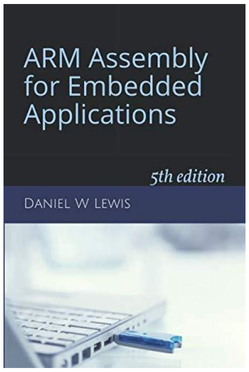


Programming Lab #2a

Functions and Parameters

Topics: Passing parameters, function return values, nested functions, preserving and restoring registers across function calls, calling C functions from assembly.

Prerequisite Reading: Chapters 1-3
Revised: August 28, 2019

Create an assembly language source code file containing four functions. If the functions were written in C, they would look like the following. (Functions `Square` and `SquareRoot` are provided in the main program.)

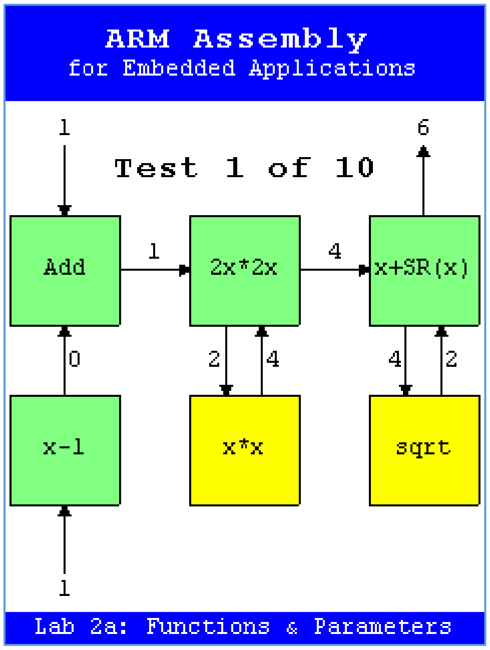
`int32_t Add(int32_t a, int32_t b)`
`{`
`return a + b ;`
`}`
	`int32_t Less1(int32_t a)`
	`{`
	`return a - 1 ;`
	`}`

`int32_t Square2x(int32_t x)`
`{`
`return Square(x + x) ;`
`}`
	`int32_t Last(int32_t x)`
	`{`
	`return x + SquareRoot(x) ;`
	`}`

Test your functions using the main program downloaded from here. If you code works correctly, the display should look like the image on the right. Press the blue pushbutton to cycle through all the test cases to verify that everything is correct. Color is used to indicate the status of a function:

Gray	Function never called.
Yellow	Function given in main program.
Orange	Function never returns.
Green	Function returning correct value.
Red	Function returning incorrect value.

IMPORTANT – The `.thumb_func` directive: The "`.thumb_func`" assembler directive specifies that the <u>next</u> label is the entry point of a function that contains instructions from the Thumb subset of the ARM processor and causes the binary representation of instructions that branch to that label to be generated somewhat differently. Thus in a source code file that contains more than one function, it is imperative that you place a `.thumb_func` directive immediately before the entry point label of <u>every</u> function.


ARM Assembly
for Embedded Applications
Test 1 of 10
1
6
Add
1
2x*2x
4
x+SR(x)
0
2
4
4
2
x-1
x*x
sqrt
1
Lab 2a: Functions & Parameters