Computer Science 418 Midterm

2003 October 23

Name (1 point):____

Student ID:____

This exam is **open-book, open-notes**. You may **not** use a calculator or any other electronic device during the exam. The exam ends at 10:45am. **Stop writing immediately when time is called.** Good luck!

(Leave this table blank.)

Name (1 pt)	
Short Questions (15 pts)	
Crossbars (15 pts)	
Disk Modeling (15 pts)	
Snooping (15 pts)	
HPAutoRAID (40 pts)	
Total (out of 100 pts)	

1 Assorted Short Questions (15 pts)

Give short answers in the space provided (about one sentence of answer per question mark).

(a) Ethernet allows variable-length messages. Why is there a minimum message size (messages must be longer than some number of bits)? Why is there a maximum message size? What is the main advantage of going to fixed-length packets, and why didn't ethernet adopt that?

(b) Jouppi's paper that we read introduced stream buffers (as well as victim caches) to improve caching. What is a stream buffer and why does it help? The paper by Goodman described using caches in a multiprocessor. How well would stream buffers work in a mulit-processor system designed the way Goodman proposed (snooping)? Why? How well would stream buffers work in DASH? Why?

(c) CPU speed improves by roughly 60% per year (from Moore's Law). Patterson et al.'s RAID paper says disk speed improves roughly 7% per year. If an application currently spends half of its time computing (CPU-bound) and half of its time accessing the disk, what will that ratio be in a year? In n years? (You do not have to do the arithmetic; just show the computation expression you'd need to evaluate.)

2 Crossbar Scheduling (15 pts)

(a) How do you solve the problem of "head of line (HOL) blocking"?

(b) For uniformly random traffic (equally likely from any input to any output), how much performance do we lose (what factor) because of HOL blocking?

(c) Shah et al. mainly use a "diagonal traffic load" instead of uniformly random traffic. How much performance loss does HOL blocking cause under this traffic load?

(d) Both the McKeown and the Shah et al. papers mention "speed-up", where we run the switch at a faster rate than the arrival rate of packets at the inputs. The more complex the switch design and the scheduling algorithm, the harder it is to speed up the switch. Suppose that if we used simple input queueing (and ignored HOL blocking), we could run the switch sped-up by 50% (1.5X the bandwidth through the switch). Would we expect better or worse throughput than a switch running at 1X speed with no HOL blocking and a good scheduling algorithm?

3 Disk Modeling (15 pts)

Ruemmler and Wilkes' paper on disk drive modeling notes that seek time has both an acceleration/deceleration portion, and a constant velocity portion. Depending on which one dominates, we model the seek time as either linear in the distance of the seek, or else proportional to the square root of the distance.

(a) If you assume a seek is spending all of its time accelerating at a constant rate of acceleration r_1 and then decelerating at a constant rate of acceleration r_2, derive the formula for the seek time as a function of distance. (Assume distance is given in tracks, and the acceleration/deceleration constants are in tracks/second/second.)

(b) In practice, you might not have the luxury of measuring the performance of disk drives and then tuning your program to that. (E.g., if you write shrink-wrapped retail software, you have no idea what kind of disks your customers will be using.) Furthermore, your program will probably be able to specify only logical tracks, not physical tracks on the disk. Here's what your program could do: do a large number of random seeks (seek from one track chosen uniformly at random to another track chosen uniformly at random), and record the times each seek required. Why can you not assume the distance of each seek is the difference in logical track numbers? (Give specific reasons; it's not good enough to say just that the logical and physical tracks don't correspond.)

(c) Instead, if you just look at the distribution of the seek times you observed (without knowing how far the seek was), how can you tell whether a constant-acceleration or a constant-velocity model is more accurate?

4 Snooping (15 pts)

In the Goodman paper and in class, we saw how a snooping cache coherene protocol can work. In particular, Goodman proposed a MESI protocol. Here, we'll consider how a MOESI protocol works:

Modified — This cache has the only copy, and main memory isn't up-to-date.

Owned — Other caches have a (shared) copy of this cache line, but the copy in main memory is not up-to-date.

Exclusive — This is the only cache with a copy of this data, but main memory also has the correct, up-to-date data. (Combination of Modified and Shared)

Shared — This cache has a copy of the data, but other caches presumably do, too. Main memory may or may not (e.g., if someone else has it Owned) have an up-to-date value.

Invalid — This cache does not have a copy of the data.

A MOESI protocol makes sense if caches have a way to send data directly to one another, without involving (slower) main memory. For example, if I have address a in my cache as Modified, and I see (via snooping) some other cache trying trying to read address a, I can send the data directly to that cache (which goes to the Shared state), and then I transition to Owned.

(a) Are there any other ways to transition to the Owned state? If so, what are they? If not, why not?

(b) If a cache line is in the Owned state, what does the cache have to do if it evicts that line?

(c) If a cache line is in the Owned state, what does the cache do if it notices (by snooping) that another cache is reading that line?

(d) If a cache line is in the Owned state, what does the cache do if it notices (by snooping) that another cache is writing to that line?

(e) What is the single word that describes why we can have a lower-level cache snooping the bus without worrying about what is contained in higher-level caches?

5 HP AutoRAID (40 pts)

This problem relates to the paper "The HP AutoRAID Hierarchical Storage System" by Wilkes et al.

(a) What is the issue being addressed in this paper?

(b) What is the proposed solution? (Describe the solution. Don't just say, "the HP AutoRAID system"!)

(c) How do the authors substantiate that their solution works? Summarize their main findings.

(d) What RAID levels are used by the HP AutoRAID?

(e) Why is write-active data handled by mirroring? Explain based on the performance characteristics from Patterson et al.'s RAID paper.

(f) When does data get moved from mirroring to RAID 5?

(g) When does data get moved from RAID 5 to mirroring?

(h) What is the purpose of the NVRAM (non-volatile RAM) write buffer? Why did they make it non-volatile?

(i) On a typical PC file system, the directory maps a file to the disk blocks that are used to store it. In AutoRAID, why is there an extra level of indirection, in which a file is mapped to relocation blocks (RBs), which are then remapped onto segments?

(j) The figure below is like Figure 4 in the paper, except there are only 3 disks in the PEG:

disk 0	disk 1	disk 2
0	1	2
3	4	5
6	7	8
9	10	11

I have labeled each segment with a number.

List a set of segments in a single PEX:______________________________

List a set of segments in a mirroring pair (if this were a mirrored PEG):________________

List a set of segments in a RAID 5 stripe (if this were a RAID 5 PEG):________________