Computer Architecture – tutorial 4

# Context, Objectives and Organization

This worksheet covers the material from lectures on Caches. The goal of the quantitative exercises in this tutorial is to familiarize you with quantitative analysis of caches (E1) and to investigate the tradeoffs between write-through and write-back caches (E2 and E3).

# E1: individual – 10 min

*Problem*

Assume we have a computer where the CPI is 1.0 when all memory accesses (including data and instruction accesses) hit in the cache. The cache is a unified (data + instruction) cache of size 256 KB, 4-way set associative, with a block size of 64 bytes. The data accesses (loads and stores) constitute 50% of the instructions. The unified cache has a miss penalty of 25 clock cycles and a miss rate of 2%. Assume 32 bit instruction and data addresses.

1. **What is the tag size for the cache?**

The 32-bit address is divided into; 6 bits (from 64B block), Index 12 bits (from 4096 set) and Tag 14 bits (32-12-6). Therefore, Answer = 14-bit tag per cache block.

1. How much faster would the computer be if all memory accesses were cache hits?

Current CPI (with 2% miss rate):

* Total memory accesses = 1 (instruction) + 0.5 (data) = 1.5 per instruction
* Misses = 1.5 × 2% = 0.03 misses/instruction
* Penalty = 0.03 × 25 cycles = 0.75
* Actual CPI = 1.0 (base) + 0.75 = 1.75

Speedup = CPI (no misses) / CPI (with misses) = 1.0 / 1.75 ≈ 1.75× slower now

Answer: The computer would be 1.75× faster with all cache hits

# E2: groups of 2 – 15 min

*Problem*

You purchased an Acme computer with the following features:

* + 95% of all memory accesses are found in the cache.
  + Each cache block is two words, and the whole block is read on any miss.
  + The processor sends references to its cache at the rate of 109 words per second.
  + 25% of those references are writes.
  + Assume that the memory system can support 109 words per second, reads or writes.
  + The bus reads or writes a single word at a time (the memory system cannot read or write two words at once).
  + Assume at any one time, 30% of the blocks in the cache have been modified.
  + The cache uses write allocate on a write miss.

1. Write-Through Cache

* Reads: 75% × 1e9 = 7.5e8/sec
* Read Misses: 5% × 7.5e8 = 3.75e7 → each read miss fetches 2 words = 7.5e7 word transfers
* Writes: All 2.5e8 writes go to memory = 2.5e8 transfers
* Write Misses: 5% × 2.5e8 = 1.25e7 → cause block fetch of 2 words = 2.5e7 transfers
* Total: 7.5e7 + 2.5e8 + 2.5e7 = 3.5e8 words/sec

Utilization = 3.5e8 / 1e9 = 35%

Write-through uses 35% of memory bandwidth

1. Write-Back Cache

* Read Misses: Same as above = 7.5e7
* Write Misses: 1.25e7 → cause 2.5e7 (block loads)
* Evictions: 30% of replaced blocks are dirty = assume write-back per block = 1 write per 2.5 evictions  
  → approx. 30% of 5% of accesses = 0.3 × 0.05 × 1e9 = 1.5e7 word writes
* Total: 7.5e7 + 2.5e7 + 1.5e7 = 1.15e8 words per second.

Utilization = 1.15e8 / 1e9 = 11.5%

Answer: Write-back uses 11.5% of memory bandwidth

# E3: groups of 2 – 15 min

*Problem*

One difference between a write-through cache and a write-back cache can be in the time it takes to write. During the first cycle, we detect whether a hit will occur, and during the second (assuming a hit) we actually write the data. Let’s assume that 50% of the blocks are dirty for a write-back cache. For this question, assume that the write buffer for the write through will never stall the CPU (no penalty). Assume a cache read hit takes 1 clock cycle, the cache miss penalty is 50 clock cycles, and a block write from the cache to main memory takes 50 clock cycles. Finally, assume the instruction cache miss rate is 0.5% and the data cache miss rate is 1%. Assuming that on average 26% and 9% of instructions in the workload are loads and stores, respectively, estimate the performance of a write-through cache with a two-cycle write versus a write-back cache with a two-cycle write.

**Solution:**

**Write-Through Cache: CPI Estimate**

Instruction miss penalty = 0.005 × 50 = **0.25**

Load miss penalty = 0.26 × 0.01 × 50 = **0.13**

Store = 0.09 × 2 = **0.18**

**Total CPI (write-through)** ≈ 1.0 (base) + 0.25 + 0.13 + 0.18 = **1.56**

**Write-Back Cache: CPI Estimate**

Instruction miss = same = 0.25

Load miss = 0.13

Store **hit** = 0.09 × 2 = 0.18

**Evicted dirty write-backs**: 50% × 0.01 × (0.26 + 0.09) × 50  
= 0.5 × 0.01 × 0.35 × 50 = **0.0875**

**Total CPI (write-back)** ≈ 1.0 + 0.25 + 0.13 + 0.18 + 0.0875 = **1.65**

**Answer**:

Write-through CPI = **1.56**

Write-back CPI = **1.65**

Write-through is slightly better **in this case**, due to zero write stall from buffer and added dirty write cost in write-back.