Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Archana Chidanandan and Larry Merkle

Homework 1
Assembly Language and Machine Language
Max Points: 45 points

Directions

This assignment is due Tuesday, September 20 for all three sections. Submit your solutions on a separate sheet of paper. You may use SPIM and a calculator as required.

Learning Objectives

In the process of completing this homework assignment, students will develop their abilities to

- Implement algorithms involving arrays, selection and iteration abstraction in assembly language.
- Write programs with procedures.
- Use the stack to store values in procedure calls.
- Predict the actual assembly language instructions corresponding to a pseudo-instruction.
- Interpret a sequence of binary data, given the rules for conversion.

Problems

1. [10 pts] For each pseudoinstruction listed below, give a minimal sequence of actual MIPS instructions to accomplish the same thing. You may need to use `$at` for some of the sequences. "`big`" refers to a 32-bit immediate value and "`small`" refers to a 16-bit immediate value.

a. `move $t5, $t3`
b. `clear $t5`
c. `li $t5, small`
d. `li $t5, big`
e. `lw $t5, big($t3)`
f. `addi $t5, $t3, big`
g. `beq $t5, small, label`
h. `beq $t5, big, label`
i. `ble $t5, $t3, l`
j. `bgt $t5, $t3, l`
k. `bge $t5, $t3, l`

2. [10 pts] Write a documented MIPS assembly language program that will use a procedure "`CountSmallerInt`" to count the number of values in an integer array that are smaller than a given number. The procedure will accept three input parameters: the address of the array, the size of the array and the number to compare the elements of the array with. The procedure must return the count. In "`main`", write the count to memory. The array values are present in memory, while the number to compare with must be accepted as an input from the console. Prompt the user for this value with a meaningful message. *Note:* Figure A.9.1 (on the CD) has the list of `syscall` codes.

3. [10 points] Given the bit pattern below,

1000 1111 1110 1111 1100 0000 0000 0000

what does it represent, assuming that is it

a. a two's complement integer?
b. an unsigned intger?
c. a sign-magnitude integer?
d. a MIPS instruction?

4 a. [10 points] Complete the MIPS procedure `power` in the following pages by filling in the provided spaces with MIPS assembly language statements such that

- The procedure returns x^y, where *x* and *y* are its parameters, and
- It follows the MIPS register usage conventions.

Assume the existence of another MIPS procedure `product` that returns the product of its two parameters.

b. [5 points] Complete the MIPS main procedure that follows procedure power by filling in the provided spaces with MIPS assembly language statements such that

- The program calls the `power` procedure to compute 3^5, and
- It follows the MIPS register usage conventions.

Read all the provided parts of the program, before you begin.

```
#
# Procedure power calculates x^y using repeated multiplication, where x
# and y are the two parameters of the procedure, and returns that value
#
#
#  In the procedure, register usage:
#
#  $t0 - x
#  $t1 - y
#  $t2 - count
#  $t3 - value

                  .text
power:

#
# Procedure entrance and initialization
#
```

Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Archana Chidanandan and Larry Merkle

```
            addi  $t2, $zero, 1   # count = 1
            addi  $t3, $zero, 1   # value = 1

#
# For each of y iterations, set value = value * x
#
loop:
      bgt   $t2, $t1, exit1   # if (count > y) done with loop
```

```
                                                    # Call “product”
                                                    # to calculate
                                                    # value * x.
     jal    product
```

```
      addi  $t2, $t2, 1   # count = count + 1       # Store result in
      j  loop                                       # value.

#
# Exit the procedure
#
exit1:
```

Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Archana Chidanandan and Larry Merkle

```
      jr  $ra


#
# Main program starts here
#
main:
      li    $s0, 3      # x
      li    $s1, 5      # y
                                                  # Call "power"
                                                  # to calculate
                                                  # x^y




      jal   power
```