ECSE 324 Computer Organization

LAB 1 REPORT

Shaluo WU 260713923

Yi ZHU 260716006

2018-10-10

Lab1 consists of four parts, largest integer program, standard deviation, centering, and the sorting program. In each program, students are asked to write ARM code operating on a list of numbers. The following report briefly introduces the approach used for each program.

1. Largest Integer Program

This part is relatively easy because the code is given in the lab instruction. The program creates a loop which returns the biggest number in the list. The code is quite straightforward, except the SUBS and BEQ operations. In here, the SUBS acts as a decrement counter and sets a flag when the result reaches zero. Next, the BEQ operation detects the flag when the counter equals zero and ends the loop accordingly.

2. Standard Deviation Program

This question is divided into three parts, finding the maximum, minimum, and adding them together and divide by four. The division operation can be achieved by simply shifting the result to the right by two bits. Here the LSR operation is used because the difference between the max and min is never negative.

The processes of finding the maximum and minimum values are done altogether in one loop. We set the first number in the list as the biggest and smallest in the beginning and take the following numbers and compare them with the first number. If it is not bigger, then send it to compare with the min. If yes, then set the number as the new maximum and skip the comparison with the min number. It is worth mentioning that, the MOV operation is used when setting the max and min values. MOV is directing the value in a register into another register. In this case, it is same as writing ADD R, R', #0. They differ from LDR and STR operations where memory must be accessed and then sent into designated register.

3. Centering Program

This part contains three steps, finding the sum of all numbers, finding the average, and subtracting the average from each entry. The three steps are conducted in three loops. The first loop is easy, adding all the numbers together into one register.

In the second loop, we divide the both the sum and the size by 2 for as many times as needed, which can be done through shifting. The loop will stop when size

equals 1, which becomes zero after another shifting. In this step, the ASR operation is used instead of LSR, because if input numbers in the list are negative, LSR cannot do the job. The current average in the register is the number resulting from the previous loop. The mechanism of this method is under the assumption that the size is the power of two, so that we will always get integer results until the size becomes zero and we get our average.

The third loop is subtraction. We used R6, which is 1 greater than the size, because the BEQ operation is after the decrement line and we start with R6 – 1. Another interesting observation here is that, if numbers in the list are not all equal, then there will be negative numbers will be returned as part of the list. Here, 2's complement hexadecimal is used to represent the numbers in the register. For example, -3 is 0xFFFFFFFD.

4. Sorting Program

Completely following the bubble sort logarithm provided in the lab instruction, we constructed our code with a while-loop and a for-loop with an if-else statement. Noticeably, high level language does not involve storing data back to memory, therefore we have created an else statement for storing the sorted values. The code has four parts, the while-loop, for-loop, if statement, and the else statement, which does nothing but storing.

In the beginning, we set the condition that, if the length of list is less than 2, then there is no need for sorting, so the program ends.

For the while-loop, we used a register R1 as what otherwise would be a Boolean variable, as shown in the instructions.

In the for-loop, we put the increment counter right after the branching line. This is different from a typical for-loop, but it operates fine since the counter will not be touched in the following. This is for the convenience of coding process, otherwise we will have to add lines after the if and else statements respectively. Also, the LDR operation is used for storing the content in the array into the register. The STR operation that is used later in swapping is writing the value back into memory.

Within the for-loop, if all entries in the list are in desired order, which means the if statement is never visited, the program counter points to the else-statement. The Boolean variable will remain as true, from the last line in the while-loop. Data will be stored and the registers pointing to the next numbers will be set. The program ends after the else-statement is accessed.