BÁO CÁO THỰC HÀNH TUẦN 7

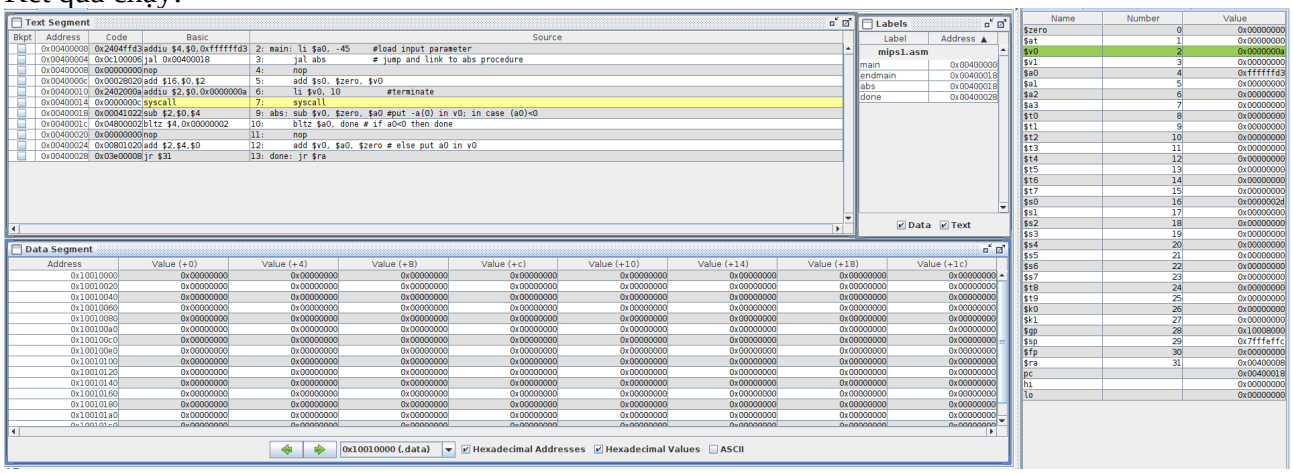
Lương Thị Tâm – 20194663

Bài 1:

Code:

```
.text
main: li $a0, -45     #load input parameter
      jal abs         # jump and link to abs procedure
      nop
      add $s0, $zero, $v0
      li $v0, 10          #terminate
      syscall
endmain:
abs: sub $v0, $zero, $a0 #put -a(0) 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 chạy:



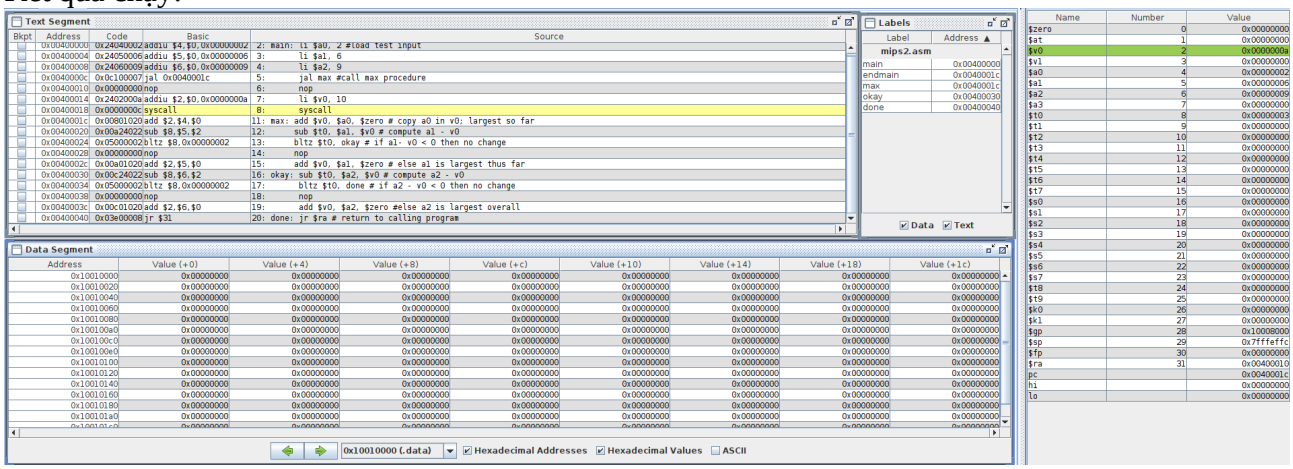
Bài 2:

Code:

```
.text
main: li $a0, 2 #load test input
      li $a1, 6
      li $a2, 9
      jal max #call max procedure
      nop
      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 a1 is largest thus far
okay: sub $t0, $a2, $v0 # compute a2 - v0
      bltz $t0, done # if a2 - v0 < 0 then no change
      nop
      add $v0, $a2, $zero #else a2 is largest overall
done: jr $ra # return to calling program
```

Kết quả chạy:

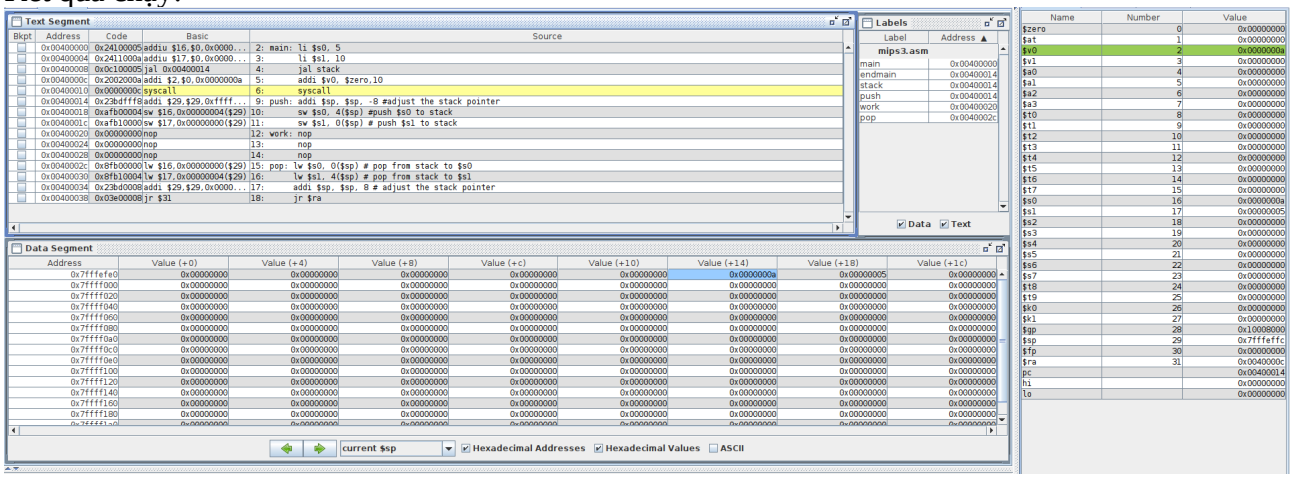


Bài 3:

Code:

```
mips1.asm  mips2.asm  mips3.asm
1  .text
2  main: li $s0, 5
3        li $s1, 10
4        jal stack
5        addi $v0, $zero,10
6        syscall
7  endmain:
8  stack:
9  push: addi $sp, $sp, -8 #adjust the stack pointer
10       sw $s0, 4($sp) #push $s0 to stack
11       sw $s1, 0($sp) # push $s1 to stack
12 work: nop
13       nop
14       nop
15 pop: lw $s0, 0($sp) # pop from stack to $s0
16       lw $s1, 4($sp) # pop from stack to $s1
17      addi $sp, $sp, 8 # adjust the stack pointer
18       jr $ra
```

Kết quả chạy:



Bài 4:
a, Khi dùng $sp:
Code:

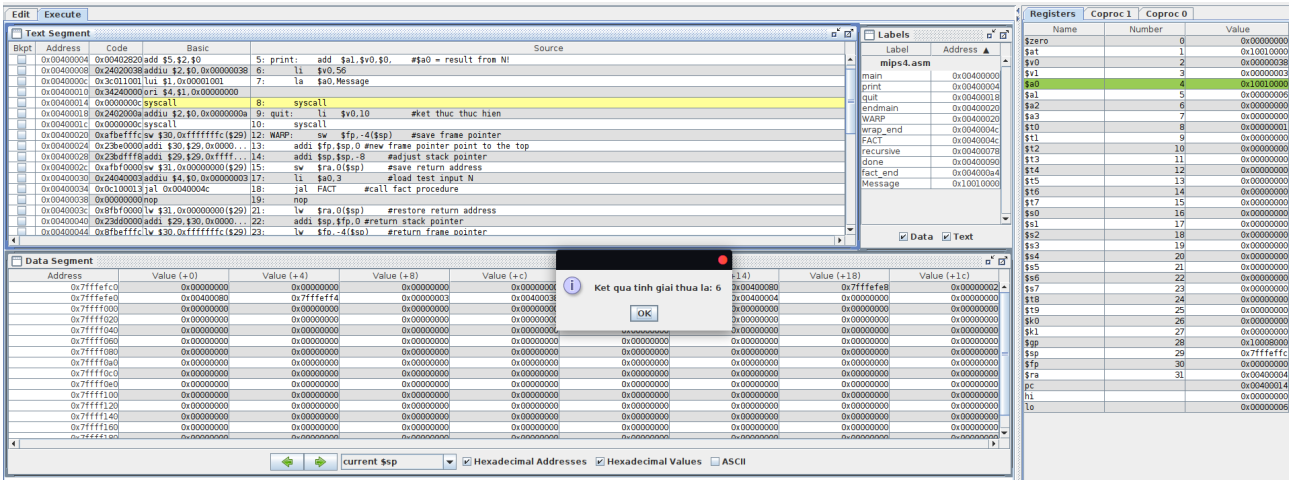
```
.data
Message:  .asciiz   "Ket qua tinh giai thua la: "
.text
main:     jal  WARP
print:    add  $a1,$v0,$0,    #$a0 = result from N!
     li   $v0,56
     la   $a0,Message
     syscall
quit:     li   $v0,10         #ket thuc thuc hien
     syscall
endmain:
WARP:     sw   $fp,-4($sp)    #save frame pointer
     addi $fp,$sp,0 #new frame pointer point to the top
     addi $sp,$sp,-8     #adjust stack pointer
     sw   $ra,0($sp)     #save return address

     li   $a0,3          #load test input N
     jal  FACT      #call fact procedure
     nop

     lw   $ra,0($sp)     #restore return address
     addi $sp,$fp,0 #return stack pointer
     lw   $fp,-4($sp)    #return frame pointer
     jr   $ra
wrap_end:
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 chạy:



b, Khi không dùng $fp:

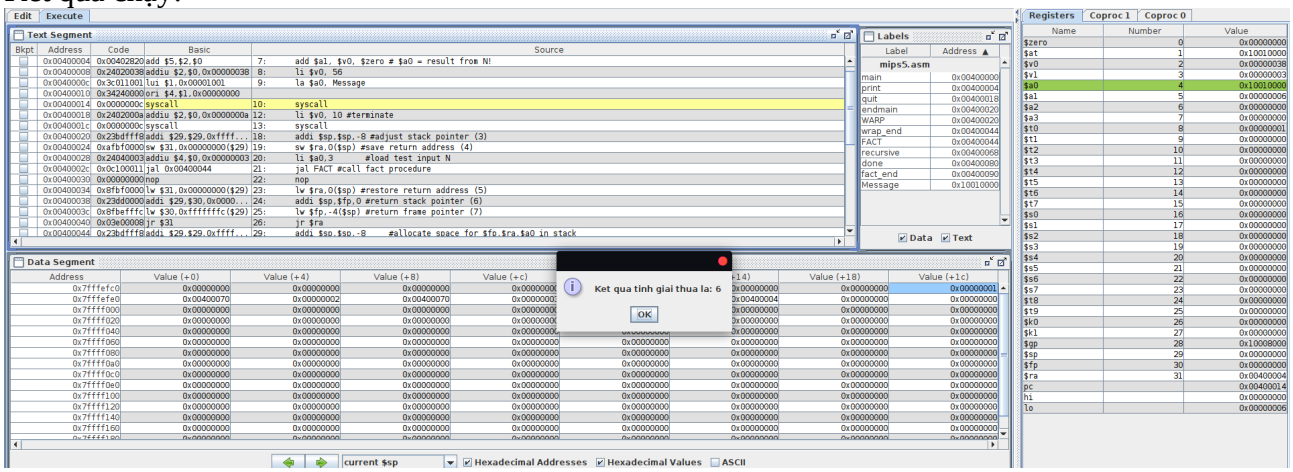
Code:

```
.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:
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:
FACT:
     addi $sp,$sp,-8     #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 $sp,$sp,8           #restore frame pointer
     jr $ra              #jump to calling
fact_end:
```

Kết quả chạy:



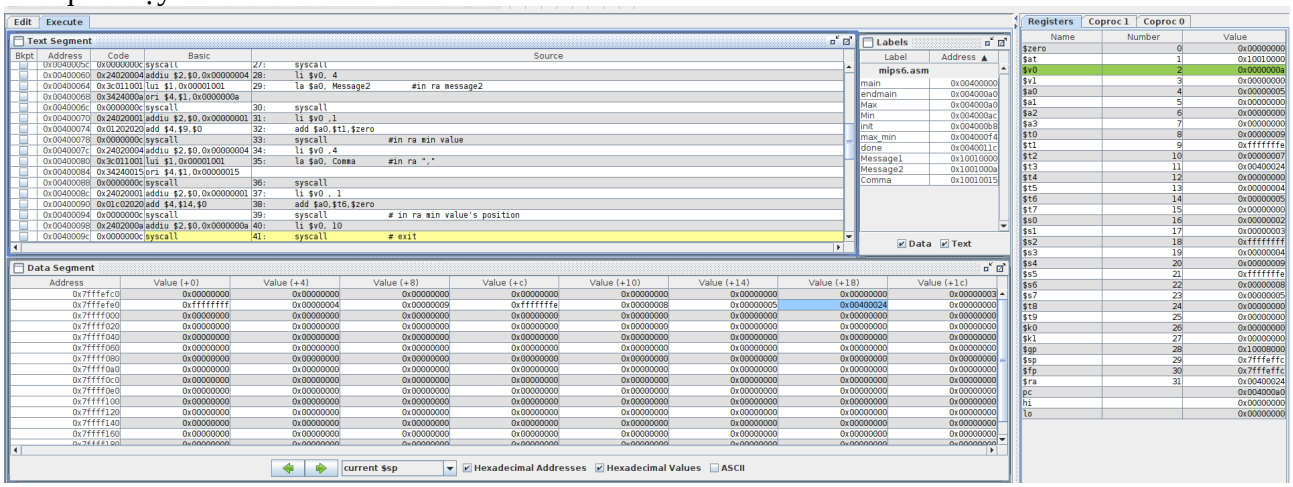
Bài 5:
Code:

```
.data
Message1:       .asciiz "Largest: "
Message2:       .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, Message1         #in ra message1
     syscall
     li $v0 , 1
     add $a0,$t0,$zero
     syscall                  #in ra max value
     li $v0, 4
     la $a0, Comma            #in ra ","
     syscall
     li $v0 , 1
     add $a0,$t5,$zero        # in ra max value's position
     syscall
     li $v0, 4
     la $a0, Message2         #in ra message2
     syscall
     li $v0 ,1
     add $a0,$t1,$zero
     syscall                  #in ra min value
     li $v0 ,4
     la $a0, Comma            #in ra ","
     syscall
     li $v0 , 1
     add $a0,$t6,$zero
     syscall                  # in ra min value's position
     li $v0, 10
     syscall                  # exit
endmain:
```

```
endmain:
Max:add $t0,$t3,$zero    # set Max = $t3
     add $t5,$t2,$zero    # set i of max = $t2
     jr $ra
Min: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, Max      # if Max - $t3 < 0, swap Max
     sub $t4,$t3,$t1          # $t3 - Min
     bltzal $t4, Min      # if $t3 – Min < 0 , swap Min
     j max_min      # repeat
done:     lw $ra, -4($sp)          # load #$ra
     jr $ra               # return
```

Line: 38 Column: 19 ☑ Show Line Numbers

Kết quả chạy:


Edit
Execute
Text Segment
Bkpt
Address
Code
Basic
Source
0x00400060 0x24020004 addiu $2,$0,0x00000004 28: li $v0, 4
0x00400064 0x3c011001 lui $1,0x00001001 29: la $a0, Message2 #in ra message2
0x00400068 0x3424003a ori $4,$1,0x0000003a
0x0040006c 0x0000000c syscall 30: syscall
0x00400070 0x24020001 addiu $2,$0,0x00000001 31: li $v0 ,1
0x00400074 0x01202020 add $4,$9,$0 32: add $a0,$t1,$zero
0x00400078 0x0000000c syscall 33: syscall #in ra min value
0x0040007c 0x24020004 addiu $2,$0,0x00000004 34: li $v0 ,4
0x00400080 0x3c011001 lui $1,0x00001001 35: la $a0, Comma #in ra ","
0x00400084 0x34240015 ori $4,$1,0x00000015
0x00400088 0x0000000c syscall 36: syscall
0x0040008c 0x24020001 addiu $2,$0,0x00000001 37: li $v0 , 1
0x00400090 0x01c02020 add $4,$14,$0 38: add $a0,$t6,$zero
0x00400094 0x0000000c syscall 39: syscall # in ra min value's position
0x00400098 0x2402000a addiu $2,$0,0x0000000a 40: li $v0, 10
0x0040009c 0x0000000c syscall 41: syscall # exit
Labels
Label
Address
mips6.asm
main 0x00400000
endmain 0x004000a0
Max 0x004000a0
Min 0x004000ac
init 0x004000b8
max_min 0x004000f4
done 0x0040011c
Message1 0x10010000
Message2 0x1001000a
Comma 0x10010015
Data
Text
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x7fffefc0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000003
0x7fffefe0 0xffffffff 0x00000004 0x00000009 0xfffffffe 0x00000008 0x00000005 0x00400024 0x00000000
0x7ffff000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff020 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff040 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff060 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff080 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff0a0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff0c0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff0e0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff100 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff120 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff140 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x7ffff160 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
current $sp
Hexadecimal Addresses
Hexadecimal Values
ASCII
Registers
Coproc 1
Coproc 0
Name
Number
Value
$zero 0 0x00000000
$at 1 0x10010000
$v0 2 0x00000004
$v1 3 0x00000000
$a0 4 0x00000005
$a1 5 0x00000000
$a2 6 0x00000000
$a3 7 0x00000000
$t0 8 0x00000009
$t1 9 0xfffffffe
$t2 10 0x00000007
$t3 11 0x00400024
$t4 12 0x00000000
$t5 13 0x00000004
$t6 14 0x00000005
$t7 15 0x00000000
$s0 16 0x00000002
$s1 17 0x00000003
$s2 18 0xffffffff
$s3 19 0x00000004
$s4 20 0x00000009
$s5 21 0xfffffffe
$s6 22 0x00000008
$s7 23 0x00000005
$t8 24 0x00000000
$t9 25 0x00000000
$k0 26 0x00000000
$k1 27 0x00000000
$gp 28 0x10008000
$sp 29 0x7fffeffc
$fp 30 0x7fffeffc
$ra 31 0x00400024
pc 0x004000a0
hi 0x00000000
lo 0x00000000