CS 61C: Great Ideas in Computer Architecture (Machine Structures)
Course Wrap-up

Instructors:
Randy H. Katz
David A. Patterson

http://inst.eecs.Berkeley.edu/~cs61c/fa10

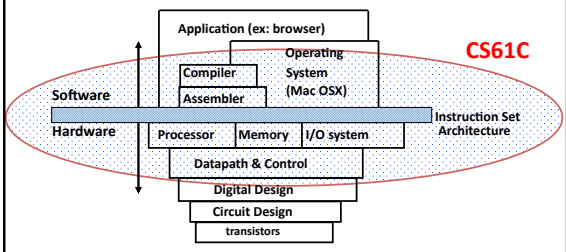
Agenda

- Course Wrap-up (Randy)
- Administrivia
- Technology Break
- Berkeley Culture (Dave)

Agenda

- Course Wrap-Up (Randy)
- Administrivia
- Technology Break
- Berkeley Culture (Dave)

Old School View of Software-Hardware Interface

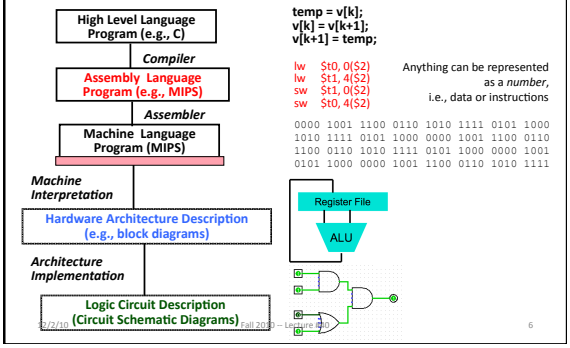


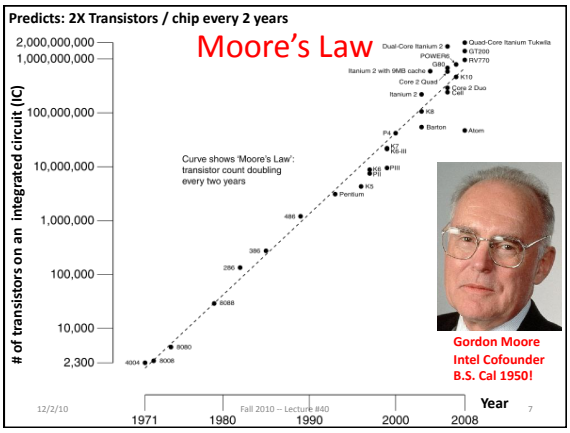
Great Ideas in Computer Architecture

- ✓ Layers of Representation/Interpretation
- ✓ Moore's Law
- ✓ Principle of Locality/Memory Hierarchy
- ✓ Parallelism
- ✓ Performance Improvement

- Learning how to program in C is *non-goal*!
  - Check out http://inst.eecs.berkeley.edu/~selfpace/class/cs9c/
  - Since C doesn't hide pointers and memory allocation, it is a good language for illustrating architectural concepts and abstracting from assembly language
  - Foundation for exploring parallelism and performance improvement, though mechanisms and libraries for SIMD, OpenMP, Map-Reduce, etc. exist for other languages

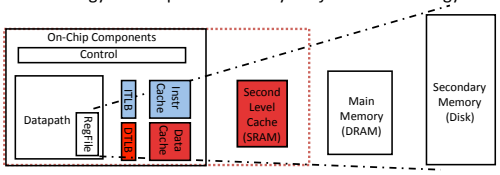
Levels of Representation/Interpretation





Typical Memory Hierarchy

- Take advantage of the principle of locality to present the user with as much memory as is available in the *cheapest* technology at the speed offered by the *fastest* technology

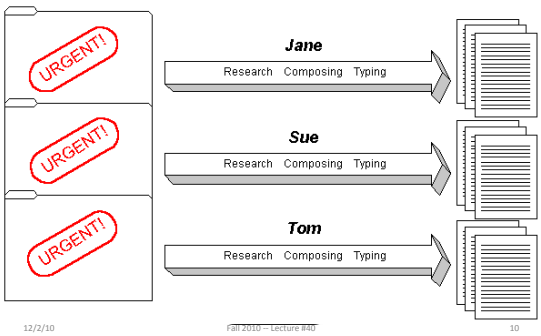


Speed (%cycles):	½'s	1's	10's	100's	10,000's
Size (bytes):	100's	10K's	M's	G's	T's
Cost:	highest				lowest

Memory Hierarchy

- Caches
  - 3Cs: Compulsory/Capacity/Conflict misses
  - Direct-mapped vs. Set-associative
  - Multi-level caches for fast clock + low miss rate
- Virtual Memory
  - Originally small physical memory that appears to be very large
  - Modern: provide isolation through separated address spaces

Parallelism



Forms of Parallelism

- Instruction Parallelism
  - Processor pipeline: multiple instructions in execution at the same time
- Task Parallelism
  - Synchronization primitives, openMP
  - Modern web services
- Data Parallelism
  - Map-Reduce, SIMD instruction set
  - Data and numerically intensive processing

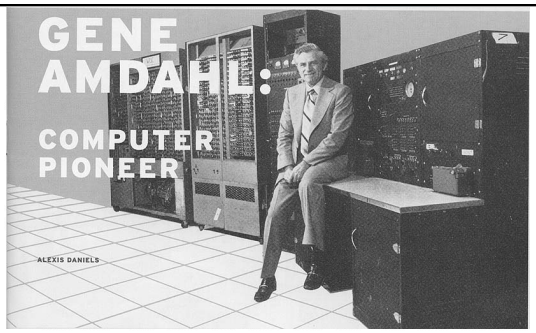


Fig 3 Amdahl's Law an Obstacle to Improved Performance Performance will not rise in the same proportion as the increase in CPU cores. Performance gains are limited by the ratio of software processing that must be executed sequentially. Amdahl's Law is a major obstacle in boosting multicore microprocessor performance. Diagram assumes no overhead in parallel processing. Years shown for design rules based on Intel planned and actual technology. Core count assumed to double for each rule generation.

Randy's Course Summary

- As the field changes, cs61c has to change too!
- It is still about the software-hardware interface
  - Programming for performance!
  - Understanding the memory hierarchy and its impact on application performance
  - Unlocking the capabilities of the architecture for performance
    - Multicore programming and task parallelism
    - Special instructions
    - Special instruction sequences
- Thanks for being our guinea pigs this semester!
  - Wait until you interview for summer internships and tell the interviewers what you did this semester!

Agenda

- Course Wrap-Up (Randy)
- Administrivia
- Technology Break
- Berkeley Culture (Dave)

Administrivia

- Final Review: Mon Dec 6, 2-5 PM, 10 Evans
- Final: Mon Dec 13 8-11AM (220 Hearst Gym)
  - Like midterm: T/F, M/C, short answers
  - Whole Course: readings, lectures, projects, labs, hw
  - Emphasize 2nd half of 61C + midterm mistakes

Agenda

- Course Wrap-Up (Randy)
- Administrivia
- Technology Break
- Berkeley Culture (Dave)

Agenda

- Course Wrap-Up (Randy)
- Administrivia
- Technology Break
- Berkeley Culture (Dave)