There are 6 questions, with the point values as shown below. You have 75 minutes with a total of 75 points. Pace yourself accordingly.

This exam must be individual work. You may not collaborate with your fellow students. However, you are permitted one page of notes.

I certify that the work shown on this exam is my own work, and that I have neither given nor received improper assistance of any form in the completion of this work.

Signature:

<table>
<thead>
<tr>
<th>#</th>
<th>Question</th>
<th>Points Earned</th>
<th>Points Possible</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Combinatorial Logic</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>Sequential Logic</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>FSMs</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>Asm Programming</td>
<td></td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>Datapaths</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>Memory Hierarchy</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td></td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>Percent</td>
<td></td>
<td>100</td>
</tr>
</tbody>
</table>
Question 1 Combinatorial Logic [15 pts]

Given the following truth-table:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

1. Write the sum-of-product formula

2. Simplify the formula

3. Write VHDL that implements the formula

```vhdl
entity q1 is
  port (  
    a : in std_logic;
    b : in std_logic;
    c : in std_logic;
    x : out std_logic);
end q1;
architecture basic of q1 is
begin

end basic;
```
Question 2 Sequential Logic [10 pts]

Consider the following VHDL fragment, in which x, and y are inputs, z is an output, and there are two DFFs (a and b) whose d inputs are a_d and b_d respectively, and whose q outputs are a_q and b_q respectively:

\[
\begin{align*}
    b_d &\leq a_q \text{ nand } x; \\
    a_d &\leq (b_q \text{ xor } a_q) \text{ nor } y; \\
    z &\leq a_q \text{ or } (\text{not } b_q);
\end{align*}
\]

Complete the waveform below (assume that the DFFs are triggered by the rising edge of clk): Note that a_q is initially 1, and b_q is initially 0:

\[
\begin{array}{c}
    \text{clk} \\
    \text{x} \\
    \text{y} \\
    \text{a_d} \\
    \text{a_q} \\
    \text{b_d} \\
    \text{b_q} \\
    \text{z}
\end{array}
\]
Question 3 FSMs [10 pts]

Draw a state machine diagram for a finite state machine which accepts a single bit input (either 0 or 1—you can just label each edge with 0 or 1). This state machine also has a single bit of output, which is initially 0.

- Whenever the FSM receives an input of three *consecutive* 1s, the output goes to 1 (after the third one is read).

- The output remains at 1 until at least three 0s (even if *consecutive*) are received as input by the FSM, at which point the output returns to 0.

- The process then repeats (another three consecutive 1s return the FSM’s output to a 1, etc).

Label each state with the bit it outputs. Be sure to indicate your start state (with an arrow to it from nowhere).
Question 4 Asm Programming [20 pts]
Translate the following C function to MIPS assembly. Answer on the next 2 pages where you have each C-code line written out for you with space to write the MIPS assembly for that line directly under it.

```c
void randomize(int * ptr, int n) {
    int i = 0 ;
    while (i < n) {
        int idx = randomNumTo(n);
        int temp = ptr[i];
        int temp2 = ptr[idx];
        ptr[i] = temp2;
        ptr[idx] = temp;
        i++;
    }
    return count;
}
```

Answer on next 2 pages
void randomize(int * ptr, int n) {

    int i = 0 ;

    while (i < n) {

        int idx = randomNumTo(n);

        int temp = ptr[i];

        int temp2 = ptr[idx];

        ptr[i] = temp2;
    }
ptr[idx] = temp;

i++;

} } return count;
Question 5 Datapaths [10 pts]

Consider the following multi-cycle data path:

1. Modify the above datapath to support the `lb $rt, imm($rs)` instruction. This instruction loads a byte from memory and sign-extends it. In modifying your datapath, you must ensure that you do not remove its capability to perform any instructions it already supports (e.g., `lw`). Clearly label any control signal(s) that you introduce.

2. Fill in the table below to show the control signals required in each cycle to execute the `lb` instruction. If the control signal is “don’t care” in a given cycle, write an X. If you added any control signals, put them in the empty columns at the right side of the table.

   - Mux selectors have 0 for the top input, 1 for the bottom.
   - Write enables (we) are 0 for disabled, 1 for enabled
   - For Op, you can write down the symbol for the mathematical operation you want (+, -, *, <<, etc).

<table>
<thead>
<tr>
<th>Cyc</th>
<th>PCwe</th>
<th>RFwe</th>
<th>BS</th>
<th>Op</th>
<th>DMwe</th>
<th>Br</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8
Question 6 Memory Hierarchy [10 pts]

Suppose that you have a memory hierarchy with the following caches:

- **L1 Data Cache** 1 cycle hit, 10% miss rate
- **L2 Cache** 10 cycle hit, 10% miss rate
- **Main Memory** 200 cycle latency

- What is the average access latency of the L1 Data Cache (in cycles)?

- Suppose that the processor’s clock frequency were doubled, what would the new average access time of the L1 Data Cache be (in cycles)?

- Instead of doubling the clock frequency, suppose that an L3 cache were added with a hit latency of 50 cycles. What hit rate is required to make the average access time of the L1 data cache 3 cycles?