Prof. Mark D. Hill

Midterm Examination I
In-Class
Wednesday, February 26, 2014
Weight: 25%

1:15 minutes.

CLOSED BOOK, etc., but one cheat sheet allowed (two-sided 8.5x11 page).
The exam is two-sided and has EIGHT pages, including two blank pages at the end.
Plan your time carefully, since some problems are longer than others.

NAME: __________________________________________________________

ID# __________________________

<table>
<thead>
<tr>
<th>Problem Number</th>
<th>Maximum Points</th>
<th>Actual Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>60</td>
<td></td>
</tr>
</tbody>
</table>
Problem 1: Parallel Programming (12 points)

(a) OpenMP allows loop-level parallelism where parallel loop iterations are **statically or dynamically scheduled**. What is static scheduling? What is dynamic scheduling? Under what circumstances is static scheduling preferred to dynamic? Under what circumstances is dynamic preferred to static?

(b) **Parallel Random Access Machine (PRAM)** provides a model of computation. What is the PRAM model? Explain three (or more) different, important ways that PRAM is an inaccurate model of a shared-memory multiprocessor using snooping cache coherence.
Problem 2: Consistency (12 points)

Assume all memory and register values are initially zero. Each execution of the following code leaves registers \((r_1, r_2, r_3, r_4)\) with a set of values \((?, ?, ?, ?)\), but different executions leave different values.

<table>
<thead>
<tr>
<th>Processor P1</th>
<th>Processor P2</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1: (x = 2;)</td>
<td>S2: (y = 3;)</td>
</tr>
<tr>
<td>L1: (r_1 = x;)</td>
<td>L2: (r_3 = y;)</td>
</tr>
<tr>
<td>L2: (r_2 = y;)</td>
<td>L4: (r_4 = x;)</td>
</tr>
</tbody>
</table>

(a) For all alternative executions allowed by *sequential consistency (SC)*, what are the final registers values? One value set is given.
\[(r_1, r_2, r_3, r_4) = (2, 3, 3, 2)\]

(b) For all alternative executions allowed by *total store order (TSO)*, what are the final registers values?
\[(r_1, r_2, r_3, r_4) = \]

(c) What is *Sequential Consistency for Data Race Free (SC for DRF)*? How is it like a strong memory (consistency) model like SC? How is like a relaxed memory (consistency) model like XC?
Problem 3: Coherence (12 points)

(a) Mesi coherence protocols includes the *Exclusive (E)* in addition to Modified (M), Shared (S), and Invalid (I). When does an Mesi protocol enter the E state? How does it leave the E state? What benefit does E state provide? What are E-state drawbacks?

(b) To perform an atomic read–modify–write instruction (e.g., test-and-set), must a core always communicate with the other cores? Why or why not?

(c) Discuss possible races with cache *writebacks* (M→I) depending on whether a system has (i) Atomic Requests and Atomic Transactions or (ii) just Atomic Transactions. Hint: recall transient state $MI^A$. 
Problem 4: From Your Reader (12 points)

(a) Hill and Marty [Computer 2008] provide three models for multicore chips. What are they and how do they differ?

(b) Compare and contrast the data parallel programming model advocated by Hillis and Steele [CACM 1986] with the shared memory programming model?

(c) Charlesworth [Micro 1998] describes the Sun E10000. What interconnect(s) does the E10000 use to send coherence requests and receive data responses? How does coherence work since it doesn’t use a single shared bus?
Problem 5: MCS Queued Lock (12 points)

(a) What are the advantages and disadvantages of an MCS queue-based lock (pseudo-code above) vs. a test-and-test-and-set lock?

(b) An initially free lock L will start with memory address L having this initially null linked list:

L [null]

1. Let thread 1 provide a qnode at address A to obtain lock L. Draw the new linked list state (including pointers and values within the qnode).
   L [ ]

2. Let thread 2 provide a qnode at address B to queue for lock L. Draw the new linked list state.
   L [ ]

3. Let thread 1 release lock L and thread 2 obtain it. Draw the new linked list state.
   L [ ]
Scratch Sheet 1 of 2 (in case you need additional space for some of your answers)
Scratch Sheet 2 of 2 (in case you need additional space for some of your answers)