SAND2011-0409P

Hardware Architectures Group Recap

June 15-16, 2010

August 23, 2010

September 8, 2010

Participants:

ASC: Sander Lee, Tina Macaluso

LANL: Parks Fields, Ken Koch, Scott Runnels, Cheryl Wampler, Andy White

LLNL: Brian Carnes, Matt Leininger, Mark Seager

SNL: Jim Ang, Doug Doerfler, Sudip Dosanjh, Scott Hemmert, John Noe

Sandia is a Multiprogram Laboratory Operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy Under Contract DE-ACO4-94AL85000.

Going-in Assumptions

- Programming continuity between 2015 and 2018 machines
  - Hardware will evolve through entire timeline of program
  - So, some evolution may occur between 2015 and 2018, but the high level features of the final programming model must exist in 2015
  - We assume an earlier (pre-2015) discontinuity is preferred
    - Allows for overlap of old and new programming models
- There will be a base platforms program plus a possible over-target exascale program
- All areas should be considered during co-design process
  - Exact co-design process still unknown, however
    - Applications will need to change based on new technology constraints
    - How much influence we have over architecture will be dependant on funding levels
      - However, good application requirements data is likely to impact architectures in some way regardless of funding

Going-in Assumptions, cont.

- Key aspect of co-design will be getting our arms around the total application space
  - Data locality/decomposition
  - Level of complexity
  - Availability of task-level parallelism
- An exascale machine will require 100 million to billion way parallelism
- Much of the system architecture will be driven by power/energy
- Resiliency will be a first order concern
  - Commodity roadmap may not drive required reliability
  - System software and runtime will likely have to deal with resiliency
  - Applications may see more of these issues in the future
- Looking for relatively general purpose solutions that are tailored to our application set
  - Not looking for multiple application specific architectures for exascale applications (i.e., no Antons or Grapes)

High Level Architecture Characteristics

- Key question for each area:
  - Which architectural features should be hidden from users, and by which level (compiler, runtime, hardware features, etc)?
  - Which features should be exposed to users, and how?
  - We suspect some users will want it all hidden and others will want it all exposed. How do we make both extremes possible?

- Multiple levels of parallelism to exploit
  - Inter-node
  - Intra-node (likely threads)
  - Intra-thread (vector, SIMD, etc.)

High Level Architecture Characteristics, cont.

- Highly NUMA memory hierarchy
  - Fast DRAM vs. slower DRAM
  - Scratchpad vs. cache
  - Global memory vs. local memory

- Data movement will dominate performance and energy
  - User, runtime and/or compiler will need to carefully manage data movement

- Memory capacity per compute will (likely) be 1 to 2 orders of magnitude lower than today
  - Can NVRAM be leveraged to help?

Cross-cutting Questions

- We expect a discontinuity in programming model
  - Should we push the discontinuity as early, or as late, as possible
- What level of cross-platform portability is required
  - Full source compatibility?
  - Compatibility at library level?
- Are domain specific languages really feasible?
- How well will applications be able to load balance across billion way parallelism?
  - What synchronization primitives could hardware provide to help?
- How much data partitioning can be done by users? How much by runtime system?
  - Data movement will be first order concern

Cross-cutting Questions, cont.

- How will NVRAM impact system software and applications?
  - Does it enable new checkpoint schemes?
  - What are the security concerns?
  - User access to help with reduced memory capacity
- What size test beds are needed for early application exploration? For early system software exploration?
  - These are pre 2015 machines that match 2015 node architecture as close as possible

Areas for Investment

- Memory (DRAM)
  - Bandwidth, latency, capacity, power
  - Synchronization
  - QoS/multiport
  - Partners:
    - Just about everybody
- Processor
  - Method of exploiting parallelism
    - Threads
    - Vectors
    - Both
  - Memory hierarchy
    - Cache vs. scratchpad
  - Power saving methods
    - Needs to be non-disruptive to performance, to extent possible
  - Partners
    - UHPC

Areas for Investment

- **Interconnects**
  - **Silicon Photonics are key**
  - **Topologies**
  - **QoS/Congestion control**
  - **Matching hardware to communication model**

- **System Simulators/Emulators**
  - **Infrastructure to test out new ideas before hardware prototypes are available**
  - **Potential Partners:**
    - **Everybody…**

NNSA National Nuclear Security Administration

Los Alamos NATIONAL LABORATORY

Sandia National Laboratories

Areas for Investment

- **Fundamental technologies**
  - **3-D packaging**
  - **Silicon photonics**
  - **Low power circuitry**

- **Packaging and Infrastructure**
  - **Power distribution**
  - **Ease of servicing components**
    - **Component density**
    - **Cooling methods**

- **Role of NVRAM**
  - **Only for I/O?**
  - **Applications accessible?**
  - **Block device or memory device?**

Early Co-design Opportunities

- Execution model and general hardware architecture
  - Based first on expected characteristics of hardware
    - Large amounts of parallelism
    - Non-coherent memory regions on-node
  - Does this include a hardware abstraction?

- Resiliency
  - What is the base hardware resiliency
  - What does software need to add
  - Need more frequency health monitoring (how often?)

- I/O requirements
  - Traditionally based largely on MTTI
  - Dramatically changes the goals of the I/O team
  - Includes all subgroups

Co-design Opportunities

- Co-location of related but separate applications
  - Using extra threads to co-locate viz, I/O, system software, etc.
  - We can talk about removing interrupts, moving to waiting threads

- Viz architecture
  - Same machine versus tightly couple separate machine
  - Same machine preferred
  - At a minimum no data movement