Bài 5. Nhập xuất dữ liệu với hàm ECALL, xử lý chuỗi ký tự

Họ và tên: Nguyễn Thành Duy

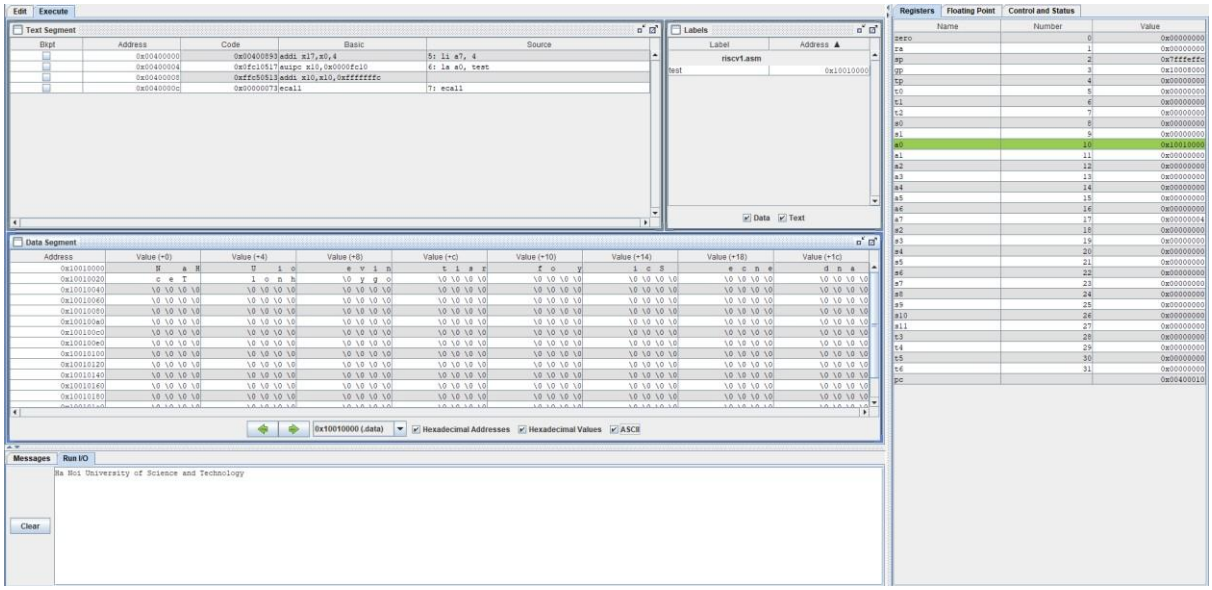
MSSV: 20235696

Assignment 1

Code:

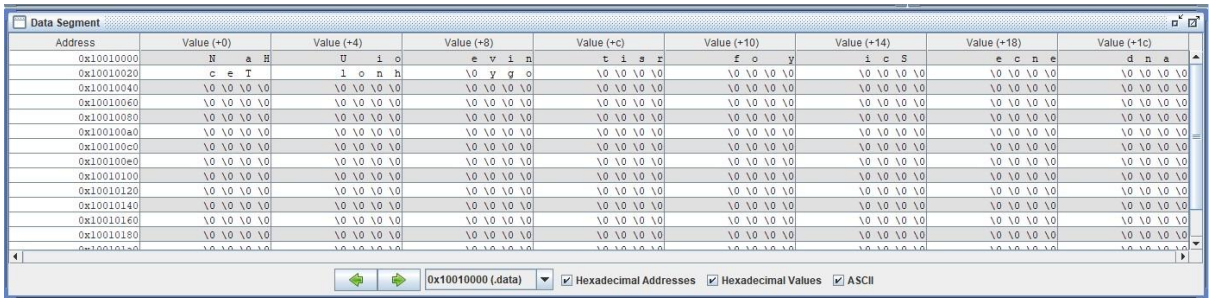
```
# Laboratory Exercise 5, Home Assignment 1
.data
test: .asciz "Ha Noi University of Science and Technology"
.text
li a7, 4
la a0, test
ecall
```

Kết quả sau khi thực hiện chạy chương trình:



Ở cửa sổ Run I/O in ra dòng xâu ký tự test:

Ở cửa sổ Data segment:



Ta thấy chuỗi ký tự lưu trữ vào bộ nhớ theo cách:

- Nạp từng bit vào bộ nhớ khi đủ 4 bit thì sẽ nhảy sang địa chỉ tiếp theo cứ vậy cho tới khi gặp ký tự kết thúc.

Assignment 2

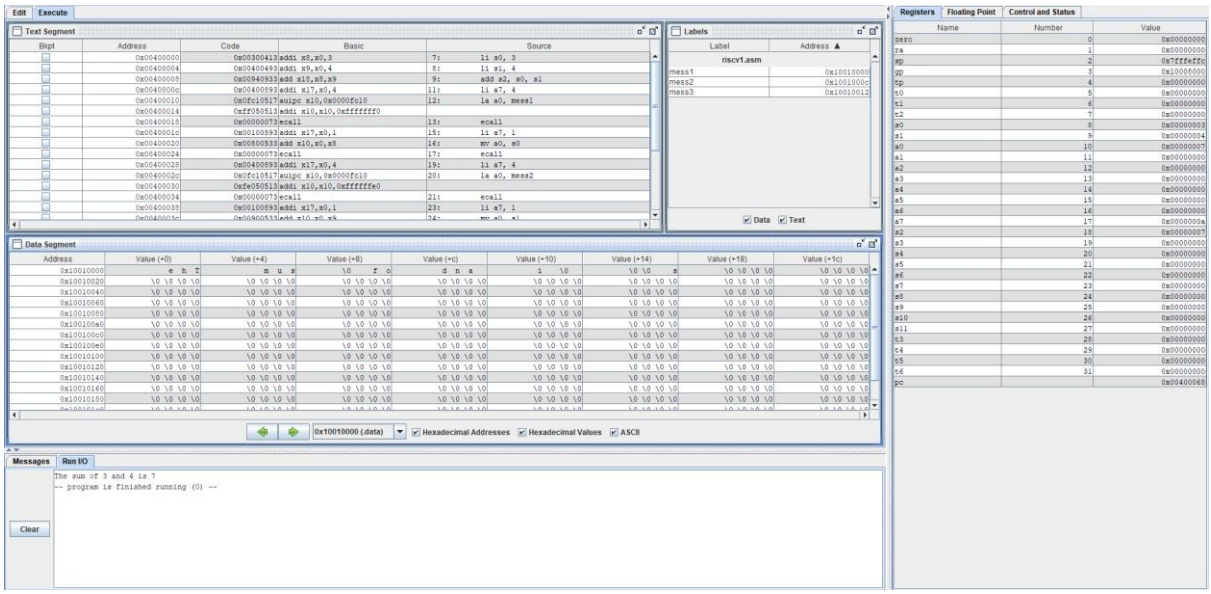
Code:

```
.data
        mess1: .asciz "The sum of "
        mess2: .asciz " and "
        mess3: .asciz " is "

.text
        li s0, 3
        li s1, 4
        add s2, s0, s1
        #
        li a7, 4
        la a0, mess1
        ecall
        #
        li a7, 1
        mv a0, s0
        ecall
        #
        li a7, 4
        la a0, mess2
        ecall
        #
```

```
li a7, 1
mv a0, s1
ecall
#
li a7, 4
la a0, mess3
ecall
#
li a7, 1
mv a0, s2
ecall
#exit
li a7, 10
ecall
```

Kết quả:

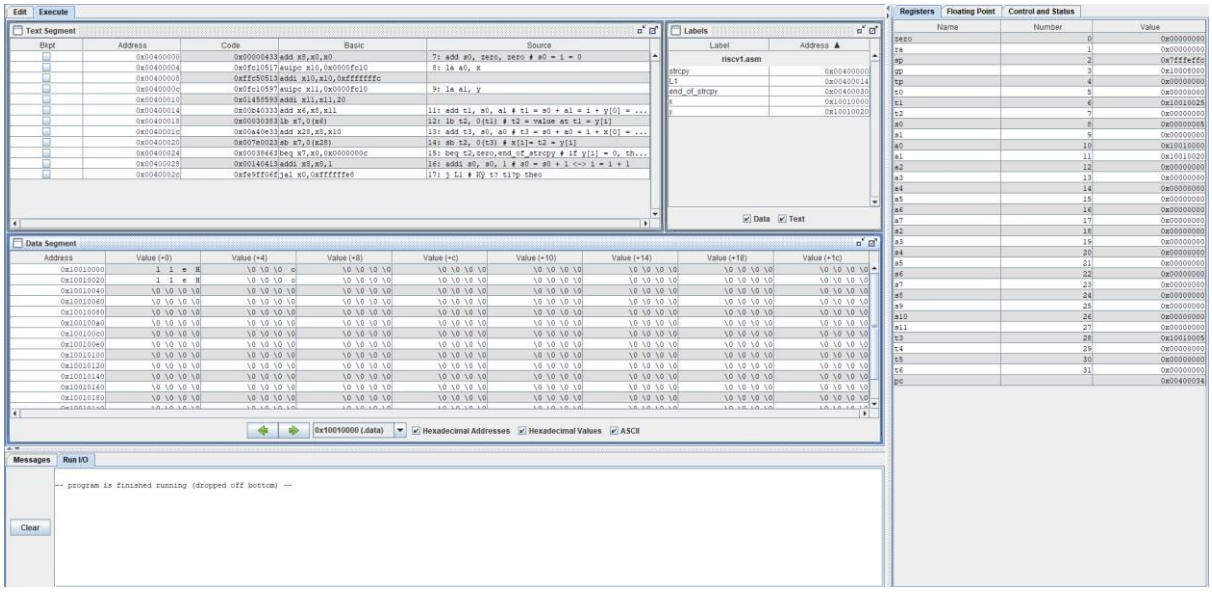


Assignment 3 Code:

```
# Laboratory Exercise 5, Home Assignment 2
.data
x: .space 32 # Chuỗi đích x, khởi tạo là buffer rỗng
y: .asciz "Hello" # Chuỗi nguồn y
 .text
```

```
strcpy:
add s0, zero, zero # s0 = i = 0
la a0, x
la a1, y
L1: add t1, s0, a1 # t1 = s0 + a1 = i + y[0] = address of y[i]
lb t2, 0(t1) # t2 = value at t1 = y[i]
add t3, s0, a0 # t3 = s0 + a0 = i + x[0] = address of x[i]
sb t2, 0(t3) # x[i]= t2 = y[i]
beq t2,zero,end_of_strcpy # if y[i] = 0, then exit
addi s0, s0, 1 # s0 = s0 + 1 <-> i = i + 1
j L1 # Ký tự tiếp theo
end_of_strcpy:
```

Kết quả sau khi chạy đoạn chương trình:



Chuỗi y đã được copy sang chuỗi x

Assignment 4

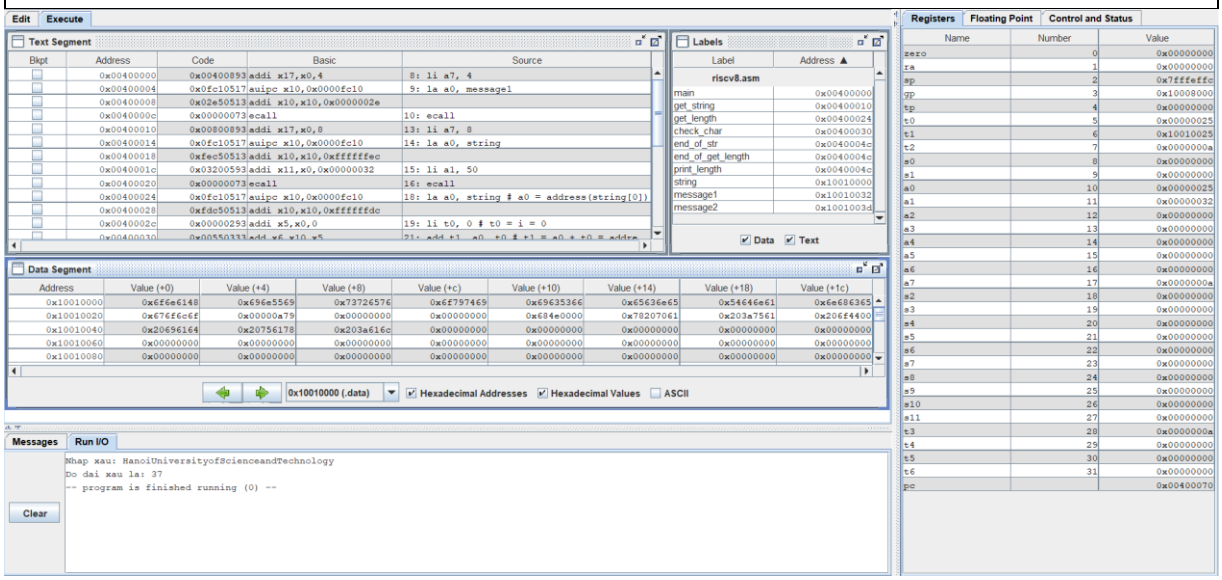
Code:

```
# Laboratory Exercise 5, Home Assignment 3

.data
```

```
string: .space 50
message1: .asciz "Nhap xau: "
message2: .asciz "Do dai xau la: "
.text
main:
li a7, 4
la a0, message1
ecall
get_string:
# TODO Nhập chuỗi ký tự từ bàn phím
li a7, 8
la a0, string
li a1, 50
ecall
get_length:
la a0, string # a0 = address(string[0])
li t0, 0 # t0 = i = 0
check_char:
add t1, a0, t0 # t1 = a0 + t0 = address(string[0]+i)
lb t2, 0(t1) # t2 = string[i]
li t3, 10
beq t2, t3, end_of_str # Nếu là ký tự \n thì kết thúc
beq t2, zero, end_of_str # Nếu là ký tự NULL thì kết thúc
```

```
addi t0, t0, 1 # t0 = t0 + 1 -> i = i + 1
j check_char
end_of_str:
end_of_get_length:
print_length:
# TODO In kết quả ra màn hình
li a7, 4
la a0, message2
ecall
# In độ dài chuỗi
li a7, 1 # Hệ thống gọi để in số nguyên (print_int)
mv a0, t0 # Di chuyển độ dài chuỗi (t0) vào a0
ecall
# Kết thúc chương trình
li a7, 10 # Hệ thống gọi để kết thúc chương trình (exit)
ecall
```



Khi nhập xâu: HanoiUniversityofScienceandTechnology

Kết quả: 37 chính là độ dài của xâu nhập vào

Chương trình sử dụng vòng lặp để tính độ dài của xâu nhập vào không quá 50 ký tự và đếm khi gặp ký tự NULL(nghĩa là khi đủ 50 ký tự) hoặc gặp ký tự xuống dòng (khi nhập không đủ 50 ký tự).

Assignment 5

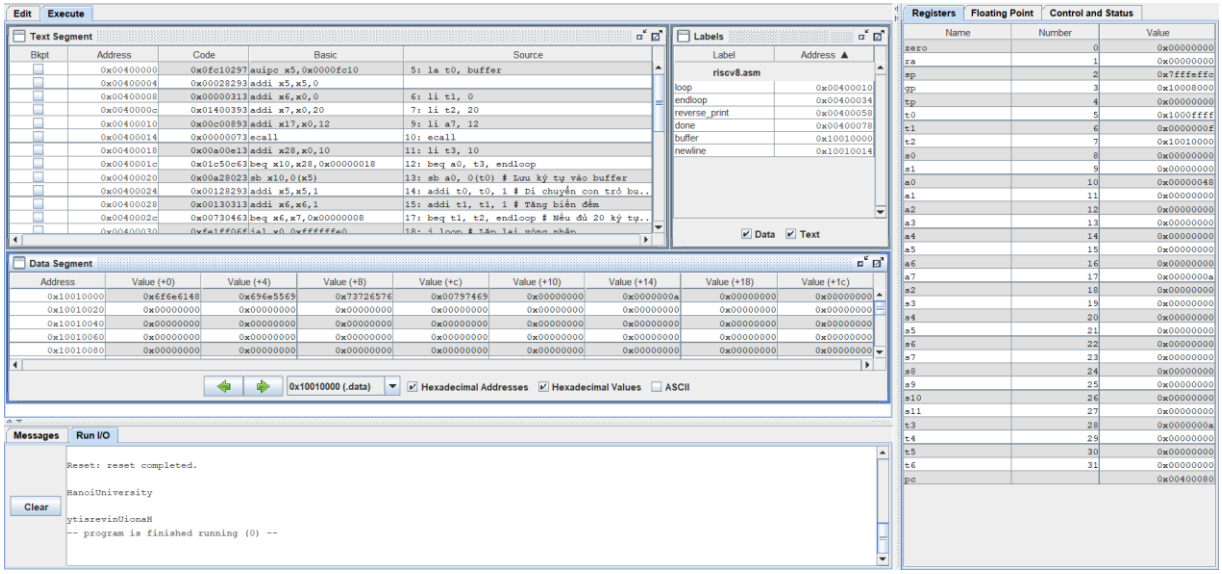
Code:

```
.data
buffer: .space 20
newline: .asciz "\n"
.text
la t0, buffer
li t1, 0
li t2, 20
loop:
li a7, 12
ecall
li t3, 10
beq a0, t3, endloop
sb a0, 0(t0) # Lưu ký tự vào buffer
addi t0, t0, 1 # Di chuyển con trỏ buffer
addi t1, t1, 1 # Tăng biến đếm
# Kiểm tra nếu đã nhập đủ 20 ký tự
beq t1, t2, endloop # Nếu đủ 20 ký tự, kết thúc nhập
j loop # Lặp lại vòng nhập
endloop:
# Kết thúc chuỗi bằng null terminator
sb zero, 0(t0) # Lưu null terminator vào cuối chuỗi
# In ra dòng mới
li a7, 4
la a0, newline
ecall
# In chuỗi ngược lại
la t0, buffer # Địa chỉ buffer
add t0, t0, t1 # Di chuyển đến cuối chuỗi (trừ null terminator)
addi t0, t0, -1 # Trỏ đến ký tự cuối cùng
reverse_print:
# Kiểm tra nếu đã đến đầu chuỗi
la t2, buffer # Địa chỉ đầu buffer
blt t0, t2, done # Nếu con trỏ < đầu buffer, kết thúc
# In ký tự tại vị trí hiện tại
lb a0, 0(t0) # Load ký tự từ buffer
```

```
li a7, 11 # Syscall: print character
ecall
# Di chuyển đến ký tự trước đó
addi t0, t0, -1 # Di chuyển con trỏ lùi lại
j reverse_print # Lặp lại
done:
# Kết thúc chương trình
li a7, 10 # Syscall: exit
ecall
```

TH1: Chuỗi nhập vào không quá 20 ký tự là HanoiUniversity

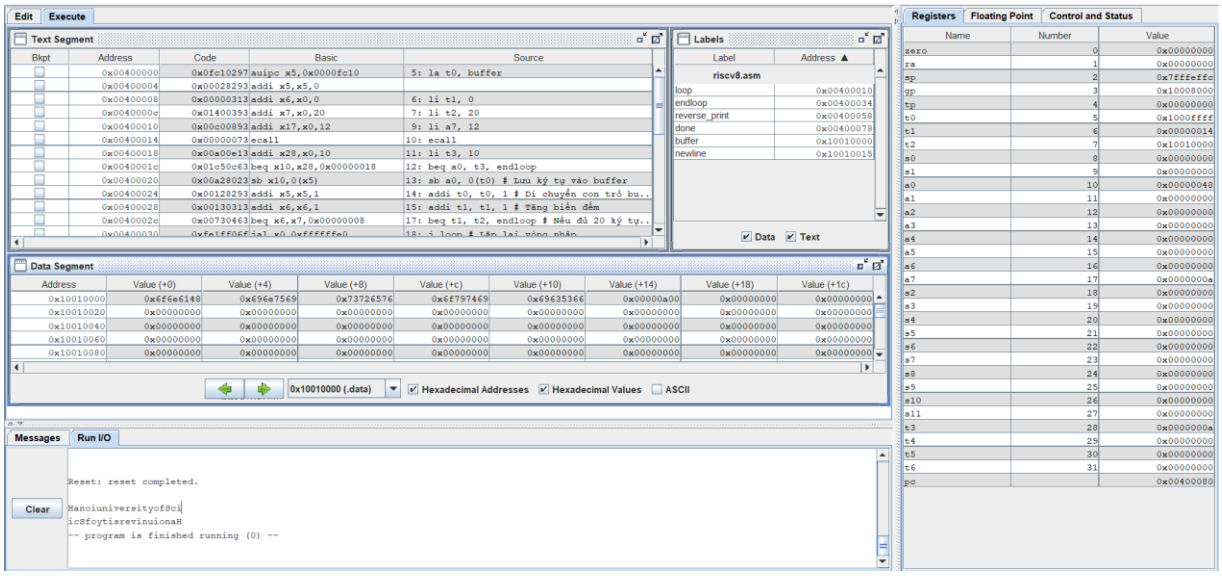
Kết quả: ytisrevinUionaH



TH2: Chuỗi nhập vào quá 20 ký tự là HanoiUniversityofScienceandTechnology

Thì đến HanoiuniversityofSci đủ 20 ký tự thì chương trình dừng và in chuỗi theo thứ tự ngược lại

Kết quả: icSfoytisrevinuionaH


Edit
Execute
Text Segment
Bkpt
Address
Code
Basic
Source
0x00400000 0x0fc10297 auipc x5,0x0000fc10 5: la t0, buffer
0x00400004 0x00028293 addi x5,x5,0
0x00400008 0x00000313 addi x6,x0,0 6: li t1, 0
0x0040000c 0x01400393 addi x7,x0,20 7: li t2, 20
0x00400010 0x00c00893 addi x17,x0,12 9: li a7, 12
0x00400014 0x00000073 ecall 10: ecall
0x00400018 0x00a00e13 addi x28,x0,10 11: li t3, 10
0x0040001c 0x01c50c63 beq x10,x28,0x00000018 12: beq a0, t3, endloop
0x00400020 0x00a28023 sb x10,0(x5) 13: sb a0, 0(t0) # Luu ký tự vào buffer
0x00400024 0x00128293 addi x5,x5,1 14: addi t0, t0, 1 # Di chuyển con trỏ bu..
0x00400028 0x00130313 addi x6,x6,1 15: addi t1, t1, 1 # Tăng biến đếm
0x0040002c 0x00730463 beq x6,x7,0x00000008 17: beq t1, t2, endloop # Nếu đủ 20 ký tự..
0x00400030 0xfe1ff06f jal x0,0xfffffffe0 18: j loop # Lặp lại vòng nhập
Labels
Label
Address ▲
riscv8.asm
loop 0x00400010
endloop 0x00400034
reverse_print 0x00400058
done 0x00400078
buffer 0x10010000
newline 0x10010015
Data
Text
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x10010000 0x6f6e6148 0x696e7569 0x73726576 0x6f797469 0x69635366 0x00000a00 0x00000000 0x00000000
0x10010020 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010040 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010060 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010080 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Messages
Run I/O
Reset: reset completed.
Clear
HanoiuniversityofScI
icSfoytisrevinuionaH
-- program is finished running (0) --
Registers
Floating Point
Control and Status
Name
Number
Value
zero 0 0x00000000
ra 1 0x00000000
sp 2 0x7fffeffc
gp 3 0x10008000
tp 4 0x00000000
t0 5 0x1000ffff
t1 6 0x00000014
t2 7 0x10010000
s0 8 0x00000000
s1 9 0x00000000
a0 10 0x00000048
a1 11 0x00000000
a2 12 0x00000000
a3 13 0x00000000
a4 14 0x00000000
a5 15 0x00000000
a6 16 0x00000000
a7 17 0x0000000a
s2 18 0x00000000
s3 19 0x00000000
s4 20 0x00000000
s5 21 0x00000000
s6 22 0x00000000
s7 23 0x00000000
s8 24 0x00000000
s9 25 0x00000000
s10 26 0x00000000
s11 27 0x00000000
t3 28 0x0000000a
t4 29 0x00000000
t5 30 0x00000000
t6 31 0x00000000
pc 0x00400080