Electrical and Computer Engineering, Purdue University Northwest
Computer Organization and Design (ECE 371), Spring 2019
ARM Hands-on Assignment, Due Date: May 3, 2019 (Hard deadline)

Objective: Get familiar with ARM assembly programming.

Note: This assignment has a weightage of two homework assignments.

Hands-on Set-up

VMware installation.

a) Download VMWare Player
Mac: https://www.vmware.com/products/fusion/fusion-evaluation.html
Windows: https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html

b) You may refer https://www.youtube.com/watch?v=NcesmPiwY44 for installation in Windows machine or https://www.vmware.com/pdf/desktop/fusion-getting-started-50.pdf for Mac book. Check VMware Fusion installation in the document.

Import Azeria Labs VM

a) Download VM image from https://drive.google.com/file/d/1uX9fRUX-IHitQVD43QUsz9Aocy8pJ8XK/view?usp=sharing. You can find the details of the VM image at https://azeria-labs.com/arm-lab-vm/. Check Level 2 in the link for the VM information.

b) Extract the downloaded VM zip file in a folder.

c) Open VMware Player. From File menu, select Open. Browse for the folder extracted in Step b. Select **.vmx** file from this folder to import the VM. You may start the Guest VM from VM tab. Password for the VM is **azerialabs**

d) You will find the access information for Raspberry pi emulator in the wallpaper of the VM.

Writing programs inside Raspberry Pi emulator

Follow the steps mentioned below for each problem. I have used Problem 1 for illustration.

a) Type command **cd**. This command will always bring you in the home directory.

b) Create a folder for the problem using command **mkdir**. Example: **mkdir problem1**

c) Go to the directory by typing **cd problem1.** This command will bring you in **problem1** directory.

d) Use **nano** editor to create and edit C and assembly files. For example, type **nano prog1.c** to create or edit **prog1.c**. To save the file, type **ctrl+o** and to exit the editor, type **ctrl+x**. VMware allows text copying from a host machine file to guest machine file and vice-versa. You may copy the text from an opened C file in your host machine and paste it in nano editor using **ctrl+shift+v**. You have to write the assembly file from scratch. You may use the assembly template provided with the assignment.

e) Once you are done with coding, then use the commands mentioned below for compiling the files.

For each programming problem, you have already given a C file. The file invokes an external function that you have to write in ARM assembly. We have provided an assembly file for **Problem 0** as template. The C and assembly files are listed below for **Problem 0**.

```
#include<stdio.h>
extern int sum(int a, int b);
int main(int argc, char *argv[]){
    int a, b;
    a = 10;
    b = 20;
    printf("Sum of %d and %d is %d\n",a, b, sum(a,b));
    return 0;
}
```

Listing 2: prog0.c

```
        .text
        .global sum
        .type   sum, %function

sum:    ADD R0, R0, R1
        BX LR
```

Listing 1: fun0.s

Commands for compiling the files:

```
gcc -g -c prog0.c -o prog0.o
gcc -g -c fun0.s -o fun0.o
gcc -g prog0.o fun0.o -o  prog0
```

These commands will create an executable file **prog0**. You can run this file by typing **./prog0**
You may follow the compilation and execution steps for each problem.

Programming Problems

Problem 1 [**5 points**]: Use **prog1.c** file for this programming problem. The program calculates the sum of **n** natural numbers by invoking **sumnat (int n)** function. Write the assembly code for this function.

Problem 2 [5 points]: Use **prog2.c** file for this programming problem. The program swaps the values stored in two variables by invoking function **swap (int *a, int *b)**. Write the assembly code for the function.

Problem 3 [10 points]: Use **prog3.c** file for this programming problem. The program performs binary search in a list for a given search item by invoking function **binsrch (int data[], int item, int low, int high)**. The list should be sorted for binary search. Write the assembly code for the function. You may find the detail for binary search here:
https://en.wikipedia.org/wiki/Binary_search_algorithm

Problem 4 [10 points]: Use **prog4.c** file for this programming problem. The program sorts a given list by invoking function **sort (int data[], int n)**. Implement selection sort algorithm in assembly for the function. You may find the details for selection sort here:
https://en.wikipedia.org/wiki/Selection_sort

Problem 5 [10 points]: Use **prog5.c** file for this programming problem. The program checks whether two given strings are reverse of each other by invoking function **strrev (char src1[], char src2[])**. Write the assembly code for the function.

Programming Tips

- ARM programming convention: Register R0-R3 are used for function arguments. Return value is stored in R0. R4-R11 stores saved variables inside a function.
- Use LDR and STR to access an integer from memory location and increment memory index by 4 for the next integer access.
- Use LDRB and STRB to access a character from memory location and increment memory index by 1 for the next character access.
- Use GDB to debug your assembly code. https://azeria-labs.com/debugging-with-gdb-introduction/.

What to submit

You need to submit assembly code and screenshots for all the problems in one file. Also, separately submit the assembly files in one zip file.