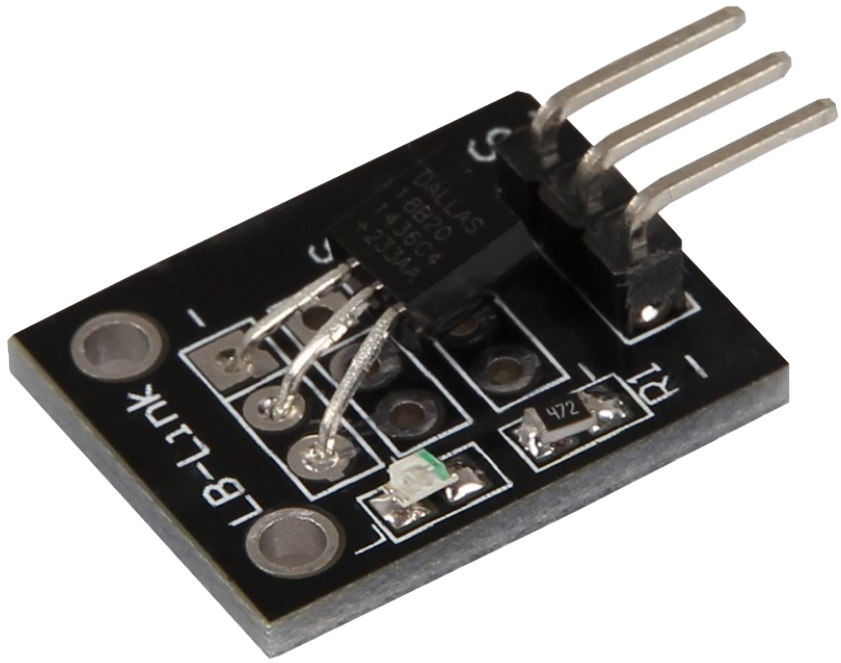
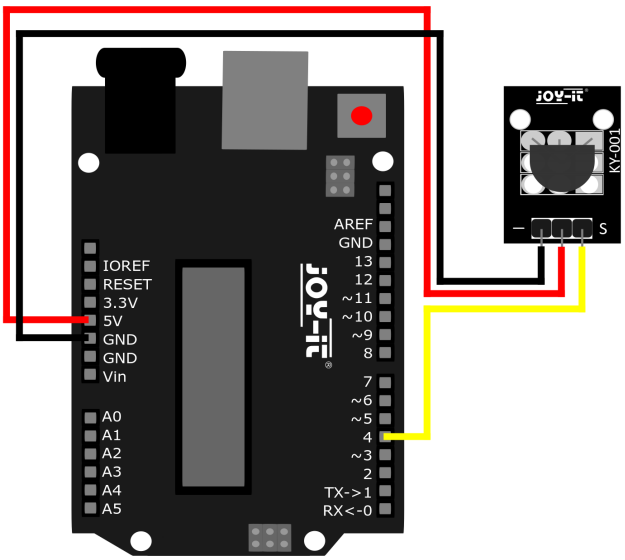
KY-001 TEMPERATURE SENSOR



1. GENERAL INFORMATION

Dear customer,
thank you very much for choosing our product. In following, we will introduce you to what to observe while starting up and using this product. Should you encounter any unexpected problems during use, please do not hesitate to contact us.

2. USAGE WITH THE ARDUINO



Sensor	Arduino
Signal	Pin 4
+V	Pin 5V
GND	Pin GND

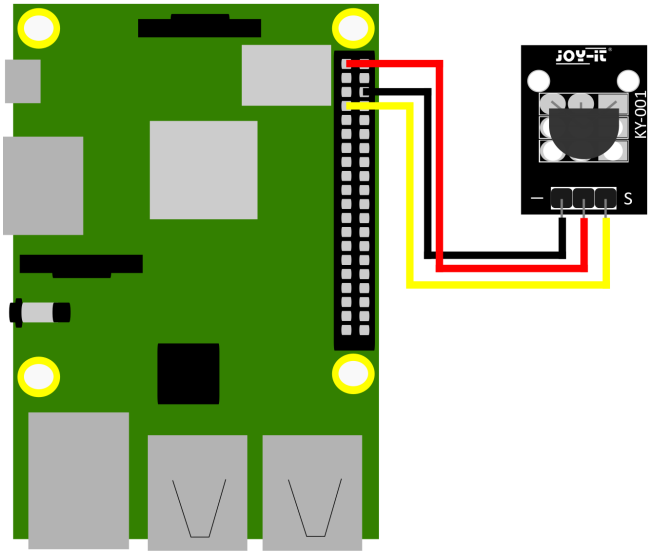
2.1 CODE EXAMPLE ARDUINO

For the following code example two additional libraries are needed. These consist of the **OneWire Library** by Paul Stoffregen, published under the MIT License, and the **Dallas Temperature Control Library** by Miles Burton, published under the LGPL License Both libraries can be managed in the Arduino IDE under Tools > Manage Libraries... and installed.

```
// Benötigte Libraries werden importiert
#include <DallasTemperature.h>
#include <OneWire.h>

// Hier wird der Eingangs-Pin deklariert, an dem das Sensor-Modul angeschlossen ist
#define KY001_Signal_PIN 4

// Libraries werden konfiguriert
OneWire oneWire(KY001_Signal_PIN);
DallasTemperature sensors(&oneWire);

void setup() {

    // Initialisierung Serielle Ausgabe
    Serial.begin(9600);
    Serial.println("KY-001 Temperaturmessung");

    // Sensor wird initialisiert
    sensors.begin();
}
```

```
//Hauptprogrammschleife
void loop()
{


    // Temperaturmessung wird gestartet...
    sensors.requestTemperatures();
    // ... und gemessene Temperatur ausgeben
    Serial.print("Temperatur: ");
    Serial.print(sensors.getTempCByIndex(0));
    Serial.write(176); // UniCode-Angabe eines char-Symbols für das "°-Symbol"
    Serial.println("C");


    delay(1000); // 5s Pause bis zur nächsten Messung
}
```

3. USE WITH THE RASPBERRY PI



Sensor	Raspberry Pi
Signal	GPIO4 [Pin 7]
+V	3,3V [Pin 1]
GND	Masse [Pin 6]

3.1 CODE EXAMPLE RASPBERRY PI

In order for the Raspberry Pi to communicate with the One-Wire Bus, with which the DS18B20 sensor sends its measurement data digitally, it must first be activated. The file **boot/config.txt** must be edited. To do this, open the terminal window and enter the following command to open the file:

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

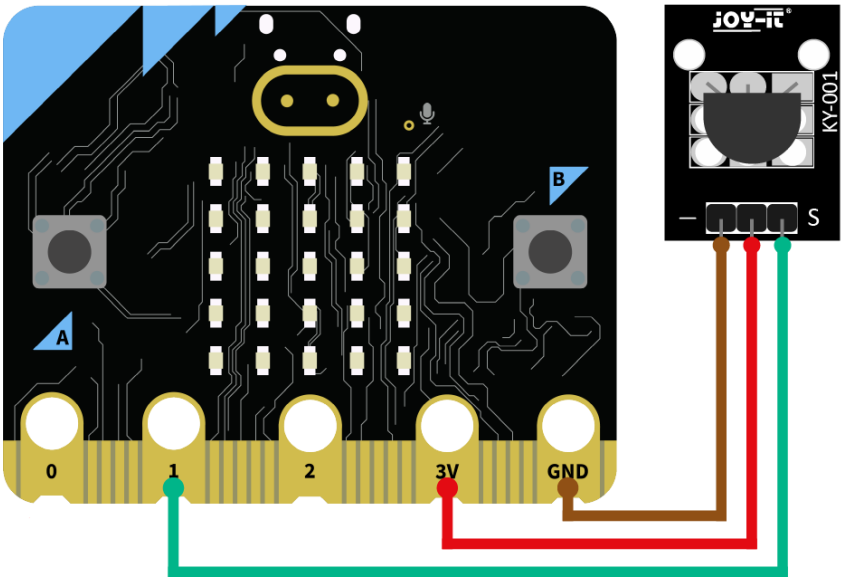
Now add the following lines to the end of the opened file:

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

Save the file with the key combination **STR+O ENTER STR+X**. Restart your Raspberry Pi with the following command to make the changes effective:

```
sudo reboot
```

4. USAGE WITH MICRO:BIT



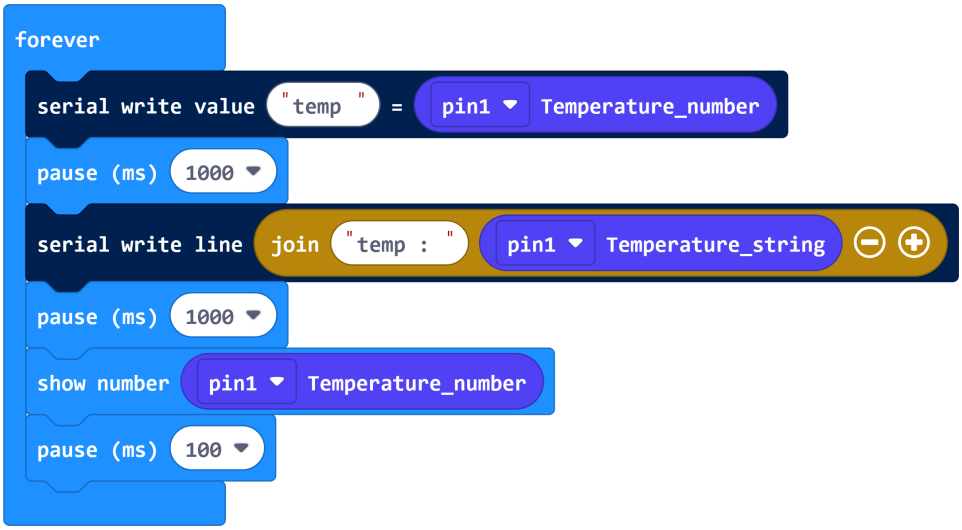
Sensor	Micro:Bit
Signal	Pin 1
+V	3 V
GND	GND

4.1 CODE EXAMPLE MICRO:BIT

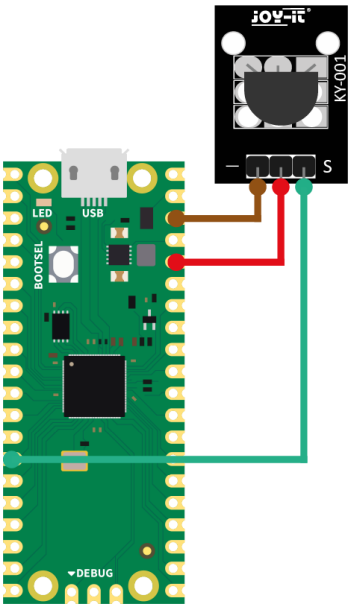
The following code example requires an additional library: pxt-ds18b20 by DFRobot | released under the GUI License.

Add the library to your project by clicking on "Extensions" and entering the following URL in the search field: https://github.com/DFRobot/pxt-ds18b20.git.

Confirm the search with [Enter]. This is an example program which outputs the measured temperature serially after initializing the sensor:



5. USAGE WITH RASPBERRY PI PICO



Sensor	Micro:Bit
Signal	GP2
+V	3,3 V
GND	GND

5.1 USAGE WITH RASPBERRY PI PICO

For the following code example two additional libraries are needed:

OneWire Library by Damien P. George | published under the MIT-Lizenz.

DS18x20 Library by Damien P. George | published under the MIT-Lizenz.

This is an example program which outputs the measured temperature serially after initialization of the sensor.

```
# Load libraries
import machine, onewire, ds18x20
from time import sleep

# Initialization of GPIO2
ds_pin = machine.Pin(10)

# Initialization of the sensor object
ds_sensor = ds18x20.DS18X20(onewire.OneWire(ds_pin))

# Search for all matching sensors
roms = ds_sensor.scan()

# Serial output
print("Found DS devices")
print("Temperature (°C)")

# Endless loop for continuous reading of the temperature
while True:
  ds_sensor.convert_temp()
  sleep(1)
  # Based on the number of compatible sensors found it will count up
  for rom in roms:
    # Serial output of the measured temperature
    print(ds_sensor.read_temp(rom))
  sleep(3)
```

You can now copy the code example for Raspberry here :

```
# coding=utf-8
# Benoetigte Module werden importiert und eingerichtet
import glob
import time
from time import sleep
import RPi.GPIO as GPIO

# An dieser Stelle kann die Pause zwischen den einzelnen Messungen eingestellt werden
sleeptime = 1

# Der One-Wire EingangsPin wird deklariert und der integrierte PullUp-Widerstand ak-
tiviert
GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.IN, pull_up_down=GPIO.PUD_UP)

# Nach Aktivierung des Pull-UP Widerstandes wird gewartet,
# bis die Kommunikation mit dem DS18B20 Sensor aufgebaut ist
print 'Warte auf Initialisierung...'

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'

# Funktion wird definiert, mit dem der aktuelle Messwert am Sensor ausgelesen werden
kann
def TemperaturMessung():
    f = open(device_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

# Zur Initialisierung, wird der Sensor einmal "blind" ausgelesen
TemperaturMessung()

# Die Temperaturauswertung: Beim Raspberry Pi werden erkannte One-Wire Slaves im Ord-
ner
# /sys/bus/w1/devices/ einem eigenen Unterordner zugeordnet. In diesem Ordner befin-
det sich die Datei w1-slave
# in dem Die Daten, die über dem One-Wire Bus gesendet wurden gespeichert.
# In dieser Funktion werden diese Daten analysiert und die Temperatur herausgelesen
und ausgegeben
```

6. ADDITIONAL INFORMATION

Our information and take-back obligations according to the Electrical and Electronic Equipment Act (ElektroG)

Symbol on electrical and electronic equipment:

This crossed-out dustbin means that electrical and electronic appliances do not belong in the household waste. You must return the old appliances to a collection point. Before handing over waste batteries and accumulators that are not enclosed by waste equipment must be separated from it.

Return options:

As an end user, you can return your old device (which essentially fulfils the same function as the new device purchased from us) free of charge for disposal when you purchase a new device. Small appliances with no external dimensions greater than 25 cm can be disposed of in normal household quantities independently of the purchase of a new appliance.

Possibility of return at our company location during opening hours:

SIMAC Electronics GmbH, Pascalstr. 8, D-47506 Neukirchen-Vluyn, Germany

Possibility of return in your area:

We will send you a parcel stamp with which you can return the device to us free of charge. Please contact us by email at Service@joy-it.net or by telephone.

Information on packaging:

If you do not have suitable packaging material or do not wish to use your own, please contact us and we will send you suitable packaging.

7. SUPPORT

If there are still any issues pending or problems arising after your purchase, we will support you by e-mail, telephone and with our ticket support system.

Email: **service@joy-it.net**

Ticket system: http://support.joy-it.net

Telephone: +49 (0)2845 9360—50 (10-17 o‘clock)

For further information please visit our website: **www.joy-it.net**

Published: 2023.09.05