CSE 378 Autumn 1999

Machine Organization and Assembly Language Programming

# Problem Set / Programming Assignment #3

Due: Thursday, October 21

This assignment has two components: (1) Problems from the text relative to the Performance of Computer Systems and (2) Two simple SPIM programs.

You should be familiar with the material covered in Chapters 2, 3, 4 (Sections 4.1 through 4.4) and Appendix A.

**Part I**

1. Problems 2.1, 2.2, 2.3, 2.4, 2.5.
2. Problems 2.10, 2.11, 2.12.
3. Problems 2.18, 2.20, 2.21, 2.22, 2.23.
4. Before doing the next exercises, be sure to read about Amdahl's law (page 101). Problems 2.41, 2.42

**Part II**

In this part of the assignment you'll have to write some simple subroutines in MIPS assembly language using SPIM to debug and test your programs.

Instructions on how to **turnin** your program as well as test input examples and how to use them will be given to you in Sections.

Recall that it is **imperative** that your programs be **commented**. Programs without comments might not be looked at, even if correct!

**Problem 1**. Write a procedure in MIPS assembly language that finds the number of elements strictly greater than the last element and the number of even elements in an array of integers (each element is a 32-bit positive, negative, or null integer).

Your procedure will take two arguments: a pointer to the array of integers that will be passed in register $a0 and the number of elements in the array that will be passed in register $a1. Your procedure will return the number of elements

strictly greater than the last element in $vo and the number of even elements in $v1.

Please name your procedure **problem1** so that it can be tested easily.

**Problem 2**. Write a procedure in MIPS assembly language that converts a character string of ASCII digits into the integer value represented by the string. The character string is null terminated and a pointer to it will be given as the argument in register $a0. The ASCII representation of characters is in your book page 142 (with the NULL character having the representation 0). Thus the ASCII string "+24" is represented by the 4 bytes (where the value of each byte is given in decimal notation) "43", "50", "52", "0".

Your procedure named **problem2** should return in register $v0 the value of the positive, negative or zero representation of the string. If the string is valid, i.e., is a sign followed by a series of digits, or a series of digits, register $v1 should contain 0. Otherwise, if the string is not valid (e.g., contains a letter, or a digit followed by a sign, or is empty, or does not contain any digit etc.) the value returned in $v0 is immaterial but register $v1 should contain a 1.