

15110 Summer 2018
Problem Set 8

Name: _____

Andrew ID: _____

Instructions

- Type or neatly write the answers to the following problems.
- Save or scan this file as a pdf and submit to Gradescope

Exercises

1. (1.5 pts) For each of the following Boolean formulae f_1 and f_2 , write a truth table that shows the value of the formula for each possible combination of assignments to the Boolean variables A , B and C . Use 1 for True and 0 for False.
 - a. $f_1 = A \wedge (B \vee C)$
 - b. $f_2 = \neg A \vee (\neg B \wedge \neg C)$
 - c. What kind of a relationship do you observe about the formulae f_1 and f_2 by looking at their truth tables, in particular, at columns for A , B , C , f_1 , and f_2 ?

Hint: Remember how we create truth tables. We systematically list all possible combinations of the truth values for the variables in the Boolean formula, and the truth values of expressions involving these variables. For example, for Part a, all you need to do is to fill in the table below.

A	B	C	$(B \vee C)$	$A \wedge (B \vee C)$
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

2. (2 pts) The laws of Boolean Algebra, presented in the slides for the lecture on "Boolean Logic, Gates, Combinational Circuits, Levels of Abstraction," can be used to show that two formulae are equivalent by transforming one into the other. For example, here is the proof that $A \wedge (B \wedge A)$ is equivalent to $A \wedge B$:

Step	Formula	Justification (Rule)
1.	$A \wedge (B \wedge A)$	
		Commutative
2.	$A \wedge (A \wedge B)$	
		Associative
3.	$(A \wedge A) \wedge B$	
		Idempotence
4.	$A \wedge B$	

The commutative rule lets us switch the order of two terms, while the associative rule lets us move parentheses around (or add or remove them) when the connectives are all the same. The above table shows that step 2 was derived from step 1 via the commutative rule, step 3 was derived from step 2 via the associative rule, and step 4 was derived from step 3 via the idempotence rule.

We can also use DeMorgan's Law to transform expressions (as in lecture). Here is the proof that $\neg (A \wedge (B \vee C))$ is equivalent to $\neg A \vee (\neg B \wedge \neg C)$:

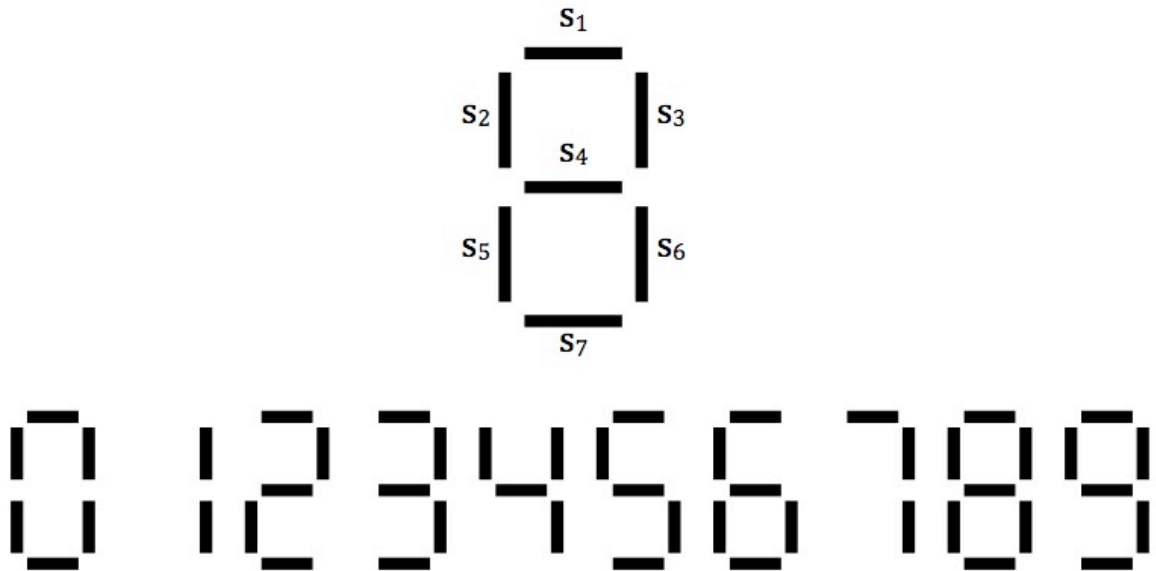
Step	Formula	Justification (Rule)
1.	$\neg (A \wedge (B \vee C))$	
		DeMorgan's Law
2.	$\neg A \vee \neg (B \vee C)$	
		DeMorgan's Law
3.	$\neg A \vee (\neg B \wedge \neg C)$	

- a. How can you explain your answer to 1.c. in terms of De Morgan's Law? Hint: See the derivation above.

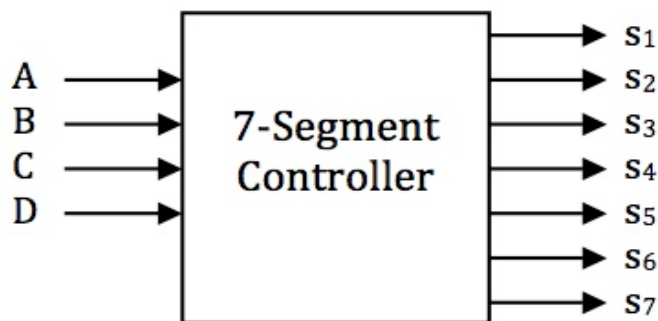
- b. The formula f_1 from Problem 1 represents a Boolean function. Draw the combinational circuit that can compute f_1 using AND, OR and NOT gates.
- c. Draw a combinational circuit that computes the equivalent of the Boolean function f_2 that uses a single AND, a single OR and a single NOT gate.
- d. How could you realize the expression $\neg (A \wedge \neg (B \wedge C))$ using only NAND gates and no other type of gate?

3. (2.5 pts) Let ABCD represent a 4-bit binary-coded decimal number. For example, the decimal number 2 would be represented by the binary number 0010, which is expressed by letting $A = 0$, $B = 0$, $C = 1$ and $D = 0$.

A seven-segment display can be used to form each of the ten decimal digits as shown below.



We can define a circuit abstractly below that requires four Boolean inputs A , B , C , and D , and produces seven Boolean outputs s_1 , s_2 , ..., s_7 to make the segments of the display form digits:



A segment s_i is lit if s_i is 1 (True) and not lit if s_i is 0 (False).

- a. Derive a Boolean formula for s_7 that is True (1) if and only if segment s_7 is lit.

Your answer should be of the following form:

$$s_7 = (\text{_____}) \vee (\text{_____}) \vee (\text{_____}) \vee \dots$$

where each missing section is a Boolean expression with all 4 input variables that is True when ABCD represents a decimal digit that results in the segment s_7 of the display being lit. To help you get started, the digit 0 (binary 0000) will light segment s_7 , so the first expression in the formula above would be

$$(\neg A \wedge \neg B \wedge \neg C \wedge \neg D)$$

since s_7 would be True (i.e. 1) if $A = 0$, $B = 0$, $C = 0$, and $D = 0$. You will need to find all of the other missing expressions for s_7 . Hint: Consider all the digits for which s_7 would be lit.

- b. Derive a simple Boolean expression for s_6 that is True (i.e. 1) if and only if the segment s_6 is lit. (HINT: Segment s_6 is lit for all digits except when the digit is the number 2.)

4. (2 pts) Draw a circuit using 2 full adders that adds two two-bit binary values. The slides from lecture show a full adder for adding 2 one-bit binary values A and B. Using that circuit as a basis build a circuit to add two two-bit values A1 A2 and B1 B2: A2 will be added to B2 (possibly with a carry) and then A1, B1, and the carry (if it exists) will be added to get the result. Your solution should show the actual circuitry in terms of gates, such as the one on Slide with title "Full Adder (FA)".

5. (2 pt)

- a. A word size is characteristic to a given computer architecture. It denotes the number of digits that a CPU can process at one time. Modern processors, usually have a word size of 8, 16, 24, 32 or 64 bits; most current general purpose computers use 32 or 64 bits. The word size of a modern computer typically also describes the size of the address space on that computer. For example, a computer that is said to be "32-bit" usually allows 32-bit memory addresses.

What is the largest amount of memory in **gigabytes** that can be directly addressable in a 32-bit architecture? Assume that the architecture supports byte-addressable memory where each addressable location corresponds to one byte of memory.

- b. What does it mean for a programming language to be high-level? In answering this question you may use external resources. However, your answer should be given in your own words and any references you use should be cited.