**Temperature and humidity sensor (DHT11) module**

Overview

The purpose of this course will use the Raspberry Pi control temperature and humidity sensor DHT11 to test the temperature and humidity values in the environment.

The Experimental Materials

Raspberry Pi \* 1

Breadboard \* 1

DHT11 \* 1

Dupont Line

The Preparatory Work

1.Install the python interpreter in your Raspberry Pi system.

2. Install the rpi.gpio library in your Raspberry Pi system.

3. Install wiring Pi library in your Raspberry Pi system.

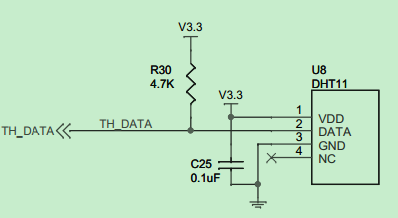
For more details, please refer to the attachment for installing the python interpreter and corresponding libraries in the Raspberry Pi system.

Product Description

Brief Introduction

The DHT11 digital temperature and humidity sensor is a temperature and humidity composite sensor with the output of calibrated digital signals.

It uses dedicated digital module acquisition technology and temperature and humidity sensing technology to ensure product have high reliability and excellent long-term stability. The sensor includes a Resistance humidity sensor and an NTC temperature measuring element.





Characteristic Parameters

**◆**Working voltage: 5VDC/3.3VDC.

**◆**Humidity measurement range: 20%~90%RH.

**◆**Measurement accuracy: ±5%RH.

**◆**Temperature measurement range: 0 ~ 50 ℃

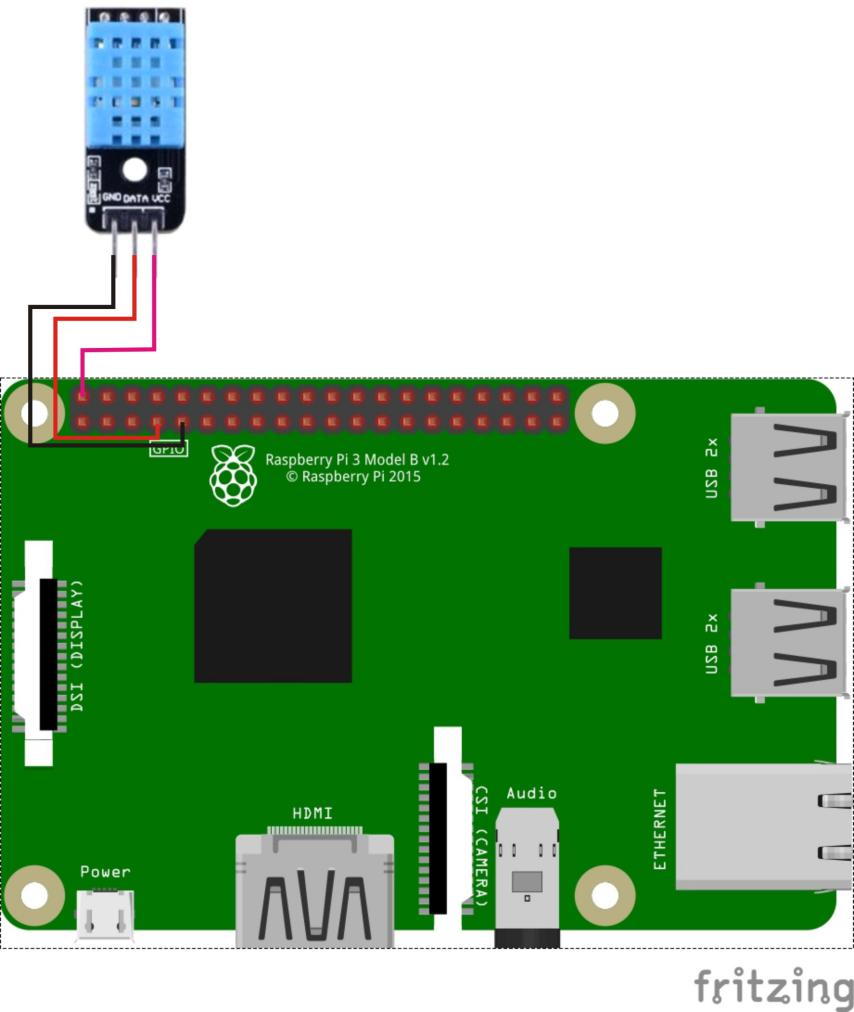
**◆**Temperature measuring accuracy: ±2 ℃

**◆**Digital signal output

**◆**The data port is equipped with a pull resistor.

**◆**It is equipped with 3mm fixed screw hole for easy installation.

Wiring diagram



**The sample code**

1. Python Code

#!/usr/bin/python

import RPi.GPIO as GPIO

import time

channel = 7

data = []

j = 0

humidity = 0

temperature = 0

check = 0

tmp = 0

GPIO.setmode(GPIO.BOARD)

def readSensorData():

global data, j

time.sleep(1)

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

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

def crc():

global temperature, humidity, check, tmp

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:

return 1

else:

return 0

def loop():

readSensorData()

if crc():

print "temperature : ", temperature, ", humidity : " , humidity

else:

print "wrong"

print "temperature : ", temperature, ", humidity : " , humidity, " check : ", check, " tmp : ", tmp

if \_\_name\_\_ == '\_\_main\_\_':

try:

loop()

except KeyboardInterrupt:

GPIO.cleanup()

1. 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;

}

**The experimental phenomena**

After the code runs, the current temperature and humidity will be printed in the command line interface of the raspberry pie system.