CS/ECE 757: ADVANCED COMPUTER ARCHITECTURE II
COMPUTER SCIENCES DEPARTMENT
UNIVERSITY OF WISCONSIN—MADISON

Prof. Mark D. Hill

Midterm Examination I
In-Class
Wednesday, February 27, 2019
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 make take longer than others.

NAME: _____

ID# _____

Problem Number	Maximum Points	Actual Points
1	12	
2	12	
3	12	
4	12	
5	12	
Total	60	

Problem 1: Performance, etc. (12 points)

(a) Consider the statement, "Local operations on thread 1 of a message-passing program are not ordered with respect to local operations on thread 2 of the same program." Is this statement true or false? Why?

.

(b) Consider multiplying an n-by-n matrix A by an n-element vector x to get a n-element vector y: y = Ax. How long will take to execute "y = Ax" on the PRAM model. Why?

(c) Let time(X,N) be the time to execute a program with problem size X on N processors. Conventionally the speedup on this program with problem size X would be time(X,1)/ time(X,N). When is this ***not*** the best definition of speedup? Why? What might you do instead?

Problem 2: Synchronization (12 points)

(a) Provide pseudo-code for a simple implementation of `Lock(L)` and `Unlock(L)` using regular instructions and the compare-and-swap or `CAS()` hardware primitive. Recall from Scott13:

```
boolean CAS(word *a, word old, word new):
    atomic { t := (*a = old); if (t) *a := new; return t }
```

(b) What is deadlock? What are three ways it can be avoided?

(c) Scott13 Chapter 4 gives many implementations of a spin lock. Which one(s) would you choose if you wanted a lock to be acquired by threads in the same order that they requested it (also known as first-come-first-service (FCFS) or first-in-first-out (FIFO)? Why?

Problem 3: Snooping Coherence (12 points)

Regarding snooping coherence with some or all of MOESI states of *modified (M), owned (O), exclusive (E), shared (S),* and *invalid (I)*.

(a) Why is a coherence protocol supporting only M and I usually a bad idea?

(b) What is the purpose of the O state? Give an example of a block 100 at processor P1 transitioning ***into*** the O state.

(c) Why might block 100 in the O state at processor P1 transition next to M? Alternatively, why from O to S? Alternatively, why from O to I?

(d) With an aggressive bus implementation, a processor P1 with block 100 in state I can issue a GETM to seek to transition into state M. Why might it not transition ***immediately*** to state M? Might it never reach M? Why or why not?

Problem 4: Miscellaneous (12 points)

(a) In Arch2030, Ceze et al. identify five future trends. Describe at least two of them and why they might be important.

(b) Esmaeilzadeh et al. 2013 discuss "dark silicon." What is it? Why does it arise? Might it get worse? Why or why not?

(c). Consider a system that supports "Sequential Consistency for Data Race Free (SC for DRF)". Must the hardware support SC? Why or why not?

Problem 5: Consistency (12 points)

Consider the following example with no other code executing. Assume that memory variables `foo` and `bar` are initially `0` and `rij` means processor `Pi`'s register `j`. A – F are statement labels. Assume *write atomicity*, i.e., when a processor's write is seen by *any other* processor, it is seen by *all other* processors.

(a) What executions does sequential consistency (SC) permit? Specify an execution by its final registers values: `(r11, r12, r21, r22) = (?, ?, ?, ?).`

Processor P1	**Processor P2**
`A: foo = 3;`	`D: r21 = bar;`
`B: r11 = foo;`	`E: bar = 5;`
`C: r12 = bar;`	`F: r22 = foo;`

(b) Insert the necessary FENCEs to make a TSO implementation allow only SC executions. Why?

Processor P1	**Processor P2**
`A: foo = 3;`	`D: r21 = bar;`
`B: r11 = foo;`	`E: bar = 5;`
`C: r12 = bar;`	`F: r22 = foo;`

(c). Insert the necessary FENCEs to make a relaxed XC implementation allow only SC executions. Why?

Processor P1	**Processor P2**
`A: foo = 3;`	`D: r21 = bar;`
`B: r11 = foo;`	`E: bar = 5;`
`C: r12 = bar;`	`F: r22 = foo;`

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)