Previous Lecture: practical and performance considerations

- Control bits
Previous Lecture: practical and performance considerations

- Control bits
  - Valid & Dirty bits
Previous Lecture: practical and performance considerations

- Control bits
  - Valid & Dirty bits
- Locality tradeoffs and compromises
Previous Lecture: practical and performance considerations

- **Control bits**
  - Valid & Dirty bits
- **Locality tradeoffs and compromises**
  - Impact of cache line size
Previous Lecture: practical and performance considerations

- Control bits
  - Valid & Dirty bits
- Locality tradeoffs and compromises
  - Impact of cache line size
  - Spatial vs. temporal locality
Previous Lecture: practical and performance considerations

- **Control bits**
  - Valid & Dirty bits
- **Locality tradeoffs and compromises**
  - Impact of cache line size
  - Spatial vs. temporal locality
  - Separating Instruction & Data caches
Previous Lecture: practical and performance considerations

- **Control bits**
  - Valid & Dirty bits

- **Locality tradeoffs and compromises**
  - Impact of cache line size
  - Spatial vs. temporal locality
  - Separating Instruction & Data caches

- **Multiple-level caches**
Previous Lecture: practical and performance considerations

- **Control bits**
  - Valid & Dirty bits

- **Locality tradeoffs and compromises**
  - Impact of cache line size
  - Spatial vs. temporal locality
  - Separating Instruction & Data caches

- **Multiple-level caches**
  - Why, how
Previous Lecture: practical and performance considerations

- Control bits
  - Valid & Dirty bits

- Locality tradeoffs and compromises
  - Impact of cache line size
  - Spatial vs. temporal locality
  - Separating Instruction & Data caches

- Multiple-level caches
  - Why, how
  - Performance model in a cache hierarchy
Previous Lecture: practical and performance considerations

- Control bits
  - Valid & Dirty bits
- Locality tradeoffs and compromises
  - Impact of cache line size
  - Spatial vs. temporal locality
  - Separating Instruction & Data caches
- Multiple-level caches
  - Why, how
  - Performance model in a cache hierarchy
- First lab
Today’s Lecture – Learning Objectives

▶ “three.pnum x C’s” model of cache performance
▶ Time penalties for starting with empty cache
▶ Systems interconnect issues with caching and solutions!
▶ Caching and Virtual Memory
Today’s Lecture – Learning Objectives

▶ “3 x C’s” model of cache performance
Today’s Lecture – Learning Objectives

▶ “3 x C’s” model of cache performance

▶ Time penalties for starting with empty cache
Today’s Lecture – Learning Objectives

▶ “3 x C’s” model of cache performance

▶ Time penalties for starting with empty cache

▶ Systems interconnect issues with caching and solutions!
Today’s Lecture – Learning Objectives

- “3 x C’s” model of cache performance
- Time penalties for starting with empty cache
- Systems interconnect issues with caching and solutions!
- Caching and Virtual Memory
Describing Cache Misses

- Compulsory Misses
  - Cold start

Even with full associativity, the cache cannot contain all the blocks of the program.
Describing Cache Misses

- **Compulsory Misses**
  - Cold start

- **Capacity Misses**
  - Even with full associativity, cache cannot contain all the blocks of the program
Describing Cache Misses

- **Compulsory Misses**
  - Cold start

- **Capacity Misses**
  - Even with full associativity, cache cannot contain all the blocks of the program

- **Conflict Misses**
  - Multiple blocks compete for the same set.
Describing Cache Misses

- Compulsory Misses
  - Cold start

- Capacity Misses
  - Even with full associativity, cache cannot contain all the blocks of the program

- Conflict Misses
  - Multiple blocks compete for the same set.
  - This would not happen in fully associative cache
Cache Performance

Today’s caches, how long does it take

▶ To fill L3 cache? (8MB)
▶ To fill L2 cache? (256KB)
▶ To fill L1 D cache? (32KB)
Cache Performance

Today’s caches, how long does it take

- To fill L3 cache? (8MB)
- To fill L2 cache? (256KB)
- To fill L1 D cache? (32KB)

- Number of lines = (cache size) / (line size)
- Number of lines = 32K/64 = 512
- 512 x memory access times at 20nS = 10 µS
- 20,000 clock cycles at 2GHz
Caches in Systems

- CPU
- Registers
- L1 Inst Cache
- L1 Data Cache
- L2
- RAM Memory
- I/O
- Interconnect

E.g., network, disk, ...

- Typical I/O bandwidth?
Caches in Systems

- CPU
- Registers
- L1 Inst Cache
- L1 Data Cache
- L2
- RAM Memory
- I/O
- Interconnect

On-chip

E.g., network, disk, ...

- Typical I/O bandwidth?
- What could go wrong?
Cache Consistency Problem (1)

Problem

I/O writes to memory
Cache data is no longer up-to-date
Cache Consistency Problem (1)

Problem

- I/O writes to memory
- Cache data is no longer up-to-date
Cache Consistency Problem (2)

Problem

I/O reads from memory
But the cache has a new, updated value
Cache Consistency Problem (2)

- Problem
  - I/O reads from memory
  - But the cache has a new, updated value
O/S knows where I/O takes place in memory
  - Mark I/O areas as non-cacheable (how?)
Cache Consistency Software Solutions

- O/S knows where I/O takes place in memory
  - Mark I/O areas as non-cacheable (how?)

- O/S knows when I/O starts and finishes
  - Clear caches before & after I/O?
Hardware Solution 1

CPU

Registers

L1 Inst Cache

L1 Data Cache

L2

RAM Memory

I/O

On-chip

fetch

data

Disadvantage

Slows down the cache
"Pollutes" the cache (replaces potentially useful data)
Hardware Solution 1

- Disadvantage
  - Slows down the cache
Disadvantages:
- Slows down the cache
- "Pollutes" the cache (replaces potentially useful data)
Hardware Solution 2 (Snooping)

- **Snoop logic in the cache**

Diagram showing:
- CPU
- Registers
- L1 Cache
- L2 Cache
- RAM Memory
- I/O
- On-chip connections between components.
Hardware Solution 2 (Snooping)

- **CPU**
  - fetch
  - data
  - Registers

- **L1**
  - Inst Cache

- **L1**
  - Data Cache

- **L2**
  - 5

- **RAM Memory**
  - 5

- **On-chip**

- **I/O**

- Snoop logic in the cache
  - Observes every memory cycle
Hardware Solution 2 (Snooping)

- Snoop logic in the cache
  - Observes every memory cycle
  - Scalability issues
Caches and Virtual Addresses

- CPU addresses – virtual

Recap...

Use Translation-Lookaside Buffer (TLB) to translate V-to-P

What addresses in cache?
Caches and Virtual Addresses

- CPU addresses – virtual
- Memory addresses – physical
Caches and Virtual Addresses

- CPU addresses – virtual
- Memory addresses – physical

- Recap...
Caches and Virtual Addresses

- CPU addresses – virtual
- Memory addresses – physical

Recap...
- Use Translation-Lookaside Buffer (TLB) to translate V-to-P
Caches and Virtual Addresses

- CPU addresses – virtual
- Memory addresses – physical

Recap...
- Use Translation-Lookaside Buffer (TLB) to translate V-to-P

- What addresses in cache?
Option 1: Cache by Physical Addresses

- **CPU**
  - Registers
  - TLB
  - On-chip
  - RAM Memory

- **Address**
- **Data**

- **Slow**
Option 1: Cache by Physical Addresses

CPU

Registers

TLB

On-chip

RAM

Memory

Address translation is in series with cache
Option 2: Cache by Virtual Addresses

- More functional difficulties
Option 2: Cache by Virtual Addresses

- More functional difficulties
  - Snooping
Option 2: Cache by Virtual Addresses

- More functional difficulties
  - Snooping
  - Aliasing
Option 3 : Translate in Parallel with Cache Lookup

- Translation only affects high-order bits of address
- Address within page remains unchanged
Option 3: Translate in Parallel with Cache Lookup

- Translation only affects high-order bits of address
- Address within page remains unchanged
- Low-order bits of Physical Address = low-order bits of Virtual Address
Option 3: Translate in Parallel with Cache Lookup

- Translation only affects high-order bits of address
- Address within page remains unchanged

- Low-order bits of Physical Address = low-order bits of Virtual Address

- Select “index” field of cache address from within low-order bits
Option 3: Translate in Parallel with Cache Lookup

- Translation only affects high-order bits of address
- Address within page remains unchanged
- Low-order bits of Physical Address = low-order bits of Virtual Address
- Select “index” field of cache address from within low-order bits
- Only “Tag” bits changed by translation
Option 3 in operation

Virtual Page address -> Index -> Within line offset -> Virtual address

TLB

Physical address (High-order bits) -> Compare -> hit / miss

Tag
Data line

Multiplexer

data
Summary

▶ “3 x C’s” model of cache performance
Summary

- “3 x C’s” model of cache performance
- Systems interconnect issues with caching and solutions!
  - Non-cacheable areas
  - Cache flushing
  - Snooping
Summary

- “3 x C’s” model of cache performance

- Systems interconnect issues with caching and solutions!
  - Non-cacheable areas
  - Cache flushing
  - Snooping

- Caching and Virtual Memory
  - Physical to virtual conversion (TLB)
  - Cache architectures to support P-to-V conversion
Why are caches essential?
Caches Module Summary

- **Why are caches essential?**
  - Speed imbalance CPU vs. RAM
Caches Module Summary

► Why are caches essential?
  ► Speed imbalance CPU vs. RAM
  ► Performance
Caches Module Summary

- Why are caches essential?
  - Speed imbalance CPU vs. RAM
  - Performance
- How do caches work?
Caches Module Summary

- **Why are caches essential?**
  - Speed imbalance CPU vs. RAM
  - **Performance**
- **How do caches work?**
  - Locality
Caches Module Summary

- **Why are caches essential?**
  - Speed imbalance CPU vs. RAM
  - **Performance**

- **How do caches work?**
  - Locality
  - Associativity
Caches Module Summary

- **Why are caches essential?**
  - Speed imbalance CPU vs. RAM
  - **Performance**

- **How do caches work?**
  - Locality
  - Associativity
  - Replacement policy
Caches Module Summary

- **Why are caches essential?**
  - Speed imbalance CPU vs. RAM
  - **Performance**

- **How do caches work?**
  - Locality
  - Associativity
  - Replacement policy
  - Line size
Caches Module Summary

- **Why are caches essential?**
  - Speed imbalance CPU vs. RAM
  - **Performance**

- **How do caches work?**
  - Locality
  - Associativity
  - Replacement policy
  - Line size
  - Cache lookup: how is data found (address splitting in tag, index, word ID, alignment)
Caches Module Summary

- Why are caches essential?
  - Speed imbalance CPU vs. RAM
  - Performance

- How do caches work?
  - Locality
  - Associativity
  - Replacement policy
  - Line size
  - Cache lookup: how is data found (address splitting in tag, index, word ID, alignment)
  - Cache hierarchy
Caches Module Summary

- **Why are caches essential?**
  - Speed imbalance CPU vs. RAM
  - **Performance**

- **How do caches work?**
  - Locality
  - Associativity
  - Replacement policy
  - Line size
  - Cache lookup: how is data found (address splitting in tag, index, word ID, alignment)
  - Cache hierarchy
  - **Cache misses (3 Cs)**
Caches Module Summary

- Why are caches essential?
  - Speed imbalance CPU vs. RAM
  - Performance

- How do caches work?
  - Locality
  - Associativity
  - Replacement policy
  - Line size
  - Cache lookup: how is data found (address splitting in tag, index, word ID, alignment)
  - Cache hierarchy
  - Cache misses (3 Cs)

- What is the impact of caches?
Caches Module Summary

- Why are caches essential?
  - Speed imbalance CPU vs. RAM
  - Performance

- How do caches work?
  - Locality
  - Associativity
  - Replacement policy
  - Line size
  - Cache lookup: how is data found (address splitting in tag, index, word ID, alignment)
  - Cache hierarchy
  - Cache misses (3 Cs)

- What is the impact of caches?
  - Performance model (including in a cache hierarchy)
Caches Module Summary

► Why are caches essential?
  ► Speed imbalance CPU vs. RAM
  ► Performance

► How do caches work?
  ► Locality
  ► Associativity
  ► Replacement policy
  ► Line size
  ► Cache lookup: how is data found (address splitting in tag, index, word ID, alignment)
  ► Cache hierarchy
  ► Cache misses (3 Cs)

► What is the impact of caches?
  ► Performance model (including in a cache hierarchy)
  ► Interaction with other system components
Caches Module Summary

► Why are caches essential?
  ► Speed imbalance CPU vs. RAM
  ► **Performance**

► How do caches work?
  ► Locality
  ► Associativity
  ► Replacement policy
  ► Line size
  ► Cache lookup: how is data found (address splitting in tag, index, word ID, alignment)
  ► Cache hierarchy
  ► Cache misses (3 Cs)

► What is the impact of caches?
  ► Performance model (including in a cache hierarchy)
  ► Interaction with other system components

► **Does this matter to you?**