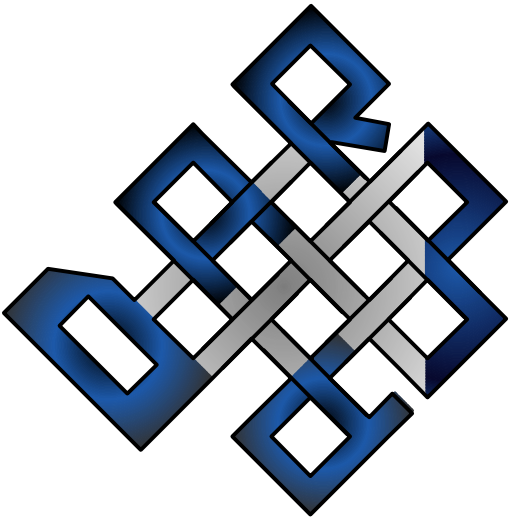
SAND2016-5212PE



DARMA: Distributed Asynchronous Resilient Models and Applications

Janine C. Bennett, Jeremiah J. Wilke, David S. Hollman, Nicole Slattengren, Hemanth Kolla, Francesco Rizzi, Keita Teranishi, Robert L. Clay

Sandia National Laboratories

Exceptional service in the national interest

CIS External Review

Sandia National Laboratories

June 7, 2016

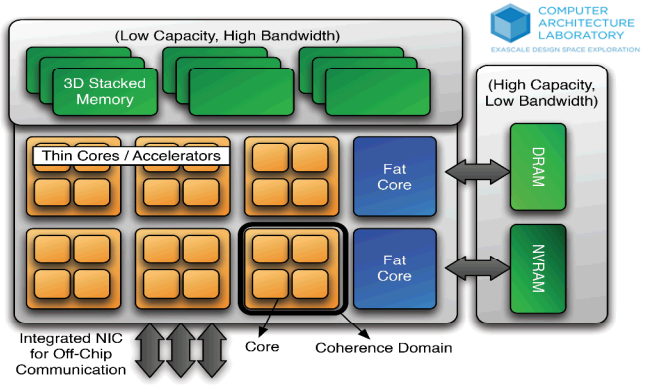
U.S. DEPARTMENT OF ENERGY | NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2011-XXXXP

Extreme-scale high performance computing (HPC) architectures introduce programming challenges

Sandia National Laboratories

System Change	Programming Challenge
Increased node-level parallelism	Expressing/managing node-level & hybrid parallelism
Diverse target architectures	Performance portability across systems
Decreased system reliability	Resilience/Fault mitigation
Increased system noise	Increased need for effective load-balancing strategies
Deeper memory hierarchies	Management of memory hierarchies/locality
Increased system scale	Increased workflow complexity



HPC's role in stockpile stewardship shapes Sandia's NNSA/ASC programming model investments

Sandia National Laboratories

Project			
Kokkos v1/ Kokkos v2	FY07-FY14/ FY10-FY16	O(450) K/year/ O(1)M/year	ATDM (CSSE, LDRD, ASCR)
C++ many-core performance portability library providing data structure and parallel execution abstractions			

Programming Challenge
Expressing/managing node-level & hybrid parallelism
Performance portability across systems
Resilience/Fault tolerance
Increased need for effective load-balancing strategies
Management of memory hierarchies/locality
Increased workflow complexity

Acronyms

National Nuclear Security Administration (NNSA) ▯ Advanced Simulation and Computing (ASC) ▯ Advanced Technology Development & Mitigation (ATDM) Computational Systems & Software Environment (CSSE) ▯ Lab Directed Research and Development (LDRD) ▯ Advanced Scientific Computing Research (ASCR)

HPC's role in stockpile stewardship shapes Sandia's NNSA/ASC programming model investments

Sandia National Laboratories

Project			
Kokkos v1/ Kokkos v2	FY07-FY14/ FY10-FY16	O(450) K/year/ O(1)M/year	ATDM (CSSE, LDRD, ASCR)
C++ many-core performance portability library providing data structure and parallel execution abstractions			
AMT Resilience /DHARMA	FY12-FY14	O(200-400) K/year	CSRF/CSSE
Structural simulation-based resilience research focused on asynchronous many-task runtimes			

Programming Challenge
Expressing/managing node-level & hybrid parallelism
Performance portability across systems
Resilience/Fault tolerance
Increased need for effective load-balancing strategies
Management of memory hierarchies/locality
Increased workflow complexity

Acronyms

National Nuclear Security Administration (NNSA) ▯ Advanced Simulation and Computing (ASC) ▯ Advanced Technology Development & Mitigation (ATDM) Computational Systems & Software Environment (CSSE) ▯ Lab Directed Research and Development (LDRD) ▯ Advanced Scientific Computing Research (ASCR)

HPC's role in stockpile stewardship shapes Sandia's NNSA/ASC programming model investments

Sandia National Laboratories

Project			
Kokkos v1/ Kokkos v2	FY07-FY14/ FY10-FY16	O(450) K/year/ O(1)M/year	ATDM (CSSE, LDRD, ASCR)
C++ many-core performance portability library providing data structure and parallel execution abstractions			
AMT Resilience /DHARMA	FY12-FY14	O(200-400) K/year	CSRF/CSSE
Structural simulation-based resilience research focused on asynchronous many-task runtimes			
DARMA	FY15-FY16	1.6 M/year	ATDM
C++ embedded domain specific language (EDSL) and specification for expressing deferred, asynchronous work			

Programming Challenge
Expressing/managing node-level & hybrid parallelism
Performance portability across systems
Resilience/Fault tolerance
Increased need for effective load-balancing strategies
Management of memory hierarchies/locality
Increased workflow complexity

Acronyms

National Nuclear Security Administration (NNSA) ▯ Advanced Simulation and Computing (ASC) ▯ Advanced Technology Development & Mitigation (ATDM) Computational Systems & Software Environment (CSSE) ▯ Lab Directed Research and Development (LDRD) ▯ Advanced Scientific Computing Research (ASCR)

FY15 shift from CSSE to ATDM triggered significant changes in DARMA team's research focus

Sandia National Laboratories

- No longer resilience-centric
- AMT models remain a focus
  - Data-flow task-graph encodes information that supports dynamic runtime optimizations and mitigation of programming challenges
  - Complementary to (and interoperable with) Kokkos compile-time performance portability library

Programming Challenge
Expressing/managing node-level & hybrid parallelism
Performance portability across systems
Resilience/Fault tolerance
Increased need for effective load-balancing strategies
Management of memory hierarchies/locality
Increased workflow complexity

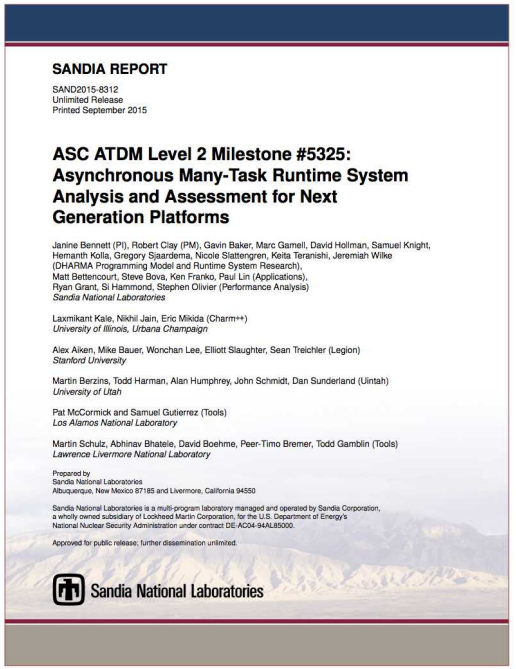
Acronyms

National Nuclear Security Administration (NNSA) ⍰ Advanced Simulation and Computing (ASC) ⍰ Advanced Technology Development & Mitigation (ATDM) Computational Systems & Software Environment (CSSE) ⍰ Lab Directed Research and Development (LDRD) ⍰ Advanced Scientific Computing Research (ASCR)

Shift in focus motivated FY15 L2 milestone: Assess leading AMT runtimes to inform ATDM's technical roadmap

Sandia National Laboratories

- Broad survey of existing AMT runtime systems
- Deep dive on Charm++, Legion, Uintah
  - Assessed *programmability, performance, mutability*
  - \+ Empirical studies highlighted performance potential
  - \- Not yet production-ready for ASC needs
    - Requirements gaps
    - Deficiencies in existing application programming interfaces (APIs)
- Study highlighted need for AMT best practices



SANDIA REPORT

SAND2015-8312
Unlimited Release
Printed September 2015

ASC ATDM Level 2 Milestone #5325: Asynchronous Many-Task Runtime System Analysis and Assessment for Next Generation Platforms

Janine Bennett (PI), Robert Clay (PM), Gavin Baker, Marc Gamell, David Hollman, Samuel Knight, Hemanth Kolla, Gregory Sjaardema, Nicole Slattengren, Keita Teranishi, Jeremiah Wilke (DHARMA Programming Model and Runtime System Research), Matt Bettencourt, Steve Bova, Ken Franko, Paul Lin (Applications), Ryan Grant, Si Hammond, Stephen Olivier (Performance Analysis)
Sandia National Laboratories

Laxmikant Kale, Nikhil Jain, Eric Mikida (Charm++)
University of Illinois, Urbana Champaign

Alex Aiken, Mike Bauer, Wonchan Lee, Elliott Slaughter, Sean Treichler (Legion)
Stanford University

Martin Berzins, Todd Harman, Alan Humphrey, John Schmidt, Dan Sunderland (Uintah)
University of Utah

Pat McCormick and Samuel Gutierrez (Tools)
Los Alamos National Laboratory

Martin Schulz, Abhinav Bhatele, David Boehme, Peer-Timo Bremer, Todd Gamblin (Tools)
Lawrence Livermore National Laboratory

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

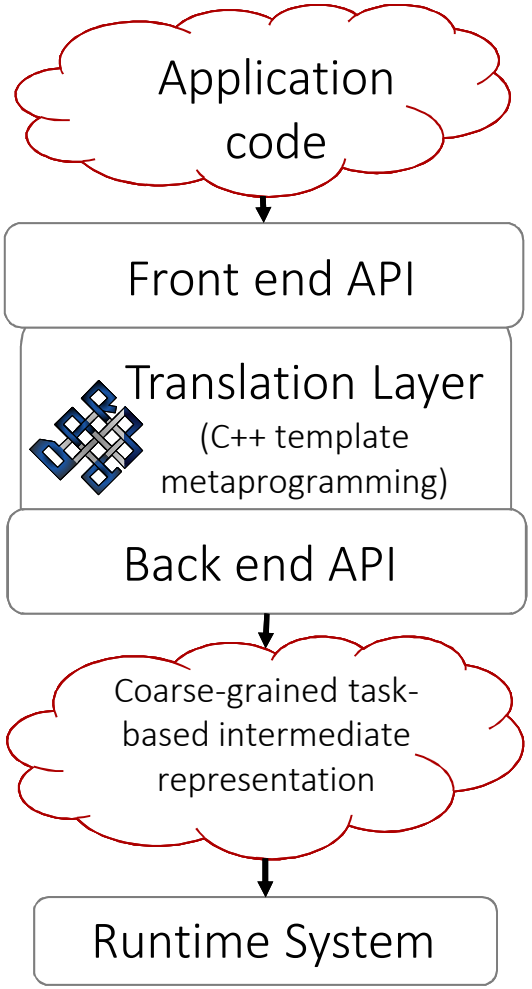
Approved for public release; further dissemination unlimited.

Sandia National Laboratories

DARMA: a C++ embedded domain specific language (DSL) for the expression of deferred, asynchronous work

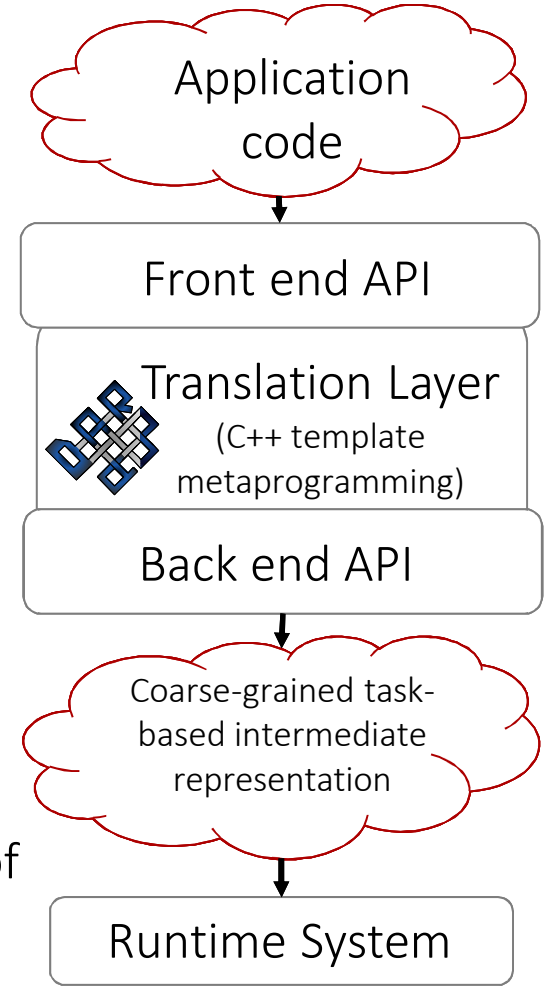
Sandia National Laboratories

- A vehicle for *community-based* resolution to issues raised in the FY15 study
- Insulate applications from runtime system idiosyncrasies
- Improve AMT runtime programmability
  - Co-develop front end API
  - Application and runtime system teams
- Synthesize application co-design activities into runtime system requirements
- Facilitate AMT design space exploration, accelerating development of AMT best practices



DARMA's translation layer maps simple application-level semantics into backend runtime system calls

- Back end API calls implicitly encode data-task dependencies for runtime system
- Dependency information enables runtime to make dynamic decisions
  - Data staging
  - Work scheduling
- Relatively execution model agnostic
  - Runtimes must:
    - Support efficient SPMD (single program multiple data) launch
    - Preserve data-flow dependencies
  - Supports runtime system buildout with a variety of existing AMT runtime system technologies



C++ embedded task annotations express deferred work while maintaining sequential semantics

Sandia National Laboratories

- Work executes asynchronously after all dependencies are met
- DARMA derives dependencies implicitly using **AccessHandle**s
- C++ features enable preservation of sequential semantics
  - Lambdas, reference counted pointers
- Deferred work can migrate after launch when beneficial

```
// .. CODE ..
auto my_handle<int> =
              initial_access("a_key");
create_work([=] {
  // dependencies are derived
  // implicitly based off the use of
  // handles
  my_handle.set_value(37);
}

// .. CODE ..
create_work([=] {
  // sequential semantics let you
  // reason about code in sequential
  // order within rank – even with
  // deferred work
  my_handle.set_value(48);
}

// .. CODE ..
```

DARMA maintains the notion of a **rank**

- Convenience mechanism for initial problem launch and distribution
- Enables locality preservation and performance for SPMD codes
- Maintains MPI-like semantics for ease of integration with existing applications

```
int darma_main(int argc, char** argv)
{
  darma_init(argc, argv);
  const int me = darma_spmd_rank();
  const int size = darma_spmd_size();

  // .. CODE ..

  darma_finalize();
}
```

Coordination semantics are used to communicate between **rank**s

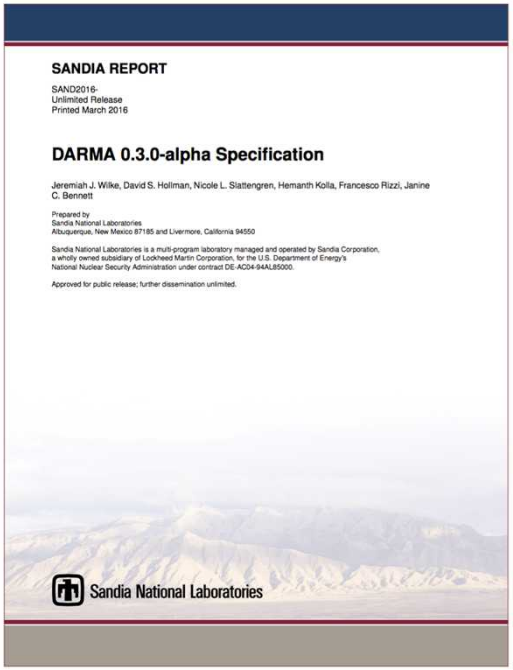
Sandia National Laboratories

- Key-value store abstraction and coordination semantics
  - publish/fetch data using **key**
  - No direct address, otherwise analogous to send/recv
- Promotes deferred execution, task migration, resilience strategies
- Enables data-driven collectives to be expressed

```
int darma_main(int argc, char** argv)
{
  darma_init(argc, argv);
  const int me = darma_spmd_rank();
  const int size = darma_spmd_size();

  // Neighbors rank id
  size_t nbr = (me == 0) ? size-1 : me-1;

  // Create a handle
  auto my_handle = init_access<float>(me);

  // Do some work that sets handle value

  // Publish handle
  my_handle.publish(n_readers=1);

  // Fetch neighbor's handle value
  auto val = read_access<float>(nbr);

  // .. CODE ..
  darma_finalize();
}
```

After only 8 months DARMA is already garnering active community engagement

- Formal specification facilitates collaborations
- Initial specification was co-developed with ATDM application partners
- Features are being added incrementally for agile community development feedback cycles
- Specification feedback process is underway with external runtime and application partners



http://darma.sandia.gov

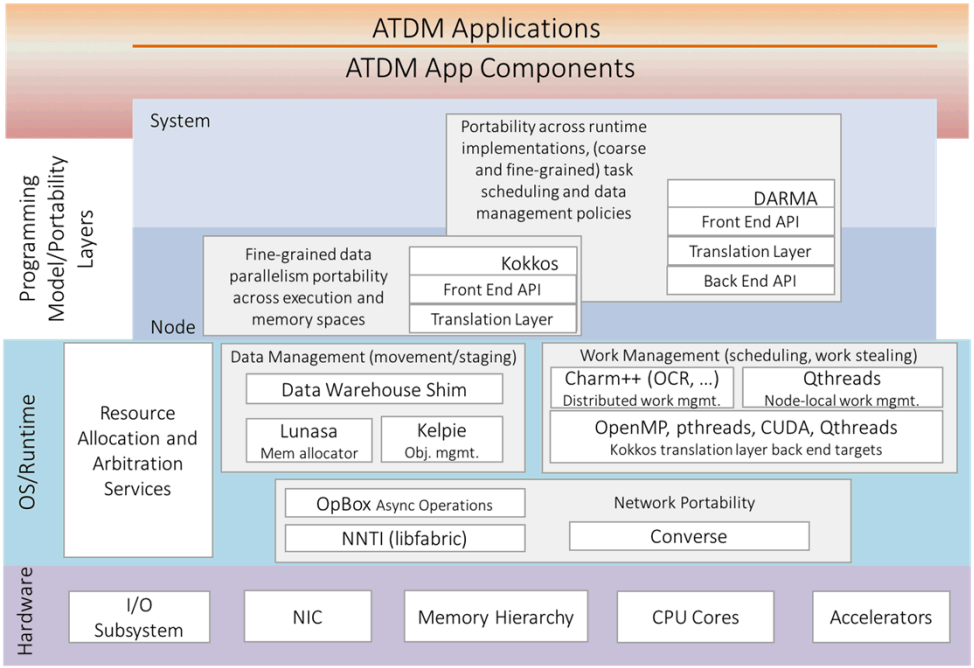
Multiple DARMA-compliant runtime activities are underway leveraging existing AMT runtime technologies

Sandia National Laboratories

Sandia-led efforts

Reference node-level implementation

ATDM computer science component teams are leveraging Charm++



Active university-led efforts

HPX-5

Habanero-UPC++

OCR

Shortlist of others we would like to engage

Legion

Uintah

CHAI+RAJA

We have aggressive timelines for feature specification and AMT runtime design exploration activities

Sandia National Laboratories

Planned specification features

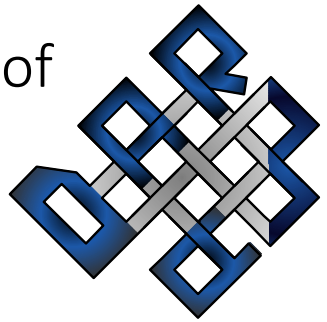
- Collectives (6/16)
- Interoperability with MPI (9/16)
- Expressive abstract machine model (9/16)
- Fine-grained deferred parallel patterns (9/16)
- Data subsetting/slicing interface (9/16)
- Programmer-directed optimization/load balancing hooks (12/16)

Runtime buildout activities

- Sandia ATDM/Charm++ initial implementation by 9/16
- ASC/ATDM Level 2 milestone in 9/17
- Continued outreach with existing runtime teams
  - Full commitment to Exascale Computing Project research efforts

Takeaway Messages

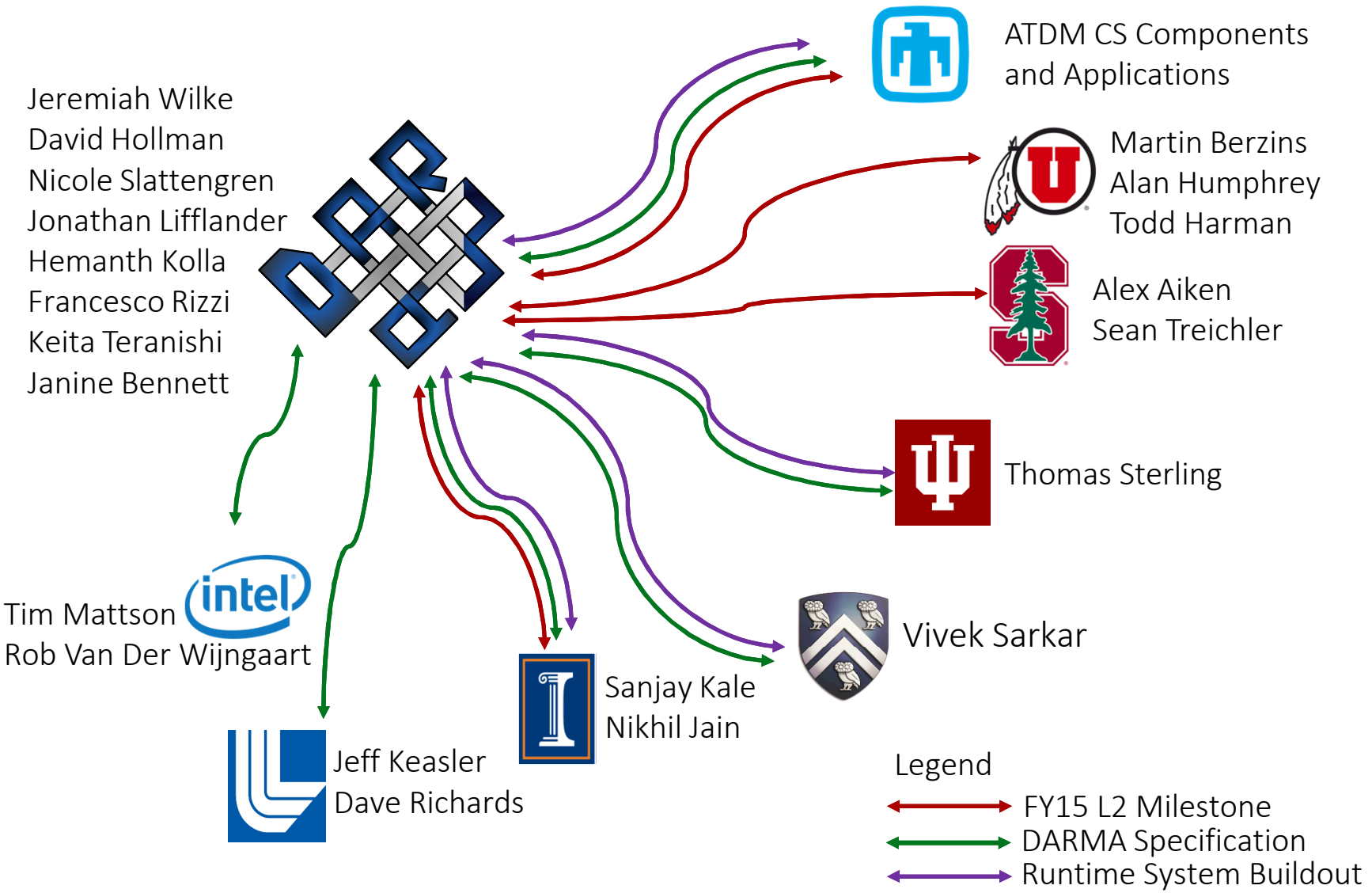
- Sandia is taking a holistic approach to mitigating programming challenges for next generation platforms
  - Complementary run-/compile-time, node-/system-level research
- In a short timeframe Sandia has re-focused our run-time/ system-level research efforts:
  - Developing infrastructure and tools to support community-based resolution to address requirements gaps and deficiencies
- Already seeing community impact through breadth of collaborations underway



Sandia National Laboratories

BACKUP SLIDES

DARMA Team and Interactions



Publications/Presentations/Outreach Activities

Co-organized Community Events:

- ISC16 Workshop on Building a European/American Community for the Development of Dynamic Runtimes in Extreme-Scale Systems
- HPDAV 16 Panel on Programming Data Analysis and Visualization at the Extreme-Scale
- SC15 Panel Asynchronous Many-Task Programming Models for Next-Generation Platforms
- SC14 BOF Asynchronous Many-Task Programming Models for Next-Generation Platforms

Reports:

- DARMA 0.3.0-alpha Specification, Jeremiah Wilke, David Hollman, Nicole Slattengren, Hemanth Kolla, Francesco Rizzi, Janine Bennett. Sandia Technical Report SAND2016, Sandia National Laboratories, 2016.
- ASC ATDM Level 2 milestone #5325: Asynchronous Many-Task runtime system analysis and assessment for next generation platforms. Janine C. Bennett, Robert Clay, et al. Sandia Technical Report SAND2015-8312, Sandia National Laboratories, 2015.

Presentations:

- The DARMA Approach to Asynchronous Many-Task Programming. Janine C. Bennett, Jeremiah Wilke, et. al. Presented at ECP Review 2016, Sandia National Laboratories, 2016.
- A Comparative Analysis of Asynchronous Many-Task Programming Models for Next Generation Platforms. Janine C. Bennett, Hemanth Kolla, et. al. Presented at SIAM CSE 2015, MS 129 DAT-Based Efficient Scalable and Portable PDE Software, 2015.

Conference Publications:

- Enabling Runtime/Application Co-Design through Common Concurrency Concepts. Jeremiah J. Wilke, Janine C. Bennett, Robert Clay. Proceedings of Runtime Systems for Extreme Scale Programming Models and Architectures SC15 Workshop, 2015.
- Lessons Learned from Porting the MiniAero Application to Charm++. David S. Hollman, Janine C. Bennett, et. al. Presented at the 13th Annual Workshop on Charm++ and its Applications, 2015.
- Evolving the Message Passing Programming Model via a Fault-Tolerant, Object-oriented Transport Layer. Jeremiah J. Wilke, Keita Teranishi, et al. Proceedings of the 5th Workshop on Fault Tolerance for HPC at eXtreme Scale, FTXS 2015. pp. 41-46, ACM, 2015.
- Extreme-Scale Viability of Collective Communication for Resilient Task Scheduling and Work Stealing. Jeremiah J. Wilke, Janine C. Bennett, et al. Proceedings of the 4th Workshop on Fault Tolerance for HPC at eXtreme Scale, FTXS 2014. pp. 756-761, 2014.
- Coordination Languages and MPI Perturbation Theory: The FOX Tuple Space Framework for Resilience. Jeremiah J. Wilke. Proceedings of IPDPS Workshops. pp. 1208-1217, 2014.

DARMA design and implementation details

Sandia National Laboratories

- Mixed imperative/declarative style of programming
  - Sequential imperative semantics in large blocks that do not execute immediately
- SPMD is still the dominant form of parallelism
- C++-embedded task annotations allow work to be deferred and performed asynchronously
  - Declarative description of coarse grained chunks that the runtime can schedule and run when dependencies are met
- Task parallelism is achieved through permissions/access qualifiers on data
  - Enables runtime to reason about what can run in parallel and make intelligent staging decisions

The Ontology of DARMA: Axioms/assumptions derived from L2 milestone and co-design activities

Sandia National Laboratories

- SPMD is the dominant parallelism
- There will too much compute (parallelism) available in the hardware for basic data parallelism to fill
- Extra asynchrony should not complicate reasoning about application correctness (intuitive semantics, debugging tools)
- There exist many applications/algorithms with dynamic load balance, dynamic sparsity, or complex workflow coupling whose development would be greatly accelerated by a more productive programming model
- The traditional MPI abstract machine model (uniform compute elements, flat memory spaces) will get further and further away from actual system architecture

Keep simple things simple, keep tractable things tractable, make difficult things tractable

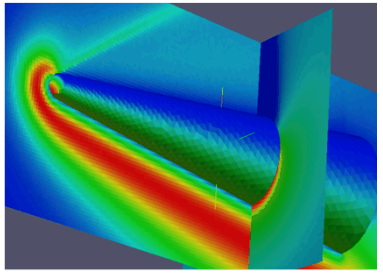
Sandia National Laboratories

- Simple/tractable
  - SPMD launch and initial problem decomposition/distribution
  - Collectives
  - Basic checkpoint/restart fault recovery supported
  - Application-specific data structures/layouts
- Difficult
  - Express/mix all forms of parallelism (data, pipeline, task)
  - Dynamic load balancing, work stealing
  - Data staging (software-managed cache)
  - Performance portability across execution spaces
  - Macro data-flow parallelism (parallelism within a task)

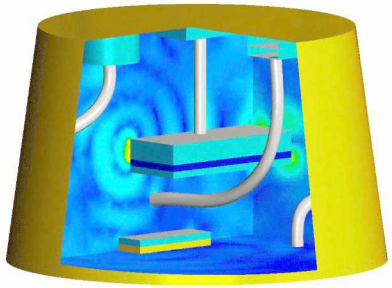
ASC/ATDM subprogram is enabling new decision-support capabilities for NNSA ASC nuclear weapons applications

Sandia National Laboratories

- Two production prototype applications
- New code-base suitable for future ASC applications
  - Component software design
  - Agility, reuse, and reduced cost
- Advanced computational science capabilities
  - Embedded geometry and meshing
  - Embedded UQ, optimization, and analytics for decision-support
  - Flexible, robust and dynamic multiscale & multi-physics



Hypersonic Re-entry



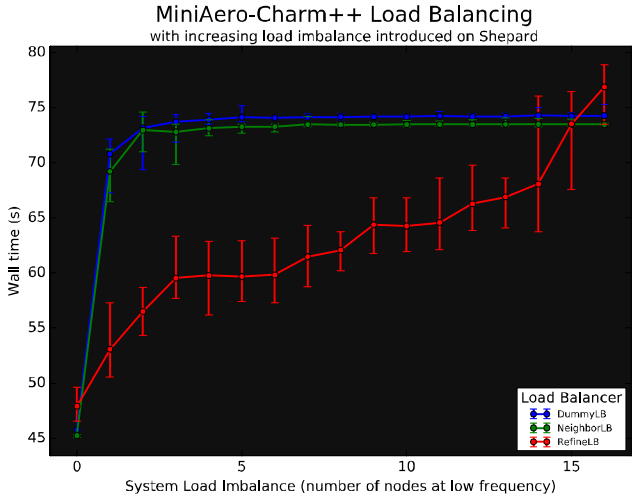
Electro-magnetic plasma in hostile environment

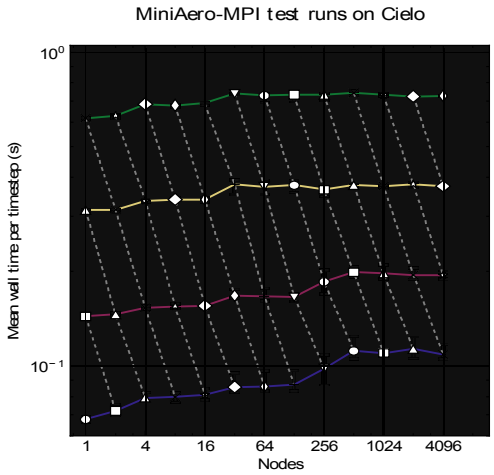
Acronyms

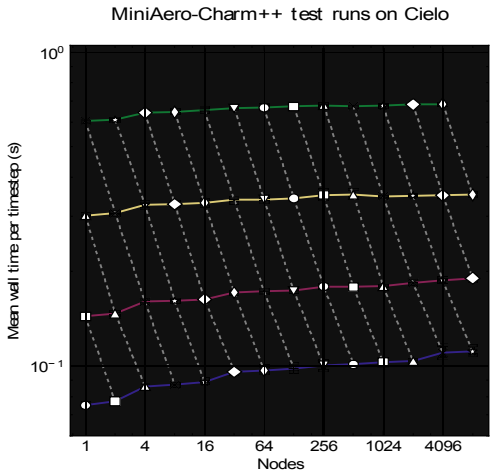
National Nuclear Security Administration (NNSA) | Advanced Simulation and Computing (ASC) | Advanced Technology Development & Mitigation (ATDM)

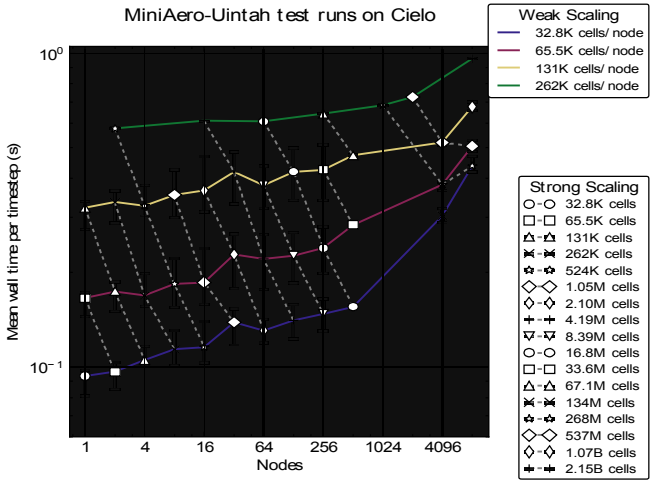
Key performance findings confirm AMT runtime potential for addressing next generation platform challenges

- Empirical studies show an AMT runtime can mitigate performance heterogeneity inherent to the machine itself
- MPI and AMT runtimes perform comparably under balanced conditions
- Summary of previous experiments show strengths of AMT runtimes for dynamic applications





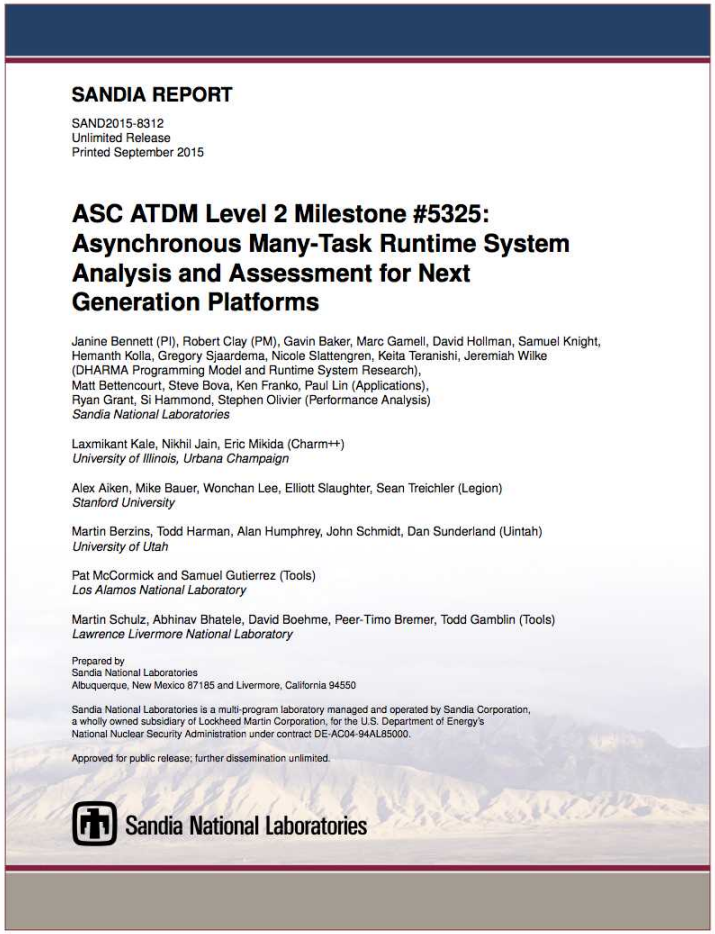




Key programmability and mutability findings show AMT runtimes are not yet ready for production use

- Requirements gaps
- Deficiencies in existing application programming interfaces (APIs)

- Highlights need for development of community best practices
  - Accelerate adoption of runtimes
  - Provide concise application-informed requirements to researchers and vendors providing lower-level components of software stack



SANDIA REPORT

SAND2015-8312
Unlimited Release
Printed September 2015

ASC ATDM Level 2 Milestone #5325: Asynchronous Many-Task Runtime System Analysis and Assessment for Next Generation Platforms

Janine Bennett (PI), Robert Clay (PM), Gavin Baker, Marc Gamell, David Hollman, Samuel Knight, Hemanth Kolla, Gregory Sjaardema, Nicole Slattengren, Keita Teranishi, Jeremiah Wilke (DHARMA Programming Model and Runtime System Research), Matt Bettencourt, Steve Bova, Ken Franko, Paul Lin (Applications), Ryan Grant, Si Hammond, Stephen Olivier (Performance Analysis)
Sandia National Laboratories

Laxmikant Kale, Nikhil Jain, Eric Mikida (Charm++)
University of Illinois, Urbana Champaign

Alex Aiken, Mike Bauer, Wonchan Lee, Elliott Slaughter, Sean Treichler (Legion)
Stanford University

Martin Berzins, Todd Harman, Alan Humphrey, John Schmidt, Dan Sunderland (Uintah)
University of Utah

Pat McCormick and Samuel Gutierrez (Tools)
Los Alamos National Laboratory

Martin Schulz, Abhinav Bhatele, David Boehme, Peer-Timo Bremer, Todd Gamblin (Tools)
Lawrence Livermore National Laboratory

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.

Sandia National Laboratories