

Comp 411 Computer Organization

Fall 2007

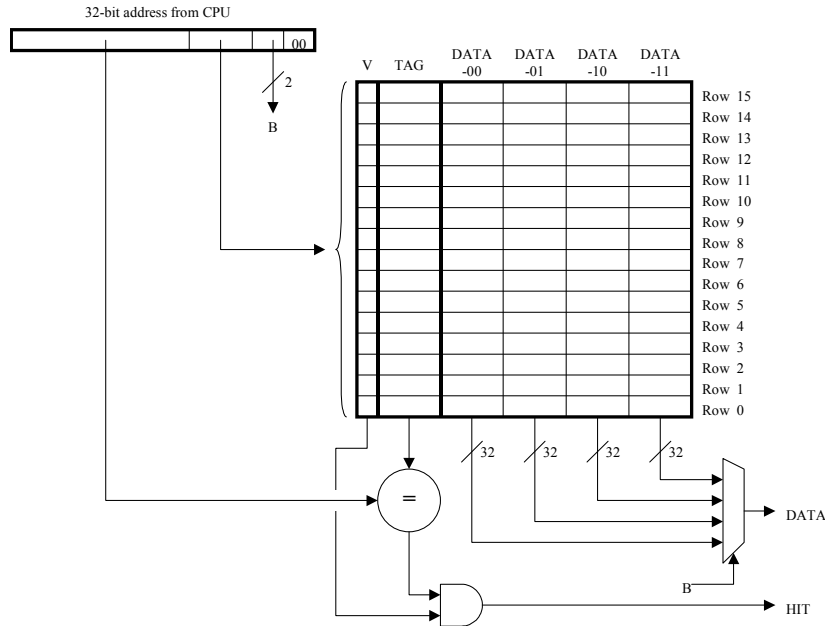
Problem Set #10

Issued Monday, 11/19/07; Due Monday, 12/03/07

Homework Information: Some of the problems are probably too long to be done the night before the due date, so plan accordingly. Late homework will not be accepted. Feel free to get help from others, but the work you hand in should be your own.

Problem 1. Cache accounting

The diagram below illustrates a blocked, direct-mapped cache for a computer that uses 32-bit data words and 32-bit byte addresses. The block size is 4 words.



- (A) What are the maximum number words of data from main memory that can be stored in the cache at any one time? How many bits of the address are used to select which line of the cache is accessed? How many bits wide is the tag field?
- (B) Briefly explain the purpose of the one-bit V field associated with each cache line.
- (C) Assume that memory location 0x3328C was present in the cache. Using the row and column labels from the figure, in what cache location(s) could we find the data from that memory location? What would the value(s) of the tag field(s) have to be for the cache row(s) in which the data appears?
- (D) Can data from locations 0x12368 and 0x322F68 be present in the cache at the same time? What about data from locations 0x2536038 and 0x10F4? Explain.
- (E) When an access causes a cache miss, how many words need to be fetched from memory to fill the appropriate cache location(s) and satisfy the request?

Problem 2. Where are the Bits?

AMD's Sempron processor is advertised as having a total on-chip cache size of 384 Kbytes. It is broken down into 128Kbytes of Level 1 (L1) cache and 256Kbytes of Level 2 (L2) cache. The first level caches are composed of separate instruction and data caches, each with 64 Kbytes. All caches use a common block size of 64 bytes (16, 32-bit words). The L1 instruction and data caches are both 2-way set associative, and the unified (used for both instructions and data) L2 cache is 16-way associative. Both L1 caches employ an LRU replacement strategy, while the L2 cache uses RANDOM replacement. The L1 data cache uses one dirty bit for each group of 4 words in a line (4 for each line). None of the caches, employ a valid bit, instead a fault handler is responsible for executing reads from a reserved region of memory to initialize the cache's contents.

- (A) Determine which bits of a 32-bit address are used as tags, cache indices, and block selection for each of the three caches.
- (B) Compute how many actual bits are needed each cache, accounting for tags, data, replacement, dirty, and any other overhead.

Both L1 caches have a 3-cycle access latency, the L2 cache requires 5 additional cycles (after an L1 miss), and main memory requires 20-cycles to load a line. A popular benchmark suite reports that this cache design achieves an instruction cache hit rate of 95%, and a data cache hit rate of 80%. A hit rate of 98% has been reported for the L2 cache for the same benchmarks.

- (C) What is the average effective access time for instruction fetches in this cache architecture (in cycles)? What is the average effective access time for data fetches (loads and stores) in this cache architecture (in cycles)? If you assume that about 20% of executed instructions access memory, what is the overall effective access time?
- (D) AMD also sells a version of the chip with a L2 cache that is half the size (128 Kbytes). In that chip the L2 hit rate is only 92% for the same benchmark, what is its overall effective access time in cycles?
- (E) Assuming that your on-chip memory budget is fixed at 384 Kbytes, what might you suggest to improve the performance of this cache architecture?

Problem 3. Trip Around the Old Block

Lee Hart is assigned the task of designing the cache architecture for an upcoming processor. He is given the following design constraints. Cache accesses, for both hits and misses, require 1 clock cycle. On cache misses, an entire cache block (multiple words) is fetched from main memory. Main memory accesses are not initiated until after the cache is inspected. A main-memory access consists of 2 cycles to send the address, 3 cycles of idle time for main-memory access, and then 1 cycle to transfer word to the cache. Moreover, the processor will stall until the last word of the block has arrived. Thus, if the block size is B words (32 bits), a cache miss will cost $1 + 2 + 3 + B$ cycles. The following table gives the average cache miss rates of a 1 Mbyte cache for various block sizes:

<i>Block size (B)</i>	<i>Miss ratio (m), %</i>
1	3.2
4	1.6
16	0.8
64	0.4
256	0.3

- (A) Write an expression for the average memory access time for a 1-Mbyte cache and a B-word block size (in terms of the miss ratio m and B).
- (B) What block size yields the best average memory access time?
- (C) If main memory bus width is expanded to 128 bits (4 words), reducing the number of cycles for accessing the block proportionately, what is the optimal block size?
- (D) Suppose that expanding the memory bus to 128 bits increases the main memory access latency by 6 clocks (for a total of 12 cycles of overhead before the first access); what then is the optimal block size?

Problem 4. Approximate LRU Replacement

Immediately after the cache-replacement-algorithm lecture in Comp 411, Lori Acan has an epiphany. It dawns on her that it is possible to approximate LRU replacement in a 4-way set with only 3 bits per set, 2 less than an exact LRU implementation and only 1 bit more than the FIFO algorithm. Her scheme works as follows. She partitions the 4-way set into two halves and treats each half as a 2-way set, thus, allotting one replacement bit for computing the LRU line in each half. This requires 2 bits, one for each half. The third bit is used to keep track of the most recently used half of the cache. On a cache miss, the least recently used line in the least recently used half of the cache is replaced.

- (A) Simulate the behavior of this replacement algorithm on a 4-word fully associate cache using the instruction sequence given in lecture and repeated below where $\$t3 = 0x4000$. Assume the initial values of the replacement bits are 0-00. Estimate the miss rate for this code sequence in the steady state while noting of the values of the 3 replacement bits on each cache access.

Address	Instruction
400:	lw $\$t0, 0(\$t3)$
404:	addi $\$t3, \$t3, 4$
408:	bne $\$t0, \$0, 40$

- (B) Explain how Lori’s approximate LRU algorithm can be used for any N-way set where $N=2^k$. How many bits are required per set in these cases?
- (C) Give a memory reference string for which Lori’s 4-way LRU replacement algorithm differs from an exact LRU replacement.
- (D) In the worse case, what will be the highest element in an ordered list of recent accesses that Lori’s approximate LRU replacement algorithm would throw out?