**IMPLEMENTATION OF A TEMPERATURE MEASUREMENT DEVICE**

***Version 1.0***

**Produced by: Daniel Felipe Barón Espitia**

# OBJECTIVE

To present the procedure for designing a device capable of measuring temperature using the LM35 integrated circuit and the Arduino programmable board. In addition, to provide the necessary code to connect the integrated circuit to the programmable board and ensure the correct data collection.

# SCOPE

To provide the necessary procedure to design a device capable of measuring temperatures in a range from -55°C to 150°C with an accuracy of 0.5°C.

# DESIGN SPECIFICATIONS AND RESTRICTIONS

The temperature measurement device uses an LM35 integrated circuit, an LCD, and an Arduino programmable board. The devices mentioned above are described below:

**INTEGRATED CIRCUIT**

The LM35 integrated circuit is a temperature sensor with a voltage output that is linearly proportional to the Celsius temperature. This means that the linear increase in the sensor's voltage output corresponds to a linear increase in the measured temperature.

Figure 1 shows the pin configuration of the LM35 integrated circuit. However, it is important to review the datasheet of the integrated circuit to be used to avoid incorrect connections and therefore malfunctioning of the device.

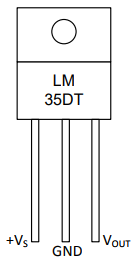
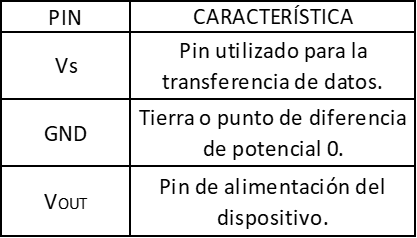


Figure 1: Pin configuration of the LM35 integrated circuit.

Table 1: Description of the pins of the LM35 integrated circuit.



The data provided by the LM35 is not temperature data but voltage data. Therefore, a conversion is necessary to present the information in temperature units. To do this, it is necessary to use the information provided by the manufacturers, which can be found in the datasheet of our device.

Finally, it is important to note that to take positive values, another topology in the circuit is necessary, but in this case, only emphasis will be placed on the collection of positive temperature data.

## PROGRAMMABLE BOARD

The programmable board used in the device design is an Arduino Mega 2560. However, it is possible to use any other version of this hardware because only the digital pins, analog pins, power pins, and GND pins are required for the implementation of the sensor, and all boards have these types of pins. The variations between the different Arduino models are the number of pins, the size, and the libraries that can be used, but all have a basic line in terms of functionality. The main feature of the Arduino Mega 2560 is its large number of pins (53 digital pins, 15 analog pins, among others).

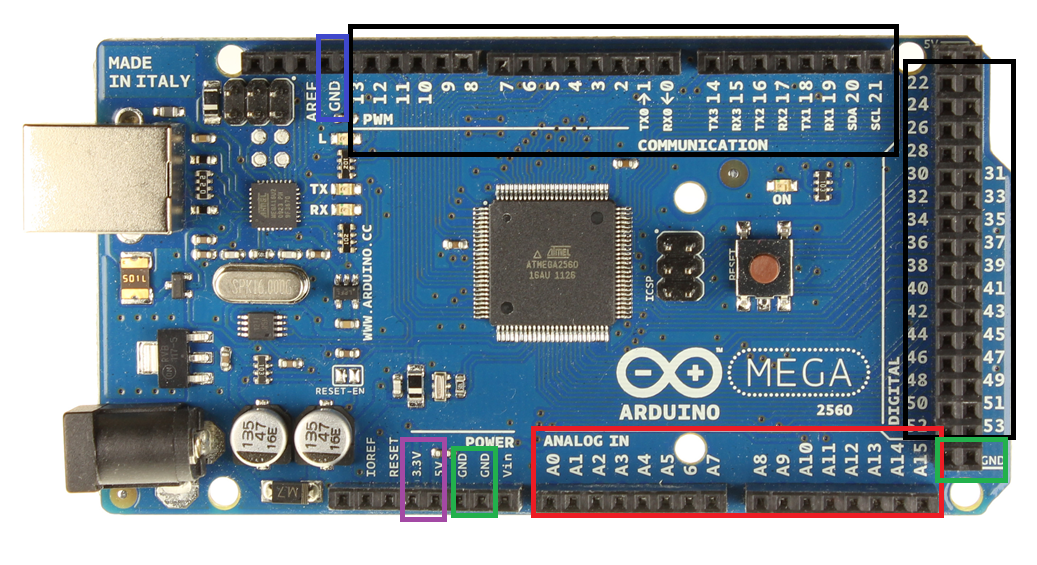


Figure 2: Schematic of the different pins that an Arduino Mega 2560 has.

**Digital Pins (Black Box):** A digital signal is one that can only take two values, 1 or 0 in the case of Arduino pins. These pins are used when the state of a device can only take two values, on or off. Usually, these pins are used to send information, such as changing the state of an LED from on to off or vice versa.

**Analog Pins (Red Box):** An analog electrical signal is one in which the voltage values vary continuously and can take any value. Therefore, these pins are useful when the data collection or the external information received takes different values, such as the temperature sensor that will be used. Usually, devices that send information are connected to these pins, and Arduino receives and processes it.

**Power pins (purple box):** These pins are used to power external devices, such as LCD screens, LEDs, among others. The Arduino Mega 2560 has two power pins, one that handles a voltage of 3.3V and another that handles a voltage of 5V. It is important to check the manufacturer's datasheet if you want to connect any external device, in order to avoid damage or malfunction of the corresponding devices.

**GND pins:** Ground pin or zero voltage point.

## LCD (Liquid Crystal Displays)

Liquid crystal displays (LCDs) offer a very fast and visually appealing way to display information in text form. The most common character LCDs are 4-bit or 8-bit, depending on the number of cables (bits) they need to have connected to the circuit in order to receive or send data.

Each LCD model is different, so it is very important to consult its specific datasheet to distinguish the different connection pins it offers and their general characteristics. However, the most common thing is that a standard LCD offers a pin to receive power and another pin to connect the screen to ground, a pin to adjust the screen contrast, three control pins generally marked as "RS", "EN" and "RW", several pins to set up parallel communication lines, and finally, two exclusive pins for the backlight circuit. (Artero, O. T., 2013).

****

Figure 3: Representation of the pins of a 16x2 4-bit LCD.

Table 2: Description of the pins of a 16x2 4-bit LCD.

|  |  |
| --- | --- |
| **PIN** | **CHARACTERISTIC** |
| GND | Ground or zero voltage point. |
| VCC | Power pin of the device. |
| Vo | Pin for setting the LCD contrast. |
| RS | Pin that serves for the microcontroller to tell the LCD whether it wants to display characters or send control commands. |
| R/W | Pin to determine whether the screen will provide or receive information. |
| E | Pin to alert the LCD that the microcontroller is going to send data. |
| D0-D7 | Used to set up parallel communication lines for transferring data and control commands from the Arduino board to the LCD. |
| A | Pin used to control the backlight, this pin is connected to Voltage, usually 5V. |
| K | Pin used to control the backlight, this pin is connected to GND. |

# STEP-BY-STEP FOR DEVICE CONNECTION

To build the design we will need the following materials:

* Protoboard
* Arduino Mega 2560 board
* LM35 integrated circuit
* 16x2 4-bit LCD
* 5K potentiometer

1. The first step is to connect the Arduino power pins to the Protoboard. It is important to use a specific convention for connecting to the voltage and GND pins. Usually, a black cable is used to represent GND and a red cable to represent voltage.

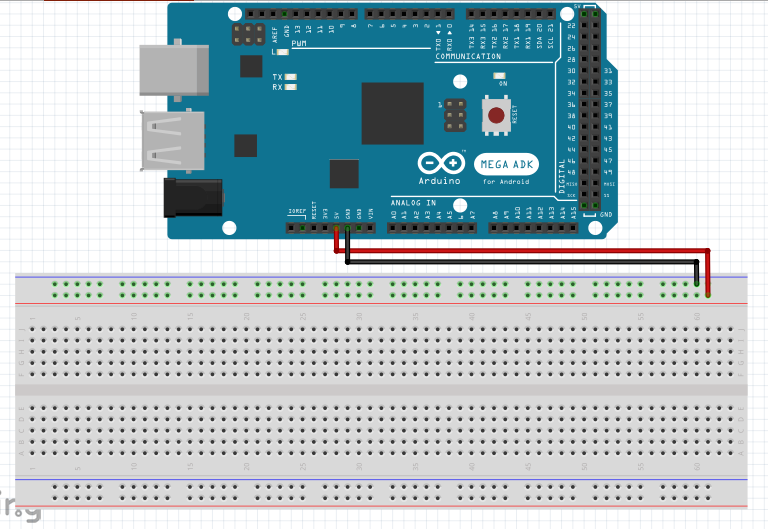


Figure 4: Connection of voltage and GND from the Arduino to the Protoboard.

1. Following this, the LM35 integrated circuit is connected to the Arduino. Pin Vs is connected to an Arduino analog pin. In this case, it was pin A1. It is important to connect pin Vs to an analog pin because the information sent by the LM35 constantly varies and can take a value in a range of -55°C to 150°C. Pin Vout is connected to the voltage path of the Protoboard and finally, the GND pin is connected to the GND path of the Protoboard.

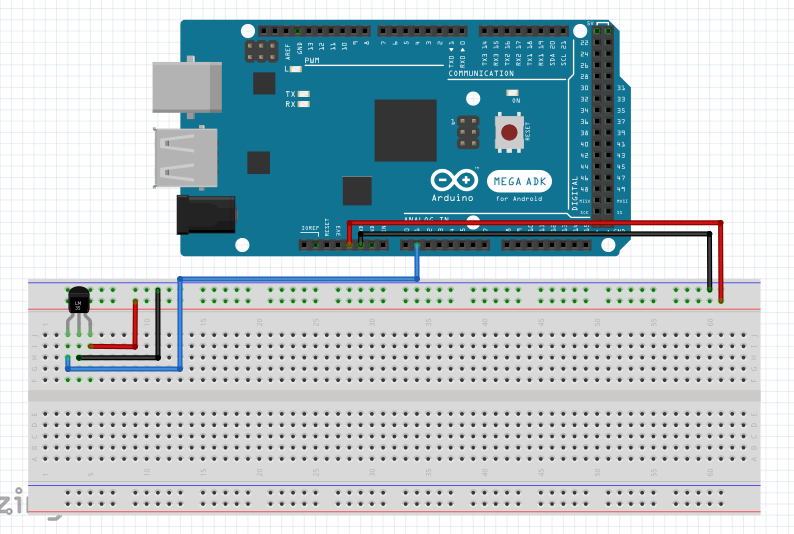


Figure 5: Connection of LM35 pins (blue cable is for the data transfer pin).

1. In order to represent the information graphically, a 16x2 LCD is used. First, the GND and VCC pins are connected to the ground and voltage pins of the Protoboard, respectively. This procedure is shown in Figure 5.

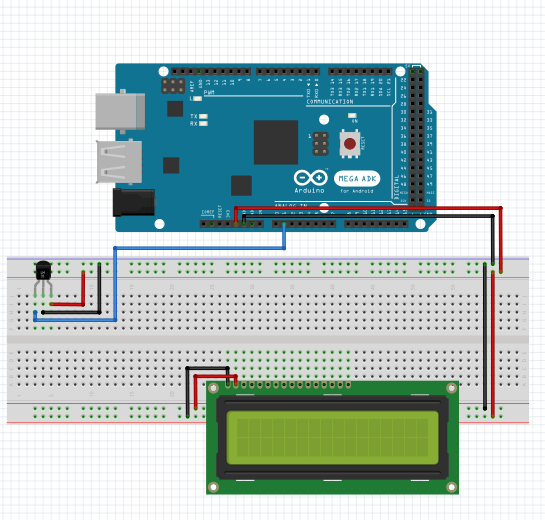


Figure 6: Connection of GND and VCC pins of the LCD.

1. The LCD used in this procedure has the advantage of allowing the variation of the contrast by means of a potentiometer, which consists of a variable resistance, which is adjusted with a knob. The potentiometer is connected to the Vo pin of the LCD. In turn, the potentiometer must be connected to ground and voltage of the Protoboard. The Cyan cable represents the pin through which the potentiometer sends the variable resistance signal. Figure 7 shows the pin schematic for a conventional potentiometer.

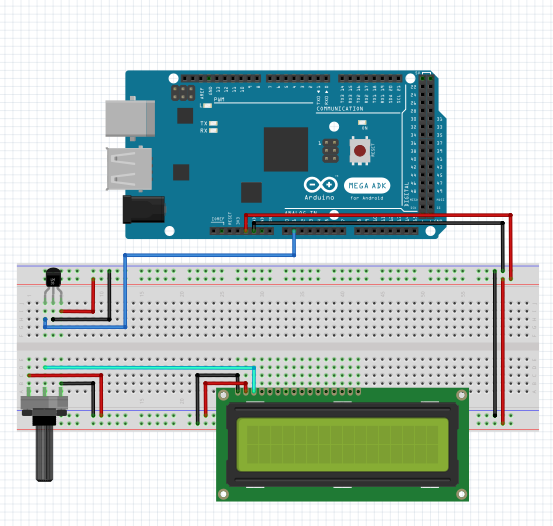


Figure 7: Connection of the LCD contrast control pin.



Figure 8: Schematic of a potentiometer's pins.

1. After connecting the potentiometer to the Vo pin of the LCD, the pins corresponding to the anode and cathode are connected to supply the backlight of the LCD. These two pins are connected to 5V and GND, respectively.

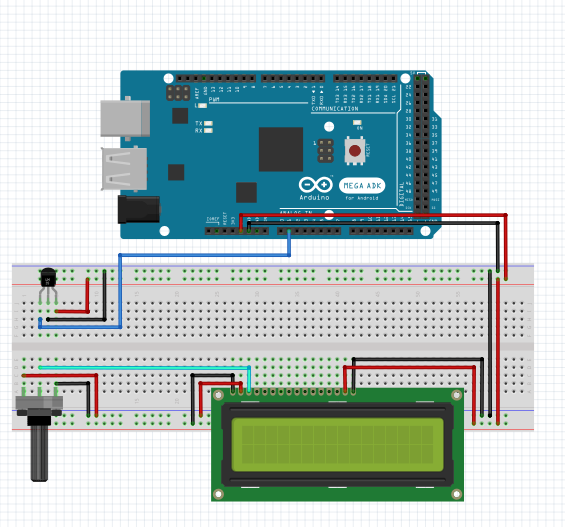


Figure 9: Connection for the backlighting of the LCD (A and K).

1. As previously mentioned, the RS, RW, and E pins are pins used to somehow alert the LCD that it will receive information, so it is necessary to connect them to digital inputs of the Arduino. However, the E (enable) pin is connected directly to ground.

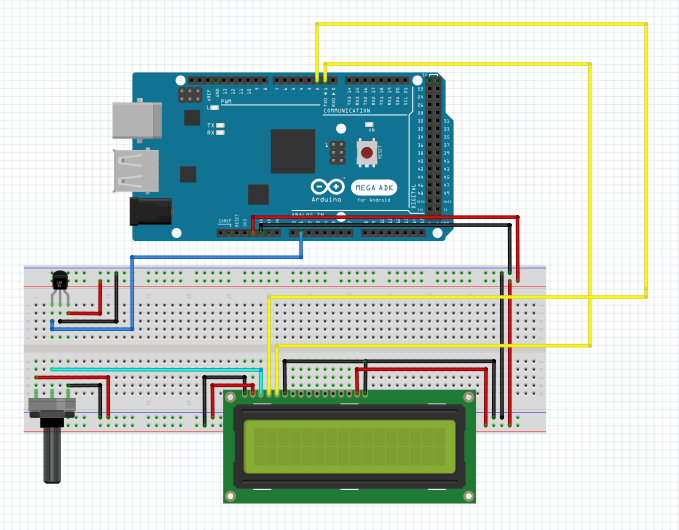


Figure 10: Connection of RS, RW, and E pins of the LCD.

1. As previously mentioned, the D0 to D7 pins are used for the transmission of information and commands with which the functions of the LCD are controlled. On the other hand, only the D4 to D7 pins need to be used to transmit information. Digital pins of Arduino are used to transmit this type of information.

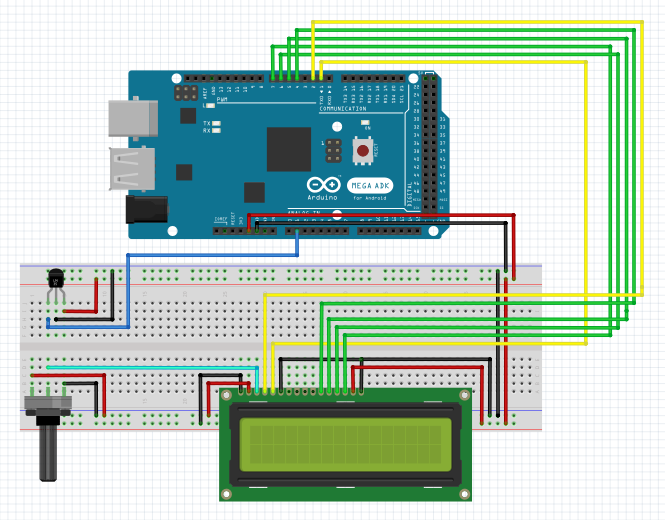


Figure 11: Connection of D4 to D7 pins of the 16x2 LCD.

## Code Construction

So far we have constructed the hardware part of our temperature measuring device. From here on, we will focus on the development of the code to read, process, and present the information collected with the LM35 integrated circuit. To write the code, we will need the Arduino software. After installing the software, proceed to open it, and the following window will appear.

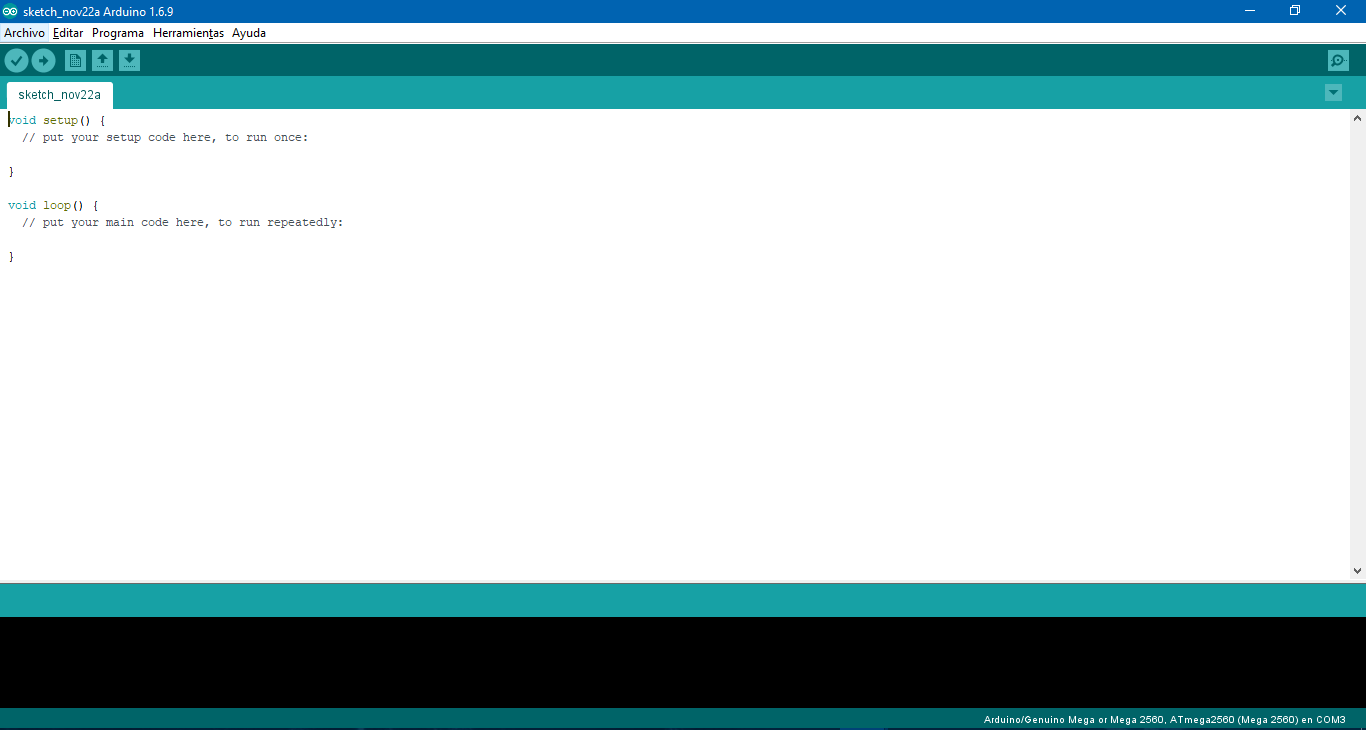


Figure 12: Arduino programming interface.

As you can see, we have two code blocks, the first one is the void setup(), where all variable and constant declarations will be written since this block only executes once. The second block observed is the void loop(), where all the functions and in general the code to be executed cyclically will be written.

To start with the code, it is necessary to load the corresponding libraries to use the LCD. The corresponding library is called LiquidCrystal. Following this, a variable of type "LiquidCrystal" is declared as follows:

*LiquidCrystal nombreLCD (rs, rw, enable, d0, d1, d2, d3, d4, d5, d6, d7)*



Figure 13: Code used to connect the Arduino, the LCD and the LM35.

In order to print the actual temperature data on the LCD, it is necessary to convert volts to degrees Celsius. This conversion is done through the following relationship:

(1)

This conversion is due to the type of information provided by the LM35, which is a number from 0 to 1024 and is voltage-dependent. As our Arduino operates at a voltage of 5 V, the voltage value delivered by the LM35 is calculated using a rule of three. Furthermore, from the datasheet of the integrated circuit, it is known that the slope or conversion value of volts to degrees Celsius is that for an increase of 1°C, the LM35 will increase by 10 mV. In addition, it is known that 10 mV is equivalent to 0.01 V. Therefore, the voltage-to-degree Celsius conversion function is equivalent to equation (1).

# REFERENCES

Artero, O. T. (2013). Arduino: curso práctico de formación. México D.F.: Alfaomega.

# LINKS

## Library for the LCD 16x2 LiquidCrystal:

Versions of Arduino prior to 1.0:

<https://playground.arduino.cc/uploads/Main/LiquidCrystal.zip>

Versions of Arduino after 1.0:

<https://playground.arduino.cc/uploads/Main/LiquidCrystal_1.zip>

Datasheet LM35:

<http://www.ti.com/lit/ds/symlink/lm35.pdf>

Download last version Arduino:

<https://www.arduino.cc/en/Main/Software>

# CHANGE CONTROL

|  |  |  |  |
| --- | --- | --- | --- |
| **CHANGE DESCRIPTION** | **DATE** | **VERSION** | **APPROVED BY** |
|  |  |  |  |