Lecture 21: Memory Hierarchy

• Today’s topics:
  - Cache organization
  - Cache hits/misses
Cache Hierarchies

• Data and instructions are stored on DRAM chips – DRAM is a technology that has high bit density, but relatively poor latency – an access to data in memory can take as many as 300 cycles today!

• Hence, some data is stored on the processor in a structure called the cache – caches employ SRAM technology, which is faster, but has lower bit density

• Internet browsers also cache web pages – same concept
Memory Hierarchy

- As you go further, capacity and latency increase

Registers
- 1KB
- 1 cycle

L1 data or instruction Cache
- 32KB
- 2 cycles

L2 cache
- 2MB
- 15 cycles

Memory
- 1GB
- 300 cycles

Disk
- 80 GB
- 10M cycles
Locality

• Why do caches work?
  ▪ Temporal locality: if you used some data recently, you will likely use it again
  ▪ Spatial locality: if you used some data recently, you will likely access its neighbors

• No hierarchy: average access time for data = 300 cycles

• 32KB 1-cycle L1 cache that has a hit rate of 95%:
  average access time = 0.95 x 1 + 0.05 x (301)
  = 16 cycles
Direct-mapped cache: each address maps to a unique cache location.

8-byte words

Sets

Data array

Offset

Byte address

101000

8 words: 3 index bits
The Tag Array

Direct-mapped cache: each address maps to a unique address

Tag array

Data array

8-byte words

Byte address

Tag

Compare

101000
Example Access Pattern

Direct-mapped cache: each address maps to a unique address

Assume that addresses are 8 bits long
How many of the following address requests are hits/misses?
4, 7, 10, 13, 16, 68, 73, 78, 83, 88, 4, 7, 10…

Tag array

Data array

8-byte words

Compare

Tag

101000

Byte address
Increasing Line Size

A large cache line size $\Rightarrow$ smaller tag array, fewer misses because of spatial locality

32-byte cache line size or block size
Set associativity → fewer conflicts; wasted power because multiple data and tags are read.
Associativity

How many offset/index/tag bits if the cache has 64 sets, each set has 64 bytes, 4 ways

Tag array → Tag array

Byte address

10100000

Compare

Way-1

Way-2

Data array
Example

• 32 KB 4-way set-associative data cache array with 32 byte line sizes

• How many sets?

• How many index bits, offset bits, tag bits?

• How large is the tag array?
Cache Misses

- On a write miss, you may either choose to bring the block into the cache (write-allocate) or not (write-no-allocate).

- On a read miss, you always bring the block in (spatial and temporal locality) – but which block do you replace?
  - no choice for a direct-mapped cache
  - randomly pick one of the ways to replace
  - replace the way that was least-recently used (LRU)
  - FIFO replacement (round-robin)
Writes

• When you write into a block, do you also update the copy in L2?
  ➢ write-through: every write to L1 → write to L2
  ➢ write-back: mark the block as dirty, when the block gets replaced from L1, write it to L2

• Writeback coalesces multiple writes to an L1 block into one L2 write

• Writethrough simplifies coherency protocols in a multiprocessor system as the L2 always has a current copy of data
Types of Cache Misses

- Compulsory misses: happens the first time a memory word is accessed – the misses for an infinite cache
- Capacity misses: happens because the program touched many other words before re-touching the same word – the misses for a fully-associative cache
- Conflict misses: happens because two words map to the same location in the cache – the misses generated while moving from a fully-associative to a direct-mapped cache
Title

• Bullet