12.DHT11

Introduction

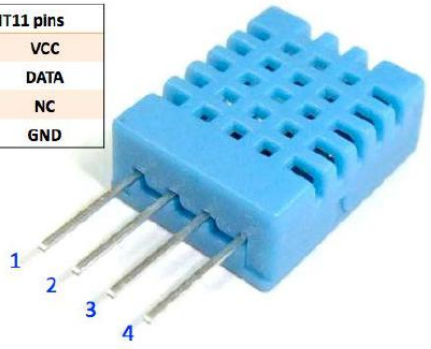
The digital temperature and humidity sensor DHT11 is a composite sensor that contains a calibrated digital signal output of temperature and humidity. The technology of a dedicated digital modules collection and the technology of the temperature and humidity sensing are applied to ensure that the product has high reliability and excellent stability.

The sensors include a wet element resistive sensor and a NTC temperature sensor and they are connected to a high performance 8-bit microcontroller.

Hardware Required

- ✓ 1 * Raspberry Pi
- ✓ 1 * T-Extension Board
- ✓ 1 * 40-pin Cable
- ✓ Several Jumper Wires
- ✓ 1 * Breadboard
- ✓ 1 * DHT-11
- ✓ 1 * Resistor 10KΩ

DHT11 pins	
1	VCC
2	DATA
3	NC
4	GND



Principle

DHT11 Temperature and Humidity Sensor

DHT11 output calibrated digital signal. It applys exclusive digital-signal-collecting-techniq ue and humidity sensing technology, assuring its reliability and stability. Its sensing eleme nts is connected with 8-bit single-chip computer.

Every sensor of this model is temperature compensated and calibrated in accurate calibr ation chamber and the calibration-coefficient is saved in type of programme in OTP mem ory, when the sensor is detecting, it will cite coefficient from memory.

Small size & low consumption & long transmission distance(100m) enable DHT11 to be s uited in all kinds ofharsh application occasions. Single-row packaged with four pins, maki ng the connection very convenient.

Supply voltage: DC 3.3 to 5.5V

Measuring range (T) : -20 to +60 Celsius(-4 to +140 Fahrenheit)

Measuring range (RH): 5 to 95% relative humidity

Typ. Temperature accuracy: ±2 Celsius

Typ. Humidity accuracy: ±5%RH at 25 Celsius

Long term drift(T): <1 Celsius/year

Long term drift(RH) : <1%RH/year

Resolution(T): 0.1 Celsius

Resolution(RH): 1%RH

Sensor Type: Capacitive sensor

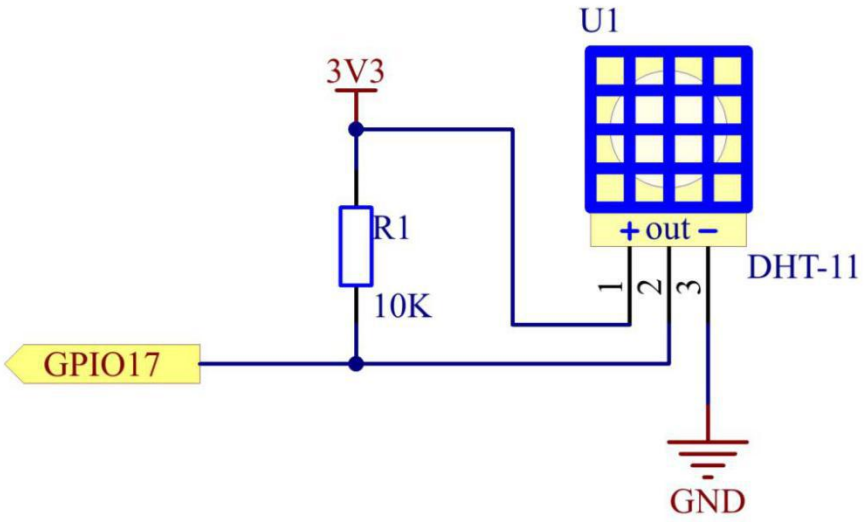
Interface: One line digital

Housing material: ABS

Net weight: 1g

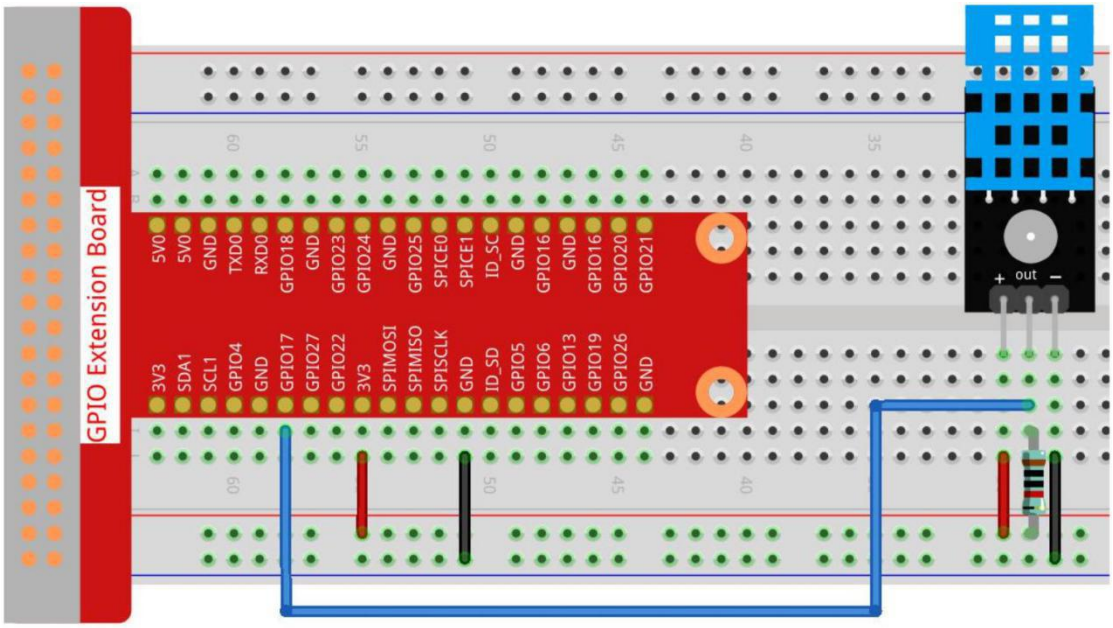
Schematic Diagram

T-Board Name	physical	wiringPi	BCM
GPIO17	Pin 11	0	17



Experimental Procedures

Step 1: Build the circuit.



For C Language Users

Step 2: Go to the folder of the code.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/12.DHT11
```

Step 3: Compile the code.

```
gcc 12.DHT11.cpp -o DHT11.out -lwiringPi
```

Note: The program contains custom header files that are compiled when CPP files are compiled.

Step 4: Run the executable file.

```
sudo ./DHT11.out
```

After the code runs, the program will print the temperature and humidity detected by DHT11 on the computer screen.

Code

Note: The following codes are incomplete. If you want to check the complete codes, you are suggested to use command nano 12.DHT11.cpp in bash.

```
#include <wiringPi.h>
#include <stdio.h>
#include <stdint.h>
```

```
#include "DHT11.hpp"
#define DHT11_Pin    0
//Function: Read DHT sensor, store the original data in bits[]
int DHT::readSensor(int pin,int wakeupDelay){
    ……
}
//Function：Read DHT sensor, analyze the data of temperature and humidity
int DHT::readDHT11(int pin){
    ……
}
int main(){
    DHT dht;            //create a DHT class object
    int check;
    if(wiringPiSetup() == -1){ //when initialize wiring failed,print message to screen
        printf("setup wiringPi failed !");
        return 1;
    }
    while(1){
        check = dht.readDHT11(DHT11_Pin);
        switch(check){
            case DHTLIB_OK:    //if the return value is DHTLIB_OK, the data is
normal.
                printf("Humidity : %.2f, \t Temperature :
%.2f\n",dht.humidity,dht.temperature);
                break;
            case DHTLIB_ERROR_CHECKSUM:        //data check has errors
                printf("Humidity : %.2f, \t Temperature : %.2f\t (this value may be
                incorrect)\n",dht.humidity,dht.temperature);
                break;
            case DHTLIB_ERROR_TIMEOUT:         //reading DHT times out
```

```
                    printf("Timeout! \n");
                    break;
                case DHTLIB_INVALID_VALUE:          //other errors
                    printf("Unknow problem! \n");
                    break;
            }
            delay(2000);
}
    return 1;
}
```

Code Explanation

```
#include "DHT11.hpp"
```

DHT11.hpp is an open source file for reading sensor values of DHT which makes it easy for us to use the DHT sensor. Here, we write the invocation functionality directly below the open source file.

```
int DHT::readSensor(int pin,int wakeupDelay){}
int DHT::readDHT11(int pin){}
```

These two functions are the content of the open source file itself, and they will read the value of the sensor. After calculation we can understand the humidity and temperature in that they themselves have a return value that monitors the status of the sensor.

```
        check = dht.readDHT11(DHT11_Pin);
        switch(check){
            case DHTLIB_OK:    //if the return value is DHTLIB_OK, the data is
            normal.
                printf("Humidity : %.2f, \t Temperature : %.2f
\n",dht.humidity,dht.temperature);
                break;
            case DHTLIB_ERROR_CHECKSUM:        //data check has errors
                printf("Humidity : %.2f, \t Temperature : %.2f\t (this value may be
incorrect)\n",dht.humidity,dht.temperature);
                break;
            case DHTLIB_ERROR_TIMEOUT:         //reading DHT times out
                printf("Timeout! \n");
                break;
            case DHTLIB_INVALID_VALUE:         //other errors
                printf("Unknow problem! \n");
                break;
 }
```

Read the value of DHT11_Pin, and store it in the variable, check. Then, if what you get is DHTLIB_OK, it means that the DHT11 sensor works in good condition; accordingly, the printf function is called to print the temperature and humidity.

On the contrary, after finishing the operation of check, if the read value is DHTLIB_ERROR_CHECKSUM，DHTLIB_ERROR_TIMEOUT or DHTLIB_INVALID_VALUE, it means that there is something wrong in the working process of modules. What's more, there appears an error message.

For Python Language Users

Step 2: Go to the folder of the code.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python
```

Step 3: Run the executable file.

```
sudo python3 12.DHT11.py
```

After the code runs, the program will print the temperature and humidity detected by DHT11 on the computer screen.

Code

```
import RPi.GPIO as GPIO
import time
DHTPin = 11
class DHT(object):
    ……
    #Read DHT sensor, store the original data in bits[]
    def readSensor(self,pin,wakeupDelay):
        ……
    #Read DHT sensor, analyze the data of temperature and humidity
    def readDHT11(self):
        ……
def loop():
    dht = DHT(DHTPin)#create a DHT class object
    while(True):
        check = dht.readDHT11()
        if (check is dht.DHTLIB_OK):
            print("Humidity : %.2f,  Temperature : %.2f %(dht.humidity,dht.temperature))
        elif (check is dht.DHTLIB_ERROR_CHECKSUM):
            print("Humidity : %.2f,   Temperature : %.2f   (this value may incorrect)"%(dht.humidity,dht.temperature))
        elif (check is dht.DHTLIB_ERROR_TIMEOUT):
            print("Timeout! ")
        else:
            print("unknow problem! ")
```

```
            time.sleep(2)
if __name__ == '__main__':
    try:
            loop()
    except KeyboardInterrupt:
            pass
            exit()
```

Code Explanation

```
class DHT(object):
    ……
    #Read DHT sensor, store the original data in bits[]
    def readSensor(self,pin,wakeupDelay):
        ……
    #Read DHT sensor, analyze the data of temperature and humidity
    def readDHT11(self):
        ……
```

This class is an open source code for reading sensor values of DHT. It makes it easy for us to use the DHT sensor. Here, we write the invocation functionality directly below the open source class.

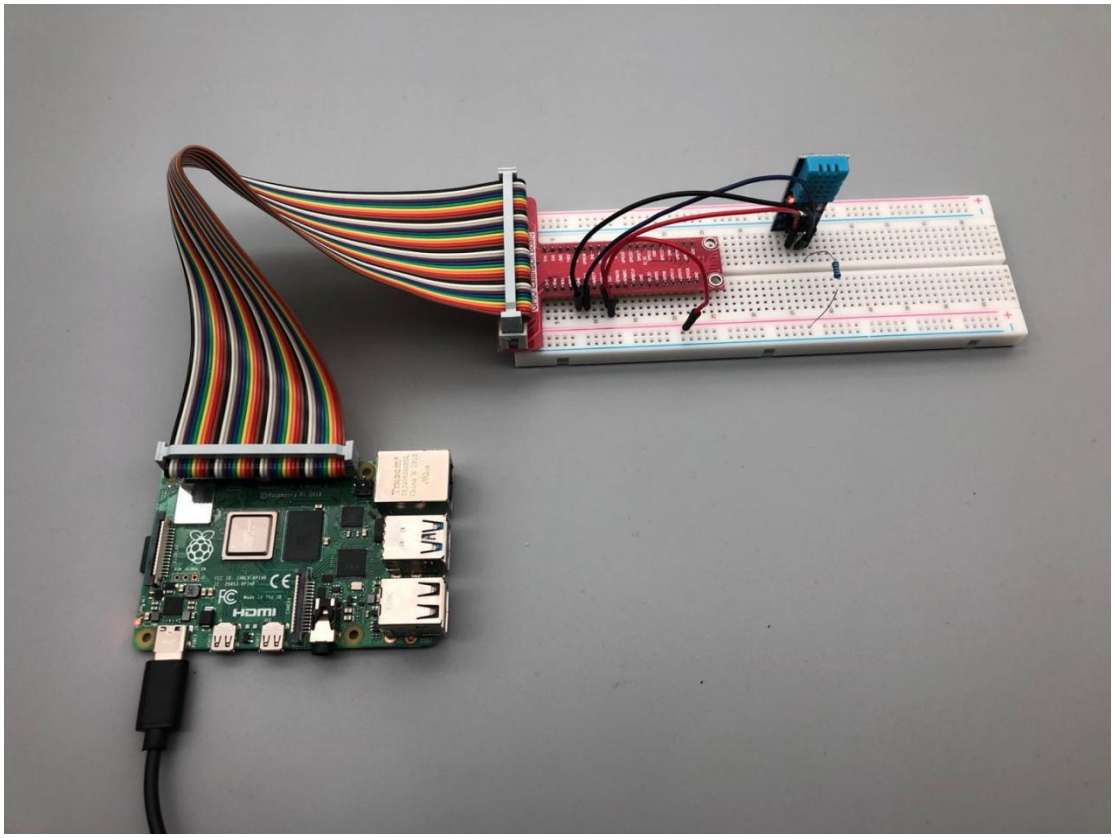
These two functions are the content of the open source file itself, they will read the value of the sensor, and after calculation we can understand the humidity and temperature. They themselves have a return value that monitors the status of the sensor.

```
        def loop():
    dht = DHT(DHTPin)#create a DHT class object
    while(True):
        check = dht.readDHT11()
        if (check is dht.DHTLIB_OK):
            print("Humidity : %.2f,    Temperature : %.2f
            "%(dht.humidity,dht.temperature))
        elif (check is dht.DHTLIB_ERROR_CHECKSUM):
            print("Humidity : %.2f,    Temperature : %.2f    (this value may
incorrect)"%(dht.humidity,dht.temperature))
        elif (check is dht.DHTLIB_ERROR_TIMEOUT):
            print("Timeout! ")
        else:
            print("unknow problem! ")
        time.sleep(2)
```

Read the value of DHT11_Pin, and store it in the variable, check. Then, if what you get is dht.DHTLIB_OK, it means that the DHT11 sensor works in good condition; accordingly, the printf function is called to print the temperature and humidity.

On the contrary, after finishing the operation of check, if the read value is DHTLIB_ERROR_CHECKSUM or DHTLIB_ERROR_TIMEOUT, it means that there is something wrong in the working process of modules. What's more, there appears an error message.

Phenomenon Picture


HDMI