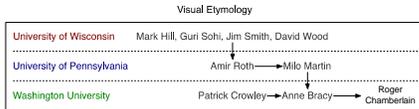


# CSE 560 Computer Systems Architecture

## Introduction



## Administrative Staff

- Instructor: Roger Chamberlain
  - Office – Jolley 210      Email – roger@wustl.edu
  - Office Hours – 4pm to 5pm Mondays and Wednesdays
- Teaching Assistants: Clayton Faber, Anthony Cabrera
  - Office – Jolley 219      Email – cfaber@wustl.edu
  - Office Hours – TBD      acabrera@wustl.edu
- Please use Piazza over email for asking questions
  - It's not yet setup, but will be this week
  - Emails get lost in the pile, Piazza posts don't
- Optional Text: J.-L. Baer, *Microprocessor Architecture: From Simple Pipelines to Chip Multiprocessors*, Cambridge University Press, 2010

## Grading Logistics

- Three elements
  - Practice problems – approx. once per week
  - Assignments – typically simulation experiments
  - Two exams – midterm and final
- Practice problems
  - Solutions posted about one week after problems are posted
  - No impact on grade, but they are practice for exams
- Assignments
  - 4 or 5 during the course of the semester
- Exams
  - Closed book, one-page crib sheet is allowed
- Grading:
  - Assignments            20%
  - Midterm – Oct 25      40% (in class)
  - Final – Dec 18        40% (wherever they assign us to be)

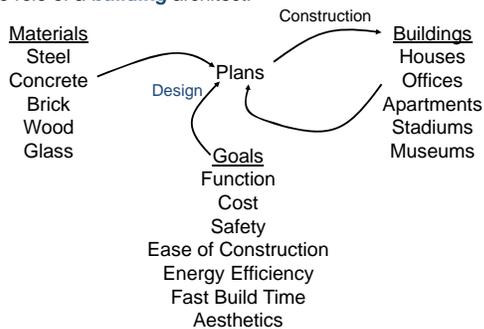
## What is Computer Architecture?

“Computer Architecture is the science and art of selecting and interconnecting hardware components to create computers that meet functional, performance and cost goals.”  
- Old WWW Computer Architecture Page

An analogy to architecture of buildings...

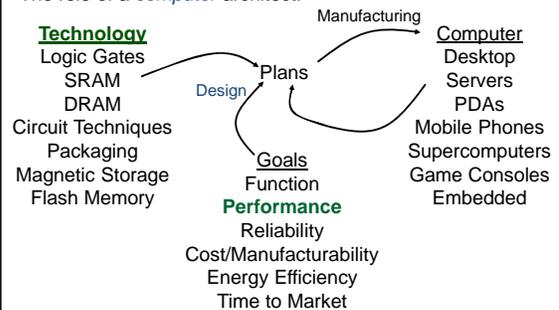
## What is ~~Computer~~ Architecture?

The role of a **building** architect:



## What is Computer Architecture?

The role of a **computer** architect:



## Important Differences...

- Age of discipline: 60 years (vs. 5,000 years)
- Automated mass production
  - Advances magnified over millions of chips
- Boot-strapping effect
  - Better computers help design next generation

*When Seymour Cray was told that Apple had just purchased a Cray computer that would be used in designing the next Macintosh, he thought for a minute, and replied that that seemed reasonable, since he was using a Macintosh to design the next Cray.*
- Rate of change
  - Technology, Applications, Goals changing **quickly**

7

## Survey Time

Which of the following statements is true?

- A:** If you can verify that a chip works correctly, you can be sure it will continue to work correctly in the future.
- B:** Chip manufacturing today has much better yield (of working chips) than it did decades ago.
- C:** Building reliable (correctly working) chips today is easier than it was decades ago.
- D:** It costs ~\$300,000,000 to build a fabrication plant.
- E:** If you have an idea that can make a CPU run at a higher frequency, you should definitely implement it (i.e., it's always a good idea).

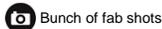


8

## Survey Time

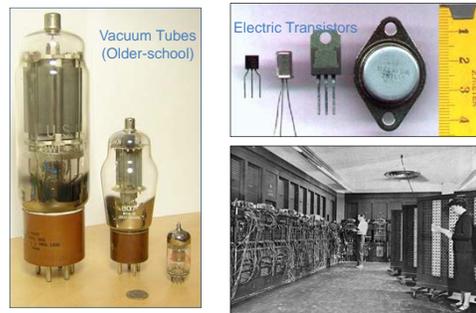
Which of the following statements is true?

- A:** If you can verify that a chip works correctly, you can be sure it will continue to work correctly in the future.
- B:** Chip manufacturing today has much better yield (of working chips) than it did decades ago.
- C:** Building reliable (correctly working) chips today is easier than it was decades ago.
- D:** It costs ~\$300,000,000 to build a fabrication plant.
- E:** If you have an idea that can make a CPU run at a higher frequency, you should definitely implement it.



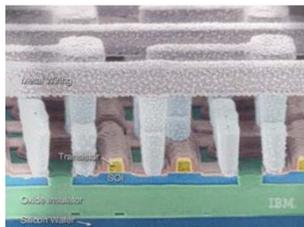
9

## Old-school Transistors → Old-school Computers



10

## Modern Transistor



IBM SOI Technology  
From slides © Krste Asanovic, MIT

11

## Building a Fab



Intel Fab 11x project, Submicron Manufacturing Facility, Rio Rancho, New Mexico: Aug 28, 2000

Building a Fab



Building a Fab



Inside a Fab



Inside a Fab



Inside a Fab



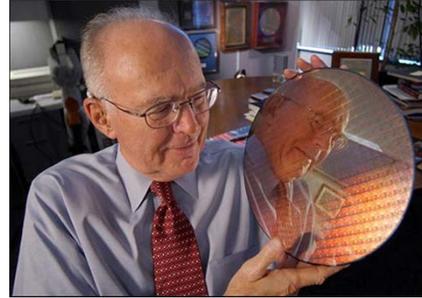
Inside a Fab



## Inside a Fab



## Gordon Moore with a Wafer



## Design Goals (1)

- **Functional**
  - Correctness – harder than software
  - What functions should it support?
- **Reliable**
  - Does it *continue* to perform correctly?
  - Hard fault vs. transient fault
  - Desktop vs. server vs. space probe reliability
- **High performance**
  - “Fast” — only meaningful in the context of set of tasks
  - Not just GHz – truck vs. sports car analogy
  - Impossible: fastest possible design for all programs

21

## Design Goals (2)

- **Low cost:** engineer's dime/fool's dollar
  - Per unit manufacturing cost (wafer cost)
  - Cost of making first chip after design (mask cost)
  - Design cost (huge design teams, why? Two reasons...)
- **Low power/energy – “the new performance”**
  - Energy in (battery life, cost of electricity)
  - Energy out (cooling and related costs)
- **Challenge: balancing these goals**
  - Balance constantly changing
  - Focus for us: **Performance**

22

## Shaping Force: Applications/Domains

Another shaping force: **applications** (usage and context)  
Different domains → different needs → different designs

- **Scientific:** weather prediction, genome sequencing
  - 1<sup>st</sup> computing application domain: ballistics tables
  - **Need:** large memory, heavy-duty floating point
  - Examples: Cray XC, IBM BlueGene
  - Making a comeback → anything that works on lots of data
- **Commercial:** database/web serving, e-commerce, Google
  - **Need:** data movement, high memory + I/O bandwidth
  - E.g., Intel Xeon, AMD Opteron

23

## Survey Time

**Which of the following is a correct quote?**

1. "I think there is a world market for maybe five computers."  
Thomas J. Watson, chairman of the board, IBM, 1943
  2. "There is no reason for any individual to have a computer in his home."  
Ken Olson, founder, Digital Equipment Corp., 1972
- A:** only 1 is correct  
**B:** only 2 is correct  
**C:** both are correct quotes  
**D:** neither are correct quotes



24

## Survey Time

Which of the following is a correct quote?

1. "I think there is a world market for maybe five computers."

Thomas J. Watson, chairman of the board, IBM, 1943

2. "There is no reason for any individual to have a computer in his home." Ken Olson, founder of Digital Equipment Corp. founder, 1972

A: only 1 is correct

B: only 2 is correct

C: both are correct quotes

D: neither are correct quotes

*But back then, #2 was correct.*

25

## More Applications/Domains

- **Desktop:** home office, multimedia, games
  - **Need:** integer, memory b/w, integrated graphics/network?
  - Examples: Intel Core 2, Core i7, AMD Athlon, PowerPC G5
- **Mobile:** laptops, mobile phones
  - **Need: low power**, integer performance, integrated wireless
  - Laptops: Intel Core 2 Mobile, Atom, AMD Turion
  - Examples: ARM chips by Samsung and others, Intel Atom
- **Embedded:** microcontrollers in automobiles, door knobs
  - **Need:** low power, **low cost**
  - Examples: ARM chips, dedicated digital signal processors (DSPs)
  - Over 1 billion ARM cores sold in 2006 (at least one per phone)
- **Deeply Embedded:** disposable "smart dust" sensors
  - **Need:** extremely low power, extremely low cost

26

## Application Specific Designs

- This class mostly about **general-purpose CPUs**
  - Processor that can do anything, run a full OS, *etc.*
  - *E.g.*, Intel Core i7, AMD Opteron, IBM Power, ARM
- In contrast to **application-specific chips**
  - Or ASICs (Application specific integrated circuits)
  - Implement critical domain-specific functionality in hardware
    - Examples: video encoding, cryptography
  - General rules
    - Hardware is less flexible than software
    - + Hardware more effective (speed, power, cost) than software
    - + Domain specific more "parallel" than general purpose
      - *But general mainstream processors becoming more parallel!*
- **Trend:** from specific to general (for a specific domain)

27

## Revolution I: The Microprocessor

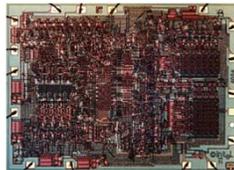
- **Microprocessor revolution:** 16-bit processor on 1 chip!
  - 1970s, ~25K transistors
  - Performance advantages: fewer slow chip-crossings
  - Cost advantages: one "stamped-out" component
- **Out with the old**
  - Microprocessor-based systems replace supercomputers, "mainframes", "minicomputers", *etc.*
- **In with the new**
  - Desktops, CD/DVD players, laptops, game consoles, set-top boxes, cell phones, digital camera, ipods, GPS...

28

## First Microprocessor

### Intel 4004 (1971)

- Application: calculators
- Technology: 10  $\mu\text{m}$  PMOS



- 2300 transistors
- 13 mm<sup>2</sup>
- 108 kHz
- 12 Volts

- 4-bit data
- Single-cycle datapath

29

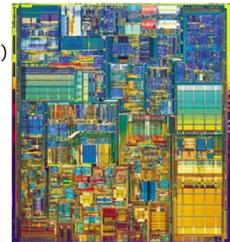
## Pinnacle of Single-Core Microprocessors

### Intel Pentium4 (2003)

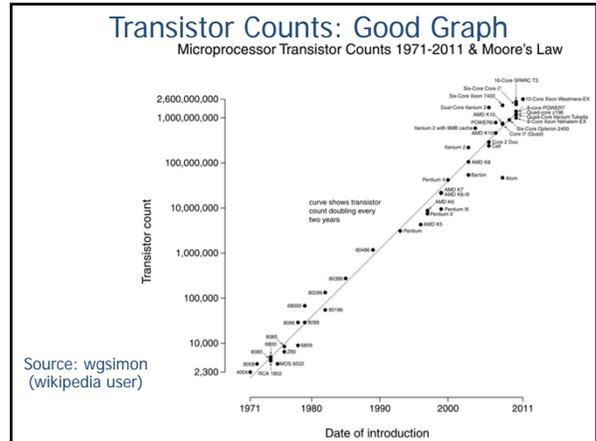
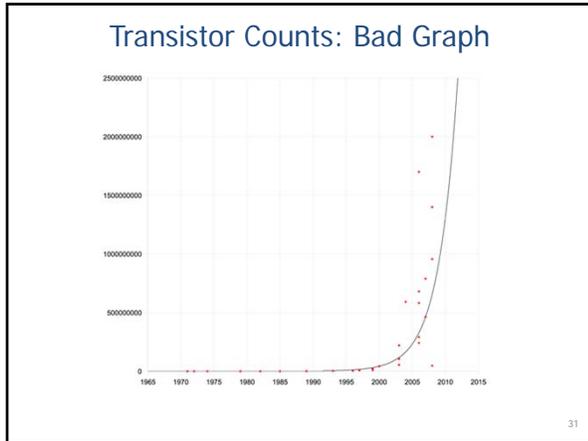
- Application: desktop/server
- Technology: 0.09  $\mu\text{m}$  CMOS (1/100X)

- 55M transistors (20,000X)
- 101 mm<sup>2</sup> (10X)
- 3.4 GHz (10,000X)
- 1.2 Volts (1/10X)

- 32/64-bit data (16X)
- 22-stage pipelined datapath
- 4 instructions per cycle (superscalar)
- Two levels of on-chip cache
- data-parallel (SIMD) instructions, hyper-threading



30



### What to do with all these transistors?

**First things first: expressiveness**

- Widen the datapath (4004: 4 bits → Pentium4: 64 bits)
- More powerful instructions
  - To amortize overhead of fetch and decode
  - To simplify programming (done by hand then)

33

### Revolution II: Implicit parallelism

**Extract implicit instruction-level parallelism (ILP)**

- Hardware parallelizes, software is oblivious

**Round 1:**

- Pipelining** → increased clock frequency
- Caches**: became necessary as frequencies increased
- Integrated floating-point

**Round 2:**

- Deeper pipelines and **branch speculation**
- Multiple issue (**superscalar**)
- Dynamic scheduling (**out-of-order execution**)

34

### Relatively Recent Multicore Processor

Intel Core i7 (2009)

- Application: desktop/server
- Technology: 45nm (1/2x)
- 774M transistors (12x)
- 296 mm<sup>2</sup> (3x)
- 3.2 GHz to 3.6 GHz (~1x)
- 0.7 to 1.4 Volts (~1x)
- 128-bit data (2x)
- 14-stage pipelined datapath (0.5x)
- 4 instructions per cycle (~1x)
- Three levels of on-chip cache
- data-parallel vector (SIMD) instructions, hyperthreading
- Four-core multicore** (4x)

35

### Revolution III: Explicit Parallelism

**Support explicit data & thread level parallelism**

- HW provides parallel resources, SW specifies usage
- Why? Diminishing returns on ILP

**Round 1: Vector instructions...**, Intel's SSE

- One instruction → 4 parallel multiplies

**Round 2:** Support for multi-threaded programs

- Coherent caches, hardware synchronization primitives

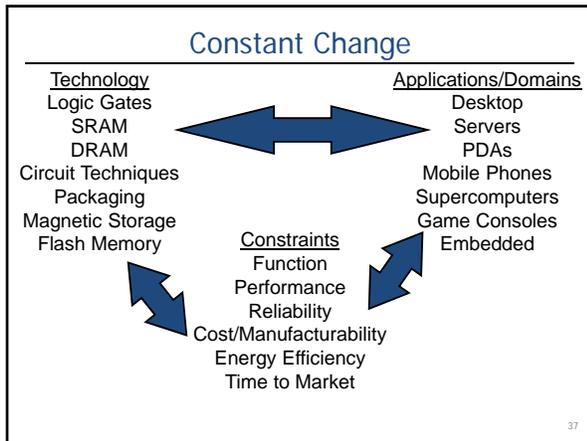
**Round 3:** Support for multiple concurrent threads on chip

- Single-core **multi-threading** → **multi-core**

**Round 4:** highly parallel Graphics processing units (GPUs)

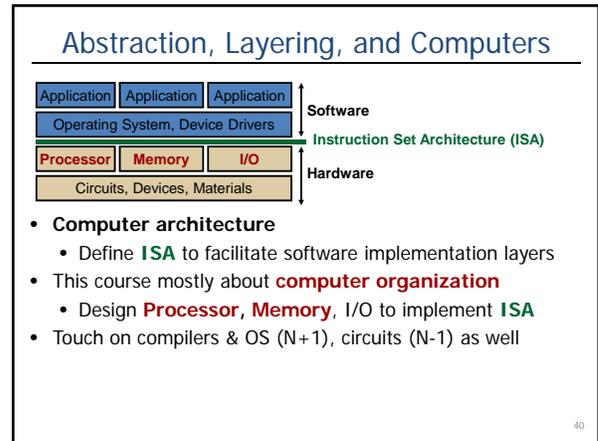
- Converging with general-purpose processors (CPUs)?
- AMD bought ATI, Intel making GPUs

36



- ### Technology Disruptions
- Classic examples:
- The transistor
  - Microprocessor
- More recent examples:
- Multicore processors
  - Flash-based solid-state storage
- Near-term potentially disruptive technologies:
- Phase-change memory (non-volatile memory)
  - Chip stacking (also called 3D die stacking)
- Disruptive "end-of-scaling"
- "If something can't go on forever, it must stop eventually"
  - Can we continue to shrink transistors for ever?
  - Even if more transistors, not getting as energy efficient as fast
- 38

- ### Pervasive Idea: Abstraction and Layering
- **Abstraction:** divide complex systems into objects with:
    - **Interface:** for the common man
    - **Implementation:** "black box" for the specialists
    - E.g., car, only mechanics understand implementation
  - **Layering**
    - Implement X using interface of layer just below
    - Ignore lower layers (sometimes helps)
  - **Inertia:** a dark side of layering
    - Interfaces become stagnant ("standards")
    - Getting layers to cooperate (Intel & Microsoft)
    - "Company X now making product Y"
  - **Opacity:** hard to reason about performance across layers
- 39



- ### Survey Time
- Which of the following statements is false?
- A: A programmer needs to understand how a processor handles memory traffic to ensure program correctness.
  - B: The number of companies designing their own chips gets smaller every year.
  - C: A programmer must understand how a particular chip's functional units work to ensure program correctness.
  - D: If a programmer doesn't care about performance, she doesn't gain much by understanding the architecture her program is running on.
  - E: All of them are false.
- 
- 41

- ### Survey Time
- Which of the following statements is false?
- A: A programmer needs to understand how a processor handles memory traffic to ensure program correctness.
  - B: The number of companies designing their own chips gets smaller every year.
  - C: A programmer must understand how a particular chip's functional units work to ensure program correctness.**
  - D: If a programmer doesn't care about performance, she doesn't gain much by understanding the architecture her program is running on.
  - E: All of them are false.
- 42

## Why Study Computer Architecture?

---

- **Understand where computers are going**
  - Future capabilities drive the (computing) world
  - Real impact: better computers make more things possible
- **Get a (design or research) hardware job**
  - Intel, AMD, IBM, ARM, Motorola, Sun, NVIDIA, NEC, Samsung
- **Get a (design or research) software job**
  - Best software designers understand hardware
  - Need to understand hardware to write fast software

43

## Course Goals

---

- **Understand “big ideas” in computer architecture**
- **Be a better scientist:** this is a *great* scientific playground
  - Good & bad engineering
  - Experimental evaluation/analysis (“science” in CS)
    - Computer performance and metrics
    - Quantitative data and experiments
    - Experimental design & Results presentation
- **Get your geek on:** think/speak like a computer architect
  - Possibly whether you want to or not 😊

44