CSCI 564 Advanced Computer Architecture

Lecture 01: Introduction and Logistics

Dr. Bo Wu (with modifications by Sumner Evans)

March 3, 2021

Colorado School of Mines

Credits

The slides in this course are Dr. Bo Wu's slides. I have transcribed them to LaTeX, and also updated a couple small things as well.

Most of the slides from the course are adapted from materials designed by four top-notch computer architecture researchers:

Onur Mutlu (CMU)

Steven Swanson (UCSD)

Rajeev Balasubramonian (Utah)

David Wentzlaff (Princeton)

Sumner

And you should call me “Sumner”.

- Jonathan is my legal first name, and I may sometimes respond to it, but it’s hit or miss.
- I do not have a PhD, so do not address me as a doctor.
- I’m a Mines alum (BS and MS in CS).
- I’m a software engineer at The Trade Desk.

Agenda

- What is architecture?
- Why is it important?
- What is in this class?
- Current state of computer architecture.
- Class logistics.
- A (very) brief history of computing.

What is Architecture?

What do you think of when you hear the word “Architecture”?

Notre Dame (Retrieved From Wikimedia, CC-BY-SA 2.0)

From the Miriam-Webster Dictionary, “Architecture” is “the manner in which the components of a computer or computer system are organized and integrated”.

What sorts of questions do Computer Architects ask?

- How do you build a machine that computes? Quickly, safely, cheaply, efficiently, etc.
- How can we provide usable abstractions to computer scientists and software engineers?
- Other questions that interest you?

iPhone 12 (Apple)

Summit Supercomputer (BBC)

Why is it Important?

Why is computer architecture important?

- For the world:
  - Computer architecture provides the engines that power all of computing.
  - Computing is everywhere! (Which is probably why you are getting a degree in CS!)

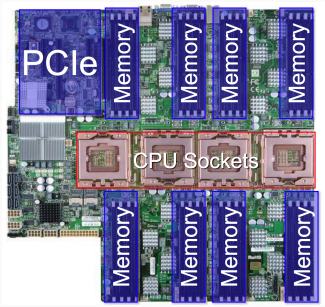
  Civilization advances by extending the number of important operations which we can perform without thinking about them.

  — *Alfred North Whitehead*

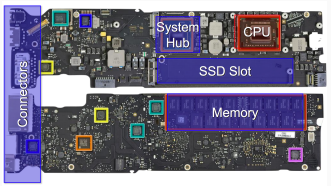
- For you:
  - As computer scientists, software engineers, and sophisticated users, understanding how computers work is essential.
  - The processor is the most important piece of this story.
  - Many performance problems have their roots in architecture.

Different Scales

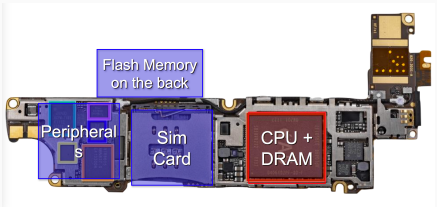
- High end server



- Ultrabook



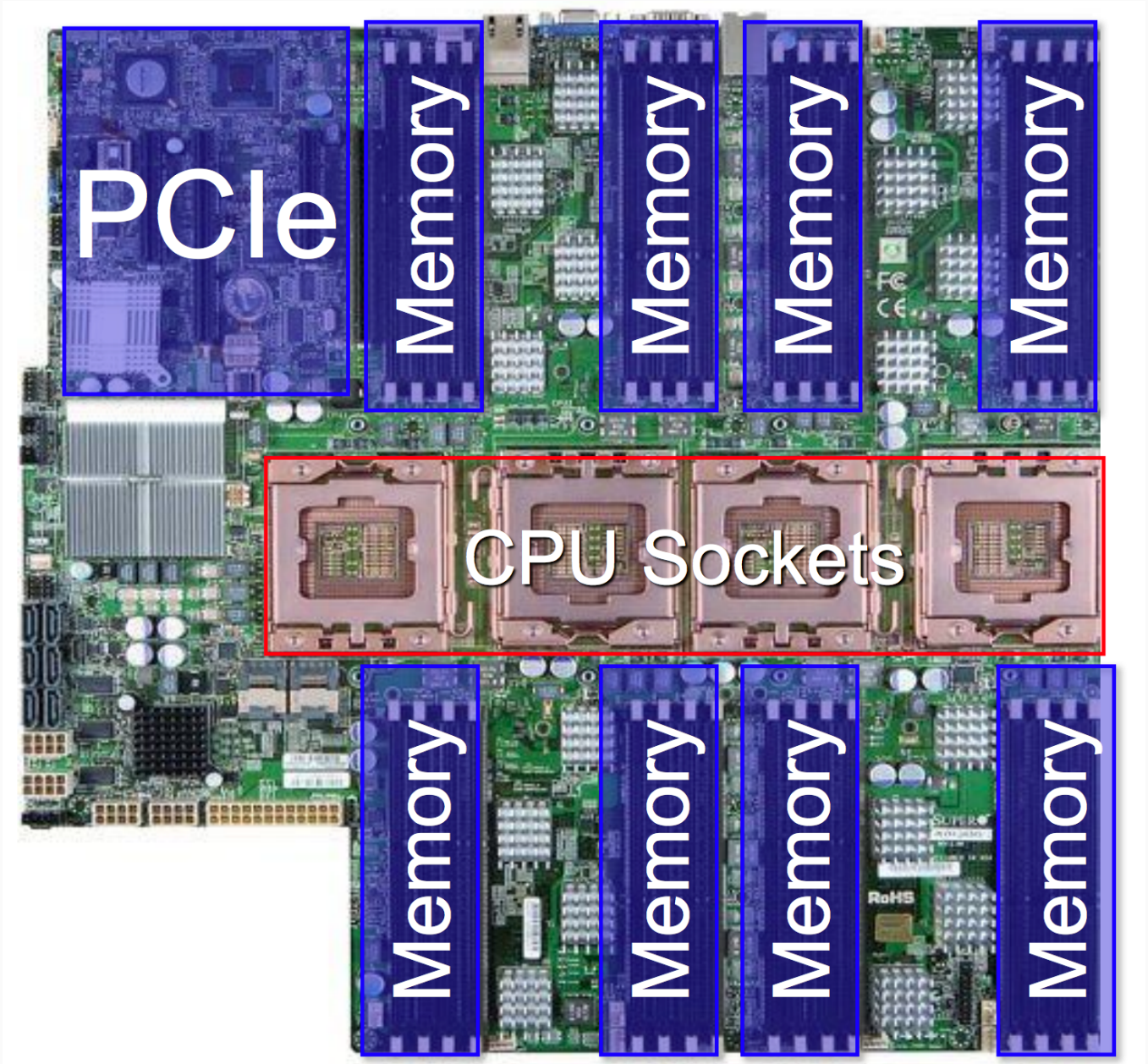
- Mobile



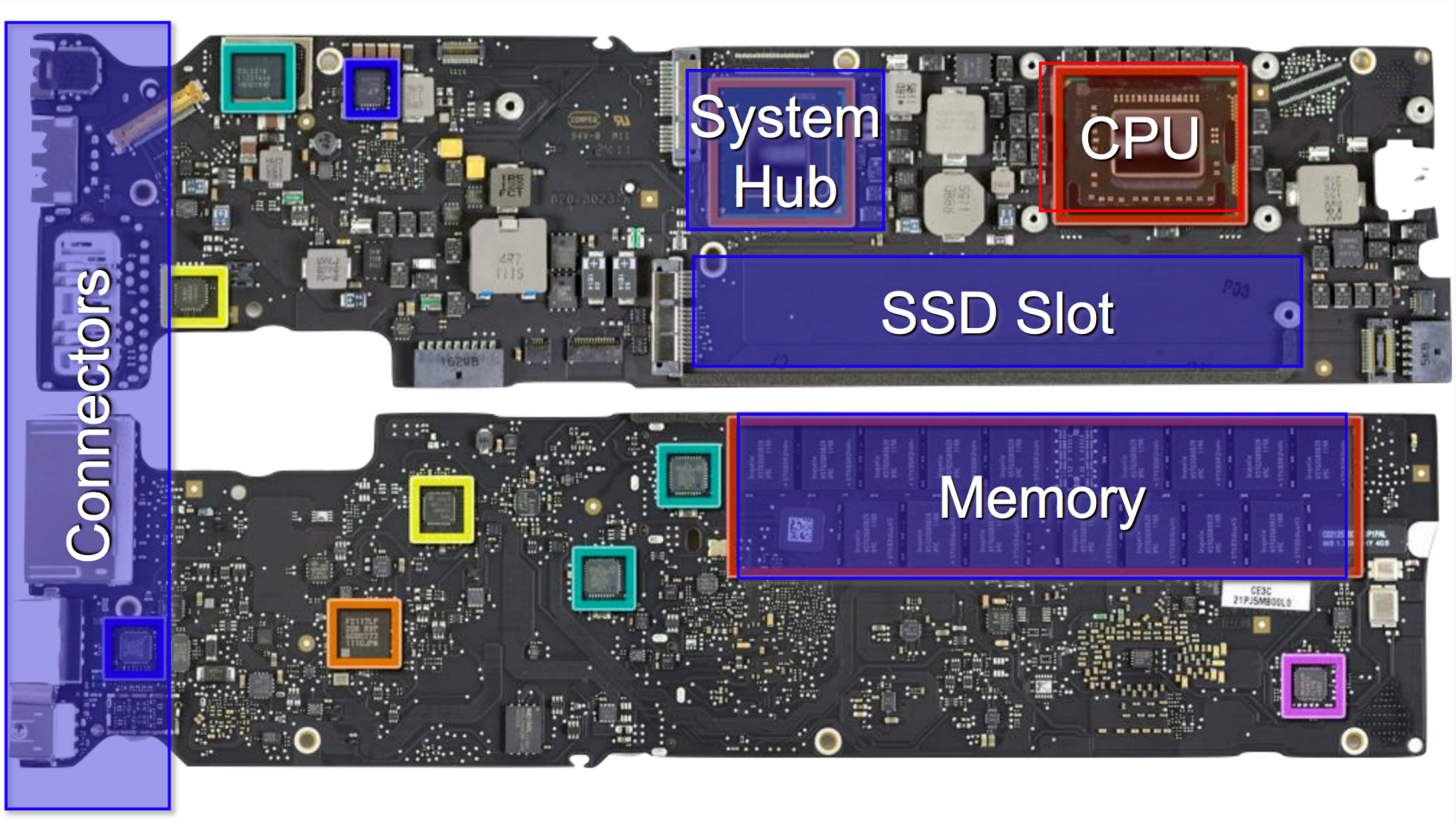
Architecturally, these machines are more similar than different.

- Same parts
- Different scale
- Different constraints

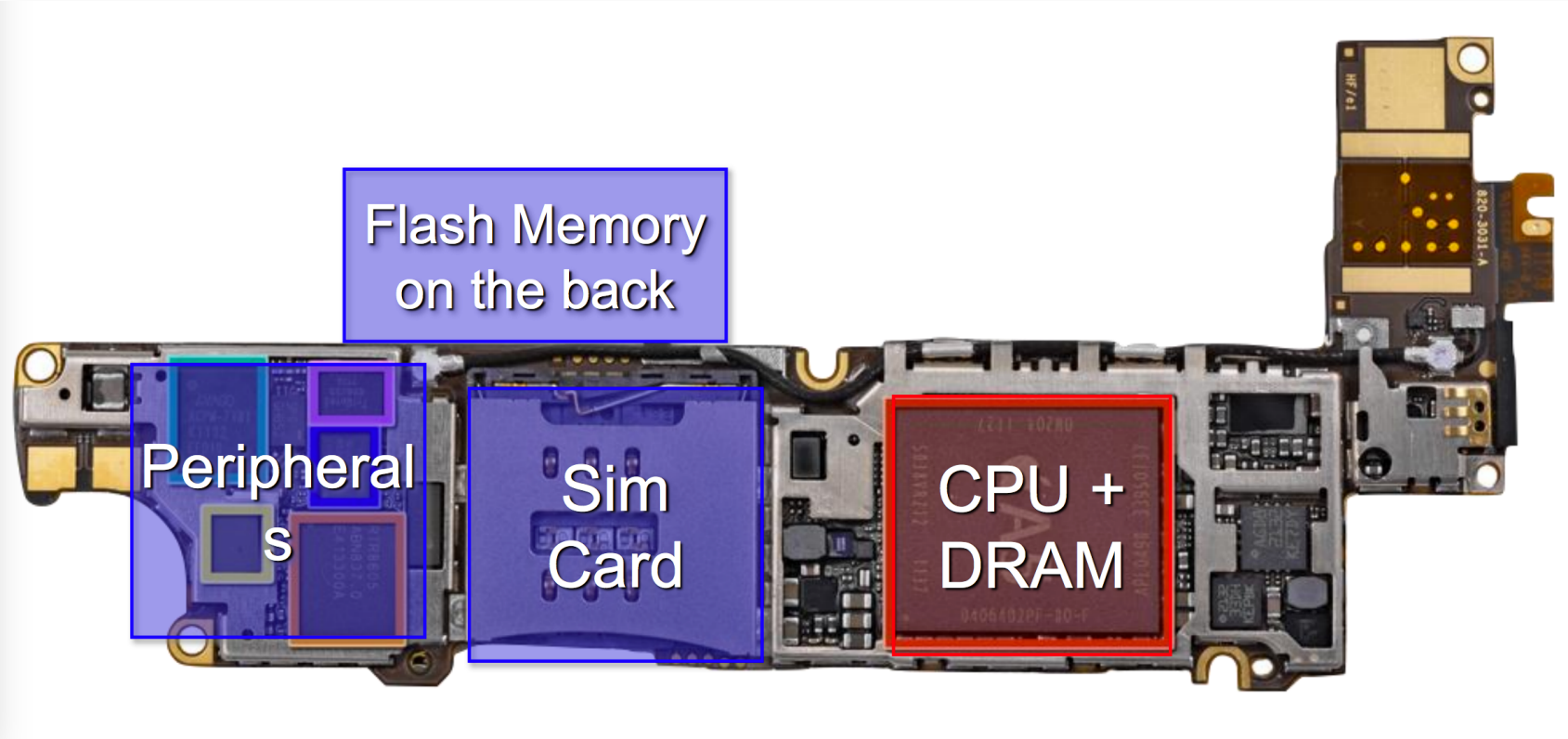
A Server


PCIe
Memory
Memory
Memory
Memory
CPU Sockets
Memory
Memory
Memory
Memory

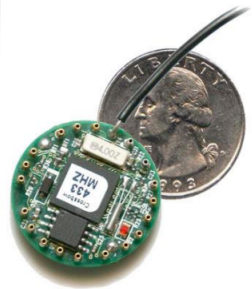
MacBook Air



iPhone 4s



Processors are everywhere!



From Sand to Applications

Cool things
happen here...

What is in this class?

Abstractions of the Physical World...

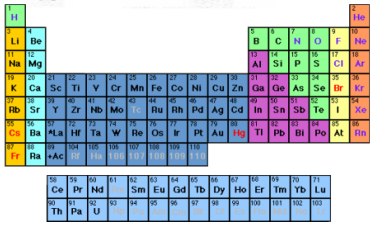
Physics/
Chemistry/
Material science

$$\oint H \cdot dl = I + \varepsilon \frac{d}{dt} \iint E \cdot ds$$

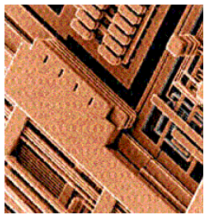
$$\oint E \cdot dl = -\mu \frac{d}{dt} \iint H \cdot ds$$

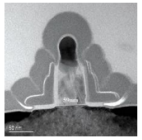
$$\mu \oiint H \cdot ds = 0$$

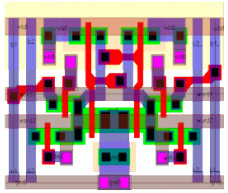
$$\varepsilon \oiint E \cdot ds = \iiint q_v dv$$



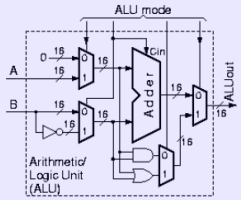
EE Dept.

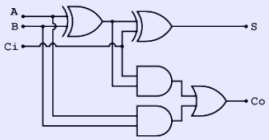


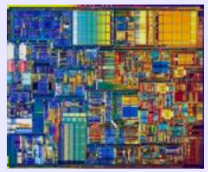


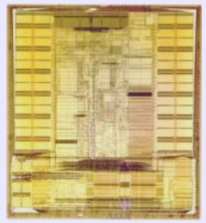


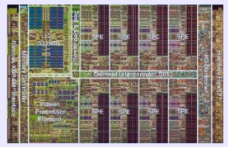
This Course











Physics/Materials | Devices | Micro-architecture | Processors | Architectures

...for the rest of the system

Architectures

Processor Abstraction

Compilers

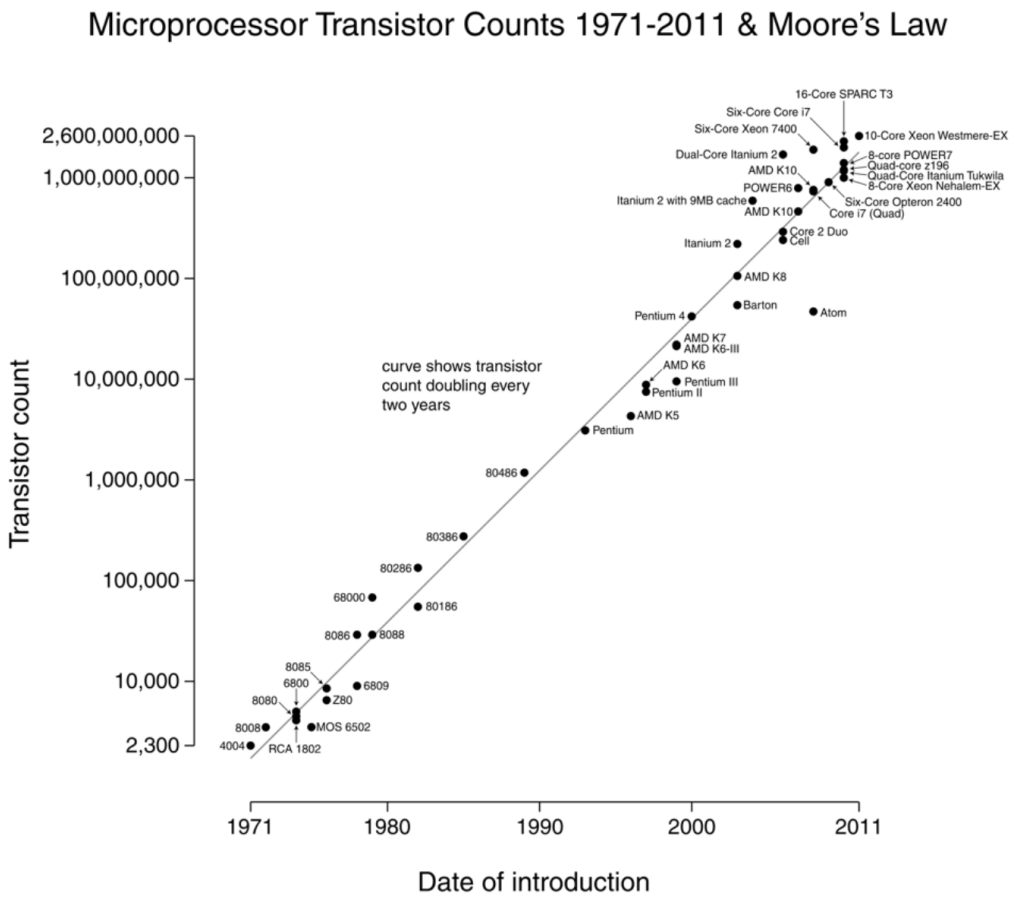
Languages

Software Engineers/Applications

Current state of architecture

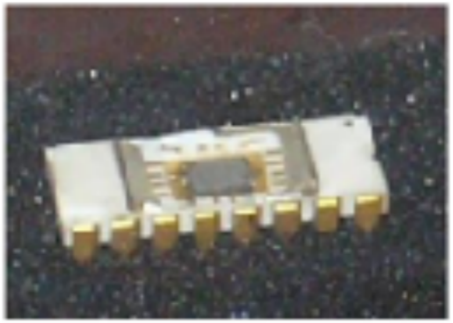
Moore's Law

The number of transistors we can build in a fixed area of silicon doubles every two years.



Moore's Law is the most important driver for historic CPU performance gains

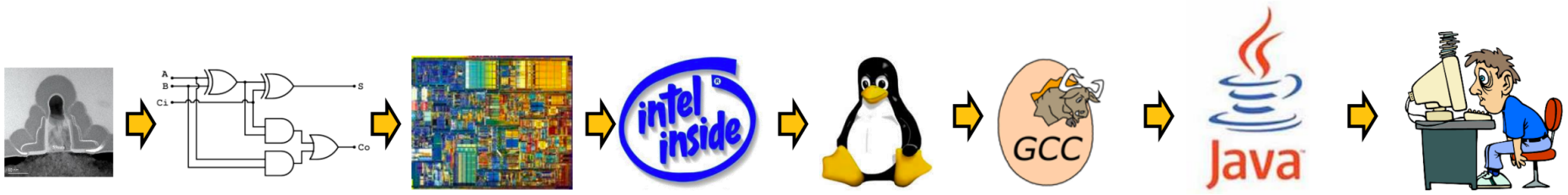
49 years of development



2300 Transistors
Intel 4004

8.5 billion transistors

Since 1940

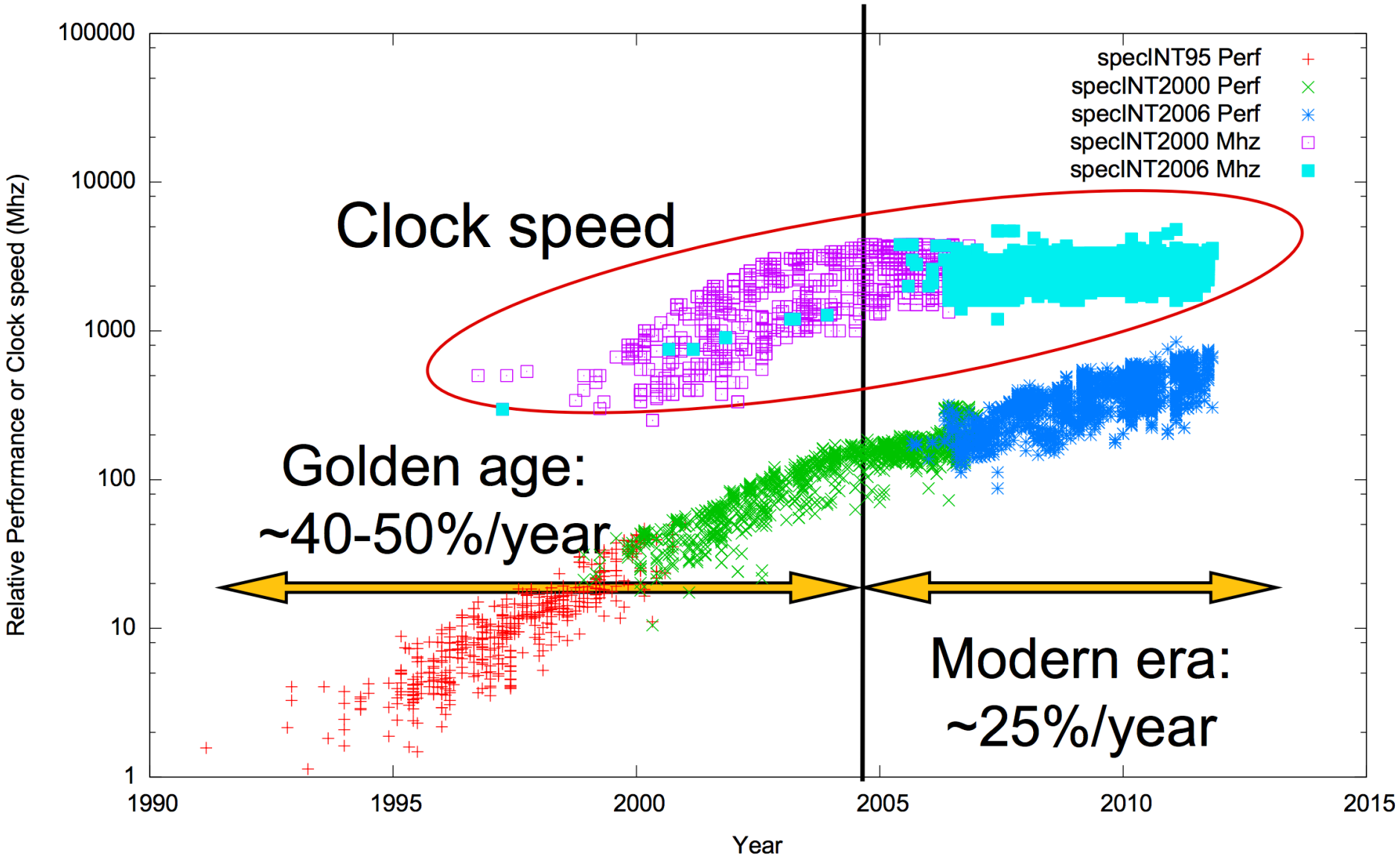


50,000 x speedup
>1,000,000,000 x density
(Moore's Law)

Plug boards -> Java
Hand assembling -> GCC
No OS -> Windows 7

We have used this performance to make computers easier to use, easier to program, and to solve ever-more complicated problems.

Evidence


specINT95 Perf
specINT2000 Perf
specINT2006 Perf
specINT2000 Mhz
specINT2006 Mhz
Clock speed
Golden age:
~40-50%/year
Modern era:
~25%/year
Relative Performance or Clock speed (Mhz)
Year
100000
10000
1000
100
10
1
1990
1995
2000
2005
2010
2015

The end of clock speed scaling

- *Clock speed* is the biggest contributor to *power consumption*.
  - Doubling the clock speed increases power by 4-8X
  - Clock speed scaling is essentially finished
- Most future performance improvements will be due to architectural and process technology improvements.
  - Indicates that computer architecture research is more important than ever!

Power and Heat

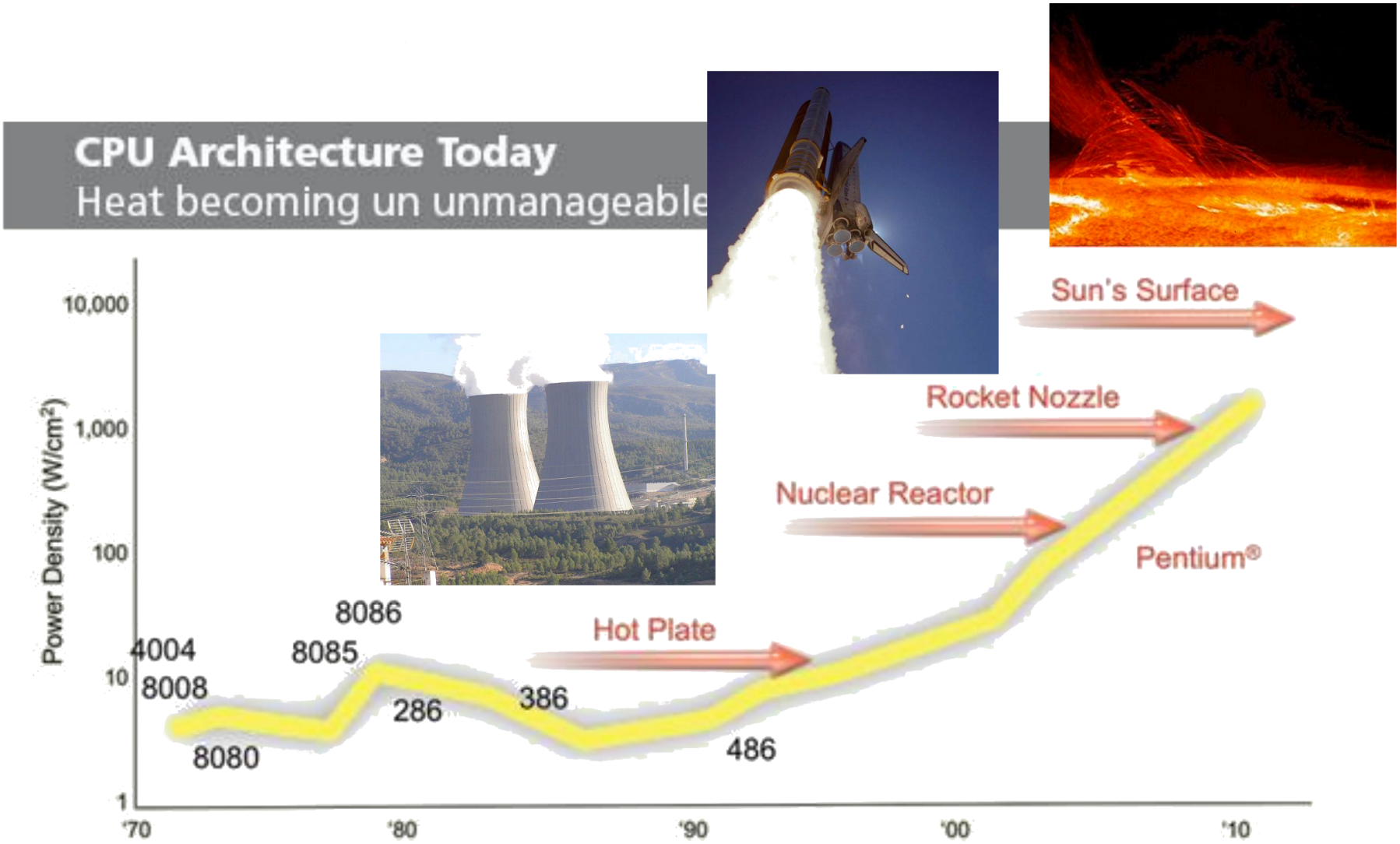
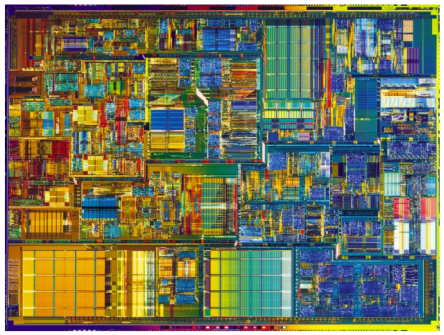


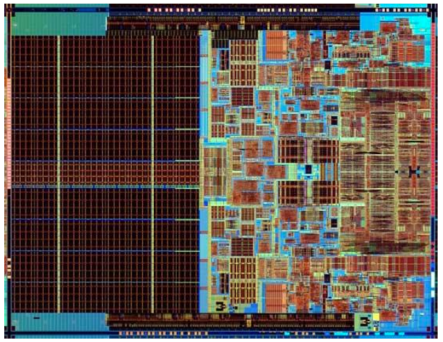
Figure 1. In CPU architecture today, heat is becoming an unmanageable problem. (Courtesy of Pat Gelsinger, Intel Developer Forum, Spring 2004)

The Rise of Parallelism

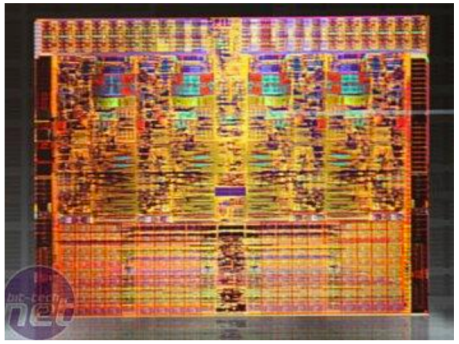
- Multi-processors
  - If one CPU is fast, two must be faster!
  - They allow you to (in theory) double performance without changing the clock speed.
- Seems simple, so why is it becoming important now?
  - Speeding up a single CPU makes everything faster!
    - An application's performance doubles every 18 months with *no effort on the programmer's part*.
  - Getting performance out of multiprocessors requires work!
    - Parallelizing code is *difficult*, it takes (lots of) work. (Take Parallel Programming if you have a chance so you can learn how hard it is.)



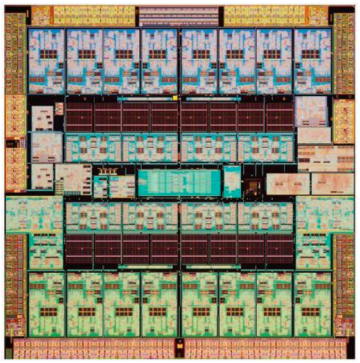
Intel P4
(2000)
1 core



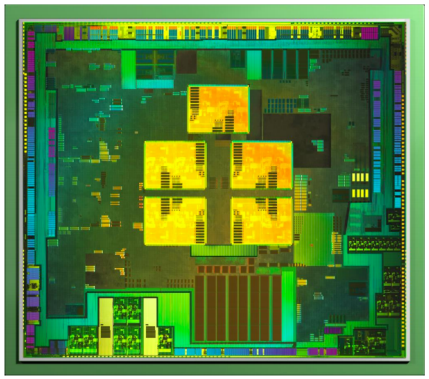
Intel Core 2 Duo
(2006)
2 cores



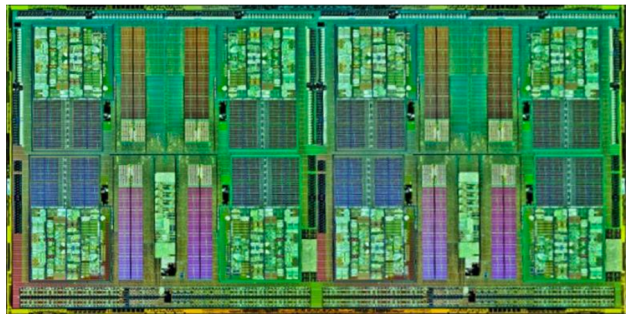
Intel Nahalem
(2010)
4 cores



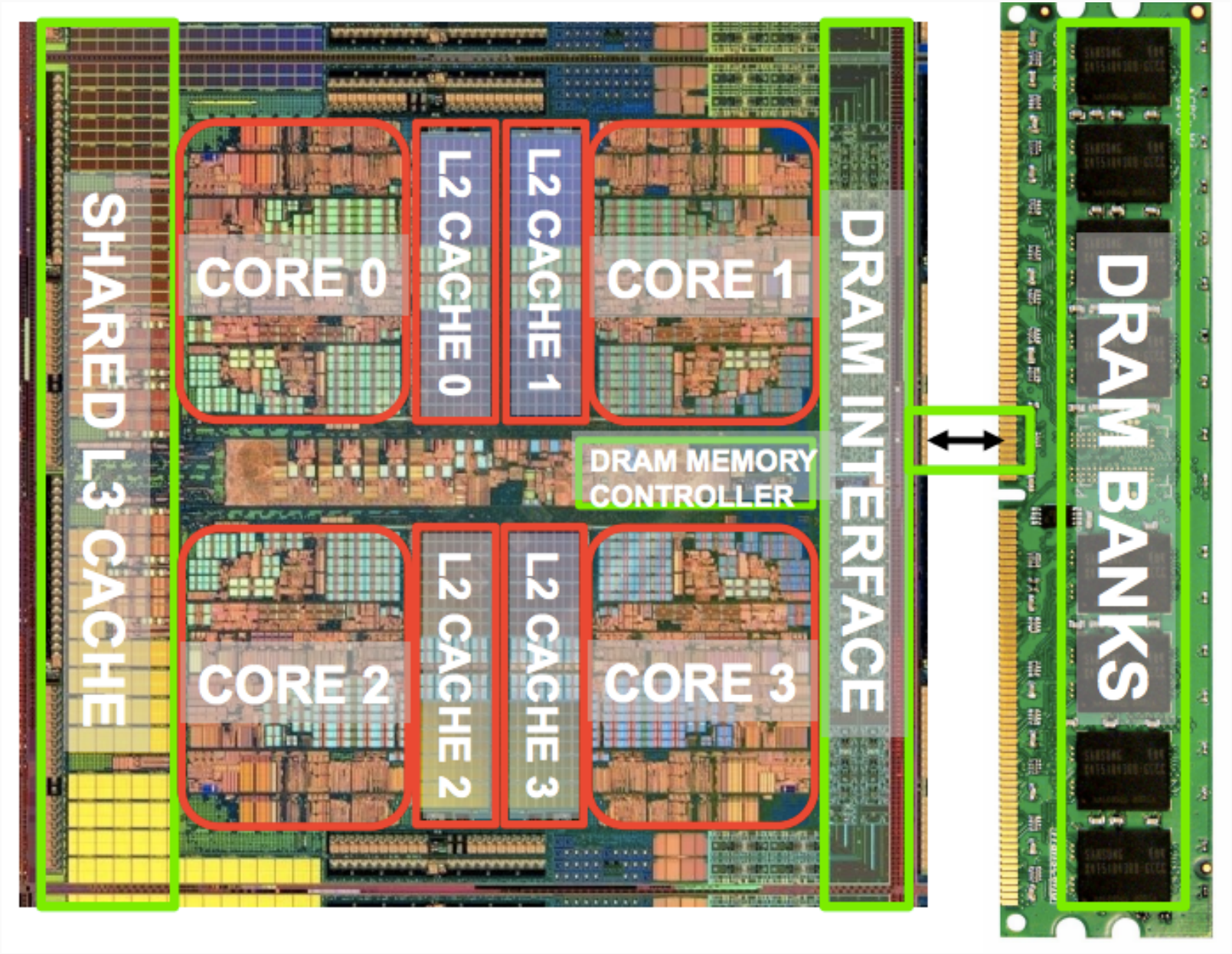
SPARC T3
(2010)
16 cores



Nvidia Tegra 3
(2011)
5 cores



AMD Zambezi
(2011)
16 cores



AMD Barcelona

Power Consumption Trends

- **Dynamic power** is proportional to

$$\text{activity} \times \text{capacitance} \times \text{voltage}^2 \times \text{frequency}$$

- Capacitance per transistor and voltage are decreasing
- Leakage power is rising
- Energy = power × time = (dynpower + leakage power) × time
- Question (true or false): low power design is always good?

Power saving techniques

- Dynamic voltage scaling
  - Good: voltage is a big contributor to power
  - Bad: it hurts performance
  - Application scenarios?
- Shut down the power supply
  - Commonly used in mobile devices
  - Good: obvious
  - Bad: may hurt performance significantly

A sample research problem

Imagine you have a server with 10 cores. Every 0.1s your server receives a request. Each request can be serviced by one core in 0.1s.

You are trying to optimize for power using the techniques discussed, **what kinds of things can you do to reduce power consumption?**

Class logistics

Learning objectives for this class

After taking this class, you will understand all of the major trends shaping computer architecture today. Some of the topics we will discuss include.

- Measuring performance
- Power issues
- ISAs
- Memory hierarchies
- Multi-processor architectures
- Processor pipelines
- Out-of-order execution
- Multithreading
- Storage systems
- Data centers
- GPUs
- Virtual machines
- *Maybe more, maybe less...*

Course Staff

- Instructor: **Sumner Evans** (jonathanevans@mines.edu)
  - Lecture: Mondays and Wednesdays, 16:30 to 17:45 in GC MET
  - Office hours: Mondays and Wednesdays, 18:00 to 19:00 (on Mumble) or by appointment
- TA: **Adam Sandstedt** (asandstedt@mymail.mines.edu)
- Course Website: sumnerevans.com/teaching/csci564-s21
- Matrix Chat: #aca-s21:sumnerevans.com
- Piazza: piazza.com/mines/spring2021/csci564

Academic Honesty

DO NOT CHEAT

- **Do not** copy other people's code
- **Do not** copy solutions during the exams
- **Do not** copy sentences from other papers or web resources
- **Seriously, cheating leads to unhappy consequences!**

What You'll be Graded On

This class is worth **1000 points**.

- Midterm — 100 points
- Final — 200 points
- Homework (4 assignments) — 200 points
- Class participation (worksheets & piazza) — 100 points
  - Not showing up will impact your grade
  - Read the textbook!*
  - I will give out worksheets during class covering what we are discussing in class. You will turn in those worksheets for participation grade. You will also receive some feedback to make sure you are on the right track.
- Projects (3 projects) — 400 points

Projects

- Project 1: Cache Replacement Policies Simulator — 150 points.
- Project 2: Cache Prefetcher Simulator — 100 points.
- Project 3: Branch Prediction Simulator — 150 points.

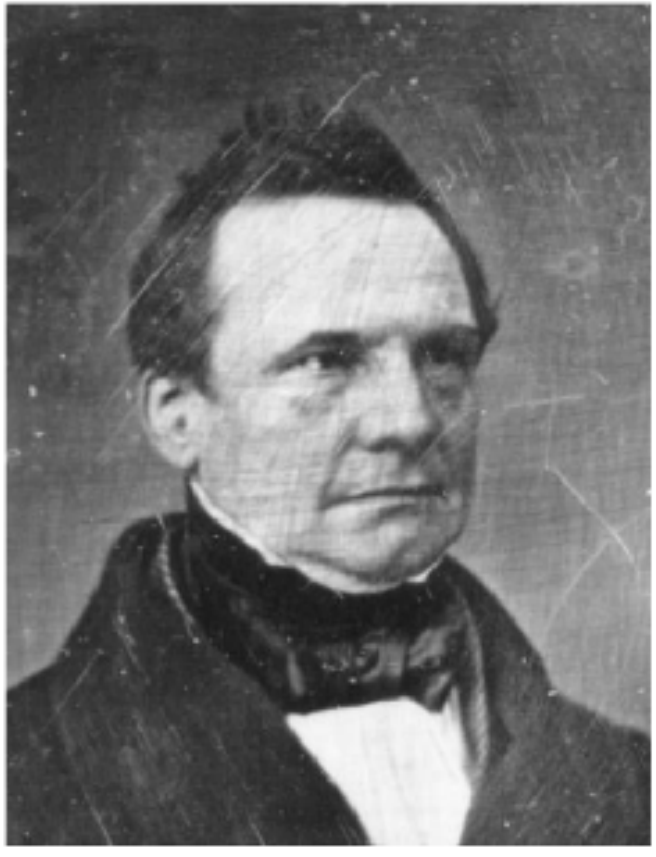
All projects are **individual assignments**. Part of your grade will depend on the code quality.

It is **highly recommended** that you use Linux for all of the projects in this course. If you choose to use a different type of system (such as macOS or Windows), the instructor nor the TAs will not be able to assist you with any platform-specific issues.

A (very) brief history of computing

Charles Babbage 1791-1871

Lucasian Professor of Mathematics,
Cambridge University, 1827-1839
First computer designer



Ada Lovelace 1815-1852

First computer programmer

Difference Engine

- Can compute any 6th degree polynomial
- *Speed:* 33 to 44 32-digit numbers per minute!

Now the machine is at the Smithsonian

Analytic Engine

The first conception of a general purpose computer

1. The *store* in which all variables to be operated upon, as well as all those quantities which have arisen from the results of the operations are placed.
2. The *mill* into which the quantities about to be operated upon are always brought.

An operation in the *mill* required feeding two punched cards and producing a new punched card for the *store.*

An operation to alter the sequence (i.e., a branch) was also provided!

Analytic Engine

1833: Babbage's paper was published
- *conceived during a hiatus in the development of the difference engine*

1871: Babbage dies
- The machine remains unrealized.

- Ada Lovelace gets less credit than she deserves -- She essentially invented programming.

It is not clear if the analytic engine could be built even today using only mechanical technology

Harvard Mark I

- Built in 1944 in IBM Endicott laboratories
  - Howard Aiken – Professor of Physics at Harvard
  - Essentially mechanical but had some electro-magnetically controlled relays and gears
  - Weighed *5 tons* and had *750,000* components
  - A synchronizing clock that beat every *0.015* seconds

Harvard Mark I

- Built in 1944 in IBM Endicott laboratories
  - Howard Aiken – Professor of Physics at Harvard
  - Essentially mechanical but had some electro-magnetically controlled relays and gears
  - Weighed *5 tons* and had *750,000* components
  - A synchronizing clock that beat every *0.015* seconds

Performance:

0.3 seconds for addition
6 seconds for multiplication
1 minute for a sine calculation

Harvard Mark I

- Built in 1944 in IBM Endicott laboratories
  - Howard Aiken – Professor of Physics at Harvard
  - Essentially mechanical but had some electro-magnetically controlled relays and gears
  - Weighed *5 tons* and had *750,000* components
  - A synchronizing clock that beat every *0.015* seconds

Performance:

0.3 seconds for addition
6 seconds for multiplication
1 minute for a sine calculation

Broke down once a week!

Electronic Numerical Integrator and Computer (ENIAC)

- Inspired by Atanasoff and Berry, Eckert and Mauchly designed and built ENIAC (1943-45) at the University of Pennsylvania
- The first, completely electronic, operational, general-purpose analytical calculator!
  - 30 tons, 72 square meters, 200KW
- Performance
  - Read in 120 cards per minute
  - Addition took 200 μs, Division 6 ms
  - 1000 times faster than Mark I
- Not very reliable!

Application: Ballistic calculations

angle = f (location, tail wind, cross wind,
air density, temperature, weight of shell,
propellant charge, ...)

Electronic Discrete Variable Automatic Computer (EDVAC)

- ENIAC's programming system was external
  - –Sequences of instructions were executed independently of the results of the calculation
  - –Human intervention required to take instructions "out of order"
- Eckert, Mauchly, John von Neumann and others designed EDVAC (1944) to solve this problem
  - –Solution was the *stored program computer*
    - ®"*program can be manipulated as data*"

And then there was IBM 701

IBM 701 -- 30 machines were sold in 1953-54

IBM 650 -- more than 120 were sold in 1954
and there were orders for 750 more!
- eventually sold about 2000 of them

Users stopped building their own machines.

Why was IBM late getting into computer technology?

And then there was IBM 701

IBM 701 -- 30 machines were sold in 1953-54

IBM 650 -- more than 120 were sold in 1954
and there were orders for 750 more!
- eventually sold about 2000 of them

Users stopped building their own machines.

Why was IBM late getting into computer technology?

IBM was making too much money!
Even without computers, IBM revenues were doubling every 4 to 5 years in 40's and 50's.

Into the 60's…:

Compatibility Problem at IBM

By early 60's, *IBM had 4 incompatible lines of computers!*

701	□	7094
650	□	7074
702	□	7080
1401	□	7010

Each system had its own

- Instruction set
- Peripherals: magnetic tapes, drums and disks
- Programming tools: assemblers, compilers, libraries,...
- market niche: business, scientific, etc....

IBM 360 : Design Premises

Amdahl, Blaauw and Brooks, 1964

http://www.research.ibm.com/journal/rd/441/amdahl.pdf

- Breaks the link between ***programmer and hardware***
- Upward and downward, machine-language compatibility across a family of machines
- General purpose machine organization, general I/O interfaces,

  storage > 32K
- Easier to use (answers-per-month vs. bits-per-second)
- Machine must be capable of *supervising itself* without manual intervention →OS/360 (simple OS's in IBM 700/7000)
- Built-in *hardware fault checking* and locating aids to reduce down time

… the use of the "ISA" as a compatibility layer was a $175 billion project (2011 dollars)


IBM

IBM 360: Implementation

	Model 30	*Model 70*
Main Storage	8K - 64 KB	256K - 512 KB
Datapath	8-bit	64-bit
Circuit Delay	30 nsec/level	5 nsec/level
Local Store	Main Store	Transistor Registers

IBM 360 instruction set architecture completely hid the underlying technological differences between various models.

With “minor” modifications this is the approach we use today.

CDC6600: Fastest Machine 1964-1969

"Last week, Control Data ... announced the 6600 system. I understand that in the laboratory developing the system there are only 34 people including the janitor. Of these, 14 are engineers and 4 are programmers... Contrasting this modest effort with our vast development activities, I fail to understand why we have lost our industry leadership position by letting someone else offer the world's most powerful computer." -- T.J. Watson, IBM CEO

"It seems like Mr. Watson has answered his own question."

-- Seymour Cray

In combination with some other factors, this explains why researchers can build such cool things!

And why social media startups can sometimes out maneuver Google and Facebook.