A Pluggable Vector Unit for an Open-Source 64-bit RISC-V Implementation

Vincenzo Maisto

Università degli Studi di Napoli Federico II, Italy, and Hensoldt Cyber GmbH, Germany

Alessandro Cilardo

Università degli Studi di Napoli Federico II, Italy

This work presents a prototypical microarchitectural design of a Vector Unit supporting the RISC-V Vector extension, version 0.8. The design is meant to be integrated in the execution stage of a scalar RISCV processor pipeline as a regular functional unit. To implement the Vector extension, an implementation of the base RISC-V ISA was necessary. As an industrial use case, we chose Ariane from ETH Zurich, an open-source scalar 64-bit RISC-V core also used by Hensoldt Cyber GmbH for its products. Nevertheless, our work is not inherently dependent on this choice, allowing the presented design to be re-used in other implementations. In fact, the Vector Unit microarchitecture was designed from scratch to minimize dependencies with architectural features of the base core and their microarchitectural implementation. To reduce design complexity and ease the testing activities, minimization of both architectural impact and source code modifications was also identified as a key requirement. In this work, we present the challenges we addressed at the microarchitectural level and the necessary modifications to the base microarchitecture, deriving a set of features which the base core must support to integrate the Vector Unit.

The work also analyses the execution model of vector instructions under microarchitectural design choices dictated by performance requirements, such as splitting vector instructions into micro-operations and allowing those to execute with chaining. Dependency extraction from vector instructions and micro-operations are discussed along with their implications. Finally, a common protocol to trace these dependencies is derived. This protocol dictates the interactions between micro-operations and the Vector Register File and covers the execution model also in its corner cases.

As an additional contribution, the presented design offers a standard framework which can host functional units from third parties. These units could be independently designed with any techniques (e.g. fully combinatorial or sequential) and with no constraints or burden caused by microarchitecture-specific complexity. Instead, third-party functional units can have any behaviour and a very simple interface since only an opaque wrapper needs to be provided, while the interaction with the rest of the Vector Unit microarchitecture is mostly hidden. This approach is aimed to ensure extensibility and maintainability, which were also identified as key requirements, allowing the microarchitecture to gain a broad use and stand the long-term evolution of the RISC-V architecture and open-source implementations.