

INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO



ELECTRÓNICA ANALÓGICA

PROF. MARTÍNEZ DÍAZ JUAN CARLOS

EVALUACIÓN PRÁCTICA

SENSOR DE TEMPERATURA

2CM5

COLABORADORES:

ALANÍS RAMÍREZ DAMIÁN

CARMONA MEDINA VÍCTOR ÁNGEL

GÓMEZ RODRÍGUEZ EDUARDO

Sensor de Temperatura - Alanís, Carmona, Gómez.

Índice

1. Objetivo general	3
2. Objetivos específicos	3
3. Material empleado	3
4. Equipo empleado	3
5. Introducción	4
6. Desarrollo	7
Planteamiento del problema	7
Diagrama a bloques	7
Bloque del sensor de temperatura LM335	7
- Circuito de calibración	7
Bloque del CAS	8
- Determinación de la ecuación del CAS	8
- Circuito de amplificador inversor o buffer inversor	10
- Circuito sumador de voltajes inversor	11
Bloque del microcontrolador	14
- ADC	14
- Pantalla LCD	14
- Bluetooth	15
7. Evidencias de mediciones y pruebas	16
8. Circuito esquemático final	18

9. Datos	19
Datos teóricos	19
Datos prácticos	21
10. Código del microcontrolador	24
11. Conclusiones	25
12. Referencias	28

Objetivo general

Diseñar y construir un circuito que permita conocer la temperatura, mostrándola en una interfaz agradable, aplicando conceptos y circuitos específicos como los acondicionadores de señal, estudiados en la unidad de aprendizaje de electrónica analógica.

Objetivos específicos

- Reforzar la comprensión del principio de funcionamiento y aplicaciones de los amplificadores operacionales en sus distintas configuraciones.
- Aplicar conocimientos teóricos para el diseño de un CAS que permita una mejor comunicación con un microcontrolador para realizar una conversión de una señal analógica a una digital.
- Observar y aprovechar el principio de funcionamiento de elementos activos como el sensor de temperatura LM335, además de comprender términos como sensibilidad y realizar su calibración adecuada.
- Observar, finalmente, el resultado de la variable física - la temperatura – por medio de una interfaz amigable visualmente, integrando, de por medio, un microcontrolador con convertidor analógico a digital y comunicación serial para la posibilidad de comunicación Bluetooth.

Material empleado

- 2 protoboards
- 1 sensor LM335
- 1 microcontrolador PIC16F886
- 2 amplificadores operacionales LM741
- 1 resistor de 2.2K Ω
- 3 resistores de 10K Ω
- 1 resistor de 100K Ω
- 1 trimpot (potenciómetro de precisión) de 100K Ω
- 1 trimpot de 10K Ω
- 1 pantalla LCD de 16x2
- 1 módulo Bluetooth HC-06
- 1 potenciómetro de 5K Ω
- Alambre cal. 22

Equipo empleado

- Fuente de alimentación de laboratorio
- Cables banana-caimán
- Multímetro
- Encendedor
- Termómetro

Introducción

Este trabajo busca plasmar el proceso de diseño y construcción de un circuito que nos permita conocer la temperatura mediante el uso de un sensor de temperatura LM335 y el uso de un circuito acondicionador de señal, donde se aplican varios aspectos abordados en las clases de electrónica analógica, para esto es necesario abordar ciertos conceptos y apartados teóricos que se desarrollan a continuación.

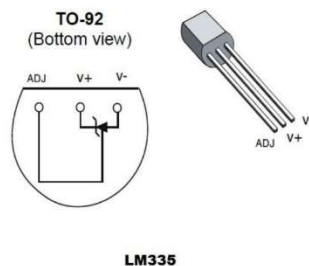
Acondicionamiento de una señal

Para el diseño de este sensor de temperatura se deberá realizar un acondicionamiento de la señal de salida para poder analizarla y trabajar con ella. Uno de los objetivos será analizar las propiedades de la señal que el sensor emite al reaccionar al entorno y tratar de aproximarla a una forma lo más lineal posible. Por lo tanto, es necesario utilizar un circuito acondicionador de señal para tratar la salida, al final lo que se obtendrá será un circuito que proporciona voltajes que van de 0V a 5V de acuerdo a la temperatura que oscila desde 0°C hasta 50°C. Esta parte del circuito del acondicionador de señal es sumamente importante y se trató de forma extensiva en clase, por lo que el diseño del CAS estuvo en todo momento bien sustentada teórica y prácticamente.

Para la implementación del CAS se debe considerar la obtención del nivel adecuado de la señal. Si la señal entrara a un convertidor analógico a digital (ADC) para después ser procesada por una computadora, que en nuestro caso es un microcontrolador, será necesario amplificarla y, además, aproximarla lo más que se pueda a un comportamiento lineal respecto a la temperatura. En la amplificación es muy común utilizar amplificadores operacionales, y de hecho eso es lo que se hará en la etapa de obtención de la señal de salida.

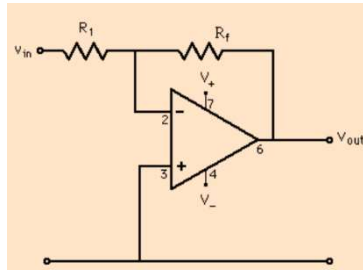
Los acondicionadores de señales son especialmente útiles para sensores o transductores con salidas no lineales, como los termopares, que llegan a ser de comportamiento polinomial aproximado. Son de especial utilidad sobre todo para limitar la señal a un rango aceptable por otro circuito como un ADC, que es lo que sucede en este caso.

Sensor de temperatura LM335



El LM335 es un sensor de temperatura de alta precisión de fácil calibración, funciona como un diodo Zener de dos terminales, tiene un voltaje de ruptura directamente proporcional a la temperatura absoluta a 10mV/°K, con menos de 1Ω de impedancia, pero aun así se implementó el buffer inversor para el acoplamiento de las impedancias y para facilitar las operaciones posteriores; el dispositivo opera sobre un rango de corriente de 400μA a 5Ma. Cuando es calibrado a 25°C, el LM335 tiene típicamente menos de 1°C de error sobre 100°C de rango de temperatura. En nuestro caso empleamos el LM335 con encapsulado TO-92.

Buffer inversor o amplificador inversor

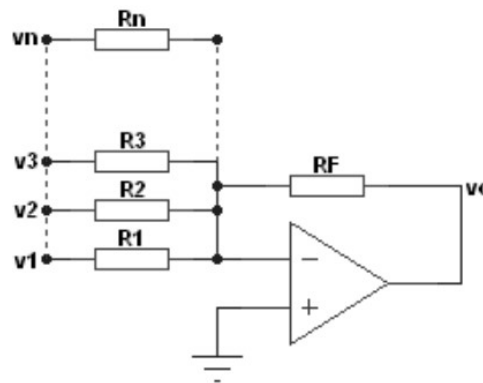


En un amplificador operacional la ganancia del mismo está determinado por la siguiente relación:

$$\frac{V_{out}}{V_{in}} = -\frac{R_f}{R_1}$$

Si ambas resistencias son iguales, se obtiene una ganancia de -1 , es decir, se tiene el mismo voltaje a la salida, solo que negativo, por este motivo a este circuito se le conoce como buffer inversor.

Sumador inversor



El circuito sumador es un circuito muy útil, basado en la configuración estándar del amplificador operacional inversor. Este circuito permite combinar múltiples entradas, es decir, permite añadir algebraicamente dos (o más) señales o voltajes para formar la suma de dichas señales. La tensión de salida del circuito sumador se define mediante la siguiente expresión:

$$V_o = -\left(\frac{R_f}{R_1}\right)V_1 + \left(\frac{R_f}{R_2}\right)V_2 + \dots + \left(\frac{R_f}{R_n}\right)V_n$$

ADC

Los ADC o convertidores analógicos a digitales son circuitos que casi siempre actúan como un dispositivo intermedio que convierte las señales de naturaleza analógica a una señal digital que pueda, posteriormente, ser computable, o ingresada a cualquier circuito digital. Estos circuitos nos permiten trabajar digitalmente con datos del entorno, que son de naturaleza analógica. Por ejemplo, nosotros encontramos una gran diversidad de sensores que convierten las características físicas del medio en señales analógicas sensores tales como la cantidad de luz con los LDR, que son sensores que modifican su resistencia de acuerdo a la cantidad de luz que recibe. Toda señal proveniente del entorno y que pueda ser ingresada a un sensor que responda modificando un parámetro eléctrico de acuerdo a dicha variable

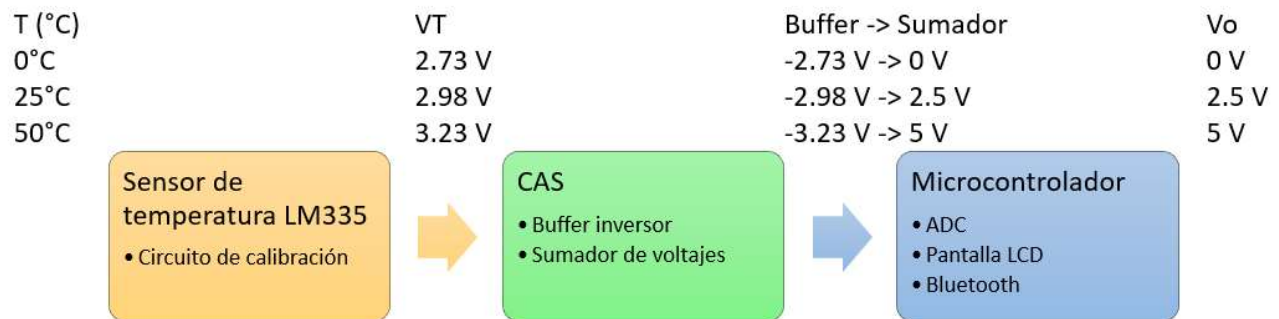
puede ingresarse a un ADC para tratarse digitalmente y así ampliar su capacidad de servir como información para sistemas, que ya son en su mayoría digitales.

Desarrollo

Planteamiento del problema

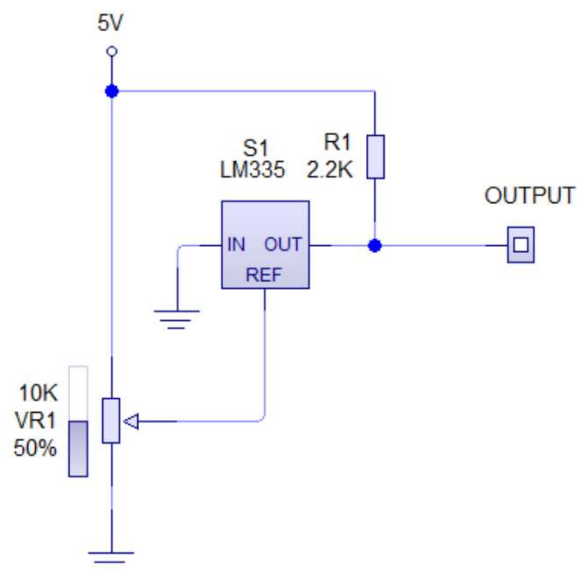
Diseñar un circuito de acondicionamiento de señal (CAS) que facilite conectar un sensor de temperatura y un convertidor A/D de un microcontrolador para mostrar la temperatura actual en una interfaz (pantalla LCD y aplicación móvil mediante comunicación Bluetooth). El margen de temperatura será de 0°C a 50°C, el rango de tensión de la salida del CAS será de 0V a 5V. Se este circuito de acondicionamiento de señal tenga un comportamiento lineal, es decir que a una temperatura de 0°C la salida del CAS sea de 0V; mientras que cuando el sensor mida 10°C, la salida del CAS será de 1V, y así sucesivamente hasta llegar a 50°C la salida del CAS sea de 5V.

Diagrama a bloques

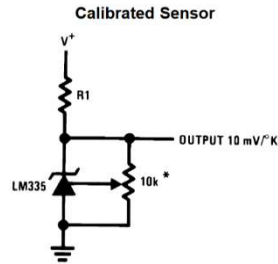


Bloque de sensor de temperatura LM335

Circuito de calibración del sensor



Para la calibración del sensor nos dirigimos a la hoja de datos, donde se indica una configuración específica, la cual se muestra a continuación:

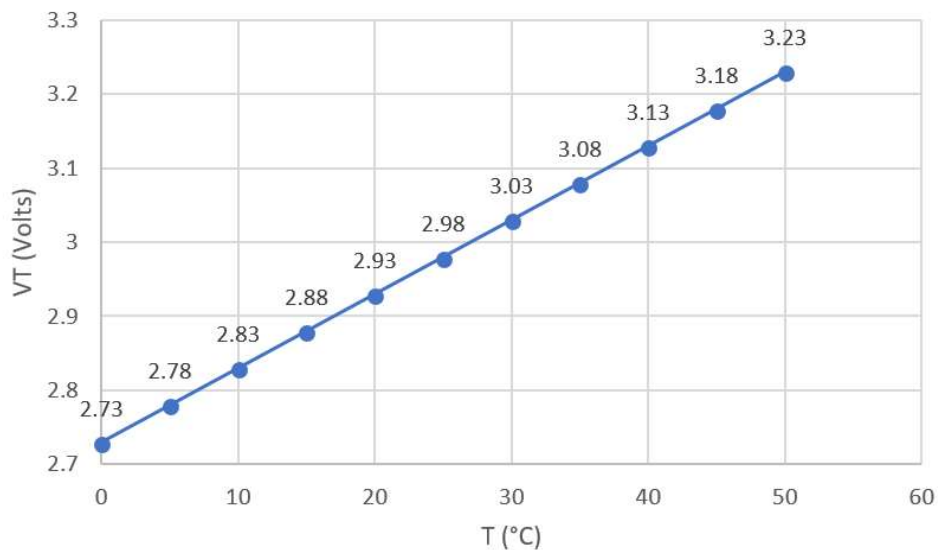


Más adelante en la hoja de datos se indica que el valor recomendado de R1 es de 2KΩ, esto con el fin de que por el sensor fluya una intensidad de corriente de 0.4 a 5mA. Por lo que se optó por el valor comercial más cercano, que es de 2.2KΩ. Lo que se debe obtener al medir la diferencia de potencial a la salida esta dado por la hoja de datos, la cual especifica 2.98V a 298.15 °K, es decir, a 25°C, por lo que se obtienen los siguientes valores, considerando que este sensor tiene un incremento de 10mV/°K y convirtiendo la temperatura a °C:

0°C	50°C
2.73V	3.23V

De acuerdo a esto, se tomó el valor de temperatura que se tenía en el laboratorio y se calculó el voltaje que debía obtenerse a la salida, de modo que si no se tenía dicho valor se ajustaba el potenciómetro de precisión hasta que se obtuviera dicho valor.

Esto nos permite obtener igualmente una primera gráfica, del comportamiento ideal del sensor, es decir, ya calibrado:



Gráfica 1: Relación del voltaje proveniente del sensor LM335 con la temperatura del entorno.

De lo cual se puede obtener la ecuación que describe su comportamiento mediante el método de punto pendiente.

Lo que obtenemos es la ecuación:

$$VT = \frac{10mV}{^{\circ}C} (Tc) + 2.73V$$

Considerando:

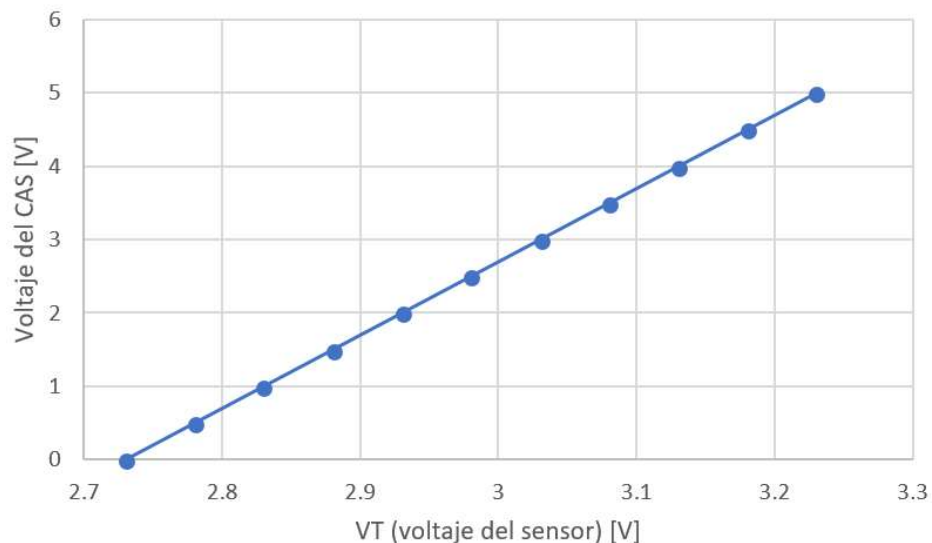
- V_T es el voltaje de salida del sensor
- $10\text{mV}/^\circ\text{C}$ es la sensibilidad nominal del LM335
- T_c es la temperatura con la que está interactuando el sensor
- Bloque de sensor de temperatura LM335

Bloque del CAS

Determinación de la ecuación del CAS

Para el CAS requerimos que la salida del mismo se encuentre en el intervalo cerrado $[0\text{V}, 5\text{V}]$, esto porque el ADC del microcontrolador soporta solo ese intervalo de voltaje a su entrada, que luego transforma a un número entre 0 y 1023, por ello se debe acondicionar de cierta forma la salida del sensor, que vale 2.73V a 0°C y 3.23V a 50°C , pues además queremos que la gráfica sea lo más lineal posible, conservando la precisión característica del sensor LM335.

A continuación, se muestra una gráfica del voltaje de salida del CAS (el que deseamos obtener) contra el voltaje del sensor LM335, esto va desde 0°C hasta 50°C con variaciones de 5°C . Se puede apreciar que tiene un comportamiento lineal, por lo que sí es posible obtener una salida lineal en el CAS.



De esta gráfica se pueden destacar que, dado su comportamiento lineal se puede describir por una ecuación de la forma:

$$V_o = mx + b$$

Donde la variable x se puede ver que es el voltaje V_T , es decir, el que proviene del sensor, por lo que se establece;

$$V_o = mV_T + b$$

Y ahora procedemos a calcular la pendiente m por medio de su definición:

$$m = \frac{\Delta V_{CAS}}{\Delta V_T}$$

Se toman los valores extremos, es decir, de 0 a 5V para el CAS y de 2.73 a 3.23V para el VT.

$$m = \frac{(5V - 0V)}{(3.23V - 2.73)} = 10$$

Obtenemos el valor de la pendiente en 10, por lo que la ecuación empieza a tomar más forma:

$$V_O = 10V_T + b$$

Ya solo resta calcular la b.

$$b = V_O - 10V_T$$

Y para obtener b tomaremos en cuenta el valor mínimo del CAS, es decir, 0V, que se obtiene a la temperatura de 0°C, por lo que la salida del sensor es 2.73V, se sustituyen estos valores en la ecuación anterior y obtenemos:

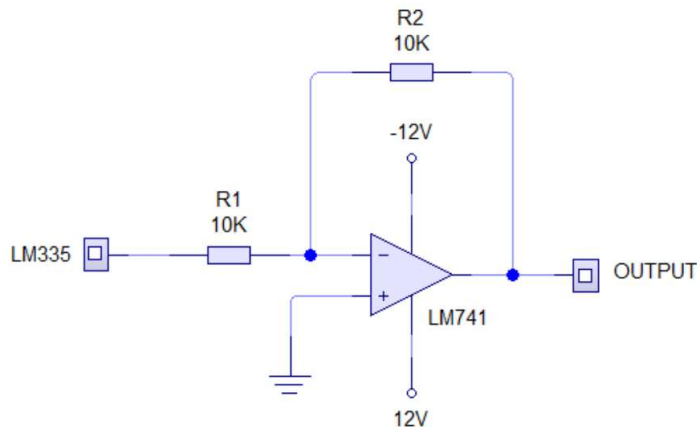
$$b = 0V - 10(2.73V) = -27.3V$$

Por lo que la ecuación final para el diseño del CAS es:

$$V_O = 10V_T - 27.3V$$

A continuación, se explicarán las etapas del CAS que nos llevarán a obtener este comportamiento a la salida.

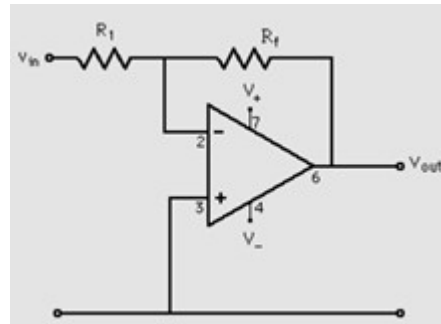
Circuito de amplificador inversor o buffer inversor



En este circuito se emplea la configuración de amplificador inversor, donde las dos resistencias al ser iguales proporcionan una ganancia de -1 , de modo que tenemos prácticamente un seguidor de voltaje, solo que con la salida invertida, a esto se le denomina buffer inversor, y su finalidad es el acoplar las impedancias que provienen del sensor LM335 con las provenientes del resto del CAS, y el propósito de invertir la señal es facilitar la operación de resta para la conformación de la ecuación final (pues este resultado será introducido al sumador de la próxima etapa).

Al acoplarse adecuadamente estas impedancias se evitan errores que podrían derivarse de señales residuales, a las cuales el microcontrolador es sumamente sensible.

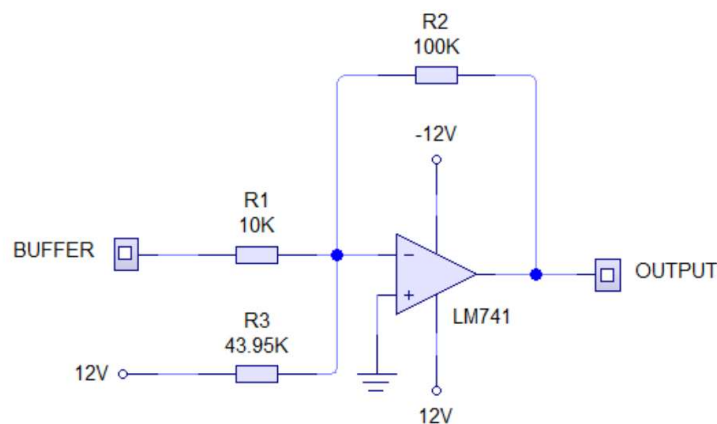
Para el cálculo de las resistencias se consideró la teoría del amplificador en modo amplificador inversor, donde se tiene que:



$$A_v = \frac{V_o}{V_i} = \frac{-R_f}{R_1}$$

Por lo que si se desea una ganancia de -1 las resistencias R_f y R_1 deben ser iguales, para este caso, de $10\text{K}\Omega$.

Circuito sumador de voltajes



Para este circuito se eligió la configuración de sumador de voltajes inversor, pues esta nos permite llegar a la ecuación $V_{out} = 10v_T - 27.3V$, esto considerando que cada miembro de la ecuación es una resistencia en la entrada inversora del operacional, donde además, recibiremos un voltaje negativo proveniente del buffer en V_a , de acuerdo a la figura 1, esto nos deja con la necesidad de incluir otro voltaje en V_b , que elegimos fuera de $12V$ aprovechando que teníamos fuente de voltaje de ese valor, el siguiente paso era calcular las resistencias que nos permitieran obtener una señal de salida con el comportamiento descrito por la ecuación que se especificó.

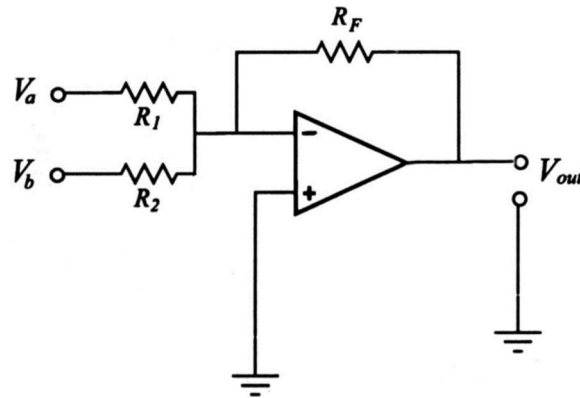


Figura 1: Operacional como sumador de voltajes (inversor), V_a es el proveniente del buffer inversor y V_b son 12V

Los valores de resistencia se calcularon de la siguiente manera, considerando primeramente la ecuación característica de este circuito en particular:

$$V_{out} = - \left[\left(\frac{R_f}{R_1} \right) V_a + \left(\frac{R_f}{R_2} \right) V_b \right]$$

Donde:

- V_{out} es el voltaje a la salida del sumador, y que, de hecho, es la salida del CAS.
- R_f es la resistencia de referencia, que se propuso de 100KΩ.
- R_1 y R_2 son las resistencias de V_a y V_b respectivamente.
- V_a es el voltaje proveniente de la salida del buffer inversor (será la parte de la variable de la ecuación).
- V_b es un valor de voltaje fijo, que se propuso de 12V.

Además, considerando la teoría existente sobre esta configuración, tenemos que se puede obtener una señal de salida que se ajuste a una ecuación deseada, que en nuestro caso es: $V_{out} = 10v_T - 27.3V$

Tomando en cuenta que a una temperatura de 0°C tenemos 2.73V a la salida del sensor y -2.73V a la salida del buffer inversor, además de que se estaría eligiendo una resistencia R_f de 100KΩ y se considerará V_a como la salida del buffer inversor, mientras que V_b será una fuente fija de 12V, entonces se tiene que:

$$0V = - \left[\left(\frac{100K\Omega}{R_1} \right) (-2.73V) + \left(\frac{100K\Omega}{R_2} \right) (12V) \right]$$

Que al desarrollar más se obtiene:

$$0V = - \left[\left(\frac{-273[V][K\Omega]}{R_1} \right) + \left(\frac{1200[V][K\Omega]}{R_2} \right) \right]$$

$$0V = \frac{273[V][K\Omega]}{R_1} - \frac{1200[V][K\Omega]}{R_2}$$

Lo cual tiene la forma de la ecuación deseada $V_{out} = 10v_T - 27.3V$

Entonces, podemos obtener R1 considerando que debemos obtener en ese miembro de la ecuación 10 veces el voltaje de salida del sensor a esa temperatura, en este caso 2.73V, por lo que $10V_T$ adquiriría el valor de 27.3V, que es lo que se debe obtener en este primer miembro de la ecuación, para el segundo miembro, se debe considerar que toda esa expresión debe ser igual a los 27.3V que aparecen al final de la ecuación original, y al final se puede ver que para °C ambos miembros son iguales (27.3V), por lo que se anulan y se cumple con que la salida debe ser 0V, desarrollando para cada miembro:

$$\begin{aligned} 27.3V &= \frac{273[V][K\Omega]}{R1} & -27.3V &= -\frac{1200[V][K\Omega]}{R2} \\ R1 &= \frac{273[V][K\Omega]}{27.3V} & R2 &= -\frac{1200[V][K\Omega]}{-27.3V} \\ R1 &= 10K\Omega & R2 &= 43.956K\Omega \end{aligned}$$

Y de esa manera fue que se obtuvieron los valores de resistencias que se muestran en el circuito, se puede comprobar que se satisface la ecuación deseada, pues, teóricamente, a 0°C se obtienen 0V a la salida, y para 50°C, considerando que la salida del buffer inversor sería -3.23V, (se sustituye en V_a), se tiene:

$$\begin{aligned} V_{out} &= -\left[\left(\frac{100K\Omega}{10K\Omega}\right)(-3.23V) + \left(\frac{100K\Omega}{43.95K\Omega}\right)(12V)\right] \\ V_{out} &= -[(-32.3V) + (27.3V)] \\ V_{out} &= -[-5V] = 5V \end{aligned}$$

Es decir, se cumple la especificación, y, por tanto, los valores de resistencia elegidos son los correctos.

Ya conformado el CAS, se toma la salida de este y se lleva al módulo de conversión analógica a digital (ADC) del microcontrolador PIC16F886.

Bloque del microcontrolador

ADC

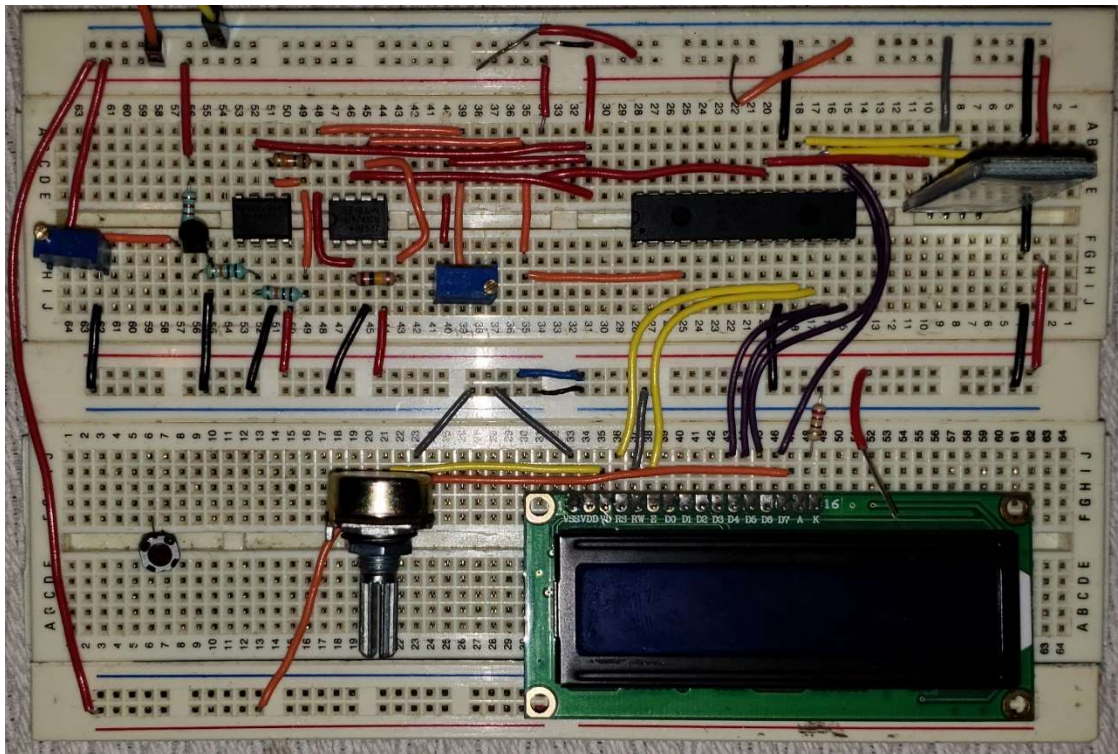
El microcontrolador empleado es el PIC16F886 de Microchip, que cuenta con convertidor analógico a decimal (ADC) de 10 bits, lo que debe ser considerado en la programación, pues el resultado de la conversión de un valor de tipo analógico (voltajes que varían de acuerdo a la temperatura) será un numero entre 0 a 1023, de modo que el resultado de la conversión esta dado por la relación:

$$Salida = \frac{5V_{in}}{1023}$$

Lo anterior multiplicado por 10 nos da la temperatura en grados Celsius, recordando que en el CAS obtenemos 0V a 0°C y 5V a 50°C, es decir, obtenemos en voltaje la décima parte de lo que tenemos en temperatura ateniéndonos al comportamiento lineal del circuito, esto nos permite establecer la relación final para la conversión dentro del código del PIC:

$$Salida = \frac{50V_{in}}{1023}$$

Como se puede apreciar en el esquemático se toman salidas del microcontrolador hacia el display LCD y al módulo Bluetooth por medio de comunicación serial con los pines Tx y Rx. Esto se conecta a los respectivos módulos, quedando el circuito final en la protoboard como se muestra a continuación:



Pantalla LCD

Para la pantalla LCD se realizaron las conexiones a los pines del puerto C del microcontrolador, desde el RC0 (pin 0 del puerto C) hasta el RC5, que corresponden, en orden, a las terminales RS, Enable, Data 4, Data 5, Data 6 y Data 7 de la LCD, esto debido a que la librería LCD.h del IDE MikroC PRO for PIC requiere definir estos pines como salidas en el programa de la siguiente manera:

```
sbit LCD_RS at RC0_bit;
sbit LCD_EN at RC1_bit;
sbit LCD_D4 at RC2_bit;
sbit LCD_D5 at RC3_bit;
sbit LCD_D6 at RC4_bit;
sbit LCD_D7 at RC5_bit;

//Direccion de los bits de la LCD
sbit LCD_RS_Direction at TRISC0_bit;
sbit LCD_EN_Direction at TRISC1_bit;
sbit LCD_D4_Direction at TRISC2_bit;
```



```
sbit LCD_D5_Direction at TRISC3_bit;
```

```
sbit LCD_D6_Direction at TRISC4_bit;
```

```
sbit LCD_D7_Direction at TRISC5_bit;
```

Además, fue necesario añadir un potenciómetro para el ajuste del contraste y una resistencia para el LED de retroiluminación de la LCD. Otras conexiones que se hicieron se muestran en el esquemático que se encuentra más adelante.

A continuación se muestra el código fuente desarrollado en “Mikro C” con el cual se programo el microcontrolador:

```
sbit LCD_RS at RC0_bit;
sbit LCD_EN at RC1_bit;
sbit LCD_D4 at RC2_bit;
sbit LCD_D5 at RC3_bit;
sbit LCD_D6 at RC4_bit;
sbit LCD_D7 at RC5_bit;
//Direccion de los bits de la LCD
sbit LCD_RS_Direction at TRISC0_bit;
sbit LCD_EN_Direction at TRISC1_bit;
sbit LCD_D4_Direction at TRISC2_bit;
sbit LCD_D5_Direction at TRISC3_bit;
sbit LCD_D6_Direction at TRISC4_bit;
sbit LCD_D7_Direction at TRISC5_bit;

//Iniciar protocolos y limpiar las cadenas de caracteres
void config();
void limpia();

//Declaramos variables
float temperatura, res, resultado = 0.00;
unsigned int lectura;
char pantallaLCD[14], pantallaBT[14];

void main(){
    config();
    UART1_Init(9600); // Inicializa el BlueTooth (BT)
    delay_ms(100);
    lectura = 1;
    while(1){
        limpia();

        //Para leer la temperatura real
        lectura = ADC_Read(2);
```



```

//Para leer la temperatura real (A/D) que nos entrega 10 bits y guardamos como uint
lectura = ADC_Read(2);

//Para poner un ciclo que simule la temperatura (Pruebas sin polarización, trabaja
solo con los 5V a 3A
/*
if(lectura > 1023){
    lectura = 0;
}
lectura = lectura +1;  */
temperatura = (int)lectura;
resultado = 50.00/1024.00*temperatura;
FloatToStr(resultado, pantallaLCD); //Conversión para la salida del LCD a string

IntToStr(lectura, pantallaBT); //Conversión para transmisión BT
UART1_Write_Text(pantallaBT); //Transmisión via BT
Lcd_Out(1,1, "Temperatura: °C");
Lcd_Out(2,3, pantallaLCD); //Transmisión vía LCD
Delay_ms(800);
}
}

void limpia(){
    pantallaLCD[0] = ' ';
    pantallaLCD[1] = ' ';
    pantallaLCD[2] = ' ';
    pantallaLCD[3] = ' ';
    pantallaLCD[4] = ' ';
    pantallaLCD[5] = ' ';
    pantallaLCD[6] = ' ';
    pantallaLCD[7] = ' ';
    pantallaLCD[8] = ' ';
    pantallaLCD[9] = ' ';
    pantallaLCD[10] = ' ';
    pantallaLCD[11] = ' ';
    pantallaLCD[12] = ' ';
    pantallaLCD[13] = ' ';
}

```

```

    pantallaBT[0] = '';
    pantallaBT[1] = '';
    pantallaBT[2] = '';
    pantallaBT[3] = '';
    pantallaBT[4] = '';
    pantallaBT[5] = '';
    pantallaBT[6] = '';
    pantallaBT[7] = '';
    pantallaBT[8] = '';
    pantallaBT[9] = '';
    pantallaBT[10] = '';
    pantallaBT[11] = '';
    pantallaBT[12] = '';
    pantallaBT[13] = '';
    Lcd_Cmd(_LCD_CLEAR);
}

void config(){
    ANSEL = 0x04;      // Configura el pin AN2 como digital
    ANSELH = 0;        // Configura el pin AN como digital
    C1ON_bit = 0;      // Deshabilita los comparadores
    C2ON_bit = 0;

    TRISA = 0xFF;
    TRISC = 0;
    TRISB = 0;
    Lcd_Init(); // Inicializa el LCD
    ADC_Init(); // Inicializa el conversor A/D
    Lcd_Cmd(_LCD_CURSOR_OFF);
    Lcd_Cmd(_LCD_CLEAR);
}

```

Bluetooth

Para el Bluetooth se emplea la comunicación serial del microcontrolador que, según la hoja de datos, se invoca por medio de la librería UART y SoftwareUART, además de que se cuentan con funciones importantes dentro de esas librerías, como el UARTx_Init(int), que nos permite inicializar cualquiera de los módulos UART disponibles a una tasa de baudios (caracteres transmitidos por segundo) específica, esta cifra se recomienda sea 9600 baudios, pues es el estándar en este tipo de comunicaciones, además de que es la recomendación de microchip para estos modelos específicos de microcontroladores.

La otra función importante que se empleó fue UARTx_Write_Text(char *) que nos permite transmitir un arreglo de caracteres, que en nuestro caso fue la lectura del ADC convertida a cadena de caracteres con la función IntToStr(int, char *) de la librería de conversiones de mikroC PRO for PIC, esta información se

recibe por el módulo Bluetooth HC-06, conectado en forma serial a las terminales Tx y Rx (RC6 y RC7 respectivamente).

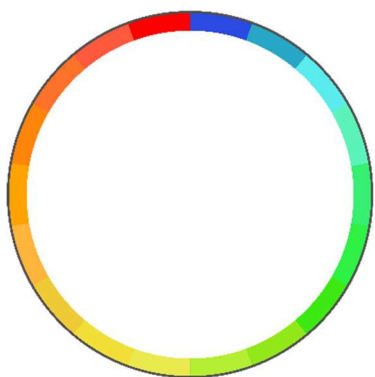
Este módulo cuenta con la característica de soportar alimentación tanto de 3.3V como de 5V, por lo que no hubo necesidad de otra fuente adicional a las que ya teníamos. Este módulo permite establecer una comunicación entre una aplicación móvil que haga uso del protocolo Bluetooth con el microcontrolador. Se optó por el HC-06 debido a que este actúa únicamente como esclavo, es decir, no puede mediar conexiones de otros módulos, solo puede obedecer al master (el módulo bluetooth del teléfono) y recibir o transmitir información, que era el propósito de este módulo, por lo que el HC-05 estaría muy desaprovechado al tener capacidad de ser master y no solo esclavo.

Finalmente, el valor transmitido se recibe por medio de la aplicación móvil y se transforma en un valor de temperatura mediante la siguiente relación, que fue explicada en la parte del ADC:

$$Salida = \frac{50V_{in}}{1023}$$


Para el desarrollo de la aplicación móvil, se trabajo mediante Android Studio, el IDE que nos proporciona Google como método ideal de desarrollo de software para la plataforma móvil Android; Para este proyecto decidimos trabaja con el SDK de Google número 23, usando las API que vienen de forma predeterminada de BlueTooth.

Para poder obtener una mejor experiencia de usuario, realizamos 2 clases, las cuales llamaremos “ConectaSensor.java” y “VerTemperatura.java” sin mencionar clases que se tuvieron que modelar para el correcto funcionamiento de la aplicación, pero por ser clases con el mero objetivo de hacer de la app una mejor experiencia de usuario, solo nos concentraremos en las primeras dos.

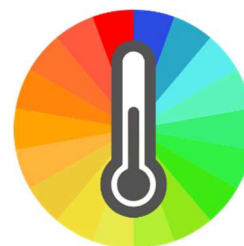


Para esta app, desarrollamos algunos recursos visuales:

Para la visualización de la temperatura, creamos un círculo con 20 divisiones para representar una temperatura a 360° donde 0°C corresponde a un Angulo de 0° desde el centro del círculo, los 50° corresponden a 359°.

Además, usamos esta flecha  como el indicador de la temperatura a la nos encontramos, y su giro esta determinado por la información que recibimos del sensor de temperatura mediante el módulo BT.

De igual modo, desarrollamos un ícono de app, para nuestra app llamada “Sensor de Temperatura” con la idea de facilitar el encuentro de la aplicación a los usuarios finales.



A continuación podremos ver algunas partes del código de la aplicación, que creemos relevantes para conocer el funcionamiento de la misma, de manera adicional anexaremos un vínculo que tendrá una vida de 2 años a partir de que este proyecto fuese entregado, desde el cual se podrá descargar el proyecto completo de la app, con el objetivo de que pueda ser estudiada para comprender su total funcionamiento:

//Funcion para visualizar los dispositivos vinculados por BT

```
private void fillDevicesView(){
    ArrayList<DeviceData> filtered = new ArrayList<>();
    for (DeviceData item : getApp().getSettings().getDevices()){
        if (!needAddToFiltered(item))
            continue;
        String name = item.getName() + "";
        if (searchFilter == null || searchFilter.isEmpty() ||
            name.toLowerCase().contains(searchFilter.toLowerCase())){
            if (getApp().showNoServicesDevices)
                filtered.add(item);
            else{
                boolean hasServices = item.getUids().size() > 0;
                if (hasServices)
                    filtered.add(item);
            }
        }
    }
    DevicesRowAdapter adapter = new DevicesRowAdapter(this, filtered,
        getApp().getSettings().getSortType());
    devicesView.setAdapter(adapter);
    String title = String.format("%s, %d/%d", getResources().getString(R.string.app_name),
        filtered.size(), getApp().getSettings().getDevices().size());
    setTitle(title);
    getApp().saveSettings();
}
```

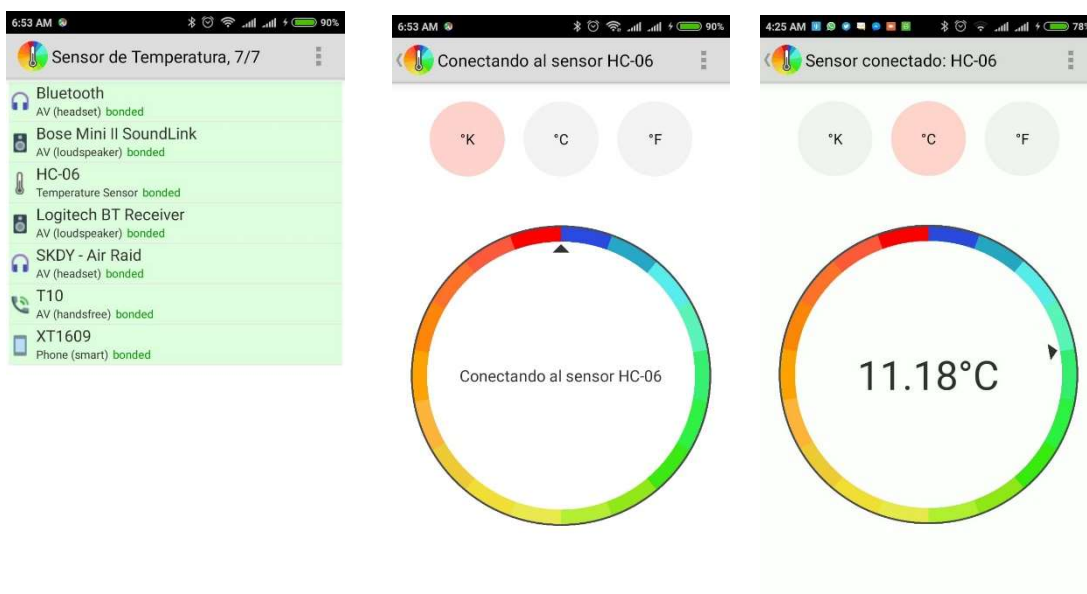
//Funcion para clasificar los dispositivos BT

```
private boolean needAddToFiltered(DeviceData item){
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.AUDIO_VIDEO &&
    getApp().showAudioVideo)
        return true;
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.COMPUTER &&
    getApp().showComputer)
        return true;
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.HEALTH &&
    getApp().showHealth)
        return true;
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.IMAGING &&
    getApp().showImaging)
        return true;
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.MISC &&
    getApp().showMisc)
        return true;
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.NETWORKING &&
    getApp().showNetworking)
        return true;
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.PERIPHERAL &&
    getApp().showPeripheral)
        return true;
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.PHONE &&
    getApp().showPhone)
        return true;
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.TOY &&
    getApp().showToy)
        return true;
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.TEMPERATURE_SENSOR
    && getApp().showTemperatureSensor)
        return true;
    if (item.getMajorDeviceClass() == BluetoothClass.Device.Major.WEARABLE &&
    getApp().showWearable)
        return true;
    return false;
}
```

//Constructor de nuestra primera clase

```
protected void onCreate(Bundle savedInstanceState){
    super.onCreate(savedInstanceState);
    if (D) Log.d(TAG, "+++ ON CREATE +++");
    setContentView(R.layout.activity_main);
    devicesView = (ListView)findViewById(R.id.devicesView);
    btnSearchDevices = (Button)findViewById(R.id.btnSearchDevices);
    ImageButton btnClearSearchBox = (ImageButton) findViewById(R.id.btnClearSearchBox);
    btnClearSearchBox.setOnClickListener(btnClearSearchBoxClick);
    ImageButton btnFilter = (ImageButton) findViewById(R.id.btnFilter);
    btnFilter.setOnClickListener(btnFilterClick);
    registerForContextMenu(btnFilter);
    searchBox = (EditText)findViewById(R.id.searchBox);
    searchBox.addTextChangedListener(searchBoxTextChangedListener);
    devicesView.setOnItemClickListener(devicesViewItemClick);
    registerForContextMenu(devicesView);
    btnSearchDevices.setEnabled(false);
    btnSearchDevices.setOnClickListener(btnSearchDevicesClick);
    createLoadingDialog();
    loadSettings();
    if (getApp().getAdapter() == null){
        showAlert(getResources().getString(R.string.no_bt_support));
    }
}
```

Mediante todo el código anterior, podemos realizar la visualización de nuestros dispositivos vinculados y habremos creado nuestro propio tipo de dispositivo llamado “Sensor de Temperatura” o “Temperature Sensor”.



//Función correspondiente a mostrar la temperatura en la graduación deseada

```
private void convertCommand(String command, int messageType) {
```

```
    try {
```

```
        Float corrTemp = Float.parseFloat(command);
```

```
        prox = corrTemp;
```

```
        rotateTemperature();
```

```
        act = prox;
```

```
        StringBuilder sb = new StringBuilder();
```

```
        String tempo;
```

```
        float res;
```

```
        switch (grad) {
```

```
            case 1:
```

```
                res = 323.15f / 1024.00f * corrTemp;
```

```
                tempo = String.format("%.2f", res);
```

```
                sb.append(tempo);
```

```
                sb.append("°K");
```

```
                break;
```

```
            case 2:
```

```
                res = 50.00f / 1024.00f * corrTemp;
```

```
                tempo = String.format("%.2f", res);
```

```
                sb.append(tempo);
```

```
                sb.append("°C");
```

```
                break;
```

```
            case 3:
```

```
                res = 122f / 1024.00f * corrTemp;
```

```
                tempo = String.format("%.2f", res);
```

```
                sb.append(tempo);
```

```
                sb.append("°F");
```

```
                break;
```

```
        }
```

```
        temperatura.setTextSize(50);
```

```
        temperatura.setText(sb.toString());
```

```
    } catch (NumberFormatException e) {
```

```
        temperatura.setTextSize(20);
```

```
        temperatura.setText(command);
```

```
    }
```

```
}
```

Por otro lado, tenemos aquí el código correspondiente a la visualización de la temperatura.

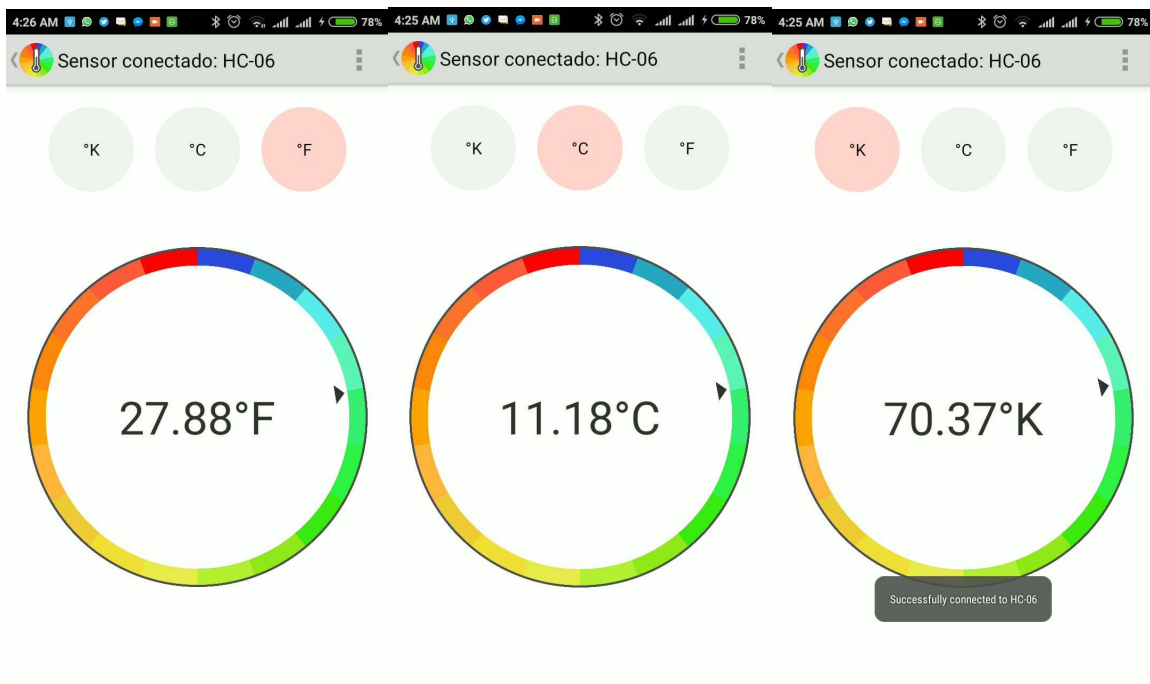
```
//Para establecer conexion con el sensor BT
public void onStart(){
    super.onStart();
    if (D) Log.d(TAG, "++ ON START ++");
    if (!getApp().getAdapter().isEnabled()){
        if (D) Log.d(TAG, "++ ON START BT disabled ++");
        Intent enableIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(enableIntent, MainActivity.REQUEST_ENABLE_BT);
    }
    else{
        if (D) Log.d(TAG, "++ ON START BT enabled ++");
        if (getApp().getConnector() == null){
            if (D) Log.d(TAG, "++ ON START setupConnector() ++");
            setupConnector();
        } else {
            if (D) Log.d(TAG, "++ ON START ++, connector state " +
getApp().getConnector().getState());
        }
    }
}
```


//Para cambiar la seleccion de unidad de temperatura

```
final Button btnKelvin = (Button) findViewById(R.id.btnKelvin);
final Button btnCelsius = (Button) findViewById(R.id.btnCelsius);
final Button btnFahrenheit = (Button) findViewById(R.id.btnFahrenheit);
btnCelsius.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        btnCelsius.setBackgroundResource(R.drawable.round_button_selected);
        btnFahrenheit.setBackgroundResource(R.drawable.round_button);
        btnKelvin.setBackgroundResource(R.drawable.round_button);
        grad=2;
    }
});
btnFahrenheit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        btnCelsius.setBackgroundResource(R.drawable.round_button);
        btnFahrenheit.setBackgroundResource(R.drawable.round_button_selected);
        btnKelvin.setBackgroundResource(R.drawable.round_button);
        grad=3;
    }
});
btnKelvin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        btnCelsius.setBackgroundResource(R.drawable.round_button);
        btnFahrenheit.setBackgroundResource(R.drawable.round_button);
        btnKelvin.setBackgroundResource(R.drawable.round_button_selected);
        grad=1;
    }
});
```

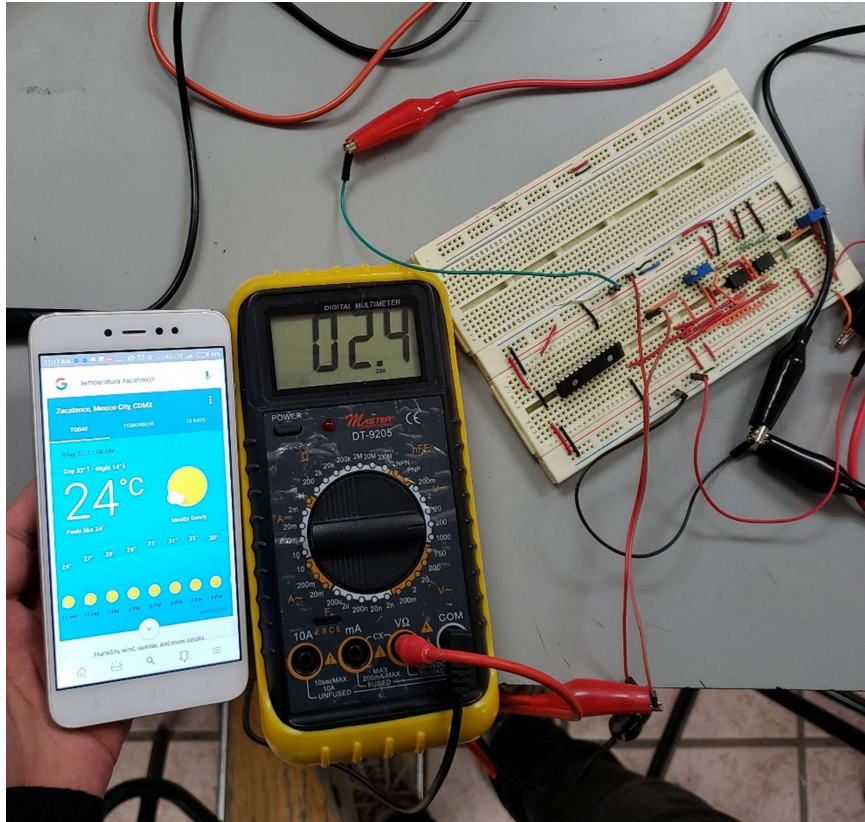
//Funcion para girar la flecha del sensor de temperatura

```
public void rotateTemperature(){  
    long delay = 798;  
    float pl, pF;  
    pl = 360.00f*act/1024.00f;  
    pF = 350.00f*prox/1024.00f;  
    ObjectAnimator rotateAnimation = ObjectAnimator.ofFloat(rotCircle, "rotation", pl,  
pF);  
    rotateAnimation.setInterpolator(new LinearInterpolator());  
    rotateAnimation.setDuration(delay);  
    AnimatorSet animatorSet = new AnimatorSet();  
    animatorSet.playTogether(rotateAnimation);  
    animatorSet.start();  
  
    rotateAnimation = ObjectAnimator.ofFloat(rotCircle, "rotation", pF, pF);  
    rotateAnimation.setDuration(1);  
    animatorSet = new AnimatorSet();  
    animatorSet.playTogether(rotateAnimation);  
    animatorSet.start();  
}
```

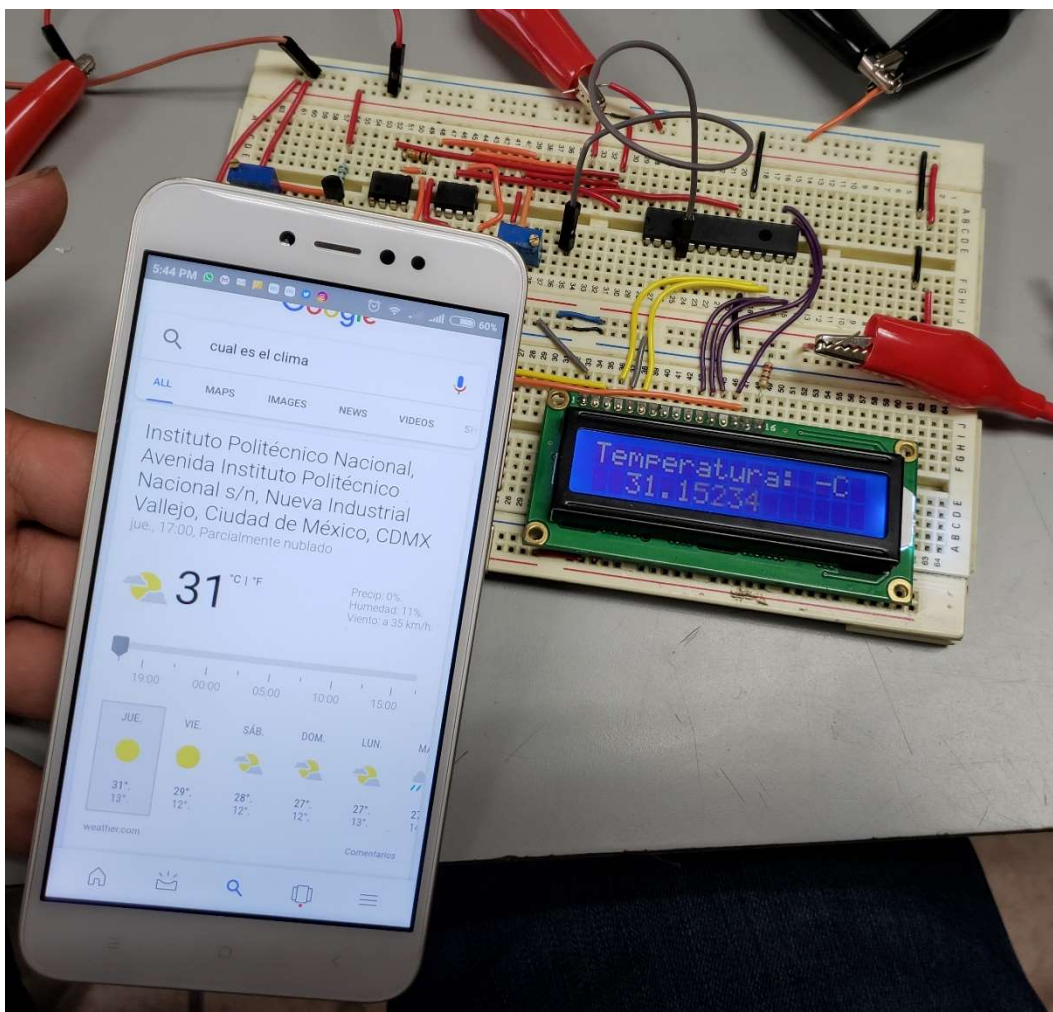


Evidencias de las mediciones y pruebas

El circuito se probó primeramente sin el Bluetooth y sin la pantalla LCD, es decir, primeramente se probó el CAS. Para una temperatura en el ambiente de 24°C se obtuvieron los siguientes resultados a la salida, del CAS, ya habiendo realizado la correspondiente calibración del LM335:

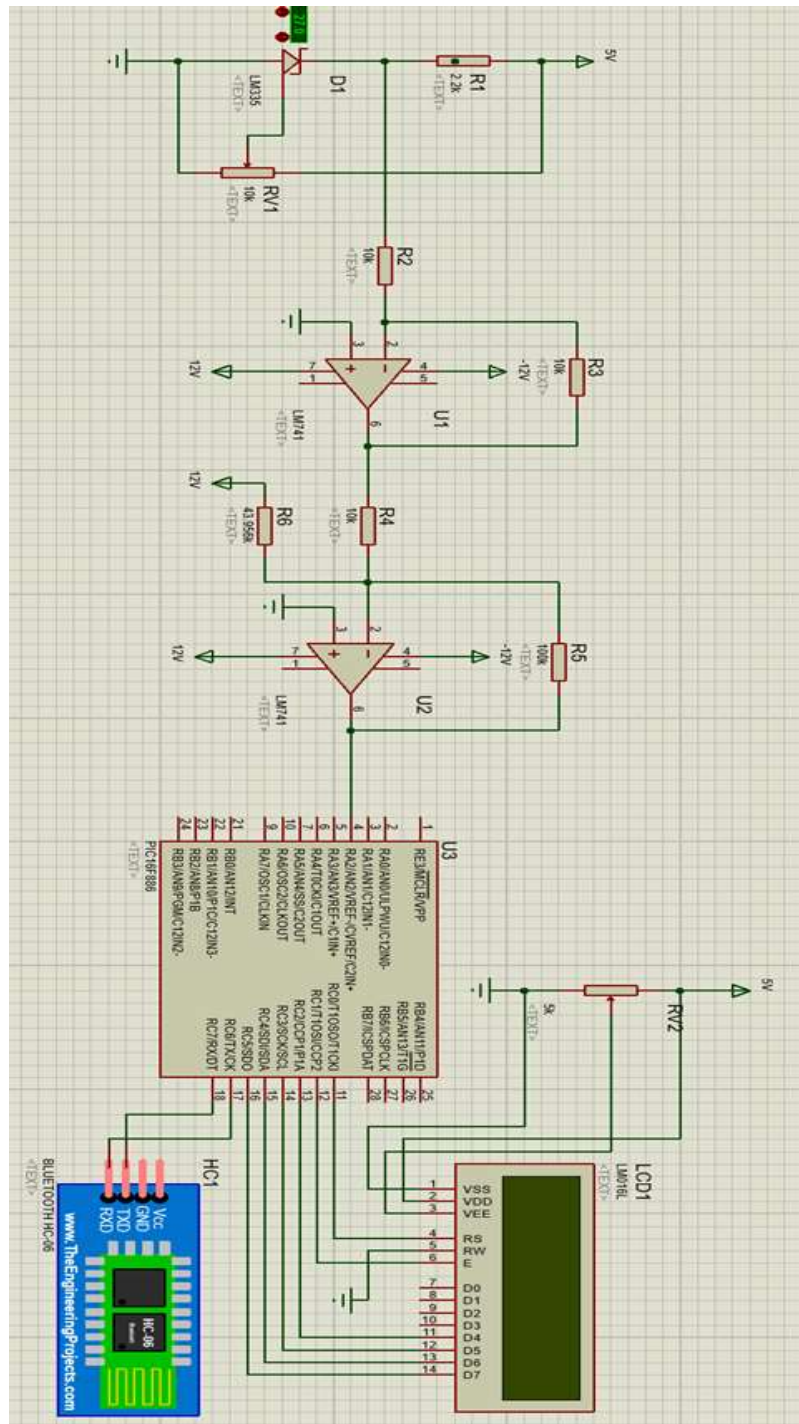


Ya habiendo comprobado el correcto funcionamiento del CAS se procedió a conectar las siguientes etapas, en concreto, el microcontrolador y la pantalla LCD para la visualización de temperatura, al haberse realizado otro día la temperatura fue diferente en esta ocasión, sin embargo, el resultado mostrado en la LCD coincidía con la temperatura del laboratorio en ese momento:



Los resultados de estas mediciones se proporcionan más adelante en la sección de datos. Las imágenes fungen únicamente como evidencia del trabajo realizado en el laboratorio.

Circuito esquemático final



Datos

Datos teóricos

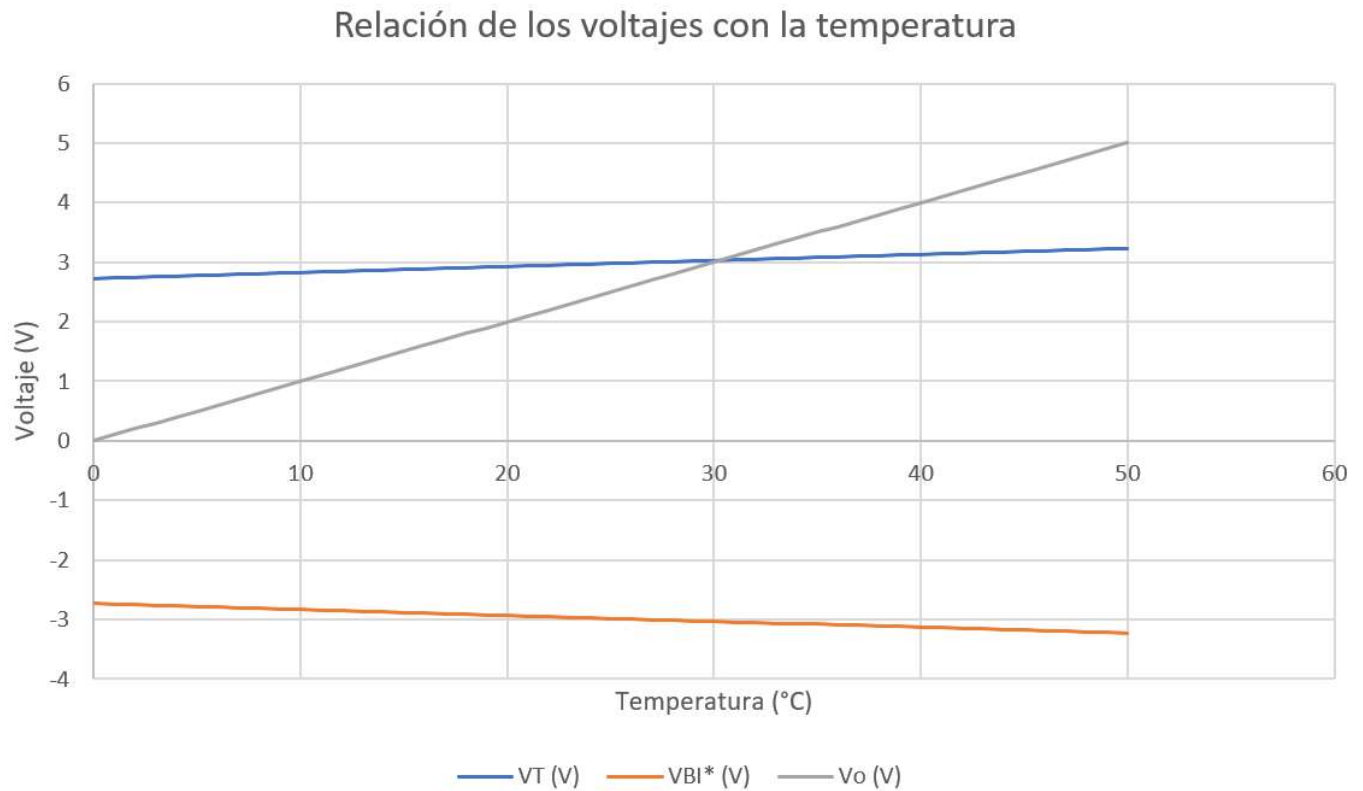
Temperatura (°C)	VT (V)	VBI* (V)	Vo (V)
0	2.73	-2.73	0
1	2.74	-2.74	0.1
2	2.75	-2.75	0.2
3	2.76	-2.76	0.3
4	2.77	-2.77	0.4
5	2.78	-2.78	0.5
6	2.79	-2.79	0.6
7	2.8	-2.8	0.7
8	2.81	-2.81	0.8
9	2.82	-2.82	0.9
10	2.83	-2.83	1
11	2.84	-2.84	1.1
12	2.85	-2.85	1.2
13	2.86	-2.86	1.3
14	2.87	-2.87	1.4
15	2.88	-2.88	1.5
16	2.89	-2.89	1.6
17	2.9	-2.9	1.7
18	2.91	-2.91	1.8
19	2.92	-2.92	1.9
20	2.93	-2.93	2
21	2.94	-2.94	2.1
22	2.95	-2.95	2.2
23	2.96	-2.96	2.3
24	2.97	-2.97	2.4
25	2.98	-2.98	2.5
26	2.99	-2.99	2.6
27	3	-3	2.7
28	3.01	-3.01	2.8
29	3.02	-3.02	2.9
30	3.03	-3.03	3
31	3.04	-3.04	3.1
32	3.05	-3.05	3.2
33	3.06	-3.06	3.3
34	3.07	-3.07	3.4
35	3.08	-3.08	3.5
36	3.09	-3.09	3.6
37	3.1	-3.1	3.7
38	3.11	-3.11	3.8
39	3.12	-3.12	3.9
40	3.13	-3.13	4

41	3.14	-3.14	4.1
42	3.15	-3.15	4.2
43	3.16	-3.16	4.3
44	3.17	-3.17	4.4
45	3.18	-3.18	4.5
46	3.19	-3.19	4.6
47	3.2	-3.2	4.7
48	3.21	-3.21	4.8
49	3.22	-3.22	4.9
50	3.23	-3.23	5

*VBI = Voltaje del buffer inversor

Estos valores teóricos se obtuvieron de evaluar los valores de temperatura a la ecuación característica del sensor, lo cual nos da su voltaje de salida V_T , mismo que se introduce a la ecuación del CAS, por lo que todos los valores obtenidos son los ideales.

A continuación, se muestra una gráfica en la que se comparan los voltajes de cada etapa del CAS respecto a la temperatura:



En esta gráfica se aprecia que la recta de VT es de una pendiente muy chica, prácticamente una línea recta, mientras que, para el VBI, es decir, el voltaje del buffer inversor se tiene lo mismo que en el sensor VT, pero invertido, de modo que conserva su pendiente. Esto nos permite comprender parte del CAS, partiendo de su entrada, que es el voltaje del sensor, VT, y que varía de acuerdo a la temperatura del ambiente hasta la parte del buffer inversor, que recibe esta señal y la invierte, es decir, ahora tendremos exactamente el mismo voltaje pero negativo, mismo que se introduce ahora al sumador inversor, donde se obtiene el comportamiento deseado para el CAS, resultando en la recta de color gris, la cual es la más ideal y que se comporta de acuerdo a la temperatura con una relación de 1/10, es decir, el voltaje de salida es una décima parte del valor numérico de la temperatura.

La diferencia se ve notable con el voltaje de salida del CAS Vo, pues este presenta una forma totalmente lineal, que parte desde 0V a una temperatura de 0°C y 5V para 50°C. Eso nos ayuda a contrastar y a ver porque fue necesario realizar el CAS para obtener una gráfica donde el voltaje sea lo más lineal posible respecto a la temperatura.

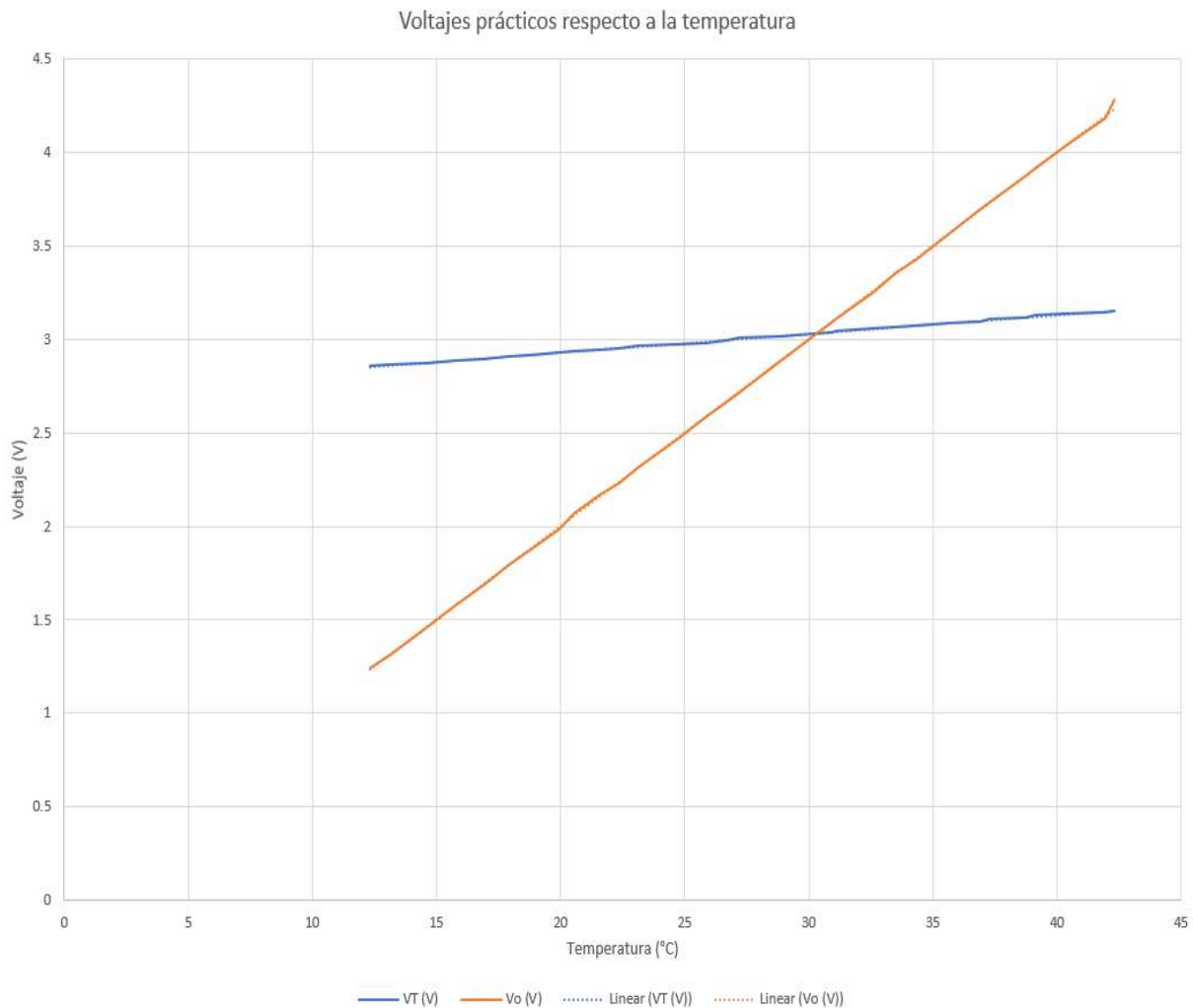
Datos prácticos

Temperatura (°C)	VT (V)	Vo (V)
12.345	2.858	1.244
13.124	2.863	1.313
14.67	2.872	1.467
15.85	2.884	1.586
16.88	2.895	1.686
17.901	2.906	1.791
18.81	2.919	1.88
19.974	2.931	1.989
20.568	2.938	2.073
21.621	2.944	2.171
22.341	2.953	2.231
23.124	2.964	2.314
24.768	2.97	2.471
25.871	2.981	2.588
26.693	2.993	2.668
27.189	3.008	2.718
28.894	3.014	2.887
29.523	3.022	2.95
30.931	3.041	3.092
31.098	3.045	3.107
32.578	3.057	3.249
33.451	3.064	3.351
34.273	3.073	3.425
35.649	3.088	3.57
36.912	3.093	3.693
37.267	3.108	3.731
38.75	3.113	3.877
39.156	3.127	3.914
40.387	3.135	4.041
41.951	3.146	4.181
42.286	3.153	4.281

Para poder medir estos datos prácticos fue necesario emplear 2 multímetros simultáneamente, uno conectado a la salida del CAS, y otro a la salida del sensor LM335. Además, se empleó un encendedor que se posicionaba a una distancia relativamente cercana al sensor, lo suficiente para que la temperatura se incrementará pero de manera muy lenta, para temperaturas inferiores se empleó aire comprimido que nos compartieron unos compañeros en el laboratorio, sin embargo, aún con esto no fuimos capaces de llegar a menos de 12°C, por otro lado, decidimos no superar los 42°C para no correr el riesgo de llegar a 50°C o más y dañar el ADC del microcontrolador.

Se puede apreciar que se dieron lecturas más precisas de la temperatura, y estas se obtuvieron de la visualización de la temperatura en la pantalla LCD.

A simple vista se puede observar que existieron pequeñas desviaciones, por lo que no se obtuvo una línea recta perfecta como en los datos teóricos, sin embargo, para ser una medición bastante sensible se obtuvieron resultados muy cercanos a los ideales, el enfriamiento fue lo más difícil por momentos, pues se debió dosificar la cantidad de aire, y complementarlo con un recipiente de refresco frío que teníamos a la mano.



En esta gráfica se puede apreciar un comportamiento que, si bien no es perfectamente lineal, representa casi de manera ideal el comportamiento deseado del CAS, vemos como hay pequeñas desviaciones de la recta ideal que se denota por las líneas punteadas, pero nada que indique un comportamiento errático en el CAS, por lo que se pudo llegar a la conclusión de que este es apto para ser introducido al ADC y es capaz de darnos un dato fiable de la temperatura actual.

Conclusiones

Damián Alanís Ramírez

Con este proyecto se logró el objetivo, ya que se logró diseñar y armar el circuito que es capaz de mostrar la temperatura implementando un sensor de temperatura LM335, circuitos acondicionadores de señal, así como elementos como el convertidor analógico a digital y la comunicación serial de un microcontrolador PIC16F886. En todo momento se aplicaron conocimientos tanto teóricos como prácticos adquiridos en la unidad de aprendizaje de electrónica analógica, primeramente, en el circuito de calibración del sensor, se consultó la hoja de datos, que nos da una idea de su funcionamiento, así como recomendaciones sobre sus parámetros eléctricos, de ahí obtuvimos el circuito de calibración empleado, que consistió en una resistencia de $2.2K\Omega$ hacia V_{cc} y un potenciómetro de $10K\Omega$, de esta primera etapa comprendimos la importancia de la calibración, pues antes de ser calibrado, al medir el voltaje a la salida del sensor no obteníamos el voltaje dado por la ecuación característica que se expuso en la etapa de diseño, es decir, $10mV(temperatura) + 2.73V$, sino que nos salíamos incluso de los $3.23V$, sin tener justamente los $50^{\circ}C$ que nos otorgan ese valor, por lo que emplear el sensor sin circuito de calibración representa el riesgo de obtener mediciones que no son precisas.

La etapa de diseño del CAS fue fundamental para el proyecto, empezando por la determinación de la ecuación del mismo, mediante la recta que se forma al graficar el voltaje del sensor a cierta temperatura contra lo que se espera de esa temperatura a la salida del CAS, esto es $0V$ a $0^{\circ}C$ y $5V$ a $50^{\circ}C$, que contrastado con los $2.73V$ a $0^{\circ}C$ que se obtienen con el sensor calibrado y $3.23V$ a $50^{\circ}C$ nos arroja que la ecuación es $V_o = 10V_T - 27.3V$, donde V_T es la salida del sensor. Esta ecuación nos permitió empezar a ver que esto podría modelarse con un sumador, pero para tener un miembro de la ecuación de la señal positivo y otro negativo pensamos en incluir un amplificador inversor o buffer inversor, el cual es un circuito que emplea conceptos vistos en clase sobre amplificadores operacionales, sobre todo el concepto de ganancia, que en este caso es de -1 , nos permite acoplar las impedancias al mismo tiempo que nos permite invertir la señal (que posteriormente será reinvertida en el sumador inversor). Para diseñar este buffer inversor se consideró que la ganancia de un amplificador inversor está dada por la R_F entre la resistencia que se tenga en la entrada inversora, esto es, si queremos una ganancia de -1 , tenemos que poner ambas resistencias del mismo valor. Se comprobó experimentalmente que si se obtenía la misma salida del sensor, pero negativa, de modo que el buffer inversor realizó adecuadamente su función, y preparaba la señal para la siguiente etapa.

La etapa del sumador inversor fue de las más completas en cuestión de aplicación de conocimientos, pues recibe la señal del buffer inversor, pero contamos con un arreglo de dos resistencias y 2 diferentes voltajes a la entrada, de modo que tendremos la salida que estará dada por $V_o = -[(R_f/R_1)V_1 + (R_f/R_2)V_2]$, considerando que nuestro V_1 es el V_T del sensor pasado por el buffer, es decir, un voltaje negativo, tendremos una ecuación de la forma $V_o = V_T R_f/R_1 - V_2 R_f/R_2$, lo cual puede tomar la forma de la ecuación que se obtuvo en el diseño del CAS, fue así como obtuvimos los valores de las resistencias R_1 y R_2 , proponiendo una R_f , de modo que se obtuviera ese comportamiento. La salida de este sumador es ya, directamente, la salida del CAS, sin necesidad de otra etapa, por lo que se pasa a la siguiente etapa.

El convertidor analógico a digital empleado fue el del microcontrolador PIC16F886, que cuenta con una resolución de 10 bits, es decir, que para valores de voltaje entre 0 y $5V$ obtenemos un número entre 0 y 1023 , que luego podemos pasar al valor de temperatura aplicando la relación matemática donde $5V$ es el valor numérico de 1023 , y obteniendo el valor de voltaje para determinado número de bits se puede multiplicar por 10 (considerando que para nuestra ecuación, la salida es lineal respecto a la temperatura a razón de una décima parte) para obtener la temperatura en $^{\circ}C$, este valor se transmite después por

medio del protocolo de comunicación Bluetooth, empleando primeramente la comunicación serial con la que cuenta el microcontrolador, esto haciendo uso del módulo Bluetooth HC-06, que sirve únicamente como esclavo, es decir, no es capaz de mediar la comunicación con otros módulos, solo ejecuta las instrucciones que recibe, es decir, transmisión o recepción de datos.

Los datos provenientes del microcontrolador, y que representan ya el paso de la señal analógica acondicionada a un dato de tipo digital son recibidas por una aplicación móvil, programada en el lenguaje de programación Java haciendo uso del IDE Android Studio, esta aplicación se programó para mostrar una interfaz amigable al usuario, que permitiera realizar la conexión al módulo bluetooth de forma sencilla y, ya conectado, conocer la temperatura en °C, °F y °K, contando con un indicador visual de frío a caliente, donde 0°C es el límite inferior y 50°C el límite superior.

Este proyecto nos dejó mucho aprendizaje y experiencia tanto en la teoría como en la práctica en lo que a diseño de circuitos de electrónica analógica se refiere, e incluso fuimos un paso más adelante, pues transformamos estas señales analógicas a digitales, que luego se pueden tratar por una computadora, que en este caso fue un microcontrolador. Cabe mencionar que este proyecto se presentó en Expo-ESCOM 2018 con una buena recepción por parte de los visitantes a nuestro espacio en la exposición, por lo que nos llevamos una gran satisfacción de haber aplicado los conceptos más importantes de la materia de electrónica analógica en un trabajo que nos permitió comprender el estrecho vínculo entre la parte analógica de la electrónica y la digital y como ambas pueden trabajar en conjunto para facilitar tareas y hechos que damos por hecho, como el saber la temperatura en un momento dado.

Víctor Ángel Carmona Medina

Con esta práctica pude notar como es el funcionamiento del circuito CAS que para empezar necesitábamos de un sensor que bien pudimos usar un LM35 pero por cuestiones de precisión usamos un LM335 esto podemos notar con una gráfica donde si comparamos la inclinación veremos que el del 335 será mayor y esto podemos interpretar como mayor índice de sensibilidad.

En la primera etapa del circuito al conectar el sensor a este entra una temperatura y este transformara esta temperatura en un voltaje de salida que se puede calcular fácilmente ya que este sensor tiene un funcionamiento lineal solo es necesario en esta casa particular multiplicar por $273 \times 10\text{mV}$ y a eso aumentarle 10mV por cada grado Celsius sin embargo el resultado medido fue con un error de un grado a temperatura ambiente.

Pero esto es solo el sensor bien pudimos solo conectar el sensor a un Arduino y pasar el dato a este y mandarlo al celular pero para manera más práctica y ver cómo funciona un termómetro de manera analógica conectaremos lo que es un CAS (Circuito de Acondicionamiento de Señal) donde utilizaremos a la entrada de este u seguidor de voltaje esto es para eliminar efectos de carga de CA que contengan nuestro circuito previo este CAS con efectos más prácticos y a diferencia de nuestros compañeros usamos un sumador inversor esto nos permitió ahorrarnos un operacional extra que digo no es mucho dinero pero aun así dinero es dinero, con respecto a los operacionales usado usamos los más básico y más comerciales que son los LM741 como no tendríamos una frecuencia mayor a 1MHz eran buenos para la práctica pero existen mejores operacionales en el mercado.

Para tener una ganancia mayor o disminuir es necesario tener una resistencia mayor a otra es decir que nuestra resistencia de retro alimentación debe ser mayor a la que va a tierra esto aumentara la ganancia

o si queremos que la ganancia sea igual a 1 serán del mismo valor. Esto se ve reflejado en el V_o y obtendremos un mayor voltaje de salida.

Para hacer el ajuste necesario tuvimos que comprar una resistencia de precisión de 50k que utilizamos para ajustar nuestro sensor y así estar a la temperatura ambiente esto lo hicimos viendo la temperatura actual y haciendo el cálculo necesario determinamos nuestro voltaje de salida y así hicimos nuestro ajuste en la resistencia de ganancia.

Para finalizar del ya saliendo nuestro circuito CAS lo pasamos a un ADC pero en este caso usamos un PIC estos integrados tienen la maravilla de contar con un ADC interno que lo ocupamos de acuerdo a nuestro diagrama a bloques observamos que la del nuestro ADC debe de ser de 0 a 5V y de ahí para efectos de visualización conectamos a un LCD pero la idea original es pasarlo a un módulo Bluetooth que solo se mandó un arreglo de 10 bits y del lado de la aplicación móvil recibirlos y transformar esa cadena a flotante y mostrar en la pantalla los resultados.

Eduardo Gómez Rodríguez

En definitiva, con este proyecto se cumple totalmente el objetivo de la materia, ya que debido a que para desarrollar el circuito debemos tener conocimiento sobre el funcionamiento de los materiales semiconductores tanto de tipo P como de tipo N; Pasando por componentes electrónicos como el Diodo, el Diodo Zener, el Diodo LED, así como el Transistor como el BJT, y finalmente con los operacionales, siendo este último un circuito integrado de transistores. Además del convertidor Analógico Digital, que nos permite crear un puente entre todos los datos analógicos que podemos recolectar sobre nuestro mundo y poderles dar una interpretación computable y además analizar todo lo que la información analógica nos provee.

De este modo, este proyecto fue un claro ejemplo del objetivo de nuestra carrera como Ingenieros en Sistemas Computacionales, debido a que estamos realizando un Sistema Computacional de principio a fin; Además de estar realizando ingeniería debido a que nosotros tuvimos que proponer y crear nuestro propio circuito para poder llevar el proyecto al éxito. A continuación, realizare mis conclusiones particulares a cada etapa de desarrollo del circuito.

Circuito de calibración del sensor; Fue una etapa totalmente exitosa, a mi parecer fue incluso una de las más sencillas, pero a la vez es la parte más esencial de nuestro proyecto, debido a que gracias a ella podemos verificar que los datos que nos ofrece el sensor sean los datos que nosotros estamos esperando del sensor, y de no ser así nosotros podemos calibrar el sensor para que obtengamos la lectura que deseamos; Cabe mencionar que gracias a esto, también pudimos comprender que efectos tiene la impedancia de que nuestro circuito de calibración fuese un divisor de voltaje, pero a su vez implicó tener que anexar un circuito seguidor de voltaje previo a nuestra etapa de acoplamiento de señal. Para nuestra calibración utilizamos una resistencia de 2.2 KOhms y un potenciómetro de 10 KOhms de precisión, de este modo logramos obtener lecturas entre 2.73V a 3.23V que corresponden a las medidas de 10mV por grado Kelvin que nos ofrece el fabricante y correspondiente a 0° C y 50° C respectivamente.

Circuito de Acoplamiento de señal; Al igual que el anterior fue una etapa con total éxito pero que requirió más tiempo de análisis que se ensamblaje debido a que teníamos que transformar la información en grados Kelvin que nos estaba llegando a una lectura en grados centígrados, por lo que para poder acoplar la señal se decidió usar una equivalencia de 0V para la entrada de 2.73V del sensor y de 5V para los 3.23V que nos entrega el sensor; De este modo podemos decir que la lectura del voltaje debería ser una décima parte de la temperatura en grados centígrados que estamos recibiendo, por lo que podemos deducir de el CAS ya nos entrega de manera implícita la temperatura en °C, de modo que para nosotros obtener la

lecturas en términos de 0 a 5 V, obtuvimos que la salida de nuestro CAS debería ser la siguiente: $V_o = 10V_t - 27.3V$ donde V_t es el que nos arroja la temperatura; De igual manera tuvimos que utilizar toda la parte teórica abordada por la materia para poder definir cual sería nuestro circuito y realizar unos cálculos correctos, y al igual que el CAS, tuvimos que agregar un potenciómetro para ajustar el comportamiento del circuito.

Circuito Analógico/Digital; Una etapa bastante exitosa que a pesar de ser muy sencilla debido a la utilización de un microcontrolador, el nulo conocimiento sobre ellos dificultó la realización de esta etapa, a pesar de que nuestro compañero Damián ya había tenido un acercamiento previo al PIC16F886; nunca lo había utilizado con el propósito de generar un dato digital desde uno analógico, por lo que investigando un poco el circuito resultante de programar una A/D nos encontramos con una salida digital de 10 bits, por lo que decidimos interpretarla como un número entero que podía ir desde 0 hasta 1023; De esta manera al realizar la relación, obtuvimos que para 0°C tenemos una lectura del número 0 y para 50°C el número 1023, de modo que dependiendo de la temperatura deseada podíamos realizar una conversión mediante la mencionada relación.

Visualización de la salida; Para esta etapa final decidimos dar dos salidas de usuario, una mediante una pantalla LCD con el mero objetivo de mostrar la independencia del circuito y una segunda con una aplicación móvil para celulares Android 6.0 +. Al igual que las etapas pasadas esta igualmente fue realizada con total éxito, sin embargo, requirió un nivel de abstracción distinto debido a que ya estábamos tratando prácticamente con puro software tanto de bajo como de alto nivel. Para el caso de la pantalla LCD, nuevamente usamos el PIC16F886 de modo que utilizando la ecuación $\text{TemperaturaEnCentigrados} = \frac{50}{1024} * \text{Lectura10Bits}$, el resultando es un número decimal que es convertido a cadena de caracteres y cortado a solo los primeros 4 decimales, para ser mandados vía LCD. En el caso de la aplicación Android, la parte más compleja fue establecer la conexión con el dispositivo BT, de modo que escuchamos la terminal de nuestra antena en la cual se encontraría una cadena de 4 dígitos que representaría el número entero de nuestra lectura de 10 bits; Por lo que desde la aplicación convertimos la cadena, en un número entero para poderlo usar matemáticamente como nuestra lectura; Esta parte se volvió bastante sencilla en la cuestión matemática, pero debido a la rueda gradual de colores que deseábamos usar, se convirtió en una tarea más complicada, sin embargo podemos concluir que es sumamente importante hacer uso de interfaces amigables con el usuario, debido a que es sumamente indispensable que nuestra carrera se enfoque en poder transformar datos analógicos a digitales y entregar al usuario final un contenido que para él sea sencillo y gratificante usar.

Referencias

UART Serial Communication. Obtenido de <https://www.mikroe.com/blog/uart-serial-communication>

Módulo Bluetooth HC-06. Obtenido de <https://www.prometec.net/bt-hc06/>

Boylestad R. L, Nashelsky L. *Electrónica: Teoría de circuitos y dispositivos electrónicos (décima edición)*. Pearson Education. México, 2009.