实验一：MIPS汇编程序设计

班级：**提高2301班**
姓名：**张禹阳**
学号：**U202314270**

实验任务及目的

采用MIPS汇编程序实现以下功能：

- 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
- 通过系统功能调用从键盘输入 `a, b` 的值（不大于20）
- 采用MIPS汇编指令实现 `c[a+b] = a * b`
- 通过系统功能调用分别显示 `c[a+b]` 所在的地址和值
- 指出程序运行结束后 `a, b, c[a+b]` 所在的数据存储位置以及取值，验证程序功能的正确性

本次实验目的为：

- 熟悉常见的MIPS汇编指令
- 掌握MIPS汇编程序设计
- 掌握MARS的调试技术
- 掌握程序的内存映像

实验环境

- *Mars* MIPS 汇编编译器
- *Windows 11* 操作系统

设计方案

- 在数据段定义 `a, b, c[40]`
- 通过 `syscall` 从键盘读取 `a, b`
- 将 `a, b` 的值装载入寄存器后，计算 `a+b, a*b`
- 将 `c` 的基地址装在入寄存器后，计算 `c[a+b]` 的偏移地址
- 通过 `syscall` 显示 `c[a+b]` 的地址和值

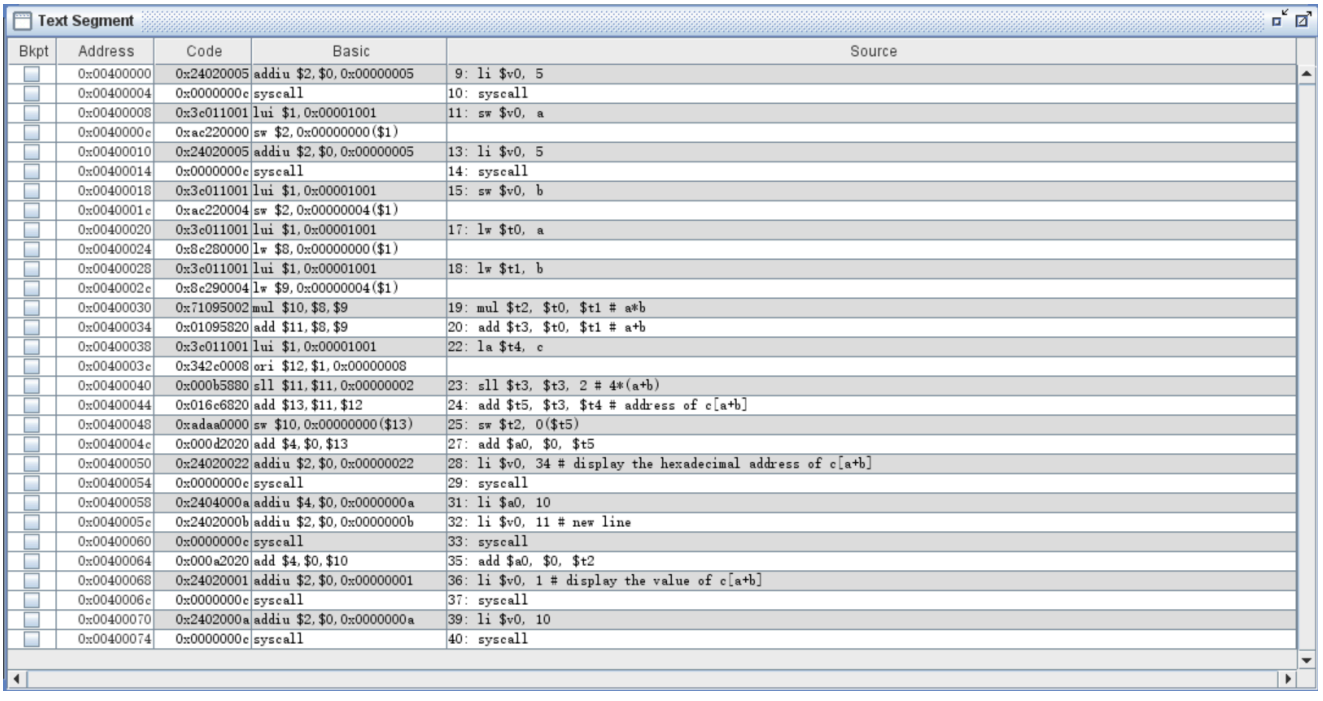
实验源代码如下：

```
.data
a: .word 0
b: .word 0
c: .space 160

.text
main:

li $v0, 5
syscall
sw $v0, a

li $v0, 5
syscall
sw $v0, b

lw $t0, a
lw $t1, b
mul $t2, $t0, $t1 # a*b
add $t3, $t0, $t1 # a+b

la $t4, c
```

```
sll $t3, $t3, 2 # 4*(a+b)
add $t5, $t3, $t4 # address of c[a+b]
sw $t2, 0($t5)

add $a0, $0, $t5
li $v0, 34 # display the hexadecimal address of c[a+b]
syscall

li $a0, 10
li $v0, 11 # new line
syscall

add $a0, $0, $t2
li $v0, 1 # display the value of c[a+b]
syscall

li $v0, 10
syscall
```

实验结果

程序代码段映像



Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x24020005	addiu $2,$0,0x00000005	9: li $v0, 5
	0x00400004	0x0000000c	syscall	10: syscall
	0x00400008	0x3c011001	lui $1,0x00001001	11: sw $v0, a
	0x0040000c	0xac220000	sw $2,0x00000000($1)	
	0x00400010	0x24020005	addiu $2,$0,0x00000005	13: li $v0, 5
	0x00400014	0x0000000c	syscall	14: syscall
	0x00400018	0x3c011001	lui $1,0x00001001	15: sw $v0, b
	0x0040001c	0xac220004	sw $2,0x00000004($1)	
	0x00400020	0x3c011001	lui $1,0x00001001	17: lw $t0, a
	0x00400024	0x8c280000	lw $8,0x00000000($1)	
	0x00400028	0x3c011001	lui $1,0x00001001	18: lw $t1, b
	0x0040002c	0x8c290004	lw $9,0x00000004($1)	
	0x00400030	0x71095002	mul $10,$8,$9	19: mul $t2, $t0, $t1 # a*b
	0x00400034	0x01095820	add $11,$8,$9	20: add $t3, $t0, $t1 # a+b
	0x00400038	0x3c011001	lui $1,0x00001001	22: la $t4, c
	0x0040003c	0x342c0008	ori $12,$1,0x00000008	
	0x00400040	0x000b5880	sll $11,$11,0x00000002	23: sll $t3, $t3, 2 # 4*(a+b)
	0x00400044	0x016c6820	add $13,$11,$12	24: add $t5, $t3, $t4 # address of c[a+b]
	0x00400048	0xadaa0000	sw $10,0x00000000($13)	25: sw $t2, 0($t5)
	0x0040004c	0x000d2020	add $4,$0,$13	27: add $a0, $0, $t5
	0x00400050	0x24020022	addiu $2,$0,0x00000022	28: li $v0, 34 # display the hexadecimal address of c[a+b]
	0x00400054	0x0000000c	syscall	29: syscall
	0x00400058	0x2404000a	addiu $4,$0,0x0000000a	31: li $a0, 10
	0x0040005c	0x2402000b	addiu $2,$0,0x0000000b	32: li $v0, 11 # new line
	0x00400060	0x0000000c	syscall	33: syscall
	0x00400064	0x000a2020	add $4,$0,$10	35: add $a0, $0, $t2
	0x00400068	0x24020001	addiu $2,$0,0x00000001	36: li $v0, 1 # display the value of c[a+b]
	0x0040006c	0x0000000c	syscall	37: syscall
	0x00400070	0x2402000a	addiu $2,$0,0x0000000a	39: li $v0, 10
	0x00400074	0x0000000c	syscall	40: syscall

输入输出端口测试

数组c的初始地址 0x10010008

1. 输入a = 3，b = 6
   c[a+b] 的地址为 0x10010008 + (3+6) × 4 = 0×1001002c
   c[a+b] 的值为 0x00000012 = 18
2. 输入a = 8，b = 12
   c[a+b] 的地址为 0x10010008 + (8+12) × 4 = 0×10010058
   c[a+b] 的值为 0x00000060 = 96

```
3
6
0x1001002c
18
— program is finished running —
```

```
8
12
0x10010058
96
— program is finished running —
```

程序数据段映像（`a = 8, b = 12`）

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000008	0x0000000c	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000060	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010140	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010160	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010180	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100101a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100101c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100101e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Registers | Coproc 1 | Coproc 0

Name	Number	Value
$zero	0	0x00000000
$at	1	0x10010000
$v0	2	0x0000000a
$v1	3	0x00000000
$a0	4	0x00000060
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000008
$t1	9	0x0000000c
$t2	10	0x00000060
$t3	11	0x00000050
$t4	12	0x10010008
$t5	13	0x10010058
$t6	14	0x00000000
$t7	15	0x00000000
$s0	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
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x00400078
hi		0x00000000
lo		0x00000060

从I/O端口输出得到的结果正确，满足实验要求

实验总结

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` 来进行数据和地址的输出，最后实验结果正确