```
/*
          __                          __
         /'{>                        /'{>
      ____) (____                 ____) (____
    //'--;   ;--'\\             //'--;   ;--'\\
   ///////\_/\\\\\\\           ///////\_/\\\\\\\
         m m                         m m
        *********** PIC18F46K80 ***********
                  +--------U--------+
       MCLR/RE3 |1              40| RB7  PICKit & LED Bit 7
      POT - RA0 |2              39| RB6  PICKit & LED Bit 6
Light Sens  RA1 |3              38| RB5  LED Bit 5
Analog 2    RA2 |4              37| RB4  LED Bit 4
Analog 3    RA3 |5              36| RB3  LED Bit 3
       VDD Core |6              35| RB2  LED Bit 2
Switch S2   RA5 |7              34| RB1  LED Bit 1
<not used>  RE0 |8              33| RB0  LED Bit 0 & Switch S5
<not used>  RE1 |9              32| VDD
<not used>  RE2 |10             31| GND
            VDD |11             30| RD7 (Rx2)-> Serial Data OUT*  Bluetooth
            GND |12             29| RD6 (Tx2)-> Serial Clock OUT* Bluetooth
  Switch S4 RA7 |13             28| RD5 To LCD
  Switch S3 RA6 |14             27| RD4 To LCD
  To LCD    RC0 |15             26| RC7 (Rx1)-> Serial Data OUT*  USB
  To LCD    RC1 |16             25| RC6 (Tx1)-> Serial Clock OUT* USB
  Buzzer    RC2 |17             24| RC5 <Not Used>
  I2C SCLK  RC3 |18             23| RC4 I2C SDA
  To LCD    RD0 |19             22| RD3 To LCD
  To LCD    RD1 |20             21| RD2 To LCD
                +-----------------+
* Note:  RA4 not available since used by VDD Core!
* for configurations of master synch serial port, see 18.3 of PDF
* YES - Rx IS SERIAL OUT!!!!

*/
#include <stdio.h>
#include <stdlib.h>
#include <p18f46K80.h>
#include <delays.h>
#include <timers.h>
#include <usart.h>


#pragma config FOSC = INTIO2    /* internal osc enable */
#pragma config FCMEN = OFF      /* Failsafe Clock Monitor Disabled */
#pragma config WDTEN = OFF      /* no Dog */
#pragma config XINST = OFF      /* Extended Instruction Set off */
#pragma config SOSCSEL = DIG    /* Configures PORTC0&1 for digital I/O */

//*****************************************************************
//*                                                               *
//*  ECNS-424                                                     *
//*  19 February 2021                                             *
//*  In Depth Lab-1                                               *
//*                                                               *
//*  The purpose of this program is to use the BulldogPic++       *
//*  A/D converter to read the light sensors present              *
//*  Voltage and Lux and display it on on the LCD screen          *
//*                                                               *
//*  Created by: Brandon Empie                                    *
//*                                                               *
//*****************************************************************

extern void LCDInit(void);      /* LCD initialize    */
extern void lcd_clr(void);      /* clear the lcd     */
extern void LCDLine_1(void);    /* get LCD to line 1 */
extern void LCDLine_2(void);    /* get LCD to line 2 */
extern void end_line(void);     /* end of line       */
extern void d_write(void);          /* write data to LCD */
extern void i_write(void);      /* position lcd curs.*/
extern unsigned int temp_wr;    /* pass ascii char   */

const char line1[14]={"           Lux"};  //Line1 constant for splash screen
const char line2[16]={"           Volts"};//Line2 constant for splash screen

unsigned int j;      //init. variable j, loop variable for splash screen
unsigned long X = 0; //init. variable X, A/D result variable
unsigned long Y = 0; //init. variable Y, Voltage result variable
unsigned long Z = 0; //init. variable Z, Lux result variable
unsigned long V = 0; //init. variable V (Voltage), used for parsing values
unsigned int B = 0;  //init. variable B (thousands)
```

```
unsigned int H = 0;  //init. variable H (Hundreds)
unsigned int T = 0;  //init. variable T (Tens)
unsigned int O = 0;  //init. variable O (Ones)

 void ADConverter();    //function prototype for subroutine "ADConverter"
 void ShowoffVolts();   //function prototype for subroutine "ShowoffVolts"
 void ShowoffLux();     //function prototype for subroutine "ShowoffLux"


//////////////////////////////////////////////////////////////
//                          MAIN                            //
//////////////////////////////////////////////////////////////
void main (void)
{
    OSCCON = 0x60;       //set clock to default 8mHz
    TRISB = 0;            // PORT B all outputs for LEDs
    LATB = 0;             // All outputs off initially
    TRISA = 0xFF;         //PORT A all inputs
    ANCON0 = 0;           // AN0, AN1, AN2, AN3 configured as digital
    ANCON1 = 0;           // All other channels are digital
    PORTAbits.RA1 = 1;    //Light sensor channel set to analog

    ADCON2bits.ADFM = 0;  //Left Justify A/D
    ADCON2bits.ADCS = 4;  //FOSC/4
    ADCON2bits.ACQT = 2;  //A/D acquisition time select 4 TAD

    ADCON0bits.CHS = 1;   //A/D now set to channel 01 (AN1) for light sensor
    ADCON0bits.ADON = 1;  //Turn on A/D

    LCDInit();          //Initialize LCD
    lcd_clr();      //clear the display

        LCDLine_1();            //Setup first Line
        for (j=0;j<14;j++)
        {
            temp_wr = line1[j];// write one char. at a time
            d_write();             //send to LCD
        }
        end_line();             //end of line call

//line one over, now line 2

        LCDLine_2();            // Setup 2nd line
        for (j=0;j<16;j++)
        {
            temp_wr = line2[j];// write one char. at a time
            d_write();             // Send to LCD
        }
        end_line();             // end of line call


    while(1)
    {
        ADConverter();        //call 'ADConverter' function to read A/D


        Y = ((X*500)/255);    //Y(Vin) = ((A/D result)* 500)/255(8-bit) (5V, 500 causes Y value to be ex. 320 instead of 3.2 to
bypass floating point)
        V = Y;                //V = Y, Voltage value stored into V, Y will now be used to calculate lux

        H = (V/100);                //parsing voltage value by Hundreds place
        T = ((V-(H*100))/10);       //parsing voltage value by Tens place
        O = (((V-(H*100))-(T*10))); //parsing voltage value by Ones place


        ShowoffVolts();             //call 'ShowoffVolts'  function to set LCD


        Z = ((Y*1000)/224);   // Z now = Lux (lux value is based on 2.240mV/Lux)

        B = (Z/1000);                   //Parsing Lux value by Thousands place
        H = ((Z-(B*1000))/100);         //Parsing Lux value by Hundreds place
        T = ((Z-(B*1000)-(H*100))/10);  //Parsing Lux value by Tens place
        O = (Z-(B*1000)-(H*100)-(T*10));//Parsing Lux value by Ones place


        ShowoffLux();            //call 'ShowoffLux'  function to set LCD

        Delay1KTCYx(250);   //250mS delay
```

```
    }
}

//****************************************
//* Subroutine: ADConverter              *
//*                                      *
//* Inputs: none                         *
//* Outputs: none                        *
//*                                      *
//* The purpose of this subroutine is to  *
//* read the A/D placing the result in    *
//* variable X                           *
//*                                      *
//* Created by: Brandon Empie            *
//****************************************
void ADConverter(void)
{
   ADCON0bits.GO = 1;          //Start A/D
   while(ADCON0bits.GO);       //wait till A/D is done

   X = ADRESH;                 //X = A/D result 8-bit

   return;                     //return to main
}
//****************************************
//* Subroutine: ShowoffVolts             *
//*                                      *
//* Inputs: none                         *
//* Outputs: none                        *
//*                                      *
//* The purpose of this subroutine is to  *
//* send the present voltage value to     *
//* the LCD display                      *
//*                                      *
//* Created by: Brandon Empie            *
//****************************************
void ShowoffVolts(void)
{
   temp_wr = 0xc6; //setup cursor 8 positions from left on 2nd line of LCD
   i_write();      //setting cursor position


   H = H + 48;     //adding 48 dec or 0x30 to Hundreds place to set correct ASCII value for 0-9
   temp_wr = H;    //write hundreds place
   d_write();      //Send to LCD


   temp_wr = 0x2E; //0x2E = decimal point in ASCII table, writing it between hundreds and tens place
   d_write();      //Send to LCD


   T = T + 48;     //adding 48 dec or 0x30 to Tens place to set correct ASCII value for 0-9
   temp_wr = T;    //write Tens place
   d_write();      //Send to LCD


   O = O + 48;     //adding 48 dec or 0x30 to Ones place to set correct ASCII value for 0-9
   temp_wr = O;    //write Ones place
   d_write();      //Send to LCD

   return;         //return to main
}

//****************************************
//* Subroutine: ShowoffLux               *
//*                                      *
//* Inputs: none                         *
//* Outputs: none                        *
//*                                      *
//* The purpose of this subroutine is to  *
//* send the present Lux value to the     *
//* the LCD display                      *
//*                                      *
//* Created by: Brandon Empie            *
//****************************************
void ShowoffLux(void)
{
   temp_wr = 0x86; //setup cursor 8 positions from left on 1st line of LCD
   i_write();      //setting cursor position
```

```
   if(B==0)           //If thousands place is equal to zero
   {
       temp_wr = 0x20;//write nothing into LCD segment
       d_write();     //Send to LCD
   }
   else               //if B != 0
   {
   B = B + 48;        //adding 48 dec or 0x30 to Thousands place to set correct ASCII value for 0-9
   temp_wr = B;       //write Thousands place
   d_write();         //Send to LCD
   }
   if(B+H==0)         //if thousands place and hundreds place are both zero
   {
      temp_wr = 0x20; //write nothing to hundreds place
      d_write();      //Send to LCD
   }
   else               //if B+H != 0
   {
       H = H + 48;    //adding 48 dec or 0x30 to Hundreds place to set correct ASCII value for 0-9
       temp_wr = H;   //write hundreds place
       d_write();     //Send to LCD
   }
   if(B+H+T == 0)     //if B,H,T are all zero
   {
      temp_wr = 0x20; //write nothing to Tens place
      d_write();      //Send to LCD
   }
   else               //if B+H+T != 0
   {
      T = T + 48;     //adding 48 dec or 0x30 to Tens place to set correct ASCII value for 0-9
      temp_wr = T;    //write Tens place
      d_write();      //Send to LCD
   }

   O = O + 48;        //adding 48 dec or 0x30 to Ones place to set correct ASCII value for 0-9
   temp_wr = O;       //write Ones place
   d_write();         //Send to LCD

   return;            //return to main
}
```