Automating RISC-V Custom Instruction Integration leveraging High-Level Synthesis

SIEMENS

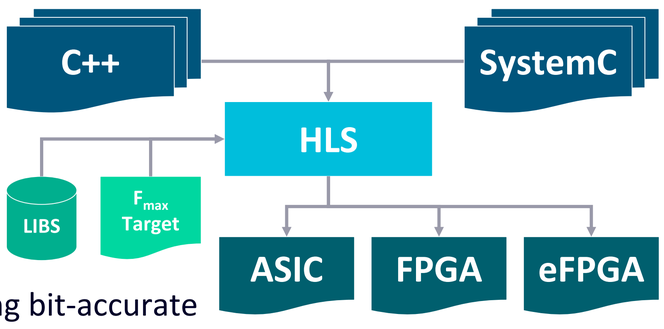
Florian Egert, Bernhard Fischer
Electronics Design and Integrated Circuits, Siemens
Contact: florian.egert@siemens.com, bernhard.bf.fischer@siemens.com

Background

- **Performance**: Maximizing performance within a tight area and power budget is of increasing demand in today's system development.
- **Specialization**: One key strategy to meet specialized workload requirements is extending processors with application-specific hardware acceleration.
- **Time-to-Market**: An adaptable and automated design flow is crucial to ensure rapid time-to-market.

High-level Synthesis

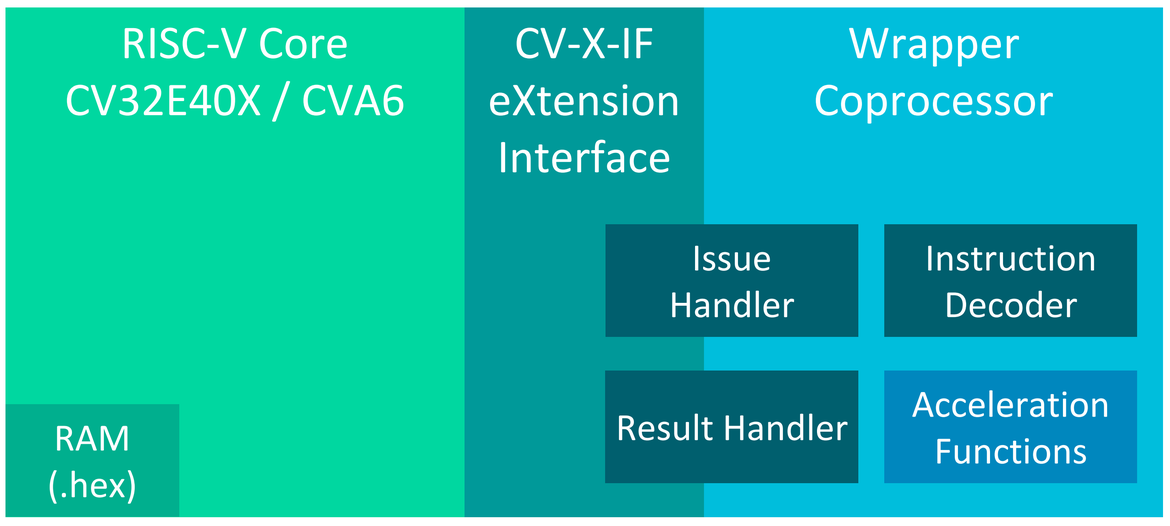
- **High-level to RTL**: HLS supports synthesis of RTL designs (e.g., in VHDL or Verilog) from high-level description languages (e.g., C, C++, or SystemC) [1].
- **Architecture Exploration**: HLS enables the evaluation of multiple RTL variants based on one high-level description, enhancing productivity and reducing manual coding errors.



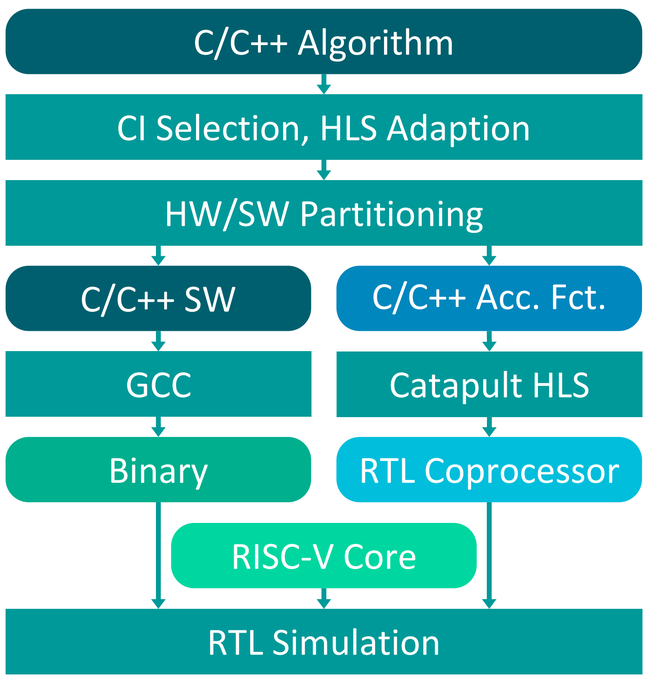
- **Verification and Validation**: Using bit-accurate data types in the high-level language ensures that computations match the synthesized RTL bit-for-bit, enabling partial validation of the algorithm in the high-level description and creation of expected results for the RTL verification.

Coprocessor Generation

- **Hardware Acceleration**: This work proposes an automated RISC-V flow
  - based on Custom Instructions (CIs).
  - to reduce time-intensive and error-prone manual coding.
  - for a broad range of applications.
- **HLS**: The flow leverages the High-level Synthesis (HLS) tool Catapult HLS [1] to generate RTL Wrapper Coprocessor implementations from a set of Custom Instructions defined by the designer in form of C/C++ accelerator functions.
- **CV-X-IF**: The generated coprocessors are compliant to the OpenHW Group's Core-V eXtension interface [2], ensuring compatibility with any processor supporting CV-X-IF, such as the CV32E40X or CVA6.



Hardware Acceleration Flow



- **CI Selection**: The designer must first analyze the given C/C++ application and define Custom Instructions in form of C/C++ accelerator functions, code snippets suitable for acceleration.
- **HLS Adaption:** Afterwards, the designer may adapt the selected C/C++ accelerator functions for HLS. While advanced HLS tools are able to synthesize almost all modern C/C++ constructs, certain coding styles, bit-accurate data types, pragmas, and HLS tool settings may improve results.
- **HW/SW Partitioning**: The HW/SW partitioning tool of the flow splits the designer's resulting C/C++ input into extracted C/C++ accelerator functions, and remaining C/C++ software including calls of the accelerator functions.
- **SW Compilation**: The C/C++ software is compiled utilizing GCC.
- **HW Generation**: The C/C++ accelerator functions are embedded in a CV-X-IF-compliant C/C++ wrapper coprocessor description. Based on this, Catapult HLS is used to generate the RTL coprocessor and deliver PPA estimations for the specified target systems and design constraints.
- **Integration, RTL simulation**: The generated RTL wrapper and RISC-V CV32E40X are integrated in a test environment, using Questa for RTL simulation to verify functionality and timing.

Results and Future Work

- **Automation**: As part of our previous work [3], we proposed analysis approaches of the potential reduction of working hours when utilizing the flow. HLS adds an important automation step to the flow.
- **PPA**: The generated system's optimization potential regarding performance, power, and area depend on the input algorithm and selected Custom Instructions. While a comprehensive PPA analysis for different use cases is ongoing within TRISTAN, first comparisons of accelerated systems with software-only solutions for an AES-128 algorithm suggest
  - a cycle count reduction of around 65% for the CV32E40X.
  - an area score of minimal impact of around 1000 compared to about 30000 for the CVA6.
- **CI Selection**: In future work, we plan to automate algorithm analysis and CI selection, which are, up to now, manually performed by the designer.

References

[1] *Catapult C++/SystemC Synthesis*. Siemens. Accessed: March, 2025. [Online]. URL: https://eda.sw.siemens.com/en-US/ic/catapult-high-level-synthesis/hls/c-cplus/.
[2] *CORE-V eXtension Interface*. OpenHW Group, 2021. URL: https://docs.openhwgroup.org/projects/openhw-group-core-v-xif.
[3] F. Egert et al. "A Methodology for Automating the Integration of User-Defined Instructions into RISC-V Systems based on the CV-X-IF Interface". In: *RISC-V Summit Europe 2024*. 2024. URL: https://riscv-europe.org/summit/2024/media/proceedings/posters/17_poster.pdf.

Acknowledgements
The TRISTAN project, nr. 101095947 is supported by Chips Joint Undertaking (CHIPS-JU) and its members and including top-up funding by the Austrian Research Promotion Agency (FFG) and the program "ICT of the Future" of the Austrian Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation and Technology (BMK).