IBM’s Watson computer (really 2,800 cores) is leading former champions $35,734 to $10,000 and $4,800. Despite a few missteps, it was correct in almost every occasion. It would clearly make a perfect backup consultant for answers like this…
Concurrency & Parallelism, 10 mi up…

Intra-computer

- Today’s lecture
- Multiple computing “helpers” are cores within one machine
- Aka “multi-core”
  - Although GPU parallelism is also “intra-computer”

Inter-computer

- Week 12’s lectures
- Multiple computing “helpers” are different machines
- Aka “distributed computing”
  - Grid & cluster computing
Anatomy: 5 components of any Computer
Anatomy: 5 components of any Computer

John von Neumann invented this architecture

Computer

- Processor
  - Control ("brain")
  - Datapath ("brawn")
- Memory
- Devices
  - Input
  - Output

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

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

UC Berkeley CS10 “The Beauty and Joy of Computing” : Concurrency (4)
But what is INSIDE a Processor?

Processor
- Control ("brain")
- Datapath ("brawn")
But what is INSIDE a Processor?

- Primarily Crystalline Silicon
- 1 mm – 25 mm on a side
- 2009 “feature size” (aka process) 
  \( \sim 45 \text{ nm} = 45 \times 10^{-9} \text{ m} \) (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.

Garcia, Spring 2011
What is this “curve”?

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

Moore’s Law

Predicts: 2X Transistors / chip every 2 years

Curve shows Moore's Law: Transistor count doubling every two years

Gordon Moore
Intel Cofounder
B.S. Cal 1950!
Moore's Law and related curves

Data partially collected by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond

UC Berkeley CS10 “The Beauty and Joy of Computing” : Concurrency (8)
Moore's Law and related curves

Transistors (Thousands)
Parallel App Performance
Single-Thread Performance (SpecINT)
Frequency (MHz)
Typical Power (Watts)
Number of Cores

Data partially collected by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond

Garcia, Spring 2011
Power Density Prediction circa 2000

- Sun’s Surface
- Rocket Nozzle
- Nuclear Reactor
- Hot Plate
- Pentium® proc
- Core 2

Source: S. Borkar (Intel)
Going Multi-core Helps Energy Efficiency

- Power of typical integrated circuit $\sim CV^2f$
  - $C =$ Capacitance, how well it "stores" a charge
  - $V =$ Voltage
  - $f =$ frequency. I.e., how fast clock is (e.g., 3 GHz)

In the same process technology...

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltage</td>
<td>1</td>
<td>Voltage</td>
<td>-15%</td>
</tr>
<tr>
<td>Freq</td>
<td>1</td>
<td>Freq</td>
<td>-15%</td>
</tr>
<tr>
<td>Area</td>
<td>1</td>
<td>Area</td>
<td>2</td>
</tr>
<tr>
<td>Power</td>
<td>1</td>
<td>Power</td>
<td>1</td>
</tr>
<tr>
<td>Perf</td>
<td>1</td>
<td>Perf</td>
<td>$\sim$1.8</td>
</tr>
</tbody>
</table>

Activity Monitor (on the lab Macs) shows how active your cores are

William Holt, HOT Chips 2005
Energy & Power Considerations

![Image of electronics]

Power = \frac{\text{Energy}}{\text{Second}} = \frac{\text{Energy}}{\text{Op}} \times \frac{\text{Ops}}{\text{Second}}

- **Power**
  - Chip Packaging
  - Chip Cooling
  - System Noise
  - Case Temperature
  - Data-Center Air Conditioning

- **Energy**
  - Battery Life
  - Electricity Bill
  - Mobile Device Weight

100W Workstation Power Constraint
1W Handheld Power Constraint

Energy per Operation vs. Performance (Ops/Second)

**Courtesy: Chris Batten**

UC Berkeley CS10 “The Beauty and Joy of Computing”: Concurrency (12)
Parallelism again? What’s different this time?

“This shift toward increasing parallelism is not a triumphant stride forward based on breakthroughs in novel software and architectures for parallelism; instead, this plunge into parallelism is actually a retreat from even greater challenges that thwart efficient silicon implementation of traditional uniprocessor architectures.”

– Berkeley View, December 2006

- HW/SW Industry bet its future that breakthroughs will appear before it’s too late
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 Multiplexing**

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

$$\text{Speedup}(P) = \frac{\text{Time}(1)}{\text{Time}(P)} \leq \frac{1}{(s + \frac{(1-s)}{P})}, \text{ and as } P \to \infty$$

$$\leq \frac{1}{s}$$

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
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
  - Dan and Luke demo
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