Towards Efficient Design Verification - PyUVM & PyVSC

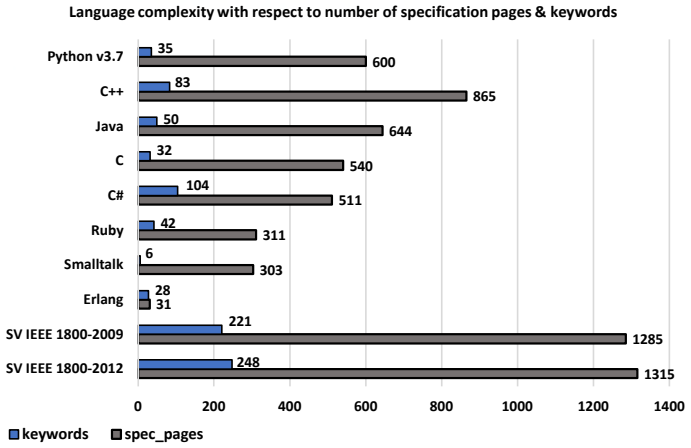
Deepak Narayan Gadde, Suruchi Kumari, Aman Kumar

Infineon Technologies Dresden GmbH & Co. KG, Dresden, Germany

accellera SYSTEMS INITIATIVE

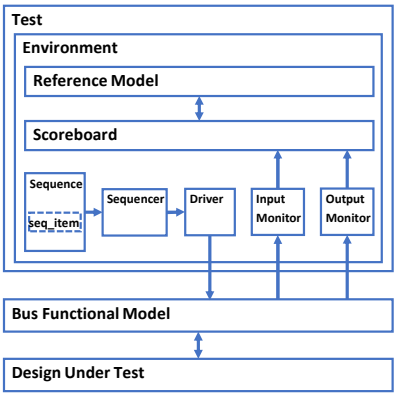
2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES
SAN JOSE, CA, USA
MARCH 4-7, 2024

Why to use Python in Verification?



Objectives

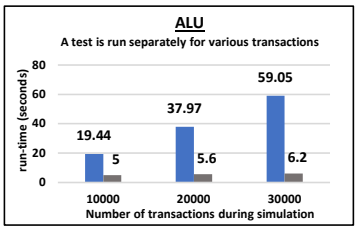
- **Python-Cocotb** → RTL simulator **plugin** and library for writing **synchronous logic**
- **PyUVM** → Python implementation of **UVM** methodology
- **PyVSC** → Enables constrained random verification methodology and coverage constructs
- **Effectiveness of PyUVM in verification with respect to SystemVerilog-UVM**
  - Simulation run-time
  - Features
  - Coverage analysis
- **Case Study on 3 design IPs**
  - ALU, ADC, and ECC

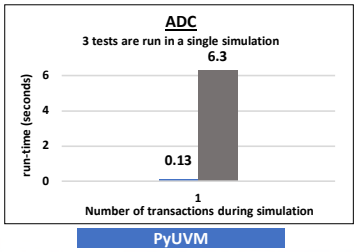


Testbench Architecture using Bus Functional Model

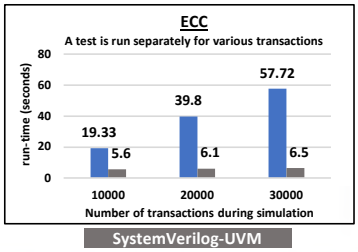
Comparison of Simulation run-time

- SystemVerilog-UVM testbenches perform better than PyUVM in ALU and ECC
- With the help of automatic tests discovery in Cocotb, PyUVM excels SystemVerilog-UVM in case of ADC testbench





PyUVM



SystemVerilog-UVM

Comparison of Features

Feature	SystemVerilog-UVM	PyUVM	Remarks
Utility Macros	Yes	No	PyUVM: All classes are already registered in factory
Field Macros	Yes	No	PyUVM: *__str__(), __eq__()*
RAL Model	Yes	In development	-
Logging	Yes	Yes	PyUVM uses logging module
Configuration Database	Yes	Yes	PyUVM: *ConfigDB().set(None, "*", "BFM", bfm)*
Importation	Yes	Yes	PyUVM: *from pyuvm import **
User-defined phases	Possible	Not possible	In PyUVM, only common phases from UVM specs are implemented
uvm_test	Required	Not required	-
Awaiting tasks	No	Yes	PyUVM: *await run test()*
TLM System	Required	Not required	In PyUVM, any uvm_component can instantiate *uvm_put_port* or *uvm_get_port* in its build phase
Covergroups	Yes	No	Since Python does not have coverage constructs, PyVSC library is used along with PyUVM

Coverage Analysis

Design IP	ALU		ADC		ECC	
SV-UVM/ PyUVM	SV-UVM	PyUVM	SV-UVM	PyUVM	SV-UVM	PyUVM
Number of distinct tests	1	1	3	3	1	1
Number of Transactions	30000	30000	-	-	30000	30000
Coverage (%)	78.29	100	100	100	95	95

- PyUVM and SystemVerilog-UVM achieved comparable coverage in case of ADC, ECC case studies
- In ALU case study, PyUVM could achieve coverage closure due to the random seeds selected for that simulation

Conclusion

- Developed verification testbenches in SystemVerilog-UVM and PyUVM for three design IPs → ALU, ADC, and ECC
- PyUVM is used in conjunction with PyVSC to make comparable testbenches as SystemVerilog-UVM
- Simulation run-time with PyUVM testbench may improve if clock generation is moved from testbench to DUT side
- PyUVM simulation easy collection of input and coverage data in a preferred format
- PyUVM can introduce new methodologies based on Machine Learning to overall improve design verification

References

- R. Salemi and T. Fitzpatrick, "Verification Learns a New Language: – An IEEE 1800.2 Implementation" 2021
- M. Ballance, PyVSC : SystemVerilog - Style Constraints, and Coverage in Python, 2019

infineon