Tên: Nguyễn Chí Cường

MSSV: 19521299 Lớp: IT012.L14.2

1. Thực hành.

1.1 Sinh viên tìm hiểu tài liệu "Một số lệnh assembly MIPS cơ bản" và mô phỏng việc thực thi các lệnh và cho biết chức năng của các lệnh cơ bản sau:

add, addi, addu, addiu, sub, subu, and, andi, or, nor, lw, sw, slt, slti, sltu, sltiu, syscall

- Lệnh ***add, addu*** (Add, Add Unsigned)

Ý nghĩa: R[rd] = R[rs] + R[n] Format: R

2 lệnh ***add, addu*** dùng để cộng giá trị của 2 thanh ghi, và lưu kết quả vào thanh ghi đích. Cú pháp:

```
<tên lệnh> <thanh ghi đích>, <thanh ghi 1>, <thanh ghi 2>\
Example:
add $s0, $s1, $s2 # $s0 = $s1 + $s2
addu $s0, $s1, $s2 # $s0 = $s1 + $s2
```

- Lệnh ***addi, addiu*** (Add Immediate, Add Imm. Unsigned)

Ý nghĩa: R[rt] = R[rs] + SignExImm Format: I

2 lệnh ***addi, addiu*** dùng để cộng một thanh ghi với 1 hằng số, rồi lưu vào thanh ghi đích. Cú pháp:

```
<tên lệnh> <thanh ghi đích>, <thanh ghi>, <hằng số>
Example:
addi $s0, $s0, 123 # $s0 = $s0 + 123
addiu $s0, $s2, -123 # $s0 = $s2 - 123
```

- Lệnh ***sub, subu*** (Subtract, Subtract Unsigned)

Ý Nghĩa: R[rd] = R[rs] - R[rt] Format: R

2 lệnh ***sub, subu*** dùng để trừ giá trị của 2 thanh ghi, và lưu kết quả vào thanh ghi đích. Cú pháp:

```
<tên lệnh> <thanh ghi đích>, <thanh ghi 1>, <thanh ghi 2>
Example:
sub $s0, $s1, $s2 # $s0 = $s1 - $s2
subu $s0, $s1, $s2 # $s0 = $s1 - $s2
```

- Lệnh ***and*** (And)

Ý Nghĩa: R[rd] = R[rs] & R[rt] Format: R

and là phép toán logic "VÀ" nó sẽ thực hiện phép tính và của 2 thanh ghi và lưu vào thành ghi đích.

```
and <thanh ghi đích>, <thanh ghi 1>, <thanh ghi 2>
and $s0, $s1, $s2 # $s0 = $s1 & $s2
```

- Lệnh ***or*** (Or)

Ý Nghĩa: R[rd] = R[rs] | R[rt] Format: R

or là phép toán logic "HOẶC" nó sẽ thực hiện phép tính hoặc của 2 thanh ghi và lưu vào thành ghi đích.

```
or <thanh ghi đích>, <thanh ghi 1>, <thanh ghi 2>
or $s0, $s1, $s2 # $s0 = $s1 | $s2
```

- Lệnh ***nor*** (Nor)

Ý Nghĩa: R[rd] = ~ (R[rs] | R[rt]) Format: R

nor là phép toán logic "NOT OR" là phủ định của OR. Thực hiện phép tính hoặc của 2 thanh ghi sau đó lấy bù 1 và lưu vào thanh ghi đích.

```
nor <thanh ghi đích>, <thanh ghi 1>, <thanh ghi 2>
nor $s0, $s1, $s2 # $s0 = ~($s1 | $s2)
```

- Cách lệnh ***lw, sw*** (Load word, Store Word)

Lệnh chuyển dữ liệu từ bộ nhớ vào thanh ghi gọi là nạp (load) (viết tắt ***lw*** – load word).

Ýnghĩa: R[rt] = M[R[rs] + SignExtImm] Format: I

Định dạng của các lệnh nạp:

```
lw <thanh ghi đích>, SignExtImm * <thanh ghi>
lw $s1, 20($s0) # $s1 = x[5]
```

Lệnh chuyển dữ liệu từ thanh ghi ra bộ nhớ, gọi là lệnh lưu (store) (viết tắt ***sw*** – store word).

Ý nghĩa: M[R[rs] + SignExtImm] = R[rt] Format: I

Định dạng của các lệnh lưu:

```
sw <thanh ghi đích>, SignExtImm ( <thanh ghi> )
sw $s1, 8($s0) # x[2] = $s1
```

- Các lệnh điều kiện: ***slt, slti, sltu, sltiu***

Set Less Than *và* Set Less Than Unsigned (slt, sltu): là phép tính so sánh giá trị 2 thanh ghi.

Ý nghĩa: R[rd] = (R[rs] < R[rt]) ? 1:0 Format: R

Cú pháp:

```
slt(sltu) <thanh ghi đích>, <thanh ghi 1>, <thanh ghi 2>
slt(sltu)$s1, $s2, $s3 # if($s2 < $ $s3) $s1 = 1, else $s1 = 0
```

Set Less Than Imm *và* Set Less Than Imm Unsigned (slti, sltiu): là phép tính so sánh giá trị thanh ghi với 1 số.

Ý Nghĩa: R[rd] = (R[rs] < SignExtImm) ? 1:0 Format: I

Cú Pháp:

```
slt(sltu) <thanh ghi đích>, <thanh ghi 1>, <thanh ghi 2>
slt(sltu)$s1, $s2, 20 # if($s2 < $ 20) $s1 = 1, else $s1 = 0
```

- Lệnh ***sycall:*** Đây là lệnh nhập xuất.

1.2 Mô phỏng các chương trình bên dưới và có biết ý nghĩa của chương trình:

Vd1: Hình ảnh phần Text Segment khi chạy chương trình.

Bkpt	Address	Code	Basic	Source
	0x00400000	0x3c011001	lui $1,0x00001001	6: lw $t0, var1
	0x00400004	0x8c280000	lw $8,0x00000000($1)	
	0x00400008	0x24090005	addiu $9,$0,0x00000005	7: li $t1, 5
	0x0040000c	0x3c011001	lui $1,0x00001001	8: sw $t1, var1
	0x00400010	0xac290000	sw $9,0x00000000($1)	

2.2_vd1.asm	
__start	0x00400000
var1	0x10010000

Lúc này, giá trị của biến var1 được tạo có địa chỉ là 0x10010000.

Mô tả chương trình:

- Dòng lw $t0, var1: Load word địa chỉ của biến var1 vào trong thang ghi $t0. Đầu tiên là sẽ load $at = address của var1, sau đó lw $t0, 0($at).
- Dòng li $t1, 5: Load word giá trị 5 tức là 0x00000005 vào thanh ghi $t1.
- Dòng sw $t1, var1: Store word giá trị thanh ghi $at cho thanh ghi $t1. Tức địa chỉ tại giá trị của $at sẽ bằng giá trị của $t1 (giá trị tại địa chỉ 0x10010000 tại Value + 0 sẽ bằng giá trị của $t1 là 0x00000005).

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000005	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	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

Vd2:

2.2_vd2.asm	
__start	0x00400000
array1	0x10010000

Lúc này, giá trị của biến array1 được tạo có địa chỉ là 0x10010000.

Mô tả chương trình

☐	0x00400000	0x3c011001	lui $1,0x00001001	4: __start:	la	$t0, array1
☐	0x00400004	0x34280000	ori $8,$1,0x00000000			
☐	0x00400008	0x24090005	addiu $9,$0,0x00000005	5:	li	$t1, 5
☐	0x0040000c	0xad090000	sw $9,0x00000000($8)	6:	sw $t1, ($t0)	
☐	0x00400010	0x2409000d	addiu $9,$0,0x0000000d	7:	li $t1, 13	
☐	0x00400014	0xad090004	sw $9,0x00000004($8)	8:	sw $t1, 4($t0)	
☐	0x00400018	0x2409fff9	addiu $9,$0,0xfffffff9	9:	li $t1, -7	
☐	0x0040001c	0xad090008	sw $9,0x00000008($8)	10:	sw $t1, 8($t0)	

- Dòng la $t0, arrray1: Dòng này thực hiện load $at = address của array1. Sau đó thực hiện phép luận lí OR giá trị thanh ghi $at với ZeroExtImm (0x00000000) và lưu vào thanh ghi $t0.
- Dòng lệnh li $t1, 5: là load 0x00000005 vào thanh ghi $t1, hoặc có thể hiểu là addiu $t1, $zero, 0x00000005.
- Dòng lệnh sw $t1, ($t0): là Store word địa chỉ tại giá trị thanh ghi $t0 = $t1 (lúc này giá trị $t0 = 0x10010000 Value + 0, và giá trị tại địa chỉ này sẽ bằng giá trị của $t1 tức là 0x00000005)
- Dòng lệnh li $t1, 13: Này đơn giản là load word giá trị của thanh ghi $t1 bằng 13 hay là 0x0000000d.
- Dòng lệnh sw $t1, 4($t0): giá trị của địa chỉ là giá trị của $t0[Value + 4] bằng giá trị của thanh ghi $t1 = 0x0000000d.

- Dòng lệnh li $t1, -7: là load giá trị -7 tức là 0xfffffff9 vào thanh ghi $t1 (giá trị thanh ghi $t1 = 0xfffffff9).
- Dòng lệnh sw $t1, 8($t0): giá trị của địa chỉ là giá trị của $t0[Value + 8] bằng giá trị thanh ghi $t1 = 0xfffffff9.

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000005	0x0000000d	0xfffffff9	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	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

Vd3: Mô tả chương trình:

	0x00400000	0x24020005	addiu $2,$0,0x00000005	1: li $v0, 5
	0x00400004	0x0000000c	syscall	3: syscall

- Dòng lệnh li $v0, 5: load word giá trị của thanh ghi bằng 5 tức là 0x00000005.
- Dòng lệnh syscall: đây là dòng lệnh nhập một giá trị bất kì và nó sẽ đưa giá trị vào thanh ghi $v0 . Example: nhập giá trị 13 thì thanh ghi $v0 sẽ nhận giá trị là 0x0000000d.

Vd4: Đây là chương trình in ra chuỗi string1.

main	0x00400000
string1	0x10010000

Địa chỉ của biến string1 là 0x10010000.

Mô tả chương trình:

	0x00400000	0x24020004	addiu $2,$0,0x00000004	5: main:	li $v0, 4
	0x00400004	0x3c011001	lui $1,0x00001001	7:	la $a0, string1
	0x00400008	0x34240000	ori $4,$1,0x00000000		
	0x0040000c	0x0000000c	syscall	8:	syscall

- Dòng lệnh li $v0, 4: là load word giá trị 4 tức là 0x00000004 vào thanh ghi $v0 (giá trị thanh ghi $v0 = 0x00000004).
- Dòng lệnh la $a0, string1: Đầu tiên là load word địa chỉ của biến string1 vào vào thang ghi $at (giá trị thanh ghi $at = 0x10010000). Sau đó, thực hiện phép tính logic ori $a0, $at, $zero (tính phép tính $at or 0x00000000 và lưu vào thanh ghi $a0).
- Dòng lệnh syscall: lệnh này thực hiện in giá giá trị của biến string1 ra. Giá trị của biến string1 là chuỗi “Print this.”

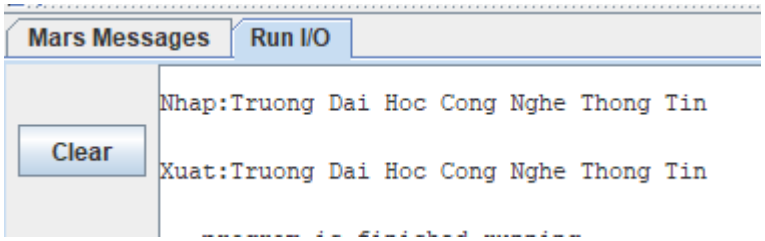
```
Print this.

-- program is finished running (dropped off bottom) --
```

2 **Bài tập:** Nhập vào một chuỗi, xuất ra cửa sổ I/O của MARS theo yêu cầu:

Xuất ra lại đúng chuỗi đã nhập:

Ví dụ:
Nhap: Truong Dai hoc Cong nghe Thong tin
Xuất: Truong Dai hoc Cong nghe Thong tin



Đây là kết quả khi chạy chương trình.

in	0x10010032
input	0x10010000
main	0x00400000
out	0x10010038

Gồm các biến là *in, out, input* có thông tin như hình bên.
Các biến được khởi tạo như bên dưới.

```
.data
        input: .space 50
        in: .asciiz "Nhap:"
        out: .asciiz "\nXuat:"
.text
```

Mô tả lại chương trình:

☐	0x00400000	0x24020004	addiu $2,$0,0x00000004	7:	li $v0,4
☐	0x00400004	0x3c011001	lui $1,0x00001001	8:	la $a0,in
☐	0x00400008	0x34240032	ori $4,$1,0x00000032		
☐	0x0040000c	0x0000000c	syscall	9:	syscall
☐	0x00400010	0x24020008	addiu $2,$0,0x00000008	11:	li $v0,8
☐	0x00400014	0x3c011001	lui $1,0x00001001	12:	la $a0, input
☐	0x00400018	0x34240000	ori $4,$1,0x00000000		
☐	0x0040001c	0x24050032	addiu $5,$0,0x00000032	13:	li $a1, 50
☐	0x00400020	0x0000000c	syscall	14:	syscall
☐	0x00400024	0x24020004	addiu $2,$0,0x00000004	16:	li $v0,4
☐	0x00400028	0x3c011001	lui $1,0x00001001	17:	la $a0,out
☐	0x0040002c	0x34240038	ori $4,$1,0x00000038		
☐	0x00400030	0x0000000c	syscall	18:	syscall
☐	0x00400034	0x3c011001	lui $1,0x00001001	20:	la $a0, input
☐	0x00400038	0x34240000	ori $4,$1,0x00000000		
☐	0x0040003c	0x24020004	addiu $2,$0,0x00000004	21:	li $v0, 4
☐	0x00400040	0x0000000c	syscall	22:	syscall

- Dòng Source 7, 8, 9: dùng để xuất ra chuỗi của biến in “Nhap:”.
- Dòng Source 11, 12, 13, 14: là dùng để làm bộ nhớ tạm lưu chuỗi sắp nhập vào biến input và có độ dài là 50 (li $a1, 50). Và tại dòng 14 là bắt đầu nhập chuỗi ví dụ: Truong Dai hoc Cong nghe Thong tin.
- Dòng Source 16, 17, 18 là xuất ra chuỗi của biến out “Xuat:”.
- Các dòng còn lại là dùng để xuất ra chuỗi vừa mới nhập ở trên.