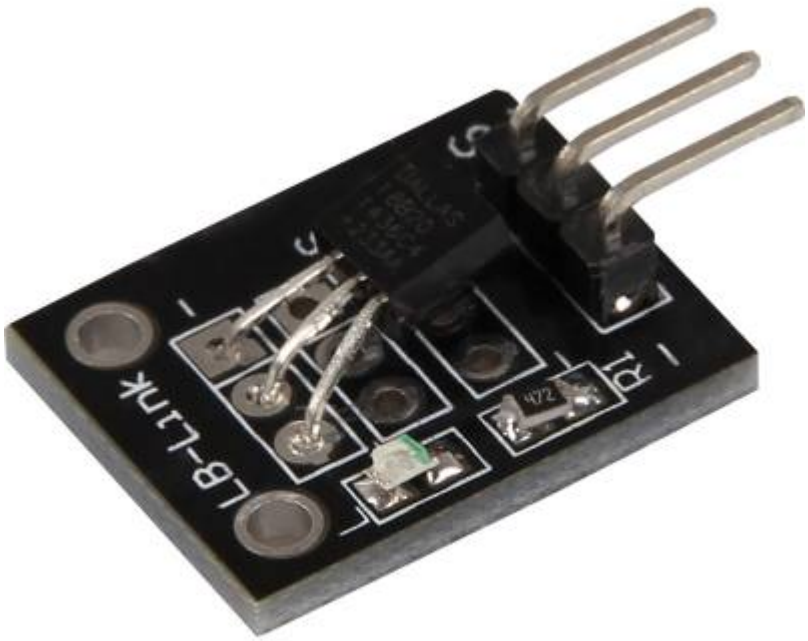
KY-001 Temperature sensor module

Contents

Picture

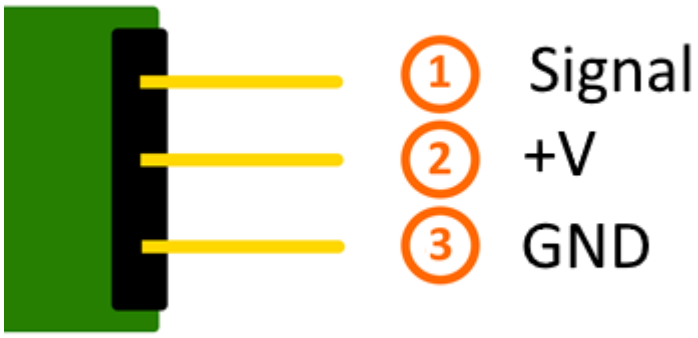


Technical Data / Short description

Chip: DS18B20 | Communication protocol: 1-Wire

9- 12Bit precise temperature measurment between –55°C and +125°C

Pinout



Code example Arduino

You need 2 additional libraries for the following example:

- [OneWire Library] from Paul Stoffregen | published under the MIT license.

- [Dallas Temperature Control Library] from Miles Burton | published under LGPL

Both libraries are part of the package and needs to be copied into the "Library" folder before starting the Arduino IDE.

You can find the path at C:\user\[username]\documents\Arduino\libraries by default.

```
// import needed libraries
#include <DallasTemperature.h>
#include <OneWire.h>

// Declaration of the input pin which is connected with the sensor module
#define KY001_Signal_PIN 4

// libraries configuration
OneWire oneWire(KY001_Signal_PIN);
DallasTemperature sensors(&oneWire);


void setup() {

        // serial output initialization
        Serial.begin(9600);
        Serial.println("KY-001 temperature measurement");

        // sensor will be initialized
        sensors.begin();
}

//main program loop
```

```
void loop()
{
        // temperature measurment will be started...
        sensors.requestTemperatures();
        // ... and measured temperature will be displayed
        Serial.print("Temperature: ");
        Serial.print(sensors.getTempCByIndex(0));
        Serial.write(176); // UniCode of the char-symbol "°-Symbol"
        Serial.println("C");

        delay(1000);    // 1s break till next measurment
}
```

Connections Arduino:

Sensor Signal	= [Pin 4]
Sensor +V	= [Pin 5V]
Sensor -	= [Pin GND]

Example program download

KY-001-TemperatureSensor

One-Wire configuration for Raspberry Pi

To activate the communication between the Raspberry Pi and the DS18B20 sensor, an additional configuration needs to be made.
You need to modify the „/boot/contig.txt“ file and add the following line to it:

```
dtoverlay=w1-gpio,gpiopin=4
```

You can modify the file by entering the following command to the console:

```
sudo nano /boot/config.txt
```

You can safe the modification by pressing [CTRL+Y] and leave the editor by pressing [CTRL+X].

At last, you need to reboot your Raspberry Pi with the following command.
If you followed these steps, your system is ready for the example below.

```
sudo reboot
```

Code example Raspberry Pi

```
# coding=utf-8
# needed modules will be imported and initialised
import glob
import time
from time import sleep
import RPi.GPIO as GPIO
```

```
# here you can modify the break between the measurements
sleeptime = 1

# the one-wire input pin will be declared and the integrated pullup-resistor will be enabl
GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.IN, pull_up_down=GPIO.PUD_UP)

# After the enabling of the pullup-resistor you have to wait till the communication with t
print 'wait for initialisation...'

base_dir = '/sys/bus/w1/devices/'
while True:
    try:
        device_folder = glob.glob(base_dir + '28*')[0]
        break
    except IndexError:
        sleep(0.5)
        continue
device_file = device_folder + '/w1_slave'


# The function to read currently measurement at the sensor will be defined.
def TemperaturMessung():
    f = open(device_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

# To initialise, the sensor will be read "blind"
TemperaturMessung()

# Analysis of temperature: At the Raspberry Pi
# noticed one-wire slaves at the directory /sys/bus/w1/devices/
# will be assigned to a own subfolder.
# In this folder is the file in which the data from the one-wire bus will be saved.<br />#
def TemperaturAuswertung():
    lines = TemperaturMessung()
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = TemperaturMessung()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        temp_c = float(temp_string) / 1000.0
        return temp_c

# main program loop
# The measured temperature will be displayed via console, between the measurements is a br
# The break time can be configured by the variable "sleeptime"
try:
    while True:
        print '--------------------------------------'
        print "Temperature:", TemperaturAuswertung(), "°C"
        time.sleep(sleeptime)

except KeyboardInterrupt:
    GPIO.cleanup()
```

Connections Raspberry Pi:

Signal	=	GPIO4	[Pin 7]
+V	=	3,3V	[Pin 1]
GND	=	GND	[Pin 6]

Example program download:

KY-001_RPi_TemperatureSensor.zip

To start the program use the command:

```
sudo python KY-001_RPi_TemperaturSensor.py
```