LED Blinking

```
#include <xc.h>                                  //Include Controller specific .h
//Configuration bit settings
#pragma config OSC = HS                                  //Oscillator Selection
#pragma config WDT = OFF                 //Disable Watchdog timer
#pragma config LVP = OFF                 //Disable Low Voltage Programming
#pragma config PBADEN = OFF               //Disable PORTB Analog inputs

//Function Prototypes
void msdelay (unsigned int time);                        //Function for delay

//Start of Program Code
void main()                              //Main Program
{

      INTCON2bits.RBPU=0;                        //To Activate the internal pull on PORTB
        ADCON1 = 0x0F;                   //To disable the all analog inputs

        TRISA = 0x00;                    //To configure PORTA    as output
        TRISB = 0x00;                    //To configure PORTB    as output
        TRISC = 0x00;                    //To configure PORTC    as output
        TRISD = 0x00;                    //To configure PORTD    as output
```

```
        TRISE = 0x00;                   //To configure PORTE     as output
        while (1)                       //While loop for repeated operation
        {
                PORTA = 0xAA;
                PORTB = 0xAA;
                PORTC = 0xAA;
                PORTD = 0xAA;
                PORTE = 0xAA;

                msdelay(200);

                PORTA = 0x55;
                PORTB = 0x55;
                PORTC = 0x55;
                PORTD = 0x55;
                PORTE = 0x55;

                msdelay(200);

        }
}                                       //End of the Program

//Function Definitions
void msdelay (unsigned int time)        //Function for delay
```

```
{
unsigned int i, j;
  for (i = 0; i < time; i++)
        for (j = 0; j < 710; j++);                          //Calibrated for a 1 ms delay in
MPLAB
}
```

16 x2 LCD Interfacing

```
#include <xc.h>
//Configuration bit setting//
#pragma config OSC = HS                 //Oscillator Selection
#pragma config WDT = OFF                //Disable Watchdog timer
#pragma config LVP = OFF                //Disable Low Voltage Programming
#pragma config PBADEN = OFF              //Disable PORTB Analog inputs


//Declarations
#define LCD_DATA      PORTD                     //LCD data port to PORTD
#define ctrl          PORTE                 //LCD control port to PORTE
#define rs             PORTEbits.RE0            //register select signal to RE0
#define rw             PORTEbits.RE1            //read/write signal to RE1
```

```
#define en              PORTEbits.RE2           //enable signal to RE2

//Function Prototypes
void init_LCD(void);                            //Function to initialize the LCD
void LCD_command(unsigned char cmd);            //Function to pass command to the LCD
void LCD_data(unsigned char data);              //Function to write character to the LCD
void LCD_write_string(char *str);               //Function to write string to the LCD
void msdelay (unsigned int time);               //Function to generate delay

//Start of Main Program
void main(void)
{
        char var1[] = "SKNCOE";                 //Declare message to be displayed
        char var2[] = "KOHLI";


        ADCON1 = 0x0F;                          //Configuring the PORTE pins as digital I/O
        TRISD = 0x00;                           //Configuring PORTD as output
        TRISE = 0x00;                       //Configuring PORTE as output


        init_LCD();                                     // call function to initialise of LCD
        msdelay(50);                            // delay of 50 mili seconds
```

```
        LCD_write_string(var1);                 //Display message on first line
        msdelay(15);

        LCD_command(0xC0);                      // initiate cursor to second line
        LCD_write_string(var2);                 //Display message on second line

        while (1);                              //Loop here
}                                               //End of Main

//Function Definitions
void msdelay (unsigned int time)            //Function to generate delay
{
unsigned int i, j;
   for (i = 0; i < time; i++)
         for (j = 0; j < 275; j++);         //Calibrated for a 1 ms delay in MPLAB
}


void init_LCD(void)                         // Function to initialise the LCD
{
      LCD_command(0x38);                        // initialization of 16X2 LCD in 8bit mode
      msdelay(15);
      LCD_command(0x01);                        // clear LCD
      msdelay(15);
```

```
    LCD_command(0x0C);                  // cursor off
    msdelay(15);
    LCD_command(0x80);                  // go to first line and 0th position
    msdelay(15);
}

void LCD_command(unsigned char cmd)     //Function to pass command to the LCD
{
    LCD_DATA = cmd;                     //Send data on LCD data bus
    rs = 0;                             //RS = 0 since command to LCD
    rw = 0;                             //RW = 0 since writing to LCD
    en = 1;                             //Generate High to low pulse on EN
    msdelay(15);
    en = 0;
}

void LCD_data(unsigned char data)       //Function to write data to the LCD
{
    LCD_DATA = data;                    //Send data on LCD data bus
    rs = 1;                             //RS = 1 since data to LCD
    rw = 0;                             //RW = 0 since writing to LCD
    en = 1;                             //Generate High to low pulse on EN
      msdelay(15);
    en = 0;
```

```
}
//Function to write string to LCD
void LCD_write_string( char *str)
{
	int i = 0;
	while (str[i] != '\0')
	{
		LCD_data(str[i]);			// sending data on LCD byte by byte
		msdelay(15);
		i++;
	}
}
```

DC Motor Interfaceing

```
#include <xc.h>

#pragma config OSC = HS				//Oscillator Selection
#pragma config WDT = OFF			//Disable Watchdog timer
#pragma config LVP = OFF			//Disable Low Voltage Programming
#pragma config PBADEN = OFF			//Disable PORTB Analog inputs

void myMsDelay (unsigned int time)		// Definition of delay subroutine
```

```
{
        unsigned int i, j;
        for (i = 0; i < time; i++)          // Loop for i time
                for (j = 0; j < 375; j++);// Calibrated for a 1 ms delay in MPLAB
}

void main()
{
        TRISCbits.TRISC0 = 0 ;                  // Set PORTC, RC0 as output (DCM IN1)
        TRISCbits.TRISC1 = 0 ;                  // Set PORTC, RC1 as output (DCM IN2)
        TRISCbits.TRISC2 = 0 ;                  // Set PORTC, RC2 as output (DCM EN1)
        TRISCbits.TRISC3 = 0 ;                  // Set PORTC, RC3 as output (DCM EN2)
        TRISCbits.TRISC4 = 0 ;                  // Set PORTC, RC4 as output (DCM IN3)
        TRISCbits.TRISC5 = 0 ;                  // Set PORTC, RC5 as output (DCM IN4)

      PORTCbits.RC0 = 0;
      PORTCbits.RC1 = 0;
      PORTCbits.RC2 = 0;
      PORTCbits.RC3 = 0;
      PORTCbits.RC4 = 0;
      PORTCbits.RC5 = 0;

   while(1)      // Endless Loop
        {
```

```
PORTCbits.RC0 = 1;
PORTCbits.RC1 = 0;
PORTCbits.RC2 = 1;
PORTCbits.RC3 = 1;
PORTCbits.RC4 = 1;
PORTCbits.RC5 = 0;

    myMsDelay(2000);

PORTCbits.RC0 = 0;
PORTCbits.RC1 = 0;
PORTCbits.RC2 = 0;
PORTCbits.RC3 = 0;
PORTCbits.RC4 = 0;
PORTCbits.RC5 = 0;

myMsDelay(2000);

PORTCbits.RC0 = 0;
PORTCbits.RC1 = 1;
PORTCbits.RC2 = 1;
PORTCbits.RC3 = 1;
PORTCbits.RC4 = 0;
PORTCbits.RC5 = 1;
```

```
            myMsDelay(2000);


            PORTCbits.RC0 = 0;
            PORTCbits.RC1 = 0;
            PORTCbits.RC2 = 0;
            PORTCbits.RC3 = 0;
            PORTCbits.RC4 = 0;
            PORTCbits.RC5 = 0;


              myMsDelay(2000);


        }

}
```

Interfacing LEDs, Switches, Buzzer and Relay

```
#include <xc.h>                         //Include Controller specific .h


//Configuration bit settings
#pragma config OSC = HS                 //Oscillator Selection
#pragma config WDT = OFF                //Disable Watchdog timer
```

```
#pragma config LVP = OFF                    //Disable Low Voltage Programming
#pragma config PBADEN = OFF                  //Disable PORTB Analog inputs

//Declarations
#define lrbit    PORTBbits.RB0              //SW1 interfaced to RB0
#define rlbit    PORTBbits.RB1              //SW2 interfaced to RB1
#define relay   PORTBbits.RB2               //Relay interfaced to RB2
#define buzzer    PORTBbits.RB3             //Buzzer interfaced to RB3


//Function Prototypes
void msdelay (unsigned int time);           //Function for delay

//Start of Program Code
void main()                                 //Main Program
{
        unsigned char val=0;                //Variable to latch the switch condition
      INTCON2bits.RBPU=0;                       //To Activate the internal pull on PORTB
        ADCON1 = 0x0F;                      //To disable the all analog inputs

        TRISBbits.TRISB0=1;                       //To configure RB4 as input for sensing SW0
        TRISBbits.TRISB1=1;                 //To configure RB5 as input for sensing SW1

        TRISBbits.TRISB2=0;                       //To configure RC1 (relay) as output
```

```
        TRISBbits.TRISB3=0;                 //To configure RC2 (buzzer) as output
        TRISD = 0x00;                       //To configure PORTD (LED) as output

        PORTD = 0x00;               //Initial Value for LED
        buzzer = 0;                         //Initial Value for Buzzer
        relay = 0;                          //Initial Value for Relay

while (1)                                   //While loop for repeated operation
        {
        if (!(lrbit))                       //To check whether SW0 is pressed
                val = 1;                    // Latch the status of switch SW0
        if (!(rlbit))                       //To check whether SW1 is pressed
                val = 2;                    // Latch the status of switch SW1

      if (val == 1)
            {
                buzzer = 1;
                relay = 1;
                PORTD = PORTD >>1;          //Shift right by 1 bit
                if (PORTD == 0x00)
                        PORTD = 0x80;       // Make the MSB bit equal to 1
                msdelay(250);
            }
        if (val == 2)
```

```
            {
                buzzer = 0;
                relay = 0;
                PORTD = PORTD<<1;                    //Shift left by 1 bit
                if (PORTD == 0x00)
                        PORTD = 0x01;                // Make the LSB bit eqaul to 1
                msdelay(250);
            }
        }
}                                                    //End of the Program

//Function Definitions
void msdelay (unsigned int time)                     //Function for delay
{
unsigned int i, j;
  for (i = 0; i < time; i++)
          for (j = 0; j < 375; j++);                 //Calibrated for a 1 ms delay in MPLAB
}
```

Temperature Sensor Interfacing

* LM35 Sensor is interfaced to AN1 (RA1)
* Output in Degree Celsius shown on LCD

```
------------------------------------------------------- -*/
/* Interface Details
  * LM35 Sensor                  - RA1 - AN1
  * LCD Data (D0 to D7)    - PORTD (RD0 to RD7)
  * LCD RS                        - RE0
  * LCD RW                        - RE1
  * LCD EN                        - RE2
-------------------------------------------------------*/

#include <xc.h>
//#include <pic18f4520.h>
//Configuration bit setting//
#pragma config OSC = HS                  //Oscillator Selection
#pragma config WDT = OFF                 //Disable Watchdog timer
#pragma config LVP = OFF                 //Disable Low Voltage Programming
#pragma config PBADEN = OFF              //Disable PORTB Analog inputs

//Declarations for LCD Connection
#define LCD_DATA       PORTD                                  //LCD data port
#define en             PORTEbits.RE2                          // enable signal
#define rw             PORTEbits.RE1                          // read/write signal
#define rs             PORTEbits.RE0                          // register select signal

//Function Prototypes
```

```
void ADC_Init(void);                          //Function to initialize the ADC
unsigned int Get_ADC_Result(void);            //Function to Get ADC result after
conversion
void Start_Conversion(void);                  //Function to Start of Conversion
void msdelay (unsigned int time);             //Function to generate delay
void init_LCD(void);                          //Function to initialise the LCD
void LCD_command(unsigned char cmd);          //Function to pass command to the LCD
void LCD_data(unsigned char data);            //Function to write character to the LCD
void LCD_write_string( char *str);            //Function to write string to the LCD


//Start of main program
void main()
{
  char msg1[] = "LM35 Interface";
  char msg2[] = "Temp.:";
  char msg3[] = {0xDF, 0x43, 0x00};
  unsigned char temp=0;
  unsigned char i=0, Thousands,Hundreds,Tens,Ones;
  unsigned int adc_val;
  unsigned char val, pot0[6];
  ADCON1 = 0x0F;                              //Configuring the PORTE pins as digital I/O
  TRISD = 0x00;                              //Configuring PORTD as output
  TRISE = 0x00;                                    //Configuring PORTE as output
```

```
ADC_Init();                          // Init ADC peripheral
init_LCD();                          // Init LCD Module
LCD_write_string(msg1);              // Display Welcome Message
LCD_command(0xC0);                   // Goto second line, 0th place of LCD
LCD_write_string(msg2);              // Display Message "Temp:"

while(1)
{
  Start_Conversion();                //Trigger conversion
  adc_val= Get_ADC_Result();         //Get the ADC output by polling GO bit
  adc_val = adc_val/2;               //Divide the value by 2 match with 10mv stepsize
  LCD_command (0xC7);          //Goto 8th place on second line of LCD

  val = (unsigned char) adc_val;
  i = (val/100);                           //Get the Hundreds place
  Hundreds = i + 0x30;                     // Convert it to ASCII
  LCD_data (Hundreds);               //Display Hundreds place

  i = (val%100)/10; //Get the Tens place
  Tens = i + 0x30;                   // Convert it to ASCII
  LCD_data (Tens);                   //Display Tens place
```

```
    i = adc_val%10 ;                    //Get the Ones place
    Ones = i + 30;                      // Convert it to ASCII
    LCD_data (i + 0x30);                        //Display Ones place



    LCD_write_string(msg3);


    msdelay(300);                       //Delay between conversions.
   }

}



//Function Definitions
void ADC_Init()
{
  ADCON0=0b00000100;            //A/D Module is OFF and Channel 1 is selected
  ADCON1=0b00001110;            // Reference as VDD & VSS, AN0 set as analog pins
  ADCON2=0b10001110;            // Result is right Justified
                                //Acquisition Time 2TAD
                                //ADC Clk FOSC/64
  ADCON0bits.ADON=1;            //Turn ON ADC module
}
```

```
void Start_Conversion()
{
  ADCON0bits.GO=1;
}

//If you do not wish to use adc conversion interrupt you can use this
//to do conversion manually. It assumes conversion format is right adjusted
unsigned int Get_ADC_Result()
{
  unsigned int ADC_Result=0;
  while(ADCON0bits.GO);
  ADC_Result=ADRESL;
  ADC_Result|=((unsigned int)ADRESH) << 8;
  return ADC_Result;
}

void msdelay (unsigned int time)                    //Function to generate delay
{
unsigned int i, j;
    for (i = 0; i < time; i++)
          for (j = 0; j < 275; j++);            //Calibrated for a 1 ms delay in MPLAB
}
```

```
void init_LCD(void)                          // Function to initialise the LCD
{
    LCD_command(0x38);                       // initialization of 16X2 LCD in 8bit mode
    msdelay(15);
    LCD_command(0x01);                       // clear LCD
    msdelay(15);
    LCD_command(0x0C);                       // cursor off
    msdelay(15);
    LCD_command(0x80);                       // go to first line and 0th position
    msdelay(15);
}

void LCD_command(unsigned char cmd)      //Function to pass command to the LCD
{
    LCD_DATA = cmd;                          //Send data on LCD data bus
    rs = 0;                              //RS = 0 since command to LCD
    rw = 0;                              //RW = 0 since writing to LCD
    en = 1;                              //Generate High to low pulse on EN
    msdelay(15);
    en = 0;
}

void LCD_data(unsigned char data)        //Function to write data to the LCD
{
```

```
	LCD_DATA = data;                    //Send data on LCD data bus
	rs = 1;                             //RS = 1 since data to LCD
	rw = 0;                             //RW = 0 since writing to LCD
	en = 1;                             //Generate High to low pulse on EN
	   msdelay(15);
	en = 0;
}
//Function to write string to LCD
void LCD_write_string(char *str)
{
	int i = 0;
	while (str[i] != 0)
	{
		LCD_data(str[i]);           // sending data on LCD byte by byte
		msdelay(15);
		i++;
	}
}
```