The Cell at Los Alamos: From Ray Tracing to Roadrunner

Allen McPherson
Los Alamos National Laboratory

Thanks to Ben Bergen, John Turner, Cornell Wright, Ken Koch, and Andy White for slides and content
Three Subjects to Talk About Today

- **Part I**
  - Visualization at Los Alamos
  - Brief history and path to Cell

- **Part II**
  - Roadrunner
  - Assessment process
  - Architecture
  - Recent accomplishments and future plans

- **Part III**
  - Thoughts on programming Roadrunner-like architectures
  - Current research program in this area
Part I: Visualization Program at Los Alamos

- **Support ASC & Open Science visualization requirements**
  - Safety, security, reliability of stockpile without underground testing
  - Climate modeling, astrophysics, etc.

- **Production team**
  - Provide and manage resources and facilities
  - Provide support to end users
  - Enable visualization of “current” size workloads

- **Research team**
  - Look 3-5 years out for solutions to “future” size workloads
  - Build and demonstrate technology that is not provided by commodity tools
  - Transition research to production

- **Three “eras” of large-scale visualization at Los Alamos**
Blue Mountain Era Visualization

- **Blue Mountain (late 1998)**
  - SGI shared memory machine
    
    "Blue Mountain ran Linpack, one of the computer industry’s standard speed tests for big computers, at a fast 1.6 trillion operations per second (teraOps), giving it a claim to the coveted top spot on the TOP500 list, the supercomputer equivalent of the Indianapolis 500."
  - Integrated Reality Engine graphics ($250K/each)
  - Initiation of “standard” visualization package (Ensight)
  - First of the large “immersive” facilities
  - Beginning of hardware infatuation for research guys
    - Interactive “movies”
    - Volume rendering
Interactive “Movies” and Volume Rendering
Commodity Cluster Era Visualization

- **Fruits of multi-year research effort**
  - Leverage commodity technology to replace SGI infrastructure
    - “Game” cards
    - PC-class nodes
    - Infiniband networks
  - Continued use of commodity software platform (Ensight)
  - Support expanded infrastructure
    - Power wall
    - Offices
    - Cave

- **Delivered late 2004**
  - 266 node cluster with nVidia Quadro cards
    - Genlock, frameland
    - “500 times faster than SGI systems”
“Next Generation” Visualization Era

- Continuing 3-5 year out philosophy…
  - How will we support coming platforms (petaflop and beyond)
  - Maintain and upgrade current commodity clusters
  - Move towards “in situ” visualization and analysis
    - For the really big stuff
    - For users that are not being served
  - Wrote proposal for exploratory effort in ray tracing on Cell

“We believe that future petaflop platforms will not be composed of commodity PCs with game graphics cards. In fact, they will likely be specialized machines with novel architectural features such as instruction sets, memory systems and interconnects. We must develop a new approach that uses the special capabilities of these petaflop platforms—much as we developed the current approach that uses the special capabilities of commodity graphics cards. This new approach will be software based (as opposed to GPU-based) and use a ray-tracing technique to generate images and provide integrated analysis of simulation data.”
Why Ray Tracing?

- **Advanced rendering model**
  - Easily handles existing modalities
  - Shadows, reflections, refractions, etc.
  - Flexible software-based approach
  - Ability to integrate compute, analysis & rendering
  - Now possible to run fast on things like Cell

Images courtesy Christiaan Gribble, Grove City College, PA (done while at Univ. of Utah)
Ray Tracing on Cell

- Research project was funded
- Acquired Cell hardware
  - 1st hardware delivered outside IBM
  - 4-node DD2 cluster
- Initial benchmarking and testing
- Put on hold for…Roadrunner
Part II: Roadrunner

- **Phase I**
  - Redtail Base System
    - 76 Teraflop/s (peak)
    - Opteron Cluster
      (Replacement for Q)

- **Phase II**
  - Assessment
    - Evaluation of Cell potential for HPC

- **Phase III**
  - Roadrunner
    - 1.3 Petaflop/s (peak DP)
    - Cell-Accelerated Opteron Cluster
Roadrunner is a Cluster-of-Clusters

Connected Unit cluster
180 Triblade compute nodes w/ Cells
12 I/O nodes

6,480 dual-core Opterons ⇒ 23.3 Tflop/s (DP)
12,960 Cell eDP chips ⇒ 1.3 Pflop/s (DP)

18 clusters

12 links per CU to each of 8 switches

Eight 2nd-stage 288-port IB 4X DDR switches
Roadrunner is CBE-accelerated, not a cluster of CBES

Node-attached Cells is what makes Roadrunner different!
Roadrunner nodes have a memory hierarchy

- **QS22 Cell blades**
  - 256 KB of “working” memory (per SPE)
  - 25.6 GB/s off-SPE BW
  - 4 GB of shared memory (per Cell)
  - 256 KB of “working” memory (per SPE)
  - 25.6 GB/s/chip (w/ 800 DDR2)

- **LS21 Opteron blade**
  - 8 GB of NUMA shared memory (per blade)
  - 8 GB of shared memory (per socket)
  - 16 GB of distributed memory (per node)
  - 5.4 GB/s/core

- **ConnectX IB 4X DDR**
  - 4 GB of memory (per core)
  - 25.6 GB/s/chip (w/ 800 DDR2)

- **PCIe x8** (2 per blade)
  - (2 GB/s, 2 us)
  - One Cell chip per Opteron core

- **~200 GB/s per Cell on EIB bus**

- **“equal memory size” concept**
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.
For the Assessment in October 2007, needed to answer the following questions…

Will Roadrunner be applicable to the current and future application workload?

1. Can we can program the Cell processor?
   - do sufficient tools (compilers, etc.) exist?

2. Can we achieve significant performance gains on the Cell processor?

3. Can we program Opteron / Cell hybrid nodes?

4. Can we achieve significant performance gains in hybrid?
Prototype hardware was used for applications testing

- InfiniBand-connected current generation Cell-blades

- Advanced Architecture Initial System aka. AAIS
  (Operational January 2007, used 4-node viz Cell cluster during 2006)

- limited time on single eDP (IBM PowerXCell 8i) chips since summer
We have focused on important application codes.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VPIC</strong></td>
<td>Fully-relativistic, charge-conserving, 3D explicit particle-in-cell code.</td>
</tr>
<tr>
<td>(8.5K lines)</td>
<td></td>
</tr>
<tr>
<td><strong>SPaSM</strong></td>
<td>Scalable Parallel Short-range Molecular Dynamics code, orig. developed for the CM-5.</td>
</tr>
<tr>
<td>(34K lines)</td>
<td></td>
</tr>
<tr>
<td><strong>Milagro</strong></td>
<td>Parallel, multi-dimensional, object-oriented code for thermal x-ray transport via Implicit Monte Carlo on a variety of meshes.</td>
</tr>
<tr>
<td>(110K lines)</td>
<td></td>
</tr>
<tr>
<td><strong>Sweep3D</strong></td>
<td>Simplified 1-group 3D Cartesian discrete ordinates (Sn) kernel representative of the PARTISN neutron transport code.</td>
</tr>
<tr>
<td>(2.5K lines)</td>
<td></td>
</tr>
</tbody>
</table>
Cell and hybrid speedup results

<table>
<thead>
<tr>
<th>Application</th>
<th>Type</th>
<th>Class</th>
<th>Cell Only (kernels)</th>
<th>Hybrid (Opteron+Cell)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>CBE</td>
<td>eDP</td>
</tr>
<tr>
<td>SPaSM (10/07)</td>
<td>Science</td>
<td>full app</td>
<td>3x</td>
<td>4.5x</td>
</tr>
<tr>
<td>SPaSM (now)</td>
<td>Science</td>
<td>full app</td>
<td>5x</td>
<td>7.5x</td>
</tr>
<tr>
<td>VPIC</td>
<td>Science</td>
<td>full app</td>
<td>9x</td>
<td>9x</td>
</tr>
<tr>
<td>Milagro</td>
<td>IC</td>
<td>full app</td>
<td>5x</td>
<td>6.5x</td>
</tr>
<tr>
<td>Sweep3D</td>
<td>IC</td>
<td>kernel</td>
<td>5x</td>
<td>9x</td>
</tr>
</tbody>
</table>

- all comparisons are to a single Opteron core
- parallel behavior unaffected, as will be shown in the scaling results
- first 3 columns are measured, last column is projected
These results were achieved with a relatively modest level of effort.

<table>
<thead>
<tr>
<th>Code</th>
<th>Class</th>
<th>Language</th>
<th>Lines of code</th>
<th>FY07 FTEs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Orig.</td>
<td>Modified</td>
</tr>
<tr>
<td>VPIC</td>
<td>full app</td>
<td>C/C++</td>
<td>8.5k</td>
<td>10%</td>
</tr>
<tr>
<td>SPaSM</td>
<td>full app</td>
<td>C</td>
<td>34k</td>
<td>20%</td>
</tr>
<tr>
<td>Milagro</td>
<td>full app</td>
<td>C++</td>
<td>110k</td>
<td>30%</td>
</tr>
<tr>
<td>Sweep3D</td>
<td>kernel</td>
<td>C</td>
<td>3.5k</td>
<td>50%</td>
</tr>
</tbody>
</table>

- all staff started with little or no knowledge of Cell / hybrid programming
- 2 x 1 denotes separate efforts of roughly 1 FTE each
- most efforts also added code
We have demonstrated that the accelerated hybrid model works for applications of interest to NNSA/ASC.

Roadrunner is applicable to the current and future application workload.

1. You can program the Cell processor.
2. You can achieve significant performance gains on the Cell processor.
3. You can program Opteron / Cell hybrid nodes.
4. You can achieve significant performance gains in hybrid.
Top 500 Petaflop Run

- Backstop modeling by LANL PAL team
- PF accomplishment largely due to:

Mike Kistler  Brad Benton  John Gunnels  Dan Brokenshire
Roadrunner first achieved a petaflop/s at 3:30 am, Monday, May 26.

Calculation: 2 hours

Performance: 1.026 petaflop/s

Finished 2 tests with the following results:
2 tests completed and passed residual checks, 0 tests completed and failed residual checks, 0 tests skipped because of illegal input values.
Achieving a petaflop/s in less than 3 days demonstrates the stability of the Roadrunner system.

- **Full system available**
  - 8:30 am, Friday, May 23
- **Full system job launch tests begin**
  - 3:00 pm, Friday, May 23
- **First full system LINPACK launch**
  - 8:30 pm, Friday, May 23
  - Node failure after running an hour
- **Successful LINPACK runs**
  - 5:45 pm, Saturday, May 24 (879 TF/s)
  - 2:45 pm, Sunday, May 25 (945 TF/s)
  - 1:10 am, Monday, May 26 (997 TF/s)
  - 3:30 am, Monday, May 26 (Petaflop/s)

Roadrunner is also very energy efficient, 437 MF/s per watt.
VPIC will address grand challenges in plasma physics.

VPIC achieved a sustained 374 TF/s on the full RR system (SP)
SPaSM is a framework for materials science research as well as epidemiology, fluid instabilities and turbulence.

SPaSM achieved a sustained 361 TF/s on the full RR system (DP).
The goal of Petavision is synthetic visual cognition.

- A billion ($10^9$) neuron simulation of visual cortex on full Roadrunner system
  - Used simple neurons with Zucker connection weights
  - First large-scale calculation with Zucker weights and spiking neurons
  - Next step is to add a complex neuron layer with stored weights to add learning
- Petavision is a collaboration of scientists from Los Alamos, Yale, MIT and IBM

Petavision achieved a sustained 1.144 PF/s on the full RR system (SP).
There are additional exciting science projects scheduled to run on Roadrunner beginning in 2009.

- Dynamic molecular simulation of breakdown of cellulose for biofuels.
- Cosmological simulation of large scale structure of the universe.
- Phylogenetic analysis of the evolution of acute HIV infection.
- Long-time evolution of the formation & deformation of metallic nanowires.
Part III: Thoughts on Programming; Our Research

- **Hardware is no longer interesting**
  - Continuing delivery of amazing hardware is a *fait accompli*
  - An embarrassment of riches

- **Challenge is productively using this hardware (programming)**
  - Seems to be no obvious or easy solution
  - Fairly wide array of options exist
  - Depends on circumstances

- **Recent pragmatic research work at Los Alamos**
  - Scout
  - Cell MPI
  - *Domain Specific Languages (DSLs)*
  - *Functional languages*
  - *Fault tolerance*
Investigate a Domain Specific Approach

- Scientists work in their own domain
  - Opposite of general purpose
  - Raise the level of abstraction
  - Let scientists work in an environment that represents the problem domain
  - Define domain specific tools, languages, and environments to match their model
  - Carries through to problem set-up, analysis, etc.
  - Portable if well designed and implemented
    - Ideally as portable as the science
    - Much of underlying infrastructure itself needs to be portable
  - Speed (performance) may come along as benefit of abstraction
    - More opportunities for optimization
    - Kernels (math) may still need to be at the metal
      - Application “construction” may be done with DSLs
Use Functional Languages for Infrastructure

- Users (scientists) never see these!
  - Need to partner computer scientists with physicists
- Single assignment semantics are attractive for concurrency
- Quite a bit of work in embedded (and external) DSL tools
- However, array support tends to be minimal
  - Typically not designed for number crunching
- Example languages
  - Haskell (lots of research effort)
  - OCaML (supposedly compiles to “fast” code, untested by us)
  - Erlang (production language from the telecommunications industry)
- Steep learning code for computer scientists too
Make an Attempt at Fault Tolerance

- **Current approaches are very basic**
  - Periodic restart dumps
  - Restart computation on machine failure
  - Ad-hoc machine monitoring tools

- **Explore basic fault tolerance approaches**
  - No free lunch
    - Restart dumps require larger (4-10X) disk subsystem
    - Simple fault tolerance requires memory copies
    - Possible to combine two approaches
Current Research Project

- **Use Erlang to implement fault tolerant domain specific computation**
- **Erlang is based on messaging paradigm**
  - Natural for MPI based codes
  - Have run early tests on messaging performance
- **Erlang provides facilities for fault tolerance**
  - Used to construct reliable switching networks
- **Erlang is not great at math**
  - Sequentially through lists can be fast, random or other access slow
  - Use external interface to kernels coded at the metal
  - Use Erlang for data marshalling and distribution
    - May also require custom code
- **Erlang has facilities for parsers**
  - Unexplored at this point
To Wrap Up

- **Exciting time at Los Alamos for computing**
  - Early results on Roadrunner look good, outlook is good
  - We have built a proven capability in heterogeneous programming
  - Exciting plans for open science applications and advanced architecture work

- **Many tough challenges remain**
  - Transition of existing multi-physics codes to Roadrunner
    - The 20 year-old FORTRAN deck problem
  - Efficient computing in the presence of failures
  - Visualization and analysis of Roadrunner and beyond data sizes
  - Even faster machines to come
    - Zia, Trinity, Sequoia, etc.

- **It’s the software that’s important and should be funded as such**
Additional Information

- mcpherson@lanl.gov

- LANL Roadrunner web page and portal
  - http://www.lanl.gov/roadrunner/

- Scout

- Cell Messaging Layer

- POPTEX (ocean visualization)

- TRex (volume rendering)
Roadrunner at a glance

- **Cluster of 17 Connected Units**
  - 12,240 IBM PowerXCell 8i accelerators
  - 6,120 AMD dual-core Opterons (comp)
  - 408 AMD dual-core Opterons (I/O)
  - 34 AMD dual-core Opterons (man)
  - 1,332 Petaflop/s peak (PowerXCell)
  - 44 Teraflop/s peak (Opteron-comp)
  - 1,026 Petaflop/s sustained Linpack

- **InfiniBand 4x DDR fabric**
  - 2-stage fat-tree; all-optical cables
  - Full CU bi-section bi-directional BW
    - 384 GB/s (CU)
    - 3.3 TB/s (system)
  - Non-disruptive expansion to 24 CUs

- **98 TB aggregate memory**
  - 49 TB Opteron
  - 49 TB Cell

- **408 GB/s peak File System I/O:**
  - 204x2 10G Ethernets to Panasas

- **RHEL & Fedora Linux**

- **SDK for Multicore Acceleration**

- **xCAT Cluster Management**
  - System-wide GigEnet network

- **2.35 MW Power (Linpack):**
  - 437 Megaflop/s per Watt

- **Other:**
  - 278 racks
  - 5200 ft²
  - 500,000 lbs.
  - 55 miles of IB cables