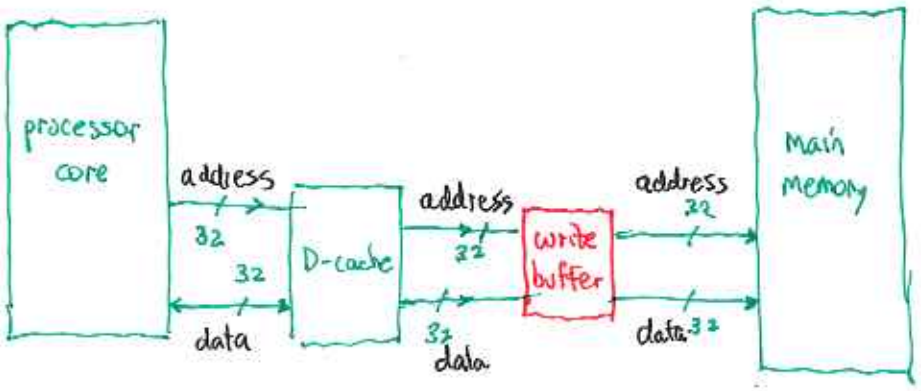
Monday, March 25, 2019

Cache write policies (continued)

Write buffers

For either policy of writing to memory in a system with caches, a write buffer is a critical component.



- simplified view of data path for memory writes.

Writes to main memory are very time-consuming, so data blocks being written back are queued in the write buffer.

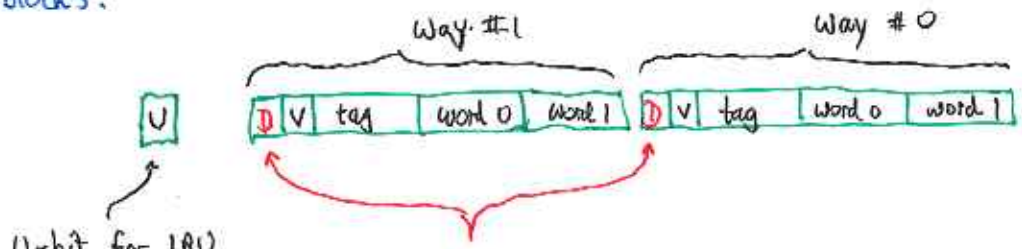
- queuing a write request happens fast (one clock cycle)
- buffer takes its time to send queued data to memory (10s - 100s of cycles)
- if buffer is full, processor stalls to let it catch up.

Writeback

The D-cache works this way:

- HIT: data is kept in the cache - no main memory write.
- MISS: (reading or writing): data block must be evicted.
  - if data block being evicted is newer than the copy in main memory, save it to memory first!

Consider one set of a 2-way set-associative cache with two-word blocks.



U-bit for LRU replacement strategy

Dirty bits
- indicates that data in the cache block is inconsistent with main memory, and will to be saved.

On a cache miss, to this set.
- The LRU tells which way must be evicted
- If the way being evicted is "dirty", write the data block to main memory first.

Write-through caches

The same write buffer is used

→ all stores to the cache are queued to the write buffer (even on cache hits)

A dirty bit is not used.

On a write hit
- Save data in cache block
- queue a write to memory through write buffer

On a cache miss
- replace the cache data block without saving the old
- retry the read or write.

Write-through vs. writeback
- write-through less complicated
- writeback tends to make less work for cache/memory interface; less traffic on bus to main memory.

Multi-level caches

- Common on modern processors
- reduces big performance penalty on cache misses.

L1-caches:
- small, very fast (32 KB)
- processor can run at high clock rates

L2-caches:
- larger and slower than L1, but still much faster than DRAM (512 KB)
- handles L1 cache misses.

L3-caches:
- larger still (6 MB)
- handles L2 cache misses.

Introduction to virtual memory (VM)

We consider a simplified view of address-space management (one processor core, Linux-like OS).