Developing RISC-V Educational Labs to Support ZPD-Based Scaffolding in Processor Education (MSc project)

Kieren Hamilton,
Supervised by Ivan Ling, University of Southampton, United Kingdom

Introduction

Given the rapid growth of the hardware industry, academia has yet to formalize the implementation of cutting-edge educational techniques into its labs, often relying on traditional styles, which are designed to maintain a minimum floor of competency, rather than push students to learn further. In recent years, the work of Vygotsky and his theory about Zone of Proximal Development (ZPD) has been used to structure labs in computer science REF and in the teaching of languages REF in Australia to the effect of higher learning outcomes, increased satisfaction for students, and an increased student capacity for labs. The theory proposes that a student will learn best when the content is in their ZPD, meaning it is just outside their current ability, which represents a task difficult enough to stimulate cognitive growth, yet not so difficult as to exceed the learners capacity for engagement. This research proposal aims to run a ZPD scaffolded RISC-V labs to study its effects on learning outcomes in the context of hardware, while also developing a generalizable framework for its implementation across engineering disciplines.

Project Description

A labs in the form of an educational RISC-V devkit will be developed using 2 implementations of RISC-V cores designed to be compatible with the RV32I instruction set and of varying architectural complexity (designs can be found here). The Devkit will include an environment with automated testing, offering instant feedback for some lab exercises, as well as integration with the RISC-V toolchain, enabling cross-compiled C binaries to be simulated into the cores.

The labs will be designed with ZPD factors which can be turned off and on, enabling experimental data collection about the efficacy of ZPD scaffolding on learning outcomes. More information can be found in Table 1.

The lab structure, inspired by Cadence lab material, will consist of a multiple lab scripts and directories inside the Devkit where learners can build and test their own designs. A linux environment will be used, to make use of open-source design tools for hardware.

ZPD Factor	Description	Implementation
Timed content	The content is spaced out rather than provided all at once, promoting mastery and avoiding cognitive overload.	Processor design concepts can be split into stages: machine code, architecture, pipeline hazards, compilation, implementation of branch prediction/forwarding, and protocols
Choice over task	Students may choose from labs of varying difficulty, with higher complexity linked to potential for higher marks.	Multiple architectures are available, ranging from single-cycle designs to full 5-stage pipelines, allowing students to select on challenge preferance.
Immediate feedback	Timely feedback keeps students engaged and able to iterate quickly through the learning process.	Automated testbenches provide verification. Feedback verbosity can be adjusted can be adjusted, offering detailed messages for beginers and minimal hints for advanced learners.
Optional enrichment	Offering advanced students challenging tasks, keeping them in their ZPD and allowing to stretch beyond the original task.	Students capable of self motivated learning can engage in optional extras like implementing custom branch prediction or extending processor core functionality.

Table 1: Description and implementation of ZPD factors

All students will be given sufficient support to complete core requirements regardless of which ZPD factors are enabled.

Methodology

2 groups of studnets will undertake this laboratory - one control group running through a traditional style labs and the ZPD group with ZPD factors enabled.

These labs will run in parallel, with a test at the beginning and end to show baseline knowledge and learning outcomes after completion. In addition, a questionnaire will be collected to measure student satisfaction.

Learning outcomes about the RISC-V processor concepts include

- Ability to implement basic algorithms at machine code level.
- Understand the 37 basic RV32I instructions, and ability to construct basic binary programs.
- Proficiency with synthesizable SystemVerilog code.
- Understand SystemVerilog structs and basic data types.
- Understand the modules that go into a Harvard style CPU architecture.
- understand Control signals involved in branching and their implementation.
- Grasp pipelined processor architecture concepts.
- Understand forwarding and its implementation.
- Understand branch prediction and its implementation.

Data protection plan

Type of Data Collected

- Academic data: pre-test and post-test scores assessing learning outcomes
- Questionnaire responses: Student satisfaction and feedback
- Demographic data: Limited, anonymized information such as age range, gender, and year of study (if collected)
- Lab interaction data: Usage logs indicating which ZPD factors were enabled and student progress.

Data collection and storage

Ethical consideration

Timeline and Key Milestone