基于MIPS汇编的冒泡排序程序

学号：PB17000289 姓名: 于佳睿

实验环境：

- MARS4_5模拟环境(基于javac 12.0.1)

实验代码

首先说明寄存器含义：

> $t0 存储计数变量i(外层)
>
> $t1 存储计数变量j(内层)
>
> $s0 - $s4 存储array原始数据
>
> $s5 存储array首地址
>
> $s7 作为比大小时slt的结果
>
> $t3, $t4 作为swap时临时变量的暂存
>
> $t7 存储i的偏移量
>
> $t8 存储i的偏移量与j的和(内层循环退出的判据)

```
.data
        MyArray: .space 20
        space: .asciiz " "
        huiche: .asciiz "\n"
        speak1: .asciiz "Before Sort : "
        speak2: .asciiz "After Sort : "
.text
main:                                   #initial
        addi $s0, $zero, 3
        addi $s1, $zero, 4
        addi $s2, $zero, 10
        addi $s3, $zero, 5
        addi $s4, $zero, 3
        addi $t0, $zero, 0
        sw $s0, MyArray($t0)
        addi $t0, $zero, 4
        sw $s1, MyArray($t0)
        addi $t0, $zero, 8
        sw $s2, MyArray($t0)
        addi $t0, $zero, 12
```

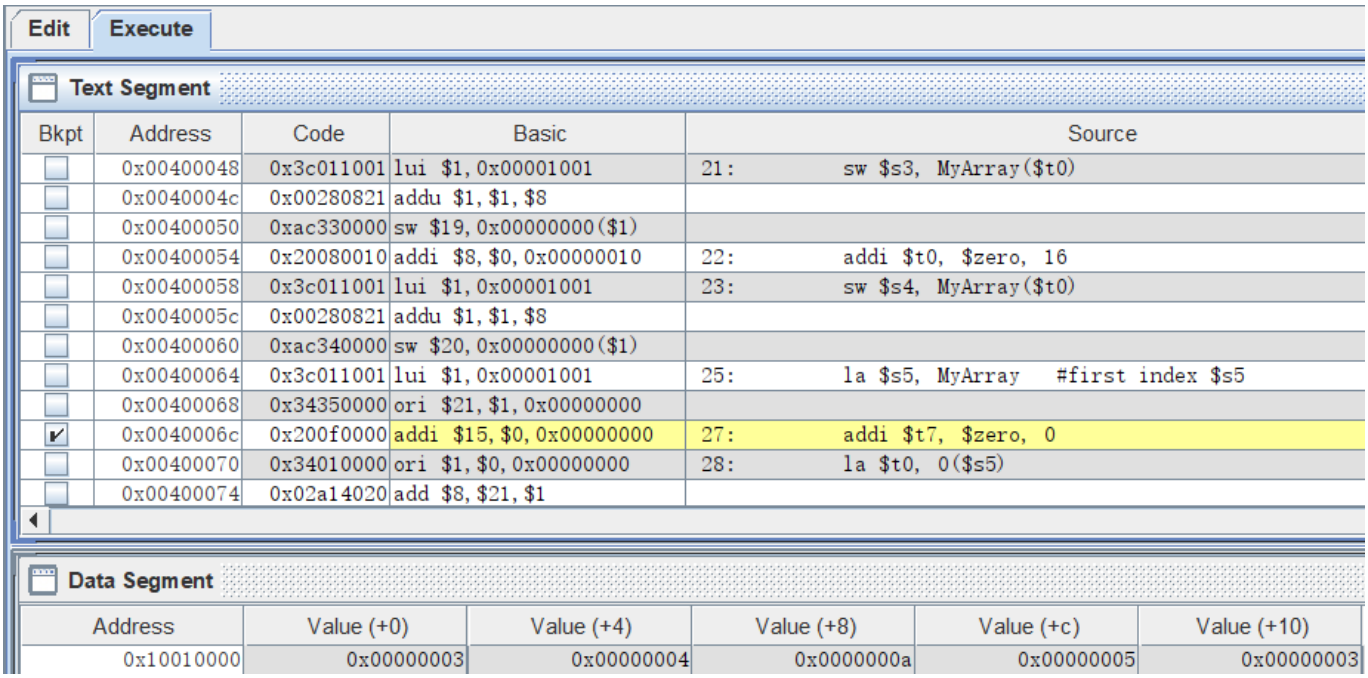
```
        sw $s3, MyArray($t0)
        addi $t0, $zero, 16
        sw $s4, MyArray($t0)            #initial end
        la $s5, MyArray                 #first index $s5
printarray1:                            #print before sort
        addi $t7, $zero, 0
        la $t0, 0($s5)
        li $v0, 4
        la $a0, speak1
        syscall
while1:
        beq $t7, 5, exit1
        lw $t8, 0($t0)
        li $v0, 1
        move $a0, $t8
        syscall
        li $v0, 4
        la $a0, space
        syscall
        addi $t7, $t7, 1
        addi $t0, $t0, 4
        j while1
exit1:
        li $v0, 4
        la $a0, huiche
        syscall
        li $v0, 30    #打印系统时间
        syscall
        li $v0, 1
        syscall
        li $v0, 4
        la $a0, huiche
        syscall
        la $t0, ($s5)                   #i  $t0
        la $t1, ($s5)                   #j  $t1
        la $t5, 20($s5)                 #   相当于20
        la $t6, 16($s5)                 #   相当于16
        addi $t7, $zero, 0
        addi $t8, $zero, 0


while_out:                              #outside while
        beq $t0, $t5, exit_out
        la $t1, ($s5)      #j = i

while_in:                               #inside while
        sub $t7, $t0, $s5                   #i 的偏移量
        add $t8, $t1, $t7
        beq $t8, $t6, exit_in
        lw $t3, 0($t1)                      #array[j]
        lw $t4, 4($t1)                      #array[j + 1]
        slt $s7, $t3, $t4
        bne $s7, $zero, goon
        sw $t3, 4($t1)
```

```
        sw $t4, 0($t1)
goon:
        addi $t1, $t1, 4
        j while_in
exit_in:
       li $v0, 30                          #在外层循环结束一次时打印时间
       syscall
       li $v0, 1
       syscall
        li $v0, 4
        la $a0, huiche
        syscall
update:
        addi $t0, $t0, 4
        j while_out
exit_out:
printarray:
        addi $t7, $zero, 0
        la $t0, 0($s5)
        li $v0, 4
        la $a0, speak2
        syscall
while:
        beq $t7, 5, exit
        lw $t8, 0($t0)
        li $v0, 1
        move $a0, $t8
        syscall
        li $v0, 4
        la $a0, space
        syscall
        addi $t7, $t7, 1
        addi $t0, $t0, 4
        j while
exit:
        li $v0, 10
        syscall
```

调试过程

我在内部循环退出和交换发生处设置了断点，下面展示调试截图

- 初始



Edit Execute

Text Segment

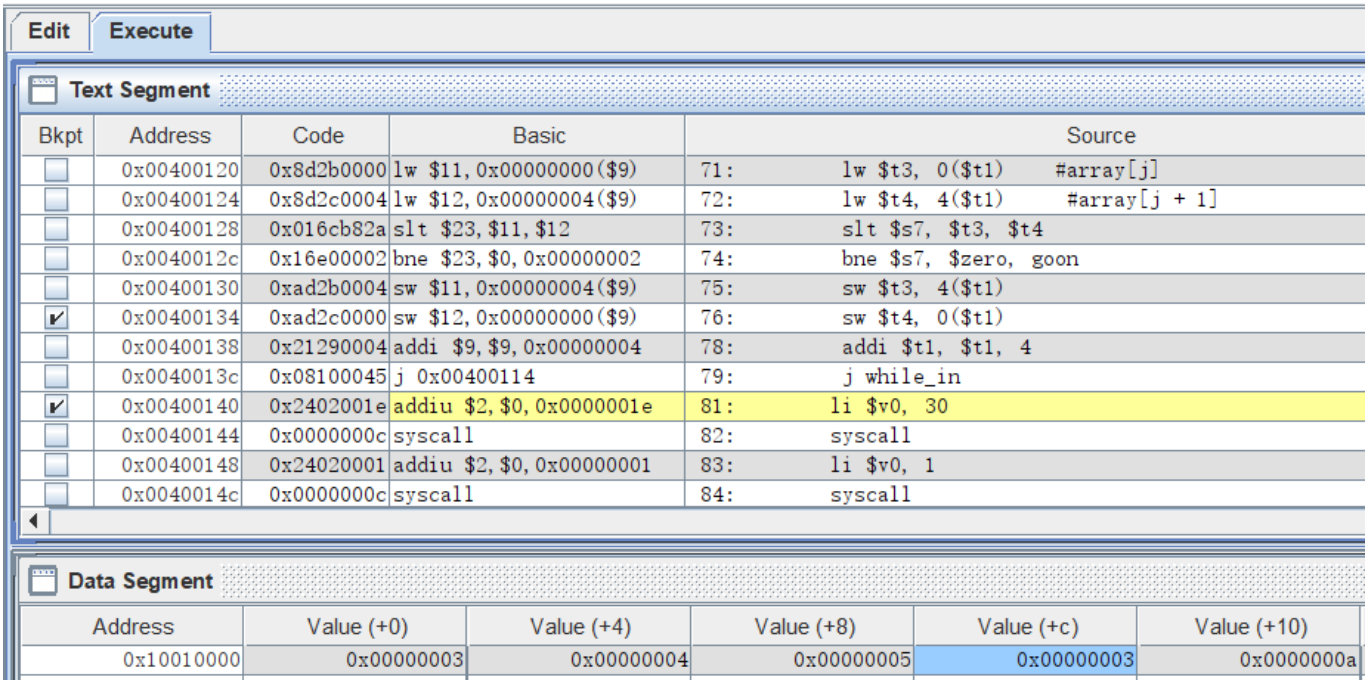
Bkpt	Address	Code	Basic	Source
	0x00400048	0x3c011001	lui $1, 0x00001001	21: sw $s3, MyArray($t0)
	0x0040004c	0x00280821	addu $1, $1, $8	
	0x00400050	0xac330000	sw $19, 0x00000000($1)	
	0x00400054	0x20080010	addi $8, $0, 0x00000010	22: addi $t0, $zero, 16
	0x00400058	0x3c011001	lui $1, 0x00001001	23: sw $s4, MyArray($t0)
	0x0040005c	0x00280821	addu $1, $1, $8	
	0x00400060	0xac340000	sw $20, 0x00000000($1)	
	0x00400064	0x3c011001	lui $1, 0x00001001	25: la $s5, MyArray #first index $s5
	0x00400068	0x34350000	ori $21, $1, 0x00000000	
✓	0x0040006c	0x200f0000	addi $15, $0, 0x00000000	27: addi $t7, $zero, 0
	0x00400070	0x34010000	ori $1, $0, 0x00000000	28: la $t0, 0($s5)
	0x00400074	0x02a14020	add $8, $21, $1	

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	0x00000003	0x00000004	0x0000000a	0x00000005	0x00000003

- 内部循环退出

第一次



Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400120	0x8d2b0000	lw $11, 0x00000000($9)	71: lw $t3, 0($t1) #array[j]
	0x00400124	0x8d2c0004	lw $12, 0x00000004($9)	72: lw $t4, 4($t1) #array[j + 1]
	0x00400128	0x016cb82a	slt $23, $11, $12	73: slt $s7, $t3, $t4
	0x0040012c	0x16e00002	bne $23, $0, 0x00000002	74: bne $s7, $zero, goon
	0x00400130	0xad2b0004	sw $11, 0x00000004($9)	75: sw $t3, 4($t1)
✓	0x00400134	0xad2c0000	sw $12, 0x00000000($9)	76: sw $t4, 0($t1)
	0x00400138	0x21290004	addi $9, $9, 0x00000004	78: addi $t1, $t1, 4
	0x0040013c	0x08100045	j 0x00400114	79: j while_in
✓	0x00400140	0x2402001e	addiu $2, $0, 0x0000001e	81: li $v0, 30
	0x00400144	0x0000000c	syscall	82: syscall
	0x00400148	0x24020001	addiu $2, $0, 0x00000001	83: li $v0, 1
	0x0040014c	0x0000000c	syscall	84: syscall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	0x00000003	0x00000004	0x00000005	0x00000003	0x0000000a

第二次

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400120	0x8d2b0000	lw $11, 0x00000000($9)	71: lw $t3, 0($t1) #array[j]
	0x00400124	0x8d2c0004	lw $12, 0x00000004($9)	72: lw $t4, 4($t1) #array[j + 1]
	0x00400128	0x016cb82a	slt $23, $11, $12	73: slt $s7, $t3, $t4
	0x0040012c	0x16e00002	bne $23, $0, 0x00000002	74: bne $s7, $zero, goon
	0x00400130	0xad2b0004	sw $11, 0x00000004($9)	75: sw $t3, 4($t1)
✓	0x00400134	0xad2c0000	sw $12, 0x00000000($9)	76: sw $t4, 0($t1)
	0x00400138	0x21290004	addi $9, $9, 0x00000004	78: addi $t1, $t1, 4
	0x0040013c	0x08100045	j 0x00400114	79: j while_in
✓	0x00400140	0x2402001e	addiu $2, $0, 0x0000001e	81: li $v0, 30
	0x00400144	0x0000000c	syscall	82: syscall
	0x00400148	0x24020001	addiu $2, $0, 0x00000001	83: li $v0, 1
	0x0040014c	0x0000000c	syscall	84: syscall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	0x00000003	0x00000004	0x00000003	0x00000005	0x0000000a

第三次

Edit Execute

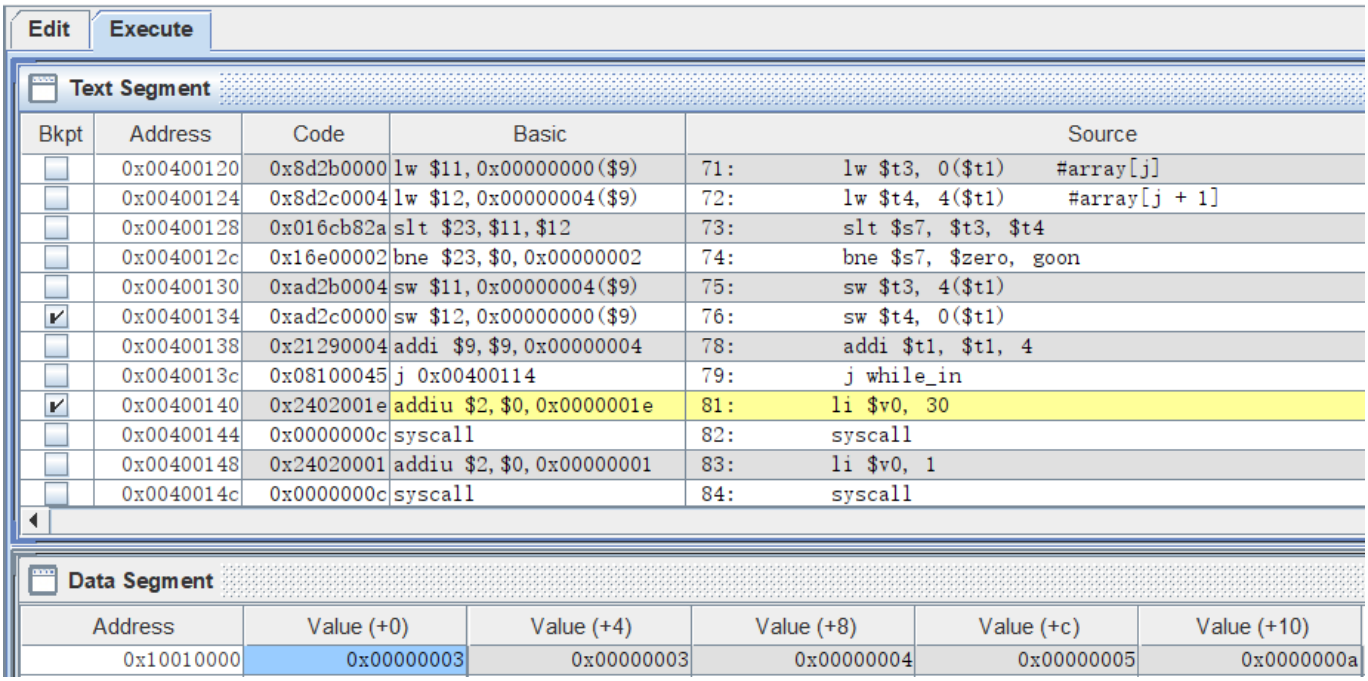
Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400120	0x8d2b0000	lw $11, 0x00000000($9)	71: lw $t3, 0($t1) #array[j]
	0x00400124	0x8d2c0004	lw $12, 0x00000004($9)	72: lw $t4, 4($t1) #array[j + 1]
	0x00400128	0x016cb82a	slt $23, $11, $12	73: slt $s7, $t3, $t4
	0x0040012c	0x16e00002	bne $23, $0, 0x00000002	74: bne $s7, $zero, goon
	0x00400130	0xad2b0004	sw $11, 0x00000004($9)	75: sw $t3, 4($t1)
✓	0x00400134	0xad2c0000	sw $12, 0x00000000($9)	76: sw $t4, 0($t1)
	0x00400138	0x21290004	addi $9, $9, 0x00000004	78: addi $t1, $t1, 4
	0x0040013c	0x08100045	j 0x00400114	79: j while_in
✓	0x00400140	0x2402001e	addiu $2, $0, 0x0000001e	81: li $v0, 30
	0x00400144	0x0000000c	syscall	82: syscall
	0x00400148	0x24020001	addiu $2, $0, 0x00000001	83: li $v0, 1
	0x0040014c	0x0000000c	syscall	84: syscall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	0x00000003	0x00000003	0x00000004	0x00000005	0x0000000a

第四次



Edit | Execute

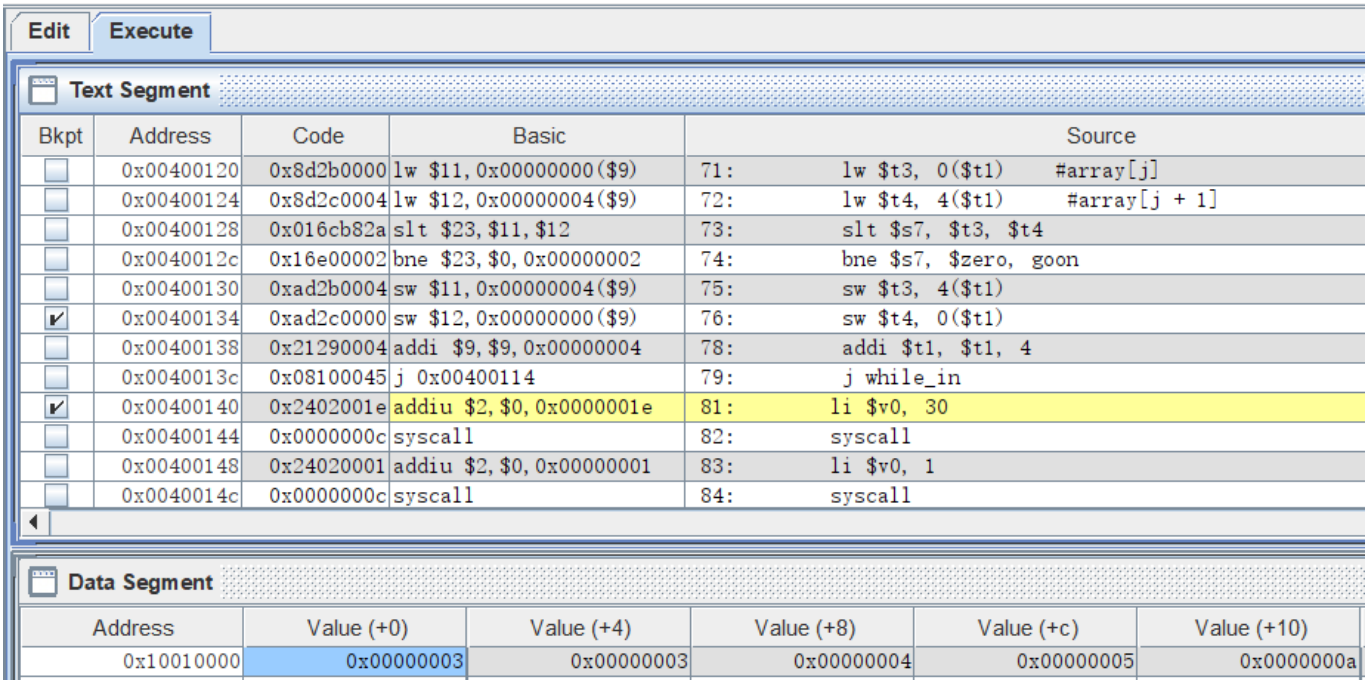
Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400120	0x8d2b0000	lw $11, 0x00000000($9)	71: lw $t3, 0($t1) #array[j]
	0x00400124	0x8d2c0004	lw $12, 0x00000004($9)	72: lw $t4, 4($t1) #array[j + 1]
	0x00400128	0x016cb82a	slt $23, $11, $12	73: slt $s7, $t3, $t4
	0x0040012c	0x16e00002	bne $23, $0, 0x00000002	74: bne $s7, $zero, goon
	0x00400130	0xad2b0004	sw $11, 0x00000004($9)	75: sw $t3, 4($t1)
✔	0x00400134	0xad2c0000	sw $12, 0x00000000($9)	76: sw $t4, 0($t1)
	0x00400138	0x21290004	addi $9, $9, 0x00000004	78: addi $t1, $t1, 4
	0x0040013c	0x08100045	j 0x00400114	79: j while_in
✔	0x00400140	0x2402001e	addiu $2, $0, 0x0000001e	81: li $v0, 30
	0x00400144	0x0000000c	syscall	82: syscall
	0x00400148	0x24020001	addiu $2, $0, 0x00000001	83: li $v0, 1
	0x0040014c	0x0000000c	syscall	84: syscall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	0x00000003	0x00000003	0x00000004	0x00000005	0x0000000a

第五次(实际未进入)



Edit | Execute

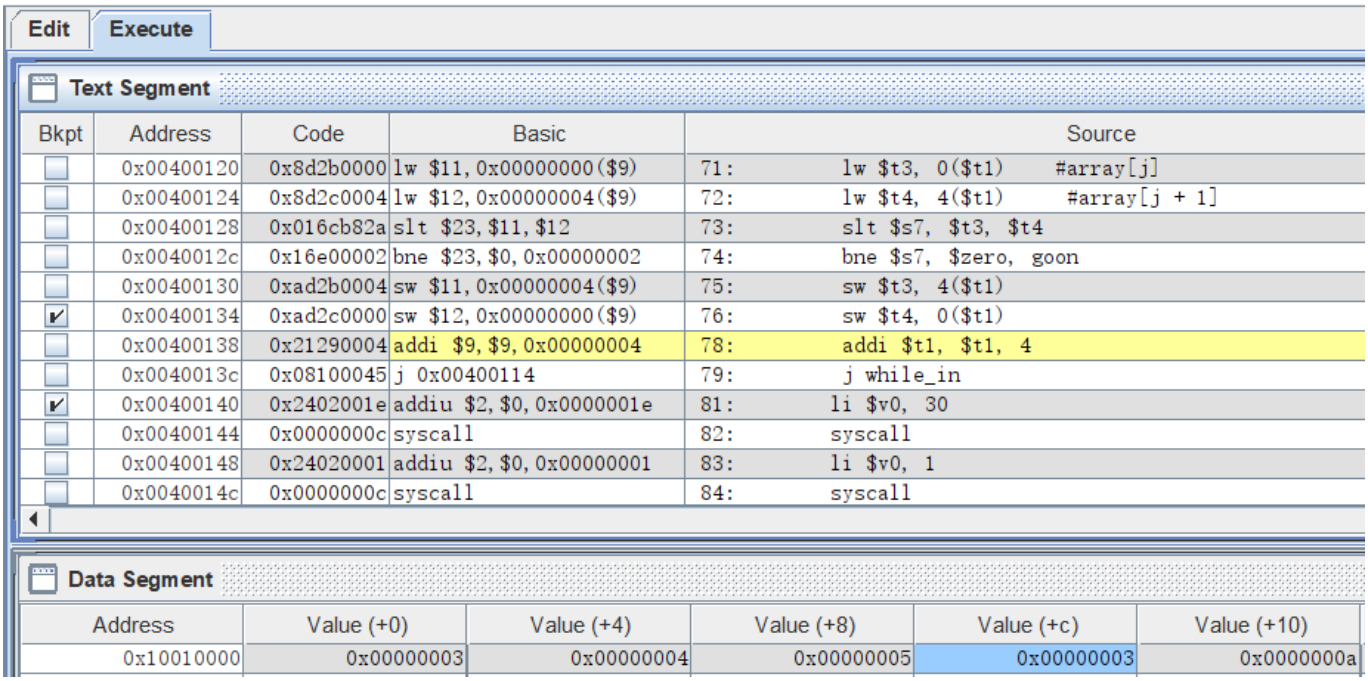
Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400120	0x8d2b0000	lw $11, 0x00000000($9)	71: lw $t3, 0($t1) #array[j]
	0x00400124	0x8d2c0004	lw $12, 0x00000004($9)	72: lw $t4, 4($t1) #array[j + 1]
	0x00400128	0x016cb82a	slt $23, $11, $12	73: slt $s7, $t3, $t4
	0x0040012c	0x16e00002	bne $23, $0, 0x00000002	74: bne $s7, $zero, goon
	0x00400130	0xad2b0004	sw $11, 0x00000004($9)	75: sw $t3, 4($t1)
✔	0x00400134	0xad2c0000	sw $12, 0x00000000($9)	76: sw $t4, 0($t1)
	0x00400138	0x21290004	addi $9, $9, 0x00000004	78: addi $t1, $t1, 4
	0x0040013c	0x08100045	j 0x00400114	79: j while_in
✔	0x00400140	0x2402001e	addiu $2, $0, 0x0000001e	81: li $v0, 30
	0x00400144	0x0000000c	syscall	82: syscall
	0x00400148	0x24020001	addiu $2, $0, 0x00000001	83: li $v0, 1
	0x0040014c	0x0000000c	syscall	84: syscall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	0x00000003	0x00000003	0x00000004	0x00000005	0x0000000a

- swap发生的时候：



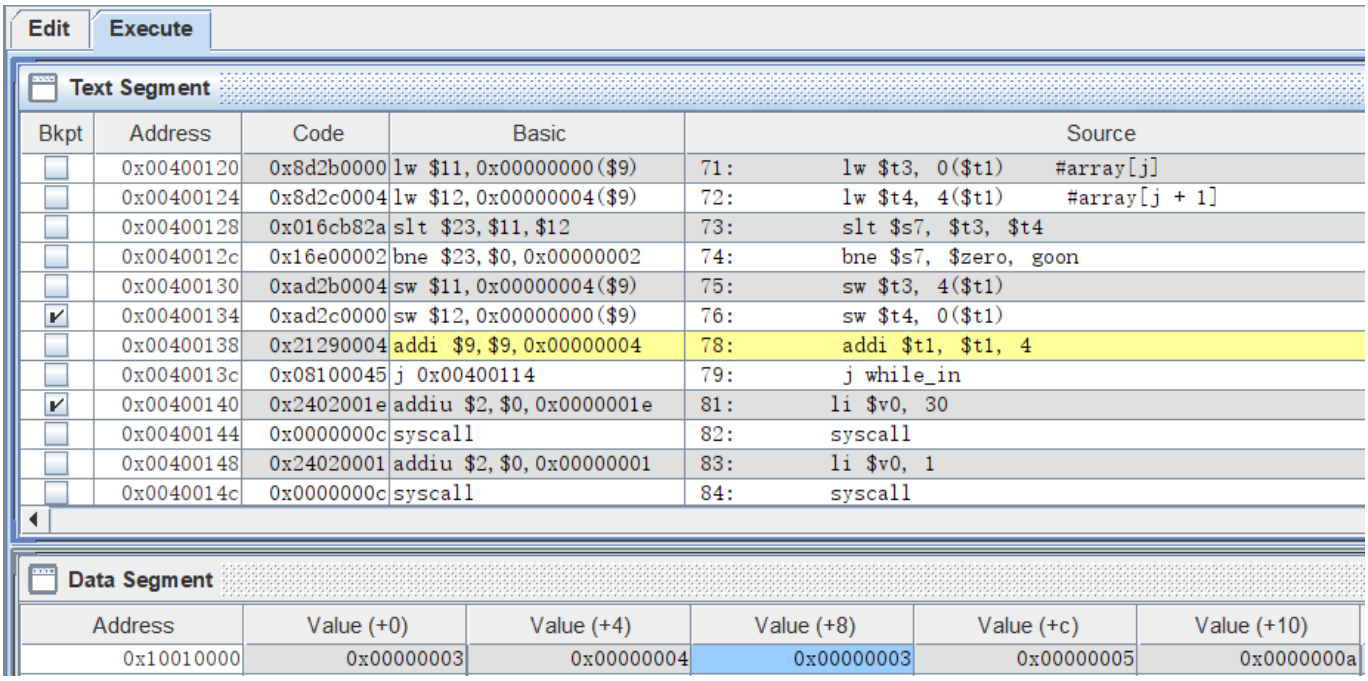
Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400120	0x8d2b0000	lw $11, 0x00000000($9)	71: lw $t3, 0($t1) #array[j]
	0x00400124	0x8d2c0004	lw $12, 0x00000004($9)	72: lw $t4, 4($t1) #array[j + 1]
	0x00400128	0x016cb82a	slt $23, $11, $12	73: slt $s7, $t3, $t4
	0x0040012c	0x16e00002	bne $23, $0, 0x00000002	74: bne $s7, $zero, goon
	0x00400130	0xad2b0004	sw $11, 0x00000004($9)	75: sw $t3, 4($t1)
✓	0x00400134	0xad2c0000	sw $12, 0x00000000($9)	76: sw $t4, 0($t1)
	0x00400138	0x21290004	addi $9, $9, 0x00000004	78: addi $t1, $t1, 4
	0x0040013c	0x08100045	j 0x00400114	79: j while_in
✓	0x00400140	0x2402001e	addiu $2, $0, 0x0000001e	81: li $v0, 30
	0x00400144	0x0000000c	syscall	82: syscall
	0x00400148	0x24020001	addiu $2, $0, 0x00000001	83: li $v0, 1
	0x0040014c	0x0000000c	syscall	84: syscall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	0x00000003	0x00000004	0x00000005	0x00000003	0x0000000a



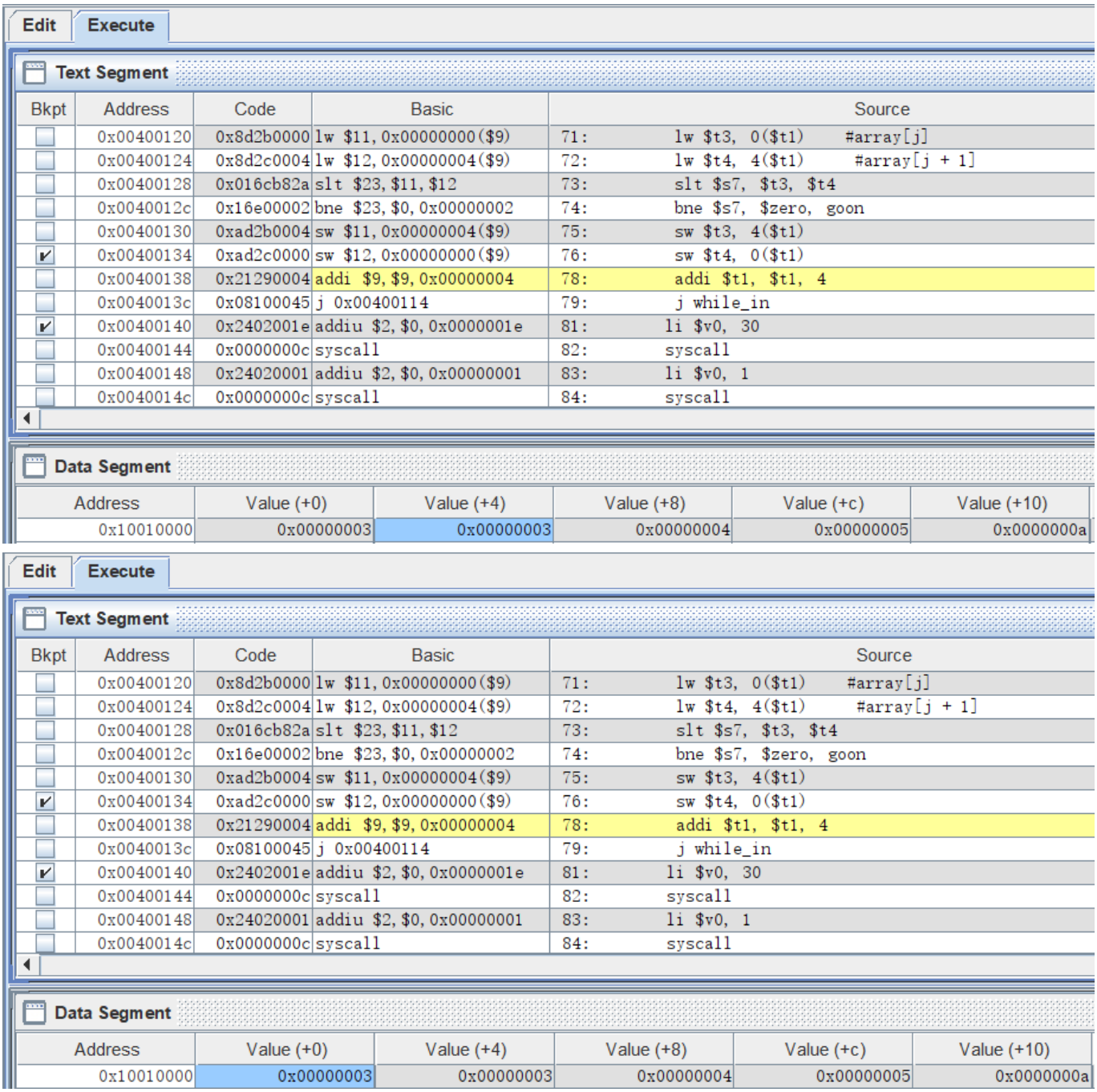
Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400120	0x8d2b0000	lw $11, 0x00000000($9)	71: lw $t3, 0($t1) #array[j]
	0x00400124	0x8d2c0004	lw $12, 0x00000004($9)	72: lw $t4, 4($t1) #array[j + 1]
	0x00400128	0x016cb82a	slt $23, $11, $12	73: slt $s7, $t3, $t4
	0x0040012c	0x16e00002	bne $23, $0, 0x00000002	74: bne $s7, $zero, goon
	0x00400130	0xad2b0004	sw $11, 0x00000004($9)	75: sw $t3, 4($t1)
✓	0x00400134	0xad2c0000	sw $12, 0x00000000($9)	76: sw $t4, 0($t1)
	0x00400138	0x21290004	addi $9, $9, 0x00000004	78: addi $t1, $t1, 4
	0x0040013c	0x08100045	j 0x00400114	79: j while_in
✓	0x00400140	0x2402001e	addiu $2, $0, 0x0000001e	81: li $v0, 30
	0x00400144	0x0000000c	syscall	82: syscall
	0x00400148	0x24020001	addiu $2, $0, 0x00000001	83: li $v0, 1
	0x0040014c	0x0000000c	syscall	84: syscall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	0x00000003	0x00000004	0x00000003	0x00000005	0x0000000a



Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400120	0x8d2b0000	lw $11, 0x00000000($9)	71: lw $t3, 0($t1) #array[j]
	0x00400124	0x8d2c0004	lw $12, 0x00000004($9)	72: lw $t4, 4($t1) #array[j + 1]
	0x00400128	0x016cb82a	slt $23, $11, $12	73: slt $s7, $t3, $t4
	0x0040012c	0x16e00002	bne $23, $0, 0x00000002	74: bne $s7, $zero, goon
	0x00400130	0xad2b0004	sw $11, 0x00000004($9)	75: sw $t3, 4($t1)
✓	0x00400134	0xad2c0000	sw $12, 0x00000000($9)	76: sw $t4, 0($t1)
	0x00400138	0x21290004	addi $9, $9, 0x00000004	78: addi $t1, $t1, 4
	0x0040013c	0x08100045	j 0x00400114	79: j while_in
✓	0x00400140	0x2402001e	addiu $2, $0, 0x0000001e	81: li $v0, 30
	0x00400144	0x0000000c	syscall	82: syscall
	0x00400148	0x24020001	addiu $2, $0, 0x00000001	83: li $v0, 1
	0x0040014c	0x0000000c	syscall	84: syscall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	0x00000003	0x00000003	0x00000004	0x00000005	0x0000000a

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400120	0x8d2b0000	lw $11, 0x00000000($9)	71: lw $t3, 0($t1) #array[j]
	0x00400124	0x8d2c0004	lw $12, 0x00000004($9)	72: lw $t4, 4($t1) #array[j + 1]
	0x00400128	0x016cb82a	slt $23, $11, $12	73: slt $s7, $t3, $t4
	0x0040012c	0x16e00002	bne $23, $0, 0x00000002	74: bne $s7, $zero, goon
	0x00400130	0xad2b0004	sw $11, 0x00000004($9)	75: sw $t3, 4($t1)
✓	0x00400134	0xad2c0000	sw $12, 0x00000000($9)	76: sw $t4, 0($t1)
	0x00400138	0x21290004	addi $9, $9, 0x00000004	78: addi $t1, $t1, 4
	0x0040013c	0x08100045	j 0x00400114	79: j while_in
✓	0x00400140	0x2402001e	addiu $2, $0, 0x0000001e	81: li $v0, 30
	0x00400144	0x0000000c	syscall	82: syscall
	0x00400148	0x24020001	addiu $2, $0, 0x00000001	83: li $v0, 1
	0x0040014c	0x0000000c	syscall	84: syscall

Data Segment

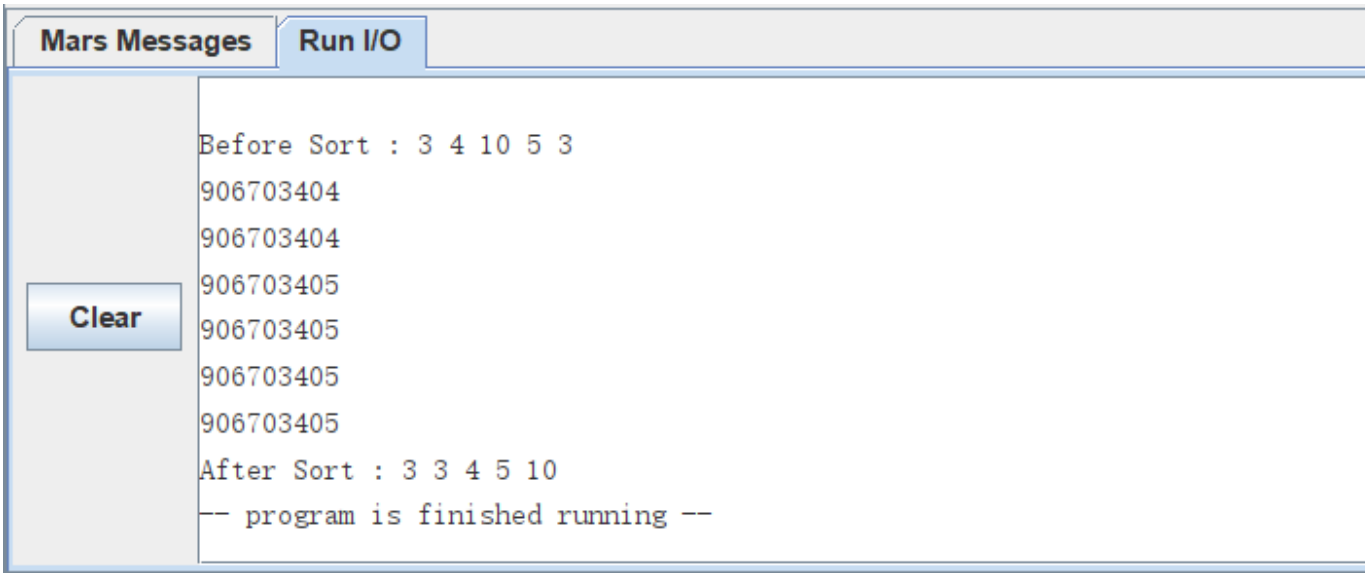
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	0x00000003	0x00000003	0x00000004	0x00000005	0x0000000a

- 寄存器，存储器截图：(不一一列出)

Registers | Coproc 1 | Coproc 0

Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000005
$v0	2	0x0000000a
$v1	3	0x00000000
$a0	4	0x10010014
$a1	5	0x0000016a
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x10010014
$t1	9	0x10010000
$t2	10	0x00000000
$t3	11	0x00000003
$t4	12	0x00000003
$t5	13	0x10010014
$t6	14	0x10010010
$t7	15	0x00000005
$s0	16	0x00000003
$s1	17	0x00000004
$s2	18	0x0000000a
$s3	19	0x00000005
$s4	20	0x00000003
$s5	21	0x10010000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x0000000a
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x004001a4
hi		0x00000000
lo		0x00000000

实验结果



结果分析：

- 第一个时间时初始时间，算法开始执行的时间，后面的时间是退出内循环的时间
- 通过syscall打印了系统时间，通过查阅资料，这一时间以ms为单位，所以代码运行时间不会超过2 ms
- 另外，前面内层循环退出时交换的次数多，比较的次数多，所以"较慢",后面的循环在ms量级下不显著

实验体会

1. 通过本次实验，我学习了mips的指令写法，学习了syscall和debug的方式
2. 通过本次实验，我学习了打印system time的方法
3. 通过本次实验，我更加理解了la和lw的区别，理解了偏移寻址的方式，感觉收获很大！

不足之处

代码复用性不好，没有发挥label的优势，写得过于冗长