
Termómetro Digital usando un Termistor NTC y un PIC16F873

Autor: Ing ° Carlos A. Narváez V. Universidad de Oriente email: cnarvaez@udo.edu.ve

Introducción

Un Termistor es un resistor sensible a la temperatura. Mientras las termocuplas son los transductores de temperatura más versátiles y los RTD son los más estables, la palabra que mejor describe a los termistores es la sensibilidad. De estos tres sensores, el termistor exhibe mayor cambio en el parámetro resistencia en función de pequeños cambios de temperatura.

Los termistores están generalmente compuestos de materiales semiconductores y existen básicamente dos tipos: los de coeficiente negativo de temperatura (NTC) y los de coeficiente positivo de temperatura (PTC). Los primeros son los más usados y disminuyen su resistencia con el incremento de temperatura.

El precio que se paga por el incremento en la sensibilidad, es la pérdida de linealidad. Efectivamente, el termistor es un dispositivo extremadamente no lineal y su curva característica varía según sea el fabricante.

La curva característica de un termistor individual puede ser aproximada a través del uso de la ecuación de Steinhart-Hart:

$$1/T = A + \ln(R) + C(\ln R)^3$$

Donde:

T = Temperatura en °Kelvin

R = Resistencia del Termistor en Kohm

A, B, C = Constante de la curva de aproximación

Las constantes A, B, y C pueden ser calculadas seleccionando tres puntos de la tabla o curva que acompaña el termistor y resolviendo un sistema de ecuaciones simultáneas de tres incógnitas.

Cuando los puntos son escogidos en un rango que abarca unos 100°C considerando el centro del rango nominal de temperatura del termistor, esta ecuación se aproxima a +/- .02% de la curva característica.

Propósito

Este trabajo tiene como propósito la realización de un Termómetro digital basado en un termistor 10Kohm +/- 1% @ 25°C y un PIC16F873, el cual resulta apropiado por tener convertidores analógico digital de 10 bits de resolución y suficientes líneas de I/O para el manejo de un módulo LCD.

Cálculo de las constantes de la ecuación de Steinhart-Hart

El termistor usado tiene la siguiente tabla característica:

Temp(°C)	R(Kohm)	Temp(°C)	R(Kohm)
-50	320.2	30	8.313
-45	247.5	35	6.941
-40	188.4	40	5.826
-35	144.0	45	4.912
-30	111.3	50	4.161
-25	86.39	55	3.537
-20	67.74	60	3.021
-15	53.39	65	2.589
-10	42.45	70	2.229
-5	33.89	75	1.924
0	27.28	80	1.669
5	22.05	85	1.451
10	17.96	90	1.366
15	14.68	95	1.108
20	12.09	100	.9375
25	10.00	105	.8575

Para el cálculo de las constantes se resuelve el sistema de ecuaciones simultáneas de tres incógnitas para los puntos -50°C, 25°C y 100°C. Recuerde que °Kelvin = °C + 273.15.

$$\begin{aligned} & \left[A + (\ln 27.28)B + (\ln 27.28)^3 C \right]^{-1} \\ & \left[A + (\ln 10.0)B + (\ln 10.0)^3 C \right]^{-1} \\ & \left[A + (\ln 2.229)B + (\ln 2.229)^3 C \right]^{-1} \end{aligned}$$

Para resolver este sistema de ecuaciones se utilizó MATLAB el cual arrojó el siguiente resultado:

A = 0.00269794 B=0.00028033 C=0.00000086

Entonces, la ecuación de Steinhart-Hart para este termistor es:

$$T_{\circ K} = 1 / (0.00269794 + 0.00028033 \ln(R) + 0.00000086 \ln(R)^3)$$

$$T_{\circ C} = (1 / (0.00269794 + 0.00028033 \ln(R) + 0.00000086 \ln(R)^3)) - 273.15$$

Interfaz Termistor-PIC

La interfaz Termistor-PIC, consiste en un divisor de voltaje realizado con el termistor (R_{term}) y una resistencia de 10K (R_1), tal como se muestra en la fig. 1. Como voltaje de referencia se utiliza +5V.

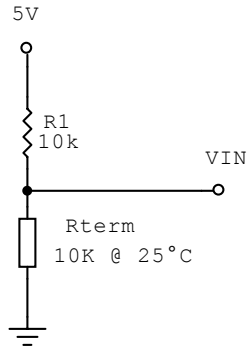


Fig. 1 Interfaz Termistor-PIC

Veamos:

$$(1) \quad V_{in} = R_{term} / (R_1 + R_{term}) * V_{ref}$$

En este ejemplo se utiliza un convertidor analógico digital de 10Bits de resolución, luego los valores entre 0.0 y V_{ref} son cuantificados como valores de 10 Bits desde 0 hasta 1024 decimal (V_{ad}).

$$(2) \quad V_{in} = V_{ad} / 1024 * V_{ref}$$

Luego:

Igualando (1) y (2) y cancelando V_{ref} , R_{term} puede ser expresada como:

$$R_{term} = (R_1 * V_{ad}) / (1024 - V_{ad})$$

Conocida R_{term} la temperatura en °Kelvin puede ser calculada usando la ecuación de Steinhart-Hart deducida anteriormente:

$$T_{\circ K} = 1 / (0.00269794 + 0.00028033 \ln(R_{term}) + 0.00000086 \ln(R_{term})^3)$$

Software

```

/*****
/*
/* Termometro.c
/*          Termómetro digital usando termistor
/*          para el PIC16F873
/*
/*
/* Autor: Carlos Narvaez 15/06/2004
/* Rutina LCD Original: J. Wimpenny
/*
/*-----
/* LCD : OPTREX Corporation DMC16249UB
/* LCD tipo: HD44780
/* Compilador: CSC PCM C compiler
/* termistor: 10K @ 25°C
/*
/* Interfaces:
/*
/*          Termistor = Port A bit 0
/*          Data_5    = Port B bit 0
/*          Data_6    = Port B bit 1
/*          Data_7    = Port B bit 2
/*          Data_6    = Port B bit 3
/*          RS        = Port B bit 4
/*          W/R       = Port B bit 5
/*          Enable    = Port B bit 6
/*          Esta version es para el modo de operación 4 bit
/*
/*
/*****
#include <16f873.h>
#define ADC=10
#include <math.h>
#define XT,NOWDT,PUT, NOPROTECT, NOBROWNOUT,NOLVP,WRT,NOCPD
#define delay (clock=4000000)
/*****
/* Prototipos
/*****
void LCD_Setup(void);
void LCD_FunctionMode(void);
void LCD_DataMode(void);
void LCD_Write_8_Bit(char);
void LCD_Write_4_Bit(char);
void LCD_Delay(void);
void delay_s(int);
void calcule_temp(long int);
void write_temp(void);
//void WriteLCDString( const char *lcdptr );
/*****
/* Definiciones Generales
/*****
#define OPTION_REG = 0x81
long int adc_var;
float r_term, T_C, LnRes, LnRes3;

```

PIC Interfaces

```

/*****
/* Definiciones para la interfaz PIC-LCD
/*****
#define TER_IN      PIN_A0      /* Interfaz Termistor-PIC
#define LCD_DATA_4  PIN_B0      /* LCD BIT 4
#define LCD_DATA_5  PIN_B1      /* LCD BIT 5
#define LCD_DATA_6  PIN_B2      /* LCD BIT 6
#define LCD_DATA_7  PIN_B3      /* LCD BIT 7
#define LCD_RS      PIN_B4      /* Register Select
#define LCD_WR      PIN_B5      /* Escribir/Leer
#define LCD_SEL     PIN_B6      /* Habilita LCD
/*
/*
/*****
/* Comandos del LCD ( Segun LCD Data Sheet )
/*****
/*
#define clear_lcd      0x01 /* Clear Display
#define return_home    0x02 /* Cursor to Home position
#define entry_mode     0x06 /* Normal entry mode
#define entry_mode_shift 0x07 /* - with shift
#define system_set_8_bit 0x38 /* 8 bit data mode 2 line ( 5x7 font )
#define system_set_4_bit 0x28 /* 4 bit data mode 2 line ( 5x7 font )
#define display_on     0x0c /* Switch ON Display
#define display_off    0x08 /* Cursor plus blink
#define set_dd_line1   0x80 /* Line 1 position 1
#define set_dd_line2   0xC0 /* Line 2 position 1
#define set_dd_ram     0x80 /* Line 1 position 1
#define write_data     0x00 /* With RS = 1
#define cursor_on     0x0E /* Switch Cursor ON
#define cursor_off    0x0C /* Switch Cursor OFF
/*
/*****
/* Variables configuracion del PIC
/*****
#define TxMode      0x40 /* Set option reg TMR0 interrupt
/* source from counter.
/* Max speed prescaler ( 1:2 )
/* ( Not needed for LCD )
#define PortAConfig 0x01 /* Configure port A ( 1 = input )
/* RA4/RTCC = Input / RA2 = Input
#define PortBConfig 0x00 /* Configure port B
/*****
/* Programa Principal
/*****
void main(void)
{

/* Configuración puertos PIC e Interrucciones */

OPTION_REG = TxMode; /* Set Option register.*/
set_tris_a( PortAConfig); /* Setup ports */
set_tris_b( PortBConfig );
output_low(LCD_SEL);
disable_interrupts(GLOBAL);

setup_adc_ports(RA0_ANALOG);
setup_adc(ADC_CLOCK_DIV_32);
SET_ADC_CHANNEL(0);

/* End of Setup */

```

PIC Interfaces

```
LCD_Setup(); /* Setup the LCD device */

LCD_Write_4_Bit('T'); /* Display a test message */
LCD_Write_4_Bit('E');
LCD_Write_4_Bit('R');
LCD_Write_4_Bit('M');
LCD_Write_4_Bit('O');
LCD_Write_4_Bit('M');
LCD_Write_4_Bit('E');
LCD_Write_4_Bit('T');
LCD_Write_4_Bit('R');
LCD_Write_4_Bit('O');
delay_s(5);
LCD_FunctionMode();
LCD_Write_4_Bit(set_dd_line2); /* Goto line 2 */
LCD_DataMode();

while( 1 )
{
    adc_var = READ_ADC();
    delay_us(10);
    calculate_temp(adc_var);
    write_temp();
    LCD_FunctionMode();
    LCD_Write_4_Bit(set_dd_line2); /* Goto line 2 */
    LCD_DataMode();
    delay_s(5);
}

}

/* END OF MAIN */

/*****
/* Setup the lcd device */
*****/
void LCD_Setup(void)
{
    /* Reset the LCD */

    delay_ms(30); /* Power up delay */
    LCD_FunctionMode();

    LCD_Write_8_Bit( system_set_4_bit ); /* This sequence resets the LCD */
    delay_ms(5);
    LCD_Write_8_Bit( system_set_4_bit );
    delay_ms(5);
    LCD_Write_8_Bit( system_set_4_bit );
    delay_ms(5);

    LCD_Write_4_Bit( system_set_4_bit );
    delay_ms(2);
    LCD_Write_4_Bit( display_off );
    delay_ms(2);
    LCD_Write_4_Bit( entry_mode );
    delay_ms(2);
    LCD_Write_4_Bit( display_on );
    delay_ms(2);
    LCD_Write_4_Bit( set_dd_ram );
}
```

```
delay_ms(2);
LCD_DataMode();
}

/*****/
/* LCD modo comandos */
/*****/
void LCD_FunctionMode(void)
{
    output_low(LCD_RS);
    LCD_Delay();
}

/*****/
/* LCD modo Data */
/*****/
void LCD_DataMode(void)
{
    output_high(LCD_RS);
    LCD_Delay();
}

/*****/
/* Escribe un byte en el LCD */
/* Modo 8 Bit */
/*****/
void LCD_Write_8_Bit(char d )
{
    output_low(LCD_WR); /* modo Write */
    LCD_Delay();

    /* Setup data */

    if ( d & 0x80 )
        output_high(LCD_DATA_7);
    else output_low(LCD_DATA_7);

    if ( d & 0x40 )
        output_high(LCD_DATA_6);
    else output_low(LCD_DATA_6);

    if ( d & 0x20 )
        output_high(LCD_DATA_5);
    else output_low(LCD_DATA_5);

    if ( d & 0x10 )
        output_high(LCD_DATA_4);
    else output_low(LCD_DATA_4);

    LCD_Delay();
    output_high(LCD_SEL); /* Select LCD */
    LCD_Delay();
    output_low(LCD_SEL); /* De-select LCD */
}
}
```

```

/*****
/* Escribe un byte en el LCD      */
/* Modo 4 Bit                    */
*****/
void LCD_Write_4_Bit(char d )
{
    output_low(LCD_WR);          /* Write Mode      */
    LCD_Delay();

                                /* Output Higher 4 bits */

    if ( d & 0x80 )
        output_high(LCD_DATA_7);
    else output_low(LCD_DATA_7);

    if ( d & 0x40 )
        output_high(LCD_DATA_6);
    else output_low(LCD_DATA_6);

    if ( d & 0x20 )
        output_high(LCD_DATA_5);
    else output_low(LCD_DATA_5);

    if ( d & 0x10 )
        output_high(LCD_DATA_4);
    else output_low(LCD_DATA_4);

    LCD_Delay();
    output_high(LCD_SEL);
    LCD_Delay();
    output_low(LCD_SEL);        /* Clock in the data */
    LCD_Delay();

    d <<= 4;                    /* Output Lower 4 bits */

    if ( d & 0x80 )
        output_high(LCD_DATA_7);
    else output_low(LCD_DATA_7);

    if ( d & 0x40 )
        output_high(LCD_DATA_6);
    else output_low(LCD_DATA_6);

    if ( d & 0x20 )
        output_high(LCD_DATA_5);
    else output_low(LCD_DATA_5);

    if ( d & 0x10 )
        output_high(LCD_DATA_4);
    else output_low(LCD_DATA_4);
}

```

```
LCD_Delay();
output_high(LCD_SEL);
LCD_Delay();
output_low(LCD_SEL);           /* Clock in the data */

}

/*****
/* Retardo temporal para LCD */
/* Adjuste segun el LCD que use */
*****/
void LCD_Delay(void)
{
    delay_ms(1);
}

/*****
/* Retardo segundos */
*****/
void delay_s(int n)
{
    for (;n!=0; n--)
        delay_ms(1000);
}

/*****
/* Calculo de Temperatura */
*****/
void calcule_temp(long int adc_var)
{
    r_term = ((10.0*adc_var)/(1024-adc_var));
    LnRes = log(r_term);
    LnRes3 = LnRes*LnRes*LnRes;
    T_C = (1/(0.00269794+0.00028033*LnRes+0.00000086*LnRes3))-273.15;
}

/*****
/* Escribe Tempertura en LCD */
*****/
void write_temp(void)
{
    printf(LCD_Write_4_Bit, "%5.2f", T_C);
}
}
```