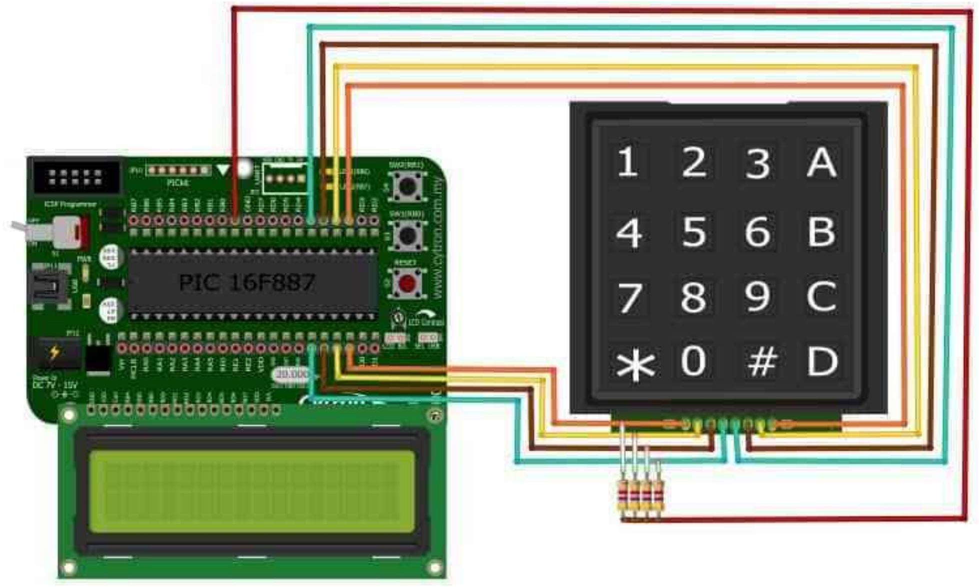
Microcontrollers Lab

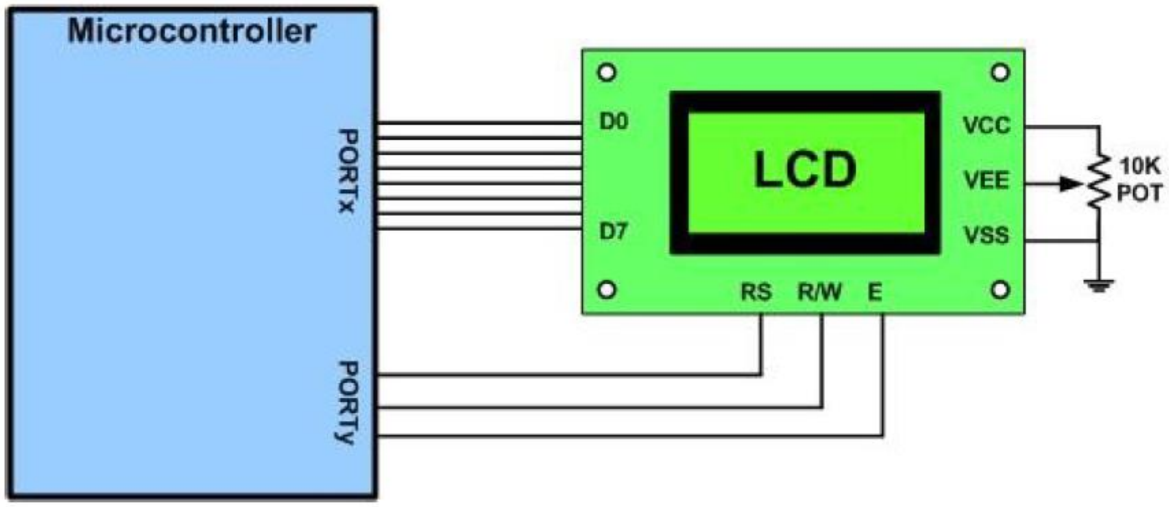
Week 6 – Introduction to ARM MCU programming

March 30th – Due April 13th 2020

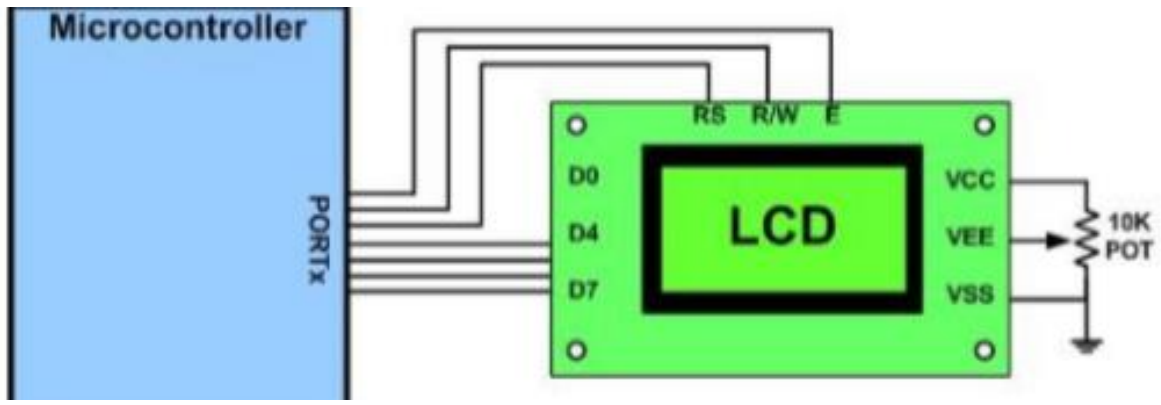
In this lab, we will see how drive multiple GPIOs in the ports of the KL25Z board. As such, we will work with the LCD screen and matrix keyboard, although the connections won't be made at this time (this part will be left for later in the course due to the special circumstances)



Part 1. Write the code for writing the Hello World of embedded systems when working with the LCD screen. The first part will be done with the 8-bit configuration seen in class. Make use of the internal ready signal instead of delays.



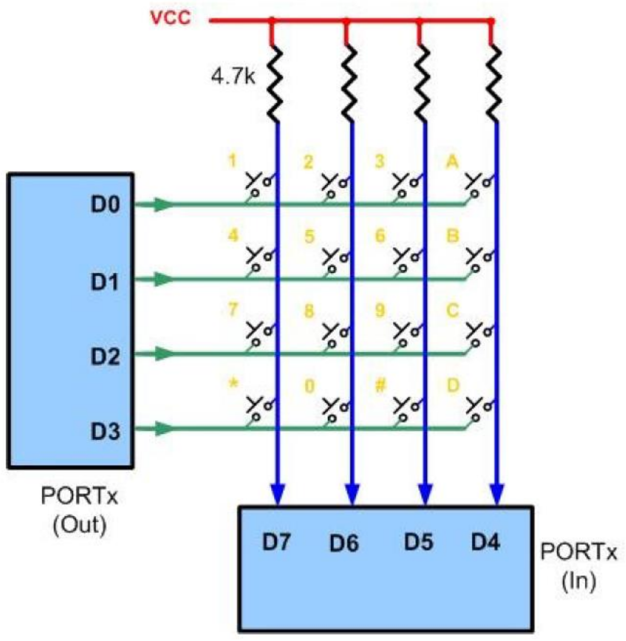
Part 2. Develop the 4-bit version of the program to save pins in your design.



Part 3. To reduce the microcontroller's I/O pin usage, keyboards are organized in a matrix of rows and columns. The CPU accesses both rows and columns through ports; therefore, with two 8-bit ports, an 8 × 8 matrix of 64 keys can be connected to a microprocessor. When a key is pressed, a row and a column make a contact; otherwise, there is no connection between rows and columns

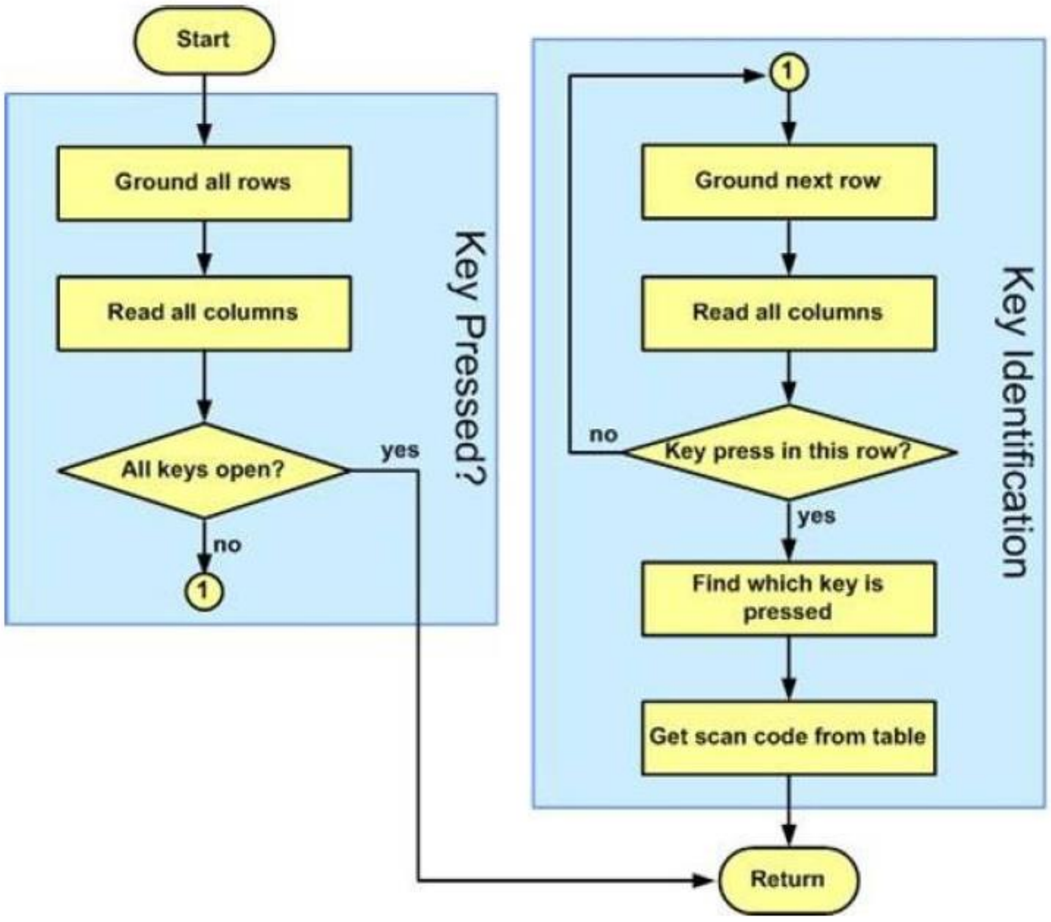
The figure shows a 4 × 4 matrix connected to two ports. The rows are connected to an output port and the columns are connected to an input port. All the input pins have pull-up resistor connected. If no key has been pressed, reading the input port will yield 1s for all columns.

If all the rows are driven low and a key is pressed, the column will read back 0. It is the function of the microprocessor to scan the keyboard continuously to detect and identify the key pressed.



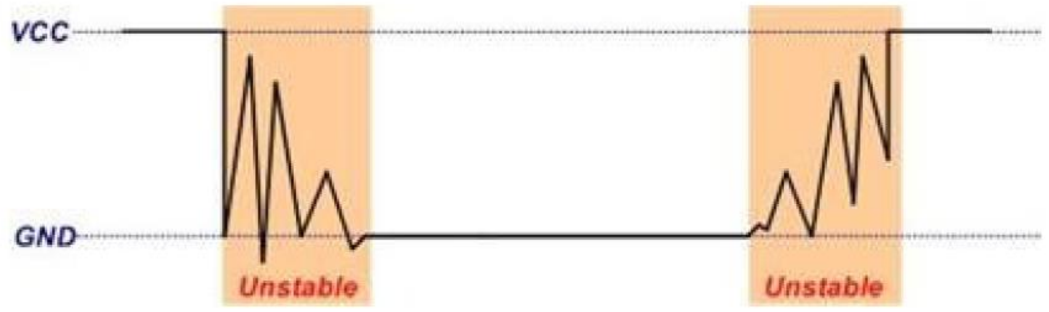
To detect the key pressed, the microprocessor drives all rows low then it reads the columns. If the data read from the columns is D7–D4 = 1111, no key has been pressed and the process continues until a key press is detected However, if one of the column bits has a zero, this means that a key was pressed. For example, if D7–D4= 1101, this means that a key in the D5 column has been pressed

After a key press is detected, the microprocessor will go through the process of identifying the key. Starting from the top row, the microprocessor drives one row low at a time; then it reads the columns. If the data read is all 1s, no key in that row is pressed and the process is moved to the next row. It drives the next row low, reads the columns, and checks for any zero. This process continues until a row is identified with a zero in one of the columns. The next task is to find out which column the pressed key belongs to. This should be easy since each column is connected to a separate input pin



In this part, you have to implement a mechanism by which the microprocessor scans and identifies the key

Part 4. Contact debouncer. When a mechanical switch is closed or opened, the contacts do not make a clean transition instantaneously, rather the contacts open and close several times before they settle. This event is called contact bounce



So, it is possible when the program first detects a switch in the keypad is pressed but when interrogating which key is pressed, it would find no key pressed. This is the reason we have a return 0 after checking all the rows. Another problem manifested by contact bounce is that one key press may be recognized as multiple key presses by the program. Contact bounce also occurs when the switch is released. Because the switch contacts open and close several times before they settle, the program may detect a key press when the key is released.

For many applications, it is important that each key press is only recognized as one action. When you press a numeral key of a calculator, you expect to get only one digit. A contact bounce results in multiple digits entered with a single key press. A simple software solution is that when a transition of the contact state change is detected such as a key pressed or a key released, the software does a delay for about 10 – 20 ms to wait out the contact bounce. After the delay, the contacts should be settled and stable.

In this part, you have to develop a contact debouncer for the code in the previous part.

Requirements for the report

1. Include some research about how the LCD screen and keyboard operate in the corresponding section. Provide enough detail to demonstrate that you understand the basics and how the concepts relate

2. Include the code for each of the sections in the lab. The code should be commented and the report should include a short description of each function and how it works, including images of the registers that have been configured for enabling the different functionalities in your code.

3. Make some research in how to connect the board to the external components. You can use a virtual program to schematize the connection between the board and the external components (LCD screen and keyboard, external components). An example is Fritzing, but there are others

4. Include research about noise in digital systems and the need for the debouncer. Include any research you have done and how you implemented the solution for the last part of the lab.