Middle East Technical University | Department of Computer Engineering

CENG 336

Int. to Embedded Systems

March 2018

Take Home Exam 2

Due date: 08 April 2018, 23:55
Submission: **via ODTUCLASS**

1 Objectives

This assignment concerns basic input/output operations together with programming with **interrupts and timers**. This will done in the context of implementing a simple version of well known pong game (see Figure 1). Clarifications and revisions on the content, scope and requirements of the assignment will be posted to the course page discussion forum in ODTUCLASS.

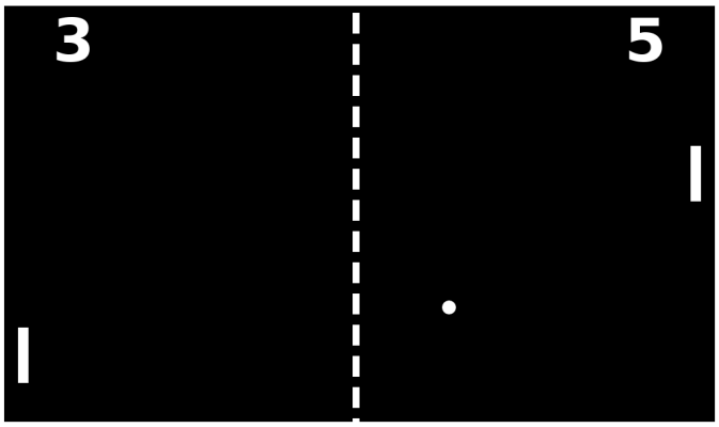

Figure 1: A screenshot from a pong game.

2 Scenario

The Pong Game is for two players. In the context of the game scenario you have have two paddles and a ball. The paddles will be located opposite ends of the game field. Each player will move one of the paddles vertically. Players will hit the ball back and forth using these paddles. You will display score for each player on 7 segment display screen. **Once one of the players gets five points, the game will be over.** You will move the paddles using buttons and in order to move the ball you will use the **Timer0 interrupt**.

3 Specifications

3.1 Initial Configuration

The game will be played in the area of the LEDs trough RA0-RA5, RB0-RB5, RC0-RC5, RD0-RD5, RE0-RE5, RF0-RF5. Figure 2 illustrates the border of the game field where the ball will move across these LEDs.
You could see the initial configuration of the game in Figure 3. Board should start with this configuration. As it could be seen from the figure, the paddles are represented with three LEDs and the ball is represented with one LED. Also the scores for the players are set zero initially.

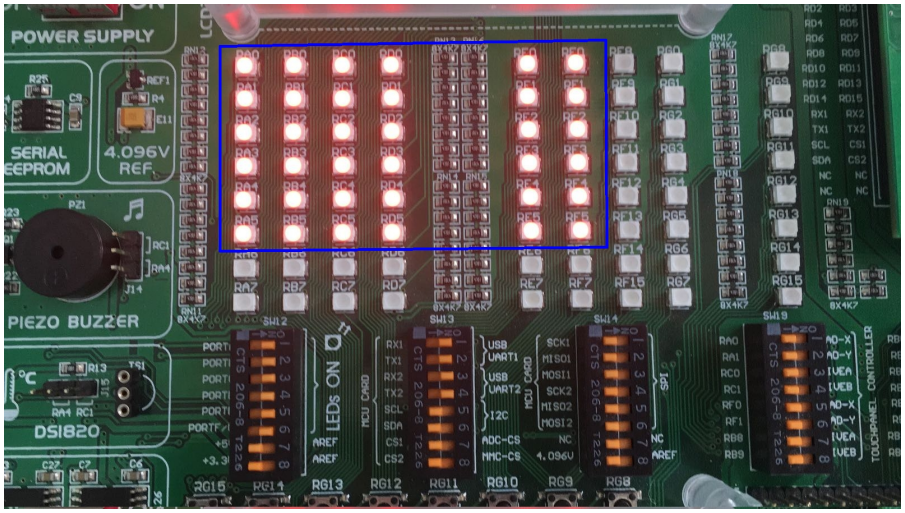

Figure 2: Predefined game field.

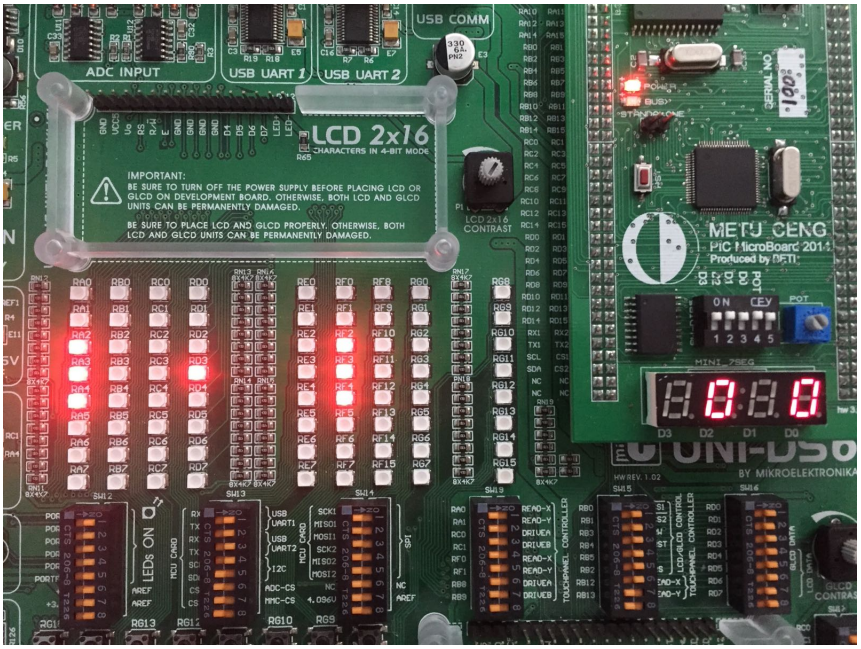

Figure 3: Start configuration of the game.

3.2 Moving the Paddles

Player1 will use RG3 and RG2 buttons and Player2 will use RG1 and RG2 buttons to move the paddles. Each time when the one of the given buttons is pressed, the paddles should move according to the button specification. **The left paddle will move between RA0 and RA5**, and **the right paddle will move between RF0 and RF5**. You should not cross the borders of the game field.

- **Player1 will use RG3 button to move the left paddle up.** Lets assume that the left paddle is on RA2-RA3-RA4 and RG3 button is pressed, after that paddle should be on RA1-RA2-RA3.
- **Player1 will use RG2 button to move the left paddle down.** Lets assume that the left paddle is on RA2-RA3-RA4 and RG2 button is pressed, after that paddle should be on RA3-RA4-RA5.
- **Player2 will use RG1 button to move the right paddle up.** Lets assume that the right paddle is on RF2-RF3-RF4 and RG1 button is pressed, after that paddle should be on RF1-RF2-RF3.
- **Player2 will use RG0 button to move the right paddle down.** Lets assume that the right paddle is on RF2-RF3-RF4 and RG0 button is pressed, after that paddle should be on RF3-RF4-RF5.

3.3 Moving the Ball

In the original version of the game, to calculate the path of the ball there are many variables to consider including speed of the paddles and ball, and the collision angel between ball and paddle/borders.
In this assignment you will use predefined rules to move the ball. In order to decide ball movement direction you will use the following **direction selection algorithm**.

1. Assume that the ball is on **RC2 and moving left.**
2. **If the rightmost 2 bits of Timer0 are b'00' or b'11'** then no change in the direction of the ball.
   (a) Change the PORT and move the ball (i.e same bit pattern).
   (b) The ball is on **RB2** now.
3. **If the rightmost 2 bits of Timer0 are b'01'**;
   (a) Change the PORT and move the ball up (i.e. from b'00000100' to b'00000010').
   (b) The ball is on **RB1** now.
4. **If the rightmost 2 bits of Timer0 are b'10'**;
   (a) Change the PORT and move the ball down (i.e. from b'00000100' to b'00001000').
   (b) The ball is on **RB3** now.

Here is an example scenario based on the Figure 4:

1. The ball is on **RC2 and moving left**.
2. The direction selection algorithm decided to move down. Now the ball is on **RB3**.
3. The direction selection algorithm decided to move up. Now the ball is on **RA2**. Since the ball is on PORTA, you should check whether the paddle misses the ball.
4. The direction selection algorithm decided to move up. Now the ball is on **RB1**.
5. The direction selection algorithm decided to move up. Now the ball is on **RC0 ***.
6. The direction selection algorithm decided to move down. Now the ball is on **RD1**.
7. The direction selection algorithm decided to move down. Now the ball is on **RE2**.
8. The direction selection algorithm decided to move down. Now the ball is on **RF3**. Since the ball is on PORTF, you should check whether the paddle misses the ball.

9. The direction selection algorithm decided to move down. Now the ball is on **RE4**.
10. The direction selection algorithm decided not to change direction. Now the ball is on **RD4**.
11. The direction selection algorithm decided not to change direction. Now the ball is on **RC4**.
12. The direction selection algorithm decided to move down. Now the ball is on **RB5 ***.
13. The direction selection algorithm decided to move up. Now the ball is on **RA4**. Since the ball is on PORTA, you should check whether the paddle misses the ball.
14. The paddle missed the ball and Player2 scored.

***** You should handle the cases when the ball is on the borders (i.e. RA0, RB0, RC0, RD0, RE0, RF0, RA5, RB5, RC5, RD5, RE5, RF5), since in these situations there are only two options for the next ball location.

The ball should move from its current location to the next location in every **300 ms**. **You should use the Timer0 interrupt to measure this duration.**

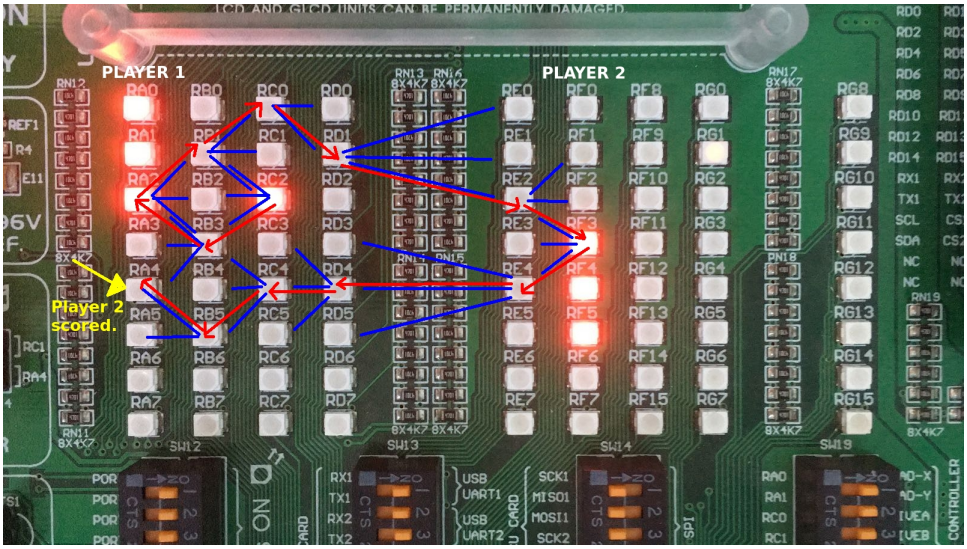

Figure 4: An example scenario from the game. The ball is on the **RC2 and moving to the left**. The blue lines show the valid locations for the next ball location. Red arrows show the ball locations after each selection.

3.4 Updating the Scores

The players will have a score when the opponent's paddle misses the ball (Figure 4). You should update 7 segment display after each score. You should also update the paddles and the ball as in start configuration (see Figure 3).

4 Implementation and Regulations

- You will code your program using PIC assembly language.
- Your program should be written for **PIC18F8722** working at **40 MHz**.
- You will use nine ports in this assignment; PORTA, PORTB, PORTC, PORTD, PORTE, PORTF, PORTH, PORTJ and PORTG.
- PORTA and PORTF will be used to show paddles.

- PORTA, PORTB, PORTC, PORTD, PORTE, PORTF will be used to show the ball.
- PORTH and PORTJ will be used as output ports to show numbers on 7-Segment displays. You can find some useful information on **Hints** section.
- PORTG will be used to move paddles.
- You are not expected to use interrupts for the buttons (PORTG). You could have button tasks based on states.
- You should use the Timer0 interrupt as stated above and configure the Timer0 to work in 8-bit mode.
- It is beneficial to avoid function calls in Timer ISRs. You may use some counters or flags and do the function calls or calculations in your main routine. For example, if you call a display subroutine in ISR, measuring the correct time interval in ISR will become harder.
- You will get most of your points from the proper usage of **Timer0** together with the performance of buttons.
- When you are writing your code, you can use the lecture notes on Input/Output and Interrupts, Recitation documents. It is also highly recommended that you make extensive use of the PIC data sheet, MCDEV Kit User Manual and Programming Manual. Please consult these resources first before you ask any questions to the assistants or on the forums.

5 Hints

5.1 7-Segment Displays

There are four common cathode 7-Segment displays mounted on the development board. PORTJ and PORTH are used as data bus and selection pins, respectively, as illustrated in figure below. Notice that PORTH pins are connected to all displays.

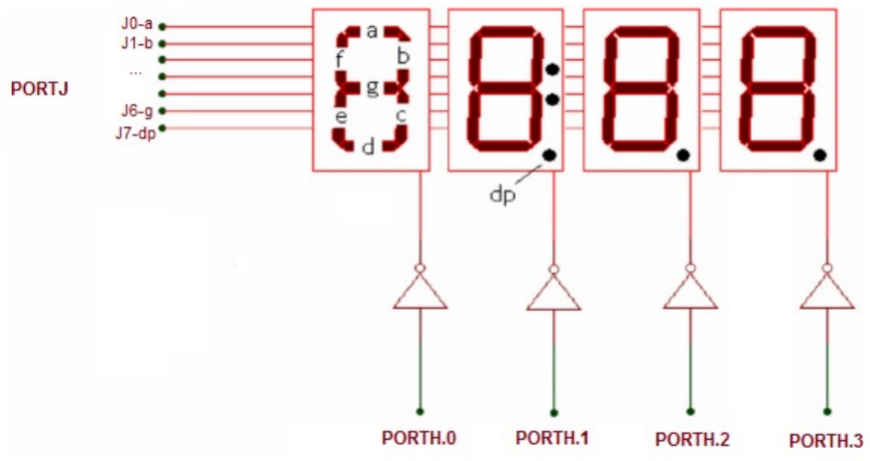

Figure 5: Connections for the 7-Segment displays on the development board.

As an example, if you want to show number "4" on leftmost display (DISP0), you should first select it by setting RH0 and clearing RH1, RH2 and RH3, then send binary "01100110" to PORTJ. Hence, a, d, e segments on DISP0 will be turned off, and b, c, f, g segments will be turned on. Note that RJ7 pin is used for dp of displays. Also note that there is no dp on DISP0 (leftmost 7-segment display) and there are 3 dp on DISP1 which run simultaneously.

If you want to show, for instance some value(1234) on the displays, you should select only DISP0 by using PORTH, write the byte necessary to turn on segments to PORTJ, and wait for a while. Then you should do the same for DISP1, DISP2 and DISP3. This is illustrated in figure below. If you adjust "on" times properly and repeat on-off cycles continuously, you show your values on the displays in a smooth manner without flicker.

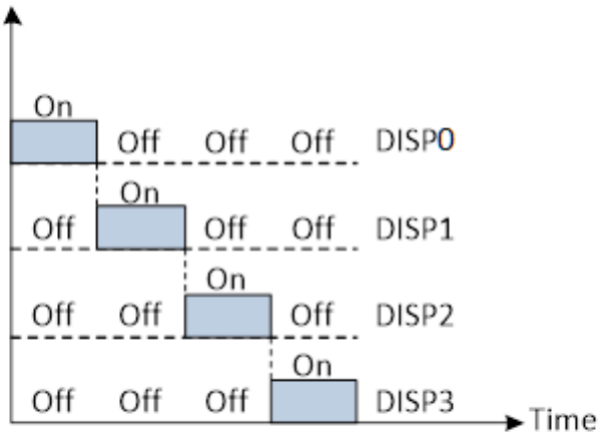

Figure 6: Graphical illustration to show characters on each 7-Segment displays simultaneously.

6 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]