Toward Full-System Heterogeneous Simulation: Merging gem5-SALAM with Mainline gem5

Akanksha Chaudhari and **Matthew D. Sinclair**

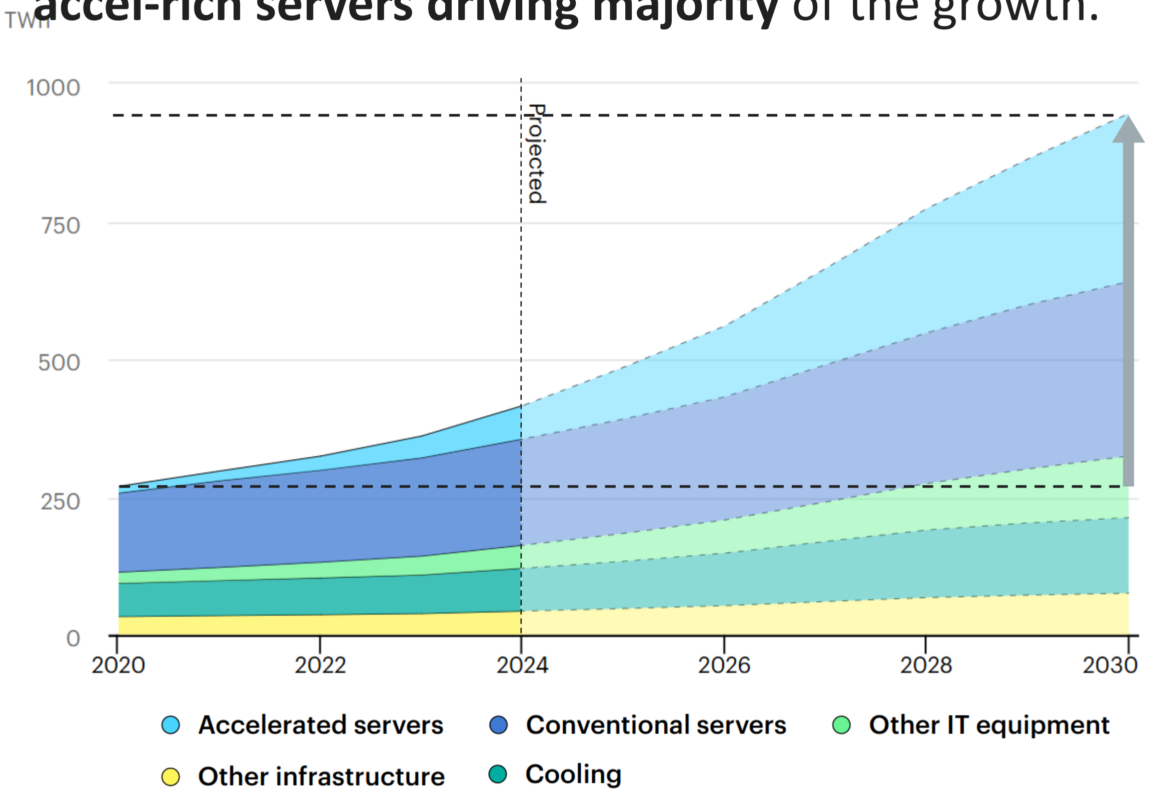
University of Wisconsin-Madison

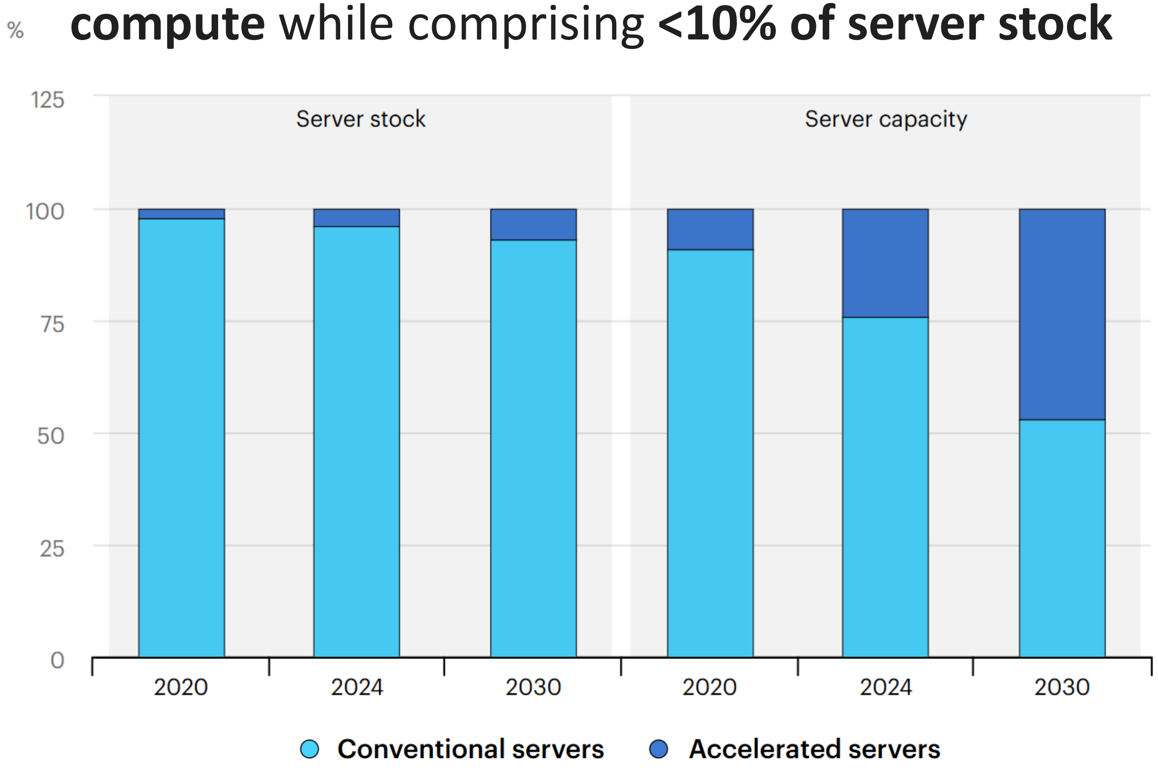
sinclair@cs.wisc.edu

Global Datacenter Electricity Consumption

Escalating Power Demands: Datacenter electricity consumption expected to **triple by 2030**, with **accel-rich servers driving majority** of the growth.

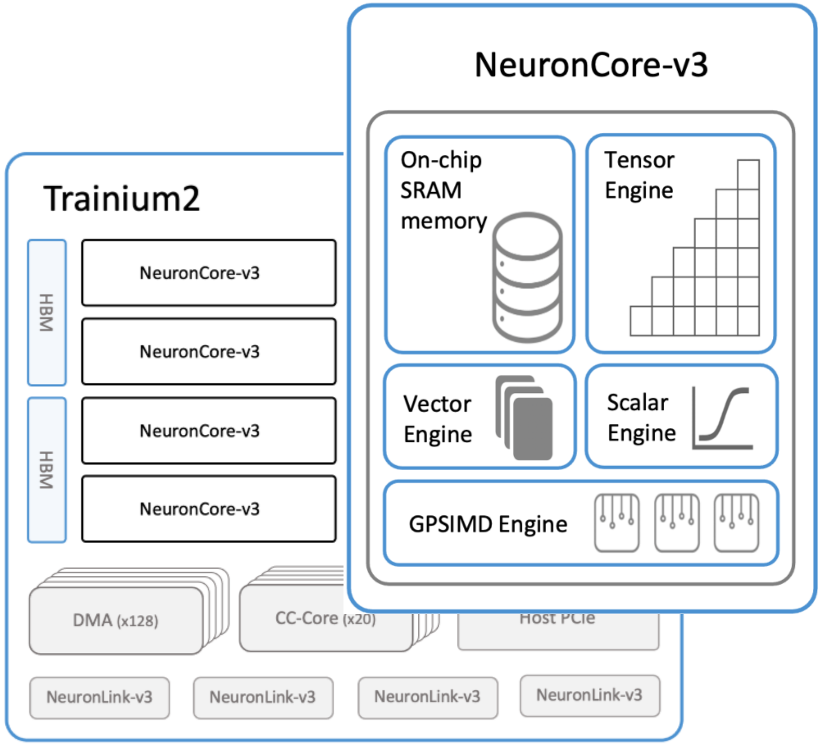
Shift toward heterogeneity: By 2030, **accelerated servers** are projected to deliver **~50% of total compute** while comprising **<10% of server stock**





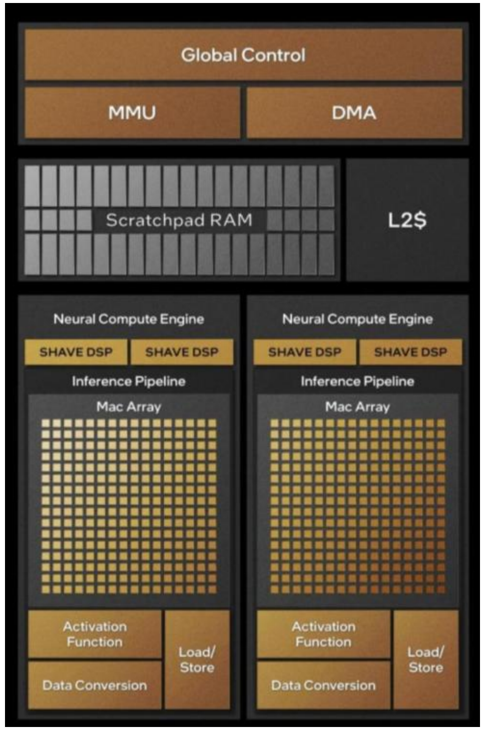
Source: IEA (2025), Energy and AI, IEA, Paris https://www.iea.org/reports/energy-and-ai, License: CC BY 4.0

Heterogeneity is the New Norm



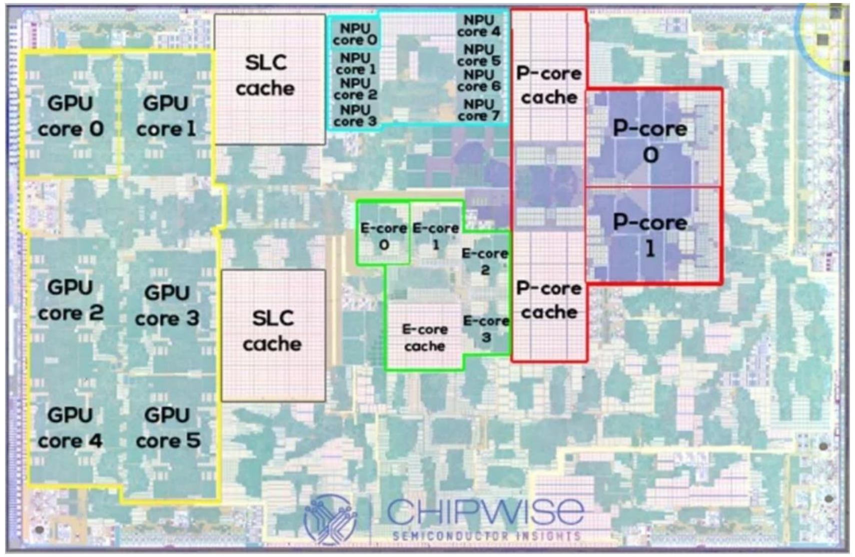
AWS Trainium2 NeuronCore-v3

Servers



Intel Meteor Lake

Laptops



Apple A15 Bionic

Phones/Tablets

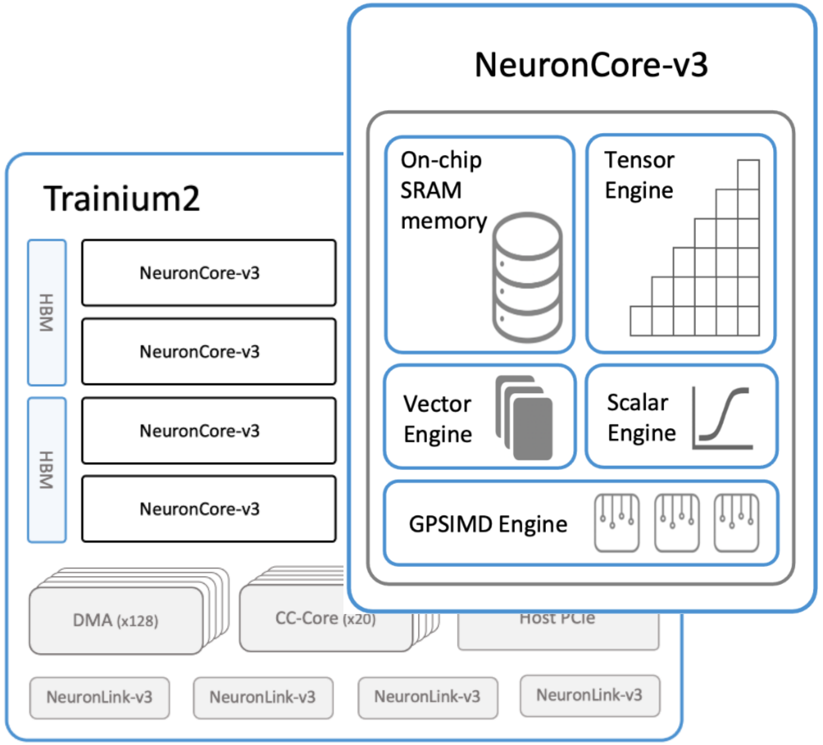
Sources: AWS Docs Neuron, Intel Tech Tour, Chipwise

Heterogeneity is the New Norm

Increasingly complex designs!

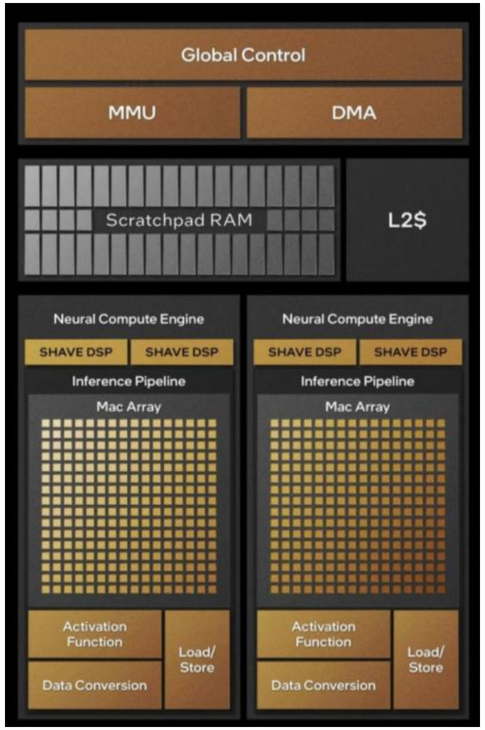
Non-trivial power/perf characteristics

Memory-compute co-optimization



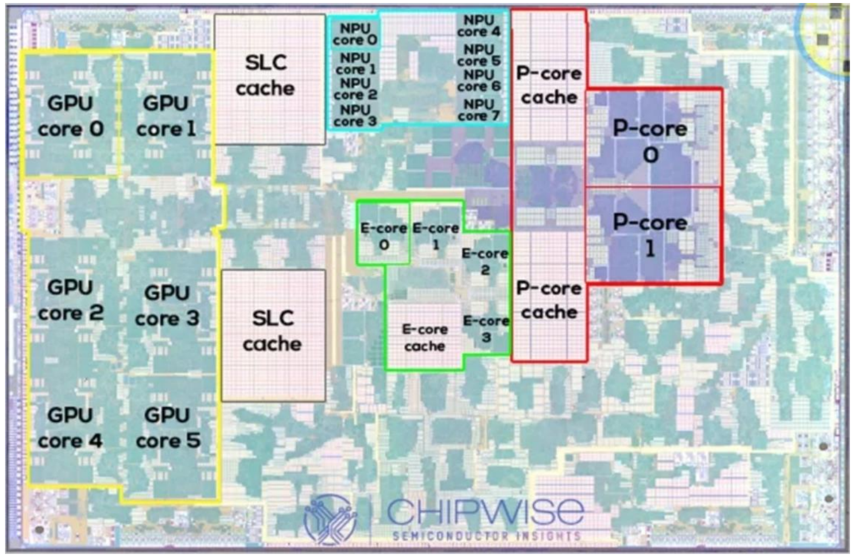
AWS Trainium2 NeuronCore-v3

Servers



Intel Meteor Lake

Laptops

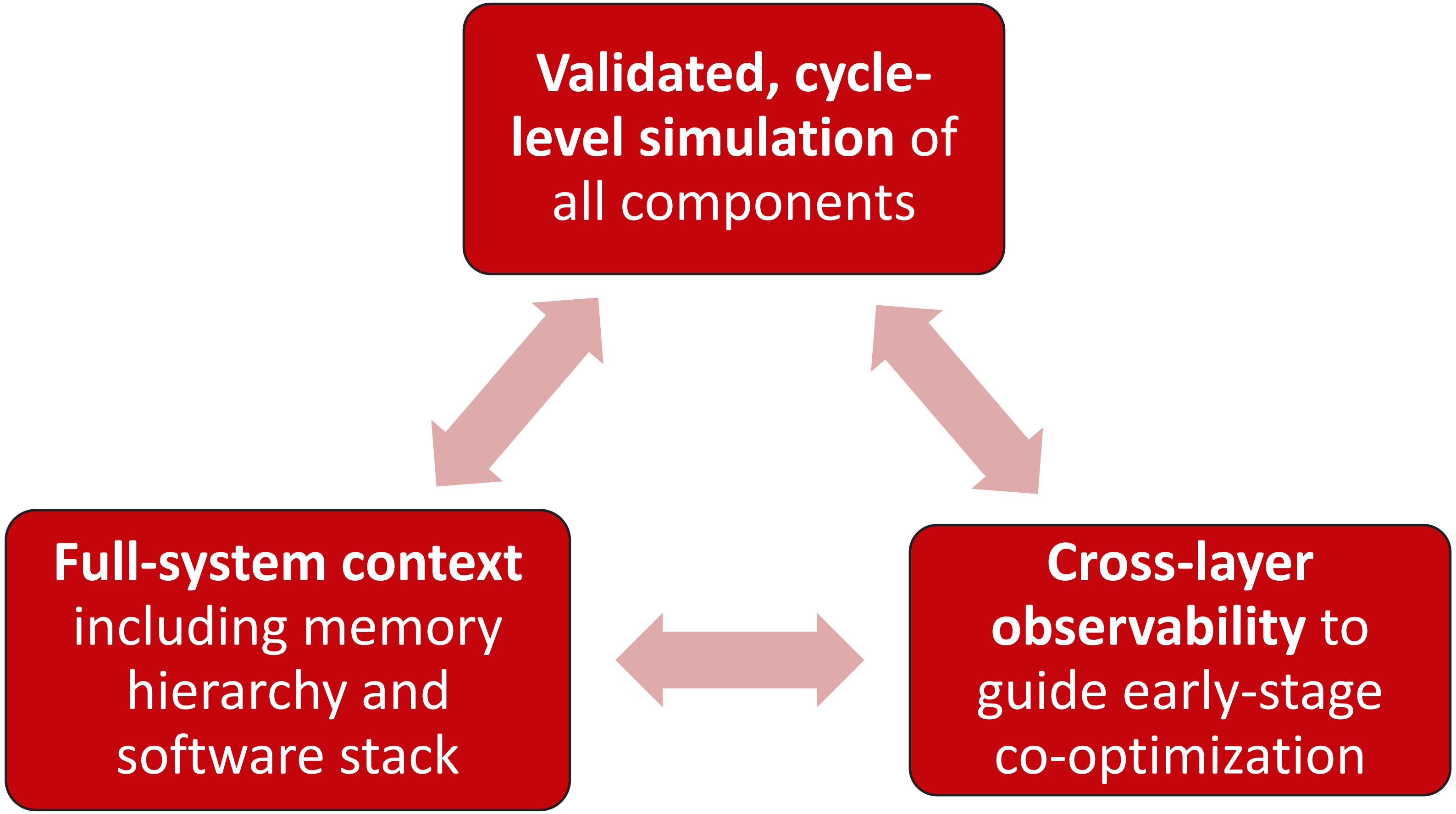


Apple A15 Bionic

Phones/Tablets

Sources: AWS Docs Neuron, Intel Tech Tour, Chipwise

Designing for Heterogeneity: What is Needed?



Validated, cycle-level simulation of all components

Full-system context including memory hierarchy and software stack

Cross-layer observability to guide early-stage co-optimization

The Simulation Gap

Simulator	CPU	GPU	Accelerators	FS Support
gem5 (v25) - SOTA	✓	✓	✗	✓
SALAM	✗	✗	✓	✗

The Simulation Gap

Simulator	CPU	GPU	Accelerators	FS Support
gem5 (v25) - SOTA	✓	✓	✗	✓
SALAM	✗	✗	✓	✗

No unified framework for simulating heterogeneous SoCs within the full-system context

The Simulation Gap

Simulator	CPU	GPU	Accelerators	FS Support
gem5 (v25) - SOTA	✓	✓	✗	✓
SALAM	✗	✗	✓	✗
gem5-SALAM	✓	✓	✓	✓

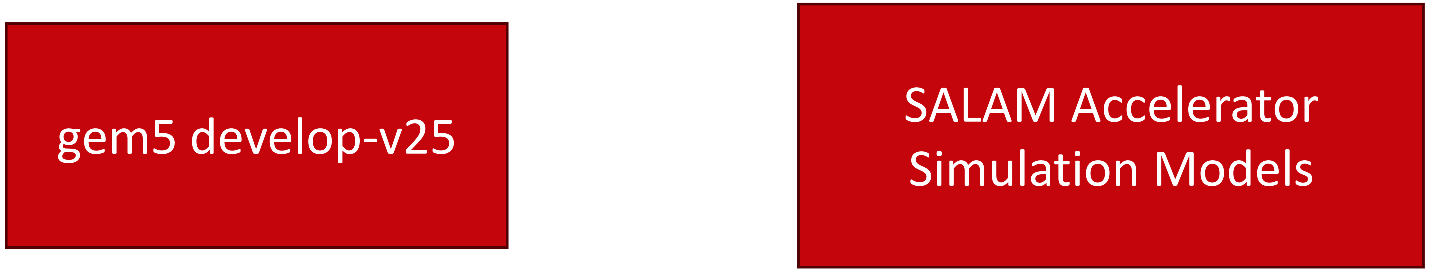
This work: unify head of gem5, gem5-SALAM

Enables unified simulation of CPUs, GPUs, and accelerators

Outline

- Introduction
- Background
- Integration Methodology
- Frequency Scaling Study
- Results and Analysis
- Conclusions and Future Work

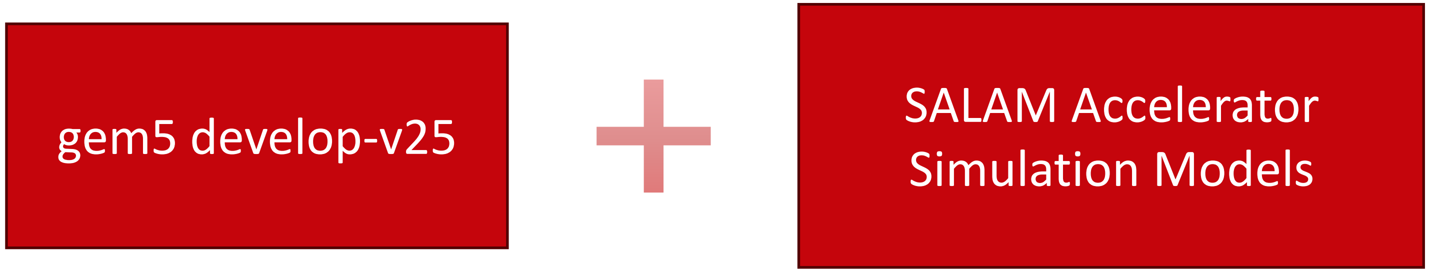
Integration Overview



gem5 develop-v25

SALAM Accelerator Simulation Models

Integration Overview

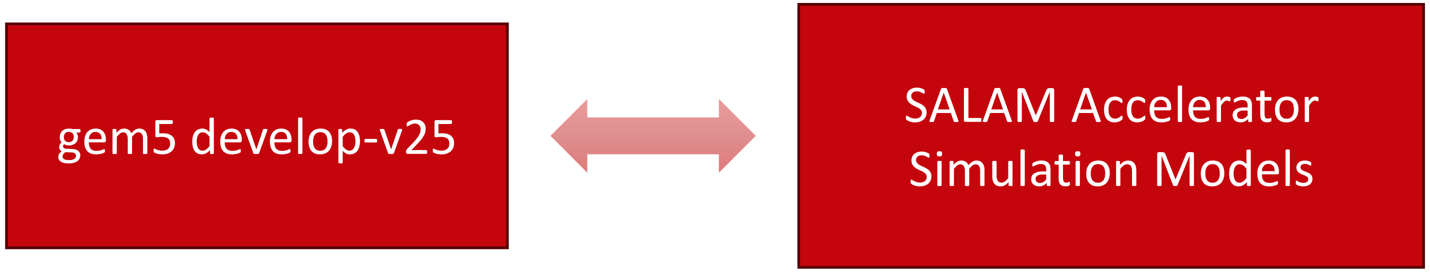


gem5 develop-v25

SALAM Accelerator Simulation Models

Incorporating key Accelerator Modeling and Communication Components

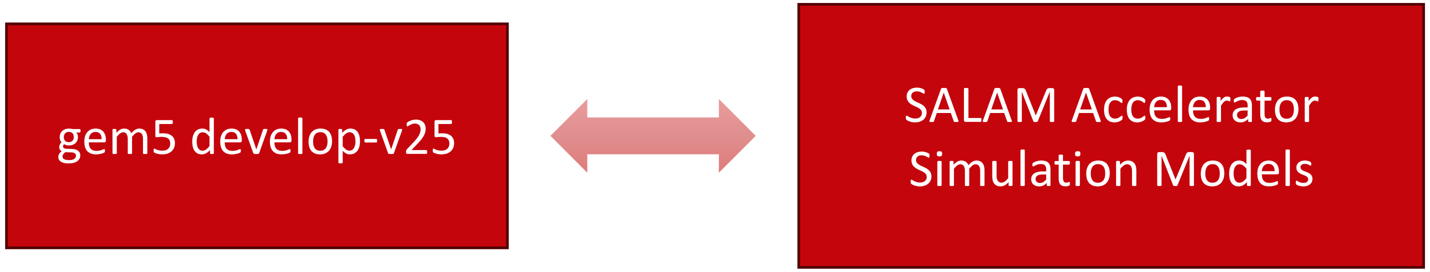
Integration Overview



Incorporating key Accelerator Modeling and Communication Components

Refactoring and Aligning Interfaces

Integration Overview



gem5 develop-v25

SALAM Accelerator Simulation Models

Incorporating key Accelerator Modeling and Communication Components

Refactoring and Aligning Interfaces

Validating Functional Equivalence

Methodology: Incorporating SALAM Components

- **LLVMInterface**: Enables cycle-level datapath simulation via runtime IR parsing.
- **CommInterface**: Exposes accelerators via memory-mapped registers and programmable interrupts.
- **Memory Models** (SPMs, DMAs, etc): For low-latency access, data movement, and streaming.
- **AccCluster:** Groups accelerators, local memories, and DMAs into modular subsystems.
- **Hardware Profile Generator:** Extended the toolchain to fully automate the generation of functional units and instruction timing models from user-defined profiles.
- **Cacti-SALAM Power Models:** Modernized the framework to support energy and timing estimation via config files for SPMs

Methodology: Refactoring and Aligning Interfaces

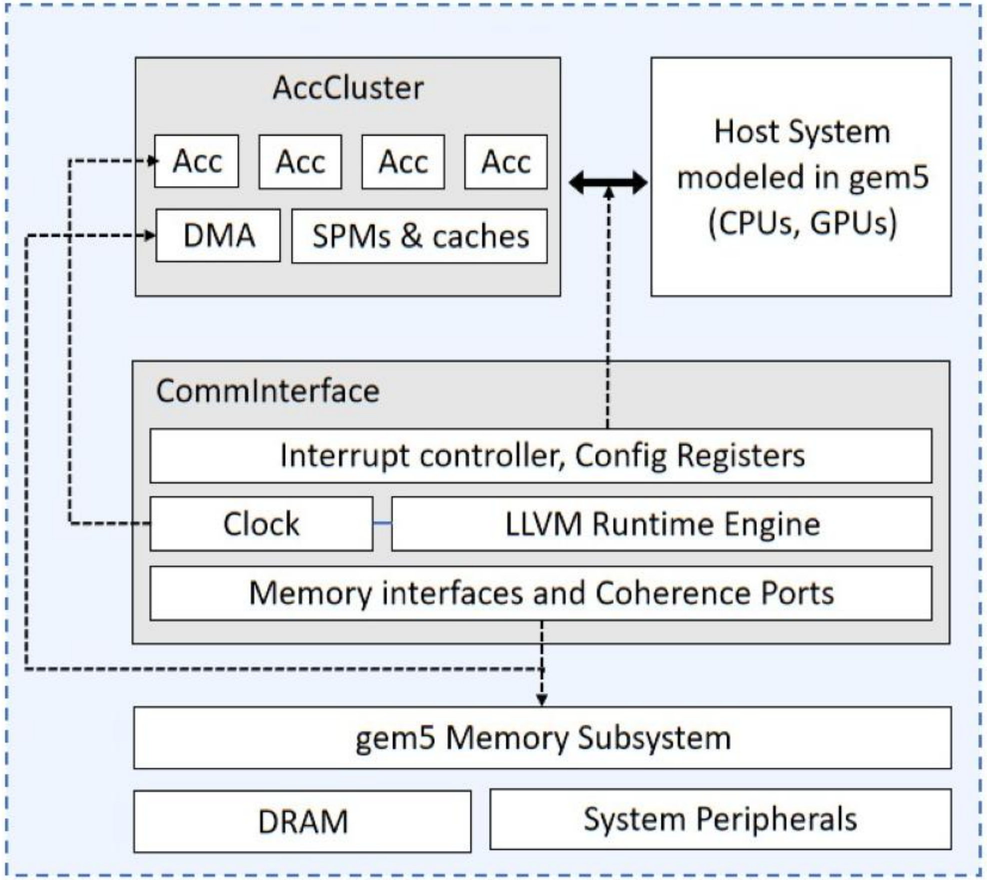
- **SimObject Alignment:** Refactored accelerator classes to conform with gem5’s latest SimObject conventions, ensuring proper initialization and parameter declaration structures.
- **Type-safety Fixes:** Replaced unsafe pointer casts in LLVM instruction simulation with intermediate 32-bit variables. This resolved array bounds and strict aliasing warnings during bitcasting.
- **Latency Generation Standardization:** Migrated custom random latency utilities to gem5’s standardized random number generation framework for reproducibility and consistency.
- **Address Range Corrections:** Fixed off-by-one errors in address range definitions for scratchpad and register bank modules to follow gem5’s inclusive-exclusive address semantics.
- **Environment and ISA Configuration Updates:** Updated build environment handling to align with gem5’s latest ISA-specific configuration mechanism, resolving prior compatibility issues.
- **Build System Integration for LLVM:** Added dynamic LLVM configuration via llvm-config to gem5’s SCons build system, enabling seamless compilation and linking for datapath simulation

Methodology: Validating Functional Equivalence

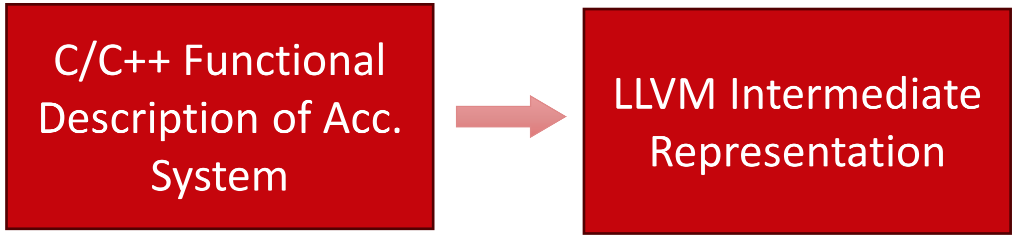
- **Validated outputs** and confirmed functional equivalence
  - Ensured compliance with gem5’s pre-commit and full regression test suite
  - Adapted SALAM’s system validation tests; cross-validated outputs against baseline to confirm functional equivalence

The Integrated Framework: What it Enables

- **Broader heterogeneity studies**
  - Co-simulate CPUs, GPUs, and custom accelerators.
  - Realistic modeling of heterogeneous systems with diverse compute elements and interactions.
- **System-level exploration**
  - Compare static vs dynamic accelerator scheduling, shared vs private local memories.
  - Evaluate placement, offloading, and sync strategies.
- **Domain-specific workload support**
  - Use built-in benchmark suite for architectural studies.
  - Model and study new workloads of interest.



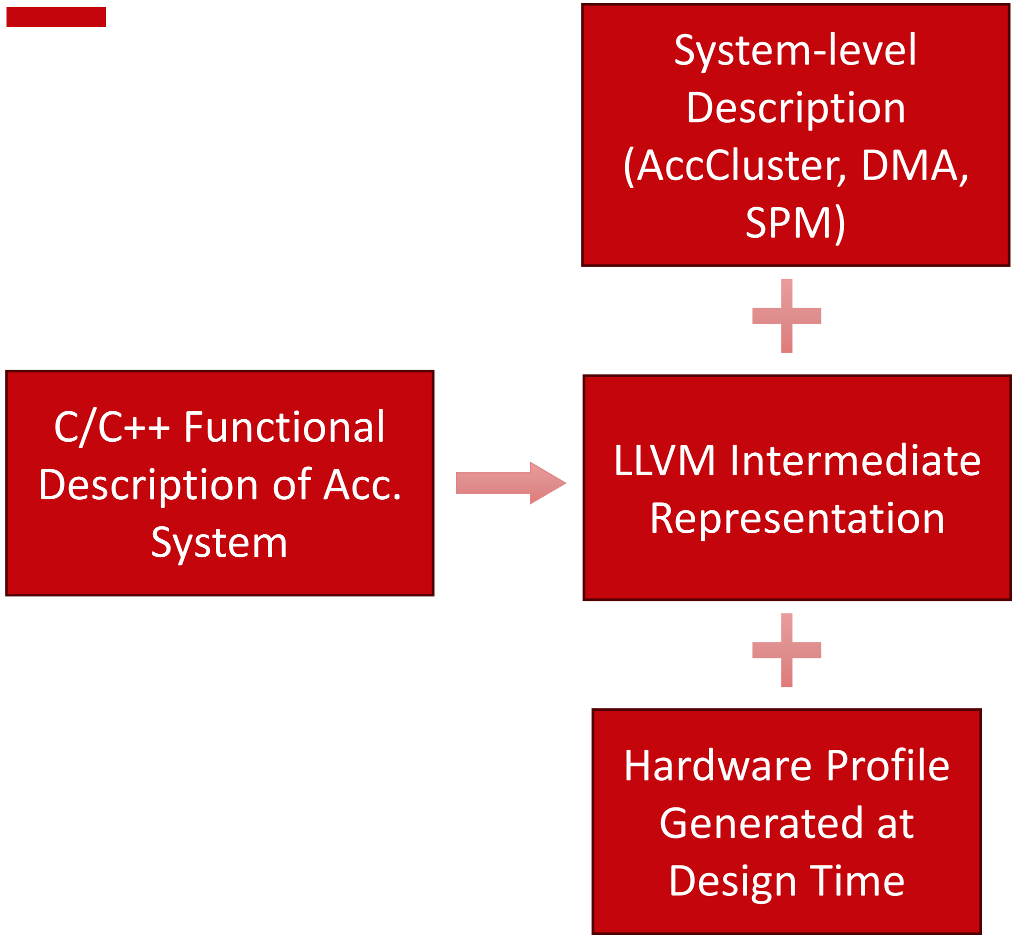
Modeling and Simulating an Accelerator



C/C++ Functional Description of Acc. System

LLVM Intermediate Representation

Modeling and Simulating an Accelerator



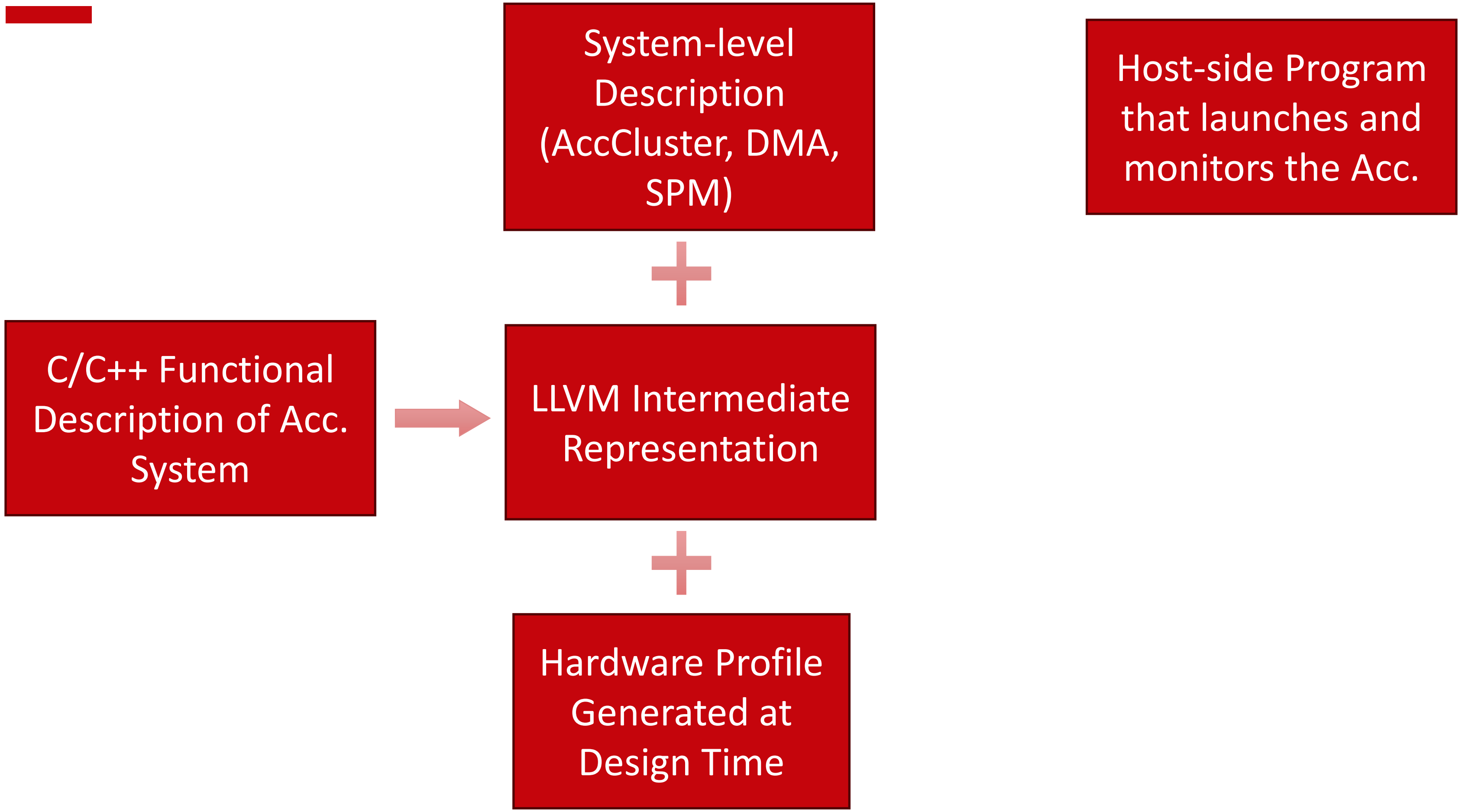
System-level Description (AccCluster, DMA, SPM)

C/C++ Functional Description of Acc. System

LLVM Intermediate Representation

Hardware Profile Generated at Design Time

Modeling and Simulating an Accelerator



System-level Description (AccCluster, DMA, SPM)

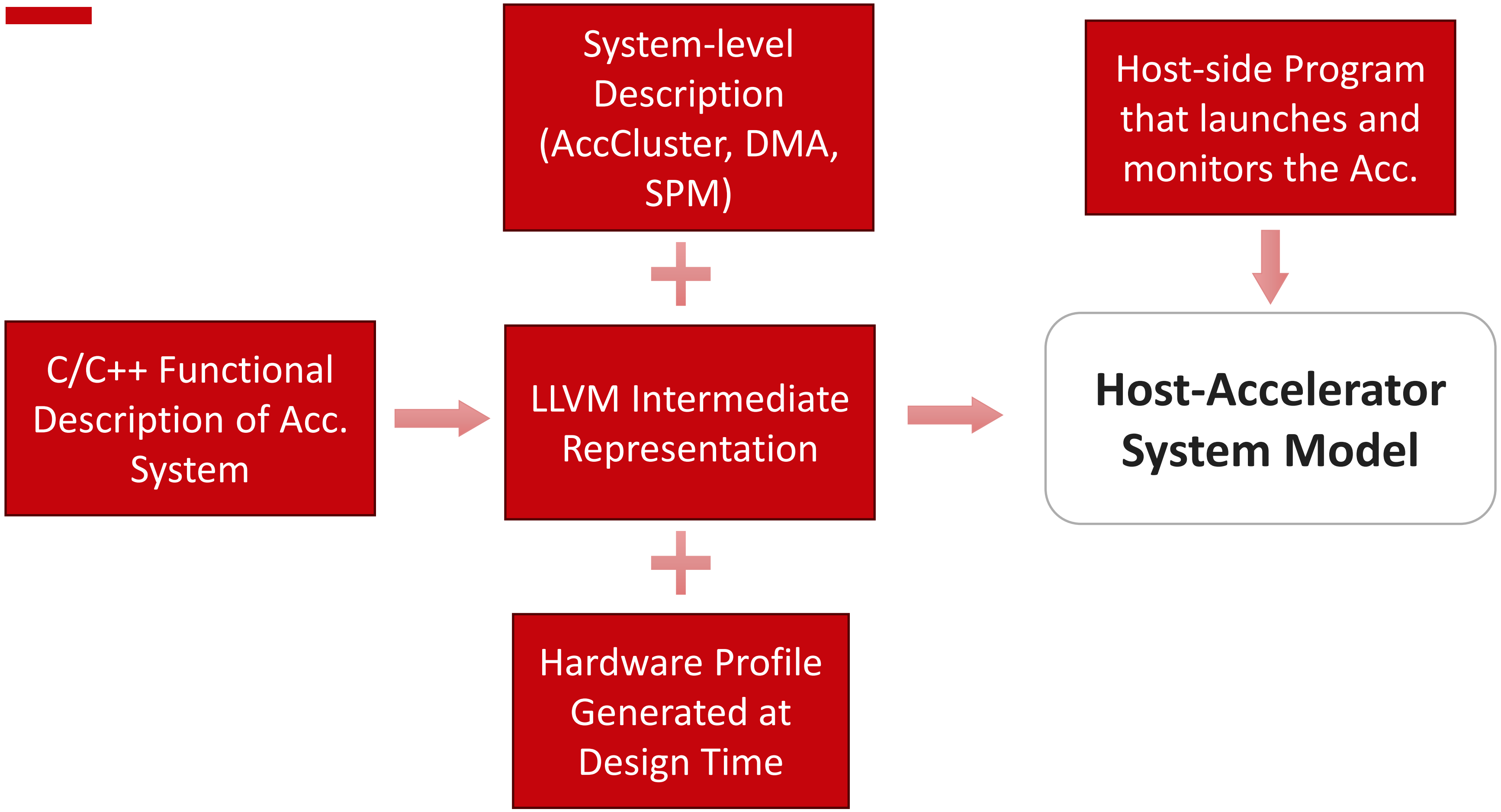
Host-side Program that launches and monitors the Acc.

C/C++ Functional Description of Acc. System

LLVM Intermediate Representation

Hardware Profile Generated at Design Time

Modeling and Simulating an Accelerator



System-level Description (AccCluster, DMA, SPM)

Host-side Program that launches and monitors the Acc.

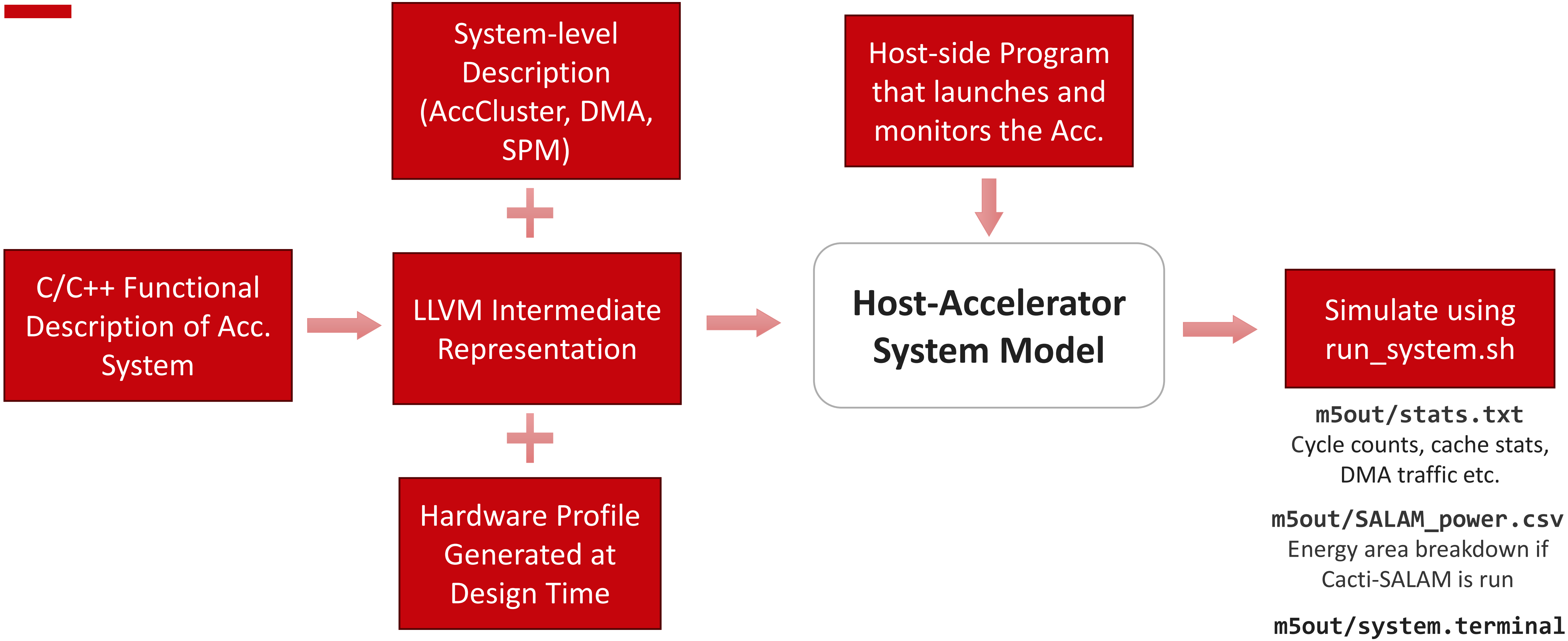
C/C++ Functional Description of Acc. System

LLVM Intermediate Representation

Host-Accelerator System Model

Hardware Profile Generated at Design Time

Modeling and Simulating an Accelerator



C/C++ Functional Description of Acc. System

System-level Description (AccCluster, DMA, SPM)

LLVM Intermediate Representation

Hardware Profile Generated at Design Time

Host-side Program that launches and monitors the Acc.

Host-Accelerator System Model

Simulate using run_system.sh

`m5out/stats.txt`
Cycle counts, cache stats, DMA traffic etc.

`m5out/SALAM_power.csv`
Energy area breakdown if Cacti-SALAM is run

`m5out/system.terminal`
Console prints from the host software

Outline

- Introduction
- Background
- Integration Methodology
- Frequency Scaling Study
- Results and Analysis
- Conclusions and Future Work

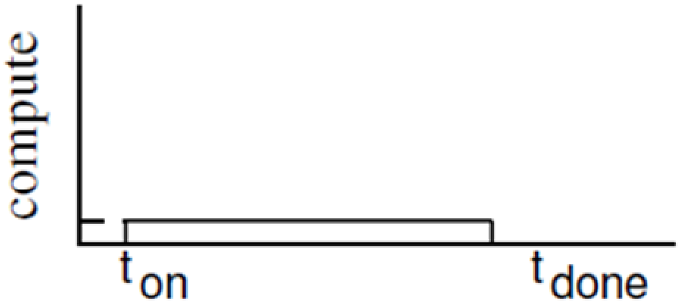
Case Study: Extreme Frequency Scaling

- Prior studies suggest accelerators are viable candidates for many GHz-scale execution
- Advanced cooling enables transient high-frequency operation (“computational sprinting”).

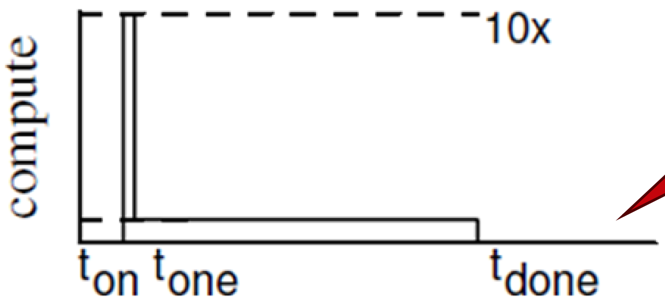
Source: Raghavan et al, HPCA 2012

Case Study: Extreme Frequency Scaling

- Prior studies suggest accelerators are viable candidates for multi GHz-scale execution
- Advanced cooling enables transient high-frequency operation (“computational sprinting”).



Normal Mode Operation



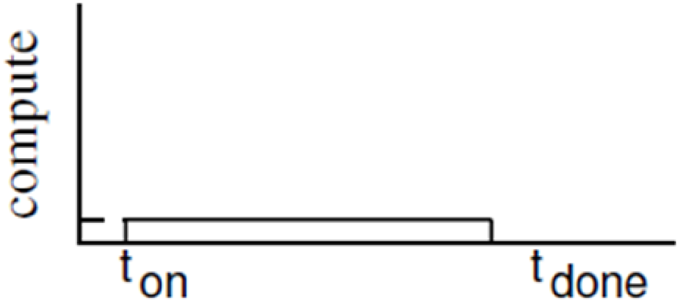
High-Frequency Operation

Exceeds TDP; cannot be sustained for too long. No meaningful work completed

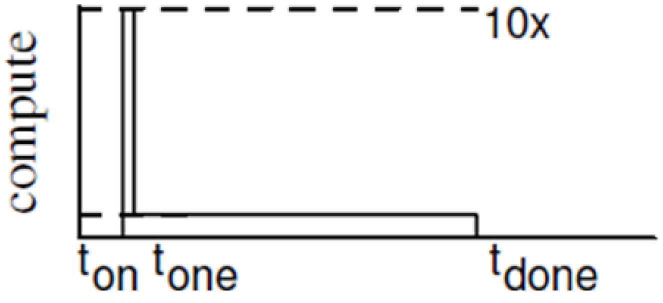
Source: Raghavan et al, HPCA 2012

Case Study: Extreme Frequency Scaling

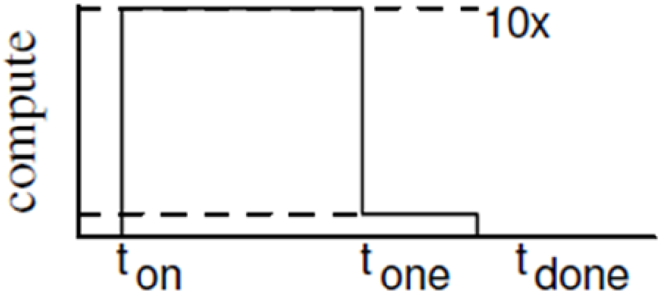
- Prior studies suggest accelerators are viable candidates for multi GHz-scale execution
- Advanced cooling enables transient high-frequency operation (“computational sprinting”).



Normal Mode Operation



High-Frequency Operation



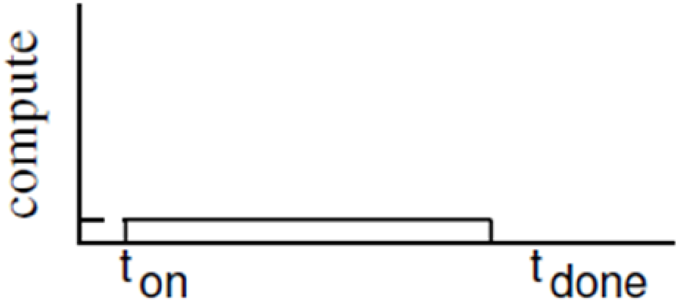
Cooled High-Frequency Operation

Cooling allows for longer “sprints” – making them actually beneficial

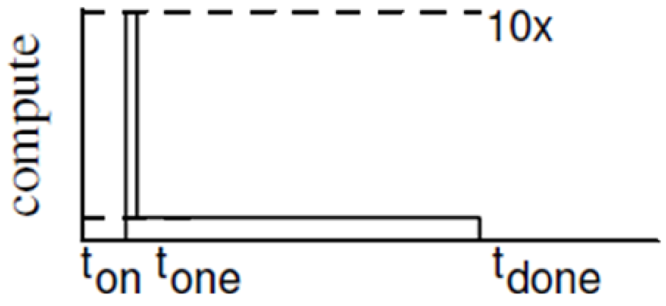
Source: Raghavan et al, HPCA 2012

Case Study: Extreme Frequency Scaling

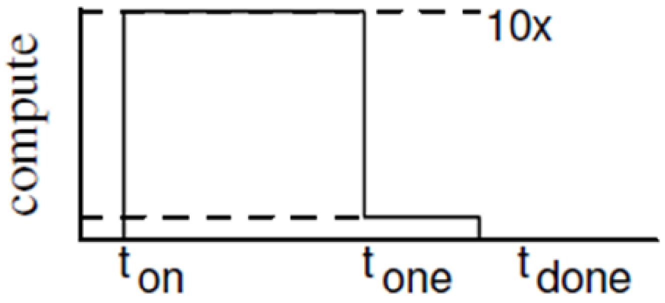
- Prior studies suggest accelerators are viable candidates for multi GHz-scale execution
- Advanced cooling enables transient high-frequency operation (“computational sprinting”).



Normal Mode Operation



High-Frequency Operation



Cooled High-Frequency Operation

- Use the integrated framework to evaluate accelerator performance up to 20 GHz

Source: Raghavan et al, HPCA 2012

Methodology

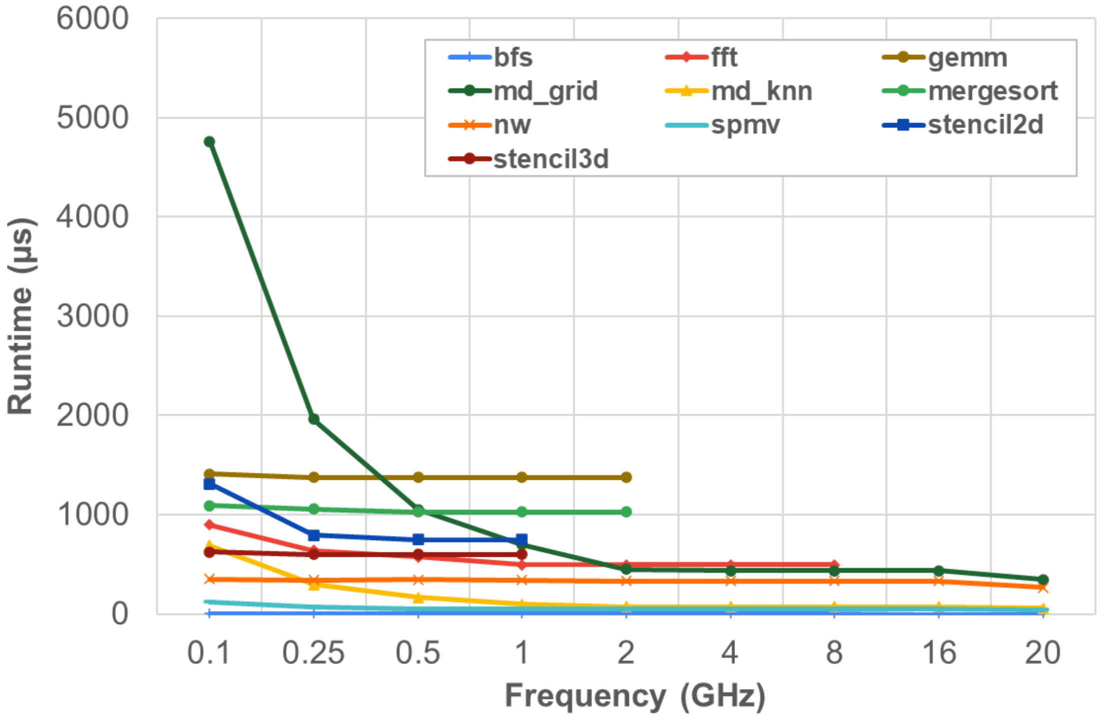
- Limit study to examine upper bound on performance and energy efficiency gains
  - Use small test inputs, configured to fit in accelerator scratchpad.
  - Isolates compute, eliminating any memory and interconnect bottlenecks
- Simulator Changes: Refined timing logic in LLVM Runtime, DMA, and interfaces to enable sub-cycle event resolution.
- Single accelerator instantiated with ARM DerivO3 CPU as host system.
- Sweep accelerator frequency from 0.1 to 20 GHz.
- Simulate MachSuite kernels to measure:
  - Runtime for Performance
  - Dynamic Power for Energy Efficiency

Outline

- Introduction
- Background
- Integration Methodology
- Frequency Scaling Study
- Results and Analysis
- Conclusions and Future Work

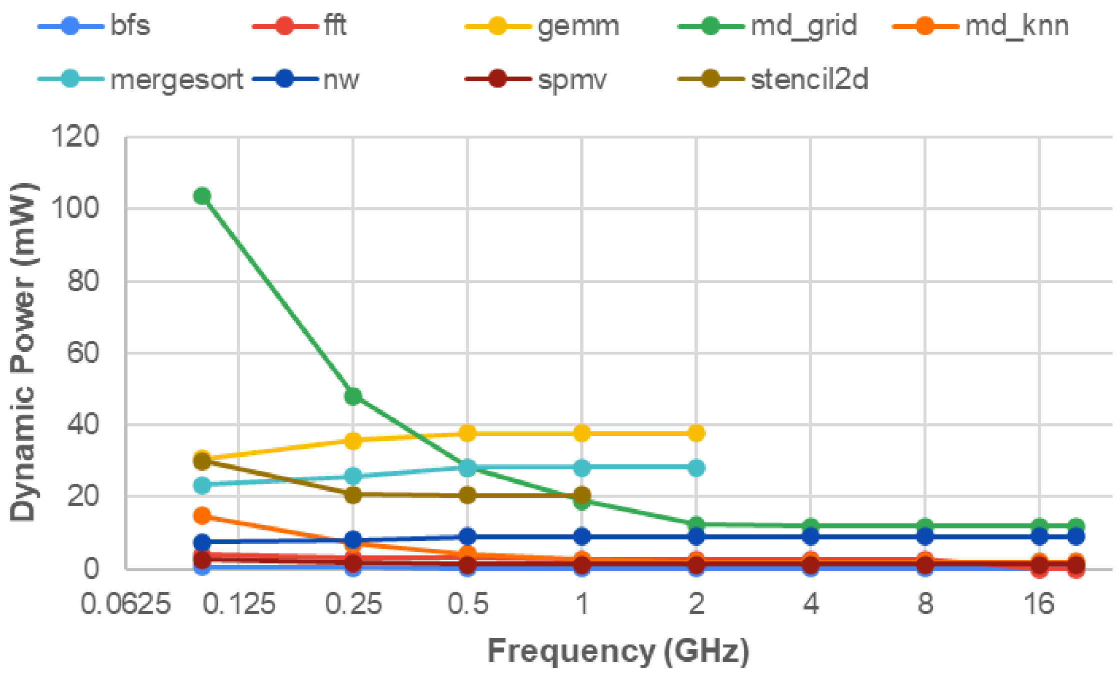
Preliminary Results: Performance

- **0.1 – 2 GHz: Perf. is compute-bound, frequency helps**.
  - Small input sizes limit utilization for some benchmarks, reducing observable gains
- **2 – 20 GHz: Performance plateaus**
  - Bottlenecks shift to CPU and DMA latency.
- **Takeaway:** Benefits of frequency scaling can be limited by both underutilization and system-level bottlenecks.



Preliminary Results: Dynamic Power

- **0.1 – 2 GHz: Power shaped by active time reduction**
  - Compute-bound kernels: Power decreases as faster execution shortens active duration; idle time dominates the average.
  - Others: Power stable/slightly rising; active time is non-negligible.
- **2 – 20 GHz: Power converges across all benchmarks**
  - Active time becomes negligible, idle state dominates.
- **Takeaway:** Time-averaged power scaling is limited by shorter active phases and increasing idle time at higher frequencies.



Outline

- Introduction
- Background
- Integration Methodology
- Frequency Scaling Study
- Results and Analysis
- Conclusions and Future Work

Conclusions and Future Work

- Modern and future systems increasingly embracing heterogeneity
- But tools are struggling to keep pace with the needs of these heterogeneous systems
- Solution: create unified full-system simulation framework for heterogeneous SoCs
- Case Study: high-frequency simulation for accelerators → shows promise
- **Integrating support into gem5 mainline (on-going)**
- **Benchmark suite expansion**
  - Add domain-specific workloads of interest to the SALAM resource set
- **Multi-ISA enablement**
  - Extend full-system support beyond ARM:
    - Route accelerator interrupts through ISA-specific interrupt controllers
    - Core-Local Interrupt Controller (CLIC) for RISC-V (working with NKAU)
  - Validate boot flow and benchmark functionality for new ISA