Efficient Verification Framework for RISC-V Instruction Extensions with FPGA Acceleration

Yang Zhong[1,2], Zijian Jiang[3*], Keran Zheng[4*], Shuoxiang Xu[5], Yungang Bao[1,2] and Kan Shi[1,2]

[1] State Key Lab of Processors, Institute of Computing Technology, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences
[3] Beijing University of Technology [4] Imperial College London [5] ShanghaiTech University

Vision & Value

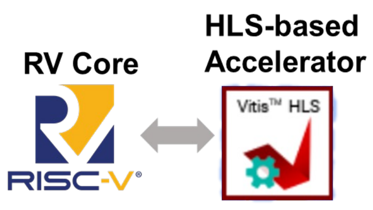

Figure 1: RISC-V core with extended HLS accelerator

- Basic RISC-V ISA contains common instructions, while the extended accelerators meet diverse needs.
- High-level synthesis (HLS) provides a way to build hardware accelerators further improving development efficiency.

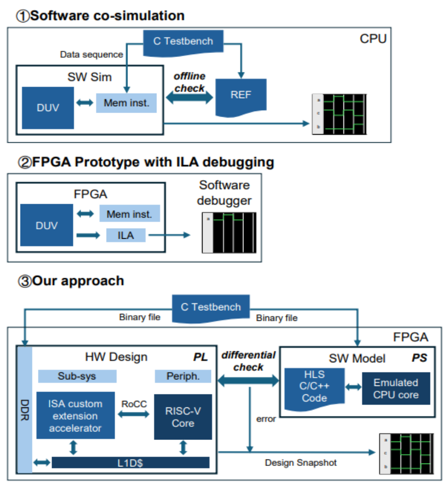

Figure 2: Comparison between the conventional verification approaches and the proposed approach

Traditional methods to verify HLS-generated designs:

C/C++ simulation.

- Lacks accurate latency information.
- May not catch issues related to hardware implementation details.

C/RTL co-simulation.

- Runs much slower than C/C++ simulation.
- May operate even more slowly when debugging features are enabled.

Our solution:

- Map both the RISC-V cores and HLS-generated hardware onto the FPGA, and corresponding software models on the hardened processor.
- Propose a set of verification-focus custom instructions for better system-level validation.

The overall architecture

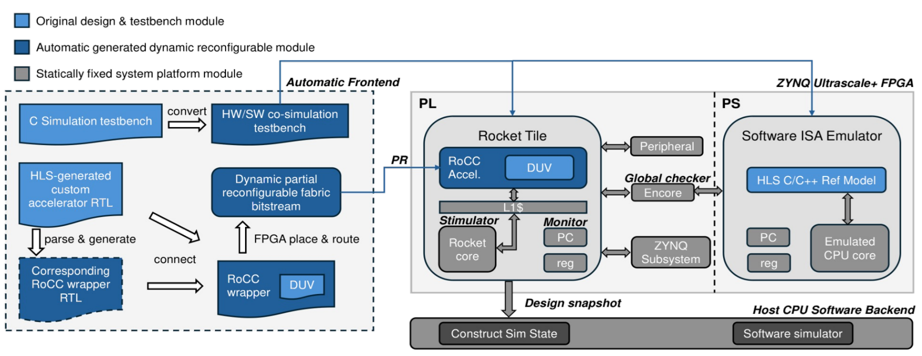

Figure 3: The overall architecture of RISC-V instruction extensions verification framework. Including automatic frontend, FPGA architecture and Host CPU Software Backend

Software Frontend

An automated scheme to:

- Integrate HLS-generated accelerators DUV into RoCC.
- Transformation a C simulation testbench into a co-simulation test program.

Software Emulation

The software reference model is placed on an ISA emulator running on the hardened processing system as a concrete implementation of the extended instructions.

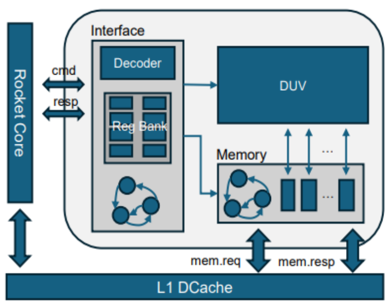

Figure 4: Architecture of the RoCC Wrapper

RoCC Wrapper

- Using the RoCC interface and protocol to link the RISC-V cores with accelerators.
- Automatically generate the decoder, register bank, cache bus convertor and FSM.
- Dynamically reconfigured to reduce deployment time when changing DUV.

Evaluation

Platform: Fidus Sidewinder board

- with a Xilinx Zynq UltraScale+ XCZU19EG FPGA and two 16GB DDR4 memories.

Host Server: connected to the server via JTAG.

- two AMD Ryzen 5950x 16-core processors
- Back-end simulator: ModelSim SE2019.4

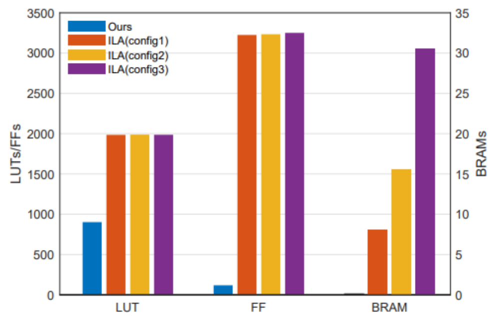

Figure 5: Comparison of area overhead between ILA and proposed approach. *The area overhead data are averaged by nine designs in MachSuite. Config 1, 2, and 3 correspond to a sampling depth of 1024, 2048 and 4096, respectively*

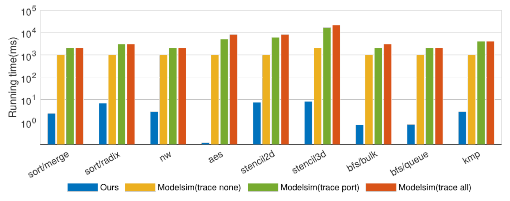

Figure 6: Comparison of running performance between our approach and software simulation.