

## Báo cáo tuần 7

### Bài 1 :

```

1  .text
2  main:
3      li $a0, -440 #load input parameter
4      jal abs #jump and link to abs procedure
5      nop
6      add $s0, $zero, $v0
7      li $v0, 10 #terminate
8      syscall
9  endmain:
10 #-----
11 # function abs
12 # param[in] $a0 the interger need to be gained the absolute
13 value:
14 # return $v0 absolute value
15 #-----
16 abs:
17     sub $v0,$zero,$a0 #put -(a0) in v0; in case (a0)<0
18     bltz $a0,done #if (a0)<0 then done
19     nop
20     add $v0,$a0,$zero #else put (a0) in v0
21
22 done:
23     jr $ra
24

```

Run

Text Segment				
Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x2404fe48	addiu \$4,\$0,-440	3: li \$a0, -440 #load input parameter
<input type="checkbox"/>	0x00400004	0x0c100006	jal 0x00400018	4: jal abs #jump and link to abs procedure
<input type="checkbox"/>	0x00400008	0x00000000	nop	5: nop
<input type="checkbox"/>	0x0040000c	0x00028020	add \$16,\$0,\$2	6: add \$s0, \$zero, \$v0
<input type="checkbox"/>	0x00400010	0x2402000a	addiu \$2,\$0,10	7: li \$v0, 10 #terminate
<input type="checkbox"/>	0x00400014	0x0000000c	syscall	8: syscall
<input type="checkbox"/>	0x00400018	0x00041022	sub \$2,\$0,\$4	17: sub \$v0,\$zero,\$a0 #put -(a0) in v0; in case (a
<input type="checkbox"/>	0x0040001c	0x04800002	bltz \$4,2	18: bltz \$a0,done #if (a0)<0 then done
<input type="checkbox"/>	0x00400020	0x00000000	nop	19: nop
<input type="checkbox"/>	0x00400024	0x00801020	add \$2,\$4,\$0	20: add \$v0,\$a0,\$zero #else put (a0) in v0
<input type="checkbox"/>	0x00400028	0x03e00008	jr \$31	23: jr \$ra

Giải thích code

Đầu tiên ta cần nạp giá trị vào biến a0

Tiếp đó ta nhảy tới abs

Ta lưu v0 = số đối của A

Ta kiểm tra xem a0 có < 0 hay không

Nếu như mà nhỏ hơn 0 thì nhảy tới done

Done nhảy tới địa chỉ của ra tức là địa chỉ của nop đầu tiên

The screenshot shows the Mars MIPS simulator interface. The 'Text Segment' window displays assembly code with addresses from 4194304 to 4194344. The 'Data Segment' window shows memory addresses from 268500992 to 268501120. The register window on the right shows the values of registers \$zero through \$ra. The register \$v0 is highlighted with a green background and contains the value 440. The register \$ra is highlighted with a red background and contains the value 4194312. The 'Mars Messages' and 'Run I/O' buttons are visible at the bottom.

Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	440
\$v1	3	0
\$a0	4	-440
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	0
\$t1	9	0
\$t2	10	0
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	0
\$s1	17	0
\$s2	18	0
\$s3	19	0
\$s4	20	0
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$t8	24	0
\$t9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	4194312
pc		4194312
hi		0
lo		0

Nếu như  $a0 < 0$  vậy thì số đối của nó tức là  $v0$  đang  $> 0$  và ta thực hiện nhảy tới nop đầu tiên, nạp giá trị vào s0 và gọi lệnh syscall để kết thúc chương trình

Còn nếu như mà  $a0 > 0$  tức là số đối  $v0$  đang nhỏ hơn 0

Thì ta thực hiện lệnh  $v0 = a0 + 0$  để  $v0 > 0 = a0$

Sau đó nhảy tới nop đầu tiên và in ra lệnh như trên

## Bài 2 :

### Code

```

1
2 .text
3 main:  li $a0,2 #load test input
4        li $a1,4
5        li $a2,0
6        jal max #call max procedure
7        nop
8        j  endmain
9
10 max:   add $v0,$a0,$zero #copy (a0) in v0; largest so far
11        sub $t0,$a1,$v0 #compute (a1)-(v0)
12        bltz $t0,okay #if (a1)-(v0)<0 then no change
13        nop
14        add $v0,$a1,$zero #else (a1) is largest thus far
15 okay:  sub $t0,$a2,$v0 #compute (a2)-(v0)
16        bltz $t0,done #if (a2)-(v0)<0 then no change
17        nop
18        add $v0,$a2,$zero #else (a2) is largest overall
19 done:  add $v0,$a2,$zero #else (a2) is largest overall
20
21        jr $ra #return to calling program
22
23 endmain:

```

### Run

The screenshot shows the MARS MIPS simulator interface. The main window displays the assembly code from the previous block, with line numbers 1 through 23. The code is assembled into machine code, and the assembly window shows the corresponding instructions and their addresses.

Below the assembly window, the "Data Segment" window shows memory addresses and values. The values are all 0, indicating that the memory has been initialized to zero.

On the right side, the "Registers" window shows the state of the MIPS registers. The registers are labeled with their names and values. The registers are:

Register	Value
\$ra	0
\$t0	0
\$t1	0
\$t2	0
\$t3	0
\$t4	0
\$t5	0
\$t6	0
\$t7	0
\$t8	0
\$t9	0
\$s0	0
\$s1	0
\$s2	0
\$s3	0
\$s4	0
\$s5	0
\$s6	0
\$s7	0
\$s8	0
\$s9	0
\$k0	0
\$k1	0
\$gp	26846...
\$sp	21474...
\$fp	0
\$ra	0
\$pc	4194304
\$hi	0
\$lo	0

The status bar at the bottom shows the assembly path: "Assemble: assembling D:\BTRTMT\week7\B2.asm".

Giải thích code

Đầu tiên ta nhập các giá trị để tìm max vào lần lượt  $a_0$ ,  $a_1$ ,  $a_2$

Sau đó ta nhảy tới max

Ta lấy  $v_0 =$  giá trị đầu tiên  $= a_0$

Ta thực hiện lệnh trừ  $t_0 = a_1 - v_0$  tức là  $a_1 - a_0$

Vậy nếu như mà  $< 0$  thì nhảy tới okay (vì ở đây  $v_0$  được coi là max, và lúc này  $a_0$  đang lớn hơn  $a_1$ )

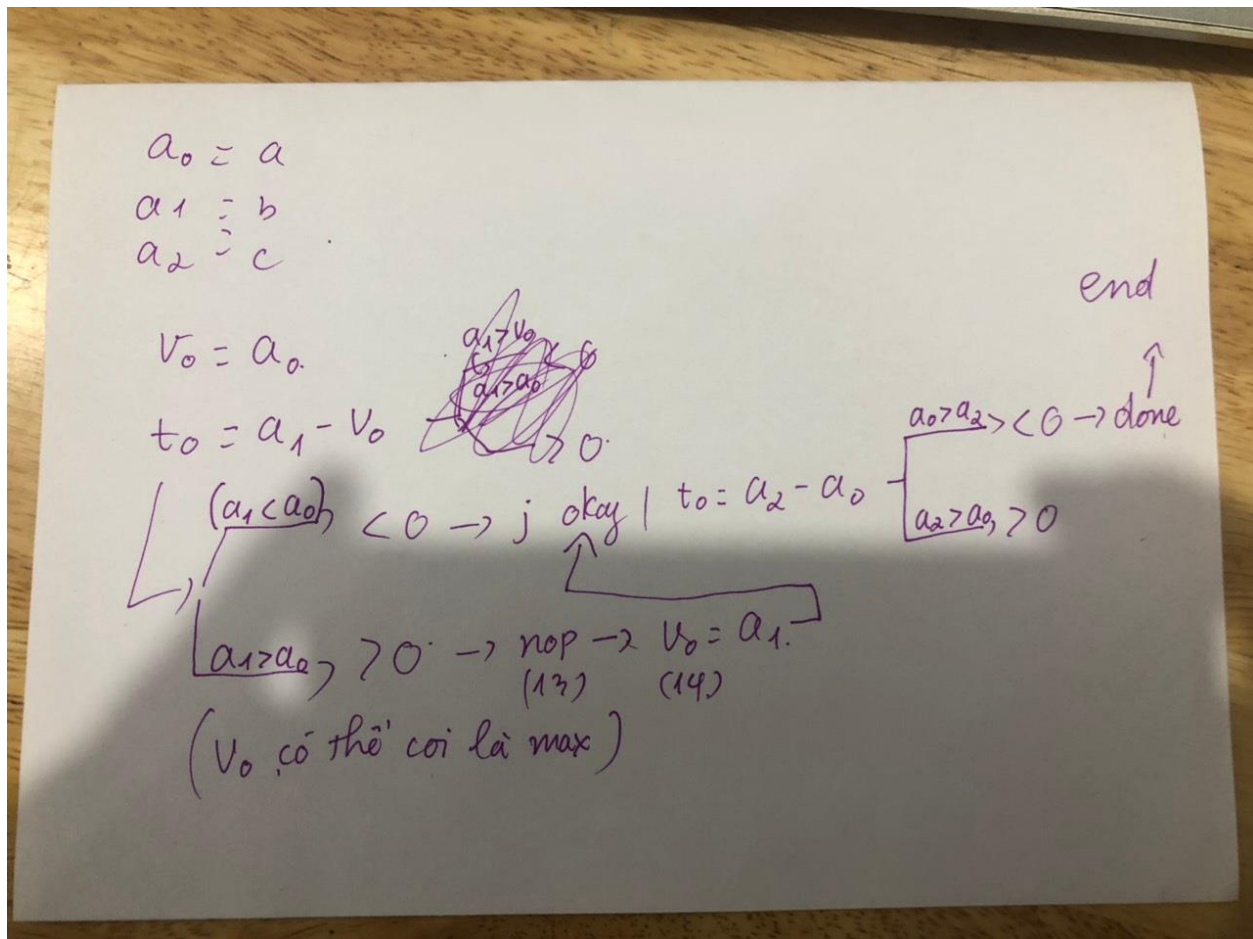
Ta thực hiện tiếp lệnh số 2 là mang  $a_0$  đi so sánh với  $a_2$

Tương tự như vậy ta nhảy tới done để nhảy tới nop kết thúc chương trình

Còn ở bước trên nếu như  $t_0 > 0$  tức là  $v_0 = a_1$  vì lúc này  $a_1$  đang lớn hơn  $a_0$

Vậy nên ta có bước số 2 là mang  $v_0$  đi so sánh với  $a_2$  tức là mang  $a_1$  đi so sánh với  $a_2$

Để cho dễ hiểu ta có thể coi thuật toán mà em trình bày ở đây



### Bài 3