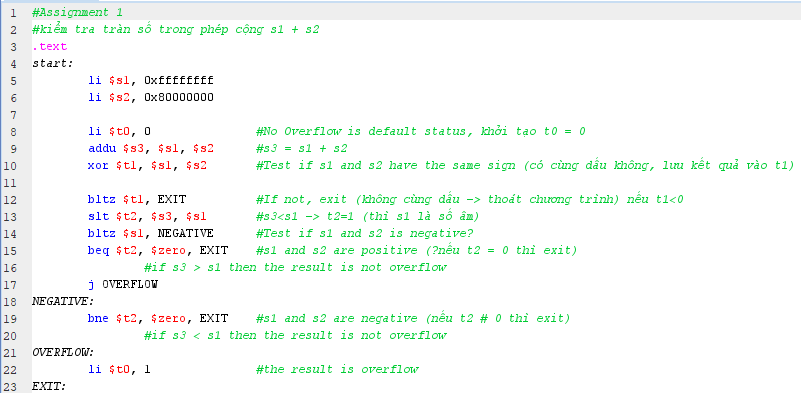
**Bài thực hành số 4**

**Lớp: 139365 – Học phần: Thực hành Kiến Trúc Máy Tính**

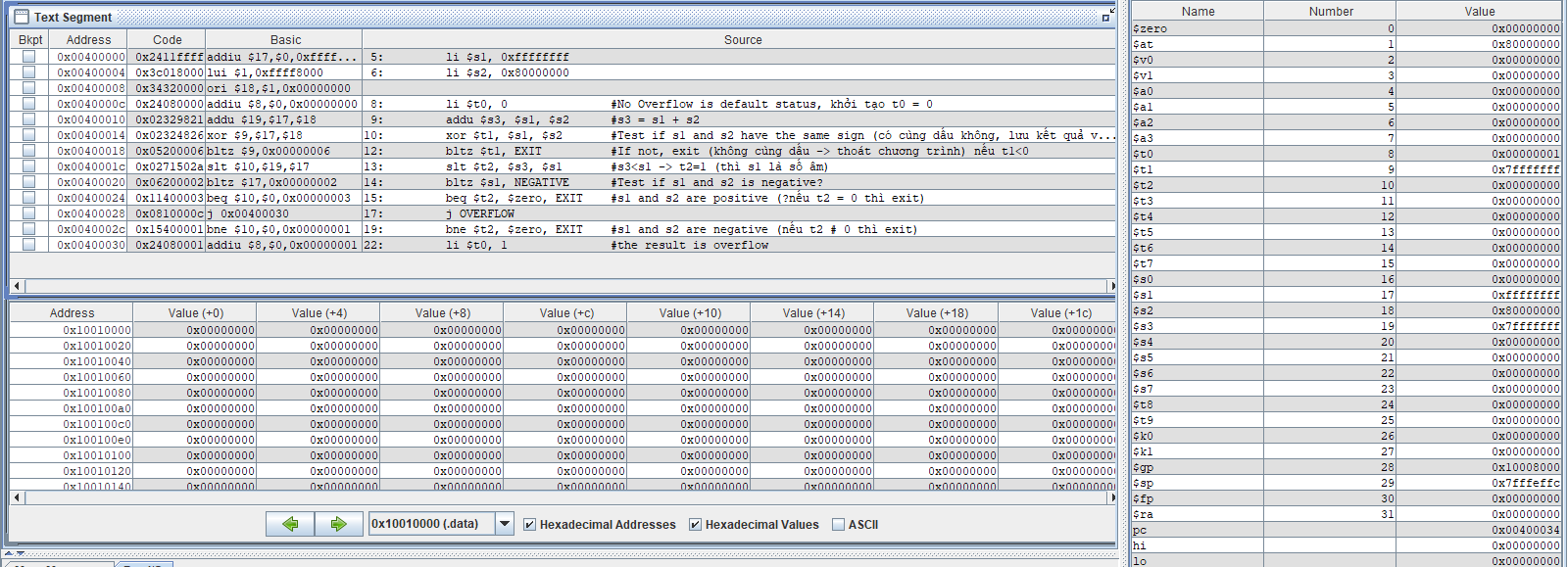
**Họ và tên: Đinh Thị Hồng Phúc MSSV: 20215118**

Bài 1.



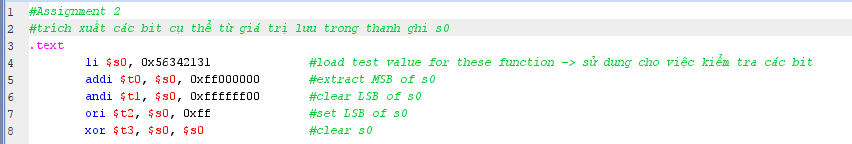
*Thực hiện gõ chương trình vào công cụ* ***MARS***

* Thực hiện dòng lệnh 5 và 6: Các thanh ghi $s1 và $s2 được gán với các giá trị lần lượt là 0xffffffff và 0x80000000
* Thực hiện dòng lệnh số 8: Thanh ghi $t0 được gán giá trị 0 (tương ứng là không bị tràn số)
* Thực hiện dòng lệnh số 9: Dòng lệnh thực hiện ghi giá trị của: s3 = s1 + s2
* Thực hiện dòng lệnh số 10: Dòng lệnh thực hiện: s1^s2, lưu giá trị và t1, kiểm tra s1 có cùng dấu với s2 không
* Thực hiện dòng lệnh số 12: Nếu t1< 0 thì 2 số không cùng dấu, thoát chương trình (EXIT)
* Thực hiện dòng lệnh số 13, 15: So sánh s3 có nhỏ hơn s1 hay không (tổng nhỏ hơn số hạng đối với số dương)
* Thực hiện dòng lệnh số 14: Kiểm tra s1, s2 có là số âm hay không, nếu là số âm thì tổng nhỏ hơn số hạng thì EXIT không thì OVERFLOW
* Thực hiện dòng lệnh số 17: chuyển đến phần OVERFLOW
* Thực hiện dòng lệnh số 19: Nếu t2 ≠ 0 thì EXIT
* Thực hiện dòng lệnh số 22: gán t0 = 1 (tràn số)



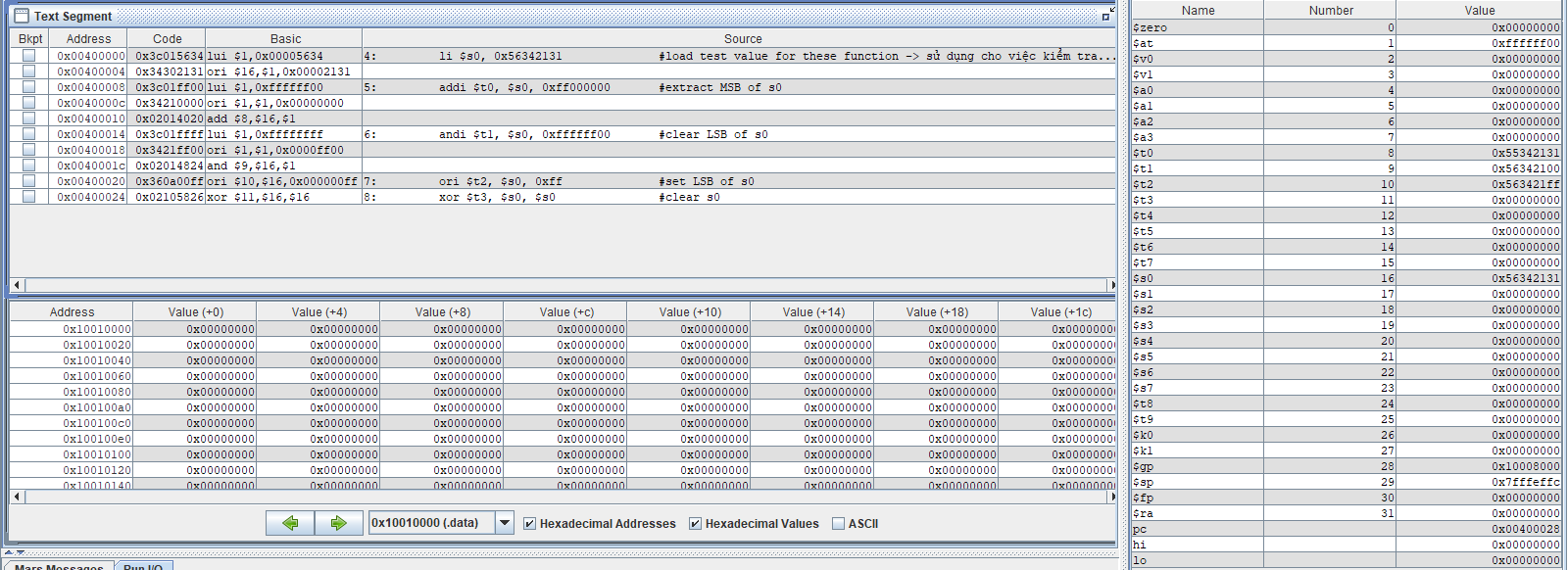
*Thực hiện chạy chương trình với* ***MARS***

Bài 2.



*Thực hiện gõ chương trình vào công cụ* ***MARS***

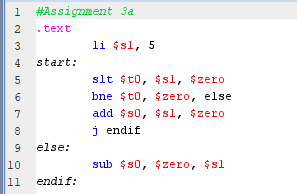
* Dòng 4: Thanh ghi s0 được gán giá trị 0x56342131
* Dòng 5: Lấy giá trị MSB của s0 bằng phép and với 0xff000000. Khi đó t0 = 0x55342131
* Dòng 6: Xóa giá trị LSB của s0 bằng phép and với 0xffffff00. Khi đó t1 = 0x56342100
* Dòng 7: Set LSB của s0 bằng phép or với 0xff. Khi đó t2 = 0x563421ff
* Dòng 8: Xóa giá trị s0 bằng cách xor s0 với chính nó. Khi đó t3 = 0x00000000



*Thực hiện chạy chương trình với* ***MARS***

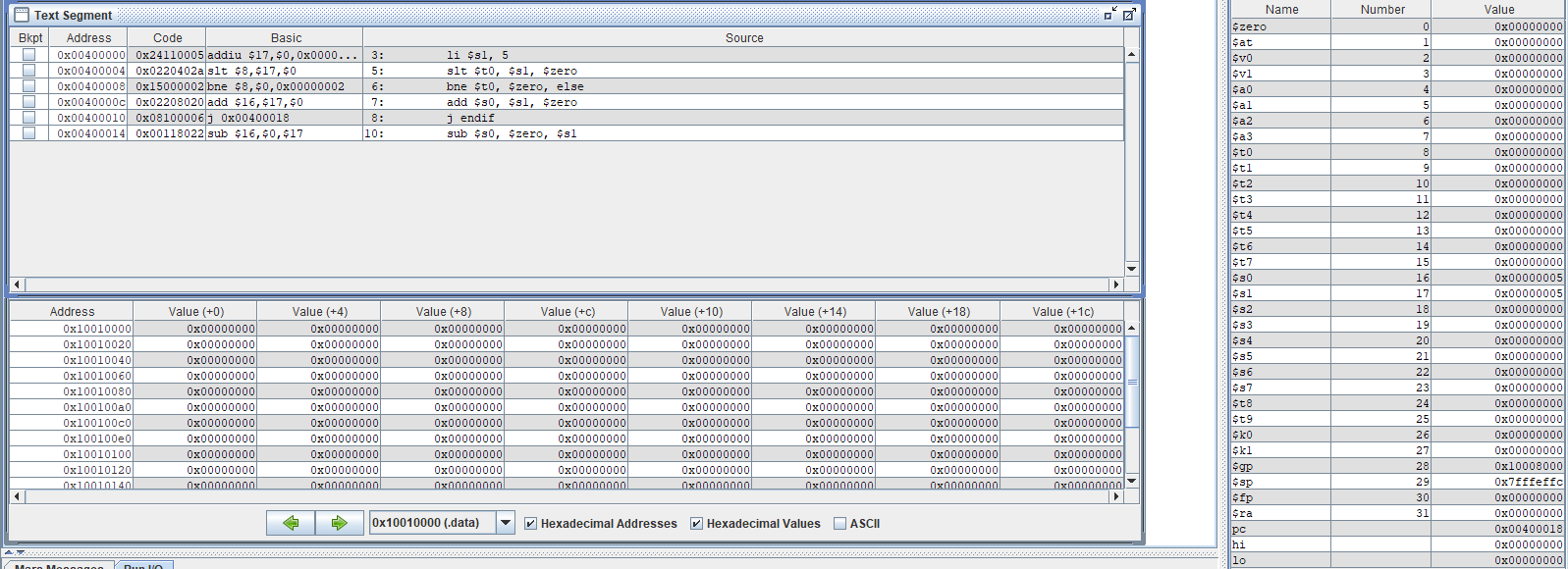
Bài 3.

1. abs



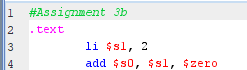
*Thực hiện gõ chương trình vào công cụ* ***MARS***

* Dòng 3: Thanh ghi s1 được gán với giá trị 5
* Dòng 5: So sánh s1 với 0, gán kết quả vào t0 (kiểm tra s1 là số âm hay dương)
* Dòng 6: Nếu t0 ≠ 0 thì chuyển đến else
* Dòng 7: Thực hiện s0 = s1
* Dòng 8: Thực hiện chuyển đến endif
* Dòng 10: Thực hiện s0 = 0 – s1



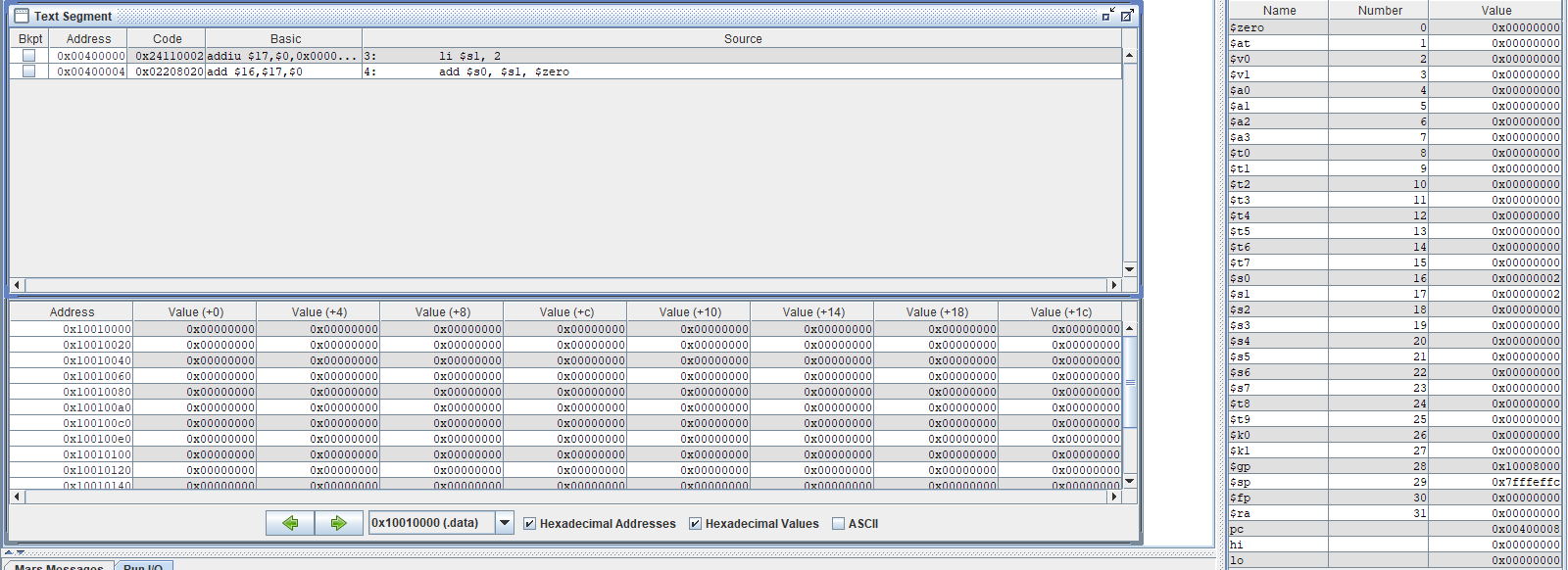
*Thực hiện chạy chương trình với* ***MARS***

1. move



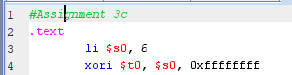
*Thực hiện gõ chương trình vào công cụ* ***MARS***

* Dòng 3: Thanh ghi s1 được gán giá trị 2
* Dòng 4: Thực hiện phép toán s0 = s1



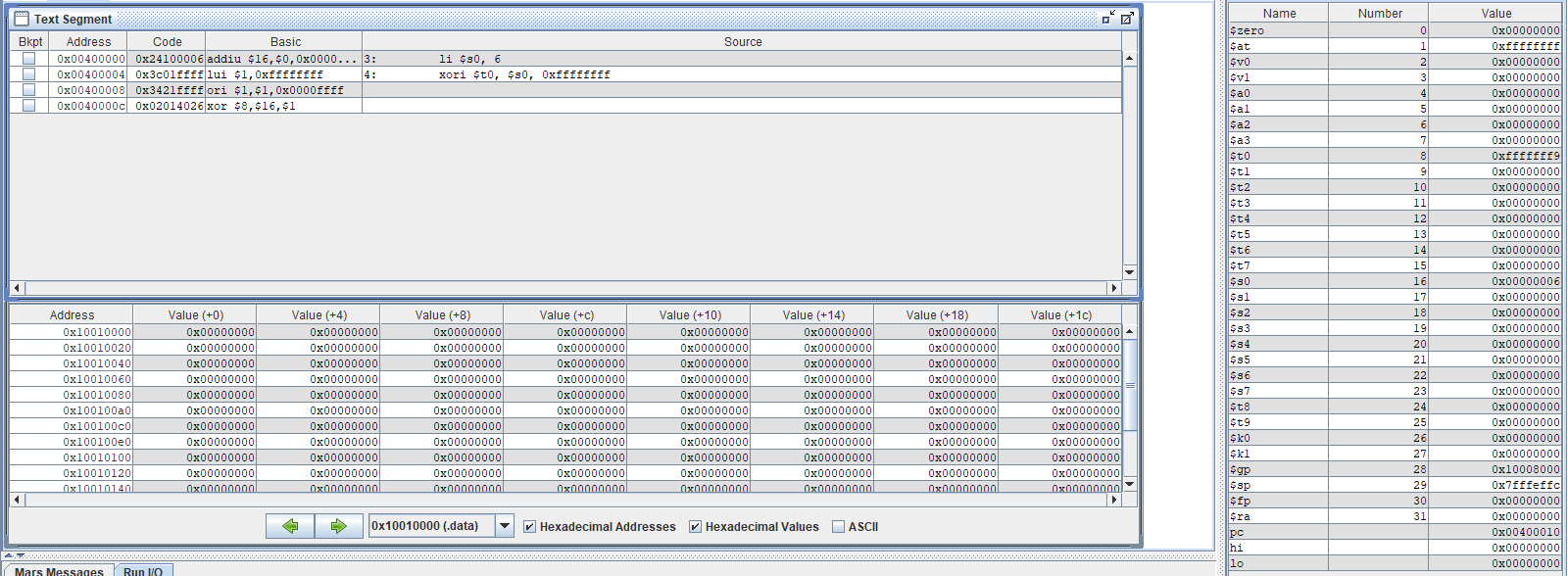
*Thực hiện chạy chương trình với* ***MARS***

1. not



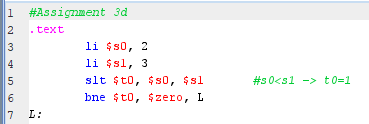
*Thực hiện gõ chương trình vào công cụ* ***MARS***

* Dòng 3: Thanh ghi s0 được gán giá trị 6
* Dòng 4: Thực hiện đảo dấu s0 bằng phép xor với 0xffffffff. Khi đó t0 = 0xfffffff9



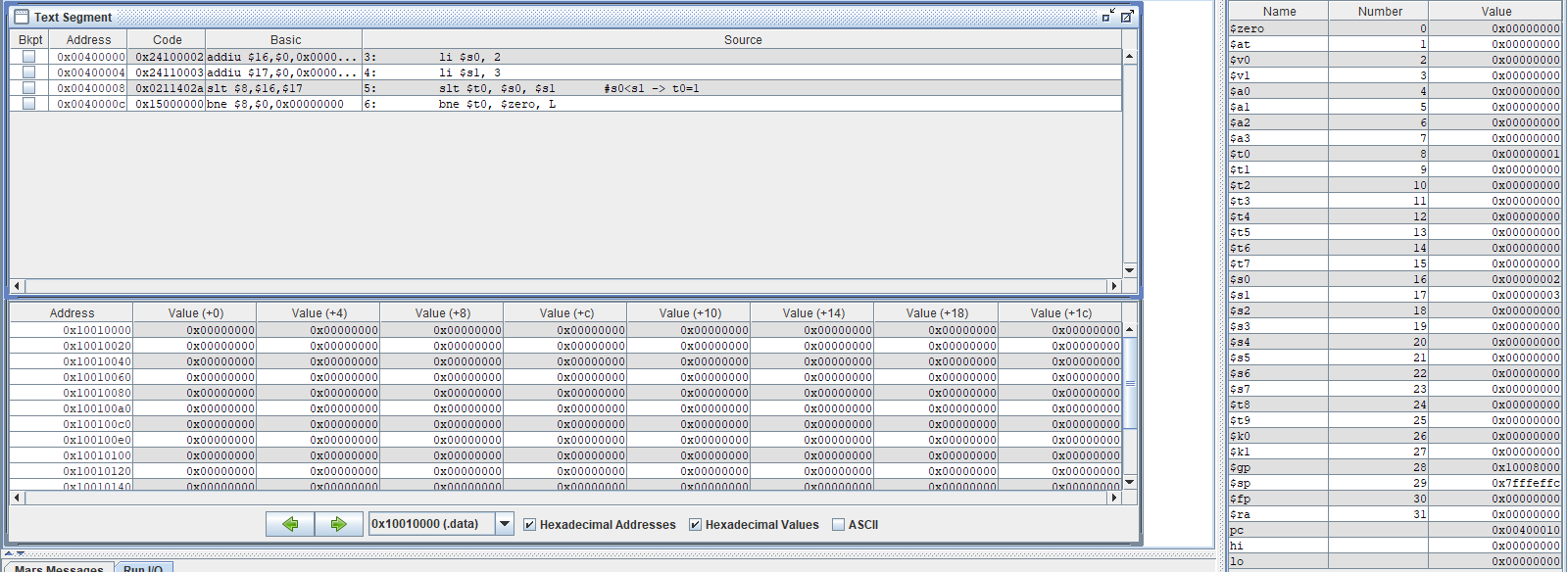
*Thực hiện chạy chương trình với* ***MARS***

1. ble



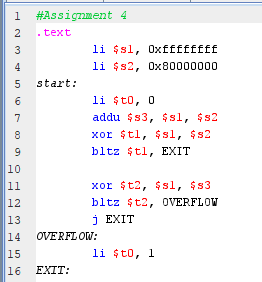
*Thực hiện gõ chương trình vào công cụ* ***MARS***

* Dòng 3, 4: Thanh ghi s0, s1 được gán giá trị lần lượt là 2 và 3
* Dòng 5: So sánh s0 có nhỏ hơn s1 hay không cho kết quả vào t0
* Dòng 6: Nếu t0 ≠ 0 (s0<=s1) thì chuyển sang L



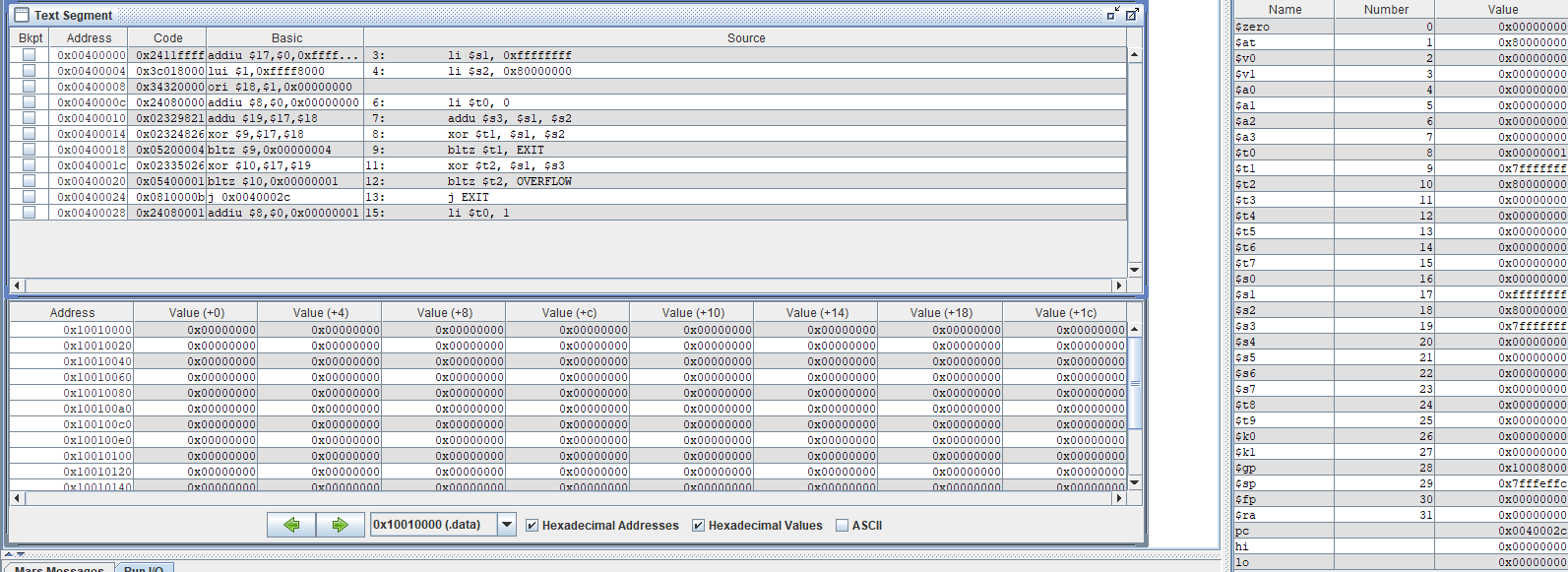
*Thực hiện chạy chương trình với* ***MARS***

Bài 4.



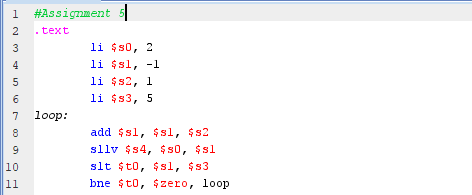
*Thực hiện gõ chương trình vào công cụ* ***MARS***

* Dòng 3, 4: Thanh ghi s1, s2 được gán giá trị lần lượt là 0xffffffff và 0x80000000
* Dòng 6: Thanh ghi t0 được gán giá trị 0 (không tràn số)
* Dòng 7: Thực hiện phép toán s3 = s1 + s2
* Dòng 8: Thực hiện phép toán s1^s2 gán kết quả vào t1
* Dòng 9: Nếu t1<0 (s1, s2 trái dấu) thì EXIT (không tràn số)
* Dòng 11: Thực hiện phép toán s1^s3 gán kết quả vào t2
* Dòng 12: Nếu t2<0 (s1, s3 trái dấu) thì OVERFLOW
* Dòng 13: chuyển đến phần EXIT
* Dòng 15: Thanh ghi t0 được gán giá trị 1 (tràn số)



*Thực hiện chạy chương trình với* ***MARS***

Bài 5.

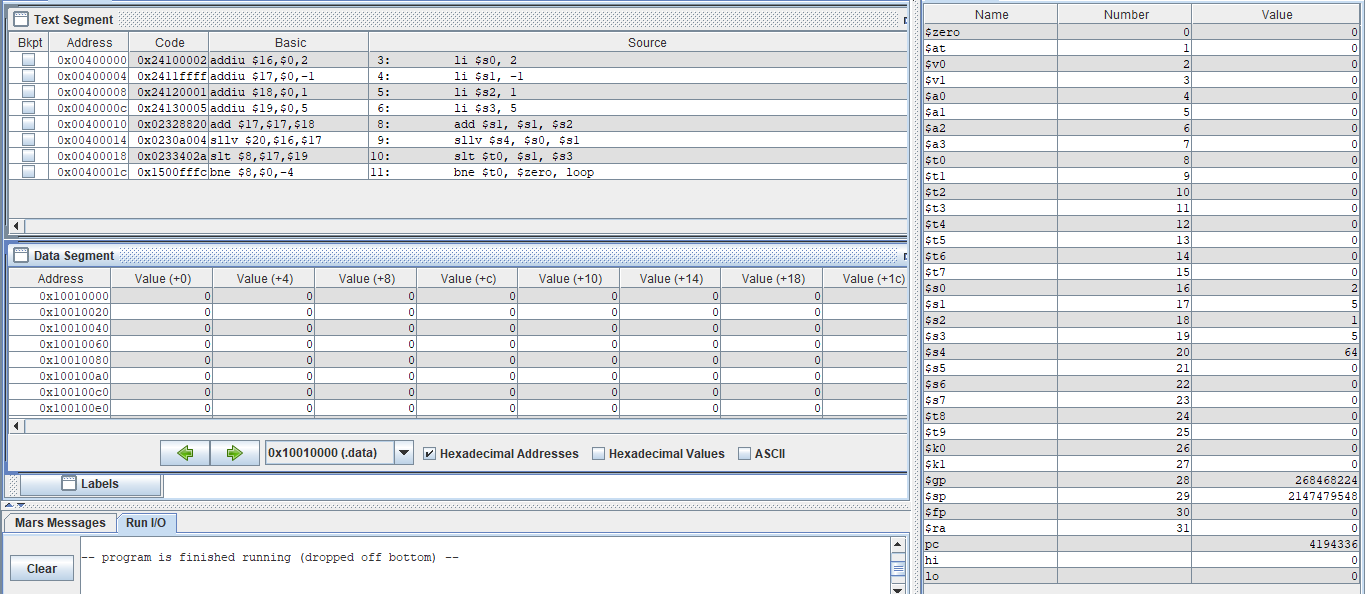


*Thực hiện gõ chương trình vào công cụ* ***MARS***

* Dòng 3, 4, 5, 6: Thanh ghi s0, s1, s2, s3 được gán giá trị lần lượt 2, -1, 1, 10

(s0: giá trị ban đầu, s1 = i = -1, s2 = step = 1, s3 = n = 10)

* Dòng 8: Thực hiện phép toán s1 = s1 + s2 (i = i + step)
* Dòng 9: Thực hiện phép dịch bit sang trái: s4 = s0 \* 2^s1
* Dòng 10, 11: Kiểm tra s1 (i) đã đếm đến s3 (n = 5) hay chưa. Nếu chưa thì tiếp tục vòng lặp



*Thực hiện chạy chương trình với* ***MARS***