DHT11

Overview

This course will use the DHT11 to measure temperature and humidity values

Experimental Materials:

Raspberry Pi *1

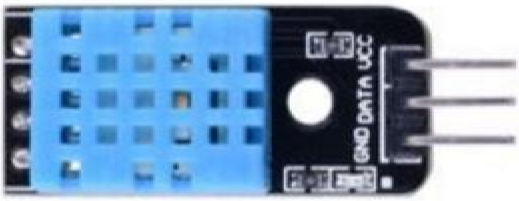
T-type expansion board *1

Breadboard *1

Some DuPont lines

DHT11 *1

Product description：



● Function: DHT11 digital temperature and humidity sensor is a temperature and humidity composite sensor with calibrated digital signal output.

● Applications: Automotive, Data Logger, Consumer Goods, Automatic Control

Technical Parameters:

◆Working voltage: 5VDC/3.3VDC

◆Humidity measurement range: 20%~90%RH

◆Humidity measurement accuracy: ±5%RH

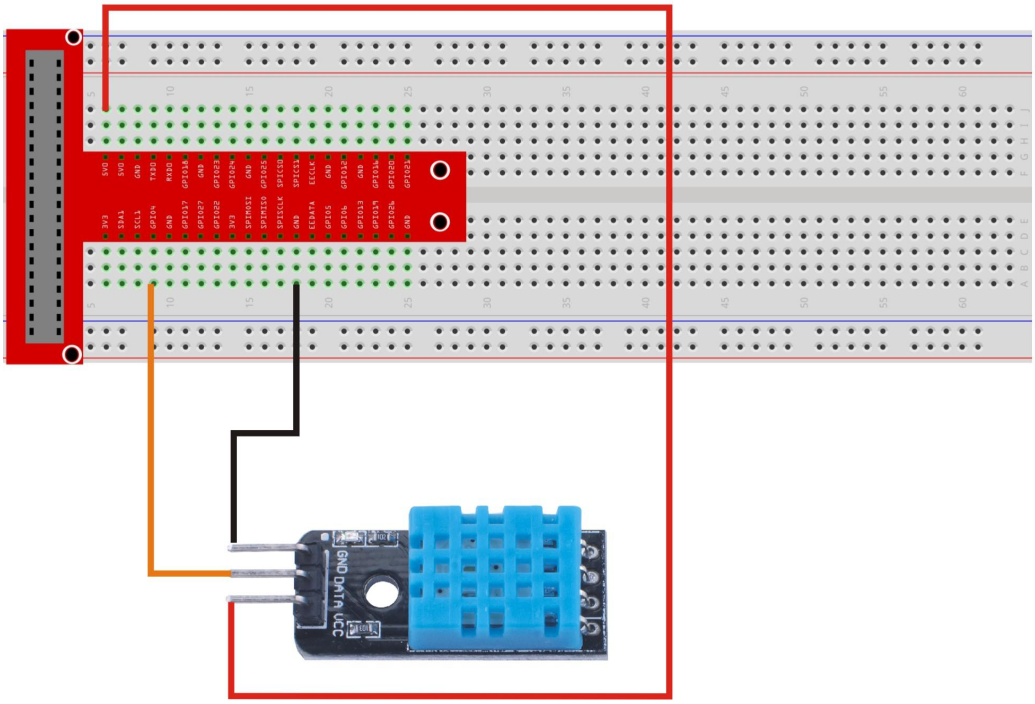
◆Temperature measurement range: 0~50°C

◆Temperature measurement accuracy: ±2°C

◆ Digital signal output

◆ Data port with pull-up resistor

Wiring diagram：



C code:

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

typedef unsigned char uint8;
typedef unsigned int  uint16;
typedef unsigned long uint32;

#define HIGH_TIME 32

int pinNumber = 7;
uint32 databuf;
```

```
uint8 readSensorData(void)
{
    uint8 crc;
    uint8 i;

    pinMode(pinNumber, OUTPUT); // set mode to output
    digitalWrite(pinNumber, 0); // output a high level
    delay(25);
    digitalWrite(pinNumber, 1); // output a low level
    pinMode(pinNumber, INPUT); // set mode to input
    pullUpDnControl(pinNumber, PUD_UP);

    delayMicroseconds(27);
    if(digitalRead(pinNumber) == 0) //SENSOR ANS
    {
        while(!digitalRead(pinNumber)); //wait to high
        for(i=0;i<32;i++)
        {
           while(digitalRead(pinNumber)); //data clock start
           while(!digitalRead(pinNumber)); //data start
           delayMicroseconds(HIGH_TIME);
           databuf*=2;
            if(digitalRead(pinNumber)==1) //1
            {
               databuf++;
            }
        }
        for(i=0;i<8;i++)
         {
           while(digitalRead(pinNumber)); //data clock start
           while(!digitalRead(pinNumber)); //data start
           delayMicroseconds(HIGH_TIME);
           crc*=2;
           if(digitalRead(pinNumber)==1) //1
            {
                 crc++;
            }
        }
        return 1;
    }
    else
    {
        return 0;
    }
}

int main (void)
{

  printf("Use GPIO1 to read data!\n");

  if (-1 == wiringPiSetup())
  {
    printf("Setup wiringPi failed!");
    return -1;
  }

  pinMode(pinNumber, OUTPUT); // set mode to output
  digitalWrite(pinNumber, 1); // output a high level

  printf("Enter OS-------\n");
  while(1)
  {
      pinMode(pinNumber,OUTPUT); // set mode to output
      digitalWrite(pinNumber, 1); // output a high level
      delay(3000);
      if(readSensorData())
      {
          printf("Congratulations ! Sensor data read ok!\n");
          printf("RH:%d.%d%\n",(databuf>>24)&0xff,(databuf>>16)&0xff);
          printf("TMP:%d.%dC\n",(databuf>>8)&0xff,databuf&0xff);
          databuf = 0;
    }
    else
    {
       printf("Sorry! Sensor dosent ans!\n");
       databuf = 0;
    }
  }
  return 0;
}
```

Python code：

```
#!/usr/bin/python
```

```
import RPi.GPIO as GPIO
import time
def collect():
    THdata = []
    channel = 7
    data = []
    GPIO.setmode(GPIO.BOARD)
    time.sleep(2)
    GPIO.setup(channel, GPIO.OUT)
    GPIO.output(channel, GPIO.LOW)
    time.sleep(0.02)
    GPIO.output(channel, GPIO.HIGH)
    GPIO.setup(channel, GPIO.IN)
    while GPIO.input(channel) == GPIO.LOW:
        continue
    while GPIO.input(channel) == GPIO.HIGH:
        continue
    j = 0
    while j < 40:
        k = 0
        while GPIO.input(channel) == GPIO.LOW:
            continue
        while GPIO.input(channel) == GPIO.HIGH:
            k += 1
            if k > 100:
                break
        if k < 8:
            data.append(0)
        else:
            data.append(1)
        j += 1

    # print(“sensor is working.”)
    # print(data)
    humidity_bit = data[0:8]
    humidity_point_bit = data[8:16]
    temperature_bit = data[16:24]
    temperature_point_bit = data[24:32]
    check_bit = data[32:40]
    humidity = 0
    humidity_point = 0
    temperature = 0
    temperature_point = 0
    check = 0
    for i in range(8):
        humidity += humidity_bit[i] * 2 ** (7 - i)
        humidity_point += humidity_point_bit[i] * 2 ** (7 - i)
        temperature += temperature_bit[i] * 2 ** (7 - i)
        temperature_point += temperature_point_bit[i] * 2 ** (7 - i)
        check += check_bit[i] * 2 ** (7 - i)
    tmp = humidity + humidity_point + temperature + temperature_point
    if check == tmp:
        print ″temperature:%d.%d″ %(temperature,temperature_point),″C″,″
humidity :″, humidity, ″%″
        THdata.append(temperature)
        THdata.append(humidity)
        return THdata
    else:
        # print(″wrong″)
        time.sleep(1)
        return collect()

while True:
    rHdata = collect()
    time.sleep(3)
```

Experimental results:

In the directory where the code file is located, execute the following command

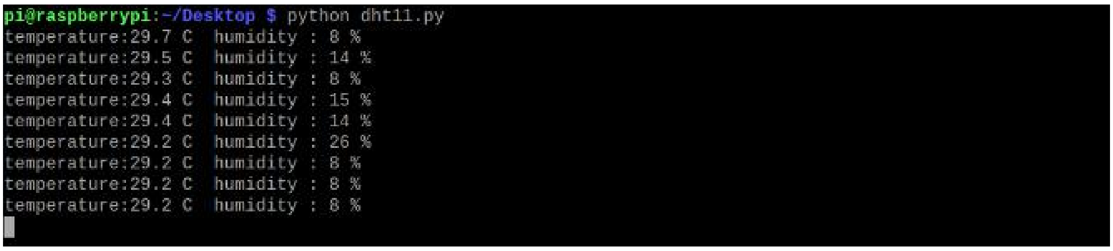
C：

```
gcc -Wall -o dht11 dht11.c -lwiringPi
sudo ./dht11
```

Python:

```
python dht11.py
```

After the instruction is executed, the temperature and humidity can be seen in the instruction window.



```
pi@raspberrypi:~/Desktop $ python dht11.py
temperature:29.7 C  humidity : 8 %
temperature:29.5 C  humidity : 14 %
temperature:29.3 C  humidity : 8 %
temperature:29.4 C  humidity : 15 %
temperature:29.4 C  humidity : 14 %
temperature:29.2 C  humidity : 26 %
temperature:29.2 C  humidity : 8 %
temperature:29.2 C  humidity : 8 %
temperature:29.2 C  humidity : 8 %
```