CS520 FALL 2019

LAB 9

Description

The goal of this lab is to write a simple Intel 64 assembly language function and learn how to call this from C code, as well as to get a better understanding of how frames work.

1. Write an Intel 64 assembly language function, called **getFP**, to return **the caller's frame pointer**, i.e. the contents of the **%rbp** register. This function takes no arguments and needs to be callable from C code. Place this function in a file called **getFP.s**. From the C code point of view this function returns either a **long** or some kind of pointer type. That is, this function returns a 64-bit value. You can use gcc as before to assemble this file, the compiler will recognize .s files as assembly source files.
2. Test the function by calling it from the **main** function of a C program and explore the stack frames of **main** and the caller of **main**. What is in the old **%rbp** slot of the last frame on the stack? Look for some way to be able to identify the last frame on the stack. (Keep this C code in a separate file from the function because you will not be submitting this code that you used for testing. This is just your initial test driver).
3. Write a C function, called **frameCount**, that takes no arguments and uses the assembly language function of Step 1 to count and return the number of stack frames that there are on the stack at the time the function is called. This count should not include the function performing the count. The count should include **main** but not the caller of **main**. Place this function in a file called **framecount.c**.
4. Test your C function by calling it from your test driver (indicated in Step 2), initially with just the call to frameCount() and later with a variety of number of frames on the stack when you call the function frameCount(). I would like to see how you test your code, submit your best (or most thorough) version of your test driver in a file called **lab9.c** (it's worth 10 points, this must a working version of the driver with the other files you submit, if we cannot build or run the executable compiled with this test driver you get 0 points).

Please turn-off any debugging code or any printf statements in the files getFP.s and framecount.c before you submit your lab (you can leave these in your test driver lab9.c).