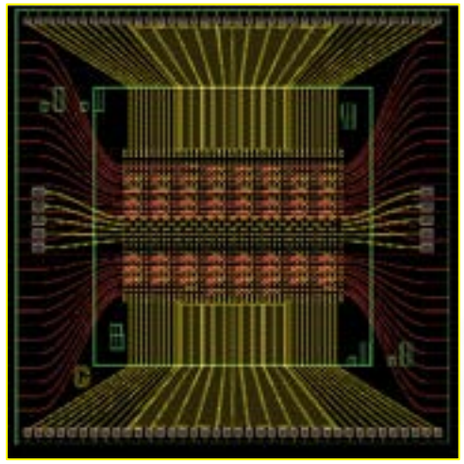
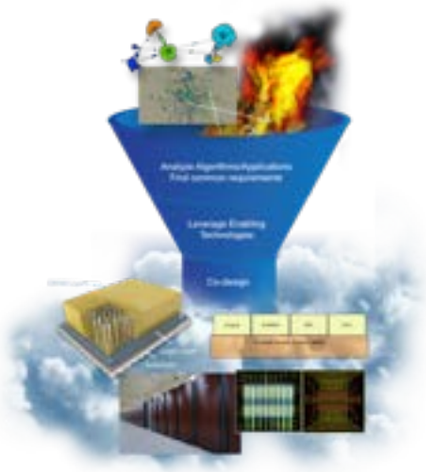
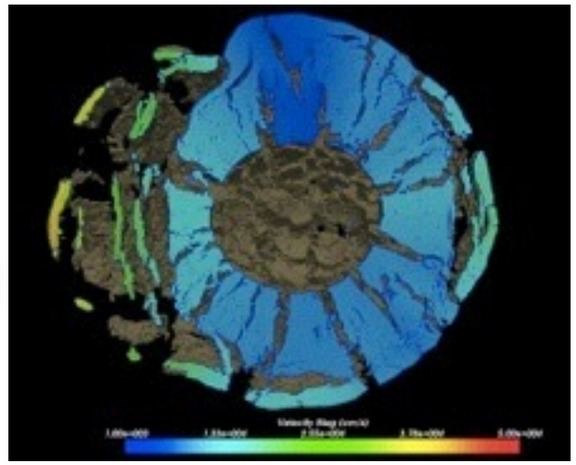
Exceptional service in the national interest

Sandia National Laboratories





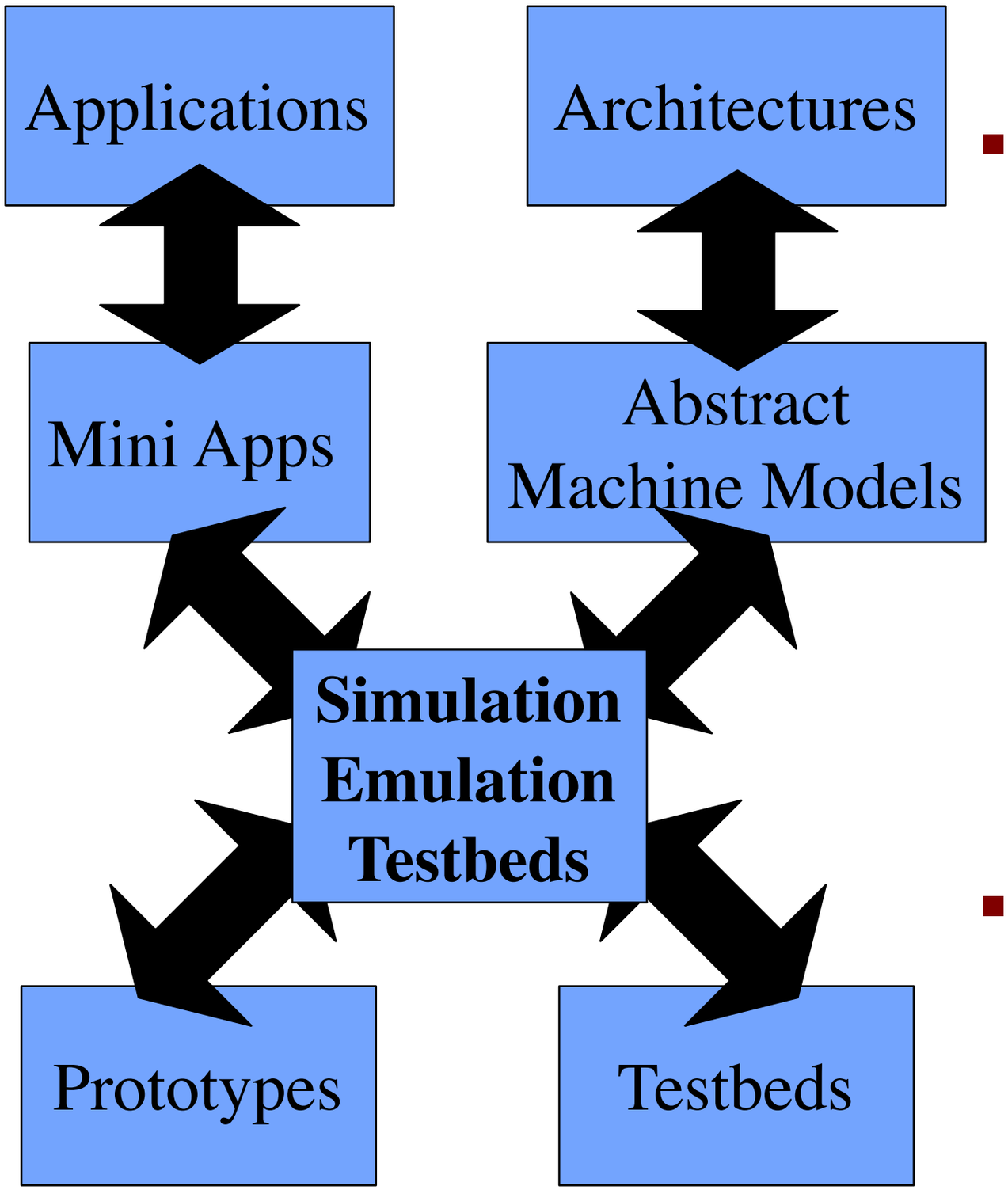


Sandia Advanced Architecture R&D: Simulation & Testbeds

Arun Rodrigues, Dylan Stark,
Simon Hammond, Jim Laros

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

Advanced Architecture R&D



- **Reduce impact on labs' mission in a rapidly changing technology environment.**
  - Significant PRODUCTION code rewrite or modification may be required
  - Ensure that when we make "the change" it is the right move for code longevity, porting efforts, performance etc.
- **CoDesign needs a meeting point between applications and architectures**

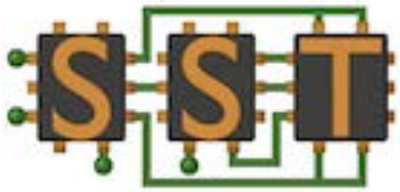
Sandia National Laboratories

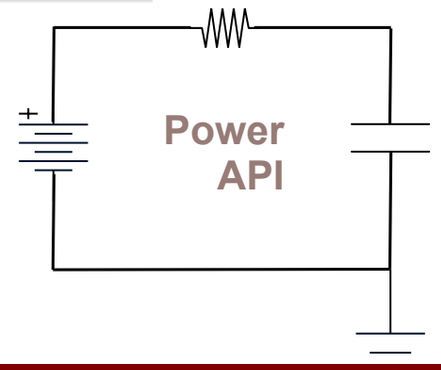
Testbeds

Advanced Architecture Test Beds

Sandia National Laboratories

- Current state of technology REQUIRES exploratory R&D of:
  - Alternative Programming Models
  - Architecture-aware algorithms
  - Advanced memory sub-system development
  - Energy Efficient Hardware, Runtime, Systems Software, AND APPLICATIONS
- Current areas of experimentation include:
  - Mantevo Proxy Application suite
    - http://mantevo.org
  - Validation of SST Architecture Simulator
    - http://SST-simulator.org
  - Portals
    - http://code.google.com/p/portals4
  - Kitten Light Weight Kernel
    - https://software.sandia.gov/trac/kitten
  - Power Research
    - Measurement and Control
      - Sandia Power API
    - Application and runtime power impacts
- ~200 Users





A Closer Look

Hostname	**Compton**	**Curie**	**Shannon**	**Teller**	**Volta**	**Morgan**
CPU	Dual Socket Intel E5-2670 (Sandy Bridge)	Interlagos	Sandy Bridge	AMD A10-5800K (Piledriver) 3.8GHz Quad-core	Intel Xeon E5-2695 V2 dual socket (Ivy Bridge) total 24 cores 2.4 GHz	Intel Xeon E5-2695 V2 dual socket (Ivy Bridge) total 24 cores 2.4 GHz
Accelerator	Pre-Production Intel Xeon Phi Knights Corner (KNC)	Nvidia Kepler K20X	Nvidia Kepler K20X and K40X	Radeon Northern Islands (on die integration)	None	Intel Phi (KNC C0) Nvidia K40X
GPU cores	57 - 1.1GHz	2688	2688	384 @ 800MHz	N/A	57 @ 1.1GHz
Nodes	42	52	32	104	56	4
Interconnect	Mellanox	Gemini	Mellanox	QLogic	Aries	Mellanox QDR IB
Other	80GB SSD per node	Full featured RAS sys	Full PCI Gen 3 NVIDIA GPU Direct	Integrated CPU/GPU+ 256GB SSD/ node	Full featured RAS system including power monitoring and control capabilities	SRN

Use: Mantevo/Mini-App Overview

Sandia National Laboratories

Node Level

Vectorization

Threading

Future Languages

	MPI	Intrins.	CUDA	OpenAcc	OpenCL	Kokkos Array	Cilk+	OpenMP		TBB	ArBB/ CEAN	qthreads	pthreads	MKL/ Math Lib.	Adv. Lang.	DSLs
								L	T							
NVIDIA (GPU)																?
AMD (CPU)																?
AMD (APU)		?														?
Intel (CPU)																?
Intel (MIC)																?
IBM										?						?
ARM																?
miniFE																
miniMD																
miniGhost																
miniAMR																
miniSMAC																
LULESH																
SNAP																
CoMD																
S3D/SMC																
NEK5000																
PathFinder																

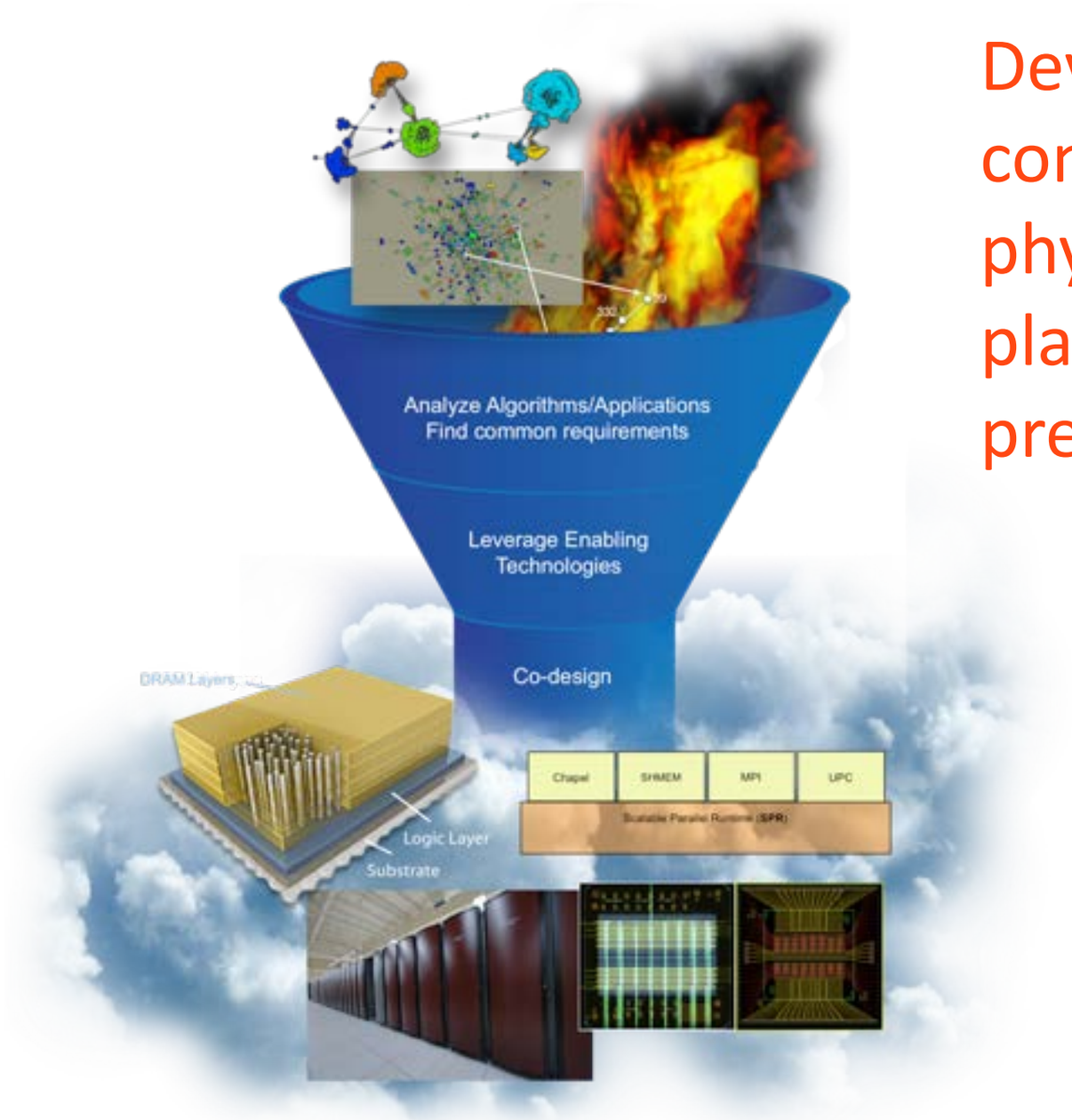
Micron

- Micron Advanced Technology
- 1 HMC Memory Cube + 4 FPGAs
- HMC supplies massive bandwidth (160 GB/s), parallelism
- Requires 4 high-speed links to drive it
- Demonstrated ~10x GUPS performance vs. DDR DIMMs
- Exploring other memory access patterns

Sandia National Laboratories

Simulation
&
eXtreme scale Grand Challenge

XGC Goal:Unified Architecture

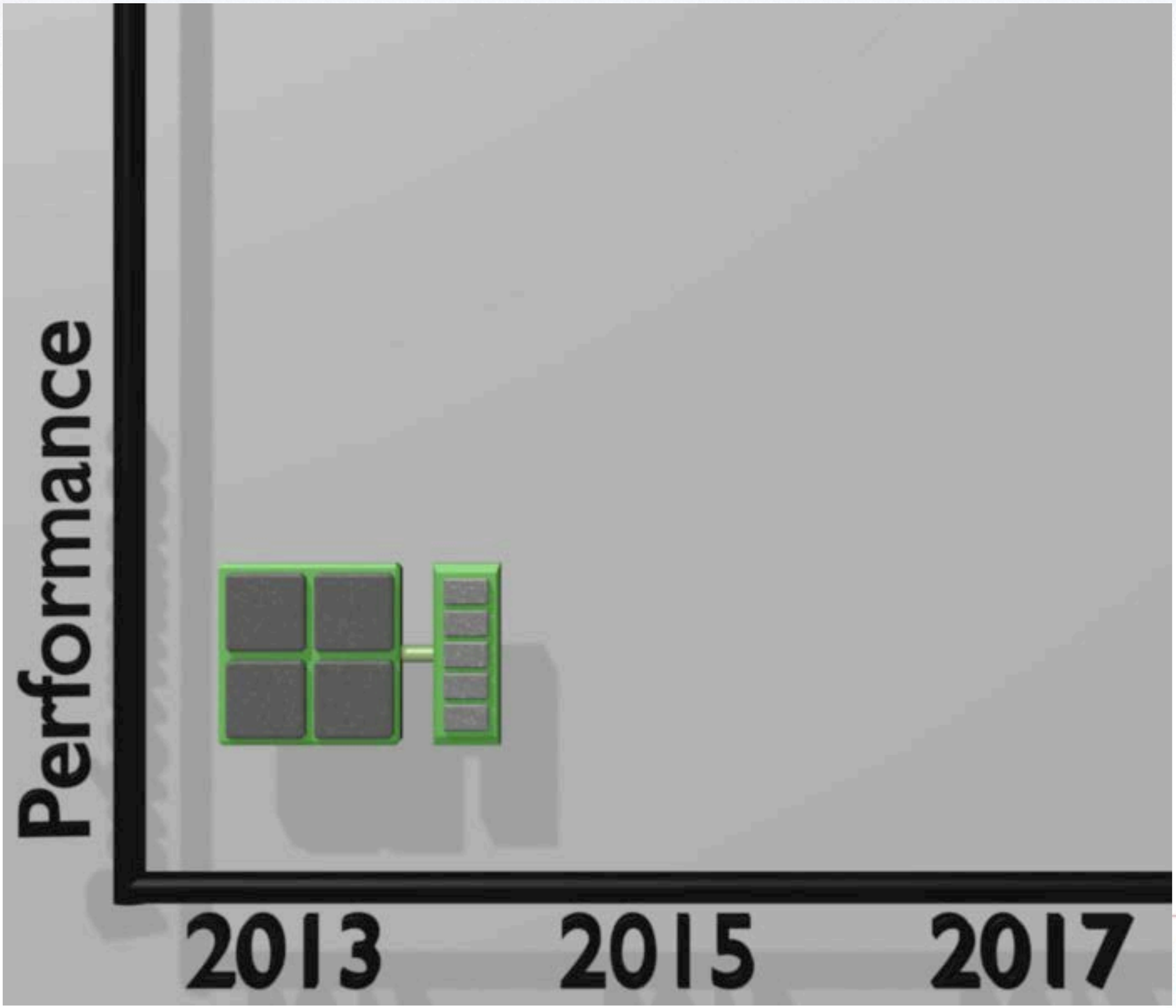


Develop key system architecture concepts that will enable a unified physics/data analytics computing platform or determine what gaps prevent it

Extreme-scale computing will be a key capability in meeting Sandia's national security missions. Understanding where application requirements for our two mission areas overlap will drive development of a set of common components to efficiently support both areas.

Continued focus on:
Data Movement
Concurrency Management

Project Forward; Place a Stake; Explore



NNSA National Nuclear Security Administration

Sandia National Laboratories

SST Simulation Project Overview

Goals

- Become the standard architectural simulation framework for HPC
- Be able to evaluate future systems on DOE/DOD workloads
- Use supercomputers to design supercomputers

Status

- Parallel Core, basic components
- Current Release (3.1)
  - Improved set of components
  - Improved portability

Technical Approach

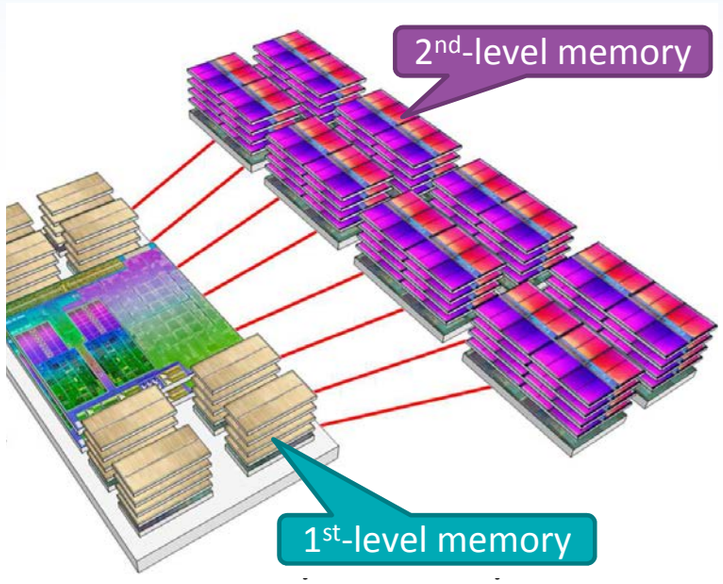
- Parallel
  - Parallel Discrete Event core with conservative optimization over MPI
- Multiscale
  - Detailed and simple models for processor, network, and memory
- Interoperability
  - gem5, DRAMSim, cache models
  - routers, NICs, schedulers, GPGPU
- Open
  - Open Core, non viral, modular

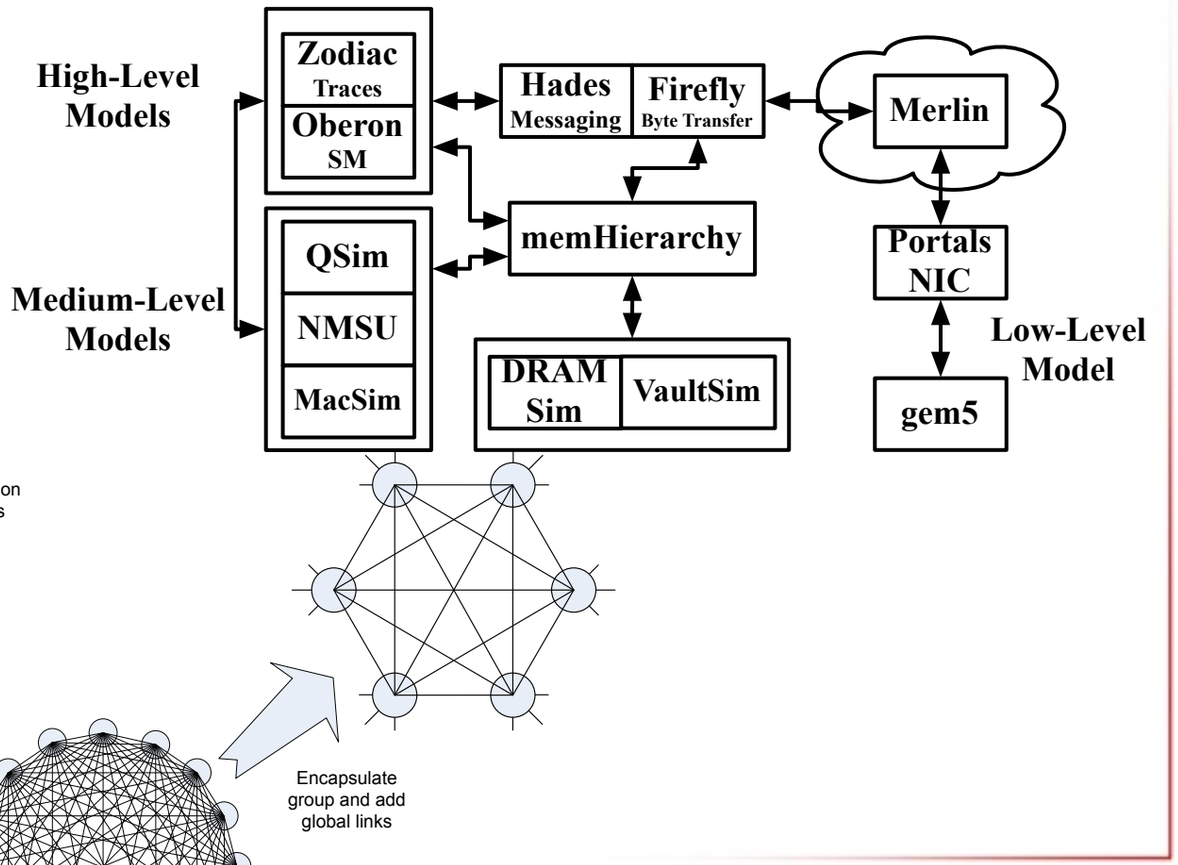
Consortium

- "Best of Breed" simulation suite
- Combine Lab, academic, & industry

SST Capabilities

- Processor & Memory Simulations
  - Extensive cache models
    - Snoopy and Directory-based coherency models
    - Multiple pre-fetcher models
  - Multiple Drivers
    - Execution driven (gem5, pintool, qsim)
    - Trace driven (ariel, oberon)
  - Accurate Memory Simulators
    - DRAMSim, VaultSim, HybridSim
- Network Simulations
  - High-performance network topologies
    - N-dim Torus, Fat-tree, Dragonfly, etc.
  - Multi-scale
    - Network-on-chip
    - 100K+ nodes
  - Multiple Drivers
    - Traces
    - State-machines
    - Pattern-based stochastic





Explorations

Sandia National Laboratories

Two-Level Memory

Stacked Memory

Point Design

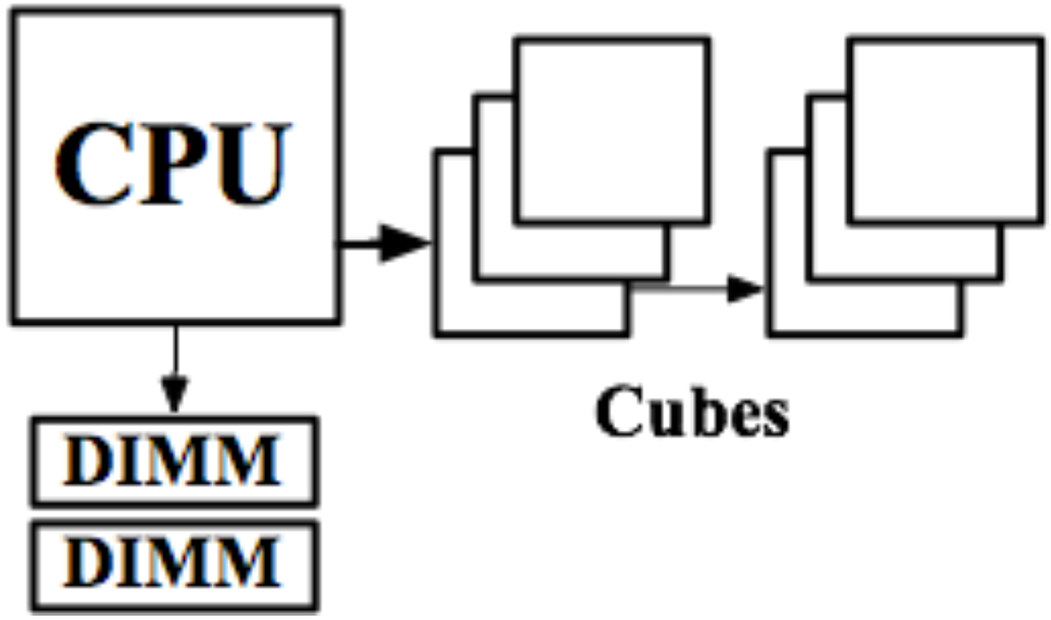
Functions In Memory

Bandwidth & Topology

Add in deps

Memory Stack

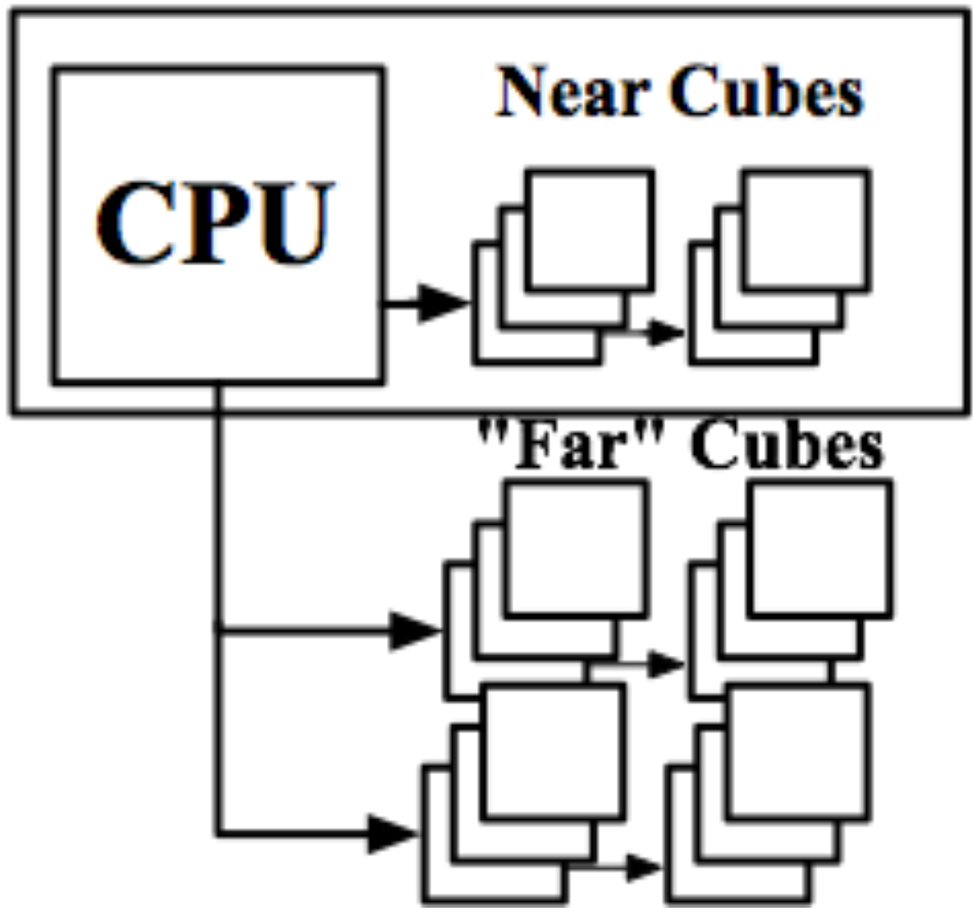
- Dimensions
  - Starting Point: 64vaults, double bw/vault 1.28TB/sec (x4 gen3 cube)
  - Bandwidth: internal, external, photonics. At least factor of 2 in each direction
  - Other: Flow control, impact on cache hierarchy
  - Topology: chains, rings
- Methodology
  - Cycle-level simulation (gem5/SST)
  - Why: Subtle interactions of proc/mem not conducive to tracing, does not require multi-node



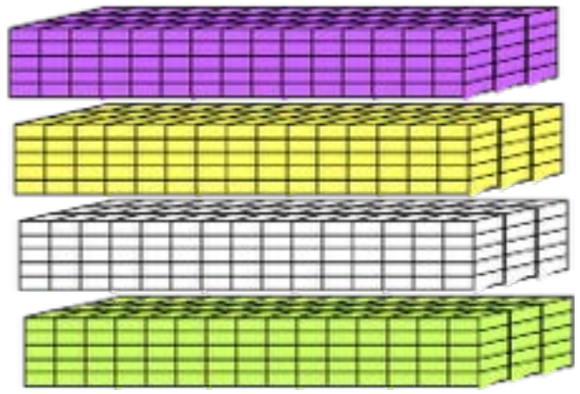
gem5 -> MemHierarchy

Multi-Level Memory

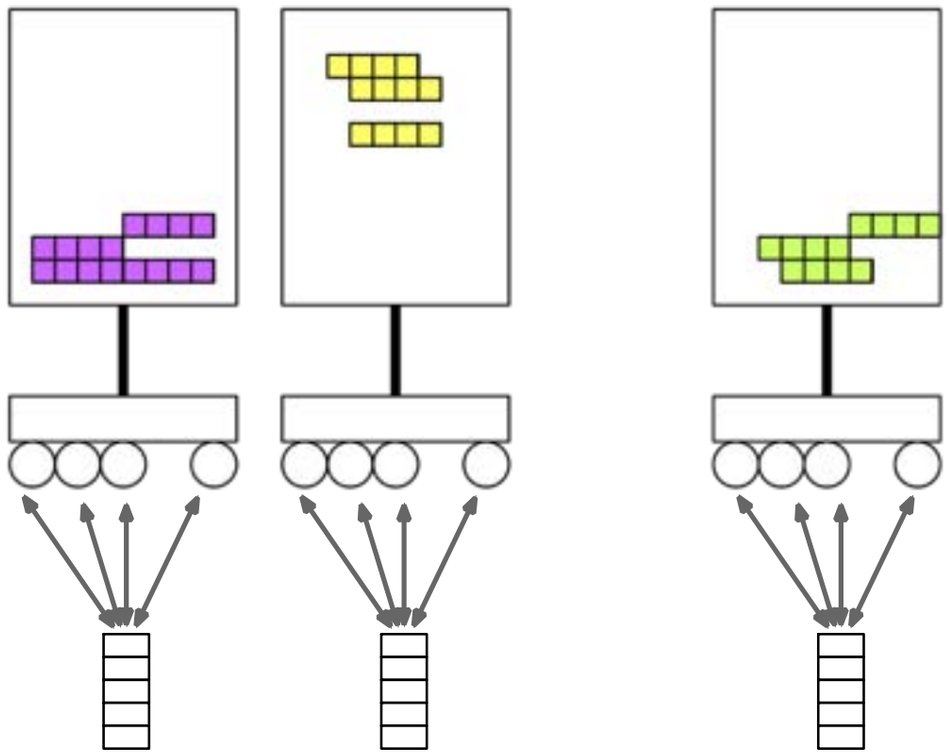
- Perhaps the most worrying HW development coming at us
- Dimensions
  - Ratio (capacity, speed)
  - Topology: separate channel, from cubes
  - Type: DRAM or NVRAM
- Priority: High
- Methodology
  - Instruction tracing & analysis with SST (mem hierarchy, Ariel Tracing)
  - Examining Sorting Algorithm
- Status
  - Benchmarks running, currently debugging
  - Finding: 2-level is difficult (‘hidden’ allocations)



Preliminary Study

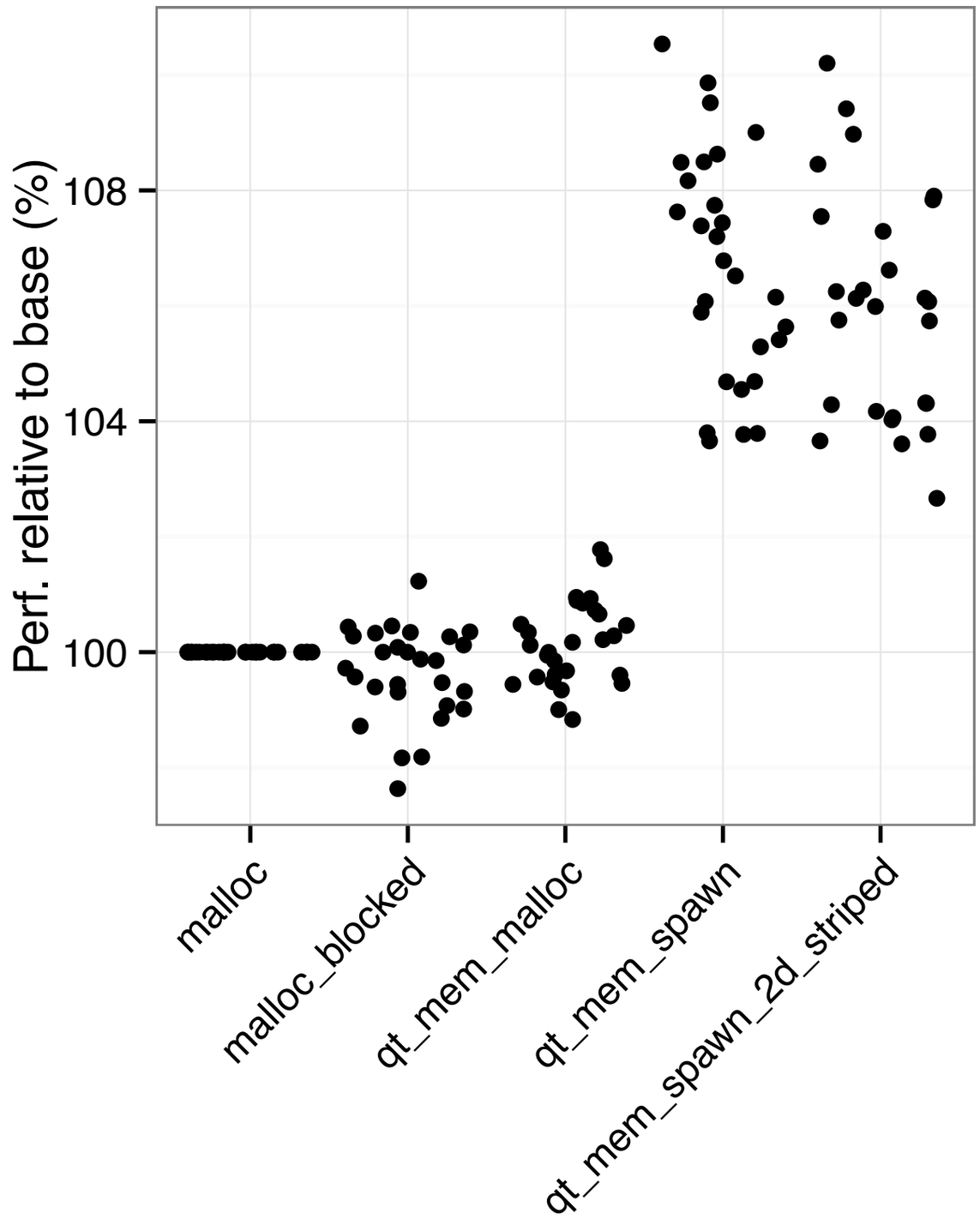


- Task-parallel miniGhost:
  - 3-D computational domain is single contiguous allocation
  - Computation is 2-D 5-point stencil
  - Not an interesting task-parallel application: regular, balanced, …
- Originally designed to study halo exchange
  - Task-parallel version for working
- Preliminary data
  - Platform: quad-socket, 8-cores per socket
  - Qthreads:
    - Multiple workers-per-queue, work-stealing
    - One queue per NUMA regions
  - Variables:
    - Level of decomposition (number of xy-slices)
    - Socket utilization (active cores per socket)
    - Allocation and spawning strategies



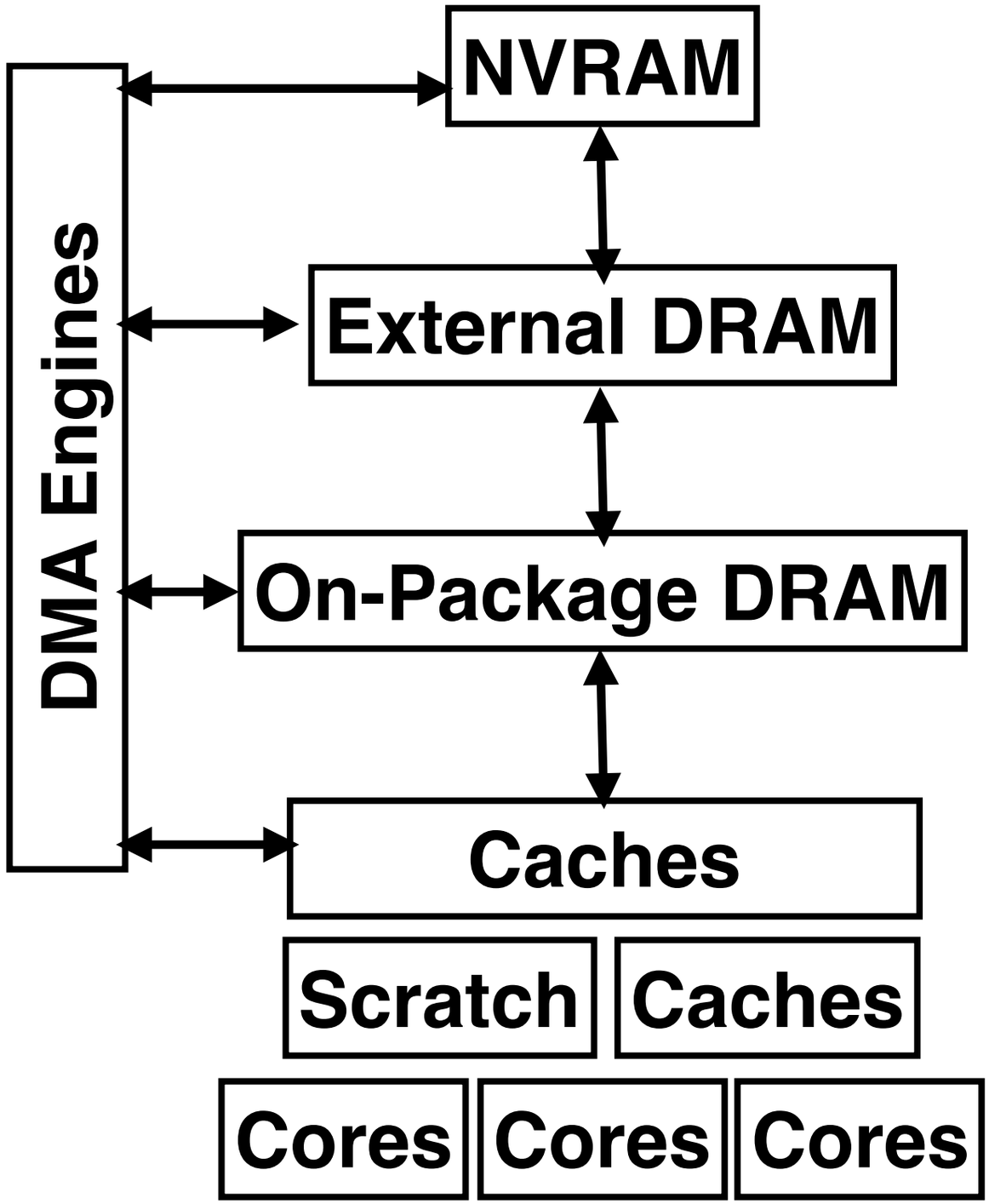
Allocation and Spawning Strategies

- Early results show small, but noticeable differences
- Differences would be larger with greater difference between memory bandwidths, greater rewriting



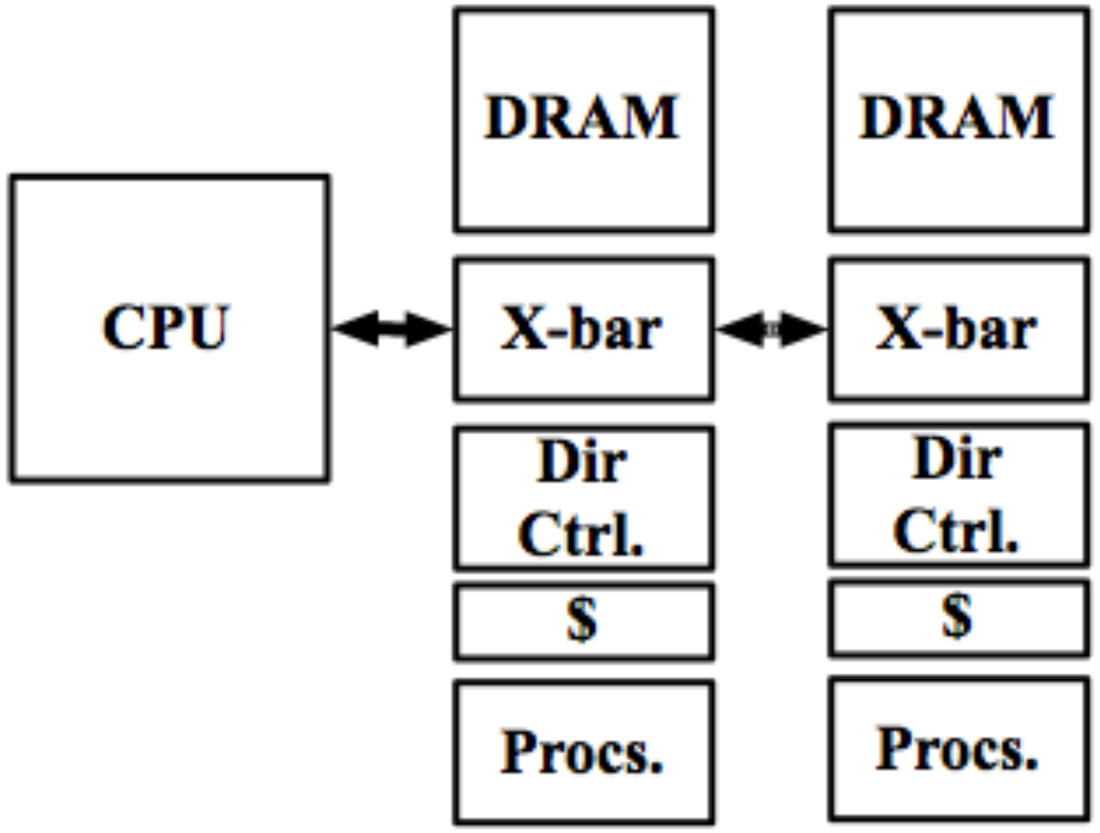
Multi-level Memory Simulations

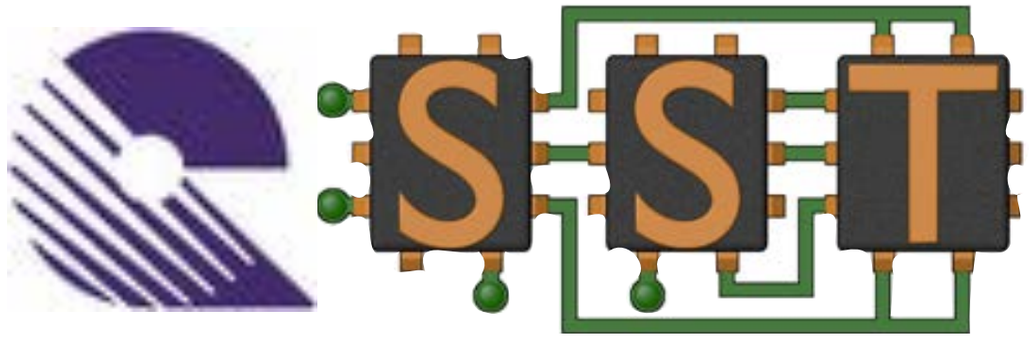
- **Future**: Rewrite Apps, simulate w/ SST
  - Explore scratchpads
  - Explore on-package
  - Explore NVRAM
- Required components
  - Multiple levels
  - Accurate DRAM, NVRAM
  - DMA Engines
  - **Cache Coherency**



Functions-in-memory

- Design Space
  - In-memory processing: #, capability of cores
    - Full ‘core’ or limited functions?
  - Locality: Ignore, NUMA, App. Hinting
  - Coherency Options: Directory-based, snoop filtering
- Methodology
  - Execution-based simulation, gem5+memHierarchy
  - Modified Qthreads, application “hinting”
  - Provides timing estimates, captures proc/ mem feedback





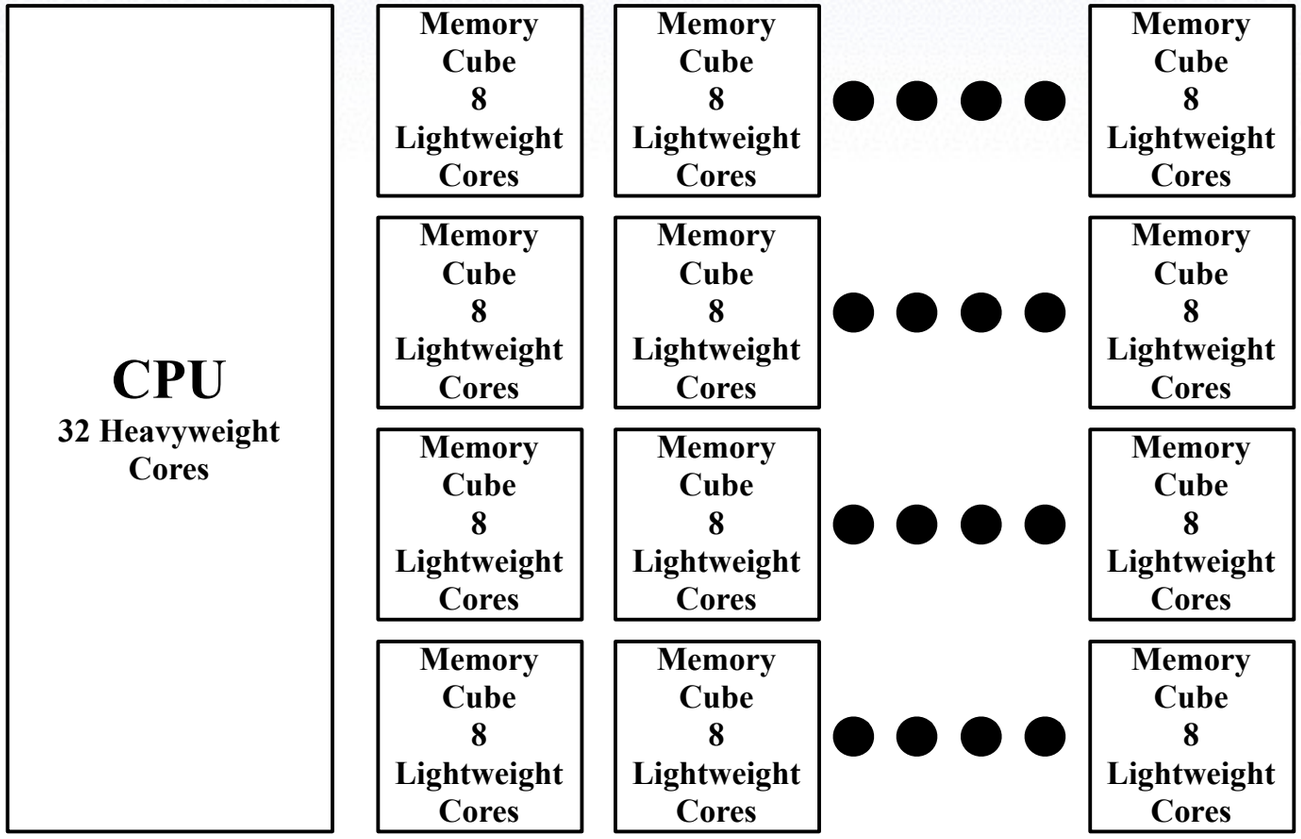
gem5 -> MemHierarchy

Processing-in-memory Questions

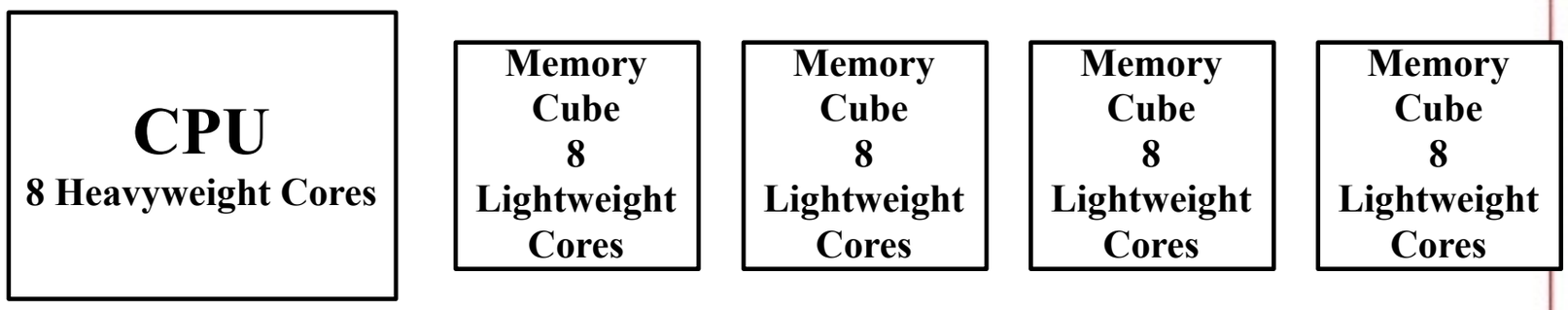
- What functions?
  - For XGC: Assume general purpose processing
  - In general: vectors? data movement engines? other?
- Major Questions
  - Architecture Design Space
    - How many processors? How big?
    - Coherency models
  - How do we layout data
    - Interleaving vs. locality
    - Placement commands?
  - How do we place / start tasks?

PIM Baseline system

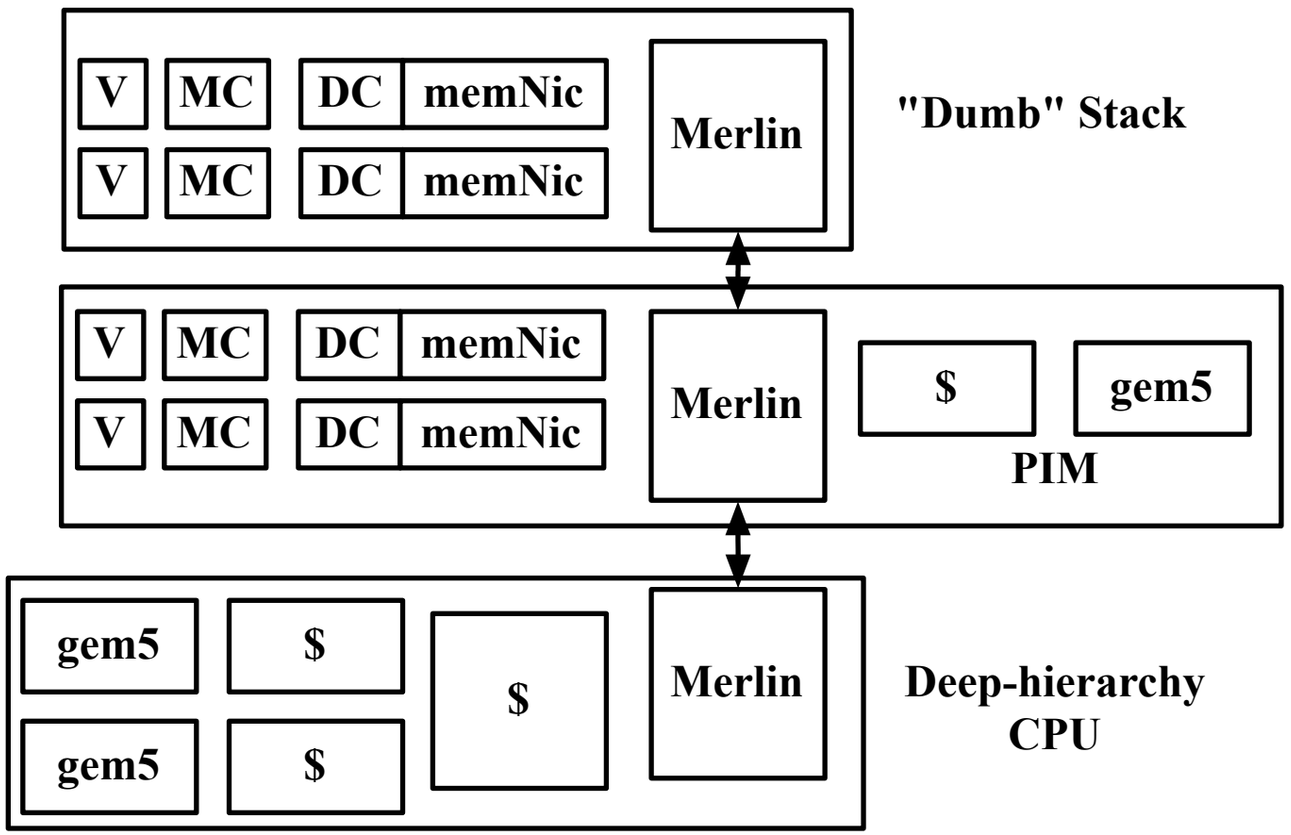
"Target System"
- CPU: 32 Heavy Cores
- PIMS: 4 chains of 8 cubes
- Each cube has 8 light weight cores



"Simulated System"
- CPU: 8 Heavy Cores
- PIMS: 1 chains of 4cubes
- Each cube has 8 light weight cores



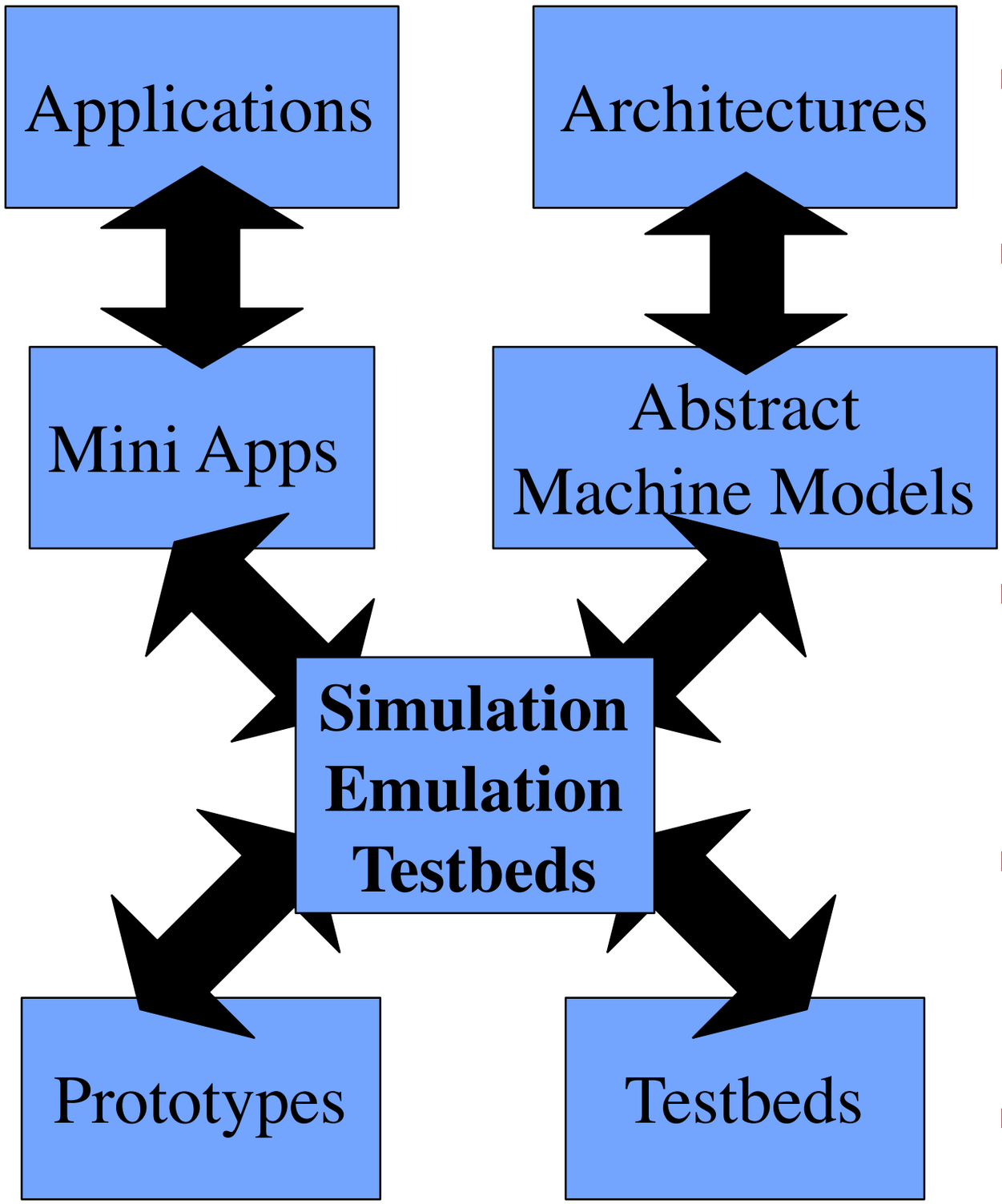
PIM Simulation Setup



- Use existing memHierarchy cache protocols
- Naive setup
  - Treats PIMs as NUMA sockets
  - Simple topology

- Explore…
  - Task placement
  - Data placement
  - Architecture Space

Summary



- **Architecture & Apps radically changing**
- **Need testbeds to explore near-term architectures & programming models before they enter production**
- **Need simulation to explore radical architecture changes**

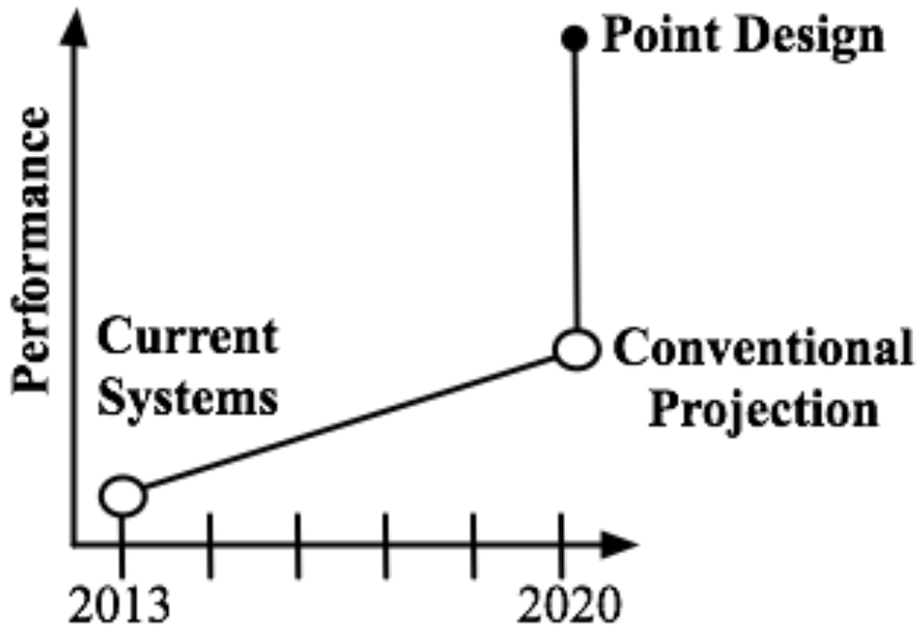
- **Infrastructure (testbeds, emulators, simulators) is major undertaking**
- **But, we are doomed without it**

Sandia National Laboratories

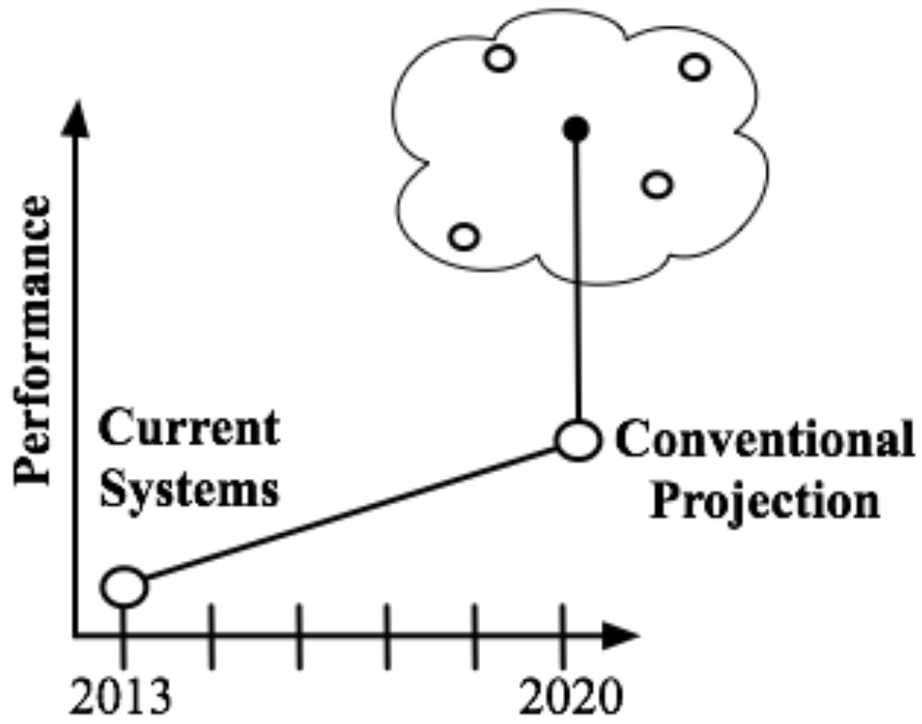
Questions

Point Design Process

- Goals
  - Demonstrate the Point Design is better than the projected state-of-the art
  - Explore design space around the Point Design to determine if a unified architecture for DS&A and NW is feasible
  - Keep design space exploration tractable
- Process
  - Compare against conventional projections
  - Explore individual axes of the design space (e.g. memory hierarchy, functions-in-memory)



Design Space Exploration



Task Placement

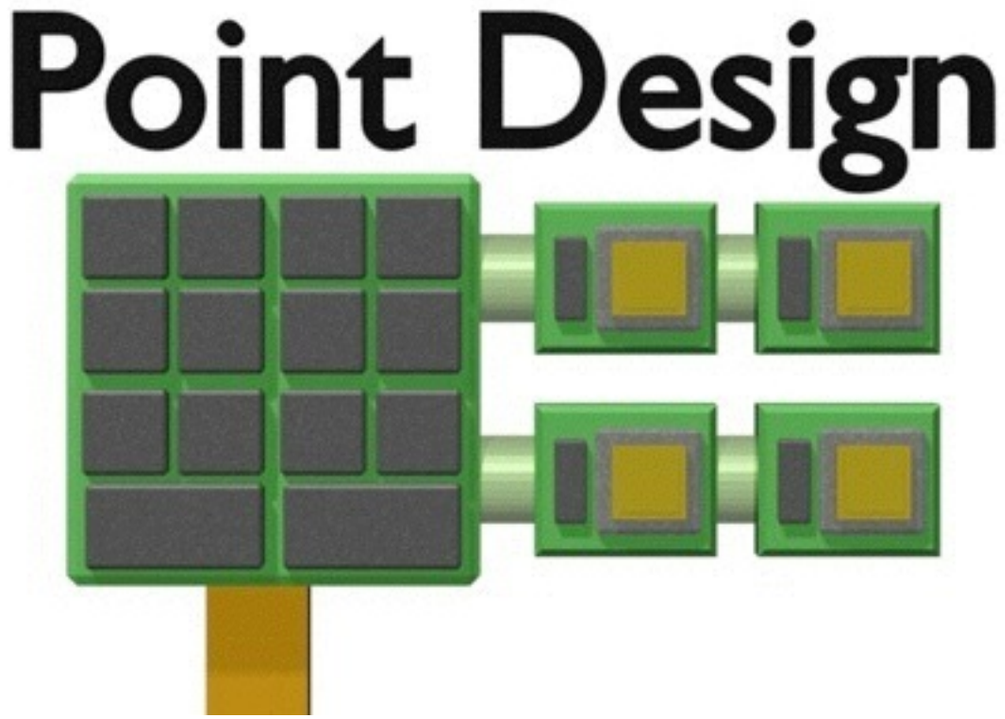
- Starting point: Ignore it
- Extend NUMA aware placement
  - sst needs to communicate libnuma-like info
  - Currently: looks at shared levels of hierarchy, does not use distances/lenghts
  - levels of hierarchy can be tagged in input file, env variable says where to place task queueus
  - Research question: what information can we add to improve shepherd placement?
- User directed
  - How does the user specify “this task goes with this data”

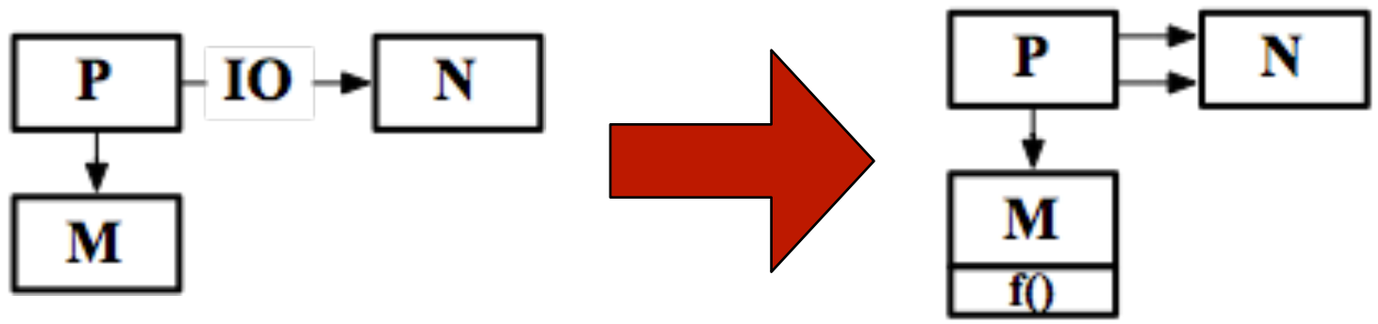
Data Placement

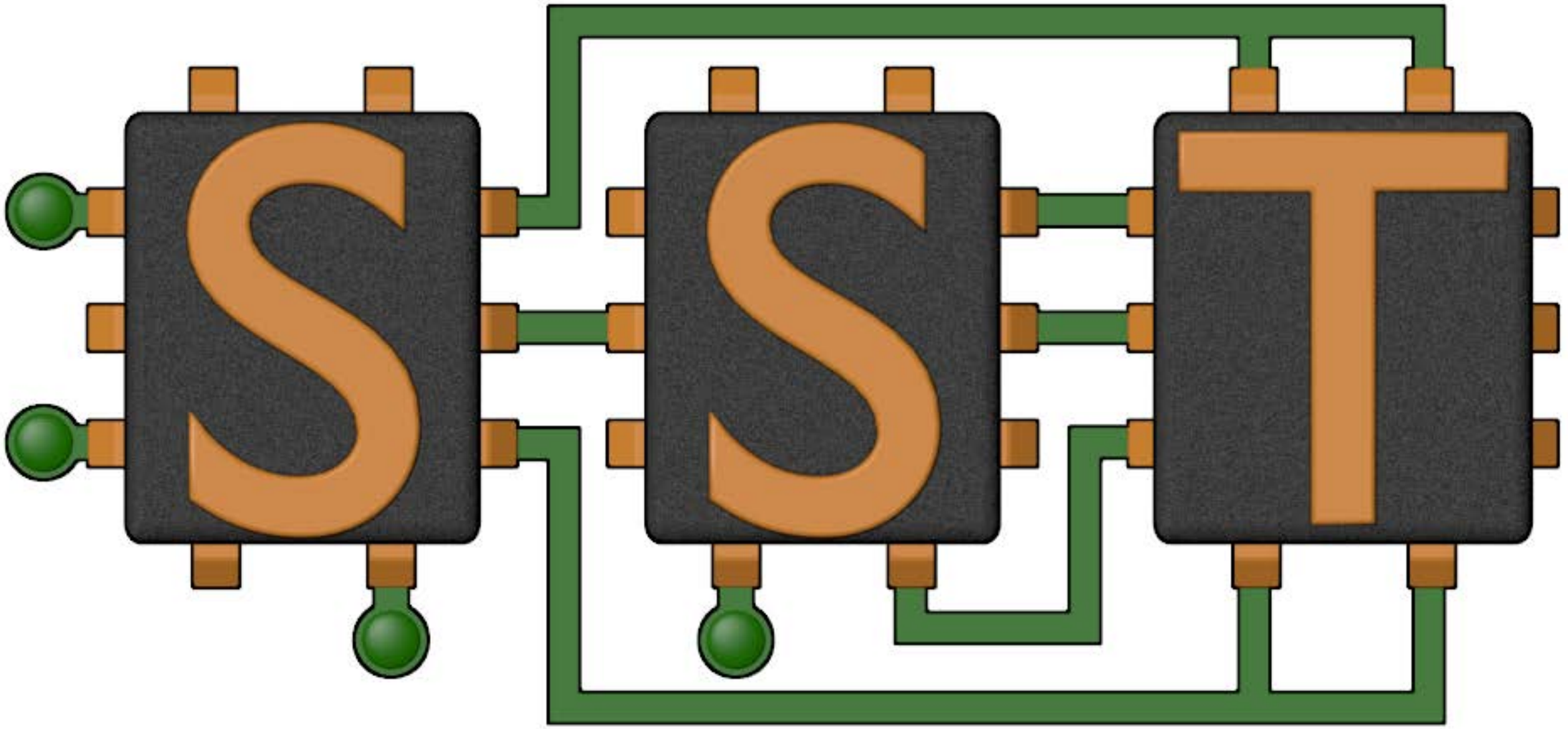
- Can reuse system map for task placement for data placement. i.e. Qthreads doing applications
- gem5 currently does simple linear allocation
- Special malloc call: malloc to stack X
  - mallocAt()
- System emulation: need free-list-based allocator so we can add allocation from multiple ranges (i.e. stacks)

Point Design

- Node
  - Cores, NIC both on-chip
  - Cores connected to NIC through coherent interface
  - Off-chip stacked memory
  - Functions in memory
- Memory
  - “Cubes”: DRAM + Logic Layer
  - Flat Memory Space

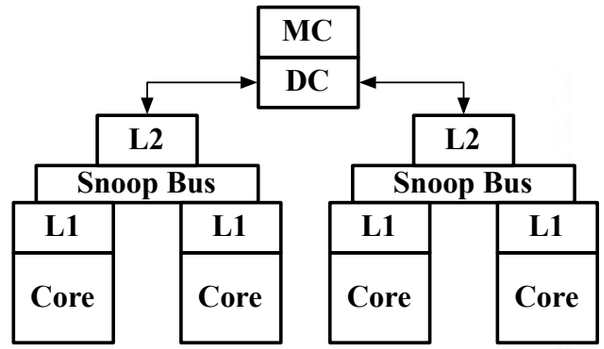




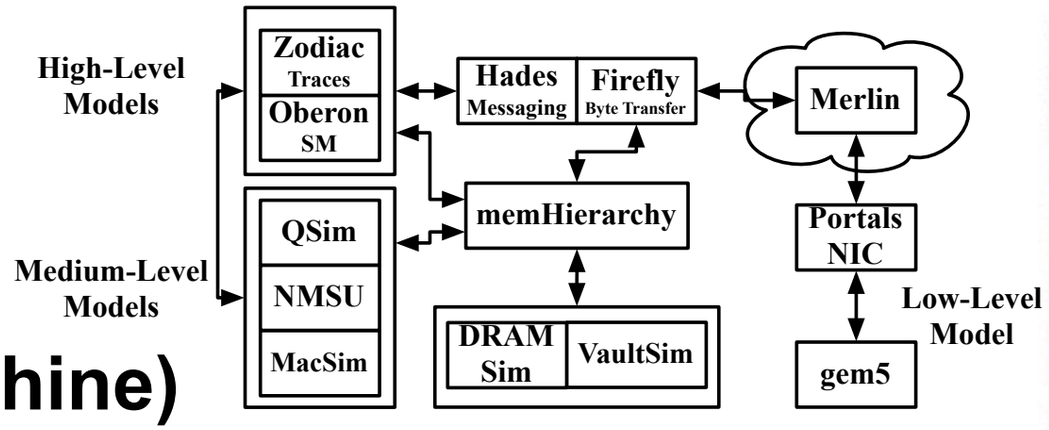

S
S
T

SST Recent Enhancements

- Improved interoperability with gem5
- memHierarchy allows more flexible memory hierarchies, NoC, off-chip, multi-level memories



- Improved programmability
  - Qthreads, OpenMP support
  - Many more applications can run in simulator now
- New / Emerging Models
  - QSim (high speed emulator)
  - VaultSim stacked DRAM
  - High-level models (trace, state machine)
  - Chisel, SystemC interoperability
  - NVRAM / DRAM Hybrids



- Improved SW Engineering (docs, usability, portability)
- New python-based configuration