**Báo cáo thực hành KTMT tuần 3**

**Phạm Đức Minh – 20215093**

**Assignment 1**

* Trường hợp 1: ta đặt j = 3, i = 5 (hay j < i)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Trạng thái chạy | Nội dung của thanh ghi | | | | | | |
| $s1 | $s2 | $t0 | $t1 | $t2 | $t3 | pc |
| Ban đầu | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000000 |
| Chạy lệnh addi thứ 1 | 0x00000000 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000004 |
| Chạy lệnh addi thứ 2 | 0x00000005 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000008 |
| Chạy lệnh slt | 0x00000005 | 0x00000003 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x4000000c |
| Chạy lệnh bne | 0x00000005 | 0x00000003 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x4000001c |
| Chạy lệnh addi thứ 1 của else | 0x00000005 | 0x00000003 | 0x00000001 | 0x00000000 | 0xffffffff | 0x00000000 | 0x40000020 |
| Chạy lệnh addi thứ 2 của else | 0x00000005 | 0x00000003 | 0x00000001 | 0x00000000 | 0xffffffff | 0x00000000 | 0x40000024 |

🡪 Như vậy, việc kiểm thử là đúng khi ta thấy j < i (với giá trị thanh ghi $s1 là 5 (0x00000005) và giá trị thanh ghi $s2 là 3 (0x00000003)) thì giá trị thanh ghi $t0 sẽ trả về 1 (địa chỉ 0x00000001) và giá trị thanh ghi này khác với thanh ghi $zero nên trình biên dịch đã thực hiện lệnh rẽ nhánh, từ đó gắn giá trị cho thanh ghi $t2 là -1 (địa chỉ là 0xffffffff) và gắn giá trị thanh ghi $t3 bằng 2 lần giá trị chính nó

* Trường hợp 2: ta đặt j = 5, i = 3 (hay j >= i)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Trạng thái chạy | Nội dung của thanh ghi | | | | | |
| $s1 | $s2 | $t0 | $t1 | $t3 | pc |
| Ban đầu | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000000 |
| Chạy lệnh addi thứ 1 | 0x00000000 | 0x00000005 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000004 |
| Chạy lệnh addi thứ 2 | 0x00000003 | 0x00000005 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000008 |
| Chạy lệnh slt | 0x00000005 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000000 | 0x4000000c |
| Chạy lệnh bne | 0x00000005 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000010 |
| Chạy lệnh addi thứ 1 của start | 0x00000005 | 0x00000003 | 0x00000000 | 0x00000001 | 0x00000000 | 0x40000014 |
| Chạy lệnh addi thứ 2 của start | 0x00000005 | 0x00000003 | 0x00000000 | 0x00000001 | 0x00000001 | 0x40000018 |
| Kết thúc lệnh endif | 0x00000005 | 0x00000003 | 0x00000000 | 0x00000001 | 0x00000001 | 0x40000024 |

🡪 Như vậy, việc kiểm thử là đúng khi ta thấy j >= i (với giá trị thanh ghi $s1 là 3 (0x00000003) và giá trị thanh ghi $s2 là 5 (0x00000005)) thì giá trị thanh ghi $t0 sẽ trả về 0 (địa chỉ 0x00000000) và giá trị thanh ghi này đúng với thanh ghi $zero nên trình biên dịch không thực hiện lệnh rẽ nhánh mà tiếp tục các lệnh addi, từ đó gắn giá trị cho thanh ghi $t0 là 1 (0x00000001) và gán giá trị thanh ghi $t3 là 1 (0x00000001)

Cả 2 trường hợp trên, giá trị thanh ghi pc đều tăng lên 4 khi thực hiện từng câu lệnh.

🡺Kết luận: Chương trình trên đã chạy đúng khi giá trị thanh ghi $v0 = 1

**Assignment 2**

Graphical user interface, text

Description automatically generated

Ban đầu, ta sẽ nạp địa chỉ và nạp giá trị cho các biến, khi đó có sự thay đổi chung về địa chỉ và nội dung của các thanh ghi $s1 đến $s5 và thanh ghi $t6 đến $t9. Sau đó sẽ xảy ra các trường hợp sau:

* Trường hợp 1: i = 0, n = 6, step = 1 (tức i < n)

Như vậy, khi chạy lệnh slt (i <= n - 1), $t2 sẽ trả về giá trị là 1, khi đó sẽ không thực hiện lệnh beq (rẽ nhánh) mà tiếp tục việc thay đổi giá trị $t1 để chứa giá trị A[i], và mỗi lần loop có giá trị A[i] khác nhau tương ứng với giá trị $t1 khác nhau để lưu trữ, và giá trị của thanh ghi $s1 sẽ tăng lên 1 (tương ứng với step = 1), và giá trị cuối cùng của thanh ghi sẽ là $s5 là 15 (đã đúng với lý thuyết). Khi giá trị i = 6 thì slt trả về giá trị 0, trình biên dịch chạy lệnh beq và kết thúc bước nhảy.

* Trường hợp 2: i = 6, n = 5, step = 1 (tức i >= n)

Khi đó, hàm slt trả về giá trị 0, kết thúc bước nhảy, khi đó không có sự thay đổi nào về giá trị của các thanh ghi kể trên.

* Trường hợp 3: Khi ta thêm một lượng phần tử lớn hơn n

Sẽ tương tự như trường hợp 1, tuy vậy nếu thêm giá trị thử n + 1 trở đi cho arr thì giá trị của thanh ghi $s5 sẽ không thay đổi khi cộng giá trị của A[i>=n]

* Chương trình trên đã chạy đúng khi giá trị thanh ghi $v0 = 1

**Assignment 3**

Ta lần lượt nạp các giá trị tương ứng vào thanh ghi, khi đó sẽ có sự thay đổi tương ứng giá trị của các thanh ghi $s0, $s1 và $t0 đến $t2, sau đó sẽ có 4 trường hợp xảy ra như sau:

* Trường hợp 1: test = 0 (.word 0)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Trạng thái chạy | | Nội dung của thanh ghi | | | | | | | |
| $s0 | $s1 | $s2 | $t0 | $t1 | $t2 | pc | $at |
| Ban đầu | | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000000 | 0x00000000 |
| Chạy lệnh la | Chạy lệnh lui | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000004 | 0x10010000 |
| Chạy lệnh ori | 0x10010000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000008 | 0x10010000 |
| Chạy lệnh lw | | 0x10010000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x4000000c | 0x10010000 |
| Chạy lệnh li (1) | | 0x10010000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000010 | 0x10010000 |
| Chạy lệnh li (2) | | 0x10010000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000000 | 0x40000014 | 0x10010000 |
| Chạy lệnh li (3) | | 0x10010000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000018 | 0x10010000 |
| Chạy beq (1) | | 0x10010000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000028 | 0x10010000 |
| Chạy addi của case 0 | | 0x10010000 | 0x00000000 | 0x00000001 | 0x00000000 | 0x00000001 | 0x00000002 | 0x4000002c | 0x10010000 |
| Chạy j continue | | 0x10010000 | 0x00000000 | 0x00000001 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000040 | 0x10010000 |

🡪Khi đó, lệnh beq thứ nhất sẽ trả về 1 và chuyển sang case\_0, khi đó giá trị của thanh ghi của thanh ghi $s2 sẽ là 1 khi dùng lệnh addi

* Trường hợp 2: test = 1 (.word 1)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Trạng thái chạy | | Nội dung của thanh ghi | | | | | | | |
| $s0 | $s1 | $s2 | $t0 | $t1 | $t2 | pc | $at |
| Ban đầu | | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000000 | 0x00000000 |
| Chạy lệnh la | Chạy lệnh lui | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000004 | 0x10010000 |
| Chạy lệnh ori | 0x10010000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000008 | 0x10010000 |
| Chạy lệnh lw | | 0x10010000 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x4000000c | 0x10010000 |
| Chạy lệnh li (1) | | 0x10010000 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000010 | 0x10010000 |
| Chạy lệnh li (2) | | 0x10010000 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000000 | 0x40000014 | 0x10010000 |
| Chạy lệnh li (3) | | 0x10010000 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000018 | 0x10010000 |
| Chạy beq (1) | | 0x10010000 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x4000001c | 0x10010000 |
| Chạy beq (2) | | 0x10010000 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000030 | 0x10010000 |
| Chạy sub của Case 1 | | 0x10010000 | 0x00000001 | 0xffffffff | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000034 | 0x10010000 |
| Chạy j continue | | 0x10010000 | 0x00000001 | 0xffffffff | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000040 | 0x10010000 |

🡪Khi đó, lệnh beq thứ nhất sẽ trả về 0 và lệnh beq thứ hai trả về 1 và chuyển sang case\_1, khi đó giá trị của thanh ghi của thanh ghi $s2 sẽ là -1 khi thực hiện lệnh sub

* Trường hợp 3: test = 2 (.word 2)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Trạng thái chạy | | Nội dung của thanh ghi | | | | | | | |
| $s0 | $s1 | $s3 | $t0 | $t1 | $t2 | pc | $at |
| Ban đầu | | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000000 | 0x00000000 |
| Chạy lệnh la | Chạy lệnh lui | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000004 | 0x10010000 |
| Chạy lệnh ori | 0x10010000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000008 | 0x10010000 |
| Chạy lệnh lw | | 0x10010000 | 0x00000002 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x4000000c | 0x10010000 |
| Chạy lệnh li (1) | | 0x10010000 | 0x00000002 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000010 | 0x10010000 |
| Chạy lệnh li (2) | | 0x10010000 | 0x00000002 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000000 | 0x40000014 | 0x10010000 |
| Chạy lệnh li (3) | | 0x10010000 | 0x00000002 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000018 | 0x10010000 |
| Chạy beq (1) | | 0x10010000 | 0x00000002 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x4000001c | 0x10010000 |
| Chạy beq (2) | | 0x10010000 | 0x00000002 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000020 | 0x10010000 |
| Chạy beq (3) | | 0x10010000 | 0x00000002 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000038 | 0x10010000 |
| Chạy add của case 2 | | 0x10010000 | 0x00000002 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x4000003c | 0x10010000 |
| Chạy j continue | | 0x10010000 | 0x00000002 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000040 | 0x10010000 |

🡪Khi đó, lệnh beq thứ nhất và thứ hai sẽ trả về giá trị 0, lệnh beq thứ ba sẽ trả về giá trị 1, khi đó trình biên dịch chuyển sang case\_2, khi đó giá trị của thanh ghi của thanh ghi $s3 sẽ là 2 lần chính nó.

* Trường hợp 4: test = 3 (hay test > 2) (.word 3)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Trạng thái chạy | | Nội dung của thanh ghi | | | | | | | |
| $s0 | $s1 | $s2 | $t0 | $t1 | $t2 | pc | $at |
| Ban đầu | | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000000 | 0x00000000 |
| Chạy lệnh la | Chạy lệnh lui | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000004 | 0x10010000 |
| Chạy lệnh ori | 0x10010000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000008 | 0x10010000 |
| Chạy lệnh lw | | 0x10010000 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x4000000c | 0x10010000 |
| Chạy lệnh li (1) | | 0x10010000 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x40000010 | 0x10010000 |
| Chạy lệnh li (2) | | 0x10010000 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000000 | 0x40000014 | 0x10010000 |
| Chạy lệnh li (3) | | 0x10010000 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000018 | 0x10010000 |
| Chạy beq (1) | | 0x10010000 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x4000001c | 0x10010000 |
| Chạy beq (2) | | 0x10010000 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000020 | 0x10010000 |
| Chạy beq (3) | | 0x10010000 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000024 | 0x10010000 |
| Chạy j default | | 0x10010000 | 0x00000003 | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x40000040 | 0x10010000 |

🡪Khi đó, cả 3 lệnh beq sẽ trả về giá trị 0, từ đó chạy về default về kết thúc chương trình

🡺Kết luận: Chương trình trên đã chạy đúng khi giá trị thanh ghi $v0 = 1

**Assignment 4**

1. i < j

#Laboratory Exercise 3, Home Assignment 1

addi $s2, $zero, 3 #j

addi $s1, $zero, 5 #i

start:

slt $t0,$s1,$s2 # i < j

beq $t0,$zero,else # branch to else if i >= j

addi $t1,$t1,1 # then part: x=x+1

addi $t3,$zero,1 # z=1

j endif # skip “else” part

else: addi $t2,$t2,-1 # begin else part: y=y-1

add $t3,$t3,$t3 # z=2\*z

endif:

li $v0, 1 # service 1 is print integer

syscall

* Khi giá trị i, j như trên

Background pattern

Description automatically generated

* Khi hoán vị 2 giá trị



* Chương trình đã chạy khi giá trị thanh ghi $v0 = 1

1. i >= j

#Laboratory Exercise 3, Home Assignment 1

addi $s2, $zero, 3 #j

addi $s1, $zero, 5 #i

start:

slt $t0,$s1,$s2 # i < j

bne $t0,$zero,else # branch to else if i < j

addi $t1,$t1,1 # then part: x=x+1

addi $t3,$zero,1 # z=1

j endif # skip “else” part

else: addi $t2,$t2,-1 # begin else part: y=y-1

add $t3,$t3,$t3 # z=2\*z

endif:

li $v0, 1 # service 1 is print integer

syscall

* Khi giá trị i, j như trên



* Khi hoán vị 2 giá trị

Background pattern

Description automatically generated

* Chương trình đã chạy khi giá trị thanh ghi $v0 = 1

1. i + j <= 0

#Laboratory Exercise 3, Home Assignment 1

addi $s2, $zero, 3 #j

addi $s1, $zero, 5 #i

add $s3, $s1, $s2 # $s3 = $s1 + $s2 (i + j)

start:

slt $t0,$zero,$s3 # i + j > 0

bne $t0,$zero,else # branch to else if i + j > 0

addi $t1,$t1,1 # then part: x=x+1

addi $t3,$zero,1 # z=1

j endif # skip “else” part

else: addi $t2,$t2,-1 # begin else part: y=y-1

add $t3,$t3,$t3 # z=2\*z

endif:

li $v0, 1 # service 1 is print integer

syscall

* Khi giá trị i, j như trên

Background pattern

Description automatically generated

* Khi ta thay i = -3, j = -5



1. i + j > m + n

#Laboratory Exercise 3, Home Assignment 1

addi $s2, $zero, 3 #j

addi $s1, $zero, 5 #i

add $s3, $s1, $s2 #i+j

addi $s4, $zero, 3 #m

addi $s5, $zero, 5 #n

add $s6, $s4, $s5 #m + n

start:

slt $t0,$s3,$s6 # i + j > m + n

beq $t0,$zero,else # branch to else if i + j <= m + n

addi $t1,$t1,1 # then part: x=x+1

addi $t3,$zero,1 # z=1

j endif # skip “else” part

else: addi $t2,$t2,-1 # begin else part: y=y-1

add $t3,$t3,$t3 # z=2\*z

endif:

li $v0, 1 # service 1 is print integer

syscall

* Khi giá trị i, j, m, n như trên

Background pattern

Description automatically generated

* Khi ta thay m = 1, n = 2



**Assignment 5**

1. i < n

#Laboratory 3, Home Assigment 2

.data

arr: .word 1, 0, 3, 4, 5, 6

.text

addi $s1, $zero, 0 # $s1 = i

la $s2, arr # loading array

addi $s3, $zero, 6 # n

addi $s4, $zero, 1 # step

addi $s5, $zero, 0 # sum

loop: slt $t2, $s1, $s3 # $t2 = i < n ? 1 : 0

bne $t2, $zero, endloop #end the loop when $t2 (i < n)

add $t1, $s1, $s1 # $t1 = 2 \* $s1

add $t1, $t1, $t1 # $t1 = 4 \* $s1

add $t1, $t1, $s2 # $t1 store the address of A[i]

lw $t0, 0($t1) # load value of A[i] in $t0

add $s5, $s5, $t0 # sum = sum + A[i]

add $s1, $s1, $s4 # i = i + step

j loop # goto loop

endloop:

li $v0, 1 # service 1 is print integer

syscall

* Khi thực hiện code trên



* Khi ta thêm phần tử 7, 8, 9, 10 vào array và đổi i = 7 thì vòng lặp sẽ chạy vô hạn nhưng khi dừng ở giá trị A[9]



* Chương trình đã chạy đúng khi thanh ghi $v0 = 1

1. i <= n

#Laboratory 3, Home Assigment 2

.data

arr: .word 1, 0, 3, 4, 5, 6

.text

addi $s1, $zero, 0 # $s1 = i

la $s2, arr # loading array

addi $s3, $zero, 6 # n

addi $s4, $zero, 1 # step

addi $s5, $zero, 0 # sum

loop: slt $t2, $s3, $s1 # $t2 = i > n ? 1 : 0

beq $t2, $zero, endloop #end the loop when $t2 = 0 (i <= n)

add $t1, $s1, $s1 # $t1 = 2 \* $s1

add $t1, $t1, $t1 # $t1 = 4 \* $s1

add $t1, $t1, $s2 # $t1 store the address of A[i]

lw $t0, 0($t1) # load value of A[i] in $t0

add $s5, $s5, $t0 # sum = sum + A[i]

add $s1, $s1, $s4 # i = i + step

j loop # goto loop

endloop:

li $v0, 1 # service 1 is print integer

syscall

* Khi chạy code trên: giá trị $s5 = 0
* Khi thêm các phần từ 7, 8 vào arr, thay step = -1, giá trị $s5 = 8
* Chương trình trên đã chạy đúng

1. sum >= 0

#Laboratory 3, Home Assigment 2

.data

arr: .word 1, 0, 3, 4, 5, 6

.text

addi $s1, $zero, 0 # $s1 = i

la $s2, arr # loading array

addi $s3, $zero, 6 # n

addi $s4, $zero, 1 # step

addi $s5, $zero, 0 # sum

loop: slt $t2, $s5, $zero # $t2 = sum < 0 ? 1 : 0

beq $t2, $zero, endloop #end the loop when $t2 = 0 (sum >= 0)

add $t1, $s1, $s1 # $t1 = 2 \* $s1

add $t1, $t1, $t1 # $t1 = 4 \* $s1

add $t1, $t1, $s2 # $t1 store the address of A[i]

lw $t0, 0($t1) # load value of A[i] in $t0

add $s5, $s5, $t0 # sum = sum + A[i]

add $s1, $s1, $s4 # i = i + step

j loop # goto loop

endloop:

li $v0, 1 # service 1 is print integer

syscall

* Khi chạy chương trình trên: giá trị $s5 = 0
* Khi thay $s5 ban đầu là -2, giá trị $s5 = 2
* Khi thay $s5 ban đầu là -1, đổi A[0] = -5, giá trị $s5 = 1

1. A[i] ==0

#Laboratory 3, Home Assigment 2

.data

arr: .word 1, 0, 3, 4, 5, 6

.text

addi $s1, $zero, 0 # $s1 = i

la $s2, arr # loading array

addi $s3, $zero, 6 # n

addi $s4, $zero, 1 # step

addi $s5, $zero, 0 # sum

loop:

add $t1, $s1, $s1 # $t1 = 2 \* $s1

add $t1, $t1, $t1 # $t1 = 4 \* $s1

add $t1, $t1, $s2 # $t1 store the address of A[i]

lw $t0, 0($t1) # load value of A[i] in $t0

beq $t0, $zero, endloop #end the loop when $t0 = 0 (A[i] == 0)

add $s5, $s5, $t0 # sum = sum + A[i]

add $s1, $s1, $s4 # i = i + step

j loop # goto loop

endloop:

li $v0, 1 # service 1 is print integer

syscall

* Khi thực hiện chương trình trên, giá trị thanh ghi $s5 = 1
* Khi thay A[1] = 4, A[2] = 0, giá trị thanh ghi $s5 = 5
* Như vậy, chương trình trên đã chạy đúng

**Assignment 6**

.data

arr: .word -5, -15, 6, 4, 5, -9

.text

addi $s1, $zero, 0 # i

la $s2, arr # loading array

addi $s3, $zero, 6 # n

addi $s4, $zero, 1 # step

addi $s5, $zero, 0 # the greatest absolute value in the array

loop: slt $t2, $s1, $s3 # i < n

beq $t2, $zero, endloop

add $t1, $s1, $s1

add $t1, $t1, $t1

add $t1, $t1, $s2

lw $t0, 0($t1) # load the value of A[i]

start\_1: # Check if A[i] is negative or not

slt $t3, $t0, $zero #A[i] < 0

beq $t3, $zero, start\_2 # if A[i] >=0, then brench. If not, do the following code.

sub $t0 $zero, $t0 # If A[i] < 0, the absolute value of A[i] is making by changing the sign

start\_2: # Find the greatest absolute value of A[i]

slt $t3, $t0, $s5 # Check if A[i] < the greatest absolute value

bne $t3, $zero, endif # If the above statement is true, then brench. If not, do the following

addi $s5, $t0, 0 # Update the new value

endif:

add $s1, $s1, $s4 # i = i + step

j loop

endloop:

li $v0, 1 # service 1 is print integer

syscall

* Khi chạy chương trình trên, giá trị của thanh ghi $s5 = 15
* Khi ta cho dãy 4, 5, -7, 0, 2, 9, giá trị của thanh ghi $s5 = 9
* Chương trình đã chạy đúng