There are 7 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. You may use 1 sheet of notes you created, but no other external resources.

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>C Programming</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>Boolean Algebra + Gates</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>Flipflops</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>Performance</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>Datapaths</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>Finite State Machines</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>Short-answer</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: C Programming [15 pts]

Given the following type-definition for a binary search tree node:

```c
struct _bst_node {
    int key;
    struct _bst_node * left;
    struct _bst_node * right;
};
typedef struct _bst_node bst_node;
```

Write the function `findLargestNotLargerThan` which returns the largest item in the tree which is less than or equal to the requested number. For example, if the tree contained 1, 2, 5, 8, 9, and 32 then if the largest item not larger than 6 were requested, 5 should be returned. Your solution should take advantage of the ordering of the bst.

```c
int findLargestNotLargerThan(bst_node * root, int target) {
```
Question 2: Boolean Algebra + Gates [10 pts]

For the following truth table

- Write the formula for the truth table in sum-of-products form.
- Simplify the formula as much as possible (Note: You can simplify it to a point where each of A, B, and C appear in the formula exactly once).
- Draw the logic gates which correspond to your simplified formula.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Out</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>1</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>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Question 3: Flipflops [10 pts]
Consider the following circuit (assume all flip-flops are positive edge triggered):

Draw the waveforms (showing the value for each signal at each time) for this circuit. Assume that A and D start at 1, and B and C start at 0.
Question 4: Performance... [10 pts]

Suppose that there are 3 processor designs to choose from, with the following characteristics:

<table>
<thead>
<tr>
<th>Name</th>
<th>CPI</th>
<th>Clock Freq (MHz)</th>
<th>Power (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>2.0</td>
<td>2400</td>
<td>120</td>
</tr>
<tr>
<td>B</td>
<td>5.0</td>
<td>3500</td>
<td>100</td>
</tr>
<tr>
<td>C</td>
<td>3.0</td>
<td>3000</td>
<td>50</td>
</tr>
</tbody>
</table>

All 3 processors execute the same ISA, so instructions/program is constant. Which processor would you select in each of the following situations? Justify your answer by stating what metric is appropriate and showing how each processor compares on that metric.

1. You are designing a high-end video game console. Your main design concern is total performance from one single chip.

2. You are purchasing systems for a large, cooling constrained data center. Your goal is to maximize aggregate performance of many chips and you are constrained by the total heat generation your cooling system can dissipate.
Question 5: Datapaths [10 pts]

RISC ISAs tend to be load-store ISAs—loads and stores are the only instructions which access data memory, and that is all that they do. Suppose that we were to want to add an instruction to the MIPS ISA (and thus datapath) which violated this principle. This new (made up) instruction beqzm imm($rs), $rt has the semantics:

\[
\text{if } (\text{rt} == 0) \text{ then } \text{PC} = \text{Mem}[$rs + \text{imm}] \\
\text{else } \text{PC} = \text{PC} + 4
\]

1. Show the datapath modifications required to the basic multi-cycle MIPS data-path we have been working with in lecture. If you add any new structures requiring control signals (e.g., muxes), you should label the control signal that you add for part 2. (Note that a subset of the existing control signals are already shown. You do NOT need to add in all of the other control signals for the baseline datapath). Note that the pipeline registers (which separate the stages) are drawn in a grey background to make it easier to see the stages.

Note: $rs$ is read out into the A register, and $rt$ is read out into the B register.
2. Fill in the table of control signals below for each cycle of the \texttt{beqzm} instruction you just added. You should add any extra control signals that your additions required into the blank columns. Note that there are perfectly correct answers which do not use all of the columns. Also, note that you should be able to accomplish \texttt{beqzm} in 5 cycles, but a sixth row is drawn in case you need it. The first two rows (read IMEM, and decode instruction/read register file) are done for you. A brief description of the control signals follows the table for you to answer in:

\begin{tabular}{|c|c|c|c|c|c|c|c|c|c|}
\hline
Cycle & PCwe & BS & Op & Br & V & & & & \\
\hline
1 & 0 & X & X & X & X & X & X & X & X \\
2 & 0 & X & X & X & X & X & X & X & X \\
3 & & & & & & & & & \\
4 & & & & & & & & & \\
5 & & & & & & & & & \\
6 & & & & & & & & & \\
\hline
\end{tabular}

Control signals already shown are:

- **PCwe** PC write enable. 0 = do not update PC on clock edge. 1 = update PC on clock edge
- **BS, V** Mux selectors 0 = top input, 1 = bottom input
- **Op** Alu Operation: 0 “add”, 1 for “sub”, 2 for “test if A input (top) is equal to zero”, 3 for “test if B input is equal to zero”, or you can note any other reasonable math function you need and assign it a number (write it below the table).
- **Br** This is the second input of the AND gate that controls the mux which selects the next PC.
Question 6: Finite State Machines [10 pts]

For this question, you will be creating a state machine 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 1. This state machine should keep track of the number of 1s that it has seen, and output a 1 whenever the number of ones it has seen is a multiple of 3 (0, 3, 6,...), and a 0 at all other times.

1. Draw the state machine diagram for this state machine. Label each state with a unique name (e.g. S0, S1, S2) and note which state(s) cause the output to be 1.

2. Suppose you want to implement this state machine using a one-hot encoding. This encoding means there is one D-flip-flop per state, and we will assume they are named after the states, with an input named _d and an output named _q (for example, S0_d is the input of the S0 DFF, and S0_q is the output of the S0 DFF). Pick any one state in your diagram and write the formula for its DFF’s _d input in terms of the _q outputs of the DFFs and the input value (I). Your answer should have this general form: S5_d = (S8_q and not I) or (S2_q and I).
Question 7: Short-answer [10 pts]

1. Explain the performance trade-offs between a single-cycle data path and a multi-cycle datapath. Specifically, name one advantage and one disadvantage that a multi-cycle data-path has over a single cycle data path, and state why we expect a multi-cycle datapath to generally be faster in terms of overall performance:

   Advantage

   Disadvantage

   Why faster overall?

2. Explain why level-triggered storage is problematic.

3. What type of storage do we use instead of level-triggered? How does this type of storage fix the problem(s)?