**山东大学 计算机科学与技术 学院**

**计算机体系结构 课程实验报告**

|  |  |  |  |
| --- | --- | --- | --- |
| **学号：**201900130133 | **姓名：**施政良 | | **班级：** 四班 |
| **实验题目：** 用 WinDLX 模拟器完成求素数程序 | | | |
| **实验学时：**2 | | **实验日期：** 2022-05-05 | |
| **实验目的：**  （1）用 WinDLX 模拟器执行求素数程序 prim.s，计算若干个整数的素数。  （2） 单步执行两轮程序，求出素数 2 和 3。  （3）在执行程序过程中，注意体验单步执行除法和乘法指令的节拍数，并和主菜单 configuration/floating point slages 中的各指令执行拍数进行比较。 | | | |
| **硬件环境：**  WinDLX (基于 Windows 的 DLX 模拟器) | | | |
| **软件环境：**  Windows 7 | | | |
| **实验步骤与内容：**  **实验内容**  本次实验主要涉及素数的求解，具体的实验步骤可以划分为如下几个步骤   1. 使用WinDLX模拟器单步跟踪素数程序，执行两轮程序，求出素数2和素数3 2. 体验乘法和除法指令的节拍数，并与主菜单configuration/floating point slages中各个指令执行排数进行对比。   本次实验的具体过程如下。   1. **汇编代码分析**   素数求解的汇编代码保存在PRIM.S文件中。分析可知，该文件主要由6个段组成，分别是   * Main函数段 * NextValue代码段 * Loop循环 * IsPrim代码段 * IsNoPrim代码段 * Finish代码段   其中main函数段时整个程序的入口，并且对后续程序需要使用的寄存器进行初始化。  具体代码如下所示：   1. main: 2. ;\*\*\* Initialization 3. addi        r1,r0,0     ;Index in Table 4. addi        r2,r0,2     ;Current value   之后，定义NextValue代码段，用于在之后的IsNoPrim函数中调用。  完成主函数的初始化之后，程序开始执行Loop循环，每一次循环判断一个数是否是素数，需要调用IsPrim函数和IsNoPrim函数。   1. Loop:       seq     r4,r1,r3    ;End of Table? 2. bnez        r4,IsPrim   ;R2 is a prime number 3. lw      r5,Table(R3) 4. divu        r6,r2,r5 5. multu       r7,r6,r5 6. subu        r8,r2,r7 7. beqz        r8,IsNoPrim 8. addi        r3,r3,4 9. j       Loop   IsPrim代码段主要负责素数的判断，如果可以被某个数整除，则说明为素数。   1. IsPrim:     ;\*\*\* Write value into Table and increment index 2. sw      Table(r1),r2 3. addi        r1,r1,4 5. ;\*\*\* 'Count' reached? 6. lw      r9,Count 7. srli        r10,r1,2 8. sge     r11,r10,r9 9. bnez        r11,Finish   IsNoPrim代码段在IsPrim代码段之后被调用，主要实现变量的自增，当某个数判断为素数或非素数之后，需要跳转到IsNoPrim代码段对变量的值加一，且自增操作通过调用之前定义的NextValue代码段实现。  当程序执行完毕之后，需要进入Finish代码段，此时程序执行trap指令。代码如下所示：   1. Finish:     ;\*\*\* end 2. trap        0   **2.指令执行流程**  通过上述分析并阅读汇编代码，本实验中程序执行的流程如下图所示：    **3. 具体实验过程**  （1）导入文件    （2）单步执行，对程序进行跟踪：通过实验可以发现，随着指令的执行，首先进入main函数，之后进入Loop循环，具体展示如下：    可以看到，此时正在取出Loop代码段的第一条指令  （3）继续单步跟踪，观察寄存器的变化：可发现寄存器R2中保存着当前正在判断的数，例如下图中正处于IsPrim代码段，且寄存器R2的取值为2，说明正在判断2是否是素数。    （4）调用NextValue代码段：判断当前的输入之后，需要调用NextValue，将变量的值自增，从而在下一个Loop循环时判断下一个数是不是素数。如下图所示，当前程序正在执行NextValue的第一条指令。    （红色表示执行部件正在执行的指令）  （5）再一次循环，得到素数3：继续按下F7单步执行程序，知道观察到coding窗口再次进入Loop循环，并执行IsPrim代码段。    （6）重复上述操作：观察寄存器R2的取值变化：重复上述操作，可以观察到R2中的值从2开始递增，两轮循环之后可以得到素数2和素数3.  （7）观察指令节拍：在单步执行中，除法节拍数为19，乘法节拍数为5，与configuration/floating point slages中的各指令执行拍数一致。具体信息如下图所示： | | | |
| **结论分析与体会：**  **结论分析**  1.分析本次实验中是否出现流水线相关（流水线冒险）的问题  分析：  本次之实验中指令的执行也存在数据相关问题，具体如下图所示。    可以看到，红色箭头表示需要一个暂 停，箭头指向处显示了暂停的原因。R-Stall（R-暂停）表示引起暂停的原因是 RAW。绿色箭头 表示定向技术的使用。  定向技术的主要思想是：在某条指令（如图 3.3.4 中的 ADD 指令）产生一个计算结果之前，其 它指令并不真正需要该计算结果，如果能够将该计算结果从其产生的地方（寄存器文件 EX/MEM）直接送到其它指令需要它的地方，那么就可以避免暂停。  基于上述分析，定向技术的要点可以归纳为：  (1) 寄存器文件 EX/MEM 中的 ALU 的运算结果总是回送到 ALU 的输入寄存器。  (2) 当定向硬件检测到前一个 ALU 运算结果的写入寄存器就是当前 ALU 操作的源寄存器 时，那么控制逻辑将前一个 ALU 运算结果定向到 ALU 的输入端，后一个 ALU 操作就不必从 源寄存器中读取操作数。  2. 指令节拍的分析  分析：  观察指令节拍：在单步执行中，除法节拍数为19，与configuration/floating point slages中的各指令执行拍数一致。    同理，乘法节拍数为5，与configuration/floating point slages中的各指令执行也拍数一致。  **体会**  通过本次实验我进一步掌握了WinDLX模拟器的基本操作和使用，熟悉了WinDLX五段流水线在执行具体程序时的流水情况，更加深入的了解计算机系统流水线的工作过程。同时，通过单步跟踪指令的执行以及分析指令节拍，我熟悉DLX指令集结构及其特点，了解了乘法和除法在流水线中的节拍数。  另外，在本次实验中指令的执行也存在数据相关问题。 | | | |

**附录**

1. ;\*\*\*\*\*\*\*\*\*\*\* WINDLX Exp.2: Generate prime number table \*\*\*\*\*\*\*\*\*\*\*\*\*
2. ;\*\*\*\*\*\*\*\*\*\*\* (c) 1991 G黱ther Raidl             \*\*\*\*\*\*\*\*\*\*\*\*\*
3. ;\*\*\*\*\*\*\*\*\*\*\* Modified 1992 Maziar Khosravipour         \*\*\*\*\*\*\*\*\*\*\*\*\*
5. ;-------------------------------------------------------------------
6. ; Program begins at symbol main
7. ; generates a table with the first 'Count' prime numbers from 'Table'
8. ;-------------------------------------------------------------------
10. .data
12. ;\*\*\* size of table
13. .global     Count
14. Count:      .word       10
15. .global     Table
16. Table:      .space      Count\*4

19. .text
20. .global main
21. main:
22. ;\*\*\* Initialization
23. addi        r1,r0,0     ;Index in Table
24. addi        r2,r0,2     ;Current value
26. ;\*\*\* Determine, **if** R2 can be divided by a value in table
27. NextValue:  addi        r3,r0,0     ;Helpindex in Table
28. Loop:       seq     r4,r1,r3    ;End of Table?
29. bnez        r4,IsPrim   ;R2 is a prime number
30. lw      r5,Table(R3)
31. divu        r6,r2,r5
32. multu       r7,r6,r5
33. subu        r8,r2,r7
34. beqz        r8,IsNoPrim
35. addi        r3,r3,4
36. j       Loop
38. IsPrim:     ;\*\*\* Write value into Table and increment index
39. sw      Table(r1),r2
40. addi        r1,r1,4
42. ;\*\*\* 'Count' reached?
43. lw      r9,Count
44. srli        r10,r1,2
45. sge     r11,r10,r9
46. bnez        r11,Finish
48. IsNoPrim:   ;\*\*\* Check next value
49. addi        r2,r2,1     ;increment R2
50. j       NextValue
52. Finish:     ;\*\*\* end
53. trap        0