Introduction to Embedded Systems Lab **Total Marks: 35**
GCU,LAHORE **ROLL NO. ________________________**

Lab-9
PIC Counter Programming using assembly & C language

Objective:

In this lab students will learn

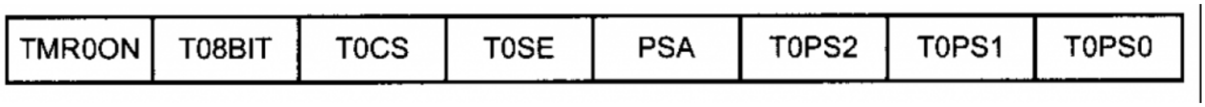
- How to program PIC18F4550 for event counting.
- How to designing a counter and display its contents on seven -segment display.
- How a buzzer can be turned ON and OFF using counters.

Theory

When the timer is used as a timer, the PIC18's crystal is used as the source of the frequency. When it is used as a counter, however, it is a pulse outside the PIC 18 that increments the TH, TL registers. In counter mode, notice that registers such as TOC0N, TMR0H, and TMR0L are the same as for the timer discussed in the Previous post; they even have the same names.

There are basically 5 timers available in PIC18 and they can be programmed either as counters or timers. It all depends on their triggering source. If PIC18's crystal is used as the source of frequency then it will act as a timer. If some external pulse causes an increment in the values of TH/TL then it acts as a counter.

What is the T0CON register?



TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0

T0CON register [/caption]

This is an 8 bit register and it is used for configurating the timers of PIC. Each bit has its specific role assigned to it.

The first three bits are used for setting the values of prescaler. Timers can be programmed in either 16 bits mode or 8 bits mode. So the maximum attainable value is 256 or 65535. But it we want to introduce larger delays then we can do this using prescaler idea. In such cases, the OSC frequency is further divided by the prescaler value.

0	0	0	1:2	FOSC/4/2
0	0	1	1:4	FOSC/4/4
0	1	0	1:8	FOSC/4/8

0	1	1	1:16	FOSC/4/16
1	0	0	1:32	FOSC/4/32
1	0	1	1:64	FOSC/4/64
1	1	0	1:128	FOSC/4/128
1	1	1	1:256	FOSC/4/256

The next bit is PSA: Prescaler Assignment bit. This bit decides either to bypass the prescaler or use it for calculating the delays. If it is equal to 1, then timer0 clock input bypasses the prescaler.

T0CS: timer 0 clock source select bit.

1=External clock will increment the timers registers value (RA4 pin/T0CK input pin)

0=Internal clock source will be used (FOSC/4).

So 4th pin of PORTA is used as a triggering and counting source for timer 0 and similarly for timer1 0 pin of PORTC (RC0) is used. The clock pulses are fed through these pins foe incrementing the count.

In contrast, when T0CS = 1, the timer is used as a counter and gets its pulses from outside the PIC 18. Therefore, when T0CS = 1, the counter counts up as pulses are fed from pin RA4 (PORTA.4). The pin is called T0CKI (Timer0 clock input). Notice that the pin belongs to Port A. In the case of Timer0, when T0CS =1, pin RA4 (PORTA.4) provides the clock pulse and the counter counts up for each clock pulse coming from that pin. Similarly, for Timer 1, when TMR1 CS = 1, each clock pulse coming in from pin RC0 (PORTC.0) makes the counter count up.

Exercise

Assume that clock pulses are fed into pin T0CK1, write a program for counter 0 in 8-bit mode to count the pulses and display the state of the TMR0L count to PORTB. Set the initial count to 0. Show the simulations results on PROTEUS. [5]

For the above problem, show the data on seven segment display. Show the simulations results on PROTEUS. [5]

Assuming that clock pulses are fed into pin T0CK1 and a buzzer is connected to pin PORTB.1, write a program for counter 0 in 8-bit mode to sound the buzzer every 100 pulses. Also show simulations on Proteus. [5]

Assume that a 1-HZ frequency pulse is connected to input for Timer1 (pin PORTC.0). Write a a program to display the counter values of TMR1H, TMR1L on PORTB and D. Set initial values to 0. Use Timer1, 16 bit mode, no prescalar, and positive-edge clock. [5]

Write a C18 program to toggle all the bits of PORTB continuously with some delay. Use Timer0, 16-bit mode, and no prescalar options to generate the delay. Show simulations on Proteus. [5]

Program the timer0 in c language to generate a square waveform of 3KHz assuming XTAL=10MHz. Use PROTEUS simulation results to verify your answer. [5]

Conclusion [5]