Báo cáo tuần 5
Thực hành kiến trúc máy tính

Họ tên: Phan Minh Anh Tuấn
MSSV: 20205227

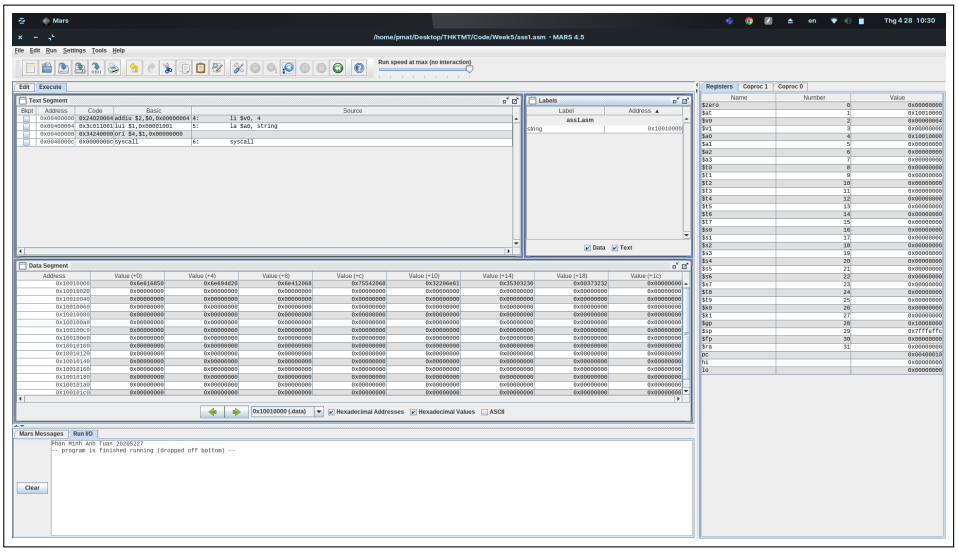
Mục lục

1 Assignment 1

```
.data
        string: .asciiz "Phan Minh Anh Tuan 20205227"
.text
        li $v0, 4
        la $a0, string
        syscall
```

Hình 1: Code của Assignment 1

Giải thích: Phần dữ liệu đầu vào được thể hiện ở 2 dòng đầu. Phần thực thi, để in ra màn hình, cần gán thanh ghi $v0 giá trị 4 (print string) sau đó load địa chỉ string vào $a0. Kết quả của đoạn code được thể hiện tại Hình 2



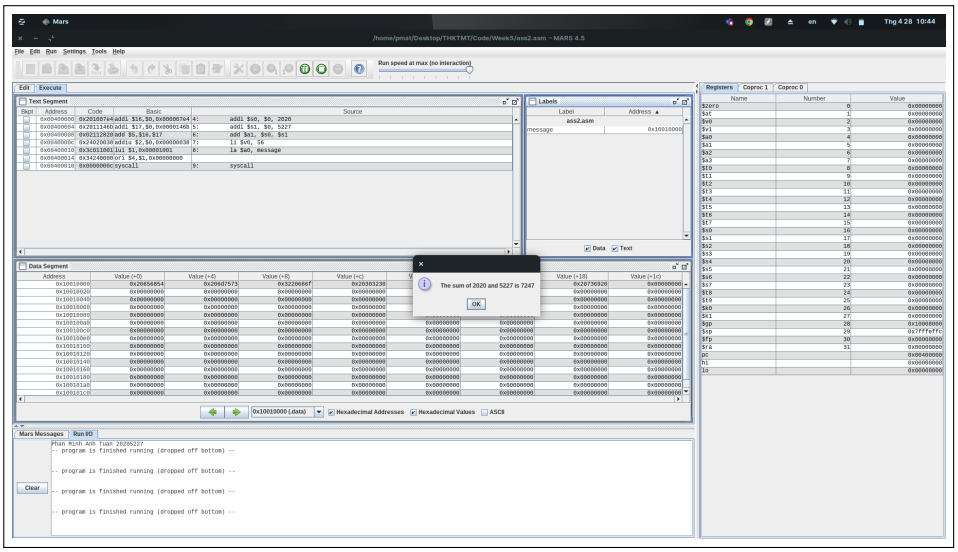
Hình 2: Kết quả của Assignment 1

2 Assignment 2

```
.data
        message: .asciiz "The sum of 2020 and 5227 is "
.text
        addi $s0, $0, 2020
        addi $s1, $0, 5227
        add $a1, $s0, $s1
        li $v0, 56
        la $a0, message
        syscall
```

Hình 3: Code của Assignment 2

Giải thích: Thực hiện gán giá trị $s0 và $s1 sau đó tính tổng và gán vào $a1. Để in ra màn hình, cần gán thanh ghi $v0 giá trị 56 (MessageDialogInt). Kết quả của đoạn code được thể hiện tại Hình 2



Hình 4: Kết quả của Assignment 2

3 Assignment 3

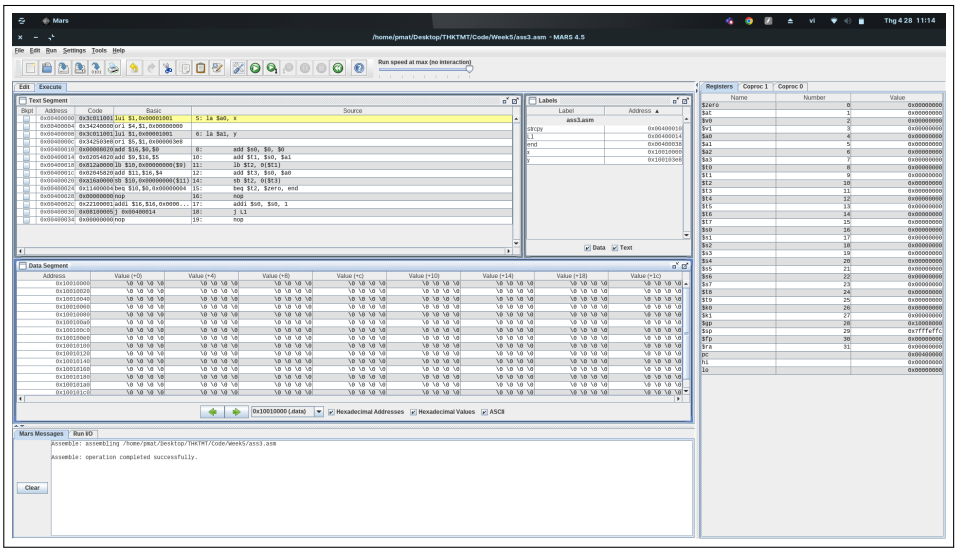
```
.data
        x: .space 1000
        y: .asciiz "Tuan"
.text
la $a0, x
la $a1, y
strcpy:
        add $s0, $0, $0
L1:
        add $t1, $s0, $a1
        lb $t2, 0($t1)
        add $t3, $s0, $a0

        sb $t2, 0($t3)
        beq $t2, $zero, end
        nop
        addi $s0, $s0, 1
        j L1
        nop
end:
```

Hình 5: Code của Assignment 3

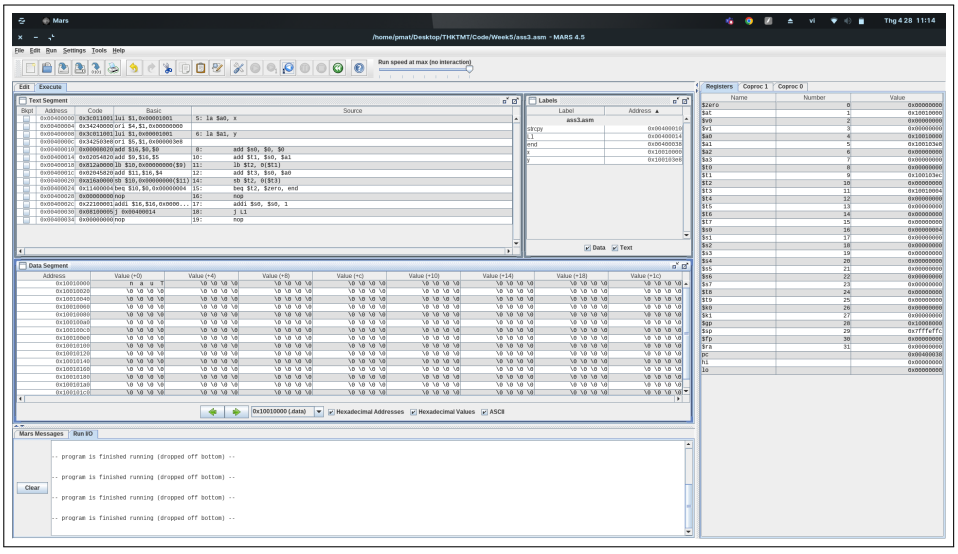
Trong đó:

- x là xâu được copy ra
- y là xâu gốc
- $s0 là current của xâu, bắt đầu từ 0
- $t1 là địa chỉ của xâu gốc tại current $s0
- $t2 là giá trị tại địa chỉ $t1
- $t3 là địa chỉ của xâu được copy tại current $s0

Giải thích: Chương trình thực hiện copy từng phần tử 1 từ xâu y sang xâu x. Với mỗi bước lặp mình gán giá trị $t2 vào giá trị tại địa chỉ $t3, cho đến khi $t2 bằng 0 thì dừng lại.



Hình 6: Kết quả trước khi chạy



Hình 7: Kết quả sau khi chạy

4 Assignment 4

```
.data
        string: .space 50
        Message1:       .asciiz "Nhap xau: "
        Message2:       .asciiz "Do dai la: "
.text
main:
get_string:
        li $v0, 54
        la $a0, Message1
        la $a1, string
        la $a2, 50
        syscall
get_length:
        la $a0, string
        xor $t5, $0, $0
        xor $t0, $0, $0
check_char:
        add $t1, $a0, $t0
        lb $t2, 0($t1)
        beq $t2, $0, end_of_str
        addi $t5, $t5, 1
        addi $t0, $t0, 1
        j check_char
end_of_str:
addi $t5, $t5, -1
print_length:
        li $v0, 56
        la $a0, Message2
        add $a1, $t5, $0
        syscall
```

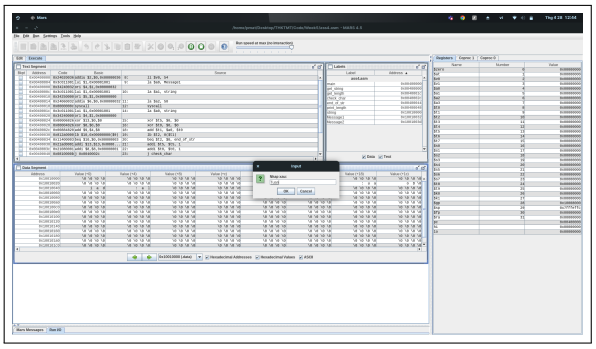
Hình 8: Code của Assignment 4

Trong đó:

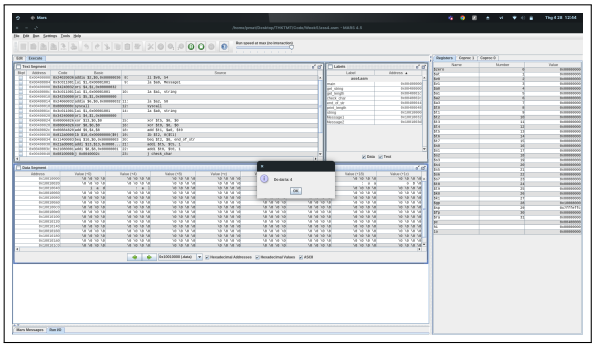
- string: Chuỗi nhập vào từ bàn phím
- Message1: Thông báo thứ 1
- Message2: Thông báo thứ 2
- get_string: Phần xử lý nhập vào từ bàn phím
- get_length: Đếm độ dài của xâu
- check_char: Kiểm tra xem đó có phải kí tự hợp lệ không, nếu không thoát khỏi vòng lặp
- print_length: In ra số phần tử của xâu

Giải thích:

- get_string: Nhập xâu vào từ bàn phím, sử dụng $v0 = 54 (InputDialogString). Xâu được nhập vào lưu trữ tại string.
- get_length: Lấy địa chỉ của xâu string lưu vào thanh ghi $a0. $t5 là thanh ghi lưu trữ độ dài của xâu, $t0 là offset.
- check_char: $t1 là địa chỉ của kí tự vị trí $t0 của xâu string. $t2 là giá trị của địa chỉ $t1. Nếu $t2 là null, kết thúc vòng lặp. Nếu không, thực hiện tăng bộ đếm và độ dài của xâu và tiếp tục vòng lặp.
- print_length: In ra số phần tử của xâu qua $v0 (MessageDialogInt)



Hình 9: Kết quả trước khi chạy



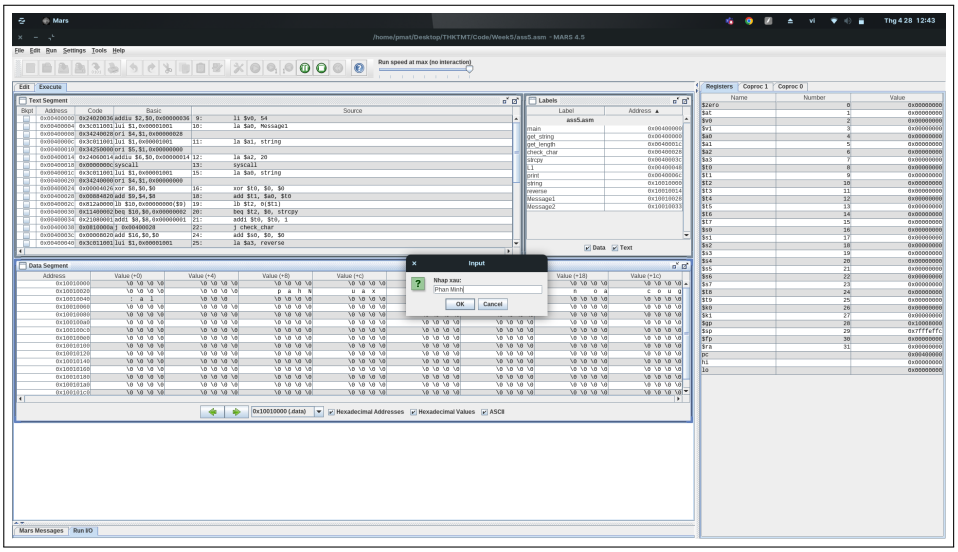
Hình 10: Kết quả sau khi chạy

5 Assignment 5

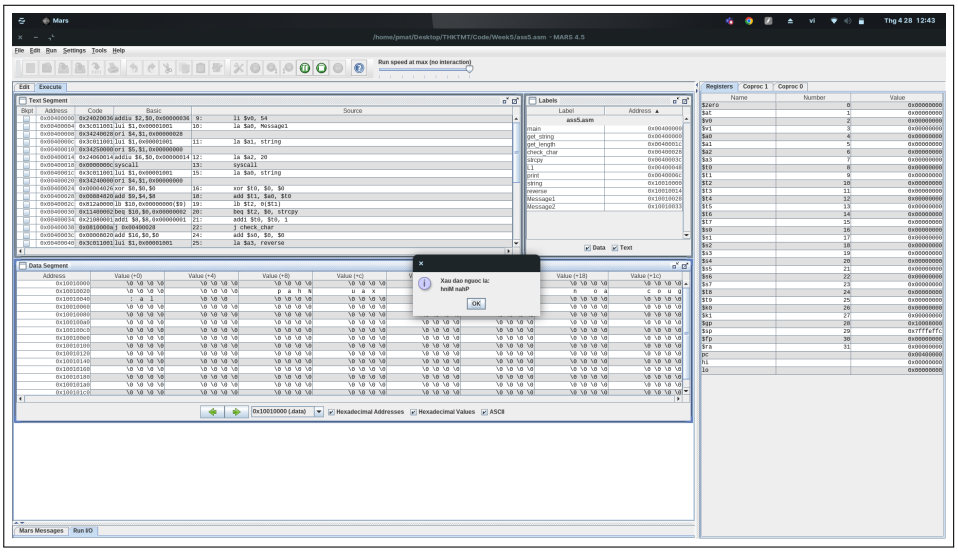
```
.data
        string: .space 20
        reverse: .space 20
        Message1:       .asciiz "Nhap xau: "
        Message2:       .asciiz "Xau dao nguoc la: "
.text
main:
get_string:
        li $v0, 54
        la $a0, Message1
        la $a1, string
        la $a2, 20
        syscall
get_length:
        la $a0, string
        xor $t0, $0, $0
check_char:
        add $t1, $a0, $t0
        lb $t2, 0($t1)
        beq $t2, $0, strcpy
        addi $t0, $t0, 1
        j check_char
strcpy:
        add $s0, $0, $0
        la $a3, reverse
L1:
        addi $t1, $t1, -1
        lb $t2, 0($t1)
        add $t3, $a3, $s0
        sb $t2, 0($t3)
        addi $s0, $s0, 1
        beq $s0, $t0, print
        nop
        j L1
        nop
print:
        li $v0, 59
        la $a0, Message2
        la $a1, reverse
        syscall
```

Hình 11: Code của Assignment 5

Giải thích: ta đếm số lượng phần tử (Assignment 4), sau đó strcpy (Assignment 3) từ cuối lên đầu. Đặt giới hạn cả string (input) và reverse(output) là 20.



Hình 12: Kết quả trước khi chạy



Hình 13: Kết quả sau khi chạy