

CS 6290: High-Performance Computer Architecture

Fall 2007

Homework 4

Due: December 4th before class

Show your work. No points will be given for providing only a numeric answer without showing or explaining (briefly, please) how you got the answer.

Problem 1: Caches

The system has the following properties:

- The processor uses 32-bit physical addresses.
 - Load/store instructions are performed in program order and each ties up all caches until it is complete.
 - There is 2KB of L1 cache and 16KB of L2 cache.
 - Both caches are physically indexed and tagged with a block size of 128 bytes
 - Both caches are 4-way set associative and use the LRU replacement policy.
 - The write policy in both caches is write-allocate, write-back.
 - Dirty bits are used to allow silent replacement of clean blocks, so only dirty blocks are written back on replacement.
 - A dirty block replaced from the L1 cache is sent to the L2 cache. If that block is still cached in the L2 cache, the L2 updates its version of the block, but does not update the block's LRU counter in L2, and does not send the block to memory. If the block is not in L2, it is sent to main memory without putting it in L2.
 - When there is an access that misses in both caches, the L2 cache is completely updated (including replacement, if any is needed) before forwarding the block to L1.
 - Hits in the L1 cache do not go to the L2 cache and do not update the LRU in the L2 cache.
 - Neither inclusion nor exclusion property is maintained.
- a) **[5 points]** For each cache, show how physical address FFF82754 is broken down into the offset, index and tag, showing the number of bits for each. Which set will this load go to in the L1 cache, and which in the L2 cache? How many tags will be checked to determine whether this address is a hit or a miss in the L1 cache?
- b) **[5 points]** Cache size tells us the number of data bytes that the cache can hold. However, the cache also has some overhead state (tags, valid, LRU, and dirty bits). Compute the number of overhead bits in the L1 cache in this problem. What is the ratio of the number of overhead bits and the number of data bits in the cache? Keeping all other specified parameters the same (cache size, associativity, etc.), what would this ratio be if the cache block size was changed to 32?

For parts (c) and (d), the processor is executing the following sequence of memory access instructions (only the type of instruction and the physical address referenced are shown):

```
LD FFF82754
ST A0034300
LD 80004F14
LD 08034700
LD 0000171C
ST FFF82754
LD 80004F18
LD 00001720
ST 7FFFFFF08
ST 08034700
```

- c) **[10 points]** The processor's caches are completely empty when it begins executing the above sequence of instructions. Show the state of both caches after the above sequence of accesses is completed. For each cache line show its valid and dirty bits, the LRU counter, as well as the tag. Using a table like the one shown below, also show for each LD/ST in the sequence which cache(s) are looked up and whether the lookup is a hit or a miss. For cache misses, also indicate which line (if any) is replaced from the cache to make room for it, and whether the replaced line was dirty.

Instruction	L1 Cache				L2 Cache			
	LU	H/M	Replaced	DR	LU	H/M	Replaced	DR
LD FFF82754								
ST A0034300								
LD 80004F14								
LD 08034700								
LD 0000171C								
ST FFF82754								
LD 80004F18								
LD 00001720								
ST 7FFFFFF08								
ST 08034700								

Explanation of table columns: LU – cache looked up; H/M – if looked up, hit or miss; Replaced – if miss, tag of replaced block or N/A if line was empty; DR (dirty replaced) – if a block was replaced, was it dirty?

- d) **[10 points]** Another system has the same parameters as the one described above, except that the L1 cache is a direct-mapped, write-allocate, write-through cache. Repeat part (c) for this system. When showing the state of the cache, show only fields that are needed in this modified system.