SoC Verification of Ethernet Subsystem in Multi-Platform Environments

Tijana Misic*,

* Vtool doo, Milutina Milankovića 1g, Belgrade, Serbia,

tijanam@thevtool.com

Abstract

There are a lot of things that can be challenging when talking about SoC hardware and software verification. Engineers are torn between high requirements to achieve verification results fast and the complexity of the system. This paper gives an overview of the flow that is followed in the multi-platform verification of the Ethernet Subsystem. There is a concept of a verification plan with a list of important segments of Ethernet SoC verification. Each segment is verified on the most suitable platform. Our success story tells about Ethernet features tested in formal verification, RTL and GLS simulation, emulation platform and FPGA. The re-usability of testbench parts over different platforms is highly promoted during the whole process. The accomplished results demonstrate that this flow can be used as a roadmap for other teams that could have some dilemmas in making decisions in the verification strategy.

Keywords: Ethernet, SoC verification, emulation, Palladium, AVIP

Introduction

Functional simulation hits the limits in the verification of complex System-on-Chip (SoC) due to its single-threaded nature. Selecting the test suite smartly and running some of it in emulation could be the key to achieving high-quality as well as fast results. The emulation is an extremely valuable step in the verification process when speaking about performance measurement, use cases and firmware testing. This paper presents a true story about the flow we use in SoC digital verification to accomplish efficiency and meet strict deadlines. There is always a question about the selection of tools and verification methodologies that we need in order to ensure a bug-free tape-out. We divided the verification plan into sections and decided what the best approach is for the verification of each section. All Intellectual Properties (IPs) developed in-house are verified in IP verification environments which are then re-used at the SoC level following principles of Universal Verification Methodology (UVM). The software is verified in hardware/software (HW/SW) co-verification where specific software is loaded into memory and code is executed by the processor inside the design. Also, it is important to note that in the whole process, Cadence Verification IPs (VIP), Accelerated VIP (AVIP), and memory models are used inside the UVM verification environment. The team was supported by Cadence support. The final netlist is released only after all RTL and Gate-Level Simulation (GLS) regressions have passed. We listed the issues we faced during the process and stated the way to mitigate them. A few directions for further improvement of the process are given in the closing point. Our experience can be a safe route for other teams when planning efficient verification of complex SoC.

Related Work

Hardware Emulation is the technique of prototyping real SoC design and accelerating the speed of design execution [1]. The main advantage is that emulation runs hundreds to thousands times faster

than RTL simulation. Limitations of emulation are the following: we are missing some checkers, removing physical models in the emulation netlist, and higher costs on the emulation side. There are good examples of the usage of emulators in power-aware verification [2] and firmware verification [3]. It is important to understand the difference between simulation and emulation. This difference can enable us to make a correct decision about which features are to be verified in simulation and which are to be verified in emulations. The emulation platform is an implementation of the design executed on special-purpose hardware such as Palladium, a parallel processor engine produced by Cadence. RTL simulators are software packages that simulate expressions written in one of the hardware description languages. It is important to provide a verification environment that will enable hardware/software co-verification early in the process, producing quick results and preventing possible bugs in the later stages [4]. Robustness of the verification environment, early performance results and power measurements are the key requirements of verification teams [5]. This approach could improve the time for SoC bring-up to be measured in hours instead of weeks as was the case before.

Methodology

The main motivation of this paper is to present how flexible and scalable solutions which can be used by multiple users can help us achieve reliable and fast verification results. The most efficient way is to create a single flow that can be reused on different platforms. Makefiles for software compilation, scripts for preloading memory, and running test scripts, once developed, can be used on multiple platforms: RTL simulation, emulation and Field programmable gate arrays (FPGA) platforms.

For RTL and GLS simulations, we use Xcelium as a simulator and vManager for regression. A part of the verification process is done in formal verification inside Jasper Gold. More complex stuff and software verification is done on Palladium. We have an adaptable flow for verification in Xcelium and Palladium. This can be expanded to FPGA prototyping platform Protium, but it was not available on the project. In Figure 1, we give a timeline of the usage of different tools on the project during the verification process.

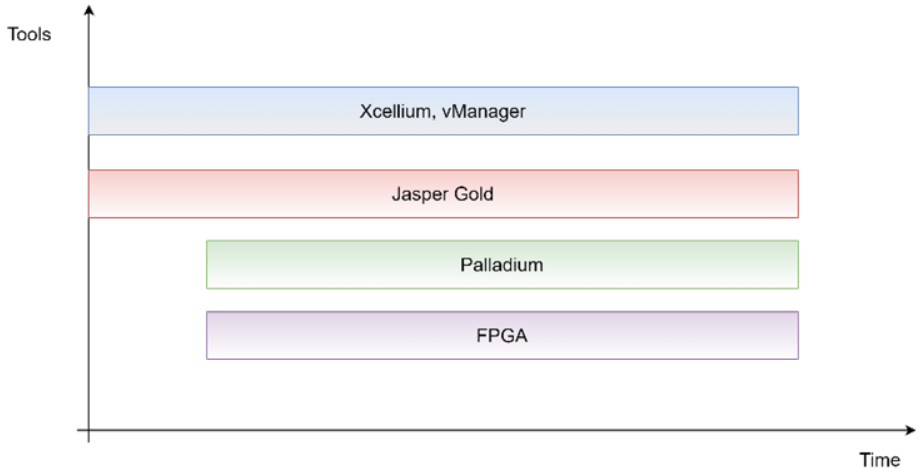

Figure 1 Tools used on the project depending on time.

The RTL verification is implemented as a combination of UVM tests and complex tests for HW/SW co-verification. In HW/SW co-verification, we have verification drivers and tests written in C, which are compiled and stored inside memories and the code is executed by the Central Processor Unit (CPU) inside the design. Part of this software integrates production drivers which were done in the early phase of the process (UART, SPI, Flash). The complex tests that cover all data paths and different speeds are selected for running in GLS regression.

For the purpose of describing the flow, we selected the Ethernet subsystem since it is connected to the external ports, its software is complex and performance results for traffic over the Ethernet interface were extremely important for the customer. The Gigabit Ethernet subsystem consists of an Ethernet controller, Physical Coding Sublayer (PCS), and Ethernet Physical Layer (PHY). The verification is implemented in multi-platform environments, which include formal verification, RTL and GLS simulation, emulation on Palladium, and FPGA prototyping.

Although the multi-platform verification was a challenging task, we managed to reuse things from one platform to another, such as Makefiles, software, and parts of tests. The schematic preview of the Palladium testbench is given in Figure 2. First, we developed Makefiles that can accommodate different requests and make them work on all supported platforms confirming Cadence's statement that this is supported in their toolset. The same code for the bare-metal boot of the CPU is used in RTL simulation and on Palladium. The code for bare-metal drivers was reused in the same way as well (e.g., Ethernet drivers and drivers for other peripherals). Consequently, Makefiles for the compilation of bare-metal software code are the same on both platforms.

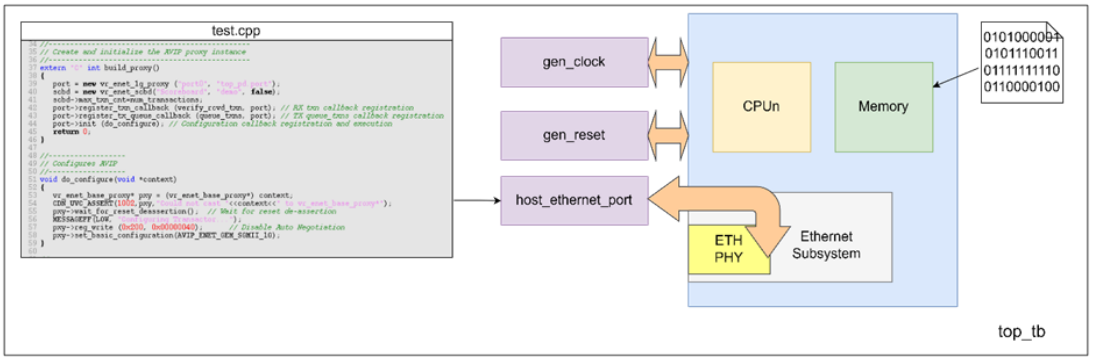

Figure 2 Palladium Testbench

Scripts for running tests and memory preloads from RTL simulation are slightly updated and reused on Palladium. The UVM part of tests used for driving Ethernet VIP in RTL simulation is not reused on Palladium. We developed a new code for Ethernet AVIP in C++ to score faster execution on the emulation platform as recommended by Cadence support and [6].

Some changes have been requested in order to create a design netlist that is run on Palladium. All external reset and clock sources necessary for the test are identified and generated in the Palladium testbench.

Memories used in RTL simulation are replaced by synthesizable models. Off-chip memories are modeled by flash memory models provided by Cadence. Ethernet PHY is replaced by an empty wrapper, since the PHY model is not synthesizable. This impacts AVIP connection to be moved to the PCS interface instead of SerDes boundary of Ethernet PHY. This is an important difference in RTL simulation, since Ethernet PHY is present in the design and the full data path is tested from memory over the Ethernet controller, PCS, and PHY to Ethernet VIP. In Palladium, we are testing part of this path bypassing Ethernet PHY.

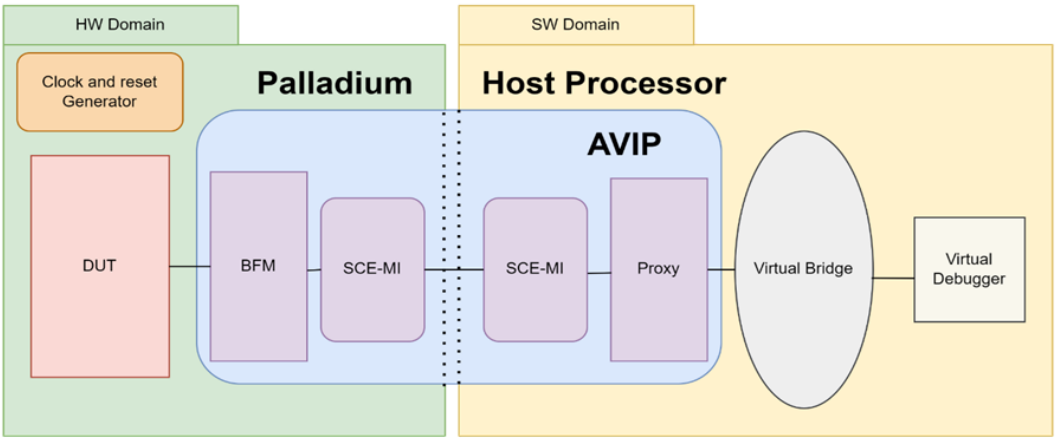

Figure 3 Verification Environment for testing on the Palladium

The process of running tests is finally automated for a different group of tests. The system is set on Palladium on the Cadence cloud. The compiled numbers showed that the design can easily fit into the Palladium Z1 board, so we started verification with one Z1 board on cloud and Cadence licenses for

Ethernet AVIP and memories. Later in the verification process, Cadence VirtualBridge is attached to the AVIP proxy. This step gives us access to the protocol stack on the remote machine which enables us to test certain traffic patterns of interest early in the development process. Performance testing and the possibility to test complex use cases are the most valuable out-turns of using Palladium in the verification process. The preview of the Palladium environment for running Ethernet tests is given in Figure 3.

The process of Ethernet verification is split into different technologies depending on the features in order to achieve results of the highest quality as soon as possible. There are 4 different platforms for testing purposes on the project: formal verification, RTL simulation, emulation, and FPGA. Our test plan is prepared in such a way that different functionalities are verified on the platform that is the most suitable for it. The tests are prioritized according to the importance of the features they cover. The test plan is given in Table 1. It is reviewed in several review cycles by architects, designers, the verification team and software developers.

Table 1 Verification plan for Ethernet multi-platform verification

Feature	**Test**	**Platform**	**Priority**	
Connectivity and integration of interrupts and GPIOs		Formal	1	Jasper Gold
Register access over the configuration port	UVM	SoC	1	Xcelium
Master port access to all targets	UVM	SoC	1	Xcelium
Functional Safety features	UVM	SoC	2	Xcelium
Basic transfer over Ethernet interfaces	C-UVM	SoC	2	Xcelium, Palladium
Low Power scenarios	C-UVM	SoC	3	Xcelium, (can be done in Palladium)
Gate Level scenarios	C-UVM	SoC	3	Xcelium
Performance testing	C (with AVIP)	SoC	3	Palladium
Bring-Up preparation	C	SoC	3	Palladium/FPGA
Software API verification	C (with AVIP)	SoC	4	Palladium/FPGA

It is important to have well-defined, measured and analyzed verification metrics. Constrained random stimulus enabled by the usage of AVIPs gives an efficient way of verifying different device states. Automatically collected functional coverage provides us with a view of the final quality that verification accomplished and a clear overview of what is covered in overall verification. Special attention should be paid to tracking configuration settings (which blocks were included in the platform at the time when coverage is collected) and what kind of firmware is run.

Conclusion

In this paper, there is a specific flow for SoC verification which is presented. It is demonstrated that verification should be done at different levels - block, subsystem, and top-level and on different platforms. The horizontal and vertical portability of the verification environment and test sequences should be supported in the sense that the parts of the environment can be used at the block, subsystem and SoC level, but also for verification on different platforms (RTL simulation, Palladium, Protium). The verification process needs to be planned and carefully reviewed by the design team, architects, verification and software team. Our success story can be used as a safe path for the verification of complex subsystems inside modern SoCs. Finally, our aspiration is to eliminate the boundary between tools, and enable users to run the same test on different platforms: RTL simulation, emulation, the FPGA prototype, and post-silicon validation, making verification and debugging fast and easy.

Copyright

References

[1] R Frank Schirrmeister, Robert Kaye, "Reducing Time to Point of Interest with Accelerated OS Boot," ARM TechCon, Santa Clara, California, USA, 2014. [Online]. Available: https://community.arm.com/cfs-file/__key/telligent-evolution-components-attachments/01-2112-00-00-00-00-58-30/Reducing_5F00_Time_5F00_to_5F00_Point_5F00_Schirrmeister.pdf

[2] Ruchi Misra et al., ”An Accelerated System Level CPU Verification through Simulation Emulation Co-Existence,” in Proceedings of Design and Verification Conference (DVCON), Munich, Germany, 2022.

[3] D. Ciaglia, T. Winkler, J. Kundrata, " Unified firmware debug throughout SoC development lifecycle," in Proceedings of Design and Verification Conference (DVCON), Munich, Germany, 2022.

[4] S. Morgansgate, J. Grinschgl, D. Lettnin, " Development of a Core-Monitor for HW/SW Co Debugging targetting Emulation Platform," in Proceedings of Design and Verification Conference (DVCON), Munich, Germany, 2022.

[5] Riad Ben Mouhoub, “Palladium Z1 and Protium S1: The Fastest Way to Make Your System-on Chip Prototyping a Success Story,” CDNLive, Silicon Valley, USA, 2018. [Online]. Available: Palladium Z1 and Protium S1: The Fastest Way to Make Your System-on-Chip Prototyping a Success Story

[6] “Ethernet Accelerated VIP User Guide”. 2022. [Online]. Available: https://support.cadence.com/