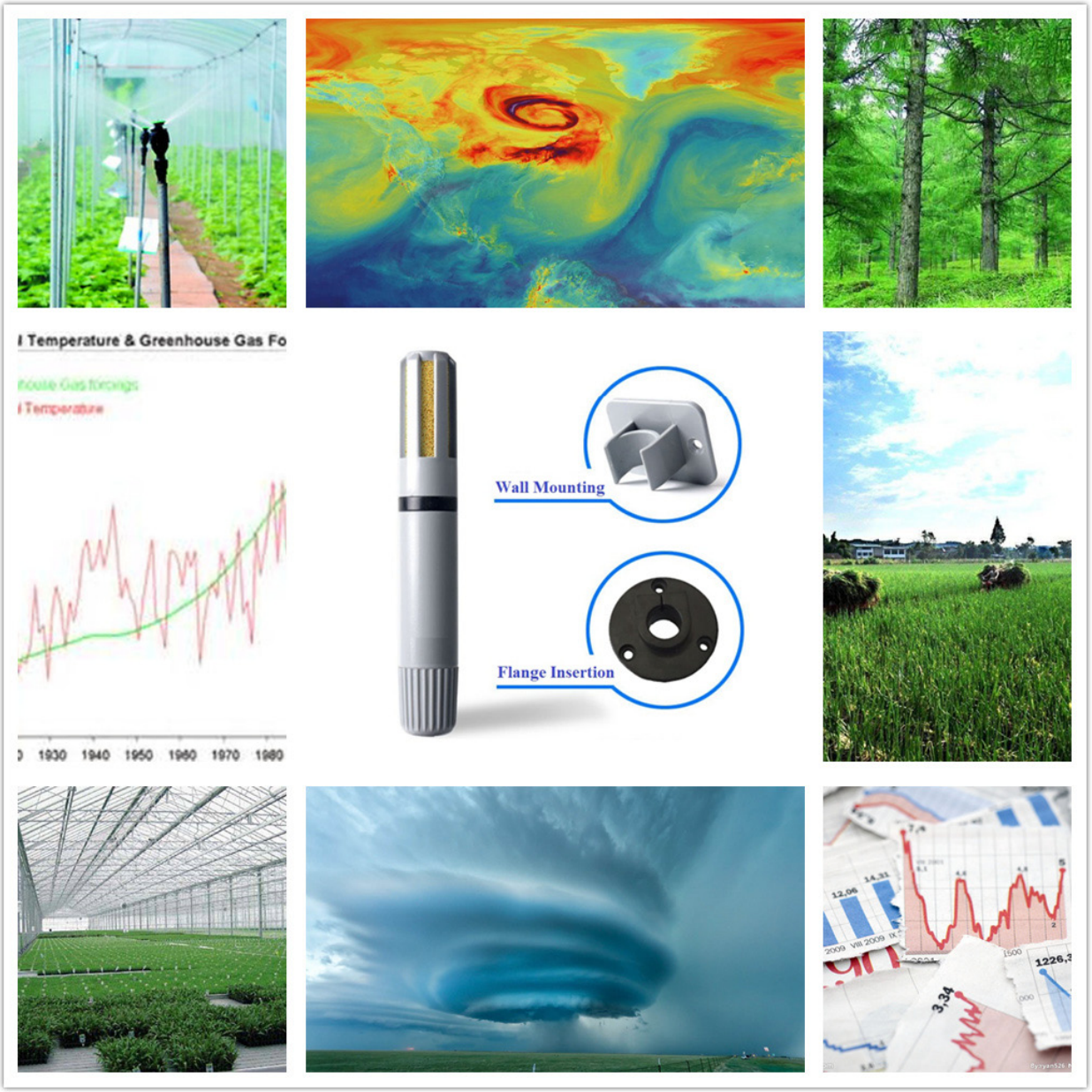
DigiTHP GEN2

Air Temperature/Humidity/Barometric Sensor (SDI-12 Interface)

Air Temperature/Humidity/Barometric Sensor (RS485 Interface)

User Manual



Index

1 Customer Support

Thank you very much for your order. Our success comes from the continuous faith in the excellence of our products and services, something we are committed to and would never sacrifice. Our customer service, especially in the after sales phase, guarantees the satisfaction of our clients. In line with this strategy, we appreciate that you can share with us your feedback at any time for our improvement, be it positive or negative, so if we can serve you better in anyway, please do inform us.

Website

http://www.infwin.com

E-Mail

infwin@163.com

Telephone

+86-411-66831953, +86-4000-511-521

Fax

+86-411-66831953

2 Introduction

2.1 Brief

DigiTHP-Gen2 is an air humidity, temperature and barometric pressure sensor. The sensing elements are located on a PCB in the tip of the sensor. The acquisition circuit is completely over-moulded. Temperature and humidity sesor is based on a precision SENSIRION® sensor. An additional BOSCH®MEMS chip measures barometric pressure. Each sensor is individually factory-calibrated. With dimensions of only 108mm x 16mm, the sensor can easily be fitted into many standard radiation shields and field application.

The SDI-12 output provides universal compatibility with any SDI-12-enabled data logger and low power applications, or use a RS485 physical interface for applications that require long cable runs or many sensors.

Features

- High precision SENSIRION humidity/Temperature sensor and BOSH barometric sensor
- Secondary measurements including dew, frost point, vapor pressure, vapor concentration, etc.
- Digital Output SDI-12 or RS485 with built-in surge protection
- Small size with high accuracy with excellent stability
- Cleanable 10um Copper sintering filter cap
- Wall mounting or Pipe flange insertion installation
- Reverse power protection and Built-in TVS/ESD protection
- Excellent price-performance ratio

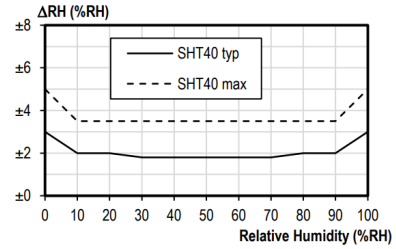
Applications

- Smart agriculture
- Greenhouse monitoring
- Canopy monitoring
- Reference evapotranspiration calculations
- Routine weather monitoring
- Building humidity monitoring
- Mold remediation
- Modeling processes that are affected by vapor pressure or humidity

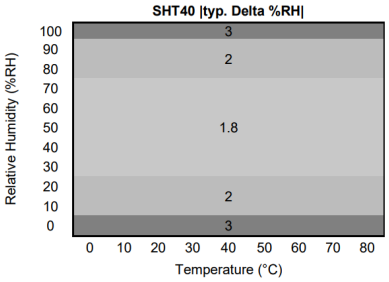
Specifications	
Output Interface	Optional: SDI-12, V1.3 Optional: RS485, Modbus-RTU
Power Supply	4.5-18V/DC
Power Consumption	SDI-12 Interface: Quiescent Current : <10uA RS485 Interface: Quiescent Current : <600uA Measuring Current : 10mA during 50ms measurement
Humidity	Range:0-100%, Resolution:0.01%, Typical Accuracy: +/-1.0% RH(SHT45) Range:0-100%, Resolution:0.01%, Typical Accuracy:+/-1.8% RH(SHT40)
Temperature	Range:-40~125°C, Resolution:0.01°C, Accuracy: +/-0.1°C(SHT45) Range:-40~125°C, Resolution:0.01°C, Accuracy: +/-0.2°C(SHT40) **Attention: The long term operating temperature is -40~80°C**
Barometric	Range: 300-1100mbar, Resolution:0.1mbar Accuracy:+/-0.15mbar(Relative Accuracy),+/-1mbar(Absolute Accuracy)
Operating	-40~80°C, No condensing
IP Ratings	IP56
Installation	Wall mounting or Pipe flange insertion
Cable Length	2 meters or Customize
Dimension	Sensor Body 110*16mm

2.2 Accuracy

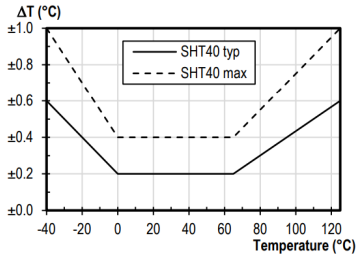
SHT40 – Temperature & Humidity accuracy



SHT40 typical and maximal relative humidity accuracy at 25 °C.

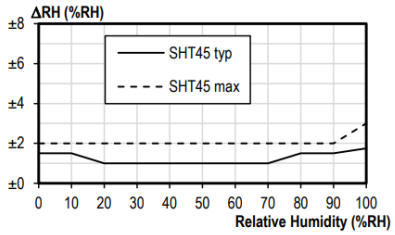


Typical RH accuracy tolerance over humidity and temperature for SHT40.

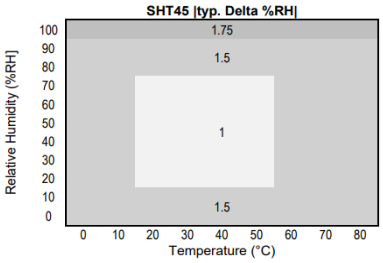


SHT40 typical and maximal temperature accuracy.

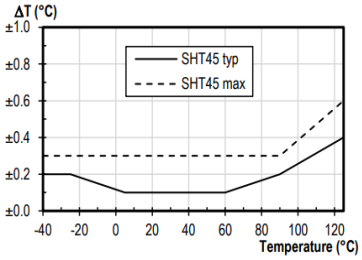
SHT45 – Temperature & Humidity accuracy



SHT45 typical and maximal relative humidity accuracy at 25 °C.



Typical RH accuracy tolerance over humidity and temperature for SHT45.



SHT45 typical and maximal temperature accuracy.

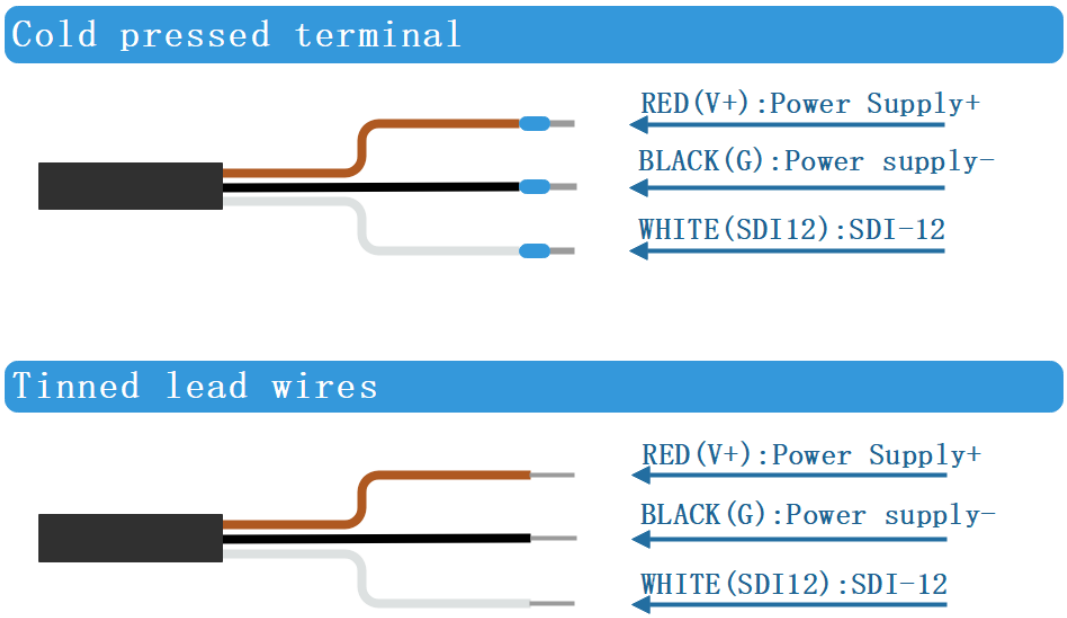
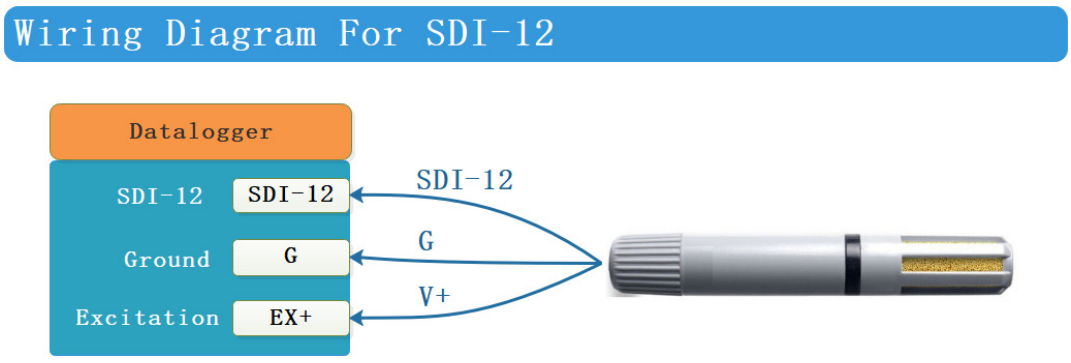
BMP280 – Barometric Sensor

Absolute Accuracy（300-1100hPa）：±1.7hPa（-20~0℃），±1.0hPa（0~65℃）

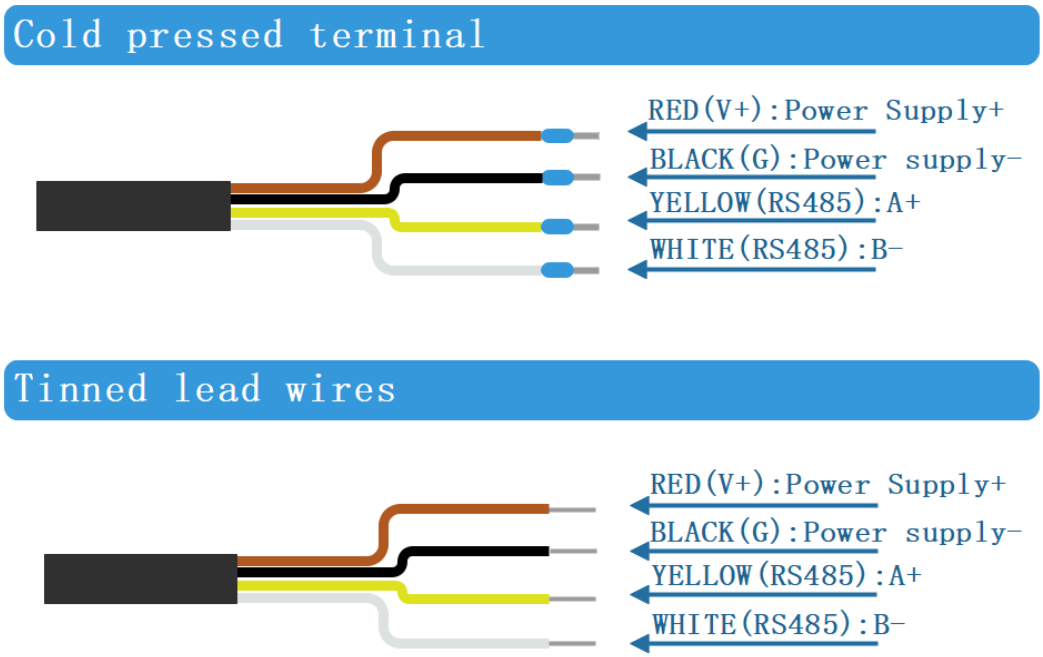
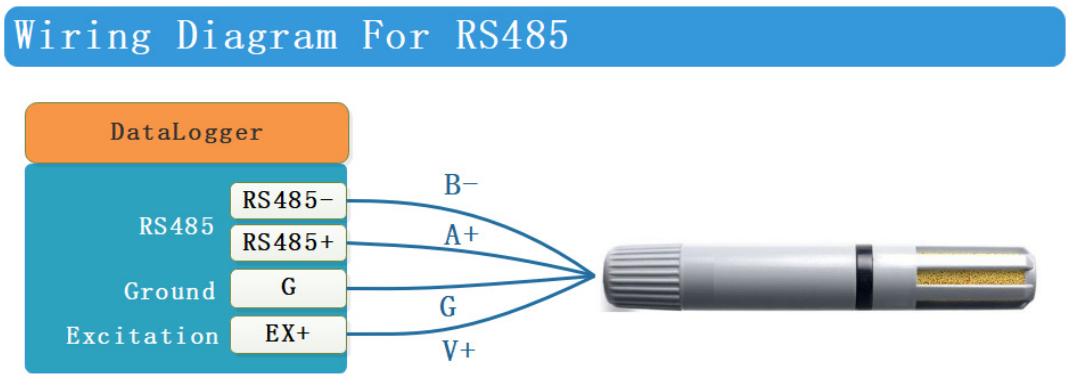
Relative Accuracy（700-900hPa）：±0.12hPa（25~40℃）

3 Wiring diagrams

3.1 SDI-12 Interface

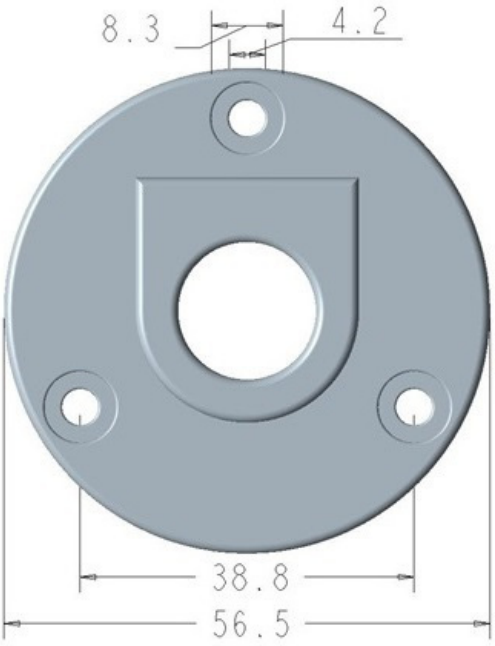
Type	Wiring diagram
SDI-12 Interface	
Connections	

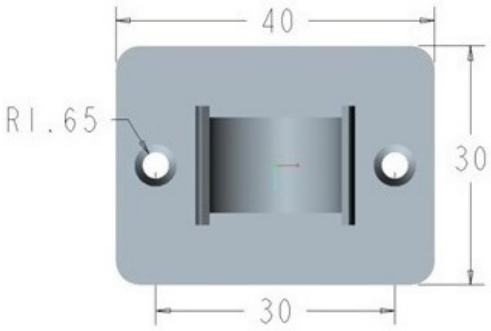
3.2 RS485 Interface

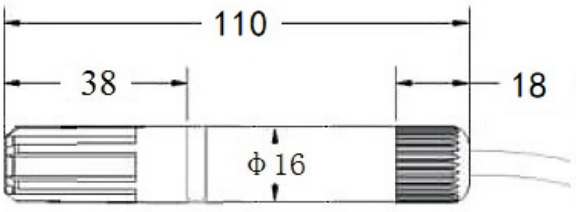
Type	Wiring diagram
RS485 Interface	
Connections	

4 Dimension and Ordering Infomation

4.1 Dimension







Unit: mm

4.2 Ordering Information

Parameters	Code	Comments
Code 1:Product Series	DigiTHP-GEN2	DigiTHP-GEN2, The 2nd Generation DigiTHP sensor
Code 2:Measuring Parameters	A	Air Humidity, Temperature, Barometric measurement, The secondary measurement including: dew, frost point, vapor pressure, vapor concentration, etc.
Code 3: Accuracy	A C	±0.2℃, ±1.8%RH, ±1hPa（SHT40 & BMP280 Typical） ±0.1℃, ±1.0%RH, ±1hPa（SHT45 & BMP280 Typical）
Code 4: Power Supply	C X	4.5-18V DC Customize
Code 5: Output Interface	A B X	RS485(Modbus-RTU) SDI-12 Customize
Code 6: Encapsulation	A X	Standard（ABS+ Copper sintering filter） Customize
Code 7: Connector	B C	Cold pressed terminal Stripped & tinned lead wires
Code 8: Cable Length	002 XXX	2 Meters Customize, XXX is required cable length(Unit: meter)
Ordering Code Example：DigiTHP-GEN2-A A C A A B 002 Product Series: DigiTHP-GEN2, The 2nd Generation DigiTHP sensor; Measuring Parameters: Air Humidity, Temperature, Barometric measurement; Accuracy: ±0.2℃, ±1.8%RH, ±1hPa（SHT40 & BMP280 Typical）; Power Supply: 4.5-18V DC Output Interface: RS485(Modbus-RTU); Encapsulation: Standard（ABS+ Copper sintering filter）; Connector: Cold pressed terminal; Cable Length: 2 Meters		

5 SDI-12 Communication

The sensor has SDI-12 interface and protocol. The description and terms used within this chapter are listed in table below:

Parameters	Unit	Description
±	-	Sign of the value
a	-	SDI-12 address
n	-	Number of measurements (fixed width of 1)
nn	-	Number of measurements with leading zero if necessary (fixed width of 2)
ttt	Seconds	Maximum measurement time (fixed width of 3)
tttt	Seconds	Maximum measurement time (fixed width of 4)
<TAB>	-	Tab character
<SAPCE>	-	Space character
<CR>	-	Carriage return character
<LF>	-	Line feed character
<Checksum>	-	SUM Checksum
<CRC_ADI>		ADI protocol CRC Checksum
<CRC>	-	SDI-12 protocol CRC Checksum
<VERIFY_STATUS>	-	Sensor Verification status
<±temperature>	℃ ℉	Temperature, the value is output according to the temperature unit setting.
<+humidity_%> <+humidity_0to1>	% 0.000-1.000	Humidity
<±dewPoint>	℃ ℉	Dew point, the value is output according to the temperature unit setting.
<+atmosphericPressure_kPa> <+atmosphericPressure_hPa>	kPa hPa	Barometric pressure
<±frostPoint>	℃ ℉	Frost point, the value is output according to the temperature unit setting.
<+vaporPressure_kPa> <+vaporPressure_hPa>	kPa hPa	Vapour pressure
<+vapourConcentration>	g/m3	Vapor concentration
<±CloudbaseEstimationRefByStation>	m	Cloud base estimation (Ref by station)
<±ElevationEstimationRefBySeaLevel>	m	Elevation estimation (Ref by sea level)

The following error value will be responded as measurement value when there is error：

Error Value	Description
-9999	Sensor Broken
-9992	Calibration data corrupted
-9991	Low Power Supply Voltage

5.1 SDI-12 Interface and Protocol

5.1.1 SDI-12 Interface

Please refer to SDI-12 standard user manual V1.3.

5.1.2 Protocol

Request	Response	Comment
a!	a<CR><LF>	**Acknowledge Active** a: Sensor address **Example:** Request: 0! Response: 0<CR><LF>
aI!	allccccccccmmmmmmmvvvxxxxxxxxxxxxxx<CR><LF>	**Send Identification** a: Sensor address ll:SDI-12 Version Number cccccccc: 8 characters vendor identification mmmmmm: 6 characters specifying the sensor model number vvv: 3 characters specifying the sensor version xxxxxxxxxxxxx: 13 characters serial number <CR><LF>: terminates the response **Example:** Request: 0I! Response: 013INFWIN DGTHP 2.02305170016000<CR><LF>
?!	a<CR><LF>	**Sensor Address Query** a:Sensor address **Example:** Request: ?!

		Response: 0<CR><LF>
aAb!	b<CR><LF>	**Change Sensor address** a:Current Sensor address b:New Sensor address **Example:** Request: 0A1! Response: 1<CR><LF>
aM!, aMC!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format： a<+vaporPressure_kPa><±temperature><+humidity_0to1><+atmosphericPressure_kPa>[<CRC>]<CR><LF>	**Vapor Pressure, Temperature, Humidity, Barometric Measurement** **Example:** Request: 0M! Response: 00014<CR><LF> Response: 0<CR><LF> Request: 0D0! Response: 0+1.655+24.2+0.5474+100.329<CR><LF>
aM1!, aMC1!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format： a<±temperature><+humidity_%><±dewPoint><+atmosphericPressure_hPa>[<CRC>]<CR><LF>	**Temperature, Humidity, Dew Point, Barometric Measurement** **Example:** Request: 0M1! Response: 00014<CR><LF> Response: 0<CR><LF> Request: 0D0! Response: 0+24.30+54.64+14.59+1003.36<CR><LF>
aM2!, aMC2!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format：	**Temperature, Humidity, Vapor Pressure, Vapor Concentration Measurement** **Example:** Request: 0M2! Response: 00014<CR><LF> Response: 0<CR><LF> Request: 0D0! Response: 0+23.55+56.46+16.40+11.97<CR><LF>

	a<±temperature><+humidity_%><+vaporPressure_hPa><+vapourConcentration>[<CRC>]<CR><LF>	
aM3!, aMC3!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format： a<±temperature><+humidity_%><±dewPoint><±frostPoint>[<CRC>]<CR><LF>	**Temperature, Humidity, Dew Point, Frost Point Measurement** **Example:** Request: 0M3! Response: 00014<CR><LF> Response: 0<CR><LF> Request: 0D0! Response: 0+23.53+56.38+14.35+14.35<CR><LF>
aM4!, aMC4!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format： a<±temperature><+humidity_%><±dewPoint><±CloudbaseEstimationRefByStation>[<CRC>]<CR><LF>	**Temperature, Humidity, Dew Point, Cloud base Measurement** **Example:** Request: 0M4! Response: 00014<CR><LF> Response: 0<CR><LF> Request: 0D0! Response: 0+23.54+56.47+14.39+1153.46<CR><LF>
aM5!, aMC5!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format： a<±temperature><+humidity_%><+atmosphericPressure_hPa><±ElevationEstimationRefBySeaLevel>[<CRC>]<CR><LF>	**Temperature, Humidity, Dew Point, Elevation Measurement** **Example:** Request: 0M5! Response: 00014<CR><LF> Response: 0<CR><LF> Request: 0D0! Response: 0+23.53+56.71+1002.92+86.31<CR><LF>
aM6!, aMC6!	a0019<CR><LF>	**Temperature, Humidity, Dew Point, Elevation**

	a:Sensor address 001: Measurement data will be ready in 001 seconds 9: Number of measurement data returned by aD0!, aD1, aD2 <CR><LF>:terminates the response aD0! Response data format： a<±temperature><+humidity_%> <±dewPoint><+atmosphericPressure_hPa>[<CRC>]<CR><LF> aD1! Response data format： a<±frostPoint><+vaporPressure_hPa> <+vapourConcentration><CR><LF> aD2! Response data format： a<±CloudbaseEstimationRefByStation> <±ElevationEstimationRefBySeaLevel><CR><LF>	**Measurement** **Example:** Request: 0M6! Response: 00019<CR><LF> Response: 0<CR><LF> Request: 0D0! Response: 0+23.52+56.44+14.36+1003.00<CR><LF> Request: 0D1! Response: 0+14.36+16.36+11.95<CR><LF> Request: 0D2! Response: 0+1154.46+85.64<CR><LF>
aC!, aCC!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format： a<+vaporPressure_kPa><±temperature><+humidity_0to1><+atmosphericPressure_kPa>[<CRC>]<CR><LF>	**Vapor Pressure, Temperature, Humidity, Barometric Measurement** **Example:** Request: 0C! Response: 00014<CR><LF> Request: 0D0! Response: 0+1.655+24.2+0.5474+100.329<CR><LF>
aC1!, aCC1!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format：	**Temperature, Humidity, Dew Point, Barometric Measurement** **Example:** Request: 0C1! Response: 00014<CR><LF> Request: 0D0! Response: 0+24.30+54.64+14.59+1003.36<CR><LF>

	a<±temperature><+humidity_%><±dewPoint><+atmosphericPressure_hPa>[<CRC>]<CR><LF>	
aC2!, aCC2!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format：a<±temperature><+humidity_%><+vaporPressure_hPa><+vapourConcentration>[<CRC>]<CR><LF>	**Temperature, Humidity, Vapor Pressure, Vapor Concentration Measurement** **Example:** Request: 0C2! Response: 00014<CR><LF> Request: 0D0! Response: 0+23.55+56.46+16.40+11.97<CR><LF>
aC3!, aCC3!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format：a<±temperature><+humidity_%><±dewPoint><±frostPoint>[<CRC>]<CR><LF>	**Temperature, Humidity, Dew Point, Frost Point Measurement** **Example:** Request: 0C3! Response: 00014<CR><LF> Request: 0D0! Response: 0+23.53+56.38+14.35+14.35<CR><LF>
aC4!, aCC4!	a0014<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format：a<±temperature><+humidity_%><±dewPoint><±CloudbaseEstimationRefByStation>[<CRC>]<CR><LF>	**Temperature, Humidity, Dew Point, Cloud base Measurement** **Example:** Request: 0C4! Response: 00014<CR><LF> Request: 0D0! Response: 0+23.54+56.47+14.39+1153.46<CR><LF>
aC5!, aCC5!	a0014<CR><LF>	**Temperature, Humidity, Dew Point, Elevation Measurement**

	a:Sensor address 001: Measurement data will be ready in 001 seconds 4: Number of measurement data returned by aD0! <CR><LF>:terminates the response aD0! Response data format： a<±temperature><+humidity_%><+atmosphericPressure_hPa><±ElevationEstimationRefBySeaLevel>[<CRC>]<CR><LF>	**Example:** Request: 0C5! Response: 00014<CR><LF> Request: 0D0! Response: 0+23.53+56.71+1002.92+86.31<CR><LF>
aC6!, aCC6!	a0019<CR><LF> a:Sensor address 001: Measurement data will be ready in 001 seconds 9: Number of measurement data returned by aD0!, aD1, aD2 <CR><LF>:terminates the response aD0! Response data format： a<±temperature><+humidity_%><±dewPoint><+atmosphericPressure_hPa>[<CRC>]<CR><LF> aD1! Response data format： a<±frostPoint><+vaporPressure_hPa><+vapourConcentration><CR><LF> aD2! Response data format： a<±CloudbaseEstimationRefByStation><±ElevationEstimationRefBySeaLevel><CR><LF>	**Temperature, Humidity, Dew Point, Elevation Measurement** **Example:** Request: 0C6! Response: 00019<CR><LF> Request: 0D0! Response: 0+23.52+56.44+14.36+1003.00<CR><LF> Request: 0D1! Response: 0+14.36+16.36+11.95<CR><LF> Request: 0D2! Response: 0+1154.46+85.64<CR><LF>
aV!	A0021<CR><LF> a:Sensor address 002: Measurement data will be ready in 002 seconds 1:Number of measurement data <CR><LF>:terminates the response aD0! Response data format： a<VERIFY_STATUS><CR><LF>	**Sensor Verification Command** **Example:** Request: 0V! Response: 00021<CR><LF> Response: 0<CR><LF> Request: 0D0! Response: 0+0<CR><LF>, “+0” indicate sensor normal, “+1” means sensor error.

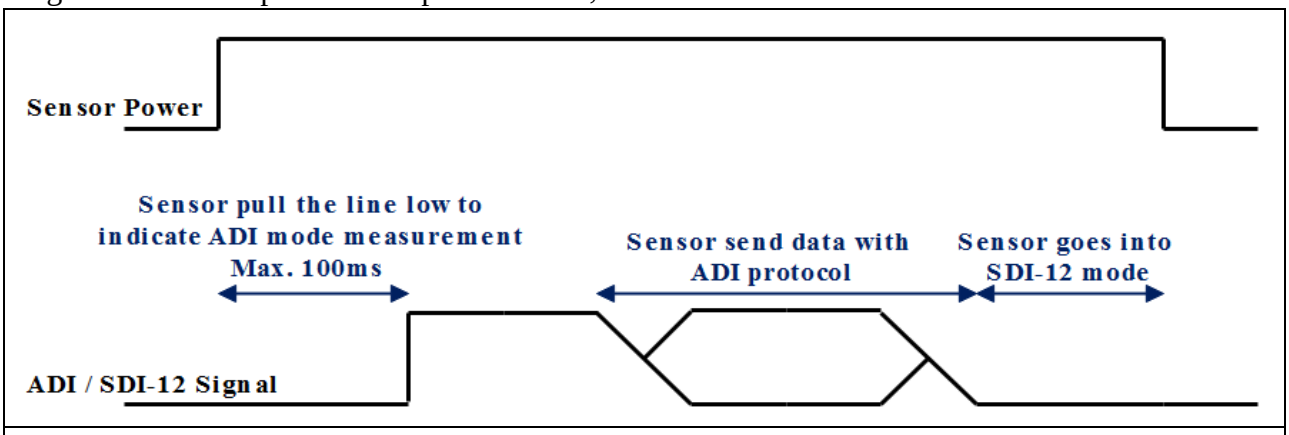
aD0! aD1! aD2!	[<svvvv><svvvv><svvvv>…][<CRC>]<CR><LF> [<svvvv>]: data value [<CRC>]: Optional 3 characters CRC checksum, <CR><LF>:terminates the response	Send Data since the last aM, aMC, aC, aCC, aV command, The data returned depends on the command sent most recently.
aR0!, aRC0!	Response data format： a<+vaporPressure_kPa><±temperature><+humidity_0to1><+atmosphericPressure_kPa>[<CRC>]<CR><LF>	**Vapor Pressure, Temperature, Humidity, Barometric Measurement** **Example:** Request: 0R0! Response: 0+1.655+24.2+0.5474+100.329<CR><LF>
aR1!, aRC1!	Response data format： a<±temperature><+humidity_%><±dewPoint><+atmosphericPressure_hPa>[<CRC>]<CR><LF>	**Temperature, Humidity, Dew Point, Barometric Measurement** **Example:** Request: 0R1! Response: 0+24.30+54.64+14.59+1003.36<CR><LF>
aR2!, aRC2!	Response data format： a<±temperature><+humidity_%><+vaporPressure_hPa><+vapourConcentration>[<CRC>]<CR><LF>	**Temperature, Humidity, Vapor Pressure, Vapor Concentration Measurement** **Example:** Request: 0R2! Response: 0+23.55+56.46+16.40+11.97<CR><LF>
aR3!, aRC3!	Response data format： a<±temperature><+humidity_%><±dewPoint><±frostPoint>[<CRC>]<CR><LF>	**Temperature, Humidity, Dew Point, Frost Point Measurement** **Example:** Request: 0R3! Response: 0+23.53+56.38+14.35+14.35<CR><LF>
aR4!, aRC4!	Response data format： a<±temperature><+humidity_%><±dewPoint><±CloudbaseEstimationRefByStation>[<CRC>]<CR><LF>	**Temperature, Humidity, Dew Point, Cloud base Measurement** **Example:** Request: 0R4! Response: 0+23.54+56.47+14.39+1153.46<CR><LF>
aR5!, aRC5!	Response data format： a<±temperature><+humidity_%><+atmosphericPressure_hPa><±ElevationEstimationRefBySeaLevel>[<CRC>]<CR><LF>	**Temperature, Humidity, Dew Point, Elevation Measurement** **Example:** Request: 0R5! Response: 0+23.53+56.71+1002.92+86.31<CR><LF>
aR6!, aRC6!	Response data format： a<±temperature><+humidity_%>	**Temperature, Humidity, Dew Point, Elevation Measurement**

	<±dewPoint><+atmosphericPressure_hPa><±frostPoint><+vaporPressure_hPa><+vapourConcentration><±CloudbaseEstimationRefByStation><±ElevationEstimationRefBySeaLevel><CR><LF>	**Example:** Request: 0R6! Response: 0+23.52+56.44+14.36+1003.00+14.36+16.36+11.95+1154.46+85.64<CR><LF>
aXR3!, aXR4!	Response data format： a<TAB><vaporPressure><SPACE><temperature><SPACE><atmosphericPressure><CR><sensorType><Checksum><CRC>	**Return data as ADI format** **Example:** Request: 0XR3! Response: 0<TAB>1.830<SPACE>29.1<SPACE>99.97{EF
aXR_TUNIT!	aTUNIT=<X> <X> is temperature unit: C: degrees centigrade F: degrees fahrenheit	**Query temperature unit** **Example:** Request: 0XR_TUNIT! Response: 0TUNIT=C<CR><LF>
aXW_TUNIT_<X>!	aTUNIT=<X>	**Configure temperature unit** **Example:** Request: 0XW_TUNIT_C! Response: 0TUNIT=C<CR><LF>
aXR_ADIEN!	aADIEN=<v> <v>: ADI output enable on sensor start up, when SDI12 is 0 on sensor start up: <v>=0 the sensor will output data in ADI format. <v>=0 the sensor will not output data in ADI format.	**Query ADI output enable** **Example:** Request: 0XR_ADIEN! Response: 0ADIEN=1<CR><LF>
aXW_ADIEN_<v>!	aADIEN=<v>	**Set ADI output enable** **Example:** Request: 0XW_ADIEN_0! Response: 0ADIEN=0<CR><LF>
aXR_SN!	aSN=<ssssssss> <ssssssss> is 8-digits serial number	**Query serial number** **Example:** Request: 0XR_SN! Response: 0SN=12345678<CR><LF>
aXW_SN_<ssssssss>!	aSN=<ssssssss>	**Configure serial number** **Example:** Request: 0XW_SN_ABCDEFGH! Response: 0SN=ABCDEFGH <CR><LF>

5.2 ADI Interface and Protocol

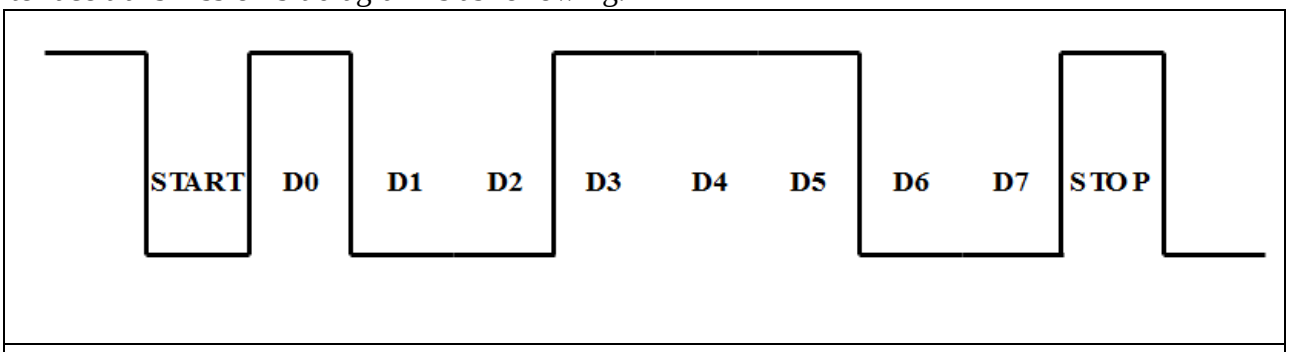
5.2.1 ADI Interface

ADI interface protocol(TTL signal),ADI is short for Active Digital Interface. Each time when sensor power up with SDI-12 address 0,the sensor firstly enter into the ADI mode and pull down the SDI-12 signal line for 100ms to indicating the measurement in processing,then release the SDI-12 signal line and output the ADI protocol data, and then enter into the SDI-12 interface mode.



ADI protocol logic

ADI interface is TTL compatiable standard(0-3.6V),protocol data stream is encoded in ASCII,Baudrate 1200bps,None parity,8 data bits,1 stop bit.The sensor enter into SDI-12 standby mode after the ADI output.You'll need to re power up the sensor again for another ADI output.ADI interface transmission bit diagram is as following.



Transmission bit diagram of the ASCII character'9'(0x39)

5.2.2 Protocol

ADI protocol format:

<TAB><vaporPressure_kPa><SPACE><temperature_C><SPACE><atmosphericPressure_kPa><CR><sensorType><Checksum><CRC_ADI>

Parameters	Description
<TAB>	Tab Character

< vaporPressure_kPa>	Vapor Pressure, Unit in kPa
<SPACE>	Space Character
<temperature_C>	Temperature, Unit in Celsius degree
<SPACE>	Space Character (Not applicable for MT22B)
<atmosphericPressure_kPa>	Atmospheric Pressure in kPa
<CR>	Carriage return character
<sensorType>	ASCII character denoting the sensor type For DigiTHP-GEN2, the character is '{'
<Checksum>	SUM Check from <TAB> to <sensorType>
<CRC_ADI>	ADI protocol CRC Check from <TAB>to < Checksum>

Example: **"<TAB>1.222<SPACE>23.4<SPACE>92.81<CR>{/6"**

Parameters	Description
<TAB>	Tab Character
1.222	Vapor Pressure is 1.222 kPa
<SPACE>	Space Character
23.4	Temperature is 23.4°C
<SPACE>	Space Character
92.81	Atmospheric Pressure is 92.81 kPa
<CR>	Carriage return character
{	sensor type
/	SUM Check
6	ADI protocol CRC Check

ADI Interface <Checksum> calculation:
Using "**<TAB>1.222<SPACE>23.4<SPACE>92.81<CR>{**" as function parameters "char * response" and you will get the <Checksum> '/'

```
uint8_t CalcADIChecksum(const char * response)
{
    uint16_t length;
    uint16_t i;
    uint16_t sum = 0;
    // Finding the length of the response string
    length = strlen(response);
    // Adding characters in the response together
    for(i = 0; i < length; i++)
    {
        sum += response[i];
        if(response[i] == '\r')
        {
            // Found the beginning of the metadata section of the response
            break;
        }
```

```
    }
    // Include the sensor type into the checksum
    sum += response[++i];
    // Convert checksum to a printable character
    sum = sum % 64 + 32;
    return sum;
}
```

ADI Interface <CRC_ADI> calculation:
Using "**<TAB>1.222<SPACE>23.4<SPACE>92.81<CR>{**/" as function parameters "char * response" and you will get the <CRC_ADI> '6'

```
uint8_t CRC6_Offset(const char *buffer)
{
    uint16_t byte;
    uint16_t i;
    uint16_t bytes;
    uint8_t bit;
    uint8_t crc = 0xfc; // Set upper 6 bits to 1's
    // Calculate total message length—updated once the metadata section is found
    bytes = strlen(buffer);
    // Loop through all the bytes in the buffer
    for(byte = 0; byte < bytes; byte++)
    {
        // Get the next byte in the buffer and XOR it with the crc
        crc ^= buffer[byte];
        // Loop through all the bits in the current byte
        for(bit = 8; bit > 0; bit--)
        {
            // If the uppermost bit is a 1...
            if(crc & 0x80)
            {
                // Shift to the next bit and XOR it with a polynomial
                crc = (crc << 1) ^ 0x9c;
            }
            else
            {
                // Shift to the next bit
                crc = crc << 1;
            }
        }
        if(buffer[byte] == '\r')
        {
            // Found the beginning of the metadata section of the response
            // both sensor type and legacy checksum are part of the crc6
```

```
                    // this requires only two more iterations of the loop so reset
                    // "bytes"
                    // bytes is incremented at the beginning of the loop, so 3 is added
                    bytes = byte + 3;
                }
            }
            // Shift upper 6 bits down for crc
            crc = (crc >> 2);
            // Add 48 to shift crc to printable character avoiding \r \n and !
            return (crc + 48);
        }
```

6 RS485 Communication

6.1 Modbus Protocol

Modbus Protocol is widely used to establish master-slave communication between intelligent devices or sensors. A MODBUS message sent from a master to a slave contains the address of the slave, the function code (e.g. 'read register' or 'write register'), the data, and a check sum (LRC or CRC).

The sensor is RS485 interface with Modbus protocol. The default serial communication settings is slave address 1, modbus rtu, 9600bps, 8 databits and 1 stop bit. All communication settings can be changed with modbus command, and take effective after re-power up the sensor.

Following modbus function code are supported by sensor.

Modbus Function Code 0x03 : used for reading holding register.

Modbus Function Code 0x04 : used for reading input register.

Modbus Function Code 0x06 : used for writing single holding register.

Modbus Function Code 0x10: used for writing multiple holding register.

6.2 Modbus Register

Parameters	Register Addr. (HEX/DEC)	Data Type	Modbus Function Code(DEC)	Range and Comments	Default Value
TEMPRATURE	0x0000 /0	INT16 RO	3/4	-4000-12500 for -40.00-125.00 (℃)； -4000-25700 for -40.00-257.00 (℉)	N/A
HUMIDITY	0x0001 /1	INT16 RO	3/4	0-10000 for 0.00-100.00 (%)	N/A
DEWPOINT	0x0002 /2	INT16 RO	3/4	-10000-12500 for -100.00-125.00 (℃) -14800-25700 for -148.00-257.00 (℉)	N/A
AtmosphericPressure	0x0003 /3	INT16 RO	3/4	3000-11000 for 300.0-1100.0 (hPa)	N/A
FrostPoint	0x0004 /4	INT16 RO	3/4	-10000-12500 for -100.00-125.00 (℃) -14800-25700 for	N/A

				-148.00-257.00 (°F)	
VaporPressure	0x0005 /5	INT16 RO	3/4	0-24000 for 0.0-2400.0 (hPa)	N/A
VaporConcentration	0x0006 /6	INT16 RO	3/4	0-13000 for 0.0-1300.0 (g/m3)	N/A
CloudbaseEstimation RefByStation	0x0007 /7	INT16 RO	3/4	0-20500 for 0-20500 (m)	N/A
ElevationEstimationR efBySeaLevel	0x0008 /8	INT16 RO	3/4	-700-9200 for -700-9200 (m)	N/A
RESERVED	0x0009 /9	INT16 RO	3/4	Reserved	0
RESERVED	0x000A /10	INT16 RO	3/4	Reserved	0
RESERVED	0x000B /11	INT16 RO	3/4	Reserved	0
RESERVED	0x000C /12	INT16 RO	3/4	Reserved	0
RESERVED	0x000D /13	INT16 RO	3/4	Reserved	0
RESERVED	0x000E /14	INT16 RO	3/4	Reserved	0
RESERVED	0x000F /15	INT16 RO	3/4	Reserved	0
TEMPUNIT	0x0020 /32	UINT16 RW	3/6/16	0:℃ 1:℉	0
SLAVEADDRESS	0x0200 /512	UINT16 RW	3/6/16	0-255	1
BAUDRATE	0x0201 /513	UINT16 RW	3/6/16	0-5 0:1200bps 1:2400bps 2:4800bps 3:9600bps 4:19200bps 5:38400bps	3:9600bps
PROTOCOL	0x0202 /514	UINT16 RW	3/6/16	0 0:Modbus RTU	0:Modbus RTU
PARITY	0x0203 /515	UINT16 RW	3/6/16	0-2 0:NONE 1:EVEN 2:ODD	0:NONE
DATABITS	0x0204 /516	UINT16	3/6/16	1	1:8 databits

		RW		1:8 databits	
STOPBITS	0x0205 /517	UINT16 RW	3/6/16	0-1 0:1 stopbit 1:2 stopbits	0:1 stopbit
RESERVED	0x0206 /518	UINT16 RW	3/6/16	Reserved	0
RESERVED	0x0207 /519	UINT16 RW	3/6/16	Reserved	0
USERSN	0x0220 /544 0x0221 /545 0x0222 /546 0x0223 /547	UINT16 RW	3/16	0x0000000000000000- 0xFFFFFFFFFFFFFFFF	N/A
TEMPRATURE_FLOAT	0x1000 /4096	FLOAT RO	3/4	-40.00-125.00 (℃) -40.00-257.00 (℉)	N/A
HUMIDITY_FLOAT	0x1002 /4098	FLOAT RO	3/4	0.00-100.00 (%)	N/A
DEWPOINT_FLOAT	0x1004 /4100	FLOAT RO	3/4	-100.00-125.00 (℃) -148.00-257.00 (℉)	N/A
AtmosphericPressure_FLOAT	0x1006 /4102	FLOAT RO	3/4	300.00-1100.00 (hPa)	N/A
FrostPoint_FLOAT	0x1008 /4104	FLOAT RO	3/4	-100.00-125.00 (℃) -148.00-257.00 (℉)	N/A
VaporPressure_FLOAT	0x100A /4106	FLOAT RO	3/4	0.00-2400.00 (hPa)	N/A
VaporConcentration_FLOAT	0x100C /4108	FLOAT RO	3/4	0.00-1300.00 (g/m3)	N/A
CloudbaseEstimationRefByStation_FLOAT	0x100E /4110	FLOAT RO	3/4	0.00-20500.00 (m)	N/A
ElevationEstimationRefBySeaLevel_FLOAT	0x1010 /4112	FLOAT RO	3/4	-700.00-9200.00 (m)	N/A
RESERVED_FLOAT	0x1012 /4114	FLOAT RO	3/4	Reserved	0
RESERVED_FLOAT	0x1014 /4116	FLOAT RO	3/4	Reserved	0
RESERVED_FLOAT	0x1016 /4118	FLOAT RO	3/4	Reserved	0
RESERVED_FLOAT	0x1018 /4120	FLOAT RO	3/4	Reserved	0
RESERVED_FLOAT	0x101A /4122	FLOAT RO	3/4	Reserved	0
RESERVED_FLOAT	0x101C /4124	FLOAT	3/4	Reserved	0

		RO			
RESERVED_FLOAT	0x101E /4126	FLOAT RO	3/4	Reserved	0
TEMPRATURE_FLOAT_INVERSE	0x1100 /4352	FLOAT _INVERSE RO	3/4	-40.00-125.00 (℃) -40.00-257.00 (℉)	N/A
HUMIDITY_FLOAT_INVERSE	0x1102 /4354	FLOAT _INVERSE RO	3/4	0.00-100.00 (%)	N/A
DEWPOINT_FLOAT_INVERSE	0x1104 /4356	FLOAT _INVERSE RO	3/4	-100.00-125.00 (℃) -148.00-257.00 (℉)	N/A
AtmosphericPressure_FLOAT_INVERSE	0x1106 /4358	FLOAT _INVERSE RO	3/4	300.00-1100.00 (hPa)	N/A
FrostPoint_FLOAT_INVERSE	0x1108 /4360	FLOAT _INVERSE RO	3/4	-100.00-125.00 (℃) -148.00-257.00 (℉)	N/A
VaporPressure_FLOAT_INVERSE	0x110A /4362	FLOAT _INVERSE RO	3/4	0.00-2400.00 (hPa)	N/A
VaporConcentration_FLOAT_INVERSE	0x110C /4364	FLOAT _INVERSE RO	3/4	0.00-1300.00 (g/m3)	N/A
CloudbaseEstimationRefByStation_FLOAT_INVERSE	0x110E /4366	FLOAT _INVERSE RO	3/4	0.00-20500.00 (m)	N/A
ElevationEstimationRefBySeaLevel_FLOAT_INVERSE	0x1110 /4368	FLOAT _INVERSE RO	3/4	-700.00-9200.00 (m)	N/A
RESERVED_FLOAT_INVERSE	0x1112 /4370	FLOAT _INVERSE RO	3/4	Reserved	0

RESERVED_FLOAT_INVERSE	0x1114 /4372	FLOAT_INVERSE RO	3/4	Reserved	0
RESERVED_FLOAT_INVERSE	0x1116 /4374	FLOAT_INVERSE RO	3/4	Reserved	0
RESERVED_FLOAT_INVERSE	0x1118 /4376	FLOAT_INVERSE RO	3/4	Reserved	0
RESERVED_FLOAT_INVERSE	0x111A /4378	FLOAT_INVERSE RO	3/4	Reserved	0
RESERVED_FLOAT_INVERSE	0x111C /4380	FLOAT_INVERSE RO	3/4	Reserved	0
RESERVED_FLOAT_INVERSE	0x111E /4382	FLOAT_INVERSE RO	3/4	Reserved	0

UINT16:16 bit unsigned integer

INT16:16bit signed integer

RO: Register is Read Only

R/W: Register is Read/Write

HEX: Hexadecimal (data with 0x/0X prefix)

DEC: Decimal

FLOAT: If IEEE754 little-endian byte order is [A,B,C,D], the FLOAT format is [B,A,D,C]

FLOAT_INVERSE: If IEEE754 little-endian byte order is [A,B,C,D], the FLOAT_INVERSE format is [D,C,B,A]

The register value will be set as following when there is sensor error：

Error Value	Register Address	Error
-32768	0x0000-0x000F	-32768
-32768	0x1000-0x101E	-32768
-32768	0x1100-0x111E	-32768

6.3 Modbus Register Detail Descripton

TEMPERATURE --- Temperature Value（16 bit signed int）		
Range	-4000-12500 for -40.00-125.00 (Temperature Unit set to ℃)； -4000-25700 for -40.00-257.00 (Temperature Unit set to℉)	Default: N/A
Parameter Storage	N/A	

TEMPERATURE_FLOAT --- Temperature Value, FLOAT Format **TEMPERATURE_FLOAT_INVERSE --- Temperature Value, FLOAT_INVERSE Format**		
Range	-40.00-125.00 (Temperature Unit set to ℃)； -40.00-257.00 (Temperature Unit set to℉)	Default: N/A
Parameter Storage	N/A	

HUMIDITY --- Humidity（16 bit unsigned int）		
Range	0-10000 for 0.00-100.00 (%)	Default: N/A
Parameter Storage	N/A	

HUMIDITY_FLOAT --- Humidity, FLOAT Format **HUMIDITY_FLOAT_INVERSE --- Humidity, FLOAT_INVERSE Format**		
Range	0.00-100.00 (%)	Default: N/A
Parameter Storage	N/A	

DEWPOINT --- Dew Point（16 bit signed int）		
Range	-10000-12500 for -100.00-125.00 (Temp Unit set to ℃)； -14800-25700 for -148.00-257.00 (Temp Unit set to℉)	Default: N/A
Parameter Storage	N/A	

DEWPOINT_FLOAT --- Dew Point, FLOAT Format **DEWPOINT_FLOAT_INVERSE --- Dew Point, FLOAT_INVERSE Format**		
Range	-100.00-125.00 (Temp Unit set to ℃)； -148.00-257.00 (Temp Unit set to℉)	Default: N/A
Parameter Storage	N/A	

AtmosphericPressure--- Atmospheric Pressure（16 bit unsigned int）		
Range	3000-11000 for 300.0-1100.0 (hPa)	Default: N/A
Parameter Storage	N/A	

AtmosphericPressure_FLOAT --- AtmosphericPressure, FLOAT Format **AtmosphericPressure_FLOAT_INVERSE --- AtmosphericPressure, FLOAT_INVERSE Format**		
Range	300.0-1100.0 (hPa)	Default: N/A
Parameter Storage	N/A	

FrostPoint--- Frost Point（16 bit signed int）		
Range	-10000-12500 for -100.00-125.00 (Temp Unit set to ℃)； -14800-25700 for -148.00-257.00 (Temp Unit set to ℉)；	Default: N/A
Parameter Storage	N/A	

FrostPoint_FLOAT --- Frost Point, FLOAT Format **FrostPoint_FLOAT_INVERSE --- Frost Point, FLOAT_INVERSE Format**		
Range	-100.00-125.00 (Temp Unit set to ℃)； ； -148.00-257.00 (Temp Unit set to ℉)；	Default: N/A
Parameter Storage	N/A	

VaporPressure---Vapor Pressure（16 bit signed int）		
Range	0-24000 for 0.0-2400.0 (hPa)	Default: N/A
Parameter Storage	N/A	

VaporPressure_FLOAT --- Vapor Pressure, FLOAT Format **VaporPressure_FLOAT_INVERSE ---Vapor Pressure, FLOAT_INVERSE Format**		
Range	0.00-2400.00 (hPa)	Default: N/A
Parameter Storage	N/A	

VaporConcentration---Vapor Concentration/ Absolute Humidity（16 bit signed int）		
Range	0-13000 for 0.0-1300.0 (g/m3)	Default: N/A
Parameter Storage	N/A	

VaporConcentration_FLOAT ---Vapor Concentration/ Absolute Humidity, FLOAT Format **VaporConcentration_FLOAT_INVERSE ---Vapor Concentration/ Absolute Humidity, FLOAT_INVERSE Format**		
Range	0.00-1300.00 (g/m3)	Default: N/A
Parameter Storage	N/A	

CloudbaseEstimationRefByStation---Cloud base estimation（16 bit signed int）		
Range	0-20500 for 0-20500 (m)	Default: N/A
Parameter Storage	N/A	

Note: Cloud base estimation (Reference by station)

CloudbaseEstimationRefByStation_FLOAT ---Cloud base estimation, FLOAT Format CloudbaseEstimationRefByStation_FLOAT_INVERSE ---Cloud base estimation, FLOAT_INVERSE Format		
Range	0.00-20500.00 (m)	Default: N/A
Parameter Storage	N/A	

Note: Cloud base estimation (Reference by station)

ElevationEstimationRefBySeaLevel---Elevation estimation（16 bit signed int）		
Range	-700-9200 for -700-9200 (m)	Default: N/A
Parameter Storage	N/A	

Note: Cloud base estimation (Reference by sea level)

ElevationEstimationRefBySeaLevel_FLOAT---Elevation estimation, FLOAT Format ElevationEstimationRefBySeaLevel_FLOAT_INVERSE---Elevation estimation, FLOAT_INVERSE Format		
Range	-700.00-9200.00 (m)	Default: N/A
Parameter Storage	N/A	

Note: Cloud base estimation (Reference by sea level)

TEMPUNIT--- Temperature Unit （16 bit unsigned int）		
Data Range	0:℃ 1:℉	Default: 0
Parameter Storage	YES	

Note: Temperature Unit

SLAVEADDRESS --- Modbus Slave Address （16 bit unsigned int）		
Data Range	0-255	Default: 1
Parameter Storage	YES	

Note: Please power up the sensor to take effective after set.

BAUDRATE --- Serial Comm Baudrate （16 bit unsigned int）		
Data Range	0-5 0:1200bps 1:2400bps 2:4800bps 3:9600bps 4:19200bps 5:38400bps	Default: 3
Parameter Storage	YES	

Note: Please power up the sensor to take effective after set.

PROTOCOL --- Serial Comm Protocol （16 bit unsigned int）		
Data Range	0 0:Modbus RTU	Default: 0
Parameter Storage	YES	

Note: Please power up the sensor to take effective after set.

PARITY --- Serial Comm Parity （16 bit unsigned int）		
Data Range	0-2 0:NONE 1:EVEN 2:ODD	Default: 0
Parameter Storage	YES	

Note: Please power up the sensor to take effective after set.

DATABITS --- Serial Comm Databits （16 bit unsigned int）		
Data Range	1 1:8 databits	Default: 1
Parameter Storage	YES	

Note: Please power up the sensor to take effective after set.

STOPBITS --- Serial Comm Stopbits （16 bit unsigned int）		
Data Range	0-1 0:1 stopbit 1:2 stopbits	Default: 0

Parameter Storage	YES	

Note: Please power up the sensor to take effective after set.

6.4 Modbus Function Code

For description below, data started with 0X/0x means that it's in HEX format.

6.4.1 Function Code 3 Protocol Example

Master Request:AA 03 RRRR NNNN CCCC

AA	1 byte	Slave Address,0-255
0x03	1 byte	Function Code 3
RRRR	2 byte	Starting Register Addr
NNNN	2 byte	Quantity of Register to read
CCCC	2 byte	CRC CHECKSUM

Slave Response:AA 03 MM VV0 VV1 VV2 VV3… CCCC

AA	1 byte	Slave Address,0-255
0x03	1 byte	Function Code 3
MM	1 byte	Register Data Byte Count
VV0,VV1	2 byte	Register Value (High8bits first)
VV2,VV3	2 byte	Register Value (High8bits first)
…	…	Register Value (High8bits first)
CCCC	2 byte	CRC CHECKSUM

Example:Read register 0x0200-0x0201,that is slave address and baudrate.

Master Request:01 03 0200 0002 C5B3

Slave Addr.	1 byte	0x01
Function Code	1 byte	0x03
Starting Register Addr.	2 byte	0x0200
Quantity of Register to read	2 byte	0x0002
Checksum	2 byte	0xC5B3

Slave Response:01 03 04 00 01 00 03 EB F2

Slave Addr.	1 byte	0x01
Function Code	1 byte	0x03
Register Data Byte Count	1 byte	0x04
Register Value: Address	2 byte	0x00(HIGH 8 Bits)
		0x01(LOW8 Bits)
Register Value: Baudrate	2 byte	0x00(HIGH 8 Bits)
		0x03(LOW8 Bits)
Checksum	2 byte	0xEBF2

6.4.2 Function Code 4 Protocol Example

Master Request:AA 04 RRRR NNNN CCCC

AA	1 byte	Slave Address,0-255
0x04	1 byte	Function Code 4
RRRR	2 byte	Starting Register Addr
NNNN	2 byte	Quantity of Register to read
CCCC	2 byte	CRC CHECKSUM

Slave Response:AA 04 MM VV0 VV1 VV2 VV3… CCCC

AA	1 byte	Slave Address,0-255
0x04	1 byte	Function Code 4
MM	1 byte	Register Data Byte Count
VV0,VV1	2 byte	Register Value (High8bits first)
VV2,VV3	2 byte	Register Value (High8bits first)
…	…	Register Value (High8bits first)
CCCC	2 byte	CRC CHECKSUM

Example:Read register 0x0000-0x0003,that is temperature, humidity, dew point, Atmospheric Pressure.

Master Request:01 04 0000 0004 F1C9

Slave Addr.	1 byte	0x01

Function Code	1 byte	0x04
Starting Register Addr.	2 byte	0x0000
Quantity of Register to read	2 byte	0x0003
Checksum	2 byte	0xF1C9

Slave Response: 01 04 08 0B 1E 12 AB 06 60 26 FE 26 63

Slave Addr.	1 byte	0x01
Function Code	1 byte	0x04
Register Data Byte Count	1 byte	0x08
Register Value: Temperature	2 byte	0x0B(HIGH 8 Bits)
		0x1E(LOW8 Bits)
Register Value: Reserved	2 byte	0x12(HIGH 8 Bits)
		0xAB(LOW8 Bits)
Register Value: Reserved	2 byte	0x06(HIGH 8 Bits)
		0x60(LOW8 Bits)
Register Value: Reserved	2 byte	0x26(HIGH 8 Bits)
		0xFE(LOW8 Bits)
Checksum	2 byte	0x2663

6.4.3 Function Code 6 Protocol Example

Master Request:AA 06 RRRR VVVV CCCC

AA	1 byte	Slave Address,0-255
0x06	1 byte	Function Code 6
RRRR	2 byte	Register Addr (High8bits first)
VVVV	2 byte	Register Value (High8bits first)
CCCC	2 byte	CRC CHECKSUM

Slave Response:AA 06 RRRR VVVV CCCC

AA	1 byte	Slave Address,0-255
0x06	1 byte	Function Code 6

RRRR	2 byte	Register Addr (High8bits first)
VVVV	2 byte	Register Value (High8bits first)
CCCC	2 byte	CRC CHECKSUM

Example:Write Register 0x0200,that is change modbus slave address to 2.

Master Request: 01 06 0200 0002 09B3

Slave Addr.	1 byte	0x01
Function Code	1 byte	0x06
Register Addr.	2 byte	0x0200
Register Value	2 byte	0x0002
Checksum	2 byte	0x09B3

Slave Response: 01 06 0200 0002 09B3

Slave Addr.	1 byte	0x01
Function Code	1 byte	0x06
Register Addr.	2 byte	0x0200
Register Value	2 byte	0x0002
Checksum	2 byte	0x09B3

6.4.4 Function Code 16 Protocol Example

Master Request:AA 10 RRRR NNNN MM VVVV1 VVVV2 …CCCC

AA	1 byte	Slave Address,0-255
0x10	1 byte	Function Code 0x10
RRRR	2 byte	Starting Register Addr
NNNN	2 byte	Quantity of Register to write
MM	1 byte	Register Data Byte Count
VVVV1	2 byte	Register Value(High8bits first)
VVVV2	2 byte	Register Value(High8bits first)
…	…	Register Value(High8bits first)
CCCC	2 byte	CRC CHECKSUM

Slave Response:AA 10 RRRR NNNN CCCC

AA	1 byte	Slave Address,0-255

0x10	1 byte	Function Code 0x10
RRRR	2 byte	Starting Register Addr
NNNN	2 byte	Quantity of Register to write
CCCC	2 byte	CRC CHECKSUM

Example:Write Register 0x0200-0x0201,that is set slave address to 1,and baudrate to 19200bp.

Master Request:01 10 0200 0002 04 0001 0004 BACC

0x01	1 byte	Slave Addr.
0x10(HEX)	1 byte	Function Code 0x10
0x0200	2 byte	Starting Register Addr
0x0002	2 byte	Quantity of Register to write
0x04	1 byte	Register Data Byte Count
0x0001	2 byte	Register Value: Slave Address 1
0x0004	2 byte	Register Value: Baudrate 19200bps
0xBACC	2 byte	CRC CHECKSUM

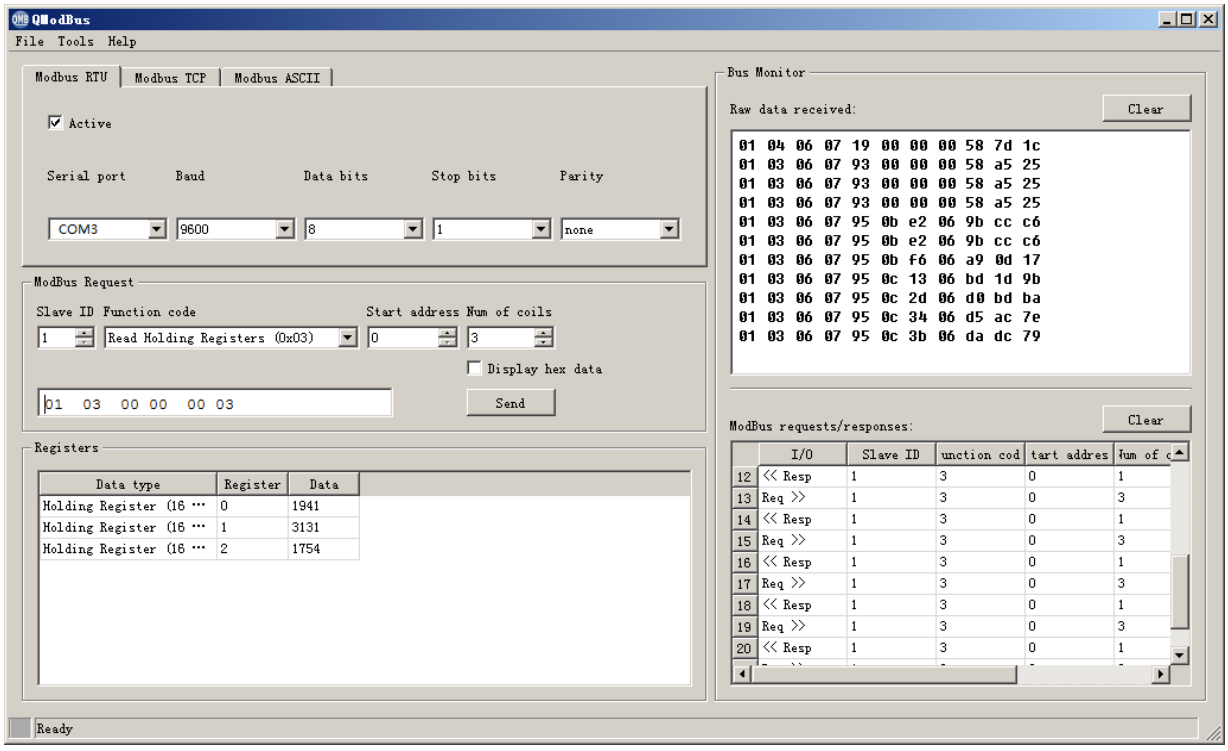
Salve Response:01 10 0200 0002 4070

0x01	1 byte	Slave Addr.
0x10(HEX)	1 byte	Function Code 0x10
0x0200	2 byte	Starting Register Addr(High8bits first)
0x0002	2 byte	Quantity of Register to write(High8bits first)
0x4070	2 byte	CRC CHECKSUM

6.5 Software Configuration Utility

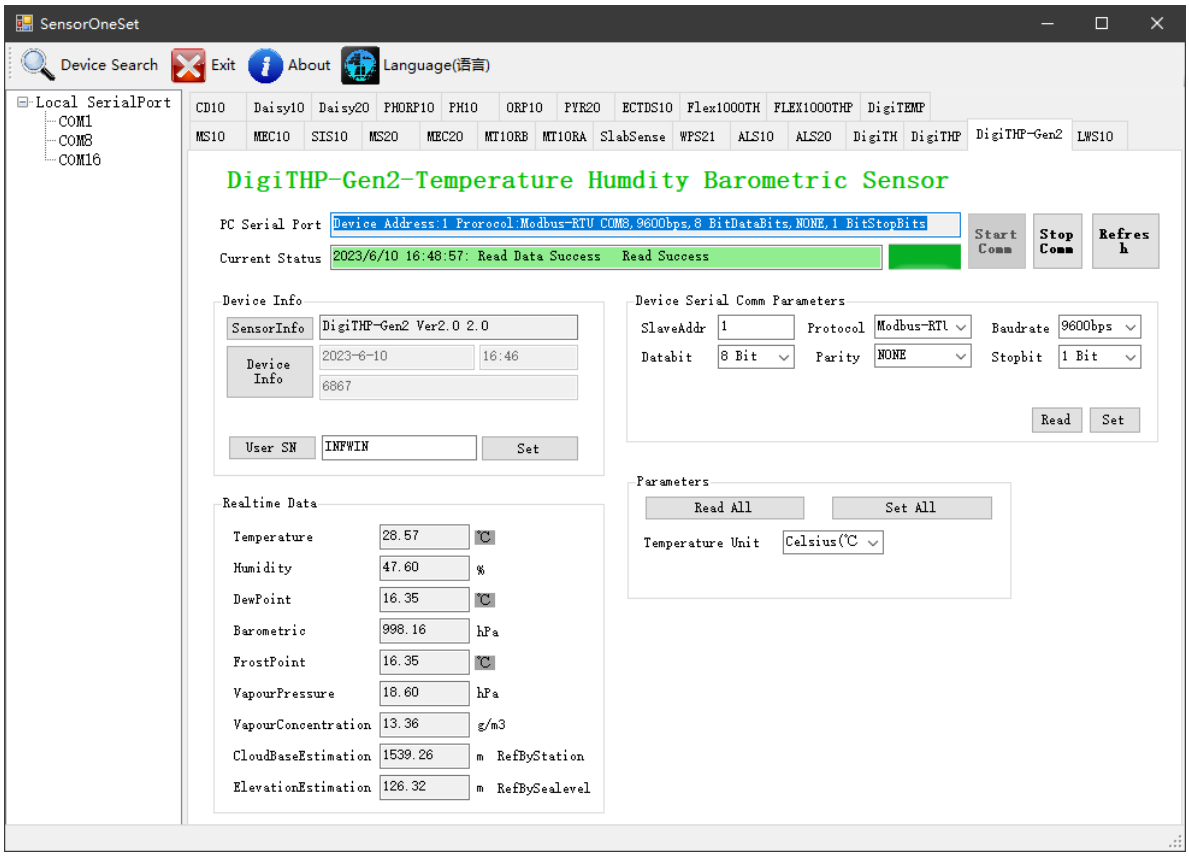
6.5.1 Universal Modbus Comm Utility

You can use software listed below to try reading/writing the register of sensor, https://github.com/ed-chemnitz/qmodbus/releases



6.5.2 SensorOneSet Configuration Utility

SensorOneSet is a configuration utility to read/set sensor config for all of our serial communication sensor products. Please download on our website: https://www.infwin.com



Appendix

Copyright and Trademark

Version Control

Date	Version	Comment	Updated by
2023-06-11	V1.0	Initial Creation	sl51930