Cache Performance and Basic Optimization

Joannah Nanjekye

July 22, 2024

Direct-Mapping Example

A cache is direct-mapped and has 64 KB data. Each block contains 32 bytes. The address is 32 bits wide. What are the sizes of the tag, index, and block offset fields?

- bits in block offset = 5, each block contains 32 = 2^5 bytes
- blocks in cache = 64 $\times$ 1024 / 32 = 2048 blocks
- bits in index field = 11, there are 2^{11} blocks
- bits in tag field = 32 - 5 - 11 = 16

Set-associative Example

A cache is 4-way set-associative and has 64 KB data. Each block contains 32 bytes. The address is 32 bits wide. What are the sizes of the tag, index, and block offset fields?

- bits in block offset = 5, each block contains $32 = 2^5$ bytes
- blocks in cache = $64 \times 1024 / 32 = 2048$, 2^{11}
- sets in cache = $2048 / 4 = 512$, 2^9 sets (a set is 4 blocks kept in the cache for each index)
- bits in index field = 9
- bits in tag field = 32 - 5 - 9 = 18

Average Memory Access Time (AMAT)

$AMAT = Hit\ Time + Miss\ Rate \times Miss\ Penalty$
$AMAT = T_{hit}(L1) + Miss\%(L1) \times T(Mem)$
Assume:

- Cache Hit = 3 cycles
- Miss rate = 20%, Miss penalty = 500 cycles

Then:

- $AMAT = 3 + 0.2 \times 500 = 103\ cycles$

CPU Time

$$\textit{CPU time} = (\textit{CPU execution clock cycles} + \textit{Memory} - \textit{stall clock cycles}) \times \textit{Clock cycle time}$$

$$\textit{Memory} - \textit{stall clock cycles} = \textit{Read} - \textit{stall cycles} + \textit{Write} - \textit{stall cycles}$$

$$\textit{Read} - \textit{stall cycles} = \frac{\textit{Reads}}{\textit{Program}} \times \textit{Read miss rate} \times \textit{Read miss penalty}$$

$$\textit{Write} - \textit{stall cycles} = (\frac{\textit{Reads}}{\textit{Program}} \times \textit{Write miss rate} \times \textit{Write miss penalty}) + \textit{Write buffer stall}$$

Combining Read and Write Stall Cycles

Using a single miss rate and miss penalty, the write and read miss penalties are the same, i.e. time to fetch a block from main memory:

$$Memory - stall\ clock\ cycles = \frac{Memory\ \ accesses}{Program} \times Miss\ rate \times Miss\ penalty$$

$$Memory - stall\ clock\ cycles = \frac{Instructions}{Program} x \frac{Miss}{Instruction} \times Miss\ penalty$$

Example: Cache Performance

Consider:

- Instruction miss rate = 2%
- Data miss rate = 4%
- CPI = 2 (without memory stalls)
- Miss penalty = 40 cycles
- 36% of instructions are load/store

Determine how much faster a machine would run with a perfect cache that never missed[1]

- Instruction miss cycles = I x 0.02 x 40 = 0.80 I
- Data miss cycles = I x 0.36 x 0.04 x 40 = 0.58 I
- Total memory stall cycles = 0.80 I + 0.58 I = 1.38 I
- $CPI_{stall} = 2 + 1.38 = 3.38$

$$\frac{CPU\ time_{stalls}}{CPU\ time_{perfect\ cache}} = \frac{I \times CPI_{stall} x Clock\ cycle}{I x CPI_{perfect} x Clock\ cycle} = \frac{3.38}{2} = 1.69$$

[1] I = number of instructions

Example: Increased Clock Rate

Assume the clock rate of the machine used in this example is doubled but the memory speed, cache misses, and miss rate are same. How much faster the machine be with the faster clock?

- New miss penalty = 2x40=80 (clock rate is doubled)
- Total memory stall cycles = (0.02 x 80) + 0.36 x (0.04 x 80) = 2.75
- $CPI_{fastclock} = 2 + 2.75 = 4.75 clock\ cycles$

$$\frac{CPU\ time_{slowclock}}{CPU\ time_{fastclock}} = \frac{I \times CPI_{slowclock} \times Clock\ cycle}{IxCPI_{fastclock}\, x\, \frac{Clockcycle}{2}}$$

$$= \frac{3.38x2}{4.75} = 1.41$$

Example: AMAT direct and Set Associative Mapping

- If a direct mapped cache has a hit rate of 95%, a hit time of 4 ns, and a miss penalty of 100 ns, what is the AMAT?
  - AMAT = Hit time + Miss rate x Miss penalty = 4 + 0.05 x 100 = 9 ns
- If replacing the cache with a 2-way set associative increases the hit rate to 97%, but increases the hit time to 5 ns, what is the new AMAT?
  - AMAT = Hit time + Miss rate x Miss penalty = 5 + 0.03 x 100 = 8 ns

Example: Split and Unified Cache

Consider:

- Previous miss rates
- Miss penalty is 50 cycles
- Hit time is 1 cycle
- 75% of the total memory accesses for instructions and 25% of the total memory accesses for data
- On the unified cache, a load or store hit takes an extra cycle, since there is only one port for instructions and data

Example: Split and Unified Cache

AMAT for the split cache:

- AMAT = 75% x (1 + 0.64% x 50) + 25% (1 + 6.47% x 50) = 2.05

AMAT for the unified cache:

- AMAT = 75% x (1 + 1.99% x 50) + 25% x (2 + 1.99% x 50) = 2.24

A unified cache has a longer AMAT, with a lower miss rate, due to conflicts for instruction and data hazards

Summary of Performance Equations

$$2^{\text{index}} = \frac{\text{Cache size}}{\text{Block size} \times \text{Set associativity}}$$

$$\text{CPU execution time} = (\text{CPU clock cycles} + \text{Memory stall cycles}) \times \text{Clock cycle time}$$

$$\text{Memory stall cycles} = \text{Number of misses} \times \text{Miss penalty}$$

$$\text{Memory stall cycles} = \text{IC} \times \frac{\text{Misses}}{\text{Instruction}} \times \text{Miss penalty}$$

$$\frac{\text{Misses}}{\text{Instruction}} = \text{Miss rate} \times \frac{\text{Memory accesses}}{\text{Instruction}}$$

$$\text{Average memory access time} = \text{Hit time} + \text{Miss rate} \times \text{Miss penalty}$$

$$\text{CPU execution time} = \text{IC} \times \left(\text{CPI}_{\text{execution}} + \frac{\text{Memory stall clock cycles}}{\text{Instruction}} \right) \times \text{Clock cycle time}$$

$$\text{CPU execution time} = \text{IC} \times \left(\text{CPI}_{\text{execution}} + \frac{\text{Misses}}{\text{Instruction}} \times \text{Miss penalty} \right) \times \text{Clock cycle time}$$

$$\text{CPU execution time} = \text{IC} \times \left(\text{CPI}_{\text{execution}} + \text{Miss rate} \times \frac{\text{Memory accesses}}{\text{Instruction}} \times \text{Miss penalty} \right) \times \text{Clock cycle time}$$

$$\frac{\text{Memory stall cycles}}{\text{Instruction}} = \frac{\text{Misses}}{\text{Instruction}} \times (\text{Total miss latency} - \text{Overlapped miss latency})$$

$$\text{Average memory access time} = \text{Hit time}_{\text{L1}} + \text{Miss rate}_{\text{L1}} \times (\text{Hit time}_{\text{L2}} + \text{Miss rate}_{\text{L2}} \times \text{Miss penalty}_{\text{L2}})$$

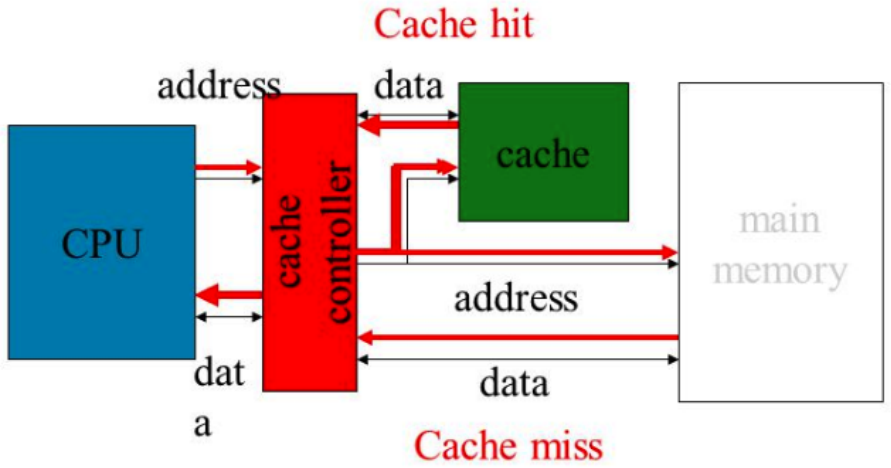
$$\frac{\text{Memory stall cycles}}{\text{Instruction}} = \frac{\text{Misses}_{\text{L1}}}{\text{Instruction}} \times \text{Hit time}_{\text{L2}} + \frac{\text{Misses}_{\text{L2}}}{\text{Instruction}} \times \text{Miss penalty}_{\text{L2}}$$

Basic Cache Optimization Techniques

Technique	Hit time	Miss penalty	Miss rate	Hardware complexity	Comment
Larger block size		–	+	0	Trivial; Pentium 4 L2 uses 128 bytes
Larger cache size	–		+	1	Widely used, especially for L2 caches
Higher associativity	–		+	1	Widely used
Multilevel caches		+		2	Costly hardware; harder if L1 block size ≠ L2 block size; widely used
Read priority over writes		+		1	Widely used
Avoiding address translation during cache indexing	+			1	Widely used

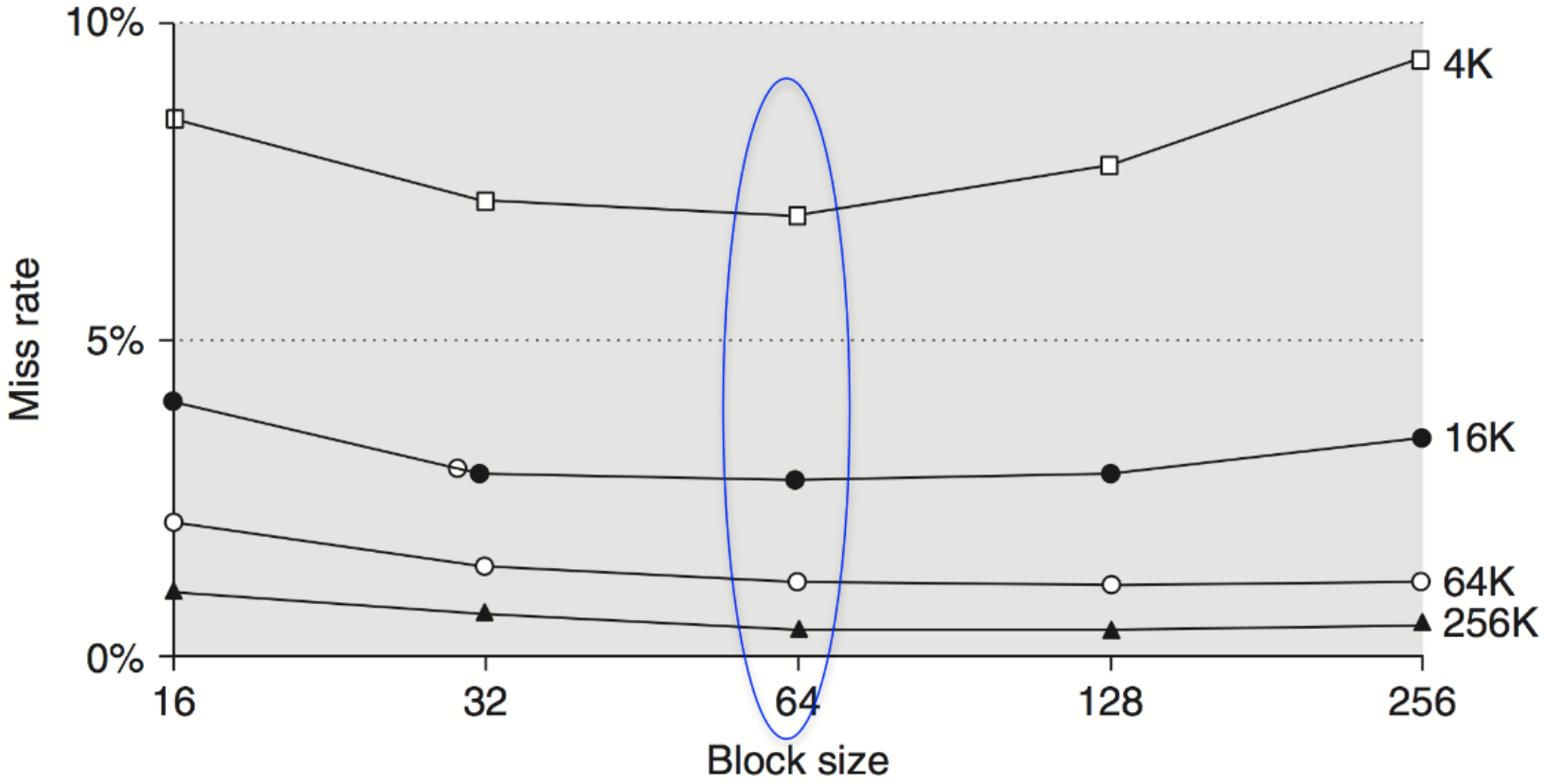
Basic Cache Optimizations

Average Memory Access Time = Hit Time + Miss Rate * Miss Penalty



Goals	Basic Approaches
Reducing Miss Rate	Larger block size, larger cache size and higher associativity
Reducing Miss Penalty	Multilevel caches, and higher read priority over writes
Reducing Hit Time	Avoid address translation when indexing the cache

1. Reduce Miss Rate via a Larger Block Size



1. Reduce Miss Rate via a Larger Block Size

A larger block size can increase the miss penalty

Block size	Miss penalty	Cache size			
		4K	16K	64K	256K
16	82	8.027	4.231	2.673	1.894
32	84	**7.082**	3.411	2.134	1.588
64	88	7.160	**3.323**	**1.933**	**1.449**
128	96	8.469	3.659	1.979	1.470
256	112	11.651	4.685	2.288	1.549

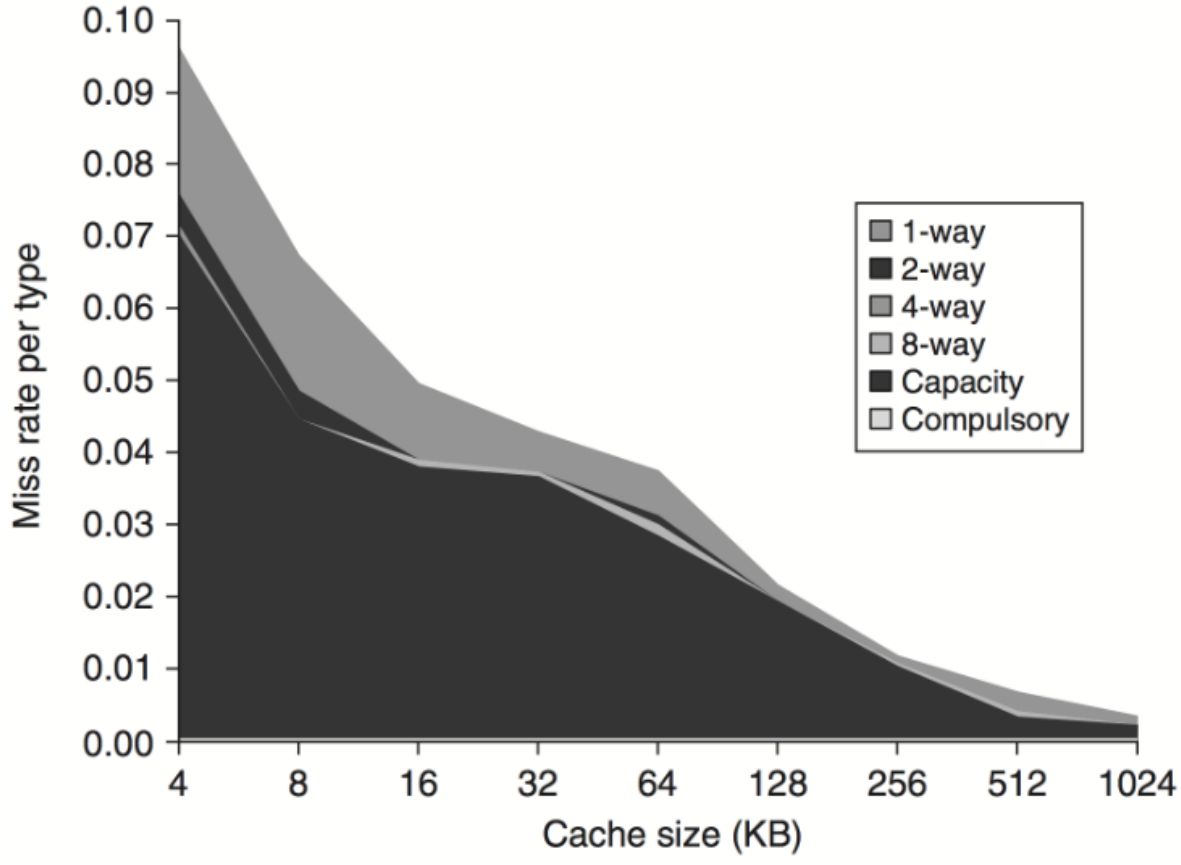
Example

Assume the memory system takes 80 clock cycles of overhead and then delivers 16 bytes every 2 block cycles. Then, it can supply 16 bytes in 82 clock cycles, 32 bytes in 84 clock cycles and so forth. Which block size has the smallest average memory access time for each cache size in the figures on the previous slides?

- If we assume the hit time is 1 clock cycle independent of block size, then the access time for a 16-byte block in a 4 KB cache is:
  - $1 + (8.57\% \times 82) = 8.027$ *clock cycles*
- For a 256-byte block in 256 KB cache the average memory access time is:
  - $1 + (0.49\% \times 112) = 1.549 clockcycles$

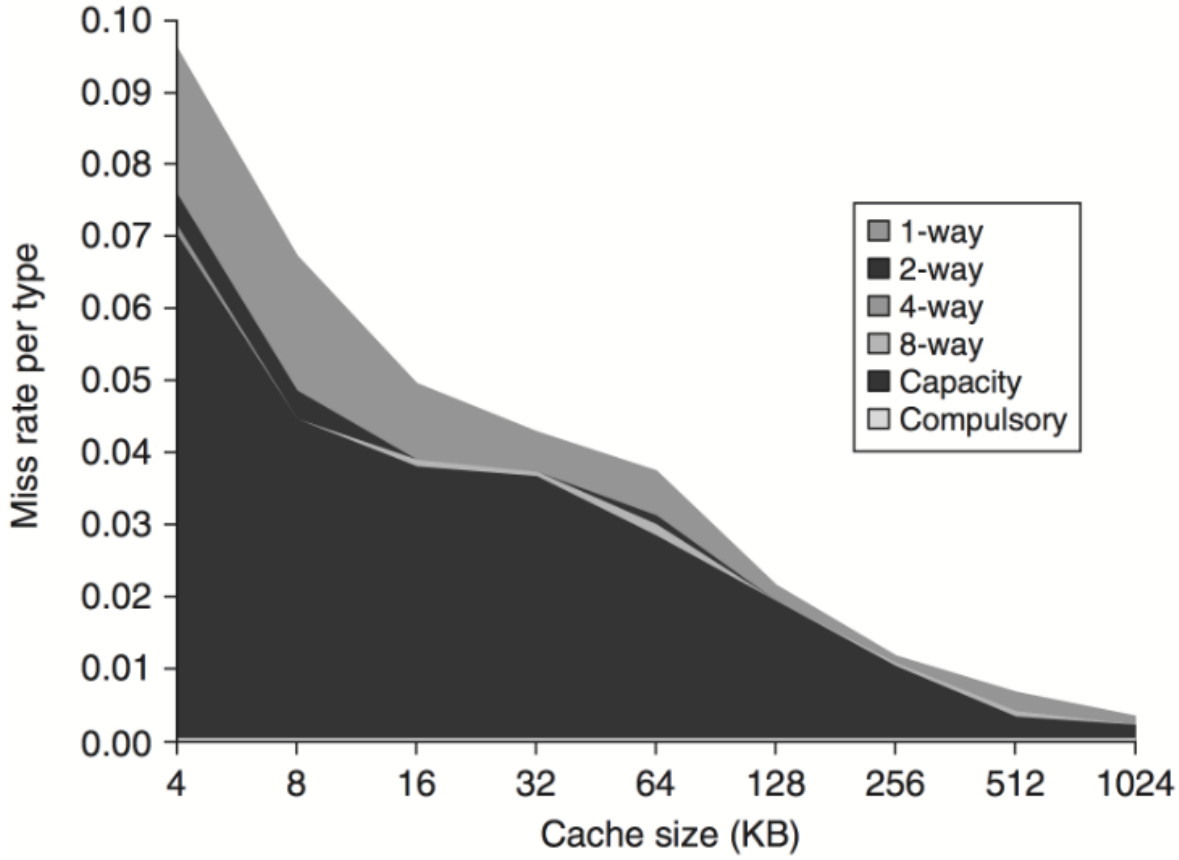
2. Reduce Miss Rate via a Larger Cache

- Capacity misses reduce when the capacity of the cache increases
- The hit time and cost can increase
- Instead use larger cache sizes for L2 and L3 off-chip caches



3. Reduce Misses via Higher Associativity

- 2:1 Rule: Miss Rate DM cache size N = Miss Rate 2-way cache size N/2
- Hit time for higher associative caches can be high and involves complex circuits



Associativity and AMAT

	Associativity			
Cache size (KB)	1-way	2-way	4-way	8-way
4	3.44	3.25	3.22	**3.28**
8	2.69	2.58	2.55	**2.62**

Associativity can lead to Higher Access Time

	Associativity			
Cache size (KB)	1-way	2-way	4-way	8-way
4	3.44	3.25	3.22	**3.28**
8	2.69	2.58	2.55	**2.62**
16	2.23	**2.40**	**2.46**	**2.53**
32	2.06	**2.30**	**2.37**	**2.45**
64	1.92	**2.14**	**2.18**	**2.25**
128	1.52	**1.84**	**1.92**	**2.00**
256	1.32	**1.66**	**1.74**	**1.82**
512	1.20	**1.55**	**1.59**	**1.66**

4. Reduce Miss Penalty via Multilevel Caches

- Techniques:
  - Make the cache faster to keep pace with the speed of CPUs
  - Make the cache larger to overcome the widening gap
- L2 Equations:

$$\text{Average memory access time} = \text{Hit time}_{L1} + \text{Miss rate}_{L1} \times \text{Miss penalty}_{L1}$$

and

$$\text{Miss penalty}_{L1} = \text{Hit time}_{L2} + \text{Miss rate}_{L2} \times \text{Miss penalty}_{L2}$$

so

$$\begin{aligned}\text{Average memory access time} = {} & \text{Hit time}_{L1} + \text{Miss rate}_{L1} \\ & \times (\text{Hit time}_{L2} + \text{Miss rate}_{L2} \times \text{Miss penalty}_{L2})\end{aligned}$$

LC Cache Example

- If a direct mapped cache has a hit rate of 95time of 4 ns, and a miss penalty of 100 ns, what is the AMAT?
  - AMAT = Hit time + Miss rate x Miss penalty = 4 + 0.05 x 100 = 9 ns
- If an L2 cache is added with a hit time of 20 ns and a hit rate of 50%, what is the new AMAT?

$$AMAT = Hit\ Time_{L1} + Miss\ Rate_{L1} x (Hit\ Time_{L2} + Miss\ Rate_{L2} x Miss\ Penalty_{L2}) = 4 + 0.05x(20 + 0.5x100) = 7.5\ ns$$

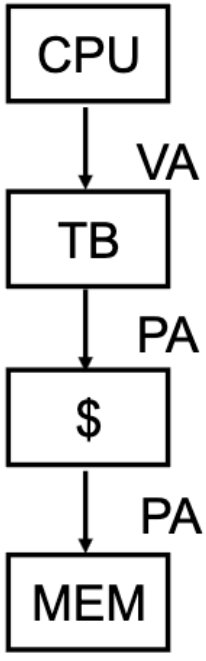
Miss Rate in Multilevel Caches

- **Local miss rate:** misses in this cache divided by the total number of memory accesses to this cache (Miss rateL1 ,Miss rateL2)
- **Global miss rate:** misses in this cache divided by the total number of memory accesses generated by the CPU (Miss rateL1, Miss RateL1 x Miss RateL2)

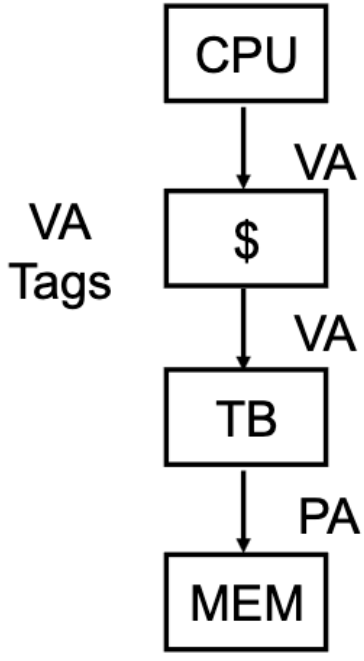
5. Reduce Miss Penalty by Giving Priority to Read Misses over Writes

- Perform any reads before any completion of writes
- Check write buffer contents before read; if no conflicts, let the memory access continue
- Copy any dirty block to a write buffer, do the read, and then do the write

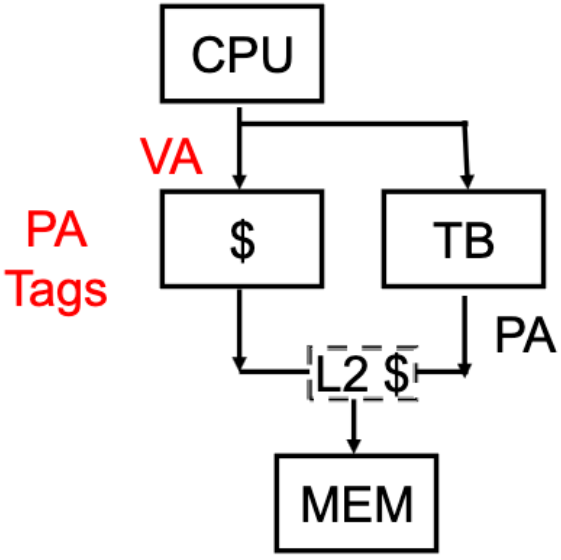
6. Reduce Hit Time by Avoiding Address Translation during Indexing of the Cache



Conventional Organization



Virtually Addressed Cache
Translate only on miss
Synonym Problem



Overlap cache access with VA translation: requires $ index to remain invariant across translation

Challenges of Virtual Cache

- Protection
- Context switching is required via flushing
- Aliases
- I/O use physical address

Advanced Cache Optimizations

- **Reduce the hit time:** Small and simple first-level caches and way- prediction.
- **Increase cache bandwidth:** Pipelined caches, multibanked caches, and nonblocking caches.
- **Reduce the miss penalty:** Critical word first and merging write buffers
- **Reduce the miss rate:** Compiler optimizations
- **Reduce the miss penalty or miss rate via parallelism:** Hardware prefetching and compiler prefetching

Resources

- https://www.info425.ece.mcgill.ca/tutorials/T08-Caches.pdf
- https://passlab.github.io/CSCE513/notes/lecture11_CacheAndPerformance.pdf
- https://passlab.github.io/CSCE513/notes/lecture12_CacheOptimizations.pdf