

THỰC HÀNH KIẾN TRÚC MÁY TÍNH

IT-3280

Họ tên	Thân Cát Ngọc Lan
MSSV	20225646

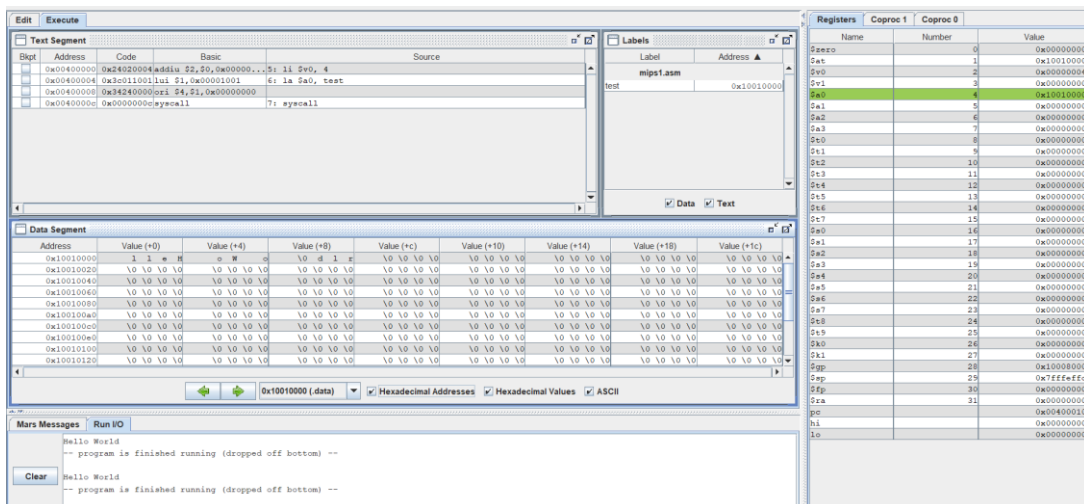
Assignment 1:

B1: Thanh ghi \$v0 thay đổi giá trị từ 0x00000000 -> 0x00000004

B2: Thanh ghi \$a0 thay đổi giá trị từ 0x00000000 -> 0x10010000

B3: Thanh ghi \$a0 thay đổi giá trị từ 0x00000000 -> 0x10010000

- Giải thích: khi gọi lệnh syscall, HĐH sẽ xem giá trị của \$v0
Vì giá trị thanh ghi \$v0 = 4 => thanh ghi \$a0 sẽ là 1 string và in ra màn hình.

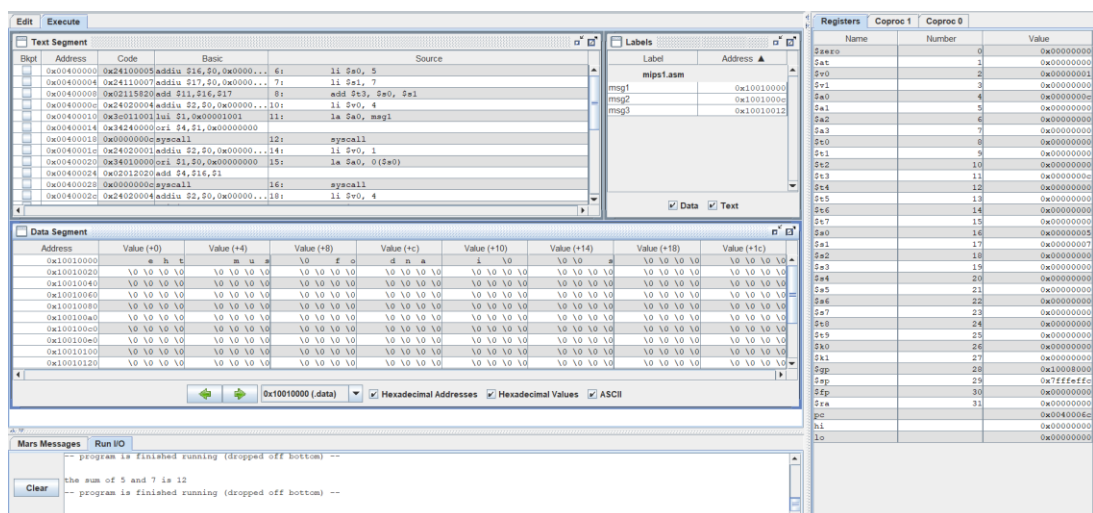


Assignment 2:

```

1  .data
2  msg1: .asciiz "the sum of "
3  msg2: .asciiz " and "
4  msg3: .asciiz " is "
5  .text
6      li $s0, 5
7      li $s1, 7
8      add $t3, $s0, $s1
9      #syscall nhap "the sum of"
10     li $v0, 4
11     la $a0, msg1
12     syscall
13     #syscall nhap $s0
14     li $v0, 1
15     la $a0, 0($s0)
16     syscall
17     #syscall nhap "and"
18     li $v0, 4
19     la $a0, msg2
20     syscall
21     #syscall nhap $s1
22     li $v0, 1
23     la $a0, 0($s1)
24     syscall
25     #syscall nhap "is"
26     li $v0, 4
27     la $a0, msg3
28     syscall
29     #syscall nhap tong cua (s0) vs (s1)
30     li $v0, 1
31     la $a0, 0($t3)
32     syscall

```



Giải thích: Mỗi khi lệnh syscall, HĐH sẽ xem giá trị của \$v0.

- Khi \$v0 = 4 => \$a0 là 1 xâu
- Khi \$v0 = 1 => \$a0 là 1 số nguyên

Sau đó in ra màn hình.

⇒ Nhận xét: Kết quả chạy đúng và in ra đúng định dạng: **“the sum of ... is ...”**

Assignment 3:

The screenshot shows the MARS MIPS simulator interface. The main window displays assembly code for a program that prints "Hello Cat Lan". The code includes comments in Vietnamese explaining the instructions. The registers window on the right shows the values of the registers, with \$s0 and \$s1 being the most relevant for this program.

Register	Value
\$s0	0x00000000
\$s1	0x00000000

Giải thích:

- Lệnh la (dòng 8,9) -> nạp địa chỉ của xâu x,y
 - Lệnh add \$t1,\$s0,\$a1 -> trở vào y[i]
 - Lệnh lb là lấy giá trị của y[i]
 - Lệnh add \$t3,\$s0,\$a0 -> trở vào x[i]
 - Lệnh sb lấy giá trị đã lấy từ y[i] nạp vào x[i]
- ➔ Thực hiện vòng lặp để duyệt xâu

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

- Text Segment:** Shows assembly code with addresses, instructions, and their sources. Instructions include `add $s0,$zero,$zero #a0 = i=0`, `la $a0, x # load $a0 = address of x[0]`, `la $a1, y # load $a1 = address of y[0]`, `add $t1,$a0,$a1 #t1 = a0 + a1 = i + y[0]`, `lb $t2,0($t1) #t2 = value at t1 = y[i]`, `add $t3,$a0,$a0 #t3 = a0 + a0 = i + x[0]`, `sb $t2,0($t3) #x[i] = t2 = y[i]`, `beq $t2,$zero,end_of_strcpy #if y[i]==0, exit`, and `addi $a0,$a0,1 #a0=a0 + 1 <-> i=i+1`.
- Data Segment:** A table showing memory addresses and their corresponding values in hexadecimal, decimal, and ASCII.
- Registers:** A table listing registers (e.g., \$zero, \$at, \$v0, \$a0, \$a1, \$a2, \$t0, \$t1, \$t2, \$t3, \$t4, \$t5, \$t6, \$t7, \$a0, \$a1, \$a2, \$a3, \$a4, \$a5, \$a6, \$a7, \$t8, \$t9, \$k0, \$k1, \$gp, \$fp, \$sp, \$ra, \$pc, \$hi, \$lo) and their current values.

At the bottom, there is a "Mars Messages" panel showing the output of the program: "Hello Cat Lan".

⇒ Kết quả chạy đúng với lí thuyết

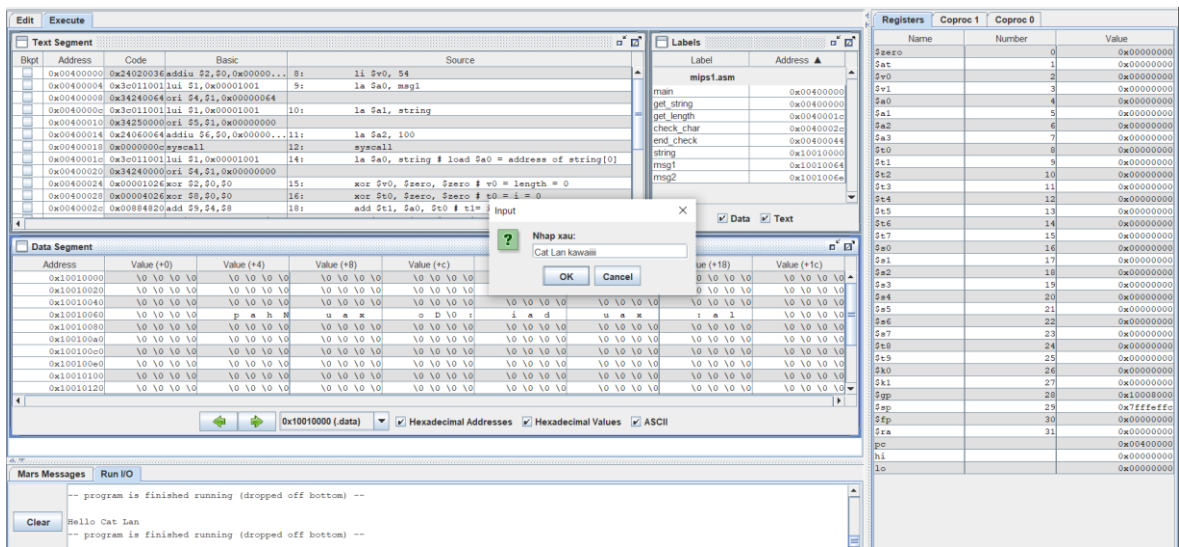
Assignment 4:

The screenshot shows a MIPS simulator window with the following assembly code:

```

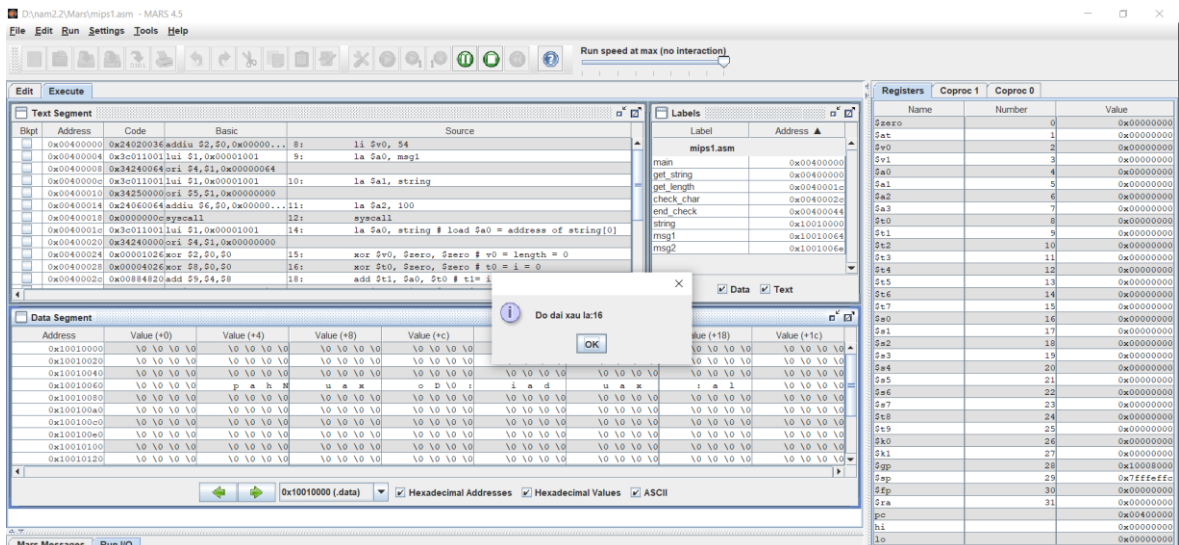
1 .data
2 string: .space 100
3 msg1: .asciiz "Nhap xau:"
4 msg2: .asciiz "Do dai xau la:"
5 .text
6 main:
7 get_string: #nhap xau vao string
8     li $v0, 54
9     la $a0, msg1
10    la $a1, string
11    la $a2, 100
12    syscall
13 get_length:
14    la $a0, string # load $a0 = address of string[0]
15    xor $v0, $zero, $zero # v0 = length = 0
16    xor $t0, $zero, $zero # t0 = i = 0
17 check_char:
18    add $t1, $a0, $t0 # t1 = i + string[0]; t1 tro vaof dia chi cua string[i]
19    lb $t2, 0($t1) #nap gia tri string[i] vao t2 ; t2 = string[i]
20    beq $t2, $zero, end_check # if t2=null-> ket thuc
21    addi $v0, $v0, 1 # v0=v0+1
22    addi $t0, $t0, 1 #i=i+1
23    j check_char # vong lap moi
24 end_check:
25 #print length
26    addi $a1, $v0, -1 #do thua dau \n nen phai -1
27    li $v0, 56
28    la $a0, msg2
29    syscall
  
```

*Nhập xâu:



=> Hiện thị hộp thoại nhập xâu do \$v0 = 54 và cho phép nhập 1 giá trị \$a1 là 1 string.

*Hiện thị độ dài xâu trong hộp thoại:



- Do \$v0 = 56, hiển thị tên hộp thoại và in ra màn hình kết quả độ dài xâu (\$a1)

=> Kết quả chạy độ dài ra đúng

Assignment 5:

*Chương trình:

```
.data
X: .space 50
Y: .space 50
.text
    la      $a3,Y                #nap dia chi cua Y
read:
    li      $v0,8                #SYSCALL doc xau X
    la      $a0,X
    li      $a1,21               #gioi han 20 ky tu
    syscall

reverse:
# dem do dai xau X
    xor     $t3,$0,$0            # t3 = 0
    li      $t0,0                # t0 = i =0
length: add    $t1,$a0,$t0        # t1 = diachi X[i]
    lb      $t2,0($t1)           # load byte t1 vao t2
    beq     $t2,$0,end_length    # t2 = null -> ket thuc
    addi    $t0,$t0,1            # i++
    addi    $t3,$t3,1            # t3++
    j       length
end_length:
    subi    $t3,$t3,1
    andi    $t0,$t0,0            # reset lai bien i
change: add    $t1,$a0,$t3        # t1 = diachi X[i]
    lb      $t2,0($t1)           # load byte t1 vao t2
    add     $t4,$a3,$t0          # t4 = dia chi Y[i]
    sb      $t2,0($t4)           # luu t2 vao t4
    beq     $t2,$0,end_change    # t2=null -> ket thuc
    addi    $t0,$t0,1            # i++
    subi    $t3,$t3,1            # t3-- (vi dang xet tu cuoi)
    j       change
end_change:
printf: # In xau Y
    li      $v0,4
    la      $a0,Y
    syscall
```

*Nhập chuỗi

EditExecute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x3c011001	lui \$1,0x00001001	6: la \$a3,Y #nap dia chi cua Y
	0x00400004	0x34270032	ori \$7,\$1,0x00000032	
	0x00400008	0x24020008	addiu \$2,\$0,0x00000...	8: li \$v0,8 #SYSCALL doc xau X
	0x0040000c	0x3c011001	lui \$1,0x00001001	9: la \$a0,X
	0x00400010	0x34240000	ori \$4,\$1,0x00000000	
	0x00400014	0x24050015	addiu \$5,\$0,0x00000...	10: li \$a1,21 #gioi han 20 ky tu
	0x00400018	0x0000000c	syscall	11: syscall
	0x0040001c	0x00005826	xor \$11,\$0,\$0	14: xor \$t3,\$0,\$0 # t3 = 0
	0x00400020	0x24080000	addiu \$8,\$0,0x00000...	15: li \$t0,0 # t0 = i =0
	0x00400024	0x00884820	add \$9,\$4,\$8	16: length: add \$t1,\$a0,\$t0 # t1 = diachi X[i]
	0x00400028	0x812a0000	lb \$10,0x00000000(\$9)	17: lb \$t2,0(\$t1) # load byte t1 v..
	0x0040002c	0x11400003	beq \$10,\$0,0x00000003	18: beq \$t2,\$0,end_length # t2 = null -> k..

Data Segment

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

Hexadecimal AddressesHexadecimal ValuesAS

Mars MessagesRun I/O

Clear

```

iiiiawaK nal taC
-- program is finished running (dropped off bottom) --

Reset: reset completed.

Cat lan kawaii

```

*Kết quả

EditExecute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x3c011001	lui \$1,0x00001001	6: la \$a3,Y #nap dia chi cua Y
	0x00400004	0x34270032	ori \$7,\$1,0x00000032	
	0x00400008	0x24020008	addiu \$2,\$0,0x00000...	8: li \$v0,8 #SYSCALL doc xau X
	0x0040000c	0x3c011001	lui \$1,0x00001001	9: la \$a0,X
	0x00400010	0x34240000	ori \$4,\$1,0x00000000	
	0x00400014	0x24050015	addiu \$5,\$0,0x00000...	10: li \$a1,21 #gioi han 20 ky tu
	0x00400018	0x0000000c	syscall	11: syscall
	0x0040001c	0x00005826	xor \$11,\$0,\$0	14: xor \$t3,\$0,\$0 # t3 = 0
	0x00400020	0x24080000	addiu \$8,\$0,0x00000...	15: li \$t0,0 # t0 = i =0
	0x00400024	0x00884820	add \$9,\$4,\$8	16: length: add \$t1,\$a0,\$t0 # t1 = diachi X[i]
	0x00400028	0x812a0000	lb \$10,0x00000000(\$9)	17: lb \$t2,0(\$t1) # load byte t1 v..
	0x0040002c	0x11400003	beq \$10,\$0,0x00000003	18: beq \$t2,\$0,end_length # t2 = null -> k..

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	t a c	n a l	a w a k	i i i i	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010020	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010040	\0 c a t	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010060	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010080	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x100100a0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x100100c0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x100100e0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010100	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010120	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0

Hexadecimal AddressesHexadecimal ValuesASCII

Mars MessagesRun I/O

Clear

```

Cat lan kawaii
iiiiawaK nal taC
-- program is finished running (dropped off bottom) --

```

Registers

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x10010000
\$v0	2	0x00000004
\$v1	3	0x00000000
\$a0	4	0x10010032
\$a1	5	0x00000015
\$a2	6	0x00000000
\$a3	7	0x10010032
\$t0	8	0x00000011
\$t1	9	0x1000ffff
\$t2	10	0x00000000
\$t3	11	0xffffffff
\$t4	12	0x10010048
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$a0	16	0x00000000
\$a1	17	0x00000000
\$a2	18	0x00000000
\$a3	19	0x00000000
\$a4	20	0x00000000
\$a5	21	0x00000000
\$a6	22	0x00000000
\$a7	23	0x00000000
\$s0	24	0x00000000
\$s1	25	0x00000000
\$s2	26	0x00000000
\$s3	27	0x00000000
\$s4	28	0x10008000
\$s5	29	0xffffffff
\$s6	30	0x00000000
\$s7	31	0x00000000
\$s8	32	0x00000000
\$s9	33	0x00000000
\$s10	34	0x00000000
\$s11	35	0x00000000
\$s12	36	0x00000000
\$s13	37	0x00000000
\$s14	38	0x00000000
\$s15	39	0x00000000
\$s16	40	0x00000000
\$s17	41	0x00000000
\$s18	42	0x00000000
\$s19	43	0x00000000
\$s20	44	0x00000000
\$s21	45	0x00000000
\$s22	46	0x00000000
\$s23	47	0x00000000
\$s24	48	0x00000000
\$s25	49	0x00000000
\$s26	50	0x00000000
\$s27	51	0x00000000
\$s28	52	0x00000000
\$s29	53	0x00000000
\$s30	54	0x00000000
\$s31	55	0x00000000

⇒ In ra xâu đảo đúng