EXOSTIV LABS

White Paper

Why we should scale FPGA tools.

Frédéric Leens – Exostiv Labs
May 2021

I. FPGA is not often 'first-time right'

In the 2020 edition of the **Wilson Research Group Verification Survey [1],** Siemens EDA, shows that **a staggering 83% share of FPGA designs went to production with bugs in 2020.** The results show that this share has remained relatively stable over the past decade.

The stability is remarkable. It means that the tools and methodologies have evolved sufficiently to keep pace with the evolution of the FPGA complexity (typically, this complexity doubles every two years for FPGA chips – see for instance Wikipedia – List of Xilinx FPGAs). **However, it also means that no matter the adoption of more advanced design and verification techniques, we have been unable so far to significantly improve the success rate of FPGA design.**

It used to be commonplace to consider FPGA engineers as non-rigorous versions of their ASIC counterparts. FPGA used to be for glue logic at best, FPGA was the cheap ASIC, with a simplified design flow composed of almost free tools and a limited verification strategy – mostly done in the lab with FPGA boards. Well, that was in the 90's…

30 years later, the FPGA has become one of the most advanced types of chips built with the latest process node. The design & verification methodology has evolved too and today we commonly find **code coverage, assertions, and constrained random stimulus generation** in the arsenal of tools of OEM companies that use FPGA as a target technology.

II. Use the end of the flow

Over the past decade, methodologies and testbench base-class libraries like UVM, OVM, OSVVM, UVVM, … have all gained some momentum in FPGA design, with notably UVM being used in close to 50% of the projects (reference [1]).

If over 80% of the FPGAs go to production with bugs – and if engineers properly apply methodologies – it means that the FPGA design & verification flow as it is today – is unable to detect bugs in some cases. **Worth noting, a close look on the FPGA flow shows that verification is largely based on simulation.**

When asked about their biggest verification challenge, engineers quote the difficulty to **reach perfect coverage closure (reference [1]).** It seems reasonable to think that a bug that arrives in production has escaped the checks conducted with the applied verification techniques. **It means that these advanced simulation-based techniques are important but not sufficient.**

Simulation is often incomplete (imperfect coverage) **or inaccurate** (models do not match reality) and this, no matter how the testcases were created.

To resolve these issues, we think that:

- Instead of modeling the environment, verification should include sessions with the FPGA integrated in the environment.
- Extending the number of test cases and the time in function in a realistic environment augments the chances to improve the test coverage.

  For instance, we know that creating randomness is challenging when we just simulate a few milliseconds of FPGA behavior in a testcase. A FPGA board placed in its target environment easily extends the time in service to seconds, minutes, hours. We just need to turn it on…

For these reasons, we think that efficient FPGA design and verification must be completed with tests conducted on physical boards.

It is rather intriguing, by the way, that ASIC and SoC designers continue to consider that FPGA prototyping is an essential part of the ASIC design flow[1] - demonstrating the technical advantage to include a 'hardware debug and test' phase in any complex chip development, including this of advanced FPGAs.

III. FPGA complexity has outgrown the traditional debug and test methods.

Test, debug, and verification with a hardware FPGA board requires the following:

1) The target FPGA board – the final board or a sufficiently close prototype.
2) An ability to place the board in the target environment.
3) An ability to watch the FPGA 'from inside'.

The goal is to have a visibility that scales with the chip complexity[2].

Traditionally, JTAG embedded logic analyzer have been used as the default tool (Chipscope / Xilinx ILA, Signal Tap, …). Unfortunately, because these tools have extremely limited capture depths – a few kilobytes at best, they cannot be used for exceptionally large datasets or capture scenario that exceed a few hundred microseconds.

Basically, observing a FPGA during extended times in its environment requires extracting at *a lot of data*. With devices commonly running at 300-400 MHz and beyond, the amount of data generated and processed in a FPGA is gigantic. Extracting actionable data for debug and verification supposes leveraging resources located inside and outside the FPGA for trace extraction and analysis. In other words: **bandwidth and storage** first.

These resources are the cost for doing an efficient debug and verification job with a FPGA running at speed in its environment. This kind of 'massive real time data capture' can be a game changer to avoid bugs in production. This is what we refer to as ***'massive real-time data capture'.***

The main benefits of being able to capture massive trace in real time from FPGA are:

- the speed of execution,
- the realism of the test scenarios
- and thereby the ability to quickly produce realistic test conditions that include random or rare events.

Looking at massive trace means a quicker and better understanding of the FPGA behavior. We think that capturing trace under the most realistic conditions shows you things that simulation will not.

Being able to explore the behavior of a system beyond the first (milli-)second – and this under realistic conditions proves to be invaluable in the test and verification process.

If scaling the resources enables massive capture, **triggering, filtering, and cross clock domain capture features** ultimately allow you to **navigate the complexity of the FPGA** and master it.

---

[1] It is also interesting to see that the three largest EDA vendors (Cadence, Synopsys, and Siemens EDA) have all released new FPGA prototyping platforms during the first 4 months of 2021.

[2] We herewith provide 2 real-world examples of what massive data capture can bring – both in the form of video replays – click here to access them: Demo 1 / Demo 2.

IV. Conclusions

In this paper, we have reviewed the reasons why we think that running part of the testing and verification of advanced FPGA chips with boards or prototypes is necessary to fill the gaps left by other verification methodologies based on simulation. We think that the traditional boundary-scan based approaches have failed to grow with the evolution of FPGA devices and that there is a need to scale the FPGA tools that we use.

Exostiv Labs solutions provide Gigabyte-range storage and gigabits per second of bandwidth coupled with triggering, filtering, and cross clock domain event detection. These solutions, easy to implement and integrated in the FPGA vendor flows, create the foundation of a renewed – at scale – exploration of complex FPGA chips.

With such an actionable and massive visibility into the chip, you will not risk going to production with bugs anymore.

References:

[1] Wilson Research Group Verification Survey, 2020 edition, by Siemens EDA.
[2] Massive Real-time FPGA Data Capture – A game changer to avoid bugs in production – Webinar replay
[3] Record FPGA during 1 hour – really – blog post on www.exostivlabs.com with video demo.

For more information about Exostiv Labs products: https://www.exostivlabs.com