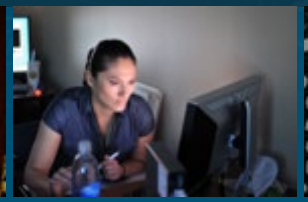
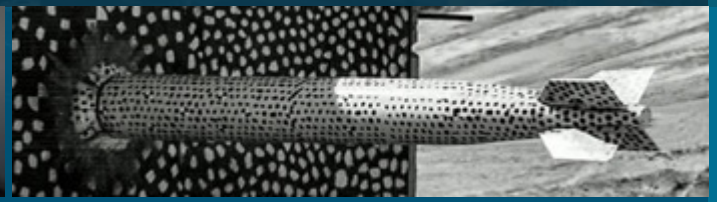
Accelerated Architectures and the Structural Simulation Toolkit





PRESENTED BY

Gwen Voskuilen, Sandia National Labs

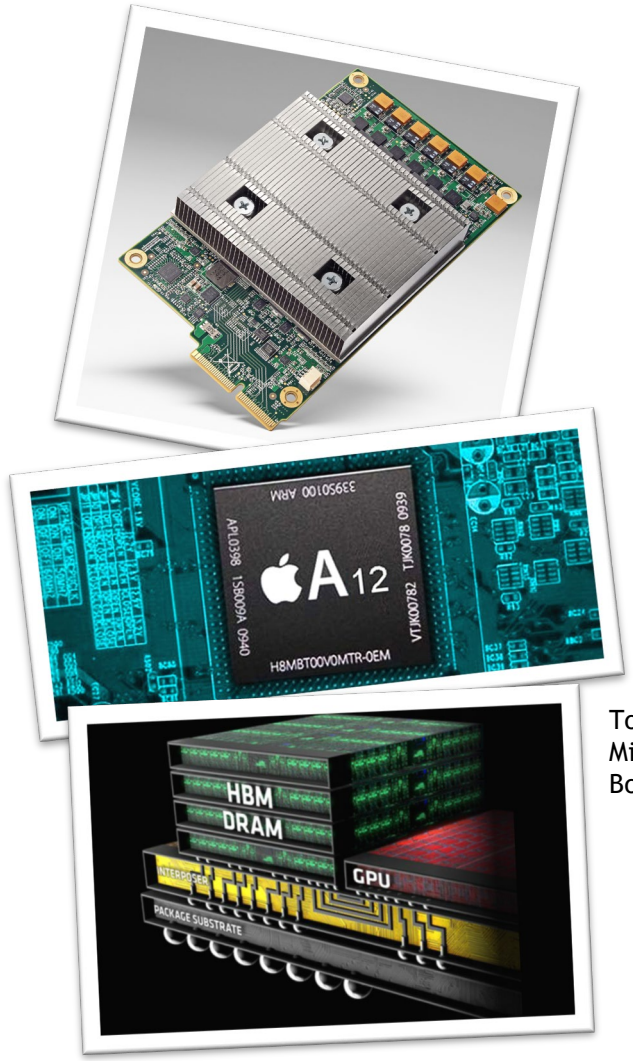
SAND2021-0918 PE

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Acceleration is becoming the norm

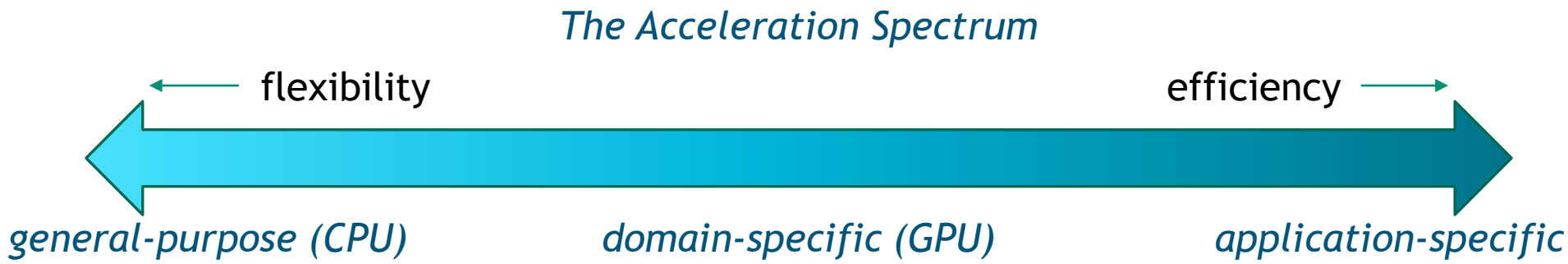
- As scaling slows, architectures become more creative
  - Instead of just bigger
  - Instead of just more parallel
- Rise of customizable and accelerator architectures
  - Heterogeneous processors, GPUs, FPGAs, etc.
  - Customizable CPU architectures (ARM, RISC-V)
  - Proliferation of programming environments for accelerators that are targeted at mainstream programmers
  - Effort to standardize new interconnect protocols (CXL, CCIX)
  - Multi-chip modules enabling closer custom/commodity integration

This talk: overview of two interesting projects looking at these accelerated architectures



Top: Google TPU
Middle: Apple A12 SoC
Bottom: AMD

Application-specific acceleration

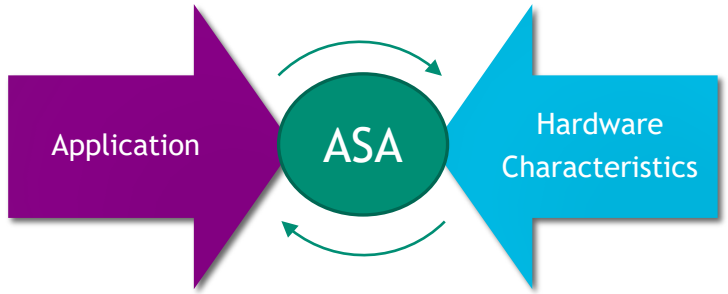


- Application specific accelerators (ASAs) map applications directly to hardware
  - Performance and energy benefits from efficiency
  - Implemented via dedicated hardware (ASIC) or reconfigurable (FPGA)
- Widely used in many domains especially where a single computation dominates
- But, not widely used in HPC
  - Too many diverse applications
  - Applications are complex and continually evolving

Co-designing FPGA-based ASAs for scientific applications

- Targeting several applications & DOE proxy apps that do not run well on GPUs
- The question is not can this be done, the question is:
  How can we enable high-performance ASAs within conventional supercomputers, for large, evolving HPC applications developed by programmers without circuit expertise?
- Joint work with Sandia and Georgia Tech
  - Tom Conte & Jeffrey Young
- The rise of accelerator archit… ecosystem in the right direction
  - Larger, more powe… …at can be more tightly integrated into systems
  - ASICs and SoCs b… …cheaper, easier to integrate
  - Renewed interest in making hardware design accessible
  - But, still a time-consuming, complex development process

Under Construction

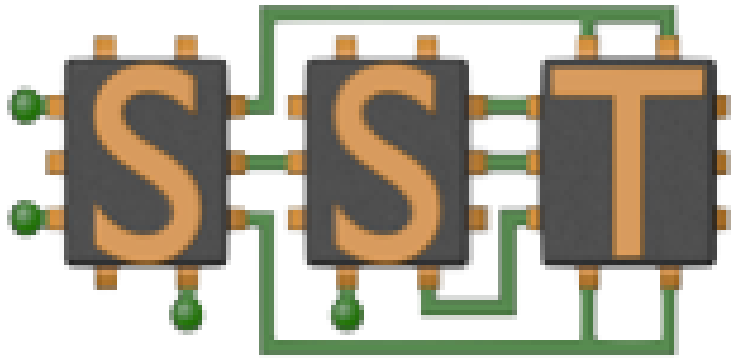


Modeling Accelerated Architectures in SST

Architecture innovations drive models

- Creative (aka accelerator) architectures are challenging to model
  - The more "stuff" to simulate, the slower it gets
  - Need to develop new models, not just scaling existing ones
  - Drowning in data

Architecture trend → slower development of slower simulations



The Structural Simulation Toolkit

Goals

- Create a standard architectural *simulation framework* for HPC*
- Ability to evaluate future systems on DOE/DOD workloads
- *Use supercomputers to design supercomputers*

Technical Approach

- **Parallel** Discrete Event core
  - With conservative optimization over MPI/Threads
- **Interoperability**
  - Node and system-scale models
- **Multi-scale**
  - Detailed (~cycle) and simple models that interoperate
- **Open**
  - Open Core, non-viral, modular

Status

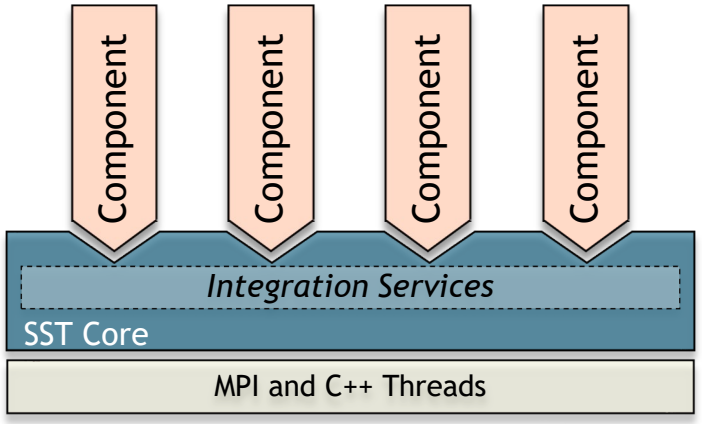
- Parallel framework (*SST Core*)
- Integrated libraries of components (*Elements*)
- Current Release (10.1)
  - https://sst-simulator.org
  - https://github/sstsimulator

Consortium

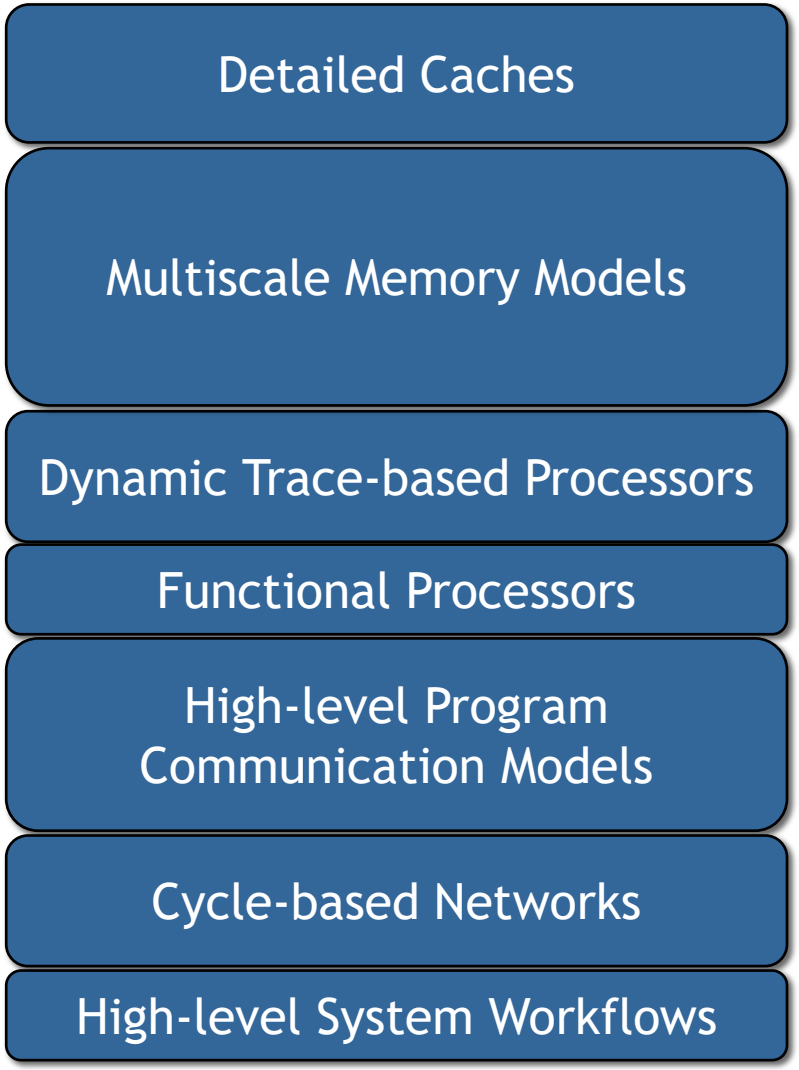
- "Best of Breed" simulation suite
- Combine Lab, Academic & Industry

SST Architecture

- SST **Core** framework
  - The backbone of simulation
  - Provides utilities and interfaces for simulation components (models)
    - Clocks, event exchange, statistics and parameter management, parallelism support, etc.
- SST **Element** libraries
  - Libraries of components that perform the actual simulation
  - Elements include processors, memory, network, etc.
    - Includes many existing simulators: DRAMSim2, Spike, HMCSim, Ramulator, etc.



Breadth and Depth…

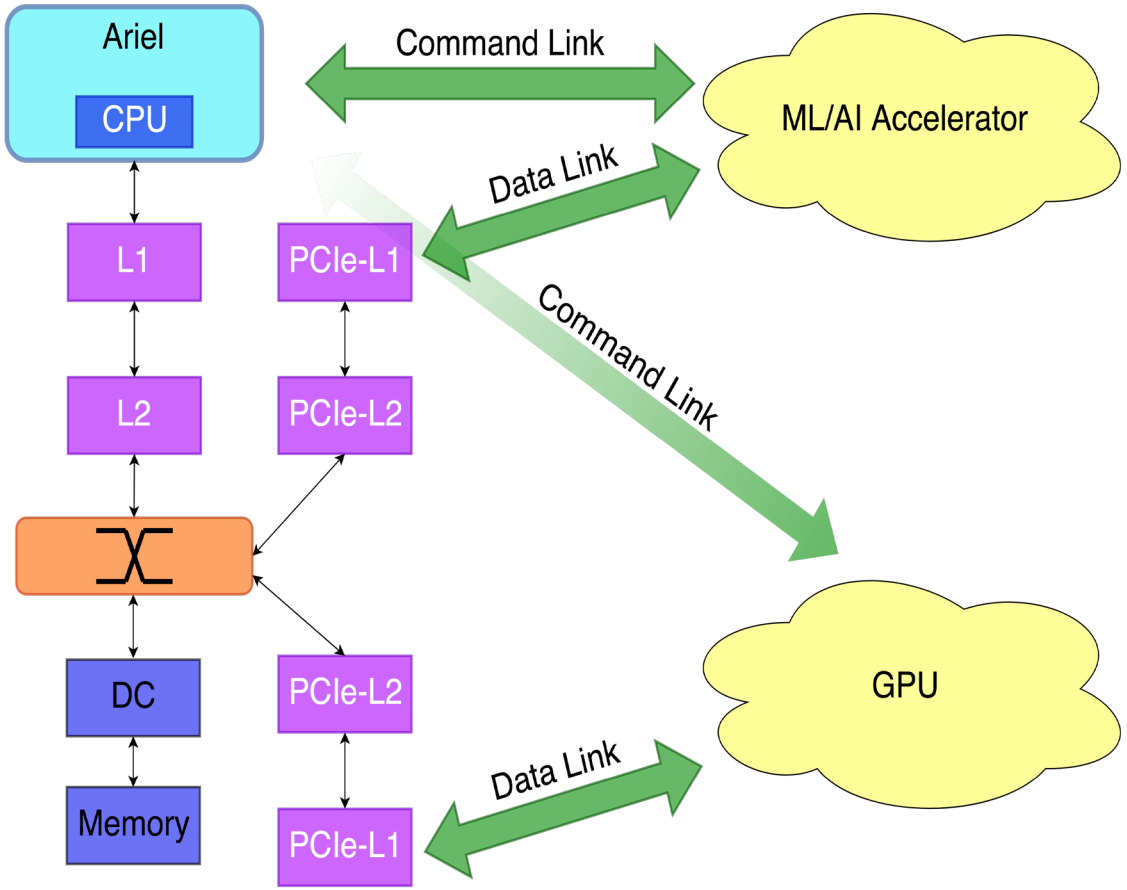


- memHierarchy - Cache and memory
- cassini - Prefetchers
- CRAMSim - DDR, HBM
- NVDIMMSim - Emerging Memories
- HMCSim - HMC
- TimingDRAM - Basic DDR model
- ariel - PIN-based dynamic tracing
- miranda - State-machine based core model
- GPGPUSim - GPU
- Spike - RISC-V ISA
- ember - State-machine message generation
- firefly - Communication protocols
- hermes - MPI-like interface
- merlin - Network router model and NIC
- kingsley - Network-on-chip model
- scheduler - Job-scheduler simulation models

Modeling Accelerated Architectures in SST: Building Blocks

- Recall: Slower development of slower simulation
- Solution: Robust ecosystem to mix-and-match architecture components
- Building up accelerator ecosystem
  - GPGPUSim (*balar*)
  - Neuromorphic processor (*gensa*)
  - Pipeline core model (*vanadis*)
- Defining interfaces to enable composability and flexibility
  - Accelerator interfaces built into existing models
  - Custom memory instruction support through *memHierarchy*
- Adding new capabilities for more flexible modeling
  - Composing HDL and C++ models in the same simulation

SST Future Directions and Challenges



- Missing pieces
  - Different accelerator architectures
    - ML/AI, quantum, neuromorphic
    - HDL simulation components
  - Interconnect protocols
    - PCIe, CXL, CCIX, etc.
  - Customizable CPU models
    - RISC-V, ARM, etc.
- Challenges:
  - How to effectively manage output of such a complex system?
    - Similar to "real-world" performance analysis, with even more data available
  - Can simulation itself be accelerated?