Notes for 10/23/09

Another example... 4KB, 2-way set associative, 512-byte blocks
Thus, we have 8 blocks, divided into 4 sets.
Tag bits | set bits | offset bits = 21 | 2 | 9.

For writes, then there are two different ways we can maintain cache coherency.
Write-through—when a write occurs, write to the cache and then through to the DRAM (or, commonly, an intermediate write buffer).
Write-back—when a write occurs to a cache line, set a dirty bit, a bit that is initially zero when the cache line is loaded from DRAM. Then, for all cache lines, when they are evicted, if they have the dirty bit set, then write it back to DRAM. So, if a dirty bit is set, then there is an additional penalty on eviction.

For our simulator timing:

Hit	Add hit time
Miss, dirty bit unset	Add hit time + miss penalty
Miss, dirty bit set	Add hit time + miss penalty + store penalty

Victim cache—instead of increasing associativity of the main cache, create a cache where evicted blocks can be stored; this cache is usually fully associative.
Multi-level cache—have hierarchy of caches that trade off speed with size/cost.
Non-blocking cache—have a load/store queue that does not require the processor to wait on the cache for every read and write.

Split cache—e.g., I-Cache vs. D-Cache
Shared cache—e.g., different cores sharing L2 cache