

# CS 6290: High-Performance Computer Architecture

*Fall 2007*

## Homework 4

**Due: December 4<sup>th</sup> 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?

In both caches, the block has 128 bytes, so the lowermost 7 bits of the address are the offset.

In the L1 cache, there are 16 lines (2048/128), so there are 4 sets (16/4) => index has 2 bits. The 23 bits that remain (32-7-2) are the tag.

In the L2 cache, there are 128 lines (16K/128), so there are 32 sets (128/4) => index is 5 bits. The 20 bits that remain are the tag.

Address FFF82754 is 1111 1111 1111 1000 0010 0111 0101 0100 in binary.

In the L1 cache, we have Tag: 1111111111110000010011, Index: 10, and Offset: 1010100, and the load goes to set 2 (offset is 10). Since the cache is 4-way SA, 4 tags are checked on each access.

In the L2 cache, we have Tag: 1111111111110000010, Index: 01110, and Offset: 1010100, and the load goes to set 14 (01110 is 14). This cache is also 4-way SA, so 4 tags are checked.

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 each line of data, we keep a valid bit, a 23-bit tag (see part a), a dirty bit, and 2 bits of LRU. In total, we have 27 (1+23+1+2) overhead bits for 1024 bits (128 bytes \* 8) of data, so the ratio is  $27/1024=0.026$  (2.6%).

In the data block were 32 bytes in size, there would be only 5 index bits in the address, and there would be 64 lines in the cache, so there would be 16 (64/4) sets, so there would be 4 index bits in the address. The number of tag bits would be  $32-5-4=23$  (same as with 128-byte blocks). Now we have 27 overhead bits per 256 bits (32 bytes) of data, so the ratio is  $27/256=0.105$  (10.5%).

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	Y	M	None		Y	M	None	
ST A0034300	Y	M	None		Y	M	None	
LD 80004F14	Y	M	None		Y	M	None	
LD 08034700	Y	M	None		Y	M	None	
LD 0000171C	Y	M	FFF82700	N	Y	M	None	
ST FFF82754	Y	M	A0034300	Y	Y	H		
LD 80004F18	Y	H			N			
LD 00001720	Y	H			N			
ST 7FFFFFF08	Y	M	08034700	N	Y	M	None	
ST 08034700	Y	M	FFF82700	Y	Y	H		

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 is line was empty; DR (dirty replaced) – if a block was replaced, was it dirty?

For LD FFF82754, the L1 set is 10 (2), and it's a miss. In L2 the set is 01110 (14), miss.

For ST A0034300, the L1 set is 10 (2), and it's a miss. In L2, the set is 00110 (6), miss. After the block is fetched from L2 and inserted into L1, the write occurs in the L1 cache, so its dirty bit is set. Note that the dirty bit is not set for this block in L2 (data in L2 is still unmodified).

For LD 80004F14, the L1 set is 10 (2), and it's a miss. In L2, the set is 11110 (30), miss.

For LD 08034700, the L1 set is 10 (2), and it's a miss. In L2, the set is 01110 (14), miss.

For LD 0000171C, the L1 set is 10 (2), and it's a miss. In L2 cache, the set is 01110 (14). Note that the set in L1 is full, so we replace the least recently used block from that set, which is block at address FFF82700. The only access to the replaced block was a LD, so we don't need a write-back to the L2 cache.

For ST FFF82754, the L1 set is 10 (2), and it's a miss (we just replaced that block). Note that the block is still in L2 cache, so it's a L2 hit. Because set 2 in L1 is full, we replace the LRU block, which is block that begins at A0034300. The replaced block is dirty, so it's written back to L2. Finally, this access is a store, so block FFF82700's dirty bit in L1 is set.

For LD 80004F18, block 80004F00 is still in the L1 cache, so it's a hit (no L2 access).

For LD 00001720, block 00001700 is still in the L1 cache, so it's a hit (no L2 lookup).

For ST 7FFFFFF08, the L1 set is 10 (2), and it's a miss. We replace the LRU block in this set, which is block that begins at 08034700 (note that block 80004F00 has been in the cache longer, but it was accessed more

recently). The replaced block is clean, so there is no write-back. In L2, the set is 11110 (30), and we have a miss. There is only one other block in set 30, so there is no replacement from L2.

For ST 08034700, the L1 set is 10 (2), it's a miss (we just replaced that block). The access in L2 is a hit. In L1, we replace the LRU block, which is the block that begins at FFF82700. This is a dirty block so we write it back to L2 cache.

At the end of this sequence, we have in L1 all sets empty (zero valid bits) except set 2. In set 2, we have 4 blocks. Assuming that the LRU counter for a block is 0 for the LRU block and 3 for the MRU block, we have (tag is in binary)

V	Tag	D	LRU	Note
1	00001000000000110100011	1	3	Block at 08034700
1	01111111111111111111111	1	2	Block at 7FFFFFF0
1	000000000000000000001011	0	1	Block at 00001720
1	1000000000000000000100111	0	0	Block at 80004F00

For the L2 cache, we show only non-empty sets:

Set 6:

V	Tag	D	LRU	Note
1	10100000000000110100	1	3	Block at A0034300
0				
0				
0				

Note: the dirty bit for this block in L2 was set when it was written back from L1, not when we did the original ST to L1.

Set 14:

V	Tag	D	LRU	Note
1	00001000000000110100	1	1	Block at FFF82754
1	01111111111111111111111	0	3	Block at 08034700
1	0000000000000000000001	0	2	Block at 0000171C
0				

Note: the dirty bit for block 08034700 is not set in L2 because we didn't write it back from L1 yet.

Set 30:

V	Tag	D	LRU	Note
1	1000000000000000000100	0	2	Block at 80004F14
1	01111111111111111111111	0	3	Block at 7FFFFFF08
0				
0				

Note: the dirty bit for block 7FFFFFF08 is not set in L2 because we didn't write it back from L1 yet.

**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.

In L1 cache, now there is no D bit and there are no LRU bits. Also, the index is now 4 bits (16 blocks) and the tag is only 21 bits.

For LD FFF82754, the L1 index is 1110 (14), and it's a miss. In L2 the set is 01110 (14), miss.

For ST A0034300, the L1 index is 0110 (6), and it's a miss. In L2, the set is 00110 (6), miss. After the block is fetched from L2 and inserted into L1, the write occurs in the L1 cache. It is a write-through cache, so the write goes to L2 where the dirty bit is set.

For LD 80004F14, the L1 index is 1110 (14), and it's a miss. We replace block FFF82754 which was occupying that slot 14 in the cache. In L2, the set is 11110 (30), miss.

For LD 08034700, the L1 index is 1110 (14), and it's a miss. We replace block 80004F00. In L2, the set is 01110 (14), miss.

For LD 0000171C, the L1 index is 1110 (14), and it's a miss. We replace block 08034700. In L2 cache, the set is 01110 (14).

For ST FFF82754, the L1 index is 1110 (14), and it's a miss. We replace block 00001700. Note that the block is still in L2 cache, so it's a L2 hit. This access is a store, so write-through to L2 set the dirty bit there.

For LD 80004F18, the L1 index is 1110 (14), and it's a miss. We replace block FFF82700. We have a L2 hit.

For LD 00001720, the L1 index is 1110 (14), and it's a miss. Block 80004F00 is replaced. Again, it's a L2 hit.

For ST 7FFFFFF08, the L1 index is 1110 (14), and it's a miss. We replace block 00001700. In L2, the set is 11110 (30), and we have a miss. Write-through to L2 sets the dirty bit.

For ST 08034700, the L1 index is 1110 (14), it's a miss, and we replace block 7FFFFFF00. We have a hit in L2. Write-through to L2 sets the dirty bit there.

Note that there are no write-backs from L1 on replacement – the L2 is always up-to-date due to write-through, so there is no need for write-backs.

At the end of the sequence, we have all blocks in L1 empty (V bit is 0), except for entry 6 (which has block A0034300 with tag 10100000000001101000) and entry 14 (which has block 08034700 with tag 00001000000000110).

In L2 the state is not the same as in ©. We have the same block in the same set, but dirty bits are different because all ST instructions directly updated dirty bits in L2. Also, LRU is different, because many L1 hits in (c) became misses in (d). These misses result in LRU updates in L2 that didn't happen in part (c).