İzmir Katip Çelebi University

Department of Electrical and Electronics Engineering

EEE316 Microprocessors

Spring 2020

Experiment VI

Interrupts, Timers and Counters

Pre-Lab Report

- Please study related topics in reference notes.
- Answer the questions under the lab activities. Prepare report in the specified format. Reports must be completed before coming to lab.
- Submit your report to CANVAS until May 6, 23:59.

Experimental Work

- Please explain your code step by step to instructors during lab hours.

Lab Objectives

- Embedded programming with C language
- Understanding of Interrupts
- Understanding of Timers and Counters

References

- Lecture notes
- Mazidi, McKinlay, Causey "PIC Microcontroller and Embedded Systems," Chapter 9, 10, 11

Lab Activities

1. Write a C code to count the packages on the production line. Let's assume that a laser checker is used to detect the packages which is connected to bit 0 of PORTB (RB0). (You can use a button to simulate the laser checker on PROTEUS.) The laser checker is sending "1" to RB0 when there is no package on the production line. However, it turns to "0" when a package is detected and it remains "0" until the package is completely passed through. There is a LED connected to bit 1 of PORTB (RB1) which is set to "off" normally. However, the LED turns on for five seconds after 10 packages are counted. Just make sure no packages are counted during the LED is on.
   Hints: You need to use two interrupt functions:
   - External hardware interrupt for the laser checker
   - Timer0 interrupt for counting the packages

2. In this experiment, you are asked to design an up- and down-counter. RB0 (PORTB.0) pin is fed with a square wave whose frequency change between 100 Hz and 300 Hz. You will program the RB0 behavior so that each time a positive (or negative) edge of the square wave occurs, an interrupt will be activated, and within the interrupt service routine a counter is incremented for the frequency values greater than 200 Hz. (i.e., between 200 Hz and 300 Hz) and decremented for the frequency values lower than 200 Hz. (i.e., between 200 Hz and 100 Hz). This counter will be displayed in the 7-segment display as a two-byte **hexadecimal** number. Also the increment and decrement frequencies of the counter will be calculated according to the input signal's frequency. Write this program code **in C** programming language.

This will be done in the following manner:

- At 200 Hz, counter will count up with a period of 1 second (increment frequency 1 Hz), i.e., will increment at every one second.
- At frequencies between 200 Hz and 300 Hz, the counter will continue to count up with a **linearly** increasing frequency up to 5 Hz (200 ms). In other words, when input frequency changes from 200 Hz to 300 Hz, the increment frequency of the counter should change from 1 Hz to 5 Hz linearly.

(Upcounter frequency = 1 Hz @ 200 Hz, Upcounter frequency = 5 Hz @300 Hz)

- At frequencies between 200 Hz and 100 Hz, the counter will start to count down with a **linearly** increasing frequency up to 5 Hz (200 ms) @100 Hz., i.e., the counter will change at every 200 ms. counting down @100 Hz.

(Downcounter frequency = 1 Hz @200 Hz, Downcounter frequency = 5 Hz @100 Hz)

In the main loop of your program, you will read the content of the counter and display on the 7-segment display. The initial value for your two byte counter is 0x7777, i.e., you will display '7777' on the seven segment display initially.

Note that in this experiment, you are performing two tasks at the same time. One is the main loop where you perform a 7-segment display action. Second is in the background with the edges of the square waves, without distracting what is done in the main loop, a counter is incremented or decremented to be displayed in the main loop.

Hints for Algorithm:

The square wave signal obtained from function generator is fed to RB0 which can be considered as external hardware interrupt. You can benefit from Section 11.3 and program 11-4C in your reference book for implementation.

You need to measure frequency of input signal to decide whether you need to count down or up, and also to calculate the incrementing or decrementing frequency. To do so, internal clock source can be used as a reference. Timer register gives you the number of ticks (Check Section 9.1 for details). So you can get the value of ticks from timer register between two rising edges (interrupts) of the input signal. However, you should consider

that the timer works with $F_{osc}/4$ clock. If our crystal frequency is 16 MHz, then we have 16 MHz/4= 4 MHz as the timer frequency.
Timer in PIC18 are 16 bits wide which means its values changes between 0000 to FFFF. Let's assume Timer0 is chosen in this application and crystal frequency is 16 MHz as default. We know that frequency of the input signal is very low (between 100 – 300 Hz) compared to the crystal frequency. So we are sure that from first interrupt (first rising edge of the input signal) to next interrupt (second interrupt), timer register counts up rapidly because of the many rising edges of crystal frequency. Let's say when interrupt comes, we read "3E80" (in decimal 16000) from timer registers. As the crystal frequency is 16 MHz, the timer frequency is 16 MHz/4= 4 MHz. If we divide 4MHz to "3E80" or 16000 in decimal, we will get 4 000 000 / 16 000 = 250. So the frequency of the input signal is 250. Since it is bigger than 200, you need to increment the counter. Then you need to calculate the frequency of increment which 2.5 Hz for 250.

The key point of the algorithm for this activity is to figure out how to calculate frequency of the input signal which is explained above.

Hint : You can examine "PROJECT 10.4 – Event Counter Using 4-Digit 7-Segment LED Display" in Doğan İbrahim's Using LEDs, LCDs, and GLCDs Book.