Báo cáo thực hành Kiến trúc máy tính

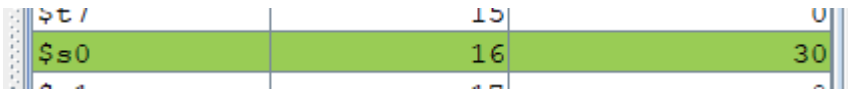
Assignment 1:

CODE:

```
#Laboratory Exercise 7 Home Assignment 1
.text
        main: li $a0,-30            #load input parameter
        jal abs                     #jump and link to abs procedure
        nop
        add $s0, $zero, $v0
        li $v0,10                   #terminate
        syscall
endmain:
#---------------------------------------------------------------
# function abs
# param[in] $a0 the interger need to be gained the absolute value
# return $v0 absolute value
#---------------------------------------------------------------
abs:
        sub $v0,$zero,$a0           #put -(a0) in v0; in case (a0)<0
        bltz $a0,done           #if (a0)<0 then done
        nop
        add $v0,$a0,$zero       #else put (a0) in v0
done:
        jr $ra
```

Kết quả:

Đầu vào $a0 = -30 đầu ra $s0 = 30



Assignment 2:

CODE:

```
#Laboratory Exercise 7, Home Assignment 2
.text
main:
        li $a0, 12              # load test input
        li $a1,-3
        li $a2,10
        jal max         # call max proceduce
        nop
        add $s0, $zero, $v0     #$s0 = max    Largest number a $s0
        li $v0,10
        syscall
endmain:
max:
        add $v0,$a0,$zero       # copy (a0) in v0; largest so far
        sub $t0,$a1,$v0                 # compute (a1)-(v0)
        bltz $t0,okay           # if (a1)-(v0)<0 then no change
        nop
        add $v0,$a1,$zero       # else max = a1
okay:
        sub $t0,$a2,$v0                 # compute (a2)-(v0)
        bltz $t0,done           # if (a2)-(v0)<0 then no change
        nop
        add $v0,$a2,$zero       # else max = a2
done:
        jr $ra                          # return to calling program
```

Kết quả

Đầu vào:

$a0	4	12
$a1	5	-3
$a2	6	10

Đầu ra:

$s0	16	12

Detail:

Nhảy tới chương trình con max lệnh jal max

V0 =a0 , t0 = a1 -v0 nếu t0 < 0 => a1< v0 và nhảy đến okay nếu không thì v0 = a1 chuyển sang okay:

T0 = a2 – v0 nếu t0 < 0 => a2 < v0 và nhảy đến done nếu không thì v0 = a2 chuyển sang done

Done: nhảy về chương trình chính bằng jr $ra

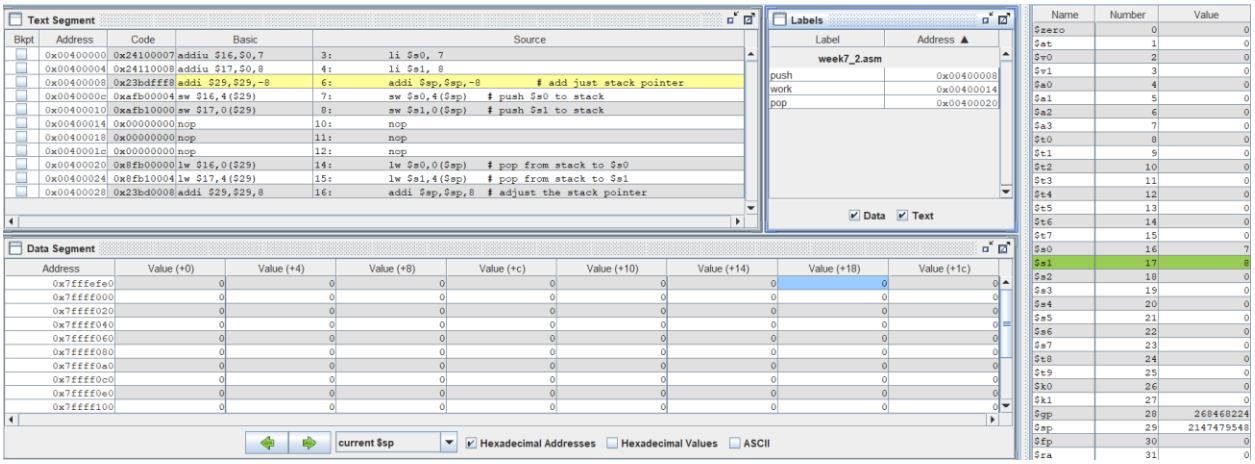
Assignment 3:

CODE:

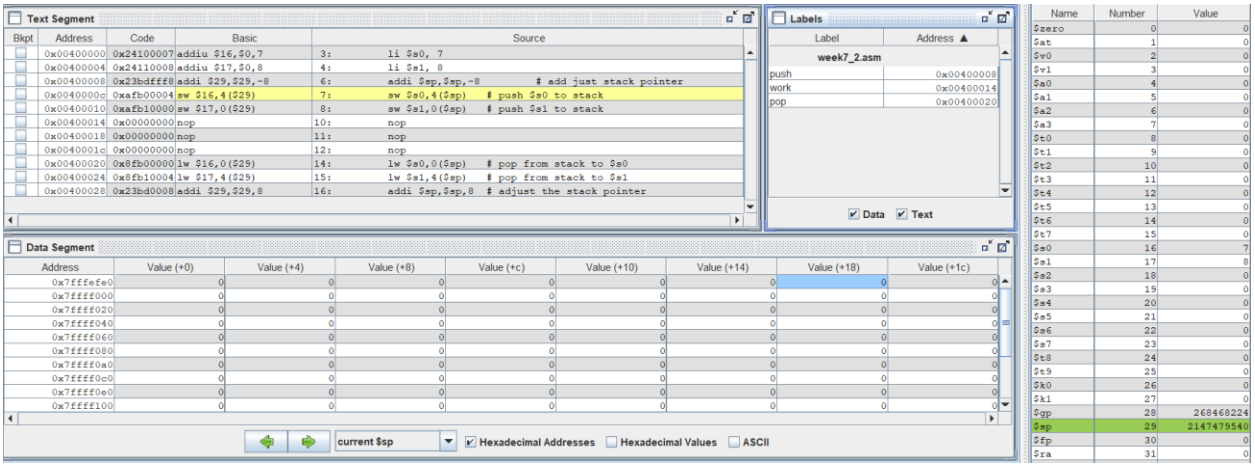
```
#Laboratory Exercise 7, Home Assignment 3
.text
        li $s0, 7
        li $s1, 8
push:
        addi $sp,$sp,-8         # add just stack pointer
        sw $s0,4($sp)   # push $s0 to stack
        sw $s1,0($sp)   # push $s1 to stack
work:
        nop
pop:
        lw $s0,0($sp)   # pop from stack to $s0
        lw $s1,4($sp)   # pop from stack to $s1
        addi $sp,$sp,8  # adjust the stack pointer
```

Kết quả:

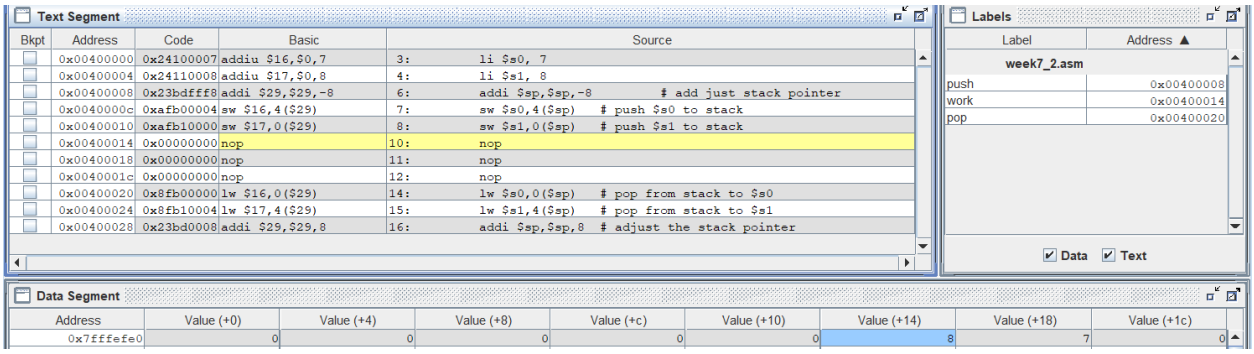
Khởi tạo giá trị $s1, $s0



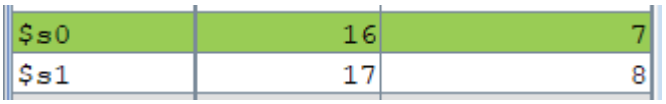
Khởi tạo stack bằng cách đẩy stack pointer xuống 8



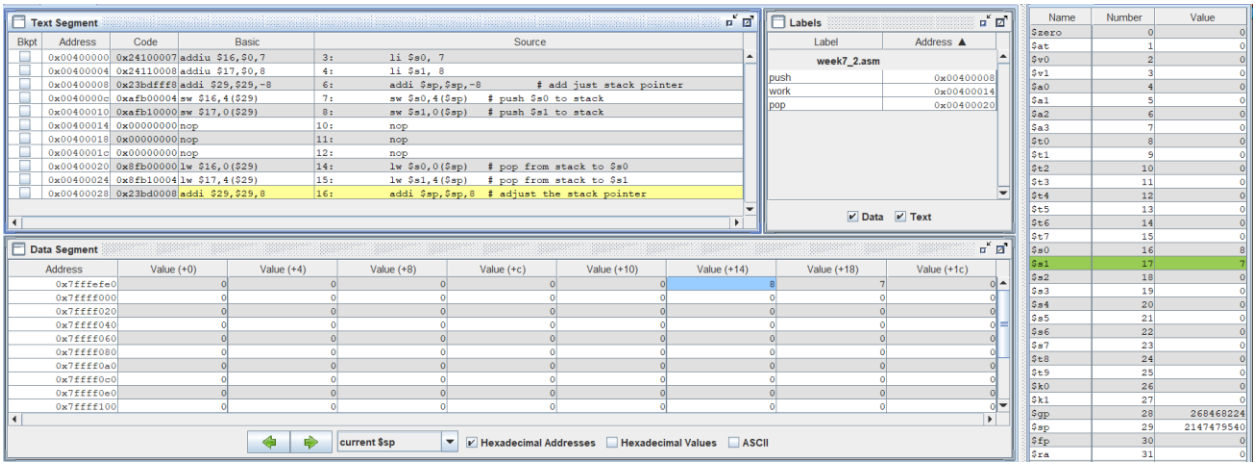
Các giá trị được load vào vùng nhớ stack



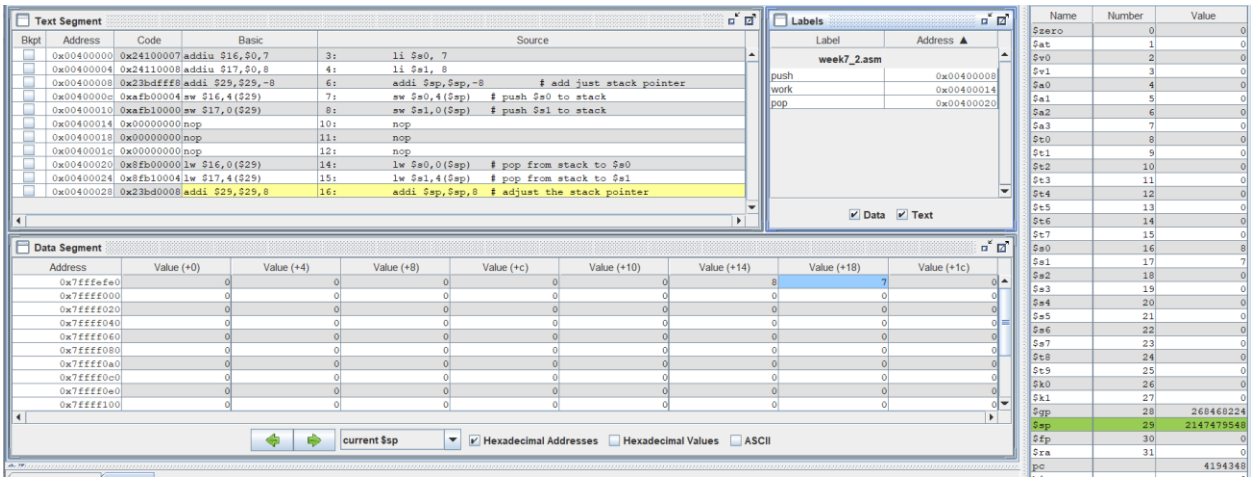
Giá trị 2 thanh ghi ban đầu



Giá trị 2 thanh ghi sau khi pop ra từ stack



Trả stack pointer về vị trí ban đầu giải phóng stack



Assignment 4:

a) Sử dụng $fp

CODE:

```
#Laboratory Exercise 7, Home Assignment 4
.data
	Message: .asciiz "Ket qua tinh giai thua la: "
.text
main:
	jal WARP
print:
	add $a1, $v0, $zero # $a0 = result from N!
	li $v0, 56
	la $a0, Message
	syscall
quit:
	li $v0, 10 #terminate
	syscall
endmain:
#---------------------------------------------------------------------
#Procedure WARP: assign value and call FACT
#---------------------------------------------------------------------
WARP:
	sw $fp,-4($sp) #save frame pointer (1)
	addi $fp,$sp,0 #new frame pointer point to the top (2)
	addi $sp,$sp,-8 #adjust stack pointer (3)
	sw $ra,0($sp) #save return address (4)
	li $a0,3			#load test input N
	jal FACT #call fact procedure
	nop
	lw $ra,0($sp) #restore return address (5)
	addi $sp,$fp,0 #return stack pointer (6)
```

```
        lw $fp,-4($sp) #return frame pointer (7)
        jr $ra
wrap_end:
#-----------------------------------------------------------------------
#Procedure FACT: compute N!
#param[in] $a0 integer N
#return $v0 the largest value
#-----------------------------------------------------------------------
FACT:
        sw $fp,-4($sp) #save frame pointer
        addi $fp,$sp,0 #new frame pointer point to stack's top
        addi $sp,$sp,-12 #allocate space for $fp,$ra,$a0 in stack
        sw $ra,4($sp) #save return address
        sw $a0,0($sp) #save $a0 register
        slti $t0,$a0,2 #if input argument N < 2
        beq $t0,$zero,recursive#if it is false ((a0 = N) >=2)
        nop
        li $v0,1 #return the result N!=1
        j done
        nop
recursive:
        addi $a0,$a0,-1 #adjust input argument
        jal FACT #recursive call
        nop
        lw $v1,0($sp) #load a0
        mult $v1,$v0 #compute the result
        mflo $v0
done:
        lw $ra,4($sp) #restore return address
        lw $a0,0($sp) #restore a0
```

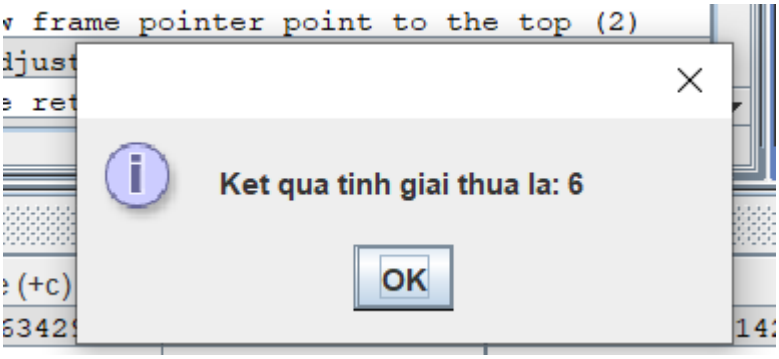
addi $sp,$fp,0 #restore stack pointer

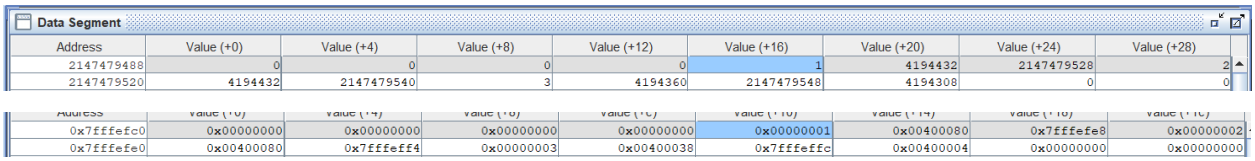
lw $fp,-4($sp) #restore frame pointer

jr $ra #jump to calling

fact_end:

Kết quả: Đầu vào a0 =3; đầu ra a1 =6





=>Stack

$a0	0x00000001
$ra	0x00400080
$fp	0x7fffefe8
$a0	0x00000002
$ra	0x00400080
$fp	0x7fffeff4
$a0	0x00000003
$ra	0x00400038
$fp	0x7fffeffc
$ra	0x00400004

$a0, $fp, $ra lần gọi vòng lặp thứ ba

0x00000001	0x00400080	0x7fffefe8

$a0, $fp, $ra lần gọi gọi vòng lặp thứ hai

0x00000002	0x7fffeff4	0x00000003

$a0, $fp, $ra lần gọi gọi vòng lặp thứ nhất

0x00000003	0x00400038	0x7fffeffc

b) Không sử dụng $fp

```
#param[in] $a0 integer N
#return $v0 the largest value
#---------------------------------------------------------------------
FACT:
        sw $fp,-4($sp) #save frame pointer
        addi $fp,$sp,0 #new frame pointer point to stack's top
        addi $sp,$sp,-12 #allocate space for $fp,$ra,$a0 in stack
        sw $ra,4($sp) #save return address
        sw $a0,0($sp) #save $a0 register

        slti $t0,$a0,2 #if input argument N < 2
        beq $t0,$zero,recursive#if it is false ((a0 = N) >=2)
        nop
        li $v0,1 #return the result N!=1
        j done
        nop
recursive:
        addi $a0,$a0,-1 #adjust input argument
        jal FACT #recursive call
        nop
        lw $v1,0($sp) #load a0
        mult $v1,$v0 #compute the result
        mflo $v0
done:
        lw $ra,4($sp) #restore return address
```

lw $a0,0($sp) #restore a0

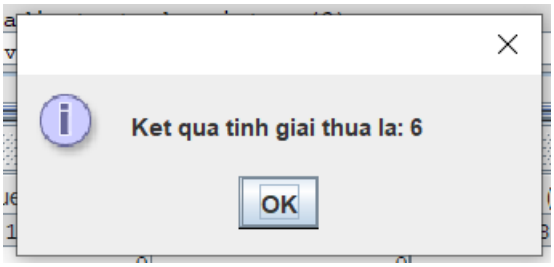
addi $sp,$fp,0 #restore stack pointer

lw $fp,-4($sp) #restore frame pointer

jr $ra #jump to calling

fact_end:

Kết quả:



Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)
2147479488	0	0	0	0	0	0	0	1
2147479520	4194416	2	4194416	3	4194352	4194308	0	0

Điểm khác không sử dụng $fp thì ta trừ lại kích cỡ của stack ở mỗi vòng lặp để lấy giá trị và quay lại vòng lặp ban đầu

Assignment 5:

#Laboratory Exercise 7, Assignment 5

.data

Mess1: .asciiz "Largest: "

Mess2: .asciiz "¥nSmallest:"

Comma: .asciiz ","

.text

main: li $s0, 2 # Load input

li $s1, 3

li $s2, -1

li $s3, 4

li $s4, 9

li $s5, -2

li $s6, 8

```
li $s7, 5
jal init
nop
li $v0, 4
la $a0, Mess1           #print mess1
syscall
li $v0 , 1
add $a0,$t0,$zero
syscall                 #print max value
li $v0, 4
la $a0, Comma           #print ","
syscall
li $v0 , 1
add $a0,$t5,$zero       # print max value's position
syscall
li $v0, 4
la $a0, Mess2           #print mess2
syscall
li $v0 ,1
add $a0,$t1,$zero
syscall                 #print min value
li $v0 ,4
la $a0, Comma                   #print ","
syscall
li $v0 , 1
add $a0,$t6,$zero
syscall                 # print min value's position
li $v0, 10
syscall                 # exit
```

```
endmain:
swapMax:add $t0,$t3,$zero     # set Max = $t3
        add $t5,$t2,$zero     # set i of max = $t2
        jr $ra
swapMin:add $t1,$t3,$zero     # set Min = $t3
        add $t6,$t2,$zero     # set i of min = $t2
        jr $ra
init:   add $fp,$sp,$zero     # save address of origin sp
        addi $sp,$sp, -32     # create space for stack
        sw $s1, 0($sp)
        sw $s2, 4($sp)
        sw $s3, 8($sp)
        sw $s4, 12($sp)
        sw $s5, 16($sp)
        sw $s6, 20($sp)
        sw $s7, 24($sp)
        sw $ra, 28($sp)       # save $ra for main
        add $t0,$s0,$zero     # set Max = $s0
        add $t1,$s0,$zero     # set Min = $s0
        li $t5, 0         # set $t5 to 0
        li $t6, 0         # set $t6 to 0
        li $t2, 0         # set $t2 to 0 , i = 0
max_min:addi $sp,$sp,4
        lw $t3,-4($sp)
        sub $t4, $sp, $fp     # check if meet $ra
        beq $t4,$zero, done   # if true then done
        addi $t2,$t2,1        # i++
        sub $t4,$t0,$t3       # Max - $t3
        bltzal $t4, swapMax   # if Max - $t3 < 0, swap Max
```

```
        sub $t4,$t3,$t1         # $t3 - Min
        bltzal $t4, swapMin     # if $t3 – Min < 0 , swap Min
        j max_min               # repeat
done:   lw $ra, -4($sp)         # load #$ra
        jr $ra                  # return
```

Kết quả:

Init : Khởi tạo giá trị cho mảng và lưu vào stack

In ra giá trị lớn nhất và vị trí : 9 tại $s4

In ra giá trị bé nhất và vị trí : -2 tại $s5