Open Source Tools for FPGAs

Rafael Mascarenhas Dal Moro
Electronic Engineering Guest Student
Hochschule Hamm-Lippstadt
Lippstadt, Germany
rafael.mascarenhas-dal-moro@stud.hshl.de

***Abstract*—Field-programmable gate arrays (FPGA) are versatile, widespread devices that contribute to the development of numerous state-of-art technologies. The design of these instruments requires a different set of tools, which are almost completely proprietary. This paper addresses the impact of open-source solutions in economic development, emphasizes the importance of open-source FPGA toolchains for the sector, and presents the most relevant open-source solutions, and their results, developed and published in the field.**

***Index Terms*—Open Source, FPGA, toolchain, OSS, tool**

I. Introduction

A. FPGA Overview

Field-programmable gate arrays (FPGA) are devices widely used in the creation of modern control systems. They significantly support the development process by enabling the reconfiguration of hardware/software throughout the device‘s lifecycle, especially during its development and installation. Beyond the topic previously mentioned, FPGAs also have many applications in different sectors of society, especially in military and aerospace equipment. [1]

Furthermore, control systems using FPGA have many advantages: higher reliability, size and cost reduction, and the possibility of features such as remote hardware updates. [1]

In order to develop ASIC and FPGA-oriented projects, Electronic Design Automation Software(EDA tooling) is crucial. This program consists of three major areas:

- Hardware description
- Frontend (synthesis tools)
- Backend (verification, testing, and simulation tools)

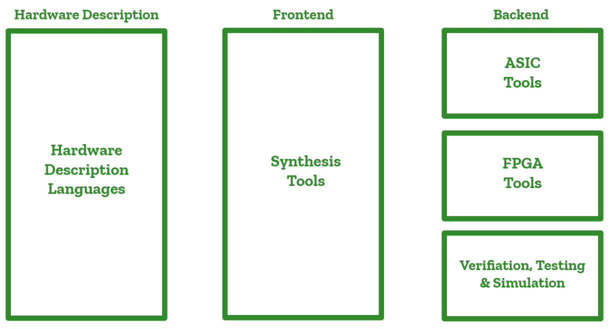

Fig. 1. Electronic Design Automation Tooling Ecosystem

While some open hardware description languages are available (i.e. Verilog, VHDL, Chisel, Migen, and Amaranth HDL), frontend and backend tooling lack established standard, vendor-neutral solutions. [2]

Another important factor for this study is the current market share of FPGA devices. In 2022, two leading vendors, Xilinx and Intel-Altera, represent more than 85% the FPGA market. Both companies provide unique FPGA circuits and proprietary development tools. [3]

B. Open Source

“Open Source” defines innovations that are jointly developed by different contributors. These solutions can be included in commercial products without the obligation of paying royalty fees. The initiative, which is mainly related to software development, has been recently discussed in the context of crowdsourcing, a specific approach within Open Innovation. [4]

More than a utopic concept, Open Source Software (OSS) has shown positive impacts on society. In 2018, companies located in the European Union invested approximately €1 billion in OSS, which resulted in an impact of €65 to €95 billion on the continent’s economy. ”The analysis estimates a cost-benefit ratio of above 1:4 and predicts that an increase of 10% of OSS contributions would annually generate an additional 0.4% to 0.6% GDP as well as more than 600 additional ICT start-ups in the EU.” [4]

Also, when the public sector procures OSS instead of proprietary software, it decreases the total cost of ownership, avoids vendor lock-in, and thus advances its digital autonomy. [4]. Finally, it is notable that OSS can have competition effects on the market and impact the price and quality of proprietary software. [4]

C. Topics addressed in this paper

This study presents an overview of the Open Source context within FPGA development. The paper seeks to answer the following questions:

- Is there any open-source FPGA toolchain?
- If so, what are they doing?
- What are the similarities and differences between open-source tools and proprietary counterparts?

II. Challenges in FPGA tool development

Traditionally, FPGA projects are created using proprietary software tools to simulate and build the design. For example, when developing an FPGA design targeted to Xilinx devices, Vivado software is utilized. Consequently, when target devices are from a different FPGA vendor, the developer has to learn how to use a different tool, slowing production. [5] Also, without a prototype CAD system for each FPGA architecture of interest, an FPGA architect cannot quantitatively evaluate new architectural ideas. [2]

Many issues involving FPGA could be solved by using open-source software. However, OSS for FPGA are particularly difficult to develop, facing many challenges along the production. First, the majority of FPGA vendors keep the contents of their bitstreams in secrecy [5]. This sealed market has resulted in a fragmented ecosystem with low interoperation. In the last years, a few set of open-source tools for FPGA development have appeared, some of them based on reverse engineering of the devices from the most extended providers. [3]

Furthermore, the tasks of FPGA synthesis and place and route are so complex that not even all microchip manufacturers can afford the development of their own solutions. For example, Russian manufacturer VZPP-S JSC suggests using the Quartus package by the Intel-Altera company for its FPGA bitstream, and Chinese company Gowin Semiconductor uses Synplify Pro by Synopsys for synthesis. [1]

III. Is there any open-source FPGA toolchain?

During this study‘s bibliographic research, numerous solutions with different degrees of complexity were encountered in literacy. the main softwares that contribute to the development of an open-source FPGA toolchain are listed below.

OpenFPGA is an open-source framework that promotes agile prototyping for modern and customizable FPGA architectures. The OpenFPGA framework consists of two design flows:

- Production flow
  It translates an XML-based FPGA architecture description to gate-level Verilog netlists of the entire FPGA fabric. This can be provided in a physical design backend to obtain a GDSII layout. Also, Production flow can autogenerate test benches to perform pre-and post-layout verification. Synopsys design constraint (SDC) files are also autogenerated to enable timing-driven backend flows and to conduct post-layout timing analysis. Once the signoff is completed, the GDSII layout is ready for tape-out. [6]
- End-user flow
  It allows FPGA developers to transcript applications written in Verilog to configure bitstream and implement them on the FPGA fabric. [6]

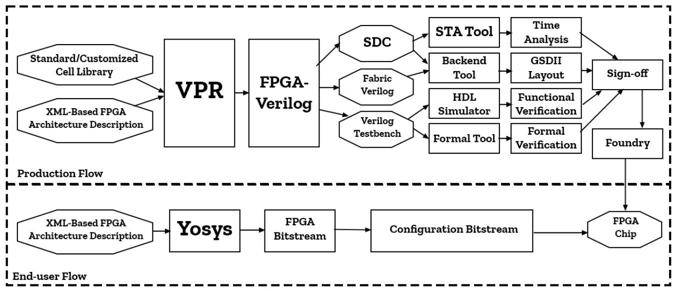

Fig. 2. OpenFPGA design flows

With this solution, standard cell FPGAs can be ported to advanced technology nodes with few manual efforts and benefit from significant performance improvements. In custom FPGAs, OpenFPGA can produce an initial layout with a 60%/20% area/performance gap when compared to a commercial state-of-the-art FPGA.

Princeton Reconfigurable Gate Array (PRGA) is a highly customizable, scalable, and complete open-source framework for building custom FPGAs. The framework can generate synthesizable Verilog from user-specified FPGA architectures, and provide an entire, auto-generated, open-source CAD toolchain for the custom FPGAs. PRGA is developed in Python, offering a user-friendly API, and supporting the use of both standalone FPGA as well as an embedded FPGA [7]

Archipelago is an open-source FPGA with tool flow support. The project resulted in a 64 Bit counter that can run up to 364MHz (on average) on three different FPGA instances. It is a parameterizable and user-expandable FPGA with tool flow support. [8]

SymbiFlow is an end-to-end FPGA synthesis toolchain to provide a fully open-source, multi-platform, and vendor-neutral design tool option for FPGA developers. It contains all of the necessary tools to convert a Verilog design to a final bitstream. [9]

In addition to these larger, coordinated projects previously mentioned, it is essential to recognize the existence of tools that are crucial to the development of these tools. Some of the solutions are:

- **Yosys** is an open-source Verilog synthesis tool that supports almost all features of the Verilog 2005 standard. [5]
- **Nextpnr tool** is an open-source place and route tool developed as a replacement for the Arachne place and route tool. [5]
- **Versatile Place and Route (VPR) tool** is an open-source place and route tool. It performs placement, routing, and analysis operations on our netlist. Also, it supports netlists generated from synthesis tools such as Yosys. [5]
- **Icarus Verilog software tool** is an open-source Verilog compiler that includes a synthesizer and simulator. [5]
- **Verilator tool** is an open-source Verilog compiler that only accepts synthesizable Verilog or SystemVerilog code. [5]

- **GHDL software** is an open-source VHDL compiler and simulator. It offers full support for the VHDL-87, 93, and 2002 standards. In addition, there is also partial support for the VHDL-2008 standard. [5]

IV. F4PGA PROJECT STRUCTURE

In 2015, a group of developers reversed-engineered a reconstructed the configuration file format for FPGA Latt ICE40. The project, entitled IceStorm, made it possible create a toolkit for implementing the whole cycle of operati needed to configure serial-produced FPGA based on H description with Verilog language using only open-sou software. [1] This was the principle of F4PGA developme

Also, this open toolchain can operate on systems fitted w only 512 MB of RAM and requires less than 1 GB of hard drive free space. This enables using the toolchain directly on the embedded computers, which opens up new opportunities for reconfiguring FPGA directly in the process of control system operation. [1]

F4PGA is an open-source toolchain for the development of different vendors' FPGAs. The software targets the Xilinx 7-Series, Lattice iCE40, Lattice ECP5 FPGAs, QuickLogic EOS S3 and is continuously expanding to provide a comprehensive end-to-end FPGA synthesis flow. [2]

In order to become a complete OSS FPGA toolchain, the solution requires a number of tools to provide an end-to-end flow. Thus, F4PGA gathers previously mentioned backend systems, such as nextpnr and Verilog to Routing, and frontend software like Yosys. Each FPGA device is targeted in a different project. For example, Project X-Ray focuses on Xilinx 7-Series, Project IceStorm on Lattice iCE40, and Project Trellis on Lattice ECP5 FPGAs. [2]

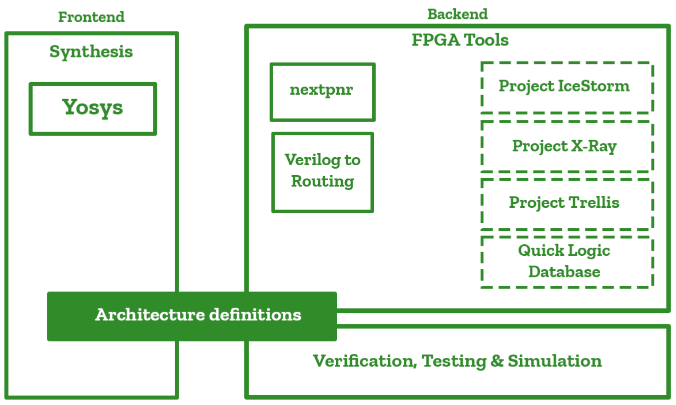

Fig. 3. F4PGA project structure

The solution's tool flows heed the following steps: Verilog sources and Synthesis Scripts are synthesized on Yosis. The synthesized design is output to nextpnr to map the design to the logic gates within the FPGA using rules and parameters for the target FPGA chip as specified. These rules and parameters have been developed from their reverse engineering of bitstreams for various FPGA chips. The figure below shows an example of this procedure. [9]

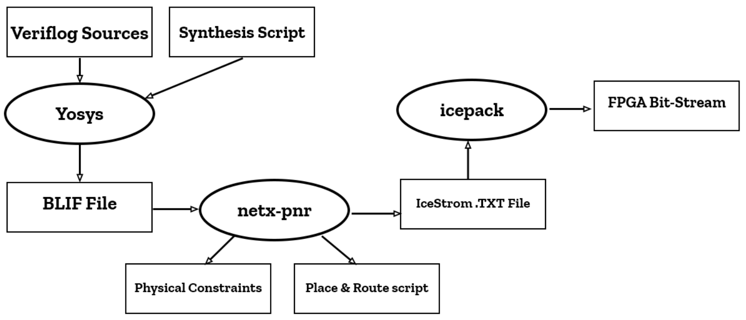

Fig. 4. F4PGA project structure

Also, until the publication of this document, the official project website stated the current status for each project:

TABLE I
CURRENT F4PGA PROJECTS STATUS

	Icestorm	Trellis	X-Ray	QuickLogic DB
Logic	yes	yes	yes	yes
Block RAM	yes	yes	partly	yes
DSP	yes	yes	no	yes
Hard Blocks	yes	yes	no	yes
Clock Tiles	yes	yes	yes	yes
IO Tiles	yes	yes	yes	yes
Logic	yes	yes	yes	yes
Clock	yes	yes	yes	yes

V. RESULTS BENCHMARKS AND ANALYSIS

This section presents relevant results in open-source FPGA tool flow building and its application in various scenarios. Each subsection represents a different science article.

A. FPGA-based control system reconfiguration

When studying the possibility of using open-source FPGA tools for reconfiguring FPGA-based control systems, the results have shown that these solutions are comparable to the ones built by well-known commercial toolchains. [1] The study demonstrates satisfactory efficiency and high productivity of the open-source tools for FPGA synthesis and bitstream generation. The optimization of the resulting solutions in terms of occupied area is currently worse than the commercial analogs but still is already acceptable for solving practical tasks. Also, the reviewed open-source toolchain has significantly lower hardware requirements and faster operation speed, which makes it possible to use it on embedded computers of adaptive control systems. [1]

This opens up the opportunity for online reconfiguration of the control system hardware, which is relevant for:

- Adaptation of the interfaces of sensors and actuating devices;
- Changes in the structure of the controller implemented on FPGA;

- Choice of the optimal connection structure of hardware neural networks etc.

Simultaneously, compared to the solutions based on partial reconfiguration, the approach suggested in the mentioned paper [1] requires no preliminary preparation of any FPGA bitstreams.

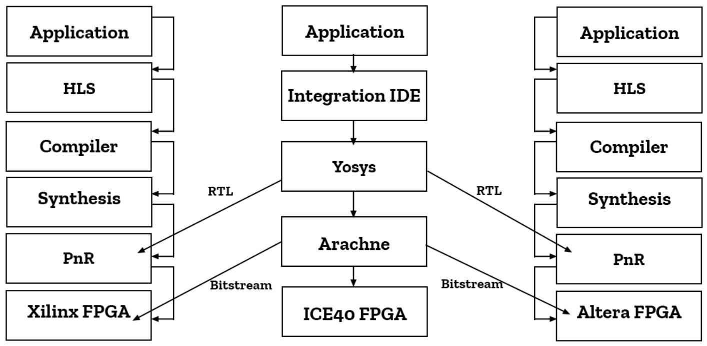

Fig. 5. Control System reconfiguration with Open FPGA tools

B. Reactive robotics using open-source fpgas

State-of-the-art open FPGA toolchain can be utilized to design reconfigurable computing for basic reactive robot behaviors. It is an easy, cost-effective, and reliable approach to developing complete reactive robot behaviors involving sensors and actuators. As experimental validation, the line-following and the obstacle avoidance reactive behaviors have been successfully implemented in a commercial mBot robot using the open FPGA IceZum Alhambra board and Icestudio visual editor. [3]

C. Drawbacks in Open-source FPGA tools

Authors argue that most open-source tools are slower than commercial tools. Verilator can be an exception to the statement since it offers performance comparable to many commercial simulators. Despite that, top-of-the-range commercial simulators perform better than Verilator in many aspects. [5]

Another problem with open-source simulators is encrypted IP blocks. This feature is rarely supported in the present solutions and if developers wish to use IP from vendors such as Xilinx, all simulation models supplied are encrypted. [5]

VI. Conclusion

Lerner and Schankerman [10] indicate the interaction between OSS and proprietary software, i.e. companies sell proprietary software while contributing to Open Source development, and users extensively mix and match the two. Thus, it is safe to conclude that both OSS and proprietary software share cost synergies. Finally, it is stated that Open-source software generates positive economic and quality impacts on society. [4]

As mentioned by [11], FPGAs offer a compelling approach to achieving the performance, power, and cost benefits of domain-specific architectures while retaining the flexibility to adapt to changing computational requirements and supporting a wide range of application domains. Therefore, open-source tools for FPGAs are a significant method to confront broader academic and commercial challenges within the sector.

Finally, it is essential to state that, by providing more information about their device architectures, the cooperation of key FPGA stakeholders in the development of open-source FPGA tools can generate powerful and profitable results.

References

[1] A. Romanov, M. Romanov, and A. Kharchenko, “Fpga-based control system reconfiguration using open source software.” Institute of Electrical and Electronics Engineers Inc., 4 2017, pp. 976–981.

[2] “Tf4pga - the gcc of fpgas,” https://f4pga.org/, 2022.

[3] J. M. Cañas, J. Fernández-Conde, J. Vega, and J. Ordóñez, “Reconfigurable computing for reactive robotics using open-source fpgas,” *Electronics (Switzerland)*, vol. 11, 1 2022.

[4] P. Grzegorzewska, A. Katz, S. Muto, S. Pätsch, and T. Schubert, “The impact of open source software and hardware on technological independence, competitiveness and innovation in the eu economy,” 2021.

[5] “Introduction to open source fpga tools,” https://fpgatutorial.com/open-source-fpga-tools/, 2022.

[6] X. Tang, E. Giacomin, B. Chauviere, A. Alacchi, and P. E. Gaillardon, “Openfpga: An open-source framework for agile prototyping customizable fpgas,” *IEEE Micro*, vol. 40, pp. 41–48, 7 2020.

[7] A. Li and D. Wentzlaff, “Prga: An open-source fpga research and prototyping framework.” Association for Computing Machinery, Inc, 2 2021, pp. 127–137.

[8] H. J. Liu, “Archipelago-an open source fpga with toolflow support,” 2014. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-43.html

[9] “Gcc for fpga: Symbiflow open source toolchain,” https://www.hackster.io/news/gcc-for-fpga-symbiflow-open-source-toolchain-254cf1ab15ff, 2022.

[10] J. Lerner and M. Schankerman, “The comingled code: Open source and economic development,” 2010.

[11] K. E. Murray, M. A. Elgammal, V. Betz, T. Ansell, K. Rothman, and A. Comodi, “Symbiflow and vpr: An open-source design flow for commercial and novel fpgas,” *IEEE Micro*, vol. 40, pp. 49–57, 7 2020.