**Semiconductor IP Core – Not Just Design**

**Mark Chen**

**First Edition**

**November 29,2022**

**VOLUME 10 Verification IP & IP Core Verification**

**Hardware Verification Language**

**HARDWARE VERIFICATION IN PYTHON**

**A Comprehensive Review**

**CHAPTER 00** **FOREWORD**

**Hardware Verification in Python – A Comprehensive Review**

This section, “**Hardware Verification in Python – A Comprehensive Review**” is part of “Hardware Verification Languages”, which again belongs to the volume of “Verification IP & IP Core Verification” of the book family titled “Semiconductor IP Core – Not Just Design”.

This book investigates, treats and presents some of the key Python technologies, methodologies, tools, packages and libraries involved in the verifications of ICs and in particular semiconductor Intellectual Property (IP) core design process, including the following sections:

Chapter 00 Forward

Hardware Verification in Python – A Comprehensive Review (An Abstract)

Chapter 01 Why Python?

Why Python is chosen as your preferred language for hardware verification? How does Python stand out among hundreds of programming languages? Here are some of our answers.

Chapter 02 Pure Python

Hardware Verification (Testbenches) in Pure Python

Python Verification Example: Automated Testbenches For MAC (Multiply and Accumulate) Using Python and IVerilog (Icarus Verilog)

Chapter 03 cocotb

CO-routine and CO-simulation of Testbench in Python

Chapter 04 cocotb Derivative - pyuvm - Python Implementation of UVM using cocotb

Chapter 05 cocotb Derivative - uvm python

UVM Port to Python

Chapter 06 cocotb Derivative – Varifog

Open Source Python and cocotb based Hardware Verification Tool

Chapter 07 cocotb Derivative - VeRLPy with Reinforcement Learning

VeRLPy Python Library for Verification of Digital Designs with Reinforcement Learning

Chapter 08 cocotb Extension - cocotb-coverage

Functional Coverage and Constrained Randomization Extensions for Cocotb

Chapter 09 cocotb Extension - cocotb-test

A python framework built around pytest

Chapter 10 cocotb Extension - USB Test Suite

Cocotb based USB 1.1 test suite

Chapter 11 MyHDL - Python Based Hardware Description And Verification Language

Chapter 12 Python Embedded Domain-Specific Language For Metaprogramming – fault

A Python Embedded Domain-Specific Language For Metaprogramming Portable Hardware Verification Components

Chapter 13 Hardware Design Language Based On Python – PyRTL

Chapter 14 PyMTL - Python-Based Hardware Generation, Simulation And Verification Framework

Chapter 15 PyHVL - A Verification Tool

PyHVL is a hardware verification language that combines Python with Verilog.

Chapter 16 FPGA Hardware Simulation Framework - FPGA\_HW\_SIM\_FWK

Simulate Hardware Containing an FPGA Programmed in VHDL Interactively

Chapter 17 Hardware Simulation Environment Integrating Python and VHDL - PyHDL

Chapter 18 SPICE-like Electronic Circuit Simulator Written in Python – AHKAB

Chapter 19 Co-Simulation of HDL Using Python and Matlab cosimtcp

Simulation and verification of the implementation of the Digital Signal Processing (DSP) algorithms written in hardware description language (HDL).

Chapter 20 CPU Simulator Written in Python – PyCpuSimulator

Chapter 21 Python Library For Interfacing To Various Simulators – PyOPUS

PyOPUS is a library for simulation-based optimization of arbitrary systems.

Chapter 22 Python Interface to the Ngspice and Xyce Circuit Simulators – PySpice

PySpice is an open source Python module which provides a Python interface to the Ngspice and Xyce circuit simulators.

Chapter 23 Constraints and Coverage - PyVSC Package

PyVSC is an embedded domain-specific language (eDSL) , effectively extending the Python language with new semantics for constraint solving and coverage capture.

Chapter 24 Constraints and Coverage - YosysHQ-MCY

Mutation Cover with Yosys

Chapter 26 Symbolic Model Checking – CoSA

CoSA (CoreIR Symbolic Analyzer) is a symbolic model checker for hardware design.

Chapter 26 Front-End Driver Program - YosysHQ-SymbiYosys

SymbiYosis a front-end driver program for Yosys-based formal hardware verification flows.

Chapter 27 Emulation - AMIQ OFC

Open-Source Framework for Co-Emulation using PYNQ

Chapter 28 Emulation – ICemu: Emulate Integrated Circuits In Python

ICemu is a C library (originally in Python) that emulates integrated circuits at the logic level.

Python as the most popular programming language is the natural choice of hardware verification language (HVL), particularly for the newcomers of the IC design sector, who are not so much involved in more demanding HVLs such as SystemVerilog, Verilog, SystemC, e, VHDL etc. Python is easy to learn, yet Python is still powerful enough to meet some of the key verification requirements. Versatility and ability to cope with almost any design philosophy is the distinguishing mark of Python, therefore it is not surprising that it has finally approached to hardware design. Hardware modules are modeled as Python functions with wrapping decorators that return sets of generators. Thus, the semantics and structure of Python functions are used to support features such as arbitrary hierarchies of components, named port association, etc...

Yet production verification engineers are not aiming to replace UVM with Python. There will always be many clever things that UVM can do that Python cannot (easily). The purpose of Python development and usage around verification is to plug the holes in mainstream verification methodologies. For stupid tests and to support designers running their own verification. To speed up standard verification flows and to support silicon bringup test development. etc etc.

This 500+-page long introductory material will introduce readers to the major Python methodologies, systems, packages and libraries dealing with the hardware verification needs, incl also prototyping, emulation and simulation. After reading, readers will become familiar with historic notes, key features, installation & configuration, usage, verification samples and results as well as their applications in the IC & IP Core verification sector. I will leave benchmarking to my readers, as the technology is still maturing and it’s still a bit too early to say who is the best among dozens of Python-based verification systems.

For details about the chapter contact the author for a complete text.