DOE CSGF

High Performance Computing Workshop
July 20, 2011, Arlington, VA

The Role of Programming Systems on the Road to Exascale Computing

Jeffrey Vetter
Oak Ridge National Lab and Georgia Tech
http://ft.ornl.gov

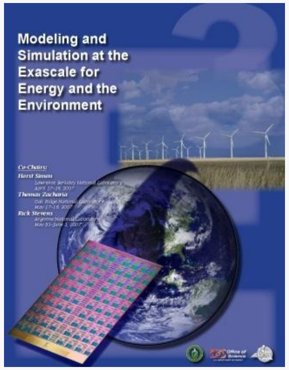
In a nutshell

- Exascale goals
  - Highlights from recent projections for exascale
- Challenges
  - Micro, macro power
  - Memory capacity and bandwidth
  - Parallelism
  - Programmability
- Programming systems play a crucial role
  - Survey of programming systems
  - Solutions are coming now
    - —Heterogeneity with GPUs
  - Programming models need a vigorous ecosystem
    - —Tools, autotuning, libraries

TOWARD EXASCALE

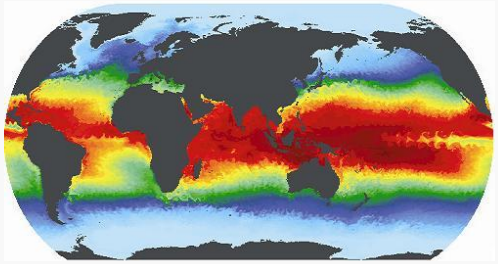
Process for identifying exascale applications and technology for DOE missions ensures broad community input

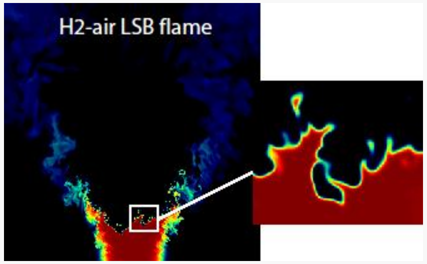
- Town Hall Meetings April-June 2007
- Scientific Grand Challenges Workshops Nov, 2008 – Oct, 2009
  - Climate Science (11/08),
  - High Energy Physics (12/08),
  - Nuclear Physics (1/09),
  - Fusion Energy (3/09),
  - Nuclear Energy (5/09),
  - Biology (8/09),
  - Material Science and Chemistry (8/09),
  - National Security (10/09)
  - Cross-cutting technologies (2/10)
- Exascale Steering Committee
  - "Denver" vendor NDA visits 8/2009
  - SC09 vendor feedback meetings
  - Extreme Architecture and Technology Workshop 12/2009
- International Exascale Software Project
  - Santa Fe, NM 4/2009; Paris, France 6/2009; Tsukuba, Japan 10/2009, etc.

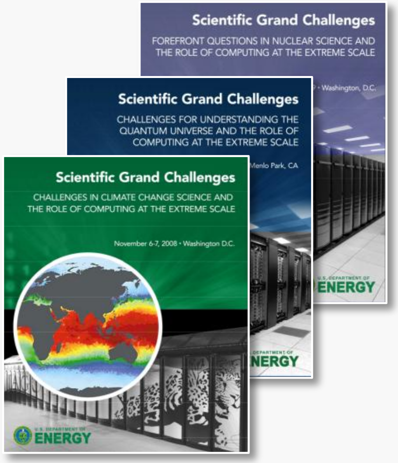


Modeling and Simulation at the Exascale for Energy and the Environment

Co-Chairs:
Horst Simon
Lawrence Berkeley National Laboratory
April 17-18, 2007
Thomas Zacharia
Oak Ridge National Laboratory
May 17-18, 2007
Rick Stevens
Argonne National Laboratory
May 31-June 1, 2007





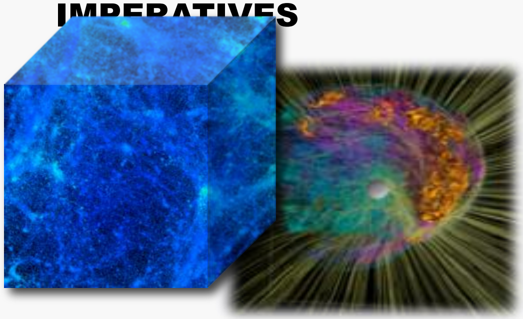


Scientific Grand Challenges
FOREFRONT QUESTIONS IN NUCLEAR SCIENCE AND THE ROLE OF COMPUTING AT THE EXTREME SCALE
· Washington, D.C.

Scientific Grand Challenges
CHALLENGES FOR UNDERSTANDING THE QUANTUM UNIVERSE AND THE ROLE OF COMPUTING AT THE EXTREME SCALE
Menlo Park, CA

Scientific Grand Challenges
CHALLENGES IN CLIMATE CHANGE SCIENCE AND THE ROLE OF COMPUTING AT THE EXTREME SCALE
November 6-7, 2008 • Washington D.C.

MISSION IMPERATIVES



FUNDAMENTAL SCIENCE

http://science.energy.gov/ascr/news-and-resources/workshops-and-conferences/grand-challenges/

http://www.exascale.org/iesp/Main_Page

Holistic View of HPC

Performance, Resilience, Power, Programmability

Applications

- Materials
- Climate
- Fusion
- National Security
- Combustion
- Nuclear Energy
- Cybersecurity
- Biology
- High Energy Physics
- Energy Storage
- Photovoltaics
- National Competitiveness

- Usage Scenarios
  - Ensembles
  - UQ
  - Visualization
  - Analytics

Programming Environment

- Domain specific
  - Libraries
  - Frameworks
  - Templates
  - Domain specific languages
  - Patterns
  - Autotuners

- Platform specific
  - Languages
  - Compilers
  - Interpreters/Scripting
  - Performance and Correctness Tools
  - Source code control

System Software

- Resource Allocation
- Scheduling
- Security
- Communication
- Synchronization
- Filesystems
- Instrumentation
- Virtualization

Architectures

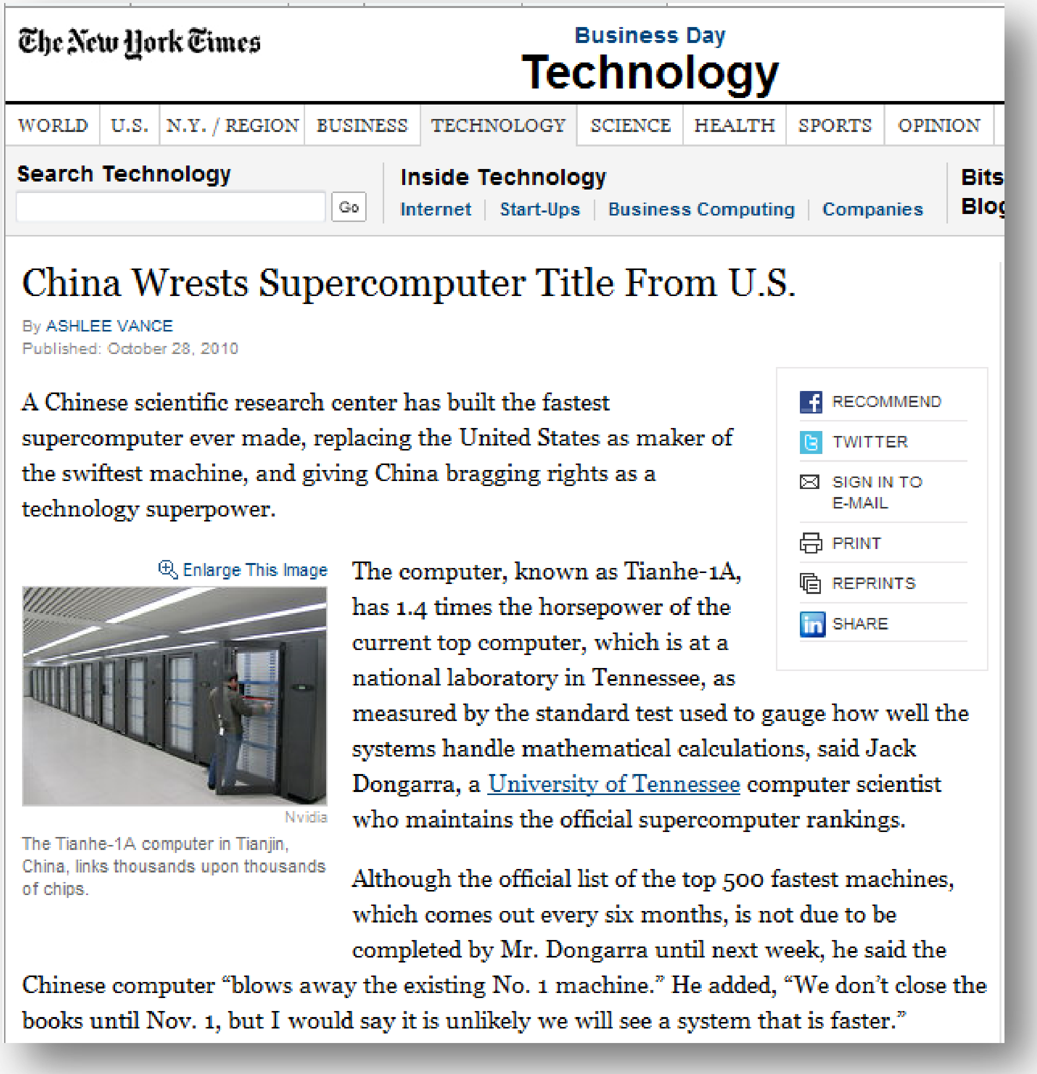
- Processors
  - Multicore
  - Graphics Processors
  - FPGA
  - DSP
- Memory and Storage
  - Shared (cc, scratchpad)
  - Distributed
  - RAM
  - Storage Class Memory
  - Disk
  - Archival
- Interconnects
  - Infiniband
  - IBM Torrent
  - Cray Gemini, Aires
  - BGL/P/Q
  - 1/10/100 GigE

Where are we now? Contemporary Systems

Date	System	Location	Comp	Comm	Parall elism	Peak (PF)	Power (MW)
2010	Tianhe-1A	NSC in Tianjin	Intel + NVIDIA	Proprietary		4.7	4.0
2010	Nebulae	NSC In Shenzhen	Intel + NVIDIA	IB		2.9	2.6
2010	Tsubame 2	TiTech	Intel + NVIDIA	IB		2.4	1.4
2011	K Computer (612 cabinets)	Kobe	SPARC64 VIIIfx	Tofu		8.7	9.8
~2012	Cray ‘Titan’	ORNL	AMD + NVIDIA	Gemini		20?	7?
~2012	BlueWaters	NCSA/UIUC	POWER7	IBM Hub		10?	10?
~2012	BlueGeneQ	ANL	SoC	IBM		10?	
~2012	BlueGeneQ	LLNL	SoC	IBM		20?	
	Others…						

Tianhe-1A uses 7000+ NVIDIA GPUs

- Tianhe-1A uses
  - 7,168 NVIDIA Tesla M2050 GPUs
  - 14,336 Intel Westmeres
- Performance
  - 4.7 PF peak
  - 2.5 PF sustained on HPL
- 4.04 MW
  - If Tesla GPU's were not used in the system, the whole machine could have needed 12 megawatts of energy to run with the same performance, which is equivalent to 5000 homes
- Custom fat-tree interconnect
  - 2x bandwidth of Infiniband QDR



The New York Times

Business Day

Technology

WORLD | U.S. | N.Y. / REGION | BUSINESS | TECHNOLOGY | SCIENCE | HEALTH | SPORTS | OPINION
Search Technology
Go
Inside Technology
Internet | Start-Ups | Business Computing | Companies
Bits Blog

China Wrests Supercomputer Title From U.S.

By ASHLEE VANCE

Published: October 28, 2010

RECOMMEND
TWITTER
SIGN IN TO E-MAIL
PRINT
REPRINTS
SHARE
Enlarge This Image

A Chinese scientific research center has built the fastest supercomputer ever made, replacing the United States as maker of the swiftest machine, and giving China bragging rights as a technology superpower.

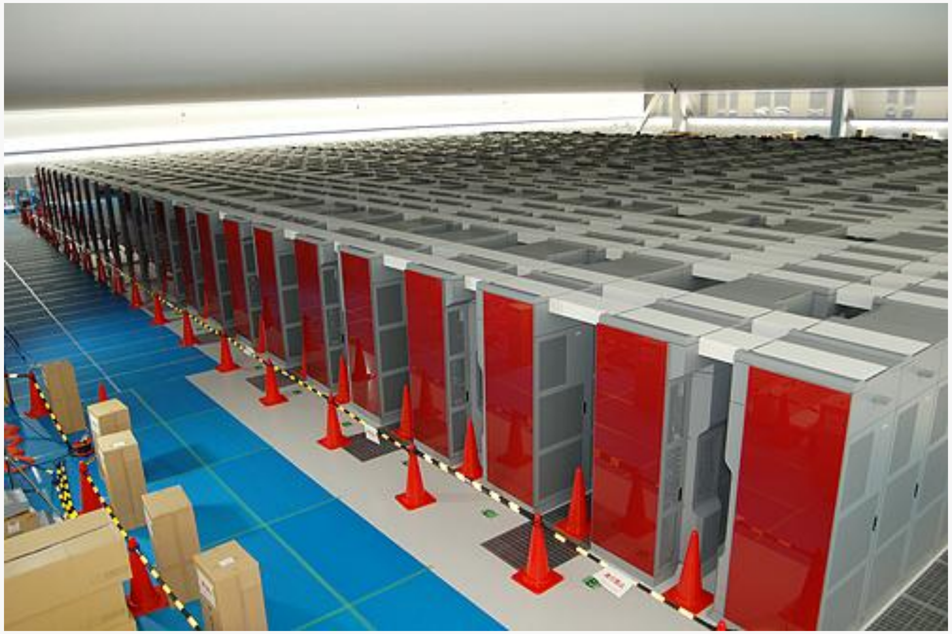
Nvidia

The Tianhe-1A computer in Tianjin, China, links thousands upon thousands of chips.

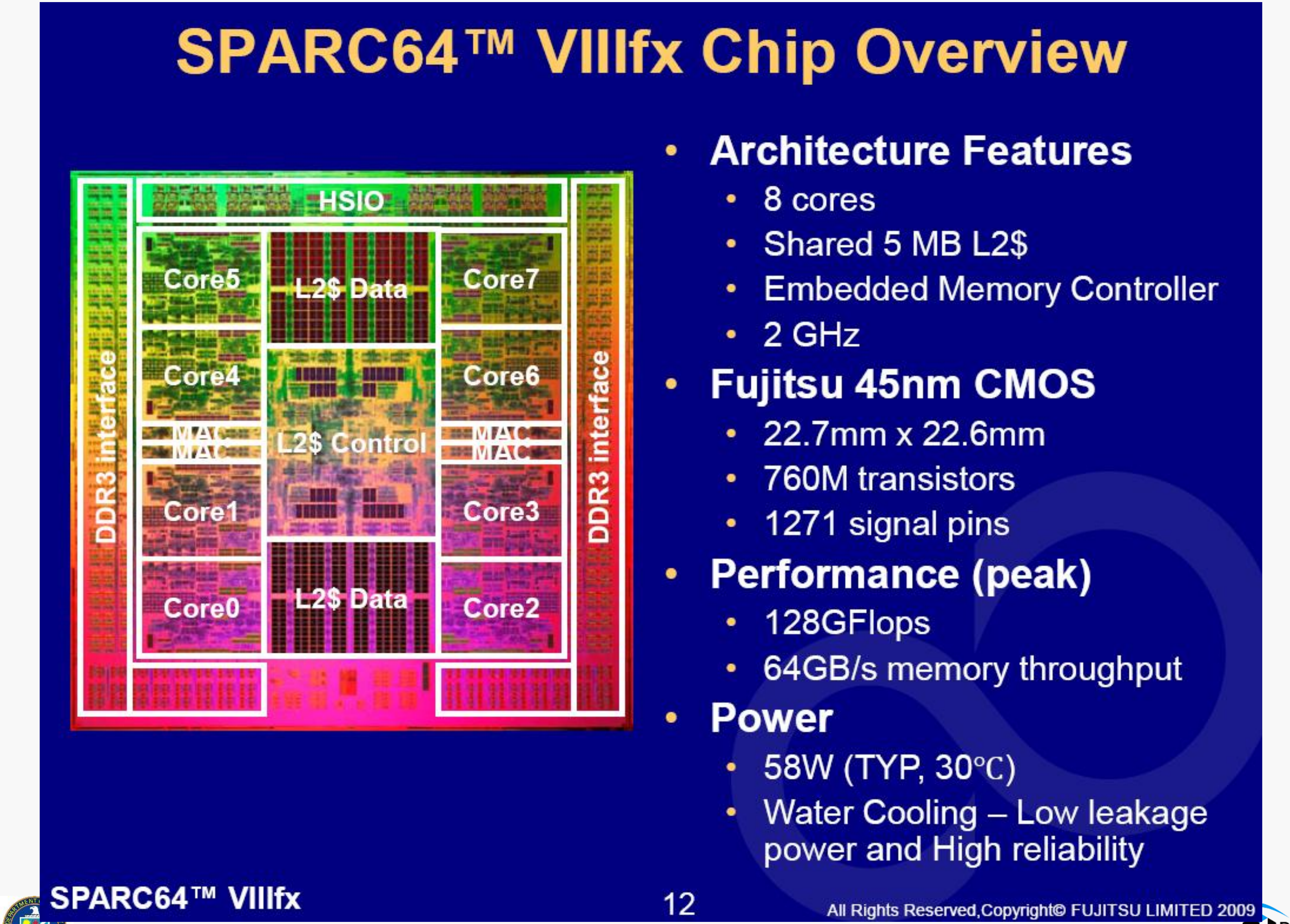
The computer, known as Tianhe-1A, has 1.4 times the horsepower of the current top computer, which is at a national laboratory in Tennessee, as measured by the standard test used to gauge how well the systems handle mathematical calculations, said Jack Dongarra, a University of Tennessee computer scientist who maintains the official supercomputer rankings.

Although the official list of the top 500 fastest machines, which comes out every six months, is not due to be completed by Mr. Dongarra until next week, he said the Chinese computer "blows away the existing No. 1 machine." He added, "We don't close the books until Nov. 1, but I would say it is unlikely we will see a system that is faster."

Recent news - K



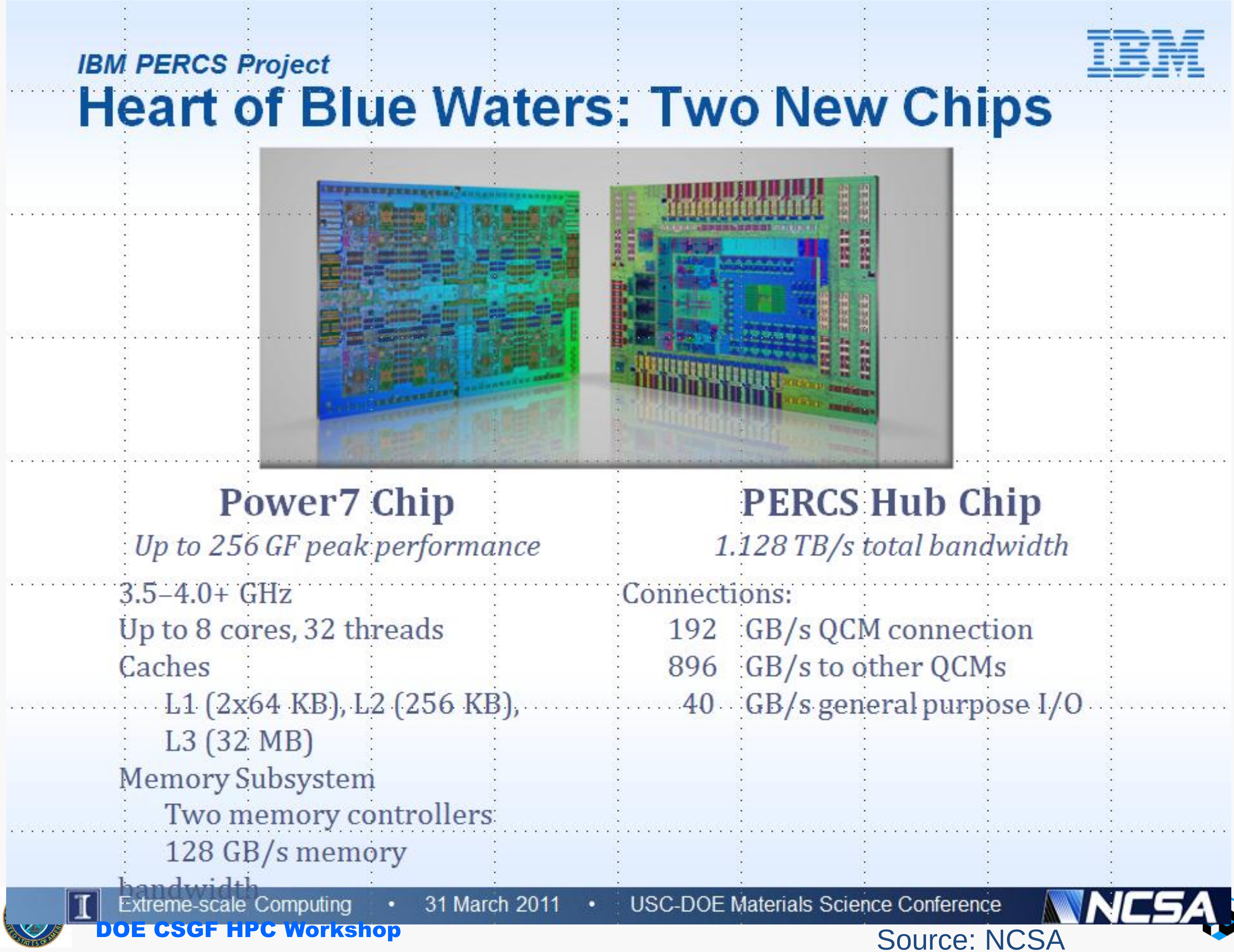
- #1 on TOP500
- 8.162 PF (93% of peak)
  - 3.1x TOP500 #2
  - 9.8 MW
- 672 racks (over 800 planned)
- 68,544 processors, 1PB memory



SPARC64™ VIIIfx Chip Overview

- Architecture Features
  - 8 cores
  - Shared 5 MB L2$
  - Embedded Memory Controller
  - 2 GHz
- Fujitsu 45nm CMOS
  - 22.7mm x 22.6mm
  - 760M transistors
  - 1271 signal pins
- Performance (peak)
  - 128GFlops
  - 64GB/s memory throughput
- Power
  - 58W (TYP, 30℃)
  - Water Cooling – Low leakage power and High reliability

Source: Fujitsu



IBM PERCS Project

Heart of Blue Waters: Two New Chips

Power7 Chip

Up to 256 GF peak performance

3.5–4.0+ GHz

Up to 8 cores, 32 threads

Caches

- L1 (2x64 KB), L2 (256 KB), L3 (32 MB)

Memory Subsystem

- Two memory controllers
- 128 GB/s memory bandwidth

PERCS Hub Chip

1.128 TB/s total bandwidth

Connections:

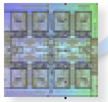
192	GB/s QCM connection
896	GB/s to other QCMs
40	GB/s general purpose I/O

Source: NCSA

Building Blue Waters

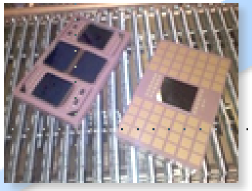
Blue Waters

Blue Waters will be the most powerful computer in the world for scientific research when it comes on line in 2011-2.



Power7 Chip

8 cores, 32 threads
L1, L2, L3 cache (32 MB)
Up to 256 GF (peak)
128 Gb/s memory bw
45 nm technology

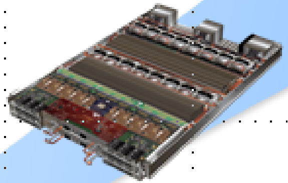


Quad-chip Module

4 Power7 chips
  Up to 1 TF (*peak*)
128 GB memory
  512 GB/s

Hub Chip

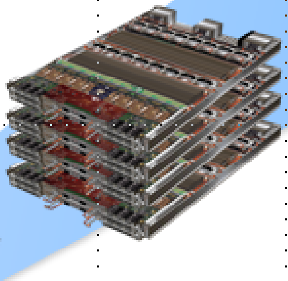
1.128 TB/s



IH Server Node

8 QCM's (256 cores)
  Up to 8 TF (*peak*)
1 TB memory
  4 TB/s
8 Hub chips
  9 TB/s
Power supplies
PCIe slots

Fully water cooled



IH Supernode

4 IH Server Nodes
  1024 cores
  Up to 32 TF (*peak*)
41 TB memory
  16 TB/s
32 Hub chips
  36 TB/s

**Blue Waters
3-Rack Building Block**

8 IH Supernodes
  256 TF (peak)
  32 TB memory
  128 TB/s memory
4 Storage systems (>500 TB)
10 Tape drive connections

Blue Waters

≳10 PF Peak
~1 PF sustained
≳300,000 cores
≳1 PB of memory
>25 PB of disk storage
500 PB of archival storage
≥100 Gbps connectivity

Blue Waters is built from components that can also be used to build systems with a wide range of capabilities—from deskside to beyond Blue Waters.

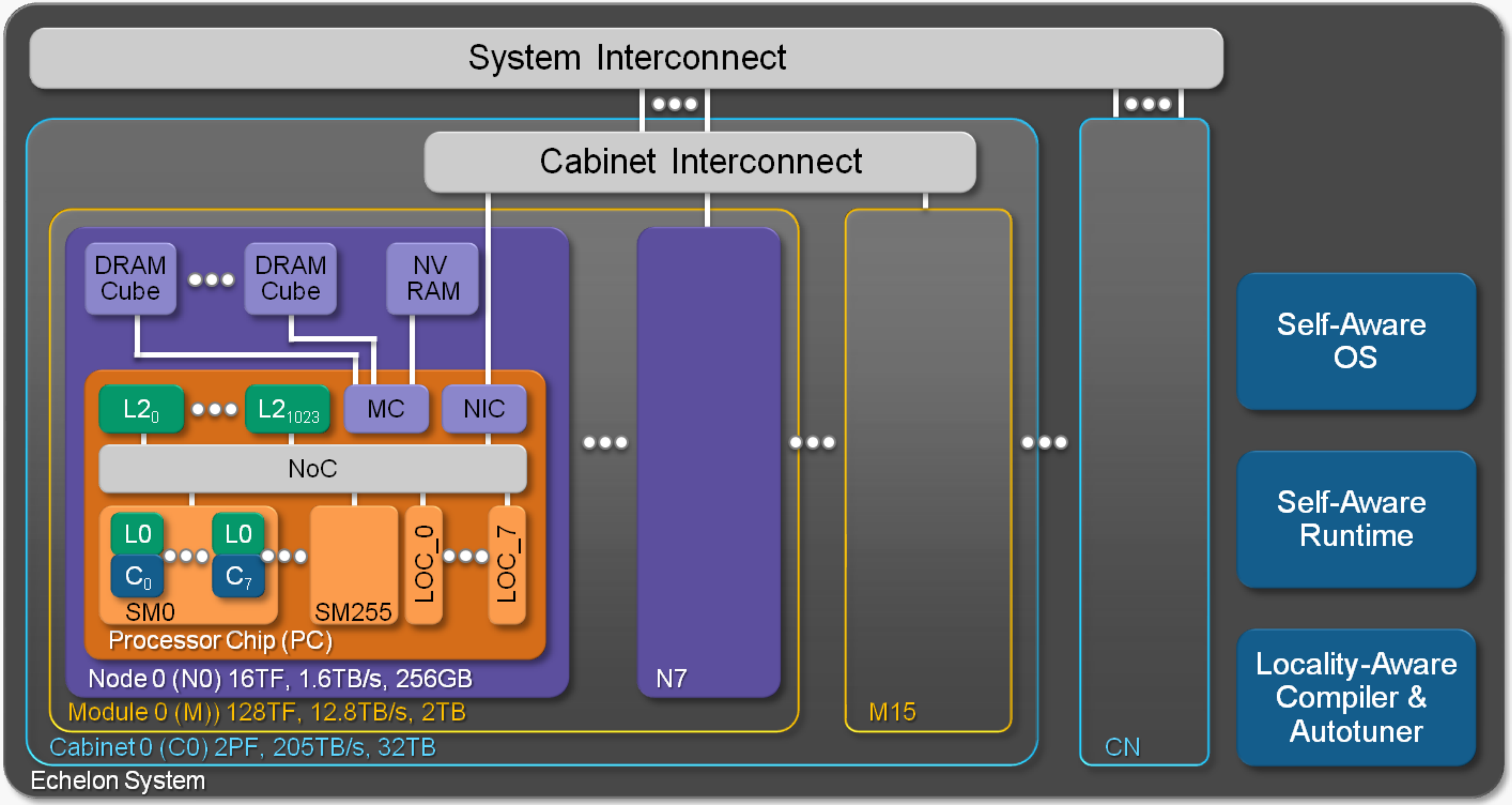
Source: NCSA

EXASCALE EXPECTATIONS AND CHALLENGES

Notional Exascale Architecture Targets

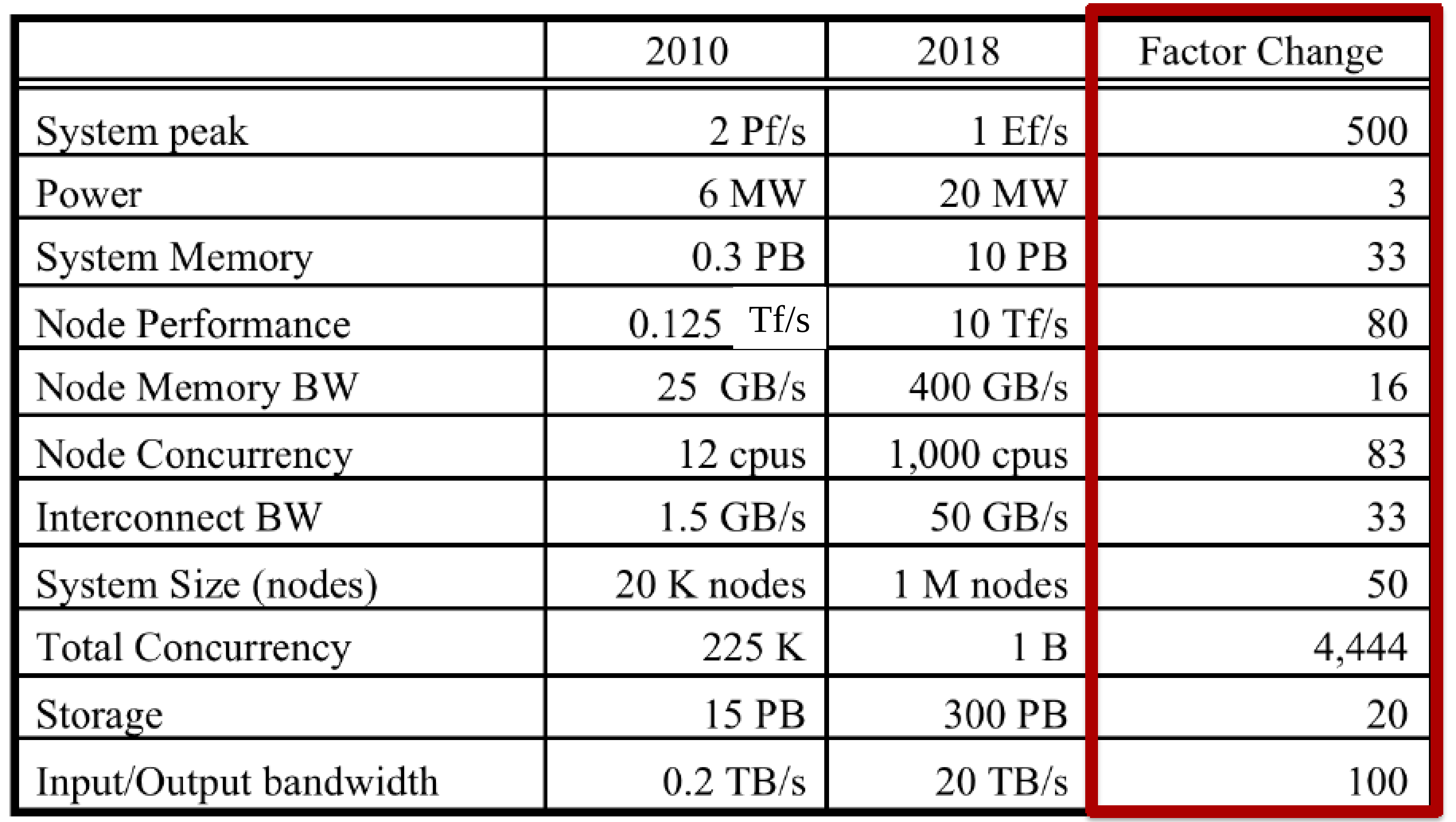
System attributes	2002	2010	“2015”		“2018”	
System peak	10 Tera	2 Peta	200 Petaflop/sec		1 Exaflop/sec	
Power	~0.8 MW	6 MW	15 MW		20 MW	
System memory	0.006 PB	0.3 PB	5 PB		32-64 PB	
Node performance	0.024 TF	0.125 TF	0.5 TF	7 TF	1 TF	10 TF
Node memory BW		25 GB/s	0.1 TB/sec	1 TB/sec	0.4 TB/sec	4 TB/sec
Node concurrency	16	12	O(100)	O(1,000)	O(1,000)	O(10,000)
System size (nodes)	416	18,700	50,000	5,000	1,000,000	100,000
Total Node Interconnect BW		1.5 GB/s	150 GB/sec	1 TB/sec	250 GB/sec	2 TB/sec
MTTI		day	O(1 day)		O(1 day)	

NVIDIA Echelon System Sketch



NVIDIA Echelon team: NVIDIA, ORNL, Micron, Cray, Georgia Tech, Stanford, UC-Berkeley, U Penn, Utah, Tennessee, Lockheed Martin

Note the Uneven Impact on System Balance!



	2010	2018	Factor Change
System peak	2 Pf/s	1 Ef/s	500
Power	6 MW	20 MW	3
System Memory	0.3 PB	10 PB	33
Node Performance	0.125 Tf/s	10 Tf/s	80
Node Memory BW	25 GB/s	400 GB/s	16
Node Concurrency	12 cpus	1,000 cpus	83
Interconnect BW	1.5 GB/s	50 GB/s	33
System Size (nodes)	20 K nodes	1 M nodes	50
Total Concurrency	225 K	1 B	4,444
Storage	15 PB	300 PB	20
Input/Output bandwidth	0.2 TB/s	20 TB/s	100

DOE Exascale Initiative Roadmap, Architecture and Technology Workshop, San Diego, December, 2009.

Challenges to ~~Exascale~~

Performance Growth

1) **System power** is the primary constraint
2) **Memory** bandwidth and capacity are not keeping pace
3) **Concurrency** (1000x today)
4) **Processor** architecture is an open question
5) **Programming model** heroic compilers will not hide this
6) **Algorithms** need to minimize data movement, not flops
7) **I/O bandwidth** unlikely to keep pace with machine speed
8) **Reliability and resiliency** will be critical at this scale
9) **Bisection bandwidth** limited by cost and energy

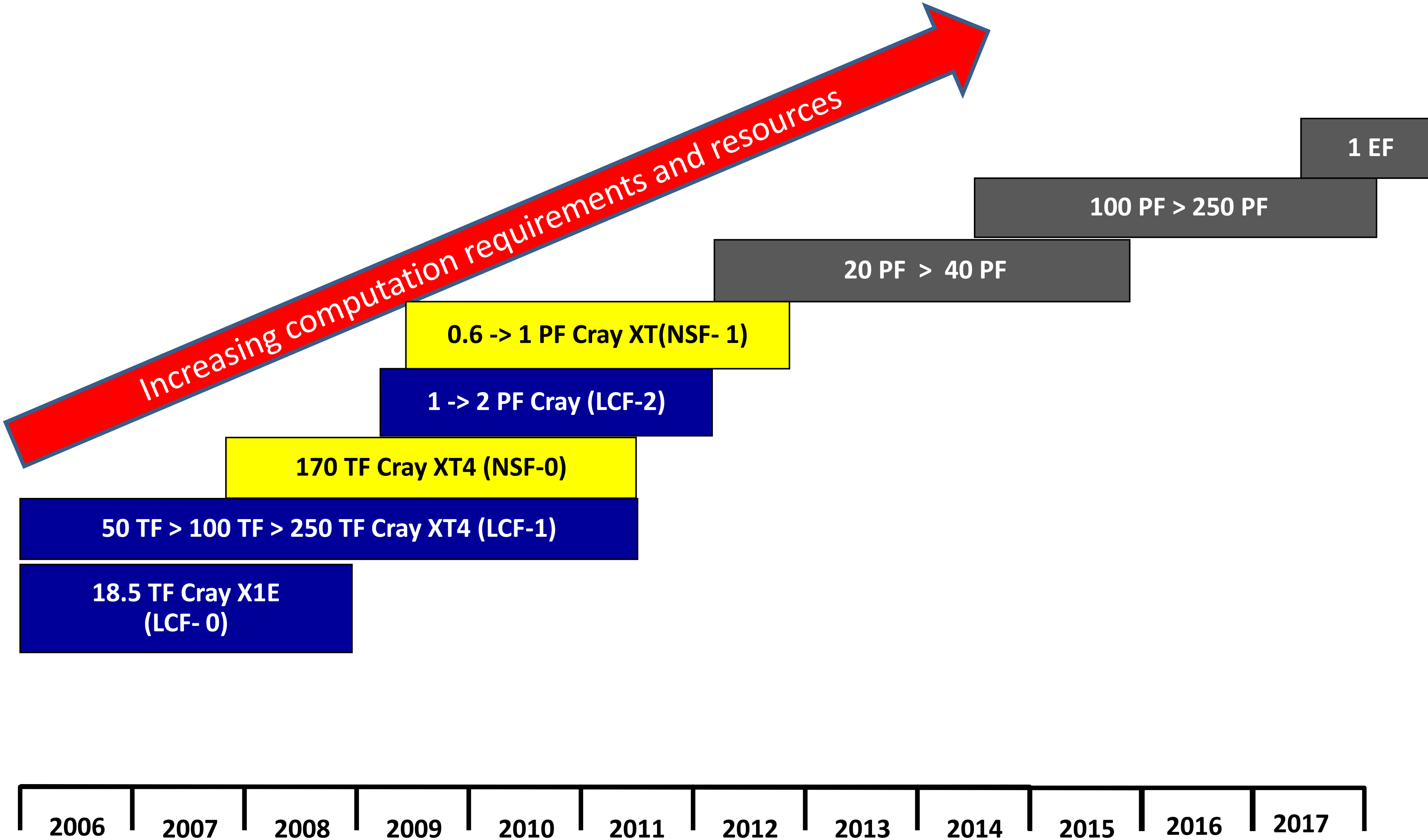
Unlike the last 20 years most of these (1-7) are equally important across scales, e.g., 100 10-PF machines

Source: Hitchcock, Exascale Research Kickoff Meeting

Both macro and micro energy trends drive all other factors

#1: POWER

ORNL Roadmap to Exascale



Increasing computation requirements and resources

1 EF

100 PF > 250 PF

20 PF > 40 PF

0.6 -> 1 PF Cray XT(NSF- 1)

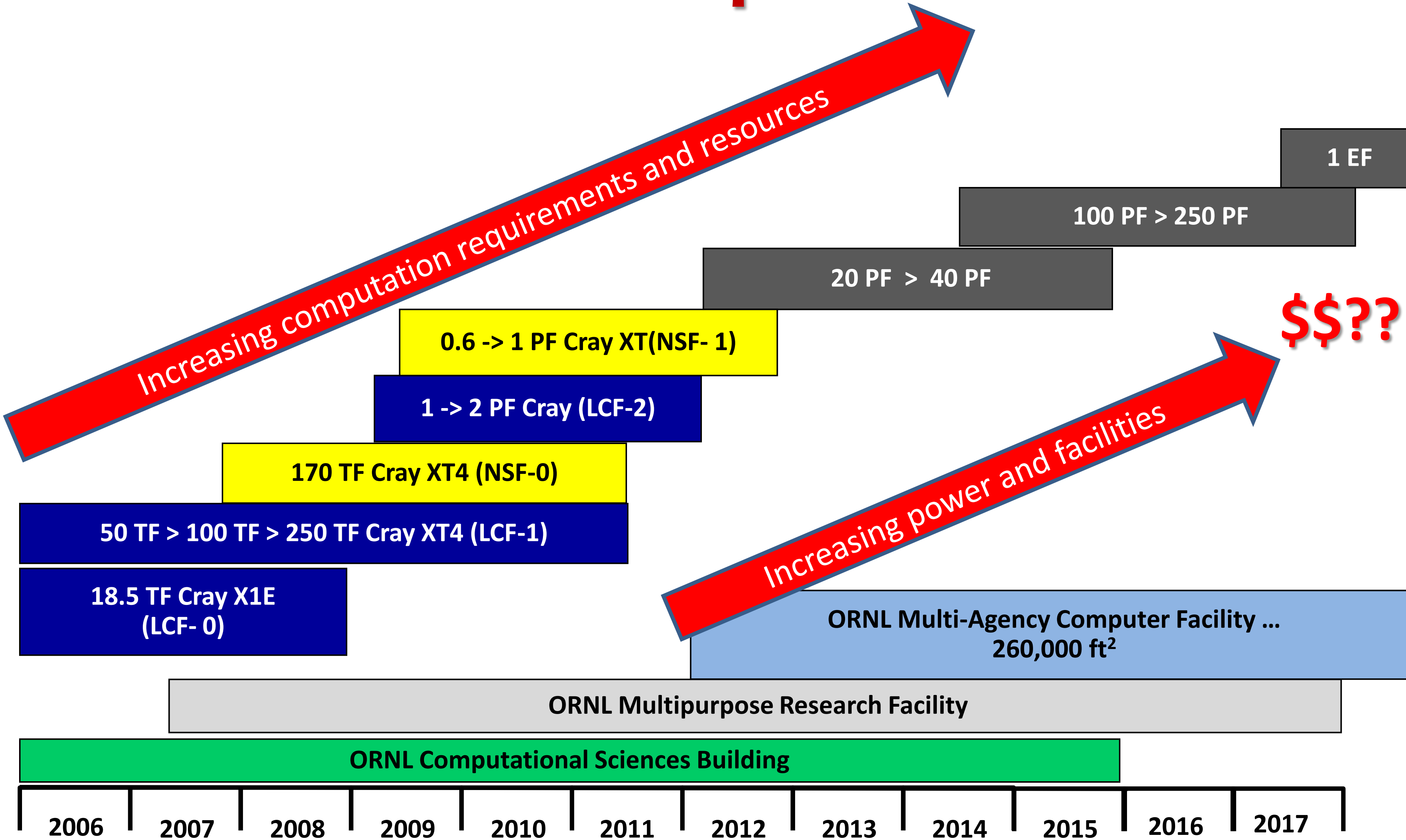
1 -> 2 PF Cray (LCF-2)

170 TF Cray XT4 (NSF-0)

50 TF > 100 TF > 250 TF Cray XT4 (LCF-1)

18.5 TF Cray X1E
(LCF- 0)

ORNL Roadmap to Exascale



Increasing computation requirements and resources

1 EF

100 PF > 250 PF

20 PF > 40 PF

0.6 -> 1 PF Cray XT(NSF- 1)

1 -> 2 PF Cray (LCF-2)

170 TF Cray XT4 (NSF-0)

50 TF > 100 TF > 250 TF Cray XT4 (LCF-1)

18.5 TF Cray X1E
(LCF- 0)

$$??

Increasing power and facilities

ORNL Multi-Agency Computer Facility …
260,000 ft^2

ORNL Multipurpose Research Facility

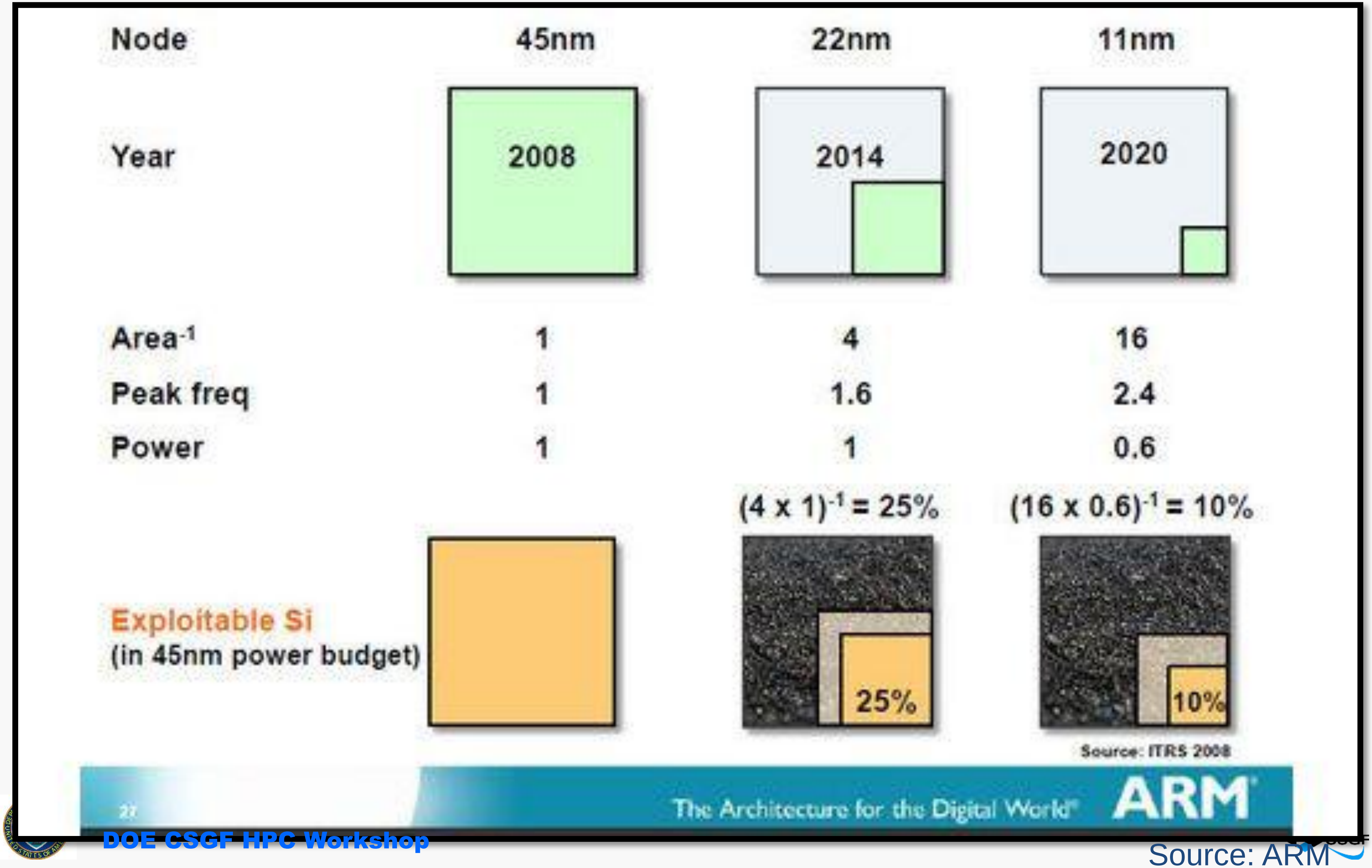
ORNL Computational Sciences Building

2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017

If energy costs ~$1/MW/yr, then how much is the energy cost for an exascale system?!?!

Facilities and Power … Not just ORNL

A more consumer-relevant trend: Dark Silicon



Node	45nm	22nm	11nm
Year	2008	2014	2020
Area^{-1}	1	4	16
Peak freq	1	1.6	2.4
Power	1	1	0.6
		$(4 \times 1)^{-1} = 25\%$	$(16 \times 0.6)^{-1} = 10\%$
Exploitable Si (in 45nm power budget)		25%	10%

Source: ITRS 2008

22

The Architecture for the Digital World®

ARM®

Source: ARM

CPU GPU SoC: Features & Block Diagram

CPU

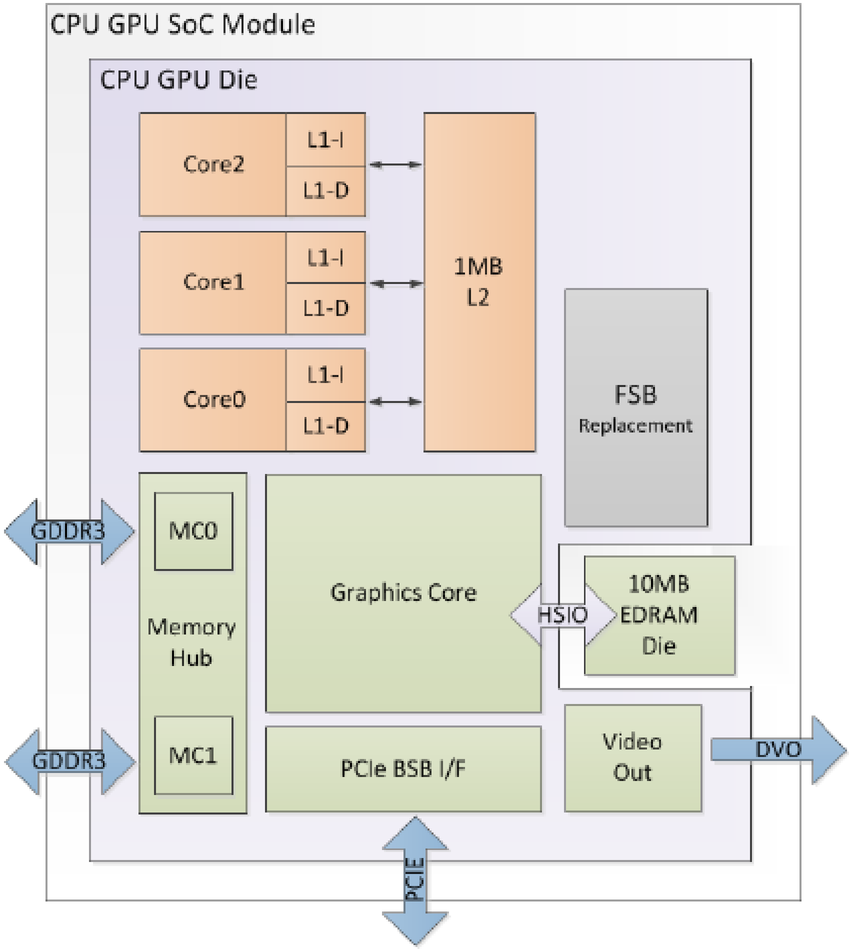
- Three 3.2 GHz PowerPC® cores
- Shared 1MB L2 cache
- Per Core:
    - Dual Thread Execution
    - 32K L1 I-cache, 32K L1 D-cache
    - 2-issue per cycle
    - Branch, Integer, Load/Store Units
    - VMX128 Units enhanced for games

GPU

- 48 parallel unified shaders
- 24 billion shader instructions per second
- 4 billion pixels/sec pixel fill rate
- 500 million triangles/sec geometry rate
- High Speed IO interface to 10 MB EDRAM

Compatibility

- Functional and Performance equivalent to prior Xbox 360 GPU/CPU
- FSB Latency and BW match prior FSB



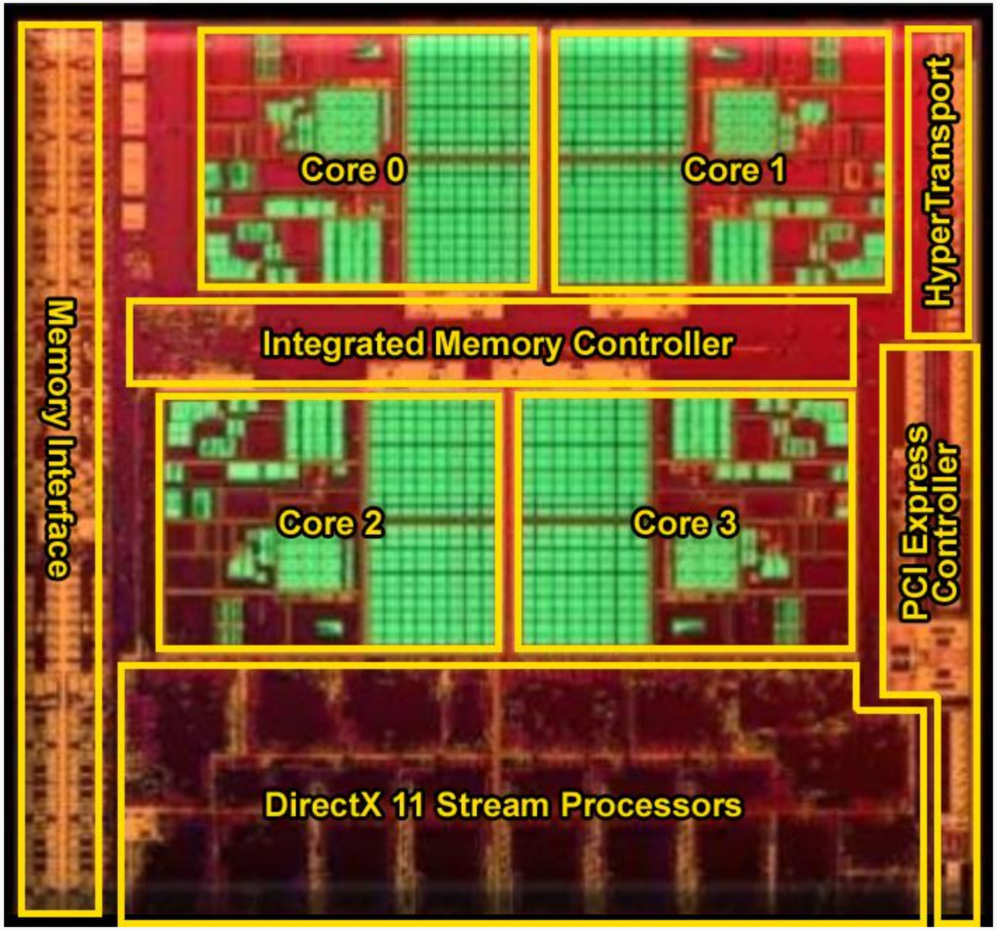
XBOX 360

Microsoft IBM

Source: Microsoft

AMD's Llano: A-Series APU

- Combines
  - 4 x86 cores
  - Array of Radeon cores
  - Multimedia accelerators
  - Dual channel DDR3
- 32nm
- Up to 29 GB/s memory bandwidth
- Up to 500 Gflops SP
- 45W TDP

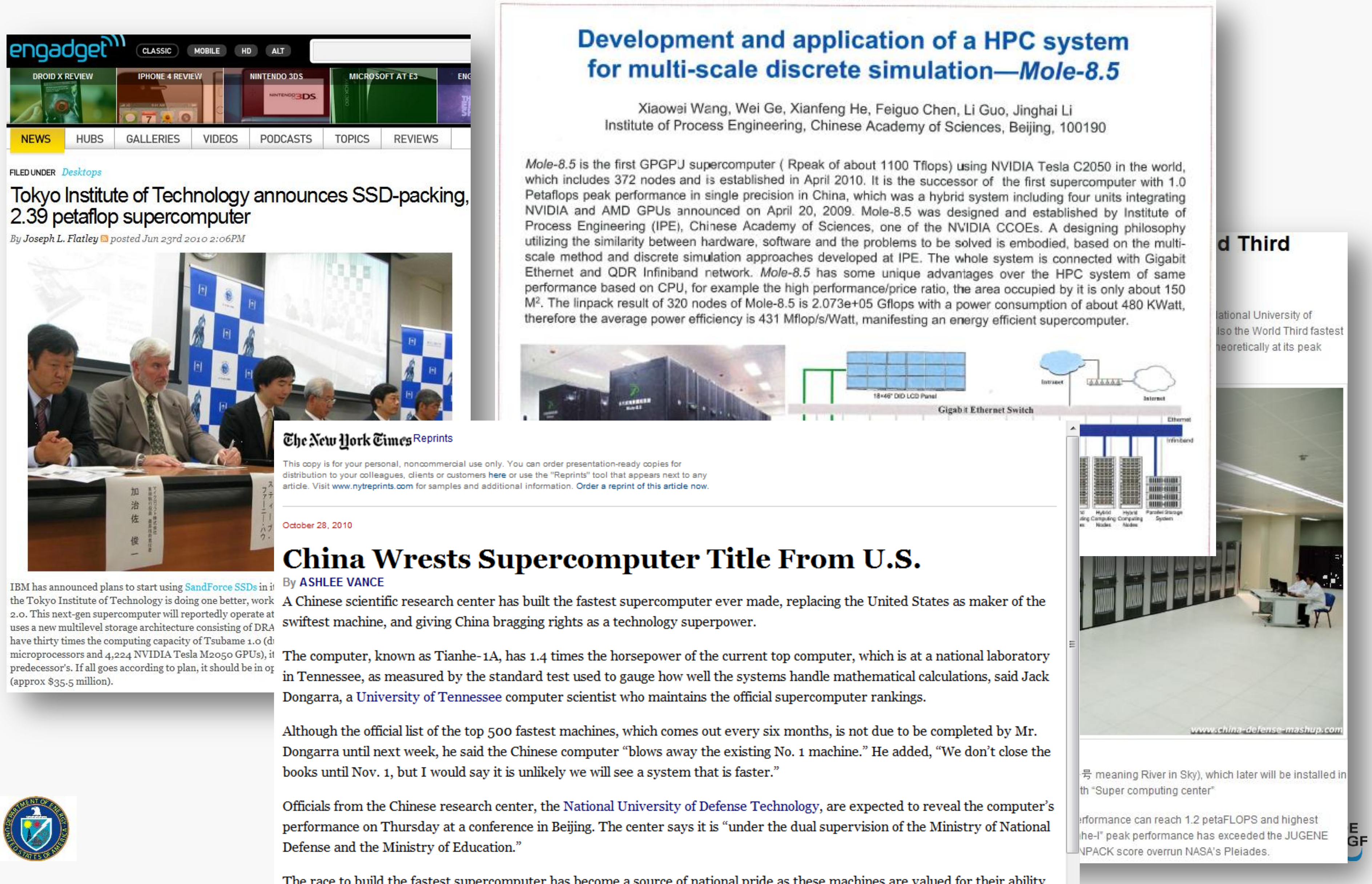


Source: AMD

A recent example…

GRAPHICS PROCESSORS

Many GPU-enabled systems blossoming worldwide….



FILED UNDER *Desktops*

Tokyo Institute of Technology announces SSD-packing, 2.39 petaflop supercomputer

By Joseph L. Flatley posted Jun 23rd 2010 2:06PM

IBM has announced plans to start using SandForce SSDs in it
the Tokyo Institute of Technology is doing one better, work
2.0. This next-gen supercomputer will reportedly operate at
uses a new multilevel storage architecture consisting of DRA
have thirty times the computing capacity of Tsubame 1.0 (d
microprocessors and 4,224 NVIDIA Tesla M2050 GPUs), it
predecessor's. If all goes according to plan, it should be in op
(approx $35.5 million).

Development and application of a HPC system for multi-scale discrete simulation—*Mole-8.5*

Xiaowei Wang, Wei Ge, Xianfeng He, Feiguo Chen, Li Guo, Jinghai Li
Institute of Process Engineering, Chinese Academy of Sciences, Beijing, 100190

Mole-8.5 is the first GPGPU supercomputer (Rpeak of about 1100 Tflops) using NVIDIA Tesla C2050 in the world, which includes 372 nodes and is established in April 2010. It is the successor of the first supercomputer with 1.0 Petaflops peak performance in single precision in China, which was a hybrid system including four units integrating NVIDIA and AMD GPUs announced on April 20, 2009. Mole-8.5 was designed and established by Institute of Process Engineering (IPE), Chinese Academy of Sciences, one of the NVIDIA CCOEs. A designing philosophy utilizing the similarity between hardware, software and the problems to be solved is embodied, based on the multi-scale method and discrete simulation approaches developed at IPE. The whole system is connected with Gigabit Ethernet and QDR Infiniband network. *Mole-8.5* has some unique advantages over the HPC system of same performance based on CPU, for example the high performance/price ratio, the area occupied by it is only about 150 M². The linpack result of 320 nodes of Mole-8.5 is 2.073e+05 Gflops with a power consumption of about 480 KWatt, therefore the average power efficiency is 431 Mflop/s/Watt, manifesting an energy efficient supercomputer.

The New York Times Reprints

This copy is for your personal, noncommercial use only. You can order presentation-ready copies for distribution to your colleagues, clients or customers here or use the "Reprints" tool that appears next to any article. Visit www.nytreprints.com for samples and additional information. Order a reprint of this article now.

October 28, 2010

China Wrests Supercomputer Title From U.S.

By ASHLEE VANCE

A Chinese scientific research center has built the fastest supercomputer ever made, replacing the United States as maker of the swiftest machine, and giving China bragging rights as a technology superpower.

The computer, known as Tianhe-1A, has 1.4 times the horsepower of the current top computer, which is at a national laboratory in Tennessee, as measured by the standard test used to gauge how well the systems handle mathematical calculations, said Jack Dongarra, a University of Tennessee computer scientist who maintains the official supercomputer rankings.

Although the official list of the top 500 fastest machines, which comes out every six months, is not due to be completed by Mr. Dongarra until next week, he said the Chinese computer "blows away the existing No. 1 machine." He added, "We don't close the books until Nov. 1, but I would say it is unlikely we will see a system that is faster."

Officials from the Chinese research center, the National University of Defense Technology, are expected to reveal the computer's performance on Thursday at a conference in Beijing. The center says it is "under the dual supervision of the Ministry of National Defense and the Ministry of Education."

The race to build the fastest supercomputer has become a source of national pride as these machines are valued for their ability

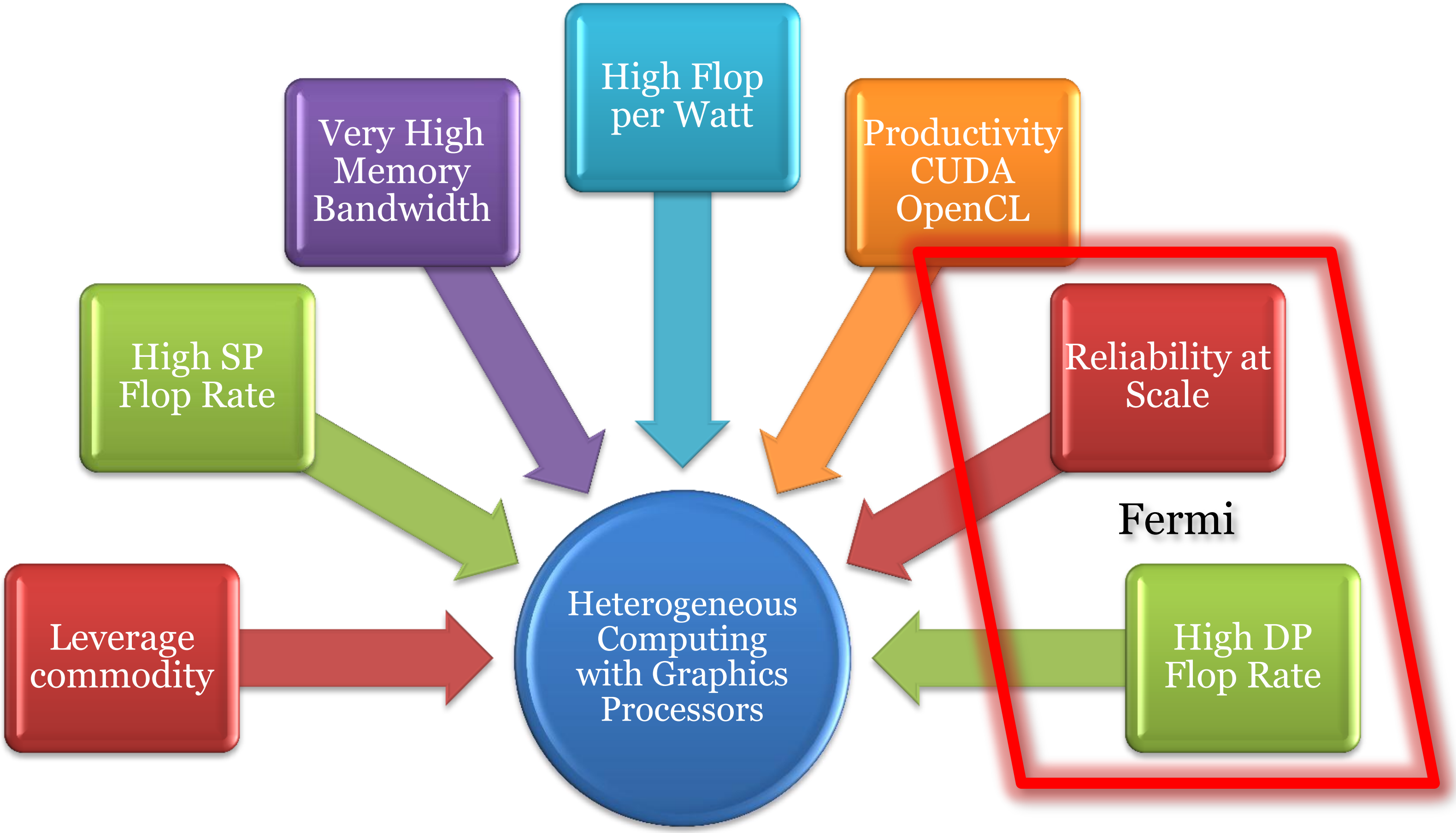
d Third

lational University of
lso the World Third fastest
heoretically at its peak

号 meaning River in Sky), which later will be installed in
th "Super computing center"

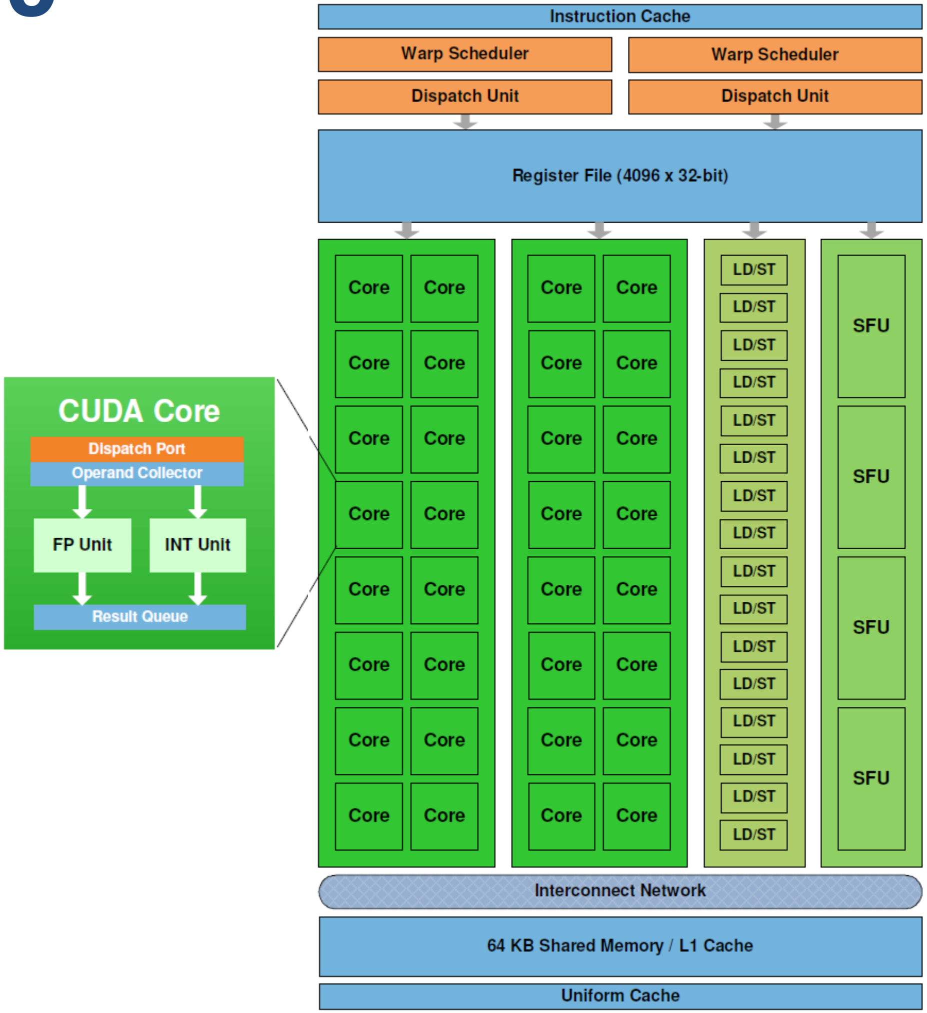
erformance can reach 1.2 petaFLOPS and highest
he-I" peak performance has exceeded the JUGENE
NPACK score overrun NASA's Pleiades.

GPU Rationale – What’s different now?



NVIDIA Fermi/GF100

- 3B transistors in 40nm
- Up to 512 CUDA Cores
  - New IEEE 754-2008 floating-point standard
    - FMA
    - 8× the peak double precision arithmetic performance over NVIDIA's last generation GPU
  - 32 cores per SM, 21k threads per chip
- 384b GDDR5, 6 GB capacity
  - ~120-144 GB/s memory BW
- C/M2070
  - 515 GigaFLOPS DP, 6GB
  - ECC Register files, L1/L2 caches, shared memory and DRAM



Keeneland – Initial Delivery System Architecture

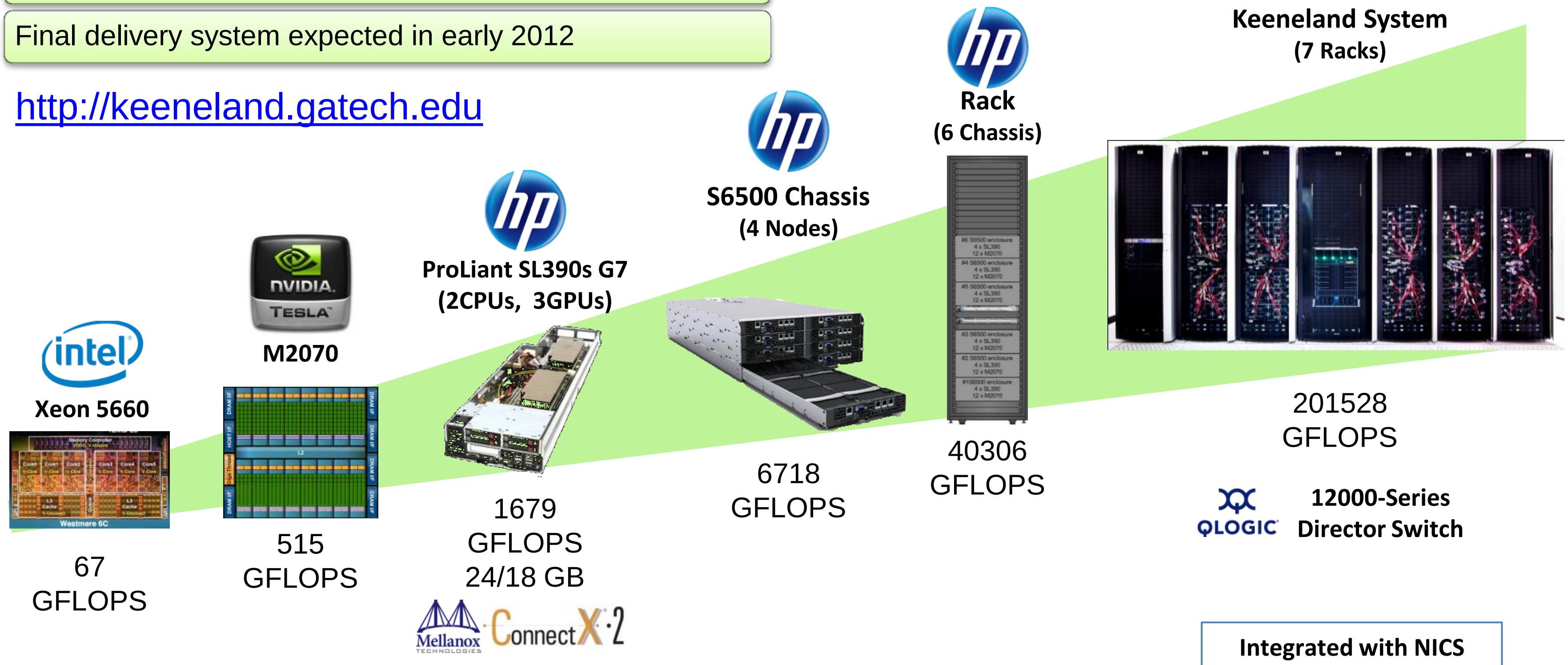
Initial Delivery system procured and installed in Oct 2010

201 TFLOPS in 7 racks (90 sq ft incl service area)

677 MFLOPS per watt on HPL

Final delivery system expected in early 2012

http://keeneland.gatech.edu



Xeon 5660

67 GFLOPS

M2070

515 GFLOPS

ProLiant SL390s G7 (2CPUs, 3GPUs)

1679 GFLOPS
24/18 GB

S6500 Chassis (4 Nodes)

6718 GFLOPS

Rack (6 Chassis)

40306 GFLOPS

Keeneland System (7 Racks)

201528 GFLOPS

12000-Series Director Switch

Full PCIe X16 bandwidth to all GPUs

Integrated with NICS Datacenter GPFS and TG

Early (Co-design) Success Stories

Computational Materials

- Quantum Monte Carlo
  - High-temperature superconductivity and other materials science
  - 2008 Gordon Bell Prize
- GPU acceleration speedup of 19x in main QMC Update routine
  - Single precision for CPU and GPU: target single-precision only cards
- *Full parallel app* is 5x faster, start to finish, on a GPU-enabled cluster on Tesla T10

GPU study: J.S. Meredith, G. Alvarez, T.A. Maier, T.C. Schulthess, J.S. Vetter, “Accuracy and Performance of Graphics Processors: A Quantum Monte Carlo Application Case Study”, *Parallel Comput., 35(3):151-63, 2009.*

Accuracy study: G. Alvarez, M.S. Summers, D.E. Maxwell, M. Eisenbach, J.S. Meredith, J. M. Larkin, J. Levesque, T. A. Maier, P.R.C. Kent, E.F. D'Azevedo, T.C. Schulthess, “New algorithm to enable 400+ TFlop/s sustained performance in simulations of disorder effects in high-Tc superconductors”, SuperComputing, 2008. [*Gordon Bell Prize winner*]

Combustion

- S3D
  - Massively parallel direct numerical solver (DNS) for the full compressible Navier-Stokes, total energy, species and mass continuity equations
  - Coupled with detailed chemistry
  - Scales to 225k cores on Jaguar
- Accelerated version of S3D’s Getrates *kernel* in CUDA on Tesla T10
  - 31.4x SP speedup
  - 16.2x DP speedup

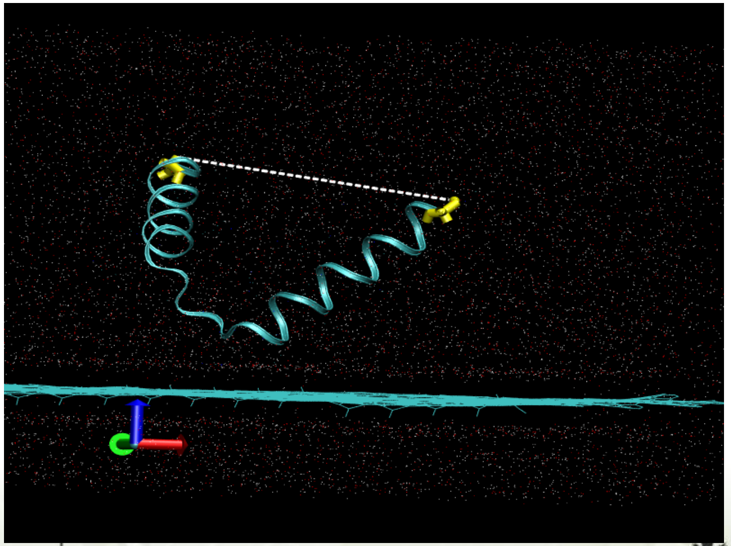
K. Spafford, J. Meredith, J. S. Vetter, J. Chen, R. Grout, and R. Sankaran. Accelerating S3D: A GPGPU Case Study. Proceedings of the Seventh International Workshop on Algorithms, Models, and Tools for Parallel Computing on Heterogeneous Platforms (HeteroPar 2009) Delft, The Netherlands.

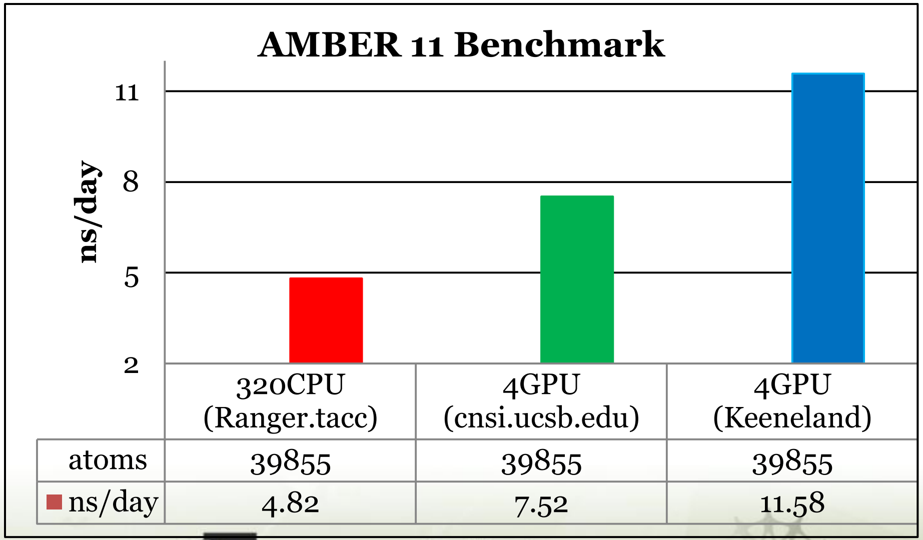
Joan-Emma Shea at UCSB

Peptide folding on surfaces

- Peptide folding on a hydrophobic surface
  - www.chem.ucsb.edu/~sheagroup
- Surfaces can modulate the folding and aggregation pathways of proteins. Here, we investigate the folding of a small helical peptide in the presence of a hydrophobic surface of graphite. Simulations are performed using explicit solvent and a fully atomic representation of the peptide and the surface.

- Benefits of running on a GPU cluster:
  - Reduction in the the number of computing nodes needed: one GPU is at least faster than 8 CPUs in GPU-accelerated AMBER Molecular Dynamics.
  - The large simulations that we are currently running would be prohibitive using CPUs. The efficiency of the CPU parallelization becomes poorer with increasing number of CPUs.
  - It can also decrease consumption of memory and network bandwidth in simulations with large number of atoms.





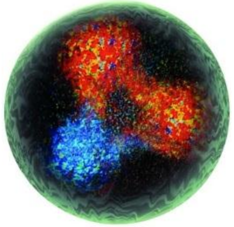
AMBER 11 Benchmark

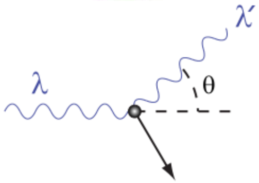
	320CPU (Ranger.tacc)	4GPU (cnsi.ucsb.edu)	4GPU (Keeneland)
atoms	39855	39855	39855
🟥 ns/day	4.82	7.52	11.58

Hadron Polarizability in Lattice QCD

Andrei Alexandru
The George Washington University

Understanding the structure of subnuclear particles represents the main challenge for today's nuclear physics. Photons are used to probe this structure in experiments carried out at laboratories around the world. To interpret the results of these experiments we need to understand how electromagnetic field interacts with subnuclear particles. Theoretically, the structure of subnuclear particles is described by Quantum Chromodynamics (QCD). Lattice QCD is a 4-dimensional discretized version of this theory that can be solved numerically. The focus of our project is to understand how the electric field deforms neutrons and protons by computing the polarizability using lattice QCD techniques.

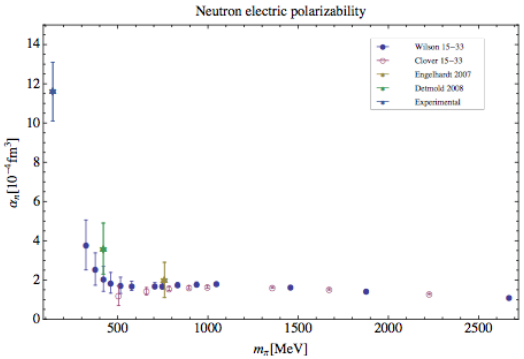


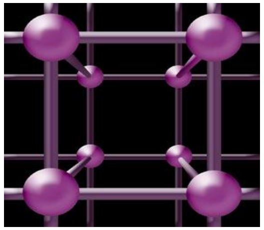


http://samurai.phys.gwu.edu/wiki/index.php/Hadron_polarizability

Why GPUs?

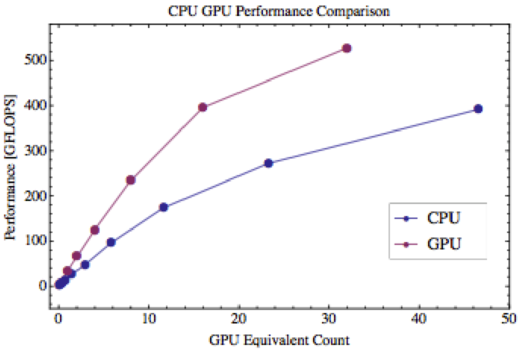
- Lattice QCD simulations require very large bandwidth to run efficiently. GPUs have 10–15 times larger memory bandwidth compared to CPUs.
- Lattice QCD simulations can be efficiently parallelized.
  - Bulk of calculation spent on one kernel.
  - The kernel requires only nearest neighbor information.
  - Cut the lattice into equal sub-lattices. Effectively use single instruction multiple-data (SIMD) paradigm.





Experimental and current values for neutron electric polarizability in lattice QCD.

Alexandru and F. X. Lee, [arXiv:0810.2833]



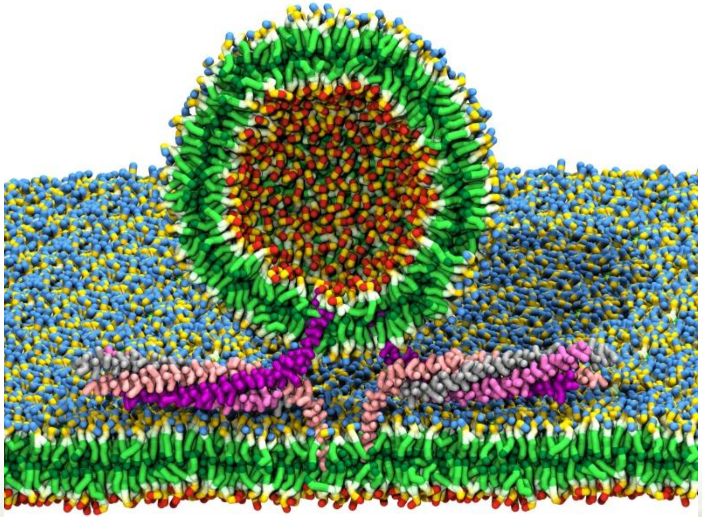
Performance comparison between Keeneland's GPU cluster and Kraken's Cray XT-5 machine. The CPU core count is translated to GPU equivalent count by dividing the total number of CPUs by 22, which is the number of CPU cores equivalent to a single-GPU performance.

A. Alexandru. *et. al*, [arXiv:1103.5103]

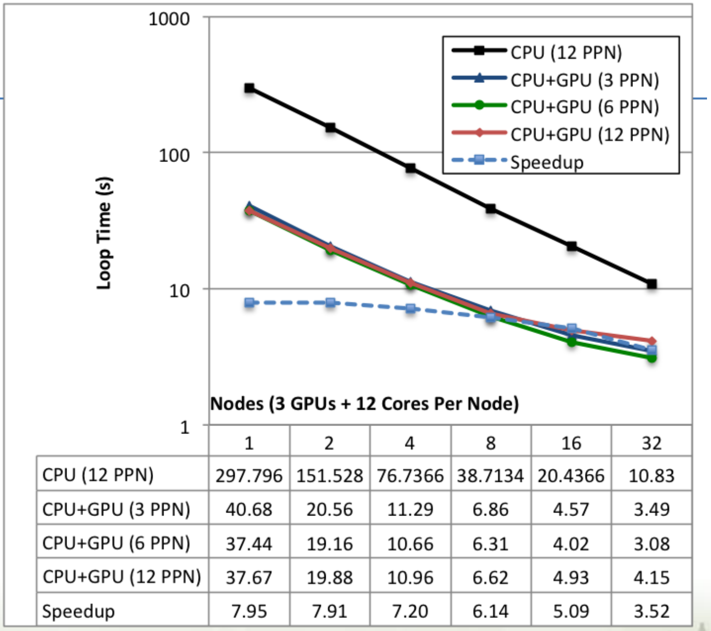
LAMMPS with GPUs

PI: Axel Kohlmeyer,
Temple University

- Parallel Molecular Dynamics
  - http://lammps.sandia.gov
  - Classical Molecular Dynamics
  - Atomic models, Polymers, Metals, Bio-simulations, Coarse-grain (picture), Ellipsoids, etc.
  - Already good strong and weak scaling on CPUs via MPI



- Better performance on fewer nodes => larger problems faster
- Neighbor, non-bonded force, and long-range GPU acceleration
- Allows for CPU/GPU concurrency
- Implementation and benchmarks by W. Michael Brown, NCCS, ORNL



	1	2	4	8	16	32
CPU (12 PPN)	297.796	151.528	76.7366	38.7134	20.4366	10.83
CPU+GPU (3 PPN)	40.68	20.56	11.29	6.86	4.57	3.49
CPU+GPU (6 PPN)	37.44	19.16	10.66	6.31	4.02	3.08
CPU+GPU (12 PPN)	37.67	19.88	10.96	6.62	4.93	4.15
Speedup	7.95	7.91	7.20	6.14	5.09	3.52

#5: PROGRAMMING SYSTEMS

Holistic View of HPC

Performance, Resilience, Power, Programmability

Applications

- Materials
- Climate
- Fusion
- National Security
- Combustion
- Nuclear Energy
- Cybersecurity
- Biology
- High Energy Physics
- Energy Storage
- Photovoltaics
- National Competitiveness

- Usage Scenarios
  - Ensembles
  - UQ
  - Visualization
  - Analytics

Programming Environment

- Domain specific
  - Libraries
  - Frameworks
  - Templates
  - Domain specific languages
  - Patterns
  - Autotuners
- Platform specific
  - Languages
  - Compilers
  - Interpreters/Scripting
  - Performance and Correctness Tools
  - Source code control

Critical

System Software

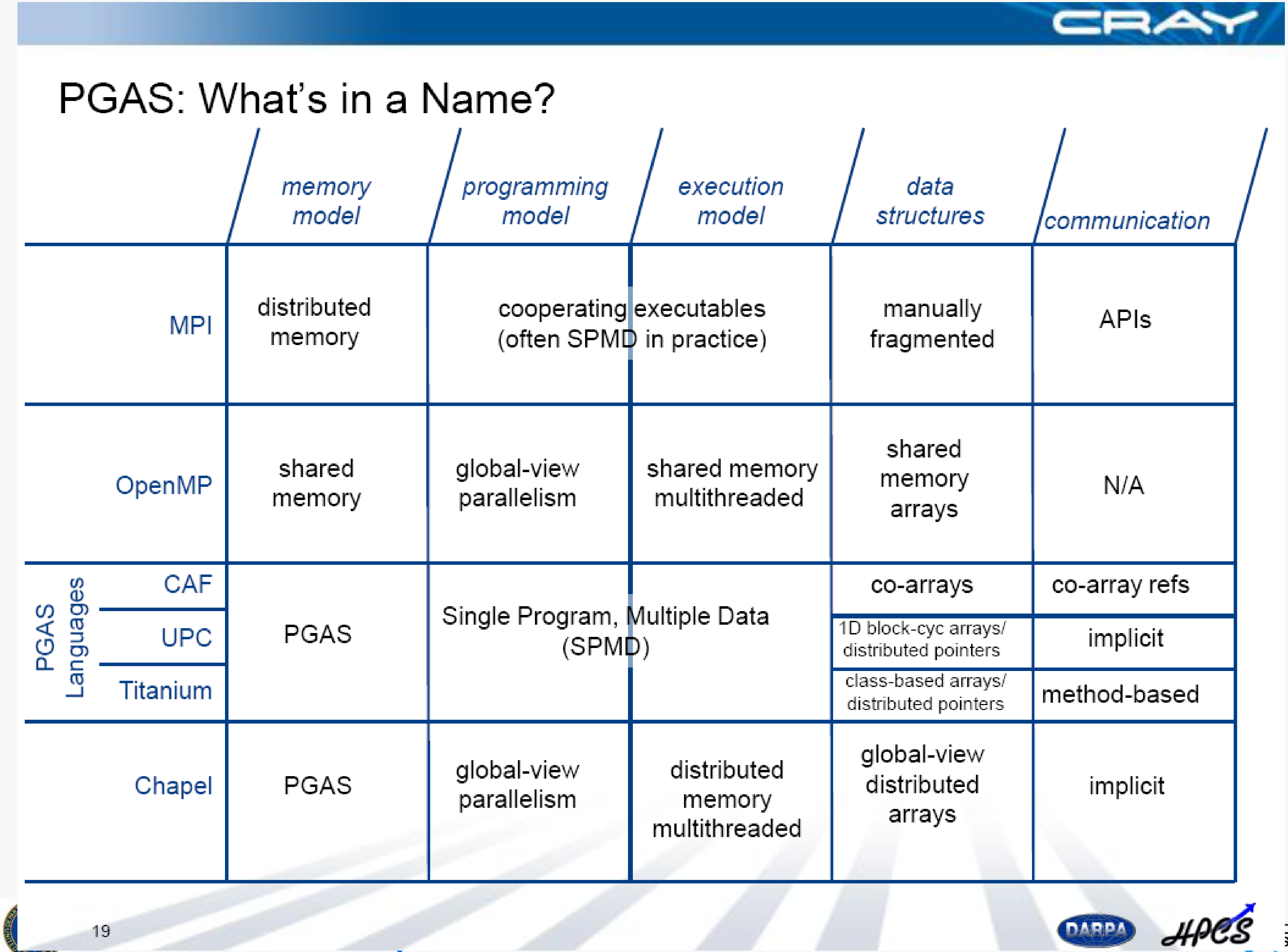
- Resource Allocation
- Scheduling
- Security
- Communication
- Synchronization
- Filesystems
- Instrumentation
- Virtualization

Architectures

- Processors
  - Multicore
  - Graphics Processors
  - FPGA
  - DSP
- Memory and Storage
  - Shared (cc, scratchpad)
  - Distributed
  - RAM
  - Storage Class Memory
  - Disk
  - Archival
- Interconnects
  - Infiniband
  - IBM Torrent
  - Cray Gemini, Aires
  - BGL/P/Q
  - 1/10/100 GigE

State of Programming Systems

- Contemporary Programming Systems USED IN HPC
  - C, C++, FORTRANXX
  - MPI, OpenMP
  - CUDA, OpenCL
  - Python, UPC, CAF
  - Combinations of these: MPI+OpenMP+CUDA
- The future is completely open:
  - Global Arrays, Charm++, ParalleX, StarSS, Cilk, TBB, CnC, parallel Matlabs, Star-P, C++AMP, Map-Reduce, Titanium, Sequoia, Chapel, etc
- Libraries and Frameworks provide functional consistency
  - BLAS, LAPACK, PetSC, Trilinos, OpenMM, FFTW, …



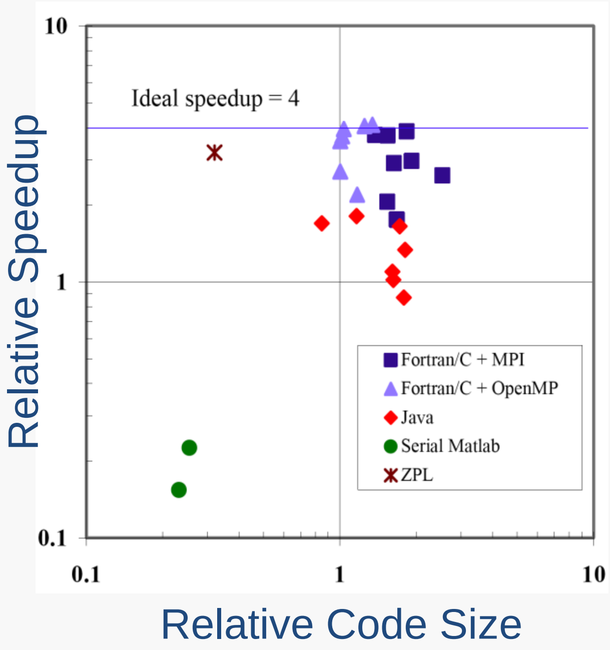
PGAS: What's in a Name?

		memory model	*programming model*	*execution model*	*data structures*	*communication*
	MPI	distributed memory	cooperating executables (often SPMD in practice)		manually fragmented	APIs
	OpenMP	shared memory	global-view parallelism	shared memory multithreaded	shared memory arrays	N/A
PGAS Languages	CAF	PGAS	Single Program, Multiple Data (SPMD)		co-arrays	co-array refs
	UPC				1D block-cyc arrays/ distributed pointers	implicit
	Titanium				class-based arrays/ distributed pointers	method-based
	Chapel	PGAS	global-view parallelism	distributed memory multithreaded	global-view distributed arrays	implicit

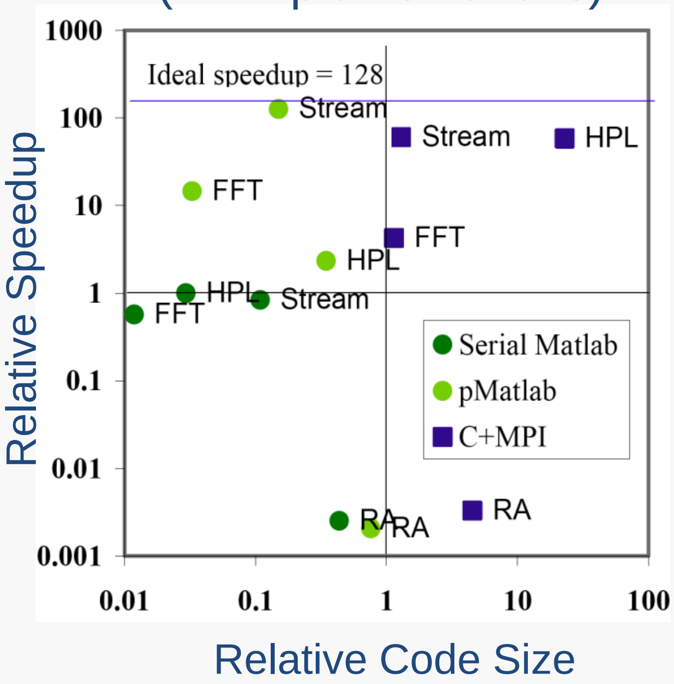
Source: Brad Chamberlain, Cray

Is it possible to write one application that can run efficiently on all these architectures?

NAS Benchmarks
(~35 implementations)



HPC Challenge
(12 implementations)



Source: DARPA HPCS Program, 2005

Questions for Exascale Programming Systems

Q: How should we expose multiple levels of parallelism?

Q: How should communication occur?

Q: Should thread-data locality/affinity be exposed to the user, or hidden managed by the runtime?

Q: How should we best enabled domain specific libraries, frameworks, and languages?

Q: How do we maintain legacy applications and software?

A concrete example…

THE RECENT QUEST ON PROGRAMMING GPUS

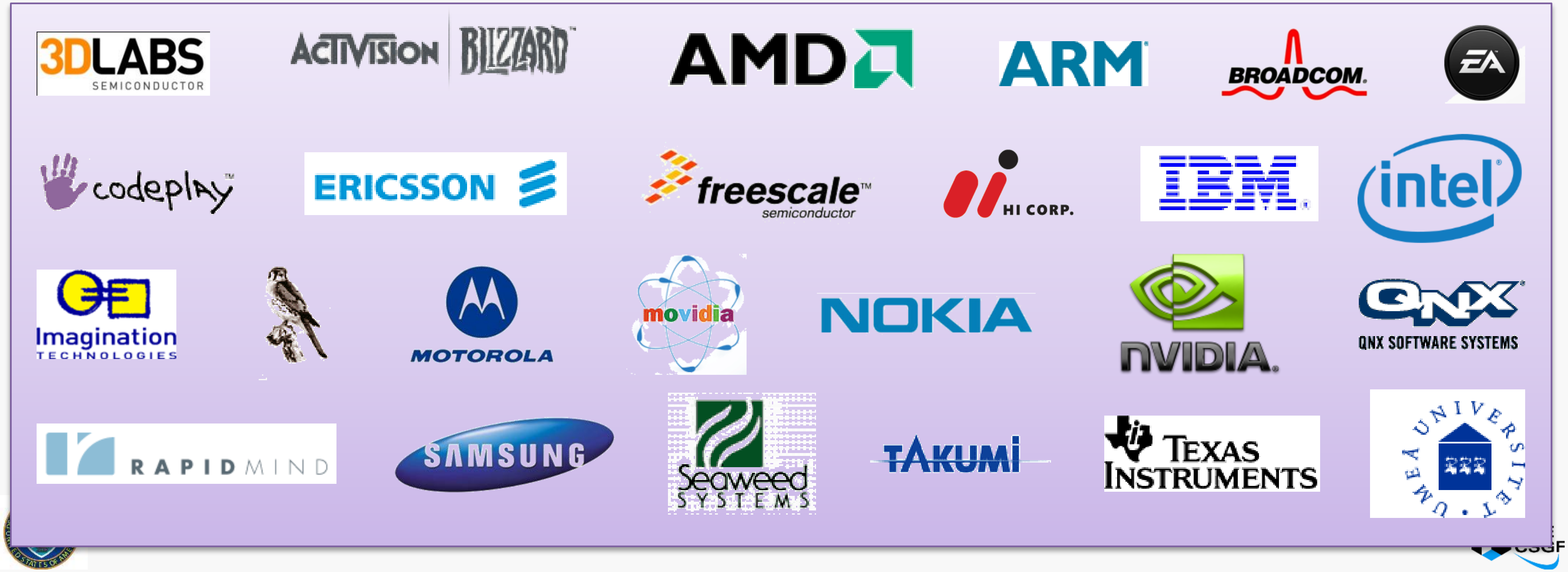
A Typical GPU Software Environment

- Integrated with NSF TeraGrid/XD
  - Including TG and NICS software stack
- Programming environments
  - CUDA
  - OpenCL
  - Compilers
    - GPU-enabled
  - Scalable debuggers
  - Performance tools
  - Libraries

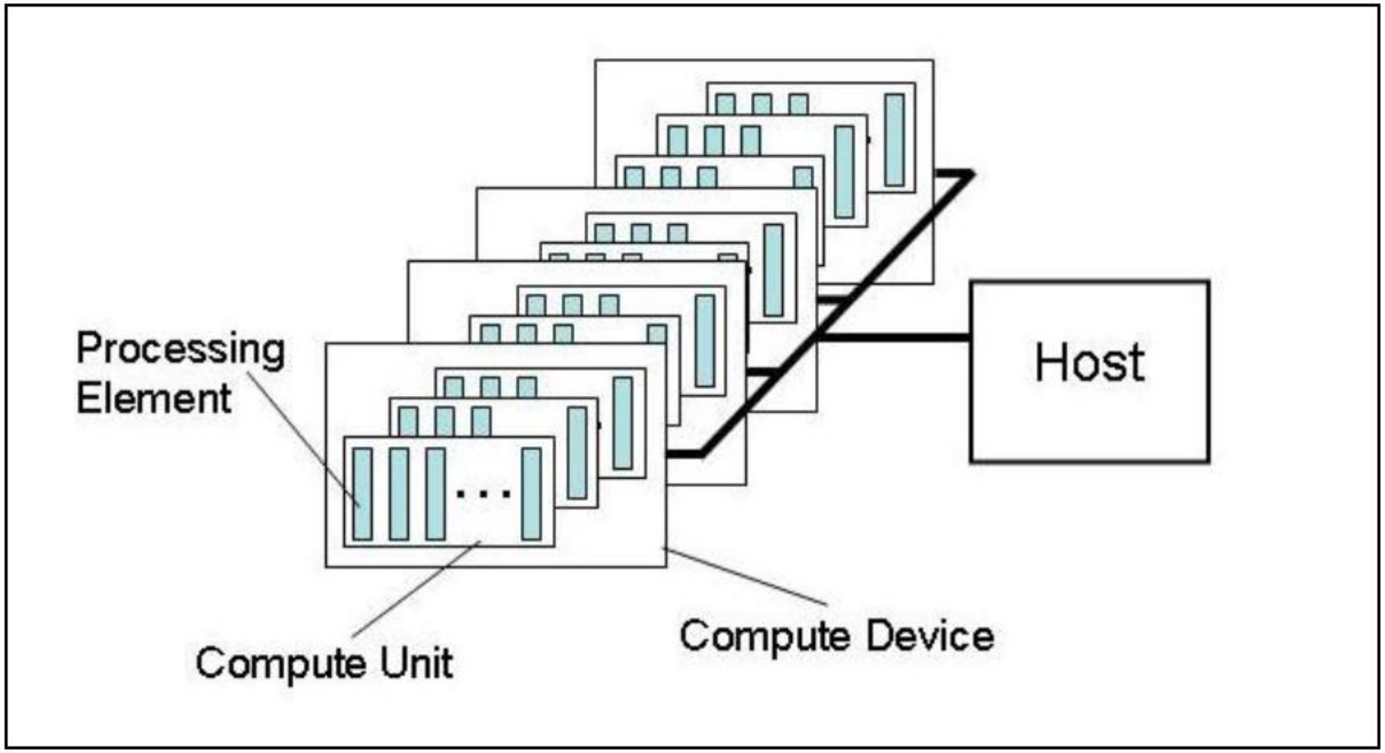
- Tools and programming options are changing rapidly
  - HMPP, PGI, LLVM, Polaris, R-stream,
- Additional software activities
  - Performance and correctness tools
  - Scientific libraries
  - Virtualization

OpenCL Working Group

- Diverse industry participation
  - Processor vendors, system OEMs, middleware vendors, application developers
- Many industry-leading experts involved in OpenCL's design
  - A healthy diversity of industry perspectives
- Apple initially proposed and is very active in the working group
  - Serving as specification editor
- Here are some of the other companies in the OpenCL working group

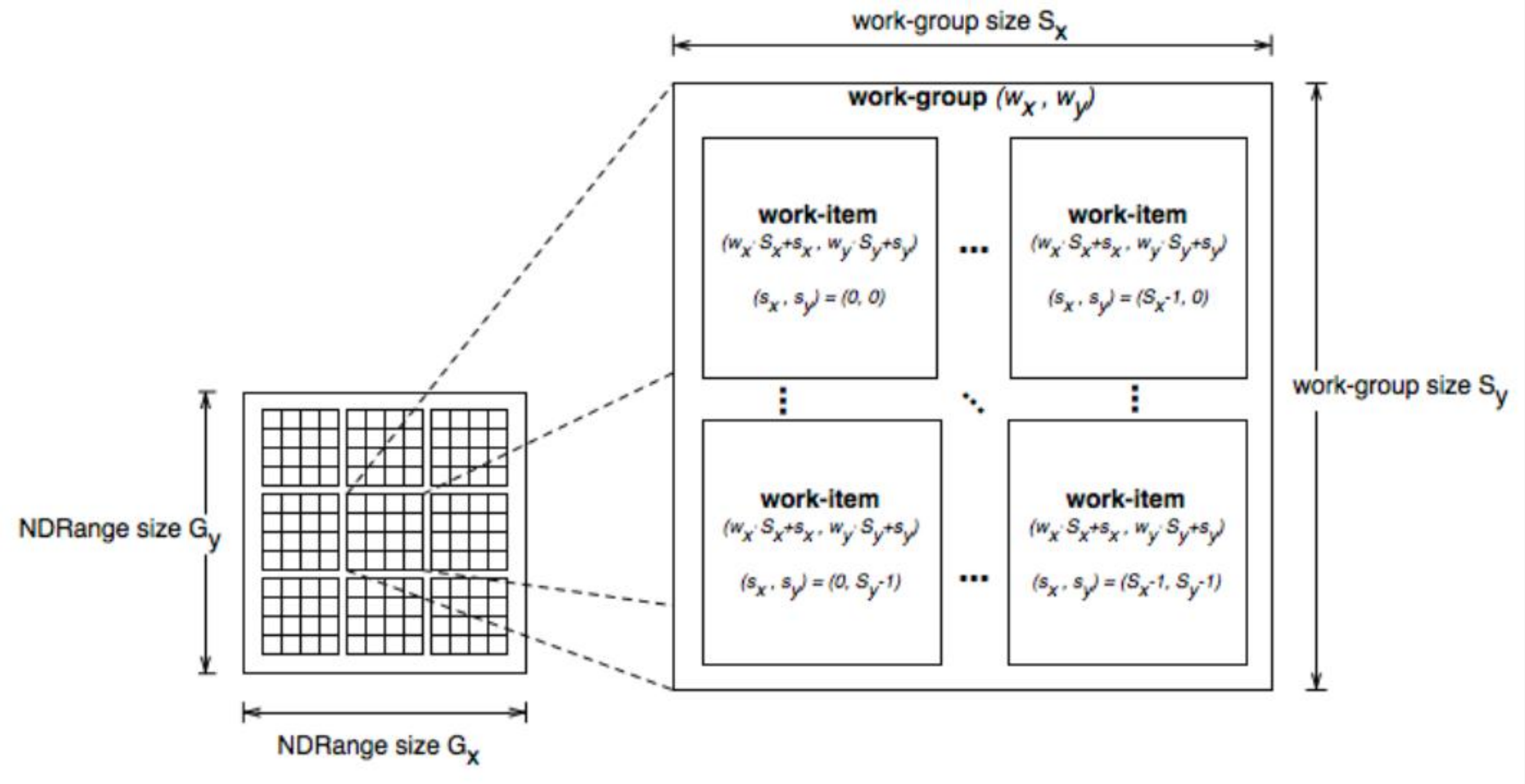


OpenCL Platform Model (Section 3.1)



- One Host + one or more Compute Devices
  - Each Compute Device is composed of one or more Compute Units
    - Each Compute Unit is further divided into one or more Processing Elements

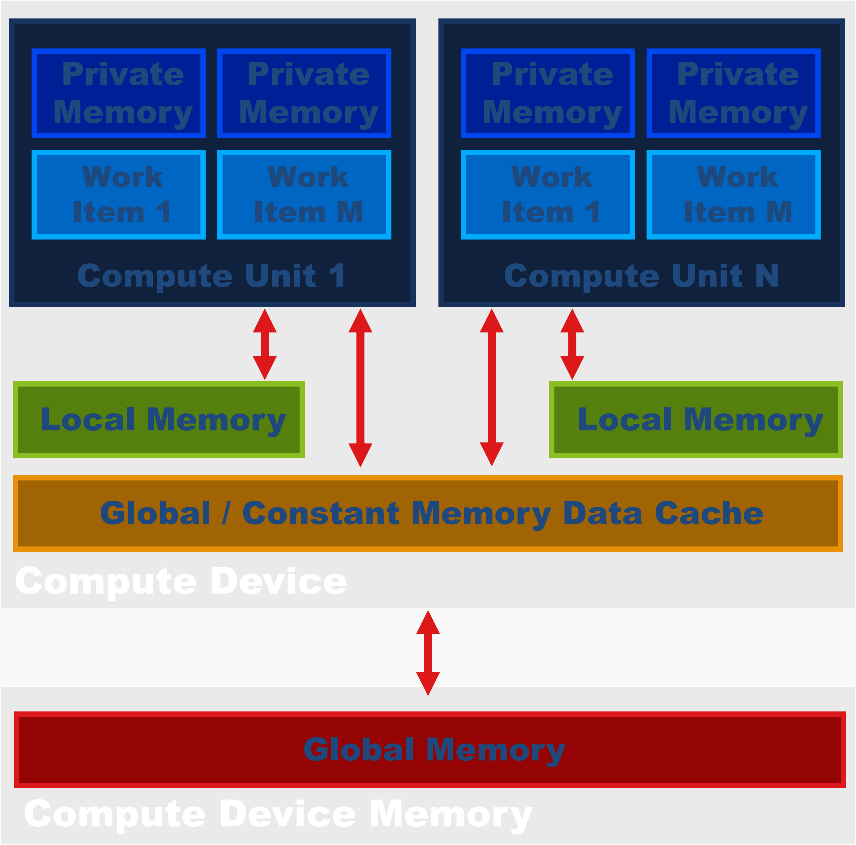
Kernel Execution


work-group size S_x
work-group (w_x, w_y)
work-item
$(w_x \cdot S_x+s_x, w_y \cdot S_y+s_y)$
$(s_x, s_y) = (0, 0)$
work-item
$(w_x \cdot S_x+s_x, w_y \cdot S_y+s_y)$
$(s_x, s_y) = (S_x-1, 0)$
work-item
$(w_x \cdot S_x+s_x, w_y \cdot S_y+s_y)$
$(s_x, s_y) = (0, S_y-1)$
work-item
$(w_x \cdot S_x+s_x, w_y \cdot S_y+s_y)$
$(s_x, s_y) = (S_x-1, S_y-1)$
work-group size S_y
NDRange size G_y
NDRange size G_x

- Total number of work-items = G_x x G_y
- Size of each work-group = S_x x S_y
- Global ID can be computed from work-group ID and local ID

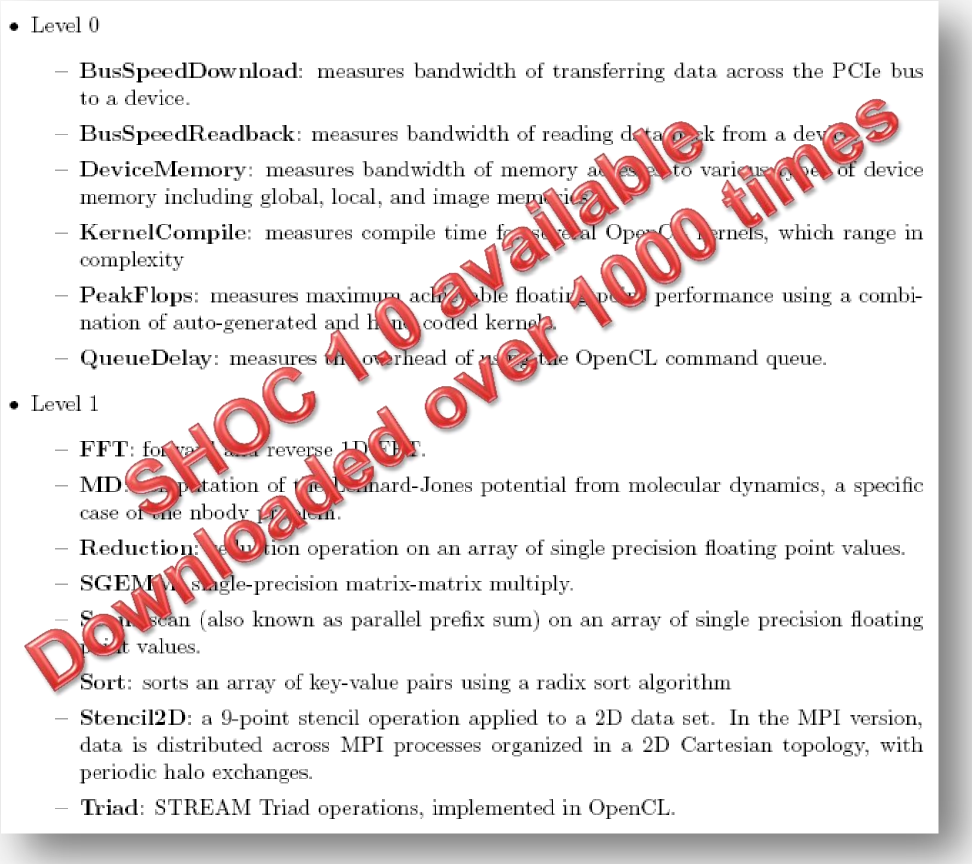
OpenCL Memory Model

- Shared memory model
  - Relaxed consistency
- Multiple distinct address spaces
  - Address spaces can be collapsed depending on the device's memory subsystem
- Address spaces
  - Private - private to a work-item
  - Local - local to a work-group
  - Global - accessible by all work-items in all work-groups
  - Constant - read only global space
- Implementations map this hierarchy
  - To available physical memories



Scalable HeterOgeneous Computing (SHOC) Benchmark Suite

- Benchmark suite with a focus on scientific computing workloads, including common kernels like SGEMM, FFT, Stencils
- Parallelized with MPI, with support for multi-GPU and cluster scale comparisons
- Implemented in CUDA and OpenCL for a 1:1 performance comparison
- Includes stability tests
- Performance portability

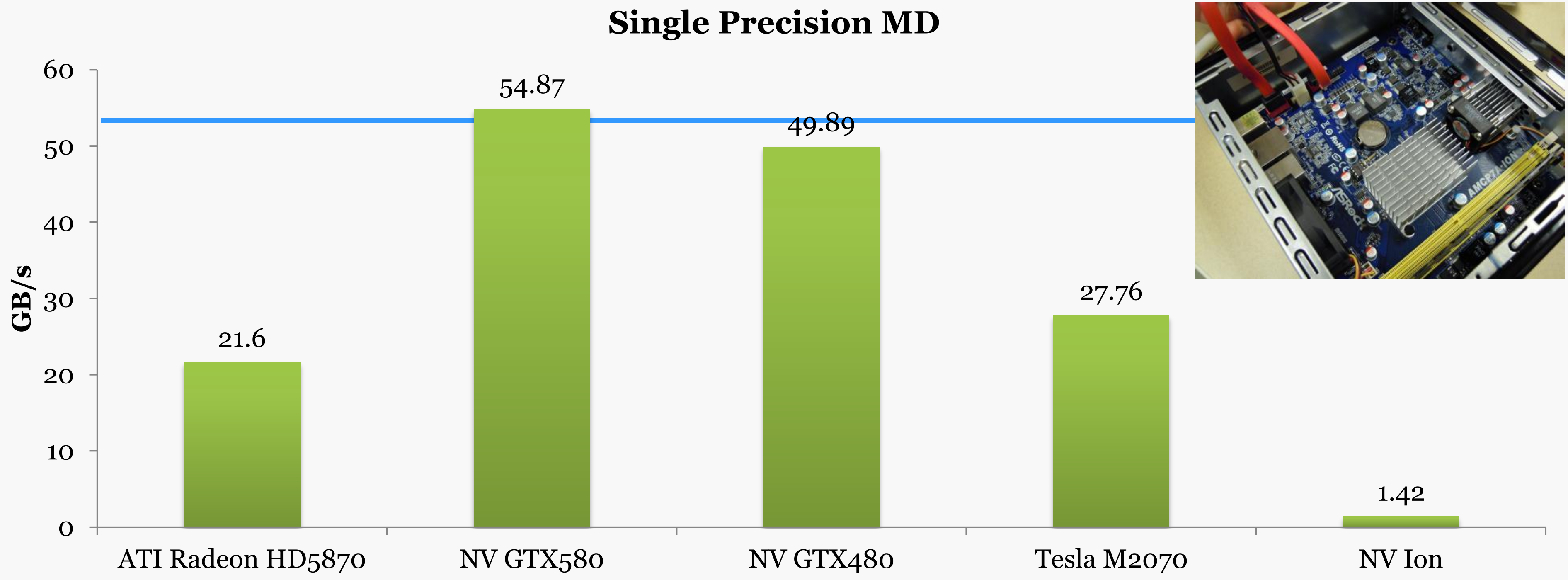


- Level 0
  - **BusSpeedDownload**: measures bandwidth of transferring data across the PCIe bus to a device.
  - **BusSpeedReadback**: measures bandwidth of reading data back from a device.
  - **DeviceMemory**: measures bandwidth of memory accesses to various types of device memory including global, local, and image memories.
  - **KernelCompile**: measures compile time for several OpenCL kernels, which range in complexity
  - **PeakFlops**: measures maximum achievable floating point performance using a combination of auto-generated and hand coded kernels.
  - **QueueDelay**: measures the overhead of using the OpenCL command queue.
- Level 1
  - **FFT**: forward and reverse 1D FFT.
  - **MD**: computation of the Lennard-Jones potential from molecular dynamics, a specific case of the nbody problem.
  - **Reduction**: reduction operation on an array of single precision floating point values.
  - **SGEMM**: single-precision matrix-matrix multiply.
  - **Scan**: scan (also known as parallel prefix sum) on an array of single precision floating point values.
  - **Sort**: sorts an array of key-value pairs using a radix sort algorithm
  - **Stencil2D**: a 9-point stencil operation applied to a 2D data set. In the MPI version, data is distributed across MPI processes organized in a 2D Cartesian topology, with periodic halo exchanges.
  - **Triad**: STREAM Triad operations, implemented in OpenCL.

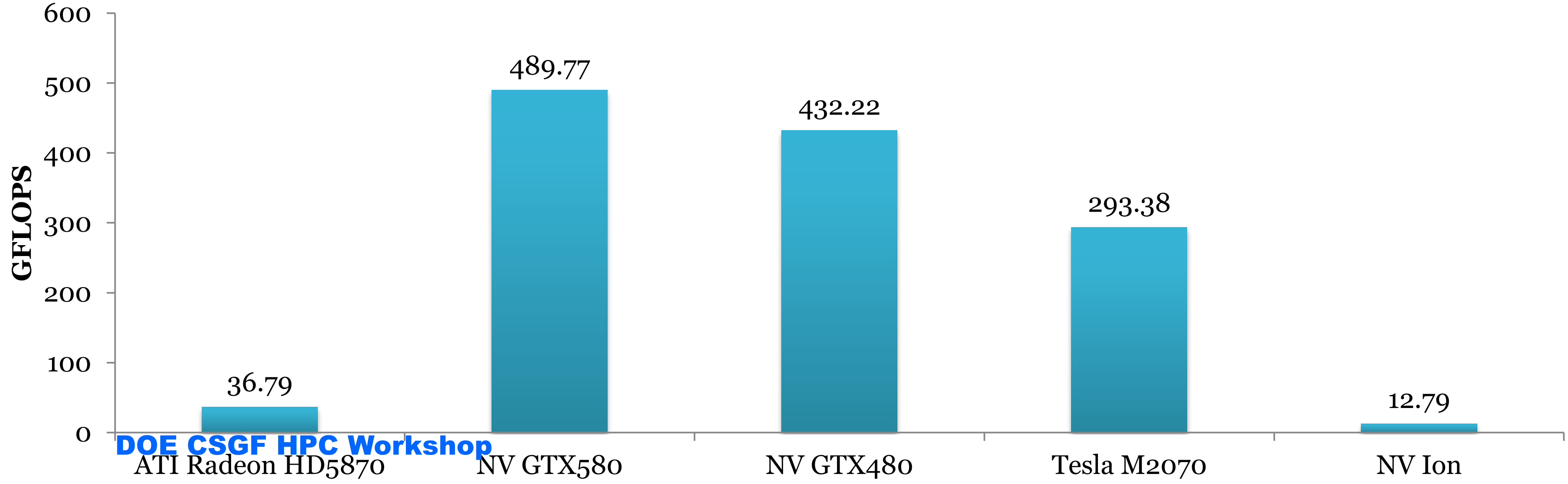
SHOC 1.0 available
Downloaded over 1000 times

Software available at *http://bit.ly/shocmarx*

A. Danalis, G. Marin, C. McCurdy, J. Meredith, P.C. Roth, K. Spafford, V. Tipparaju, and J.S. Vetter, "The Scalable HeterOgeneous Computing (SHOC) Benchmark Suite," in Third Workshop on General-Purpose Computation on Graphics Processors (GPGPU 2010)`. Pittsburgh, 2010



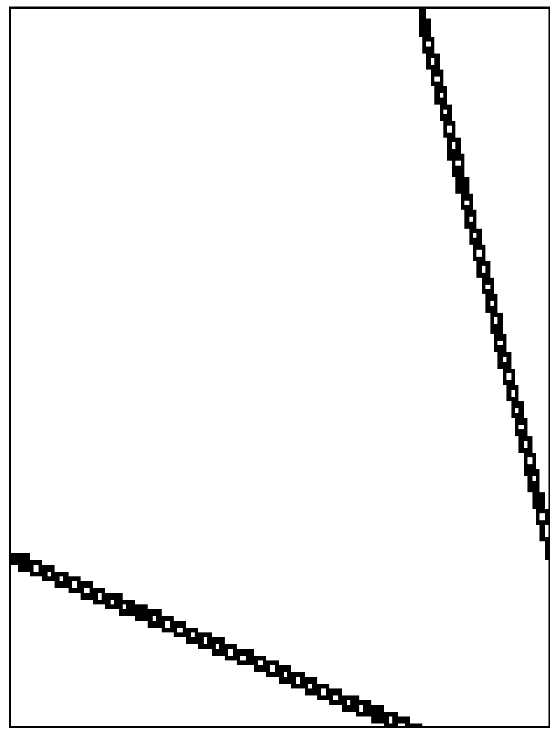
Single Precision MD



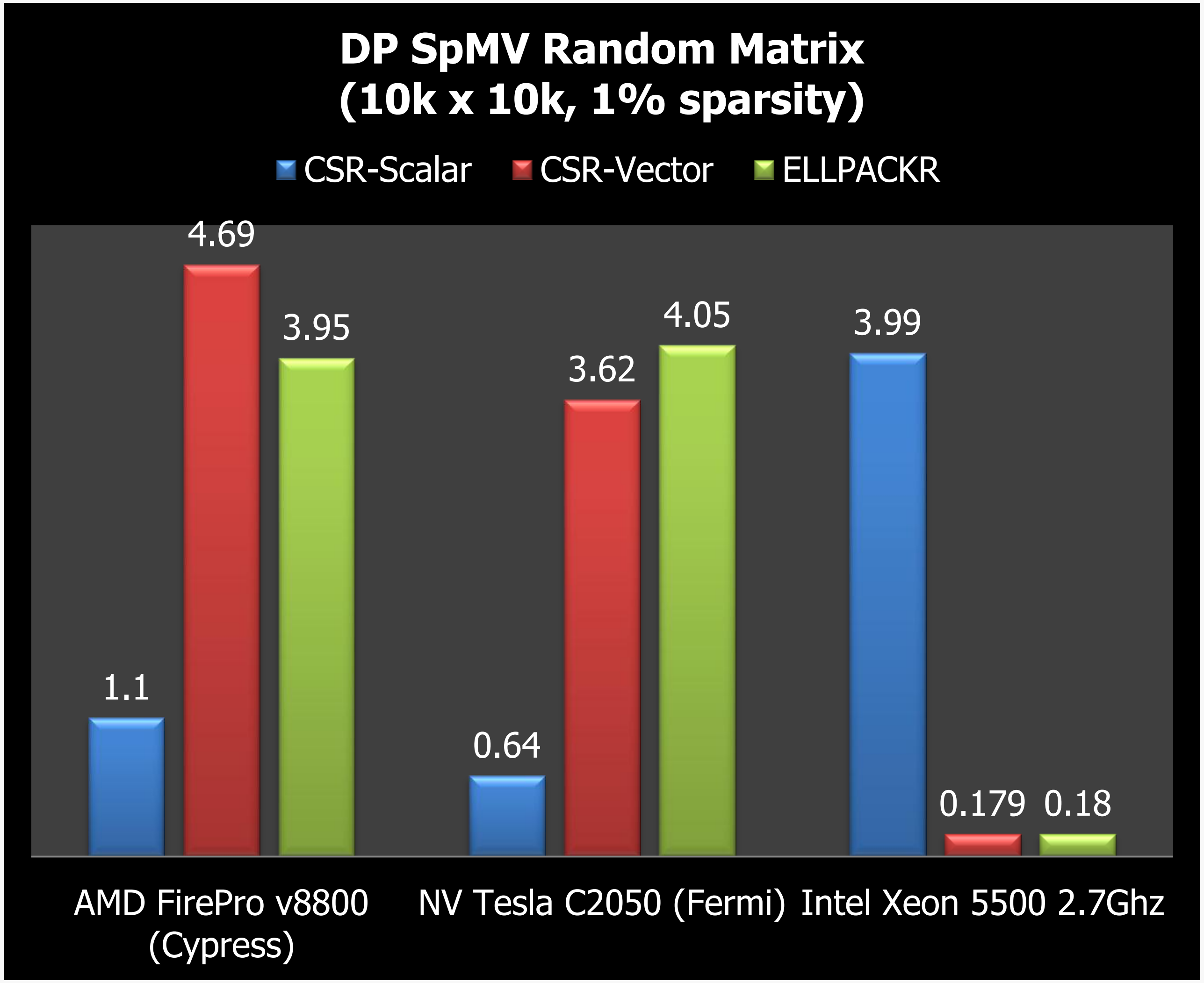
Single Precision FFT

Example: Sparse Matrix Vector Multiplication (SpMV)

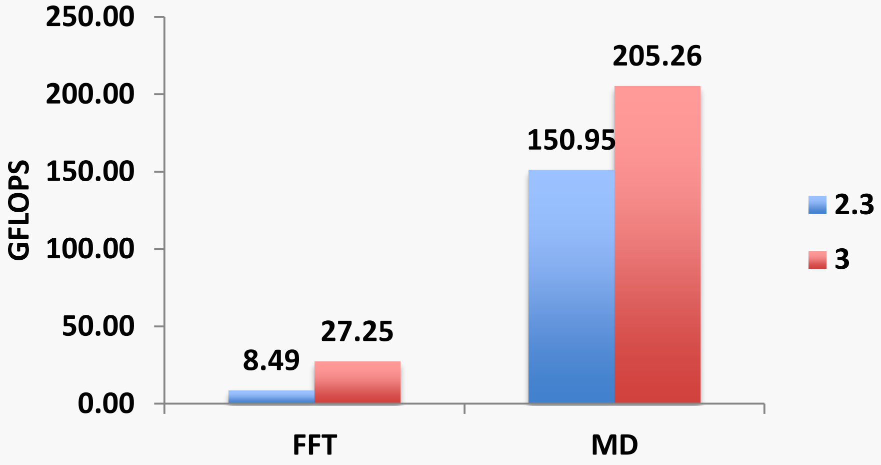
- Motivation
  - Extremely common scientific kernel
  - Bandwidth bound, and much harder to get performance than GEMM
- Basic design
  - 3 Algorithms, padded & unpadded data
  - CSR and ELLPACKR data formats
  - Supports random matrices or matrix market format
  - Example: Gould, Hu, & Scott: expanded system-3D PDE.



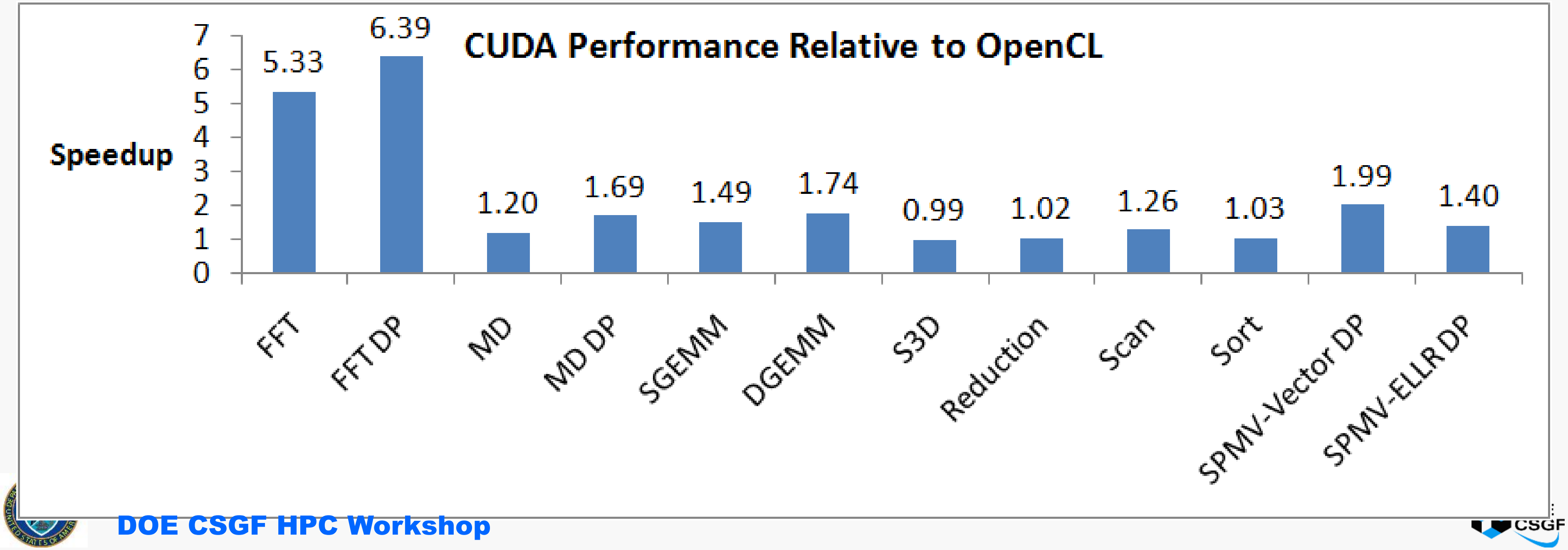
SpMV Performance



Comparing CUDA and OpenCL



Single precision, Tesla C1060 GPU
Comparing NVIDIA OpenCL implementation from 2.3 and 3.0 GPU Computing SDK



Questions for Exascale Programming Systems
Answers for current GPU systems

Q: How should we expose multiple levels of parallelism?

Explicit: MPI+threads+OpenCL/CUDA

Q: How should communication occur?

Mostly Explicit: MPI+DMA+SharedMem+Scratchpad

Q: Should thread-data locality/affinity be exposed to the user, or hidden managed by the runtime?

Explicit

Q: How should we best enabled domain specific libraries, frameworks, and languages?

?

Q: How do we maintain legacy applications and software?

Partially

MOVING BEYOND *EXPLICIT* PROGRAMMING OF GPUS

OpenMPC (OpenMP extended for CUDA)

- OpenMPC = OpenMP + a new set of directives and environment variables for CUDA
- OpenMPC provides
  - A high level abstraction of the CUDA programming model (Programmability)
  - An easy tuning environment to generate CUDA programs in many optimization variants (Tunability)

Seyong Lee and Rudolf Eigenmann, OpenMPC: Extended OpenMP Programming and Tuning for GPUs, SC10: Proceedings of the 2010 ACM/IEEE conference on Supercomputing (Best Student Paper Award), November 2010.

Seyong Lee, Seung-Jai Min, and Rudolf Eigenmann, OpenMP to GPGPU: A Compiler Framework for Automatic Translation and Optimization, Symposium on Principles and Practice of Parallel Programming (PPoPP09), February 2009

OpenMPC Approach

- Use OpenMP for easier programming on CUDA-based GPGPUs.
- Provide various compile-time optimizations for performance.
- Extend OpenMP to allow fine-grained control of CUDA-related parameters and optimizations.

OpenMPC: Directive Extension and Environment Variables

- OpenMPC Directive Format

  #pragma cuda gpurun [clause [,] clause]…]

  #pragma cuda cpurun [clause [,] clause]…]

  #pragma cuda nogpurun

  #pragma cuda ainfo procname(pname) kernelid(kID)

- OpenMPC Environment Variables
  - Control the program-level behavior of various optimizations or execution configurations for an output CUDA program.

OpenMPC Compilation System

- Overall Compilation Flow

Inpu
Ope
Ope
Pro

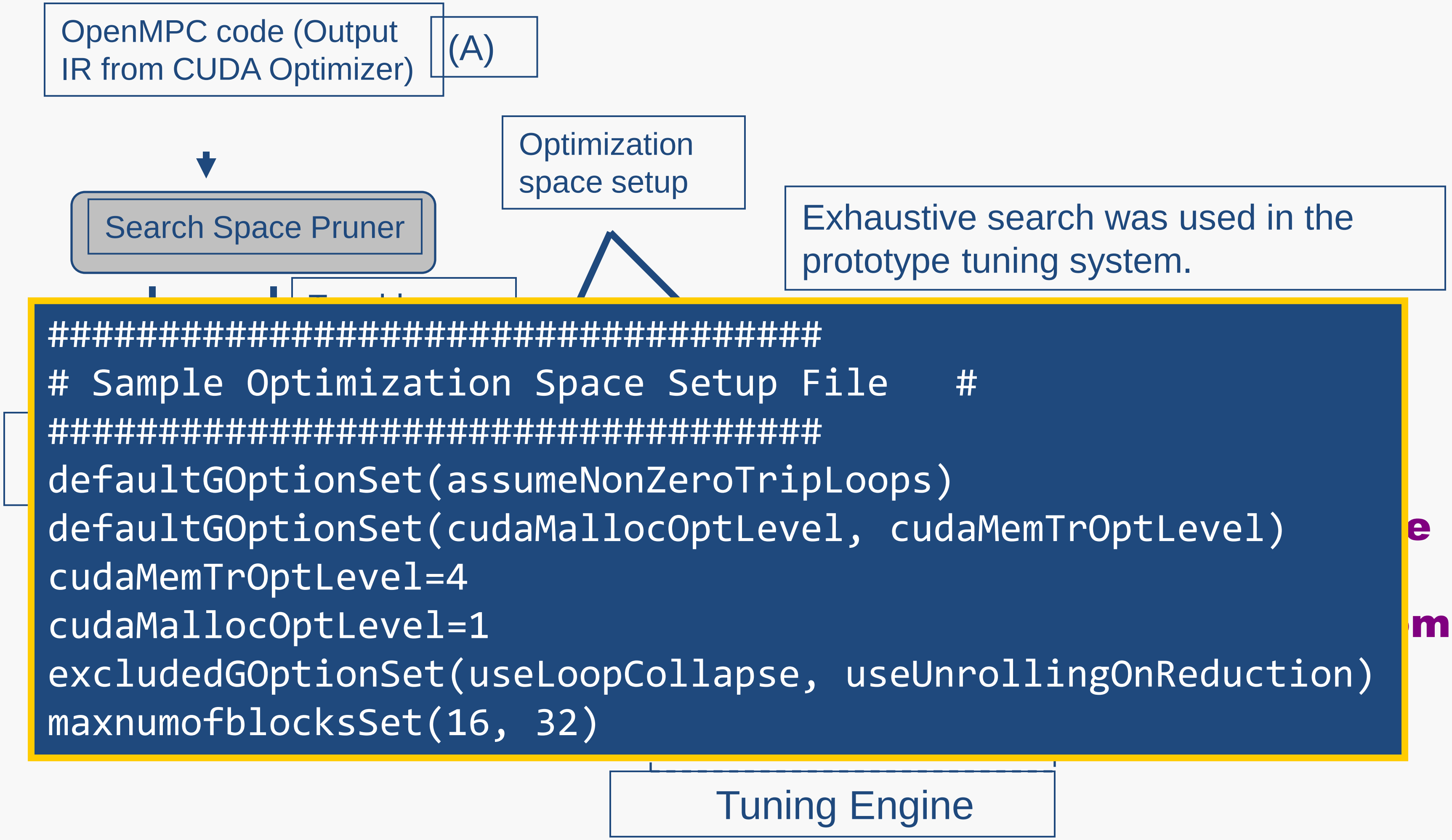
Out
CUI
Pro

ctive

For
O2C

```
#pragma omp parallel shared(firstcol, lastcol, x, z)
private(j) reduction(+: norm_temp11, norm_temp12)
#pragma cuda ainfo kernelid(1) procname(main)
#pragma cuda gpurun noc2gmemtr(x, z)
#pragma cuda gpurun nocudamalloc(x, z)
#pragma cuda gpurun nocudafree(firstcol, lastcol, x, z)
#pragma cuda gpurun nog2cmemtr(firstcol, lastcol, x, z)
#pragma cuda gpurun sharedRO(firstcol, lastcol)
#pragma cuda gpurun texture(z)
{
    #pragma omp for private(j) nowait
    for (j=1; j<=((lastcol-firstcol)+1); j ++ ) {
        norm_temp11=(norm_temp11+(x[j]*z[j]));
        norm_temp12=(norm_temp12+(z[j]*z[j]));
    }
}
```

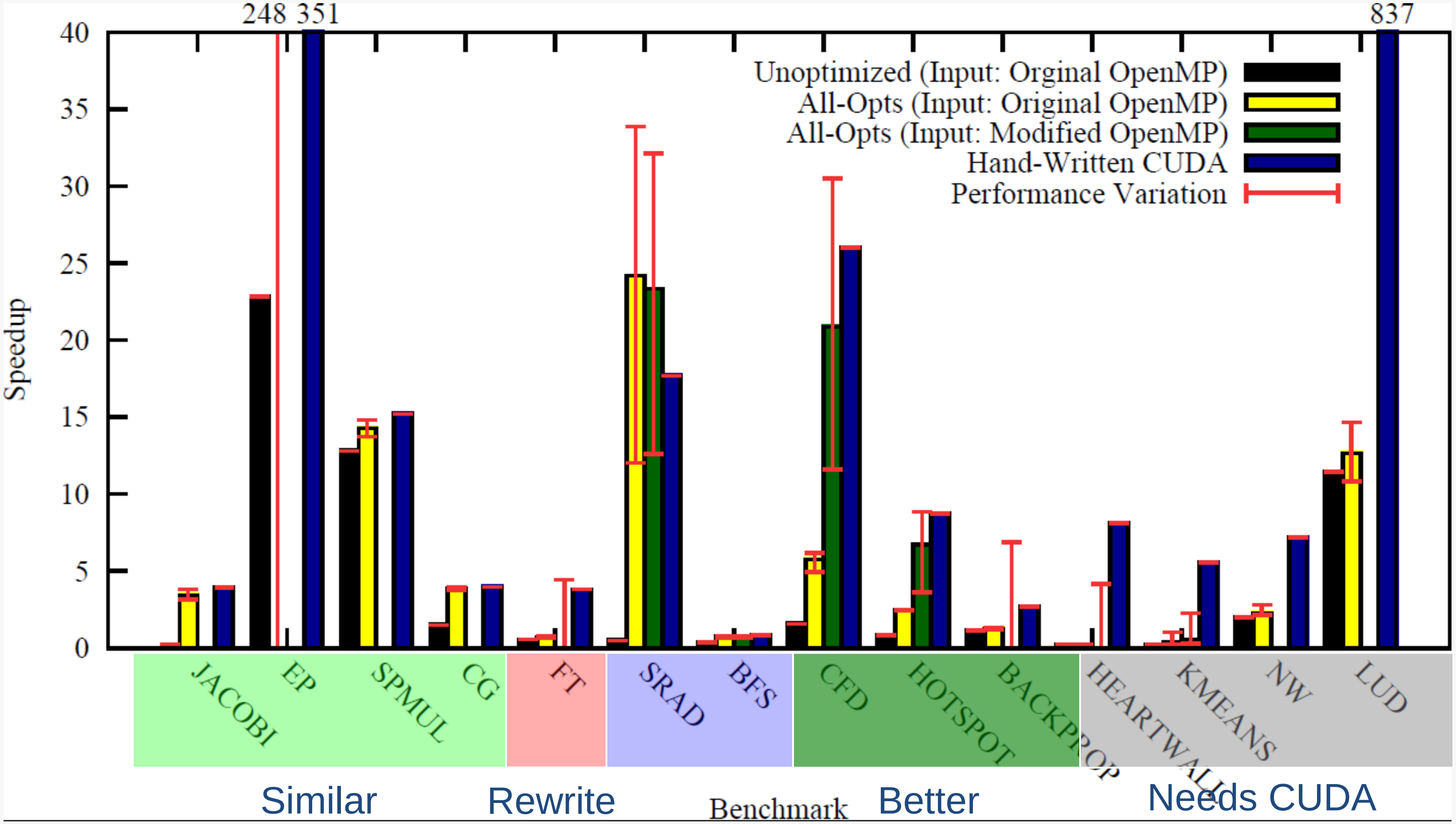
OpenMPC Tuning Framework



Exhaustive search was used in the prototype tuning system.

```
####################################
# Sample Optimization Space Setup File   #
####################################
defaultGOptionSet(assumeNonZeroTripLoops)
defaultGOptionSet(cudaMallocOptLevel, cudaMemTrOptLevel)
cudaMemTrOptLevel=4
cudaMallocOptLevel=1
excludedGOptionSet(useLoopCollapse, useUnrollingOnReduction)
maxnumofblocksSet(16, 32)
```

Performance of OpenMP Programs on CUDA



- Speedups are over serial on the CPU, when the largest available input data were used.
- Experimental Platform: CPU: two Dual-Core AMD Opteron at 3 GHz
  GPU: NVIDIA Quadro FX 5600 with 16 multiprocessors at 1.35GHz

Overall Tuning Performance

- Performance Summary

Translator Input	Performance Improvement over All-Opt Versions			Relative Performance over Manual Versions		
	MIN	MAX	AVG	MIN	MAX	AVG
Orig. OpenMP	1	4.23	1.19	0.02 (0.03)	1.92 (1.92)	0.5 (0.58)
Mod. OpenMP	1	7.71	1.24	0.02 (0.33)	2.68 (2.68)	0.75 (0.92)

In A(B) format, B refers the performance when the results of LUD are excluded.

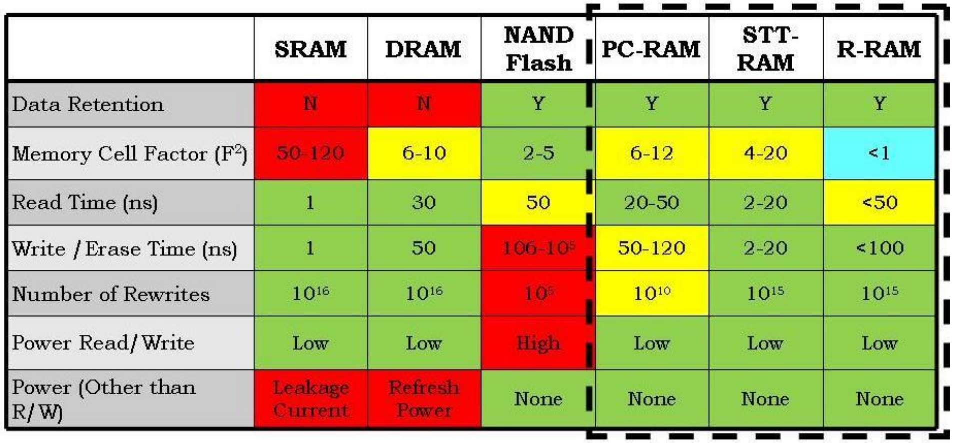
- Optimization Search Space Reduction
  - 98.7% on average for program-level tuning

#2: MEMORY BANDWIDTH AND CAPACITY

Blackcomb: Hardware-Software Co-design for Non-Volatile Memory in Exascale Systems

Jeffrey Vetter, ORNL
Robert Schreiber, HP Labs
Trevor Mudge, U Michigan
Yuan Xie, PSU

A comparison of various memory technologies



	SRAM	DRAM	NAND Flash	PC-RAM	STT-RAM	R-RAM
Data Retention	N	N	Y	Y	Y	Y
Memory Cell Factor (F^2)	50-120	6-10	2-5	6-12	4-20	<1
Read Time (ns)	1	30	50	20-50	2-20	<50
Write / Erase Time (ns)	1	50	106-10^5	50-120	2-20	<100
Number of Rewrites	10^{16}	10^{16}	10^5	10^{10}	10^{15}	10^{15}
Power Read/ Write	Low	Low	High	Low	Low	Low
Power (Other than R/ W)	Leakage Current	Refresh Power	None	None	None	None

Novel Ideas

- **New resilience-aware designs for non-volatile memory applications**
  - Mechanical-disk-based data-stores are completely replaced with energy-efficient non-volatile memories (NVM).
  - Most levels of the hierarchy, including DRAM and last levels of SRAM cache, are completely eliminated.
- **New energy-aware systems/applications for non-volatile memories (nanostores)**
  - Compute capacity, comprised of balanced low-power simple cores, is co-located with the data store.

Impact and Champions

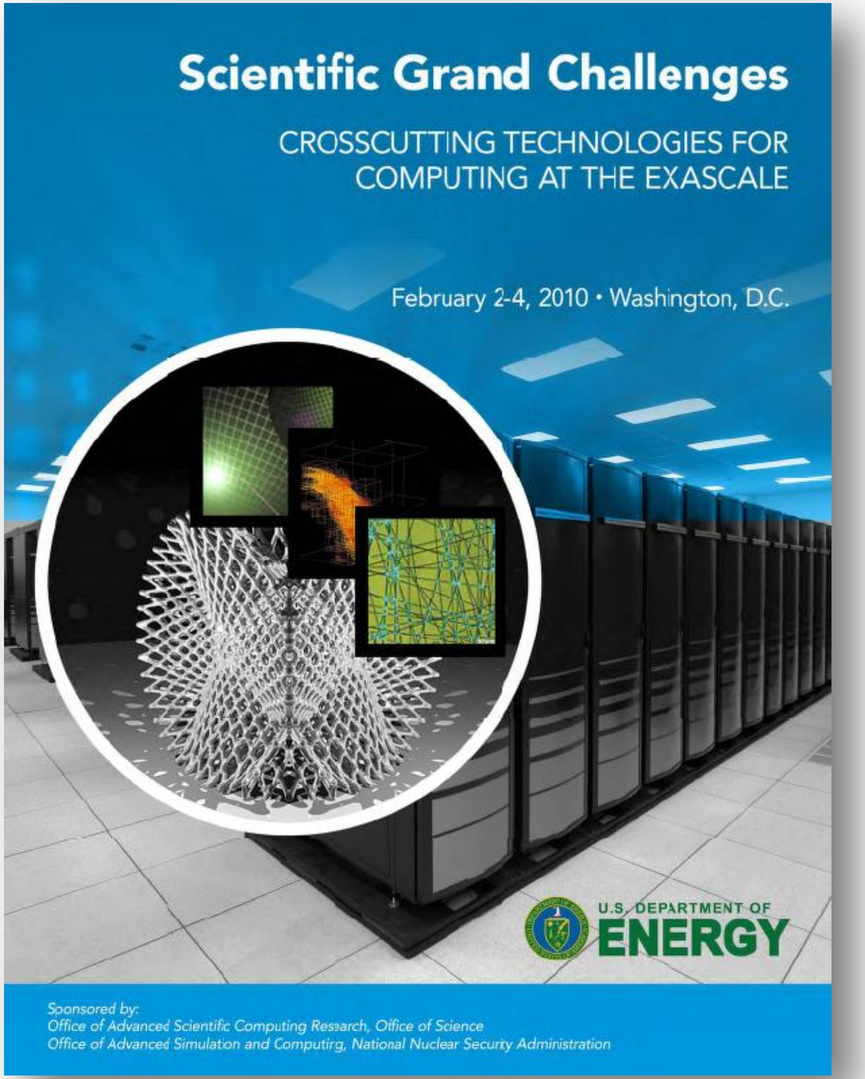
- **Reliance on NVM addresses device scalability, energy efficiency and reliability concerns associated with DRAM**
  - **More memory** – NVM scalability and density permits significantly more memory/core than projected by current Exascale estimates.
  - **Less power** – NVMs require zero stand-by power.
  - **More reliable** – alleviates increasing DRAM soft-error rate problem.
- **Node architecture with persistent storage near processing elements enables new computation paradigms**
  - Low-cost checkpoints, easing checkpoint frequency concerns.
  - Inter-process data sharing, easing in-situ analysis (UQ, Visualization)

Milestones

- Identify and evaluate the most promising non-volatile memory (NVM) device technologies.
- Explore assembly of NVM technologies into a storage and memory stack
- Build the abstractions and interfaces that allow software to exploit NVM to its best advantage
- Propose an exascale HPC system architecture that builds on our new memory architecture
- Characterize key DOE applications and investigate how they can benefit from these new technologies

Summary

- Exascale goals
  - Highlights from recent projections for exascale
- Challenges
  - Micro, macro power
  - Memory capacity and bandwidth
  - Parallelism
  - Programmability
- Programming systems play a critical role
  - Survey of programming systems
  - Solutions are coming now
    - — Heterogeneity with GPUs
  - Programming models need a vigorous ecosystem
    - — Tools, autotuning, libraries



Scientific Grand Challenges

CROSSCUTTING TECHNOLOGIES FOR
COMPUTING AT THE EXASCALE

February 2-4, 2010 • Washington, D.C.

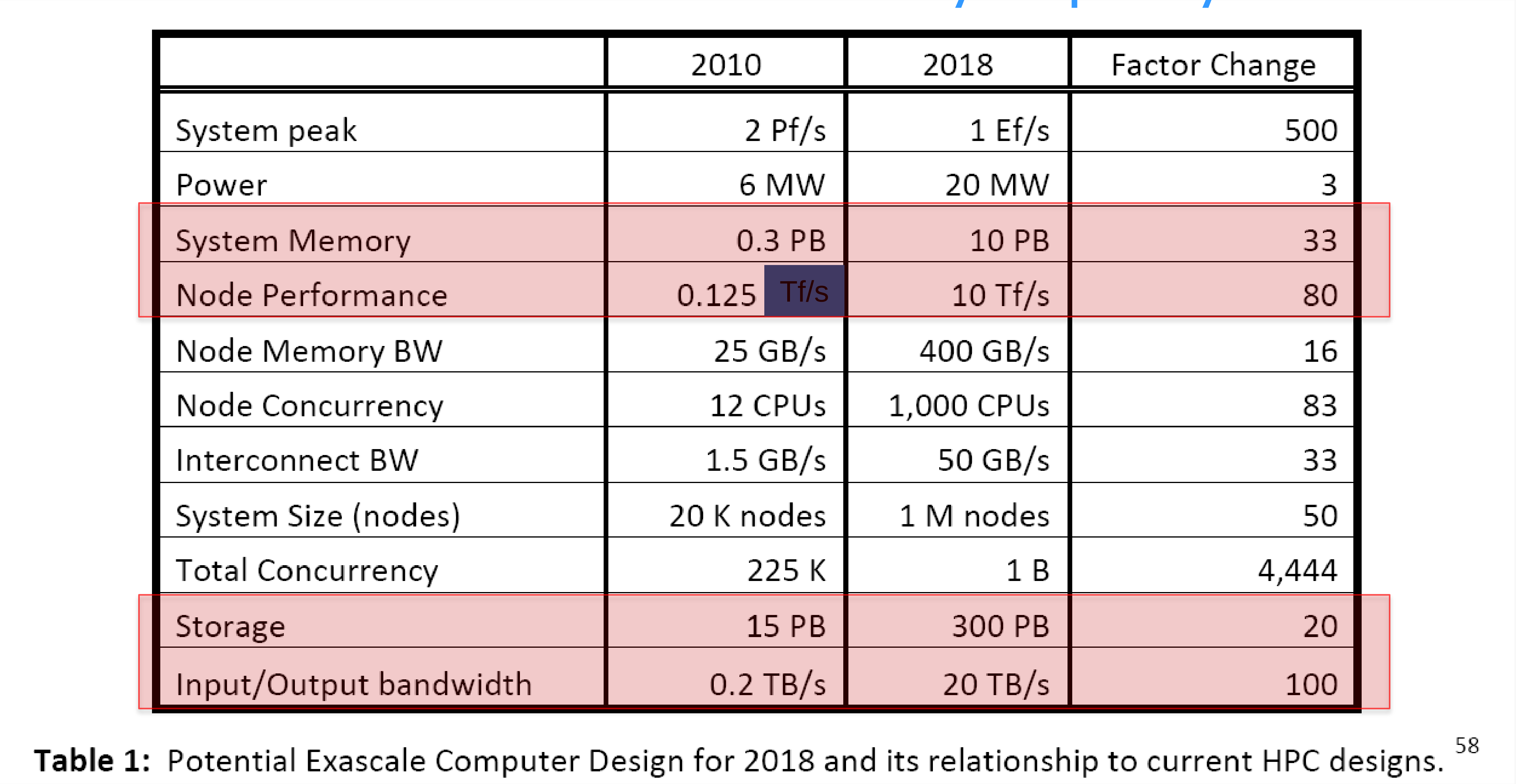
U.S. DEPARTMENT OF
ENERGY

Sponsored by:
Office of Advanced Scientific Computing Research, Office of Science
Office of Advanced Simulation and Computing, National Nuclear Security Administration

http://science.energy.gov/ascr/news-and-resources/workshops-and-conferences/grand-challenges/

BONUS SLIDES

Critical Concern : Memory Capacity



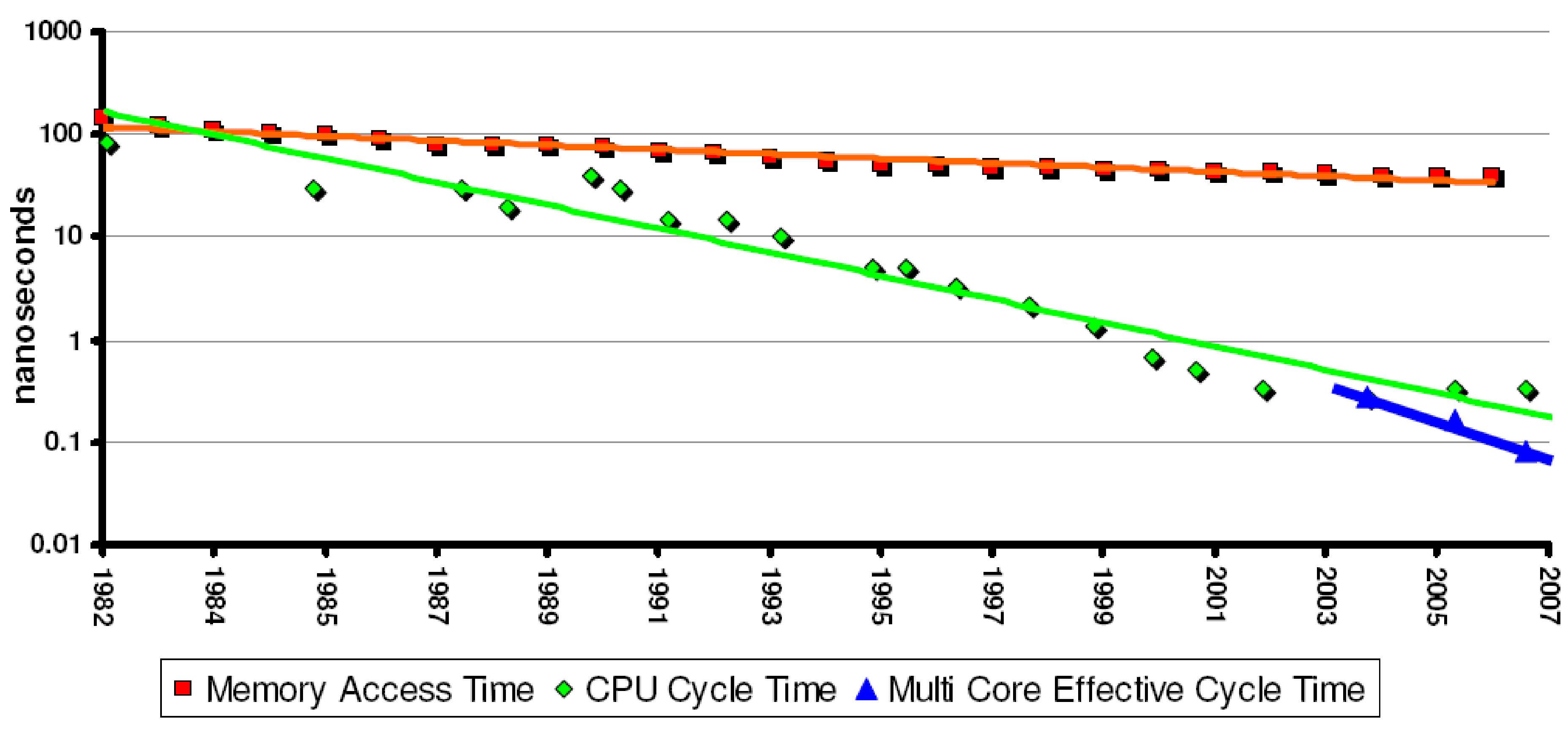
	2010	2018	Factor Change
System peak	2 Pf/s	1 Ef/s	500
Power	6 MW	20 MW	3
System Memory	0.3 PB	10 PB	33
Node Performance	0.125 Tf/s	10 Tf/s	80
Node Memory BW	25 GB/s	400 GB/s	16
Node Concurrency	12 CPUs	1,000 CPUs	83
Interconnect BW	1.5 GB/s	50 GB/s	33
System Size (nodes)	20 K nodes	1 M nodes	50
Total Concurrency	225 K	1 B	4,444
Storage	15 PB	300 PB	20
Input/Output bandwidth	0.2 TB/s	20 TB/s	100

Table 1: Potential Exascale Computer Design for 2018 and its relationship to current HPC designs.[58]

- Small memory capacity has profound impact on other features

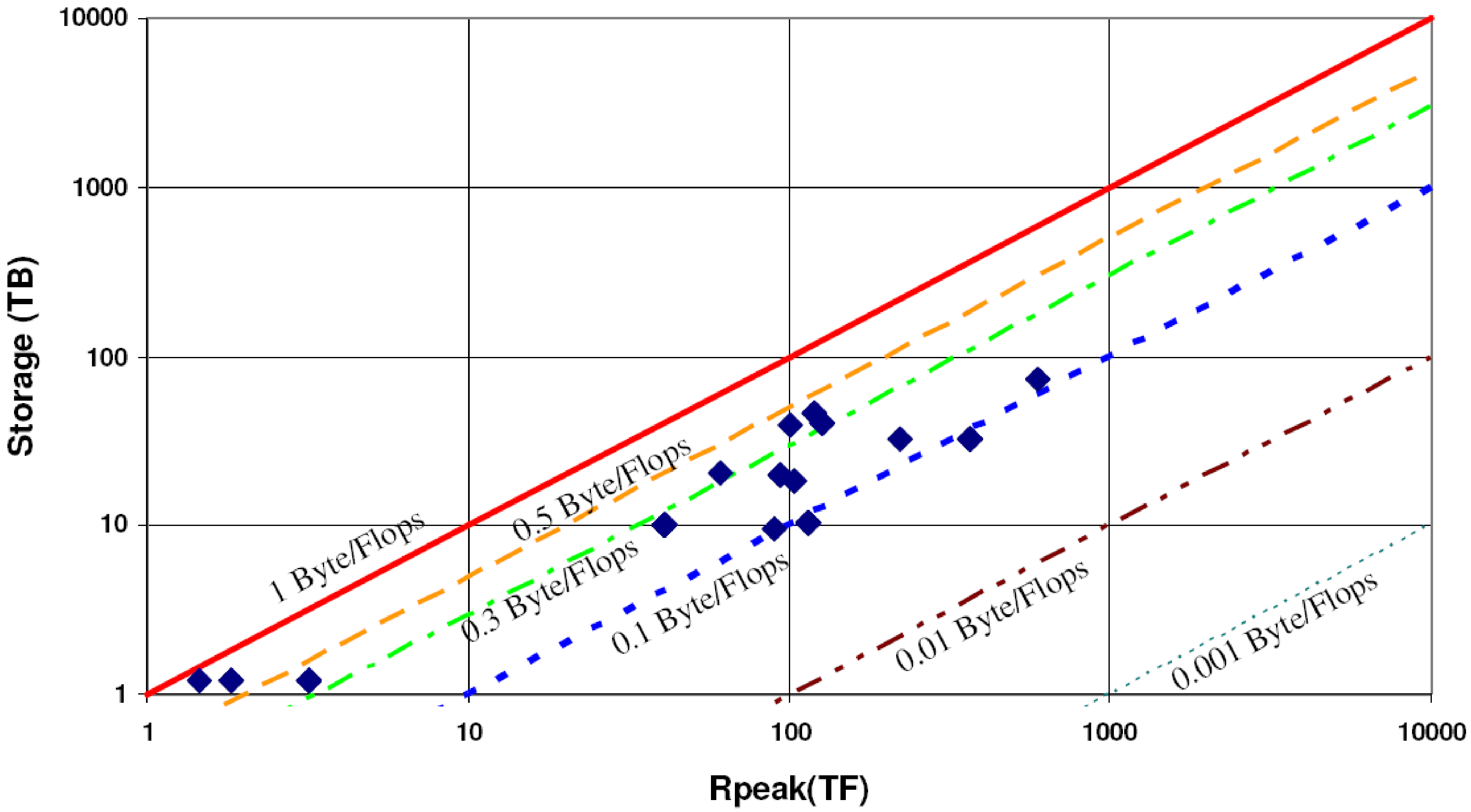
- Feeding the core
- Poor efficiencies
- Small messages, I/O

Memory Performance



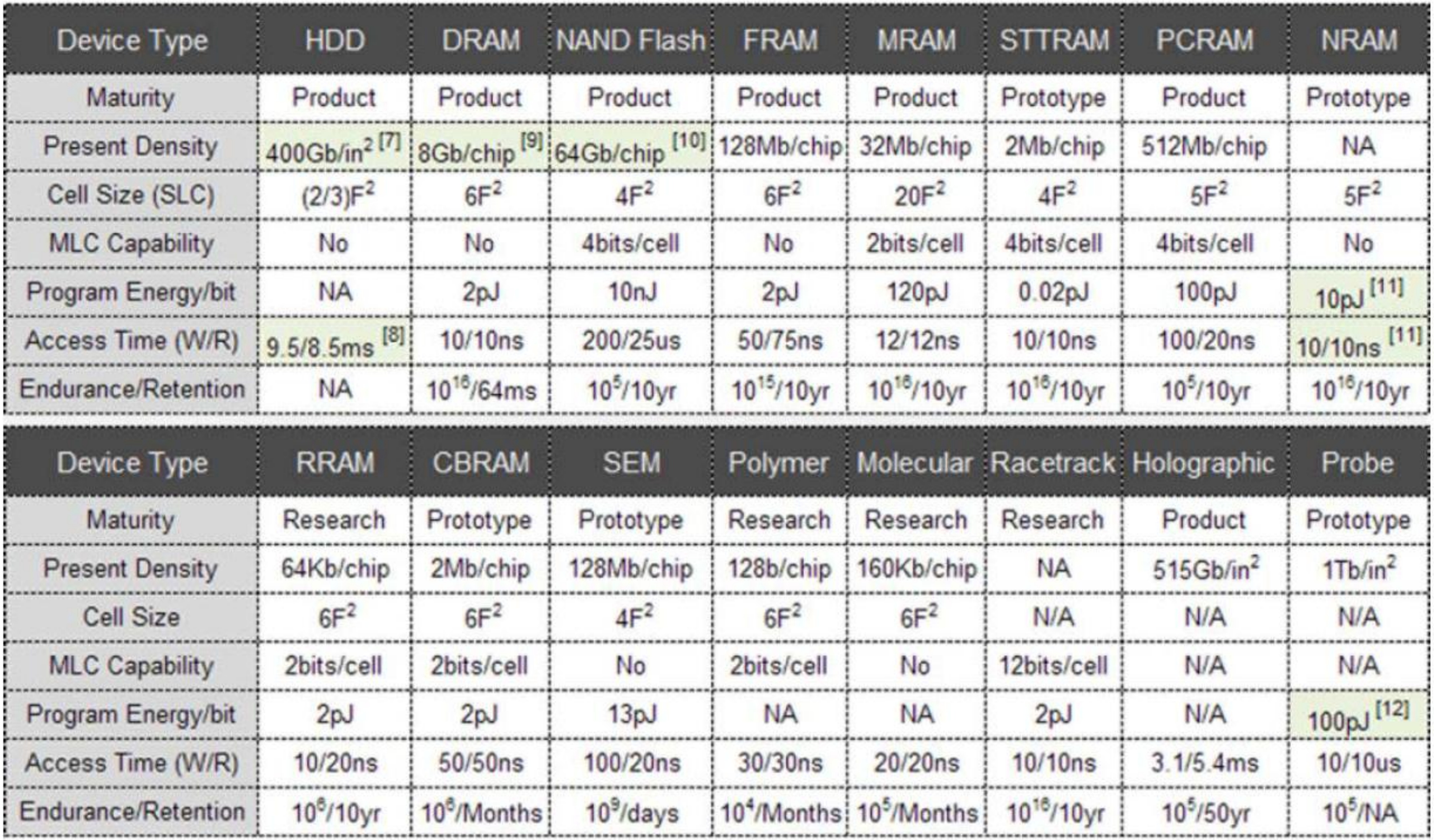
Source: DARPA Exascale Computing Study

Memory Capacity



Source: DARPA Exascale Computing Study

New Technologies Offer Promise



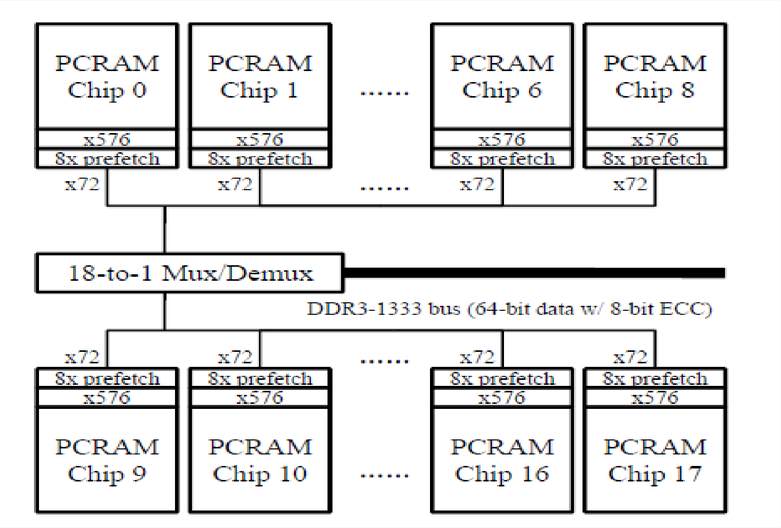
Device Type	HDD	DRAM	NAND Flash	FRAM	MRAM	STTRAM	PCRAM	NRAM
Maturity	Product	Product	Product	Product	Product	Prototype	Product	Prototype
Present Density	400Gb/in^2 [7]	8Gb/chip [9]	64Gb/chip [10]	128Mb/chip	32Mb/chip	2Mb/chip	512Mb/chip	NA
Cell Size (SLC)	(2/3)F^2	6F^2	4F^2	6F^2	20F^2	4F^2	5F^2	5F^2
MLC Capability	No	No	4bits/cell	No	2bits/cell	4bits/cell	4bits/cell	No
Program Energy/bit	NA	2pJ	10nJ	2pJ	120pJ	0.02pJ	100pJ	10pJ [11]
Access Time (W/R)	9.5/8.5ms [8]	10/10ns	200/25us	50/75ns	12/12ns	10/10ns	100/20ns	10/10ns [11]
Endurance/Retention	NA	10^{16}/64ms	10^5/10yr	10^{15}/10yr	10^{16}/10yr	10^{16}/10yr	10^5/10yr	10^{16}/10yr

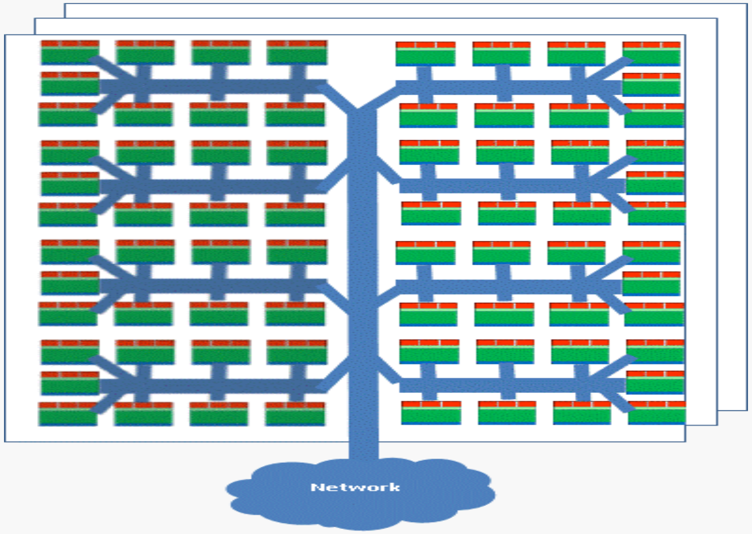
Device Type	RRAM	CBRAM	SEM	Polymer	Molecular	Racetrack	Holographic	Probe
Maturity	Research	Prototype	Prototype	Research	Research	Research	Product	Prototype
Present Density	64Kb/chip	2Mb/chip	128Mb/chip	128b/chip	160Kb/chip	NA	515Gb/in^2	1Tb/in^2
Cell Size	6F^2	6F^2	4F^2	6F^2	6F^2	N/A	N/A	N/A
MLC Capability	2bits/cell	2bits/cell	No	2bits/cell	No	12bits/cell	N/A	N/A
Program Energy/bit	2pJ	2pJ	13pJ	NA	NA	2pJ	N/A	100pJ [12]
Access Time (W/R)	10/20ns	50/50ns	100/20ns	30/30ns	20/20ns	10/10ns	3.1/5.4ms	10/10us
Endurance/Retention	10^6/10yr	10^6/Months	10^9/days	10^4/Months	10^5/Months	10^{16}/10yr	10^5/50yr	10^5/NA

M.H. Kryder et al., “After Hard Drives,” IEEE Trans. Mag.,

Opportunities go far beyond a plugin replacement for disk drives…

- New distributed computer architectures that address exascale resilience, energy, and performance requirements
    - replace mechanical-disk-based data-stores with energy-efficient non-volatile memories
    - explore opportunities for NVM memory, from plug-compatible replacement (like the NV DIMM, below) to radical, new data-centric compute hierarchy (nanostores)
    - place low power compute cores close to the data store
    - reduce number of levels in the memory hierarchy
- Adapt existing software systems to exploit this new capabilities

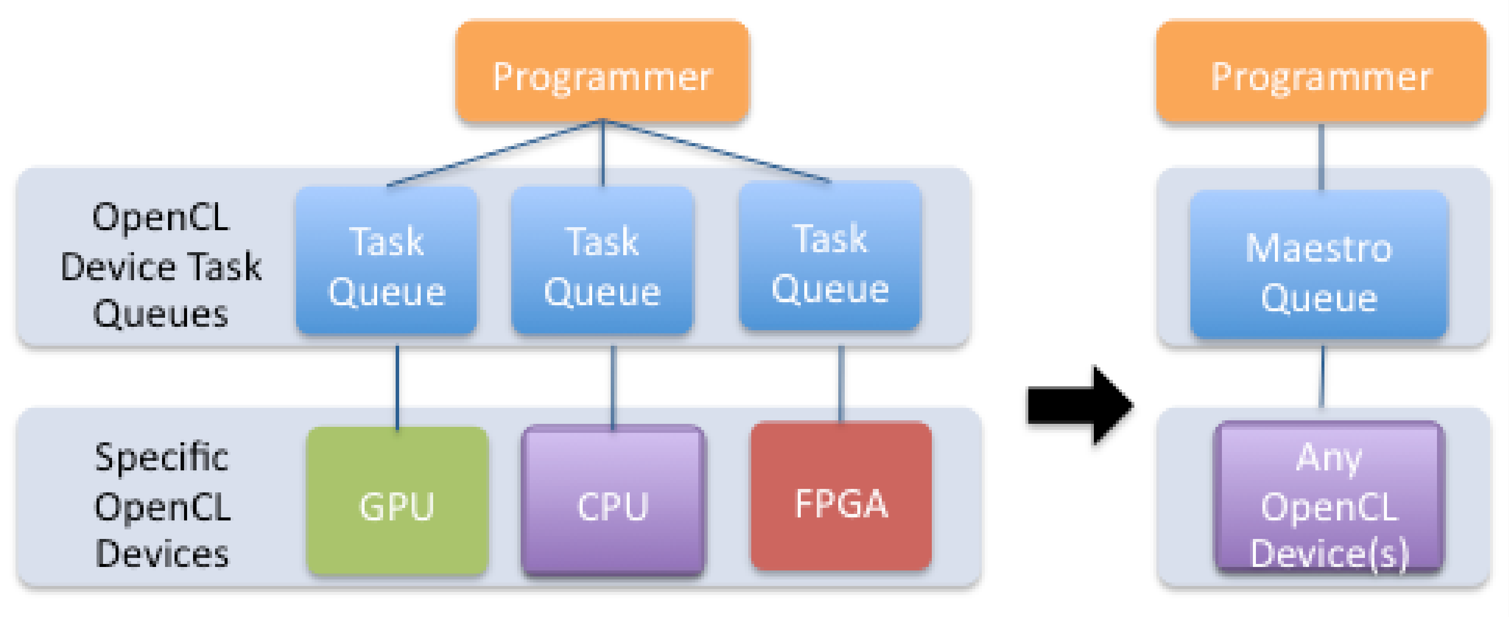




Providing Performance Portability

AUTOTUNING

Maestro



- Portability
- Load balancing
- Autotuning

K. Spafford, J. Meredith, and J. Vetter, “Maestro: Data Orchestration and Tuning for OpenCL Devices,” in *Euro-Par 2010 - Parallel Processing, vol. 6272, Lecture Notes in Computer Science, P. D’Ambra, M. Guarracino et al., Eds.: Springer Berlin / Heidelberg, 2010, pp. 275-86.*

Maestro: Multibuffering

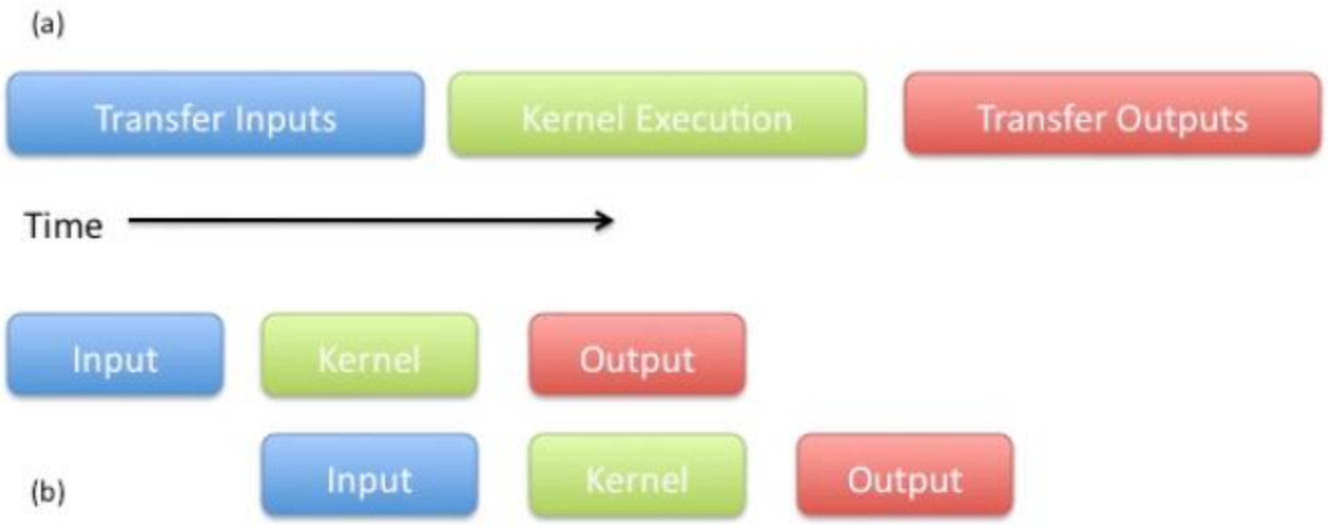


Fig. 2. Double Buffering–This figure contrasts the difference between (a) the function offload model and (b) a very simple case of double buffering. Devices which can concurrently execute kernels and transfer data are able to hide some communication time with computation.

Maestro : Autotuning Workgroups

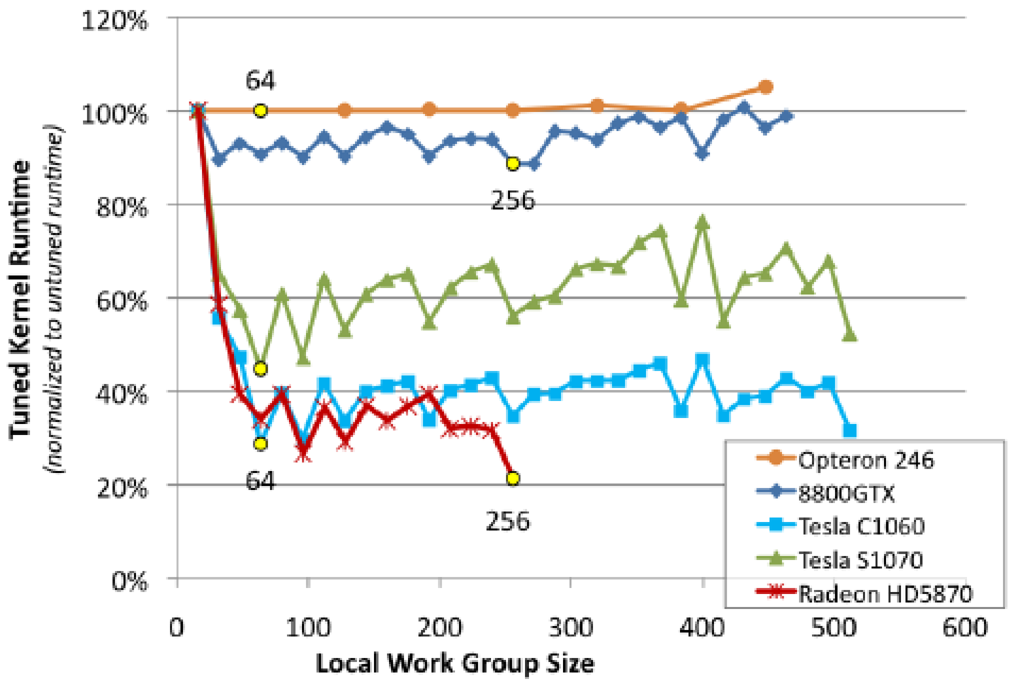


Fig. 3. Autotuning the local work group size – This figure shows the performance of the MD kernel on various platforms at different local work group sizes, normalized to the performance at a group size of 16. Lower runtimes are better.

Combined Autotuning Results

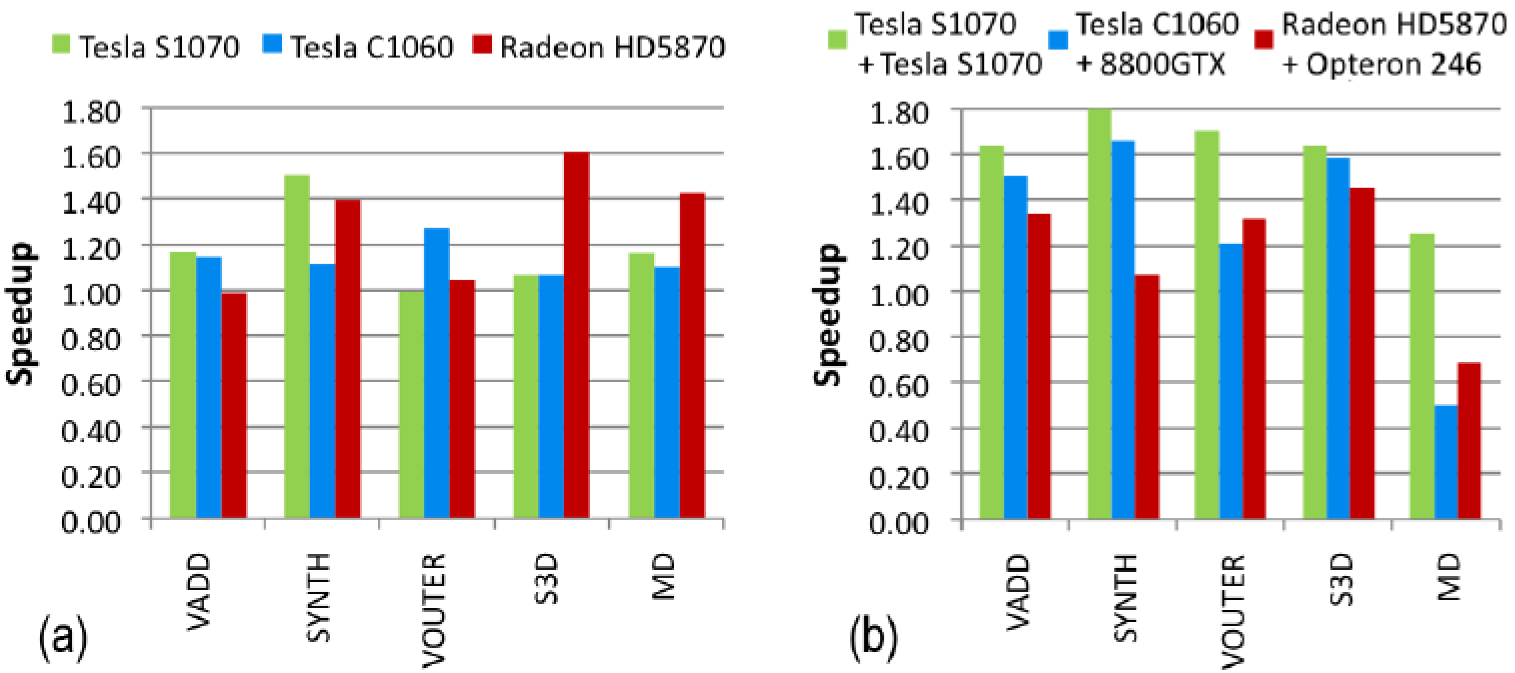
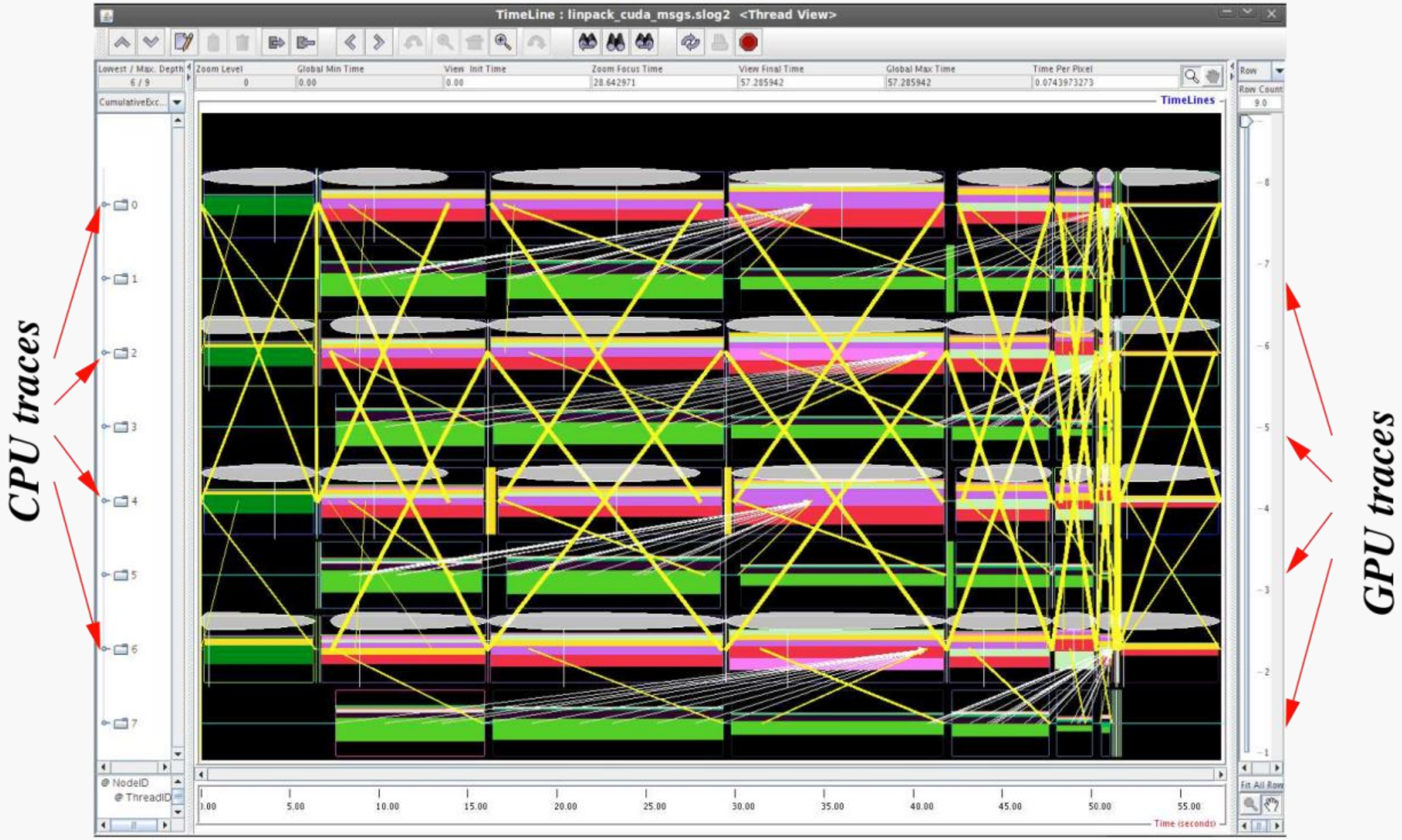


Fig. 6. Combined autotuning results – (a) Shows the combined benefit of autotuning both the local work group size the double buffering chunk size for a single GPU of the test platforms. (b) Shows the combined benefit of autotuning both the local work group size and the multi-GPU load imbalance using both devices (GPU+GPU or GPU+CPU) of the test platforms. Longer bars are better.

PERFORMANCE AND CORRECTNESS

Vancouver: Integrated Performance Analysis of MPI/GPU Applications



TimeLine : linpack_cuda_msgs.slog2 <Thread View>

Lowest / Max. Depth: 6 / 9
Zoom Level: 0
Global Min Time: 0.00
View Init Time: 0.00
Zoom Focus Time: 28.642971
View Final Time: 57.285942
Global Max Time: 57.285942
Time Per Pixel: 0.0743973273
Row Count: 9.0

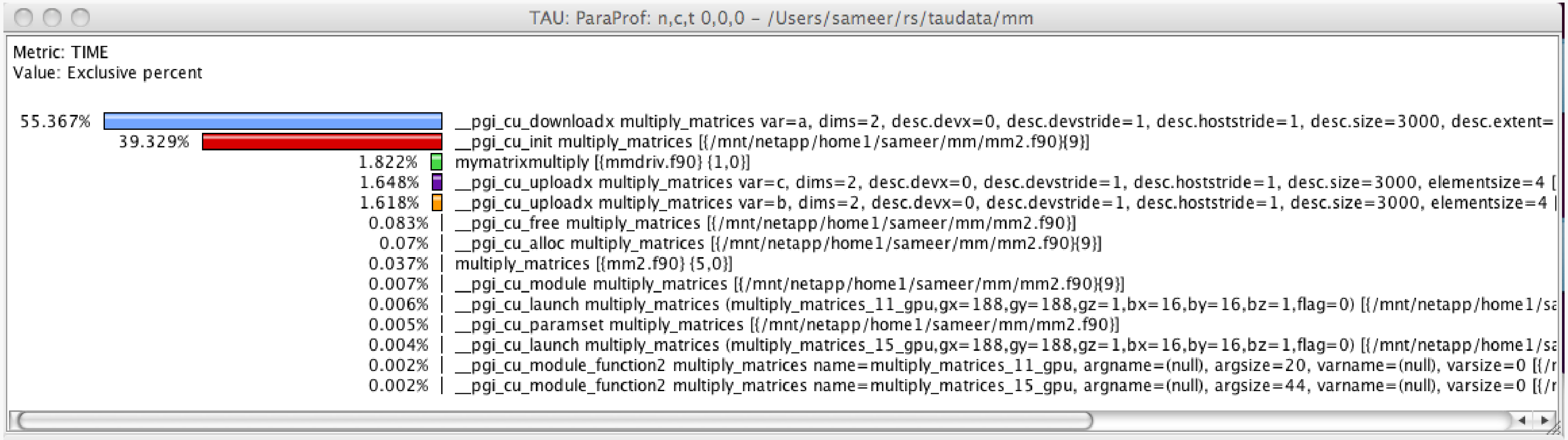
CPU traces

GPU traces

CUDA memory transfer (white)

MPI communication (yellow)

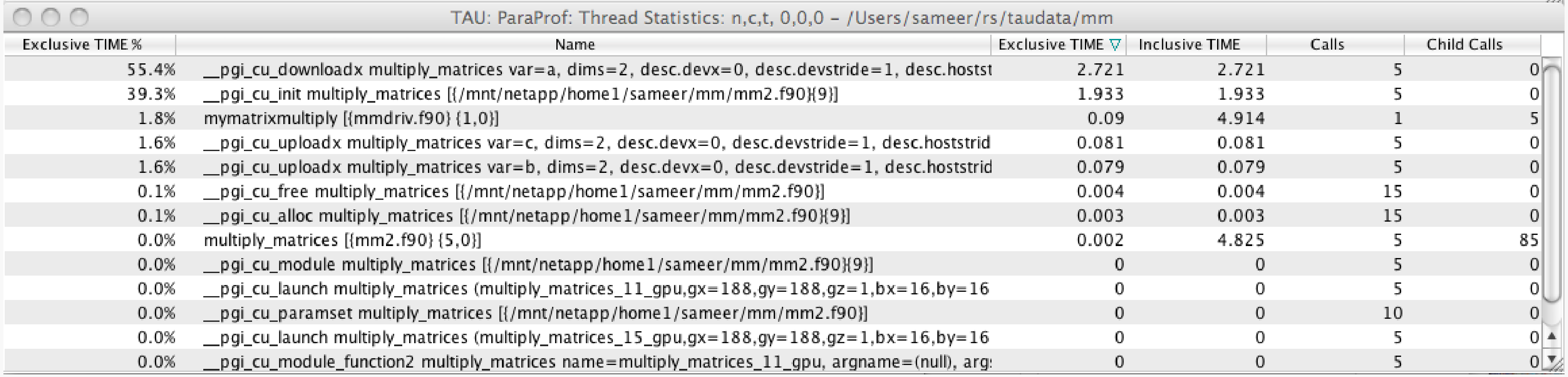
Vancouver: Integrated Performance Analysis of Compiler CUDA Generated Applications



TAU: ParaProf: n,c,t 0,0,0 – /Users/sameer/rs/taudata/mm

Metric: TIME
Value: Exclusive percent

```
55.367%  __pgi_cu_downloadx multiply_matrices var=a, dims=2, desc.devx=0, desc.devstride=1, desc.hoststride=1, desc.size=3000, desc.extent=
39.329%  __pgi_cu_init multiply_matrices [{/mnt/netapp/home1/sameer/mm/mm2.f90}{9}]
 1.822%  mymatrixmultiply [{mmdriv.f90} {1,0}]
 1.648%  __pgi_cu_uploadx multiply_matrices var=c, dims=2, desc.devx=0, desc.devstride=1, desc.hoststride=1, desc.size=3000, elementsize=4 [
 1.618%  __pgi_cu_uploadx multiply_matrices var=b, dims=2, desc.devx=0, desc.devstride=1, desc.hoststride=1, desc.size=3000, elementsize=4 |
 0.083%  __pgi_cu_free multiply_matrices [{/mnt/netapp/home1/sameer/mm/mm2.f90}]
  0.07%  __pgi_cu_alloc multiply_matrices [{/mnt/netapp/home1/sameer/mm/mm2.f90}{9}]
 0.037%  multiply_matrices [{mm2.f90} {5,0}]
 0.007%  __pgi_cu_module multiply_matrices [{/mnt/netapp/home1/sameer/mm/mm2.f90}{9}]
 0.006%  __pgi_cu_launch multiply_matrices (multiply_matrices_11_gpu,gx=188,gy=188,gz=1,bx=16,by=16,bz=1,flag=0) [{/mnt/netapp/home1/sa
 0.005%  __pgi_cu_paramset multiply_matrices [{/mnt/netapp/home1/sameer/mm/mm2.f90}]
 0.004%  __pgi_cu_launch multiply_matrices (multiply_matrices_15_gpu,gx=188,gy=188,gz=1,bx=16,by=16,bz=1,flag=0) [{/mnt/netapp/home1/sa
 0.002%  __pgi_cu_module_function2 multiply_matrices name=multiply_matrices_11_gpu, argname=(null), argsize=20, varname=(null), varsize=0 [{/r
 0.002%  __pgi_cu_module_function2 multiply_matrices name=multiply_matrices_15_gpu, argname=(null), argsize=44, varname=(null), varsize=0 [{/r
```



TAU: ParaProf: Thread Statistics: n,c,t, 0,0,0 – /Users/sameer/rs/taudata/mm

Exclusive TIME %	Name	Exclusive TIME ▽	Inclusive TIME	Calls	Child Calls
55.4%	__pgi_cu_downloadx multiply_matrices var=a, dims=2, desc.devx=0, desc.devstride=1, desc.hostst	2.721	2.721	5	0
39.3%	__pgi_cu_init multiply_matrices [{/mnt/netapp/home1/sameer/mm/mm2.f90}{9}]	1.933	1.933	5	0
1.8%	mymatrixmultiply [{mmdriv.f90} {1,0}]	0.09	4.914	1	5
1.6%	__pgi_cu_uploadx multiply_matrices var=c, dims=2, desc.devx=0, desc.devstride=1, desc.hoststrid	0.081	0.081	5	0
1.6%	__pgi_cu_uploadx multiply_matrices var=b, dims=2, desc.devx=0, desc.devstride=1, desc.hoststrid	0.079	0.079	5	0
0.1%	__pgi_cu_free multiply_matrices [{/mnt/netapp/home1/sameer/mm/mm2.f90}]	0.004	0.004	15	0
0.1%	__pgi_cu_alloc multiply_matrices [{/mnt/netapp/home1/sameer/mm/mm2.f90}{9}]	0.003	0.003	15	0
0.0%	multiply_matrices [{mm2.f90} {5,0}]	0.002	4.825	5	85
0.0%	__pgi_cu_module multiply_matrices [{/mnt/netapp/home1/sameer/mm/mm2.f90}{9}]	0	0	5	0
0.0%	__pgi_cu_launch multiply_matrices (multiply_matrices_11_gpu,gx=188,gy=188,gz=1,bx=16,by=16	0	0	5	0
0.0%	__pgi_cu_paramset multiply_matrices [{/mnt/netapp/home1/sameer/mm/mm2.f90}]	0	0	10	0
0.0%	__pgi_cu_launch multiply_matrices (multiply_matrices_15_gpu,gx=188,gy=188,gz=1,bx=16,by=16	0	0	5	0
0.0%	__pgi_cu_module_function2 multiply_matrices name=multiply_matrices_11_gpu, argname=(null), arg:	0	0	5	0

Partners: U of Oregon Tau Group, PGI