ENGR 270 LAB #5 – Timer

Objective
Using Timers to schedule tasks and Pulse Width Modulation(PWM).

Related Principles
- Computer Organization and Design
- Microprocessors
- Hardware and Software Interface
- Digital Design
- Assembly language

Equipment
- Windows-based PC with MPLAB Simulation Solutions Software
- USB hard disk or other removable drives
- Microchip PICKit programmer
- EDbot Micro V12 Platform

Preparation/Background
Prior to start of this lab, you are expected to have completed all prior labs successfully and have reviewed Chapters 2, 4 and 5 of *Computer Organization and Microprocesso*r textbook.

Following example code demonstrates the use of Timer 0 in order to generate an interrupt every one milli-second. Using the interrupt code to change LED light color every 0.5 seconds.

```
;-------------------------------------------------------------------------------
; File: Lab5Exp1.asm
; Desc: Timer Example- Demonstrates use of Timer 0
;
;  Use Timer 0 to step Red, Green and Blue of RGB LED thorugh the
;  following sequence (each step should be Approx. 0.5 seconds)
;  R G B
;  0 0 1
;  0 1 0
;  1 0 0
;  "repeat"
;
; Last Update: Janaury, 2019
; Auth: Class
;-------------------------------------------------------------------------------

   list    p=18F1220          ;processor type
   radix   hex                ;default radix for data
   ; Disable Watchdog timer, Low V. Prog, and RA6 as I/O
   config  WDT=OFF,LVP=OFF,OSC=INTIO2

   #include    p18F1220.inc            ;header file

   #define          dCount      0x80
   #define          dCountInner 0x81
   #define          LastValue   0x82
   #define          StateLSB    0x83  ;  State variable is 16 bit and increments
   #define          StateMSB    0x84  ;  and increments every 1 msec.
   #define          WregBack    0x85


   org       0x000            ; Executes after rest
   GOTO      StartL

   org       0x008            ; Executes after high priority interrupt
   GOTO      HPRIO

   org       0x20             ; Code start here
```

```
StartL:
   ; initialize all I/O ports
   CLRF    StateLSB        ; clear LSB of current state variable
   CLRF    StateMSB        ; clear MSB of current state variable
   CLRF    PORTA           ; Initialize PORTA
   CLRF    PORTB           ; Initialize PORTB
   MOVLW   0x7F
   MOVWF   ADCON1          ; Configure PortA<0:7> Digital and PortA<> Analog
   MOVLW   0x35
   MOVWF   TRISA           ; Set Port A direction Per EDbot Micro Spec.
   MOVLW   0xC3
   MOVWF   TRISB           ; Set Port A direction Per EDbot Micro Spec.
   MOVLW   0x60
   IORWF   OSCCON          ; Set internal System Clock to 4 Mhz

   ; Wait until INT0 Button is pressed (include SW Debounce)
   Call Int0Press

   ; Enable INT0 and INT1
   BSF     INTCON, PEIE            ; enable all peripheral interrupts
   BSF     INTCON, TMR0IE          ; enable Timer 0 Interrupt
   BSF     INTCON2, TMR0IP         ; Set Timer 0 Interrupt to High priroity
   BSF     RCON, IPEN              ; enable priority levels on interrupts
   BCF     INTCON, TMR0IF          ; clear Timer 0 Interruplt flag
   MOVLW   0x41            ; 8-bit, internal clock, 1:4 scale
   MOVWF   T0CON                   ;Timer 0 tick is 4 usec
   CLRF    TMR0L                   ; 256 count or 1 msec to Timer 0 Interrupt
   BSF     T0CON,TMR0ON            ; enable TMR0
   BSF     INTCON, GIE             ; enable interrupts globally
   BSF     PORTA,3                 ; Turn Red LED on
   BSF     PORTB,5                 ; Turn Green LED on
   BSF     PORTB,2                 ; Turn Blue LED on
MainL:                             ; Main loop
   BRA     MainL

; Delay function waits for (Wreg/10) seconds before returning
Delay:
   MOVWF   dCount
DelayLoop:
   CALL    DelayOnce
   DECF    dCount
   BNZ     DelayLoop
   RETURN
DelayOnce:
   CLRF    dCountInner     ; Internal delay loop
DelayOnceLoop:
   NOP
   INCF    dCountInner
   BNZ     DelayOnceLoop
   RETURN  ; Delay Function

   ; Wait until INT0 Button is pressed (include SW Debounce)
Int0Press:
   MOVLW   .1
   CALL    Delay
   MOVF    PORTB,0
   ANDLW   0x01
   BZ      Int0Press ; wait for button to be released
Int0PressZ:
   MOVLW   .1
   CALL    Delay
   MOVF    PORTB,0
   ANDLW   0x01
   BNZ     Int0PressZ ; wait for buton to be pressed
   RETURN  ; Int1Press Function

; Interrupt Handeling Section
```

```
HPRIO:     ; High priority interrupts including Timer 0 Int.
   MOVWF   WregBack         ; Save Wreg.
   BTFSC   INTCON, TMR0IF  ; Check for Timer 0 high priority Interrupt
   BRA     TMR0int
   BRA     HPRIOdone        ; return from interrupt

TMR0int:   ; handel Timer 0 Interrupt
   ; Increment 16-bit variable state every 1 mSec.
   INCF    StateLSB
   TSTFSZ  StateLSB
   BRA     TMR0cont
   INCF    StateMSB
TMR0cont:  ; Start of code to be executed after every TMR 0 Int.
   MOVLW   .2
   CPFSEQ   StateMSB
   BRA      TMR0Step2
   ; set RGB to 001
   BCF      PORTA,3                ; Red
   BCF      PORTB,5                ; Green
   BSF      PORTB,2                ; Blue
   BRA      HPRIOdone
TMR0Step2:
   MOVLW   .4
   CPFSEQ   StateMSB
   BRA      TMR0Step3
   ; set RGB to 010
   BCF      PORTA,3                ; Red
   BSF      PORTB,5                ; Green
   BCF      PORTB,2                ; Blue
   BRA      HPRIOdone
TMR0Step3:
   MOVLW   .6
   CPFSEQ   StateMSB
   BRA      HPRIOdone
   ; set RGB to 100
   BSF      PORTA,3                ; Red
   BCF      PORTB,5                ; Green
   BCF      PORTB,2                ; Blue
   CLRF    StateLSB                ; reset State variable
   CLRF    StateMSB

HPRIOdone:  ; code to clean up and return from interrupt
   BCF      INTCON, TMR0IF         ; Clear Timer 0 interrupt Flag
   MOVF    WregBack,0              ; Recover Wreg.
   RETFIE

 end                        ; end program
```

Experiment

Write an assembly code that repeatedly cycles RGB LED from red to green to blue (holding each color for 1.5 seconds) in main code. Use the timer functionality to cycle one motor at a time through the following steps (0.5 seconds per step):

1. Right motor forward at 100% power
2. Reduce power to 50%
3. Left motor backward at 100% power
4. Reduce power to 50%
5. Left motor forward at 100% power
6. Reduce power to 50%
7. Right motor backward at 100% power
8. Reduce power to 50%
9. Go to step 1

It is recommended that you experiment with provided sample code to gain an understanding of Timers prior to working on this experiment.

This experiment requires that you review your high level design (flow chart or pseudo code) and demonstrate your system to the instructor upon completion. Include the approval signature in your report.

Team Members: • • • •	LAB5 Demo Instructor Approval Signature & Date:

Report Requirements

All reports must be computer printed (formulas and diagrams may be hand drawn) and at minimum include:

For each experiment:

a) Clear problem statement; specify items given and to be found.
b) Specific responses to each question asked in the experiment.
c) Documentation of resulting high level design, disassembled code, system diagram, schematics and any other supporting material.

For the report as a whole

a) Cover sheet with your name, course, lab title, date of completion and your teammates' name.
b) Lessons learned from this lab.
c) A new experiment and expected results which provide additional opportunity to practice the concepts in this lab.