EECS 470 *Final Exam* SOLUTION KEY

Fall 2007

Name: _____ unique name: _____

Sign the honor code:

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

_____

Scores:

#	Points
1	**/15**
2	**/11**
3	**/15**
4	**/24**
5	**/15**
6	**/20**
Total	***/100***

NOTES:

- Closed book/notes
- 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. 15 minutes per problem).
- There are *?* pages including this one. Please ensure you have all pages.
- Many thanks to Prof. Babak Falsafi and Prof. James Hoe for contributing some of the exam problems
- **Be sure to show work and explain what you've done when asked to do so.**

1) **Memory [11 points]**

a) Draw a diagram of an SRAM cell. (You may use the logic symbol for an inverter where appropriate to simplify your diagram or draw individual transistors. Be sure to include and label the bit line(s) and word line(s) that connect to the SRAM cell.) **[5 points]**

Diagram

b) Draw a diagram of a DRAM cell. (Same notes as above.) **[5 points]**

Diagram

c) In a given technology generation (e.g., 45nm), which memory technology requires more area per bit of storage? **[1 point]**

SRAM

2) Page Table Designs. [15 points]

In class, we discussed three different designs for page tables: the SPARC top-down hierarchical design, the MIPS bottom-up hierarchical design, and the PowerPC inverted (a.k.a hashed) design. Let's examine the design and performance of each of these page table structures. Throughout this question, assume that virtual addresses are 32 bits, pages are 4KB (and, where relevant, the pages that make up the page table are 4KB as well), and that Translation Lookaside Buffer (TLB) misses are handled in hardware.

a) First, consider a 2-level top-down hierarchical page table.
   i) On a TLB miss, in the common case, how many memory accesses are required to load a page table entry into the TLB? **[1 point]**

   2 accesses, one for 1st level, one for 2nd level. Then, original memory access can proceed.

   ii) During a TLB miss, what happens if a page table entry cannot be found? **[3 points]**

   Page fault. Load page from disk (or, access is to an invalid virtual address)

   iii) Suppose a program touches exactly 256 words of memory. What is the largest amount of storage the page tables for these 256 words could require (i.e., given a worst-case data layout)? **[3 points]**

   1 1st level, 256 2nd level pages, total 257 pages, 1028KB, 1052672 bytes.

iv) What is the smallest amount of page table storage that could be required (i.e., in a best-case data layout)? **[3 points]**

2 page tables (1 1st, 1 2nd). 8KB.

b) Now, suppose the page table follows the MIPS bottom-up design. What is the most common number of memory accesses during a TLB miss in this design? **[1 point]**

Common case is a single access – usually, the translation for the last level of the hierarchy is present in the TLB, so it can be accessed without a nested miss.

c) Suppose the page table uses a hashed (inverted) design, like the PowerPC.
i) Now, how many memory accesses typically occur during a TLB miss? **[1 point]**

Again, just one access, in the case where the inverted lookup succeeds

ii) If no page table entry is found in the hashed page table under this design, what does this indicate? (Your answer should focus on how this case is different from the top-down page table). **[3 points]**

Unlike the hierarchical tables, the PowerPC's inverted table is not guaranteed to hold mappings for all pages present in memory – it is just a performance optimization. If the hashed lookup fails, a software handler must perform a slower lookup in a backup data structure before we can be sure the access causes a page fault.

3) Power [15 points]

Go Blue! Computers is trying to design a high-performance processor for a next generation video game console. Their baseline processor design executes at 1500 MIPS and at a nominal operating frequency of 2GHz and voltage of 1.1V. At this operating point, the chip requires 80W of power. (Throughout this problem, assume that operating frequency and voltage are linearly related, and performance is proportional to frequency).

a) The console manufacturer has a strict power budget that allows at most 60W for the processor. What is the performance (in MIPS) of the *Go Blue!* design if its voltage and frequency are scaled to meet this power budget? **[6 points]**

Decrease in Power = (80-60)/80 = 25%
1-.25 = 0.75 = x^3
x = 0.9086
Effective MIPS = 1500 * x = 1363 MIPS

b) The console manufacturer has set a performance target of 1400 MIPS. When voltage and frequency are scaled to precisely meet this target, the design still falls short of meeting the power budget. The designers propose to use clock gating to try to further reduce power consumption without impacting performance.

i) What is clock gating? **[3 points]**

Activate (or deactive) clock of block of logic when work needs to be done. Prevents switching of logic when nothing needs to be done.

ii) By what fraction must clock gating reduce switching activity to meet the power budget? **[6 points]**

Effective MIPS = 1400 MIPS
x = 1400/1500 = 0.9333333
x^3 = 1 - % decrease
% decrease = 18.7%
Power = 65W
Power ~ activity factor, and must reduce activity by 5W / 65W = 7.7%

4) Cache Design. [24 points]

a) Name and briefly describe an effective technique to remove bandwidth conflicts to a single block in a multi-banked cache. **[5 points]**

Line buffer – can service many hits to one block

b) *Go Blue! Computers* just released a high-end microprocessor with 48K 6-way set-associative L1 I and D caches. Assuming that the processor uses 4K pages, is it possible to do parallel address translation when accessing the cache (i.e., access TLB and cache at the same time)? Why or why not? **[5 points]**

No, the maximum size of a virtually indexed physically tagged cache is Page Size * Assoc = 4KB * 6 = 24KB

Also OK:
Yes, if the data cache checks the 2 locations were synonyms might exist in parallel on each lookup/allocation.

c) Either way, the design team decide to use a virtually-addressed I-cache, but a virtually-indexed physically-tagged D-cache. Explain why the architects chose different strategies for the instruction vs. data cache. **[5 points]**

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.

Also, must either flush I$ or include ASIDs in tags to distinguish homonyms.

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

	Miss Ratio	***Bandwidth***	***Tag Array Size***
(a) vs. (b)	(b) spatial locality	(a), block size is smaller, less useless data transferred	(b), larger block size means fewer sets and fewer tag bits
(b) vs. (c)	(b) spatial locality	(c), transfer size is smaller, less useless data transferred	(b), as (c) requires extra valid bits
(a) vs. (c)	(a) fewer conflict misses	same	(c), as (a) has more sets

5) Multicore and Multithreading [15 points]

For each of the scenarios described below, match the workload to the most appropriate processor design. You must **explain your choice to receive full credit.** Your choices are:

A. normal superscalar (single-threaded, single core)
B. vector machine
C. 2-way simultaneous multithreaded superscalar
D. 4-core chip multiprocessor (single-threaded cores)
E. 10-thread fine-grain multithreading

a) A scientific kernel that primarily performs matrix multiplication **[3 points]**

vector

b) A network packet processing application where many independent packets must be checked against virus signatures. The data structure for virus signatures is too large to fit in an L1 cache, but can fit in an L2. **[3 points]**

Fine-grain

c) Compiling a large C source file with gcc. **[3 points]**

superscalar

d) A parallel computational fluid dynamic solver that consists of long sequences of complex, dependant floating operations. The application's performance is bounded by floating point execution bandwidth. **[3 points]**

CMP

e) A multiprogrammed workload consisting of ray tracing software, which heavily utilizes the floating point unit, and a text parser, which contains no floating point code. **[3 points]**

2-way SMT

5) Spatial Pattern Prediction [20 points]

Applications written in C/C++ often have a large number of data structures with complex, but identical layouts. For example, consider the database maintained by Wolverine Access to track your name, address, major, uniqname, and GPA. Each record (corresponding to a struct in C) is laid out in the same way. Often, when traversing this data, an application only wants to access a subset of the fields in each structure, for example, a query that computes the average GPA of computer engineering students. Although there is some spatial locality to the data (i.e., your major and GPA are somewhere near each other), a cache block that is large enough to capture the entire record also transfers a great deal of unneeded data.

Sub-blocking can help eliminate the bandwidth cost of transferring the unneeded data. However, this bandwidth savings comes at the price of increased misses – each sub-block that is accessed incurs an extra miss. A *spatial pattern predictor* tries to predict the additional set of sub-blocks that will be needed in the future, and fetch all of those sub-blocks at the same time (when the first sub-block within a block is fetched). When the spatial pattern predictor makes correct predictions, we achieve the same bandwidth savings as a sub-blocked cache, but have the miss rate of a non-sub-blocked cache!

a) Draw a diagram of a 32KB 2-way sub-blocked cache with 64-byte blocks and 8-byte sub-blocks (assume addresses are 32 bits). Be sure to show the tag array, data array, valid bits, tag comparison logic and indexing signals. Label the width of each storage array and signal. Unlabeled signals are assumed to be 1 bit wide. **[6 points]**

diagram

b) Propose a design for a spatial pattern predictor. Draw your design. Explain why your design will achieve high prediction accuracy (i.e., it won't predict fewer or more sub-blocks than actually needed), fast (predictor) lookup time, and practical size. **[6 points]**

Best design is a PC-based lookup table. But, any reasonable idea will do.

c) Suppose the non-sub-blocked cache achieves a miss ratio of 5%, while a sub-blocked cache (without your predictor) has a miss ratio of 15%.
   i) Given these ratios, on average, how many sub-blocks are accessed per block? **[2 points]**

3 sub-blocks

   ii) Suppose we add your predictor to the sub-blocked cache. The predictor will be activated each time there is a cache miss that allocates a new block (i.e., it won't be used if there are additional misses to sub-blocks for a block that is already present). Assume a single level hierarchy where hit time is 1 cycle and miss time is 20 cycles for both cache designs. *Predictor coverage* is defined as:

(# of sub-blocks predicted correctly) / (# of sub-blocks actually accessed - 1)

Note that you don't count the first sub-block requested by the processor as correctly predicted; that first sub-block is the reason for the -1 in the denominator. So, for example: the processor will access a total of 4 sub-blocks, the predictor makes its prediction when the first is accessed and it correctly predicts two of the three remaining sub-blocks, coverage is 66%.

What coverage must your predictor achieve to get an average effective access time of 2.4 clock cycles? **[6 points]**

2.4 clock cycles means 7% miss rate. This is 20% of the way from the non-sub-blocked miss rate to the sub-blocked miss rate. Hence, predictor must have 80% coverage.