Requirements Specification for EELE465 Lab Project 4: Comms with Multiple MSPs to read Keypads and drive an LED/LCD

Academic Dishonesty Clause: It is expected that you, along with your partner, write this code uniquely for this project during this semester. It is acceptable to use code YOU have previously written as examples and guides. It is not acceptable to copy/paste code written by other individuals nor by you during previous semesters or for other classes.

Lab project goal: Configure an MSP as a Master and two MSP's as slaves. The Master will read inputs from a Keypad and control the processes. The slaves will control an LED lightbar and an I2C-compatible LCD (one output per slave).

The keypad will be used to input a user's selection. The selection will be reported to an LCD. If the selection is a letter, the light patterns from project 3 will be displayed on the LED bar. An unlock code will be required to begin the display.

Outcomes:

After this lab you should be able to:

- Program microcontrollers off-board from the development platform.
- Create a master/slave relationship between microcontrollers
- Design embedded systems which use more than one I2C slave.
- Configure and communicate with a I2C-compatiable LCD.

Background:

At this point in the semester, hopefully you have started to gain a real understanding of how code affects the microcontroller at the bit level. Now that we have developed this deeper understanding, it is acceptable for you to begin to program in C-code. From now on, you may choose to implement any code for this class in either Assembly or C-code. Additionally, you should have a pretty firm grasp on I2C protocol after Project 2. Therefore, it is also acceptable for you to start using the I2C module on the MSP430.

To further aid in a seamless transition from assembly-to-C-code and from I2C-bit-banging-to-I2C-modules, I have provided an example code which pretty much does everything developed so far this semester in a single program. It is fairly well commented and is worth reviewing. You can find this code in D2L/Content/App Notes and Sample Code

CAUTION: as with any example code, you must be sure to apply it correctly – *it will not work exactly correct in YOUR projects and direct use of the code will likely slow your own program development down – besides, attempting to copy/paste my code would be an act of plagiarism and grounds of academic misconduct. I am trying to help you, please don't inappropriately take advantage of the gift.*

Acknowledgements: Randy Larimer

SUGGESTION #1: keep your breadboard neat and organized. Use like-colored wires for different connections. Attempt to avoid large loops in your jumpers (ie: cut and strip wires).

SUGGESTION#2: Be prepared to save your work and start new project if/when you get something working. Do this often! A good project name might be: 465.04_TestLCD_v1. It will be easy to compile the wrong version of code if all your code is named *main*. Rename your *main* file with a name that is related to the project name and version. A good *main* filename might be: TestLCD_v1.

Getting Started

So far you have been told essentially how to get started on each project. Going forward, you will be given a few ideas to think about, but not necessarily *exactly* what you need to do. Eventually, you will just be told what the project requirements are with little guidance. Therefore, would be an excellent idea to *plan* out what you are going to do before starting to develop any code. That said, some guidance is still provided in this project.

Part 1: Project Planning (Notice: this step is asking you to PLAN, not DO.)
There are a number of issues you must address before getting started on this project. These include:

1. Use the in-circuit programmer to program an off-board MSP.
2. Communicate between the 2310 and the LED lightbar.
3. Communicate between the 2355 (Master) and one of the 2310s (Slaves).
4. Communicate between the keypad/2355 and the 2310
5. Set up LCD hardware, write code to display basic characters

The exact order of how you handle this does not particularly matter. One possible breakdown was presented in lecture, but you are free to approach this however makes the most sense to you. Before you go any further, make a plan with your partner in the form of a Gantt Chart similar to what was presented in class (you may use that one if you wish).

Acknowledgements: Randy Larimer

Part 2: System Planning (Notice: this step is also asking you to PLAN, not DO.)

A. *Designing* the circuit diagram now would be appropriate and will aid in laying out your hardware efficiently (ie: plan carefully, then build once). Eventually, you will be required to produce a well annotated diagram anyway; you might as well do that upfront and then use that tool to aid in your project development.

   Although now is a great time to *plan* the circuit, *constructing* the entire circuit at this point may not be the most efficient use of time

   It is your choice to produce an official (softcopy) draft of the diagram at this time, or quickly hand sketch (neatly!) your plan. You will need to submit an official copy eventually, but sometimes it is quicker to produce a hand sketched drawing at this stage.

   You won't be able to complete the LCD connections yet, but since this is conceptual you can just have a signal connection between 1 MSP2310 and the LCD. You can add the detail later.

B. Produce a high-level draft of a flow chart. Consider the requirements described below. How should this code work? What functionality should be on the memory-intensive MSP2355 and what functionality should be on the memory-limited MSP2310? Additionally, produce somewhat more detailed flowcharts for each slave code (you can use a total of 3 pages for the flow charts)

Part 3: Development Prep
Before really starting on your project, develop the necessary functionality in small blocks.

Consider the following:

- Establish the ability to use the in-circuit programmer
    - Run the previously developed Heartbeat code on a 2310
- Run the desired LED patterns on the MSP2310
    - Write code to check Variable-X and initiate the desired LED pattern
    - Modify this code to run 2 select patterns (you can develop the others later)
- Develop I2C comm between the Master/Slave
    - Develop code to communicate with a known I2C slave (perhaps the RTC)
    - Develop slave code for one 2310 which turns on an LED as soon as it recognizes its address (make sure you also ACK).
    - Develop the ability to choose which MSP2310 acknowledges the master.
- Detect key presses from Keypad (wait till Lect04b)
    - Modify existing code from Project 3 to detect a specific key press
    - Send an ASCII value to the 2310 based on the selected key
- Set up LCD hardware and write basic code (wait till Lect04b)
    - Hardcode an ASCII value to display on the LCD, then display that value

Acknowledgements: Randy Larimer

Requirements for lab project completion:
Notice: eventually, you will need three separate projects/programs, one for the Master, one for the LED slave, and one for the LCD slave; the Master code will need to control both slaves.

Part 4: Keypad-Master Code
Develop a program for the master-device that controls the LED-slave and LCD-slave. This program should do the following:

1. Upon reset, the master should require an unlock code to start.
2. The master should <u>immediately/instantly</u> detect the press of a keypad button.
3. The master should communicate the button to the LCD-slave for any button press.
4. IF the button press was a letter, the master should communicate that button to the LED-slave.

Part 5: LED-Slave Code
Develop a program for the LED-slave device to control the LED lightbar:

1. Upon slave-reset, all LEDs should be off.
1. When the master transmits a letter to the LED-slave, the slave should wake up and turn on an LED to indicate that the slave has responded.
2. The slave should then display a pattern on the LED light bar based on which letter was communicated. The patterns should function the same as in the previous project. For full credit all four letter-buttons must be functional (as opposed to the previous project where you could select a subset to get started).
3. The slave should continue to display/update any pattern until a new pattern is selected.

Part 6: LCD-Slave Code
Develop a program for the LCD-slave to control the LCD:

2. Upon slave-reset, the LCD should be blank.
3. When the master transmits any button to the LCD-slave, the slave should wake up and turn on an LED to indicate that the slave has responded.
4. The LCD should display which button was pressed. All buttons should be able to display for full credit. However, there will only be a slight deduction if you exchange the representation of the symbols (# or *) with something else.
5. The LCD should display 16 characters on the first line, then start the second line with the 17th. After 31 or 32 (your choice) characters are displayed, the LCD should clear, and the carriage should return to the upper-row-first position.

<u>Project Stretch Goal</u> (worth +5%, 0.5pts, each):
Using the * button, turn off all LEDs. Reinitialize the current LED pattern; for example, the current display is step iv of pattern C (XXX0XXXX), the display should turn off and the next time C is pressed, the pattern will restart at step i (0XXXXXXX).

Using the # button, clear the LCD so that nothing is displayed. When the next button is pressed, the LCD should display that character in the first position of row 1. Obviously, if you attempt this stretch goal, you can receive full credit if the # symbol never displays (contrary to the instructions in Part5-step2).

Acknowledgements: Randy Larimer

Assessment for project completion:
(This is a longer project, you may use upto 10min to demo, rather than standard 5min)

Your project grade will be based:

• Working code	6pts
• Professional Demo	2pts
• Documenation (flow chart and circuit diagram)	2pts

Your **Demonstration** must include (in order!):

a. Concise introduction of what the project was.
   1. Concise discussion of your <u>circuit diagram</u>
   2. Concise discussion of your <u>Master's flowchart</u>
      (You do not have to report any anything you have previously reported on for another project, IE: you reported on how you achieved the unlock code for project 3, so here you can just mention that "the unlock code was previously covered in project 3.")

b. Discussion and Demo of step 6 (yes, this is out of order), including:
   1. Discussion of the <u>flowchart</u> for the LCD-slave's code
   2. Demonstration of all buttons being pressed. If you used # or * in the stretch goals, you do not have to demo those now.
   3. Demonstration of carriage-return after row 1 and after row 2.

c. Discussion and Demo of step 5, including:
   1. Discussion of the <u>flowchart</u> for the LED-slave's code
   2. Demonstration of each pattern (make a note of where each pattern stopped)
   3. Demonstration of returning to a previous pattern
      i. Pattern B should reset
      ii. Patterns C and D should continue

d. Additional items from lecture
   1. Show where/how you found the max votlage levels for the LCD
   2. Demonstrate the output of the voltage converter (should be neg for full credit)
   3. Discuss your use of Bypass Caps.

e. If you attempted the stretch goals:
   1. Demonstrate that the * button can turn the LED display off
      i. Show that pattern C and/or D started over while the other pattern did not
   2. Demonstrate that the # button can clear the LCD
      i. Show that the next character is displayed in the upper left cell
   3. Demonstrate that you can control the contrast on the LCD
   4. If you implemented anything else, explain that and demonstrate.

Acknowledgements: Randy Larimer