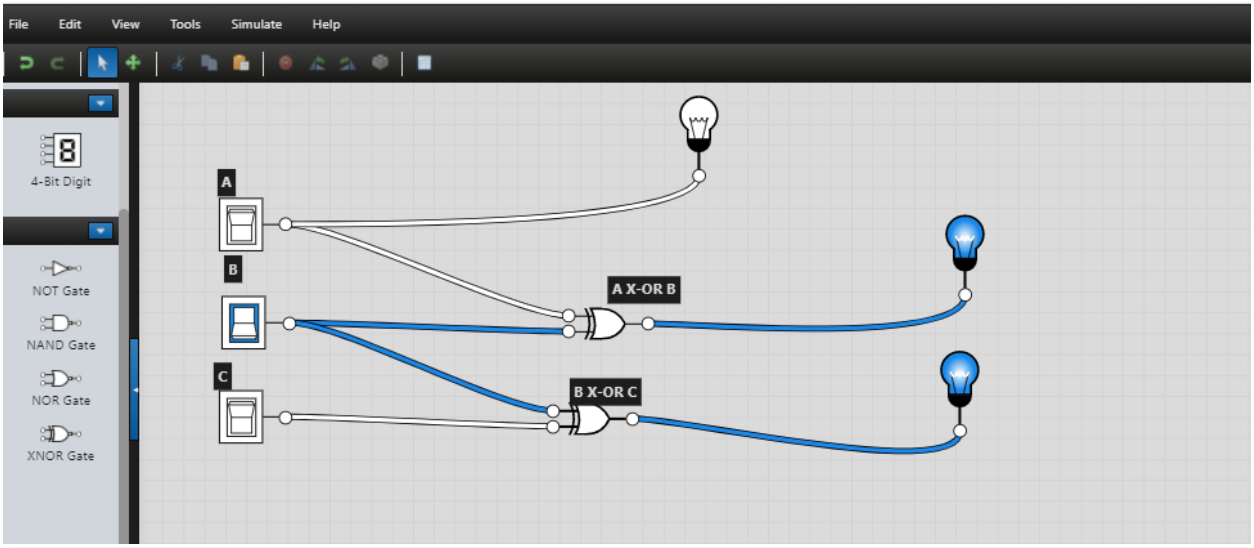
c) Logic Diagram




a) Software Simulation (Show here your results for a few combinations to verify the circuit)

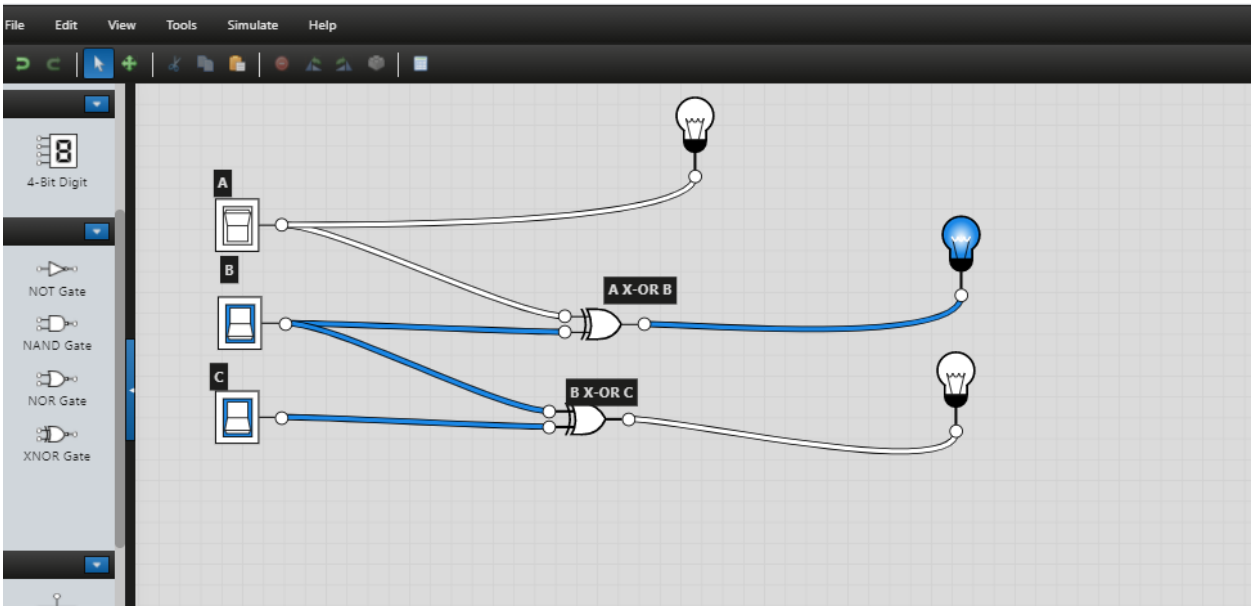
https://logic.ly/demo/Muzamil/p200108/2D-1

Gmail YouTube Maps Flex | Student Google Account WhatsApp SLATE : My Worksp... C++ Lab Assignme... Programiz: Learn to...




https://logic.ly/demo/Muzamil/p200108/2D-1

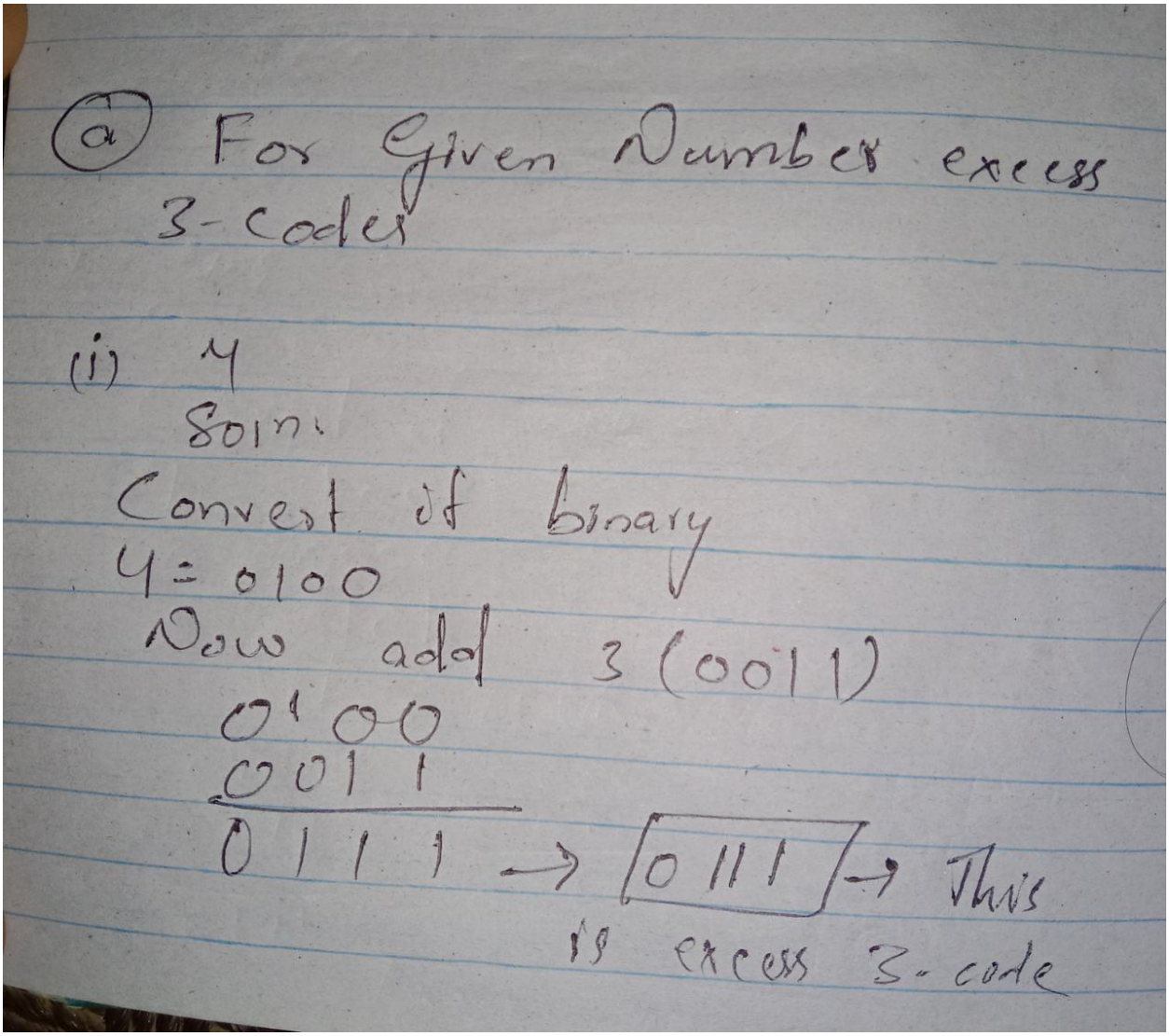
mail YouTube Maps Flex | Student Google Account WhatsApp SLATE : My Worksp... C++ Lab Assignme... Programiz: Learn to...



**Task 3**

a) **For the given numbers, find excess-3 codes. Mention all the steps of conversion.**

4, 9, 25, 50, 250, 405.



9

(i) Soln.

Convert it into BCD

9 = 1001

Now add 3

= 1001

+0011

1100 → 1100 → This is excess-3

(ii) 25

Soln.

~~So~~ Convert it into BCD (seperately)

25 =

2 : 0010

5 : 0101

Now add (+3) Seperately

2 → 0010 + 0011 → 0101

5 → 0101 + 0011 → 1000

→ 01011000 → This is excess-3

(iv) 50

Same

Convert it into BCD (Seprately)

50

⇒ 5 = 0101

0 = 0000

Add 3 Seprately

```
 0000        0101
+0011       +0011
------      ------
 0011        1000
```

= [0011 1000] Answer

(v) 405

Same

Convert it into BCD (Seprately)

⇒ 4 = 0100

0 = 0000

5 = 0101

= (Add 3 with All three)

```
 0100        0000        0101
+0011       +0011       +0011
------      ------      ------
 0111        0011        1000
```

= [0111 0011 1000] Answer

**b)** Devise a truth table for BCD to Excess-3 conversion (as discussed in the lecture). Write simplified expressions and simulate the logic circuit in Logicly software to verify your truth table. You may paste screenshots of a few combinations from the truth table to check your circuit.
Show your KMap simplification as well.

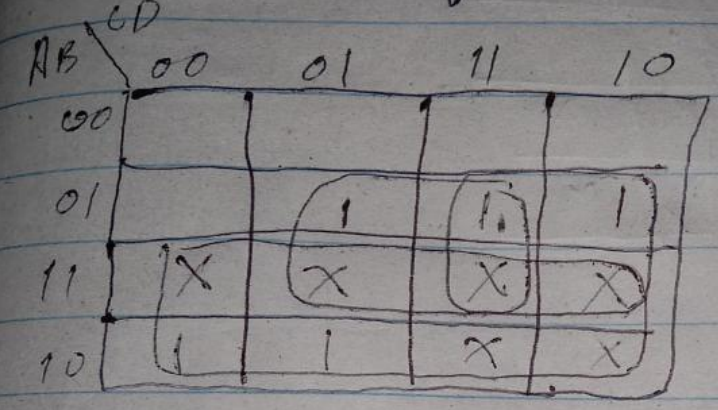
BCD to Excess-3 Converter

b) Truth Table

| BCD | | | | Excess-3 | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | X | X | X | X |
| 1 | 0 | 1 | 1 | X | X | X | X |
| 1 | 1 | 0 | 0 | X | X | X | X |
| 1 | 1 | 0 | 1 | X | X | X | X |
| 1 | 1 | 1 | 0 | X | X | X | X |
| 1 | 1 | 1 | 1 | X | X | X | X |

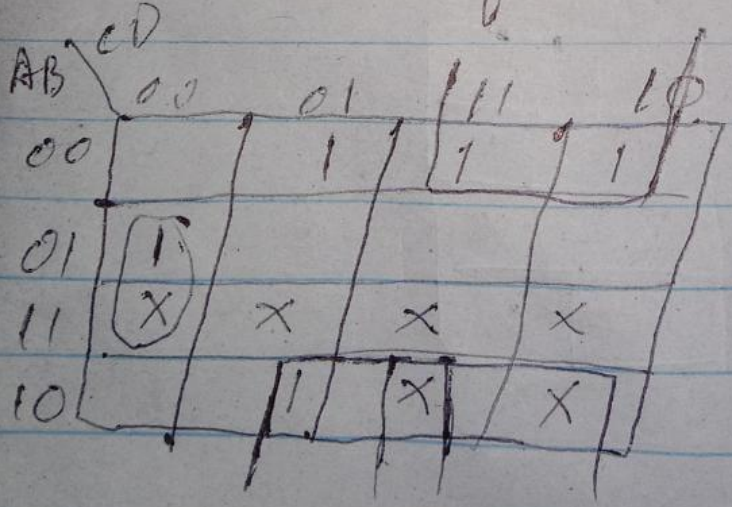
c) Boolean Expression (Simplified Using KMap)

K-map of w



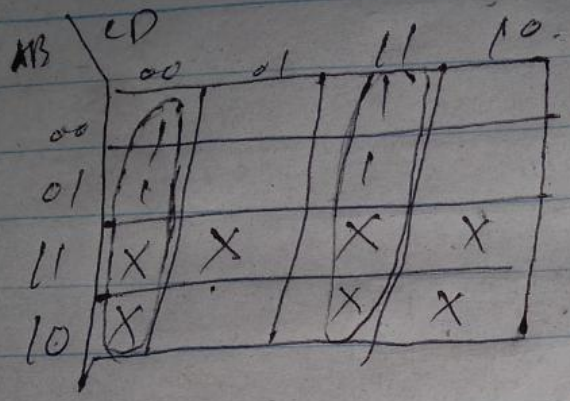

$w = A + BD + Bc$

K-map of x



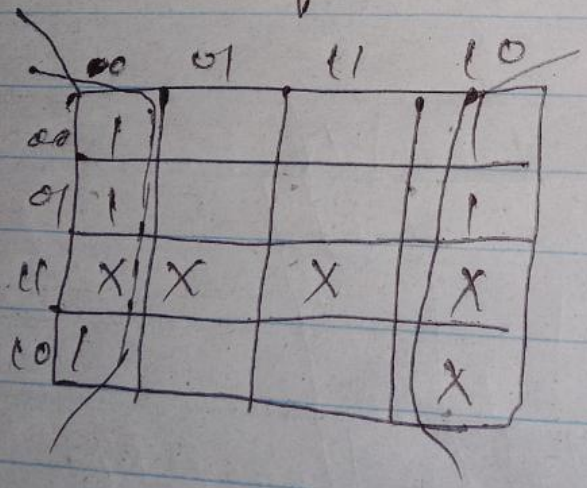

$X = \bar{B}D + \bar{B}c + B\bar{C}\bar{D}$

K- map of Y



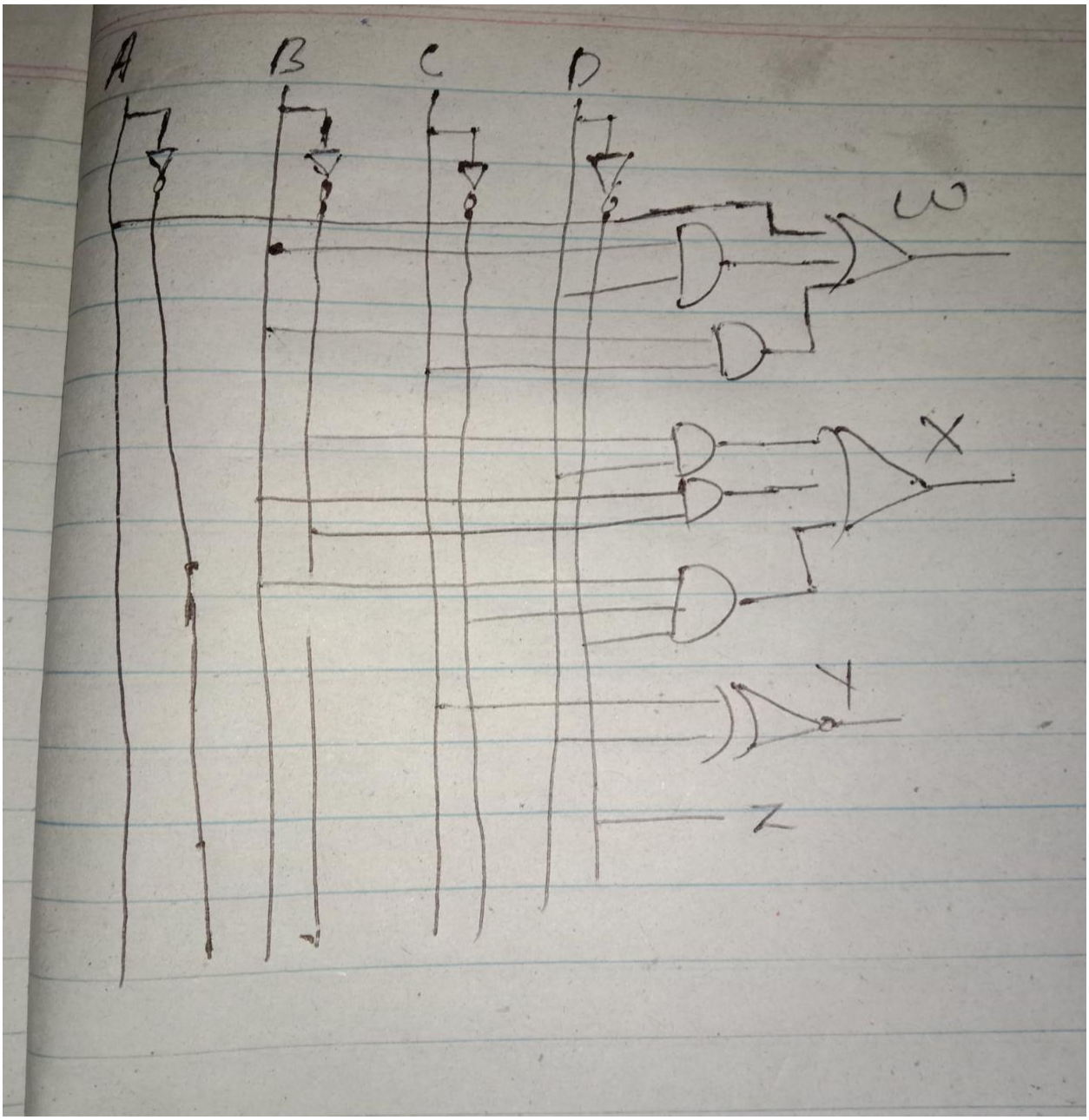

$Y = \bar{C}\bar{D} + CD$

K- map for Z

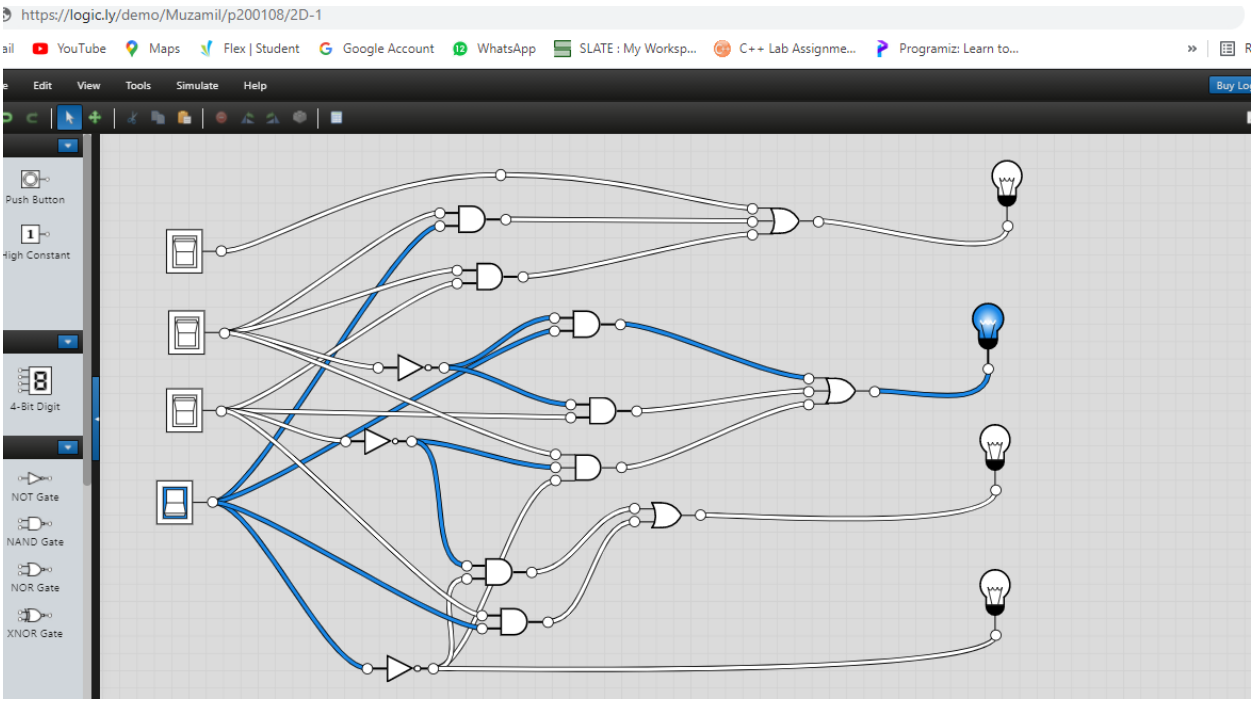


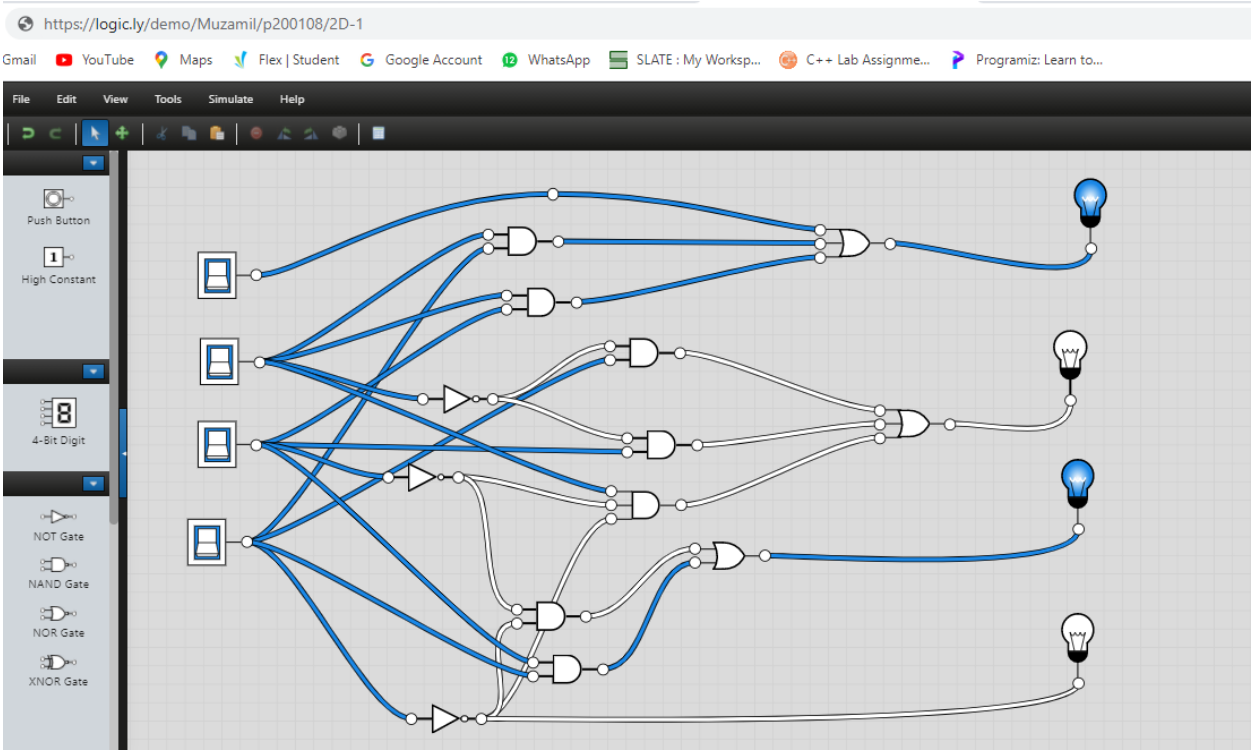

$Z = \bar{D}$

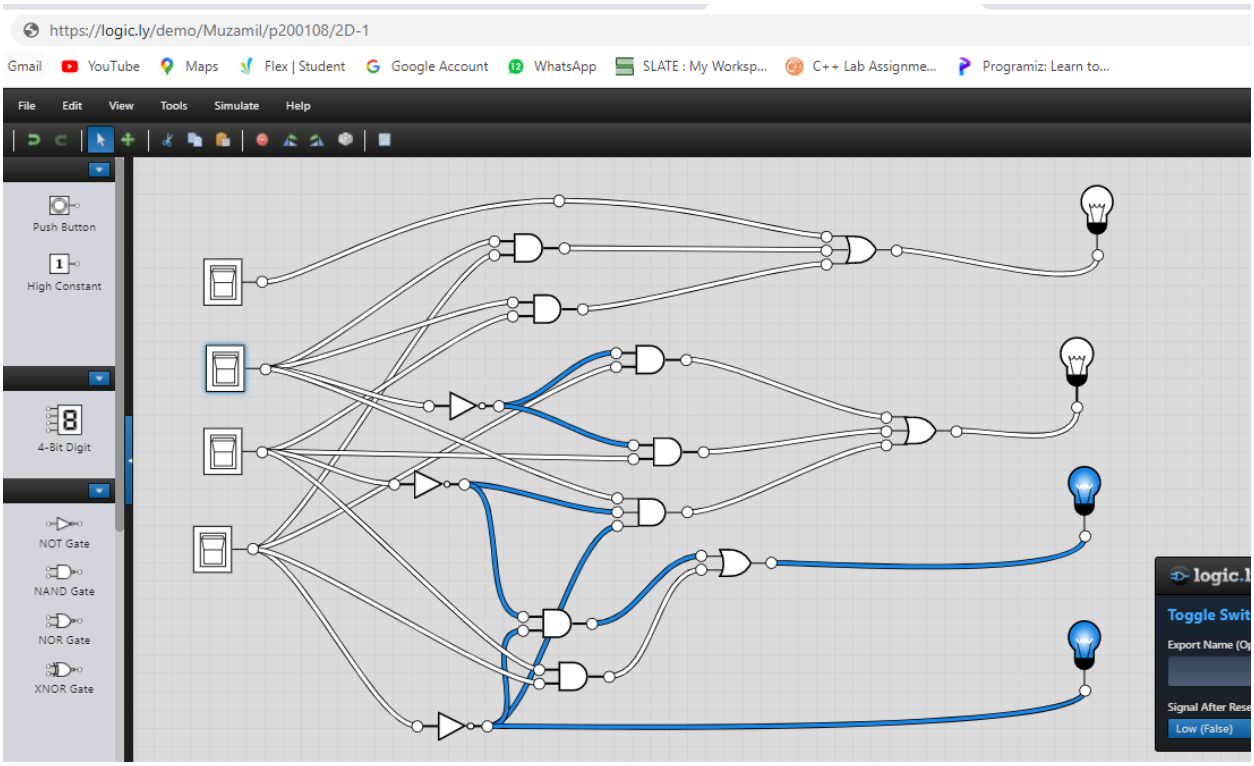
d) Logic Diagram




e) Software Simulation (Show here your results for a few combinations to verify the circuit)


https://logic.ly/demo/Muzamil/p200108/2D-1
YouTube
Maps
Flex | Student
Google Account
WhatsApp
SLATE : My Worksp...
C++ Lab Assignme...
Programiz: Learn to...
Edit
View
Tools
Simulate
Help
Push Button
High Constant
4-Bit Digit
NOT Gate
NAND Gate
NOR Gate
XNOR Gate


https://logic.ly/demo/Muzamil/p200108/2D-1
Gmail
YouTube
Maps
Flex | Student
Google Account
WhatsApp
SLATE : My Worksp...
C++ Lab Assignme...
Programiz: Learn to...
File
Edit
View
Tools
Simulate
Help
Push Button
High Constant
4-Bit Digit
NOT Gate
NAND Gate
NOR Gate
XNOR Gate




# Muzamil

# P20-0108

# 2D-1