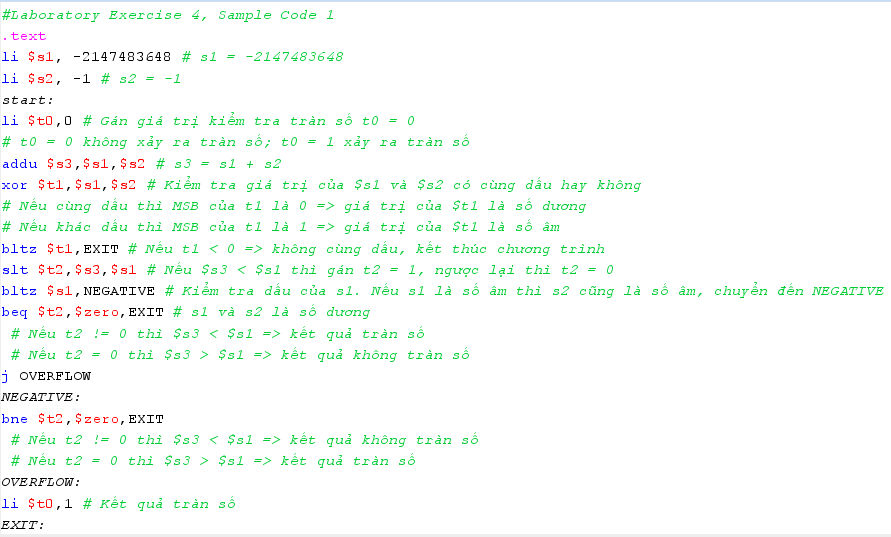
Họ và tên: Trần Quang Nam Trường Đại Học Bách Khoa Hà Nội

MSSV: 20184161 Viện CNTT&TT

**BẢN BÁO CÁO THỰC HÀNH**

**-----TUẦN 04-----**

**Bài 1.**

****

MSB: Most Significant Bit

1. TH1: s1 = -2147483648, s2 = -1 => Kết quả tràn số, t0 = 1

Quá trình thực hiện chương trình và sự thay đổi giá trị của các thanh ghi:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | $s1 | $s2 | $s3 | $t0 | $t1 | $t2 |
| li $s1, -2147483648 | **0x80000000** |  |  |  |  |  |
| li $s2, -1 |  | **0xffffffff** |  |  |  |  |
| li $t0,0 |  |  |  | **0x00000000** |  |  |
| addu $s3,$s1,$s2 |  |  | **0x7fffffff** |  |  |  |
| xor $t1,$s1,$s2 |  |  |  |  | **0x7fffffff** |  |
| bltz $t1,EXIT |  |  |  |  |  |  |
| slt $t2,$s3,$s1 |  |  |  |  |  | **0x00000000** |
| bltz $s1,NEGATIVE |  |  |  |  |  |  |
| bne $t2,$zero,EXIT |  |  |  |  |  |  |
| li $t0,1 |  |  |  | **0x00000001** |  |  |

1. TH2: s1 = 55, s2 = 68 => Kết quả không tràn số, t0 = 0

Quá trình thực hiện chương trình và sự thay đổi giá trị của các thanh ghi:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | $s1 | $s2 | $s3 | $t0 | $t1 | $t2 |
| li $s1, -2147483648 | **0x00000037** |  |  |  |  |  |
| li $s2, -1 |  | **0x00000044** |  |  |  |  |
| li $t0,0 |  |  |  | **0x00000000** |  |  |
| addu $s3,$s1,$s2 |  |  | **0x0000007b** |  |  |  |
| xor $t1,$s1,$s2 |  |  |  |  | **0x00000073** |  |
| bltz $t1,EXIT |  |  |  |  |  |  |
| slt $t2,$s3,$s1 |  |  |  |  |  | **0x00000000** |
| bltz $s1,NEGATIVE |  |  |  |  |  |  |
| beq $t2,$zero,EXIT |  |  |  |  |  |  |

1. TH3: s1 = 5, s2 = -6 => Khác dấu, không kiểm tra trong trường hợp này

Quá trình thực hiện chương trình và sự thay đổi giá trị của các thanh ghi:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | $s1 | $s2 | $s3 | $t0 | $t1 | $t2 |
| li $s1, 5 | **0x00000005** |  |  |  |  |  |
| li $s2, -6 |  | **0xfffffffa** |  |  |  |  |
| li $t0,0 |  |  |  | **0x00000000** |  |  |
| addu $s3,$s1,$s2 |  |  | **0xffffffff** |  |  |  |
| xor $t1,$s1,$s2 |  |  |  |  | **0xffffffff** |  |
| bltz $t1,EXIT |  |  |  |  |  |  |

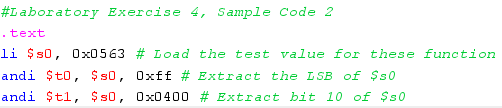
**Bài 2.**

Phép toán thao tác bit AND có thể được dùng để xác định xem bit được chọn có giá trị là 1 hay 0. Bằng cách thực hiện phép AND giữa bit được chọn với bit 1, nếu kết quả trả về là 1 thì bit đó là 1, ngược lại thì bit đó là 0.

Từ đó mở rộng ra, ta có thể kiểm tra với dãy bit. Để xác định xem bit n trong dãy bit là 1 hay 0, ta thực hiện phép toán thao tác bit AND với một dãy bit mà chỉ chứa số 1 duy nhất ở bit n.

Bảng chân trị cho AND:

|  |  |  |
| --- | --- | --- |
| A | B | A AND B |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

****

* li $s0, 0x0563: tạo giá trị kiểm tra s0 = 0x0563
* andi $t0, $s0, 0xff

- Thực hiện phép toán thao tác bit AND với 2 toán hạng s0 và 0xff

s0 = 010101100011(2)

0xff = 000011111111(2)

- Giá trị trả về là LSB của s0 được lưu vào thanh ghi $t0

t0 = 000001100011(2)

* andi $t1, $s0, 0x0400

- Thực hiện phép toán thao tác bit AND với 2 toán hạng s0 và 0x0400

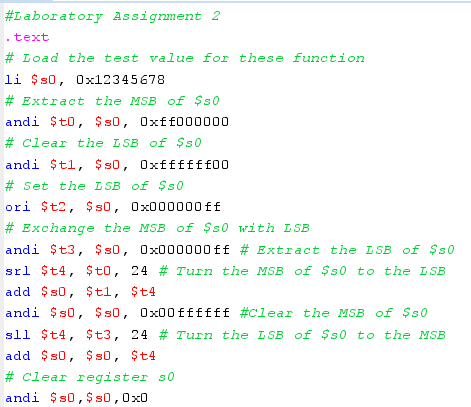
s0 = 010101100011(2)

0x0400 = 010000000000(2)

- Giá trị trả về được lưu vào thanh ghi $t1

t1 = 010000000000(2)

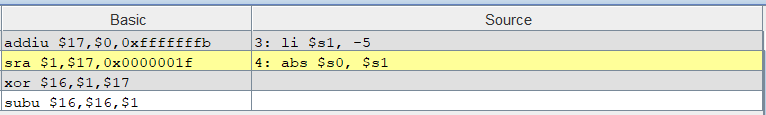
Bit 10 của t1 chính là bit 10 của s0



**Bài 3.**

1. abs $s0, $s1

Ví dụ: s1 = -5 => s0 = |s1| = 5



* sra $1, $17, 0x0000001f

Dịch $s1 sang phải 31 bit, lưu giá trị mới nhận được vào $at, các bit thừa ra được điền bằng bit dấu.

Câu lệnh này được dùng để lấy dấu của $s1. Nếu $s1 dương thì $at = 0, nếu $s1 âm thì $at = 1.

Trong VD này $s1 = -5 nên toàn bộ các bit thừa ra được điền 1.

$at = 0xffffffff

* xor $16, $1, $17

Thực hiện phép toán thao tác bit $s1 XOR $at, lưu giá trị nhận được vào $s0.

Câu lệnh này được dùng để đảo ngược tất cả các bit của $s1 nếu $s1 âm ($at = 0xffffffff), hoặc giữ nguyên giá trị các bit nếu $s1 dương ($at = 0x0).

Lưu ý: Số nhị phân A đảo ngược tất cả các bit được giá trị mới là B, khi biểu diễn thông qua mã bù 2 thì B = (-A) – 1

Trong VD này $s1 = -5 = 1011(2) nên các bit bị đảo ngược và lưu giá trị mới vào $s0

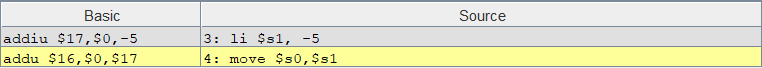
$s0 = 0100(2) = 0x00000004

* subu $16, $16, $1

Thực hiện phép trừ $s0 = $s0 - $at. Lúc này ta có s0 = |s1|

1. move $s0, $s1

Ví dụ: s1 = -5 => s0 = s1 = -5



* addu $16, $0, $17

Thực hiện phép cộng $s0 = $s1 + $zero = -5 + 0 = -5

Câu lệnh này được dùng để gán giá trị của $s0 bằng giá trị của $s1

1. not $s1, $s0

Ví dụ: s1 = -5, s0 = not(s1) = 4



* nor $16, $17, $0

Thực hiện phép toán thao tác bit $s1 NOR $at, lưu giá trị nhận được vào $s0.

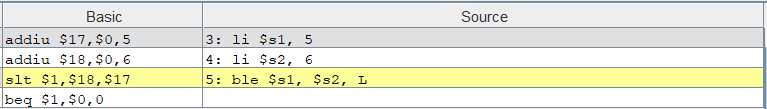
Phép toán thao tác bit NOR có thể được dùng để đảo bit của 1 số nhị phân. Bằng cách thực hiện phép NOR giữa bit được chọn với bit 0, giá trị của bit đó sẽ bị đảo ngược (từ 1 thành 0 hoặc từ 0 thành 1)

Bảng chân trị cho NOR:

|  |  |  |
| --- | --- | --- |
| A | B | A NOR B |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

Trong VD này, s1 = -5 = 1011(2) => $s1 NOR $zero có kết quả là 0100(2). Vậy s0 = 0100(2) = 4

1. ble $s1,s2,L



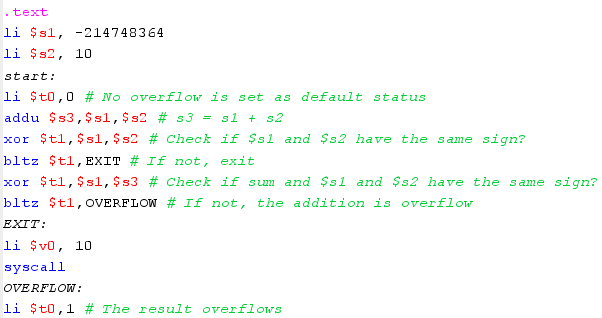
* slt $1, $18, $17

So sánh $s2 và $s1. Nếu $s2 < $s1 thì gán $at = 1, ngược lại nếu $s2 >= $s1 thì $at = 0.

* beq $1, $0, 0

So sánh $at và $zero. Nếu $at = 0 thì rẽ nhánh đến nhãn L và thực hiện lệnh.

**Bài 4.**

****

**Bài 5.**

Assignment5.asm