Laboratory Exercise 4

Arithmetic and Logical operation

Goals

After this laboratory exercise, you should know how to use arithmetic, logical and shift instructions. In addition, you should also understand overflow in arithmetic operation and how to detect it.

Literature

Behrooz Parhami (CAMS): Section 5.3

Preparation

Before you start the exercise, you should review the textbook, section 5.3 and read this laboratory carefully. You should also review the Laboratory Exercise 2.

Assignments at Home and at Lab

Home Assignment 1

The sum of two 32-bit integers may not be representable in 32 bits. In this case, we say that an overflow has occurred. Overflow is possible only with operands of the same sign. For two nonnegative (negative) operands, if the sum obtained is less (greater) than eitheir operand, overflow has occurred. The following program dectects overflow based on this rule. Two operands are stored in register $s1 and $s2, the sum is stored in register $s3. If overflow occur, $t0 register is set to 1 and clear to 0 in otherwise.

```
#Laboratory Exercise 4, Home Assignment 1
.text
start:
      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:
```

Home Assignment 2

The following program demonstrates how to use logical instructions to extract information from one register. We can extract one bit or more according to the mask we use. Read this example carefully and explain each lines of code.

```
#Laboratory Exercise 4, Home Assignment 2
.text
      li    $s0, 0x0563       #load test value for these function
      andi  $t0, $s0, 0xff    #Extract the LSB of $s0
      andi  $t1, $s0, 0x0400  #Extract bit 10 of $s0
```

Home Assignment 3

This example show how the shift operations used to implement other instructions, such as multiply by a small power of 2.

```
#Laboratory Exercise 4, Home Assignment 3
.text
      li    $s0,1             #s0=1
      sll   $s1,$s0,2         #s1=s0*4
```

Assignment 1

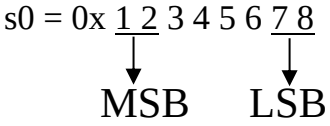
Create a new project to implement the Home Assigment 1. Compile and upload to simulator. Initialize two operands (register $s1 and $s2), run this program step by step, observe memory and registers value.

Assignment 2

Write a program to do the following tasks:

- Extract MSB of $s0
- Clear LSB of $s0
- Set LSB of $s0 (bits 7 to 0 are set to 1)
- Clear $s0 (s0=0, must use logical instructions)

MSB: Most Significant Byte
LSB: Least Significant Byte



Assignment 3

Pseudo instructions in MIPS are not-directly-run-on-MIPS-processor instructions which need to be converted to real-instructions of MIPS. Re-write the following pseudo instructions using real-instructions understood by MIPS processors:

a. `abs  $s0,s1`
   `s0 <= | $s1 |`

b. `move  $s0,s1`
   `s0 <= $s1`

c. `not  $s0`
   `s0 <= bit invert (s0)`

d. `ble  $s1,s2,L`

```
if (s1 <= $s2)
     j L
```

Assignment 4

To dectect overflow in additional operation, we also use other rule than the one in Assignment 1. This rule is: when add two operands that have the same sign, overflow will occur if the sum doesn't have the same sign with either operands. You need to use this rule to write another overflow detection program.

Assignment 5

Write a program that implement multiply by a small power of 2. (2, 4, 8, 16, etc for example).

Conclusions

Before you pass the laboratory exercise, think about the questions below:

- What is the difference between SLLV and SLL instructions?
- What is the difference between SRLV and SRL instructions?