

北京邮电大学 计算机学院
《计算机系统结构》实验报告

姓名 王睿嘉

学号 2015211906

班级 2015211307

实验三 MIPS 指令实现数组点积

一、实验目的和环境描述

1. 实验目的

- 1) 通过本次实验，进一步熟悉实验1和2的内容；
- 2) 增强汇编语言编程能力；
- 3) 学会使用模拟器中的定向功能进行优化；
- 4) 了解对代码进行优化的方法。

2. 实验平台

指令级和流水线操作级模拟器 MIPSsim。

二、实验原理和步骤描述

1. 实验原理

通过 MIPS 指令实现所要求的功能。在 MIPSsim 上执行代码，观察相应操作的通用寄存器，判断执行结果是否正确。同时，观察统计栏中的执行周期总数，判断优化及定向前后执行效率的变化，并理解静态调度的情况。

2. 实验步骤

- 1) 自行编写计算两向量点积的汇编程序，要求可以求得两向量点积后的结果。
向量元素使用数组存储，要求维度不得小于10；
- 2) 启动 MIPSsim；
- 3) 载入自己编写的程序，观察流水线输出结果。

代码如下：

```
.text
main:
    ADDIU $r1,$r0,array1      #取向量一首地址
    ADDIU $r2,$r0,array2      #取向量二首地址
    ADDI  $r3,$r0,10           #向量维度
    ADDI  $r7,$r0,0            #清零结果寄存器
loop:
    LW    $r4,0($r1)           #取相应元素
    LW    $r5,0($r2)
    MUL   $r6,$r4,$r5
```

```

ADD    $r7,$r6,$r7
ADDI   $r1,$r1,4           #计算下一元素地址
ADDI   $r2,$r2,4
ADDI   $r3,$r3,-1
BGTZ   $r3,loop
TEQ    $r0,$r0

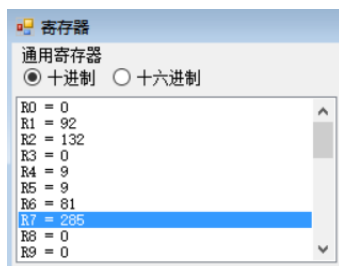
```

.data

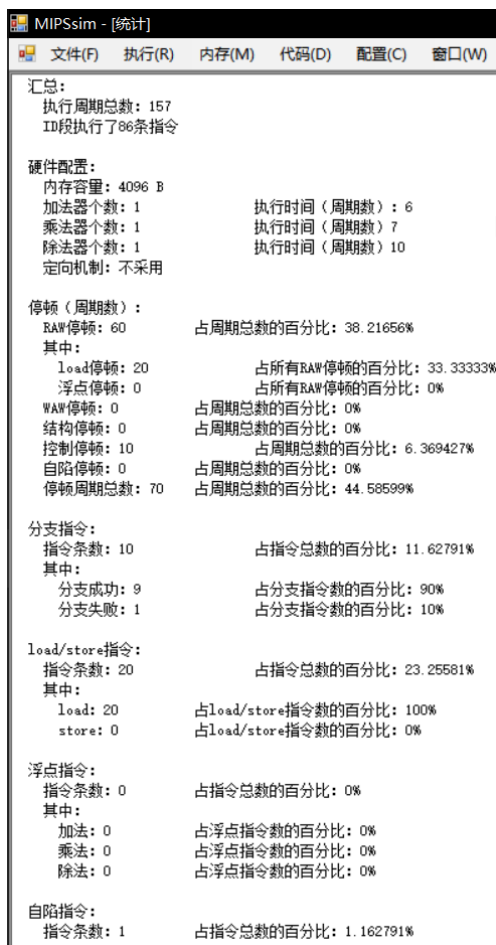
array1:.word 0,1,2,3,4,5,6,7,8,9

array2:.word 0,1,2,3,4,5,6,7,8,9

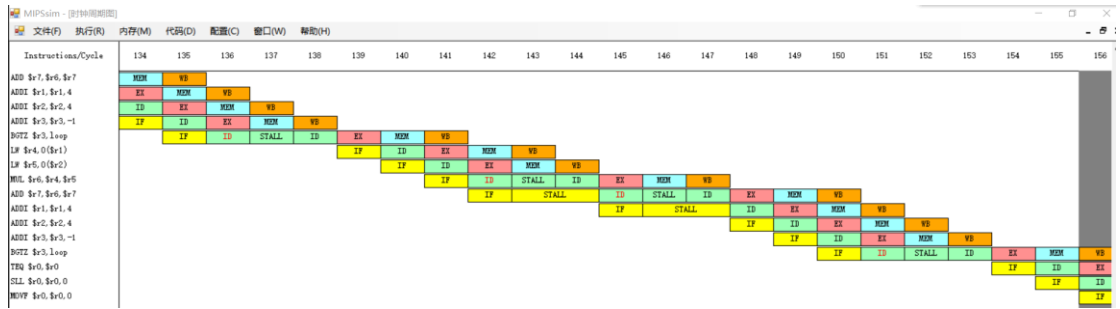
运行结果如下：



统计情况如下：

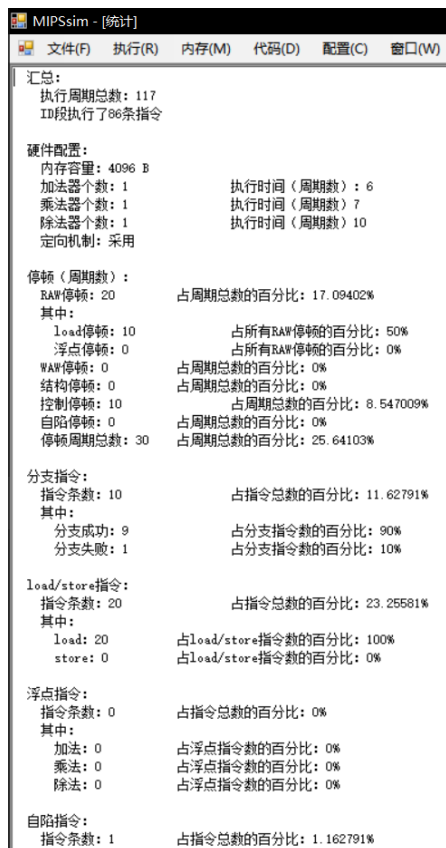


时钟周期图如下：



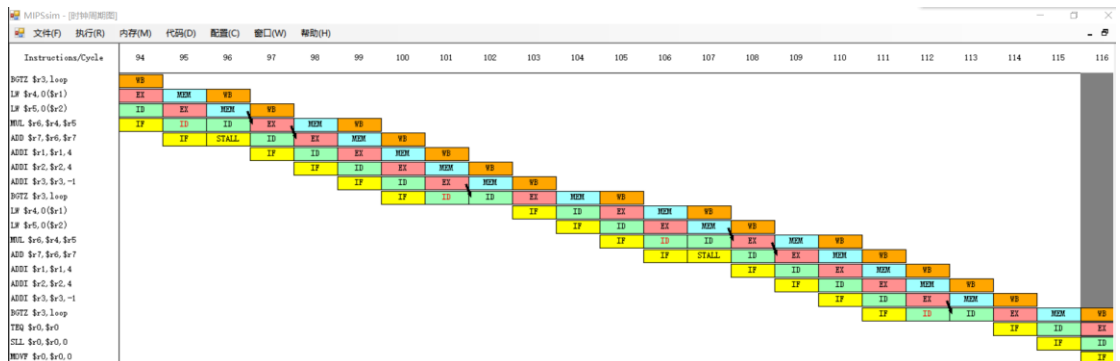
4) 使用定向功能再次执行代码，与刚才的结果进行比较，观察执行效率的不同。

统计情况如下：



由此可见，定向后的执行效率为定向前的134.188%，提高效率所减少的时钟周期均为RAW停顿周期。

时钟周期图如下：



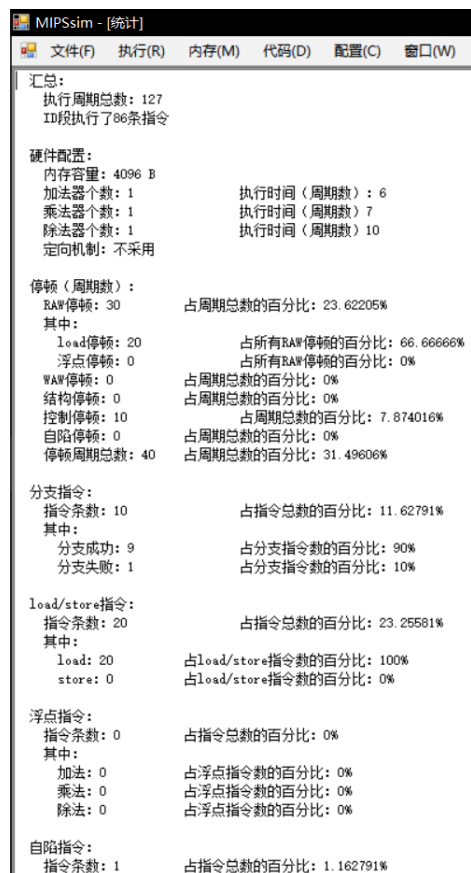
5) 采用静态调度方法重排指令序列，减少相关，优化程序。

优化后代码如下：

```
.text
main:
ADDIU $r1,$r0,array1           #取向量一首地址
ADDIU $r2,$r0,array2           #取向量二首地址
ADDI  $r3,$r0,10                #向量维度
ADDI  $r7,$r0,0                 #清零结果寄存器
loop:
LW    $r4,0($r1)                #取相应元素
LW    $r5,0($r2)
MUL   $r6,$r4,$r5
ADDI  $r1,$r1,4                 #计算下一元素地址
ADDI  $r2,$r2,4
ADDI  $r3,$r3,-1
ADD   $r7,$r6,$r7               #优化
BGTZ  $r3,loop
TEQ   $r0,$r0

.data
array1:.word 0,1,2,3,4,5,6,7,8,9
array2:.word 0,1,2,3,4,5,6,7,8,9
```

统计情况如下：



MIPSSim - [统计]

文件(F) 执行(R) 内存(M) 代码(D) 配置(C) 窗口(W)

汇总:
 执行周期总数: 127
 ID段执行了86条指令

硬件配置:
 内存容量: 4096 B
 加法器个数: 1
 乘法器个数: 1
 除法器个数: 1
 定向机制: 不采用

停顿 (周期数):
 RAW停顿: 30 占周期总数的百分比: 23.62205%
 其中:
 load停顿: 20 占有所有RAW停顿的百分比: 66.66666%
 浮点停顿: 0 占有所有RAW停顿的百分比: 0%
 WAW停顿: 0 占周期总数的百分比: 0%
 结构停顿: 0 占周期总数的百分比: 0%
 控制停顿: 10 占周期总数的百分比: 7.874016%
 自陷停顿: 0 占周期总数的百分比: 0%
 停顿周期总数: 40 占周期总数的百分比: 31.49606%

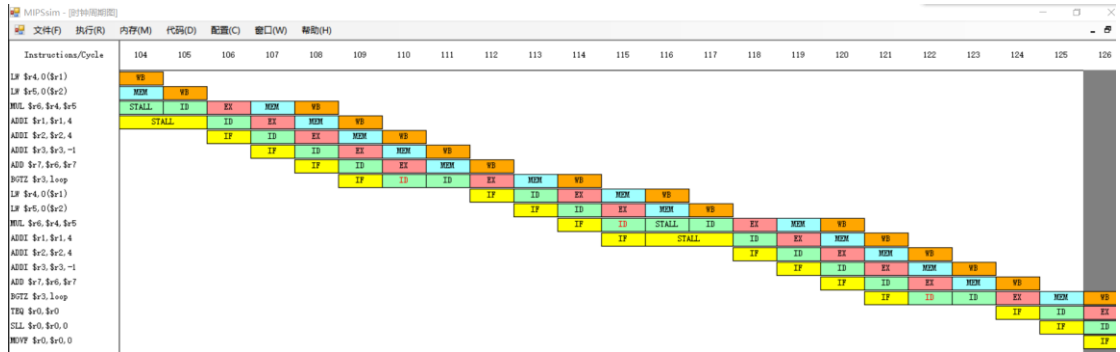
分支指令:
 指令条数: 10 占指令总数的百分比: 11.62791%
 其中:
 分支成功: 9 占分支指令数的百分比: 90%
 分支失败: 1 占分支指令数的百分比: 10%

load/store指令:
 指令条数: 20 占指令总数的百分比: 23.25581%
 其中:
 load: 20 占load/store指令数的百分比: 100%
 store: 0 占load/store指令数的百分比: 0%

浮点指令:
 指令条数: 0 占指令总数的百分比: 0%
 其中:
 加法: 0 占浮点指令数的百分比: 0%
 乘法: 0 占浮点指令数的百分比: 0%
 除法: 0 占浮点指令数的百分比: 0%

自陷指令:
 指令条数: 1 占指令总数的百分比: 1.162791%

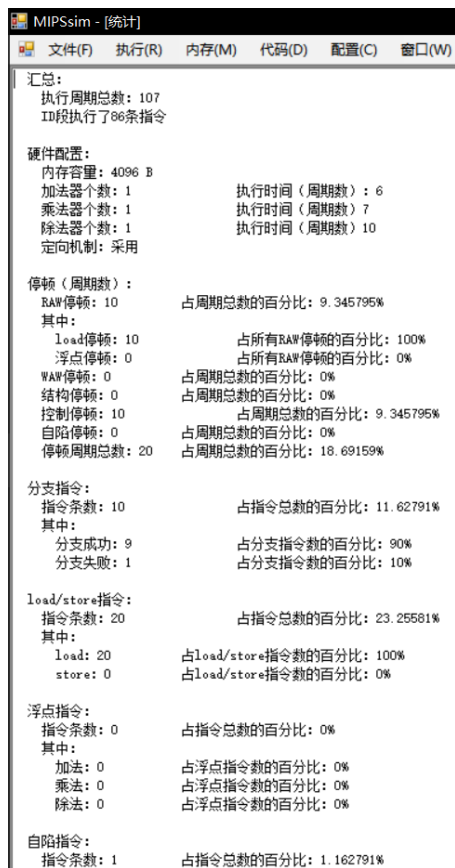
时钟周期图如下：



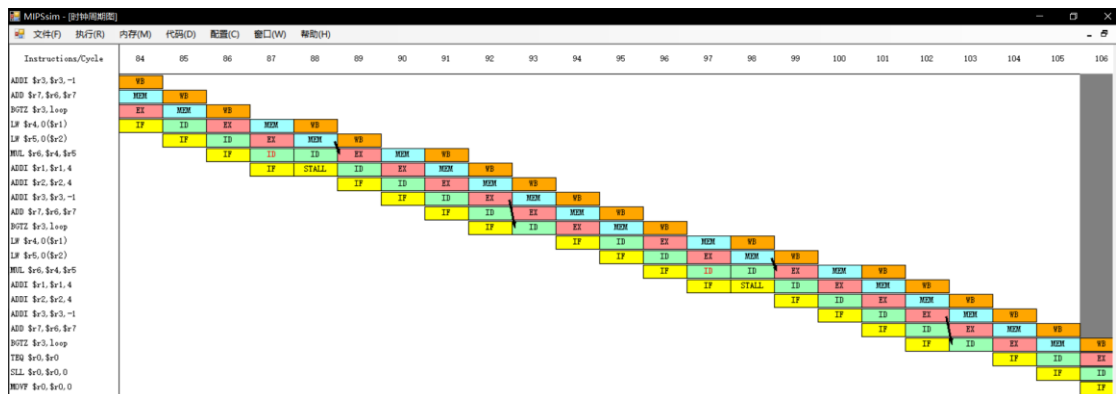
由此可见, 优化后的执行效率为优化前的 123.622%, 提高效率所减少的时钟周期均为 RAW 停顿周期。

6) 对优化后的程序使用定向功能执行, 与刚才的结果进行比较, 观察执行效率的不同。

统计情况如下:



时钟周期图如下:



由此可见，定向后的执行效率为定向前的 118.691%，提高效率所减少的时钟周期均为 RAW 停顿周期。

三、 实验结果

使用定向功能和优化代码技术，均能使 RAW 停顿周期数显著减少，从而大幅提高执行效率。

四、 实验心得

在本次实验中，遇到的主要问题有以下两点：

- ① 数组地址数据类型。起初，使用 ADDI 指令进行向量首地址的操作，编译时报错：无效的立即数。后经查阅有关资料，地址为无符号数，应使用 ADDIU 指令。修改后，该问题得到解决；
- ② 注释方式。利用 emu8086 进行代码的编写。在该集成环境中，采用“;”标志注释的开始。而对于 MIPSsim，编译无法通过。改为“#”后，该问题得到解决。

实践出真知，本次 MIPS 指令实现数组点积实验是对课堂和书本所学知识的补充。通过数据的比较，强烈感受到了定向技术和代码优化对于缓解数据冲突，提高 CPU 利用率的显著作用。平时应注重分析指令间的相关性，从而得到较优的代码。同时，通过自己动手、亲力亲为编写 MIPS 语句，加深了对汇编语言的理解和记忆，收获颇丰。