# Performance of Cache Memory

# Hits and Misses

If the data requested by the processor appears in some block in the upper memory level, this is called a **hit**.

Otherwise, the request is called a **miss** and the next memory level is then accessed to retrieve the block containing the requested data.

The **hit rate** is the fraction of memory accesses found in the upper memory level – often used as a measure of the performance of the memory hierarchy.

The **miss rate (1−hit rate)** is the fraction of memory accesses not found in the upper memory level.

# Hits and Misses

**Hit time** is the time to access the upper memory level, which includes the time needed to determine whether the access is a hit or a miss.

**Miss penalty** is the time to replace a block in the upper memory level with the corresponding block from the next memory level, plus the time to deliver this block to the processor. The time to access the next memory level is the major component of the miss penalty.

# Average Memory Access Time (AMAT)

(AMAT) is the average time to access memory considering both hits and misses.

AMAT = Time for a Hit + Miss Rate * Miss Penalty

(Memory Performance Equation.)

What is the AMAT for a processor with a 200 ps clock, a miss penalty of 50 clock cycles, a miss rate of 0.02 misses per instruction and a cache access time of 1 clock cycle?.

1 + 0.02 * 50 = 2 clock cycles, or 2 * 200 = 400 ps

# Multiple Cache Levels

New AMAT Calculation:

AMAT = L1 Hit Time + L1 Miss Rate * L1 Miss Penalty,

L1 Miss Penalty = L2 Hit Time + L2 Miss Rate * L2 Miss Penalty

and so forth (final miss penalty is Main Memory access time)

# New AMAT Example

1 cycle L1 hit time, 2% L1 miss rate,

5 cycle L2 hit time, 5% L2 miss rate.

100 cycle main memory access time.

Without L2 cache:

AMAT = 1 + .02*100 = 3

With L2 cache:

AMAT = 1 + .02*(5 + .05*100) = 1.2

# Calculating AMAT

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

# Cache Performance

Princeton (Unified) Memory Architecture

- **CPUtime = Instruction count x CPI x Clock cycle time**
- $CPI_{execution}$ = CPI with ideal memory
- CPI = $CPI_{execution}$ + Mem Stall cycles per instruction
- CPUtime = Instruction Count x ($CPI_{execution}$ + Mem Stall cycles per instruction) x Clock cycle time
- Mem Stall cycles per instruction = Mem accesses per instruction x Miss rate x Miss penalty
- **CPUtime = IC x ($CPI_{execution}$ + Mem accesses per instruction x Miss rate x Miss penalty) x Clock cycle time**
- Misses per instruction = Memory accesses per instruction x Miss rate
- CPUtime = IC x ($CPI_{execution}$ + Misses per instruction x Miss penalty) x Clock cycle time

# Cache Impact On Performance: *An Example*

Assuming the following execution and cache parameters:

- Cache miss penalty = 50 cycles
- Normal instruction execution CPI ignoring memory stalls = 2.0 cycles
- Miss rate = 2%
- Average memory references per instruction = 1.33

**CPU time = IC x [CPI $_{execution}$ + ( Memory accesses per instruction x Miss rate x Miss penalty ) ] x Clock cycle time**

CPU time $_{with\ cache}$ = IC x (2.0 + (1.33 x 2% x 50)) x clock cycle time

= IC x 3.33 x Clock cycle time

→ *Lower CPI $_{execution}$ increases the impact of cache miss clock cycles*

→ *Calculate the effect on CPI rather than the average memory access time.*

# Cache Performance Example

- Suppose a CPU executes at Clock Rate = 200 MHz (5 ns per cycle) with a single level of cache. $CPI_{execution} = 1.1$
- Instruction mix: 50% arith/logic, 30% load/store, 20% control
- Assume a cache miss rate of 1.5% and a miss penalty of 50 cycles.

CPI = $CPI_{execution}$ + mem stalls per instruction

**Mem Stalls per instruction = Mem accesses per instruction x Miss rate x Miss penalty.**

**For computer that always hits, CP I = 1**

Mem accesses per instruction = 1 + .3 = 1.3

**Instruction fetch** **Load/store**

Mem Stalls per instruction = 1.3 x .015 x 50 = 0.975

CPI = 1.1 + .975 = 2.075

The ideal memory CPU with no misses is 2.075/1.1 = 1.88 times faster

# Cache Performance Example

Given

Instruction-cache miss rate = 2%

- Data-cache miss rate = 4%
- Miss penalty = 100 cycles
- Base CPI (with ideal cache performance) = 2
- Load & stores are 36% of instructions.

Miss cycles per instruction:

Instruction-cache: $0.02 \times 100 = 2$

Data-cache: $0.36 \times 0.04 \times 100 = 1.44$

Actual CPI = 2 + 2 + 1.44 = 5.44

Ideal CPU is 5.44/2 =2.72 times faster.

We spend 3.44/5.44 = 63% of our execution time on memory stalls!

# Types of Cache Misses

- Compulsory (cold) misses: First access to a block. Compulsory misses occur even for infinite size cache

  Could be reduced by prefetching blocks before they are demanded.

- Capacity misses: A cache cannot contain all blocks needed in a program. some blocks are discarded then later accessed. Capacity misses occur in a fully-associative cache.
- Could be reduced with insertion/replacement policies and dead block prediction

# Types of Cache Misses

- Conflict misses: Blocks mapping to the same set may be discarded (in direct-mapped and set-associative caches).

- Could be reduced by increasing associativity or better replacement/insertion policies

# Methods for improving performance

- increase (change) cache size
- increase (change) block size
- increase (change) associativity
- add a 2nd level cache