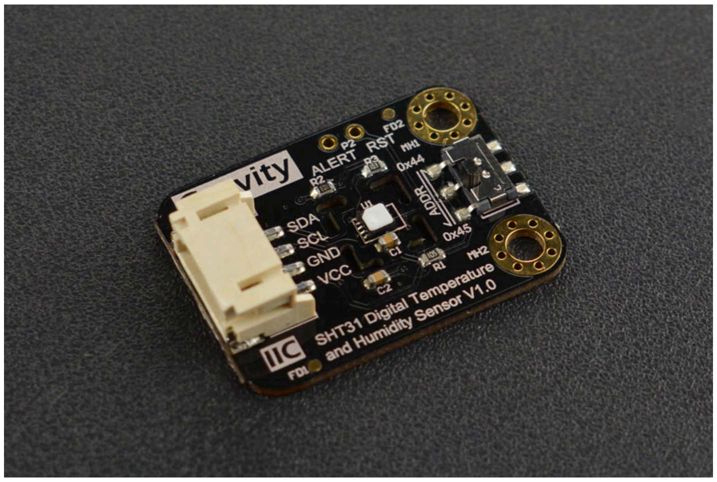


(https://www.dfrobot.com/product-2014.html)

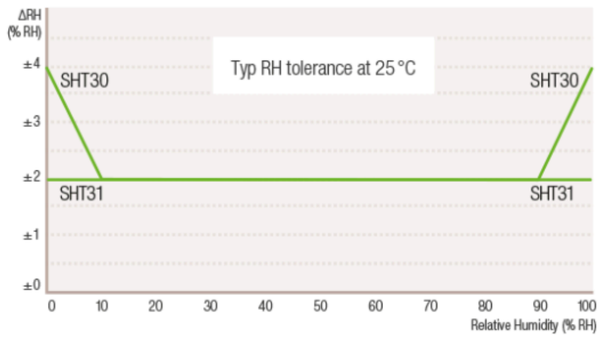
Introduction

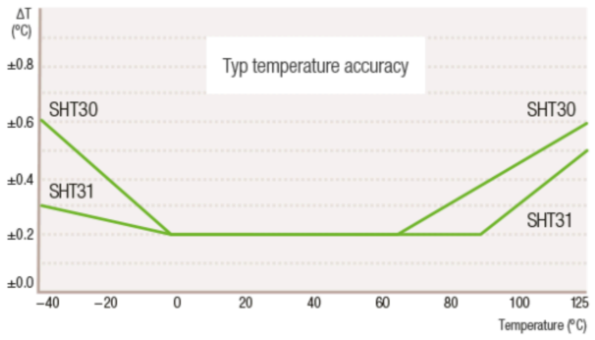
Gravity: SHT31-F digital temperature and humidity sensor is based upon the Sensirion SHT31-F sensor IC. Thanks to Sensirion’s CMOSens® technology, highly integrated capacitive humidity sensing components and band-gap temperature sensing components, the SHT31-F offers high reliability and long-term stability with low power consumption, fast response and strong anti-interference ability. The sensor supports IIC communication, and is compatible with 3.3V/5V controllers like Arduino, micro:bit, ESP32. It is easy to achieve precise and high-reliability temperature and humidity sensing for urban environment monitoring, intelligent buildings, industrial automation, smart home and other Internet of Things applications.

SHT31-F 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℃-90℃ (typical).

The SHT31-F chip features a PTFE membrane dedicated to protecting the sensor opening from liquids and dust according to IP67. It thus allows the sensor to work under harsh environmental conditions, where spray water and high exposure to dust might be challenging for accurate sensor performance. Due to minimal package volume and the membrane's high water vapor permeability, the response time of the relative humidity and temperature signal is identical to

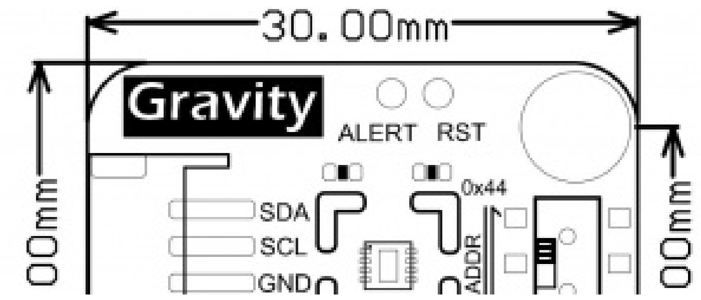
the value achieved by the uncovered sensor. Although the membrane option provides perfect protection against water and dust ingress, it does not in general prevent contamination from volatile chemicals.

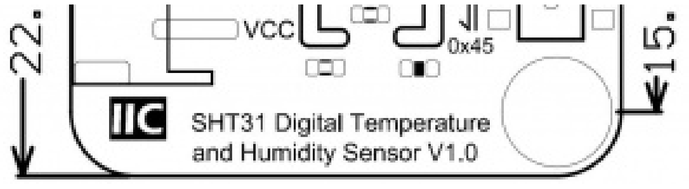




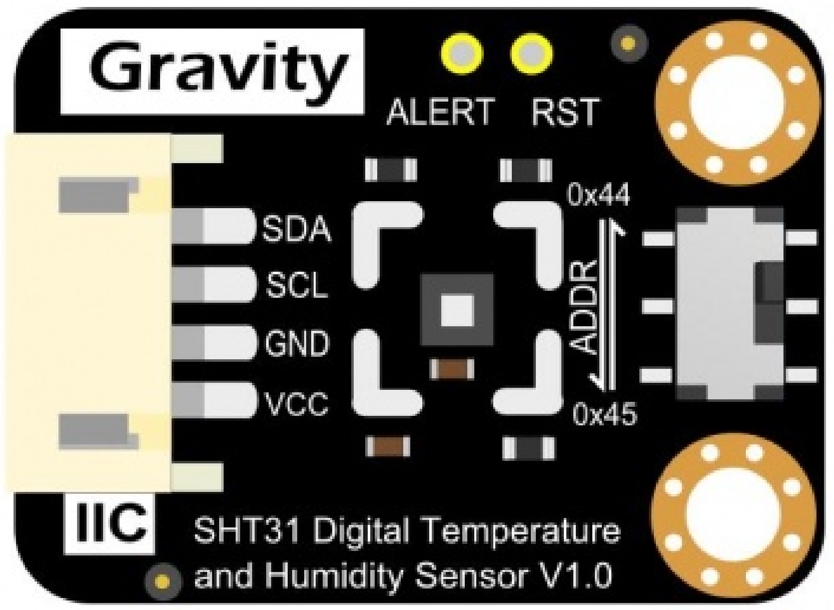
Specification

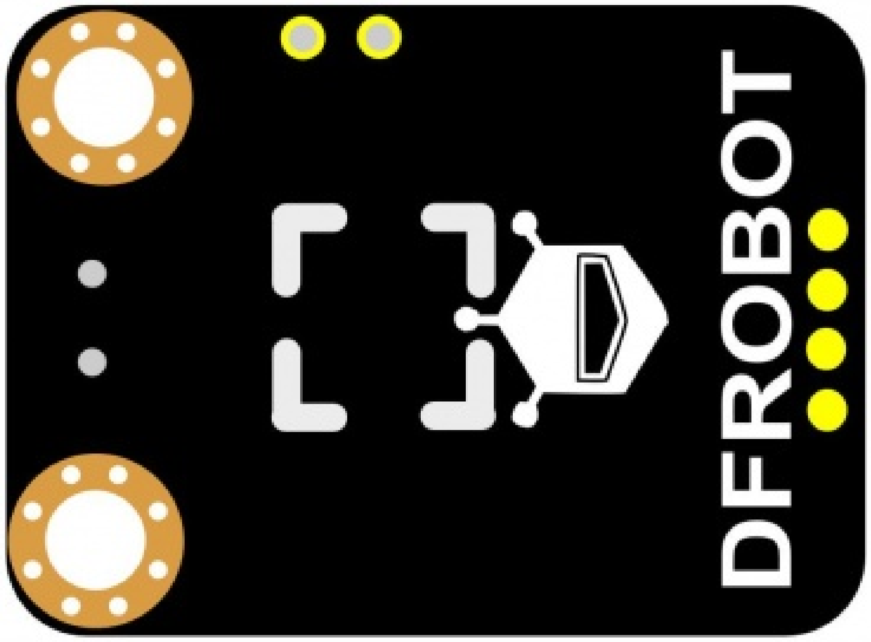
- Operating Voltage: 3.3~5.5V
- Operating Current: <1.5mA
- Humidity Accuracy: ±2%RH
- Humidity Detection Range: 0%RH~100%RH
- Temperature Accuracy: ±0.2℃
- Temperature Detection Range: -40℃~125℃
- Communication port: Gravity-IIC 4pin
- Outline Dimension: 32x22mm/1.26x0.87"
- Mounting Hole Size: 3mm/0.12"
- Mounting Hole Pitch: 15mm/0.59"





Board Overview





Name	Description
SDA	Data line

SCL	Clock Line
GND	Negative

Name	Description
VCC	Positive
INT	Alarm Interrupt
RST	Reset

Tutorial

Note: please pay attention to the hardware IIC addresss before using, the default initial IIC address of the program is 0x45. These two addresses should be the same, otherwise the chip cannot be successfully initialized.

Requirements

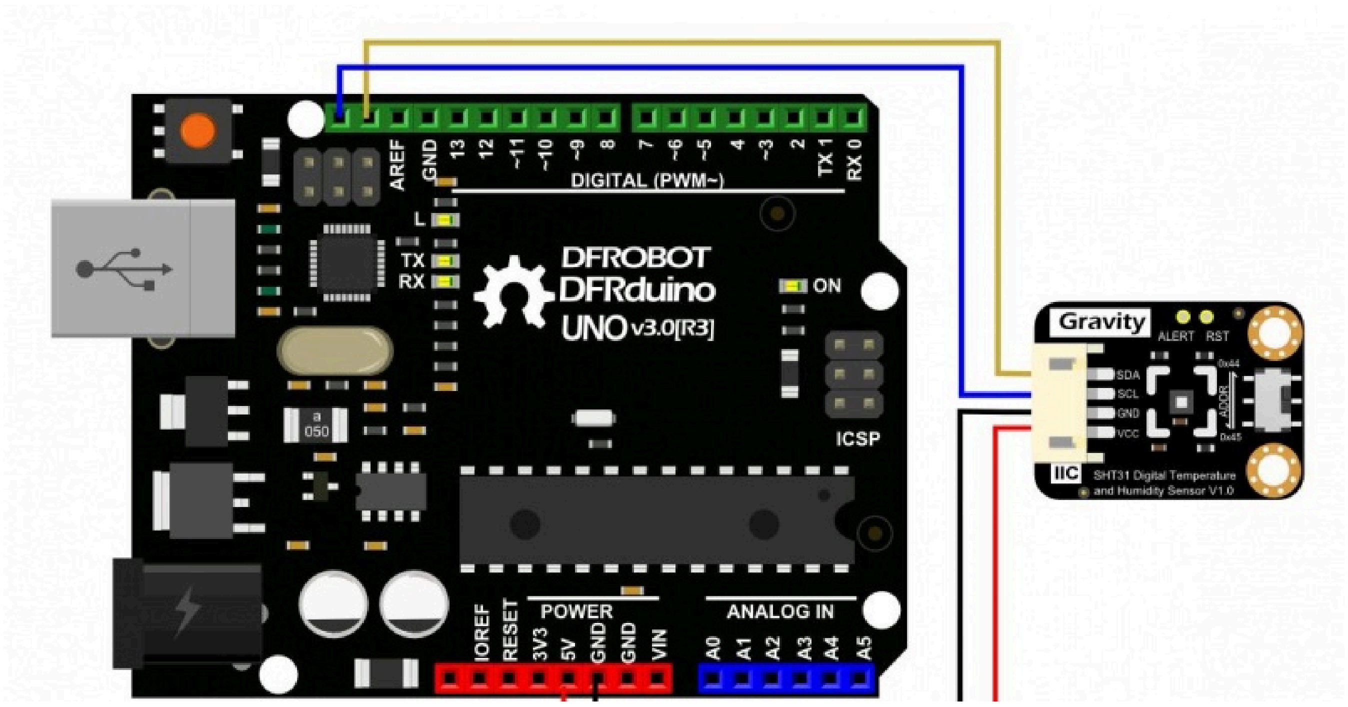
- **Hardware**
  - DFRduino UNO R3 (https://www.dfrobot.com/product-838.html) (or similar) x 1
  - Gravity: SHT31-F Digital Temperature and Humidity Sensor x1
  - Connectors

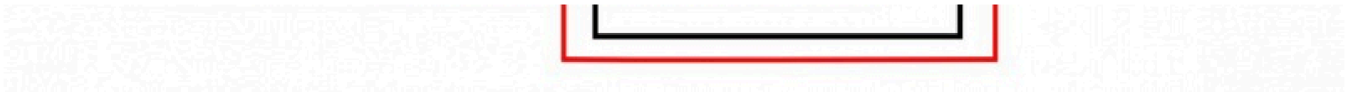
- **Software**
  - Arduino IDE (https://www.arduino.cc/en/Main/Software)
  - Download and install the **SHT3x Library and Example Programs** (https://github.com/DFRobot/DFRobot_SHT3x) (About how to install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))

- 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





Sample Code 1- Single Measurement Mode

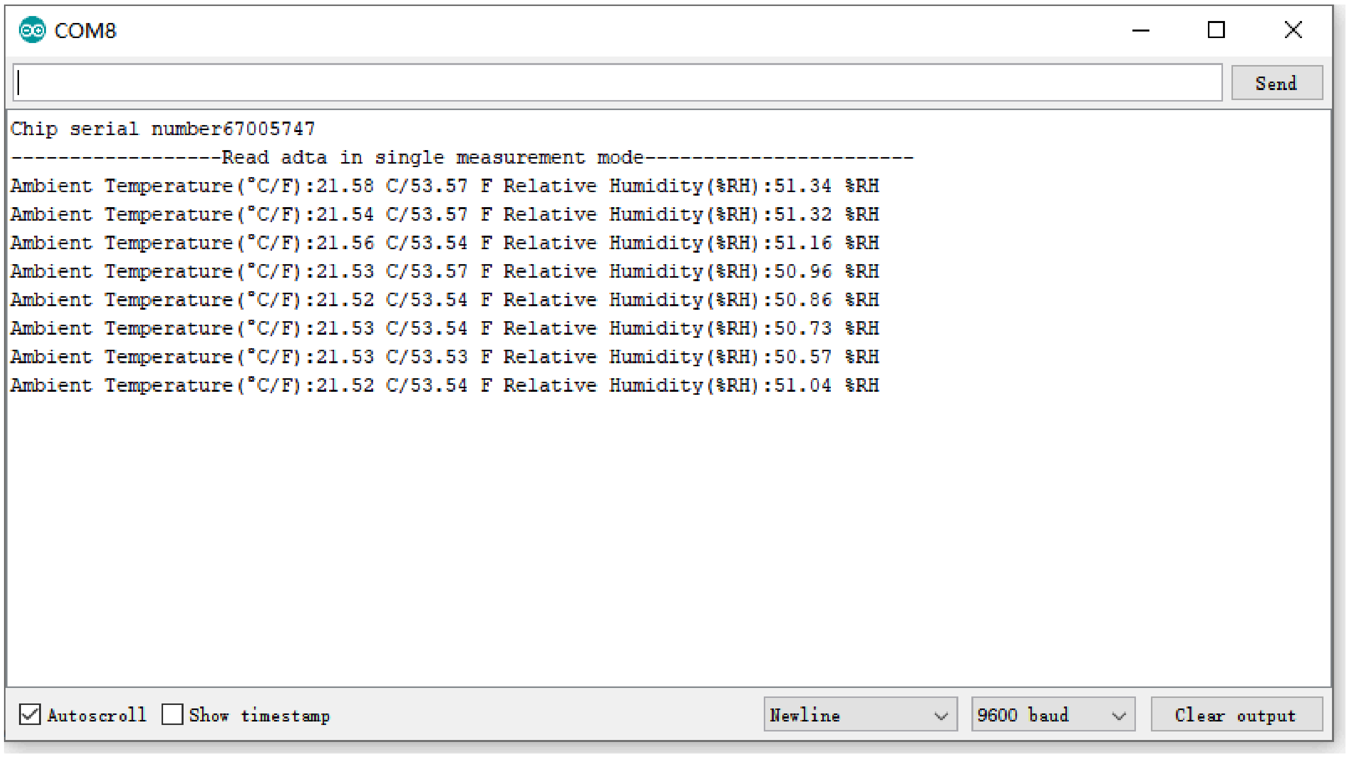
In single measurement mode, the sensor collects data every time the controller board sends out the data collecting command. The power consumption could be very low in this mode since users can read data according to their needs.

```
/*!
 * @file singleMeasurement.ino
 * @brief Read ambient temperature (C/F) and relative humidity (%RH) in si
 * @n Experimental phenomenon: the chip defaults in this mode, we need to
 * which means the repeatability of the read needs to be set (the differer
 * then read the temperature and humidity data and print the data in the s
 * @n Single measure mode: read data as needed, power consumption is relat
 * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (https://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 IIC bus pointer object and construction device, both can p
 * Wire in default.
 * @param address Chip IIC address, two optional addresses 0x44 and 0x45(0
 * @param RST RST Chip reset pin, 4 in default.
 * @n IIC address is determined by the pin addr on the chip.
 * @n When the ADR is connected to VDD, the chip IIC address is 0x45.
 * @n When the ADR is connected to GND, the chip IIC address is 0x44.
 */

//DFRobot_SHT3x sht3x(&Wire,/*address=*/0x45,/*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
    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 IIC, enter the chip's default mode
   * turn off the heater, and clear the alert of the ALERT pin.
   * @return Read the register status to determine whether the command was
   * 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 sens
   * @return Read the status of the register to determine whether the comm
   * and return true indicates success.
   * @note Heaters should be used in wet environments, and other cases of
   */

  //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 humidi
     * @note  Optional parameters:
                 eRepeatability_High /**In high repeatability mode, the humi
                 eRepeatability_Medium,/**In medium repeatability mode, the 
                 eRepeatability_Low, /**In low repeatability mode, the humic
     * @return Return a structure containing celsius temperature (°C), Fahre
     * @n Return O indicates right data return.
    DFRobot_SHT3x::sRHAndTemp_t data = sht3x.readTemperatureAndHumidity(sht3
    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);
  }
```

- Result

Print out the temperature and humidity information from the serial port.



```
Chip serial number67005747
------------------Read adta in single measurement mode-----------------------
Ambient Temperature(°C/F):21.58 C/53.57 F Relative Humidity(%RH):51.34 %RH
Ambient Temperature(°C/F):21.54 C/53.57 F Relative Humidity(%RH):51.32 %RH
Ambient Temperature(°C/F):21.56 C/53.54 F Relative Humidity(%RH):51.16 %RH
Ambient Temperature(°C/F):21.53 C/53.57 F Relative Humidity(%RH):50.96 %RH
Ambient Temperature(°C/F):21.52 C/53.54 F Relative Humidity(%RH):50.86 %RH
Ambient Temperature(°C/F):21.53 C/53.54 F Relative Humidity(%RH):50.73 %RH
Ambient Temperature(°C/F):21.53 C/53.53 F Relative Humidity(%RH):50.57 %RH
Ambient Temperature(°C/F):21.52 C/53.54 F Relative Humidity(%RH):51.04 %RH
```

Sample Code 2- Period Measurement Mode

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

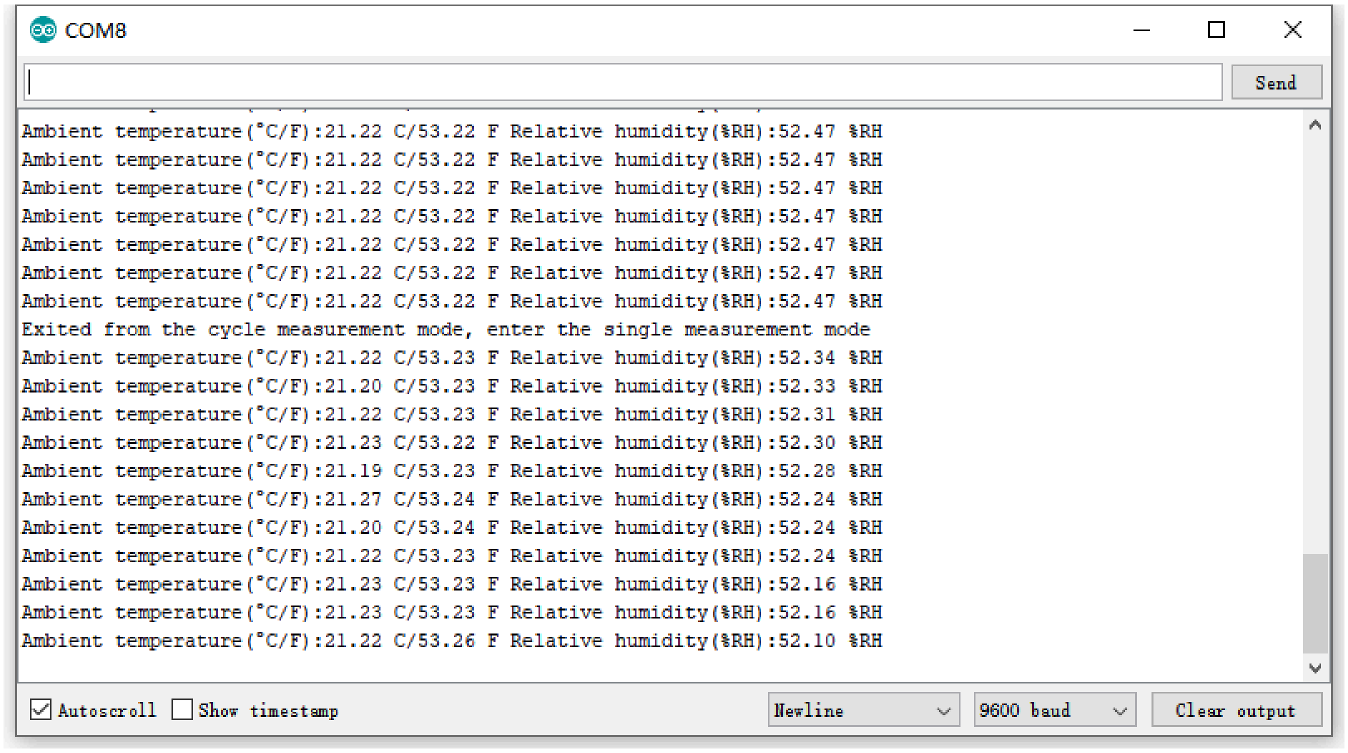
```
/*!
 * @file periodicDataReading.ino
 * @brief Read ambient temperature (C/F) and relative humidity (%RH) in cy
 * @n Experimental phenomenon: Before we start, please set the read freque
 * (the difference between the data measured by the chip under the same me
 * and enter the periodic read mode, and then read the temperature and hum
 * @n The temperature and humidity data will be printed at the serial port
 * @n It will exit the cycle mode and enter 2 measurement mode: Single mea
 * @n Single measurement mode: reflect the difference between the two mode
 * @n Cycle measurement mode: the chip periodically monitors temperature a
 * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (https://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 IIC bus pointer object and construction device, can both p
 * @param address Chip IIC address, two optional addresses 0x44 and 0x45(0
 * @param RST Chip reset pin, 4 in default.
 * @n The IIC address is determined by the pin addr on the chip.
 * @n When the ADR is connected to VDD, the chip IIC address is 0x45.
 * @n When the ADR is connected to GND, the chip IIC address is 0x44.
 */
//DFRobot_SHT3x sht3x(&Wire,/*address=*/0x45,/*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
  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 IIC, enter the chip's default mode
 * turn off the heater, and clear the alert of the ALERT pin.
 * @return Read the status register to determine whether the command was
 * 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
 * turn off the heater, and clear the alert of the ALERT pin.
 * @return The status register has a data bit that detects whether the c
 * and return true indicates success.
 * @note When using this API, the reset pin of the chip nRESET should be
 */
//if(!sht3x.pinReset()){
  //Serial.println("Failed to reset the chip");
//}

/**
 * heaterEnable() Turn on the heater inside the chip so that the sensor
 * @return Read the status register to determine whether the command was
 * @NOTE Heaters should be used in wet environment, and other cases of u
 */
//if(!sht3x.heaterEnable()){
  // Serial.println("Failed to turn on the heater");
//}
```

```
  /**
   * startPeriodicMode Enter cycle measurement mode and set repeatability
   * @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 0.25
               eMeasureFreq_10Hz   /**the chip collects data in every 0.1s
   * @param repeatability Read the repeatability of temperature and humidi
   * @note  Optional parameters:
               eRepeatability_High /**In high repeatability mode, the humi
               eRepeatability_Medium,/**In medium repeatability mode, the
               eRepeatability_Low, /**In low repeatability mode, the humic
   * @return Read the status of the register to determine whether the comm
   */
  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 t
  //The frequency to read data must be greater than the frequency to colle
  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 co
     * and return true indicates success.
     */
    sht3x.stopPeriodicMode();
    Serial.println("Exited from the cycle measurement mode, enter the sing
  }
  /**
   * readTemperatureAndHumidity Get temperature and humidity data in cycle
   * @return Return a structure containing celsius temperature (°C), Fahre
   * @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

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.



```
Ambient temperature(°C/F):21.22 C/53.22 F Relative humidity(%RH):52.47 %RH
Ambient temperature(°C/F):21.22 C/53.22 F Relative humidity(%RH):52.47 %RH
Ambient temperature(°C/F):21.22 C/53.22 F Relative humidity(%RH):52.47 %RH
Ambient temperature(°C/F):21.22 C/53.22 F Relative humidity(%RH):52.47 %RH
Ambient temperature(°C/F):21.22 C/53.22 F Relative humidity(%RH):52.47 %RH
Ambient temperature(°C/F):21.22 C/53.22 F Relative humidity(%RH):52.47 %RH
Ambient temperature(°C/F):21.22 C/53.22 F Relative humidity(%RH):52.47 %RH
Exited from the cycle measurement mode, enter the single measurement mode
Ambient temperature(°C/F):21.22 C/53.23 F Relative humidity(%RH):52.34 %RH
Ambient temperature(°C/F):21.20 C/53.23 F Relative humidity(%RH):52.33 %RH
Ambient temperature(°C/F):21.22 C/53.23 F Relative humidity(%RH):52.31 %RH
Ambient temperature(°C/F):21.23 C/53.22 F Relative humidity(%RH):52.30 %RH
Ambient temperature(°C/F):21.19 C/53.23 F Relative humidity(%RH):52.28 %RH
Ambient temperature(°C/F):21.27 C/53.24 F Relative humidity(%RH):52.24 %RH
Ambient temperature(°C/F):21.20 C/53.24 F Relative humidity(%RH):52.24 %RH
Ambient temperature(°C/F):21.22 C/53.23 F Relative humidity(%RH):52.24 %RH
Ambient temperature(°C/F):21.23 C/53.23 F Relative humidity(%RH):52.16 %RH
Ambient temperature(°C/F):21.23 C/53.23 F Relative humidity(%RH):52.16 %RH
Ambient temperature(°C/F):21.22 C/53.26 F Relative humidity(%RH):52.10 %RH
```

Sample Code 3- Temperature & Humidity Alarm

Set the temperature and humidity threshold. The pin INT generates alarm signal when exceeding the threshold.

When using this example, the pin INT of the sensor needs to be connected to the corresponding interrupt pin of main-board, refer to the pin list in the code below:

Note: the pin INT only works properly in period measurement mode.

```
/*!
 * @file alert.ino
 * @brief Temperature and humidity over-threshold alarm.
 * @n Experimental phenomenon: The user can customize the temperature and
 * and the ALERT pin generates an alarm signal once the values exceed the
 * @n NOTE: The ALERT pin on the sensor should be connected to the interru
 * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (https://www.dfrobot.com)
 * @licence  The MIT License (MIT)
 * @author [fengli](li.feng@dfrobot.com)
 * @version  V1.0
 * @date  2019-08-26
 * @get from https://www.dfrobot.com
 * @url https://github.com/DFRobot/DFRobot_SHT3x
 */

#include <DFRobot_SHT3x.h>
/*!
 * @brief Construct the function
 * @param pWire IIC bus pointer object and construction device, can both p
 * @param address Chip IIC address, two optional addresses 0x44 and 0x45(0
 * @param RST Chip reset pin, 4 in default.
 * @n IIC address is determined by the pin addr on the chip.
 * @n When the ADR is connected to VDD, the chip IIC address is 0x45.
 * @n When the ADR is connected to GND, the chip IIC address is 0x44.
 */
//DFRobot_SHT3x sht3x(&Wire,/*address=*/0x45,/*RST=*/4);

DFRobot_SHT3x sht3x;
//The non-alarm status of the alert pin is low;
volatile  int alertState = 0;
void alert(){
  alertState = 1 - alertState;
}
void setup() {
  Serial.begin(9600);
  #ifdef ARDUINO_ARCH_MPYTHON
```

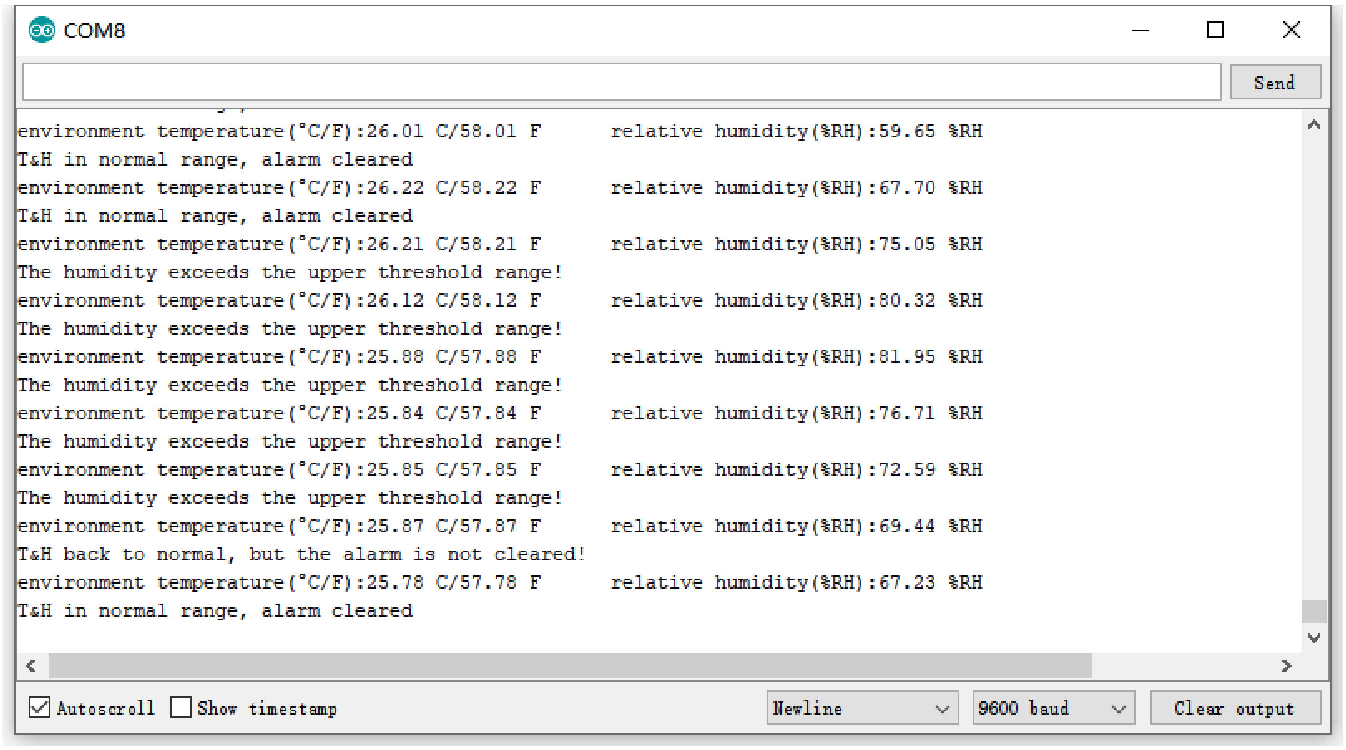
```
/*                    The Correspondence Table of ESP32 Interrupt Pins A
 * -----------------------------------------------------------------------
 * |            |  DigitalPin  | P0-P20 can be used as an external inter
 * |    esp32   |------------------------------------------------------
 * |            | Interrupt No |  DigitalPinToInterrupt (Pn) can be usec
 * |----------------------------------------------------------------------
 */
attachInterrupt(digitalPinToInterrupt(P16)/*Query the interrupt number c
//Open esp32's P16 pin for external interrupt, bilateral edge trigger, A
#else
/*    The Correspondence Table of AVR Series Arduino Interrupt Pins And
 * -----------------------------------------------------------------------
 * |                                        |  DigitalPin  | 2  | 3  |
 * |    Uno, Nano, Mini, other 328-based    |--------------------------
 * |                                        | Interrupt No | 0  | 1  |
 * |----------------------------------------------------------------------
 * |                                        |    Pin       | 2  | 3  | 2
 * |                Mega2560                |--------------------------
 * |                                        | Interrupt No | 0  | 1  | 2
 * |----------------------------------------------------------------------
 * |                                        |    Pin       | 3  | 2  | 0
 * |    Leonardo, other 32u4-based          |--------------------------
 * |                                        | Interrupt No | 0  | 1  | 2
 * |----------------------------------------------------------------------
 */
/*                      The Correspondence Table of micro:bit Interrupt
 * -----------------------------------------------------------------------
 * |              micro:bit                      | DigitalPin |P0-P20 ca
 * |  (When using as an external interrupt,      |----------------------
 * |no need to set it to input mode with pinMode)|Interrupt No|Interrupt
 * |----------------------------------------------------------------------
 */
attachInterrupt(/*Interrupt No*/0,alert,CHANGE);//Open the external inte
   //UNO(2), Mega2560(2), Leonardo(3), microbit(P0).
#endif
  //Initialize the chip to detect if it can communicate properly
while (sht3x.begin() != 0) {
  Serial.println("The initialization of the chip is failed, please confi
  delay(1000);
}
/**
 * readSerialNumber Read the serial number of the chip
```

```
   * @return Return 32-digit serial number
   */
  Serial.print("The chip serial number");
  Serial.println(sht3x.readSerialNumber());
  /**
   * softReset Send command resets via iiC, enter the chip's default mode
   * and clear the alert of the ALERT pin.
   * @return Read the status register to determine whether the command was
   */
  if(!sht3x.softReset()){
     Serial.println("Failed to reset the chip");
   }
  /**
   * @brief All flags (Bit 15, 11, 10, 4) in the status register can be cl
   * @n ALERT can work properly only when the bit:15 is set to 0, otherwis
   */
  sht3x.clearStatusRegister();
  /**
   * startPeriodicMode Enter cycle measurement mode and set repeatability,
   * @param measureFreq Read the data frequency, data type eMeasureFrequer
   * @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 0.25
               eMeasureFreq_10Hz   /**the chip collects data in every 0.1s
   * @param repeatability Read the repeatability of temperature and humidi
   * @note  Optional parameters:
               eRepeatability_High /**In high repeatability mode, the humi
               eRepeatability_Medium,/**In medium repeatability mode, the
               eRepeatability_Low, /**In low repeatability mode, the humid
   * @return Read the status of the register to determine whether the comm
   */
  if(!sht3x.startPeriodicMode(sht3x.eMeasureFreq_10Hz)){
    Serial.println("Failed to enter the periodic mode");
  }
  /**
   * setTemperatureLimitC Set the threshold temperature and alarm clear te
   * setTemperatureLimitF Set the threshold temperature and alarm clear te
   * @param highset High temperature alarm point, when the temperature is
   * @param highClear High temperature alarm clear point, alarming when th
   * @param lowset Low temperature alarm point, when the temperature is lo
```

```
     * @param lowclear Low temperature alarm clear point, alarming when the
     * @note The filled value should be an integer (range: -40 to 125 degree
     */
    //sht3x.setTemperatureLimitF(/*highset=*/35,/*highClear=*/34,/*lowSet=*/
    if(sht3x.setTemperatureLimitC(/*highset=*/35,/*highClear=*/34,/*lowSet=*
      Serial.println("Failed to set the temperature limit");
    }
    /**
     * setHumidityLimitRH Set the relative humidity threshold temperature ar
     * @param highset High humidity alarm point, when the humidity is greate
     * @param highClear High humidity alarm clear point, alarming when the h
     * @param lowset Low humidity alarm point, when the humidity is lower th
     * @param lowclear Low humidity alarm clear point, alarming when the hum
     * @note The filled value should be an integer (range: 0 - 100 %RH,highs
     */
    if(sht3x.setHumidityLimitRH(/*highset=*/70,/*highClear=*/68,/*lowSet=*/1
      Serial.println("Failed to set the humidity limit");
    }
    //Serial.println(F("string") Save stings to flash to save the dynamic ra
    Serial.println(F("----------------------Alarm Detection----------------
    Serial.println(F("Alarms raised when temp and humidity are out of the th
    Serial.println(F("-Different main contorl UNO(2), Mega2560(2), Leonardo(
    Serial.println(F("----------------------the humidity limit(%RH)---------
    /**
     * @brief Measure relative humidity threshold temperature and alarm clea
     * @return Return true indicates successful data acquisition
     */
    if(sht3x.measureHumidityLimitRH()){
      Serial.print("high set:");
      //getHumidityHighSetRH() Get the high humidity alarm point
      Serial.print(sht3x.getHumidityHighSetRH());
      Serial.print("               low clear:");
      //getHumidityHighClearRH() Get the high humidity alarm clear point
      Serial.println(sht3x.getHumidityLowClearRH());
      Serial.print("high clear:");
      //getHumidityLowClearRH() Get the low humidity alarm clear point
      Serial.print(sht3x.getHumidityHighClearRH());
      Serial.print("               low set:");
      //getHumidityLowSetRH() Get the low humidity alarm point
      Serial.println(sht3x.getHumidityLowSetRH());
    } else {
      Serial.println("Failed to get the humidity limit");
```

```
  }
  /**
   * measureTemperatureLimitC Measure the threshold temperature and alarm
   * measureTemperatureLimitF Measure the threshold temperature and alarm
   * @return Return true indicates successful data acquisition
   */
  Serial.println("----------------------temperature limit(°C)-------------
  //Serial.println(F("----------------------temperature limit(°F)---------
  if(sht3x.measureTemperatureLimitC()){
    Serial.print("high set:");
    //getTemperatureHighSetC() Get high temperature alarm points(°C)
    //getTemperatureHighSetF() Get high temperature alarm points(°F)
    Serial.print(sht3x.getTemperatureHighSetC());
    Serial.print("               low clear:");
    //getTemperatureHighClearC() Get high temperature alarm clear points(°
    //getTemperatureHighClearF() Get high temperature alarm clear points(°
    Serial.println(sht3x.getTemperatureLowClearC());
    Serial.print("high clear:");
    //getTemperatureLowClearC() Get low temperature alarm clear points(°C)
    //getTemperatureLowClearF() Get low temperature alarm clear points(°F)
    Serial.print(sht3x.getTemperatureHighClearC());
    Serial.print("               low set:");
    //getTemperatureLowSetC() Get low temperature alarm points(°C)
    //getTemperatureLowSetF() Get low temperature alarm points(°F)
    Serial.println(sht3x.getTemperatureLowSetC());
    Serial.println("--------------------------------------------------------
  } else {
    Serial.println("Failed to get temperature limit");
  }
  /**
   * readAlertState Read the status of the ALERT pin.
   * @return High returns 1, low returns 0.
   */
  //To initialize the state of ALERT
  if(sht3x.readAlertState() == 1){
    alertState = 1;
  } else {
    alertState = 0;
  }
}
void loop() {
  Serial.print("environment temperature(°C/F):");
  /**
   * getTemperatureC Get the measured temperature (in degrees Celsius)
   * @return Return temperature data of the type float
   */
  Serial.print(sht3x.getTemperatureC());
  Serial.print(" C/");
  /**
   * getTemperatureF Get the measured temperature (in degrees Celsius)
   * @return Return temperature data of the type float
   */
  Serial.print(sht3x.getTemperatureF());
  Serial.print(" F      ");
  Serial.print("relative humidity(%RH):");
  /**
   * getHumidityRH Get measured humidity (in %RH)
   * @return Return humidity data of the type float
   */
  Serial.print(sht3x.getHumidityRH());
  Serial.println(" %RH");
  //The read data frequency should greater than the frequency to collect c
  if(alertState == 1){
    /**
     * @brief Determine if the temperature and humidity are out of the thr
     * @return Return the status code, representing as follows
     * @n 01 Indicates that the humidity exceeds the lower threshold range
     * @n 10 Indicates that the temperature exceeds the lower threshold ra
     * @n 11 Indicates that both the humidity and the temperature exceed t
     * @n 02 Indicates that the humidity exceeds the upper threshold range
     * @n 20 Indicates that the temperature exceeds the upper threshold ra
     * @n 22 Indicates that both the humidity and the temperature exceed t
     * @n 12 Indicates that the temperature exceeds the lower threshold ra
     //and the humidity exceeds the upper threshold range
     * @n 21 Indicates that the temperature exceeds the upper threshold ra
     //and the humidity exceeds the lower threshold range
     * @n 0  Back to normal, but the alarm is not cleared.
     */
    uint8_t state = sht3x.environmentState();
    //Serial.println(F("string") Save stings to flash to save the dynamic
    if(state == 1)  Serial.println(F("The humidity exceeds the lower thres
    else if(state == 10)  Serial.println(F("The temperature exceeds the lc
    else if(state == 11)  Serial.println(F("The humidity and the temperatu
    else if(state == 2)   Serial.println(F("The humidity exceeds the upper
    else if(state == 20)  Serial.println(F("The temperature exceeds the up
    else if(state == 22)  Serial.println(F("The humidity and the temperatu
    else if(state == 12)  Serial.println(F("The temperature exceeds the lo
    else if(state == 21)  Serial.println(F("The temperature exceeds the up
    else Serial.println(F("T&H back to normal, but the alarm is not cleare
  } else {
    Serial.println(F("T&H in normal range, alarm cleared"));
  }
  delay(1000);
}
```

- Result



```
environment temperature(°C/F):26.01 C/58.01 F      relative humidity(%RH):59.65 %RH
T&H in normal range, alarm cleared
environment temperature(°C/F):26.22 C/58.22 F      relative humidity(%RH):67.70 %RH
T&H in normal range, alarm cleared
environment temperature(°C/F):26.21 C/58.21 F      relative humidity(%RH):75.05 %RH
The humidity exceeds the upper threshold range!
environment temperature(°C/F):26.12 C/58.12 F      relative humidity(%RH):80.32 %RH
The humidity exceeds the upper threshold range!
environment temperature(°C/F):25.88 C/57.88 F      relative humidity(%RH):81.95 %RH
The humidity exceeds the upper threshold range!
environment temperature(°C/F):25.84 C/57.84 F      relative humidity(%RH):76.71 %RH
The humidity exceeds the upper threshold range!
environment temperature(°C/F):25.85 C/57.85 F      relative humidity(%RH):72.59 %RH
The humidity exceeds the upper threshold range!
environment temperature(°C/F):25.87 C/57.87 F      relative humidity(%RH):69.44 %RH
T&H back to normal, but the alarm is not cleared!
environment temperature(°C/F):25.78 C/57.78 F      relative humidity(%RH):67.23 %RH
T&H in normal range, alarm cleared
```

FAQ

For any questions, advice or cool ideas to share, please visit the **DFRobot Forum** (https://www.dfrobot.com/forum)

More Documents

- Schematics

(https://dfimg.dfrobot.com/nobody/wiki/354724e58ab5cc70ce3b8c1c50de7e25.pdf)

- Layout with Dimension (https://dfimg.dfrobot.com/nobody/wiki/c3d3b0d46236be21769520b2d9ddeae9.pdf)
- SHT3x Datasheet (https://dfimg.dfrobot.com/nobody/wiki/88b31350da4f54d00989c74c6fa392f7.pdf)
- SHT3X Handling Instructions (https://dfimg.dfrobot.com/nobody/wiki/ed23ca262d11f01046f949078f45a37c.pdf)