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

Akanksha Chaudhari Matthew D. Sinclair
{akankshac, sinclair}@cs.wisc.edu

1 Motivation

The slowdown of process technology-driven improvements [3, 15] has accelerated the shift toward heterogeneous computing systems, where conventional general-purpose cores are increasingly combined with GPUs and specialized accelerators to continue scaling performance and energy efficiency gains. However, as these systems grow more diverse, architectural design and system-level optimization become significantly complex. Fully leveraging the benefits of such architectures demands rigorous early-stage exploration using validated, cycle-level, full-system simulation frameworks that capture both component behavior and cross-layer interactions.

The gem5 simulator [2, 5] has provided this functionality for both CPUs and GPUs [4, 8–10]. However, accelerator modeling within gem5 remains fragmented. Frameworks such as gem5-SALAM [12, 13] provide high-fidelity accelerator simulation atop gem5 v21.1, but operate outside the mainline development and remain limited to modeling accelerators in isolation. Consequently, they cannot leverage improvements introduced in recent versions of gem5, nor can mainline users test and validate their changes do not break accelerator support. To bridge this gap, we are integrating and merging SALAM's accelerator modeling infrastructure into the latest gem5 mainline, enabling tightly coupled simulation of accelerators alongside gem5's existing CPU and GPU components within a unified, full-system framework. This integration is critical for advancing early-stage architectural exploration, enabling researchers to systematically evaluate design trade-offs, study cross-layer interactions, and co-optimize compute, memory, and communication components for emerging heterogeneous computing platforms.

2 Implementation & Methodology

To integrate gem5-SALAM into the gem5 mainline, we used three key thrusts to make it fully compatible with gem5's latest version (develop branch, v24.1). First, we integrated key components from gem5-SALAM into gem5 including: LLVMInterface (cycle-level datapath simulation), CommInterface (exposing accelerators to the rest of the system), as well as various memory organizations including scratchpads, DMAs, and stream buffers, AccCluster (to group accelerators and private memories). We also extended gem5-SALAM's hardware generation scripts to automate generating source files for functional units and instruction configuration objects based on user-defined hardware profiles. Finally, we refactored the CACTI-SALAM toolchain in Python to improve its compatibility with gem5, modifying path handling, argument interfaces, and error handling mechanisms, while enabling robust timing and energy estimation for scratchpad memories using CACTI's file-based configuration methodology. Together, these changes provide native gem5 support for accelerator modeling with cycle-level timing, full-system memory interaction, and scalable design space exploration.

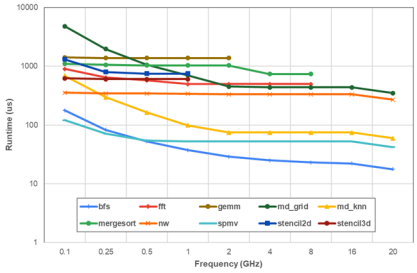

Figure 1: High frequency accelerator gem5 results.

Second, we refactored the integrated accelerator components to comply with gem5's evolution since 2021. This included adopting modern port binding APIs, replacing deprecated constructor patterns, aligning port interfaces with gem5's SimObject conventions, updating accelerator latency modeling conventions, changing address range specifications to follow gem5's inclusive-exclusive semantics, updating environment configurations and compiler issues, and formalizing LLVM integration by extending scons.

Finally, we validated our integration using gem5's pre-commit checks and regression test suite to ensure gem5's functionality remained intact. To verify the integrated framework's correctness, we adapted gem5-SALAM's system validation tests to run within the unified setup and cross-validated simulation outputs against those produced by the original gem5-SALAM baseline. These comparisons helped confirm functional equivalence.

Preliminary Results: To demonstrate the value of our changes, we conducted preliminary experiments to evaluate the integrated support for a variety of high frequency accelerators running up to 20 GHz [11]. Although traditionally accelerators are not run at such frequencies, recent work indicated they may be strong candidates for high frequencies [1, 6, 7, 14]. Figure 1 shows our results when sweeping frequencies from 0.1-20 GHz (some bars are incomplete because the accelerator currently cannot run at those frequencies). Configured to use a local scratchpad to eliminate external memory bottlenecks, the accelerators often exhibit progressively reduced runtimes as frequency increases, especially near 20 GHz. These findings establish an upper bound on achievable speedup in compute-bound scenarios for accelerator design studies targeting extreme frequency scaling.

3 Conclusion & Future Work

Future systems will be even more heterogeneous, combining CPUs, GPUs, and myriad accelerators to improve overall system efficiency. Simulation tools must keep pace and enable early-stage co-design for these systems. By integrating gem5-SALAM into gem5's mainline and open-sourcing this support (ongoing), our work facilitates

'6th gem5 Users' Workshop 2025', June 21–25, 2025, Tokyo, Japan
2025. ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

system-level studies on task placement, scheduling, memory hierarchy design, and interconnect behavior. Our high-frequency accelerator study highlights gem5's potential for early-stage exploration. We are also working on extending our support beyond ARM to other gem5-supported ISAs to further broaden its applicability to emerging accelerator-rich platforms.

Acknowledgments

This work is supported in part by the Semiconductor Research Corporation and by the DOE's Office of Science, Office of Advanced Scientific Computing Research through EXPRESS: 2023 Exploratory Research for Extreme Scale Science.

References

[1] James Ang, Andrew A. Chien, Simon David Hammond, Adolfy Hoisie, Ian Karlin, Scott Pakin, John Shalf, and Jeffrey S. Vetter. 2022. Reimagining Codesign for Advanced Scientific Computing: Report for the ASCR Workshop on Reimagining Codesign. *DOE ASCR Workshop on Reimagining Codesign* (4 2022), 77 pages. https://doi.org/10.2172/1822199

[2] Nathan Binkert, Bradford Beckmann, Gabriel Black, Steven K. Reinhardt, Ali Saidi, Arkaprava Basu, Joel Hestness, Derek R. Hower, Tushar Krishna, Somayeh Sardashti, Rathijit Sen, Korey Sewell, Muhammad Shoaib, Nilay Vaish, Mark D. Hill, and David A. Wood. 2011. The gem5 simulator. *ACM SIGARCH Computer Architecture News* 39, 2 (2011), 1–7.

[3] Daniel S. Green. 2018. *Heterogeneous Integration at DARPA: Pathfinding and Progress in Assembly Approaches.* Technical Report. DARPA.

[4] Anthony Gutierrez, Bradford M. Beckmann, Alexandru Dutu, Joseph Gross, Michael LeBeane, John Kalamatianos, Onur Kayiran, Matthew Poremba, Brandon Potter, Sooraj Puthoor, Matthew D. Sinclair, Michael Wyse, Jieming Yin, Xianwei Zhang, Akshay Jain, and Timothy Rogers. 2018. Lost in Abstraction: Pitfalls of Analyzing GPUs at the Intermediate Language Level. In *IEEE International Symposium on High Performance Computer Architecture (HPCA).* IEEE Computer Society, Washington, DC, USA, 608–619.

[5] Jason Lowe-Power, Abdul Mutaal Ahmad, Ayaz Akram, Mohammad Alian, Rico Amslinger, Matteo Andreozzi, Adrià Armejach, Nils Asmussen, Srikant Bharadwaj, Gabe Black, Gedare Bloom, Bobby R. Bruce, Daniel Rodrigues Carvalho, Jeronimo Castrillon, Lizhong Chen, Nicolas Derumigny, Stephan Diestelhorst, Wendy Elsasser, Marjan Fariborz, Amin Farmahini-Farahani, Pouya Fotouhi, Ryan Gambord, Jayneel Gandhi, Dibakar Gope, Thomas Grass, Bagus Hanindhito, Andreas Hansson, Swapnil Haria, Austin Harris, Timothy Hayes, Adrian Herrera, Matthew Horsnell, Syed Ali Raza Jafri, Radhika Jagtap, Hanhwi Jang, Reiley Jeyapaul, Timothy M. Jones, Matthias Jung, Subash Kannoth, Hamidreza Khaleghzadeh, Yuetsu Kodama, Tushar Krishna, Tommaso Marinelli, Christian Menard, Andrea Mondelli, Tiago Mück, Omar Naji, Krishnendra Nathella, Hoa Nguyen, Nikos Nikoleris, Lena E. Olson, Marc Orr, Binh Pham, Pablo Prieto, Trivikram Reddy, Alec Roelke, Mahyar Samani, Andreas Sandberg, Javier Setoain, Boris Shingarov, Matthew D. Sinclair, Tuan Ta, Rahul Thakur, Giacomo Travaglini, Michael Upton, Nilay Vaish, Ilias Vougioukas, Zhengrong Wang, Norbert Wehn, Christian Weis, David A. Wood, Hongil Yoon, and Éder F. Zulian. 2020. The gem5 Simulator: Version 20.0+. arXiv:2007.03152 [cs.AR]

[6] Dongmoon Min, Ilkwon Byun, Gyu-Hyeon Lee, Seongmin Na, and Jangwoo Kim. 2020. CryoCache: A Fast, Large, and Cost-Effective Cache Architecture for Cryogenic Computing. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS).* Association for Computing Machinery, New York, NY, USA, 449–464. https://doi.org/10.1145/3373376.3378513

[7] Kunal Pai, Anusheel Nand, and Jason Lowe-Power. 2024. Potential and Limitation of High-Frequency Cores and Caches. arXiv:2408.03308 [cs.AR] https://arxiv.org/abs/2408.03308

[8] Vishnu Ramadas, Matthew Poremba, Bradford M. Beckmann, and **M. D. Sinclair**. 2023. Improving gem5's GPU FS Support. In *The 5th gem5 Users' Workshop.* 4 pages.

[9] Vishnu Ramadas, Matthew Poremba, Bradford M. Beckmann, and **M. D. Sinclair**. 2024. Simulation Support for Fast and Accurate Large-Scale GPGPU and Accelerator Workloads. In *3rd Open-Source Computer Architecture Research Workshop (OSCAR).* 4 pages.

[10] Vishnu Ramadas and **M. D. Sinclair**. 2024. Simulating Machine Learning Models at Scale. In *SRC TECHCON.* 7 pages.

[11] Brandon Reagen, Robert Adolf, Yakun Sophia Shao, Gu-Yeon Wei, and David Brooks. 2014. MachSuite: Benchmarks for accelerator design and customized architectures. In *2014 IEEE International Symposium on Workload Characterization (IISWC).* 110–119. https://doi.org/10.1109/IISWC.2014.6983050

[12] Samuel Rogers, Joshua Slycord, Mohammadreza Baharani, and Hamed Tabkhi. 2020. gem5-SALAM: A System Architecture for LLVM-based Accelerator Modeling. In *53rd Annual IEEE/ACM International Symposium on Microarchitecture.* 471–482. https://doi.org/10.1109/MICRO50266.2020.00047

[13] Zephaniah Spencer, Samuel Rogers, Joshua Slycord, and Hamed Tabkhi. 2024. Expanding Hardware Accelerator System Design Space Exploration with gem5-SALAMv2. *Journal of Systems Architecture* 154 (2024), 103211. https://doi.org/10.1016/j.sysarc.2024.103211

[14] Swamit S. Tannu, Poulami Das, Michael L. Lewis, Robert Krick, Douglas M. Carmean, and Moinuddin K. Qureshi. 2019. A case for superconducting accelerators. In *Proceedings of the 16th ACM International Conference on Computing Frontiers* (Alghero, Italy) *(CF '19).* Association for Computing Machinery, New York, NY, USA, 67–75. https://doi.org/10.1145/3310273.3321561

[15] Thomas N. Theis and H.-S. Philip Wong. 2017. The End of Moore's Law: A New Beginning for Information Technology. *Computing in Science & Engineering* 19, 2 (2017), 41–50. https://doi.org/10.1109/MCSE.2017.29