ASSIGNMENT WEEK 3

-Họ tên: Hồ Tuấn Huy
-MSSV: 20225856

Assignment 1:

Assignment 1

Create a new project to implement the code in Home Assignment 1. Initialize for i and j variable. Compile and upload to the simulator. Run this program step by step, observe the changing of memory and the content of registers at each step.

```
1   start:
2   li $s1 2
3   li $s2 7
4   slt $t0, $s2, $s1
5   bne $t0, $zero, else
6   addi $t1, $t1, 1
7   addi $t3, $zero, 2
8   j endif
9   else:
10  addi $t2, $t2, -1
11  add $t3, $t3, $t3
12  endif:
```

=>slt: so sánh 2 thanh ghi $s2 và $s1 (j và i), kết quả lưu vào thanh ghi $t0.

-Nếu thanh ghi t0 != 0 (i<=j) thì:

x=x+1

z=2

Sau đó jump đến endif (bỏ qua else).

-Nếu t0=0 (j<i) thì jump đến else:

y=y-1

z=2z

Trong trường hợp trong bài ta có i=2 và j=7 nên t0 != 0. Do đó x=x+1 và z=2.

$t1	9	0x00000001
$t2	10	0x00000000
$t3	11	0x00000002

Assignment 2:

Assignment 2

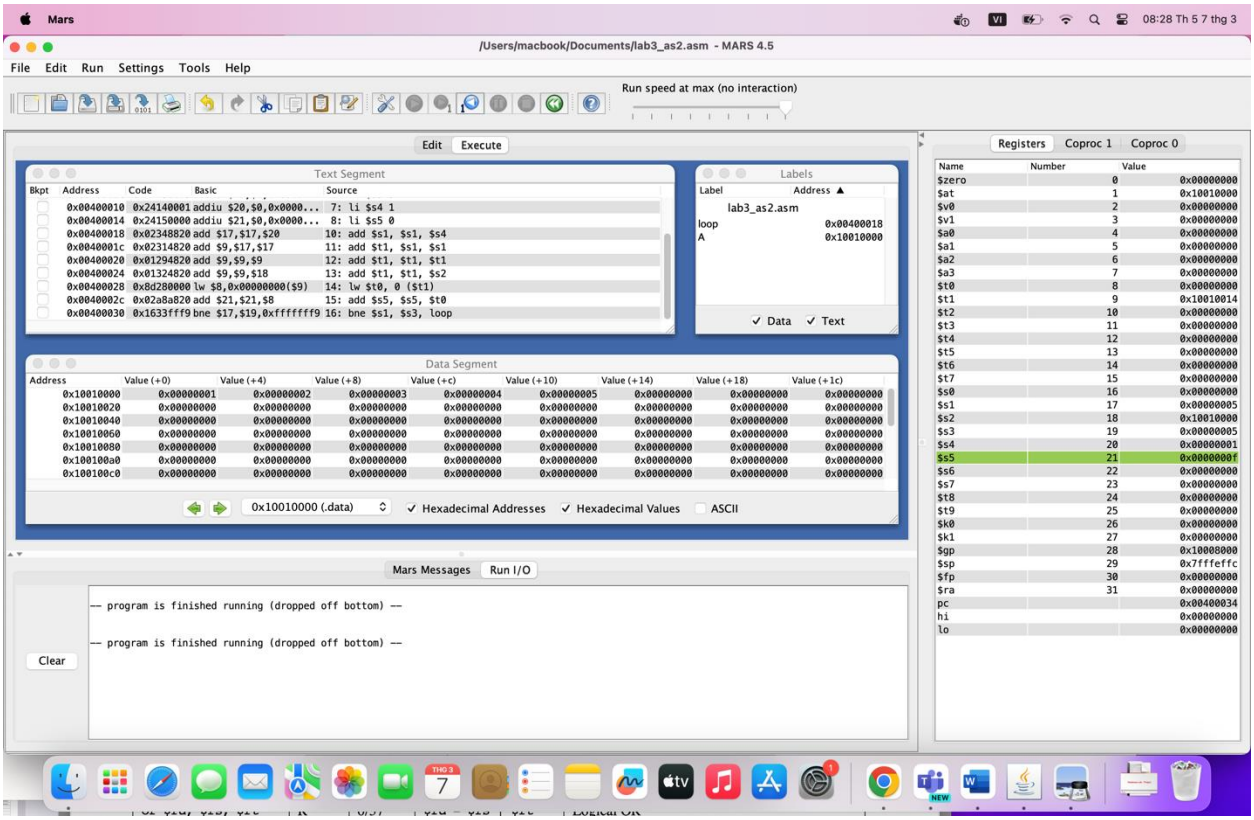
Create a new project implementing the code in Home Assignment 2. Initialize for i, n, step, sum variables and array A. Compile and upload to the simulator. Run this program step by step, observe the changing of memory and the content of registers by each step. Try to test with some more cases (change the value of variables).

```
1   .data
2   a: .word 1, 2, 3, 4, 5
3   .text
4   li $s1, -1
5   la $s2, A
6   li $s3 5
7   li $s4 1
8   li $s5 0
9   loop:
10  add $s1, $s1, $s4
11  add $t1, $s1, $s1
12  add $t1, $t1, $t1
13  add $t1, $t1, $s2
14  lw $t0, 0 ($t1)
15  add $s5, $s5, $t0
16  bne $s1, $s3, loop
17  # $s1: i
18  # $s2: A
19  # $s3: n
20  # $s4: step
21  # $s5: sum
```

Do A có 5 phần tử => Để tính tổng các phần tử trong A thì phải lặp 5 lần (vì thế nên n=4 thì lặp 5 bước do i=-1).

Mà: Do để tính cả phần tử đầu tiên trong A (với vòng lặp trên) thì offset phải ở ngay trước phần tử đầu tiên của mảng, tức là i phải bằng -1.

=>Ta thu được kết quả sum (A)=15, được lưu trong thanh ghi $s5:



Assignment 3:

Assignment 3

Create a new project implementing the code in Home Assignment 3. Compile and upload to the simulator. Run this program step by step; observe the changing of memory and the content of registers by each step. Change the value of test variable and run this program some times to check all cases.

```
1  .data
2  test: .word 2
3  .text
4  la $s0, test #load the address of test variable
5  lw $s1, 0($s0) #load the value of test to register $t1
6  li $t0, 0 #load value for test case
7  li $t1, 1
8  li $t2, 2
9  li $s2, 4
10 li $s3, 8
11 beq $s1, $t0, case_0
12 beq $s1, $t1, case_1
13 beq $s1, $t2, case_2
14 j default
15 case_0: addi $s2, $s2, 1 #a=a+1
16 j continue
17 case_1: sub $s2, $s2, $t1 #a=a-1
18 j continue
19 case_2: add $s3, $s3, $s3 #b=2*b
20 j continue
21 default:
22 continue:
```

Ở đây test=2 =>Thực hiện case_2=>b=16: (0x00000010 tương đương 16 trong hệ thập phân)

$s3	19	0x00000010

Assignment 4:

Assignment 4

Modify the Assignment 1, so that the condition tested is

a. i <j
b. i >= j
c. i+j <= 0
d. i+j > m+n

---

a) i<j:

```
.data
.text
addi      $s1,$zero,2                 #s1=i
addi      $s2,$zero,7                 #s2=J

start :             #i<j
slt       $t0,$s1,$s2                 #s1<s2(I<J) thi t0=1 nguoc lai t0=0
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:
```

Sửa 2 lệnh ngay sau start thành:

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

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

Vì i = 2 < j = 7 nên kết quả là x= 1 và z = 1:

$t1	9	1
$t2	10	0
$t3	11	1

b) i >=j:

```
.data
.text
addi      $s1,$zero,2                 #s1=i
addi      $s2,$zero,7                 #s2=J

start :             # i >= j
slt       $t0,$s1,$s2                 #s1<s2(I<J) thi t0=1 nguoc lai t0=0
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:
```

I < j nên z = 2z = 0, y = y-1 = -1:

$t1	9	0
$t2	10	-1
$t3	11	0

c) i+j <=0:

```
.data
.text
addi      $s1,$zero,2                 #s1=i
addi      $s2,$zero,7                 #s2=J
add       $s3,$s1,$s2                 #s3=i+j

start :             # i + j <= 0
slt       $t0,$zero,$s3               # 0 < i+j
bne       $t0,$zero,else              # branch to else if 0 < 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:
```

Vì i+j = 9 > 0 nên kết quả y = -1,z =0:

$t0	8	1
$t1	9	0
$t2	10	-1

d)i+j>m+n:

Đầu tiên ta khởi tạo giá trị m, n và tính m+n lưu vào $s6

```
addi      $s4,$zero,4                 #s4=m
addi      $s5,$zero,5                 #s5=n
add       $s6,$s4,$s5                 #s6=m+n
```

Sau đó so sánh $s6(m+n) với $s3(i+j)

```
.data
.text
addi      $s1,$zero,2                 #s1=i
addi      $s2,$zero,7                 #s2=J
add       $s3,$s1,$s2                 #s3=i+j
addi      $s4,$zero,4                 #s4=m
addi      $s5,$zero,5                 #s5=n
add       $s6,$s4,$s5                 #s6=m+n

start :             # i + j > m + n
slt       $t0,$s6,$s3                 # m + n < i + j
beq       $t0,$zero,else              # branch to else if m + n >= 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:
```

Do i+j = m+n = 9 nên kết quả y = -1, z = 0:

$t1	9	0
$t2	10	-1
$t3	11	0

Assignment 5:

Assignment 5

Modify the Assignment 2, so that the condition tested at the end of the loop is

a. i < n
b. i <= n
c. sum >= 0
d. A[i] == 0

---

a) i<n:

```
# Lab Ex3, Assignment 5
.data
        A: .word 1,2,3,4,5
.text
        li $s1, -1          # i = -1
        la $s2, A           #s2 store the address of array A
        li $s3, 5           # number of elements of A
        li $s4, 1           # step
        li $s5, 0           # sum

loop:           # i < n
        add $s1, $s1, $s4   # i = i + step
        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]
        slt $t2, $s1, $s3   # i < n
        bne $t2, $zero,loop # if i < n, goto loop
```

TH i!=n và i<n giống nhau nên kết quả sum = 15 :

$s4	20	1
$s5	21	15
$s6	22	0

b) i<=n:

Do A có 5 phần tử nên dù i có chạy tới n thì kết quả sum = 15 không đổi.

```
# Lab Ex3, Assignment 5
.data
        A: .word 1,2,3,4,5
.text
        li $s1, -1          # i = -1
        la $s2, A           #s2 store the address of array A
        li $s3, 5           # number of elements of A
        li $s4, 1           # step
        li $s5, 0           # sum

loop:           # i <= n
        add $s1, $s1, $s4   # i = i + step
        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]
        slt $t2, $s3, $s1   # n < i
        beq $t2, $zero,loop # if i <= n, goto loop
```

c) sum>=0:

```
# Lab Ex3, Assignment 5
.data
          A: .word 1,-2,3,-4,5
.text
          li $s1, -1          # i = -1
          la $s2, A           #s2 store the address of array A
          li $s3, 5           # number of elements of A
          li $s4, 1           # step
          li $s5, 0           # sum

loop:               # sum >= 0
          add $s1, $s1, $s4   # i = i + step
          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]
          slt $t2, $s5, $zero # sum < 0
          beq $t2, $zero,loop # if sum >= 0, goto loop
```

Kết quả sum = -1:

$s4	20	1
$s5	21	-1
$s6	22	0

Do ngay tại step 2, sum = 1 + (-2) = -1 <0 => vòng lặp bị dừng lại.

d) A[i] == 0:

Do vòng lặp trên dạng do while, step 1: sum = 1 , A[i] = 1 != 0 => break

```
# Lab Ex3, Assignment 5
.data
          A: .word 1,0,3,-4,5
.text
          li $s1, -1          # i = -1
          la $s2, A           #s2 store the address of array A
          li $s3, 5           # number of elements of A
          li $s4, 1           # step
          li $s5, 0           # sum

loop:               # A[i] == 0
          add $s1, $s1, $s4   # i = i + step
          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]
          beq $t0, $zero,loop # if A[i] == 0, goto loop
```

$s4	20	1
$s5	21	1
$s6	22	0
$s7	23	0

Assignment 6:

Assignment 6

Using all of above instructions and statements, create a new project to implement this function: find the element with the largest absolute value in a list of integers. Assuming that this list is store in an integer array and we know the number of elements in it.

```
1   .data
2   A: .word 1, -2, 6, -4, 5, -7
3   .text
4   li $s1, -1
5   la $s2, A
6   li $s3, 6
7   li $s4, 1
8   li $s5  0
9   li $s6, 0
10  li $s7, 1
11  loop:
12  add $s1, $s1, $s4
13  add $t1, $s1, $s1
14  add $t1, $t1, $t1
15  add $t1, $t1, $s2
16  lw $t0, 0($t1)
17  start:
18  slt $t6, $t0, $zero
19  bne $t6, $zero, else
20  slt $t7, $t0, $s5
21  beq $t7, $s6, case_0
22  beq $t7, $s7, case_1
23  j default
24  case_0:
25  add $s5, $t0, $zero
```

```
14  add $t1, $t1, $t1
15  add $t1, $t1, $s2
16  lw $t0, 0($t1)
17  start:
18  slt $t6, $t0, $zero
19  bne $t6, $zero, else
20  slt $t7, $t0, $s5
21  beq $t7, $s6, case_0
22  beq $t7, $s7, case_1
23  j default
24  case_0:
25  add $s5, $t0, $zero
26  case_1:
27  j continue
28  default:
29  continue:
30  j endif
31  else:
32  mul $t0, $t0, -1
33  slt $t7, $t0, $s5
34  beq $t7, $s6, case_0
35  beq $t7, $s7, case_1
36  j default
37  endif:
38  bne $s1, $s3, loop
```

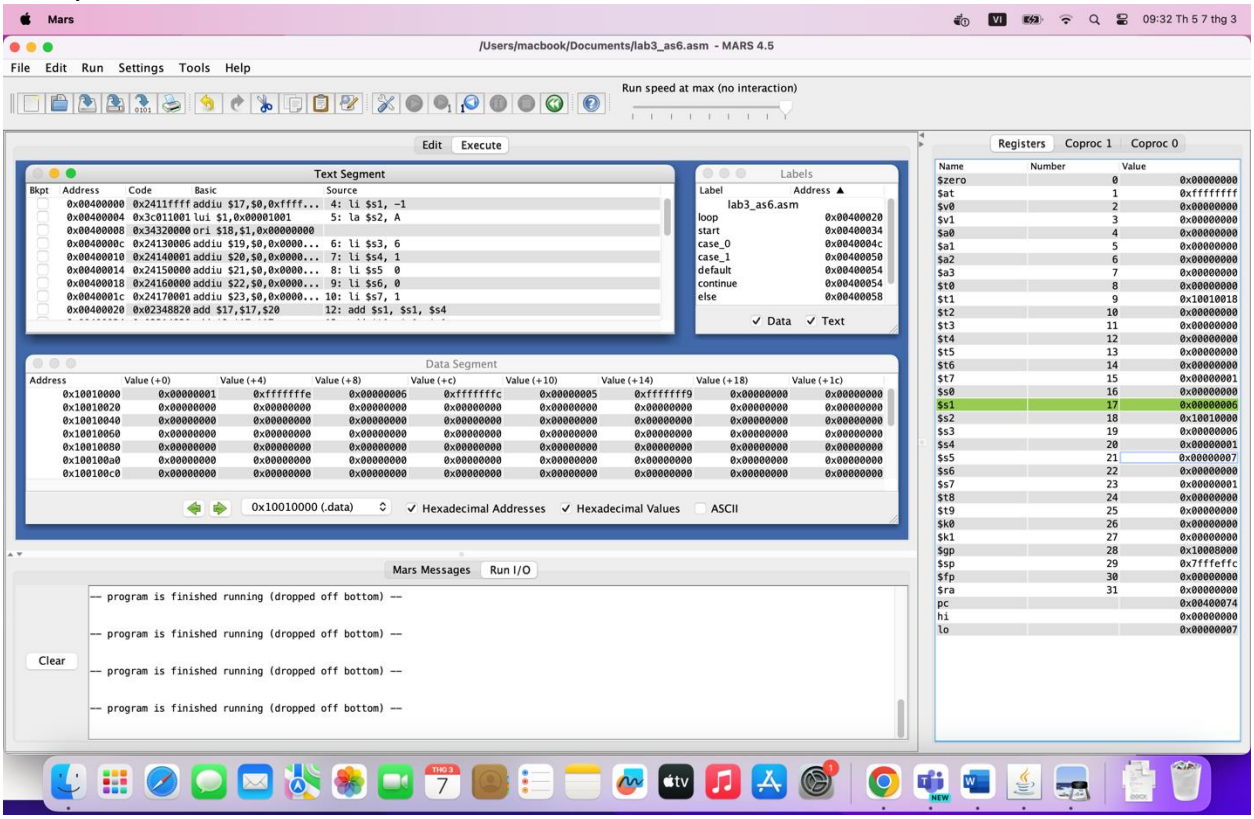
-Tạo một mảng A có 6 phần tử do đó vòng loop sẽ phải lặp 6 lần.
-Lưu địa chỉ của mảng A vào $s2 và lưu giá trị A[i] vào $t0.
-Khai báo giá trị tuyệt đối-max : li $s5, 0 (max = 0).
-Tạo hàm if else, so sánh xem giá trị A[i] < 0 hay không và trả về giá trị 1(TRUE), 0(FALSE) lưu vào $t6.
* bne $t6, $zero, else : so sánh nếu $t6 != 0 (A[i] < 0) thì thực hiện else, nếu $t6 = 0 (A[i] >=0) thì thực hiện tiếp các lệnh tiếp theo
+ Nếu A[i] >= 0 : thì abs của A[i] bằng chính nó , ta sẽ so sánh xem A[i] < max hay không và trả về giá trị 1(TRUE), 0(FALSE) lưu vào $t7.

Ở đây ta sử dụng switch case để test giá trị của $t7:

+ $t7 = 0 (A[i] >= max) : ta sẽ gán giá trị cho max = A[i]
+ $t7 != 0 (A[i] < max) => break

+ Nếu A[i] < 0 : Lấy abs của A[i] bằng cách lấy đối của nó , sau đó lại so sánh giá trị A[i] với max và thực hiện lệnh switch case như trên.

Kết quả : max = 7


Mars
09:32 Th 5 7 thg 3
/Users/macbook/Documents/lab3_as6.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute
Text Segment
Bkpt Address Code Basic Source
0x00400000 0x2411ffff addiu $17,$0,0xffff... 4: li $s1, -1
0x00400004 0x3c011001 lui $1,0x00001001 5: la $s2, A
0x00400008 0x34320000 ori $18,$1,0x00000000
0x0040000c 0x24130006 addiu $19,$0,0x0000... 6: li $s3, 6
0x00400010 0x24140001 addiu $20,$0,0x0000... 7: li $s4, 1
0x00400014 0x24150000 addiu $21,$0,0x0000... 8: li $s5 0
0x00400018 0x24160000 addiu $22,$0,0x0000... 9: li $s6, 0
0x0040001c 0x24170001 addiu $23,$0,0x0000... 10: li $s7, 1
0x00400020 0x02348820 add $17,$17,$20 12: add $s1, $s1, $s4
Labels
Label Address
lab3_as6.asm
loop 0x00400020
start 0x00400034
case_0 0x0040004c
case_1 0x00400050
default 0x00400054
continue 0x00400054
else 0x00400058
Data Text
Data Segment
Address Value (+0) Value (+4) Value (+8) Value (+c) Value (+10) Value (+14) Value (+18) Value (+1c)
0x10010000 0x00000001 0xfffffffe 0x00000006 0xfffffffc 0x00000005 0xfffffff9 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
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Registers Coproc 1 Coproc 0
Name Number Value
$zero 0 0x00000000
$at 1 0xffffffff
$v0 2 0x00000000
$v1 3 0x00000000
$a0 4 0x00000000
$a1 5 0x00000000
$a2 6 0x00000000
$a3 7 0x00000000
$t0 8 0x00000000
$t1 9 0x10010018
$t2 10 0x00000000
$t3 11 0x00000000
$t4 12 0x00000000
$t5 13 0x00000000
$t6 14 0x00000000
$t7 15 0x00000001
$s0 16 0x00000000
$s1 17 0x00000006
$s2 18 0x10010000
$s3 19 0x00000006
$s4 20 0x00000001
$s5 21 0x00000007
$s6 22 0x00000000
$s7 23 0x00000001
$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 0x00400074
hi 0x00000000
lo 0x00000007
Mars Messages Run I/O
Clear
-- program is finished running (dropped off bottom) --
-- program is finished running (dropped off bottom) --
-- program is finished running (dropped off bottom) --
-- program is finished running (dropped off bottom) --