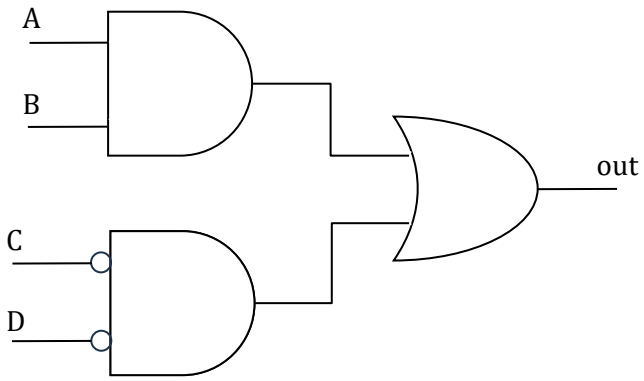
CMSC B240 Computer Organization - Spring 2024

Lab Activity #3 Solutions

Question #1

Draw the circuit diagram for the logical expression:
(A and B) or (not C and not D)

The diagram should look like this:



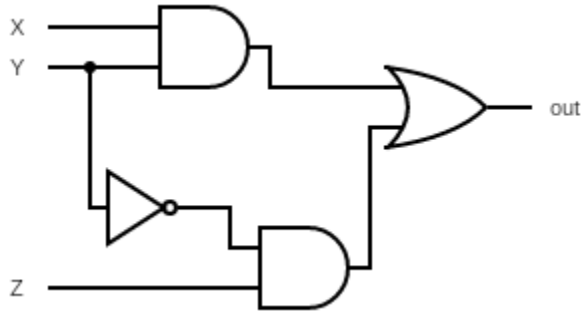
How many rows does the truth table for this circuit contain?
Since there are four inputs, there would be 2^4 = 16 combinations of input values, thus 16 rows.

What is the output of this circuit when all inputs are 0?
We would get (0 and 0) or (not 0 and not 0) → (0 and 0) or (1 and 1) → 0 or 1 → 1

Lab Activity #3 Solutions

Question #2

Complete the truth table for the following circuit diagram:



This is (X and Y) or (not Y and Z)

X	Y	Z	out
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

How many **transistors** does this circuit have?

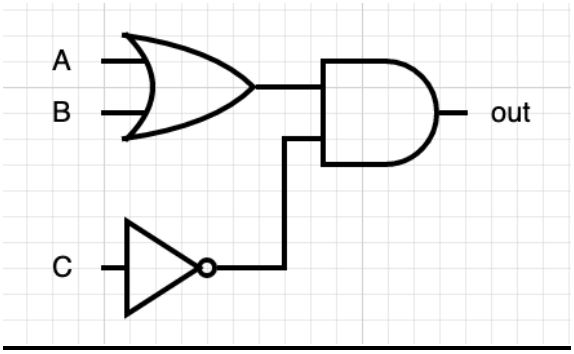
As seen in class, a NOT gate has two transistors, and an OR gate has six (four for NOR, then two more to invert it).

Since there is one NOT gate, two ANDs, and one OR in this circuit, the total is 2 + 2*6 + 6 = 20 transistors.

Lab Activity #3 Solutions

Question #3

The following circuit is attempting to implement the logical expression **(A and B) or (not C)**:



For what inputs does this circuit produce the **incorrect** output?

This circuit is implementing (A or B) and (not C). Its truth table is as follows:

```
A B C | out
0 0 0 | 0 ← incorrect
0 0 1 | 0
0 1 0 | 1
0 1 1 | 0
1 0 0 | 1
1 0 1 | 0
1 1 0 | 1
1 1 1 | 0 ← incorrect
```

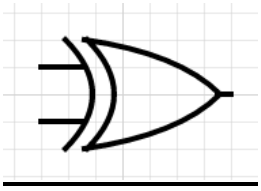
Lab Activity #3 Solutions

Question #4

Recall from our discussion of boolean logical operations that there was an “Exclusive-OR”, or XOR, function with the following truth table:

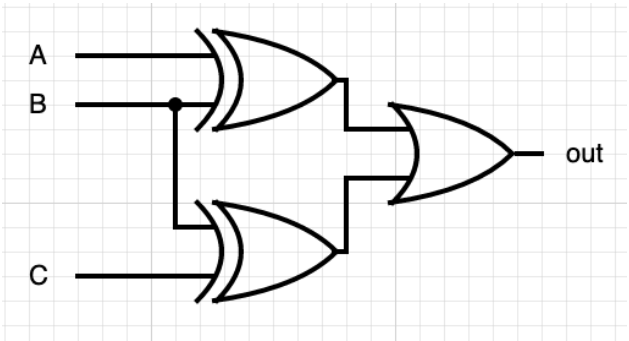
A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0

That is, the output is 0 if the inputs are the same, and 1 if they are different. The XOR gate is represented with the following symbol:



Draw the circuit diagram for the logical expression (**A xor B) or (B xor C)**, then complete the truth table.

The circuit diagram is as follows:



CMSC B240 Computer Organization - Spring 2024

Lab Activity #3 Solutions

A	B	C	(A xor B) or (B xor C)
0	0	0	0 or 0 = 0
0	0	1	0 or 1 = 1
0	1	0	1 or 1 = 1
0	1	1	1 or 0 = 1
1	0	0	1 or 0 = 1
1	0	1	1 or 1 = 1
1	1	0	0 or 1 = 1
1	1	1	0 or 0 = 0

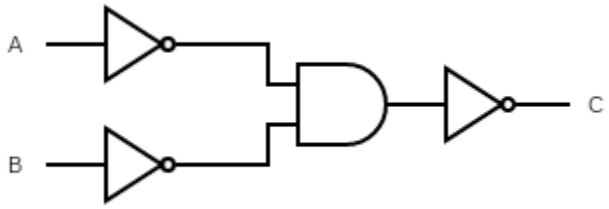
What do you notice about this truth table? That is, how could you briefly describe the operation that this circuit implements?

It returns 0 if the three inputs are the same, and 1 otherwise.

Lab Activity #3 Solutions

Question #5

Consider the following circuit diagram:



What is the logical expression for the output C in terms of the inputs A and B?

not (not A and not B)

Complete the truth table for this circuit:

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

What other logical operation has the same truth table? *Hint! This is known as DeMorgan's Law!*

This is the same as "OR".

DeMorgan's Law states that
(not A) and (not B) = not (A or B)
and that
(not A) or (not B) = not (A and B)

Lab Activity #3 Solutions

Question #6

Consider a circuit that has three inputs A, B, and C, and a single output, which should be equal to 1 if exactly two of the inputs are 1, but equal to 0 otherwise.

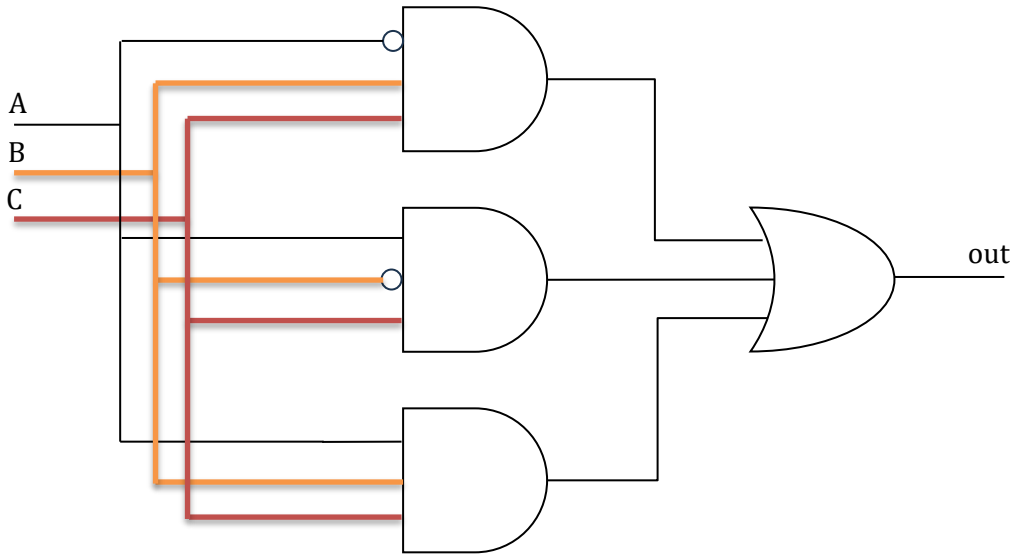
Complete the truth table for this circuit, then draw the circuit diagram.

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

This is (not A and B and C) or (A and not B and C) or (A and B and not C)

We can derive this using the “Sum of Products” algorithm, which says:

- Find each row where the output is 1
- Use “NOT” and “AND” gates for each input value in that row to create an expression so that the output is 1; this is the “products” part
- Use “OR” gates to connect the outputs from the rows in the truth table that are equal to 1; this is the “sum” part



CMSC B240 Computer Organization - Spring 2024

Lab Activity #3 Solutions

Question #7

Design a circuit that converts an 8-bit 2's-complement binary number to its negation. The circuit should have eight input lines representing the number to be converted, and eight output lines representing the conversion.

For instance, if the eight input lines were 00110110 (representing +54) then the output lines should be 11001010 (representing -54).

Here are some hints!

- Think about the algorithm we use for converting a number to its negation, and then how you would implement that in a circuit.
- You can assume the existence of a 1-bit adder like we saw in class, and draw it as a box with three inputs (A, B, and C_{in}) and two outputs (S and C_{out}).
- Where needed, you can "hardcode" values such as 0 or 1 in the circuit.

To solve this, we need to do the "flip the bits and add 1" approach.

To flip the bits, we just send each of the eight inputs to a NOT gate to invert it.

Then, to add 1, we create an 8-bit adder out of the 1-bit adders like we saw in class, with C_{out} for one adder becoming C_{in} for the next.

The outputs of the NOT gates are each sent to one of the 1-bit adders, and the other value sent to the adders is 00000001, indicating that we'll add 1.

The outputs of the 8-bit adder would represent the negated value!