

KY015 DHT11 TEMPERATURE SENSOR

Temperature and humidity sensor

1. GENERAL INFORMATION

Dear Customer,
Thank you for choosing our product. In the following, we will show you what to consider during commissioning and use. If you encounter any unexpected problems during use, please feel free to contact us.

The DHT11 is a low-cost and straightforward digital temperature and humidity sensor that is particularly well suited for use with Raspberry Pi and Arduino.

2. BASIC INFORMATION

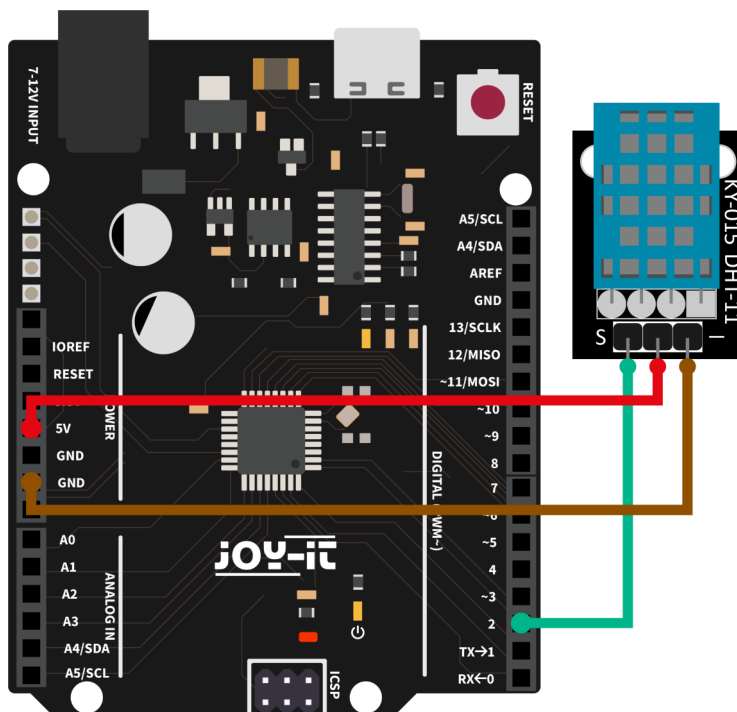
This sensor is a combination of temperature sensor and humidity sensor, united in a compact design. The sampling rate of the measurements is 2 seconds. This sensor is therefore particularly suitable for long-term measurements.

Technical Specifications

| | |
|------------------------------------|---------------|
| Chipset | DHT11 |
| Communication protocol | 1-Wire |
| Measuring range temperature | 0 °C to 50 °C |
| Measurement accuracy (temperature) | ±2 °C |
| Humidity measuring range | 20-90 %RH |
| Measurement accuracy (humidity) | ±5 %RH |

3. USE WITH THE ARDUINO

3.1 Connection



3. ASSEMBLY AND INSTALLATION

| Arduino | DHT11 |
|---------|--------|
| Pin D2 | Signal |
| 5 V | +V |
| GND | GND |

3.2 code example

We offer a code example for the use with the Arduino, which you can download [here](#). This sensor does not output its measurement result as an analog signal to an output pin, but communicates it digitally and coded. There are several possibilities to control this sensor module. The

DHT-sensor-library published by the company [Adafruit](#) under the [MIT-license](#), has proven to be particularly accessible. You can add the library directly in the Arduino IDE.

To do this, click on **Sketch → Include Library → Manage Libraries ...**. There you can search for the **DHT sensor library** in the search bar and install it.

To use the sensor you now only need to copy the following modified code into your Arduino IDE and upload it to your Arduino. Before you start the code, make sure that the correct **board** is selected under **Tools → Board:** and the correct **COM port** is selected under **Tools → Port.**

```
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 2      // Digital pin connected to the DHT sensor

#define DHTTYPE      DHT11      // DHT 11

DHT_Unified dht(DHTPIN, DHTTYPE);

uint32_t delayMS;

void setup() {
  Serial.begin(9600);
  // Initialize device
  dht.begin();
  Serial.println(F("DHTxx Unified Sensor Example"));
  // Print temperature sensor details
  sensor_t sensor;
  dht.temperature().getSensor(&sensor);
  Serial.println(F("-----"));
  Serial.println(F("Temperature Sensor"));
  Serial.print (F("Sensor Type: ")); Serial.println(sensor.name);
  Serial.print (F("Driver Ver:  ")); Serial.println(sensor.version);
  Serial.print (F("Unique ID:   ")); Serial.println
(sensor.sensor_id);
  Serial.print (F("Max Value:   ")); Serial.print(sensor.max_value);
  Serial.println(F("°C"));
```

```

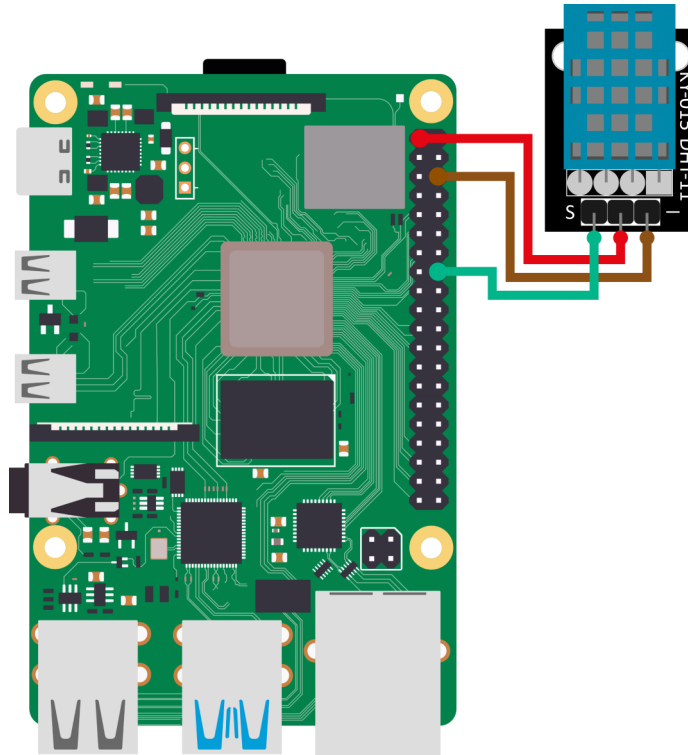
    Serial.print (F("Min Value:  ")); Serial.print(sensor.min_value);
    Serial.println(F("°C"));
    Serial.print (F("Resolution:  ")); Serial.print(sensor.resolution);
    Serial.println(F("°C"));
    Serial.println(F("-----"));
    // Print humidity sensor details
    dht.humidity().getSensor(&sensor);
    Serial.println(F("Humidity Sensor"));
    Serial.print (F("Sensor Type:  ")); Serial.println(sensor.name);
    Serial.print (F("Driver Ver:  ")); Serial.println(sensor.version);
    Serial.print (F("Unique ID:  ")); Serial.println
(sensor.sensor_id);
    Serial.print (F("Max Value:  ")); Serial.print(sensor.max_value);
    Serial.println(F("%"));
    Serial.print (F("Min Value:  ")); Serial.print(sensor.min_value);
    Serial.println(F("%"));
    Serial.print (F("Resolution:  ")); Serial.print(sensor.resolution);
    Serial.println(F("%"));
    Serial.println(F("-----"));
    // Set delay between sensor readings based on sensor details.
    delayMS = sensor.min_delay / 1000;
}

void loop() {
    // Delay between measurements.
    delay(delayMS);
    // Get temperature event and print its value.
    sensors_event_t event;
    dht.temperature().getEvent(&event);
    if (isnan(event.temperature)) {
        Serial.println(F("Error reading temperature!"));
    }
    else {
        Serial.print(F("Temperature:  "));
        Serial.print(event.temperature);
        Serial.println(F("°C"));
    }
    // Get humidity event and print its value.
    dht.humidity().getEvent(&event);
    if (isnan(event.relative_humidity)) {
        Serial.println(F("Error reading humidity!"));
    }
    else {
        Serial.print(F("Humidity:  "));
        Serial.print(event.relative_humidity);
        Serial.println(F("%"));
    }
}

```

4. USE WITH THE RASPBERRY PI

4.1 Connection



| Raspberry Pi | DHT11 |
|------------------|--------|
| GPIO 23 (Pin 16) | Signal |
| +3,3 V (Pin 1) | +V |
| GND (Pin 6) | GND |

4.2 code example

We provide a code example for use with the Raspberry Pi which you can download [here](#). This code example uses the [Adafruit_CircuitPython_DHT](#) library from [Adafruit](#) which is released under the [MIT-License](#).

But first you have to execute the following commands:

```
sudo apt-get update
sudo apt-get install build-essential python3-dev
sudo apt install gpod
```

Now create the necessary virtual environment for the project with the following commands:

```
mkdir dein_projekt
```

```
cd dein_projekt
```

```
python -m venv --system-site-packages env
```

```
source env/bin/activate
```

Now install the library with the following command:

```
pip3 install adafruit-circuitpython-dht
```

After you have executed the commands, you can also alternatively, if you have not downloaded the code sample, copy the following identical code sample into a newly created file. To do this, you must enter the following command into the console on your Raspberry Pi.

```
nano DHT11.py
```

Once you have done this, all you need to do is paste the following code into the file you have just created.

```
import time
import board
import adafruit_dht

# Initialize the DHT with the data pin connected to pin 16 (GPIO 23)
# of the Raspberry Pi:
dhtDevice = adafruit_dht.DHT11(board.D23)

# You can pass DHT22 use_pulseio=False if you do not want to use pul-
# seio.
# This may be necessary on a Linux single board computer like the
# Raspberry Pi, but it will not work in CircuitPython.
# dhtDevice = adafruit_dht.DHT22(board.D18, use_pulseio=False)

while True:
    try:
        # Output of the values via the serial interface
        temperature_c = dhtDevice.temperature
        temperature_f = temperature_c * (9 / 5) + 32
        humidity = dhtDevice.humidity
        print("Temp: {:.1f} F / {:.1f} C    Humidity: {}% ".format
              (temperature_f, temperature_c, humidity))

    except RuntimeError as error:
        # Mistakes happen quite often, DHT's are hard to read, just
        # move on
        print(error.args[0])
        time.sleep(2.0)
        continue
    except Exception as error:
        dhtDevice.exit()
        raise error

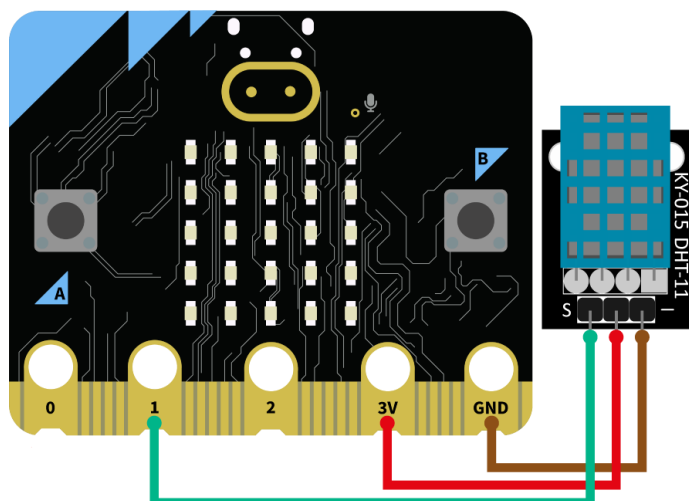
    time.sleep(2.0)
```

You can save the file with **CTRL+O** and then close it with **CTRL+X**.
You can execute the file in the console with the command

```
python3 DHT11.py
```

5. USAGE WITH MICRO:BIT

5.1 Connection



| Micro:Bit | DHT11 |
|-----------|--------|
| Pin 1 | Signal |
| +3,3 V | +V |
| GND | GND |

5.2 Codeexample

For the following code example an additional library is needed. To install it, click on "Add extension" in your make code sketch and search for "DHT11" here. Install here the [„DHT11_DHT22“ library by alankrantas](#).

Transfer the following sample code into your sketch, or import the [.hex-file](#).

on start

pause (ms) 1000 ▾

forever

Query DHT11 ▾

Data pin P1 ▾

Pin pull up true ▾

Serial output false ▾

Wait 2 sec after query true ▾

show string join "T:" Read temperature ▾ "C" − +

pause (ms) 2000 ▾

Query DHT11 ▾

Data pin P1 ▾

Pin pull up true ▾

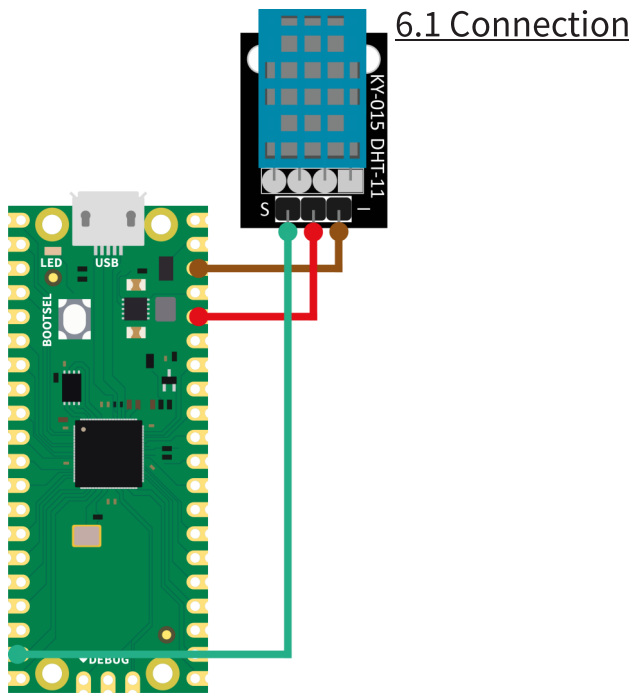
Serial output false ▾

Wait 2 sec after query true ▾

show string join "H:" Read humidity ▾ "%" − +

pause (ms) 2000 ▾

6. USAGE WITH RASPBERRY PI PICO



| Raspberry Pi Pico | DHT11 |
|-------------------|--------|
| GPIO14 | Signal |
| +3,3 V | +V |
| GND | GND |

6.2 Code example

For the following code example an additional library is needed: [micropython-stubs](#) by Jos Verlinde | published under the [MIT license](#). Download the code sample [here](#) or transfer the following code completely to your Raspberry Pi Pico.

```
# Load libraries
from machine import Pin
from utime import sleep
from dht import DHT11

# Initialization GPIO14 and DHT11
sleep(1)
dht11_sensor = DHT11(Pin(14, Pin.IN, Pin.PULL_UP))

# Repeat (endless Loop)
while True:
    # Perform measurement
    dht11_sensor.measure()
    # Read values
    temp = dht11_sensor.temperature()
    humi = dht11_sensor.humidity()
    # Output values
    print("Temperature: {}°C   Humidity: {:.0f}% ".format(temp, humi))
    print()
    sleep(3)
```

7. 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.

8. SUPPORT

If any questions remained open or problems may arise after your purchase, we are available by e-mail, telephone and ticket support system to answer these.

E-Mail: service@joy-it.net

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

Telephone: +49 (0)2845 9360-50 (Mo - Do: 09:00 - 17:00 o'clock CET,
Fri: 09:00 - 14:30 o'clock CET)

For further information visit our website:

www.joy-it.net