CSE 490/590 Computer Architecture

Cache III

Steve Ko
Computer Sciences and Engineering
University at Buffalo

Last time…

- Basic cache architecture
  - Placement policy
  - Replacement policy
- Average memory access time =
  hit time + miss rate * miss penalty
- To improve performance, reduce:
  - hit time
  - miss rate
  - and/or miss penalty
- Primary cache parameters:
  - Total cache capacity
  - Cache line size
  - Associativity

Causes for Cache Misses

- *Compulsory:* first-reference to a block *a.k.a.* cold start misses
  - misses that would occur even with infinite cache
- *Capacity:* cache is too small to hold all data needed by the program
  - misses that would occur even under perfect replacement policy & full associativity
- *Conflict:* misses that occur because of collisions due to block-placement strategy
  - misses that would not occur with full associativity

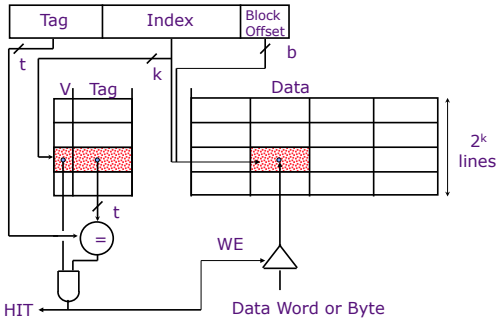
Effect of Cache Parameters on Performance

- Larger cache size
  + reduces capacity and conflict misses
  - hit time will increase
- Higher associativity
  + reduces conflict misses
  - may increase hit time
- Larger block size
  + reduces compulsory and capacity (reload) misses
  - increases conflict misses and miss penalty

Write Policy Choices

- Cache hit:
  - ***write through:*** write both cache & memory
    - » Generally higher traffic but simpler pipeline & cache design
  - ***write back:*** write cache only, memory is written only when the entry is evicted
    - » A dirty bit per block further reduces write-back traffic
    - » Must handle 0, 1, or 2 accesses to memory for each load/store
- Cache miss:
  - ***no write allocate:*** only write to main memory
  - ***write allocate*** *(aka fetch on write):* fetch into cache
- Common combinations:
  - write through and no write allocate
  - write back with write allocate

Write Performance



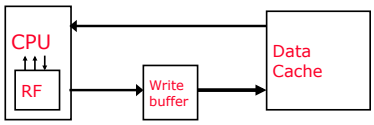
Reducing Write Hit Time

Problem: Writes take two cycles in memory stage, one cycle for tag check plus one cycle for data write if hit

Solutions:

- Design data RAM that can perform read and write in one cycle, restore old value after tag miss
- Write buffer: Hold write data for store in single buffer ahead of cache, and write returns immediately

Write Buffer to Reduce Read Miss Penalty



Processor is not stalled on writes, and read misses can go ahead of write to main memory

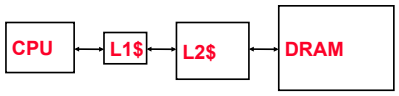
Problem: Write buffer may hold updated value of location needed by a read miss

Simple scheme: on a read miss, wait for the write buffer to go empty

Faster scheme: Check write buffer addresses against read miss addresses, if no match, allow read miss to go ahead of writes, else, return value in write buffer

Multilevel Caches

Problem: A memory cannot be large and fast
Solution: Increasing sizes of cache at each level



Local miss rate = misses in cache / accesses to cache

Global miss rate = misses in cache / CPU memory accesses

Misses per instruction = misses in cache / number of instructions

Presence of L2 influences L1 design

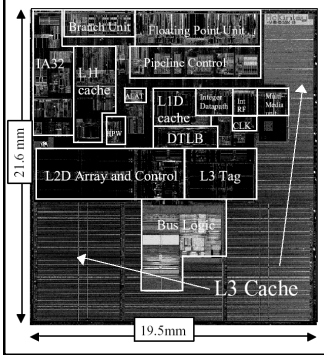
- Use smaller L1 if there is also L2
  - Trade increased L1 miss rate for reduced L1 hit time and reduced L1 miss penalty
- Use simpler write-through L1 with on-chip L2
  - Write-back L2 cache absorbs write traffic, doesn't go off-chip
  - At most one L1 miss request per L1 access (no dirty victim write back) simplifies pipeline control
  - Simplifies error recovery in L1 (can use just parity bits in L1 and reload from L2 when parity error detected on L1 read)

Inclusion Policy

- Inclusive multilevel cache:
  - Inner cache holds copies of data in outer cache
  - External coherence snoop access need only check outer cache
- *Exclusive* multilevel caches:
  - Inner cache may hold data not in outer cache
  - Swap lines between inner/outer caches on miss
  - Used in AMD Athlon with 64KB primary and 256KB secondary cache

Why choose one type or the other?

Itanium-2 On-Chip Caches
(Intel/HP, 2002)



Level 1: 16KB, 4-way s.a., 64B line, quad-port (2 load+2 store), single cycle latency

Level 2: 256KB, 4-way s.a, 128B line, quad-port (4 load or 4 store), five cycle latency

Level 3: 3MB, 12-way s.a., 128B line, single 32B port, twelve cycle latency

Power 7 On-Chip Caches [IBM 2009]

32KB L1 I$/core

32KB L1 D$/core

3-cycle latency

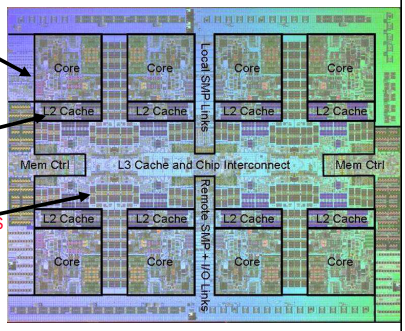
256KB Unified L2$/core

8-cycle latency

32MB Unified Shared L3$

Embedded DRAM

25-cycle latency to local slice



CSE 490/590 Administrivia

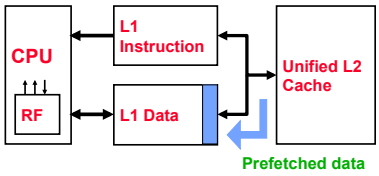
- Very important to attend
  - Recitations this week
- Quiz 1
  - Grading will be done by next Monday.
- Office hours changed
  - Tuesdays 3pm-6pm

Prefetching

- Speculate on future instruction and data accesses and fetch them into cache(s)
  - Instruction accesses easier to predict than data accesses
- Varieties of prefetching
  - Hardware prefetching
  - Software prefetching
  - Mixed schemes
- *What types of misses does prefetching affect?*

Issues in Prefetching

- Usefulness – should produce hits
- Timeliness – not late and not too early
- Cache and bandwidth pollution



Acknowledgements

- These slides heavily contain material developed and copyright by
  - Krste Asanovic (MIT/UCB)
  - David Patterson (UCB)
- And also by:
  - Arvind (MIT)
  - Joel Emer (Intel/MIT)
  - James Hoe (CMU)
  - John Kubiatowicz (UCB)
- MIT material derived from course 6.823
- UCB material derived from course CS252