ME 430 Exam 2, Fall 2017 - 2018, All Sections

Name_____________________________________ Class periods _________

You may use only:

- Any paper notes (including course handouts) you brought to the exam, or electronic notes residing on your local (C:) hard drive.
- The course website, including any code from the website. This is the only approved use of the internet for this exam.
- Code written by you or by your lab partner(s).
- Pencil/pen and a calculator (optional).
- The green demo board and its accessories.
- Moodle for code submission.

Anything not specifically allowed is prohibited. In particular, you may not use code written by someone outside your lab group unless it came from the course website.

If your code for one of the problems works properly, you should get it checked off. There are points associated with the check off itself. You have 3 total tries for each checkoff.

The only code in the programs should be the code which is necessary to accomplish the task—points will be deducted if there is extra stuff that we need to sort through. At the end of the test submit your .c files to Moodle. You do not need to submit LCD Module.c.

Problem	Points	Check off
1a	/ 15	/3
1b	/ 33	/3
2a	/ 25	/3
2b	/ 15	/3
	/ 88	/ 12
Total	/100	

For all checkoffs, for an individual part or for an entire problem, please use the MPLAB X “Make and Program Device” button to **PROGRAM** your board then remove the PICkit3. The program will continue to run on its own. It will speed checkoffs for everyone if your program is simply ready to go and it lets you move on to the next part while waiting for a checkoff.

Problem 1a – Print on LCD

Start this problem from "**template with interrupts.c**", but rename it to "**Exam2_Lastname_P1.c**". This problem will use the LCD display so perform all necessary steps to use "LCD Module.h" and "LCD Module.c".,

Print "COUNTDOWN:" to the first line of the LCD. Initialize a char variable called **`count`** and assign it a value of 50. Print the value of **`count`** to line 2 of the LCD. We recommend that you make an updateLCD function that, when called, will print the current value of **`count`**. Your display should look like this:

C	O	U	N	T	D	O	W	N	:						
5	0														

You can check this part off now or keep working to check off all of Problem 1 at the same time (your choice).

If you choose to checkoff now, **program** your green board, remove the PICkit3, and call your instructor over to get this part checked off.

Problem 1b – Timer Interrupt

This problem starts from your Problem 1a code.

Using Timer 0 and a 500 kHz clock, setup a timer interrupt that will fire every 0.1s. You are required to use Timer 0 as a **high** priority interrupt. Add the line of code to set the priority explicitly even if that is the default. Use a **1:64 prescaler**.

Clock = **500 kHz**

Timer 0 Prescaler = **1:64**

Parameter passed into WriteTimer0 = ____________________

When the Timer 0 interrupt fires, decrease **`count`** by 1 and update the LCD display. When **`count`** reaches 0 (after five seconds), the countdown is complete and **<u>all eight LEDs (RC0 through RC7) should light up</u>**. Additionally, **`count`** should **stop decreasing** at the end of the countdown and your LCD display should look like this:

C	**O**	**U**	**N**	**T**	**D**	**O**	**W**	**N**	**:**						
0															

Note, in this problem it is acceptable to display leading zeros or not display leading zeros (for 09 to 01 and 00). So you final LCD display might look like this, which is fine:

C	**O**	**U**	**N**	**T**	**D**	**O**	**W**	**N**	**:**						
0	**0**														

Call your instructor over to get this part checked off. When you demo this part make sure you PROGRAM the PIC (don't use the debugger) because we will press the reset button to reset **`count`** to 50. Your instructor will watch your demo **and** look at your answer above. If either is incorrect you will not get checked off.

To avoid guessing at the value **many** times you only get 3 attempts to get the value above correct.

Instructor area	☐ Attempt 1	☐ Attempt 2	☐ Attempt 3

Problem 2a – ADC lights

Start this problem from "**template.c**", but rename it to "**Exam2_Lastname_P2.c**".

Read the horizontal joystick values (analog channel 2) and control RC0 and RC7 as follows.

Voltage	LED state
0 - 2.0 volts	**RC0 is on** and RC7 is off
2.0 - 4.0 volts	RC0 and RC7 are both off
4.0 - 5.0 volts	RC0 is off and **RC7 is on**

What threshold values will you use for 2.0 and 4.0 volts in your code? Record your answer here:

2.0 volts = ______________________

4.0 volts = ______________________

Your program should run forever, updating the LEDs based on the ADC reading of the horizontal joystick.

When you get this part working you can check it off or continue onto Problem 2b. Problem 2b builds onto 2a, so you can do a single checkoff for all of Problem 2.

If you choose to checkoff now, **program** your green board, remove the PICkit3, and call your instructor over.

To avoid guessing at the values **many** times you only get 3 attempts to get the values above correct.

Instructor area	☐ Attempt 1	☐ Attempt 2	☐ Attempt 3

Problem 2b – ADC lights

This problem starts from your Problem 2a code.

If the user is pressing down on the joystick (RA4) while either RC0 or RC7 is lit, then lock in the value of the LED and ignore future ADC readings.

For example, if the joystick is pushed far left RC0 is on as before, but then you press down on the joystick (while still holding to the left), then RC0 will remain on even if the joystick moved. RC0 stays on forever and the program no longer changes the LEDs.

Another example, the joystick is in the middle, the user presses down on the joystick. That press has no impact since neither LED is lit at that time.

Another example, the user presses down on the joystick in the middle then moves the joystick over to the right. Once above 4.0 volts, RC7 comes on. Since the joystick is being held down and RC7 is lit, that causes RC7 to remain on forever even if the joystick moves (note that the press event time doesn't matter, just that state of RA4). RC7 stays on forever in that case and future ADC reading and RA4 readings are meaningless.

When you demo this part (or any part) make sure you **program** the PIC.