Requirements, Challenges, and Progress towards Exascale Computing

Thomas Sterling

Professor of Informatics and Computing, Indiana University

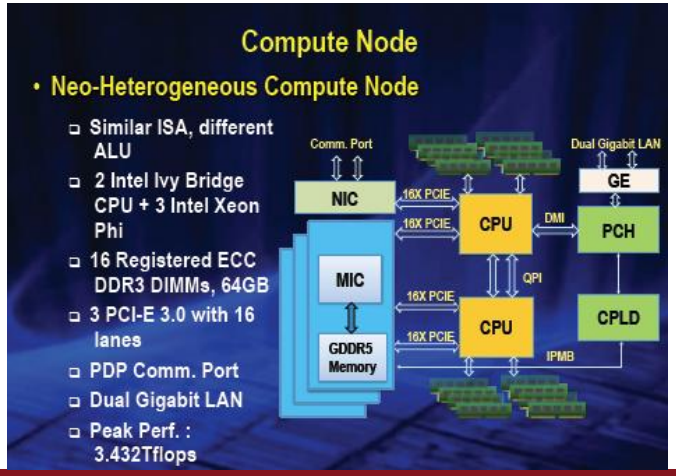
Chief Scientist and Associate Director
Center for Research in Extreme Scale Technologies (CREST)
Pervasive Technology Institute at Indiana University

Fellow, Sandia National Laboratories CSRI

June 25, 2013

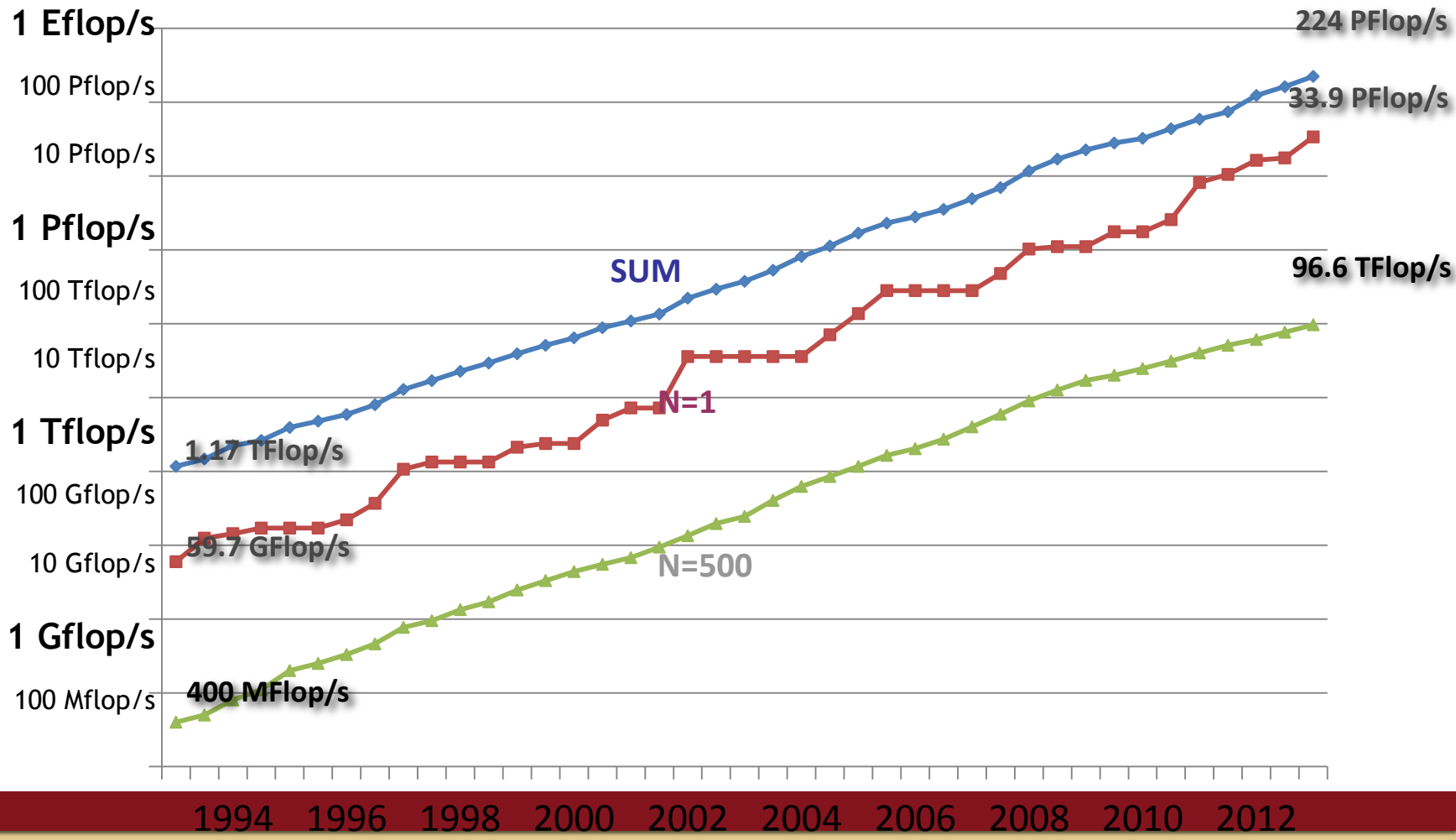
Tianhe-2: Half-way to Exascale

- **China, 2013**: *the 30 PetaFLOPS dragon*
- Developed in cooperation between NUDT and Inspur for National Supercomputer Center in Guangzhou
- Peak performance of 54.9 PFLOPS
  - 16,000 nodes contain 32,000 Xeon Ivy Bridge processors and 48,000 Xeon Phi accelerators totaling 3,120,000 cores
  - 162 cabinets in 720m^2 footprint
  - Total 1.404 PB memory (88GB per node)
  - Each Xeon Phi board utilizes 57 cores for aggregate 1.003 TFLOPS at 1.1GHz clock
  - Proprietary TH Express-2 interconnect (fat tree with thirteen 576-port switches)
  - 12.4 PB parallel storage system
  - 17.6MW power consumption under load; 24MW including (water) cooling
  - 4096 SPARC V9 based Galaxy FT-1500 processors in front-end system



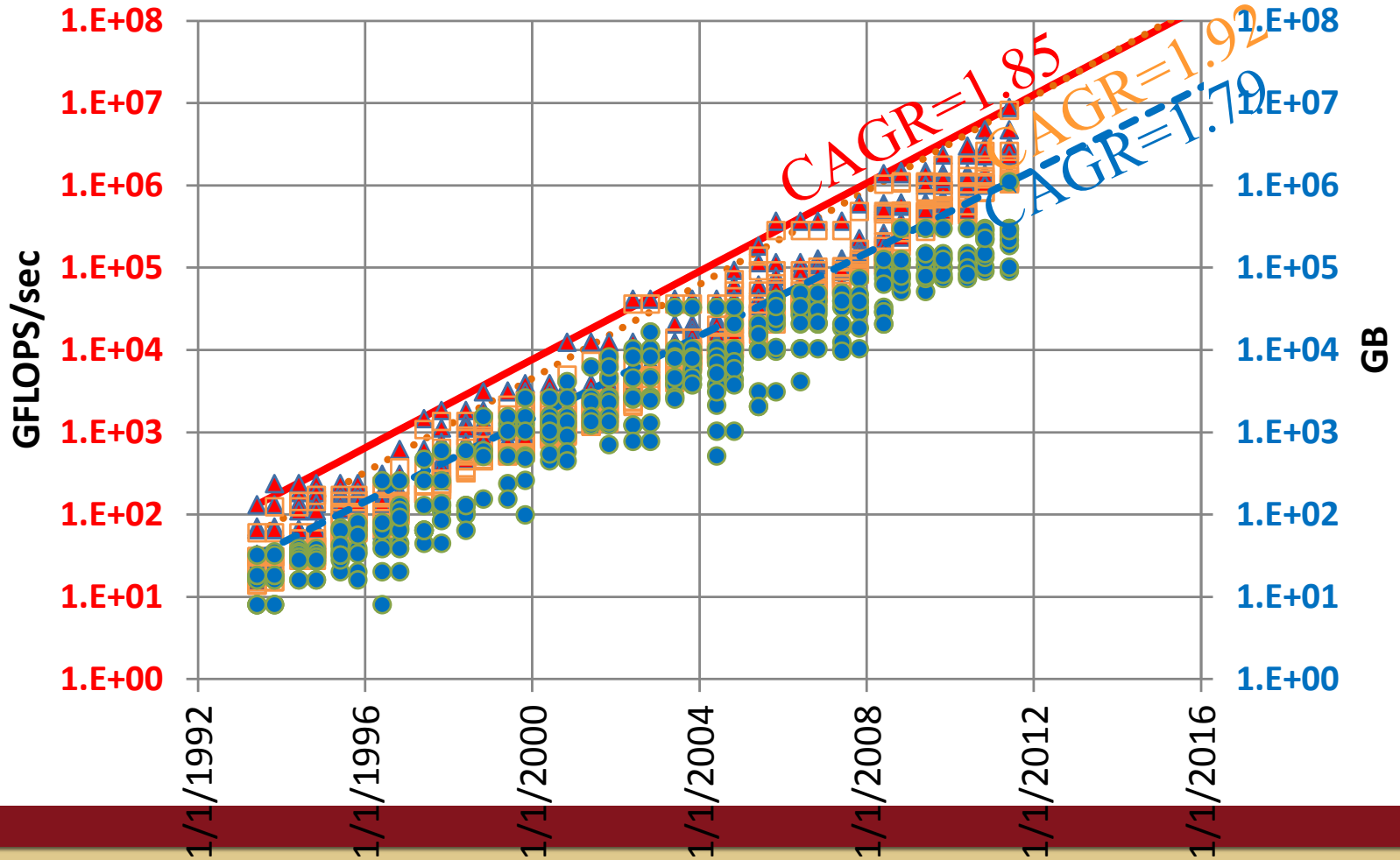
Courtesy of Erich Strohmaier LBNL

Performance Development



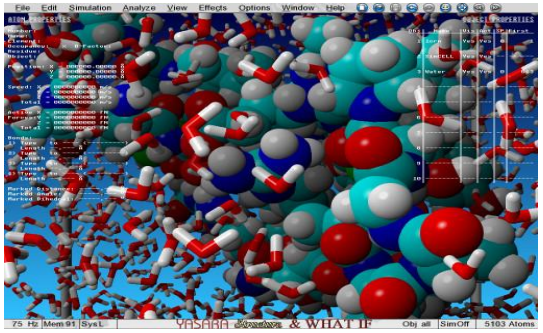
Courtesy of Peter Kogge, UND

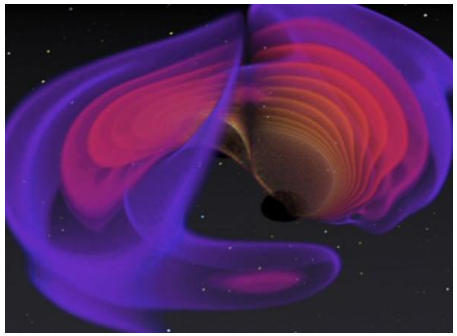
Overall Metrics

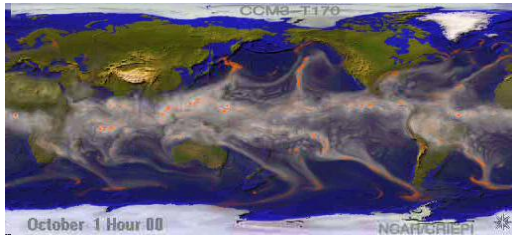


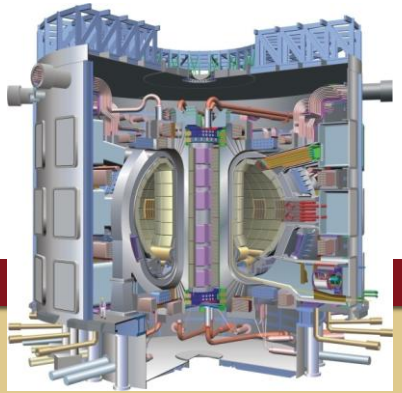
Motivation – Applications for Science, Commerce, Security, and the Social Welfare

- Cosmology and Astronomy
- Weather and Climate
- Energy and Combustion
- Aerospace and Auto Crash
- Nuclear Reactors and Controlled Fusion
- Biology and Molecular Dynamics
- Medical and Drug Design
- Visualization and Entertainment
- Electronic Technology and Design
- Manufacturing and Distribution
- Chemistry
- Genomics
- Financial
- Crypto-Analysis and IDing

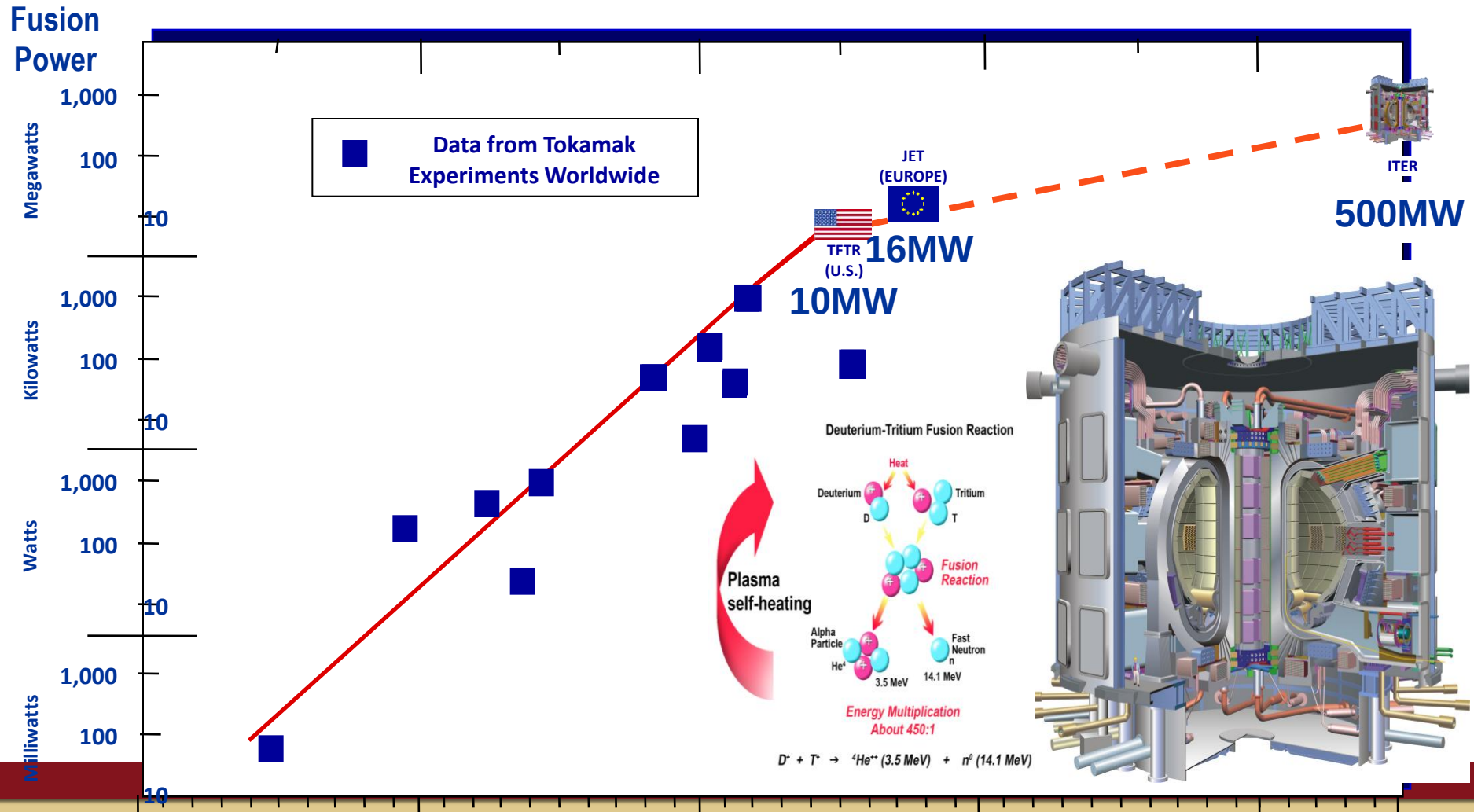






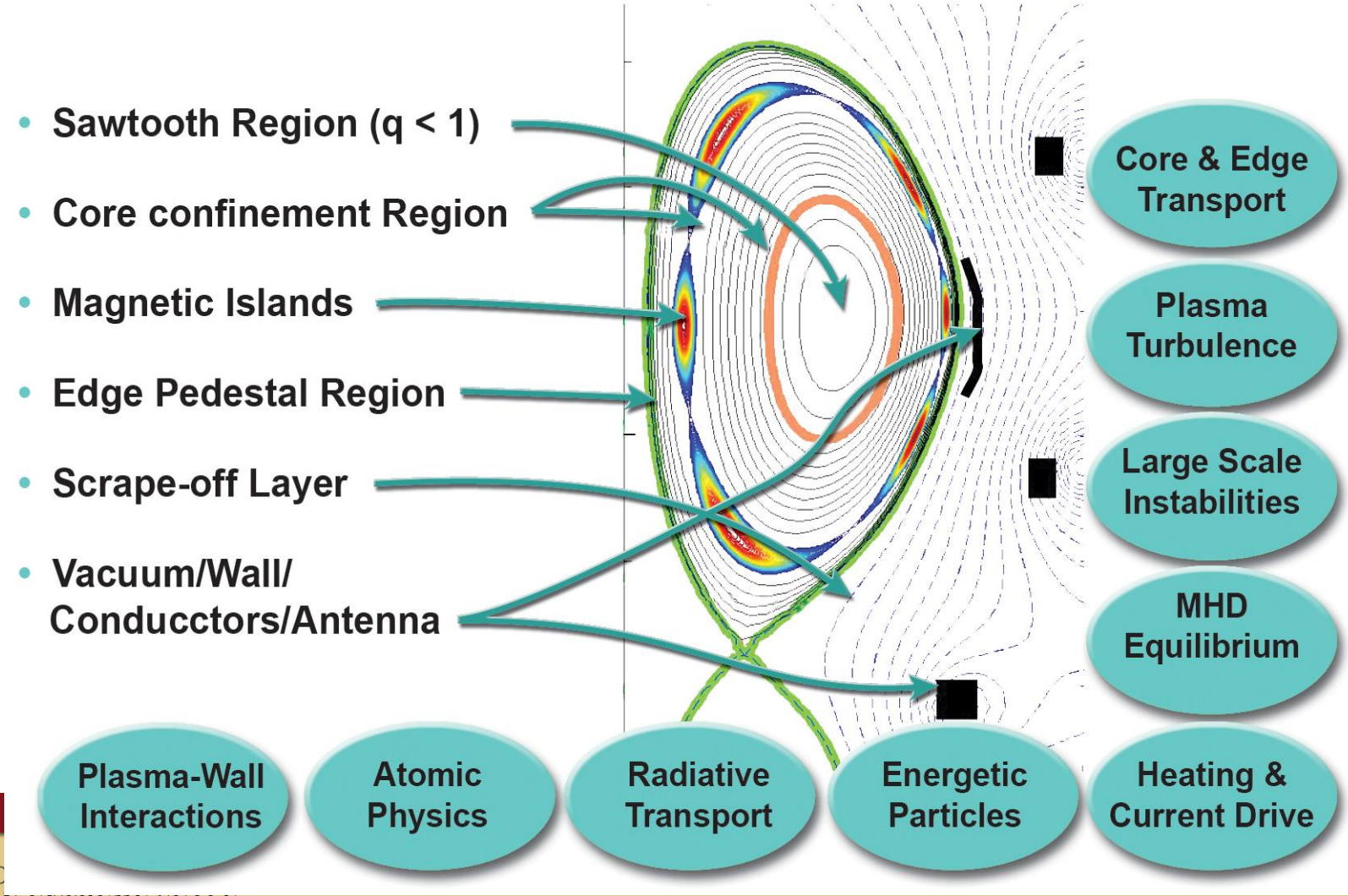


Progress in Magnetic Fusion Energy (MFE) Research



Courtesy of Bill Tang, Princeton

Elements of an MFE Integrated Model ➔ *Complex Multi-scale, Multi-physics Processes*



Courtesy of Bill Tang, Princeton

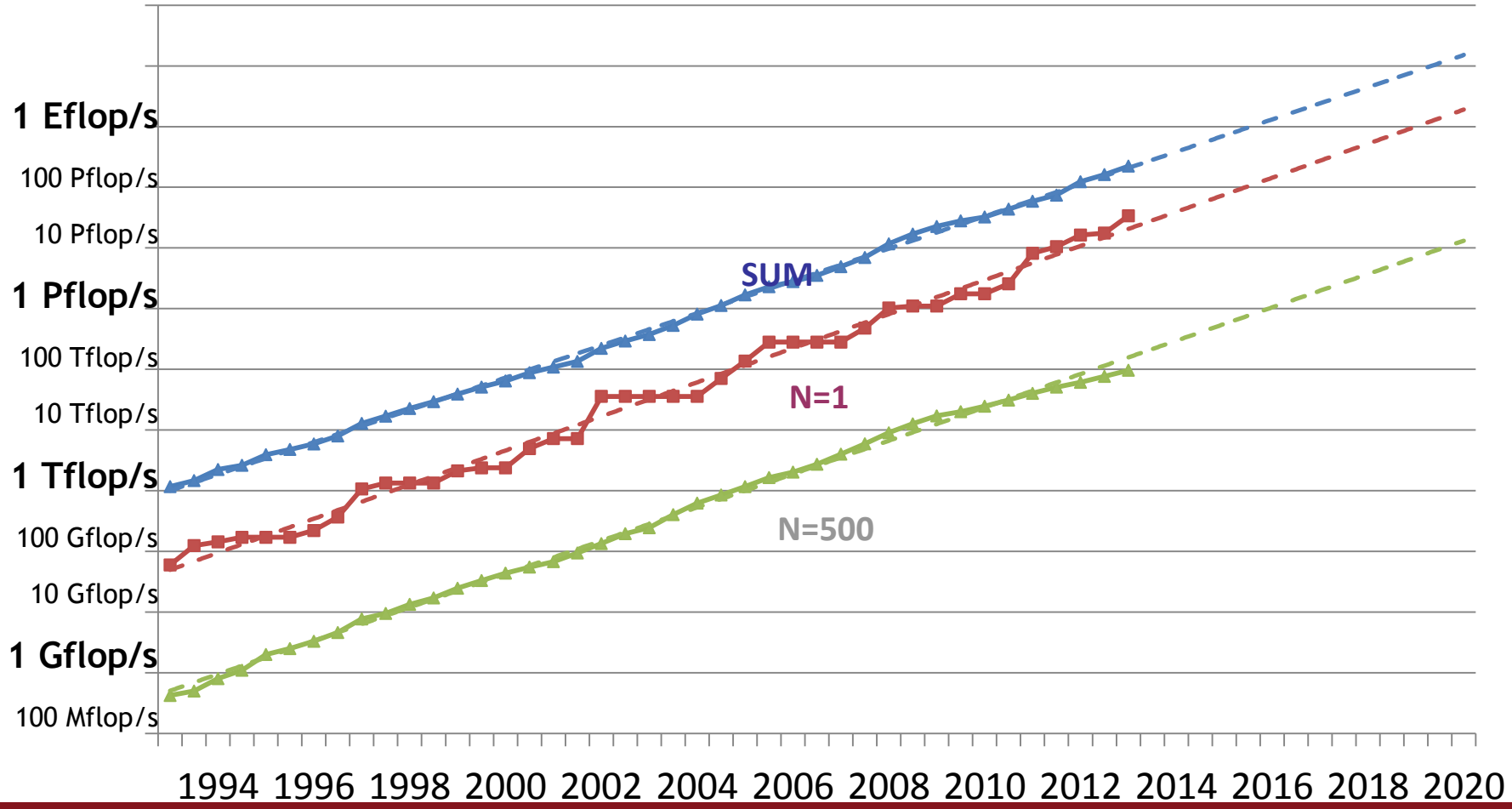
Progress in Turbulence Simulation Capability: Faster Computer ➔ Achievement of Improved Fusion Energy Physics Insights

GTC simulation	Computer name	PE# used	Speed (TF)	Particle #	Time steps	Physics Discovery (*Publication*)
1998	Cray T3E NERSC	10^2	10^{-1}	10^8	10^4	Ion turbulence zonal flow (*Science, 1998*)
2002	IBM SP NERSC	10^3	10^0	10^9	10^4	Ion transport scaling (*PRL, 2002*)
2007	Cray XT3/4 ORNL	10^4	10^2	10^{10}	10^5	Electron turbulence (*PRL, 2007*); EP transport (*PRL, 2008*)
2009	Jaguar/Cray XT5 ORNL	10^5	10^3	10^{11}	10^5	Electron transport scaling (*PRL, 2009*); EP-driven MHD modes
2012-13 (current)	Cray XT5➔Titan ORNL Tianhe-1A (China)	10^5	10^4	10^{12}	10^5	Kinetic-MHD; Turbulence + EP + MHD
2018 (future)	To Extreme Scale HPC Systems		10^6	10^{13}	10^6	Turbulence + EP + MHD + RF

* *Example here of GTC code (Z. Lin, et al.) delivering production runs @ TF in 2002 and PF in 2009*

Courtesy of Bill Tang, Princeton

Exaflops by 2019 (maybe)



Courtesy of Erich Strohmaier LBNL

Practical Constraints for Exascale

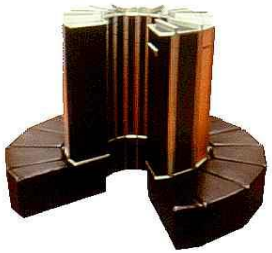
- Sustained Performance
  - Exaflops
  - 100 Petabytes
  - 125 Petabytes/sec.
- Cost
  - Deployment – 8 billion Rubles
  - Operational support
- Power
  - Energy required to run the computer
  - Energy for cooling (remove heat from machine)
  - 20 Megawatts
- Reliability
  - One factor of availability
- Generality
  - How good is it across a range of problems
- Usability
  - How hard is it to program and manage
- Size
  - Floor space – 4,000 sq. meters
  - Access way for power and signal cabling

Where Does Performance Come From?

- Device Technology
  - Logic switching speed and device density
  - Memory capacity and access time
  - Communications bandwidth and latency
- Computer Architecture
  - Instruction issue rate
    - Execution pipelining
    - Reservation stations
    - Branch prediction
    - Cache management
  - Parallelism
    - Parallelism – number of operations per cycle per processor
      - Instruction level parallelism (ILP)
      - Vector processing
    - Parallelism – number of processors per node
    - Parallelism – number of nodes in a system

A History of HPC






1 One OPS

1823 Babbage Difference Engine

1943 Harvard Mark 1

1949 Edsac

10^3 KiloOPS

1951 Univac 1

1959 IBM 7094

10^6 MegaOPS

1964 CDC 6600

1976 Cray 1

10^9 GigaOPS

1982 Cray XMP

1988 Cray YMP

1991 Intel Delta

10^{12} TeraOPS

1996 T3E

1997 ASCI Red

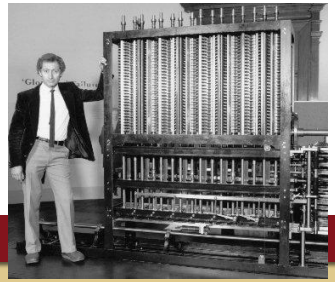
2001 Earth Simulator

2003 Cray X1

10^{15} PetaOPS

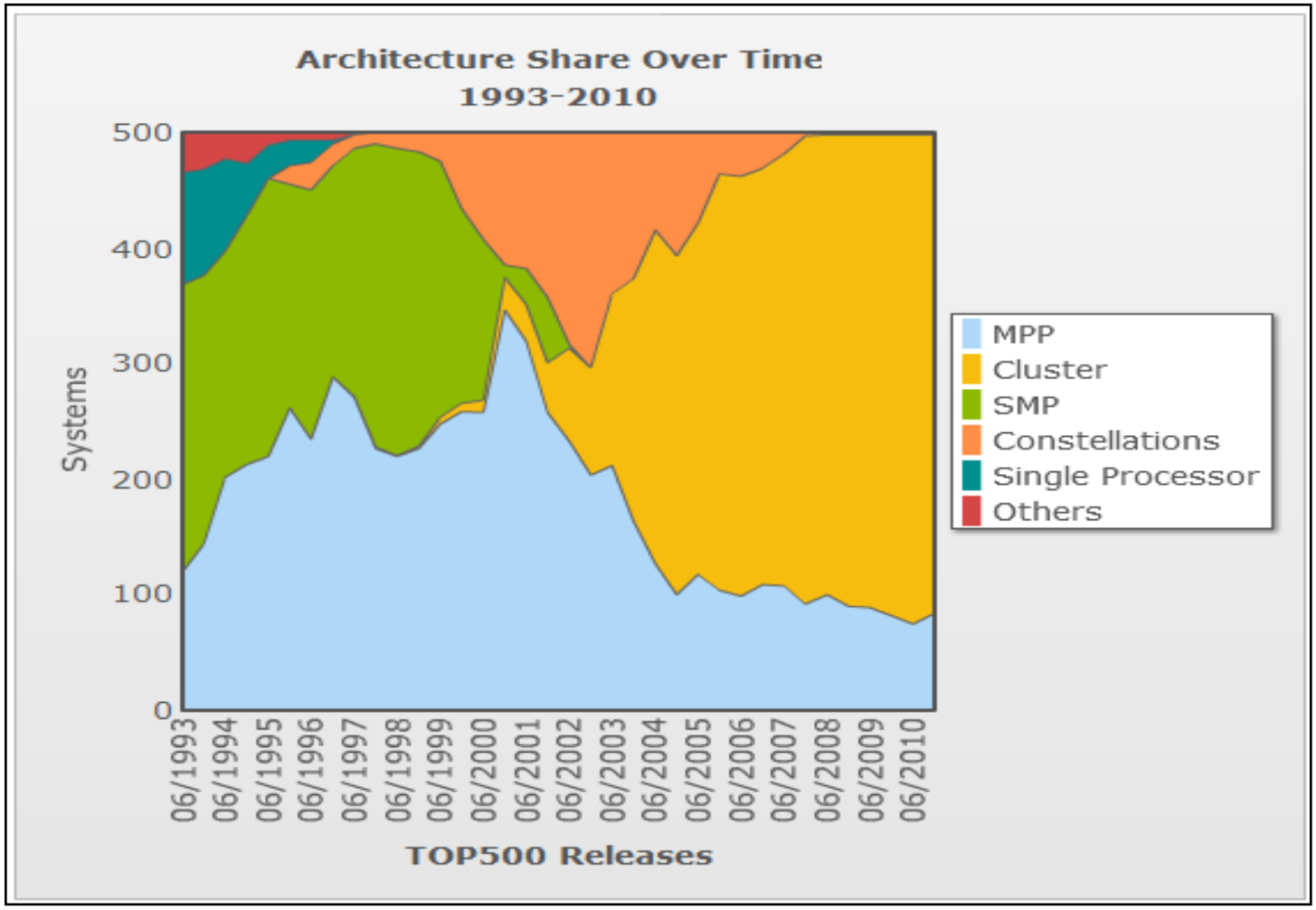
2006 BlueGene/L

2009 Cray XT5



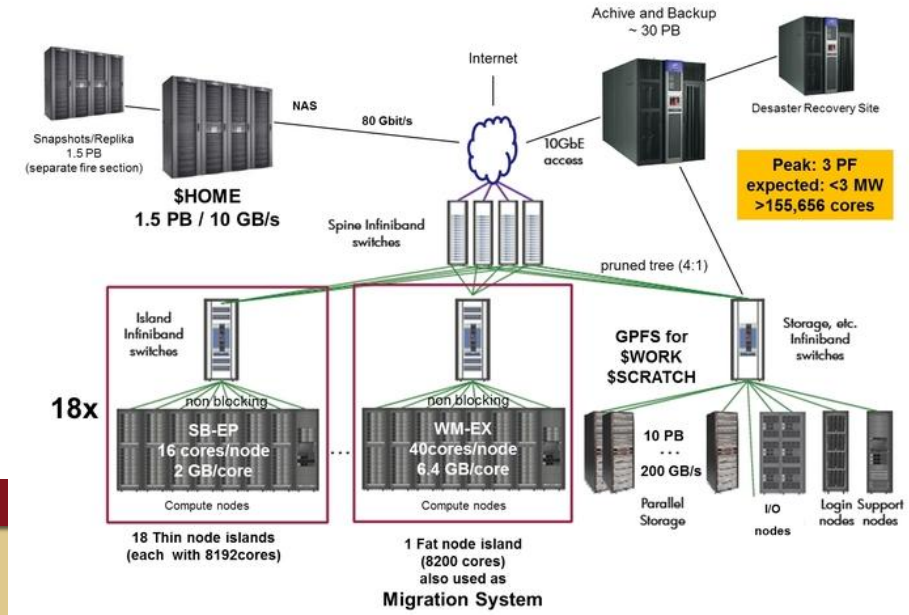


Clusters Dominate HPC System Architecture



An HPC Cluster

- Name: SuperMUC
- Top-500 Rank: 6 (Nov. 2012)
- Site: Leibniz Rechenzentrum
- Type: iDataPlex DX360M4
- System URL: http://www.lrz.de
- Manufacturer: IBM
- Linpack Performance (Rmax) 2.897 Pflop/s
- Processor: Intel Xeon E5-2680, 2.7 GHz
- Cores: 147,456/155,656
- Theoritical Peak (Rpeak): 3.185 Pflop/s
- Power: 3.423 Megawatts
- Memory: > 300 TB RAM
- Interconnect: Infiniband FDR10
- Operating System: Linux



Strengths and Limitations of Commodity Clusters

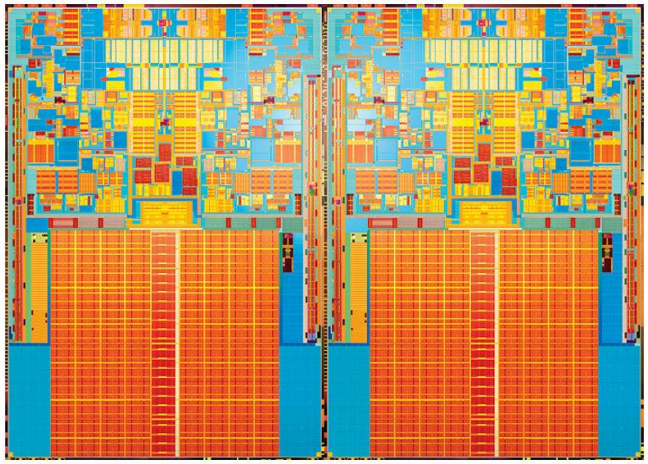
Strengths

- Excellent performance to cost
- Exploits economy of scale
  - Thru mass-production of components
  - Many competing cluster vendors
- Flexible Just-in-place configuration
  - Scalable up and down
- Rapid tracking of technology
  - First to exploit newest components
- Programmable
  - Uses industry standard programming languages and tools
- User empowerment
  - Low cost, ubiquitous systems
  - Programming systems make it relatively easy to program for expert users
- 82.2% of TOP-500 deployed systems

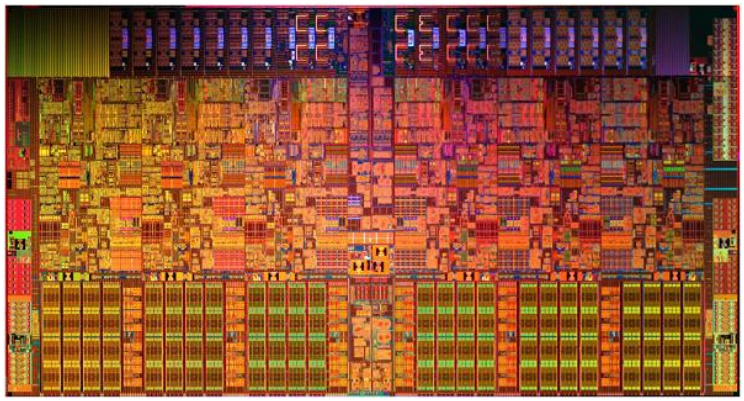
Limitations

- Not ideal for all workload classes
- Reduced efficiencies for many problems
  - Compared to MPPs
  - Due to networks exhibiting lower bandwidth and higher latencies & overheads
- Lower space density
  - Takes up more room (floor space) than comparable peak performance MPP

Multi-core Intel Xeon



4-core Clovertown package

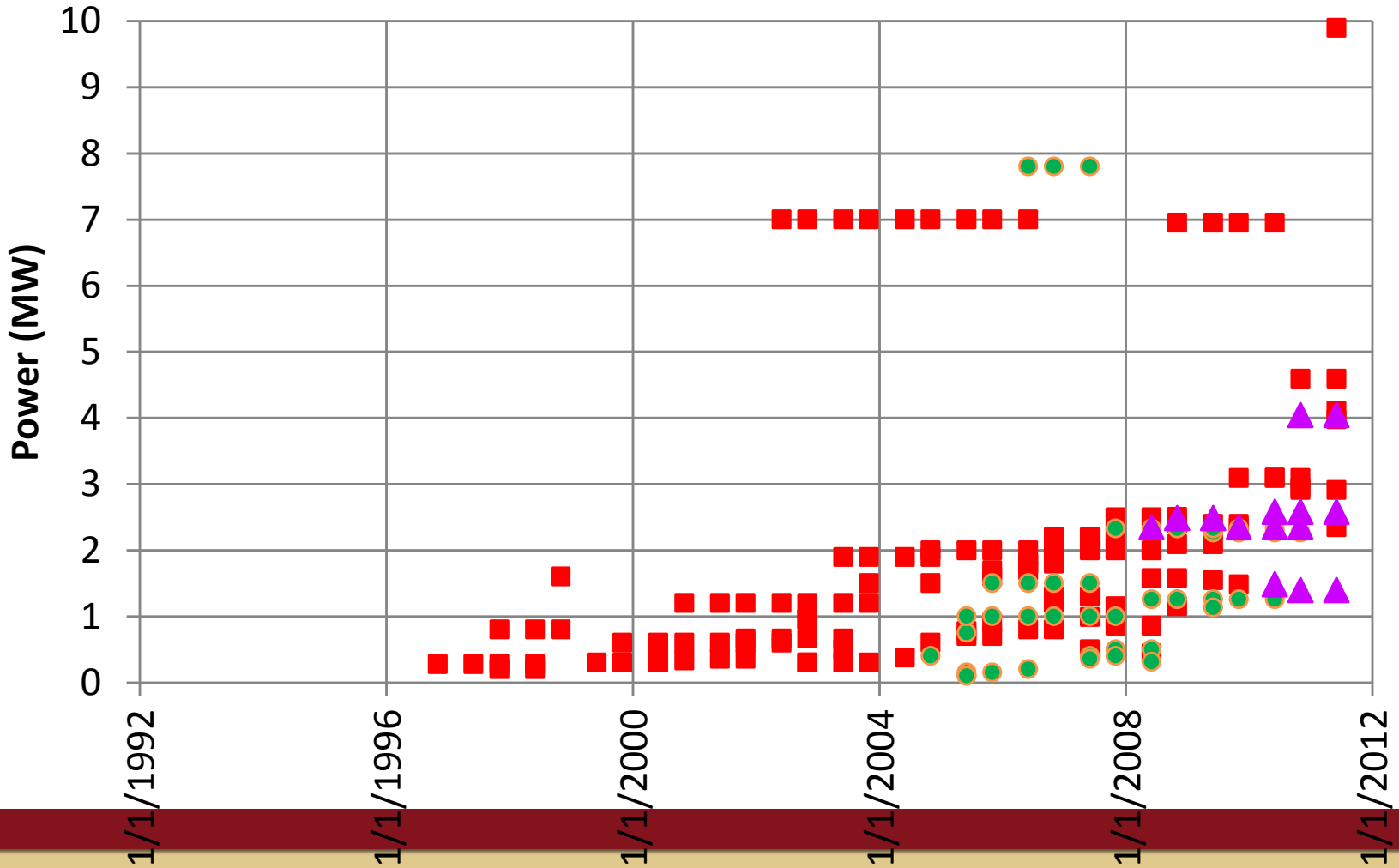


6-core Westmere die

- EMT64 processors for servers and workstations
- Popular in many installations Quad-core Clovertown was introduced in late 2006
  - Based on Core2 architecture
  - Two dual-core Woodcrest chips in one package
  - 48 GFLOPS peak at 3GHz
  - 582 mil. transistors in 65nm technology
  - 1333 MHz bus speed
  - 150W TDP at 3GHz, 80W at 2.33GHz or less
- Westmere lineup (launched in 2010) enabled efficient multiprocessor configurations
  - 4- and 6-core version of Nehalem architecture
  - QPI for intra-node coherency traffic
  - 83.04 GFLOPS peak at 3.46GHz
  - 1.17 bil. transistors in 32nm (6-core)
  - 3-channel DDR3 memory controller at 1333MHz (32GB/s)
  - 130W TDP for 6-core at 3.47GHz (or 95W at 3.07GHz)
  - Westmere-EX increased number of cores to 10 per die

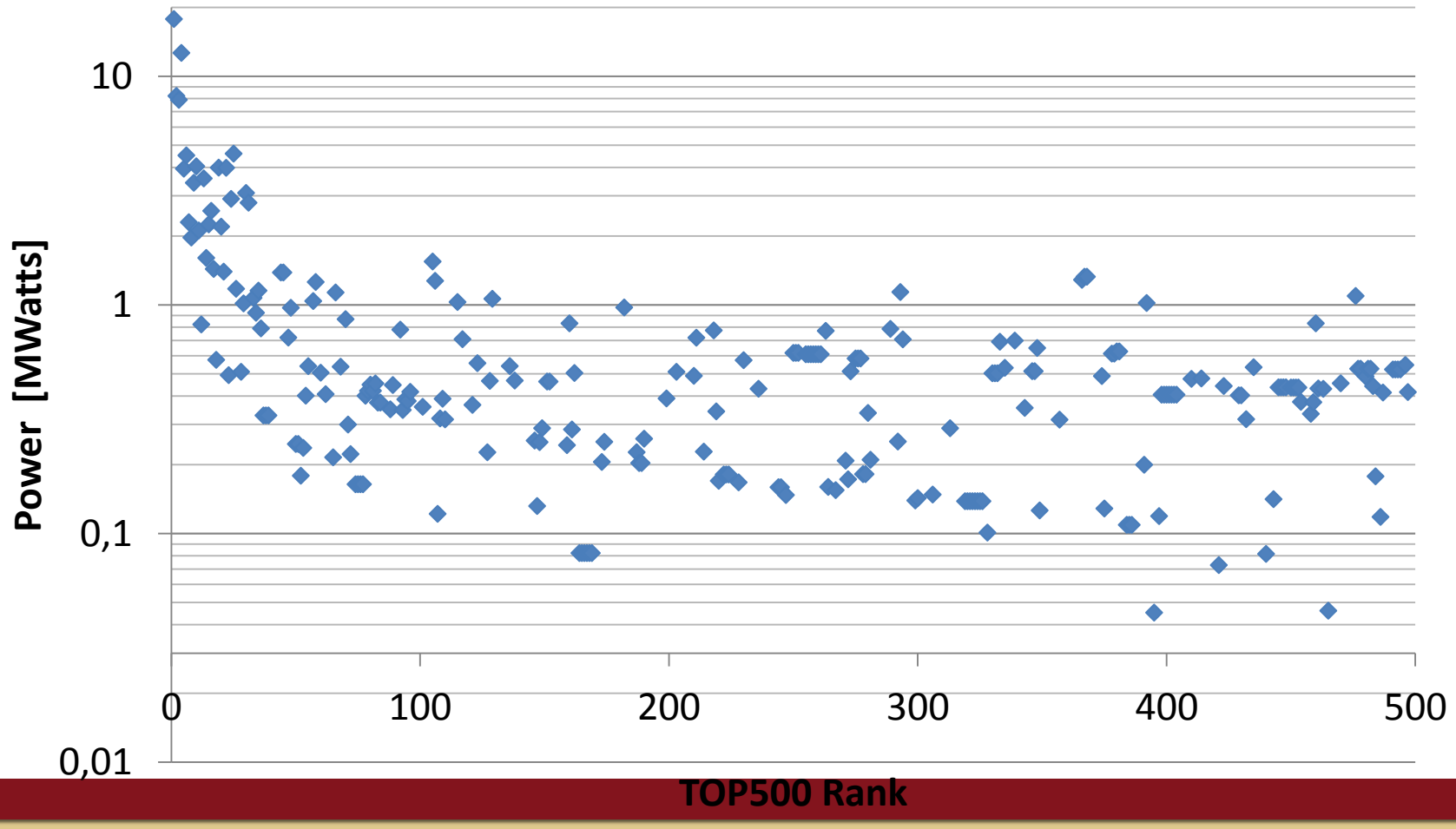
Courtesy of Peter Kogge, UND

Total Power



Courtesy of Erich Strohmaier LBNL

Absolute Power Levels



Technology Demands new Response

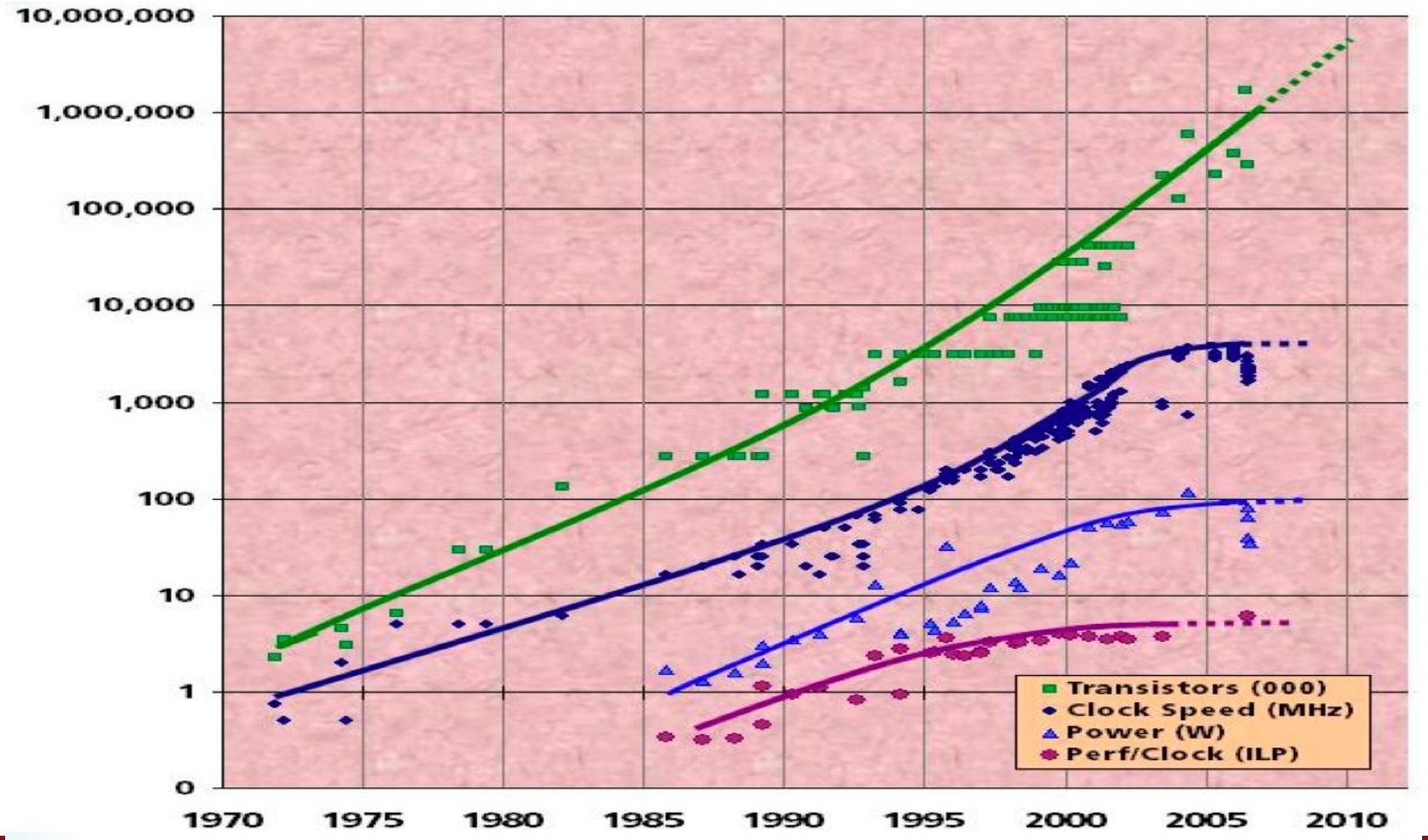
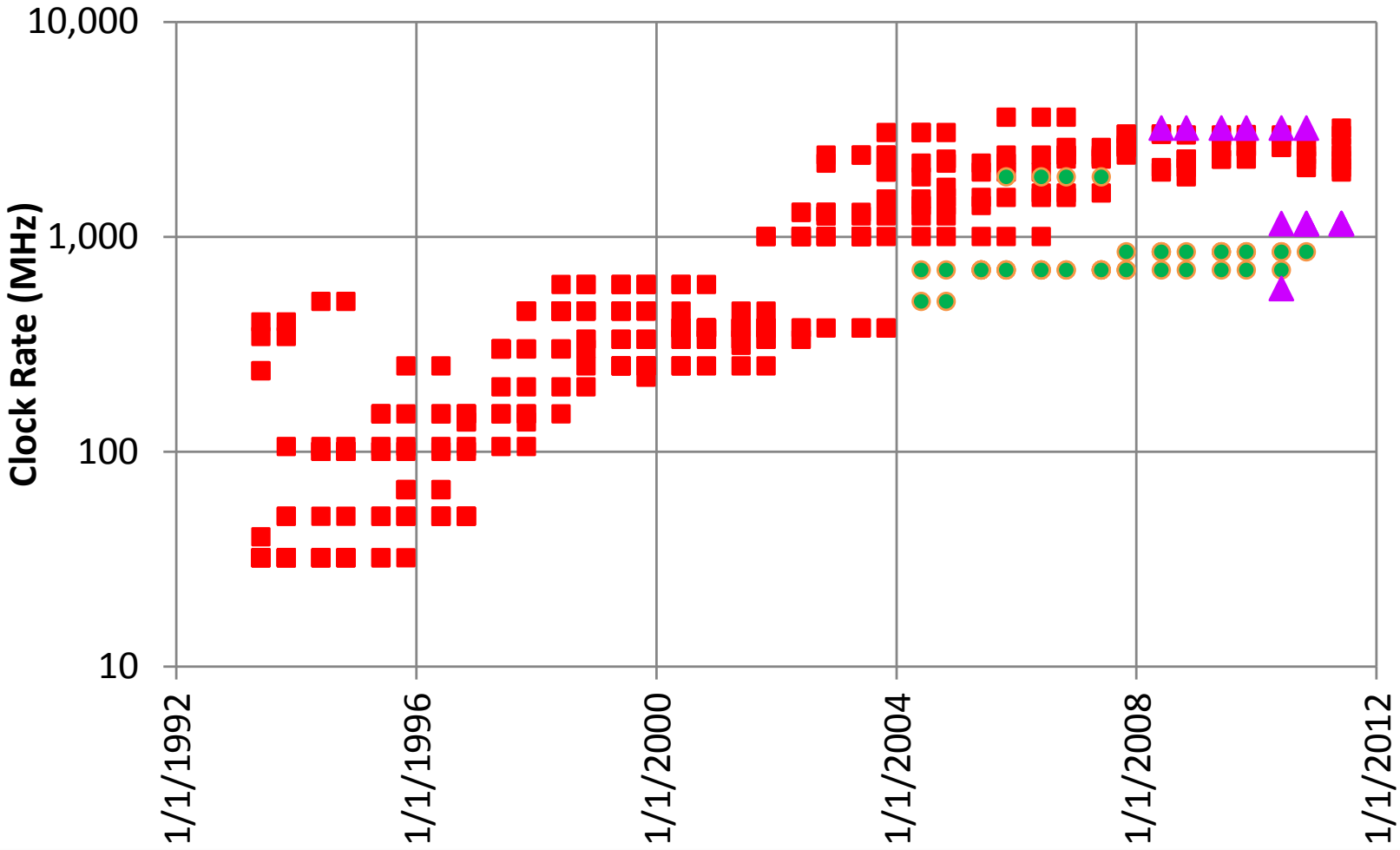


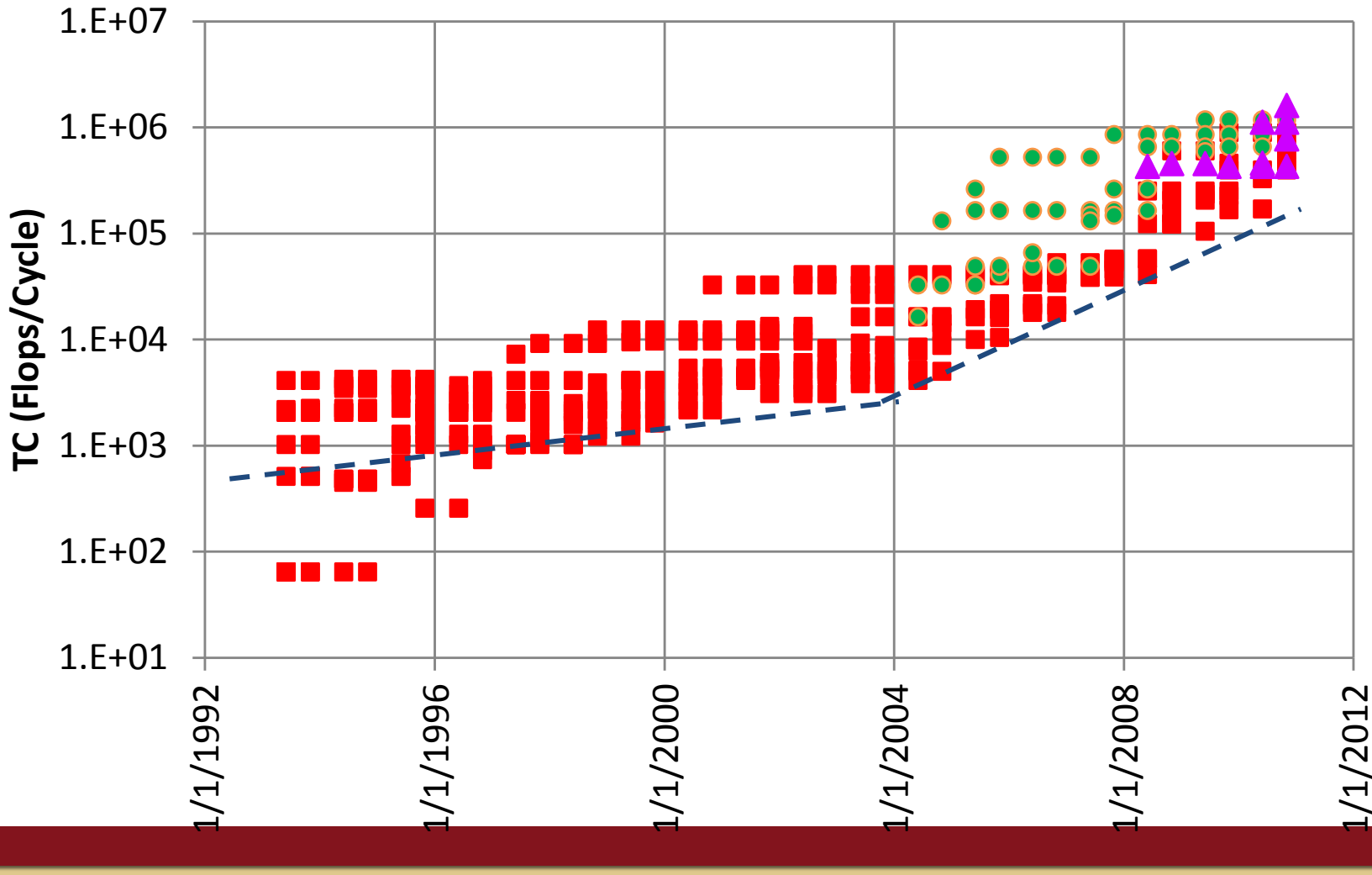
Figure courtesy of Kunle Olukotun, Lance Hammond, Herb Sutter, and Burton Smith

Clock Rate



Courtesy of Peter Kogge, UND

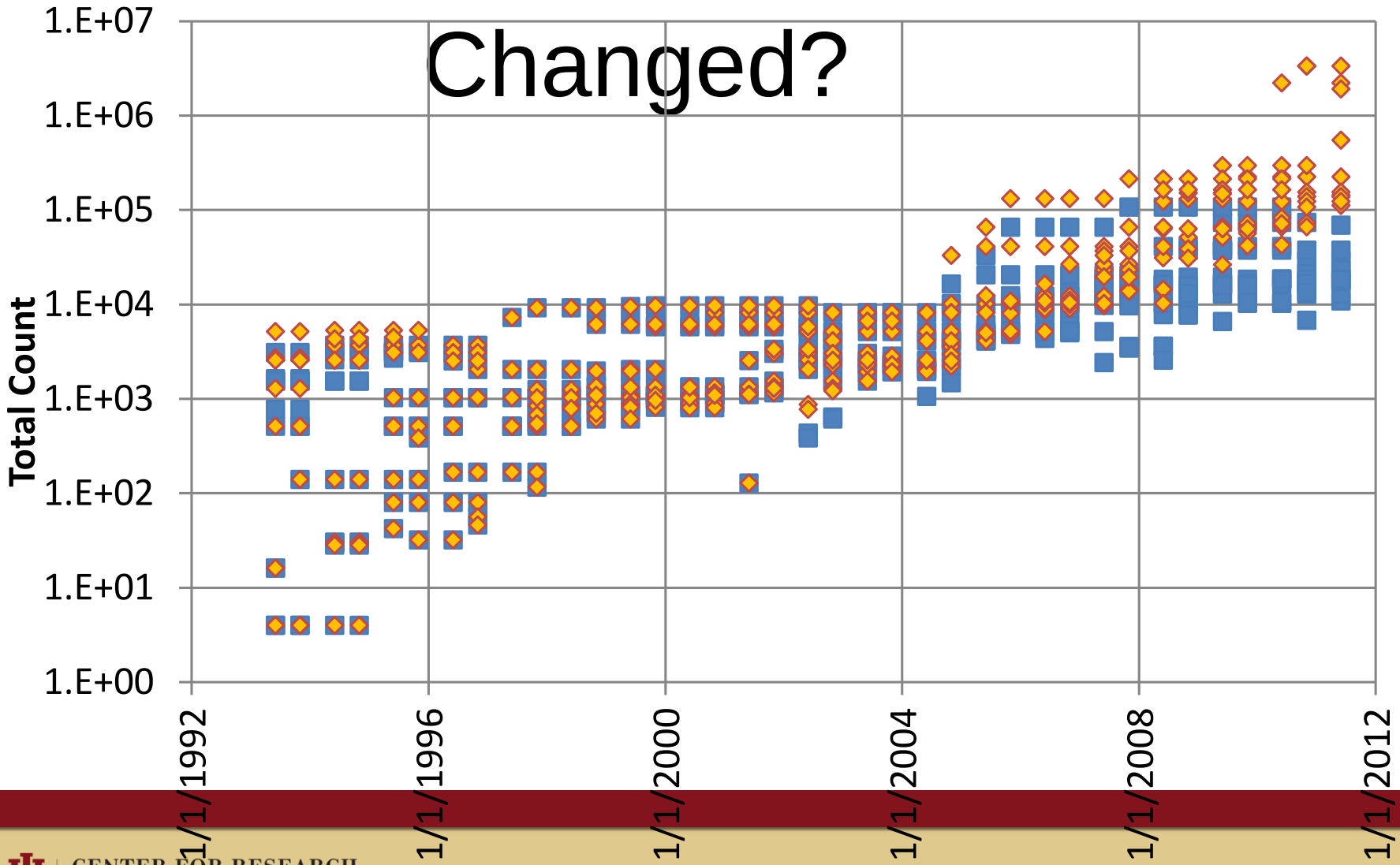
Total Concurrency



Courtesy of Peter Kogge,

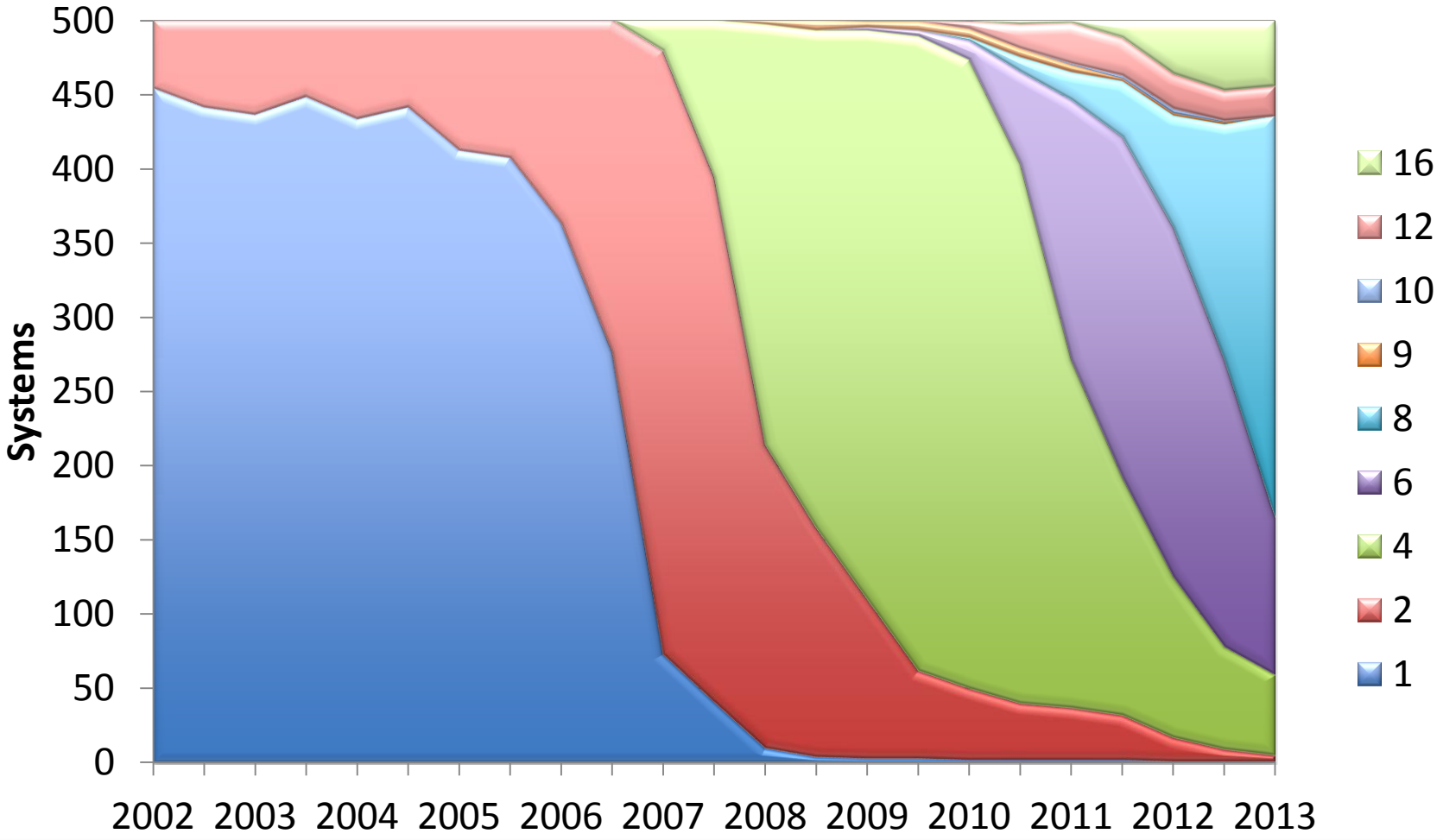
Courtesy of Peter Kogge, UND

How Has "Processor Count" Changed?



Socket Count is Flat, Core Count is Rising

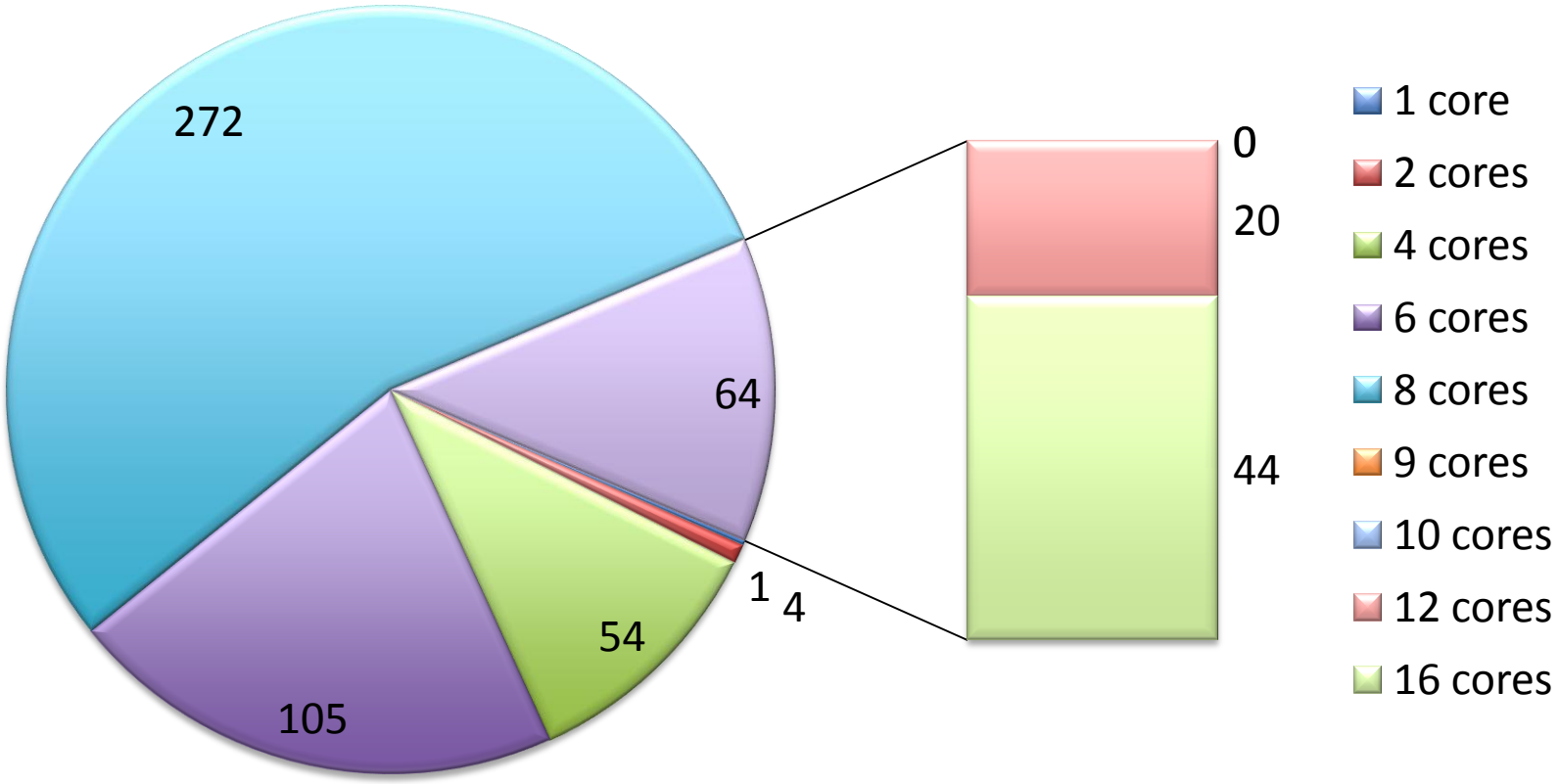
Cores per Socket



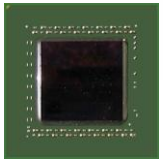
Courtesy of Erich Strohmaier LBNL

Courtesy of Erich Strohmaier LBNL

Cores per Socket



Nvidia Tesla



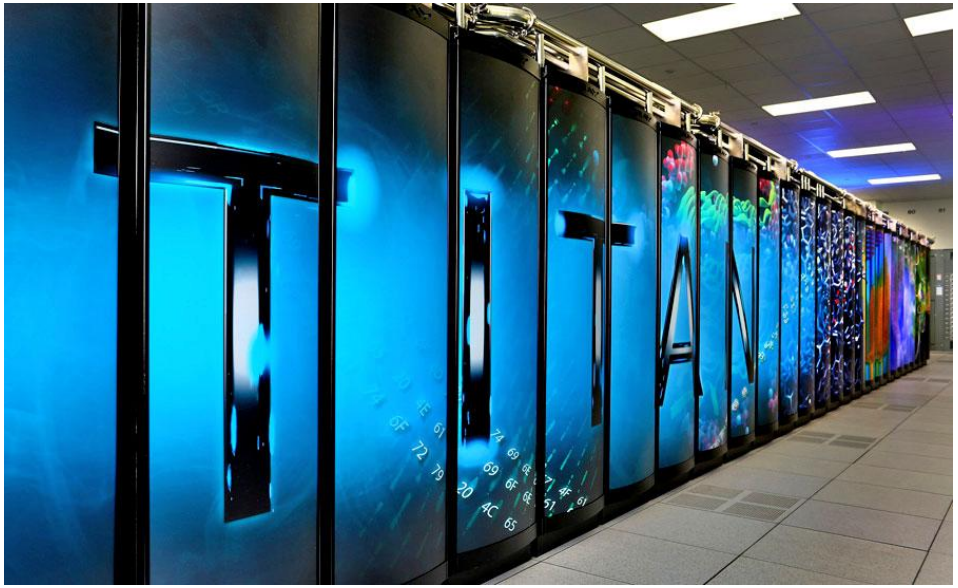
Tesla C870

K20X Kepler

- GPGPU accelerator
- Major technology revisions: Tesla (original), Fermi, and Kepler
- C870 introduced in 2007 was based on GeForce 8 (G80) shader architecture
  - 128 thread processors at 1350 MHz
  - 518.4 single-precision GFLOPS (no dual percision)
  - 1.5 GB GDDR3 memory with 76.8 GB/s throughput
  - 170.9W TDP
- The newest K20X, based on GK110 architecture, launched in 2012
  - 2688 thread processors at 732 MHz
  - 3950 GFLOPS (single-precision), 1310 GFLOPS double-precision FMA
  - 6144 GB GDDR5 memory, throughput 250 GB/s
  - 235W TDP

Titan

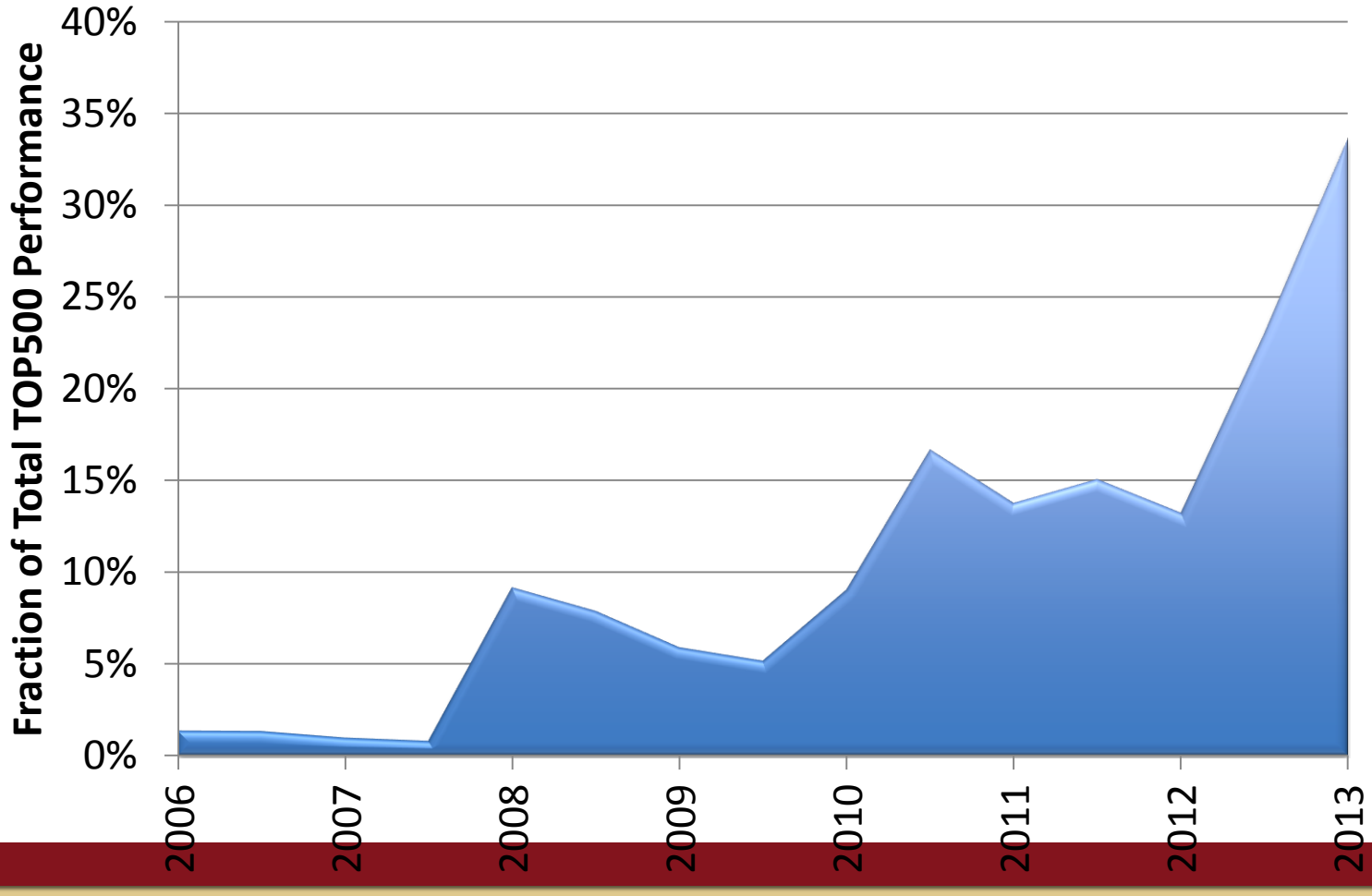
- **USA, 2012**: *Jaguar avenged*
- Manufactured by Cray for ORNL
- Hybrid design with AMD Opterons and Nvidia Tesla GPGPUs
- Cray XK7 architecture:
  - 299,008 Opteron 6274 CPU cores
  - 18,688 Tesla K20 (Kepler) accelerators
  - 710,144 GB memory (598,016 GB DDR3 attached to processors; 112,128 GB GDDR5 on GPU boards)
  - 17,590 TFLOPS in HPL (#1 in November 2012)
  - Theoretical peak: 27,112.5 TFLOPS
  - Cray Gemini interconnect
  - 40 PB storage at aggregate 1.4 TB/s
  - 200 cabinets in 404m^2 area
  - $97 million contractual cost



OAK RIDGE National Laboratory

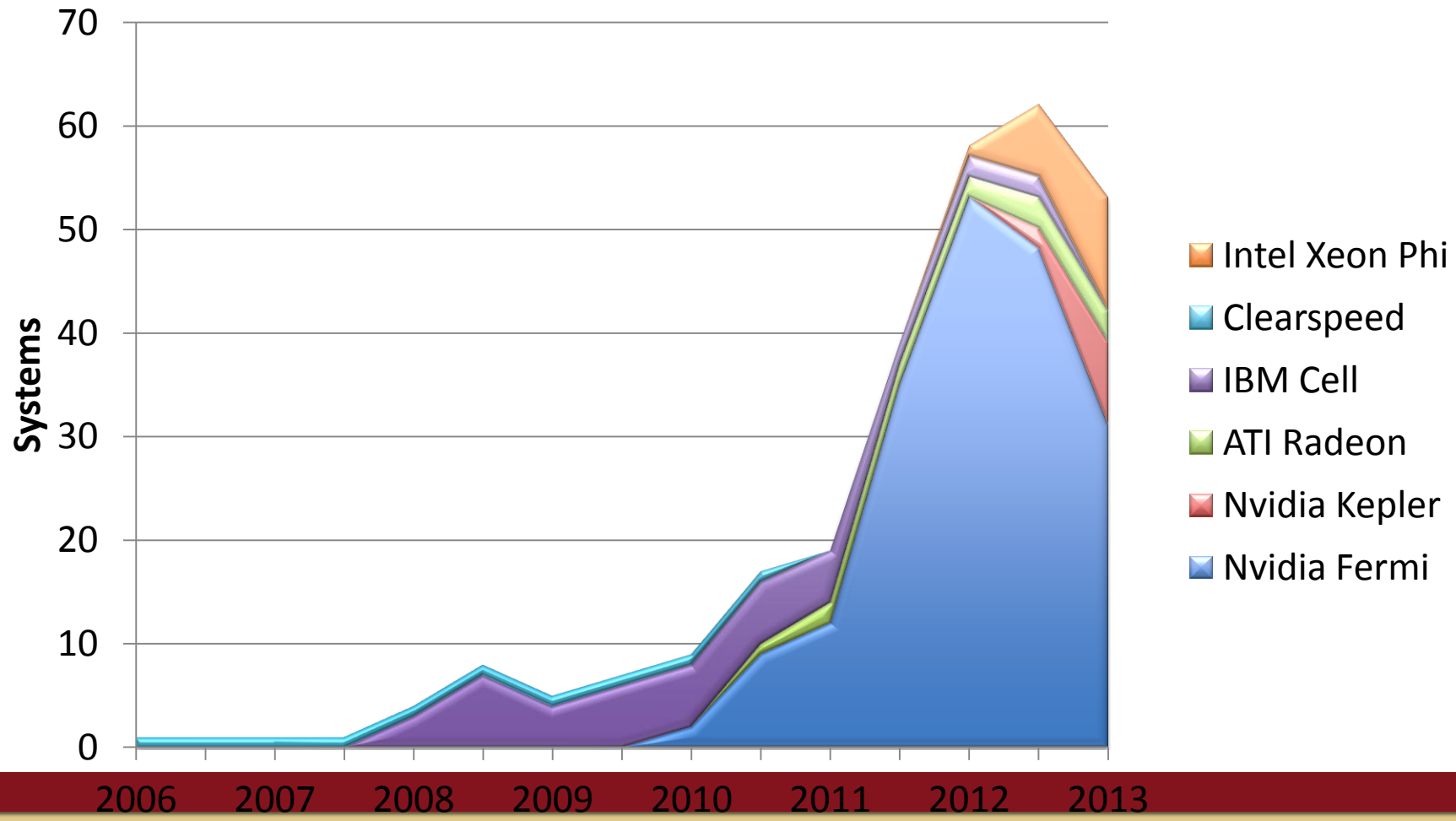
Courtesy of Erich Strohmaier LBNL

Performance Share of Accelerators



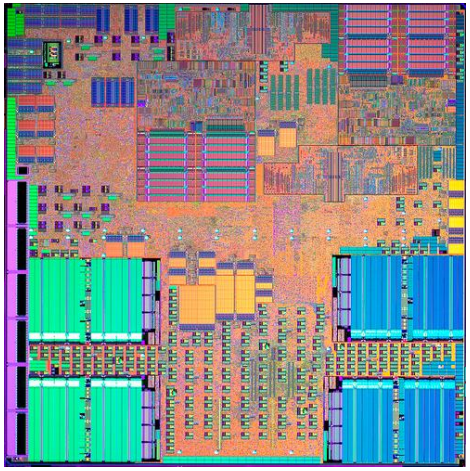
Courtesy of Erich Strohmaier LBNL

Accelerators

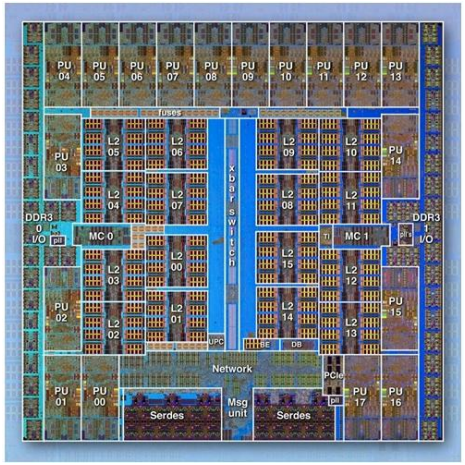


IBM PowerPC

BG/L PowerPC440 die



BG/Q PowerPC A2 die



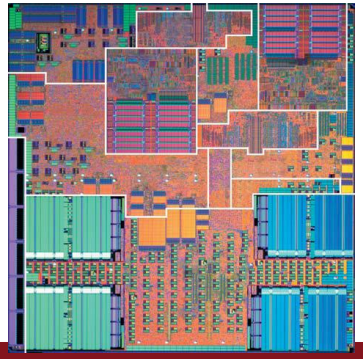
- Power efficient line of RISC cores
- Blue Gene/L ASIC used 32-bit PowerPC 440
  - Dual-core modules with two FPUs per core
  - Not cache-coherent
  - 5.6 GFLOPS at 700 MHz
  - IBM Cu-11 0.13 µm process, 95 mil. transistors
  - 12W power usage per ASIC
- Blue Gene/P utilized PowerPC 450 cores
  - Cache-coherent across ASIC (4 cores)
  - 13.6 GFLOPS at 850 MHz
  - 208 mil. transistors in IBM Cu-08 90nm process
  - 16W power
- Blue Gene/Q: 64-bit PowerPC A2
  - 18 cores
  - 204.8 GFLOPS at 1.6 GHz
  - 1.47 billion transistors in 45 nm
  - 55 W power draw

Courtesy of Peter Kogge, UND

"Light Weight" Strawman

2 Nodes per "Compute Card." Each node:
- **A low power compute chip**
- **Some memory chips**
- **"Nothing Else"**

System Architecture:
- **Multiple Identical Boards/Rack**
- **Each board holds multiple Compute Cards**
- **"Nothing Else"**



- **2 simple dual issue cores**
- **Each with dual FPUs**
- **Memory controller**
- **Large eDRAM L3**
- **3D message interface**
- **Collective interface**
- **All at subGHz clock**

"Packaging the Blue Gene/L supercomputer," IBM J. R&D, March/May 2005
"Blue Gene/L compute chip: Synthesis, timing, and physical design," IBM J. R&D, March/May 2005

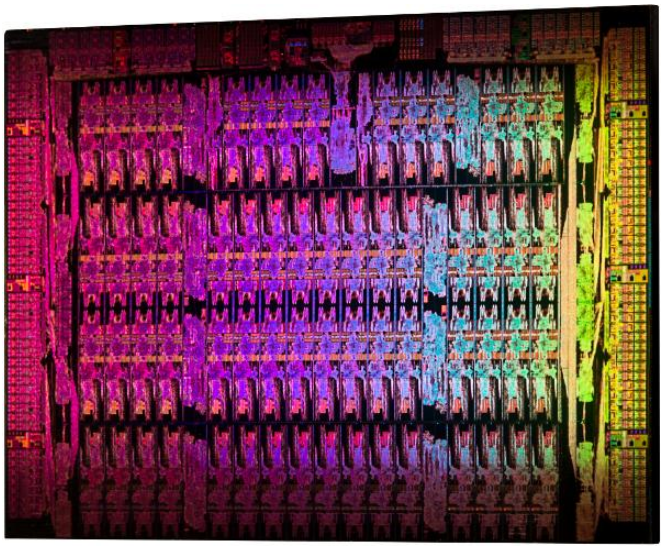
Sequoia

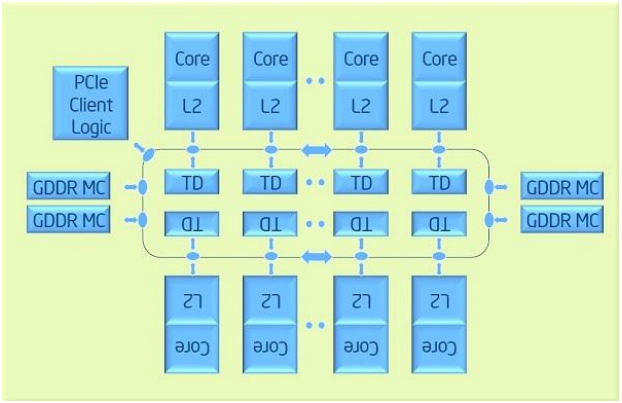
- **USA, 2012**: *BlueGene strikes back*
- Built by IBM for NNSA and installed at LLNL
- 20,123.7 TFLOPS peak performance
  - Blue Gene/Q architecture
  - 1,572,864 total PowerPC A2 cores
  - 98,304 nodes in 96 racks occupy 280m^2
  - 1,572,864 GB DDR3 memory
  - 5-D torus interconnect
  - 768 I/O nodes
  - 7890kW power, or 2.07 GFLOPS/W
  - Achieves 16,324.8 TFLOPS in HPL (#1 in June 2012), about 14 PFLOPS in HACC (cosmology simulation), and 12 PFLOPS in Cardioid code (electrophysiology)

Lawrence Livermore National Laboratory

National Nuclear Security Administration

Intel Many Integrated Core (MIC)

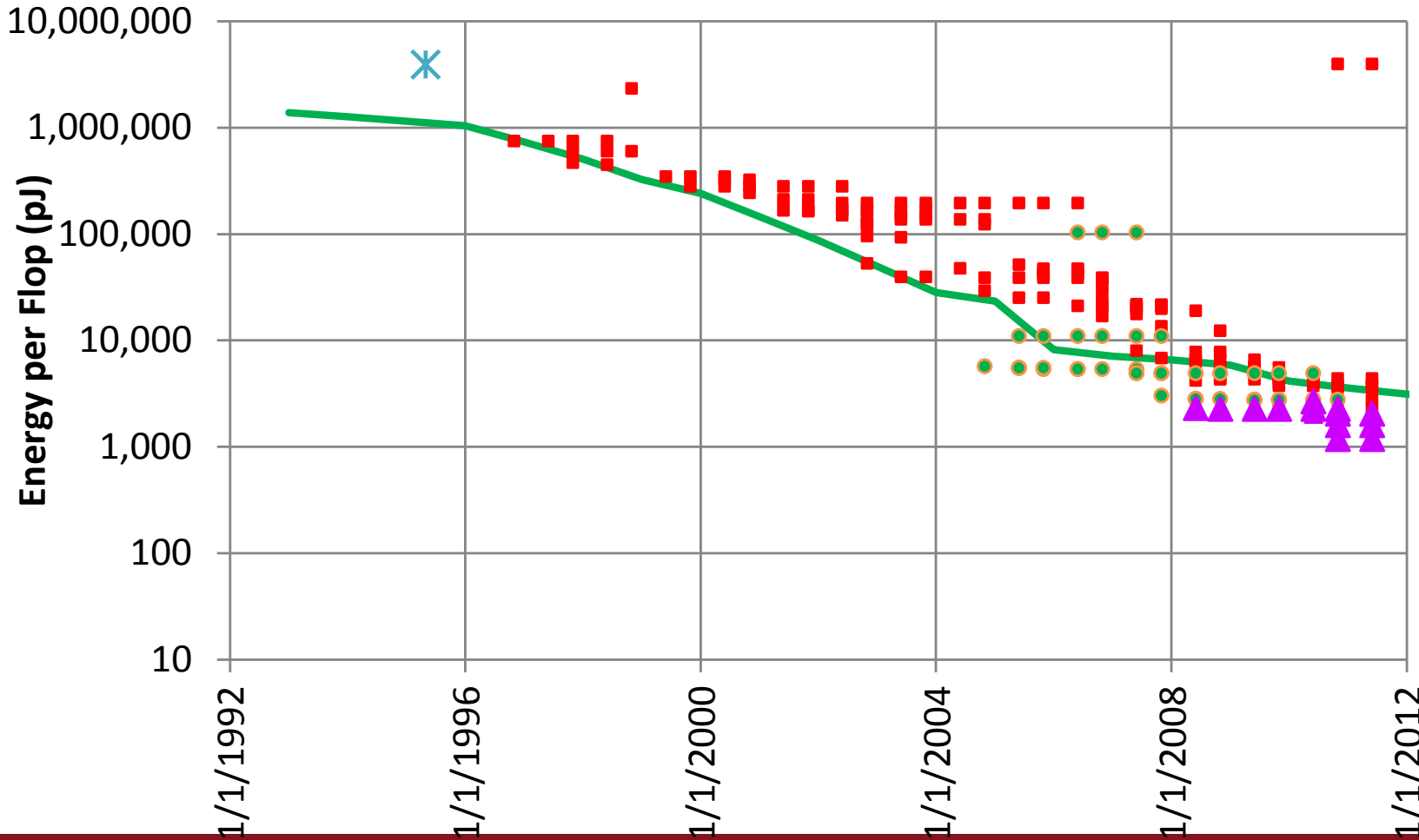




- Based on Larrabee many core architecture
- Official branding: Xeon Phi
- Modified Pentium P54C cores
  - Up to 61 cores per die, 4 threads per core
  - Up to 1220 GFLOPS at 1.25 GHz
  - Max. 8 GB GDDR5 memory at 5.5 GHz, 512 bit bus
  - 22nm technology, approx. 5 billion transistors
  - 225-300W TDP
  - Available as PCIe x16 board
- Main processing component of Tianhe-2 and TACC Stampede

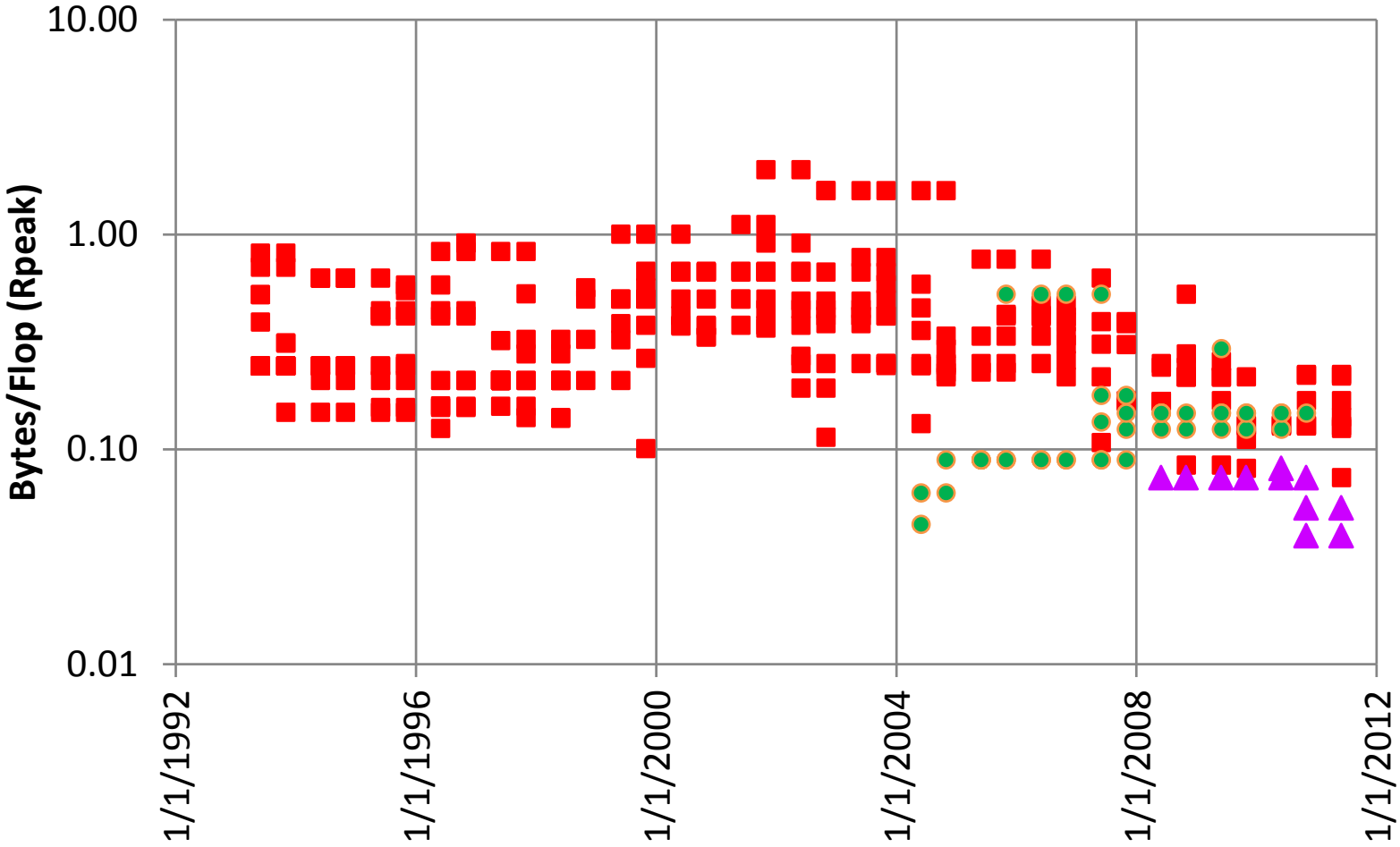
Courtesy of Peter Kogge, UND

Energy per Flop



Courtesy of Peter Kogge, UND

Bytes per Flops (Peak)



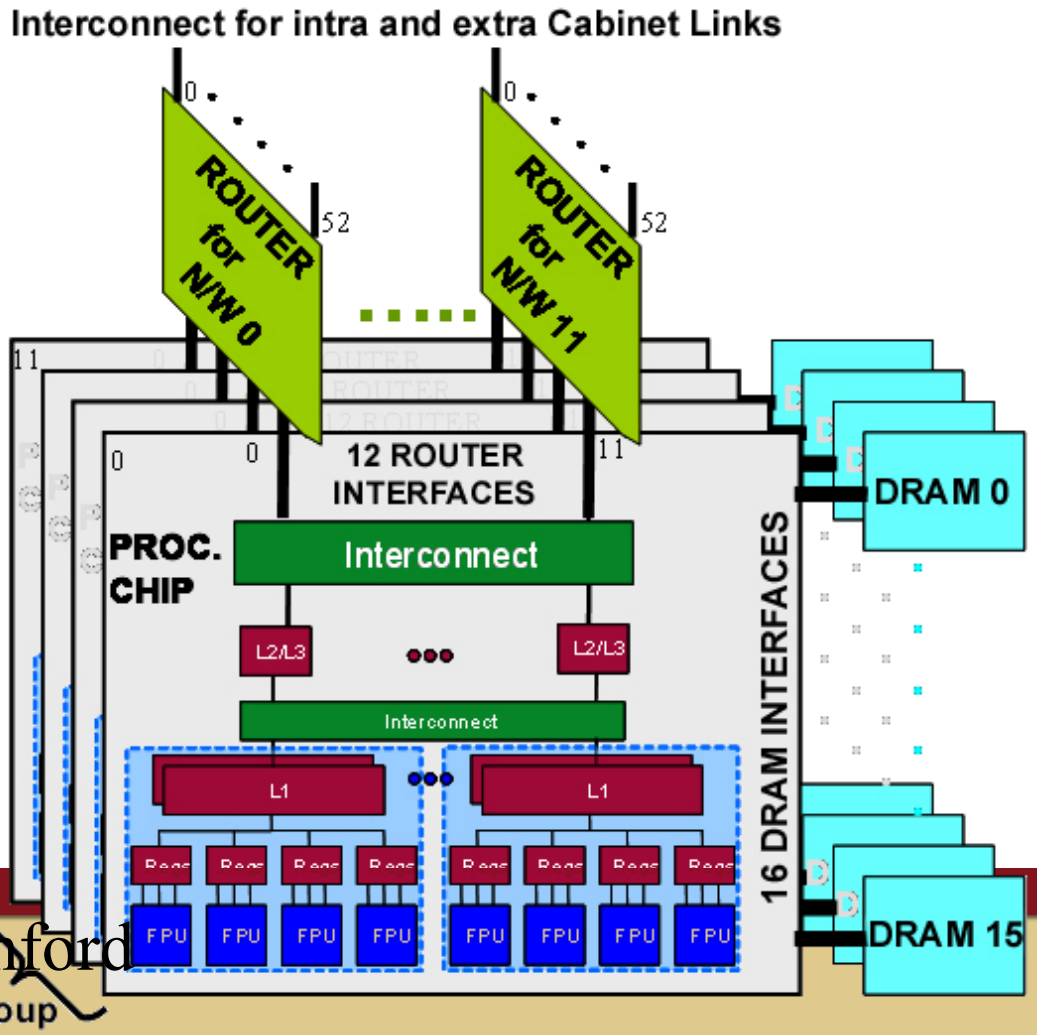
Systems are getting less memory rich

Courtesy of Peter Kogge, UND

1 EFlop/s "Clean Sheet of Paper" Strawman

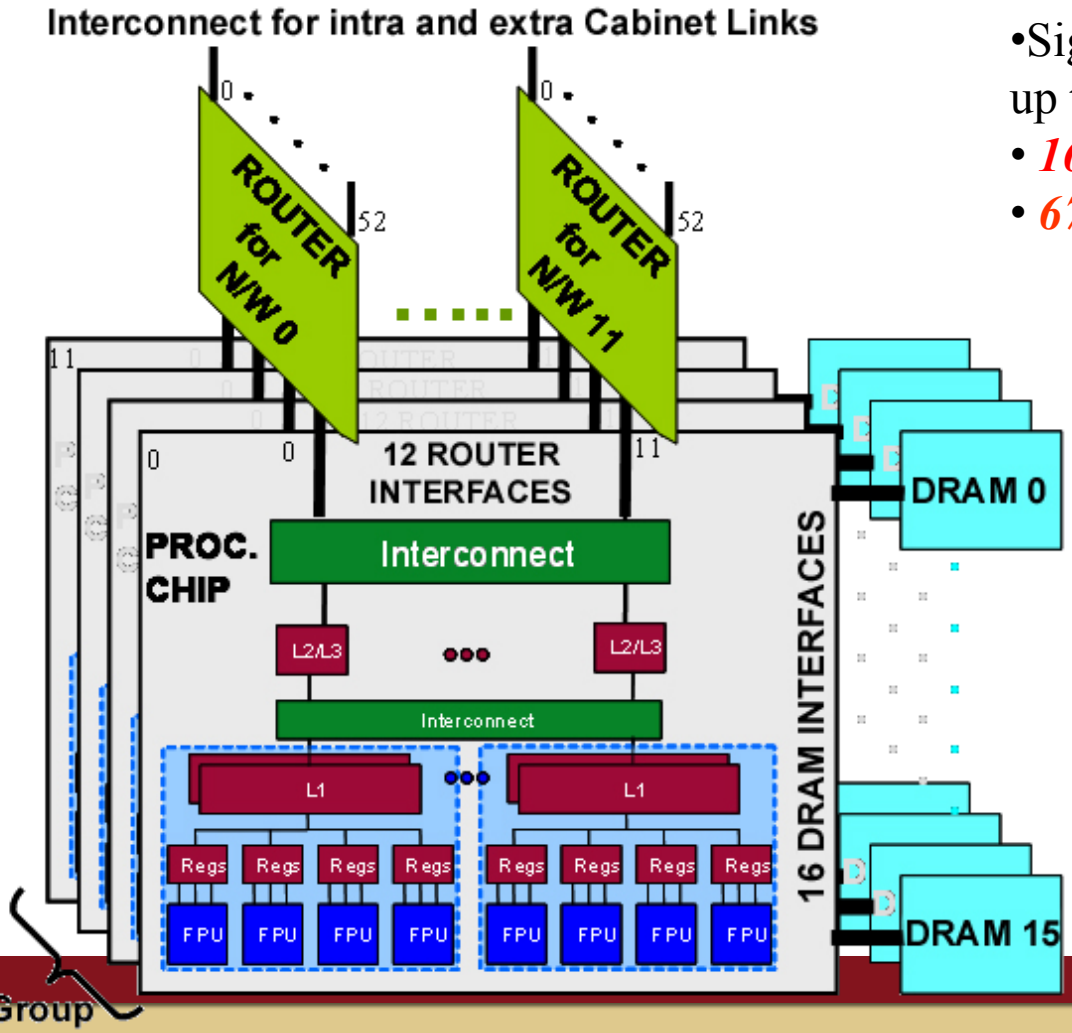
Sizing done by "balancing" power budgets with achievable capabilities

- 4 FPUs+RegFiles/Core (=6 GF @1.5GHz)
- **1 Chip = *742* Cores** (=4.5TF/s)
  - 213MB of L1I&D; 93MB of L2
- 1 Node = 1 Proc Chip + 16 DRAMs (16GB)
- 1 Group = 12 Nodes + 12 Routers (=54TF/s)
- 1 Rack = 32 Groups (=1.7 PF/s)
  - 384 nodes / rack
- 3.6EB of Disk Storage included
- 1 System = 583 Racks (=1 EF/s)
  - **166 MILLION cores**
  - 680 MILLION FPUs
  - 3.6PB = 0.0036 bytes/flops
  - **68 MW** w'aggressive assumptions

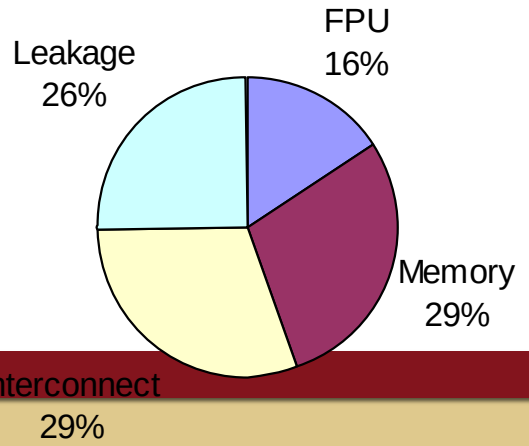


Largely due to Bill Dally, Stanford

The Exascale Strawman

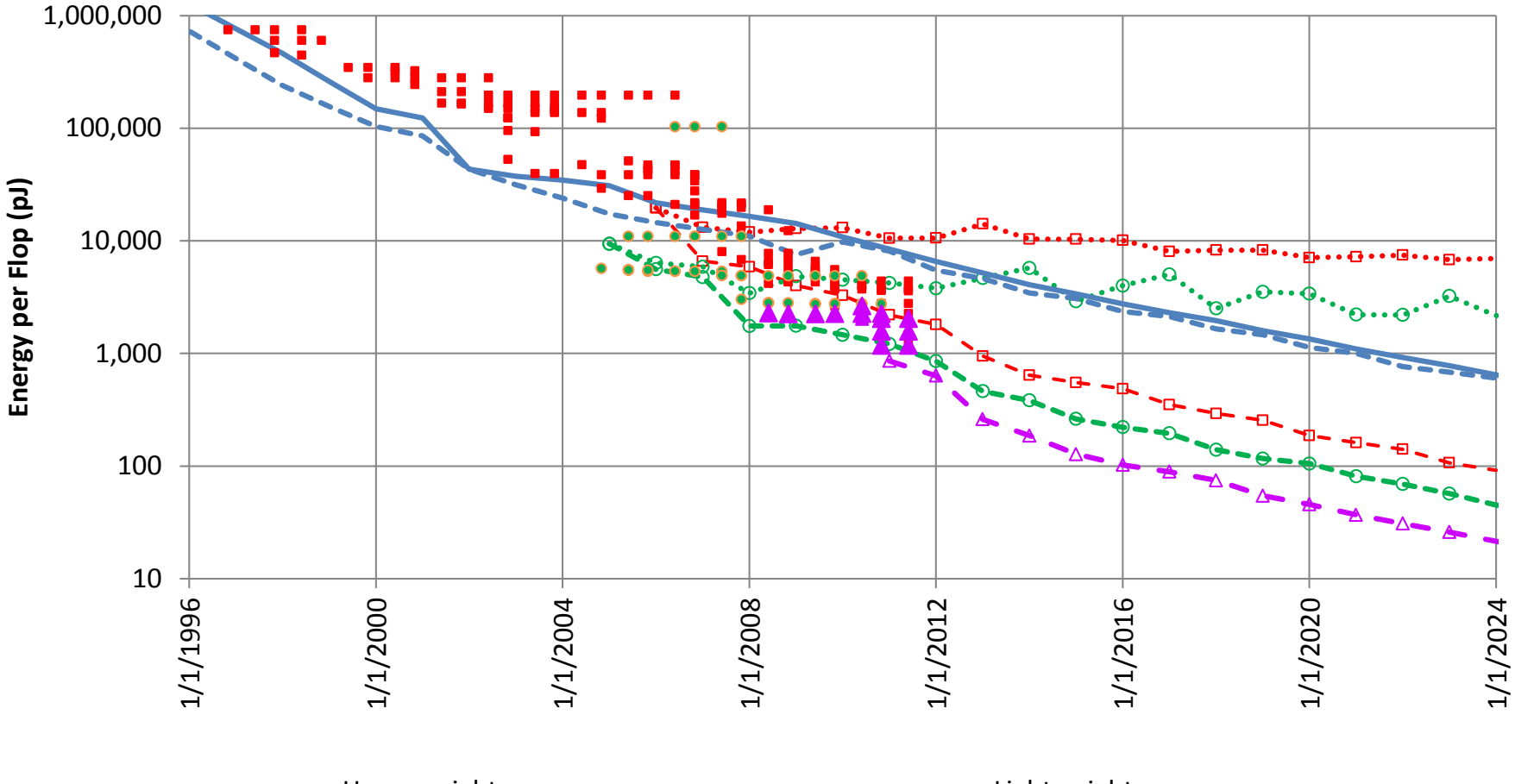


- Significant tail in bandwidth going up the memory hierarchy
- ***166 million cores***
- ***67 Mwatts = 67pJ/flop***
  - ***2/3 in memory & Interconnect***



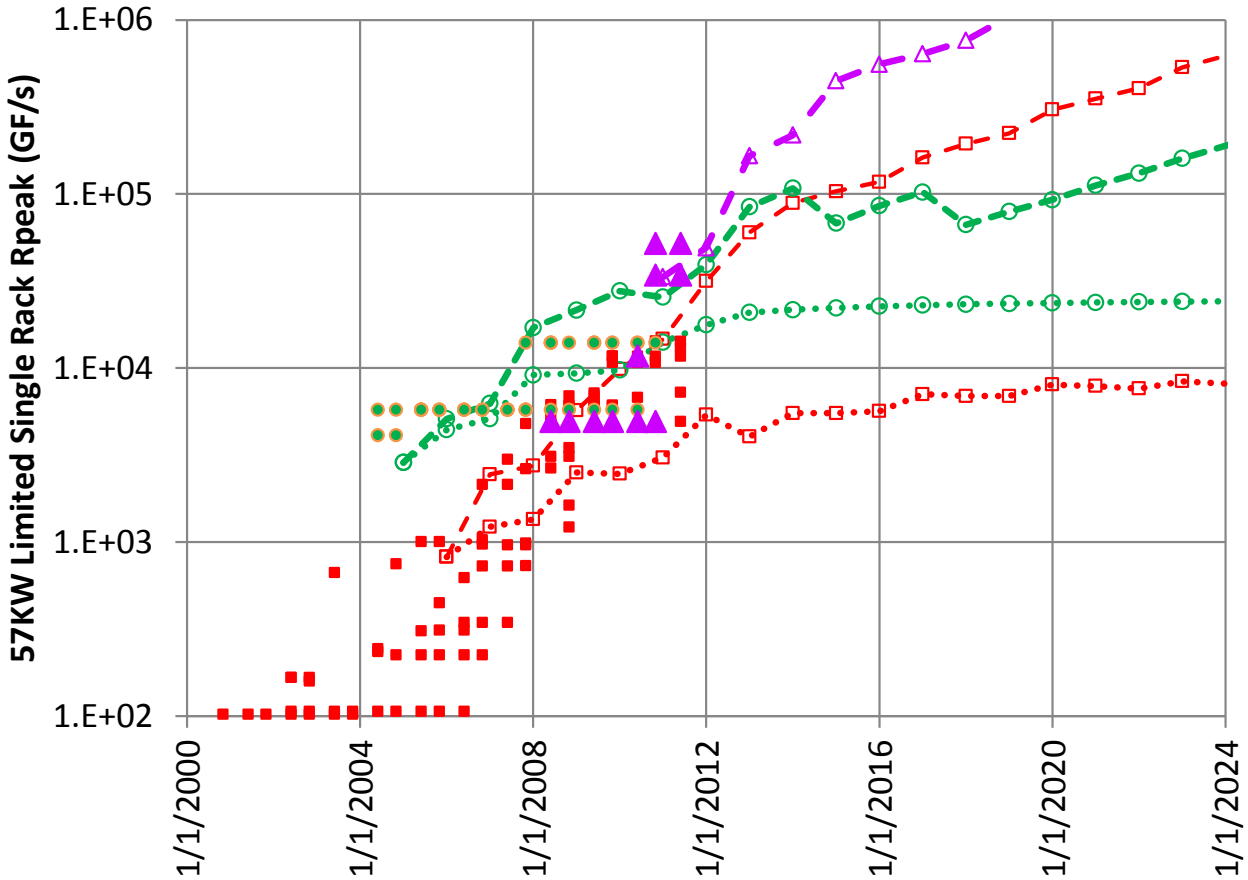
Courtesy of Peter Kogge, UND

Energy Projections



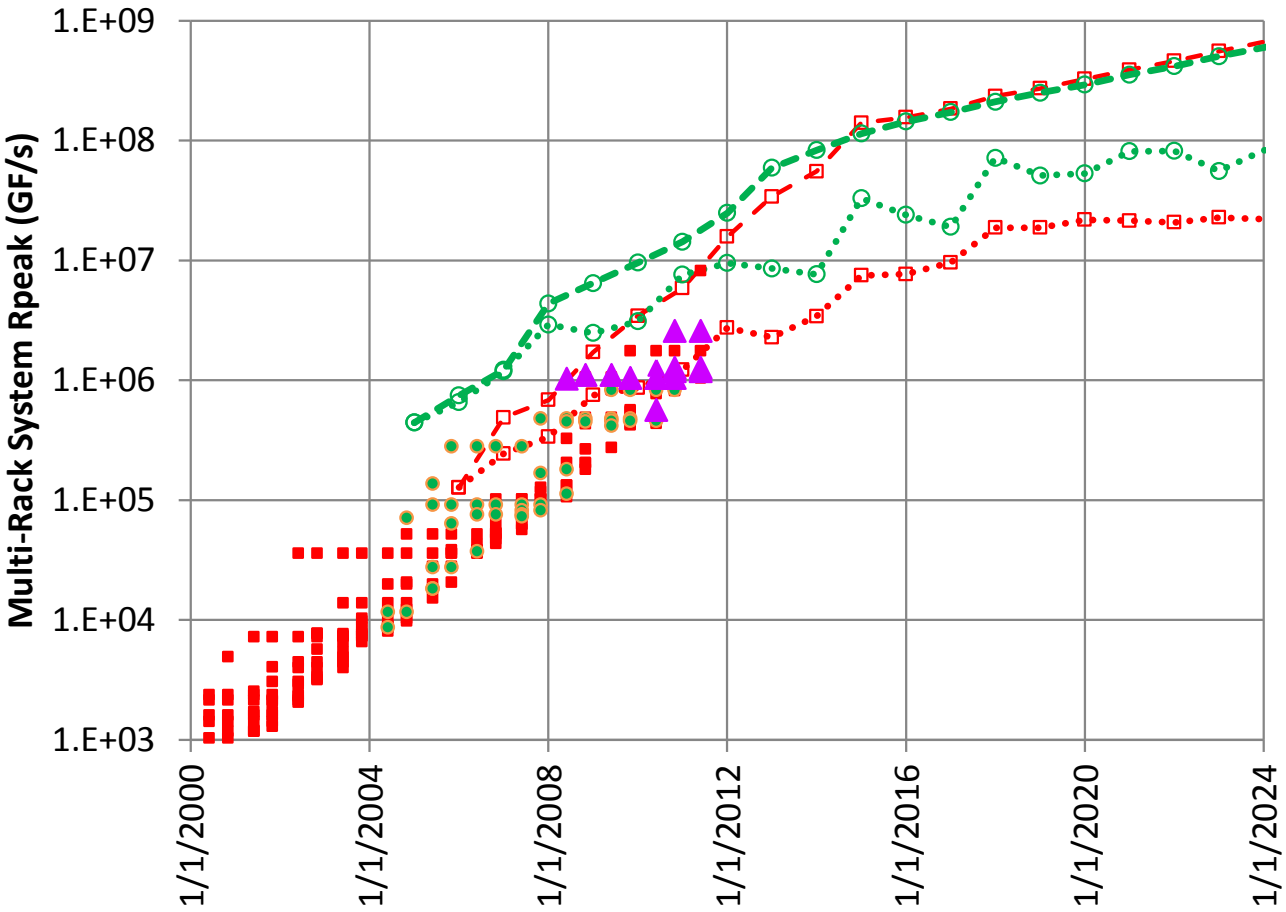
Courtesy of Peter Kogge, UND

Performance per 57KW rack



Courtesy of Peter Kogge, UND

Performance Projections



Dally's 2-3-4 Rule for Power Improvement

- Stated at ISC-13 Keynote address
- Needs a factor of 25X energy efficiency improvement
- 2.2X through device technology fabrication process
- 3X through logic circuit design
- 4X through architecture

Ethernet

- Gigabit Ethernet is the prevalent class of interconnects during the last decade
  - Gained the lead in TOP500 system count in June 2005, dethroning the Myrinet
  - Reaches the peak of 56.6% of all TOP500 systems in June 2008
  - The dominance continues until June 2012, when it slips to #2 behind InfiniBand
- Standards
  - Fast Ethernet (100Mbps, primarily over 4-pair Cat3 cable) is in decline or used in auxiliary networks
  - Gigabit Ethernet (1000Mbps; 1000BASE-T over Cat5 UTP cables or 1000BASE-X over optical fiber) becomes the prevalent implementation
  - 10Gigabit Ethernet (10Gbps; 10GBASE-CX over twin-ax cable, 10GBASE-T over Cat5e or better twisted pair, multiple standards for fiber optics), gains popularity in new installations
  - 40 and 100Gbps speeds defined by IEEE 802.3ba-2010

Intel Pro 1000 GigE NIC with PCI-X interface

Cat5 copper cable

Cisco Catalyst stackable GigE switches

10GbE NIC with dual SFP+ ports and PCIe interface

Dell Powerconnect 10GbE switch

10G cable with SFP+ connectors

InfiniBand

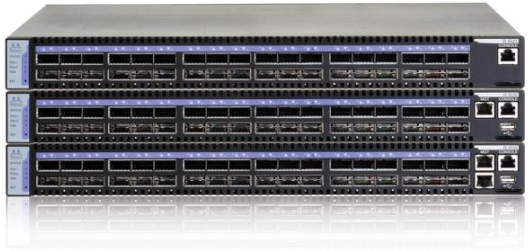
- Scalable switched fabric communications technology
  - High throughput and low latency
  - Point-to-point bidirectional serial links
  - Quality of service and failover features
  - Superset of VIA zero-copy model
- Performance
  - 2.5Gbps signaling rate in each direction per link in SDR speed
  - 8b/10b encoding results in effective data rate of 2Gbps per link in SDR mode
  - FDR and EDR variants use more efficient 64b/66b symbol encoding
  - Connections typically utilize 1, 4, or 12 bonded links, with 4x QDR installations being the most common
  - 100ns typical QDR switch latency
  - Slightly above 1µs end-to-end MPI latency with typical QDR HCAs
- Since June '12 InfiniBand is the dominant class of interconnects in TOP500
  - SDR installations appeared in 2003, DDR in 2006, QDR in 2009, FDR in 2011; EDR instances are still uncommon
- Primary HCA and switch manufacturers: Mellanox and Intel (after last year's acquisition of QLogic)

	SDR	DDR	QDR	FDR-10	FDR	EDR
1X	2 Gbit/s	4 Gbit/s	8 Gbit/s	10.3125 Gbit/s	13.64 Gbit/s	25 Gbit/s
4X	8 Gbit/s	16 Gbit/s	32 Gbit/s	41.25 Gbit/s	54.54 Gbit/s	100 Gbit/s
12X	24 Gbit/s	48 Gbit/s	96 Gbit/s	123.75 Gbit/s	163.64 Gbit/s	300 Gbit/s

Theoretical data rate of InfiniBand implementations

QLogic 4x QDR Host Channel Adapter

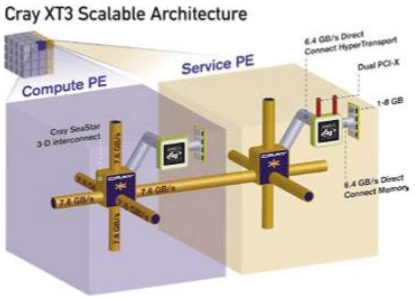
40Gbps cable with QSFP connectors



Mellanox 40Gbps/port switches with 2.88Tbps non-blocking bandwidth

Cray Interconnects

- Seastar (Red Storm, Cray XT3), 2004
  - 3-D mesh topology with link throughput of 2.5GB/s in each direction
  - Contains router, independent send and receive DMAs connected to HyperTransport bus (no PCI bottlenecks), and PowerPC440 processor with 384KB of scratch memory
- SeaStar2 (Cray XT4), 2006
  - Peak bidirectional bandwidth per link: 7.6GB/s,
  - 6.4GB/s bandwidth to node processors over HT bus
- SeaStar2+ (Cray XT5), 2009
  - 9.6GB/s peak bidirectional bandwidth per link
- Gemini (Cray XE6, XK6, XK7), 2010
  - 4 links per X and Z direction, 2 links per Y (10 total per NIC)
  - Peak bandwidth per direction: 8.3GB/s; inter-node latency of 1.5s on quiet network
  - Link level reliability and adaptive routing
- Aries/Dragonfly (Cray XC30), 2012
  - High radix tiled router (48 tiles), 8 processor tiles, 4 NICs
  - Each tile provides one bidirectional link 3 lanes wide, 12.5Gbps optical, 14Gbps electrical bandwidth
  - 120 million gets/puts per second per node
  - ASIC: 40nm technology, 184 lanes of high-speed SerDes
  - But: PCIe increases per packet overhead



SeaStar interconnect

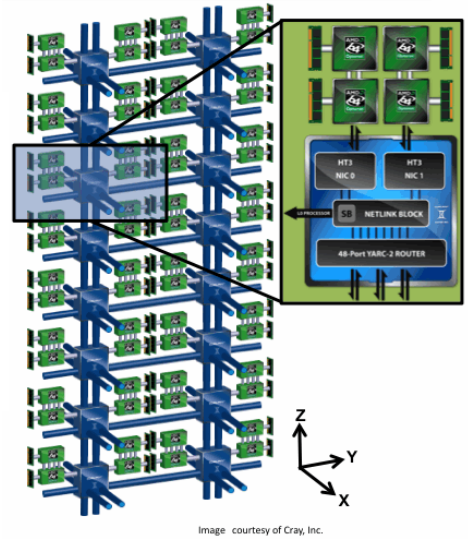
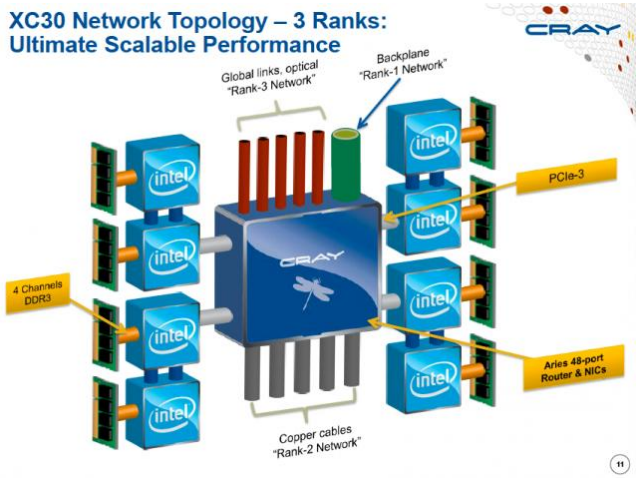


Image courtesy of Cray, Inc.

Gemini interconnect and ASIC



Node with Aries router; uses electrical links for short connections and optical for longer

Performance Factors – SLOWER

$$P = e * S * a * U$$

P – performance
e – efficiency
S – application's parallelism,
a – availability, reliability, maintenance
U - normalization factor/compute unit

- Starvation
  - Insufficiency of concurrency of work
  - Impacts scalability and latency hiding
  - Effects programmability
- Latency
  - Time measured distance for remote access and services
  - Impacts efficiency
- Overhead
  - Critical time additional work to manage tasks & resources
  - Impacts efficiency and granularity for scalability
- Waiting for contention resolution
  - Delays due to simultaneous access requests to shared physical or logical resources

Sources of Asynchrony for Exascale

- Scale
- Increase range of network latencies and opportunities for packet collisions and routing variations
- Deeper memory hierarchy
- Scheduling conflicts for threads to cores
- Active response to errors
- Variable instruction rate from clock and voltage change
- Finer grain threads as normalization factor of time delays

Exascale Programming Issues

- Is there support for specific science domains?
- Does the language support programming both within and between compute nodes?
- What types of parallelism (implicit, SPMD, data, task) are supported?
- What types of synchronization exist in the languages?
- How is communication between tasks handled? Is it limited by task structure, type constraints, etc.?
- What other novel features exist for managing energy, resilience, reproducibility, or other systems features?

Requirements for Exascale Programming Models (1)

- Expose and exploit sufficient parallelism for scalability and efficiency
  - > billion-way concurrency
  - Diversity of granularity of size
  - Parallelism discovery from meta-data
  - Overlapping of phases of computation
- High efficiency of execution
  - Low overheads for mechanisms to manage tasks and resources
    - Powerful synchronization primitives
  - Overlapping of communication and computation for latency hiding
  - Dynamic adaptive resource management and task scheduling
  - Global address space

Requirements for Exascale Programming Models (2)

- Portability
  - Across range of scales
  - Between systems of different types and configurations
  - Spanning multiple generations of technologies and architectures
  - Suggests elements of declarative style
- Enhancing reliability
  - Detection of errors
    - Lightweight test calculations to verify correctness
  - Isolation of errors between global side-effects
    - Encapsulate calculations
  - Recovery
    - Establish action in presence of errors
    - Retain temporary values until following computation is validated

Requirements for Exascale Programming Models (3)

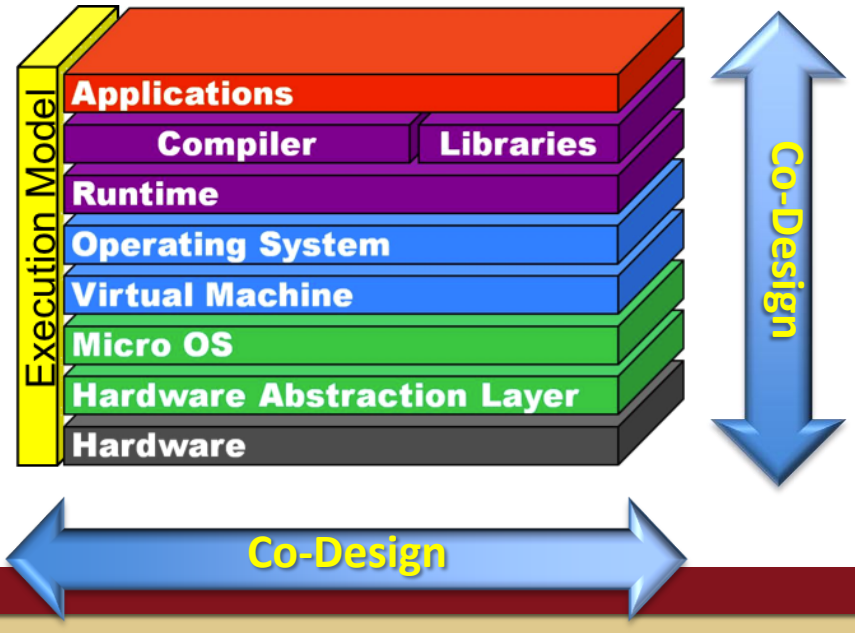
- Runtime system interface
  - Delineate application parallelism
  - Discover meta-data parallelism through data-directed execution
  - Describe locality relations among data and control components
- Energy/Power
  - Interface with introspection subsystem
  - Predict critical path of threads
- Interoperability
  - With legacy codes
  - Libraries

Requirements for System Software

- Runtime System
  - Global address space
    - Beyond PGAS (AGAS)
    - Efficient system wide access
  - Lightweight multi-threading
    - First class objects
    - Medium granularity parallelism
  - Active message transport
    - Move work to data
  - Asynchrony response
    - Efficient synchronization
    - Dataflow, Futures
- Operating System
  - Lightweight kernel
  - Runtime system support
  - Legacy support
  - Job management
  - Thread management
  - System organization
  - Introspection
  - Memory management
  - Network interface

Exascale Co-Design

- Application-driven co-design is the process by which:
  - Scientific problem requirements guide computer architecture and system software design
  - Technology capabilities and constraints inform formulation and design of algorithms and software
- **Shared global perspective across the design-space establishes shared conceptual framework for co-design and interoperability**
  - Parallelism
  - Latency
  - Overhead
  - Dependability



DOE Co-design Centers

Exascale Co-Design Center for Materials in Extreme Environments (ExMatEx)

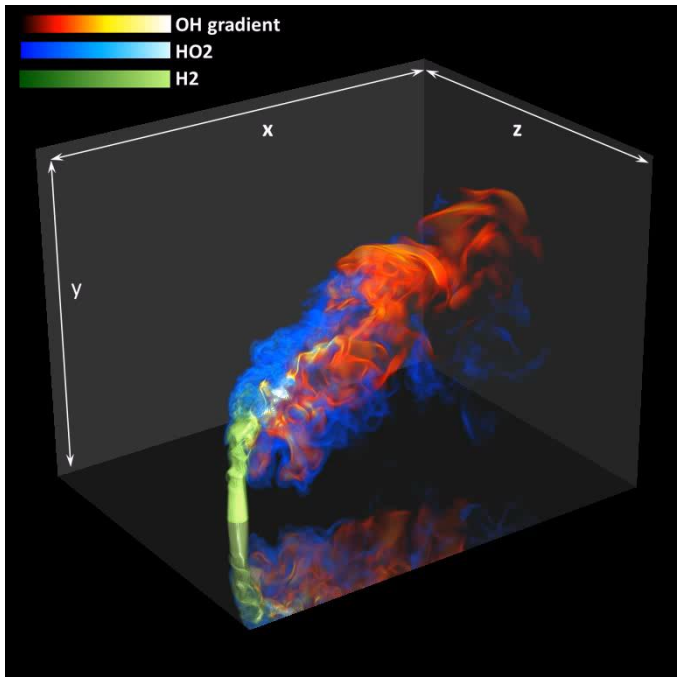
Director: Timothy Germann (LANL)

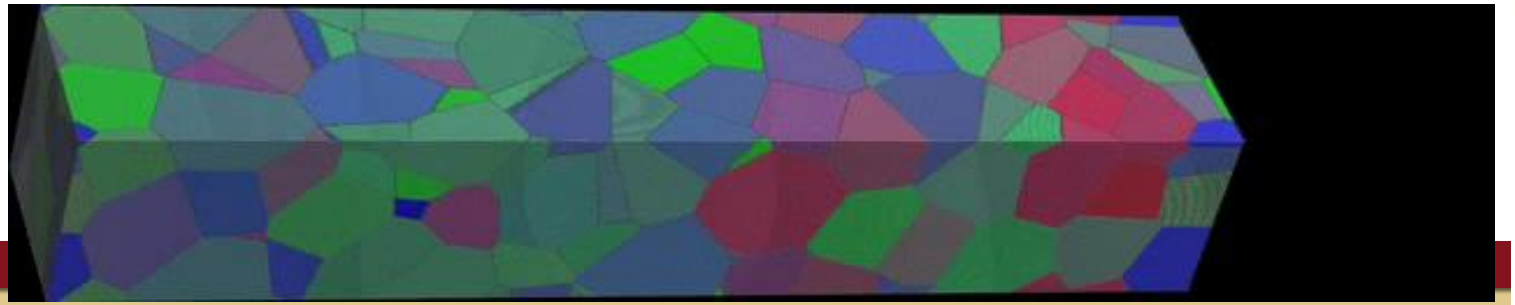
Center for Exascale Simulation of Advanced Reactors (CESAR)

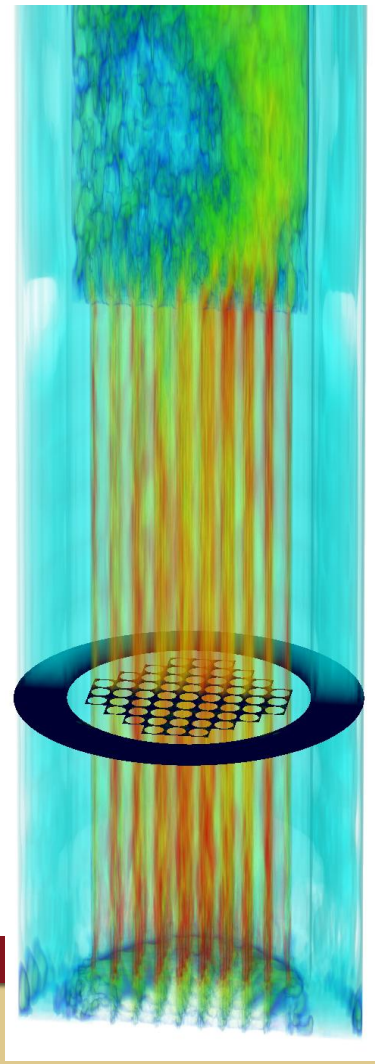
Director: Robert Rosner (ANL)

Combustion Exascale Co-Design Center (CECDC)

Director: Jacqueline Chen (SNL)







FastForward Program

- DOE Office of Science and NNSA
- Kickstart R&D exascale foundational technologies
  - Processors, memory, storage
- Awards:
  - Intel ($19M)
  - AMD ($12.6M)
  - NVIDIA ($12M)
  - Whamcloud (?)

X-Stack Focus

- **Programming Models**: new approaches to managing parallelism and data movement through innovations in interfaces
- **Runtime Systems**: dynamically adapt to changing application goals and system conditions
- **Locality-aware and energy efficient mechanisms:** manage locality and optimize for energy
- **Interoperability:** interoperability with different HPC languages and interfaces as well as cross-cutting execution models

X-Stack Portfolio

- **DEGAS** (Kathy Yelick)

  Hierarchical and resilient programming models, compilers and runtime support.

- **Traleika** (Shekhar Borkar)

  Exascale programming system, execution model and runtime, applications, and architecture explorations, with open and shared simulation infrastructure.

- **D-TEC** (Dan Quinlan)

  Complete software stack solution, from DSLs to compilers to optimized runtime systems.

- **XPRESS** (Ron Brightwell)

  Software architecture and interfaces that exploit the ParalleX execution model, prototyping several of its key components.

- **DynAX** (Rishi Khan)

  ovel programming models, dynamic adaptive xecution models and runtime systems.

- **X-Tune** (Mary Hall)

  Unified autotuning framework that integrates programmer-directed and compiler-directed autotuning.

- **GVR** (Andrew Chien)

  Global view data model for architecture support for resilience.

CORVETTE (Koushik Sen)

Automated bug finding methods to eliminate non-determinism in program execution and to make concurrency bugs and floating point behavior reproducible.

SLEEC (Milind Kulkarni)

Semantics-aware, extensible optimizing compiler that treats compilation as an optimization problem.

X-Stack: Vision in Progress

Energy Efficiency, Resilience, Programmability, Scalability, Performance Portability, Interoperability

HPC Programmer

Parallel Language

Existing MPI + X code

Migration Process

Existing MPI + X code

Domain Expert

Program in DSL Xi

DSL X1…Xn compilers

DSL Designers

DSLs Generator

Feedback to Apps

High Level IR

Performance Programmer

Compiler Transformations

Lowering

Manual Refinement

Low Level IR

Optimization Management

Runtimes

Fault Detection and Mitigation

X-Stack Front end

Compiler-optimized code

X-Stack Back end

Runtime optimized code

Architecture Feedback

CENTER FOR RESEARCH IN EXTREME SCALE TECHNOLOGIES

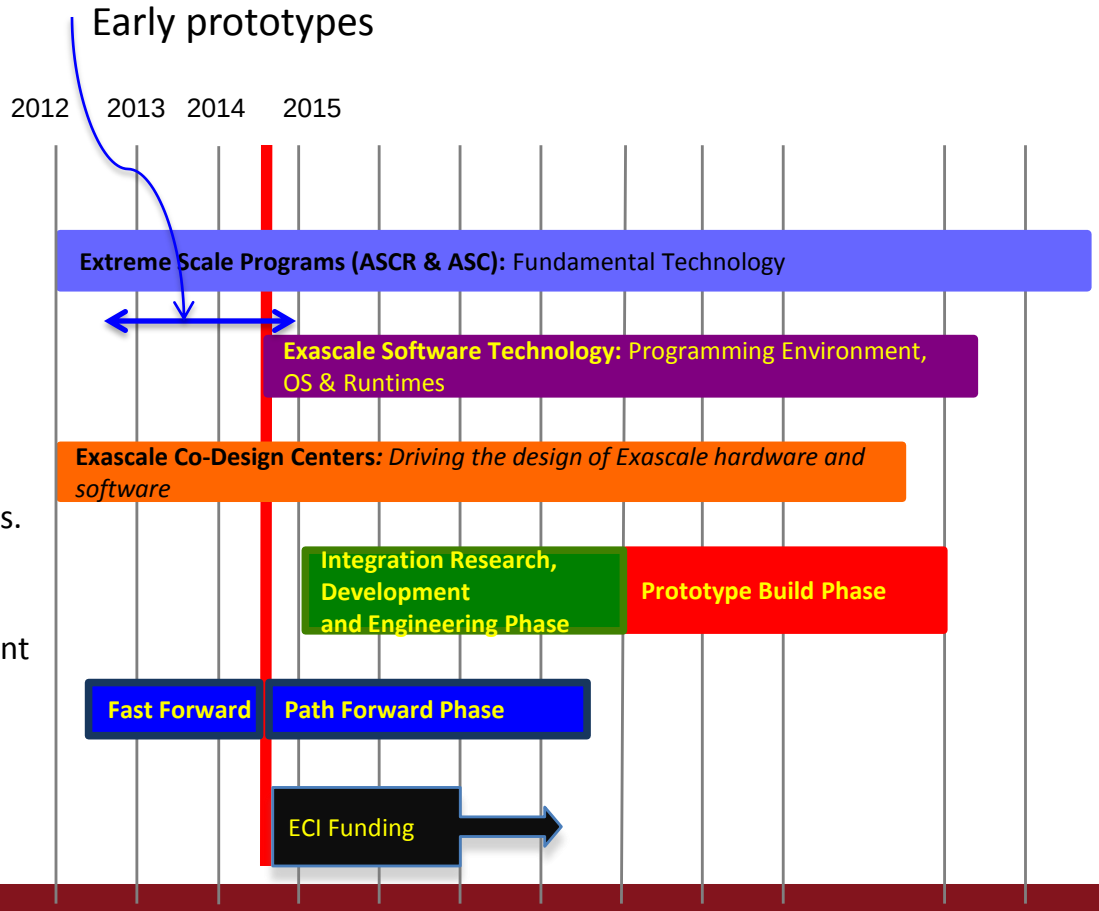
INDIANA UNIVERSITY

Pervasive Technology Institute

X-Stack Software Vision

Aligned with the Exascale Research Initiative

- ECI Goals: Deploy exascale computers:
  - 500 to 1,000 more **performance** than today's HPC systems
  - Under **20MW** Power
  - Highly **programmable**
- ECI Strategy:
  - Conduct critical R&D efforts.
  - Develop exascale software stacks.
  - Fund computer technology vendors
  - Fund the design and development of exascale computer systems.
  - **Joint effort with NNSA.**
  - Collaboration with other government agencies and other countries.



Exascale Programming Interfaces

- Building Solver-Aided DSLs with Rosette (Emina Torlak, UCB)
  - Rosette is a new framework for rapid design and prototyping of solver-aided DSLs.
- XPI: the eXascale ParalleX Intermediate form (Andrew Lumsdaine, Indiana University)
  - The XPRESS team will discuss the eXascale ParalleX Intermediate form (XPI), a low-level API to HPX functionality that can be directly called as a library or used as a compiler target.
- Sketch: a system for program synthesis (Armando Solar-Lezama, MIT)
  - The Sketch system uses
- Swift: a parallel scripting language (Michael Wilde, Argonne)
  - The Swift scripting language, a portable parallel language for composing parallel applications.

		X10	Swift (/K and /T)*	UPC	Habanero-C (HC)	CAF 2.0	SLEEC	SWARM /SCALE	Sketch	XPI
1	DSL?	No	No	No	No	No	Yes	No	Yes	No
2	Inter/Intra-Node?	Same	Inter	Both	Both	Same	Neither	Both	Intra	Same
3	Types of parallelism	Task, Map-Reduce	Task	SPMD	Task -> SPMD	SPMD	N/A	Task	Implicit	Tasks
4	Synch	Conditional atomics	Implicit dataflow	Barriers and locks	Finish, Asynch, Phasers; --> Futures, Nested atomics, Actors	Events, locks, cofence, barriers, finish. Adding atomics.	N/A	Depen-dences, codelet chaining, latches, barriers, locks, etc.		Futures Dataflow
5	Comm	Shared memory		PGAS & collect-ives	Shared mem + {MPI, GASNet}	PGAS & collect-ives.	N/A	Futures, remote codelets		Active GAS
6	Energy, resilience,...	Resilience (in progress)	Automatic retry	Resilience (in progress)	Resilience (in progress)	Resilience (in progress)	Heterogeneity	Resilience (in progress)		Resilience (in progress)

XPI: Extreme-scale ParalleX Interface

- A low level C-language library interface to a ParalleX runtime system (e.g. HPX)
- Intermediate Form for source-to-source translators from high level languages
- Readable syntax for low level programming
- A set of tools to assist in:
  - Code analysis
  - Generation of boilerplate

Runtime Systems

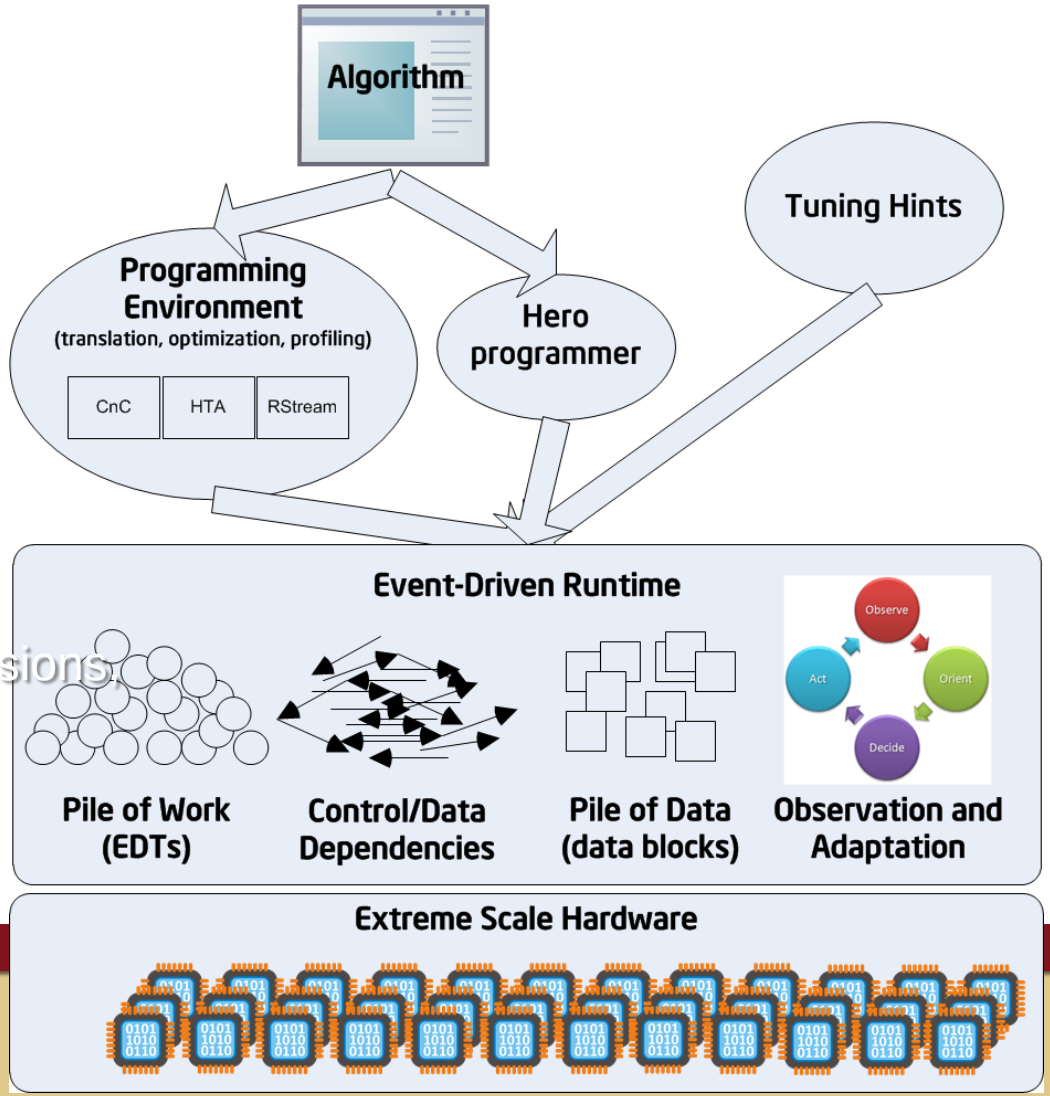
- Exascale systems will impose a fresh set of requirements on runtime systems including
  - targeting nodes with hundreds of homogeneous and heterogeneous cores
  - energy, data movement and resiliency constraints within and across nodes.
- focus on the fundamental research challenges that need to be addressed in the area of runtimes for exascale systems.

- Intra-node MPI
- Open Community Runtime
- GASNet
- HPX
- SWARM
- TASCEL
- X10 Runtime

Open Community Runtime

- Fine-grained, asynchronous event-driven runtime framework with movable data and computation

- Hosted on 01.org
- Introduced at SC 2012
- Goals
  - Modularity
  - Stable APIs
  - Flexible implementation
  - Transparency
- Development process
  - Continuous integration
  - Quarterly milestones
  - Mailing lists for technical discussions, build status, etc
- Organization
  - Steering Committee
  - Core Team



GASNet

GASNet: 1999 to present

- "**G**lobal **A**ddress **S**pace **Net**working"
- API for implementing PGAS languages/libraries (UPC, CAF, Chapel, OpenSHMEM, Ti, and others)
- for compilers and low-level code authors
- widely portable
- MPI-interoperable on most platforms
- performs comparably to (and often better than) MPI send-recv
- has influenced MPI-3 design for one-sided operations (a.k.a. RMA)
- Key API Features include…
  - a rich set of one-sided Put/Get interfaces mapping well to modern RDMA-capable network h/w
  - Active Messages (a.k.a. "Function Shipping" or "Remote Procedure Call") providing powerful mechanism for implementing language-specific features

GASNet-EX: present and future

- Part of the DEGAS project
- A re-design & re-implementation for an **EX**ascale PGAS environment:
  - Numerous complex nodes
  - Constrained by memory and power
  - Advanced asynchronous clients and multi-client (e.g. UPC+CAF)
  - Resilient implementation with support for resilient clients
- Will support current and future DoE supercomputers
  - Dropping legacy support to improve maintainability
- Apply the lessons learned from GASNet work, including feedback from current and potential clients (Rice, UofH, Cray, IBM …)

HPX

STE||AR
stellar.cct.lsu.edu

- Active global address space (AGAS) over PGAS
- Message driven computation over message passing
- Lightweight control objects over global barriers
- Latency hiding over latency avoidance
- Adaptive locality control over static data distribution
- Moving work to data over moving data to work
- Fine grained parallelism of lightweight threads instead of Communicating Sequential Processes

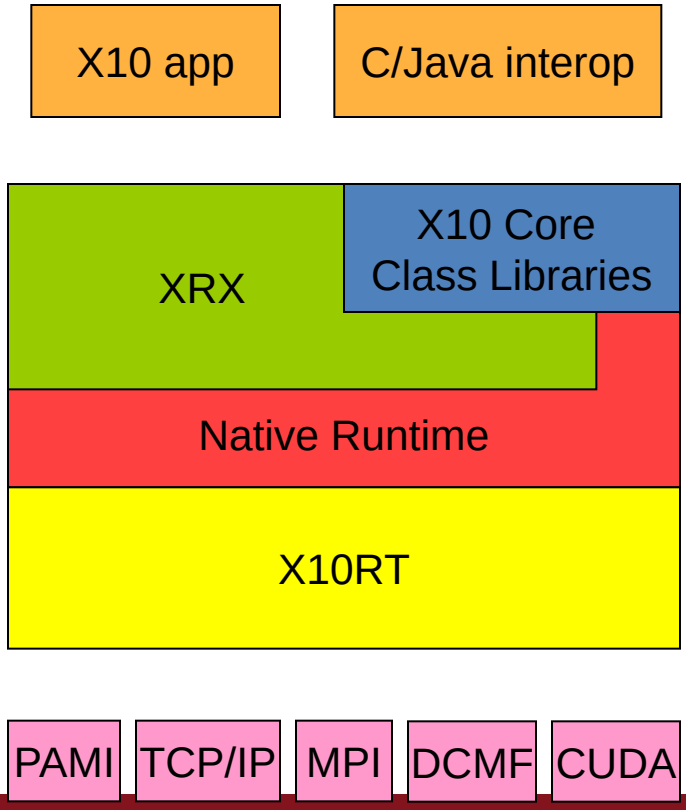
X10 Runtime

- Open source
- Scales [HPCC'12]
- Implements APGAS
  - PGAS + async + finish + when

- X10 runtime transport
- X10 runtime in X10
  - compiles to C++ and Java

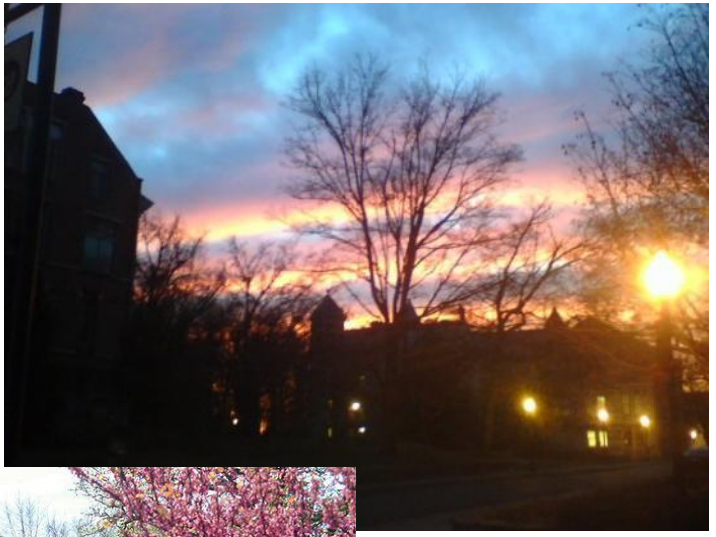
Roadmap to Exascale

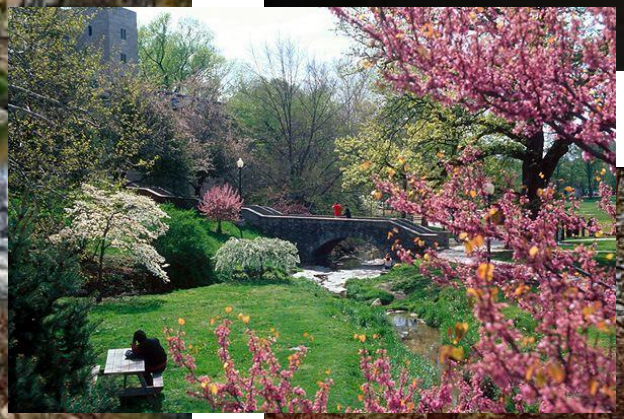
- interop. (C, MPI, ROSE) & DSLs
- >> parallelism (many-cores & accel.)
- elasticity & resilience



Conclusions

- Exascale is a major challenge in sustained performance and power
- 2 tracks to be pursued
  - Near term incremental changes to Exaflops by 2020
    - Brute force technology extensions with minimum ISA changes
    - An hoc hybrid programming: MPI + OpenMP + OpenACC + x86 assembler
    - A stunt machine measured by Linpack
    - Also supports some legacy codes but diminishing few at scale
  - Exascale system capable of valuable science computations by 2024
    - New architectures designed for system wide operation
    - New programming models for dynamic adaptive execution and portability
- We will not ever get to Zettaflops using silicon semiconductor technology and discrete numeric operations






CardCow.com