Ense 352 - Fall 2013 - Lab 4

Created by: Diego Castro-Hernandez – Electronics Systems
University of Regina

Hand out: week of Mon 2013-11-04
Due: At the beginning of the next lab period (Week of Nov 18th)

1. Objective:
   The objective of this lab is to implement recursive subroutines in ARM assembly language to evaluate the following concepts: subroutine calls, parameter passing, return from subroutines, stack overflow, working with strings of characters.

Sorting arrays of characters

Consider the following algorithm for sorting characters in an array:
Input array:

E	**B**	**F**	**Z**	**A**	**C**	**G**	**L**	**D**	**A**

Steps:

1. Generate two sublists by dividing input array approximately in half

Sublist1					Sublist2				
E	**B**	**F**	**Z**	**A**	**C**	**G**	**L**	**D**	**A**

2. Recursively sort each "sublist". This means subdivide each sublist in half and sort the resulting arrays recursively.
   After this step, sublist1 and sublist2 should look like this:

Sublist1					Sublist2				
A	**B**	**E**	**F**	**Z**	**A**	**C**	**D**	**G**	**L**

3. Merge the two sorted sublists to get the final sorted array
   Output array:

A	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**L**	**Z**

Example showing the process step by step:

- Divide input array

Sublist1 Sublist2

E	B	F	Z	A	C	G	L	D	A

- Subdivide sublist1 in half and keep doing this recursively until getting sublists of just 2 characters

Sublist1_A Sublist1_B Sublist2

E	B	F	Z	A	C	G	L	D	A

- Sublist1_B just have to characters, we can just sort it by switching Z and A. However, Sublist1_A has three characters, we can still subdivide it in two:

Sublist1_B Sublist2

E	B

F

A	Z

C	G	L	D	A

Sort each sublist obtained after subdividing

B	E

F

A	Z

C	G	L	D	A

- Merging the sublists of Sublist1_A to generate sorted version of Sublist1_A

Sublist1_A Sublist1_B Sublist2

B	E	F

A	Z

C	G	L	D	A

- Now Sublist1_A and Sublist1_B are sorted, so we can just merge them to generate sorted version of Sublist1.

Sublist1 Sublist2

A	B	E	F	Z

C	G	L	D	A

- Repeat process for Sublist2 and merge with Sublist1

Sublist1 Sublist2

A	B	E	F	Z

A	C	D	G	L

Final array:

A	A	B	C	D	E	F	G	L	Z

This algorithm is known as “Merge Sort” (more info: http://en.wikipedia.org/wiki/Merge_sort). It can be implemented iteratively, but in this lab we are interested in its recursive implementation. The pseudocode is presented next:

```
;Function “sort” takes as inputs the pointer to the array as well as its size
;It returns the ptr to the sorted array
sort(ptr_input_array, size_input_array){

    if (size_input_array = 1)
        ;do not modify input array
        return ptr to sorted array
    elseif (size_input_array = 2)
        ;sort both elements
        return ptr to sorted array
else
    ;divide input array in half and sort each half separately
    ptr_sublist1 = ptr_input_array
    size_sublist1 = size_input_array / 2
    ;Recursive call:
    sort(ptr_sublist1,  size_sublist1)

    ptr_sublist2 = ptr_input_array + size_sublist1
    size_sublist2 = size_input_array / 2
    ;Recursive call:
    sort(ptr_sublist2,  size_sublist2)

    ptr_sorted_array = merge(ptr_sublist1 , ptr_sublist2, size_input_array )
    return ptr_sorted_array

}
```

Note: The previous pseudocode assumes the input string has an even number of characters, your subroutine must accept strings of any size.

Procedure:

Write a subroutine that will sort in ascending order a string of characters using the algorithm described before, it should take as inputs a pointer to a buffer containing the string of character to be sorted and the size of the string. It then modifies the buffer to contain the sorted version of the string.

Furthermore, the function will use R10 as an error register. In case of an error (for example stack overflow) the subroutine must return a 1 in this register, else it must return 0. The subroutine must not modify any register.

You do not need to implement the subroutine “merge”, it is already given in the file lab4_sort.s uploaded to URCourses. That subroutine merges two “adjacent” sublists sorted in ascending order. Read its header to understand how to pass the parameters to this subroutine.

Download the file lab4_sort.s and include the code of your subroutine in the corresponding section.