CPTR380 RISC-V RV32I FPGA Implementation

Jaron Brown & Ethan Jansen

***Abstract*—TBD**

***Index Terms*—ISA, RISC, RISC-V, RV32I FPGA, VHDL.**

I. Introduction

SINCE designing a custom ALU-based CPU last quarter in the Intro to Digital Design class, both of the authors developed an interest in implementing a more commonly supported instruction set architecture (ISA). When presented with the opportunity to pursue a project related to computer architecture in their Computer Architecture class, both authors jumped at the opportunity to implement another CPU design. They initially considered implementing the MIPS ISA but, after conducting more research, decided on implementing one of the RISC-V standard ISAs, RV32I. This decision was made based on the abundant documentation for the RISC-V ISAs, RV32I's relative simplicity, and the high commercial interest in these ISA specifications.

One of the dominant reasons many companies are developing RISC-V implementations is it open-source nature. The dominant ISAs (x86, ARM, MIPS, etc.) all have steep licensing fees. The RISC-V ISAs, in comparison, are completely free. Additionally, the closed-source nature of the dominant ISAs limits the freedom those designing implementation of these ISAs. Having a completely open-source ISA just makes sense–it lowers the barrier to entry for designing hardware to implement standard ISAs and allows for more innovation in the realm of CPU design.

II. A Brief Introduction to RISC-V

The original design of RISC-V was begun in 2010 at the University of California, Berkeley. It was designed not only as an academic learning aid but also a set of ISAs that could have commercial viability. As its name suggests, RISC-V is a RISC (Reduced Instruction Set Computer) architecture. This means, in comparision to CISC (Complex Instruction Set Computer) ISAs, RISC ISAs have simple instructions. This means more individual instructions will be required to complete a given task compared to CISC ISAs but also allows the hardware implementation to be much simpler than that for CISC ISAs. In fact, implementations of x86-64, the most widely adopted CISC ISA, breaks its CISC instructions into "microinstructions" which are easier to implement in hardware, similar to RISC ISAs.

There has been reference to RISC-V ISAs. The reason for the plural ISAs is RISC-V is actually a collection of ISAs, ranging from 32-bit to 128-bit standards and supporting various levels of complexity by way of extensions to the base ISAs. Some official extensions include the M extension which adds multiplication and division support, the A extension which adds support for atomic instructions, and the F and D extensions which add support for single- and double-precision floating numbers, respectively. Developers are encouraged to design their own compatible extensions to customize the base RISC-V ISA to their specific application.

Manuscript created March, 2024.

III. Project Goals

In this project, version 2.1 of the RV32I ISA will be implemented in VHDL and tested on a Xilinx Artix 7 FPGA (Field-Programmable Gate Array). RV32I is the RISC-V base integer ISA and implements the core functionality of the RISC-V architecture family. Although RV32 supports pipelining, this projects implemetation will not implement this feature due to time constrains. I/O, multiplication and division, full support for C code compilation, and pipelining, are stretch goals of this project. The authors hope to continue this project beyond their class and continue to refine the design and add more features.

IV. Implementation Overview

This project's implementation of non-pipelined RV32I consists of a few core components:

- instruction memory
- data memory
- a register block
- an ALU (Arithmetic Logic Unit)
- a control module
- various other components related to sign-extension, incrementing the program counter, and routing signals.

Unlike the author's previous CPU designs which used and accumulator, this implementation uses a register block with 32 32-bit wide registers. This is in keeping with the RV32I standard and allows for more flexibility when programming the CPU.

V. Conclusion

TBD

References

[1] RISC-V Foundation, *The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Document Version 20191213*, pp. i–30, Dec 2019. [Online]. Available: https://riscv.org/wp-content/uploads/2019/12/riscv-spec-20191213.pdf

Jane Doe Biography text here without a photo.

IEEE Publications Technology Team In this paragraph you can place your educational, professional background and research and other interests.