EECE.3170: Microprocessor Systems Design I

Spring 2016

Lecture 31: Key Questions

April 20, 2016

1. (Review) Explain how interrupts can be set up and managed in the PIC microcontrollers.

2. (Review) Explain the operation of the programs used to rotate the LEDs using interrupts (interrupt.asm and interrupt.c).

3. Explain how the analog to digital converter module is configured in PIC microcontrollers.

4. Explain the operation of the programs used to test the ADC (a2d.asm and a2d.c).

5. Explain the programs that use the ADC result to vary the speed of rotation (vs_rotate.asm and vs_rotate.c).

```
; *******************************************************************
; Lesson 4 - "Analog to Digital"
;
; This shows how to read the A2D converter and display the
; High order parts on the 4 bit LED display.
; The pot on the Low Pin Count Demo board varies the voltage
; coming in on in A0.
;
; The A2D is referenced to the same Vdd as the device, which
; is nominally is 5V.  The A2D returns the ratio of the voltage
; on Pin RA0 to 5V.  The A2D has a resolution of 10 bits, with 1024
; representing 5V and 0 representing 0V.
;
;
; The top four MSbs of the ADC are mirrored onto the LEDs. Rotate the potentiometer
; to change the display.
;
;
; PIC: 16F1829
; Assembler: MPASM v5.43
; IDE: MPLABX v1.10
;
; Board: PICkit 3 Low Pin Count Demo Board
; Date: 6.1.2012
;
; *******************************************************************
; * See Low Pin Count Demo Board User's Guide for Lesson Information*
; *******************************************************************

#include <p16F1829.inc>
     __CONFIG _CONFIG1, (_FOSC_INTOSC & _WDTE_OFF & _PWRTE_OFF & _MCLRE_OFF & _CP_OFF & _CPD_OFF &     ↙
    _BOREN_ON & _CLKOUTEN_OFF & _IESO_OFF & _FCMEN_OFF);
     __CONFIG _CONFIG2, (_WRT_OFF & _PLLEN_OFF & _STVREN_OFF & _LVP_OFF);

    errorlevel -302                     ;supress the 'not in bank0' warning

    ; -------------------LATC-----------------
    ; Bit#:  -7---6---5---4---3---2---1---0---
    ; LED:   ---------------|DS4|DS3|DS2|DS1|-
    ; -----------------------------------------


    ORG 0                               ;start of code at address 0x0000
Start:
                                        ;Setup main init
     banksel        OSCCON              ;bank1
     movlw          b'00111000'         ;set cpu clock speed
     movwf          OSCCON              ;move contents of the working register into OSCCON

                                        ;Configure the ADC/Potentimator
                                        ;already in bank1
     bsf            TRISA, 4            ;Potentimator is connected to RA4....set as input
     movlw          b'00001101'         ;select RA4 as source of ADC and enable the module (carefull, this ↙
    is actually AN3)
     movwf          ADCON0
     movlw          b'00010000'         ;left justified - Fosc/8 speed - vref is Vdd
     movwf          ADCON1
     banksel        ANSELA              ;bank3
     bsf            ANSELA, 4           ;analog for ADC

                                        ;Configure the LEDs
     banksel        TRISC               ;bank1
     clrf           TRISC               ;make all of PORTC an output
     banksel        LATC                ;select the bank where LATC is
     movlw          b'00001000'         ;start the rotation by setting DS1 ON
     movwf          LATC                ;write contents of the working register to the latch
```

```
MainLoop:
                                        ;Start the ADC
    nop                                 ;requried ADC delay of 8uS => (1/(Fosc/4)) = (1/(500KHz/4)) = 8uS
    banksel         ADCON0
    bsf             ADCON0, GO          ;start the ADC
    btfsc           ADCON0, GO          ;this bit will be cleared when the conversion is complete
    goto            $-1                 ;keep checking the above line until GO bit is clear

                                        ;Grab Results and write to the LEDs
    swapf           ADRESH, w           ;Get the top 4 MSbs (remember that the ADC result is LEFT justified↙
    !)
    banksel         LATC
    movwf           LATC                ;move into the LEDs
    bra             MainLoop

    end                                 ;end code
```

```
/**
 *******************************************************************
 * Lesson 4 - "Analog to Digital"
 *
 * This shows how to read the A2D converter and display the
 * High order parts on the 4 bit LED display.
 * The pot on the Low Pin Count Demo board varies the voltage
 * coming in on in A0.
 *
 * The A2D is referenced to the same Vdd as the device, which
 * is nominally is 5V.  The A2D returns the ratio of the voltage
 * on Pin RA0 to 5V.  The A2D has a resolution of 10 bits, with 1023
 * representing 5V and 0 representing 0V.
 *
 *
 * The top four MSbs of the ADC are mirrored onto the LEDs. Rotate the potentiometer
 * to change the display.
 *
 *  PIC: 16F1829
 *  Compiler: XC8 v1.00
 *  IDE: MPLABX v1.10
 *
 *  Board: PICkit 3 Low Pin Count Demo Board
 *  Date: 6.1.2012
 *
 * *******************************************************************
 * See Low Pin Count Demo Board User's Guide for Lesson Information*
 * ******************************************************************
 */

#include <htc.h>                                    //PIC hardware mapping
#define _XTAL_FREQ 500000                           //Used by the XC8 delay_ms(x) macro

//config bits that are part-specific for the PIC16F1829
__CONFIG(FOSC_INTOSC & WDTE_OFF & PWRTE_OFF & MCLRE_OFF & CP_OFF & CPD_OFF & BOREN_ON & CLKOUTEN_OFF &     ↙
    IESO_OFF & FCMEN_OFF);
__CONFIG(WRT_OFF & PLLEN_OFF & STVREN_OFF & LVP_OFF);

    /* -------------------LATC-----------------
     * Bit#:  -7---6---5---4---3---2---1---0---
     * LED:   ---------------|DS4|DS3|DS2|DS1|-
     *-----------------------------------------
     */

void main(void) {
    OSCCON = 0b00111000;                            //500KHz clock speed
    TRISC = 0;                                      //all LED pins are outputs

                                                    //setup ADC
    TRISAbits.TRISA4 = 1;                           //Potentiamtor is connected to RA4...set as input
    ANSELAbits.ANSA4 = 1;                           //analog
    ADCON0 = 0b00001101;                            //select RA4 as source of ADC and enable the module     ↙
    (AN3)
    ADCON1 = 0b00010000;                            //left justified - FOSC/8 speed - Vref is Vdd

    while (1) {
        __delay_us(5);                              //wait for ADC charging cap to settle
        GO = 1;
        while (GO) continue;                        //wait for conversion to be finished
        LATC = (ADRESH >> 4);                       //grab the top 4 MSbs
    }


}
```

```
; *******************************************************************
; Lesson 5 - "Variable Speed Rotate"
;
; This lesson combines all of the previous lessons to produce a variable speed rotating
; LED display that is proportional to the ADC value. The ADC value and LED rotate
; speed are inversely proportional to each other.
;
; Rotate the POT counterclockwise to see the LEDs shift faster.
;
;
; PIC: 16F1829
; Assembler: MPASM v5.43
; IDE: MPLABX v1.10
;
; Board: PICkit 3 Low Pin Count Demo Board
; Date: 6.1.2012
;
; *******************************************************************
; * See Low Pin Count Demo Board User's Guide for Lesson Information*
; *******************************************************************

#include <p16F1829.inc>
     __CONFIG _CONFIG1, (_FOSC_INTOSC & _WDTE_OFF & _PWRTE_OFF & _MCLRE_OFF & _CP_OFF & _CPD_OFF &      ↙
    _BOREN_ON & _CLKOUTEN_OFF & _IESO_OFF & _FCMEN_OFF);
     __CONFIG _CONFIG2, (_WRT_OFF & _PLLEN_OFF & _STVREN_OFF & _LVP_OFF);

    errorlevel -302                     ;supress the 'not in bank0' warning
    cblock 0x70                         ;shared memory location that is accessible from all banks
Delay1                                  ;Define two file registers for the delay loop in shared memory
Delay2
     endc

    ; -------------------LATC-----------------
    ; Bit#:  -7---6---5---4---3---2---1---0---
    ; LED:   ---------------|DS4|DS3|DS2|DS1|-
    ; -----------------------------------------


    ORG 0                               ;start of code
Start:
                                        ;Setup main init
     banksel        OSCCON              ;bank1
     movlw          b'00111000'         ;set cpu clock speed
     movwf          OSCCON              ;move contents of the working register into OSCCON

                                        ;Configure the ADC/Potentimator
                                        ;already in bank1
     bsf            TRISA, 4            ;Potentimator is connected to RA4....set as input
     movlw          b'00001101'         ;select RA4 as source of ADC and enable the module (carefull, this ↙
    is actually AN3)
     movwf          ADCON0
     movlw          b'00010000'         ;left justified - Fosc/8 speed - vref is Vdd
     movwf          ADCON1
     banksel        ANSELA              ;bank3
     bsf            ANSELA, 4            ;analog for ADC

     ;Configure the LEDs
     banksel        TRISC       ;bank1
     clrf           TRISC               ;make all of PORTC an output
     banksel        LATC                ;bank2
     movlw          b'00001000'         ;start the rotation by setting DS4 ON
     movwf          LATC                ;write contents of the working register to the latch
MainLoop:
     call           A2d                 ;get the ADC result
                                        ;top 8 MSbs are now in the working register (Wreg)
     movwf          Delay2              ;move ADC result into the outer delay loop
     call           CheckIfZero         ;if ADC result is zero, load in a value of '1' or else the delay   ↙
    loop will decrement starting at 255
     call           DelayLoop           ;delay the next LED from turning ON
     call           Rotate              ;rotate the LEDs

     bra            MainLoop            ;do this forever

CheckIfZero:
     movlw          d'0'                ;load wreg with '0'
     xorwf          Delay2, w           ;XOR wreg with the ADC result and save in wreg
     btfss          STATUS, Z           ;if the ADC result is NOT '0', then simply return to MainLoop
     return                             ;return to MainLoop
     movlw          d'1'                ;ADC result IS '0'. Load delay routine with a '1' to avoid         ↙
    decrementing a rollover value of 255
     movwf          Delay2              ;move it into the delay location in shared memory (RAM)
     return                             ;return to MainLoop

A2d:
    ;Start the ADC
    nop                                 ;requried ADC delay of 8uS => (1/(Fosc/4)) = (1/(500KHz/4)) = 8uS
    banksel         ADCON0
    bsf             ADCON0, GO          ;start the ADC
    btfsc           ADCON0, GO          ;this bit will be cleared when the conversion is complete
    goto            $-1                 ;keep checking the above line until GO bit is clear
    movf            ADRESH, w           ;Get the top 8 MSbs (remember that the ADC result is LEFT justified↙
    !)

    return

DelayLoop:
                                        ;Delay amount is determined by the value of the ADC
    decfsz         Delay1,f             ;will always be decrementing 255 here
    goto           DelayLoop            ;The Inner loop takes 3 instructions per loop * 255 loops (required↙
     delay)
    decfsz         Delay2,f             ;The outer loop takes and additional 3 instructions per lap * X    ↙
    loops (X = top 8 MSbs from ADC conversion)
    goto           DelayLoop

    return

Rotate:
    banksel        LATC                 ;change to Bank2
    lsrf           LATC                 ;logical shift right
    btfsc          STATUS,C             ;did the bit rotate into the carry?
    bsf            LATC,3               ;yes, put it into bit 3.

    return

    end                                 ;end code
```

```
/**
 *******************************************************************
 *  Lesson 5 - "Variable Speed Rotate"
 *
 *  This lesson combines all of the previous lessons to produce a variable speed rotating
 *  LED display that is proportional to the ADC value. The ADC value and LED rotate
 *  speed are inversely proportional to each other.
 *
 *  Rotate the POT counterclockwise to see the LEDs shift faster.
 *
 *  PIC: 16F1829
 *  Compiler: XC8 v1.00
 *  IDE: MPLABX v1.10
 *
 *  Board: PICkit 3 Low Pin Count Demo Board
 *  Date: 6.1.2012
 *
 * *******************************************************************
 * See Low Pin Count Demo Board User's Guide for Lesson Information*
 * ******************************************************************
 */

#include <htc.h>                                        //PIC hardware mapping
#define _XTAL_FREQ 500000                               //Used by the XC8 delay_ms(x) macro

//config bits that are part-specific for the PIC16F1829
__CONFIG(FOSC_INTOSC & WDTE_OFF & PWRTE_OFF & MCLRE_OFF & CP_OFF & CPD_OFF & BOREN_ON & CLKOUTEN_OFF &    ↙
    IESO_OFF & FCMEN_OFF);
__CONFIG(WRT_OFF & PLLEN_OFF & STVREN_OFF & LVP_OFF);

    /* -------------------LATC-----------------
     * Bit#:  -7---6---5---4---3---2---1---0---
     * LED:   ---------------|DS4|DS3|DS2|DS1|-
     *-----------------------------------------
     */

unsigned char adc(void);                                //prototype

void main(void) {
    unsigned char delay;


    OSCCON = 0b00111000;                                //500KHz clock speed
    TRISC = 0;                                          //all LED pins are outputs
    LATC = 0;
    LATCbits.LATC3 = 1;                                 //start sequence with DS4 lit


                                                        //setup ADC
    TRISAbits.TRISA4 = 1;                               //Potentiamtor is connected to RA4...set as input
    ANSELAbits.ANSA4 = 1;                               //analog
    ADCON0 = 0b00001101;                                //select RA4 as source of ADC and enable the module ↙
     (AN3)
    ADCON1 = 0b00010000;                                //left justified - FOSC/8 speed - Vref is Vdd

    while (1) {
       delay = adc();                                   //grab the top 8 MSbs
       __delay_ms(5);                                   //delay for AT LEAST 5ms
       while (delay-- != 0)
           __delay_ms(2);                               //decrement the 8 MSbs of the ADC and dealy 2ms for ↙
     each
       LATC >> = 1;                                     //shift to the right by 1 to light up the next LED
       if(STATUSbits.C)                                 //when the last LED is lit, restart the pattern
           LATCbits.LATC3 = 1;
    }

}
```

```
unsigned char adc(void) {
    __delay_us(5);                                      //wait for ADC charging cap to settle
    GO = 1;
    while (GO) continue;                                //wait for conversion to be finished

    return ADRESH;                                      //grab the top 8 MSbs

}
```