**中山大学数据科学与计算机学院本科生实验报告**

**（2019学年秋季学期）**

课程名称：**计算机组成原理实验**  任课教师：郭雪梅 助教：汪庭葳、刘洋旗

|  |  |  |  |
| --- | --- | --- | --- |
| 年级&班级 | **2018级1班** | 专业(方向) | **计算机类** |
| 学号 | **18308045** | 姓名 | **谷正阳** |
| 电话 | **13355426001** | Email | [**Guzy0324@163.com**](mailto:Guzy0324@163.com) |
| 开始日期 | **2019.10.9** | 完成日期 | **2019.10.9** |

**一、实验题目**

计算机结构与组成 实验2

1. **实验目的**

熟悉MARS，编一个简短的MIPS程序

**三、实验内容**

**1.** 实验步骤

练习 1

假定你想编写一个MIPS程序foo，该程序使用5个字的数组，数组元素初始化为整数1, ..., 5.

.data

foo: .word 1,2,3,4,5

你用程序来把数组foo中的每个数加2再写回数组foo

main:

addi $t0, $0, foo

lw $t1,0($t0)

addiu $t1, $t1,2

sw $t1, 0($t0)

.....

你也可以使用循环来完成。

回答下列关于MARS的问题.

a. .data, .word, .text 指示器（directives）的含义是什么(即, 在每段中放入什么内容)?

b. 在MARS中如何设置断点breakpoint?

c. 在程序运行到断点处停止时，如何继续执行? 如何单步调试代码?

d. 如何知道某个寄存器register的值是多少? 如何修改寄存器的值.

将你的代码存储到文件lab\_ex.s 中，然后给老师检查.

练习2

从键盘输入两个数，计算并输出这两个数的和

.data

str1: .asciiz "Enter 2 numbers:"

str2: .asciiz "The sum is "

.text

main:

ori $v0, $0, 4 #System call code 4 for printing a string

la $a0, Str1 #address of Str1 is in $a0

syscall #print the string

ori $v0, $0, 5 #System call code 5 for read integer，$v0 contains integer read

add $t0, $v0, $zero #

….

ori $v0, $0, 4 #System call code 4 for printing a string

la $a0, Str2 #address of Str2 is in $a0

syscall #print the string

ori $v0, $0, 1 #System call code 4 for print integer，$a0 = integer to print

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ #calculate the sum

syscall #print the sum

exit: ori $v0, $0, 10 #System call code 10 for exit

syscall #print the sum

3. 计算12+22+…+1002 , 参照练习2输出结果

.data

.align 2

Str: .asciiz "The sum of square from 1 to 100 is "

4. 编写两个版本的first1pos (starting from first1pos.s) 函数, 在$a0中给定一个数, 而在$v0 中返回$a0字中最左边的非零位的位置. 如果$a0 的值是0, 在$v0 中存-1 . 在查找此位置的过程中,允许你修改$a0 值.位置从0 (最右位)到31 (符号位). 其中一种解应该重复移位$a0, 每次移位后,检查符号位. 另一种方法是初始时使用0x80000000 作为掩码,并不断右移该掩码来检查$a0的每一位.

可以和你的同伴分别来做此工作, 其中一个同学做第一个版本,另一个同学做第二个版本. 工作完成后,向你的同伴解释你的代码, 然后把运行情况给老师检查.

first1pos.s

main:

lui $a0,0x8000

jal first1pos

jal printv0

lui $a0,0x0001

jal first1pos

jal printv0

li $a0,1

jal first1pos

jal printv0

add $a0,$0,$0

jal first1pos

jal printv0

li $v0,10

syscall

first1pos: # your code goes here

printv0:

addi $sp,$sp,-4

sw $ra,0($sp)

add $a0,$v0,$0

li $v0,1

syscall

li $v0,11

li $a0,'\n'

syscall

lw $ra,0($sp)

addi $sp,$sp,4

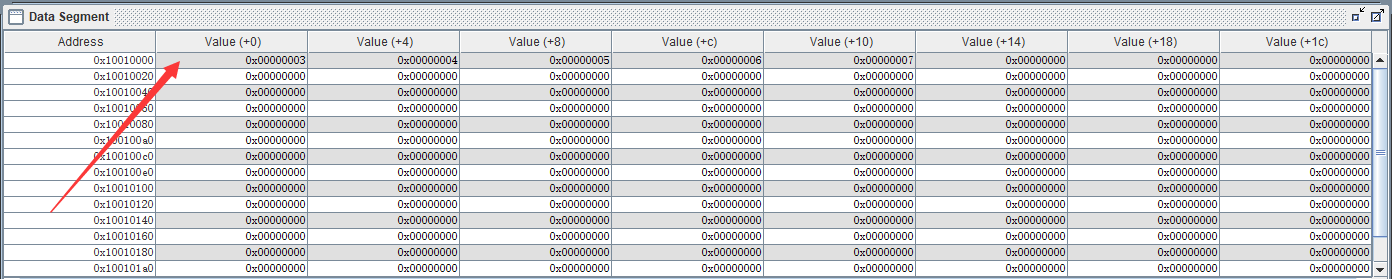
jr $ra

**2.** 实验原理

见上

**四、实验结果**

练习1：

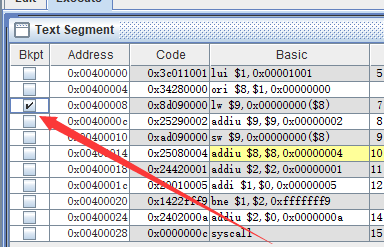


a. .data: 定义用户数据段

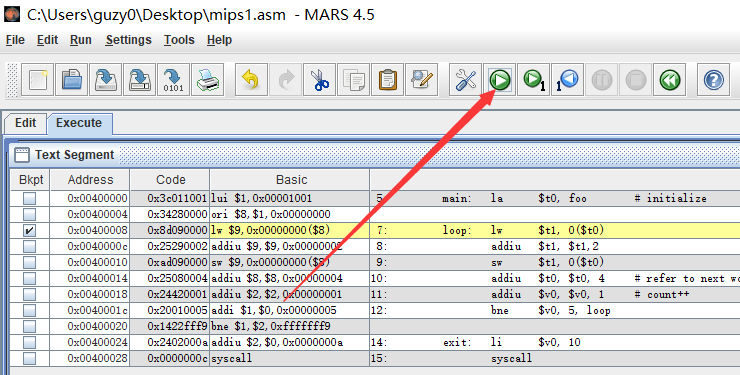
.word: 在内存中连续存放n个字

.text: 定义用户代码段

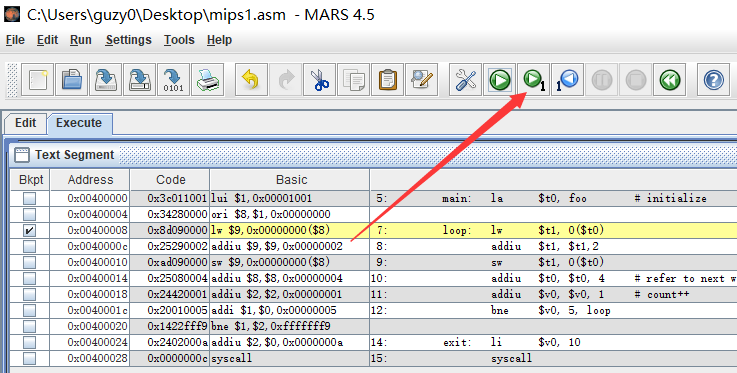
b.



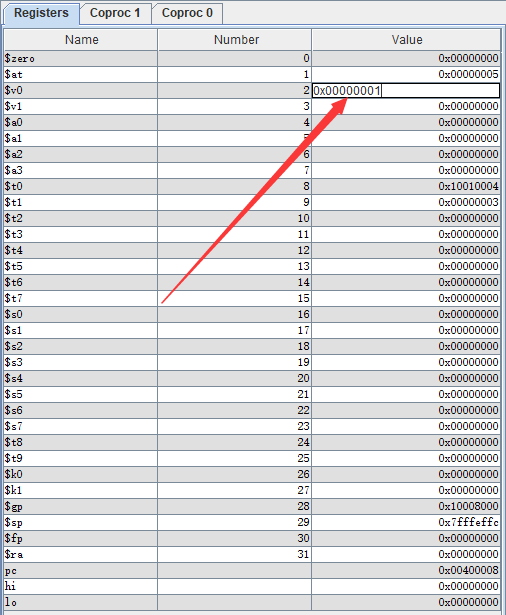
c. 继续执行：



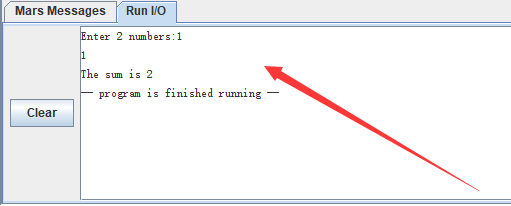
单步执行：



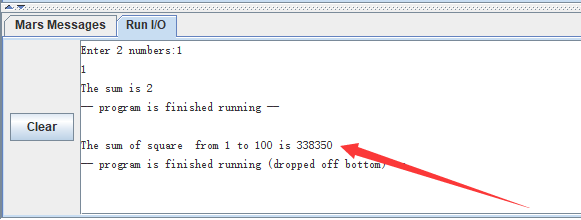
d.



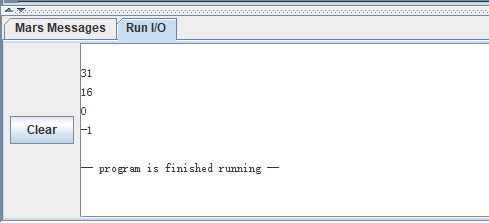
练习2：



练习3：



练习4：



**五、实验感想**

左移+掩码实现寻找第一个1位置的两种算法都很巧妙，可以解决如判断奇偶数的问题。

**附录（流程图，注释过的代码）：**

练习1：

.data

foo: .word 1,2,3,4,5

.text

main: la $t0, foo # initialize

loop: lw $t1, 0($t0)

addiu $t1, $t1,2

sw $t1, 0($t0)

addiu $t0, $t0, 4 # refer to next word

addiu $v0, $v0, 1 # count++

bne $v0, 5, loop

exit: li $v0, 10

syscall

练习2：

.data

Str1: .asciiz "Enter 2 numbers:"

Str2: .asciiz "The sum is "

.text

main:

ori $v0, $0, 4 #System call code 4 for printing a string

la $a0, Str1 #address of Str1 is in $a0

syscall #print the string

ori $v0, $0, 5 #System call code 5 for read integer，$v0 contains integer read

syscall #read the first integer

add $t0, $v0, $zero #store the first integer temporarily

ori $v0, $0, 5 #System call code 5 for read integer，$v0 contains integer read

syscall #read the second integer

add $t1, $v0, $zero #store the second integer temporarily

ori $v0, $0, 4 #System call code 4 for printing a string

la $a0, Str2 #address of Str2 is in $a0

syscall #print the string

ori $v0, $0, 1 #System call code 4 for print integer，$a0 = integer to print

add $a0, $t0, $t1 #calculate the sum

syscall #print the sum

exit: ori $v0, $0, 10 #System call code 10 for exit

syscall #print the sum

练习3：

.data

.align 2

Str: .asciiz "The sum of square from 1 to 100 is "

.text

main: addiu $t0, $zero, 1 #initialize

mulu $t1, $t0, $t0

addu $t2, $zero, $t1

loop: addiu $t0, $t0, 1

mulu $t1, $t0, $t0 #calculate the square of $t0

addu $t2, $t2, $t1 #add the suare to the sum

bne $t0, 100, loop

print: la $a0, Str #load the Str

li $v0, 4 #System call code 4 for printing a string

syscall

li $v0, 1 #System call code 1 for print integer

move $a0, $t2

syscall

exit: li $v0, 10

syscall

练习4：

version1：

main:

lui $a0,0x8000

jal first1pos

jal printv0

lui $a0,0x0001

jal first1pos

jal printv0

li $a0,1

jal first1pos

jal printv0

add $a0,$0,$0

jal first1pos

jal printv0

li $v0,10

syscall

first1pos: addi $sp, $sp, -12

sw $a0, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

addi $t0, $zero, 31 #index = 31

loop: beq $t0, -1, looped

slt $t1, $a0, $zero

beq $t1, 1, looped #while (index != -1 && a0 >= 0)

sll $a0, $a0, 1 #a0<<

addi $t0, $t0, -1 #index--

j loop

looped: move $v0, $t0

lw $a0, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

addi $sp, $sp, 12

jr $ra #return index

printv0:

addi $sp,$sp,-4

sw $ra,0($sp)

add $a0,$v0,$0

li $v0,1

syscall

li $v0,11

li $a0,'\n'

syscall

lw $ra,0($sp)

addi $sp,$sp,4

jr $ra

version2：

first1pos: addi $sp, $sp, -12

sw $a0, 0($sp)

sw $t0, 4($sp)

sw $t1, 8($sp)

addi $t0, $zero, 31 #index = 31

loop: beq $t0, -1, looped

andi $t1, $a0, 0x80000000

beq $t1, 0x80000000, looped #while (index != -1 && (a0 & 0x80000000) != 0x80000000)

sll $a0, $a0, 1 #a0<<

addi $t0, $t0, -1 #index--

j loop

looped: move $v0, $t0

lw $a0, 0($sp)

lw $t0, 4($sp)

lw $t1, 8($sp)

addi $sp, $sp, 12

jr $ra #return index