CS 61C: Great Ideas in Computer Architecture (Machine Structures)
Caches Part 3

Instructors:

Bernhard Boser & Randy H. Katz

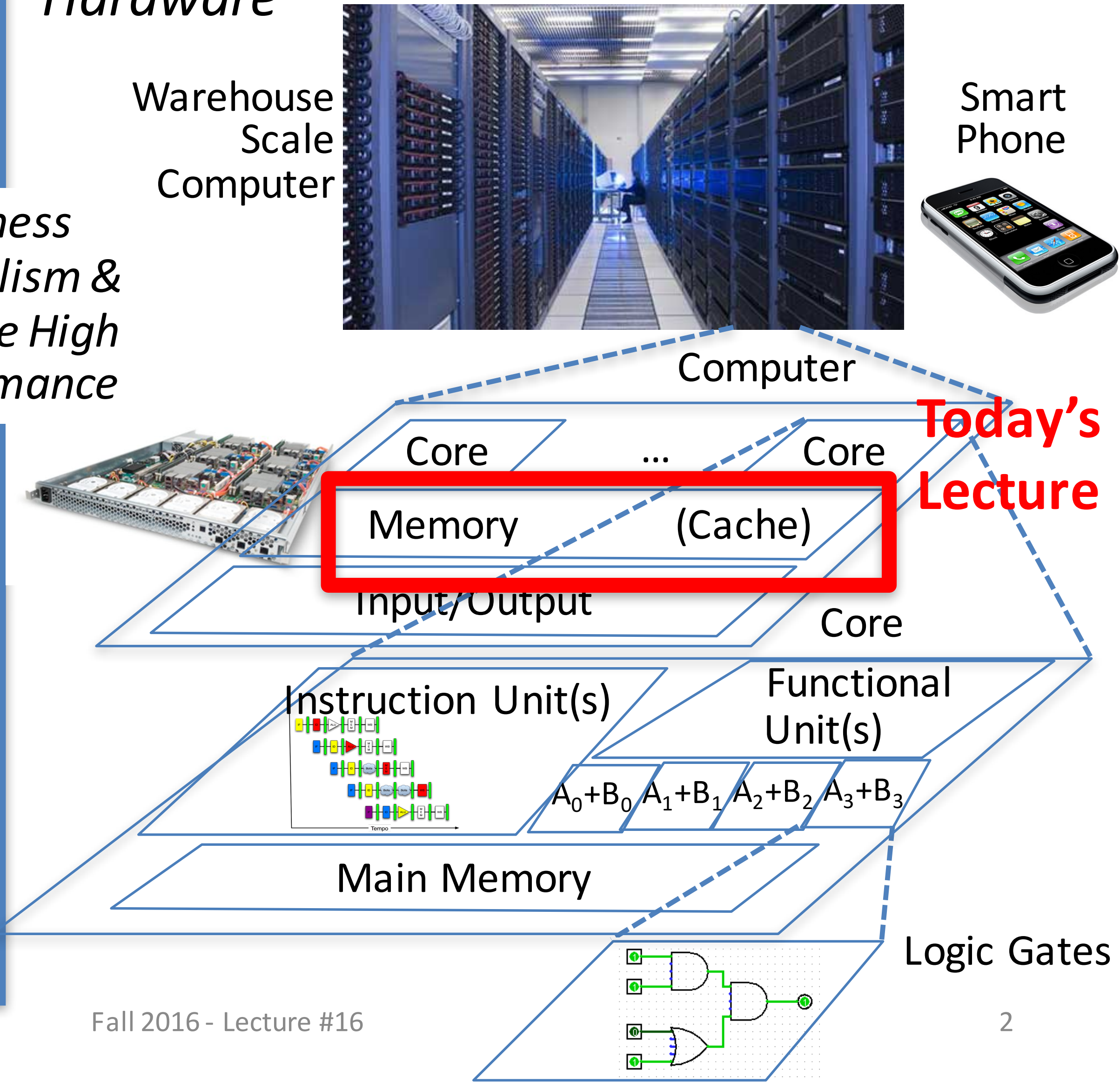
http://inst.eecs.berkeley.edu/~cs61c/

You Are Here!

Software | *Hardware*

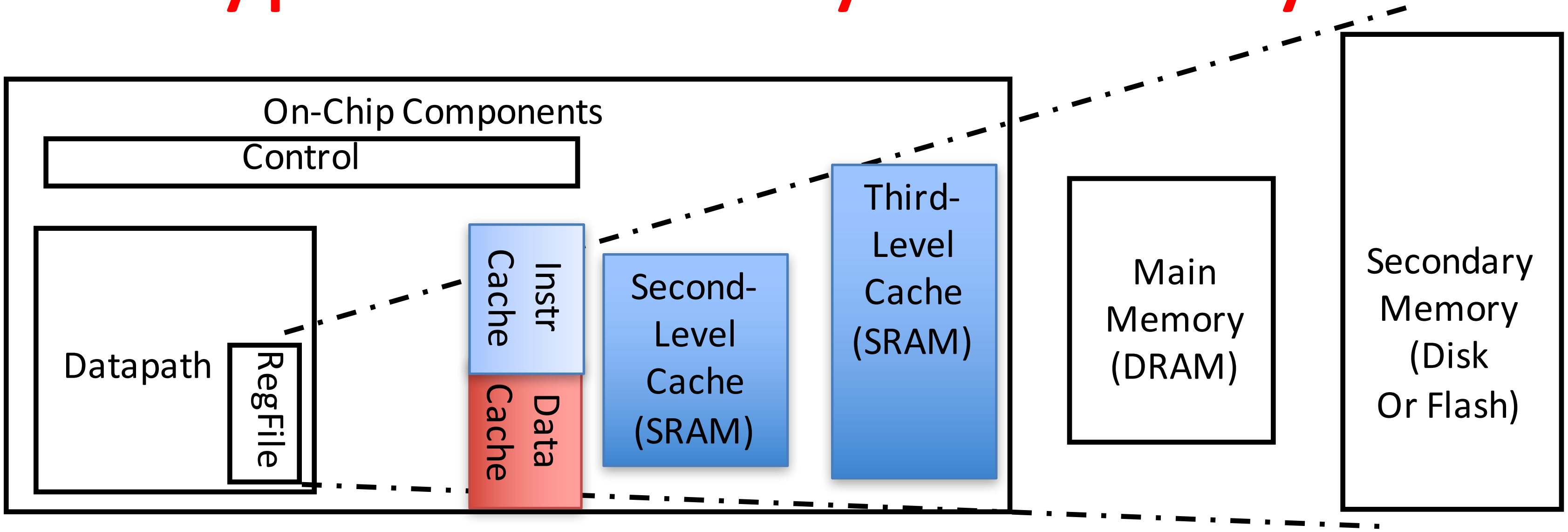
- Parallel Requests
  Assigned to computer
  e.g., Search "Katz"
- Parallel Threads
  Assigned to core
  e.g., Lookup, Ads
- Parallel Instructions
  >1 instruction @ one time
  e.g., 5 pipelined instructions
- Parallel Data
  >1 data item @ one time
  e.g., Add of 4 pairs of words
- Hardware descriptions
  All gates @ one time
- Programming Languages

Harness Parallelism & Achieve High Performance



Today's Lecture

Typical Memory Hierarchy



On-Chip Components

Control

Datapath

RegFile

Instr Cache

Data Cache

Second-Level Cache (SRAM)

Third-Level Cache (SRAM)

Main Memory (DRAM)

Secondary Memory (Disk Or Flash)

Speed (cycles):	½'s	1's	10's	100's-1000	1,000,000's
Size (bytes):	100's	10K's	M's	G's	T's
Cost/bit:	highest	↔			lowest

- Principle of locality + memory hierarchy presents programmer with ≈ as much memory as is available in the *cheapest* technology at the ≈ speed offered by the *fastest* technology

Write Policy Choices

- Cache Hit:
  - **Write through**: writes both cache & memory on every access
    - Generally higher memory traffic but simpler pipeline & cache design
  - **Write back**: writes cache only, memory written only when dirty entry evicted
    - A dirty bit per line 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

Average Memory Access Time (AMAT)

- Average Memory Access Time (AMAT) is the average time to access memory considering both hits and misses in the cache

AMAT = Time for a hit
+ Miss rate × Miss penalty

Important Equation!

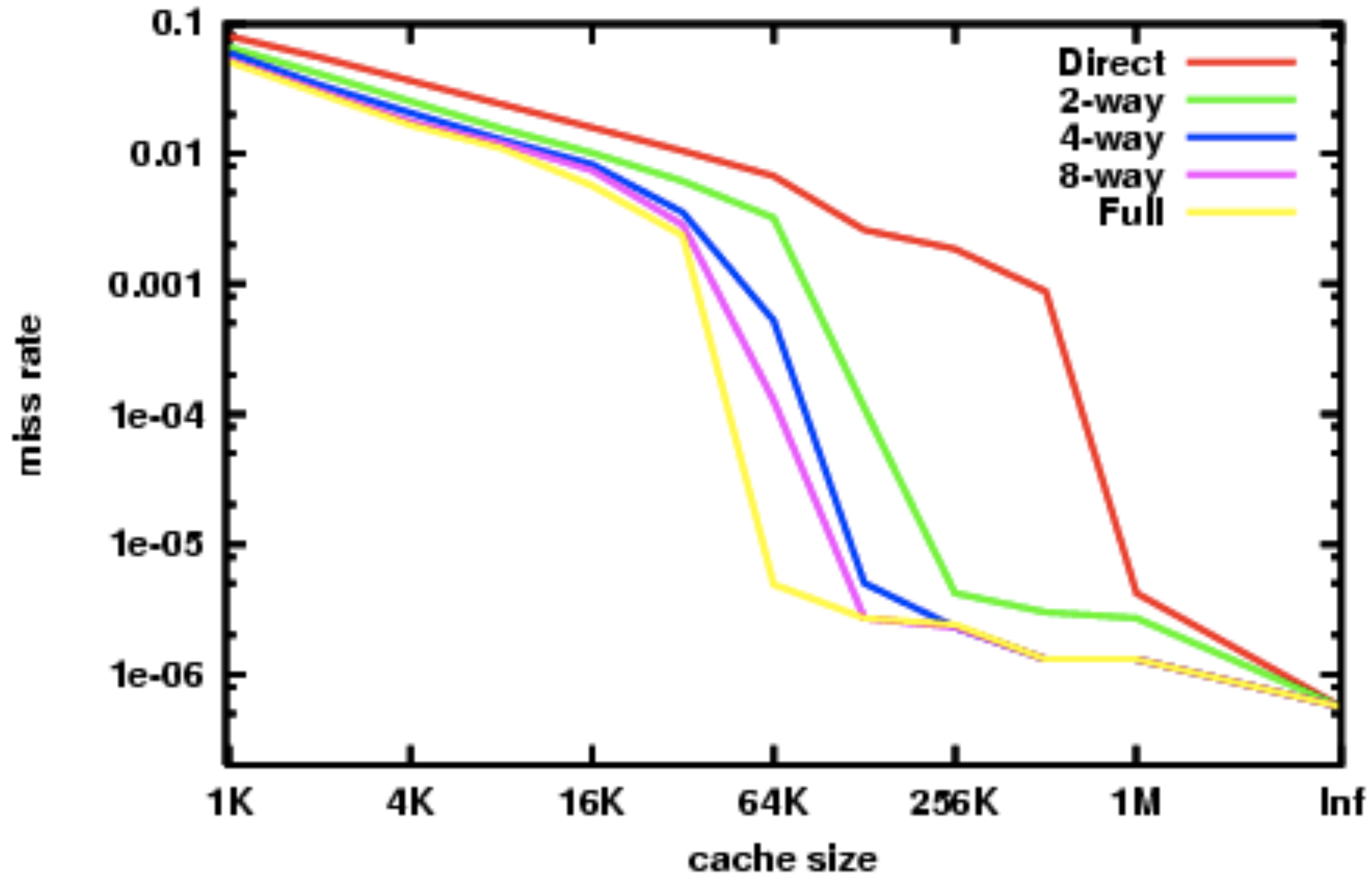
Outline

- Understanding Cache Misses
- Increasing Cache Performance
- Performance of multi-level Caches (L1,L2, …)
- Real world example caches
- And in Conclusion …

Outline

- Understanding Cache Misses
- Increasing Cache Performance
- Performance of multi-level Caches (L1,L2, …)
- Real world example caches
- And in Conclusion …

Miss Rate vs. Cache Size on the Integer Portion of SPECCPU2000



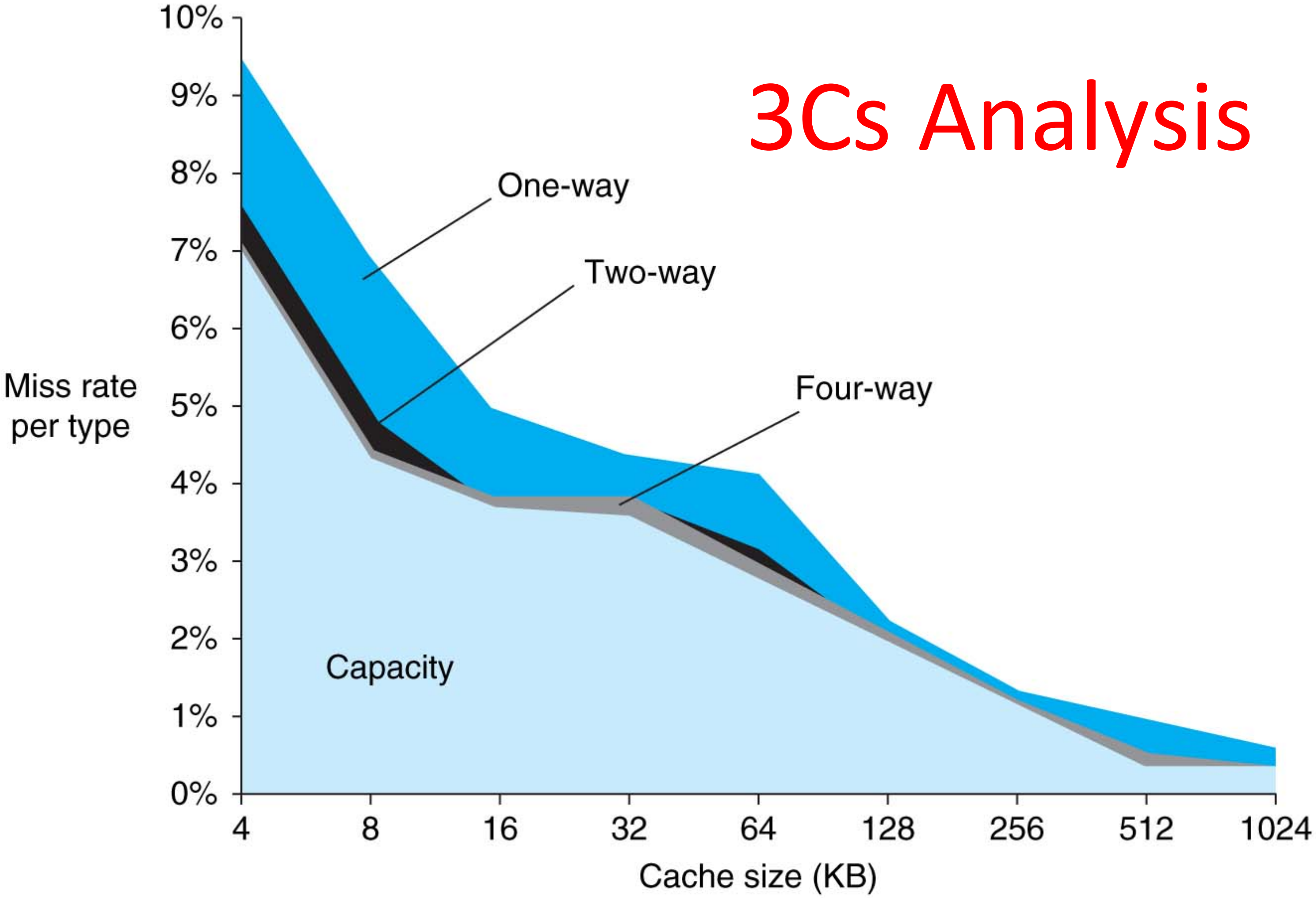
Sources of Cache Misses (3 C's)

- *Compulsory* (cold start, first reference):
  - 1st access to a block, not a lot you can do about it
    - If running billions of instructions, compulsory misses are insignificant
- *Capacity*:
  - Cache cannot contain all blocks accessed by the program
    - Misses that would not occur with infinite cache
- *Conflict* (collision):
  - Multiple memory locations mapped to same cache set
    - Misses that would not occur with ideal fully associative cache

How to Calculate 3C's Using Cache Simulator

1. *Compulsory*: set cache size to infinity and fully associative, and count number of misses
2. *Capacity*: Change cache size from infinity, usually in powers of 2, and count misses for each reduction in size
   - 16 MB, 8 MB, 4 MB, … 128 KB, 64 KB, 16 KB
3. *Conflict*: Change from fully associative to n-way set associative while counting misses
   - Fully associative, 16-way, 8-way, 4-way, 2-way, 1-way

3Cs Analysis

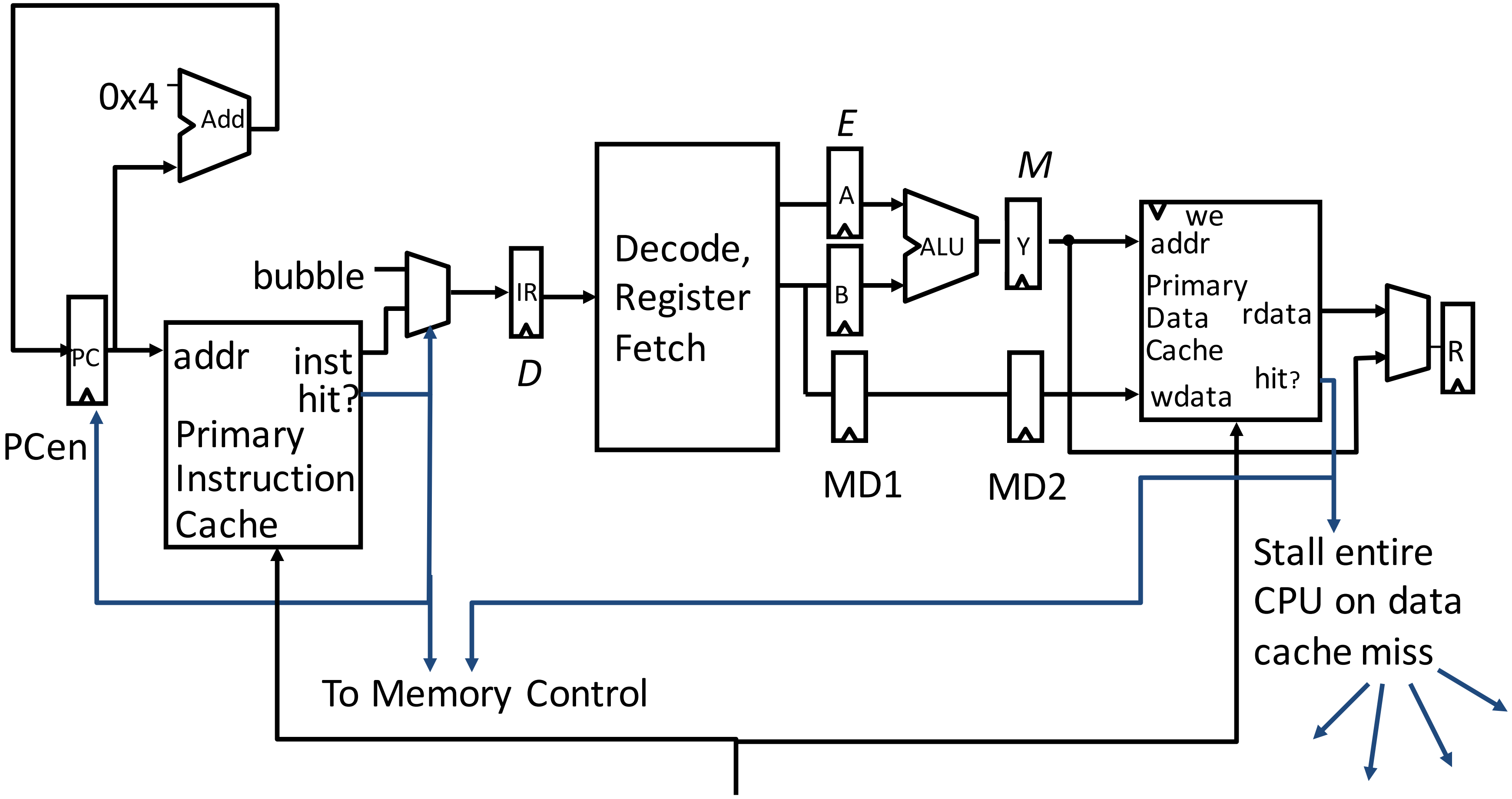


- Three sources of misses (SPEC2000 integer and floating-point benchmarks)
  - Compulsory misses 0.006%; not visible
  - Capacity misses, function of cache size
  - Conflict portion depends on associativity and cache size

Outline

- Understanding Cache Misses
- Increasing Cache Performance
- Performance of multi-level Caches (L1,L2, …)
- Real world example caches
- And in Conclusion …

CPU-Cache Interaction
(5-stage pipeline)



Stall entire CPU on data cache miss

Cache Refill Data from Lower Levels of Memory Hierarchy

Improving Cache Performance

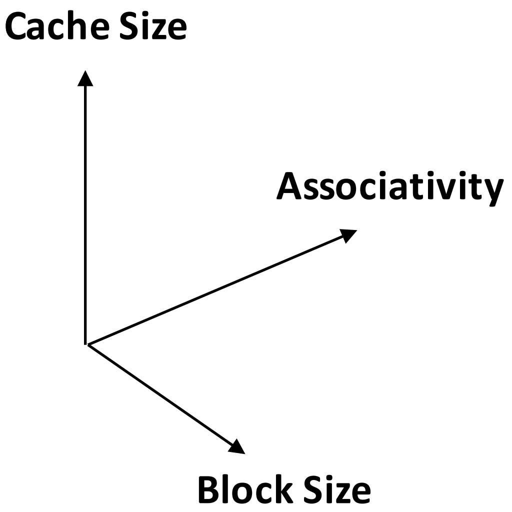
AMAT = Time for a hit + Miss rate x Miss penalty

- Reduce the time to hit in the cache
  - E.g., Smaller cache
- Reduce the miss rate
  - E.g., Bigger cache
- Reduce the miss penalty
  - E.g., Use multiple cache levels

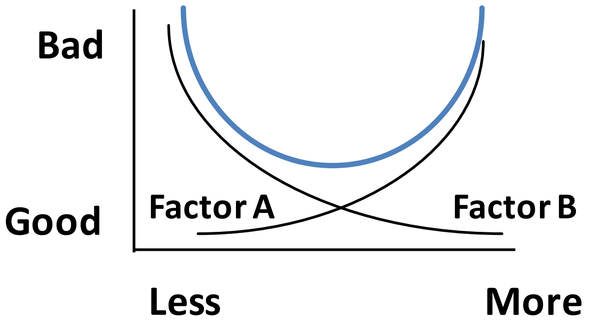
Cache Design Space

Computer architects expend considerable effort optimizing organization of cache hierarchy – big impact on performance and power!

- Several interacting dimensions
  - Cache size
  - Block size
  - Associativity
  - Replacement policy
  - Write-through vs. write-back
  - Write allocation



- Optimal choice is a compromise
  - Depends on access characteristics
    - Workload
    - Use (I-cache, D-cache)
  - Depends on technology / cost
- Simplicity often wins



Primary Cache Parameters

- Block size
  - How many bytes of data in each cache entry?
- Associativity
  - How many ways in each set?
  - Direct-mapped => Associativity = 1
  - Set-associative => 1 < Associativity < #Entries
  - Fully associative => Associativity = #Entries
- Capacity (bytes) = Total #Entries * Block size
- #Entries = #Sets * Associativity

Impact of Larger Cache on AMAT?

- 1) Reduces misses (what kind(s)?)
- 2) Longer Access time (Hit time): smaller is faster
  - Increase in hit time will likely add another stage to the pipeline
- At some point, increase in hit time for a larger cache may overcome the improvement in hit rate, yielding a decrease in performance
- Computer architects expend considerable effort optimizing organization of cache hierarchy – big impact on performance and power!

Increasing Associativity?

- Hit time as associativity increases?
  - Increases, with large step from direct-mapped to >=2 ways, as now need to mux correct way to processor
  - Smaller increases in hit time for further increases in associativity
- Miss rate as associativity increases?
  - Goes down due to reduced conflict misses, but most gain is from 1->2->4-way with limited benefit from higher associativities
- Miss penalty as associativity increases?
  - Unchanged, replacement policy runs in parallel with fetching missing line from memory

Increasing #Entries?

- Hit time as #entries increases?
  - Increases, since reading tags and data from larger memory structures
- Miss rate as #entries increases?
  - Goes down due to reduced capacity and conflict misses
  - *Architects rule of thumb: miss rate drops ~2x for every ~4x increase in capacity (only a gross approximation)*
- Miss penalty as #entries increases?
  - Unchanged

At some point, increase in hit time for a larger cache may overcome the improvement in hit rate, yielding a decrease in performance

Increasing Block Size?

- Hit time as block size increases?
  - Hit time unchanged, but might be slight hit-time reduction as number of tags is reduced, so faster to access memory holding tags
- Miss rate as block size increases?
  - Goes down at first due to spatial locality, then increases due to increased conflict misses due to fewer blocks in cache
- Miss penalty as block size increases?
  - Rises with longer block size, but with fixed constant initial latency that is amortized over whole block

Administrivia

- Midterm #2 1.5 weeks away! November 1!
  - In class! 3:40-5 PM
  - Synchronous digital design and Project 3 (processor design) included
  - Pipelines and Caches
  - ONE Double sided Crib sheet
  - Review Session, Sunday, 10/30, 1-3 PM, ~~10 Evans~~

    155 Dwinelle

Clickers/Peer Instruction

For a cache of fixed capacity and blocksize, what is the impact of increasing associativity on AMAT:

[✓ A: Increases hit time, decreases miss rate]

B: Decreases hit time, decreases miss rate

C: Increases hit time, increases miss rate

D: Decreases hit time, increases miss rate

Clickers/Peer Instruction

Impact of Larger Blocks on **AMAT**:

- For fixed total cache capacity and associativity, what is effect of larger blocks on each component of AMAT:

  A: Decrease

  B: Unchanged

  C: Increase

Shorter tags +, mux at edge -

Hit Time? C: Unchanged (but slight increase possible)

Miss Rate? A: Decrease (spatial locality; conflict???)

Miss Penalty? C: Increase (longer time to load block)

Write Allocation? It depends!

Clickers/Peer Instruction

Impact of Larger Blocks on **Misses**:

- For fixed total cache capacity and associativity, what is effect of larger blocks on each component of miss:

  A: Decrease

  B: Unchanged

  C: Increase

Compulsory? A: Decrease (if good Spatial Locality)

Capacity? B: Increase (smaller blocks fit better)

Conflict? A: Increase (more ways better!)
Less effect for large caches

How to Reduce Miss Penalty?

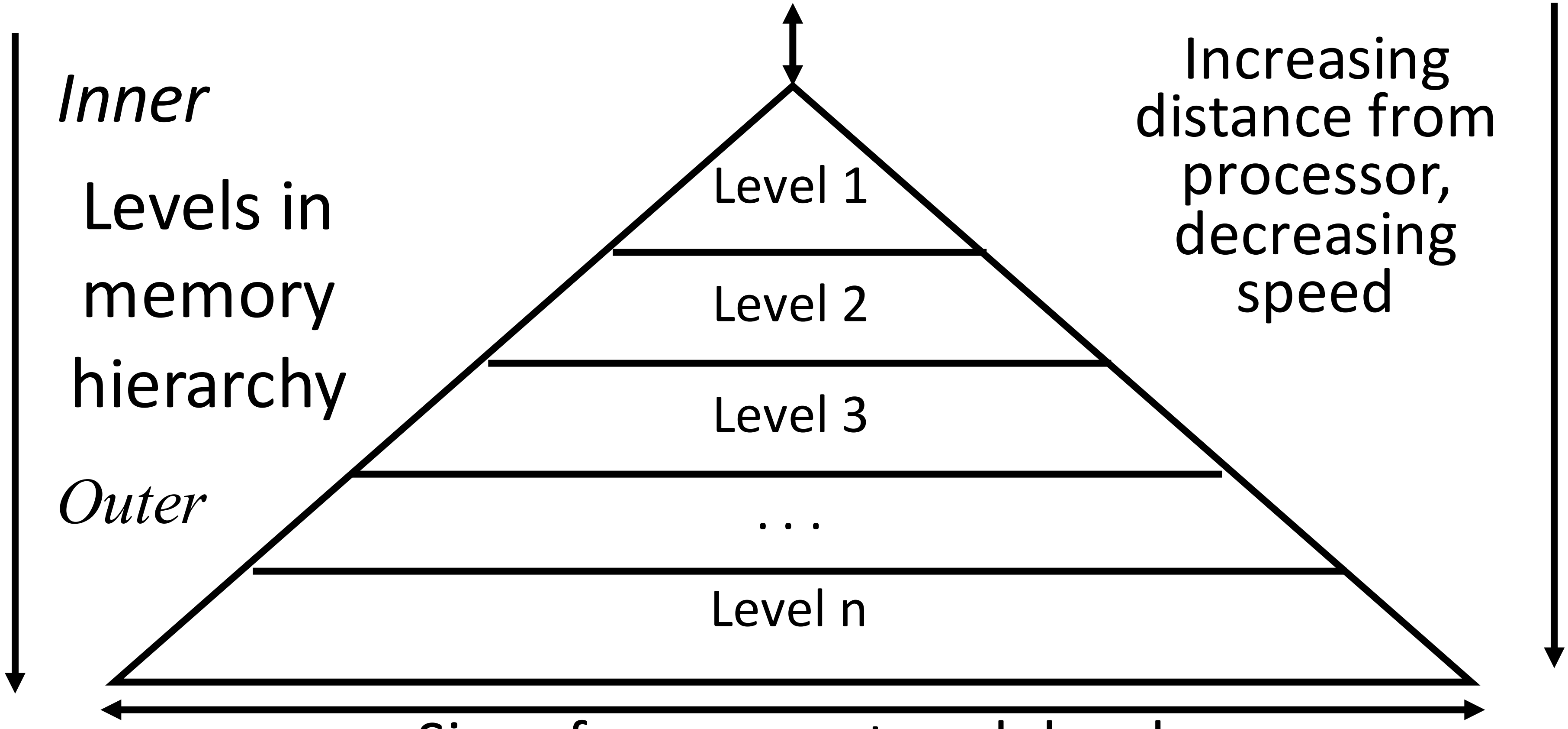
- Could there be locality on misses from a cache?
  - Use multiple cache levels!
  - With Moore's Law, more room on die for bigger L1$ and for second-level L2$
  - And in some cases even an L3$!
- Mainframes have ~1GB L4 cache off-chip

Outline

- Understanding Cache Misses
- Increasing Cache Performance
- Performance of Multi-level Caches (L1,L2, …)
- Real world example caches
- And in Conclusion …

Memory Hierarchy

Processor



Inner

Levels in memory hierarchy

Outer

Level 1

Level 2

Level 3

. . .

Level n

Increasing distance from processor, decreasing speed

Size of memory at each level

As we move to outer levels the latency goes up and price per bit goes down.

Local vs. Global Miss Rates

- *Local miss rate:* the fraction of references to one level of a cache that miss
  - Local Miss rate L2$ = L2$ Misses / L1$ Misses
    = L2$ Misses / total_L2_accesses
- *Global miss rate:* the fraction of references that miss in all levels of a multilevel cache
  - L2$ local miss rate >> than the global miss rate

Local vs. Global Miss Rates

- *Local miss rate* – the fraction of references to one level of a cache that miss
  - Local Miss rate L2$ = $L2 Misses / L1$ Misses
- *Global miss rate* – the fraction of references that miss in all levels of a multilevel cache
  - L2$ local miss rate >> than the global miss rate
- Global Miss rate = L2$ Misses / Total Accesses

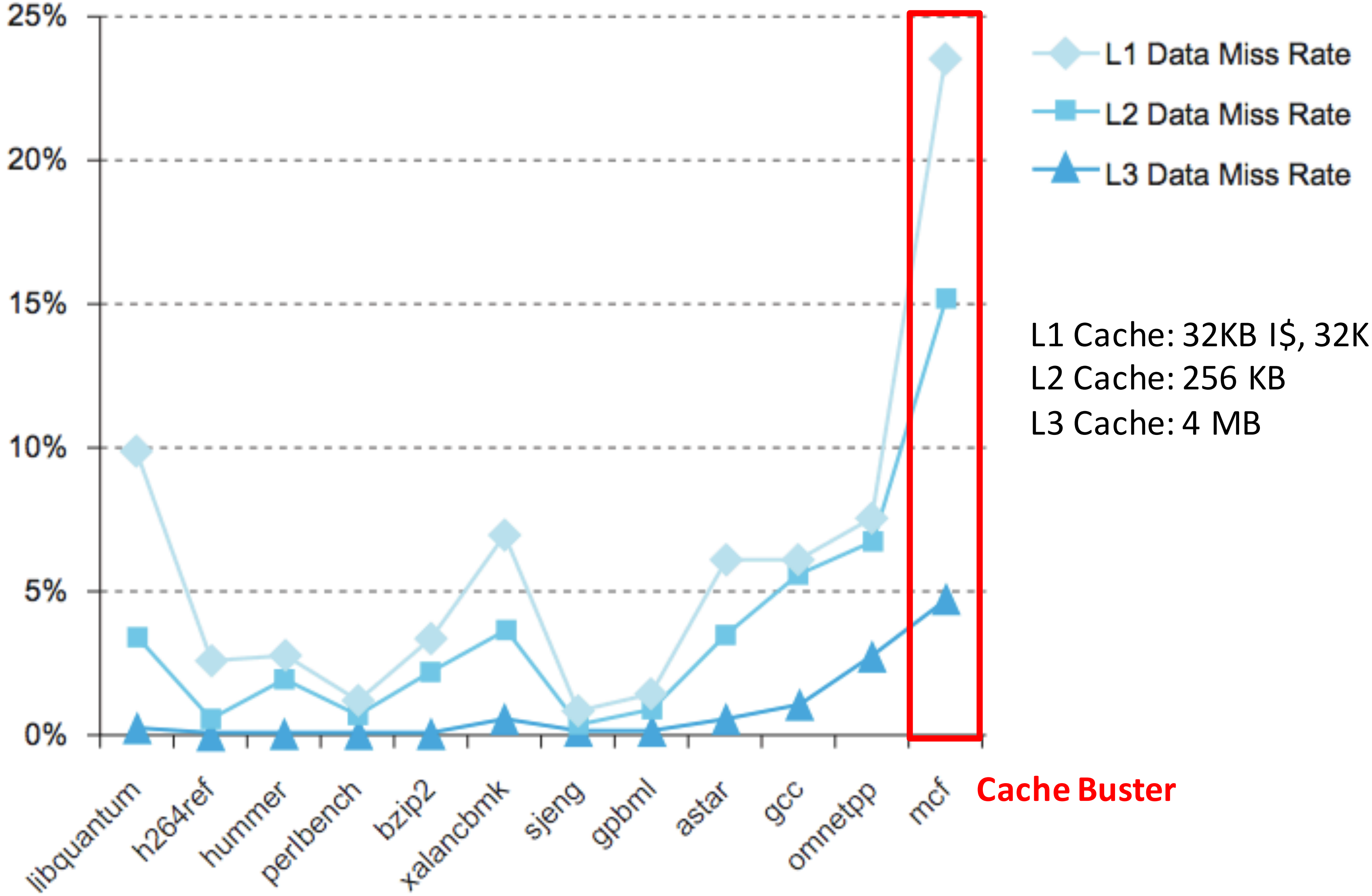
  = (L2$ Misses / L1$ Misses) × (L1$ Misses / Total Accesses)

  = Local Miss rate L2$ × Local Miss rate L1$

- AMAT = Time for a hit + Miss rate × Miss penalty
- AMAT = Time for a L1$ hit + (local) Miss rate L1$ × (Time for a L2$ hit + (local) Miss rate L2$ × L2$ Miss penalty)

Multilevel Cache Considerations

- Different design considerations for L1$ and L2$
  - L1$ focuses on fast access: minimize hit time to achieve shorter clock cycle, e.g., smaller $
  - L2$, L3$ focus on low miss rate: reduce penalty of long main memory access times: e.g., Larger $ with larger block sizes/higher levels of associativity
- Miss penalty of L1$ is significantly reduced by presence of L2$, so can be smaller/faster even with higher miss rate
- For the L2$, fast hit time is less important than low miss rate
  - L2$ hit time determines L1$'s miss penalty
  - L2$ local miss rate >> than the global miss rate



L1 Cache: 32KB I$, 32KB D$

L2 Cache: 256 KB

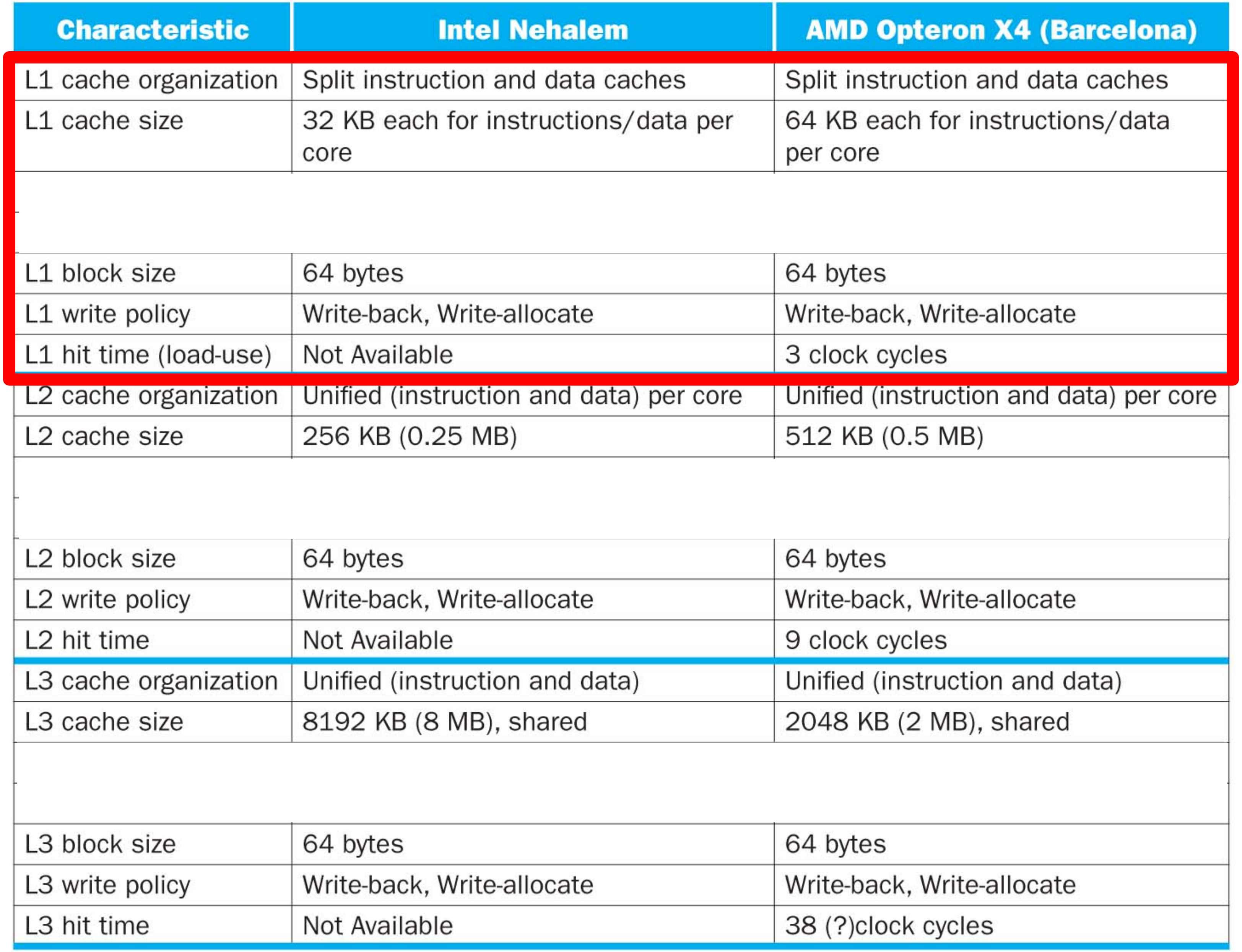
L3 Cache: 4 MB

Cache Buster

FIGURE 5.47 The L1, L2, and L3 data cache miss rates for the Intel Core i7 920 running the full integer SPECCPU2006 benchmarks.

Outline

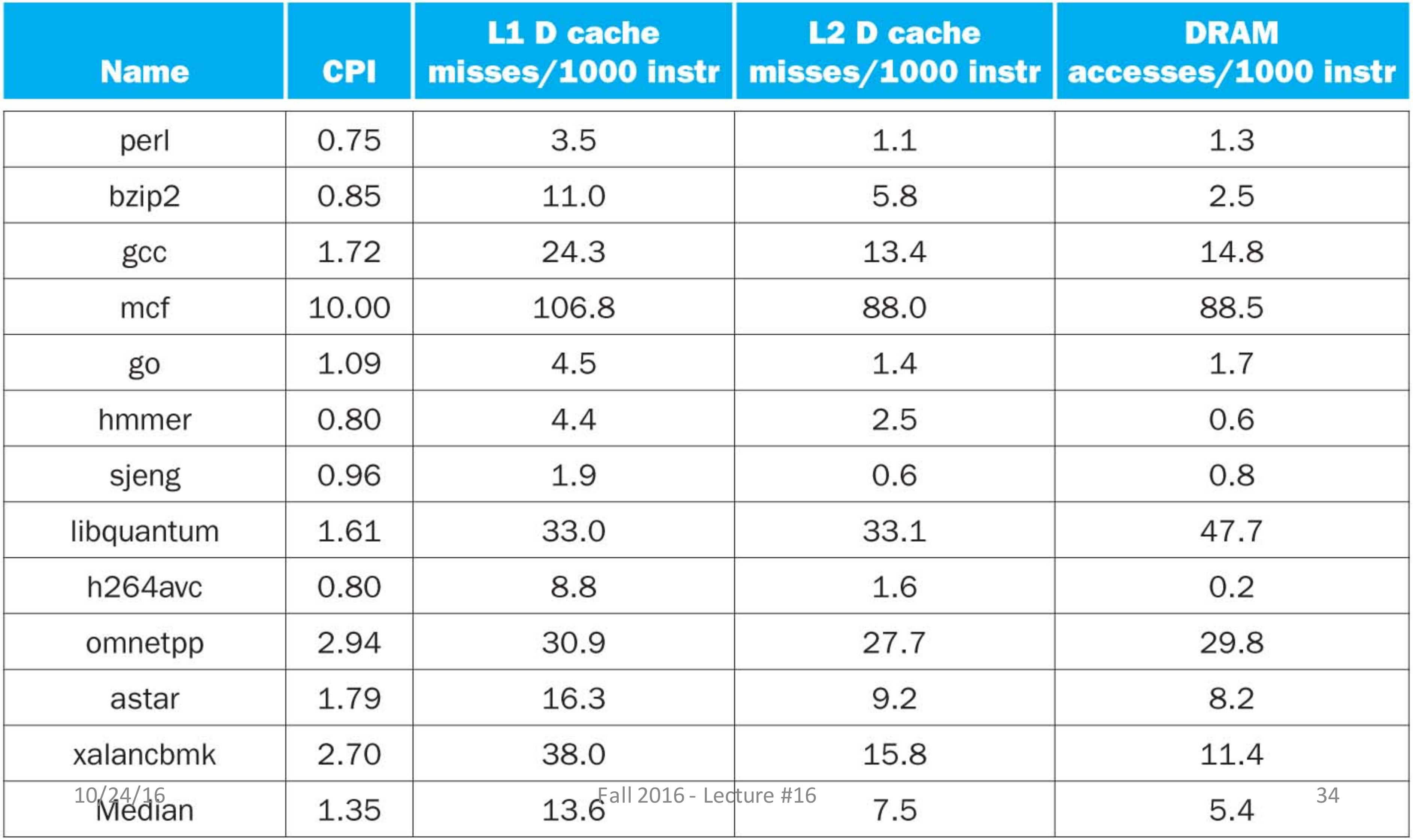
- Understanding Cache Misses
- Increasing Cache Performance
- Performance of Multi-level Caches (L1,L2, …)
- Real world example caches
- And in Conclusion …



Characteristic	Intel Nehalem	AMD Opteron X4 (Barcelona)
L1 cache organization	Split instruction and data caches	Split instruction and data caches
L1 cache size	32 KB each for instructions/data per core	64 KB each for instructions/data per core
L1 block size	64 bytes	64 bytes
L1 write policy	Write-back, Write-allocate	Write-back, Write-allocate
L1 hit time (load-use)	Not Available	3 clock cycles
L2 cache organization	Unified (instruction and data) per core	Unified (instruction and data) per core
L2 cache size	256 KB (0.25 MB)	512 KB (0.5 MB)
L2 block size	64 bytes	64 bytes
L2 write policy	Write-back, Write-allocate	Write-back, Write-allocate
L2 hit time	Not Available	9 clock cycles
L3 cache organization	Unified (instruction and data)	Unified (instruction and data)
L3 cache size	8192 KB (8 MB), shared	2048 KB (2 MB), shared
L3 block size	64 bytes	64 bytes
L3 write policy	Write-back, Write-allocate	Write-back, Write-allocate
L3 hit time	Not Available	38 (?)clock cycles

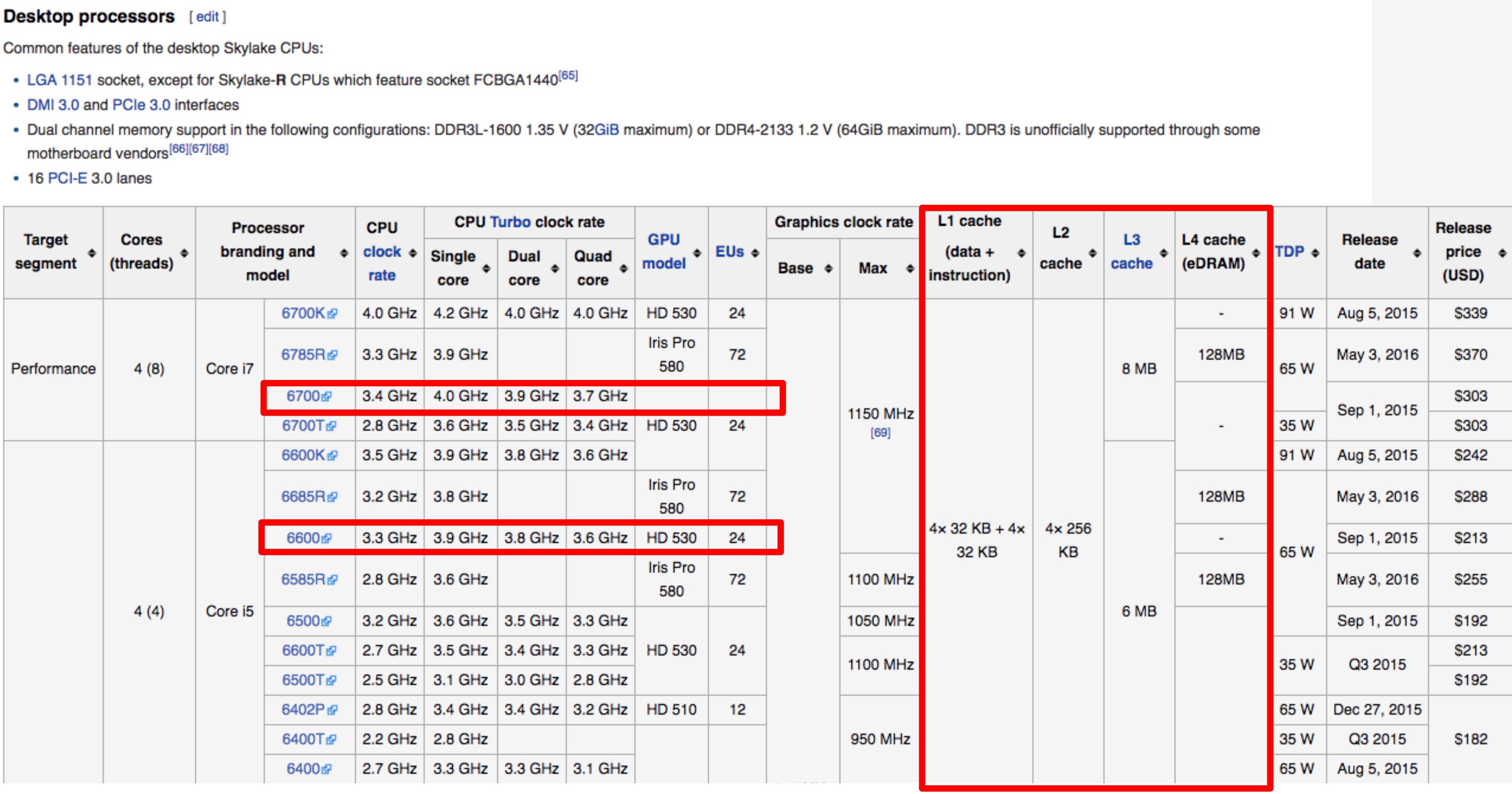
CPI/Miss Rates/DRAM Access SpecInt2006

Data Only | Data Only | Instructions and Data



Name	CPI	L1 D cache misses/1000 instr	L2 D cache misses/1000 instr	DRAM accesses/1000 instr
perl	0.75	3.5	1.1	1.3
bzip2	0.85	11.0	5.8	2.5
gcc	1.72	24.3	13.4	14.8
mcf	10.00	106.8	88.0	88.5
go	1.09	4.5	1.4	1.7
hmmer	0.80	4.4	2.5	0.6
sjeng	0.96	1.9	0.6	0.8
libquantum	1.61	33.0	33.1	47.7
h264avc	0.80	8.8	1.6	0.2
omnetpp	2.94	30.9	27.7	29.8
astar	1.79	16.3	9.2	8.2
xalancbmk	2.70	38.0	15.8	11.4
Median	1.35	13.6	7.5	5.4

Skylark: Intel's Latest Generation Laptop/Tablet Class CPUs



Desktop processors [edit]

Common features of the desktop Skylake CPUs:

- LGA 1151 socket, except for Skylake-**R** CPUs which feature socket FCBGA1440[65]
- DMI 3.0 and PCIe 3.0 interfaces
- Dual channel memory support in the following configurations: DDR3L-1600 1.35 V (32GiB maximum) or DDR4-2133 1.2 V (64GiB maximum). DDR3 is unofficially supported through some motherboard vendors[66][67][68]
- 16 PCI-E 3.0 lanes

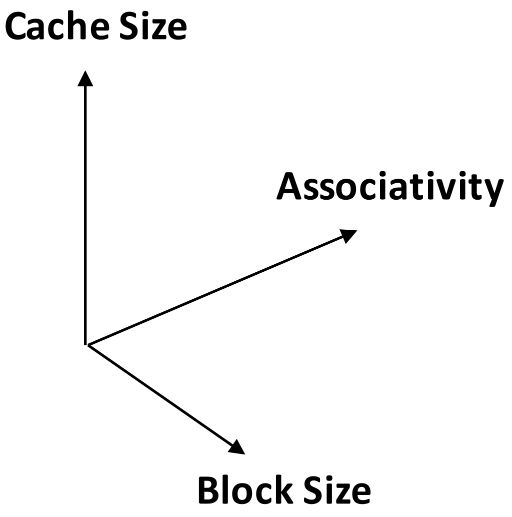
Target segment	Cores (threads)	Processor branding and model		CPU clock rate	CPU Turbo clock rate			GPU model	EUs	Graphics clock rate		L1 cache (data + instruction)	L2 cache	L3 cache	L4 cache (eDRAM)	TDP	Release date	Release price (USD)
					Single core	Dual core	Quad core			Base	Max							
Performance	4 (8)	Core i7	6700K	4.0 GHz	4.2 GHz	4.0 GHz	4.0 GHz	HD 530	24		1150 MHz [69]	4× 32 KB + 4× 32 KB	4× 256 KB	8 MB	-	91 W	Aug 5, 2015	$339
			6785R	3.3 GHz	3.9 GHz			Iris Pro 580	72						128MB	65 W	May 3, 2016	$370
			6700	3.4 GHz	4.0 GHz	3.9 GHz	3.7 GHz	HD 530	24						-		Sep 1, 2015	$303
			6700T	2.8 GHz	3.6 GHz	3.5 GHz	3.4 GHz									35 W		$303
	4 (4)	Core i5	6600K	3.5 GHz	3.9 GHz	3.8 GHz	3.6 GHz							6 MB		91 W	Aug 5, 2015	$242
			6685R	3.2 GHz	3.8 GHz			Iris Pro 580	72						128MB	65 W	May 3, 2016	$288
			6600	3.3 GHz	3.9 GHz	3.8 GHz	3.6 GHz	HD 530	24						-		Sep 1, 2015	$213
			6585R	2.8 GHz	3.6 GHz			Iris Pro 580	72		1100 MHz				128MB		May 3, 2016	$255
			6500	3.2 GHz	3.6 GHz	3.5 GHz	3.3 GHz	HD 530	24		1050 MHz						Sep 1, 2015	$192
			6600T	2.7 GHz	3.5 GHz	3.4 GHz	3.3 GHz				1100 MHz					35 W	Q3 2015	$213
			6500T	2.5 GHz	3.1 GHz	3.0 GHz	2.8 GHz											$192
			6402P	2.8 GHz	3.4 GHz	3.4 GHz	3.2 GHz	HD 510	12		950 MHz					65 W	Dec 27, 2015	$182
			6400T	2.2 GHz	2.8 GHz											35 W	Q3 2015	
			6400	2.7 GHz	3.3 GHz	3.3 GHz	3.1 GHz									65 W	Aug 5, 2015	

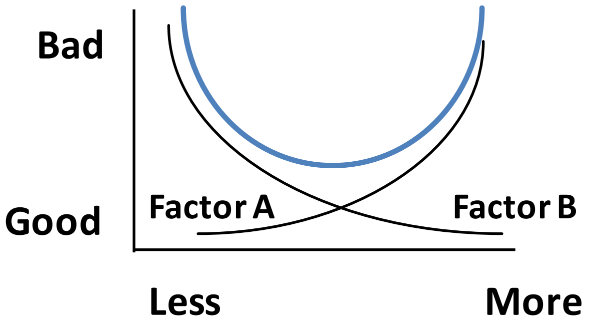
Outline

- Understanding Cache Misses
- Increasing Cache Performance
- Performance of Multi-level Caches (L1,L2, …)
- Real world example caches
- And in Conclusion …

Bottom Line: Cache Design Space

- Several interacting dimensions
  - Cache size
  - Block size
  - Associativity
  - Replacement policy
  - Write-through vs. write-back
  - Write allocation
- Optimal choice is a compromise
  - Depends on access characteristics
    - Workload
    - Use (I-cache, D-cache)
  - Depends on technology / cost
- Simplicity often wins


Cache Size
Associativity
Block Size


Bad
Good
Factor A
Factor B
Less
More