Extensible Runtime Thermal Modeling in gem5

Akanksha Chaudhari, Alex Smith, Matthew D. Sinclair
University of Wisconsin-Madison
aschaudhari@wisc.edu adsmith@cs.wisc.edu sinclair@cs.wisc.edu

I. Introduction

Power and thermal constraints are increasingly shaping chip and architectural design requirements as modern systems pack more compute into smaller form factors [1]. With Dennard Scaling ended, performance scaling now relies increasingly on wider parallelism, advanced packaging, and 3D integration, thereby increasing both power density and thermal stress on the system. Operating power now ranges from 1.4 kW for a single AMD MI355X GPU [2] to 135 kW for NVIDIA's liquid-cooled rack-scale GB300 NVL72 [3], making thermals a first-order design constraint alongside performance and power. Thermally aware architecture exploration is therefore essential to designing next-generation systems. Such exploration requires tools that connect workload-driven activity to temperature. State-of-the-art thermal solvers [4]–[6] estimate temperature from power traces and physical inputs such as floorplans, layer stacks, and cooling parameters; however, despite the fidelity of these solvers, they often operate at physical and device-level detail, which is a lower abstraction level compared to most architectural tools. Prior frameworks [7]–[9] integrate these solvers with architectural simulators to improve usability but do not perform any in-situ thermal analysis. While gem5 includes native support for runtime thermal modeling [10], using it directly requires users to explicitly instantiate the RC thermal network, including its thermal nodes and couplings, which is impractical for modern systems. Moreover, changes to the thermal model, such as modifying the numerical methods used by the solver, require modifications to gem5's C++ modules. We take prior work that addresses a similar extensibility problem and extend it to decouple gem5's thermal modeling abstractions from the runtime thermal model [11], [12].

II. Implementation and Methodology

Implementation: Our changes build on top of gem5's native power and thermal modeling support. We preserve gem5's native thermal support by including *thermal domains*, which represent groups of components whose temperature is modeled, and *thermal models*, which hold the temperature state updated by the thermal solver. Then, we build on top of the gem5's power model interface by extending prior work [11]. Our thermal modeling interface includes configuration code that lets users configure how SimObjects are mapped to a thermal domain. These mappings serve as an architectural description, akin to a floorplan, on how components are grouped under thermal domains that are identified and used by the thermal solver. Since thermal models require power values as inputs, at each thermal step, our interface obtains power sampled from all thermal domains, checks that samples belong to a consistent runtime epoch, and passes the power and temperature for each thermal domain to a Python-based thermal solver. The solver then outputs updated temperatures for each thermal domain and these are propagated to the matching thermal nodes for each domain as runtime thermal feedback. This design keeps the simulator-side interface solver-agnostic so that new thermal backends can be swapped in through Python without rewriting gem5's core thermal model.

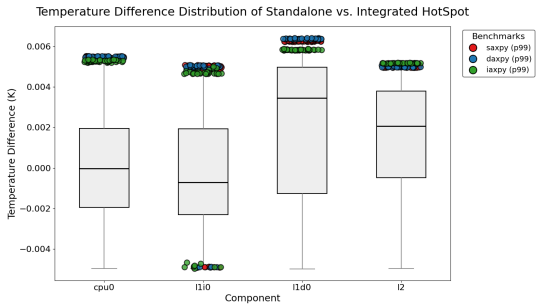

Fig. 1: Signed temperature difference by component; markers show per-benchmark P99 difference.

Methodology: To demonstrate the interface, we couple runtime-sampled McPAT-based power models [11] to a Python reimplementation of the HotSpot 7.0 [4], [13] thermal solver on a simulated Arm Cortex-A9 platform comprising a timing CPU model, private 32 KiB L1I/L1D caches, a shared 1 MiB L2, and LPDDR3 main memory. We evaluate three representative kernels, `saxpy`, `iaxpy`, and `daxpy`, sampling power and temperature every 0.25 ms. The HotSpot block model is configured with four thermal domains, a cold-start initial condition, and a 300 K ambient temperature. We validate the runtime temperature traces against a standalone HotSpot reference driven by the same power traces.

Results: Figure 1 shows the distribution of signed temperature differences between our integrated model and the standalone HotSpot tool. Across all workloads and domains, the integrated model tracks standalone HotSpot closely with an overall mean absolute difference of ≈ 46 mK. Most samples differ by only a few tens of milli-Kelvin, and the per-benchmark P99 markers remain small across components. The largest differences occur for L2, which has the highest modeled static power and is therefore more sensitive to small numerical, block-coupling, and output-precision differences between the runtime and standalone paths.

III. Conclusion and Future Work

This work shows that in-situ, workload-driven thermal modeling can be integrated into gem5 without binding to a specific thermal backend. Validation against standalone HotSpot proves the interface reproduces consistent transient temperature behavior. The framework enables architects to jointly study workload behavior, power, cooling assumptions, package effects, and die temperature, thus treating thermals as a first-order design constraint. Future work includes publicly releasing our implementation to gem5 and studying end-to-end workloads and thermal-management scenarios.

Acknowledgments

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

References

[1] T. Mudge, "Power: A First-class Architectural Design Constraint," *Computer*, vol. 34, no. 4, pp. 52–58, 2001.

[2] AMD, "AMD Instinct MI355X Platform," https://www.amd.com/en/products/accelerators/instinct/mi350/mi355x.html, 2025, accessed: 2025-11.

[3] NVIDIA, "NVIDIA GB300 NVL72," https://www.nvidia.com/en-us/data-center/gb300-nvl72/, 2025, accessed: 2025-11.

[4] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "HotSpot: A Compact Thermal Modeling Methodology for Early-Stage VLSI Design," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 14, no. 5, pp. 501–513, 2006.

[5] A. Sridhar, A. Vincenzi, M. Ruggiero, T. Brunschwiler, and D. Atienza, "3D-ICE: Fast Compact Transient Thermal Modeling for 3D ICs with Inter-Tier Liquid Cooling," in *IEEE/ACM International Conference on Computer-Aided Design*, ser. ICCAD, 2010, pp. 463–470.

[6] L. Pfromm, A. Kanani, H. Sharma, P. Solanki, E. Tervo, J. Park, J. R. Doppa, P. P. Pande, and U. Y. Ogras, "MFIT: Multi-FIdelity Thermal Modeling for 2.5D and 3D Multi-Chiplet Architectures," *arXiv preprint arXiv:2410.09188*, 2025.

[7] A. Pathania and J. Henkel, "HotSniper: Sniper-Based Toolchain for Many-Core Thermal Simulations in Open Systems," *IEEE Embedded Systems Letters*, vol. 11, no. 2, pp. 54–57, 2018.

[8] L. Siddhu, R. Kedia, S. Pandey, M. Rapp, A. Pathania, J. Henkel, and P. R. Panda, "CoMeT: An Integrated Interval Thermal Simulation Toolchain for 2D, 2.5D, and 3D Processor-Memory Systems," *ACM Transactions on Architecture and Code Optimization*, vol. 19, no. 3, pp. 1–25, 2022.

[9] R. Wang, Z. Wang, T. Lin, J. M. Raby, M. R. Stan, and X. Guo, "Cool-3D: An End-to-End Thermal-Aware Framework for Early-Phase Design Space Exploration of Microfluidic-Cooled 3DICs," *arXiv preprint arXiv:2503.07297*, 2025.

[10] Q. Forcioli, "Modeling of micro-architecture for security with gem5," Ph.D. dissertation, Institut Polytechnique de Paris, 2024.

[11] A. Smith, B. Bruce, J. Lowe-Power, and M. D. Sinclair, "Designing Generalizable Power Models For Open-Source Architecture Simulators," in *3rd Open-Source Computer Architecture Research Workshop*, ser. OSCAR, 2024.

[12] A. Smith and M. D. Sinclair, "Implementing Support for Extensible Power Modeling in gem5," in *6th gem5 Users Workshop*, June 2025.

[13] J.-H. Han, R. E. West, K. Skadron, and M. R. Stan, "Thermal Simulation of Processing-in-Memory Devices Using HotSpot 7.0," in *27th International Workshop on Thermal Investigations of ICs and Systems*, ser. THERMINIC, 2021, pp. 1–5.