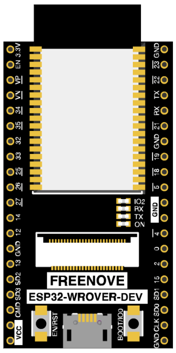
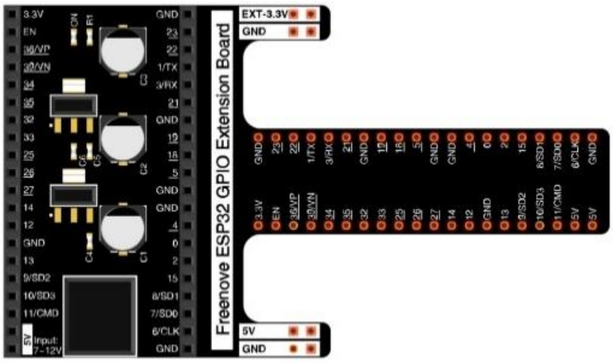
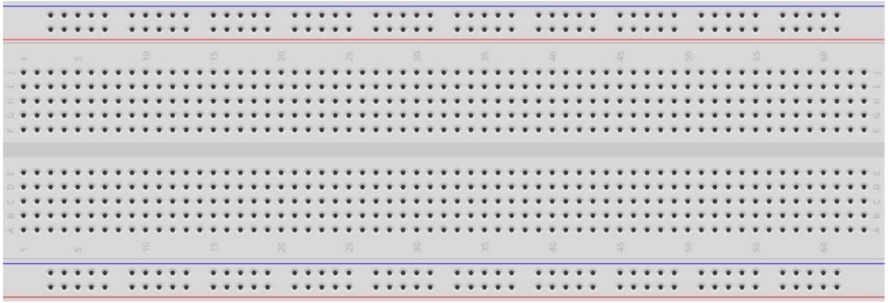
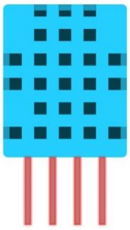
Chapter 24 Hygrothermograph DHT11

In this chapter, we will learn about a commonly used sensor called a Hygrothermograph DHT11.

Project 24.1 Hygrothermograph

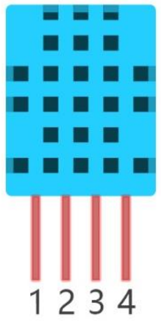
Hygrothermograph is an important tool in our lives to give us data on the temperature and humidity in our environment. In this project, we will use the ESP32 to read temperature and humidity data of the DHT11 Module.

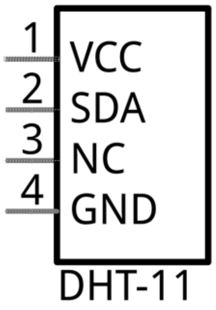
Component List

<table>
<tr><td colspan="2">ESP32-WROVER x1
 </td><td colspan="2">GPIO Extension Board x1
 </td></tr>
<tr><td colspan="4">Breadboard x1
</td></tr>
<tr><td>Jumper M/M x4
 </td><td colspan="2">DHT11 x1
</td><td>Resistor 10kΩ x1
 </td></tr>
</table>

Component knowledge

The temperature & humidity sensor DHT11 is a compound temperature & humidity sensor, and the output digital signal has been calibrated by its manufacturer.





DHT11 uses customized single-line communication protocol, so we can use the library to read data more conveniently.

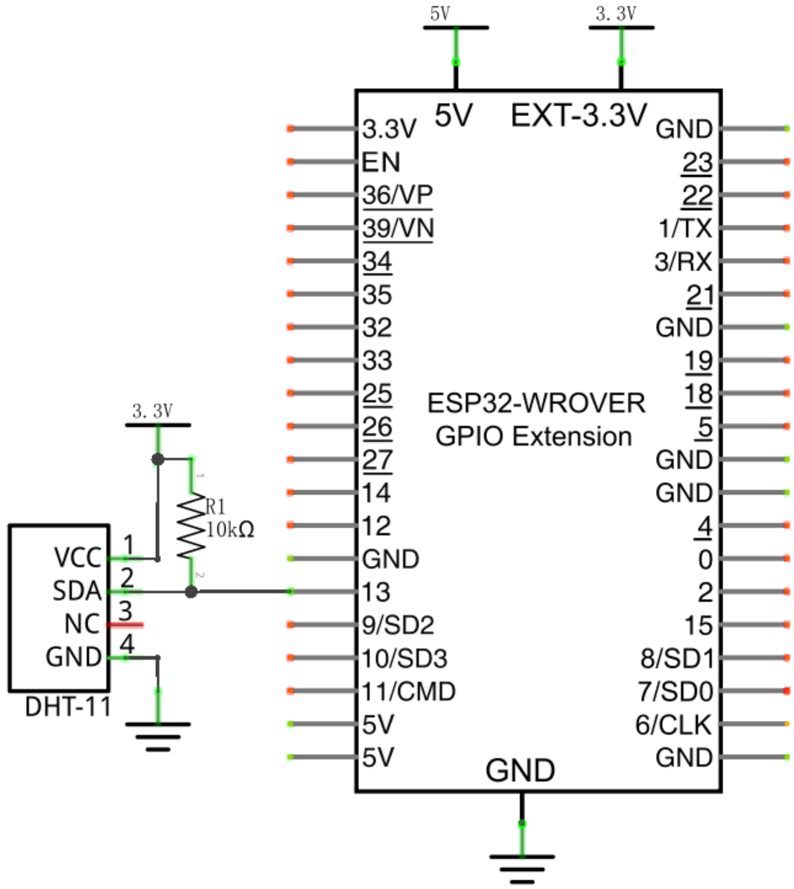
After being powered up, it will initialize in 1s. Its operating voltage is within the range of 3.3V-5.5V.

The SDA pin is a data pin, which is used to communicate with other devices.

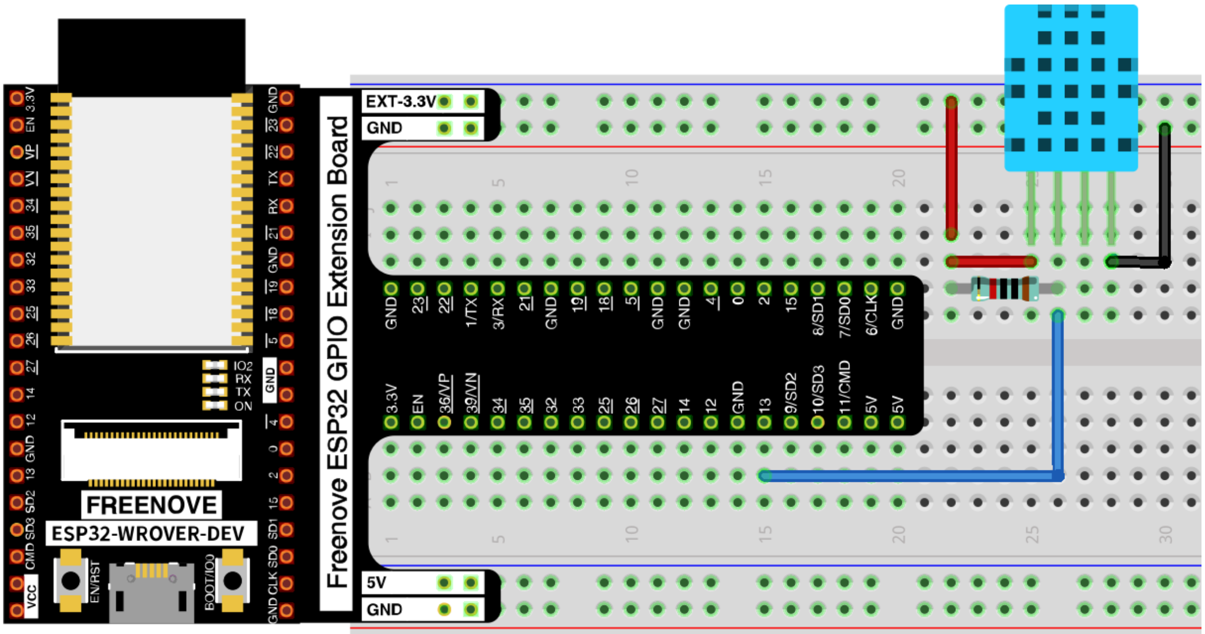
The NC pin (Not Connected Pin) is a type of pin found on various integrated circuit packages. Those pins have no functional purpose to the outside circuit (but may have an unknown functionality during manufacture and test). Those pins should not be connected to any of the circuit connections.

Circuit

Schematic diagram



Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com



Any concerns? ✉ support@freenove.com

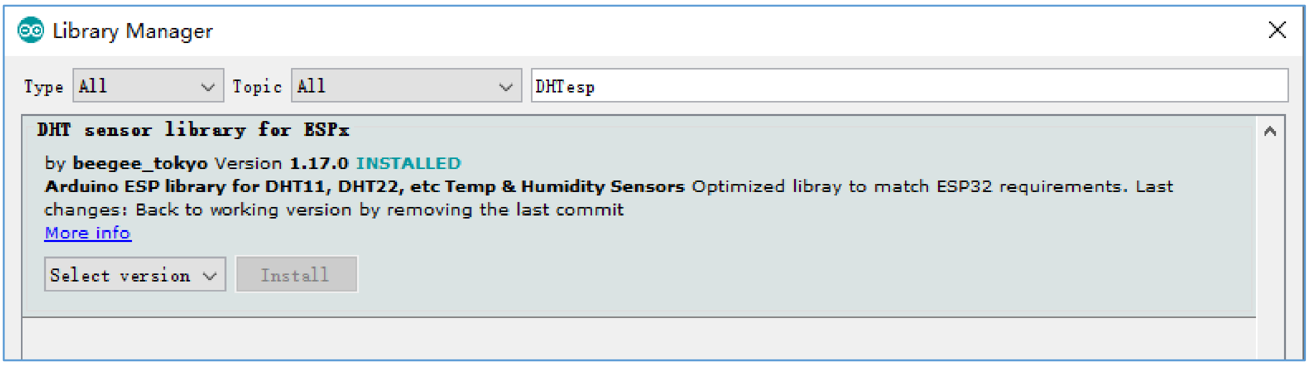
Sketch

How to install the library

The code is used to read the temperature and humidity data of DHT11, and print them out.

We use the third party library DHTesp. If you haven't installed it yet, please do so now. The steps to add third-party libraries are as follows: open arduino->Sketch->Include library-> Manage libraries. Enter " DHTesp " in the search bar and select " DHTesp " for installation.

Refer to the following operations:



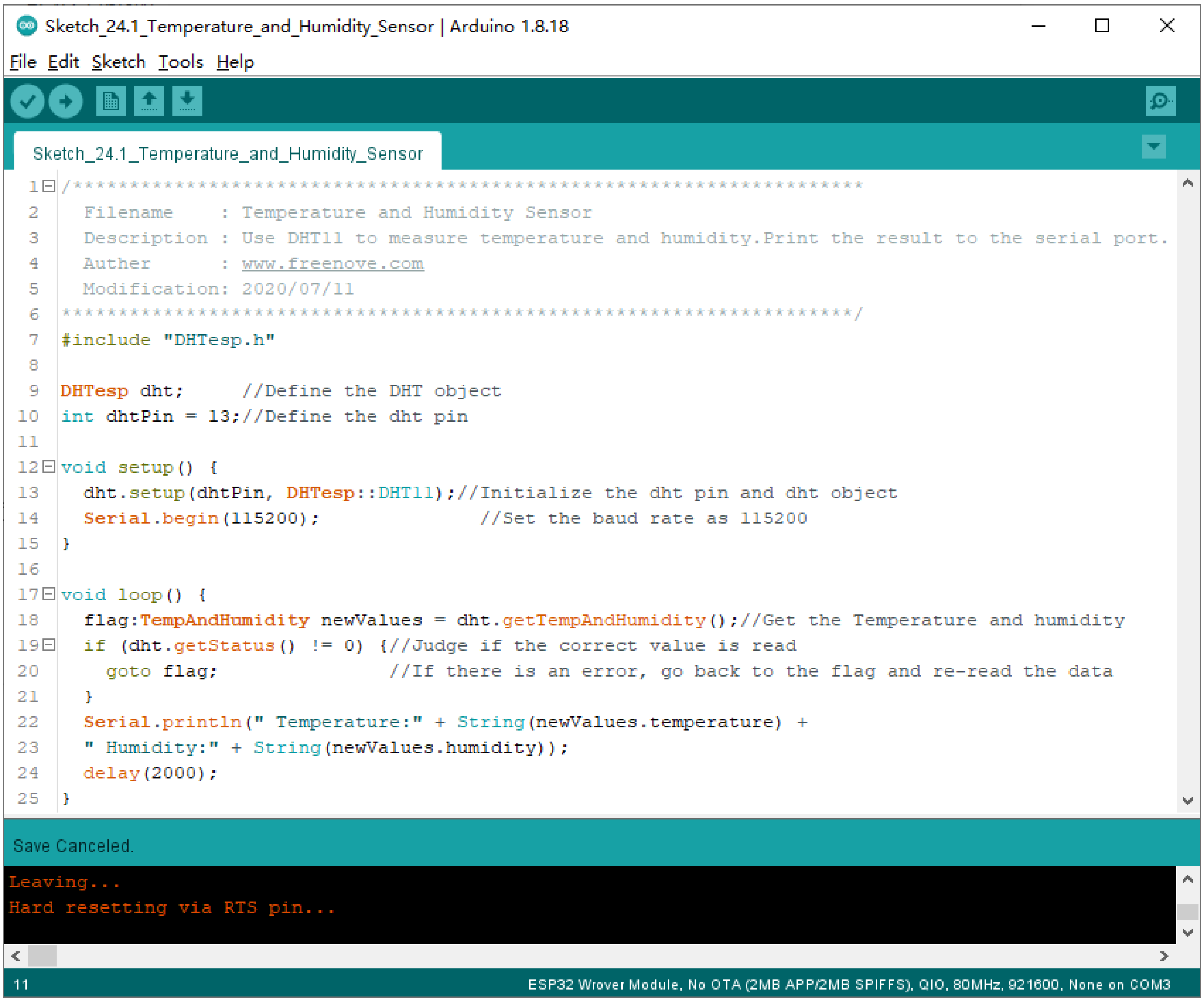
DHTesp

DHT sensor library for ESPx

by beegee_tokyo Version 1.17.0 INSTALLED

Arduino ESP library for DHT11, DHT22, etc Temp & Humidity Sensors Optimized libray to match ESP32 requirements. Last changes: Back to working version by removing the last commit

Sketch_24.1_Temperature_and_Humidity_Sensor


Sketch_24.1_Temperature_and_Humidity_Sensor | Arduino 1.8.18
File Edit Sketch Tools Help
Sketch_24.1_Temperature_and_Humidity_Sensor

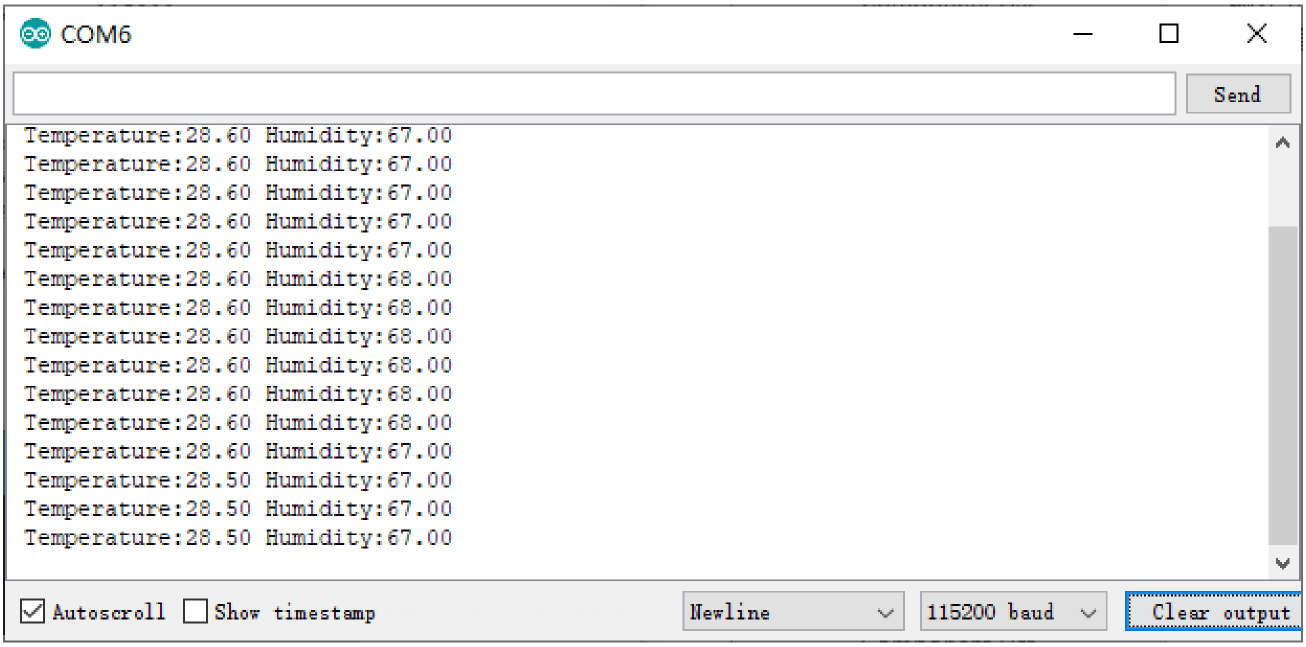
```
/**********************************************************************
  Filename    : Temperature and Humidity Sensor
  Description : Use DHT11 to measure temperature and humidity.Print the result to the serial port.
  Auther      : www.freenove.com
  Modification: 2020/07/11
**********************************************************************/
#include "DHTesp.h"

DHTesp dht;     //Define the DHT object
int dhtPin = 13;//Define the dht pin

void setup() {
  dht.setup(dhtPin, DHTesp::DHT11);//Initialize the dht pin and dht object
  Serial.begin(115200);              //Set the baud rate as 115200
}

void loop() {
  flag:TempAndHumidity newValues = dht.getTempAndHumidity();//Get the Temperature and humidity
  if (dht.getStatus() != 0) {//Judge if the correct value is read
    goto flag;                //If there is an error, go back to the flag and re-read the data
  }
  Serial.println(" Temperature:" + String(newValues.temperature) +
  " Humidity:" + String(newValues.humidity));
  delay(2000);
}
```

Save Canceled.

```
Leaving...
Hard resetting via RTS pin...
```

11 ESP32 Wrover Module, No OTA (2MB APP/2MB SPIFFS), QIO, 80MHz, 921600, None on COM3

Compile and upload the code to the ESP32-WROVER, turn on the serial monitor, and set the baud rate to 115200. Print out data of temperature and humidity sensor via the serial port.



```
Temperature:28.60 Humidity:67.00
Temperature:28.60 Humidity:67.00
Temperature:28.60 Humidity:67.00
Temperature:28.60 Humidity:67.00
Temperature:28.60 Humidity:67.00
Temperature:28.60 Humidity:68.00
Temperature:28.60 Humidity:68.00
Temperature:28.60 Humidity:68.00
Temperature:28.60 Humidity:68.00
Temperature:28.60 Humidity:68.00
Temperature:28.60 Humidity:68.00
Temperature:28.60 Humidity:67.00
Temperature:28.50 Humidity:67.00
Temperature:28.50 Humidity:67.00
Temperature:28.50 Humidity:67.00
```

☑ Autoscroll □ Show timestamp Newline 115200 baud

The following is the program code:

```
#include "DHTesp.h"

DHTesp dht;      //Define the DHT object
int dhtPin = 13;//Define the dht pin

void setup() {
  dht.setup(dhtPin, DHTesp::DHT11);//Initialize the dht pin and dht object
  Serial.begin(115200);              //Set the baud rate to 115200
}

void loop() {
  flag:TempAndHumidity newValues = dht.getTempAndHumidity();//Get the Temperature and humidity
  if (dht.getStatus() ! = 0) {                              //Judge if the correct value is
read
    goto flag;                                              //If there is an error, go back to
the flag and re-read the data
  }
  Serial.println(" Temperature:" + String(newValues.temperature) +
  " Humidity:" + String(newValues.humidity));
  delay(2000);
}
```

In this project code, we use a third party library, DHTesp, and we need to define the objects for it first; Otherwise we could not use its functionality.

```
1   #include "DHTesp.h"
3   DHTesp dht;      //Define the DHT object
```

Initialize the connection pin of DHT and select the type of temperature and humidity sensor as DHT11. If the temperature and humidity sensor is DHT12, we can also change it to DHT12.

```
7   dht.setup(dhtPin, DHTesp::DHT11);//Initialize the dht pin and dht object
```

Due to the use of the single-line protocol, data may be lost in the transmission process. So each time when getting the data of the temperature and humidity sensor, we need to call the getStatus function to determine whether the data is normal. If not, use goto to go back to line 12 and re-execute the program.

```
12  flag:TempAndHumidity newValues = dht.getTempAndHumidity();//Get the Temperature and
13  humidity
14    if (dht.getStatus() ! = 0) {       //Judge if the correct value is read
15      goto flag;                      //If there is an error, go back to the flag and re-read
16  the data
17    }
```

Get the temperature and humidity data and store it in a TempAndHumidity class called newValues.

```
12  TempAndHumidity newValues = dht.getTempAndHumidity();//Get the Temperature and humidity
```

Reference

`class DHTesp`

Make sure that the library and header files are added before using the object every time.

setup(Pin, DHTesp::DHTxx): Select the type of DHTxx and associate Pin with the DHTesp class.

Parameter 1: the pin to be associated.

Parameter 2: select the type of sensor, DHT11 or DHT12.

getTempAndHumidity(): Obtain temperature and humidity data. The received data must be stored in the 'TempAndHumidity' class.

getStatus(): To judge whether the obtained data format is normal, the return value of 0 means the data is normal, and the return value of non-0 means the data is abnormal or the data fails to be obtained.