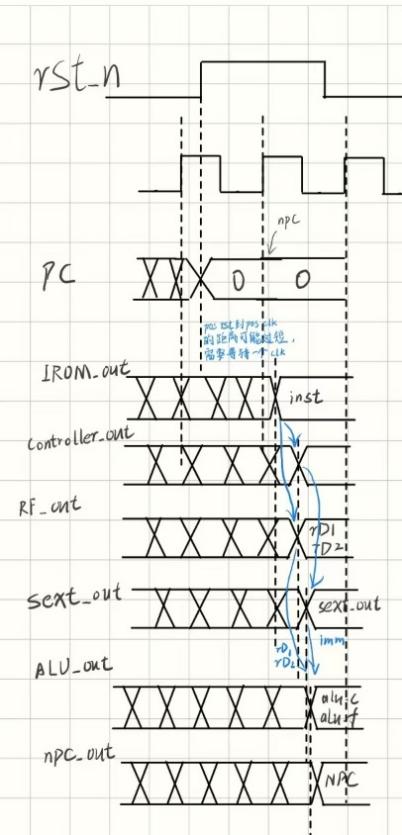
|  |  |  |  |
| --- | --- | --- | --- |
| 姓名 | XXX | 指导老师 | 陈翔 |
| 项目名称 | RISC-V指令集实现 | | |
| 时间 | 11.8——11.15 | 工作量 | 7天 |
| **任务情况** | **1.通读了mini-RISCV 的CPU核代码，完成了时序分析、功能分析**  **2.阅读文献《RISC-V处理器设计》** | | |

* **工作具体情况：**

1.通读了mini-RISCV CPU核心的代码，完成了以下手绘时序分析：

****

**Rs之后的时序图**

如图，在rst后，PC归零。由于rst信号距离下一个clk信号的时间间隔不确定，不能够保证留给NCP以及前置部件运算出正确的NCP，因此要等待一个间隔，保证至少一个Tclk间隔的NCP的计算时间。

接着分析各个部件正确结果的时间延时。为了保证一般性，图中从clk信号上升沿开始分析，之后的PC同理。首先是IROM取信号延时，用于将PC值转化为对应的指令（inst）。这是一级延时。

然后从inst到controller正确计算出控制信号的延时。由于寄存器组的读取信号rD1和rD2直接依赖于inst，因此不需要等待controller结果，直接从inst算出开始。这两个合为一种延时，称为二级延时。

接着，sext根据controller的运算指示计算出正确的指令立即数的符号扩展，称为三级延时。

接着，ALU根据sex的输出、controller的运算类型指示、RF的读输出，计算出对应的结果。主要是两种，一种是alu\_c，32位输出结果；一种是alu\_f，是条件分支语句的跳转结果。该组合逻辑计算过程需要一定的时间，称为四级延时。

最后，NPC根据controller操作码、ALU结果计算出正确的下一个PC值。所需要的时间称为五级延时。

Mini RISCV内限制CPU的运算的主频的主要就是这五级延时。

**改进方法：**

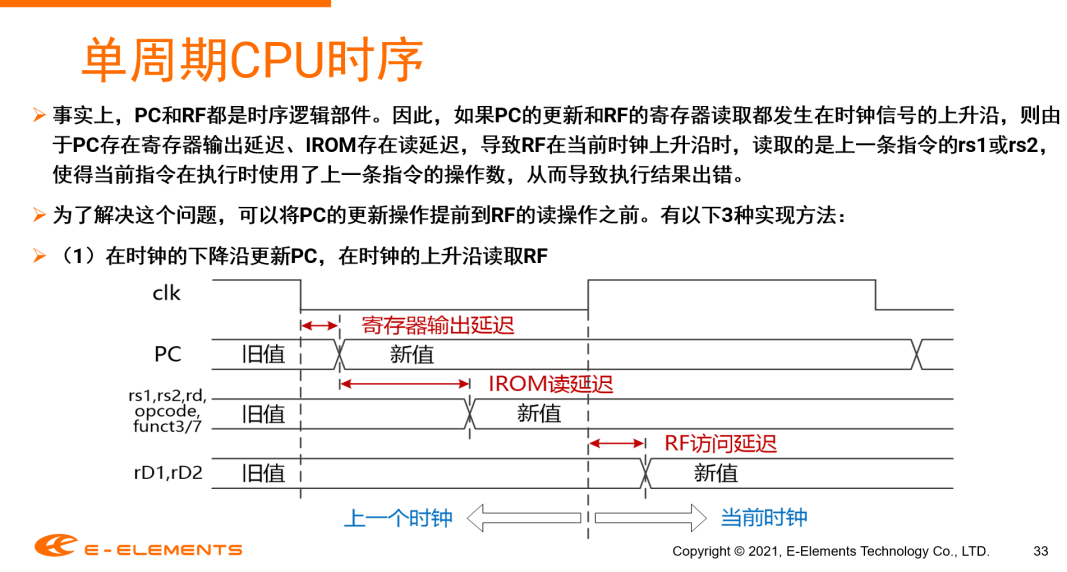
使用流水线设计，分割运算，使延时级数变小。至于其他的算法，比如分支预测，对现在的我有点困难，之后考虑。

使用分支预测加流水线冲刷技术。相当于NPC不需要等待ALU运算结果，从而省去第五级延时。在文献《RISC-V处理器》这本书里，分支预测直接使用的是静态预测，非常简单。我觉得这个也可以是一个改进点。而且实现性较高，只需要改进mini-RISCV里NPC，增加静态分支预测功能，然后再增加一个交付模块，用于判断预测是否正确就可以了。

除了改进主频，还可以加入其他的指令集。在《RISC-V处理器设计》这本书里，有M指令集（也就是乘除指令集）和A指令集（原子操作指令集）其他指令集。可以依照这本书和开源代码尝试加入其他指令集，实现mini-RISCV的改进。

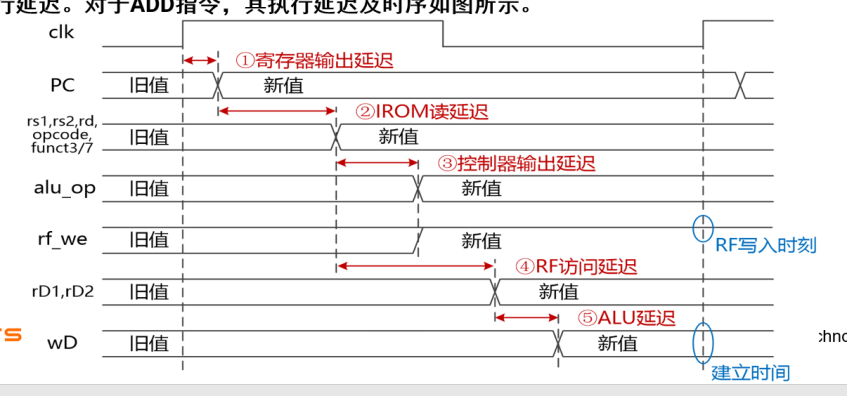
**错误指正：**

Mini-RISCV的课件中有关RF的解释存在错误。首先是现有RF读操作是组合逻辑，而不是时序逻辑。只有写操作是时序逻辑。

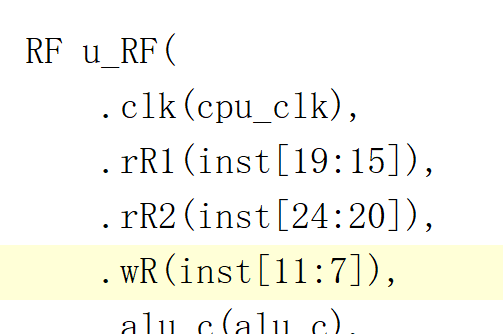


ppt中的说明

其次RF的读取直接依赖inst的内容，属于二级延时，而不依赖controller。只有写操作的操作码依赖controller：



ppt中的错误时序



实际代码

**2.文献阅读：**

阅读了《RISC-V 处理器设计》的ALU、交付和写回部分。然后发现这个处理器除了写回操作也是单级CPU。因此没有任何参考的情况下独立设计一个多级流水线的CPU好像有点困难。要么就找一下其他教材。这个还没有想好。

**下周计划：**

查阅其他文献，了解多级流水线结构和设计步骤，并查阅M指令集。初步尝试改进miniRISC-V CPU。