

ECE 588/688

Advanced Computer

Architecture II

Instructor: Alaa Alameldeen
alaa@ece.pdx.edu

Fall 2009

Portland State University

When and Where?

- When: Monday & Wednesday 7:00-8:50 PM
- Where: CAP 1315
- Office hours: By appointment
- Webpage: <http://www.cecs.pdx.edu/~alaa/ece588/>
- Go to webpage for:
 - ◆ Class Slides
 - ◆ Papers
 - ◆ Simulator information
 - ◆ Homework and project assignments

Course Description

- Parallel computing and multiprocessors
 - ◆ Symmetric Multiprocessors (SMPs)
 - ◆ Chip Multiprocessors (CMPs), aka multicore processors
 - ◆ Multithreading and parallel programming models
 - ◆ Multiprocessor memory systems
- Emphasis on papers readings NOT on a textbook
 - ◆ Tutorial papers
 - ◆ Original sources and ideas papers
 - ◆ Papers covering most recent trends

Expected Background

- ECE 587/687 or equivalent
 - ◆ Superscalar processor microarchitecture
 - ◆ Branch prediction
 - ◆ Cache organization
 - ◆ Memory ordering
 - ◆ Speculative execution
 - ◆ Multithreading
- Programming experience in “C”

Grading Policy

- Class Participation 10%
- Homeworks (including paper reviews) 20%
- Project 30%
- Midterm Exam 20%
- Final Exam 20%
- Grading Scale:
 - ◆ A: 92-100%
 - ◆ A-: 86-91.5%
 - ◆ B+: 80-85.5%
 - ◆ B: 76-79.5%
 - ◆ B-: 72-75.5%
 - ◆ C+: 68-71.5%
 - ◆ C: 64-67.5%
 - ◆ C-: 60-63.5%
 - ◆ D+: 57-59.5%
 - ◆ D: 54-56.5%
 - ◆ D-: 50-53.5%
 - ◆ F: Below 50%

Why Study Computer Architecture

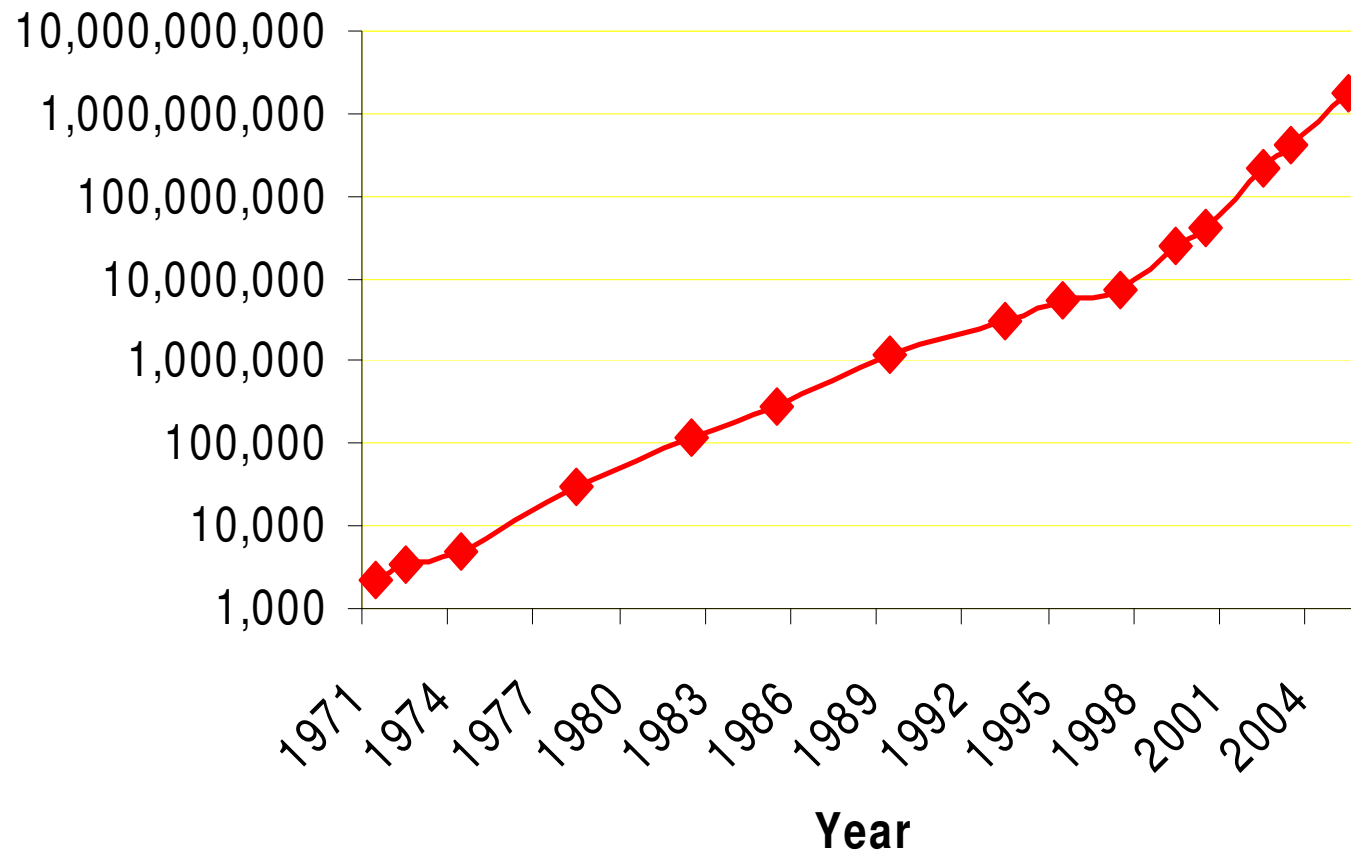
- Technology advancements require continuous optimization of cost, performance, and power
 - ◆ Moore's law
 - Original version: Transistor scaling exponential
 - Popular version: Processor performance exponentially increasing
- Innovation needed to satisfy market trends
 - ◆ User and software requirements keep on changing
 - ◆ Software developers expecting improvements in computing power

Why Study Parallel Computing

- Technology made multicore processors both feasible AND necessary for performance
 - ◆ Moore's law: too many transistors on a die than can be used for a single processor
 - ◆ Traditional out-of-order processors face memory and power walls
- Software requirements need more computing power than a single processor
 - ◆ Scientific computations
 - ◆ Commercial applications

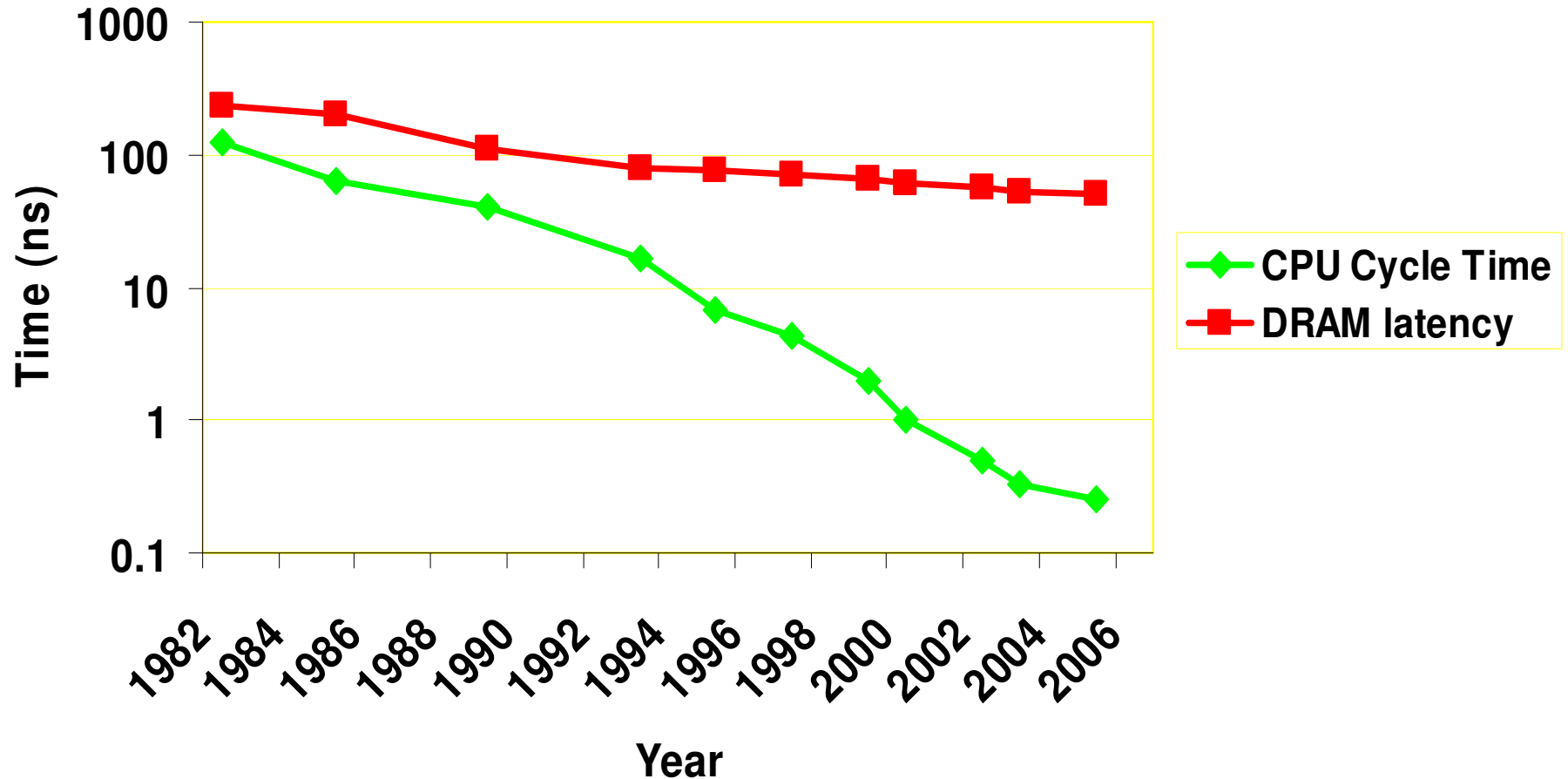
Moore's Law (1965)

#Transistors Per Chip (Intel)



**Almost 75%
increase per
year**

Memory Wall



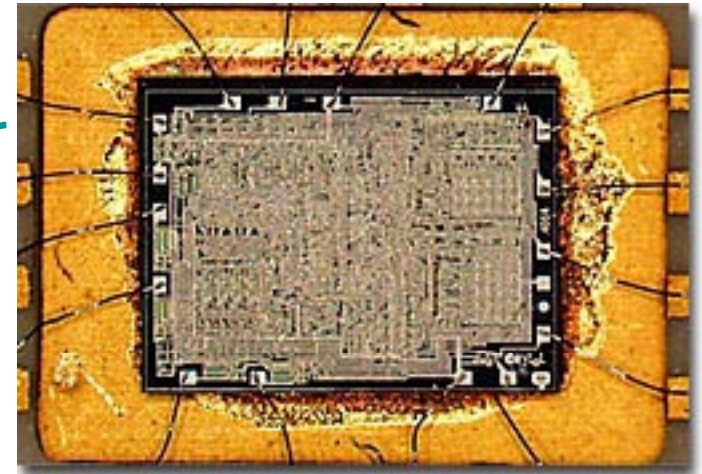
- **CPU cycle time: 500 times faster since 1982**
- **DRAM Latency: Only ~5 times faster since 1982**

Why Not Build Larger OoO Processors?

- Memory Wall
 - ◆ 1980 Memory Latency ~ 1 instruction
 - ◆ 2006 Memory Latency ~ 1000 instructions
- Power Wall
 - ◆ Dynamic power increases quadratically with frequency, static power increases exponentially
- ILP Limitations
 - ◆ Serial programs don't have an infinite amount of instructions to execute in parallel
- Other Problems
 - ◆ Temperature & cooling
 - ◆ On-chip interconnect
 - ◆ Complexity & testing

Technology Trends: Growing #Transistors Causes Inflection Points

- Most famous inflection point enabled simple microprocessor
 - ◆ 1971 Intel 4004
 - ◆ 2300 transistors
 - ◆ Could access 300 bytes of memory (0.0003 megabytes)



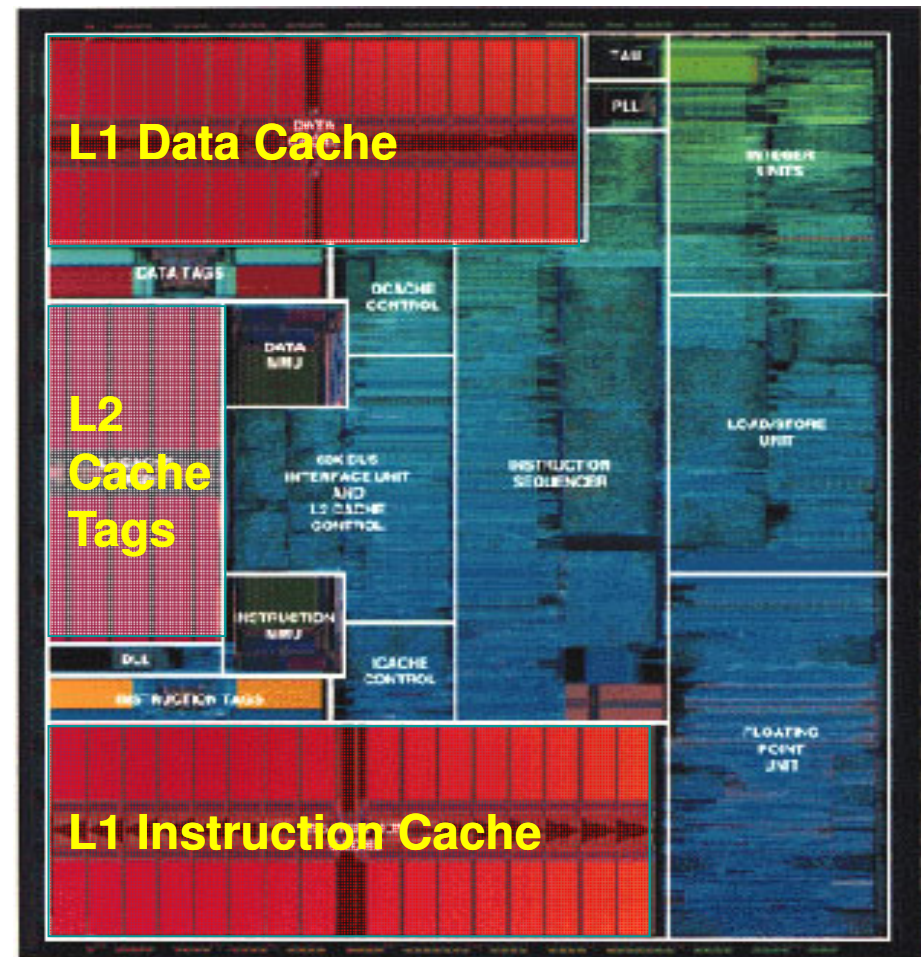
- Bigger and better-performing processors were enabled by more transistors

©2006, Prof. Mark Hill

Microprocessor Design Complexity

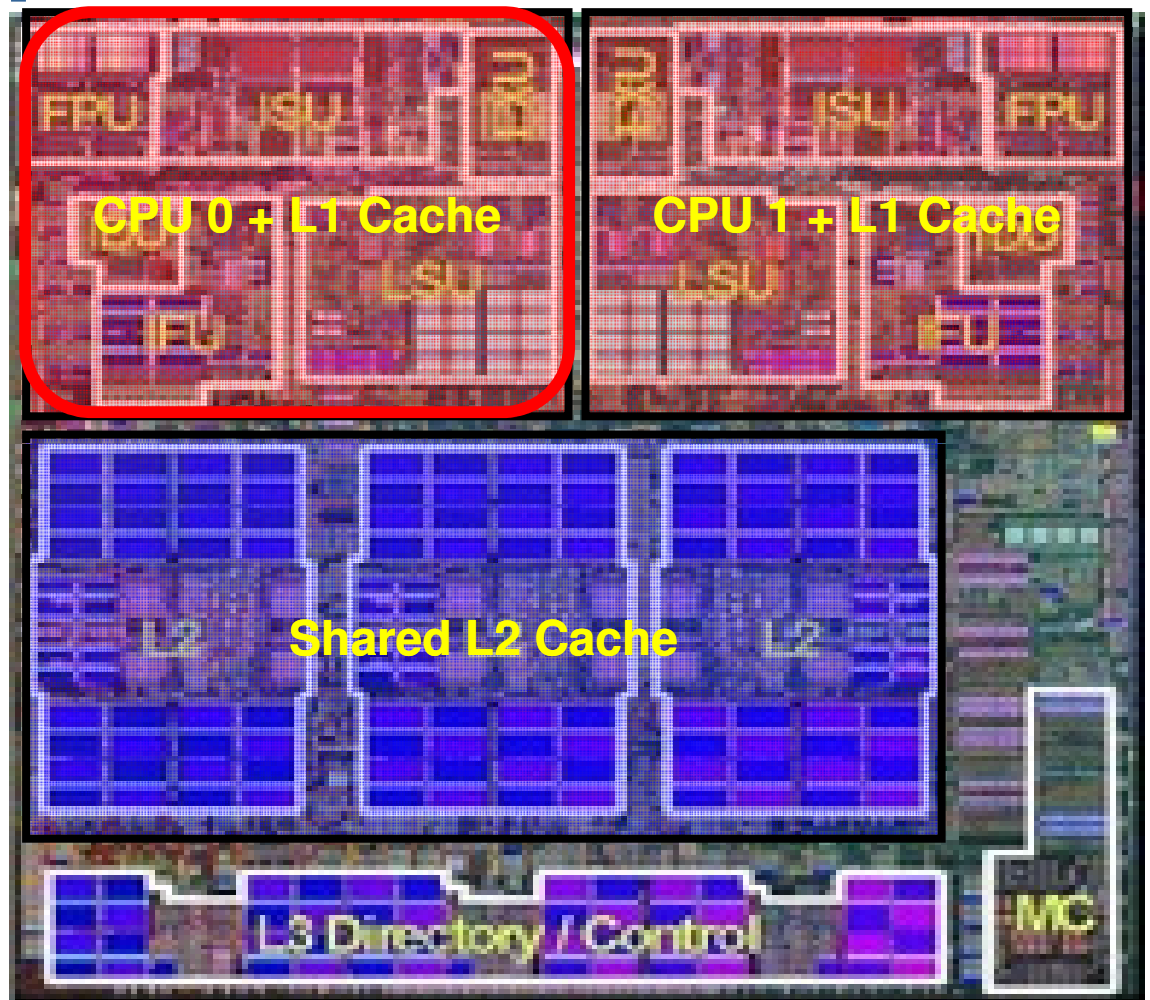
- Millions of Transistors
- Bit-level Parallelism (BLP)
 - ◆ 64-bit multiply in one/two cycles
- Instruction-Level Parallelism (ILP)
 - ◆ Dozens of instruction in flight
 - ◆ Branch prediction
 - ◆ Out-of-order
- Dominating Use of Caches
 - ◆ Level-one cache
 - ◆ Level-two cache
- Emerging Chip Multiprocessing

PowerPC 750 (1998)



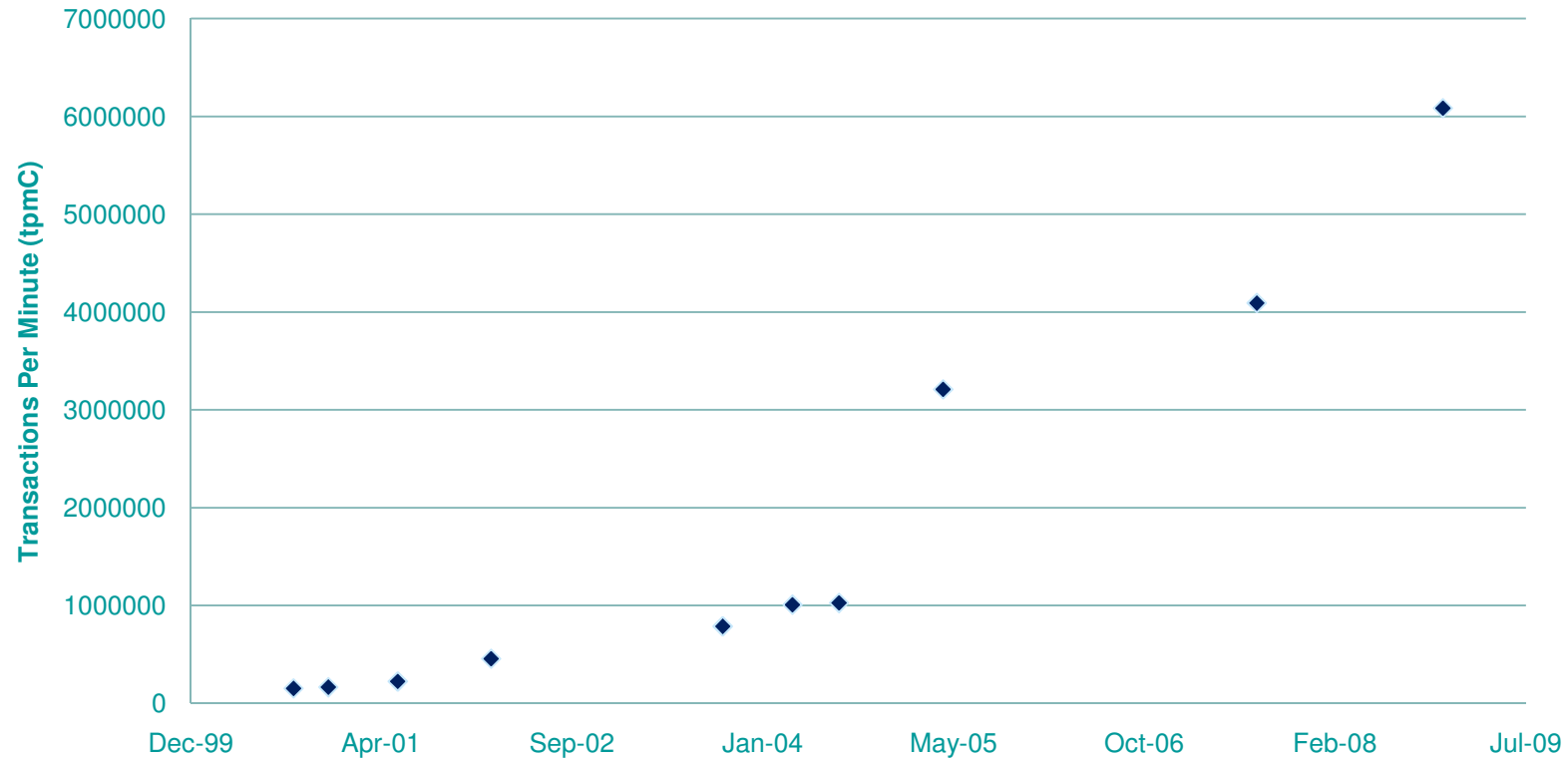
Multicore Processors (Chip Multiprocessors)

- Replicate
 - ◆ processor “core” &
 - ◆ Caches
- Uses more transistors
- Tolerates “memory wall”
- Simpler lower-power cores
- Reduces complexity



IBM Power 4

Software Trends

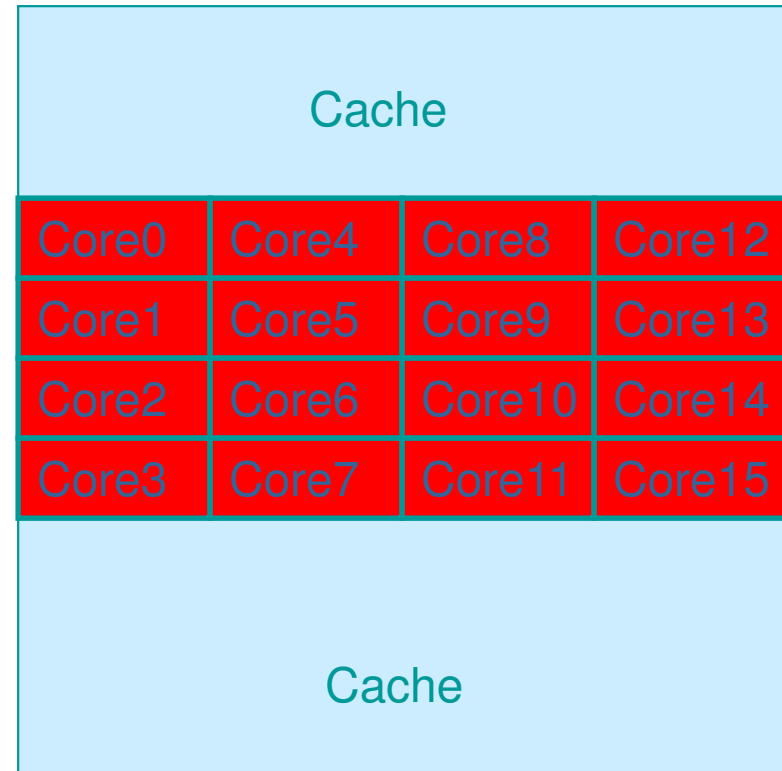
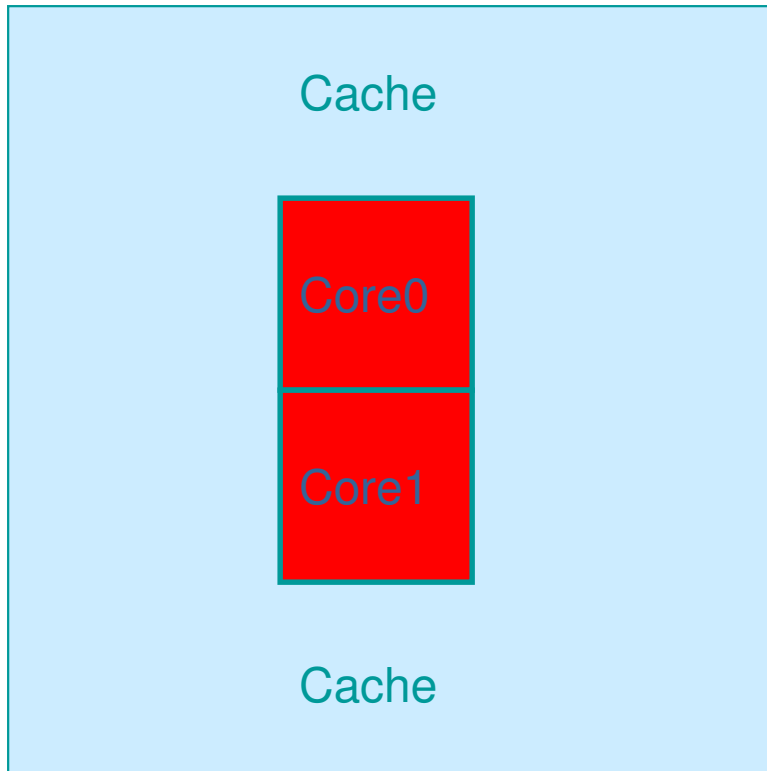


- TPC-C Throughput increased by more than 75% per year over eight years
- What about desktop applications?
- What about scientific applications?

Multi-Core Performance

- Recall Popular Moore's Law:
 - ◆ Microprocessor performance doubles every 1.5-2 years
- Future Multi-Core Performance Doublings Require
 - ◆ Effective multithreaded programming
 - ◆ Better communication
 - ◆ Faster synchronization
 - ◆ More cores

Multicore Processor Architecture



- Which design is better?
 - ◆ Balance cores, caches, and pin bandwidth

Programming Models

- Single-core
 - ◆ One program runs on core
 - ◆ Programmers write efficient programs
 - ◆ Architects design for fast instruction execution
- Multicore
 - ◆ Each core can run a thread/task/program
 - ◆ Programmers can split up their programs?
- Multiprocessors
 - ◆ Shared memory
 - ◆ Message passing
 - ◆ Threads
 - ◆ Data Parallel

Introduction to Parallel Computing

- Serial vs. Parallel Computation (tutorial pictures)
- Parallel computing includes
 - ◆ Break up problem into smaller discrete parts that can be solved concurrently
 - ◆ Break each part further into a sequence of (serial) instructions
 - ◆ All parts are run simultaneously on different CPUs
- Why use parallel computing?
 - ◆ Save time
 - ◆ Solve larger problems
 - ◆ Provide concurrency (do more than one thing at the same time)
 - ◆ Save money (use multiple cheaper resources vs. a single expensive supercomputer)
 - ◆ Use more memory than available on a single computer
 - ◆ Use non-local resources
- Who uses parallel computing and why? (tutorial graphs)

Reading Assignment

- Introduction to Parallel Computing, Lawrence Livermore National Laboratory tutorial
- Read before Wed class
- Monday: Olukotun et al., “The Case for a Single-Chip Multiprocessor,” ASPLOS 1996
- Submit review before the beginning of Mon class:
 - ◆ Paper summary
 - ◆ Strong points (2-4 points)
 - ◆ Weak points (2-4 points)
- Review due in my email inbox before 7PM Mon
 - ◆ Plain text, no attachments
 - ◆ Subject line has to be: “ECE588_REVIEW_10_05”