微机原理第一次实验报告

通信2303班 岳康 U202314327

一、实验任务

- 编写子程序PENO(&X,N,SP,SN)求长度为N的字类型数组X中所有正奇数的和和所有负偶数的和，并分别保存到SP 和SN 中。已知$a0 保存X 的地址，$a1 保存数组长度N，正奇数的和保存在$v0,负偶数的和保存在$v1 中。并编写主程序验证子程序功能,要求将计算结果输出到console。
- 测试以下数组序列：
  int X[10]={1,-4,8,-9,5,6,-10,19,22,23};
  Int X[10]={121,-124,138,-199,255,2566,-1034,1019,2032,2033};

二、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

三、实验环境

MIPS汇编和运行模拟器 Mars。
采用Java编写，需要J2SE Java运行时环境。

四、设计方案

要求正奇数和负偶数的和，则遍历数组，判断为正奇数则加入$v0中，判断为负偶数则加入$v1中

五、实验源代码

```
.data
    array:      .word 1,-4,8,-9,5,6,-10,19,22,23   # 数组
    array_len:  .word 10                           # 数组长度
    msg_addr:   .asciiz "Array address:"           # 显示数组地址的提示语
    msg_len:    .asciiz "\nArray length (N): "     # 显示数组长度的提示语
    msg_sp:     .asciiz "\nSum of positive odd numbers (SP): "  # 显示正奇数和
    msg_sn:     .asciiz "\nSum of negative even numbers (SN): " # 显示负偶数和
    newline:    .asciiz "\n"

.text
.globl main

main:
    # 打印数组地址提示语
    li $v0, 4
    la $a0, msg_addr
    syscall

    # 打印数组的实际地址
    li $v0, 34
    la $a0, array
    syscall

    # 打印数组长度提示语
    li $v0, 4
    la $a0, msg_len
    syscall

    # 打印数组长度 N
    li $v0, 1
    lw $a0, array_len
    syscall

    # 调用子程序 PENO(&array, N, SP, SN)
    la $a0, array       # 数组地址
    lw $a1, array_len   # 数组长度
    jal PENO            # 跳转调用子程序

    # 将返回值保存到 $s0（SP）和 $s1（SN）
    move $s0, $v0       # SP（正奇数和）
```

```
    move $s1, $v1       # SN（负偶数和）

    # 打印 SP 提示语
    li $v0, 4
    la $a0, msg_sp
    syscall

    # 打印 SP 值
    li $v0, 1
    move $a0, $s0
    syscall

    # 打印 SN 提示语
    li $v0, 4
    la $a0, msg_sn
    syscall

    # 打印 SN 值
    li $v0, 1
    move $a0, $s1
    syscall

    # 退出程序
    li $v0, 10
    syscall

===== PENO 子程序 =====
输入：$a0 = 数组地址，$a1 = 数组长度 N
输出：$v0 = SP（正奇数的和），$v1 = SN（负偶数的和）
PENO:
    li $v0, 0           # 初始化 SP = 0
    li $v1, 0           # 初始化 SN = 0
    li $t0, 0           # 初始化 i = 0（循环计数器）

peno_loop:
    bge $t0, $a1, peno_end  # 如果 i >= N，跳出循环

    lw $t1, 0($a0)          # 加载数组元素 X[i]

    # 先判断是否为正奇数
    ble $t1, $0, check_negative_even  # 如果 X[i] <= 0，跳过，进入检测负偶数
    andi $t2, $t1, 1        # 与1按位与，检查是否为奇数（最低位为 1）
    beq $t2, $0, check_negative_even  # 是偶数，跳过，进入检测负偶数
```

```
    add $v0, $v0, $t1       # SP += X[i]
    j next_element          # 跳到下一个元素

check_negative_even:
    bge $t1, $0, next_element  # 如果 X[i] >= 0，则此元素为0，跳过，进入下一个元素的检测
    andi $t2, $t1, 1        # 检查是否为奇数（最低位为 1）
    bne $t2, $0, next_element  # 如果是奇数，跳过
    add $v1, $v1, $t1       # SN += X[i]

next_element:
    addi $a0, $a0, 4        # 指针移动到下一个数组元素（每个元素占4字节）
    addi $t0, $t0, 1        # i++
    j peno_loop             # 回到循环开始

peno_end:
    jr $ra                  # 返回主程序
```

六、实验结果

1. 程序代码段映像

Address	Code	Basic	Source
0x00400000	0x24020004	addiu $2,$0,4	15: li $v0, 4
0x00400004	0x3c011001	lui $1,4097	16: la $a0, msg_addr
0x00400008	0x3424002c	ori $4,$1,44	
0x0040000c	0x0000000c	syscall	17: syscall
0x00400010	0x24020022	addiu $2,$0,34	20: li $v0, 34
0x00400014	0x3c011001	lui $1,4097	21: la $a0, array
0x00400018	0x34240000	ori $4,$1,0	
0x0040001c	0x0000000c	syscall	22: syscall
0x00400020	0x24020004	addiu $2,$0,4	25: li $v0, 4
0x00400024	0x3c011001	lui $1,4097	26: la $a0, msg_len
0x00400028	0x3424003b	ori $4,$1,59	
0x0040002c	0x0000000c	syscall	27: syscall
0x00400030	0x24020001	addiu $2,$0,1	30: li $v0, 1
0x00400034	0x3c011001	lui $1,4097	31: lw $a0, array_len
0x00400038	0x8c240028	lw $4,40($1)	
0x0040003c	0x0000000c	syscall	32: syscall
0x00400040	0x3c011001	lui $1,4097	35: la $a0, array # 数组地址
0x00400044	0x34240000	ori $4,$1,0	
0x00400048	0x3c011001	lui $1,4097	36: lw $a1, array_len # 数组长度
0x0040004c	0x8c250028	lw $5,40($1)	
0x00400050	0x0c100027	jal 0x0040009c	37: jal PENO # 跳转调用子程序
0x00400054	0x00028021	addu $16,$0,$2	40: move $s0, $v0 # SP（正奇数和）
0x00400058	0x00038821	addu $17,$0,$3	41: move $s1, $v1 # SN（负偶数和）
0x0040005c	0x24020004	addiu $2,$0,4	44: li $v0, 4
0x00400060	0x3c011001	lui $1,4097	45: la $a0, msg_sp
0x00400064	0x3424004f	ori $4,$1,79	
0x00400068	0x0000000c	syscall	46: syscall
0x0040006c	0x24020001	addiu $2,$0,1	49: li $v0, 1
0x00400070	0x00102021	addu $4,$0,$16	50: move $a0, $s0
0x00400074	0x0000000c	syscall	51: syscall
0x00400078	0x24020004	addiu $2,$0,4	54: li $v0, 4
0x0040007c	0x3c011001	lui $1,4097	55: la $a0, msg_sn

Address	Code	Basic
0x00400000	0x24020004	addiu $2, $0, 4
0x00400004	0x3c011001	lui $1, 4097
0x00400008	0x34240001	ori $4, $1, 1
0x0040000c	0x00000000	syscall
0x00400010	0x24020004	addiu $2, $0, 4
0x00400014	0x3c011001	lui $1, 4097
0x00400018	0x34240001	ori $4, $1, 59
0x0040001c	0x00000000	syscall
0x00400020	0x24020001	addiu $2, $0, 1
0x00400024	0x00000000	syscall
0x00400028	0x34240000	ori $4, $1, 0
0x0040002c	0x3c011001	lui $1, 4097
0x00400030	0x24020004	addiu $2, $0, 4
0x00400034	0x3c011001	lui $1, 4097
0x00400038	0x24240001	addiu $4, $4, 1
0x0040003c	0x00000000	syscall
0x00400040	0x00000000	move $s0, $v0
0x00400044	0x00000000	move $s1, $v1
0x00400048	0x24020004	addiu $2, $0, 4
0x0040004c	0x3c011001	lui $1, 4097
0x00400050	0x34240001	ori $4, $1, 1
0x00400054	0x00000000	syscall
0x00400058	0x00000000	move $a0, $s0
0x0040005c	0x24020004	addiu $2, $0, 4
0x00400060	0x3c011001	lui $1, 4097
0x00400064	0x34240001	ori $4, $1, 1
0x00400068	0x00000000	syscall
0x0040006c	0x00000000	move $a0, $s1

2. 程序数据段映像

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	1	-4	8	-9	5	6	-10	19
0x10010020	22	23	10	1634890305	1684086905	1936028260	167787123	1634890305
0x10010040	1701585017	1752459118	692987936	167780410	544044371	1881171567	1953067887	543520361
0x10010060	543450223	1651340654	544436837	693130024	167780410	544044371	1847617135	1952540517
0x10010080	543520361	1852143205	1836412448	1936876898	1314072608	2112041	10	0
0x100100a0	0	0	0	0	0	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

变量名	地址	数据	定义值
array	0x10010000	0x00	
	0x10010001	0x00	
	0x10010002	0x00	
	0x10010003	0x00	
	0x10010004	0x00	
	0x10010005	0x00	
	0x10010006	0x00	
	0x10010007	0x01	1
	0x10010008	0xff	
	0x10010009	0xff	
	0x1001000A	0xff	
	0x1001000B	0xff	
	0x1001000C	0xff	
	0x1001000D	0xff	
	0x1001000E	0xff	
	0x1001000F	0xfe	-8
.....	.....	.....	.....

3. 输入输出端口测试：

- 输入：{1,-4,8,-9,5,6,-10,19,22,23}
- 输出：

```
Array address:0x10010000
Array length (N): 10
Sum of positive odd numbers (SP): 48
Sum of negative even numbers (SN): -14
— program is finished running —
```

- 输入：{121,-124,138,-199,255,2566,-1034,1019,2032,2033}
- 输出：

```
Array address:0x10010000
Array length (N): 10
Sum of positive odd numbers (SP): 48
Sum of negative even numbers (SN): -14
-- program is finished running --
```

七、心得体会

本次实验使用了Mars软件进行汇编语言的学习和练习，学会了使用syscall来进行数据的输入和输出，第一次直观地感受了汇编语言的撰写、与人进行交互的过程，收获很大！