Project 4 Proposal

Zekai Cheng (zc2747), Denzel Farmer (df2817)
Johannes Losert (jal2340), Luobin Ni (ln2516)
Riona Westphal (raw2183), Ning Xia (nx2173)

March 21, 2024

Introduction

This proposal outlines the objectives and planned execution for our project 4 implementation. Our goal is to enhance the VeriSimpleV RISC-V pipeline from project 3 into an out-of-order processor based on the R10K architecture. We will attempt to implement several advanced features aimed at optimizing performance.

Base Design

The foundation of our design is an out-of-order processor inspired by the R10K architecture. Our base design will take the Tomasulo implementation presented in class, and extend it to include a re-order buffer (ROB) split from the reservation stations (RS). In addition, we will replace the by-value style of register renaming from Tomasulo with 'true register renaming'. In this scheme, values will remain in a single register file rather than be passed through the pipeline, and we will add/extend components to track the states of physical registers as either free or in use.

Our design will include multiple functional units for different instruction operations, separate 256-byte instruction and data caches, and an (initially) simple branch predictor.

Advanced Features

Time permitting, our more ambitious goal is to incorporate the following advanced features into our design:

- (Major) Superscalar 2-way execution to improve throughput
- (Major) Early branch resolution to minimize delays caused by branch predictions
- (Minor) A more sophisticated branch predictor to avoid expensive mispredictions
- (Minor) Implementation of instruction (and potentially data) prefetching to reduce cache miss penalties
- (Minor) Associative caches to improve cache utilization
- (Minor) A robust GUI debugger to facilitate testing and debugging

Milestone Schedule

1. **Milestone 1 (3/28):** Complete the reservation station module along with a testbench to verify its functionality.
2. **Milestone 2 (4/4):** Implement all base features and the GUI debugger, although we may not handle more complex memory operations or have other advanced features implemented.
3. **Milestone 3 (4/11):** Complete implementations for memory operations and our 'early branch resolution' advanced feature.
4. **Milestone 4 (4/18):** Finalize the remaining components, implement as many of our remaining advanced features as possible, and conduct extensive testing.

Group Charter

Our team commits to the following guidelines to ensure a productive and cooperative working environment:

- Each member will dedicate approximately 8 hours per week to the project.
- Weekly working meetings will be held on Monday afternoons and evenings, where we will work (potentially in smaller sub-teams) to implement modules.
- We will hold additional meetings for big picture task discussion and planning on Wednesdays at 10:30 am.
- Time conflicts have been discussed, and no major issues were identified.

Tasks and Responsibilities

Although the group will work together on the whole project, each member will 'own' certain tasks that are easily isolated:

- **Denzel**: Management/logistics, GUI debugger
- **Zekai**: Instruction and data caches
- **Luobin**: Expanded functional units
- **Riona**: Sophisticated branch predictor
- **Ning**: Instruction/data prefetching
- **Johannes**: Associative caches