Programming for Hybrid Architectures Today and in the Future

John E. Stone

Theoretical and Computational Biophysics Group

Beckman Institute for Advanced Science and Technology

University of Illinois at Urbana-Champaign

http://www.ks.uiuc.edu/Research/gpu/

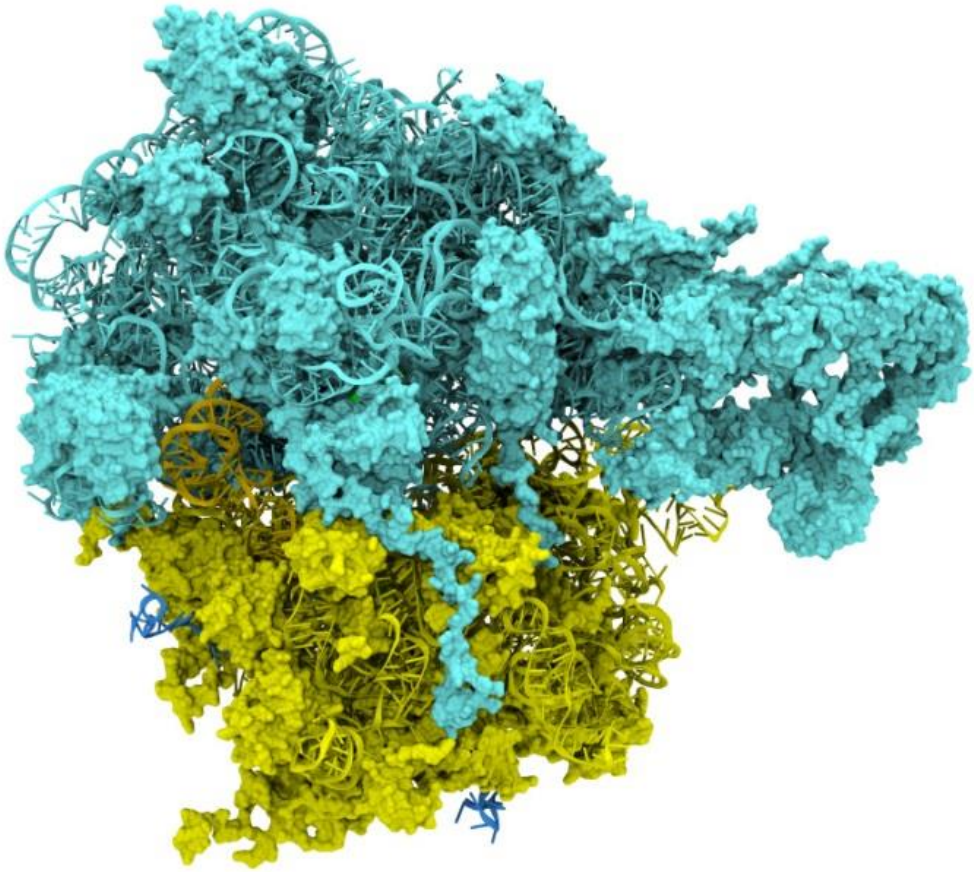
LSU Third Annual HPC User Symposium

Louisiana State University, June 5, 2014

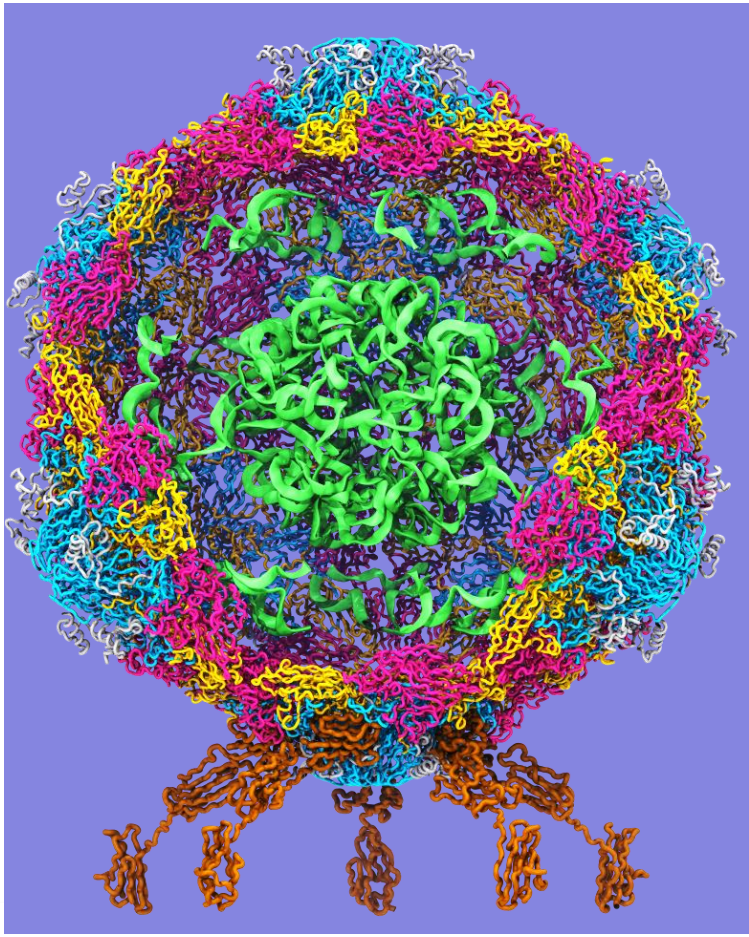
Goal: A Computational Microscope

Study the molecular machines in living cells

Ribosome: target for antibiotics

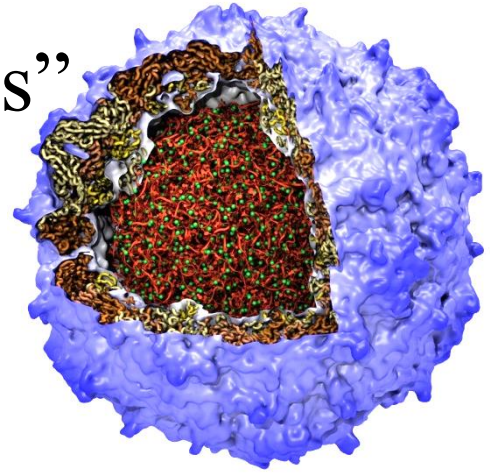


Poliovirus

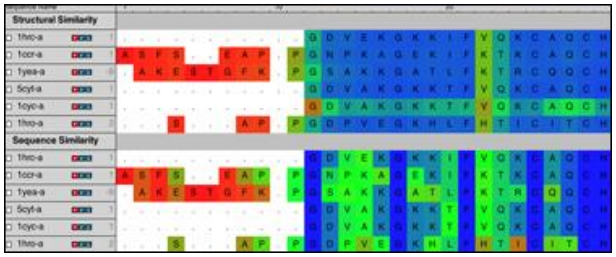


VMD – "Visual Molecular Dynamics"

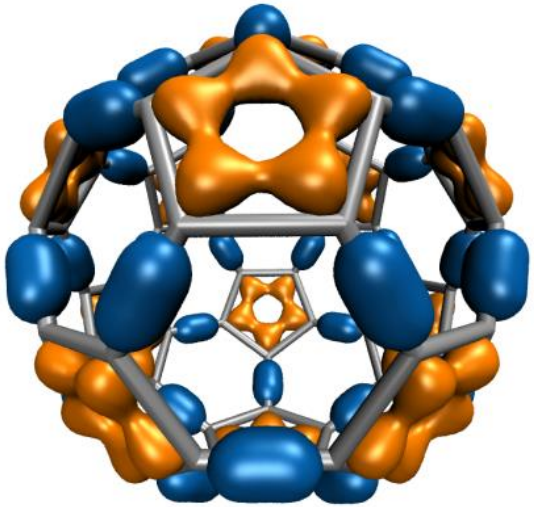
- Visualization and analysis of:
  - molecular dynamics simulations
  - quantum chemistry calculations
  - particle systems and whole cells
  - sequence data
- User extensible w/ scripting and plugins
- http://www.ks.uiuc.edu/Research/vmd/



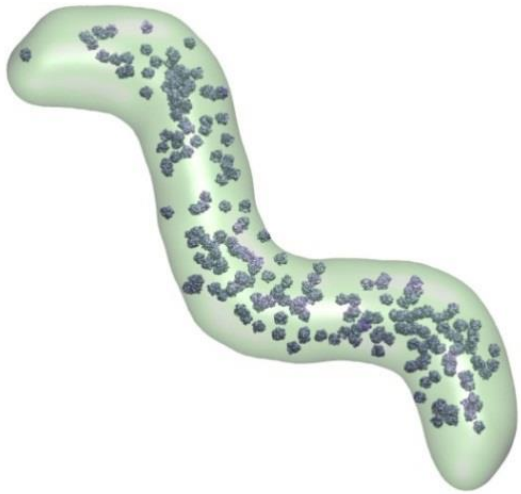
Poliovirus



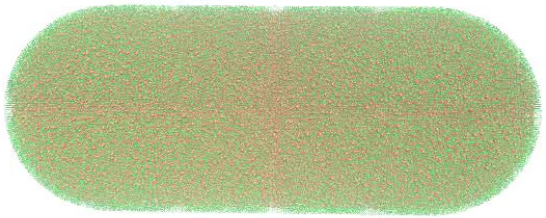
Ribosome Sequences



Electrons in
Vibrating Buckyball

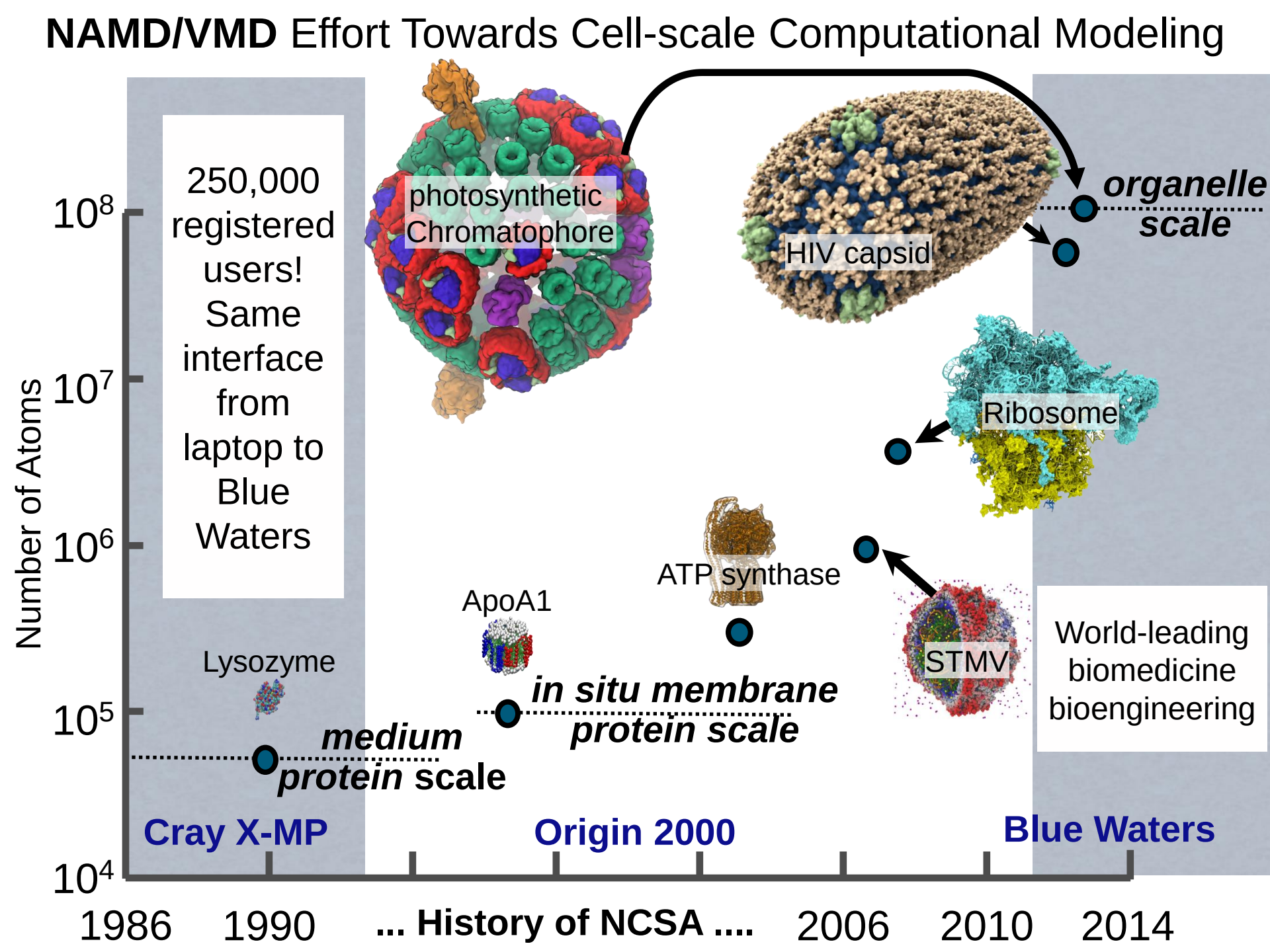


Cellular Tomography,
Cryo-electron Microscopy



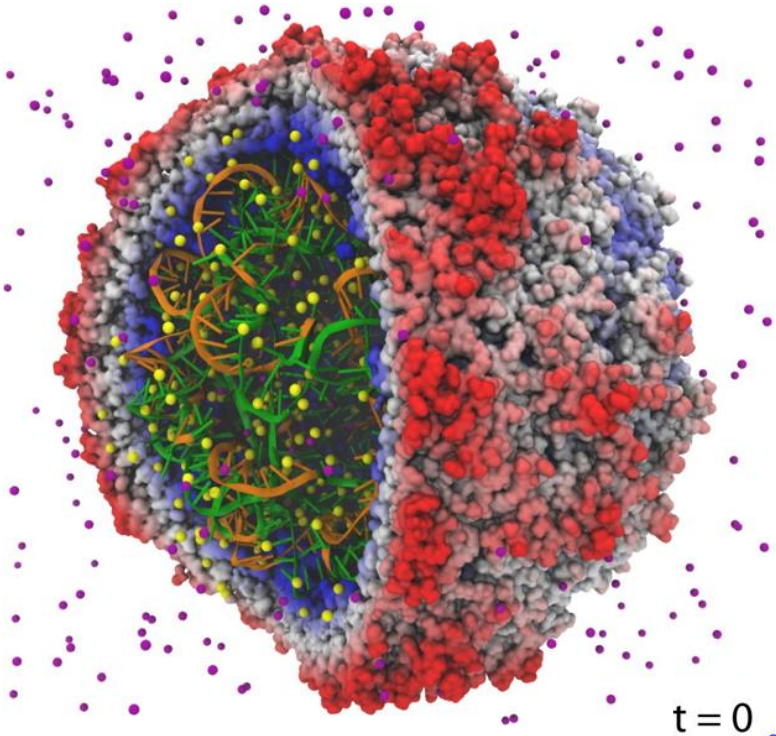
Whole Cell Simulations

NAMD/VMD Effort Towards Cell-scale Computational Modeling



First Simulation of a Virus Capsid (2006)

Satellite Tobacco Mosaic Virus (STMV)

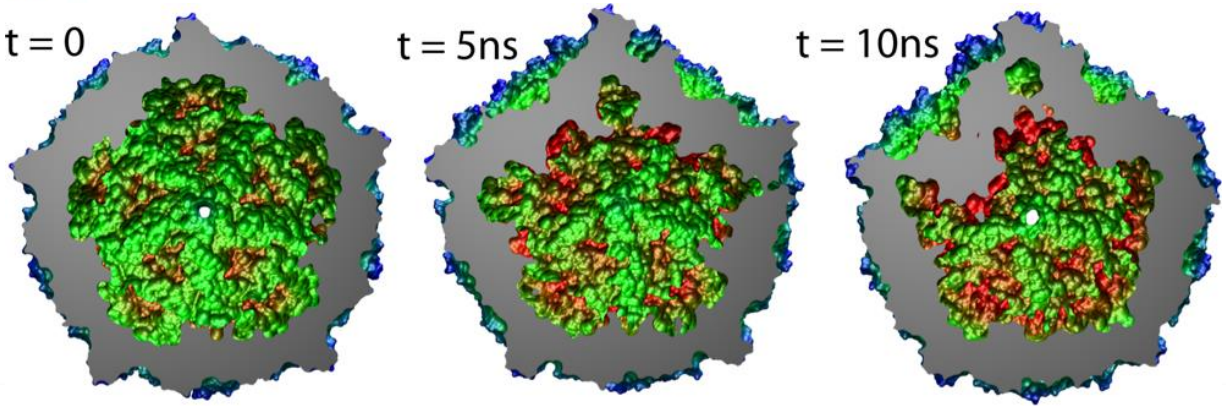


First MD simulation of a complete virus capsid

STMV smallest available capsid structure

STMV simulation, visualization, and analysis pushed us toward GPU computing!

MD showed that STMV capsid collapses without its RNA core



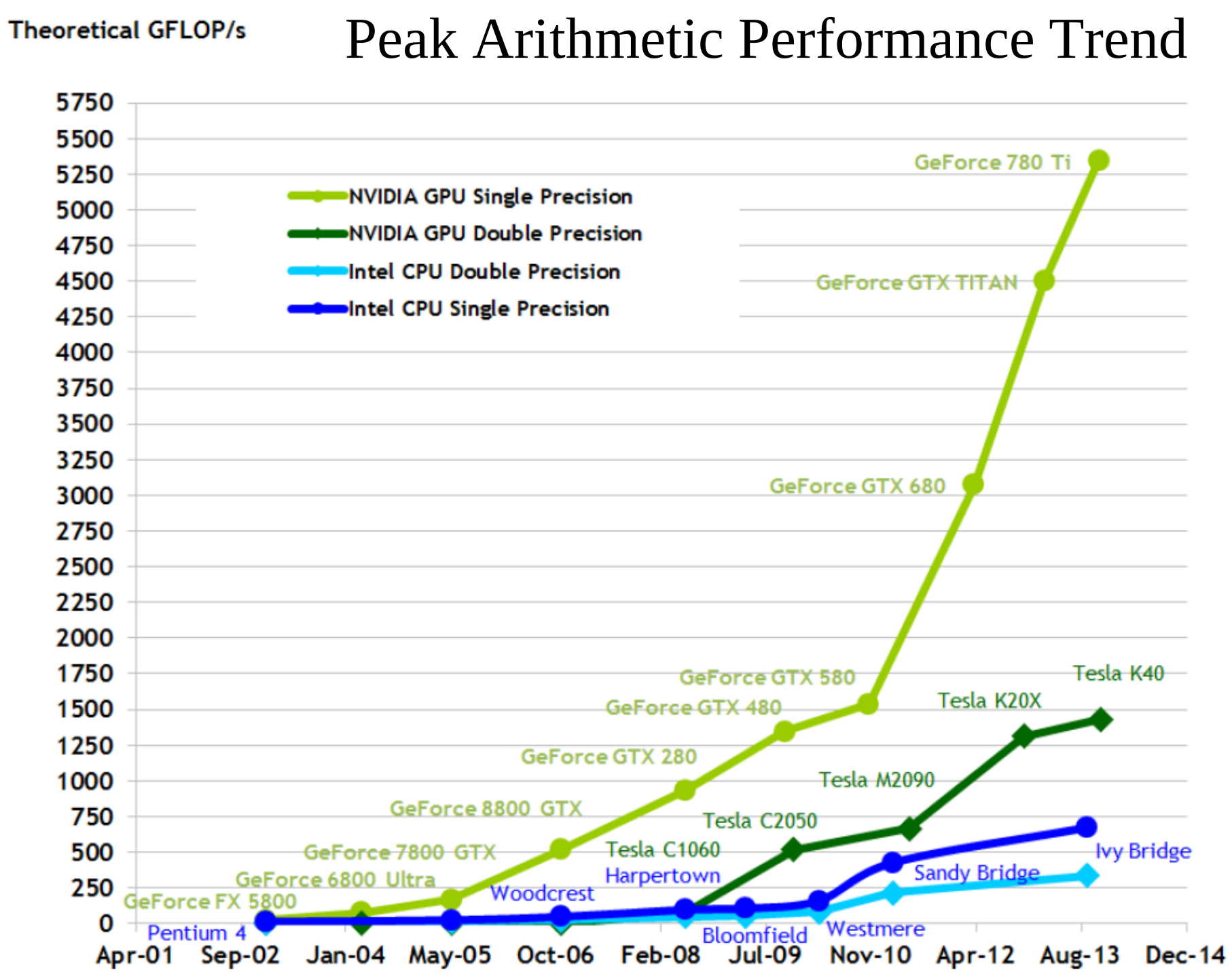
1 million atoms
A huge system for 2006

Freddolino et al., ***Structure***, **14**:437 (2006)

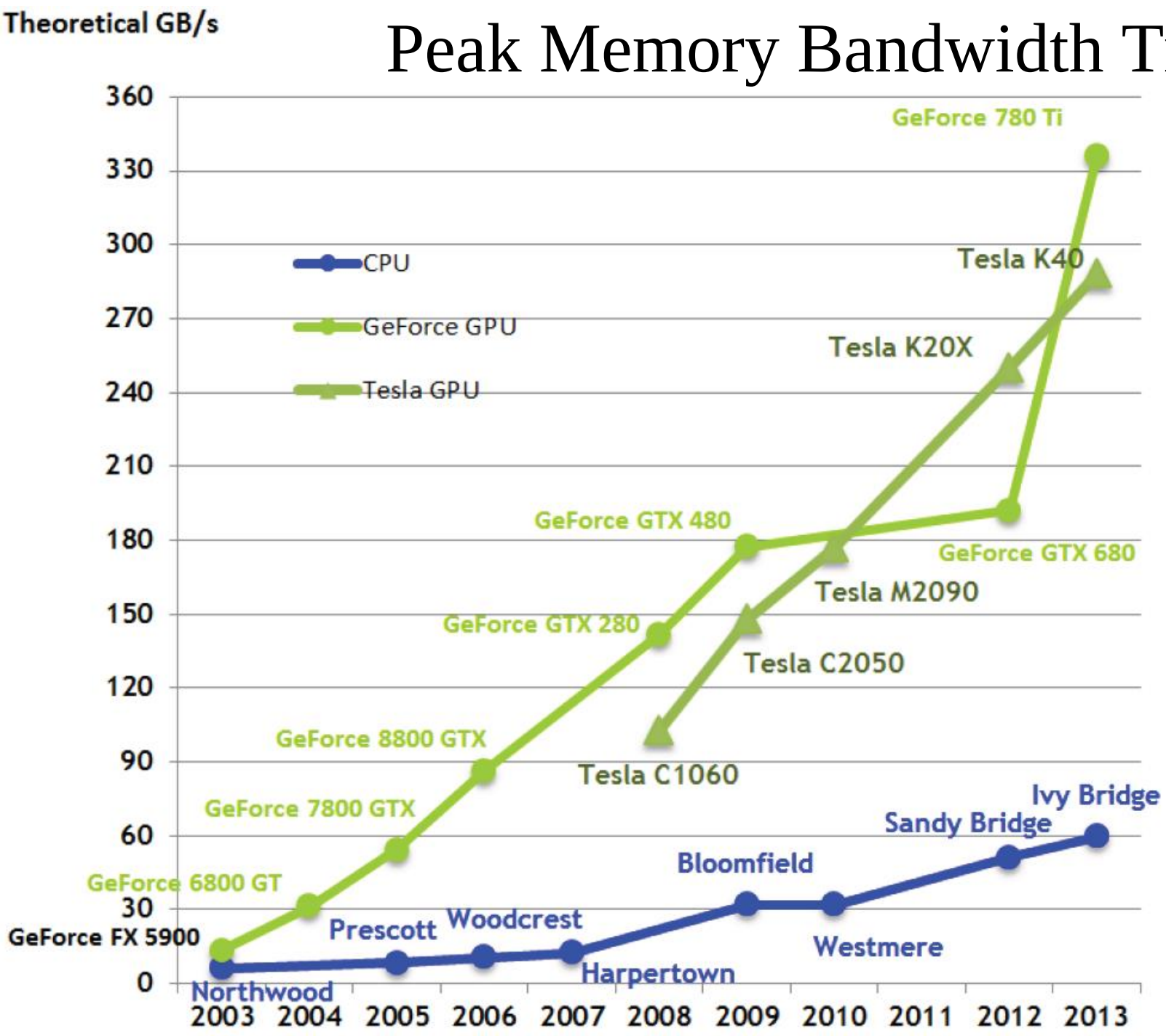
GPU Computing

- Commodity devices, omnipresent in modern computers (over a **million** sold per **week**)
- Massively parallel hardware, hundreds of processing units, **throughput oriented architecture**
- Standard integer and floating point types supported
- Programming tools allow software to be written in dialects of familiar C/C++ and integrated into legacy software
- GPU algorithms are often multicore friendly due to attention paid to **data locality** and **data-parallel** work decomposition

Peak Arithmetic Performance Trend



Peak Memory Bandwidth Trend

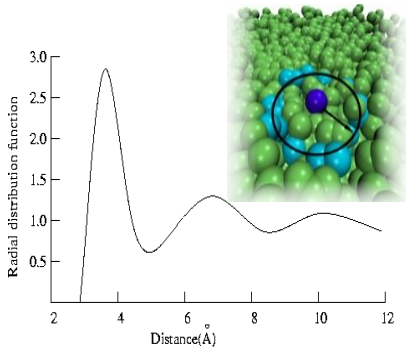


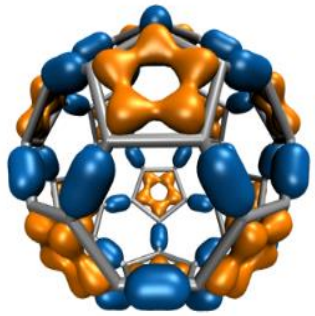
What Speedups Can GPUs Achieve?

- Single-GPU speedups of **2.5x** to **8x** vs. one CPU socket are common
- Best speedups can reach **25x** or more, attained on codes dominated by floating point arithmetic, especially native GPU machine instructions, e.g. expf(), rsqrtf(), …
- **Amdahl's Law** can prevent legacy codes from achieving peak speedups with shallow GPU acceleration efforts

CUDA GPU-Accelerated Trajectory Analysis and Visualization in VMD

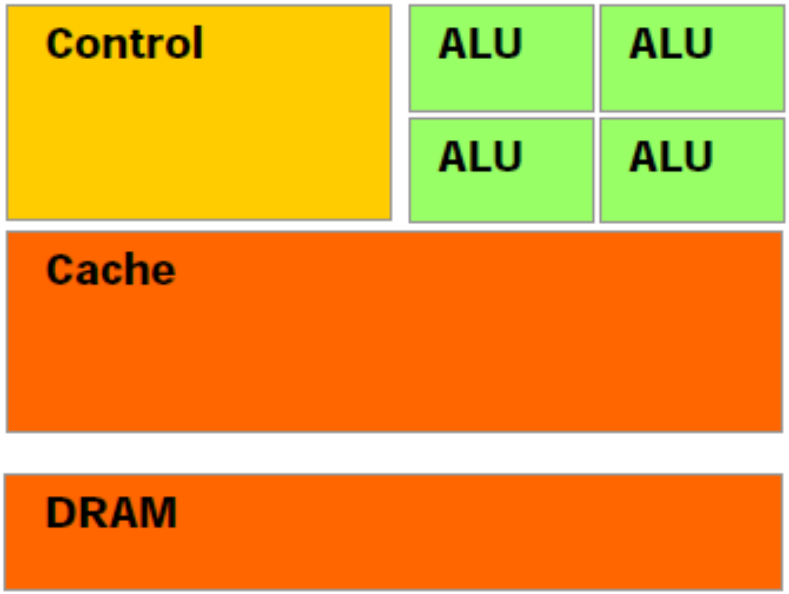
VMD GPU-Accelerated Feature or Kernel	Typical speedup vs. multi-core CPU (e.g. 4-core CPU)
Molecular orbital display	**30x**
Radial distribution function	**23x**
Molecular surface display	**15x**
Electrostatic field calculation	**11x**
Ray tracing w/ shadows, AO lighting	**7x**
Ion placement	6x
MDFF density map synthesis	6x
Implicit ligand sampling	6x
Root mean squared fluctuation	6x
Radius of gyration	5x
Close contact determination	5x
Dipole moment calculation	4x



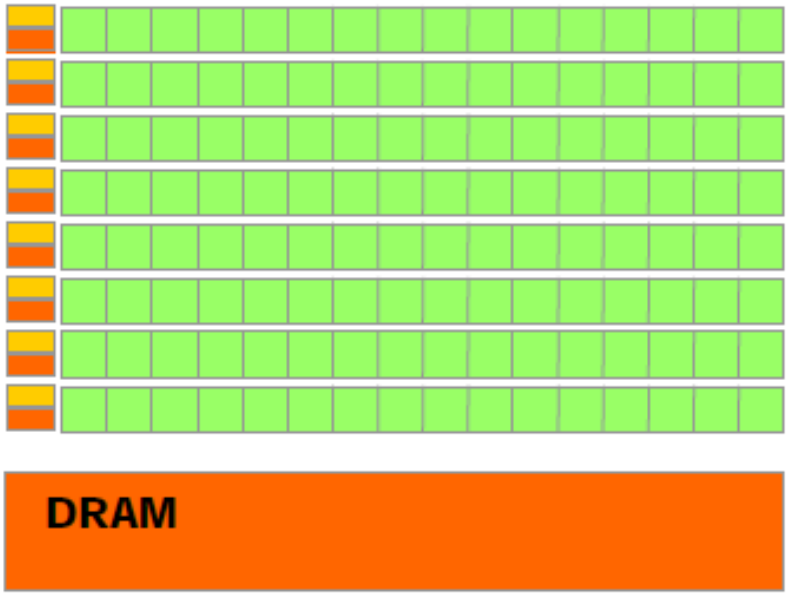


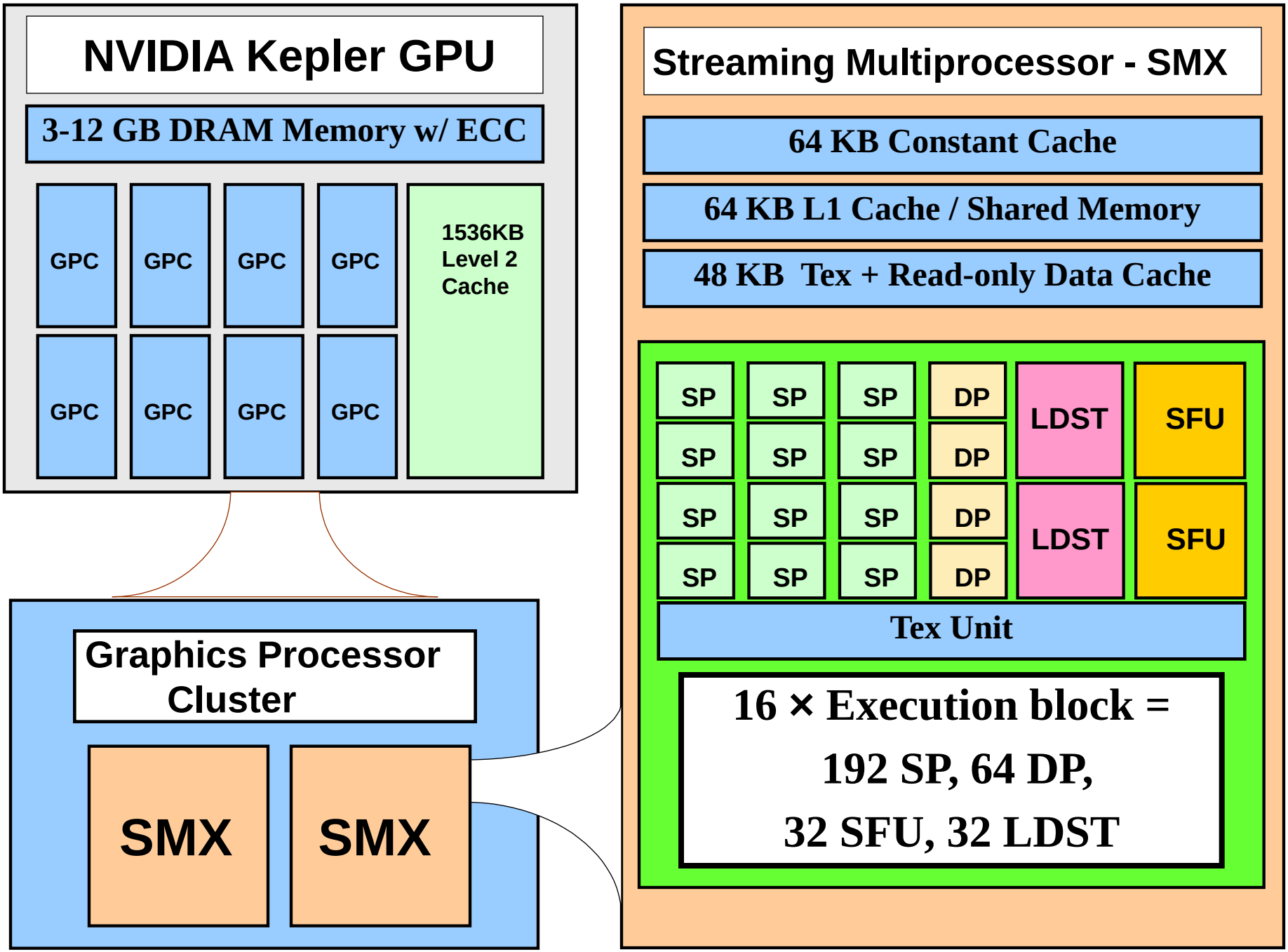
Comparison of CPU and GPU Hardware Architecture

CPU: Cache heavy, focused on individual thread performance



GPU: ALU heavy, massively parallel, throughput oriented




NVIDIA Kepler GPU
3-12 GB DRAM Memory w/ ECC
GPC
GPC
GPC
GPC
GPC
GPC
GPC
GPC
GPC
1536KB Level 2 Cache
Graphics Processor Cluster
SMX
SMX
Streaming Multiprocessor - SMX
64 KB Constant Cache
64 KB L1 Cache / Shared Memory
48 KB Tex + Read-only Data Cache
SP
DP
LDST
SFU
Tex Unit
16 × Execution block =
192 SP, 64 DP,
32 SFU, 32 LDST

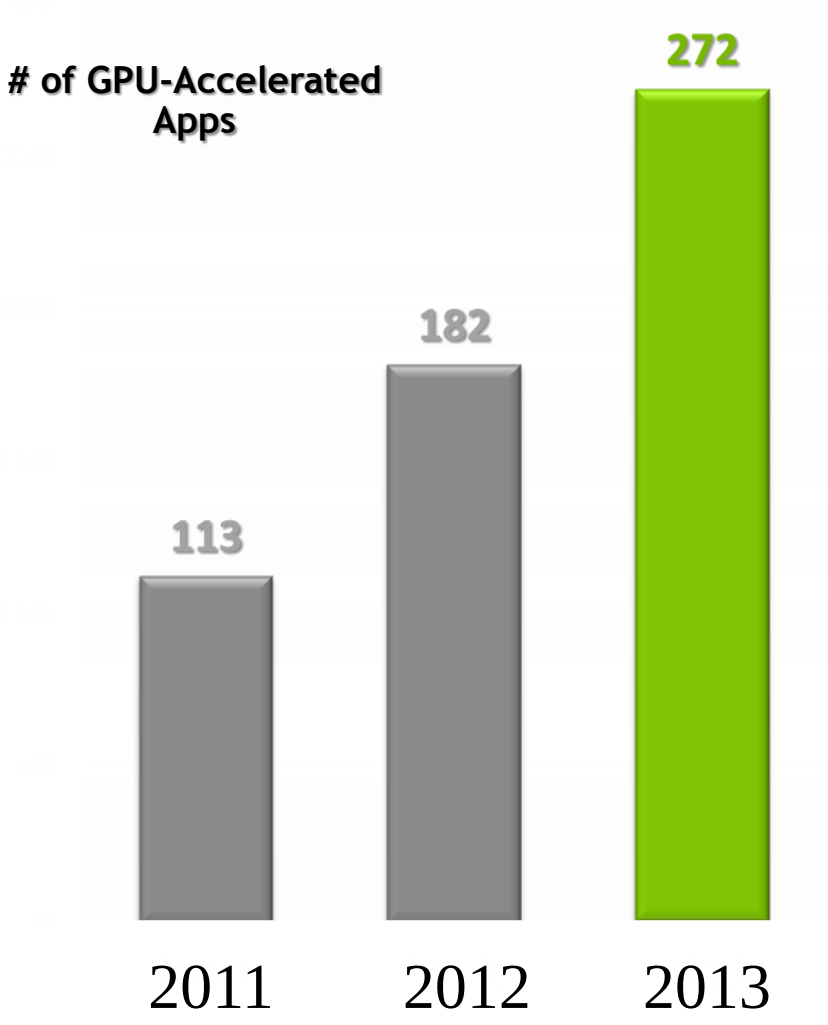
Hybrid Systems Lead Top500, Green500

- Four of the top ten Top500 systems are hybrid architectures
  - #1 NSCC Tianhe-2
  - #2 ORNL Titan
  - #6 CSCS Piz Daint
  - #7 TACC Stampede
- All of top ten Green500 systems are GPU-based hybrid architectures

Hybrid Architectures Becoming Common for Mid-range Systems

- Upcoming LSU Cluster: ~960 GPUs
- 2013: Indiana Big Red II: ~600 GPUs
- 2012: Georgia Tech Keeneland: 792 GPUs

Solid Growth of GPU Accelerated Apps



Courtesy NVIDIA

Top HPC Applications

Molecular Dynamics	AMBER CHARMM DESMOND	GROMACS LAMMPS NAMD
Quantum Chemistry	Abinit Gaussian	GAMESS NWChem
Material Science	CP2K QMCPACK	Quantum Espresso VASP
Weather & Climate	COSMO GEOS-5 HOMME	CAM-SE NEMO NIM WRF
Lattice QCD	Chroma	MILC
Plasma Physics	GTC	GTS
Structural Mechanics	ANSYS Mechanical LS-DYNA Implicit MSC Nastran	OptiStruct Abaqus/Standard
Fluid Dynamics	ANSYS Fluent	Culises (OpenFOAM)

Accelerated, In Development

Major Approaches For Programming Hybrid Architectures

- Use drop-in libraries in place of CPU-based libraries
  - Little or no code development
  - Speedups limited by Amdahl's Law and overheads associated with data movement between CPUs and GPU accelerators
  - Examples: MAGMA, BLAS-variants, FFT libraries, etc.
- Generate accelerator code as a variant of CPU source, e.g. using OpenMP w/ OpenACC, similar methods
- Write lower-level accelerator-specific code, e.g. using CUDA, OpenCL, other approaches

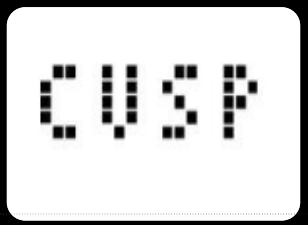
GPU Accelerated Libraries

"Drop-in" Acceleration for your Applications

Linear Algebra

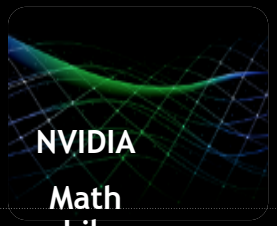
FFT, BLAS,

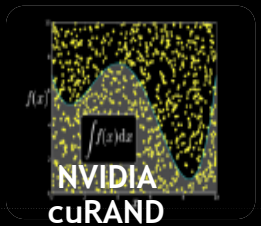
SPARSE, Matrix



Numerical & Math

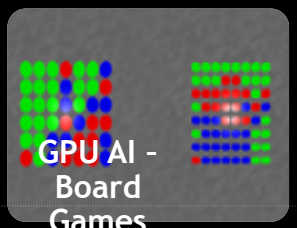
RAND, Statistics





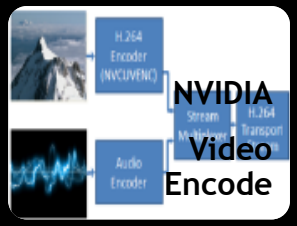
Data Struct. & AI

Sort, Scan, Zero Sum



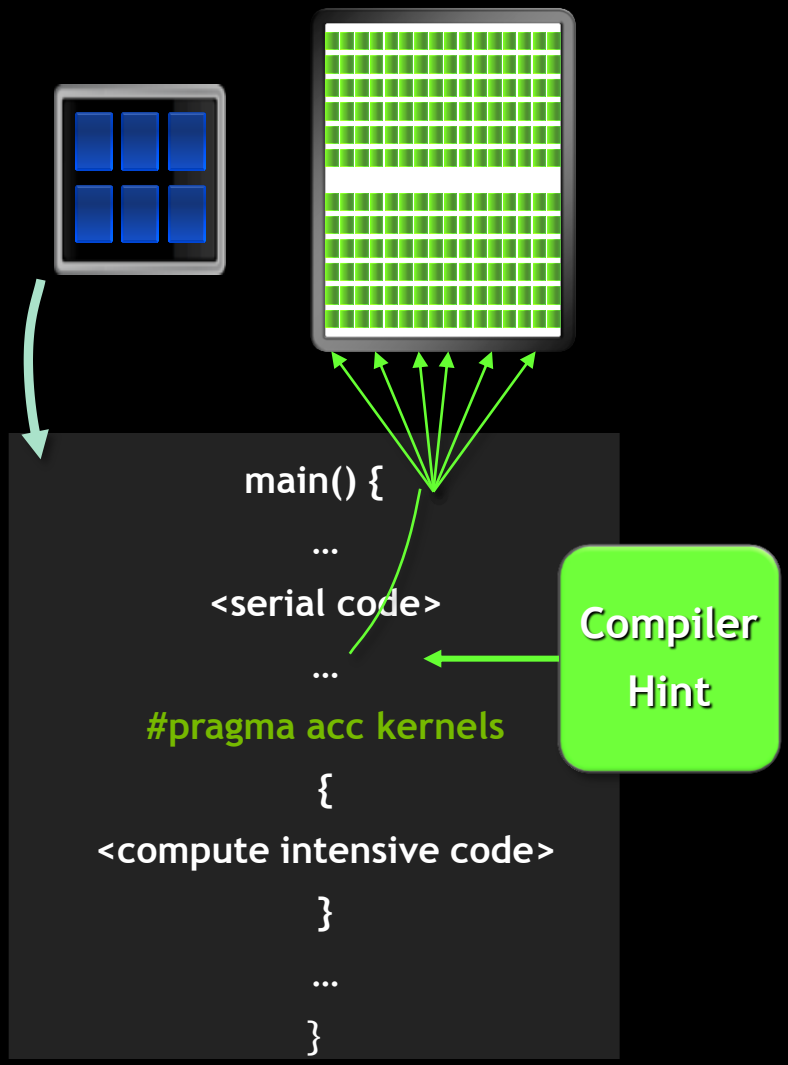
Visual Processing

Image & Video



Courtesy NVIDIA

OpenACC: Open, Simple, Portable

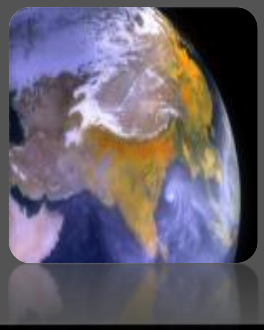


- Open Standard
- Easy, Compiler-Driven Approach
- Portable on GPUs and Xeon Phi

CAM-SE Climate

6x Faster on GPU

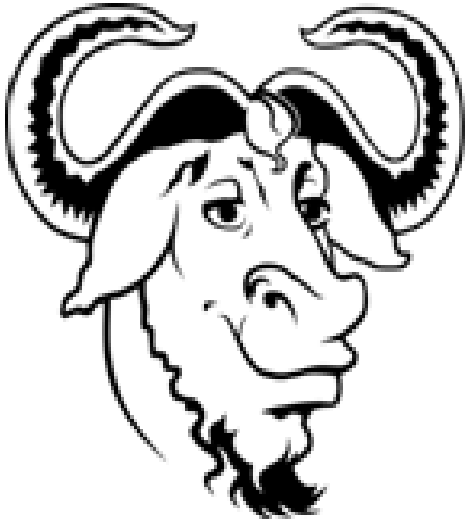
Top Kernel: 50% of Runtime



Courtesy NVIDIA

Linux GCC Compiler to Support GPU Accelerators

- **Open Source**
  - GCC Efforts by Samsung & Mentor Graphics
- **Pervasive Impact**
  - Free to all Linux users
- **Mainstream**
  - Most Widely Used HPC Compiler

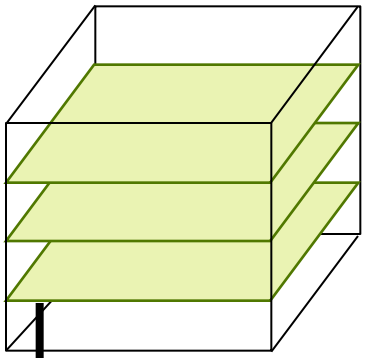


> *"Incorporating OpenACC into GCC is an excellent example of open source and open standards working together to make accelerated computing broadly accessible to all Linux developers."*
>
> Oscar Hernandez
>
> Oak Ridge National Laboratories

OAK RIDGE National Laboratory

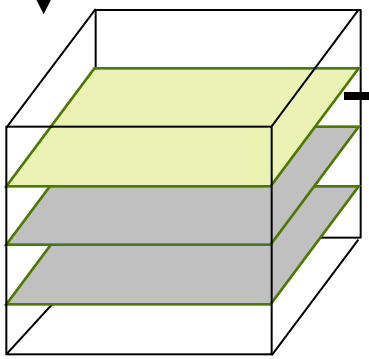
GPU Solution: Computing C_{60} Molecular Orbitals

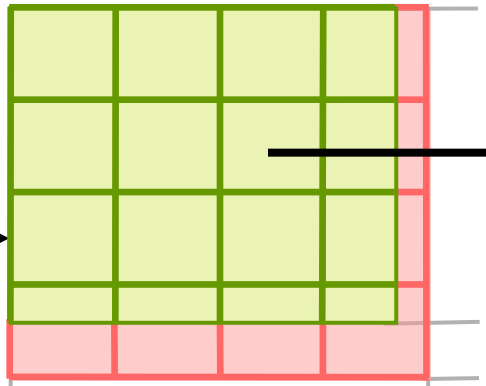
3-D orbital lattice: millions of points



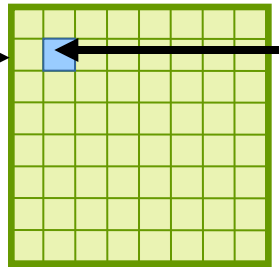
Device	CPUs, GPUs	Runtime (s)	Speedup
2x Intel X5550-SSE	8	4.13	1
GeForce GTX 480	1	0.255	16
GeForce GTX 480	4	0.081	51

Lattice slices computed on multiple GPUs





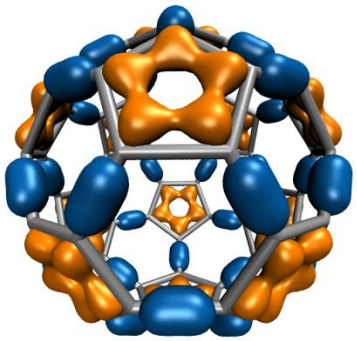
2-D CUDA grid on one GPU



CUDA thread blocks

GPU threads each compute one point.

Molecular Orbital Inner Loop, Hand-Coded x86 SSE
Hard to Read, Isn't It? (And this is the "pretty" version!)

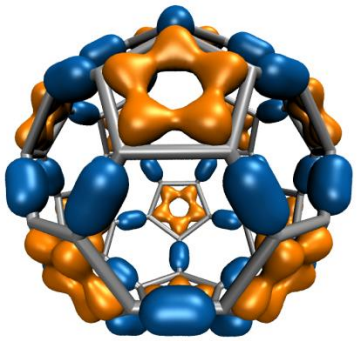


```
for (shell=0; shell < maxshell; shell++) {
  __m128 Cgto = _mm_setzero_ps();
  for (prim=0; prim<num_prim_per_shell[shell_counter]; prim++) {
    float exponent       = -basis_array[prim_counter    ];
    float contract_coeff =  basis_array[prim_counter + 1];
    __m128 expval = _mm_mul_ps(_mm_load_ps1(&exponent), dist2);
    __m128 ctmp = _mm_mul_ps(_mm_load_ps1(&contract_coeff), exp_ps(expval));
    Cgto = _mm_add_ps(contracted_gto, ctmp);
    prim_counter += 2;
  }
  __m128 tshell = _mm_setzero_ps();
  switch (shell_types[shell_counter]) {
    case S_SHELL:
      value = _mm_add_ps(value, _mm_mul_ps(_mm_load_ps1(&wave_f[ifunc++]), Cgto));   break;
    case P_SHELL:
      tshell = _mm_add_ps(tshell, _mm_mul_ps(_mm_load_ps1(&wave_f[ifunc++]), xdist));
      tshell = _mm_add_ps(tshell, _mm_mul_ps(_mm_load_ps1(&wave_f[ifunc++]), ydist));
      tshell = _mm_add_ps(tshell, _mm_mul_ps(_mm_load_ps1(&wave_f[ifunc++]), zdist));
      value = _mm_add_ps(value, _mm_mul_ps(tshell, Cgto));      break;
```

Writing SSE kernels for CPUs requires assembly language, compiler intrinsics, various libraries, or a really smart autovectorizing compiler **and lots of luck...**

Molecular Orbital Inner Loop in CUDA

```
for (shell=0; shell < maxshell; shell++) {
    float contracted_gto = 0.0f;
    for (prim=0; prim<num_prim_per_shell[shell_counter]; prim++) {
      float exponent          = const_basis_array[prim_counter     ];
      float contract_coeff = const_basis_array[prim_counter + 1];
      contracted_gto += contract_coeff * exp2f(-exponent*dist2);
      prim_counter += 2;
    }
    float tmpshell=0;
    switch (const_shell_symmetry[shell_counter]) {
      case S_SHELL:
        value += const_wave_f[ifunc++] * contracted_gto;    break;
      case P_SHELL:
        tmpshell += const_wave_f[ifunc++] * xdist;
        tmpshell += const_wave_f[ifunc++] * ydist
        tmpshell += const_wave_f[ifunc++] * zdist;
        value += tmpshell * contracted_gto;   break;
```



Aaaaahhhh….

Data-parallel CUDA kernel looks like normal C code for the most part….

GPU On-Board Global Memory

- GPU arithmetic rates dwarf memory bandwidth
- For Kepler K40 hardware:
  - ~4.3 SP TFLOPS vs. ~288 GB/sec
  - The ratio is roughly **60 FLOPS per memory reference** for single-precision floating point
- Peak performance achieved with **"coalesced"** memory access patterns – patterns that result in a single hardware memory transaction for a SIMD **"warp" – a contiguous group of 32 threads**

Getting Performance From GPUs

- Don't worry (much) about counting arithmetic operations…at least until you have nothing else left to do
- GPUs provide tremendous memory bandwidth, but even so, **memory bandwidth often ends up being the performance limiter**
- Keep/reuse data in **registers** as long as possible
- The main consideration when programming GPUs is **accessing memory efficiently**, and storing operands in the **most appropriate memory system** according to data size and access pattern

Using the CPU to Optimize GPU Performance

- GPU performs best when the work evenly divides into the number of threads/processing units
- Optimization strategy:
  - Use the CPU to ***"regularize"*** the GPU workload
  - Use fixed size bin data structures, with "empty" slots skipped or producing zeroed out results
  - Handle exceptional or irregular work units on the CPU; GPU processes the bulk of the work concurrently
  - On average, the GPU is kept highly occupied, attaining a high fraction of peak performance

GPU On-Chip Memory Systems

- GPU arithmetic rates dwarf global memory bandwidth
- GPUs include multiple fast **on-chip** memories to help **narrow the gap**:
  - **Registers**
  - Constant memory (64KB)
  - **Shared memory (48KB / 16KB)**
  - Read-only data cache / Texture cache (~48KB)
    - Hardware-assisted 1-D, 2-D, 3-D locality
    - Hardware range clamping, type conversion, interpolation

Avoiding Shared Memory Bank Conflicts:
Array of Structures (AOS) vs. Structure of Arrays (SOA)

- AOS:

```
typedef struct {
  float x;
  float y;
  float z;
} myvec;
myvec aos[1024];
aos[threadIdx.x].x = 0;
aos[threadIdx.x].y = 0;
```

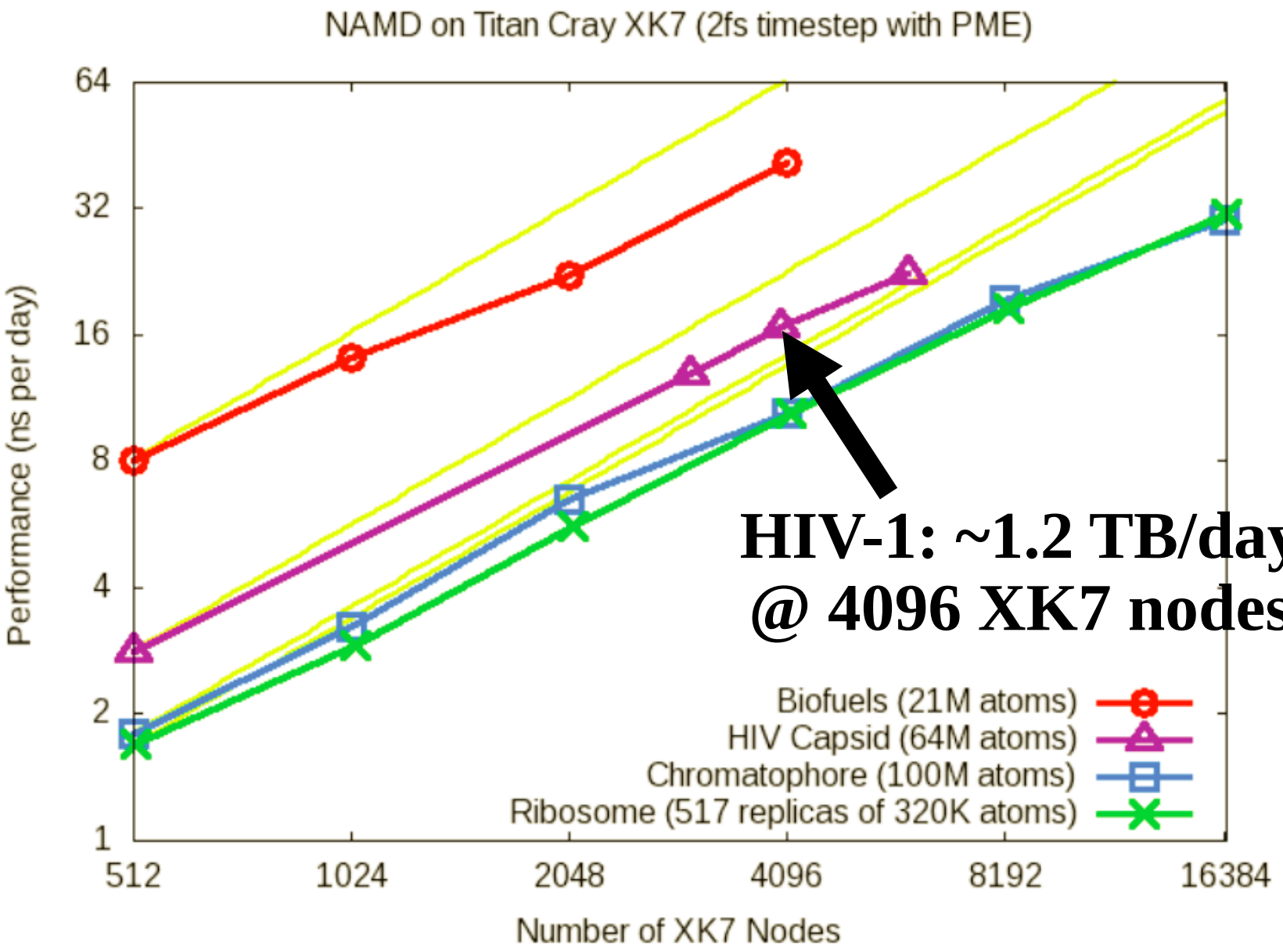
- SOA:

```
typedef struct {
  float x[1024];
  float y[1024];
  float z[1024];
} myvecs;
myvecs soa;
soa.x[threadIdx.x] = 0;
soa.y[threadIdx.x] = 0;
```

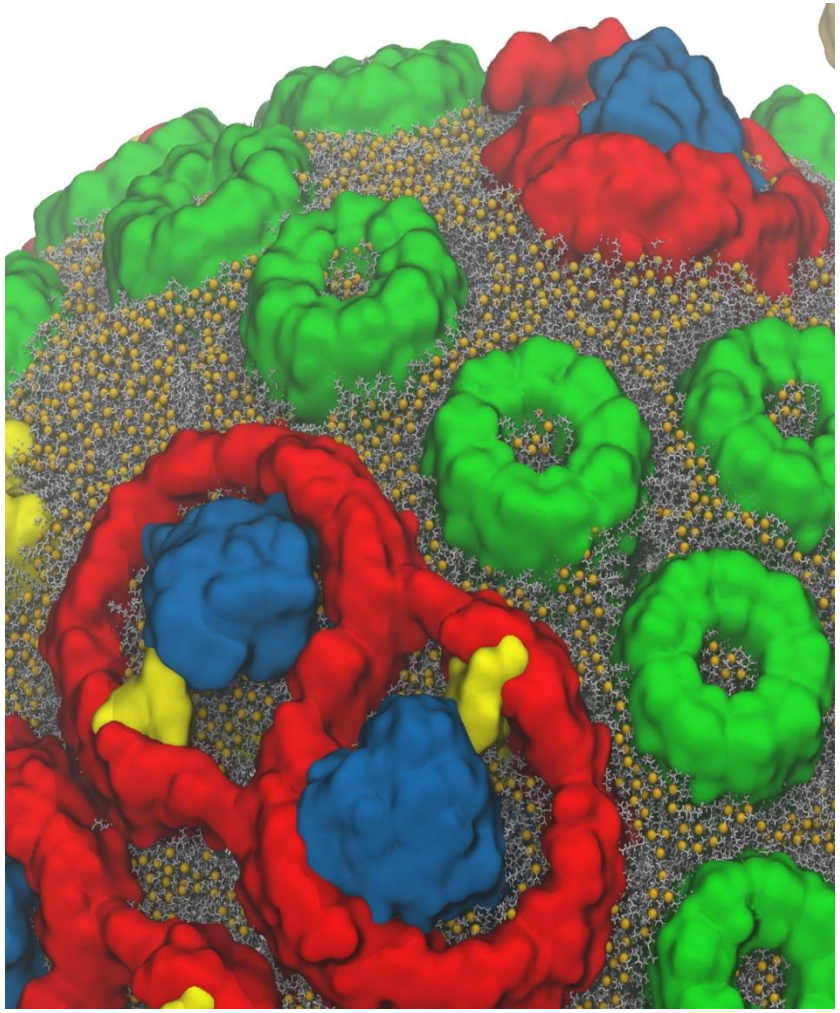

nature
THE INTERNATIONAL WEEKLY JOURNAL OF SCIENCE
THE HIV-1 CAPSID
Atomic structure of the AIDS pathogen's protein coat
PAGE 643
COSMOLOGY
THE FIRST LIGHT
In pursuit of the most distant galaxies
PAGE 554
CITATION
CROSSING THE BORDERS
International collaborations make the most impact
PAGE 557
ANTICANCER DRUGS
A SITTING TARGET
An indirect hit on 'undruggable' KRAS protein
PAGES 577 & 630
NATURE.COM/NATURE
30 May 2013


VOLUME 18, ISSUE 1
IAVIReport
The Publication on AIDS Vaccine Research
WWW.IAVIREPORT.ORG | VOLUME 18, ISSUE 1 | 2014
CROI: Progress on Prevention and Cure
Plus:
A report from the Keystone Symposium

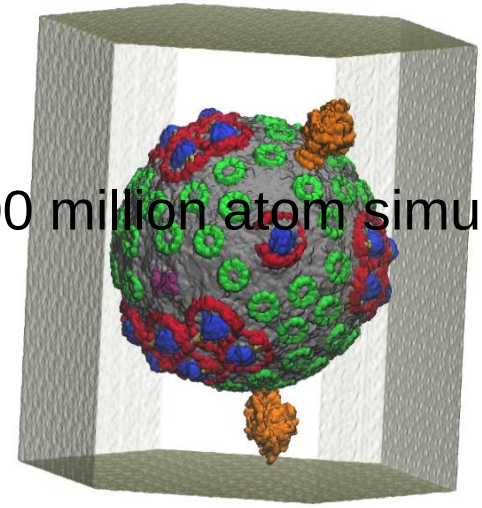
NAMD Titan XK7 Performance August 2013



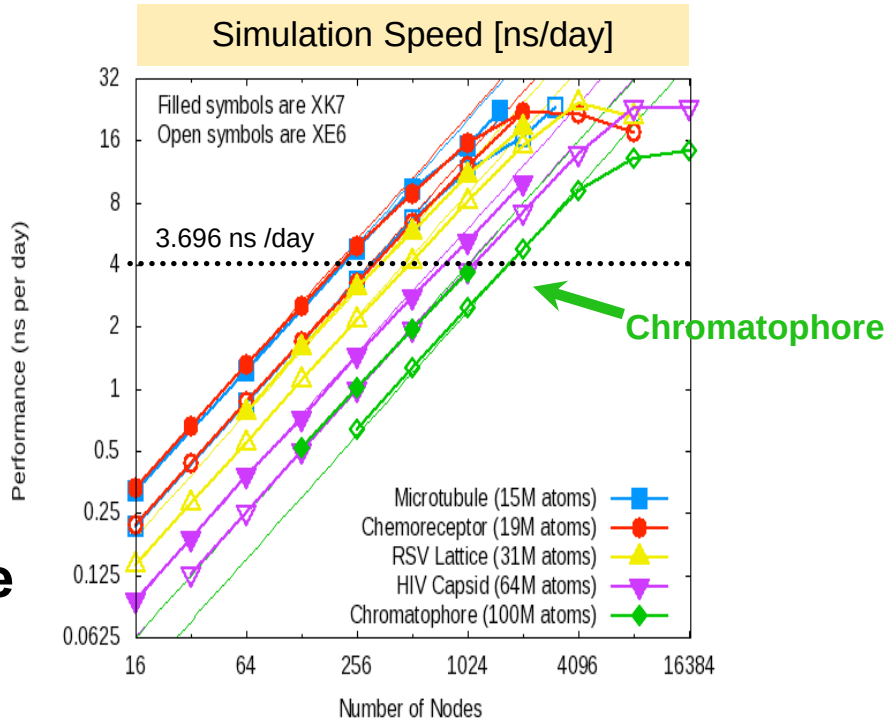
100-million Atom Simulation of Chromatophore



segment of simulated chromatophore showing lipids



100 million atom simulation



VMD Petascale Visualization and Analysis

- Analyze/visualize large trajectories too large to transfer off-site:
  - Compute time-averaged electrostatic fields, MDFF quality-of-fit, etc.
  - User-defined parallel analysis operations, data types
  - Parallel rendering, movie making
- Parallel I/O rates up to **275 GB/sec** on 8192 Cray XE6 nodes – can read in **231 TB in 15 minutes!**
- Multi-level dynamic load balancing tested with up to 262,144 CPU cores
- **Supports GPU-accelerated Cray XK7 nodes for both visualization and analysis usage**

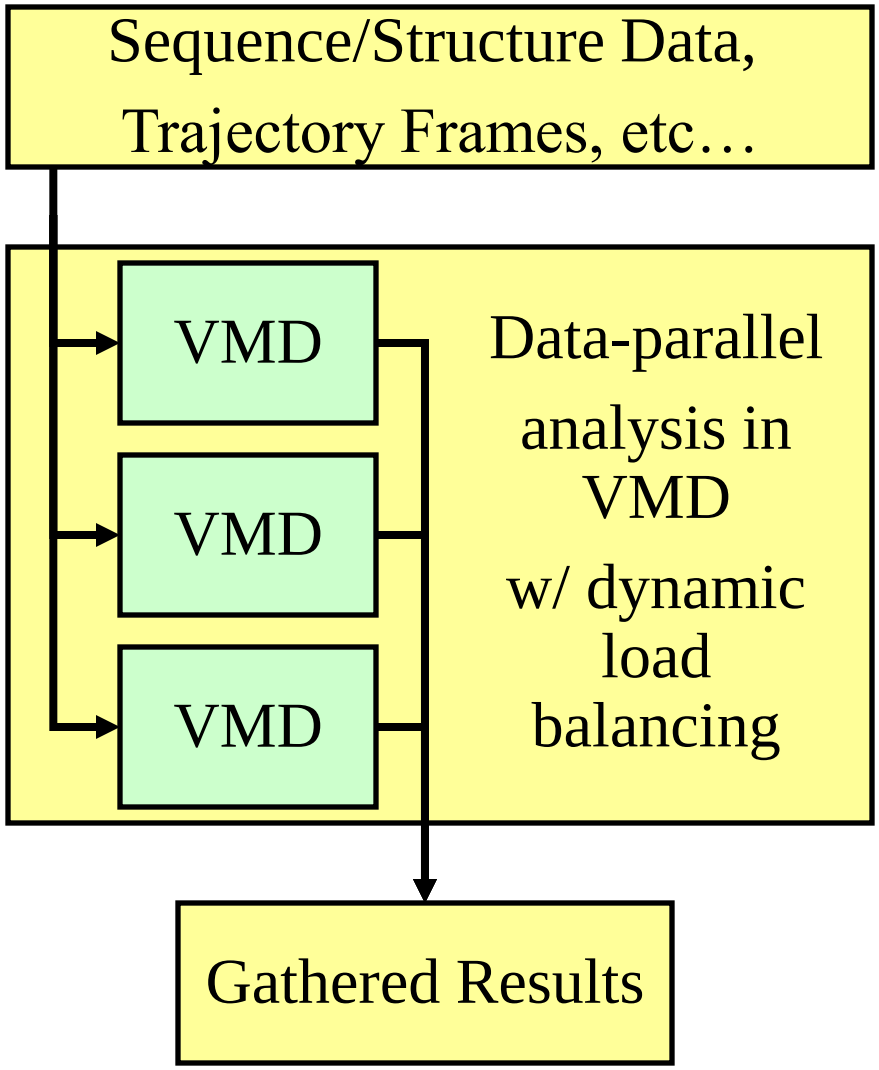
NCSA Blue Waters Hybrid Cray XE6 / XK7 Supercomputer

22,640 XE6 CPU nodes

4,224 XK7 nodes w/ GPUs support fast VMD OpenGL movie rendering and visualization

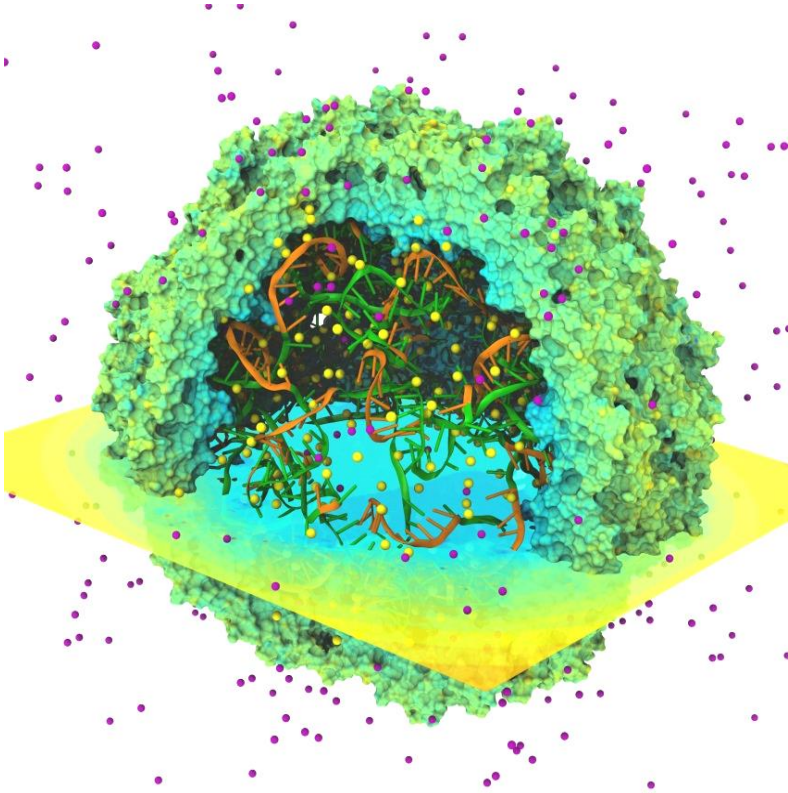
VMD for Demanding Analysis Tasks
Parallel VMD Analysis w/ MPI

- Compute time-averaged electrostatic fields, MDFF quality-of-fit, etc.
- Parallel rendering, movie making
- User-defined parallel reduction operations, data types
- **Parallel I/O on Blue Waters:**
  - **109 GB/sec on 512 nodes**
  - **275 GB/sec on 8,192 nodes**
- **Timeline per-residue SASA calc. achieves 800x speedup @ 1000 BW XE6 nodes**
- **Supports GPU-accelerated clusters and supercomputers**



Time-Averaged Electrostatics Analysis on Energy-Efficient GPU Cluster

- **1.5 hour** job (CPUs) reduced to **3 min** (CPUs+GPU)
- Electrostatics of thousands of trajectory frames averaged
- Per-node power consumption on NCSA "AC" GPU cluster:
  - CPUs-only: 448 Watt-hours
  - CPUs+GPUs: 43 Watt-hours
- GPU Speedup: **25.5x**
- Power efficiency gain: **10.5x**



Quantifying the Impact of GPUs on Performance and Energy Efficiency in HPC Clusters. J. Enos, C. Steffen, J. Fullop, M. Showerman, G. Shi, K. Esler, V. Kindratenko, J. Stone, J. Phillips. *The Work in Progress in Green Computing*, pp. 317-324, 2010.

Time-Averaged Electrostatics Analysis on NCSA Blue Waters

NCSA Blue Waters Node Type	**Seconds per trajectory frame for one compute node**
Cray XE6 Compute Node: 32 CPU cores (2xAMD 6200 CPUs)	9.33
Cray XK6 GPU-accelerated Compute Node: 16 CPU cores + **NVIDIA X2090 (Fermi) GPU**	2.25
Speedup for GPU XK6 nodes vs. CPU XE6 nodes	**XK6 nodes are 4.15x faster overall**
Tests on XK7 nodes indicate MSM is CPU-bound with the Kepler K20X GPU. **Performance is not much faster (yet) than Fermi X2090** **Need to move spatial hashing, prolongation, interpolation onto the GPU…**	**In progress….** **XK7 nodes 4.3x faster overall**

Preliminary performance for VMD time-averaged electrostatics w/ Multilevel Summation Method on the NCSA Blue Waters Early Science System

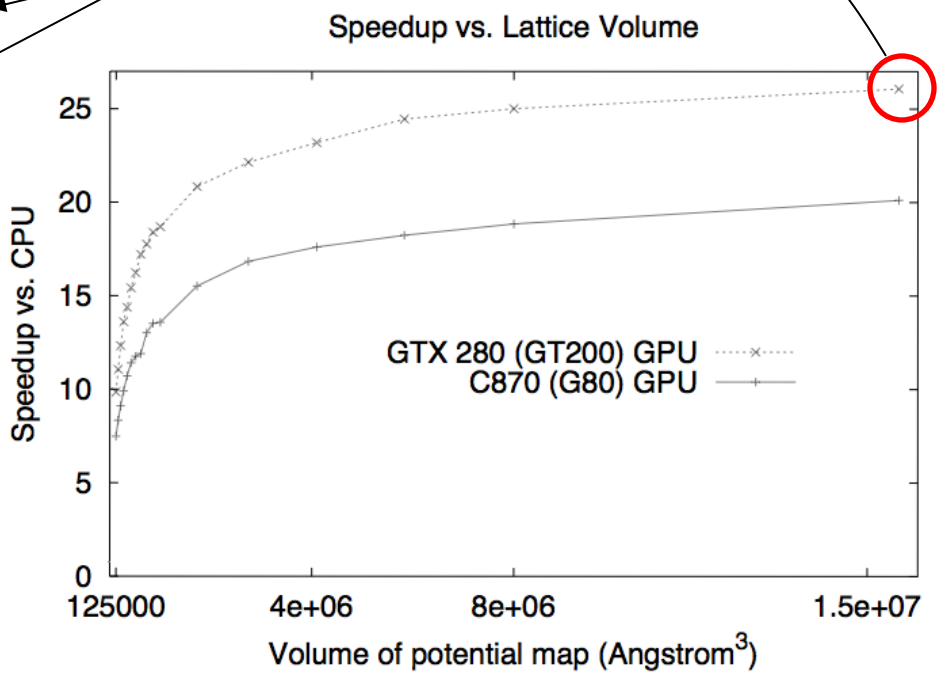
Multilevel Summation on the GPU

Accelerate **short-range cutoff** and **lattice cutoff** parts

Performance profile for 0.5 Å map of potential for 1.5 M atoms.

Hardware platform is Intel QX6700 CPU and NVIDIA GTX 280.

Computational steps	CPU (s)	w/ GPU (s)	Speedup
Short-range cutoff	480.07	14.87	32.3
Long-range anterpolation	0.18		
restriction	0.16		
lattice cutoff	49.47	1.36	36.4
prolongation	0.17		
interpolation	3.47		
Total	533.52	20.21	26.4

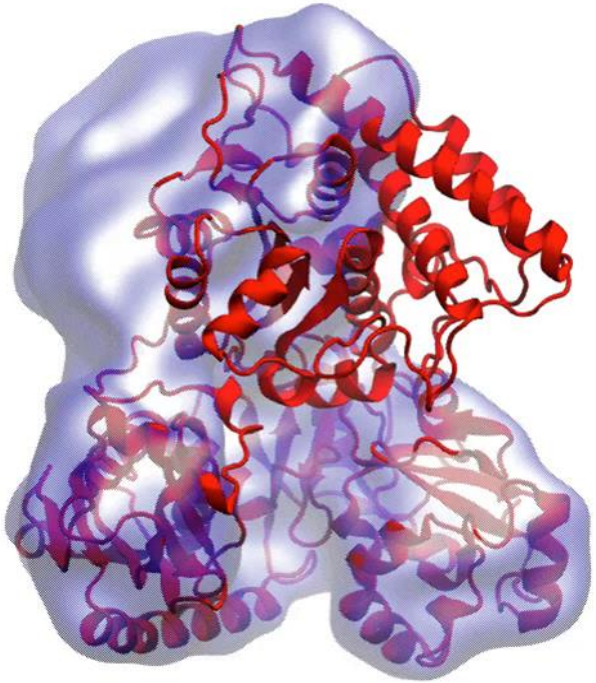


Multilevel summation of electrostatic potentials using graphics processing units.
D. Hardy, J. Stone, K. Schulten. *J. Parallel Computing*, 35:164-177, 2009.

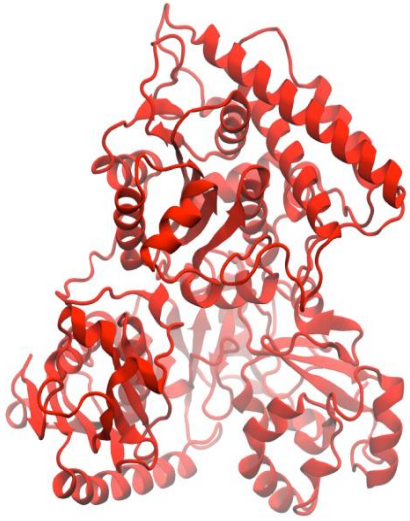
Molecular Dynamics Flexible Fitting (MDFF)

X-ray crystallography → MDFF ← Electron microscopy

APS at Argonne

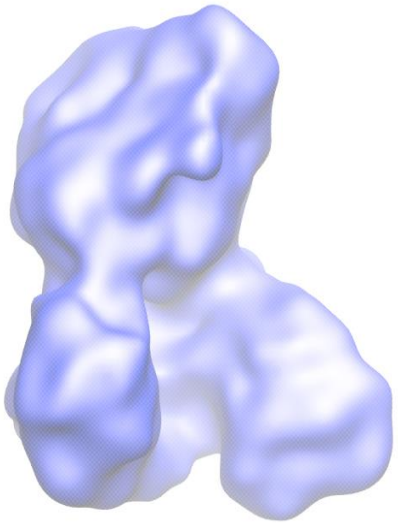


FEI microscope



Acetyl - CoA Synthase

ORNL Titan

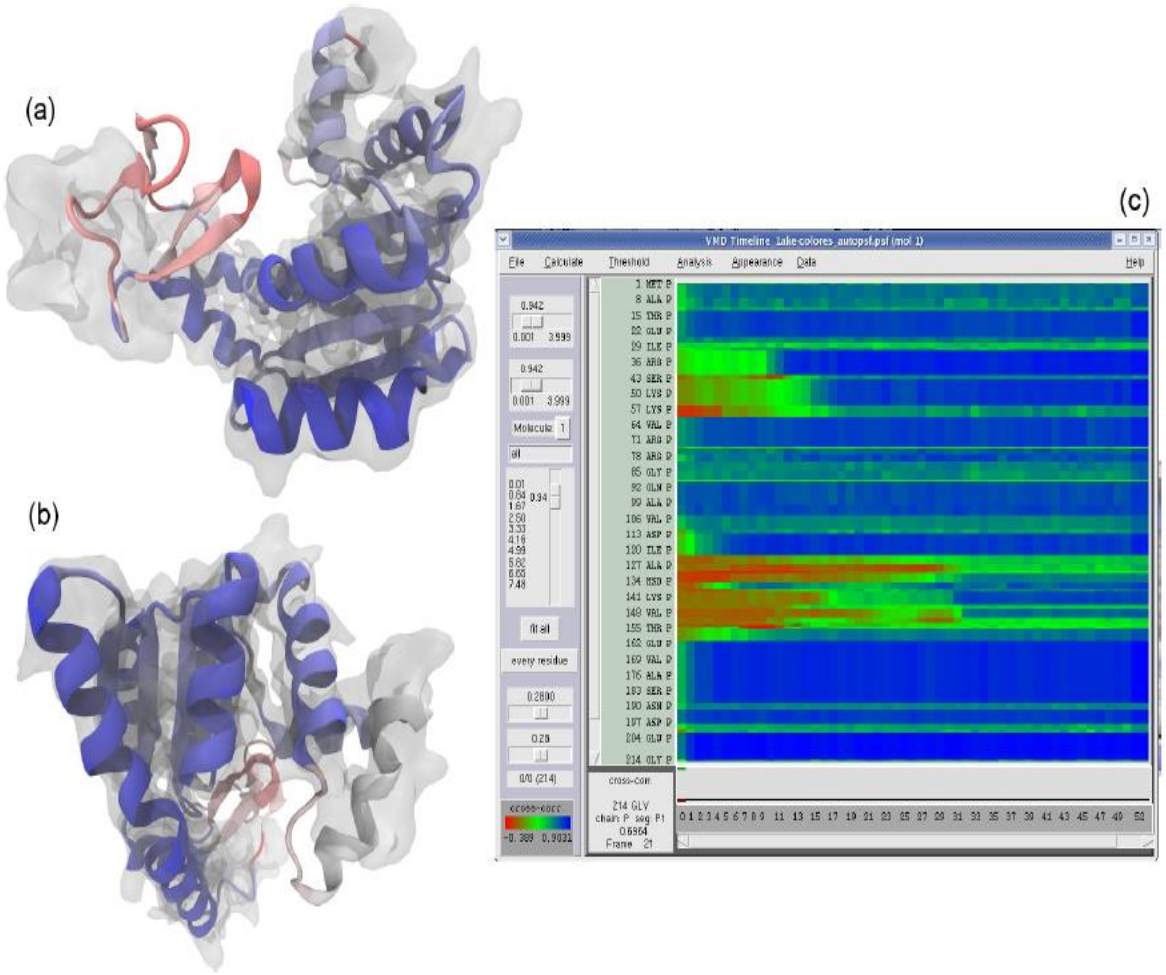


Flexible fitting of atomic structures into electron microscopy maps using molecular dynamics.
L. Trabuco, E. Villa, K. Mitra, J. Frank, and K. Schulten. Structure, 16:673-683, 2008.

GPUs Can Reduce Trajectory Analysis Runtimes from Hours to Minutes

GPUs enable laptops and desktop workstations to handle tasks that would have previously required a cluster, or a very long wait…

GPU-accelerated petascale supercomputers enable analyses that were previously impractical, allowing detailed study of very large structures such as viruses



GPU-accelerated MDFF Cross Correlation Timeline

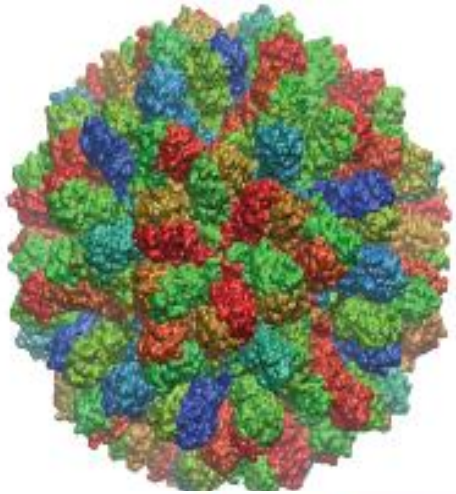
Regions with poor fit **Regions with good fit**

VMD GPU Cross Correlation Performance

	RHDV	Mm-cpn open	GroEL	Aquaporin
Resolution (Å)	**6.5**	**8**	**4**	**3**
Atoms	**702K**	**61K**	**54K**	**1.6K**
VMD-CUDA Quadro K6000	**0.458s 34.6x**	**0.06s 25.7x**	**0.034s 36.8x**	**0.007s 55.7x**
VMD-CPU-SSE 32-threads, 2x Xeon E5-2687W	0.779s 20.3x	0.085s 18.1x	0.159s 7.9x	0.033s 11.8x
Chimera 1-thread Xeon E5-2687W	**15.86s 1.0x**	**1.54s 1.0x**	**1.25s 1.0x**	**0.39s 1.0x**
VMD CPU-SEQ (plugin) 1-thread Xeon E5-2687W	62.89s 0.25x	2.9s 0.53x	1.57s 0.79x	0.04s 9.7x

GPU-accelerated analysis and visualization of large structures solved by molecular dynamics flexible fitting. J. E. Stone, R. McGreevy, B. Isralewitz, and K. Schulten. Faraday Discussion 169, 2014.

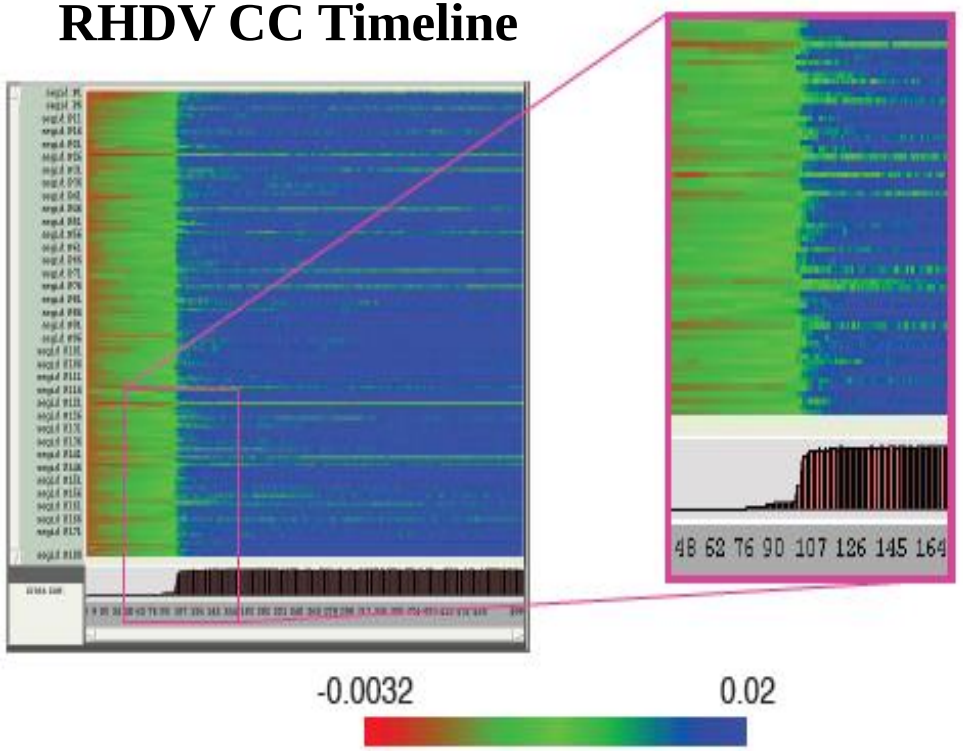
VMD RHDV Cross Correlation Timeline on Cray XK7



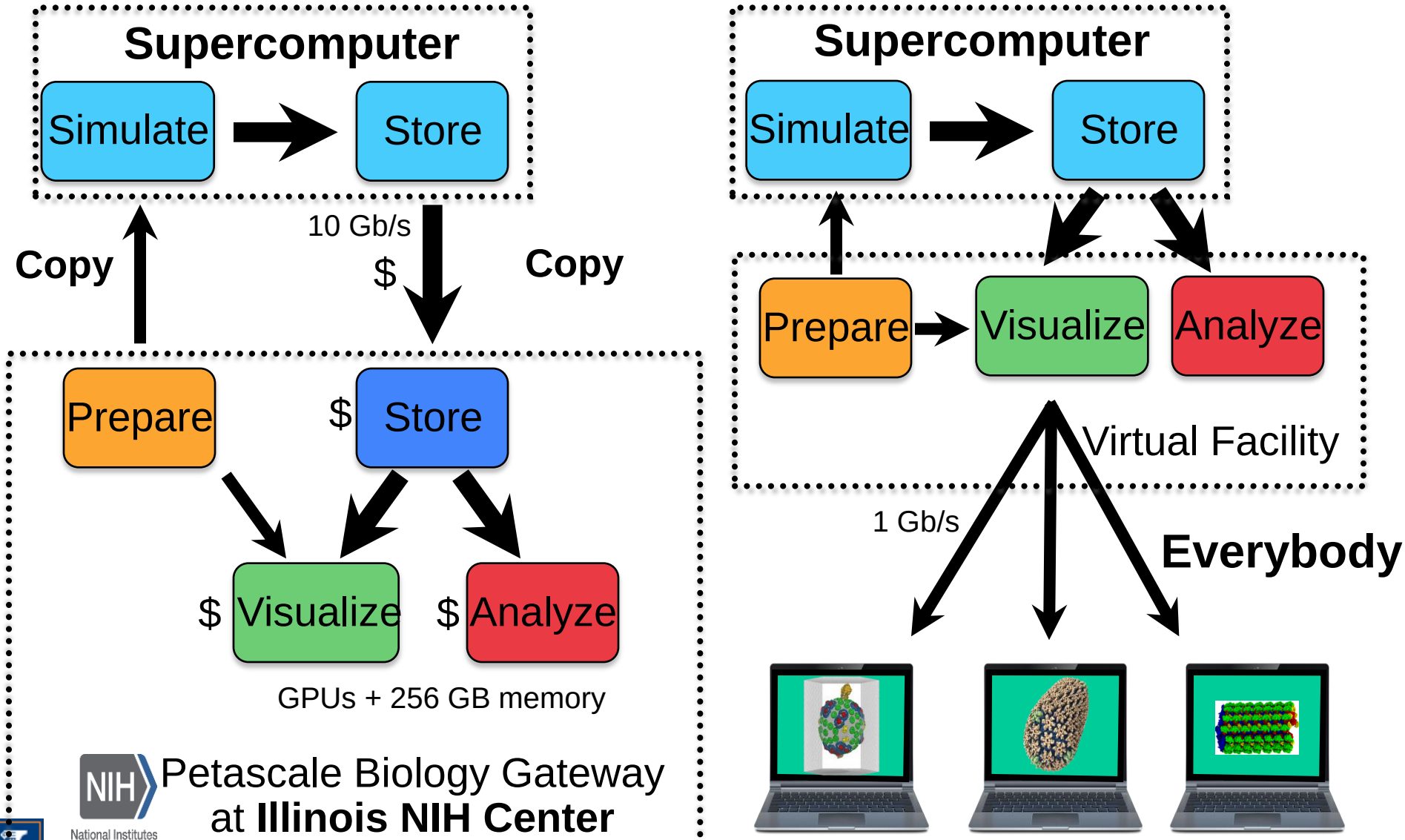
	RHDV
Atoms	**702K**
Traj. Frames	**10,000**
Component Selections	**720**
Single-node XK7 (projected)	**336 hours (14 days)**
128-node XK7	**3.2 hours 105x speedup**
2048-node XK7	**19.5 minutes 1035x speedup**

Calculation would take **5 years** using original serial VMD CC plugin on a workstation!

RHDV CC Timeline

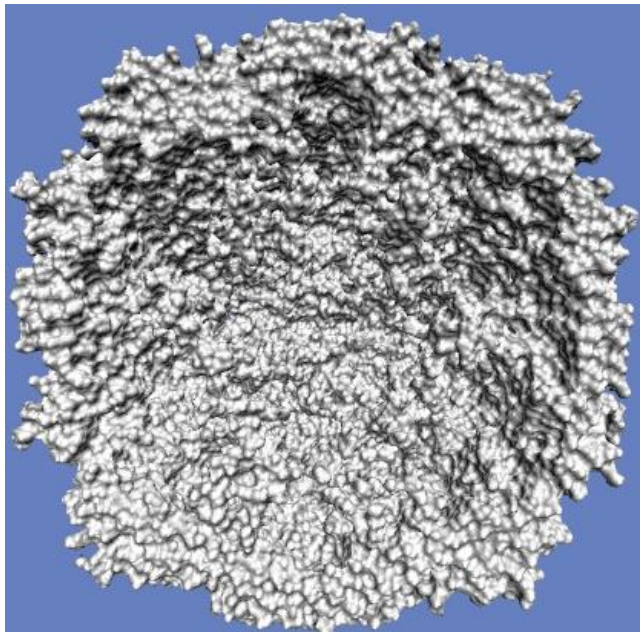


Large Memory Remote Visualization & Analysis Nodes Would Broaden User Base and Accelerate Discovery

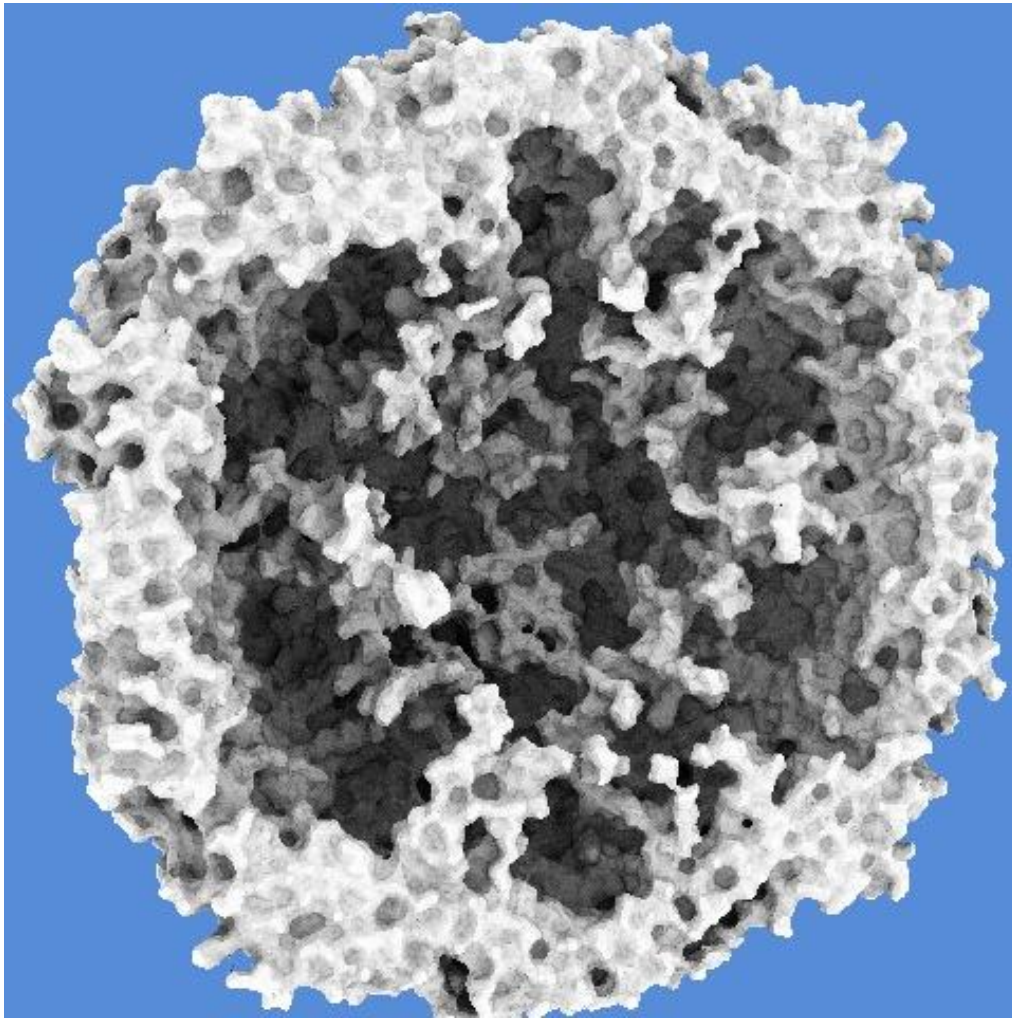


Ray Tracing Molecular Graphics w/ OptiX+CUDA

- Ambient occlusion lighting, shadows, reflections, transparency, and more…
- Satellite tobacco mosaic virus capsid w/ ~75K atoms

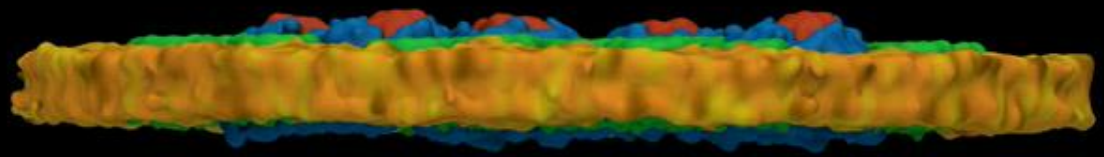


Standard OpenGL rasterization



Prototype VMD/OptiX GPU ray tracing w/ ambient occlusion lighting

BW VMD/Tachyon Movie Generation

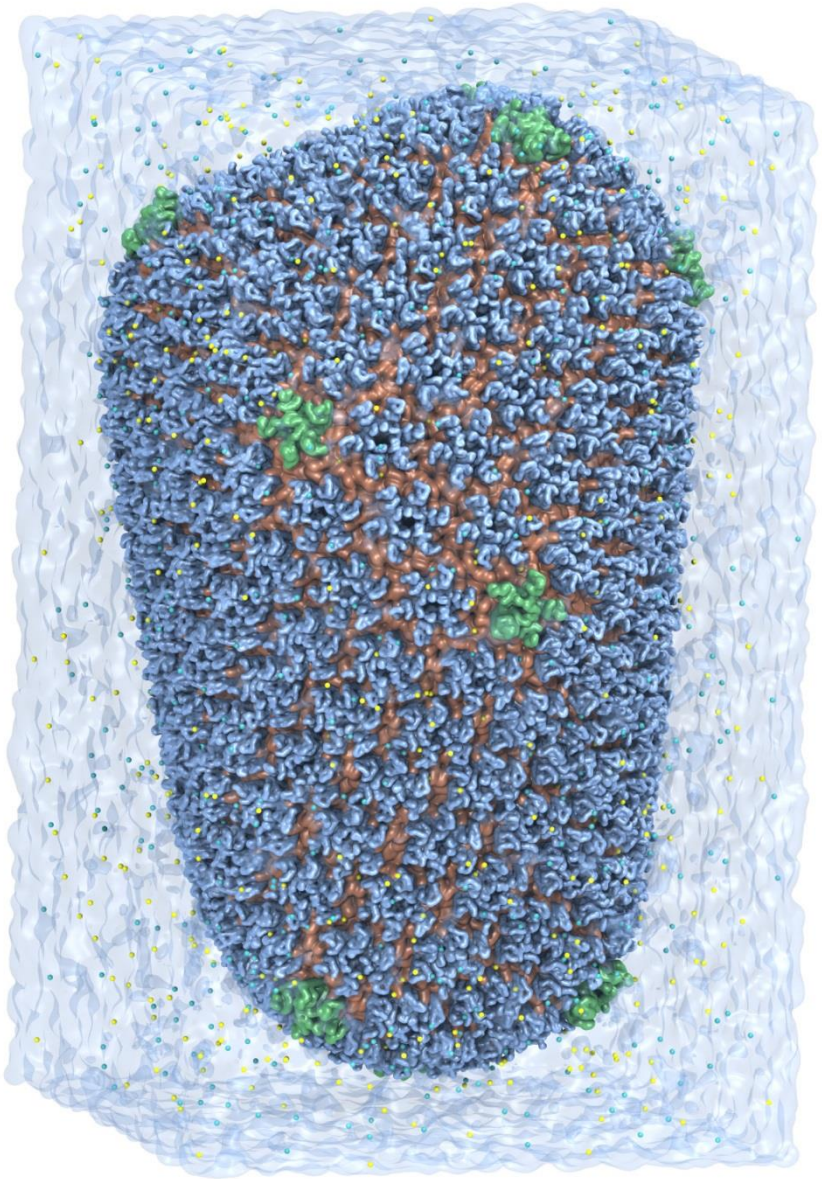


20 M atom chromatophore patch

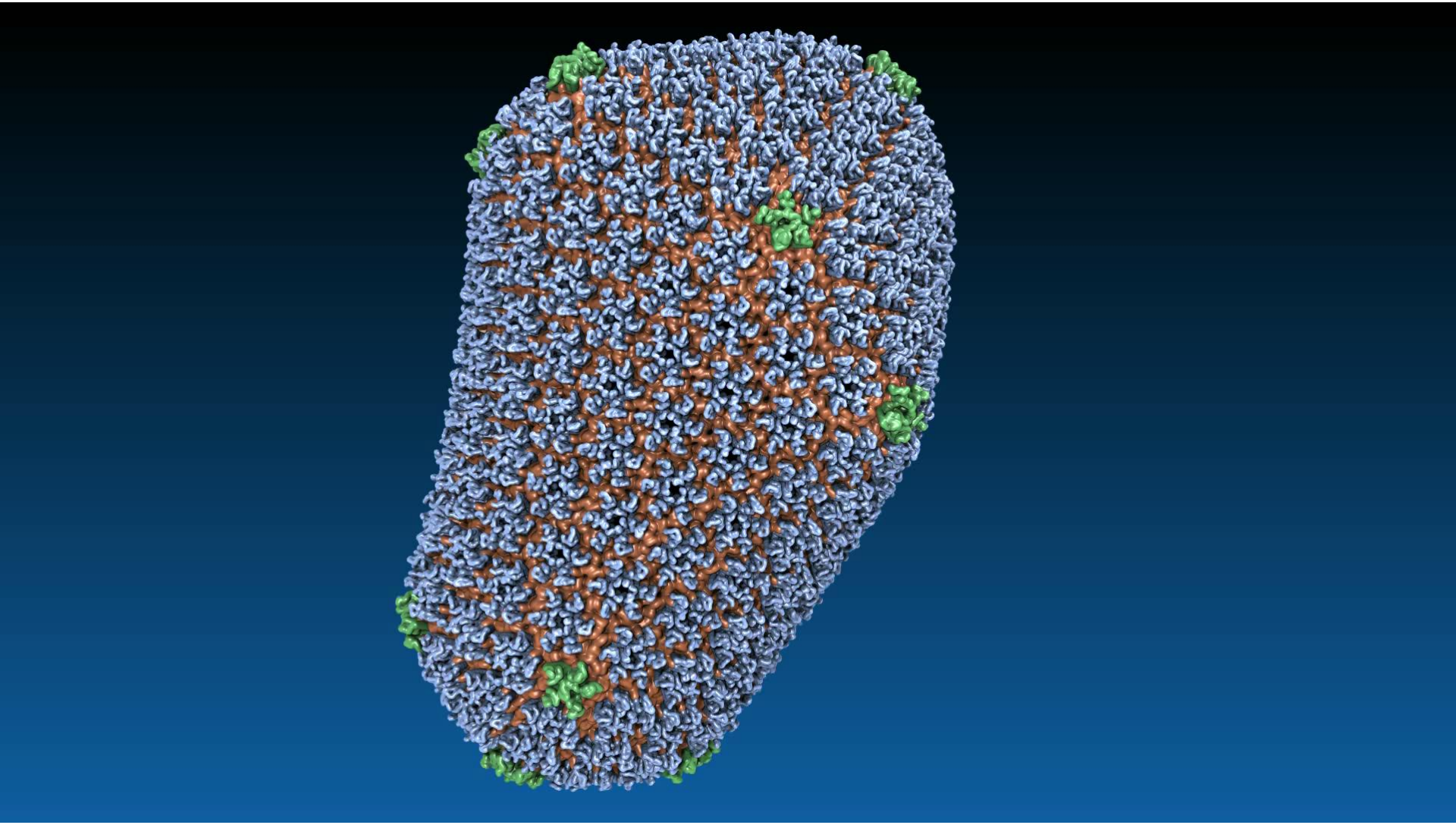
360 XE6 nodes for 3h50m @ 4096x2400

GPU Ray Tracing of HIV-1 on Blue Waters

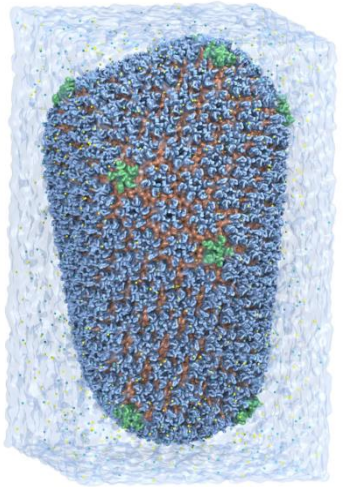
- **Ambient occlusion lighting**, shadows, transparency, antialiasing, depth cueing, **144 rays/pixel minimum**
- 64 million atom virus simulation
- 1000+ movie frames
- Surface generation and ray tracing stages each use >= 75% of GPU memory



VMD GPU Ray Tracing of HIV-1 Capsid



HIV-1 Parallel HD Movie Rendering on Blue Waters Cray XE6/XK7



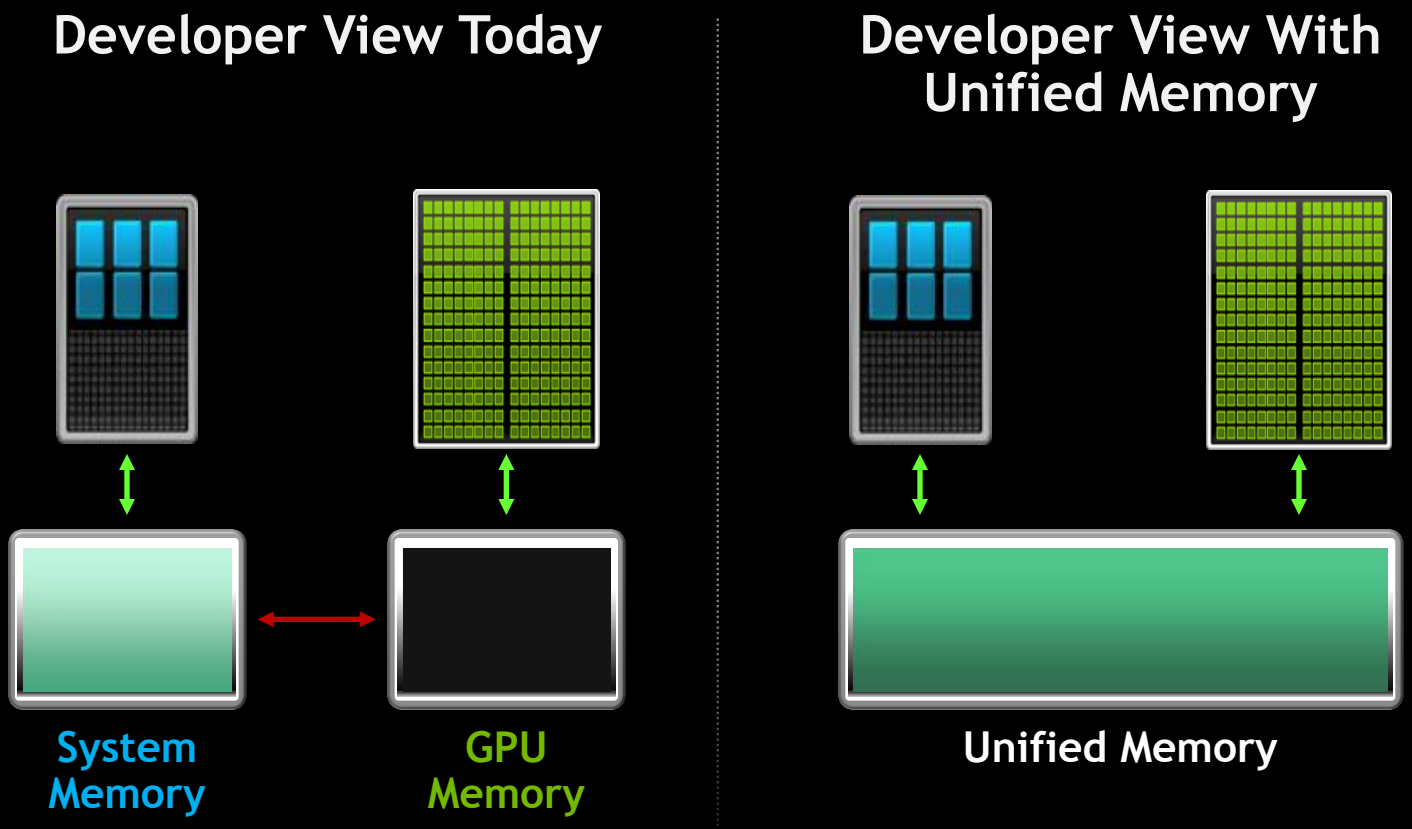
New “TachyonL-OptiX” on XK7 vs. Tachyon on XE6:
K20X GPUs yield **up to eight times** geom+ray tracing speedup

Node Type and Count	**Script Load Time**	**State Load Time**	**Geometry + Ray Tracing**	**Total Time**
256 XE6 CPUs	7 s	160 s	**1,374 s**	**1,541 s**
512 XE6 CPUs	13 s	211 s	808 s	1,032 s
64 XK7 Tesla K20X GPUs	2 s	38 s	655 s	695 s
128 XK7 Tesla K20X GPUs	4 s	74 s	331 s	410 s
256 XK7 Tesla K20X GPUs	7 s	110 s	**171 s**	**288 s**

GPU-Accelerated Molecular Visualization on Petascale Supercomputing Platforms.
Stone et al. In UltraVis'13: Eighth Workshop on Ultrascale Visualization Proceedings, 2013.

Unified Memory

Dramatically Lower Developer Effort



Courtesy NVIDIA

Super Simplified Memory Management Code

CPU Code

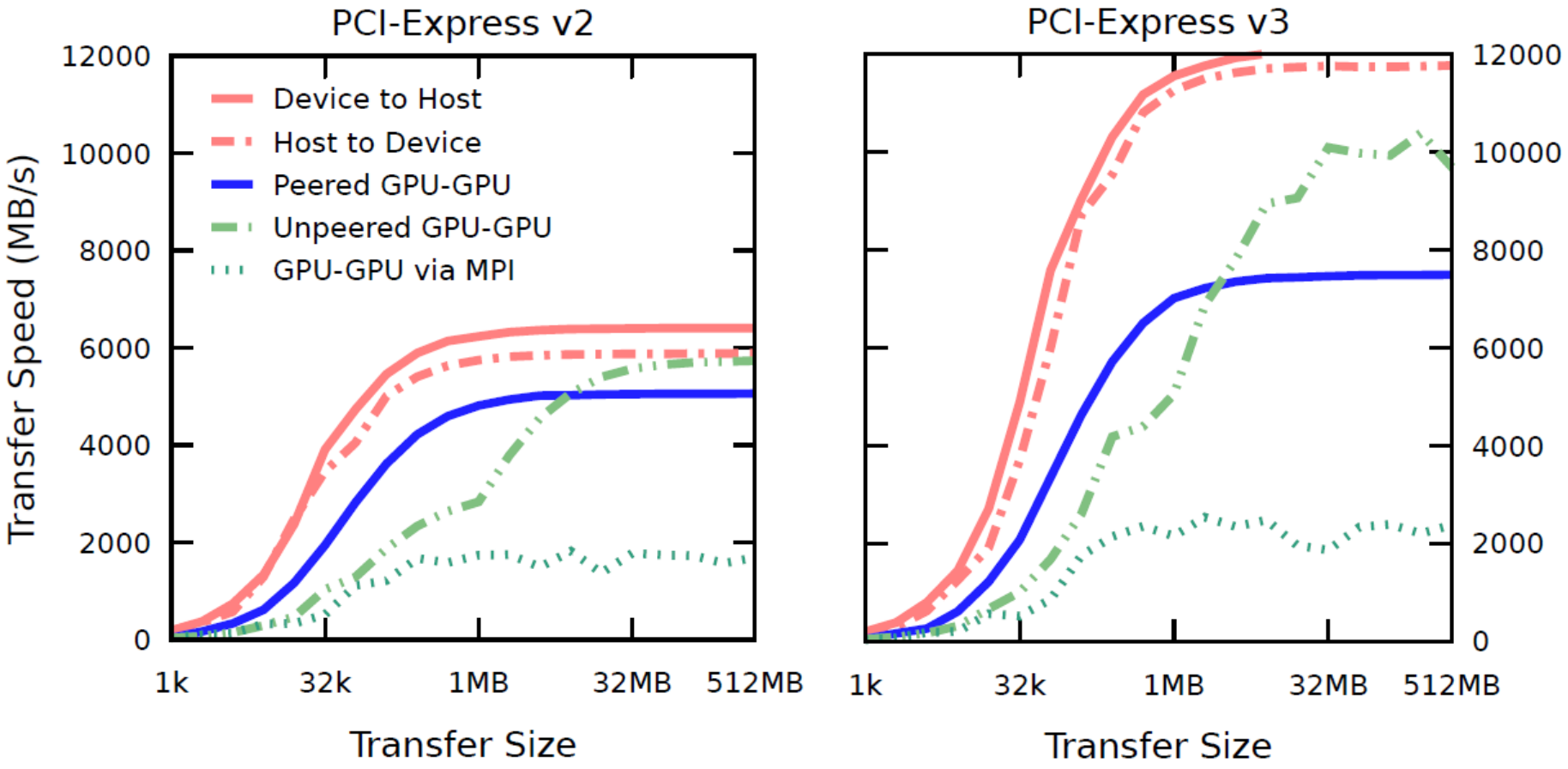
```
void sortfile(FILE *fp, int N) {
    char *data;
    data = (char *)malloc(N);

    fread(data, 1, N, fp);

    qsort(data, N, 1, compare);


    use_data(data);

    free(data);
}
```

CUDA 6 Code with Unified Memory

```
void sortfile(FILE *fp, int N) {
    char *data;
    cudaMallocManaged(&data, N);

    fread(data, 1, N, fp);

    qsort<<<...>>>(data,N,1,compare);
    cudaDeviceSynchronize();

    use_data(data);

    cudaFree(data);
}
```

Courtesy NVIDIA

GPU PCI-Express DMA

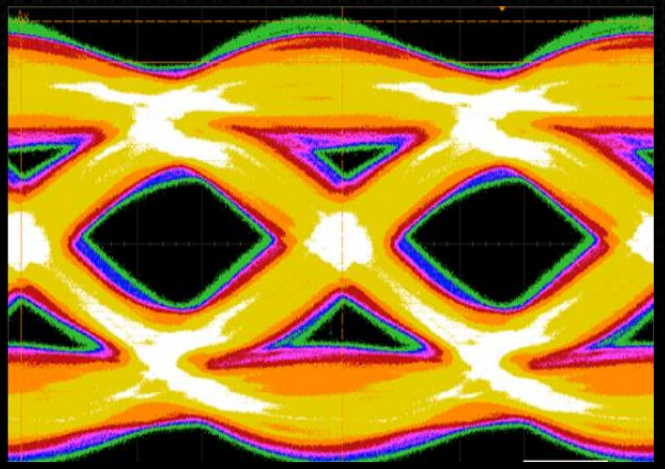


Simulation of reaction diffusion processes over biologically relevant size and time scales using multi-GPU workstations
Michael J. Hallock, John E. Stone, Elijah Roberts, Corey Fry, and Zaida Luthey-Schulten.
Journal of Parallel Computing, 2014. **(In press)**
http://dx.doi.org/10.1016/j.parco.2014.03.009

Future: NVLINK and Stacked Memory

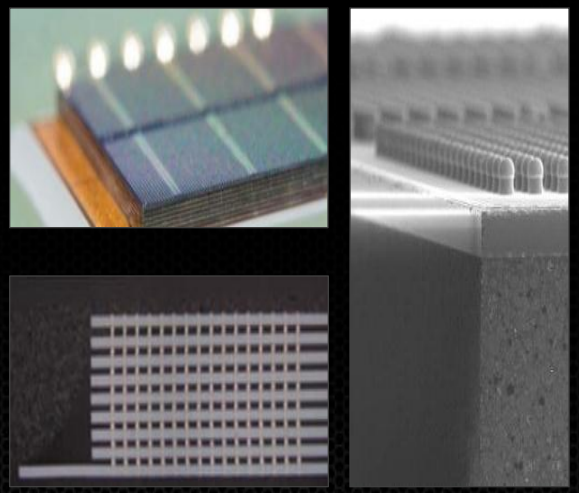
NVLINK

- GPU high speed interconnect
- 80-200 GB/s
- Planned support for POWER CPUs

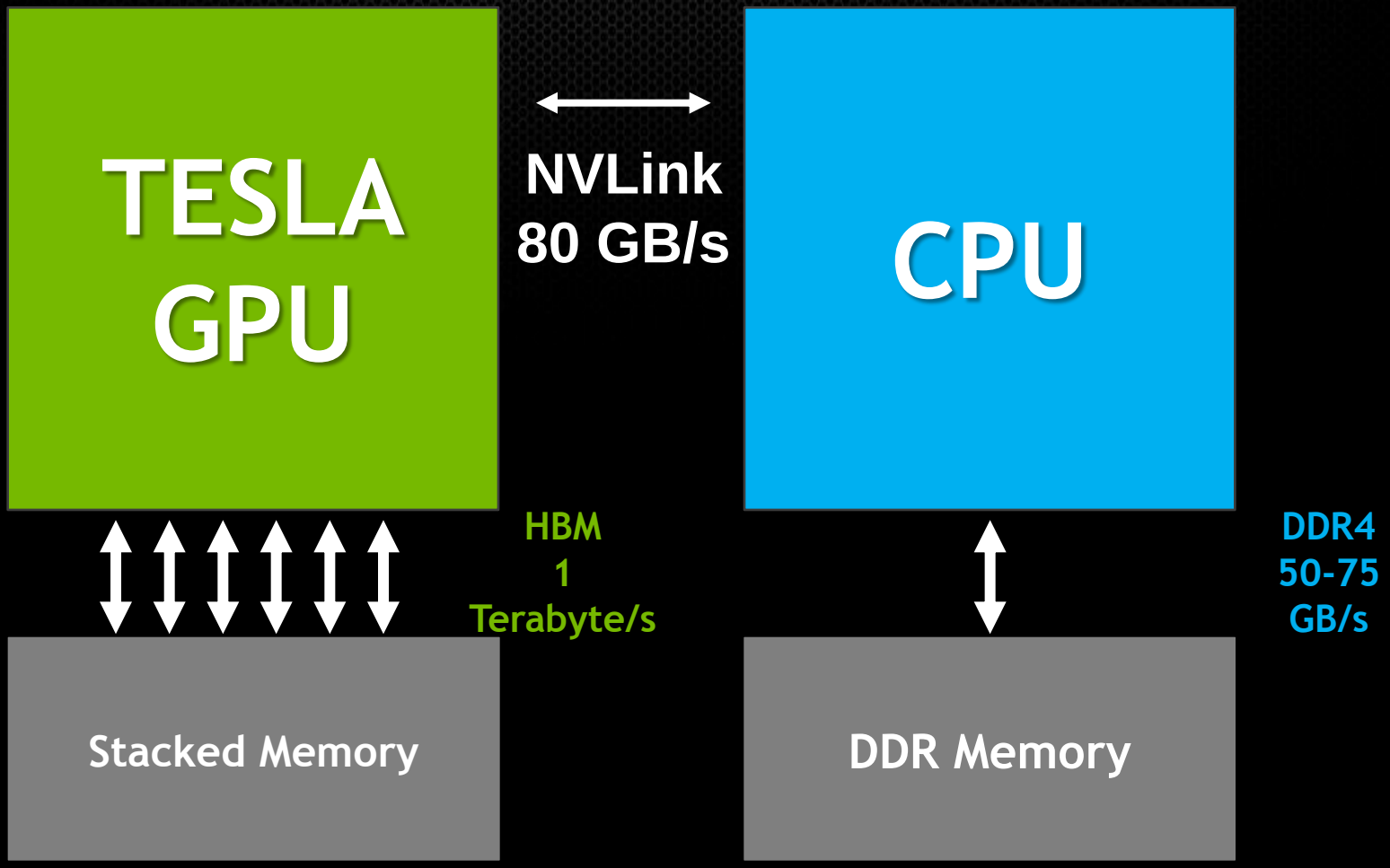


Stacked Memory

- 4x Higher Bandwidth (~1 TB/s)
- 3x Larger Capacity
- 4x More Energy Efficient per bit

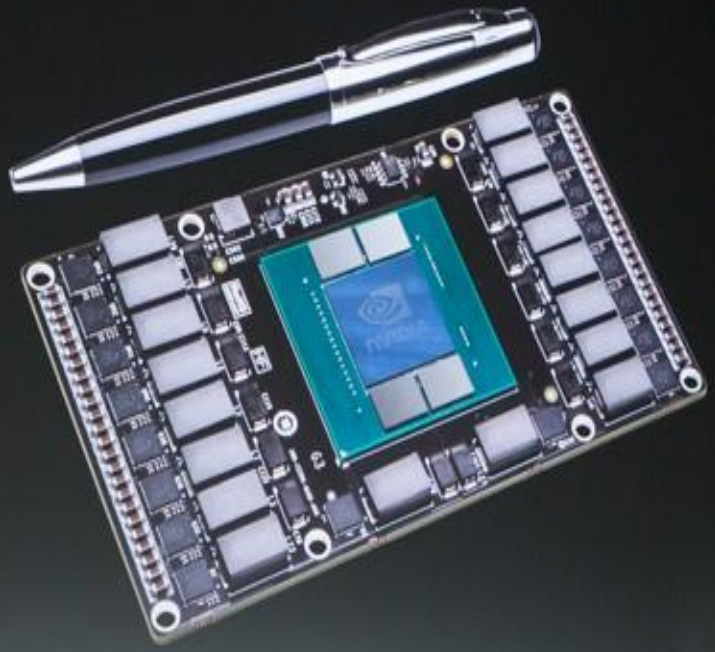


NVLink Enables Data Transfer At Speed of CPU Memory



PASCAL

NVLink	5 to 12X PCIe 3.0
3D Memory	2 to 4X memory BW & size
Module	1/3 size of PCIe card



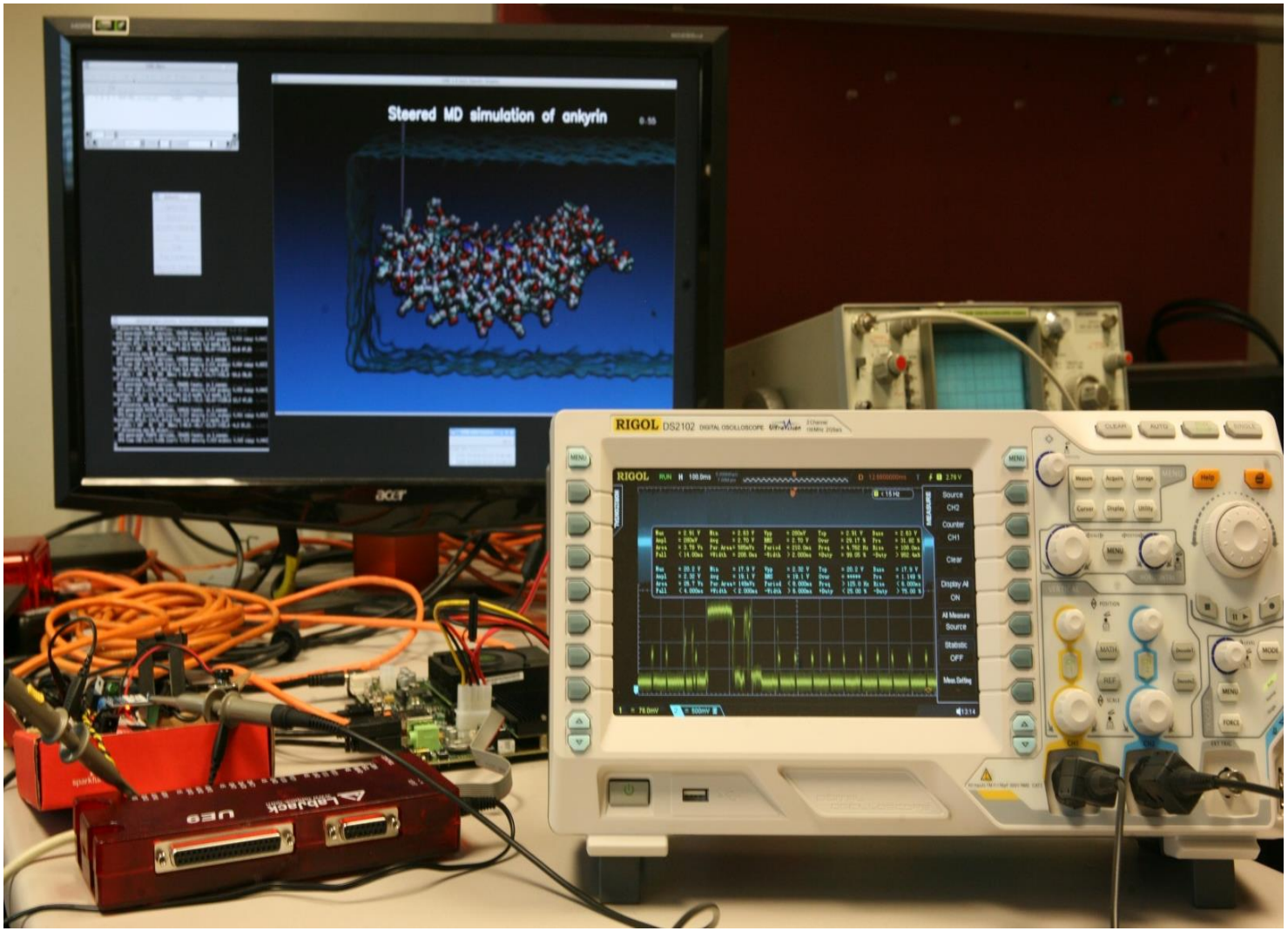
Optimizing GPU Algorithms for Power Consumption

NVIDIA "Carma",

"Kayla",

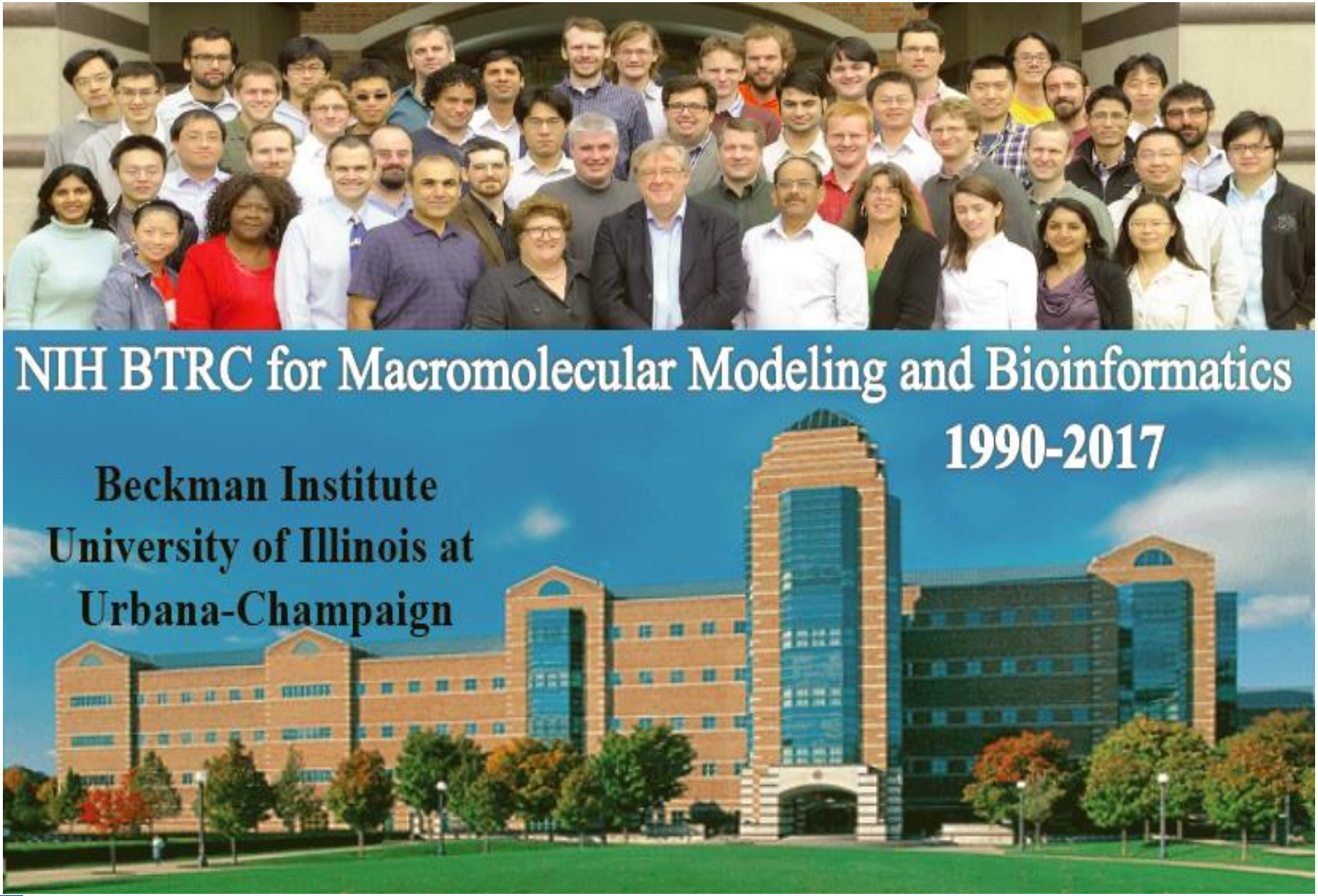
"Jetson" single board computers

Tegra+GPU energy efficiency testbed



Acknowledgements

- Theoretical and Computational Biophysics Group, University of Illinois at Urbana-Champaign
- NVIDIA CUDA Center of Excellence, University of Illinois at Urbana-Champaign
- NVIDIA CUDA team
- NVIDIA OptiX team
- NCSA Blue Waters Team
- Funding:
  - NSF OCI 07-25070
  - NSF PRAC “The Computational Microscope”
  - NIH support: 9P41GM104601, 5R01GM098243-02


NIH BTRC for Macromolecular Modeling and Bioinformatics
1990-2017
Beckman Institute
University of Illinois at
Urbana-Champaign

GPU Computing Publications

http://www.ks.uiuc.edu/Research/gpu/

- **Runtime and Architecture Support for Efficient Data Exchange in Multi-Accelerator Applications** Javier Cabezas, Isaac Gelado, John E. Stone, Nacho Navarro, David B. Kirk, and Wen-mei Hwu. IEEE Transactions on Parallel and Distributed Systems, 2014. (Accepted)
- **Unlocking the Full Potential of the Cray XK7 Accelerator** Mark Klein and John E. Stone. Cray Users Group, 2014. (In press)
- **Simulation of reaction diffusion processes over biologically relevant size and time scales using multi-GPU workstations** Michael J. Hallock, John E. Stone, Elijah Roberts, Corey Fry, and Zaida Luthey-Schulten. Journal of Parallel Computing, 2014. (In press)
- **GPU-Accelerated Analysis and Visualization of Large Structures Solved by Molecular Dynamics Flexible Fitting** John E. Stone, Ryan McGreevy, Barry Isralewitz, and Klaus Schulten. Faraday Discussion 169, 2014. (In press)
- **GPU-Accelerated Molecular Visualization on Petascale Supercomputing Platforms.** J. Stone, K. L. Vandivort, and K. Schulten. UltraVis'13: Proceedings of the 8th International Workshop on Ultrascale Visualization, pp. 6:1-6:8, 2013.
- **Early Experiences Scaling VMD Molecular Visualization and Analysis Jobs on Blue Waters.** J. E. Stone, B. Isralewitz, and K. Schulten. In proceedings, Extreme Scaling Workshop, 2013.
- **Lattice Microbes: High-performance stochastic simulation method for the reaction-diffusion master equation.** E. Roberts, J. E. Stone, and Z. Luthey-Schulten. J. Computational Chemistry 34 (3), 245-255, 2013**.**

GPU Computing Publications

http://www.ks.uiuc.edu/Research/gpu/

- **Fast Visualization of Gaussian Density Surfaces for Molecular Dynamics and Particle System Trajectories.** M. Krone, J. E. Stone, T. Ertl, and K. Schulten. *EuroVis Short Papers*, pp. 67-71, 2012.
- **Fast Analysis of Molecular Dynamics Trajectories with Graphics Processing Units – Radial Distribution Functions.** B. Levine, J. Stone, and A. Kohlmeyer. *J. Comp. Physics*, 230(9):3556-3569, 2011.
- **Immersive Out-of-Core Visualization of Large-Size and Long-Timescale Molecular Dynamics Trajectories.** J. Stone, K. Vandivort, and K. Schulten. G. Bebis et al. (Eds.): *7th International Symposium on Visual Computing (ISVC 2011)*, LNCS 6939, pp. 1-12, 2011.
- **Quantifying the Impact of GPUs on Performance and Energy Efficiency in HPC Clusters.** J. Enos, C. Steffen, J. Fullop, M. Showerman, G. Shi, K. Esler, V. Kindratenko, J. Stone, J Phillips. *International Conference on Green Computing*, pp. 317-324, 2010.
- **GPU-accelerated molecular modeling coming of age. J. Stone, D. Hardy, I. Ufimtsev, K. Schulten.** *J. Molecular Graphics and Modeling*, 29:116-125, 2010.
- **OpenCL: A Parallel Programming Standard for Heterogeneous Computing. J. Stone, D. Gohara, G. Shi.** *Computing in Science and Engineering*, 12(3):66-73, 2010.

GPU Computing Publications
http://www.ks.uiuc.edu/Research/gpu/

- **An Asymmetric Distributed Shared Memory Model for Heterogeneous Computing Systems**. I. Gelado, J. Stone, J. Cabezas, S. Patel, N. Navarro, W. Hwu. *ASPLOS '10: Proceedings of the 15th International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 347-358, 2010.
- **GPU Clusters for High Performance Computing**. V. Kindratenko, J. Enos, G. Shi, M. Showerman, G. Arnold, J. Stone, J. Phillips, W. Hwu. *Workshop on Parallel Programming on Accelerator Clusters (PPAC)*, In Proceedings IEEE Cluster 2009, pp. 1-8, Aug. 2009.
- **Long time-scale simulations of in vivo diffusion using GPU hardware**. E. Roberts, J. Stone, L. Sepulveda, W. Hwu, Z. Luthey-Schulten. In *IPDPS'09: Proceedings of the 2009 IEEE International Symposium on Parallel & Distributed Computing*, pp. 1-8, 2009.
- **High Performance Computation and Interactive Display of Molecular Orbitals on GPUs and Multi-core CPUs**. J. Stone, J. Saam, D. Hardy, K. Vandivort, W. Hwu, K. Schulten, *2nd Workshop on General-Purpose Computation on Graphics Pricessing Units (GPGPU-2), ACM International Conference Proceeding Series*, volume 383, pp. 9-18, 2009.
- **Probing Biomolecular Machines with Graphics Processors**. J. Phillips, J. Stone. *Communications of the ACM*, 52(10):34-41, 2009.
- **Multilevel summation of electrostatic potentials using graphics processing units**. D. Hardy, J. Stone, K. Schulten. *J. Parallel Computing*, 35:164-177, 2009.

GPU Computing Publications

http://www.ks.uiuc.edu/Research/gpu/

- **Adapting a message-driven parallel application to GPU-accelerated clusters**. J. Phillips, J. Stone, K. Schulten. *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*, IEEE Press, 2008.
- **GPU acceleration of cutoff pair potentials for molecular modeling applications**. C. Rodrigues, D. Hardy, J. Stone, K. Schulten, and W. Hwu. *Proceedings of the 2008 Conference On Computing Frontiers*, pp. 273-282, 2008.
- **GPU computing**. J. Owens, M. Houston, D. Luebke, S. Green, J. Stone, J. Phillips. *Proceedings of the IEEE*, 96:879-899, 2008.
- **Accelerating molecular modeling applications with graphics processors**. J. Stone, J. Phillips, P. Freddolino, D. Hardy, L. Trabuco, K. Schulten. *J. Comp. Chem.*, 28:2618-2640, 2007.
- **Continuous fluorescence microphotolysis and correlation spectroscopy**. A. Arkhipov, J. Hüve, M. Kahms, R. Peters, K. Schulten. *Biophysical Journal*, 93:4006-4017, 2007.