

# 计算机组成原理 实验报告

姓名: 雷雨轩 学号: PB18111791 实验日期: 2020-4-6

## 实验题目

MIPS排序实验

## 实验平台

MARS

## 实验过程

- 阅读相关文档(MARS官方文档, 特别是syscall使用文档)以及熟悉常用MIPS指令后, 在C语言的冒泡排序基础上完成了MIPS的代码
- 需注意的几个点:
  - MIPS汇编格式: `.data` 和 `.text`
  - C语言循环与汇编的对应:

```
for(i=0; i<n-1; i++){ //比较次数为n-1
    for(j=0; j<n-1-i; j++){
        if(a[j]>a[j+1]){
            temp=a[j];
            a[j]=a[j+1];
            a[j+1]=temp;
        }
    }
}
```

则为MIPS寄存器的数据存放做如下规定:

t0:i; t1:j; t2:n-1(n为排序总数); t3:a[j]; t4:a[j+1]; t5:输出时间的临时寄存器;  
t6:n-1-i; t7:当前参加比较的两数据的起始地址; t8:偏移;  
t9:数据基地址(默认人工把数据放入以0x10010100开始的内存地址中, 一个数据占一个字)

- syscall使用: 用于打印输出字符串, 整数, 记录程序运行时间

实验中数据手动设置在内存之后, 程序运行时会先打印请求输入排序总数, 然后程序执行过程中在进入外循环前后所有循环结束后各会记录一次当前时间并以10进制输出到Run I/O: 先高32位后低32位(中间用空格隔开)

## 实验结果

- 代码加载并设置好数据后(数据见下方0x10010100处), 此处设置了5个数据

Text Segment					Registers		
Bkpt	Address	Code	Basic	Source	Name	Number	Value
<input type="checkbox"/>	0x00400000	0x24020004	addiu \$2,\$0,0x00000004	7: main: li \$v0,4 #print string	\$zero	0	0x00000000
<input type="checkbox"/>	0x00400004	0x3c011001	lui \$1,0x00001001	8: la \$a0,buffer	\$at	1	0x00000000
<input type="checkbox"/>	0x00400008	0x34240000	ori \$4,\$1,0x00000000		\$v0	2	0x00000000
<input type="checkbox"/>	0x0040000c	0x0000000c	syscall	9: syscall	\$v1	3	0x00000000
<input type="checkbox"/>	0x00400010	0x24020005	addiu \$2,\$0,0x00000005	10: li \$v0,5	\$a0	4	0x00000000
<input type="checkbox"/>	0x00400014	0x0000000c	syscall	11: syscall	\$a1	5	0x00000000
<input type="checkbox"/>	0x00400018	0x00025021	addu \$10,\$0,\$2	12: move \$t2,\$v0	\$a2	6	0x00000000
<input type="checkbox"/>	0x0040001c	0x214affff	addi \$10,\$10,0xffffffffff	13: addi \$t2,\$t2,-1	\$a3	7	0x00000000
<input type="checkbox"/>	0x00400020	0x2408ffff	addiu \$8,\$0,0xffffffffff	15: li \$t0,-1	\$t0	8	0x00000000
<input type="checkbox"/>	0x00400024	0x24040000	addiu \$13,\$0,0x00000000	16: li \$t5,0	\$t1	9	0x00000000
<input type="checkbox"/>	0x00400028	0x3c011001	lui \$1,0x00001001	17: li \$t9,0x10010100	\$t2	10	0x00000000
<input type="checkbox"/>	0x0040002c	0x34390100	ori \$25,\$1,0x00000100		\$t3	11	0x00000000
					\$t4	12	0x00000000
					\$t5	13	0x00000000
					\$t6	14	0x00000000
					\$t7	15	0x00000000
					\$t0	16	0x00000000
					\$s1	17	0x00000000
					\$s2	18	0x00000000
					\$s3	19	0x00000000
					\$s4	20	0x00000000
					\$s5	21	0x00000000
					\$s6	22	0x00000000
					\$s7	23	0x00000000
					\$t8	24	0x00000000
					\$t9	25	0x00000000

  

Data Segment						
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Val
0x10010000	0x65746e65	0x68742072	0x6f742065	0x206c6174	0x626d756e	
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010100	0x00000050	0x00000100	0x00000020	0x00000020	0x00000001	

Text Segment					Registers	
Bkpt	Address	Code	Basic		Source	
<input type="checkbox"/>	0x004000b8	0x24020001	addiu \$2,\$0,0x00000001	52:	li \$v0,1	\$zero 0 0x00000000
<input type="checkbox"/>	0x004000bc	0x00046821	addu \$13,\$0,\$4	53:	move \$t5,\$a0	\$at 1 0x10010000
<input type="checkbox"/>	0x004000c0	0x00052021	addu \$4,\$0,\$5	54:	move \$a0,\$a1	\$v0 2 0x00000001
<input type="checkbox"/>	0x004000c4	0x0000000c	syscall	55:	syscall	\$v1 3 0x00000000
<input type="checkbox"/>	0x004000c8	0x24020004	addiu \$2,\$0,0x00000004	56:	li \$v0,4	\$a0 4 0x4e86c2ad
<input type="checkbox"/>	0x004000cc	0x3c011001	lui \$1,0x0001001	57:	la \$a0,buffer1	\$a1 5 0x00000171
<input type="checkbox"/>	0x004000d0	0x3424001b	ori \$4,\$1,0x0000001b			\$a2 6 0x00000000
<input type="checkbox"/>	0x004000d4	0x0000000c	syscall	58:	syscall	\$a3 7 0x00000000
<input type="checkbox"/>	0x004000d8	0x24020001	addiu \$2,\$0,0x00000001	59:	li \$v0,1	\$t0 8 0x00000004
<input type="checkbox"/>	0x004000dc	0x000d2021	addu \$4,\$0,\$13	60:	move \$a0,\$t5	\$t1 9 0x00000001
<input type="checkbox"/>	0x004000e0	0x0000000c	syscall	61:	syscall	\$t2 10 0x00000004
<input checked="" type="checkbox"/>	0x004000e4	0x2402000a	addiu \$2,\$0,0x0000000a	62:	li \$v0,10	\$t3 11 0x00000020
						\$t4 12 0x00000001
						\$t5 13 0x4e86c2ad
						\$t6 14 0x00000001
						\$t7 15 0x10010100
						\$t0 16 0x00000000
						\$t1 17 0x00000000
						\$t2 18 0x00000000
						\$t3 19 0x00000000
						\$t4 20 0x00000000
						\$t5 21 0x00000000
						\$t6 22 0x00000000
						\$t7 23 0x00000000
						\$t8 24 0x00000000
						\$t9 25 0x10010100
						\$k0 26 0x00000000
						\$k1 27 0x00000000
						\$sp 28 0x10008000
						\$sp 29 0x7fffffcf
						\$fp 30 0x00000000
						\$ra 31 0x00000000
						pc 0x004000e4
						hi 0x00000000
						lo 0x00000000

  

Data Segment						
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Val
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010100	0x00000001	0x00000020	0x00000200	0x00000500	0x00010100	
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010140	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010160	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x10010180	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	
0x100101a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	

☒ Hexadecimal Addresses 
 ☒ Hexadecimal Va

  

**Mars Messages**    Run I/O

enter the total number n: 5  
 369 1317444803  
Clear
 369 1317454509

## 源码

```

li $t5,0
li $t9,0x10010100
li $v0,30
syscall
li $v0,1
move $t5,$a0
move $a0,$a1
syscall
li $v0,4
la $a0,buffer1
syscall
li $v0,1
move $a0,$t5
syscall
li $v0,4
la $a0,buffer2
syscall
####外循环
loop1:addi $t0,$t0,1
beq $t0,$t2,done
li $t1,0
sub $t6,$t2,$t0
####内循环
loop2: beq $t1,$t6,loop1
sll $t8,$t1,2
add $t7,$t9,$t8
lw $t3,0($t7)
lw $t4,4($t7)
ble $t3,$t4,target #<=才跳转
sw $t4, 0($t7)
sw $t3, 4($t7)
target: addi $t1,$t1,1
b loop2
####循环结束段，再次调用syscall输出当前时间
done: li $v0,30
syscall
li $v0,1
move $t5,$a0
move $a0,$a1
syscall
li $v0,4
la $a0,buffer1
syscall
li $v0,1
move $a0,$t5
syscall
li $v0,10
syscall

```