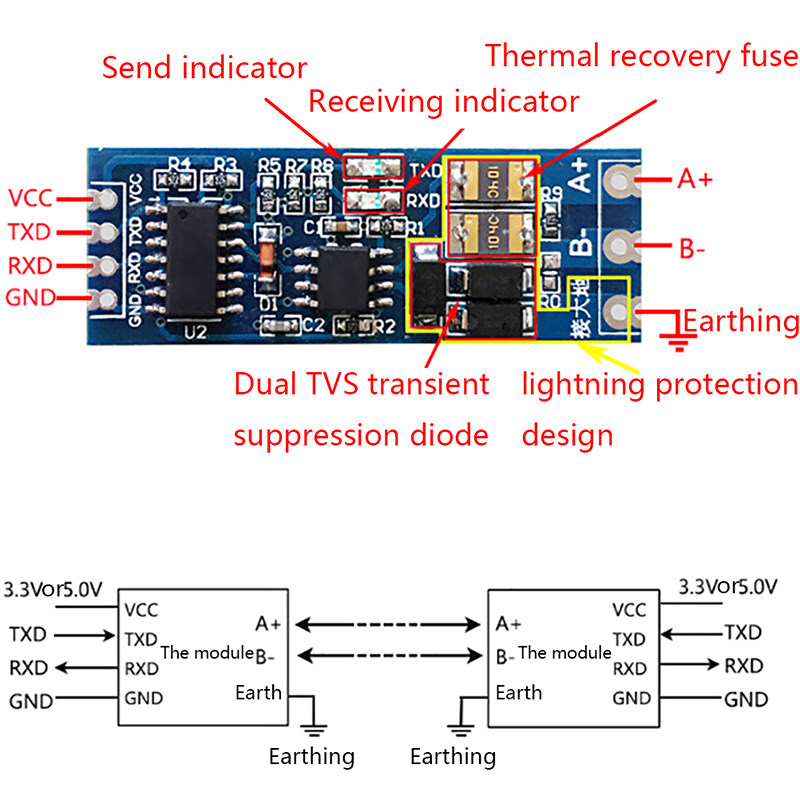
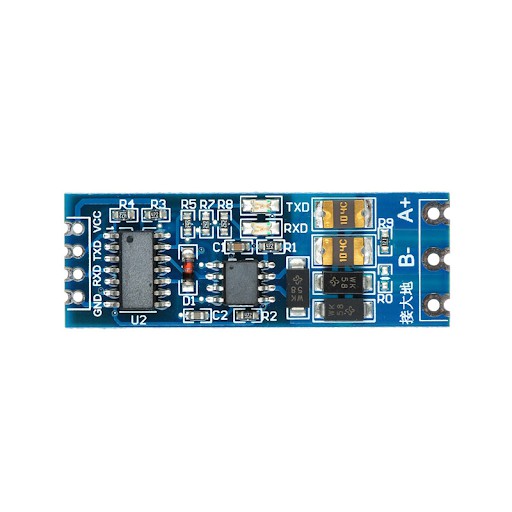
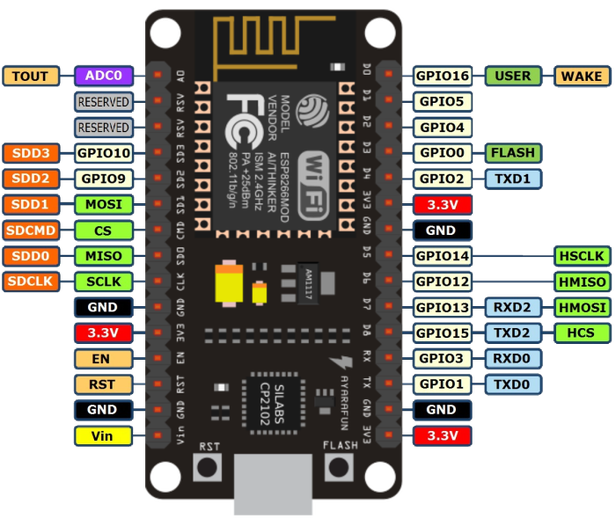
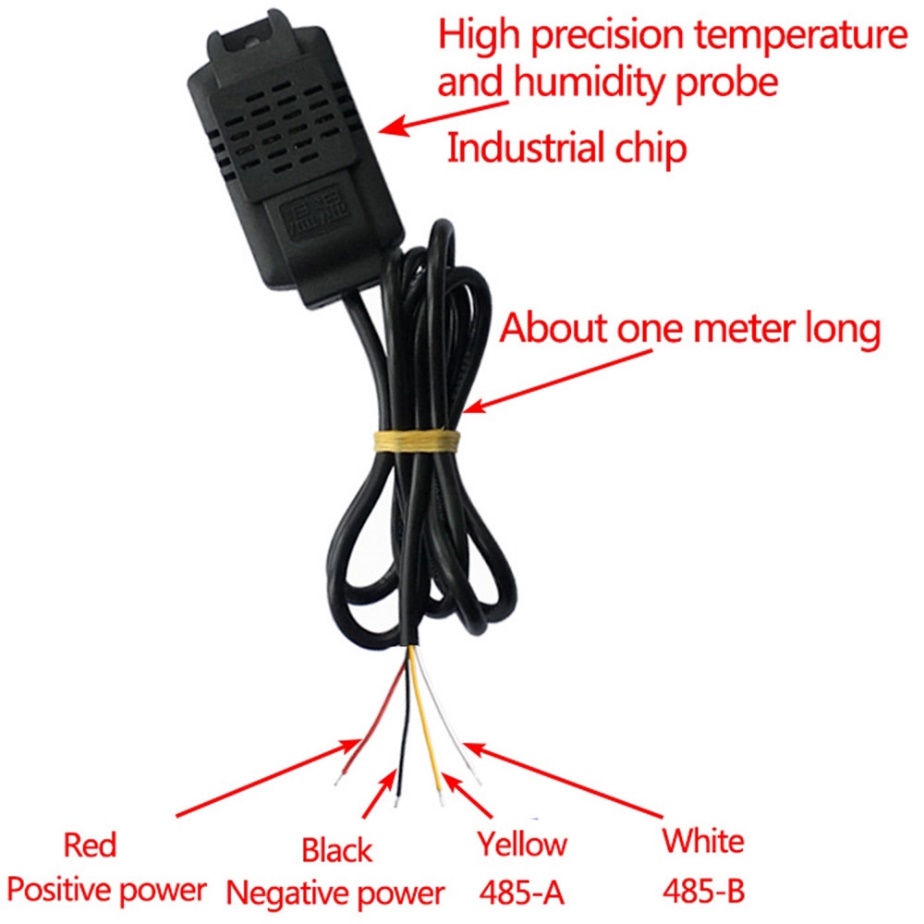
Modbus RS485









Temperature and Humidity Transmitter SHT20 RS485 Sensor Module Digital IIC I2C High Precision Industrial Modbus Rtu Protocol Monitoring

Product introduction:  
The product uses industrial-grade chips and high-precision imported SHT20 temperature and humidity sensors to ensure the excellent reliability, high precision, and interchangeability of the product. Using RS485 hardware interface (with lightning protection design), the protocol layer is compatible with the standard industrial Modbus-RTU protocol. This product integrates the MODBUS protocol and the common protocol. The user can choose the communication protocol by himself. The common protocol has an automatic upload function (connecting RS485 and passing the serial port adjustment tool will automatically output temperature and humidity).  
  
Features:  
Industrial grade product, high progress SHT20 temperature, and humidity sensor, RS485 communication;  
Standard MODBUS protocol and common protocol are integrated, and users can choose communication protocol by themselves;  
The baud rate can be set by yourself;  
The standard protocol has an automatic upload function, and the upload rate can be set by yourself.  
  
Product parameters:  
Working voltage: DC4-30V (the highest should not exceed 33V)  
Maximum power: 0.2W  
Working temperature: temperature -20℃+60℃, humidity 0%RH-100%RH  
Control accuracy: temperature ±0.3℃(25℃), humidity ±3%RH(25℃)  
Output interface: RS485 communication (standard MODBUS protocol and custom common protocol), see protocol description for details  
Device address: 1-247 can be set, the default is 1  
Baud rate: default 9600 (users can set by themselves), 8 data, 1 stop, no parity.

Modbus protocol

products used in the function code:

0 x03:read holding registers

0 x04:read input registers

0 x06:write a single holding register

0 x10:write multiple hold registers

|  |  |  |  |
| --- | --- | --- | --- |
| register type | register Address | the data content | the number of bytes |
| the input register | 0x0001 | the temperature value | 2 |
| 0x0002 | humidity value | 2 |
| the holding register | 0x0101 | device address  (1~247 (in Chinese)) | 2 |
| 0x0102 | the baud rate  0:9600  1:14400  2:19200 | 2 |
| 0x0103 | temperature correction(/10)  -10.0-10.0 | 2 |
| 0x0104 | humidity correction value(/10)  -10.0-10.0 | 2 |

Modbus communication Format:

the host sends a data frame:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | register Address  the high byte | register Address  the low byte | number of registers  the high byte | number of registers  the low byte | CRC  the high byte | CRC  the low byte |
|  |  |  |  |  |  |  |  |

slave response data frame:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | in response to the function code | the number of bytes | register1data  the high byte | register1data  the low byte | registerNdata  the high byte | registerNdata  the low byte | CRC  the high byte | CRC  the low byte |
|  |  |  |  |  |  |  |  |  |

MODBUSthe command frame

host reads temperature command frames(0 x04):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | register Address  the high byte | register Address  the low byte | number of registers  the high byte | number of registers  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x04 | 0x00 | 0x01 | 0x00 | 0x01 | 0x60 | 0x0a |

slave response data frame:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | the number of bytes | temperature  the high byte | temperature  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x04 | 0x02 | 0x01 | 0x31 | 0x79 | 0x74 |

the temperature value= 0 x131,to decimal conversion305actual temperature value= 305 / 10=30.5℃

note: temperature is marked16decimal number, temperature value= 0 xFF33,to decimal conversion-205the actual temperature= -20.5℃;

host reads humidity command frames(0 x04):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | register Address  the high byte | register Address  the low byte | number of registers  the high byte | number of registers  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x04 | 0x00 | 0x02 | 0x00 | 0x01 | 0xC1 | 0xCA |

slave response data frame:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | the number of bytes | humidity  the high byte | humidity  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x04 | 0x02 | 0x02 | 0x22 | 0xD1 | 0xBA |

humidity value= 0 x222,to decimal conversion546actual humidity value= 546 / 10 = 54.6%;

continuously read temperature and humidity command frames(0 x04):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | register Address  the high byte | register Address  the low byte | number of registers  the high byte | number of registers  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x04 | 0x00 | 0x01 | 0x00 | 0x02 | 0x20 | 0x0B |

slave response data frame:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | the number of bytes | temperature  the high byte | temperature  the low byte | humidity  the high byte | humidity  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x04 | 0x04 | 0x01 | 0x31 | 0x02 | 0x22 | 0x2A | 0xCE |

read the content of the maintenance register(0 x03):

take reading the slave address as an example:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | register Address  the high byte | register Address  the low byte | number of registers  the high byte | number of registers  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x03 | 0x01 | 0x01 | 0x00 | 0x01 | 0xD4 | 0x0F |

slave response frame:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | the number of bytes | the slave address  the high byte | the slave address  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x03 | 0x02 | 0x00 | 0x01 | 0x30 | 0x18 |

modify the content of the maintenance register(0 x06):

take modifying the slave address as an example:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | register Address  the high byte | register Address  the low byte | the register value  the high byte | the register value  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x06 | 0x01 | 0x01 | 0x00 | 0x08 | 0xD4 | 0x0F |

modify the slave address: 0x08=8

the slave response frame(and send the same):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | register Address  the high byte | register Address  the low byte | the register value  the high byte | the register value  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x06 | 0x01 | 0x01 | 0x00 | 0x08 | 0xD4 | 0x0F |

continuous modify holding register(0 x10):

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | the start address  the high byte | the start address  the low byte | number of registers  the high byte | number of registers  the low byte | the number of bytes | register1the high byte | register1the low byte | register2the high byte | register2the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x06 | 0x01 | 0x01 | 0x00 | 0x02 | 0x04 | 0x00 | 0x20 | 0x25 | 0x80 | 0x25 | 0x09 |

modify the slave address: 0x20=32

the baud rate: 0x2580=9600

slave response frame:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| the slave address | function Code | register Address  the high byte | register Address  the low byte | number of registers  the high byte | number of registers  the low byte | CRC  the high byte | CRC  the low byte |
| 0x01 | 0x06 | 0x00 | 0x11 | 0x00 | 0x04 | 0xD4 | 0x0F |

the ordinary version protocol

the baud rate of the default9600(Users can set it by themselves),8bit Data,1bit stop, no verification

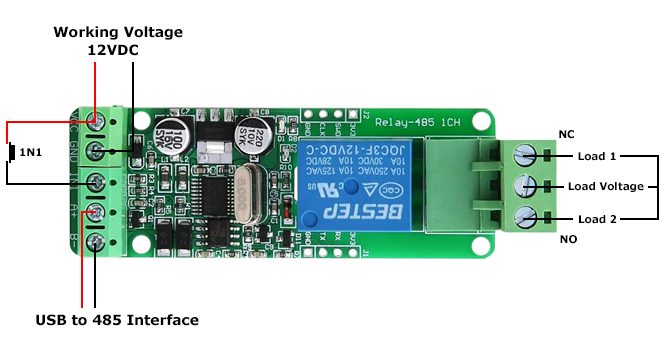
RS485 communication

|  |  |
| --- | --- |
| serial commands | description |
| READ | trigger a temperature and humidity report  (27.4℃67.7%temperature27.4℃ humidity67.7%) |
| AUTO | start the automatic report function of temperature and humidity  (ditto) |
| STOP | stop Automatic temperature and humidity reporting function |
| BR:XXXX | set the baud rate  from 9600 to 19200  (BR:9600the baud rate9600) |
| TC:XX.X | set the temperature correction  (-10.0~10.0)  (TC:02.0temperature correction2.0℃) |
| HC:XX.X | setting the humidity calibration  (-10.0~10.0)  (HC:-05.1humidity correction value is-5.1%) |
| HZ:XXX | set the temperature and humidity report rate  (0.5,1,2,5,10)  (HZ:2not only automatically reported rate2) |
| PARAM | read the current system settings |

PARAMinstructions:

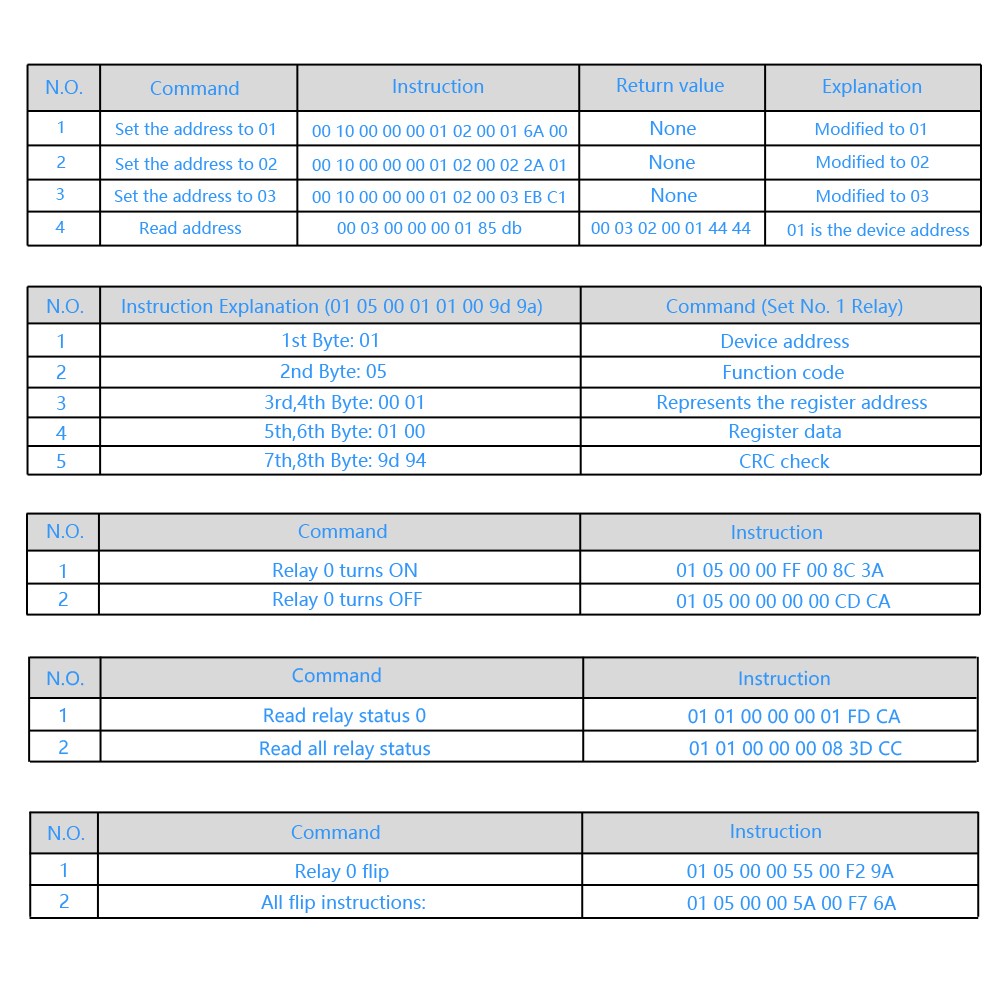
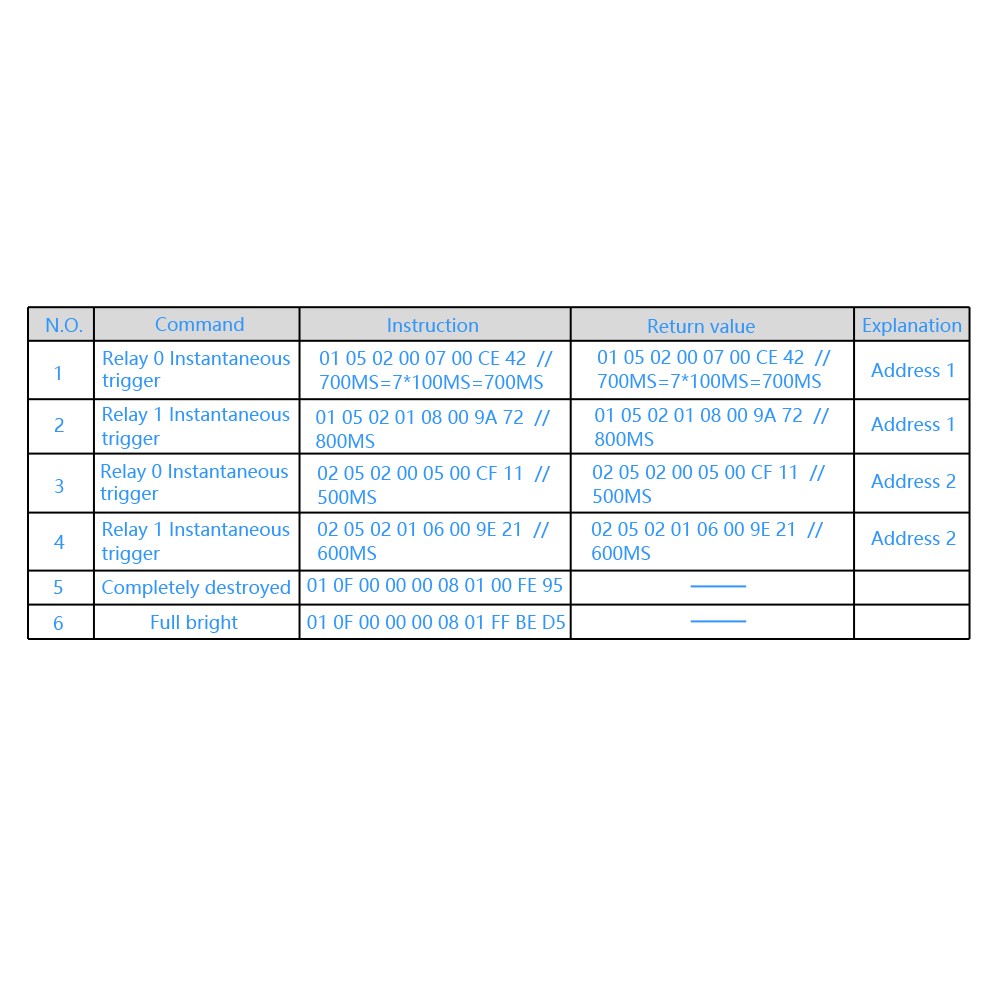
TC:0.0,HC:0.0,BR:9600,HZ:1->temperature correction0.0humidity correction value0.0the baud rate9600report rate1Hz

SLAVE\_ADD:1->MODBUSthe slave address0x01

  
Arduino UNO R3 -> RS485 module  
            • 5V -> VCC  
            • GND -> GND  
            • 2 -> RO  
            • 3 -> DI  
            • 4 -> DE  
            • 5 -> RE  
  
Arduino UNO R3 -> SHT20  
            • 5V -> ( + )  
            • GND -> ( - )  
  
RS485 module -> SHT20  
            • VCC -> ( + )  
            • GND -> ( - )  
            • B -> B-  
            • A -> A+

This is a one channel Modbus RTU relay module equipped with stable 8-bit MCU and RS485 level communication chip, adopting standard MODBUS RTU format RS485 communication protocol. It can realize 2-bit input signal detection and provides a 2-bit relay output. It can be used for digital detection or power control occasions.  
  
Quick Spec

* Model: GY18123
* Work Voltage: 7 ~ 24VDC
* Baud Rate: 4800/9600/19600bps (default 9600bps)
* Optocoupler Input Signal: 3.3 ~ 30VDC
* Set Address: 1 ~ 255
* Relay Contorl Mode: ON/OFF, Delay\_ON, Delay\_OFF mode
* Delay Time: 0 ~ 6553.5s
* Load: AC 250V 10A / DC 28V 10A
* Protocol: Modbus RTU
* Interface: RS485/TTL UART
* Control Channel: 1 channel
* Operating Temperature: -20 ~ +85°C
* Operating Humidity: 5% ~ 95%RH
* Module Size: 85 x 49 x 19mm
* Weight: 24g



Crc check

https://crccalc.com/

http://www.ip33.com/crc.html