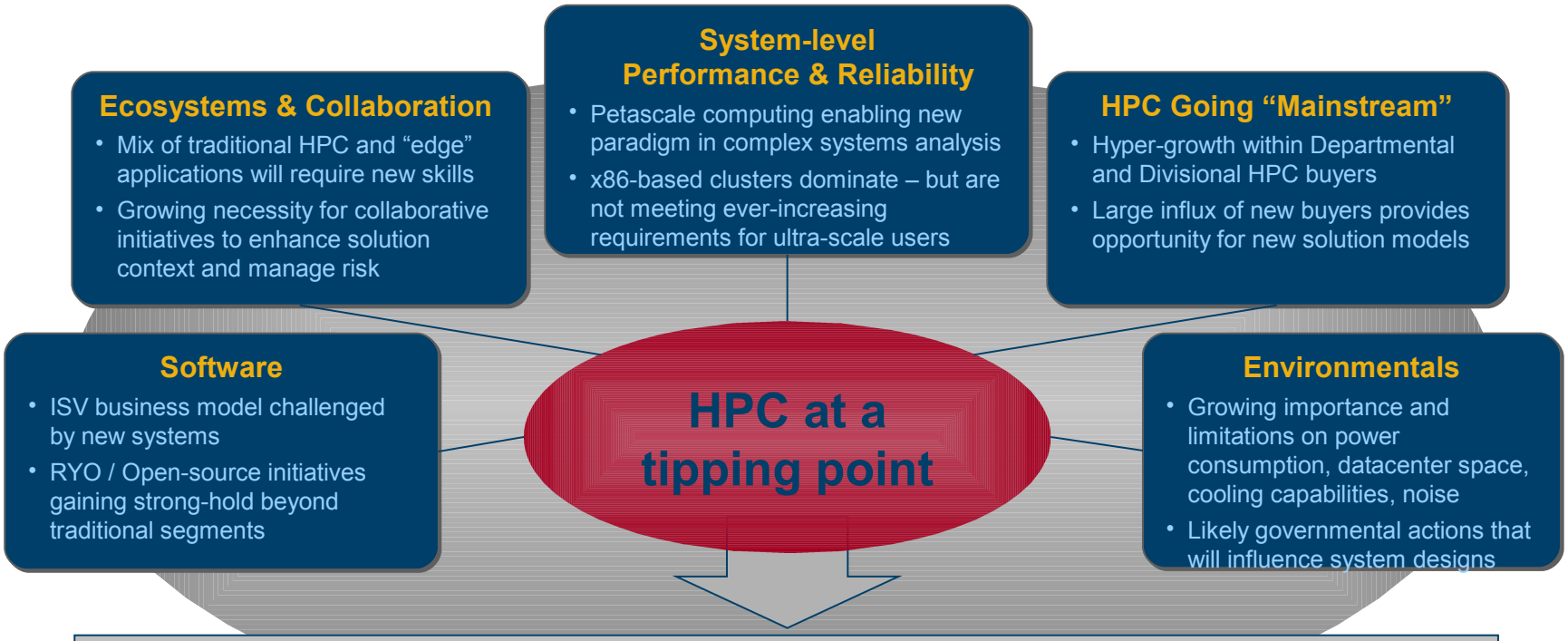
HPC and Accelerators

Ken Rozendal
Chief Architect, IBM Linux Technology Cener

November, 2008

THE LINUX FOUNDATION

A mix of market forces, technological innovations and business conditions have created a tipping point in HPC



How can we exploit these shifts to meet our Performance & Productivity requirements?

Market Dynamics and Design Principles; Petascale and Beyond

- Existing and emerging workloads with diverse and highly specialized requirements
- Increased societal awareness and regulatory mandate on environmentals/carbon footprint (power consumption, noise, space efficiency, resource lifecycle)
- Open, adaptive software ecosystem that supports legacy applications and hardware architectures while enabling new, advanced capabilities
- Robust, modular hardware architecture framework that enables new and existing processor types to be applied for workloads acceleration
- Reliability and Scalability for 10^{15} and beyond....

THE LINUX FOUNDATION

Technology Trends and Challenges

There are 3 major challenges on the road to and beyond Petascale computing:

- Supplying and paying for the power required to run the machines which can be equivalent to a small town!
- Reliability, Availability, and Serviceability because of the law of large numbers and Soft Errors
- Finding efficient ways to program machines that will support MILLIONS of simultaneous processes!

Performance and Productivity Challenges require a Multi-Dimensional Approach

Highly Productive Systems

POWER

HPCS PERCS

Highly Scalable Multi-core Systems

BLUE GENE

Hybrid Systems

Comprehensive (Holistic) System Innovation & Optimization

IBM Ultra Scale Approaches

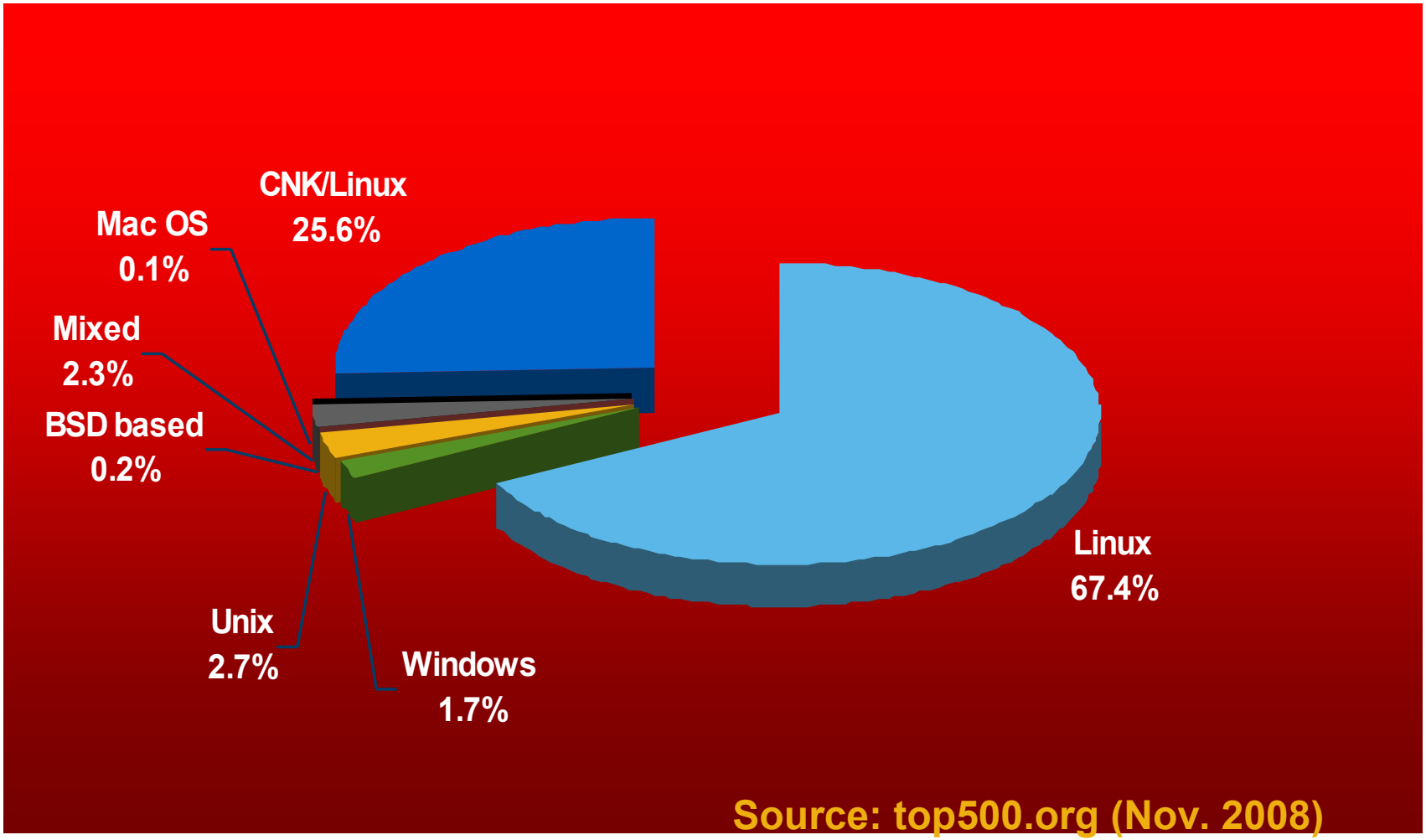
- The right architecture for Scaling HPC applications depends on the application and the pinch points of both the application and the organization using it

- Blue Gene – Maximize Flops Per Watt
  with Homogeneous Cores
  by reducing Single Thread Performance

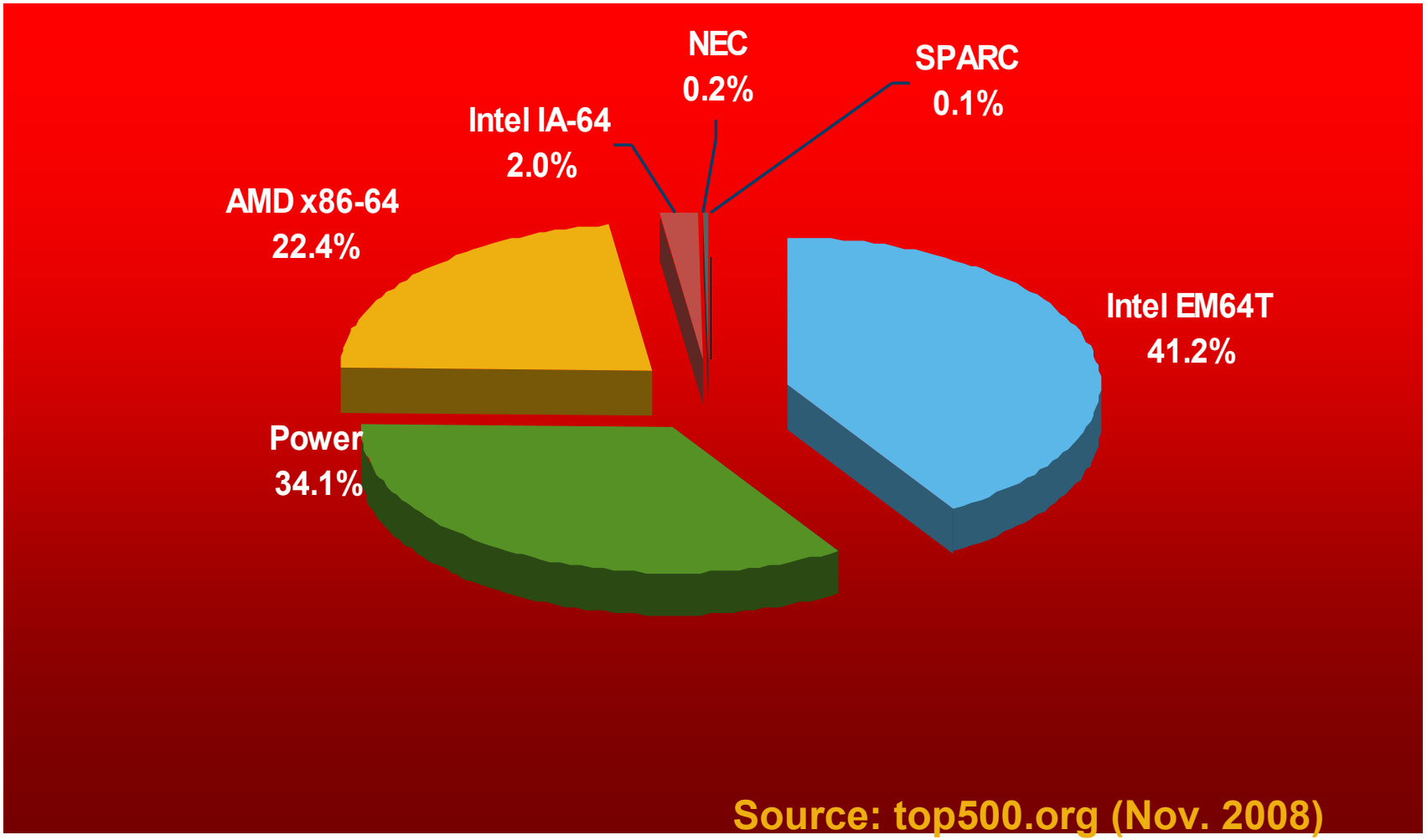
- Power/PERCS – Maximize Productivity and Single Thread Performance with Homogeneous Cores

- Roadrunner – Use Heterogeneous Cores and
  an Accelerator Software Model
  to Maximize Flops Per Watt
  and keep High Single Thread Performance

HPC Top 500 OS Breakdown (by Processor)



HPC Top 500 Processor Architecture Breakdown



Advantages of Accelerators

- **Substantially more energy efficient than CPUs**
  - higher percentage of transistors dedicated to floating point
- **Substantially less space used than for general purpose systems**
  - substantially more FLOPs per socket
- **Substantially less cooling required**
  - More energy efficient generates less heat for the same computation.
- **More reliable**
  - Less overall system components for the same computation.

HPC Green 500 (green500.org)

Rank	Top 500	Site	Manufacturer	Computer
1	220	Interdisciplinary Centre for Mathematical and Computational Modelling, University of Warsaw	IBM	BladeCenter QS22 Cluster
2	429	Repsol YPF	IBM	BladeCenter QS22 Cluster
2	430	Repsol YPF	IBM	BladeCenter QS22 Cluster
2	431	Repsol YPF	IBM	BladeCenter QS22 Cluster
5	41	DOE/NNSA/LANL	IBM	BladeCenter QS22/LS21 Cluster
5	42	IBM Poughkeepsie Benchmarking Center	IBM	BladeCenter QS22/LS21 Cluster
7	1	DOE/NNSA/LANL	IBM	BladeCenter QS22/LS21 Cluster
8	75	ASTRON/University Groningen	IBM	Blue Gene/P Solution
9	56	IBM - Rochester	IBM	Blue Gene/P Solution
9	57	RZG/Max-Planck-Gesellschaft MPI/IPP	IBM	Blue Gene/P Solution
9	127	Centre for High Performance Computing	IBM	Blue Gene/P Solution
9	128	Moscow State University	IBM	Blue Gene/P Solution
9	129	Oak Ridge National Laboratory	IBM	Blue Gene/P Solution
9	130	Stony Brook/BNL, New York Center for Computational Sciences	IBM	Blue Gene/P Solution
15	24	EDF R&D	IBM	Blue Gene/P Solution
16	5	Argonne National Laboratory	IBM	Blue Gene/P Solution
17	11	Forschungszentrum Juelich (FZJ)	IBM	Blue Gene/P Solution
17	16	IDRIS	IBM	Blue Gene/P Solution
19	59	HPC2N - Umea University	IBM	BladeCenter HS21 Cluster
20	496	Universiteit Gent	IBM	BladeCenter HS21 Cluster

Types of Accelerators – Attachment Mechanism

- **In core:**
  - part of instruction set, synchronous
  - examples: SSE, Altivec
- **On chip:**
  - usable by all cores on chip
  - access mediated by HW or SW
  - synchronous or asynchronous
  - example: cell SPE
- **Off chip, closely attached:**
  - attached by system bus or special bus
  - access mediated by HW or SW
  - synchronous or asynchronous
  - example: x87 chip
- **PCI-E attach:**
  - I/O model
  - synchronous or asynchronous
  - example: GPUs
- **Network attach:**
  - IP or RDMA model
  - generally asynchronous
  - example: cell in tri-blade

Types of Accelerators – Programming Mechanism

- **Part of instruction set:**
  - in line code, synchronous
  - examples: x87, SSE, Altivec, Blue Gene
- **Job submission:**
  - asynchronous
  - set up descriptor with parameters
  - enqueue for accelerator
  - examples: early GPUs
- **IB or Network attached appliance:**
  - asynchronous
  - send network packets with data, code, and/or operation requests
  - examples: cell in tri-blade
- **Simple programmability:**
  - load code into accelerator
  - usually very restricted set of instructions
  - examples: more modern GPUs
- **Full programmability:**
  - load code into accelerator
  - full functionality (though less rich than CPUs)
  - examples: cell SPE

Types of Accelerators – Capabilities

- **Single vs double precision floating point**
  - accelerators targeted at graphics tend to be single precision floating point
  - HPC tends to use double precision floating point
- **Scalar vs vector operation**
  - vector or matrix operation – SIMD
  - scalar operation – MIMD
- **Single core vs multicore**
  - multiple threads of execution running concurrently
- **Degree of parallelism**
  - number of floating point operations initiated simultaneously
- **Degree of pipelining**
  - number of floating point operations in different stages of execution simultaneously
- **Physical memory model**
  - local memory for accelerator use
    - mechanisms for moving data to/from local memory
  - ability of accelerator to access global memory

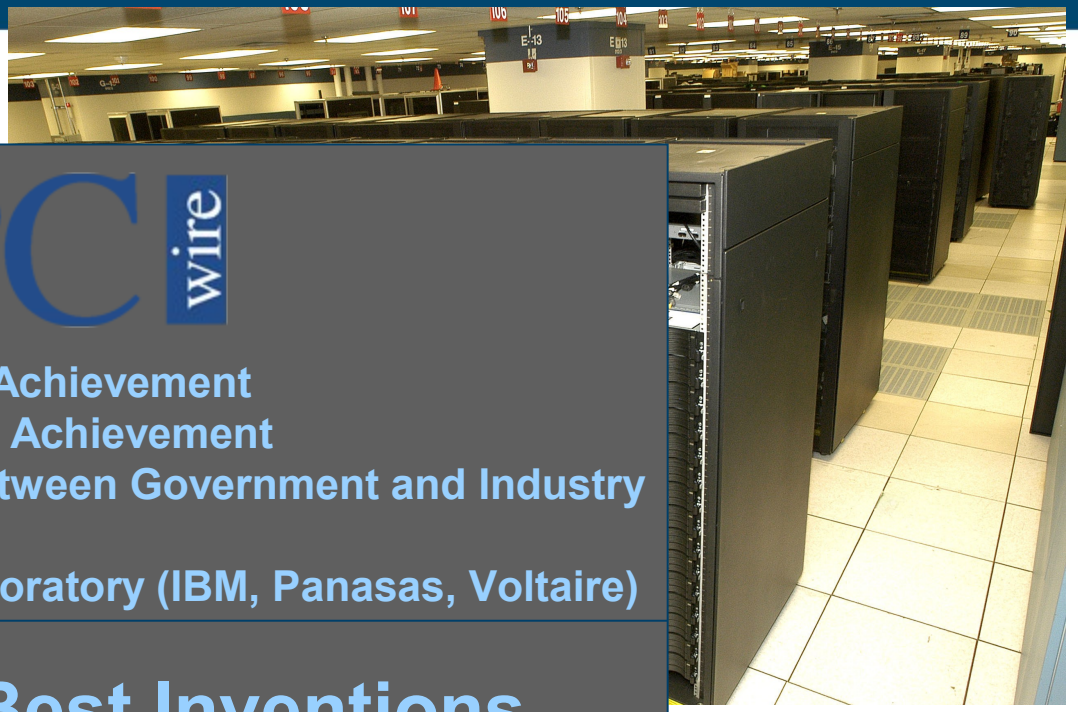
Other Important System Issues for Performance

- Overall memory bandwidth
- Memory latency (and NUMA issues)
- Amount of physical memory per computational thread of execution
- Hardware threads vs cores
- Caches – latencies, amounts, line sizes, degree of hierarchy
- SMP – number of cores in cluster node

Programming Models
User Level -> Runtime

June 18, 2008 – Petascale Delivered!

Roadrunner Breaks Petaflop Milestone



HPC wire

Editors' Choice Top Supercomputing Achievement
Readers' Choice Top Supercomputing Achievement
Readers' Choice Top Collaboration between Government and Industry

Roadrunner, Los Alamos National Laboratory (IBM, Panasas, Voltaire)

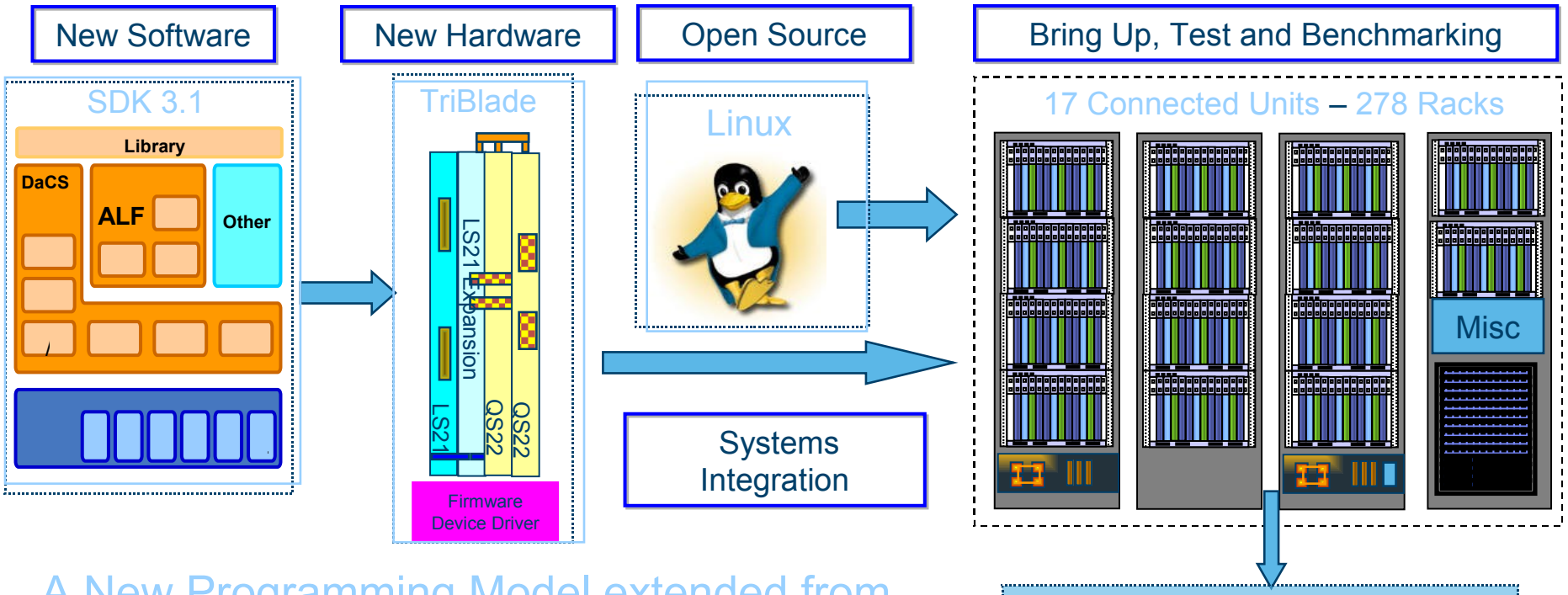
TIME's Best Inventions of 2008

10. The World's Fastest Computer

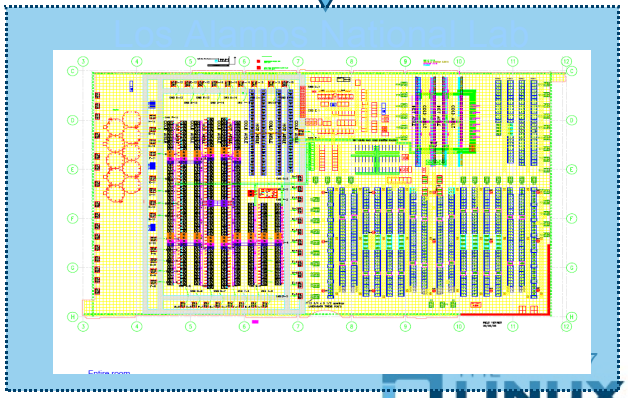
Groundbreaking system will have a profound impact on science, business and society

THE LINUX FOUNDATION

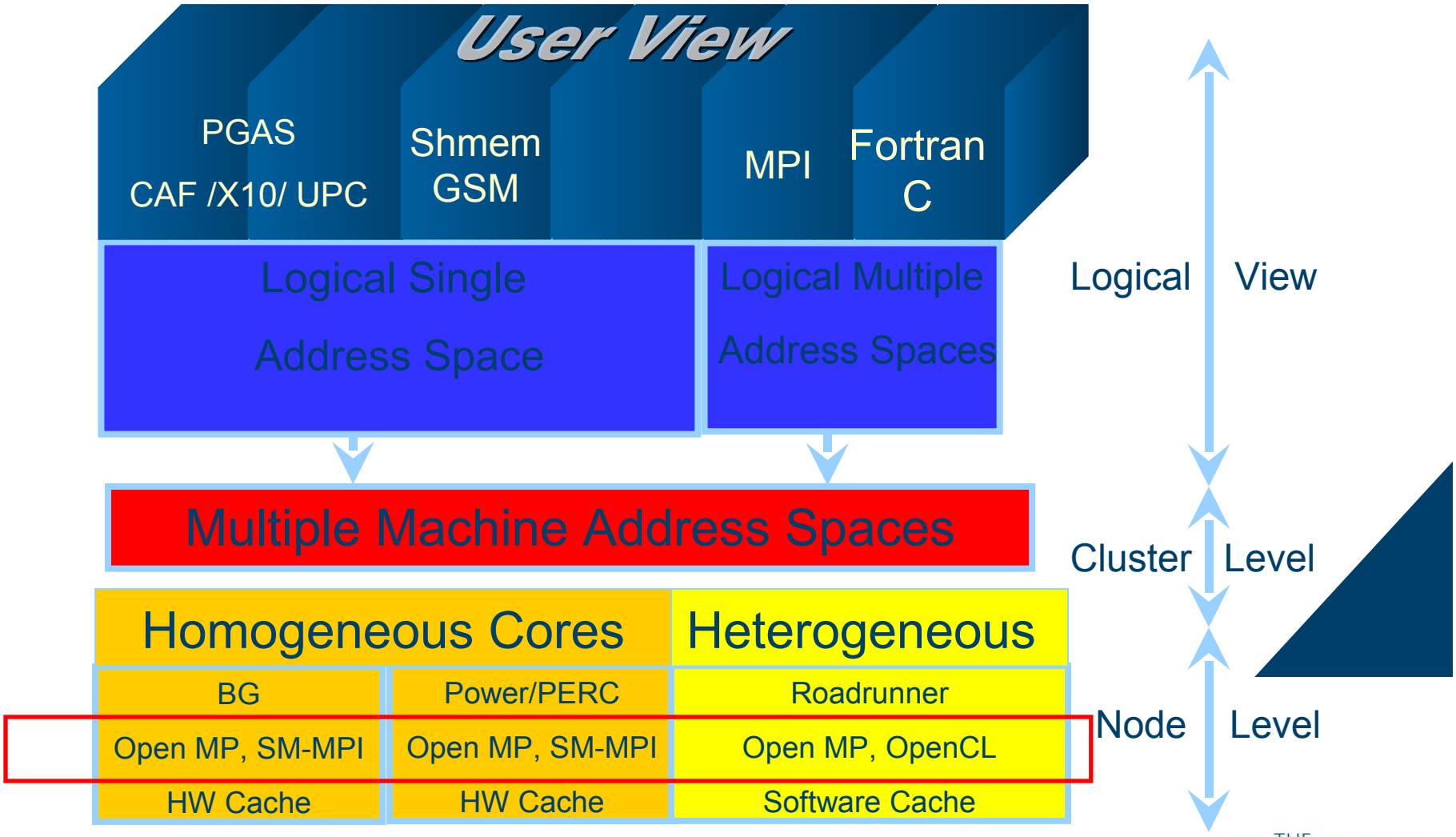
The making of Roadrunner – Key Building Blocks



- A New Programming Model extended from standard, cluster computing (*Innovation derived through close collaboration with LANL*)
- Hybrid and Heterogeneous Hardware
- Built around BladeCenter and Industry IB-DDR



Programming Models: Architecture

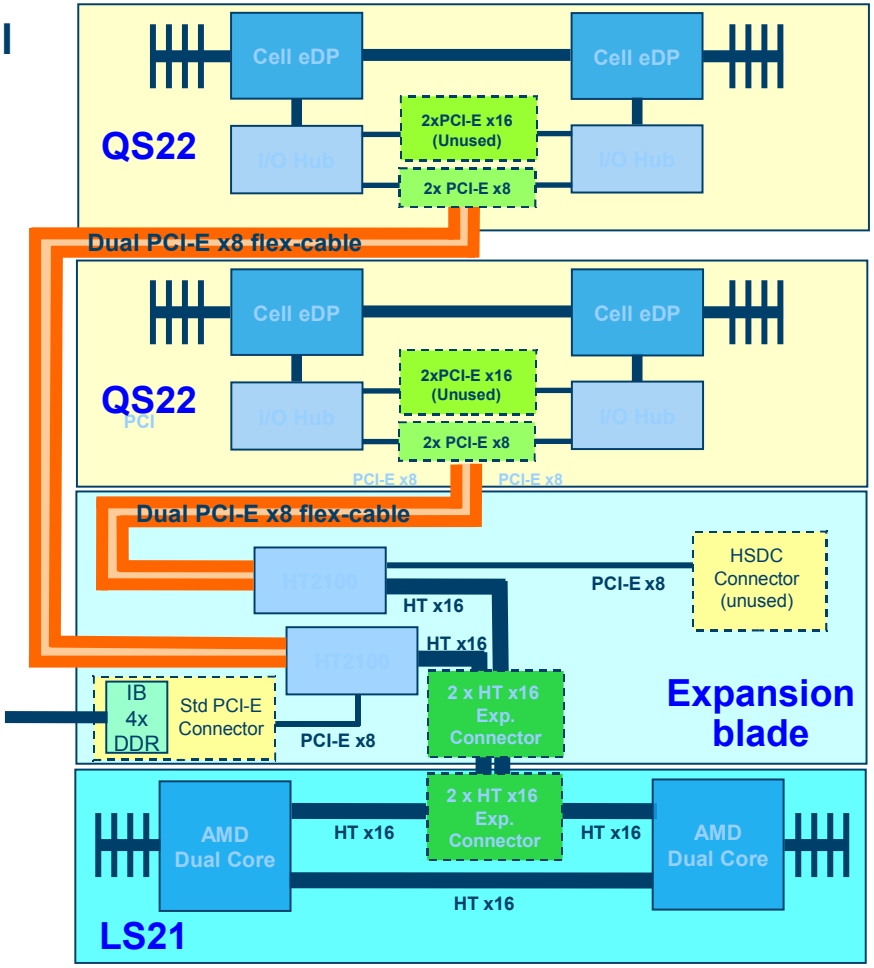


Roadrunner
A First Step in Hybrid/Heterogeneous Computing

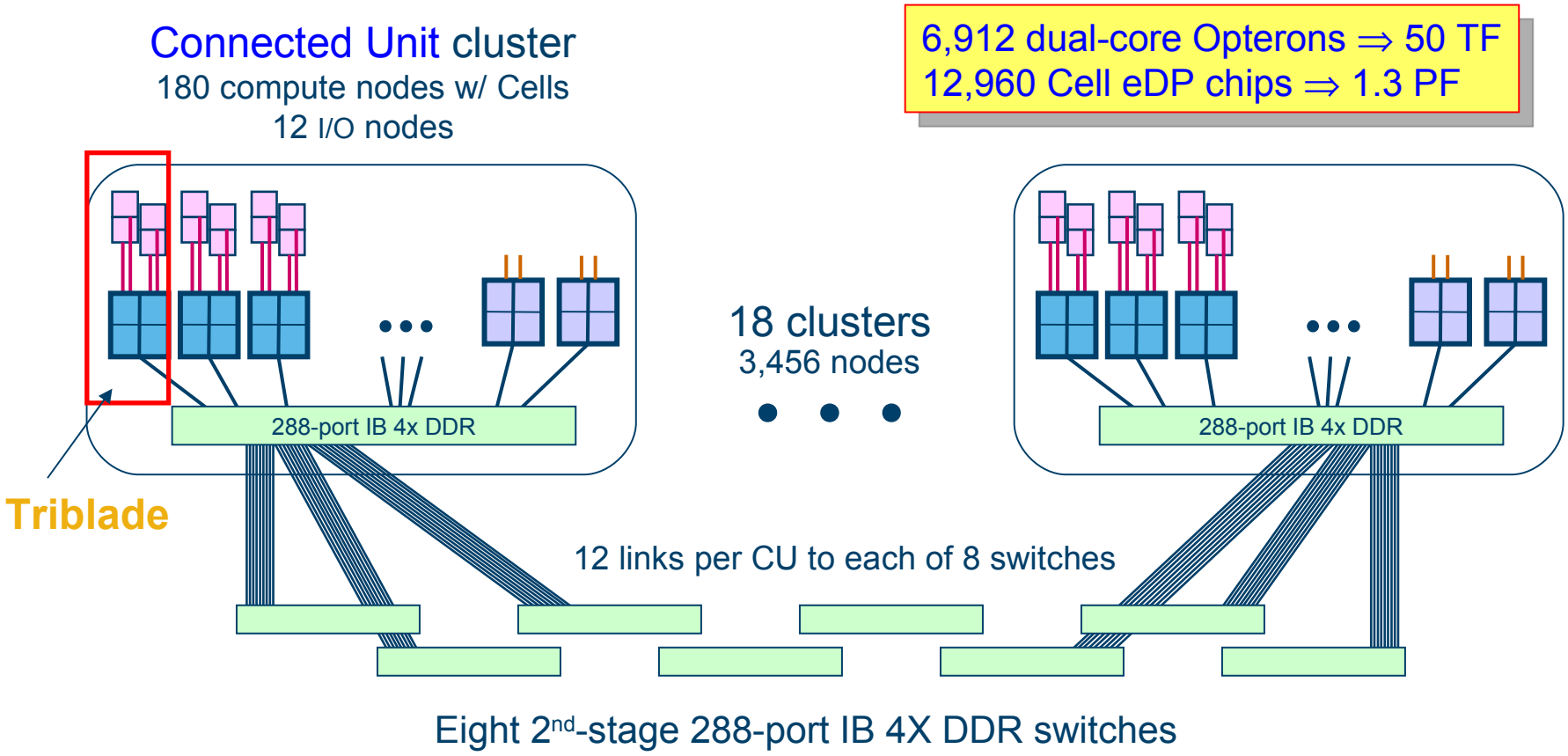
THE LINUX FOUNDATION

A Roadrunner Triblade node integrates Cell and Opteron blades

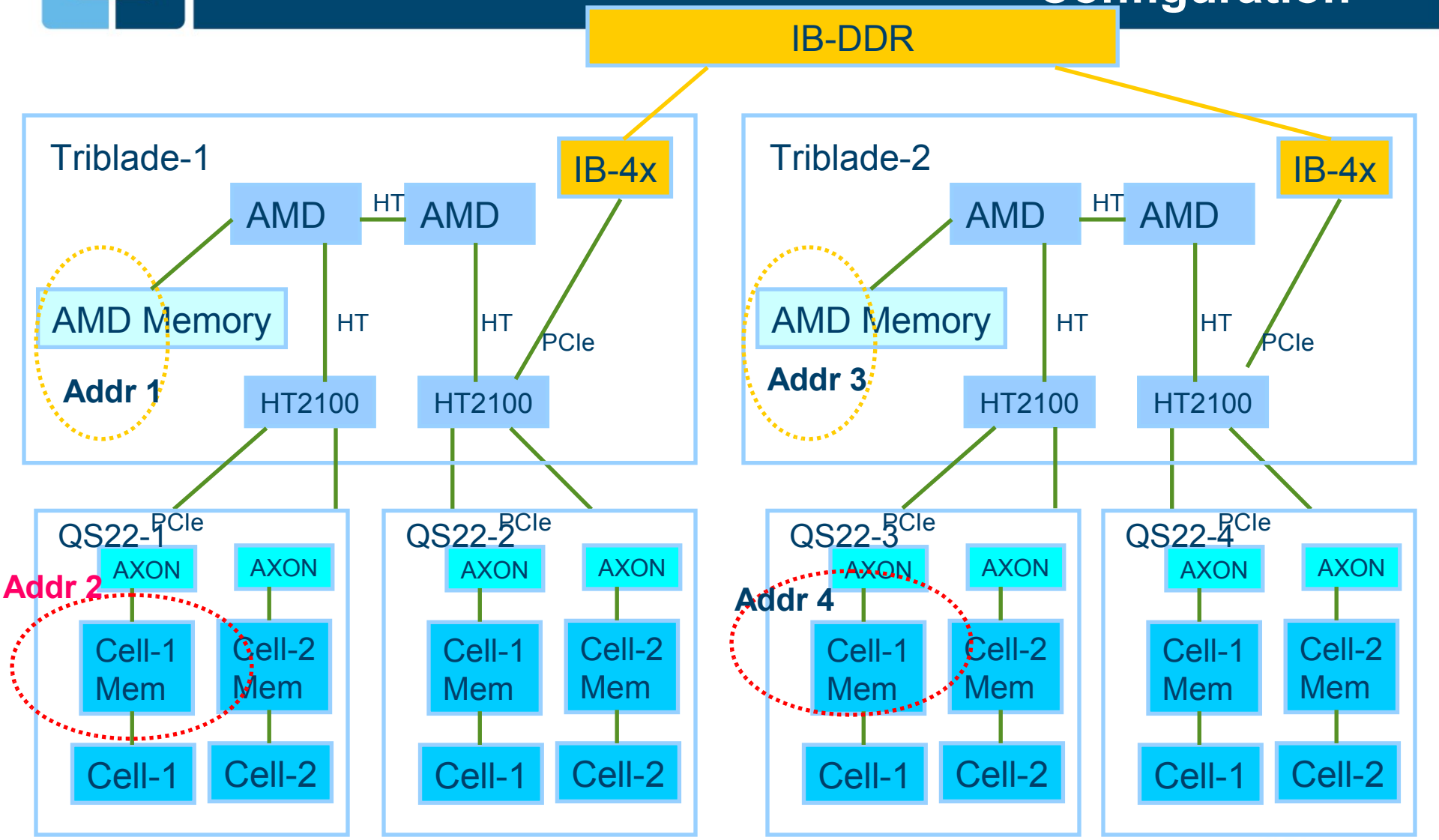
- **QS22** is the newly announced IBM Cell blade containing two new enhanced double-precision (eDP/PowerXCell™) Cell chips
- Expansion blade connects two **QS22** via **four PCI-e x8** links to **LS21** & provides the node's ConnectX IB 4X DDR cluster attachment
- **LS21** is an IBM dual-socket Opteron blade
- 4-wide IBM BladeCenter packaging
- Roadrunner Triblades are completely diskless and run from RAM disks with NFS & Panasas only to the LS21
- Node design points:
  - One Cell chip per Opteron core
  - ~400 GF/s double-precision & ~800 GF/s single-precision
  - 16 GB Cell memory & 16 GB Opteron memory



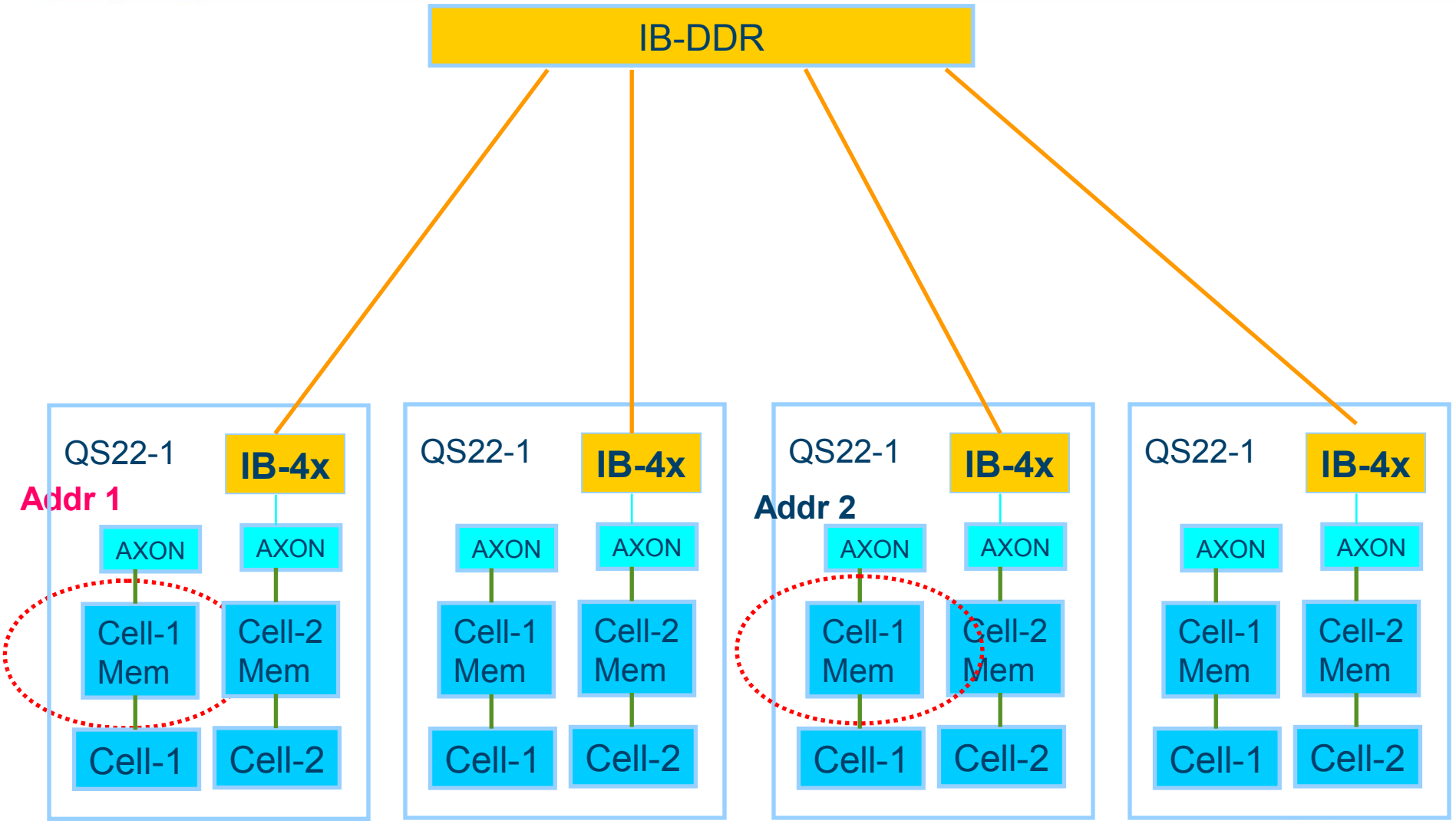
Roadrunner is a hybrid petascale system delivered in 2008



Roadrunner System Address Space Configuration



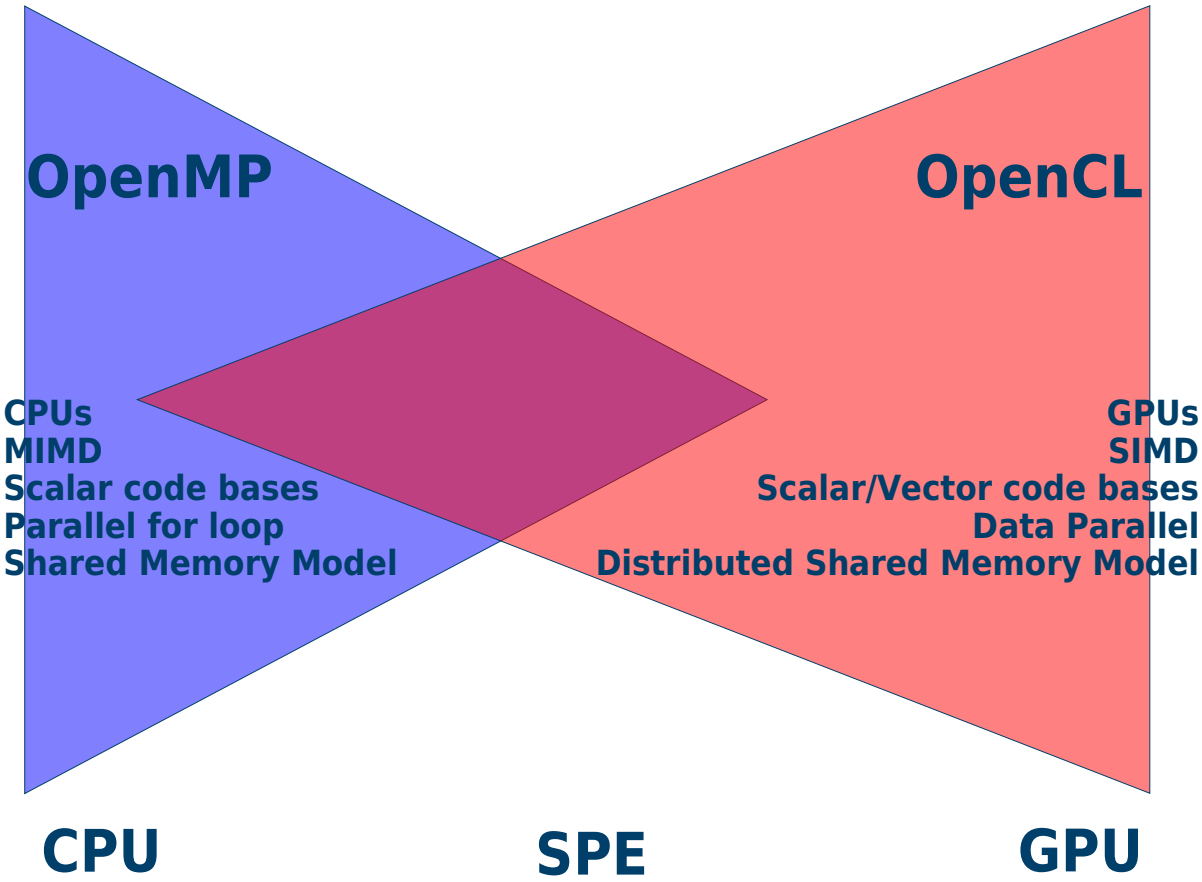
Heterogeneous System Address Space Configuration



Node Level Software
OpenMP – OpenCL - Migration

Two Standards

- **Two standards evolving from different sides of the market**



The OpenCL specification consists of three main components

- **A Platform API**
- **A language for specifying computational kernels**
- **A runtime API.**

- The platform API allows a developer to query a given OpenCL implementation to determine the capabilities of the devices that particular implementation supports.

- Once a device has been selected and a context created, the runtime API can be used to queue and manage computational and memory operations for that device.

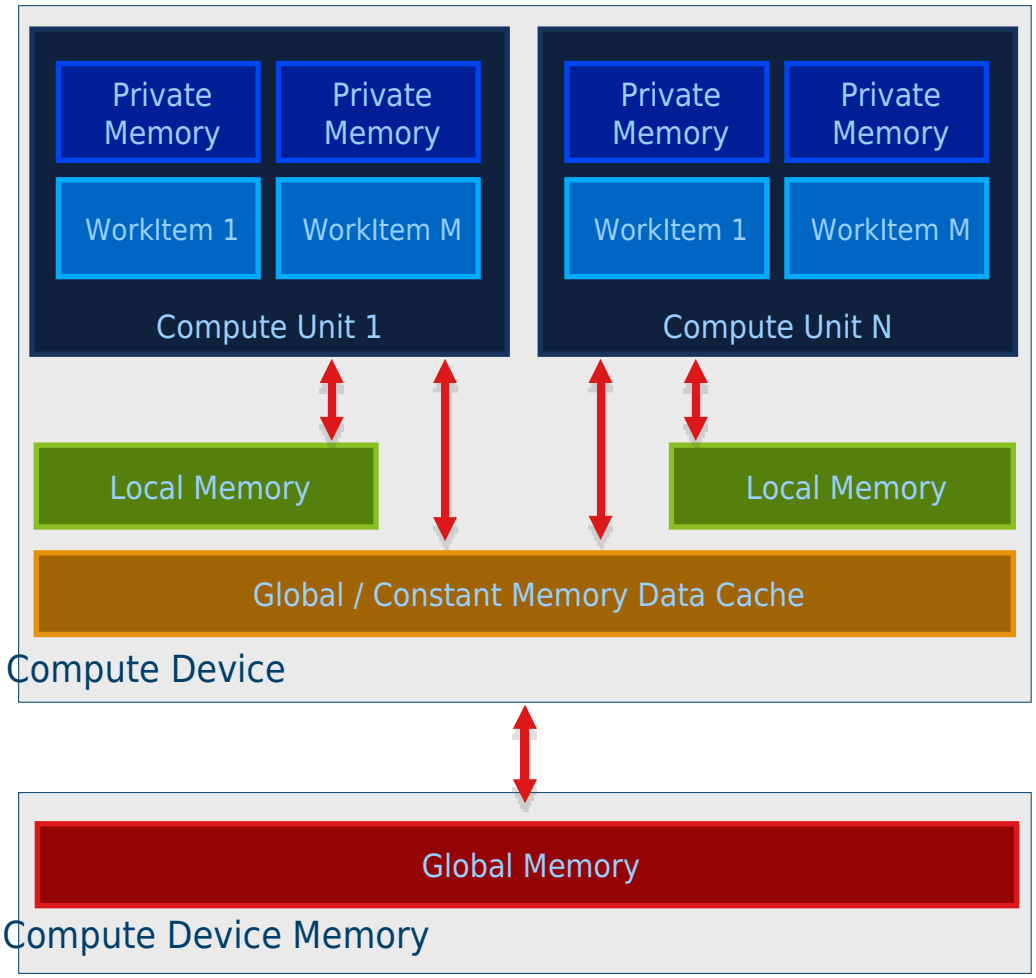
- OpenCL manages and coordinates such operations using an asynchronous command queue.

- OpenCL command queues can include data‐parallel computational kernels as well as memory transfer and map/unmap operations.

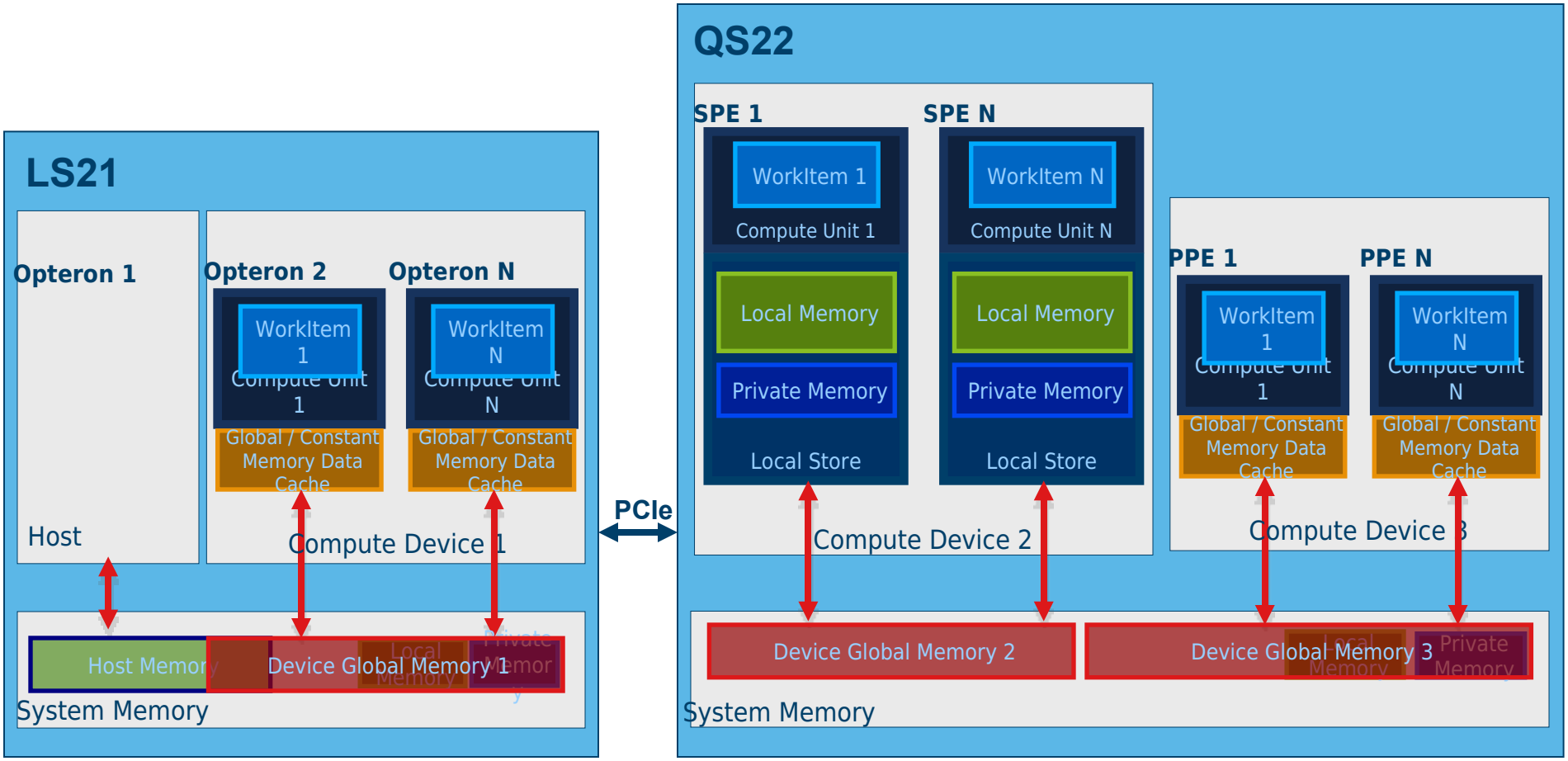
- Asynchronous memory operations are included in order to efficiently support the separate address spaces and DMA engines used by many accelerators.

OpenCL Memory Model

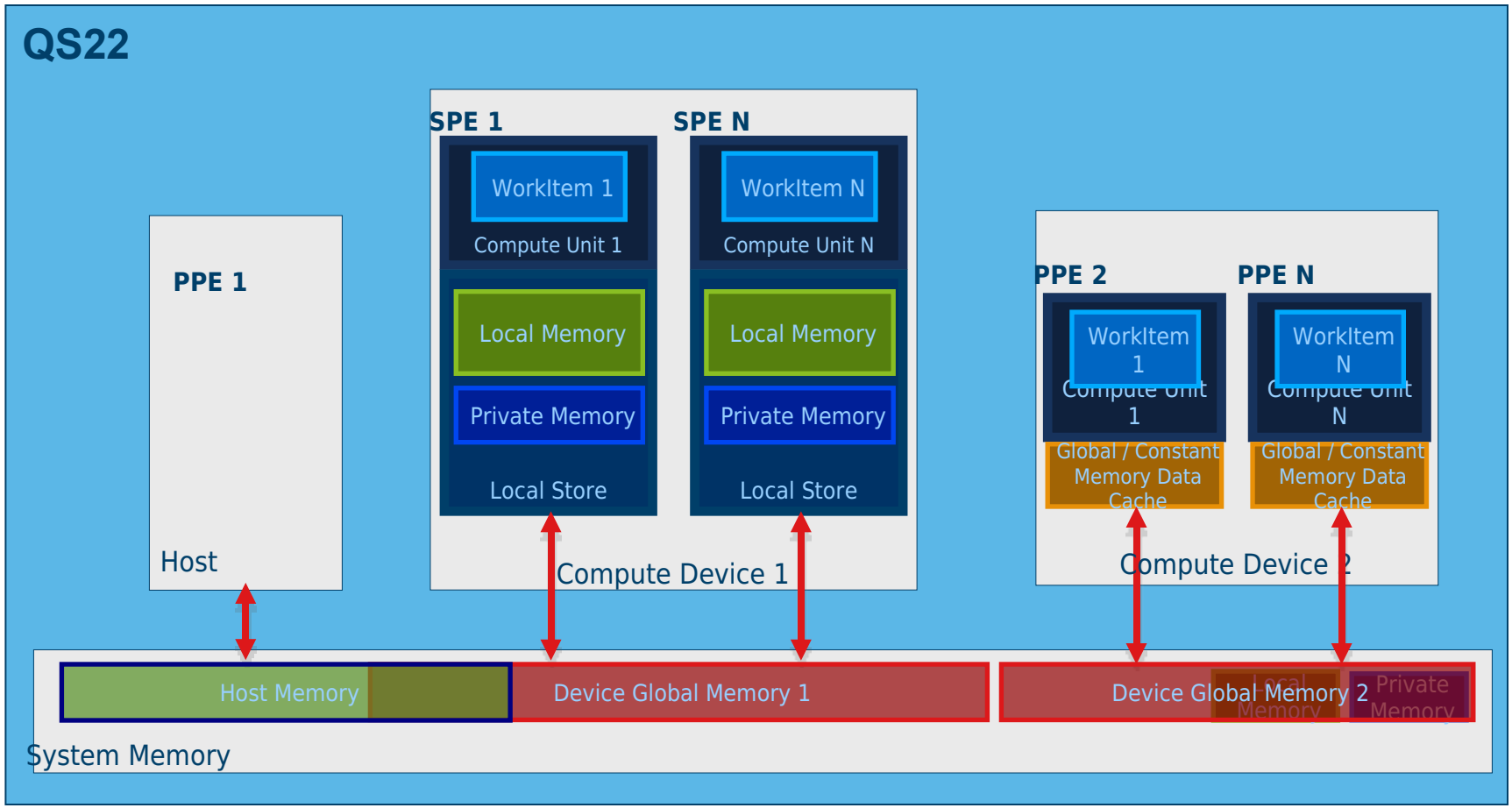
- **Shared memory model**
  - Release consistency
- **Multiple distinct address spaces**
  - Address spaces can be collapsed depending on the device's memory subsystem
- **Address Qualifiers**
  - __private
  - __local
  - __constant and __global
- **Example:**
  - __global float4 *p;



OpenCL on RoadRunner



OpenCL on QS22



Application Issues

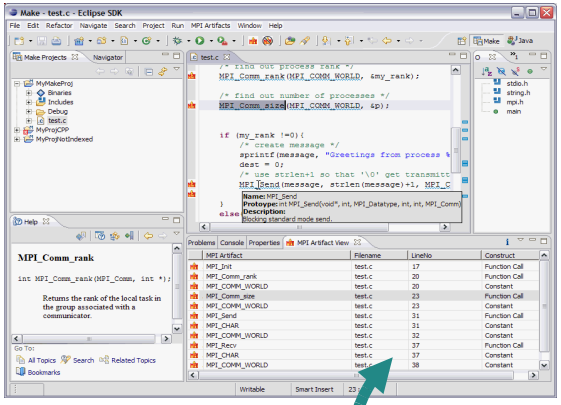
- **Source available?**
  - binary ISVs
  - public open source
  - private source code
  - new code being developed
- **Open standards**
  - established APIs
  - compiler generated and optimized
  - language generated (OpenMP, OpenCL, etc.)

Linux Issues

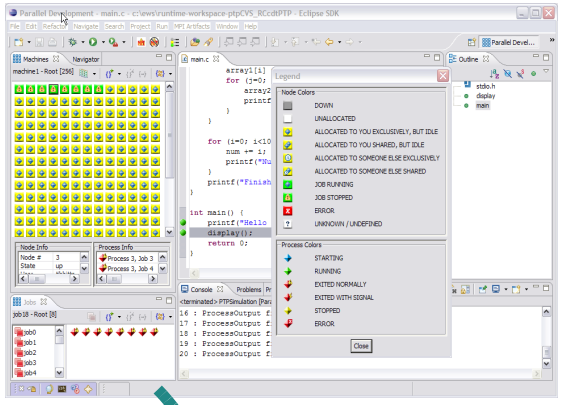
- **Accelerator access and control**
  - Data and code movement to/from accelerator memory
  - Accelerator control
  - Scheduling of accelerator work
  - Synchronization of accelerator access
  - Security, performance monitoring, application debugging, serviceability, metering
- **Compilers**
  - Support for accelerator instruction set architecture
  - Fat binaries?
  - Support for OpenCL

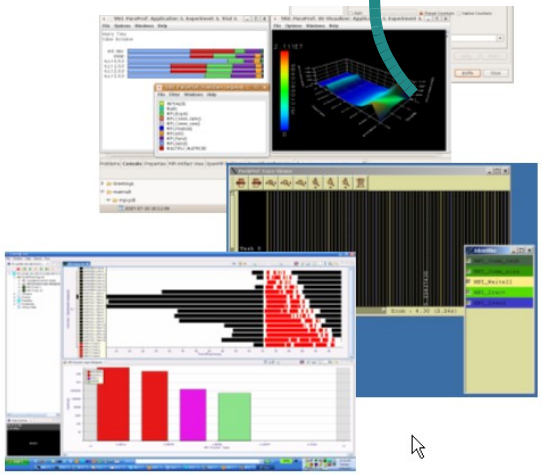
Hybrid Development Application Environment

Coding & Analysis Tools

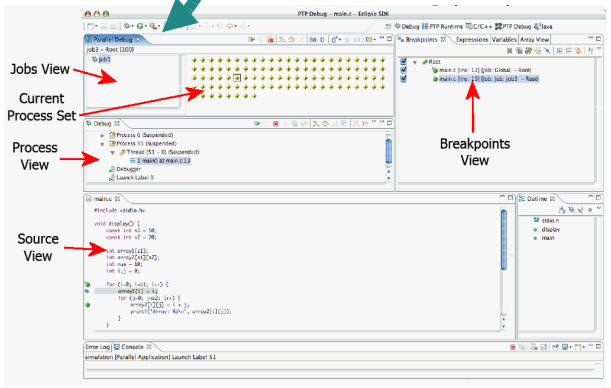


Launching & Monitoring Tools





Performance Tuning Tools



Debugging Tools

Thank you...

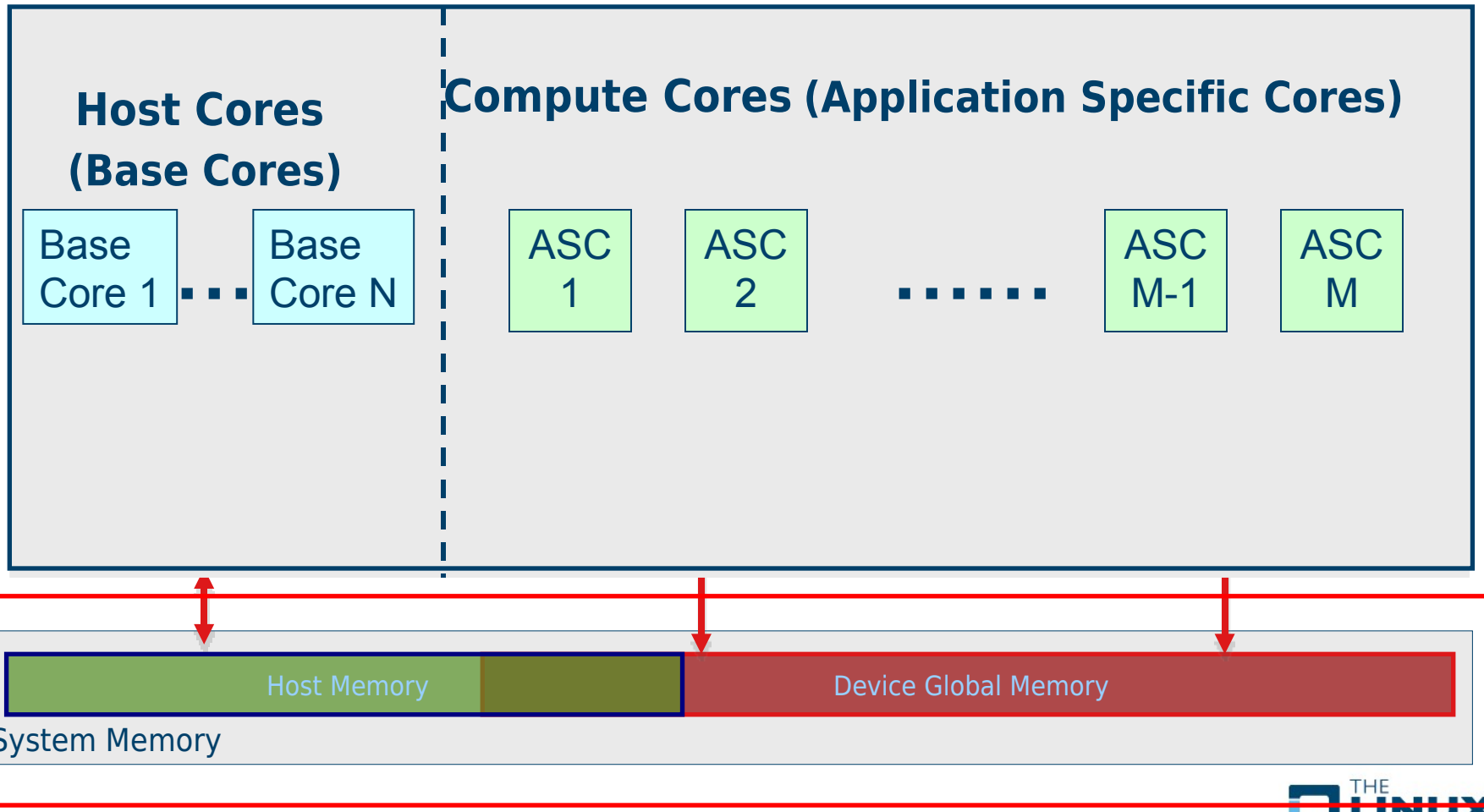
...any Questions?

Heterogeneous Computing vs Hybrid Computing (Backup)

Heterogeneous Node/Server Model

Single Memory System – 1 Address Space

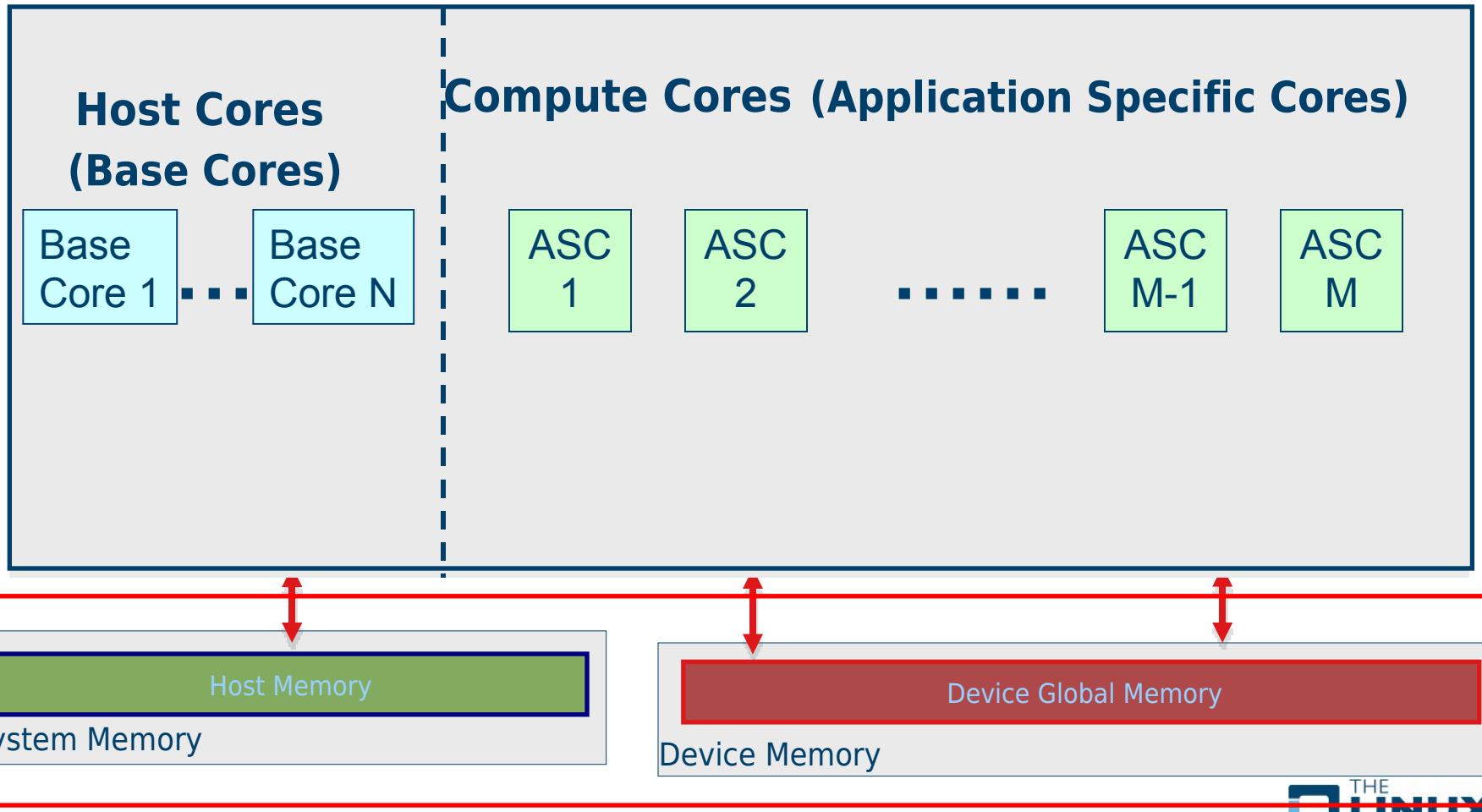
Physical Chip Boundaries Not Architectural Requirement



Hybrid/Accelerator Node/Server Model

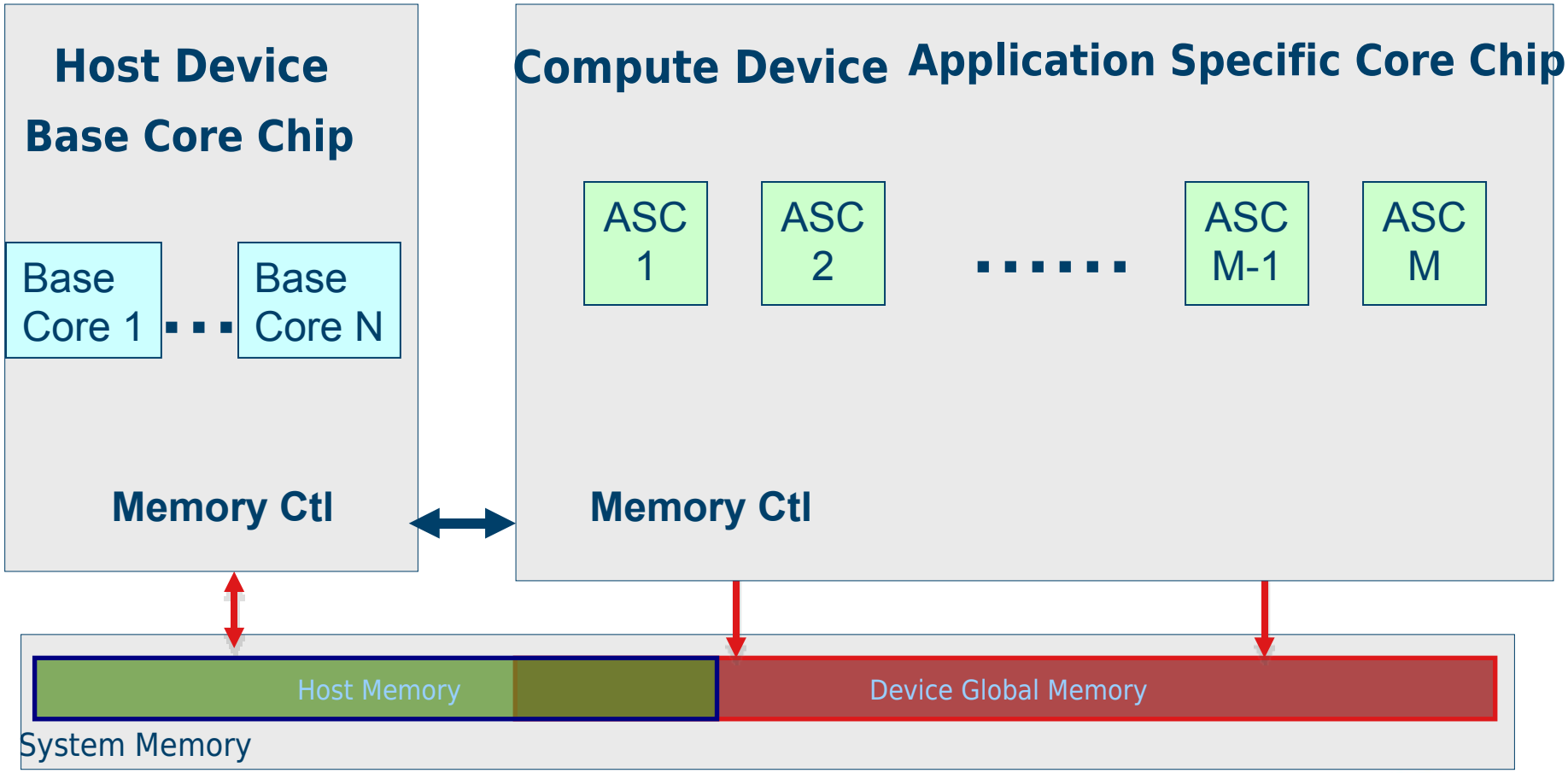
2 Address Spaces

Physical Chip Boundaries not Architectural Requirement



Heterogeneous Node/Server Model

One Possible Multi-Chip Split
Memory Connectivity is Required



Hybrid/Accelerator Node/Server Model

One Possible Multi-Chip Split
Memory Connectivity is NOT Required