Hw5.2

Homework Assignment 5.2

Due: Mon, 28 Feb 2022 17:30:30 (approximately 2 days from the time this page was loaded)
[13 points possible] [100 penalties possible] [0 penalties graded so far]
Weight = 1.00

This homework is a review of programming techniques for GPIO, Interrupt handling, and basic recursion. To complete this homework, you should create a project in System Workbench with the Standard Peripheral firmware. Write the following subroutines, and put them all in the same file named hw52.c. You should use the SystemWorkbench template file, but remove the **`main()`** subroutine from it. The result should look like this:

```
        #include "stm32f0xx.h"

        // That's it.  The #include is all you need to start.
```

To test your subroutines, you can link hw52.c against the test.c file (that can be found at some point) on the homework web page. The test module will exercise these subroutines and check their results and effects.

The problems are exactly the same as hw5. You already know how to do them! You're simply going to practice doing them with the C language instead of assembly language. (Question 1 is pretty easy, yes?) You now have a deep understanding for exactly what is happening when you write C statements.

Repeat: This is just a copy-and-paste of homework 5. All the places below that talk about assembly language are what you did for homework 5. Now you will do the same thing in C. Yes, quesion 1 is obviously easy.

The test.c file provides many good examples of how to manipulate configuration registers using the C language. Almost everything we do in the class after the midterm lab practical exam will be done using C. We find it takes a little effort to get used to working with pointers and structure fields to do peripheral configuration. This homework assignment will get you ready for lab 6.

Q1: recur [1 point]

Translate ("hand compile") the following C function into assembly language.

```
// Just another strange recursive function.
unsigned int recur(unsigned int x) {
        if (x < 3)
                return x;
        if ((x & 0xf) == 0)
                return 1 + recur(x - 1);
        return recur(x >> 1) + 2;
}
```

Q2: enable_portb [1 point]

Write an assembly language subroutine named **enable_portb** that configures the RCC to enable the clock to GPIO Port B but leaves the other clock control bits as they were.

Q3: enable_portc [1 point]

Write an assembly language subroutine named **enable_portc** that configures the RCC to enable the clock to GPIO Port C but leaves the other clock control bits as they were.

Q4: setup_pb3 [1 point]

Write an assembly language subroutine named **setup_pb3** that configures pin PB3 for the following:

- Input
- Pull-down resistor enabled

It should not change any other configuration for any other pins.

Q5: setup_pb4 [1 point]

Write an assembly language subroutine named **setup_pb4** that configures pin PB4 for the following:

- Input
- Neither pull-down nor pull-up resistor enabled

It should not change any other configuration for any other pins.

Q6: setup_pc8 [1 point]

Write an assembly language subroutine named **setup_pc8** that configures pin PC8 for the following:

- Output
- Output Speed: High Speed

It should not change any other configuration for any other pins.