FireSim: Fast, Cycle-Accurate Datacenter Simulation in the Public Cloud

https://fires.im
@firesimproject

Sagar Karandikar, Howard Mao, Donggyu Kim, David Biancolin, Alon Amid, Dayeol Lee, Kyle Kovacs, Borivoje Nikolic, Randy Katz, Jonathan Bachrach, Krste Asanović
Why simulate datacenters?

Next-gen datacenters won’t be built only from commodity components:

- **The end of Moore’s Law**
- **Fast networks**
  - e.g. Silicon Photonics
- **Deeper memory/storage hierarchies**
  - e.g. 3DXPoint
- **New DC organizations**
  - e.g. disaggregation

[1]
...and custom HW is changing faster than ever

FPGAs:

Agile HW Design for ASICs:

[2]
So, what does our simulator need to do?

• Model HW at scale:
  • CPUs down to microarchitecture
  • Fast networks, switches
  • Novel accelerators

• Run real SW:
  • Real OS, networking stack (Linux)
  • Real frameworks/applications (not microbenchmarks)

• Make it usable:
  • Run on a commodity platform
  • Want to encourage collaboration between systems, architecture
Evaluating Computer Systems

1. Build the hardware
2. Build a software simulator
3. Build a hardware-accelerated simulator
(1) Build the hardware: “Tapeout”

- Architects build hardware by writing RTL (Verilog, VHDL, etc.)
  - Considered “hard”
- Validate/unit test in software RTL simulation
- Push through ASIC tools (licenses very expensive + sign NDAs for tools/process)
- Iterate for “Quality of Results” (QoR) aka hitting your target frequency, area, passing design rules of the process
- Send a final design to the fab, give them millions of $, wait weeks/months for the result
- Get chips back, network together into a DC
(2) Write/Use a software simulator

- Lots of options out there
- Easy to prototype new HW: write C++ code
- Also easy to model something that you can’t really build
- Very slow to run (at best 100s of KIPS for a single-node)
  - Either run microbenchmarks or use sampling/skip-forward
- Once you come up with a good design, then write RTL
- Can network together many instantiations of a SW simulator to simulate a datacenter
(3) Build a HW-accelerated simulator: DIABLO

• Need to hand-write RTL models (even harder than “tapeout-ready” RTL)
• Tied to a custom host-platform
• DIABLO [4]:
  • Simulated 3072 servers, 96 ToRs at ~2.7 MHz
  • Booted Linux, ran apps like memcached
  • $100k+ host platform, custom built
  • Abstract processor, switch models
    • Can’t take this RTL and tape it out
## Evaluating Systems for Evaluating Computer Systems

<table>
<thead>
<tr>
<th>Metric</th>
<th>SW Simulator</th>
<th>HW-accel Simulator</th>
<th>Build the real HW</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>$</td>
<td>$$$</td>
<td>$$$$$</td>
</tr>
<tr>
<td>Time-to-first-cycle</td>
<td>Fastest (recompile C++)</td>
<td>Medium (FPGA tools)</td>
<td>Slowest (CAD tools + fab)</td>
</tr>
<tr>
<td>Runtime Slowdown</td>
<td>100,000x</td>
<td>1,000x</td>
<td>1x</td>
</tr>
<tr>
<td>Can tapeout?</td>
<td>X</td>
<td>X / ✔</td>
<td>✔</td>
</tr>
<tr>
<td>Run real SW</td>
<td>X (too slow)</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Commodity Platform</td>
<td>✔</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
How do we improve?

Useful hardware trends:

Open ISA

Open, Silicon-Proven SoC Implementations

High-Productivity Hardware Design Language

FPGAs in the Cloud
So, what are we building?

- A datacenter simulator that...
  - Models servers, accelerators, switches, and datacenter interconnect
  - Uses a commodity host platform with FPGAs (EC2 F1)
  - Lets users work with:
    - RTL (Chisel/Verilog) for customizing server blades, building accelerators, etc.
    - Software models (C++) for switches
  - Automates high-performance simulated network transport + the process of using FPGAs, mapping simulation to the host, and building a simulator out of RTL and software models
  - Runs real software stacks at reasonable speed (Linux + apps)
FireSim Target Design (Servers + Network)

- Server blades, each with:
  - Quad-core RISC-V Rocket @ 3.2 GHz
  - 16 KiB I$, 16 KiB D$, 256 KiB L2
  - 16 GB DRAM
  - 200 Gbps Ethernet NIC
  - Optional Accelerators
- High-performance network:
  - Parameterizable BW/link latency
    - e.g. 200 Gbps, 2μs
  - Easy to add your own link-layer
    - We provide Ethernet
  - Switch models with configurable # of ports
  - Configurable topology
Mapping a simulation to EC2 F1

• Server Simulation
  • Lots of inherent parallelism
  • We have the RTL: Xform it (FAME-1)
  • Put it on the FPGAs

• Network simulation
  • Little parallelism in switch models (e.g. a thread per port)
  • Need to coordinate all of our distributed server simulations
  • So use CPUs + host network
FAME-1 Transforming RTL

• Given RTL, we want to **automatically transform** a design into decoupled cycle-accurate simulator RTL that we can run on the FPGA

• See MIDAS/Strober [5,6] from CARRV/ISCA’16
Single-host-node simulation metrics

- We pack four quad-core server simulations per FPGA
  - = 32 server simulations per f1.16xlarge
  - = 128 simulated cores per f1.16xlarge
  - One simulation management thread per FPGA
- 32-port, 200 Gbps per-port ToR switch model
  - One thread-per-port (16xlarge has 64 vCPUs)
- Runs at ~5 MHz -> ~400 million insts/sec
- $13.20/hr on-demand, ~$2.60/hr spot
Reproducing tail latency effects from real systems

- Leverich and Kozyrakis show effects of thread-imbalance in *memcached* in [3]
- We can observe this effect in simulation
Scaling to a 1024 Node RISC-V Datacenter

- 1024 server blades (4096 cores), 32 ToR switches
  - 32 f1.16xlarges
- 1 Root + 4 Aggregation switches
  - 5 m4.16xlarges
- Runs at 3.4 MHz (13 billion insts/s)
- Sample `memcached` run:

<table>
<thead>
<tr>
<th></th>
<th>50th Percentile (µs)</th>
<th>95th Percentile (µs)</th>
<th>Aggregate Queries-Per-Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross-ToR</td>
<td>79.3</td>
<td>128.2</td>
<td>4.7 million</td>
</tr>
<tr>
<td>Cross-aggregation</td>
<td>87.1</td>
<td>111.3</td>
<td>4.5 million</td>
</tr>
<tr>
<td>Cross-datacenter</td>
<td>93.8</td>
<td>119.5</td>
<td>4.1 million</td>
</tr>
</tbody>
</table>
Summing Up

• We can prototype a datacenter built on any RISC-V core
  • You bring the SoCs, accelerators

• Simulation is automatically built and deployed

• ssh into the simulated system, just like a real cluster

Automatically deployed, high-performance, distributed simulation
Thanks!

Talk to us about:
• Automating building/distributing simulations across EC2 instances
• OoO Cores (e.g. BOOM) / integrating your high-performance core
• “Functional” network simulation
  • e.g. Automatically run all of SPECInt06-ref on Rocket Chip @ 150 MHz, in the cloud, in < 1 day
  • Check out our demo/blog post on the AWS Compute Blog
• TCO of simulation in the cloud
• RISC-V in the cloud
• Scaling further
References

[4] Zhangxi Tan, Zhenghao Qian, Xi Chen, Krste Asanovic, and David Patterson. DIABLO: A Warehouse-Scale Computer Network Simulator using FPGAs. ASPLOS '15