A logo for a laboratory

AI-generated content may be incorrect.A close-up of a logo

AI-generated content may be incorrect.

Customization of RISCV Code and Backend Design of a RISCV Processor Project report

**Students:**

Muhammad Biadsy

Omar Sharafy

**Supervised By:**

Amnon Stanislavsky

Table of Contents

**Type chapter title (level 1)1**

Type chapter title (level 2)2

Type chapter title (level 3)3

**Type chapter title (level 1)4**

Type chapter title (level 2)5

Type chapter title (level 3)6

# **Abstract**

Our primary objective in this project is to modify and prepare the pipelined RISC-V processor, as taught in the “Digital Systems & Computer Structure” course, for a VLSI backend flow and potential chip fabrication while improving some of its functionally and adding debug capabilities.

The processor design provided in the course was intended for simulation and educational purposes only and is not synthesizable in its original form. To prepare it for fabrication, we made several structural and functional changes.

These include replacing the non-synthesizable memory initialization method with an external memory interface, switching to area-efficient compiled memories, and later on improving the memory loading method and functionally of the module we worked on. and adding debugging capabilities that allow access to internal registers after fabrication.

\*As part of this effort, we also improved the Verilog implementation to support higher clock frequencies and better performance in the physical design flow.

On the path to achieving our goal, we modified and verified the RTL, followed by a backend flow including synthesis, scan insertion, logical equivalence checking, floor planning, power grid design, placement, clock tree synthesis, routing, and timing analysis.

# **Introduction**

## Project Requirements and Specifications

* Technology: Tower memories.
* Maximum area & power efficiency.
* optimizing chip memory.
* Adding debug capabilities for accessing internal registers post-fabrication.
* Optimizing chip Frequency.

## Alternative Solutions

The RISC-V processor we worked with can utilize FPGA-based memories and using the control unit as a debug tool to monitor internal signals. It also leverages the initial lines of code to load specific content into the memory. While this approach enables functional testing and debugging during development, it presents several limitations. FPGA memories are not efficient in terms of area and power consumption, and the Verilog code used to integrate them is not optimal.

Additionally, the original design imposes restrictions on accessible memory addresses, which limits flexibility and scalability in more advanced use cases.

## Selected Solution

Using SRAM memories (Designed by Tower), which are both area & power-efficient, allowing simultaneous read and write operations to multiple addresses, and making the chip synthesizable.

Additionally, Creating a dedicated debugging unit, separate from the control unit, to load the memories, provide additional debugging capabilities & accessing internal registers after fabrication.