Carleton UNIVERSITY

Department of Systems and Computer Engineering

SYSC-3310: Laboratory 5

Basic I/O using Interrupts

As this lab needs the physical board, this will be a group lab. Each group should submit one deliverable set mentioning the group number and students' names/IDs. Any student in the group can submit the deliverable but make sure to write all student names of the group.

The lab deliverable is ***two-weeks*** *due from your lab session date.*

Lab 5
Basic I/O with interrupts

This lab assignment is identical to the previous one, except switch interrupts should be used, rather than polling. In this lab, you will exercise how to configure interrupts and implement ISRs.

Things to look out for:

- Configuring switch interrupts correctly.
- Actuating on the LEDs correctly and having all the required and correct output configurations in place.
- Performing the bitwise operations correctly (i.e., performing only the required operations without touching any unwarranted bits).
- Obeying correct C coding guidelines and required steps (e.g., disabling the Watchdog Timer, proper file inclusion, not using global variables needlessly, etc).
- Implementing switch debouncing correctly.
- Implementing switch ISR correctly.

Follow the steps explained in the previous lab to create a project.

Include file "msp.h" – this gives you the peripheral devices register definitions we talked about. This will be required in every project.

```
#include "msp.h"
```

In this project, you should:

1 Disable the Watchdog timer!
2 Configure the switches (P1.1 and P1.4) as inputs, using pull-up internal resistors.
3 Configure switch interrupts. This has to be done at port level (device),
4 NVIC, and CPU (consult lecture slides and Reference Manual)
5 Configure the LEDs (P1.0 and P2.0, P2.1, P2.2) as outputs. 5 Initialize LEDs states (all turned off).
6 Create an infinite loop ("while (1)") after your configurations, that does nothing (code will loop forever, handling interrupts)
7 Implement the required ISR(s) (without debouncing at first, then with debouncing) such that.
  - When Button 1 is pressed, the selected LED changes (default, RED LED).
  - When Button 2 is pressed, the current LED changes state.

For the RED LED, there are only two states: on and off. **This should be performed through bitwise toggling.**

For the RGB LED, there are 8 states, corresponding to off and 7 different colors. **This should be performed through bitwise clearing and setting.**

Remember the ISR must clear all required interrupt flags! Global variables should not be used needlessly: **"static" is a very useful keyword.**

In other words, one button changes between the RED LED and the RGB LED: the other button changes the current state of the current LED, allowing us to turn the RED LED on and off and cycle through all the colors of the RGB LED

Deliverables

To get the marks for this lab, submit the following:

1) Your project folder in a single zip file. Name your project zip file like this: **Lab5Proj_<Group_number>.zip**
   Your zip folder MUST include the entire Keil project folder
2) A ***video*** that shows a DEMO of your project testing on the board.
3) A single page that includes
   a. Half page summary of what you have done and learned from this lab
   b. Your group number and students' names/IDs