A Review of Open-source Systems on Chip, Case of LiteX, RubyRTL, and PyMTL

Hasna Elmaaradi[1][a], Abdelhakim Alali[2], Mohammed Khaldoun[1] and Mohamed Sadik[1]
[1]*Research and Engineering Laboratory, National High School for Electricity and Mechanics, Hassan II University of Casablanca, Casablanca, Morocco*
[2]*Laboratory of Information Processing, Faculty of Sciences Ben M'Sick, Hassan II University of Casablanca, Casablanca, Morocco*

Keywords: SoC, DSL, Open-source, Python, HDL, FPGA.

Abstract: FPGA-based SoCs have become popular daily compared with traditional solutions and expanding application areas to new areas. Intellectuals Properties sharing and reuse are a practical process for electronics designers to reduce the design complexity of SoC. However, open-source hardware has become a compelling concept for improving design productivity. In this article, we present platforms, aiming to make hardware design easier and flexible. They ensure a very high level of abstraction. We expose the aspects of similarity and difference between LiteX, RubyRTL, and PyMTL to highlight the progress that LiteX has made as an open-source framework that has accelerated RD (research and development) in EDA (electronic design automation). LiteX has helped emphasize the concept of codesign and provide engineers and researchers with highly developed and reliable tools.

1 INTRODUCTION

Flow design of semiconductor devices has experienced exponential development in recent years. Therefore, we have many and different HDL (Hardwar Description Language). However, VHDL and Verilog, created in 1980, still being present in their structure; eventually, EDA relies on those two languages. They can describe all the electronic devices with a high abstraction at different levels. It facilitates relatively the work of designing engineers and makes process design more flexible. There are new challenges becomes with micro and nanoelectronics, so new HDL spring. The first one is System Verilog, standardized as IEEE 1800. It provides tools to describe, verify, simulate, test, and implement electronic systems. The second language is SystemC. It contributes to improving abstraction and modelling levels. (Black and Keist, Python 2009).

The DSLs (Domain Specific Languages) renewal EDA and give birth to "high-level language" (recognized programming language) that embedded a DSL. That is why these languages are called Embedded DSLs. In this context, we have Chisel/SpinalHDL (high-level language is Scala), HardCamel (high-level language is Scala), Haskell, and Lava. What is familiar to these DSLs is the ease of learning, access to libraries of language hosts, and the aim of developing complex devices and improving time to market.

In addition, MyHDL (J.Decaluwe, 2004), based on Python, contributes to improving the RTL (Register Transfer Language) circuits design. Python is a very high-level language; consequently, scientific research heads towards developing other Frameworks based in Python, such as LiteX platform (Migen), PyMTL, and RubyRTL.

In this paper, we present a comparative study between three open-source LiteX, PyMTL, and RubyRTL. It is organized as follows. The next section makes a short review of Migen Python DSL. The third section presents RubyRTL. Section 4 will present PyMTL. In section 5, we expose the principal difference between these DSLs, then a discussion is giving in section 6. We complete the article with the conclusion in section 7.

[a] https://orcid.org/0000-0002-6557-0990

Elmaaradi, H., Alali, A., Khaldoun, M. and Sadik, M.
A Review of Open-source Systems on Chip, Case of LiteX, RubyRTL, and PyMTL.
DOI: 10.5220/0010729500003101
In *Proceedings of the 2nd International Conference on Big Data, Modelling and Machine Learning (BML 2021)*, pages 119-124
ISBN: 978-989-758-559-3

2 MIGEN STRENGTHENS RTL DESIGN: LITEX PLATFORM

Migen FHDL (Florent Kermarrec and Badier, 2020) (Fragmented Hardware Description Language) is founded on Python but includes Abstract Syntax Tree (AST). It provides a Verilog language for circuit design. Python language presents different advantages such as being object-oriented programming, support essential elements like functions and generators, operators, and libraries, to build well Hardwar designs embedded. In cooperation with "ENJOY DIGITAL" (Digital, 2021), they established the LiteX platform:

-Provides ten open-source IPs (intellectual property), for instance: LiteDRAM, LiteEth, LiteUSB, etc.

- supports four softcore (LM32, PicoRV32, VEXRiscV, Mor1Kx).

-Support 21 Boards (statistics of 2018).

-Extends Migen based on FHDL.

-Allows building different structures: Xilinx, Altera, Lattice, Microsemi, and yosys.

-Building SoC: NeTV2

3 RUBYRTL NEW DSL RTL

RubyRTL (LE LANN and Florent, 2020) bases on an object oriented and multi-free paradigm named Ruby. The new DSL RTL ensures the efficiency of Hardwar design. Even if RubyRTL is also relying on MyHDL and Migen Python. It generates a VHDL code and can give an AST viewing (Graphviz dot file), the tool that drawing graphs.

The Sexpir compiler interprets an IP address described in Migen to Ruby RTL. Then, it generates VHDL code to ensure communication between them. These three tools (Migen, Sexpir, and RubyRTL) results in a platform that can be strong in handling complex electronics systems.

However, Sexpir is not yet checked and verified on the hardware level. It is under development, and we are waiting for the full version in future works.

4 PYMTL: UNIFIED FRAMEWORK FOR HARDWARE DESIGNING

PyMTL (Shunning Jiang and Batten, 2018) is a Hardware generation, simulation, and verification framework. Based on Python and completed by HGFs (Hardware generation frameworks), it is a DSL significantly developing and describes different levels of abstraction design.

The PyMTL workflow starts from the functional level (FL) programmed in Python, using different functions: Py.test, coverage.py, and hypothesis. After getting the PyMTL RTL model with the help of cycle level (CL), you can generate Verilog format. A test bench (TB) gives the possibility to co-simulate/ simulate the achievements.

This framework uses the new methodology named modelling towards layout (MTL) (Derek Lockhart and Batten, 2014) for multi-level modelling (FL, CL, RTL, PL: physical level), different types of testing (Test Harness: Unit tester, Integration tester, PBRT tester Property-based random test), and Evaluating (simulation, generation, characterization) On-Chip Networks.

Currently, we have a recent version: PyMTL3 (Cheng Tan and Batten, 2019), which comes with updates to bring that framework as an open-source Hard- war ecosystem.

5 LITEX, RUBYRTL, PYMTL: REFLECTION RESULTING

5.1 High-Level Language

In general (although RubyRTL also includes the Ruby language), the three frameworks are based on the Python language. So, a question arises, why this one exactly?

In previous years, in the field of electronic system design, two leading players have been involved in circuit designing: hardware and software engineers. Each takes care of a part that the other cannot do because it is not their speciality.

The field of computer science has developed considerably; indeed, several programming languages were paired. Researchers have thought of exploiting these languages to facilitate the work of hardware engineers and immigrated to what is called Codesign. They will not need the intervention of

software engineers; a hardware designer alone can perform the same tasks.

The choice was Python; first of all, it is the easiest to learn, and it is the most crucial criterion for designers. The understanding of the instructions is easy since the syntax is quite clear.

Secondly, it is a high-level language used in almost all scientific fields due to its flexibility and in-interpretation.

Thirdly, it is less complicated in terms of programming. In only a few lines, you can create functions. Besides, Python has many modules and scripts ready for use. Last but not least, it is an open-source language and runs on various existing operating systems (OS).

5.2 Generated Language HDL

Figure 1: languages generating from different HDLs studies.

As shown in Figure 1 (Florent Kermarrec and Badier, 2020) (LE LANN and Florent, 2020) (JIANG Shunning and al, 2020), each platform generates either HDL: VHDL, Verilog, or SystemVerilog. These languages are similar in terms of functionality but have very different syntaxes. Some frameworks convert in both directions.

The two HDLs Verilog/VHDL were the first languages to allow better abstraction at the RTL level. System-Verilog is a fusion of Verilog and HVL (Hardware verification languages) to verify the circuits designed on HDL.

5.3 Toolbox

As shown in Figure 2, RubyRTL uses the functionality of the RUBY language but also takes advantage of the libraries indirectly offered by Migen. The Sexpir compiler handles the exchange between Migen and RubyRTL to generate the VHDL code.

LiteX is also based on Migen. It also uses MiSoC for exploitation purposes of this platform (SoC design). PyMTL develops its toolbox based on the Python language. In addition to (Shunning Jiang and Batten, 2018) (Derek Lockhart and Batten, 2014).

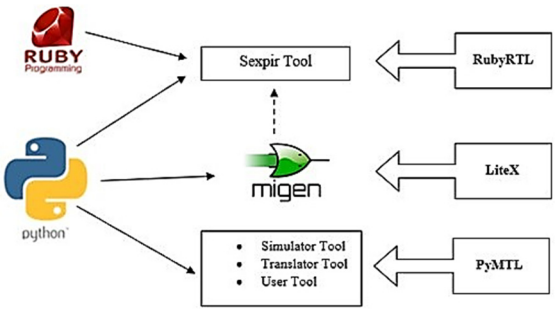

Figure 2: Description of Tools uses in different Framework studies (Florent Kermarrec and Badier, 2020) (LE LANN and Florent, 2020) (Derek Lockhart and Batten, 2014)

Simulator Tool, Translator Tool, and User Tool, the packages Py.Test (testing Framework), coverage.py (code coverage), and hypothesis (PBRT property-Based Random Testing) are part of the processing chain to build the Python simulator and PyMTL RTL.

5.4 Simulation Performance

In this part, the comparison is between Migen, Verilator, and Python simulators. Indeed, RubyRTL uses Migen Simulator, LiteX uses LiteXSim based on Verilator, and PyMTL uses Python simulator.

5.4.1 LiteXSim SoC simulator

LiteXSim Soc inspires from Verilator (Florent Kermarrec and Badier, 2020). It (Veripool, 2021) compiles code synthesized in Verilog / System Verilog into C++ or SystemC. It makes it the fastest tool compared to other interpreted simulators.

LiteX takes advantage of these multiple functionalities to build (Piotr Binkowski, 2021) a 'Litex.build.sim' library, of which two main classes 'SimPlatform' and 'SimConfig'. The first one creates the platform for simulating. The second one defines its modules via different functions such as 'add. Module' and 'add. Clocker'. The builder class emulates the defined system on chip SoC.

5.4.2 Migen Simulator

Migen. Sim (Sebastien Bourdeauducq, 2021) the library that provides you with several simulation and testbench (TB) functions, such as 'run simulation'. It starts the simulation of either FHDL or TB. The simultaneous stimulation of FHDL and TB ensures via four processes: Read, Write, Clocking, and Composition. They offer flexible handling of the simulator according to the designer's needs.

5.4.3 PyMTL Simulator

The simulation leverages the functionalities of Python via the PyPy system and Mamba HGSF (S.Jiang and al., 2018) (Hardware Generation and Simulation Frameworks). It (Shunning Jiang and Batten, 2018) is carried out according to modelling level (FL, CL, TB) and generated Verilog.

The latest PyMTL3 (JIANG Shunning and al, 2020) version, launched in 2020, is similar to the LLVM (formerly called Low-Level Virtual Machine) in its overall structure. The PyMTL3 DSL (JIANG Shunning and al, 2020) provides the PyMTL3 IMIR (In Memory Intermediate Representation), the hardware model to be processed by the analysis, instrumentation, and transformation pass. These passes aim to improve the hardware model. they introduce modifications to the PyMTL3 IMIR on several levels. In a simulation (JIANG Shunning and al, 2020), for example, EventDrivenPass, list updates for the blocks of the pure-RTL model created by it. Besides, StaticSchedulingPass applies an execution methodology for each cycle.

5.5 Experiments and Synthesis

The figure below illustrates the various experiments and completed projects developed with the platforms highlighted in this review.

According to this listing, RubyRTL is recently launched (in 2020). However, it developed a toolchain to verify the reliability of IP interchange by using an IP URAT. The results are encouraging for further research in the future. While LiteX and PyMTL have made significant progress. PyMTL contributed to the prototyping of devices (TORNG Christopher and al., 2016) Computer Architecture Test Chips, i.e., Celerity, BRGTC1, and BRTC2. They are chips based on transistors in the CRAFT program (Circuit Realization at Faster Timescales).

In addition, PyMTL has not handled SoCs. So far, it is intended for the FPGA and ASIC. However, LiteX is generalized (FPGA, and SoC). It is one of the reasons why this platform has become an essential part of the global environment for the hardware design of complex systems.

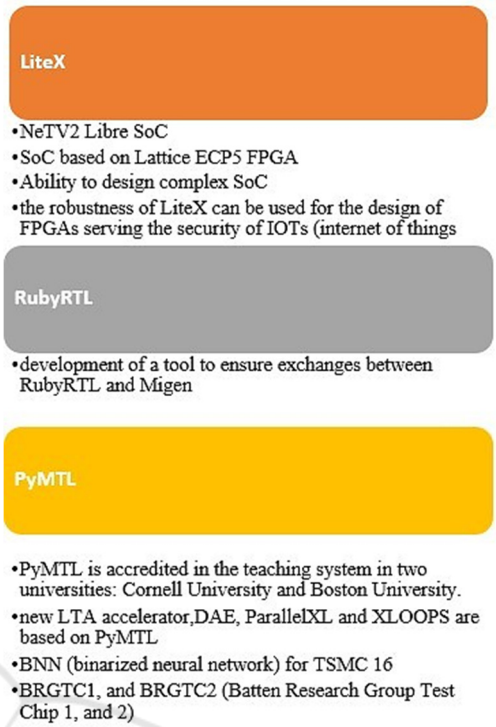

Figure 3: the achievements of each platform.

Several and different projects have been realized with the LiteX platform, such as the one marketed by the company 'Enjoy-digital', and also offers cooperation with other open sources (Florent Kermarrec and Badier, 2020):

-NeTV2 tests/validation: video processing development platform.

-SoC based on Lattice ECP5 FPGA.

-HDMI2USB.

-Fupy: FPGA MicroPython.

-Axiom SDI model.

-PCIe Screamer: FPGA PCI.

6 DISCUSSION

These three open sources aim to provide designers with a very high-performance DSL, even if they differ from the methodology adopted to achieve it.

At the level of the design language, they agreed on the reliability of Python. However, RubyRTL uses the Ruby language, knowing for its flexibility and ease in web development. The first feature is shared

with Python. Nevertheless, the second has not brought much added value (in terms of performance) to the hardware co-design.

Verilog, VHDL, and System Verilog being synthesis languages, allow the implementing of the source code in an electronic system. VHDL guaranteed the behavioural description of the circuits and the verification of the simulated model safely. If we are looking for speed in design, we need to call Verilog. System Verilog tries to combine both performances and provide the best synthesis tool. These HDLs develop and adapt to high technology, making it very complicated to compare and choose between them. Hence the existence of the conversion from one to the other.

Since Sexpir is not yet implemented, we limit our study to Migen and PyMTL tools. The platform's toolbox is based on Python, but each uses it differently from the other. In Migen, we have libraries that provide the software description resources.

In addition, they organize the design flow and provide the infrastructure necessary for the synthesis of SoCs. PyMTL consists of dividing the tools into modules. It ensures independence between the stages of the co-design and the mastery of each abstraction level.

The performances of the simulators used in the present ecosystems are demonstrated and tested. Effectively, LiteXSim, MigenSim, and PyMTL simulators are characterized by the speed in the compilation. They are easy to handle by the designer via the functionalities of Python (the simple functions to launch the simulation).

No one cannot deny that the synthesis and implementation of a model in an electronic circuit is an indisputable criterion that assesses the cohesion and pragmatics of the framework. As a result, we favour LiteX, which has marked a significant progression on this path. Also, the prospects for future work promise improvements and updates that will make this platform more global (including the different structures) and suitable for the most complex systems.

7 CONCLUSION

In this review, we have compared three Python-based open sources. Several aspects have been discussed: the high-level language, toolbox, simulation performance, experiments, and synthesis. RubyRTL, PyMTL, and other platforms which are not covered in this paper (PyRTL (John Clow and Sherwood, 2017), SysPy (Evangelos Logaras and Manolakos, 2014), Open ESP (Paolo Mantovani and P.Carloni, 2020)) compete with LiteX, which is ahead of them, given its use and handling in the co-design of SoCs.

We highlighted the generality, maturity, and ability of the LiteX platform. It designs complex components for applications that are the future of intelligent systems. LiteX makes us confident in future work for its ability to design synthesizable SoCs based on custom FPGAs that can hold very complex algorithms (Embedded computing, cloud solutions, Computational Intelligence….).

REFERENCES

Black, D.C, D. J. B. B. and Keist, A. (2009). SystemC: From the ground up. In (Vol.71). Springer Science Business Media.

Cheng Tan, Yanghui Ou, S. J. P. P. C. T. S. A. and Batten, C. (2019). Pyocn: A unified framework for modelling, testing, and evaluating on-chip networks. 37th International Conference on Computer Design (ICCD).

Derek Lockhart, G. Z. and Batten, C. (2014). PyMTL: A unified framework for vertically integrated computer architecture research. 47th Annual IEEE/ACM International Symposium on Microarchitecture.

Digital, E. (18/02/2021). Various projects powered by LiteX. http://www.enjoy-digital.fr.

Evangelos Logaras, O. G. H. and Manolakos, E. S. (February 2014). Python to accelerate embedded soc design: A case study for systems biology. ACM Transactions on Embedded Computing Systems, Vol. 13, No. 4, Article 84.

Florent Kermarrec, Sebastian Bourdeauducq, J.-C. L. L. and Badier, H. (05/05/2020). Litex: an open-source soc builder and library based on Migen Python DSL. arXiv: 2005.02506v1 [cs.AR].

J.Decaluwe (2004). Myhdl: A Python-based hardware description language. In P.5. Linux Journal.

JIANG Shunning, PAN Peitian, O. Y. and al (2020). Pymtl3: A Python framework for open-source hardware modelling, generation, simulation, and verification. IEEE Micro, 2020, vol. 40, no 4, p. 58-66.

John Clow, Georgios Tzimpragos, D. D. S. G. J. M. and Sherwood, T. (2017). A pythonic approach for rapid hardware prototyping and instrumentation. 27th International Conference on Field Programmable Logic and Applications (FPL).

K.O.Setetemela, K.Keta, M., and S.Winberg (2019). Python-based FPGA implementation of AES using migen for the internet of things security. 2019 IEEE 10th International Conference on Mechanical and Intelligent Manufacturing Technologies (ICMIMT 2019).

Le Lann, Jean-Christophe, B. H. K. and Florent (2020). Towards a hardware DSL ecosystem: Rubyrtl and friends. OSDA 2020 Open Source Hardware Design, colocated with DATE 20.

Paolo Mantovani, Davide Giri, G. D. G. L. P. J. Z. E. G. C. M. P. C. P. and P.Carloni, L. (2/09/2020). Agile soc development with open esp. arXiv:2009.01178v1 [cs.AR].

Piotr binkowski, Jedrzej Boczar, E.-d. (20/02/2021). Litexsim.https://github.com/enjoy-digital/litex/blob/master/litex/tools/litexsim.py.

Shunning Jiang, C. T., and Batten, C. (08/11/2018). Open-source Python-based hardware, generation, simulation, and verification framework. WOSET 18.

S.Jiang and al. (Juin 2018). Mamba: Closing the performance gap in productive hardware development frameworks. Design Automation Conference (DAC).

Sebastien Bourdeauducq, W. (16/02/2021). Migen-Sim. https://github.com/mlabs/migen/blob/master/doc/simulation.rst.

TORNG Christopher, WANG Moyang, S. B. and al. (2016). Experiences using a novel Python-based hardware modelling framework for computer architecture test chips. Hot Chips Symposium.p.1.

Veripool (20/02/2021). the fastest Verilog/SystemVerilog simulator. https://github.com/verilator/verilator.