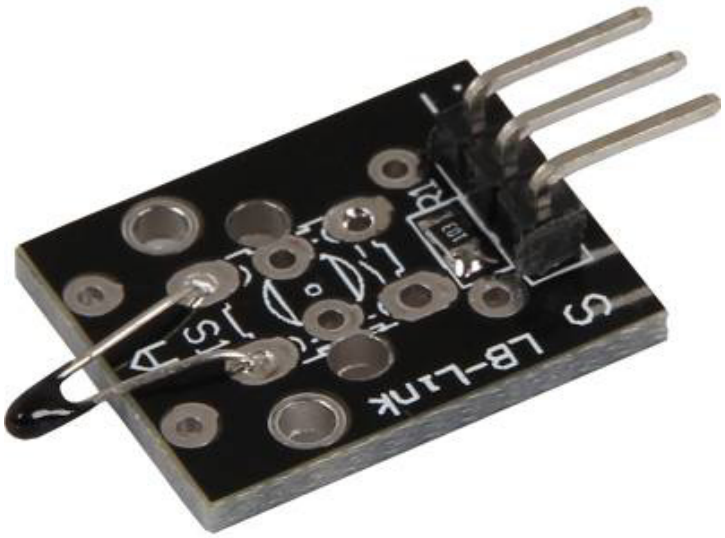
KY-013 Temperature-Sensor module

Contents

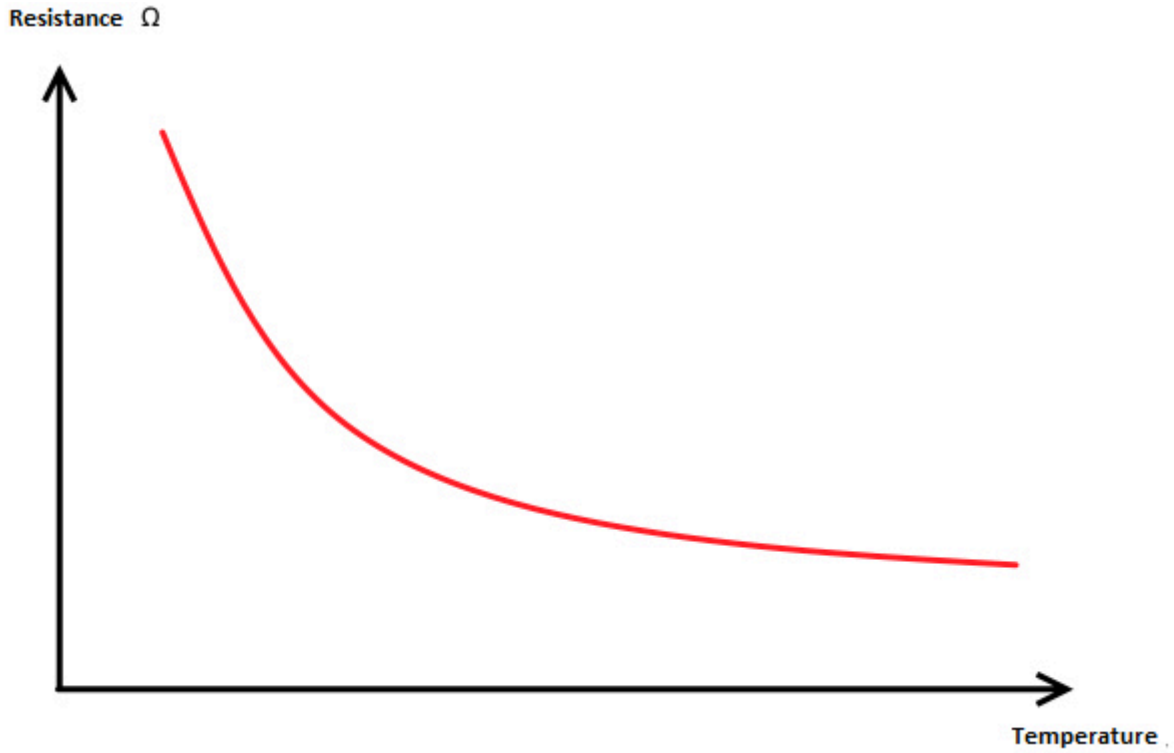
Picture



Technical data / Short description

Temperature measuring range: -55°C / +125°C

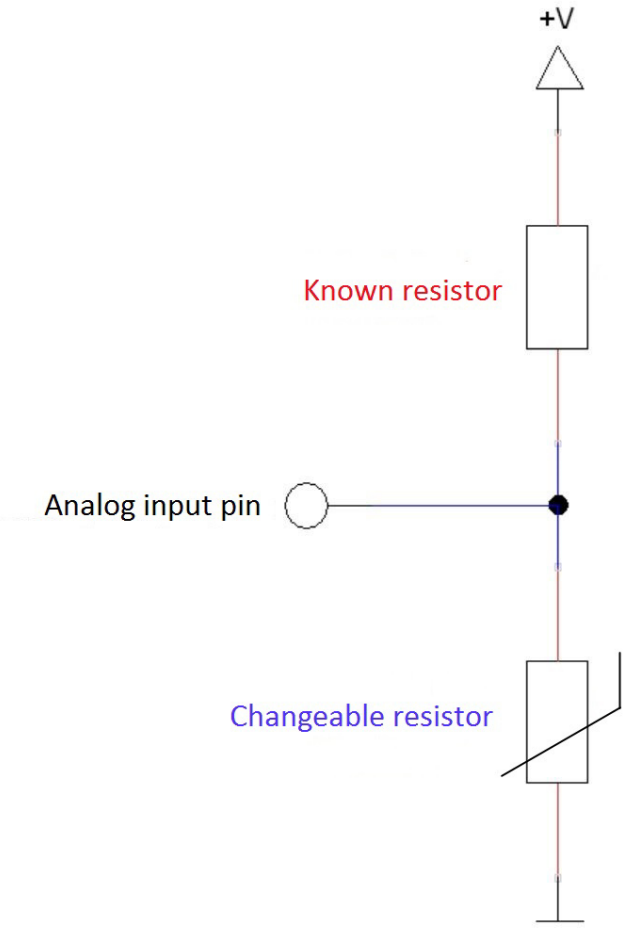
This module provides a NTC thermistor - it will have a lower resistant on higher temperatures.



You can draw near to the resistant change via maths and convert it into a linear course. With that you can determine the temperature coefficient (addicted from resistant change to temperature change). With that you can determine the actual temperature if you know the current resistance.

This resistor can be determinded via voltage devider, where a known voltage splits up between a known resistor and an unknown (variable) resistor.

With that Voltage you can determine the resistance of the resistor - you can see the full calculation in the example below.



Pinout

Code example Arduino

The program measures the actual voltage from the NTC, calculate the temperature and translates the result to °C for the serial output.

```
#include <math.h>

int sensorPin = A5; // Declaration of the input pin

// These function translates the recorded analog measurement
double Thermistor(int RawADC)
{
        double Temp;
        Temp = log(10000.0 * ((1024.0 / RawADC - 1)));
        Temp = 1 /(0.001129148 + (0.000234125 + (0.0000000876741 * Temp * Temp)) * Temp);
        Temp = Temp - 273.15;            // convert from Kelvin to Celsius
        return Temp;
}

// Serial output in 9600 Baud

void setup()
```

```
void setup()
{
        Serial.begin(9600);
}

// The program measures the current voltage value on the NTC
// and translates it intp °C for the serial output
void loop()
{
        int readVal = analogRead(sensorPin);
        double temp =  Thermistor(readVal);

        // Output on the serial interface
        Serial.print("Current temperature is:");
        Serial.print(temp);
        Serial.print(char(186)); //Output <°> Symbol
        Serial.println("C");
        Serial.println("---------------------------------------");

        delay(500);
}
```

Connections Arduino:

Sensor +V	= [Pin 5V]
Sensor GND	= [Pin GND]
Sensor Signal	= [Pin A5]

Example program Download

KY-013_TemperatureSensor

Code example Raspberry Pi

!! Attention !! Analog Sensor !! Attention !!

Unlike the Arduino, the Raspberry Pi doesn't provide an ADC (Analog Digital Converter) on its Chip. This limits the Raspbery Pi if you want to use a non digital Sensor.

To evade this, use our *Sensorkit X40* with the *KY-053* module, which provides a 16 Bit ADC, which can be used with the Raspberry Pi, to upgrade it with 4 additional analog input pins. This module is connected via I2C to the Raspberry Pi.
It measures the analog data and converts it into a digital signal which is suitable for the Raspberry Pi.

So we recommend to use the KY-053 ADC if you want to use analog sensors along with the Raspberry Pi.

For more information please look at the infosite: KY-053 Analog Digital Converter

!! Attention !! Analog Sensor !! Attention !!

The program uses the specific ADS1x15 and I2C python-libraries from the company Adafruit to control the ADS1115 ADC. You can find these here: [https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code] published under the BSD-License [Link]. You can find the needed libraries in the lower download package.

```
### Copyright by Joy-IT
### Published under Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License
### Commercial use only after permission is requested and granted
###
### KY-053 Analog Digital Converter - Raspberry Pi Python Code Example
###




# This code is using the ADS1115 and the I2C Python Library for Raspberry Pi
# This was published on the following link under the BSD license
# [https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code]
from Adafruit_ADS1x15 import ADS1x15
from time import sleep

# import needed modules
import math, signal, sys, os
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

# initialise variables
delayTime = 0.5 # in Sekunden

# assigning the ADS1x15 ADC

ADS1015 = 0x00  # 12-bit ADC
ADS1115 = 0x01  # 16-bit

# choosing the amplifing gain
gain = 4096  # +/- 4.096V
# gain = 2048  # +/- 2.048V
# gain = 1024  # +/- 1.024V
# gain = 512   # +/- 0.512V
# gain = 256   # +/- 0.256V

# choosing the sampling rate
# sps = 8    # 8 Samples per second
# sps = 16   # 16 Samples per second
# sps = 32   # 32 Samples per second
sps = 64   # 64 Samples per second
# sps = 128  # 128 Samples per second
# sps = 250  # 250 Samples per second
# sps = 475  # 475 Samples per second
# sps = 860  # 860 Samples per second

# assigning the ADC-Channel (1-4)
adc_channel_0 = 0    # Channel 0
adc_channel_1 = 1    # Channel 1
adc_channel_2 = 2    # Channel 2
adc_channel_3 = 3    # Channel 3

# initialise ADC (ADS1115)
adc = ADS1x15(ic=ADS1115)

# temperature calculation function
def calcTemp(voltage):
        temperature = math.log((10000/voltage)*(3300-voltage))
       temp = (0.0000000876741 * temperature * temperature)
        temperature = 1 / (0.001129148 + (0.000234125 + temp) * temperature);
        temperature = temperature - 273.15;
        return temperature
```



```
# ########
# Main Loop
# ########
# Reading the values from the input pins and print to console

try:
        while True:
                #read voltage-value and calculate temperature
                temp0 = round(calcTemp(adc.readADCSingleEnded(adc_channel_0, gain, sps)), 2)
                temp1 = round(calcTemp(adc.readADCSingleEnded(adc_channel_1, gain, sps)), 2)
                temp2 = round(calcTemp(adc.readADCSingleEnded(adc_channel_2, gain, sps)), 2)
                temp3 = round(calcTemp(adc.readADCSingleEnded(adc_channel_3, gain, sps)), 2)


                # print to console
                print "Channel 0:", temp0, "C"
                print "Channel 1:", temp1, "C"
                print "Channel 2:", temp2, "C"
                print "Channel 3:", temp3, "C"
                print "----------------------------------------"

                sleep(delayTime)



except KeyboardInterrupt:
        GPIO.cleanup()
```

Connections Raspberry Pi:

Sensor

+V	= 3,3V	[Pin 1 (RPi)]
GND	= GND	[Pin 06 (RPi)]
analog Signal	= Analog 0	[Pin A0 (ADS1115 - KY-053)]

ADS1115 - KY-053:

VDD	= 3,3V	[Pin 17]
GND	= GND	[Pin 09]
SCL	= GPIO03 / SCL	[Pin 05]
SDA	= GPIO02 / SDA	[Pin 03]
A0	= s.o.	[Sensor: analog Signal]

Example program download

KY-013_Temperature-Sensor_RPi

To start, enter the command:

```
sudo python KY-013_RPi_TemperaturSensor.py
```