Indian Institute of Technology Tirupati

CS2610 Assembly Language Programming

Credit: L-T-P-C : 0-0-3-2
Instructor: Jaynarayan T Tudu <jtt@iittp.ac.in>

Experiment 4: Assembly Programming in MIPS for Memory and Load-store Instructions

---

1 Objective

To understand data movement instruction and memory architecture. In this experiments you are expected to write a set of assembly programs to understand, primarily, the function of load and store instructions along with the arithmetic, branch and jump instructions.

2 Programs

Go through in detail description of the program that you need to write in MIPS32 instructions. The restriction of using the instructions need to be followed.

1. Counting the sequence:
   (a) Write a MIPS32 program to count a number of ones and zeros in a 32 bit binary number. The count results are to be kept back in the data section of memory (you need to find out where to store back the results). Assume that the given number is stored in register $r1$.
   (b) Write a MIPS32 program to detect whether a given signed integer number is positive of negative. If the number is positive indicate that by storing value zero in register r2 and indicate the negative by storing value 1 in register r2.
   (c) Write a program to count the binary sequence group, zero group or one group, in a given binary number of 32 bits. For example your program should indicate 3 and 2 for the binary number 00000011111111000001111110000000 for zero sequence and one sequence respectively.
2. More than 32 bit: Write a program to perform an addition of two 128 bit signed integer numbers. Assume that the two numbers are stored in the memory in such a way the first number occupies first four locations and the second number occupies the next four locations. Assume that the memory is byte addressable and the data width is 32bit. The final results has to be stored back in the subsequent memory location. (Hints: You need to to detect the overflow for every part of 32 bit addition. There are several ways to detect this, but my suggestion is to use the method that has been taught in digital design class)
3. Array and Array:
   (a) Write a program to swap the content of two arrays, a and b, having 32 bit integer numbers. The size each array is a variable stored in one of the GPRs. You are advised to use the either displacement addressing (example: add r1, 20(r2)) or indexed addressing (example: add r1, (r2 + r3)) mode to access the arrays. Assume that both the arrays are stored in memory location in .data segment.
   (b) Repeat the above program for the ASCII character as the content of the arrays.
4. Multiply without mult:
   Write a program to perform multiplication operation on two unsigned binary number of 32 bit each without using any *mult* type instruction. This program suppose to implement the array multiplier. The intermediate partial product and the final results needs to be stored in a feasible memory location at .data section. This program needs to be implemented, largely, by use of shift, add, branch instructions.

3 Report

You have to submit a hand written report on the 128 bit number addition with detail description of the algorithm that you have used for performing the addition operation. In the report you need to demonstrate the working of your program for two 128 bit numbers.

4 Evaluation

Demonstration + Viva: 3 + 1 points,
Report: 1 points

5 Timeline

Demonstration: 5:30 pm
Uploading of programs in github: 6:00 pm
Report: 22^{nd} Feb, 2018