1. (8 points) What is the dual of the expression \( A'B' + (C' + D)(A + B') + (A+B)'C \)?

What is its complement?

Using Boolean algebra, derive an expression equivalent to \( A'B' + (C' + D)(A + B') + (A+B)'C \) that uses at most 2 AND operations and 4 OR operations.
2. (8 points) How many bits are required to represent the decimal number 3472 in ASCII?

How many bits are required to represent 3472 in BCD?

How many bits are required to represent 3472 in binary?

Give the bits that represent the excess 3 representation of 3472.
3. (12 points) The simulation output shows selected signals from the processor introduced in section 1 of the course notes. The portions of the output corresponding to four different instructions are outlined. Identify the instructions that are being executed. Give the name of the instruction (e.g. direct load, branch-on-positive) and its complete numeric representation. Note that some parts of the simulation output have been blanked out.

- **0000** halt – halt execution
- **0001** negate – $ACC := -ACC$
- **1xxx** immediate load – if sign bit of $xxx$ is 0 then $ACC := 0xxx$ else $ACC := fxxx$
- **2xxx** direct load – $ACC := M[0xxx]$
- **3xxx** indirect load – $ACC := M[M[0xxx]]$
- **4xxx** direct store – $M[0xxx] := ACC$
- **5xxx** indirect store – $M[M[0xxx]] := ACC$
- **6xxx** branch – $PC := 0xxx$
- **7xxx** branch if zero – if $ACC = 0$ then $PC := 0xxx$
- **8xxx** branch if positive – if $ACC > 0$ then $PC := 0xxx$
- **9xxx** branch if negative – if $ACC < 0$ then $PC := 0xxx$
- **axxx** add – $ACC := ACC + M[0xxx]$
4. (12 points) Consider the circuit shown below. Suppose this circuit is implemented directly, using CMOS, with each AND gate implemented using a NAND and an inverter (similarly for OR gates). What is the worst-case delay for the circuit, assuming that NAND and NOR gates have a delay of 1 ns each, and inverters have a delay of 0.5 ns? Highlight the path through the circuit that accounts for the worst-case delay.

Show an alternate implementation, using NANDs, NORs and inverters that reduces the delay by at least 1.5 ns.
5. (12 points) Use a Karnaugh map to find a simplest \textit{sum-of-products} expression for \( F(X,Y,Z) = \Sigma m(0,2,5) \), \( d(X,Y,Z) = \Sigma m(3,6) \).

Use a Karnaugh map to find a simplest \textit{product-of-sums} expression for \( F(A,B,C,D) = \Sigma m(0,2,4,5,8,10,15) \), \( d(A,B,C,D) = \Sigma m(3,6,7,11,13) \)
6. (15 points) The circuit below implements a four bit first mismatch function. That is, the output \( x_i \) is high, if \( i \) is the largest integer for which \( a_i \neq b_i \). Note that at most one of the \( x \) outputs is high and the valid output is high if there is a mismatch in some input bit pair.

What is the worst-case propagation delay for a 32 bit version of this circuit, if every simple gate has a delay of 1 ns?

![Circuit Diagram]

Draw another version of the circuit above that uses lookahead to make it faster.
7. (12 points) For each VHDL code fragment shown below, write an equivalent set of simple signal assignments (each involving just a single signal, not a vector or signals).

(a) \[ z(2 \text{ downto } 0) <= (a \text{ or } b) \& \& (b \text{ xor } c); \]

(b) \[ c <= x \text{ and } y; \]
    \[ \text{if } a /= b \text{ then} \]
    \[ d <= x \text{ or } y; \]
    \[ \text{else} \]
    \[ c <= \text{not } y; \]
    \[ d <= '0'; \]
    \[ \text{end if}; \]

(c) \[ \text{for } i \text{ in } 1 \text{ to } 2 \text{ loop} \]
    \[ a(i) <= a(i-1) \text{ and } x(i); \]
    \[ x(i) <= b(i) \text{ or } x(i-1); \]
    \[ \text{end loop}; \]
library IEEE;
use IEEE.std_logic_1164.all;

entity foo is port (
    A: in std_logic_vector(0 to 3);
    B: in std_logic_vector(0 to 3);
    C: in std_logic;
    X: out std_logic_vector(0 to 4));
end foo;

architecture bar of foo is
begin
    process(A, B, U) begin
        U <= '0';
        if A(0) = '1' and B(0) = '1' then
            U(0) <= '1';
        else
            U(0) <= '0';
        end if;
        if A(1) = '1' and B(1) = '1' then
            U(1) <= '1';
        else
            U(1) <= '0';
        end if;
        if A(2) = '1' and B(2) = '1' then
            U(2) <= '1';
        else
            U(2) <= '0';
        end if;
        if A(3) = '1' and B(3) = '1' then
            U(3) <= '1';
        else
            U(3) <= '0';
        end if;
        if C = '1' then
            U(4) <= '1';
        else
            U(4) <= '0';
        end if;
    end process;
end bar;