CS226 Practice Problem Set 1 (Spring 2016)

Date posted: Jan 13, 2017 **Expected Solving Time: 45 mins**

- *Be brief, complete and stick to what has been asked.*
- *Unless asked for explicitly, you may cite results/proofs covered in class without reproducing them.*
- *If you need to make any assumptions, state them clearly.*
- ***These are ungraded practice questions. You are strongly encouraged to solve these on your own to ensure you understand the material being taught in class.***
- ***Mutual discussion is allowed, but copying is not. Please read the guidelines on the course webpage if you don't understand the distinction between the two.***

1. Consider the following algorithm for computing z_1 and z_2 as functions of a, b, c.

```
bool a, b, c, z1, z2;
read(a); read(b); read(c);

while (a xor b) {
  if (a) then
    z1 = b xor c;
    z2 = b nand c;
    b = z1 or z2;
  else {
    z1 = b and c;
    z2 = b xnor c;
    a = z1 and z2;
  }
}
output(z1); output(z2);
```

You are told that the input combination $a = 0, b = 1, c = 0$ is never fed as input to the above algorithm (the algorithm doesn't terminate in this case). Implement a circuit with as few 2-input AND, 2-input OR and NOT gates as you can (total gate count should be mnimized) that achieves the same purpose as the above algorithm, i.e. given three boolean inputs a, b and c that are not $0, 1$ and 0 respectively, it outputs the same values of $z1$ and $z2$ as done by the above algorithm.

2. Consider the Boolean functions $f_1(x, y, z) = x.y + \overline{x}.z + y.z$ and $f_2(x, y.z) = x.y + \overline{x}.z$

Construct K-maps for the two functions using the templates given below. Note the listing of variables along the different dimensions of the two K-maps are different.

K-map for f_1

x,y→ / z↓	00	01	11	10
0				
1				

K-map for f_2

y,z→ / x↓	00	01	11	10
0				
1				

Can you say whether the two functions are semantically equivalent (i.e. have the same truth tables) from the above K-maps?

Now draw the K-maps of both f_1 and f_2 using the template give below. Note that the listing of variables along different dimensions of the two K-maps are now the same.

K-map for f_1

x,y→ / z↓	**00**	**01**	**11**	**10**
0				
1				

K-map for f_2

x,y→ / z↓	**00**	**01**	**11**	**10**
0				
1				

Can you now say whether the two functions are semantically equivalent from the two K-maps above?

3. Show that any Boolean function can be implemented if you have sufficiently many of *any one* of the following basic circuits, and access to the Boolean constants 0 and 1:

   (a) 1-bit comparator with only the GT (greater than) output.

   (b) A half-adder with sum and carry outputs.

   (c) A multiplexor.

4. Let a, b, c, d be four Boolean variables. How many distinct functions $f(a, b, c, d)$ exist such that if we try to implement f in a sum-of-products form (like we did when we implemented functions from truth tables or K-maps), we *must* use AND gates with 4- (or more) inputs? In other words, in every implementation of the function in sum-of-products form, every product term has four arguments, and this can't be simplified further.