LinkerKit LK-Temp2 | Raspberry Pi und Arduino

Sehr geehrter Kunde,

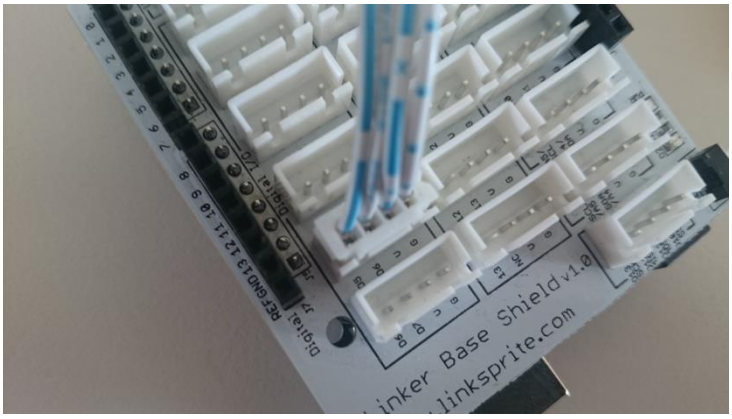
vielen Dank, dass Sie sich für unser Produkt entschieden haben.
Im Folgenden haben wir aufgelistet, was bei der Inbetriebnahme zu beachten ist:

Artikelnummer	**LK-Temp2**
Spannungsversorgung	3.0V—5.5V
Temperatur-Meßbereich	-55° bis 125°C [empfohlen max. 100°C]
Chipsatz	DS18B20
Kabellänge	1m
Features	9 Bit / 12 Bit Auflösung auswählbar ±0.5°C Empfindlichkeit von -10°C to +85°C einmalige One-Wire ID im Chip eingebrannt
Eigenschaften	Wasserdicht, One-Wire, 1m langes Kabel

Anschlussbelegung

Das LinkerKit Modul muss am Digitalport mit der Nummer 4 angeschlossen werden. Der entsprechende LinkerKit Steckplatz lautet:

[D4 | D5 | V | G **]**



Verwendungshinweis

Bitte beachten Sie, dass unser LK-Temp2 Temperatursensor mit dem eingebauten Pull-Up Widerstand betrieben wird.
Dafür ist für den Arduino eine angepasste Programmbibliothek erforderlich, die sie auf der LinkerKit Website zum Downloaden finden.
Auch für den Raspberry Pi ist ein angepasster Code notwendig. Hier genügt allerdings die Verwendung unseres Codebeispiels.

Sollten Sie beabsichtigen, mehrere Sensoren gleichzeitig zu verwenden, oder den Sensor über ein Kabel mit einer Länge von mehreren Metern zu betreiben, so reicht der interne Pull-Up Widerstand nicht mehr aus.
Ein externer Pull-Up Widerstand von 4,7kΩ ist in diesem Fall notwendig, der zwischen die gelbe Signalleitung und die rote Spannungsleitung geschaltet wird.

Codebeispiel Raspberry

Damit der Raspberry Pi mit dem One-Wire Bus, mit der Sensor DS18B20 seine Messdaten digital sendet, kommunizieren kann, muss dieser vorerst aktiviert werden. Hierbei muss die Datei "/boot/config.txt" editiert und um folgende Zeile ergänzt werden:

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

Die Datei können Sie editieren, indem Sie den Befehl...

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

... in die Konsole eingeben. Mit der Tastenkombination [STRG+X] können Sie das Editieren beenden und mit [STRG+Y] abspeichern.

Nachdem Sie den Raspberry Pi mittels...

```
sudo reboot
```

... neugestartet haben, können Sie das untenstehende Beispiel anwenden.

Programmierbeispiel in der Programmiersprache Python

```
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 aktiviert

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 erkennte one-Wire Slaves
im Ordner
# /sys/bus/w1/devices/ einem eigenen Unterordner zugeordnet. In diesem Ordner
befindet 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
```

```
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

# Hauptprogrammschleife
# Die gemessene Temperatur wird in die Konsole ausgegeben - zwischen den
einzelnen Messungen
# ist eine Pause, deren Länge mit der Variable "sleeptime" eingestellt werden
kann

try:
	while True:
		print '---------------------------------------'
		print "Temperatur:", TemperaturAuswertung(), "°C"
		time.sleep(sleeptime)
except KeyboardInterrupt:
	PIO.cleanup()
```

Codebeispiel Arduino

Bitte beachten Sie, dass veränderte Versionen der Zusatzbibliotheken „OneWire“ und „Dallas Temperature Control Library“ erforderlich sind, um den Sensor problemlos verwenden zu können. Die angepassten Versionen finden Sie auf der LinkerKit Website.

Beide Libraries müssen vor dem Start der Arduino IDE in den "library"-Ordner kopiert werden. Diesen finden Sie standardmäßig unter dem folgenden Pfad Ihrer Windows-Installation:

C:\Benutzer\[Benutzername]\Dokumente\Arduino\libraries

Codebeispiel Arduino

```
// 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); // 1s Pause bis zur nächsten Messung
}
```