Formal models for monotonic pipeline architectures

J.-P. Bodeveix, A. Bonenfant, T. Carle, M. Filali, C. Rochange

February 7, 2025

Due to ecological concerns and current lab policies regarding funding for international travel, the team is hesitant about participating in the workshop in person. However, we agree to register for the workshop and deliver our presentation remotely via Zoom.

In this proposal, we consider pipeline architectures intended for deployment in real time critical systems [GCR+23]. Our goal is the development of formal models for these architectures. The utilization of formal methods will permit the verification of relevant properties for architectures designed according to these models. As a matter of fact, with respect to the instruction lifecycle over the pipeline, we are interested by properties like the strict progress property and the monotonicity hyper property [BCF+24].

Our study builds upon two existing scalar pipeline architectures: [HR20] and [GBCR24]. Both architectures adhere to a scalar design: each stage contains at most one instruction, at most one instruction exits a stage per cycle, and instructions are pipelined in order. Key differences lie in the number of instructions permitted within a stage during a cycle and the presence of "true" stages with multiple incident stages. [HR20] provides rigorous proofs of their architecture, validated through model checking. In contrast, [GBCR24] adopts a formal development methodology, using the Coq proof assistant to model and prove properties. Both developments are tailored to specific architecture topologies and instruction sets.

Our work focuses on developing a generic pipeline architecture model. We represent the architecture as an acyclic graph with designated pre and post stages, where instructions are defined by their paths through this graph. A formal model is constructed, based on abstract properties of the graph's topology and the instruction set, and subject to a set of clearly defined assumptions. The central benefit of this approach is that fundamental properties of the generic architecture are proven only once. Pipeline designers can then "inherit" these proven properties for their particular designs by demonstrating that their designs adhere to the model's assumptions. To ensure the practical applicability of our model, we evaluate the validity of our assumptions against the two existing pipeline architectures, as described in [HR20] and [GBCR24].

TLA+ was selected for its set-based notation, which simplifies the formal representation of the problem. Close collaboration with the hardware team is essential for developing both the model and its associated properties. Although property verification is in progress, we anticipate that TLA+'s explicit proof language will be beneficial for articulating and discussing the arguments behind the fundamental hardware mechanisms. Finally, the model checking and simulation features of the TLA+ toolbox offer valuable support for this study.

References

[BCF+24] Jean-Paul Bodeveix, Thomas Carle, Elie Fares, Mamoun Filali, and Thai Son Hoang. Verifying hyperLTL properties in Event-B. In Silvia Bonfanti, Angelo Gargantini, Michael Leuschel, Elvinia Riccobene, and Patrizia Scandurra, editors, *Rigorous State-Based Methods - 10th International Conference, ABZ 2024, Bergamo, Italy, June 25-28, 2024, Proceedings*, volume 14759 of *Lecture Notes in Computer Science*, pages 255–261. Springer, 2024.

[GBCR24] Alban Gruin, Armelle Bonenfant, Thomas Carle, and Christine Rochange. Modelling and proving the monotonicity of processor pipelines in Coq. In *22nd ACM-IEEE International Symposium on Formal Methods and Models for System Design, MEMOCODE 2024, Raleigh, NC, USA, October 3-4, 2024*, pages 12–21. IEEE, 2024.

[GCR+23] Alban Gruin, Thomas Carle, Christine Rochange, Hugues Cassé, and Pascal Sainrat. Minotaur: A timing predictable RISC-V core featuring speculative execution. *IEEE Trans. Computers*, 72(1):183–195, 2023.

[HR20] Sebastian Hahn and Jan Reineke. Design and analysis of SIC: a provably timing-predictable pipelined processor core. *Real Time Syst.*, 56(2):207–245, 2020.