ECE 571 – Advanced Microprocessor-Based Design Lecture 15

Vince Weaver

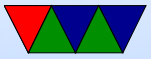
https://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

7 October 2024

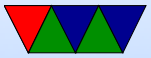
Announcements

- Don't forget HW#5 (caches)
- Useful reading: "When prefetching works, when it doesn't and why" paper by Lee et al.



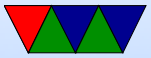
Cache Coherency

- Only gets complicated when you start writing.
- Need some way to know if other core's caches have the address you are accessing
  - Snoopy – "snoop" the bus
  - Directory – have central logic (doesn't scale as well)
- Cache coherency protocols
  - MESI
  - MOESI



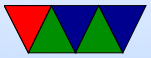
MESI

- State machine: modified, exclusive, shared, invalid
- Invalid – starts out here
- Shared – only has been read, same as memory, can be in multiple caches
- Exclusive – a core is requesting to write, so it gets exclusive access, invalidates all other copies
- Modified – dirty, has been written to. Write back and then can change to S

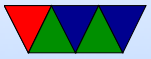


Other Concerns

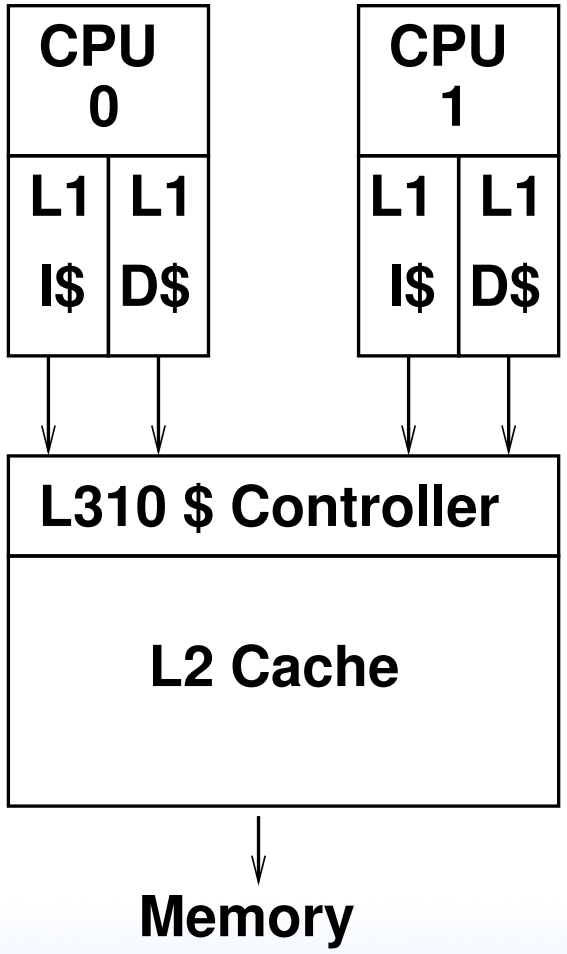
- What about load/store values in the pipeline that haven't been retired yet?
- What about Memory Model? Can loads pass stores? x86 has strict model, others no so much
  Apple M2 has special "act like x86 mode" to make emulation easier
- Special assembly instructions to flush cache, memory barriers. etc
- Locking in multi-threaded apps

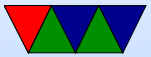


Example Real-World Cache layouts



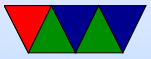
Cortex A9 Cache Layout





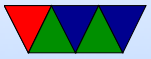
Cortex A9 Cache Layout

- OMAP4430 processor
- 32kB 4-way associative, separate L1-I and L1-D
  pseudo-round-robin or pseudo random replacement
  8-word line size (32B)
  critical-word first filling
  instruction: VIPT, data: PIPT
- Optional L2 cache controller
  Pandaboard has L310 L2 cache controller, 1MB 16-way



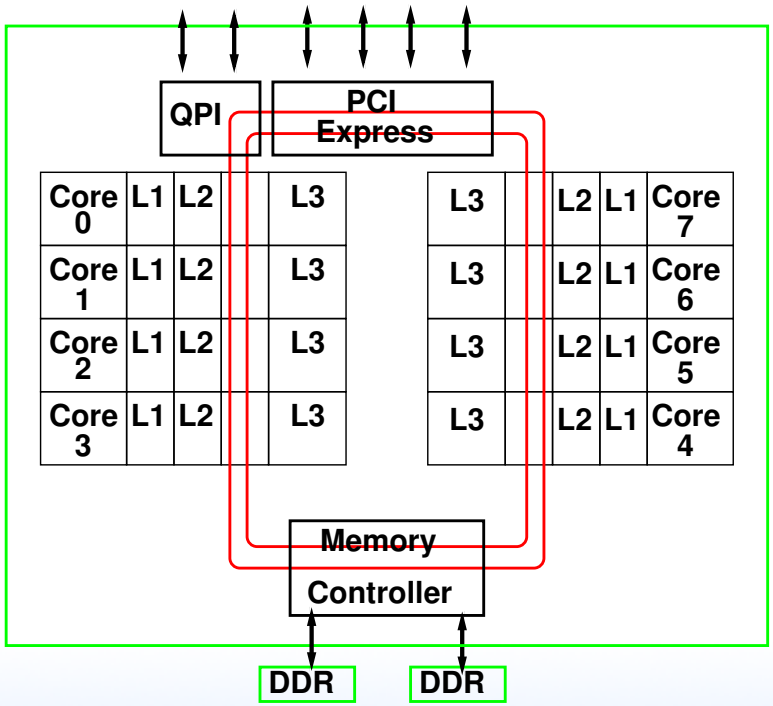
Optional prefetcher

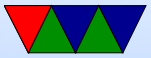
- data cache reads/writes non-blocking, 4 outstanding misses
  write buffer: 4 64-bit, allowing write combining



Haswell-EP Cache Layout

Note: see "Cache Coherence Protocol and Memory Performance of the Intel Haswell-EP Architecture" by Molka, Hackenberg, Schöne, Nagel.



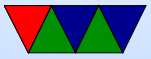


Haswell-EP Cache Parameters

- per core 32kB L1 I/D – 4 clocks
  64B/line, 8-Way
  (shared if hyper-threaded)
  writeback

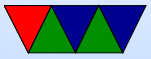
- mOp cache? 1.5K instructions, 8-way, 6Mop/line
  Loop stream detector, can execute w/o accessing icache

- per core 256kB L2 unified – 11 clocks

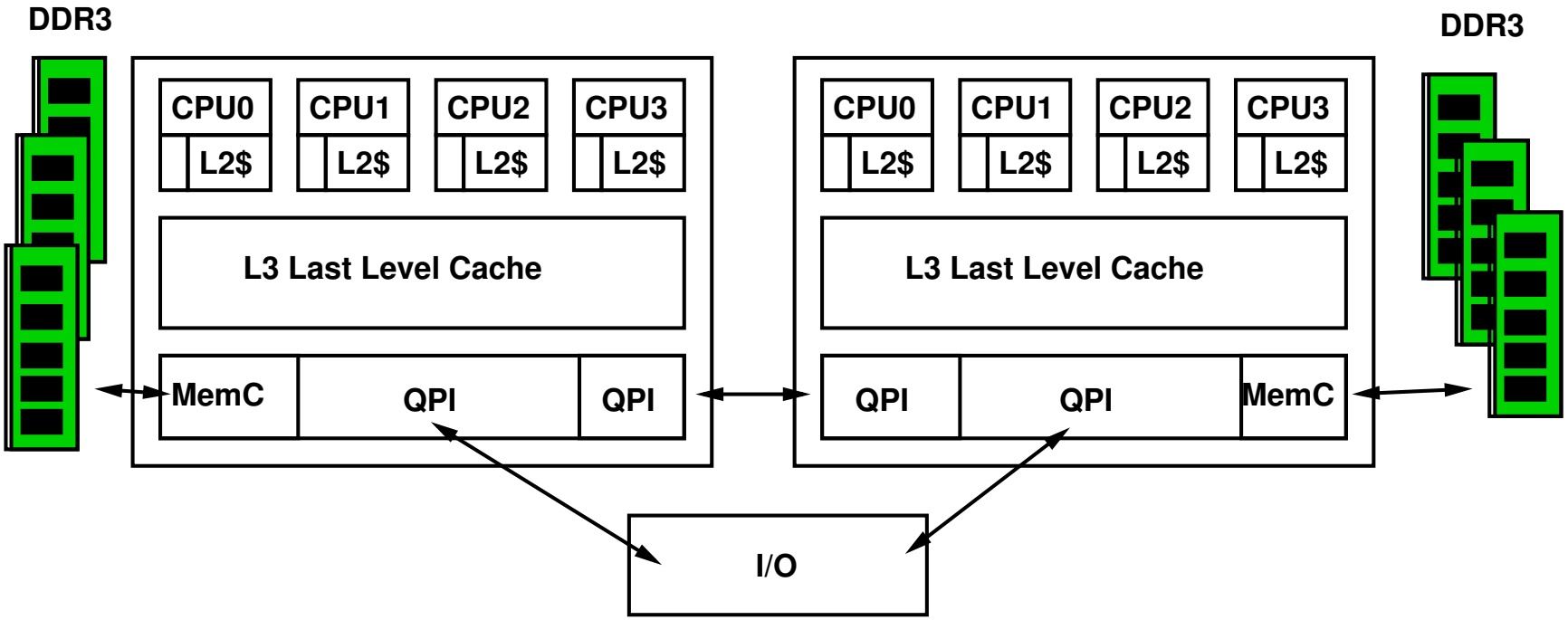


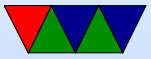
64B/line, 8-way
writeback. non-inclusive

- shared L3 20MB, writeback
- various hw prefetchers operating



SandyBridge Cache Layout



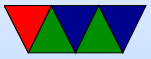


SandyBridge Cache Parameters

- per core 32kB L1 I/D – 4 clocks
  64B/line, 8-Way
  (shared if hyper-threaded)
  writeback

- mOp cache? 1.5K instructions, 8-way, 6Mop/line
  Loop stream detector, can execute w/o accessing icache

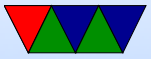
- per core 256kB L2 unified – 12 clocks



64B/line, 8-way
writeback. non-inclusive

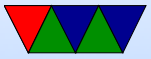
- shared L3 1MB-20MB – 26-31 clocks
  64B/line. 12-way (varies)
  writeback, inclusive

- various hw prefetchers operating



Prefetching

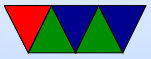
- Cold misses can be common.
- Try to avoid cache misses by bringing values into the cache before they are needed.
- Caches with large blocksize already bring in extra data in advance, but can we do more?



Prefetching Concerns

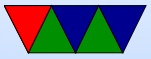
- When?
  We want to bring in data before we need it, but not too early or it wastes space in the cache.

- Where? What part of cache? Dedicated buffer?



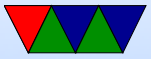
Limits of Prefetching

- May kick data out of cache that is useful
- Costs energy, especially if we do not use the data



Implementation Issues

- Which cache level to bring into? (register, L1, L2)
- Faulting, what happens if invalid address
- Non-cacheable areas (MTRR, PAT).
  Bad to prefetch mem-mapped registers!



Software Prefetching

- ARM has PLD instruction
- PREFETCHW for write (3dnow, Alpha) cache protocol
- Prefetch, evict next (make it LRU) Alpha
- Prefetch a stream (Altivec)
- Prefetch0, 1, 2 to all cache levels (x86 SSE)
  Prefecthnta, non-temporal