一个跑上 300MHz 的 CPU

RISC-V CPU 报告

金乐盛 517030910403

2019 年 1 月 7 日

1 简介

此次体系结构大作业要求实现一个基于 RISC-V 指令集的 CPU。
我的 GitHub 仓库地址：hhttps://github.com/MrTree0-0/Architecture-2018

2 难点

1. CPU 与 Memory 的带宽为 8 位，所以对 32 位数据的操作需要花起码 4 个周期
2. 如何在上板之后提高运行速度

3 设计

实现了标准的五级流水，附带了一些微小的优化：

1. Data forwarding: 在 EX/MEM 后把将要写入内存的数据 forwarding 至 ID；在 MEM/WB 后把将要 Write Back 的数据 forwarding 至 ID

2. Static Prediction: 采用 Not taken 机制，如果后来 EX 阶段发现错误，将前面流水线寄存器倒空。

3. Over Clock: 提高 CPU 时钟频率，最高可达 300MHz。(2.8s 跑完 pi，100MHz 情况下约 9s)

4 模拟

4.1 主要问题

1. 对于时序逻辑和组合逻辑认知不清，以及掌握了一切不正确的语法
解决方法：《夏宇闻-Verilog 经典教程》

2. 调试方法十分低劣
解决方法：机智运用 $dispaly，采用 10ns 单步调试观察波形图

3. 改代码及其低效
解决方法：直接将相关模块推倒重写！并且在想清楚模块功能和具体接口之后才落手！

4. 无法确定自己错误的根源

解决方法：助教给的测试文件往往很长，可以考虑自己写一份十分简短的测试文件。附送一份内存测试代码

5 上板

5.1 主要改动

1. 重构与 Memory 相关部分 + 重构 Register

2. 查找锁存器

TIPS: 使用 Synthesis 里面的 Report Methodology，你可以在 Methodology 中找到找到所有锁存器 latch，鼠标悬停可以找到 Source Code

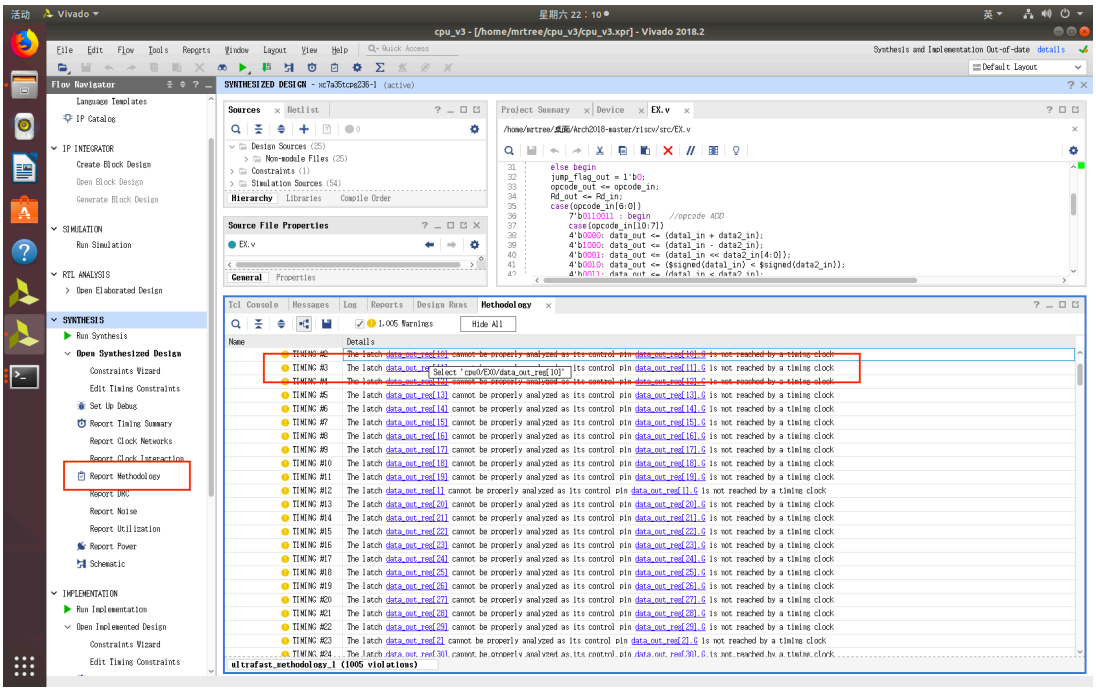

图 1: 注意红框

6 关于调频

6.1 最后情况

300MHz 是达到的最高频率，pi 的测试结果 2.8s，qsort 测试能稳定运行。

100MHz 情况下，pi 的测试结果约为 9s。

频率	Old CPU	New CPU
100MHz	Yes(MNC ≈ 0.3)	Yes(MNC ≈ 0.3)
150MHz	Yes(MNC ≈ 0)	Yes
2000MHz	No	Yes
250MHz	-	Yes
300MHz	-	Yes(MNC ≈ -2.9)
350MHz	-	No

表 1: MNC means Most Negative Slack

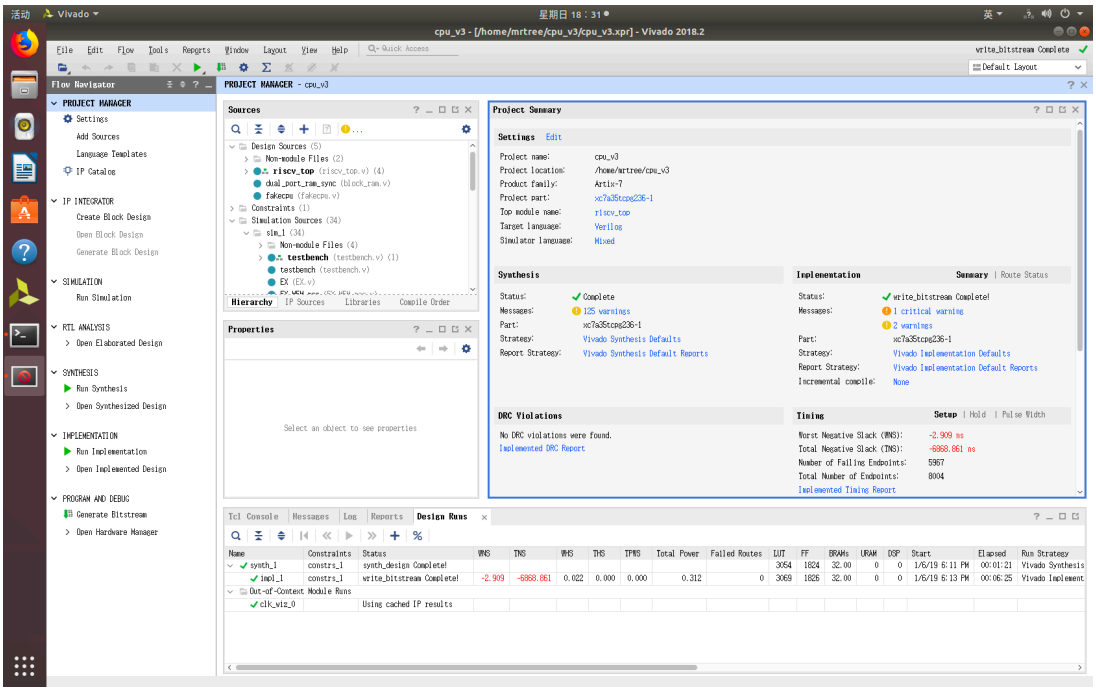

图 2: 300MHZ -2.9MNC

6.2 短板效应

整个 CPU 中限制频率的是耗时最长的模块。诸如像 ID、EX、WB 这些模块都是组合逻辑，always@* 的操作在布线的时候或多或少会被简化，想当于一堆 assign 连线，个人认为优化空间不大。有时序逻辑的地方应该是被简化的重点，所以就开始重构。

6.3 优化过程

在流水线寄存器方面没什么优化，因为本来就写成了 DQ 锁存器的形式。然后重点攻克 Memory 方面，重点是完全采用时序逻辑，完全采用非阻塞赋值，在数据读取或者写入完成后 hold 一个周期（这个可能可以淡化一部分电路延迟的弊端）。

7 总结与思考

7.1 用最简单的想法写最简单的代码做最简单的事!

7.2 与 Memory 的斗争

CPU_V0 在 Memory 方面，本人不惜代价去抢拍，采用各种奇怪的写法，其中多数是时序 + 组合混用。结果是几乎把能抢到拍都抢了，就是数据一旦读取完成，在 busy 信号拉低的同时，把数据取走。这种混乱的抢拍在模拟的时候波形图无比完美，但是一旦在 FPGA 上运行时，一点小小的延迟就会让程序乱跑。

CPU_V1 就如上面所说本人开始放弃抢拍的操作了，甚至在 mem_ctrl 把数据读完以后，过一个周期 CPU 才会去读。这种简单的写法不仅大大简化了电路，并且保证了数据的稳定性。

8 感谢

感谢梁老师、助教和各位同学们的帮助。特别感谢苏起冬同学对本人的指导——“小而快”。特别感谢陈林淇同学对本人上板调试的帮助。

参考文献

[1] 雷思磊. *自己动手写 CPU*, 电子工业出版社, 2014.

[2] 夏宇闻. *Verilog 经典教程*

[3] John L. Hennessy, David A. Patterson, et al. *Computer Architecture: A Quantitative Approach*, Fifth Edition, 2012.

[4] David A. Patterson. PPT of *CS252 Graduate Computer Architecture*, 2001.