This exam is closed book and note. You may use one double-sided sheet of notes, but no magnifying glasses! Calculators, computers, and cell phones are not permitted. Slide rules, abaci, sextants, astrolabes, and sundials are permitted—in fact, they are encouraged.

This exam has 70 points. Basically, 1 point per minute plus 10 minutes of buffer. You should treat the point system as a timing mechanism. If you see a 3-point question, ask yourself “Can I answer this question in 3 minutes?”. If the answer is “no”, you may want to go on to another question.

Please ensure that your answers are legible. And please show your work.

Finally, please answer the question that is asked. If I give you 1 inch of space in which to write the answer, it’s because I am looking for a very short specific answer to a very specific question.

<table>
<thead>
<tr>
<th>Question</th>
<th>Topic</th>
<th>Max</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Performance</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pipelining</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Instruction Sets</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Integer Arithmetic</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Verilog</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Feedback</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>68</td>
<td></td>
</tr>
</tbody>
</table>
1. [1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 = 10 Points] **Performance.**

   (a) [1 point] Processor A has a CPI of 2 and a clock frequency of 2 GHz. What is its MIPS (millions of instructions per cycle)?

   (b) [1 point] Processor B has a CPI of 3 and a clock frequency of 2.5 GHz. Which processor is likely to have the deeper pipeline, A or B?

   (c) [1 point] Which processor has higher overall performance?

   (d) [1 point] What is one mechanism by which Moore’s Law, the continued miniaturization of MOS transistors, results in improved performance?

   (e) [1 point] What is another mechanism by which Moore’s Law results in improved performance?

   (f) [1 point] What is a microbenchmark?

   (g) [1 point] Microbenchmarks should *not* be used to report machine performance. Why?

   (h) [1 point] In a given design, different transistors will have different “widths.” What is an advantage of making a transistor wider?

   (i) [1 point] What is a disadvantage of making a transistor wider?

   (j) [1 points] In a given design, all transistors have the same minimal “length” (which is a characteristic of the technology). Why?
2. [2 + 1 + 1 + 1 + 3 + 2 + 4 + 1 + 1 + 1 + 2 = 20 Points] **Pipelining.**

(a) [2 points] Consider a five stage pipeline. Doubling the number of stages in the pipeline to ten is likely to increase the clock frequency, but not quite to double it. What are two reasons for this?

(b) [1 point] Whereas doubling the number of stages is likely to increase clock frequency by less than a factor of two, it is likely to increase MIPS by even less than that. Why?

(c) [1 point] What are *software interlocks*?

(d) [1 point] What is the disadvantage of software interlocks relative to hardware interlocks?

(e) [1 point] Most instruction sets have multi-cycle arithmetic operations like multiply. In a typical pipelined implementation, multi-cycle instructions are typically split off the main pipeline at execute and reconverge with it at writeback. What is the advantage of this organization as opposed to putting those operations in the main line of the pipeline?
(f) [3 points] Consider a FDXMW pipeline with only single cycle arithmetic operations. The pipeline has no MX bypassing, only WX and WM bypassing. Fill out the pipeline diagram for the code below.

<table>
<thead>
<tr>
<th>Cycle</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>8</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>12</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sub</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>3</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>16</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>slti</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td>512000</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>beq</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td>1</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(g) [2 points] Write out the stall logic for this pipeline.

(h) [4 points] Reschedule the code to reduce data hazards and fill in the pipeline diagram. Remember, there is no MX bypassing. Also, the branch has to remain at the end.
(i) [1 point] What is an advantage of a tagged BTB (branch target buffer) over an untagged BTB?

(j) [1 point] A return address stack (RAS) cannot be tagged. Why?

(k) [1 point] Consider the following accuracy results for a tagged BTB with 8, 16, 32, 64, and 128 entries. The “Cond” column is the accuracy for conditional branches, “Return” is the accuracy for return instructions, “Other” is the accuracy for all control transfers other than conditional branches and returns, and “Overall” overall accuracy.

<table>
<thead>
<tr>
<th>BTB</th>
<th>Cond</th>
<th>Return</th>
<th>Other</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>50%</td>
<td>0%</td>
<td>40%</td>
<td>46%</td>
</tr>
<tr>
<td>16</td>
<td>54%</td>
<td>0%</td>
<td>46%</td>
<td>50%</td>
</tr>
<tr>
<td>32</td>
<td>68%</td>
<td>0%</td>
<td>78%</td>
<td>69%</td>
</tr>
<tr>
<td>64</td>
<td>73%</td>
<td>0%</td>
<td>96%</td>
<td>78%</td>
</tr>
<tr>
<td>128</td>
<td>73%</td>
<td>33%</td>
<td>97%</td>
<td>79%</td>
</tr>
</tbody>
</table>

What is a possible explanation for return accuracy jumping from 0% to 33% when BTB size is increased from 64 to 128 entries?

(l) [2 points] The fact that conditional branch prediction accuracy appears to saturate at 73% suggests that conditional branches are harder to predict than other forms of control instructions. Why is this?
3. $[1 + 1 + 3 + 2 + 1 + 1 + 1 + 2 + 2 + 3 = 20$ Points] **Instruction Sets**

(a) [1 point] What is one advantage of fixed-length instructions?

(b) [1 point] What is another advantage of fixed-length instructions?

(c) [3 points] One measure of the quality of an instruction set is the degree to which it facilitates or, at least does not limit, compiler optimizations. One of the important goals of a compiler is to reduce the number of loads in a program. What is one specific optimization that achieves this and what does it do? Give a code example.

(d) [2 points] Which ISA feature is a greater impediment to reducing the number of loads in a program: accumulator-style instructions that over-write one of their inputs or a small number of registers? Explain.

(e) [1 point] What is one consideration that limits the number of registers an ISA can have?

(f) [1 point] What is another one?
(g) [1 point] Early instruction sets, like x86, were CISC and included many complex high-level instructions. What was the rationale for this?

(h) [1 point] What event led to the development of RISC instruction sets?

(i) [2 points] Although RISC ISAs have technical advantages over CISC ones, the dominant ISA for high performance processors today is a CISC, x86. What technological force has helped x86 survive, and how?

(j) [2 points] What economic force helped x86 survive, and how?

(k) [3 points] What is a uISA (micro ISA)? What problem does a uISA solve?
4. [1 + 2 + 1 + 1 + 1 + 1 + 3 = 10 Points] **Integer Arithmetic**

(a) [1 points] In a real world pipeline, clock period is typically determined by the latency of what?

(b) [2 points] Consider a 32-bit adder that uses a three piece 10-10-12 carry select design. What is the latency of this adder in gate delays?

(c) [1 point] How many 1-bit full-adders does this design use?

(d) [1 point] True or false. A carry-lookahead (CLA) adder has a delay that is linear in the number of bits to be added.

(e) [1 point] What does it mean for an N-cycle multiplier to be “pipelined”?

(f) [1 point] One way to pipeline a multiplier is simply to implement it as a giant Wallace tree and then cut that tree up into stages. Consider an 8-bit by 8-bit multiplier with a 16-bit product. A combinational implementation of this multiplier uses a 7-gate delay conventional adder and carry-save adders (CSA). How many CSAs are needed?

(g) [3 points] What is the minimum number of layers in which these CSAs would need to be arranged? As a result, what is the gate delay of this multiplier? Remember the muxes and conventional adder.
5. \[1 + 1 + 2 + 1 + 1 + 1 = 7\]\ points\] Hardware Design and Verilog

(a) \[1\] point\] What is the difference between \textit{structural} Verilog and \textit{behavioral} Verilog?

(b) \[1\] point\] What is the Verilog Programming Interface? What is it used for?

(c) \[2\] points\] When synthesizing a multiplier to an FPGA, it is preferred to specify the multiplier as \textit{*} rather than as a collection of gates and wires. What are two reasons for this?

(d) \[1\] point\] Consider the following Verilog declaration.

\begin{verbatim}
wire [15:0] A, B, C, D;
assign B = {8'h00, A[7:0]};
\end{verbatim}

In English, what does the following line of Verilog synthesize to?

\begin{verbatim}
C = \neg A + 1;
\end{verbatim}

(f) \[1\] point\] What does the following line of Verilog synthesize to?

\begin{verbatim}
\end{verbatim}
6. [3 points] **Feedback and Evaluation.**

(a) [1 point] On a scale of 1-10, how difficult was this exam?

(b) [1 point] How many hours did you spend studying for this exam?

(c) [1 point] Which of these did you feel were most useful in terms of studying: attending lectures? reading the notes after lecture? reading the book? homeworks? office hours? old exams? the review session?