DEPARTMENT OF
Electrical and Computer Engineering

11-203 Donadeo Innovation Centre for Engineering
9211-116 Street NW
University of Alberta
Edmonton, Alberta
Canada T6G 1H9

RISC-V FPGA-based CPU and Language

Objective

Design and implement a RISC-V compliant CPU on an FPGA. Define a high-level language and implement the toolchain necessary to run programs on the custom CPU.

Motivation

The RISC-V open standard instruction set architecture [1] has been gaining in popularity as the basis for custom processor design. Its open nature means that many of the barriers to design and commercialization of general or application-specific systems are much reduced. Designers have great flexibility in terms of how processor architectures are defined and realized. In addition, for student and enthusiasts, there are a plethora of example implementations that are actively maintained and reviewed, with rapid turnaround for bug fixes and feature additions [2].

For the U of A Electrical and Computer Engineering (ECE) program, having an in-house example of CPU design and a matching custom high-level programming language would provide the basis for further projects and learning by motivated students.

Most Suitable Background for the Project

Computer Engineering (traditional)

Description

There are three main design challenges:

1. Design and implementation of an FPGA-based RISC-V compliant CPU,
2. Definition of a RISC-V compliant high-level language, and
3. Design and implementation of the high-level language with demonstration on the CPU.

A key goal is to ensure that the complete system is implementable using components and resources readily available to a UofA ECE student. All aspects should be open source and available online, with GitHub or GitLab the preferred sites.

For the RISC-V compliant CPU implementation, it would make most sense to use an FPGA already familiar to ECE students. However, this is not mandatory.

Ideally, all a RISC-V instruction set will be implemented, but time constraints may get in the way.

The high-level language may be based on an existing language like Rust [3], Zig [4], or Go [5].

A stretch goal would be to design the system so that it can be augmented by radio-based meteor detection and tracking techniques.

Functional and Performance Requirements / Objectives

1. The FPGA-based CPU :
    a. Shall be compliant with the RV32I instruction set at minimum (extensions optional)
    b. Shall have general-purpose input/output pins
2. The custom language :
    a. Shall define basic mathematical and logic operations
    b. Shall define a minimum set of control operations and structures
    c. Shall support simple general-purpose input/output signal (GPIO) operations
    d. May support formatted character input/output
3. The custom language toolchain :
    a. Shall be supported on Linux and may be supported on Windows and MacOS

Specifications / Constraints

The project constraints include :

1. The FPGA system must be readily available to UofA students; this means either a Zynq or Cyclone based system
2. While the custom language may be fully specified, it is not necessary to support it all for the prototype. However, those features that are supported must be demonstratable. That is, the toolchain must be able to generate working programs based on a subset of the language and state of the CPU implementation. No partially-working implementations are permitted.
    1. Anything that is not yet working shall be identified for future development.

Information Resources / Links

References:

[1] Wikipedia contributors. "RISC-V." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 3 Jan. 2024. Web. 3 Jan. 2024.

[2] Various authors, Risc-V Cores, Github Collections, https://github.com/collections/riscv-cores accessed Jan 3, 2024

[3] Rust Team, The Rust Language, https://www.rust-lang.org, accessed Jan 3, 2024

[4] Zig Software Foundations, The Zig Language, https://ziglang.org, accessed Jan 3, 2024

[5] The Go Programming Language, https://go.dev, accessed Jan 4, 2024

Prototyping/Testing Resources

- FPGA systems available from the ECE 492 capstone, specifically various Terasic DE-xx systems (up to DE-10)
- Zynq-based FPGA systems are available, specifically the Zybo 7010. Zybo 7020 is available on request, and possibly other Zynq-based boards.
- The project sponsor may be able to provide some materials.

Intellectual Property Restrictions

Note: Public presentation and, hence, disclosure is a course requirement.

- Ideally the entire system is open source.
- Any intellectual property identified by the student team may be considered for assignment to the team.

Contacts

Steven Knudsen,
Tel: (403) 949 0089
Email: knud@ualberta.ca

Antonio Lara, ECE Lab Technician
Email: andarala@ualberta.ca