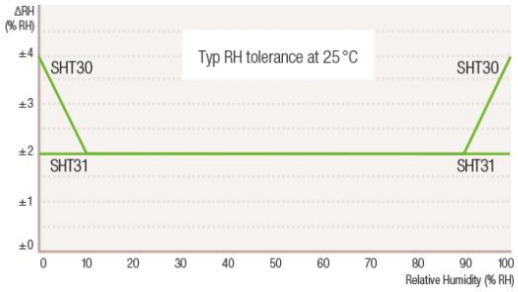
SKU:SEN0385

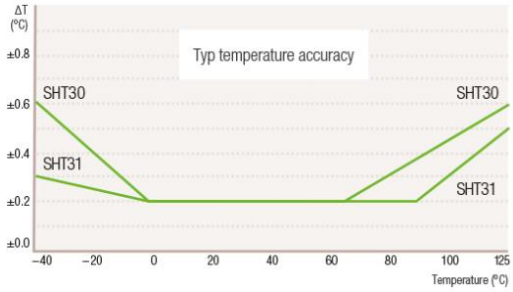
Introduction

This is a SHT31 Temperature & Humidity sensor housed in a weather-proof enclosure. The sensor is designed to be waterproof so you can use it safely in outdoor projects. Please note that although the sensor is waterproof, it is not suggested to submerge it in water. For in-water measurement, DS18B20 Sensor is highly recommended.

Thanks to Sensirion's CMOSens® technology, highly integrated capacitive humidity sensing components and band-gap temperature sensing components, the SHT31 offers high reliability and long-term stability with low power consumption, fast response and strong anti-interference ability. The sensor supports I2C communication, and is compatible with 3.3V/5V controllers like Arduino, micro:bit, ESP32.

SHT31 is the standard version of the SHT3x series. It provides humidity accuracy ± 2%RH@0% RH~100% RH (at 25°C), and temperature accuracy±0.2°C@ 0°C ~90°C (typical).



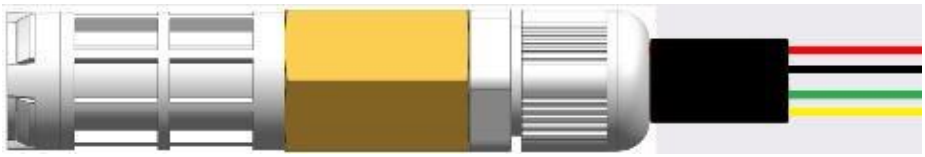


Specification

- Operating Voltage: 3.3~5V
- Operating Current: <1.5mA
- Humidity Detection Range: 0%RH~100%RH
- Humidity Accuracy: ±2%RH

- Temperature Detection Range: -40℃~125℃
- Temperature Accuracy: ±0.2℃
- Communication: I2C
- Cable Length: about 1m

PinOut



Color	Name	Description
Red	VCC	+
Black	GND	-
Green	SDA	Data line
Yellow	SCL	Clock line

Tutorial

Note: The I2C address of this sensor is 0x44. Please make sure the I2C address is correct when using it.

Requirements

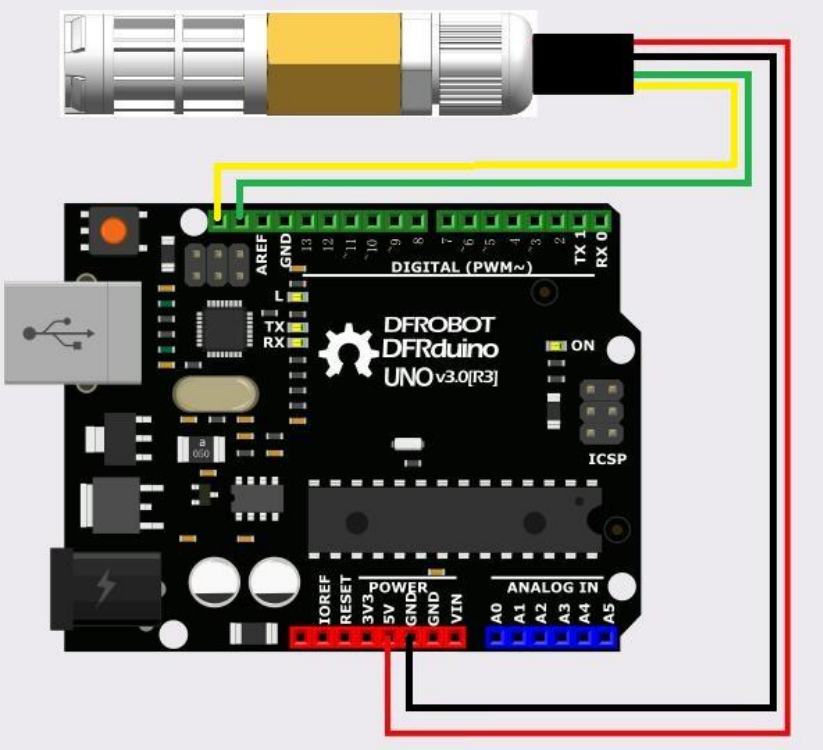
- **Hardware**
  - DFRduino UNO R3 (or similar) x 1
  - SHT31 Temperature & Humidity Sensor (Weather-proof) x1
  - Jumper wires

- **Software**
  - Arduino IDE
  - Download and install the **SHT3X Library** (About how to install the library?)

- API Function List

```
/**
 * @brief Get the measured temperature (in degrees Celsius).
 * @return Return the temperature data of float type.
 */
float getTemperatureC();

/**
 * @brief Get the measured temperature (in degrees Fahrenheit).
 * @return Return the temperature data of float type.
 */
float getTemperatureF();

/**
 * @brief Get measured humidity(%RH).
 * @return Return the humidity data of float type.
 */
float getHumidityRH();
```

Connection Diagram



Sample Code 1-Single Measurement Mode

In single measurement mode, the sensor collects data every time the controller board sends out the data collecting command. The power consumption could be very low in this mode since users can read data according to their needs.

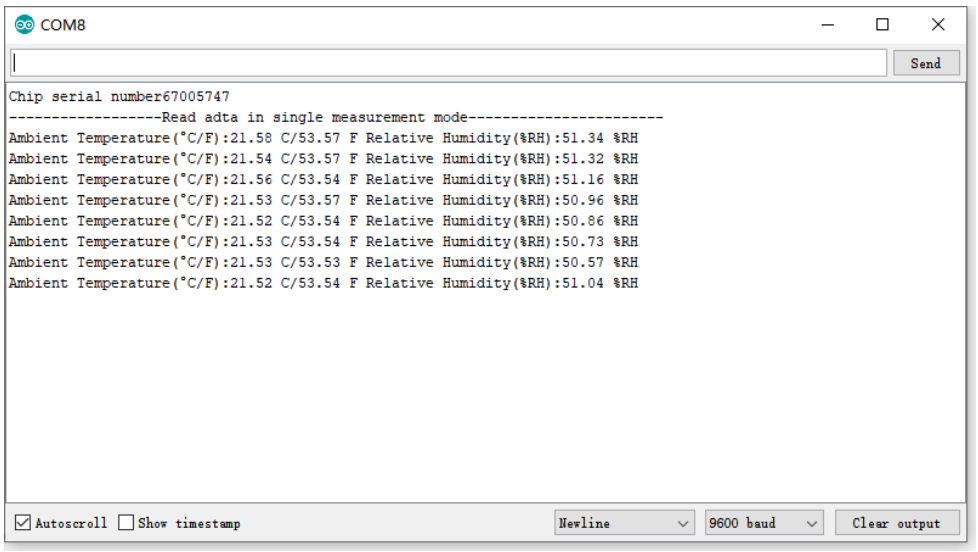
```
/*!
     * @brief Construct the function
     * @param pWire I2C bus pointer object and construction device, can both pass or
not pass parameters, Wire in default.
     * @param address Chip I2C address, two optional addresse.
    */
    /*!
     * @file singleMeasurement.ino
     * @brief Read ambient temperature (C/F) and relative humidity (%RH) in single-
read mode.
     * @n Experimental phenomenon: the chip defaults in this mode, we need to send
instructions to enable the chip collect data,
     * which means the repeatability of the read needs to be set (the difference
between the data measured by the chip under the same measurement conditions)
     * then read the temperature and humidity data and print the data in the serial
port.
     * @n Single measure mode: read data as needed, power consumption is relatively
low, the chip idle state only costs 0.5mA.
     * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
     * @licence     The MIT License (MIT)
     * @author [fengli](li.feng@dfrobot.com)
     * @version  V1.0
     * @date  2019-08-21
     * @get from https://www.dfrobot.com
     * @url https://github.com/DFRobot/DFRobot_SHT3x
    */

    #include <DFRobot_SHT3x.h>

    /*!
     * @brief Construct the function
     * @param pWire I2C bus pointer object and construction device, both can pass or
not pass parameters,
     * Wire in default.
     * @param address Chip I2C address, two optional addresses 0x44 and 0x45(0x45 in
default).
     * @param RST RST Chip reset pin, 4 in default.
     * @n I2C address is determined by the pin addr on the chip.
     * @n When the ADR is connected to VDD, the chip I2C address is 0x45.
     * @n When the ADR is connected to GND, the chip I2C address is 0x44.
     */

    DFRobot_SHT3x sht3x(&Wire,/*address=*/0x44,/*RST=*/4);
    //DFRobot_SHT3x   sht3x;

    void setup() {
      Serial.begin(9600);
      //Initialize the chip
      while (sht3x.begin() != 0) {
        Serial.println("Failed to Initialize the chip, please confirm the wire
connection");
        delay(1000);
      }
      /**
```

```
       * readSerialNumber Read the serial number of the chip.
       * @return Return 32-digit serial number.
       */
      Serial.print("Chip serial number");
      Serial.println(sht3x.readSerialNumber());

      /**
       * softReset Send command resets via I2C, enter the chip's default mode single-
measure mode,
       * turn off the heater, and clear the alert of the ALERT pin.
       * @return Read the register status to determine whether the command was
executed successfully,
       * and return true indicates success.
       */
       if(!sht3x.softReset()){
         Serial.println("Failed to Initialize the chip....");
       }

      /**
       * heaterEnable(): Turn on the heater inside the chip to enable the sensor get
correct humidity value in wet environments.
       * @return Read the status of the register to determine whether the command was
executed successfully,
       * and return true indicates success.
       * @note Heaters should be used in wet environments, and other cases of use
will result in incorrect readings
       */

      //if(!sht3x.heaterEnable()){
      // Serial.println("Failed to turn on the heater....");
      //}
      Serial.println("------------------Read adta in single measurement mode---------
--------------");
    }

    void loop() {
      Serial.print("Ambient Temperature(°C/F):");
      /**
       * getTemperatureC Get the meansured temperature(℃).
       * @return Return float temperature data.
       */
      Serial.print(sht3x.getTemperatureC());
      Serial.print(" C/");
      /**
       * getTemperatureF:Get the meansured temperature(℉).
       * @return Return float temperature data.
       */
      Serial.print(sht3x.getTemperatureF());
      Serial.print(" F ");
      Serial.print("Relative Humidity(%RH):");
      /**
       * getHumidityRH: Get the meansured humidity (%RH)
       * @return Return float humidity data
       */
      Serial.print(sht3x.getHumidityRH());
```

```
      Serial.println(" %RH");

      /**
       * @brief Get temperature and humidity data in single measurement mode.
       * @param repeatability Set repeatability to read temperature and humidity data
with the type eRepeatability_t.
       * @note  Optional parameters:
                  eRepeatability_High /**In high repeatability mode, the humidity
repeatability is 0.10%RH, the temperature repeatability is 0.06°C
                  eRepeatability_Medium,/**In medium repeatability mode, the
humidity repeatability is 0.15%RH, the temperature repeatability is 0.12°C.
                  eRepeatability_Low, /**In low repeatability mode, the humidity
repeatability is0.25%RH, the temperature repeatability is 0.24°C
       * @return Return a structure containing celsius temperature (°C), Fahrenheit
temperature (°F), relative humidity(%RH), status code.
       * @n Return 0 indicates right data return.
      DFRobot_SHT3x::sRHAndTemp_t data =
sht3x.readTemperatureAndHumidity(sht3x.eRepeatability_High);
      if(data.ERR == 0){
        Serial.print("Ambient Temperature(°C/F):");
        Serial.print(data.TemperatureC);
        Serial.print(" C/");
        Serial.print(data.TemperatureF);
        Serial.print(" F ");
        Serial.print("Relative Humidity(%RH):");
        Serial.print(data.Humidity);
        Serial.println(" %RH");
      }
      */
      delay(1000);
    }
```

Copy

Result 1



Sample Code 2- Period Measurement Mode

In period measurement mode, the sensor collects data at the user-set frequency.

```
/*!
     * @file periodicDataReading.ino
     * @brief Read ambient temperature (C/F) and relative humidity (%RH) in cycle
read mode.
     * @n Experimental phenomenon: Before we start, please set the read frequency and
repeatability of the read
     * (the difference between the data measured by the chip under the same
measurement conditions),
     * and enter the periodic read mode, and then read the temperature and humidity
data.
     * @n The temperature and humidity data will be printed at the serial port, after
10 seconds of operation.
     * @n It will exit the cycle mode and enter 2 measurement mode: Single
measurement mode and Cycle measurement mode.
     * @n Single measurement mode: reflect the difference between the two modes of
reading data.
     * @n Cycle measurement mode: the chip periodically monitors temperature and
humidity, only in this mode the ALERT pin will work.
     * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
     * @licence     The MIT License (MIT)
     * @author [fengli](li.feng@dfrobot.com)
     * @version  V1.0
     * @date  2019-08-20
     * @get from https://www.dfrobot.com
     * @url https://github.com/DFRobot/DFRobot_SHT3x
    */

    #include <DFRobot_SHT3x.h>

    /*!
     * @brief Construct the function
     * @param pWire I2C bus pointer object and construction device, can both pass or
not pass parameters, Wire in default.
     * @param address Chip I2C address, two optional addresses 0x44 and 0x45(0x45 in
default).
     * @param RST Chip reset pin, 4 in default.
     * @n The I2C address is determined by the pin addr on the chip.
     * @n When the ADR is connected to VDD, the chip I2C address is 0x45.
     * @n When the ADR is connected to GND, the chip I2C address is 0x44.
     */
    DFRobot_SHT3x sht3x(&Wire,/*address=*/0x44,/*RST=*/4);

    //DFRobot_SHT3x sht3x;

    void setup() {

      Serial.begin(9600);
        //Initialize the chip to detect if it can communicate properly.
      while (sht3x.begin() != 0) {
```

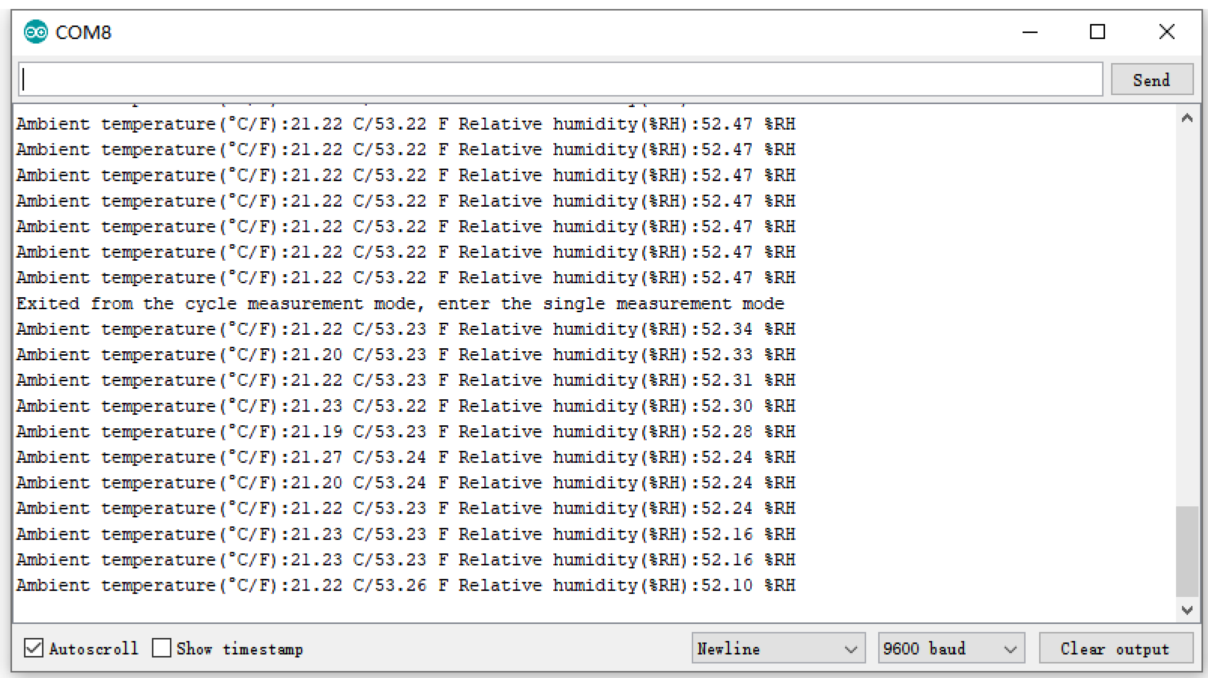
```
        Serial.println("Failed to initialize the chip, please confirm the chip
connection");
        delay(1000);
      }

      /**
       * readSerialNumber Read the serial number of the chip
       * @return Return 32-digit serial number
       */
      Serial.print("chip serial number: ");
      Serial.println(sht3x.readSerialNumber());
      /**
       * softReset Send command resets via I2C, enter the chip's default mode single-
measure mode,
       * turn off the heater, and clear the alert of the ALERT pin.
       * @return Read the status register to determine whether the command was
executed successfully,
       * and return true indicates success.
       */
       if(!sht3x.softReset()){
         Serial.println("Failed to reset the chip");
       }

      /**
       * pinReset Reset through the chip's reset pin, enter the chip's default mode
single-measure mode,
       * turn off the heater, and clear the alert of the ALERT pin.
       * @return The status register has a data bit that detects whether the chip has
been reset,
       * and return true indicates success.
       * @note When using this API, the reset pin of the chip nRESET should be
connected to RST (default to pin4) of arduino.
       */
      //if(!sht3x.pinReset()){
        //Serial.println("Failed to reset the chip");
      //}

      /**
       * heaterEnable() Turn on the heater inside the chip so that the sensor can
have accurate humidity data even in humid environment.
       * @return Read the status register to determine whether the command was
executed successfully, and return true indicates success.
       * @NOTE Heaters should be used in wet environment, and other cases of use will
result in incorrect readings.
       */
      //if(!sht3x.heaterEnable()){
        // Serial.println("Failed to turn on the heater");
      //}
      /**
       * startPeriodicMode Enter cycle measurement mode and set repeatability and
read frequency.
       * @param measureFreq Read the eMeasureFrequency_t data frequency.
       * @note  Selectable parameters:
                  eMeasureFreq_Hz5,   /**the chip collects data in every 2s
                  eMeasureFreq_1Hz,   /**the chip collects data in every 1s
```

```
                    eMeasureFreq_2Hz,   /**the chip collects data in every 0.5s
                    eMeasureFreq_4Hz,   /**the chip collects data in every 0.25s
                    eMeasureFreq_10Hz   /**the chip collects data in every 0.1s
       * @param repeatability Read the repeatability of temperature and humidity
data, the default parameter is eRepeatability_High.
       * @note  Optional parameters:
                    eRepeatability_High /**In high repeatability mode, the humidity
repeatability is 0.10%RH, the temperature repeatability is 0.06°C
                    eRepeatability_Medium,/**In medium repeatability mode, the
humidity repeatability is 0.15%RH, the temperature repeatability is 0.12°C.
                    eRepeatability_Low, /**In low repeatability mode, the humidity
repeatability is0.25%RH, the temperature repeatability is 0.24°C
       * @return Read the status of the register to determine whether the command was
executed successfully, and return true indicates success
       */
      if(!sht3x.startPeriodicMode(sht3x.eMeasureFreq_1Hz)){
        Serial.println("Failed to enter the periodic mode");
      }
      Serial.println("------------------Read data in cycle measurement mode----------
-------------");
    }

    void loop() {

      Serial.print("Ambient temperature(°C/F):");
      /**
       * getTemperatureC Get the measured temperature (in degrees Celsius).
       * @return Return the float temperature data.
       */
      Serial.print(sht3x.getTemperatureC());
      Serial.print(" C/");
      /**
       * getTemperatureF Get the measured temperature (in degrees Fahrenheit).
       * @return Return the float temperature data.
       */
      Serial.print(sht3x.getTemperatureF());
      Serial.print(" F ");
      Serial.print("Relative humidity(%RH):");
      /**
       * getHumidityRH Get measured humidity(%RH)
       * @return Return the float humidity data
       */
      Serial.print(sht3x.getHumidityRH());
      Serial.println(" %RH");
      //Please adjust the frequency of reading according to the frequency of the chip
collection data.
      //The frequency to read data must be greater than the frequency to collect the
data, otherwise the returned data will go wrong.
      delay(100);
      if(millis() > 10000 && millis() < 10200){
        /**
         * stopPeriodicMode() Exit from the cycle read data
         * @return Read the status of the register to determine whether the command
was executed successfully,
         * and return true indicates success.
```

```
         */
        sht3x.stopPeriodicMode();
        Serial.println("Exited from the cycle measurement mode, enter the single
measurement mode");
      }
      /**
       * readTemperatureAndHumidity Get temperature and humidity data in cycle
measurement mode and use structures to receive data
       * @return Return a structure containing celsius temperature (°C), Fahrenheit
temperature (°F), relative humidity (%RH), status code.
       * @n A status of 0 indicates that the right return data.

      DFRobot_SHT3x::sRHAndTemp_t data = sht3x.readTemperatureAndHumidity();
      if(data.ERR == 0){
        Serial.print("ambient temperature(°C/F):");
        Serial.print(data.TemperatureC);
        Serial.print("C/");
        Serial.print(data.TemperatureF);
        Serial.print("F");
        Serial.print("relative humidity(%RH):");
        Serial.print(data.Humidity);
        Serial.println("%RH");
      }
      */
    }
```

Result 2

Serial print the temperature and humidity information in period measurement mode for 10s, then exit from this mode and enter single measurement mode, and print the information.



FAQ

For any questions, advice or cool ideas to share, please visit the **DFRobot Forum**.

More Documents

- SHT3x Datasheet
- SHT3x Instruction