BÀI THỰC HÀNH TUẦN 4
KIẾN TRÚC MÁY TÍNH

Họ và tên: Đinh Huy Dương

MSSV: 20215020

Bài 1:

```
.text
start:
        addi    $s1, $zero, 0x00009999
        addi    $s2, $zero, 0x7fffffff
        li      $t0, 0          # No Overflow is default status
        addu    $s3, $s1, $s2   # s3 = s1 + s2
        xor     $t1, $s1, $s2   # Test if $s1 and $s2 have the same sign
        bltz    $t1, EXIT       # If not, exit
        slt     $t2, $s3, $s1
        bltz    $s1, NEGATIVE   # Test if $s1 and $s2 is negative?
        beq     $t2, $zero, EXIT  # s1 and $s2 are positive
                # if $s3 > $s1 then the result is not overflow
        j       OVERFLOW
NEGATIVE:
        bne     $t2, $zero, EXIT  # s1 and $s2 are negative
                # if $s3 < $s1 then the result is not overflow
OVERFLOW:
        li      $t0, 1          # the result is overflow
EXIT:
```

Đặt giá trị của $s1 = 0x9999

$s2 = 0x7fffffff = 2^31 -1 là giá trị lớn nhất được biểu diễn bởi 32 bit

➔ Kết quả tổng chắc chắn sẽ bị tràn

Kết quả:

Name	Number	Value
$zero	0	0x00000000
$at	1	0x7fffffff
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0x7fff6666
$t2	10	0x00000001
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x00009999
$s2	18	0x7fffffff
$s3	19	0x80009998
$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		0x00400040
hi		0x00000000
lo		0x00000000

Chương trình kết thúc để thanh $t0 biến thành giá trị 1 (đã phát hiện tràn).

- Trong trường hợp này, chương trình đầu tiên kiểm tra dấu của $s1 và $s2 bằng việc XOR 2 giá trị $s1 và $s2 để kiểm tra xem các bit đầu liệu có XOR ra được giá trị f, tương ứng với kết quả đó bé hơn 0. Gán vào thanh ghi $t1. Nếu $s1 và $s2 là khác dấu thì Overflow sẽ không xảy ra, ngược lại ta sẽ đi tiếp kiểm tra tiếp tổng không dấu của $s1 và $s2. Trong trường hợp này, 0x7ffffff và 0x9999 có kết quả XOR là 0x7fff6666 >0, 2 số cùng dấu dương nên tiếp tục kiểm tra.

- Gán giá trị tổng không dấu của $s1 và $s2 vào $s3. Nếu $s3 > $s1 thì kết quả không bị tràn trong trường hợp $s1 và $s2 cùng dương và sẽ không bị tràn nếu $s3< $s1 nếu $s1 và $s2 cùng âm. Vì ở trên ta đã kiểm tra được $s1 và $s2 cùng dấu, nên chỉ cần kiểm tra xem 1 trong 2 giá trị là âm hay dương để xác định được dấu của 2 giá trị. Nếu 2 số âm, chương trình sẽ rẽ nhánh sang nhãn NEGATIVE để kiểm tra điều kiện

của $s3. Trong trường hợp của chúng ta, kiểm tra được 2 số cùng dương, nên chỉ cần so sánh liệu $s3 có lớn hơn $s1 không. Ta thấy tổng = 0x80009998 < $s1 = 0x7fffffff, dẫn tới chương trình chạy đến lệnh nhảy "j" đến nhãn OVERFLOW để gán giá trị $t0 =1, kết thúc chương trình.

- Ta xét trường hợp 2 số khác dấu:

```
.text
start:
        addi    $s1, $zero, 0x00009999
        addi    $s2, $zero, 0xffffffff
        li      $t0, 0          # No Overflow is default status
        addu    $s3, $s1, $s2   # s3 = s1 + s2
        xor     $t1, $s1, $s2   # Test if $s1 and $s2 have the same sign
        bltz    $t1, EXIT       # If not, exit
        slt     $t2, $s3, $s1
        bltz    $s1, NEGATIVE   # Test if $s1 and $s2 is negative?
        beq     $t2, $zero, EXIT  # s1 and $s2 are positive
                # if $s3 > $s1 then the result is not overflow
        j       OVERFLOW
NEGATIVE:
        bne     $t2, $zero, EXIT  # s1 and $s2 are negative
                # if $s3 < $s1 then the result is not overflow
OVERFLOW:
        li      $t0, 1          # the result is overflow
EXIT:
```

$s1 = 0x9999

$s2 = 0xffffffff = -1

Name	Number	Value
$zero	0	0x00000000
$at	1	0x00009999
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0xffff6666
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x00009999
$s2	18	0xffffffff
$s3	19	0x00009998
$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

Sau phép XOR ta thấy $t1 = 0xffff6666 < 0 nên chương trình rẽ nhánh sang EXIT, không thay đổi $t0 mặc định =0. Không thể bị tràn.

Bài 2:

```
#Laboratory Exercise 4, Home Assignment 2
.text
      li    $s0, 0x0563ffa3       # load test value for these function
      andi  $t0, $s0, 0xff000000
      srl   $t0, $s0, 24          # Extract the MSB of $s0
      andi  $s0, $s0, 0xffffff00  # Clear the LSB of $s0
      ori   $s0, $s0, 0x000000ff  # Set the LSB of $s0 to 1
      andi  $s0, $s0, 0x00000000  # Clear $s0
```

- Gán giá trị $s0 = 0x0563ffa3
- Lấy MSB của $s0 = 05

  + andi $t0, $s0, 0xff000000: Gán giá trị thanh $t0 = 0x05000000

  + srl $t0, $s0, 24: Dịch sang phải 24 bit để $t0=0x00000005=MSB

$t0	8	0x00000005
$t1	9	0x00000000
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x0563ffa3

- Xóa LSB của $s0: $s0 AND 0xffffff00 = 0x0564ff00

  $s0: 

- Đặt LSB của $s0: (8 bit cuối =1) $s0 OR 0x000000ff = 0x0564ffff

  $s0: 

- Xóa $s0: $s0 AND 0 = 0

Bài 3:

a, abs $s0, $s1

Lệnh cơ bản được dịch:

```
sra         $at, $s1, 0x1f
xor         $s0, $at, $s1
subu        $s0, $s0, $at
```

Giải thích:

- Gán giá trị tạm thời ở thanh ghi $at bằng 32 bit của 1 hoặc không tùy theo dấu của $s1 thông qua lệnh dịch phải có dấu 31 bit, "sra". Ví dụ số 0xe1223444 = -51785092 <0, khi dịch phải 31 bit, ta sẽ nhận được giá trị tại $at = ffffffff (biểu diễn có dấu của số trên đầy đủ là 0xffffe123444). Hoặc nếu cho giá trị $s1 >0 thì $at = 0.
- Từ đó tiếp tục XOR $at với $s1. Nếu $s1 >0 thì $s0 = $s1, $s1 <0 thì $s0 = NOT ($s1).
- Cuối cùng trừ giá trị $s0 với $at, nếu $s1 >0 thì ra chính nó, ngược lại $s1 <0 thì sau phép trừ ra được số bù 2 của $s1 và abs($s1) =số bù 2 của $s1, $s1<0

b, move $s0, $s1

Lệnh cơ bản được dịch:

```
addu        $s0, $zero, $s1
```

Đây chỉ là lệnh gán giá trị của $s1 vào $s0

c, not $s0, $s1

Lệnh cơ bản được dịch:

```
nor         $s0, $zero, $s1
```

NOT được biến thành lệnh "nor" $s1 với số 0:

$s1 OR 0 = $s1 -> $s1 NOR 0 = NOT($s1)

d, ble $s1, $s2, LABEL: Nhảy đến LABEL nếu $s1 <= $s2

Lệnh cơ bản được dịch:

```
slt         $at, $s2, $s1
beq         $at, $zero, LABEL
```

Bài 4:

```
#Assignment 4
.text
        addi   $s1, $zero, 0x00009999
        addi   $s2, $zero, 0x7fffffff
        li     $t0, 0              # No Overflow is default status
        addu   $s3, $s1, $s2       # s3 = s1 + s2
        xor    $t1, $s1, $s2       # Test if $s1 and $s2 have the same sign
        bltz   $t1, EXIT           # If not, EXIT
        xor    $t2, $s1, $s3       # Check if $s1 and $s1 + $s2 have the same sign
        bgtz   $t2, EXIT           # If yes, then EXIT
        li     $t0, 1              # Overflow detected
EXIT:
```

Kết quả:

Name	Number	Value
$zero	0	0x00000000
$at	1	0x7fffffff
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0x7fff6666
$t2	10	0x80000001
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x00009999
$s2	18	0x7fffffff
$s3	19	0x80009998
$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

Bài 5:

```
#Assignment 5
.text
	li	$s0, 46		# The multiplicand
	li	$s1, 8		# The multiplier, a power of 2
MUL:
	beq	$s1, 1, EXIT		# If the multiplier =1, EXIT
	sll	$s0, $s0, 1		# $s3 = $s0 *2
	srl	$s1, $s1, 1		# $s1 = $s1/2
	j	MUL
EXIT:
```

Phép tính: 46 x 8

Kết quả phép nhân được lưu vào thanh ghi $s0:

Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000001
$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	0x00000170
$s1	17	0x00000001
$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		0x0040001c
hi		0x00000000
lo		0x00000000

Kết quả: $s0 = 0x170 = 368 = 46 x 8