EECS 470 Final Exam
Winter 2004

Name: __________________________ unique name: __________

Sign the honor code:

I have neither given nor received aid on this exam nor observed anyone else doing so.

______________________________

Scores:

<table>
<thead>
<tr>
<th>#</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 1</td>
<td>/40</td>
</tr>
<tr>
<td>2.1</td>
<td>/10</td>
</tr>
<tr>
<td>2.2</td>
<td>/15</td>
</tr>
<tr>
<td>2.3</td>
<td>/20</td>
</tr>
<tr>
<td>2.4</td>
<td>/15</td>
</tr>
<tr>
<td>Section 2</td>
<td>/60</td>
</tr>
<tr>
<td>Total</td>
<td>/100</td>
</tr>
</tbody>
</table>

NOTES:

1. Open book and Open notes
2. Calculators are allowed, but no PDAs, Portables, Cell phones, etc.
3. Don’t spend too much time on any one problem.
4. You have about 120 minutes for the exam.
5. Be sure to show work and explain what you’ve done when asked to do so.
6. The last page is a copy of the answer template for question 2.3. If you want that graded rather than the template in question 2.3 you will need to CLEARLY indicate that on the template for question 2.3!
Section I – Short answer – 40 points

1. Consider a standard 5-stage pipeline where:
   - Branches are resolved in the EX stage
   - All branches are predicted not taken
   - All data hazards which can be dealt with by forwarding to the EX stage are implemented (and no others)
   - The CPI would be 1.0 if it were not for control or data hazards

The program running on the machine has the following characteristics:
   - 20% of all instructions are branches.
   - 40% of all branches are taken
   - 20% of all instructions are loads
   - 25% of all loads are dependent on the instruction in front of them
   - 10% of all instructions immediately following a load are dependent on that load.
   - 20% of all instructions are stores
   - The program is quite long (millions of instructions)

What is the CPI that would be achieved by the machine on the program described? Show your work. [7]

2. One difference between a BTB and a cache is that it is acceptable to use a partial tag for a BTB but you cannot do so for a cache. In 25 words or less, explain why this is the case. [5]
3. Consider two caches. Both are 4 cache lines in size and each cache line is 16 bytes and both start out with all lines marked invalid. The only difference is one is direct-mapped and one is two-way associative. [8]
   a. Find the shortest reference stream where the 2-way associative cache would get a hit and the direct-mapped cache would get no hits. Provide the reference stream in hex.

   b. Find the shortest reference stream where the direct-mapped cache would get a hit and the 2-way associative cache would get no hits. Provide the reference stream in hex.

4. In Verilog, what is the difference between a reg and a wire? Be clear and complete. [5]
5. Fill-in-the-blank or circle the correct answer. [15 points, -1 for each wrong or blank answer, minimum of zero]

a. Test-and-set is an ________ operation, meaning that the test (read) and set (write) happen back-to-back without any operation occurring between the read and write.

b. Comparing the MESI protocol to the MSI protocol, the only time the “E” state helps is when the line is in the E state and a Read or Write is performed. Under the MSI protocol the processor would have been in the M or S or I state.

c. A directory based scheme usually indicates that the machine a NUMA or UMA machine and SIMD or MIMD or SISD. In the directory scheme discussed in class the initiating node must send a request to the home node unless ____________________.

d. Tomasulo’s original algorithm was created for a computer that was built in the 60s or 70s or 80s or 90s.

e. Given the access pattern A, B, C, D, A where each letter corresponds to a unique cache block and there were no accesses to any block before this, the stack distance of the first A is ________ while the stack distance of the second A is ________.

f. As wires get narrower their resistance goes Up or Down. As wires get shorter their resistance goes Up or Down.

g. Adding architected or physical registers can provide the compiler more flexibility in avoiding __________ dependencies

h. In IA-64 a ______________ load can be safely moved above a branch.

i. If the multiplier is in the critical path for the clock cycle, decreasing the number of pipeline stages used to do a multiply can be expected to increase or decrease or not effect the clock period while increasing or decreasing or not effecting the CPI.

j. A ______________ predictor actually consists of two (or more) predictors, where one is usually global and the other local. The predictor that chooses which predictor to use only is updated when the two underlying predictors ________________.

k. If the BTB is in the critical path for the clock cycle, making it more associative can be expected to increase or decrease or not effect the clock period while increasing or decreasing or not effecting the CPI.
Section II – Longer answer – 60 points

2.1. Consider the following program segment:

```
R2=R3+R4             //A
R3=MEM[R2+8]         //B
R7=R3+R2             //C
R6=R2+R1             //D
If(R4<1000)goto next //E
R10=R9+3             //F
R6=MEM[R2+40]        //G
Next:
R9=R6+12             //H
```

Perform static *reordering* of the code so that the loads are as far from their first use as possible while insuring that the new program segment behaves the same as the original program segment. You may not change the instructions in any way nor add or subtract instructions; you may only reorder them. In the event that a given change “helps” one load and harms another, make the choice that helps the load that is always executed. [10]
2.2. Consider the following programs

**Program A**

R1 = MEM[R2 + 0]  
R2 = R1 + 4  
if (R2 > 8) goto bob  
R3 = R1 + R2  

**Bob:**

R4 = R3 + 6  
R5 = R1 + R4  
R6 = R2 + R5  
R7 = R5 + 19  
R8 = R7 + R6  
R9 = R6 + 4

**Program B**

R1 = MEM[R2 + 0]  
R2 = R3 + 4  
if (R2 > 8) goto bob  
R3 = R4 + R1  

**Bob:**

R4 = R3 + 6  
R5 = R1 + R8  
R6 = R3 + R10  
R7 = R1 + 19  
R8 = R7 + R3  
R9 = R4 + 4

Say we have an out-of-order machine (Using Tomasulo’s II and not retiring branches until the branch is at the head of the ROB) with 4 general RSs and 8 ROB entries. Both branches are actually not taken but are predicted taken and the BTB provides the correct address for “bob”. Further, say the first load stalls for a long time. Both programs will end up stalling, waiting for the load to finish. For each program indicate the last instruction that will be placed in the ROB previous to the load finishing. You are to assume an instruction stays in its RS until it completes execution. Be sure to clearly show/explain your work. [15]
2.3. Provided on the following page is a register address table (RAT), a ROB, a set of physical registers, and a free-list queue for the PRF all as found at some fixed point in time $T_0$.

At time $T_0$:
- None of the instructions shown (A-E) have yet issued.
- Instruction A is next in line to be issued.

At some later time $T_1$:
- All instructions (A-E) have issued.
- Instructions A, B and D have completed execution.
- All instructions which could have retired (committed) have done so. That is, there are no additional instructions at this time that could retire until something else completes execution.

Redraw the RAT, ROB and PRF as they would appear at time $T_1$. Also indicate the head and tail of the ROB and the current state of the PRF free queue at time $T_1$. [20]

Code (also found on the following page)

$$R1 \leftarrow R2 + R1 \quad \text{// A}$$
$$R2 \leftarrow R3 + R1 \quad \text{// B}$$
$$R2 \leftarrow R2 + R1 \quad \text{// C}$$
$$R1 \leftarrow R1 + R1 \quad \text{// D}$$
$$R3 \leftarrow R2 + R1 \quad \text{// E}$$

An extra copy of page 8 is found at the end of the exam. If you use it to answer the question you must clearly indicate this on page 8!
ROB starts empty, tail=1. PRF free queue=4, 5, 0 where 4 is the head of the queue. The smaller box in the ROB Back Pointer field should be checked if execution is completed.

R1 \leftarrow R2 + R1 \quad // A
R2 \leftarrow R3 + R1 \quad // B
R2 \leftarrow R2 + R1 \quad // C
R1 \leftarrow R1 + R1 \quad // D
R3 \leftarrow R2 + R1 \quad // E

ROB head:
ROB tail:
PRF free queue, starting with the head:
2.4. Consider a case of having 3 processors using a snoopy MESI protocol where the memory can snarf data. All three have a 2 line direct-mapped cache with each line consisting of 16 bytes. The caches begin with all lines marked as invalid. Fill in the following table indicating

- If the processor gets a hit or a miss in its cache
- What bus transaction(s) (if any) the processor performs (BRL, BWL, BRIL, BIL)
- What supplies the data

Finally, indicate the state of the processor after all of these memory operations have completed. The operations occur in the order shown. [15 points, -2 per 3 boxes wrong or blank minimum of zero]

<table>
<thead>
<tr>
<th>Processor</th>
<th>Address</th>
<th>Read/Write</th>
<th>Hit/Miss</th>
<th>Bus transaction</th>
<th>Data supplied by:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0x14</td>
<td>Read</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0x24</td>
<td>Write</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0x48</td>
<td>Read</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0x18</td>
<td>Write</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0x40</td>
<td>Write</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0x28</td>
<td>Read</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0x30</td>
<td>Read</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0x10</td>
<td>Write</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0x30</td>
<td>Read</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Proc 1**

<table>
<thead>
<tr>
<th>Address</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set 0</td>
<td></td>
</tr>
<tr>
<td>Set 1</td>
<td></td>
</tr>
</tbody>
</table>

**Proc 2**

<table>
<thead>
<tr>
<th>Address</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set 0</td>
<td></td>
</tr>
<tr>
<td>Set 1</td>
<td></td>
</tr>
</tbody>
</table>

**Proc 3**

<table>
<thead>
<tr>
<th>Address</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set 0</td>
<td></td>
</tr>
<tr>
<td>Set 1</td>
<td></td>
</tr>
</tbody>
</table>
This is a copy of the answer sheet for problem 4 of section 2.

<table>
<thead>
<tr>
<th>Register Rename Table</th>
<th>ROB</th>
<th>PRF</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Register#</strong></td>
<td><strong>Location</strong></td>
<td><strong>ROB#</strong></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ROB starts empty, tail=1. PRF free queue=4, 5, 0 where 4 is the head of the queue. The smaller box in the ROB Back Pointer field should be checked if execution is completed.

```plaintext
R1  R2 + R1 // A
R2  R3 + R1 // B
R2  R2 + R1 // C
R1  R1 + R1 // D
R3  R2 + R1 // E
```

<table>
<thead>
<tr>
<th>Register Rename Table</th>
<th>ROB</th>
<th>PRF</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Register#</strong></td>
<td><strong>Location</strong></td>
<td><strong>ROB#</strong></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ROB head:
ROB tail:
PRF free queue, starting with the head: