



UNIVERSITAT POLITÈCNICA DE CATALUNYA

BARCELONATECH

Escola Superior d'Enginyeries Industrial,
Aeroespacial i Audiovisual de Terrassa



Informe práctica 2

Medida de temperatura con el sensor LM36

Curso 2019-2020 Cuatrimestre de primavera

Grado en ingeniería en tecnologías aeroespaciales

Grupo 11

Autores:

Alexis Leon Delgado
Juan Garrido Moreno

Profesor:

Juan Mon González

Índice

1. Introducción	3
2. Conexión del sistema hardware (tarea 2-1)	4
2.1. Conexionado básico para la lectura de la temperatura	4
2.2. Conexionado básico con voltímetros	4
2.3. Conexionado del LED	5
3. Calibración	6
3.1. Visualización del resultado de la función <i>analogRead(pinT)</i> (tarea 2-2)	6
3.2. Medición del valor V_{sensor} del LM36 para T_{amb} y $T_{\text{máx}}$ (tarea 2-3)	6
3.3. Medición de los valores V_A y V_B del divisor de tensiones y comentario sobre el ajuste del acondicionador mediante el potenciómetro (tarea 2-4)	7
3.4. Calibración del cero de salida mediante el potenciómetro del acondicionador	7
4. Adquisición de datos con el Arduino (tarea 2-6)	9
4.1. Código Arduino implementado	9
4.2. Estudio de funcionamiento	10
5. Simulación con respuesta del LED RGB (tarea 2-7)	12
5.1. Escala tricolor	12
5.1.1. Código implementado	12
5.1.2. Estudio de funcionamiento	13
5.2. Escala de colores secundarios	15
5.2.1. Código implementado	15
5.2.2. Estudio de funcionamiento	17
5.3. Escala de colores ampliada	20
5.3.1. Código implementado	20
5.3.2. Estudio de funcionamiento	23
6. Conclusiones	29

1. Introducción

El objetivo de esta práctica es analizar fundamentalmente el procesado, primeramente analógico, y a continuación de forma digital, de la señal medida mediante el sensor de temperatura LM36.

El proceso realizado ha consistido en una serie de **tareas prácticas** que se detallan a continuación:

Conexión del sistema hardware (tarea 2-1)

Se ha conectado el sensor al circuito acondicionador analógico, y su salida a la entrada analógica del Arduino. A esta configuración se le han añadido voltímetros complementarios.

Posteriormente, se ha añadido un LED RGB (junto a las pertinentes resistencias acondicionadoras) para obtener una respuesta visual de la medición. Por último, las conexiones del LED con la placa Arduino han sido reajustadas para la introducción del PWM. Estas modificaciones (a excepción de la última, que solamente se ha expuesto a nivel de código) pueden ser consultadas en la sección 2.

Calibración

En este caso las tareas han sido las siguientes:

- **Tarea 2-2:** Visualización por el monitor serie de la lectura no calibrada del sistema sensor-acondicionador y comprobación de la linealidad existente (ver apartado 3.1).
- **Tarea 2-3:** Medición del valor de la tensión de salida del tensor LM36 para la temperatura ambiental (25°C) y máxima (35°C) y comprobación de su disparidad respecto de la teoría (ver apartado 3.2).
- **Tarea 2-4:** Medición de las tensiones V_A y V_B del divisor de tensiones del potenciómetro y comprobación de su valor respecto de la teoría. También se comenta la posibilidad de realizar una calibración mediante el potenciómetro (ver apartado 3.3).
- **Tarea 2-5:** Calibración del cero de salida mediante el potenciómetro, con el objetivo de obtener $V_{\text{out}} = 2,5 \text{ V}$ para temperatura ambiente (ver apartado 3.4).

Adquisición de datos con Arduino (tarea 2-6)

En la sección 4 se ha adjuntado el código realizado en el estudio previo, a su vez que se han modificado los valores de N_{amb} y $N_{\text{máx}}$ de acuerdo con los resultados obtenidos en la calibración. Adicionalmente, se comprueba que, en efecto, dichos valores proporcionan unas correctas lecturas de temperatura mediante el *Serial Monitor*.

Simulación con respuesta del LED RGB (tarea 2-7)

En la sección 5 se han implementado un total de tres códigos, cada uno de ellos con una escala de color diferente. A modo de comprobación, se han adjuntado diversas capturas, corroborando así la adecuada iluminación del LED y el correcto valor mostrado por el *Serial Monitor*.

Hipervínculos al Tinkercad

En primer lugar, se presenta el montaje para la medición con escala tricolor. Se puede acceder a esta configuración del sistema haciendo clic [aquí](#) o en el siguiente link:

<https://www.tinkercad.com/things/8MDKiPt11Ax-clean-27-colors0/editel?sharecode=Zc0c9fEVDuPEtAGEnMuD5cRvHchWERZOC2IFnaS34nw>

En segundo lugar, se presenta el montaje para la medición con escala de colores primarios+secundarios. Se puede acceder a esta configuración del sistema haciendo clic [aquí](#) o en el siguiente link:

<https://www.tinkercad.com/things/lYkFUyjm7v2-clean-27-colors1-ledbien-/editel?sharecode=a0nYIxzd1hT3ubbjEFmcnPSTA85i0Wp7xBhaKyr4B6I>

En tercer lugar, se presenta el montaje para la medición con escala ampliada de colores obtenidos mediante PWM. Se puede acceder a esta configuración del sistema haciendo clic [aquí](#) o en el siguiente link:

https://www.tinkercad.com/things/lTaNk3Ov4EF-clean-27-colors2-ledbien-/editel?sharecode=ej5NeQtuF81_OtbumVrG2lFzE5lTrubGm_f8MEe4Uo

2. Conexión del sistema hardware (tarea 2-1)

2.1. Conexionado básico para la lectura de la temperatura

Con el objetivo de ofrecer un montaje más claro a nivel visual, se ha optado por usar un código de colores predeterminado: las conexiones asociadas al conjunto del potenciómetro y el amplificador U_1 han sido representadas en color amarillo, las conexiones asociadas al sensor LM36 y al amplificador U_2 han sido representadas en color turquesa y para las conexiones asociadas al amplificador U_3 se ha usado el color naranja.

En adición, las conexiones de alimentación a Vcc se muestran en color rojo, mientras que las conexiones a tierra se muestran en negro.

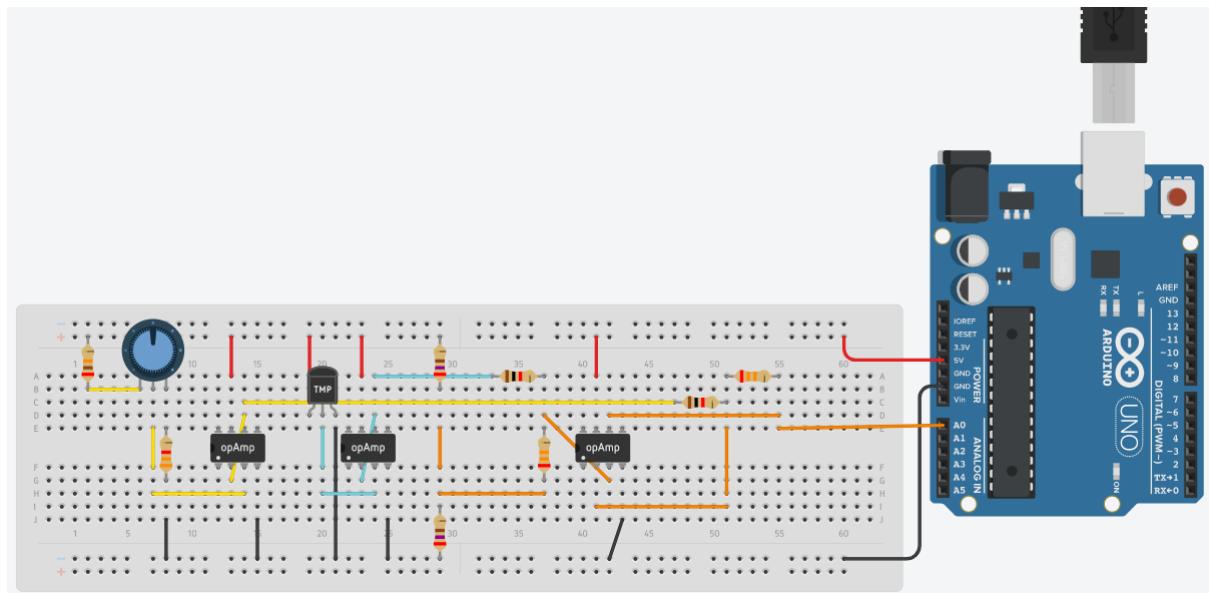


Figura 1: Conexión del sensor y su circuito acondicionador y de lectura.

2.2. Conexionado básico con voltímetros

Respecto de la configuración anterior se han añadido dos voltímetros para medir V_A y V_B , uno para medir la tensión de salida del sensor, y otro para medir la tensión de salida V_{out} de U_3 .

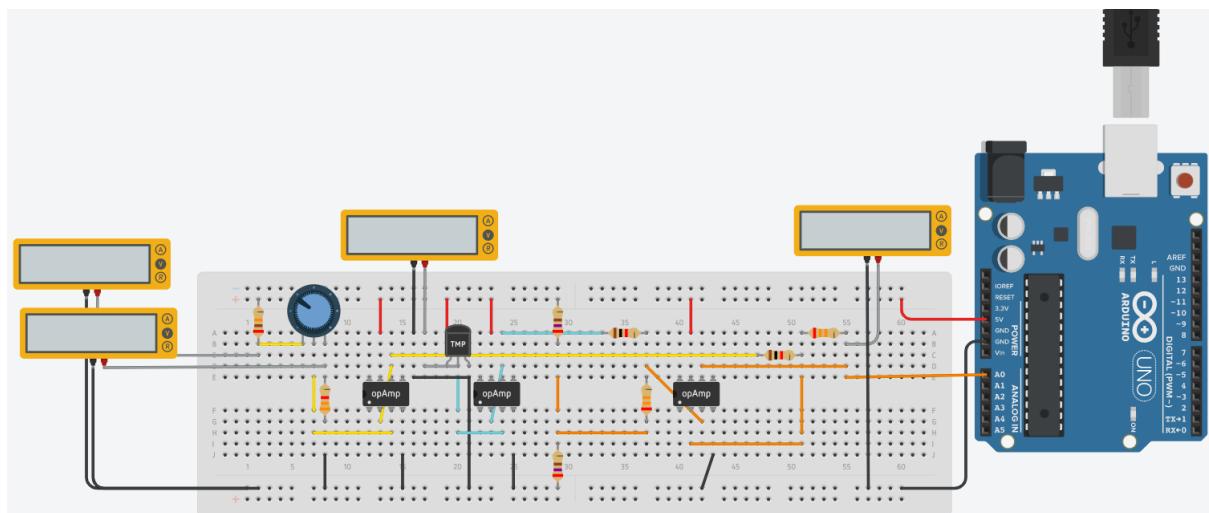


Figura 2: Conexión del sensor y su circuito acondicionador y de lectura con voltímetros de soporte.

2.3. Conexionado del LED

Para la tarea 2-7 será necesaria la conexión de un LED RGB. Consta de 4 entradas, una para cada color, más el cátodo, debido a que el LED es de cátodo común. Para su representación de forma ordenada y clara, los cables que abandonan la Figura 3 por la derecha son continuos a los que lo hacen por la parte superior de la misma Figura, evitando así errores de interpretación.

Obsérvese también como se han incluido las resistencias correspondientes a cada rama de color del LED. El objetivo de estas es proporcionar una caída de tensión en el LED igual a la nominal.

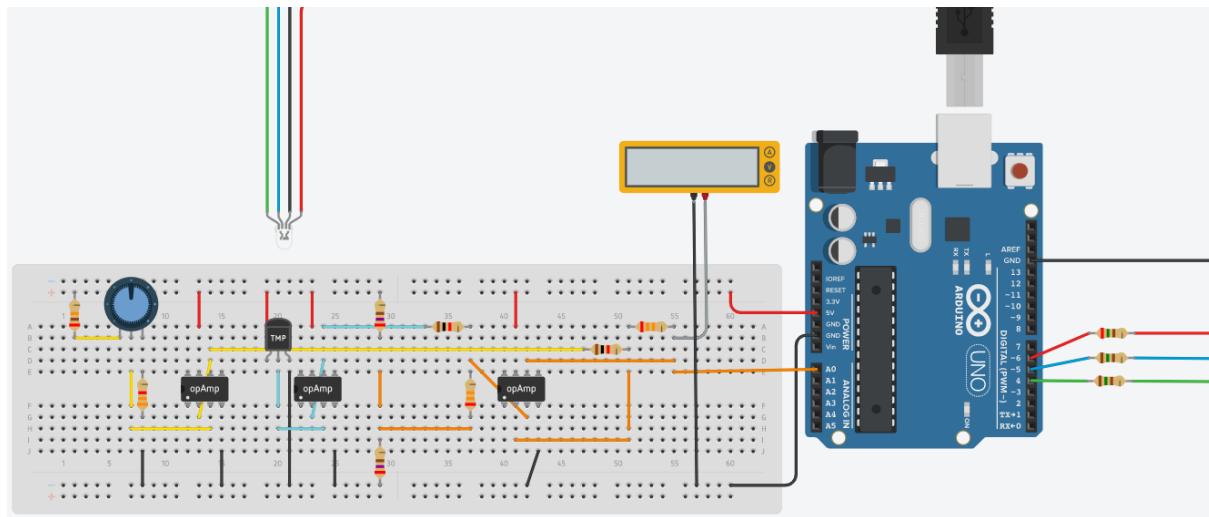


Figura 3: Conexión del sensor, su circuito acondicionador y de lectura y el LED RGB de respuesta.

Como se comentará más adelante, cuando se requiera el uso de PWM será necesario reajustar los puertos de alimentación del LED.

3. Calibración

3.1. Visualización del resultado de la función *analogRead(pinT)* (tarea 2-2)

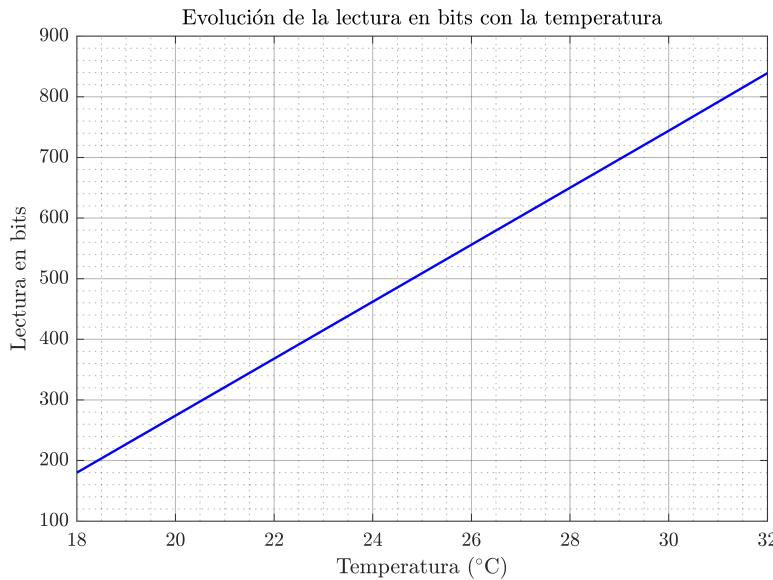


Figura 4: Gráfica de la progresión de la lectura en bits de la tensión de salida del amplificador U3.

En la Figura 4 se muestra la clara evolución lineal de la lectura del *pinT* a partir de la evolución controlada y regular de la temperatura. Concretamente, se han tomado un rango de temperaturas de 18 a 32°C con intervalos de temperatura de $\Delta T = 2^\circ\text{C}$. Para todo el tramo, la progresión es perfectamente lineal y de pendiente positiva y de valor 47,0417 bits/ $^\circ\text{C}$. No obstante, se debe tener en cuenta que esta representación, si bien es útil para observar la linealidad existente, no presenta unos resultados completamente correctos debido a que el potenciómetro no ha sido ajustado, y por tanto la salida del amplificador U3 tampoco es la correcta.

Un detalle importante a mencionar es referente al cursor que permite seleccionar la temperatura que lee el sensor LM36. Cuando se hace un *zoom* suficiente, para un mismo valor entero de temperatura, el cursor permite variar ligeramente entre dos posiciones, las cuales repercuten a nivel del voltaje de salida, y en consecuencia sobre la lectura del *pinT*. De estas dos posiciones, se ha tomado como buena la que proporciona una salida menor, que como se verá a continuación, es la más correcta.

3.2. Medición del valor V_{sensor} del LM36 para T_{amb} y $T_{\text{máx}}$ (tarea 2-3)

Con la finalidad de llevar a cabo la medición de la diferencia de potencial en bornes del sensor de temperatura LM36, se hace uso de un multímetro en modo voltímetro proporcionado por Tinkercad. Teniendo en cuenta que las temperaturas ambiente y máxima son 25°C y 35°C, respectivamente, se obtienen los siguientes valores:

$$V_{\text{sensor}}(T = T_{\text{amb}}) = 749 \text{ mV}$$

$$V_{\text{sensor}}(T = T_{\text{máx}}) = 849 \text{ mV}$$

Por tanto, teniendo en cuenta que la sensibilidad del sensor LM36 es de $s = 10 \text{ mV}/^\circ\text{C}$, se pueden obtener los valores de temperatura a partir de las mediciones realizadas.

$$T_{\text{amb}}|_{\text{experimental}} = \frac{1}{s} [V_{\text{sensor}}|_{\text{exp}} - V_{\text{sensor}}(T_{\text{amb}})] + T_{\text{amb}} = \frac{1}{10} (749 - 750) + 25 = 24,9^\circ\text{C}$$

$$T_{\text{máx}}|_{\text{experimental}} = \frac{1}{s} [V_{\text{sensor}}|_{\text{exp}} - V_{\text{sensor}}(T_{\text{amb}})] + T_{\text{amb}} = \frac{1}{10} (849 - 750) + 25 = 34,9^\circ\text{C}$$

Es evidente que existe cierto error respecto los valores teóricos, no obstante, dicha discrepancia es menor a 0,4 %.

3.3. Medición de los valores V_A y V_B del divisor de tensiones y comentario sobre el ajuste del acondicionador mediante el potenciómetro (tarea 2-4)

El voltaje V_A se corresponde al valor medido en el nodo situado entre el potenciómetro y la resistencia R_1 y su valor debe ser la tensión de salida del sensor para la temperatura máxima, es decir $V_A = 0,85$ V.

Por otro lado, V_B es la diferencia de potencial medida entre el nodo inferior al potenciómetro y la resistencia R_2 y su valor de tensión en la salida del sensor para la temperatura mínima $V_B = 0,65$ V.

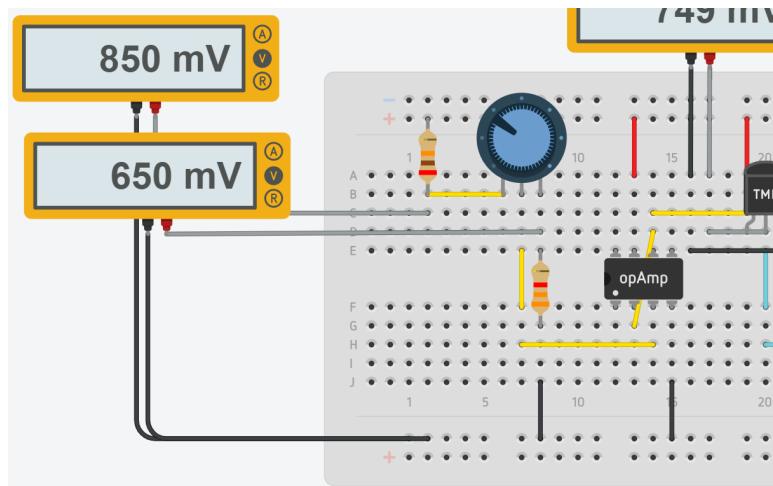


Figura 5: Imagen donde se visualizan las lecturas de V_A (multímetro superior) y V_B (multímetro inferior).

En la Figura 5 se ilustra que las lecturas obtenidas en el Tinkercad, mostrando valores exactamente idénticos a la teoría.

Mediante la variación de la posición del dial del potenciómetro será posible hallar una tensión de salida del amplificador U_1 igual a la tensión del sensor para temperatura ambiente.

3.4. Calibración del cero de salida mediante el potenciómetro del acondicionador

El proceso de ajuste se realiza heurísticamente, de manera que se va incrementando progresivamente la posición del dial del potenciómetro hasta conseguir el voltaje V_{out} deseado. Así mismo, haciendo uso de un voltímetro colocado a la salida del amplificador operacional U_3 se comprueba el valor de la mencionada diferencia de potencial. En este caso, se ha alcanzado la condición de $V_{out} = 2,47$ V para la posición del dial mostrada a continuación:

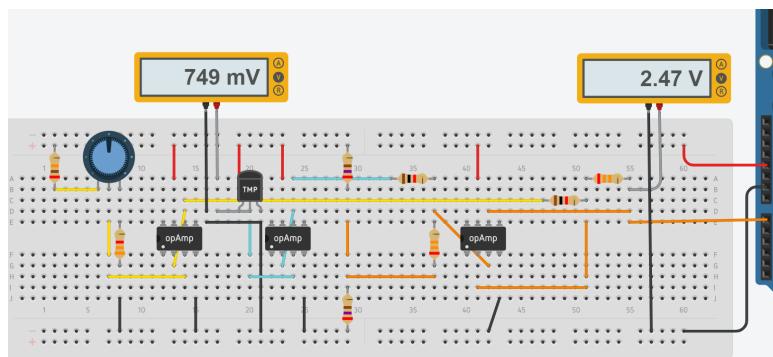


Figura 6: Comprobación del voltaje a la salida del amplificador operacional durante el proceso de calibración

Cabe mencionar que este valor de ajuste es el mejor que se ha podido obtener, siendo ligeramente inferior al deseado con un error relativo correspondiente de un 1,2 %. Este hecho es básicamente debido a la presencia de diversos voltímetros en

el circuito, cuyas elevadas pero no idealmente infinitas resistencias causan la obtención de un voltaje distinto al calculado teóricamente. Cabe remarcar que, en caso de retirar el voltímetro ubicado en bornes del sensor, la tensión V_{out} asciende a 2,49 V, acercándose más a su valor teóricamente correcto. Por este motivo, dicho voltímetro ha sido retirado en las siguientes tareas.

En lo que a la justificación del valor de V_{out} obtenido se refiere, se conoce que la calibración se ha llevado a cabo con la finalidad de situar el cero del sistema en la temperatura ambiente (25°C), valor central entre la temperatura mínima (15°C) y la máxima (35°C). Además, se sabe que la alimentación del amplificador operacional es de 5 V, por lo que el rango de salida será $\pm 2,5$ V respecto el valor de referencia de 2,5 V. Es por este motivo que el cero no puede situarse en 0 V, lo que supondría unos valores negativos para temperaturas $T_{\min} < T < T_{\text{amb}}$, que lógicamente quedarían fuera del rango de la entrada analógica del Arduino (0 – 5 V). Cabe mencionar que, a diferencia del amplificador empleado en Tinkercad, en la realidad el uA741 no es *rail-to-rail*, siendo los voltajes de saturación de salida de 0,2 V y 4,8 V. No obstante, el cero del sistema seguiría situándose en 2,5 V, por lo que no existiría diferencia alguna respecto el caso teórico.

En cuanto al V_{off} , de acuerdo con la calibración llevada a cabo mediante el potenciómetro, se debe cumplir que el voltaje obtenido entre la pata *wiper* del mismo y GND sea el equivalente a V_{sensor} a temperatura ambiente (25°C). A continuación se adjunta una captura del proceso de medición de V_{off} :

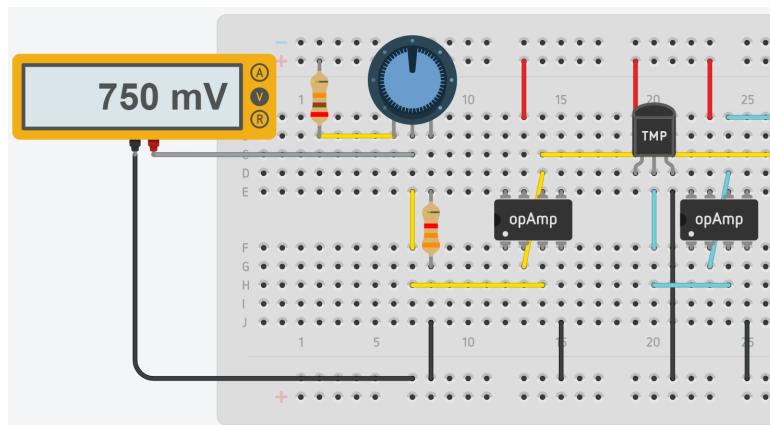


Figura 7: Comprobación del voltaje V_{off} del potenciómetro ya calibrado

Se visualiza que el voltaje de *offset* obtenido es de 750 mV, valor prácticamente idéntico al obtenido en el apartado 3.2, de valor $V_{\text{sensor}}(T = T_{\text{amb}}) = 749$ mV. Es así que se comprueba que el procedimiento de calibración se ha realizado, en efecto, correctamente.

En el siguiente apartado se implementará un código para obtener por el monitor serie el valor real de la temperatura medida. Para que la calibración sea exacta, se requieren los resultados de la función *analogRead(pinT)* para las temperaturas ambiental y máxima, siendo estas lecturas N_{amb} y $N_{\text{máx}}$, respectivamente:

$$N_{\text{amb}} = 509 \quad N_{\text{máx}} = 980$$

Cabe mencionar que en el estudio previo se obtuvieron unos valores de $N_{\text{amb}} = 517$ y $N_{\text{máx}} = 982$ para unos voltajes de salida del uA741 de 2,5 V y 4,8 V, respectivamente. No obstante, la ligera diferencia respecto los valores obtenidos mediante Tinkercad se justifican dada la ya mencionada característica *rail-to-rail* del amplificador en la realidad, en contraposición con la salida no saturada proporcionada por el Tinkercad.

4. Adquisición de datos con el Arduino (tarea 2-6)

4.1. Código Arduino implementado

En base a los valores de N_{amb} y $N_{\text{máx}}$, así como en la posición calibrada del potenciómetro, hallados en la sección anterior, se adaptará el código del estudio previo. Se recuerda que el cálculo de la temperatura medida se realiza mediante una linealización de pendiente m_{temp} , cuyo cálculo se adjunta a continuación:

$$m_{\text{temp}} = (T_{\text{max}} - T_{\text{amb}})/(N_{\text{max}} - N_{\text{amb}})$$

Una ligera variación introducida ha consistido en la eliminación dicho cálculo del interior del *loop* para ubicarlo en la definición de constantes. De esta forma, se evita que un cálculo que depende exclusivamente de constantes se repita continuamente.

A partir de la pendiente hallada, el cálculo de la temperatura aplica la linealización comentada:

$$\text{Temp} = T_{\text{amb}} + m_{\text{temp}} \cdot (\text{lectura_temperatura} - N_{\text{amb}})$$

Se puede observar como, en caso de que la temperatura medida sea igual a la ambiente, el interior del paréntesis será nulo y solamente quedará el primer sumando, el cual es igual a la temperatura ambiente.

El código implementado para procesar la lectura del sistema acondicionador, y a continuación imprimir su resultado por el monitor serie, se adjunta a continuación:

```

1 // Definición de pines analógicos
2 const int pinT = A0;
3 // Definición de temperaturas
4 const double Tamb = 25;
5 const double Tmax = 35;
6 // Lectura asociada a 25°C
7 const int Namb = 509;
8 // Lectura asociada a 35°C
9 const int Nmax = 980;
10 // Variable de la lectura de V_out
11 volatile int lectura_temperatura=0;
12 // Variable de la temperatura medida
13 float Temp=0.0;
14 // Pendiente de la ecuación de temperatura
15 double pendiente_temp=(double)(Tmax-Tamb)/(double)(Nmax-Namb);
16
17 void setup() {
18     // Establecimiento de la velocidad de comunicación
19     Serial.begin(9600);
20 }
21
22 void loop() {
23     // Lectura de la tensión de salida de U3
24     lectura_temperatura=analogRead(pinT);
25     // Cálculo de la Temperatura medida
26     Temp=Tamb+pendiente_temp*(lectura_temperatura-Namb);
27     // Impresión de resultados
28     Serial.print("Lectura temperatura=");
29     Serial.println(lectura_temperatura); Serial.println();
30     Serial.print("Temperatura");
31     Serial.println();
32     Serial.println(Temp);
33     // Se aplica un retraso de 500 ms entre medición y medición
34     delay(500);
35 }
```

4.2. Estudio de funcionamiento

En primer lugar, se ha elegido una temperatura de 25°C y la posición del cursor era la “mínima” o la de la izquierda. Como se muestra en la Figura 8, el valor de la variable `lectura_temperatura` es exactamente igual a N_{amb} . En consecuencia, la temperatura medida es 25°C y la medición es correcta.

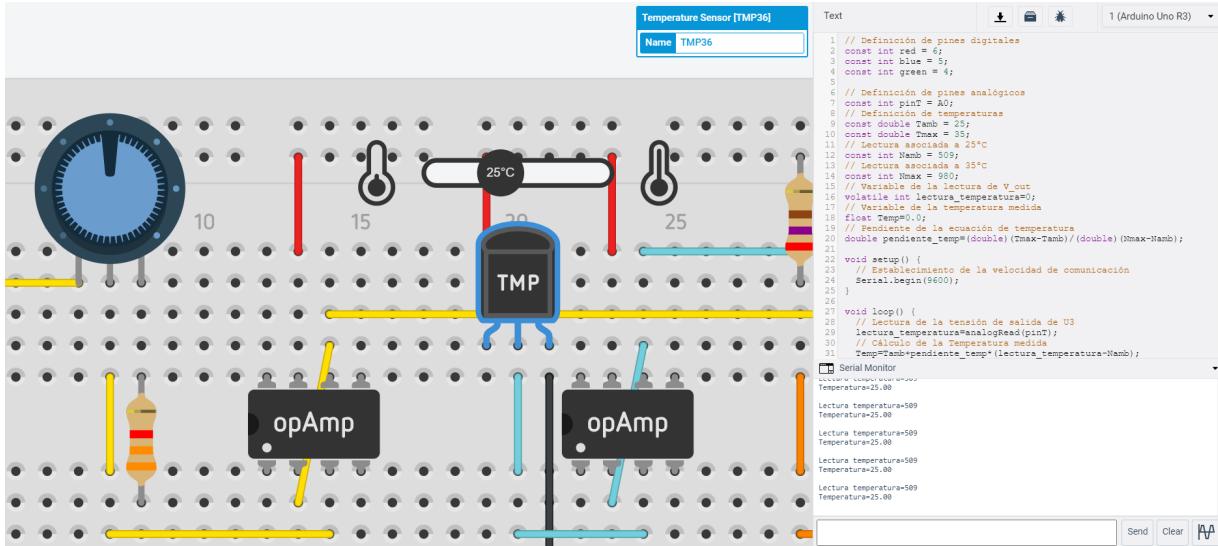


Figura 8: Comprobación de la medida para 25°C.

A continuación, se impone una temperatura de 35°C, y la lectura del monitor serie puede comprobarse en la Figura 9. Ahora el valor de lectura_temperatura coincide exactamente con $N_{\text{máx}}$, y por esta razón la temperatura de salida son 35°C, de forma correcta.

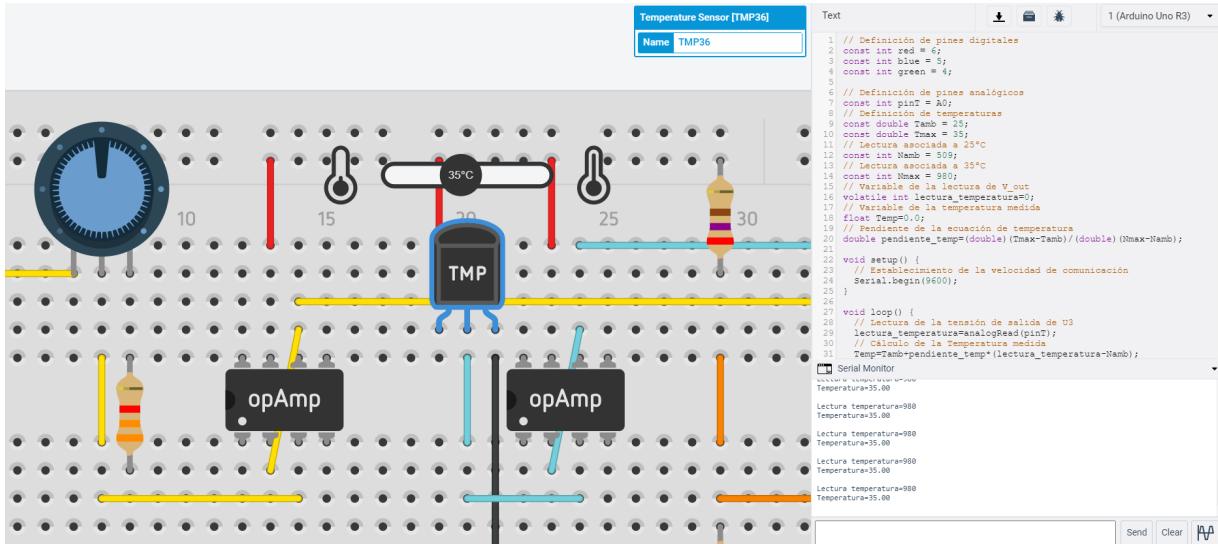


Figura 9: Comprobación de la medida para 35°C.

Ajustando una temperatura intermedia de 30°C en el sensor, el monitor serie muestra una temperatura medida de 29, 99°C. La desviación es negligible entre ambos valores y se puede considerar que la calibración funciona correctamente para este caso.

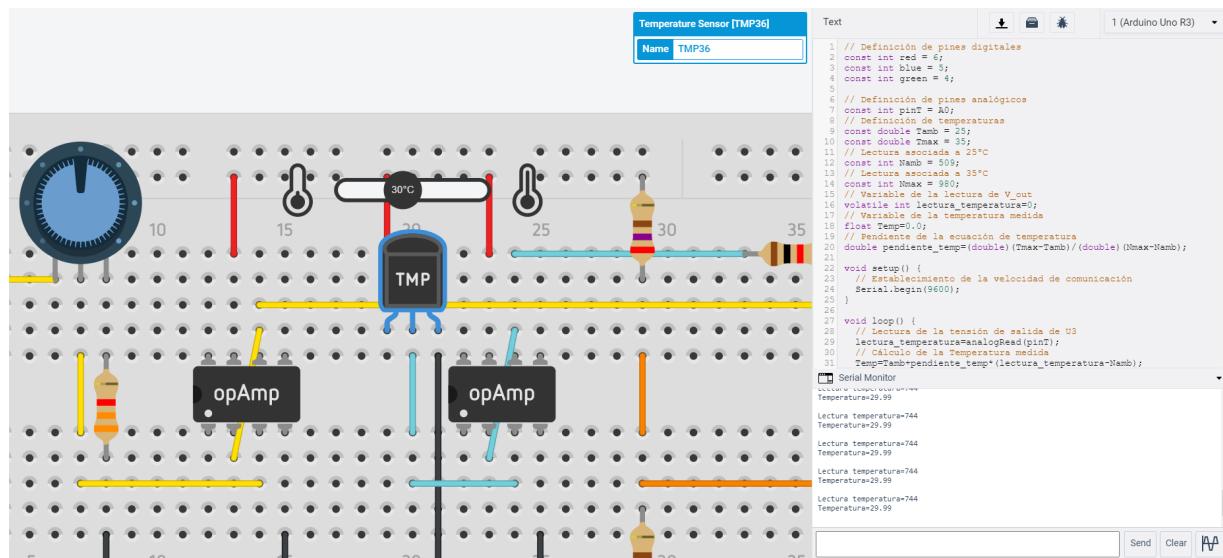


Figura 10: Comprobación de la medida para 30°C.

Finalmente se ha tomado una temperatura por debajo de la ambiente, concretamente de 15°C, y se ha observado que el valor impreso por pantalla son 15,02°C, tal y como se puede comprobar en la Figura 11. Nuevamente se puede afirmar que la calibración, así como los resultados impresos por el monitor serie, son correctos.

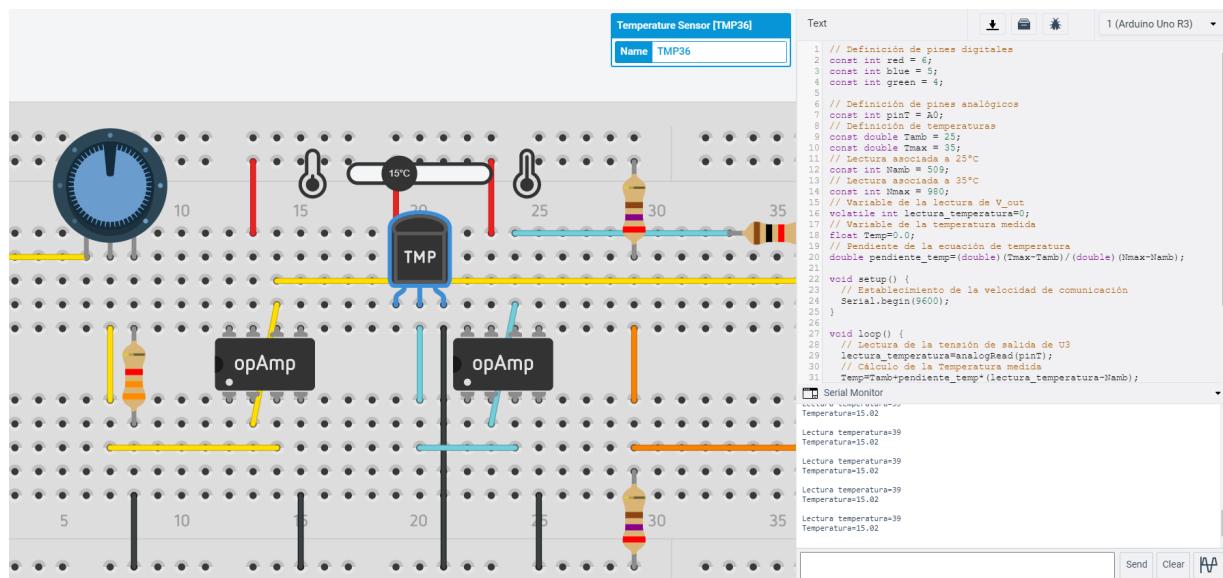


Figura 11: Comprobación de la medida para 15°C.

5. Simulación con respuesta del LED RGB (tarea 2-7)

En la tarea 2-6 se comprobó que la lectura de la temperatura a través del monitor serie era la correcta, pero ahora se incluirá un LED RGB que indicará de forma visual y clara el rango de temperaturas en el que se encuentra la medición.

5.1. Escala tricolor

En primer lugar, el LED se iluminará en **verde** si la temperatura está contenida entre 25°C y 35°C , se iluminará en **rojo** si la temperatura que mide el sensor excede el límite superior, o se iluminará en color **azul** si la temperatura está por debajo del límite inferior. En adición, y siguiendo el código del estudio previo, los límites superior e inferior han sido ligeramente recortados.

5.1.1. Código implementado

Los cambios principales aplicados sobre el código Arduino han sido: la definición de pines digitales de alimentación del LED, la definición del modo de trabajo de estos pines como salida (OUTPUT) en el *setup*, y finalmente la inclusión de un condicional que distingue los diferentes tramos de temperatura y enciende el color correspondiente mediante la función *digitalwrite*.

El código implementado se adjunta a continuación:

```

1 // Definición de pines digitales de alimentación del LED
2 const int red = 6;
3 const int blue = 5;
4 const int green = 4;
5
6 // Definición de pines analógicos
7 const int pinT = A0;
8 // Definición de temperaturas
9 const double Tamb = 25;
10 const double Tmax = 35;
11 // Lectura asociada a 25°C
12 const int Namb = 509;
13 // Lectura asociada a 35°C
14 const int Nmax = 980;
15 // Variable de la lectura de V_out
16 volatile int lectura_temperatura=0;
17 // Variable de la temperatura medida
18 float Temp=0.0;
19 // Pendiente de la ecuación de temperatura
20 double pendiente_temp=(double)(Tmax-Tamb)/(double)(Nmax-Namb);
21
22 void setup() {
23   // Establecimiento de la velocidad de comunicación
24   Serial.begin(9600);
25   // Definición del modo de trabajo de los pines digitales
26   pinMode(red, OUTPUT);
27   pinMode(green, OUTPUT);
28   pinMode(blue, OUTPUT);
29 }
30
31 void loop() {
32   // Lectura de la tensión de salida de U3
33   lectura_temperatura=analogRead(pinT);
34   // Cálculo de la Temperatura medida
35   Temp=Tamb+pendiente_temp*(lectura_temperatura-Namb);
36   // Impresión de resultados
37   Serial.print("Lectura temperatura=");
38   Serial.println(lectura_temperatura);
39   Serial.print("Temperatura=");
40   Serial.println(Temp); Serial.println();
41
42   // Iluminación de LEDs en función de la temperatura
43   // Se enciende el LED en rojo si la temperatura supera el límite superior
44   if (Temp > Tmax-0.5)
45   {

```

```

46     digitalWrite(red,HIGH);
47     digitalWrite(green,LOW);
48     digitalWrite(blue,LOW);
49 }
50 // Se enciende el LED en azul si la temperatura no alcanza el límite inferior
51 else if (Temp < Tamb+0.5)
52 {
53     digitalWrite(red,LOW);
54     digitalWrite(green,LOW);
55     digitalWrite(blue,HIGH);
56 }
57 // Se enciende el LED en verde si la temperatura está dentro del rango válido
58 else
59 {
60     digitalWrite(red,LOW);
61     digitalWrite(green,HIGH);
62     digitalWrite(blue,LOW);
63 // Se aplica un retraso de 500 ms entre medición y medición
64 delay(500);
65 }

```

5.1.2. Estudio de funcionamiento

En primer lugar, tomando una temperatura comprendida en el rango válido, como pueden ser 30°C, el LED se iluminará en verde, tal y como muestra la Figura 12.

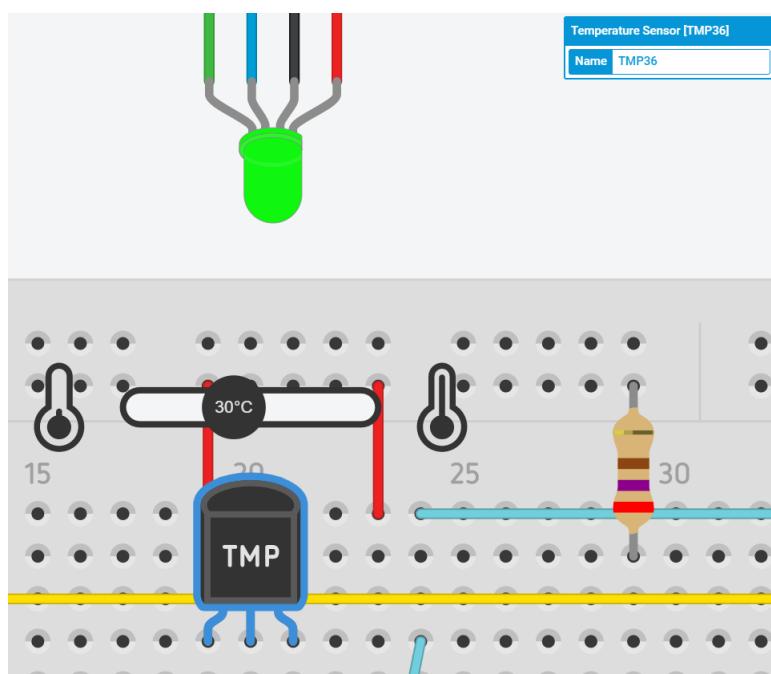


Figura 12: Respuesta del LED RGB para 30°C.

En cambio, si se toma una temperatura de 50°C , la cual excede el límite superior, el LED se encenderá en rojo de acuerdo a la Figura 13.

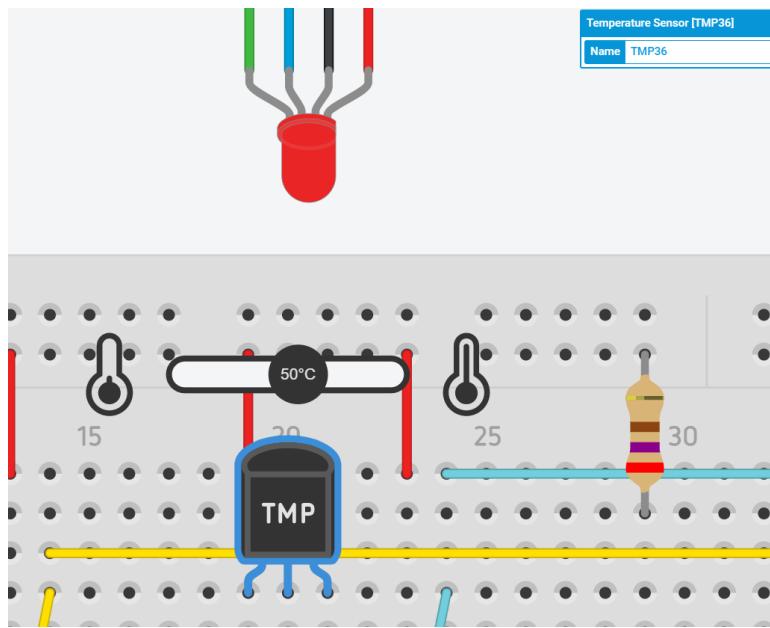


Figura 13: Respuesta del LED RGB para 50°C .

Por último, situando el cursor a 10°C , valor inferior a la temperatura ambiente, el LED se iluminará en color azul como muestra la Figura 14.

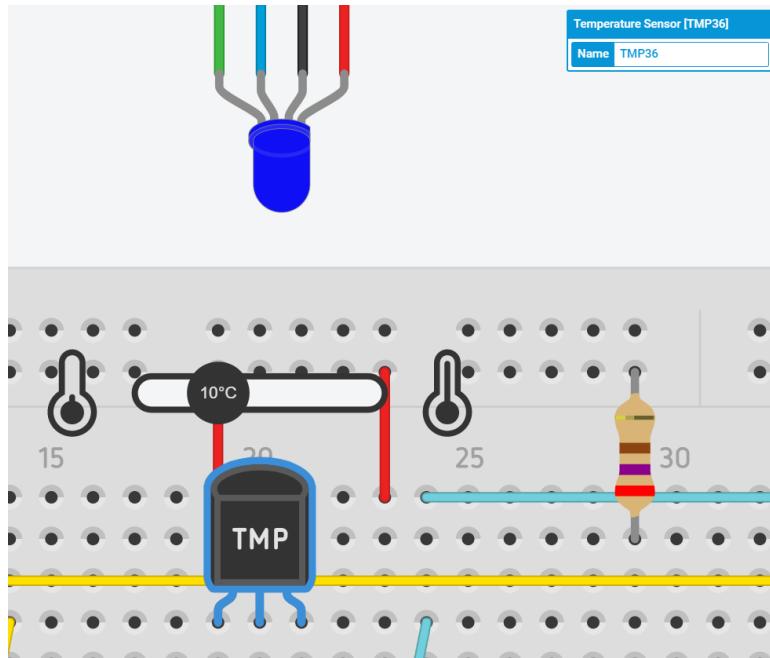


Figura 14: Respuesta del LED RGB para 10°C .

5.2. Escala de colores secundarios

A continuación, el rango válido de temperaturas será subdividido en tramos, y los colores disponibles para estos serán los primarios: rojo, azul y verde, más los colores secundarios, los cuales son combinación de dos de los colores primarios.

Concretamente, la escala de colores elegida ha sido la siguiente:



Por tanto, el rango de temperaturas empleado es el que se muestra en la siguiente tabla:

Color de iluminación	Rango de temperaturas	Límite inferior (°C)	Límite superior (°C)
Azul	$T < T_{amb} + 10/6$	-	26,67
Violado (azul+rojo)	$T_{amb} + 10/6 \leq T < T_{amb} + 2 \cdot 10/6$	26,67	28,33
Cian (azul+verde)	$T_{amb} + 2 \cdot 10/6 \leq T < T_{amb} + 3 \cdot 10/6$	28,33	30
Verde	$T_{amb} + 3 \cdot 10/6 \leq T < T_{amb} + 4 \cdot 10/6$	30	31,67
Amarillo (rojo+verde)	$T_{amb} + 4 \cdot 10/6 \leq T < T_{amb} + 5 \cdot 10/6$	31,67	33,33
Rojo	$T_{amb} + 5 \cdot 10/6 \leq T$	33,33	-

Tabla 1: Escala de colores primarios y secundarios.

5.2.1. Código implementado

La única diferencia respecto del código anterior es al ampliación del condicional presente en el *loop*, el cual ahora hace una distinción más fina del rango de temperaturas para así encender el LED del color correspondiente.

```

1 // Definición de pines digitales
2 const int red = 6;
3 const int blue = 5;
4 const int green = 4;
5
6 // Definición de pines analógicos
7 const int pinT = A0;
8 // Definición de temperaturas
9 const double Tamb = 25;
10 const double Tmax = 35;
11 // Lectura asociada a 25°C
12 const int Namb = 509;
13 // Lectura asociada a 35°C
14 const int Nmax = 980;
15 // Variable de la lectura de V_out
16 volatile int lectura_temperatura=0;
17 // Variable de la temperatura medida
18 float Temp=0.0;
19 // Pendiente de la ecuación de temperatura
20 double pendiente_temp=(double)(Tmax-Tamb)/(double)(Nmax-Namb);
21
22 void setup() {
23     // Establecimiento de la velocidad de comunicación
24     Serial.begin(9600);
25     // Definición del modo de trabajo de los pines digitales
26     pinMode(red, OUTPUT);
27     pinMode(green, OUTPUT);
28     pinMode(blue, OUTPUT);
29 }
30
31 void loop() {
32     // Lectura de la tensión de salida de U3
33     lectura_temperatura=analogRead(pinT);
34     // Cálculo de la Temperatura medida
35     Temp=Tamb+pendiente_temp*(lectura_temperatura-Namb);
36     // Impresión de resultados

```

```
37 Serial.print("Lectura temperatura=");  
38 Serial.println(lectura_temperatura);  
39 Serial.print("Temperatura=");  
40 Serial.println(Temp); Serial.println();  
41  
42 // Iluminación de LEDs en función de la temperatura  
43 // Se enciende el LED AZUL si la temperatura se encuentra dentro del rango válido  
44 if (Temp < Tamb+double(10.0/6.0))  
45 {  
46     digitalWrite(red,LOW);  
47     digitalWrite(green,LOW);  
48     digitalWrite(blue,HIGH);  
49 }  
50 // Se enciende el LED VIOLADO si la temperatura se encuentra dentro del rango válido  
51 else if (Temp >= Tamb+double(10.0/6.0) && Temp < Tamb+double(2*10.0/6.0))  
52 {  
53     digitalWrite(red,HIGH);  
54     digitalWrite(green,LOW);  
55     digitalWrite(blue,HIGH);  
56 }  
57 // Se enciende el LED CIAN si la temperatura se encuentra dentro del rango válido  
58 else if (Temp >= Tamb+double(2*10.0/6.0) && Temp < Tamb+double(3*10.0/6.0))  
59 {  
60     digitalWrite(red,LOW);  
61     digitalWrite(green,HIGH);  
62     digitalWrite(blue,HIGH);  
63 }  
64 // Se enciende el LED VERDE si la temperatura se encuentra dentro del rango válido  
65 else if (Temp >= Tamb+double(3*10.0/6.0) && Temp < Tamb+double(4*10.0/6.0))  
66 {  
67     digitalWrite(red,LOW);  
68     digitalWrite(green,HIGH);  
69     digitalWrite(blue,LOW);  
70 }  
71 // Se enciende el LED AMARILLO si la temperatura se encuentra dentro del rango válido  
72 else if (Temp >= Tamb+double(4*10.0/6.0) && Temp < Tamb+double(5*10.0/6.0))  
73 {  
74     digitalWrite(red,HIGH);  
75     digitalWrite(green,HIGH);  
76     digitalWrite(blue,LOW);  
77 }  
78 // Se enciende el LED ROJO si la temperatura se encuentra dentro del rango válido  
79 else if (Temp >= Tamb+double(5*10.0/6.0))  
80 {  
81     digitalWrite(red,HIGH);  
82     digitalWrite(green,LOW);  
83     digitalWrite(blue,LOW);  
84 }  
85 // Se aplica un retraso de 500 ms entre medición y medición  
86 delay(500);  
87 }
```

5.2.2. Estudio de funcionamiento

En primer lugar, seleccionando una temperatura de 26°C la cual se halla en el primer tramo definido, cuya lectura asociada en el *Serial.monitor* es $26,00^{\circ}\text{C}$, el color del LED es azul como muestra la Figura 15.

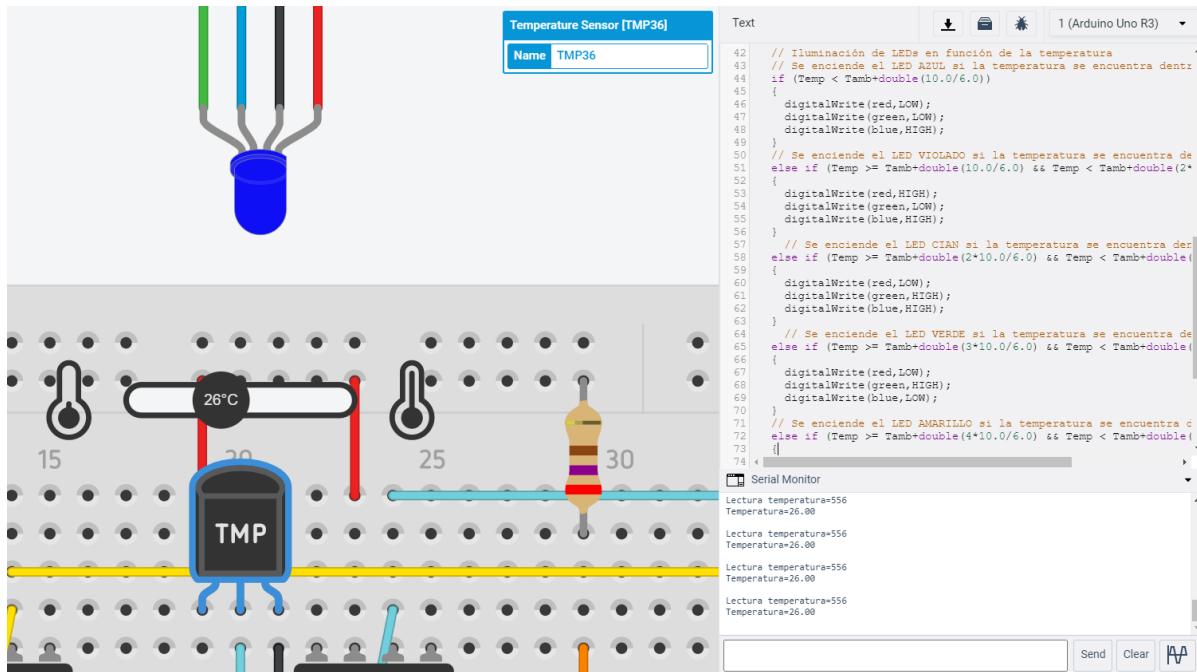


Figura 15: Respuesta del LED RGB para 26°C y escala de colores secundarios.

A continuación, ascendiendo la temperatura a 28°C , se obtiene una medición digital de $27,99^{\circ}\text{C}$. Este resultado se halla en el segundo tramo, motivo por el cual el LED adquiere un color violado, exactamente como se ilustra en la Figura 16.

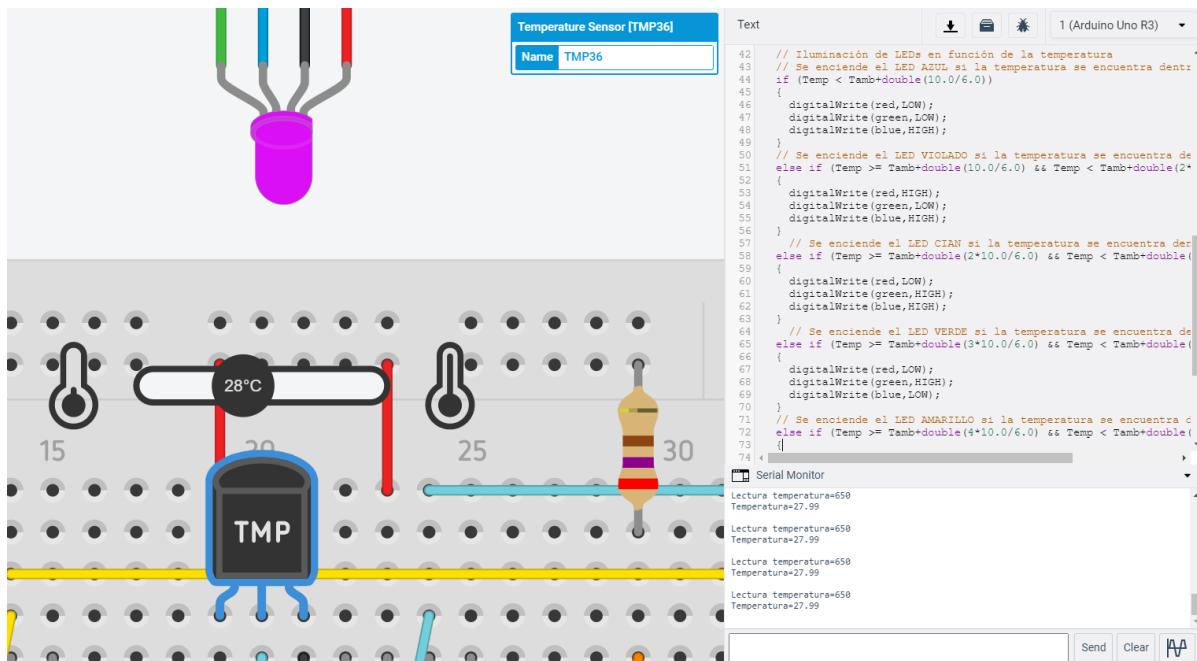


Figura 16: Respuesta del LED RGB para 28°C y escala de colores secundarios.

Posteriormente la temperatura se sitúa a 29°C, obteniendo así 28,99°C por el *Serial Monitor*, la cual se halla en el tercer tramo. En consecuencia, el color de salida es cian, como se puede comprobar en la Figura 17.

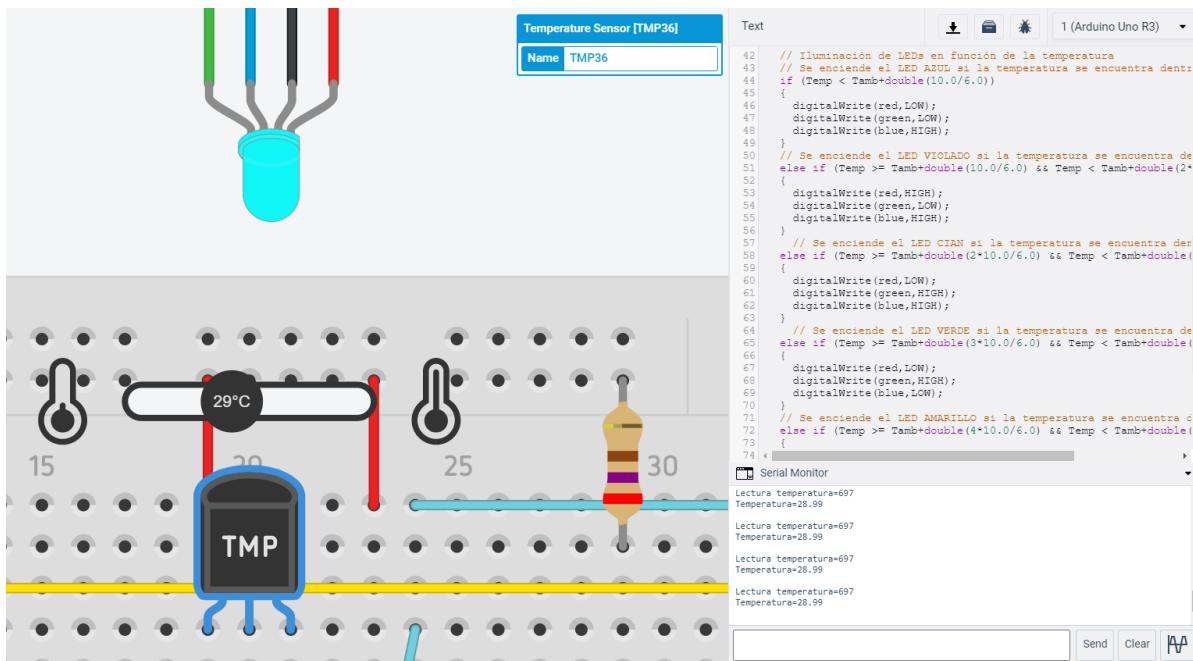


Figura 17: Respuesta del LED RGB para 29°C y escala de colores secundarios.

Para avanzar al siguiente tramo se toma una temperatura de 31°C, que arroja un valor de 31,01°C por el *Serial Monitor*. Siguiendo la tendencia anterior, el LED varía su color a verde, como se muestra en la Figura 18.

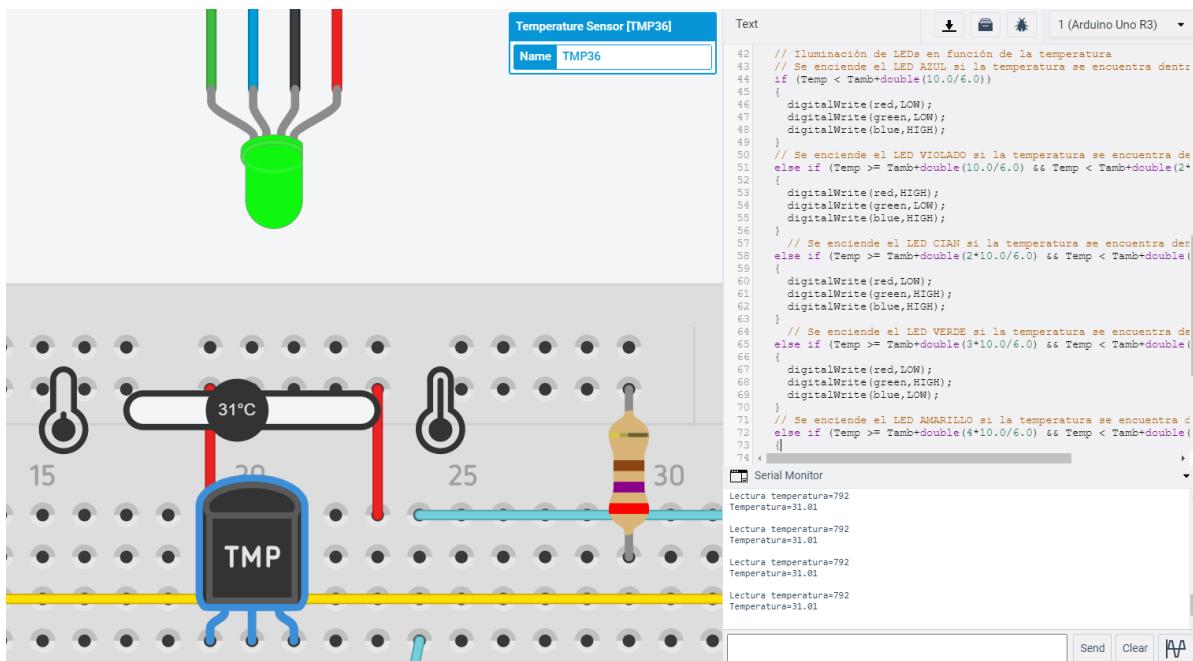


Figura 18: Respuesta del LED RGB para 31°C y escala de colores secundarios.

Seleccionando ahora una temperatura de 32°C, el color del LED cambia correspondientemente al amarillo dado el valor de 32,01°C mostrado por el *Serial Monitor*. Esto se puede comprobar en la Figura 19.

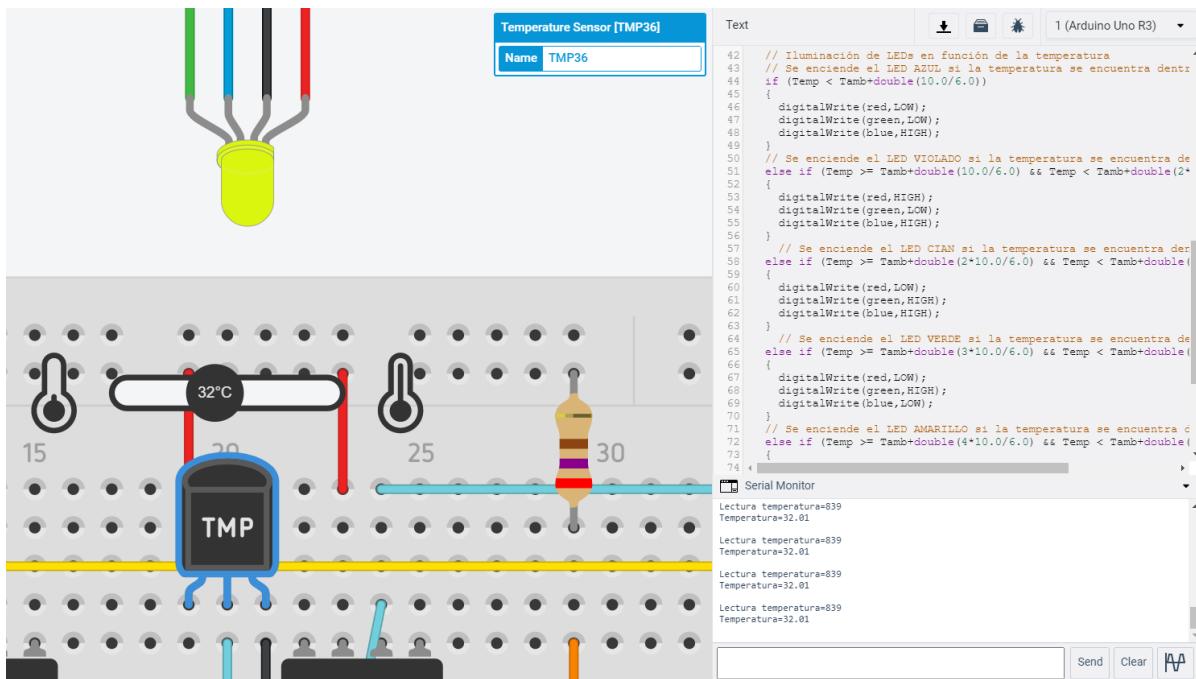


Figura 19: Respuesta del LED RGB para 32°C y escala de colores secundarios.

Por último, se desplaza el cursor hasta los 34°C, siendo el valor obtenido por el *Serial Monitor* de 34,00°C, temperatura que se halla en el último tramo. En la Figura 20 se comprueba como la iluminación cambia al color rojo.

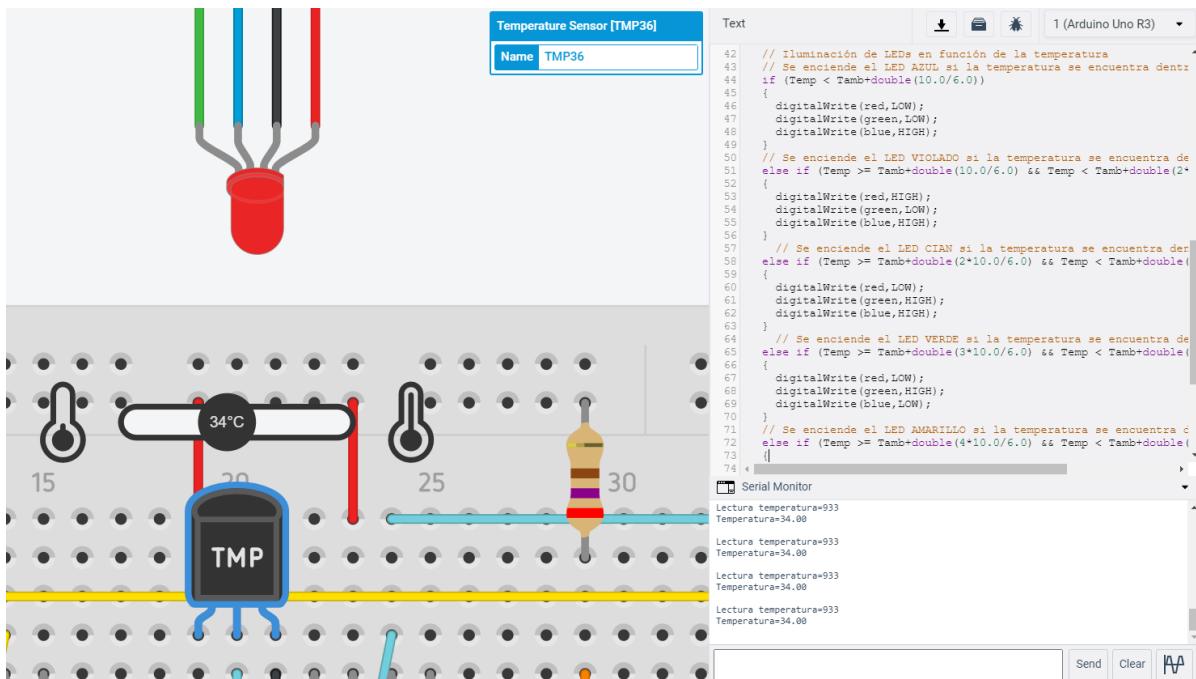


Figura 20: Respuesta del LED RGB para 34°C y escala de colores secundarios.

5.3. Escala de colores ampliada

Con la finalidad de obtener un abanico de colores más amplio en la escala RGB, se ha alimentado el LED con señales analógicas moduladas en PWM. Por este motivo, se ha hecho uso de la función *analogWrite*, en contraposición con el *digitalWrite* empleado anteriormente. Así mismo, considerando un total de 11 colores posibles, se han determinado los respectivos rangos de temperaturas para cada color. En este caso, conociendo que $T_{amb} = 25^{\circ}\text{C}$ y $T_{máx} = 35^{\circ}\text{C}$, los incrementos de temperatura son de $10/11 = 0,91^{\circ}\text{C}$.

A modo de resumen, en la siguiente tabla se adjuntan los rangos de temperaturas y los respectos colores:

Rango de temperaturas ($^{\circ}\text{C}$)	$T < T_{amb} + \frac{10}{11}$	$T_{amb} + \frac{10}{11} \leq T < T_{amb} + 2\frac{10}{11}$	$T_{amb} + 2\frac{10}{11} \leq T < T_{amb} + 3\frac{10}{11}$	$T_{amb} + 3\frac{10}{11} \leq T < T_{amb} + 4\frac{10}{11}$	$T_{amb} + 4\frac{10}{11} \leq T < T_{amb} + 5\frac{10}{11}$	$T_{amb} + 5\frac{10}{11} \leq T < T_{amb} + 6\frac{10}{11}$	$T_{amb} + 6\frac{10}{11} \leq T < T_{amb} + 7\frac{10}{11}$	$T_{amb} + 7\frac{10}{11} \leq T < T_{amb} + 8\frac{10}{11}$	$T_{amb} + 8\frac{10}{11} \leq T < T_{amb} + 9\frac{10}{11}$	$T_{amb} + 9\frac{10}{11} \leq T < T_{amb} + 10\frac{10}{11}$	$T \geq T_{amb} + 10\frac{10}{11}$	
Color												
R	255	195	98	49	0	0	0	0	128	255	255	255
G	0	0	0	0	0	158	255	255	255	255	64	64
B	225	255	255	255	255	255	195	37	0	0	0	0
Matiz	218	202	186	178	170	144	117	91	64	42	0	0
Saturación	255											
Luminosidad	128											
		Añil			Azul			Verde		Amarillo		

Tabla 2: Colores RGB empleados con sus correspondientes rangos de temperatura

Cabe mencionar que los colores mostrados en Tinkercad difieren ligeramente en algunos casos de los colores reales mostrados en la tabla. No obstante, todos ellos se pueden distinguir con mayor o menor dificultad.

5.3.1. Código implementado

Para la realización de este apartado, que implica modelar el nivel de luminosidad de cada bombilla del LED RGB, se ha sustituido la función *digitalWrite* por la función *analogWrite*. Mientras que la primera solamente permite fijar la salida en encendido (HIGH) o apagado (LOW), la segunda realiza una modulación por ancho de pulso (*Pulse-Width Modulation*, PWM) que permite ajustar la tensión equivalente que recibe el subcircuito resistencia-bombilla LED. Para ello, el pin PWM se sitúa en HIGH únicamente durante la fracción del periodo igual a $N/255$, donde N es escogido por el usuario; y posteriormente se sitúa en LOW durante el resto del periodo. Esta fracción de periodo recibe el nombre de *Duty Cycle*.

Un detalle muy importante consiste en que no todos los puertos digitales de la placa Arduino tienen la función PWM, sino que se debe revisar que junto al número del puerto en la placa aparezca un tilde ~. Por este motivo, se ha alterado la numeración de los pines digitales en el código y se han reajustado las conexiones de forma correspondiente.

Otra modificación introducida ha sido la ampliación del número de rangos de temperatura, debido a la introducción de más colores. En consecuencia, ha sido necesario añadir más entradas al condicional del final del *loop*.

El código implementado para este caso se adjunta a continuación:

```

1 // Definición de pines digitales
2 const int red = 6;
3 const int blue = 5;
4 const int green = 3;
5
6 // Definición de pines analógicos
7 const int pinT = A0;
8 // Definición de temperaturas
9 const double Tamb = 25;
10 const double Tmax = 35;
11 // Lectura asociada a 25°C
12 const int Namb = 509;
```

```
13 // Lectura asociada a 35°C
14 const int Nmax = 980;
15 // Variable de la lectura de V_out
16 volatile int lectura_temperatura=0;
17 // Variable de la temperatura medida
18 float Temp=0.0;
19 // Pendiente de la ecuación de temperatura
20 double pendiente_temp=(double)(Tmax-Tamb)/(double)(Nmax-Namb);
21
22 void setup() {
23     // Establecimiento de la velocidad de comunicación
24     Serial.begin(9600);
25     // Definición de los pines de alimentación del LED como salida
26     pinMode(red, OUTPUT);
27     pinMode(green, OUTPUT);
28     pinMode(blue, OUTPUT);
29 }
30
31 void loop() {
32     // Lectura de la tensión de salida de U3
33     lectura_temperatura=analogRead(pinT);
34     // Cálculo de la Temperatura medida
35     Temp=Tamb+pendiente_temp*(lectura_temperatura-Namb);
36     // Impresión de resultados
37     Serial.print("Lectura temperatura=");
38     Serial.println(lectura_temperatura);
39     Serial.print("Temperatura=");
40     Serial.println(Temp); Serial.println();
41
42     // Iluminación de LEDs en función de la temperatura
43     // Se enciende el LED en el color mostrado
44     if (Temp < Tamb+double(10.0/11.0))
45     {
46         analogWrite(red,255);
47         analogWrite(green,0);
48         analogWrite(blue,225);
49     }
50     // Se enciende el LED en color AÑIL
51     else if (Temp >= Tamb+double(10.0/11.0) && Temp < Tamb+double(2*10.0/11.0))
52     {
53         analogWrite(red,195);
54         analogWrite(green,0);
55         analogWrite(blue,255);
56     }
57     // Se enciende el LED en el color mostrado
58     else if (Temp >= Tamb+double(2*10.0/11.0) && Temp < Tamb+double(3*10.0/11.0))
59     {
60         analogWrite(red,98);
61         analogWrite(green,0);
62         analogWrite(blue,255);
63     }
64     // Se enciende el LED en el color mostrado
65     else if (Temp >= Tamb+double(3*10.0/11.0) && Temp < Tamb+double(4*10.0/11.0))
66     {
67         analogWrite(red,49);
68         analogWrite(green,0);
69         analogWrite(blue,255);
70     }
71     // Se enciende el LED en color AZUL
72     else if (Temp >= Tamb+double(4*10.0/11.0) && Temp < Tamb+double(5*10.0/11.0))
73     {
74         analogWrite(red,0);
75         analogWrite(green,0);
76         analogWrite(blue,255);
77     }
78     // Se enciende el LED en el color mostrado
79     else if (Temp >= Tamb+double(5*10.0/11.0) && Temp < Tamb+double(6*10.0/11.0))
80     {
81         analogWrite(red,0);
82         analogWrite(green,158);
```

```

83     analogWrite(blue,255);
84 }
85 // Se enciende el LED en el color mostrado
86 else if (Temp >= Tamb+double(6*10.0/11.0) && Temp < Tamb+double(7*10.0/11.0))
87 {
88     analogWrite(red,0);
89     analogWrite(green,255);
90     analogWrite(blue,195);
91 }
92 // Se enciende el LED en color VERDE
93 else if (Temp >= Tamb+double(7*10.0/11.0) && Temp < Tamb+double(8*10.0/11.0))
94 {
95     analogWrite(red,0);
96     analogWrite(green,255);
97     analogWrite(blue,37);
98 }
99 // Se enciende el LED en el color mostrado
100 else if (Temp >= Tamb+double(8*10.0/11.0) && Temp < Tamb+double(9*10.0/11.0))
101 {
102     analogWrite(red,128);
103     analogWrite(green,255);
104     analogWrite(blue,0);
105 }
106 // Se enciende el LED en color AMARILLO
107 else if (Temp >= Tamb+double(9*10.0/11.0) && Temp < Tamb+double(10*10.0/11.0))
108 {
109     analogWrite(red,255);
110     analogWrite(green,255);
111     analogWrite(blue,0);
112 }
113 // Se enciende el LED en color ROJO
114 else if (Temp >= Tamb+double(10*10.0/11.0))
115 {
116     analogWrite(red,255);
117     analogWrite(green,64);
118     analogWrite(blue,0);
119 }
120 // Se aplica un retraso de 500 ms entre medición y medición
121 delay(500);
122
123 }
```

5.3.2. Estudio de funcionamiento

A modo de comprobación, se adjuntan las capturas del LED durante su funcionamiento a la vez que los datos mostrados por el *Serial Monitor*. Primeramente, referente al rango $T(^{\circ}\text{C}) < T_{\text{amb}} + \frac{10}{11}$, se observa que el color mostrado para $25,00^{\circ}\text{C}$, en efecto, es el correspondiente a RGB = [255, 0, 225].

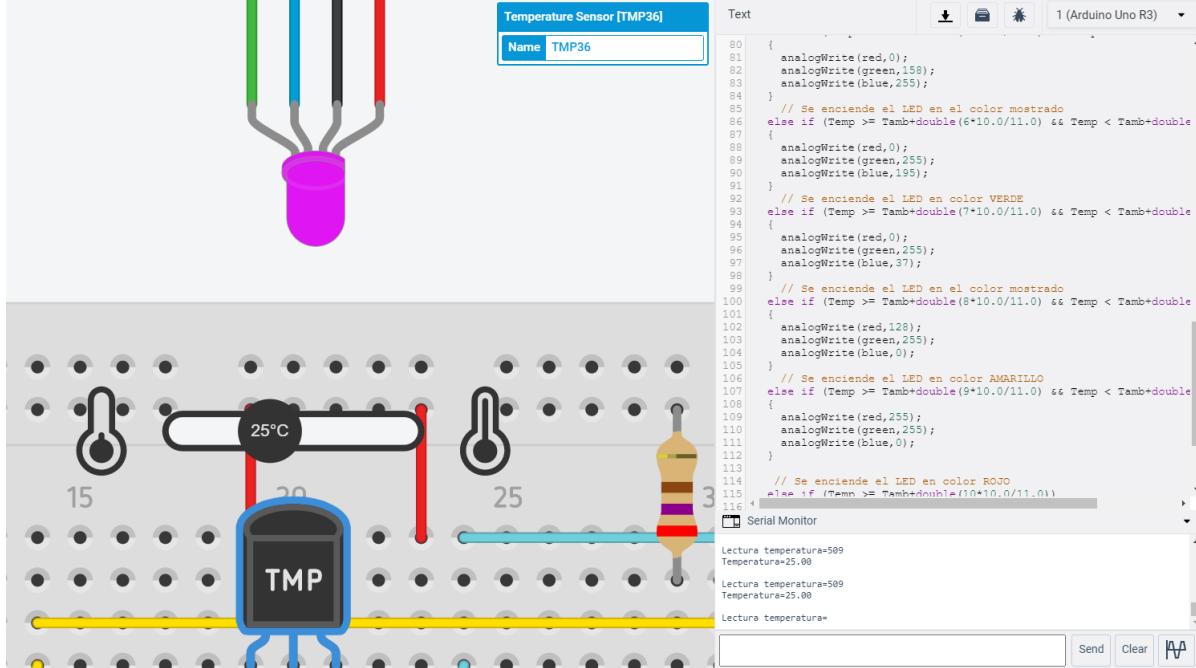


Figura 21: Respuesta del LED RGB para 25°C y escala de colores ampliada.

En segundo lugar, para $26,00^{\circ}\text{C}$, dentro del intervalo $T_{\text{amb}} + \frac{10}{11} \leq T(^{\circ}\text{C}) < T_{\text{amb}} + 2\frac{10}{11}$, se obtiene el color añil o RGB = [195, 0, 255], tal y como se muestra en la siguiente imagen.

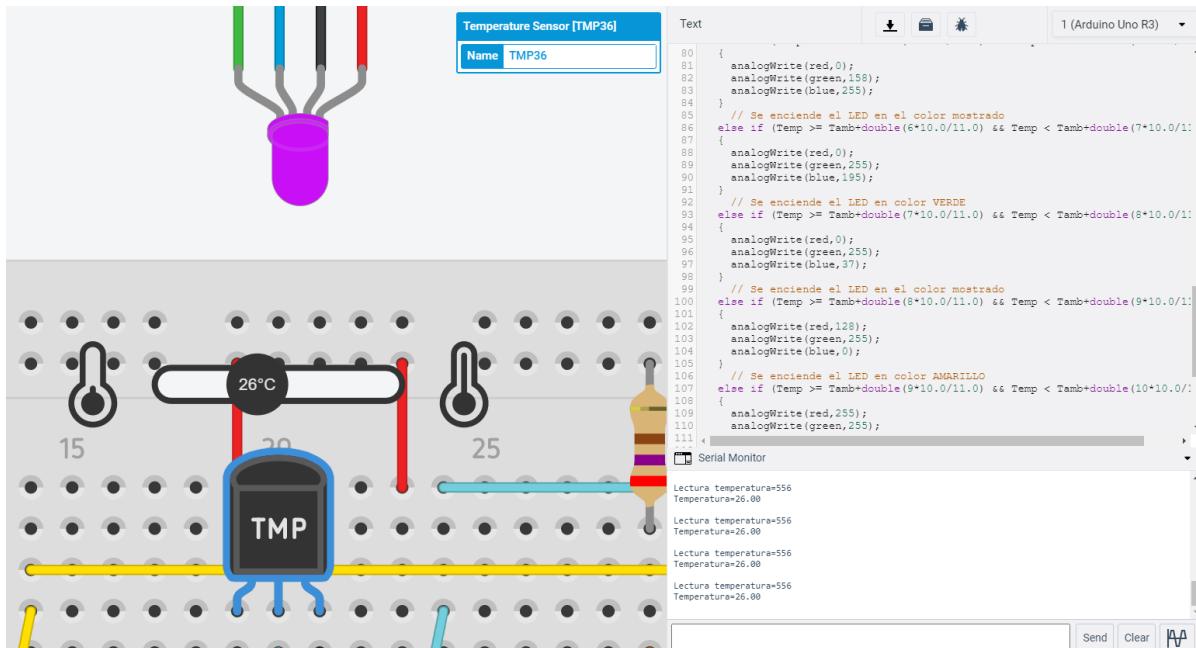


Figura 22: Respuesta del LED RGB para 26°C y escala de colores ampliada.

En cuanto a $27, 00^{\circ}\text{C}$, situado en el rango $T_{\text{amb}} + 2\frac{10}{11} \leq T(^{\circ}\text{C}) < T_{\text{amb}} + 3\frac{10}{11}$, se obtiene el color RGB = [98, 0, 255]:

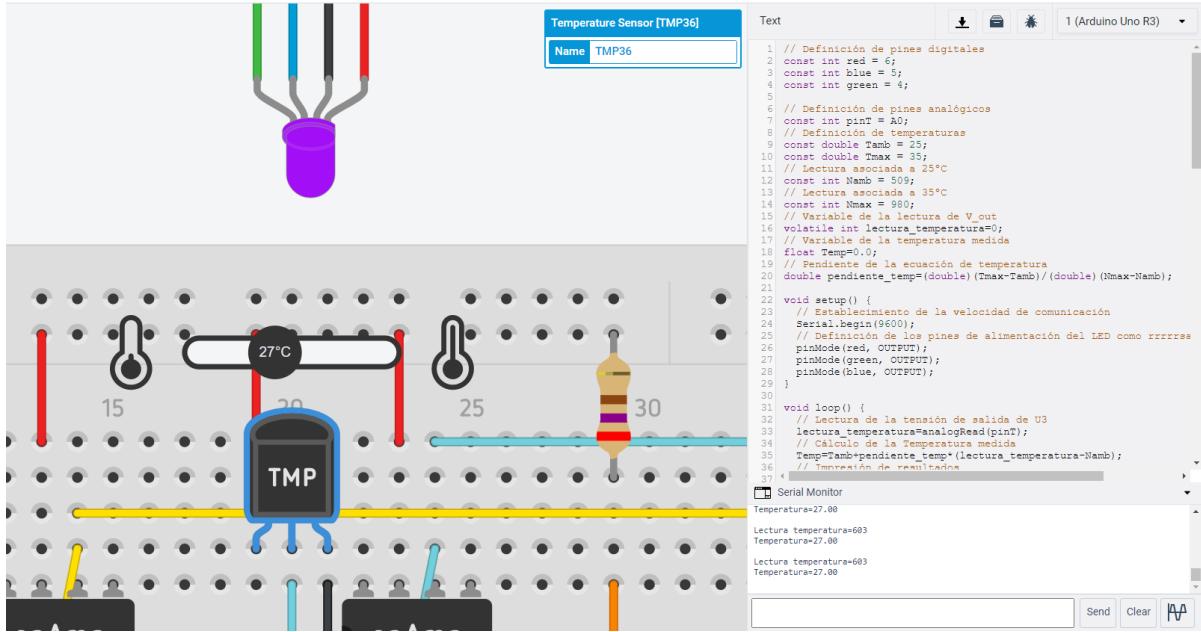


Figura 23: Respuesta del LED RGB para 27°C y escala de colores ampliada.

Para $27, 99^{\circ}\text{C}$, correspondiente a $T_{\text{amb}} + 3\frac{10}{11} \leq T(^{\circ}\text{C}) < T_{\text{amb}} + 4\frac{10}{11}$, el color del LED es RGB = [49, 0, 255]:

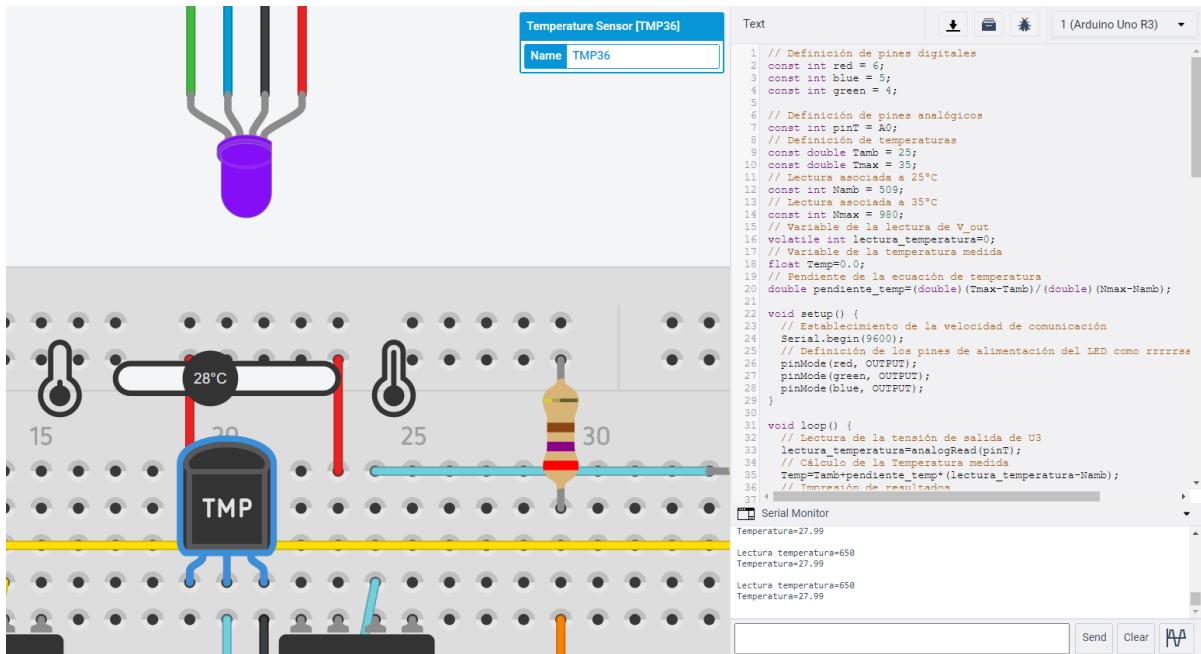


Figura 24: Respuesta del LED RGB para 28°C y escala de colores ampliada.

En el caso de $28, 99^{\circ}\text{C}$, dentro de $T_{\text{amb}} + 4\frac{10}{11} \leq T(^{\circ}\text{C}) < T_{\text{amb}} + 5\frac{10}{11}$, la tonalidad resultante es azul o RGB = [0, 0, 255]:

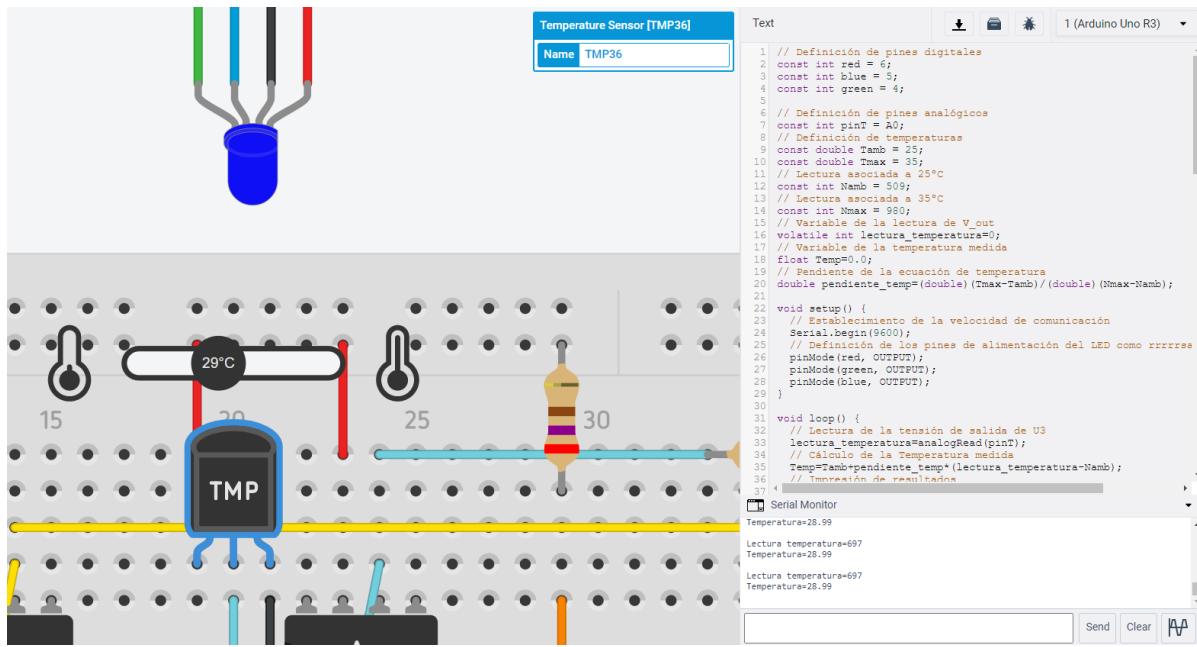


Figura 25: Respuesta del LED RGB para 29°C y escala de colores ampliada.

Para $29,99^{\circ}\text{C}$, en el intervalo $T_{\text{amb}} + 5\frac{10}{11} \leq T(^{\circ}\text{C}) < T_{\text{amb}} + 6\frac{10}{11}$, se obtiene el color RGB = [0, 158, 255]:

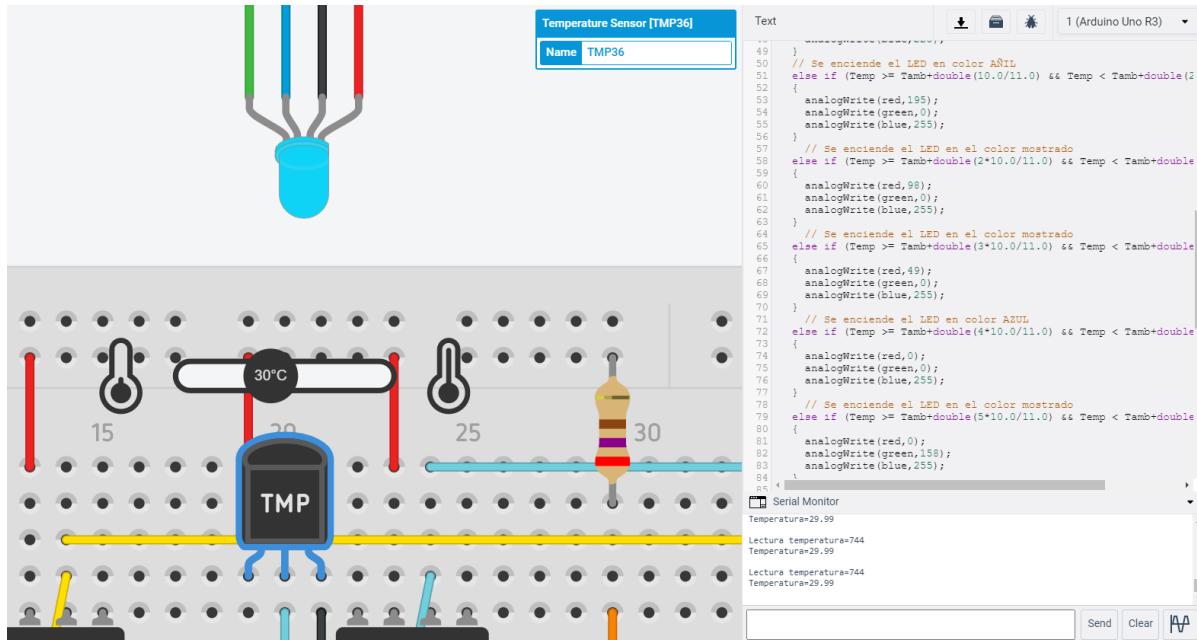


Figura 26: Respuesta del LED RGB para 30°C y escala de colores ampliada.

En referencia a la medición de $31,01^{\circ}\text{C}$, dentro del rango $T_{\text{amb}} + 6\frac{10}{11} \leq T(^{\circ}\text{C}) < T_{\text{amb}} + 7\frac{10}{11}$, el color resultante es RGB = [0, 255, 195]

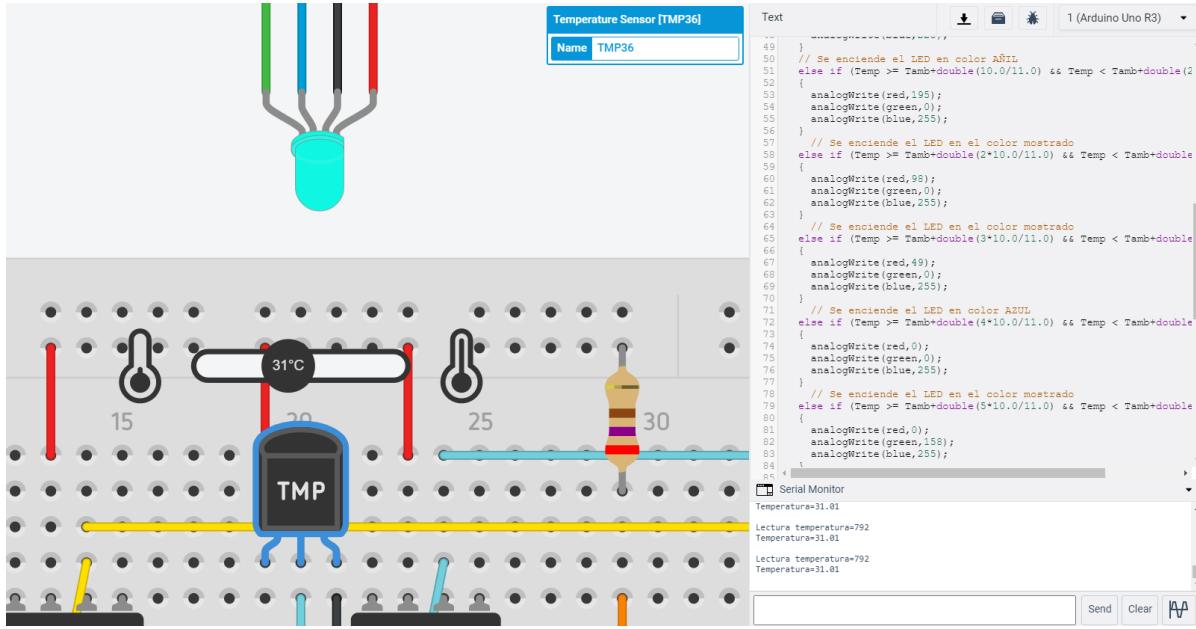


Figura 27: Respuesta del LED RGB para 31°C y escala de colores ampliada.

Para $32,01^{\circ}\text{C}$, contenido en $T_{\text{amb}} + 7\frac{10}{11} \leq T(^{\circ}\text{C}) < T_{\text{amb}} + 8\frac{10}{11}$, el tono obtenido es $\text{RGB} = [0, 255, 37]$:

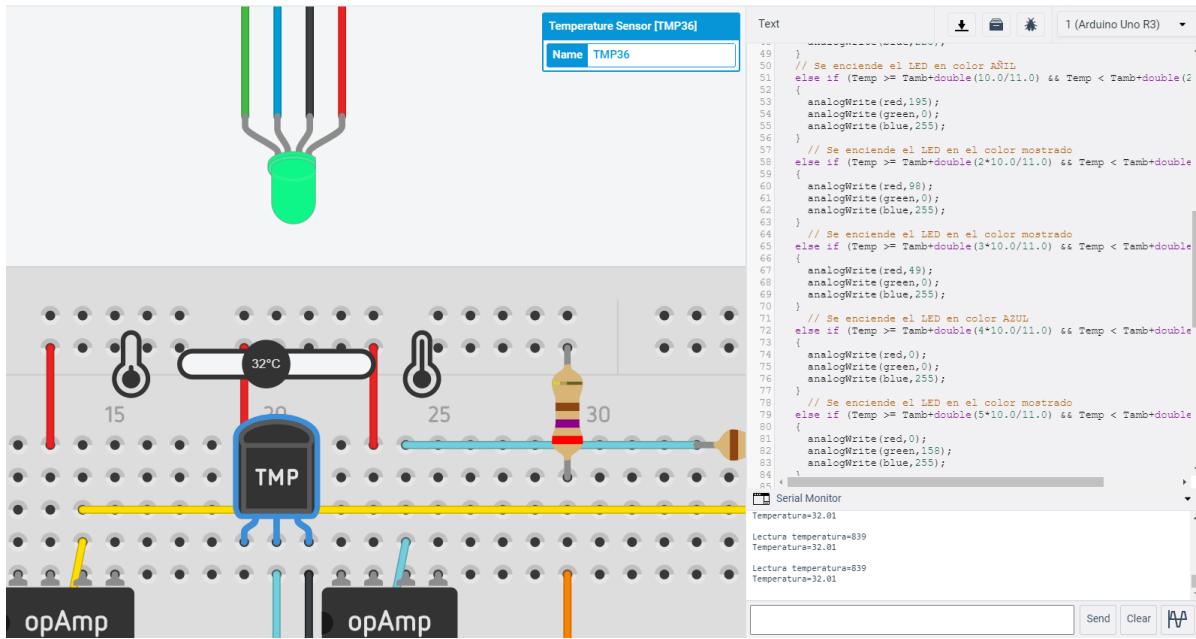


Figura 28: Respuesta del LED RGB para 32°C y escala de colores ampliada.

En el caso de $33,00^{\circ}\text{C}$, para $T_{\text{amb}} + 8\frac{10}{11} \leq T(^{\circ}\text{C}) < T_{\text{amb}} + 9\frac{10}{11}$, se consigue el color $\text{RGB} = [128, 255, 0]$:

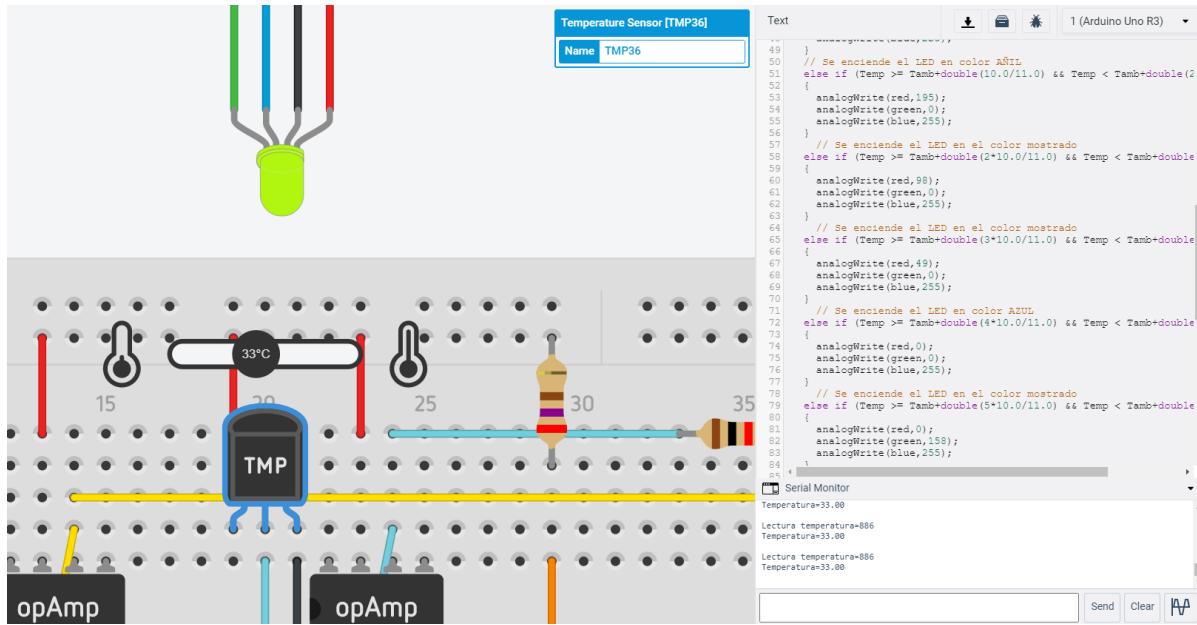


Figura 29: Respuesta del LED RGB para 33°C y escala de colores ampliada.

En penúltimo lugar, un valor de 34,00 °C, dentro del intervalo $T_{amb} + 9\frac{10}{11} \leq T(^{\circ}\text{C}) < T_{amb} + 10\frac{10}{11}$, el LED se ilumina en amarillo:

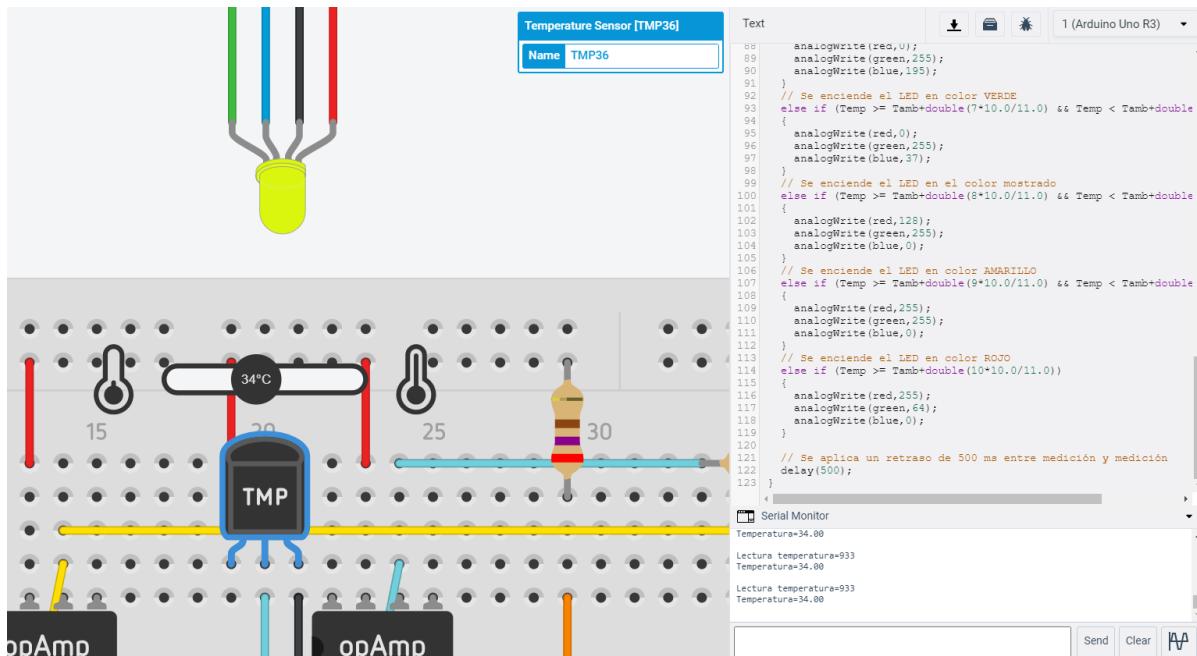


Figura 30: Respuesta del LED RGB para 34°C y escala de colores ampliada.

Finalmente, el LED adquiere una tonalidad anaranjada o $\text{RGB} = [255, 64, 0]$ para 35,00 °C correspondiente a $T(^{\circ}\text{C}) \geq T_{amb} + 10\frac{10}{11}$:

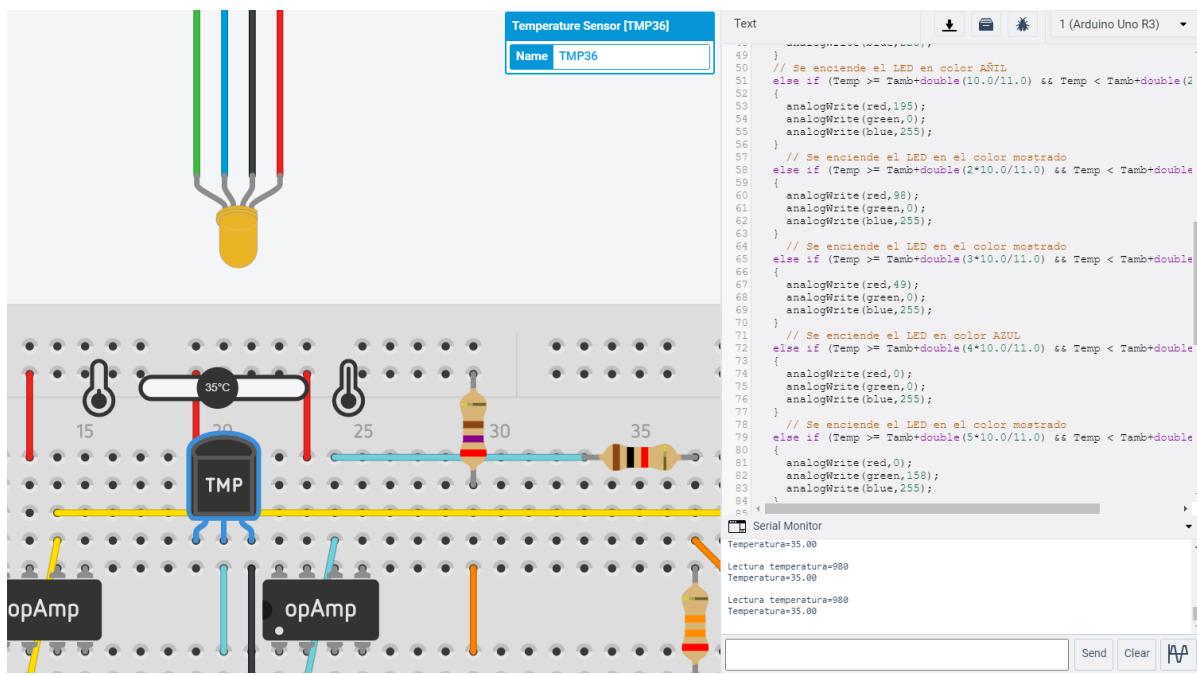


Figura 31: Respuesta del LED RGB para 35°C y escala de colores ampliada.

6. Conclusiones

En esta práctica se ha hecho uso del sensor LM36 con la finalidad de realizar mediciones de temperatura para, posteriormente, poderlas ilustrar en una escala de color mediante el uso de un LED RGB. Para asegurar un amplio rango de entrada se ha implementado un circuito acondicionador basado en el amplificador operacional uA741.

Una vez realizadas las conexiones pertinentes tanto en la protoboard como en el Arduino, se ha procedido al proceso de calibración mediante un potenciómetro. De esta manera, se ha ajustado su dial con tal de obtener un cero del sistema para la temperatura ambiente y centrada en 2,5 V (concretamente, el valor de cero conseguido ha sido de 2,49 V).

Finalmente, se han llevado a cabo diversas simulaciones con 3 tipos de escalas de colores diferentes: tricolor, secundaria y ampliada. Mientras que en las dos primeras la función empleada ha sido la *digitalWrite*, en la última ha sido la *analogWrite* con la finalidad de alimentar el LED mediante una señal PWM, obteniendo así un mayor abanico de tonalidades. En cuanto al error obtenido se refiere, destacar que el *Serial Monitor* ha arrojado valores de temperatura que en ningún caso han superado el 0,04 % respecto la temperatura real. Por tanto, se puede considerar que el sistema utilizado ofrece una considerable fiabilidad y precisión.

Igual que se dio para la práctica 1, todo el montaje y simulación ha sido llevado a cabo en la plataforma en línea Tinkercad. Un pequeño inconveniente ha sido que el cursor deslizante de la temperatura “real” que lee el sensor LM36 tenía un paso de $\pm 1^{\circ}\text{C}$, hecho que impedía configurar temperaturas exteriores con decimales. En segunda instancia, para una misma temperatura entera, el cursor ofrecía dos posiciones, siendo la situada más a la derecha incorrecta, de forma que se requería precisión en la ubicación del cursor deslizante. Por último, el control del cursor es manual con el ratón de forma que hay que ser muy cuidadoso para ajustar el valor deseado, teniendo a la vez en cuenta el problema de las dos posiciones del cursor.