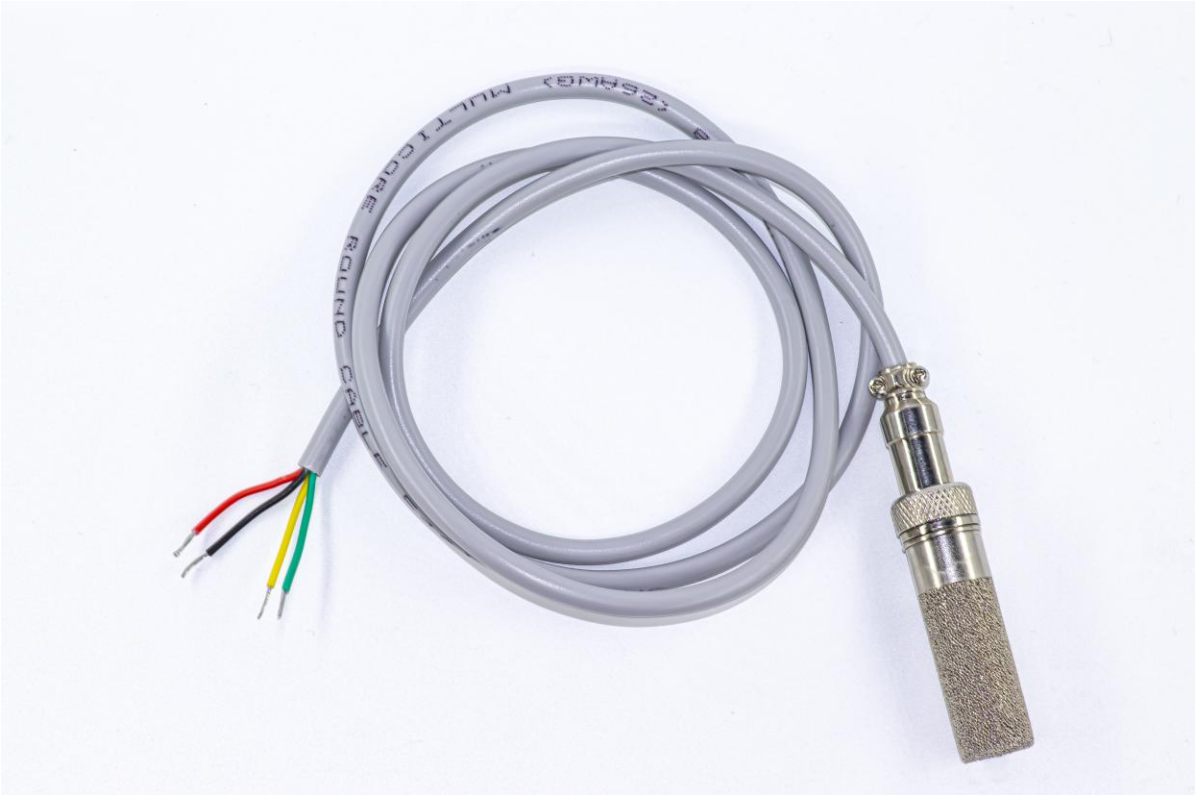
IST

IST

Industrial UART Humidity, Pressure, Temperature and VOC Sensor Probe

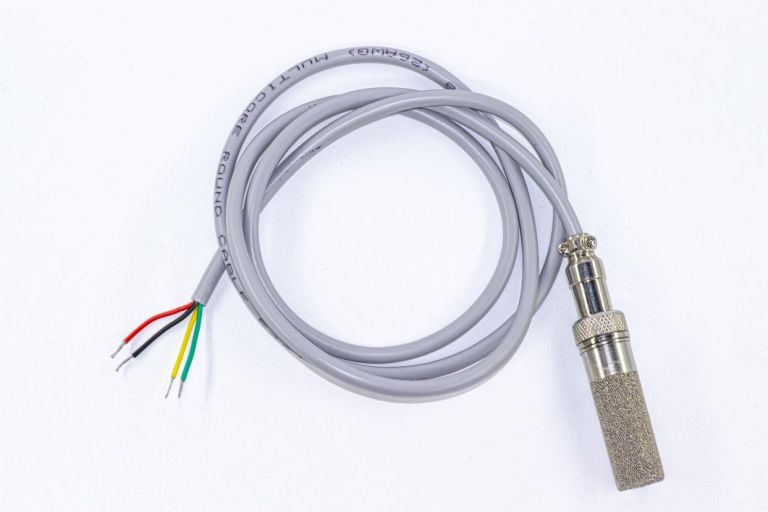
Version 1.0



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—from indoor air quality monitoring to advanced analytics and control.

The board outputs preformatted UART data frames containing decoded temperature, humidity, pressure, and IAQ 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**
- **Air Quality monitors**
- **Internet of things**
- **Weather Forecasting**
- **Educational Tools**
- **GPS Enhancement & Navigation**
- **Fitness monitoring/ well being**

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
- Gas sensor: -40°C to +85°C, 10% RH to 95% RH

Absolute accuracy:

- Pressure: ±0.6 hPa *(at 0 – 65°C)*
- Temperature: ±1.0°C *(at 0 – 65°C)*
- Humidity: ±3% RH *(20 – 80% RH range, 25°C)*

Operating Voltage:

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

IAQ range: 0 - 500

Communication protocol: UART

Baud rate for UART: 9600

Index for Air Quality (IAQ) classification

IAQ Index	Air Quality	Impact (long-term exposure)	Suggested action
0 – 50	Excellent	Pure air; best for well-being	No measures needed
51 – 100	Good	No irritation or impact on well-being	No measures needed
101 – 150	Lightly polluted	Reduction of well-being possible	Ventilation suggested
151 – 200	Moderately polluted	More significant irritation possible	Increase ventilation with clean air
201 – 250	Heavily polluted	Exposition might lead to effects like headache depending on type of VOCs	optimize ventilation
251 – 350	Severely polluted	More severe health issue possible if harmful VOC present	Contamination should be identified if level is reached even w/o presence of people; maximize ventilation & reduce attendance
> 351	Extremely polluted	Headaches, additional neurotoxic effects possible	Contamination needs to be identified; avoid presence in room and maximize ventilation

2.2 Sensor protocol

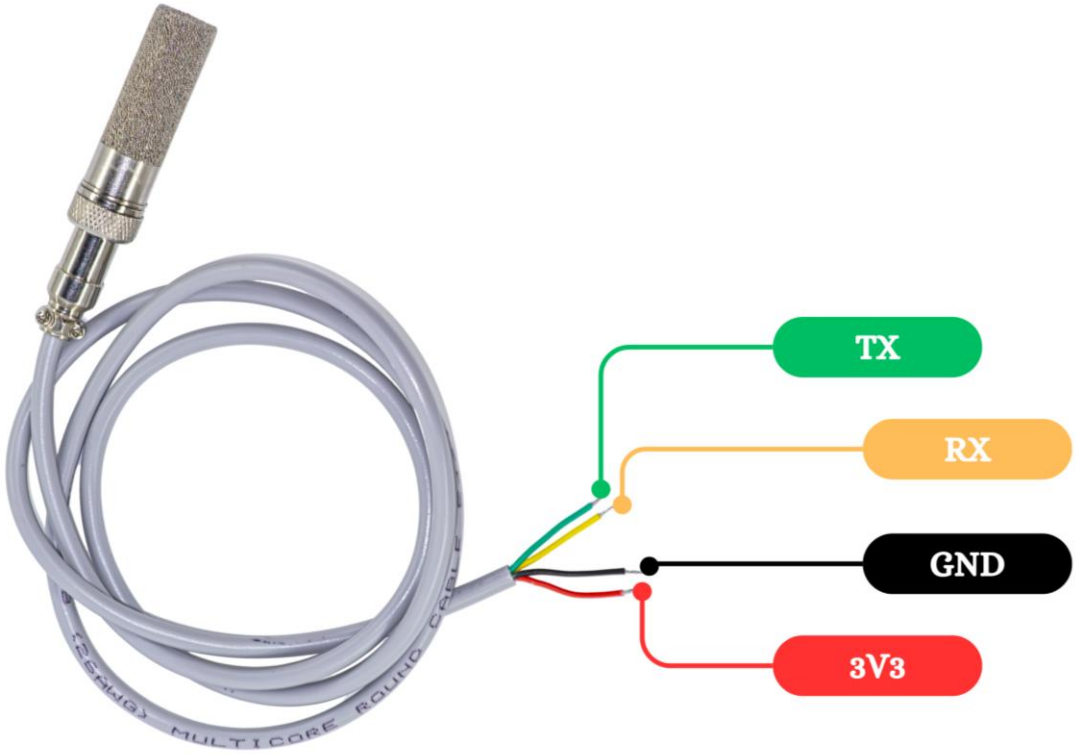
The sensor outputs the data frame using UART communication.

7E 10 BE FE 05 42 1F 34 C5 47 76 30 29 42 00 00 A0 40 7F

Field	Bytes	Description
Start Byte	7E	Packet start marker
Payload length	10	Payload size in bytes (Hex 10 = Decimal 16)
Temperature	BE FE 05 42	4 bytes, Little Endian
Pressure	1F 34 C5 47	4 bytes, Little Endian
Humidity	76 30 29 42	4 bytes, Little Endian
IAQ Score	00 00 A0 40	4 bytes, Little Endian
End Byte	7F	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.

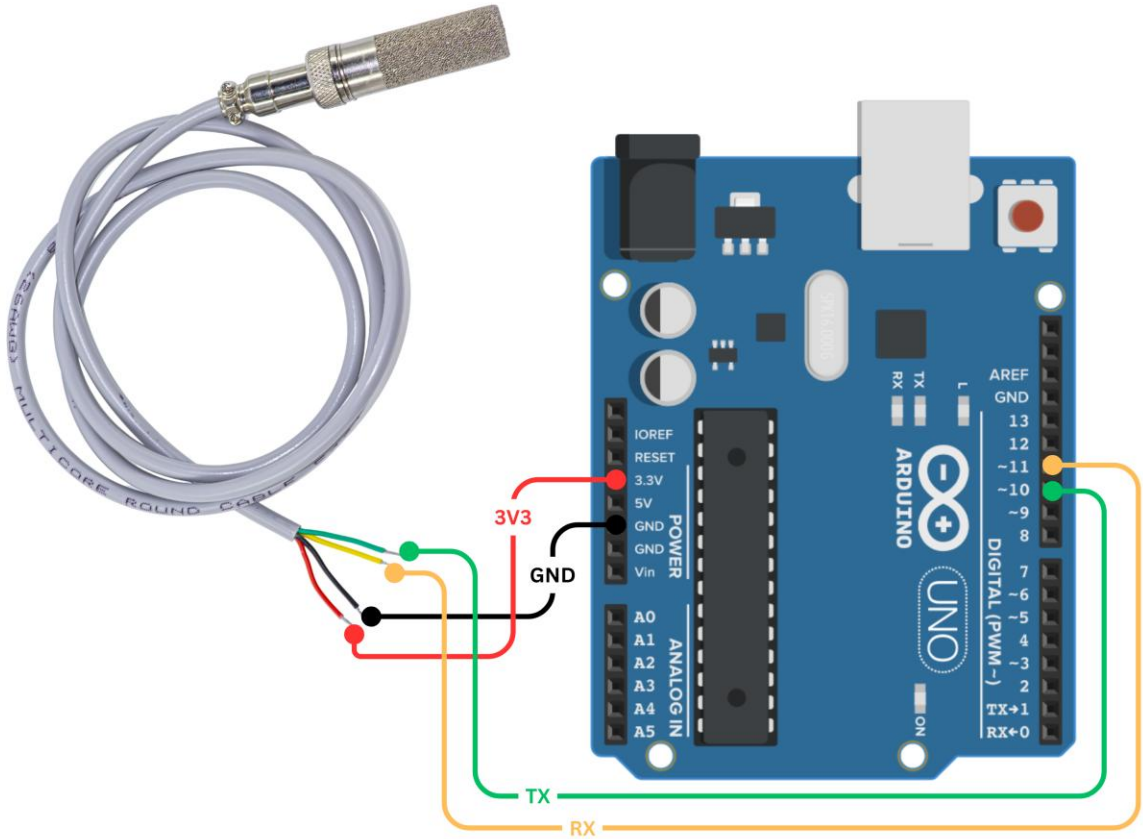
GND: Ground pin, connect to the system's ground.

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>

// Define SoftwareSerial pins

SoftwareSerial mySerial(10, 11); // RX on pin 10, TX on pin 11

// Frame constants

const uint8_t START_BYTE = 0x7E;  // Start byte

const uint8_t END_BYTE = 0x7F;    // End byte

const uint8_t PAYLOAD_LENGTH = 16; // Payload length (4 floats: temperature, pressure, humidity, IAQ)

const uint8_t FRAME_LENGTH = 2 + PAYLOAD_LENGTH + 1; // Start byte + length + payload + end byte

// Buffer to store the frame

uint8_t frameBuffer[FRAME_LENGTH];

uint8_t receivedIndex = 0;

bool frameInProgress = false;


void setup() {

  Serial.begin(9600);   // Debugging serial port

  mySerial.begin(9600); // SoftwareSerial for STM32 communication

  Serial.println("Waiting for data...");

}

void loop() {

  while (mySerial.available()) {

    uint8_t byteReceived = mySerial.read();

  // Debugging: Print received bytes

  // Serial.print("Received Byte: 0x");
```

```
    //Serial.println(byteReceived, HEX);

    // Handle start of frame
    if (!frameInProgress && byteReceived == START_BYTE) {
      frameInProgress = true;
      receivedIndex = 0;
      frameBuffer[receivedIndex++] = byteReceived; // Store start byte
    } else if (frameInProgress) {
      // Store the received byte in the buffer
      frameBuffer[receivedIndex++] = byteReceived;

      // Check if the frame is complete
      if (receivedIndex == FRAME_LENGTH) {
        // Verify the end byte
        if (frameBuffer[FRAME_LENGTH - 1] == END_BYTE) {
          processFrame(frameBuffer); // Process the received frame
        } else {
          Serial.println("Invalid frame: missing END_BYTE");
        }
        frameInProgress = false; // Reset for the next frame
        receivedIndex = 0;
      }
    }
  }
}
// Function to process a complete frame
void processFrame(uint8_t *frame) {
 // Validate payload length
```

```
 if (frame[1] != PAYLOAD_LENGTH) {
    Serial.println("Invalid payload length");
    return;
  }
  // Extract the payload
  float temperature, pressure, humidity, iaqScore;
  memcpy(&temperature, &frame[2], sizeof(float)); // Temperature starts at index 2
  memcpy(&pressure, &frame[6], sizeof(float));    // Pressure starts at index 6
  memcpy(&humidity, &frame[10], sizeof(float));   // Humidity starts at index 10
  memcpy(&iaqScore, &frame[14], sizeof(float));   // IAQ score starts at index 14


  // Print the received data
  Serial.println("Received Frame:");
  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.println(" °C");
  Serial.print("Pressure: ");
  Serial.print(pressure);
  Serial.println(" Pa");
  Serial.print("Humidity: ");
  Serial.print(humidity);
  Serial.println(" %");
  Serial.print("IAQ Score: ");
  Serial.println(iaqScore);
  Serial.println("");
}
```

4.2 Code explanation

1. **Imports and Setup**
   - #include <SoftwareSerial.h> brings in the Arduino library that lets you use "soft" RX/TX pins on most Arduino boards.
   - A SoftwareSerial object, mySerial, is created on pins 10 (RX) and 11 (TX). This will be used to communicate with the STM32.
   - In setup(), both the default Serial (for debugging) and mySerial are initialized at 9600 baud.
2. **Frame Definition**
   - The code specifies a frame format with a well-defined start byte (0x7E), end byte (0x7F), and a fixed payload length of 16 bytes.
   - The total frame length is calculated as 2 + PAYLOAD_LENGTH + 1, which includes:
     - 1 byte for the start marker (START_BYTE)
     - 1 byte that indicates the payload length
     - 16 bytes of payload data
     - 1 byte for the end marker (END_BYTE)
   - A buffer frameBuffer is declared to hold exactly one entire frame at a time.
3. **Reading Bytes in the loop()**
   - The code continuously checks (while (mySerial.available())) if there are incoming bytes from the controller.
   - When a new byte (byteReceived) arrives:
     1. If *no frame* is currently in progress and the byte is the START_BYTE, the code marks frameInProgress as true, resets the index to 0, and stores this start byte in the frameBuffer.
     2. If a frame *is* in progress, it just stores the incoming byte in frameBuffer and increments the receivedIndex.
     3. Once receivedIndex reaches the total FRAME_LENGTH, the code checks:
        - If the last byte is indeed the END_BYTE (0x7F).
        - If so, it calls processFrame(frameBuffer). Otherwise, it reports an invalid frame.

4. In either case, after processing or finding an error, it resets frameInProgress to false and receivedIndex to 0, so it can look for the next frame.

4. **Processing a Complete Frame (processFrame())**

- The function first checks if the second byte in the frame (frame[1]) equals the defined PAYLOAD_LENGTH (16). This is a quick sanity check.
- Next, it uses memcpy() to extract four float values (temperature, pressure, humidity, IAQ) from the payload region of the buffer:
    - frame[2..5] → temperature
    - frame[6..9] → pressure
    - frame[10..13] → humidity
    - frame[14..17]→iaqScore
      (Indices 2, 6, 10, 14 are exactly where each float begins in the payload.)

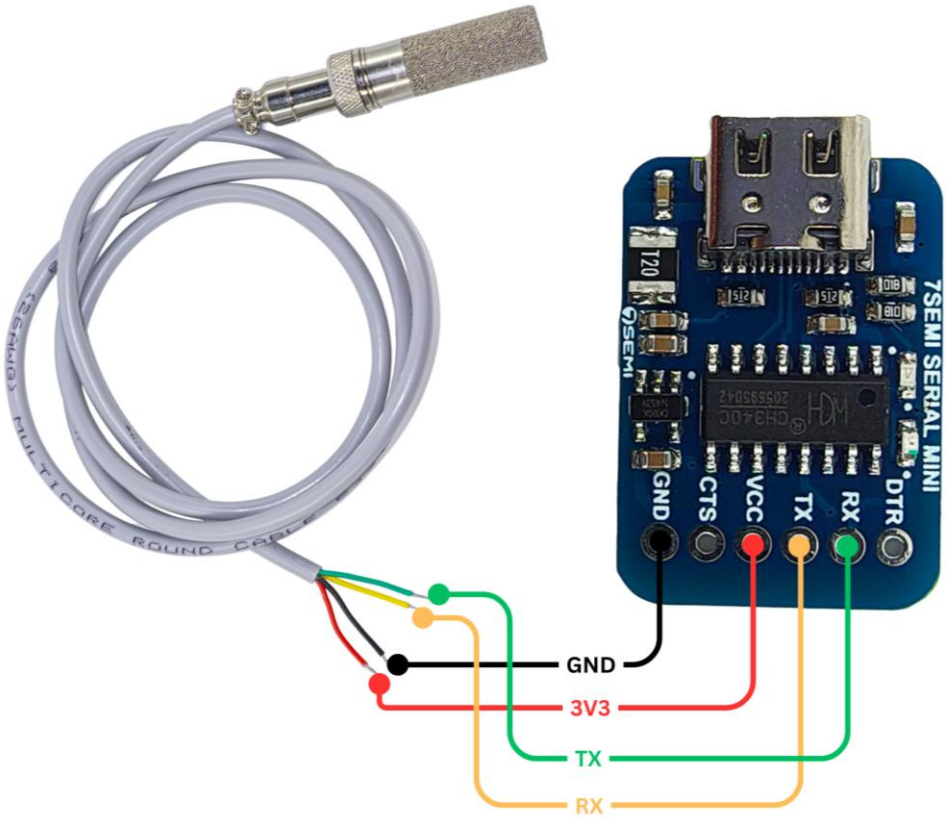
4.3 Serial output

```
Received Frame:
Temperature: 33.69 °C
Pressure: 100934.42 Pa
Humidity: 34.94 %
IAQ Score: 20.00

Received Frame:
Temperature: 33.70 °C
Pressure: 100934.35 Pa
Humidity: 35.02 %
IAQ Score: 20.00

Received Frame:
Temperature: 33.70 °C
Pressure: 100934.25 Pa
Humidity: 35.07 %
IAQ Score: 20.00

Received Frame:
Temperature: 33.93 °C
Pressure: 100933.83 Pa
Humidity: 34.11 %
IAQ Score: 20.00

Received Frame:
Temperature: 33.94 °C
Pressure: 100933.76 Pa
Humidity: 33.86 %
IAQ Score: 20.00
```

4.4 Sample code link

Sample output Image of USB to TTL :

```
7E 10 BE FE 05 42 1F 34 C5 47 76 30 29 42 00 00 A0 40 7F
```



Sample code link:- [ES-50104-U680_UART_RX]

5.Mechanical specification