1. (16 points) Find the simplest sum-of-products equivalent to the expression below using Boolean algebra only.

\[(A'B'D' + BCD' + BC'D' + AC')'

Show how your simplified version of the above expression can be implemented using NAND gates and inverters only (you need not limit yourself to simple gates).
Draw a CMOS two input NAND gate using NFETs and PFETs. Show how to extend this to a three input NAND gate using two more transistors.

If the NAND circuit you drew on the last page is implemented using CMOS NAND gates, how many transistors does it require?
2. (6 points) Fill in the Karnaugh map below so that it corresponds to the function
\( F(A,B,C,D) = \Sigma m(1,2,6,8,10,14) \), \( d(A,B,C,D) = \Sigma m(0,4,7,9,13) \). Derive a product of sums expression for this function, taking full advantage of the don’t care conditions.

\[
\begin{array}{cccc}
\text{CD} & 00 & 01 & 11 & 10 \\
00 & & & & \\
01 & & & & \\
11 & & & & \\
10 & & & & \\
\end{array}
\]
3. (10 points) Draw a logic diagram that directly corresponds to the VHDL module shown below. Use simple gates only. Every signal that appears in the VHDL should be labeled in your logic diagram.

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

entity foo is port (
    A: in std_logic;
    B: in std_logic_vector(3 downto 0);
    X: out std_logic_vector(3 downto 0);
    Y: out std_logic);
end foo;
architecture a1 of foo is
begin
    signal Z: std_logic_vector(4 downto 0);
    process(A, B, Z) begin
        Z(4 downto 1) <= B; Z(0) <= A;
        for i in 0 to 3 loop
            if i = 0 or i = 2 then
                Z(i+1) <= B(i) and Z(i);
            elsif i = 1 then
                Z(i+1) <= B(i) or Z(i);
            end if;
        end loop;
        X <= B xor Z(3 downto 0);
        Y <= Z(4);
    end process;
end a1;
```
4. (15 points) With respect to the circuit shown below, give a general expression for output $X_i$ in terms of inputs $B, A_0, \ldots, A_i$.

Suppose this circuit is extended to 64 bits and the exclusive-or gates are implemented using NAND gates. What is the worst-case delay from the time that an input changes until the time an output changes, assuming 1 ns of delay per gate.
Complete the schematic diagram shown below so that it implements a lookahead version of the circuit. In the final circuit, each gate should have two inputs.

What is the worst-case delay for a 64 bit version of the lookahead circuit?
5. (15 points) The simulation output shows a program executing on the dual processor system from design problem 2. Fill in the blanks below with the information that belongs in each of the blanked out areas in the simulation output.

A. ______________________ B. ______________________ C. ______________________
D. ______________________ E. ______________________ F. ______________________
G. ______________________ H. ______________________ I. ______________________
J. ______________________

0000 halt
0001 negate
0002 load pid
1xxx immediate load
2xxx direct load
3xxx indirect load
4xxx direct store

5xxx indirect store
6xxx branch
7xxx branch if zero
8xxx branch if positive
9xxx branch if negative
axxx add
6. (12 points) For the sequential circuit shown below, complete the timing diagram to show when the input must be stable and when the output can be changing. For the input, label the start and end of each stable period relative to the rising clock edge (so write $-3$ to indicate 3 ns before the clock edge, for example). Similarly label the ends of the period when the output can be changing. Assume that the flip flop setup time is 1.5 ns, the flip flop propagation delay is between 1.5 and 2.5 ns, the hold time is 1 ns and all gates have delays that vary between .4 ns and 1.3 ns.

What is the smallest clock skew for which it’s possible for the circuit to experience internal hold-time violations?

For a clock skew of 1 ns, what is the largest clock period for which the circuit can experience internal setup time violations?
7. (10 points) Complete the state table shown below, so that it corresponds to the state diagram shown at left.

<table>
<thead>
<tr>
<th>$S_1S_0\ AB$</th>
<th>$XY$</th>
<th>$D_1D_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>00 00</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>00 01</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>00 10</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>00 11</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>01 00</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>01 01</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>01 10</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>01 11</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>11 00</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>11 01</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>11 10</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>11 11</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Write an output equation for $X$. Use the don’t care conditions to simplify the expression.
8. (8 points) Show how to configure the PAL shown below to implement the logic equations.

\[ X = A'D + BCD' + AC' \quad Y = A'B' + BC + C'D \quad Z = A'D + BCD' + AC' \]

Do not simplify the equations first.

If you were to implement the same equations using a PLA, what is the \emph{smallest} number of product terms that would be required for the PLA (again, assuming that the equations are not simplified first)?
9. (12 points) The table shown below represents a 2-way set associative cache (combined instruction and data cache). The ‘<’ and ‘>’ symbols in the center point toward the least recently used of the two entries in a given row. So for example, the right entry in row 4 has been used more recently than the left entry.

<table>
<thead>
<tr>
<th></th>
<th>tag</th>
<th>data</th>
<th></th>
<th>tag</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0a4</td>
<td>401f</td>
<td>&lt;</td>
<td>0</td>
<td>0a5</td>
</tr>
<tr>
<td>1</td>
<td>b23</td>
<td>1001</td>
<td>&lt;</td>
<td>1</td>
<td>0a5</td>
</tr>
<tr>
<td>2</td>
<td>0a4</td>
<td>a01d</td>
<td>&gt;</td>
<td>2</td>
<td>0a5</td>
</tr>
<tr>
<td>3</td>
<td>0a4</td>
<td>c012</td>
<td>&gt;</td>
<td>3</td>
<td>20b</td>
</tr>
<tr>
<td>4</td>
<td>20c</td>
<td>1025</td>
<td>&lt;</td>
<td>4</td>
<td>20b</td>
</tr>
<tr>
<td>5</td>
<td>20c</td>
<td>0001</td>
<td>&lt;</td>
<td>5</td>
<td>102</td>
</tr>
<tr>
<td>6</td>
<td>302</td>
<td>a01d</td>
<td>&gt;</td>
<td>6</td>
<td>b23</td>
</tr>
<tr>
<td>7</td>
<td>c25</td>
<td>6c5d</td>
<td>&gt;</td>
<td>7</td>
<td>1fc</td>
</tr>
<tr>
<td>8</td>
<td>b23</td>
<td>4738</td>
<td>&gt;</td>
<td>8</td>
<td>0a5</td>
</tr>
<tr>
<td>9</td>
<td>acb</td>
<td>c012</td>
<td>&lt;</td>
<td>9</td>
<td>20b</td>
</tr>
<tr>
<td>a</td>
<td>07b</td>
<td>1025</td>
<td>&gt;</td>
<td>a</td>
<td>20b</td>
</tr>
<tr>
<td>b</td>
<td>20c</td>
<td>0001</td>
<td>&gt;</td>
<td>b</td>
<td>20b</td>
</tr>
<tr>
<td>c</td>
<td>0a3</td>
<td>2eeb</td>
<td>&lt;</td>
<td>c</td>
<td>20b</td>
</tr>
<tr>
<td>d</td>
<td>0c2</td>
<td>25e5</td>
<td>&gt;</td>
<td>d</td>
<td>0a4</td>
</tr>
<tr>
<td>e</td>
<td>20c</td>
<td>1001</td>
<td>&lt;</td>
<td>e</td>
<td>0a3</td>
</tr>
<tr>
<td>f</td>
<td>0a4</td>
<td>a01d</td>
<td>&gt;</td>
<td>f</td>
<td>0a5</td>
</tr>
</tbody>
</table>

Show how the state of the cache changes, when the processor fetches and executes the two instructions shown below, assuming that a new value replaces the least recently used entry. Assume that cache entries are selected using the low order 4 bits of the memory address and that the tag consists of the high order 12 bits. Be sure to update the ‘<’ and ‘>’ symbols where appropriate.

<table>
<thead>
<tr>
<th>address</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td>b236</td>
<td>1002</td>
</tr>
<tr>
<td>b237</td>
<td>ac2d</td>
</tr>
</tbody>
</table>

What is the value in the accumulator, after the above two instruction complete?

What is the value of the instruction register after the next instruction executes.
10. (20 points) The VHDL module defined below looks for the longest string of consecutive 1s in the input and outputs the length of this longest string. More specifically, the result output is the number of 1s in the longest string seen so far (since reset went low). Complete the block diagram/schematic on the following page so that it corresponds to the circuit specified by the VHDL module. Do this by adding wires and simple gates only.

```
entity maxRunner is port (  
  clk, reset, A : in std_logic;  
  result: out std_logic_vector(3 downto 0));
end maxRunner;

architecture a1 of maxRunner is
signal run, maxRun: std_logic_vector(3 downto 0);
begin
  process (clk) begin
    if clk'event and clk = '1' then
      if reset = '1' then
        run <= "0000"; maxRun <= "0000";
      elsif A = '0' then
        run <= "0000";
      else
        run <= run + '1';
        if run = maxRun then
          maxRun <= run + '1';
        end if;
      end if;
    end if;
  end process;
  result <= maxRun;
end a1;
```
What is the smallest clock period for which this circuit is guaranteed to have no internal setup time violations. Assume that all gates have a 1 ns delay, that flip flops have a setup time of 1.5 ns and a propagation delay of 2.5 ns and that there is zero clock skew. Also assume that the equality comparison circuits are implemented using exnor gates with outputs and-ed together and that the increment circuit is implemented using ripple-carry.