Báo cáo thực hành KTMT tuần 4

Phạm Thành Lập – 20215076

Assignment 1

- **$s1 âm, $s2 âm và $s3 tràn giá trị**

```
 .text
start:
        li      $s1, 0x9fffffff
        li      $s2, 0x9fffffff
        li      $t0, 0
        addu    $s3, $s1, $s2
        xor     $t1, $s1, $s2
        bltz    $t1, EXIT
        slt     $t2, $s3, $s1
        bltz    $s1, NEGATIVE
        beq     $t2, $zero, EXIT
        j       OVERFLOW
NEGATIVE:
        bne     $t2, $zero, EXIT
OVERFLOW:
        li      $t0, 1
EXIT:
```

$zero	0	0x00000000
$at	1	0x9fff0000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0x00000000
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x9fffffff
$s2	18	0x9fffffff
$s3	19	0x3ffffffe
$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		0x00400038
hi		0x00000000
lo		0x00000000

- Khởi tạo $t0 = 0; $s1 = 0x9fffffff; $s2 = 0x9fffffff
- Lệnh addu giúp có thể lưu giá trị khi kết quả phép cộng nằm ngoài vùng giá trị của thanh ghi, sau lệnh addu ta thực hiện lệnh xor 2 thanh ghi $s1 và $s2 để biết 2 thanh ghi này cùng dấu hay không, nếu cùng dấu dương hoặc cùng âm thì sẽ trả về giá trị thanh ghi $t1 là âm ngược lại sẽ trả về giá trị dương, nếu 2 thanh ghi này khác dấu thì lệnh btlz sẽ nhảy về thẻ EXIT kết thúc chương trình và trả về kết quả thanh ghi $t0 là 0 (giá trị tổng không bị tràn) ngược lại chương trình sẽ so sánh giá trị thanh ghi $s3 và $1 hay không nếu $s1 mang dấu âm thì lệnh bltz thứ 2 sẽ nhảy đến thé NEGATIVE và so sánh thẻ $t2 với 0 nếu $s3 tràn giá trị thì nó sẽ lớn hơn $s1 nếu $s3 bé hơn thì không tràn và nhảy đến thẻ EXIT còn nếu $s1 dương mà $t2 = 0 ($s3 > $s1) thì sẽ nhảy đến thẻ EXIT còn ngược lại thì sẽ nhảy đến thẻ OVERFLOW và thiết lập lại thanh ghi $t0 = 1 từ đó kết luật được rằng tổng 2 toán hạng bị tràn giá trị.
- Sự thay đổi giá trị thanh ghi

Trạng thái	$s3	$t0	$t1	$t2	pc
Ban đầu	0x00000000	0x00000000	0x00000000	0x00000000	0x00400000
Sau lệnh addu	0x3ffffffe	-	-	-	0x00400018
Sau lệnh xor	-	-	-	-	0x0040001c
Sau lệnh slt	-	-	-	-	0x00400020
Sau lệnh btlz nhảy đến thẻ NEGATIVE	-	-	-	-	0x0040002c (địa chỉ của thẻ NEGATIVE)
Sau lệnh li	-	0x00000001	-	-	0x00400038

- **$s1 âm, $s2 âm , $s3 không tràn giá trị**

```
 .text
start:
        li      $s1, 0x9fffffff
        li      $s2, 0xffffffff
        li      $t0, 0
        addu    $s3, $s1, $s2
        xor     $t1, $s1, $s2
        bltz    $t1, EXIT
        slt     $t2, $s3, $s1
        bltz    $s1, NEGATIVE
        beq     $t2, $zero, EXIT
        j       OVERFLOW
NEGATIVE:
        bne     $t2, $zero, EXIT
OVERFLOW:
        li      $t0, 1
EXIT:
```

Name	Number	Value
$zero	0	0x00000000
$at	1	0x9fff0000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x60000000
$t2	10	0x00000001
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x9fffffff
$s2	18	0xffffffff
$s3	19	0x9ffffffe
$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		0x00400034
hi		0x00000000
lo		0x00000000

- Khởi tạo $s1 = 0x9fffffff; $s2 = 0xffffffff
- Có $s1+$s2 không tràn giá trị thanh ghi do $s1 và $s2 cùng dấu âm nên giá trị thanh ghi $s3 sẽ nhỏ hơn $s1 khiến cho giá trị thanh ghi $t2 = 1 sau lệnh btlz thứ 2 chương trình lập tức nhảy đến thẻ NEGATIVE so sánh giá trị $t2 với 0 và nhảy đến thẻ EXIT kết thúc chương trình và giả về giá trị $t0 = 0 tức là không bị tràn giá trị thanh ghi
- Sự thay đổi giá trị thanh ghi

Trạng thái	$s3	$t0	$t1	$t2	pc
Ban đầu	0x00000000	0x00000000	0x00000000	0x00000000	0x00400000
Sau lệnh addu	0x9ffffffe	-	-	-	0x00400018
Sau lệnh xor	-	-	0x60000000	-	0x0040001c
Sau lệnh slt	-	-	-	0x00000001	0x00400020
Sau lệnh btlz nhảy đến thẻ NEGATIVE	-	-	-	-	0x0040002c (địa chỉ của thẻ NEGATIVE)
Sau lệnh bne	-	-	-	-	0x00400034

- **$s1 dương, $s2 dương và $s3 tràn giá trị**

```
.text
start:
        li      $s1, 0x7fffffff
        li      $s2, 0x456
        li      $t0, 0
        addu    $s3, $s1, $s2
        xor     $t1, $s1, $s2
        bltz    $t1, EXIT
        slt     $t2, $s3, $s1
        bltz    $s1, NEGATIVE
        beq     $t2, $zero, EXIT
        j       OVERFLOW
NEGATIVE:
        bne     $t2, $zero, EXIT
OVERFLOW:
        li      $t0, 1
EXIT:
```

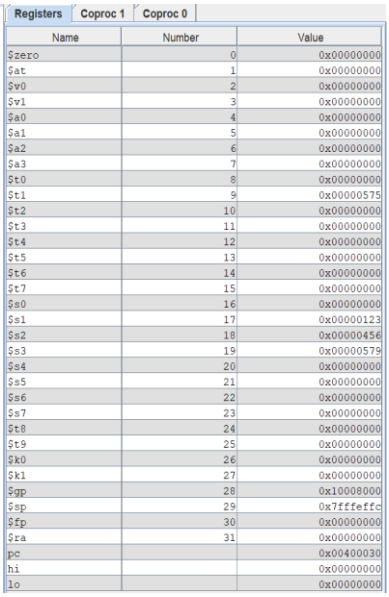
Name	Number	Value
$zero	0	0x00000000
$at	1	0x7fff0000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0x7ffffba9
$t2	10	0x00000001
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x7fffffff
$s2	18	0x00000456
$s3	19	0x80000455
$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		0x00400034
hi		0x00000000
lo		0x00000000

- Khởi tạo $s1 = 0x7fffffff; $s2 = 0x456
- Có $s1+$s2 tràn giá trị thanh ghi $s1 và $s2 cùng dấu nên khi chương trình thực hiện lệnh xor sẽ trả về $t1 giá trị dương và do $s1 dương nên chương trình sẽ so sánh $s3 và $s1 do $s3 bị tràn nên giá trị sẽ bé hơn $s1 nên chương trình lập tức nhảy đến thẻ OVERFLOW để thiết lập giá trị $t0 = 1 và kết luận tổng 2 toán hạng bị tràn giá trị
- Sự thay đổi giá trị thanh ghi

Trạng thái	$s3	$t0	$t1	$t2	pc
Ban đầu	0x00000000	0x00000000	0x00000000	0x00000000	0x00400000
Sau lệnh addu	0x80000455	-	-	-	0x00400018
Sau lệnh xor	-	0x7ffffba9	-	-	0x0040001c
Sau lệnh slt	-	-	-	0x00000001	0x00400020
Sau lệnh j OVERFLOW	-	-	-	-	0x00400030 (địa chỉ của thẻ OVERFLOW)
Sau lệnh li	-	0x00000001	-	-	0x00400034

- **$s1 dương, $s2 dương và $s3 không tràn giá trị**

```
 .text
start:
        li      $s1, 0x123
        li      $s2, 0x456
        li      $t0, 0
        addu    $s3, $s1, $s2
        xor     $t1, $s1, $s2
        bltz    $t1, EXIT
        slt     $t2, $s3, $s1
        bltz    $s1, NEGATIVE
        beq     $t2, $zero, EXIT
        j       OVERFLOW
NEGATIVE:
        bne     $t2, $zero, EXIT
OVERFLOW:
        li      $t0, 1
EXIT:
```



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	0x00000575
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x00000123
$s2	18	0x00000456
$s3	19	0x00000579
$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		0x00400030
hi		0x00000000
lo		0x00000000

- Khởi tạo $s1 = 0x123; $s2 = 0x456
- Có $s1+$s2 không tràn giá trị thanh ghi $s1 và $s2 cùng dấu nên khi chương trình thực hiện lệnh xor sẽ trả về $t1 giá trị âm và do $s1 dương nên chương trình sẽ so sánh $s3 và $s1 do $s3 không bị tràn nên giá trị sẽ lớn hơn $s1 nên chương trình lập tức nhảy đến thẻ EXIT do đó giá trị $t0 = 0 và kết luận tổng 2 toán hạng không bị tràn giá trị
- Sự thay đổi giá trị thanh ghi

Trạng thái	$s3	$t0	$t1	$t2	pc
Ban đầu	0x00000000	0x00000000	0x00000000	0x00000000	0x00400000
Sau lệnh addu	0x00000579	-	-	-	0x00400018
Sau lệnh xor	-	0x00000575	-	-	0x0040001c
Sau lệnh slt	-	-	-	-	0x00400020
Sau lệnh beq nhảy đến thẻ EXIT	-	-	-	-	0x00400030

- **$s1 âm, $s2 dương**

```
 .text
start:
        li      $s1, 0x9fffffff
        li      $s2, 0x569
        li      $t0, 0
        addu    $s3, $s1, $s2
        xor     $t1, $s1, $s2
        bltz    $t1, EXIT
        slt     $t2, $s3, $s1
        bltz    $s1, NEGATIVE
        beq     $t2, $zero, EXIT
        j       OVERFLOW
NEGATIVE:
        bne     $t2, $zero, EXIT
OVERFLOW:
        li      $t0, 1
EXIT:
```

$zero	0	0x00000000
$at	1	0x9fff0000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x9ffffa96
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x9fffffff
$s2	18	0x00000569
$s3	19	0xa0000568
$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		0x00400034
hi		0x00000000
lo		0x00000000

- Khởi tạo $s1 = 0x9fffffff; $s2 = 0x569.
- Có $s1+$s2 không tràn giá trị thanh ghi mà $s1 và $s2 trái dấu nên khi chương trình thực hiện lệnh xor sẽ trả về giá trị âm lên lập tức nhảy đến thẻ EXIT kết thúc chương trình và giả về giá trị $t0 = 0 tức là không bị tràn giá trị thanh ghi
- Sự thay đổi giá trị thanh ghi

Trạng thái	$s3	$t0	$t1	$t2	pc
Ban đầu	0x00000000	0x00000000	0x00000000	0x00000000	0x00400000
Sau lệnh addu	0xa0000568	-	-	-	0x00400018
Sau lệnh xor	-	-	0x9ffffa96	-	0x0040001c
Sau lệnh slt	-	-	-	-	0x00400020
Sau lệnh beq nhảy đến thẻ EXIT	-	-	-	-	0x00400034 (địa chỉ của thẻ EXIT)

Assignment 2

- Extract MSB of $s0

```
.text
        li      $s0, 0x12345678
        andi    $t0, $s0, 0xff000000
```

$t0	8	0x12000000
$t1	9	0x00000000
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x12345678

- Clear LSB of $s0

```
.text
        li      $s0, 0x12345678
        andi    $t1, $s0, 0xffffff00
```

$t1	9	0x12345600
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x12345678

- Set LSB of $s0 (bits 7 to 0 are set to 1)

```
.text
        li      $s0, 0x12345678
        ori     $t2, $s0, 0x000000ff
```

$t2	10	0x123456ff
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x12345678

- Clear $s0 ($s0 = 0, must use logical instructions)

```
.text
        li      $s0, 0x12345678
        and     $s0, $s0, $zero
```

$zero	0	0x00000000
$at	1	0x12340000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x00000000
$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		0x0040000c
hi		0x00000000
lo		0x00000000

Assignment 3

a) abs $s0, $s1

```
.text
        slt     $t0, $s1, $zero
        beqz    $t0, ABS
        sub     $s1, $zero, $s1
ABS:
        add     $s0, $s1, $zero
```

b) move $s0, $s1

```
.text
        add     $s0, $s1, $zero
```

c) not $s0, $s1

```
.text
```

```
        sub     $t0, $zero, $s1
        li      $t1, 1
        sub     $s0, $t0, $t1
```

d) ble $s1,$s2,label

```
.text
        sle     $t0, $s1, $s2
        bnez    $t0, label
label:
```

Assignment 4

#Result

Registers | Coproc 1 | Coproc 0

Name	Number	Value
$zero	0	0x00000000
$at	1	0x9fff0000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0xa0000001
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x9fffffff
$s2	18	0x9fffffff
$s3	19	0x3ffffffe
$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		0x00400024
hi		0x00000000
lo		0x00000000

#Text assigment

```
.text
start:
        li      $s1, 0x9fffffff
        li      $s2, 0x9fffffff
        li      $t0, 0
        addu    $s3, $s1, $s2           #s3 = s1 + s2
        xor     $t1, $s1, $s2           #kiem tra s1 va s2 cung hay khac dau
        bltz    $t1, EXIT
        xor     $t1, $s3, $s1           #kiem tra s1 va s3 cung hay khac dau
        bltz    $t1, OVERFLOW
OVERFLOW:
        li      $t0, 1
EXIT:
```

Assignment 5

#Result

$s0	16	4
$s1	17	3
$s2	18	16
$s3	19	48
$s4	20	0
$s5	21	0
$s6	22	0

#Text assignment

```
.text
        li      $s0, 1
        li      $t0, 1
        li      $s1, 3
        li      $s2, 16
LOOP:
        sllv    $t1, $t0, $s0           # t1 = t0 * s0 ( t0 luon bang 1 )
        beq     $t1, $s2, MUL           # if t1 < s2 thuc hien lenh nhan
        bgt     $t1, $s2, EXIT          # t1 > s2 tuc s2 khong phai luy thua cua 2 break
        addi    $s0, $s0, 1             # tang gia tri dich bit len 1
        j       LOOP
MUL:
```

```
        sllv    $s3, $s1, $s0           #s3 = s1 * s2
EXIT:
```