BERKELEY LAB

LAWRENCE BERKELEY NATIONAL LABORATORY

U.S. DEPARTMENT OF ENERGY

Green Flash: Designing Energy Efficient Computers for Cloud System Resolving Climate Models and Other Exascale Applications

Michael Wehner

John Shalf, Lenny Oliker, David Donofrio, Tony Drummond,
Shoaib Kamil,
Norman Miller, Marghoob Mohiyuddin,
Woo-Sun Yang, Kathy Yelick

Lawrence Berkeley National Laboratory

http://www.lbl.gov/CS/html/greenflash.html

Exascale Computing is a Critical Resource

"…exascale computing will *revolutionize our approaches to global challenges in energy, environmental sustainability, and security.*"

Simulation & Modeling at the Exascale for Energy & the Environment **– DOE E3 report**

Global Cloud System Resolving Climate Modeling

Individual cloud physics fairly well understood

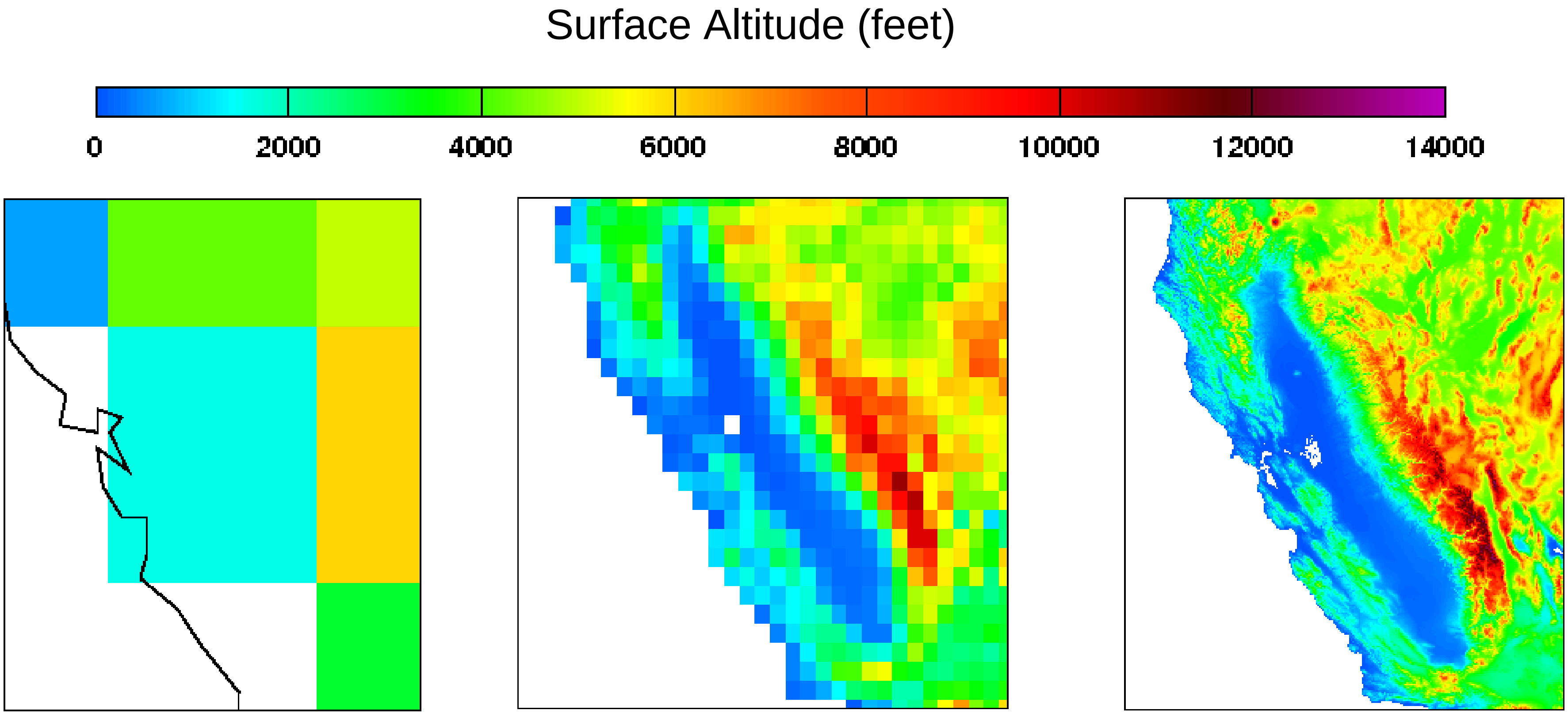
Parameterization of mesoscale cloud statistics performs poorly.

Direct simulation of cloud systems in global models requires exascale!

- Direct simulation of cloud systems replacing statistical parameterization.
  - This approach recently was called for by the 1st WMO Modeling Summit.

- Championed by Prof. Dave Randall, Colorado State University

Global Cloud System Resolving Models are a Transformational Change



200km
Typical resolution of
IPCC AR4 models

25km
Upper limit of climate models
with cloud parameterizations

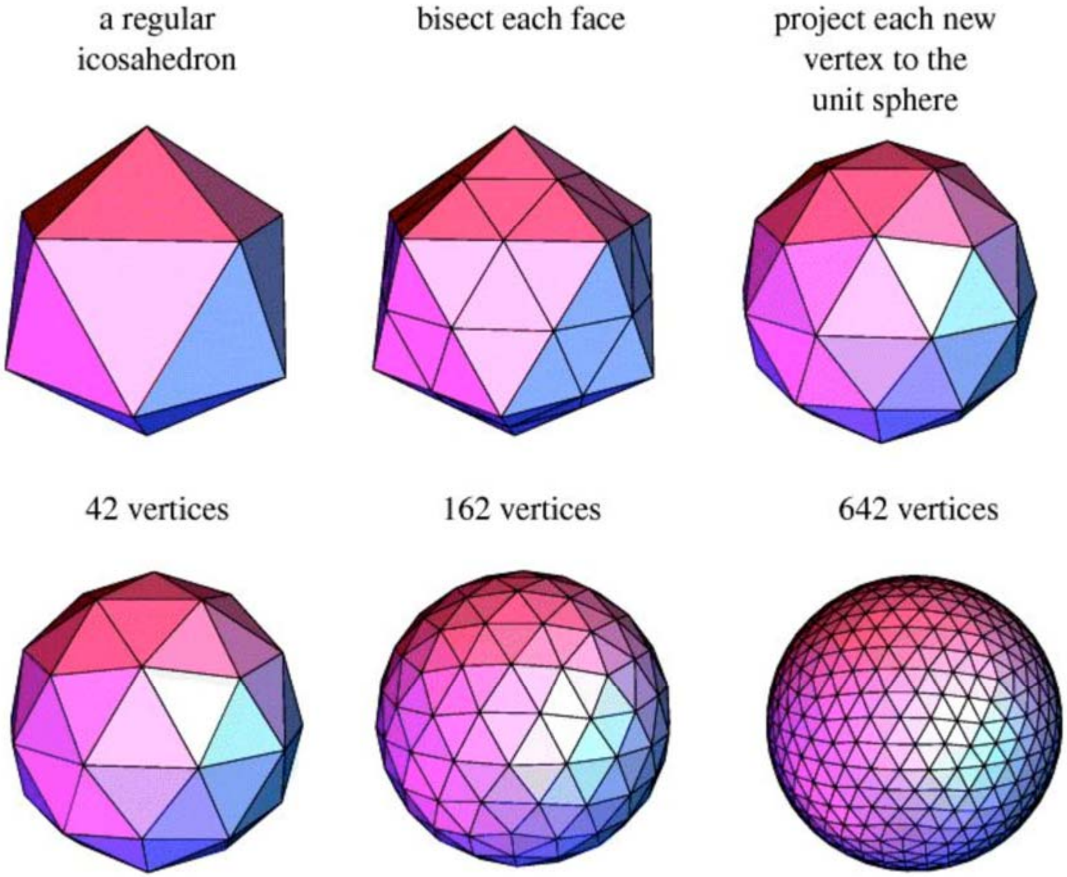
1km
Cloud system resolving models

How expensive is a GCSRM?

- GCSRM: Global Cloud System Resolving Model
  - A complete one does not actually exist…
- Build a model to quantify code requirements by measuring and extrapolating the parts
- Four parts for the atmospheric model:
  - Dynamics
  - Fast physics (cloud processes and turbulence)
  - Slow physics (radiation transport)
  - MultiGrid solver (elliptic equation solution)
- Code requirements model will predict necessary flops, memory, communication, memory i/o to achieve the throughput goals.
  - Target is to simulate time 1000 times faster than real time.

CSU atmospheric model

- **Target resolution is 167,772,162 vertices, ~128 vertical levels, ~1.75 km**



Ross Heikes CSU

Code Requirements Model

- Measure and extrapolate:
  - Operation count
  - Main memory footprint
  - Cache memory footprint (local store, not cache coherent)
  - Memory bandwidth (bytes/flop)
  - Instruction mix
  - Interconnect bandwidth
  - Interconnect latency
  - Interconnect topology
- Derived constraints
  - Power (core + memory+interconnect)
  - Pins (memory + interconnect)
  - Mix of instruction in hardware (Flops, integer ops , branch, etc)

CSU atmospheric model

- 167,772,162 vertices, ~128 vertical levels, ~1.75 km
  - –A truly transformational change to climate change modeling
  - –12.6+ Pflops <u>*sustained* </u>(for 1000x speedup)
  - –560TB total memory

- Ensembles of simulations (~10) → 100 Pflops sustained
- Climate codes typically run at 5% of peak or less

2 Exaflops

(or its equivalent)

Traditional Sources of Performance Improvement are Flat-Lining (2004)

New Constraints

- 15 years of *exponential* clock rate growth has ended

Moore’s Law reinterpreted:

- How to leverage transistors to increase performance at historical rates?
- Power is the new design constraint.
    - Nonlinear: CPU speed & size
- Multicore: # cores double 18-24 months.

Accelerating supercomputing demand:

- End of straightforward serial improvements
- Much higher parallelism will be required to exploit this technology

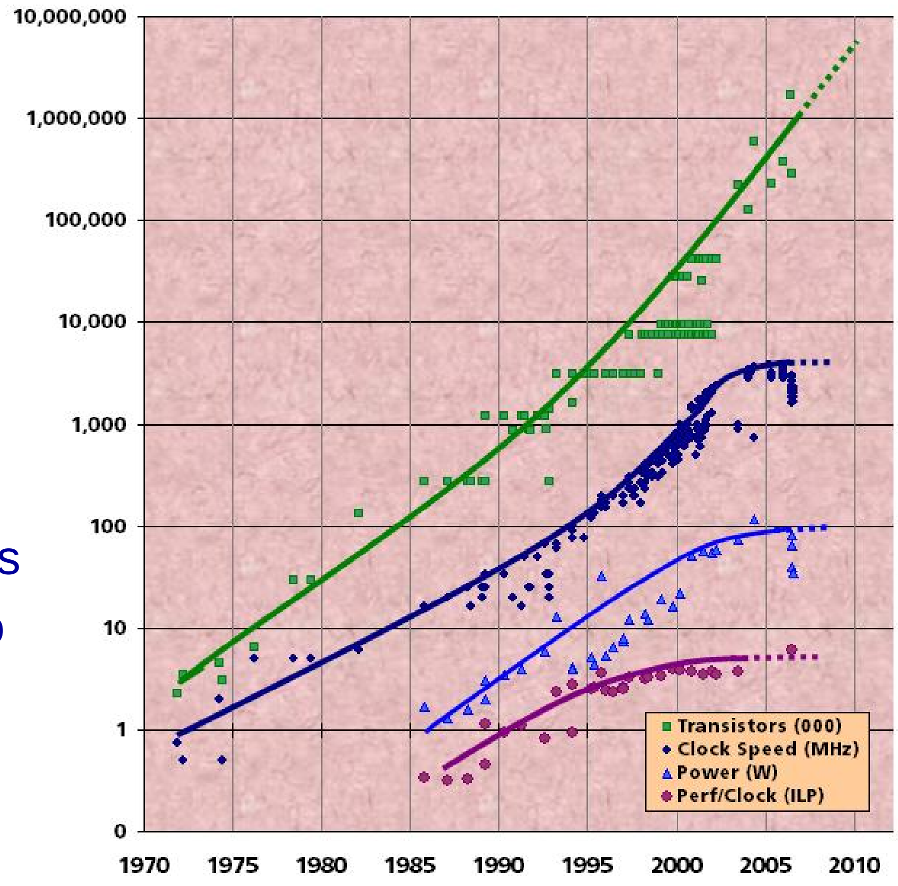
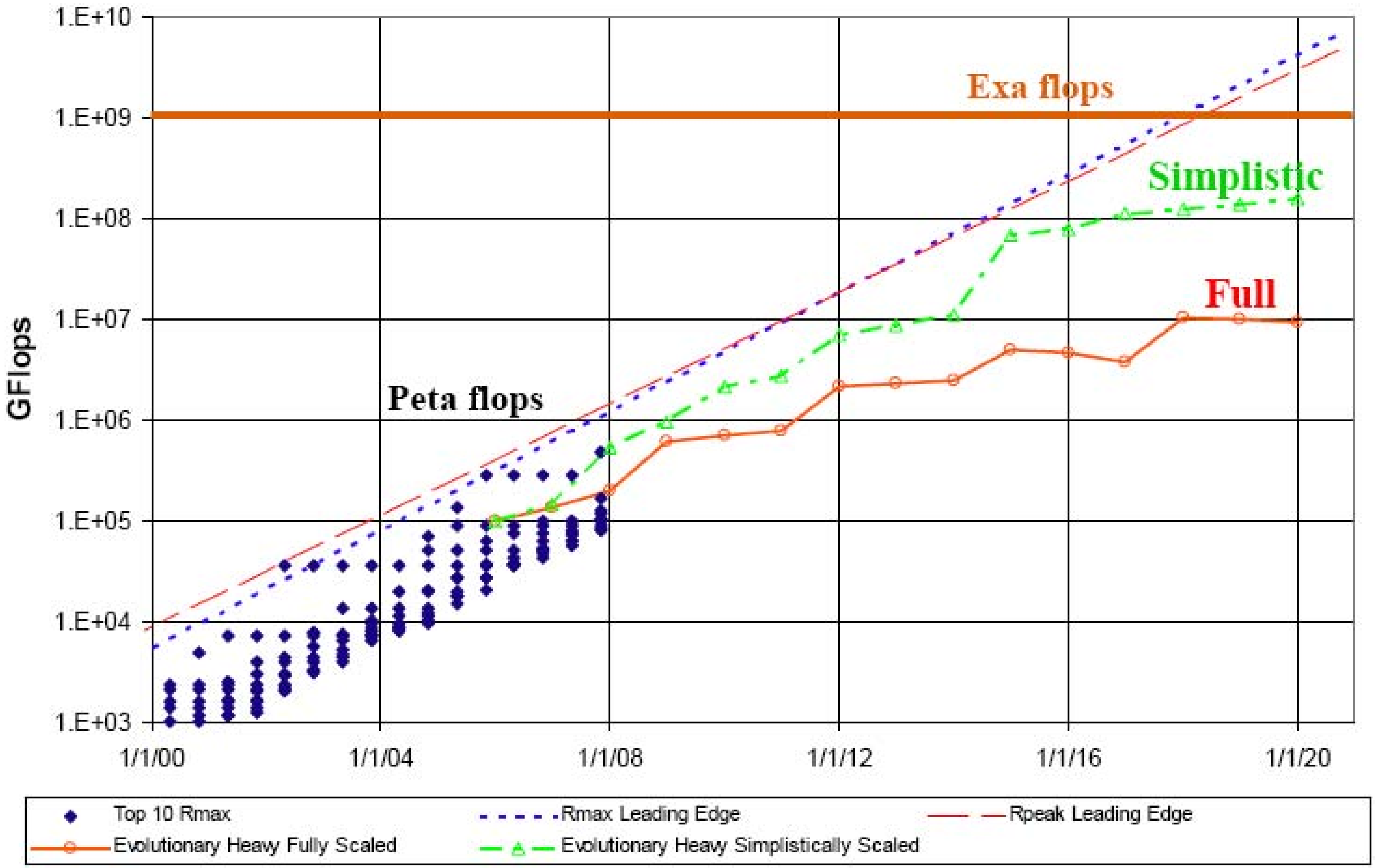


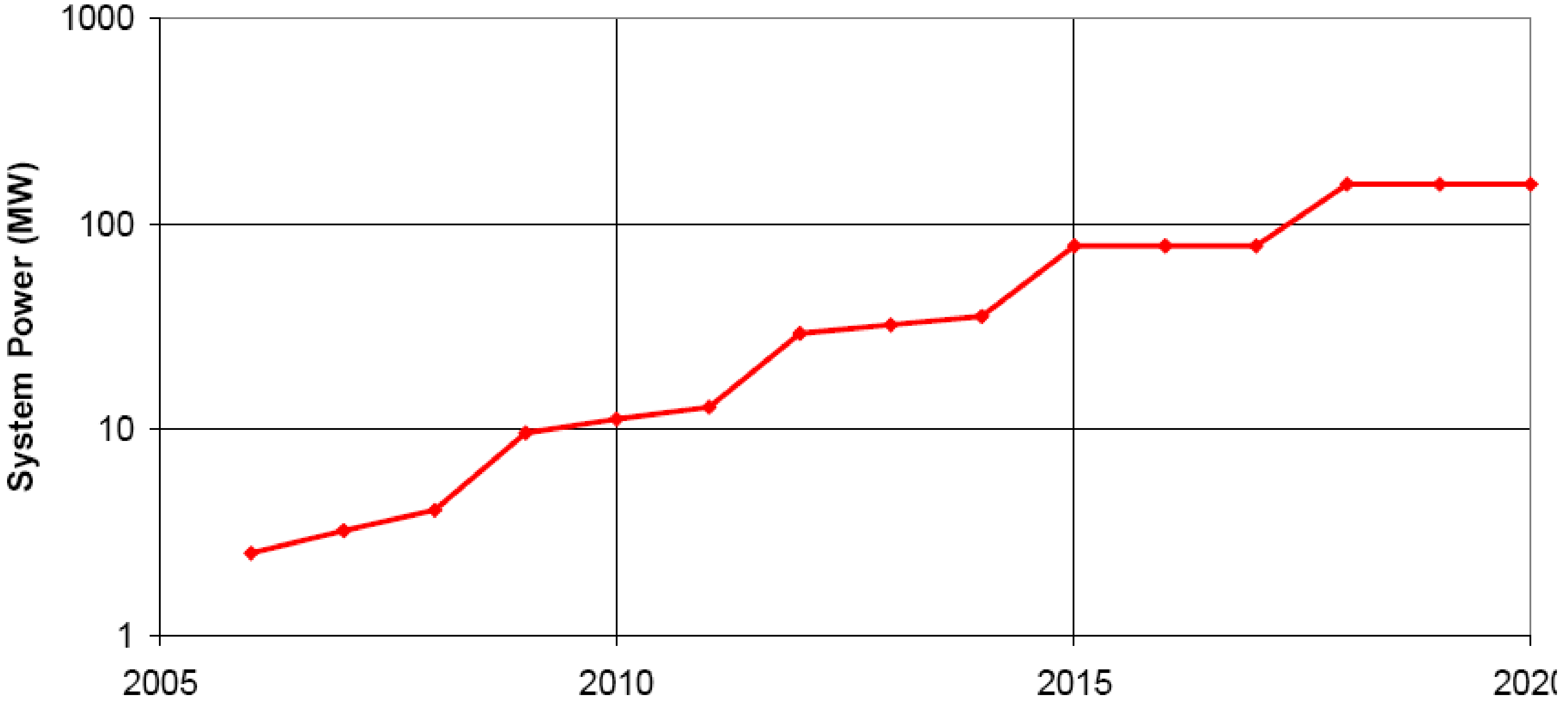
Figure courtesy of Kunle Olukotun, Lance Hammond, Herb Sutter, and Burton Smith

Exaflops will be hard!



From Peter Kogge, DARPA Exascale Study

… and the power costs will still be staggering



From Peter Kogge,
DARPA Exascale Study

The Challenge

- How to get to this level of performance without an annual electric bill greater than today’s procurement costs?
- How do you achieve this in a decade with a finite development budget?

Green Flash: Overview

We present an alternative approach to developing systems to serve the needs of scientific computing

- Choose our science target first to drive design decisions
- Leverage new technologies driven by consumer market
- ***Auto-tune software*** for performance, productivity, and portability
- Use hardware-accelerated architectural emulation to rapidly prototype designs (***auto-tune the hardware too!***)

- **A holistic approach: innovate algorithm/software/hardware together (Co-tuning)**

Achieve 100x energy efficiency improvement over mainstream HPC approach

Path to Power Efficiency

Reducing Waste in Computing

The portable consumer electronics market:

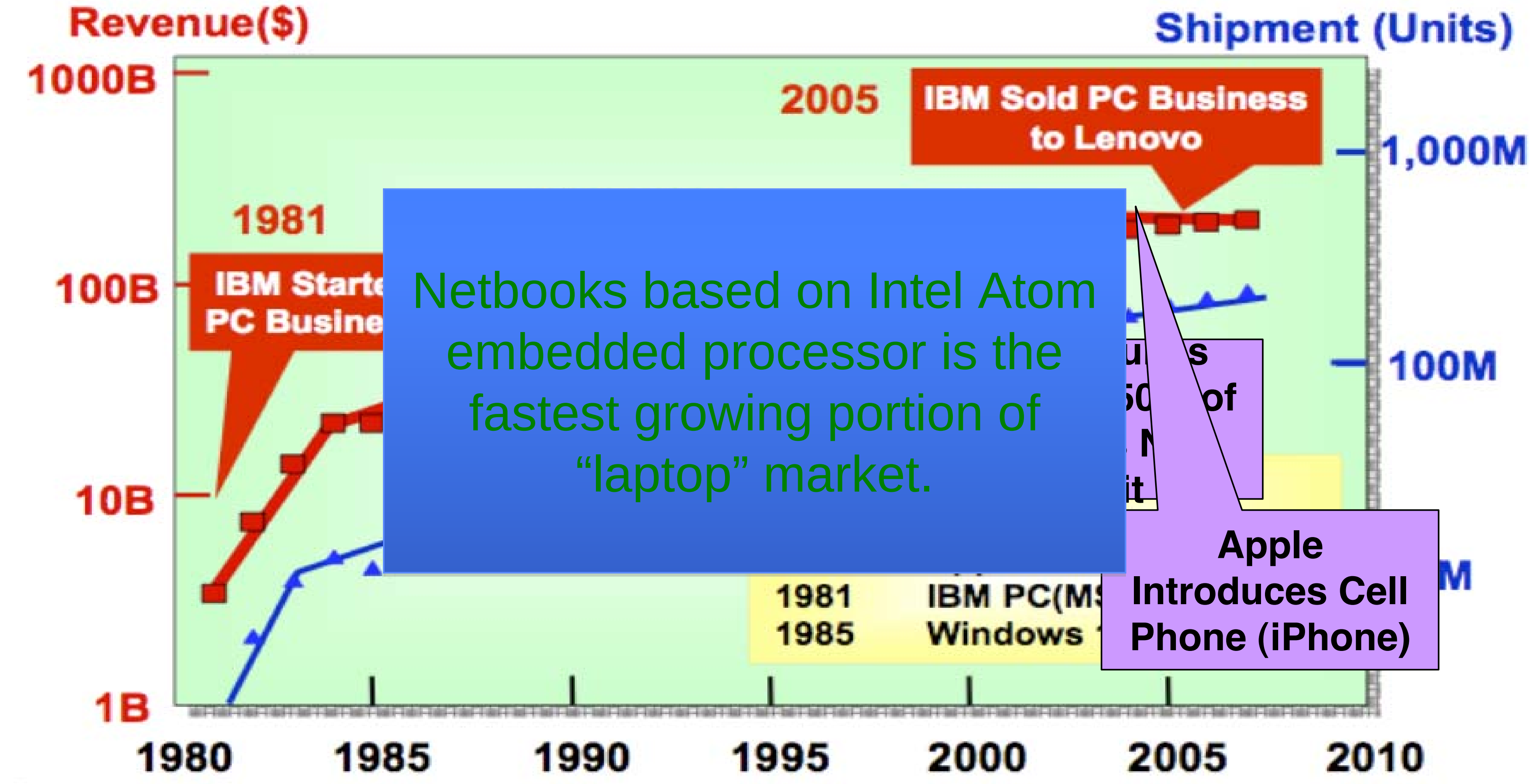
- **Optimized for low power, low cost, and high computational efficiency**

> *"Years of research in low-power embedded computing*
> *have shown only one design technique*
> *to reduce power: reduce waste."*
>
> —Mark Horowitz, Stanford University & Rambus Inc.

Sources of Waste:

- **Wasted transistors (surface area)**
- **Wasted computation (useless work/speculation/stalls)**
- **Wasted bandwidth (data movement)**

Consumer Electronics has Replaced PCs as the Dominant Market Force in CPU Design



Netbooks based on Intel Atom embedded processor is the fastest growing portion of “laptop” market.

Apple Introduces Cell Phone (iPhone)

Source: IDC

From Tsugio Makimoto: ISC2006

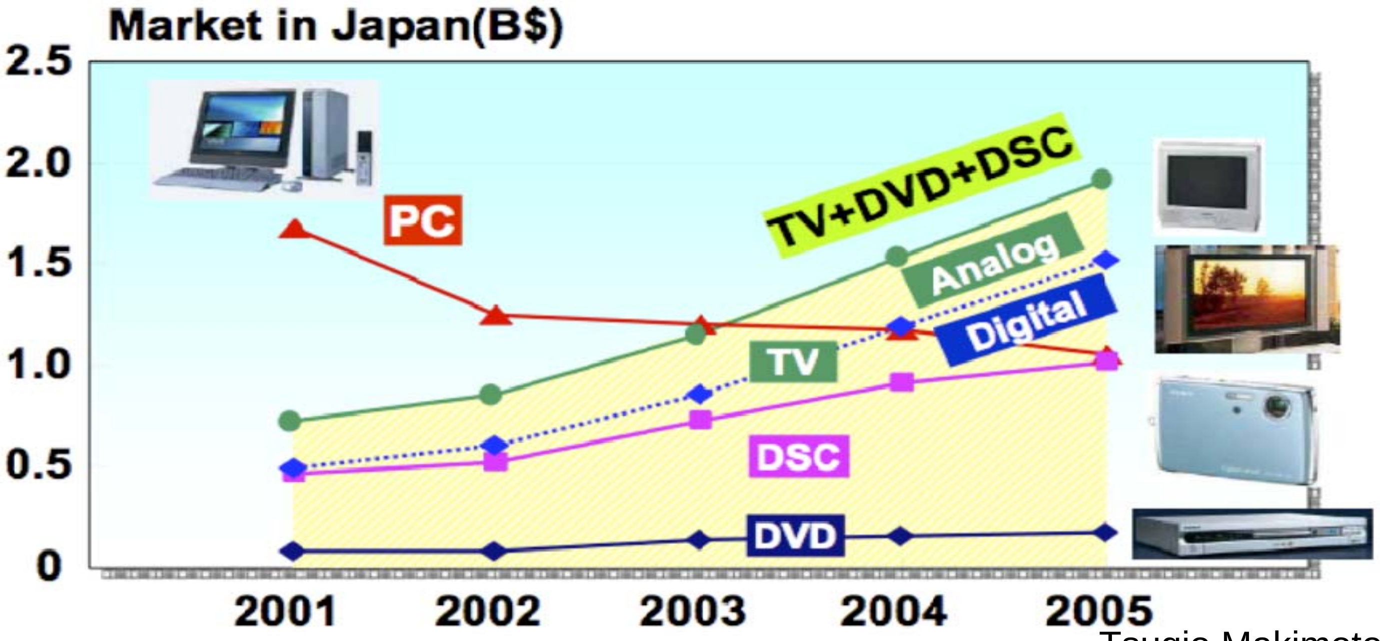
History repeats itself

1990s – HPC made the transition from vector to highly parallel platforms

- Had to learn how to use desktop COTS technology for scientific computing

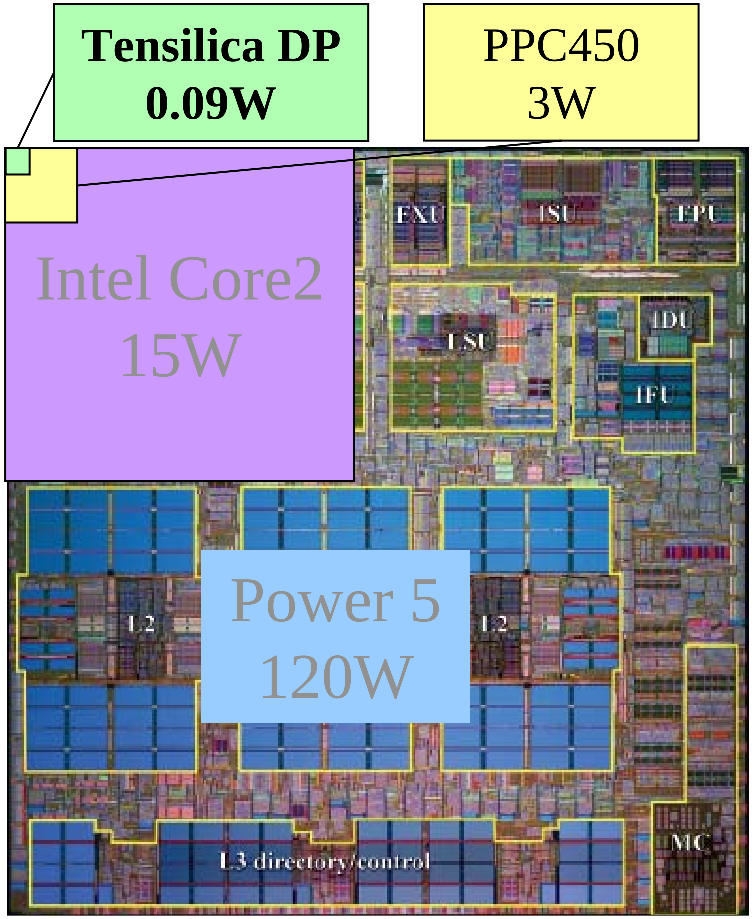
Now- R&D investments moving to consumer electronics/embedded processing

- Must learn to leverage embedded technology for future HPC systems



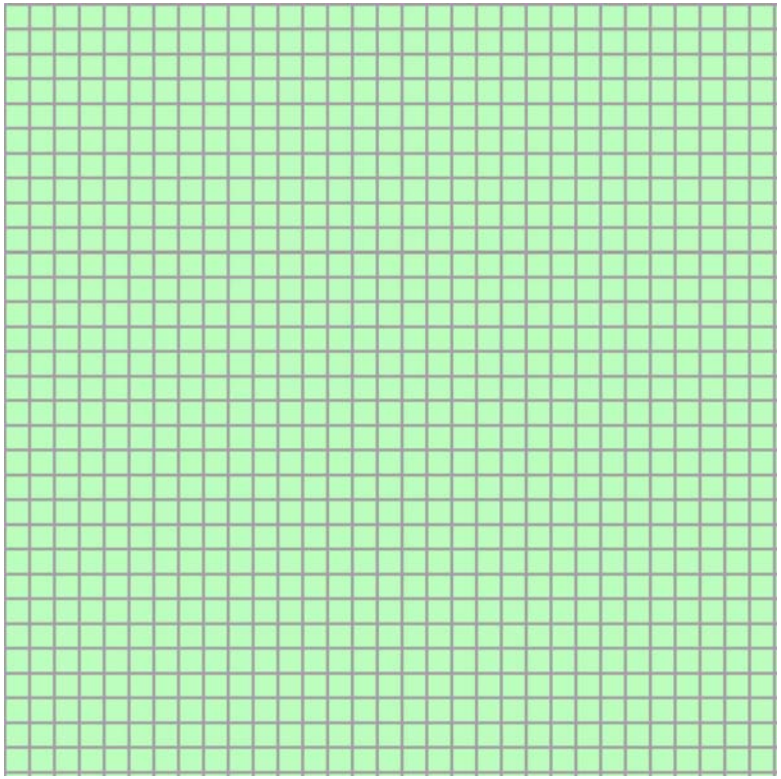
Tsugio Makimoto ISC2006

Design for Low Power: More Concurrency



- IBM Power5 (server)
  - 120W@1900MHz
- **Cubic power improvement with lower clock rate due to V^2F**
  - Intel Core2 sc (laptop) 15W@1000MHz
- **Slower clock rates enable use of simpler cores – shorter pipelines, less area, lower leakage**
  - IBM PPC 450 (BG/P - low power) 3W@800MHz
- **Tailor design to application to reduce waste**
  - Tensilica XTensa (Moto Razor) 0.09W@600MHz

Low Power Design Principles



Even if each core operates at 1/3 of the frequency of fastest available processors, you can pack 100s of simple cores onto a chip and consume 1/10 the power

One IBM Power5

- 120W
- 1900MHz

128 Tensilica Xtensa DP

- 11.5W
- equivalent to 76,800MHz

If the application has enough parallelism the many-core chip can be much faster and consume less power.

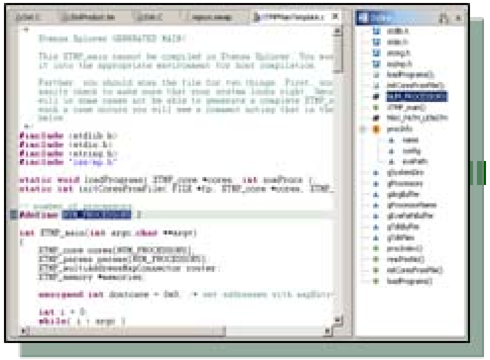
CS101: How to design a power efficient computer

- Spec out the requirements of your code.
  - Aim for a class of codes, not just one
- Learn how to design processors and interconnects.
  - We obtained chip design tools from Tensilica, a leading designer of chips for cell phones and other consumer elextronics.
    - Each chip design comes with its own C compiler and debugger
- Emulate your chip design on your code.
  - RAMP emulates chips with FPGAs
    - Hardware emulation is more accurate and thousands of times faster than software emulation
- Iterate your chip design and your software.
  - Autotuning takes advantage of the specific C compiler
  - Profile the code to determine chip parameters.

Embedded Design Automation

(Example from Existing Tensilica Design Flow)

Leverage mainstream tools, design processes, and commodity IP. Allows potential for HW/SW co-design

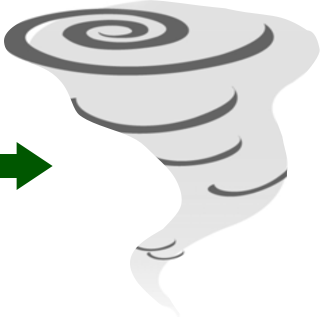


```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

int XTMP_main(int argc, char **argv)
{
```

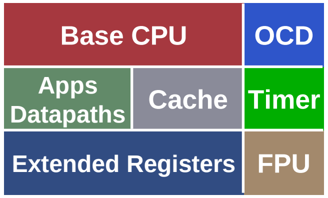
Processor configuration

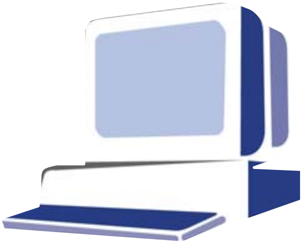
1. Select from menu
2. Automatic instruction discovery
3. Explicit instruction description



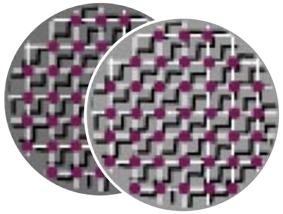
Processor Generator (*Tensilica*)

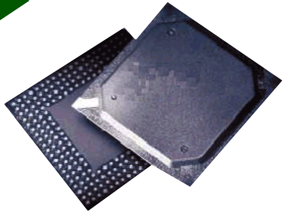
Application-optimized processor implementation (RTL/Verilog)





Tailored SW Tools: Compiler, debugger, simulators, Linux, other OS Ports
Derived HW characterics: size, power, etc.





Build with any process in any fab

Or emulate performance prior to fab

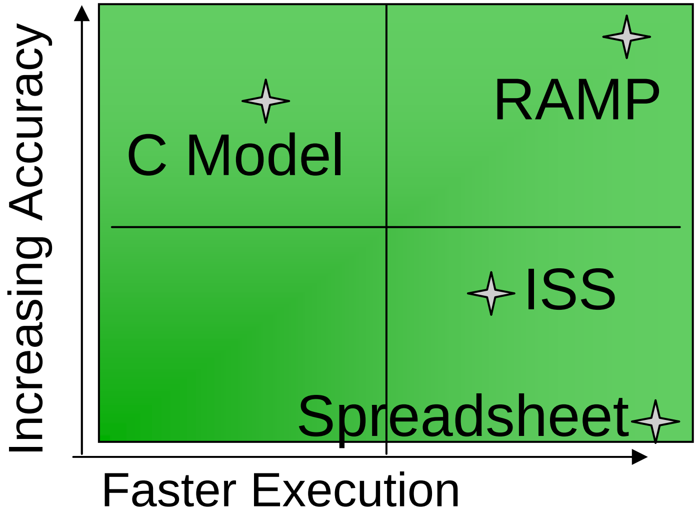
Advanced Hardware Simulation (RAMP)

Research Accelerator for Multi-Processors (RAMP)

- Utilize FGPA boards to emulate multicore systems
- 1000x speedup versus software emulation
- Allows fast performance validation
- Emulates entire application (not just kernel)
- Break slow feedback loop for system designs
- Enables tightly coupled hardware/software/science co-design

Technology partners:

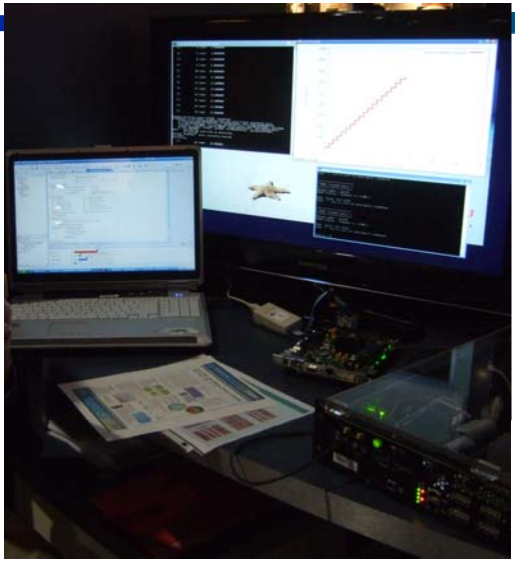
- UC Berkeley: John Wawrzynek, Jim Demmel, Krste Asanovic, Kurt Keutzer
- Stanford/ Rambus Inc.: Mark Horowitz
- Tensilica Inc.: Chris Rowen



Demonstration of Green Flash Approach

SC ’09 Demo of approach feasibility

- CSU limited-area atmospheric model ported to Tensilica architecture
- Dual-core Tensilica processor running atmospheric model
- Eight and Sixteen Core configuration coming online
- MPI Routines ported to custom interconnect



Emulation performance advantage

- Processor running at 25MHz vs. Functional model at 100 kHz (250x speedup)
- *Actual code running - not representative benchmark*

Auto-Tuning for Green Flash

Challenge: How to optimize the climate code for the differing chip designs under consideration.

Solution: Auto-tuning

- Different chip designs may require vastly different optimizations
- Labor-intensive
- Automate search across a complex optimization space

> “Never send a human to do a machine's job.”
>
> – Agent Smith, *The Matrix*

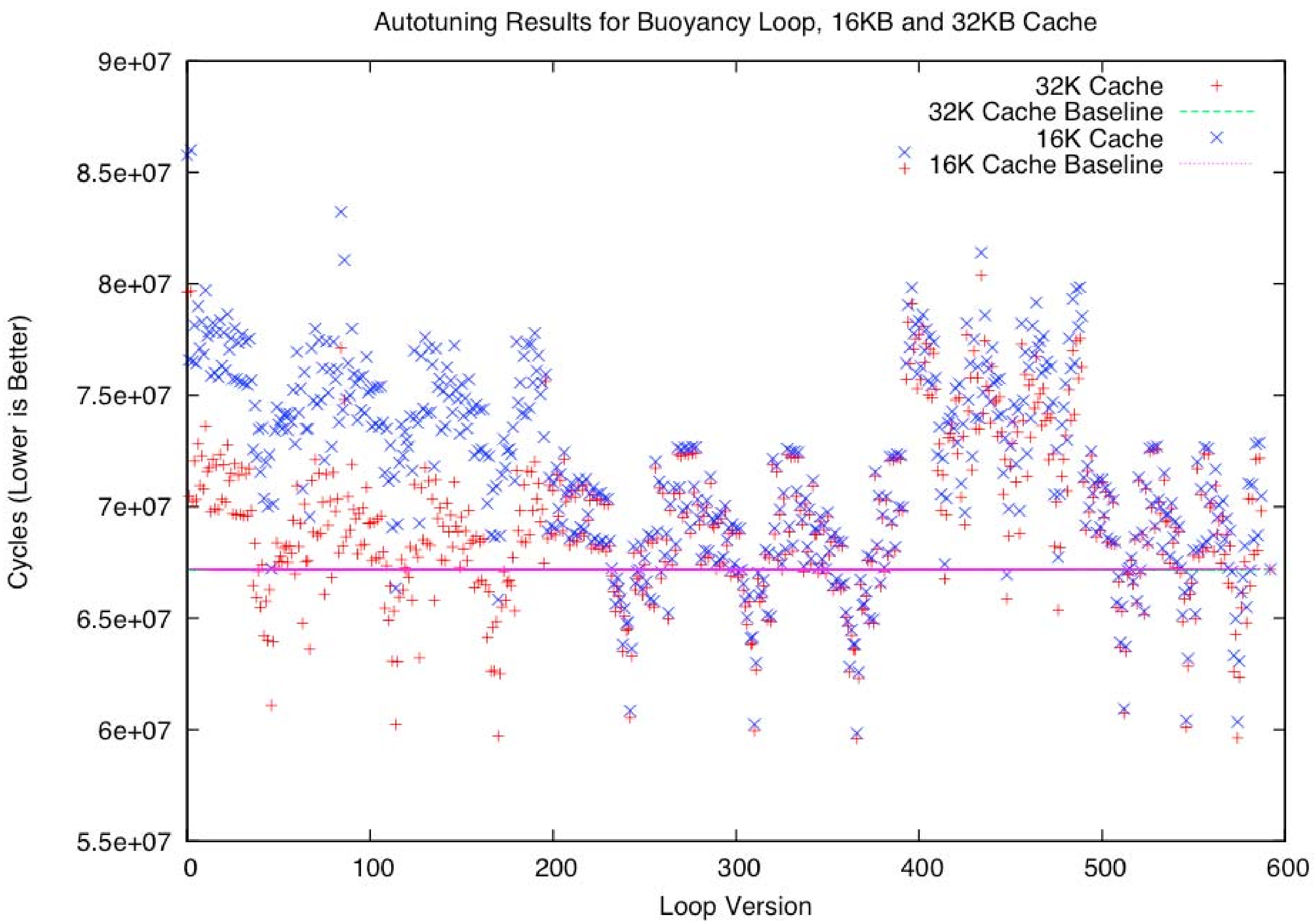
Autotuning Example

- **Operators extracted from climate code before and after auto-tuning**

```
 do k=0,km,1
  do iprime=1,nside,1
    do i=2,im2nghost-1,1
    ia  = i + ii(iprime)
      do j=2,jm2nghost-1,1
         ja  = j + jj(iprime)
        buoyancy_gen(i,j,iprime,k)
           =-1.0*g*(theta(ia,ja,k) -
           theta(i,j,k))
          /(theta00(k)*el(iprime))
      enddo
    enddo
  enddo
enddo
```

```
do G14906=0,km,4
do G14907=1,nside,6
do G14908=2,im2nghost - 1,25
do G14909=2,jm2nghost - 1,25
do k=G14906,G14906 + 1,1
do iprime=G14907,G14907 + 5,1
do i=G14908,G14908 + 24,1
ia = i + ii(iprime)
do j=G14909,G14909 + 24,1
ja = j + jj(iprime)
buoyancy_gen(i,j,iprime,k) = -1.0 * g * theta(ia,ja,k) - theta(i,j,k) /
                 theta00(k) * el(iprime)
enddo
enddo
enddo
enddo
enddo
enddo
enddo
do G14907=1,nside,6
do G14908=2,im2nghost - 1,25
do G14909=2,jm2nghost - 1,25
do k=G14906 + 2,G14906 + 3,1
do iprime=G14907,G14907 + 5,1
do i=G14908,G14908 + 24,1
ia = i + ii(iprime)
do j=G14909,G14909 + 24,1
ja = j + jj(iprime)
buoyancy_gen(i,j,iprime,k) = -1.0 * g * theta(ia,ja,k) - theta(i,j,k) /
                 theta00(k) * el(iprime)
enddo
enddo
enddo
enddo
enddo
enddo
enddo
enddo
```

Auto-tuning Results

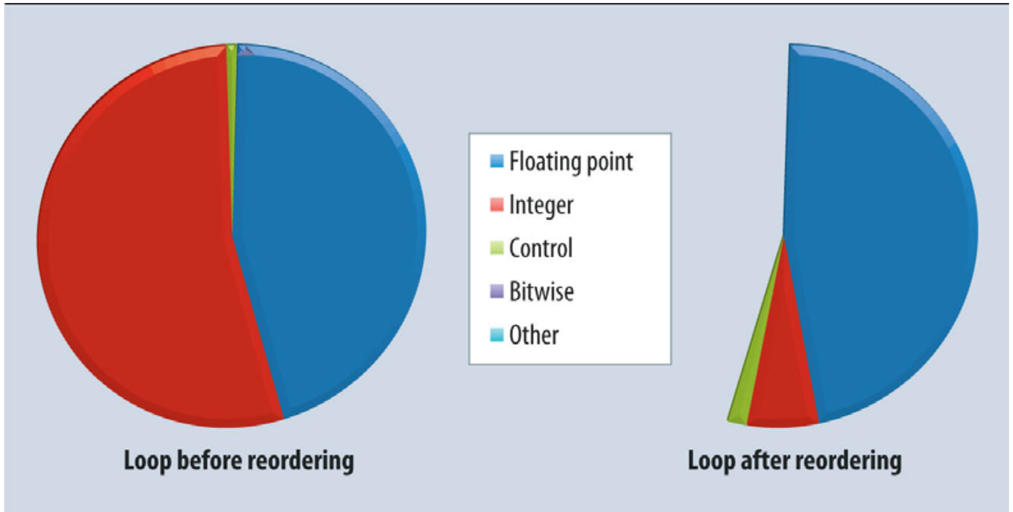


Shoaib Kamil UC Berkeley

Co-tuning: feedback on chip design

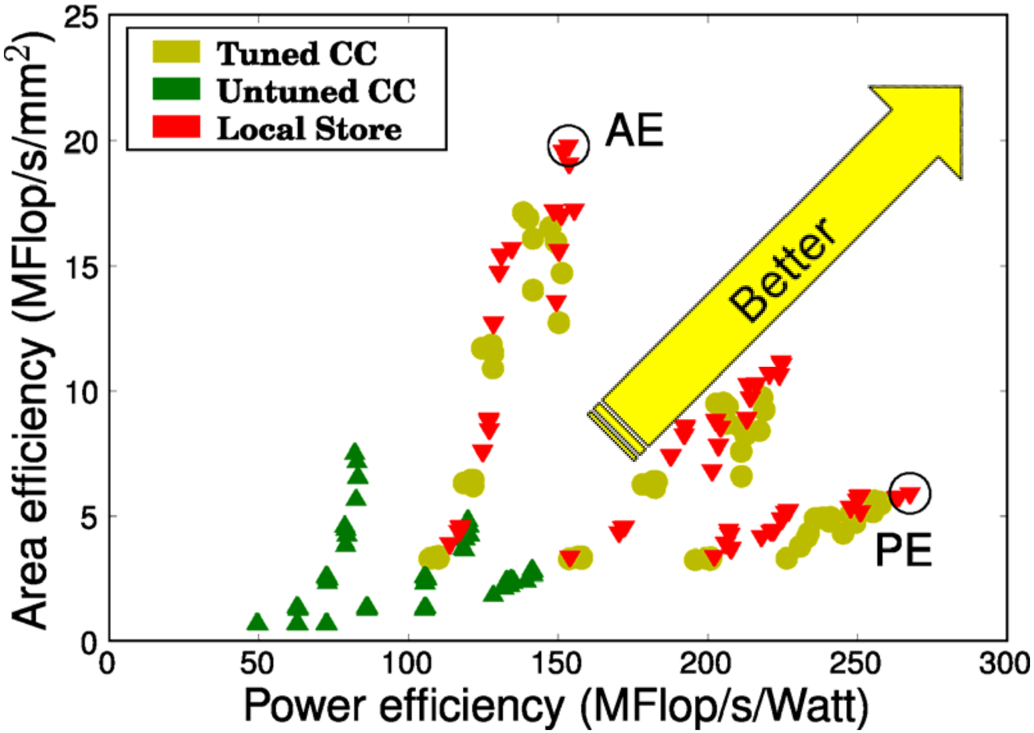
- We are fully profiling each loop in the code ->
- Auto-tuning can reduce the number of instructions but changes the mix of instructions.
  - Iterate this information in the chip design
- Example: the loop with the largest footprint
  - decreased the cache footprint from 160kb to 1kb
  - halved the instruction count

unique_addrs
unique_clines
footprint
footprintcache
tot_ins
Bytes/Inst
Bandwidth (MB/s)
fpload
fpstore
FP Arith
fpmov fprf
FP L/S
intload
intstore
intmov
Int Arith
Int L/S
j/call
branch
control
loop entry
Control
bitwise
shift
Logic



Hardware/Software Co-tuning

- The information from auto-tuning is relevant to the optimal chip design
- Iterate on the design process and autotuning

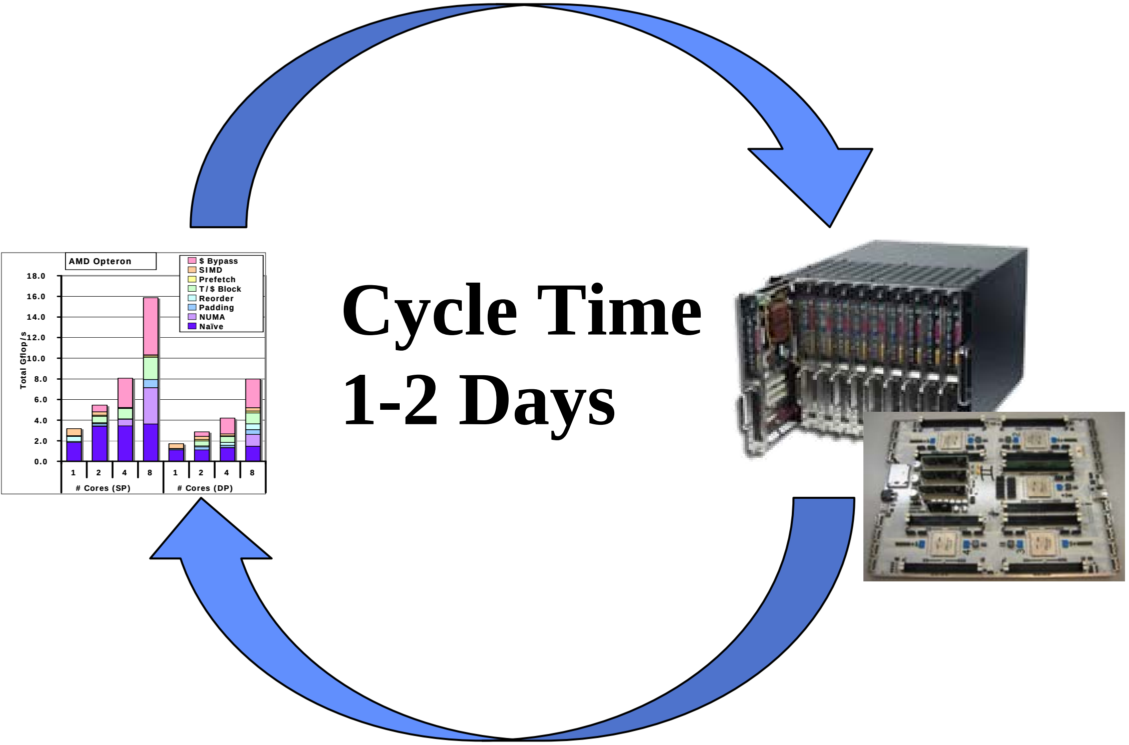


Marghoob Mohiyuddin, UC Berkeley

Holistic Hardware/Software Co-Design

How long does it take for a full scale application to influence architectures?

Synthesize SoC (Hours)



Cycle Time
1-2 Days

Auto-Tune Software (Hours-days)

Emulate Hardware RAMP (Hours)

Build Application

A strawman design concept

- CSU icosahedral code
  - –167,772,162 vertices, ~128 vertical levels

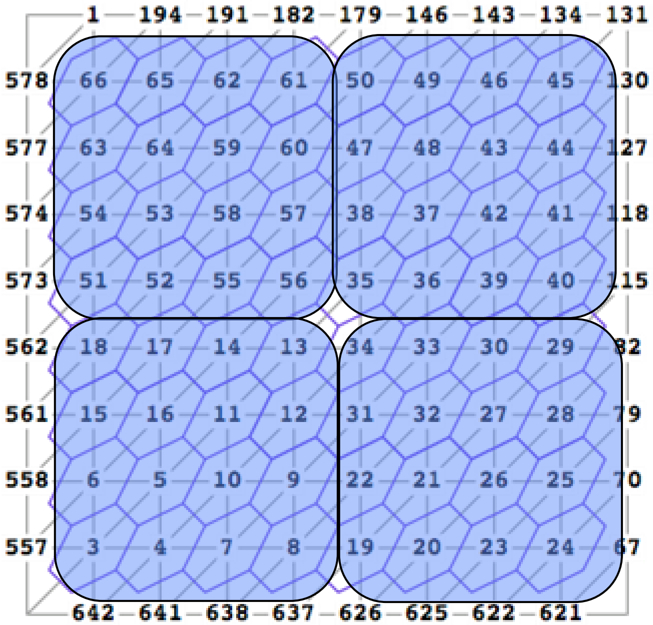
- A strawman design concept
  - –2,621,440 horizontal subdomains (logically rectangular, 8x8 cells each)
  - –8 vertical subdomains of 16 levels each.
    - 20,971,520 processing cores.
    - 163,840 chips with 128 cores each.
    - Specific processor and network properties to be determined from the code requirement model.

Code Requirements Model

- Preliminary results for CSU code
  - 167,772,162 vertices, ~128 vertical levels

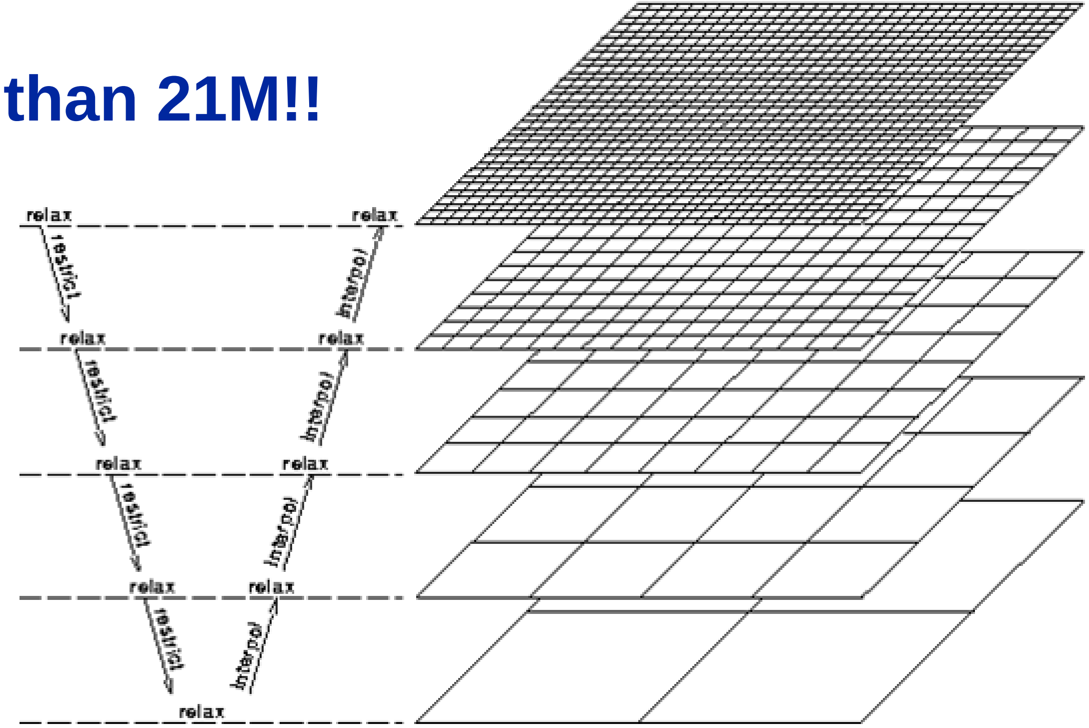
- Anelastic Dynamics only (w/o multigrid solve)
  - Sustained speed of 12 Pflops (600 Mflop per processor)
  - Total memory 560TB (27MB per processor)
  - 7623 messages per second per processor
  - Bandwidth 78MB/sec per processor

- Strong scaling w/ smaller domains
  - 10x lower latency & higher BW on chip
  - Many-core actually enables 20 million way parallelism!



Multigrid solver

- The MG solve is communication bound if the subdomains are too small.
- But multi-core chips helps this dramatically by reducing the communication costs of first few levels of the solve.
- 165K chips is better than 21M!!



Green Flash 100PF (peak) Strawman System Design

We examined three different approaches (in 2009 technology)

- **AMD Opteron:** Commodity approach, lower efficiency for scientific codes offset by advantages of mass market. Constrained by legacy/binary compatibility.
- **BlueGene:** Generic embedded processor core and customize system-on-chip (SoC) to improve power efficiency for scientific applications
- **Tensilica XTensa:** Customized embedded CPU w/SoC provides further power efficiency benefits but maintains programmability. Mainstream design process, tool chain, commodity IP

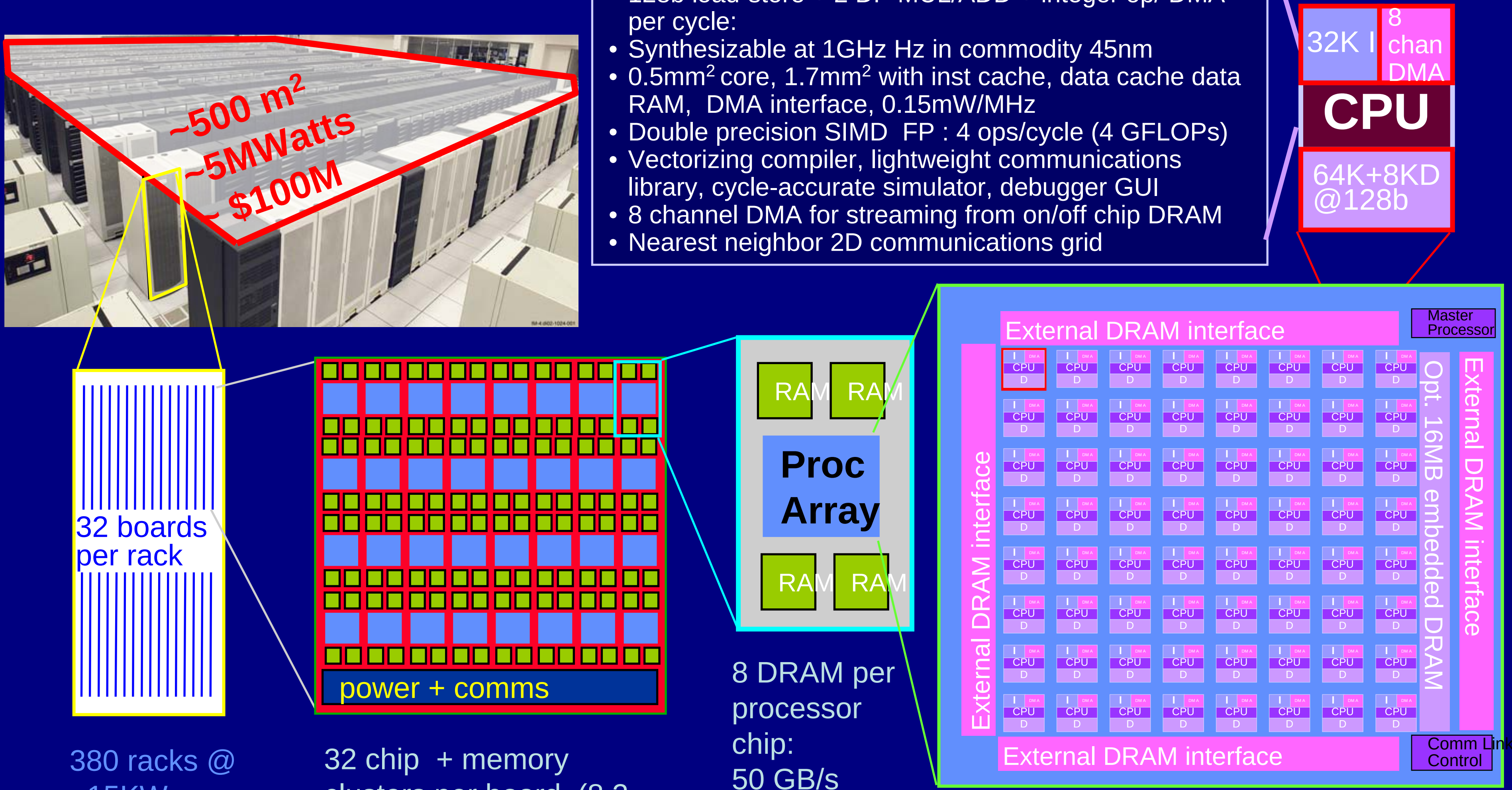
Processor	Clock	Peak/ Core (Gflops)	Cores/ Socket	Sockets	Cores	Power
AMD Opteron	2.2GHz	8.8	6	1.8M	11M	142MW
IBM BG/P	0.8GHz	3.4	4	7M	29M	198MW
Green Flash / Tensilica XTensa	1GHz	4	64	0.4M	25M	5 MW

Climate Modeling System

Strawman 100PF Design

VLIW CPU:

- 128b load-store + 2 DP MUL/ADD + integer op/ DMA per cycle:
- Synthesizable at 1GHz Hz in commodity 45nm
- 0.5mm^2 core, 1.7mm^2 with inst cache, data cache data RAM, DMA interface, 0.15mW/MHz
- Double precision SIMD FP : 4 ops/cycle (4 GFLOPs)
- Vectorizing compiler, lightweight communications library, cycle-accurate simulator, debugger GUI
- 8 channel DMA for streaming from on/off chip DRAM
- Nearest neighbor 2D communications grid



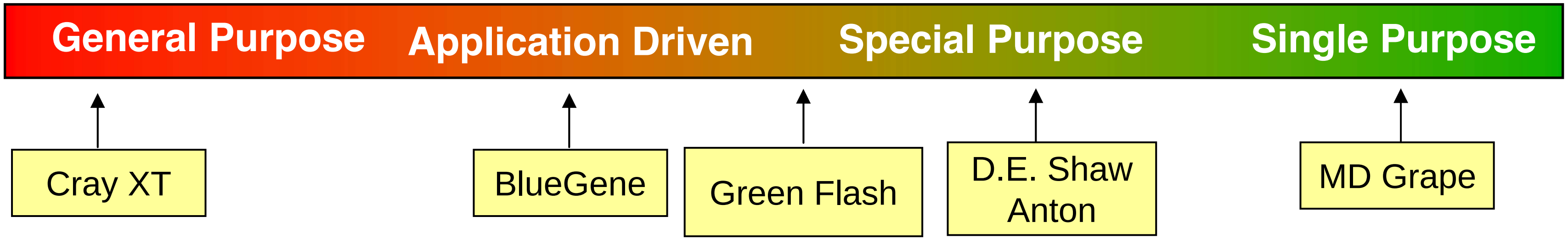
380 racks @ ~15KW

32 chip + memory clusters per board (8.2 TFLOPS @ 450W

8 DRAM per processor chip: 50 GB/s

64 processors per 45nm chip
512 GFLOPS @ 10W

Green Flash Customization Continuum



General Purpose

Application Driven

Special Purpose

Single Purpose

Cray XT

BlueGene

Green Flash

D.E. Shaw Anton

MD Grape

Application-driven does NOT necessitate a special purpose machine (or exotic tech)

Riken MD-Grape: Full custom ASIC design

- 1 Petaflop performance for one application using 260 kW for $9M

D.E. Shaw Anton System: Full and Semi-custom design (bio-molecular)

- Simulate 100x–1000x timescales vs any existing HPC system (~200kW)

Application-Driven Architecture (Green Flash): Semicustom design

- Highly programmable core architecture using C/C++/Fortran
- Goal of 100x power efficiency improvement vs general HPC approach

Summary

We propose a new approach to high-end computing with potentially transformational impact on science

- **Choose the science target first *(climate in this case)***
- **Design systems for applications *(rather than the reverse)***
- **Leverage power efficient embedded technology**
- **Design hardware, software, scientific algorithms together using auto-tuning, co-tuning, hardware emulation**
- **Achieve exascale computing sooner and more cost/power efficiently**

Applicable to broad range of exascale-class applications

A concluding thought

At the exascale, numerical experimentalists must take a lesson from actual experimentalists.

A concluding thought

At the exascale, numerical experimentalists must take a lesson from actual experimentalists.

Design machines to answer specific scientific questions rather than limit our questions by available machines.

Questions?

http://www.lbl.gov/cs/html/greenflash.html

LBNL Contributors: John Shalf, Lenny Oliker, David Donofrio, Tony Drummond, Shoaib Kamil, Norman Miller, Marghoob Mohiyuddin, Woo-Sun Yang, Kathy Yelick

External Collaborators: UC Berkeley, Stanford, Colorado State University, Tensilica

This work was funded in part the DOE Office of Science under contract number DE-C02-05CH11231.

Fault Tolerance/Resilience

- Our Design does not expose unique risks
  - Faults proportional to # sockets (not # cores) and silicon surface area
  - We expose less surface area and fewer sockets with our approach
- Hard Errors
  - Spare cores in design (Cisco Metro)
  - SoC design (fewer components and fewer sockets)
  - Use solder (not sockets)
- Soft Errors
  - ECC for memory and caches
  - On-board NVRAM controller for localized checkpoint
  - Checkpoint to neighbor for rollback

Green Flash:
Fault Tolerance/Resilience

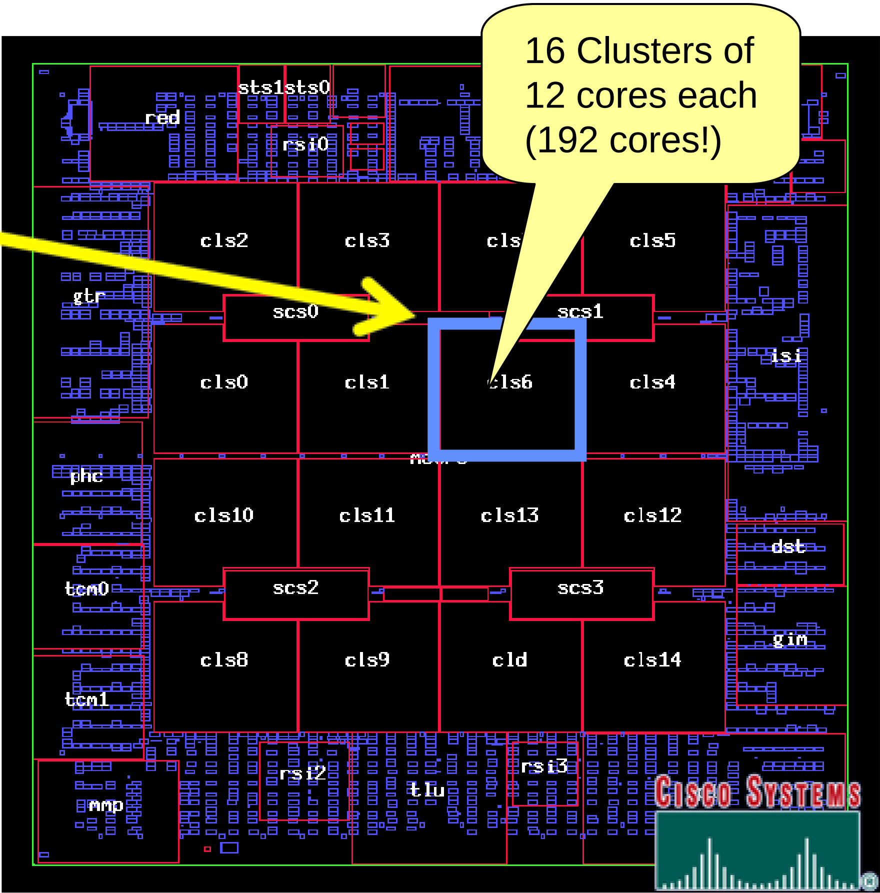
- ***Large scale applications must tolerate node failures***
- **Our design does not expose unique risks**
  - **Faults proportional to sockets (not cores) & silicon surface area**
  - **Low-power manycore uses less s**
    sockets

Hard Errors

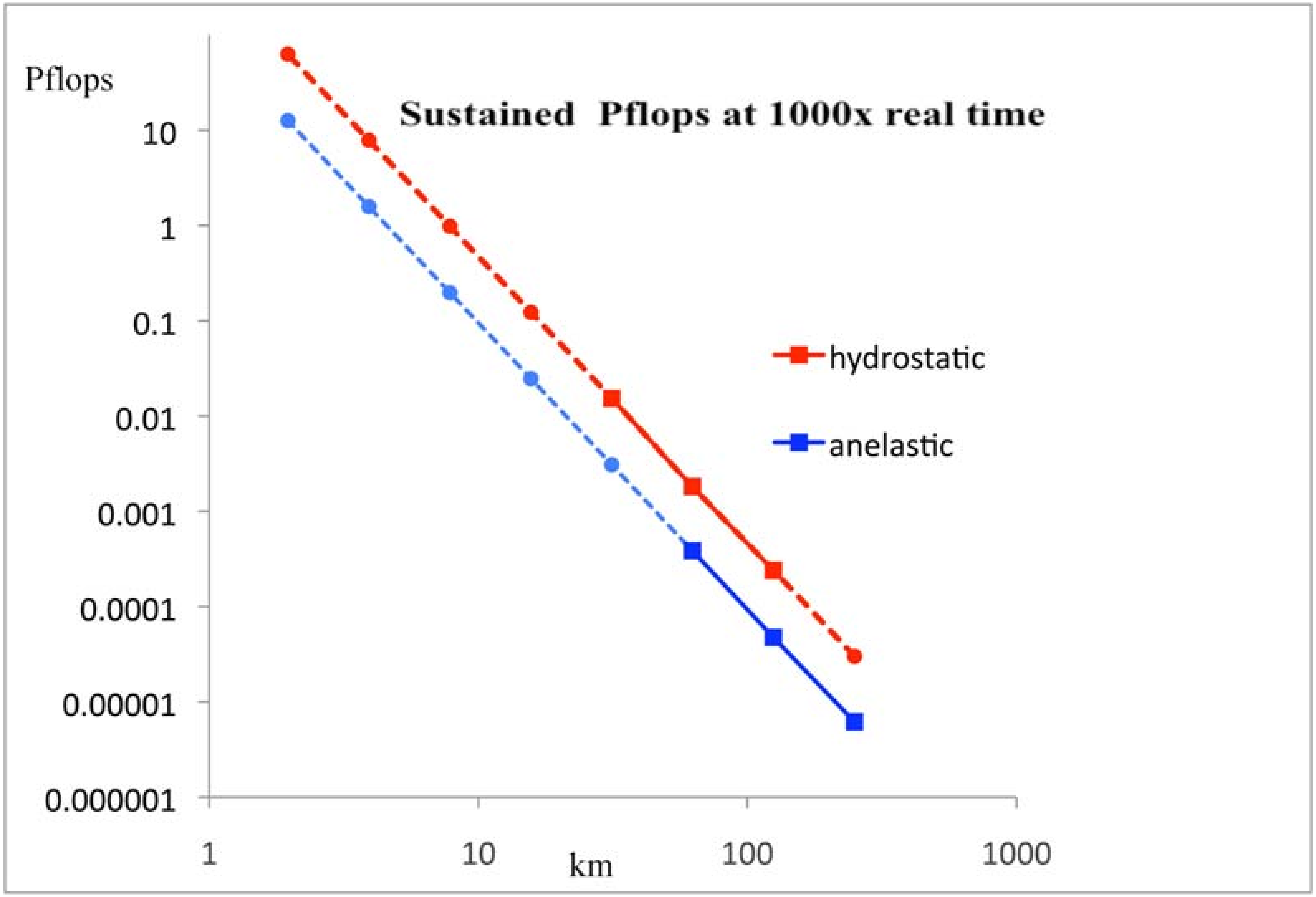
- Spare cores in design (Cisco Metro: 188 cores + 8 spares)
- SystemOnChip design (fewer components→fewer sockets)

Soft Errors

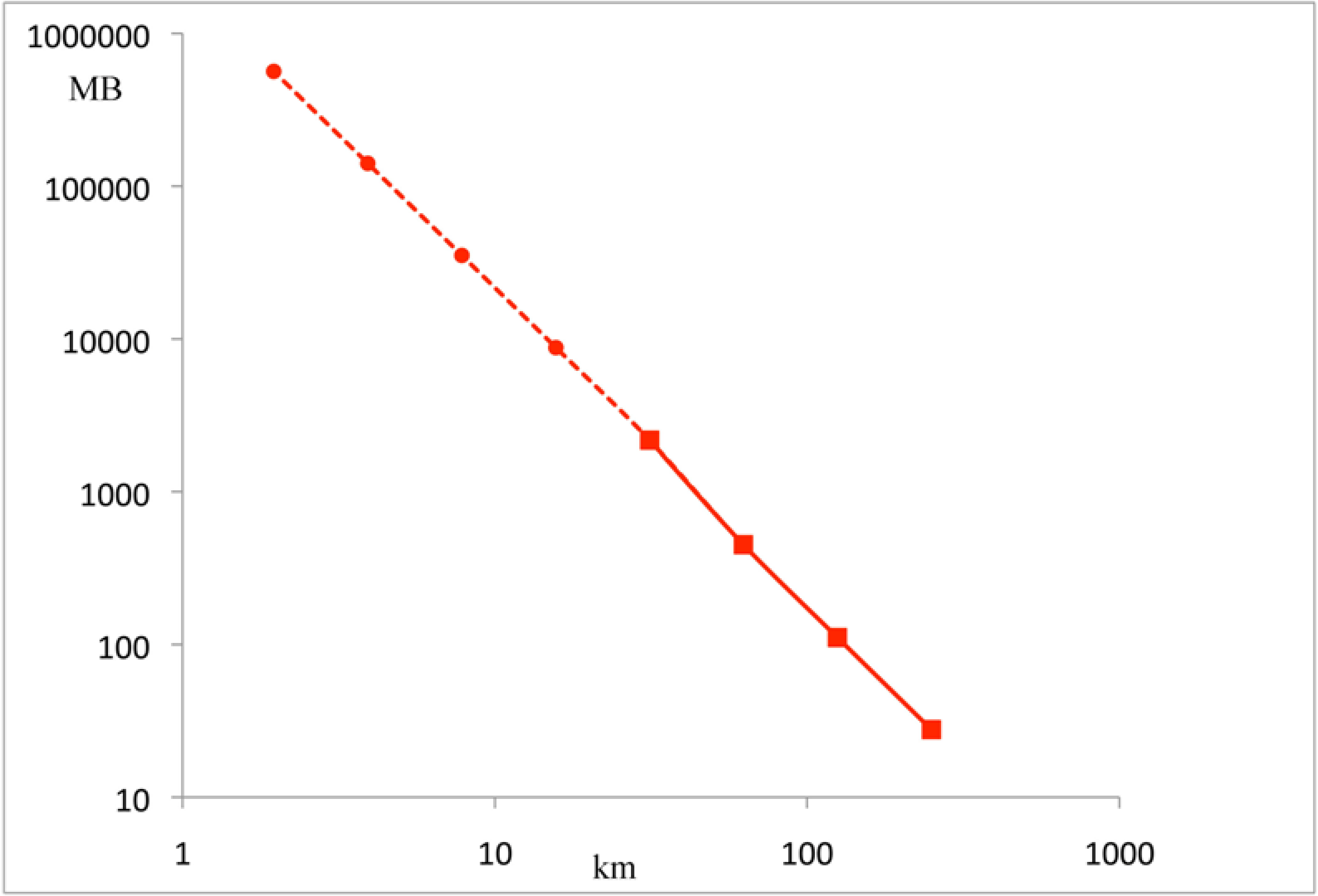
- ECC for memory and caches
- On-board NVRAM controller for localized checkpoint


16 Clusters of 12 cores each (192 cores!)

Sustained speed



Total Memory requirement


MB
km