Lab Time_____ Name_____

Lab 3. Subroutines in ARM Assembly

Preparation

You will need a LaunchPad and a laptop/computer with Keil uVision5 installed. Download the Lab3 starter project from Canvas.

Starter project Lab3

Purpose

The purpose of this lab is for you to become familiar with writing your own subroutines and making nested subroutine calls in ARM Assembly. You will also gain experience with passing parameters to subroutines and returning values from subroutines.

Introduction

Subroutines in ARM assembly are analogous to functions in higher level languages such as C. According the ARM Architecture Procedure Call Standard, parameters are passed to subroutines in R0-R3, starting with R0. The stack is used if the subroutine requires more than 4 input parameters. If there is a resulting value to be returned, then the subroutine returns it in R0. Subroutines can also assume they are freely able to modify R0-R3 and R12. This means that any important value located in one of those registers runs the risk of being crunched/overwritten when a subroutine is called. If the subroutine itself needs to use additional registers then it must push the current values of those registers onto the stack before they are used and then restore those values at the end before the subroutine returns. Subroutines that fail to follow the standard are at risk of breaking the program that calls them. **This lab is coded in such a way that it will be difficult, if not impossible to successfully complete without following this standard.**

Procedure

1. Using the same process as in Lab 2, create a 25 ms delay subroutine in "delay.s". Prove your delay subroutine is correct by calling it from "main.s" to toggle your choice of LED every 25 ms. You will need to export the name of your subroutine at the top of "delay.s" and then import it at the top of "main.s". Use the oscilloscope to verify that the LED is toggling every 25 ms. Demonstrate this to the TA.

Signature Cah Terrell Date 2/18

2. Now create a variable delay subroutine in "delay.s" that is capable of generating delays that are multiples of 25 ms. The multiple will be the value passed to your subroutine in R0. Use this subroutine to toggle an LED every 500 ms. This subroutine should call your 25 ms delay subroutine from Part 1. Use the oscilloscope to verify that the LED is toggling every 500 ms. Demonstrate this to the TA.

Signature Matthew Russell Date 2/18/2020

Lab Time_____ Name_____

3. Write a subroutine "exp" in "subroutine.s" that takes two input values A and B, adds them together, raises their sum to the power of the first input value A, and returns the result. You can assume the value of A will be greater than 0 in order to simplify your code. The inputs A and B should be passed in R0 and R1, respectively, while the result should be returned in R0. The section of code in "main.s" titled "exptest" will be used by the TA to verify your subroutine. The Green LED will turn on if the subroutine is correct or the Red LED will turn on if the subroutine is incorrect. **Be sure your function follows the Procedure Call Standard or else you might crunch some of the registers needed by "exptest".** You might find it useful to feed your own inputs to the subroutine and then check the results in the debugger. An example of this is provided in the project itself. Demonstrate your correct subroutine to the TA.

Signature Cole Farrell Date 2/18

4. Write a subroutine "mulsum4" in "subroutine.s" that will perform the operation Z=((A+B+C)*(B+C+D)*2) where A, B, C, and D are values passed to the subroutine in R0-R3, respectively, and Z is the result returned in R0. Your subroutine should call the provided subroutine "sum3" which sums three values. You should also write and use a similar subroutine called "mul3" which multiplies three input values. **You will need to be VERY careful not to crunch any current register values when you call "sum3" and "mul3".** The section of code in "main.s" titled "mulsum4test" will be used by the TA to verify your subroutine. The Blue LED will turn on if the subroutine is correct or the Red LED will turn on if the subroutine is incorrect. As with the last part, you might find it useful to feed your own inputs to the subroutine and then check the results in the debugger. Demonstrate your correct subroutine to the TA.

Signature Cole Farrell Date 2/18