计算机体系结构 (1) 大作业报告*

李沐阳　516021910346
Shanghai Jiao Tong University

2018 年 1 月 12 日

*https://github.com/lmxyy/Computer-Architecture-Task-2

1 设计思路

设计整体思路是按照《自己动手写 CPU》的实现的一个 Riscv 五级流水。相对于普通的 mips 五级流水，本 cpu 去掉了一些历史包袱，如延迟槽，HI/LO 寄存器等等。本 cpu 使用的是冯诺依曼结构（即指令也存放与内存中）。具体实现思路如下：

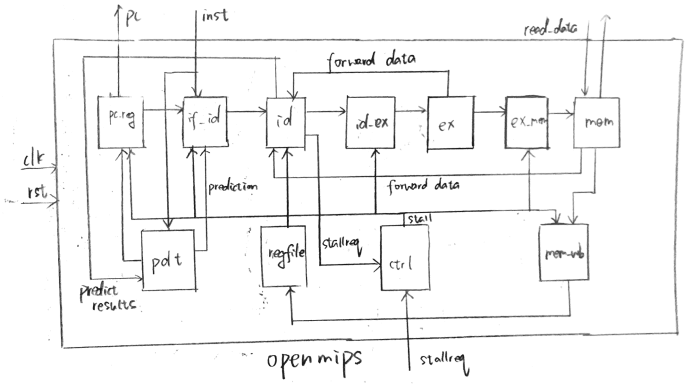

图 1: openmisp 示意图

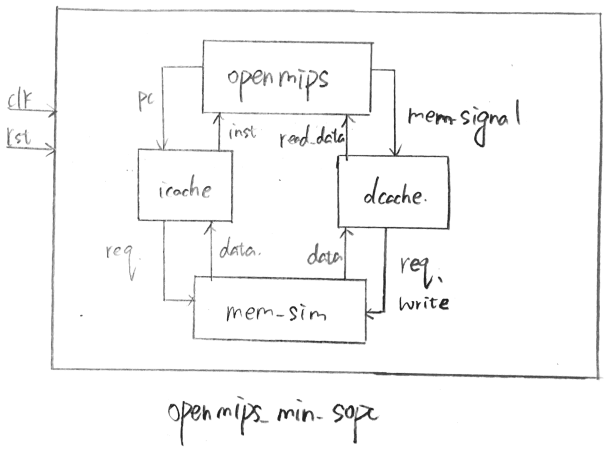

图 2: openmips_min_sopc 示意图

2 创新之处

2.1 分支预测器

pdt 模块为一个分支预测器——它采用的是 Tournament 的预测策略：

- 根据 pc 地址的 2~11 位，选择一个 alloyed branch predictor 的两位饱和计数器，由此判断使用全局预测器还是局部预测器；
- 若使用全局计数器，则直接根据最近 10 次分支结果，选择相应的全局饱和计数器。
- 若使用局部计数器，则根据 pc 地址后 2~11 位**以及最近三次分支结果**，选择相应的饱和计数器。~~（我猜这样准确率更高）~~

Predict 过程在 if 阶段就得完成，同时将结果传到 pc 和 id 阶段。id 阶段分支结果出来后，也应相应地对预测器产生相应的反馈。

2.2 Cache

- 两路主关联 cache，每块中放 8Bytes，index 位长度可调；
- 替换策略：LRU；
- Write 策略：Write through，同时只有读不命中时才会产生替换。
- 若需要替换，cache 会使 cpu 暂停直到替换结束且数据读出（icache 将 if 暂停，dcache 将除了 wb 以外的阶段全部暂停）；
- 由于实现的是内存模拟器，故可以连多根线到内存。所以 icache 和 dcache 可以同时从内存中读取数据，不需要 stall。

3 遇到问题

3.1 Cache 问题

我的 Cache 虽然很简单，但是实现起来费了很大的劲，尤其是时序一直弄不对，always 的条件一直不知道该怎么填写。我现在虽然 Cache 能够正确跑出结果，但是还是不能明白其中的原理，感觉是瞎改改出来的：

我的 icache.v 里有这样一段代码：

```
always @ (clk)
  begin
    ...
  end // always @ (clk)
```

若改为上升沿 clk 触发，那么我每次 pc=0 该 stall 的时候，pc 值都会莫名其妙变为了 4。我感觉应该是在 stall 信号来之前，pc 先加了 4。改成了 clk 触发，就没了这样的情况。

3.2 通讯问题

在我的 adapter 里，我如果写内存，我就需要发送至少 9Bytes 的数据。而助教的 uart buffer 只有 8Bytes。故无法做到将数据存入缓冲，流水继续执行，必须得 stall 到第 9 个数据也存入 buffer 之中。我的想法是需要把这个缓冲扩大，但我不知道助教为什么缓冲开到 8 就足够了。

3.3 一些想法

在五级流水之中，如果实现了像 tomasulo 那样的寄存器表用于 rename，在 data forward 的时候打上相应 tag，我感觉或许可以很大程度上面减少 stall 的节拍。

4 特别感谢

- 张凯羿同学提供的技术上的支持。
- 范舟同学提供的数据上的支持。

参考文献

[1] 《自己动手写 CPU》

[2] riscv-spec-v2.2.pdf

[3] Branch Prediction Wikipedia

[4] 助教的 MIPS CPU 实现

[5] cpu-judge

[6] Basys3-FPGA-开发板实验参考资料