EECS 470 Final Exam
Fall 2011 - Solutions

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>1</td>
<td>/ 24</td>
</tr>
<tr>
<td>2</td>
<td>/ 27</td>
</tr>
<tr>
<td>3</td>
<td>/ 27</td>
</tr>
<tr>
<td>4</td>
<td>/ 22</td>
</tr>
<tr>
<td>Total</td>
<td>/</td>
</tr>
</tbody>
</table>

NOTES:

- Closed book.
- Calculators are allowed, but no PDAs, Portables, Cell phones, etc.
- Don’t spend too much time on any one problem.
- You have about 120 minutes for the exam (avg. 24 minutes per problem).
- There are 11 pages including this one. Please ensure you have all pages.
- Be sure to show work and explain what you’ve done when asked to do so.
1) **Short Answer [24 points]**

a) State one advantage of an inclusive cache hierarchy. [3 points]

*Coherence snoops that miss in the last level cache do not need to be propagated further up the hierarchy.*

b) State one disadvantage of an inclusive cache hierarchy. [3 points]

*Back-invalidations required to maintain inclusion increase L1 miss rates.*

c) Name a technique that can eliminate compulsory misses [3 points]

*Prefetching. In particular, stride prefetching.*

d) Describe briefly how the “critical word first” optimization improves memory system performance. [3 points]

*Mitigates the transfer time penalty of a large cache block size by transferring the requested word to the processor before the rest of the cache line.*
e) Name a data structure for which stride prefetching is typically effective. [3 points]

*Arrays. Matrices.*

f) Name a data structure for which correlation prefetching is typically effective. [3 points]

*Linked lists.*

g) Under the "4 C's" classification of cache misses, what kind of miss does a skewed-associativity cache seek to reduce? [3 points]

*Conflict misses.*

h) Does simultaneous multithreading seek to improve per-thread execution latency or overall system throughput? [3 points]

*Throughput.*
2) Memory Hierarchy [27 points]

a) Chips'n'Dip Processors is trying to improve memory system performance in their next-generation processor design. Their current design has a 64KB 2-way L1 and a 512MB 4-way L2. The L1 has a hit time of 2 cycles, the L2 access latency is 12 cycles, and main memory access latency is 250 cycles (L1, L2, and main memory accesses are performed in series). For the suite of benchmarks they use to evaluate their design, they achieve a 5% L1 miss rate and a 40% (local) L2 miss rate.

i) What is the effective access time of their current design? [5 points]

\[
\text{Effective time} = L1 \text{ hit time} + L1 \text{ miss rate} \times L2 \text{ hit time} + L1 \times L2 \text{ miss rate} \times \text{mem time}
\]
\[
= 2 + 0.05 \times 12 + 0.05 \times 0.4 \times 250 = 7.6
\]

ii) They are considering increasing the L2 to a 2MB 8-way design. The larger L2 has a 14 cycle access latency. What must the (local) L2 miss rate of the new design be to achieve a 15% reduction in overall average memory access time? [5 points]

\[
\text{Target time} = 0.85 \times 7.6 = 6.46
\]
\[
6.46 = 2 + 0.05 \times 14 + 0.05 \times \text{L2 miss rate} \times 250
\]
\[
\text{L2 miss rate} = 30%
\]
b) Chips'n'Dip processors is designing a new virtual memory system and cache hierarchy for their next generation processor. Their memory system has 8KB pages. Their 32KB L1 data cache is 4-way associative with 32 byte blocks. Can they use a virtually-indexed physically-tagged cache to perform TLB lookups in parallel with cache indexing? Why or why not? [4 points]

\[ \text{Block Size} \times \# \text{Sets} \leq \text{Page Size}, \text{ so they can use VIPT cache.} \]

c) Either way, the Chips’n’Dip design team decide to use a virtually-addressed I-cache, but a physically-indexed D-cache. Explain why the architects chose different strategies for the instruction vs. data cache. [4 points]

\[ \text{I-cache is read-only, so synonyms are not a problem. Note that this solution requires extra work to deal with self-modifying code, which we assume is rare. (3 points)} \]

\[ \text{Also, must either flush IS or include ASIDs in tags to distinguish homonyms. (1 point)} \]
Go Blue! Computers cache architects are debating among three design variations for the data cache blocks: (a) 16-byte blocks, (b) 32-byte blocks, and (c) 32-byte address, 16-byte transfer blocks (choice (c) might also be called “sub-blocked with 32-byte blocks and 16-byte sub-blocks”). In the table below, identify which design provides better miss ratio, bandwidth, and tag array size (assume that cache size remains constant; ignore writeback traffic).

**Explain why for full credit.** [9 points]

<table>
<thead>
<tr>
<th></th>
<th>Miss Ratio</th>
<th>Bandwidth</th>
<th>Tag Array Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) vs. (b)</td>
<td>(a) fewer conflicts =or= (b) spatial locality</td>
<td>(a), block size is smaller, less useless data transferred</td>
<td>(b), larger block size means fewer sets and fewer tag bits</td>
</tr>
<tr>
<td>(b) vs. (c)</td>
<td>(b) spatial locality</td>
<td>(c), transfer size is smaller, less useless data transferred</td>
<td>(b), as (c) requires extra valid bits</td>
</tr>
<tr>
<td>(a) vs. (c)</td>
<td>(a) fewer conflict misses</td>
<td>same</td>
<td>(c), as (a) has more sets</td>
</tr>
</tbody>
</table>
3) Multiprocessors [27 points]

a) Why do naïve implementations of sequential consistency lead to poor performance in multiprocessors with out-of-order cores? [3 points]

*Strict implementation of sequential consistency requires memory operations to be done one at a time in order, eliminating many of the advantages of out of order processing.*

b) How can the system described in part (a) be improved to mitigate this performance loss? [3 points]

*Either use speculation to hide ordering penalties, or switch to relaxed consistency.*

b) Briefly explain two scalability bottlenecks of bus-based snoopy multiprocessors (a.k.a. “symmetric multiprocessors”)? [6 points]

1) *Limited bandwidth on the bus.*

2) *Limited snoop bandwidth in caches*
c) Complete the state transition diagram for a four-state MESI (Modified-Exclusive-Shared-Invalid) coherence protocol for a snoopy bus-based symmetric multiprocessor. The “Exclusive” state is a state where a processor has a clean copy of a cache line, but also has permission to write the line. The protocol enters “Exclusive” state from “Invalid” when a processor issues a Read for a line that is not present in any other cache; this transition has been filled in for you. Fill in the remaining transitions (including transitions back to the same state). For each, label it with “Event => Outgoing Bus Message” or “Incoming Bus Message => Outgoing Bus Message”. [15 points]

<table>
<thead>
<tr>
<th>Processor Events</th>
<th>Bus Messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>PrRead</td>
<td>BusRead</td>
</tr>
<tr>
<td>PrWrite</td>
<td>BusReadX</td>
</tr>
<tr>
<td>PrEvict</td>
<td>BusInv</td>
</tr>
<tr>
<td></td>
<td>BusWB</td>
</tr>
<tr>
<td></td>
<td>BusReply</td>
</tr>
</tbody>
</table>

Diagram:

```
E ----> M
        |
        V
        E
```

PrRd / BusRd (no sharers)
4) Storage [22 points]

a) One of the key challenges of FLASH-based storage systems is that FLASH cells tend to wear out (no longer retain a value) after a fairly small number of write-erase cycles. Multi-level cell (MLC) FLASH devices wear out particularly fast, after as few as 10,000 writes.

Suppose we design a server system with a 128GB MLC FLASH-based solid state disk. Imagine that the disk controller has a perfect wear-leveling mechanism, so that the data blocks stored on the disk are reshuffled as they are written so that the number of writes to every FLASH cell remains balanced. The I/O subsystem has a maximum write bandwidth of 512MB/sec. If we write data to the disk continuously, how long do we expect the disk to last before any cell wears out. (1GB = 1024MB). [4 points]

\[
128\text{GB} / 512\text{MB/sec} \times 10,000\text{ writes} = 2,560,000 \text{ sec} = 29.6 \text{ days}
\]

b) In many high-performance storage deployments (e.g., large transaction processing databases), system architects deploy a very large number of disks, but intentionally use only a fraction of the storage capacity of each. Why might they do this? [6 points]

*Using only the outermost cylinders of disks increases bandwidth (more bits per revolution) and decreases seek times (shorter seek distances). Moreover, more parallel disks allows for greater aggregate bandwidth and more concurrent seeks.*
c) Consider a 10,000 RPM disk drive. Seek operations consist of a “head positioning” phase, which requires 0 to 3ms depending on how far the head must move (for simplicity, assume positioning time varies linearly with the seek distance) and 1ms in a “settling” phase where the head locks on to the correct track. Once the head has settled, the disk can transfer data continuously at 100MB/sec.

Suppose we access this disk with a random sequence of 1MB reads. Assume each read returns a contiguous 1MB block from a single track. Further assume that the amount of data stored in each track is constant (even though that is not true for real disks).

i. What is the expected per-read seek time? [3 points]

The expected seek distance between consecutive random reads is 1/3 of the diameter of the disk. Hence, seek time will be 1ms positioning + 1ms settling, or 2ms.

ii. What is the expected rotation delay? [3 points]

On average, 1/2 of a revolution is needed to bring the requested data under the disk head. Each rotation requires 6ms, so the expected rotation delay is 3ms.

iii. What is the expected transfer time? [3 points]

1MB takes 10ms.

iv. What is the overall bandwidth achieved by the random read sequence (in MB/s)? [3 points]

1MB / 15ms or 66.6MB/sec.