**THE CITY COLLEGE OF NEW YORK**

160 Convent Avenue, New York, NY 10031

*The City University of New York*

**EE 42500 Computer Engineering Lab**

**Experiment 4**

**Interrupt Service Routines**

**Prof. Hakan Peckan**

Students: Jonathan Martinez & Maciej Jachimowicz

Summer 2020

## Introduction

In this lab we will be implementing our knowledge about programing the microcontroller and we will be analyzing a program that utilizes the interrupt feature of the microcontroller. This is an important feature as external interrupts at any given moment may be of importance to a programmer. PORTB of the microcontroller is what allows us to make use of such features given that PORTB is a bidirectional I/O port.

## Analysis

The code has configured the interrupt functions in the Initial subroutine portion of code which can be referenced at the end of this report, we now look at the vectors to further understand.

org 0x0000 ;Reset vector

nop

goto Mainline

org 0x0008 ;High priority interrupt vector

goto HPISR ;execute High Priority Interrupt Service Routine

org 0x0018 ;Low priority interrupt vector

goto LPISR ;execute Low Priority Interrupt Service Routine

Here we see that a priority interrupt has been written at address 0x0008 that redirects to subroutine HPISR(high priority interrupt service routine) and we also see that a low priority interrupt has been written at address 0x0018 which redirects to subroutine LPISR.

The HPISR and LPISR are as follows:

;;;;High Priority Interrupt Service Routine;;;;

HPISR

bcf PORTE,RE2 ;Stop pulse train from RE2

bcf PORTA,RA3

bcf PORTA,RA2

bcf PORTA,RA1

HPLoop

rcall Analog2Dig ; start Analog to Digital Conversion

;DISPLAY ADRESH

movlw B'00000000'

cpfseq ADRESH

bra HPLoop

bcf INTCON,INT0IF

retfie FAST

;;;;Analog-to-Digital Conversion Code;;;;

Analog2Dig

bsf ADCON0,1

ADCloop

btfsc ADCON0,1

bra ADCloop

return

In the HPISR we see that all the bits on port A are cleared until the next stage of the program execution, If HP interrupt was fired during the LP interrupt the bits are all cleared and in the next cycle of the LP interrupt, it resumes where it left off.

;;;;Low Priority Interrupt Service Routine;;;;

LPISR

movff STATUS, STATUS\_TEMP ; save STATUS and W

movf W,WREG\_TEMP

bcf PORTE, RE2 ; Stop pulse train from RE2

rcall LEDSteps ; Blink the LEDs

bcf PORTA, RA3

bcf PORTA, RA2

bcf PORTA, RA1

movf WREG\_TEMP,W

movff STATUS\_TEMP,STATUS

bcf INTCON3,INT1IF

retfie

return

;;;;;LEDDelay;;;;;;

Delay1ms

MOVLF 10,TIMECOUNT

DelayLoop

rcall LoopTime

decf TIMECOUNT,F

bnz DelayLoop

MOVLF 10, TIMECOUNT

return

;;;;;;; LEDSteps;;;;;;;;;;;;;;;;;;;

LEDSteps

;;;Count up from STEP 1 to STEP 8

bsf PORTA,RA3

bsf PORTA,RA2

bsf PORTA,RA1

rcall Delay1ms

bsf PORTA,RA3

bsf PORTA,RA2

bcf PORTA,RA1

rcall Delay1ms

bsf PORTA,RA3

bcf PORTA,RA2

bsf PORTA,RA1

rcall Delay1ms

bsf PORTA,RA3

bcf PORTA,RA2

bcf PORTA,RA1

rcall Delay1ms

bcf PORTA,RA3

bsf PORTA,RA2

bsf PORTA,RA1

rcall Delay1ms

bcf PORTA,RA3

bsf PORTA,RA2

bcf PORTA,RA1

rcall Delay1ms

bcf PORTA,RA3

bcf PORTA,RA2

bsf PORTA,RA1

rcall Delay1ms

bcf PORTA,RA3

bcf PORTA,RA2

bcf PORTA,RA1

rcall Delay1ms

return

Here we see that the low priority points to the a subroutine called LEDsteps where the ports act as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| STEP | D4 | D5 | D6 |
| 1 | on | on | on |
| 2 | on | on | off |
| 3 | on | off | on |
| 4 | on | off | off |
| 5 | off | on | on |
| 6 | off | on | off |
| 7 | off | off | on |
| 8 | off | off | off |

Table 1

We take a quick look at the schematic for the PIC18 microcontroller:

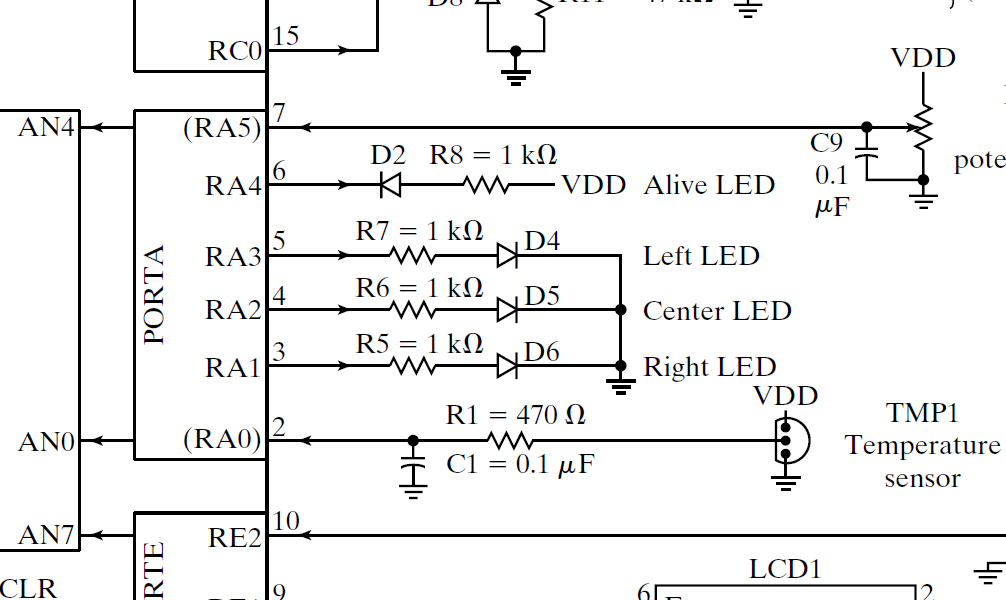


Figure 1

From table 1 and figure 1 we see that D4,D5 and D6 correspond to RA3, RA2 and RA1 respectively.

## Code Execution

After compiling and executing we set up the interrupt buttons to use during the code execution.

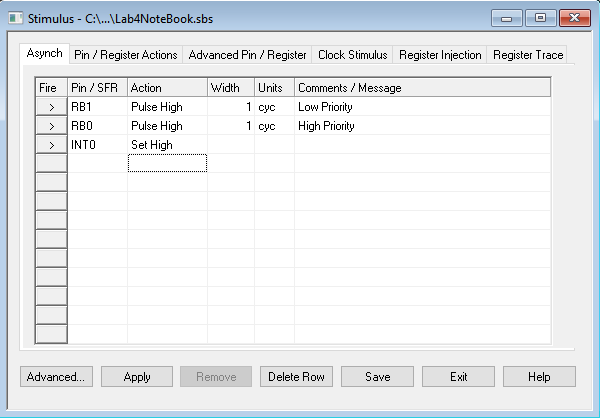


Figure 2

We run the code while monitoring RE2, RA1,RA2, RA3, RB0, INT0, RB1 and INT1. While the code is running we also “fire” the interrupts in ways that allow us to test the code to make sure it works as intended. This is the result:

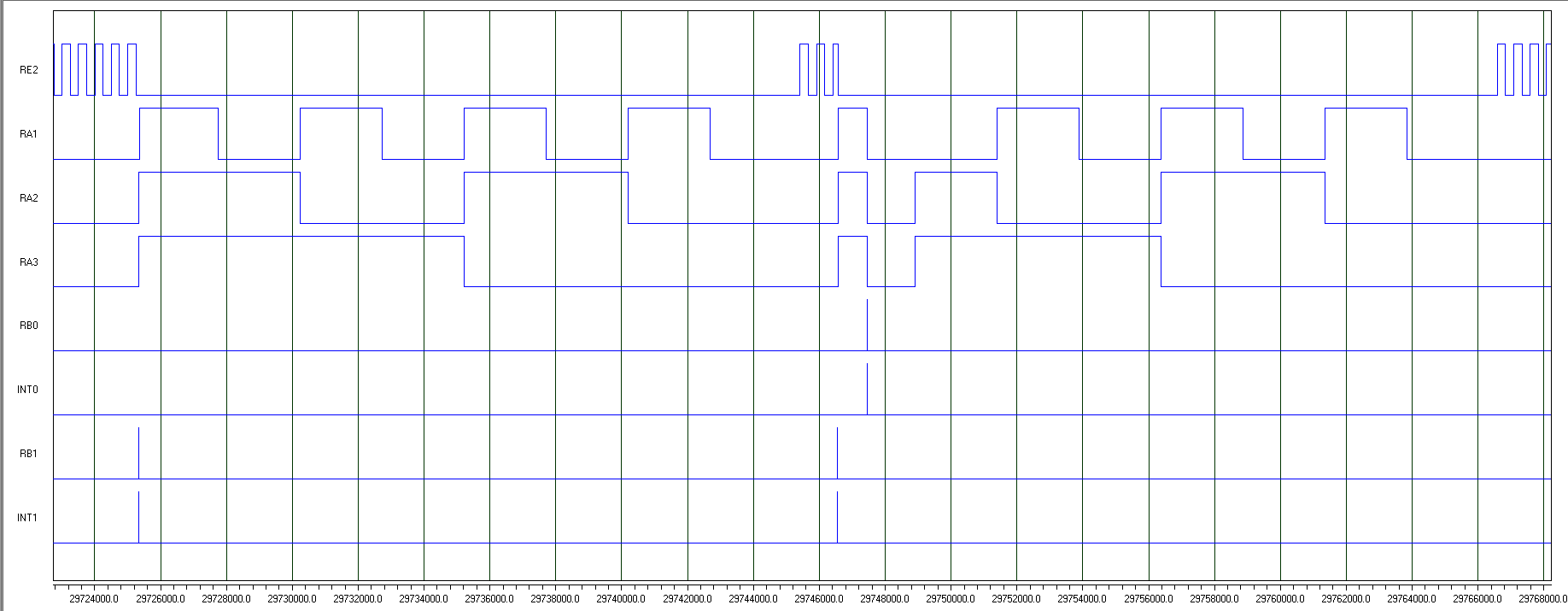


Figure 3

There are many things we can look at in figure3, to better understand how this works I will highlight some key features in figure 4.

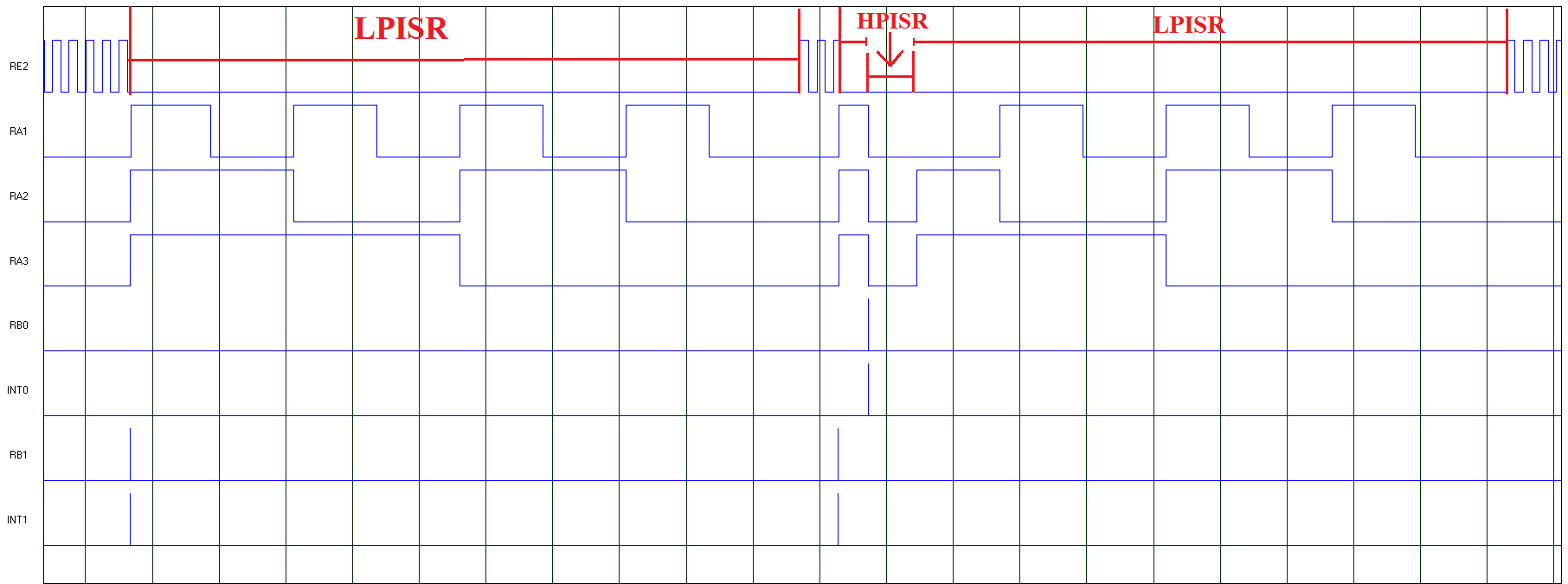


Figure 4

## Conclusion

The first thing to note is that in between the low priority interrupts we see that RE2 behaves as intended with its 50% duty cycle and a half period of 0.1ms. This is what we consider our “main program”. During the first LP interrupt we also see that PORTA follows the signal sequence from table 1. This program effectively counts down from 7 to 0 in binary on portA. However, if a high priority interrupt is “fired” the output of the port becomes 0, because all the bits of interest go into the “off” state. This is a temporary state of the port because the counter will continue after as if nothing happened, the LPISP keeps counting down as is shown in figure 4. We must take a look at the interrupt pins labeled INT0 and INT1. INT0 corresponds to the interrupt signal with address 0x0008 which sends the code execution into the HPISR, this is seen in the vectors mentioned earlier. We also see that INT1 corresponds to the interrupt signal with address 0x0018 which sends the code execution into subroutine LPISP. This is a key feature of the PIC18 microcontroller which can be very useful, essentially we can interrupt a code execution at any given moment and perform any given task. This concludes the experiment.

## Code Used

;;;;;;; P2 for QwikFlash board ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;

; Use 10 MHz crystal frequency.

; Use Timer0 for ten millisecond looptime.

; Blink "Alive" LED every two and a half seconds.

; Display PORTD as a binary number.

; Toggle C2 output every ten milliseconds for measuring looptime precisely.

;

;;;;;;; Program hierarchy ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;

; Mainline

; Initial

; InitLCD

; LoopTime

; BlinkAlive

; ByteDisplay (DISPLAY macro)

; DisplayC

; T40

; DisplayV

; T40

; LoopTime

;

;;;;;;; Assembler directives ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

list P=PIC18F4520, F=INHX32, C=160, N=0, ST=OFF, MM=OFF, R=DEC, X=ON

#include <P18F4520.inc>

\_\_CONFIG \_CONFIG1H, \_OSC\_HS\_1H ;HS oscillator

\_\_CONFIG \_CONFIG2L, \_PWRT\_ON\_2L & \_BOREN\_ON\_2L & \_BORV\_2\_2L ;Reset

\_\_CONFIG \_CONFIG2H, \_WDT\_OFF\_2H ;Watchdog timer disabled

\_\_CONFIG \_CONFIG3H, \_CCP2MX\_PORTC\_3H ;CCP2 to RC1 (rather than to RB3)

\_\_CONFIG \_CONFIG4L, \_LVP\_OFF\_4L & \_XINST\_OFF\_4L ;RB5 enabled for I/O

errorlevel -314, -315 ;Ignore lfsr messages

;;;;;;; Variables ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

cblock 0x000 ;Beginning of Access RAM

TMR0LCOPY ;Copy of sixteen-bit Timer0 used by LoopTime

TMR0HCOPY

INTCONCOPY ;Copy of INTCON for LoopTime subroutine

COUNT ;Counter available as local to subroutines

ALIVECNT ;Counter for blinking "Alive" LED

BYTE ;Eight-bit byte to be displayed

BYTESTR:10 ;Display string for binary version of BYTE

WREG\_TEMP

STATUS\_TEMP

PORTA\_TEMP

ADCON0\_TEMP

TIMECOUNT

endc

;;;;;;; Macro definitions ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

MOVLF macro literal,dest

movlw literal

movwf dest

endm

POINT macro stringname

MOVLF high stringname, TBLPTRH

MOVLF low stringname, TBLPTRL

endm

DISPLAY macro register

movff register,BYTE

call ByteDisplay

endm

;;;;;;; Vectors ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

org 0x0000 ;Reset vector

nop

goto Mainline

org 0x0008 ;High priority interrupt vector

goto HPISR ;execute High Priority Interrupt Service Routine

org 0x0018 ;Low priority interrupt vector

goto LPISR ;execute Low Priority Interrupt Service Routine

;;;;;;; Mainline program ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Mainline

rcall Initial ;Initialize everything

;MAIN LOOP BELOW

L1

btg PORTE,RE2 ;Toggle pin, to support measuring loop time

rcall LoopTime ;Make looptime be ten milliseconds

bra L1

;;;;;;; Initial subroutine ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;

; This subroutine performs all initializations of variables and registers.

Initial

MOVLF B'00010011',ADCON0

MOVLF B'10001110',ADCON1 ;Enable PORTA & PORTE digital I/O pins

MOVLF B'01000111',ADCON2

MOVLF B'11100001',TRISA ;Set I/O for PORTA

MOVLF B'11011111',TRISB ;Set I/O for PORTB

MOVLF B'11010000',TRISC ;Set I/0 for PORTC

MOVLF B'00001111',TRISD ;Set I/O for PORTD

MOVLF B'00000000',TRISE ;Set I/O for PORTE

MOVLF B'10001000',T0CON ;Set up Timer0 for a looptime of 10 ms

MOVLF B'00010000',PORTA ;Turn off all four LEDs driven from PORTA

bsf RCON,IPEN ;Enable high/low interrupt structure

bsf INTCON,GIEH ;Enable high-priority interrupts

bsf INTCON,GIEL ;Enable low-priority interrupts

bsf INTCON2, INTEDG0

bsf INTCON2, INTEDG1

bcf INTCON,INT0IF

bcf INTCON3, INT1IP

bsf INTCON3, INT1IE

bsf INTCON, INT0IE

bcf INTCON3, INT1IF

return

;;;;;;; LoopTime subroutine ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Bignum equ 65536-250+12+2

LoopTime

btfss INTCON,TMR0IF ;Wait until ten milliseconds are up

bra LoopTime

movff INTCON,INTCONCOPY ;Disable all interrupts to CPU

bcf INTCON,GIEH

movff TMR0L,TMR0LCOPY ;Read 16-bit counter at this moment

movff TMR0H,TMR0HCOPY

movlw low Bignum

addwf TMR0LCOPY,F

movlw high Bignum

addwfc TMR0HCOPY,F

movff TMR0HCOPY,TMR0H

movff TMR0LCOPY,TMR0L ;Write 16-bit counter at this moment

movf INTCONCOPY,W ;Restore GIEH interrupt enable bit

andlw B'10000000'

iorwf INTCON,F

bcf INTCON,TMR0IF ;Clear Timer0 flag

return

;;;;;LEDDelay;;;;;;

Delay1ms

MOVLF 10,TIMECOUNT

DelayLoop

rcall LoopTime

decf TIMECOUNT,F

bnz DelayLoop

MOVLF 10, TIMECOUNT

return

;;;;;;; LEDSteps;;;;;;;;;;;;;;;;;;;

LEDSteps

;;;Count up from STEP 1 to STEP 8

bsf PORTA,RA3

bsf PORTA,RA2

bsf PORTA,RA1

rcall Delay1ms

bsf PORTA,RA3

bsf PORTA,RA2

bcf PORTA,RA1

rcall Delay1ms

bsf PORTA,RA3

bcf PORTA,RA2

bsf PORTA,RA1

rcall Delay1ms

bsf PORTA,RA3

bcf PORTA,RA2

bcf PORTA,RA1

rcall Delay1ms

bcf PORTA,RA3

bsf PORTA,RA2

bsf PORTA,RA1

rcall Delay1ms

bcf PORTA,RA3

bsf PORTA,RA2

bcf PORTA,RA1

rcall Delay1ms

bcf PORTA,RA3

bcf PORTA,RA2

bsf PORTA,RA1

rcall Delay1ms

bcf PORTA,RA3

bcf PORTA,RA2

bcf PORTA,RA1

rcall Delay1ms

return

;;;;Low Priority Interrupt Service Routine;;;;

LPISR

movff STATUS, STATUS\_TEMP ; save STATUS and W

movf W,WREG\_TEMP

bcf PORTE, RE2 ; Stop pulse train from RE2

rcall LEDSteps ; Blink the LEDs

bcf PORTA, RA3

bcf PORTA, RA2

bcf PORTA, RA1

movf WREG\_TEMP,W

movff STATUS\_TEMP,STATUS

bcf INTCON3,INT1IF

retfie

return

;;;;High Priority Interrupt Service Routine;;;;

HPISR

bcf PORTE,RE2 ;Stop pulse train from RE2

bcf PORTA,RA3

bcf PORTA,RA2

bcf PORTA,RA1

HPLoop

rcall Analog2Dig ; start Analog to Digital Conversion

;DISPLAY ADRESH

movlw B'00000000'

cpfseq ADRESH

bra HPLoop

bcf INTCON,INT0IF

retfie FAST

;;;;Analog-to-Digital Conversion Code;;;;

Analog2Dig

bsf ADCON0,1

ADCloop

btfsc ADCON0,1

bra ADCloop

return

;;;;;;; Constant strings ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

LCDstr db 0x33,0x32,0x28,0x01,0x0c,0x06,0x00 ;Initialization string for LCD

CODE1 db "\x80CODE \x00" ;Write "TESTING:" to first line of LCD

UNLOCKED db "\xc0UNLOCKED \x00"

LCDclear1 db "\x80 \x00"

LCDclear2 db "\xc0 \x00"

end