Project 3: CS61CPU

Part A Deadline: Monday, July 19, 11:59:59 PM PT

Part B Deadline: Thursday, July 29, 11:59:59 PM PT

Overview

You're probably curious about that "Sea Pea You" thing in your computer. How exactly does a CPU execute those RISC-V instructions you've been writing? It's time to uncover another piece of that black box: welcome to Project 3!

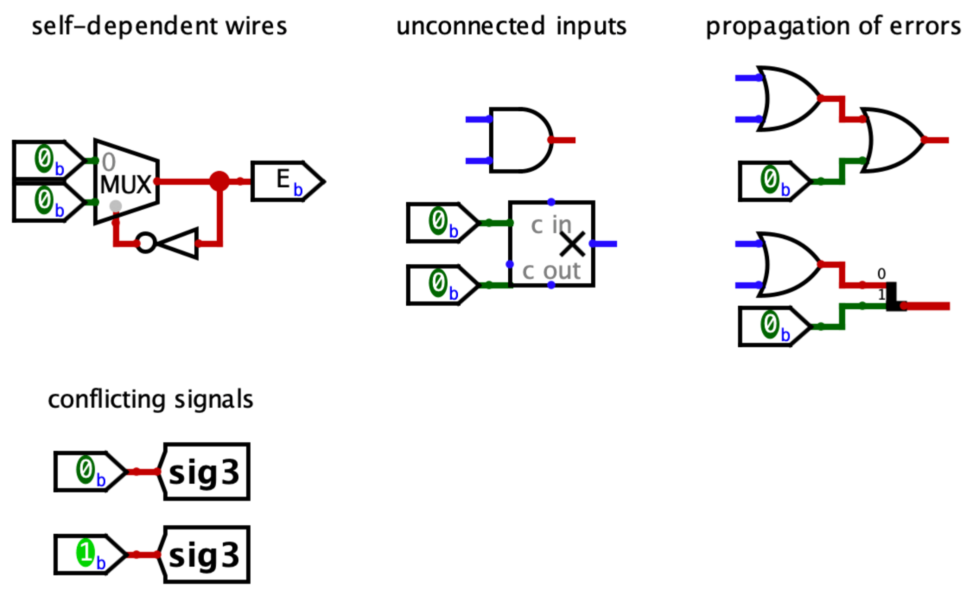
In Part A (Tasks 1-4), you'll be wiring up the ALU and RegFile for a basic RISC-V CPU, as well as implementing the CPU datapath for executing `addi` instructions. In Part B (Tasks 5-6), you'll use these components (and others) to wire up a working CPU that runs actual RISC-V instructions!

Tips and Guidelines

- You are only allowed to use Logisim's built-in components from the following libraries for all parts of this project: `Wiring` (except `Transistor`, `Transmission Gate`, `POR`, `Pull Resistor`, `Power`, `Ground`, `POR`, `Do not connect`), `Gates` (except `PLA`), `Plexers`, `Arithmetic` (except `Divider`), `Memory` (except `RAM`, `Random Generator`).
- Save frequently and commit frequently! Try to save your circuits every 5 minutes or so, and commit every time you produce a new feature, even if it is small.
- `.circ` files use the XML format, which makes it hard for Git to automerge. We recommend working on a single computer at a time; if you use multiple computers, make sure that you have pushed and pulled your code before switching devices.

- **Do not** move or edit the locked input/output pins in your circuits, since this could knock the pins out of alignment (in other words, autograder tests will fail)! Check the harness circuits, as well as `cpu.circ`, to make sure your circuits fit in the testing harnesses. You can add pins in subcircuits you create, but **avoid** making new pins in circuits with the provided, locked pins.
- You may make new subcircuits, but they must be located in the `.circ` files given in the `cpu/` folder in the starter code. You may not make new `.circ` files; the autograder will fail you if you do this!
- You must use unique names for each subcircuit across all `.circ` files. Failing to do this will result in reduced autograder points. You may not change the names of any circuits provided in the starter files.
- For Part A, the project starter code includes all tests (i.e., there are no hidden tests).
- For Part B, the project starter code includes sanity tests and those unit tests that are also visible on the Gradescope autograder.
- For Part B, the Gradescope autograder has visible unit tests, hidden unit tests, hidden integration tests, hidden edge tests, and a visible test coverage metric to help you build a comprehensive test suite.
- More information is available under the Testing sections of the spec.
- We expect you have completed Lab 5 before starting on Part A of the project and Lab 6 before starting on Part B of this project. Both labs cover many Logisim basics that will be useful for the respective parts of the project.
- A Logisim library reference can be found by going to `Help -> Library Reference` within Logisim.

Some common sources of Logisim errors, for your debugging convenience:



Getting Started

Please visit https://galloc.cs61c.org/ and get your `proj3` repository. Then, clone your repository locally and add the starter remote:

```
$ git clone YOUR_REPO_URL
$ cd YOUR_REPO_NAME
$ git remote add starter https://github.com/61c-teach/su21-proj3-
$ git pull starter main
```

If we make changes to the starter code, you can update your repository with `git pull starter main`.

Please click `Part A` on the sidebar to the left to access the spec for Part A.