Báo cáo thực hành môn Kiến trúc máy tính

Họ tên: Nguyễn Thanh Hưng

MSSV: 20225633

1. Lab assignment
2. Lab assignment 1

A screenshot of a computer

Description automatically generated

* Ở lệnh 1 sau khi thực hiện lệnh addi $s0, $zero, 0x3007 thì giá trị của thanh ghi $0 thanh đổi từ 0 thành 0x00003007 vì đây là lệnh cộng 2 giá trị của thanh ghi $zero với 0x3007 và kết quả được lưu ở thanh ghi $s0. 

Thanh ghi pc đã được nạp thêm 4 bytes để thực hiện lệnh addi này và việc nạp thêm xảy ra trước lệnh cộng 2 giá trị trên.

\

* Ở lệnh thứ 2 khi thực hiện lệnh add $s0, $zero, $0 thì thanh pc tiếp tục được nạp thêm 4 bytes trước khi thực hiện lệnh add



Lệnh add $s0, $zero, $0 thực hiện việc gán giá trị $s0 bằng tổng 2 giá trị của thanh ghi $zero và giá trị 0 nên giá trị lúc này của $s0 trở thành 0



+) Khi sử dụng lệnh addi $s0, $zero, 0x2110003d thì ở đây giá trị chuẩn bị được thực hiện là 1 giá trị 32 bit vượt quá số bit của thanh ghi nên để cộng giá trị này được chia thành 2 lệnh



Với 2 lệnh này giá trị 32 bit sẽ được nạp vào thanh ghi tạm của trình biên dịch là $at sau 2 lần nạp





Sau khi giá trị này được nạp vào đành đủ thì sẽ thực hiện các bước tương tụ như ở trên.

1. Lab assignments 2

A screenshot of a computer

Description automatically generated

* Ở bài này các lệnh thực hiện tương tự ý 2 câu 1 chỉ có 1 điều thay đổi là ở đây giá trị 32 bit được nạp thẳng vào thanh ghi đích là $s0 là thay vì sử dụng thanh ghi tạm là $at như ở bài trên.



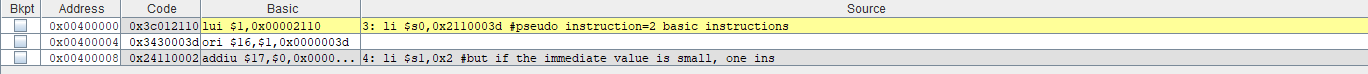


1. Lab assignment 3

A screenshot of a computer

Description automatically generated

* Ở đây lệnh li $s0, 0x2110003d thực hiện việc gán giá trị thanh ghi $s0 = 0x2110003d nhưng đây là 1 giá trị 32 bit nên sẽ bị chia ra.



Khi thực hiện bước đầu tiên thì phần giá trị upper sẽ được lưu trên thanh tạm là thanh ghi $at



Và ở bước tiếp theo khi nạp vào giá trị lower thì nó ngay lập tức được được add với giá trị trước đó được lưu trên $at và gán vào giá trị đích cuối cùng là $s0 mà cần phải lưu qua thanh ghi $at rồi mới gán vào $s0.



* Lệnh li $s1, 0x2 sẽ gán giá trị thanh ghi $s1 = 0x2 và đây là 1 giá trị nhỏ nên sẽ được gán trực tiếp



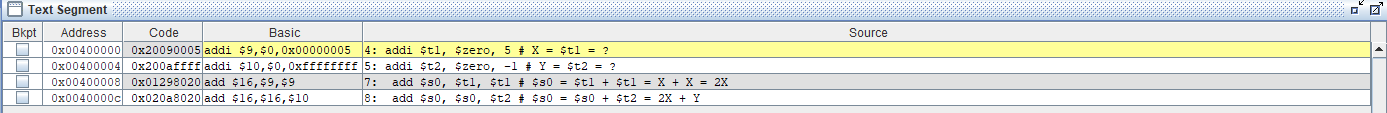
1. Lab assignment 4

A screenshot of a computer

Description automatically generated

A white and black text

Description automatically generated with medium confidence



* Ở đây lệnh addi $t1, $zero, 5 (0x2009005)



• opcode là addi: 6 bit đầu tiên, tương ứng với 0x20 (hoặc 001000 trong hệ nhị phân), là mã lệnh cho addi.

• rs là $zero: 5 bit tiếp theo, tương ứng với 0x0 (hoặc 00000 trong hệ nhị phân), là mã thanh ghi cho $zero.

• rt là $t1: 5 bit tiếp theo, tương ứng với 0x9 (hoặc 01001 trong hệ nhị phân), là mã thanh ghi cho $t1.

• immediate là 5: 16 bit cuối cùng, tương ứng với 0x5 (hoặc 0000000000000101 trong hệ nhị phân), là giá trị ngay lập tức.

Vì vậy, addi tương đồng với khuôn mẫu lệnh kiểu I

* Tương tự như thế với lệnh addi còn lại.
* Sự thay đổi của thanh ghi trong việc tính kết quả phép tính 2x + y sẽ tương tự như các bài trên vì ở đây ta sẽ thực hiện phép tính x + x + y thanh cho việc nhân.

Ở đây lệnh add $s0, $t1, $t1 (0x01298020)(0b00000001001010011000000000100000: Từ hex chuyển qua binary)



• opcode là add: 6 bit đầu tiên, tương ứng với 0x0 (hoặc 000000 trong hệ nhị phân), là mã lệnh cho add.

• rs là $t1: 5 bit tiếp theo, tương ứng với 0x9 (hoặc 01001 trong hệ nhị phân), là mã thanh ghi cho $t1.

• rt là $t1: 5 bit tiếp theo, tương ứng với 0x9 (hoặc 01001 trong hệ nhị phân), là mã thanh ghi cho $t1.

• rd là $s0: 5 bit tiếp theo, tương ứng với 0x10 (hoặc 10000 trong hệ nhị phân), là mã thanh ghi cho $s0

• shamt: 5 bit tiếp theo, tương ứng với 0x0 (hoặc 00000 trong hệ nhị phân)

• func: 6 bit cuối cùng, tương ứng với 0x20 (hoặc 10000 trong hệ nhị phân) là mã lệnh thực hiện lệnh add.

Vì vậy, add tương đồng với khuôn mẫu lệnh kiểu R

1. Lab assignment 5

A screenshot of a computer

Description automatically generated

* Lệnh addi $t1, $zero, 4 và addi $t2, $zero, 5 sẽ thực hiện phép cộng và sau khi thực hiện xong thì 2 thanh $t1 và $t2 sẽ có giá trị lần lượt là 4 và 5



* Khi thực hiện lệnh mul $s0, $t1, $t2 thì giá trị ở thanh ghi Lo ngay lập tức trở thành 0x00000014 (20) và thanh $s0 cũng biểu thị giá trị này bởi nó là thanh ghi đích của phép nhân. Ở đây thanh ghi Hi chỉ biểu thị 0 bởi vì kết quả của phép nhân chưa vượt quá 32 bit.

A screenshot of a spreadsheet

Description automatically generated

* Lệnh mul $s0, $s0, 3 thực hiện sẽ nạp thêm giá trị 3 vào thanh ghi tạm $at và sẽ thực hiện phép nhân tương tự như trên



A table with numbers and a green line

Description automatically generated

1. Lab assignment 6

A close up of a message

Description automatically generated

A screenshot of a computer

Description automatically generated

* Lệnh la $t8, X thì nó sẽ lưu địa chỉ của X (ở đây là địa chỉ của giá trị 5) (0x1001) vào $at bởi câu lệnh lui sau đó mới lưu vào thanh ghi $t8 bởi câu lệnh ori



* Lệnh la $t9, Y nó sẽ lưu địa chỉ của Y (ở đây là địa chỉ của giá trị -1) (0x0004) nhưng mà ở $at giá trị địa chỉ không thay đổi mà thay vào đó nó sẽ lưu giá trị địa chỉ này vào ngay sau địa chỉ của X trước đó và lưu ở $t9



Như vậy ở đây lệnh la có tác dụng load lại các giá trị địa chỉ của các biến và lưu liên tiếp nó với nhau và lưu vào giá trị của 1 thanh ghi

* Lệnh lw có tác dụng nhận ra giá trị địa chỉ của lệnh la lưu trên thanh ghi khác và giải mã ra giá trị của nó rồi lưu vào thanh ghi đích muốn đến.



ở đây lw đã giải mã địa chỉ ở $t8 và lưu vào $t1 giá trị 5 tương tự $t2 được giải mã và lưu giá trị là -1

* 2 lệnh add tiếp theo thực hiện tương tự như các bài trên.