Software Breakout Group 1-2
Intranode Programming Model

Kathy Yelick, Mike Heroux, Barbara Chapman, Yutaka Ishikawa, Mateo Valero, Jesus Labarta, Luc Giraux, John Shalf, Thomas Sterling, Jack Dongarra, Pete Beckman
And ??
Agreement

• Thousands of functional units on a socket
  – 100s of cores with 10s of FPUs per core (data parallel hardware)
  – OR, 1000s of cores
  – OR, Lightweight PIM cores; heavy weight cores with data parallel instructions and temporal locality needs

• DRAM Memory per FPU capacity will drop
  – Memory per socket will go up, but will not keep up with FPU growth

• “Core” is not well-defined
  – PC / FPU >> 1 → multithreaded architecture
  – PC / FPU << 1 → vector/SIMD architecture

• Cache coherence across a 2020 socket is not feasible
Agreement

• Memory bandwidth “wall” will be solved
  – Yes, technology will exist and market forces will cause this to happen (even outside HPC)
    • Photonics, 3D stacking,…

• Memory latency problems will be solved by hardware and algorithms
  – Need massive concurrency, lightweight threading combined vectors
  – 1B way concurrency including threads, vectors, and/or prefetching
Programming Model Requirements

• Sustainability (13)
  – Backward compatibility through interoperability
  – Incremental adoption support through interoperability
  – Composable
  – Portable and standardized
  – Develop in collaboration with industry
  – Familiarity with old models
Programming Model Requirements

• Address load imbalance (from the hardware, faults, and the applications/algorithms) (12)
  – Dynamic concurrency generation (Do you need it and can you afford it?)
  – Express more parallelism than you need (e.g., hierarchical task graph) \( \times 2 \)
  – Dynamic runtime
  – Less synchronous than fork/join or SPMD
Programming Model Requirements

• Alleviate bandwidth and latency problems (9)
  – Ability to specify data scope and locality
  – Support for scratch pad memory

• Performance transparency and feedback (6)
  – Including runtime adaptation
Programming Model Requirements

- Multiple precision support: ½, full, extended (3)
- I/O (2)
- Fault detection and ability to respond (1)
  - E.g., transient memory errors
- 1B way concurrency (1)
- Global address space (1)
- Message-driven computation (1)
Programming Model Requirements

• Lightweight synchronization objects
• User-defined distributed data structures
• Energy management support (queries)
• Overlapping communication and computation
Issues for a Programming Model Program

• Some of these problems will be solved by industry
• Support for multiple hardware solutions
• If we did nothing, where would we get (MPI + CUDA?)
• If we did a lot, are there things we could not solve?
Why is 2009 unlike 1999?

• Transition from Gigascale (vectors) to Terascale (clusters) required software revolution
• The transition from Terascale to Petascale did not
• The transition from Petascale to Eascale will require another software revolution
  – No clock speed scaling; all performance from concurrency
  – Energy bill for large machine (for 5 years) \( \sim \) hardware cost
  – Linpack is now a reliability benchmark
  – Even homogeneous machines appear heterogeneous
• Separate
  1) The on-chip concurrency problems
  2) Between chip availability problems
  3) Other software model and ecosystem problems:
     multi-physics/scale/institutions/languages and sustainability