Name _____

CE-1921-11 – Dr. Durant – Quiz 9
Spring 2016, Week 10

---

1. (1 point) ***List*** the two quantities that we are attempting to simultaneously optimize by adding levels in a cache hierarchy (hint: consider why a large main memory is not made of only SRAM).
2. (1 point) ***Define spatial*** locality and ***explain*** how a cache might exploit it.
3. (1 point) For a cache in a 24-bit address memory space with 1024 B of cache space, what ***additional piece of information*** do you need in order to determine how many blocks there are?
4. (1 point) You are additionally told that the above cache has ***16 blocks*** and is a ***4-way*** cache. ***Calculate*** how many ***sets*** are in the cache.
5. (1 point) Continuing the example above, ***calculate*** the tag size for each block in the cache.
6. (1 point) ***Explain*** the use of a valid bit in a cache.
7. (2 points) ***Explain*** how ***early restart*** reduces the penalty for a cache miss.
8. (1 point) ***Describe*** the replacement strategy for a ***direct***-mapped cache. (Hint: Some people would call this a trick question; the answer is very simple.)
9. (1 point) When is a write-back cache written down to the next level in the memory hierarchy (recall that other methods of handling cache writes are the write-through cache and using a write buffer).

Answers

1. Speed and cost (or physical size)
2. Spatial locality means that nearby memory locations are likely to be accessed around the same time. This is often true for code (instructions tend to be one after the other in memory) and for data (e.g., elements of an array that are near each other are often needed by an algorithm). Caches exploit this by loading nearby elements before they're requested; this can be accomplished by increasing the blocksize.
3. The size of each block.
4. 16 blocks / 4 ways (blocks/set) = ***4*** sets
5. Blocksize = 1024 B / 16 blocks = 64 B. Offset = log2(blocksize) = 6 bits. Set is encoded with log2(4) = 2 bits. Tag bits = address bits – set bits – offset bits = 24 – 2 – 6 = ***16 bits***.
6. The valid bit indicates whether the corresponding block in the cache currently has useful data. It is initially 0. In basic usage, it wouldn't be cleared except perhaps while a new read was pending. In advanced usage, it might be cleared to indicate another source has modified the underlying memory, invalidating what is in the cache.
7. While reading a potentially large block, the cache will return the requested word as soon as it is completely read from the underlying memory instead of waiting for the entire block to be read.
8. There is only 1 block per set, so it is the victim if another tag in the same set is requested.
9. Only when the block is about to be evicted, likely because it hasn't been used for some time and another block is about to be read into its space.

Name _____

CE-1921-12 – Dr. Durant – Quiz 9
Spring 2016, Week 10

1. (1 point) ***List*** the two quantities that we are attempting to simultaneously optimize by adding levels in a cache hierarchy (hint: consider why a large main memory is not made of only SRAM).
2. (1 point) ***Define temporal*** locality and ***explain*** how a cache might exploit it.
3. (1 point) For a cache in a 32-bit address memory space with 4096 B of cache space, what ***additional piece of information*** do you need in order to determine how many blocks there are?
4. (1 point) You are additionally told that the above cache has ***16 blocks*** and is a ***8-way*** cache. ***Calculate*** how many ***sets*** are in the cache.
5. (1 point) Continuing the example above, ***calculate*** the tag size for each block in the cache.
6. (1 point) ***Explain*** the use of a dirty bit in a cache.
7. (2 points) ***Explain*** how ***requested word first*** reduces the penalty for a cache miss.
8. (1 point) ***Describe*** the least-recently used replacement strategy for an n-way set associative cache.
9. (1 point) When is a write-through cache written down to the next level in the memory hierarchy (recall that other methods of handling cache writes are the write-back cache and using a write buffer)?

Answers

1. Speed and cost (or physical size)
2. Temporal locality means that if a memory location has been accessed recently, it is more likely than a random memory location to be accessed again soon. Caches exploit this by keeping recently used items in the cache.
3. The size of each block.
4. 16 blocks / 8 ways (blocks/set) = ***2*** sets
5. Blocksize = 4096 B / 16 blocks = 256 B. Offset = log2(blocksize) = 8 bits. Set is encoded with log2(2) = 1 bit. Tag bits = address bits – set bits – offset bits = 32 – 1 – 8 = ***23 bits***.
6. The dirty bit indicates that data has been written by the processor (or other immediately higher level) but has not yet been written down to the memory (or other immediately lower level). When the write down happens depends on the write strategy (question 9).
7. Instead of reading the block the beginning, the cache requests the exact word requested by the processor first, letting it quickly return the result to the processor. The cache then continues to load the rest of the block while the processor is doing other work.
8. When all the ways in the set of a requested block are full, the cache replaces the one that hasn't been used (read or written) for the longest amount of time. This applies temporal locality since the other, more recently used, blocks are more likely to be used again soon than the evicted block.
9. Immediately. The good news is that only the written words need to be written down. The bad news is that the cache performance will be degrade if one block is written to repeatedly (since many write downs happen, keeping the cache busy).