Laboratory 3

Load/ Store, Jump & Branch instructions

Goals

After this laboratory exercise, you should know how to use load/store, jump and branch instructions. You should also be able to implement high level programming language structures such as conditional statement (if-then-else), loop and selection statement (switch-case)

Literature

Behrooz Parhami (CAMS): Section 5.4, 5.5

Preparation

Before starting this laboratory, you should review textbook and read the entire laboratory exercise in detail. You also need to review Laboratory Exercise 2 and try to experience MARS by yourself.

Assignments at Home and at Lab

Home Assignment 1

This home assignment implements "if-then-else" statement using some fundamental instructions, such as slt, addi, jump and branch.

```
if (i<=j)
      x=x+1;
      z=1;
else
      y=y-1;
      z=2*z;
```

At first, you should draw the algorithm chart for this statement. After that, you read this example carefully, try to clarify the function of each intructions.

```
#Laboratory Exercise 3, Home Assignment 1
start:
      slt   $t0,$s2,$s1       # j<i
      bne   $t0,$zero,else    # branch to else if j<i
      addi  $t1,$t1,1         #     then part: x=x+1
      addi  $t3,$zero,1       # z=1
      j     endif             # skip "else" part
else: addi  $t2,$t2,-1        # begin else part: y=y-1
      add   $t3,$t3,$t3       # z=2*z
endif:
```

Home Assignment 2

The following example demonstrates how to implement loop statement. This program computes the sum of elements of array A.
With C language:

```
sum = 0;
for (int i = 0; i < n; i += step)
    sum += A[i];
```

With assembly language:

```
        sum = 0
        i = 0
loop:   if (i >= n) goto endloop
        sum = sum + A[i]
        i = i + step
        goto loop
endloop:
```

Assuming that the index i, the starting address of A, the comparison constant n, step and sum are found in registers $s1, $s2, $s3, $s4 and $s5, respectively. You should try to understand each line in this code.

```
#Laboratory 3, Home Assigment 2
.text
      addi  $s5, $zero, 0     # sum = 0
      addi  $s1, $zero, 0     # i = 0
loop: slt   $t2, $s1, $s3     # $t2 = i < n ? 1 : 0
      beq   $t2, $zero, endloop
      add   $t1, $s1, $s1     # $t1 = 2 * $s1
      add   $t1, $t1, $t1     # $t1 = 4 * $s1
      add   $t1, $t1, $s2     # $t1 store the address of A[i]
      lw    $t0, 0($t1)       # load value of A[i] in $t0
      add   $s5, $s5, $t0     # sum = sum + A[i]
      add   $s1, $s1, $s4     # i = i + step
      j     loop              # goto loop
endloop:
```

Home Assignment 3

A switch/case statement allows multiway branching based on the value of an integer variable. In the following example, the switch variable test can assume one of the three values in [0, 2] and a different action is specified for each case.

```
switch(test) {
        case 0:
            a=a+1; break;
        case 1:
            a=a-1; break;
        case 2:
            b=2*b; break;
        }
```

Assuming that **a** and **b** are stored in registers $s2 and $s3. You should read this code section carefully, understand how to implement switch/case statement.

```
#Laboratory Exercise 3, Home Assignment 3
.data
test: .word 1
```

```
.text
            la    $s0, test   #load the address of test variable
            lw    $s1, 0($s0) #load the value of test to register $t1
            li    $t0, 0      #load value for test case
            li    $t1, 1
            li    $t2, 2
            beq   $s1, $t0, case_0
            beq   $s1, $t1, case_1
            beq   $s1, $t2, case_2
            j     default
case_0:     addi  $s2, $s2, 1       #a=a+1
            j     continue
case_1:     sub   $s2, $s2, $t1     #a=a-1
            j     continue
case_2:     add   $s3, $s3, $s3     #b=2*b
            j     continue
default:
continue:
```

Assignment 1

Create a new project to implement the code in Home Assignment 1. Initialize for i and j variable. Compile and upload to the simulator. Run this program step by step, observe the changing of memory and the content of registers at each step.

Assignment 2

Create a new project implementing the code in Home Assignment 2. Initialize for i, n, step, sum variables and array A. Compile and upload to the simulator. Run this program step by step, observe the changing of memory and the content of registers by each step. Try to test with some more cases (change the value of variables).

Assignment 3

Create a new project implementing the code in Home Assignment 3. Compile and upload to the simulator. Run this program step by step; observe the changing of memory and the content of registers by each step. Change the value of test variable and run this program sometimes to check all cases.

Assignment 4

Modify the Assignment 1, so that the condition tested is

a. i <j
b. i >= j
c. i+j <= 0
d. i+j > m+n

Assignment 5

Modify the Assignment 2, so that the condition tested at the end of the loop is

a. i < n
b. i <= n
c. sum >= 0
d. A[i] == 0

Assignment 6

Using all the above instructions and statements, create a new project to implement this function: find the element with the largest absolute value in a list of integers. Assuming that this list is stored in an integer array, and we know the number of elements in it.

Conclusions

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

- Which registers are affected by branch instruction?