UC Berkeley
EECS Lecturer
Pierce Vollucci

The Beauty and Joy of Computing

Lecture #8
Concurrency

HW1+2

MEDICAL INVESTIGATION USING DNA SEQUENCING

The NIH is funding more medical research for undiagnosed diseases and a large part of DNA sequencing is made possible by parallel computing.

http://www.latimes.com/science/sciencenow/la-sci-sn-medical-sleuths-mystery-diseases-offer-20140701-story.html#page=1

FACEBOOK CALLED OUT FOR MANIPULATIVE STUDY

The Electronic Privacy Information Center and others are asking questions about Facebook's studies. Facebook apologizes but says it's allowed and standard practice for companies as research.

http://bits.blogs.nytimes.com/2014/07/03/privacy-group-complains-to-f-t-c-about-facebook-emotion-study/?ref=technology

Concurrency: A Definition

Concurrency: **A** property of computer systems in which several computations are executing simultaneously, and potentially interacting with each other.

Garcia + Vollucci

Concurrency is Everywhere!

Examples:

- Mouse cursor movement while Snap*!* calculates.
- Screen clock advances while typing in a text.
- Busy cursor spins while browser connects to server, waiting for response
- Walking while chewing gum

Garcia + Vollucci

Concurrency & Parallelism

Intra-computer

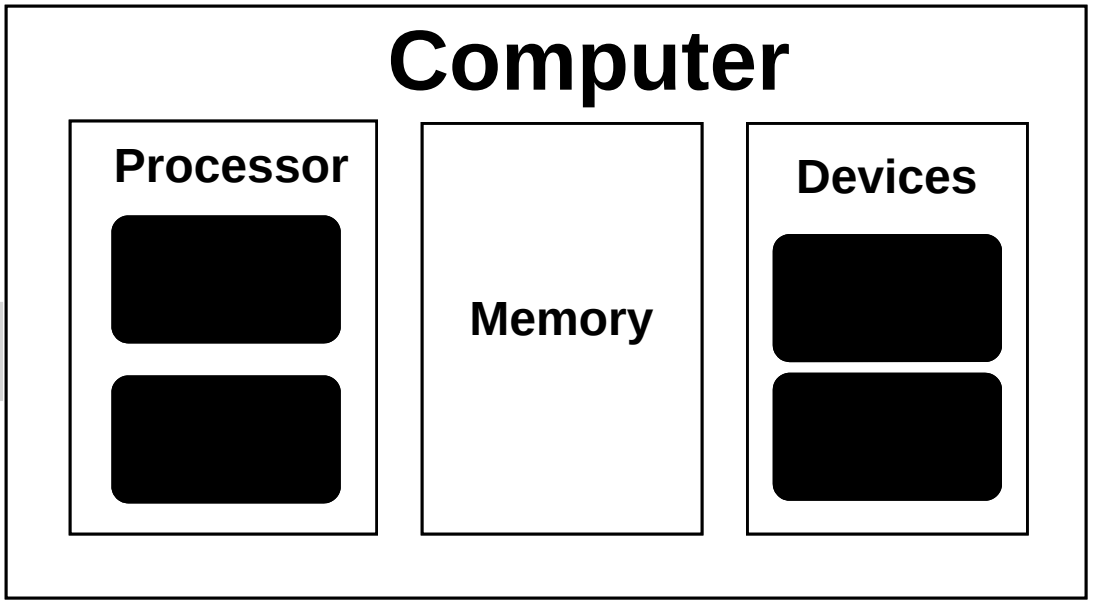
- Today's lecture
- Multiple computing "helpers" are cores within one machine
- Aka "multi-core"
  - Although GPU parallism is also "intra-computer"

Inter-computer

- Future lecture
- Multiple computing "helpers" are different machines
- Aka "distributed computing"
  - Grid & cluster computing

Anatomy: 5 components of any Computer

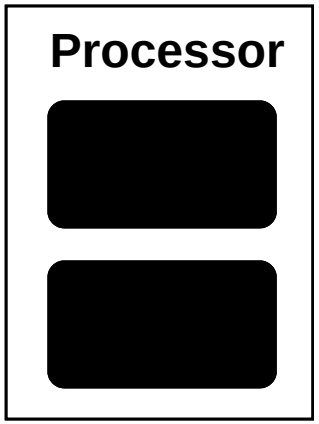
John von Neumann invented this architecture



a) Control
b) Datapath
c) Memory
d) Input
e) Output

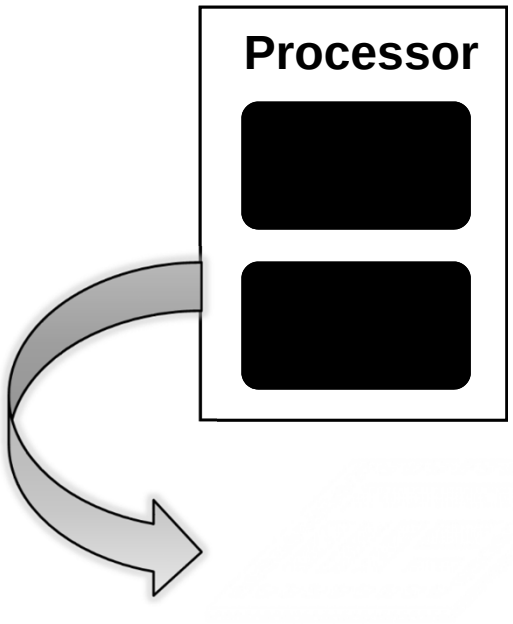
What causes the most headaches for SW and HW designers with multi-core computing?

But what is INSIDE a Processor?



Garcia + Vollucci

But what is INSIDE a Processor?



Bare Processor Die

- Primarily Crystalline Silicon
- 1 mm – 25 mm on a side
- 2009 “feature size” (aka process)
  ~ 45 nm = 45 x 10^{-9} m
  (then 32, 22, and 16 [by yr 2013])
- 100 - 1000M transistors
- 3 - 10 conductive layers
- “CMOS” (complementary metal oxide semiconductor) - most common

- **Package provides:**
  - **spreading of chip-level signal paths to board-level**
  - **heat dissipation.**
- **Ceramic or plastic with gold wires.**

Chip in Package

en.wikipedia.org/wiki/Moore's_law

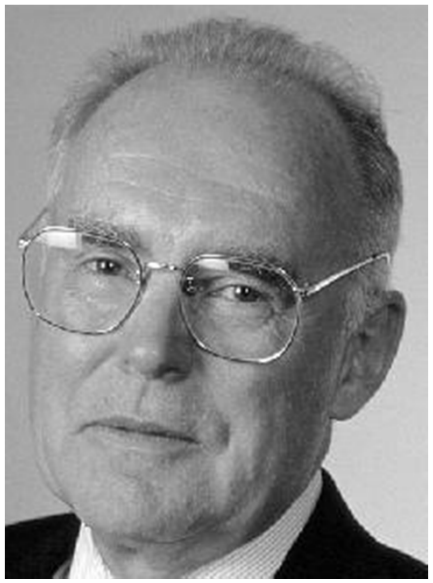
Moore's Law

Predicts: 2X Transistors / chip every 2 years

# of transistors on an integrated circuit (IC)

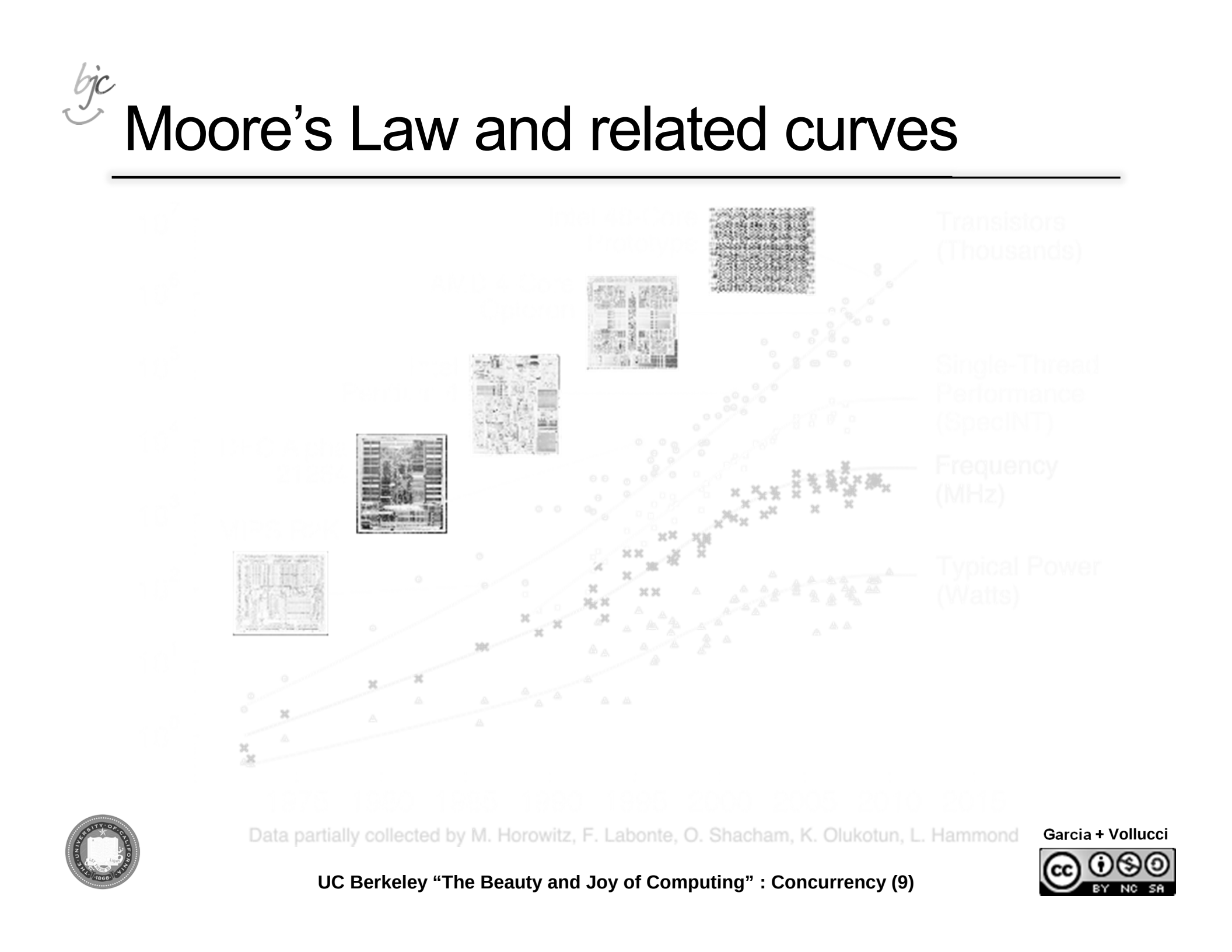
What is this "curve"?

a) Constant
b) Linear
c) Quadratic
d) Cubic
e) Exponential



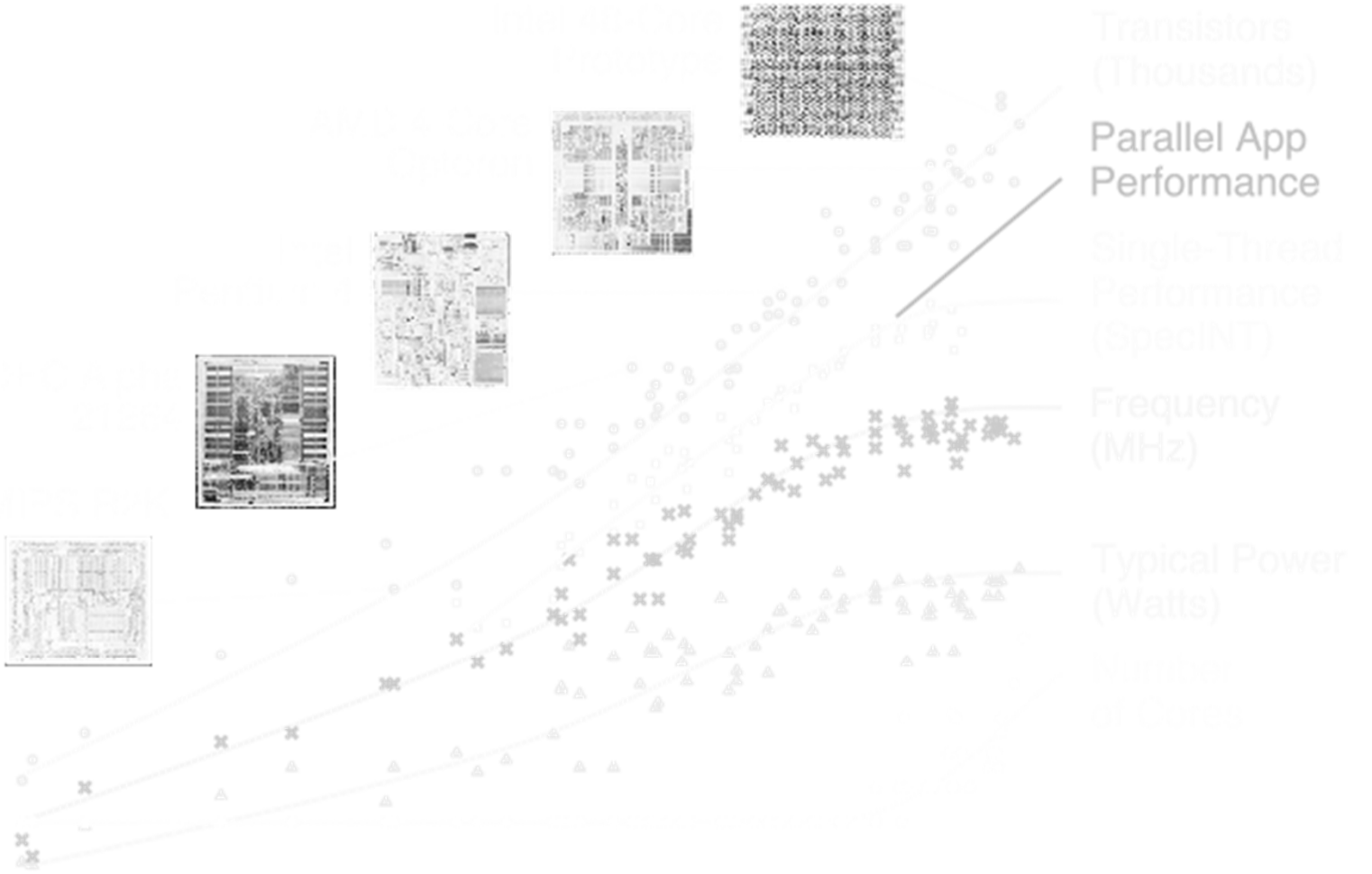
Gordon Moore
Intel Cofounder
B.S. Cal 1950!

Year

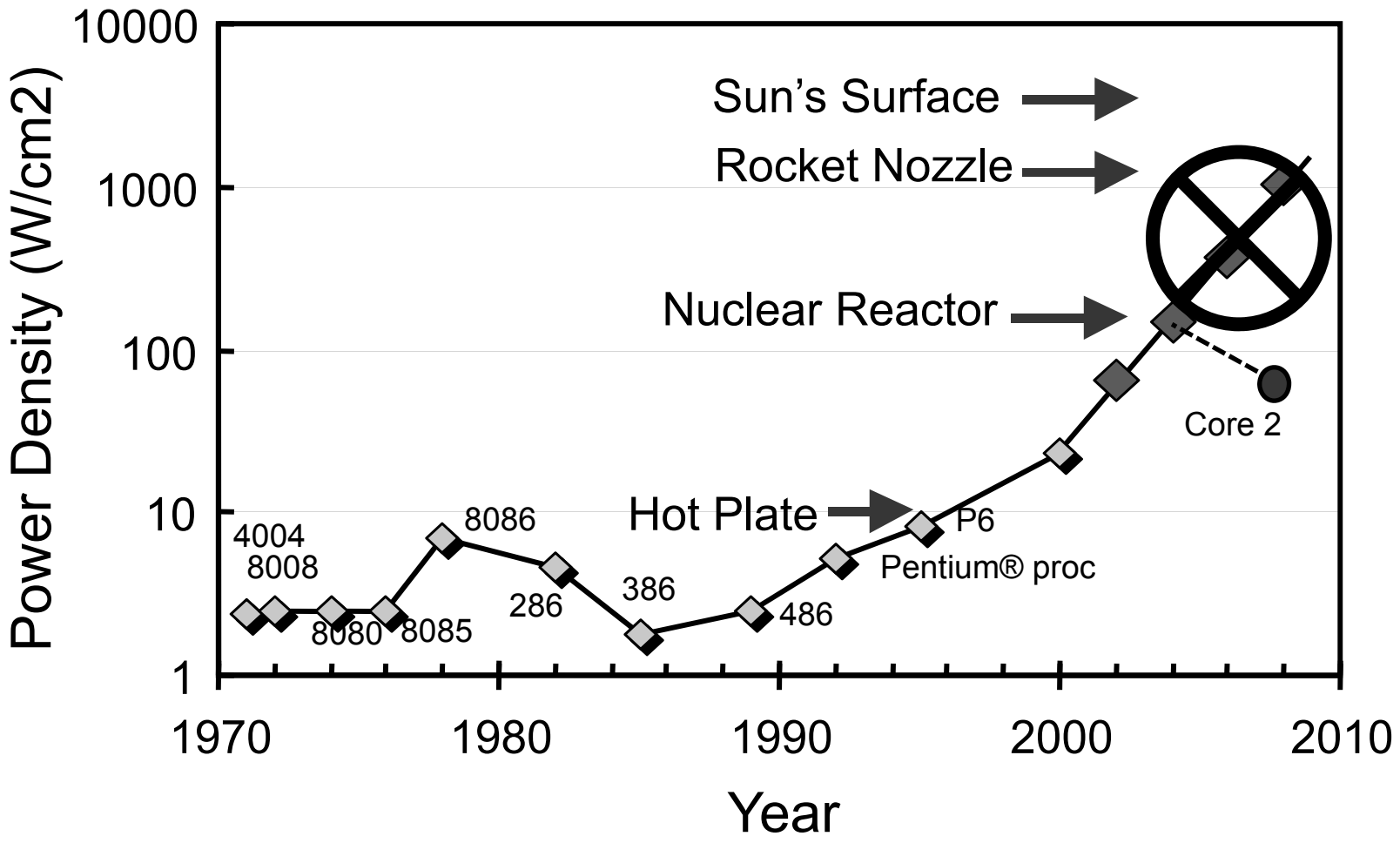


Moore's Law and related curves

Moore's Law and related curves



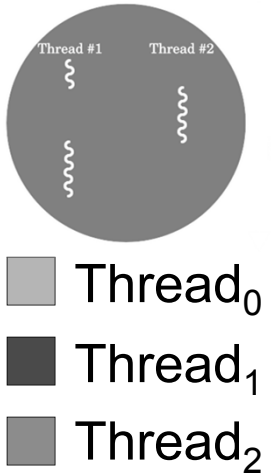
Power Density Prediction circa 2000



Source: S. Borkar (Intel)

Background: Threads

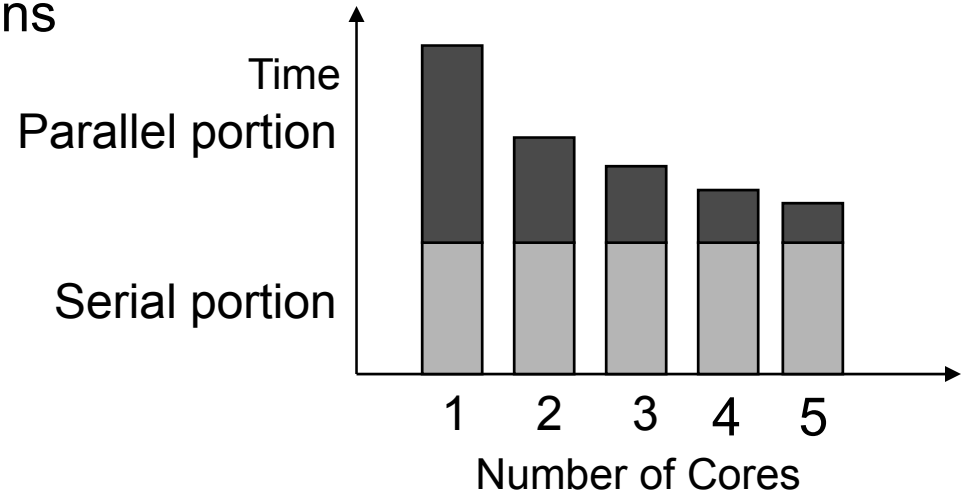
- A *Thread* stands for "thread of execution", is a single stream of instructions
  - A program / process can split, or fork itself into separate threads, which can (in theory) execute simultaneously.
  - An easy way to describe/think about parallelism
- A single CPU can execute many threads by *Time Division Multipexing*



- *Multithreading* is running multiple threads through the same hardware

Speedup Issues : Amdahl's Law

- Applications can almost never be completely parallelized; some serial code remains



- s is serial fraction of program, P is # of cores (was processors)
- **Amdahl's law:**

Speedup(P) = Time(1) / Time(P)

≤ 1 / (s + [(1-s) / P)]

as P ⌧ ∞ ≤ 1 / s

$$\frac{normal\ run\ time}{serial\ portion\ run\ time\ + \frac{parallel\ portion\ run\ time}{number\ of\ cores}}$$

$$\frac{normal\ (1)}{serial\ portion\ (fraction)\ + \frac{parallel\ portion\ (fraction)}{number\ of\ cores}}$$

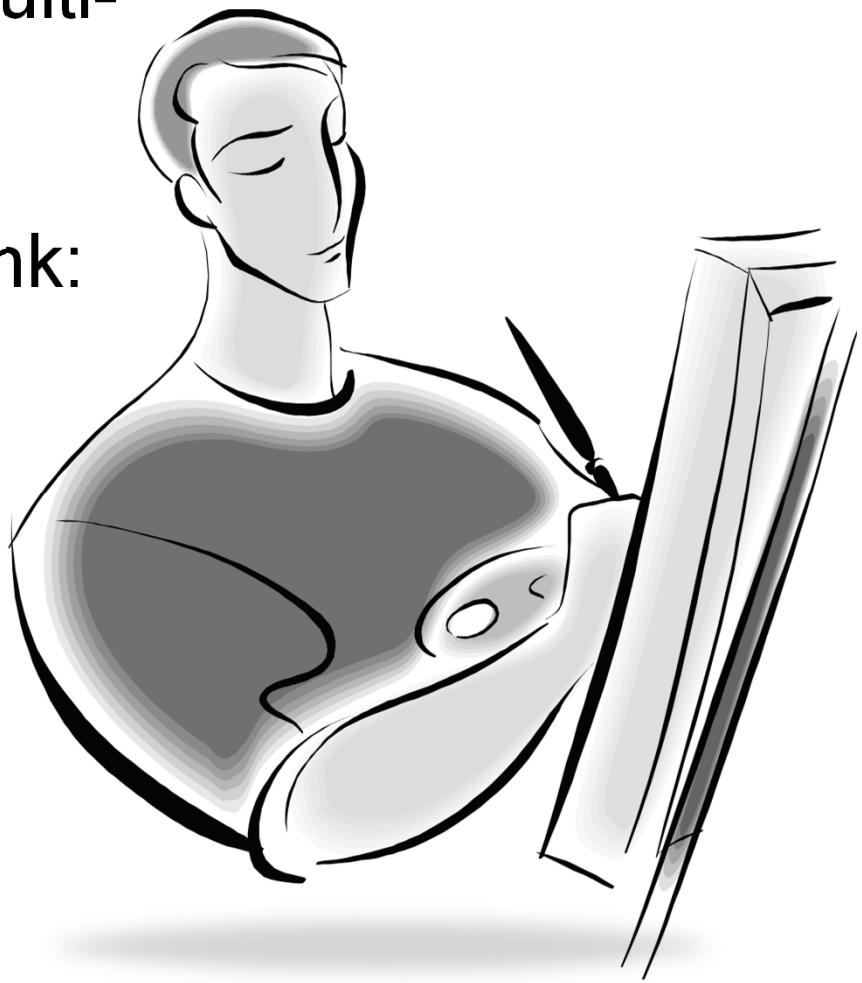
- Even if the parallel portion of your application speeds up perfectly, your performance may be limited by the sequential portion

Speedup Issues : Overhead

- Even assuming no sequential portion, there's…
  - Time to think how to divide the problem up
  - Time to hand out small "work units" to workers
  - All workers may not work equally fast
  - Some workers may fail
  - There may be contention for shared resources
  - Workers could overwriting each others' answers
  - You may have to wait until the last worker returns to proceed (the slowest / weakest link problem)
  - There's time to put the data back together in a way that looks as if it were done by one

Life in a multi-core world…

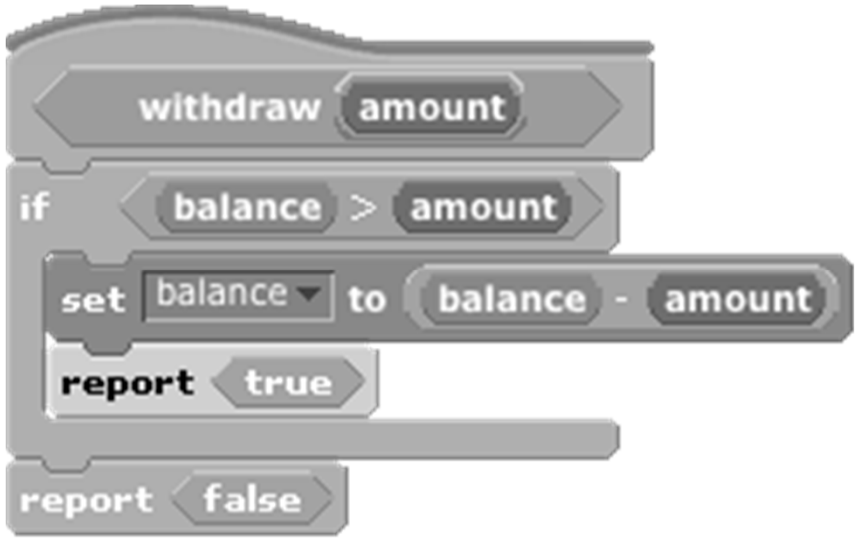
- This "sea change" to multi-core parallelism means that the computing community has to rethink:
  a) Languages
  b) Architectures
  c) Algorithms
  d) Data Structures
  e) All of the above



en.wikipedia.org/wiki/Concurrent_computing

But parallel programming is hard!

- What if two people were calling withdraw at the same time?
  - E.g., balance=100 and two withdraw 75 each
  - Can anyone see what the problem *could* be?
  - This is a race condition
- In most languages, this is a problem.
  - In Scratch, the system doesn't let two of these run at once.



Another concurrency problem … deadlock!

- Two people need to draw a graph but there is only one pencil and one ruler.
  - One grabs the pencil
  - One grabs the ruler
  - Neither release what they hold, waiting for the other to release
- Livelock also possible
  - Movement, no progress

Summary

- "Sea change" of computing because of inability to cool CPUs means we're now in multi-core world
- This brave new world offers lots of potential for innovation by computing professionals, but challenges persist