ECSE 324 – Lab Report #1 – Group 34

Jia Wei Sun, 260866206
Karl Michel Koerich, 260870321
12 October 2019

Overview

For this lab, we worked with the DE1-SoC Computer System and learned how to program it in ARM assembly. After performing the required step from Lab 0 and reading the Introduction to the ARM Processor Using Alter Toolchain, we were still not completely familiarized with the board and the how to program it, but we managed to implement all the desired functionalities of lab 1.

Firstly, we followed the steps from section #1: *Working with the DE-SOC Computer System*. By the end, we had generated and ran a simple code that finds the maximum number from a list of "Numbers" with length "N". The next step was to find a range of a set of data. We adapted the code from the first part to find the maximum and minimum value from a list of "Numbers" with length "N," and then performed a subtraction to get the value of the range. The last task was to find the maximum and minimum value of an algebraic expression of length N.

Although not experienced and familiar with the board and assembly programming, we managed to perform all the proposed challenges and we were satisfied with the final results.

Approaches

To solve the 2 proposed challenges, we first though of the problems in a high level. After reading the requirements of part 2.1, we concluded that some code from part 1 could be reused. The goal was to find the range of a list of numbers, so first we would have to find the maximum and minimum value of the list, then subtract them. To implement such logic, we used the logic of part 1, where we have 1 loop to iterate over the list of numbers and find the maximum.

Instead of using one loop, we used 2: one to find the maximum, one to find the minimum. At the end, after iterating over the list of numbers twice, we calculate the difference and store it in a new register. We use a total of 8 registers, where the main ones are R0, R6 and R8. The first holds the value of the maximum, the second holds the value of the minimum, and the third hold the value of the difference.

After reading the requirements of part 2.2, we understood that we were supposed to find the maximum and minimum value of an arithmetic expression composed of 4 numbers. What was not entirely clear to us was if we were supposed to account for the case when N = 1 and M = 3, or simply when N = M = 2. In case of doubt, we decided to consider both cases, therefore we found all possibilities for both cases: $(x_1 + x_2) * (y_1 + y_2)$ and $(x_1) * (y_1 + y_2 + y_3)$. The high-level approach for this problem consists of mainly 3 loops: one to calculate the sum of all the terms in the list, one to computer all possibilities when N=M=2, and one to compute all possibilities when N=1 and M= 3. We first step was to iterate over the list of numbers and storing the sum (S) in register R0. After having the value of S, we reset the loop counter and prepared the pointer to iterate over the list again, but this time to find the value of the arithmetic expression when N=M=2. To do that, we fixed one pointer at the first element in the loop and used one pointer to iterate over the 3 remaining elements. To get the sum of $(x_1 + x_2)$, we would simply add the value of the first element in the list with the value the iterator pointer was pointing to. To get the sum $(y_1 + y_2)$, we would simply do $S – (x_1 + x_2)$. After having the value of the sums of the arithmetic expression, all it was left to do was to multiply them and compare the result with the current maximum and minimum. If the result was greater than max or smaller than min, we would update the current max or min. Since N=M=2, this loop would only have 3 iterations. The next step would be to reset the counter and try the case where N = 1 and M = 3. To do so, we used the same principle of a pointer to a value in the list but this time, we would be incrementing the position in which the pointer pointed at, meaning we would go over all the 4 numbers in the list. The number of the current pointed

position would be (x_1) while ($y_1 + y_2 + y_3$) would be S – (x_1). After having the value of the sums of the arithmetic expression, all it was left to do was to multiply them and compare the result with the current maximum and minimum. If the result was greater than max or smaller than min, we would update the current max or min. At the end, R12 held the value of max, and R11 held the value of min.

Challenges

The main challenge we had was to understand the registers that held value and the ones that held pointers to other locations. This concept was very abstract for us in the beginning of the lab, which led us to spend a lot of time trying to understand the given code. After analysing the problems, reading the book and online sources, and asking for help from TAs, we better understood the logic behind assembly programing and it became a matter of minutes until we had the entire logic for al problems designed.

After a few tests, we verified our code was working very well, but we were making use of more registers than needed. The following step was to try and reduce the number of used registers to avoid memory problems and make the overall program mode efficient in terms of register usage.

Conclusion

The main purpose of this lab was to not only make us more familiar with the board and assembly specifications, but also to start solving problems thinking in terms of assembly programming. We were satisfied with the results, even though it took us more time and expected to solve all the problems and demo. We also noticed an inconsistence in the Intel programming platform, which sometimes does not open the editor window properly.