Báo cáo môn Thực hành kiến trúc máy tính

Bài thực hành 2

Võ Anh Khôi 20225870

Bài 1

	$s0	$pc
Ban đầu	0x00000000	0x00400000
addi $s0, $zero, $0x3007	0x00003007	0x00400004
add $s0, $zero, $0	0x00000000	0x00400008

Lệnh addi $s0, $zero, $0x3007 (I-type)

Mã máy 0x20103007

Mã nhị phân 0010 00|00 000|1 0000| 0011 0000 0000 0111

Opcode: 001000 (0x08 – addi)

rs: 00000 ($zero/$0)

rt: 10000 ($s0/$16)

imm: 0011 0000 0000 0111 (0x3007)

Lệnh add $s0, $zero, $0

Mã máy 0x00008020

Mã nhị phân 0000 00|00 000|0 0000| 1000 0|000 00|10 0000

Opcode: 000000 (0x00)

rs: 00000 ($zero/$0)

rt: 00000 ($zero/$0)

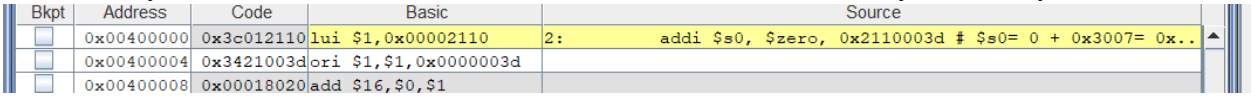
rd: 10000 ($s0/$16)

sh: 00000

fn: 100000 (0x20 - add)

➔Mã máy của các lệnh trên đã đúng với khuôn dạng lệnh như tập lệnh đã quy định

Sau khi thay lệnh addi trên thành addi $s0, $zero, $0x2110003d ta thấy có sự thay đổi như hình

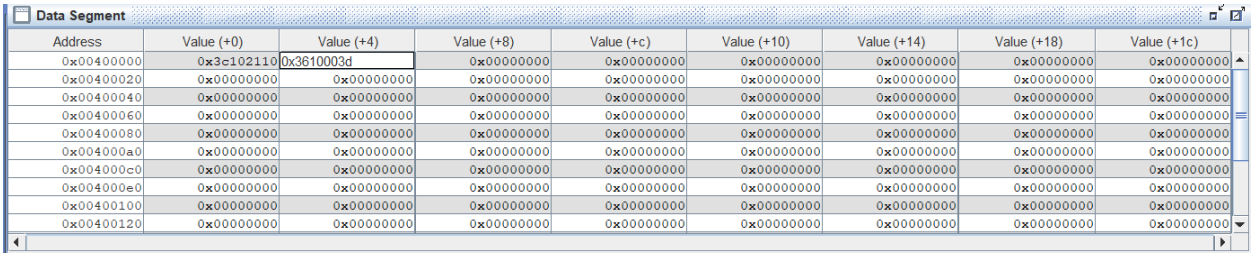


Bkpt	Address	Code	Basic	Source
	0x00400000	0x3c012110	lui $1,0x00002110	2: addi $s0, $zero, 0x2110003d # $s0= 0 + 0x3007= 0x..
	0x00400004	0x3421003d	ori $1,$1,0x0000003d	
	0x00400008	0x00018020	add $16,$0,$1	

Giải thích: Vì 0x2110003d là 32-bit mà nên khi chạy sẽ tách thành việc gán 2 lần số 16-bit với phần bên trái (byte cao) là 0x2110 bằng lệnh lui (ghi vào thanh ghi giá trị bắt đầu từ bit thứ 16) vào thanh ghi $1 và phần bên phải (byte thấp) 0x003d bằng lệnh ori (logical or, nên nếu byte thấp đó bị dính giá trị nào đó có thể dẫn đến việc kết quả không mong muốn nếu dùng với mục đích gán) vào thanh ghi $1, sau đó cộng giá trị $0 (0) và giá trị $1 (0x2110003d) và ghi vào thanh ghi đích là $s0 ($s0 = 0 + 0x2110003d)

Bài 2

	$s0	$pc
Ban đầu	0x0000|0000	0x0040|0000
lui $s0, 0x2110	0x2110|0000	0x0040|0004
ori $s0, $s0, 0x003d	0x2110|003d	0x0040|0008

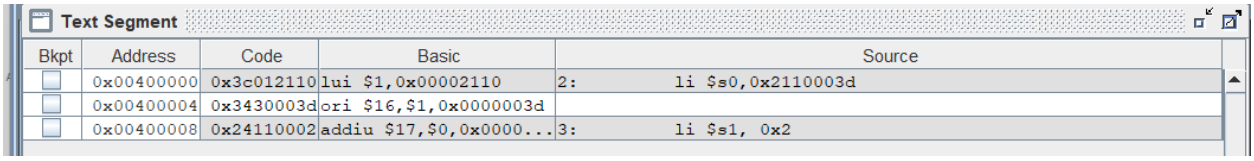


Ở cửa sổ Data Segment:

lệnh lui (0x3c102110) được ghi ở địa chỉ 0x00400000 tương ứng với hàng 2, cột 2 trong bảng

lệnh ori (0x3610003d) được ghi ở địa chỉ 0x00400004 tương ứng với hàng 2, cột 3 trong bảng

Bài 3



Lệnh Li (load immediate) : lệnh này được sử dụng để nạp một giá trị vào thanh ghi ngay lập tức

Vì số 0x2110003d là một số rất lớn nên việc load trực tiếp số này vào thanh ghi có thể dẫn đến những trường hợp không mong muốn, do đó lệnh Li được tách thành 2 lệnh lui và ori như đã giải thích ở bài 1.

Bài 4

	$t1	$t2	$s0	$pc
addi $t1, $zero, 5	0x00000005	0x00000000	0x00000000	0x00400004
addi $t2, $zero, -1	0x00000005	0xffffffff	0x00000000	0x00400008
add $s0, $t1, $t1	0x00000005	0xffffffff	0x0000000a (1)	0x0040000c
add $s0, $s0, $t2	0x00000005	0xffffffff	0x00000009 (2)	0x004000010

ở (1) $s0 = 10 (2*$t1| $t1=5)

ở (2) $s0 = 9 (2*$t1-$t2| $t1 = 5, $t2 = -1)

➔Giá trị kết quả đúng

addi $t1, $zero, 5

Mã máy: 0x20090005

Mã nhị phân: 0010 00|00 000|0 1001| 0000 0000 0000 0101

Opcode: 001000 (0x08 –addi)

rs: 00000 ($zero/$0)

rt: 01001 ($t1/$9)

imm: 0000 0000 0000 0101 (5)

addi $t2, $zero, -1

Mã máy: 0x200affff

Mã nhị phân: 0010 00|00 000|0 1010| 1111 1111 1111 1111

Opcode : 001000 (0x08 –addi)

rs: 00000 ($zero/$0)

rt: 01010 ($t2/$10)

imm: 1111 1111 1111 1111 (-1)

add $s0, $t1, $t1

Mã máy: 0x01298020

Mã nhị phân: 0000 00|01 001|0 1001| 1000 0|000 00|10 0000

Opcode: 000000

rs: 01001 ($t1/$9)

rt: 01001 ($t1/$9)

rd: 10000 ($s0/$16)

sh: 00000

fn: 100000 (0x20 - add)

add $s0, $s0, $t2

Mã máy: 0x020a8020

Mã nhị phân: 0000 00|10 000|0 1010| 1000 0|000 00|10 0000

Opcode: 000000

rs: 10000 ($s0/$16)

rt: 01010 ($t2/$10)

rd: 10000 ($s0/$16)

sh: 00000

fn: 100000 (0x20 - add)

⇨ Đúng với khuôn lệnh I và R

Bài 5 :

addi $t1, $zero, 4 : gán giá trị của thanh ghi $t1 là 4

addi $t2, $zero, 5 gán giá trị của thanh ghi $t2 là 5

Biểu thức Z = 3 * XY:

mul $s0, $t1, $t2: Nhân giá trị của X và Y và lưu kết quả vào $s0. Sản phẩm được tính toán chính xác.

mul $s0, $s0, 3: Tiếp theo, chúng ta nhân kết quả của $s0 với 3 (để được 3 * XY).

Z' = Z:

Assignment 5: phép nhân

-Biên dịch và quan sát các lệnh mã máy trong cửa sổ Text Segment. Giải

thích điều bất thường?

Gán giá trị cho X và Y:

addi $t1, $zero, 4: Lệnh này đặt giá trị của thanh ghi $t1 (X) thành 4.

addi $t2, $zero, 5: Tương tự, điều này đặt giá trị của thanh ghi $t2 (Y) thành 5.

Biểu thức Z = 3 * XY:

mul $s0, $t1, $t2: Nhân giá trị của X và Y và lưu kết quả vào $s0. Tính toán chính xác.

mul $s0, $s0, 3: nhân kết quả của $s0 với 3 (để được 3 * XY).

Z' = Z:

mflo $s1: Lệnh này di chuyển giá trị từ phần thấp nhất của kết quả nhân (được lưu trong $s0) sang $s1.

Vì kết quả phép nhân sẽ là **64bit** nên kết quả của phép nhân sẽ gồm **2 phần hi và lo**

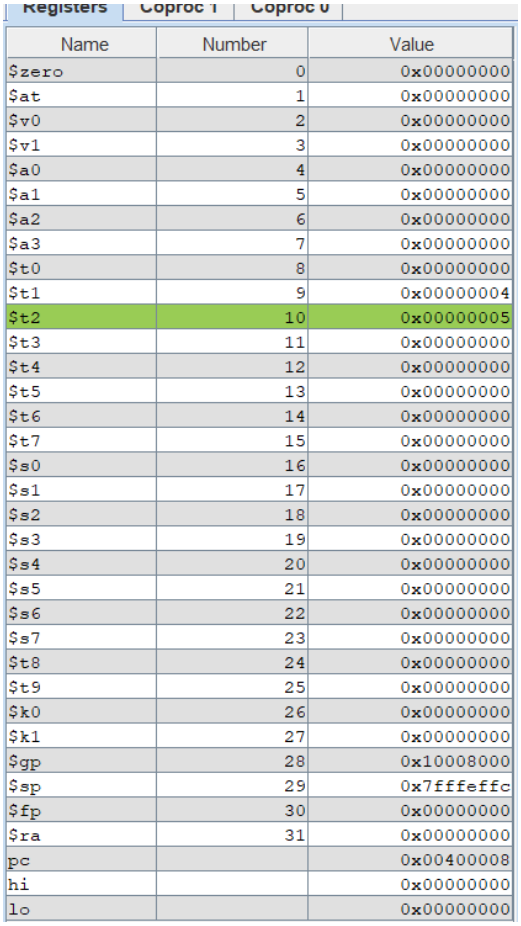
addi $t1, $zero, 4

Registers | Coproc 1 | Coproc 0

Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x00000004
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x00000000
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x00400004
hi		0x00000000
lo		0x00000000

Thanh ghi $t1 có giá trị 0x00000004

addi $t2, $zero, 5



Registers | Coproc 1 | Coproc 0

Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x00000004
$t2	10	0x00000005
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x00000000
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x00400008
hi		0x00000000
lo		0x00000000

Thanh ghi $t2 có giá trị 0x00000005

mul $s0, $t1, $t2

Registers | Coproc 1 | Coproc 0

Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x00000004
$t2	10	0x00000005
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000014
$s1	17	0x00000000
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x0040000c
hi		0x00000000
lo		0x00000014

Thanh ghi $s0 và lo có giá trị 0x00000014

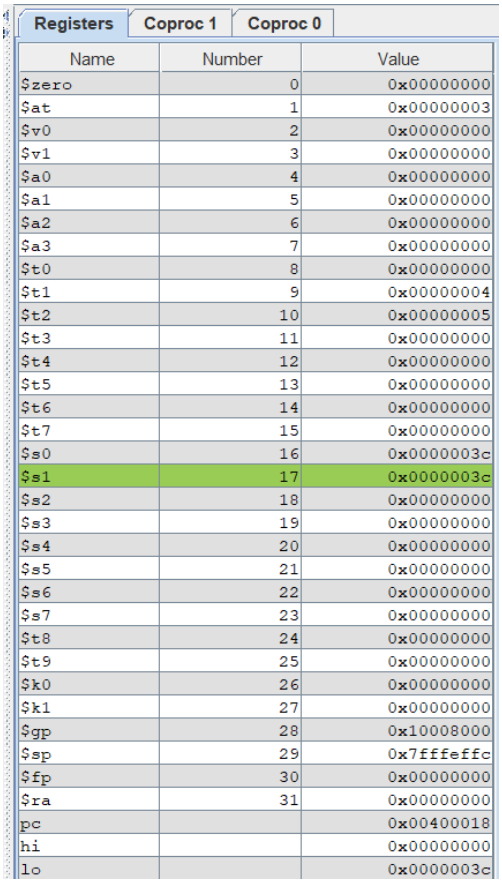
mul $s0, $s0, 3

Registers | Coproc 1 | Coproc 0

Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000003
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x00000004
$t2	10	0x00000005
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x0000003c
$s1	17	0x00000000
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x00400014
hi		0x00000000
lo		0x0000003c

Thanh ghi $s0 và lo có giá trị 0x0000003c

mflo $s1



Registers | Coproc 1 | Coproc 0

Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000003
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x00000004
$t2	10	0x00000005
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x0000003c
$s1	17	0x0000003c
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x00400018
hi		0x00000000
lo		0x0000003c

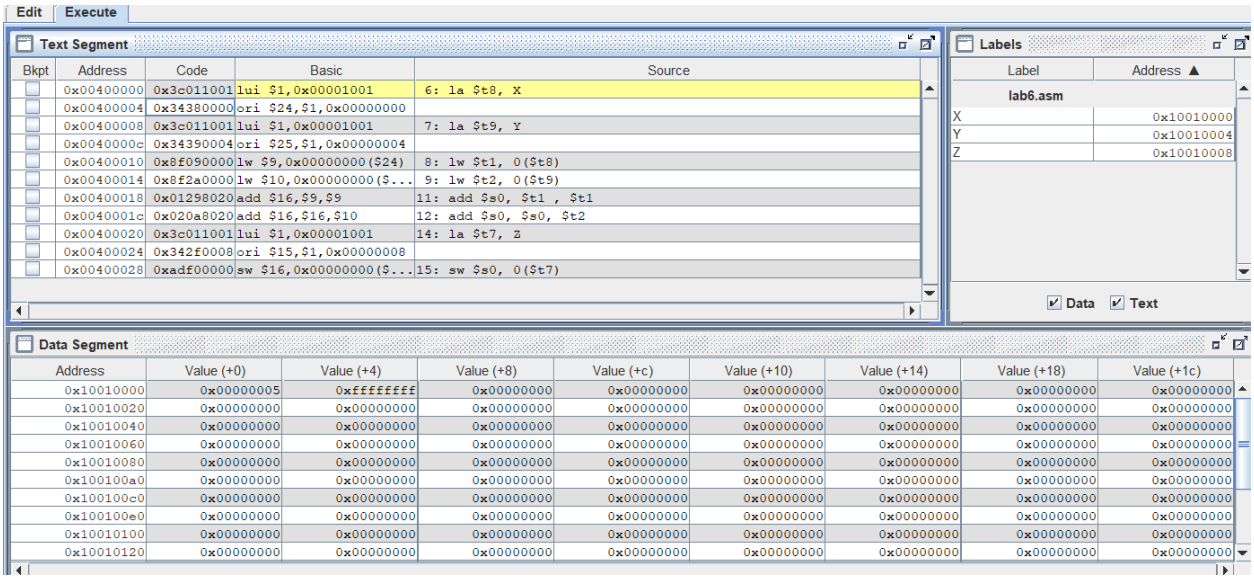
Thanh ghi $s1 có giá trị 0x0000003c

Kết quả cho được có giá trị bằng (3*16+12 =) 60. Đúng với kết quả phép nhân

Bài 6

Lệnh la được biên dịch thành 2 lệnh lui và ori như đã giải thích ở các bài trên từ đó có thể gán địa chỉ của X và Y vào thanh ghi $t8 và $t9

-Khi biên dịch lệnh la địa chỉ của X.Y,Z: 16 bit đầu của địa chỉ là imm của lệnh lui, và 16 bit sau của địa chỉ là imm của ori;



Sau khi so sánh trong bảng labels và Data Segment, các giá trị đúng như giá trị đã khởi tạo.

1. lw $t1, 0($t8):

Lệnh này tải giá trị của biến X từ bộ nhớ có địa chỉ được lưu trong thanh ghi $t8.

Sau khi thực hiện lệnh này, giá trị của X được ghi vào thanh ghi $t1.

2. lw $t2, 0($t9):

Lệnh này tải giá trị của biến Y từ bộ nhớ có địa chỉ được lưu trong thanh ghi $t9.

Sau khi thực hiện lệnh này, giá trị của Y được ghi vào thanh ghi $t2.

3. add $s0, $t1, $t1 và add $s0, $s0, $t2:

Hai lệnh này dùng để thực hiện phép tính Z = 2X + Y.

Giá trị của X nhân 2 và cộng giá trị của Y để tính toán giá trị cần tìm.Sau đó kết quả được lưu trong thanh ghi $s0.

4. sw $s0, 0($t7):

Lệnh này lưu giá trị của Z từ thanh ghi $s0 vào vị trí bộ nhớ có địa chỉ được lưu trong thanh ghi $t7. Giá trị của Z được cập nhật trong bộ nhớ

.

Lệnh lb và sb

Tương tự như với lw và sw nhưng thay vì 4 byte thì sẽ là 1 byte

Lệnh lb:

Tải 1 byte từ trong bộ nhớ vào thanh ghi

Syntax: lb rt, rs trong đó rt là thanh ghi đích và rs là một dịa chỉ nhãn hoặc địa chỉ dịch chuyển cơ sở.

Lệnh sb

Lưu 1 byte từ trong thanh ghi vào bộ nhớ

Syntax: lb Rx, ADDR trong đó Rx là thanh ghi mang giá trị cần lưy và ADDR là một dịa chỉ nhãn hoặc địa chỉ dịch chuyển cơ sở của biến cần lưu giá trị vào