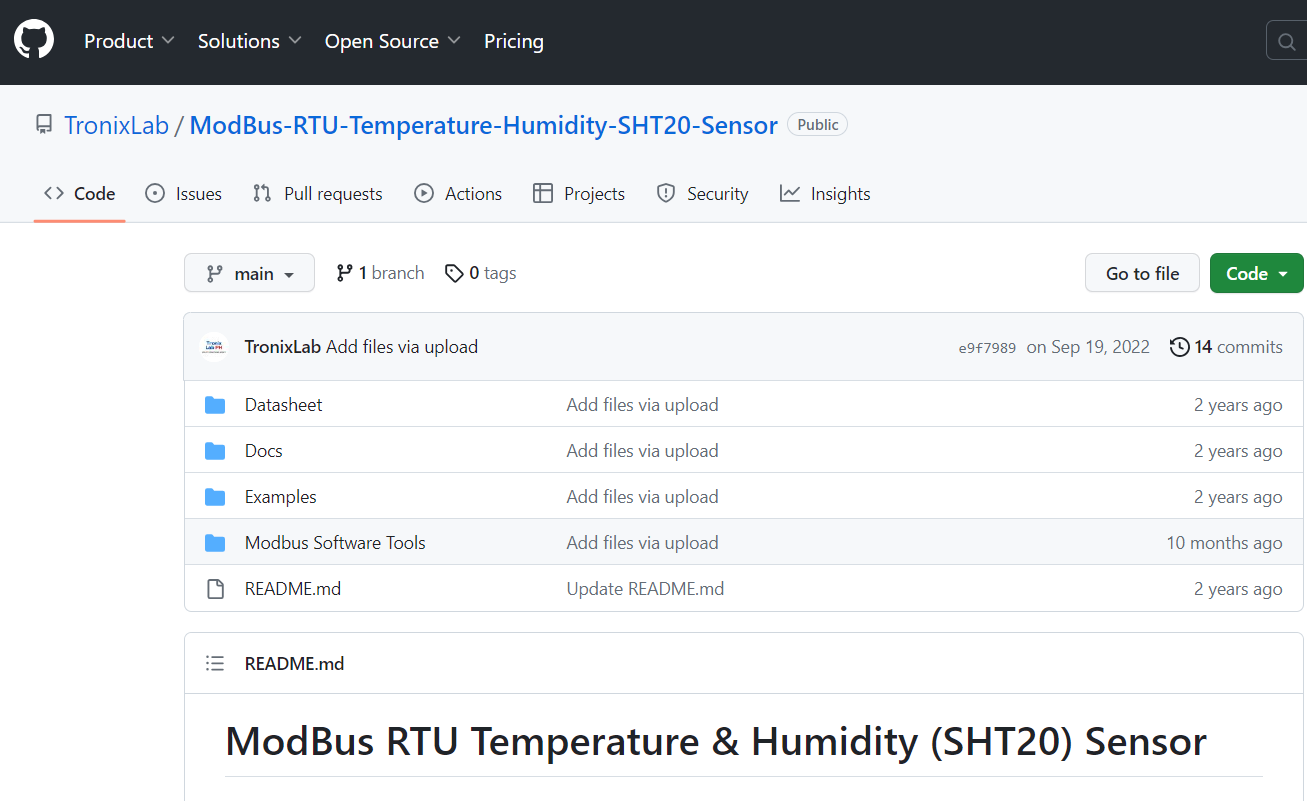
資料來源: <https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor>



## MAX485 TTL to RS-485 Interface Module

| **[space-1.jpg](https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor/blob/main/Docs/MAX485-RS-485-Interface-Module-Connections.jpg)** |
| --- |
| **Fig. 1** MAX485 TTL to RS-485 Interface Module |

### Description

This low-power transceiver module uses the Maxim MAX485 IC to enable serial communication over extremely long cable runs (up to 4000 feet / 1200 meters). At a data rate of up to 2.5Mbps, serial data can be transmitted in both directions (half duplex). The modules run on a standard 5V power supply and use 5V logic levels, allowing them to be interfaced to the hardware serial ports of microcontrollers like Atmega, PIC, ESP, Pico, and among others. It is also possible to connect multiple modules together to communicate with up to 32 devices. In Modbus RTU, RS-485 allows for master/slave communication, with a maximum of 247 slaves for one master. A repeater module is required in the circuit to achieve this.

### Key Features of MAX485 TTL to RS-485 Interface Module

* Use MAX485 Interface chip
* Uses differential signaling for noise immunity
* Distances up to 1200 meters
* Speeds up to 2.5Mbit/Sec
* Multi-drop supports up to 32 devices on same bus
* Red power LED
* 5V operation

**Note**: RS-485 is widely used in industrial environments because it allows for reliable serial communications over large distances of up to 1200 meters (4000 feet) or in electrically noisy conditions. It can support data speeds of up to 2.5MBit/sec, however as distance increases, the maximum data rate that can be comes down.

#### Multi-Drop Support for Multiple Devices

RS-485 has the advantage of supporting numerous devices (up to 32) on the same wire, a feature known as "multi-drop." As demonstrated below, this works by running the bus through each device, which picks up the signal as it passes by. With one Master and one or more Slave devices, these devices are often set up in a Master / Slave arrangement. Because they all ride the same bus, the Slave devices only communicate with each other when the Master requests something, such as a temperature reading.

| **[space-1.jpg](https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor/blob/main/Docs/rs485_line.jpg)** |
| --- |
| **Fig. 2** Multiple Devices Wiring Example |

#### Differential Signaling

Only two wires and a common ground are required for the RS-485, which employs differential signaling. Differential signals work by connecting one wire to the signal and the other wire to the inverse of the signal. As noise tends to couple into both lines equally and thus cancels out at the receiving end, this enhances the signal's noise tolerance and capacity to recover the signal at the far end of the cable.

#### Pull-up Resistors

On the data lines of the module, there are four 10K pull-up resistors. On the A/B differential lines, two 20K resistors are used. When data is not being transferred, these pull the lines to a known condition. A single 120-ohm resistor completes the circuit (R7). To prevent reflections, place this resistor between the A/B differential lines on each end of the cable. The resistors on the two ends of the line should be kept if employed in a multi-drop setup.

### Module Wiring

| **[space-1.jpg](https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor/blob/main/Docs/RS485_Module_HCMODU0081_Diagram.png)** |
| --- |
| **Fig. 3** RS-485 Wiring Guide |

The module has two 4-pin headers on the assembly. The headers are spaced 1.6″ apart, so if using with solderless breadboards, it is necessary to bridge two different breadboards.

**1 x 4 Header (Data side)**

* RO = Receiver Output. Connects to a serial RX pin on the microcontroller
* RE = Receiver Enable. Active LOW. Connects to a digital output pin on a microcontroller. Drive LOW to enable receiver, HIGH to enable Driver
* DE = Driver Enable. Active HIGH. Typically, jumper to RE Pin
* DI = Driver Input. Connects to serial TX pin on the microcontroller

**1 x 4 Header (Output side)**

* VCC = 5V
* B = Data ‘B’ Inverted Line. Common with the B
* A = Data ‘A’ Non-Inverted Line. Connects to A on far end module
* GND = Ground

**1 x 2 Screw Terminal Block (Output side)**

* B = Data ‘B’ Inverted Line. Connects to B on far end module
* A = Data ‘A’ Non-Inverted Line. Connects to A on far end module

A and B are the labels for these two differential data lines. These can be found on the screw terminal block of the module, as well as the two center pins on that end. When connecting the modules, the wiring should be straight through, with A connected to A on one end and B connected to B on the other.

Ideally, the wires should be twisted pair. When running longer distances or where there is a lot of electrical noise, using twisted pair becomes more important. It is not required for simple breadboard testing or other short runs. A common ground is required, but for shorter runs, this can often be provided by the earth ground at each end. Network cable is commonly used for connecting RS-485 because it provides twisted pair and can also provide a ground wire.

### Further Readings

* [Arduino ModBus Tutorial](https://arduinogetstarted.com/tutorials/arduino-modbus)
* [Modbus RTU Master Library for Industrial Automation](https://www.industrialshields.com/blog/arduino-industrial-1/post/modbus-rtu-master-library-for-industrial-automation-200)

## Industrial Grade Temperature and Humidity (SHT20) Sensor

| **MD01 Temperature and Humidity Transmitter SHT20 Sensor** | **XY-MD02 Temperature and Humidity SHT20 Sensor** |
| --- | --- |
|  |  |

### Description

The product uses an industrial-grade chip and high-precision SHT20 temperature and humidity sensors, which ensures the product's reliability, precision, and interoperability. Adopt an RS-485 hardware interface (with lightning protection), with a protocol layer that is compatible with the standard industrial ModBus RTU protocol via RS-485 module serial interface.

### Sensor Wiring Diagram

| **MD01 Temperature and Humidity Transmitter SHT20 Sensor** | **XY-MD02 Temperature and Humidity SHT20 Sensor** |
| --- | --- |
| [https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor/raw/main/Docs/e0b25240316389ce.jpg](https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor/blob/main/Docs/e0b25240316389ce.jpg) | [https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor/raw/main/Docs/sht20-temperature-humidity-sensor-rs485-pmdway-7_856x368.jpg](https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor/blob/main/Docs/sht20-temperature-humidity-sensor-rs485-pmdway-7_856x368.jpg) |

### Dependencies

The [exmample sketches](https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor/tree/main/Examples) are tested in Arduino Uno, Mega and ESP32 boards. To work with, install the [ModBusMaster](https://github.com/4-20ma/ModbusMaster) library for communicating with Modbus slaves over RS232/485 (via RTU protocol).

Refer to the [datasheet](https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor/tree/main/Datasheet) to determine the default Slave ID, baud rate, and hex address of the register that holds the temperature and humidity data. For more information, configuration, and testing procedures, you could use [ModBus software tools](https://github.com/TronixLab/ModBus-RTU-Temperature-Humidity-SHT20-Sensor/tree/main/Modbus%20Software%20Tools) through the use of RS-485 to USB module.