**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：丁文、汪庭葳

|  |  |  |  |
| --- | --- | --- | --- |
| 年级&班级 | **2019级04班** | 专业(方向) | **计算机科学与技术（超算方向）** |
| 学号 | **19335112** | 姓名 | **李钰** |
| 电话 | **19847352856** | Email | **1643589912@qq.com** |
| 开始日期 | **2020.10.16** | 完成日期 | **2020.10.17** |

**一、实验题目**

实验六 MIPS汇编程序设计

1. **实验目的**

1.熟悉MIPS汇编程序开发环境，学习使用MARs工具。知道如何查看内存空间分配。

2.了解C语言语句与汇编指令之间的关系。

3. 掌握MIPS汇编程序设计，掌握MARs的调试技术。

4.了解MIPS汇编语言与机器语言之间的对应关系 。

5.熟悉常见的MIPS汇编指令

6.掌握程序的内存映像。

熟悉MIPS汇编程序开发环境，学习使用MARs工具。知道如何查看内存空间分配

**三、实验内容**

1. 编写 swap 函数实现两个数之间的交换

swap 函数

swap:

move $fp, $sp #FRAME POINTER NOW POINTS TO THE TOP OF STACK

addiu $sp,$sp,-16 # ALLOCATE 16 BYTES IN THE STACK

lw $t0 0($a0) #a0中存着n1的地址，将该地址处的数存到$t0中

sw $t0 0($sp) #再将$t0中的数写入到栈里

lw $t0 0($a1) #同理

sw $t0 0($a0) #将$t里的数直接写到$a0中

lw $t0 0($sp) #出栈

sw $t0 0($a1)

addiu $sp,$sp,16

jr $31

2. 用汇编程序实现以下伪代码， 要求使用移位指令实现乘除法运算

Int main ()

{

Int K,Y;

Int Z[50];

Y=56;

For(k=0;k<50;K++) Z[k]=Y-16\*(k/4+210);

}

1.C语言分析：

有两个变量是int型，一个数组型；还有一个循环执行过程。

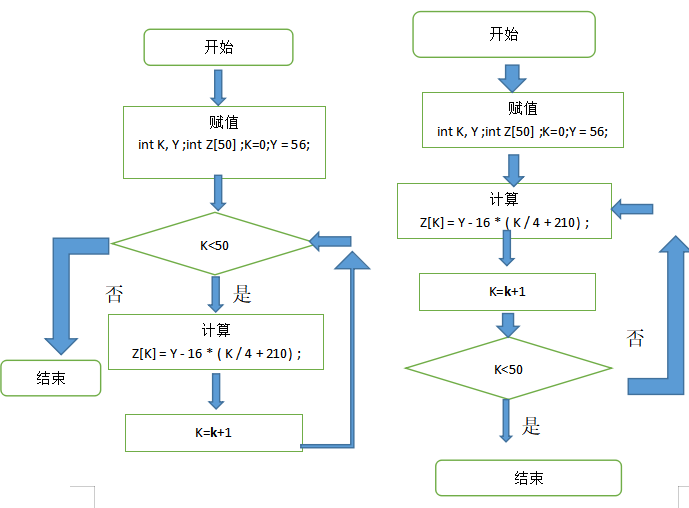
2.汇编程序实现分析：

首先需要定义用户数据段，获得一个内存空间作为数组空间。再选定几个寄存器作为K，Y以及输出，其中输出输出和Y可以合用一个寄存器。

3.设计思路：

分配完空间地址后，最重要的是完成循环控制。循环控制有两个思路：可以是先判断后循环；或者是先循环后判断

即如图



4.程序实现及调试分析

1. 汇编程序代码实现：

**<方法一>**

.data

z: .space 50 #为数组z申请50个内存空间

.text

main:

la $t0, z #将数组的地址写入$t0

li $s0, 0 #s0 = 0 ,用于计数 即 k =0

li $s1, 56 #s1 = 56 即Y

loop:

slti $t1, $s0, 50 #若 $s0 < 50 则 $t1 = 1 ，循环继续

bne $t1, 1, end #若 $t1 != 1 则循环结束

srl $t6, $s0, 2 #k/4

addi $t6, $t6, 210 #k/4 + 210

# mul $t9, $t9, 16

sll $t6, $t6, 4 #乘以16相当于左移4位 16 \*(k/4 + 210)

sub $t6, $s1, $t6 # 56 - 16 \*(k/4 + 210)

sw $t6 0($t0) #将$t6 存入当前数组空的位置

addu $t0, $t0, 4 # 将数组地址加上了偏移量t0，得到下一个地址

addi $s0, $s0, 1 #k++

j loop

end:

**<方法二>**

.data

z: .space 50 #为数组z申请50个内存空间

.text

main:

la $t0, z #将数组的地址写入$t0

li $s0, 0 #s0 = 0 ,用于计数 即 k =0

li $s1, 56 #s1 = 56 即Y

loop:

srl $t6, $s0, 2 #k/4

addi $t6, $t6, 210 #k/4 + 210

sll $t6, $t6, 4 #乘以16相当于左移4位 16 \*(k/4 + 210)

sub $t6, $s1, $t6 # 56 - 16 \*(k/4 + 210)

sw $t6 0($t0) #将$t6 存入当前数组空的位置

addu $t0, $t0, 4 # 将数组地址加上了偏移量t0，得到下一个地址

addi $s0, $s0, 1 #k++

slti $t1, $s0, 50 #若 $s0 < 50 则 $t1 = 1 ，循环继续

beq $t1, 1, loop

1. 调试过程

编写程序：详细见代码

装载程序

如果没有错误，便运行。

运行之后点击不同的窗口便可得到我们想要的结果。具体详细结果如下图

内存占用情况映像

分析：数组地址

数据段内存映像

表格如下（数值都采用16进制）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 内存地址（16进制） | 变量名 | 值 | 内存地址（16进制） | 变量名 | 值 |
| **0x10010000** | **Z[0]** | **0xfffff318** | **0x10010064** | **Z[25]** | **0xfffff2b8** |
| **0x10010004** | **Z[1]** | **0xfffff318** | **0x10010068** | **Z[26]** | **0xfffff2b8** |
| **0x10010008** | **Z[2]** | **0xfffff318** | **0x1001006c** | **Z[27]** | **0xfffff2b8** |
| **0x1001000c** | **Z[3]** | **0xfffff318** | **0x10010070** | **Z[28]** | **0xfffff2a8** |
| **0x10010010** | **Z[4]** | **0xfffff308** | **0x10010074** | **Z[29]** | **0xfffff2a8** |
| **0x10010014** | **Z[5]** | **0xfffff308** | **0x10010078** | **Z[30]** | **0xfffff2a8** |
| **0x10010018** | **Z[6]** | **0xfffff308** | **0x1001007c** | **Z[31]** | **0xfffff2a8** |
| **0x1001001c** | **Z[7]** | **0xfffff308** | **0x10010080** | **Z[32]** | **0xfffff298** |
| **0x10010020** | **Z[8]** | **0xfffff2f8** | **0x10010084** | **Z[33]** | **0xfffff298** |
| **0x10010024** | **Z[9]** | **0xfffff2f8** | **0x10010088** | **Z[34]** | **0xfffff298** |
| **0x10010028** | **Z[10]** | **0xfffff2f8** | **0x1001008c** | **Z[35]** | **0xfffff298** |
| **0x1001002c** | **Z[11]** | **0xfffff2f8** | **0x10010090** | **Z[36]** | **0xfffff288** |
| **0x10010030** | **Z[12]** | **0xfffff2e8** | **0x10010094** | **Z[37]** | **0xfffff288** |
| **0x10010034** | **Z[13]** | **0xfffff2e8** | **0x10010098** | **Z[38]** | **0xfffff288** |
| **0x10010038** | **Z[14]** | **0xfffff2e8** | **0x1001009c** | **Z[39]** | **0xfffff288** |
| **0x1001003c** | **Z[15]** | **0xfffff2e8** | **0x100100a0** | **Z[40]** | **0xfffff278** |
| **0x10010040** | **Z[16]** | **0xfffff2d8** | **0x100100a4** | **Z[41]** | **0xfffff278** |
| **0x10010044** | **Z[17]** | **0xfffff2d8** | **0x100100a8** | **Z[42]** | **0xfffff278** |
| **0x10010048** | **Z[18]** | **0xfffff2d8** | **0x100100ac** | **Z[43]** | **0xfffff278** |
| **0x1001004c** | **Z[19]** | **0xfffff2d8** | **0x100100b0** | **Z[44]** | **0xfffff268** |
| **0x10010050** | **Z[20]** | **0xfffff2c8** | **0x100100b4** | **Z[45]** | **0xfffff268** |
| **0x10010054** | **Z[21]** | **0xfffff2c8** | **0x100100b8** | **Z[46]** | **0xfffff268** |
| **0x10010058** | **Z[22]** | **0xfffff2c8** | **0x100100bc** | **Z[47]** | **0xfffff268** |
| **0x1001005c** | **Z[23]** | **0xfffff2c8** | **0x100100c0** | **Z[48]** | **0xfffff258** |
| **0x10010060** | **Z[24]** | **0xfffff2b8** | **0x100100c4** | **Z[49]** | **0xfffff258** |

地址 机器码 汇编指令

[00400014] 0c100009  **jal 0x00400024 [main]**

[00400018] 00000000  **nop**

[0040001c] 3402000a  **ori $2, $0, 10**            
[00400020] 0000000c  **syscall**                  
[00400024] 3c101001  **lui $16, 4097 [z]**

1. **实验感想**

**通过这次实验，学会了堆栈的使用以及为数组赋值的方法，理解了二者之间的不同之处，编写代码的过程中要格外注意取地址还是取值。**