Diseño de una Plataforma de Medida de Luz y Temperatura (Enero 2011)

Iván López Espejo

El presente documento recoge el desarrollo de una plataforma de instrumentación para la medida de temperatura e iluminación ambiental, la cual se fundamenta en el uso del microcontrolador PIC16F876 de Microchip. Las medidas se muestran a través de un LCD.

I. Desarrollo

En este punto vamos a tratar cómo se llegó hasta la solución obtenida en el laboratorio.

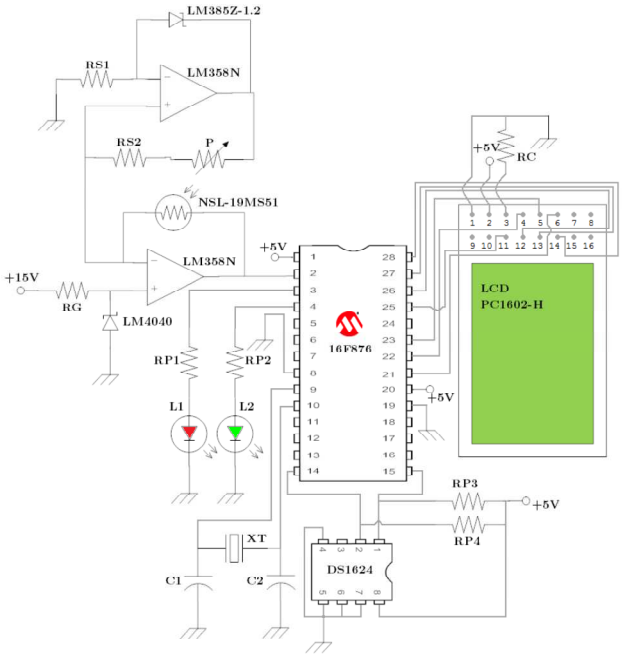

Fig. 1. Montaje experimental.

Para comenzar, la figura anterior muestra un esquemático con el montaje experimental que finalmente fue implementado.

A. Sensor de iluminación

La señal procedente del sensor de iluminación es introducida por la patilla *A0* del microcontrolador. Para poder realizar una medición adecuada, debemos ajustar la señal procedente del sensor LDR (Light Dependent Resistance) en el intervalo de 0V a 5V, aprovechando así el rango del convertidor A/D del microcontrolador.

Para lograr este ajuste, partimos del esquemático de transducción de luminosidad de la figura 2. La fuente de corriente constante se encarga de la polarización del elemento sensor de luminosidad. Sabiendo que para máxima iluminación la LDR posee una resistencia de 360Ω, así como una de 150kΩ para mínima iluminación, diseñamos las resistencias de la fuente de corriente. Como ya hemos dicho, deseamos aprovechar todo el rango del convertidor del PIC, así que para la resistencia máxima de la LDR (mínima iluminación), imponemos una tensión de salida de 0V.

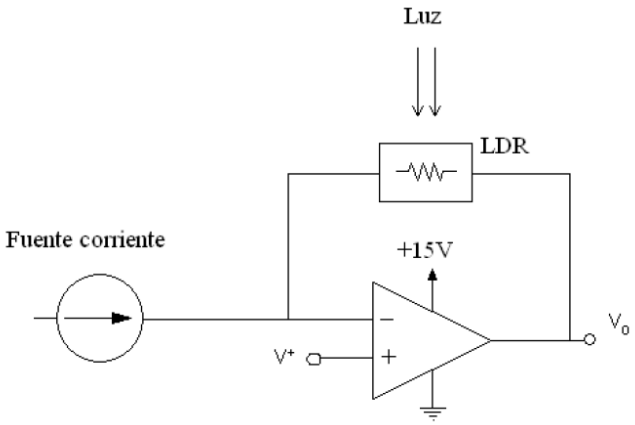

Fig. 2. Transductor de luz.

Fijamos otra restricción al imponer una tensión de salida máxima de 5V con máxima iluminación (en este caso aproximamos la resistencia de la LDR por 0Ω).

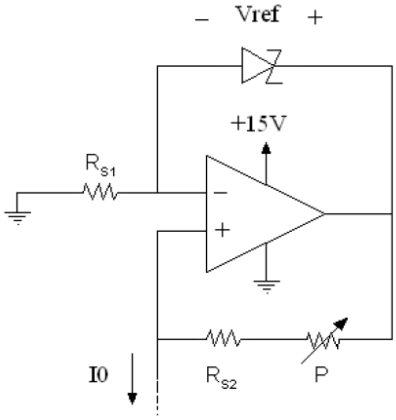

Fig. 3. Esquemático de la fuente de corriente.

Asumiendo el modelo ideal del amplificador operacional, y dado que tenemos 5V procedentes de la referencia de tensión en la entrada no inversora del mismo, tendremos otros 5V en la entrada inversora del LM358N de nuestro transductor de luz. En consecuencia, la fuente habrá de proporcionar una corriente de valor

$$I_{Fuente} = \frac{V^- - V_o}{R_{LDR}} = \frac{5V - 0V}{150k\Omega} = 0.033mA.$$

A partir de la referencia de tensión que impone el LM385Z-1.2, podemos calcular el valor de la resistencia R_{S2} en función de la corriente deseada, de la forma,

$$R_{S2} = \frac{1.23V}{I_{Fuente}} = \frac{1.23V}{0.033mA} = 36.9k\Omega.$$

El potenciómetro (P) se incluye para un ajuste fino, ya que no existe una resistencia comercial del valor calculado para R_{S2}. En la implementación final, únicamente se situó el potenciómetro, ajustando previamente su valor al obtenido del diseño.

De otro lado, la resistencia R_{S1} se estableció a 100kΩ tras consultar en la hoja de características de la referencia de tensión usada para la fuente de corriente.

Es importante notar que es imprescindible aportar dichos 5V en la entrada no inversora del operacional del transductor, pues, si queremos una relación lineal entre la resistencia dada por la LDR y la tensión a la entrada del convertidor del microcontrolador, necesitaremos dichos 5V para que el sistema funcione con un valor de corriente constante proporcionado por la fuente. En caso de seguir queriendo obtener a la entrada del convertidor del PIC un valor en el intervalo de 0V a 5V de dependencia lineal con la resistencia de la LDR y utilizando otra tensión de referencia en la entrada no inversora del operacional, precisaríamos de una fuente de corriente variable y dependiente del valor de resistencia de la LDR en cada instante, lo que es una complicación innecesaria.

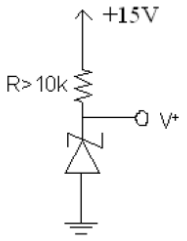

Fig. 4. Referencia de tensión de 5V.

La referencia de tensión de 5V se obtiene a partir del uso del dispositivo LM4040. Este dispositivo encapsula un diodo zéner con una tensión de ruptura de 5V. En la configuración presentada, tenemos un sencillo regulador de tensión, por lo que utilizando una resistencia de valor superior a 10kΩ (se hizo uso de una de valor nominal 22kΩ) aseguramos el estar trabajando en la zona de ruptura al utilizar una tensión de entrada al regulador de 15V, así como al tener una corriente circulando por el mismo comprendida entre unos límites adecuados impuestos por la característica I-V del dispositivo.

Notar que no debemos conectar la alimentación de los distintos elementos digitales a la salida de dicha referencia, pues al funcionar el regulador mediante el consumo de corriente por parte del diodo zéner, puede que no se proporcione una corriente suficiente para la alimentación del resto de dispositivos.

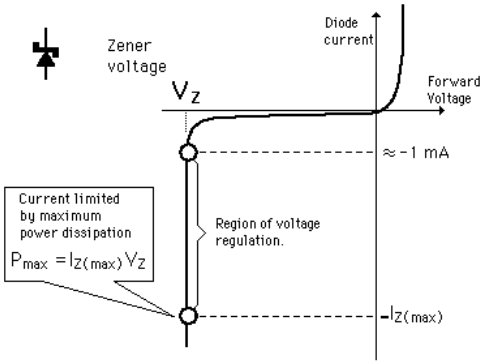

Fig. 5. Regulación de voltaje del diodo zéner.

Finalmente, en el laboratorio se comprobó que, para baja iluminación, se obtenía una tensión a la entrada del ADC de poco más de 1V mientras que, para alta iluminación, se lograban los 5V.

B. Sensor de temperatura

Como sensor de temperatura se hizo uso del dispositivo DS1624. Los 3 bits de dirección de dispositivo con comunicación I2C se fijaron a cero mediante la conexión de las patillas *A0*, *A1* y *A2* a masa.

Como resistencias de pull-up se hizo uso de dos resistencias de 4.7kΩ nominales (una para la patilla SDA y otra para SCL).

Los pines SDA y SCL se conectaron al microcontrolador de forma apropiada (entradas *C4* y *C3* del PIC, respectivamente).

C. LCD, LEDs y reloj

Las conexiones de dichos elementos ya se encuentran indicadas en el esquemático de la primera página.

Para el contraste de la pantalla LCD, se hizo uso de una resistencia de 2.2kΩ nominales.

El reloj se configuró mediante la utilización de un oscilador a 4MHz. Los dos condensadores adheridos (C1 y C2) tienen ambos una capacidad de 27pF.

Las resistencias de pull-up para los LEDs (uno rojo para indicar un exceso de temperatura y otro verde para cuando existe baja iluminación) se seleccionaron de tal forma que la corriente se estableciese en torno a 15mA a fin de prolongar la vida útil de los LEDs sabiendo que el potencial barrera de los mismos (dependiente de la frecuencia de emisión) se mueve entre 2V y 3V aproximadamente, así como que la salida en alto por una de las patillas del PIC se corresponde con una tensión de 5V. Finalmente, las resistencias usadas fueron de valor 220Ω nominales.

D. Microcontrolador

Para terminar, vamos a comentar brevemente la configuración del microcontrolador. El mismo se encuentra conectado tal y como se especifica en el esquemático del comienzo.

El programa volcado al PIC para la consecución de las

funciones de la plataforma, fue el siguiente:

```
#include "16F876.H"
#fuses XT, NOWDT, BROWNOUT, NOPROTECT, PUT, NOCPD
#use delay(clock = 4000000, restart_wdt)
#use i2c (master, sda=PIN_C4, scl=PIN_C3, noforce_sw)

#define DAL_SCL PIN_C3
#define DAL_SDA PIN_C4
#include "ds1624.c"  // Funciones básicas del termómetro.
#define use_portb_lcd TRUE
#include "LCD.c"     // Fichero para la gestión del LCD.

// Constantes para el control del LCD.
#define LCD_COMANDO 0
#define LCD_CLEAR  0x01

// Optimización de las funciones de E/S.
#use fast_io(a)
#use fast_io(b)
#use standard_io(c)

int controla = 0; // Control temporal de las interrupciones.
float temp = 20; // Variable para el almacenaje de la temperatura medida.
int8 analog = 0; // Para la lectura del convertidor A/D.

// Rutina del timer1.
#int_TIMER1
void rutina_timer1(){

  SET_TIMER1(3035);

  // RUTINA DE LUMINOSIDAD Y MUESTRA DE RESULTADOS.
  if ((controla == 1) || (controla == 3) || (controla == 5)){
    // Lectura del conversor A/D.
    analog = read_adc();
    lcd_send_byte(LCD_COMANDO,LCD_CLEAR);
    lcd_gotoxy(1,1);
    // Definimos varios rangos de luminosidad.
    if (analog <= 51){
      printf(lcd_putc, "Lumin: MUY BAJA");
    } else {
      if ((51 < analog) && (analog <= 102)){
        printf(lcd_putc, "Lumin: BAJA");
      } else {
        if ((102 < analog) && (analog <= 153)){
          printf(lcd_putc, "Lumin: MEDIA");
        } else {
          if ((153 < analog) && (analog <= 204)){
            printf(lcd_putc, "Lumin: ALTA");
          } else {
            printf(lcd_putc, "Lumin: MUY ALTA");
          }
        }
      }
    }
    // Mostramos también la temperatura.
    lcd_gotoxy(1,2);
    printf(lcd_putc, "Temp: %d",(int)(temp));
    //(temp-32)/1.8) -> Fahrenheit a Celsius.
    // Comprobamos si fuera preciso encender el LED verde.
    if (analog <= 102){
      output_high(PIN_A2);
    } else {
      output_low(PIN_A2);
    }
    // Comprobamos si fuera preciso encender el LED rojo.
    if ((int)(temp) > 30){
      output_high(PIN_A1);
    } else {
      output_low(PIN_A1);
    }
  }

  // RUTINA DE LECTURA DE TEMPERATURA.
  if (controla == 5){
    temp = read_temp();
    init_temp();
    controla = 0;
  } else {
    controla++;
  }

}

void main(){

  // Inicialización del LCD.
  lcd_init();
  lcd_send_byte(LCD_COMANDO,LCD_CLEAR);
  lcd_gotoxy(1,1);
  printf(lcd_putc, "Iniciando...");
  delay_ms(2000);

  // Configuración de puertos.
  set_tris_a(0x01); // RA0 entrada y demás salidas.
  set_tris_b(0x00); // Puerto b: salidas.

  // Configuración de la conversión A/D.
  setup_adc_ports(AN0);
  setup_adc(adc_clock_div_32);
  // Selección del canal 0 (PIN RA0).
  set_adc_channel(0);

  // Inicialización del sensor de temperatura.
  init_ds1624();
  init_temp(); // Inicializamos la primera medida.

  // Configuración de las interrupciones.
  SETUP_TIMER_1(T1_INTERNAL|T1_DIV_BY_8);
  // T = 4*(1/fosc)*8*(65535-x). Dado que el máximo prescaler es de 8,
  // para una frecuencia de 4MHz, el máximo tiempo de desbordamiento
  // es de aproximadamente medio segundo. El TIMER1 es de 16bits.
  SET_TIMER1(3035); // Desborda cada medio segundo.
  // Habilitación de interrupciones.
  enable_interrupts(INT_TIMER1);
  enable_interrupts(GLOBAL);

  while(1){
  }

}
```

En dicho código se cargan las librerías necesarias, se definen algunas constantes, se establece la frecuencia del reloj (así como que dicha señal de reloj es externa) y se definen las variables que serán usadas.

A continuación se escribe la rutina del TIMER1. Una variable controla la secuencia de interrupciones hasta un total de 6 y se reinicia. Esto se hace así debido a dos motivos. El primero es que dicha rutina debe contemplar tanto la medida de la temperatura como la medida de la luminosidad, cuyos períodos son distintos (3 y 1 segundos, respectivamente). El segundo motivo se debe a que, con el reloj de 4MHz y las características del TIMER1 (contador de 16 bits, incremento del mismo cada ciclo de instrucción (4 ciclos de reloj) y máximo prescaler de 8), el máximo período de desbordamiento del contador asociado es de aproximadamente

0.5s. El período de desbordamiento se fijó mediante el establecimiento del contador al valor inicial 3035 según la siguiente expresión:

$$T = 4 \times \frac{1}{f_{osc}} \times Prescaler \times (2^{16} - x) \Rightarrow$$

$$\Rightarrow x = 2^{16} - \frac{T \times f_{osc}}{4 \times Prescaler} =$$

$$= 2^{16} - \frac{0.5s \times 4MHz}{4 \times 8} = 3036.$$

De esta forma, contando períodos de medio segundo, cada dos cuentas actualizamos el LCD, comprobamos si es preciso encender alguno de los LEDs y tomamos la lectura procedente del convertidor analógico/digital. El resultado de dicha lectura se almacena en una variable entera de 8 bits, por lo que la luminosidad oscilará entre 0 (oscuridad total) y 255 (máxima luminosidad). Dividiendo dicho rango en 5 partes iguales, establecemos la lectura en el LCD según sea luminosidad *muy baja*, *baja*, *media*, *alta* o *muy alta*. En caso de que la luz medida sea baja o muy baja, el LED verde es encendido. En el caso de que la temperatura fuera superior a 30ºC, será encendido el LED rojo.

Por otra parte, cada 6 cuentas de medio segundo, se procede a la lectura de la temperatura mediante la llamada a las funciones de la librería. Notar que la conversión de grados Fahrenheit a Celsius ya se encuentra incluida dentro del método de lectura de temperatura en la librería.