**Computer Architecture Lab Report Week 2**

**Full name: Nguyen Hong Phuc**

**Student ID: 20225659**

Assignment 1: lệnh gán số 16-bit

A screenshot of a computer

Description automatically generated

Sau khi chạy xong lệnh đầu tiên, giá trị của $s0 từ 0x00000000 => 0x00003007

Giá trị thanh ghi pc chuyển từ 0x00400000 => 0x00400004 có nghĩa là thanh ghi pc đã tăng lên 4 bits.

Sau khi chạy xong lệnh thứ 2, giá trị của $s0 từ 0x00003007=>0x00000000

Giá trị thanh pc tăng thêm 4 bits 0x00400004 => 0x00400008.

A computer screen shot of a computer screen

Description automatically generated

- Addi $s0, $zero, 0x3007:

Addi $16, $0, 3007

Opcode: 8 => 001000

Rs: $0 => 00000

Rt: $16 =>10000

3007=> 0011000000000111

0010 0000 0001 0000 0011 0000 0000 0111=>0x20103007

-add add $s0, $zero, $0:

Add $16, $0, $0

Opcode: 0=>000000

Rs: $0 =>00000

Rt: $0 =>00000

Rd: 16=>10000

Shamt: 0=>00000

Fuc: 32=>100000

0000 0000 0000 0000 1000 0000 0010 0000 => 0x00008020

Ta thay lệnh addi thành addi $s0, $zero, 0x2110003d

0x2110003d vượt quá 16 bits nên được chia thành 2 lệnh cơ bản là lui và ori.

Assignment 2: lệnh gán số 32-bit

A screenshot of a computer

Description automatically generated

Sau khi chạy xong lệnh lui, giá trị của $s0 từ 0x00000000 => 0x21100000

Giá trị thanh ghi pc chuyển từ 0x00400000 => 0x00400004 có nghĩa là thanh ghi pc đã tăng lên 4 bits.

Sau khi chạy xong lệnh ori, giá trị của $s0 từ 0x21100000=> 0x2110003d

Giá trị thanh pc tăng thêm 4 bits 0x00400004 => 0x00400008.

A screenshot of a computer

Description automatically generated

Các Byte đầu tiên ở vùng lệnh trùng với cột code ở vùng text segment

**Assignment 3: lệnh gán (giả lệnh)**

**A close-up of a message

Description automatically generated**

A screenshot of a computer

Description automatically generated

Đối với thanh ghi $s0: Giá trị được load vào là 0x2110003d.

Đối với thanh ghi $s1:Giá trị được load vào là 0x2.

Sau khi biên dịch đoạn mã này trong công cụ MARS và quan sát các lệnh mã máy trong cửa sổ Text Segment, bạn có thể thấy rằng câu lệnh "li" đã được chuyển đổi thành các lệnh mã máy tương ứng.

Tuy nhiên, điều bất thường là các lệnh mã máy cho câu lệnh "li $s0,0x2110003d" sẽ không chỉ là một lệnh duy nhất như bạn có thể mong đợi từ một câu lệnh "li" tiêu chuẩn. Thay vào đó, nó sẽ được chuyển thành nhiều lệnh mã máy để thực hiện việc load giá trị 0x2110003d vào thanh ghi $s0.

Điều này xảy ra vì giá trị 0x2110003d vượt quá phạm vi của một immediate value có thể được biểu diễn bằng một lệnh duy nhất trong MIPS assembly. Do đó, trình biên dịch sẽ phải phân rã giá trị này thành các phần nhỏ hơn và sử dụng nhiều lệnh mã máy để load giá trị này vào thanh ghi $s0.

Assignment 4: tính biểu thức 2x + y = ?

Bài làm :

Sau khi chạy lần 1 thì thanh ghi t1 được gán giá trị = 5 (0x00000005)

Sau khi chạy lần 2 thì thanh ghi t2 được gán giá trị = -1 (0xffffffff)

Sau khi chạy lần 3 thì thanh ghi s0 được gán giá trị = t1 + t1 = 10 (0x0000000a)

Sau khi chạy lần 4 thì thanh ghi s0 được gán giá trị = s0 + t2 = 9 (0x00000009)

🡺 Sau khi kết thúc chương trình, kết quả đúng với lí thuyết

🡺 Ở cửa sổ Text Segment, các lệnh addi có điểm tương đồng với hợp ngữ và mã máy

+value được gán theo 4 bit cuối (theo hệ số hexa)

+thanh ghi được để trước giá trị value

Assignment 5: phép nhân

A computer screen with a group of people

Description automatically generated

- Sau khi chạy lần 1 thì thanh ghi t1 được gán giá trị = 4 (0x00000004)

- Sau khi chạy lần 2 thì thanh ghi t2 được gán giá trị = 5 (0x00000005)

- Sau khi chạy lần 3 lệnh mul thì thanh ghi s0 được gán giá trị = t1 \* t2 = 4\*5 ,giá trị s0 nhận giá trị là 20. Giá trị thanh Lo là 20.

- khi chạy dòng lệnh mul $s0, $s0, 3, máy tính tách câu lệnh thành 2 câu lệnh khác

+) addi $1, $0, 0x00000003

+) mul $16, $16, $1

Máy tính sẽ gán 3 cho $at sau đấy mới thực hiện phép nhân. $s0=$s0 \* $at =60, Lo=60

- Sau đấy máy tính thực hiện dòng lệnh mflo $s1 là gán giá trị L0 cho $s1. => $s1 = 60.

Giá trị cuối cùng là 60. Đúng với phép nhân 3\*4\*5.

Assignment 6: tạo biến và truy cập biến

A screenshot of a computer

Description automatically generated

Lệnh la được biên dịch bằng cách tính địa chỉ tuyệt đối của biến và lưu nó vào thanh ghi được chỉ định. Điều này thường được thực hiện bằng cách cộng địa chỉ cơ sở của Data Segment với địa chỉ tương đối của biến. Khi biên dịch lệnh la, các giá trị của X, Y, Z trong Data Segment sẽ được quan sát để đảm bảo rằng địa chỉ được tính toán chính xác và phản ánh vị trí của chúng trong bộ nhớ. Các lệnh lw và sw được sử dụng để đọc và ghi dữ liệu từ và vào bộ nhớ, tương ứng. Lệnh lb và sb được sử dụng để đọc và ghi byte từ và vào bộ nhớ.