RISC-V VERIFICATION AUTOMATION FRAMEWORK

Pavan Dheeraj Kota, Kilho Chang, Stacey Dao
E-mail: {pkota, khjchang, ssdao}@ucdavis.edu

1 INTRODUCTION AND MOTIVATION

The functional verification process ensures that a hardware design correctly implements its specification. Practically, this involves applying carefully crafted stimulus to the design-under-test (DUT) and comparing its behavior against a trusted reference model at both register-transfer-level (RTL) and gate-level netlist stages. Traditionally, establishing this verification environment is tedious and time-consuming, primarily due to the diverse industry standards and unique specifications associated with various chip designs. On the other hand, establishing one such flow for a processor design would be highly beneficial, as a processor is virtually part of every chip design.

RISC-V, with its open and modular Instruction Set Architecture (ISA), presents a practical opportunity for automation in verification. High-quality, freely available reference simulators (such as Spike and Sail) and advanced stimulus generators like Google's RISC-V DV simplify automation significantly. As RISC-V designs expand across both industry and academia, encompassing extensions for atomic operations, multiplication, floating-point arithmetic, vector processing, and more, the sheer volume of required tests has become impractical for manual methods. Therefore, automated flows have become essential for validating specification compliance of cores implementing various combinations of the available extensions, and ensuring software compatibility.

2 PROPOSED FRAMEWORK

The proposed framework integrates standard verification components—the ISA simulator, stimulus generator, and SystemVerilog/UVM testbench—under a single automation script or Makefile. A single command initiates the entire verification flow, including configuration, test generation, compilation via the GNU RISC-V toolchain, golden reference simulation, RTL simulation (using open-source tools like Verilator), trace comparison, and comprehensive report generation. Design engineers only need to specify the DUT, select ISA extensions, and optionally set test parameters; the framework then automatically adapts accordingly. Due to the fully scripted nature, the flow seamlessly integrates into continuous integration (CI) pipelines, offering rapid feedback on code changes. Moreover, the use of open-source tools reduces costs and licensing constraints, facilitating widespread adoption across industry and academia. Finally, the automated environment promotes reusability and consistency across different verification projects, further increasing the long-term efficiency of the verification efforts.

3 VALIDATION ON THE IBEX CORE

We will initially validate this automation framework using the Ibex core. Ibex provides a well-documented, open-source verification environment with built-in integrations for RISC-V DV and Spike, significantly easing initial setup. Its straightforward two-stage, in-order micro-architecture facilitates focused debugging on the verification flow itself rather than complex internal micro-architectural corner cases. Once successfully validated on Ibex, the framework can readily extend to more advanced RISC-V designs by simply incorporating a lightweight adapter for capturing retirement information and updating the configuration to match the new core. The underlying generator, reference model, and comparator will remain unchanged. Through rigorous testing of the Ibex core, we will establish best practices and documentation, which will significantly ease future integrations with more complex cores.

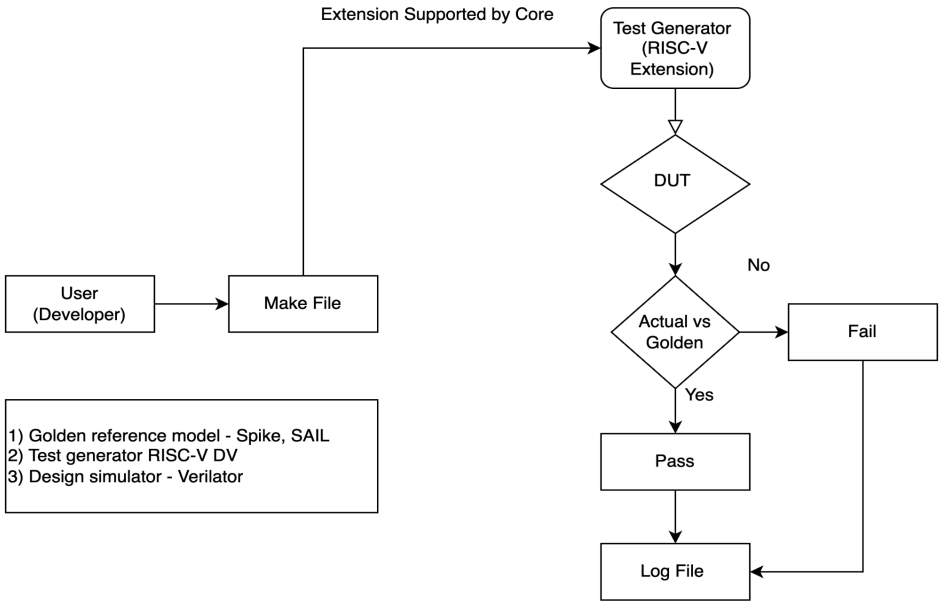

Figure 1 RISC-V Verification Automation Framework

4 TIME SCHEDULE

Proposed activity	Proposed period
Problem definition and an overview	April 1 - April 20
Literature review and recreating the Ibex co-simulation environment	April 20 - May 11
Set up automation environment (Developing MakeFile and subsequent integration into the co-simulation environment)	May 11 - May 18
End-to-end framework test (Verify the robustness of the integration)	May 18 - Jun 1
Extend automation environment to support additional RISC-V cores	No commitment

5 CONCLUSION

This project will deliver a comprehensive, automated verification framework that significantly reduces the verification effort while increasing confidence in the correctness of RISC-V cores. Given the accelerating global adoption of RISC-V, this automation approach is timely and impactful. By enabling industry and academic teams to develop robust, compliant processors efficiently, this project will contribute to a rapidly growing ecosystem of thoroughly verified RISC-V hardware, allowing companies and designers to "move fast without breaking things".

REFERENCE

[1] *03_reference/verification*, Ibex Core Documentation. [Online]. Available: https://ibex-core.readthedocs.io/en/latest/03_reference/verification.html. [Accessed: Apr. 17, 2025].