



Universidad de Valencia

FACULTAD DE FÍSICA

TERMÓMETRO DIGITAL

Autores:

Pablo Nicolás Fuentes y Tere Campayo Gutiérrez

Profesora:

Silvia Casans Berga

Marzo - Mayo 2025

Índice

1. Introducción	2
2. Sensado y transducción	2
3. Amplificación	3
4. Digitalización o conversión a digital	4
5. Visualización	6
6. Activación de la alarma	6
7. Desactivación de la alarma	8
8. Manejo del Arduino Nano	9
9. Cuestionario	13
10. Conclusión	14

1. Introducción

Este proyecto que hemos realizado es un trabajo final de la asignatura de electrónica, el cual tiene como objetivo montar un termómetro digital. En resumen, el circuito consta de un medidor de temperatura y cuando la temperatura es mayor a $28,5^{\circ}\text{C}$ aproximadamente se enciende un LED situado en el centro del montaje y este no se apaga hasta que la temperatura baja de $25,5^{\circ}\text{C}$ aproximadamente. Además, incorpora un Arduino Nano que nos extrae el valor de la temperatura por segundo por la pantalla, pudiendo utilizar estos datos como queramos. El circuito está diseñado para que se encienda el LED a los 30°C y se apague a los 27°C , sin embargo, por un offset del conversor esto no es exactamente así. Entraremos en detalle más adelante.

El proyecto nos llevará unas 3-4 sesiones de trabajo en el laboratorio y algo de trabajo en casa. El proyecto puede dividirse en 5 fases:

1. El sensado: se encarga de medir los valores de la temperatura, se basa en propiedades de materiales variables con la temperatura, en nuestro caso es el *LM35*. Este es un sensor de temperatura basado en semiconductores, aprovechando la sensibilidad que tienen las uniones semiconductoras frente a la temperatura.
2. La transducción: consiste en transcribir los valores de la temperatura a valores de tensión, en nuestro caso también lo hace directamente el *LM35*.
3. Acondicionamiento: es la fase de modificar nuestra señal ya sea amplificándola, atenuándola o filtrándola. En nuestro caso, íbamos a amplificar la señal con el amplificador no inversor *UA741*, sin embargo, dado que el error es menor para *OP07* vamos a utilizar este último a pesar de lo que indica el guión.
4. Digitalización: convertiremos las señales de tensión analógicas en digitales. Para ello, utilizaremos el conversor *ADC0804*.
5. Tratamiento y almacenamiento: procesamos los datos obtenidos mediante sistemas digitales y los almacenamos para trabajar con ellos posteriormente.

Cada una de las fases la hemos dividido en más secciones en función de los componentes que vamos añadiendo al circuito, y cada una está explicada en profundidad.

2. Sensado y transducción

En primer lugar, para comenzar este proyecto, vamos a comprobar el funcionamiento del sensor *LM35*, este se encargará de medir la temperatura que hay en su entorno y extraerla como voltaje. Primero, es importante consultar en la ficha técnica la relación entre la temperatura y el voltaje de salida; en su caso es 10mV por cada $^{\circ}\text{C}$. En la figura 1 podemos ver sus terminales para, a continuación, saber cómo conectarlo.

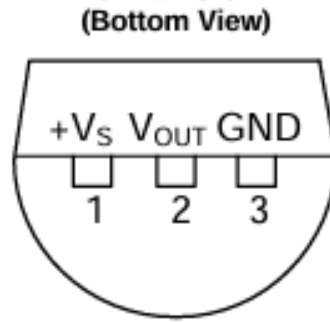


Figura 1: Sensor LM35

A continuación, para comprobar su funcionamiento, conectamos el sensor *LM35* a 5V de corriente continua mediante el microinstructor, a masa y al voltímetro, como se ve en la figura 2.

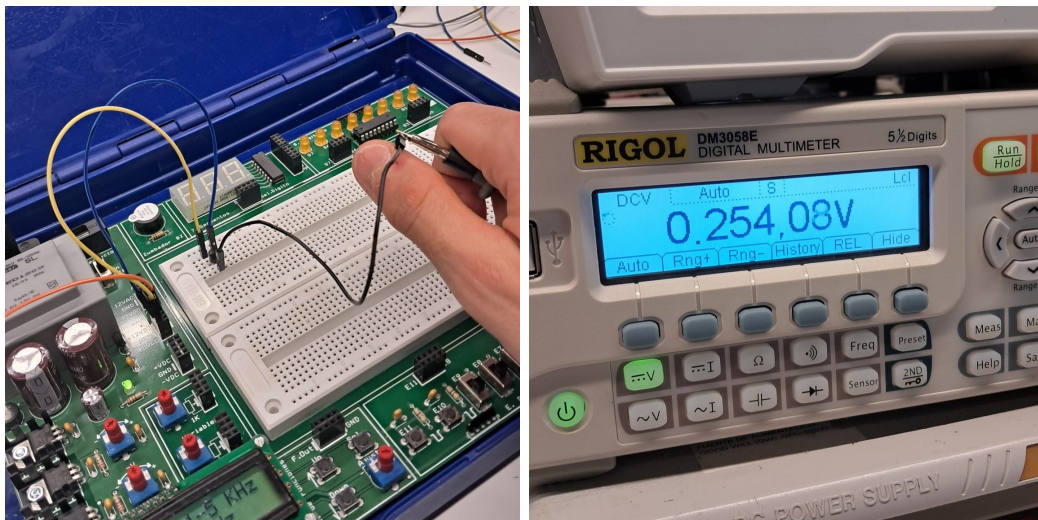


Figura 2: Montaje del sensor y voltaje observado en el voltímetro

En el voltímetro observamos que el voltaje es alrededor 0,254V (figura 2) que correspondería con la temperatura ambiente de 25,4°C. Al colocar el dedo sobre el sensor, vemos que la temperatura aumenta ya que hay un cambio en el voltaje que medimos con el voltímetro y disminuye al dejar de tocar el sensor.

3. Amplificación

Continuando con la amplificación, tomamos el amplificador no inversor *OP07*. El motivo de escoger este y no el *UA741* es debido a que su error es mucho menor. Lo primero que tenemos que hacer es diseñar la ganancia de nuestro amplificador para que abarque todo el rango de temperatura. Por lo que dado que el rango de voltaje es $V_{LM35} = 0 - 0,5V$ y $V_{ADC0804} = 0 - 5V$ calculamos la ganancia

$$G = \frac{V_{ADC0804}}{V_{LM35}} = \frac{5}{0,5} = 10$$

Así, dado que la ganancia del amplificador no inversor viene dada por

$$G = 1 + \frac{R_f}{R} = 10 \Rightarrow \frac{R_f}{R} = 9$$

por lo que escogemos una resistencia $R = 10k\Omega$ y para la resistencia $R_f = 90k\Omega$ utilizamos el potenciómetro para poder ajustar correctamente el voltaje de la salida del amplificador.

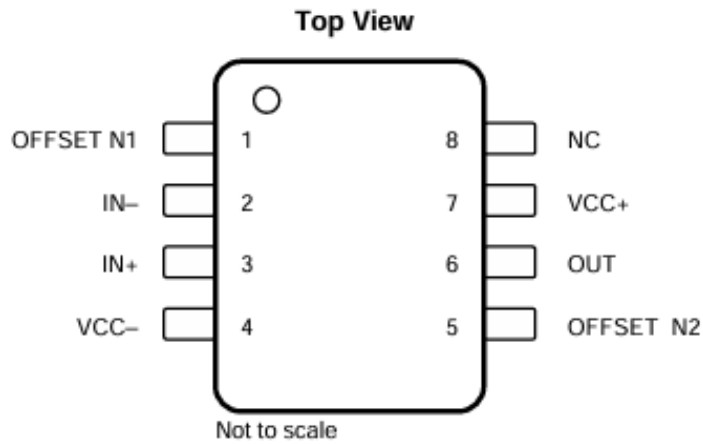


Figura 3: Top view amplificador OP07

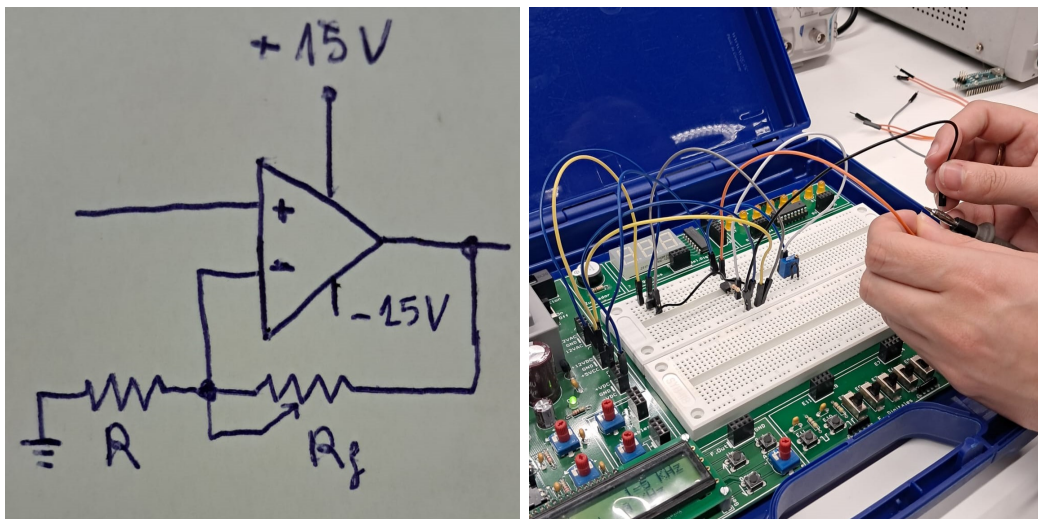


Figura 4: Diseño del amplificador no inversor y montaje con el amplificador

Por lo tanto, una vez diseñado (figura 4), montamos por separado el circuito del amplificador y comprobamos que introduciendo $1V$ en V_+ obtenemos $10V$ a la salida del amplificador, que ajustamos con el potenciómetro. Una vez comprobamos que el funcionamiento es correcto, conectamos la salida del sensor con V_+ y vemos que el valor que nos da el voltímetro es el esperado: en torno a $2,5V$.

4. Digitalización o conversión a digital

En lo que se refiere a la conversión analógico a digital, queremos que finalmente nos quede un circuito tal y como el de la figura 5.

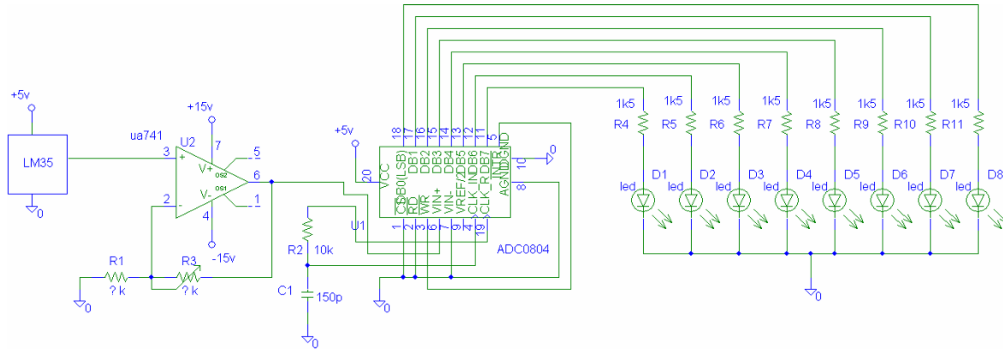


Figura 5: Diseño completo del sensado, amplificador, convertor y visualización

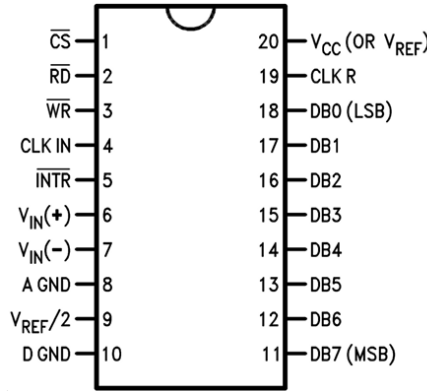


Figura 6: Top view ADC0804

Antes de montar el circuito de la figura 5 debemos comprobar que convertor analógico-digital funciona correctamente. Para ello, montamos el convertor de manera individual (a partir de la figura 6) conectamos a la entrada del convertor (pin 6) del convertor primero $0V$ y observamos con el voltímetro que a la salida (pin 11-18) el voltaje es aproximadamente 0 . Seguidamente, conectamos a la entrada (pin 6) un voltaje de $5V$ y vamos comprobando que tal y como esperábamos todos los pines del 11 al 18 están aproximadamente a $5V$ según el voltímetro. Ahora, dado que el funcionamiento es correcto, procedemos a conectar la salida del amplificador con la entrada del *ADC0804*. Como la temperatura ambiente estaba a unos $25^{\circ}C$ aproximadamente tenemos que ver qué número es en binario $2,5V$. Para eso debemos determinar 1 LSB (Least Significant Bit) que representa la resolución mínima que puede distinguir un convertor analógico-digital

$$1LSB = \frac{FS}{2^n} = \frac{5}{256} \simeq 20mV = 0,02V$$

sabiendo podemos calcular el número de decimal que corresponde $2,5V$ que llegan al convertor y pasarlo a binario, estos son:

$$N^{\circ}decimal = \frac{2,5}{0,02} = 125, \quad N^{\circ}binario = 1111101$$

así comprobamos que en las salidas 18, 16, 15, 14, 13, 12 y 11 del convertor tenemos alrededor de $5V$ y alrededor de $\sim 1V$ en el pin 17, verificando el buen funcionamiento del convertor.

5. Visualización

Ahora sí, comenzando con la parte de visualización, conectamos tal y como aparece en la figura 5, las salidas del conversor (pines del 11 al 18) a los LEDs del microinstructor. Completando el diseño de la figura 5, el montaje puede verse en la figura 7. En la figura 8 vemos que los LEDs se encienden tal y como esperábamos para 25°C .

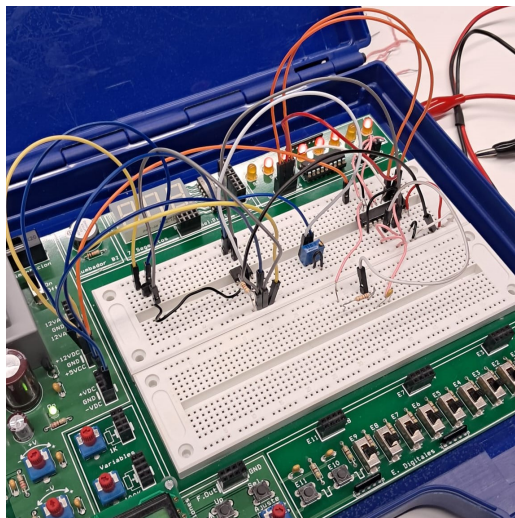


Figura 7: Montaje completo del sensado, amplificador, conversor y visualización

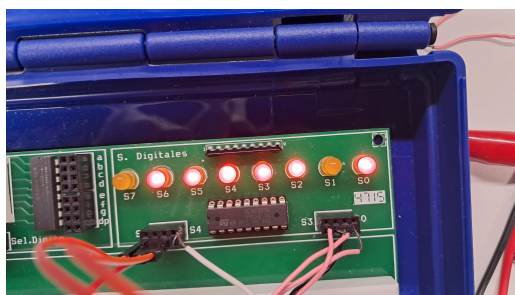


Figura 8: Visualización digital de la temperatura

Además, comprobamos que al contactar el dedo con el sensor aumenta la temperatura lentamente y los LEDs se van encendiendo y apagando, marcando el número binario, lo que indica el buen funcionamiento del circuito hasta el momento.

6. Activación de la alarma

A continuación, nos toca afrontar una parte importante y engorrosa del proyecto. Vamos a conectar a nuestro circuito dos comparadores de 4 bits ($74LS85$) en cascada.



Figura 9: Top view comparador 74LS85

Como siempre, primero comprobamos que funcionan correctamente por separado. Esto lo hacemos a través de las entradas digitales: colocamos dos números en las entradas digitales del microinstructor y las salidas las conectábamos a los LEDs del microinstructor y, dependiendo de qué número era mayor o igual, se encendía el LED correspondiente. Posteriormente, conectamos ambos comparadores en cascada, el diseño se puede ver en la figura 10. El diseño será el mismo para la parte de desactivación de la alarma.

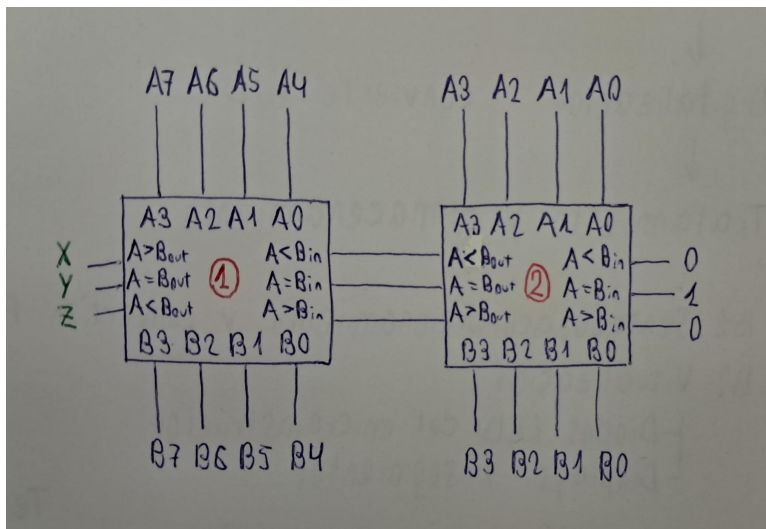


Figura 10: Diseño de los comparadores en cascada

El comparador 1 (señalado en rojo en la figura del diseño) es el comparador que manda sobre el 2, es el que decide qué número binario es mayor, por eso el 2 está conectado a él. Las entradas del comparador 2: $A \leq B_{in}$, $A = B_{in}$ y $A \geq B_{in}$ (inciso: overleaf no me deja poner los símbolos de 'menor que' y 'mayor que' por lo que he tenido que poner 'mayor-igual' y 'menor-igual') están conectadas a las entradas digitales con los valores 0, 1 y 0 respectivamente. Las entradas A7-A0 irán conectadas a las salidas del conversor analógico-digital y las entradas B7-B0 irán conectadas a las entradas digitales del microinstructor. De forma que indicaremos en el microinstructor el número que corresponde a 30°C en binario y este es comparado con la temperatura en binario actual y si la temperatura es mayor, entonces por la salida $A \geq B_{out}$ (X en la figura 10) tendremos un voltaje cercano a 5V que pasará por una resistencia y un LED y

este se encenderá.

Entendido esto, vayamos a la práctica. El número de $30^{\circ}C$ en binario es 10010110; este lo marcamos en las entradas digitales y conectamos la salida del comparador 1, como ya hemos mencionado, y la salida $A \geq B_{out}$ irá a una resistencia conectada a un LED y se encenderá si la temperatura es mayor a $30^{\circ}C$. Vemos que el LED está encendido justo al superar el número binario 10010110 (figura 11). Ya entraremos en los detalles del LED y de la resistencia más adelante porque la salida X no irá directamente a la resistencia y luego al LED. Esto es la comprobación de que funciona correctamente esta parte.

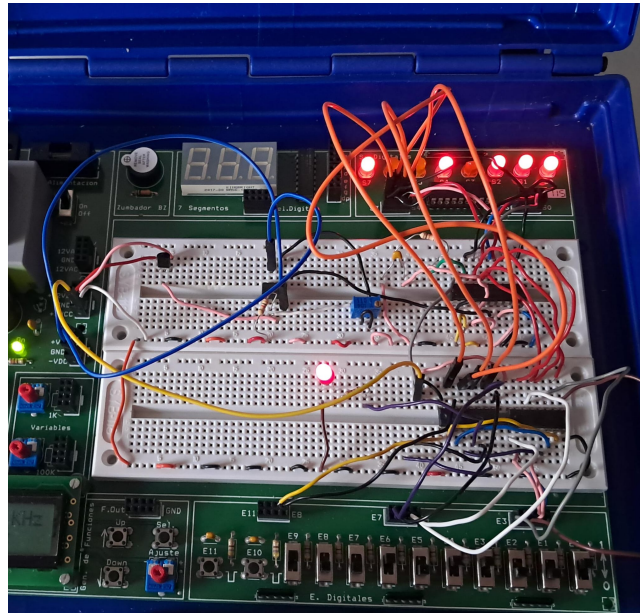


Figura 11: Montaje y comprobación de los comparadores con el resto del circuito

7. Desactivación de la alarma

Como ya hemos mencionado, el LED se debe apagar cuando la temperatura baje de $27^{\circ}C$. Nosotros, para que esto ocurra, hemos tomado otros dos comparadores de 4 bits cada uno, y los hemos montado en cascada de la misma forma que mencionamos anteriormente y que se puede ver en la figura 10. Las entradas A7-A0 irán conectadas también al *ADC0804* y las entradas B7-B0 irán a las entradas digitales, con la particularidad de que $27^{\circ}C$ corresponde con el número binario 10000111, que se asemeja a 10010110 que es $30^{\circ}C$ en binario, en que tienen cuatro bits que son 1 y cuatro que son 0. Esto hace que en las entradas digitales pongamos 10010110 y para comparar 10000111 sólo tenemos que cambiar el orden de los cables. La conexión de los cables de los comparadores la visualizamos en la figura 12. Somos conscientes de que no hacía falta hacerlo de esta manera y podríamos haber buscado que fuesen $26^{\circ}C$ cuando se apague el LED, pero dado que ya nos funcionaba consideramos que era mejor no tocar nada más dada la cantidad de cables.

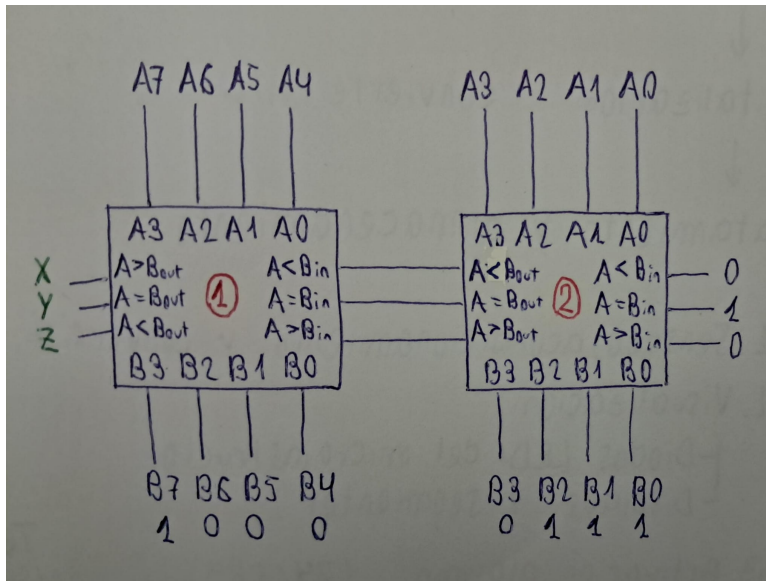


Figura 12: Diseño de los otros 2 comparadores y conexiones B7-B0 con las entradas digitales

Conectados los comparadores vamos a hacer uso del biestable *CD4043B*, este tiene muchas entradas y salidas pero simplemente conectamos la salida $A \geq B_{out}$ de los comparadores que comparan $30^{\circ}C$ a S1 (pin 4), la salida $A \leq B_{out}$ de los comparadores que comparan $27^{\circ}C$ a R1 (pin 3), V_{SS} (pin 8) lo conectamos a masa y ENABLE (pin 5) y V_{DD} (pin 16) los conectamos a 5V. Finalmente, Q1 (pin 2) irá conectado a la resistencia que luego irá al LED que mencionamos anteriormente. Utilizaremos una resistencia de 100Ω para que al LED no le llegue demasiado voltaje y le entre la corriente adecuada; por último, el terminal negativo del LED irá a masa.

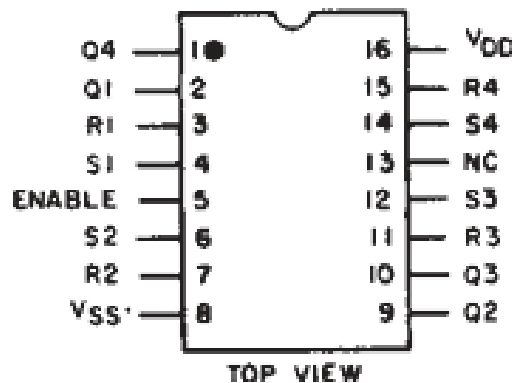


Figura 13: Biestable CD4043B

8. Manejo del Arduino Nano

Ya tenemos prácticamente el circuito completo, solo nos falta colocar el Arduino Nano. Su conexión es muy simple. Al colocarlo en la placa base, conectamos el terminal GND a tierra y el terminal A0 a la salida del amplificador (pin 6). A su vez, estará conectado mediante el puerto USB a nuestro ordenador. En la figura 14, podemos ver el diseño final del circuito.

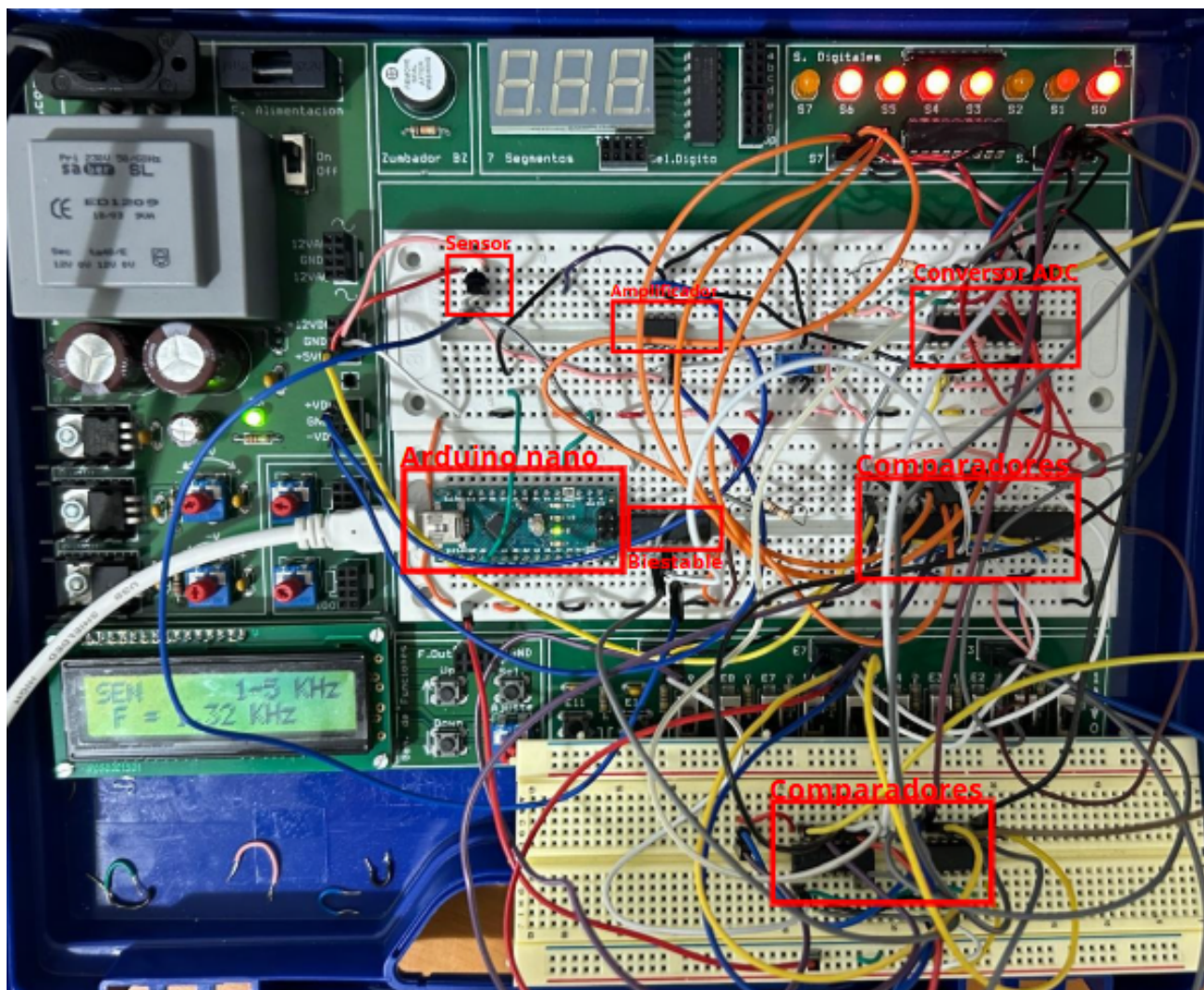


Figura 14: Montaje circuito final

En cuanto a la parte de software, hemos realizado un código que permite visualizar cada segundo la temperatura por el monitor serie de Arduino IDE, cuál es el mínimo y el máximo de temperatura alcanzado y la media. Además, cuando la temperatura es mayor a $29,67^{\circ}\text{C}$ (aproximadamente la temperatura según el programa cuando se enciende el LED de nuestro circuito) hemos indicado que aparezca también el mensaje de '¡ALARMA!'. Esto se puede ver en la figura 15.

```

Output  Serial Monitor X
Message (Enter to send message to 'Arduino Nano' on 'COM9')
Temp: 27.08 °C | Min: 0.00 °C | Máx: 28.93 °C | Media: 16.96 °C
Temp: 27.27 °C | Min: 0.00 °C | Máx: 28.93 °C | Media: 16.98 °C
Temp: 27.27 °C | Min: 0.00 °C | Máx: 28.93 °C | Media: 16.99 °C
Temp: 28.01 °C | Min: 0.00 °C | Máx: 28.93 °C | Media: 17.01 °C
Temp: 28.59 °C | Min: 0.00 °C | Máx: 28.93 °C | Media: 17.03 °C
Temp: 28.69 °C | Min: 0.00 °C | Máx: 28.93 °C | Media: 17.04 °C
Temp: 29.03 °C | Min: 0.00 °C | Máx: 29.03 °C | Media: 17.06 °C ¡ALARMA!
Temp: 29.18 °C | Min: 0.00 °C | Máx: 29.18 °C | Media: 17.08 °C ¡ALARMA!
Temp: 29.28 °C | Min: 0.00 °C | Máx: 29.28 °C | Media: 17.10 °C ¡ALARMA!
Temp: 29.42 °C | Min: 0.00 °C | Máx: 29.42 °C | Media: 17.11 °C ¡ALARMA!
Temp: 29.52 °C | Min: 0.00 °C | Máx: 29.52 °C | Media: 17.13 °C ¡ALARMA!
Temp: 29.62 °C | Min: 0.00 °C | Máx: 29.62 °C | Media: 17.15 °C ¡ALARMA!
Temp: 29.67 °C | Min: 0.00 °C | Máx: 29.67 °C | Media: 17.17 °C ¡ALARMA!

```

Figura 15: Visualización temperaturas en el monitor serie

Para la realización del código, primero definimos las variables que vamos a utilizar (figura 16). Convertimos el voltaje en temperatura y realizamos el bucle para determinar el máximo, el mínimo y la media (figura 17).

```
1 // Variables para medición
2 float pinEntrada = A0; // Pin analógico donde se conecta el sensor de temperatura
3 float voltaje = 0.0; // Variable para almacenar el voltaje leído
4 float temperatura = 0.0; // Temperatura calculada en °C
5
6 // Variables estadísticas
7 float tempMin = 1000.0; // Inicialmente un valor alto para registrar mínimos reales
8 float tempMax = -1000.0; // Inicialmente un valor bajo para registrar máximos reales
9 float sumaTemp = 0.0; // Suma acumulada para calcular media
10 int numLecturas = 0; // Número de lecturas realizadas
```

Figura 16: Variables definidas

```
12 void setup() {
13     Serial.begin(9600); // Inicia comunicación serie a 9600 baudios
14 }
15
16 void loop() {
17     // Leer sensor analógico y convertir a voltaje
18     float lectura = analogRead(pinEntrada);
19     voltaje = lectura * 5.0 / 1023.0; // Conversión a voltaje (resolución 10 bits)
20     temperatura = voltaje * 10.0; // Conversión LM35: 10 mV/°C → Voltios * 10 = °C
21
22     // Actualización de estadísticas
23     if (temperatura > tempMax) tempMax = temperatura;
24     if (temperatura < tempMin) tempMin = temperatura;
25     sumaTemp += temperatura;
26     numLecturas++;
```

Figura 17: Conversión de voltaje a temperatura

Y finalmente sacamos todos los valores de la temperatura por el monitor serie y establecemos el bucle para que si la temperatura es mayor a $29,67^{\circ}\text{C}$ escriba '¡ALARMA!' (figura 18).

```
29 // Mostrar información por el monitor serie
30 Serial.print("Temp: ");
31 Serial.print(temperatura);
32 Serial.print(" °C | Mín: ");
33 Serial.print(tempMin);
34 Serial.print(" °C | Máx: ");
35 Serial.print(tempMax);
36 Serial.print(" °C | Media: ");
37 Serial.print(sumaTemp / numLecturas);
38 Serial.print(" °C");
39
40 // Mostrar mensaje de alerta personalizado
41 if (temperatura > 29.67) {
42     Serial.print(" ¡ALARMA!");
43 }
44
45 Serial.println(); // Nueva línea
46
47 delay(1000); // Espera 1 segundo antes de la siguiente lectura
48 }
```

Figura 18: Imprimimos las variables por el monitor serie

Además de esto, hemos realizado un código en python de forma que nos imprima los valores en tiempo real en él y nos los almacene en un fichero '.txt'. Como se puede ver en las figuras 19 y 20. De esta forma podemos graficar los datos o utilizarlos para múltiples funciones.

```
import serial
import time


ser = serial.Serial(port='COM9', baudrate=9600)
time.sleep(2)

try:
    with open('DatosTemp.txt', 'w', encoding='utf-8') as file:
        while True:
            data = ser.readline().decode().strip()
            file.write(data + '\n')
            file.flush()
            print(data)

except KeyboardInterrupt:
    print("\nLectura interrumpida por el usuario.")

finally:
    ser.close()
    print("Puerto serie cerrado correctamente.")
```

Figura 19: Imprimimos las variables por el monitor serie

 Jupyter DatosTemp.txt ✓ hace 4 minutos

	File	Edit	View	Language
1	Temp: 24.29 ºC	Mín: 24.29 ºC	Máx: 24.29 ºC	Media: 24.29 ºC
2	Temp: 24.34 ºC	Mín: 24.29 ºC	Máx: 24.34 ºC	Media: 24.32 ºC
3	Temp: 24.34 ºC	Mín: 24.29 ºC	Máx: 24.34 ºC	Media: 24.32 ºC
4	Temp: 24.39 ºC	Mín: 24.29 ºC	Máx: 24.39 ºC	Media: 24.34 ºC
5	Temp: 24.44 ºC	Mín: 24.29 ºC	Máx: 24.44 ºC	Media: 24.36 ºC
6	Temp: 24.39 ºC	Mín: 24.29 ºC	Máx: 24.44 ºC	Media: 24.36 ºC
7	Temp: 24.44 ºC	Mín: 24.29 ºC	Máx: 24.44 ºC	Media: 24.38 ºC
8	Temp: 24.39 ºC	Mín: 24.29 ºC	Máx: 24.44 ºC	Media: 24.38 ºC
9	Temp: 24.98 ºC	Mín: 24.29 ºC	Máx: 24.98 ºC	Media: 24.44 ºC
10	Temp: 25.46 ºC	Mín: 24.29 ºC	Máx: 25.46 ºC	Media: 24.55 ºC
11	Temp: 25.86 ºC	Mín: 24.29 ºC	Máx: 25.86 ºC	Media: 24.66 ºC
12	Temp: 25.71 ºC	Mín: 24.29 ºC	Máx: 25.86 ºC	Media: 24.75 ºC
13	Temp: 25.71 ºC	Mín: 24.29 ºC	Máx: 25.86 ºC	Media: 24.83 ºC
14	Temp: 26.00 ºC	Mín: 24.29 ºC	Máx: 26.00 ºC	Media: 24.91 ºC
15	Temp: 26.25 ºC	Mín: 24.29 ºC	Máx: 26.25 ºC	Media: 25.00 ºC

Figura 20: Imprimimos las variables por el monitor serie

9. Cuestionario

1. **¿Qué rango de temperaturas podríamos medir si la ganancia del amplificador fuera 5?**

Si la ganancia del amplificador fuera 5 y suponiendo que el sensor se comporta de la misma forma ($10\text{mV}/^\circ$) entonces $5V$, que es lo máximo que acoge el ADC, equivaldría a que le llega al amplificador $1V$ que a su vez son 1000mV . Por lo que el rango de voltaje sería $0 - 1V$ lo que equivale a temperatura de 0 a $100^\circ C$.

2. **Expresión de la función de conversión de la temperatura al valor binario a la salida del conversor A/D.**

Dado un sensor que entrega $10\text{mV}/^\circ C$, una ganancia de amplificación $G = 10$, y un conversor A/D de 8 bits (con 256 niveles, de 0 a 255) y referencia de voltaje $V_{\text{ref}} = 5V$, se tiene:

$$\text{Valor digital} = \left\lfloor \frac{V_{\text{in}}}{V_{\text{ref}}} \cdot 256 \right\rfloor = \left\lfloor \frac{0,1 \cdot T}{5} \cdot 256 \right\rfloor = \lfloor 5,12 \cdot T \rfloor$$

3. **Valor mínimo de temperatura medible del sistema diseñado.**

El valor mínimo sería $0^\circ C$ que equivaldría a $0V$. No sería posible medir temperatura negativa.

4. **Valor de la ganancia del amplificador para que el valor binario a la salida del conversor A/D fuera directamente el valor de la temperatura.**

El valor de la ganancia debería ser tal que el número digital es igual a la temperatura (Valor digital = T). De esta forma, utilizando la relación anterior tenemos:

$$\text{Valor digital} = \left\lfloor \frac{0,01 \cdot T \cdot G}{5} \cdot 256 \right\rfloor$$

despejando obtenemos la ganancia:

$$G = \left\lfloor \frac{5}{256 \cdot 0,01} \right\rfloor = 1,953125$$

5. **Proponer una nueva configuración del circuito de activación/desactivación de la alarma (apartado 3.6 – actividad 5), en el que se sustituyan los comparadores lógicos por una etapa analógica basada en el uso de un comparador de Schmitt.**

En vez de utilizar otros dos comparadores de 4 bits como hemos utilizado nosotros y un biestable, es posible hacerlo de otra manera: los comparadores de 4 bits solo irán conectados por separado comparando únicamente los números más significativos del número binario $30^\circ C$ y $26^\circ C$ de forma que las salidas de estos comparadores irán conectadas al comparador de Schmitt. Cuando la entrada supera el umbral superior ($30^\circ C$ la salida pasa a nivel alto (1). Cuando la entrada cae por debajo del umbral inferior ($27^\circ C$ la salida pasa a nivel bajo (0). Entre ambos umbrales, la salida mantiene su estado anterior (esto es la histéresis).

10. Conclusión

A lo largo de este proyecto se hemos desarrollado un sistema completo de adquisición, procesamiento y visualización de temperatura, empleando distintos bloques electrónicos que permiten ilustrar los principios básicos de la instrumentación electrónica. Nuestro objetivo era que la alarma se encendiera para un valor superior a 30°C y se apagara al bajar de 27°C , sin embargo, por un *offset* los valores a los que se encienden son menores y lo sabemos gracias al voltímetro sabiendo el voltaje que sale del amplificador. Y el rango conseguido en el que la alarma está encendida es $25,5 - 28,5^{\circ}\text{C}$. Además, ocurre que al incluir el Arduino Nano hemos visto que la alarma se enciende en el rango $26,93 - 29,67^{\circ}\text{C}$ lo cuál, también tiene mucho sentido.

Este proyecto nos ha permitido iniciarnos en la electrónica y con ello: comprender un montón de conceptos, de errores posibles, solucionarlos a veces, aprender a manejar nuestra frustración, pelar cables, conectar cientos de ellos, colaborar con nuestros compañeros, diseñar circuitos, etc.

Finalizamos este proyecto agradeciendo a nuestra profesora Silvia por la paciencia depositada y por la transmisión de sus conocimientos. Y a nuestros compañeros que siempre nos ayudaron con cualquier cosa.

Referencias

- [1] *GUIÓN LABORATORIO DE ELECTRÓNICA*, GRADO FÍSICA, UV, 2025
- [2] *datasheet lm35*
- [3] *datasheet ua741*
- [4] *datasheet adc0804*
- [5] *datasheet sn74ls85*
- [6] *datasheet cd4043b*