1) From mycourses download this pdf, dxp_Lab5_c1.c, and dxp_Lab5_a1.asm.
2) The **objective** of this lab is to learn how to configure and usedigital output ports to drive a multiplexed LED display pattern.
3) **Part 1: Build, run and understand the dxp_Lab5_a1.asm code.**
4) Create a new assembly only project in your workspace named fmlxxxx_Lab5_a1. Create a copy of the assembly code dxp_Lab5_a1.asm in the project folder, and rename it fmlxxxx_Lab5_a1.asm. Build the project.
5) Install the capacitive touch booster board (CTB) onto the Launch Pad board. Install before connecting it to the computer, i.e. as long as it is not powered on. Make sure the ti logos have the same orientation.
6) Enter debug mode.
7) Analyze the operation of the program and document your observations in your report. To help in understanding the code, refer also to the CTB user's guide pdf document on myCourses. Add any additional comments you feel help prove your understanding to the assembly source code.
8) **Part 2: Modifying the fmlxxxx_Lab5_a1.asm code to fmlxxxx_Lab5_a2.asm.**
9) In the code provided, the LEDs 1-4 are turned on and off through the use of explicit/exhaustive code.
10) A more elegant/concise mode is to use a loop, which is how the LEDs 5-8 are turned on and off.
11) Create a new assembly only project in your workspace named fmlxxxx_Lab5_a2. Create a copy of fmlxxxx_Lab5_a1.asm, and rename it fmlxxxx_Lab5_a2.asm.
12) Modify the code so that it displays counter clockwise. You are also required to use loops for both sets of LEDs. Only 1 LED should be on at any given point, unlike the original code.
13) Build, debug, and run the program on the board.
14) What happens if you decrease the delay value to 0x0fff? How about 0x00ff? Mention this in your report.
15) Also, notice that we are driving 8 LEDs using only 5 pins. This is because we are multiplexing the outputs, i.e. we are turning them on and off in a sequence. If we do this fast enough, it will look as if we have a continuous display.
16) Section 3.4.4 "Instruction cycles and lengths" of the UG contains information about how many cycles each instruction takes to execute. Knowing the instructions executed in the loop, and how many cycles each take to execute, compute the delay time for each of the above three cases (0x7fff, 0x0fff, 0x00ff). Once you have calculated these delay values, check them against an oscilloscope reading. You can probe the output pin that drives one of the LEDs, and measure how long it is ON for. Demonstrate this to your TA, and include your calculated value vs your measured value in your report.
17) **Part 3: Build, run and understand dxp_Lab5_c1.c.**
18) Create a new project in your workspace named fmlxxxx_Lab5_c1. Create a

copy of dxp_Lab5_c1.c, and rename it fmlxxxx_Lab5_c1.c.

19) Build, debug, and run the project. Analyze the operation of the program and document your observations in your report. Add any additional comments you feel help prove your understanding to the source code.

20) **<u>Part 4:</u> Modifying the fmlxxxx_Lab5_c1.asm code to fmlxxxx_Lab5_c2.asm.**

21) In the code provided, the LEDs 1-4 are turned on and off through the use of explicit/exhaustive code.

22) A more elegant/concise mode is to use a loop, which is how the LEDs 5-8 are turned on and off.

23) Create a new project in your workspace named fmlxxxx_Lab5_c2. Create a copy of fmlxxxx_Lab5_c1.c, and rename it fmlxxxx_Lab5_c2.c.

24) Modify the code so that it displays counter clockwise. You are also required to use loops for both sets of LEDs. Only 1 LED should be on at any given point, unlike the original code.

25) Build, debug, and run the program on the board.

26) What happens if you decrease the delay value to 0x0fff? How about 0x00ff?

27) Also, notice that we are driving again 8 LEDs using only 5 pins. This is because we are multiplexing the outputs, i.e. we are turning them on and off in a sequence. If we do this fast enough, it will look as if we have a continuous display.

28) For the code written in C, to exactly calculate the delay for a given count value, we need to look at the disassembled code and determine the instructions used in the delay loop.

29) Because you have gone through this exercise already, let's try something else. Eliminate from the delay loop one of the decrement statements. Check the disassembled code. Try stepping through the disassembled code one instruction at a time. Describe what you discover. We'll talk about it in the lecture.

30) Make sure you write the report and upload it along with your project archives on mycourses. As time permits, demo the intermediate steps to the TA.

31) Grading:

    a. Part 1 = 10 points
    b. Part 2 = 10 points
    c. Part 3 = 10 points
    d. Part 4 = 10 points.