**Wenzhou-Kean University Group Assignment-2 Fall 2021**

**CPS 2390 W\_\_ Computer Organization & Architecture   
Name: \_Wang Xinyu\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID Number: \_\_\_\_1098648\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

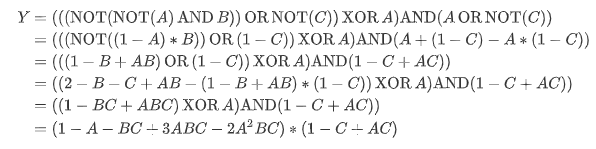
**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID Number: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID Number: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# Problem 1 (0.5 Mark)

1. Draw the logic circuit corresponding to the following logic expression. Use only 2- input AND gates, 2- input OR gates, 2-input XOR gate and 1- input NOT gate.
2. Determine output Y when inputs A=’1’, B=’0’ and C=’1’. Y= (((NOT(NOT(A) AND B)) OR NOT(C))XOR A) AND (A OR NOT (C))

**Solution**

Here we use analytical representation of AND, OR, NOT, and XOR gate to simplify the expression:

**(a)** Draw the logic circuit corresponding to the following logic expression. Use only 2- input AND gates, 2- input OR gates, 2-input XOR gate and 1- input NOT gate

Since for all A = 0, 1, so, the expression above could be simplified as below:

When , we have

When , we have

The right operand has two state 0 and 1,

It is obvious that when , the value of Y is always equal to 0, in other words, ;

when , we have , hence

**Therefore, whenever or , it would not influence the value of**

So, the original logic expression could be rewrite to:

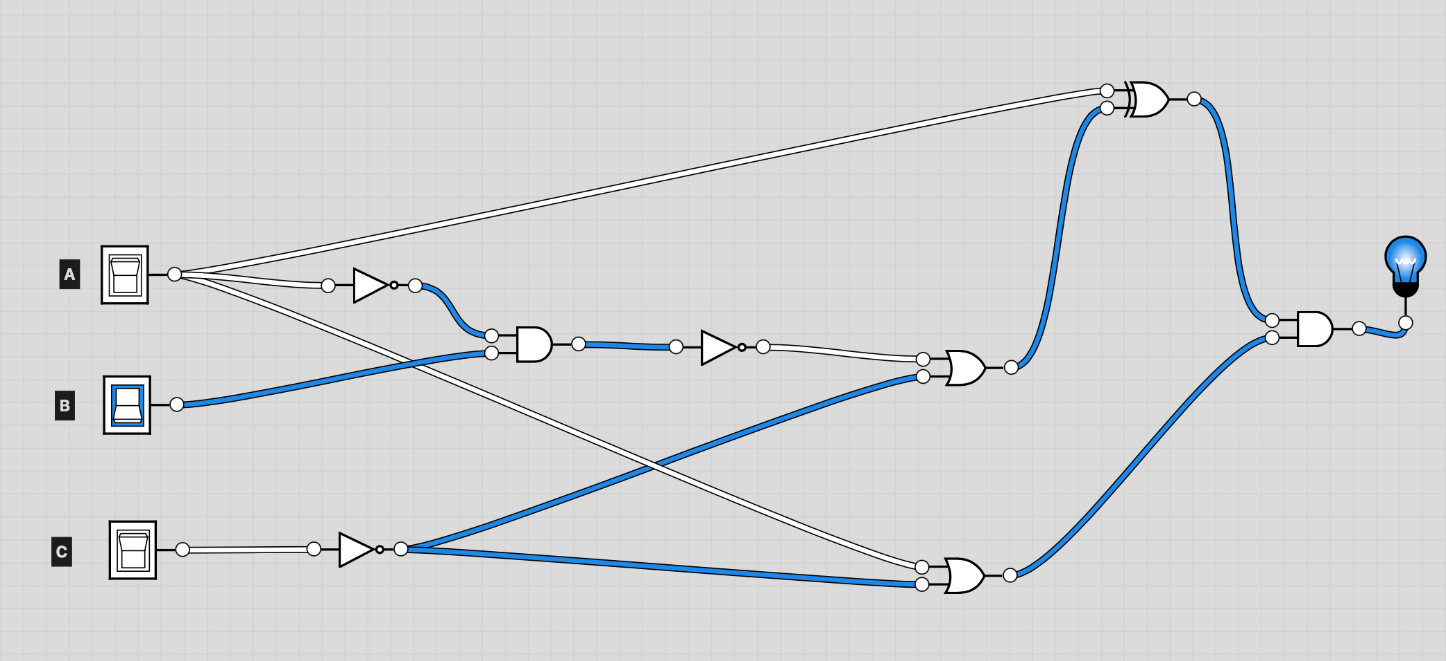
Y= (((NOT(NOT(A) AND B)) OR NOT(C))XOR A) AND (A OR NOT (C))

= ((A OR B OR NOT(C))XOR A) AND (A OR NOT (C))

Chart

Description automatically generated

I misinterpreted the title as “Use only 2 AND gates, 2 OR gates, 2 XOR gate and 1 NOT gate.” at first and after proving the value of B does not matter, I realized that. TOT. The logic circuit above is equivalent to the logic circuit below (the right one based on the original logical expression)



**(b)** Determine output Y when inputs A=’1’, B=’0’ and C=’1’

By the equation, when , the value of Y is:

# Problem 2 (1 Mark)

1. Implement NOT function using XOR logic gate.
2. Similarly, implement NOT function using XNOR logic gate.

**Solution**

**(a)** Implement NOT function using XOR logic gate.

For the input A, NOT(A) = A XOR(True)

**(b)** Similarly, implement NOT function using XNOR logic gate

For the input A, NOT(A) = A XNOR(False)

# Problem 3 (1 Mark)

1. Write the Truth –table for the following logic expression.

Y= (NOT((A AND B) OR(B AND C) OR (C AND A))) XOR (NOT(A))

Text, letter

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | OUT |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

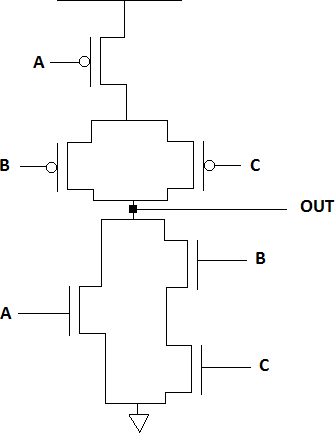
1. Based on the truth table, draw the two-level logic diagram. You can use three-input gates.

Diagram

Description automatically generated

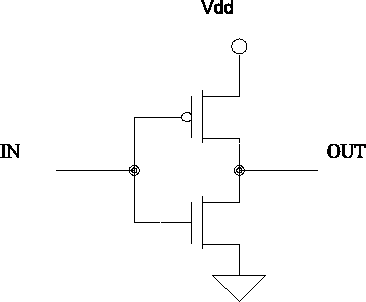
# Problem 4 (1 Mark)

* 1. Complete a truth table for the transistor-level circuit given below.



|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | OUT |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

* 1. Complete a truth table for the transistor-level circuit given below. Replace the circuit with a logic gate.

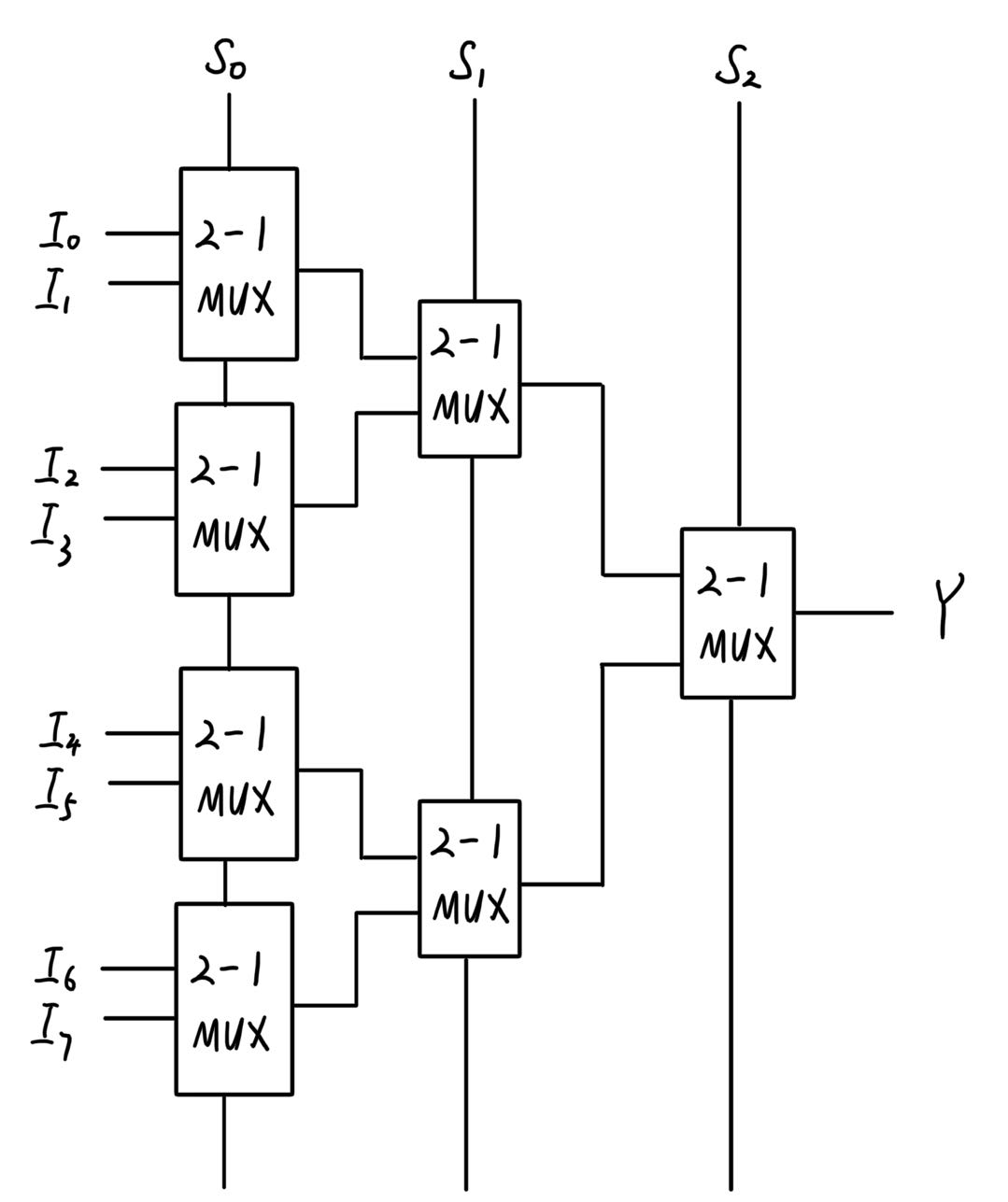


|  |  |
| --- | --- |
| IN | OUT |
| 0 | 1 |
| 1 | 0 |

# 

# Problem 5 (1 Mark)

1. Use 2:1 Multiplexers to implement an 8:1 multiplexer. (Hint: We use three 2:1 multiplexers to implement a 4:1 multiplexer)
2. In your diagram, label inputs i0 – i7, and use select line 510 to determine output Y.



|  |  |  |  |
| --- | --- | --- | --- |
| S0 | S1 | S2 | Y |
| 0 | 0 | 0 | i0 |
| 1 | 0 | 0 | i1 |
| 0 | 1 | 0 | i2 |
| 1 | 1 | 0 | i3 |
| 0 | 0 | 1 | i4 |
| 1 | 0 | 1 | i5 |
| 0 | 1 | 1 | i6 |
| 1 | 1 | 1 | i7 |

# Problem 6 (0.5 Mark)

Complete the table below. A, B and Cin are the inputs to a full adder. S is the sum bit, and Cout is the carry-out bit.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | Cin | S | Cout |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |