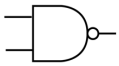
Max Score = 15 points

CS 250 2018 Spring Homework 02 SOLUTION & GRADING GUIDE

This assignment is due at 11:59:00 pm Thursday, January 25, 2018.

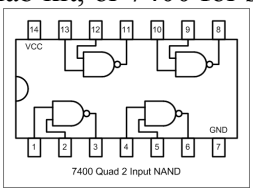
Upload your typewritten answer document in either PDF or Word format to Blackboard. Download from Blackboard to be sure that your upload was successful. Your last upload that is not marked "LATE" by Blackboard is the upload that will be graded. There is no "grace" period for late uploads

1. **[3 points across parts b, c, and d]** Consider the computation of the logic function NAND. We can represent this computation in diagrams using the NAND gate symbol shown here.



   a. Figure 2.5 in the textbook shows a representation of the NAND computation at a lower level of abstraction than the gate symbol. What is shown in that textbook figure for NAND that is omitted at the higher level of abstraction used when depicting the NAND computation using the gate symbol shown here? List the lower level symbols omitted when using the gate symbol.
   Answer: The omitted symbols are two each of the two types of MOSFET transistor and connections to + voltage and to 0 volts.

   b. This symbol is the pinout diagram representation of the SN74HTC00N quadruple 2-input NAND chip in your lab kit, or 7400 for short.



   Is the level of abstraction of the 7400 pinout diagram higher, lower, or the same with respect to a set of four NAND gate symbols?
   [3 points; 1 point each for higher, lower, same as is correct for parts b, c, and d]
   Answer: Lower. The pinout diagram contains more information, more detail, than does four copies of the 2-input NAND gate symbol, therefore the pinout diagram level of abstraction is lower.

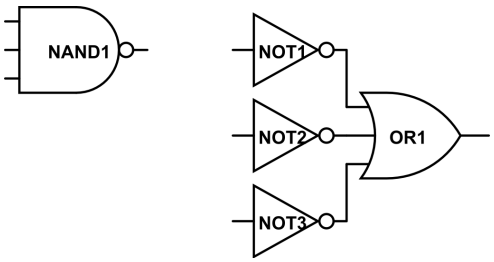
   c. Is the level of abstraction of the logic signal input pins for one gate of the 7400 pinout diagram higher, lower, or the same with respect to the logic signal inputs for Figure 2.5?
   Answer: Higher. [**Same also** accepted for full credit because of my earlier thinking and advice to students that this was the best answer before I improved my understanding of the question.] Figure 2.5 shows how the inputs are connected internally to the gate while the pinout diagram does not. The textbook calls the input signal pins A and B for each copy, while the pinout diagram uses integers to name these inputs. Letters and numbers are equally abstract naming schemes. So, overall, the abstraction of pinout diagram is higher.

d. Is the level of abstraction of the VCC and GND power and ground pins for the 7400 pinout diagram higher, lower, or the same as the + voltage (V_{dd}) and 0 volts connections for the NAND circuit in Figure 2.5?
Answer: Higher. The VCC and GND pins of the pinout diagram provide power and ground to four NAND gates without showing how the connections are made. Figure 2.5 shows how the connections for power and ground actually are made for just one NAND gate.

2. Fill in the missing computed result columns of the following truth table.

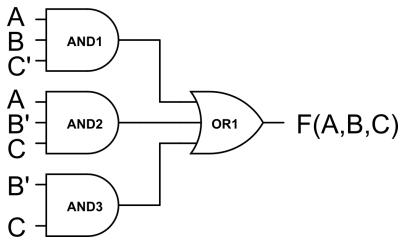
Inputs			Computed results					
A	B	C	ABC	(ABC)'	A'	B'	C'	(A'+B'+C')
0	0	0	**0**	**1**	**1**	**1**	**1**	**1**
0	0	1	**0**	**1**	**1**	**1**	**0**	**1**
0	1	0	**0**	**1**	**1**	**0**	**1**	**1**
0	1	1	**0**	**1**	**1**	**0**	**0**	**1**
1	0	0	**0**	**1**	**0**	**1**	**1**	**1**
1	0	1	**0**	**1**	**0**	**1**	**0**	**1**
1	1	0	**0**	**1**	**0**	**0**	**1**	**1**
1	1	1	**1**	**0**	**0**	**0**	**0**	**0**

What does the content of the truth table prove about the relationship between these two logic gate circuits? How is this relationship proven?



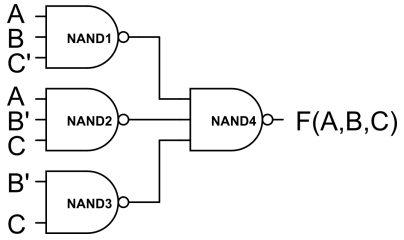
[3 points; 1 point for column (ABC) entirely correct (no point otherwise), 1 point for the (A'+B'+C' column entirely correct, and 1 point for saying the two circuits are the same.] Answer: These two gate structures compute the same function or result. The proof is that for every possible input combination of A, B, and C the table columns (ABC)' and (A'+B'+C') contain the identical result. This proof technique is called Proof by Example.

3. Consider the Boolean expression F(A,B,C) = ABC' + AB'C + B'C.
   a. Draw a gate-level circuit diagram for the logic circuit that computes F(A,B,C) and containing only gates that implement one of the three Boolean algebra operations. Use gates with more than two inputs when this will make the circuit simpler. Abstract out NOT gates for inputs by labeling an input X' as needed.
   Answer:



b. Draw a gate-level diagram containing only NAND gates for a logic circuit computing F(A,B,C). Use gates with more than two inputs when this will make the circuit simpler.

Answer:



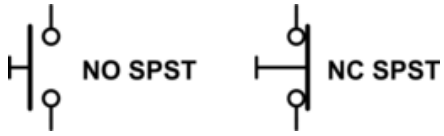
4. By fixing one input of a two input NAND gate to always receive a logic 1 the gate then functions to invert the logic value sent to the free input. Is there a 2-input function that can invert a free input when the other input is held to logic 1 and also not invert the free input when the other input is held to logic 0? If this function has a name, what is it? Otherwise answer "no name."
[3 points for naming the function XOR, 2 points for the correct truth table without the function name XOR, if that happens]
Answer: The truth table described by the question is this.

A	B	F(A,B)
0	0	0
0	1	1
1	0	1
1	1	0

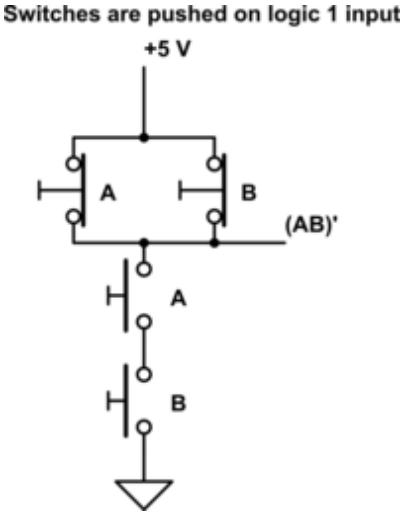
The name for this function is XOR.

5. What error is present in Figure 2.6 of the textbook?
Answer: The AND, OR, and XOR gates are all shown with a single input instead of two inputs.

6. The normally-open (NO) single-pole, single-throw (SPST) push button switch (same type as in the lab kit) transitions from high resistance to low resistance when pushed. A normally-closed (NC) SPST push button switch reverses this behavior. The schematic symbols for these two switches are shown here.



Using only switches of these two types, wire, and connections to +5 V and ground, draw a schematic to implement (AB)'. Clearly label your inputs and output.
GRADE [3 points] Answer: The circuit schematic is



7. Consider this truth table for the partially specified function F(A,B,C).

A	B	C	F(A,B,C)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	X
1	0	0	0
1	0	1	1
1	1	0	X
1	1	1	1

a. Show the K-map for F(A,B,C).
[3 points total; 1 point per part] [1 point for an entirely correct K-map (no point otherwise)] Answer:

C \ AB	00	01	11	10
0	**0**	**1**	**X**	**0**
1	**1**	**X**	**1**	**1**

b. Think about how you want to choose the values of the two Don't Care computed results, then show an image of the K-map with circles around the optimal groupings of K-map minterms to most simplify an SOP implementation of F(A,B,C). Write out this simplest SOP expression for F(A,B,C).
[1 point for F(A,B,C) = B + C] Answer: Choosing both Don't Care results to be minterms (having a logic value of 1) lets all the minterms be gathered into two groups of 4 each, which is the largest possible groupings and fewest number of groupings.

C \ AB	00	01	11	10
0	0	1	X=1	0
1	1	X=1	1	1

This corresponds to factoring out as much as possible from F(A,B,C). The central group has only the variable B that is constant at B=1. The group of the lower row of the K-Map has only C=1 as its constant valued variable. Because we are grouping minterms to form ANDed product terms, we need the variables that are not factored out to be in the form of logic 1. Here, both variables are 1 for their respective minterm groupings. Finally, we OR together these ANDed minterm groups to form the SOP expression.
Therefore, **the simplest SOP expression is F(A,B,C) = B + C.**

c. On a new image of the K-map, and re-considering the choice of values for the Don't Cares, circle the optimal groupings of maxterms to most simplify a POS implementation of F(A,B,C). Write out this simplest POS expression for F(A,B,C).
[1 point for F(A,B,C) = B + C] Answer: Choosing either of the Don't Care results to be 0 does not help form larger groups of maxterms.

C \ AB	00	01	11	10
0	0	1	X=1	0
1	1	X=1	1	1

So, leave the Don't Cares as 1 and group the maxterms as they are. A grouping of two K-Map cells factors out one variable, a variable that takes on both 0 and 1 values within the grouping, which here is input A. Because we are grouping maxterms, the we must OR together 0 forms of the remaining variables. For both maxterms here, the values of B and C are 0, so we do not need to invert them.

Therefore, **the simplest POS expression is F(A,B,C) = B + C.**
Interestingly, simplified SOP and POS forms can be the same expression.