Quiz3 ECE2560 Sp2022

Collaboration with other students is not allowed

Write a **subroutine** named **MyDiv** which satisfies the following contract:

```
;-------------------------------------------------------------------------
;                   Subroutine: MyDiv
;                       -----------------
;         Input: R12 Word (Number to divide); R12 will be modified
;         Input: R13 Word (Log2 of the Number to divide by); R13 will not be modified
;         Output: R14 : R12/(2^R13),fractional part will be discarded; R14 will be modified
;-------------------------------------------------------------------------
```

This subroutine uses **bit shifting** to divide a number stored in R12 by another number that can be written as 2 to the power of an integer, i.e., the number to divide by is 2^R13. In other words, R13 = Log2(number to divide by). So, if you want to divide by 8 = 2^3, then you will place 3 in R13. Note that 3 = Log2(8)

Use R11 as the index to the loop in the subroutine you will need to do the bit shifting. Make sure to preserve the register R11 on the stack, as explained in Screencast14. Do not use any variables stored in the .text or .data regions inside the subroutine. Only use registers R11, R12, R13 and R14 inside the subroutine.

Write a **main program** which calls this subroutine twice. Make space for two word sized variables in the .data region named **result1** and **result2**. Call the subroutine the first time to divide **880** by **4**, store the result in **result1**. Call the subroutine a second time to divide **3520** by **8** and store the result in **result2**. There is no need for pseudo-code or flow chart in this quiz. Also, there is no need to write your own contract for the subroutine, simply include the contract given above, at the beginning of your subroutine. It is mandatory to watch Screencast13 & 14 to understand subroutines. Also, look at the posted files, BitDivisionSample and BitDivisionSample_sol, as examples of writing code to divide by using bit shifting.

Instructions: Use the word template and instructions contained on our web site to submit your screenshots to Carmen. Do not email directly to your TA or me. Files emailed to the TA or me will not be accepted. Include the following in your submission:

i) Assembly language source code of the whole program, including the **main** program and the **subroutine**. The code should be in **Word format** so that the TAs can run it in CCS.
ii) Screenshot of the "General" tab of the properties screen of your project.
iii) Screenshots of the memory browser showing the variables **result1** and **result2** after the program has been executed/resumed (by pressing the green play button in the debugger):

and paused/suspended (by pressing the pause button in the debugger):

Use "16-bit Signed Int" format in the memory browser. For your own debugging purposes you may want to use "step over" and "step into" in the debugger to observe the values in the Core Registers (in the registers tab) as explained in the screencasts mentioned above (use Signed Int format to observe the values in the core registers as signed integers).