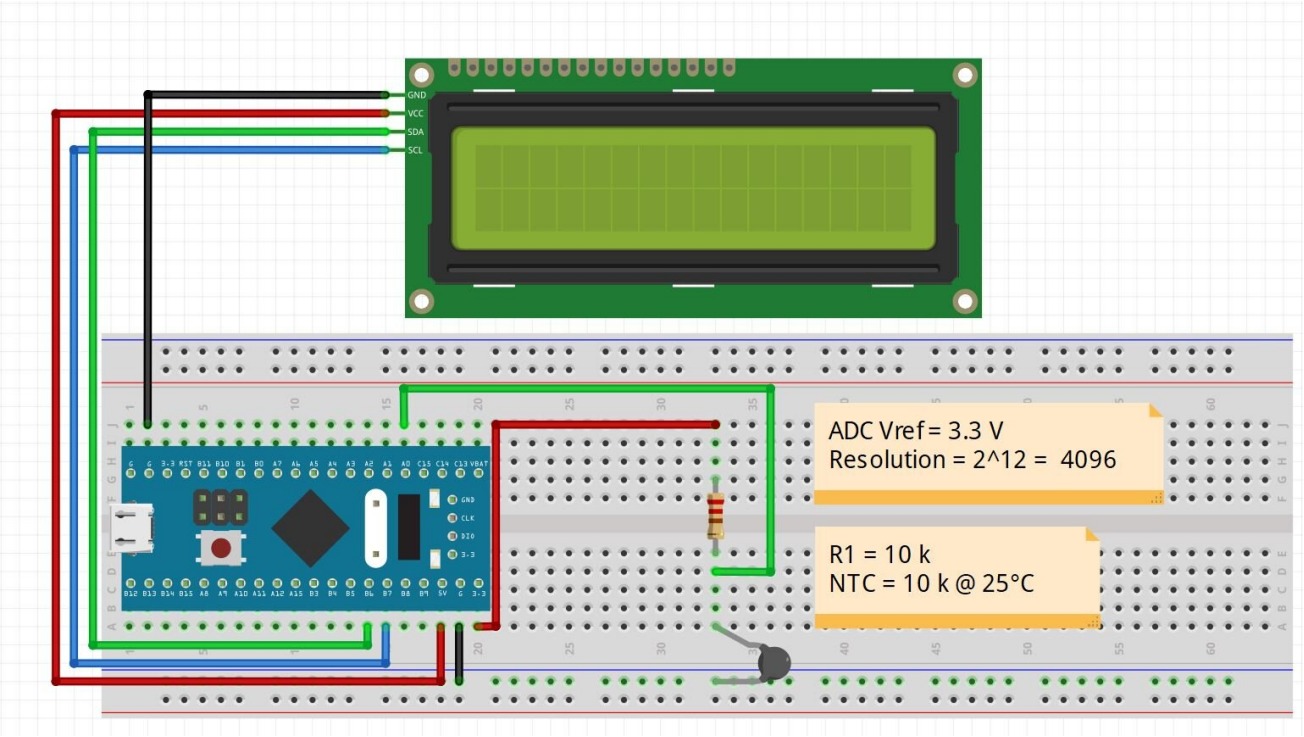
Measurement
Oct 3

NTC thermistor-based thermometer

In this video I show you how to use a NTC thermistor as a thermometer. It is a very simple and relatively accurate way to measure the temperature. The temperature range is quite good for normal use, you can typically go between -50°C to 250°C, but this value can be different based on the exact unit you are using. All you need is a microcontroller, a 10 k resistor and a 10 k NTC thermistor to acquire values. The circuit and the code works both with Arduinos and STM32F103C8T6 (blue pill) circuits. STM32 is a better choice for this task because of its better ADC resolution. Nevertheless, the project works just fine using an Arduino.

Schematics



Schematics of the circuit shown in the video. The R1 is identical to the value of the NTC thermistor: 10 kOhm. Thus, the output voltage of the voltage divider at 25°C is the half of the supply voltage. When you use the STM32, make sure that the ADC pins are never subjected to voltages higher than 3.3 V. You can avoid this by replacing the R1 with a larger resistor than the value of the NTC (3-5x larger is OK).

Arduino/STM32 source code

```
//Code for 10k NTC thermistor thermometer
//The code belongs to the following tutorial video: https://youtu.be/QINcUi-A7dU
//If you use the code, please subscribe to my channel:
https://www.youtube.com/c/CuriousScientist?sub_confirmation=1
//More fun projects on my website: https://curiousscientist.tech


float Vsupply = 3.3; //power supply voltage (3.3 V rail) -STM32 ADC pin is NOT 5 V tolerant
float Vout; //Voltage divider output
float R_NTC; //NTC thermistor resistance in Ohms
```

```
float R_10k = 9840; //10k resistor measured resistance in Ohms (other element
in the voltage divider)
float B_param = 3700; //B-coefficient of the thermistor
float T0 = 298.15; //25°C in Kelvin
float Temp_K; //Temperature measured by the thermistor (Kelvin)
float Temp_C; //Temperature measured by the thermistor (Celsius)
const int VoutPin = PA0; //ADC0 pin of STM32

//Time-related
float TimeNow_1;
float TimeNow_2;
float UpdateInterval = 500;

//16x2 LCD
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
//STM32F103: SDA1: PB7, SCL1: PB6.
//Arduino: SDA: A4, SCL: A5.


void setup()
{

  Serial.begin(115200); //Starting serial
  //-----------------
  //LCD
  lcd.begin();
  lcd.setCursor(0,0); //Defining positon to write from first row,first column
.
  lcd.print("NTC Thermometer");
  lcd.setCursor(0,1);
  lcd.print("Demonstration"); //You can write 16 Characters per line .
  //
  delay(3000); //wait 3 sec
  PrintLCD();
  //-------------------------------------------------------------------------
--
  pinMode(VoutPin, INPUT_ANALOG);
  TimeNow_1 = millis(); //"noting down" start time
}

void loop()
{
  //time condition
  TimeNow_2 = millis(); //update time
  if(TimeNow_2 - TimeNow_1 > UpdateInterval) //500 millis elapsed
  {
  UpdateLCD();
  TimeNow_1 = millis(); //update time. this resets the update interval
  }

  ConvertToTemperature();

}
```

```
void ConvertToTemperature()
{
  Vout = analogRead(VoutPin)* (3.3/4095); //4095 ' 12 bit resolution of the
blue pill
  //For Arduino users: (5.0 / 1023)
  Serial.print("vout: ");
  Serial.println(Vout);
  R_NTC = (Vout * R_10k) /(Vsupply - Vout); //calculating the resistance of
the thermistor
  Serial.print("RNTC: ");
  Serial.println(R_NTC);
  Temp_K = (T0*B_param)/(T0*log(R_NTC/R_10k)+B_param); //Temperature in Kelvin
  Temp_C = Temp_K - 273.15; //converting into Celsius

}

void PrintLCD()
{
  //printing on the LCD
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Temp(K): ");
  lcd.setCursor(9,0);
  lcd.print(Temp_K);    //Print the value
  //
  lcd.setCursor(0,1);
  lcd.print("Temp(C): ");
  lcd.setCursor(9,1);
  lcd.print(Temp_C);    //Print the value
}

void UpdateLCD()
{
  //printing on the LCD - updating
  lcd.setCursor(9,0);
  lcd.print(Temp_K);    //Print the value
  Serial.print("Kelvin: ");
  Serial.println(Temp_K);
  //
  lcd.setCursor(9,1);
  Serial.print("Celsius: ");
  lcd.print(Temp_C);    //Print the value
  Serial.println(Temp_C);
}

//Some discussion
/*
 * STM32 has 12 bit resolution (4096), while Arduinos only have 10 bit (1024)
 * STM32 is 5 V tolerant, but preferrably used with 3.3 V, while Arduino used
with 5 V
 * This also affects their ADC Vref (STM32 Vref = 3.3 V, Arduino Vref = 5 V)
 * This means, that the full scale resolution is 4.89 mV for the Arduino and
0.08 mV for the STM32
```

```
 * This is roughly 6x difference
 * For the NTC, the supply voltage doesn't matter. We feed 3.3 V to the
voltage divider.
 * At 25°C, this will result ~1.65 V on the output. Even at very low
temperatures, we won't surpass 3.3 V
 * You can also avoid surpassing the 3.3 V at low temperatures if you use
large R1. (3-5x larger than NTC) * /
 */
```