CS2208 Assignment 4

Issued on Sunday, October 25, 2020

Due by: 11:55 pm on Monday, November 09, 2020

For this assignment, only an electronic submission (***attachments***) at owl.uwo.ca is required.

- Attachments must include:
  - ***ONE pdf*** file (named **`report1.pdf`**) that has a flowchart for question 1.
  - ***ONE Text*** file (named **`question1.s`**) that has the softcopy of your assembly source program.
- So, in total, you will submit 1 + 1 = 2 files (**`report1.pdf`** and **`question1.s`**)
- **Failure to follow the above format may cost you 10% of the total assignment mark.**

Late assignments are strongly discouraged

- 10% will be deducted from a late assignment (up to 24 hours after the due date/time)
- After 24 hours from the due date/time, late assignments will receive a zero grade.

In this assignment, you will use the *micro Vision ARM simulator* by *Keil* to develop the required programs in this assignment.

Programming Style

The programming style is very important in assembly language.
It is expected to do the following in your programs:

- Using EQU directive to give a symbolic name to a numeric constant to make it more readable.
- Applying neat spacing and code organization:
  - Assembly language source code should be arranged in three columns: *label*, *instruction*, and *comments*:
    - the *label* field starts at the beginning of the line,
    - the *instruction* field (opcodes + operands) starts at the next TAB stop, and
    - the *comments* are aligned in a column on the right.
- Using appropriate label names.
- Commenting *each* assembly line
- Commenting *each* logical part of your code.

Great Ways to Lose Marks

- Not appropriately using whitespace to lineup your program
- Not appropriately using EQU to make your code more readable
- Not appropriately using labels to make your code more readable
- Not bothering to comment your code
- Commenting the code by just stating what you're doing, instead of why, e.g.,

```
MOV r0, #5    ;move 5 into r0
```

- Not grouping your lines into logical ideas
- Not paying attention to the programming style (see the previous paragraph)
- Not optimizing your code by using unnecessary assembly instructions. The more instructions in your program the less your mark will be.
- Handing in your code as soon as it assembles, without testing and validating your code
- Not using proper flowchart symbols
- Not following the flowchart rules

QUESTION 1 (20 marks)

Draw a *detailed flowchart* and write an ARM assembly program to determine whether a string of ***printable*** ASCII letters (from a to z or from A to Z, case insensitive) stored in memory is a palindrome (i.e., the letters in the string are the same from left to right as from right to left, case insensitive) or not. If palindrome, you should store 1 in `r0`, if not, you store 2 in `r0`. **Your code should be highly optimized. Use as few instructions as possible (as little as 21 assembly instructions only NOT including any assembly directives or data definitions)!!.**

Ignore all *characters* that are ***not letters***. You should also treat capital and small letters the same, i.e., case insensitive. For example, "*madam*", "*deleveled*", "*Noon*", "*He lived as a devil, eh?*", and "*Was it a car or a cat I saw?*" are palindrome strings. However, "*madam, I am Adam.*" is not a palindrome string.

A string can have an *even* or *odd* number of characters and must end with character **`0x00`** (i.e., the ASCII code of the null character).

Define in your assembly program the string as follow:

```
STRING DCB "He lived as a devil, eh?" ;string
EoS    DCB 0x00                       ;end of string
```

Make sure to test your program with many cases, where some of them are palindrome, and some are not. Make sure that your test cases will cover each branch of your flowchart.