***Proyecto Microcontroladores***

***(Sistema de calefacción Utilizando Arduino)***

Fabián Andrés Bastidas Arenas

Ingeniería Electrónica y Telecomunicaciones

Universidad del Cauca

1. **Módulo De Entrada o Censado:** Para este módulo se utiliza un sensor de temperatura que nos brinda una señal análoga, que en este caso será la señal de entrada de nuestro sistema, aunque es posible que la señal transmitida por este sensor sea muy pequeña (esto con respecto a voltajes), por lo que será necesario primero amplificar la señal y así poder ser procesada con mayor facilidad por el micro controlador.

**Módulo Central:** Este módulo se encarga de realizar las operaciones entres las entradas y salidas, además de mostrarlas al usuario; por lo que se utilizarán dos componentes, la tarjeta programable Arduino UNO, con el fin de recibir la señal proveniente del sensor analógico, procesarla con valores de 0 a 1023 y poder realizar como se mencionó antes las operaciones lógicas, para así poder determinar cuándo encender el ventilador; para poder observar estos resultados y los procesos se utilizó una pantalla TFT ILI9341, en la cual se muestran los valores de entrada.

**Módulo de Salida o Actuador:** Para este módulo se utiliza un ventilador el cual recibirá la señal de salida con respecto a los resultados obtenidos en el módulo central, por lo que se encenderá o no si es necesario, es decir, si la temperatura supera los 20 grados o no, puede ser necesario utilizar una mayor señal para poner el ventilador en funcionamiento.

**Módulo De Entrada o Censado**

**Módulo Central**

**Módulo de Salida o Actuador**

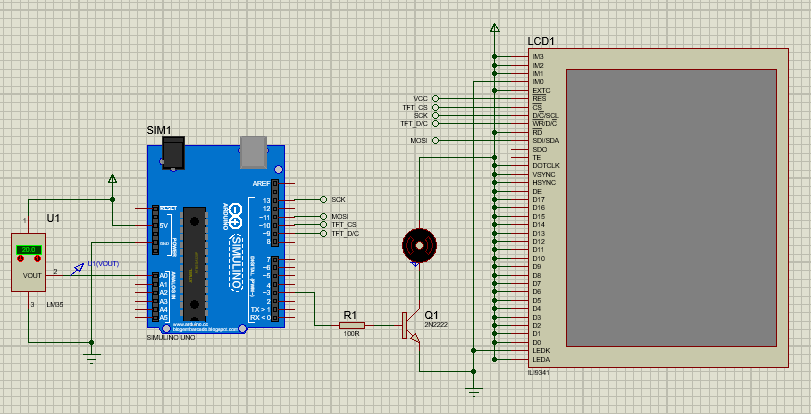
1. **Análisis y Diseño:** Para el análisis del proyecto primero se debían tener en cuenta las cosas a utilizar, como son un sensor LM35, tarjeta Arduino UNO, pantalla TFT ILI9341 y un ventilador; teniendo en cuenta esto podemos determinar los puertos a utilizar en la tarjeta, y que aquí es donde luego cargaremos nuestro programa y se encargará de utilizar cada uno de los componentes.

En primer lugar los pines a utilizar por el sensor LM35, al ser un sensor analógico, es claro que debemos elegir un puerto analógico de la tarjeta (A0, A1, A2, A3, A4, A5), para así poder transformar ese valor de entrada y realizar operaciones y procesos con él. El primer problema que se presenta es que las entradas analógicas aceptan un rango de valores de 0 a 1023, donde 0 equivale a 0V y 1023 a 5V (0 y 1 lógico); y el sensor envía una señal correspondiente con su temperatura, por ejemplo si la temperatura es 20°C el valor en voltios es 0.2V o 200mV, por lo que para obtener su valor dentro del rango se realiza una regla de tres simple, lo que nos ayudó a determinar así también el valor de referencia y así mismo obtener la temperatura tan solo dividendo entre 10 ese valor ,esto con el fin de poder mostrar dicho valor en la pantalla.

En segundo lugar los pines a utilizar por la pantalla TFT ILI9341, para la pantalla elegimos puertos o pines digitales, en este caso serán 4, los puerto 13, 11, 10 y 9 que será utilizados por los pines SCK, SDI, TFT\_CS y TFT\_D/C de la pantalla respectivamente, dentro del código en Arduino, configuraremos los colores el tamaño de la letra y la ubicación de los datos a mostrar, en este caso la temperatura obtenida por el sensor y un mensaje para cuando el ventilador esté encendido; por lo que utilizaremos las diferentes funciones para poder llevar estos datos a los pines escogidos.

Finalmente un pin o puerto digital, configurado como salida para el ventilador (puerto 3), en la lógica del código, este se encenderá o pasará de un estado de 0 a 1, si y solo si, la tempera excede el valor de la Referencia, además de en la pantalla mostrar un mensaje que dice “El ventilador está encendido!”, esto con el fin de informar al usuario que se excedió la temperatura de referencia (en este caso 20°C).

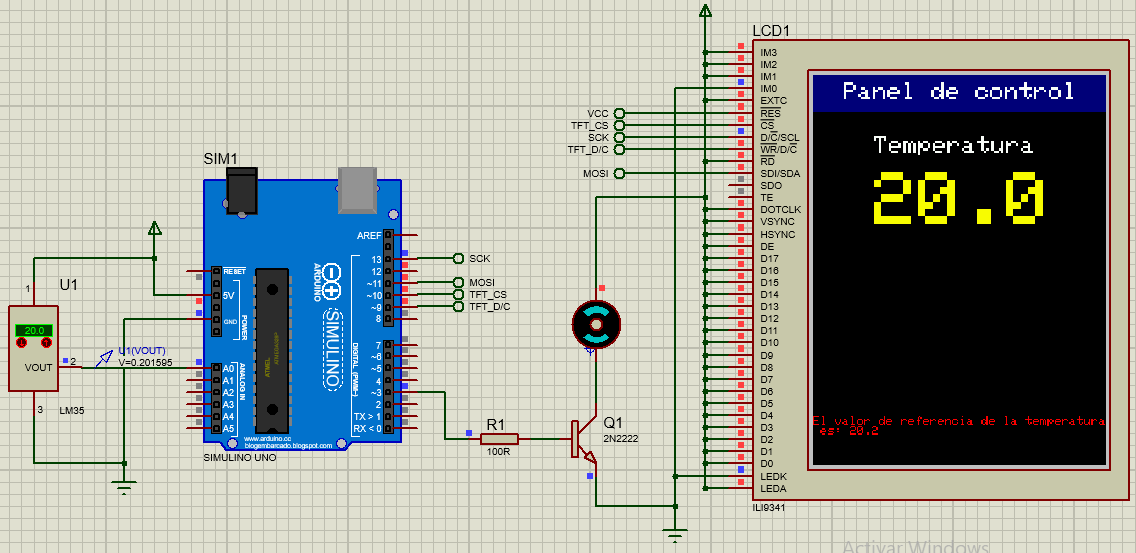
Teniendo en cuenta esto, y haciendo uso de la herramienta de simulación Proteus, se realizó el esquema del circuito y quedó así:



*Figura 1. Esquema del circuito implementado en Proteus.*

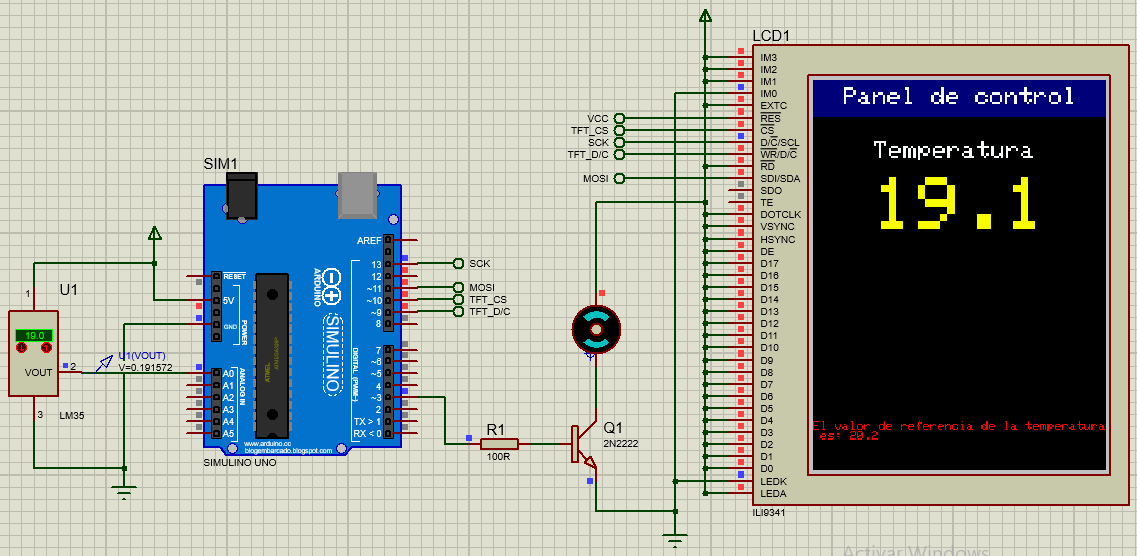
Una vez obtenido el diagrama circuital, se realizó el código en Arduino, para así poder programar la tarjeta y que se realicen las operaciones y procesos necesarios para obtener un resultado satisfactorio, para esto agregamos las librerías necesarias para utilizar la pantalla, creamos una variable de ese tipo y la inicializamos, además de esto se crearon diferentes variables que nos ayudarían a lograr el objetivo, como: SENSOR, TEMPERATURA, REFERENCIA y MOTOR, en el setup(), definimos y realizamos los procesos que solo se harán una vez dentro del programa y en el loop, los que van a ir cambiando teniendo en cuenta un delay, demora o tiempo de retardo; el código y su lógica se realizaron como se muestra en archivo “ProyectoArduinoCotrolTemperatura” adjunto al informe.

Finalmente, una vez realizada la verificación del programa creado en Arduino, se exportan los archivos binarios compilados para así poder subirlos a la tarjeta dentro de la simulación en Proteus y así poder observar el funcionamiento del circuito así:



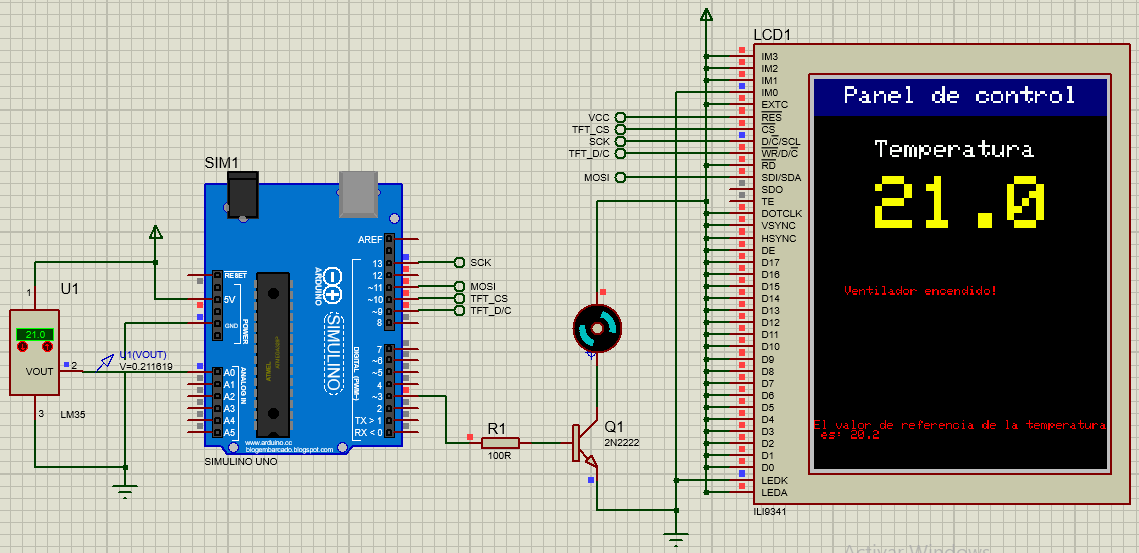
*Figura 2. Circuito en Funcionamiento con un valor de 20°C.*

Como podemos observar el figura 2, en la pantalla TFT ILI9341, tenemos la temperatura, que es exactamente la misma que la que se observa en el sensor; además de esto como se puede ver el pin 3 que fue el seleccionado para poder encender el motor, se encuentra apagado o en 0 por lo que el motor no está funcionando y se evidencia que mientras la temperatura sea igual o inferior a 20°C, el ventilador se mantendrá apagado, como también podemos observar en la figura 3.



*Figura 3. Circuito en Funcionamiento con un valor de 19°C.*

Como podemos observar en la figura 4, una vez el sensor supera la temperatura de 20°C, el puerto 3 pasa de 0 a 1 y se enciende el ventilador, además se muestra en la pantalla u mensaje que indica que el ventilador se encuentra encendido y se mantendrá así hasta que la temperatura baje.



*Figura 4. Circuito en Funcionamiento con un valor de 21°C y Ventilador Encendido.*

En conclusión, la implementación y diseño de este programa y circuito fue considerablemente más sencilla, que si se utilizara un microntrolador PIC16F877a y el lenguaje de ensamblador, ya que Arduino facilita bastante las cosas al utilizar el lenguaje de programación C.