Lab 6 Subroutines

Submit lab6.asm at the end of your lab.

I. Subroutines

In first year programming language courses such as csc 110, csc 111, we wrote many functions. In assembly language, the term procedure, function, or subroutine can be used interchangeably. A function call implies a transfer (branch, jump) to an address representing the entry point of the function, causing execution of the body of the function. When the function is finished, another transfer occurs to resume execution at the statement following the call. The first transfer is the function call (for invocation), the second transfer is the return. Together, this constitutes the processor's call-return mechanism.

Example: passing two parameters by values. Download **"params.asm"** (kindly provided by Jason) The diagram of the internal memory of the SRAM when "lds ZH, Z+9" is executed:

Address	content	details	notes
0x0000 ~ 0x01FF		General Purpose Registers and I/O Registers	
0x0200			.DSEG
	……..		
		<Z-and SP	
	r1	saved register	Programmer pushes those registers onto the stack in the subroutine
	r0	saved register	
	r31	saved register	
	r30	saved register	
	ret	return address	Assembler pushes the return address onto the stack automatically when "call do_something" is executed.
	ret	return address	
	ret	return address	
	0xCC	parameter (Z + 8)	Programmer pushes the values onto the stack in the "main" in the example.
0x21FF	0xEE	parameter (Z + 9)	

Build params.asm and run the code, observe the contents of the registers SP, Registers Z, r16, r1 and r0.

II. Exercises: download lab6.asm, read void strcpy (src, dest) subroutine. Understand it, reconstruct the stack frame. Implement unsigned int strlength (str) subroutine using the two subroutines we have just learnt as examples.

Read the code in lab6.asm. Write on a piece of paper the stack frame for strcpy.

Address	content	details	notes
	……..		
		< SP	
		……	Programmer pushes those registers onto the stack in the subroutine
	ret	return address	Assembler pushes the return address onto the stack automatically when “call do_something” is executed.
	ret	return address	
	ret	return address	
	…..		Programmer pushes the parameters (could be values and/or addresses) onto the stack in the “main” in the example.
0x21FF			

Design the stack frame for strlength

Address	content	details	notes
	……..		
		< SP	
		……	Programmer pushes those registers onto the stack in the subroutine
	ret	return address	Assembler pushes the return address onto the stack automatically when “call do_something” is executed.
	ret	return address	
	ret	return address	
	…..		Programmer pushes the parameters (could be values and/or addresses) onto the stack in the “main” in the example.
0x21FF			

Submit lab6.asm at the end of your lab.