Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Archana Chidanandan and J.P. Mellor

Homework 1

Assembly Language
Max Points : 40

Directions

This assignment is due Thursday, September 16, 2004. Submit your solutions on a separate sheet of paper. *Hint*: Use SPIM.

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.
- Predict the actual assembly language instructions corresponding to a pseudoinstruction.

Problems

1. [18 points = 6 x 3 points] `li` is an example of a MIPS pseudoinstruction. It is not an instruction that is implemented in hardware and is not really a MIPS assembly language instruction. However, the assembler can translate the instruction to "real" MIPS instructions and thus makes programming simpler. A pseudoinstruction can therefore be translated to one or more "real" MIPS instructions. For the pseudoinstructions in the following table, determine a minimal sequence of actual MIPS instructions to accomplish the same thing. You may need to use $at for some of the sequences. In the following table, `big` refers to a specific number that requires 32 bits to represent and `small` to a number that can be expressed using 16 bits.

Pseudoinstruction	What it accomplishes
`move $t5, $t3`	`$t5 = $t3`
`clear $t5`	`$t5 = 0`
`li $t5, big`	`$t5 = big`
`li $t5, small`	`$t5 = small`
`beq $t5, small, L`	`If ($t5 == small) go to L`
`ble $t5, $t3, L`	`If ($t5 <= $t3) go to L`

2. [7 points] Write a procedure *Find* in MIPS assembly language, adhering to MIPS register usage conventions. The procedure should take three arguments. The first argument is a pointer to an array of integers, the second is the number of elements in the array, and the third is an integer value. *Find* must find and return for the third argument, the index value of its first occurrence in the integer array. If the integer value does not exist in the array, then *Find* must return -1. Your procedure must follow MIPS register conventions.

Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Archana Chidanandan and J.P. Mellor

3. [15 points] In (hmwk1-3.asm), we see a procedure called MoveObject. The procedure accepts two arguments, a velocity component and a position component of a moving "ball". Using these values, it determines the new position of the "ball", such that the "ball" remains with the specified boundaries.
    a. Write a procedure "MoveBall" that calls the "MoveObject" procedure twice, first to update the "x" position and next to update the "y" position of the "ball".
    b. Write a main program that initializes the x and y positions and velocities of the "ball" and then calls the procedure "MoveBall" repeatedly to update the "x" and "y" positions.

   The program must follow the MIPS register conventions (pages A22-23 of your book). You may not modify the "MoveObject" procedure.