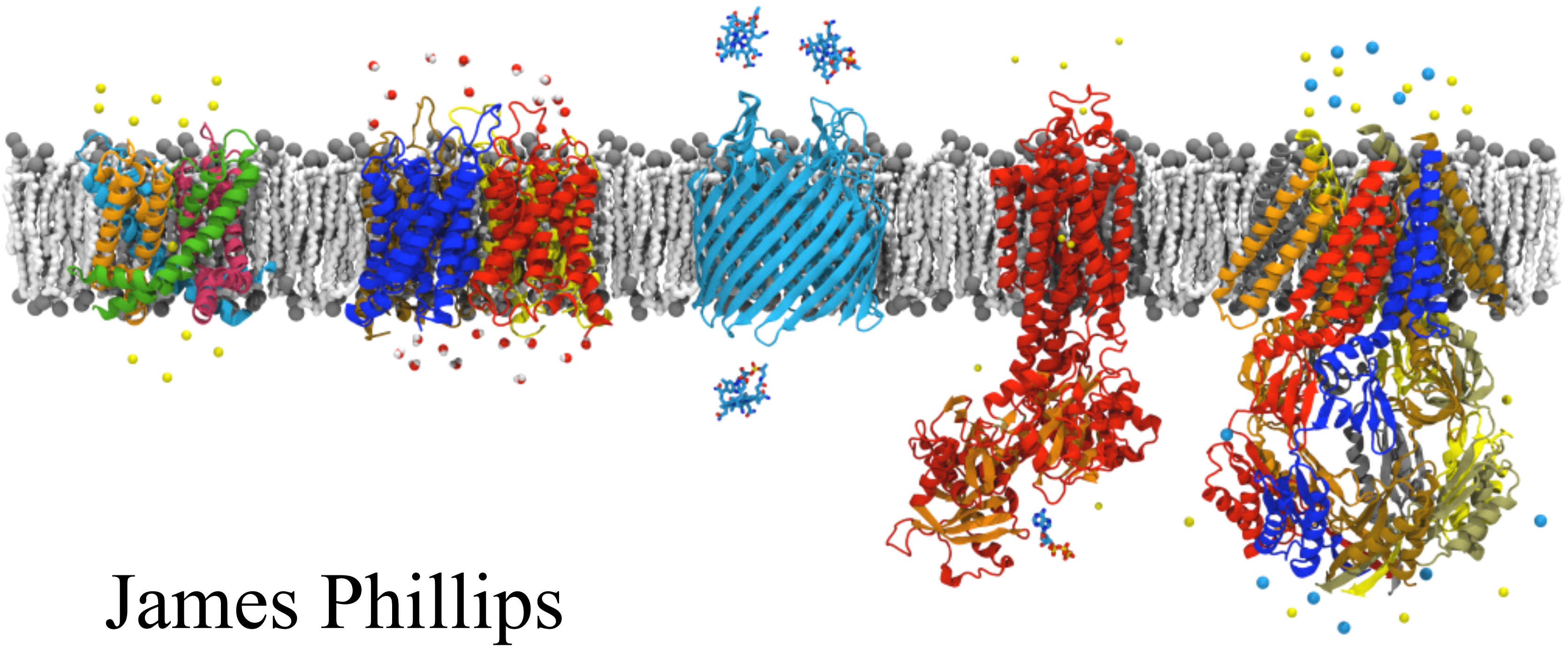
Graphics Processor Acceleration and YOU



James Phillips

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

Goals of Lecture

After this talk the audience will:

- Understand how GPUs differ from CPUs
- Understand the limits of GPU acceleration
- Have knowledge for equipment purchases
- Not hate the speaker for delaying lunch

NAMD: Practical Supercomputing

- 30,000 users can't all be computer experts.
  - 18% are NIH-funded; many in other countries.
  - 5600 have downloaded more than one version.

- User experience is the same on all platforms.
  - No change in input, output, or configuration files.
  - Run any simulation on **any number of processors**.
  - Precompiled binaries available when possible.
- Desktops and laptops – setup and testing
  - x86 and x86-64 Windows, and Macintosh
  - Allow both shared-memory and network-based parallelism.

- Linux clusters – affordable workhorses
  - x86, x86-64, and Itanium processors
  - Gigabit ethernet, Myrinet, InfiniBand, Quadrics, Altix, etc

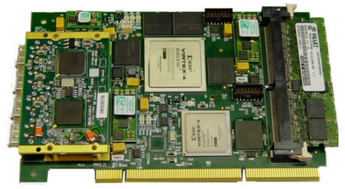
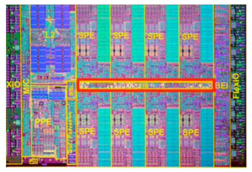
Phillips *et al.*, *J. Comp. Chem.* **26**:1781-1802, 2005.

Our Goal: Practical Acceleration

- Broadly applicable to scientific computing
  - Programmable by domain scientists
  - Scalable from small to large machines
- Broadly available to researchers
  - Price driven by commodity market
  - Low burden on system administration
- Sustainable performance advantage
  - Performance driven by Moore’s law
  - Stable market and supply chain

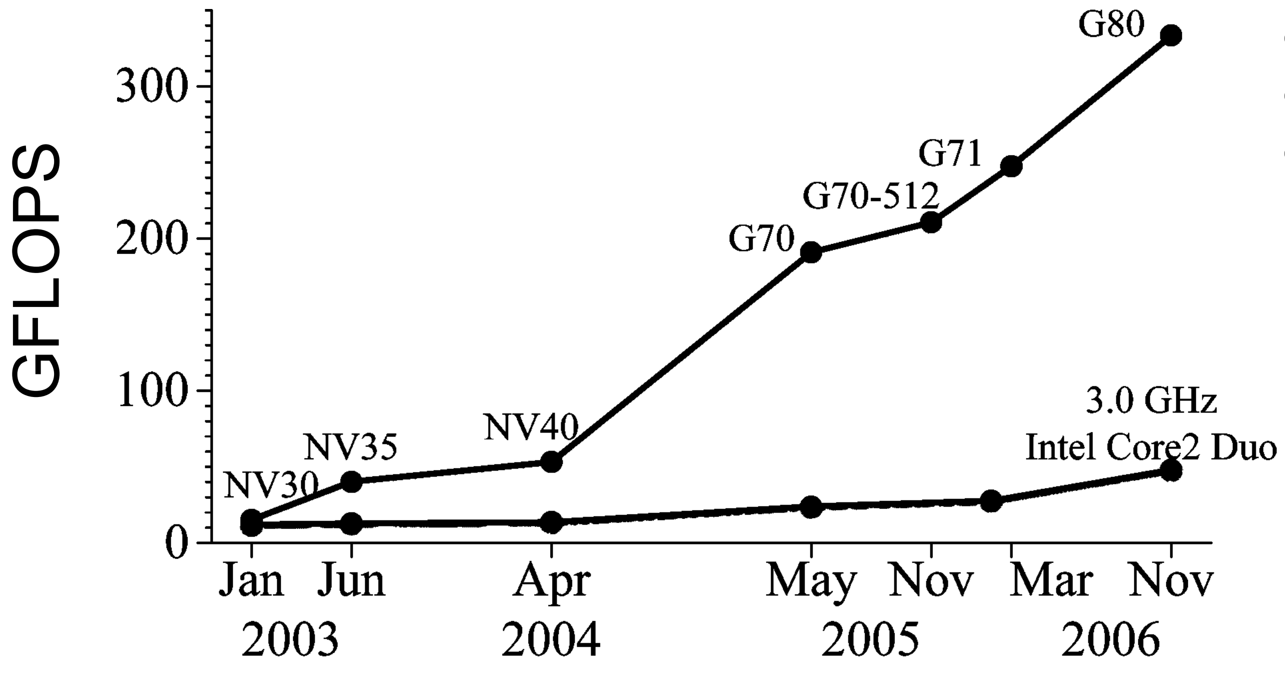
Acceleration Options for NAMD

- Outlook in 2005-2006:
  - FPGA reconfigurable computing (with NCSA)
    - Difficult to program, slow floating point, expensive
  - Cell processor (NCSA hardware)
    - Relatively easy to program, expensive
  - ClearSpeed (direct contact with company)
    - Limited memory and memory bandwidth, expensive
  - MDGRAPE
    - Inflexible and expensive
  - Graphics processor (GPU)
    - Program must be expressed as graphics operations

GPU vs CPU: Raw Performance

– Calculation: 450 GFLOPS vs 32 GFLOPS
– Memory Bandwidth: 80 GB/s vs 8.4 GB/s



G80 = GeForce 8800 GTX
G71 = GeForce 7900 GTX
G70 = GeForce 7800 GTX
NV40 = GeForce 6800 Ultra
NV35 = GeForce FX 5950 Ultra
NV30 = GeForce FX 5800

CUDA: Practical Performance

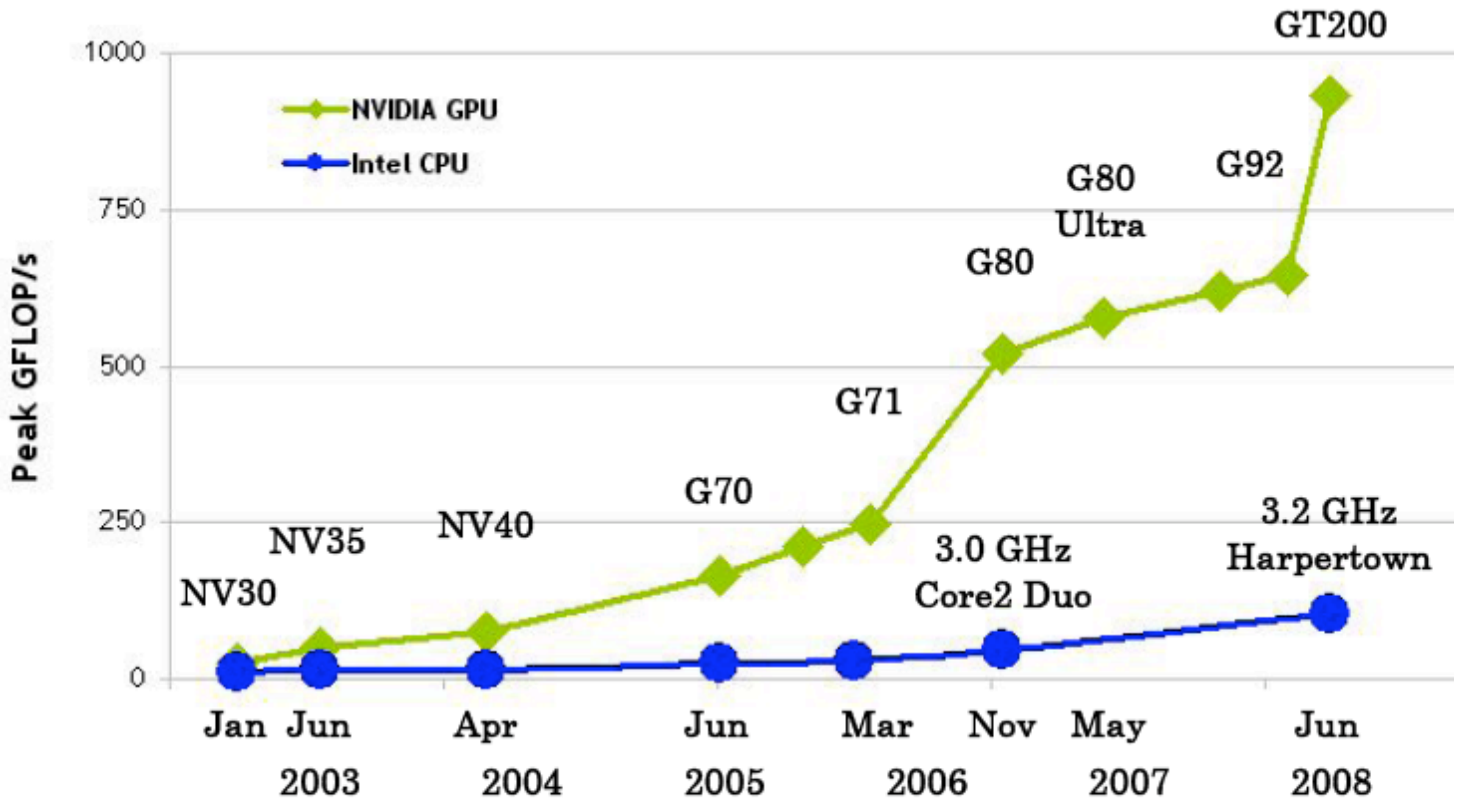
November 2006: NVIDIA announces CUDA for G80 GPU.

- CUDA makes GPU acceleration usable:
  - Developed and supported by NVIDIA.
  - No masquerading as graphics rendering.
  - New shared memory and synchronization.
  - No OpenGL or display device hassles.
  - Multiple processes per card (or vice versa).
- Resource and collaborators make it useful:
  - Experience from VMD development
  - David Kirk (Chief Scientist, NVIDIA)
  - Wen-mei Hwu (ECE Professor, UIUC)

Fun to program (and drive)

Stone *et al.*, *J. Comp. Chem.* **28**:2618-2640, 2007.

Peak Single-precision Arithmetic Performance Trend

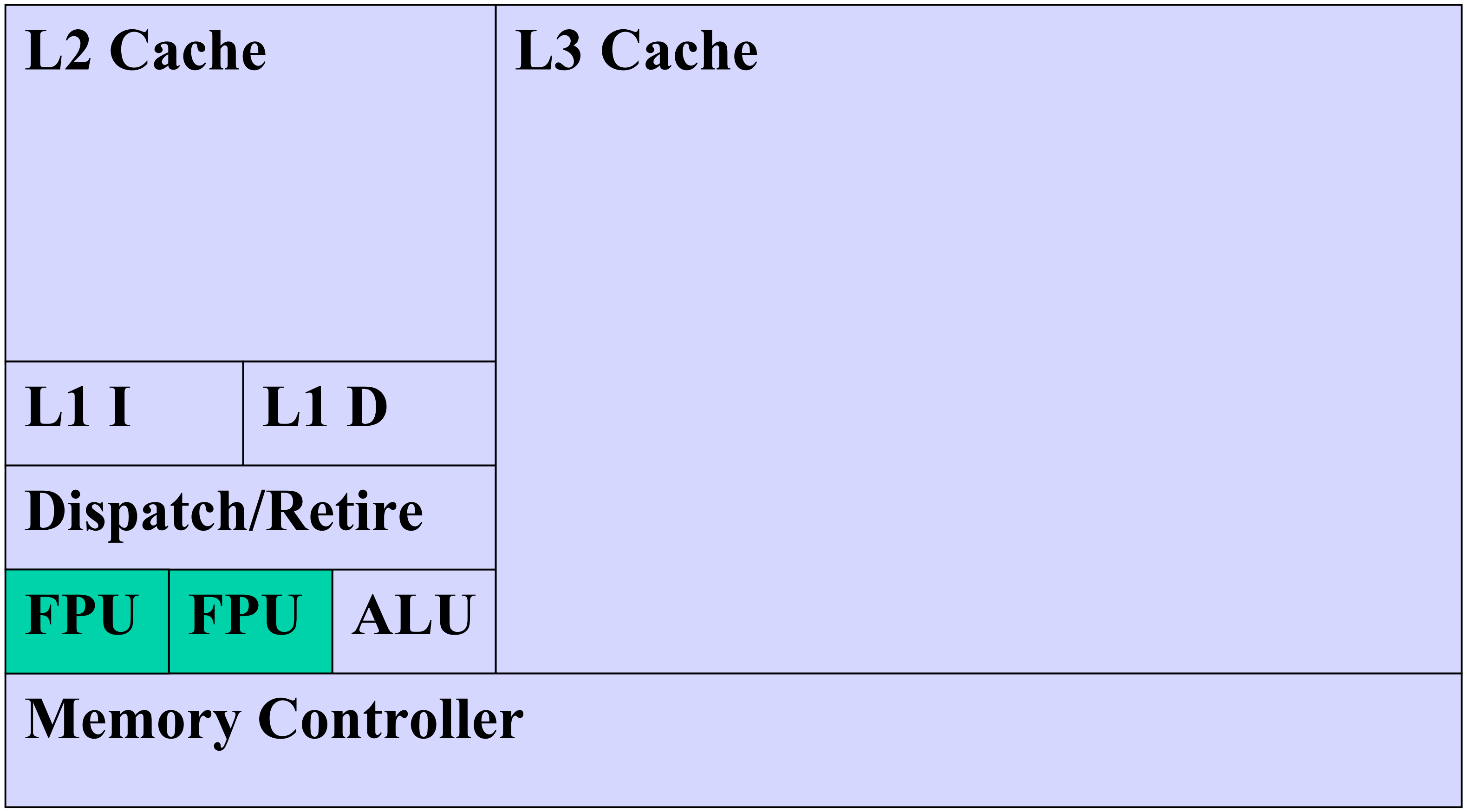


How can a GPU attain such impressive performance?

Strong Technical Language Advisory

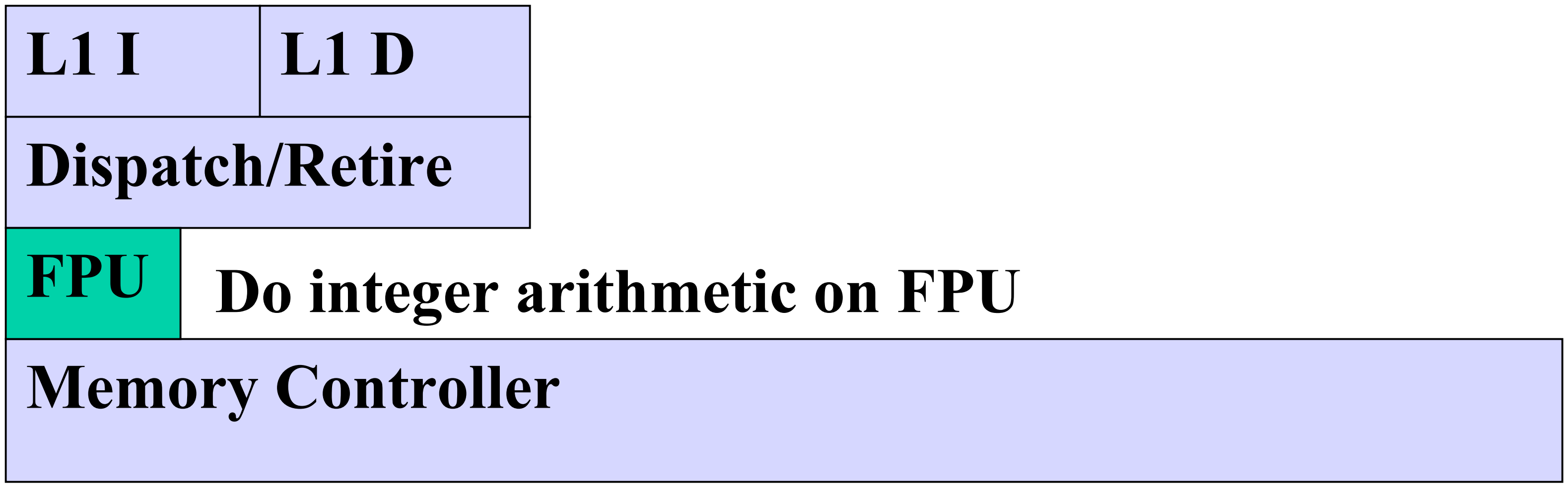
The following slides contain explicit technical content that may not be suitable for all scientists. Audience discretion is advised (feel free to check your email).

Typical CPU Architecture



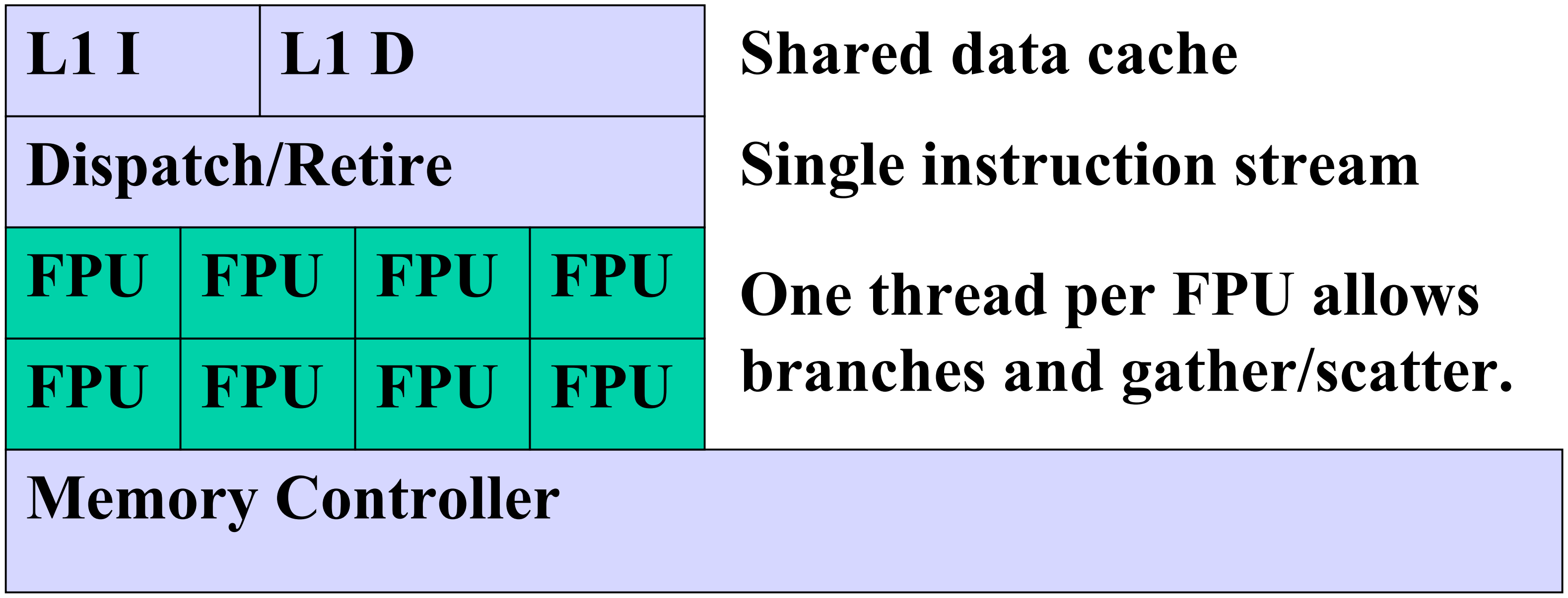
Minimize the Processor

No large caches or multiple execution units

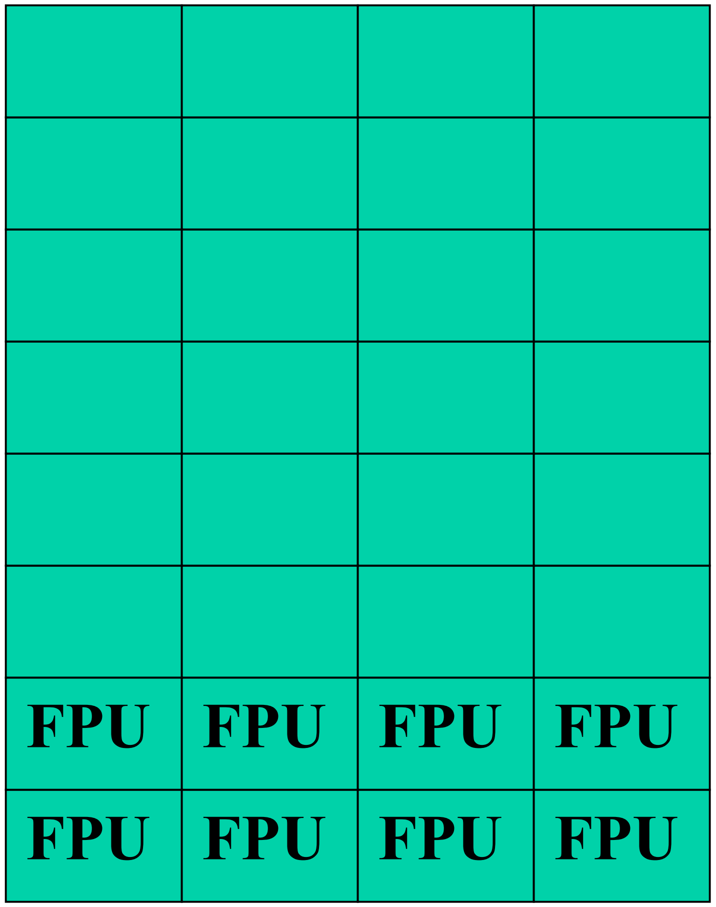


Maximize Floating Point

8 FP pipelines per SIMD unit



Add More Threads



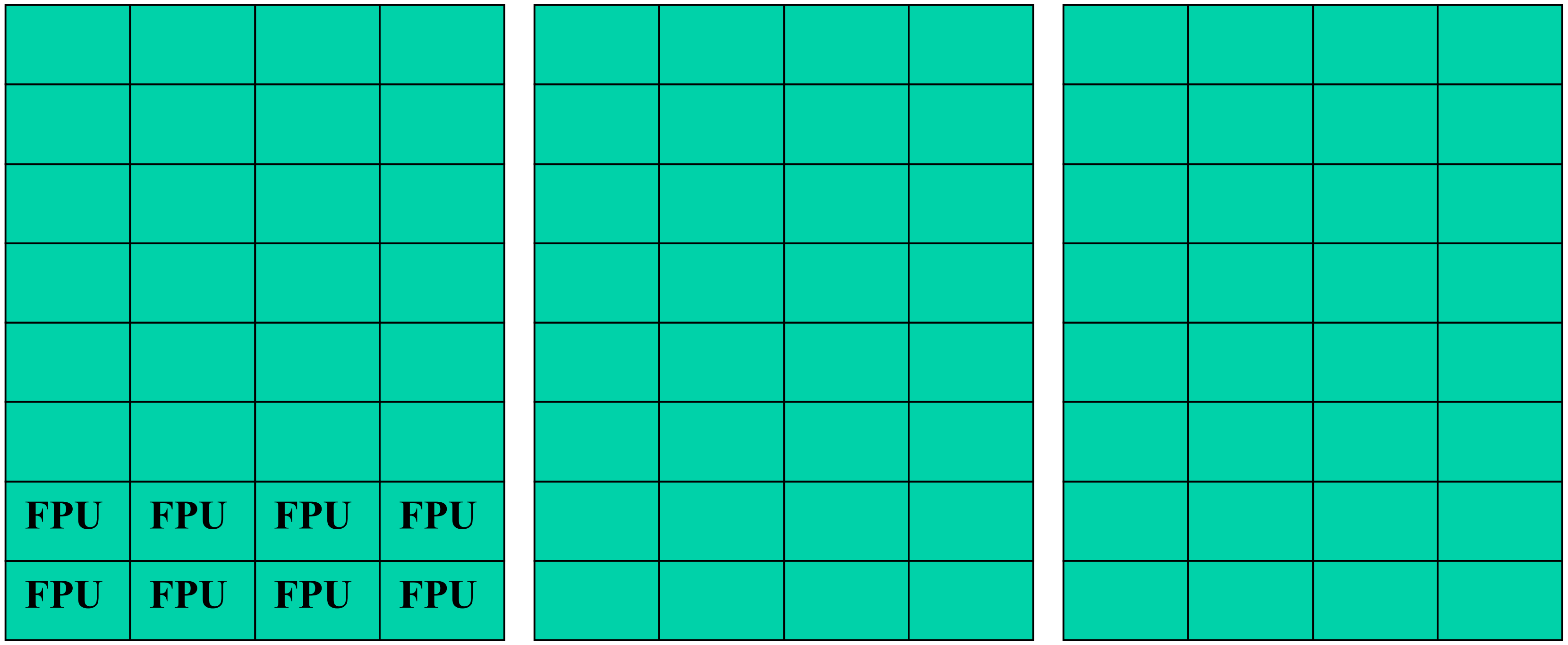
Pipeline 4 threads per FPU to hide 4-cycle instruction latency.

All 32 threads in a "warp" execute the same instruction.

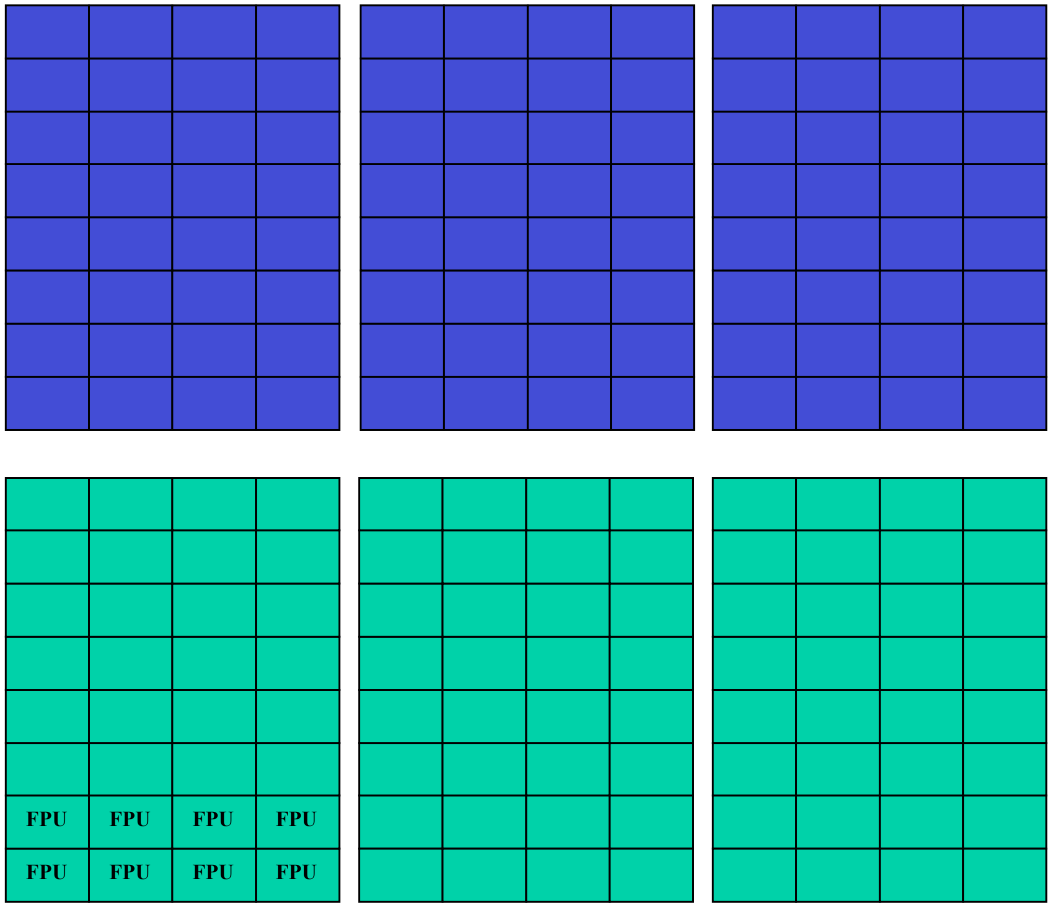
Divergent branches allowed through predication.

Add Even More Threads

Multiple warps in a "block" hide main memory latency and can synchronize to share data.



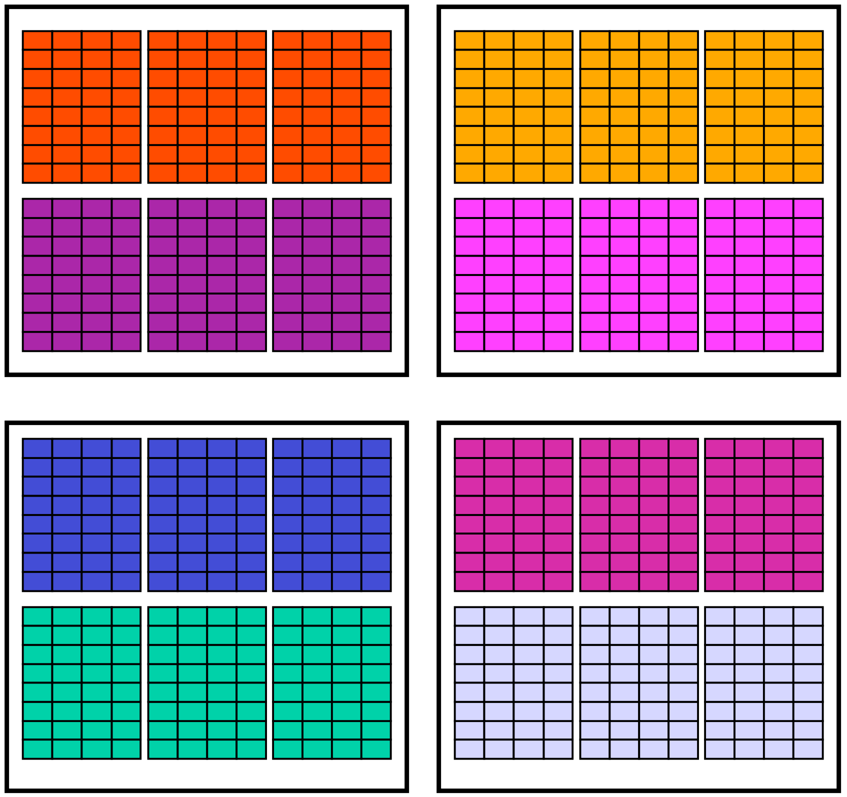
Add More Threads Again



Multiple blocks on a single multiprocessor hide both memory and synchronization latency.

All blocks execute a “kernel” function independently without synchronization or memory coherency.

Add Cores to Suit Customer



Kernel is invoked on a "grid" of uniform blocks.

Blocks are dynamically assigned to available multiprocessors and run to completion.

Synchronization occurs when all blocks complete.

Support Fine-Grained Parallelism

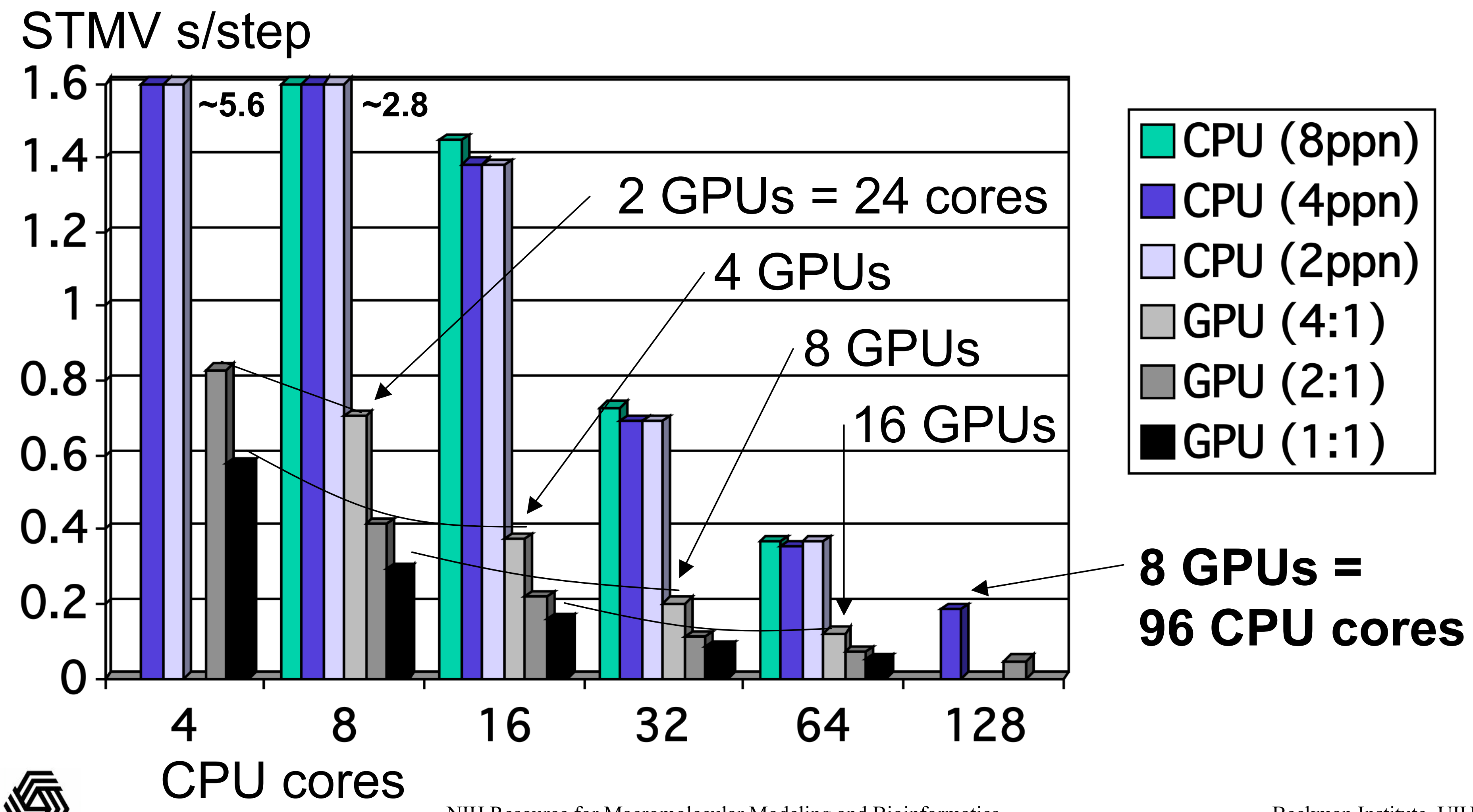
- Threads are cheap but desperately needed.
  - How many can *you* give?
  - 512 threads will keep all 128 FPUs busy.
  - 1024 threads will hide some memory latency.
  - 12,288 threads can run simultaneously.
  - Up to 2×10^{12} threads per kernel invocation.

GPU Acceleration in NAMD

- Only basic non-bonded force calculation
  - GPU and CPU calculations overlap
- Most features should “just work”
  - Not alchemical free energy methods, etc.
- Energy evaluation is not accelerated
  - Use outputEnergies 100 or higher
- GPU work is not load-balanced

NCSA Lincoln Cluster Performance

(8 cores and 2 GPUs per node)



What To Buy Today

- High-end GeForce for desktop/laptop
- Serious compute box:
  - 1000W power supply and 3-4 Tesla C1060
- Cluster:
  - InfiniBand (www.colfaxdirect.com)
  - Desktops with 1-4 GeForce or Tesla C1060
  - 1U servers with Tesla S1070
  - 1U servers with 1-2 built-in C1060
  - Check out www.colfax-intl.com/nvidiaGPU.html

Non-CUDA GPU Acceleration

- AMD/ATI FireStream
  - Stream-based programming didn't catch on
- OpenCL
  - Apple's multi-vendor CUDA-like standard
  - Currently only AMD CPUs and NVIDIA GPUs
  - Write once, but still tune everywhere?
- Intel Larrabee
  - Itanium got a lot of press too

Keep Your Codes Off GPUs

- They can’t accelerate all algorithms.
- You need to rewrite the ones they can.
- Redundancy is a maintenance nightmare.
- Programming models are evolving.
- Have you tuned your CPU code?
- Have you looked for a better algorithm?

Any Questions?

- Macaroni and cheese with choice of mixed baby greens or Polish sausage
- Italian beef on baguette
- Black bean veggie burger on kaiser roll or hummus
- Mushroom brie bisque with Madera wine
- Greek salad, fresh fruit salad, or mixed baby greens with raspberry vinaigrette