ExMatEx All Hands
9/23/14

Cray Design Forward Program
Chapel

Greg Titus

Safe Harbor

This presentation may contain forward-looking statements that are based on our current expectations. Forward looking statements may include statements about our financial guidance and expected operating results, our opportunities and future potential, our product development and new product introduction plans, our ability to expand and penetrate our addressable markets and other statements that are not historical facts. These statements are only predictions and actual results may materially vary from those projected. Please refer to Cray's documents filed with the SEC from time to time concerning factors that could affect the Company and these forward-looking statements.

Copyright 2014 Cray Inc.

Cray Design Forward Goals

- **Research and Design features for high performance networks**
  - Standard APIs for communication
  - Standard APIs for network management
  - Standard wire protocols
- **Co-Design with E7 Labs and Industry**

Copyright 2014 Cray Inc.

Focus Area – Communication API

A key research element of the project is to determine the structures required in the host NIC interfaces in order to meet the performance targets, especially those for message passing and remote memory access rates. Key questions that may drive research in this area may include:

- Which functions must be offloaded to the NIC to maintain performance?
- What structures are required to achieve scalability of a diverse range of traffic patterns and how can they be provisioned so as not to impact performance of the common cases?
- What functionality should be provided by the NIC in order to facilitate efficient execution of the network API by the core?
- What functionality should be in a future core in order to facilitate efficient wakeup on the arrival of new data?

Copyright 2014 Cray Inc.

Focus Area – Management API

Another element of the project is to determine the proper structure and APIs for managing scalable high performance networks. Cray believes that out-of-band monitoring and control is fundamental at scale, in order to meet the requirements of functionality, reliability, and quality of service. Key questions that may drive research in this area may include:

- What are the important management functions to provision?
- What structure of system management best serves those functions?
- What APIs can be standardized to allow common management across a variety of high performance networks?

Copyright 2014 Cray Inc.

DOE/NNSA Co-design activities

- **Telecons with all three ASCR plus NNSA Co-design centers**
  - Bearing on communication and network management APIs
- **Network API deep dives previously held:**
  - ExaCT (coupled applications), CESAR (scaling issues and coupled applications), ExMatEx (coupled applications), NNSA
- **Simulation strategy**
  - Discussions with Sandia on use of SST
- **Support for advanced runtimes**
  - Spoke with Charm++ and Legion teams

Copyright 2014 Cray Inc.

Some Highlights from Co-Design Centers

- **CESAR concerns about strong scaling requirements**
  - Decreases amount of computation per communication operation and so makes communication finer-grain
- **CESAR multigrid**
  - Desire parallel prefix operations in the network
- **Great interest in coupled applications and in-situ analysis**
  - Significant differences in desired techniques for communication
    - Loose coupling – possibly MPI (CESAR) or other comms (ExMatEx)
    - Tight coupling (e.g multi-physics) – shared memory
    - Dynamic coupling (changes every few seconds)
  - ExaCT coupling via dataspaces.org
  - All interested in stable common APIs
- **Good set of miniApps available**

Copyright 2014 Cray Inc.

What is Chapel?

- **An emerging parallel programming language**
  - Design and development led by Cray Inc.
    - in collaboration with academia, labs, industry; domestically & internationally
- **A work-in-progress**
- **Goal: Improve productivity of parallel programming**
- **Open source (Apache license) at GitHub**
- **Portable design and implementation, targeting:**
  - Laptops to HPC, where data and execution locality matters
  - HPC systems from Cray and other vendors
- **This year: performance, manycore, heterogeneous nodes**

Copyright 2014 Cray Inc.

Chapel LULESH Co-design History

Apr 2011: LLNL expresses interest in Chapel at Salishan
- Introduced us to the LULESH benchmark

Summer 2011: Cray intern ports LULESH to Chapel
- *caveat:* used structured mesh to represent data arrays

Nov 2011: Chapel team tunes LULESH for single-node performance

Dec 2011: Chapel team visits LLNL (talk, tutorial, 1-on-1 sessions)

Mar 2012: Jeff Keasler (LLNL) visits Cray to pair-program
- in one afternoon, converted from structured to unstructured mesh
- impact on code minimal (mostly in declarations)

Apr 2012: LLNL reports on collaboration at Salishan

Apr 2012: Chapel 1.5.0 release includes LULESH as an example code

Sep-Nov 2012: performance tuning

Nov 2012: SC12
- Chapel HPC Challenge entry
- LLNL talk at the Chapel Lightning Talks BoF
- Cray talk at Proxy Apps BoF

Dec 2012: Multi-institution LULESH paper accepted to IPDPS '13

ExMatEx Co-design Summer School

- **Students studied new programming models in the context of a molecular dynamics code.**
  - Chapel was one of about 5 other models
- **Chapel team member gave seminar on Chapel at LNLL.**
  - Followed by further discussions on Chapel with the students afterward
- **Editorial note: everybody had a lot of fun!**

Copyright 2014 Cray Inc.

Chapel: the Next Five Years

- **Harden to production-grade**
  - Complete/improve lacking features
  - Optimize performance
- **Target more complex/modern compute node types**
  - Hierarchy and heterogeneity in memories and processors
  - e.g., Intel MIC, CPU+GPU, AMD APU, …
- **Continue to grow the user and developer communities**
  - Including nontraditional circles: desktop parallelism, “big data”
  - Transition Chapel from Cray-managed to community-governed

Copyright 2014 Cray Inc.

Legal Disclaimer

Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.

Cray Inc. may make changes to specifications and product descriptions at any time, without notice.

All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.

Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, URIKA, and YARCDATA. The following are trademarks of Cray Inc.: ACE, APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.

Copyright 2014 Cray Inc.