Fast and Effortless FPGA-accelerated Hardware Simulation with On-Prem and Cloud Flexibility

https://fires.im

@firesimproject

Speaker: Sagar Karandikar

Berkeley Architecture Research

The architect/chip-developer's design flow

1. High-level Simulation
2. Write RTL + Software, plug into your favorite ecosystem (e.g. Chipyard)
3. Co-design in software RTL sim (e.g. Verilator, VCS, etc.)
   - Run microbenchmarks
4. Co-design in FPGA-accelerated simulation
   - Boot an OS and run the complete software stack,
     obtain realistic performance measurements
5. Tapeout → Chip
   - Boot OS and run applications, but no more opportunity for co-design

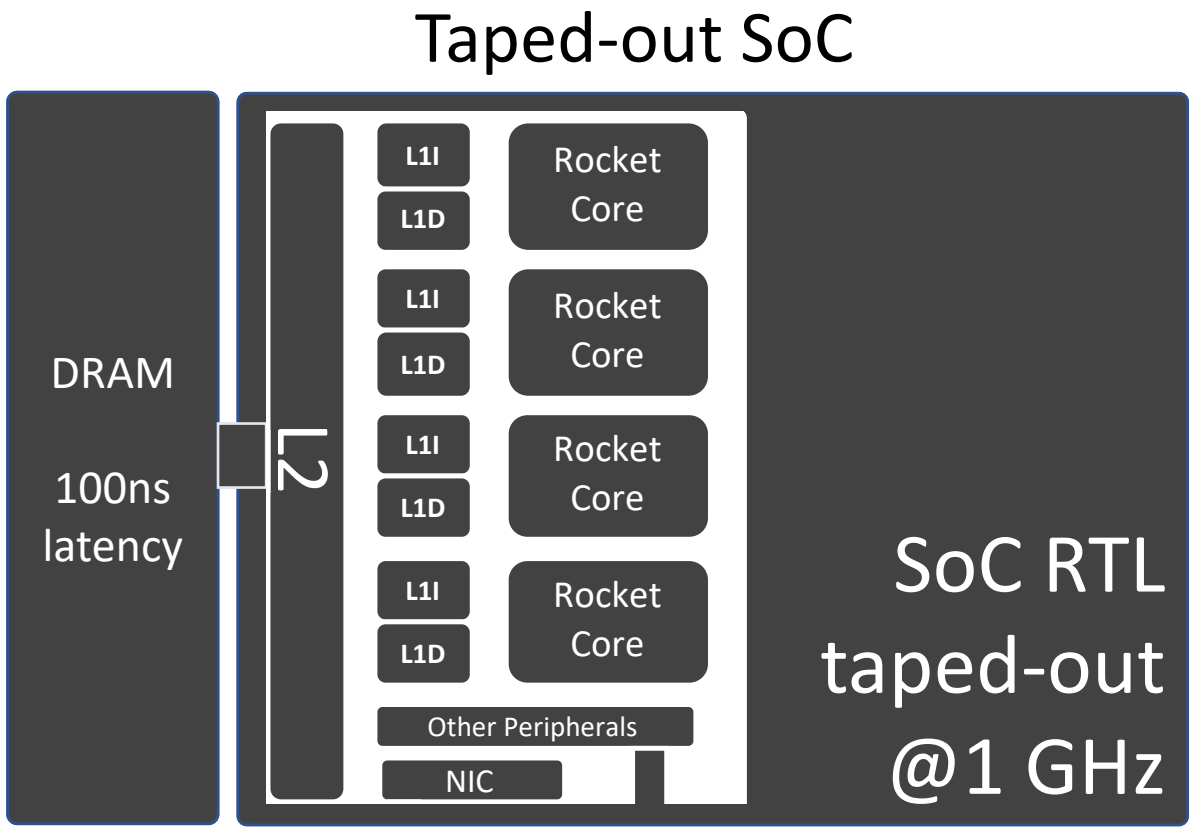
The architect/chip-developer's design flow

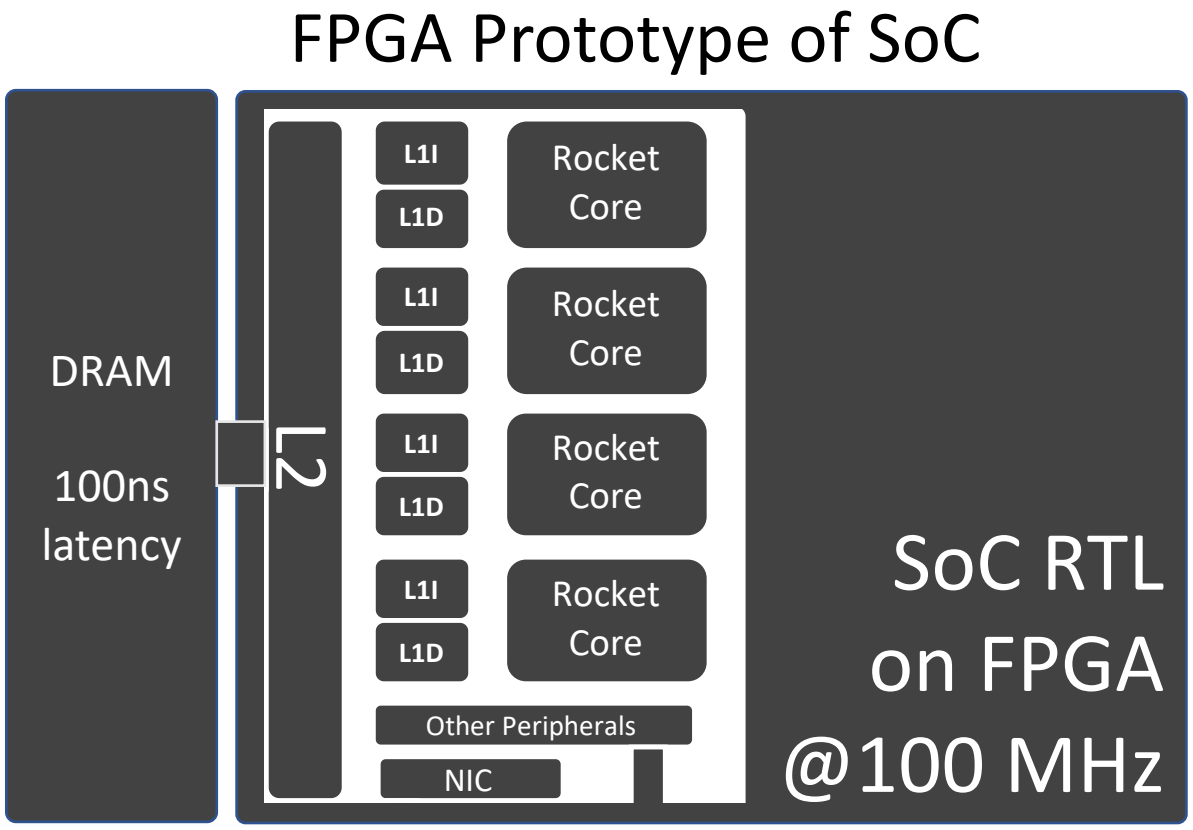
1. High-level Simulation
2. Write RTL + Software, plug into your favorite ecosystem (e.g. Chipyard)
3. Co-design in software RTL sim (e.g. Verilator, VCS, etc.)
   - Run microbenchmarks
4. **Co-design in FPGA-accelerated simulation**
   - **Boot an OS and run the complete software stack, obtain realistic performance measurements**

FireSim

5. Tapeout → Chip
   - Boot OS and run applications, but no more opportunity for co-design

What about FPGA prototyping?





SoC sees 100 cycle DRAM latency

SoC sees 10 cycle DRAM latency

Incorrect by a factor of 10!

Difficulties with FPGA Prototypes

In an FPGA prototype:

- Every FPGA clock executes one cycle of the simulated target
- Performance of FPGA-attached resources is exposed to the simulated world, e.g. DRAM, SD Card, UART, Ethernet, etc.

This leads to three problems:

1) Incorrect performance modeling: FPGA resources probably not an accurate representation of target system
   a) E.g., DRAM performance off by ***10x*** on previous slide
2) Simulations are non-deterministic
3) Different host FPGAs produce different simulation results

Want HW simulators that:

- Are as fast as silicon
- Are as detailed as silicon
- Have all the benefits of SW-based simulators
- Are low-cost

Our Thesis:

- FPGAs are the only viable basis technology

→ Build *FPGA-accelerated* simulators with SW-like flexibility using an *open-source* tool

How? Useful Trends Throughout the Stack

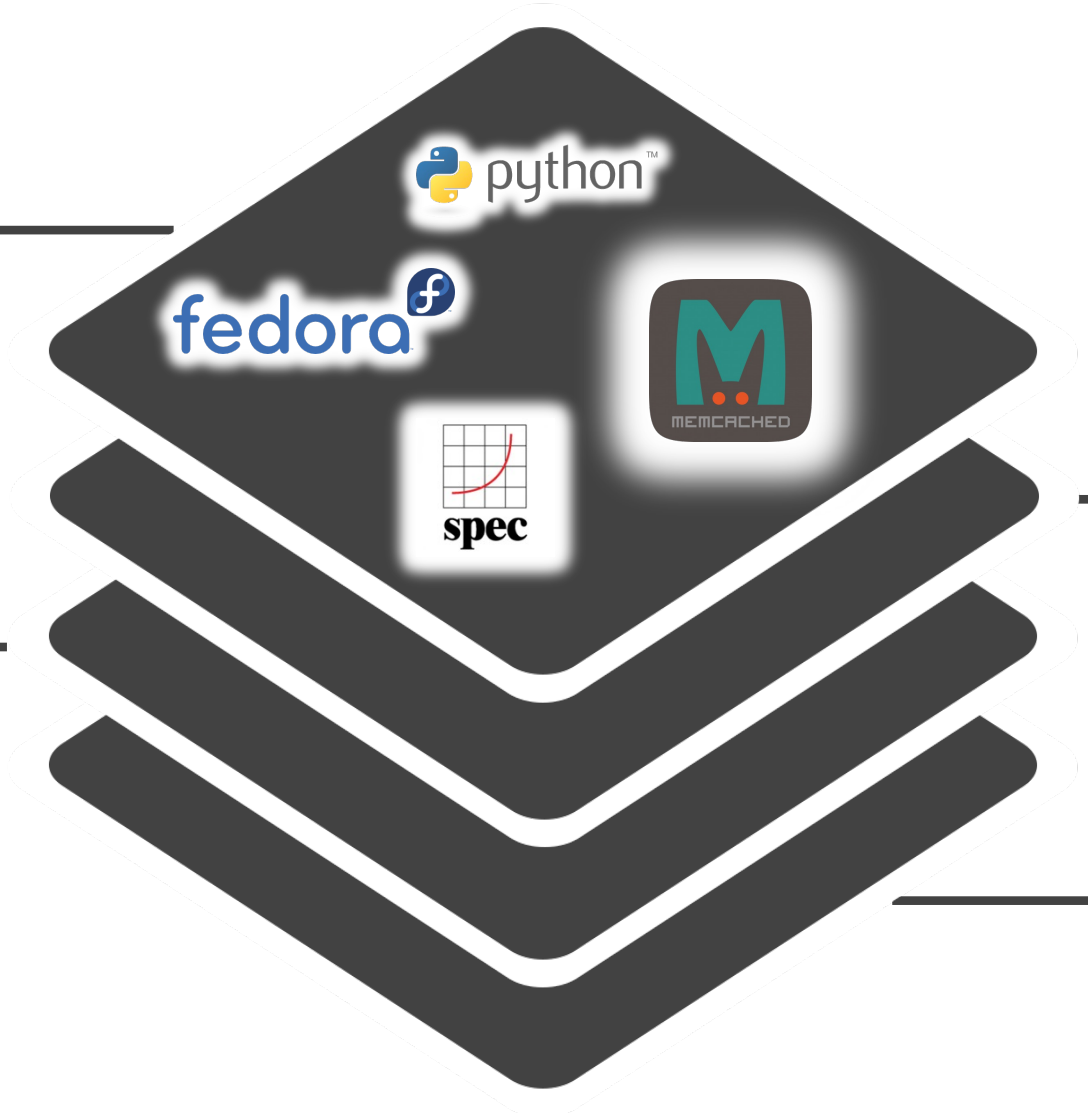
Open ISA

RISC-V

CHISEL

High-Productivity Hardware Design Language & IR

FIRRTL



Open, Silicon-Proven SoC Implementations

FPGAs in the Cloud

Amazon EC2 F1 Instances

Run Customizable FPGAs in the AWS Cloud

FireSim at 35,000 feet

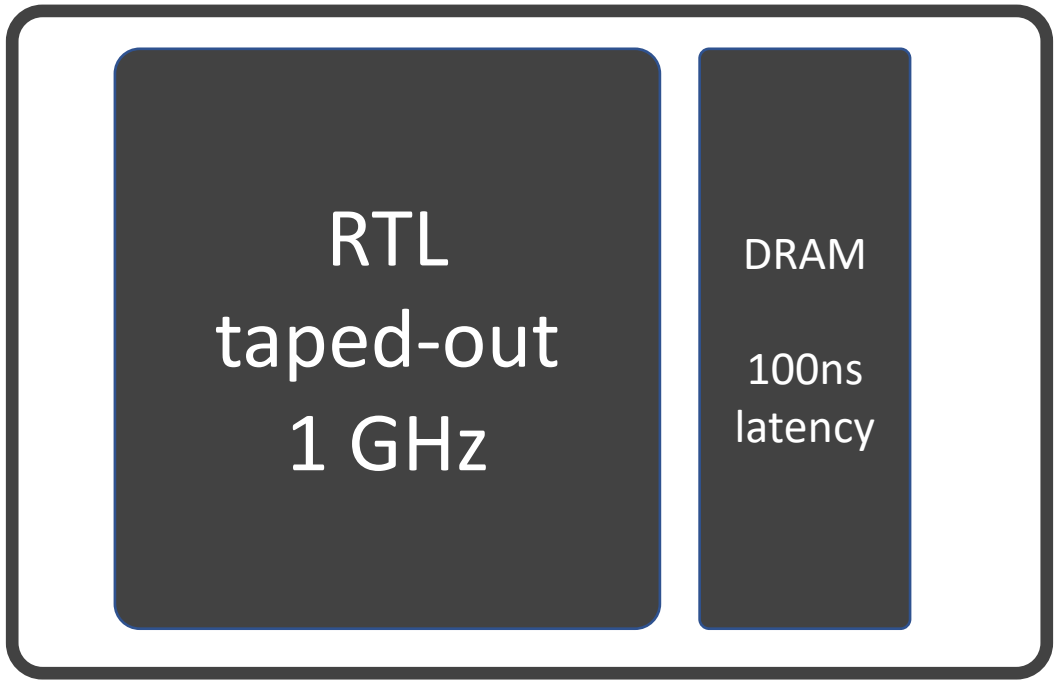
- Open-source, fast, automatic, deterministic FPGA-accelerated hardware simulation for pre-silicon verification and performance validation
- Ingests:
  - Your RTL design: FIRRTL (Chisel), blackbox Verilog
    - Or Chipyard-generated designs with Rocket Chip, BOOM, NVDLA, PicoRV32, and more
  - HW and/or SW IO models (e.g. UART, Ethernet, DRAM, etc.)
  - Workload descriptions
- Produces: Fast, cycle-exact simulation of your design + models around it
- Automatically deployed to on-prem or cloud FPGAs
  - E.g., Xilinx Alveo, VCU118, AWS EC2 F1, etc.

Three Distinguishing Features of FireSim

1) Not FPGA prototypes, rather FPGA-accelerated simulators
   - Automatic transformation of RTL designs into FPGA-accelerated simulators
   - Enables new debugging, resource optimization, and profiling capabilities
2) Flexible scaling from on-prem to cloud FPGAs
   - Scale easily from one or more on-prem FPGAs to massively parallel simulations on elastic supply of cloud FPGAs
   - Standardized host platforms = easy to collaborate with other researchers and perform artifact evaluation
   - Heavy automation to hide FPGA complexity, regardless of on-prem or cloud platform
3) Open-source (https://fires.im)

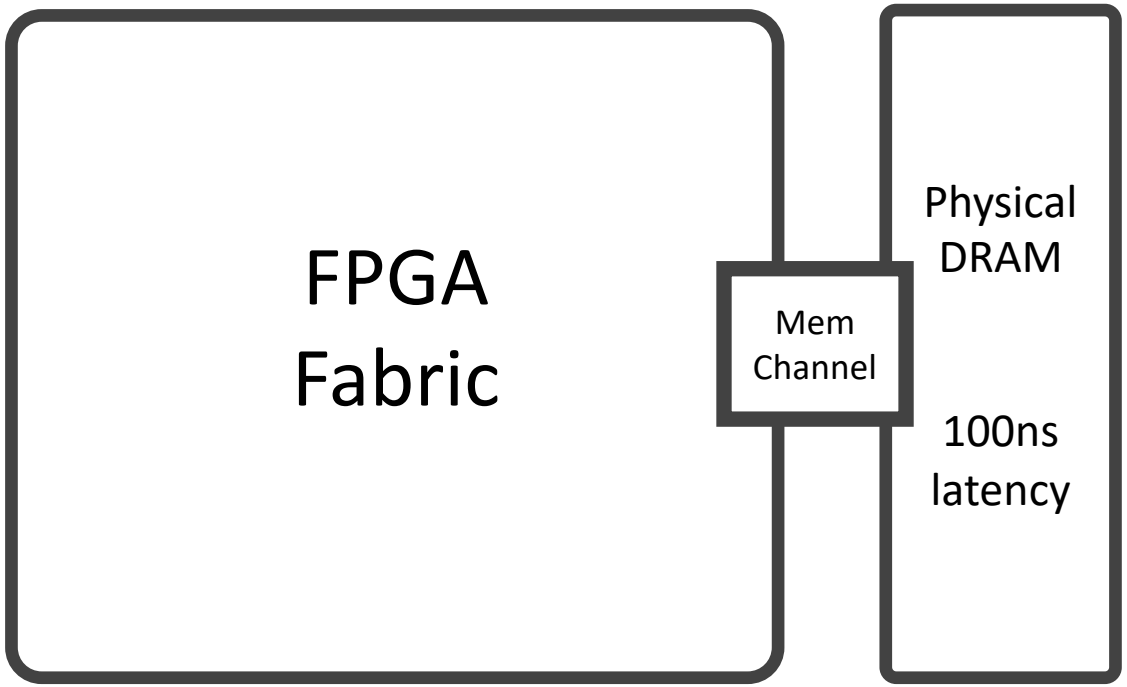
Separating Target and Host

Target: the machine under simulation



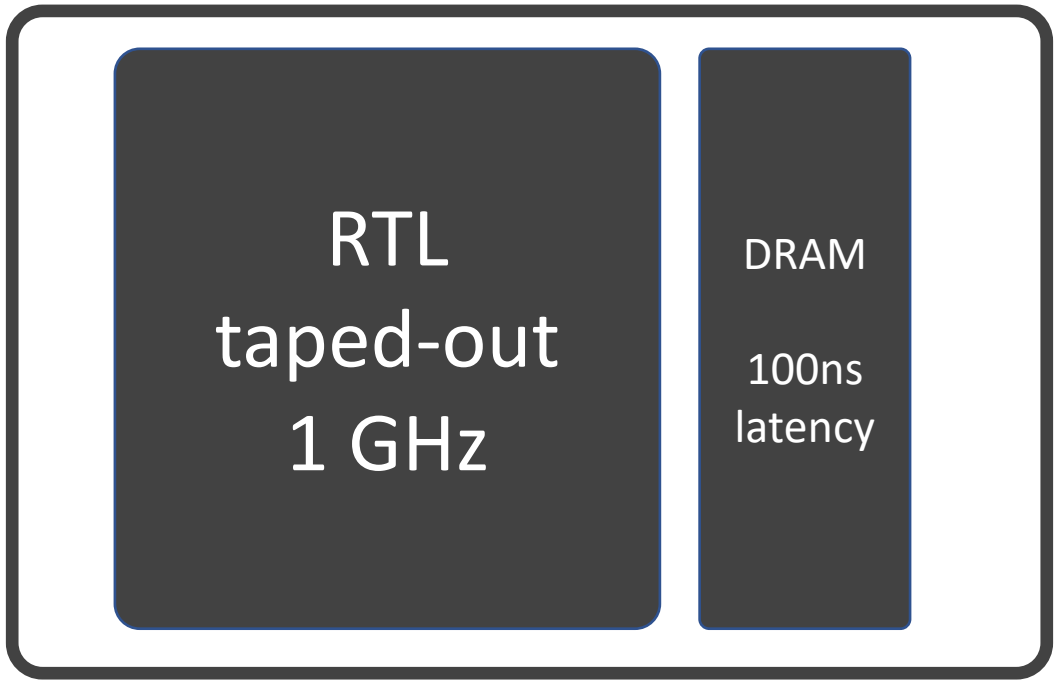
Closed simulation world.

Host: the machine executing (*hosting)* the simulation



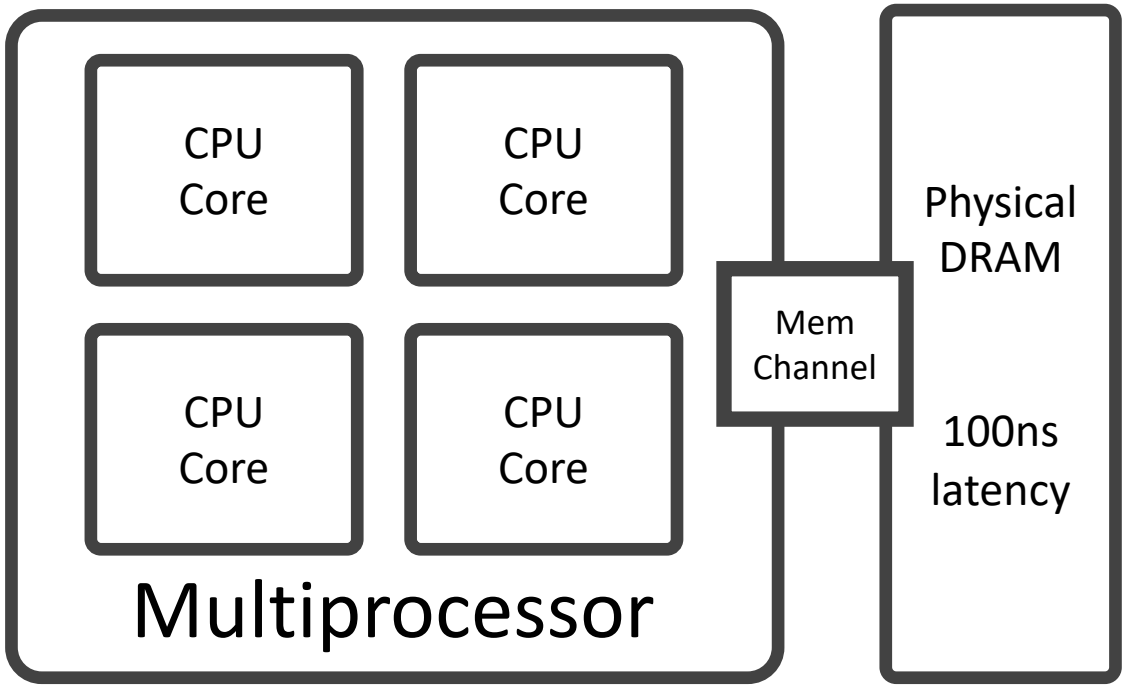
Separating Target and Host

Target: the machine under simulation

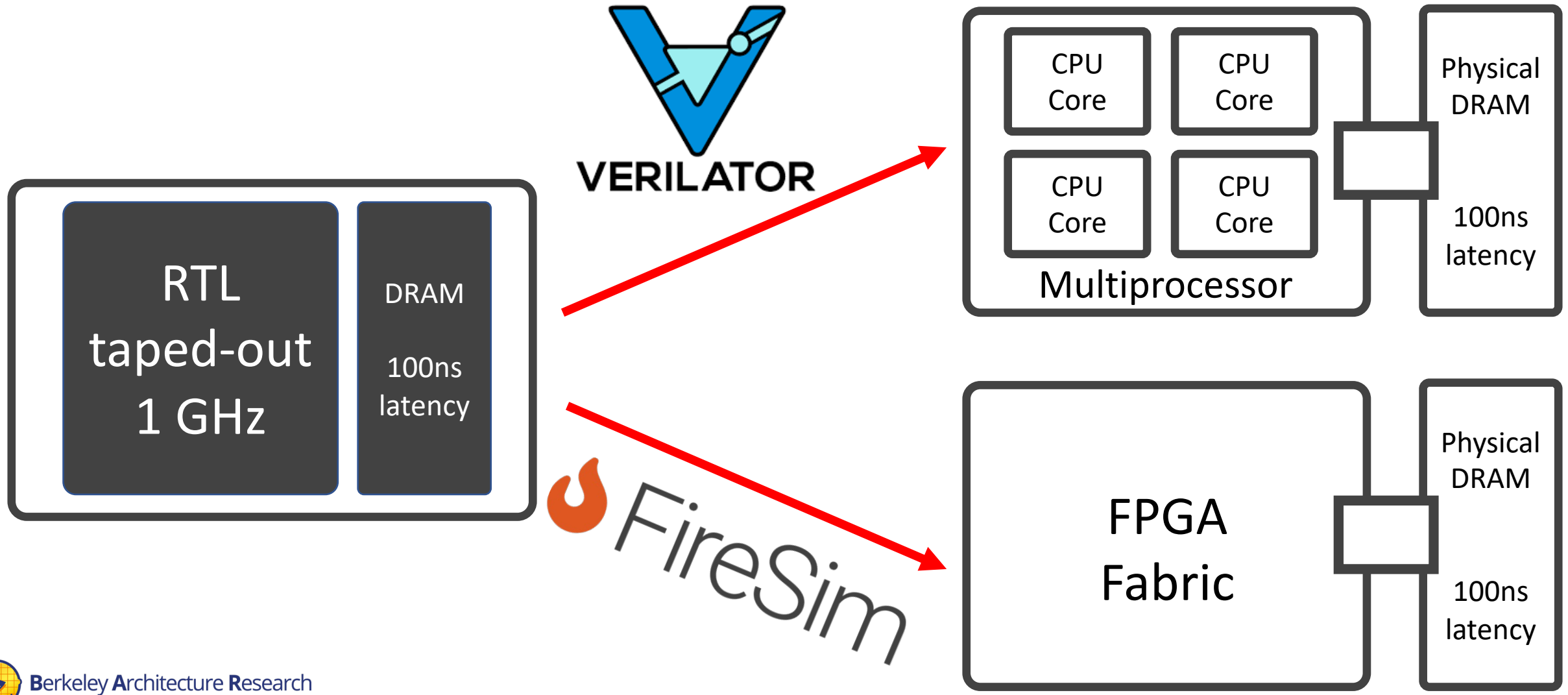


Closed simulation world.

Host: the machine executing (*hosting)* the simulation

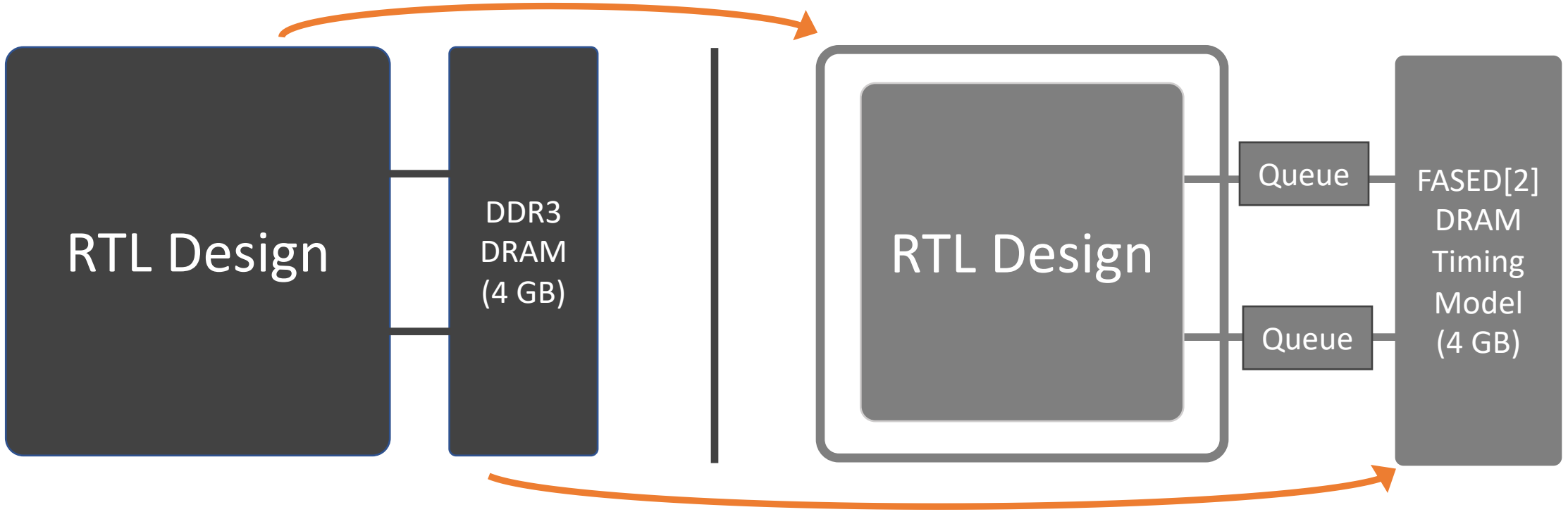


FireSim Generates FPGA-Hosted Simulators



Host Decoupling in FireSim: Transforming the Target

1) Convert RTL into a latency-insensitive [1] model using FIRRTL transform



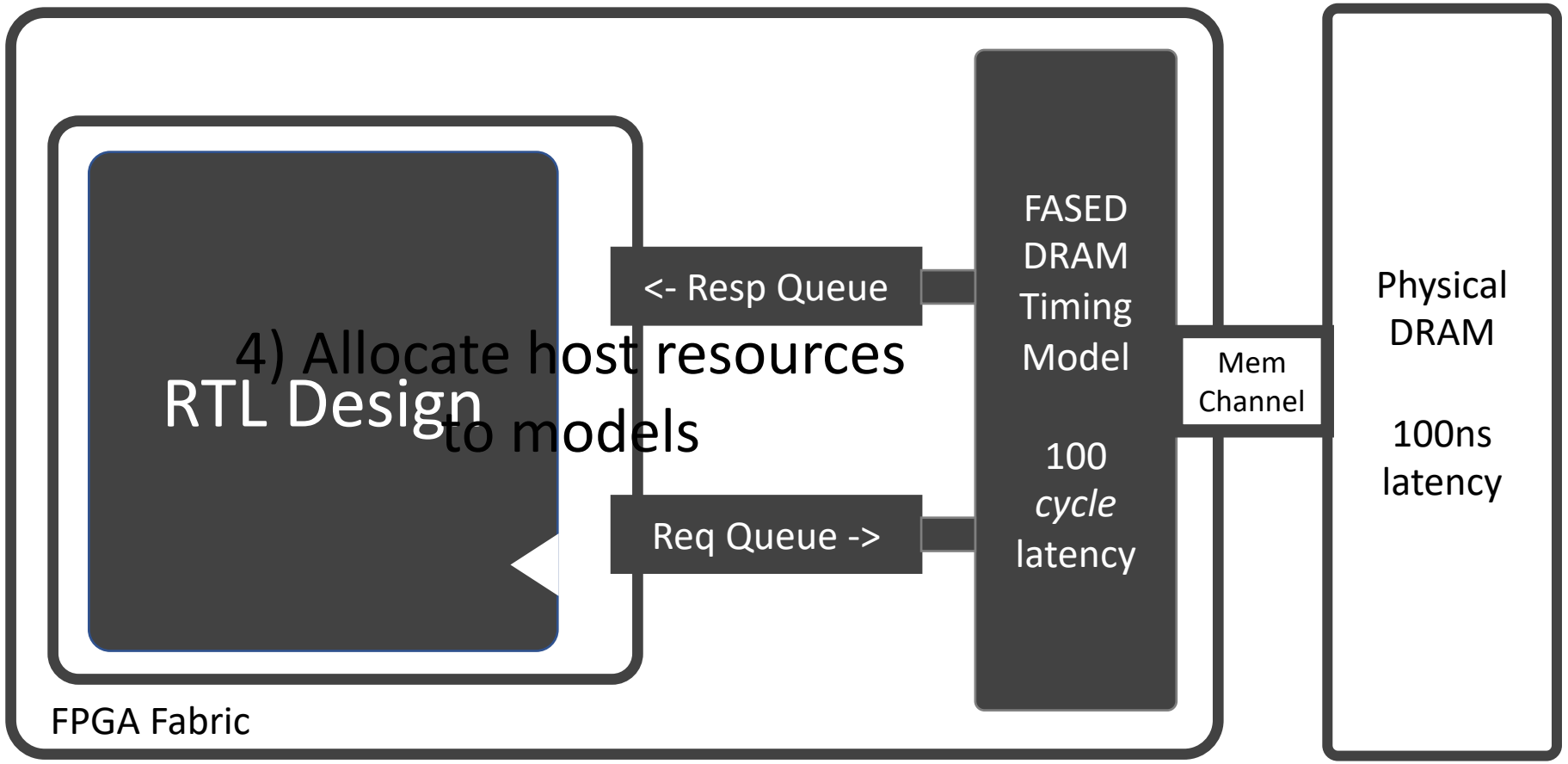
2) Generate FPGA-hosted model for DRAM [2] (think DRAMSim on an FPGA)

3) Generate queues (token channels) to connect the target models

[1] Theory of Latency Insensitive Design, Carloni et al, also see: RAMP

[2] FASED: FPGA-accelerated Simulation and Evaluation of DRAM, Biancolin et al

Host Decoupling in FireSim: Mapping to the FPGA



SoC sees realistic DRAM latency

Benefits of Host Decoupling on FPGAs

Simulations will now:

- Execute deterministically
- Produce identical results on different hosts (FPGAs & CPUs)

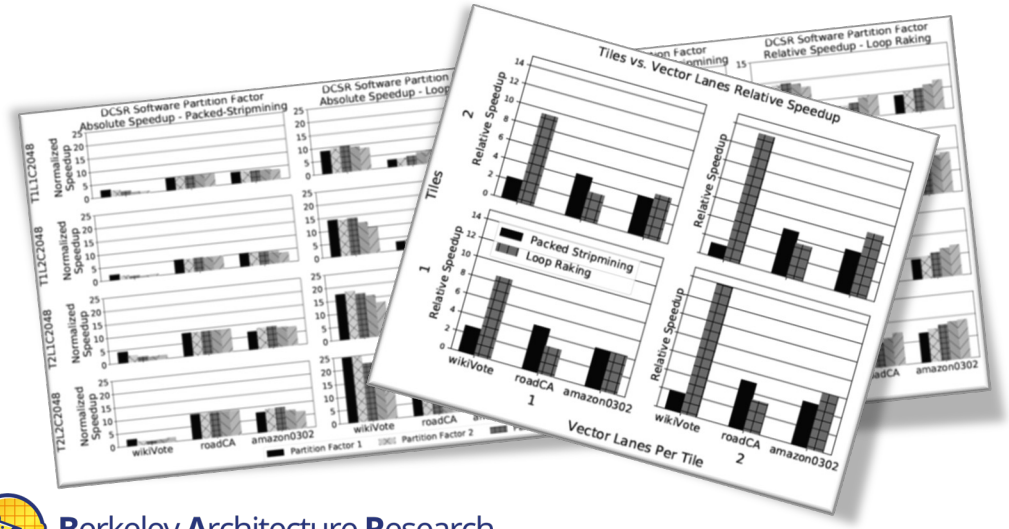
Decoupling enables support for:

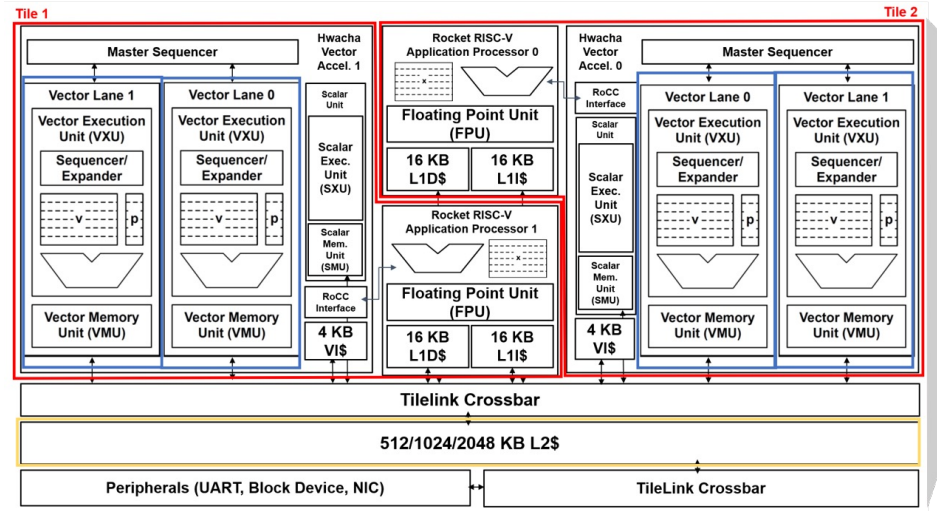
1. SW co-simulation (e.g. block device, network models)
2. Simulating large targets over distributed hosts (ISCA ‘18, Top Picks ‘18)
3. Non-invasive debugging and instrumentation (FPL ‘18, ASPLOS ’20, ASPLOS ‘23)
4. Multi-cycle resource optimizations (ICCAD ‘19)

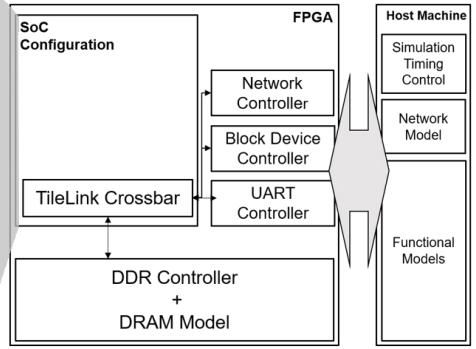
What Can You Do With FireSim?

Example use cases: Evaluating SoC Designs

- "Classical" Performance Measurement
  - Run SPECint 2017 with full reference inputs on Rocket Chip in parallel on ~10 FPGAs within a day (e.g., in D. Biancolin, et. al., *FASED*, FPGA '19)
- Rapid Full-System Design Space Exploration
  - Can rapidly sweep parameter space of a design with FireSim automation
  - Data-parallel accelerators (Hwacha) and multi-core processors
  - Complex software stacks (Linux, OpenMP, GraphMat, Caffe)

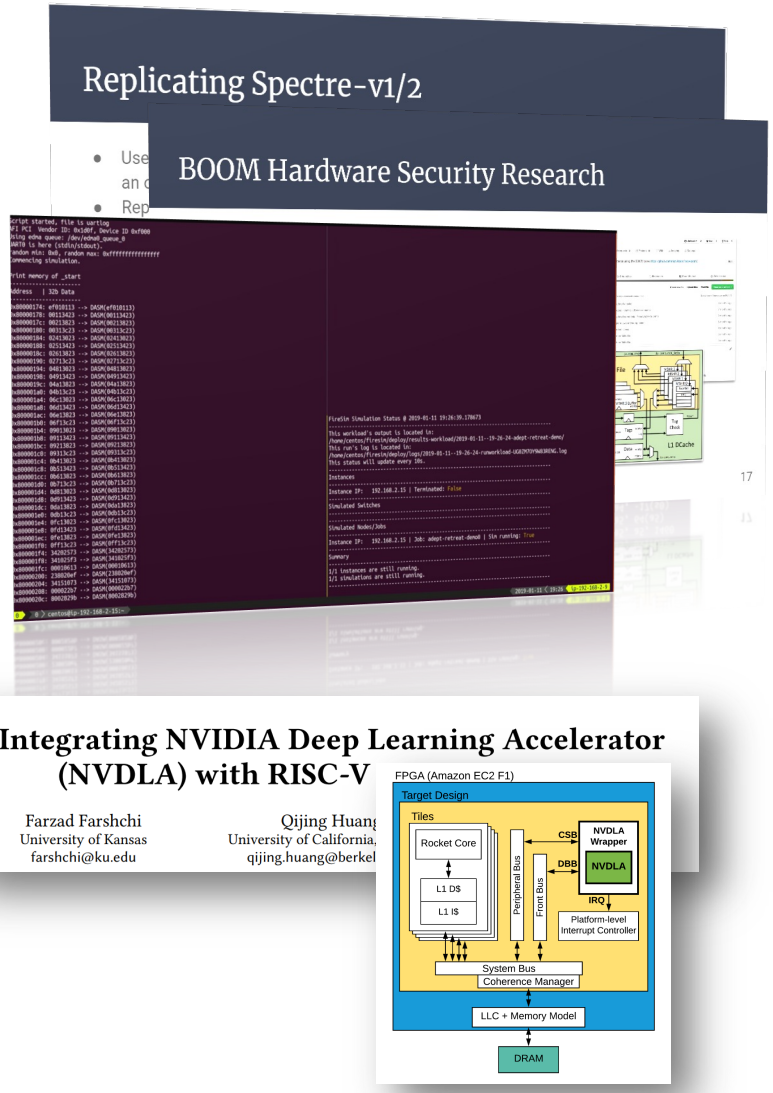






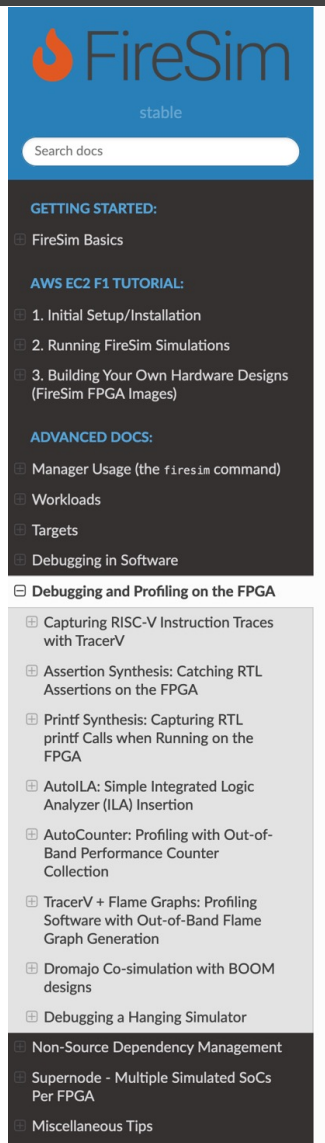
Example use cases: Evaluating SoC Designs

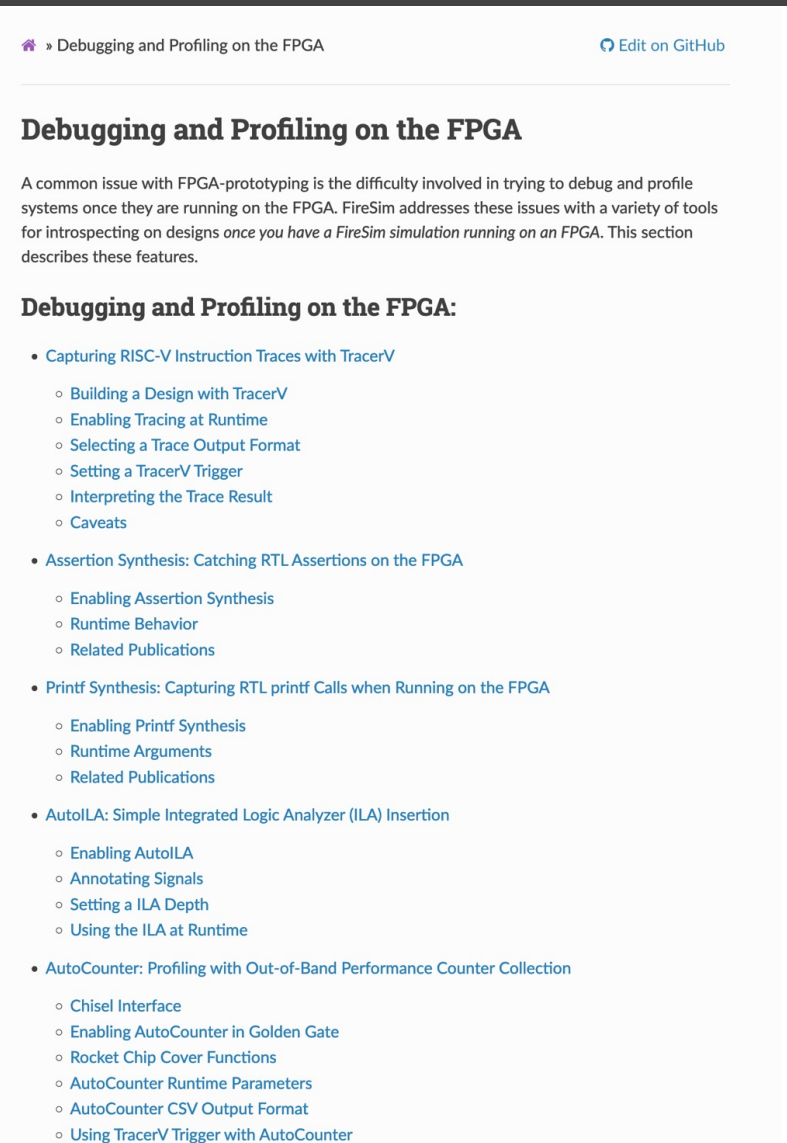
- Security:
  - BOOM Spectre replication
    - A. Gonzalez, et. al., *Replicating and Mitigating Spectre Attacks on an Open Source RISC-V Microarchitecture,* CARRV '19
  - Keystone Enclave performance evaluation
    - D. Lee, et. al., *Keystone*, EuroSys '20
- Accelerator evaluation
  - Chisel-based accelerators:
    - Machine learning (H. Genc, et. al., *Gemmini*, DAC 2021)
    - Garbage collection (M. Maas, et. al., *A Hardware Accelerator for Tracing Garbage Collection,* ISCA '18)
  - Integrating Verilog-based accelerators:
    - NVDLA (F. Farshchi, et. al. *Integrating NVIDIA Deep Learning Accelerator (NVDLA) with RISC-V SoC on FireSim*. EMC2 '19)
    - HLS-based rapid prototyping (Q. Huang, et. al., *Centrifuge*, ICCAD '19)
  - Scale-out accelerators
    - nanoPU NIC-CPU co-design (S. Ibanez, et. al., *nanoPU*, OSDI '21)
    - Protobuf Accelerator (S. Karandikar, et. al., *A Hardware Accelerator for Protocol Buffers*, MICRO '21. MICRO-54 Distinguished Artifact Winner.)



Example use cases: Debugging and Profiling SoC Designs

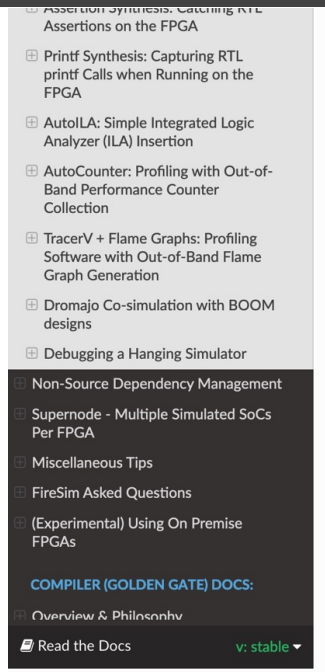
- Debugging a Chisel design at FPGA-speeds
  - Many FireSim debugging features: Assertion synthesis, printf synthesis, ILA insertion, etc.
  - e.g. FireSim Debugging Docs

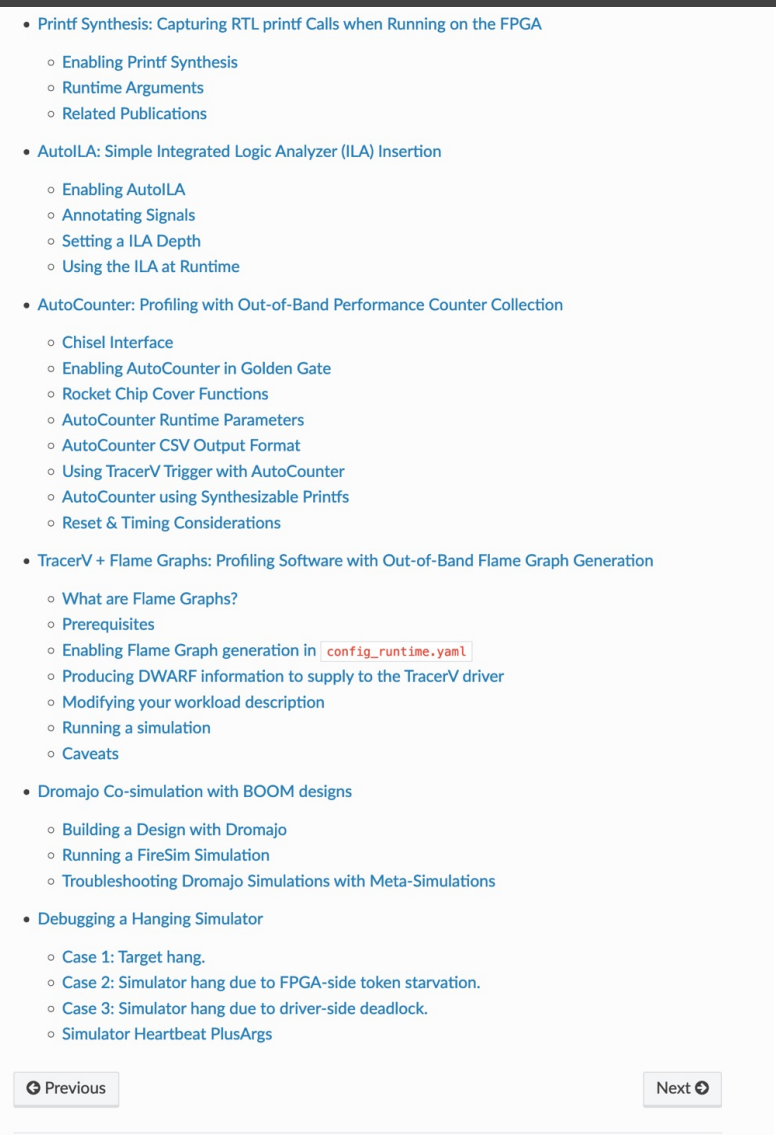




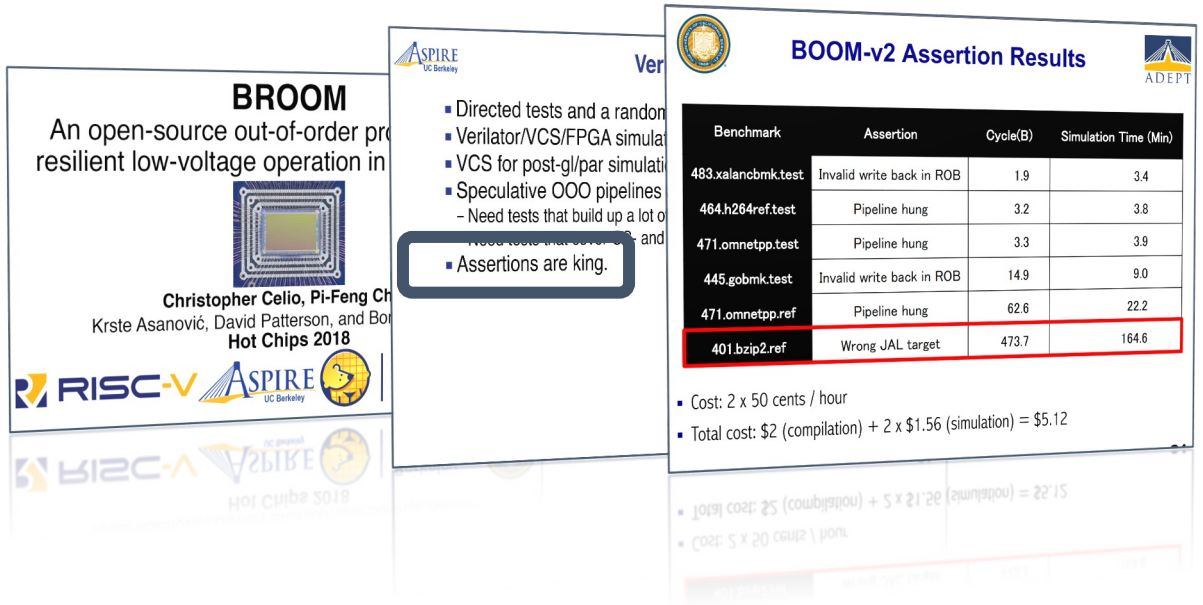
Example use cases: Debugging and Profiling SoC Designs

- Debugging a Chisel design at FPGA-speeds
  - Many FireSim debugging features: Assertion synthesis, printf synthesis, ILA insertion, etc.
  - e.g. FireSim Debugging Docs





Example use cases: Debugging and Profiling SoC Designs



- Debugging a Chisel design at FPGA-speeds
  - Many FireSim debugging features: Assertion synthesis, printf synthesis, ILA insertion, etc.
  - e.g. FireSim Debugging Docs
  - e.g. Fixing BOOM Bugs (D. Kim, et. al., *DESSERT*, FPL '18)
- Profiling a custom RISC-V SoC at FPGA-speeds
  - e.g. HW/SW Co-design of a networked RISC-V system (S. Karandikar, et. al., *FirePerf*, ASPLOS 2020)

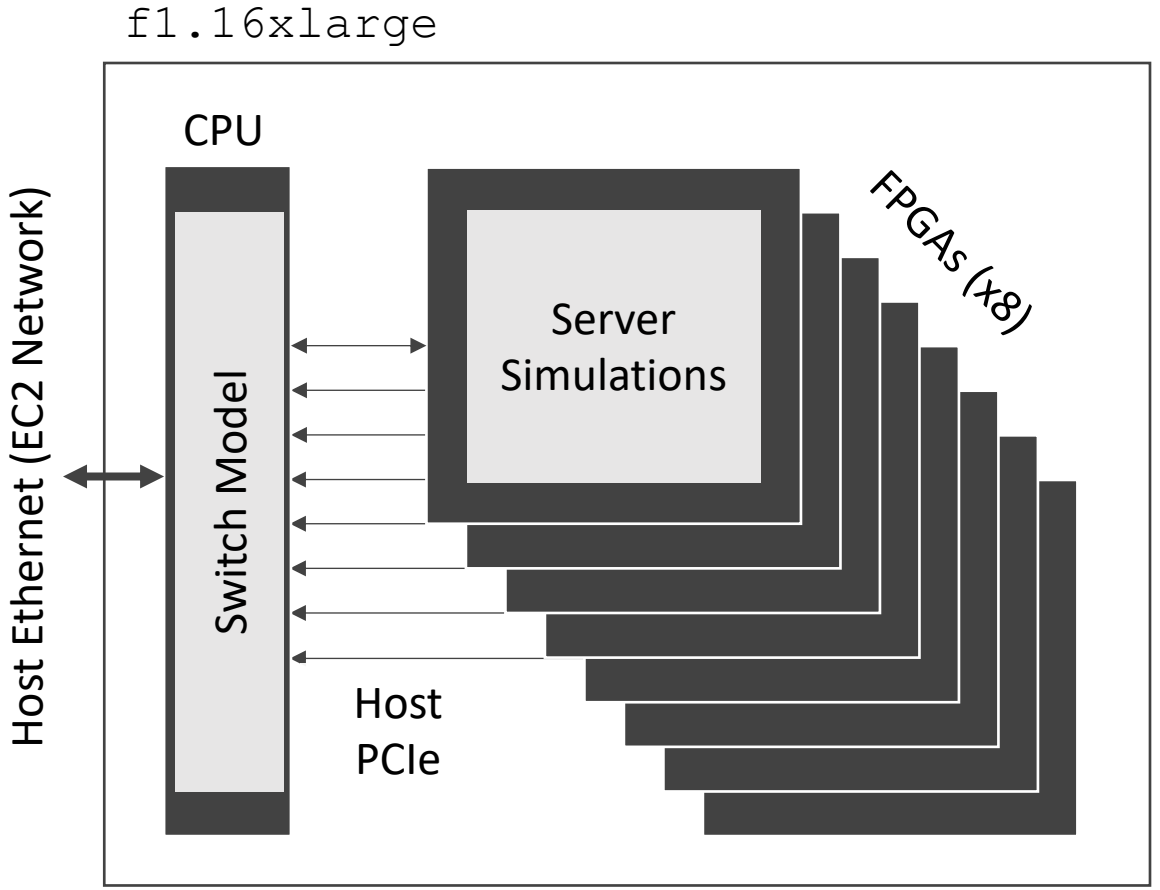
FirePerf

How-to-build a *datacenter-scale* FireSim simulation

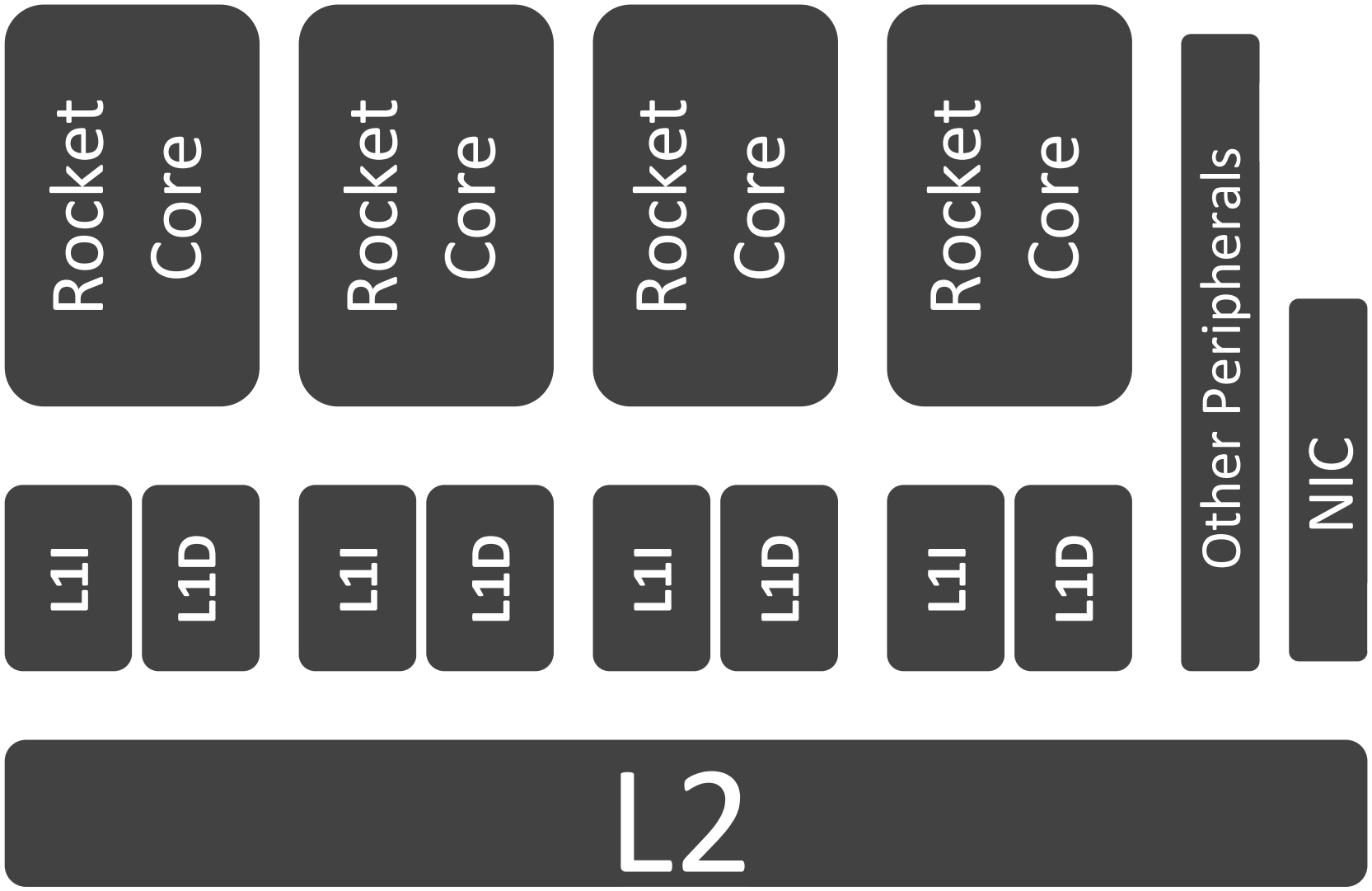
[1] S. Karandikar et. al., “FireSim: FPGA-Accelerated Cycle-Exact Scale-Out System Simulation in the Public Cloud.” *ISCA 2018*
[2] S. Karandikar et. al., “FireSim: FPGA-Accelerated Cycle-Exact Scale-Out System Simulation in the Public Cloud.” *IEEE Micro Top Picks 2018*

Mapping a datacenter simulation

- DC simulation requires:
  - Model hardware at scale, cycle-accurately
  - Run real software
- RTL and abstract SW model co-simulation
- Server Simulations
  - Good fit for the FPGA
  - We have tapeout-proven RTL: FAME-1 transform w/Golden-Gate
- Network simulation
  - Little parallelism in switch models (e.g. a thread per port)
  - Need to coordinate all the distributed server simulations
  - So use CPUs + host network



Step 1: Server SoC in RTL



Modeled System

- 4x RISC-V Rocket Cores @ 3.2 GHz
- 16K I/D L1$
- 256K Shared L2$
- 200 Gb/s Eth. NIC

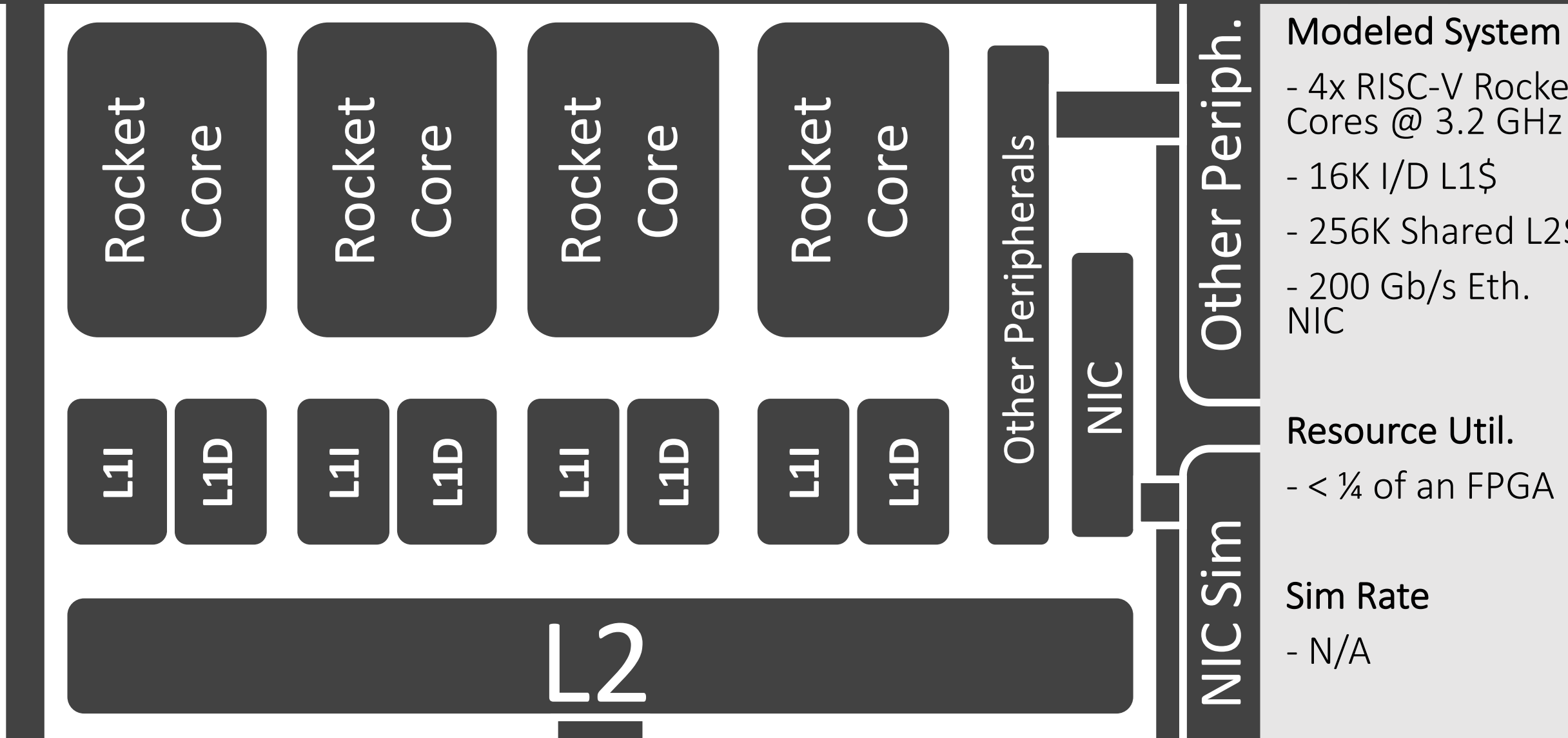
Resource Util.

- < ¼ of an FPGA

Sim Rate

- N/A

Step 1: Server SoC in RTL



Modeled System

- 4x RISC-V Rocket Cores @ 3.2 GHz
- 16K I/D L1$
- 256K Shared L2$
- 200 Gb/s Eth. NIC

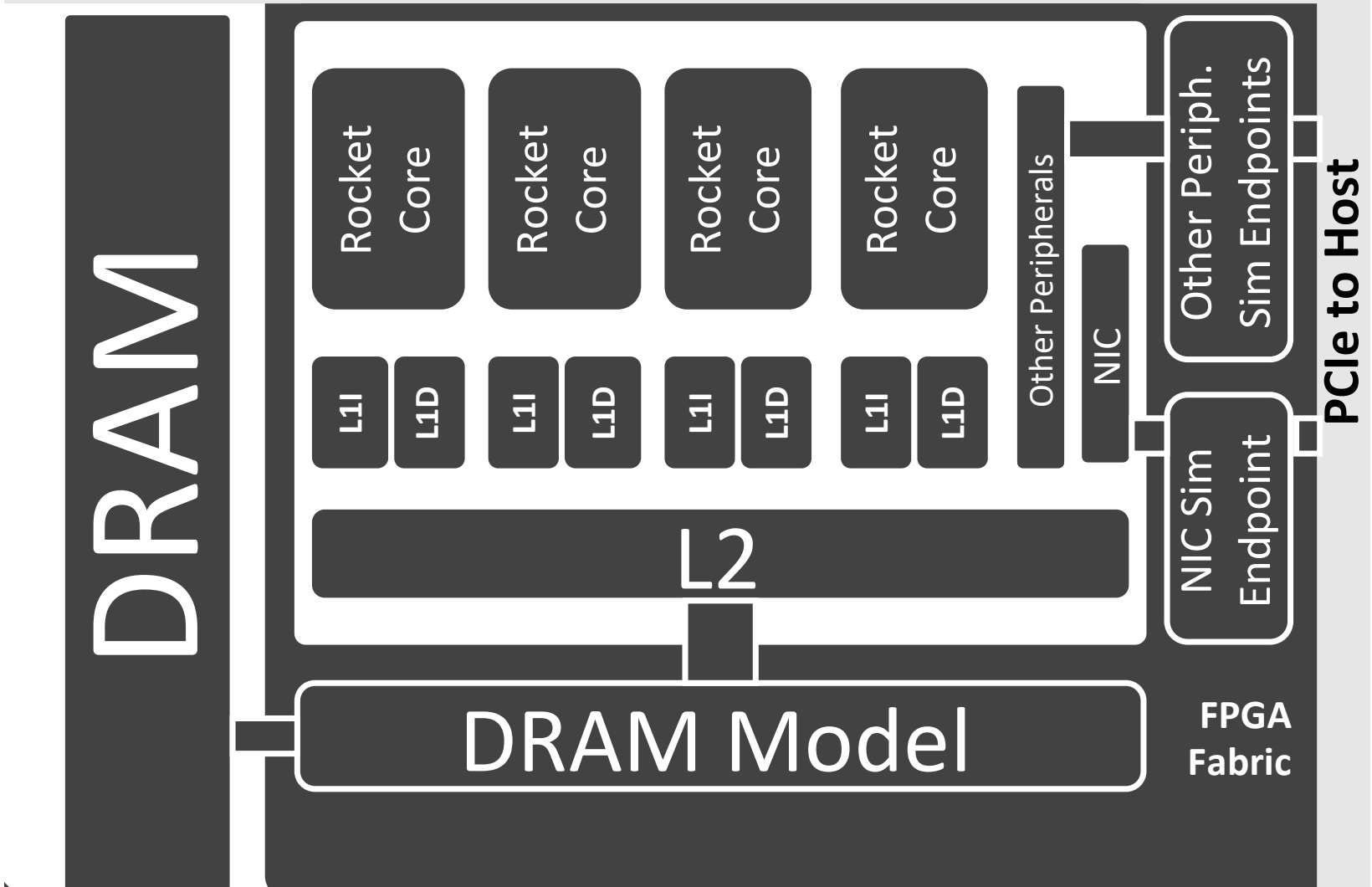
Resource Util.

- < ¼ of an FPGA

Sim Rate

- N/A

Step 2: FPGA Simulation of one server blade



Modeled System

- 4x RISC-V Rocket Cores @ 3.2 GHz
- 16K I/D L1$
- 256K Shared L2$
- 200 Gb/s Eth. NIC
- 16 GB DDR3

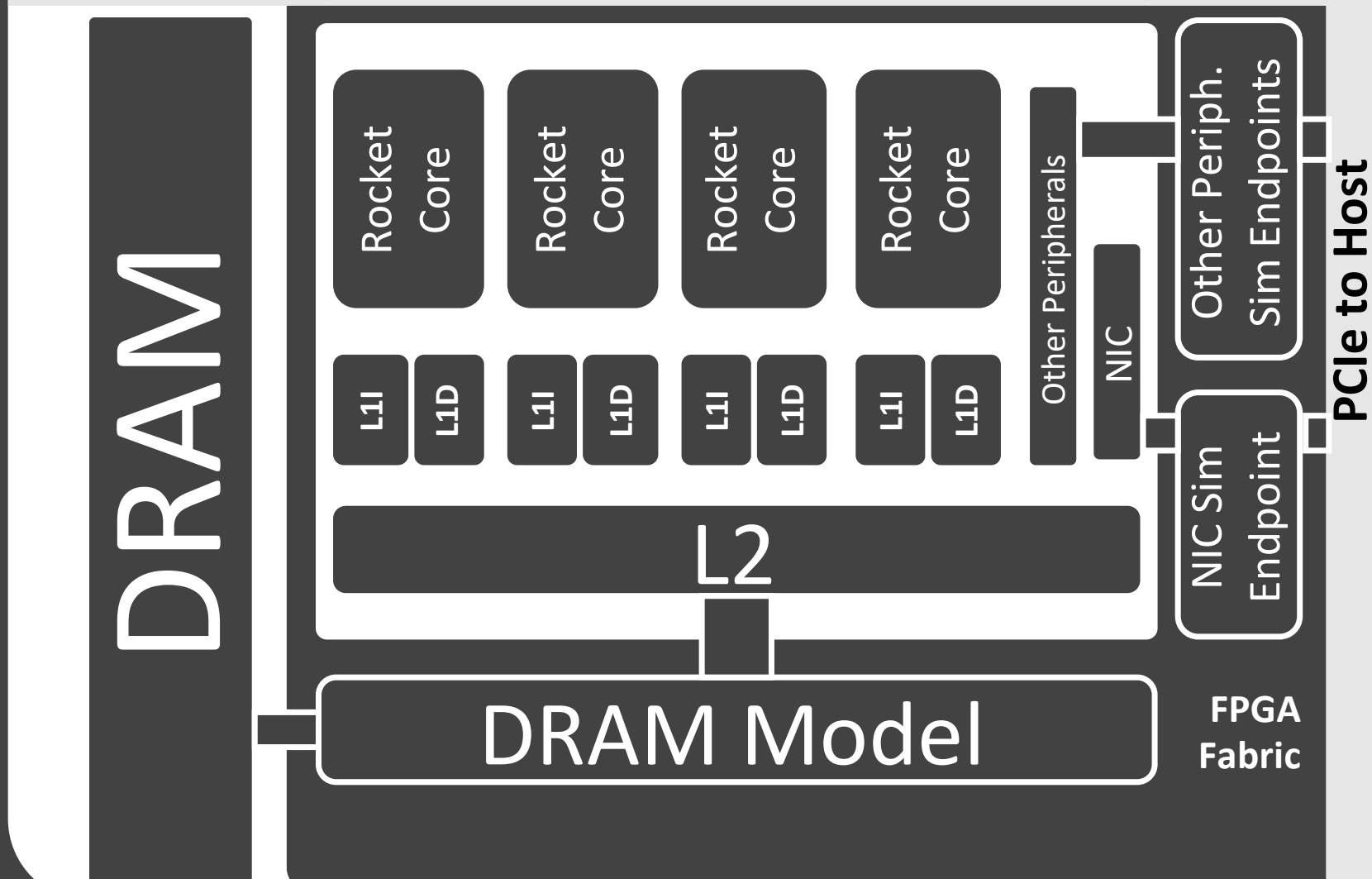
Resource Util.

- < ¼ of an FPGA
- ¼ Mem Chans

Sim Rate

- ~150 MHz
- ~40 MHz (netw)

Step 2: FPGA Simulation of one server blade



Modeled System

- 4x RISC-V Rocket Cores @ 3.2 GHz
- 16K I/D L1$
- 256K Shared L2$
- 200 Gb/s Eth. NIC
- 16 GB DDR3

Resource Util.

- < ¼ of an FPGA
- ¼ Mem Chans

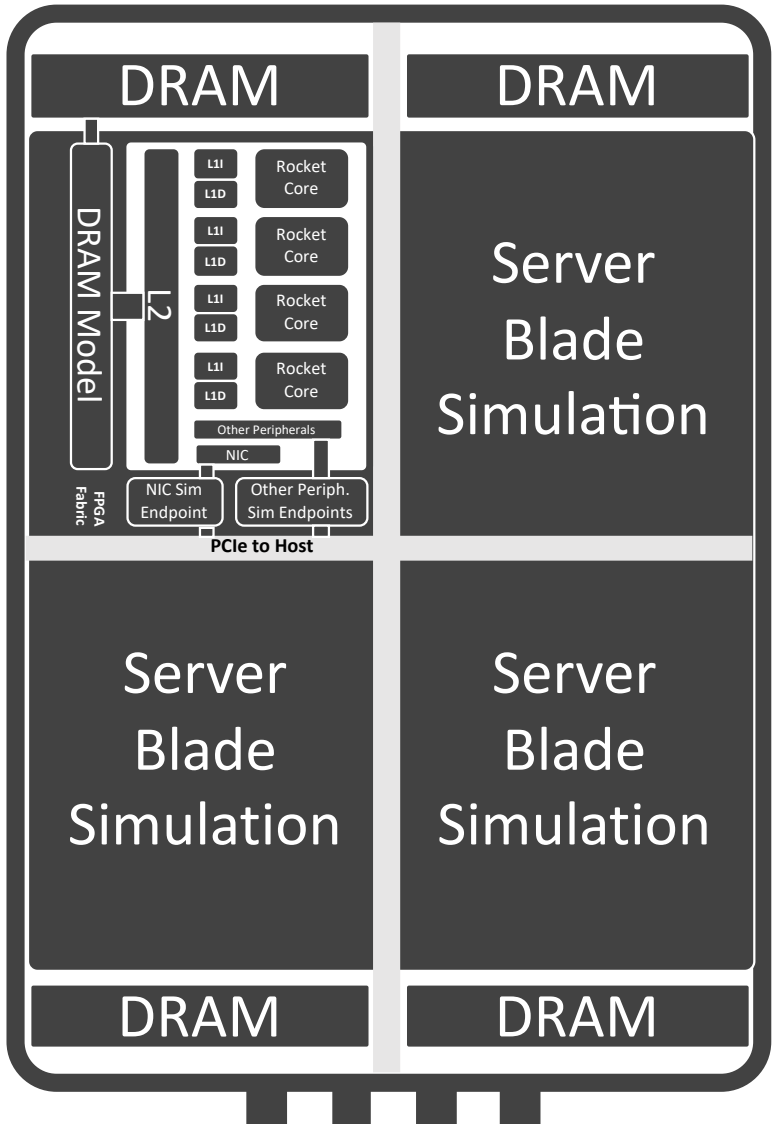
Sim Rate

- ~150 MHz
- ~40 MHz (netw)

Step 3: FPGA Simulation of 4 server blades

Cost:
$0.49 per hour (spot)

$1.65 per hour (on-demand)



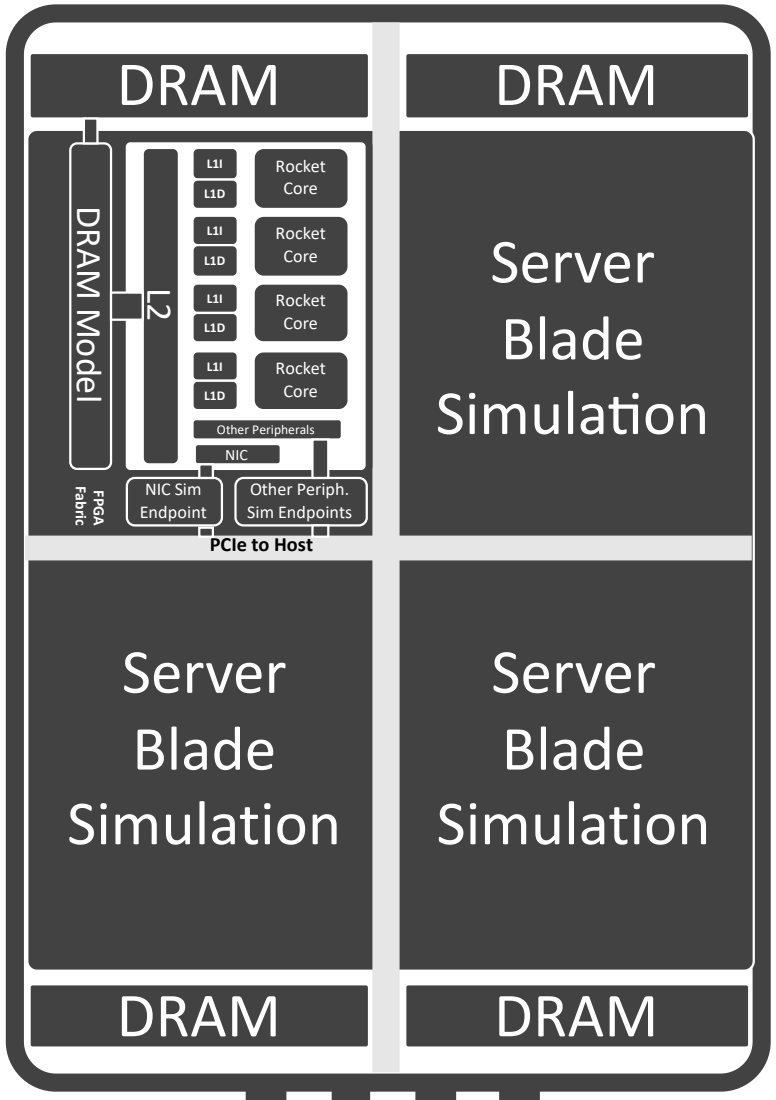
Modeled System
- 4 Server Blades
- 16 Cores
- 64 GB DDR3

Resource Util.
- < 1 FPGA
- 4/4 Mem Chans

Sim Rate
- ~14.3 MHz (netw)

Step 3: FPGA Simulation of 4 server blades

FPGA
4 Sims)



FPGA
(4 Sim

Modeled System

- 4 Server Blades
- 16 Cores
- 64 GB DDR3

Resource Util.

- < 1 FPGA
- 4/4 Mem Chans

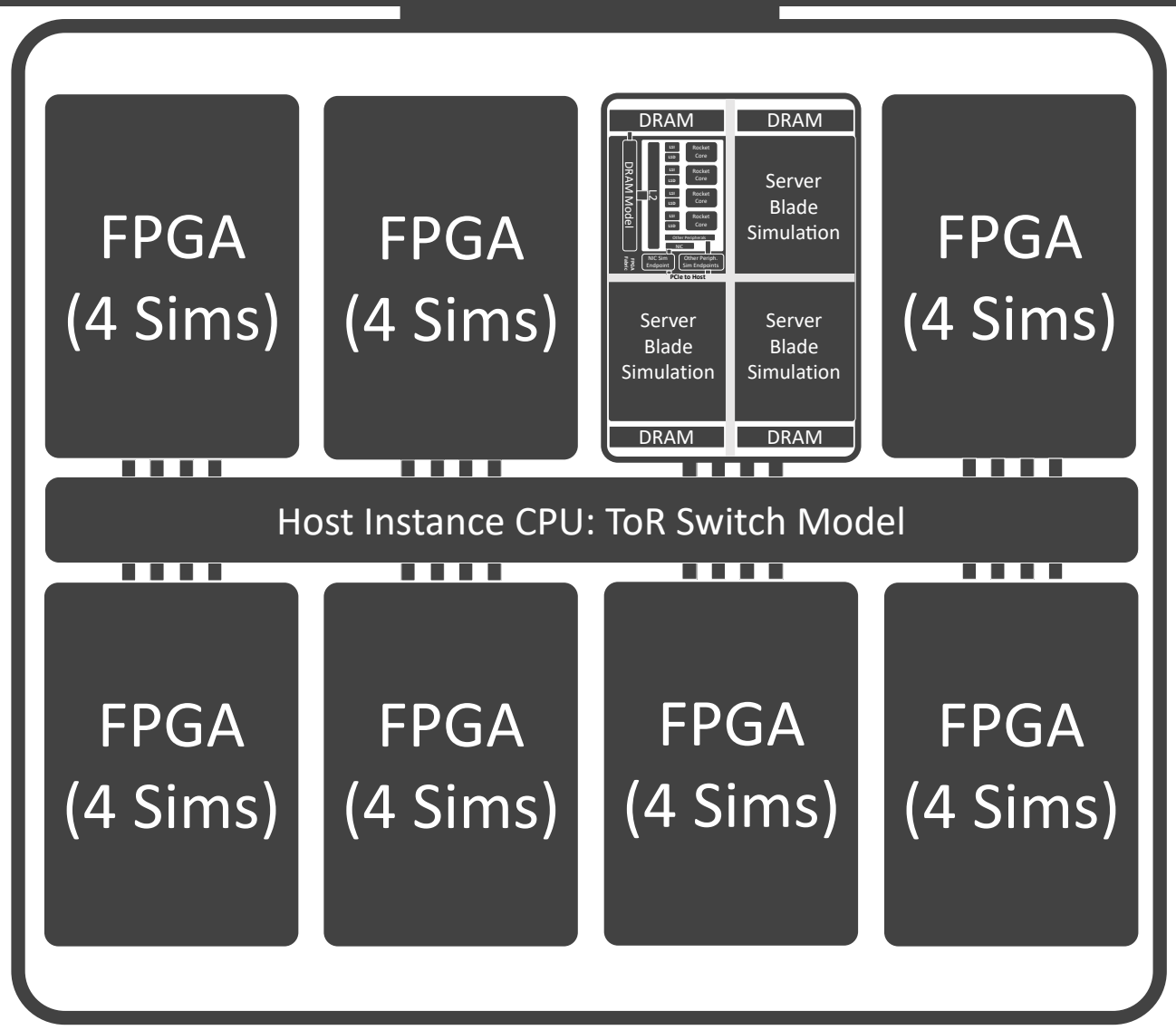
Sim Rate

- ~14.3 MHz (netw)

Step 4: Simulating a 32 node rack

Cost:
$2.60 per hour (spot)

$13.20 per hour (on-demand)



Modeled System

- 32 Server Blades
- 128 Cores
- 512 GB DDR3
- 32 Port ToR Switch
- 200 Gb/s, 2us links

Resource Util.

- 8 FPGAs =
- 1x f1.16xlarge

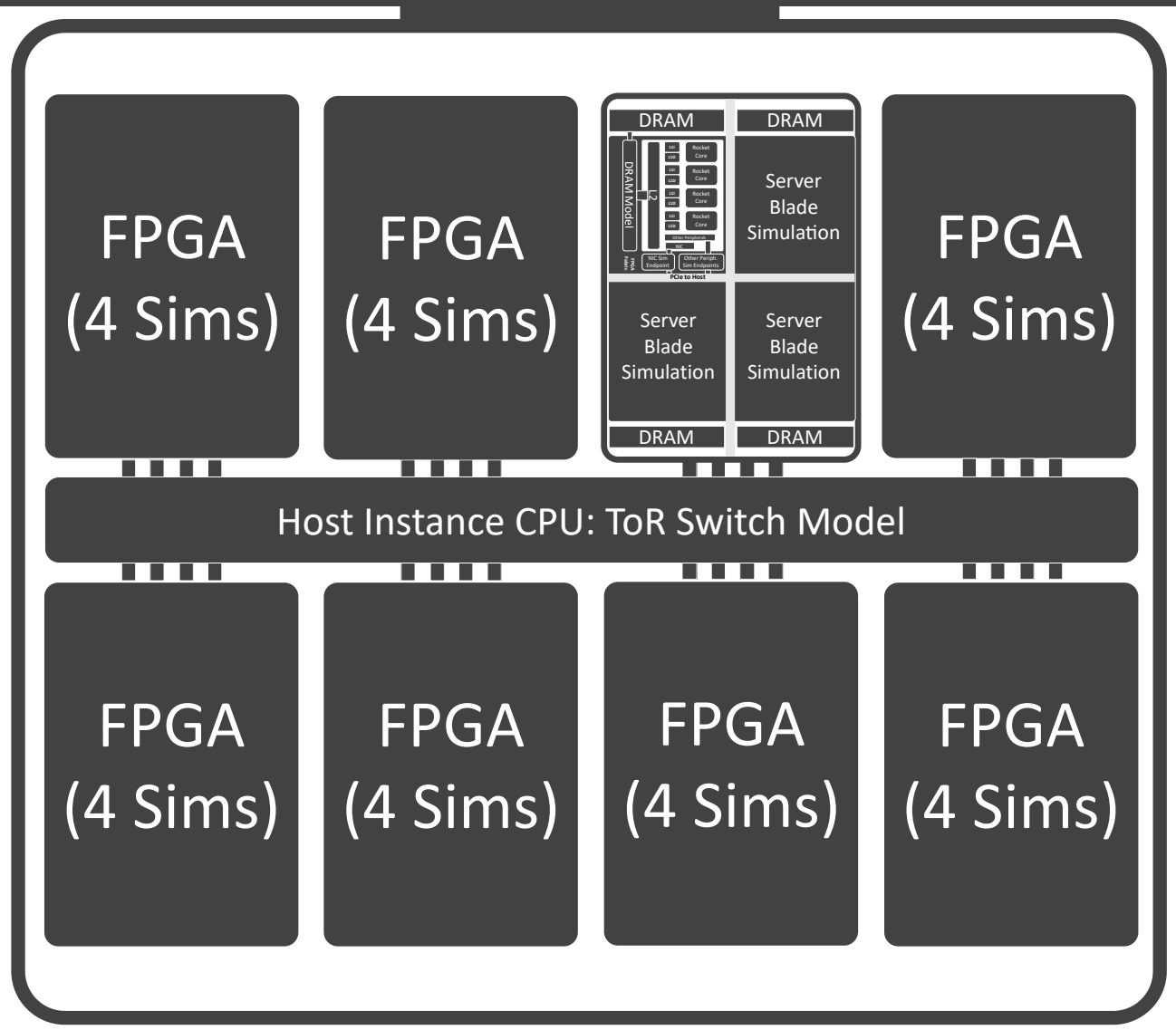
Sim Rate

- ~10.7 MHz (netw)

Step 4: Simulating a 32 node rack

Cost:
$2.60 per hour (spot)

$13.20 per hour (on-demand)



Modeled System

- 32 Server Blades
- 128 Cores
- 512 GB DDR3
- 32 Port ToR Switch
- 200 Gb/s, 2us links

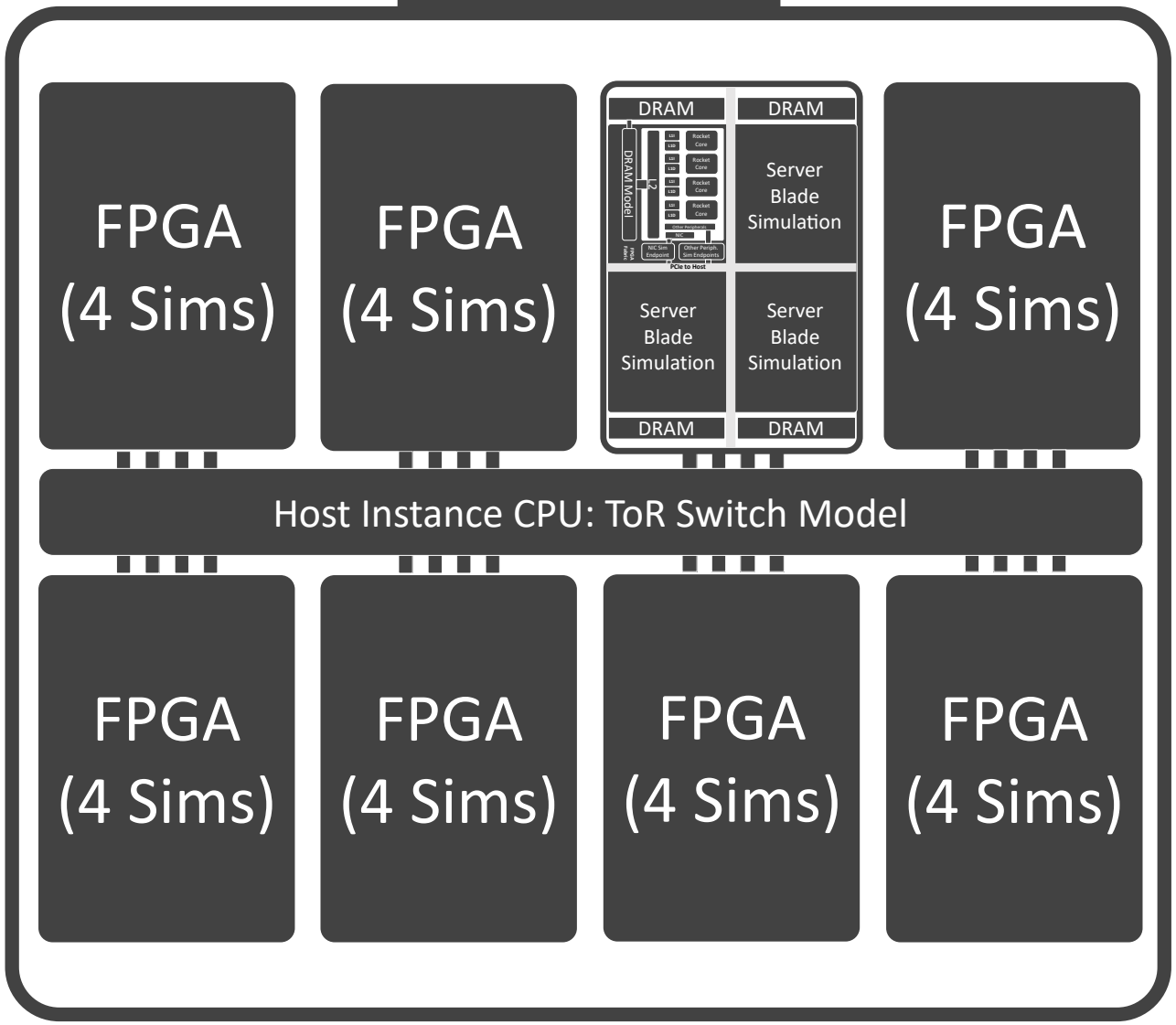
Resource Util.

- 8 FPGAs =
- 1x f1.16xlarge

Sim Rate

- ~10.7 MHz (netw)

Step 4: Simulating a 32 node rack



Modeled System

- 32 Server Blades
- 128 Cores
- 512 GB DDR3
- 32 Port ToR Switch
- 200 Gb/s, 2us links

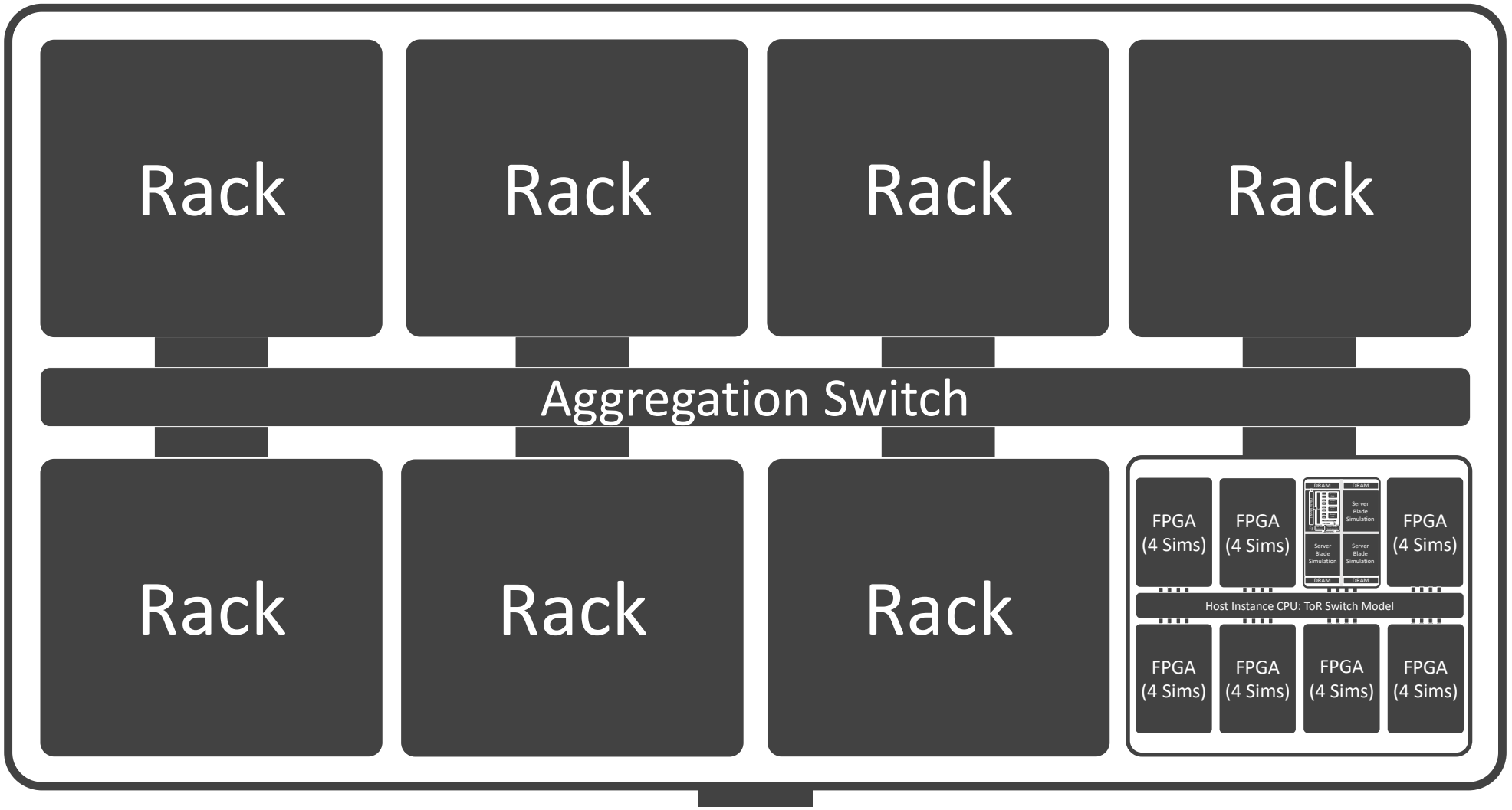
Resource Util.

- 8 FPGAs =
- 1x f1.16xlarge

Sim Rate

- ~10.7 MHz (netw)

Step 5: Simulating a 256 node “aggregation pod”



Modeled System

- 256 Server Blades
- 1024 Cores
- 4 TB DDR3
- 8 ToRs, 1 Aggr
- 200 Gb/s, 2us links

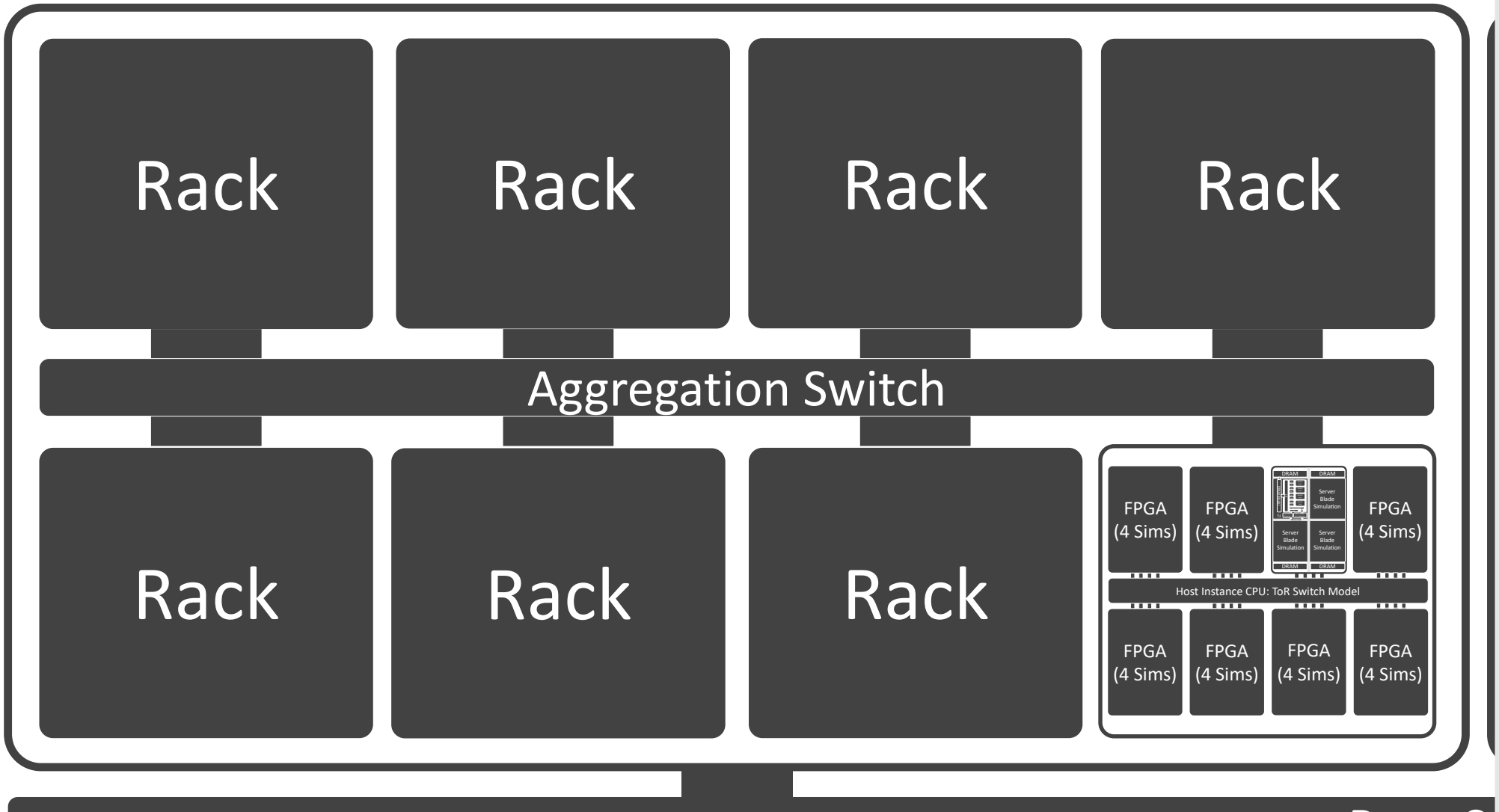
Resource Util.

- 64 FPGAs =
- 8x f1.16xlarge
- 1x m4.16xlarge

Sim Rate

- ~9 MHz (netw)

Step 5: Simulating a 256 node “aggregation pod”



Modeled System

- 256 Server Blades
- 1024 Cores
- 4 TB DDR3
- 8 ToRs, 1 Aggr
- 200 Gb/s, 2us links

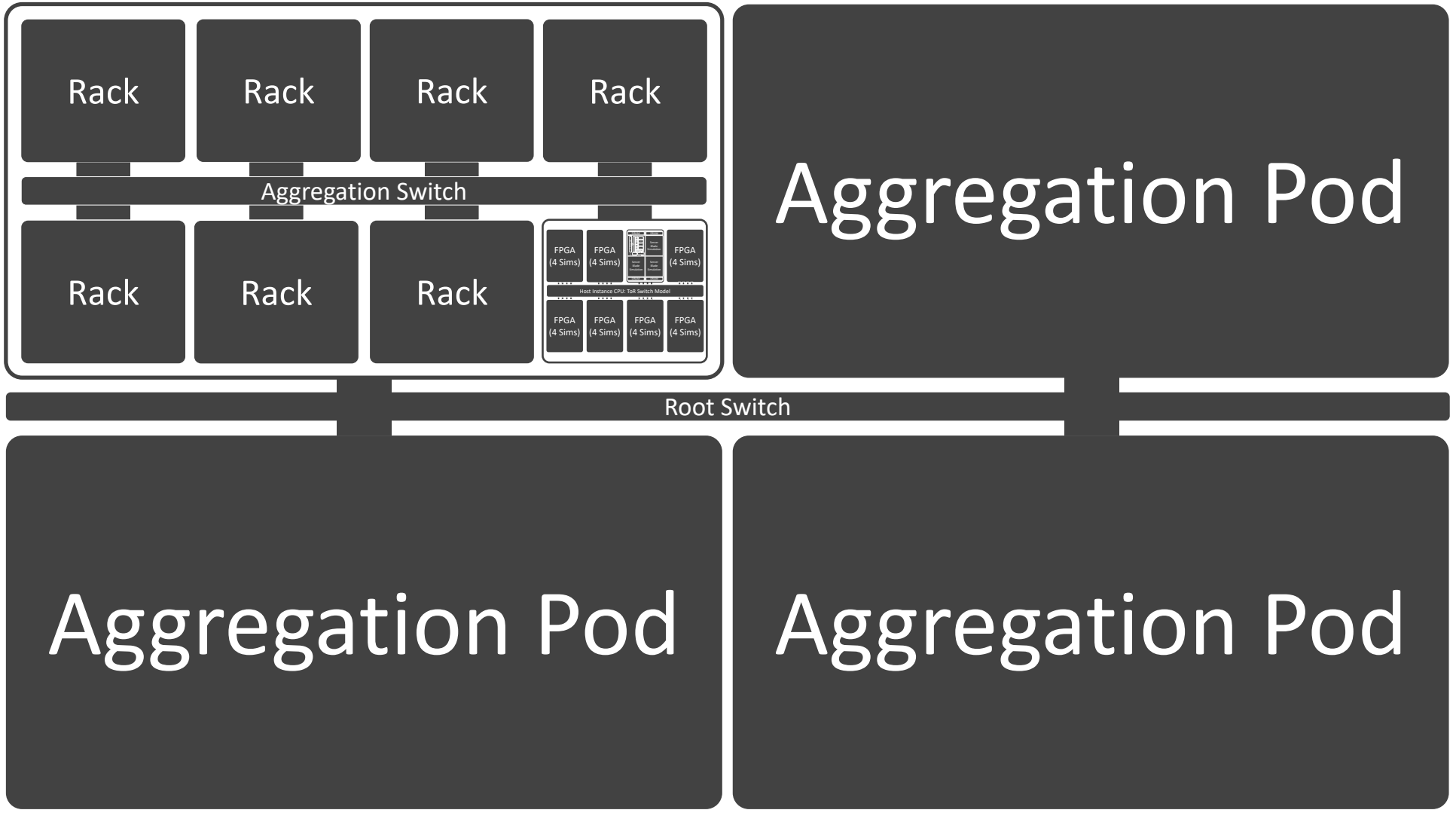
Resource Util.

- 64 FPGAs =
- 8x f1.16xlarge
- 1x m4.16xlarge

Sim Rate

- ~9 MHz (netw)

Step 6: Simulating a 1024 node datacenter



Modeled System

- 1024 Servers
- 4096 Cores
- 16 TB DDR3
- 32 ToRs, 4 Aggr, 1 Root
- 200 Gb/s, 2us links

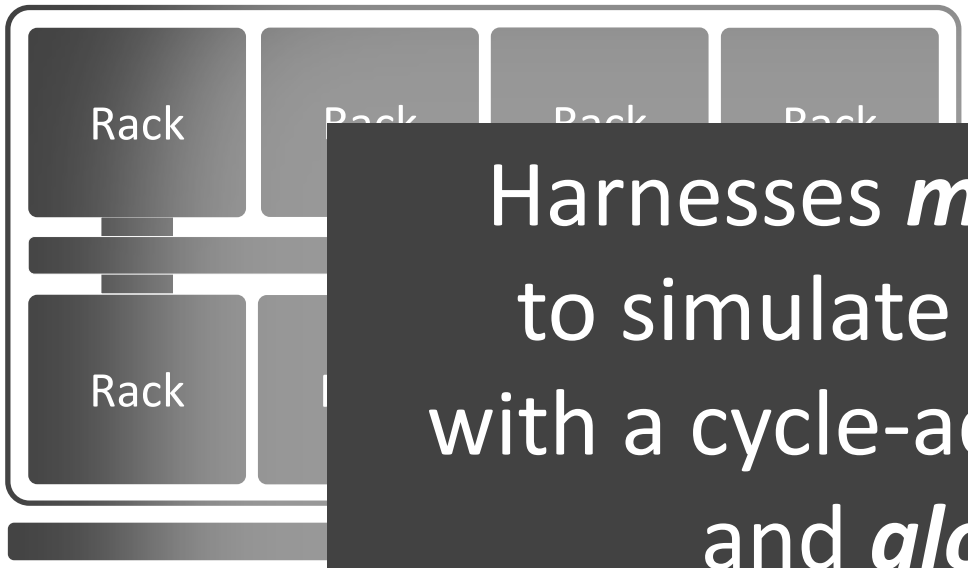
Resource Util.

- 256 FPGAs =
- 32x f1.16xlarge
- 5x m4.16xlarge

Sim Rate

- ~6.6 MHz (netw)

Step 6: Simulating a 1024 node datacenter



Harnesses ***millions of dollars*** of FPGAs
to simulate ***1024 nodes cycle-exactly***
with a cycle-accurate ***network simulation***
and ***global synchronization***
at a cost-to-user of only ***100s of dollars/hour***

Aggregation Pod

Aggregation Pod

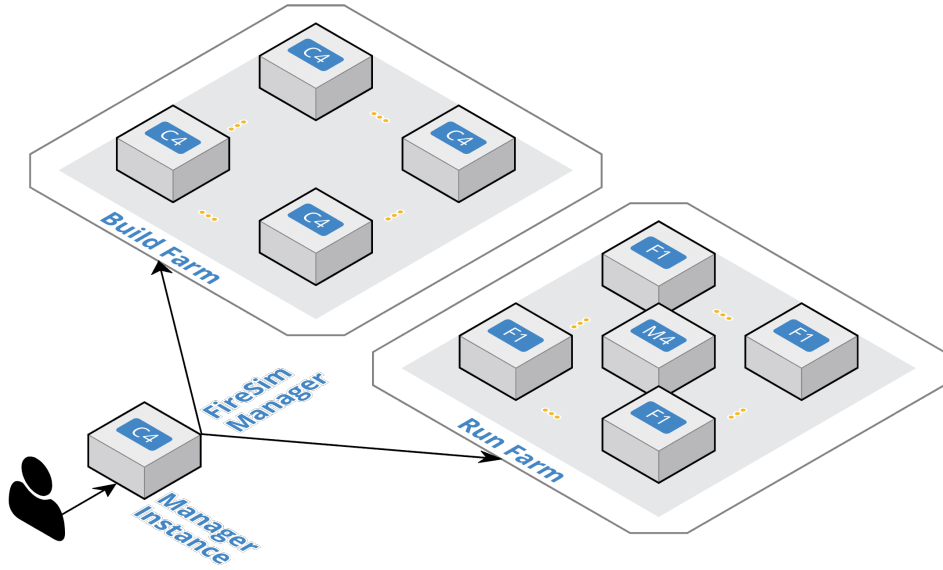
Modeled System
- 1024 Servers
- 6 Cores
- TB DDR3
- oRs, 4 Aggr, 1
- Gb/s, 2us

urce Util.
- 256 FPGAs =
- 32x f1.16xlarge
- 5x m4.16xlarge

Sim Rate
- ~6.6 MHz (netw)

Productive Open-Source FPGA Simulation

- github.com/firesim/firesim, BSD Licensed
- An “easy” button for fast, FPGA-accelerated full-system simulation
  - Plug in your own RTL designs, your own HW/SW models
  - One-click: Parallel FPGA builds, Simulation run/result collection, building target software
  - Scales to a variety of use cases:
    - Networked (performance depends on scale)
    - Non-networked (150+ MHz), limited by your budget
- `firesim` command line program
  - Like `docker` or `vagrant`, but for FPGA sims
  - User doesn’t need to care about distributed magic happening behind the scenes

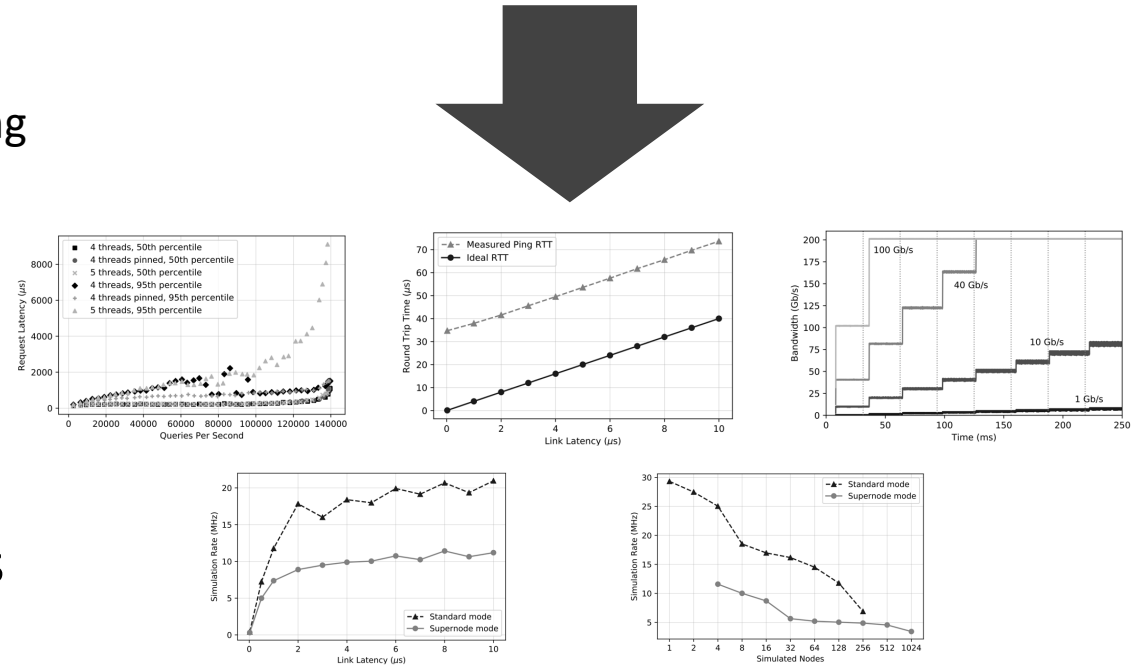


FireSim Developer Environment

Productive Open-Source FPGA Simulation

- Scripts can call `firesim` to fully automate distributed FPGA sim
  - **Reproducibility**: included scripts to reproduce ISCA 2018 results
  - e.g. scripts to automatically run SPECInt2017 with full **reference inputs** in ≈1 day
  - Many others included
  - Several user papers have gone through artifact evaluation using FireSim (*nanoPU*, *FirePerf*, *Protobuf accel.*, *MoCA*, *Simulator Independent Coverage*, etc.)
- 200+ pages of documentation: https://docs.fires.im
- AWS provides grants for researchers: https://aws.amazon.com/grants/
- Xilinx University Program provides FPGA donations for university researchers: https://www.xilinx.com/support/university.html

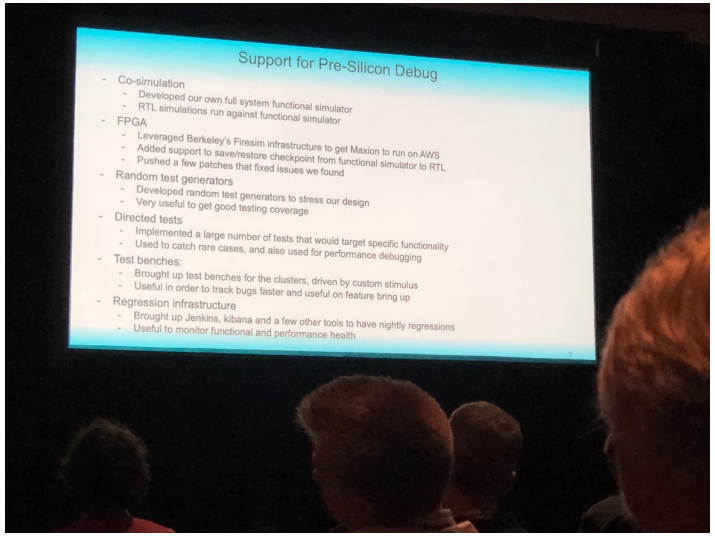
```
$ cd fsim/deploy/workloads
$ ./run-all.sh
```



Join the FireSim Community!: Open-source users and industrial users

- More than 200 mailing list members and 850 unique cloners per-week
- Projects with public FireSim support
  - Chipyard
  - Rocket Chip
  - BOOM
  - Hwacha Vector Accelerator
  - Keystone Secure Enclave
  - Gemmini
  - NVIDIA Deep Learning Accelerator (NVDLA)
    - NVIDIA Blog post: https://devblogs.nvidia.com/nvdla/
  - BOOM Spectre replication/mitigation
  - Protobuf Accelerator
  - Too many to list here!

- Companies publicly announced using FireSim
  - SiFive validation paper @ VLSI'20
  - Esperanto Maxion ET
  - Intensivate IntenCore
  - Tenstorrent
  - Galois (DARPA SSITH/FETT)



Esperanto announcement at RISC-V Summit 2018

FireSim in DARPA FETT

- DARPA SSITH: Building hardware defenses to address common software vulnerabilities
- DARPA FETT: How good are the defenses built in SSITH?
  - Multiple designs hosted for attack in FireSim [1]
- “Morpheus II: A RISC-V Security Extension for Protecting Vulnerable Software and Hardware”
  - Developed by UT Austin, U Mich., Agita Labs
  - Hosted on FireSim for FETT [2]
  - Over 500 attackers tried to break Morpheus II defenses, working for large bug bounties. None succeeded [3]

[1] K. Hopfer. Leveraging Amazon EC2 F1 Instances for Development and Red Teaming in DARPA’s First-Ever Bug Bounty Program. AWS APN Blog. May 2021.
[2] A. Harris, et. al., “Morpheus II: A RISC-V Security Extension for Protecting Vulnerable Software and Hardware”. In proceedings of the 2021 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), December 2021.
[3] T. Austin., et. al., “Morpheus II: A RISC-V Security Extension for Protecting Vulnerable Software and Hardware”. In HotChips 33, August 2021.

Join the FireSim Community!: Academic Users and Awards

- **ISCA ‘18**: Maas et. al. HW-GC Accelerator (**Berkeley**)
- **MICRO ‘18**: Zhang et. al. “Composable Building Blocks to Open up Processor Design” (**MIT**)
- **RTAS ‘20**: Farshchi et. al. BRU (**Kansas**)
- **EuroSys ‘20**: Lee et. al. Keystone (**Berkeley**)
- **OSDI ‘21**: Ibanez et. al. nanoPU (**Stanford**)
- **USENIX Security ‘21**: Saileshwar et. al. MIRAGE (**Georgia Tech**)
- **CCS ‘21**: Ding et. al. “Hardware Support to Improve Fuzzing Performance and Precision” (**Georgia Tech**)
- **MICRO ’21:** Karandikar et. al. “A Hardware Accelerator for Protocol Buffers” (**Berkeley/Google**)
- **MICRO ‘21**: Gottschall et. al. TIP (**NTNU**)
- **Over 20 *additional* user papers on the FireSim website:**
  - https://fires.im/publications/#userpapers

FireSim and User Awards:

- ISCA ‘18 FireSim Paper:
  - IEEE Micro Top Pick
  - CACM Research Highlights Nominee from ISCA ’18
- ISCA ‘18 Maas et. al.:
  - IEEE Micro Top Pick
- MICRO ‘18 Zhang et. al.:
  - IEEE Micro Top Pick
- MICRO ‘21 Gottschall et. al.:
  - MICRO-54 Best paper runner-up
- MICRO ‘21 Karandikar et. al.:
  - MICRO-54 Distinguished Artifact winner
  - IEEE Micro Top Pick Honorable Mention
- DAC ‘21 Genc et. al.:
  - DAC 2021 Best Paper winner

Join the FireSim Community!: Academic Users and Awards

- **ISCA '18**: Maas et. al. HW-GC Accelerator (**Berkeley**)
- **MICRO '18**: Zhang et. al. "Composable Building Blocks to Open up Processor De
- **RTAS '20**: Fa
- **EuroSys '20**:
- **OSDI '21**: Iban
- **USENIX Securi**
- **CCS '21**: Din Performance
- **MICRO '21:** Karandikar et. al. "A Hardware Accelerator for Protocol Buffers" (**Berkeley/Google**)
- **MICRO '21**: Gottschall et. al. TIP (**NTNU**)
- **Over 20 *additional* user papers on the FireSim website:**
  - https://fires.im/publications/#userpapers

FireSim has been used in published work from authors at over 20 academic and industrial institutions*

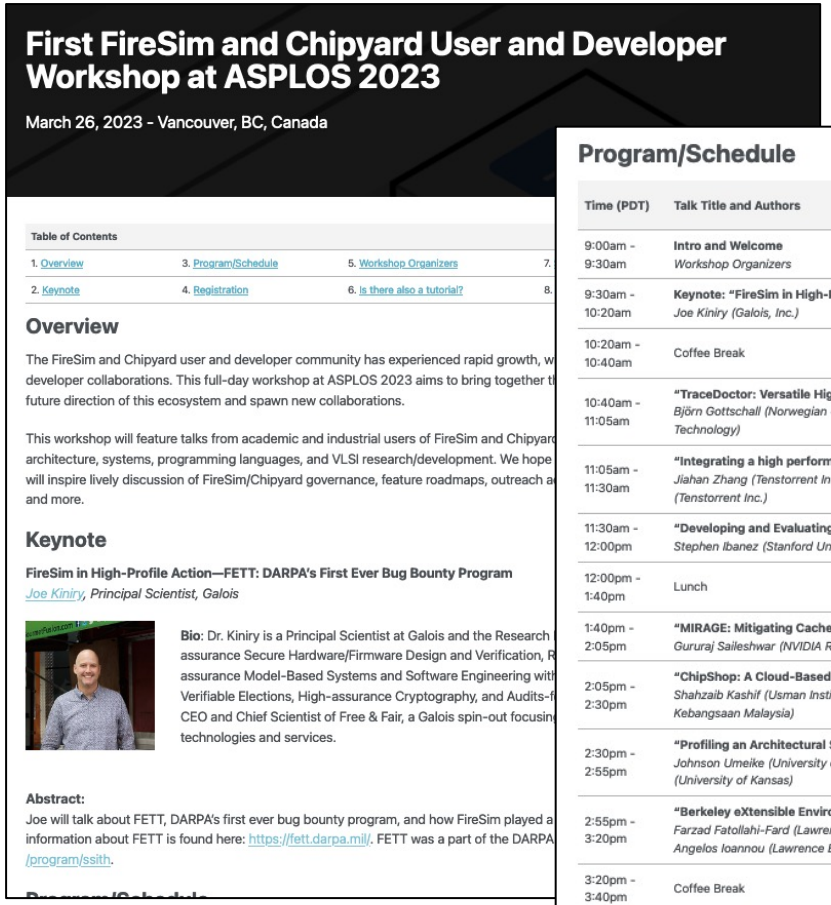
**actually used, not only cited*

FireSim and User Awards:

- ISCA '18 FireSim Paper:
- ghts Nominee
- :
- . al.:
- unner-up
- MICRO '21 Karandikar et. al.:
  - MICRO-54 Distinguished Artifact winner
  - IEEE Micro Top Pick Honorable Mention
- DAC '21 Genc et. al.:
  - DAC 2021 Best Paper winner

Learn more about FireSim use-cases, directly from the users!

- Videos from the First FireSim/Chipyard Workshop, co-located with ASPLOS 2023, are now available on YouTube!
  - 10 great talks from users across academia and industry
- Links available at: https://fires.im/workshop-2023/



First FireSim and Chipyard User and Developer Workshop at ASPLOS 2023

March 26, 2023 - Vancouver, BC, Canada

Table of Contents

1. Overview
2. Keynote
3. Program/Schedule
4. Registration
5. Workshop Organizers
6. Is there also a tutorial?
7.
8.

Overview

The FireSim and Chipyard user and developer community has experienced rapid growth, w
developer collaborations. This full-day workshop at ASPLOS 2023 aims to bring together t
future direction of this ecosystem and spawn new collaborations.

This workshop will feature talks from academic and industrial users of FireSim and Chipyar
architecture, systems, programming languages, and VLSI research/development. We hope
will inspire lively discussion of FireSim/Chipyard governance, feature roadmaps, outreach a
and more.

Keynote

FireSim in High-Profile Action—FETT: DARPA's First Ever Bug Bounty Program

Joe Kiniry, Principal Scientist, Galois

Bio: Dr. Kiniry is a Principal Scientist at Galois and the Research
assurance Secure Hardware/Firmware Design and Verification, R
assurance Model-Based Systems and Software Engineering with
Verifiable Elections, High-assurance Cryptography, and Audits-f
CEO and Chief Scientist of Free & Fair, a Galois spin-out focusin
technologies and services.

Abstract:
Joe will talk about FETT, DARPA's first ever bug bounty program, and how FireSim played a
information about FETT is found here: https://fett.darpa.mil/. FETT was a part of the DARPA
/program/ssith.

Program/Schedule

Time (PDT)	Talk Title and Authors	Slides (coming soon)
9:00am - 9:30am	**Intro and Welcome** *Workshop Organizers*	PDF
9:30am - 10:20am	**Keynote: "FireSim in High-Profile Action—FETT: DARPA's First Ever Bug Bounty Program"** *Joe Kiniry (Galois, Inc.)*	PDF
10:20am - 10:40am	Coffee Break	
10:40am - 11:05am	**"TraceDoctor: Versatile High-Performance Tracing for FireSim"** *Björn Gottschall (Norwegian Univeristy of Science and Technology), Magnus Jahre (Norwegian Univeristy of Science and Technology)*	PDF
11:05am - 11:30am	**"Integrating a high performance instruction set simulator with FireSim to cosimulate operating system boots"** *Jiahan Zhang (Tenstorrent Inc.), Varun Koyyalagunta (Tenstorrent Inc.), Joe Rahmeh (Tenstorrent Inc.), Divyang Agrawal (Tenstorrent Inc.)*	PDF
11:30am - 12:00pm	**"Developing and Evaluating the nanoPU and nanoSort using Chipyard and Firesim"** *Stephen Ibanez (Stanford University & Intel), Theo Jepsen (Stanford University & Intel)*	PDF
12:00pm - 1:40pm	Lunch	
1:40pm - 2:05pm	**"MIRAGE: Mitigating Cache Attacks with a Randomized Fully-Associative Cache"** *Gururaj Saileshwar (NVIDIA Research & University of Toronto), Moinuddin Qureshi (Georgia Tech)*	PDF
2:05pm - 2:30pm	**"ChipShop: A Cloud-Based GUI for Accelerating SoC Design"** *Shahzaib Kashif (Usman Institute of Technology), Talha Ahmed (Usman Institute of Technology), Farhan Ahmed Karim (Universiti Kebangsaan Malaysia)*	PDF
2:30pm - 2:55pm	**"Profiling an Architectural Simulator"** *Johnson Umeike (University of Kansas), Alex Manley (University of Kansas), Neel Patel (University of Kansas), Mohammad Alian (University of Kansas)*	PDF
2:55pm - 3:20pm	**"Berkeley eXtensible Environment: A Cloud-Based Open-Source Computer Architecture Simulation Environment"** *Farzad Fatollahi-Fard (Lawrence Berkeley National Laboratory), Nirmalendu Patra (Lawrence Berkeley National Laboratory), Angelos Ioannou (Lawrence Berkeley National Laboratory), John Shalf (Lawrence Berkeley National Laboratory)*	PDF
3:20pm - 3:40pm	Coffee Break	
3:40pm - 4:05pm	**"FireSim on Xilinx U250 and Other Custom Host Platforms"** *David Christoph Metz (Norwegian University of Science and Technology), Magnus Själander (Norwegian University of Science and Technology)*	PDF
4:05pm - 4:30pm	**"Ocelot Vector Unit and Integrating SV-based Modules in BOOM"** *Dongjie (DJ) Xie (Tenstorrent Inc.), Srikanth Arekapudi (Tenstorrent Inc.)*	PDF
4:30pm - 5:00pm	**Wrap-up and Discussion** *Workshop Organizers/Attendees*	PDF

On-premises FPGA support now available!

- High-level of automation/reproducibility enabled by FireSim on AWS F1 cloud now extended to local/on-prem FPGAs:
  - Went from new machine with no FPGA attached to working FPGA-accelerated simulation in 1 hour and 40 mins
- New in latest release: Xilinx VCU118, Xilinx Alveo U250 / U280 XDMA-shell, RHS Research Nitefury II
- Use existing FireSim features at-scale and locally!
  - Cycle-accurate simulation
  - Debugging
    - Integrated logic analyzers, trace dumps, synth. assert/prints, co-simulation
  - Software support
    - FireMarshal workload management
  - … and more!

ISCA-50 FireSim Retrospective

- Stay tuned for the release of the ISCA-50 retrospectives, including FireSim!

RETROSPECTIVE: FireSim: FPGA-Accelerated Cycle-Exact Scale-Out System Simulation in the Public Cloud

Sagar Karandikar[1], Howard Mao[2,*], Donggyu Kim[3,*], David Biancolin[4,*], Alon Amid[5,*], Dayeol Lee[6,*], Nathan Pemberton[7,*], Emmanuel Amaro[8,*], Colin Schmidt[4,*], Aditya Chopra[2,*], Qijing Huang[5,*], Kyle Kovacs[9,*], Borivoje Nikolić[1], Randy Katz[1], Jonathan Bachrach[10,*], Krste Asanović[1]

[1]UC Berkeley, [2]Google, [3]Apple, [4]SiFive, [5]NVIDIA, [6]Anyscale, [7]Amazon Web Services, [8]VMWare Research, [9]DiCon Fiberoptics, [10]JITX

I. Introduction

"Why is it called *FireSim*?" is a question we receive often, posed by users who today employ FireSim to simulate a variety of systems beyond its initial goal: as a "from-the-RTL-up" simulator for a specialized Warehouse-Scale Computer (WSC) architecture called *FireBox* [O8][1]. When we set out to build FireSim, the published mandate [O8] was to:

> *"...simulate an entire FireBox, including the fiber-optic network, the switch, the NIC, and 1000 SoCs, with every core running the full BDAS stack (from the AMP Lab) and the Linux OS, as well as interactive services and batch applications, with only a factor of 1000x slowdown from realtime."*

The FireSim ISCA'18 paper [13] exceeded these objectives,

FPGAs in the public cloud became broadly available [5]. Cloud computing, established for years in systems research [O23], could now benefit architects. Academics/startups could elastically scale high-fidelity simulations to 1000s of nodes without buying millions of dollars of FPGAs. In large organizations, architects/systems SW developers, who rarely get access to costly "big metal" HW-accelerated simulators, could now co-design HW/SW directly using real RTL.

FPGA capacity grew sufficiently to host interesting research targets without *immediately* requiring "tricks" (multi-threading, abstract modeling, partitioning, etc.) from the FPGA simulator literature. Many were later added to FireSim, but critically, were not initially *required* to ship a useful simulator.

Open-source, industry-verified hardware implementa-

Questions?

Learn More:

Web: https://fires.im

Docs: https://docs.fires.im

GitHub: https://github.com/firesim/firesim

Mailing List: https://groups.google.com/forum/#!forum/firesim

 **@firesimproject**

Email: sagark@eecs.berkeley.edu

Berkeley Architecture Research

The information, data, or work presented herein was funded in part by the Advanced Research Projects Agency-Energy (ARPA-E), U.S. Department of Energy, under Award Number DE-AR0000849, by DARPA, Award Number HR0011-12-2-0016, and by NSF CCRI ENS Chipyard Award #2016662. Research was also partially funded by SLICE/ADEPT Lab industrial sponsors and affiliates Amazon, Apple, Google, Intel, Qualcomm, and Western Digital, and RISE Lab sponsor Amazon Web Services. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.