实验一：MIPS汇编程序设计

专业班级：**通信2101班**

姓名：　　**罗畅**

学号：　　**U202113940**

实验名称

MIPS汇编程序设计

实验目的：

1. 熟悉常见的MIPS汇编指令
2. 掌握MIPS汇编程序设计
3. 了解MIPS汇编语言与机器语言之间的对应关系
4. 了解C语言语句与汇编指令之间的关系
5. 掌握MARS的调试技术
6. 掌握程序的内存映像

实验仪器

***Mars MIPS*汇编编译器**

实验任务

- 在数据段定义两个int型变量a, b;
- 在数据段定义一个int型数组c[40]，不初始化
- 通过系统调用功能从键盘输入a, b的值（不大于20）
- 采用MIPS汇编指令实现c[a+b] = a*b
- 通过系统调用功能分别显示c[a+b]所在的存储地址和值
- 指出程序运行后a, b, c[a+b]所在的数据段储存位置以及取值，验证程序功能的正确性

实验源代码

```
.data
a: .word 0
b: .word 0
c: .word 0:20

inputAStr: .asciiz "Please input the value of a:"
inputBStr: .asciiz "Please input the value of b:"

outputAddr: .asciiz "The data address is:"
outputData: .asciiz "\nThe data is:"

.text
.globl main

main:
    la $a0, inputAStr #print "Please input the value of a:"
    li $v0, 4
    syscall

    li $v0, 5 #read a's value from terminal
    syscall

    add $t0, $v0, $zero #save a's value to $t0

    la $a0, a
    sw $t0, 0($a0) #save a's value to memory


    la $a0, inputBStr #print "Please input the value of b:"
    li $v0, 4
    syscall

    li $v0, 5 #read b's value from terminal
    syscall

    add $t1, $v0, $zero #save b's value to $t1

    la $a0, b #save b's value to memory
    sw $t1, 0($a0)

    add $t2, $t0, $t1 #calculate the array subscript (a+b)
    mul $t3, $t0, $t1 #calculate the data (a*b)
    addi $s0,$zero,4
    mul $t4,$s0,$t2 #offset correction

    la $a0, c #get c's first address
    add $a0, $a0, $t4 #get right addr

    sw $t3, 0($a0) #save data to memory
```

```
    add $s0, $a0,$zero #save the address to $to

    la $a0,outputAddr #print "The data address is:"
    li $v0, 4
    syscall

    add $a0, $s0,$zero #print the data address
    li $v0, 1
    syscall

    la $a0,outputData #print "\nThe data is:"
    li $v0,4
    syscall

    add $a0,$t3,$zero #print the data value
    li $v0,1
    syscall

    li $v0,10 #exit
    syscall
```

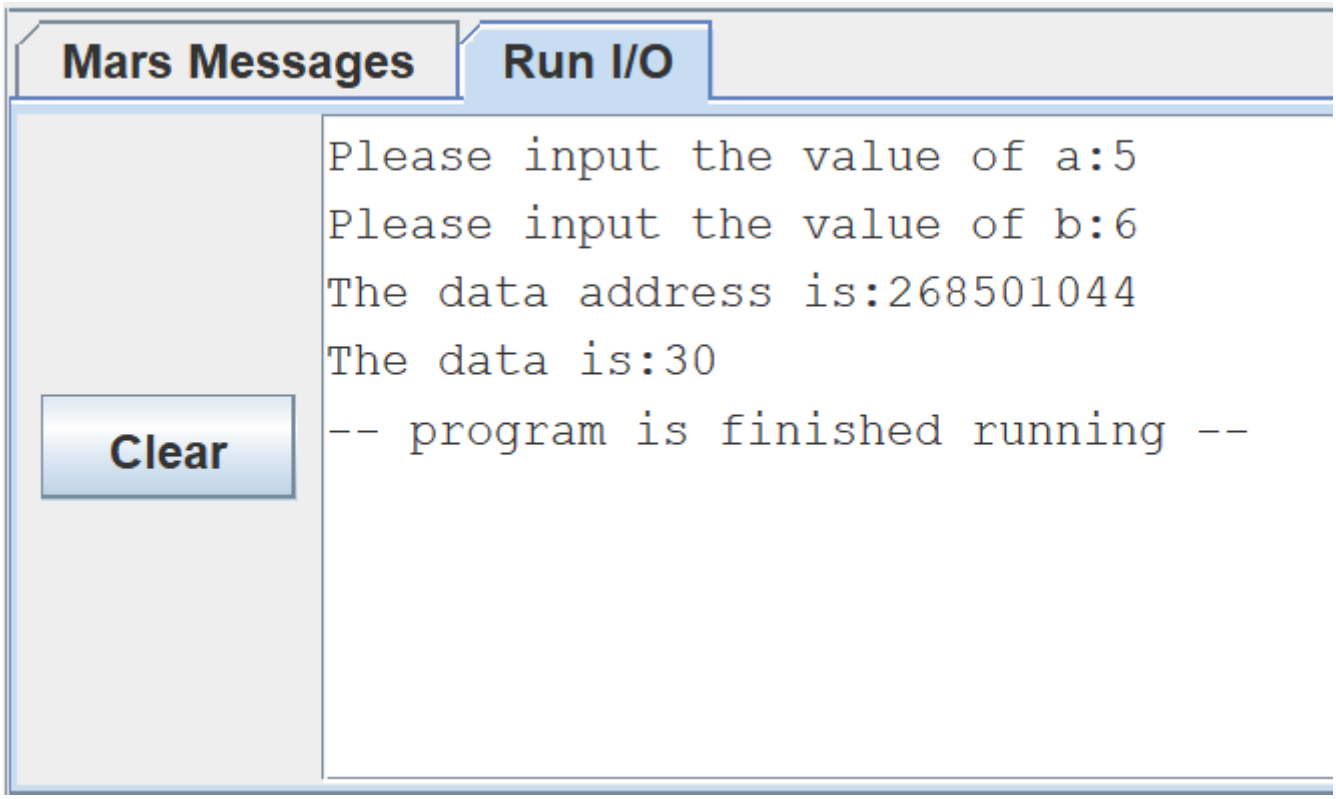
实验结果

程序代码段映像

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x3c011001	lui $1,4097	16: la $a0, inputAStr #print "Please input the value of a:"
	0x00400004	0x34240058	ori $4,$1,88	
	0x00400008	0x24020004	addiu $2,$0,4	17: li $v0, 4
	0x0040000c	0x0000000c	syscall	18: syscall
	0x00400010	0x24020005	addiu $2,$0,5	20: li $v0, 5 #read a's value from terminal
	0x00400014	0x0000000c	syscall	21: syscall
	0x00400018	0x00404020	add $8,$2,$0	23: add $t0, $v0, $zero #save a's value to $t0
	0x0040001c	0x3c011001	lui $1,4097	25: la $a0, a
	0x00400020	0x34240000	ori $4,$1,0	
	0x00400024	0xac880000	sw $8,0($4)	26: sw $t0, 0($a0) #save a's value to memory
	0x00400028	0x3c011001	lui $1,4097	29: la $a0, inputBStr #print "Please input the value of b:"
	0x0040002c	0x34240075	ori $4,$1,117	
	0x00400030	0x24020004	addiu $2,$0,4	30: li $v0, 4
	0x00400034	0x0000000c	syscall	31: syscall
	0x00400038	0x24020005	addiu $2,$0,5	33: li $v0, 5 #read b's value from terminal
	0x0040003c	0x0000000c	syscall	34: syscall
	0x00400040	0x00404820	add $9,$2,$0	36: add $t1, $v0, $zero #save b's value to $t1
	0x00400044	0x3c011001	lui $1,4097	38: la $a0, b #save b's value to memory
	0x00400048	0x34240004	ori $4,$1,4	
	0x0040004c	0xac890000	sw $9,0($4)	39: sw $t1, 0($a0)
	0x00400050	0x01095020	add $10,$8,$9	41: add $t2, $t0, $t1 #calculate the array subscript (a+b)
	0x00400054	0x71095802	mul $11,$8,$9	42: mul $t3, $t0, $t1 #calculate the data (a*b)
	0x00400058	0x20100004	addi $16,$0,4	43: addi $s0,$zero,4
	0x0040005c	0x720a6002	mul $12,$16,$10	44: mul $t4,$s0,$t2 #offset correction
	0x00400060	0x3c011001	lui $1,4097	46: la $a0, c #get c's first address
	0x00400064	0x34240008	ori $4,$1,8	
	0x00400068	0x008c2020	add $4,$4,$12	47: add $a0, $a0, $t4 #get right addr
	0x0040006c	0xac8b0000	sw $11,0($4)	49: sw $t3, 0($a0) #save data to memory
	0x00400070	0x00808020	add $16,$4,$0	51: add $s0, $a0,$zero #save the address to $to
	0x00400074	0x3c011001	lui $1,4097	53: la $a0,outputAddr #print "The data address is:"
	0x00400078	0x34240092	ori $4,$1,146	
	0x0040007c	0x24020004	addiu $2,$0,4	54: li $v0, 4
	0x00400080	0x0000000c	syscall	55: syscall
	0x00400084	0x02002020	add $4,$16,$0	57: add $a0, $s0,$zero #print the data address
	0x00400088	0x24020001	addiu $2,$0,1	58: li $v0, 1
	0x0040008c	0x0000000c	syscall	59: syscall
	0x00400090	0x3c011001	lui $1,4097	61: la $a0,outputData #print "\nThe data is:"
	0x00400094	0x342400a7	ori $4,$1,167	
	0x00400098	0x24020004	addiu $2,$0,4	62: li $v0,4
	0x0040009c	0x0000000c	syscall	63: syscall
	0x004000a0	0x01602020	add $4,$11,$0	65: add $a0,$t3,$zero #print the data value
	0x004000a4	0x24020001	addiu $2,$0,1	66: li $v0,1
	0x004000a8	0x0000000c	syscall	67: syscall
	0x004000ac	0x2402000a	addiu $2,$0,10	69: li $v0,10 #exit
	0x004000b0	0x0000000c	syscall	70: syscall

输入输出端口测试



程序数据段映像

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	5	6	0	0	0	0	0	0
0x10010020	0	0	0	0	0	30	0	0
0x10010040	0	0	0	0	0	0	1634036816	1763730803
0x10010060	1953853550	1701344288	1818326560	1864394101	979443814	1701597184	543519585	1970302569
0x10010080	1752440948	1635131493	543520108	1646290543	1750335546	1633951845	1629512052	1701995620
0x100100a0	1763734387	167787123	543516756	1635017060	980642080	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0
0x10010140	0	0	0	0	0	0	0	0
0x10010160	0	0	0	0	0	0	0	0
0x10010180	0	0	0	0	0	0	0	0

Registers | Coproc 1 | Coproc 0

Name	Number	Value
$zero	0	0
$at	1	268500992
$v0	2	10
$v1	3	0
$a0	4	30
$a1	5	0
$a2	6	0
$a3	7	0
$t0	8	5
$t1	9	6
$t2	10	11
$t3	11	30
$t4	12	44
$t5	13	0
$t6	14	0
$t7	15	0
$s0	16	268501044
$s1	17	0
$s2	18	0
$s3	19	0
$s4	20	0
$s5	21	0
$s6	22	0
$s7	23	0
$t8	24	0
$t9	25	0
$k0	26	0
$k1	27	0
$gp	28	268468224
$sp	29	2147479548
$fp	30	0
$ra	31	0
pc		4194484
hi		0
lo		44

结果分析

从I/O端口输入的数据正确的运算并存入了a、b、c[a+b]对应的内存映像中，实验正确

实验小结

本次实验我使用了Mars软件进行汇编语言的练习，学会了使用syscall来进行数据的输入和输出，最后实验结果正确，收获很大！