Take Home Exam III
Revision 1.1
Deadline: 30.04.2018, 23:55

Any clarifications and revisions to the assignment will be posted to the ODTUCLASS discussion forum.

1. Introduction

The purpose of this assignment is to familiarize you with the **ADC module** of the PIC, as well as using the **LCD module** together with **TIMERs** and **INTERRUPTs**. You will implement an application for a product called "very safe" which is a steel safe protected through an electronic lock.

The safe will be pin-protected. Your first job is to set a pin. After setting the pin, there will be a test period. In this period, you will try to enter the correct pin. If the correct pin is entered, a message should be shown, indicating a successful login. If the pin is incorrect, the system should deny the login attempt and wait for other attempts. After two consecutive unsuccessful attempts, the system should suspend the user from subsequent attempts for 20 seconds.

When the PIC is powered up, the initial screen on the LCD module should be as shown in **Figure 1**.

	$	**>**	**V**	**e**	**r**	**y**			**S**	**a**	**f**	**e**	**<**	**$**	
	$	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	

Figure 1. The first screen when PIC is powered up

If the user pushes and releases the RE1 push button, then the LCD screen should remain like the image shown in **Figure 1** for **approximately 3 seconds (+/- 5 ms can be acceptable as error range)**. For only this particular 3 second wait, **you can use busy-wait "loops" instead of a timer (you are, however, not allowed to use built-in functions for this task)**. During this time interval, 7-segments displays should remain unchanged.

Following the press and release of the **RE1** button and the subsequent 3-second wait, **the pin-setting period** starts. Four '**#**' symbols should appear in the 12^{th}, 13^{th}, 14^{th}, and 15^{th} cells of the first row of the character LCD module corresponding to the four digits of the pin to be entered after the "Set a pin:" message. **Figure 2** shows how the LCD should look like at this stage. Meanwhile, the '**#**' symbol in the 12^{th} cell should be **blinking,** meaning that the corresponding digit is **active** to be set by the user. The blinking for the '#' symbol on the currently active cell should be done with a **250 ms time interval**. In other words, in the first 250 ms, the **'#'** character should be shown and in the next 250 ms the space character (' ') should be shown. You should use **TIMER0** and the associated interrupt to produce the required 250 ms time delay.

To select a number for the currently active digit, you will be using the **ADC potentiometer**. When the user uses the potentiometer, a number must appear instead of the '#' symbol based on the ADC value on the active cell. **Do not blink the number** on the active cell (only the active '#' character blinks).

Following the acquisition, the ADC value will be in between 0 and 1023. You should map these **ADC values** to **the numbers between 0 to 9** based on the ranges given in **Table 1**.

	S	**e**	**t**		**a**		**p**	**i**	**n**	**:**	**#**	**#**	**#**	**#**	

Figure 2. The initial screen for the LCD module after RE1 is pressed and reeased and then 3 seconds pass. Time to set a pin.

Table 1. Mapping of ADC Values to numbers.

ADC Value	Number
0-99	0
100-199	1
200-299	2
300-399	3
400-499	4
500-599	5
600-699	6
700-799	7
800-899	8
900-1024	9

To change the currently active digit to be set, the **RB6** push button (**with PORTB change interrupt**) should be used. Suppose that the number '2' was selected as the first digit of the pin. Suppose, also, that the user pushes the RB6 button to select the second digit. At this point, the LCD should look like Figure 3, with the '#' sign in the 13th position blinking. The same process is repeated for the 3rd and 4th digits. After all pins are set as shown in **Figure 4**, the **RB7** push button is used for **setting the pin**. When the pin is set, the character LCD module shows the message in **Figure 5,** including the pin. The newly set pin should be shown to the user for **exactly 3 seconds**, while the entire message is blinked with a 500 ms time interval. You should use **TIMER0** and interrupts to produce the 500 ms time interval for a total of 3 seconds. In other words, in the first 500 ms you should show the message, and in the next 500 ms the LCD module should be clear. During the pin-setting period, the **7-segment displays** must show '-' characters as shown in **Figure 6**.

	S	**e**	**t**		**a**		**p**	**i**	**n**	**:**	**2**	**#**	**#**	**#**	

Figure 3. Character LCD when the first digit of the pin is set.

	S	**e**	**t**		**a**		**p**	**i**	**n**	**:**	**2**	**4**	**8**	**3**	

Figure 4. The LCD module when all digits of the pin are set.

	T	**h**	**e**		**n**	**e**	**w**		**p**	**i**	**n**		**i**	**s**	
			-	**-**	**-**	**2**	**4**	**8**	**3**	**-**	**-**	**-**			

Figure 5. The LCD module when the pin is set using the RB7 button.

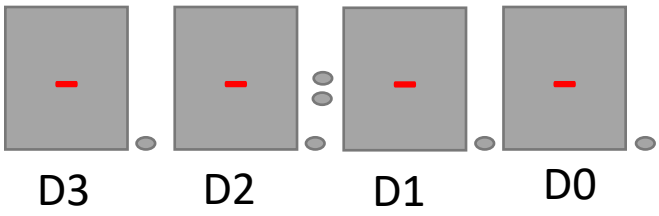

Figure 6. 7-Segment displays during the pin-setting period.

After showing the new pin to the user for **exactly** 3 seconds, **a test period of 120 seconds** begins. The remaining time is shown on the 7-segment displays starting from 120. This counter should be decremented once every second, exactly (You should use **TIMER1** and interrupts to produce this one second time delay). Figure 7 shows the initial case for the 7-segment displays in the test period. D3 should show '0' during the test period. Also, D2 should show '0' when the counter is between 99 and 0.

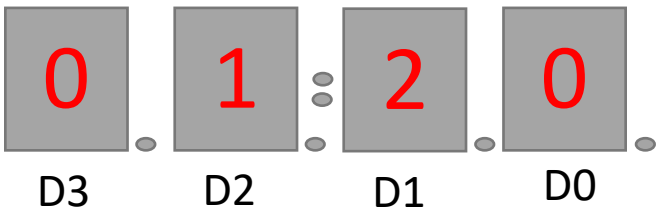

Figure 7. 7-Segment displays at the beginning of the test period.

In the test period, the pin should be entered by the user with a process similar to the pin-setting period. So, each digit of the pin is entered with the help of the ADC potentiometer. The **RB6** button is used for changing the currently active digit and the **RB7** button (**with PORTB change interrupt**) is used to finish pin selection when all four digits are ready. Figure 8 shows a snapshot of when the first and the second digits of the pin are ready. Four digits of the pin should be shown after the "**Enter pin:**" message at the first row. At the second row, the **number of allowed attempts** is shown after the "**Attempts:**" message.

	E	**n**	**t**	**e**	**r**		**p**	**i**	**n**	**:**	**2**	**5**	**#**	**#**	
		A	**t**	**t**	**e**	**m**	**p**	**t**	**s**	**:**	**2**				

Figure 8. The screen for the first attempt of the user in the test period.

If the user enters an incorrect pin**, the counter for the number of allowed attempts** is decreased and the corresponding cell on the character LCD module is updated (2nd row, 12th column) as in the case Figure 9. In Figure 9, the first attempt of the user is unsuccessful. If the user enters an incorrect another time (consecutively, for a total of two times), the system should suspend subsequent attempts for **20 seconds**. **During these 20 seconds:**

- **The 7-segment displays should keep showing the current time**. For example, if the 2nd incorrect pin is entered when the remaining time is 83 seconds, the system suspends new attempts until 63 seconds left.
- The character LCD module shows the message as described in Figure 10.
    - The '**#**' characters should be replaced with the 'X' characters.
- The RB6/RB7 push buttons and the ADC potentiometer should have no effect.

	E	**n**	**t**	**e**	**r**		**p**	**i**	**n**	**:**	**1**	**#**	**#**	**#**	
		A	**t**	**t**	**e**	**m**	**p**	**t**	**s**	**:**	**1**				

Figure 9

	E	**n**	**t**	**e**	**r**		**p**	**i**	**n**	**:**	**X**	**X**	**X**	**X**	
T	**r**	**y**		**a**	**f**	**t**	**e**	**r**		**20**		**s**	**e**	**c**	**.**

Figure 10

When the pin is entered correctly, the LCD module should show the message in Figure 11. Also, the 7-segment display should stop decreasing and keep showing the remaining time of the test period.

S	**a**	**f**	**e**		**i**	**s**		**o**	**p**	**e**	**n**	**i**	**n**	**g**	**!**
$	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**	**$**

Figure 11

If the user cannot enter the correct pin within **120** seconds (in the test period), the program should be started again.

2. Specifications

- **40 MHz** frequency and **10-bit** adjusted ADC will be used. The ADC value will be used to cycle through numbers.

- The ADC value should be 0 when you turn the ADC potentiometer clock-wise to its leftmost position, and it should be 1023 when you turn the ADC potentiometer clock-wise to its **rightmost** position.

- In this exam, you will use **TIMER0** interrupt with a period of **100 ms**. The **Timer0 ISR should start ADC conversion**. You should also use an **ADC ISR** to detect the end of conversion and read the converted value. Consequently, you should be sampling the potentiometer value with a frequency of **10 Hz**. Each ADC read should update the active LCD module line (explained below). **Timer0** should be configured to be **8-bit**.

- **In your ADC implementation, it is not acceptable to use any implementation without using (Timer and ADC Interrupt).**

Coding Rules

- You will code your program using PIC C language. You should use **XC8 C compiler** for MPLAB X. You can find the related documents on the Recitation04 and ODTUCLASS.
- Your program should be written for PIC18F8722 working at **40 MHz.**
- **When the system is started, the LCD module must be all clear, with the configuration that the cursor and blink are off.**
- **You should control RB6 and RB7 buttons with PORTB change interrupt. The pull-up enabling or disabling (using INTCON2bits.NOT_RBPU) cases are up to your design. You should obey the rules corresponded with the push button connected to the RE1 pin.**
- **You should obey the specifications above about TIMER0 and TIMER1 interrupt implementations, and also the busy wait loop specification.**
- **The corresponded switch configuration is also included in THE3 files on ODTUCLASS. If you have a different configuration, it is possible, upload also this configuration figure with showing the differences.**
- You can modify the LCD module codes if you need for latency or etc...

Resources

→PIC 18F8722 Datasheet
→The LCD Module Datasheet
→ PIC Development Tool Programming & User Manual
→ Course web page and newsgroup
→ Each related recitation demo codes and notes can be used.
→ LCD example and a template code in Recitation4 documents on ODTUCLASS.

HOW TO ADD XC8 COMPILER TO MPLAB X IDE ON INEK MACHINES

1. Type "mplab_ide" to open the Mplab X IDE on terminal.

2. `Look at Tool > Options > Embedded` menu. The xc8 compiler is already installed.

3. Select `File -> New -> Standalone Project`. Create new project with the following configurations.
   - Select device -> PIC18F8722
   - Select tool -> PICKIT2:SN BETI
   - Select Compiler -> XC8

Hand in Instructions

→ You should submit your code as a single zip file named as the3_##.zip through ODTUCLASS (## represents your group number). This file will include the3.c, lcd.c, Includes.h and lcd.h. At the top of the3.c, you should write your ID, name, and surname as commented out in "the3.c" file. Do not forget to replace ## with your group number. **Only one group member must submit the the3_##.zip by considering the naming issue as before. Otherwise, you will lose partial points.**
→ **You should add comments on specific codes or code blocks.**

Grading

Total of the take-home exam worth 100 points. You should include brief but descriptive comments in your code, including a larger comment at the beginning of your **main.c** file describing the structure of your program with function descriptions (For example the place, function name of your Timer0/1 ISR, ADC ISR) and their relations to one another. Your grade will also depend on the quality of your design, not just its functional correctness. If there is an unclear part of your code, we may ask any of the group members to describe that code segment. Also, group members may get different grades. We reserve the right to evaluate some or all of the groups to determine the contribution of each group member to the assignment.

→ A correct implementation of the timer ISRs and proper timing… Related code and the resulting values will be checked,
→ Proper ADC operation, ADC interrupt implementation and related code will be checked,
→ Correct and problem-free use of LCD functionality,
→ Proper button use and correct implementation of PORTB change interrupt,
→ Proper usage of the 7 Segment Displays,
→ Proper operation of your program.

will be considered while grading.

Note: Some examples of improper usage of the modules are below:
- Too much flicker on the LCD module and/or 7-segment displays is unacceptable.
- If the digits or characters on the LCD module and/or 7-segment displays disappear while using a push button, that is unacceptable.

Cheating

We have zero tolerance policy for cheating. People involved in cheating will be punished according to the university regulations.

Cheating Policy: Students/Groups may discuss the concepts among themselves or with the instructor or the assistants. However, when it comes to doing the actual work, it must be done by the student/group alone. As soon as you start to write your solution or type it, you should work alone. In other words, if you are copying text directly from someone else - whether copying files or typing from someone else's notes or typing while they dictate - then you are cheating (committing plagiarism, to be more exact). This is true regardless of whether the source is a classmate, a former student, a website, a program listing found in the trash, or whatever. Furthermore, plagiarism even on a small part of the program is cheating. Also, starting out with code that you did not write, and modifying it to look like your own is cheating. Aiding someone else's cheating also constitutes cheating. Leaving your program in plain sight or leaving a computer without logging out, thereby leaving your programs open to copying, may constitute cheating depending upon the circumstances. Consequently, you should always take care to prevent others from copying your programs, as it certainly leaves you open to accusations of cheating. We have automated tools to determine cheating. Both parties involved in cheating will be subject to disciplinary action. [Adapted from http://www.seas.upenn.edu/~cis330/main.html]