

CS 135 Team Homework 2. Fall 2010: Due Sept.20th 10am.

Note: Your submission must include a description of how the work was divided amongst the team members.

1. Question 3.5
2. Question 3.8—Error in textbook question...The logic equation is $Y = \text{NOT} (A \text{ AND } (B \text{ OR } C))$
3. Question 3.7
4. Your task is to design the logic to control a overhead light in a room based on the position of two light switches. Because the switches are at two different entrances to the room, either switch should be able to change the state (off/on) of the light independently. If both switches are in the “down” position (represented by a zero), the light must be off (represented by a zero). No matter what position the switches are in or the current state of the light, flipping either switch must change the state of the light.
 - (a) First complete the truth table for this circuit/function.

S ₀ (Switch 0)	S ₁ (Switch 1)	L (Light)
Down (0)	Down (0)	
Down (0)	Up (1)	
Up (1)	Down (0)	
Up (1)	Up (1)	

- (b) Next, create a gate level circuit using AND, OR, and NOT gates only.
 - (c) Is there a common name for this function ?
5. Tyrone Shoelaces has invested a huge amount of money into the stock market and doesn't trust just anyone to give him buying and selling information. Before he will buy a certain stock, he must get input from three sources. His first source is Pain Webster, a famous stock broker. His second source is Meg A. Cash, a self-made millionaire in the stock market, and his third source is Madame LaZora, world famous psychic. After several months of receiving advice from all three, he has come to the following conclusions:
 - a. Buy if Pain and Meg both say yes and the psychic says no.
 - b. Buy if the psychic says yes.
 - c. Don't buy otherwise.

Construct a truth table and find the Boolean function to implement the logic telling Tyrone when to buy.

6. More on Trinary logic. In the first group assignment you discussed the theory behind Base 3 logic and provided definitions for AND and OR operations. Specifically, we can define $OR = \max(A,B)$ and $AND = \min(A,B)$ (for two trinary inputs A and B). There are several ways in which logical complement, the NOT operator, could be defined in base 3. One such definition is $NOT(0)=2$, $NOT(1)=1$, $NOT(2) = 0$.

- (a) Using the above definitions for AND, OR, NOT, does DeMorgan's Law still apply for base 3 logic? Verify using truth tables.
- (b) Is the base 3 logic a Boolean algebra (using only the above operators of AND, OR, and NOT in base 3 logic)? Give a proof or counterexample.

7. Review your Boolean algebra and Karnaugh maps material from CS123 – Chapter 15 (first two sections at least) of your CS123/124 textbook. Apply the laws of Boolean algebra to simplify the expression

$$F(x,y,z) = x'yz + x'yz' + xz$$

8. Download the CedarLogic simulator tool (or any other logic simulation tool that you can find) – the link is provided on the lecture notes webpage—and examine the circuit provided in the link titled “Team HW2 Circuit1”. What does circuit accomplish? Provide the Boolean function that the circuit is implementing. Next examine the circuit provided in the link “Team HW2 Circuit 2”. What does this circuit accomplish? Is it different, and how, from Circuit 1?

9. The NOR gate is logically complete (i.e., any logic function can be built using only NOR gates). Use only NOR gates to construct a circuit that computes the AND function (i.e., circuit takes two inputs A,B and output is A AND B).

10. **Error detection:** A binary string can get corrupted when sent over a communication network, due to transmission errors, or in memory due to material properties or changes in voltage. (Such communication channels or storage mediums can never be completely error-free.) Therefore, the need for error detection schemes that can detect if a binary string has been corrupted, and thus no longer contains the original (correct) data. One of the earliest and simplest such schemes is to add a parity bit to the ASCII byte (and thus utilize all 8-bits in the byte). A parity bit is either turned “off” (0) or “on” (1) depending on whether the sum of the other bits (i.e., sum of the 1's) in the byte is even or odd. For example, if we decide to use even parity and we want to send ASCII character A, the lower 7 bits are 100 0001. Because the sum of the bits is even, the parity bit would be set to off and we would transmit 0100 0001. If we want to send ASCII character C, then lower 7 bits are 100 0011 and the parity bit would be set to 1 and we transmit 1100 0011. When the string is received, the parity bit is checked for consistency. For example, if a string 1100 0001 is received then checking the parity shows that there is an error – since the sum of the bits is even but the parity bit is set to 1.

- Does this Parity bit scheme detect all possible errors in transmission/storage? Explain and justify.

