Grupo de Arquitectura y Tecnología de Computadores

Learning Computer Architecture with a Visual Simulation of RISC-V Processors

Esteban Stafford, Borja Pérez, and Jose Luis Bosque

Department of Computer Science and Electronics Engineering, Universidad de Cantabria

Financiado por la Unión Europea NextGenerationEU

Plan de Recuperación, Transformación y Resiliencia

MINISTERIO DE CIENCIA E INNOVACIÓN

AGENCIA ESTATAL DE INVESTIGACIÓN

Abstract

This work presents an interactive approach to teaching computer architecture through Logisim Evolution, enabling students to construct and debug single-cycle and pipelined processors. Enhancements to Logisim Evolution allow dynamic editing of components, reinforcing theoretical concepts with hands-on experience. The proposed laboratory assignments progressively guide students through key processor concepts, improving engagement and comprehension.

Contributions

New components for RISC-V processor.

- **Standard** Register file, ALU, SE.
- **Editable** Main decoder, ALU decoder, forwarding, and hazard detection.

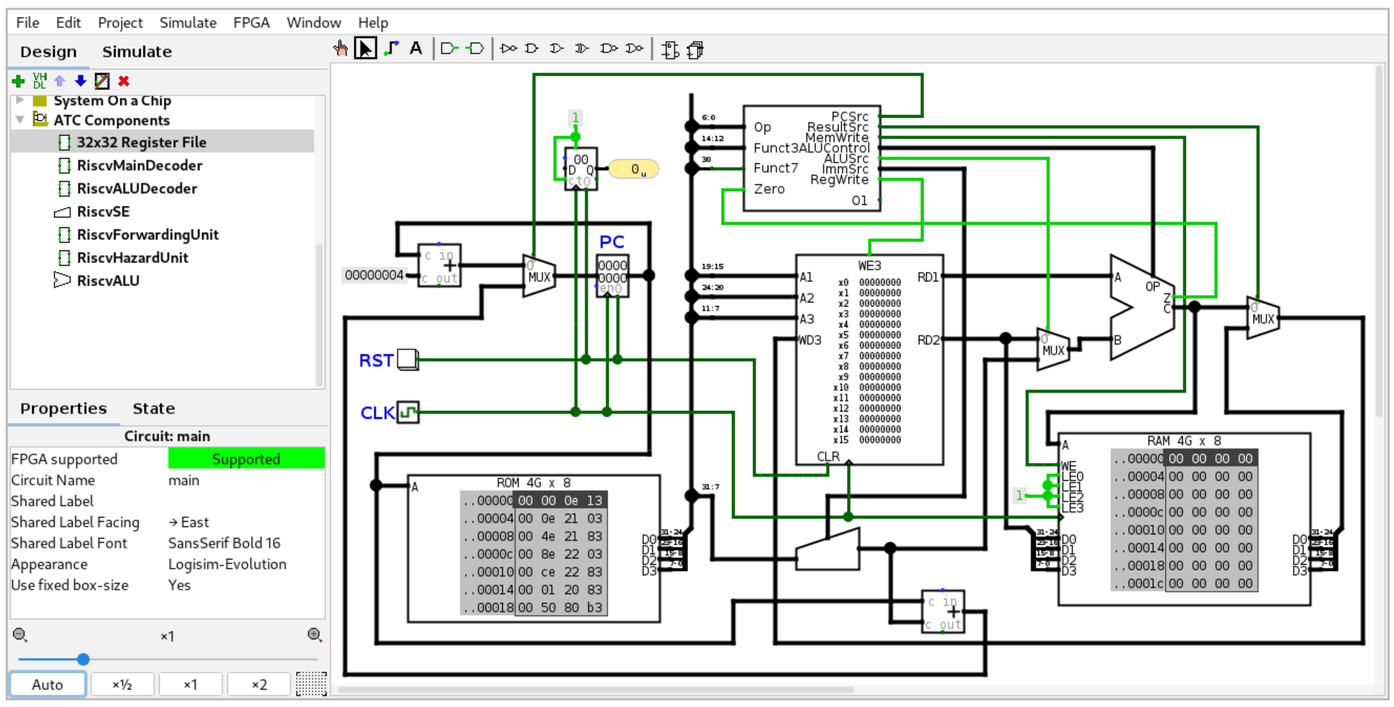
Hands-on lab assignments proposals.

- Building a **Single-cycle** RISC-V processor from scratch.
- Improving a 5-stage **Pipelined** processor with hazards.

Enhancements to Logisim Evolution

- Modified memory components for 32-bit addressing.
- Created editable Java-based components with real-time updates.
- Integrated real-time modification capabilities for students.

Single-cycle Processor



Lab Assignment 1

Instruction memory access Implement PC increment and instruction fetch.

Arithmetic functionality (ADD, SUB). Add Decoder, Register file and ALU components.

Data memory access (LW, SW) Add SE and memory. Mux to choose reg. or imm.

Branch instructions (BEQ). Implement conditional branch logic. Unconditional: BEQ x0,x0.

Extending the processor (LWAC) Instruction to load word and accumulate on register.

Lab Assignment 2

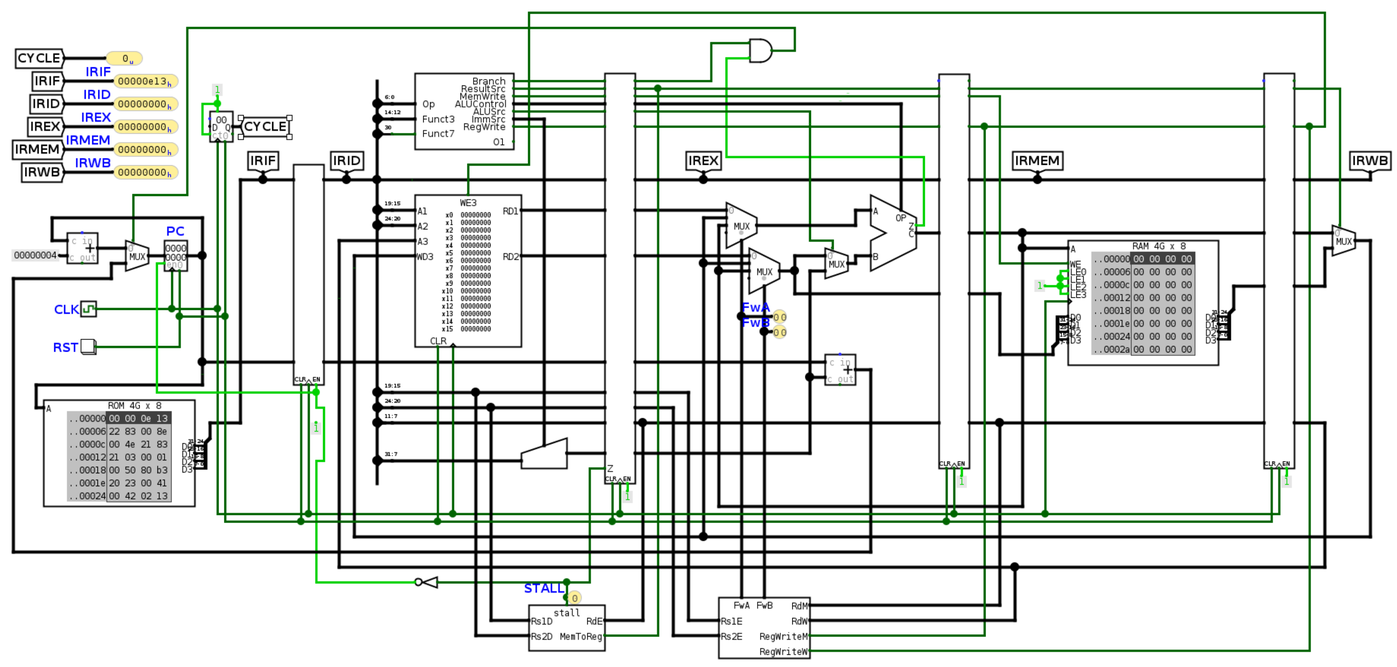
Basic pipeline Identify hazards and solve by adding NOPs to program.

Structural Hazards Move branch to ID stage and remove unnecessary NOPs.

Data Hazards Implement forwarding unit and remove NOPs.

Hazard Detection Implement stall mechanism to solve LW hazard.

Pipelined Processor



Numbers

Metric	Value
Time spent per assignment	4 hours
Total students per year	70
Students per group	20

Results and Experience

- Extensive use of Logisim Evolution with editable components for faster prototyping.
- Viewing the processor internals in action increases student engagement.
- Higher comprehension of processor control signals, data flow and hazards impact.
- Useful in exams with automatically generated auto-corrected hands-on exercises.

Acknowledgements

With support from the Spanish Science and Technology Commission under contract PID2022-136454NB-C21, the Ministerio de Ciencia e Innovación; Proyectos de Transición Ecológica y Digital 2021 under grant TED2021-131176B-I00 and the European HiPEAC Network of Excellence.