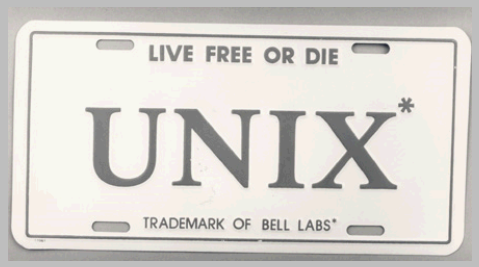
SAND2014-17328PE



Sandia National Laboratories

Exceptional service in the national interest

An Overview of Sandia Lightweight Kernel Operating System R&D

September 2, 2014

Kevin Pedretti
1423 – Scalable System Software

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2011-XXXXP

Outline

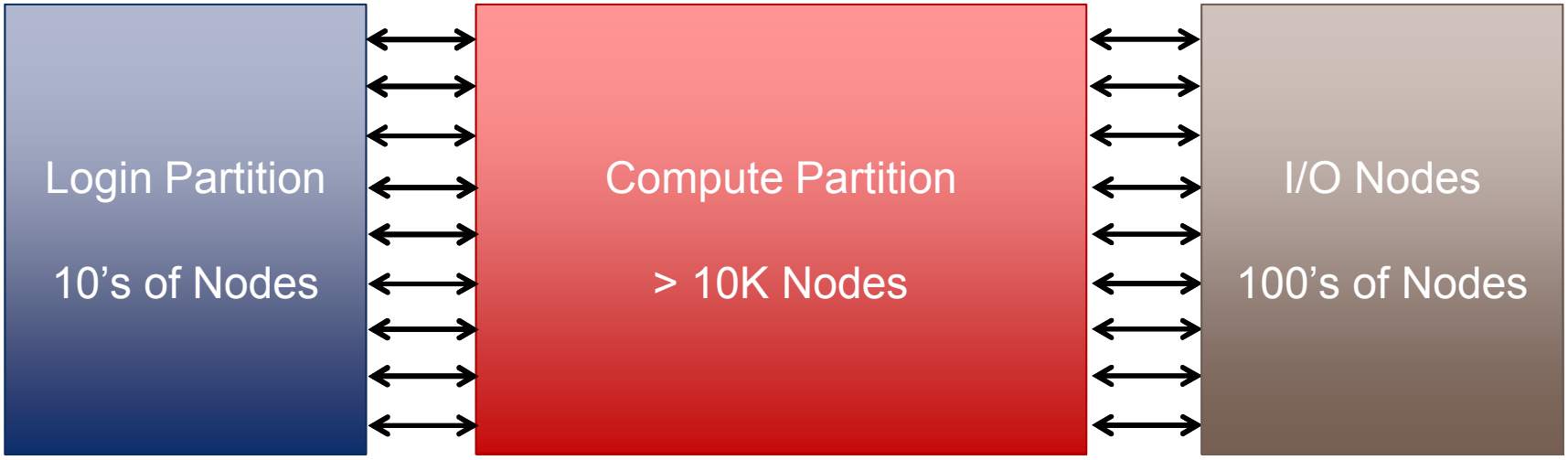
Sandia National Laboratories

- History of SNL Lightweight Kernel Operating Systems (LWK OS)
- Are LWKs still relevant?
- What we're working on
  - XPRESS project (FY13-FY15, DOE/ASCR X-Stack Program)
  - Hobbes project (FY14-FY16, DOE/ASCR OS/Runtime Program)
  - Power API and power management (CSSE, FY14 L2 Milestone)
  - Task mapping (LDRD + CSSE)
- Conclusion

Large-Scale HPC Systems in General

Sandia National Laboratories

- Distributed-memory MIMD
- Message passing used between nodes (MPI)
- Custom interconnect, leverage commodity technologies as much as possible
- Several node types, system software for each type specialized to task
- Big NNSA systems typically $100M - $200M procurements, some NRE funds



Runs Full Linux Distribution (SUSE, RedHat)

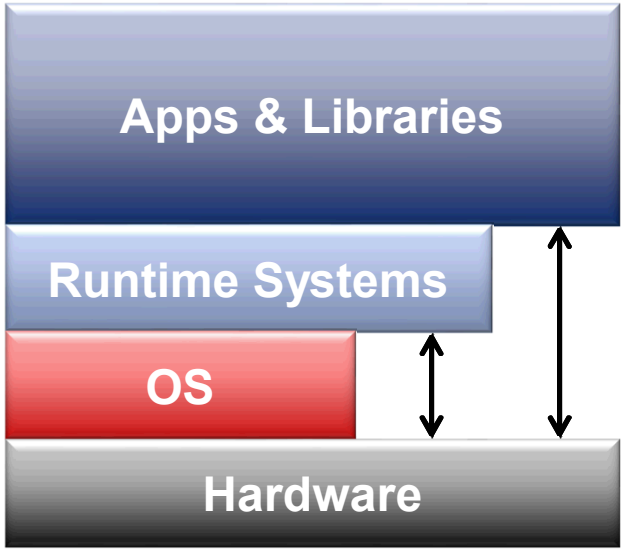
Runs Slimmed-Down OS, Nodes Typically Diskless (LWK, Customized Linux)

Runs Full Linux Distribution, Optimized for Storage I/O

Why a Lightweight Kernel?

Sandia National Laboratories

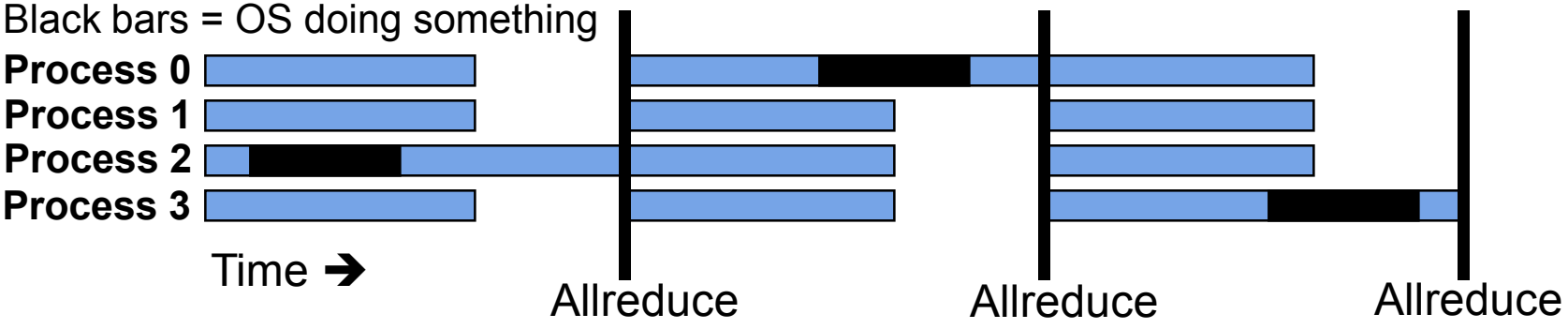
- Scalable apps don't ask for much from OS
  - Relatively easy to support what they want
  - Prevent non-scalable behavior early
- Minimize OS-induced overhead
- Inverted resource management
  - Linux: You get what I want
  - LWK: What do you want?
- Ability to tune, do HPC specific things
  - Memory mgmt. and network stack integration
  - Can't separate OS from node-level architecture
- Reliability, less code fails less often



Compute Node System Software Stack, OS Bypass

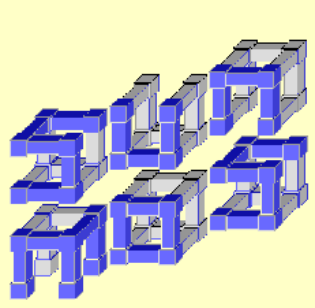
OS Noise Example

Black bars = OS doing something



SNL LWK Timeline (1/5)

Sandia National Laboratories



- 1993 – SUNMOS (Sandia + UNM + OS)
  - Intel Paragon, 1840 compute nodes
  - Replaced OSF/1 distributed OS, was really bad
  - Huge success… why wouldn't you want to double your available memory and get 5x better network bandwidth?
  - See http://pages.swcp.com/~mccurley/humor/sunmos_humor.html
- 1997 – Cougar
  - Intel ASCI Red / TFLOPS, 4536 compute nodes (2 CPUs ea.)
  - SUNMOS was such a great success, let's do it again
  - Intel productized SUNMOS/Puma as Cougar
  - SNL virtual node mode, basic OpenMP support
  - SNL Portals 2.0 networking, "zero-copy" data movement
  - As experiment ported Linux to Red; Cougar was much better

SNL LWK Timeline (2/5)

Sandia National Laboratories

- 2001 – Cplant Linux
  - Ran on several clusters, largest Ross ~1800 nodes
  - Let's try Linux on the compute nodes
  - Cplant idea was to build MPP style system out of commodity HW + SW
  - Developed scalable job launch and system control mechanisms
  - SNL Portals 3.3 – Linux memory management caused pain
  - See Brightwell 2002
    "Why Linux is a Bad Idea as a Compute Node OS (for Balanced Systems)"

SNL LWK Timeline (3/5)

Sandia National Laboratories

- 2004 – Catamount
  - Red Storm, 12960 compute nodes
  - We're doing a LWK, period. It worked on Red.
  - Ported Cougar -> Catamount
    - SNL did most of work on port, with Cray assistance and productization
  - Portals 3.3, Custom SeaStar interconnect, heavy SNL involvement
  - Development of Red Storm was difficult
    - Several near no-go decisions
    - May 2005 press release warning Cray was about to be delisted:

      "Cray said manufacturing-related charges and unusually high research costs for its Sandia Red Storm program hurt results."
  - Turned out to be most successful Cray product line ever
    - Cray productized Red Storm as Cray XT3
    - Evolved through XT4, XT5, and XT6 generations
  - Red Storm turned off in 2012, still running Catamount

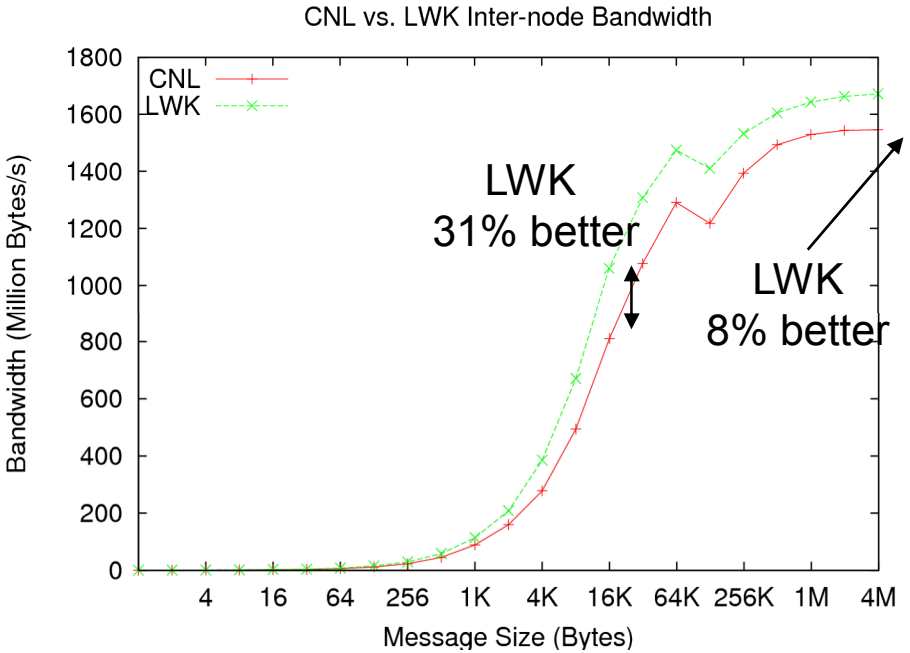
SNL LWK Timeline (4/5)

Sandia National Laboratories

- 2007 – Cray ships Compute Node Linux Environment (CLE)
  - Customers asking for Linux, desire to expand market
  - Catamount seen as too difficult to maintain in parallel
  - Set within 10% performance target, tested at ORNL
  - CLE not full Linux, not what most people think of when you say “Linux”

	CNL 2.0.03+ PGI 6.1.6	Catamount 2.0.05+ PGI 6.1.3	CNL vs. Catamount % CNL worse
GTC			
1024 XT3 only	595.6	584.0	2.0
4096 XT3 only	614.6	593.8	3.5
20000 XT3/XT4	786.5	778.9	1.0
VH1			
1024 XT3 only	22.7	20.9	8.6
4096 XT3 only	137.1	117.4	16.8
20000 XT3/XT4	1186.0	981.7	20.8
POP			
4800 XT3 only	90.6	77.6	16.8
20000 XT3/XT4	98.8	75.2	31.4

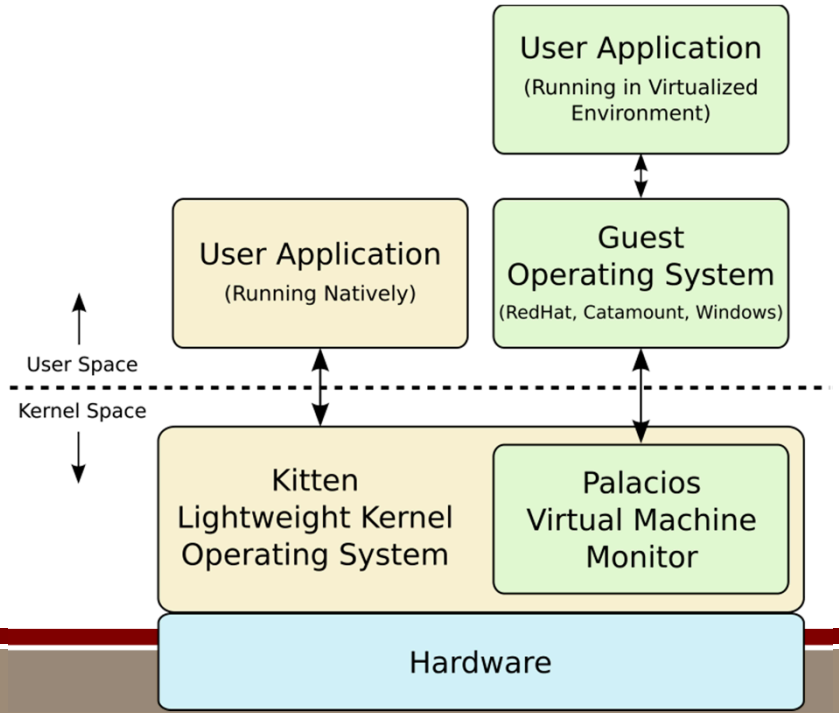
Testing performed June 16-17, 2007 at ORNL

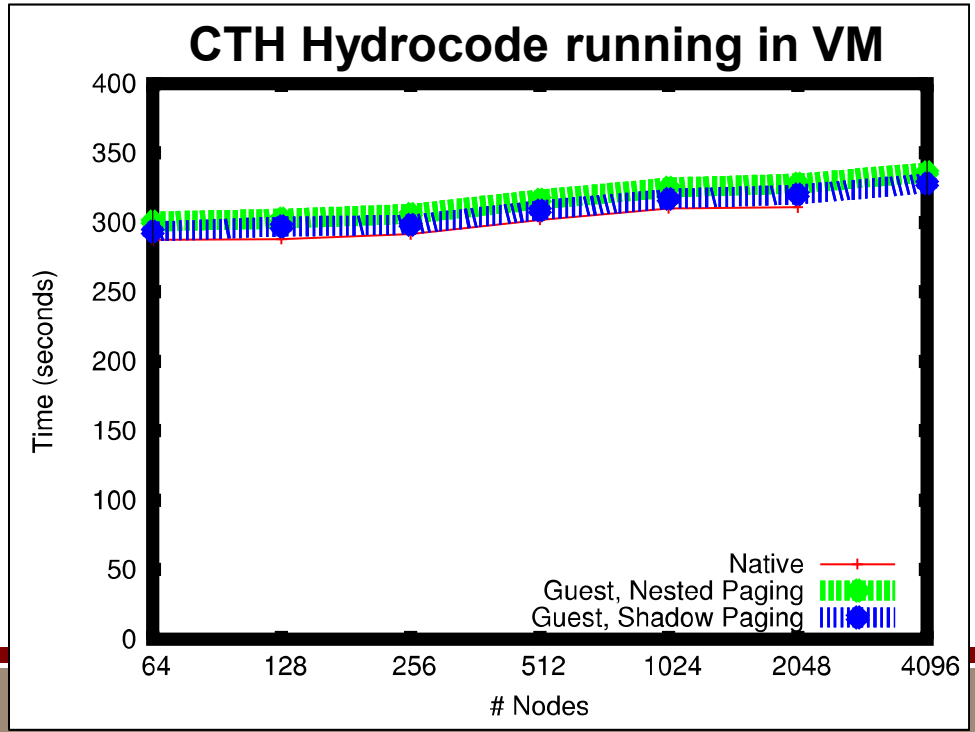


SNL Timeline (5/5)

Sandia National Laboratories

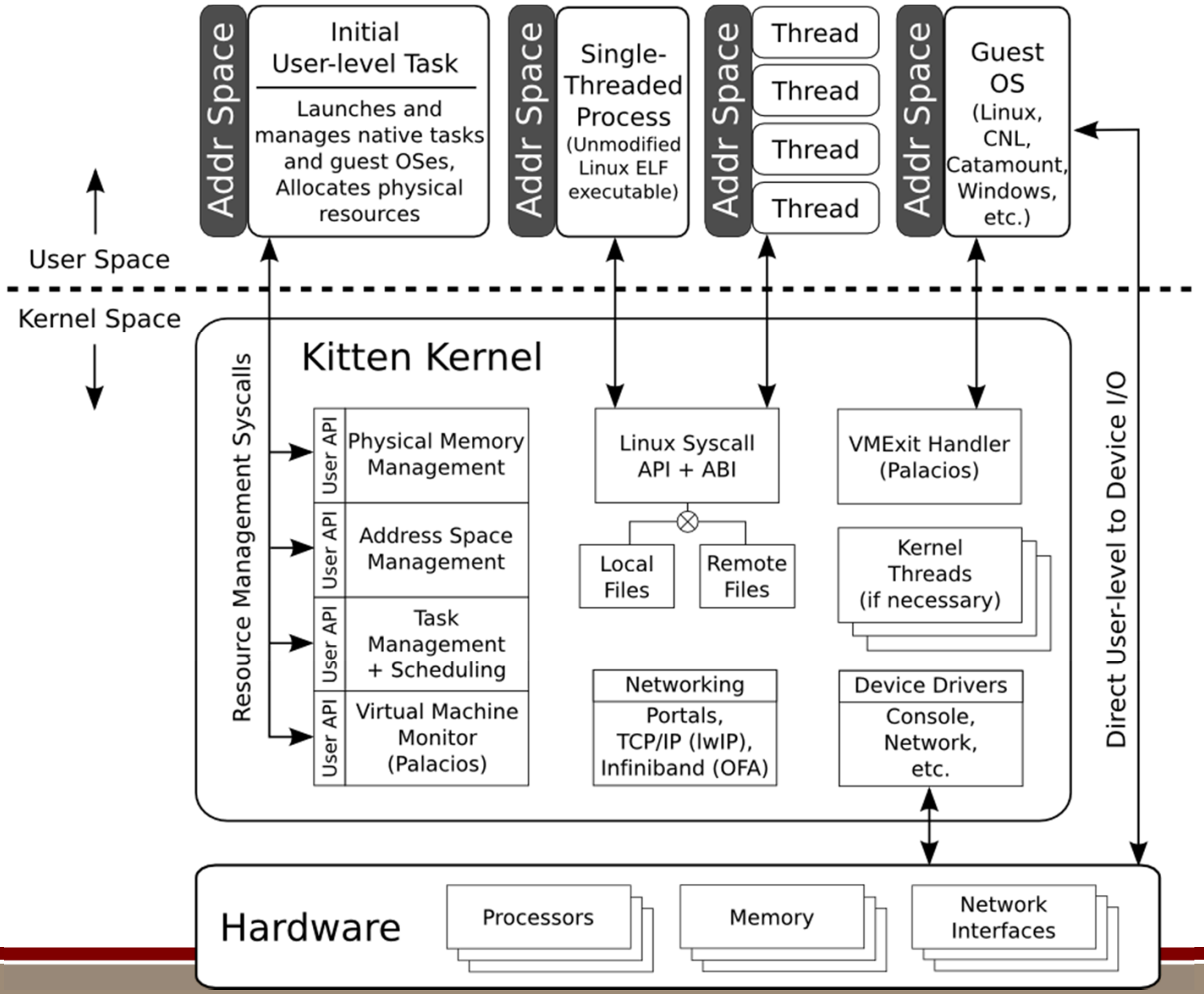
- Kitten LWK LDRD (FY-08-10)
  - Create “modern” open-source LWK platform
    - Linux ABI compatible… compile on Linux, run on Kitten
    - Support for multi-threading, POSIX Pthreads
    - Explore use of hardware virtualization to support full-OS functionality
  - Retain scalability and determinism of Catamount
  - Get ready for next machine, find vendor to partner with





Kitten Architecture

Sandia National Laboratories

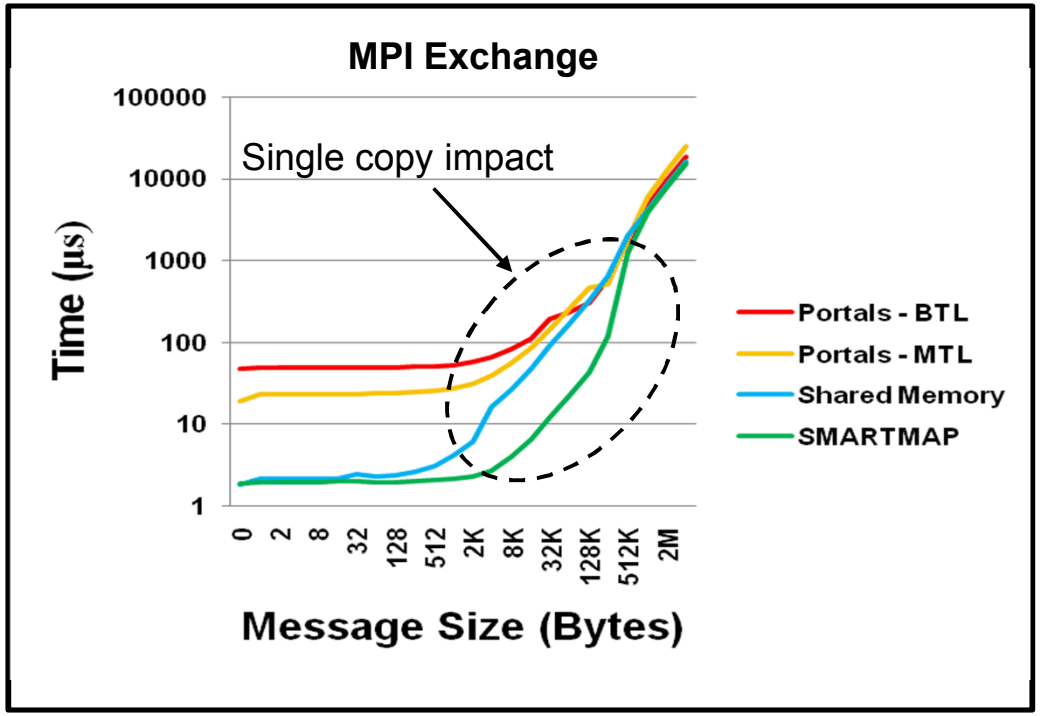


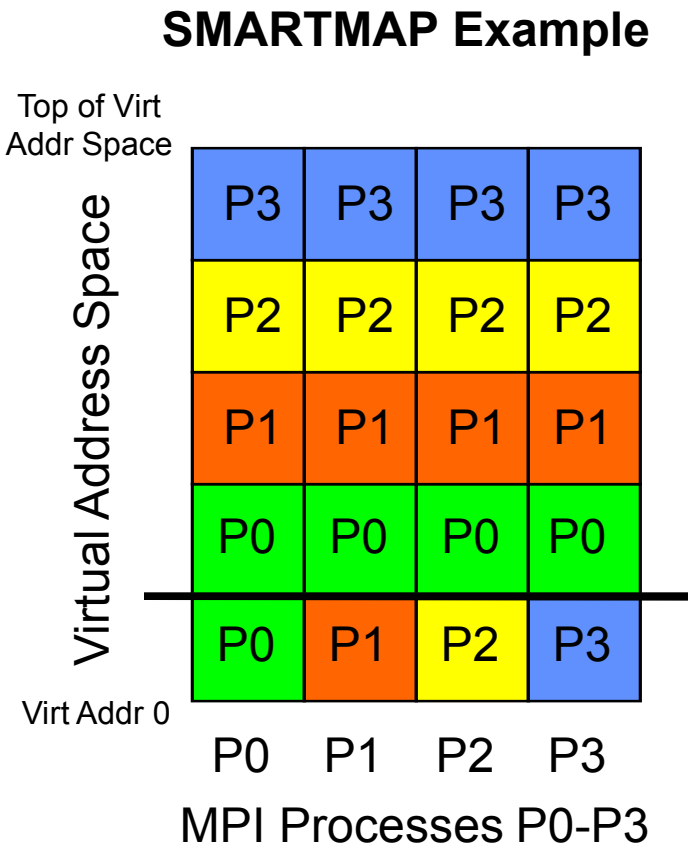
Kitten SLOC
14K Core
13K Arch
14K Includes

SMARTMAP Intra-node Optimization Eliminates Unnecessary Memory Copies

Sandia National Laboratories

- **Basic Idea: Each process on a node maps the memory of all other processes on the same node into its virtual address space**
- **Enables single copy process to process message passing (vs. multiple copies in traditional approaches)**





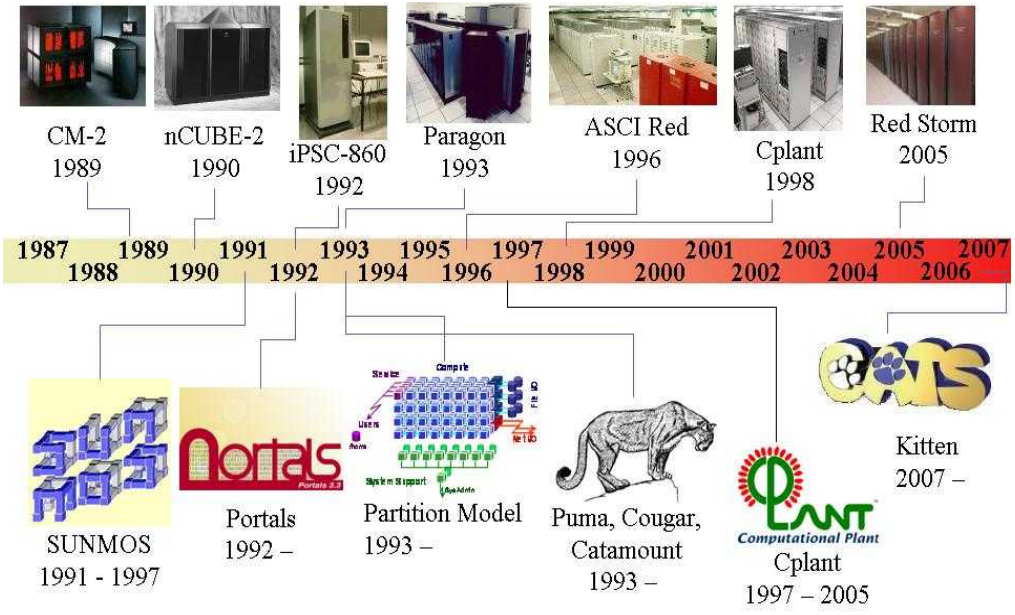
SNL 2013 Patent, influenced Cray to use similar technique in their MPI

System Software@Sandia

- Established the functional partition model for HPC systems
  - Tailor system software to function (compute, I/O, user services, etc.)
- Pioneered the research, development, and use of lightweight kernel operating systems for HPC
  - Only DOE lab to deploy OS-level software on large-scale production machines
  - Provided blueprint for IBM BG/L,P,Q CNK
- Set the standard for scalable parallel runtime systems for HPC
  - Fast application launch on tens of thousands of processors
- Significant impact in the design and of scalable HPC interconnect APIs
  - Only DOE lab to deploy low-level interconnect API on large-scale production machines

AWARDS:

- **1998** Sandia Meritorious Achievement Award, TeraFLOP Computer Installation Team
- **2006** Sandia Meritorious Achievement Award, Red Storm Design, Development and Deployment Team
- **2006** NOVA Award Red Storm Design and Development Team
- **2009** R&D 100 Award for Catamount N-Way Lightweight Kernel
- **2010** Excellence in Technology Transfer Award, Federal Laboratory Consortium for Technology Transfer
- **2010** National Nuclear Security Administration Defense Programs Award of Excellence



Outline

Sandia National Laboratories

- History of SNL Lightweight Kernel Operating Systems (LWK OS)
- Are LWKs still relevant?
- What we're working on
  - XPRESS project (FY13-FY15, DOE/ASCR X-Stack Program)
  - Hobbes project (FY14-FY16, DOE/ASCR OS/Runtime Program)
  - Power API and power management (CSSE, FY14 L2 Milestone)
  - Task mapping (LDRD + CSSE)
- Conclusion

Today: LWK Environment

Sandia National Laboratories

- There is still a Top-10 LWK-based system on Top-500 list, LLNL/Sequoia IBM BlueGene/Q (#3)
  - IBM is discontinuing BlueGene line, presumably CNK as well
  - Several IBM CNK people now at Intel, pursuing “FusedOS” approach
- Cray heavily invested in Linux, not going to change
  - Linux is running on Cielo compute nodes, world did not end
  - Linux can be made to work with enough engineering effort
- Intel HPC “Pathfinding” group is doing something LWK-related
- Google is doing some non-Linux OS work, secretive, not HPC

- Unlikely Sandia will dictate a home-grown LWK again
  - Paired with LANL through ACES and NERSC on big procurements
  - Technical arguments for LWK not enough, 10% not enough
  - Runtime system, abstraction layer R&D has higher potential payoff

Today: LWK Drivers Are Still Valid

Sandia National Laboratories

- Lots of new hardware challenges to tackle
  - 2-level memory, node-local NVRAM, complex on-chip network topologies, power management, heterogeneous cores, …
  - LWK is a good vehicle for exploring solutions
- Still can't separate OS from architecture
  - BlueGene used embedded cores with weak MMU/TLB -> Linux sucked
  - GPUs don't run an OS, but do have a 20M+ SLOC driver stack + firmware
  - D.E. Shaw Anton, Cray MTA/XMT, … so strange they can't run a traditional OS, need custom system software development
  - Strange hardware capabilities, like non-cache-coherent core groups, break traditional OS assumptions
- Ability to do HPC-specific things, without huge battle with Linux "community"
  - Examples: mmunotify patches, huge pages, OOM killer
  - Cray does a ton of work on Linux kernel, pushes almost nothing back

In the Report from the Task Force on HPC of the Secretary of Energy EAB, Aug. 10 2014

Sandia National Laboratories

To best of my knowledge, we did not contribute this:

"Effective management of hardware resources requires the development of an **operating system tailored to the specific system architecture**. The operating system schedules selected resources (memory and processors) for the one or more concurrently running applications. It must manage a hierarchy of memories with different performance characteristics as well as input/output devices and network connections. New algorithms to map data onto memories with predictable/known access patterns by processors are needed. Dynamic remapping may enhance performance. It is likely that the operating system, and possibly language compilers, will participate in energy management, as well as managing routinely failing hardware, and possibly software, elements."

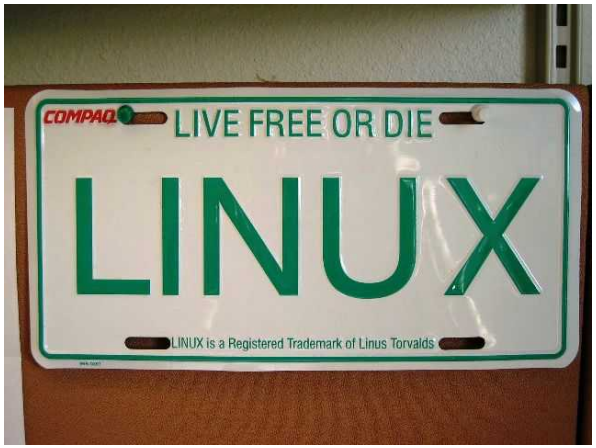
"Ideally, the operating system will monitor its own health and performance, reporting in terms that permit administrators to incrementally tune the operation of the system to attain higher performance and higher reliability. Building such an operating system for a new architecture will be challenging. **It is unlikely that extant operating system software can be re-purposed to manage the resources of a new and novel architecture.** To extract the potential speed from a novel system, the operating system software needs to be well matched to the hardware architecture in order to exploit its capabilities."

Linus Torvalds on Linux

Sandia National Laboratories

- "I mean, sometimes it's a bit sad that we are definitely not the streamlined, small, hyper-efficient kernel that I envisioned 15 years ago...The kernel is huge and bloated, and our icache footprint is scary. I mean, there is no question about that. And whenever we add a new feature, it only gets worse." – Linus Torvalds in email discussion, September 2009
  http://www.theregister.co.uk/2009/09/22/linus_torvalds_linux_bloated_huge/

- Don't get me wrong, I love Linux. Depend on it every day.



Outline

Sandia National Laboratories

- History of SNL Lightweight Kernel Operating Systems (LWK OS)
- Are LWKs still relevant?
- What we're working on
  - XPRESS project (FY13-FY15, DOE/ASCR X-Stack Program)
  - Hobbes project (FY14-FY16, DOE/ASCR OS/Runtime Program)
  - Power API and power management (CSSE, FY14 L2 Milestone)
  - Task mapping (LDRD + CSSE)
- Conclusion

XPRESS and Hobbes Projects

Sandia National Laboratories

- XPRESS
  - SNL Role: Determine OS functionality needed to support emerging runtime systems like HPX, Qthreads, OCR, …
    Create and prototype “Runtime Interface to the Operating System (RIOS) interface.
  - DOE/ASCR Funding, FY13-15 (possible FY16 extension)
    - SNL lead institution, PI: Brightwell, Pedretti LXK+RIOS lead
    - $2.2 M budget/year, 3 Labs (SNL, ORNL, LBL), 5 Universities
- Hobbes
  - SNL Role: Develop necessary OS/R interfaces and system services to support complex simulation and analysis workflows. Leverage virtualization to support multiple co-located OS/R stacks.
  - DOE/ASCR Funding, FY14-16
    - SNL lead institution, PI: Brightwell, Pedretti NVL thrust lead
    - $2.2 M budget/year, 4 Labs (SNL, LANL, ORNL, LBL), 8 universities

Collaborators and SNL Staff

- Collaborators
  - XPRESS
    - Labs (3)
      - Sandia, LBL, ORNL
    - Universities (5)
      - Indiana, Louisiana State, Houston, North Carolina, Oregon
  - Hobbes
    - Labs (4)
      - Sandia, LBL, LANL, ORNL
    - Universities (8)
      - Georgia Tech, Indiana, North Carolina State, Northwestern, Arizona, Berkeley, New Mexico, Pittsburgh

- SNL Staff
  - XPRESS
    - Ron Brightwell (overall PI)
    - David DeBonis
    - Kurt Ferreira
    - Stephen Olivier
    - Kevin Pedretti
    - Dylan Stark
  - Hobbes
    - Ron Brightwell (overall PI)
    - Mike Levenhagen
    - Jay Lofstead
    - Kevin Pedretti
    - Brian Gaines

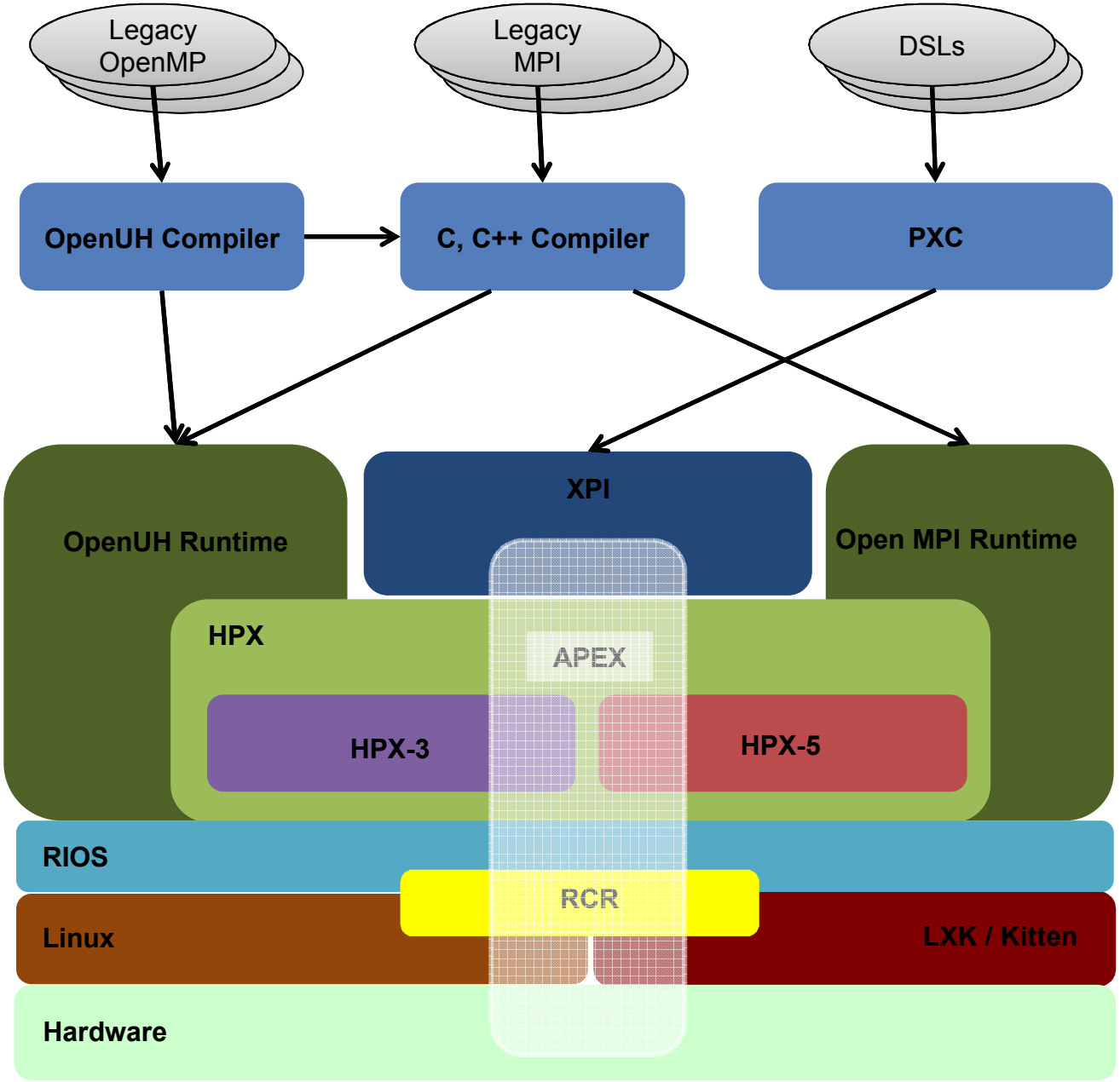
XPRESS: LXK/RIOS Research Goals

Sandia National Laboratories

- XPRESS aims to increase synergy of compute node OS kernel and user-level runtime systems
  - Today: Runtime must work around host OS, assume worst case
  - Vision: Runtime cooperates with host OS, delegated more control
- Key RIOS drivers (Runtime Interface to the OS)
  - Runtime needs guarantees about resource ownership and behavior
  - OS needs way to shift resources between multiple runtimes
  - Two-way interfaces needed for key resources
    - Runtime tells OS what it needs, OS tells runtime what it gets
    - OS remembers original request, notifies runtime if more resources become available. Notifies runtime of resources need to be reclaimed.
  - Event-based protocol to notify of dynamic events (e.g., power state change, transient error)
- LXK = Kitten + RIOS

XPRESS Programming Environment Vision

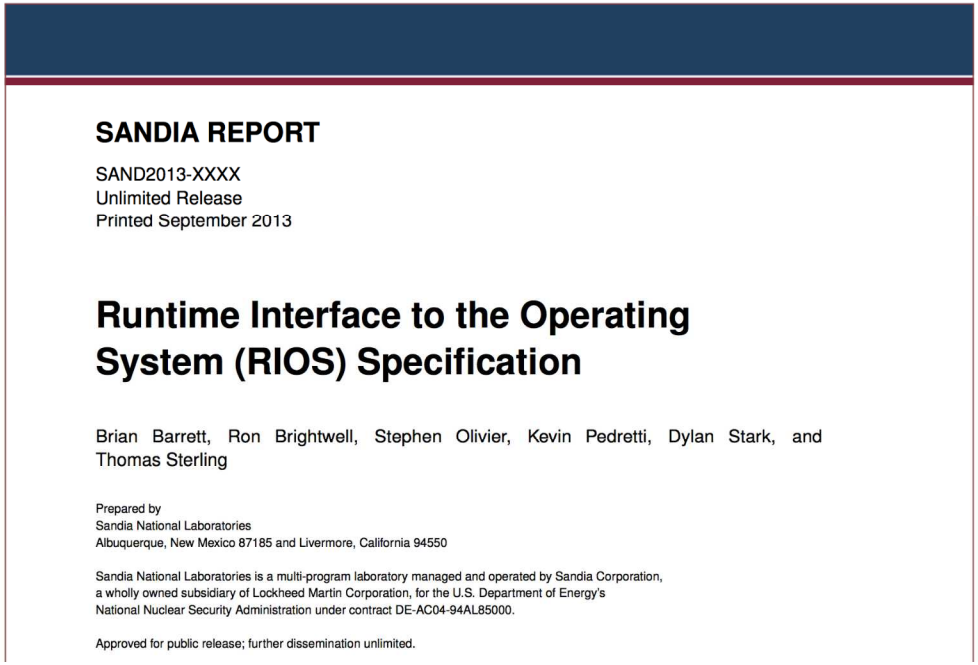
Sandia National Laboratories



XPRESS: Areas Covered by RIOS

Sandia National Laboratories

- Legacy support services
- Job management
- Memory management
- Thread management
- Network interface
- System topology and locality
- Introspection
- File I/O
- Power management



SANDIA REPORT

SAND2013-XXXX
Unlimited Release
Printed September 2013

Runtime Interface to the Operating System (RIOS) Specification

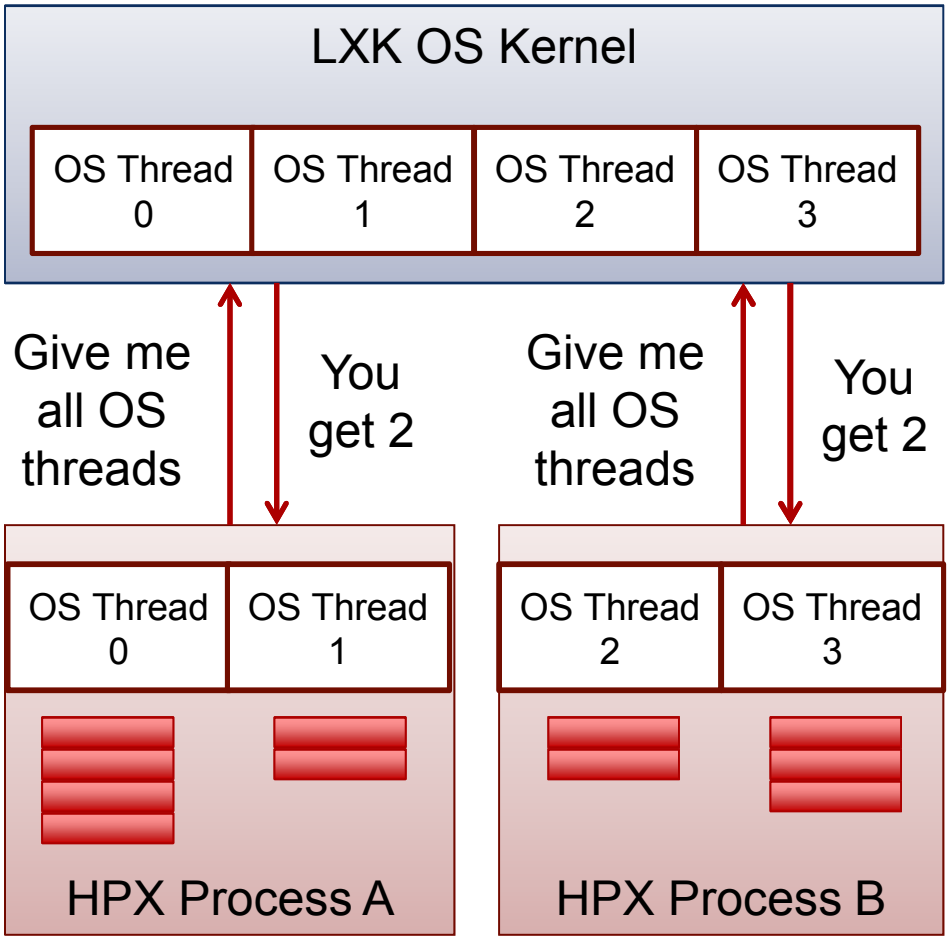
Brian Barrett, Ron Brightwell, Stephen Olivier, Kevin Pedretti, Dylan Stark, and Thomas Sterling

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

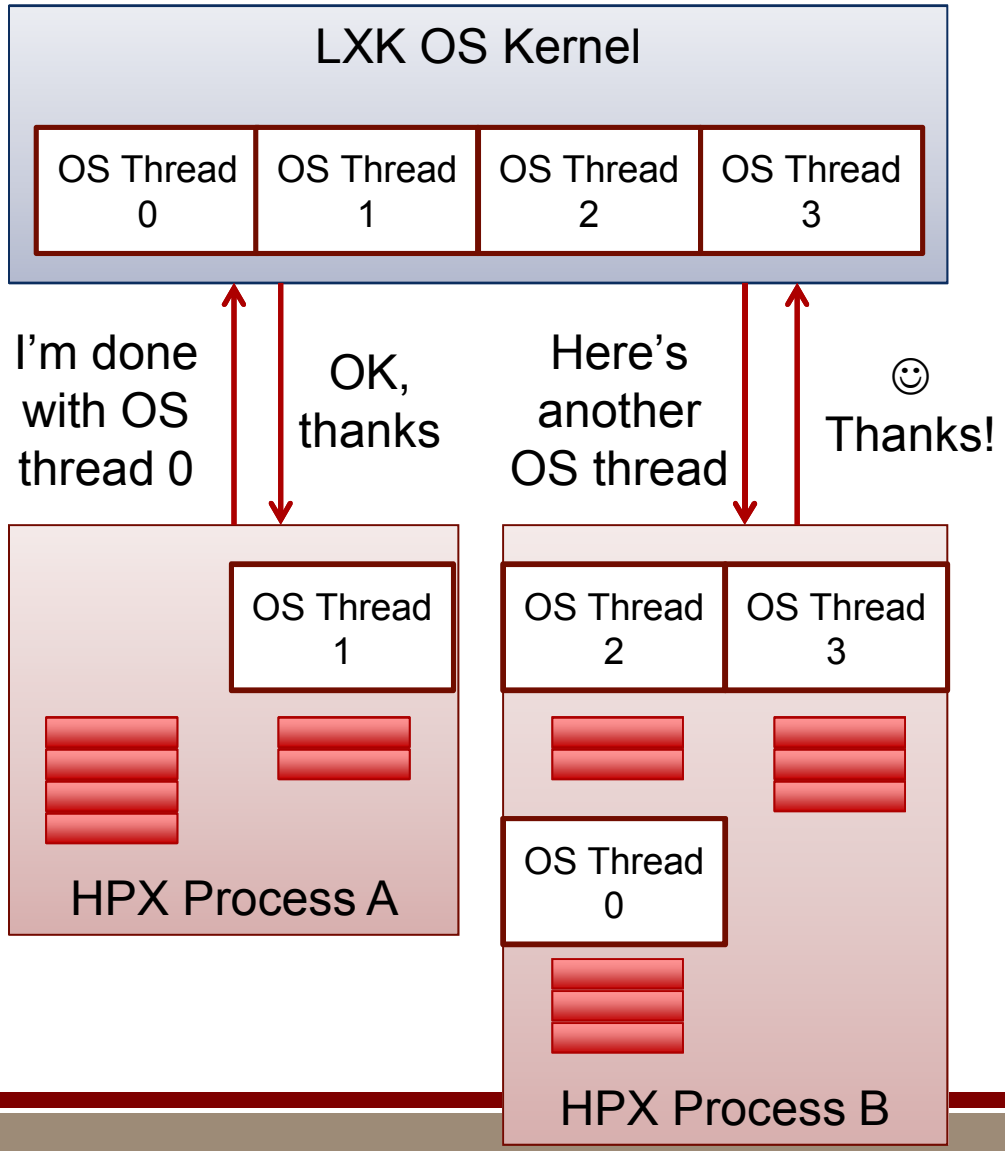
Approved for public release; further dissemination unlimited.

Two-level Thread Scheduling



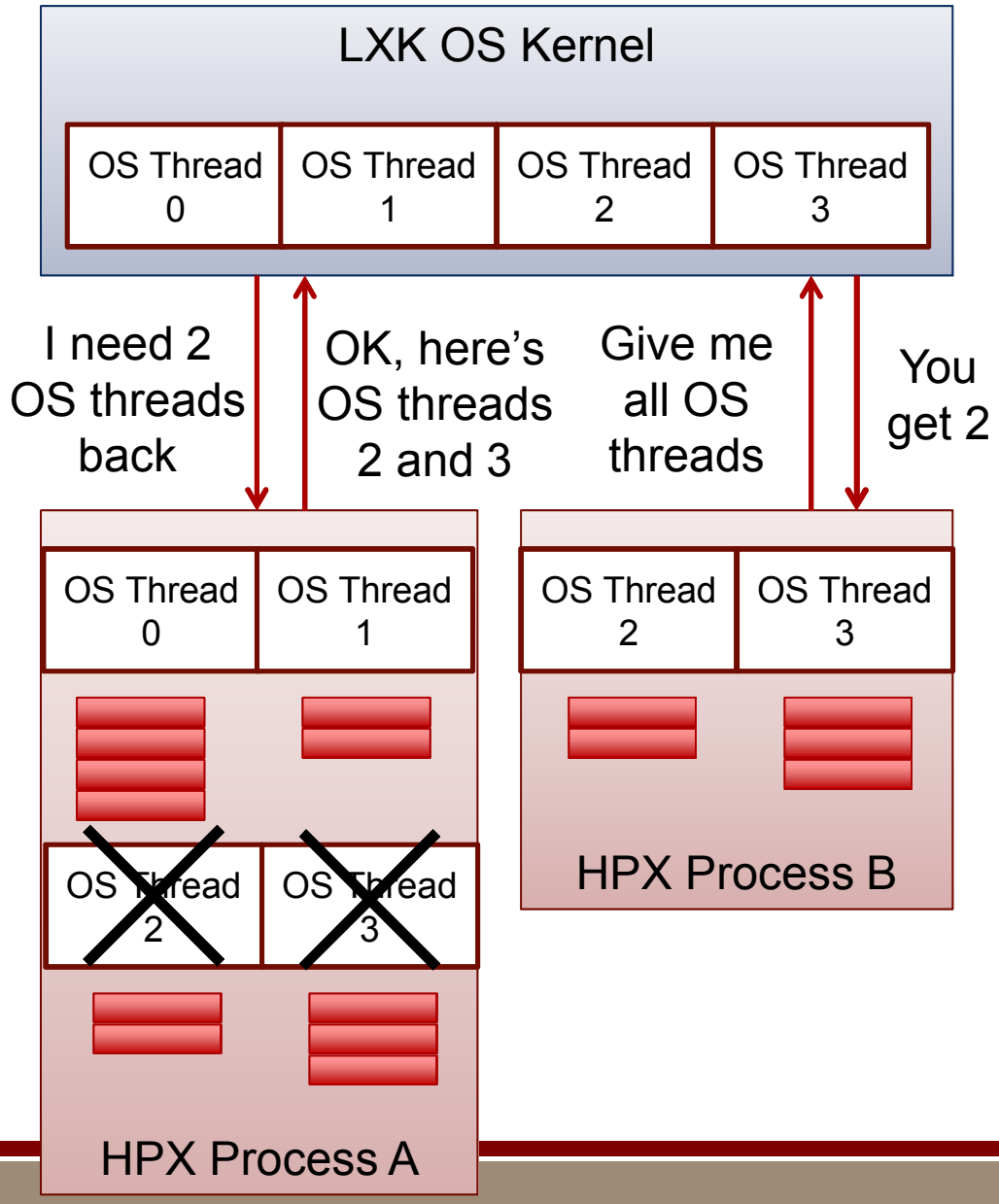
- OS threads track hardware contexts of execution (e.g., physical cores)
- Runtime requests OS threads from OS, runtime schedules its task queues onto OS threads
- RIOS defines protocol runtime uses to ask for OS threads, protocol OS uses to tell runtime what it gets

Two-level Thread Scheduling (2)



- OS remembers original request. In this case, each HPX process could use all available OS threads but each initially allocated only two
- HPX process A gives up an OS thread, LXK decides to reallocate it to HPX process B
- HPX process B initializes a new task queue and starts scheduling tasks on the new OS thread it was allocated

Two-level Thread Scheduling (3)



- OS can ask a runtime for OS threads back at any time
  - Runtime must cooperate or be killed by OS
  - OS gives runtime some time to react, move tasks off of the OS threads being returned
- In this example HPX Process B arrives at some time after HPX Process A; OS allocates B two OS threads that it reclaims from A

Outline

Sandia National Laboratories

- History of SNL Lightweight Kernel Operating Systems (LWK OS)
- Are LWKs still relevant?
- What we're working on
  - XPRESS project (FY13-FY15, DOE/ASCR X-Stack Program)
  - Hobbes project (FY14-FY16, DOE/ASCR OS/Runtime Program)
  - Power API and power management (CSSE, FY14 L2 Milestone)
  - Task mapping (LDRD + CSSE)
- Conclusion

Hobbes: Focusing on Application Composition as Fundamental Driver

Sandia National Laboratories

- More complex workflows are driving need for advanced OS services and capability
  - Exascale applications will continue to evolve beyond a space-shared batch scheduled approach
- HPC application developers are employing ad-hoc solutions
  - Interfaces and tools like mmap, ptrace, python for coupling codes and sharing data
- Tools stress OS functionality because of these legacy APIs and services
- More attention needed on how multiple applications are composed
- Several use cases
  - Ensemble calculations for uncertainty quantification
  - Multi-{material, physics, scale} simulations
  - In-situ analysis
  - Graph analytics
  - Performance and correctness tools
- Requirements are driven by applications
  - Not necessarily by parallel programming model
  - Somewhat insulated from hardware advancements

Hobbes Has Seven Components

Sandia National Laboratories

- Node Virtualization Layer
- Enclave OS
- Scheduling
- Programming Models
- Global Information Bus
- Resilience
- Power/Energy

Hobbes Team (NVL group in bold)

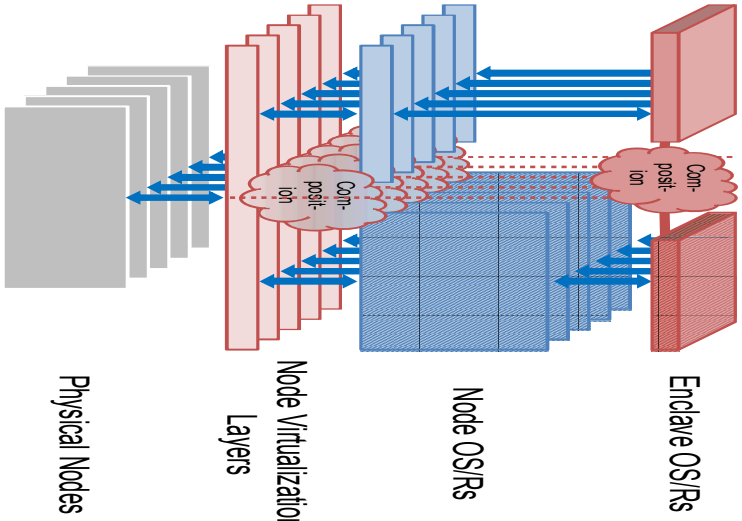
Sandia National Laboratories

Institution	Person	Role
Georgia Institute of Technology	Karsten Schwan	PI
Indiana University	Thomas Sterling	PI
Los Alamos National Lab	**Mike Lang**	**PI**
Lawrence Berkeley National Lab	Costin Iancu	PI
North Carolina State University	Frank Mueller	PI
Northwestern University	**Peter Dinda**	**PI**
Oak Ridge National Laboratory	David Bernholdt	PI
Oak Ridge National Laboratory	Arthur B. Maccabe	Chief Scientist
Sandia National Laboratories	**Ron Brightwell**	**Coordinating PI**
University of Arizona	David Lowenthal	PI
University of California – Berkeley	Eric Brewer	PI
University of New Mexico	**Patrick Bridges**	**PI**
University of Pittsburgh	**Jack Lange**	**PI**

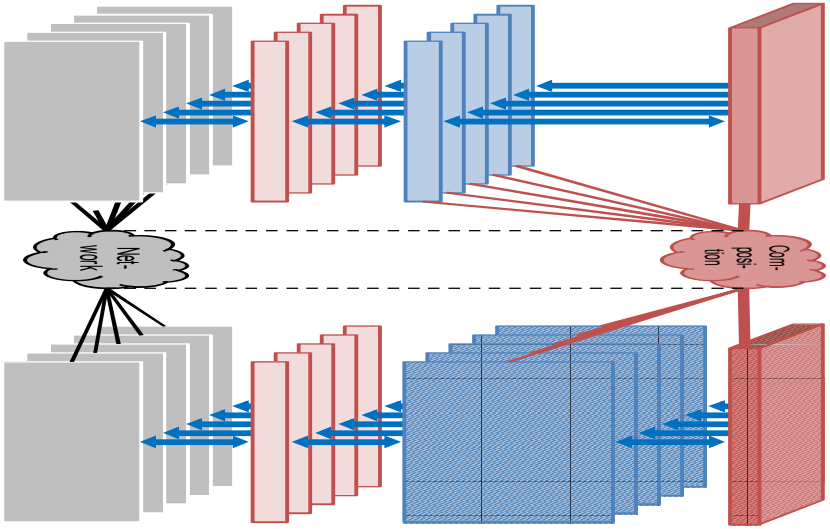
Hobbes: Composition Examples

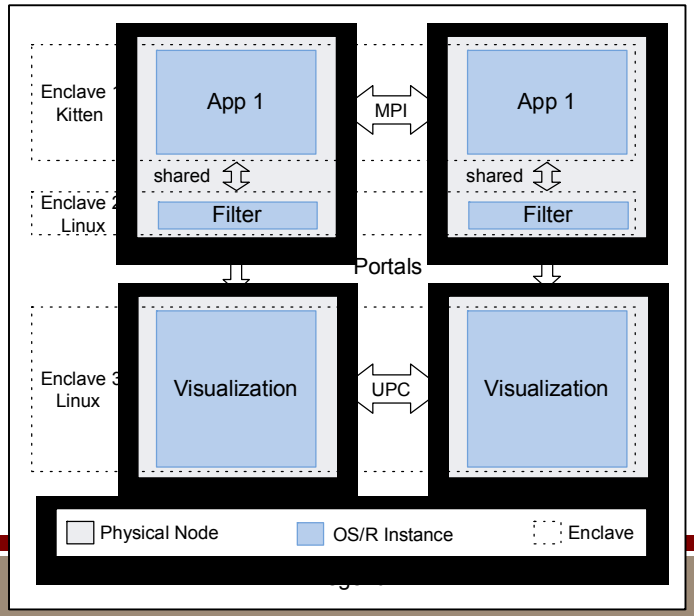
Sandia National Laboratories

Intra-node Composition



Inter-node Composition





Example Use Cases:

- Coupling CTH + Paraview/Catalyst on same node
  - CTH has few OS/R requirements
  - Paraview/Catalyst has some “full-OS” dependencies
  - Like previous in-transit case, but co-located like in-situ
- Coupling high fidelity simulation and low fidelity model
  - Useful for combustion and fusion examples
  - Tight coupling or loose coupling, elastic enclaves
- CASL multiphysics coupling, massive collisions
- LAMMPS and SmartPointer Analysis Pipeline
- Goldrush-style cycle stealing for analysis

Why Node Virtualization Layer?

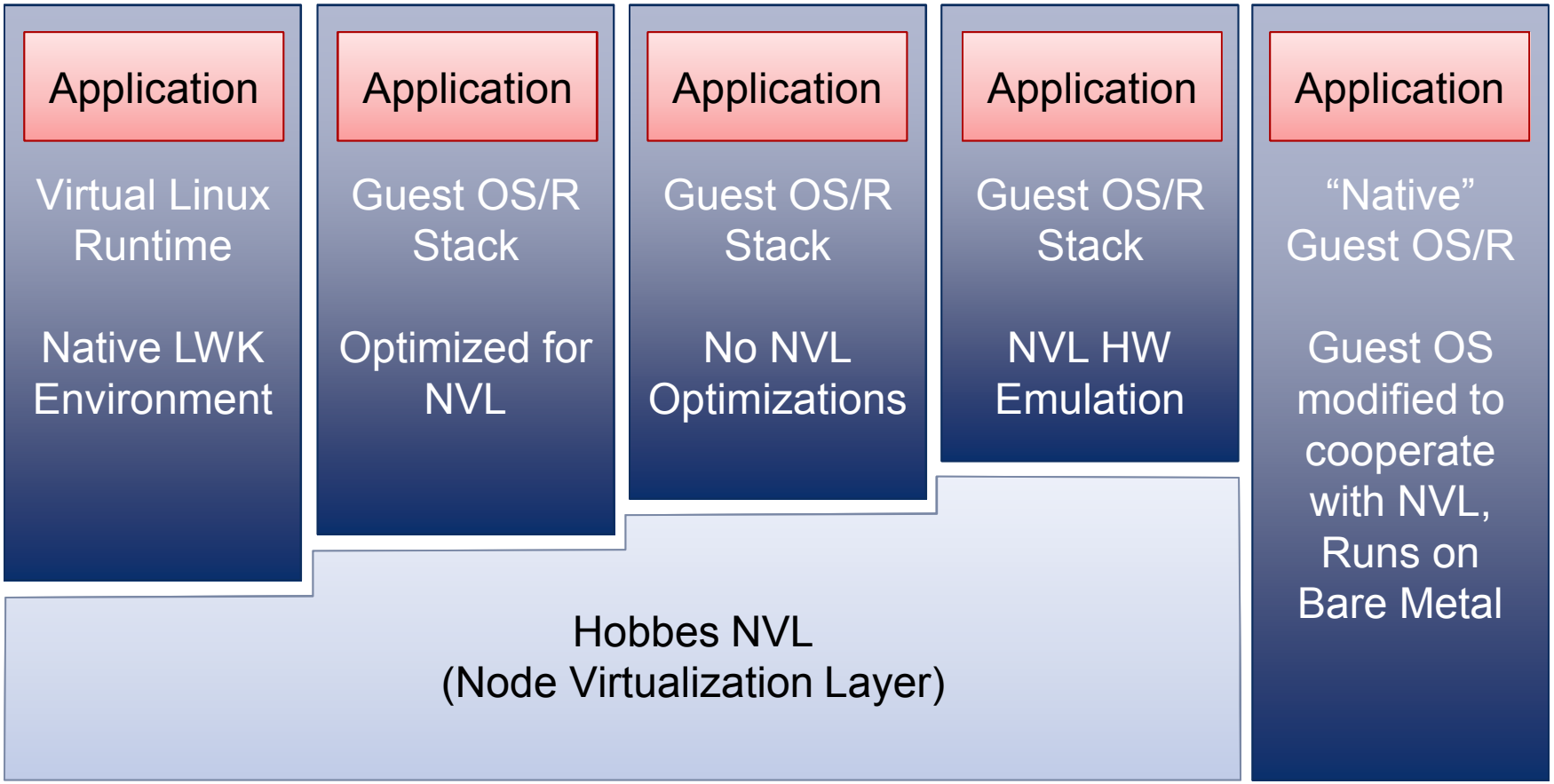
Sandia National Laboratories

- Flexibility, support multiple OS/R stacks simultaneously
  - There is likely to be no one-size-fits-all OS/R stack, lots of exploration
  - Co-location of VMs, efficient sharing of resources between enclaves
  - Native environment freed from legacy constraints
- Low overhead
  - Our past work has shown CPU and memory overheads negligible
  - Network I/O is still an issue, but tractable
- Industry momentum
  - Virtualization has been commoditized, is everywhere
  - Academic and student mindshare, where the jobs are
- Mostly orthogonal to “FusedOS” approach others are taking
  - FusedOS could run in NVL VM or natively, in the same machine
  - NVL could be co-designed with FusedOS

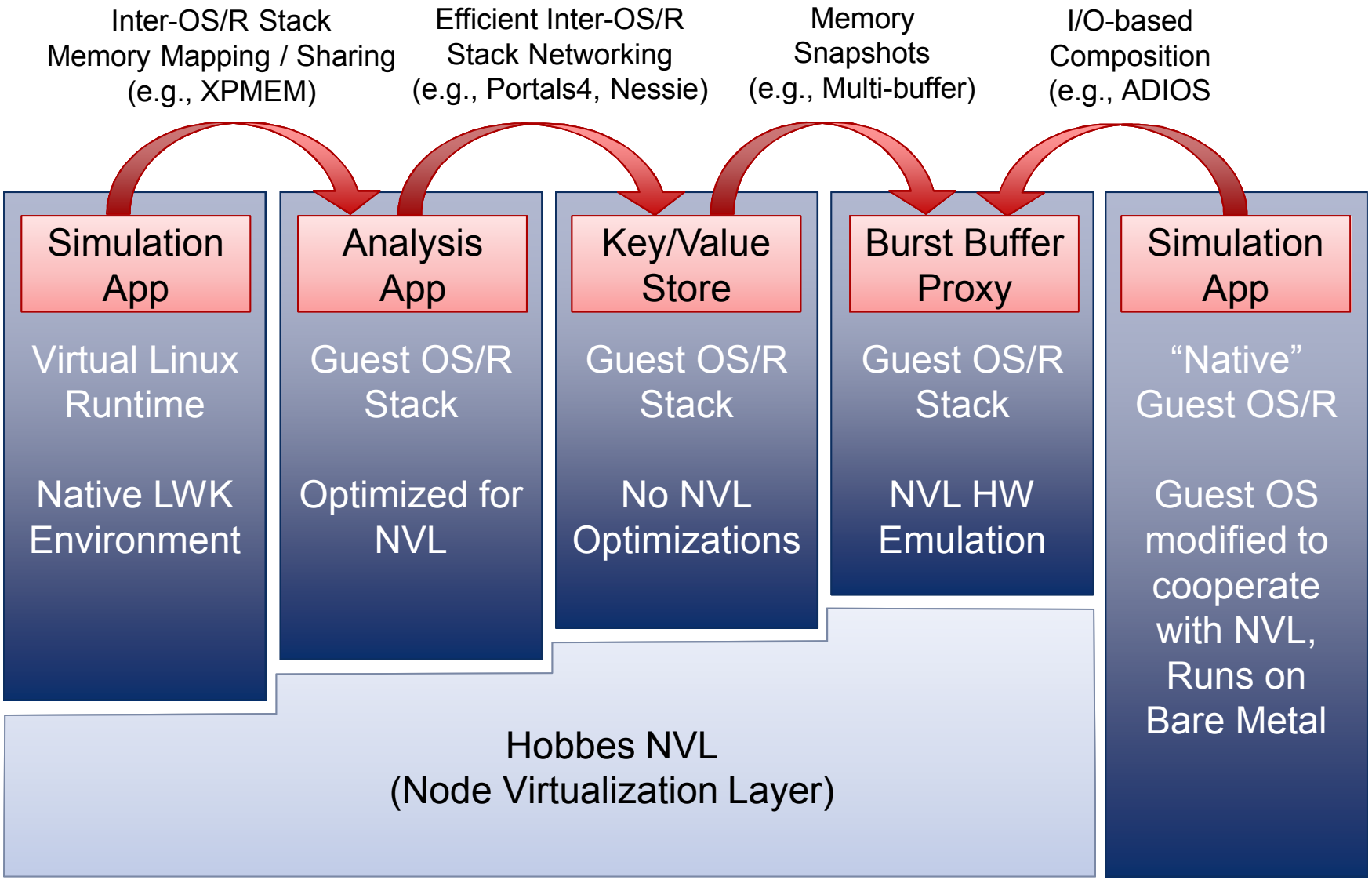
Hobbes NVL Has Multiple Levels of Virtualization

Sandia National Laboratories

- Existing Hypervisors typically support one level, strict isolation
- NVL couples LWK "native" runtime with guest OS/R stacks



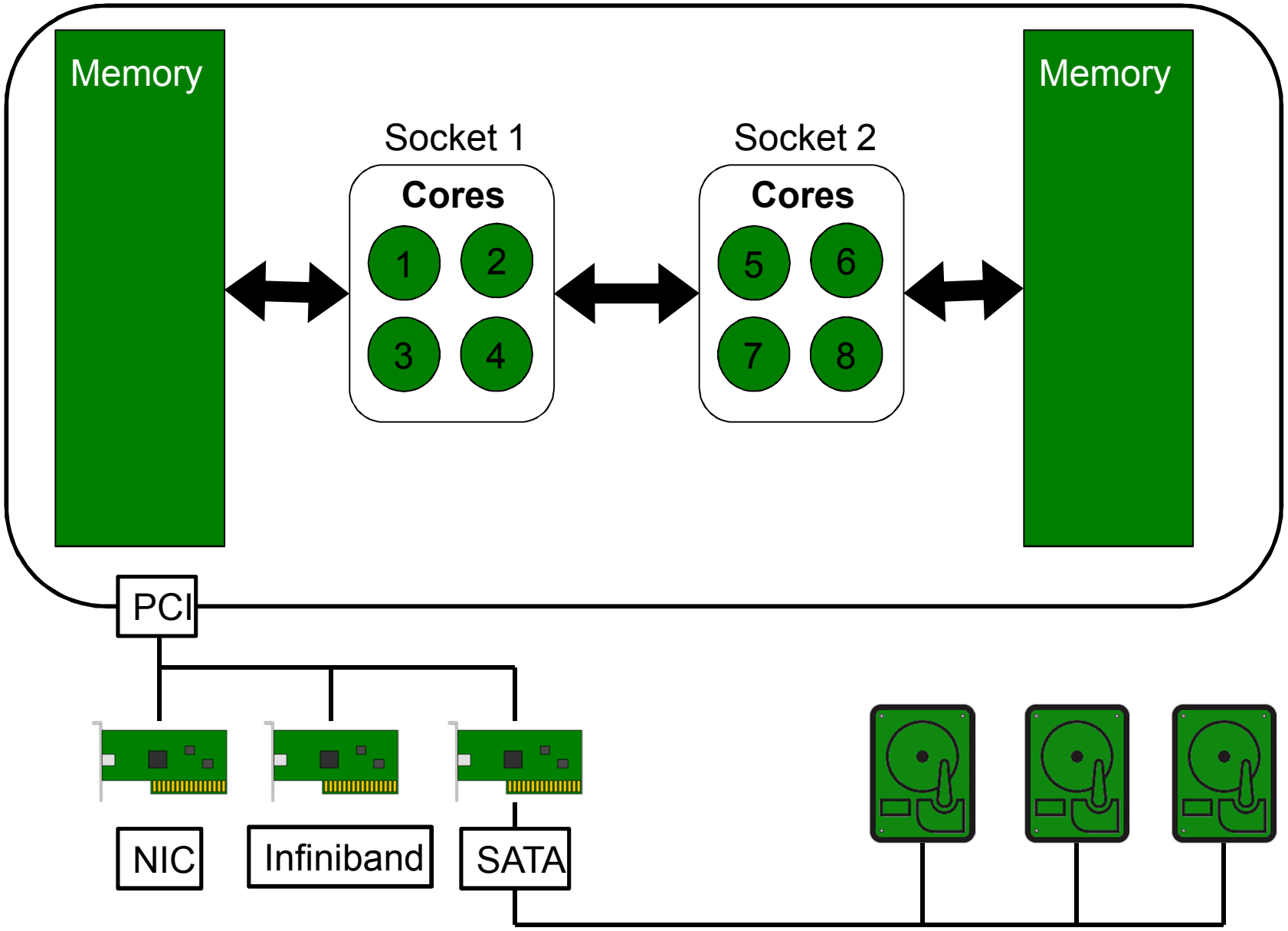
Hobbes NVL Provides Composition Mechanisms



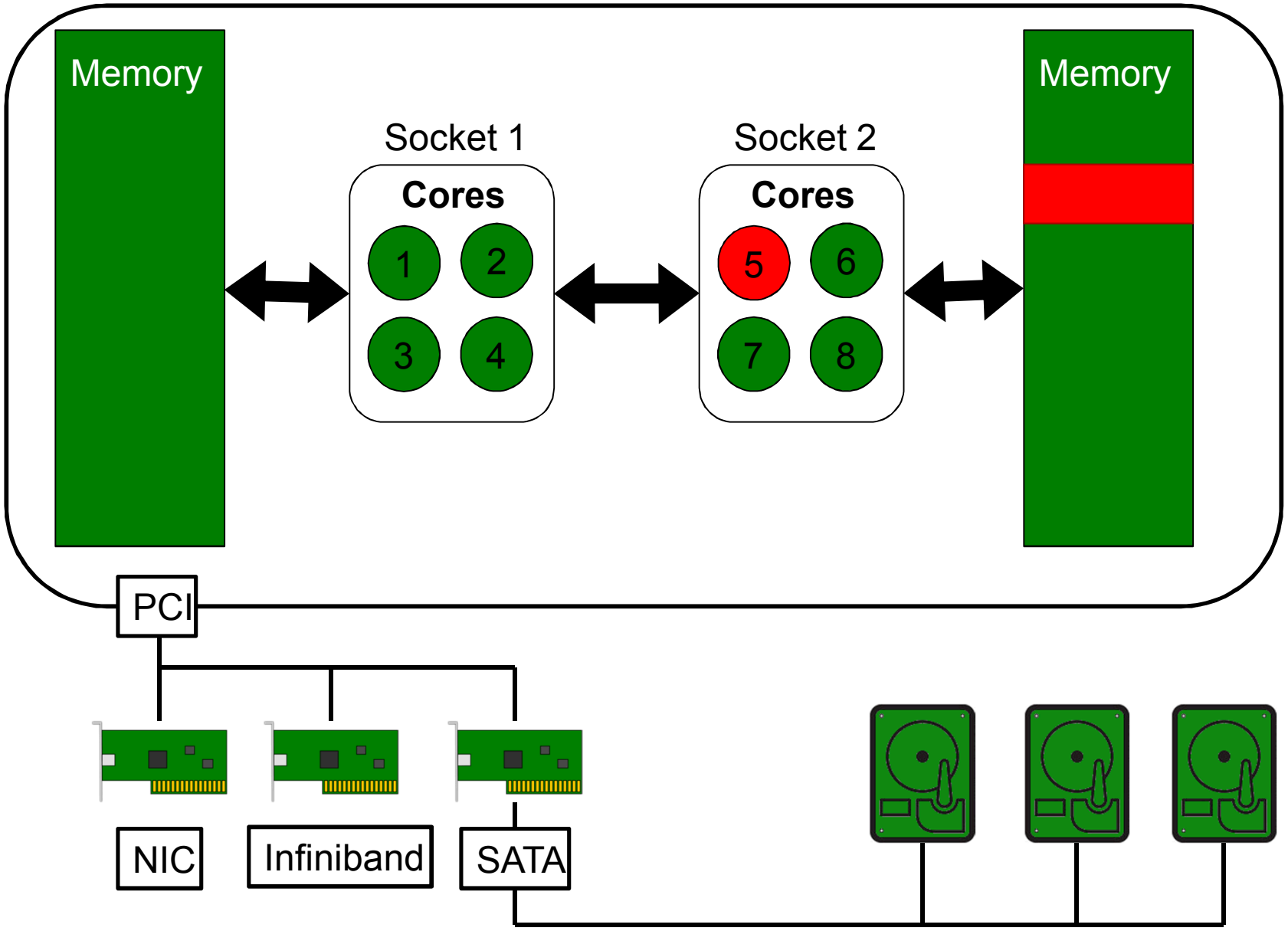
Hobbes: NVL Current Status

Sandia National Laboratories

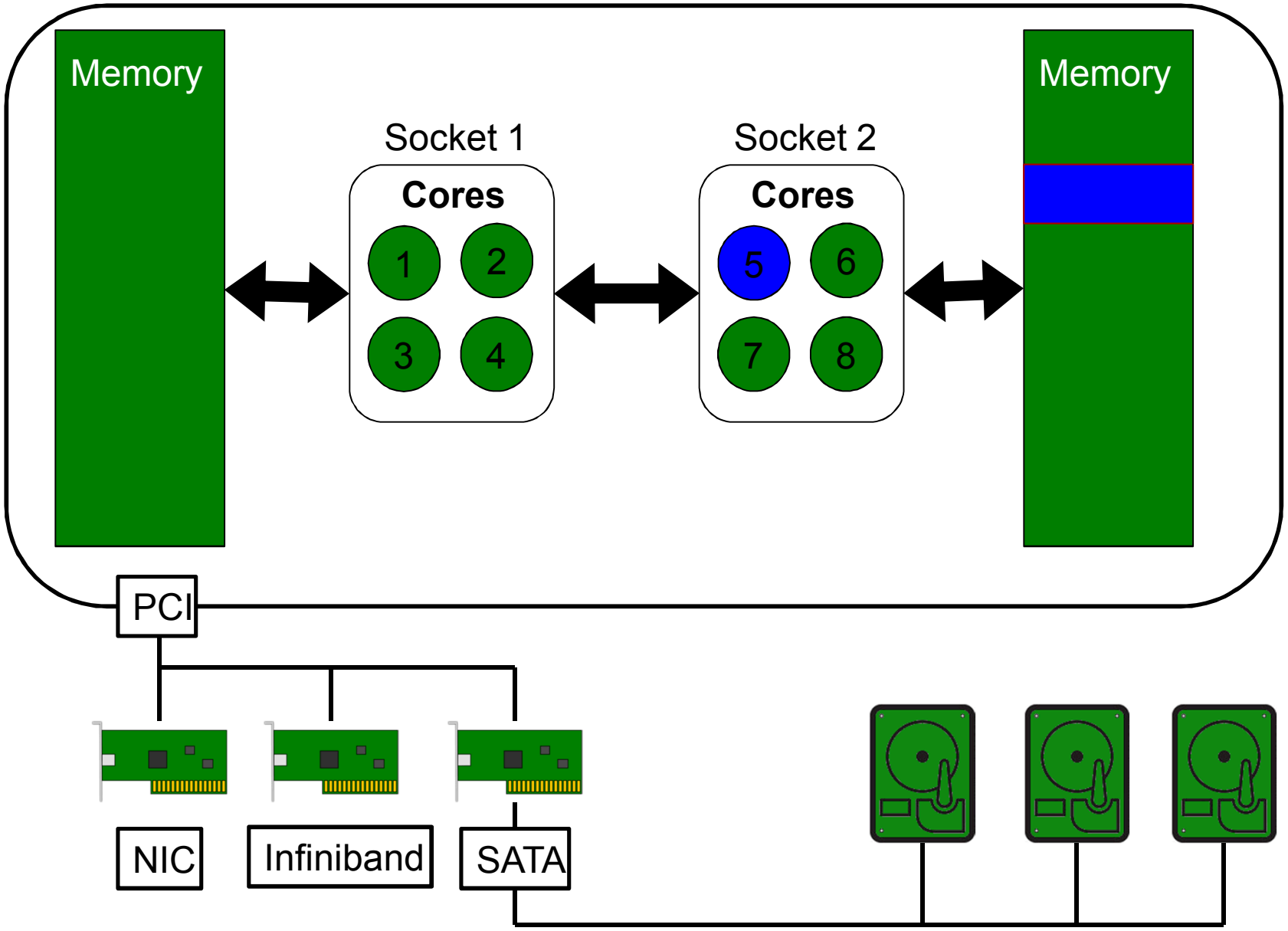
- NVL is booting on Cray XK6 Curie testbed at Sandia
  - Able to start multiple enclaves on a single node
  - Able to map memory between enclaves with XPMEM and TACSM
  - Networking is now critical item, working to expose Cray uGNI networking APIs to enclaves
- Getting ready for testing on ORNL/Titan Cray XK6
  - 30M hours through ALCC, ability to reboot system into NVL
  - Lining up test cases, mini-workflow examples
  - First tests planned in October timeframe, before SC
  - Interested in comparing three scenarios
    - Boot NVL native, run Cray Linux in NVL enclave (recreate VEE'11 tests)
    - Boot CNL native, uses Pisces to boot multiple NVL-managed enclaves
    - Boot NVL native, test native LWK environment at scale

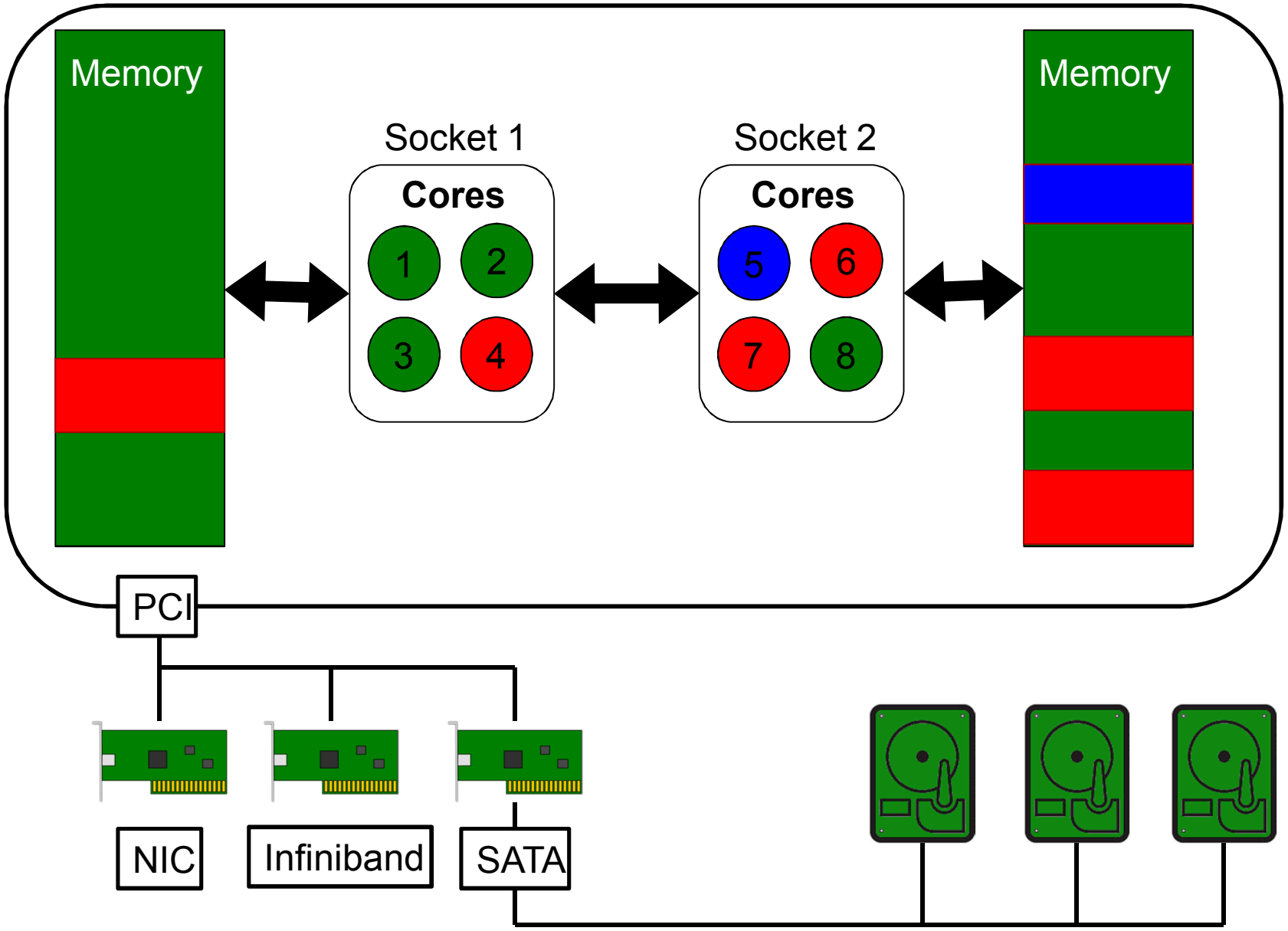


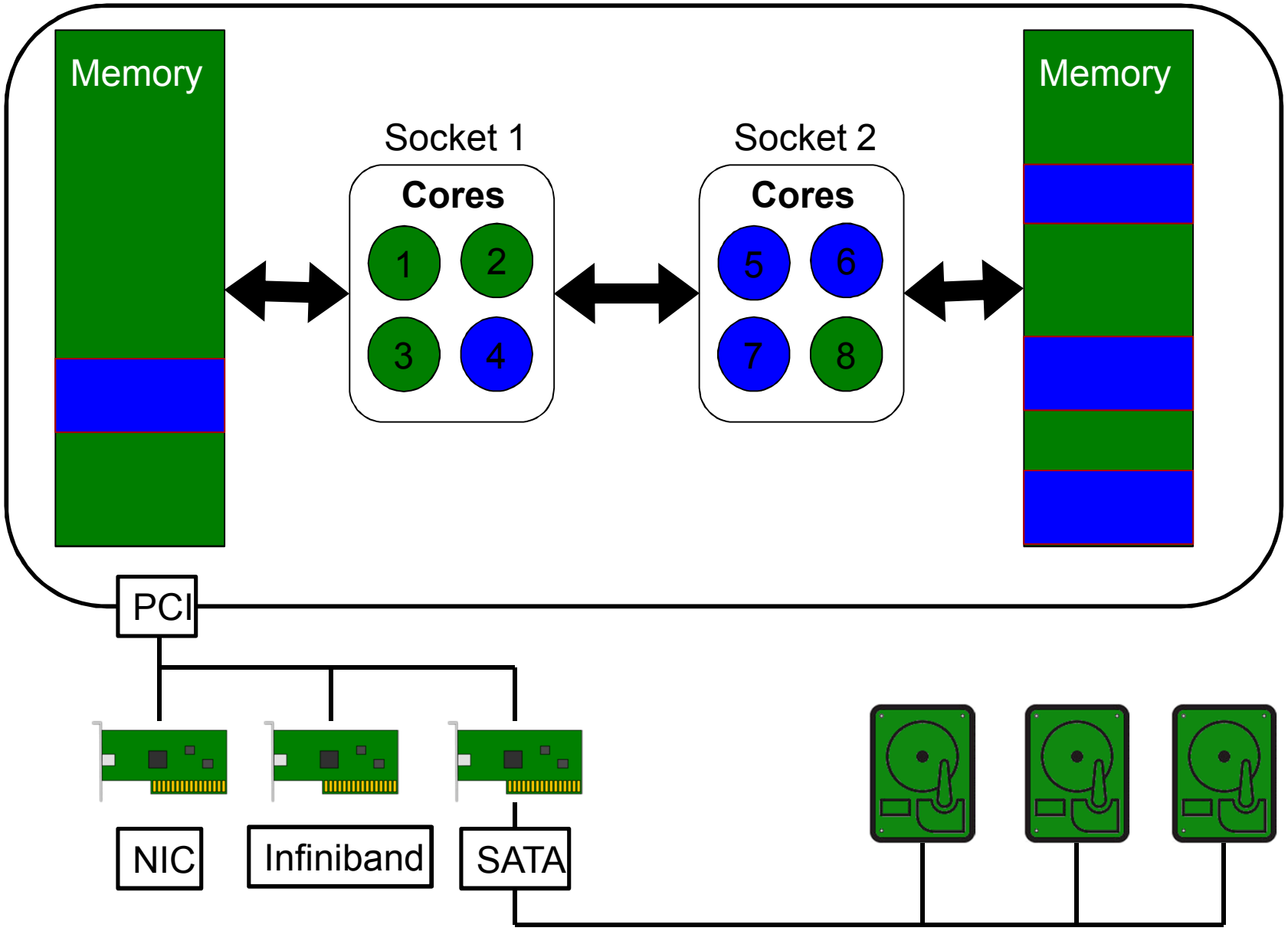
Linux Offline Kitten

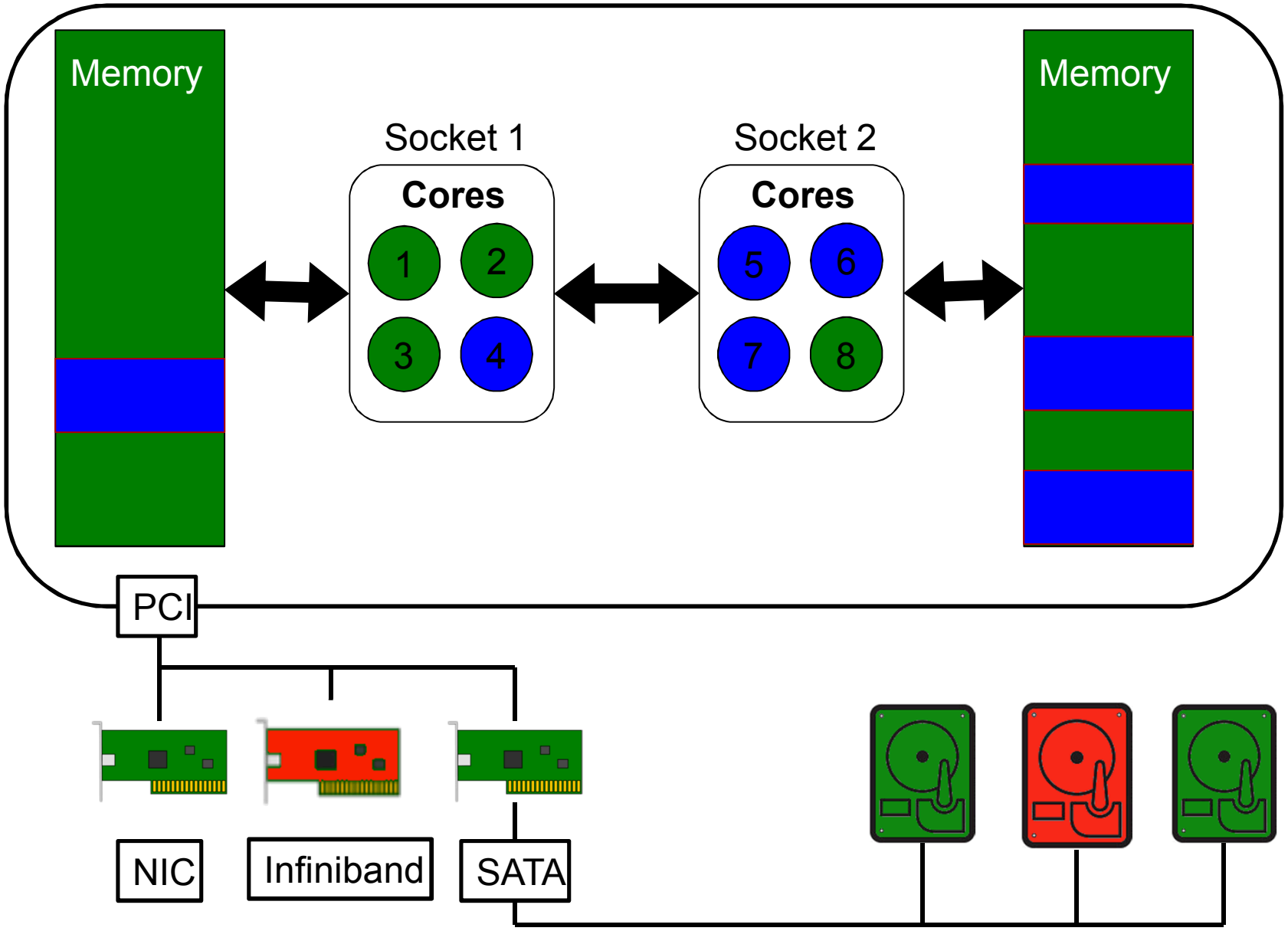


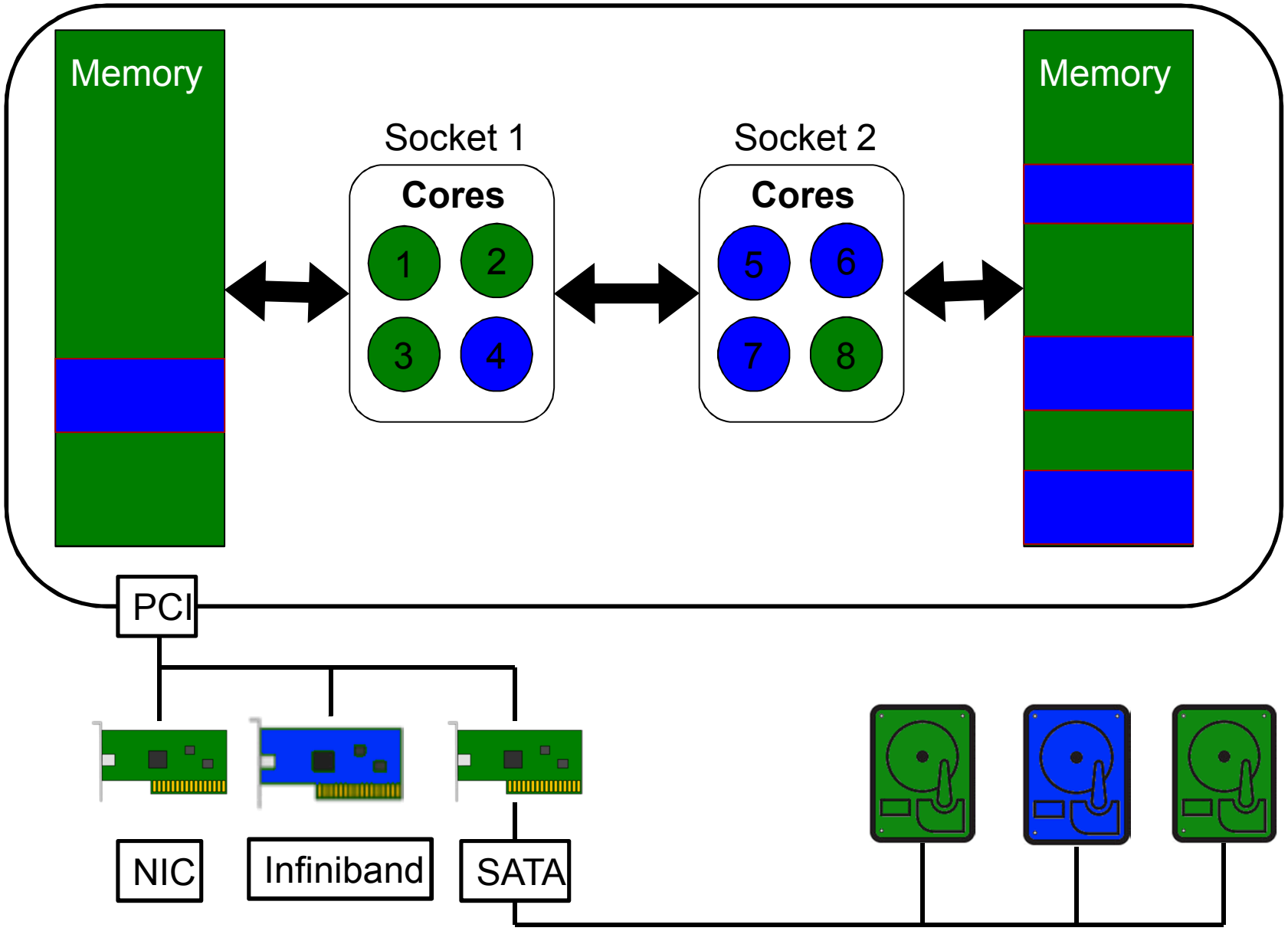
Linux Offline Kitten


Memory
Socket 1
Cores
1
2
3
4
Socket 2
Cores
5
6
7
8
Memory
PCI
NIC
Infiniband
SATA
Linux
Offline
Kitten


Memory
Socket 1
Cores
1
2
3
4
Socket 2
Cores
5
6
7
8
Memory
PCI
NIC
Infiniband
SATA
Linux
Offline
Kitten


Memory
Socket 1
Cores
1
2
3
4
Socket 2
Cores
5
6
7
8
Memory
PCI
NIC
Infiniband
SATA
Linux
Offline
Kitten

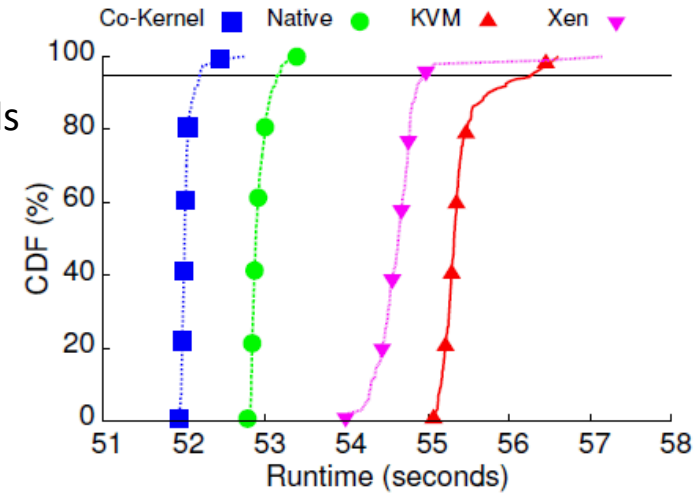

Memory
Socket 1
Cores
1
2
3
4
Socket 2
Cores
5
6
7
8
Memory
PCI
NIC
Infiniband
SATA
Linux
Offline
Kitten


Memory
Socket 1
Cores
1
2
3
4
Socket 2
Cores
5
6
7
8
Memory
PCI
NIC
Infiniband
SATA
Linux
Offline
Kitten

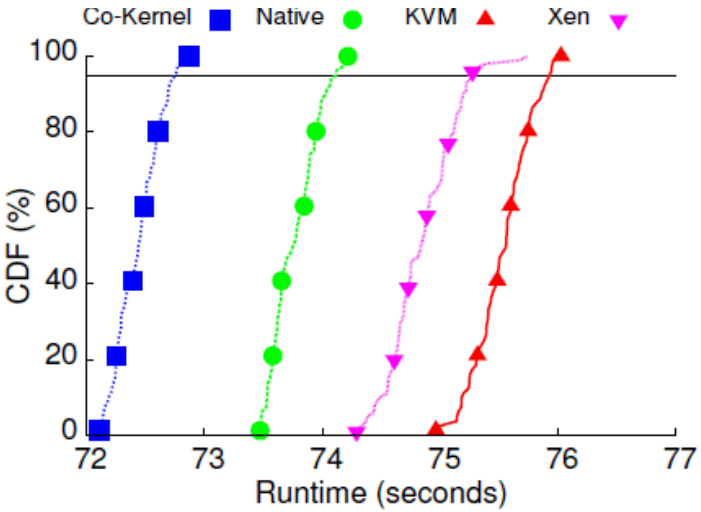
Pisces: A Framework for Composing Operating Systems

Sandia National Laboratories

- Problem
  - Mixing cloud and HPC workloads on the same system leads to sub-optimal performance for all
    - Linux has high run-to-run performance variability
    - Job interference when resources are oversubscribed
- Solution
  - Use Pisces framework to partition a node's resources, run different OS kernel in each partition
    - Commodity workloads run in Linux OS partition, HPC workloads run in lightweight kernel (LWK) Kitten OS partition
    - Linux and Kitten OS kernels run as co-kernels, modified to use Pisces framework for cooperation
- Recent Results
  - Pisces shown to improve performance, reduce variability
    - Ran HPCCG and CloverLeaf mini-apps, compared to native runs in Linux and guest runs in KVM and Xen hypervisors
    - Runtime CDFs shown in right plots (Co-Kernel = Pisces)
- Impact
  - Can deploy lightweight OS environments on commodity cloud platforms, achieve LWK performance and consistency



(a) HPCCG



(b) CloverLeaf

Outline

Sandia National Laboratories

- History of SNL Lightweight Kernel Operating Systems (LWK OS)
- Are LWKs still relevant?
- What we're working on
  - XPRESS project (FY13-FY15, DOE/ASCR X-Stack Program)
  - Hobbes project (FY14-FY16, DOE/ASCR OS/Runtime Program)
  - Power API and power management (CSSE, FY14 L2 Milestone)
  - Task mapping (LDRD + CSSE)
- Conclusion

Power Will Limit Performance

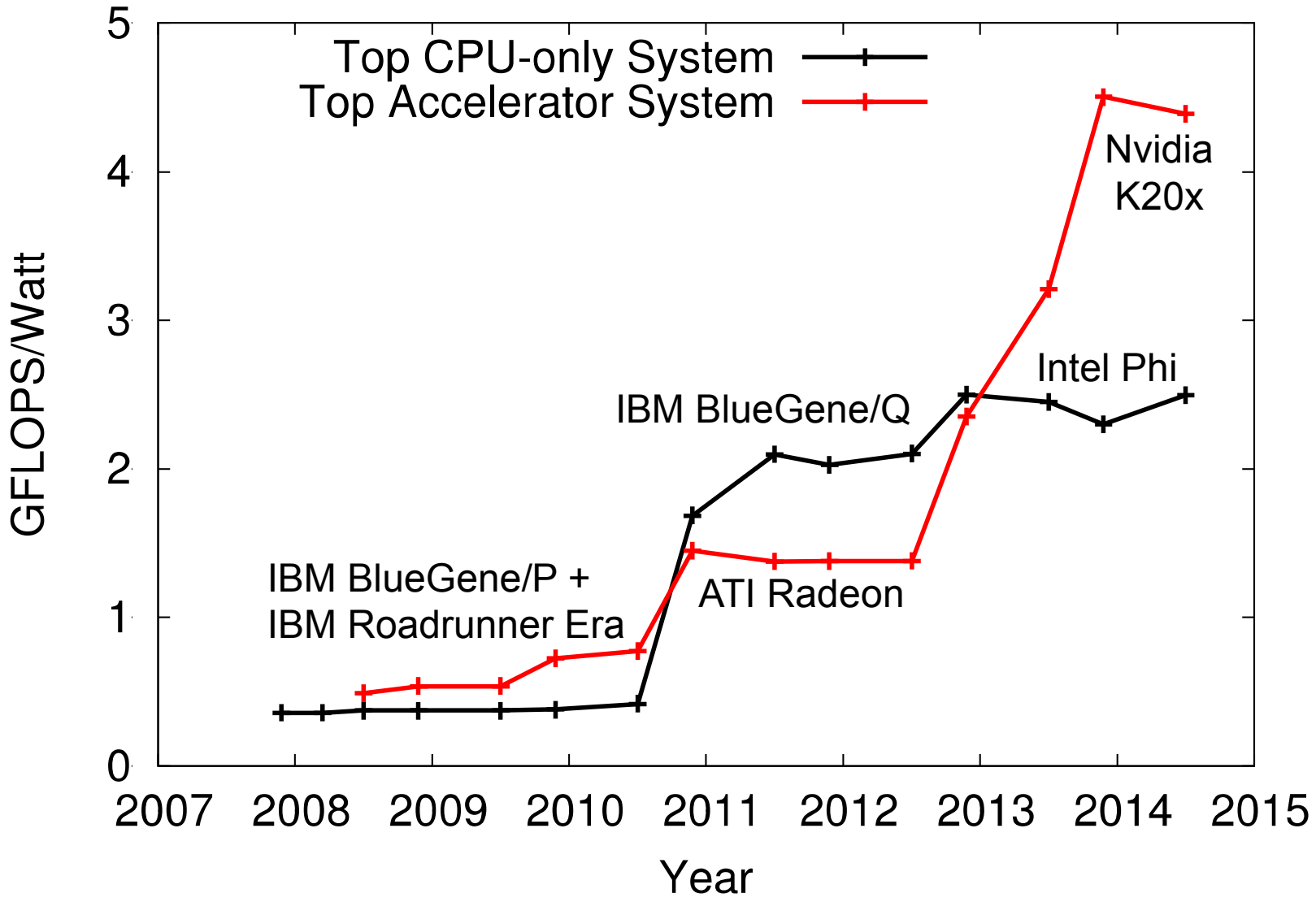
Sandia National Laboratories

- DOE is shooting for 20 Megawatts (MW) for an ExaFLOPS
- At $0.10 per KWh, each MW costs $876K per year
  - Rule of thumb => $1M per MW-year
- 1 EFLOPS system @ 20 MW = 50 GFLOPS/W
  - HPL on state-of-the-art 12-core Intel Ivy Bridge => **1.4 GFLOPS/W**
  - HPCG on same system => **0.045 GFLOPS/W**
  - HPCG off by a factor of > 1000x, HPL off by a factor of ~36x
- Hardware trends
  - Dynamic voltage and frequency scaling (DVFS) $P \propto C * F * V^2$
  - Finer-grained power control (e.g., integrated VRM on Haswell)
  - Power capping – do not exceed wattage for some time window
  - Power measurement capabilities – power and energy counters
  - Active power management – on-die power processor + firmware
  - Dark silicon – what hardware should I turn on? Who decides?

OS + Runtime (+App?) Will Have To Manage Power Budgets

Power Efficiency Trends (Green500)

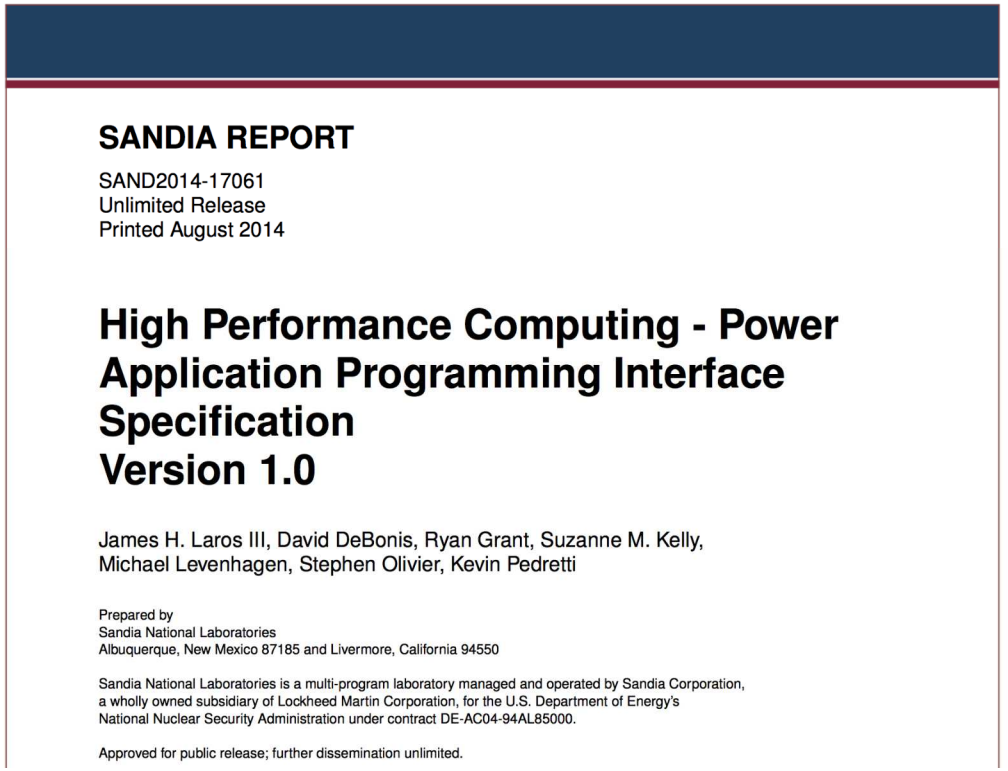
Sandia National Laboratories



PowerAPI Fills an Important Gap

Sandia National Laboratories

- Need a portable API for measuring and controlling power
  - Today there are several power interfaces, every system is different
  - This makes it hard to write tools, add power measurement to apps, …
- SNL developed a Power API specification to fill this gap, with input from vendor community (FY14 L2 milestone)
  - Covers broad spectrum of use cases, from platform-level, to resource manager, to runtime system, to OS, to applications
  - Will be implemented for upcoming Trinity system
  - Expect to remain available on future DOE/NNSA ATS systems
  - Will evolve over time



SANDIA REPORT

SAND2014-17061
Unlimited Release
Printed August 2014

High Performance Computing - Power Application Programming Interface Specification Version 1.0

James H. Laros III, David DeBonis, Ryan Grant, Suzanne M. Kelly, Michael Levenhagen, Stephen Olivier, Kevin Pedretti

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.

Power API Has Vendor Input

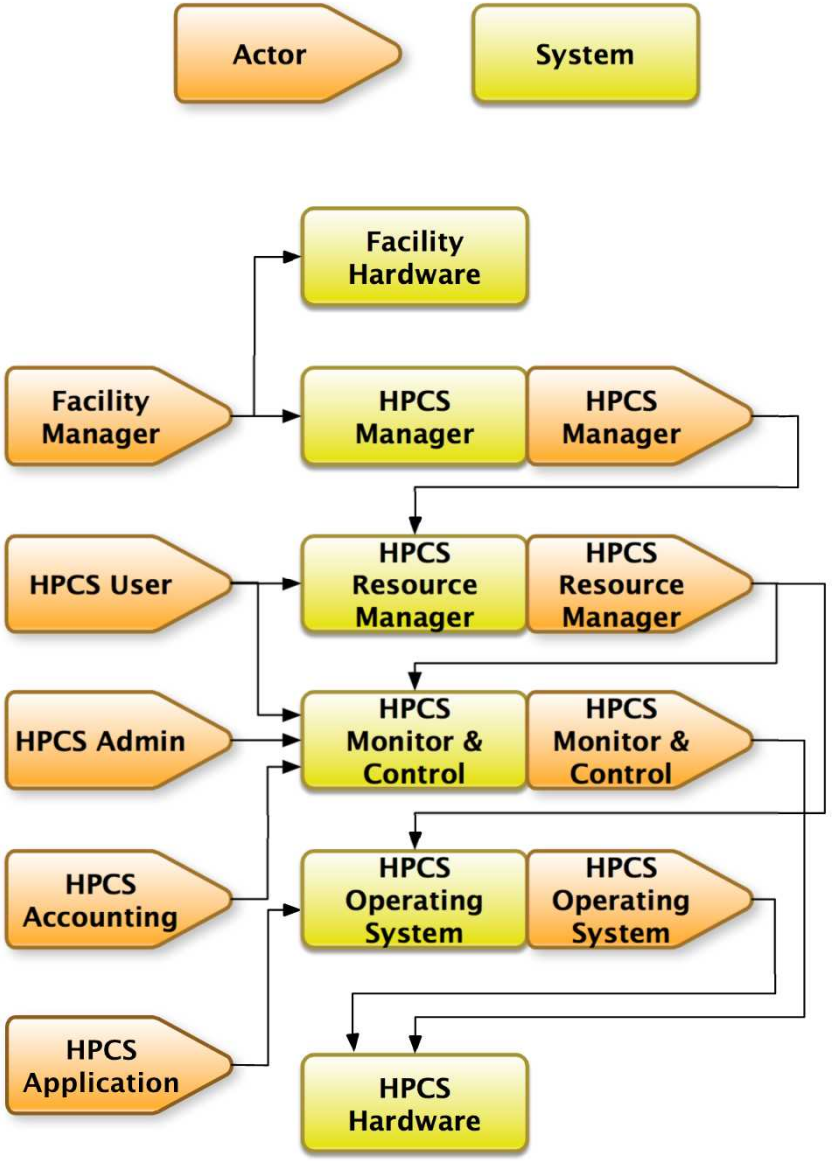
Sandia National Laboratories

- Held initial review meeting with vendors July 2014 at CSRI
- Wider-audience meeting in Denver for public comment September 2014

Power API Covers Many Use Cases

Sandia National Laboratories

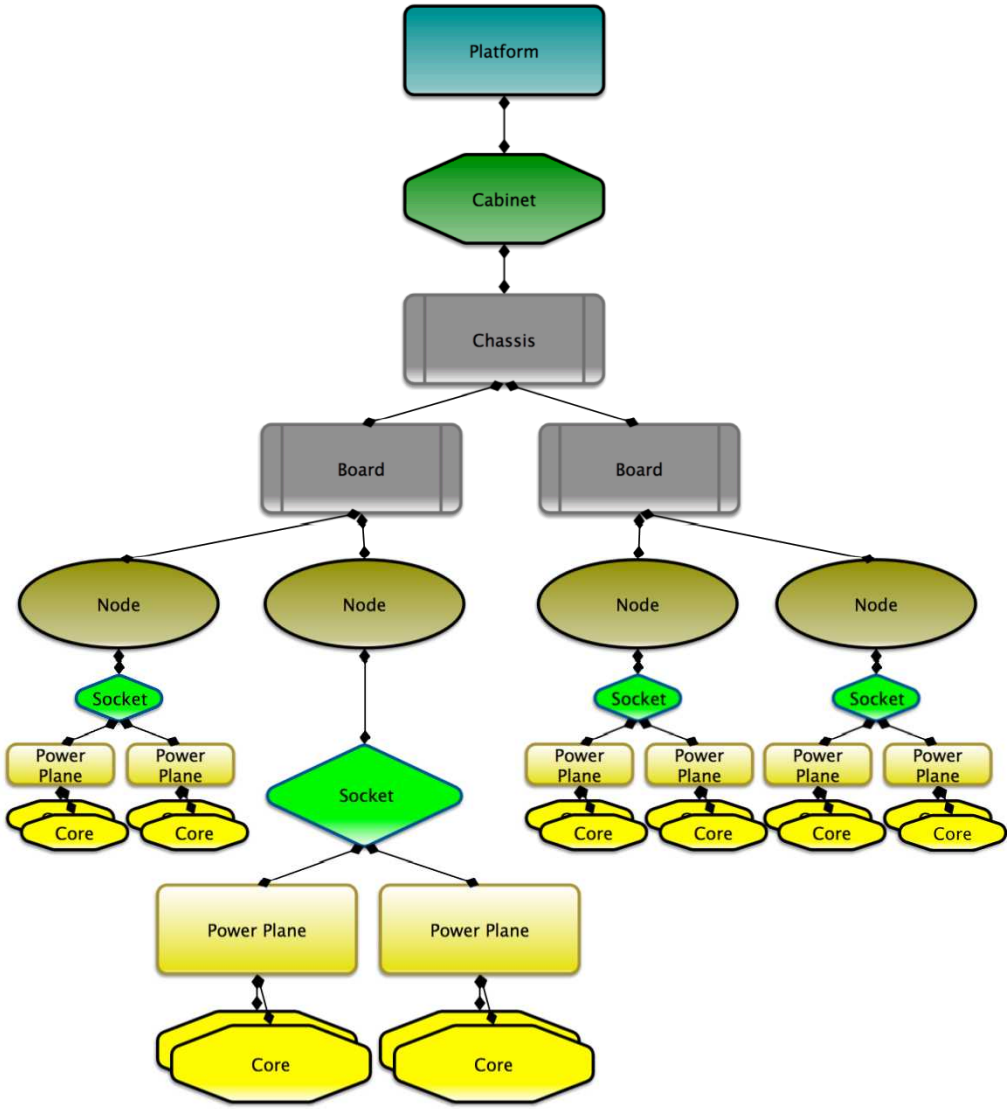
- Multiple actors can interact with the system at different levels
- Each interaction represents an interface that is defined in the PowerAPI



Power API Provides an Object Model

Sandia National Laboratories

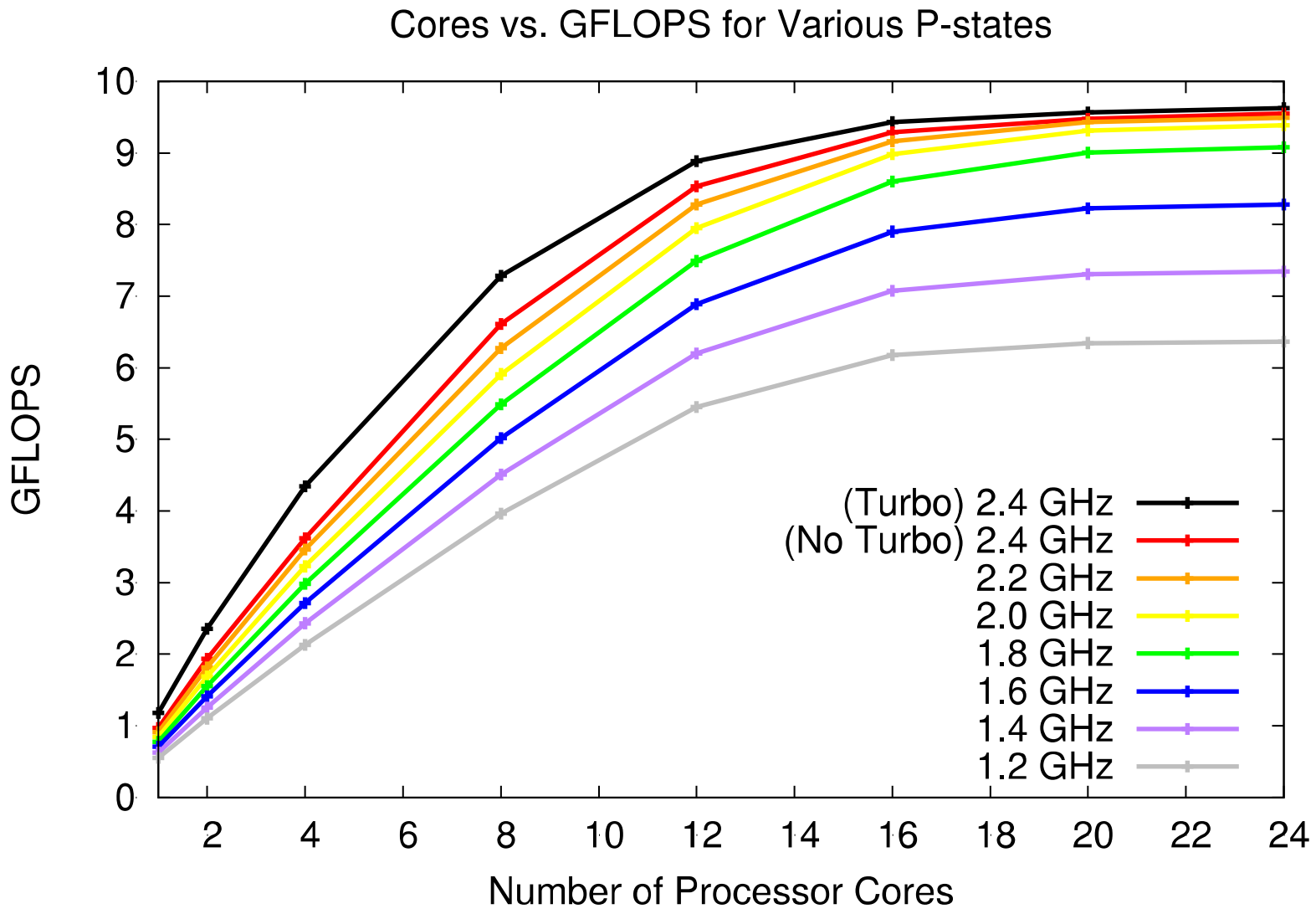
- Hierarchy of objects, discovered at runtime
  - Base system description
  - Vendors extended description, vendors can add their own objects
- Set/Get attributes on objects:
  - P-state, C-state, S-state, Power, Current, Voltage, Max_Power, Min_Power, Frequency, Energy, Temp
- Metadata interface to provide detail on what an attribute means (e.g., the accuracy of measurement)



HPCG on Volta Cray XC30, GFLOPS

Sandia National Laboratories

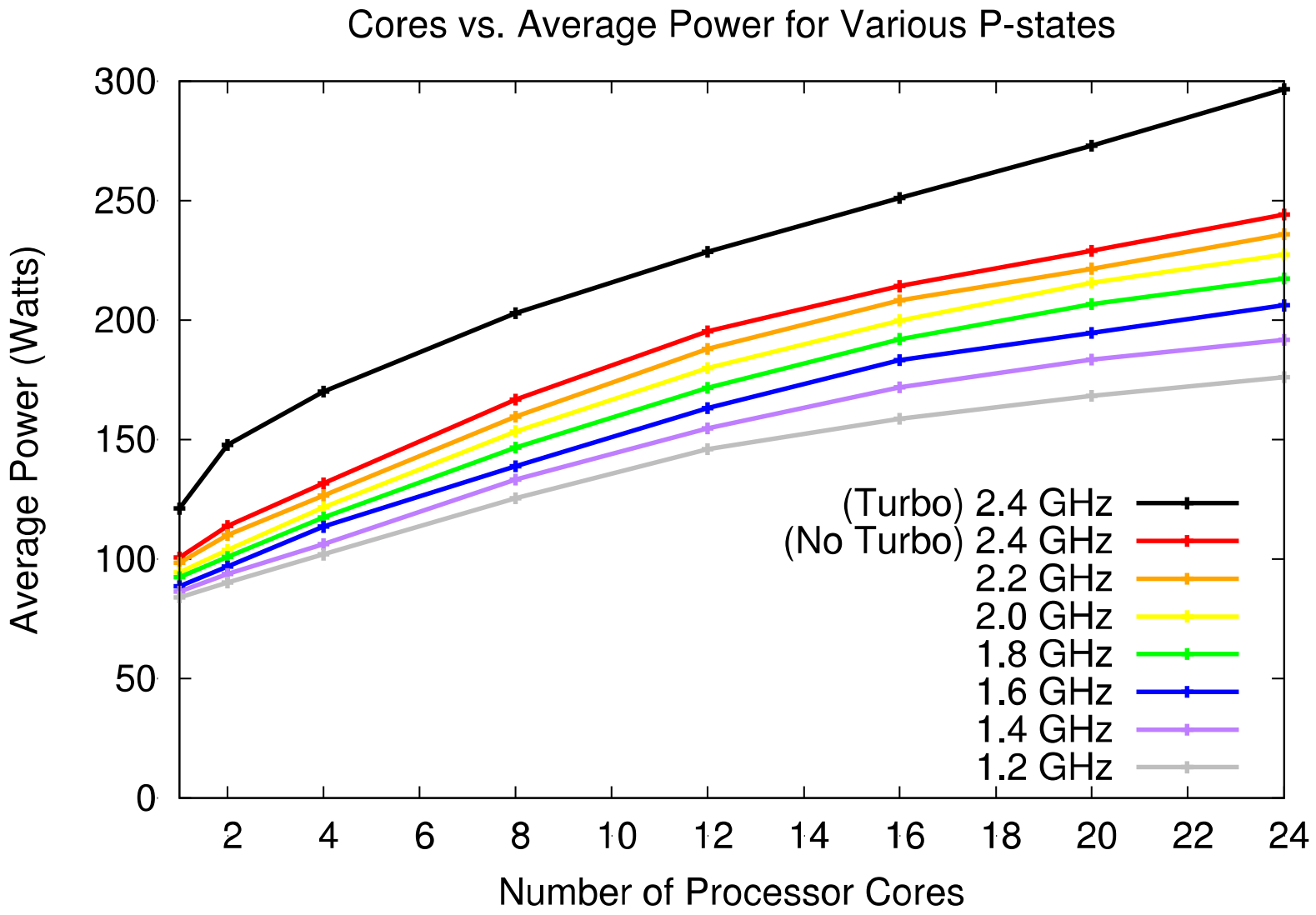
1 MPI Process Per Core, Weak Scaling, 104x104x104, 600 second run
1 Volta Compute Node = Intel 12-core Ivy Bridge 2.4 GHz (E5-2695 v2) x 2 sockets
Peak: 460.8 GFLOPS, ~ 80 GB/s memory bandwidth



HPCG on Volta Cray XC30, Power

Sandia National Laboratories

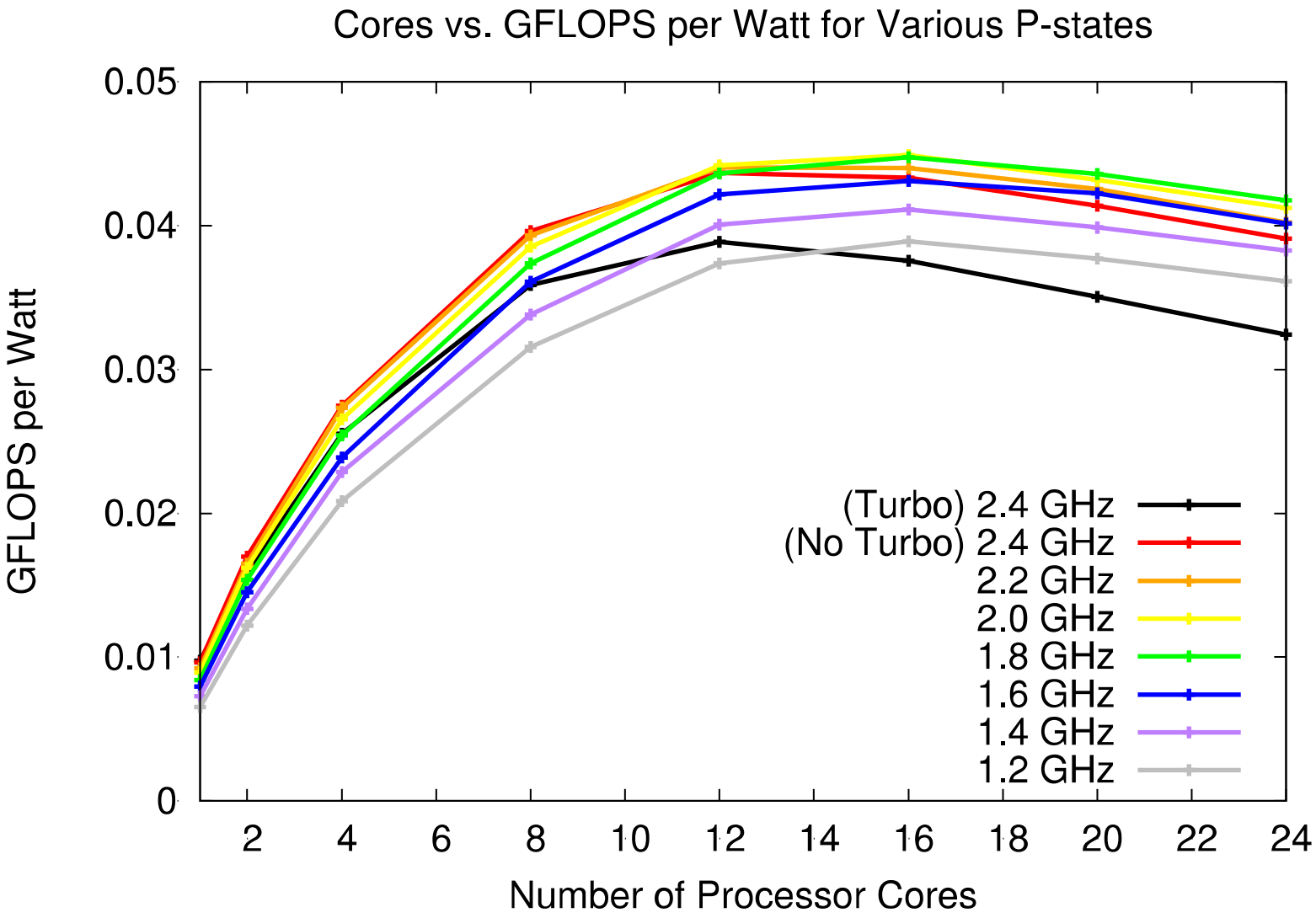
Average Power increases with P-state frequency
Turbo-boost is an outlier, maximum Turbo-boost frequency is 3.2 GHz



HPCG on Volta Cray XC30, GFLOPS/W

Sandia National Laboratories

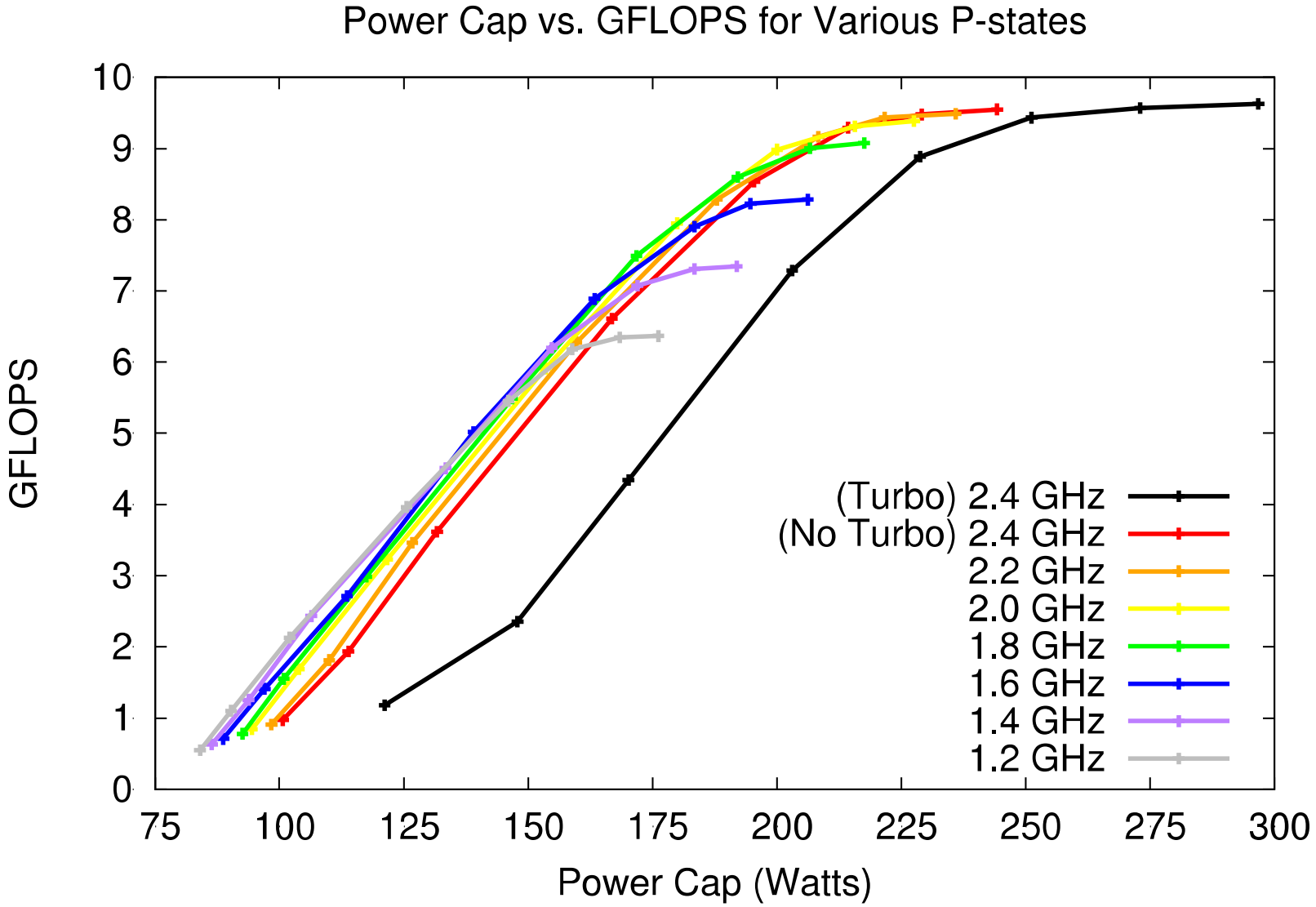
Curve rolls over, optimal point at 16 cores, 1.8 GHz
HPCG stresses memory system, shows memory system fails to feed all cores



HPCG on Volta Cray XC30, Power Cap

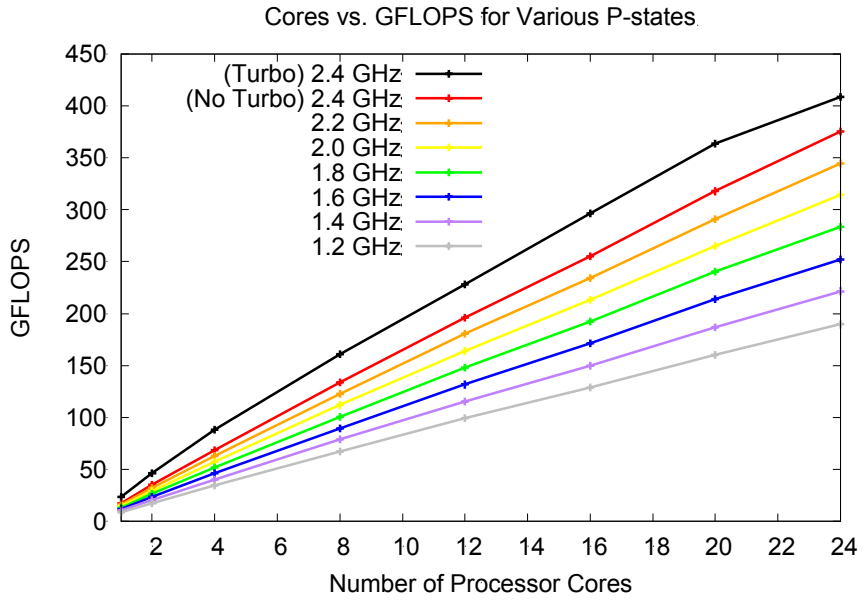
Sandia National Laboratories

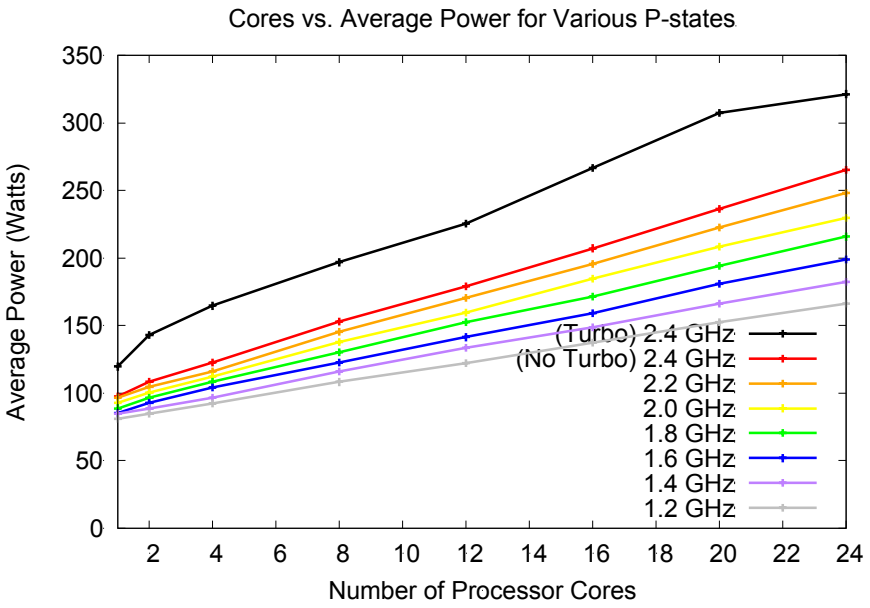
Plot shows maximum performance obtainable for a given power budget.
For example if 135 W cap, should run @ 1.4 GHz with 8 cores (Turbo would be silly)
This plot will vary part to part due to manufacturing variability, TBD how much

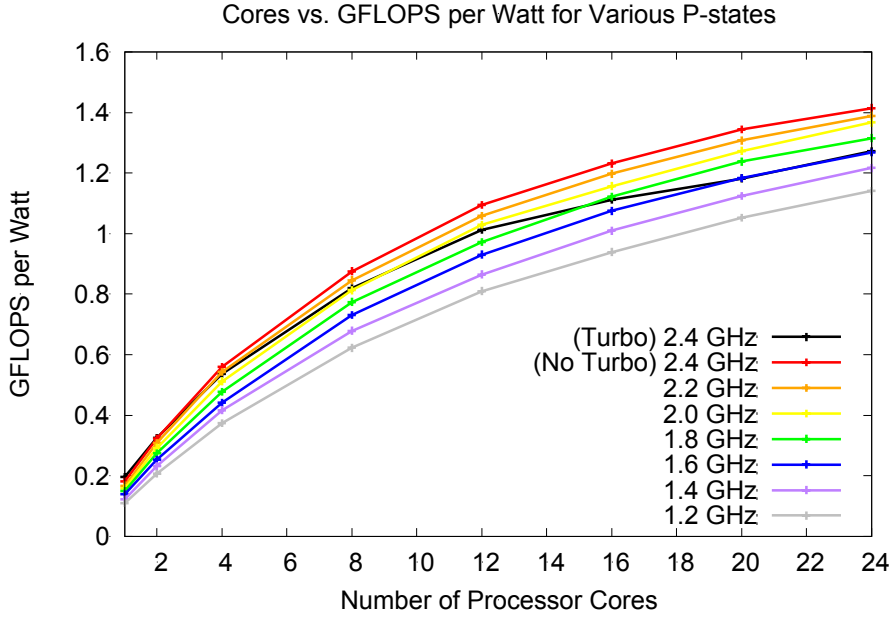


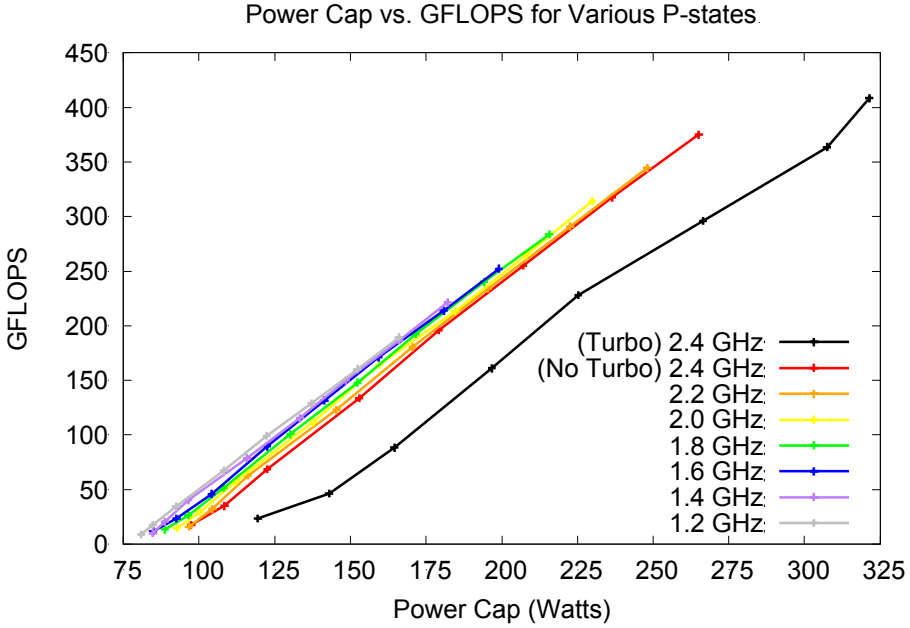
HPL on Volta Cray XC30

Sandia National Laboratories









Power Takeaways

Sandia National Laboratories

- PowerAPI provides portable power measurement and control interfaces, covers full spectrum of platform to application
- Working to understand power usage of our applications
- Working to understand software-implications of power mgmt.
  - MPI model fairly static, time consuming to find optimal operating point
  - Task-based runtime systems have more elasticity; can dynamically apply DVFS, reallocate power budgets, and/or shut things off based on observed behavior and system-level power cap reconfigurations
  - OS must provide suitable mechanisms to runtime system(s), possibly coordinate between multiple runtime systems
- Goal is to achieve maximum performance subject to a given power constraint; expect to be power limited in near future
  - Upcoming Trinity system will not be power-limited
  - Trinity contract includes NRE $ to get ready for power-limited, collaboration of Cray, SNL, and LANL (SNL PI: Laros)

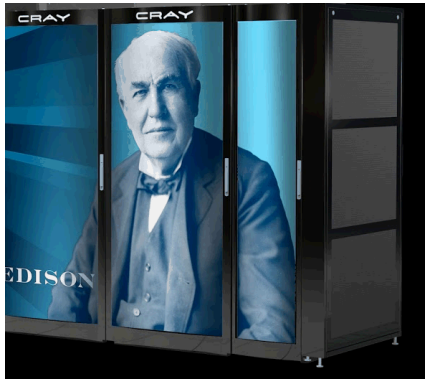
Outline

Sandia National Laboratories

- History of SNL Lightweight Kernel Operating Systems (LWK OS)
- Are LWKs still relevant?
- What we're working on
  - XPRESS project (FY13-FY15, DOE/ASCR X-Stack Program)
  - Hobbes project (FY14-FY16, DOE/ASCR OS/Runtime Program)
  - Power API and power management (CSSE, FY14 L2 Milestone)
  - Task mapping (LDRD + CSSE)
- Conclusion

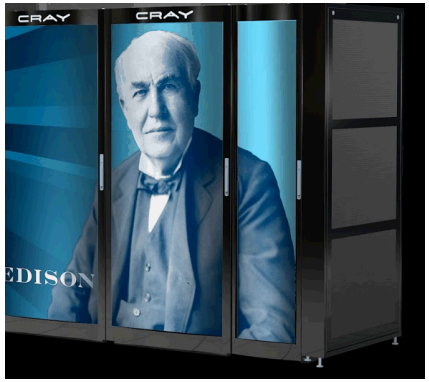
Scalable Networks Are Sparse

Sandia National Laboratories

1997 – 2006 SNL ASCI Red	2004 - 2012 SNL Red Storm	2011 – ACES Cielo	2013 – NERSC Edison
			
Intel Custom Network	Cray XT3 SeaStar	Cray XE6 Gemini	Cray XC30 Aries
3-D Mesh	3-D Mesh	3-D Torus	Dragonfly
38 x 32 x 2	27 x 20 x 24	16 x 12 x 24	3-Levels: 16, 6, 14
4510 Nodes	12960 Nodes	8944 Nodes	5192 Nodes
3.15 TFLOPS/s	284 TFLOP/s	1374 TFLOP/s	2390 TFLOP/s

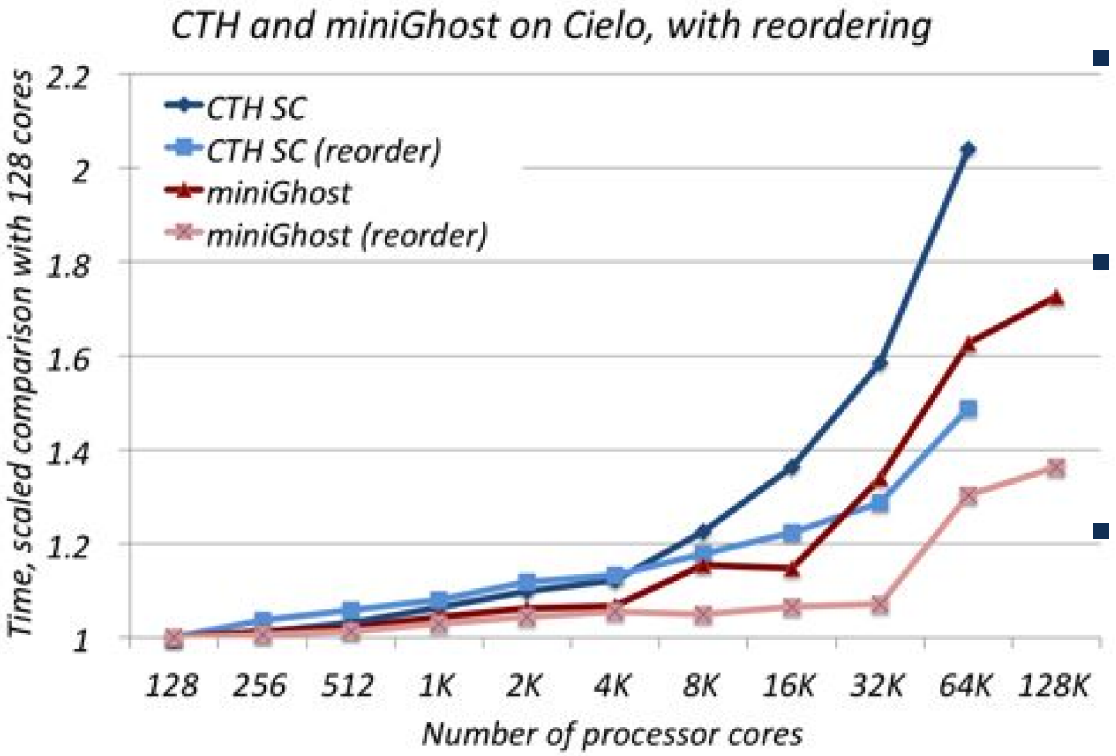
BW / Injection Ratios Getting Worse

Sandia National Laboratories

1997 – 2006 SNL ASCI Red	2004 - 2012 SNL Red Storm	2011 – ACES Cielo	2013 – NERSC Edison
			
Intel	SeaStar / 3D Mesh	Gemini / 3D Torus	Aries / Dragonfly
Total Node Injection: 1443 GB/s	22 TB/s	55 TB/s	48 TB/s
Total Network (all links): 4752 GB/s	357 TB/s	281 TB/s	156 – 204 TB/s
Ratio: 3.3	**16.2**	**5.1**	**3.3 – 4.25**

Example Case of "Bad" Task Mapping

Sandia National Laboratories

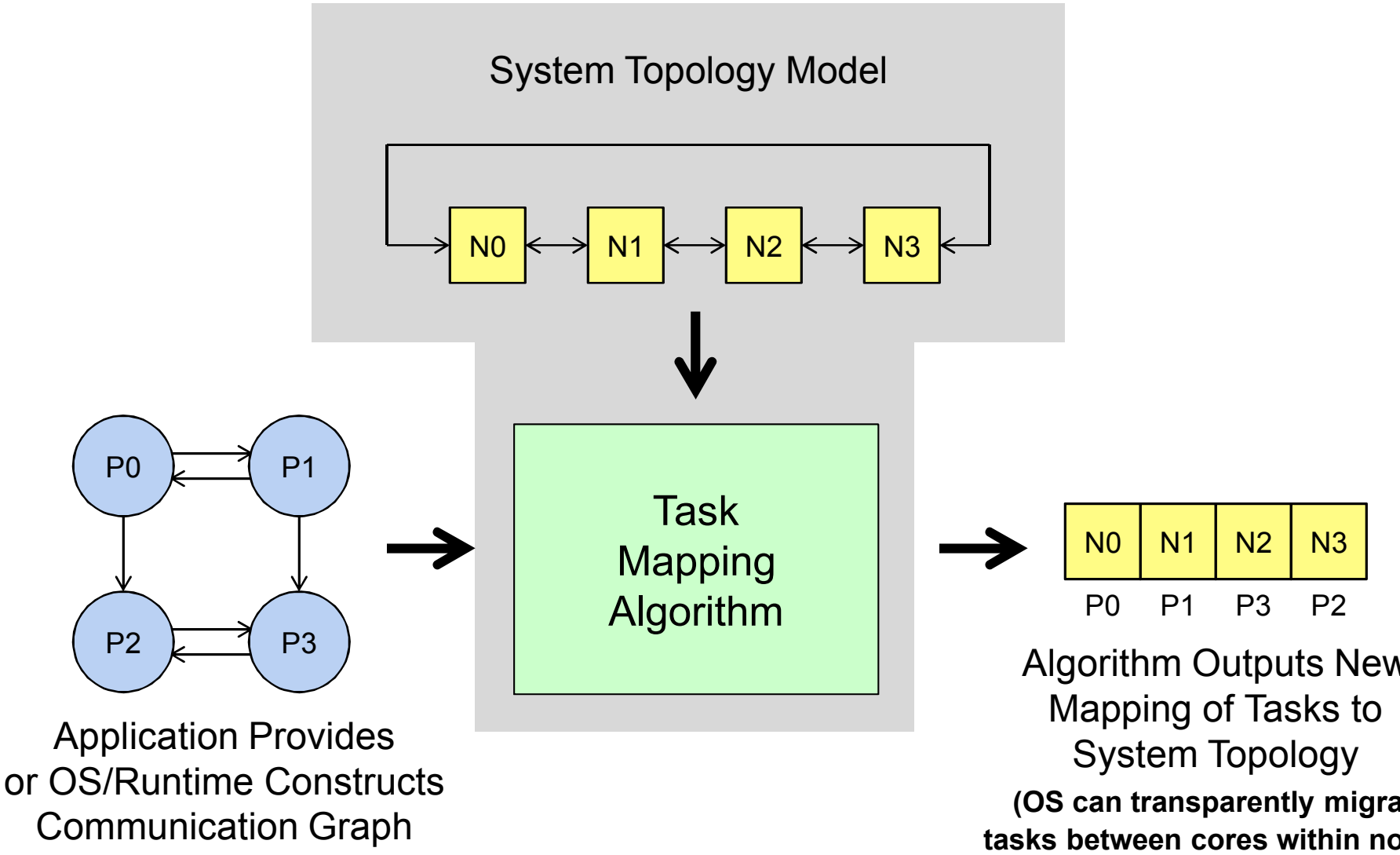


Data from Courtenay Vaughan and Richard Barrett

Interconnect is a 3-D torus.
Application talks to nearest 3-D neighbors.
Should be match made in heaven,
So what's going on?

- MiniGhost is a proxy application, represents CTH full application
- Explicit time-stepping, synchronous communication, 27-point stencil across 3-D grid
- Dark Red Curve:
  Original configuration scaled poorly after 16K cores (1024 nodes, 512 Geminis)
- Light Red Curve:
  Reorder MPI rank to node mapping to reduce off-node communication
  - Original: 1x1x16 ranks/node
  - Reorder: 2x2x4 ranks/node

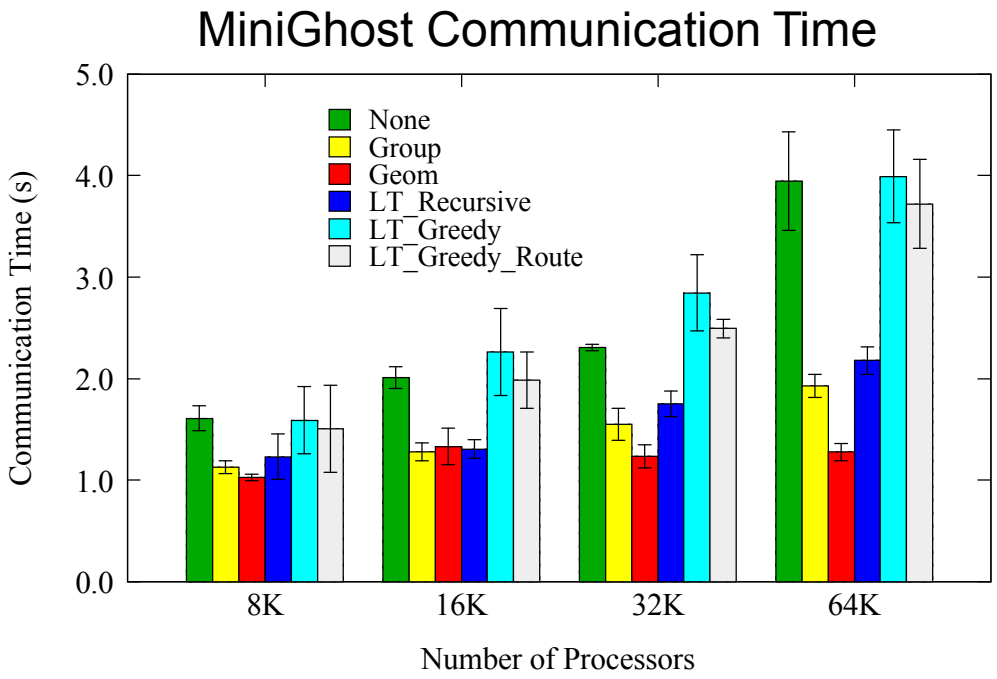
Task Mapping Example



MiniGhost Scalability is Improved by Task Mapping

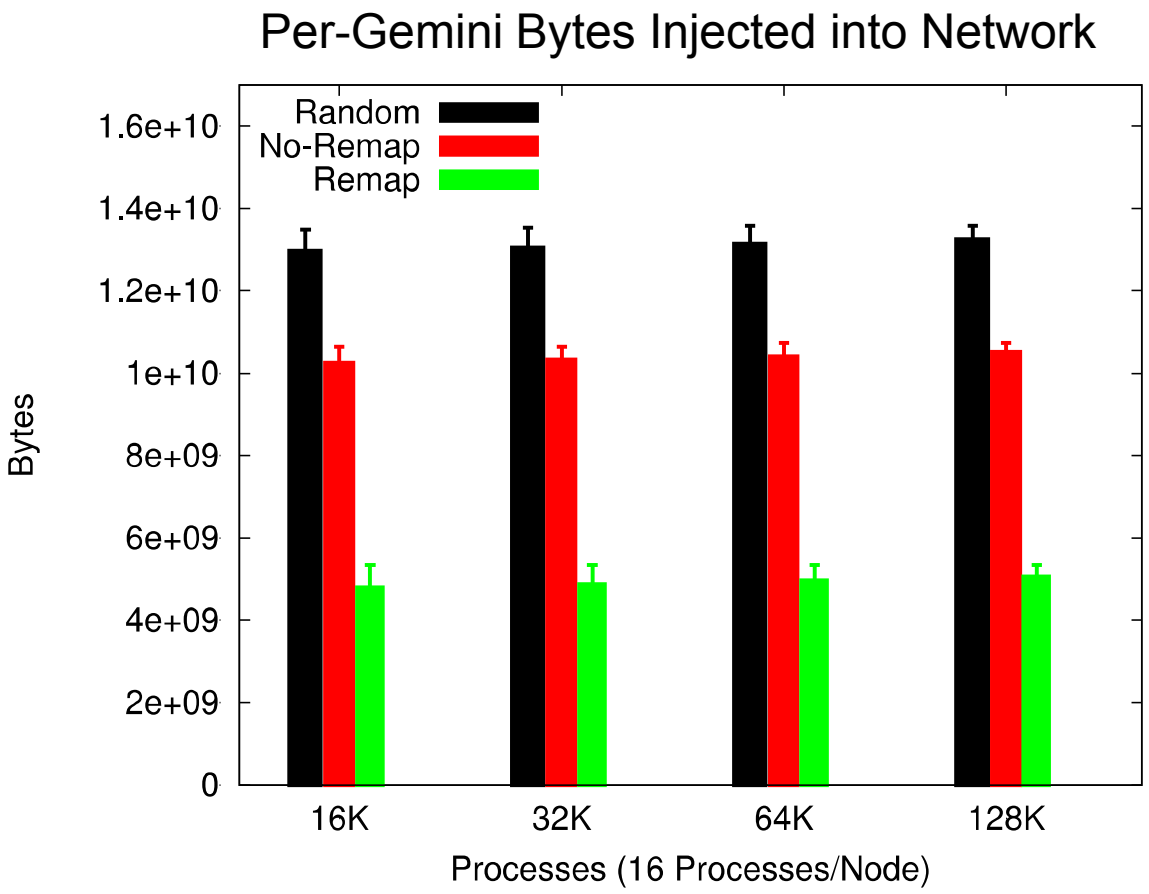
Sandia National Laboratories

- MiniGhost configuration
  - Bulk synchronous mode
  - 27-point stencil 3-D grid
  - Weak scaling mode
  - Runs on Cielo, MPI ppn=16
  - Avg. of 5 production runs, error bars are standard deviation
- Observations
  - Reordering for multicore important, still upticking (“Group”)
    - Minimize surface area by putting 2x2x4 subprob per node vs. 1x1x16
  - Leveraging geometric information pays off in this case
    - But, not all applications will have geometric information
  - Libtopomap’s recursive bisection strategy is its best in this case, similar to reordering for multicore (LT uses Parmetis internally to do multicore ordering)



From IPDPS’14 Paper, M. Deveci, et al.
FY12-14 LDRD (PI: Karen Devine)

Task Mapping Impacts Communication



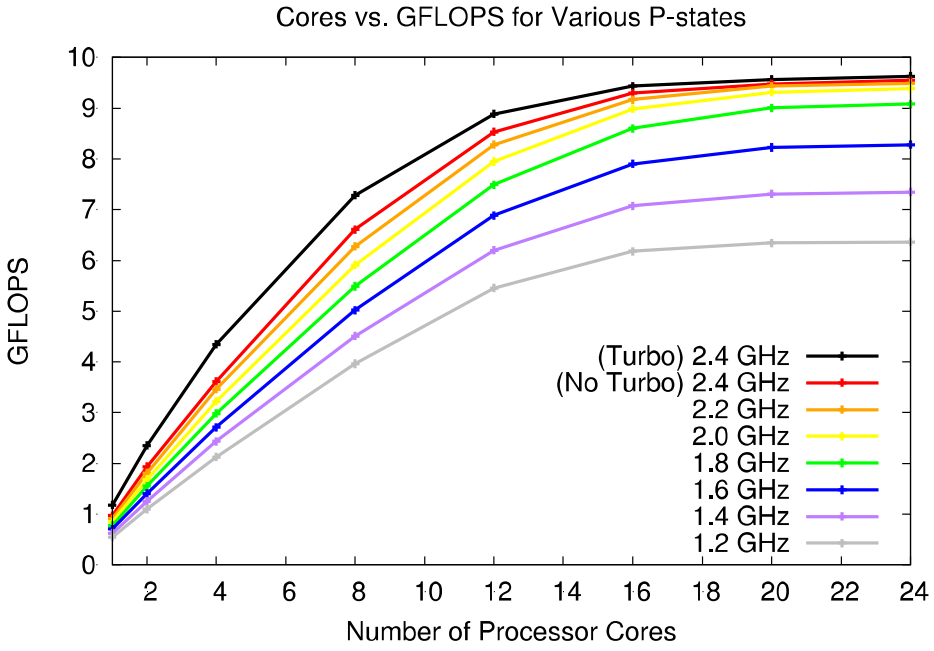
- Changing the mapping of MPI processes to nodes affects off-node communication
- Used Gemini tile counters to measure traffic injected on the host links
- The reordered “Remap” scheme (2x2x4) reduces off-node communication by more than a factor of 2x compared to the original “No-Remap” scheme (1x1x16)

Another Bad Task Mapping Example

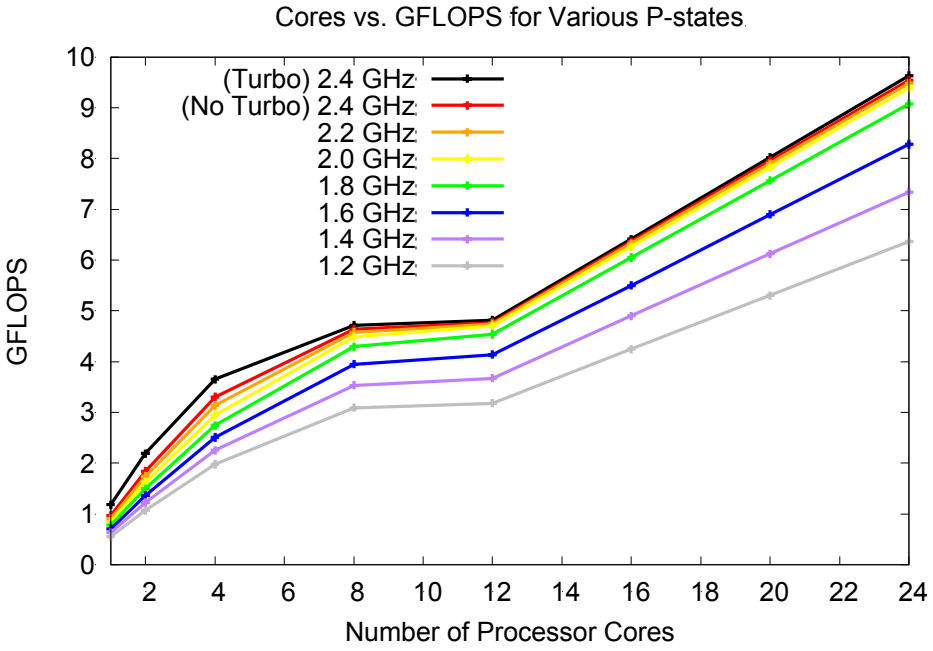
Sandia National Laboratories

Default on Volta (all Cray XC30's?) is to fill up a NUMA node before moving onto the next. Tasks that spill over to the second NUMA node complete quickly, then wait for the overloaded NUMA node to complete => bad load balance due to bad task mapping

Good Intra-Node Task Mapping



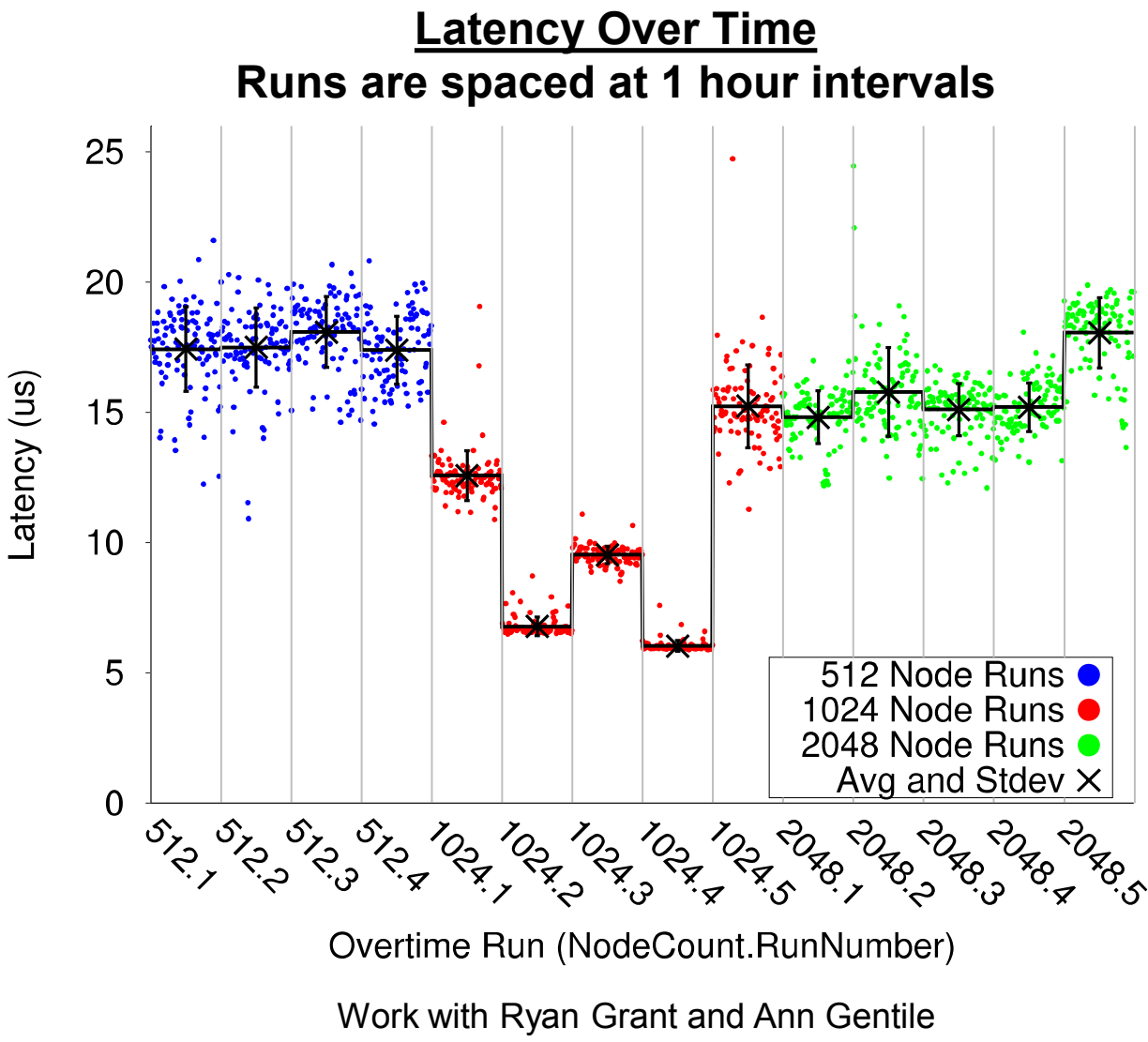
Bad Intra-Node Task Mapping



Overtime: A Benchmark for Analyzing Performance Variation due to Network Interference

Sandia National Laboratories

- Run-to-run network performance variation is a concern
  - Cray Gemini significant
  - Cray Aries likely some
- In some cases, can minimize by observing interference and dynamically adapting, calculate new task mapping
- Plot shows series of 14 Overtime benchmark runs on Blue Waters, spread apart at 1 hour intervals



Work with Ryan Grant and Ann Gentile

Task Mapping Takeaways

Sandia National Laboratories

- Task mapping has a significant impact on data movement
- Contention for shared resources can lead to performance variability
- Have been looking at task mapping in the context of inter-node performance optimization, getting experience and understanding
- In future, on-chip networks will have similar characteristics. OS and Runtime task mapping will likely be important
  - Today's operating systems (Linux) are not aware of the application communication graph, do basic load balancing of tasks across CPUs
  - Intra-node task mapping is easier than inter-node task mapping, likely a lot lower overhead

Outline

Sandia National Laboratories

- History of SNL Lightweight Kernel Operating Systems (LWK OS)
- Are LWKs still relevant?
- What we're working on
  - XPRESS project (FY13-FY15, DOE/ASCR X-Stack Program)
  - Hobbes project (FY14-FY16, DOE/ASCR OS/Runtime Program)
  - Power API and power management (CSSE, FY14 L2 Milestone)
  - Task mapping (LDRD + CSSE)
- Conclusion

Conclusion

Sandia National Laboratories

- Hobbes and XPRESS projects continuing SNL LWK R&D
  - XPRESS: Increase synergy of compute node OS and runtime systems
  - Hobbes: OS/R functionality needed to support application composition
  - Path of least resistance to impact, demonstrate benefit via large-scale testing, roll ideas into vendor's existing system software stack
- Power API provides portable interfaces for power management
- Operating systems and runtime systems becoming more dynamic and adaptive, must provide support for more diverse applications and workflows

Acknowledgements

- Ron Brightwell
- Jack Lange
- Jim Laros
- Ryan Grant
- XPRESS, Hobbes, PowerAPI, and RAAMP teams

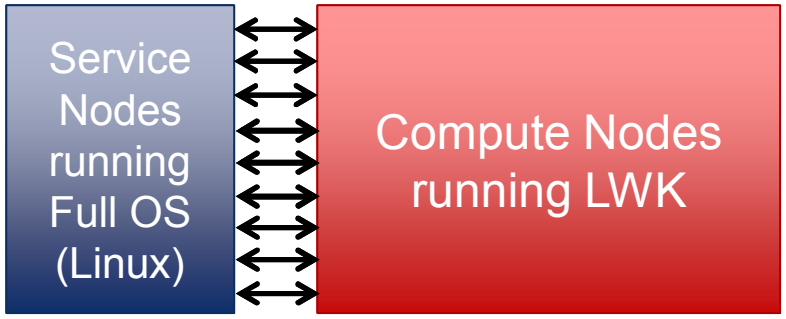
Sandia National Laboratories

Backup

What is a Lightweight Kernel? (LWK)

Sandia National Laboratories

- LWK is one component in the overall machine operating system
  - Relies on full-service OS/Linux functionality elsewhere in system
  - Minimizes OS and runtime overhead, exposes full HW capability



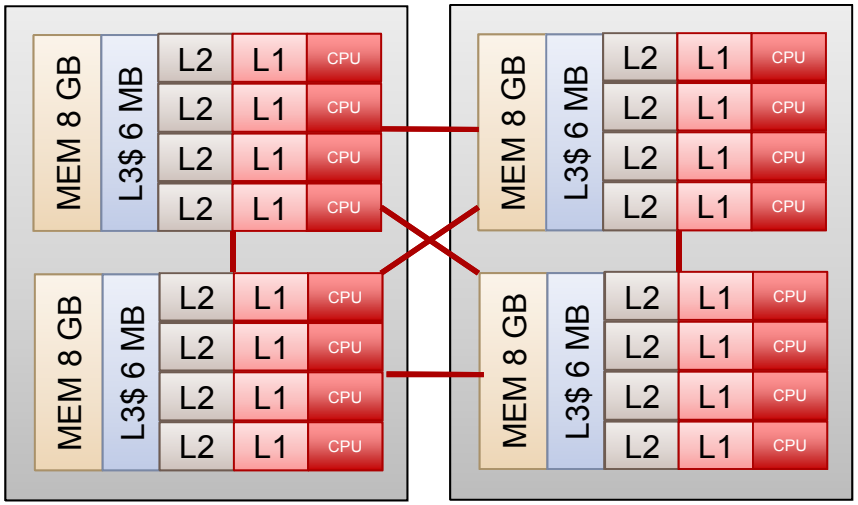
Distributed Memory MIMD Message Passing Programming Model

- Sometimes called a "Compute Node OS"
  - A bit of a misnomer, more like an application runtime
  - Derives from partition model: specialize HW/SW for compute nodes, service nodes, login nodes, I/O nodes, etc.
- To a first order, goal is to deliver maximum hardware performance to scalable HPC applications
  - Trade functionality for performance
  - Extends beyond MPI (threads, OpenMP, SHMEM, UPC, …)

Node Memory Hierarchy is Evolving

Sandia National Laboratories

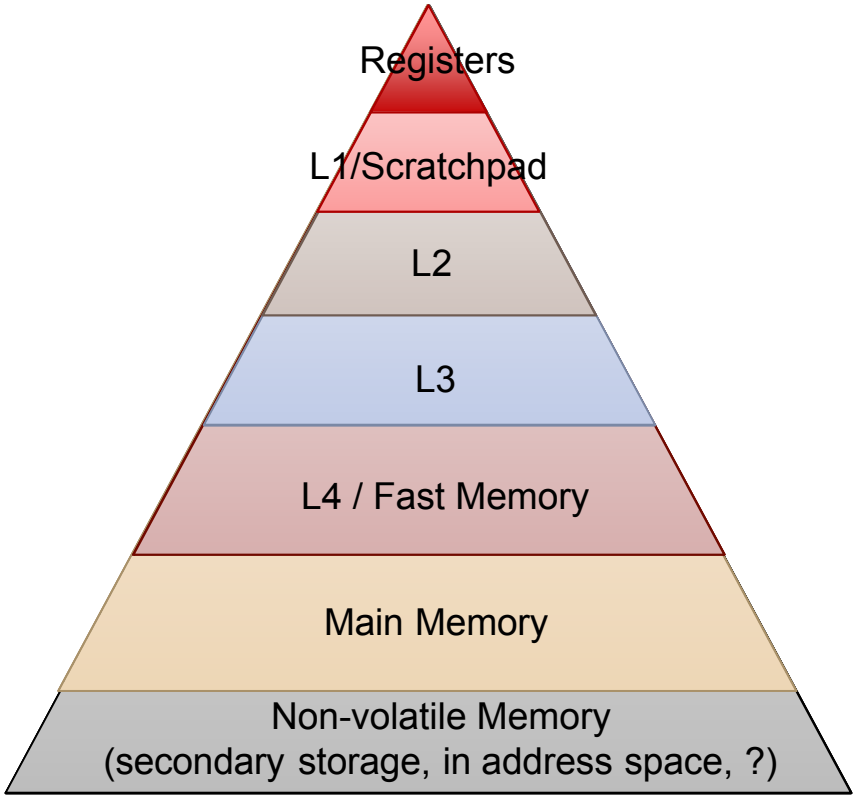
Cielo
2010
AMD "MagnyCours"



NUMA
(even within socket)

Bytes/FLOPs:
0.21

Exascale (?)
2022 (?)
CPU / GPU / APU (?)

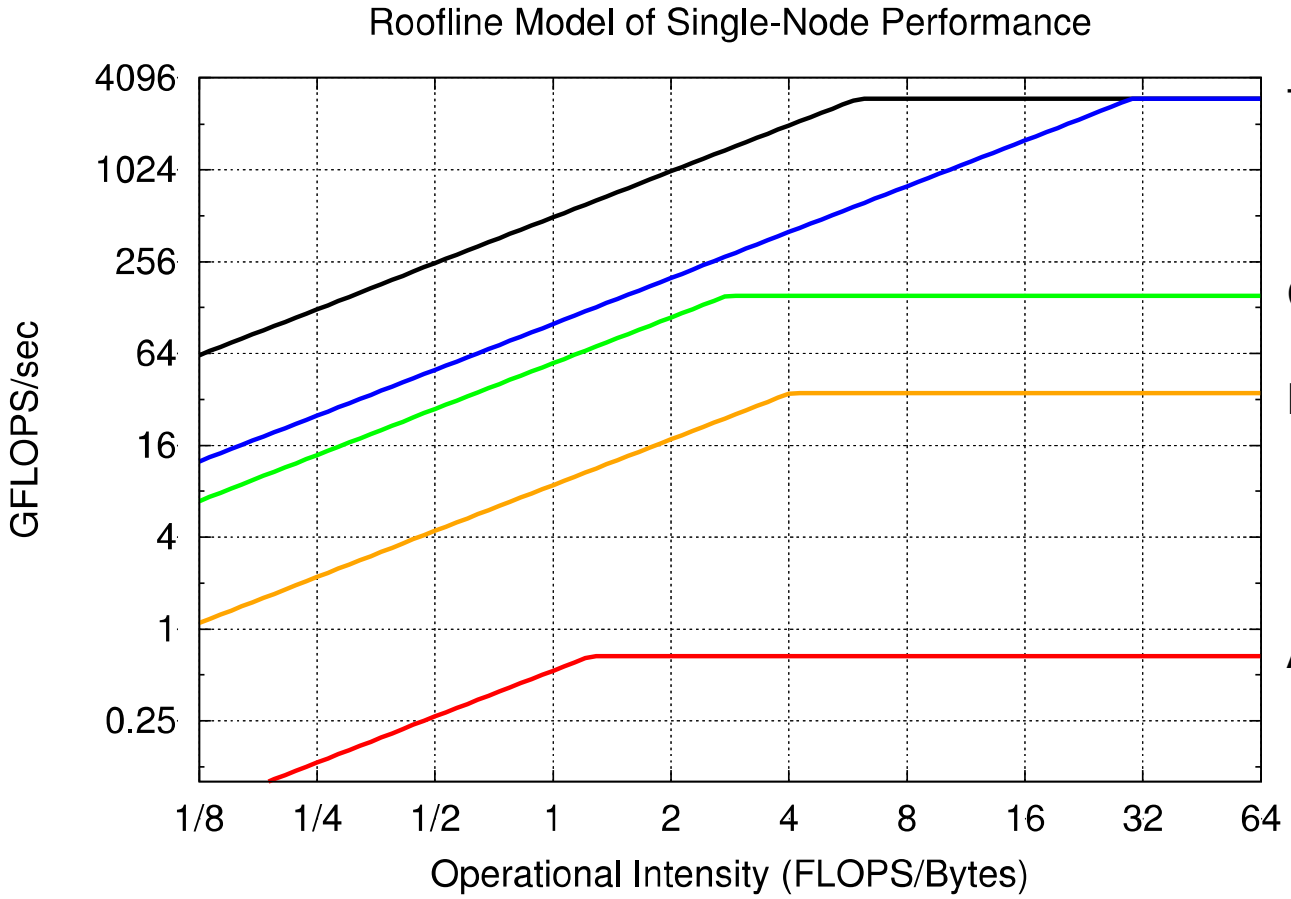


.02 (?)

Node-level Programming is Getting (a lot) Harder

Sandia National Laboratories

Must deal with Task-level Parallelism (TLP), Single-Instruction-Multiple-Data (SIMD | SIMT), Core-level Parallelism (CLP), Architecture-level Parallelism (ALP), Network-level Parallelism (NLP), Memory-level Parallelism (MLP), Cache topology, Non-uniform memory access (NUMA), N-level memory, …



Trinity KNL/Phi (2016)
- 50+cores, 4 hw_threads/core
- 16 FLOPS/clock/core
- 2-level HMC+DRAM memory

Cielo (2011)
- 16-cores, 4 FLOPS/clock/core, NUMA

Red Storm Quad Core (2008)
- 4-cores
- 4 FLOPS/clock/core
- Uniform memory access

ASCI Red / TFLOPS (1997)
- Single core (2 sockets)
- 1 FLOP/clock/core
- Uniform memory access

DDOT WAXPY SpMV 3DStencil 3DFFT MMUL →

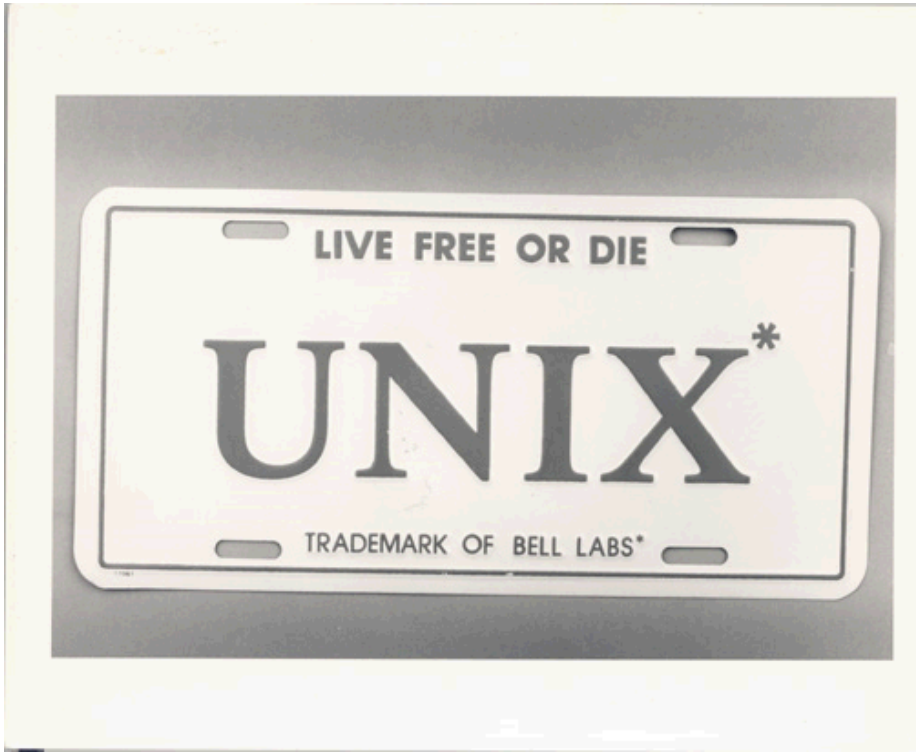
LWK Timeline Summary

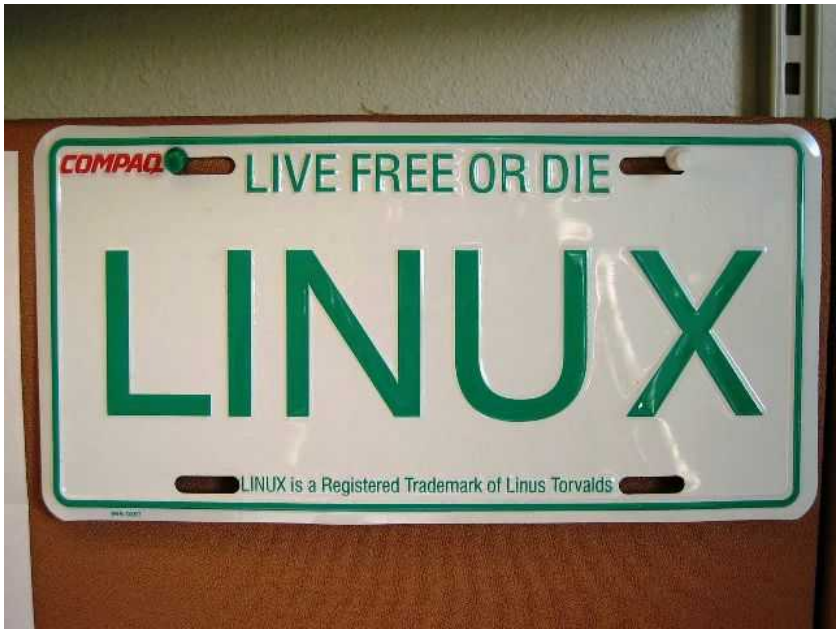
Sandia National Laboratories

- 1991 – Linux 0.02
- 1993 – **SUNMOS LWK** for Paragon (Sandia + UNM + OS)
  - Replaced Intel's really bad OSF/1 (used all node's mem, 17% net perf)
  - SUNMOS 250 KB/node memory, 85% of network's peak performance
- 1994 – Linux 1.0
- 1997 – ASCI Red, **Puma/Cougar LWK**, Portals 2.0
- 1999 – **Cplant**, Linux-based OS, Portals 3.0
- 2004 – Red Storm, **Catamount LWK**, Portals 3.3
- 2004 – IBM develops CNK LWK, modeled on Catamount
- 2007 – Cray ships Compute Node Linux (CNL / CLE)
- 2008 – **Kitten LWK**, explore virtualization, LDRD FY08-10
- 2013 – Hobbes **Node Virtualization Layer (NVL)**
  - NVL derived from Kitten LWK and Palacios Hypervisor

Four+ Decades of UNIX
23 Years of Linux

Sandia National Laboratories





Operating System = Collection of software and APIs
Users care about environment, not implementation details
LWK is about getting details right for scalability

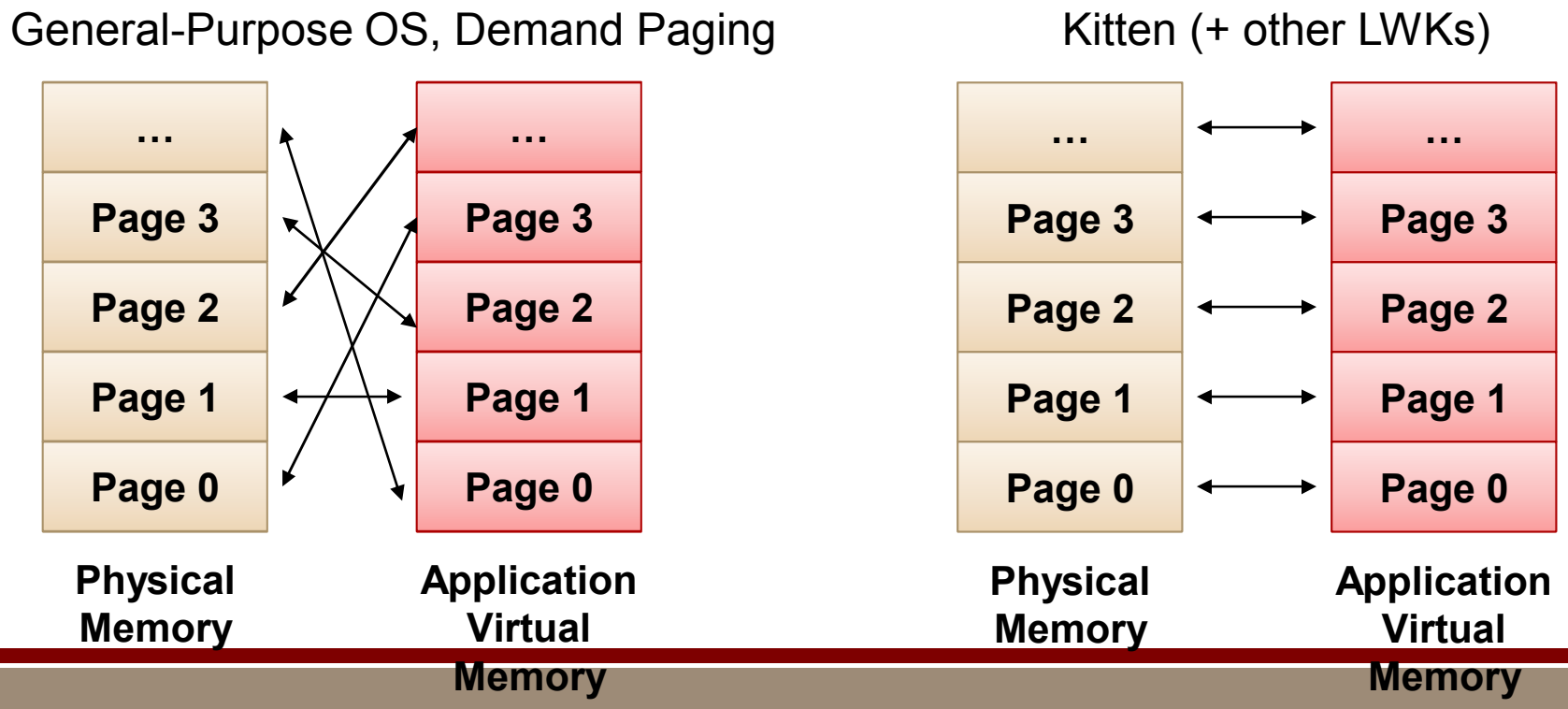
Kitten LWK Implementation

Sandia National Laboratories

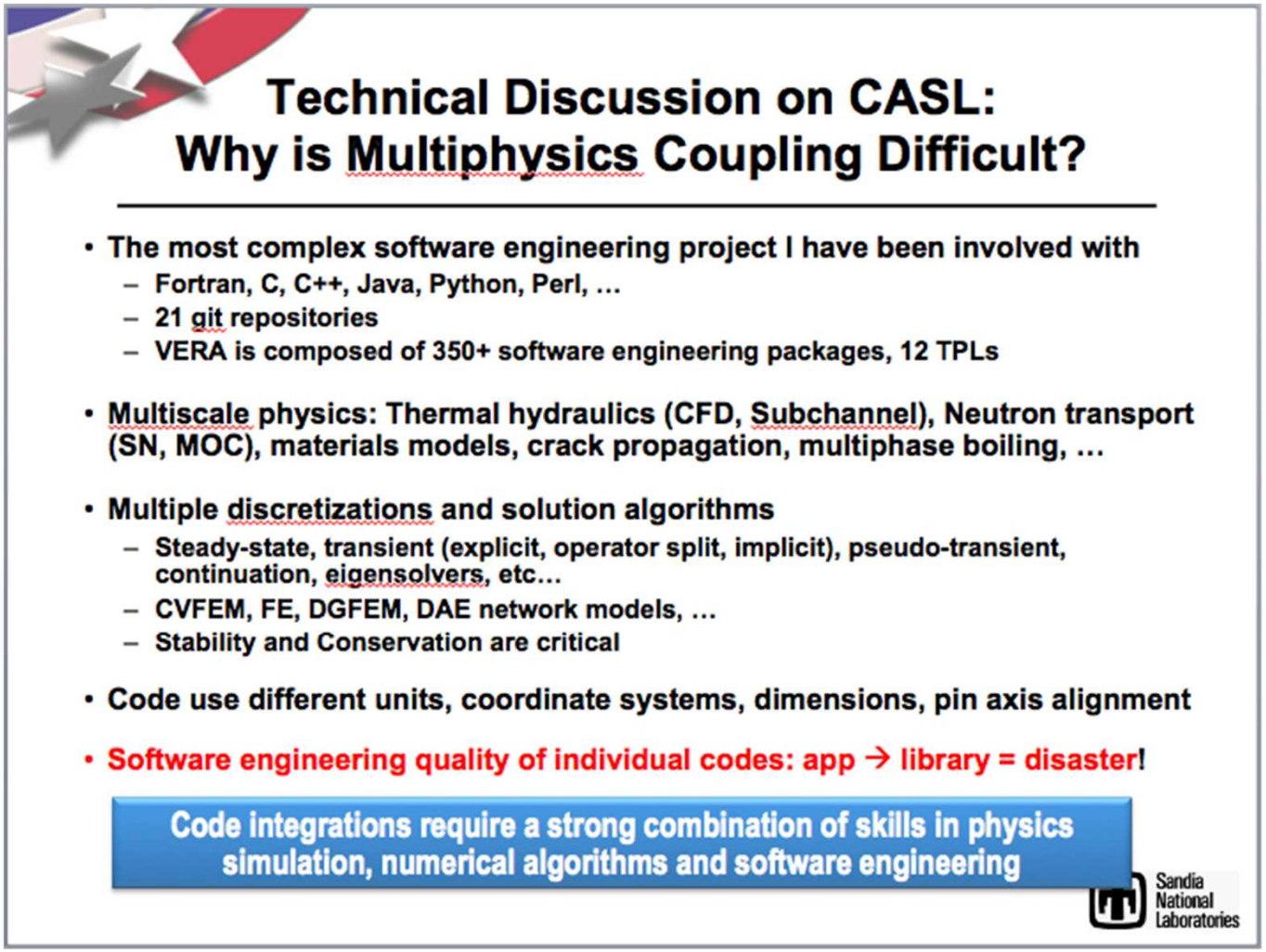
- Monolithic, C code, GNU toolchain, Kbuild configuration
  - Core Kernel 12K SLOC, x86_arch 12K SLOC, include 22K SLOC
- Supports x86-64 architecture, porting to ARM
  - Boots on standard PC architecture, Cray XE, and in virtual machines
  - Boots identically to Linux (Kitten bzImage and init_task)
- Repurposes basic functionality from Linux
  - Hardware bootstrap
  - Basic OS kernel primitives (lists, locks, wait queues, etc.)
  - Directory layout similar to Linux, arch dependent/independent dirs
- Custom address space management and task management interfaces
  - User-level API for managing physical memory, building virtual address spaces
  - User-level API for creating tasks, which run in virtual address spaces

Memory Management

- Simple, static virtual to physical mapping
  - Eliminates non-determinism
  - Enables straightforward use of large page sizes
  - Enables optimization in network stack
  - Physical memory managed by user-level process



Multiphysics Example



Multiphysics Example (cont'd)

Sandia National Laboratories

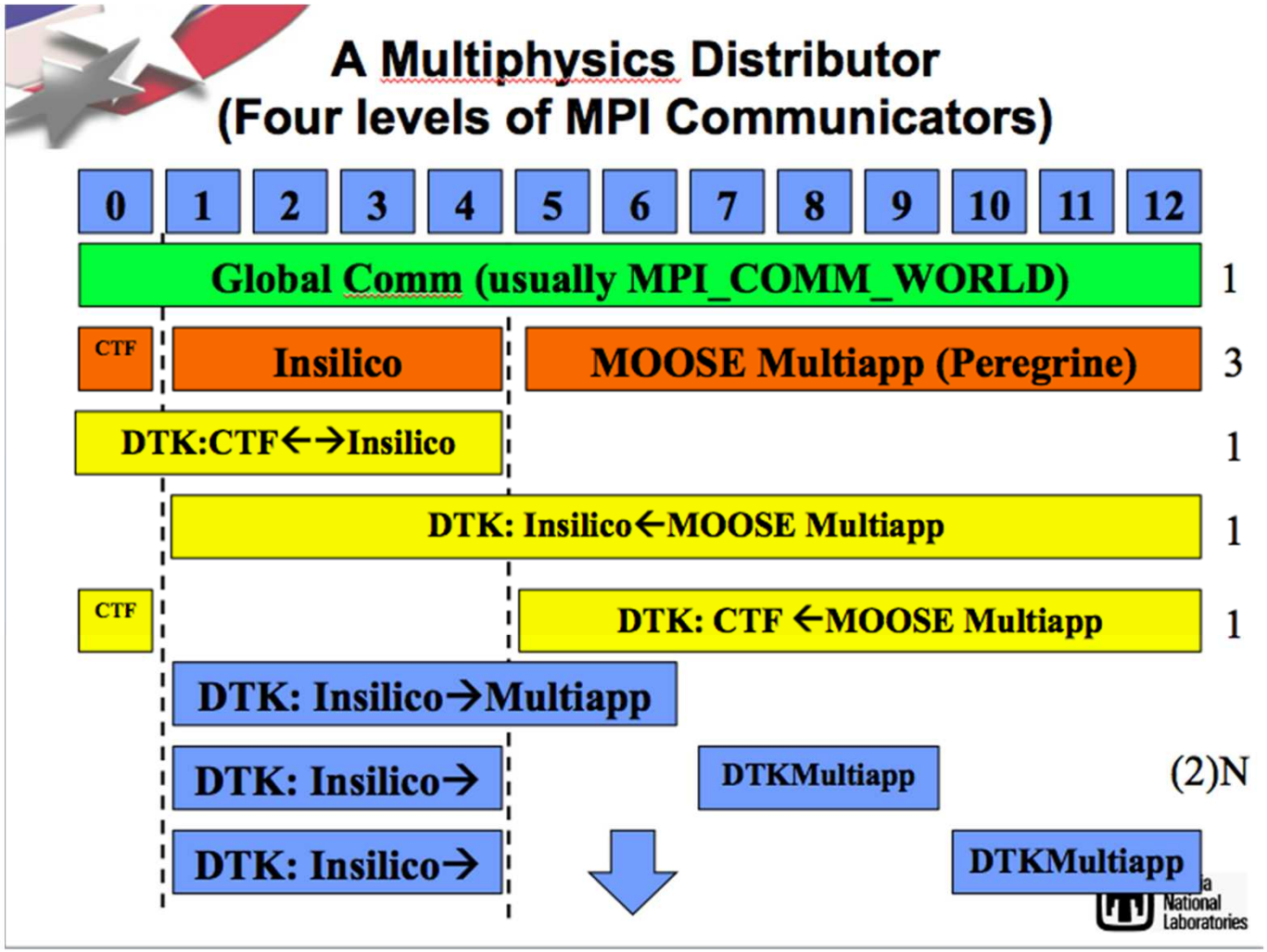
Peregrine/Insilico/CTF Executable (Only ONE of many executables in VERA)

- VRIPSS
- COBRA-TF
- Exnihilio (Insilico, Denovo, nemesis)
- Drekar
- MOOSE/Peregrine
- Qt
- SCALE (200+ libraries, 30+ years of NRC codes)
- LIBMESH
- Data Transfer Kit
- LIME
- Trilinos (35+ libraries)
- PETSc
- HYPRE
- Netcdf
- HDF5
- Boost
- Many others…

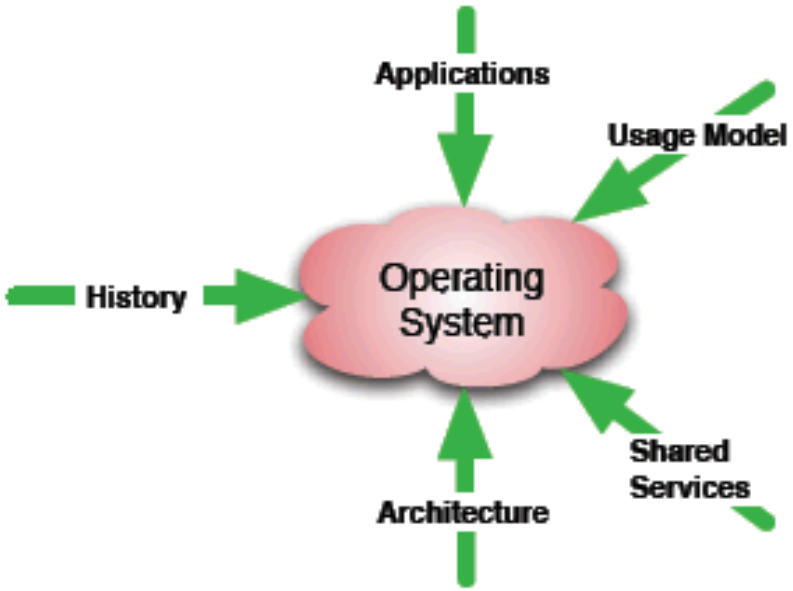
We are pulling in almost every general HPC library under one executable and dealing with massive collisions!

Sandia National Laboratories

Multiphysics Example (concl'd)



OS Influences

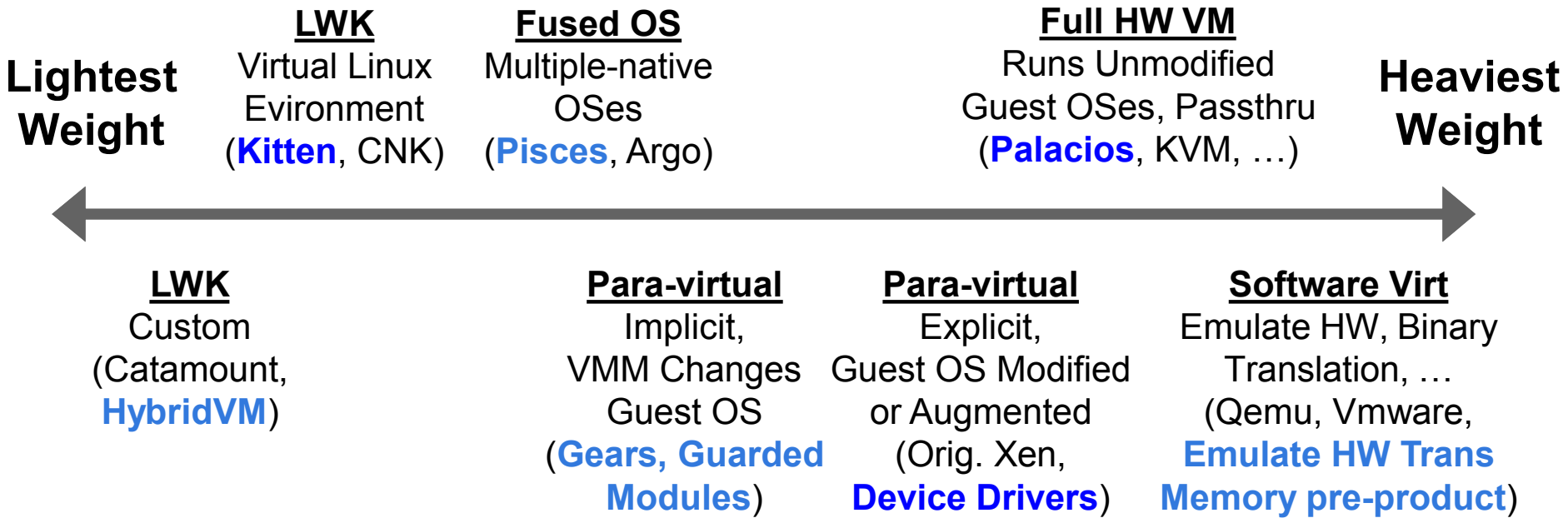


- Lightweight OS
  - Small collection of apps
    - Single programming model
  - Single architecture
  - Single usage model
  - Small set of shared services
  - No history
- Puma/Cougar/Catamount
  - MPI
  - Distributed memory
  - Space-shared
  - Parallel file system
  - Batch scheduler

Hobbes Exploring Spectrum of Virtualization

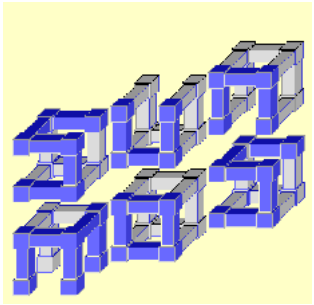
Sandia National Laboratories

- Virtualization doesn't have to be "big and heavy"
  - Don't have to trap everything
  - VMM can setup paths to hardware, then get out of way
- There are multiple virtualization architectures, not just one
  - Hobbes NVL team working across spectrum (Blue items, research in Light Blue)



LWK Timeline – SUNMOS (1993)

Sandia National Laboratories



- SUNMOS on Intel Paragon (Sandia + UNM + OS)
- Intel supplied OSF/1, distributed OS, really bad
  - Used over 50% of compute node's memory (8-12 MB of 16 MB)
  - Limited network bandwidth to 35 MB/s, peak was ~200 MB/s
- => SUNMOS created out of necessity, huge success
  - 250 KB memory footprint
  - 170 MB/s network bandwidth
- "Field guide to SUNMOS"
  http://pages.swcp.com/~mccurley/humor/sunmos_humor.html

Intel Paragon (1993)

Peak:	140 GFLOPS
Compute Nodes:	1840 (3680 Intel i860 CPUs)
Total Memory:	40 GB
Network Link:	200 MB/s
Power:	Unknown
System SW:	SUNMOS LWK / Portals

LWK Timeline – Cougar (1997)

Sandia National Laboratories

Intel ASCI Red / TFLOPS (1997)

Peak:	1800 GFLOPS (1999 -> 3150 GFLOPS)
Compute Nodes:	4536 (9072 Intel Pentium Pro CPUs)
Total Memory:	1212 GB
Network Link:	400 MB/s, 15 us Latency
Power:	800 KW
System SW:	Cougar LWK, Portals 2.0

LWK Timeline – Cplant Linux (2001)

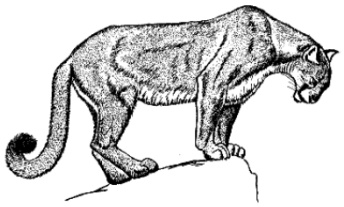
Sandia National Laboratories

Cplant / Ross (2001)

Peak:	1782 GFLOPS (Linpack 996 GFLOPS)
Compute Nodes:	1800 (DEC Alpha 21264 EV6)
Total Memory:	448 GB
Network Link:	100 MB/s, 60 us
Power:	Unknown
Compute OS:	Linux Kernel, Portals 3.0

LWK Timeline – Catamount (2004)

Sandia National Laboratories



Red Storm (2004 - 2012)

Peak:	42 TFLOPS (127 TFLOPS final)
Compute Nodes:	10368 (12960 final)
Total Memory:	31 TB (77 TB final)
Network Link:	2 GB/s, 5 us
Power:	1.7 MW (2.5 MW final)
Compute OS:	Catamount LWK, Portals 3.3