浙 江 大 學

本科生实验报告

课程名称 计算机体系结构

姓　　名 吴同

学　　院 计算机科学与技术学院

专　　业 计算机科学与技术

学　　号 **3170104848**

指导教师 陈文智

浙江大学实验报告

专业：计算机科学与技术
姓名：吴同
学号：3170104848
日期：2019 年 11 月 20 日
地点：曹西 301

课程名称：计算机体系结构　指导老师：陈文智　电子邮件：wutongcs@zju.edu.cn
实验名称：五级流水线 CPU 解决控制竞争　实验类型：综合型　同组同学：徐欣苑

一、 实验目的和要求

1. 实验目的

- 理解控制竞争的产生原因和产生时间
- 掌握解决控制竞争的方法
- 掌握 Predict-not-taken 和 Predict-taken 的一周期停顿方法
- 掌握旁路失灵的条件
- 掌握解决数据竞争的流水线 CPU 的程序验证

2. 实验要求

- 用 Predict-taken 实现控制竞争的一周期停顿
- 用程序验证 CPU 并观察程序的执行

二、 实验内容和原理

1. 控制竞争的解决

控制竞争是由转移指令带来的由于 PC 地址不确定导致的流水线竞争。在简单的五级流水线 MIPS CPU 中，遇到转移指令时，要在 MEM 阶段才能获知是否转移和转移地址，所以必须插入三个停顿，用于等待正确的 PC 地址。这样会极大地影响程序的执行效率。

为解决控制竞争带来的停顿，应尽早计算出转移条件满足情况和转移地址。原先的计算发生在 MEM 阶段，在本实验中提前到 ID 阶段，将三个停顿减少到一个停顿。为了将这一个停顿也消除掉，使用延迟槽，调整软件层面的汇编代码，将转移指令的后一条指令调整为一定会执行，且调整顺序后对程序执行结果没有影响的指令。

2. 数据通路图

本实验的数据通路图有以下变化：

- EXE 阶段负责旁路模块的多路选择器调整到 ID 阶段
- MEM 阶段的转移地址计算和转移条件判断调整到 ID 阶段
- MEM 阶段增加新的旁路，用于处理 LW-SW 的情况，增加 fwd_m 控制信号

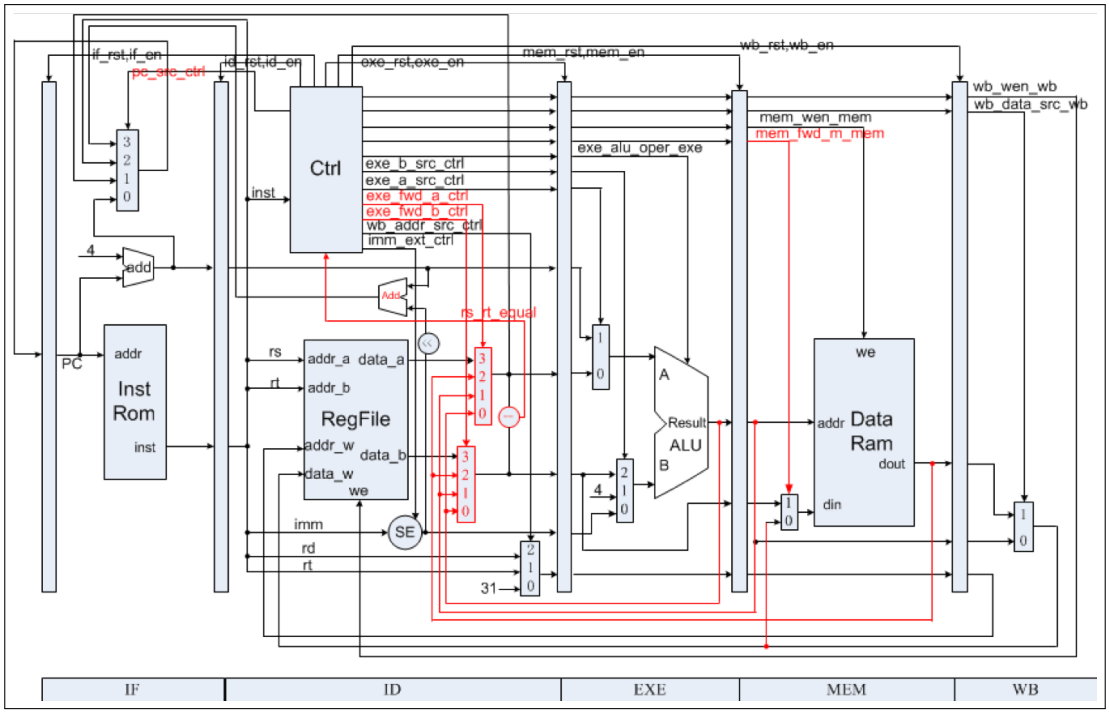

图 1: 解决控制竞争的数据通路图

三、 实验过程和数据记录

1. 将转移条件和转移地址计算提前

修改数据通路，将 ID 阶段计算转移地址，并计算 rs 与 rt 的数据是否相等。控制器译码得到 BEQ 或 BNE 指令时，根据数据通路中计算出的 rs 与 rt 是否相等，直接给出是否转移的信号。

2. 将旁路提前

修改数据通路和控制器，将旁路模块提前到 ID 阶段。修改后的逻辑如下：

- 如果本条指令读上一条指令（当前处于 EXE 阶段）写的寄存器，上一条指令不是 LW 指令，则将上一条指令的 ALU 输出送入当前指令的 rs 或 rt；如果上一条指令是 LW 指令，则其写入寄存器的内容当前不可知，要停顿一个周期。
- 如果本条指令读上一条指令的上一条指令（当前处于 MEM 阶段）写的寄存器，上一条指令不是 LW 指令，则将当前 MEM 阶段的 ALU 输出送入当前指令的 rs 或 rt；如果上一条指令是 LW 指令，则将从内存中取出的数据送入当前指令的 rs 或 rt。
- 特别地，如果发生数据竞争时，本条指令是 SW 指令，上一条指令是 LW 指令，SW 指令中参与数据竞争的是 rt 寄存器，由于 rt 的内容要到 MEM 阶段才使用，所以可添加新的旁路解决。即在本条指令到 MEM 阶段时，将处于 WB 阶段的寄存器写回数据送入内存的写口。

3. 修改验证程序

由于转移地址在 ID 阶段得到，转移指令会带来一个周期的停顿，这个问题无法通过改进硬件设计来解决，所以采用延迟槽技术，在软件层面解决这一问题。即在转移指令后加入一条一定会执行的指令，这一指令执行顺序的改变不能影响程序的执行结果。

本实验的测试程序中，0x0000000C 处的指令为 jal sum，是无条件转移指令。0x00000008 处的 addi 指令与这条转移指令的执行结果互不影响，所以调整两者顺序。

0x00000040 处的 bne $3, $0, loop 指令为条件转移指令，0x00000038 处的 addi $4, $4, 4 指令与 0x0000003C 和 0x00000040 处的指令执行结果互不影响，所以将 addi 指令的顺序调整到 bne 指令后。由于偏移地址发生变化，调整后的 bne 指令改为 0x1460FFFB。

0x00000048 处的 jr $ra 指令为无条件转移指令，0x00000044 处的 or 指令与这条转移指令的执行结果互不影响，所以调整两者顺序。

4. 仿真验证

对修改后的 MIPS 核进行仿真。仿真测试中，程序的运行时间比上次实验少了 240ns，为 12 个时钟周期。仿真结果如下：

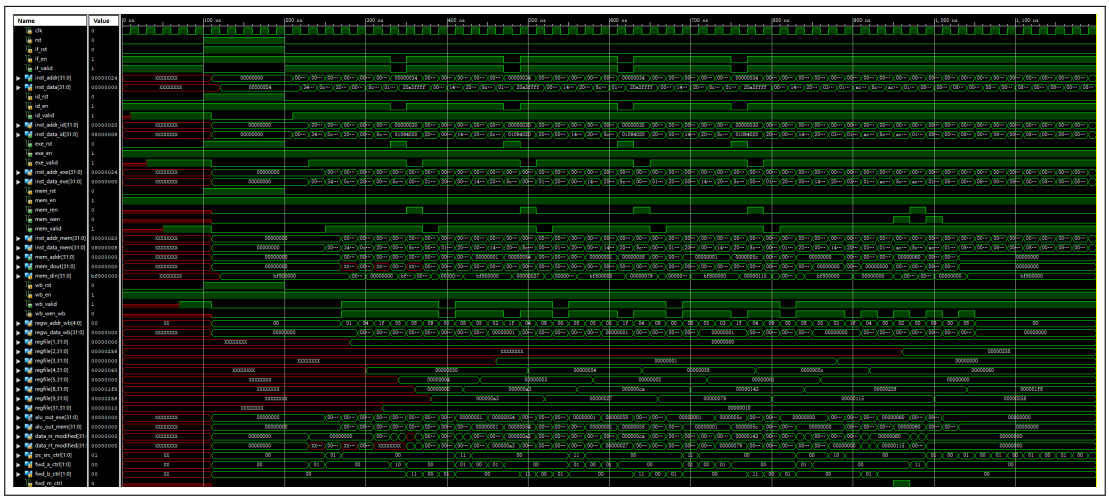

图 2: 仿真测试结果（0 ∼ 1200ns）

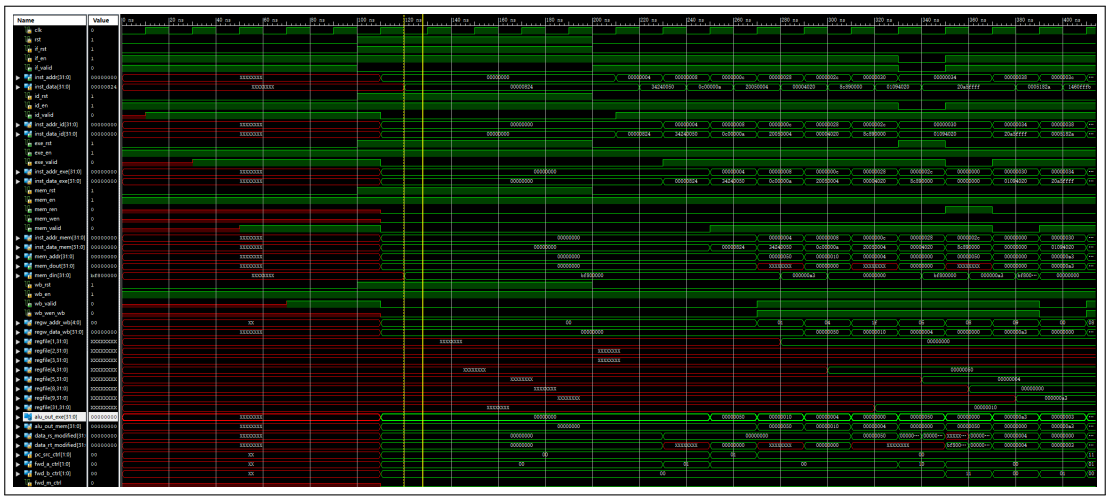

图 3: 仿真测试结果（0 ∼ 400ns）

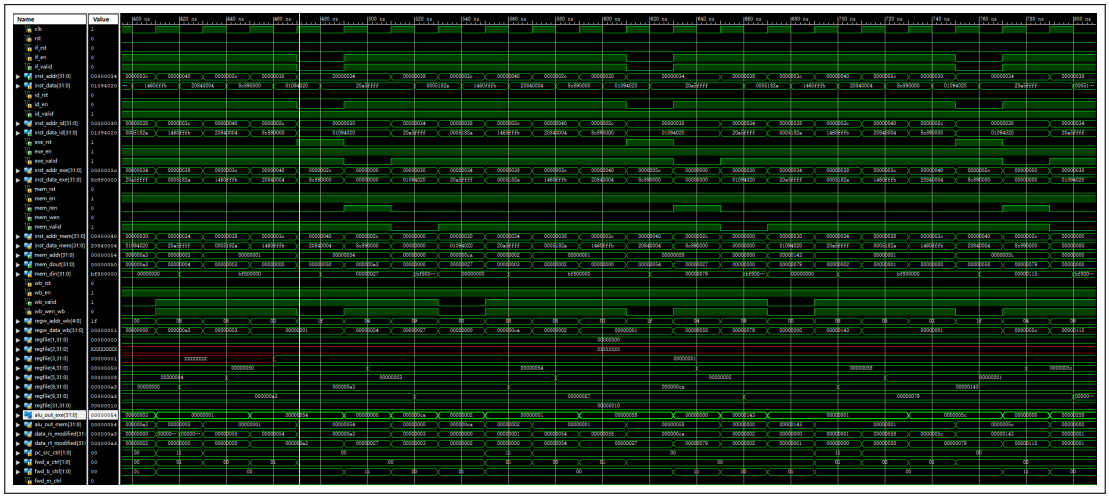

图 4: 仿真测试结果（400 ~ 800ns）

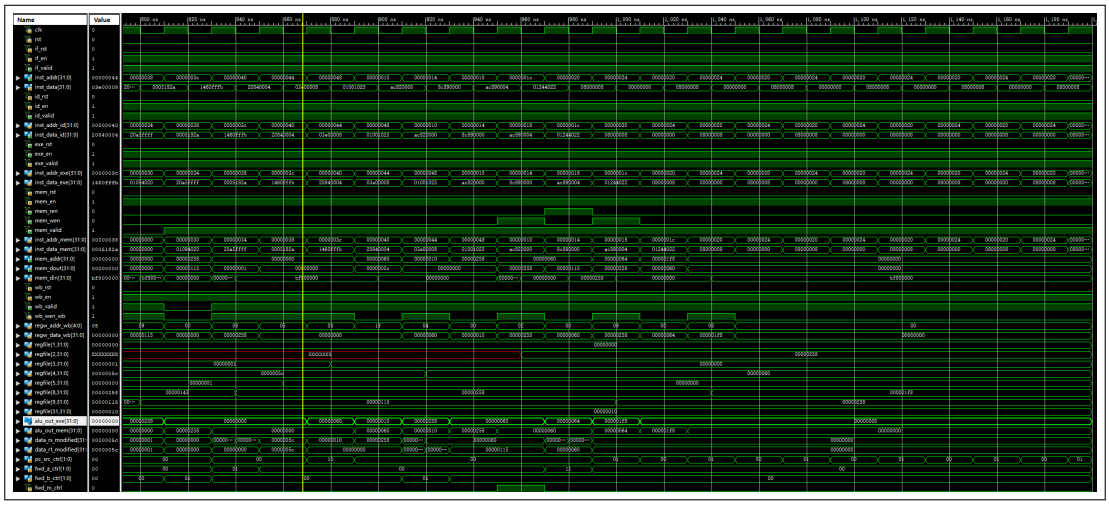

图 5: 仿真测试结果（800 ~ 1200ns）

5. 物理测试

将修改后的 MIPS 核与存储器及外设器件综合，下载到 SWORD 板上，程序的执行结果符合预期。

四、 实验结果分析

图 6是 jal 指令执行情况，jal 指令译码期间，IF 取下一条 0x20050004 指令，再下一条是跳转后位于 0x00000028 的指令。

图 7是 bne 指令执行情况，bne 指令译码期间，IF 取下一条 0x20840004 指令，再下一条是跳转后位于 0x0000002C 的指令。

图 8是 jr 指令执行情况，jr 指令译码期间，IF 取下一条 0x01001025 指令，再下一条是跳转后位于 0x00000010 的指令。

图 9是 LW-SW 数据竞争的解决情况，lw $9, 0($4) 指令后紧跟 sw $9, 4($4)，fwd_m_ctrl 信号被置为高电平，使用新增的旁路解决了数据竞争。

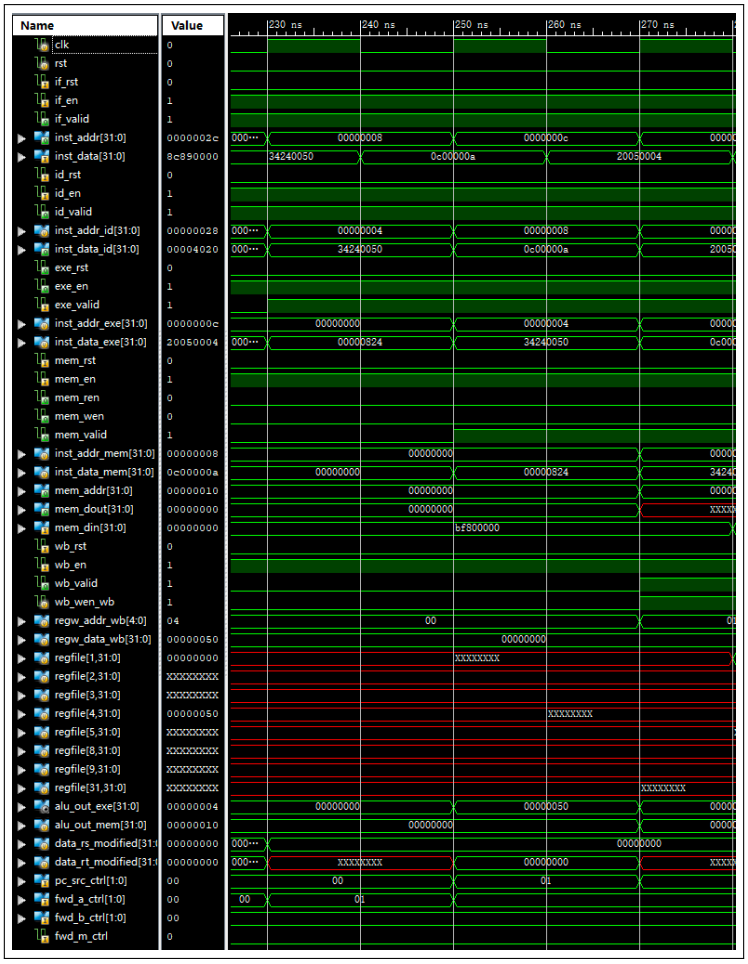

图 6: jal 指令的执行

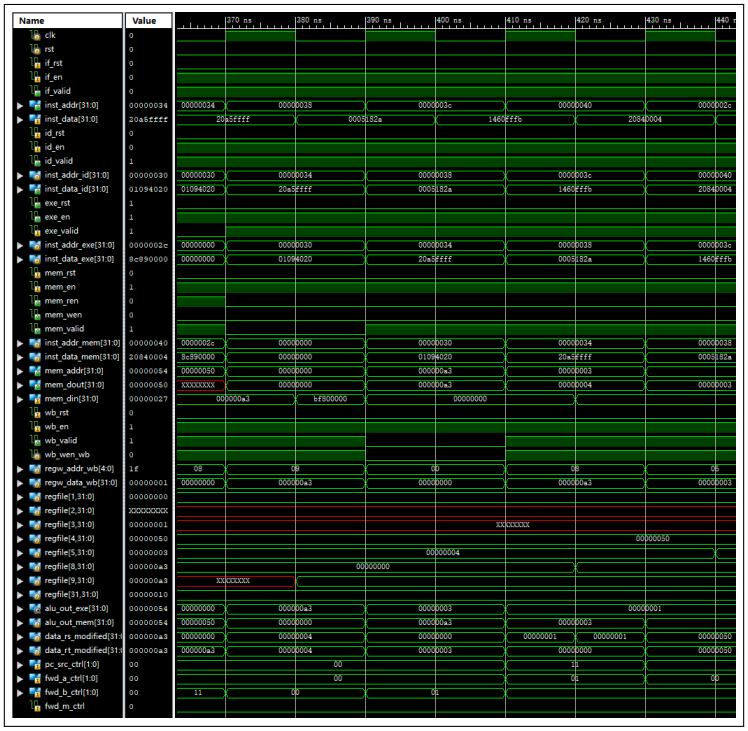

图 7: bne 指令的执行

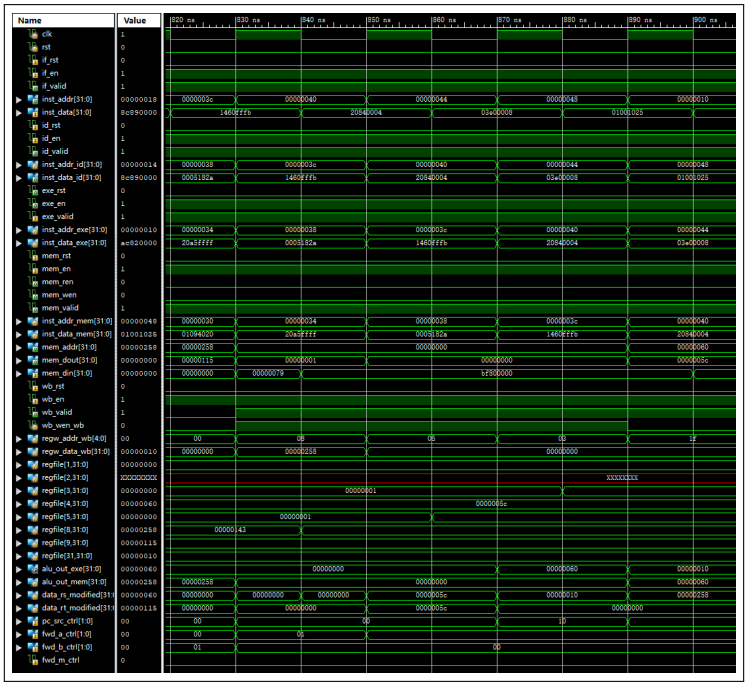

图 8: jr 指令的执行

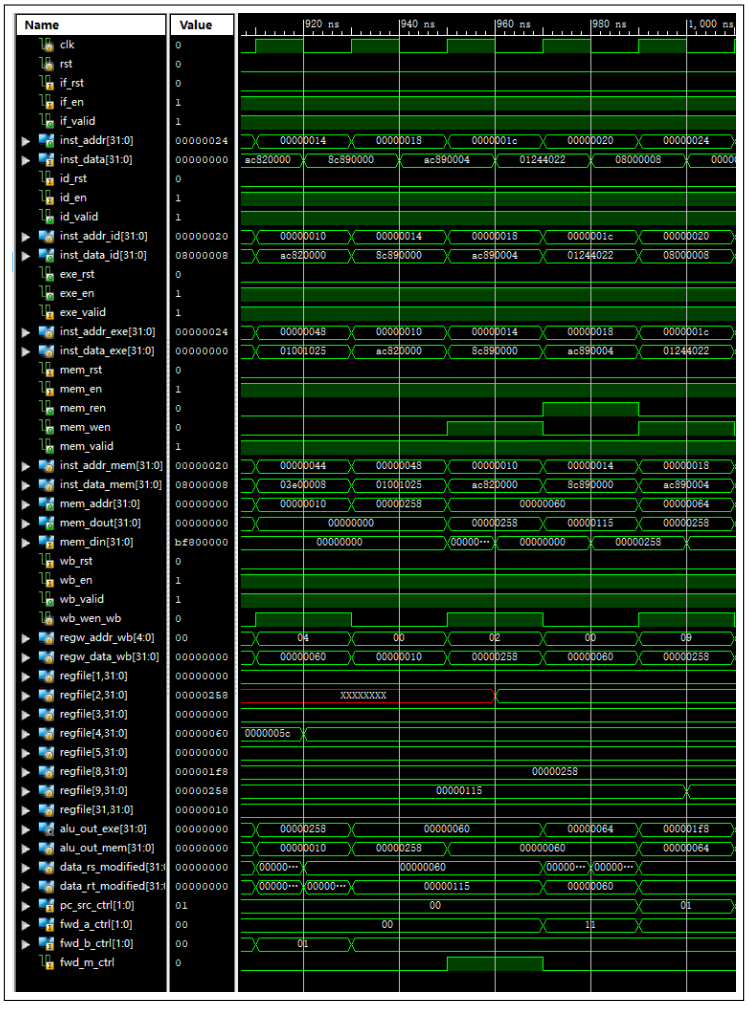

图 9: LW-SW 数据竞争的解决

五、 讨论与心得

这是与流水线设计有关的最后一个实验。在本次实验中，所有类型的竞争都得到了解决，由于竞争导致的停顿被减到最少。从最初的五级流水线 CPU，到解决了所有竞争的 CPU，同样程序的执行时间一次次减少，硬件性能得到了优化和提升。通过实验的开展，我对理论课上学到的流水线知识有了更加深入的理解，对于各种竞争产生的条件和处理方式掌握得更加熟练。