**深 圳 大 学 实 验 报 告**

**课 程 名 称： 计算机系统(3)**

**实验项目名称： 处理器结构实验一**

**学 院： 计算机与软件学院**

**专 业： 计算机与软件学院所有专业**

**指 导 教 师： 罗秋明**

**报告人： 马庭俊 学号： 2018152102 班级： 数计班**

**实 验 时 间： 2020年11月19日星期五**

**实验报告提交时间： 2020年11月25日星期三**

**教务处制**

**一、 实验目标：**

了解MIPS的五级流水线，和在运行过程中的所产生的各种不同的流水线冒险

通过指令顺序调整，或旁路与预测技术来提高流水线效率

更加了解流水线细节和其指令的改善方法

更加熟悉MIPS指令的使用

**二、实验内容**

观察一段代码并运行，观察其中的流水线冒险，并记录统计统计信息。

对所给的代码进行指令序列的调整，以期避免数据相关，并记录统计信息。

启动forward功能，以获得性能提升，并且记录统计信息。

（选做：用perf记录x86中的数据相关于指令序列调整后的时间统计、

调整指令，以避免连续乘法间的阻塞。）

**三、实验环境**

硬件：桌面PC

软件：Windows，WinMIPS64仿真器

**四、****实验步骤及说明**

首先，我们给出一段C代码，该段代码实现的是两个矩阵相加。

设有4\*4矩阵A和4\*4矩阵B相加，得到4\*4矩阵C：

for(int i = 0; i < 4; i++)

For(int j = 0; j < 4; j++)

C[i][j] = A[i][j] + B[i][j];

根据上述的C代码，我们将其转换成MIPS语言，然后运行，并进行分析。

MIPS代码如下：

.data

a: .word 1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4

b: .word 4,4,4,4,3,3,3,3,2,2,2,2,1,1,1,1

c: .word 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

len: .word 4

control: .word32 0x10000

data: .word32 0x10008

.text

start:daddi r17,r0,0

daddi r21,r0,a

daddi r22,r0,b

daddi r23,r0,c

ld r16,len(r0)

loop1: slt r8,r17,r16

beq r8,r0,exit1

daddi r19,r0,0

loop2: slt r8,r19,r16

beq r8,r0,exit2

dsll r8,r17,2

dadd r8,r8,r19

dsll r8,r8,3

dadd r9,r8,r21

dadd r10,r8,r22

dadd r11,r8,r23

ld r9,0**(r9**)

ld r10,0(r10)

dadd r12,r9,r10

sd r12,0(r11)

daddi r19,r19,1

j loop2

exit2:daddi r17,r17,1

j loop1

exit1: halt

实验前请保证winMIPS64配置中“Enable Forwarding”没有选中。将这段代码加载到WinMIPS64中，运行后观察结果（提供Statistic窗口截图）。 从Statistic窗口记录：本程序运行过程中总共产生了多少次RAW的数据相关。接下来，我们对产生数据相关的代码逐个分析，请列出产生数据相关的代码，并在下一步中进行分析和优化。

如图 1，产生了220次RAW的数据相关。

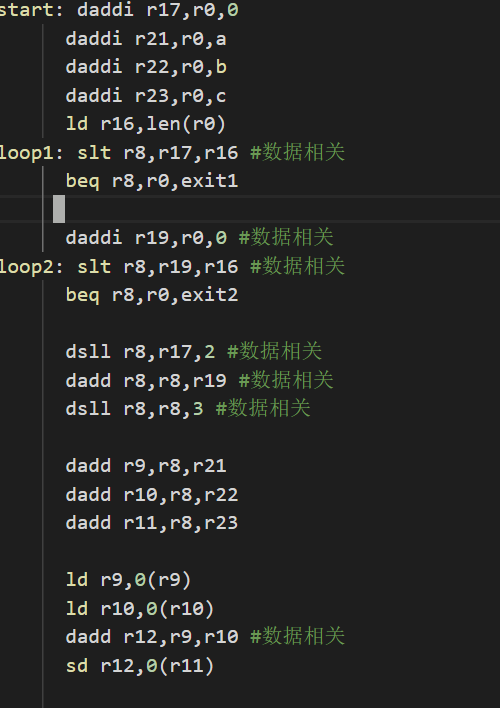
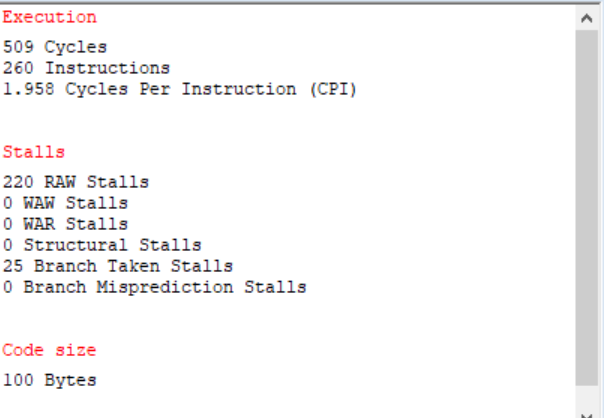


图 1，初始时数据相关次数和产生数据相关的指令

产生数据相关的指令为：

slt r8,r17,r16

daddi r19,r0,0

slt r8,r19,r16

dsll r8,r17,2

dadd r8,r8,r19

dsll r8,r8,3

dadd r12,r9,r10

1. 调整指令序列

在这一部分，我们利用指令调整的方法对数据相关代码进行优化，规避数据相关。

通过**调整序列**来规避这个数据相关，在statics窗口中记录其效果。将此结果与初始的结果进行对比，报告**RAW相关的次数减少**的数量。

调整如下指令，如图 2。

daddi r19,r19,1 #

dadd r11,r8,r23#

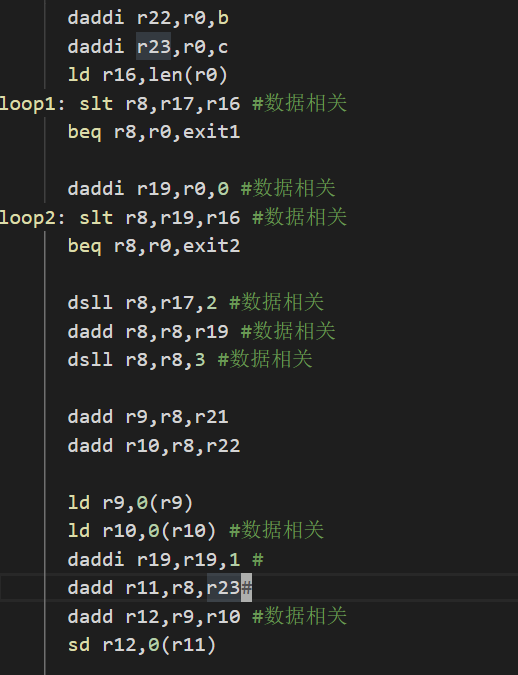
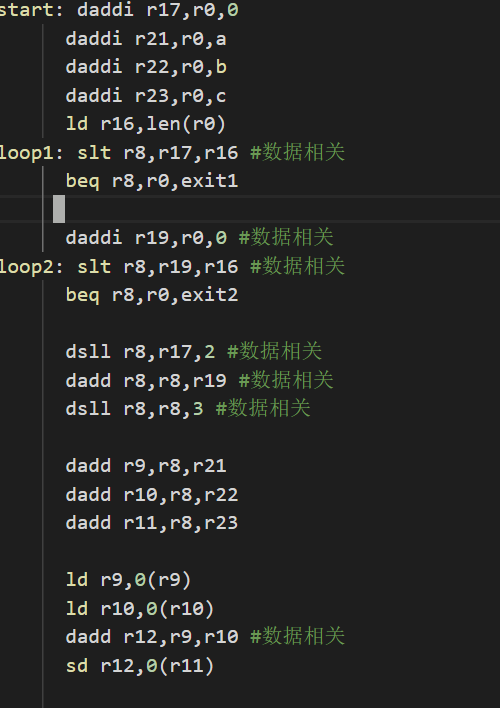


图 2，调整前（左）与调整后指令比较（右）

RAW的相关次数减少为204，如图 3所示。

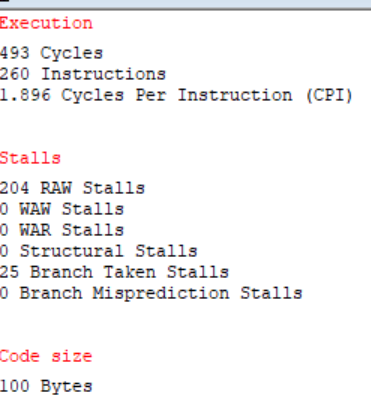
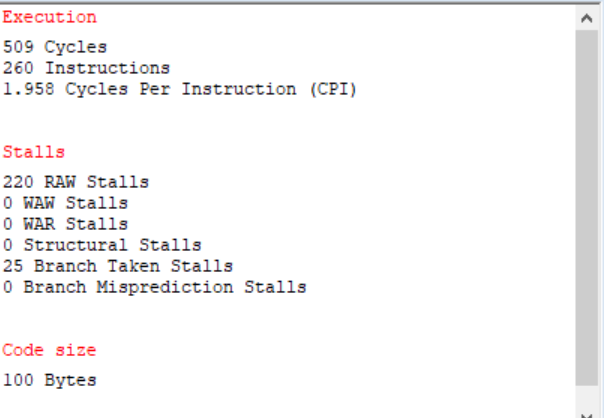
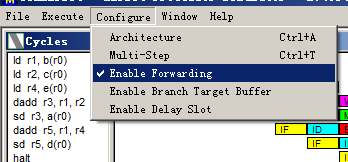


图 3，调整前（左）与调整后（右）的raw次数比较

1. Forwarding功能开启

接下来，我们要展示Forwarding功能的优化效果。

首先，我们要知道如何开启Forwarding功能。法如下：点开***configure***下拉窗口，给***Enable Forwarding***选项左侧点上勾。



开启了Forwarding功能之后，我们再运行，查看结果，解释哪些数据相关的问题得到解决，并以截图说明问题解决前后的差异所在。

开启forwarding之后，raw的次数大幅降低，结果如图 4所示。

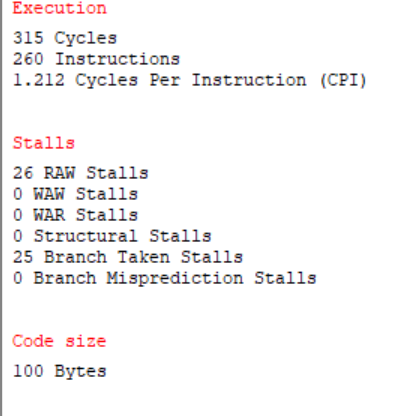
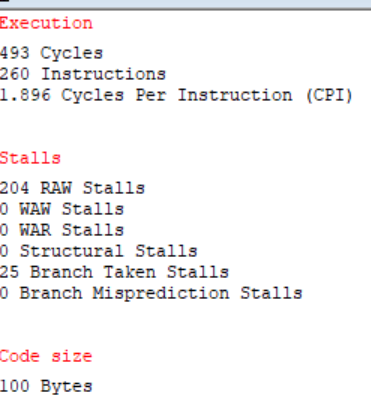


图 4，开启forwarding前（左）与开启forwarding之后（右）RAW次数比较

解释，某些因为数据相关而阻塞的指令可以不用在上一条指令得到结果再执行（如图 5），可以通过forwarding的方式在ex阶段就可以将数据传到下一条指令。

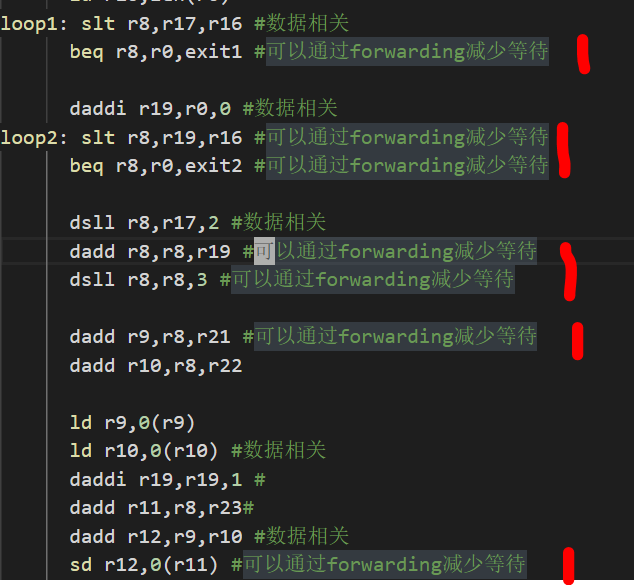


图 5，通过forwarding方式减少raw的指令（红色标记）

1. 结构相关优化

流水线中的结构相关，指的是流水线中多条指令在同一时钟周期内争用同一功能部件现象。即因硬件资源满足不了指令重叠执行的要求而发生的冲突。

在WinMIPS64中，我们可以在除法中观察到这种现象。要消除这种结构相关，我们可以采取调整指令位置的方法进行优化。在这个部分，我们首先给出几条C代码，然后将该代码翻译成MIPS代码（为了观察的方便，我们这里MIPS代码并不是逐一翻译，而是调整代码，使得其他部分数据相关已经优化，而两条除法指令连续出现），运行并查看结果。接着，调整代码序列，重新运行。观察优化效果。

下面是给出的C代码：

a = a / b

c = c / d

e = e + 1

f = f + 1

g = g + 1

h = h + 1

i = i + 1

j = j + 1

根据上述的C代码，我们给出数据相关优化的指令如下：

.data

a: .word 12

b: .word 3

c: .word 15

d: .word 5

e: .word 1

f: .word 2

g: .word 3

h: .word 4

i: .word 5

.text

start:

ld r16,a(r0)

ld r17,b(r0)

ld r18,c(r0)

ld r19,d(r0)

ld r20,e(r0)

ld r21,f(r0)

ld r22,g(r0)

ld r23,h(r0)

ld r24,i(r0)

ddiv r16,r16,**r17**

ddiv r18,r18,r19

daddi r20,r20,1

daddi r21,r21,1

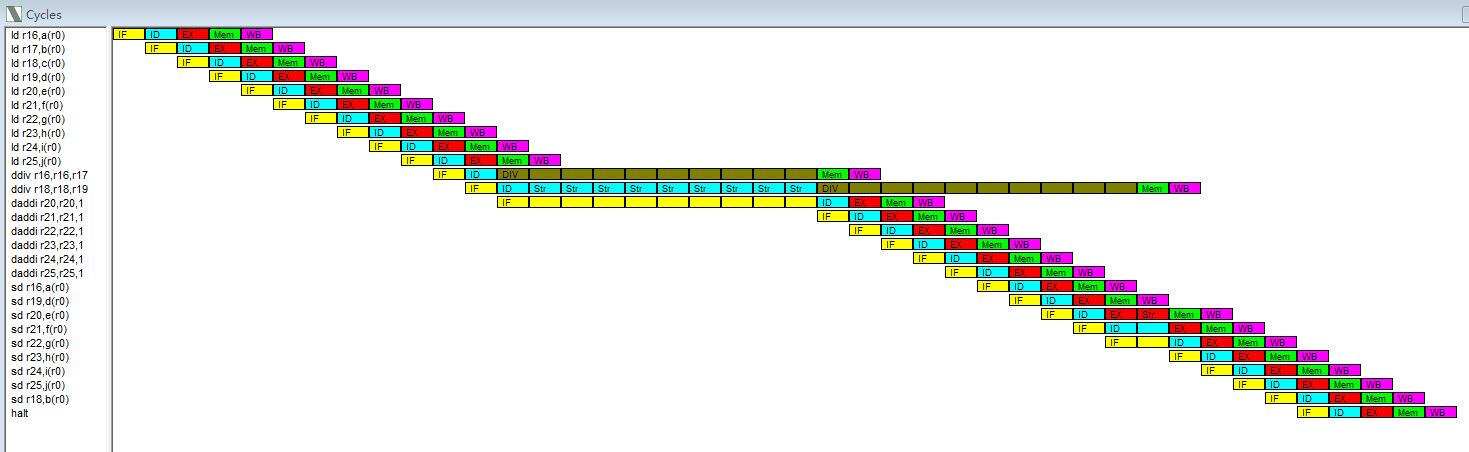
daddi r22,r22,1

daddi r23,r23,1

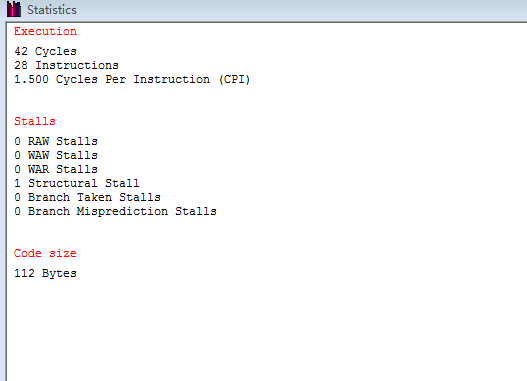
daddi r24,r24,1

halt

上面的指令运行，在***Cycle***窗口结果如下（程序运行前请将configure->architecture->division latency改为10）：



在***Statistics***窗口的结果如下：



通过观察，我们可以发现，两个连续的除法产生了明显的结构相关，第二个除法为了等待上一个除法指令在执行阶段所占用的资源，阻塞了9个周期。

显然，这样的连续的除法所导致的结构相关极大的降低了流水线效率，为了消除结构相关，我们需要做的是调整指令序列，将其他无关的指令塞入两条连续的除法指令中。

给出指令序列的调整方案并给出流水线工作状态的截图，做出解释。

调整指令如图 6。

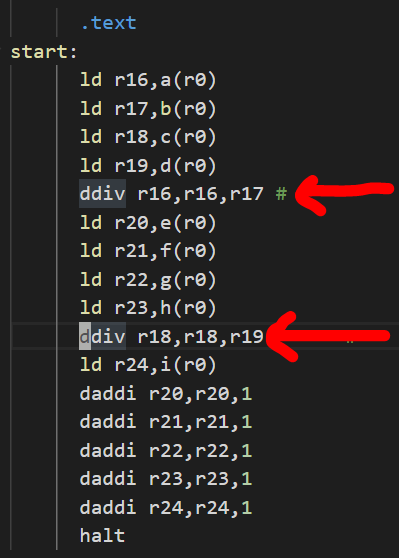


图 6，调整除法后代码

调整后下降到28个周期，如图 7所示。

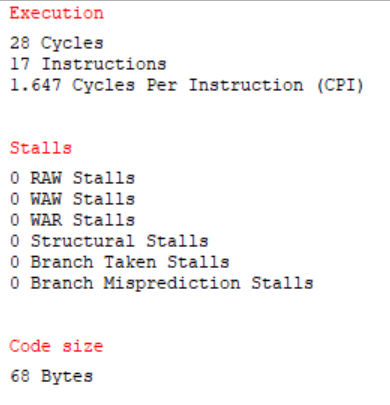
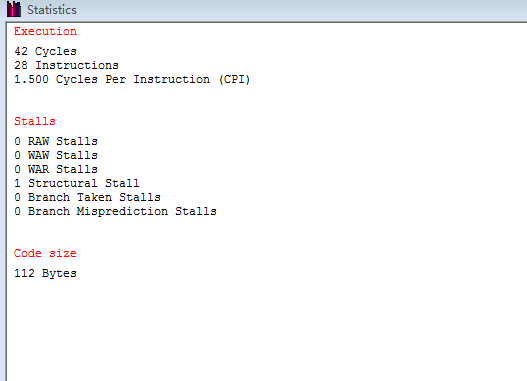


图 7，调整指令前（左）与调整后（右）的结果比较

解释：

如(图 8)左，观察阻塞，发现两个阻塞都过长，叠加在一起的周期数就更多。通过在这两个条指令之间添加指令，(如图 8)右可以减少总的周期数。

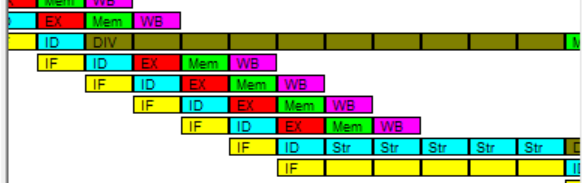
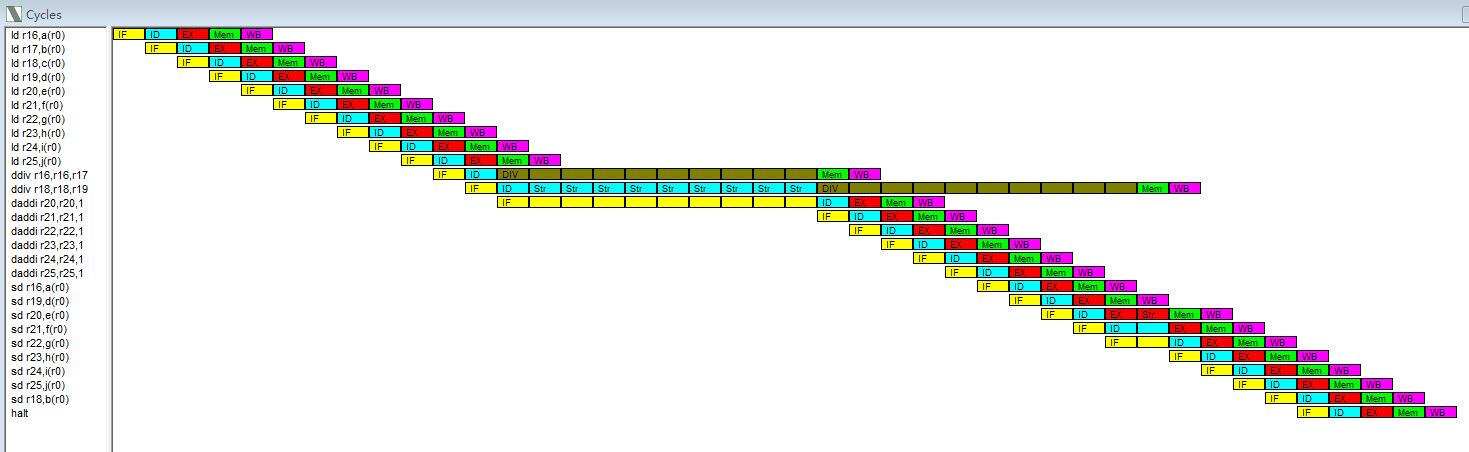


图 8，插入指令减少结构性相关

1. 提交报告

记录实验过程，保存实验截图，给出分析结果，形成实验报告。初始代码准备（10分），后面每个优化方法各30分。

**五、实验结果**

**通过调整指令顺序和forwarding技术，可以减少代码运行的时间。如**图 3**，**图 4**，**图 7**所示。**

**五、实验总结与体会**

**1. 可以通过调整指令的顺序减少数据相关，通过硬件上的优化如增加器件，forwarding技术提升程序性能。**

**2. 优化一个程序需要从软件和硬件等方面同时考虑。**

|  |
| --- |
| **指导教师批阅意见：**  **成绩评定：**  指导教师签字：  年 月 日 |
| 备注： |