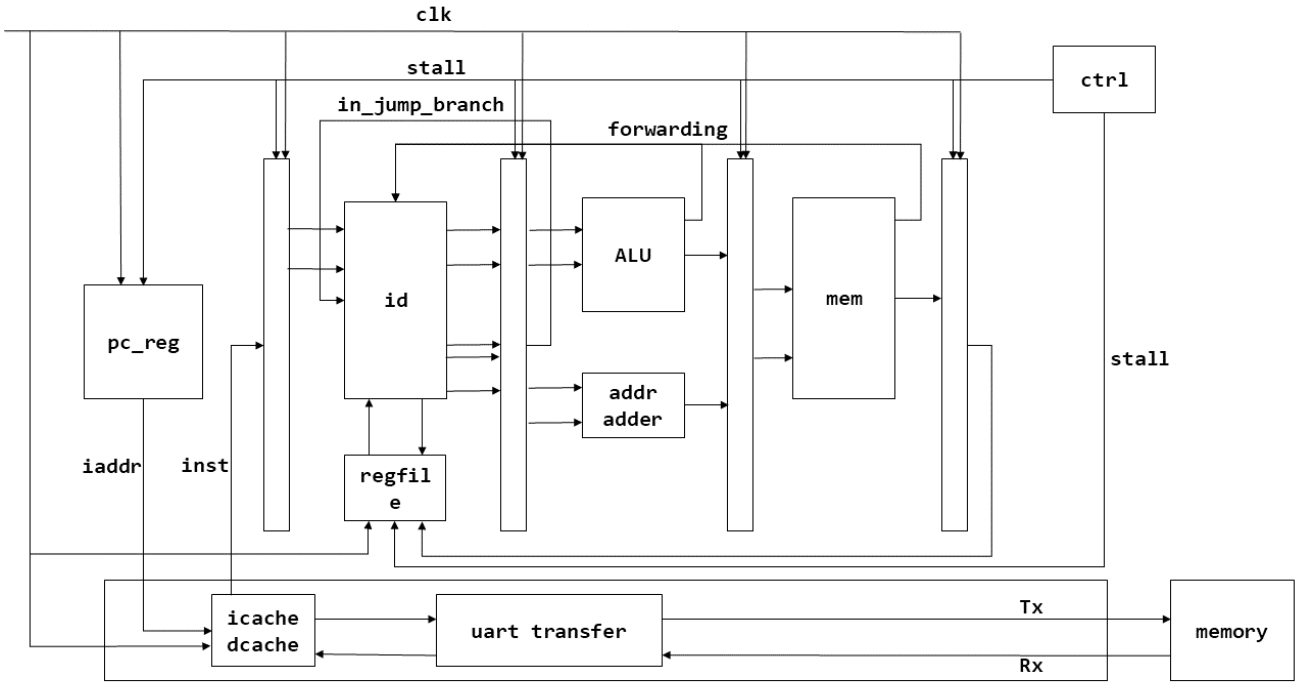
体系结构大作业报告

学号：516030910572

姓名：丁尧尧

邮箱：idy002@163.com

1. 数据流图



基本介绍：

实现的指令集是RISC-V的32位处理器，算数指令目前只支持整数相关的（即只实现了RISC-V的最基本的子集），使用冯洛伊曼结构（支持SMC），框架是五级流水，支持数据前推，用uart串口协议与内存通信。（最开始设计时考虑加icache和dcache，但最后想让CPU和计算机通信，就搁置了）

2. 创新之处

- 因为RISC-V的branch指令比MIPS要复杂一些（支持两个寄存器的值得比较，然后跳转），所以不能在id阶段就直接计算比较结果和PC的目的地址（否则的话五级流水将会极为不平衡，导致ALU模块失去意义）。我的解决方法是在执行阶段新增一个地址加法器，在解码阶段得到需要比较的寄存器x_1, x_2和当前指令地址pc及地址偏移量$offset$，然后在执行阶段同时计算比较结果和目的地址，这个时候将pc_reg模块和if_id模块暂停住，并且pc_reg模块和if_id模块监控执行阶段的结果，如果是跳转指令则pc_reg得到目的地址，if_id清空，否则pc_reg和if_id不变。最后发现所有和地址相关的指令都可以用这一机制很漂亮的解决，所以实现的所有和内存地址相关的指令都用到了地址加法器这个小模块。

  总的效果就是，如果指令会跳转，则会有两拍暂停，如果指令不会跳转，只会有一拍必要的暂停，换来的结果是五级流水的平衡性。

- 在板子中自己实现了一个内存，内存和CPU通过uart串口协议进行通信，这一个通信的底层模块uart_comm是使用助教的代码，我上面自己写了CPU和内存的交互接口，每次通信，CPU会先发一个$head[39:0]$给内存(当然是拆成一个字节一个字节)，$head[39:36]$表示读还是写，$head[35:32]$存储$mask$，$head[31:0]$为访问的内存地址$addr$，如果是写内存，则再发一个word给内存，否则就等待内存给CPU发一个word，每次通信都是以字为单位。实现的时候发现一个小技巧，让访存操作在CPU其他模块看来仅仅是一个组合逻辑，每次必定可以一周期得到结果，而真真实现的是等待很多周期与内存通信。

3.遇到的困难

按照时间顺序列一下遇到的困难及解决措施：

- 最开始实现的时候发现RISC-V指令集的branch指令比MIPS复杂一些，不能参考书，就自己想了上面的结构解决这个问题。
- 在模拟完所有指令之后，发现我的解码阶段的代码不能合成，向刘啸远请教后知道了always @ (*)的奥妙，首先是不能在两个always中对同一个变量赋值，其次是always的依赖关系不能有环（即使逻辑上没有环），最后发现所有逻辑没有环的组合逻辑都可以通过增加一些变量和always块加以解决。
- 写cache和通信模块的时候，一开始不是如何下手，想了很久才慢慢理清这种需要多个周期协调完成的任务的写法，首先本质上这种模块都是一个自动机，需要用寄存器保存当前模块任务的完成情况，然后每个周期根据当前的状态以及当前的输入决定下一个状态，用组合逻辑来实现各种操作及计算下一状态，用时序逻辑来完成状态的转移。
- 准备烧板子的时候，装各种库及配置环境也花了不少时间，其中吴章昊给了我很多帮助。然后发现实现的时候会报错，寻找解决方案良久依旧无解，用了clk_wiz并且加了助教代码中一段神奇的代码就好了。
- 最后模拟完hello world的时候，发现和计算机交互并不能很好工作，不得已又写了一个内存的模拟模块（因为格式和助教不一样，不得已自己写，我也很绝望），然后在linux下调试成功，搬到windows下发现居然不对，究竟调试，发现是for循环的表现行为有些不同，至今不解，改了后在windows下也可以成功模拟。
- 最后是uart_comm协议用的是助教的，然后助教使用了偶校验，我的adpter.hpp里没有设置对，多亏了刘天怡同学的提醒，才发现这个问题（一开始至关注了波特率，所以没注意到这个不同）。
- 最最后，发现模拟正确，但是烧到板子上就不正常工作了，改了波特率，改了各种warning后还是不行，时间已是凌晨3:30，弃疗睡觉去了。

4.参考资料

- 《自己动手写CPU》 - 雷思磊
- riscv-spec-v2.2 （https://riscv.org/specifications/）。
- 张哲凯助教代码（特别是cpu_sim.v,uart_comm.v和fifo.v）（https://github.com/sxtyzhangzk/mips-cpu）。