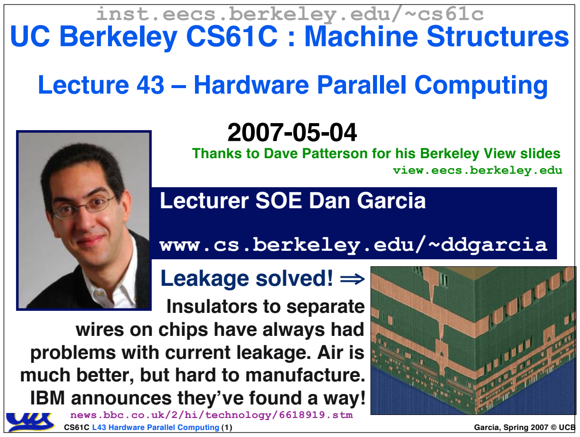


inst.eecs.berkeley.edu/~cs61c

UC Berkeley CS61C : Machine Structures

Lecture 43 – Hardware Parallel Computing

2007-05-04

Thanks to Dave Patterson for his Berkeley View slides
view.eecs.berkeley.edu

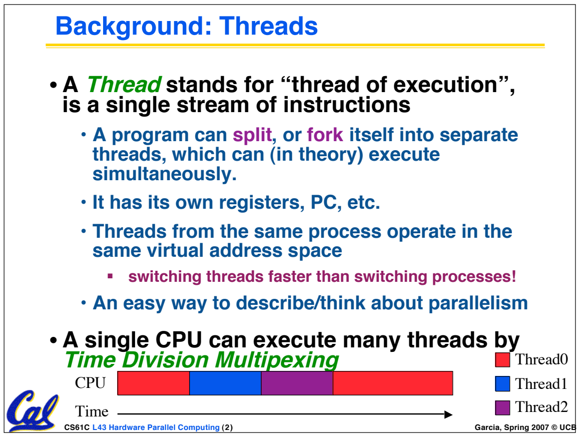
Lecturer SOE Dan Garcia

www.cs.berkeley.edu/~ddgarcia

Leakage solved! ⇒

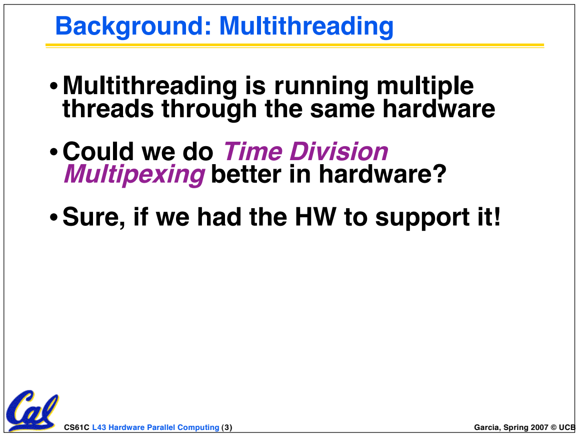
Insulators to separate wires on chips have always had problems with current leakage. Air is much better, but hard to manufacture. IBM announces they've found a way!

news.bbc.co.uk/2/hi/technology/6618919.stm



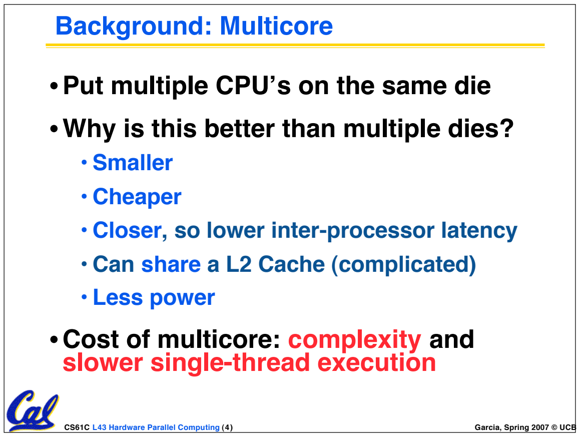
Background: Threads

- A *Thread* stands for "thread of execution", is a single stream of instructions
  - A program can split, or fork itself into separate threads, which can (in theory) execute simultaneously.
  - It has its own registers, PC, etc.
  - Threads from the same process operate in the same virtual address space
    - switching threads faster than switching processes!
  - An easy way to describe/think about parallelism
- A single CPU can execute many threads by *Time Division Multipexing*



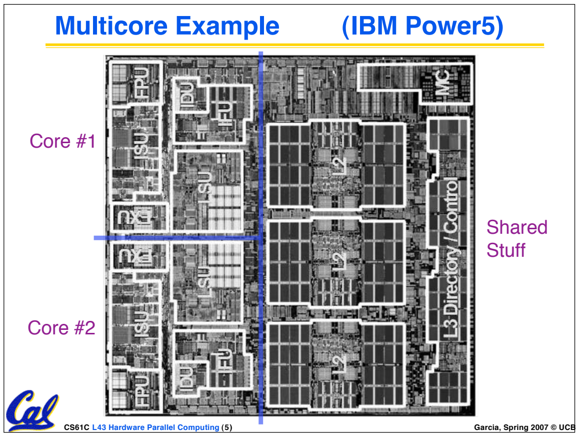
Background: Multithreading

- Multithreading is running multiple threads through the same hardware
- Could we do *Time Division Multipexing* better in hardware?
- Sure, if we had the HW to support it!

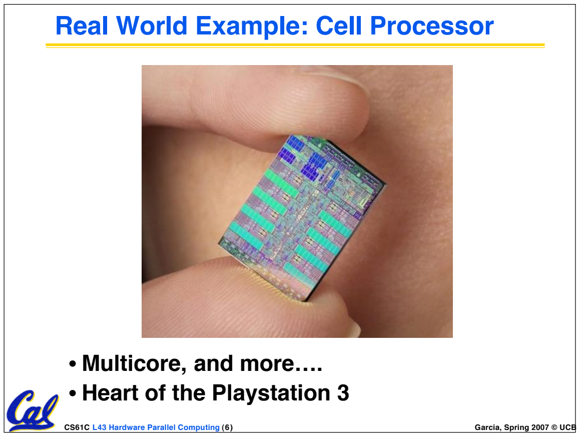


Background: Multicore

- Put multiple CPU's on the same die
- Why is this better than multiple dies?
  - Smaller
  - Cheaper
  - Closer, so lower inter-processor latency
  - Can share a L2 Cache (complicated)
  - Less power
- Cost of multicore: complexity and slower single-thread execution



Multicore Example (IBM Power5)



Real World Example: Cell Processor

- Multicore, and more….
- Heart of the Playstation 3

Real World Example 1: Cell Processor

- 9 Cores (1PPE, 8SPE) at 3.2GHz
- Power Processing Element (PPE)
  - Supervises all activities, allocates work
  - Is multithreaded (2 threads)
- Synergystic Processing Element (SPE)
  - Where work gets done
  - Very Superscalar
  - No Cache, only "Local Store"

Peer Instruction

A. The majority of PS3's processing power comes from the Cell processor

B. Berkeley profs believe multicore is the future of computing

C. Current multicore techniques can scale well to many (32+) cores

	ABC
0:	FFF
1:	FFT
2:	FTF
3:	FTT
4:	TFF
5:	TFT
6:	TTF
7:	TTT

Upcoming Calendar

Week #	Mon	Wed	Thu Lab	Fri
#15 This week	Re-configurable Computing (Michael)	Parallel Computing in Software (Matt)	Parallel? I/O Networking & 61C Feedback Survey	Parallel Computing in Hardware (Dan's last OH 3pm)
#16 Last week o' classes	LAST CLASS Summary, Review, & HKN Evals	Perf comp due Tues Wed 2pm Review 10 Evans		

FINAL EXAM Sat 2007-05-12 @ 12:30pm-3:30pm 2050 VLSB

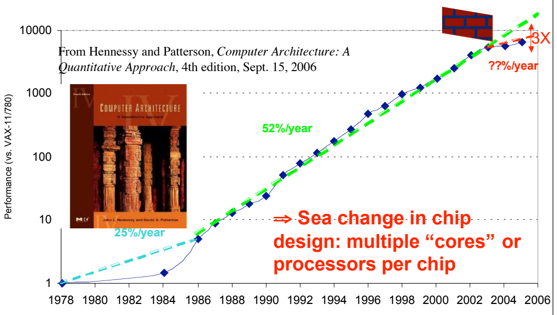
High Level Message

- Everything is changing
- Old conventional wisdom is out
- We desperately need new approach to HW and SW based on parallelism since industry has bet its future that parallelism works
- Need to create a "watering hole" to bring everyone together to quickly find that solution
  - architects, language designers, application experts, numerical analysts, algorithm designers, programmers, …

Conventional Wisdom (CW) in Computer Architecture

1. Old CW: Power is free, but transistors expensive
- New CW: Power wall Power expensive, transistors "free"
  - Can put more transistors on a chip than have power to turn on
2. Old CW: Multiplies slow, but loads fast
- New CW: Memory wall Loads slow, multiplies fast
  - 200 clocks to DRAM, but even FP multiplies only 4 clocks
3. Old CW: More ILP via compiler / architecture innovation
  - Branch prediction, speculation, Out-of-order execution, VLIW, …
- New CW: ILP wall Diminishing returns on more ILP
4. Old CW: 2X CPU Performance every 18 months
- New CW is Power Wall + Memory Wall + ILP Wall = Brick Wall

Uniprocessor Performance (SPECint)

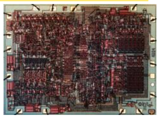


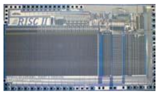
From Hennessy and Patterson, *Computer Architecture: A Quantitative Approach*, 4th edition, Sept. 15, 2006

⇒ Sea change in chip design: multiple "cores" or processors per chip

- VAX : 25%/year 1978 to 1986
- RISC + x86: 52%/year 1986 to 2002
- RISC + x86: ??%/year 2002 to present

Sea Change in Chip Design





- Intel 4004 (1971): 4-bit processor, 2312 transistors, 0.4 MHz, 10 micron PMOS, 11 mm^2 chip
- RISC II (1983): 32-bit, 5 stage pipeline, 40,760 transistors, 3 MHz, 3 micron NMOS, 60 mm^2 chip
- 125 mm^2 chip, 0.065 micron CMOS = 2312 RISC II+FPU+Icache+Dcache
  - RISC II shrinks to ≈ 0.02 mm^2 at 65 nm
  - Caches via DRAM or 1 transistor SRAM or 3D chip stacking
  - Proximity Communication via capacitive coupling at > 1 TB/s ? (Ivan Sutherland @ Sun / Berkeley)
- Processor is the new transistor!

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

view.eecs.berkeley.edu

Need a New Approach

- Berkeley researchers from many backgrounds met between February 2005 and December 2006 to discuss parallelism
  - Circuit design, computer architecture, massively parallel computing, computer-aided design, embedded hardware and software, programming languages, compilers, scientific programming, and numerical analysis
- Krste Asanovic, Ras Bodik, Jim Demmel, John Kubiatowicz, Edward Lee, George Necula, Kurt Keutzer, Dave Patterson, Koshik Sen, John Shalf, Kathy Yelick + others
- Tried to learn from successes in embedded and high performance computing
- Led to 7 Questions to frame parallel research

7 Questions for Parallelism

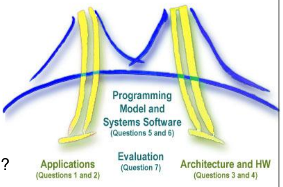
- Applications:

  1. What are the apps?
  2. What are kernels of apps?
- Hardware:

  3. What are HW building blocks?
  4. How to connect them?
- Programming Model & Systems Software:

  5. How to describe apps & kernels?
  6. How to program the HW?
- Evaluation:

  7. How to measure success?


Programming Model and Systems Software (Questions 5 and 6)
Applications (Questions 1 and 2)
Evaluation (Question 7)
Architecture and HW (Questions 3 and 4)

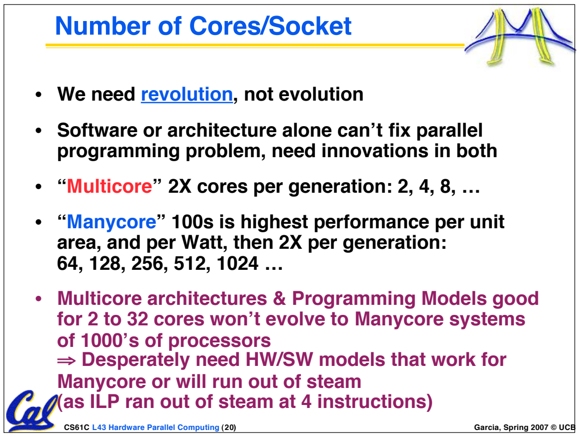
(Inspired by a view of the Golden Gate Bridge from Berkeley)

Hardware Tower: What are the problems?

- Power limits leading edge chip designs
  - Intel Tejas Pentium 4 cancelled due to power issues
- Yield on leading edge processes dropping dramatically
  - IBM quotes yields of 10 – 20% on 8-processor Cell
- Design/validation leading edge chip is becoming unmanageable
  - Verification teams > design teams on leading edge processors

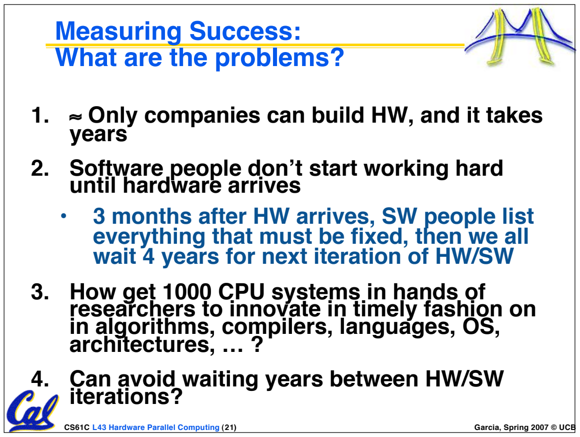
HW Solution: Small is Beautiful

- Expect modestly pipelined (5- to 9-stage) CPUs, FPUs, vector, Single Inst Multiple Data (SIMD) Processing Elements (PEs)
  - Small cores not much slower than large cores
- Parallel is energy efficient path to performance: $P \approx V^2$
  - Lower threshold and supply voltages lowers energy per op
- Redundant processors can improve chip yield
  - Cisco Metro 188 CPUs + 4 spares; Sun Niagara sells 6 or 8 CPUs
- Small, regular processing elements easier to verify
- One size fits all?
  - Amdahl's Law ⇒ Heterogeneous processors?



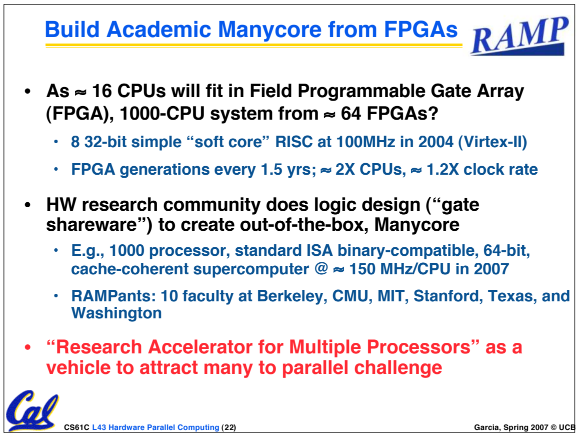
Number of Cores/Socket

- We need revolution, not evolution
- Software or architecture alone can't fix parallel programming problem, need innovations in both
- "Multicore" 2X cores per generation: 2, 4, 8, …
- "Manycore" 100s is highest performance per unit area, and per Watt, then 2X per generation: 64, 128, 256, 512, 1024 …
- Multicore architectures & Programming Models good for 2 to 32 cores won't evolve to Manycore systems of 1000's of processors
  ⇒ Desperately need HW/SW models that work for Manycore or will run out of steam (as ILP ran out of steam at 4 instructions)



Measuring Success: What are the problems?

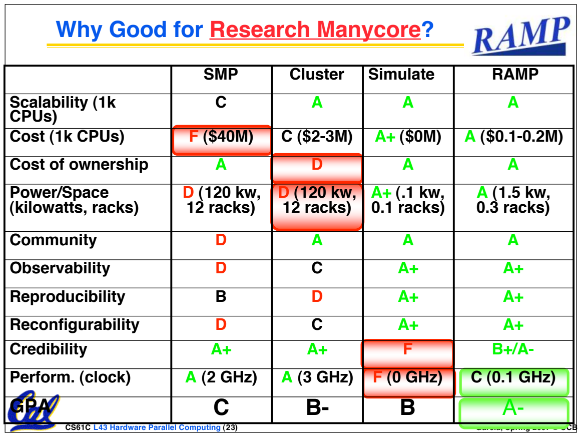
1. ≈ Only companies can build HW, and it takes years
2. Software people don't start working hard until hardware arrives
   - 3 months after HW arrives, SW people list everything that must be fixed, then we all wait 4 years for next iteration of HW/SW
3. How get 1000 CPU systems in hands of researchers to innovate in timely fashion on in algorithms, compilers, languages, OS, architectures, … ?
4. Can avoid waiting years between HW/SW iterations?



Build Academic Manycore from FPGAs

RAMP

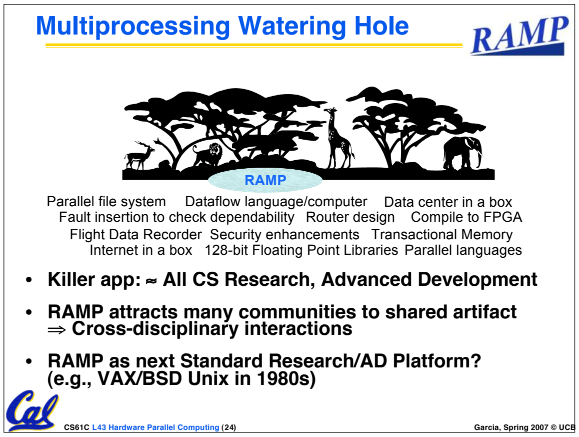
- As ≈ 16 CPUs will fit in Field Programmable Gate Array (FPGA), 1000-CPU system from ≈ 64 FPGAs?
  - 8 32-bit simple "soft core" RISC at 100MHz in 2004 (Virtex-II)
  - FPGA generations every 1.5 yrs; ≈ 2X CPUs, ≈ 1.2X clock rate
- HW research community does logic design ("gate shareware") to create out-of-the-box, Manycore
  - E.g., 1000 processor, standard ISA binary-compatible, 64-bit, cache-coherent supercomputer @ ≈ 150 MHz/CPU in 2007
  - RAMPants: 10 faculty at Berkeley, CMU, MIT, Stanford, Texas, and Washington
- "Research Accelerator for Multiple Processors" as a vehicle to attract many to parallel challenge



Why Good for Research Manycore?

RAMP

	SMP	Cluster	Simulate	RAMP
Scalability (1k CPUs)	C	A	A	A
Cost (1k CPUs)	F ($40M)	C ($2-3M)	A+ ($0M)	A ($0.1-0.2M)
Cost of ownership	A	D	A	A
Power/Space (kilowatts, racks)	D (120 kw, 12 racks)	D (120 kw, 12 racks)	A+ (.1 kw, 0.1 racks)	A (1.5 kw, 0.3 racks)
Community	D	A	A	A
Observability	D	C	A+	A+
Reproducibility	B	D	A+	A+
Reconfigurability	D	C	A+	A+
Credibility	A+	A+	F	B+/A-
Perform. (clock)	A (2 GHz)	A (3 GHz)	F (0 GHz)	C (0.1 GHz)
GPA	C	B-	B	A-

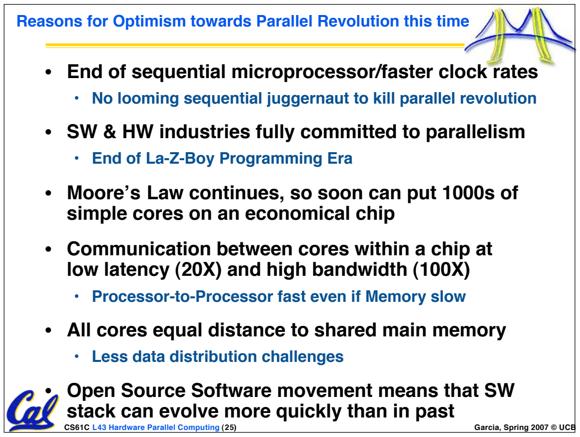


Multiprocessing Watering Hole

RAMP

Parallel file system Dataflow language/computer Data center in a box
Fault insertion to check dependability Router design Compile to FPGA
Flight Data Recorder Security enhancements Transactional Memory
Internet in a box 128-bit Floating Point Libraries Parallel languages

- Killer app: ≈ All CS Research, Advanced Development
- RAMP attracts many communities to shared artifact ⇒ Cross-disciplinary interactions
- RAMP as next Standard Research/AD Platform? (e.g., VAX/BSD Unix in 1980s)



Reasons for Optimism towards Parallel Revolution this time

- End of sequential microprocessor/faster clock rates
  - No looming sequential juggernaut to kill parallel revolution
- SW & HW industries fully committed to parallelism
  - End of La-Z-Boy Programming Era
- Moore's Law continues, so soon can put 1000s of simple cores on an economical chip
- Communication between cores within a chip at low latency (20X) and high bandwidth (100X)
  - Processor-to-Processor fast even if Memory slow
- All cores equal distance to shared main memory
  - Less data distribution challenges
- Open Source Software movement means that SW stack can evolve more quickly than in past