Implementation of Branch Treatment Strategies in the Ripes RISC-V Simulator

Silvia González-Rodríguez, Francisco José Alfaro-Cortés, Rafael Rodríguez-Sánchez,
Jesús Escudero-Sahuquillo
Departamento de Sistemas Informáticos. Universidad de Castilla-La Mancha. Spain

Abstract

RISC-V is an open standard instruction set architecture (ISA) developed at the University of Berkeley as an alternative to proprietary ISAs, like x86 and ARM. Its main goal is to provide a flexible and cost-effective platform for companies, universities, and developers to design processors without licensing fees. Additionally, it supports customization and performance optimization through specialized instructions. In education, simulators are key tools for understanding processor operation, showing details of instruction cycles and memory. The Ripes simulator, being open source, encourages customization and collaboration. This work enhances Ripes by adding a branch implementation and new branch resolution strategies. The interface has also been improved to simplify option selection, making it a complete and more useful tool for teaching.

Introduction

RISC-V is a free and open-source instruction set architecture (ISA) [1] developed at UC Berkeley as an alternative to proprietary ISAs such as x86 and ARM. It aims to provide a flexible and modular platform for processor design, enabling hardware designers to create processors efficiently and without expensive licensing fees. One of the key goals of RISC-V is to democratize processor design, allowing companies, universities, and even individual developers to contribute without commercial restrictions. This is achieved by providing an open specification that anyone can use, modify, and adapt to their needs. In addition, RISC-V also seeks to improve the security and customization of systems by allowing the definition of customized instructions (i.e., extensions) that optimize the processor implementation performance in specific areas. As more companies and research communities adopt this architecture, RISC-V is gaining ground as a viable and scalable alternative to traditional options on the market.

Simulation tools play a crucial role in the field of computer architecture for teaching and understanding complex systems. They provide a controlled, interactive environment where students can visualize the datapath and control signals of processors, account for the instruction cycles or memory accesses, and experiment with various configurations. In the context of RISC-V, there is a significant number of simulators, such as Ripes, which are invaluable educational resources. Ripes offers a user-friendly, graphical interface that allows students to explore the behavior of a RISC-V processor step by step, making complex concepts like pipelining, data dependencies, and branching easier to grasp.

As part of a final degree project at our school, we have introduced significant improvements to Ripes to support a more detailed study of conditional branch handling. These modifications include the ability for students to select between different branch strategies, such as varying the number of delay slots in the execution pipeline. By providing multiple configurations, students can better understand the impact of different design choices on processor performance and program execution. Additionally, the interface has been upgraded to simplify the selection of processor configurations and branch options, making the tool even more accessible for educational purposes. We're looking into integrating these enhancements into the public repository, offering a valuable resource to the Ripes user community.

Simulation Tools

Simulation tools are key to analyzing and predicting the behavior of complex systems without the need for costly or unfeasible experiments. They allow the user to optimize processes, evaluate scenarios, and make informed decisions with less risk. They also facilitate learning and innovation in multiple disciplines. These tools have become key resources in modern teaching by allowing students to experiment and observe the behavior of systems in real time, visualizing abstract concepts more concretely and understandably. By providing immediate feedback, these programs encourage more autonomous and deeper learning, helping students to identify errors and reinforce their understanding.

In the field of RISC-V education, tools such as Ripes [2] or Spike [3] provide students with an interactive environment where they can observe how instructions are executed, how registers are managed, and how the different components of the processor, such as execution units, memory, and cache, interact. These simulators allow detailed visualization of each step of the instruction cycle,

facilitating the understanding of complex concepts, all in a controlled and accessible environment.

Ripes

Ripes [2] is an interactive simulation tool that allows students and developers to explore and understand the operation of a processor based on the RISC-V architecture. It is a graphical simulator that provides a visual interface where one can observe how program instructions are executed step-by-step inside a RISC-V processor, showing the flow of data through registers, memory, and execution units in real time. Ripes allows users to study the instruction execution cycle in detail, helping to understand fundamental concepts such as pipelining, data dependencies, and branch handling. This tool is especially useful in teaching, as it offers an accessible and visual way to study processor architecture without the need for physical hardware or complex resources. On the other hand, Ripes is an open-source project, which means that its source code is available to anyone who wants to examine it, modify it, or contribute to its development. This makes it an accessible tool for students, researchers, and developers who wish to customize or adapt it to their specific needs. Being open source, Ripes also encourages collaboration and learning, as users can share new functionalities or extensions of the tool.

What distinguishes Ripes from other simulators is its ability to customize and modify both the processor architecture and instructions, making it easy to teach the more advanced concepts of processor design. Students can experiment with different ISA configurations (e.g., RV32I or RV64I), extensions (e.g., M or C), processor implementation improvements (e.g., pipelining, forwarding, etc.), and observe how different configurations affect performance and program execution. In addition, Ripes supports both simulation and debug modes, allowing users to follow the execution of a program step-by-step, examine the contents of registers and memory at each clock cycle, or look at the values of control signals. This improves understanding of the instruction execution and reinforces key learning concepts related to the RISC-V architecture.

Improvements introduced in Ripes

As a result of a Final Degree Project developed in our School, Ripes has been upgraded with several configurations for conditional branch handling. By default, Ripes resolves the conditional branch in the EX stage (which means it has two delay slots) and applies the strategy of predict-not-taken. To allow the student to study the effects of different branch implementations, options for executing branches with one or three delay slots (i.e., resolving the conditional branch in the ID or MEM stages, respectively) and applying the delayed branch strategy have been added. This allows the student to choose between two different implementations of the conditional branch with one, two, or three delay slots.

Figure 1 shows the flow of program execution using the delayed branch strategy solving the conditional branch at the ID stage. Remark that in this strategy the instruction immediately after the branch is not flushed even if it is resolved as taken.

To this end, we modified the implementation of the different available datapaths and changed the connections of the components involved in the resolution of the conditional branch and their position in the different stages of the datapath. In addition, the interface for choosing the type of processor has also been improved, and the selection of the branch options has been integrated into this interface. This simplifies the selection of the configuration options for the student. The new interface is shown in Figure 2.

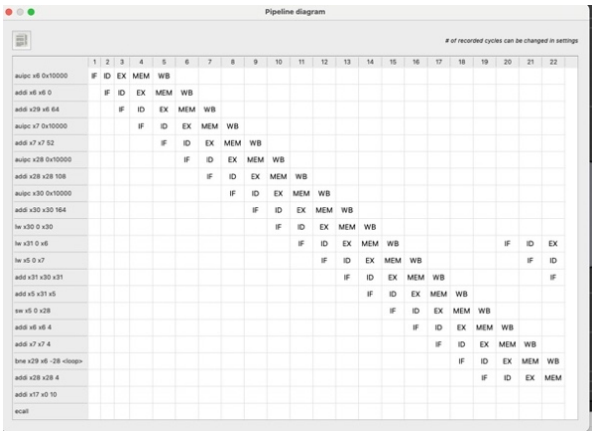

Figure 1 Ripes's Delayed branch strategy execution.

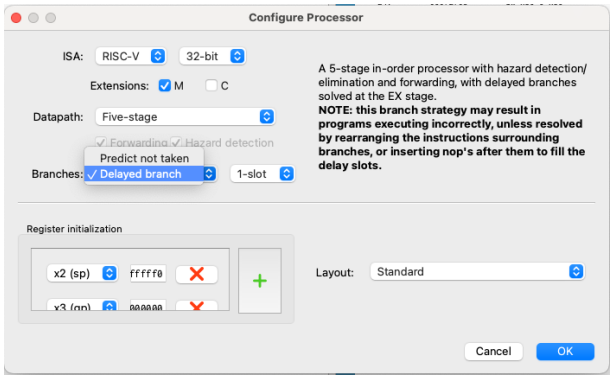

Figure 2. New Ripes processor selection interface.

Conclusions and Future Work

This work will be used from next year onwards in our School in a compulsory course in Computer Architecture in the second year of the Degree for Computer Engineering. It is also intended to incorporate what has been developed here into the tool's public repository so that it can be made available to anyone who might be interested.

As future work, other end-of-studies projects have already been offered, consisting of improving the memory management that Ripes incorporates, incorporating the execution of floating-point instructions, allowing dynamic scheduling, the use of interrupts, and some other ideas that may be interesting for teaching.

Acknowledgements

This work has been funded by the Junta de Comunidades de Castilla-La Mancha, under the project SBPLY/21/180225/000103 and the Spanish Ministry of Science, Innovation and Universities under the projects PID2021-123627OB-C52 and TED2021-130233B-C31. Moreover, this work is also funded by Grant Cátedra PERTE Chip University of Castilla-La Mancha (TSI-069100-2023-0014) funded by the Ministry of Digital Transformation (Cátedras PERTE Chip Programme) through the “European Union NextGenerationEU/PRTR”.

References

[1] A. Waterman, Y. Lee, R. Avizienis, H. Cook, D. Patterson and K. Asanovic, "The RISC-V instruction set," 2013 IEEE Hot Chips 25 Symposium (HCS), Stanford, CA, USA, 2013, pp. 1-1.

[2] Petter, M. (2020). Ripes: A RISC-V simulator with a visual interface for educational use. GitHub. https://github.com/mortbopet/Ripes.

[3] Sundararajan, V., & Karczmarek, R. (2018). Spike: RISC-V ISA simulator. GitHub Repository. https://github.com/riscv/riscv-isa-sim.