Homework 1
(Due date: January 21st @ 5:30 pm)
Presentation and clarity are very important!

PROBLEM 1 (25 pts)

a) Simplify the following functions using ONLY Boolean Algebra Theorems. For each resulting simplified function, sketch the logic circuit using AND, OR, XOR, and NOT gates. (12 pts)

\[
F = A(B \oplus C) + B \quad \quad F = (\overline{C} + \overline{B})(C + A)(\overline{B} + A) + CA
\]

\[
F(X,Y,Z) = \Pi(M_1, M_3, M_6, M_7) \quad \quad F = (X + Z)Y + \overline{X}YZ
\]

b) Based on the formula \(x \oplus y = xy + \overline{xy}\), demonstrate that \((a \oplus b) \oplus c = a \oplus (b \oplus c) = b \oplus (a \oplus c)\). You can express each function using the canonical sum of products, or complete the truth table for each function. (5 pts)

c) For the following Truth table with two outputs: (8 pts)

- Provide the Boolean functions using the Canonical Sum of Products (SOP), and Product of Sums (POS).
- Express the Boolean functions using the minterms and maxterms representations.
- Sketch the logic circuits as Canonical Sum of Products and Product of Sums.

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>z</th>
<th>f₁</th>
<th>f₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</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>0</td>
</tr>
</tbody>
</table>

PROBLEM 2 (10 pts)

- Design a circuit (simplify your circuit) that verifies the logical operation of a 3-input NAND gate. \(f = '1'\) (LED ON) if the NAND gate does NOT work properly. Assumption: when the NAND gate is not working, it generates 1’s instead of 0’s and vice versa.

PROBLEM 3 (15 pts)

- Complete the truth table for a circuit with 4 inputs \(x,y,z,w\) that activates an output \((f = 1)\) when the number of 1’s in the inputs is odd. For example: If \(xyzw = 1100 \rightarrow f = 0\). If \(xyzw = 1011 \rightarrow f = 1\).
- Provide the Boolean function using the minterm representation.
- Sketch the logic circuit using ONLY 2-input NAND gates. Tip: try to simplify the function using XOR gates.

PROBLEM 4 (20 pts)

a) Complete the timing diagram of the logic circuit whose VHDL description is shown below: (5 pts)

```vhdl
library ieee;
use ieee.std_logic_1164.all;

entity circ is
  port ( a, b, c: in std_logic;
        f: out std_logic);
end circ;

architecture st of circ is
  signal x, y: std_logic;
  begin
  x <= not(a xor b);
  y <= x nand c;
  f <= y xor (not a);
end st;
```
b) The following is the timing diagram of a logic circuit with 3 inputs. Sketch the logic circuit that generates this waveform. Then, complete the VHDL code. (10 pts)

```vhdl
library ieee;
use ieee.std_logic_1164.all;

entity wav is
  port ( a, b, c : in std_logic;
         f : out std_logic);
end wav;

architecture st of wav is
  -- ???
begin
  -- ???
end st;
```


c) Complete the timing diagram of the following circuit: (5 pts)

![Timing Diagram]

**PROBLEM 5 (30 PTS)**

- A numeric keypad produces a 4-bit code as shown below. We want to design a logic circuit that converts each 4-bit code to a 7-segment code, where each segment is an LED: A LED is ON if it is given a logic ‘1’. A LED is OFF if it is given a logic ‘0’.

- Complete the truth table for each output \((a, b, c, d, e, f, g)\).
- Provide the simplified expression for each output \((a, b, c, d, e, f, g)\). Use Karnaugh maps for \(a, b, c, d, e\) and the Quine-McCluskey algorithm for \(f, g\). Note it is safe to assume that the codes \(1010\) to \(1111\) will not be produced by the keypad.

<table>
<thead>
<tr>
<th>Value</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>W</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- Complete the truth table for each output \((a, b, c, d, e, f, g)\).
- Provide the simplified expression for each output \((a, b, c, d, e, f, g)\). Use Karnaugh maps for \(a, b, c, d, e\) and the Quine-McCluskey algorithm for \(f, g\). Note it is safe to assume that the codes \(1010\) to \(1111\) will not be produced by the keypad.