Extended Abstract for Student Poster at the 28th ACM/IEEE Symp. on High-Performance Chips (HOTCHIPS-28), August 2016

Experiences Using a Novel Python-Based Hardware Modeling Framework for Computer Architecture Test Chips

Christopher Torng, Moyang Wang, Bharath Sudheendra, Nagaraj Murali, Suren Jayasuriya,
Shreesha Srinath, Taylor Pritchard, Robin Ying, and Christopher Batten

School of Electrical and Computer Engineering, Cornell University, Ithaca, NY
{clt67,mw828,cbatten}@cornell.edu

This poster will describe a taped-out 2×2 mm 1.3 M-transistor test chip in IBM 130 nm designed using our new Python-based hardware modeling framework. The goal of our tapeout was to demonstrate the ability of this framework to enable Agile hardware design flows.

PyMTL: A Unified Framework for Vertically Integrated Computer Architecture Research. Computer architects have long traded off simulation time and accuracy by leveraging multiple modeling abstractions including functional-level (FL), cycle-level (CL), and register-transfer-level (RTL) modeling. However, these modeling abstractions each use distinct languages, design patterns, and tools, creating a large impediment to vertically integrated, iterative refinement of a design from algorithm, to exploration, to implementation. PyMTL is a unified and highly productive framework for FL, CL, and RTL modeling based on a common high-productivity language (Python2.7) [1]. The PyMTL framework consists of model specifications and tools. Different tools (e.g., Simulation Tool, Verilog Translation Tool) act on the model specification to provide various outputs (e.g., simulation traces, VCD dumps, Verilog for use with commercial EDA toolflows). In addition to enabling incremental refinement from algorithm to hardware implementation, the framework also supports automated testing and integration of PyMTL-generated Verilog, multi-level co-simulation of FL, CL, and RTL models, as well as construction of highly parameterized RTL chip generators.

PyMTL for Computer Architecture Test Chips. Computer architecture test chips are a key aspect in Agile hardware design flows that reduce the cost of validation by rapidly iterating towards "building the right thing" [2]. Test chips build research credibility while also providing highly reliable power and energy estimates for new techniques. Design methodologies differ significantly between large-scale commercial chips and small computer architecture test chips. Design methodologies for commercial chips target high-volume and high-yield production and depend on large teams to overcome design challenges. Design methodologies for test chips target low volumes with reasonable yield and overcome design challenges despite small teams with limited resources. Agile hardware design flows can help small teams address the tremendous challenges associated with building chips. One such environment can be created by combining the PyMTL framework with a highly automated standard-cell design flow. PyMTL provides the unified design and testing environment for not only FL, CL, and RTL models but also for post-synthesis and post-PAR gate-level models. Embracing a highly automated standard-cell design flow allows small teams to meet low-volume and reasonable-yield production targets while shortening the time to tapeout.

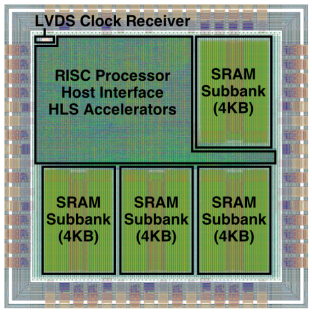

Figure 1. **BRGTC1 Test Chip –** A taped-out 2×2 mm 1.3 M-transistor test chip in IBM 130 nm designed and implemented using our new PyMTL hardware modeling framework. Chip includes a simple pipelined 32-bit RISC processor, custom LVDS clock receiver, 16 KB of on-chip SRAM, and application-specific accelerators generated using commercial C-to-RTL high-level synthesis tools.

Taped Out: March 2016
Expected Return: Fall 2016

Such development environments can potentially enable rapid design iteration from RTL to layout as well as a validated gate-level netlist within a day.

PyMTL in Practice: Detailed Methodology and ASIC Test Chip. Figure 1 shows BRGTC1, a taped-out 2×2 mm 1.3 M-transistor chip in IBM 130 nm designed and implemented using the PyMTL hardware modeling framework. Our design methodology uses PyMTL not only for design but also for composition with a commercial high-level synthesis (HLS) toolflow. The PyMTL RTL design is translated to Verilog RTL to target both a Xilinx-based FPGA backend and a Synopsys-based ASIC backend. The PyMTL testing framework features integration with mature, full-featured software testing tools (i.e., *pytest*). PyMTL-based testing enables using the same test vectors to validate the PyMTL FL/-CL/RTL designs, the Verilog RTL design, the programmed FPGA logic, the post-synthesis gate-level netlist, and the post-PAR gate-level netlist. With this approach, bugs found at each level of abstraction help improve the same test suite used across all levels. For example, FPGA emulation helped catch reset bugs not previously caught in four-state RTL simulation. After fixing our design, our new test vectors for these corner cases helped validate both FPGA and ASIC backend designs. Our ASIC backend was highly automated, enabling our small team to respin tapeout-ready layout with new RTL design features within a day.

Our experience with BRGTC1 demonstrates the potential of the PyMTL framework to enable Agile hardware design flows for future computer architecture test chips.

Acknowledgements. Chip fabrication was made possible by the MOSIS Educational Program. This work was supported in part by NSF CAREER Award #1149464, NSF XPS Award #1337240, NSF CRI Award #1512937, and equipment/tool/IP donations from Intel, Synopsys, Cadence, Mentor Graphics, Xilinx, and ARM. We acknowledge and thank Mark Buckler for EDA toolflow development and Ivan Bukreyev for advice on full-custom design.

[1] D. Lockhart, G. Zibrat, and C. Batten. PyMTL: A Unified Framework for Vertically Integrated Computer Architecture Research. *Int'l Symp. on Microarchitecture*, Dec 2014.

[2] D. Patterson and B. Nikolic. Agile Design for Hardware. *EE Times*, Jul 2015.