2024 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS) | 979-8-3503-7638-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISPASS61541.2024.00037

gem5-based evaluation of CVA6 SoC: Insights into the Architectural Design

Umer Shahid
Department of Electrical Engineering
University of Engineering & Technology
Lahore, Pakistan
umershahid@uet.edu.pk

Ayesha Ahmad
Department of Electrical Engineering
University of Engineering & Technology
Lahore, Pakistan
2021ee52@student.uet.edu.pk

Shanzay Wasim
Department of Electrical Engineering
University of Engineering & Technology
Lahore, Pakistan
2021ee154@student.uet.edu.pk

***Abstract*—Hardware design is both resource and time intensive. Hardware modelling and simulation are essential to cut down on these costs. This paper introduces a technique that addresses architectural disparities between System-on-Chips (SoCs) and gem5, a widely-used architectural simulator. The approach presents a solution for modeling application-class SoCs, exemplified by the construction and evaluation of CVA6 SoC on the gem5 platform.The model's performance was assessed using MiBench and RISC-V micro-benchmarks, showcasing less than 5% error in RISC-V microbenchmark test suite and less than 10% error in both SE and FS mode for the MiBench suite on the tuned CVA6 model. The aim of this research is to contribute to the understanding of the CVA6 performance, uncover any potential bottlenecks at the micro-architecture level, and to standardize the autotuning and modeling techniques of SoCs by leveraging machine learning algorithms, with the goal of identifying areas for further improvement in the CVA6 SoC model.**

***Index Terms*—RISC-V SoC, Performance Model, gem5**

I. Introduction

The gem5 simulator [1] serves as a modular framework which offers coverage of both system-level architecture and processor micro-architecture simulation. Gem5 was primarily developed for academia, but it is now extensively used in the industry. Gem5 can operate in two different modes: the Syscall Emulation (SE) mode and the Full system (FS). The SE mode focuses on simulating the CPU and memory system and does not emulate all the peripheral devices in a system. It only emulates Linux system calls, and thus only models user-mode code. While the FS mode emulates the entire hardware and its operation is closer to a virtual machine.

CVA6, previously known as Ariane [2] is an application class RISC-V core which is compliant with the RISC-V ISA specifications and comes in different configurations with varying levels of performance and features. The selection of gem5 for CVA6 modelling is due to its ability to boot unmodified Linux-based operating systems. The capability to generate checkpoints makes it capable of running simulations with various settings, further justifying this choice.

In this paper, the performance model of CVA6 is presented which has been developed using analysis of gem5 configurable parameters against the real hardware working on Kintex-7 FPGA. An earlier research paper [3] used gem5 to simulate the CVA6 processor but it primarily focused on the model validation. Unlike previous research, this paper provides an insight on the influence of SoC parameters on benchmarks and goes further to elucidate the specific aspects that each benchmark evaluates. Furthermore, it offers a thorough insight into the process of modeling an SoC in gem5. This includes an in-depth analysis of latency optimization methods, such as mathematical modeling and contour plots, and a comparative analysis of executions in both SE and FS modes.

II. Methodology

The RISC-V micro-benchmarks [4] and MiBench benchmark suite [5] were executed for CVA6 on two platforms: Kintex xc7k325tffg900- 2 FPGA (50MHz) and the gem5 simulator. These benchmarks were cross-compiled for SE mode and were added as payload in Linux image for the FS mode to test the model. All benchmarks were run for 100, 1k, 10k, 100k, and 1M times to understand the behavior of cache and branch predictor. The micro-architecture parameters affecting a particular micro-benchmark were documented. This information was used to understand which model parameters of micro-architectures were leading to discrepancies. The accuracy of the model was quantitatively determined by calculating the $\%$ errors in IPC between software and hardware simulation of each benchmark.

Parameters obtained directly from CVA6 documentation and RTL code were retained, while other parameters were adjusted based on their observed impacts on the IPC. To fine-tune the model, the effect of gem5 parameters on IPC behavior was assessed through graphical representations, particularly employing contour plots. To quantitatively evaluate the performance, errors in each benchmark, under various parameter settings, were aggregated using two metrics: the mean of absolute normalized errors and the root mean square (RMS). Employing these metrics, a linear regression model was constructed from the data points to discern optimal parameter values. The result of this optimization process was visually represented through a contour plot, providing a detailed portrayal of the interplay between parameters and performance in the gem5 model.

This approach led to an optimal configuration and the results obtained for the micro-benchmarks were then compared with the base model, as illustrated in Figure 2 while the errors of the MiBench suite were also minimized to less than 10% and

2766-0486/24/$31.00 ©2024 IEEE
DOI 10.1109/ISPASS61541.2024.00037

TABLE I
GEM5 PARAMETER OBSERVATIONS

Module	Parameter	Behavior
Core	Integer Op Lat	Inverse relation with all IPCs.
	Integer Multiplication Op Lat	Inverse relation with IPCs, trivial except on execution_int_mul_ind.
	Integer Memory Op Lat	Inverse relation with all IPCs.
CPU	fetch1LineSnapWidth	LineSnapWidth and LineWidth need to be equal and multiples of 4
	fetch1LineWidth	Inverse relation of IPC with LineSnapWidth and LineWidth
	fetch2CycleInput	IPC greater when False
	decodeToExecuteForwardDelay	Varies inversely with significant effect on all IPCs.
	decodeCycleInput	IPC greater when False
	executeInputWidth	Inverse relation with IPC.
	executeMemoryIssueLimit	Inverse relation with IPC
	executeBranchDelay	Inverse relation with IPC
L1Cache	data_latency	Varies inversely with significant effect on all IPCs for both I- and D-cache.
	mshrs	Direct relation but only slight effect on IPC for D-cache and no effect on IPC for I-cache.
	response_latency	Varies inversely with slight effect on all IPCs. It has a slightly greater effect on D- rather than I-cache.
	tag_latency	Varies inversely with significant effect on all IPCs for both I- and D-cache.
	tgts_per_mshr	No effect on IPC for both I- and D-cache.
Branch Predictor	globalCtrBits	Direct Relation with IPC
	globalPredictorSize	Inverse relation with IPCs, trivial except on control benchmarks

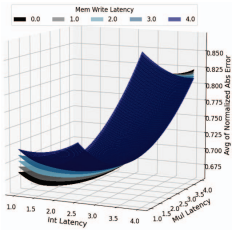

Fig. 1. Fine tuning results of micro-benchmarks

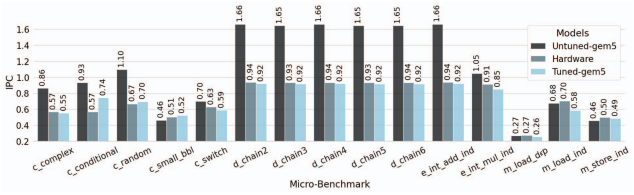

Fig. 2. Fine tuning results of micro-benchmarks

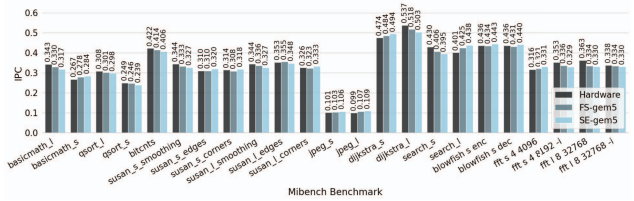

Fig. 3. Fine tuning results of the MiBench suite

8% for the SE and FS modes of gem5, respectively, as shown in Figure 3.

III. Observations and Results

The effect of various parameters on the IPC of the micro-benchmarks was studied. Some trends that were observed are documented in Table I. The three-dimensional relationships among the parameters were visually rendered. The contour plot, prominently featured in the accompanying Figure 1, served as an insightful representation, allowing for the clear depiction of how changes in basic integer operation latency, integer multiply latency, and memory write latency collectively influenced the system. Linear regression was employed to determine optimal parameter values and gain insight into their collective effects, providing both a graphical representation for identification and a nuanced understanding of inter-dependencies among the variables studied. The observed difference of 5% and 10% may be attributed to factors such as initial states of caches, simulation resolution, compiler optimizations, and inherent abstractions in simulations.

The errors of the MiBench suite were lesser for the FS than the SE mode of gem5, as shown in Figure 3. The major difference between FS and SE modes is that the FS mode simulates an OS that does scheduling while the SE mode only emulates the system calls. However in this study, the SE and FS mode results were similar attributed to the nature of the benchmarks, which primarily involved arithmetic operations and generated few system calls.

IV. Conclusion

This research introduced a comprehensive technique for modelling application-class SoCs within the FS and SE modes of the gem5 simulator, with a focus on the CVA6 SoC. By optimizing model parameters and conducting detailed benchmark analyses, CVA6 gem5 model was tuned. The IPC errors were minimized to less than 10% and 5% for the Mibench suite and the RISCV micro-benchmarks, respectively. The findings of this study serve as a foundation for future research aimed at enhancing the accuracy and applicability of simulation-based hardware design methodologies.

REFERENCES

[1] N. Binkert et al., "The gem5 simulator," ACM SIGARCH Computer Architecture News, vol. 39, no. 2, pp. 1–7, May 2011, doi: https://doi.org/10.1145/2024716.2024718.

[2] F. Zaruba and L. Benini, "The Cost of Application-Class Processing: Energy and Performance Analysis of a Linux-Ready 1.7-GHz 64-Bit RISC-V Core in 22-nm FDSOI Technology," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 11, pp. 2629–2640, Nov. 2019, doi: https://doi.org/10.1109/TVLSI.2019.2926114.

[3] P. Ravenel, A. Perais, B. De Dinechin, and F. Pétrot, "A gem5-based CVA6 Framework for Microarchitectural Pathfinding," presented at the RISC-V Summit Europe, Jun. 2023. Accessed: Aug. 28, 2023. [Online]. Available: https://riscv-europe.org/media/proceedings/posters/2023-06-06-Pierre-RAVENEL-abstract.pdf

[4] "riscv-validation," GitHub, Jul. 24, 2023. https://github.com/darchr/riscv-validation (accessed Sep. 27, 2023).

[5] "mibench," GitHub, Jul. 24, 2023. https://github.com/aakahlow/mibench (accessed Sep. 27, 2023).