Study of compiler optimizations for supporting mixed-precision arithmetic operations on RISC-V architectures

Research project

RISC-V is an open-source architecture that has gained considerable popularity in recent years due to its flexibility, simplicity, and extensibility. The TRISTAN (Together for RISC-V Technology and Applications) project, within which this research activity is framed, aims to expand the adoption of RISC-V architectures in Europe by making them competitive with alternatives on the market. To this end, TRISTAN includes activities to develop the hardware and the entire supporting software infrastructure.

Mixed-precision arithmetic involves performing calculations using different data types (e.g., single-precision or half-precision floating-point numbers) to balance result accuracy and computational cost. In this context, the role of the compiler is critical in associating the appropriate data types with program variables and generating efficient machine code. This research activity aims to study, design and implement compiler optimization techniques to exploit arithmetic operations in mixed precision by finding a trade-off with the final accuracy of the computation. In particular, the researcher will explore the impact of mixed-precision arithmetic on the open-source RISC-V architectures developed as part of the TRISTAN project, including dedicated architectural extensions.

Plan of activities

The activities planned for the research project consist of the following phases:

Phase 1: Preliminary analysis and knowledge acquisition (2 months).

1. Literature review: Conducting a detailed study of the current scientific literature regarding compiler optimizations for mixed-precision arithmetic on RISC-V architectures. Understand the challenges, existing techniques, and recent developments in this field.

2. Study of RISC-V architecture: Deepening the knowledge of RISC-V architecture, its instructions, registers, and performance and power consumption characteristics.

3. Fundamentals of mixed-precision arithmetic: studying the concepts of single-, double-, and half-precision arithmetic, including conversion and rounding techniques between different accuracies.

Phase 2: Optimization framework design (2 months).

1. Goal definition: Identifying specific optimization goals, e.g., improving performance, reducing energy consumption, or balancing performance and accuracy.

2. Analysis design: Developing a set of analyses to identify opportunities for applying mixed-precision arithmetic to the source code.

3. Design of transformations: Define code transformations that allow mixed-precision arithmetic to be applied safely and efficiently.
Phase 3: Implementation of the framework (6 months).

1. Compiler development: Implementing the optimization framework in the compiler, extending existing functionality to support mixed-precision arithmetic.

2. Integration with RISC-V architecture: Ensuring that the optimized compiler is compatible and optimized for the RISC-V architecture.

Step 4: Performance evaluation (2 months).

1. Data collection: Performing tests on benchmarks and reference applications to evaluate the performance and energy efficiency of the optimized compiler compared to baseline implementations.

2. Analysis of results: Evaluating the performance achieved by the optimized compiler and compare it with the results of conventional implementations to understand the benefits of the proposed optimizations.

Step 5: Documentation and final presentation (1 month).

1. Reporting: Documenting the work done, including the results of the evaluations and the research project.

2. Presentation preparation: Creating a presentation to communicate the findings and conclusions of the research project.