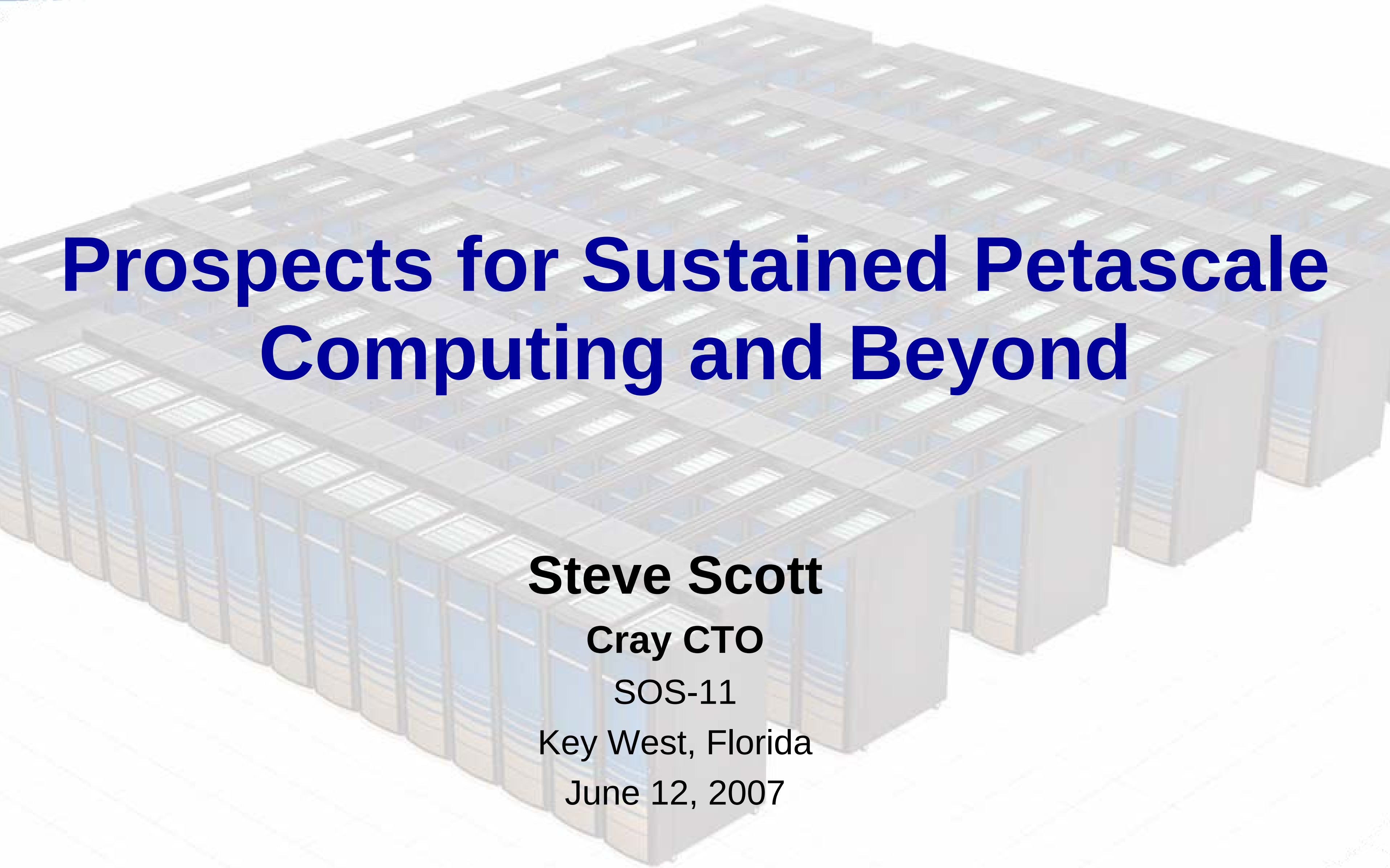


Prospects for Sustained Petascale Computing and Beyond

Steve Scott

Cray CTO

SOS-11

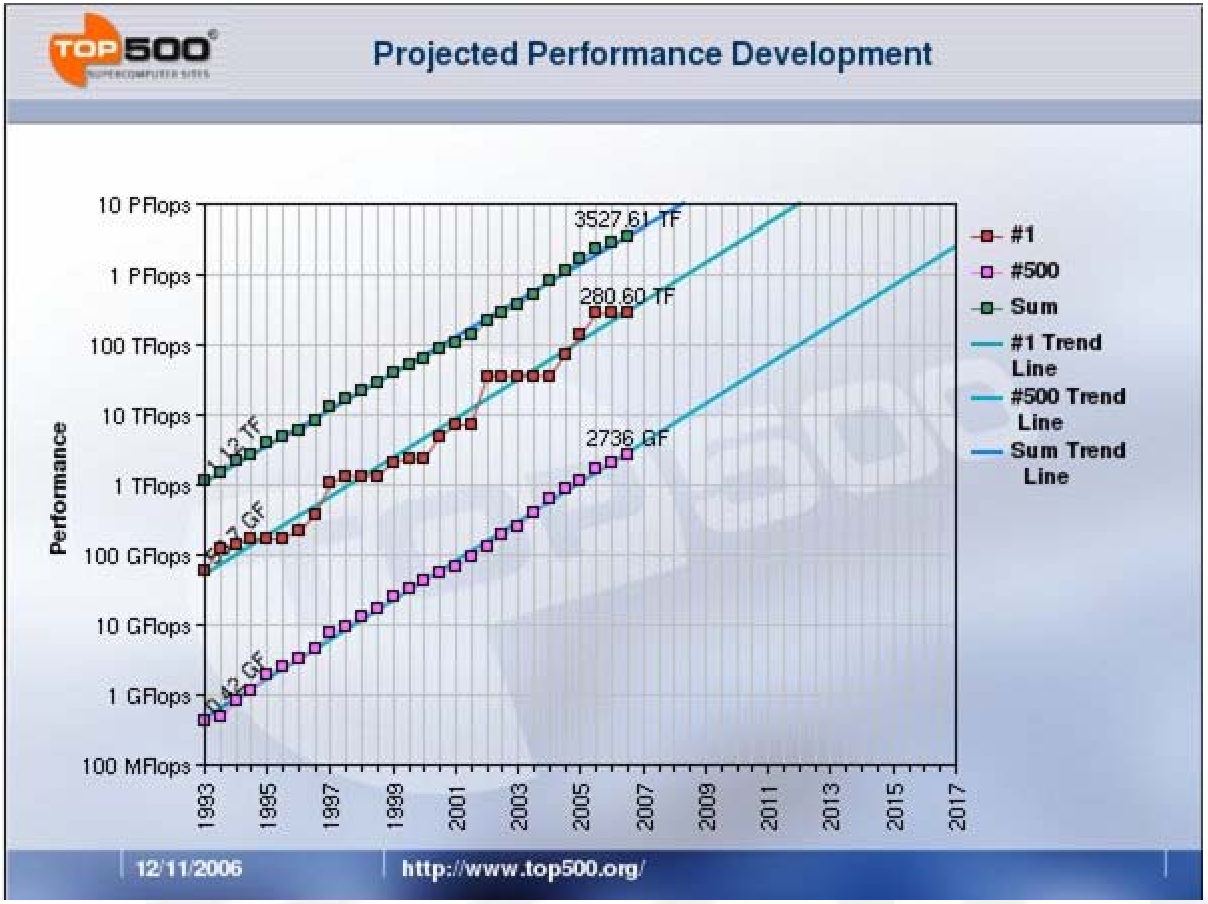
Key West, Florida

June 12, 2007

Copyright 2007, Cray Inc.

March to a Petaflop

- Milestones: Peak, Linpack, Application, Broad set of apps



Major Processor Inflection Point

- Have pushed pipelining about as far as practical
- Have mined most of the available instruction level parallelism
  - ⇒ Benefit of more complex processors has significantly diminished
- Power becoming a major issue, both heat density and electricity cost
  - ⇒ Must moderate frequencies
  - ⇒ Focus on parallel performance vs. fast thread performance
- Flops getting increasingly cheap relative to bandwidth
  - Energy cost of moving data (even across chip) dwarfs that of computation
  - ⇒ Okay to overprovision flops to maximum utilization of memory bandwidth
- Wire delay starting to dominate over transistor delay
  - ⇒ Can’t treat whole chip as single processor; must exploit locality

- Commercial response has been to go multi-core
  - Helps alleviate many of these problems
    - Keeps processors simpler and smaller
  - Will work likely work well on many applications
    - Transaction processing, web serving
    - Highly scalable, dense, regular, blockable, well load-balanced, HPC applications

Copyright 2007, Cray Inc.

Concerns With Multicore

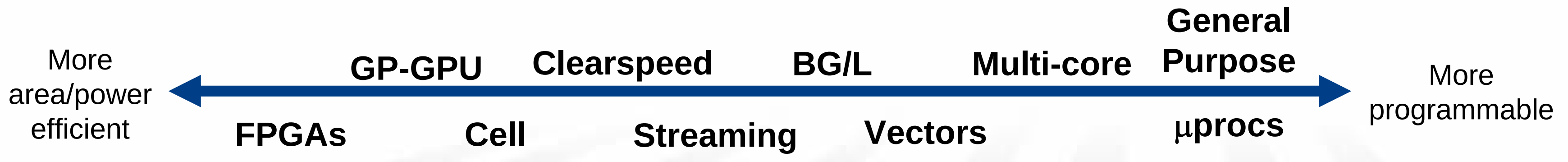
- Rapidly increasing number of processors per system
  - Million-thread MPI applications, anyone?
- Contention for bandwidth off chip
  - Ratios are continuing to worsen
- Synchronization, load balancing and managing parallelism across cores
  - Synchronization is very expensive/heavyweight in traditional processors
  - Dynamic problems create severe load imbalances, which require sophisticated runtime software, synchronization and bandwidth to address
- Memory latency continues to grow and is becoming more variable
  - Traditional processors are not latency tolerant
- Still a lot of control overhead in conventional processors
  - Complex pipelines, extensive prediction schemes, highly associative cache hierarchies, replay logic, high instruction processing bandwidth, etc.
  - Flies in the face of efficient *parallel* performance

- Lots of experimentation with alternative microarchitectures and accelerators

Copyright 2007, Cray Inc.

So, Can We Just Pack Chips with Flops?

- Key is making the system easily programmable
- Must balance peak computational power with generality
  - How easy is it to map high level code onto the machine?
  - How easy is it for computation units to access global data?
- Some examples:
  - FPGAs, Clearspeed CSX600, IBM Cell, GP-GPUs
  - Every one requires significant changes to the source code
  - None have a whole program compiler
- Flop efficiency vs. generality/programmability spectrum:
  - Qualitative only
  - Also influenced by memory architecture, memory system and network



Copyright 2007, Cray Inc.

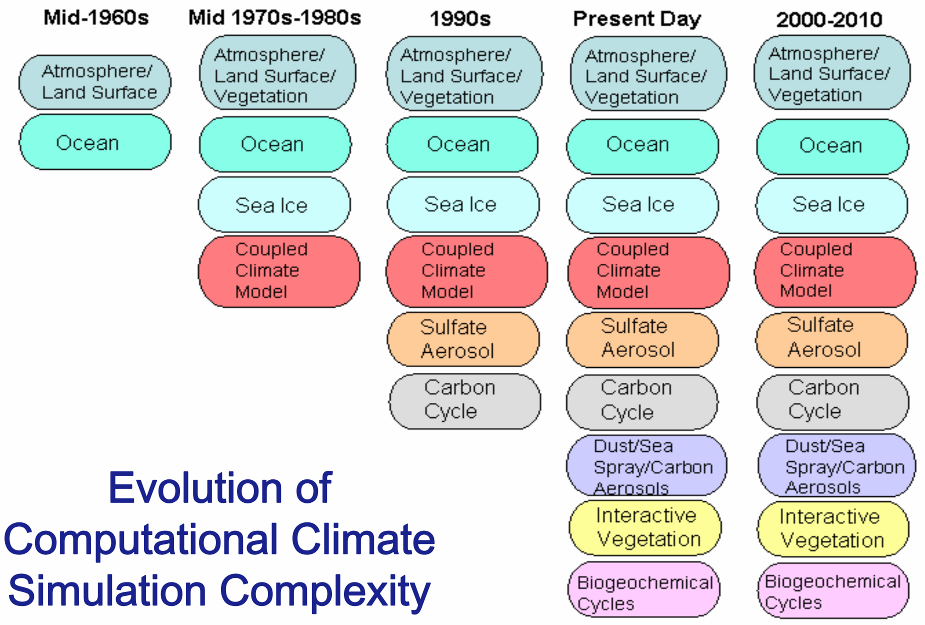
Opportunities to Exploit Heterogeneity

- Applications vary considerably in their demands
- Any HPC application contains some form of parallelism
  - Many HPC apps have rich, SIMD-style *data-level parallelism*
    - Can significantly accelerate via vectorization
  - Those that don’t generally have rich *thread-level parallelism*
    - Can significantly accelerate via multithreading
  - Some parts of applications are not parallel at all
    - Need fast serial scalar execution speed (Amdahl’s Law)
- Applications also vary in their communications needs
  - Required memory bandwidth and granularity
    - Some work well out of cache, some don’t
  - Required network bandwidth and granularity
    - Some ok with message passing, some need shared memory
- No one processor/system design is best for all apps

Copyright 2007, Cray Inc.

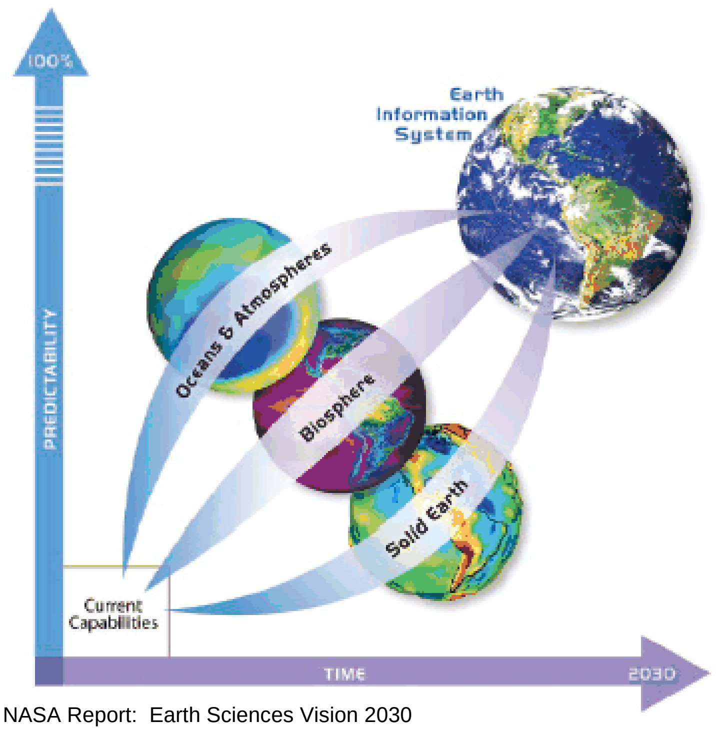
Increasingly Complex Application Requirements

Earth Sciences Example



Evolution of Computational Climate Simulation Complexity

International Intergovernmental Panel on Climate Change, 2004, as updated by Washington, NCAR, 2005



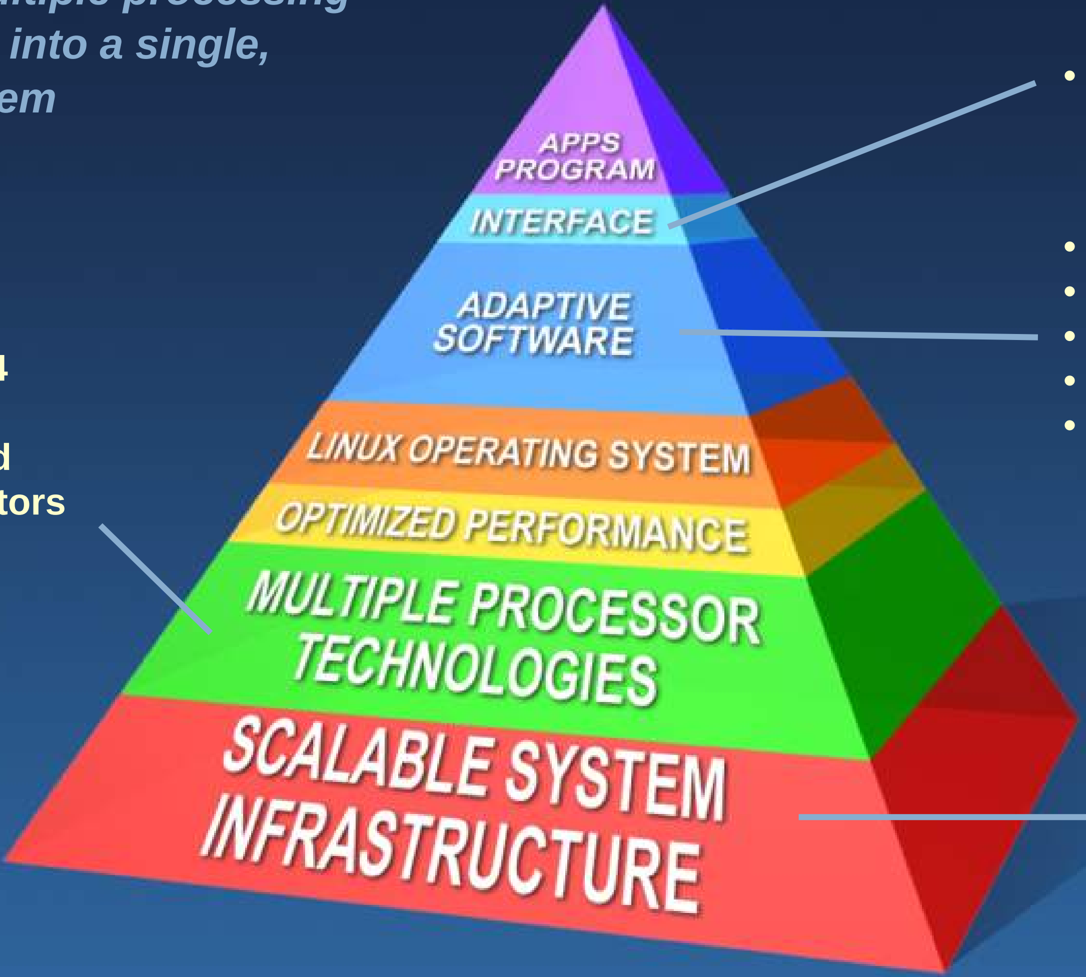
NASA Report: Earth Sciences Vision 2030

Increased complexity and number of components lends itself well to a variety of processing technologies

- Similar trends in astrophysics, nuclear engineering, CAE, etc.
  - Higher resolution, multi-scale, multi-science

Adaptive Supercomputing

Combines multiple processing architectures into a single, scalable system



- Transparent Interface

- Libraries Tools
- Compilers
- Scheduling
- System Management
- Runtime

- Scalar X86/64
- Vector
- Multithreaded
- HW Accelerators

- Interconnect
- File Systems
- Storage
- Packaging

Adapt the system to the application – not the application to the system

Cascade Approach to Higher Productivity

Design an **adaptive, configurable** machine

- Serial (single thread, latency-driven) performance
- SIMD data level parallelism (vectorizable)
- Fine grained MIMD parallelism (threadable)
- Regular and sparse bandwidth of varying intensities

⇒ Increases performance

⇒ Significantly eases programming

⇒ Makes the machine much more broadly applicable

Ease the development of parallel codes

- Legacy programming models: MPI, OpenMP
- Improved variants: SHMEM, UPC and CoArray Fortran (CAF)
- New alternative: Global View (Chapel)

Provide programming tools to ease debugging and tuning at scale

- Automatic performance analysis; comparative debugging

Copyright 2007, Cray Inc.

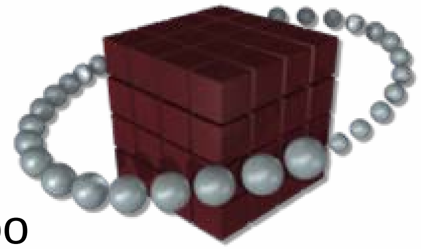
Processing Technologies

Start with best of class microprocessor: AMD Opteron™

- Industry standard x86/64 architecture
- Integrated memory controller ⇒ Very low memory latency (~50ns)
- Open standard, high speed interface (HyperTransport)
- Strong product roadmap

Cray communications acceleration

- Globally addressable memory
- Scalable addressing, translation and synchronization
- Unlimited concurrency for latency tolerance
- Support for low latency, low overhead message passing too

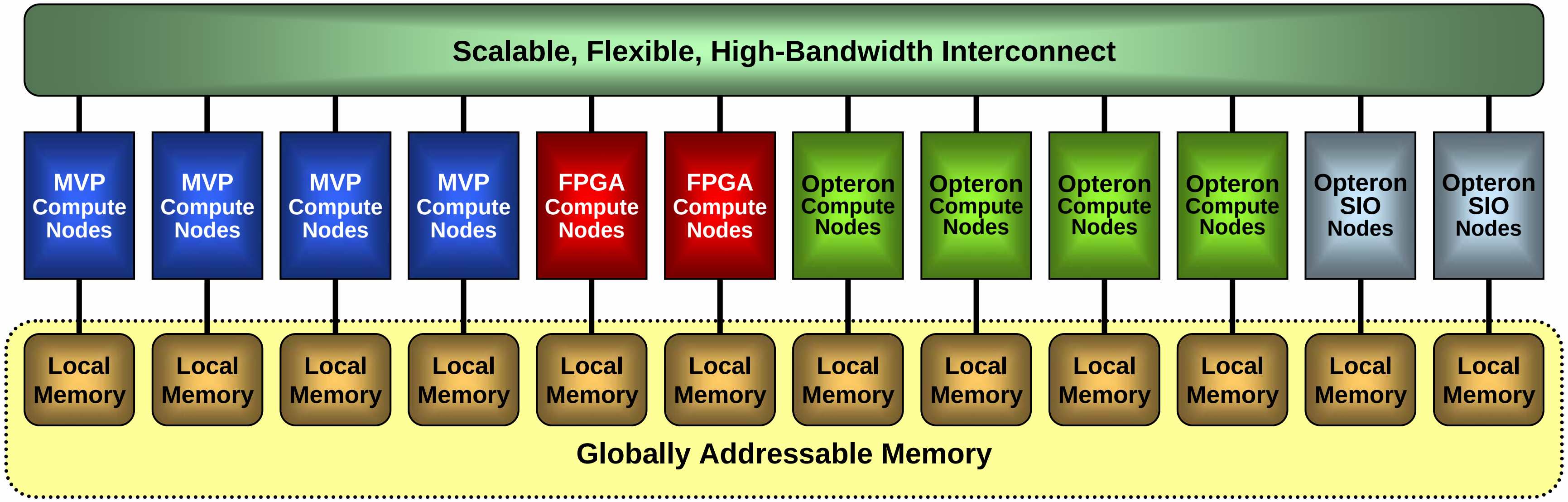


Cray computational acceleration

- Multi-threaded and vector processing
- Exploits compiler-detected parallelism within a node
- Extremely high single-processor performance

Copyright 2007, Cray Inc.

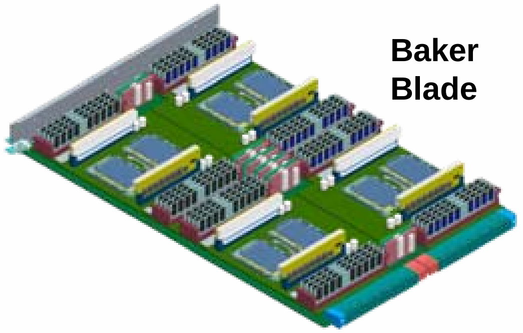
Cascade System Architecture

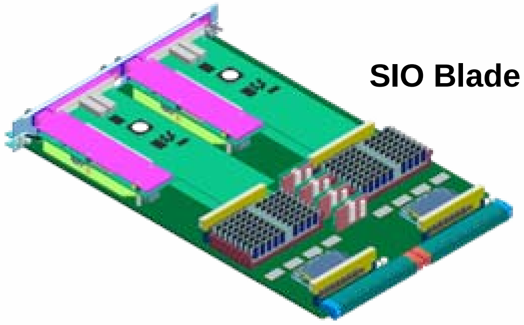


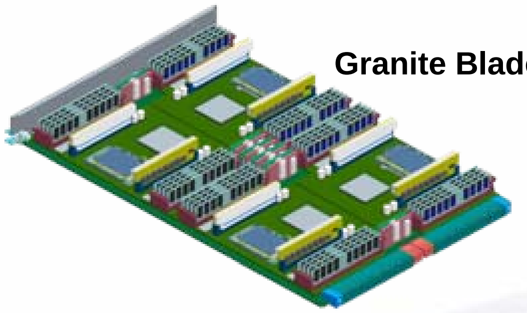
- Globally addressable memory with unified addressing architecture
- Configurable network, memory, processing and I/O
- Heterogeneous processing across node types, and within MVP nodes
- Can adapt at configuration time, compile time, run time

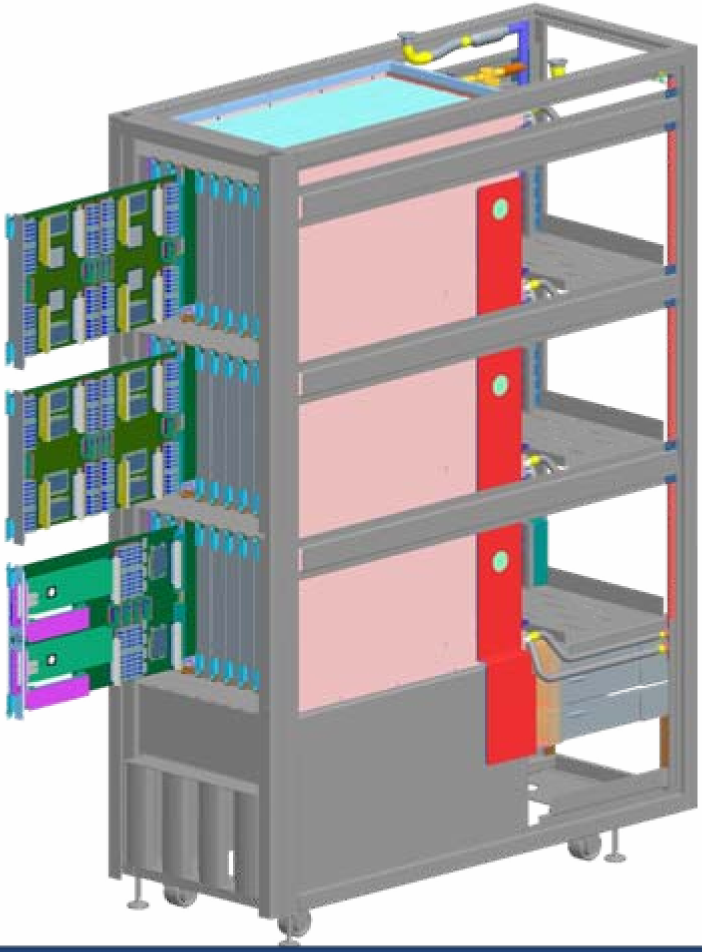
Copyright 2007, Cray Inc.

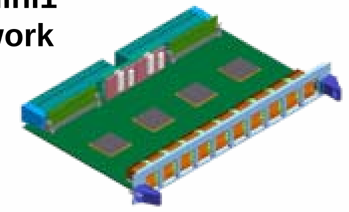
Cascade Packaging Overview

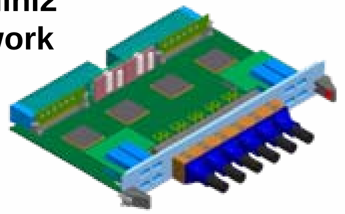


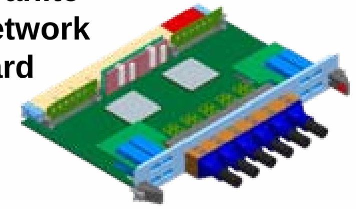








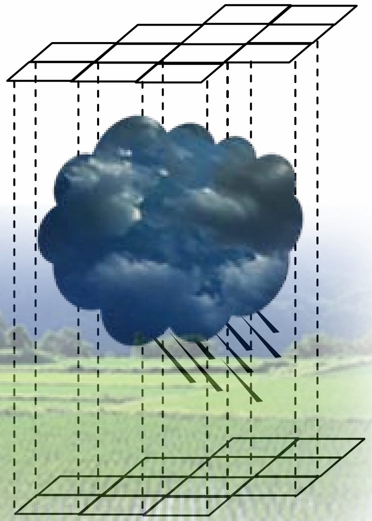




Cascade Cabinet
- Improved density and cooling over XT4
- Extensible over multiple years

Example Application:
Weather Research & Forecasting (WRF) Model

- Mesoscale numerical weather prediction system
  - Regional forecast model (meters to thousands of kilometers)
- Operational forecasting, environmental modeling, & atmospheric research
  - Key application for Cray (both vector & scalar MPP systems)
- Accelerating WRF performance:
  - Part of the code is serial:
    - Runs on Opteron for best-of-class serial performance
  - Most of the code vectorizes really well
    - Dynamics and radiation physics
    - Runs on Granite accelerator in vector mode
  - Cloud physics doesn't vectorize
    - Little FP, lots of branching and conditionals
    - Degrades performance on vector systems
    - Vertical columns above grid points are all independent
    - Runs on Granite accelerator in multithreaded mode

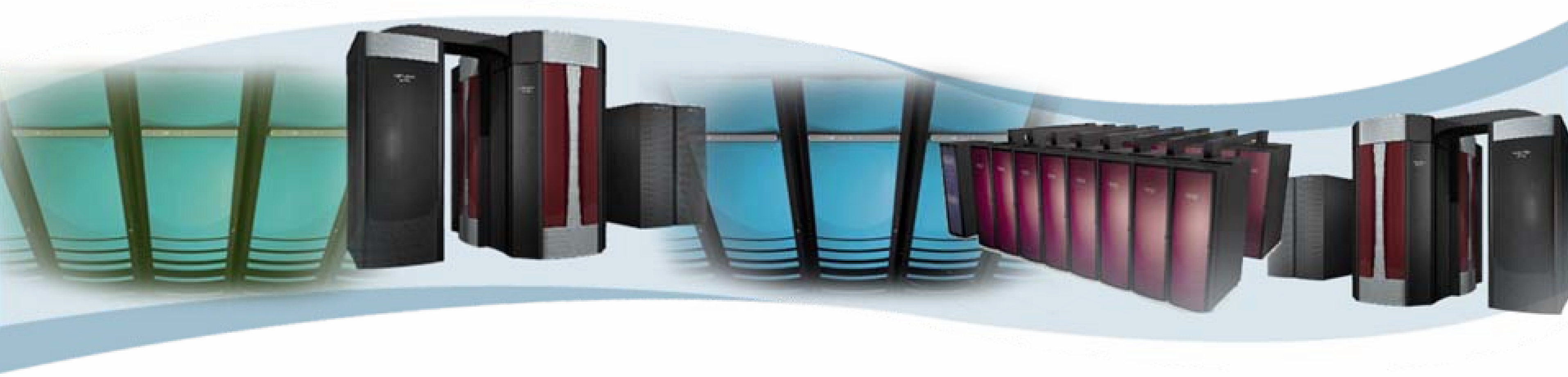


Key Challenges for Sustained PF Computing

- System and application reliability and resiliency
  - Must focus on system resiliency
  - Completing applications much harder than keeping system up
- Programming difficulty
  - MPI is a very low-productivity programming model
  - Debuggers don’t scale
  - Performance tools don’t scale
- Processor microarchitecture to exploit locality
  - Resolving the tension between programmability and efficiency
  - Both hardware and software issues
  - Won’t be critical at petascale; *will* be critical at exascale
- Bandwidth technologies
  - Won’t be critical at petascale; *will* be critical at exascale
- Building systems for tomorrow’s applications
  - Irregular, dynamic, sparse, heterogeneous….

Copyright 2007, Cray Inc.

Thank you.



Questions?

sscott@cray.com

Copyright 2007, Cray Inc.