Báo Cáo bài thực hành Lab 3 Môn: Tổ chức & Cấu trúc máy tính II

Nguyễn Trường Giang MSSV: 24520011 Nguyễn Ngọc Hưng MSSV: 24520610

Ngày 5 tháng 5 năm 2025

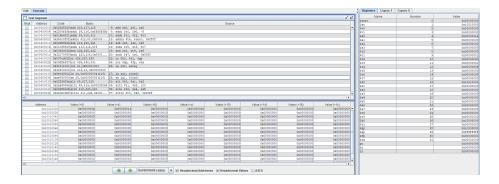
Bài 2.1

Các lệnh MIPS cơ bản và chức năng:

- add: Cộng hai thanh ghi (có dấu).
- addi: Cộng một thanh ghi với một hằng số (có dấu).
- addu: Cộng hai thanh ghi (cộng không dấu).
- addiu: Cộng một thanh ghi với một hằng số (cộng không dấu).
- sub: Trừ 2 thanh ghi (trừ có dấu).
- subu: Trừ 2 thanh ghi (trừ không dấu).
- and: Phép AND logic (and 2 thanh ghi).
- andi: Phép AND logic (and một thanh ghi với một hằng số).
- or: Phép OR logic. (or 2 thanh ghi).
- nor: Phép NOT-OR logic (not-or 2 thanh ghi).
- lw: Load word Tải word (4-byte) từ bộ nhớ.
- sw: Store word Lưu word (4-byte) từ bộ nhớ.
- slt: So sánh nhỏ hơn thao tác có dấu (giữa 2 thanh ghi).
- slti: So sánh nhỏ hơn thao tác có dấu (giữa một thanh ghi và một hằng số).
- sltu: So sánh nhỏ hơn thao tác không dấu (giữa 2 thanh ghi).
- sltiu: So sánh nhỏ hơn thao tác không dấu (giữa một thanh ghi và một hằng số).

```
1 .data
    array: .word 10, 20, 30, 40 # Khai báo mảng trong .data
 2
 3
 4
    main:
 5
    add $s0, $s1, $s2 # s0 = s1 + s2
 6
 8
    addi $t0, $s0, -5 \# t0 = t0 -5
    addu $t1, $t2, $t3 # t1 = t2 + t3 (khong dau)
10
11
    addiu $t4, $zero, 3 # t4 = 0 + 3 (khong dau)
12
13
14
    sub $s3, $s4, $s5 # s3 = s4 - s5
15
16 subu $t5, $t6, $t7 # t5 = t6 - t7 (khong dau)
17
   and $t8, $t9, $s6 # t8 = t9 and s6
18
19
20 andi $s7, $s0, 2 # s7 = s0 + 2
21
22 or $t9, $t8, $t7 #
23
24 nor $s7, $s6, $s5
25
26 la $s0, array # dua di chi array vao s0
   lw $a0, 4($s0) # lay s0[1] (tuc la gia tri 20) vao a0
27
28
29
   sw $a0, 8($s0) # luu a0 vao s0[2] (thay gia tri 30 thanh 20)
30
31 slt $t0, $s1, $s2 # t0 = (s1 < s2)
32
33 slti $t1, $s3, 100 # t1 = (s3 < 100)
34
35
   sltu $t2, $s4, $s5 # t2 = (s4 < s5) (khong dau)
37 sltiu $t3, $s6, 4 # t3 = (s6 < 4) (khong dau)
```

Hình 1: Mẫu các lệnh cơ bản của MIPS



Hình 2: Mô phỏng thực thi các lệnh trong MIPS

Bài 2.2 Chuyển code sang MIPS

```
addi $t0, $zero, 0x10010018

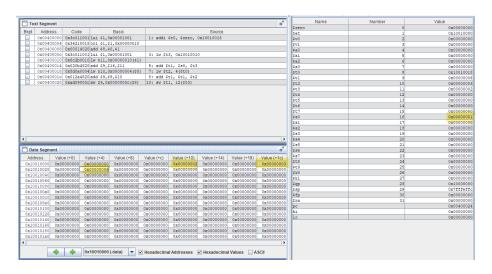
lw $t3, 0x10010010

add $t1, $s0, $t3

lw $t2, 4($t0)
add $t1, $t1, $t2

sw $t1, 12($t0)
```

- dòng **addi** đầu tiên là thay vì phải tự chỉnh tay giá trị t0=0x10010018 thì mình tự cộng giá trị đó vào t0 luôn.
- lw là mình sẽ load word giá trị từ địa chỉ 0x10010010 vào t3, đại diện cho biến C.
- lw ở dòng thứ 4 là mình sẽ lấy giá trị từ B[1] vào t2.
- 2 dòng add là mình sẽ coi t1 là đáp án cần tìm, cộng 3 biến A, C và B[1] vào.
- cuối cùng là sw (store word) đáp án t1 vào B[3].



Hình 3: Kết quả code thao tác mảng

Bài 2.3 Chuyển code điều kiện và vòng lặp

a. Câu lệnh if-else:

```
beq $s0, $s1, then

sub $s2, $t0, $t1

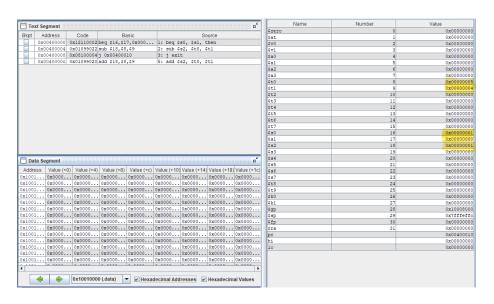
j exit

then:

add $s2, $t0, $t1

exit:
```

• Ý tưởng chính là so sánh i và j xem có bằng nhau không, nếu bằng thì đi xuống nhãn then, chạy lệnh f=g+h, ngược lại thì f=g-h và nhảy xuống exit.



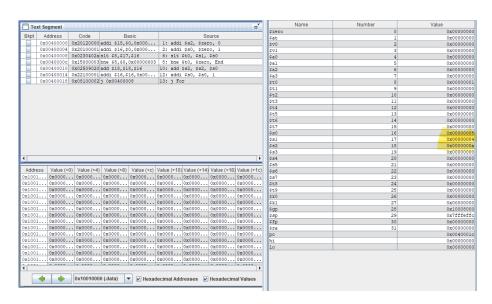
Hình 4: Kết quả code if-else

b. Vòng lặp for:

```
addi $s2, $zero, 0
addi $s0, $zero, 1
For:
slt $t0, $s1, $s0
bne $t0, $zero, End
add $s2, $s2, $s0
```

```
addi $s0, $s0, 1
j For
End:
```

- Đầu tiên là khởi tạo s2 là sum = 0, khởi tạo s0 là i = 1
- tạo biến t0 kiểm tra xem n < i hay không, nếu n < i thì nhảy xuống nhãn End.
- sum = sum + i
- i = i + 1
- cuối cùng là nhảy lại về nhãn For



Hình 5: Kết quả code vòng lặp for