Democratizing Computer System Simulation with a Components Library

Bobby R. Bruce, Jason Lowe-Power
University of California, Davis
Computer Science Department

The gem5 simulator is one of the most popular tools available for computer architecture research. Open-source and community supported, the tool is used frequently by researchers to test new designs, and as a teaching tool to help students understand the relationship between hardware design, software, and performance.

The gem5 architecture simulator differentiates itself from other simulators by being a highly configurable, modular platform. From the architectural to the microarchitecture level, researchers may configure parameters and swap components to understand how a particular hardware target may perform. This is done via Python scripts which the simulator uses to set up and execute a simulation. While this has clear benefits, the number of parameters exposed to the user can be overwhelming, and requires users to make a number of design choices; many of which are confusing, expose the user to potentially erroneous choies, and may not be important to the task at hand. For example, creating a basic simulation consisting of a single core CPU connected directly to main memory (no caches or IO) requires 36 lines of Python code[1]. If a private L1/shared L2 cache is added, this number jumps to 111[2]. Neither of these designs are capable of full-system simulation, which in even the most basic of cases, requires design considerations such as the layout of memory and the configuration of IO buses. To avoid this tedium, users presently rely on a patchwork of copy-and-pasted configuration scripts, most of which are endlessly extended and patched as new versions of gem5 are released. It is for this reason we have begun development into an official, standard library of gem5 components.

This standard library will significantly reduce the boilerplate needed to set up a simulation. From a user perspective the library will function in a manner analogous to a computer hardware store. Users will be presented with a plethora of hardware components all of which conform to standard interfaces compatible with a set of common motherboards. Each component will come preset with sensible defaults which aim to replicate, as much as possible, real-world values. In the most vanilla of cases, a user will be able to use an already pre-build system directly from the library and replace components (with those available within the library or those they have developed themselves) to see if their design improves or worsens the system. The library is engineered to both provide a simple high-level view of a design consisting of “off-the-shelf” components, while also allowing the user to change, replace, or expand the capabilities of these components at any level of detail they require.

We hope this will ease the learning curve for gem5 users by allowing them to jump into simulations with a few easy-to-understand statements specifying which components are to be used. In keeping with the open source and community driven ethos of the gem5 project, this component library will be easily expandable by others in the community and will provide a centralized location for gem5 contributors to share their simulation components to reflect both real-world and experimental systems. In addition, the library will enable the project to share more easily configurations with known likeness to real-world equivalents, which will improve the quality of hardware architecture research.

---

[1] https://gem5.googlesource.com/public/gem5/+/refs/heads/stable/configs/learning_gem5/part1/simple.py
[2] https://gem5.googlesource.com/public/gem5/+/refs/heads/stable/configs/learning_gem5/part1