**《计算机系统结构》**

**使用MIPS指令实现冒泡排序法**



**学院：计算机学院（国家示范性软件学院）**

**班级： 2018211314**

**姓名： 李志毅**

**学号： 2018211582**

**实验四 使用MIPS指令实现冒泡排序法**

**一、实验目的**

（1）掌握静态调度方法

（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

**二、实验环境**

指令级和和流水线操作级模拟器MIPSsim

**三、实验原理**

冒泡排序算法的运作如下：

①比较相邻的元素。如果第一个比第二个大，就交换他们两个。

②对每一对相邻元素作同样的工作，从开始第一对到结尾的最后一对。在这

一点， 最后的元素应该会是最大的数。

③针对所有的元素重复以上的步骤，除了最后一个。

④持续每次对越来越少的元素重复上面的步骤，直到没有任何一对数字需要比较。

采用静态调度方法重排指令的顺序，从而使得RAW等冲突所导致的空操作得以利用，减少因为RAW等冲突而占用的无用周期，可以通过将不想关指令前移等方式来进行静态调度优化。

**四、冒泡排序代码清单及注释说明**

|  |
| --- |
| .text  main**:**  ADDIU $r1, $r0, 15 #保存数组大小  ADDIU $r2, $r0, 14 #外循环计数 i  LOOP1**:** #外循环  ADDIU $r3, $r0, array #数组array  ADDIU $r4, $r0, 0 #内循环计数 j  LOOP2**:** #内循环  LW $r5, 0($r3) # array[j]  LW $r6, 4($r3) # array[j+1]  DSUB $r7, $r5, $r6 #array[j] - array[j+1]  BLTZ $r7, bk #若array[j] < array[j+1],则跳转到bk  SW $r6, 0($r3) #若array[j] > array[j+1],则交换位置  SW $r5, 4($r3)  bk**:**  ADDIU $r4, $r4, 1 #j=j+1  ADDIU $r3, $r3, 4 #下一个数  DSUB $r8, $r2, $r4 #i - j  BGTZ $r8, LOOP2  ADDIU $r2, $r2, -1 #i=i-1  BGTZ $r2, LOOP1 #i > 0 继续外循环  TEQ $r0, $r0 #End  .data  array**:**  .word 7,34,8,22,28,49,25,44,35,14,11,4,42,3,2 |

**五、优化后的代码清单**

|  |
| --- |
| .text  main**:**  ADDIU $r1, $r0, 15 #保存数组大小  ADDIU $r2, $r0, 14 #外循环计数 i  LOOP1**:** #外循环  ADDIU $r3, $r0, array #数组array  ADDIU $r4, $r0, 0 #内循环计数 j  LOOP2**:** #内循环  LW $r5, 0($r3) # array[j]  LW $r6, 4($r3) # array[j+1]  ADDIU $r4, $r4, 1 #j=j+1  DSUB $r7, $r5, $r6 #array[j] - array[j+1]  DSUB $r8, $r2, $r4 #i - j  BLTZ $r7, bk #若array[j] < array[j+1],则跳转到bk  SW $r6, 0($r3) #若array[j] > array[j+1],则交换位置  SW $r5, 4($r3)  bk**:**  ADDIU $r3, $r3, 4 #下一个数  BGTZ $r8, LOOP2  ADDIU $r2, $r2, -1 #i=i-1  BGTZ $r2, LOOP1 #i > 0 继续外循环  TEQ $r0, $r0 #End  .data  array**:**  .word 7,34,8,22,28,49,25,44,35,14,11,4,42,3,2 |

**六、实验步骤**

**1.** **自行编写一个实现冒泡排序的汇编程序，该程序要求可以实现对一维整数数组进行冒泡排序。**

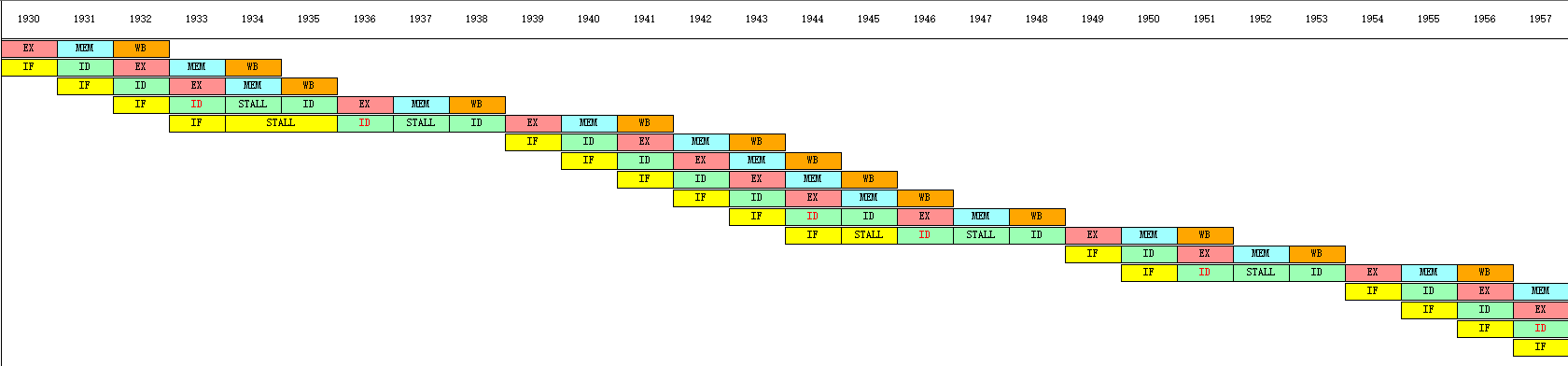
要求数组长度不得小于 10

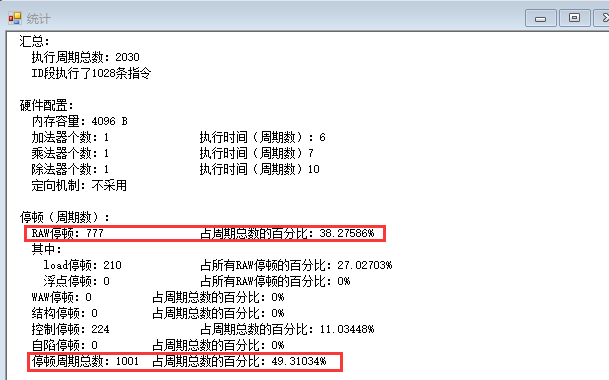
源代码见附录

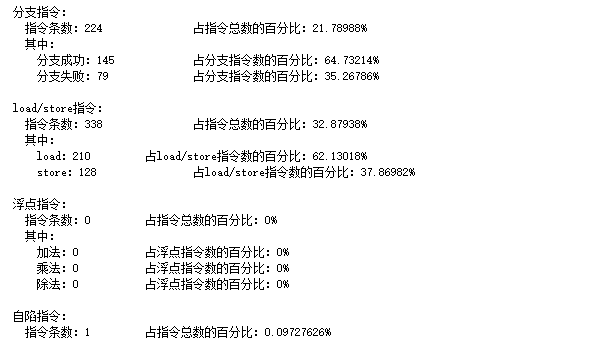
**2.启动MIPSsim**

**3. 载入自己编写的程序，观察流水线输出结果**

流水线的部分执行情况：



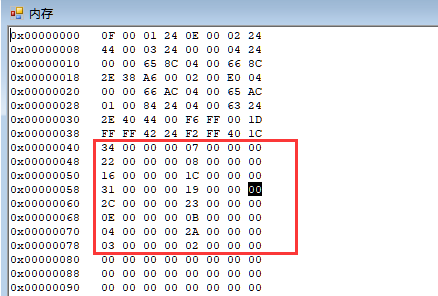




程序计算的是7,34,8,22,28,49,25,44,35,14,11,4,42,3,2的冒泡排序结果

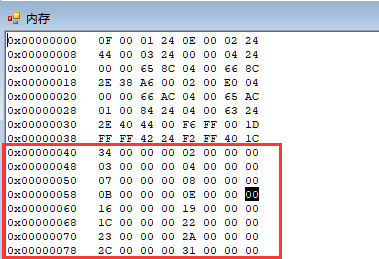
程序执行**2030**个周期，其中RAW冲突占用**777**个周期，占比**38.27586%**

程序执行前未排序的数组：

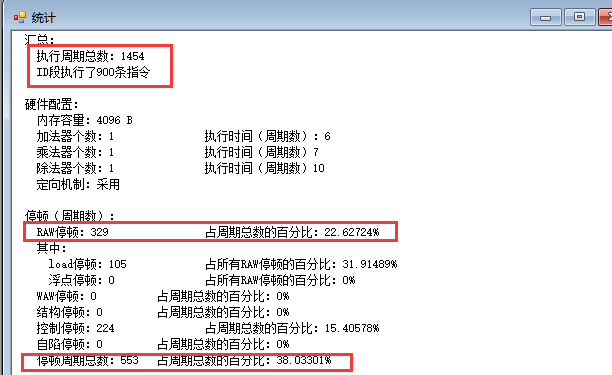


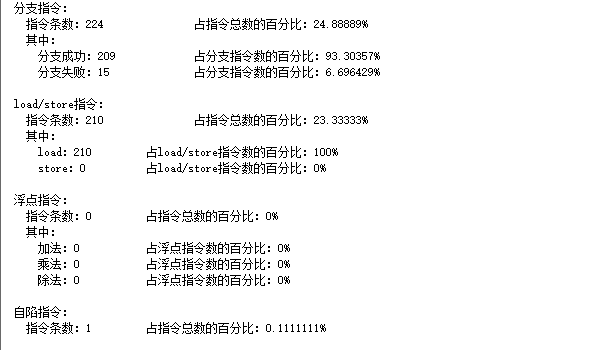
可以看到7,34,8,22,28,49,25,44,35,14,11,4,42,3,2依次存储在内存中

程序执行后排序后的数组：

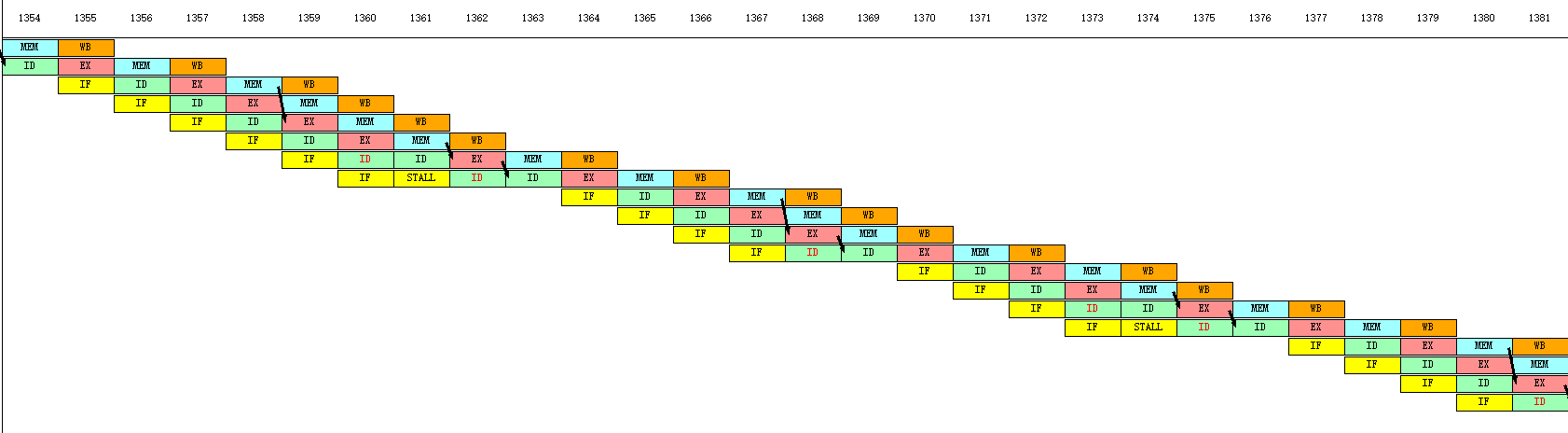


排序后的数组为2,3,4,7,8,11,14,22,25,28,34,35,42,44,49

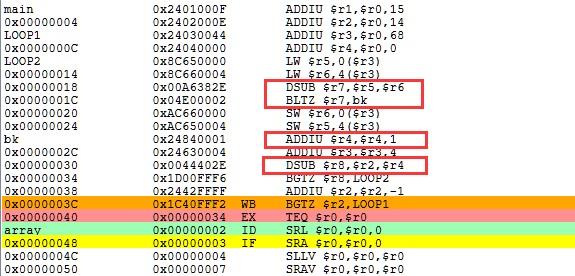
**4. 使用定向功能再次执行代码，与刚才执行结果进行比较，观察执行效率的不同。**



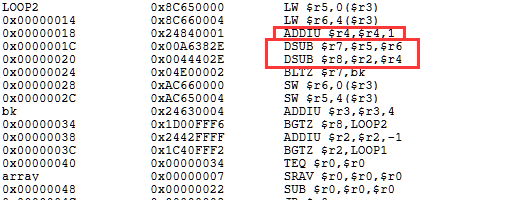
可以看到，采用定向技术后，执行总周期数变为**1454**，其中RAW停顿**329**个周期，占比为**22.62724%**，执行的效率变为原来的2030/1454=**1.39**倍



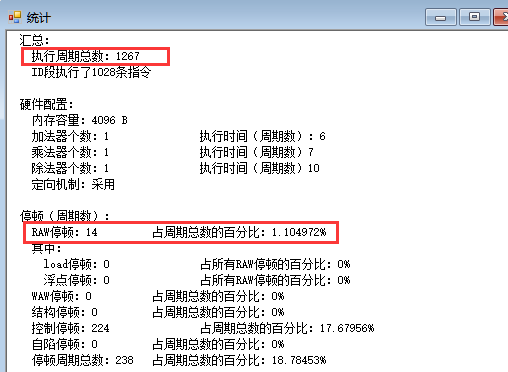
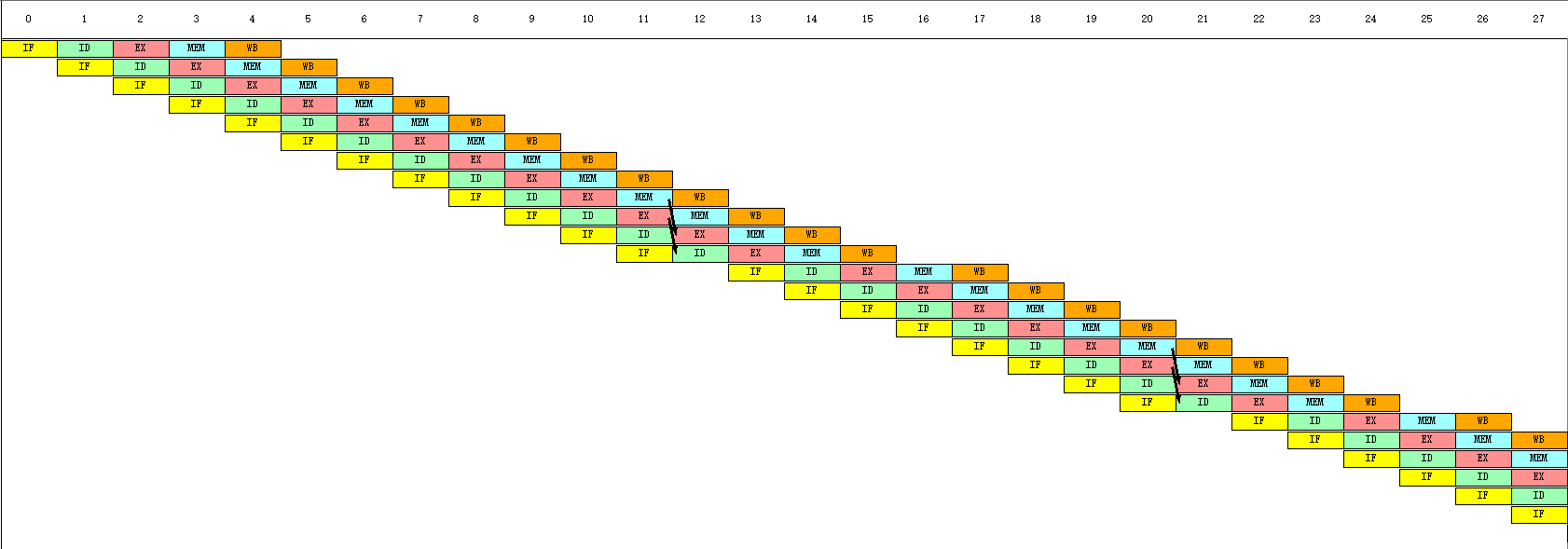
**5. 采用静态调度方法重排指令序列，减少相关，优化程序**



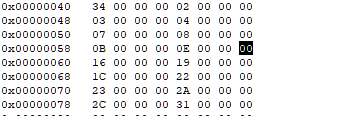
可以看到，LW $r5, 0($r3)指令和LW $r6, 4($r3)指令和DSUB $r7, $r5, $r6和BLTZ $r7, bk存在数据冲突，DSUB $r8, $r2, $r4和BGTZ $r8, LOOP2存在数据冲突，将无关指令穿插中再其中，减少RAW相关



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



可以看到采用静态调度的方法优化程序，并使用定向功能后，执行总周期数为**1267**，RAW停顿为**14**，停顿数明显减少，效率变为原来的1454/1267=**1.1476**倍。执行结果也正确。



**七、实验问题与心得**

本次实验使用指令级和流水线操作级模拟器MIPSsim分析了冒泡排序程序优化的过程，加深了我对于计算机流水线基本概念的理解，理解了MIPS结构是如何使用5段流水线来实现的，理解了各段的功能和基本操作，加深了我对数据冲突和结构冲突的理解，以及采用定向技术解决数据冲突带来的好处和性能的提升，进一步掌握了解决数据冲突的方法，掌握了如何应用定向技术来减少数据冲突引起的停顿，增强汇编语言编程能力，了解了对代码进行优化的方法。