**Lab06 – Assembly: Control Structures, Arrays and Functions**

COMP256 – Computing Abstractions

Dickinson College

Spring 2023

Prof. Grant Braught

**Introduction:**

We’ve spent a good bit of time learning how a compiler could translate different HLL features into Assembly language. We started with simple computations (add, subtract…). We saw how control structures (if, if/else, while, for) can be translated into assembly using conditional and unconditional branches. We learned how arrays and more generally references are translated by using indirect addressing mode instructions. Finally, we learned how a compiler can use a stack to translate the HLL function calls we are used to into assembly language. This lab will give you some practice “being the compiler” by having you translate some other HLL programs into assembly language.

**Setup:**

1. You will be using your Comp256Assembly container for this assignment. To get started:

* Start the Comp256Assembly container using Docker Desktop.
* Connect to the container using the TigerVNC Viewer.
* Use the terminal or File Manager (i.e. the file cabinet icon in the launcher at the bottom of the screen) to create a new directory named Lab06 within your home directory. You’ll save all of the files that you create for this lab into that directory.

**Control Structures:**

2. Consider the following HLL Program that finds smallest of the three numbers entered:

read a

read b

read c

if (a < b) {

d = a

}

else {

d = b

}

if (d < c) {

e = d

}

else {

e = c

}

return e

Translate the above HLL program into assembly language and save it in a file named No2.asm within your Lab06 directory.

Be sure to assemble and test your program. Your program should work when the minimum value is entered first, second or third.

**Arrays:**

3. Consider the following HLL Program that sums up all of the values in an array:

int[] list = {7, 6, 9, 3, 1, 8, 2}

length=7

total = 0

for (i=0; i<length; i++) {

total = total + list[i]

}

return total

Translate the above HLL program into assembly language and save it in a file named No3.asm within your Lab06 directory.

Be sure to assemble and test your program. Your program should work with any array that it is given (e.g. when the minimum number appears first, last or anywhere in between).

**Functions:**

The following example was used several times in class when introduction function calling and implementation in assembly language:



In this section you will translate this high-level language program into an equivalent assembly language program using the techniques that we learned in class.

The slide below, from the end of LA6, outlines all of the steps that are necessary to properly translate both the function calls and the function implementations. For full credit your solutions to the next two questions must use this approach to calling and implementing functions.



The max function:

4. Let’s start by implementing the max function. Consider the following simpler program that uses only the max function from above:

main () {

read p

read q

r = max(p,q)

print r

}

Translate the above HLL program and the max function into assembly language and save it in a file named No4.asm within your Lab06 directory.

Note the following:

* The No4.asm file must contain both translation of main and max.
* The code for main must come first and must end with a HALT statement.
* The code for max should come after the HALT statement.
* Your code for max should use only registers for its parameters and local variables (i.e. **no label created using .word may appear anywhere in the code for max**).

Be sure to assemble and test your program. Your program should work whether the maximum number is entered first or second.

The max3 function:

A correct solution to this question is required to receive a score of 3 on the lab. A score of 2 may be achieved without completing this question if questions 2-4 are correct.

5. Translate the full HLL program from above including main, max3 and max into assembly language and save it in a file named No5.asm within your Lab06 directory.

Note the following:

* The No5.asm file must contain translations of main, max3 and max.
* The code for main must come first and must end with a HALT statement.
* You may copy your working code for max from No4.asm into No5.asm.
* Your code for max and max3 should use only registers for their parameters and local variables.

Be sure to assemble and test your program. Your program should work whether the maximum number is entered first, second or third.

**Submitting the Lab:**

6. Submit this lab as follows:

a. Use the following commands in a terminal window in the container:

cd /home/student

tar -zcvf Lab06.tar.gz Lab06

b. Use the FireFox browser in the container to go to the course Moodle site and submit the Lab06.tar.gz file to the LAB06 assignment. Note that there is no assignment sheet to submit for this lab, just your assembly code.