Homework 4

Homework 4 is due on Friday, September 29 at 2:30pm.

Remember to include your Discussions section (e.g. AD1) and follow the complete Homework submission guidelines.

Please ask all questions about this assignment during the TA and UA office hours or in the comment section at bottom of this page. Questions sent via e-mail will not be answered.

Please remember to submit your homework before the deadline and staple your submission. Homework that is not stapled or late will NOT be graded. No exceptions.

Most of the problems in this homework require showing your work. If you do not show your work, half of the points may be taken away!

CMOS logic, Boolean expressions, minimization

1. CMOS Logic
a) Complete the circuit below so that it implements a NAND gate.
Specifically, in each of the four boxes draw a p-type or n-type transistor as needed to implement $C = A \text{ NAND } B$. 
b) Fill in a truth table for the following CMOS circuit. What logic gate does this circuit implement?

![CMOS Circuit Diagram]

2. Canonical Forms

The Boolean functions $f$ and $g$, with operands $x, y, z$, are defined below by their truth tables.

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>$z$</th>
<th>$f$</th>
<th>$g$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

1. Express function $f$ and function $f+g$ in sum of minterms (canonical SOP) form.
2. Express function $g$ and function $fg$ in product of maxterms (canonical POS) form.

3. Truth Table in C

Copy the C program below. This program generates the truth table for a three input OR function.
#include <stdio.h>

int main()
{
    int x, y, z, f;
    printf("\nTruth Table\n");
    printf("x y z   f\n");

    for (x = 0; x <= 1; x=x+1)
    {
        for (y = 0; y <= 1; y=y+1)
        {
            for (z = 0; z <= 1; z=z+1)
            {
                f = (x | y | z)&1;
                printf("%d %d %d   %d\n", x, y, z, f);
            }
        }
    }
    return 0;
}

1. Modify the program to print a truth table for function \( f \) given by its canonical SOP form from your solution to problem 2.1 above.
2. Compare the printout produced by your program to the truth table in problem 2. Turn in a printout of your program and the printout of the
   truth table the program produces as output.
3. Look at the code, line 15. Notice that each logic calculation uses bitwise operators (~, &, and |), and that the last operation uses &1 to
   mask out all but the least significant bit. Edit the code and remove the &1 operation from your answer to part 1, then recompile and
   re-execute the code. Turn in a printout of the truth table the program produces as output. In fewer than 20 words, explain why the
   function \( f \) values appear as negative values in the resulting output.

4. Simplifying with K-maps

The functions \( f(w,x,y,z) \) and \( g(w,x,y,z) \) are specified by their Karnaugh maps shown below:
For each function:

1. Circle on the K-map and list all prime implicants.
2. Find a minimal SOP expression. Is it unique? If not, give an alternative minimal SOP expression.
3. Find a minimal POS expression. Is it unique? If not, give an alternative minimal POS expression.

5. Simplifying with K-maps

Let $f(a,b,c,d) = a'b'+(cd+c'd')'$

1. Using a K-map like the one shown below, derive a minimal SOP expression for $f$. Show your work.
2. Using a K-map like the one shown below, derive a minimal POS expression for $f$. Show your work.

6. C Program Modification

Download or copy the following program: factorial.c. The code calculates the factorial of a number provided by the user. Unfortunately, the code gives erroneous answers for integer inputs that are negative valued and values for which their factorial exceeds $2^{31}$. Your mission is to extend this program so that it accepts only those inputs that will produce correct outputs, asking the user to try again until a valid entry is given. (Hint:
Use a loop construct. For each unacceptable input, the program should print the appropriate warning message "The input is not acceptable, try again."

a) Create a flow chart for your new, enhanced version of this program. Turn it in as part of your homework solution. Create a complete flow chart in a standard manner as showed in textbook and lecture. By standard, we need start and end, proper use of boxes (rectangles, parallelograms, etc.), conditions..... If you are uncertain of how to draw correct flow charts, please refer to textbook chapter 13. There are plenty of examples.

b) Write, compile, run, test, and debug your program code to confirm that it provides the correct output for integer inputs within 32-bit range. Turn in a printout of your program code as part of your homework solution.