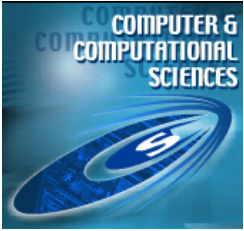




What does Heterogeneity bring?

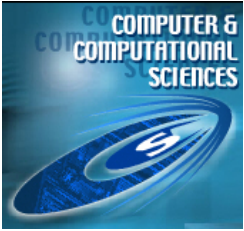
Ken Koch
Scientific Advisor, CCS-DO
Los Alamos National Laboratory

LACSI 2006 Symposium
October 18, 2006



Some Terminology

- **Homogeneous**
 - Of the same or similar nature or kind
 - Uniform in structure or composition throughout.
- **Heterogeneous:**
 - Consisting of dissimilar elements or parts; not homogeneous
- **Hybrid**
 - combine two or more different technologies
 - A computer designed to handle both analog and digital data. Also known as analog-digital computer.
- **Metacomputer**
 - coupling of parallel systems
 - recognize the strengths of each machine, and use it accordingly



Chip-Level Heterogeneity

- Past:
 - Moore's Law has provided increased logic density
 - Chips got more powerful and more capable
 - Flops, registers, caches, out-of-order, pending memory ops
 - Processor speed has increased due to smaller feature sizes
- Present
 - Moore's law still works BUT
 - Clock speeds have reached a plateau
 - Processor complexity and instruction level parallelism appears to have reached a plateau as well
 - Multi-core processors are now the norm
 - memory & off-chip BW per core is actually decreasing



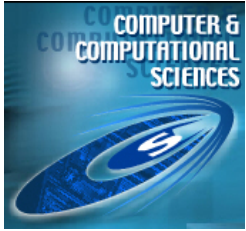
Chip-Level Heterogeneity

- Future
 - More and more devices per chip (4,8,64,128,...)
 - Moore's Law still applies
 - Large device counts could lead to arrays and tiles of “interconnected devices”
 - Not just multi-core
 - Multiple memory hierarchies are likely
 - Special purpose devices almost certainly will be added



System-Level Heterogeneity

- You are probably using one now!
 - Desktop PC or gaming laptop with CPU and a separate graphics card
- Current Machines & Devices
 - Cray XD1 (FPGA)
 - SRC MapStation (FPGA)
 - GPGPU (computing on graphics cards)
 - Clearspeed SIMD PCI cards
 - FPGA PCI cards
 - Cell BE chip



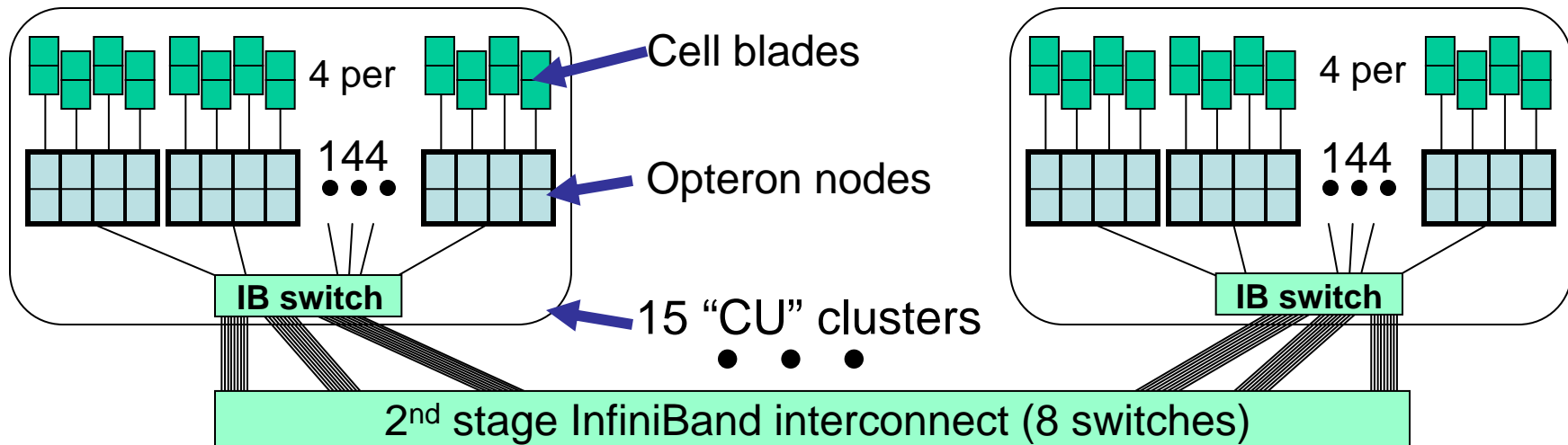
System-Level Heterogeneity

- New node-level initiatives for accelerators
 - AMD Torrenza (socket, HT & HTX level devices)
 - Intel/IBM Geneseo (PCIe card level devices)
- New heterogeneous/hybrid HPC systems
 - LANL Roadrunner (Opterons w/ Cell Blades)
 - Tokyo Institute of Technology TSUBAME (Opterons w/ Clearspeed cards)
 - TeraSoft (Cell Blade cluster)

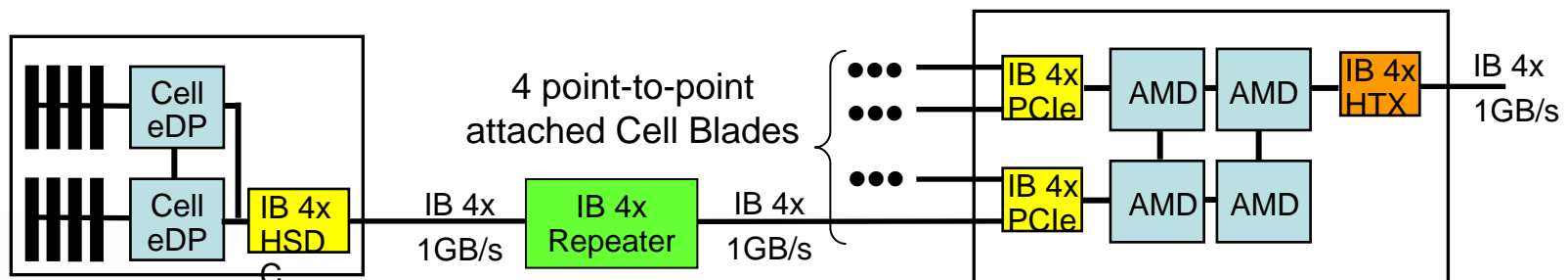


Roadrunner Architecture

Final 2008 Cell-Accelerated System



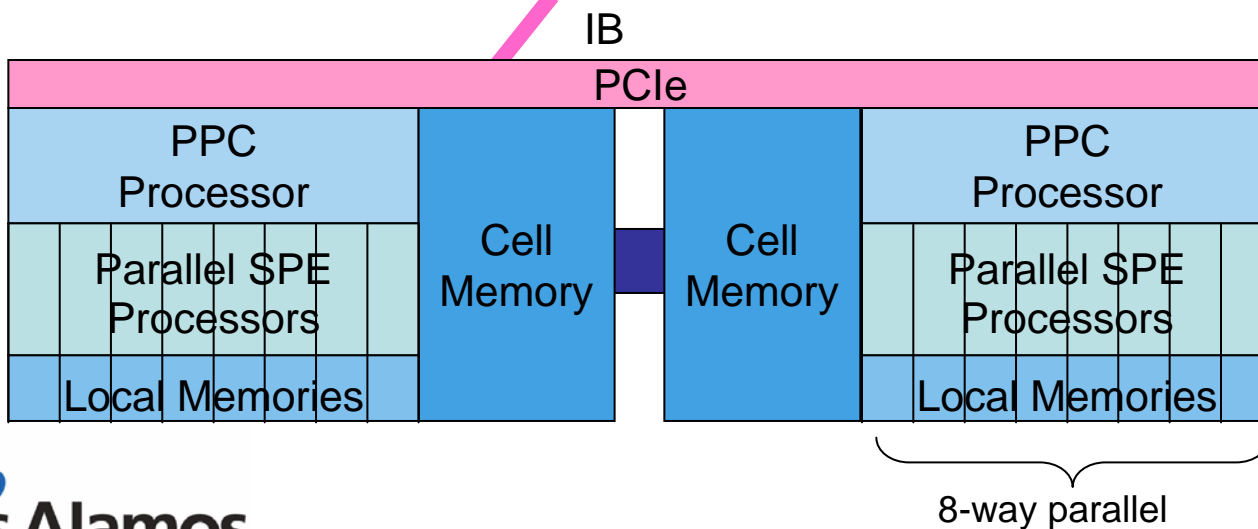
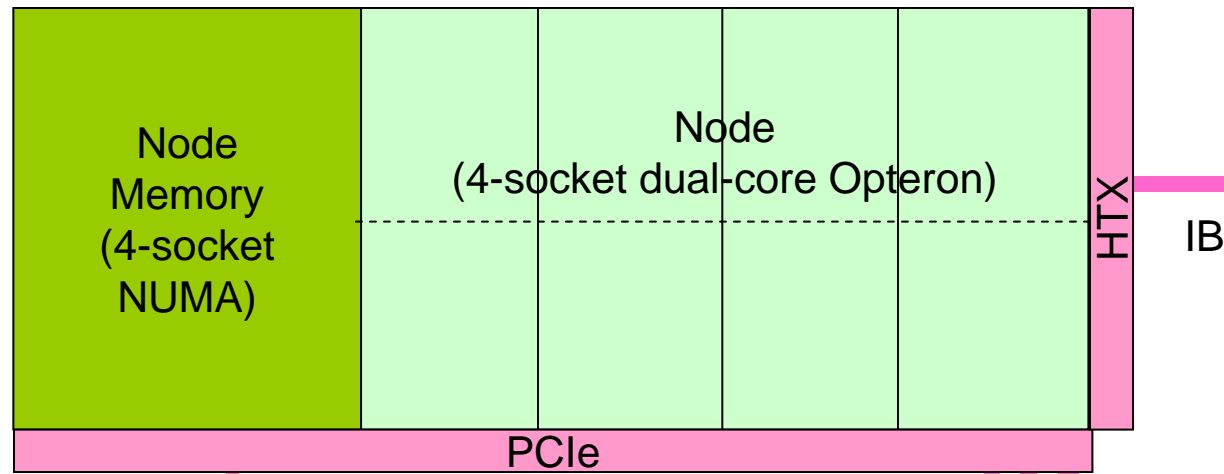
One Accelerated Node





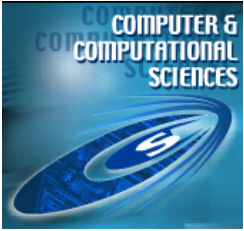
Roadrunner Heterogeneity

An ~850 GFlop Node!



3 more dual-Cell Blades per node





How is programming heterogeneous systems/devices similar to how we program now?

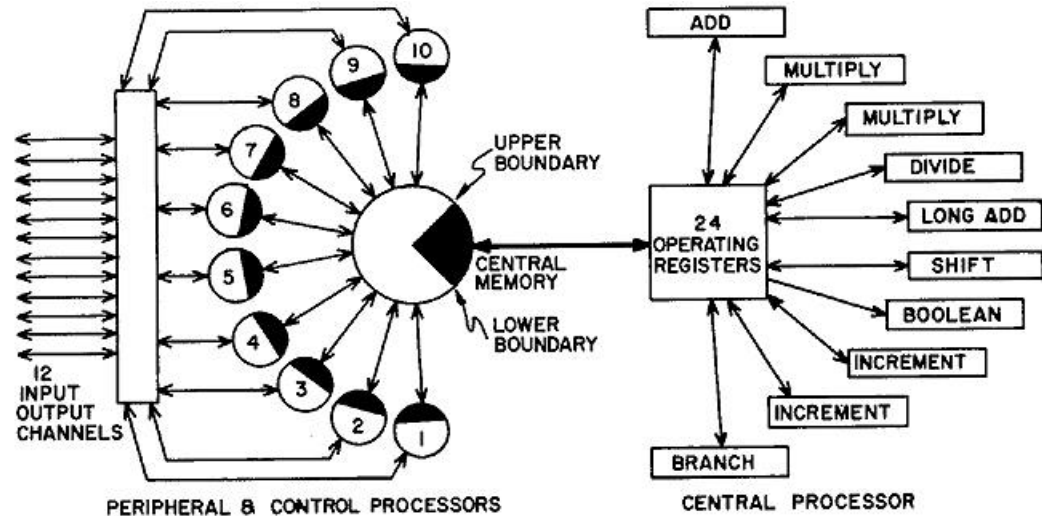
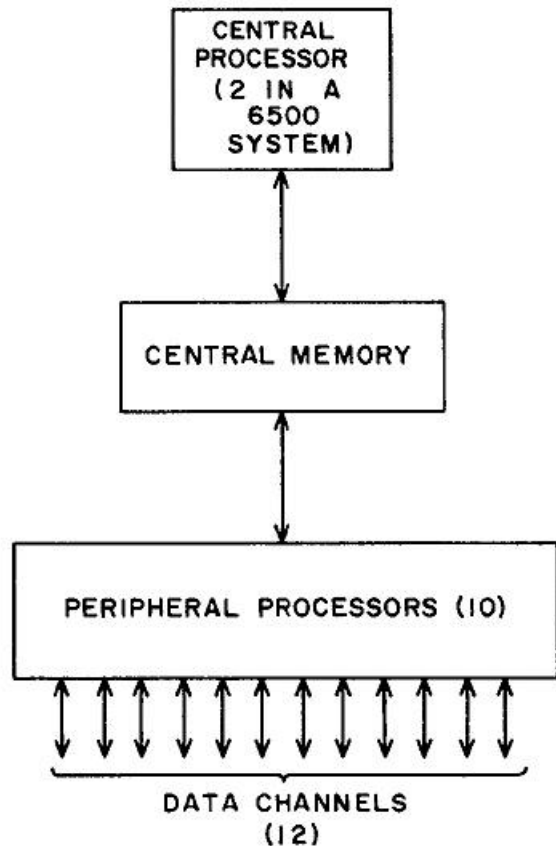
- Parallel execution
 - Threads
 - OpenMP (possibly, just NOT at the loop level)
- Tiling, work queues, blocking of data
- Communication/Data Transfers
 - Overlapped communication & computation, not just amortization
 - send/recv/put/get data communications (MPI like)
- Vector operations (SSE, Cray, data dependencies)
- Amdahl's law still applies to any parallel speedup
- C language (lowest common denominator)



How is programming heterogeneous systems/devices different to how we program now?

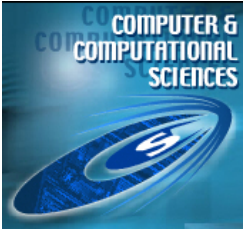
- Decompose code into 2, 3, or more cooperating parts with different functions
 - Compute, coordination, upload/download, relay messages
- Explicit user visibility of memories & data transfers
 - DMAs and overlap of communication & computation
 - Local memories of limited sizes
 - Data alignment constraints
- Worry about granularity of code parts
 - Flops vs. data bandwidth requirements
 - Blocks of code, entire routines, collection of routines implementing an algorithm, entire “physics package”
- Endian conversions may be required (e.g. Opteron-Cell)

Deja Vu



CDC 6600

First acknowledged supercomputer
 It was HETEROGENEOUS!
 Mid 1960's



Deja Vu

- “Experienced” (i.e. old) programmers may actually have some knowledge advantages!
 - “Teach new dogs old tricks”
- Techniques from the past with similarities to heterogeneous programming ideas
 - CDC 6600/7600 Peripheral Processors (10-way I/O coprocessor)
 - CDC 6600 ECS & 7600 LCM/SCM (explicit memory hierarchies)
 - Out-of-core buffered I/O (overlapped data streaming)
 - Fortran overlays (program image size)
 - Cray vector techniques for non-obvious kernels
 - Floating Point Systems Array processors
 - Analog-Digital computer (hybrid programming)
 - IBM JCL & DD cards (just kidding)

