Name: Nick Petro **CoE 0147**

Lab 7, Spring 2016

K-Maps

Important:

- While collaborations are allowed, names of all the collaborators must be listed in the assignment. Each student should submit individual assignments for grading, despite collaboration.
- Solution files (***.circ files**) for this assignment should be uploaded to the courseweb before the next recitation. **Please zip the files together and name the zip file as your Pitt email id**.
- **Paper copy** of this assignment is due in the classroom before the beginning of the next recitation.

Use of multi-input gates is permitted for this assignment.

1 Combination Lock

In this section you will design a combination lock. Your combination lock works with a 4-bit binary combination and can be unlocked only when the input combination is 1101.

This circuit is built using two components: **LockDetect** and **PopCount**. Please use the circuit diagram available on the courseweb as `lab7part1` for this part. In this circuit diagram, the combination lock detector uses two components — LockDetect and Popcount. However, the circuit is incomplete because the subcircuits LockDetect and PopCount are not yet drawn. You have to complete these subcircuits according to their descriptions below.

Save the changes in lab7part1.circ

LockDetect will output a '1' only when the input is the correct unlock code (i.e., 1101), otherwise LockDetect will output a '0'. Assume that the input bits are labeled as: 3rd bit as A, 2nd bit as B, 1st bit as C and 0th bit as D. If the output of LockDetect is Y, then Y = $AB\overline{C}D$. This can be easily implemented using AND and NOT gates.

PopCount is a small helper circuit which displays how many '1's are correctly placed on the input. It does not consider the '0's in the input. For example, if the input is 1001, then the output from PopCount will be 2 because the 0th bit and 3rd bit are '1' and are in the correct position according to the unlock code 1101. Similarly, if the input is 1111, then the output of PopCount will be 3 because three ones are correctly placed on the input.

Assume the input bits are as follows: 3rd bit is A, 2nd bit is B, 1st bit is C and 0th bit is D. The output from PopCount ranges from 0 to 3 in base 10 (decimal). Thus, you require a 2-bit output from PopCount. Assume the output bits are as follows: 1st bit is X1 and 0th bit is X0.

Complete the truth table and the K-Maps for X1 and X0 shown below. After you have populated the the K-Maps, circle the minterms and write your simplified equation as a sum of products (SOP). Be sure to show all of your work.

You are required to turn in the hard copy for this part.

input				**output**	
A	B	C	D	X1	X0
0	0	0	0		
0	0	0	1		
0	0	1	0		
0	0	1	1		
0	1	0	0		
0	1	0	1		
0	1	1	0		
0	1	1	1		
1	0	0	0		
1	0	0	1		
1	0	1	0		
1	0	1	1		
1	1	0	0		
1	1	0	1		
1	1	1	0		
1	1	1	1		

AB \ CD	00	01	11	10
00				
01				
11				
10				

AB \ CD	00	01	11	10
00				
01				
11				
10				

Figure 1: Solve the Kmaps for X_0 and X_1

2 Parity generation

This circuit is built using **ParityGen**. Please use the circuit diagram available on the courseweb as `lab7part2` for this part. In this circuit diagram, you will find a parity generator module. However, the circuit is incomplete because the circuit ParityGen is not yet drawn. You have to draw the circuit of ParityGen according to its description below.

Save the changes in lab7part2.circ

ParityGen is a circuit which generates a parity for the input. We count the number of '1's (T) in the input. Then we calculate the input parity (P) and add the parity into total number of '1's. Finally, we output the sum (S = T + P). Let us assume the input bits are labeled as: 3rd bit as A, 2nd bit as B, 1st bit as C and 0th bit as D. The input parity is given by the equation P = A `XOR` B `XOR` C `XOR` D. For example, if a 4-bit input is 1011, the total number of '1's (T) is 3, input parity (P) is 1 by calculating 1 `XOR` 0 `XOR` 1 `XOR` 1. Hence the sum (S) is 100 (4).

The sum output is at most 3 bits and can be represented by X2 X1 X0. X2 is the 2nd bit, X1 is the 1st bit and X0 is the 0th bit. Complete the truth table and the K-Maps for X2, X1, and X0. After you have filled out the K-Maps, circle the minterms and write your simplified equation as a sum of products. Be sure to show all of your work.

You are required to turn in the hard copy for this part.

input				**output**		
A	B	C	D	X2	X1	X0
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			
0	1	0	0			
0	1	0	1			
0	1	1	0			
0	1	1	1			
1	0	0	0			
1	0	0	1			
1	0	1	0			
1	0	1	1			
1	1	0	0			
1	1	0	1			
1	1	1	0			
1	1	1	1			

AB \ CD	00	01	11	10
00				
01				
11				
10				

AB \ CD	00	01	11	10
00				
01				
11				
10				

AB \ CD	00	01	11	10
00				
01				
11				
10				

Figure 2: Solve the Kmaps for X_0, X_1, and X_2