Project 38 success requires ModSim

David J. Mountain
Advanced Computing Systems Research Program (ACS)

Laboratory for Physical Sciences, Research Park

davidjmountain@ieee.org

Outline

What the heck is Project 38?

How is Project 38 using ModSim?

Why is ModSim valuable?

Project 38 is a team effort

National Nuclear Security Administration

Argonne NATIONAL LABORATORY

Sandia National Laboratories

BERKELEY LAB

Pacific Northwest NATIONAL LABORATORY

BROOKHAVEN NATIONAL LABORATORY

Los Alamos NATIONAL LABORATORY EST. 1943

OAK RIDGE National Laboratory

U.S. DEPARTMENT OF ENERGY | Office of Science

Lawrence Livermore National Laboratory

IARPA

Background

September 2016 meeting(s) with HPC experts, held to discuss/respond to the June 2016 announcement of China's TaihuLight supercomputer, reached the following consensus (a report [1] is available):

- The HPC technology ecosystem is changing in ways that are less favorable to HPC
- There are national security implications to this change
- Leadership in innovative architectures is critical
- Joint architectural explorations might be useful and interesting

September 2017 technical deep dive plus follow on meetings, VTC, telecons

Improved understanding of key applications
Specialized architecture development process and examples
Possible architectures and applications of interest
Value proposition for collaborating
Refinement of exploration space and exploration process

Joint explorations/collaborations on purpose built architectures have promise

It requires a new approach to exploring and developing HPC systems
Project 38 is an attempt to define this new approach

[1] https://www.nitrd.gov/nitrdgroups/images/b/b4/NSA_DOE_HPC_TechMeetingReport.pdf

Project 38 at a glance

Project 38 is a set of vendor-agnostic architectural explorations involving NSA, the DOE Office of Science, and NNSA (these latter 2 organizations are referred to below as “DOE”). These explorations are expected to accomplish the following:

Near-term goal: Quantify the performance value and identify the potential costs of specific architectural concepts against a limited set of applications of interest to both the DOE and NSA.

Long-term goal: Develop an enduring capability for DOE and NSA to jointly explore architectural innovations and quantify their value.

High level guidelines

Improvements in data access are the initial focus for architectural ideas and applications – primarily on the node

Cost-benefit analysis

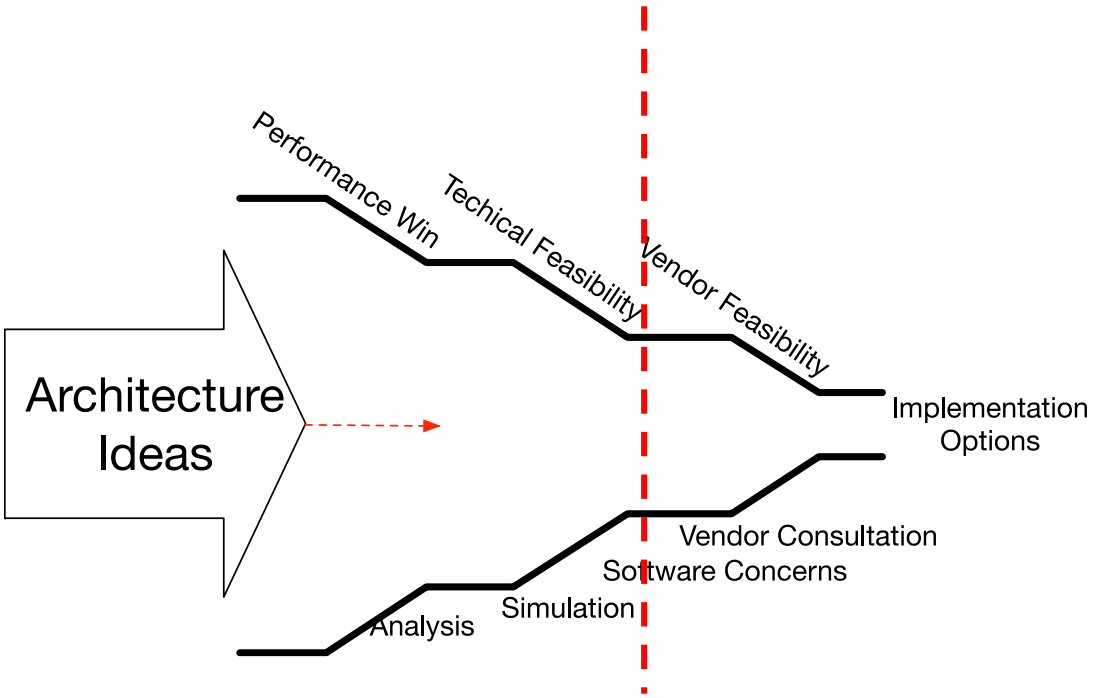
- Baseline comparison is to expected roadmaps/ECP^{+} (business as usual)
- Primarily a performance comparison
- “Cost” is adverse changes to programming models, SW stacks, etc.

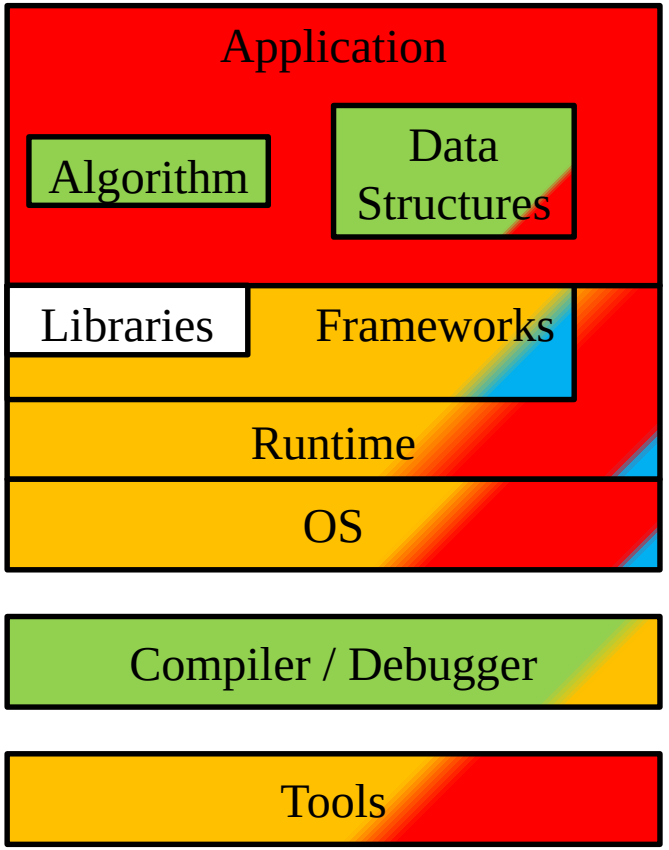
Milestones

2018 Quantify the benefits of the explorations
ModSim is the primary exploration path

2019 Complete the existing explorations -- define the primary SW issues
Document the results

2020 Improve cost-benefit analyses
Push best ideas towards implementation





Application Kernels

1D FFT

HPC Challenge benchmark

Kripke

Kunen et al, “Kripke – A massively parallel transport mini-app”, American Nuclear Society M&C, April 2015.

HPGMG

Adams et al, “HPGMG 1.0: A benchmark for ranking high performance computing systems,” Tech report, hpgmg.org, 2014.

Tensor Contraction Engine

Baumgartner et al, “Synthesis of High-Performance Parallel Programs for a Class of Ab Initio Quantum Chemistry Models,” Proceedings of the IEEE, Feb 2005.

PIC codes, represented by PICSAR

picsar.net

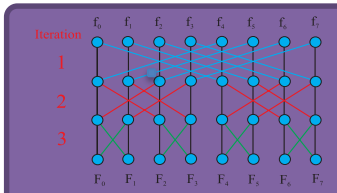
HipMer

Georganas et al, “MerBench: PGAS benchmarks for High Performance Genome Assembly,” PAW17, November 2017.

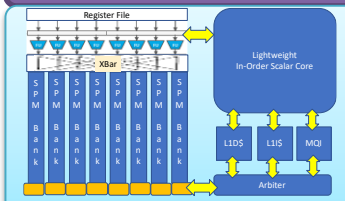
Sparse Matrix Trisolve and other common sparse matrix operations

Architectural ideas

Word Granularity Scratchpad Memory
DRE and DRE (?)
Message Queues
Scatter/Gather
Remote Atomics

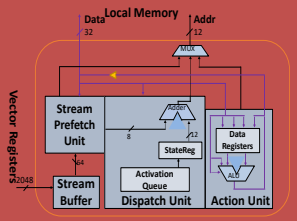


Fixed Function Accelerators



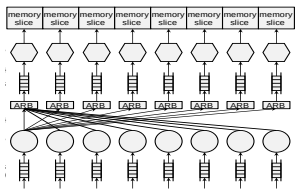
Word Granularity Scratchpad Memory:

Gather-scatter within processor tile
more effective SIMD



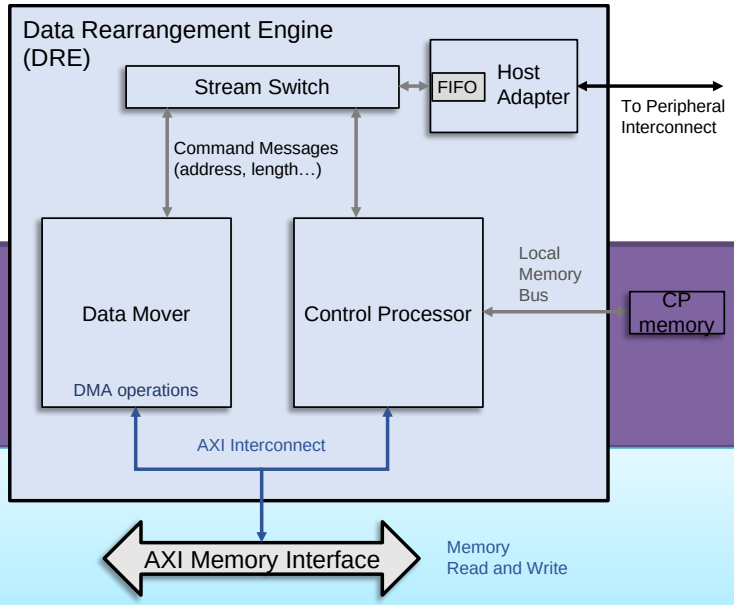
Data Recoding Engine

Sub-word granularity
Handles branch-heavy code (avg. 20x improvement over using processor core)



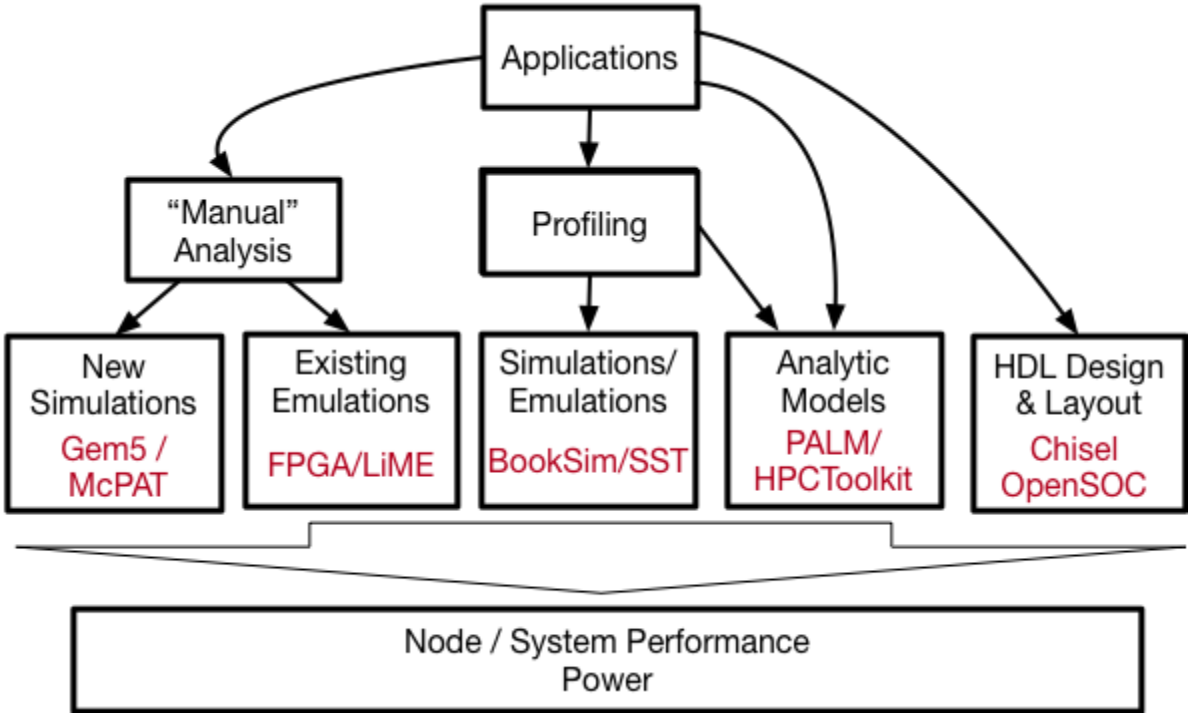
Hardware Message Queues (with atomic queue/dequeue)

Gather-scatter between processor tiles
Async between tiles to eliminate overhead of barriers



ModSim tools used

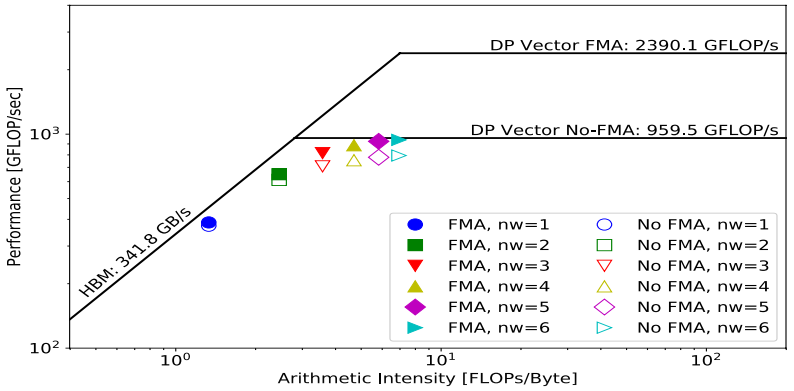
Higher level architectural exploration



More ModSim tools used
lower level design explorations

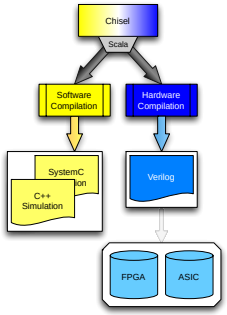
- **Performance Counters…**
  - **Full apps in distributed memory**
  - DRAM, Cache, FPU, IPC Performance Counters w/LIKWID, VTune, NVProf
  - x86, KNL, NVIDIA GPU support
- **Analytic Modeling**
  - Code Analysis: ExaSAT (code opt design space)
  - SRAM Model: Cacti and p-Cacti for sub-22nm
  - Microsoft Excel
- **Roofline Model** (+roofline advisor)
  - x86, KNL, and GPU support
  - Multilevel hierarchy, Multi-bottleneck analysis, stride-k acc… divides, …
- **Simulation & Instrumentation…**
  - **Kernels (trade speed for detail)**
  - Cache Simulator: SDE (Intel Advisor, x86)
  - Core: Chisel "Spike" simulator and Verilator
  - NOC Model: OpenSoC + BookSim for Parameter sweeps
- **Emulation (Chisel HW Generators)**
  - FPGA and synthesis

Empirical GPP Roofline on KNL



Chisel

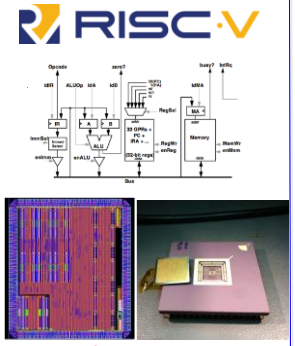
DSL for rapid prototyping of circuits, systems, and arch simulator components



Back-end to synthesize HW with different devices Or new logic families

RISC-V

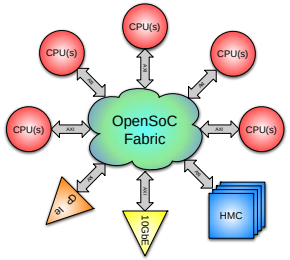
Open Source Extensible ISA/Cores



Re-implement processor With different devices or Extend w/accelerators

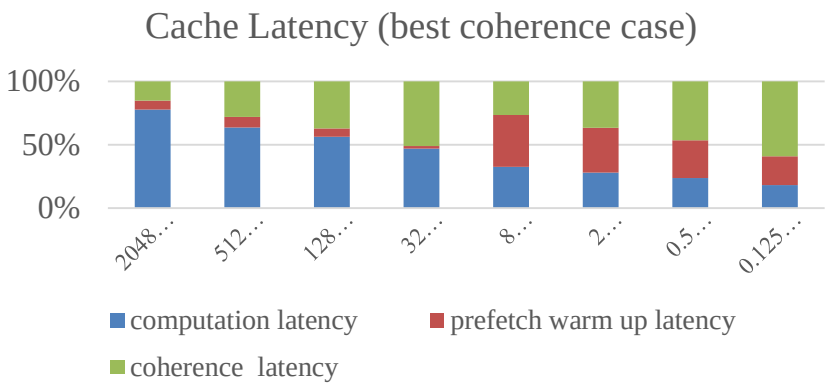
OpenSOC

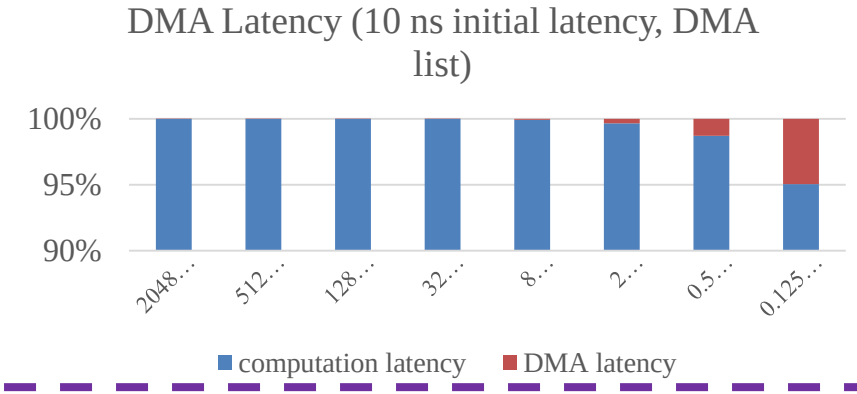
Open Source fabric To integrate accelerators And logic into SOC

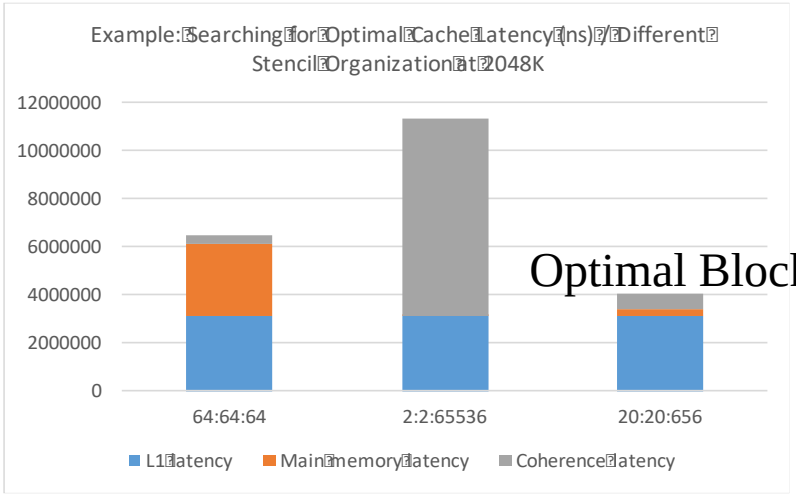


Platform for experimentation Parameterized hardware generation

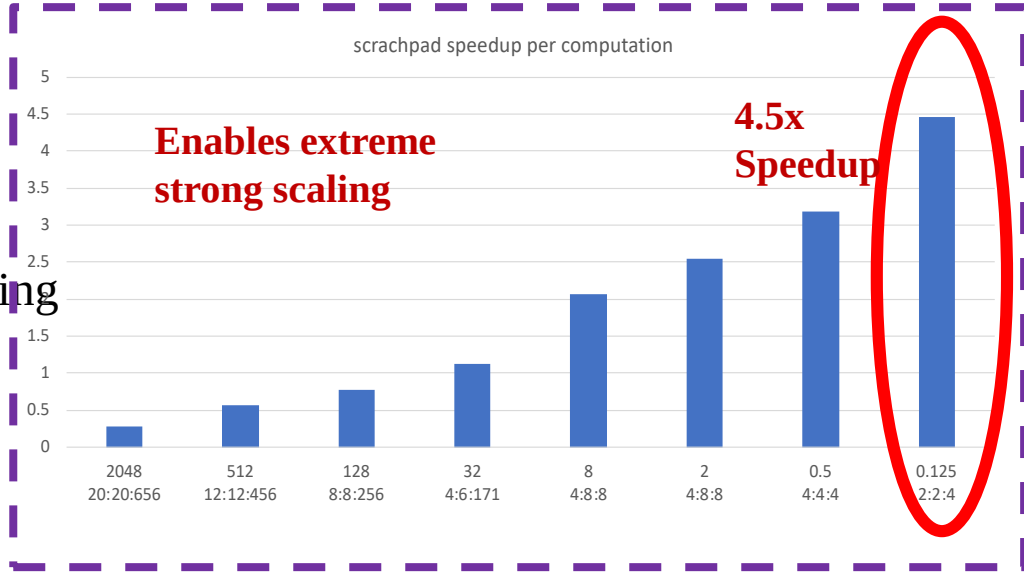
Stencil Study (effect of coherence and word granularity DMA on basic stencil performance)







Optimal Blocking



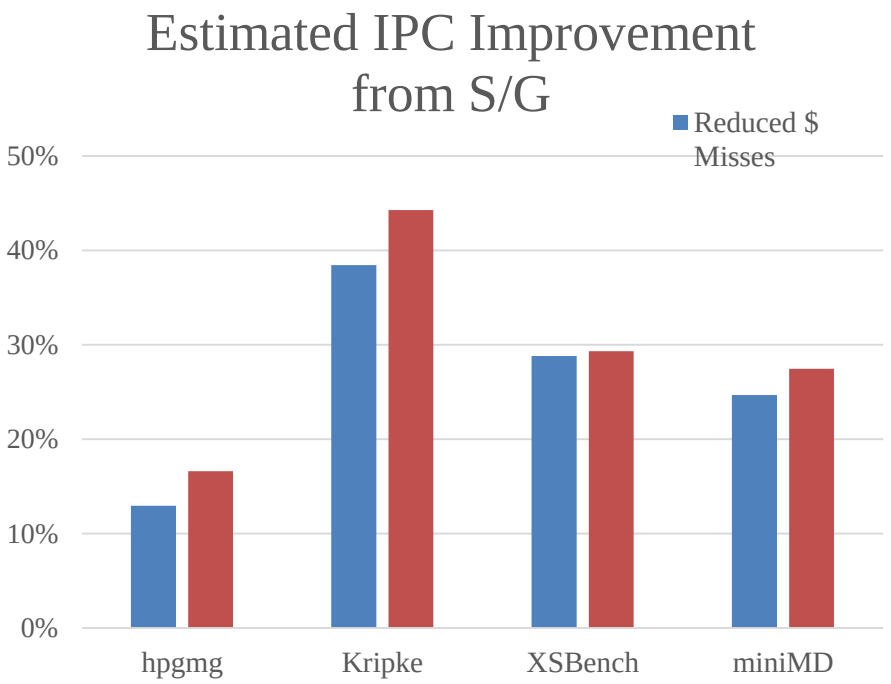
Scatter/Gather results

Simple IPC Calculation
- Lots of caveats

Effect 1: Reduce $ Misses
- Assume covered L1 misses become hits
- 13-40% IPC improvement

Effect 2: Integer offload
- Assume half of address-calc integer operations are offloaded, require 0.1 cycles (vs. 0.4)
- 17-44% IPC Improvement

Effect 3: Improved $ performance
- If covered cache accesses go to scratchpad, other accesses more efficient?



Here's a great idea!

Scatter/Gather

- Could have a substantial positive impact on performance
- Possible to identify regions amenable to lots of in-memory operations, high reuse, good compaction
- Good progress on design (DRE work) – high feasibility

Word-Granularity Scratchpad

- Can reduce post-$ accesses / make better use of cache
- Not useful for all apps (Kripke, XSBench?)

Combine!

Verdict: Combine them!

- Focus on S/G
- Target Arch: S/G to/from scratchpad
- Option: w/ HW Synchronization
- Option: w/ Recoding engine

High level results

Three Architectures

Scatter-Gather: Good body of evidence for performance, programmability, good foundation for design

Word-granularity Scratchpad: Some evidence for improved performance, programmability

Atomics: Chicken-and-Egg, coherency issues, need better application drivers

Benefit	Scatter/Gather	Word-granularity Scratchpad
Kripke	Good >20%	Not Word-Gran
hpgmg	Good >15%	Ongoing
XSBench	Good >28%	Mild?

Additional value of ModSim

Detailed knowledge transfer

- OCCAM

Extension of knowledge

- Classified/unclassified boundaries
- Proprietary/open boundaries

Thanks!

National Nuclear Security Administration

Argonne NATIONAL LABORATORY

Sandia National Laboratories

BERKELEY LAB

BROOKHAVEN NATIONAL LABORATORY

Pacific Northwest NATIONAL LABORATORY

OAK RIDGE National Laboratory

U.S. DEPARTMENT OF ENERGY | Office of Science

Los Alamos NATIONAL LABORATORY EST. 1943

Project 38

Lawrence Livermore National Laboratory

IARPA

Special thanks to Arun Rodrigues and John Shalf