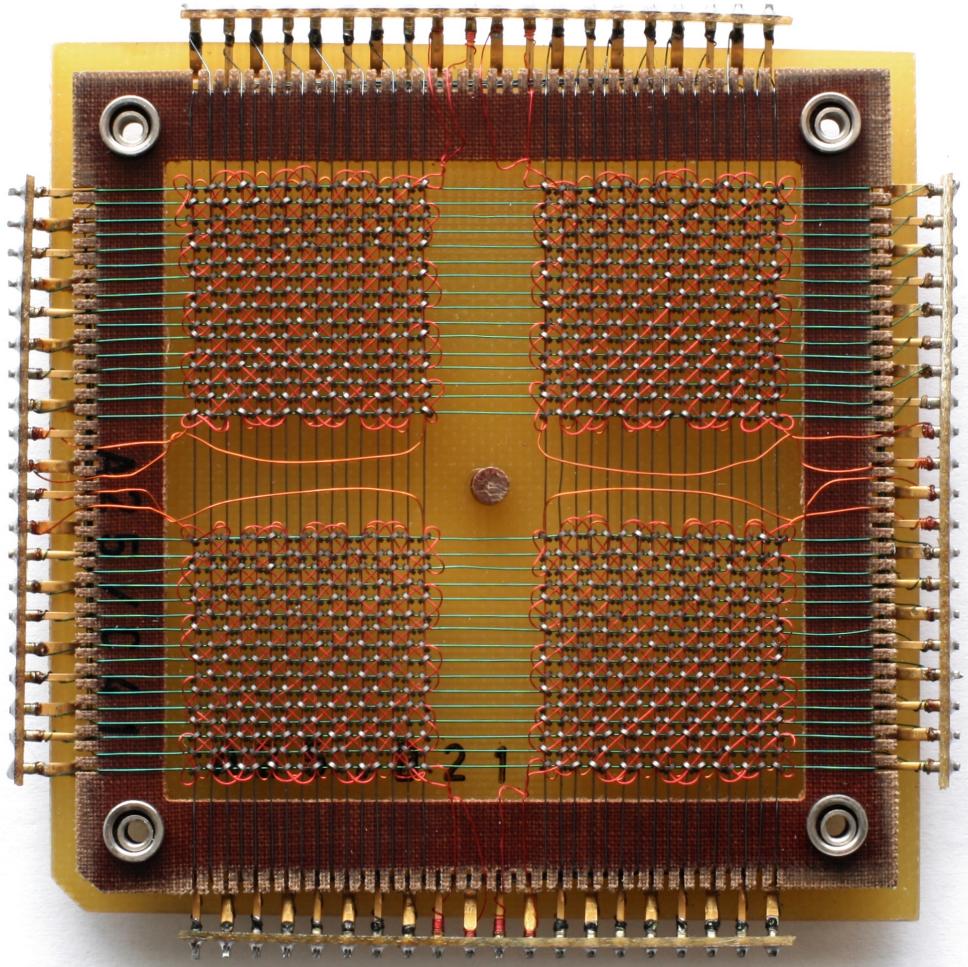
CSCI 210: Computer Architecture
Lecture 35: Caches IV

Stephen Checkoway

Oberlin College

CS History: Magnetic-core Memory

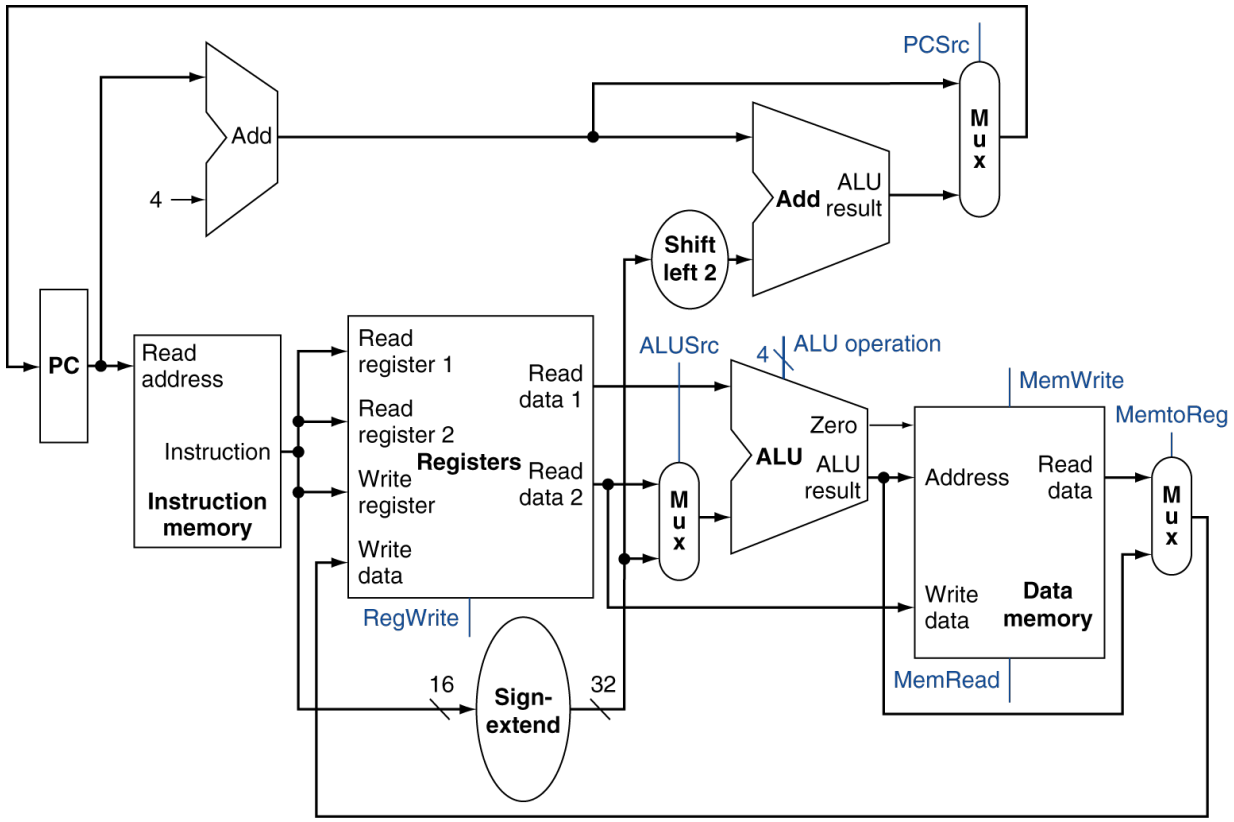


A 32 x 32 core memory plane storing 1024 bits. Photo: Konstantin Lanzet, CC BY-SA 3.0

- Predominant form of memory between 1955 and 1975
- Rings (cores) of hard magnetic material with wires woven through them
- Each core can be magnetized either clockwise or counter-clockwise
- Represents a 0 or 1 based on direction
- Process of reading the cores erases the values

CACHE PERFORMANCE

I-cache vs D-cache



- Separate caches for instruction memory and data memory
- I-cache: instruction cache
- D-cache: data cache

Measuring Cache Performance

- Components of CPU time
  - Program execution cycles
    - Includes cache hit time
  - Memory stall cycles
    - Mainly from cache misses
- With simplifying assumptions:

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

$$= \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Misses}}{\text{Instruction}} \times \text{Miss penalty}$$

- Miss penalty is the number of cycles the pipeline stalls on a cache miss waiting for the data to arrive from memory (or from the next level of cache)

Cache Miss Cycles Per Instruction

Given

- I-cache miss rate = 2%
- D-cache miss rate = 4%
- Miss penalty = 100 cycles
- Load & stores are 36% of instructions

	I-cache	D-cache
A	.02 * 100	.04 * 100
B	.02	.04
C	.02 * .36 * 100	.04 * .36 * 100
D	.02 * 100	.04 * .36 * 100

Cache Performance Example

- Given
  - I-cache miss rate = 2%
  - D-cache miss rate = 4%
  - Miss penalty = 100 cycles
  - Base CPI (ideal cache) = 2
  - Load & stores are 36% of instructions
- Miss cycles per instruction
  - I-cache: $0.02 \times 100 = 2$
  - D-cache: $0.36 \times 0.04 \times 100 = 1.44$
- Actual CPI = 2 + 2 + 1.44 = 5.44

Average Memory Access Time

- Hit time is also important for performance
- Average memory access time (AMAT)
  - AMAT = Hit time + Miss rate × Miss penalty
- Example
  - hit time = 1 cycle, miss penalty = 20 cycles, I-cache miss rate = 5%
  - AMAT =

AMAT for multi-level cache hierarchies

- H_i = Hit time for level i of the cache hierarchy
- MR_i = Miss rate for level i
- AMP_i = Average miss penalty for level i

$AMAT = H_1 + MR_1 * AMP_1$

$AMP_i = H_{i+1} + MR_{i+1} * AMP_{i+1}$

Cache Speed Factors

- Memory lookup time
- Hit rate
- Size
- Frequency of collisions

How Much Associativity

- Increased associativity decreases miss rate
  - But with diminishing returns
- Simulation of a system with 64 kB D-cache, 64-byte blocks
  Miss rate:
  - 1-way: 10.3%
  - 2-way: 8.6%
  - 4-way: 8.3%
  - 8-way: 8.1%

```
For (int i = 0; i < 10000000; i++)
    sum += A[i];
```

Assume each element of A is 4 bytes and sum is kept in a register. Assume a direct-mapped 32 kB cache with 32 byte blocks. Which changes would help the hit rate of the above code?

Selection	Change
A	Increase to 2-way set associativity
B	Increase block size to 64 bytes
C	Increase cache size to 64 kB
D	A and C combined
E	A, B, and C combined

Performance Summary

- When CPU performance increases
  - The miss penalty becomes more significant
- When we decrease the base CPI
  - A greater proportion of time spent on memory stalls
- When we increase the clock rate
  - Memory stalls account for more CPU cycles
- Can’t neglect cache behavior when evaluating system performance

Reading

- Next lecture: More Caches!