实验：单周期CPU lab2

实验人：谢志康

学号：22307110187

日期：2024.3.19

代码架构建立在lab1的基础上，仅新增了Fetch模块，并且修改了部分Control.sv和MEM.sv内容，其它部分基本不变，并将整个代码框架与测试环境相接来测试CPU运行状态。每个模块的接口功能以及CPU总流程图均与lab1所交实验报告中介绍相同。

解决内存访问冲突的总思路：利用stall, stall\_next\_ibus, stall\_this\_dbus三个使能来控制PC。

**stall\_next\_ibus：当前指令执行阶段，判断下一条指令是否已经从内存中读取出来。**

**stall\_this\_dbus：当前指令执行阶段，判断访存操作是否已经完成。**

**stall = stall\_next\_ibus || stall\_this\_dbus**

由于访存操作（主要是MEM、FETCH中涉及）需要多个周期，因此依靠这两个stall作为使能来控制PC，只要有一个还没结束就把PC按在原地空转，没法把下一条指令取出来。

简述每个具有stall的模块的执行逻辑——

**Fetch.sv**：取指，并依靠这个模块更新ibus总线的iresp、ireq信息。

NPC模块计算完后给出的下个pc值还是直接赋给ireq地址就行，若被stall，PC会空转，NPC不会继续往前更新。

若当前指令有访存操作，且未完成，则stall\_this\_dbus会被设为1，ireq.valid应置为无效，vice versa。（每个CPU周期从取指开始，把指令取出来之后经过一系列组合逻辑去译码，译码后如果发现有访存操作的话就需要把取指的ireq.valid先变成0，按住不让直接取下一条指令，然后等访存结束之后再重新把ireq.valid置成1，等待下一条指令被取出，开始下一个CPU周期）

指令响应中，若data或addr任意一个没有准备好，都不应该访存取出下一个指令，否则会导致冲突，依靠stall\_next\_ibus来阻塞，将其置1，不让取下一条指令。

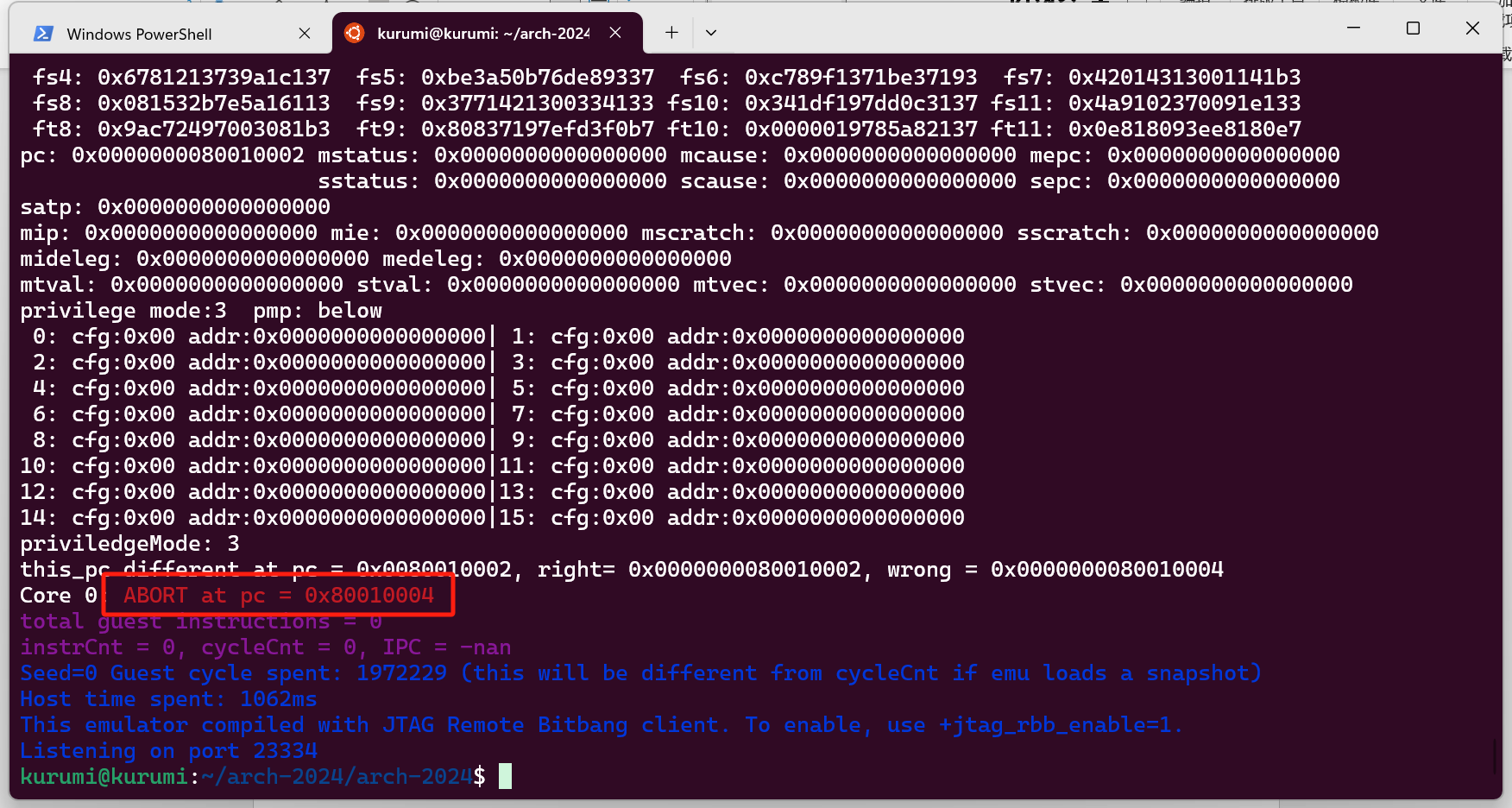
**PC.sv**：当CPU运行时若stall为1，则表示之前的操作还没有执行完，这个时候PC不能再往后更新了，则在PC模块中设置让PC空转一段时间（PC<=PC），直到stall为0，也即之前的操作全部做完，并且在时钟上升沿时，再将PC更新为NPC模块给出的下一个值。

（补充说一下：lab1交上去的代码sext中u型指令扩展错了不好意思……已经改正）

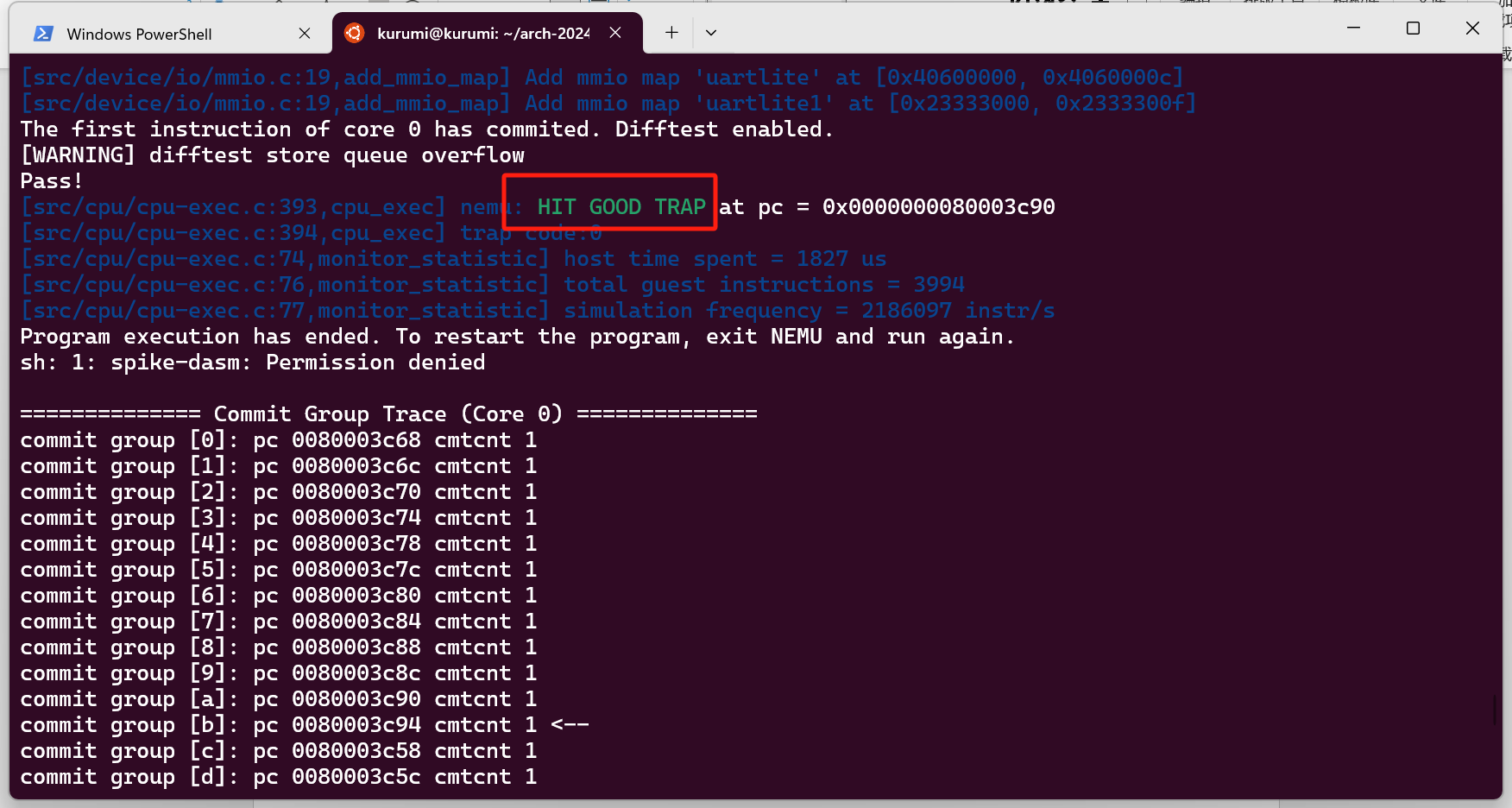
**RegFile.sv**：传入stall信号，起阻塞作用，若stall为1，则不对Reg执行写入操作，保持regfile的值不改变。

**MEM.sv**：总线dreq的data和addr直接写入更新即可。若当前CPU周期解析的指令有访存操作，则在control模块中dbus\_wre会被置为1表示读写内存使能，同时，当这个访存操作未完成时，dreq.valid持续拉高，stall\_this\_dbus被置为1，表示当前指令执行阶段访存操作还未完成，CPU应该空转一段时间等它完成。

Test-lab1运行截图：



Test-lab2运行截图：



实验课笔记——

1. **总线：**

BUS： AXI4——CBUS——IBUS & DBUS

CPU不包括RAM——如何与CPU外部的RAM通信——协议——一组信号

主(CPU)——>从(RAM) （request）

主(CPU)<——从(RAM) （respond）（有延迟，需要多个周期返回）

Req需要有：valid、addr

Resp需要有：addr\_ok、data\_ok、data

1. **CPU有哪些划分：**
   1. Datapath + control （只适合较简单CPU）
   2. IF + ID + EX + MEM + WB （最常见划分）
   3. Frontend (IF\_unit) + backend (ID\_unit+rr\_unit+EX\_unit)
2. **Difftest如何接入：**

在指令提交的时候（valid拉高的时候），其产生的影响恰好生效（不早不晚）

接到RegFile模块，提交寄存器内容做差分测试。

General problems:

1. 5000cyc未提交：若第一条未提交，检查是否有指令、PC是否是0x8000\_0000。若第一条已提交，则是某条指令卡住了。
2. InstrCommit和IntRegState时序对不上：可能是提交时数据还未写入。
3. Different at this\_pc = xxx……：自己代码有问题 ^\_\_^
4. **Example：**

