LAB 4 RAPORT

Creator: Sampo Kajaste 1881563

SISÄLTÖ

1 GOAL

In this lab, you work with interrupt. You learn how to write an Interrupt Service Routine (ISR) to handle timer interrupts.

1.1 Objectives

- Read chapter 11 of the textbook.
- Write assembly code for sections 4.4. You should email your assembly code to your lab and course instructors before the start of the lab.

1.2 Objectives

Upon completion of this lab, you will be able to:

- Understand the behaviour of interrupts.
- What is an ISR and how to write it.
- How to measure frequency of a pulse signal using a combination of timer and interrupt

1.3 Interrupt

The PIC board has an LCD with two lines and 16 characters (2×16 LCD). The LCD is connected to the PIC through an SPI I/O expander: MCP23S17. MCP23S17 receives commands and data from microcontroller for LCD through SPI port. Then, it converts serial bits into bytes and send 2 them to the LCD. The course website has a sample program for LCD. Figure 5.1 shows the main() function of the program. Before sending any data to the LCD, the main() function initializes the LCD through LCDInit(). Then, it configures SPI port to communicate with MCP23S17. Inside the while() loop, "Hello PIC18" is sent to the LCD. void main(void) { // Initialize the LCD display LCDInit(); TXSTA = 0b10100100; SPBRG = 0xff; RCSTA = 0b10010000; while(1) { // Write the command to start on line 1 LCDLine_1(); // Write the data one char at a time. d_write('H'); d_write('e'); d_write('l'); d_write('l'); d_write('0'); // Write the command to start on line 2 LCDLine_2(); // Write the data to line 2 one char at a time d_write('P'); d_write('I'); d_write('C'); d_write('1'); d_write('8'); delay_1s(); } } Figure 5.1 Sample program for LCD In PIC Explorer board, temperature sensor is connected to RA1. Configure RA1 as input to the A/D. Write a C program that generates interrupt when A/D finishes the conversion. Inside the interrupt service routine, read digital value of temperature from A/D and send it to the LCD.

1.4 Frequency Measurement

In this part, you write an assembly program to measures frequency of a square wave. Use a function generator to generate a 1-KHz square wave. Connect the function generator to RA4. Figure 1 shows counter0 and timer3 that you will use in this section of the lab. 2 Counter0: The clock of counter0 is connected to RA4. Configure counter0 to increment on the rising edge of the clock. Timer3: Timer3 should be configured to generate 1-second intervals using interrupt. After 1- seond, read counter0 (TMR0L, TMR0H). The value of counter0 shows frequency of the 1-kHz signal. Timer3 sends measured frequency to the output LEDs. (hint1: when you read from TMR0, you need to read from TMR0L first, and then read from TMR0H. Hint2: Counter0 is 16- bit but the board has 8-LED. Timer3 should send low and high bytes of Counter0 to the LEDs, alternatively. Hint3: set T1OSCEN in T1CON)

2 CODE

```
; T3CON setup:
 ; TMR3ON = 0
 ; TMR3CS = 1
; T1SYNC = 0
 ; T3CKPS = 00
 ; T3CCP = 00
 ; RD16 =0

 ; T0CON setup:
 ; TMR0ON = 0
 ; T08BIT = 0
 ; T0CS = 1
 ; T0SE = 0
 ; PSA = 1
 ; T0PS = 000
 ;T3CON = 0x80
 ;T0CON = 0x28

 ;TMR3H&TMR3L: RegValue =
 ;32kHz/4 = 8kHz
 ;1s= (1/(8khz))*(FFFF)-init ➔ 1000000 * 0,000125=FFFF-init
 ;init=65536-125=65410 = FF82
hilo equ 0x20
 org 0x0
 goto start
org 0x08
 goto ISR
```

```
ISR:
 ;check which is making the interrupt
 ;1second is up go to interrupt for 3
 btfsc PIR2,TMR3IF
 bra timer3isr
 ;interrupt for timer0 womething went wrong? start over
 btfss INTCON,TMR0IF
 bra start
 retfie

timer3isr:
 ;stop timers for duration of sending values to leds
 bcf INTCON,GIE
 bcf T3CON, TMR3ON
 bcf T0CON, TMR0ON
 ;testi which value should be sent 0=High 1=low
 btfss hilo, 0
 bra lowbyte
 ;send first lowbyte the high byte
 movff TMR0L,PORTD
 movff TMR0H,PORTD
 ;setting hilo for next round
 bsf hilo,0
 bra ending
lowbyte:
;only lowbyte needs to be sent
movff TMR0L,PORTD
;setting hilo for next round
```

```
bcf hilo,0
ending:
 ;restart the values on timers to do it again
bcf PIR2, TMR3IF
bsf INTCON,GIE
movlw 0xff
movwf TMR3H
movlw 0x82
movwf TMR3L
clrf TMR0H
clrf TMR0L
;Start timers again
bsf T3CON,TMR3ON
bsf T0CON,TMR0ON
retfie

intsetup:
 ;set ports
 clrf TRISD
 bsf TRISA,4
 ;timer1 values need to be set for timer3
 bsf T1CON,TMR1CS
 ;clear values for timer 0 to begin counting
 clrf TMR0H
 clrf TMR0L
 ;interrupt setup
 bcf T0CON, TMR0IF
 bsf INTCON,TMR0IE
 bcf PIR2, TMR3IF
```

```
bsf PIE2, TMR3IE
bsf INTCON, PEIE
bsf INTCON, GIE
;initial values for counter and timer
movlw 0x28
movwf T0CON
movlw 0x80
movwf T3CON
movlw 0xff
movwf TMR3H
movlw 0x82
movwf TMR3L
return
start:
clrf hilo
call intsetup
;turn on counters and stay on busy loop interrupts should handle the value resets
bsf T0CON,TMR0ON
bsf T3CON,TMR3ON
bra $
end
```

3 SELF-REFLECT

This assignment was hard to understand how it worked. I wasn't sure if the timers can just be left on and end the code by looping. The conversion from Hertz to time had also its problems since I had no previous knowledge in this.