汇编与接口课程设计 模块一

1120141831 朴泉宇

1. 实验目的
2. 掌握MIPS 体系结构与指令系统。
3. 熟悉MIPS 指令格式以及汇编指令助记符。
4. 掌握MIPS 汇编语言与机器语言之间的对应关系。
5. 学习MIPS 汇编程序设计。
6. 了解使用MIPS 汇编模拟器的方法。
7. 完成简单算法汇编程序实现实验（排序等）。
8. 实验原理
   1. MIPS 指令集架构参见附录，MIPS 汇编主要基本格式：
      1. .text //代码段（指令段）
      2. .data //数据段 （本次实验可以不适用数据段）
      3. .globl //全局符号声明
      4. .align 0 //关闭所有的自动对齐
      5. .asciiz //字符串（带终止符）
      6. 跳转标记格式如“lable：”，为标记名+冒号
   2. MARS 是MIPS 汇编语言的模拟机，由密苏里州立大学开发，最近的版本是2014年更新的4.5。 备注：可以使用其来编写并调试MIPS 汇编程序，将MIPS 汇编程序转换成机器码，供《计算机组成原理课程设计》测试用。其下载地址是www.cs.missouristate.edu/MARS。使用方法见：MARS Tutorial.doc。
9. 实验内容
10. 对实验素材Fibonacci数列进行测试并记录。
11. 实现一个排序算法，对输入的数进行排序。
12. 实验过程
13. 对实验素材Fibonacci数列进行测试并记录

输入n：8

结果：21

寄存器号 内容 寄存器号 内容 寄存器号 内容

$0 0x00000000 $11 0x00000000 $22 0x00000000

$1 0x10010000 $12 0x00000000 $23 0x00000000

$2 0x00000001 $13 0x00000000 $24 0x00000000

$3 0x00000000 $14 0x00000000 $25 0x00000000

$4 0x00000015 $15 0x00000000 $26 0x00000000

$5 0x00000000 $16 0x00000000 $27 0x00000000

$6 0x00000000 $17 0x00000000 $28 0x10008000

$7 0x00000000 $18 0x00000000 $29 0x7fffeffc

$8 0x0000000d $19 0x00000000 $30 0x00000000

$9 0x00000015 $20 0x00000000 $31 0x00000000

$10 0x00000000 $21 0x00000000

1. 实现一个排序算法，对输入的数进行排序

代码见下，已用注释将代码全部解释：

1. ##### Bubble Sort #####
2. ##### ShYy, at 2017.09.15 #####
3. .text
4. .globl main
5. main:
6. la $t0, array # t0 is the array's addr. Const.
7. move $t1, $zero # t1 is looping int i.
8. move $t2, $zero # t2 is looping int j.
9. li $t3, 5 # t3 is the length of the array. In this program, is 5. Const.
11. li $v0, 4 # Ready to input. Print the msg1.
12. la $a0, msg1
13. syscall
14. input:
15. li $v0, 5 # Syscall and get the number at v0.
16. syscall
18. move $t4, $t1 # Like array operation in C, array[i \* 4(Byte)] is the number. $4 is i \* 4.
19. mul $t4, $t4, 4 # i \* 4
20. addu $t5, $t4, $t0 # array[i \* 4]
21. sw $v0, 0($t5)
23. addi $t1, $t1, 1 # i++
24. blt $t1, $t3, input # If i < n, goto input, loop.
26. move $t1, $zero # Initial, i = 0.
27. loop1:
28. move $t2, $zero # for(i = 0; i < n - 1; i++) -> for(j = 0; j < n - i - 1; j++)
29. loop2:
30. move $t4, $t2 # Get $t6 = array[j \* 4(Byte)].
31. mul $t4, $t4, 4
32. addu $t8, $t4, $t0 # &array[j \* 4] = $t8
33. lw $t6, 0($t8)
35. addi $t4, $t2, 1 # Get $t7 = array[(j + 1) \* 4(Byte)]
36. mul $t4, $t4, 4
37. addu $t9, $t4, $t0 # &array[(j + 1) \* 4] = $t9
38. lw $t7, 0($t9)
40. blt $t6, $t7, continue # If array[j] < array[j + 1], continue.
41. sw $t6, 0($t9) # Else Swap two numbers.
42. sw $t7, 0($t8)
44. continue:
45. addi $t2, $t2, 1 # j++
46. addi $t4, $t1, 1 # In j < n - i - 1, firstly (i + 1).
47. sub $t5, $t3, $t4 # n - i - 1
48. blt $t2, $t5, loop2 # If j < n - i - 1, continue.
50. addi $t1, $t1, 1 # Else go loop1, i++.
51. move $t4, $zero # Set $t4 = 0.
52. addi $t4, $t4, 1 # Set $t4 = 1.
53. sub $t5, $t3, $t4 # In i < n - 1, solve n - 1.
54. blt $t1, $t5, loop1 # If i < n - 1, continue.
56. li $v0, 4 # Ready to print. Print msg2.
57. la $a0, msg2
58. syscall
59. move $t1, $zero # Else Set i = 0, used to print the array.
60. print:
61. move $t4, $t1 # Get array[i \* 4(Byte)].
62. mul $t4, $t4, 4
63. addu $t5, $t4, $t0
64. lw $a0, 0($t5) # Set print's parameter, array[i \* 4].
65. li $v0, 1
66. syscall # Print the number.
67. la $a0, sepr # Print the space between two number.
68. li $v0, 4
69. syscall
70. addi $t1, $t1, 1 # i++
71. blt $t1, $t3, print # Print loop.

74. .data
75. msg1: .asciiz "Please input 5 numbers. Seperate by Enter.\n"
76. msg2: .asciiz "The sorted array is:\n"
77. sepr: .asciiz " "
78. array: .space 1024
79. 心得体会

这次实验，在之前x86汇编语言的基础上，学习了MIPS汇编的语法、寄存器、调用方法等，并且实现了冒泡排序算法。在这个过程中，不单单是MIPS，还有对RISC的更多了解。在接下来的计算机组成原理实验中，将更底层地去从CPU实现MIPS的代码，进而系统地学习计算机的底层设计。