Arch-2025-Lab1 实验报告

基础流水线构建与运算符指令

23307130064 周弈成

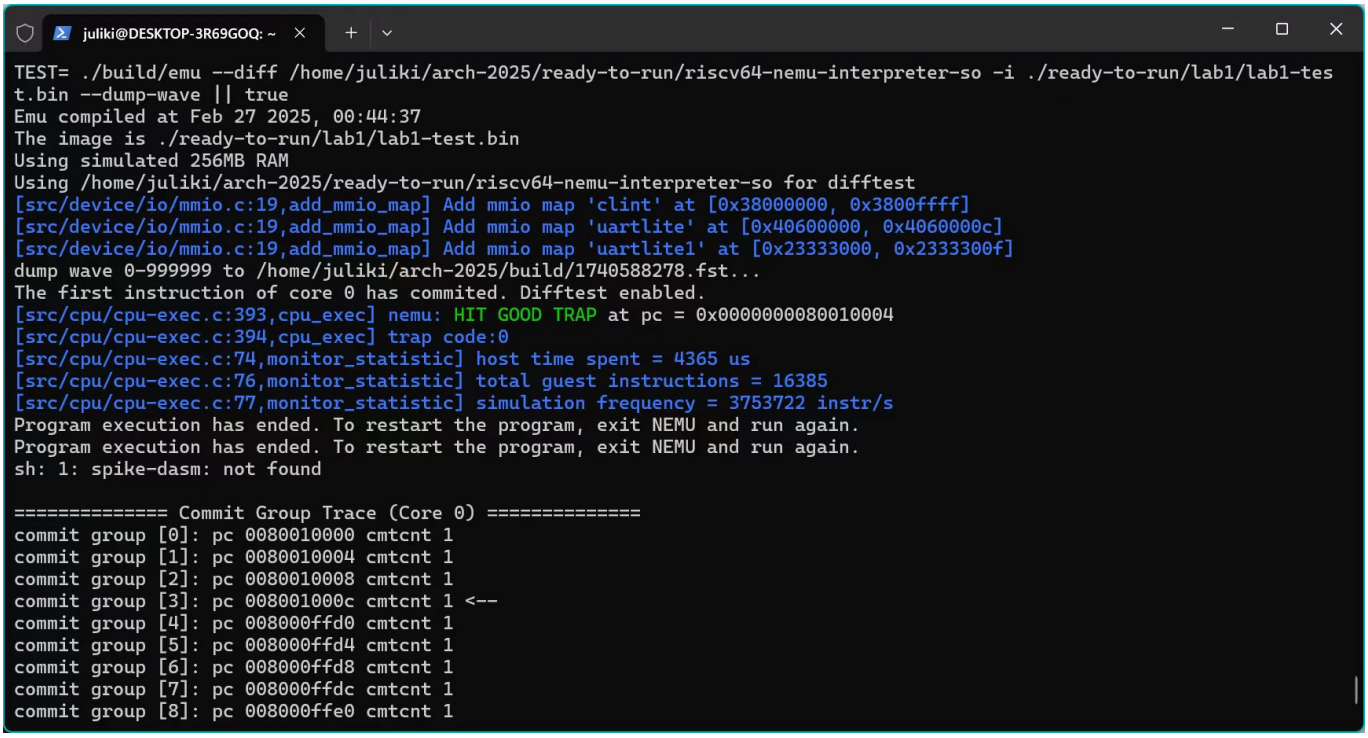
内容简介

根据五级流水线的CPU框架，构建能够实现基本运算指令的CPU。实现的运算指令有：

- 算术运算与逻辑运算：addi, xori, ori, andi, add, sub, and, or, xor
- 扩展指令：addiw, addw, subw

实验结果

能够通过测试，显示“HIT GOOD TRAP”。



文件结构

在`/vsrc`下，有`/include`、`/pipeline`、`/src`三个文件夹，其中：

- `/include`：代表头文件，其中新建的`pipes.sv`包含必要的数据类型与结构体定义；
- `/pipeline`：代表流水线执行的各个阶段模块；
- `/src`：放置调用流水线执行模块的CPU核心流程模块`core.sv`。

阶段介绍

分为取指、译码、执行、访存、写回五阶段。

取指阶段：仅把指令传入dataF，并使程序计数器pc改为原值+4。

译码阶段：将指令译码为控制信号，立即数和寄存器地址，需要把立即数符号扩展，并访问寄存器取出读入寄存器的值。

- 根据dataF中指令，按照RISC-V指令格式拆分，先判断op部分确定I/R指令以及是否为64位扩展的指令，再根据F3和F7判断运算符号。
- 所有指令均需要写回，相关控制信号设为真并记录地址；
- I指令ALU一端为立即数，R指令则为寄存器值，控制信号相应区分；
- w型运算指令需要截取并符号扩展ALU结果，需要相应控制信号；
- 运算符号也作为控制信号解析；
- I指令需要符号扩展12位立即数值，一并编写扩展模块。

执行阶段：将译码阶段获取的寄存器值放入ALU，根据控制信号确定的运算符号计算。

- ALU即根据传入控制信号运算；
- w型指令需要符号扩展32位立即数值，一并编写扩展模块。

访存阶段：目前无效果，传递执行阶段的值。

写回阶段：根据访存阶段的运算结果和控制信号传来的写回地址，写入寄存器模块。

寄存器文件需要单独编写，同时具有读取和写入功能，根据控制信号开放使能，读取功能为立即读取的组合逻辑，而写入需要等待一周期，为时序逻辑。

流程与接口设计

此次实验的流水线由于无非运算指令，不会冲突，原则简单：上一阶段的数据处理完毕后下一阶段开始工作。在数据结构体中，通过valid标记，收到上阶段数据valid信号后，下阶段工作。

取指阶段的开始需要等待外部接口传入数据，且内部指令处理完毕。

接口需要输出写回阶段的状态和写回操作。

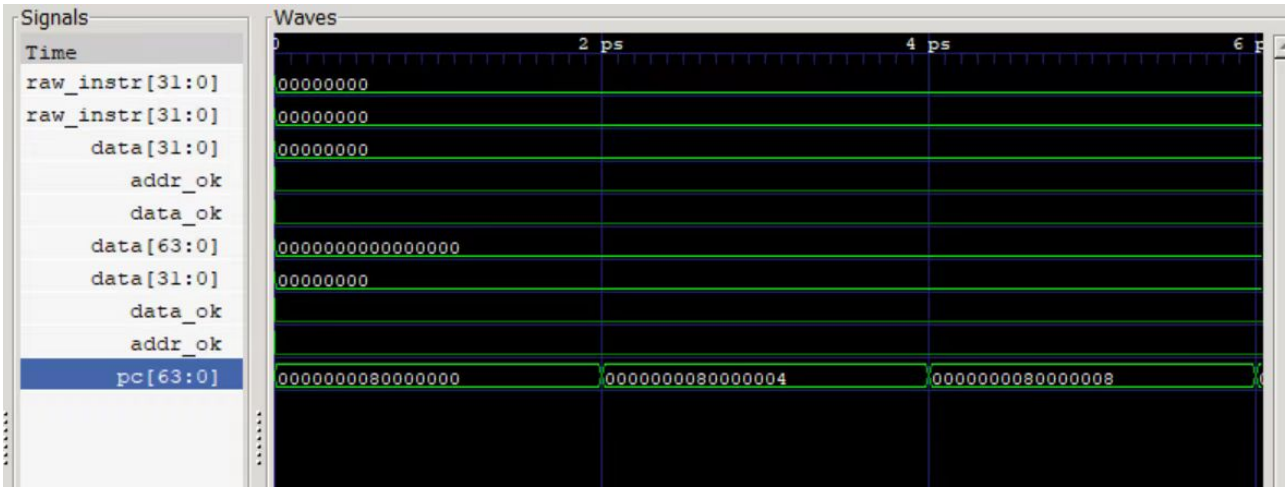
需要把实时更新的寄存器文件（下一周期）输出，因而设计组合逻辑写入临时寄存器next_reg并作为decode阶段输出，到下阶段时序逻辑同步。

实验过程

先设计模块，再设计流程和接口。

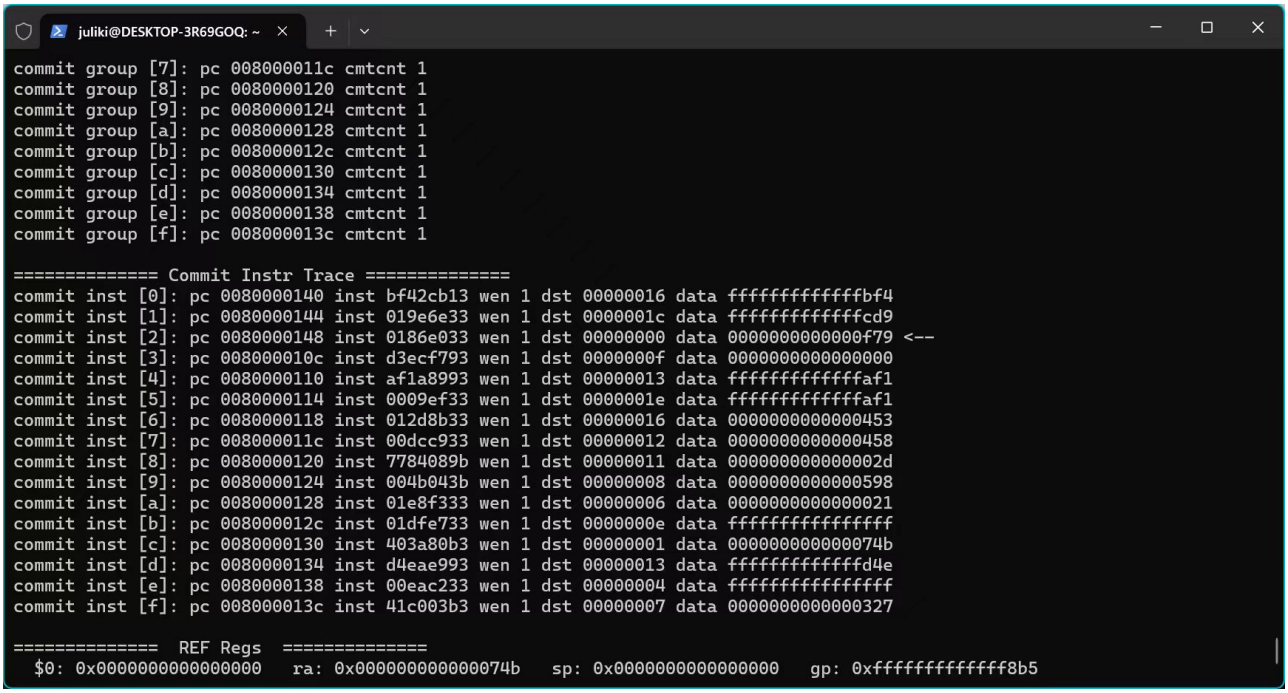
- 模块设计参照给出的MIPS32位流水线CPU模块，改建为RISC-V64位，接口几乎相同（指令格式有变化）。
- 设计后，起初将各个模块的刷新条件统一设置为外部接口传入数据，导致各模块无法工作。

- 调试时，原因排查许久，最后索性设置无提交，观察完整的pc与指令执行波形图，发现了问题。



无法正常显示指令的问题图

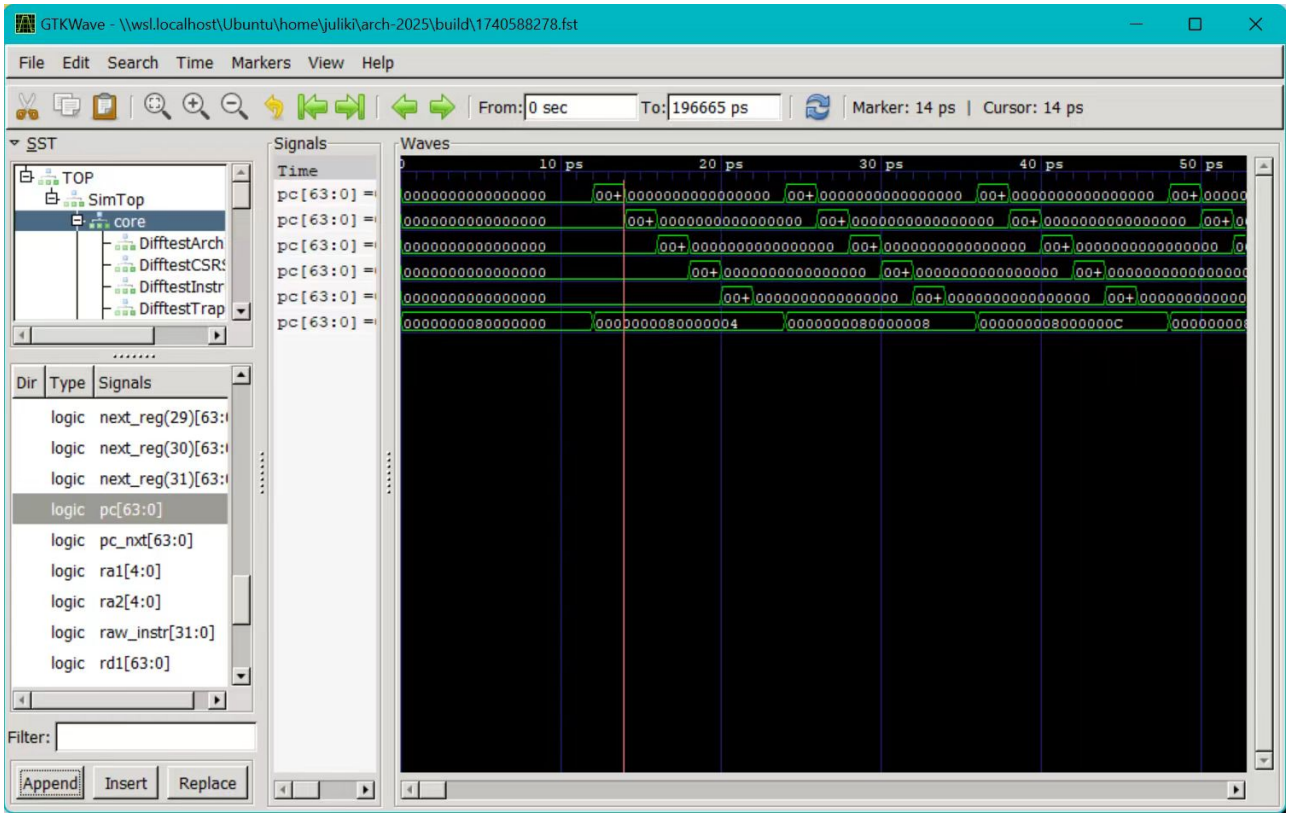
- 修复后，可以成功运行，但是出现新的问题：寄存器x0不能写入非0值。



调试写入x0的错误

- 修改写回阶段控制信号逻辑：地址为0时关闭写使能，后能正常运行。

- 波形图呈周期六段。



最终的指令波形图