High Performance Computing with Application Accelerators

Volodymyr Kindratenko

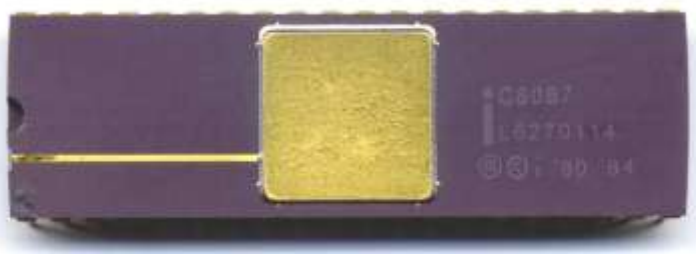
Innovative Systems Laboratory @ NCSA

Institute for Advanced Computing Applications and Technologies (IACAT)

National Center for Supercomputing Applications
University of Illinois at Urbana-Champaign

A Bit of History

- ***Application accelerator* is not a new concept**
  - e.g., Intel C8087 Math Coprocessor



- **But it received a renewed interest around 2002 due to the single thread performance stall**
  - Frequency scaling became unsustainable with smaller IC feature sizes
  - Instruction-level parallelism (IPL) can go only so far

Accelerators from early 2000s

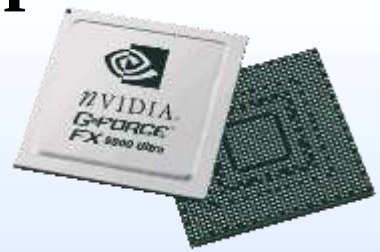
- **Sony Emotion Engine**
  - Designed for Sony PS2 game console
  - PS2 compute clusters, such as one at NCSA

- **Field-Programmable Gate Arrays (FPGAs)**
  - In early 2000s reached size of millions of gates
  - The field of reconfigurable computing emerged
  - FPGA performance growth trends pointed towards outperforming CPUs

ALTERA

XILINX

- **Graphics Processing Units (GPUs) used for General-Purpose computing (GPGPUs)**
  - Programmable shaders



Mainstream Accelerators from mid-2000s

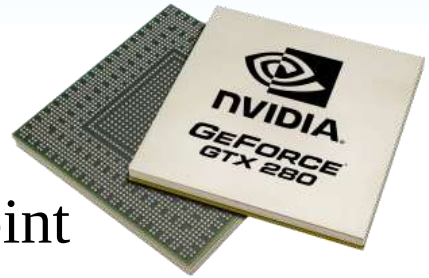
- **FPGAs**
  - Finally large enough for running useful floating point calculations
  - Many vendors offer a variety of products for reconfigurable computing

- **Sony/Toshiba/IBM Cell Broadband Engine**
  - Designed for Sony PS3 game console
  - Delivers over 200 GFLOPS in single-precision (SP)

- **ClearSpeed**
  - First floating-point accelerator specifically made for high-performance computing
  - 25 GFLOPS peak @ 10 Watt power

Application Accelerators Today

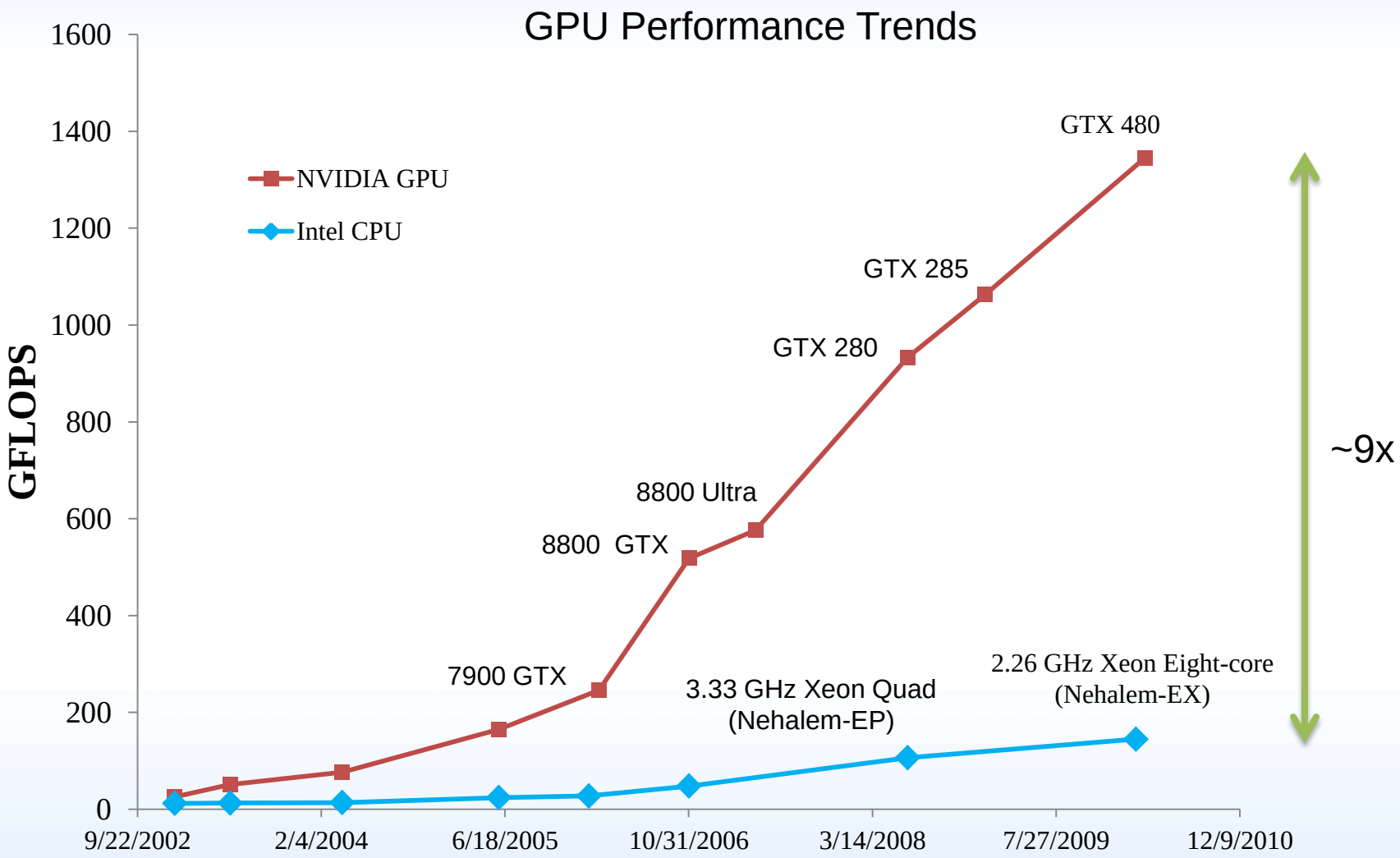
- **GPUs**
  - Dominated by NVIDIA
  - Offer unsurpassed performance for floating-point applications



- **IBM PowerXCell 8i**
  - Cell/B.E. variant with improved double-precision floating-point support
  - Used in RoadRunner supercomputer

- **FPGAs**
  - CPU socket plug-ins and PCI-based accelerator boards offered by a number of vendors

Why Application Accelerators

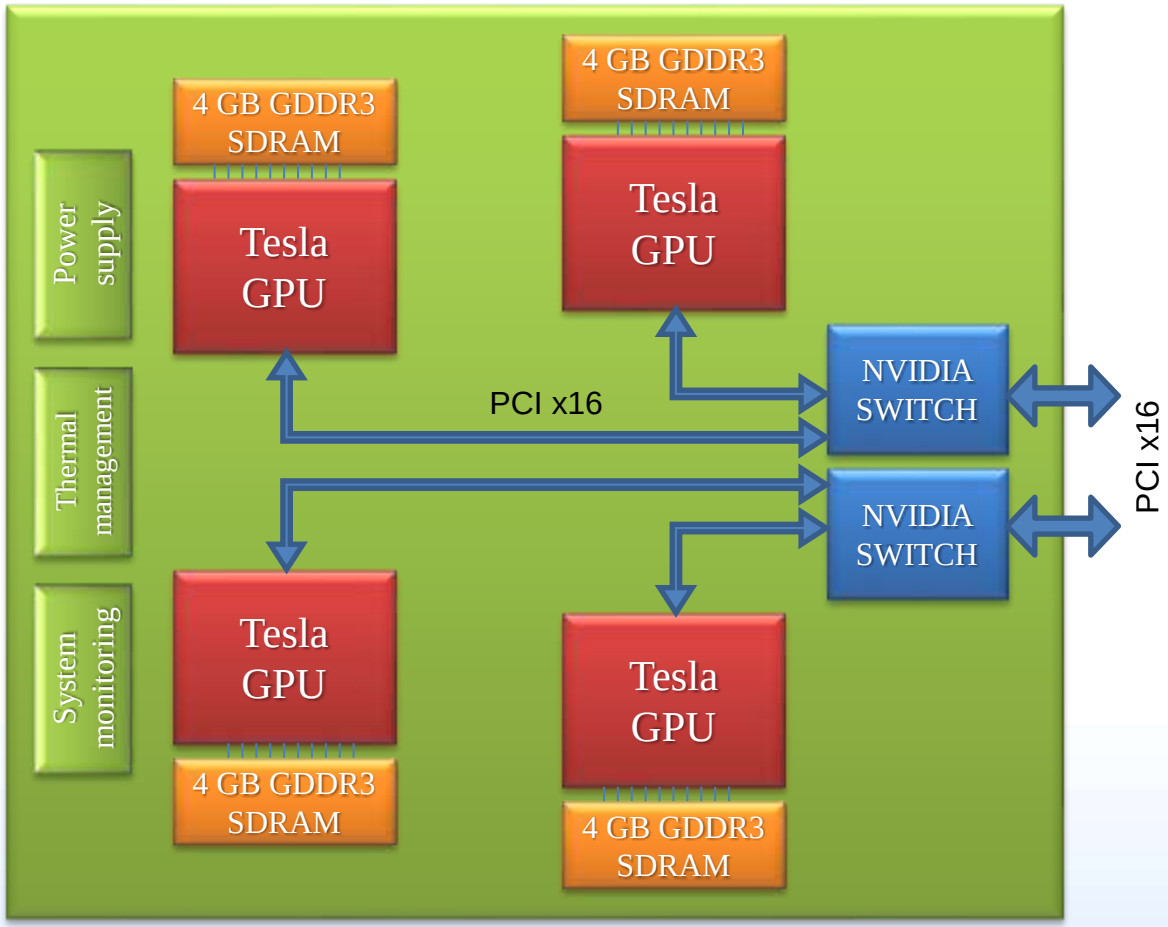


Major HPC systems with accelerators

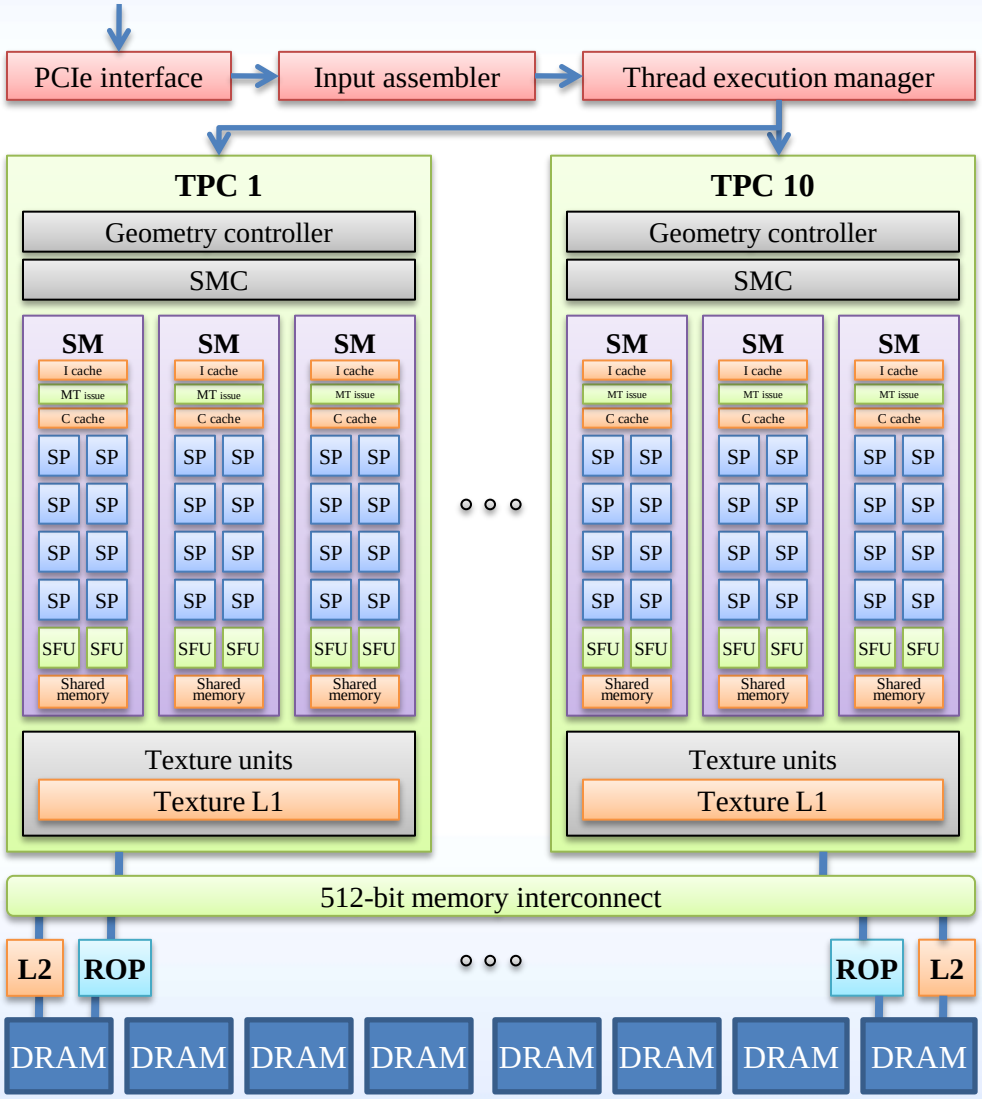
- **Nebulae**
  - NVIDIA Tesla C2050 GPU
  - #2 on June 2010 TOP-500 list
  - ~2.8 PFLOPS peak, ~1.2 PFLOPS Linpack
  - National Supercomputing Centre in Shenzhen, China
- **RoadRunner**
  - PowerXCell 8i
  - #3 on June 2010 TOP-500 list
  - ~1.3 PFLOPS peak, ~1 PFLOPS Linpack
  - Los Alamos National Lab, USA
- **Tianhe-1**
  - ATI Radeon HD 4870 GPU
  - #7 on June 2010 TOP-500 list
  - ~1.2 PFLOPS peak, ~0.5 PFLOPS Linpack
  - Chinese National University of Defense Technology, China

NVIDIA Tesla S1070 GPU Computing Server

- **4 T10 GPUs**



NVIDIA Tesla GPU Architecture

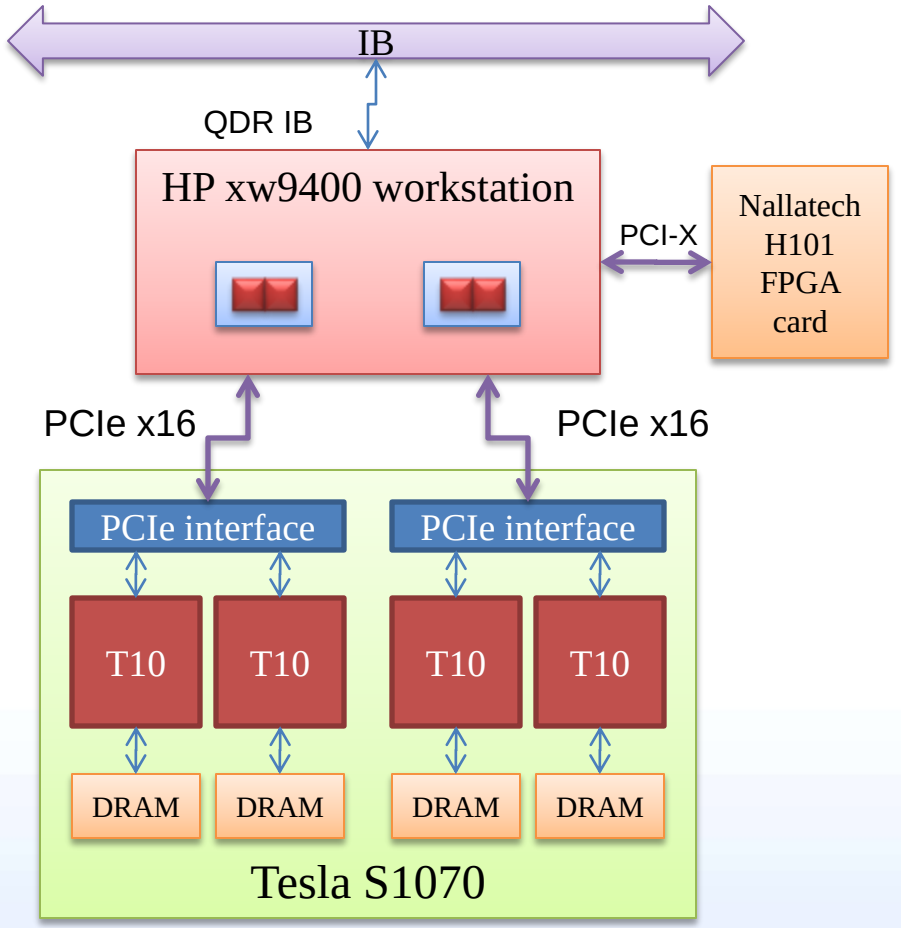


- 240 streaming processors arranged as 30 streaming multiprocessors
- At 1.3 GHz this provides
  - 1 TFLOPS SP
  - 86.4 TFLOPS DP
- 512-bit interface to off-chip GDDR3 memory
  - 102 GB/s bandwidth

NCSA *AC* Cluster

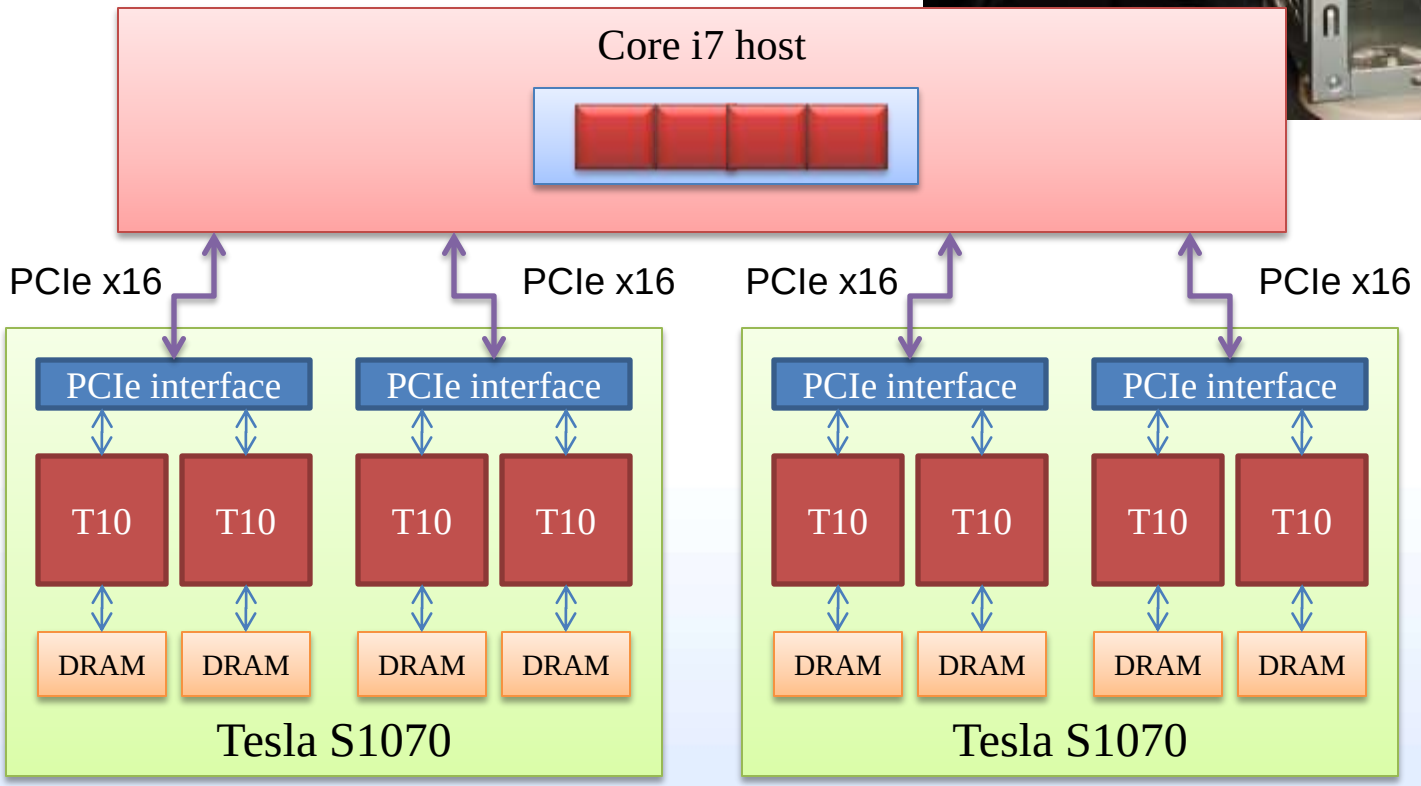
AC01-32 nodes

- **HP xw9400 workstation**
  - 2216 AMD Opteron 2.4 GHz dual socket dual core
  - 8GB DDR2 in ac04-ac32
  - 16GB DDR2 in ac01-03
  - PCIe gen 1.0
  - Infiniband QDR
- **Tesla S1070 1U GPU Computing Server**
  - 1.3 GHz Tesla T10 processors
  - 4x4 GB GDDR3 SDRAM
  - 1 per host



AC33 node

- CPU cores (Intel core i7): 8
- Accelerator Units (S1070): 2
- Total GPUs: 8
- Host Memory: 24 GB DDR3
- GPU Memory: 32 GB
- CPU cores/GPU ratio: 1
- PCIe gen 2.0
- Dual IOH (72 lanes PCIe)



AC34-AC41 nodes (New)

- **Supermicro A+ Server**
  - Dual six-core AMD Istanbul
  - 32 GB DDR2
  - PCIe gen 2.0
  - QDR IB (32 Gbit/sec)
  - 3 Internal ATI Radeon 5870 GPUs

AC Cluster Software Stack

Conventional cluster

- **Shared system software**
  - Torque / Moab
  - ssh
- **Programming tools**
  - Matlab
  - Intel compiler
- **Other tools**
  - mvapich2 MPI (IB)

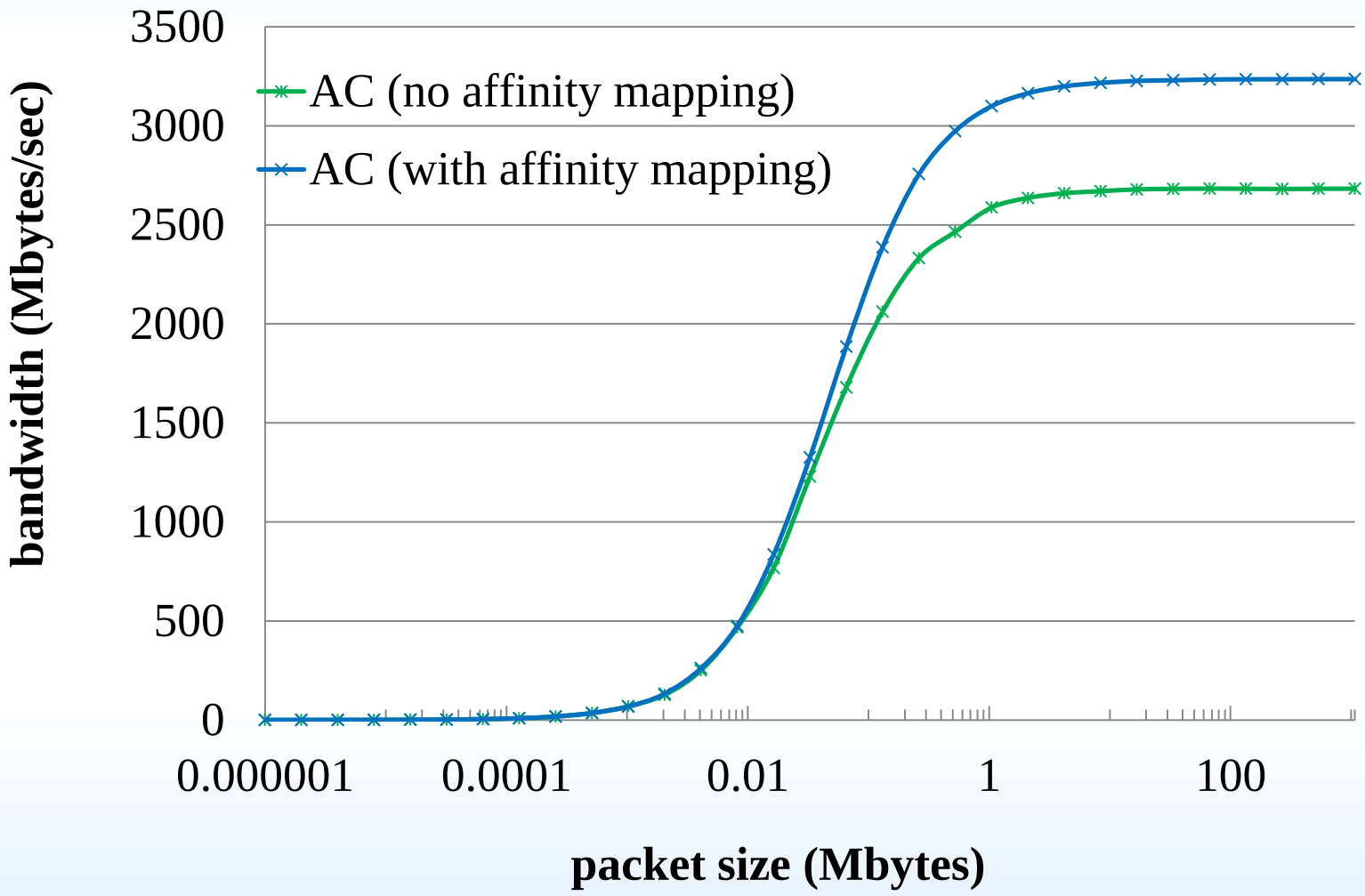
GPU cluster enhancements

- **Shared system software**
  - Torque extensions
  - CUDA/OpenCL wrapper
- **Programming tools**
  - CUDA C SDK
  - OpenCL SDK
  - PGI+GPU compiler
- **Other tools**
  - CUDA memory test
  - Power monitoring

CUDA/OpenCL Wrapper

- **Basic operation principle**
  - Use /etc/ld.so.preload to overload (intercept) a subset of CUDA/OpenCL functions, e.g. {cu|cuda}{Get|Set}Device, clGetDeviceIDs, etc.
  - Transparent operation
- **Purpose**
  - Enables controlled GPU device visibility and access, extending resource allocation to the workload manager
  - Provides a platform for rapid implementation and testing of HPC relevant features not available in NVIDIA APIs
- **Features**
  - NUMA Affinity mapping
    - Sets thread affinity to CPU core(s) nearest the GPU device
  - Shared host, multi-GPU device fencing
    - Only GPUs allocated by scheduler are visible or accessible to user
    - GPU device numbers are virtualized, with a fixed mapping to a physical device per user environment
    - User always sees allocated GPU devices indexed from 0

Host to device Bandwidth Comparison



CUDA/OpenCL Wrapper

- **Additional utilities**
  - showgputime utility
    - Shows percent time CUDA linked processes utilized GPU
    - Displays last 15 records (showallgputime shows all)
  - wrapper_query utility
    - Within any job environment, get details on what the wrapper library is doing
  - Memory Scrubber
    - Independent utility from wrapper, but packaged with it
    - Linux kernel does no management of GPU device memory
    - Must run between user jobs to ensure security between users
- **Availability**
  - NCSA/UofI Open Source License
  - https://sourceforge.net/projects/cudawrapper/

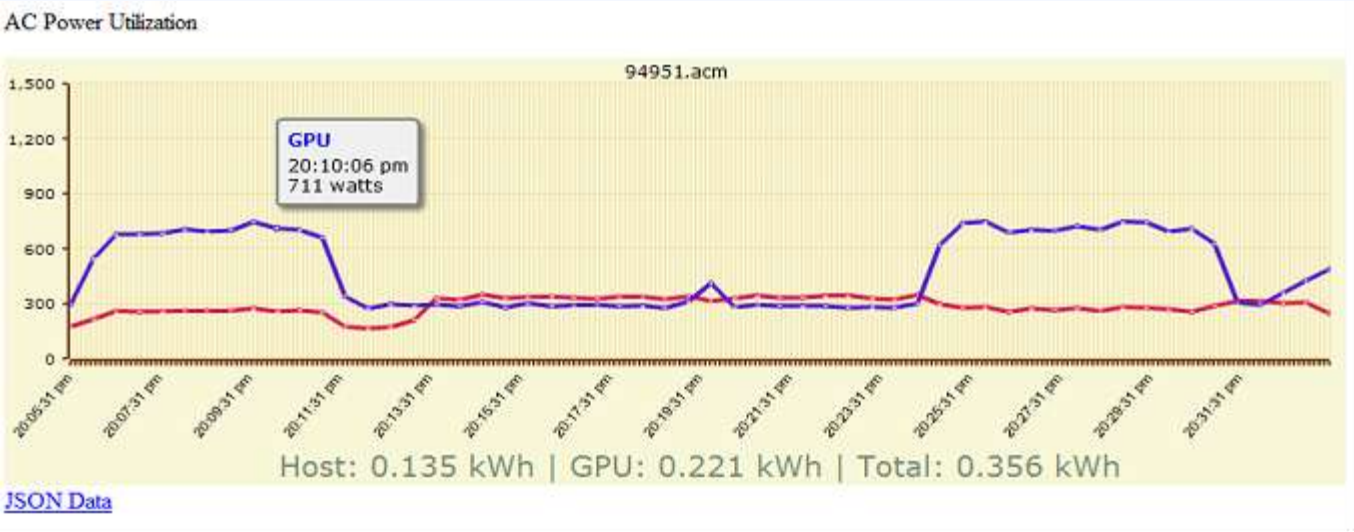
CUDA Memtest

- **4GB of Tesla GPU memory is not ECC protected**
  - Potential for producing erroneous results due to “soft” errors
- **Features**
  - Full re-implementation of every test included in memtest86
    - Random and fixed test patterns, error reports, error addresses, test specification
  - Email notification
  - Includes additional stress test for software and hardware errors
- **Usage scenarios**
  - Hardware test for defective GPU memory chips
  - CUDA API/driver software bugs detection
  - Hardware test for detecting soft errors due to non-ECC memory
  - Stress test for thermal loading
- **Availability**
  - NCSA/UofI Open Source License
  - https://sourceforge.net/projects/cudagpumemtest/

Power Profiling

- **Goals**
  - Accurately record power consumption of GPU and workstation (CPU) for performance per watt efficiency comparison
  - Make this data clearly and conveniently presented to application scientist
  - Accomplish this with cost effective hardware
- **Solution**
  - Modify inexpensive power meter to add logging capability
  - Integrate monitoring with job management infrastructure
    - submit job with prescribed resource (powermon)
  - Use web interface to present data in multiple forms to user

Power Profiling Example



improvement in performance-per-watt

$$e = p/p_a * s$$

Application	t (sec)	t_a (sec)	s	p (watt)	p_a (watt)	e
NAMD	6.6	1.1	6	316	681	2.78
VMD	1,465.2	57.5	25.5	299	742	10.48
QMCPACK			61.5	314	853	22.6
MILC	77,324	3,881	19.9	225	555	8.1

Future of Application Accelerators in HPC

- **Application accelerators will continue to play a role in HPC**
  - Cost, power, and size are the driving forces and application accelerators have an advantage
- **HPC clusters with application accelerators will continue to be the dominant architecture**
- **GPUs will continue to dominate the accelerator market**
  - NVIDIA has a GPU product line specifically designed for HPC
- **New major deployments are coming**
  - NSF Track 2D system, Keeneland
- **New accelerator architectures are coming**
  - Intel Many Integrated Core (MIC) architecture, AMD Fusion
- **We are starting to see more and more HPC applications ported to accelerators**