

# **Diseño de Sistemas Mecatrónicos**

## **Laboratorio**



## Medición de temperatura por computadora

---

### Introducción

---

*[removido por copyright]*

#### Sensor utilizado

El sensor utilizado esta práctica es el LM35DZ. Este sensor ofrece una salida de voltaje proporcional a la temperatura, con una tasa de cambio de  $10 \frac{mV}{C^{\circ}}$ . La temperatura puede ser determinada conociendo el voltaje de salida determinado en cierto tiempo, y detectando cambios en el voltaje de salida, interpretándolos con la tasa de cambio mencionada. La ecuación que relaciona la salida de voltaje con la temperatura es:

$$V_o(T) = V_0 + (T - 2) \left( 10 \frac{mV}{C^{\circ}} \right)$$

Donde  $V_0$  es el voltaje en  $mV$  cuando la temperatura es  $2 C^{\circ}$  (el sensor fue conectado en configuración básica, donde el rango de medición es de 2 a  $150 C^{\circ}$ ).

### Trabajo

---

#### Desarrollo

En esta sección del reporte presentamos los detalles de la implementación de la práctica. Lo principal que hay que comentar al respecto es que se optó por trabajar en una plataforma diferente a la propuesta por el documento de la práctica (NI ELVIS y LabView). Las herramientas utilizadas en la PC *host* fueron:

- **pyserial** [2] para acceso a puerto serial.
- **numpy** [3] para operaciones con arreglos de números de punto flotante.
- **matplotlib** [4] para graficas en 2-D.
- **wxpython** [5] para construcción de interfaces gráficas.

La interfaz entre sensor y PC *host* fue implementada con un microcontrolador AVR modelo ATMEGA164PA, montada en la tarjeta de desarrollo STK500. Las herramientas utilizadas para el programa del ATMEGA164PA fueron:

- **AVR libc** [6]. Librería para microcontroladores AVR.
- **GCC** [7] para compilar el código C a código máquina del AVR.

Y el hardware:

- **ATMEGA164PA**
- **LM35DZ**

El primer paso del procedimiento fue realizar la práctica sin el sensor. La variación de voltaje que el microcontrolador detecta (por medio de un módulo integrado de conversión análoga-digital) y reporta (por medio de un módulo integrado UART) fue emulada con un simple potenciómetro. La idea era que al comprobar que todo el sistema funcionaba hasta este punto, solo restaría agregar el sensor y realizar una calibración correspondiente a las temperaturas que se desean medir.

El microcontrolador y la aplicación en la PC host están enlazados por medio del puerto serial con una configuración estándar (ver código C de microcontrolador anexo). Cada cierto lapso de tiempo, la aplicación le pide al microcontrolador que realice una conversión análoga-digital y reporte el valor obtenido. Esto se realiza periódicamente, al mismo tiempo que la gráfica del voltaje contra el tiempo se actualiza en la pantalla.

La interfaz gráfica fue inicialmente construida con **wxGlade**, un editor gráfico para la plataforma **wxWidgets** que produce código **wxpython** entre otros (ver Figura 1).

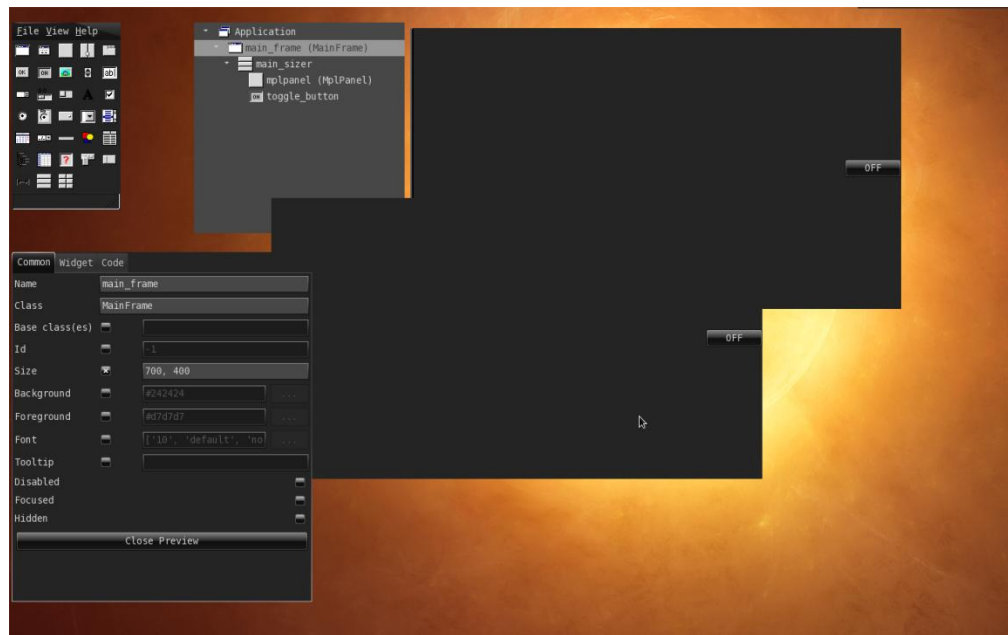


Figure 1. Construyendo la interfaz gráfica.

Para terminar con una interfaz gráfica ilustrada en Figura X.

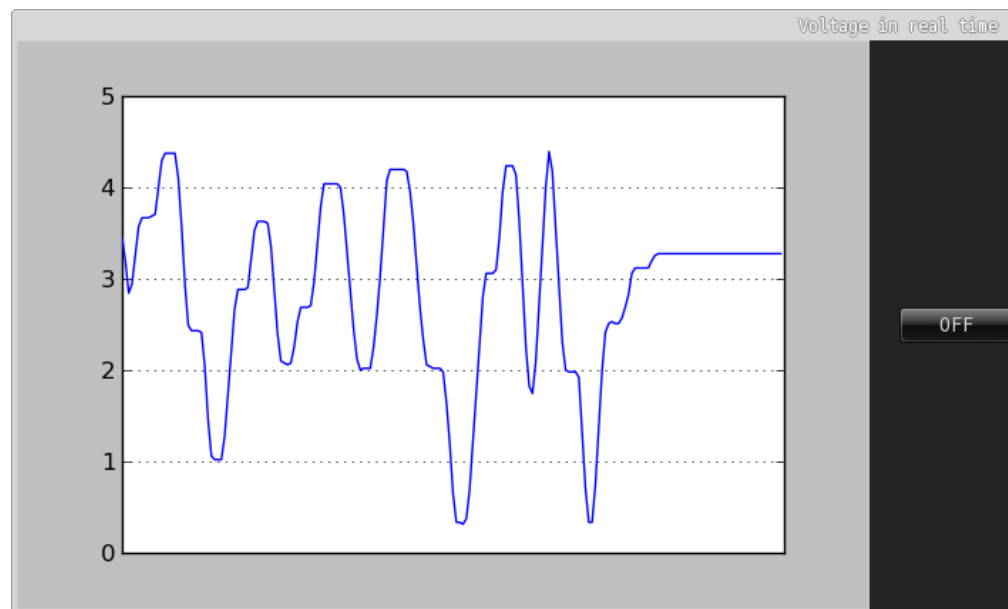
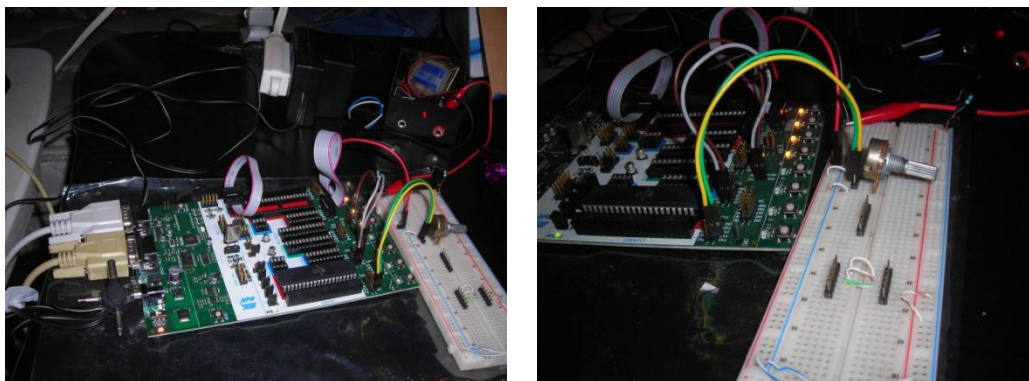


Figure 2. Interfaz gráfica de prototipo de aplicación, en ejecución.

El código resultante de la sesión en wxGlade debe ser modificado para incluir lo siguiente:

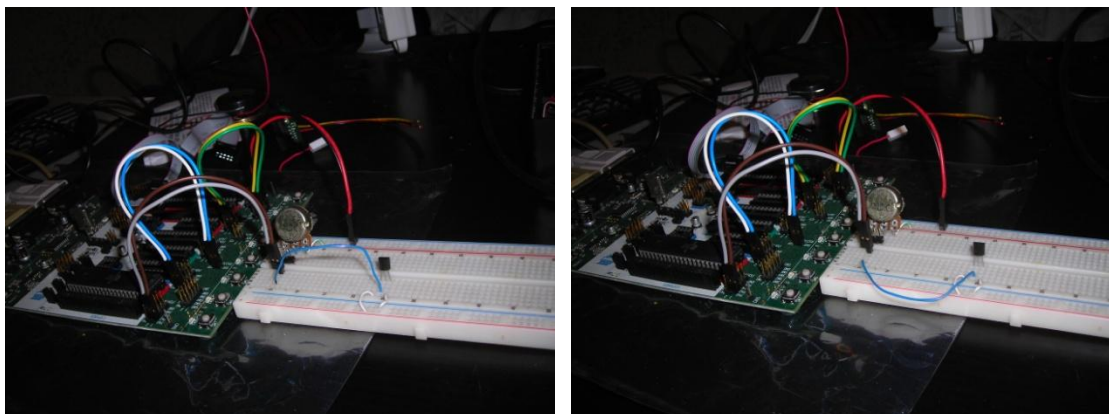
- El área de la gráfica.
- La actualización periódica de la gráfica.
- La conducta de encendido-apagado con el botón.
- La recepción y transmisión de la aplicación al microcontrolador para ejecutar la conversión análoga-digital y recuperar el dato.

Todo esto se implementó en **Python**, y se puede revisar en el código anexo.



**Figure 3. Etapa de pruebas.**

Después de comprobar que el sistema se comporta como se esperaba, el LM35DZ se añade al circuito.



**Figure 4. Pruebas con el LM35DZ.**

También se actualizó la interfaz gráfica (Figura 5). No es necesario medir la temperatura a frecuencias muy altas (1 MHz por ejemplo) ya que usualmente las frecuencias de variación de temperatura son relativamente lentas. En otras palabras, es poco común que en procesos o cuerpos se den cambios de temperatura (transferencias de calor) en periodos de tiempo cortos. Ajustar la resolución a un nivel aceptable permite ahorrar recursos del sistema.

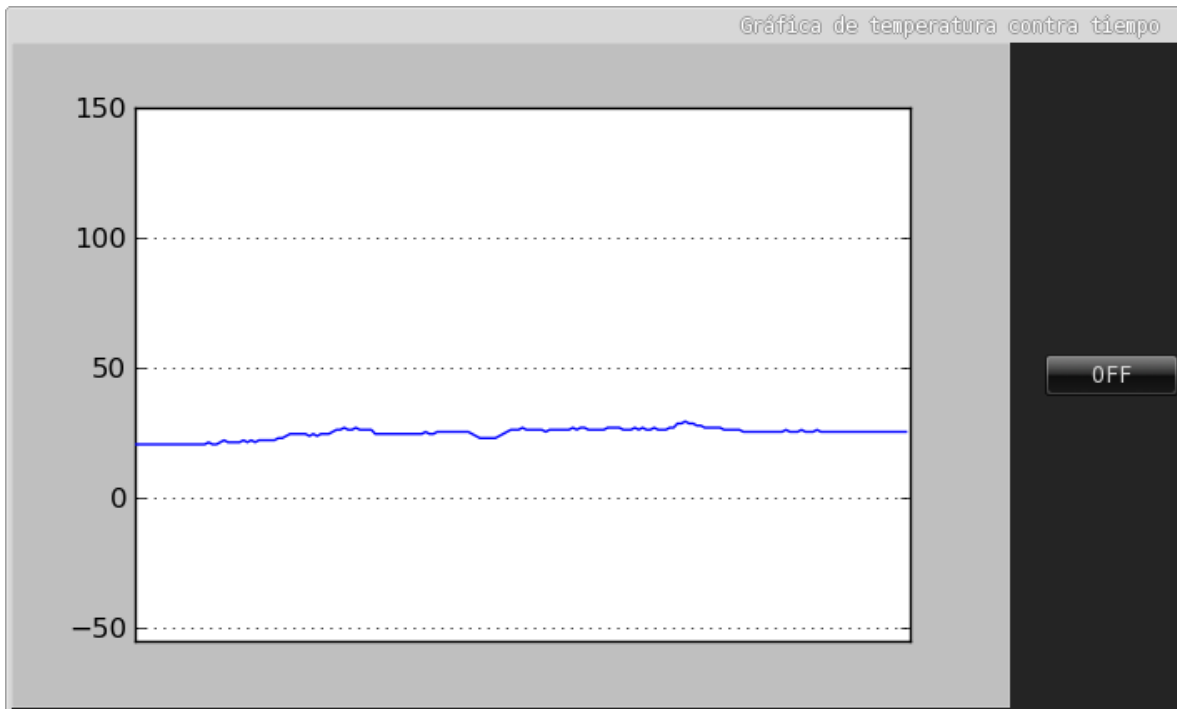
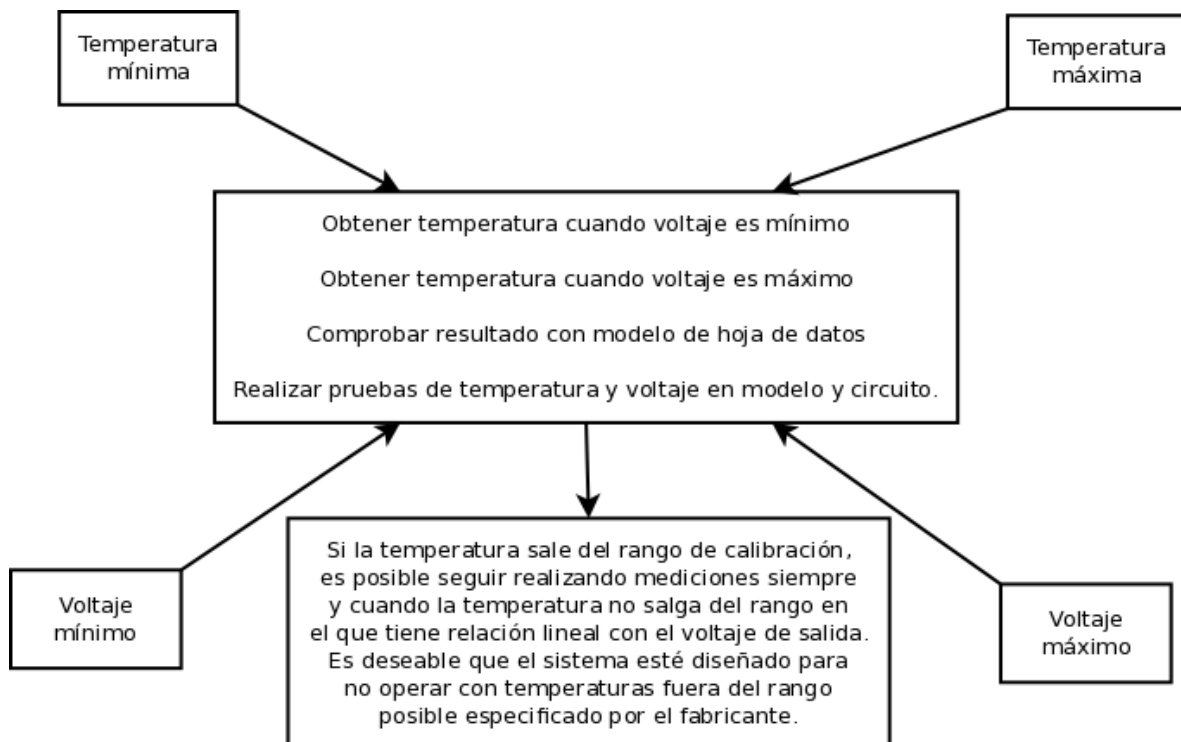


Figure 5.

## Calibración



---

## Conclusión

---

Es frecuente que en sistemas de control se requiera medir la temperatura de un cuerpo o proceso. La presente práctica implementa un sistema de medición de temperatura por computadora con uno de los varios sensores que se pueden utilizar para tal propósito.

La experimentación y el reporte de la práctica fueron realizados en un periodo en el que por diversas razones fue difícil el acceso al equipo de laboratorio de la materia; por eso fue necesario utilizar otra plataforma de desarrollo. Además de lo apenas mencionado, las razones por las cuales se utilizó una plataforma de desarrollo diferente son:

- Cada una de las herramientas de software utilizadas pueden ser adquiridas gratuitamente.
- La aplicación resultante es portable, y no está atada a una sola plataforma, fabricante, etc (la aplicación puede ser ejecutada en ambientes UNIX y NT, fue probada en Fedora 13 y Windows XP Home Edition).
- El que la aplicación pueda ser desarrollada y ejecutada en muchos tipos de computadoras personales significa que el tiempo de desarrollo e implementación puede ser utilizado de manera más eficiente que tener que asistir a un laboratorio de la universidad.

La calibración es un proceso que consiste en comparaciones de varias medidas, en relación con una medida considerada estándar o guía. En la práctica se revisó el proceso de calibración.

---

## Referencias

---

- [1] Pallás Areny, Ramón (2007). *Sensores y Acondicionadores de Señal* (480 páginas). 4ª Edición. Alfaomega - Marcombo.
- [2] <http://pyserial.sourceforge.net/>
- [3] <http://www.numpy.org/>
- [4] <http://matplotlib.sourceforge.net/>
- [5] <http://wxpython.org/>
- [6] <http://www.nongnu.org/avr-libc/>
- [7] <http://gcc.gnu.org/>