Validating Full-System RISC-V Simulator: A Systematic Approach

Karan Pathak[1,2]*, Joshua Klein[1], Giovanni Ansaloni[1], Marina Zapater[1,3] and David Atienza[1]

[1]Embedded Systems Laboratory, École Polytechnique Fédérale de Lausanne
[2]Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology
[3]REDS. School of Engineering and Management Vaud, HES-SO University of Applied Sciences and Arts Western Switzerland

Abstract

RISC-V-based Systems-on-Chip (SoCs) are witnessing a steady rise in adoption in both industry and academia. However, the limited support for Linux-capable Full System-level simulators hampers development of the RISC-V ecosystem. We address this by validating a full system-level simulator, gXR5 (gem5-eXtensions for RISC-V), against the SiFive HiFive Unleashed SoC, to ensure performance statistics are representative of actual hardware. This work also enriches existing methodologies to validate the gXR5 simulator against hardware by proposing a systematic component-level calibration approach. The simulator error for selected SPEC CPU2017 applications reduces from 44% to 24%, just by calibrating the CPU. We show that this systematic component-level calibration approach is accurate, fast (in terms of simulation time), and generic enough to drive future validation efforts.

Introduction

A Linux-capable Full System-level simulator is one that executes benchmarks (e.g. SPEC CPU2017 suite) atop a kernel, system interface, and detailed hardware models built-in software. To date, a full system-level simulator for RISC-V has not been publicly validated against fabricated hardware to target the precise modeling of the execution of benchmark suites. gXR5 is a RISC-V-based Linux-capable, full system-level simulator built into the gem5 architectural simulator that is capable of simulating the run-time of high-level applications [1]. Since it is built on gem5 [2], an open-source, cycle-accurate, and event-driven architecture simulator, it supports simulation of SoCs by providing tunable architectural and micro-architectural models. It can simulate multiple instruction set architectures using ISA-agnostic CPU, bus and memory models.

Although gem5 and gXR5 are easily tunable, validating the simulator for the SPEC CPU2017 benchmark suite is a non-trivial task. The SPEC CPU2017 suite, when run uncalibrated (hereafter 'baseline') on gXR5, has a mean absolute percentage error (simulated vs actual hardware, hereafter, 'error') of 44.3% in execution time. The existing methodologies for validating simulator models for the SPEC suites use synthetic micro-benchmarks [3, 4, 5, 6] to reduce the error (depicted as "micro-architectural-level" calibration in Fig 1). These micro-benchmarks are not representative of time complexity of actual workloads, thereby leading to poor performance accuracy of simulators when real user applications are run. Desikan et al. [3] achieve 18% simulator error in IPC for macro-benchmarks derived from the SPEC CPU2000 benchmark suite. Attempts have been made to use the correlation between micro-architectural events and error in IPC (e.g. Pearson's correlation) [5] and Hardware Performance Counters (HPCs) for validating the simulator. Huppert et al. [4] achieve an error of $20 - 25\%$ in IPC for the SPEC CPU2017 benchmark suite.

Our proposed methodology introduces component-level calibration of gem5 models targeting the Sifive HighFive Unleashed System-on-Chip (SoC), which is faster (in terms of simulation time), accurate, and extensible to other benchmarks. A CPU model in gXR5 is calibrated using the stress-ng benchmark suite[1], reducing the error in IPC from 36% to 11.8%. Once calibrated, the execution time error reduces from 44% to 24% while running the selected SPEC CPU2017 benchmark suite. We intend to release and open source the validated gXR5 CPU model and supporting materials for adoption by the RISC-V community.

Methodology

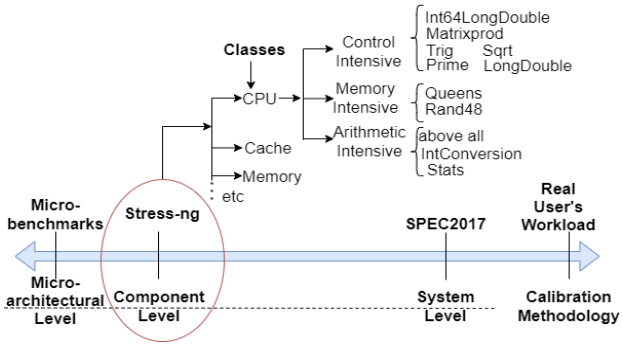

Figure 1: *Component-level calibration using the stress-ng benchmark suite.*

The target of this validation focuses on the CPU model (four-stage MinorCPU pipeline), leaving the tuning of the memory hierarchy for future work. The proposed methodology employs component-level calibration using selected "CPU class" *stressors* (collection of micro-benchmarks) of the stress-ng benchmark suite to target the FU540-C000 in-order CPU design of Linux-capable Sifive HiFive Unleashed SoC. The FU540-C000 (also referred to as U54) is a RV64GC core. The HiFive Unleashed SoC is a quin-core SoC, with one small CPU that supports real-time constraints by hosting RTOS, while the other four U54

*Corresponding Author: `karan.pathak@epfl.ch`

[1] https://wiki.ubuntu.com/Kernel/Reference/stress-ng

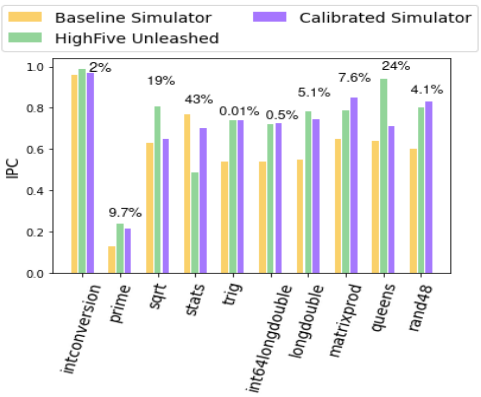

Figure 2: *Performance of Simulator vs HighFive Unleashed SoC for stress-ng benchmarks.*

cores are pipelined in-order processors[2] typically targeting workloads in user space. We base our simulated model and validation effort on single-core core workloads executing on the U54 core.

stress-ng Calibration

The stress-ng suite stresses a particular component or set of components of the system (in this case the CPU) by using *stressors*. The selected CPU class stressors were classified into three categories:

- **Control Intensive:** having complex control structures (eg. nested for-if-else) with at least 100 branches per 1000 instructions.
- **Memory Intensive:** having 20% or higher load/store instructions in the total op-code mix of the program. A maximum 30% instructions are load/store (for "queens" and "rand48").
- **Arithmetic Intensive:** having 20% or more Instructions using Integer/Float functional units.

The most significant attributes of the baseline and calibrated MinorCPU models are summarised in Table 1. Figure 2 depicts the IPC of Baseline, Calibrated simulator and Hardware for stress-ng benchmarks, along with the absolute percentage of IPC error in the Calibrated simulator. The simulator achieves a Mean Absolute Percentage Error in IPC of 11.8%.

Table 1: *Simulator Model- MinorCPU Attributes*

Component	Attribute	Simulated Model	
		Baseline	Calibrated
ReadMemFU	OpLat (cycles)	4	2
IntDivFU	OpLat (cycles)	33	19
Fetch unit	fetch1Tofetch2 BackwardDelay	1 (cycles)	0 (cycles)
Branch-Predictor	Type	Tournament	Multi perspective Perceptron

Experimental Setup

The simulated CPU frequency is the same as the HiFive SiFive Unleashed, running at 1GHz. Likewise, the simulated system uses 8GB of 2400MHz DDR4 RAM. The L1 and L2 caches are implemented using the classical cache models available in gXR5. They are 32KB, 8-way associative, and 2MB, 16-way associative, respectively. The software stack of both the simulated model and the HiFive Unleashed SoC includes the OpenSBI bootloader, the Linux kernel v5.8, and a 24GB buildroot filesystem.

[2] https://riscv.org/technical/specifications/

Results and Discussion

Five of the SPEC CPU2017 integer-rate benchmarks (out of 10 total applications) were successfully run on the validated simulator. The applications with a similar op-mix to stress-ng CPU class stressors have a mere 3.4% (500.*deepsjeng_r*) and 0.17% (541.*leela_r*) error in execution time. Figure 3 compares the execution time of SPEC CPU2017 applications running on the HiFive Unleashed SoC and on gXR5. A relatively large error in execution time for other applications is expected as they are much more memory intensive (i.e., they have nearly 40% Load/Store instructions), given only the MinorCPU model was calibrated. Calibration reduces the overall error in execution time from **44.3%** to **23.9%**.

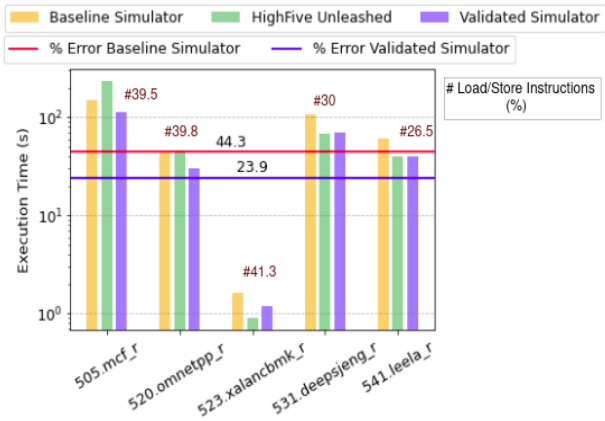

Figure 3: *Performance of Simulator vs actual hardware for SPEC CPU2017 suite*

This work has been partially supported by the EC H2020 WiPLASH project (GA No. 863337), the EC H2020 FVLLMONTI project (GA No. 101016776), and the ECO4AI project from HES-SO.

References

[1] *gXR5: A gem5-based full-system RISC-V simulator.* URL: https://www.epfl.ch/labs/esl/research/2d-3d-system-on-chip/gXR5.

[2] N. Binkert et al. "The gem5 simulator". In: *ACM SIGARCH computer architecture news* 39.2 (2011), pp. 1–7.

[3] Rajagopalan Desikan et al. "Measuring experimental error in microprocessor simulation". In: *ISCA* (2001).

[4] Quentin Huppert et al. "Memory hierarchy calibration based on real hardware in-order cores for accurate simulation". In: *DATE* (2021).

[5] Ayaz Akram and Lina Sawalha. "Validation of the GEM5 simulator for x86 architectures". In: *IEEE PMBS* (2019).

[6] Marco A Z Alves et al. "SiNUCA: A Validated Micro-Architecture Simulator". In: *2015 IEEE ICHPCC, IEEE ISCSS, and IEEE ICESS* (2015), pp. 605–610.