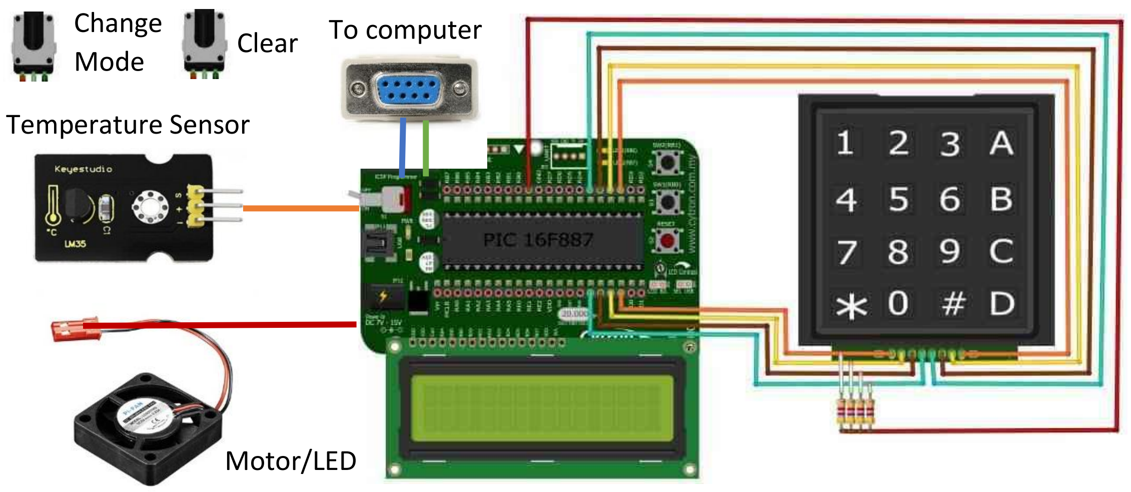
Microcontrollers Lab

Week 13 – LCD, Keyboard and PWN integration

May 25th – <u>Due June 3rd 2020</u>

In this lab the idea is to integrate aspects of the various modules we have seen so far: LCD display, 4x4 matrix keyboard, interrupts, ADC and PWM. Before proceeding, be sure that your code for the LCD and the keyboard works properly. You can use either the 8- or 4-bit option for the LCD code.



The lab will be divided into four parts, described as follows:

Part 1. PWM Part 1. Modify the first example seen in class to generate PWM signals with a frequency of 60Hz (TPMx_MOD = 43702) but different duty cycles: 0, 25%, 50%,75% and 100% (since we are using non -inverted PWM we will have the LED completely on for 0 DT and off for 100%.

You have to calculate the values for the TPMx_CnV register for the different duty cycle values

Part 2. PWM Part 2. Implement the code for the second example seen in class, in which the PWM signal is changed by increments of 1% in the CnV register. Observe how the intensity of the LED decreases as we increment the duty cycle (because the LED is active low)

Part 3. Simple industrial control with PWM. Imagine that we want to create an industrial grinder. Usually, just like a with house blender, these systems have several power configurations depending on how "hard" are the components (i.e. gravel, stones) we wish to grind and thus the motor changes speed (and thus torque) to do the work. In this sense, we will create a simple application that can change the speed depending on a setting defined by the user

1. When you start your application, the message next message should be displayed.

Set the mode input mode

Mode 1: M   Mode 2: A

Mode M stands for manual and mode 2 stands for Automatic. We will see what each mode implies. You can use each mode by pressing whichever button in the keyboard you choose and pressing # toe execute the rest of your application

2. In manual mode, the idea is that you integrate the first part of this lab, but modifying the code for generating inverted PWM signals. Thus, the LCD should display

Select Speed

1: L 2:M 3:MH 4:H

(standing for Low, Medium, Medium High and High)

After pressing the one of the keys the corresponding CnV register values should be sent to a function the initializes the PWM (duty cycles of 25%, 50%, 75% and 100%). Please send the value as an argument to the function (if you use if statements the code becomes very long and it is very inefficient). The LED is this case will go from very low intensity to completely on (as the case of a motor). If you have a fan or a small motor you can try as we do not require and H bridge

2. After this, if we want to modify the speed of the motor, we can use a push button to send an interrupt to the MCU and display the above menu again. A second button can be used to stop the system the motor at any given time (emergency signal)

Just as a note, the ADC could be used here to monitor the motor current or heat and send a signal to the microcontroller if there is an overload (the system cannot grind something for instance) and increase the power or switch the motor off. Something in the context is explored in the next part of the lab (optional)

Automatic Mode (extra points)

1. This system is very similar to the previous case: there are 4 predefined power levels, which are chosen depending on the value of the CnV register. However, in this case, we monitor continuously the value of the ADC connected to a potentiometer (simulating the load of the motor). If the value is between 0 – 0.75V the motor should run on Mode 1; between 0.76 and 1.5V on Mode 2; between 1.51 and 2.25V on mode 3 and finally, between 2.25 and 3V on Mode 4. We also should use a button to go the main menu and a second button to stop the motor.

Requirements for the report

1. Include the code for each of the functions of your code. 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 (you should include the image of the registers as seen in class and in your commented code put which bits have been configured for certain purposes)

2. Provide a state machine or a flow diagram for the entire code (only applies for the third part)

3. For the last part, provide a schematic view of the connections of your design.

4. Attach a short video demonstrating the system working. Alternatively, you can share the link to the video in your google drive for me to evaluate it.