Lab Sheet 6

Timers

1. Overview

Microcontrollers handle time-sensitive applications using timers. Timers are used for:

1. Measuring time between:
    a. Two externally generated events,
    b. An event generated by the microcontroller and an external one.
2. Generating delays,
3. Counting.

Two types of timers are available in the PIC16F84A microcontroller:

1. Timer0 module (TMR0),
2. Watchdog Timer (WDT).

2. Timer0 module (TMR0) – External Clock/Event

Program 1: First of all, create a project for this lab as was shown in the first week. Write the following program in the source code file:

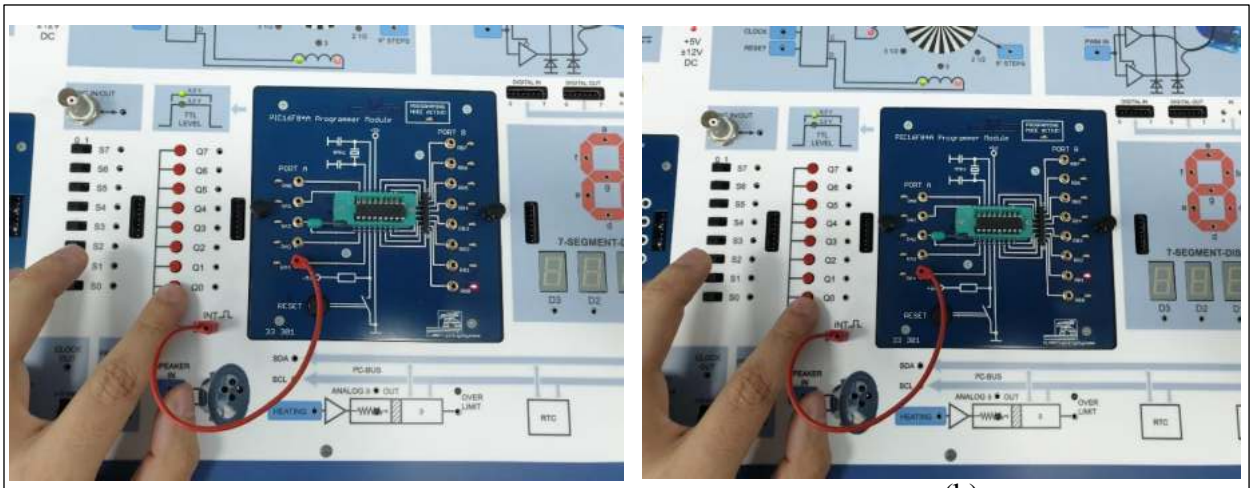
```
        CLRF    TMR0
LOOP    MOVF    TMR0, 0
        MOVWF   PORTB
        GOTO    LOOP
```

Code 1 – Code for Program 1

The program is used to count the number of presses of a button connected to RA4 (external clock/event input) and display this event on PORTB. Check Figure 1 below that describes PORTB after each press on external TMR0. After the first Press in Figure 1 (a), PORTB shows the value of 1. While after the second press in Figure 1 (b), PORTB shows the value of 2.

Exercise 1: Write the setup required for this code to behave correctly. To do some you must:

- Set up the ports of the microcontroller at the beginning of your program.
- Set up the timer functionality (OPTION_REG).



(a) (b)

Figure 1 – Program 1 executed on the board

Exercise 2: Modify the program in exercise 1 such that TMR0 is incremented once for every two presses of the button.

Timer0 module (TMR0) – Internal Clock

Program 2: Write the following program in the source code file:

```
                MOVLW   D'X'
                MOVWF   TMR0
                CLRF    PORTB
LOOP            GOTO    LOOP

ISR             BCF     INTCON, T0IF
                COMF    PORTB
                MOVLW   D'X'
                MOVWF   TMR0
                RETFIE
```

Code 2 – Code for Program 2

The program is used to toggle the state of PORTB every time the timer module overflows.

Exercise 4: Write the setup required for this code to behave correctly. To do some you must:

- Set up the ports of the microcontroller at the beginning of your program.
- Set up the timer functionality (OPTION_REG and X) such that it overflows every 50 µs.

(a) (b)

Figure 2 – Program 2 frequency measurement

Check Figure 2 above that takes a measurement of frequency from the board to the multimeter. With the default 'X' value code showing the actual frequency of the toggle on PORTB shown in Figure 2 (a) and the modified value of 'X' showing the new frequency of the toggle on PORTB in Figure 2 (b).

From Figure 2 shown above, for Exercise 4a answer the following questions:

- What is the expected frequency of the generated square wave?
- What is the measured frequency of the generated square wave?

From Figure 2 shown above, for Exercise 4b answer the following question:

- How would the value of X be changed for the correct frequency to be generated?

3. Report Exercises

a. Modify the program in exercise 4 such that it generates a 350 Hz square wave.