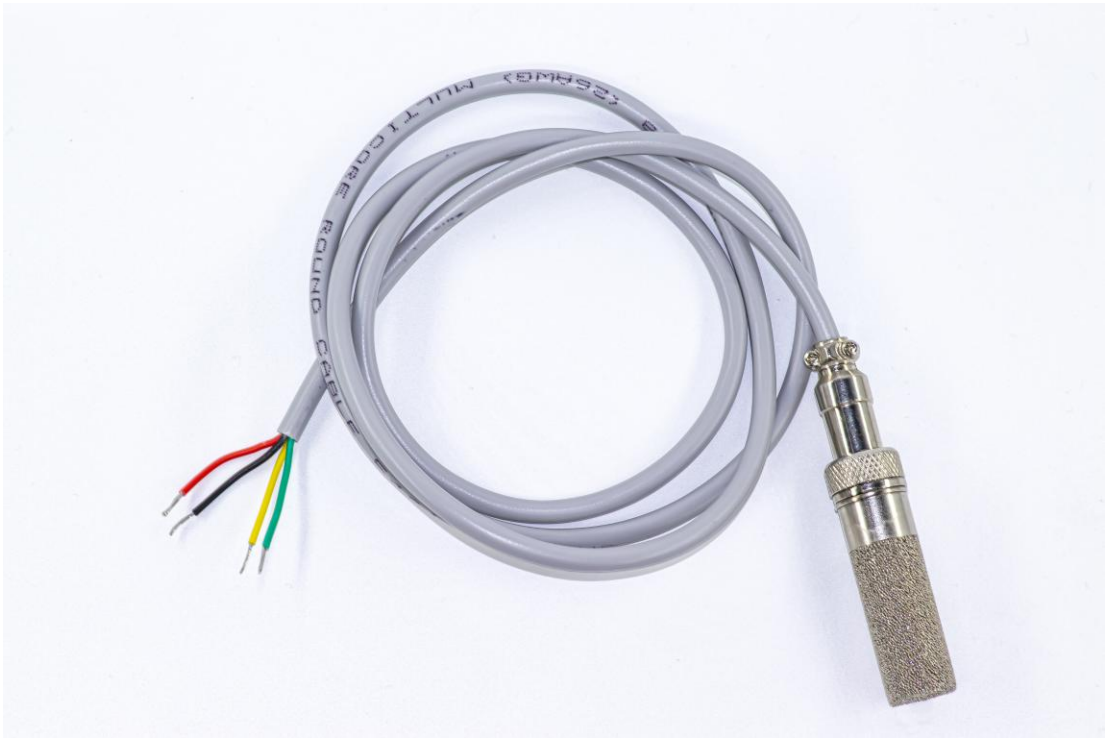
IST

IST

Industrial UART Humidity, Pressure, Temperature and VOC Sensor Probe

Version 1.0



IST +91 8655821342 info@7semi.com www.7semi.com

Table of Contents

1.Introduction

This all-in-one sensor is an excellent choice for developers needing accurate and low-power environmental measurements. Its compact design and versatile sensing capabilities make it suitable for everything from consumer wearables to industrial monitoring systems.

1.1 Features

- Digital humidity and temperature sensor probe with multicore cable.
- Compact and robust design.
- Suitable for indoor and outdoor applications.
- Seamless compatibility with common development platforms.
- Ideal for HVAC, agriculture, weather stations, and smart home systems.
- Long-term stability for reliable performance over time

2.Technical specification

This compact board is designed to provide real-time collection and processing of various environmental parameters, all in an ultra-low-power form factor. Its integrated design, featuring both data acquisition and on-board computation, makes it well-suited for IoT applications, and home automation projects. Flexible interfaces help ensure easy integration into existing systems, while the board's accuracy and reliability support demanding use cases.

The board outputs preformatted UART data frames containing decoded temperature, humidity, pressure values simplifying integration with external systems. Its energy-efficient design and compact form factor make it ideal for both portable and embedded applications.

Applications:-

- **Home Automation & control**
- **IoT Integration**
- **Weather Forecasting**
- **Educational Tools**
- **GPS Enhancement & Navigation**
- **Indoor & outdoor navigation**

This sensor board is the ideal solution for applications that demand high accuracy, low power consumption, and easy integration into existing systems.

2.1 General specifications

Operating Ranges:

- Pressure: 300 hPa to 1100 hPa
- Temperature: -40°C to +85°C *(Full accuracy:- 0°C to 65°C)*
- Humidity: 0% RH to 100% RH

Accuracy:

- Pressure: ±1.0 hPa (at 0 – 65°C)
- Temperature: ±0.5°C (at 0 – 65°C)
- Humidity: ±3% RH (20–80% RH range)

Operating Voltage:

- VDD (supply voltage): 1.71 V to 3.6 V
- VDDIO (interface voltage): 1.2 V to 3.6 V

Communication protocol: UART

Baud rate for UART: 9600

2.2 Sensor protocol

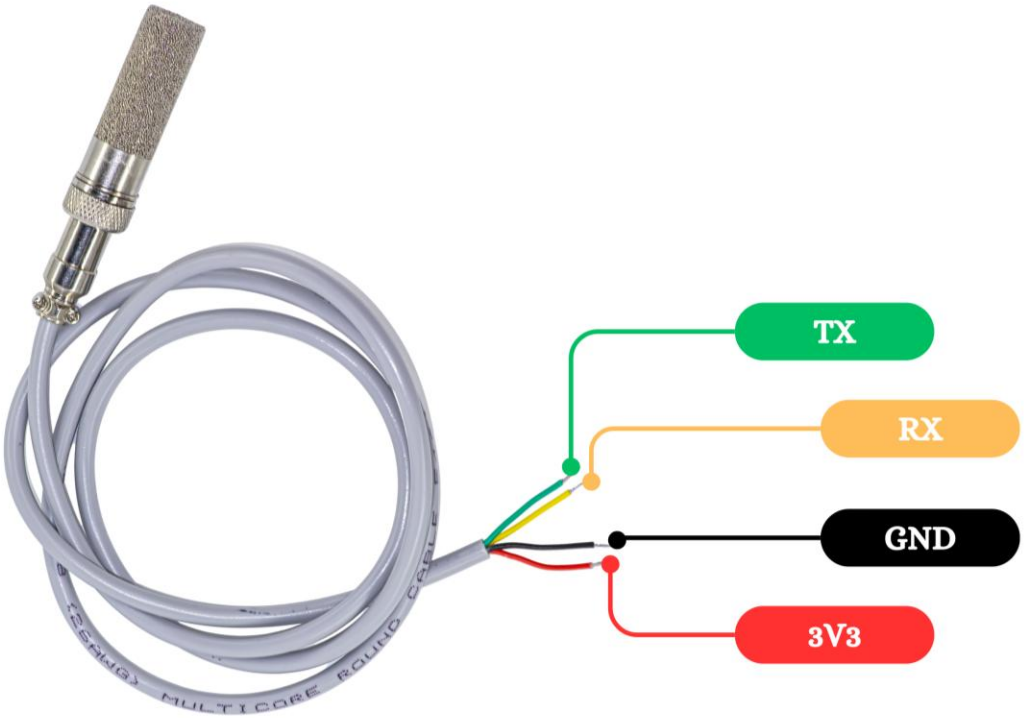
The sensor outputs the data frame using UART communication.

AA 14 AE D9 41 E2 C9 C4 47 00 77 5A 42 55

Field	Bytes	Description
Start Byte	AA	Packet start marker
Temperature	14 AE D9 41	4 bytes, Little Endian
Pressure	E2 C9 C4 47	4 bytes, Little Endian
Humidity	00 77 5A 42	4 bytes, Little Endian
End Byte	55	Packet end marker

3. Hardware setup

3.1 Pinouts



RX: UART receives pin, connect to the TX of the external device.

TX: UART transmits pin, connect to the RX of the external device.

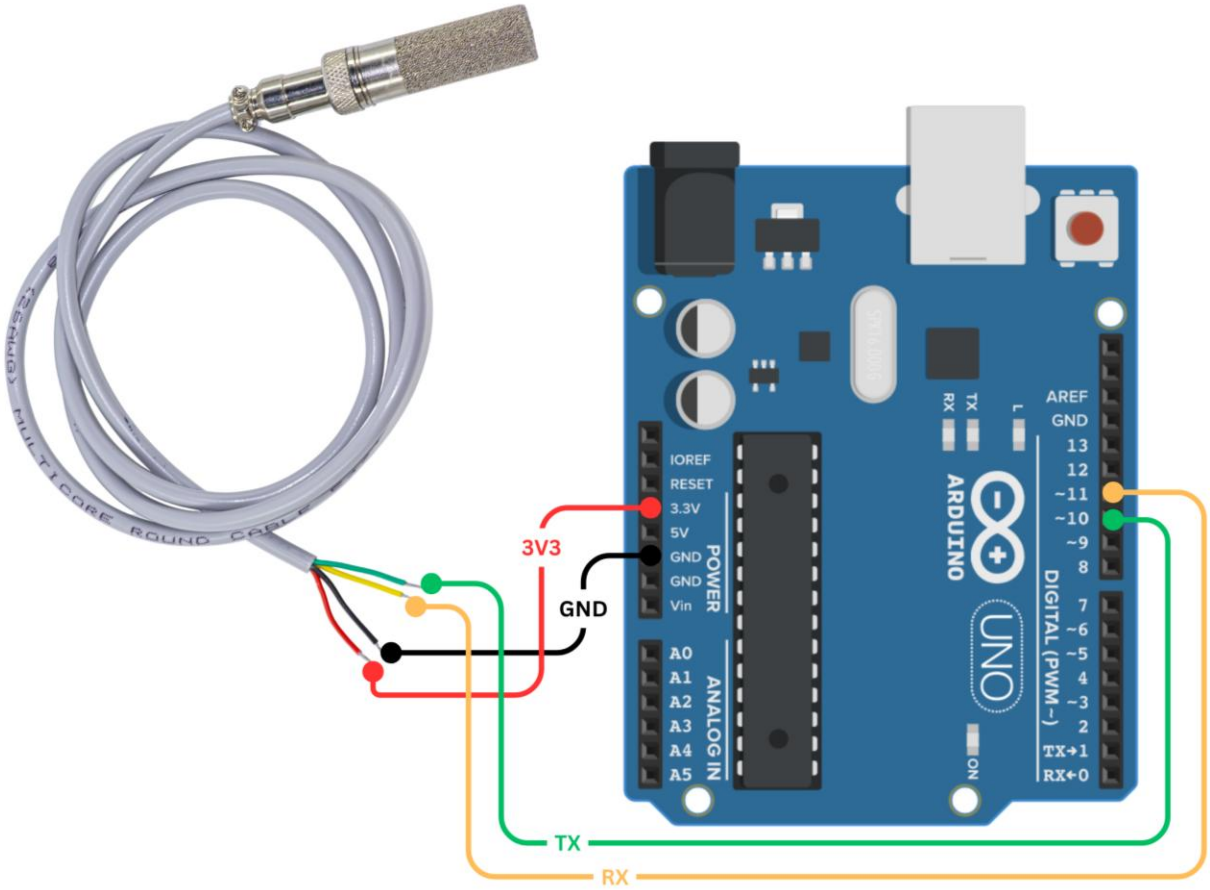
GND: Ground pin, connect to the system's ground.

3V3: Power input, connect to a 3.3V DC supply.

Connection Guidelines

- **Power Supply**: Ensure a clean and stable 3.3V DC power source is connected to the 3V3 pin, with GND tied to the power supply's ground.
- **UART Communication**: Connect the **RX** pin of the sensor board to the **TX** pin of the external device and the **TX** pin of the board to the **RX** pin of the external device.
- Properly match baud rates and settings (e.g., stop bits, parity) for UART communication.

3.2 Hardware connection



4.Sample code and explanation

4.1 Sample code

```
#include <SoftwareSerial.h>

// Constants for frame validation
#define FRAME_START_BYTE 0xAA
#define FRAME_END_BYTE 0x55
#define FRAME_SIZE 14

// Define SoftwareSerial RX and TX pins
#define RX_PIN 10
#define TX_PIN 11

// Create a SoftwareSerial object
SoftwareSerial softSerial(RX_PIN, TX_PIN);

// Variables to hold received data
float receivedTemperature = 0.0;
float receivedPressure = 0.0;
float receivedHumidity = 0.0;

void setup() {
  // Initialize SoftwareSerial
  softSerial.begin(9600);

  // Initialize Serial Monitor
```

```
  Serial.begin(9600);
  Serial.println("Starting BME280 Frame Receiver with SoftwareSerial...");
}

void loop() {
  // Check if enough data is available in the software serial buffer
  if (softSerial.available() >= FRAME_SIZE) {
    uint8_t frame[FRAME_SIZE];
    for (int i = 0; i < FRAME_SIZE; i++) {
      frame[i] = softSerial.read(); // Read frame data byte by byte
    }

    // Debug: Print the raw frame
    //Serial.println("Received Raw Frame:");
    for (int i = 0; i < FRAME_SIZE; i++) {
      // Serial.print("0x");
      // Serial.print(frame[i], HEX);
      // Serial.print(" ");
    }
    Serial.println();

    // Validate the start and end bytes
    if (frame[0] == FRAME_START_BYTE && frame[13] == FRAME_END_BYTE) {
      // Extract data using memcpy
      memcpy(&receivedTemperature, &frame[1], 4);
      memcpy(&receivedPressure, &frame[5], 4);
      memcpy(&receivedHumidity, &frame[9], 4);
```

```
      // Debug: Print the received data
      Serial.print("Temperature: ");
      Serial.print(receivedTemperature);
      Serial.println(" °C");

      Serial.print("Pressure: ");
      Serial.print(receivedPressure);
      Serial.println(" hPa");

      Serial.print("Humidity: ");
      Serial.print(receivedHumidity);
      Serial.println(" %RH");
      Serial.println(" ");
      Serial.println(" ");
    } else {
      // Invalid frame received
      Serial.println("Error: Invalid frame received.");
    }
  } else {
    // No data available
    Serial.println("Waiting for data...");
    delay(1000); // Wait for 1 second before checking again
  }
}
```

4.2 Code explanation

```
#include <SoftwareSerial.h>
```

Includes the SoftwareSerial library, which allows using different pins for serial communication (instead of just pins 0 and 1).

```
#define FRAME_START_BYTE 0xAA
#define FRAME_END_BYTE 0x55
#define FRAME_SIZE 14
```

Defines the format of a valid data frame:

- Starts with 0xAA
- Ends with 0x55
- Total size: 14 bytes

```
#define RX_PIN 10
#define TX_PIN 11
```

SoftwareSerial softSerial(RX_PIN, TX_PIN);

Sets up software serial communication on pins 10 (RX) and 11 (TX).

```
void setup() {
  softSerial.begin(9600);
  Serial.begin(9600);
}
```

Initializes both the SoftwareSerial and the Serial Monitor at 9600 baud.

```
if (softSerial.available() >= FRAME_SIZE) {
```

```
Checks if a complete frame (14 bytes) is available.
```

```
for (int i = 0; i < FRAME_SIZE; i++) {
  frame[i] = softSerial.read();
}
```

Reads the 14 bytes into an array called frame.

```
if (frame[0] == FRAME_START_BYTE && frame[13] == FRAME_END_BYTE) {
```

Checks if the frame starts with 0xAA and ends with 0x55.

```
memcpy(&receivedTemperature, &frame[1], 4);
memcpy(&receivedPressure, &frame[5], 4);
memcpy(&receivedHumidity, &frame[9], 4);
```

Copies bytes from the frame into float variables:

Bytes 1–4: Temperature

Bytes 5–8: Pressure

Bytes 9–12: Humidity

```
Serial.print("Temperature: ");
Serial.print(receivedTemperature);
```

Prints the extracted temperature, pressure, and humidity values to the Serial Monitor.

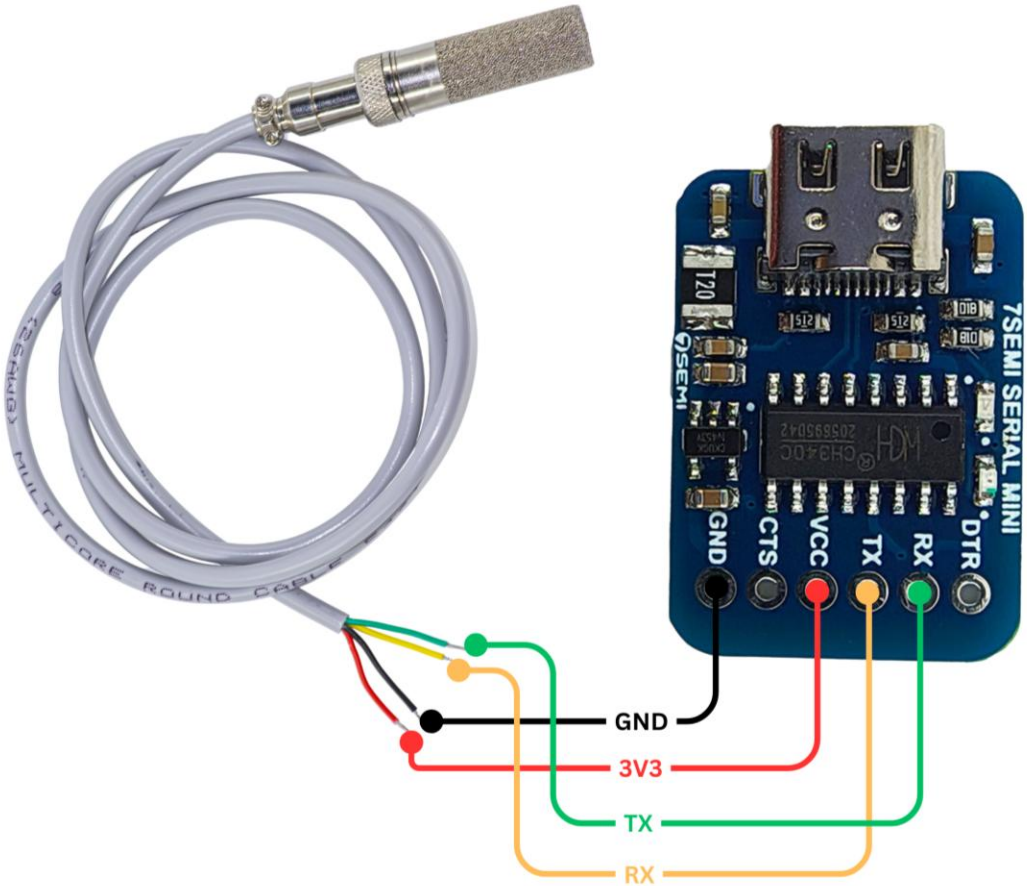
4.3 Serial output

```
Waiting for data...

Temperature: 27.61 °C
Pressure: 100756.52 hPa
Humidity: 53.39 %RH



Temperature: 27.61 °C
Pressure: 100756.57 hPa
Humidity: 53.42 %RH


Waiting for data...

Temperature: 27.61 °C
Pressure: 100756.80 hPa
Humidity: 53.42 %RH



Temperature: 27.61 °C
Pressure: 100756.97 hPa
Humidity: 53.40 %RH


Waiting for data...

Temperature: 27.61 °C
Pressure: 100756.19 hPa
Humidity: 53.38 %RH
```

4.4 Sample code link

Sample output Image of USB to TTL :

```
AA 14 AE D9 41 E2 C9 C4 47 00 77 5A 42 55
```



Sample code link:- [ES-50104-U280_RX]

5.Mechanical specification