Designing Generalizable Power Models For Open-Source Architecture Simulators

Alex Smith*, Bobby R. Bruce†, Jason Lowe-Power†, Matthew D. Sinclair*
*University of Wisconsin-Madison, †University of California, Davis
adsmith@cs.wisc.edu bbruce@ucdavis.edu jlowepower@ucdavis.edu sinclair@cs.wisc.edu

I. Introduction

As transistor sizes shrink, power consumption has increasingly become a first-class design constraint [1] for modern systems. Although co-optimizing power and performance is important for a wide range of applications, including high performance computing (HPC) and graph analytics, artificial intelligence (AI) in particular is driving future system requirements. In recent years, AI has transformed society with significant improvements in speech recognition [2], image classification [3]–[8], machine translation [9], autonomous agents [10], language processing [11], [12], text generation [13], and other tasks [14]. This tremendous transformative effect has been enabled by a virtuous synergy of (1) better hardware systems, (2) larger datasets, and (3) improved AI models (e.g., Transformers) and algorithms that further benefit from more efficient hardware and larger datasets.

However, meeting the computational needs of AI applications and other important workloads is challenging. These applications are ravenous, often requiring exponentially more compute [15]. With the slowing of Moore's Law and end of Dennard's Scaling, systems are increasingly turning towards heterogeneous accelerators to scale performance, especially for AI workloads. Accordingly, systems must also optimize their increasingly heterogeneous systems and applications for power consumption, without compromising performance.

To drive these efforts, we require a credible open-source infrastructure to study novel improvements to the existing state-of-the-art and evaluate the potential of radical computer system changes. Traditionally, developers rely on simulation and modeling techniques to estimate a prototypes' performance and power consumption. While existing tools provide accurate performance predictions, the tools for modeling power are lacking. Low-level Spice models are accurate, but require proprietary information and scale poorly to increasingly large, complex systems. Tools built by extrapolating first-principles models (e.g., CACTI [16], [17] and McPAT [18], [19]) have not been updated in 8 years and are no longer representative. Likewise, power analysis tools dependent on design tape outs are time consuming, expensive, and prevent co-design from happening early in the design process. Finally, state-of-the-art tools like AccelWattch [20], [21], analytical models [22], and machine learning models to predict power consumption [23]–[25] often do not generalize, giving inaccurate results for even minor configuration perturbations.

Part of the challenge is that these power models are often tightly coupled to specific architectures. Thus, we need accurate, generalizable, and usable power models to enable early-stage research and development of next generation systems. Accordingly, we propose developing new methodologies to make it as easy to model power consumption as it is to model performance (Figure 1). To address these challenges, we propose creating a flexible power methodology that allows architects to easily incorporate different power models at a fine granularity in open-source simulators. In particular, we focus on integrating this support into gem5, a widely used, open-source, cycle-level computer system simulator, although the ideas can also be applied to other simulators.

II. Background

At its core, gem5 contains an event-driven simulation engine [26], [27]. On top of this simulation engine gem5 implements a large number of models for system components for CPUs (out-of-order designs, in-order designs, and others), AMD and ARM GPUs [28], accelerators [29], [30], various memories, on-chip interconnects, coherent caches, I/O devices, and many others. Moreover, gem5 provides two modes: Syscall Emulation (SE) and Full System (FS). SE mode simulates an application's user mode code in detail but emulates the OS instead of simulating it in detail. Conversely, FS mode simulates both the OS and user mode code in detail, allowing users to study OS-architecture interactions.

The gem5 simulator also has some support for power and thermal modeling [31]. For example, prior work has added power models into gem5 for DRAM [32], networks-on-chip [33], [34], ARM CPUs [35], or integrated McPAT [36]. However, like CACTI and McPAT themselves, some of these models have not been updated in many years. Thus, while these additions represent useful building blocks, none of them provide support for modeling power consumption for the entire system gem5 models. Moreover, many are tied to specific models (e.g., McPAT) or vendors, limiting their flexibility. In comparison, we seek to create a power modeling framework that decouples **which power model to use** from **how open source simulators support power models**.

III. Proposed Approach

Figure 1 demonstrates our overall approach. Like prior work [19] we propose a hierarchical power model where the overall system power combines the sum of the main system components, each of which may have one or more levels of sub-components. The user determines how many

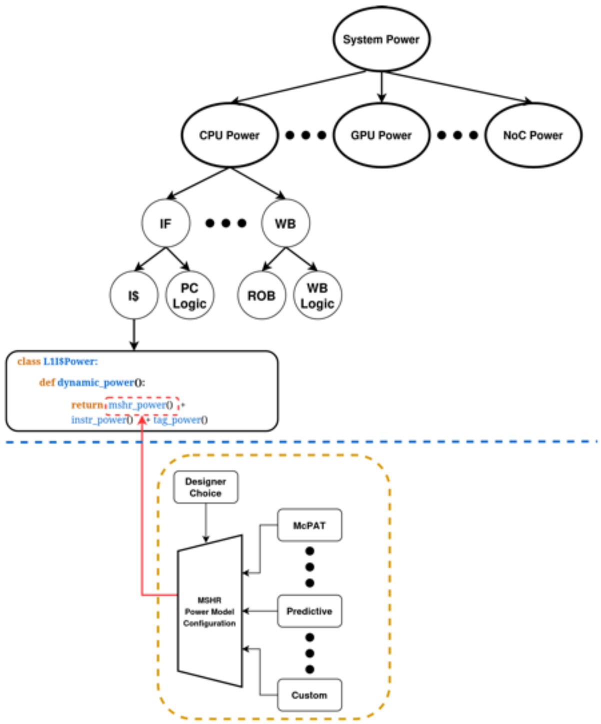

Fig. 1: Proposed Hierarchical Power Model. Our changes are under the blue line.

levels gem5 should output. However, unlike prior work our approach separates how simulators (e.g., gem5) model power consumption from which power model (e.g., CACTI, McPAT, Spice) is used. We do this by creating a new interface (dotted blue line, discussed further in **Interface**) that connects gem5's hierarchical power model (**gem5 Side**) to one or more power models that a user may want to utilize to model an architecture's power consumption in gem5 (**Power Model Side**).

gem5 Side: If a user wants to define a new power model in gem5, they must use gem5's Python scripting interface to define the behavior of each component's power model. To do this, each component must have a class describing its power states (e.g., separate classes for the power behavior when a component is on, off, or clock gated). For example, in Figure 1 the user defines `L1I$Power` class for the L1 instruction cache. Within this class the dynamic power is the summation of its' three key components: MSHR accesses (for misses) and data/tag accesses (for hits and misses). Similarly, at higher levels of the hierarchical power model, the user must specify what the key sub-components are and how they should be summed together. This allows gem5's to output power information (stats) in summarized form for the higher levels and detailed form for the lower levels.

Power Model Side: We will also integrate or create a variety of power models. Some of these power models could be the state-of-the-art approaches like AccelWattch, CACTI, McPAT, or Spice. However, researchers can also to create their own models (e.g., for a new accelerator). Ultimately, the model must provide information (via the **Interface**) about the dynamic power and static power for each system component. Collectively, this information will be passed via the interface to each component's power model class. For example, in Figure 1, the power model tells the `L1I$Power` classes' `dynamic_power` function about how the CPU's instruction cache should model the power for accessing an MSHR entry (`mshr_power`), the power for accessing the instruction data in the cache line (`instr_power`), and the energy for accessing the corresponding tag in the cache line (`tag_power`). Although we envision the power models supplying this information such that gem5 can utilize this information on a per access basis, since our approach flexibly represents the power model in Python, alternatives are also possible. Similar to gem5's validated *known good models* [27] for system configuration, we will create validated *known good power models* for a variety of these approaches.

Interface: To separate the gem5 implementation from the power model, in Python we a flag that takes input from the user (*Designer Choice*) to pick between different *known good power models* that have been integrated into gem5. Thus, if a user wants to utilize two different power models with the same system configuration, all they need to do is change the *Designer Choice* flag. Likewise, users can also choose to utilize different power models for different components (e.g., for different CPUs in the system).

Overall, our approach has two main benefits. First, since defining a power model per component uses gem5's scripting interface, researchers can change the underlying power model by changing the Python code. For example, if a user wants to change the Functional Unit or ALU power models, they could do change the corresponding dynamic power values in Python. Likewise, if the user wants to create a new power model, they can add this as another option the *Design Choice* logic can select, without needing to modify the underlying gem5 simulator. Furthermore, since it is simpler to implement power modeling for components of interest, the designer can specify what granularity to model power and report results. Thus, users can either create their own power models or select from our *known good power models* like McPAT. Accordingly, researchers are neither restricted to certain power modeling tools, nor required to make their own. Instead they can select the power model which best suits their needs.

IV. Conclusion

Co-designing systems for both power and performance is paramount. High fidelity, open source tools like gem5 are critical in this process, because they allow researchers to determine how effective their optimizations early in the design stage. However, these tools are facing challenges from both increasingly heterogeneous systems and power modeling tools that are struggling to keep pace. Accordingly, we propose to **decouple** how these tools model power from the power models using an open source, flexible Python-based interface. This allows users to integrate both existing and novel power models into gem5, without requiring complex simulator changes. In turn, this enables researchers to more easily develop efficient power models for these increasingly heterogeneous systems.

References

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

[2] W. Xiong, J. Droppo, X. Huang, F. Seide, M. L. Seltzer, and A. Stolcke, "Toward Human Parity in Conversational Speech Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, pp. 2410–2423, Sept 2017.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12. USA: Curran Associates Inc., 2012, pp. 1097–1105. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999134.2999257

[5] M. Lin, Q. Chen, and S. Yan, "Network In Network," in *2nd International Conference on Learning Representations*, ser. ICLR, Y. Bengio and Y. LeCun, Eds. OpenReview.net, 2014. [Online]. Available: http://arxiv.org/abs/1312.4400

[6] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *3rd International Conference on Learning Representations*, ser. ICLR, Y. Bengio and Y. LeCun, Eds. OpenReview.net, 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR. Piscataway, NJ, USA: IEEE Press, 2015, pp. 1–9.

[8] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR. Piscataway, NJ, USA: IEEE Press, 2016, pp. 2818–2826.

[9] H. Hassan, A. Aue, C. Chen, V. Chowdhary, J. Clark, C. Federmann, X. Huang, M. Junczys-Dowmunt, W. Lewis, M. Li, S. Liu, T.-Y. Liu, R. Luo, A. Menezes, T. Qin, F. Seide, X. Tan, F. Tian, L. Wu, S. Wu, Y. Xia, D. Zhang, Z. Zhang, and M. Zhou, "Achieving Human Parity on Automatic Chinese to English News Translation," 2018.

[10] S.-C. Lin, Y. Zhang, C.-H. Hsu, M. Skach, M. E. Haque, L. Tang, and J. Mars, "The Architectural Implications of Autonomous Driving: Constraints and Acceleration," in *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS. New York, NY, USA: ACM, 2018, pp. 751–766. [Online]. Available: http://doi.acm.org/10.1145/3173162.3173191

[11] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, ser. NAACL-HLT. Association for Computational Linguistics, 2019, pp. 4171–4186. [Online]. Available: https://doi.org/10.18653/v1/n19-1423

[12] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, 2019.

[13] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models are Few-Shot Learners," in *Advances in Neural Information Processing Systems*, ser. NeurIPS, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Red Hook, NY, USA: Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[14] N. Benaich and I. Hogarth, "State of AI Report 2022," https://www.stateof.ai/, 2022.

[15] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," 2020.

[16] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "Cacti 6.0: A tool to model large caches," *HP laboratories*, vol. 27, p. 28, 2009.

[17] P. Shivakumar and N. Jouppi, *CACTI 5.0. Technical Report HPL-2007-167*, HP Laboratories, 2007.

[18] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures," in *Proceedings of the 42nd annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO, 2009, pp. 469–480.

[19] ——, "The McPAT Framework for Multicore and Manycore Architectures: Simultaneously Modeling Power, Area, and Timing," *ACM Transactions on Architecture & Code Optimization*, vol. 10, no. 1, pp. 5:1–5:29, Apr. 2013. [Online]. Available: http://doi.acm.org/10.1145/2445572.2445577

[20] J. Leng, T. Hetherington, A. ElTantawy, S. Gilani, N. S. Kim, T. M. Aamodt, and V. J. Reddi, "GPUWattch: enabling energy optimizations in GPGPUs," in *Proceedings of the 40th Annual International Symposium on Computer Architecture*, ser. ISCA '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 487–498. [Online]. Available: https://doi.org/10.1145/2485922.2485964

[21] V. Kandiah, S. Peverelle, M. Khairy, A. Manjunath, J. Pan, T. G. Rogers, T. M. Aamodt, and N. Hardavellas, "AccelWattch: A Power Modeling Framework for Modern GPUs," in *Proceedings of the 54th IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO, October 2021.

[22] A. Stillmaker and B. Baas, "Scaling Equations for the Accurate Prediction of CMOS Device Performance from 180nm to 7nm," *Integration*, vol. 58, pp. 74–81, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167926017300755

[23] W. Lee, Y. Kim, J. H. Ryoo, D. Sunwoo, A. Gerstlauer, and L. K. John, "PowerTrain: A learning-based calibration of McPAT power models," in *IEEE/ACM International Symposium on Low Power Electronics and Design*, ser. ISLPED. IEEE, 2015, pp. 189–194.

[24] J. S. Shim, B. Han, Y. Kim, and J. Kim, "DeepPM: Transformer-based Power and Performance Prediction for Energy-aware Software," in *Design, Automation & Test in Europe Conference & Exhibition*, ser. DATE. IEEE, 2022, pp. 1491–1496.

[25] G. Wu, J. L. Greathouse, A. Lyashevsky, N. Jayasena, and D. Chiou, "GPGPU Performance and Power Estimation Using Machine Learning," in *IEEE 21st International Symposium on High Performance Computer Architecture*, ser. HPCA, 2015, pp. 564–576.

[26] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, pp. 1–7, 2011.

[27] J. Lowe-Power, A. M. Ahmad, A. Akram, M. Alian, R. Amslinger, M. Andreozzi, A. Armejach, N. Asmussen, S. Bharadwaj, G. Black, G. Bloom, B. R. Bruce, D. R. Carvalho, J. Castrillon, L. Chen, N. Derumigny, S. Diestelhorst, W. Elsasser, M. Fariborz, A. Farmahini-Farahani, P. Fotouhi, R. Gambord, J. Gandhi, D. Gope, T. Grass, B. Hanindhito, A. Hansson, S. Haria, A. Harris, T. Hayes, A. Herrera, M. Horsnell, S. A. R. Jafri, R. Jagtap, H. Jang, R. Jeyapaul, T. M. Jones, M. Jung, S. Kannoth, H. Khaleghzadeh, Y. Kodama, T. Krishna, T. Marinelli, C. Menard, A. Mondelli, T. Mück, O. Naji, K. Nathella, H. Nguyen, N. Nikoleris, L. E. Olson, M. Orr, B. Pham, P. Prieto, T. Reddy, A. Roelke, M. Samani, A. Sandberg, J. Setoain, B. Shingarov, M. D. Sinclair, T. Ta, R. Thakur, G. Travaglini, M. Upton, N. Vaish, I. Vougioukas, Z. Wang, N. Wehn, C. Weis, D. A. Wood, H. Yoon, and Éder F. Zulian, "The gem5 simulator: Version 20.0+," 2020.

[28] A. Gutierrez, B. M. Beckmann, A. Dutu, J. Gross, M. LeBeane, J. Kalamatianos, O. Kayiran, M. Poremba, B. Potter, S. Puthoor, M. D. Sinclair, M. Wyse, J. Yin, X. Zhang, A. Jain, and T. Rogers, "Lost in Abstraction: Pitfalls of Analyzing GPUs at the Intermediate Language Level," in *2018 IEEE International Symposium on High Performance Computer Architecture*, ser. HPCA, Feb 2018, pp. 608–619.

[29] S. Rogers, J. Slycord, M. Baharani, and H. Tabkhi, "gem5-SALAM: A System Architecture for LLVM-based Accelerator Modeling," in *53rd Annual IEEE/ACM International Symposium on Microarchitecture*, 2020, pp. 471–482.

[30] Y. S. Shao, S. L. Xi, V. Srinivasan, G.-Y. Wei, and D. Brooks, "Co-designing accelerators and SoC interfaces using gem5-Aladdin," in *49th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO, 2016, pp. 1–12.

[31] gem5, "Power and Thermal Model," https://www.gem5.org/documentation/general_docs/thermal_model, 2024.

[32] R. Jagtap, M. Jung, W. Elsasser, C. Weis, A. Hansson, and N. Wehn, "Integrating DRAM Power-down Modes in gem5 and Quantifying Their Impact," in *Proceedings of the International Symposium on Memory Systems*, ser. MEMSYS 2017. ACM, Oct. 2017. [Online]. Available: http://dx.doi.org/10.1145/3132402.3132444

[33] A. B. Kahng, B. Li, L.-S. Peh, and K. Samadi, "ORION 2.0: A fast and accurate NoC power and area model for early-stage design space exploration," in *Design, Automation & Test in Europe Conference & Exhibition*, ser. DATE, 2009, pp. 423–428.

[34] C. Sun, C.-H. O. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, L.-S. Peh, and V. Stojanovic, "DSENT - A Tool Connecting Emerging Photonics with Electronics for Opto-Electronic Networks-on-Chip Modeling," in *IEEE/ACM Sixth International Symposium on Networks-on-Chip*, ser. NOCS, 2012, pp. 201–210.

[35] T. E. Hansen, "ARM Power Modelling," https://www.gem5.org/documentation/learning_gem5/part2/arm_power_modelling/, 2024.

[36] R. Tashiro and M. S. Oyamada, "An Environment for Design Space Exploration Using gem5-McPAT," in *VI Brazilian Symposium on Computing Systems Engineering*, ser. SBESC, 2016, pp. 220–225.