Laboratory Exercise 4

Arithmetic and Logical operation

Goals

After this laboratory exercise, you can know how to use arithmetic, logical and shift instructions. In addition, you can understand what is overflow in arithmetic operations and how it is dectected.

Preparation

You need to review the textbook and read this learning material carefully.

Sample Code and Assignments

Sample Code 1

When the result of a arithmetic operation cannot be represented with the available hardware (e.g. a 32-bit word), the result overflows. In case of addition, overflow occurs when adding two positive numbers and the sum is negative, or vice versa. In case of substraction, overflow occurs when we subtract a negative number from a positive number and get a negative result, or when we subtract a positive number from a negative number and get a positive result. However, the following program shows how an overflow in addition is dectected based on a different rule that is "*overflow in addition occurs when the sum of two non-negative (or negative) numbers is less (or greater) than eitheir operand*". In this program, the two operands of addtion are stored in registers s1 and s2, the sum is stored in register s3. If overflow occurs, t0 = 1; otherwise, t0 = 0.

```
#Laboratory Exercise 4, Sample Code 1
.text
start:
      li    $t0,0             # No overflow is set as default status
      addu  $s3,$s1,$s2       # s3 = s1 + s2
      xor   $t1,$s1,$s2       # Check if $s1 and $s2 have the same
                              # sign?
      bltz  $t1,EXIT          # If not, exit
      slt   $t2,$s3,$s1
      bltz  $s1,NEGATIVE      # Check if $s1 and $s2 is negative?
      beq   $t2,$zero,EXIT    # s1 and $s2 are positive
                              # If $s3 > $s1 then the result does not
                              # overflow

      j     OVERFLOW
NEGATIVE:
      bne   $t2,$zero,EXIT    # s1 and $s2 are negative
                              # If $s3 < $s1 then the result
                              # does not overflow
OVERFLOW:
      li    $t0,1             # The result overflows
EXIT:
```

Sample Code 2

The sample code shows how logical instructions can be used to extract information from a register. Depending on the mask, more than one bit can be extracted. Read this sample code and explain it line by line.

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

Sample Code 3

This sample code shows how a shift instruction is used to calculate a power of 2, instead of using a multiply instruction.

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

Assignment 1

Create a new project to run the Sample Code 1 on your MARS simulator. Initialize the two operands of addition (i.e. s1 and s2). Then, run this program step by step and observe contents in the related memory locations and registers.

Assignment 2

Write a program to do the following tasks:

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

MSB: Most Significant Byte
LSB: Least Significant Byte

```
s0 = 0x 1 2 3 4 5 6 7 8
          ↓         ↓
         MSB       LSB
```

Assignment 3

Pseudo instructions are not directly run on a MIPS processor, and are converted into real instructions, which the MIPS processor can understand. Convert the following pseudo instructions into the corresponding real instructions and explain why they work.

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 an overflow in addition, we can use the following rule that is "*when we add two numbers that have the same sign, if the sum doesn't have the same sign as the two numbers, overflow will occur*". Write another overflow detection program using this rule. (You can refer to Sample Code 1).

Assignment 5

Write a program that implement multiplying an integer by a small power of 2 (e.g. 2, 4, 8, 16, …).

Quizes

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