**­­­­Computer Architecture Lab Report Week 7**

**Full name: Nguyễn Hồng Phúc**

**Student ID: 20225659**

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:**

-Trước khi chạy lệnh jal:

A screenshot of a computer

Description automatically generated

-Sau khi chạy xong:

A screenshot of a computer

Description automatically generated

Nhận xét:  
- Khi chạy lệnh jal abs (địa chỉ lệnh 0x00400004) thì thanh ghi $ra  
được gán bằng địa chỉ của câu lệnh tiếp theo (0x00400008) và thanh ghi  
pc được gán bằng địa chỉ 0x00400018 (địa chỉ tại nhãn abs)  
- Kết quả cuối cùng ta lấy được giá trị tuyệt đối của số được nạp vào trong  
thanh ghi $a0 và ghi kết quả đó vào thanh ghi $s0

Assignment 2

Code:

#Laboratory Exercise 7, Home Assignment 2

.text

main: li $a0,2 #load test input

li $a1,-6

li $a2,-9

jal max #call max procedure

nop

li $v0, 10 #terminate

syscall

endmain:

#---------------------------------------------------------------------

#Procedure max: find the largest of three integers

#param[in] $a0 integers

#param[in] $a1 integers

#param[in] $a2 integers

#return $v0 the largest value

#---------------------------------------------------------------------

max:

add $v1,$a0,$zero #copy (a0) in v0; largest so far

sub $t0,$a1,$v1 #compute (a1)-(v0)

bltz $t0,okay #if (a1)-(v0)<0 then no change

nop

add $v1,$a1,$zero #else (a1) is largest thus far

okay: sub $t0,$a2,$v1 #compute (a2)-(v0)

bltz $t0,done #if (a2)-(v0)<0 then no change

nop

add $v1,$a2,$zero #else (a2) is largest overall

done: jr $ra #return to calling program

Kết quả chạy chương trình:

A screenshot of a computer

Description automatically generated

Số lớn nhất là 2 được lưu vào $v1

**Nhận xét:**  
- Khi chạy lệnh jal thì thanh ghi $ra được gán bằng giá trị của địa chỉ  
của câu lệnh tiếp theo sau jal trong nhãn main. Thanh ghi pc được gán  
bằng địa chỉ của nhãn max để câu lệnh tiếp tục được thực hiện bắt đầu  
từ nhãn max. Sau khi chạy đến jr $ra thì pc được gán bằng địa chỉ  
trong $ra (địa chỉ của nop)

Assignment 3

Code:

#Laboratory Exercise 7, Home Assignment 3

.text

li $s1,7

li $s0,9

push: addi $sp,$sp,-8 #adjust the stack pointer

sw $s0,4($sp) #push $s0 to stack

sw $s1,0($sp) #push $s1 to stack

work: nop

nop

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:**

- Trước khi chạy lệnh addi ở nhãn push:

A screenshot of a computer

Description automatically generated

$sp = 0x7fffeffc

- Sau khi chạy lệnh addi ở nhãn push:

$sp = 0x7fffeff4

A screenshot of a computer

Description automatically generated

+ Thanh ghi $sp được giảm đi 8 byte (tức là có sự cấp phát cho bộ nhớ  
stack 8 byte)  
+ Sau đó lần lượt ghi giá trị trong $s0 vào $sp + 4, giá trị trong $s1 vào  
$sp + 0  
- Sau khi chạy lệnh addi ở nhãn pop:

A screenshot of a computer

Description automatically generated

+ Thực hiện đổi chỗ hai số bằng cách load giá trị tại địa chỉ $sp + 0 vào  
$s1, load giá trị tại địa chỉ $sp + 4 vào $s1  
+ Lệnh add $sp, $sp, 8 (giúp giải phóng stack, trả lại đỉnh stack)

Assignment 4

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

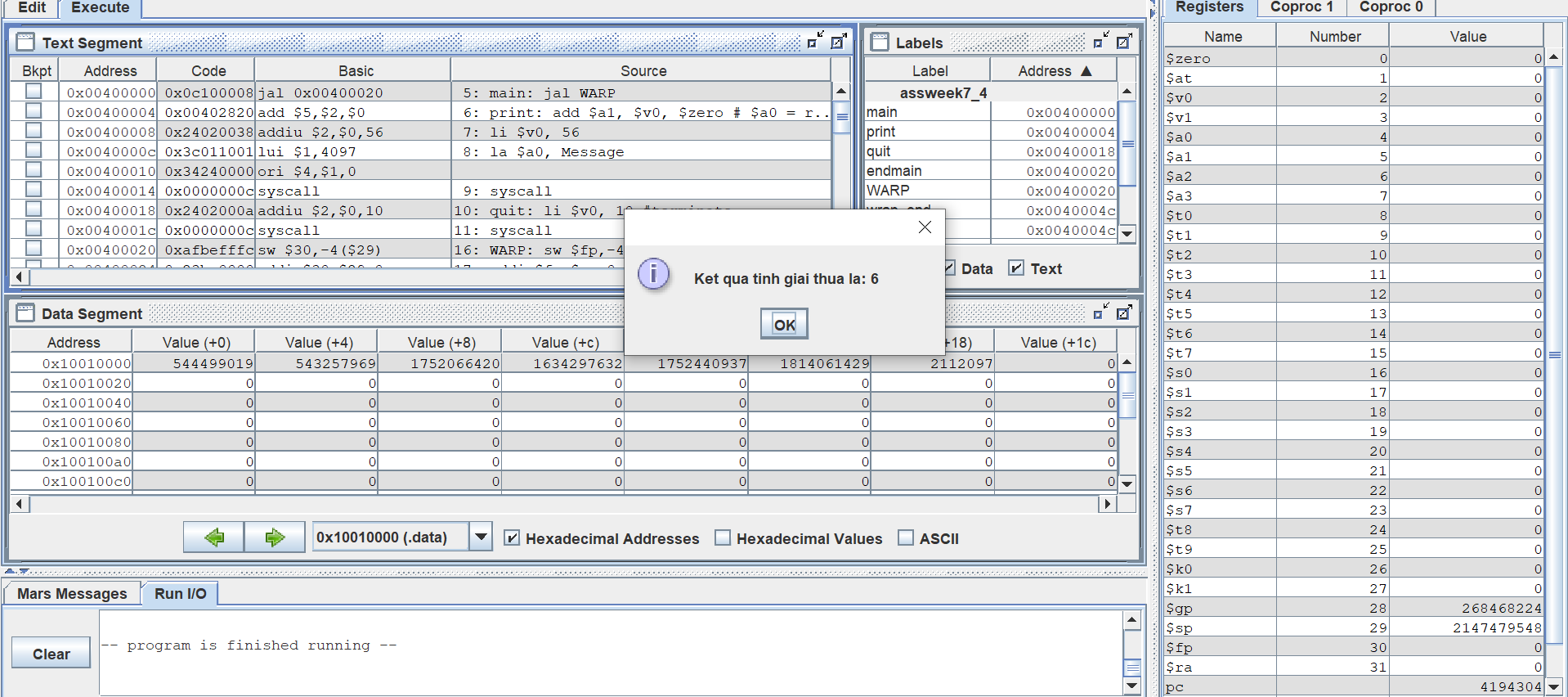
lw $fp,-4($sp) #restore frame pointer

jr $ra #jump to calling

fact\_end:

Kết quả:

Với n=3 => kết quả được 6



|  |  |
| --- | --- |
| 0x7fffeff8 | $fp = 0x00000000 |
| 0x7fffeff4 | $ra = 0x00400004 |
| 0x7fffeff0 | $fp = 0x7fffeffc |
| 0x7fffefec | $ra = 0x00400038 |
| 0x7fffefe8 | $a0 = 0x00000003 |
| 0x7fffefe4 | $fp = 0x7fffeff4 |
| 0x7fffefe0 | $ra = 0x00400080 |
| 0x7fffefdc | $a0 = 0x00000002 |
| 0x7fffefd8 | $fp = 0x7fffefe4 |
| 0x7fffefd4 | $ra = 0x00400080 |
| 0x7fffefd0 | $a0 = 0x00000001 |

Assignment 5

Code:

.data

# Initialize the list of 8 elements

elements: .word 5, 69, -33, 6, 12, 8, 45, 7

.text

main:

# Initialize variables

li $t0, -2147483648 # Largest value (minimum possible value)

li $t1, 2147483647 # Smallest value (maximum possible value)

li $t2, 0 # Index of largest value

li $t3, 0 # Index of smallest value

# Loop through the list of elements

li $t4, 0 # Initialize loop counter

la $a3,elements # Pointer to the head of the list

loop:

beq $t4, 8, done # Exit loop when all elements processed

# Load the current element from the list

lw $t5, 0($a3)

# Compare with largest value

bgt $t5, $t0, update\_largest

j compare\_smallest

update\_largest:

move $t0, $t5 # Update largest value

move $t2, $t4 # Update index of largest value

j compare\_smallest

compare\_smallest:

blt $t5, $t1, update\_smallest

addi $t4, $t4, 1

addi $a3, $a3, +4

j loop

update\_smallest:

move $t1, $t5 # Update smallest value

move $t3, $t4 # Update index of smallest value

# Increment loop counter

addi $t4, $t4, 1

# Update pointer

addi $a3, $a3, +4

j loop

done:

li $v0, 4

la $a0, largest\_str

syscall

li $v0, 1

move $a0, $t0 # Largest value

syscall

li $v0, 4

la $a0, comma

syscall

li $v0, 1

move $a0, $t2 # Index of largest value

syscall

li $v0, 4

la $a0, newline

syscall

li $v0, 4

la $a0, smallest\_str

syscall

li $v0, 1

move $a0, $t1 # Smallest value

syscall

li $v0, 4

la $a0, comma

syscall

li $v0, 1

move $a0, $t3 # Index of smallest value

syscall

# Exit program

li $v0, 10

syscall

.data

newline: .asciiz "\n"

largest\_str: .asciiz "Largest: "

smallest\_str: .asciiz "Smallest: "

comma: .asciiz ", "

Kết quả:

