

Báo cáo thực hành kiến trúc máy tính

Assignment 1:

Home Assignment 1

This home assignment implements “if-then-else” statement using some fundamental instructions, such as `slt`, `addi`, `jump` and `branch`.

```
if (i<=j)
    x=x+1;
    z=1;
else
    y=y-1;
    z=2*z;
```

```
week3_a1.asm
1  #Laboratory Exercise 3, Home Assignment 1
2
3  .text
4      addi $s1, $zero 2 # i = 2
5      addi $s2, $zero 1 # j = 1
6
7      addi $t1, $zero 2 # x = 2
8      addi $t2, $zero 3 # y = 3
9      addi $t3, $zero 4 # z = 4
10 start:
11     slt $t0,$s2,$s1 # j<i
12     bne $t0,$zero,else # branch to else if j<i
13     addi $t1,$t1,1 # then part: x=x+1
14     addi $t3,$zero,1 # z=1
15     j endif # skip "else" part
16 else:
17     addi $t2,$t2,-1 # begin else part: y=y-1
18     add $t3,$t3,$t3 # z=2*z
19 endif:
```

Khởi tạo giá trị $i = 2, j = 1, x = 2, y = 3, z = 4$

\$t1	9	0x00000002
\$t2	10	0x00000003
\$t3	11	0x00000004
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000002
\$s2	18	0x00000001

- So sánh j và i ($1 < 2$) $\text{t0} = 1$

Edit Execute		Registers		Cproc 1	Cproc 0		
Text Segment v w							
Bkpt	Address	Code	Basic	Source			
<input type="checkbox"/>	0x00400000	0x20110002	addi \$t7,\$0,0x00000...	4:	add \$s1, \$zero 2 # i = 2		
<input type="checkbox"/>	0x00400004	0x20120001	addi \$t8,\$0,0x00000...	5:	add \$s2, \$zero 1 # j = 1		
<input type="checkbox"/>	0x00400008	0x20090002	addi \$9,\$0,0x00000002	7:	add \$t1, \$zero 2 # x = 2		
<input type="checkbox"/>	0x0040000c	0x200a0003	addi \$t0,\$0,0x00000...	8:	add \$t2, \$zero 3 # y = 3		
<input type="checkbox"/>	0x00400010	0x200b0004	addi \$t1,\$0,0x00000...	9:	add \$t3, \$zero 4 # z = 4		
<input type="checkbox"/>	0x00400014	0x0251402a	sllt \$t0,\$t8,\$t7	11:	sllt \$t0,\$s2,\$s1 # j<i		
<input type="checkbox"/>	0x00400018	0x15000003	bne \$t8,\$0,0x00000003	12:	bne \$t0,\$zero,else # bra..		
<input type="checkbox"/>	0x0040001c	0x21290001	addi \$9,\$9,0x00000001	13:	add \$t1,\$t1,1 # then pa..		
<input type="checkbox"/>	0x00400020	0x200b0001	addi \$t1,\$0,0x00000...	14:	addi \$t3,\$zero,1 # z=1		
<input type="checkbox"/>	0x00400024	0x0810000c	j 0x00400030	15:	j endif # skip "else" part		
<input type="checkbox"/>	0x00400028	0x214affff	addi \$t0,\$t0,0xffff...	17:	add \$t2,\$t2,-1 # begin ...		
<input type="checkbox"/>	0x0040002c	0x016b5820	add \$t1,\$t1,\$t1	18:	add \$t3,\$t3,\$t3 # z=2*z		
<input checked="" type="checkbox"/> Data <input checked="" type="checkbox"/> Text							
Data Segment							
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
<input checked="" type="checkbox"/> Hexadecimal Addresses <input checked="" type="checkbox"/> Hexadecimal Values <input type="checkbox"/> ASCII							
Name	Number	Value					
\$zero	0	0x00000000					
\$at	1	0x00000000					
\$v0	2	0x00000000					
\$v1	3	0x00000000					
\$a0	4	0x00000000					
\$a1	5	0x00000000					
\$a2	6	0x00000000					
\$a3	7	0x00000000					
\$t0	8	0x00000001					
\$t1	9	0x00000002					
\$t2	10	0x00000003					
\$t3	11	0x00000004					
\$t4	12	0x00000000					
\$t5	13	0x00000000					
\$t6	14	0x00000000					
\$t7	15	0x00000000					
\$s0	16	0x00000000					
\$s1	17	0x00000002					
\$s2	18	0x00000001					
\$s3	19	0x00000000					
\$s4	20	0x00000000					
\$s5	21	0x00000000					
\$s6	22	0x00000000					
\$s7	23	0x00000000					
\$t8	24	0x00000000					
\$t9	25	0x00000000					
\$k0	26	0x00000000					
\$k1	27	0x00000000					
\$gp	28	0x10008000					
\$sp	29	0x7ffffeffc					
\$fp	30	0x00000000					
\$ra	31	0x00000000					

So sánh \$t0 = 1, \$zero = 0 nhảy đến lệnh else

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x20110002	addi \$t7,\$0,0x0000...	4: addi \$a1, \$zero 2 # i = 2
<input type="checkbox"/>	0x00400004	0x20120001	addi \$t8,\$0,0x0000...	5: addi \$a2, \$zero 1 # j = 1
<input type="checkbox"/>	0x00400008	0x20090002	addi \$t9,\$0,0x00000002	7: addi \$t1, \$zero 2 # x = 2
<input type="checkbox"/>	0x0040000c	0x200a0003	addi \$t10,\$0,0x0000...	8: addi \$t2, \$zero 3 # y = 3
<input type="checkbox"/>	0x00400010	0x200b0004	addi \$t11,\$0,0x0000...	9: addi \$t3, \$zero 4 # z = 4
<input type="checkbox"/>	0x00400014	0x0251402a	slt \$t8,\$t8,\$t7	11: slt \$t0,\$a2,\$a1 # j<i
<input type="checkbox"/>	0x00400018	0x15000003	bne \$t8,\$0,0x00000003	12: bne \$t0,\$zero,else # bra...
<input type="checkbox"/>	0x0040001c	0x21290001	addi \$t9,\$t9,0x00000001	13: addi \$t1,\$t1,1 # then pa...
<input type="checkbox"/>	0x00400020	0x200b0001	addi \$t11,\$0,0x0000...	14: addi \$t3,\$zero,1 # z=1
<input type="checkbox"/>	0x00400024	0x0810000c	j 0x00400030	15: j endif # skip "else" part
<input type="checkbox"/>	0x00400028	0x214affff	addi \$t10,\$t10,0xffff...	17: addi \$t2,\$t2,-1 # begin ...
<input type="checkbox"/>	0x0040002c	0x016b5820	add \$t11,\$t11,\$t11	18: add \$t3,\$t3,\$t3 # z=2*z

Labels

Label	Address
week3_a1.asm	
start	0x00400014
endif	0x00400028
else	0x00400030

☒ Data
 ☒ Text

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x

←

→

0x10010000 (.data)

☒ Hexadecimal Addresses
 ☒ Hexadecimal Values
 ☐ ASCII

Mars Messages

Run I/O

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000001
\$t1	9	0x00000002
\$t2	10	0x00000003
\$t3	11	0x00000004
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$a0	16	0x00000000
\$a1	17	0x00000002
\$a2	18	0x00000001
\$a3	19	0x00000000
\$a4	20	0x00000000
\$a5	21	0x00000000
\$a6	22	0x00000000
\$a7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7fffffc0
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400028
hi		0x00000000
lo		0x00000000

- $Tính\ y = y - 1\ (3 - 1 = 2)$

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x20110002	addi \$t7,\$0,0x00000...	4: addi \$s1, \$zero 2 # i = 2
<input type="checkbox"/>	0x00400004	0x20120001	addi \$t8,\$0,0x00000...	5: addi \$s2, \$zero 1 # j = 1
<input type="checkbox"/>	0x00400008	0x20090002	addi \$t9,\$0,0x00000002	7: addi \$t1, \$zero 2 # x = 2
<input type="checkbox"/>	0x0040000c	0x200a0003	addi \$t0,\$0,0x00000...	8: addi \$t2, \$zero 3 # y = 3
<input type="checkbox"/>	0x00400010	0x200b0004	addi \$t1,\$0,0x00000...	9: addi \$t3, \$zero 4 # z = 4
<input type="checkbox"/>	0x00400014	0x0251402a	slt \$t0,\$t8,\$t7	11: slt \$t0,\$s2,\$s1 # j<i
<input type="checkbox"/>	0x00400018	0x15000003	bne \$t0,\$0,0x00000003	12: bne \$t0,\$zero,else # bra...
<input type="checkbox"/>	0x0040001c	0x21290001	addi \$t5,\$5,0x00000001	13: addi \$t1,\$t1,1 # then pa...
<input type="checkbox"/>	0x00400020	0x200b0001	addi \$t1,\$0,0x00000...	14: addi \$t3,\$zero,1 # z=1
<input type="checkbox"/>	0x00400024	0x0810000c	j 0x00400030	15: j endif # skip "else" part
<input type="checkbox"/>	0x00400028	0x214affff	addi \$t0,\$t0,0xffff...	17: addi \$t2,\$t2,-1 # begin ...
<input type="checkbox"/>	0x0040002c	0x016b5820	add \$t1,\$t1,\$t1	18: add \$t3,\$t3,\$t3 # z=2*z

Labels

Label	Address
week3_a1.asm	
start	0x00400014
else	0x00400028
endif	0x00400030

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x

Mars Messages

Run I/O

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x00000001
\$t2	10	0x00000002
\$t3	11	0x00000004
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000002
\$s2	18	0x00000001
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7ffffffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x0040002c
hi		0x00000000
lo		0x00000000

- $Tính\ z = z * 2\ (2 * 4 = 8)$

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x20110002	addi \$t7,\$0,0x00000...	4: addi \$s1, \$zero 2 # i = 2
<input type="checkbox"/>	0x00400004	0x20120001	addi \$t8,\$0,0x00000...	5: addi \$s2, \$zero 1 # j = 1
<input type="checkbox"/>	0x00400008	0x20090002	addi \$t9,\$0,0x00000002	7: addi \$t1, \$zero 2 # x = 2
<input type="checkbox"/>	0x0040000c	0x200a0003	addi \$t0,\$0,0x00000...	8: addi \$t2, \$zero 3 # y = 3
<input type="checkbox"/>	0x00400010	0x200b0004	addi \$t1,\$0,0x00000...	9: addi \$t3, \$zero 4 # z = 4
<input type="checkbox"/>	0x00400014	0x0251402a	slt \$t0,\$t8,\$t7	11: slt \$t0,\$s2,\$s1 # j<i
<input type="checkbox"/>	0x00400018	0x15000003	bne \$t0,\$0,0x00000003	12: bne \$t0,\$zero,else # bra...
<input type="checkbox"/>	0x0040001c	0x21290001	addi \$t5,\$5,0x00000001	13: addi \$t1,\$t1,1 # then pa...
<input type="checkbox"/>	0x00400020	0x200b0001	addi \$t1,\$0,0x00000...	14: addi \$t3,\$zero,1 # z=1
<input type="checkbox"/>	0x00400024	0x0810000c	j 0x00400030	15: j endif # skip "else" part
<input type="checkbox"/>	0x00400028	0x214affff	addi \$t0,\$t0,0xffff...	17: addi \$t2,\$t2,-1 # begin ...
<input type="checkbox"/>	0x0040002c	0x016b5820	add \$t1,\$t1,\$t1	18: add \$t3,\$t3,\$t3 # z=2*z

Labels

Label	Address
week3_a1.asm	
start	0x00400014
else	0x00400028
endif	0x00400030

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x

Mars Messages

Run I/O

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x00000002
\$t2	10	0x00000002
\$t3	11	0x00000008
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000002
\$s2	18	0x00000001
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7ffffffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400030
hi		0x00000000
lo		0x00000000

Assignment 2:

Home Assignment 2

The following example demonstrates how to implement loop statement. This program computes the sum of elements of array A.

With C language:

```
sum = 0;
for (int i = 0; i < n; i += step)
    sum += A[i];
```

With assembly language:

```
sum = 0
i = 0
loop:  if (i >= n) goto endloop
        sum = sum + A[i]
        i = i + step
        goto loop
endloop:
```

```
week3_a2.asm
1  #Laboratory 3, Home Assignment 2
2  .data
3  A: .word 4,1,5,10,5
4  .text
5      addi $s5, $zero, 0 # sum = 0
6      addi $s1, $zero, 0 # i = 0
7
8      la    $s2, A      # gan dia chi dau mang A cho $s2
9      li    $s3, 5      # n = 5
10     li    $s4, 1      # step = 1
11
12 loop: slt $t2, $s1, $s3 # $t2 = i < n ? 1 : 0
13       beq $t2, $zero, endloop
14       add $t1, $s1, $s1 # $t1 = 2 * $s1
15       add $t1, $t1, $t1 # $t1 = 4 * $s1
16       add $t1, $t1, $s2 # $t1 store the address of A[i]
17       lw  $t0, 0($t1) # load value of A[i] in $t0
18       add $s5, $s5, $t0 # sum = sum + A[i]
19       add $s1, $s1, $s4 # i = i + step
20       j   loop # goto loop
21 endloop:
```


Assignment 3:

Home Assignment 3

A switch/case statement allows multiway branching based on the value of an integer variable. In the following example, the switch variable test can assume one of the three values in [0, 2] and a different action is specified for each case.

```
switch(test) {  
    case 0:  
        a=a+1; break;  
    case 1:  
        a=a-1; break;  
    case 2:  
        b=2*b; break;  
}
```

TH test =0

```
week3_a3.asm  
1  #Laboratory Exercise 3, Home Assignment 3  
2  .data  
3  test: .word 0  
4  .text  
5      la $s0, test #load the address of test variable  
6      lw $s1, 0($s0) #load the value of test to register $t1  
7      li $t0, 0 #load value for test case  
8      li $t1, 1  
9      li $t2, 2  
10     beq $s1, $t0, case_0  
11     beq $s1, $t1, case_1  
12     beq $s1, $t2, case_2  
13     j default  
14     case_0:  
15         addi $s2, $s2, 1 #a=a+1  
16         j continue  
17     case_1:  
18         sub $s2, $s2, $t1 #a=a-1  
19         j continue  
20     case_2:  
21         add $s3, $s3, $s3 #b=2*b  
22         j continue  
23     default:  
24     continue:
```

- Test = 0 => \$s1 =0 chạy vào case 0: a = a +1 (a=0+1=1)

The screenshot shows the MARS MIPS simulator interface. The **Text Segment** window displays the assembly code for `week3_a3.asm`, with addresses and sources. The **Data Segment** window shows memory values for addresses from `0x10010000` to `0x10010120`. The **Registers** window shows the state of registers, with `$s2` highlighted at address `0x10010000` containing the value `1`.

Register	Name	Number	Value
\$zero		0	0
\$at		1	268500992
\$v0		2	0
\$v1		3	0
\$a0		4	0
\$a1		5	0
\$a2		6	0
\$a3		7	0
\$t0		8	0
\$t1		9	1
\$t2		10	2
\$t3		11	0
\$t4		12	0
\$t5		13	0
\$t6		14	0
\$t7		15	0
\$s0		16	268500992
\$s1		17	0
\$s2		18	1
\$s3		19	0
\$s4		20	0
\$s5		21	0
\$s6		22	0
\$s7		23	0
\$s8		24	0
\$s9		25	0
\$k0		26	0
\$k1		27	0
\$gp		28	268468224
\$sp		29	2147479548
\$fp		30	0
\$ra		31	0

TH1 test = 1

```
#Laboratory Exercise 3, Home Assignment 3
.data
test: .word 1
.text
    la $s0, test #load the address of test variable
    lw $s1, 0($s0) #load the value of test to register $t1
    li $t0, 0 #load value for test case
    li $t1, 1
    li $t2, 2
    beq $s1, $t0, case_0
    beq $s1, $t1, case_1
    beq $s1, $t2, case_2
    j default
case_0:
    addi $s2, $s2, 1 #a=a+1
    j continue
case_1:
    sub $s2, $s2, $t1 #a=a-1
    j continue
case_2:
    add $s3, $s3, $s3 #b=2*b
    j continue
default:
continue:
```

- Test = 1 => \$s1 = 1 chạy vào case 1: a = a - 1 = 0 - 1 = -1

The screenshot shows a MIPS simulator interface with three main panels:

- Text Segment:** Displays the assembly code being executed. The instruction `beq $s1, $t1, case_1` is highlighted, indicating the current execution point.
- Data Segment:** Shows memory addresses and their corresponding values. The value at address `0x10010000` (labeled `test`) is `1`. The value at address `0x10010004` (labeled `$s2`) is `-1`.
- Registers:** Shows the state of registers. Register `$s2` is highlighted with a value of `-1`.

TH3: test = 2

```

week3_a3.asm*
1  #Laboratory Exercise 3, Home Assignment 3
2  .data
3  test: .word 2
4  .text
5      la $s0, test    #load the address of test variable
6      lw $s1, 0($s0)  #load the value of test to register $t1
7      li $t0, 0       #load value for test case
8      li $t1, 1
9      li $t2, 2
10     addi $s3, $zero, 3 # Khoi tao b = 3
11
12     beq $s1, $t0, case_0
13     beq $s1, $t1, case_1
14     beq $s1, $t2, case_2
15     j default
16     case_0:
17         addi $s2, $s2, 1 #a=a+1
18         j continue
19     case_1:
20         sub $s2, $s2, $t1 #a=a-1
21         j continue
22     case_2:
23         add $s3, $s3, $s3 #b=2*b
24         j continue
25     default:
26     continue:

```

- - Test = 2 => \$s1 = 2 chạy vào case 1: $b = 2 * b = 2 * 3 = 6$

The screenshot displays the Mars MIPS simulator interface. The main window is divided into several panes:

- Text Segment:** Shows the assembly code for week3_a3.asm. The code includes instructions for loading the test variable, setting up registers, and executing a series of conditional branches based on the value of \$s1. The code ends with a default case and a continue label.
- Data Segment:** A table showing memory addresses and their corresponding values. The values for registers \$s0 through \$s3 are shown, with \$s3 containing the value 6.
- Registers:** A table showing the current state of registers. The register \$s3 is highlighted, showing its value as 6.
- Labels:** A table showing the mapping of labels to addresses. The labels case_0, case_1, case_2, default, continue, and test are listed with their respective addresses.

The status bar at the bottom indicates that the program is running, with the instruction pointer (PC) at address 0x10010000.

Assignment 4:

a) $i < j$

```
week3_a4a.asm
1  #Laboratory Exercise 3, Home Assignment 1
2
3  .text
4      addi $s1, $zero 2 # i = 1
5      addi $s2, $zero 2 # j = 1
6
7      addi $t1, $zero 2 # x = 2
8      addi $t2, $zero 3 # y = 3
9      addi $t3, $zero 4 # z = 4
10 start:
11      bge $s1,$s2,else # i >= j
12      addi $t1,$t1,1 # then part: x=x+1
13      addi $t3,$zero,1 # z=1
14      j endif # skip "else" part
15 else:
16      addi $t2,$t2,-1 # begin else part: y=y-1
17      add $t3,$t3,$t3 # z=2*z
18 endif:
19
```

b) $i \geq j$

#Laboratory Exercise 3, Home Assignment 1

```
.text
    addi $s1, $zero 1 # i = 1
    addi $s2, $zero 5 # j = 2

    addi $t1, $zero 2 # x = 2
    addi $t2, $zero 3 # y = 3
    addi $t3, $zero 4 # z = 4

start:
    slt $t0, $s1, $s2 # i < j
    bne $t0, $zero, else # branch to else if j < i
    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:
```

c) $i+j \leq 0$

Edit	Execute
week3_a4a.asm	
1	<i>#Laboratory Exercise 3, Home Assignment 1</i>
2	
3	<i>.text</i>
4	addi \$s1, \$zero 1 <i># i = 1</i>
5	addi \$s2, \$zero 5 <i># j = 2</i>
6	
7	addi \$t1, \$zero 2 <i># x = 2</i>
8	addi \$t2, \$zero 3 <i># y = 3</i>
9	addi \$t3, \$zero 4 <i># z = 4</i>
10	start:
11	add \$t0, \$s1, \$s2 <i># \$t0 = i + j</i>
12	bgtz \$t0, else <i># \$t0 > 0 jump to else</i>
13	addi \$t1, \$t1, 1 <i># then part: x=x+1</i>
14	addi \$t3, \$zero, 1 <i># z=1</i>
15	j endif <i># skip "else" part</i>
16	else:
17	addi \$t2, \$t2, -1 <i># begin else part: y=y-1</i>
18	add \$t3, \$t3, \$t3 <i># z=2*z</i>
19	endif:
20	

d) $i+j > m+n$

#Laboratory Exercise 3, Home Assignment 1

.text

```
addi $s1, $zero 1 # i = 1
addi $s2, $zero 5 # j = 5
addi $s3, $zero 1 # m = 1
addi $s4, $zero -1 # n = -1
```

```
addi $t1, $zero 2 # x = 2
addi $t2, $zero 3 # y = 3
addi $t3, $zero 4 # z = 4
```

start:

```
add $t0, $s1, $s2 # $t0 = i + j
add $t4, $s3, $s4 # $t4 = m + n
ble $t0, $t4, else # $t0 > 0 jump to else
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:

Assignment 5:

a) $i < n$:

```
#Laboratory 3, Home Assignment 2
.data
A: .word 4,1,5,10,5
.text
    addi $s5, $zero, 0 # sum = 0
    addi $s1, $zero, 0 # i = 0

    la    $s2, A      # gan dia chi dau mang A cho $s2
    li    $s3, 5      # n = 5
    li    $s4, 1      # step = 1

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

b) $i \leq n$:

week3_a5a.asm

```
1  #Laboratory 3, Home Assigment 2
2  .data
3  A: .word 4,1,5,10,5,5
4  .text
5      addi $s5, $zero, 0 # sum = 0
6      addi $s1, $zero, 0 # i = 0
7
8      la  $s2, A      # gan dia chi dau mang A cho $s2
9      li  $s3, 5      # n = 5
10     li  $s4, 1      # step = 1
11
12 loop:
13     sgt $t2, $s1, $s3 # $t2 = i > n ? 1 : 0
14     bne $t2, $zero, endloop
15
16     add $t1, $s1, $s1 # $t1 = 2 * $s1
17     add $t1, $t1, $t1 # $t1 = 4 * $s1
18     add $t1, $t1, $s2 # $t1 store the address of A[i]
19     lw $t0, 0($t1) # load value of A[i] in $t0
20     add $s5, $s5, $t0 # sum = sum + A[i]
21     add $s1, $s1, $s4 # i = i + step
22     j loop # goto loop
23 endloop:
24
```

c) $\text{sum} \geq 0$

```

#Laboratory 3, Home Assignment 2
.data
A: .word 4,1,5,-15,5,5
.text

addi $s5, $zero, 0 # sum = 0
addi $s1, $zero, 0 # i = 0

la    $s2, A      # gan dia chi dau mang A cho $s2
li    $s3, 5      # n = 5
li    $s4, 1      # step = 1

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

    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

    bltz $s5, endloop # sum < 0 branch endloop
    j loop # goto loop
endloop:

```

d) $A[i] == 0$

#Laboratory 3, Home Assignment 2

.data

A: .word 0,0,0,3,5

.text

addi \$s5, \$zero, 0 # sum = 0

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

la \$s2, A # gan dia chi dau mang A cho \$s2

li \$s3, 5 # n = 5

li \$s4, 1 # step = 1

loop:

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

beq \$t2, \$zero, endloop

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 # A[i] == 0 branch endloop

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

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

j loop # goto loop

endloop:

Assignment 6:

```
A: .word 1,0,0,-25,5
.text
    addi $s5, $zero, 0 # max = 0
    addi $s1, $zero, 0 # i = 0
    la    $s2, A        # gan dia chi dau mang A cho $s2
    li    $s3, 5         # n = 5
    li    $s4, 1         # step = 1

loop:
    slt $t2, $s1, $s3 # $t2 = i < n ? 1 : 0
    beq $t2, $zero, endloop
    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
    bgtz $t0, if
    sub $t0, $zero, $t0

if:
    slt $t6, $s5, $t0 # $t6 = max < n? 1 : 0
    beq $t6, $zero, endif # $t6 = 0 branch endif
    add $s5, $t0, $zero # max = A[i]
endif:
    add $s1, $s1, $s4 # i = i + step
    j loop # goto loop
endloop:
```

