Design Exploration of RISC-V Soft-Cores through Speculative High-Level Synthesis

Jean-Michel Gorius[1], Dylan Leothaud[1], Simon Rokicki[1] and Steven Derrien[2*]

[1]Univ Rennes, Inria, CNRS, IRISA [2]Université de Bretagne Occidentale

Abstract

The RISC-V ecosystem is quickly growing and has gained a lot of traction in the FPGA community, as it permits free customization of both ISA and micro-architectural features. However, the design of the corresponding micro-architecture is costly and error-prone. We address this issue by providing a flow capable of automatically synthesizing pipelined micro-architectures directly from an Instruction Set Simulator in C/C++. Our flow is based on HLS technology and bridges part of the gap between Instruction Set Processor design flows and High-Level Synthesis tools by taking advantage of speculative loop pipelining. Our results show that our flow is general enough to support a variety of ISA and micro-architectural extensions, and is capable of producing circuits that are competitive with manually designed cores.

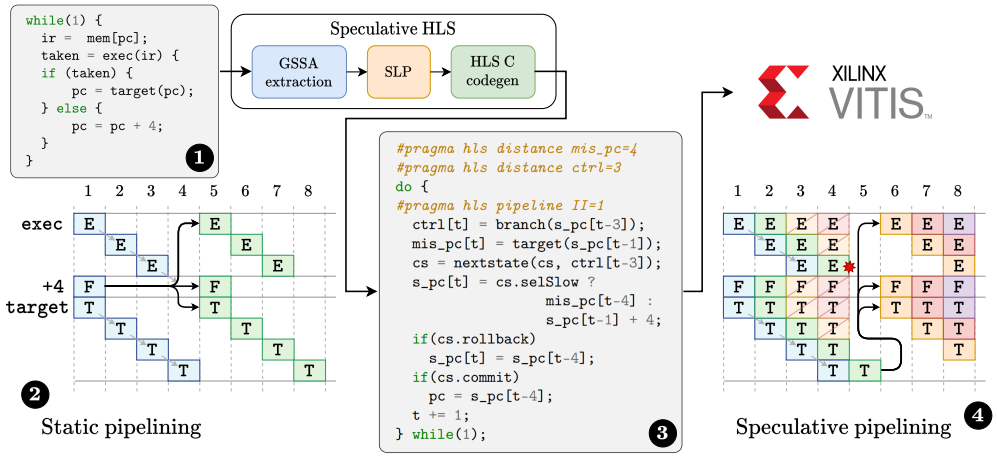

Figure 1: *Our SLP source-to-source transformation flow. The toolchain takes C code* ❶ *as an input and produces transformed C code* ❸*.* ❷ *and* ❹ *show the respective schedules of the input and output code.*

Speculative HLS of RISC-V Cores

The RISC-V ecosystem is quickly growing and has gained a lot of traction in the FPGA community, as it permits free customization of both the ISA and the micro-architecture.

Retargeting a compiler to a new ISA is a widely studied problem, but automatically synthesizing the corresponding instruction set micro-architecture has received less attention. Existing tools and technique offer significant room for improvement: they either lack generality [1, 2] or operate from low-level structural models that are not fundamentally different from RTL specifications.

In the meantime, High-Level Synthesis (HLS) technology, which compiles C and C++ code directly to hardware circuits, has continuously improved. For example, several recent research results have shown how High-Level-Synthesis techniques could be extended to synthesize efficient speculative hardware structures [3, 4]. In particular, Speculative Loop Pipelining (SLP) appears as a promising approach as it can handle both control-flow and memory speculations within a classical HLS framework [5].

Our work bridges part of the gap between Instruction Set Processor design flows and High-Level Synthesis tools. We show how to take advantage of SLP to automatically synthesize in-order pipelined micro-architectures from Instruction Set Simulator (ISS) models written in C, focusing on the RISC-V ISA. Our contributions are the following:

- We show how SLP can serve as a foundation to perform fully automatic micro-architectural synthesis from a behavioral description of a processor, in the form of an ISS. We extend SLP to support the synthesis of in-order pipelined CPU micro-architectures and their hazard recovery logic.
- We evaluate our approach in terms of supported features (both from an ISA and micro-architectural perspective) and quality of results (performance and area). Our results show that our

*Corresponding author: simon.rokicki@irisa.fr

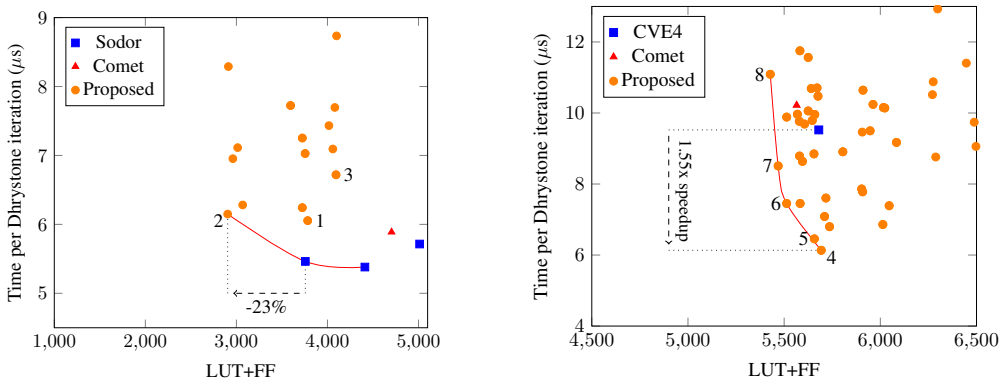

Figure 2: *Area/performance result set for 21 variants of RV32I core (left) and 105 variants of RV32IM core (right) synthesized through our approach. Results for Sodor, Comet and CVE4 are reported on the figure as baselines.*

flow can handle complex mechanisms like branch prediction and hardware Control-Flow Integrity while providing QoR similar to manual designs.

Our source-to-source transformation flow, depicted in Figure 1, accepts C code as input and produces speculatively pipelined C code targeting an HLS toolchain. The latter can then be compiled/synthesized to obtain an RTL-level description of the processor core.

The key idea in SLP resides in rescheduling critical operations to extend their dependence distance before calling the HLS tool. The HLS static pipelining pass will harness this additional schedule slack to produce more aggressive (i.e., deeper) pipelined schedules with higher clock speeds. The output C code shown in Part ❸ of Figure 1 is produced from the input C code in Part ❶. Its corresponding execution trace is provided in Part ❹.

Results

To demonstrate that our proposed approach can generate competitive pipelined micro-architectures, we generate a large set of processors by exploring different speculation setups: no speculation on the register file, pipeline interlocking, or forwarding. We also modify the latency of the different operational blocks used in the ISS to explore several different pipeline depths. As baselines, we also synthesize the three Sodor pipelined cores (2-, 3-, and 5-stage pipelines) and the CV32E40P core [6]. We synthesize two configurations of the Comet processor [7], RV32I and RV32IM. As our generated cores do not implement RISC-V CSR registers, we remove the CSR unit from the Sodor and CVE4 cores.

Our experiments target an Artix7 XC7A200TISBG-1L and use Vitis HLS 2021.2 as the HLS backend. Performance results were obtained by executing the Dhrystone benchmark, compiled using `newlibc`.

Results of the automatic design space exploration are provided in Figure 2. The leftmost part represents the results obtained for the RV32I ISA, and the rightmost part represents the results obtained with the RV32IM ISA. The generated micro-architectures are slower than the Sodor and Comet baselines for the RV32I ISA, while we are able to generate faster cores for the RV32IM target (55% faster than CVE4). Our generic approach generates extra control logic on the critical path of the RV32I cores, reducing their maximal frequency. On the other hand, the critical path of RV32IM cores is located in the multiplication/division unit. The extra logic that hinders the RV32I performance could be optimized during the SLP transformation, but this improvement is left for future work.

References

[1] Gai Liu, Joseph Primmer, and Zhiru Zhang. "Rapid generation of high-quality RISC-V processors from functional instruction set specifications". In: *2019 56th ACM/IEEE Design Automation Conference (DAC)*. IEEE. 2019, pp. 1–6.

[2] Peter Yiannacouras, Jonathan Rose, and J Gregory Steffan. "The microarchitecture of FPGA-based soft processors". In: *Proceedings of the 2005 international conference on Compilers, architectures and synthesis for embedded systems*. 2005, pp. 202–212.

[3] Steven Derrien et al. "Toward Speculative Loop Pipelining for High-Level Synthesis". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39.11 (2020), pp. 4229–4239.

[4] Lana Josipović, Andrea Guerrieri, and Paolo Ienne. "Speculative Dataflow Circuits". In: *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. FPGA '19. Seaside, CA, USA: Association for Computing Machinery, 2019, pp. 162–171. ISBN: 9781450361378. DOI: 10.1145/3289602.3293914.

[5] Jean-Michel Gorius, Simon Rokicki, and Steven Derrien. "SpecHLS: Speculative Accelerator Design Using High-Level Synthesis". In: *IEEE Micro* 42.5 (2022), pp. 99–107. DOI: 10.1109/MM.2022.3188136.

[6] Andreas Traber et al. "PULPino: A small single-core RISC-V SoC". In: *3rd RISCV Workshop*. 2016.

[7] Simon Rokicki et al. "What You Simulate Is What You Synthesize: Design of a RISC-V Core from C++ Specifications". In: *RISC-V Workshop 2019*. 2019, pp. 1–2.