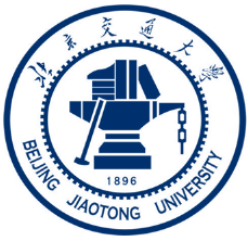


北京交通大學

BEIJING JIAOTONG UNIVERSITY

《计算机体系结构》

实验报告

实验名称:	实验 3_ MIPS 流水线实验报告
学　号:	
姓　名:	
学　院:	计算机与信息技术学院
日　期:	2022 年 10 月 25 日

目录

1. winMIPS64 模拟器

熟悉 winMIPS64 模拟器，并确定指令格式中各个域的具体值。包括如下内容:
① 将附件中的 winmips64.zip 文件解压到你的电脑中。
② 阅读附件中的 winmipstut-6-3.3.docx 文件,并按其中的步骤操作、学习 winMIPS64 模拟器。

按照教程中的样例一步步进行。下边是剪影

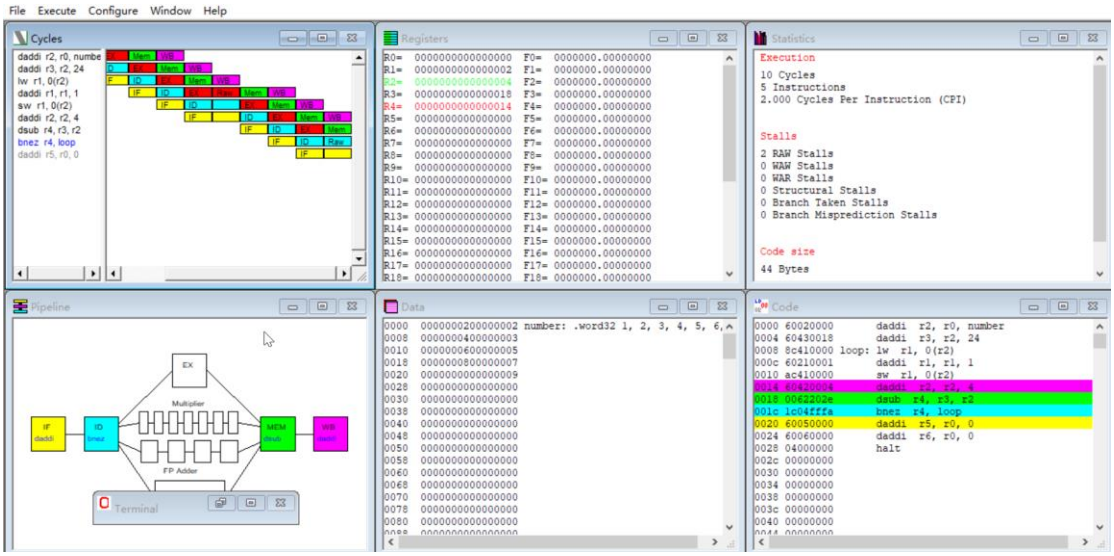


图 1-1 模拟器学习

2. 了解指令在 MIPS 流水线中的运行过程

了解 RISC-V 的在线模拟器 Venus 的使用方法。包括如下内容:

- 通过阅读理解程序 incre.s，指出该程序在运行时可能会出现哪些数据相关和控制相关。
- 将 incre.s 读入到模拟器中，关闭直送(forwarding)，单步运行程序，考察一个循环内程序的各个指令在各个周期的运行情况。对最下面的状态栏上指示的 RAW 和 Branch taken 冒险，解释其产生的原因。

2.1. 可能出现的数据相关和控制相关

数据相关：RAW

RAW 如

```
    daddi  r2, r0, number
    daddi  r3, r2, 24
```

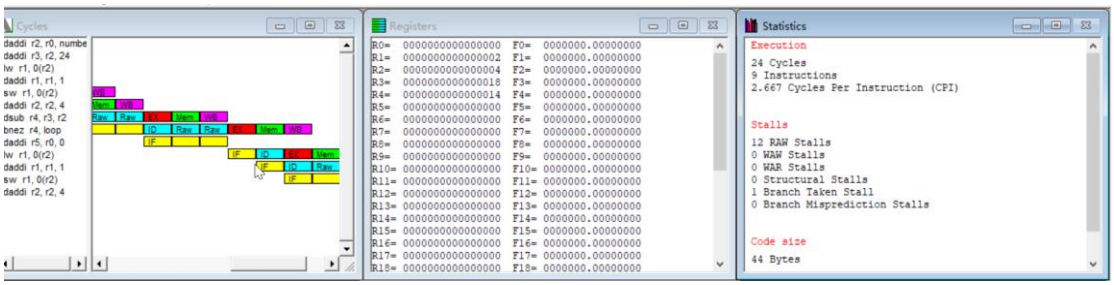
和

```
     lw  r1, 0(r2)
    daddi  r1, r1, 1
```

控制相关：Branch taken

```
    dsub  r4, r3, r2
    bnez  r4, loop
```

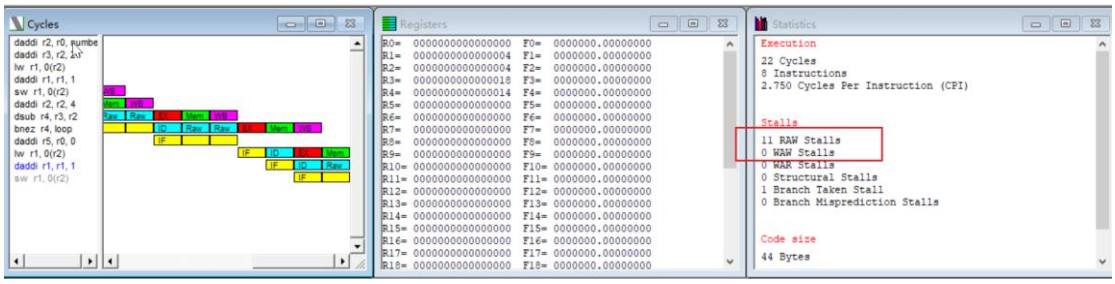
2.2. RAW Branch taken 的分析



图表 2-1 程序运行截图

```
    daddi  r2, r0, number
    daddi  r3, r2, 24
loop: lw  r1, 0(r2)
    daddi  r1, r1, 1
    sw  r1, 0(r2)
    daddi  r2, r2, 4
    dsub  r4, r3, r2
    bnez  r4, loop
```

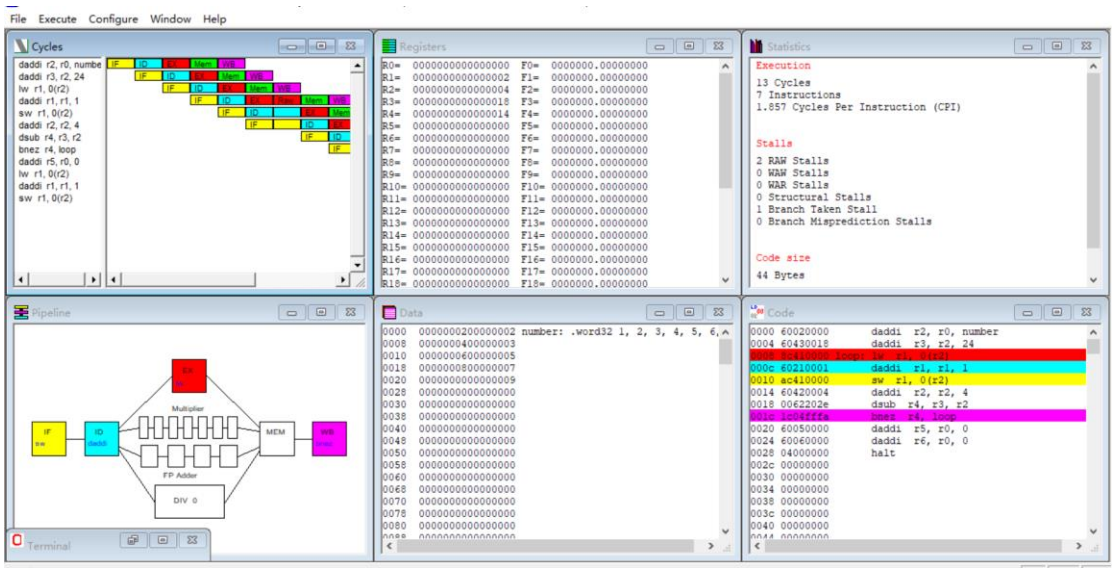
第一、二句 RAW R2。
三四句，R1 RAW
四五句，R1 RAW 。SW 读 R1，daddi 写 R1。
六七句，R2 RAW。
最后两句，R4 branch taken，bnez 需要读 R4。



图表 2-2 一轮的 incre.s 执行结果

3. Forwarding 对流水线的影响

考察直送(forwarding)技术对流水线性能的影响
① 在打开直送(forwarding)功能的情况下，单步运行 incre.s 并考察一个循环内程序的各个指令在各个周期的运行情况，解释直送是如何解决了很多冒险情况的。
② 指出哪些仍未消失的冒险为什么直送也无法解决。



图表 3-1 打开直送时的情况

直送的解决办法：

通过在 exe 阶段后由流水线寄存器直送到所需要的 ALU 的输入端，可以解决两个计算指令的 RAW 的问题。

不能解决的问题：

Lw 后的 alu 不能消除。Lw 要到 DM 后才能得到操作数，而后边紧接着的 ALU 算数指令，要在 exe 之前得到操作数，这样就会发生 RAW。

还有 ALU 算数指令之后的 branch 指令是不能解决的。转移中采用的是预测不成功，这样如果，转移成功，需要阻塞一个周期，重新恢复现场。并且这里情况很特殊，根据模拟器来说，应该是 RAW stall 和 branch taken stall 都有。Exe 和 ID 是同一周期的 ID 没法得到 R4 的值，所以只能阻塞；同时转移成功的 stall 也有，两个合并了。

4. 通过调度减少冒险

① 考察 incre.s，看看能否在不改变程序运行结果的情况下，通过改变指令的执行顺序来进一步减少冒险产生的 stall。要求：可以修改指令的操作数，但不能增加或减少指令。

② 解释你的做法为什么能减少冒险。

可以这样修改：

原程序：

```
    daddi  r2, r0, number
    daddi  r3, r2, 24
loop: lw  r1, 0(r2)
    daddi  r1, r1, 1
    sw  r1, 0(r2)
    daddi  r2, r2, 4
    dsub  r4, r3, r2
    bnez  r4, loop
```

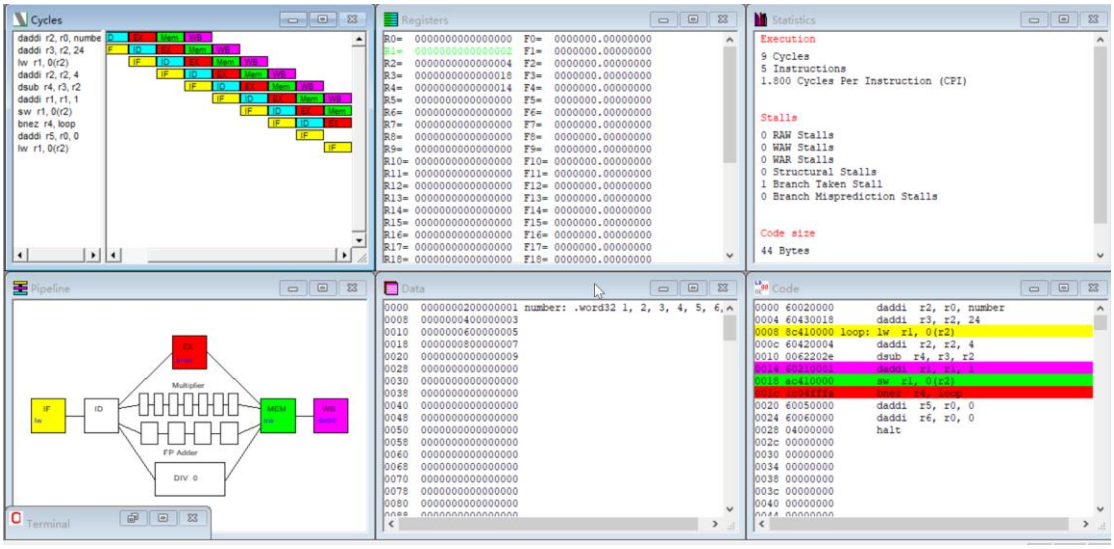
修改为

```
    daddi  r2, r0, number
    daddi  r3, r2, 24
loop: lw  r1, 0(r2)
     daddi  r2, r2, 4
    dsub  r4, r3, r2

    daddi  r1, r1, 1
    sw  r1, -4(r2)
      bnez  r4, loop
```

原理解释：

原程序前 4 条指令的相对顺序是不能修改的，他们彼此之间是有关联关系的。这样我们可以把 R2 的计算的自增计算放前面，R4 的也提前，这样可以减少 LW 后的 RAW 和 BNEZ 的后 RAW。同时，由于 R2 的变化，我们在 SW 时要把偏移量改为-4。



图表 4-1 调度之后的结果

这样，就只有一次由于转移预测错误的 STALL，RAW 的完全解决了。

5. 实验总结

对以上的三个步骤，分别考察统计信息（显示在 winMIPS64 的 statistics 窗口中）。通过列表的方式对本次实验做一个总结。

fowarding	调度	Cycle	instruction	CPI	RAW stall	branch taken stall
1	1	9	5	1.8	0	1
1	0	11	6	1.833	2	1
0	0	19	7	2.714	10	1

表格 1 三种情况比较

Forwarding 和调度 1 代表有，0 代表无。

通过实验的汇总表格，我们可以知道，forwarding 技术，在流水线中可以有效减少 RAW Stall 的现象，但是对由于转移造成的 Stall 和 LW 后算数指令的 stall 无能为力，这个时候，我们就可以采用调度的方法进行控制，进一步减少 stall 的数量。同时，使用 forwarding 和调度可以极大的减少 CPI，也就是我们减少 Stall 数的效果。

通过这次实验，我对流水线的运行方式和 stall 的产生与处理有了更深的理解，更感受到了计算机科学家的思维魅力。