Indian Institute of Technology Kharagpur

AUTUMN Semester, 2019
COMPUTER SCIENCE AND ENGINEERING

Computer Organization Laboratory

Assignment-3: MIPS-32 Assembly Language Programming

Full Marks: 20

Time allowed: 6 hours

INSTRUCTIONS: ATTEMPT BOTH PROBLEMS. Make one submission per group of your source code on Moodle. Name your submitted source files following the format `Assgn_3_Prob_1_Grp_<Group_no>.s` **(e.g.** `Assgn_3_Prob_1_Grp_25.s`**), etc. Inside each submitted file, there should be a clear header describing the assignment no., problem no., semester, group no., and names of group members. Liberally comment your code to improve its comprehensibility.**

1. **[String Operation in MIPS-32]** Write a complete MIPS-32 program to convert a null-terminated ASCII string stored in memory, possibly containing a mixture of upper-case letters, lower-case letter, numbers and spaces, to lower-case, and store it back to memory. First store the input string in memory by creating space for it using the `.space` directive (you may assume a maximum length of the string) and collecting the string from the user, or hard-code it in your program using the `.asciiz` directive. After conversion, store back the converted string starting at an address created by the `.space` directive (remember: each character in an ASCII-encoded string occupies exactly one byte in memory). You may or may not use MIPS procedure call to solve the problem. After sorting, print the converted string on the console, accompanied by a proper message. (10 marks)

2. **[Moving Average in MIPS-32]** Write a complete MIPS-32 program to calculate the moving window average of integers stored in an one-dimensional array. The code accepts the following parameters from the user: (i) the size of the array (n); (ii) the window size (w), and, (iii) the values in the array (to be read in a loop). Use the `.space` directive to create large enough space at the beginning, and the value of n should be such that the numbers entered would fit in the allocated space (remember: each integer occupies exactly 4 bytes in memory). The averaging is carried out using the following scheme: `for` (i=0; i≤ n-w; i++) a[i] = $\frac{1}{w}\sum_{j=0}^{w-1}$ `a`[i + j]. **Note that your code should not use any extra array**. Finally, print the integer part of the average to the user with a proper message. (10 marks)