**深 圳 大 学 实 验 报 告**

**课 程 名 称： 计算机系统(3)**

**实验项目名称： 取指和指令译码设计**

**学 院： 计算机与软件学院**

**专 业： 计算机与软件学院所有专业**

**指 导 教 师： 刘 刚**

**报告人： 马咏韬 学号： 2021152024 班级： 计科01**

**实 验 时 间： 2023/10/30-2023/11/13**

**实验报告提交时间： 2023/11/13**

**教务处制**

**一、 实验目标：**

设计完成一个连续取指令并进行指令译码的电路，从而掌握设计简单数据通路的基本方法。

**二、实验内容**

本实验分成三周（三次）完成：1）首先完成一个译码器（30分）；2）接着实现一个寄存器文件（30分）；3）最后添加指令存储器和地址部件等将这些部件组合成一个数据通路原型（40分）。

**三、实验环境**

硬件：桌面PC

软件：Linux Chisel开发环境

**四、****实验步骤及说明**

本次试验分为三个部分：

1. 设计译码电路，输入位32bit的一个机器字，按照课本MIPS 指令格式，完成add、sub、lw、sw指令译码，其他指令一律译码成nop指令。输入信号名为Instr\_word，对上述四条指令义译码输出信号名为add\_op、sub\_op、lw\_op和sw\_op，其余指令一律译码为nop，输出信号均为1bit。

给出Chisel设计代码和仿真测试波形，观察输入Instr\_word为add R1,R2,R3; sub R0,R5,R6，lw R5,100(R2), sw R5,104(R2)、JAL 100时，对应的输出波形。

Instr\_decoder

Instr\_word[31:0]

add\_op

sub\_op

lw\_op

sw\_op

nop

1. 设计寄存器文件，共32个32bit寄存器，允许两读一写，且0号寄存器固定读出位0。五个输入信号为RS1、RS2、WB\_data、Reg\_WB、RF\_WrEn，寄存器输出RS1\_out和RS2\_out；寄存器内部保存的初始数值等同于寄存器编号。

给出Chisel设计代码和仿真测试波形，观察RS1=5，RS2=8，WB\_data=0x1234，Reg\_WB=1，RF\_WrEn=1的输出波形和受影响寄存器的值。

Reg\_file

RS1[4:0]

RS2[4:0]

WB\_data[31:0]

Reg\_WB[4:0]

RS1\_out[31:0]

RS2\_out[31:0]

RF\_WrEn

1. 实现一个32个字的指令存储器，从0地址分别存储4条指令add R1,R2,R3; sub R0,R5,R6，lw R5,100(R2), sw R5,104(R2)。然后组合指令存储器、寄存器文件、译码电路，并结合PC更新电路（PC初值为0），最终让电路能逐条指令取出、译码（不需要完成指令执行）。

给出Chisel设计代码和仿真测试波形，观察四条指令的执行过程波形，记录并解释其含义。

PC

（初值为0）

+4

指令内存

Instr\_decoder

Reg\_file

RS1[4:0]

RS2[4:0]

WB\_data[31:0]

RS1\_out[31:0]

RS2\_out[31:0]

add\_op

sub\_op

lw\_op

sw\_op

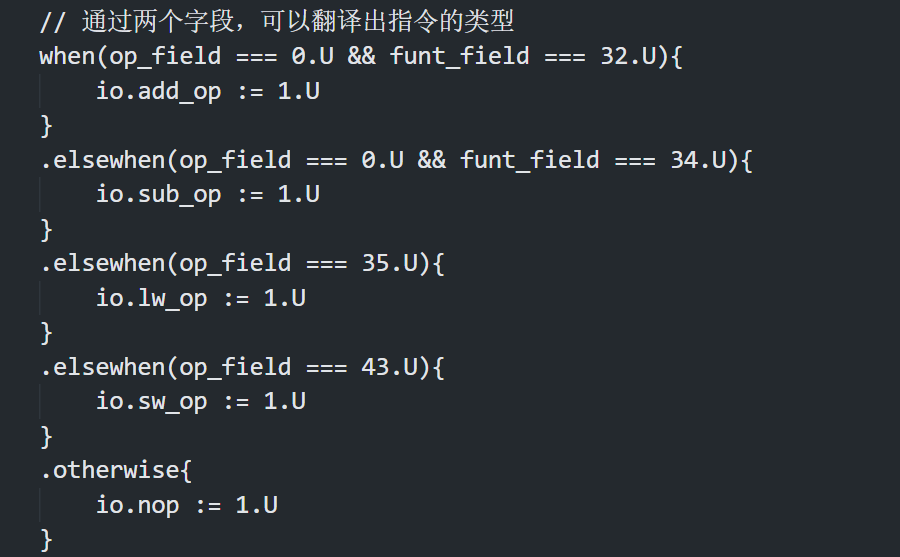
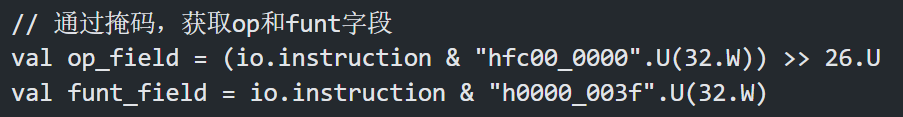
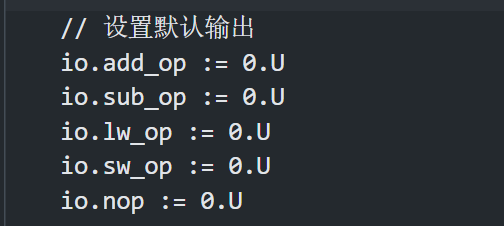
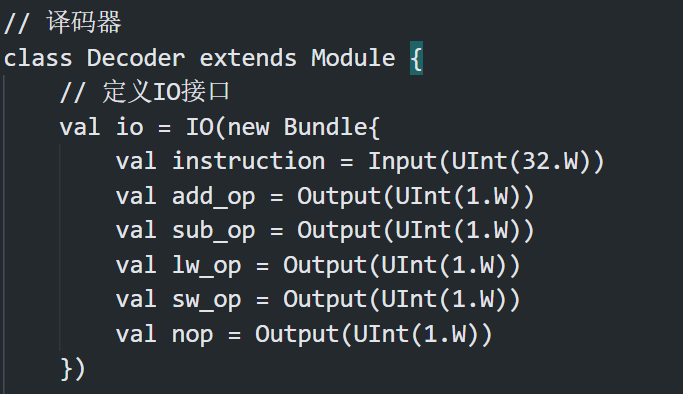
nop

Reg\_WB[4:0]

RF\_WrEn

**五、实验结果**

**1、编写译码器**



以上代码即可完成译码器。

接下来测试译码器的代码。首先将汇编指令翻译成机器码：

Add r1, r2, r3 => 0043 0820

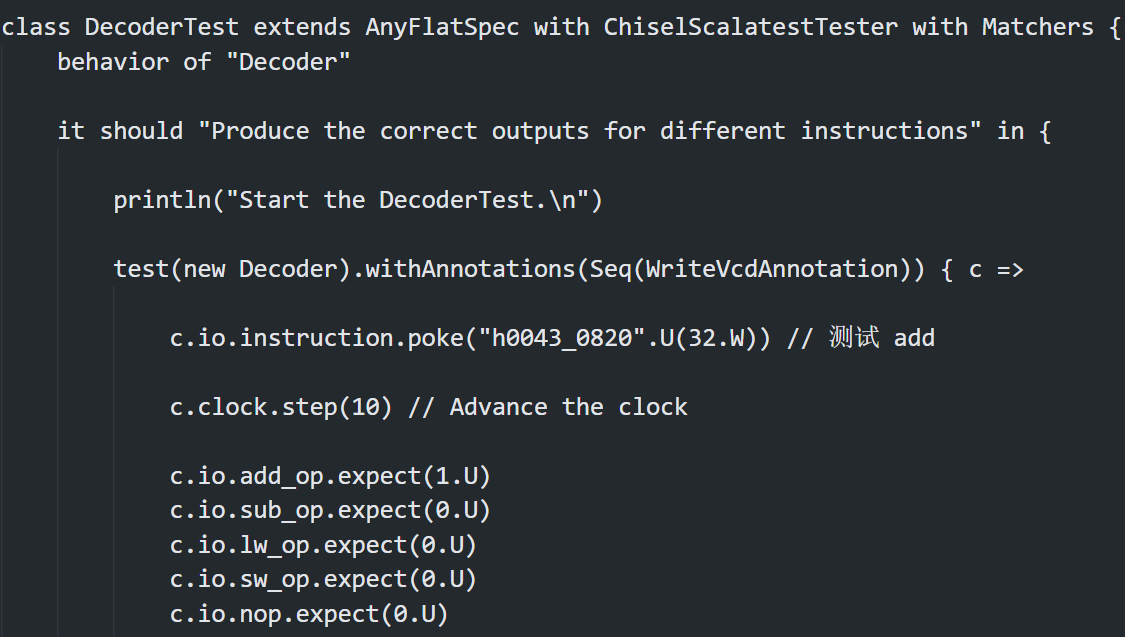
Sub r0, r5, r6 => 00A6 0022

Lw r5, 100(r2) => 8C45 0064

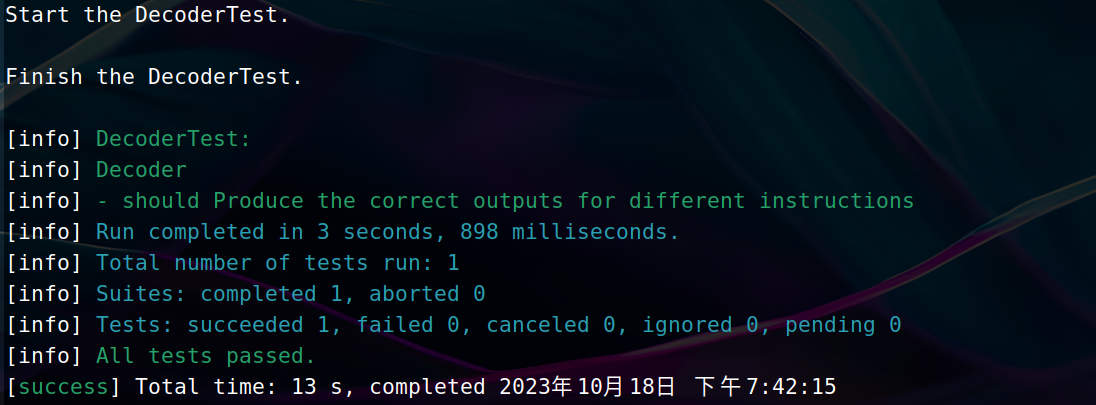
Sw r5, 104(r2) => AC45 0068

Jar ra, 100(r2) => 0C5F 0064

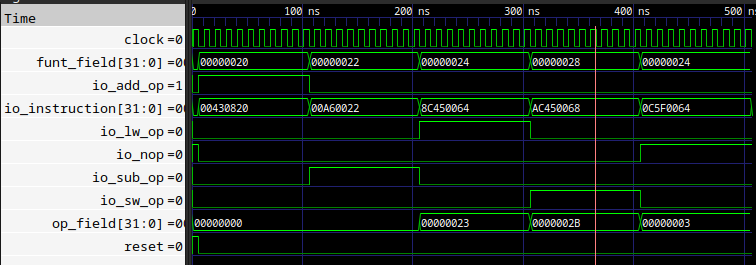
使用chiseltest编写测试函数。



测试结果：



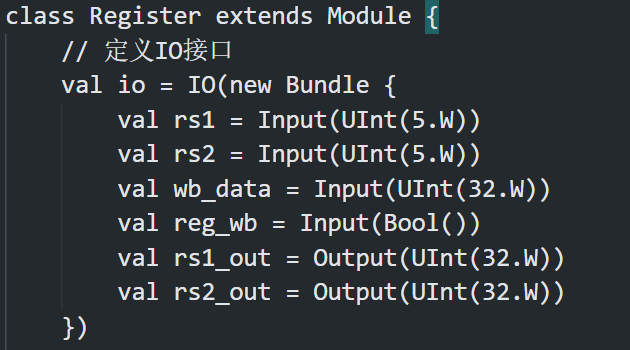
查看仿真波形图：



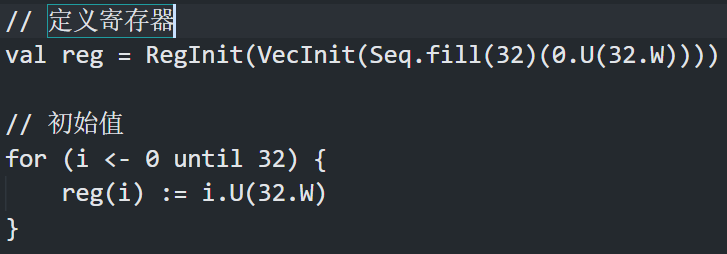
波形分为5个时间段，输出为1的接口分别为add\_op, sub\_op, lw\_op, sw\_op, nop,与预测结果一致

**2、编写寄存器文件**

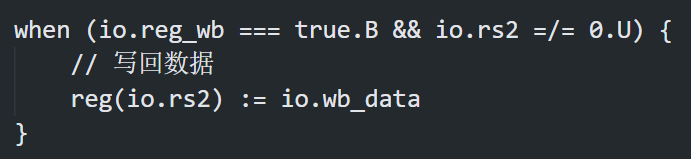
定义寄存器类以及IO接口



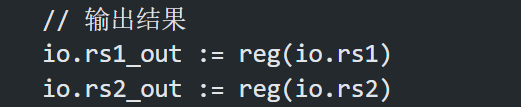
定义寄存器，并初始化



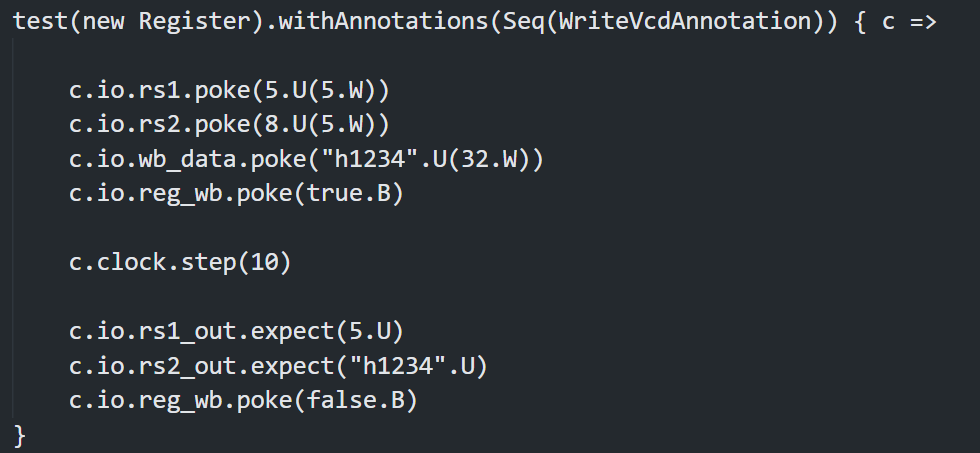
寄存器首先会判断是否有写寄存器的操作



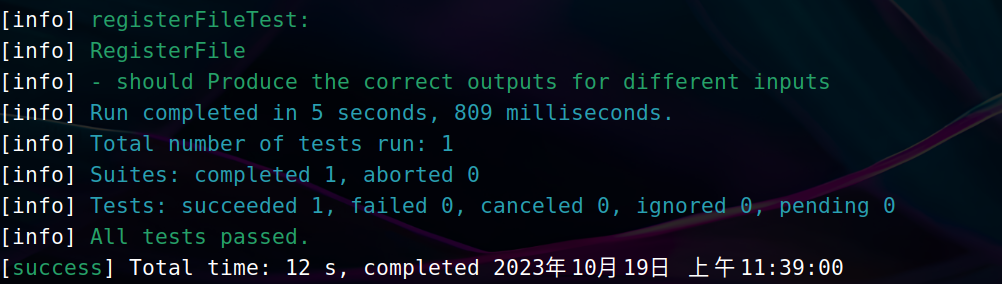
最后通过两个输出端口输出读取结果



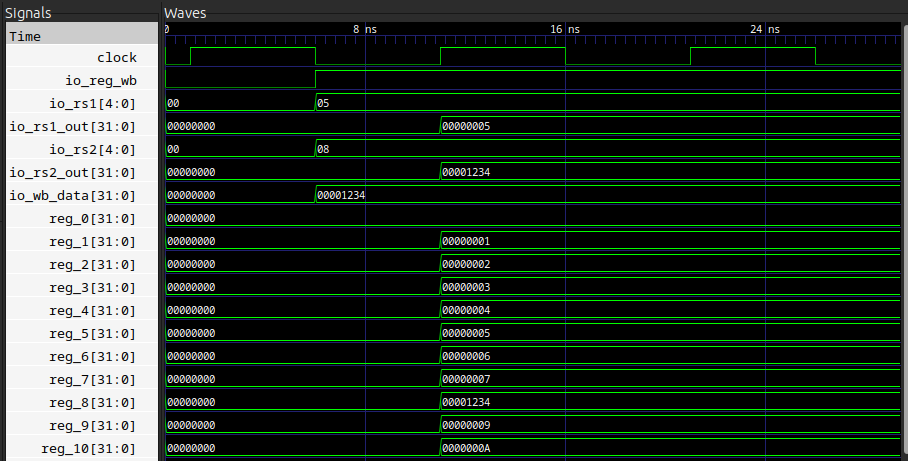
接下来编写测试文件



运行测试，测试结果：



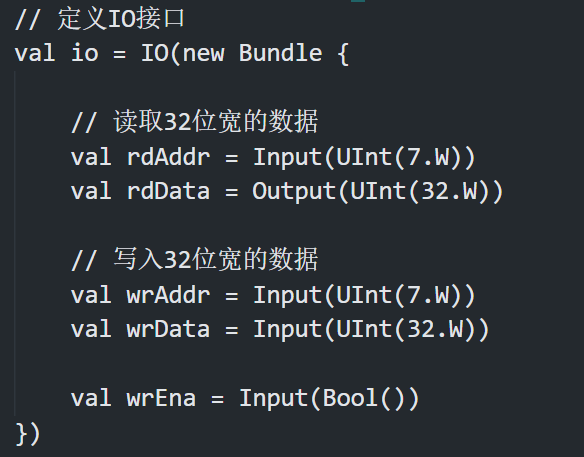
查看仿真波形图



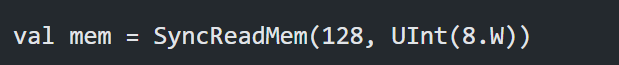
最开始时， reg\_wb赋值为1，寄存器会从wb\_data读取数据，并将其写入rs2指向的寄存器，即r8，可以看到，第二个时钟时， r8从0x8变成了0x1234。rs1\_out输出为0x05，rs2\_out输出为0x1234

**3、编写指令存储器**

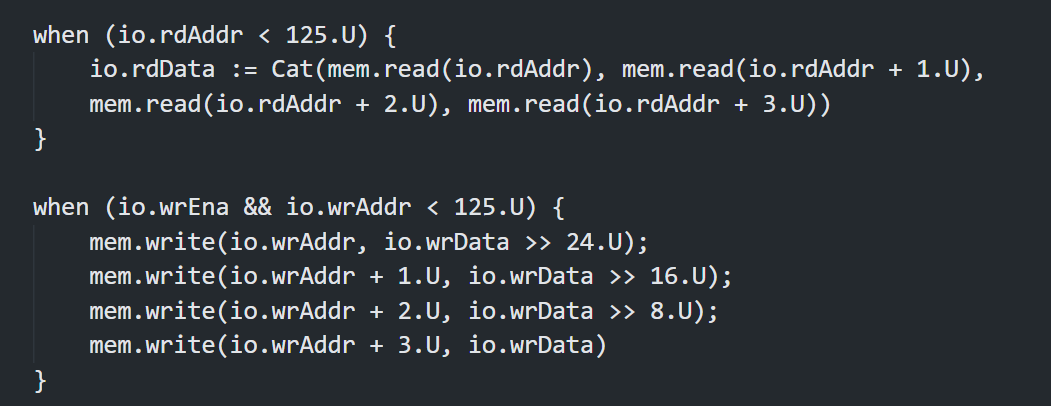
首先定义IO接口，存储器共有128个字节，因此需要7位宽的地址。



定义一块内存区域，作为存储

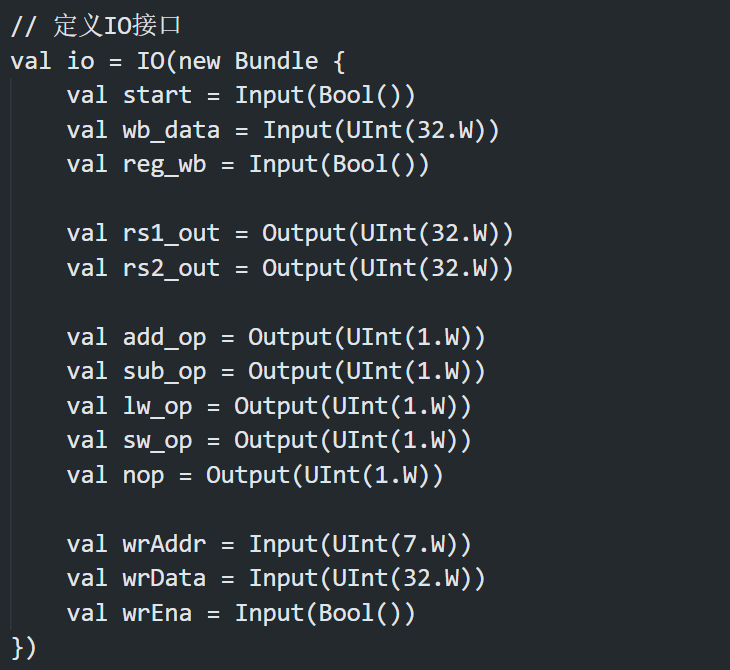


实现存储器的读取和写入功能

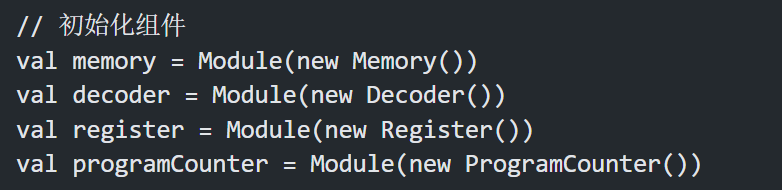


**4、完成组合电路**

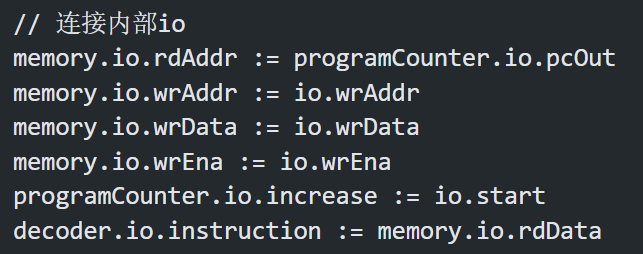
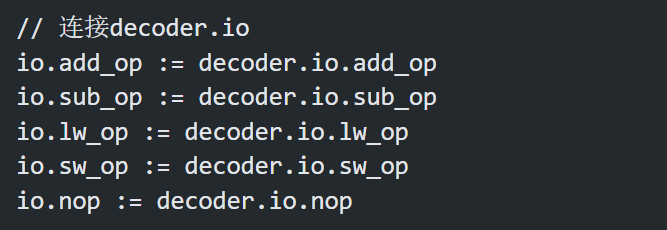
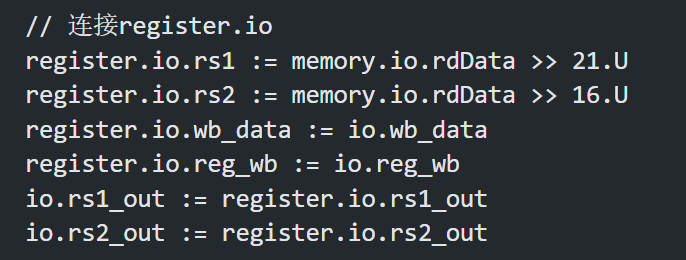
将以上三个类，与程序计数器组合在一起，完成组合电路，首先定义IO接口



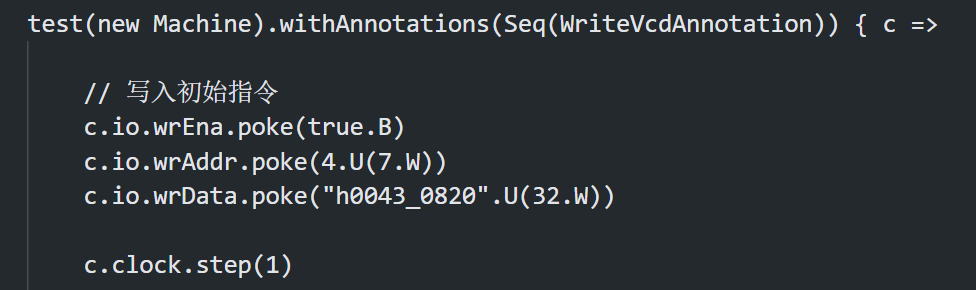
然后在类中初始化其余组件



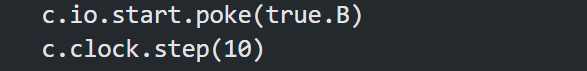
将所有组件按照逻辑连接在一起



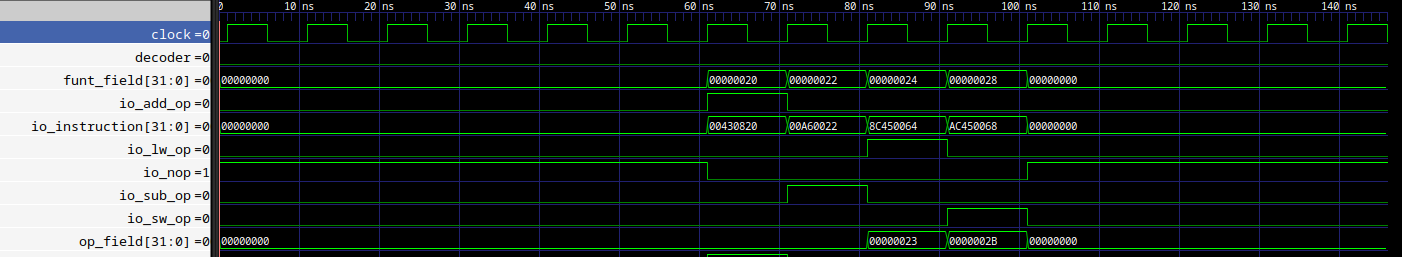
在测试程序中，通过io接口向存储器中写入指令



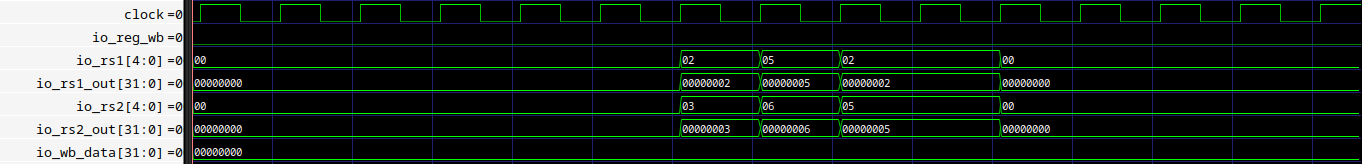
然后通过设置程序计数器的启动标志，来启动程序计数器的递增过程



查看仿真波形图：



其中add\_op, sub\_op, lw\_op, sw\_op，分别在第7、8、9、10个时钟周期为1，与测试程序预想的一致（前五个周期在写入指令到存储器，第六个周期在读取第一条指令0x00）



查看寄存器文件的输出波形，可以看到寄存器正确地读取内容并输出

**五、实验总结与体会**

通过本次实验，完成译码器、寄存器以及一个指令存储器的电路实现，并且将其组合成为连续取指令并进行指令译码的电路，掌握了设计简单数据通路的基本方法

|  |
| --- |
| **指导教师批阅意见：**  **成绩评定：**  指导教师签字： 刘刚  年 月 日 |
| 备注： |