Laboratory Exercise 4 – Report:

Arithmetic and Logical operation

Lê Văn Duẩn - 20194508

1. Assignment 1

- Mã nguồn:

```
#Laboratory Assignment 1
.text
li      $s1, 2019
li      $s2, 4508
start:
li      $t0, 0          # default status
addu    $s3, $s1, $s2         # s3 = s1 + s2
xor     $t1, $s1, $s2# test if $s1 and $s2 have the same sign

bltz    $t1,EXIT       # If not, exit
slt     $t2,$s3,$s1
bltz    $s1,NEGATIVE       # test if $s1 and $s2 is negative?
beq     $t2,$0,EXIT  # s1 and s2 are positive

# if $s3 > $s1 then the result is overflow
j       OVERFLOW
NEGATIVE:
```

bne $t2,$0,EXIT

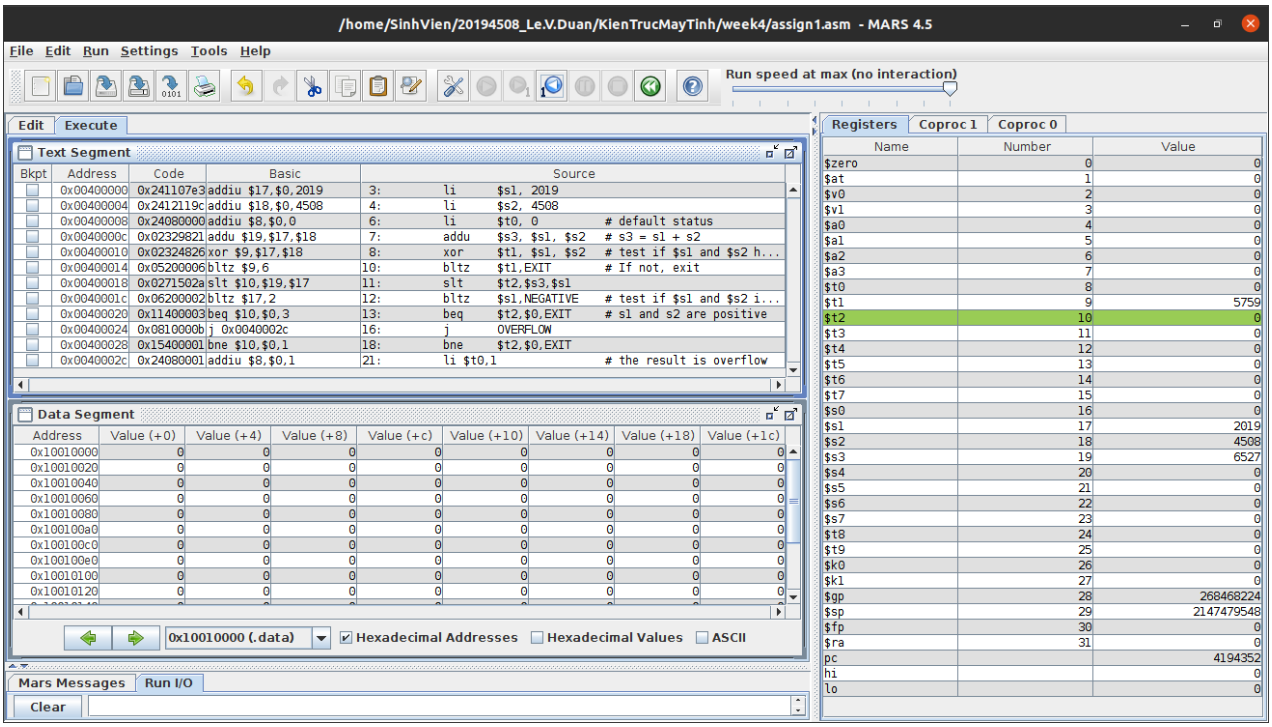
if $s3 < $s1 the the result is not overflow

OVERFLOW:

li $t0,1 # the result is overflow

EXIT:

- Kết quả chạy mô phỏng:



- Giải thích:

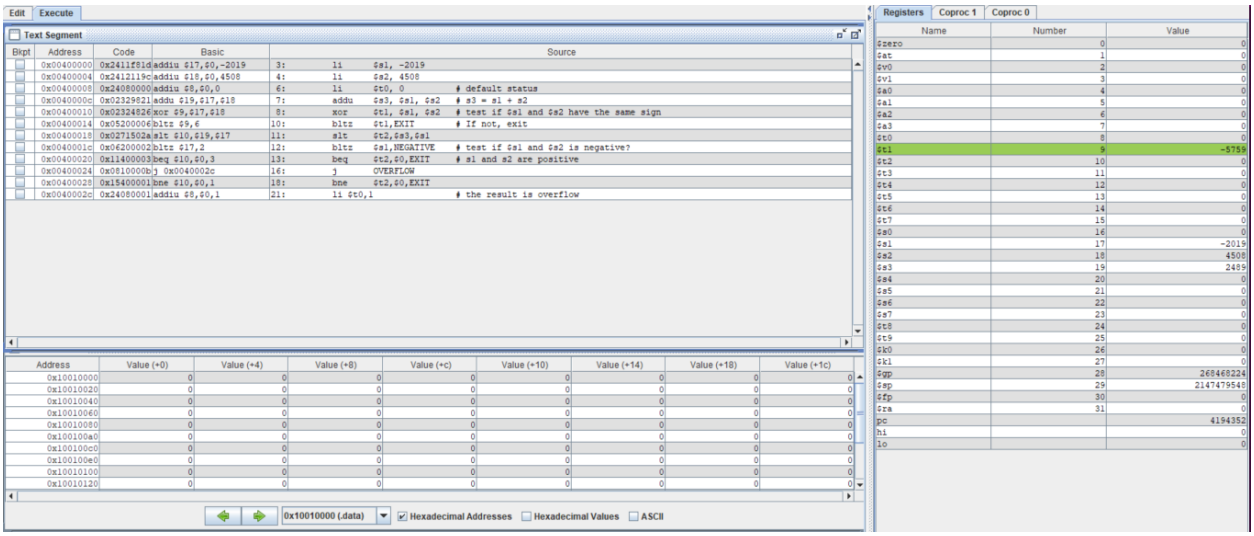
Vì giá trị ban đầu khởi tạo cho s1 và s2 là cùng dấu và dương nên sau lệnh xor kết quả trong thanh ghi t1 là số dương vì vậy khi đến câu lệnh so sánh bltz nó sẽ không đi đến nhãn lệnh EXIT

Tại slt $t2,$s3,$s1 so sánh giá trị s1 và s3 vì trong ví dụ này s3 > s1 -> t2 = 0

Lệnh bltz $s1,NEGATIVE -> kiểm tra xem s1 có âm hay không -> ví dụ này dương -> không đến nhãn NEGATIVE

lệnh beq $t2,$0,EXIT kiểm tra xem t2 = 0 hay ko nếu = 0 nghĩa là s3 không bị tràn -> EXIT vì vậy t2 = 0

- TH khác dấu:



Trong TH này s1 < 0 nên sau khi thực hiện lệnh xor kết quả trong thanh ghi t1 < 0 vì vậy 2 số trái dấu và sẽ không xảy ra tràn -> lệnh bltz $t1,EXIT sẽ nhảy đến nhãn lệnh EXIT và kết thúc.

2. Assignment 2

- Mã nguồn:

#Laboratory Assignment 2

.text

li $s0,0x20194508 # test value MSSV

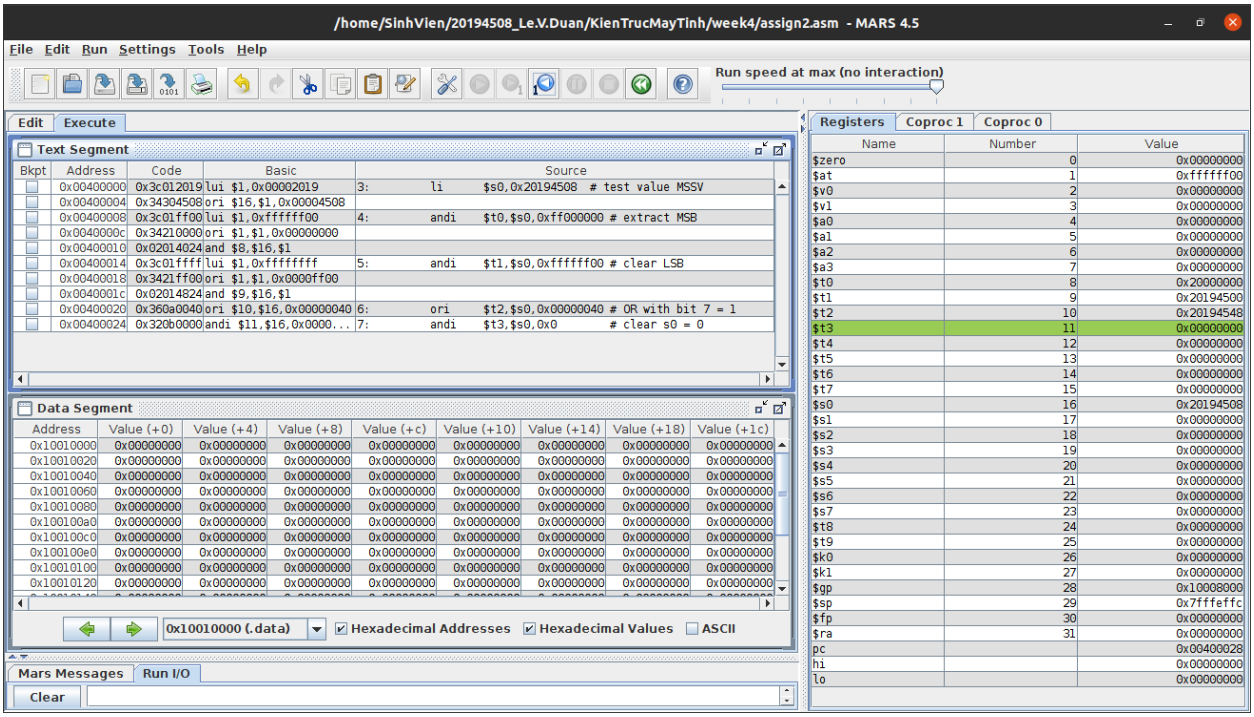
andi $t0,$s0,0xff000000 # extract MSB

andi $t1,$s0,0xffffff00 # clear LSB

ori $t2,$s0,0x00000040 # OR with bit 7 = 1

andi $t3,$s0,0x0 # clear s0 = 0

- Kết quả chạy:



Thanh ghi t0 = 0x20000000 -> etract MSB

Thanh ghi t1 = 0x20194500 -> clear LSB

Thanh ghi t2 = 0x20194548 -> sau khi OR với dãy bit toàn 0 và bit thứ 7 = 1 (hex: 00000040)

Thanh ghi t3 = 0x00000000 -> reset s0 bằng cách and với 0.

3. Assignment 3

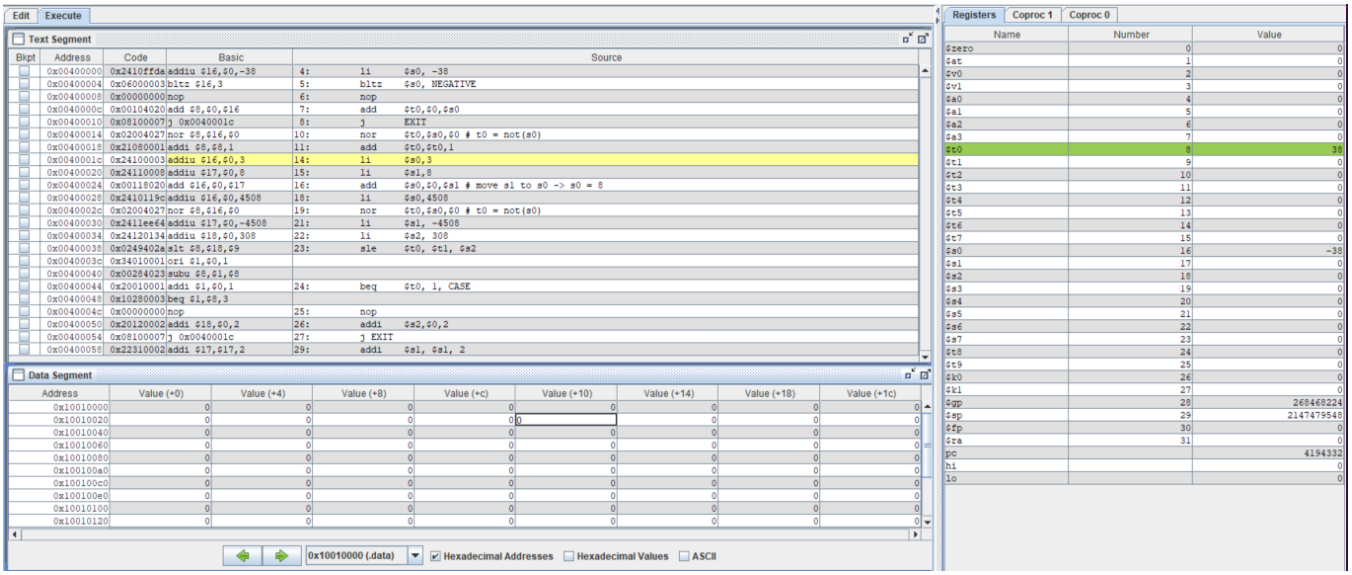
-Mã nguồn:

```
#Laboratory Assignment 3

.text

abs:

li      $s0, -38

bltz    $s0, NEGATIVE

nop
```

```
add    $t0,$0,$s0
j      EXIT
NEGATIVE:
nor    $t0,$s0,$0 # t0 = not(s0)
add    $t0,$t0,1
EXIT:

move:
li     $s0,3
li     $s1,8
add    $s0,$0,$s1 # move s1 to s0 -> s0 = 8
not:
li     $s0,4508
nor    $t0,$s0,$0 # t0 = not(s0)
ble:
li     $s1, -4508
li     $s2, 308
sle    $t0, $t1, $s2
beq    $t0, 1, CASE
nop
addi   $s2,$0,2
j EXIT
CASE:
addi   $s1, $s1, 2
```

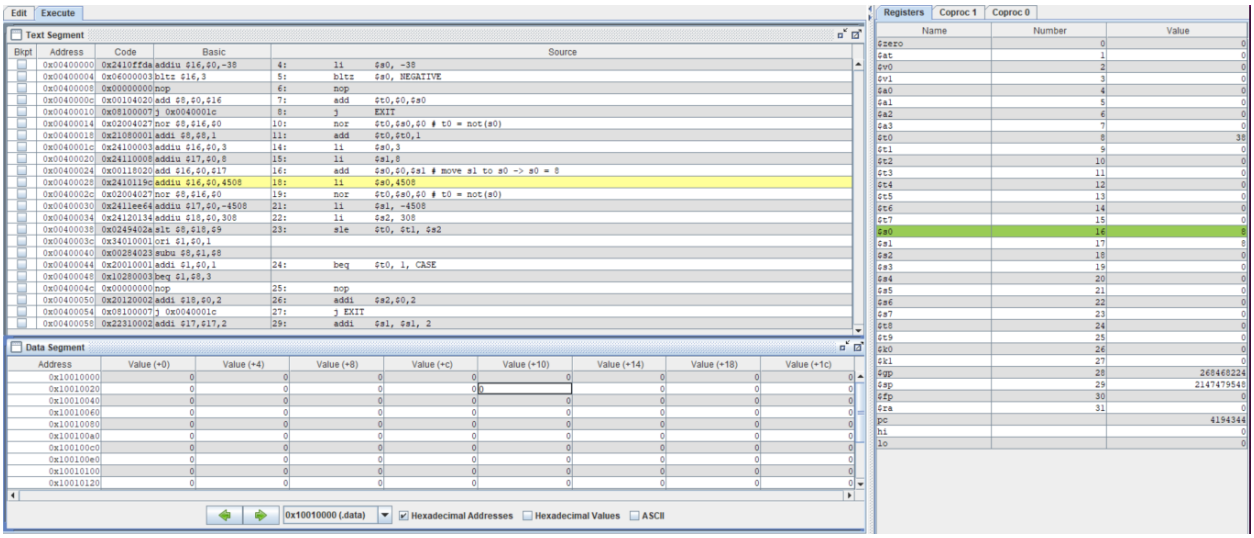
- Kết quả chạy mô phỏng:

+ abs:

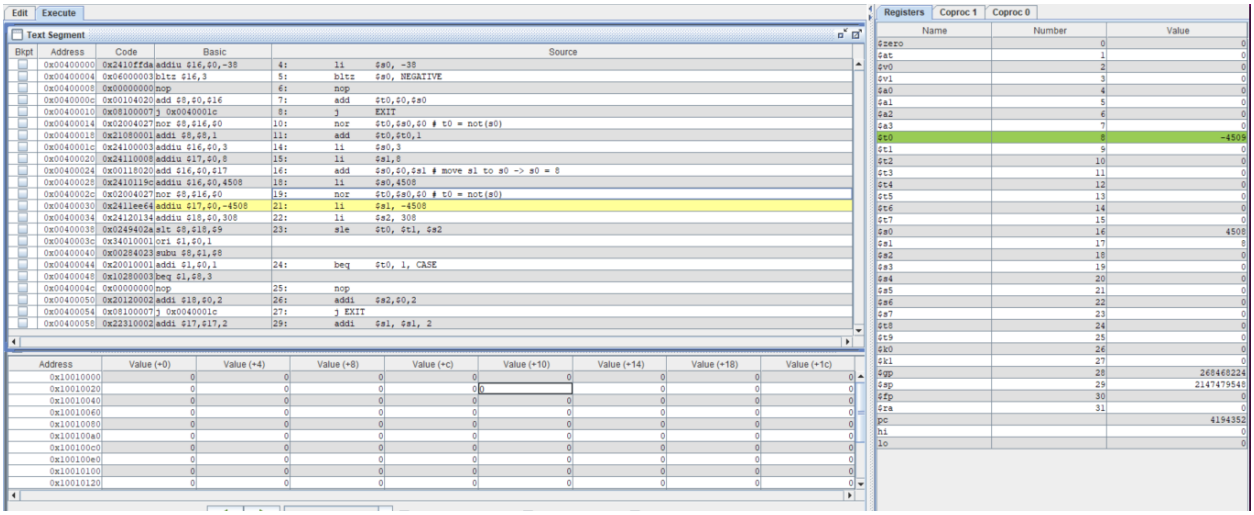


- Nếu s0 là dương -> abs là chính nó. TH s0 âm -> abs là số đối của s0 -> tính not bằng lệnh nor và +1 để tính số bù 2 hay số đối

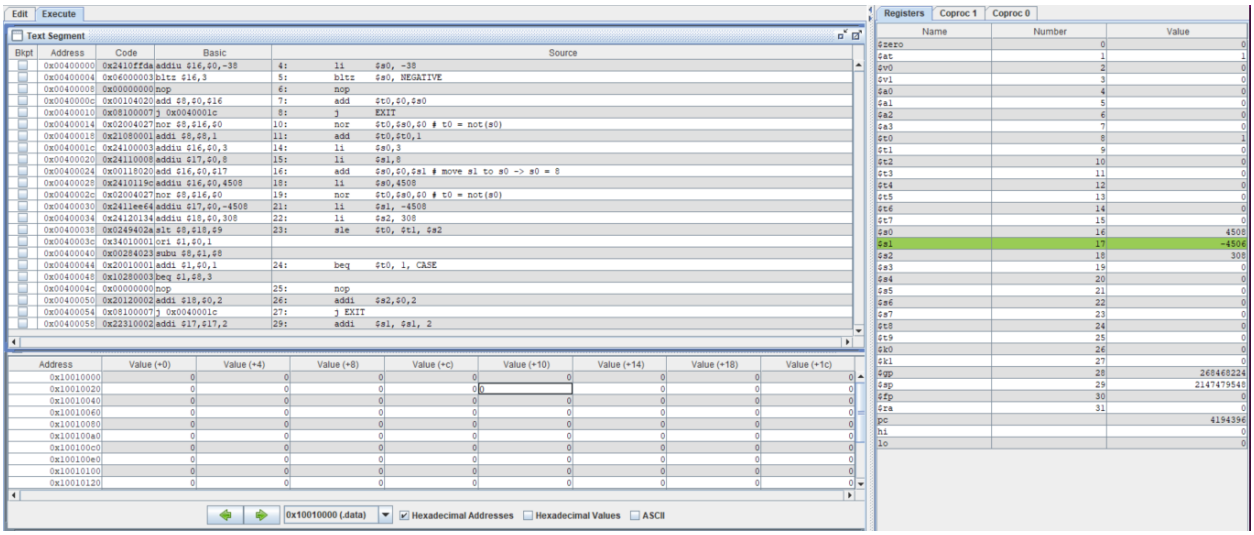
+ move:



+ not: dùng lệnh nor để thay thế cho toán tử not



+ ble:



4. Assignment 4

- Mã nguồn:

#Laboratory Assignment 4

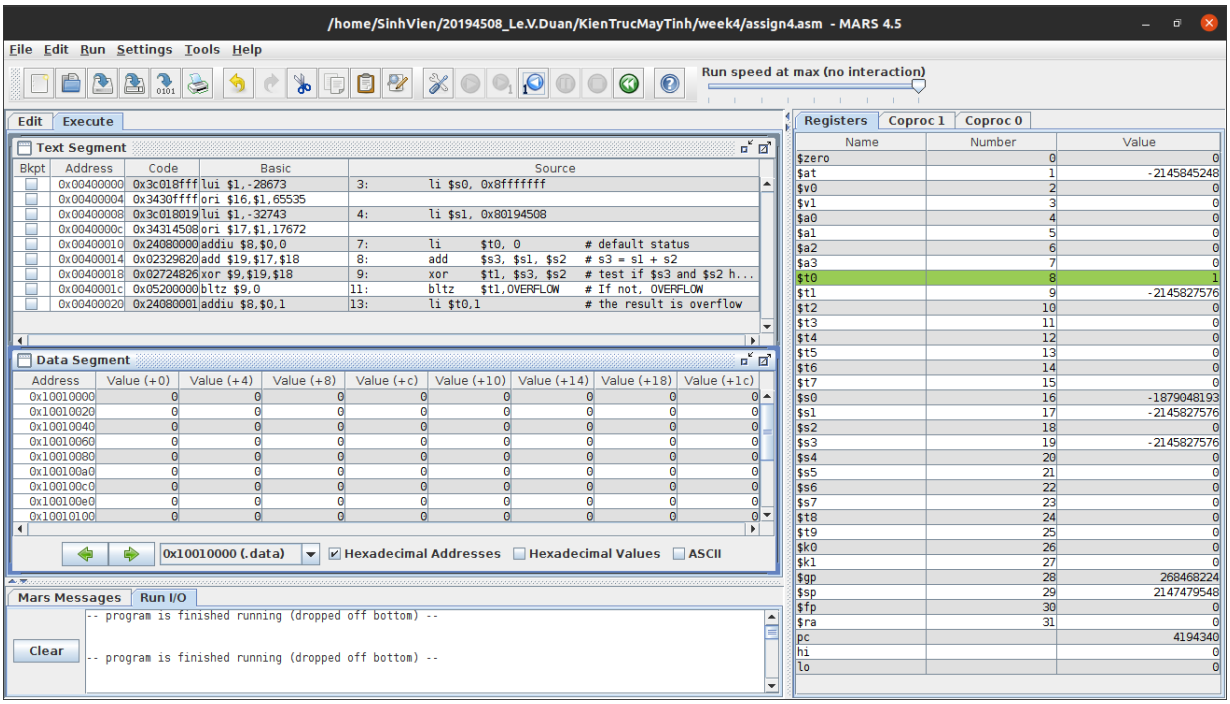
.text

li $s0, 0x8fffffff

li $s1, 0x80194508

```
start:
li      $t0, 0          # default status
add     $s3, $s1, $s2           # s3 = s1 + s2
xor     $t1, $s3, $s2# test if $s3 and $s2 have the same sign

bltz    $t1,OVERFLOW      # If not, OVERFLOW
OVERFLOW:
li $t0,1                # the result is overflow
```

- Kết quả chạy:



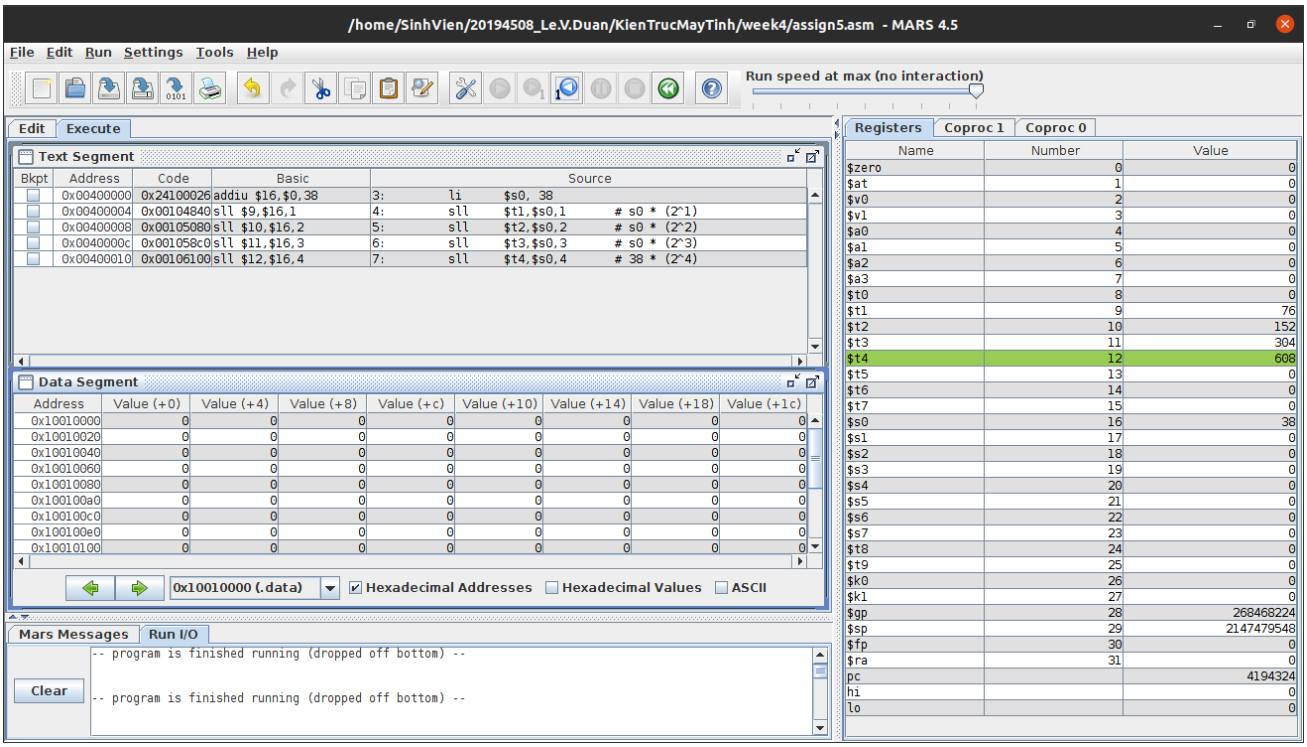
5. Assignment 5

- Mã nguồn:

```
#Laboratory Assignment 5
.text
```

```
li      $s0, 38
sll     $t1,$s0,1     # s0 * (2^1)
sll     $t2,$s0,2     # s0 * (2^2)
sll     $t3,$s0,3     # s0 * (2^3)
sll     $t4,$s0,4     # 38 * (2^4)
```

- Kết quả chạy:



- Nhân một số với lũy thừa với 2 bằng cách dịch bit

t1 = 38 * 2 = 76

t2 = 38 * 4 = 152

t3 = 38 * 8 = 304

t4 = 38 * 16 = 608