Lab 3 – User Input and LCD

1 INTRODUCTION

In this lab you will develop a PSoC Creator project to write messages to the LCD, and update the LCD based on user pushbutton usage.

The LCD can be useful for debugging, and for application development.

2 PART 1, PROCEDURE

Add a "Character LCD" device to your project. Your main.c program will consist of an initialization part, and an infinite loop. In the initialization part put any component initialization procedure calls that are necessary.

Add a digital input pin, and map its pin to one of the pushbuttons on the board – remember the pullup resistor!

Add to the initialization part code to display the following message:

```
My First Microcontroller I/O Console Program
Microcontroller Programmer
Press a switch to continue...
```

Note that the character display size is very limited, and you'll have to come up with a strategy for effectively displaying this information to a user!

Write code in the infinite loop that waits for a button press, and then updates the display to:

```
Program completion succeeded
```

At this point the program will just stay in an infinite loop.

Build your project and download it to the development board. Keep notes as you debug your project, and include your debugging notes in your report. Take a photo or video of your LCD display screens for inclusion in your report.

Demonstrate your working project to your instructor.

3 PART 2, PROCEDURE

Create a project to meet the following requirements:

1) Upon program START, the system will turn on two LED's.
2) The system will display a greeting message of your choosing.

3) The system will display a prompt message asking the user to push either pushbutton.
4) The system will enter an infinite loop where:
    a. It waits for an event where either button is down
    b. It toggles the state of the associated LED
    c. It displays the following message:

```
Button N count
in hex: HHHHHHHH
```

    where **N** is either 1 or 2, depending on the button just pressed, and **HHHHHHHH** represents the count of button presses, in hexadecimal, with the letters A-F being displayed in upper case. You should use the 'C' function **sprintf()** to generate the formatted strings, and force leading zeros to be included.
    d. Button de-bounce delay
    e. It waits for the state where both buttons are up
    f. Button de-bounce delay
    g. Wash-rinse-repeat
5) Read the info below on switch debouncing! Try to capture the switch bouncing phenomenon on the scope. To do this:
    a. Set your scope probe to 10x mode (to reduce tip capacitance).
    b. Set the time sweep to 500 ns/div
    c. Set the scope trigger subsystem to single, and set for falling edge trigger, trigger level to about $V_{DD} - 0.5$ Volts.

4 NOTES

1) You will need to initialize your LCD component.
2) You will need to map the pins of your LCD component to the physical chip pins that are wired to the LCD on the board. You can find the port/pins that are connected to the LCD by reading the Development Kit Guide.
3) You will need to manage the display of lots of characters on the limited 16x2 line character display. You can use the horizontal scrolling procedure included in the API, or work your own vertical scrolling, etc., using time delays to make the display pleasant for humans.
4) **You will need a pullup resistor on your pushbutton line** – this can be accomplished as a configuration of the digital input pin component.
5) **IMPORTANT AND NOT OBVIOUS**: A pushbutton has a very long "bounce" – meaning that each time the button goes up or down, the PSoC sees an erratic stream of 0's and 1's, until ultimately settling on the new state. The bounce period is generally less than 100 ms. Your software must take this into account, so that the toggling of your displayed message and LED state only happens once for each button press. There are many ways that you can "debounce" the button – I expect you to do the following:
    a. Use a while () {} idle loop to wait for the "button down"
    b. Handle button down
    c. Delay 100 ms

d. Use a while () {} idle loop to wait for the “button up”
e. Delay 100 ms, before waiting for button down, or doing anything else

Note that you’ll also need to create an infinite loop around this whole process.

To capture pin bounce on a scope set the trigger for “Single”, the trigger level for about 1 Volt, and a rising edge trigger. Set the horizontal scale for about 200 μsec/div. Put the scope probe on the “high” side of the switch leads (the side that normally reads V_{DD} when switch is up). Arm the Single trigger. Press button down. Let button up. You may find a bounce on either edge, or you may need to repeat a number of times to get a good trace like the one in Figure 1.

Figure 1: Scope Trace Showing Bounce from Button Release

5 REFERENCES

- CY8CKIT-050 PSoC® 5LP Development Kit Guide

6 WRITE A REPORT

The report is to include, but not limited to the following:

a) **(0.5 pts) Cover Sheet** with Title, Class, Names, etc.
   ALSO – on the Cover Sheet, indicate %contribution of each student to the
   - **i) Code Development**
   - **ii) Lab work and debug**
   - **iii) Report**

b) **(0.5 pts) Introduction**.

c) **(0.5 pts)** Brief recap of **Procedure**. Include any equations or other relevant information that helps you to explain what, why and how you did what you did.

d) **Results**
   a. **(1 pts)** Insert a movie, or series of snapshots depicting the behavior of your LCD.
   b. **(0.5 pts)** Scope trace showing the bouncing phenomenon, volts/div and usec/div must be legible, or otherwise indicated clearly.
   c. **(0.5 pts)** Debug experiences.

e) **(2 pts) Discussion** of considerations involved in driving the LCD, and in the pushbutton debounce issue. Explain why you need to "debounce" both button down and button up. Consider Googling "switch debounce" and doing a little research on the subject.

f) **(0.5 pts)** A **Conclusion** of your results and discussion of anything you found especially interesting or not expected from your work on this project.

g) **(2 pts)** Check in your code, properly formatted and fully commented.

h) **(2 pts)** Overall professionalism of code and report.

REPORT NOTES:

- One report per team
- You may use the IEEE paper format, if you like: (template is 2014_04_msw_usltr_format.doc). In this case the cover sheet info is embedded at the top.
- Microsoft Word, .docx file.
- Upload via Canvas (**one upload per team**).
- Also upload all code files that you wrote (and **only** those files!), via Canvas (**one upload per team**).