ECE315 – Lab #2
Introduction to Interrupts and Interfacing for a 16-Button Keypad

Lab Dates, Report Dates and Demo Due Dates:

Please login to Moodle at eclass.srv.ualberta.ca to access all relevant lab dates. The lab schedule section contains the important dates for this semester.

Objectives:

- To gain more experience with microcomputer hardware including the 16-button keypad, the keypad encoder, the multiplexor and the oscilloscope.
- To design an interrupt-driven method for reading the keypad.
- To control the display of a sprite on the LCD using a selection of the sixteen buttons of the keypad.

Equipment, Parts, and Provided Software:

See Lab #1 for a full list of equipment, parts and software used to date.

New Portion of Lab Kit for Lab #2

- 16-Button keypad.
- MM74C922 keypad encoder integrated circuit (IC)
- SN74HC257 Multiplexor IC
- 0.1 uF ceramic capacitor
- 1.0 uF ceramic capacitor

Documentation:

Please login to the Moodle lab pages at eclass.srv.ualberta.ca to see a list of necessary documents for the ColdFire board used in ECE315. The documents are listed under the Lab Reference Materials heading.

Briefly, the manuals of most use in Lab #2 are:

MM74C922 Keypad Encoder Datasheet contains all the necessary information for the keypad encoder that handles the key presses on the 16-button keypad.

SN54H257 Multiplexor Datasheet contains all the necessary information for the multiplexor that expands the numbers of inputs available on the buffer board from four to eight.

ColdFire MCF 54415 Reference Manual chapters 17 and 18 contain information on the interrupt processing model and edge port module, respectively, of the ColdFire MCF 54415 microprocessor.

uCOS Library Manual contains the documentation on the subroutines needed to create, initialize, pend and post to queues.

NetBurner Runtimes Libraries Reference Manual chapter 11 contains the documentation on the SetPinIrq and DisableIrq subroutines needed to implement interrupts on the Coldfire Board. The manual does not include the MCF 54415 chip in the list of supported chips. It is indeed supported. Consult the file C:\nburn\MOD5441X\system\pin_irq.cpp for the legal pins for our MCF 54415-based board.

Introduction:

The MM74C922 keypad encoder chip allows the circuit designer to offload the scanning and reading of the 16-button keypad from the microprocessor to a dedicated integrated circuit (IC) that handles the timing, de-bounce, and line usage. The encoder writes the value of the last key pressed to its four data lines (DOA-DOD). It also signals that a button has been pressed by asserting the Data Available line. Note that the Data Available line is active high. The encoder also guarantees that the Data Available line will be pulled low between any two consecutive button presses. The external capacitor sizes control the timing of the keypad de-bouncing circuitry.

We'll be using the edge port module of the ColdFire MCF 54415 microprocessor to handle our encoder interrupts. Several registers of the edge port module need to be initialized in order for our encoder to function in interrupt mode. Fortunately, we have two main wrapper functions to help us with the configuration. SetPinIrq and DisableIrq can be used to configure and control interrupts for our keypad encoder/multiplexor pair.

The schematic for the lab 2 hardware circuit determines which interrupt line is used by the encoder (via the multiplexor). Once an interrupt has been accepted and handled by the ColdFire microprocessor, the programmer must signal to the microprocessor that it should resume normal processing. This will be discussed at the beginning of the lab period.

The Coldfire-based boards we are using were originally used in another course that did not require many inputs. The buffer board underneath the Coldfire-based board limits the number of inputs to four. To expand the inputs from four to eight, a multiplexor has been inserted. Four inputs are connected to the A side of the multiplexor and the data available line has been connected to B side. The code package provided manages switching between the A side and the B side for you using the PinIO class to set and clear the selector line on the multiplexor. The circuit, including the keypad encoder and the multiplexor, will be discussed briefly during the lab introduction.

Generally, all microprocessors need to be informed when the processing has been completed for any interrupt condition. Think about an interrupt generated by a keypad. How do we tell the microprocessor which code to run as a result of an interrupt? What condition has arisen that needs to be processed? Lastly, how do we tell the microprocessor that our interrupt service routine has completed its processing and that it should resume normal processing? Using the SetPinIrq wrapper function takes care of instructing the processor what code to run and when to resume normal processing. If you are interested in the actual code, consult the pin_irq.cpp file mentioned above.

Prelab:

1. You are to arrive at the lab with the circuit pre-wired according to the schematic. The lab instructor will check your circuit at the beginning of the lab.
2. Read the lab handout and encoder datasheet to learn about the keypad the keypad encoder, and the multiplexor.
3. Skim Chapters 17 and 18 of the ColdFire Reference Manual for general information on the interrupt processing model and the edge port module of the ColdFire.
4. Consult the uCOS Library manual for information on creating, initializing, and using queues.
5. Consult the NetBurner runtimes libraries for the SetPinIrq and DisableIrq subroutines that you'll need to complete the lab.

Exercise 1:

Goal: To use the oscilloscope to compare a few timing parameters against the encoder datasheet values.

Steps:

1. DO NOT connect the 50-pin ribbon cable between the ColdFire board and your breadboard just yet. Connect 3.3V from the power supply to power the encoder. You will start by taking measurements while unconnected to the ColdFire board.
2. Consult the oscilloscope user manual for instructions on how to do single shot capture. You will also need to make use of the measuring cursors to complete the following two measurements.
3. You will be measuring some timing characteristics of the encoder using the oscilloscope. The Y lines coming into the encoder via the keypad are normally pulled high. Once any button is pressed, the Y line is driven low, the Data Available line is driven high and the output lines are driven with the number of the button that was pressed (0-15). Measure the time between any of the Y lines going low and the Data Available line going high.
4. Measure the time when the Data Available line goes low between two consecutive button presses. Generate this scenario by pressing a button and then pressing a second button without releasing the first button. Now release the first button and measure how long the Data Available line is low. Take several readings and record the minimum and maximum for all of the presses.
5. Compare the expected values from the MM74C922 datasheet. Do your measured results match with the expected results? Discuss this in your report.

Measurement	Time	Matches Datasheet ?
Time from any Y line going low to Data Available going high.		
Press any button. Next, press another button without releasing the first button. Now release the first button and measure how long the Data Available line is low. Take several measurements and record the min and max.		

Exercise 2:

Goal: To build a new project for lab 2.

Steps:

1. Connect the 50-pin cable from the ColdFire microprocessor to your breadboard.
2. Get the lab2.zip file from the eClass course website.
3. Create a new project called lab2 in your ECE 315 workspace. Follow the same procedure as you did in the tutorial.
4. Unzip and import all of the files into your project and delete the main.cpp file before building your project.
5. Run the project and check the heartbeat and context switching pins (J2[3]andJ2 [4]). Verify that the default webpage is being served.
6. We will be using the LCD for output in this lab. Please remember that the encoder, multiplexor, and the LCD use 3.3V. Using the wrong voltage may damage the board.
7. Please note that at this point none of your keypad hardware will work as expected. You need to complete exercise three to see the keypad working.

Exercise 3:

Goal: To use interrupts to catch the key presses as they occur and echo them to the Mttty console.

Steps:

1. Modify the existing polling code in Keypad::Init and Keypad::read_data to include the necessary interrupt code. You'll be using the SetPinIrq and DisableIrq functions for this.
2. Complete the Keypad::KeyPadISR1 to include the code that handles the keypad presses and transmits that info into the appropriate task.
3. You should see each button press represent a different character or string on the keypad. The provided mapping is given in keypad.cpp. Change the map to get unique outputs for each of the sixteen keys. Use the serial console to display the value that each button maps to.
4. Once your interrupts are working correctly, you must pass the value read from the keypad encoder to the UserMain runtime loop or to a new task created for the purpose. Using a queue to pass the keypad values is the recommended way. Consult the uC documentation on how to use queues.

How you choose to pass this information via the queue is a design decision on your part.

Exercise 4:

Goal: To use 4 of the 16 buttons on the keypad to move a sprite on the LCD.

Steps:

1. The four corner keys will be mapped to four separate functions: up, down, left, and right. The movement of the sprite should be bounded by the outer edge of the LCD meaning at no time should the display of the characters rollover. Pressing the up, down, left and right keys when the box is at the edge should result in no change to the display.
2. During the demo, the lab instructor will ask for your key layout for testing.
3. The other 12 buttons on the keypad should be ignored.

Report Requirements:

- Your typed report is to briefly describe your design solution for the two lab exercises. You must also include your table entries for exercise one.
- Be sure to provide neatly formatted and well-commented code for all exercises.
- Briefly describe your test cases for verifying that each of your solutions worked properly. Describe which case was tested, as well as the anticipated and actual results for each case. Place this data in a table. **Do not write out your test cases. Please tabulate them instead.** Non-tabulated test cases will not receive full credit on the reports.

Consult the report writing guidelines if you have any questions regarding the report format. The report writing guidelines are available on the Moodle eClass page under Lab Reference Materials.

Marking Scheme:

Lab #2 is worth 20% of the final lab mark. Please view the lab 2 marking sheet to ensure that you have completed all of the requirements of the lab. The marking sheet also contains a limited test suite in the demo section. Please make use of it.

Report Submission:

Reports should be submitted via eClass by section. Please note that late reports and demos will not be accepted.