CPU with Five-Stage MIPS Pipeline

Bowen Tan

2017 年 6 月 25 日

目录

1 大作业要求

在模拟环境 (ModelSim) 下实现 mips 五级流水。

1.1 要求实现的指令集

add, addi, sub, and, andi, or, ori, xor, xori, slt, slti,
beq, bne, j, jr
lb, lui, lw, sb, sw

2 代码实现

实现五级流水 cpu 的过程主要参考《自己动手写 CPU》[1]。

2.1 实现功能

2.1.1 指令集

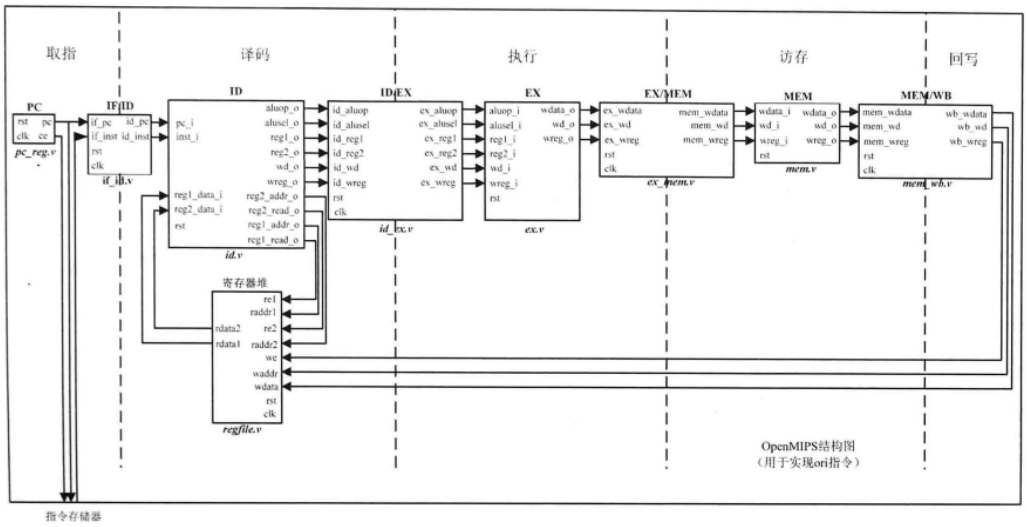
在实现的过程中，实际实现了下列指令：
and, or, xor, nor, andi, ori, xori, lui, sll, sllv, sra, srav, srl, srlv,
movn, movz. mfhi, mthi, mflo, mtlo,
add, addu, sub, subu, slt, sltu, addi, addiu,
slti, sltiu, clo, clz,
multu, mult, mul,
jr, jalr, j, jal,
j, jr, beq, bne,
lb, lw, sb, sw

2.1.2 其他结构

为了防止 Hazards，通过 ctrl 模块实现了流水线的暂停。同时通过实现 forwarding 来避免数据相关。

2.2 大致结构

实现的模块和大致结构如下图（仅 ori 指令）：



之后为了实现其他指令、forwarding、流水线 stall 等陆续添加了其他若干接口。

2.3 代码实现模块

在代码上，实现了这些模块：

PC

给出指令地址。

IF-ID

暂时保存取指令阶段取得的指令，以及对应的指令地址，并在下一个时钟传递到译码阶段。

Regfile

实现了 32 个 32 位通用整数寄存器，可以同时进行两个寄存器的读操作和一个寄存器的写操作。

ID

对指令进行译码，得到最终运算的类型、子类型、源操作数、目的寄存器地址等信息。

ID-EX

将译码阶段取得的结果在下一个时钟传递到流水线执行阶段。

EX

根据获得的数据进行运算。

EX-MEM

将执行阶段取得的结果在下一个时钟传递到流水线访存阶段。

MEM

访问数据储存器 RAM。

MEM-WB

将访存阶段取得的结果在下一个时钟传递到流水线回写阶段。

RAM

实现了 4 个按字节寻址的 8 位储存器。

MIPS

对各个模块进行实例化，按照链接关系图链接。

2.4 遇到的问题

ModelSim 软件运行比较慢。而且没有找到除了直接看代码之外比较有效率的调试方法。由于是模拟硬件运行，很多规则都是自己规定，在体系结构对于运算速度的影响上缺少一些直观感受。

2.5 正确性验证方法

首先，《自己动手写 CPU》[1] 中对于不同类型的指令都给出了测试程序，测试这些程序后波形与预期一致。
其次，通过了郑怜悯同学构造的数据。
所以，本人实现的代码应该基本正确。

2.6 关于 FPGA

作业中的 bonus 项（在 FPGA 上实现 cpu）没有完成。之前的部分代码不算太多但调试花费的时间太多，在 ddl 之前没有足够的时间学习和实现 FPGA 的使用。

3 收获与感悟

首先，通过这次大作业，更加熟悉了 MIPS 五级流水线的结构和实现。
其次，在代码方面，学习了使用 Verilog 这样的硬件语言编程。
最后一点小困惑是为什么没有人为这样类似的硬件语言开发更加好用的 ide 呢，或者只是我没有发现而已。

4 附加问题

如果明年有另一个同学担任体系结构的助教, 你认为他需要知道什么?
答：对硬件感兴趣，熟悉课程内容，最好是有体系结构方面的实际设计经验等等

参考文献

[1] S. Lei. 自己动手写 cpu.