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

This Poster...

Describes a taped-out 2x2 mm 1.3M-transistor test chip in IBM 130nm designed and implemented using PyMTL, a novel Python-based hardware modeling framework.

Goal of tapeout was to demonstrate the ability of this framework to enable Agile hardware design flows.

PyMTL: A Unified Python-Based Framework for FL, CL, and RTL Modeling

Functional-Level Modeling (FL)
- Behavior

Cycle-Level Modeling (CL)
- Behavior
- Cycle-Approximate
- Analytical Area, Energy, Timing

Register-Transfer-Level Modeling (RTL)
- Behavior
- Cycle-Accurate Timing
- Gate-Level Area, Energy, Timing

What Does PyMTL Enable?

1. Incremental refinement from algorithm to hardware implementation
2. Automated testing and integration of PyMTL-generated Verilog
3. Multi-level co-simulation of FL, CL, and RTL models
4. Construction of highly parameterized RTL chip generators
**PyMTL for Computer Architecture Test Chips**

**Why Build Computer Architecture Test Chips?**

**Key Aspect of Agile Hardware Design**
- Rapid design iteration
- "Building the right thing"
- Reduces cost of validation

**Benefits Research**
- Builds research credibility
- Highly reliable power and energy estimates for new architecture techniques

**Design Methodologies: Large Chips vs. Small Chips**

**Large-Scale Commercial Chips**
- High-volume and high-yield production
- Overcome design challenges with large teams

**Computer Architecture Test Chips**
- Low-volume and reasonable-yield production
- Overcome design challenges despite small teams and limited resources

Provide small teams with highly productive development frameworks to shorten time to tapeout

---

**PyMTL for Agile Hardware Design**

**Highly Automated Standard-Cell Design Flow**

1. Tape-in
2. ASIC flow
3. FPGA
4. C++
5. FL Simulation
6. CL Simulation
7. RTL Simulation
8. Post-Synthesis Gate-Level Simulation
9. Post-Place-and-Route Gate-Level Simulation
10. Tape-out

---

*Adapted from Yunsup Lee, IEEE Micro 2016*
Experiences Using a Novel Python-Based Hardware Modeling Framework for Computer Architecture Test Chips

IBM 130nm SRAM Compiler
PyMTL RTL of Full Design
Verilog RTL of Full Design
C-Based Accelerator Tested in C
Commercial HLS Tool
PyMTL Verilog Import
PyMTL to Verilog Translator

Verilog RTL Modules Specially Annotated for FPGA Synthesis
FPGA Logic w/ Full Design
Commercial Xilinx-Based FPGA Tools

SRAM Macros Front-End Views
SRAM Specification
IBM 130nm SRAM Compiler
GDS of SRAM Macros

Standard Cell Front-End Views
Post-Synthesis Gate-Level Netlist
Synopsys Design Compiler

Post-PAR Gate-Level Netlist
GDS of Full Design
DRC-Clean GDS of Full Design
Calibre LVS

VCD Traces
Synopsys vcat utility
Verilog Gate-Level Simulator

Mature full-featured software testing tools
Tapeout-Ready GDS of Full Design
Calibre DRC
GDS of Full Design

Full-Custom LVDS Receiver GDS & LEF
Standard / Pad Cell Back-End Views
IBM 130nm PDK

Detailed Methodology
Using PyMTL for Agile Hardware Design

PyMTL-Driven Testing Framework

PyMTL Simulator w/ Unit Tests and Assembly Test Suite

HLS
FPGA
ASIC

Commercial Accelerator
C-Based

PyMTL FL / CL Models

Synopsys IC Compiler

Christopher Torng
Cornell University
Experiences Using a Novel Python-Based Hardware Modeling Framework for Computer Architecture Test Chips

PyMTL in Practice: BRG Test Chip 1

Taped-out Layout for BRGTC1

Testing Plans After Fabrication
The testing platform enables running small test programs on BRGTC1 to compare the performance and energy of pure-software kernels versus the HLS-generated sorting accelerator.

Taped-out Layout for BRGTC1
2x2mm 1.3M transistors in IBM 130nm RISC processor, 16KB SRAM HLS-generated accelerators Static Timing Analysis Freq. @ 246 MHz