| | Document Number | Document Version | Document Classification |
|---|---|---|---|
| | 05130010 | A/4 | Internal Public |
| **Protocol Name** | U3-EC MODBUS | | |

# U3-EC MODBUS MODBUS

# Contents

# Air conditioner and Host Computer Modbus Protocol

## 1 Description

This protocol describes the protocol for command control and data exchange between the air conditioner and its dedicated upper computer monitoring module.

The functions stipulated in the protocol mainly include:

1) The host computer obtains the relevant information of the air conditioner by sending read commands.

2) The host computer sets relevant parameters and action control by sending write commands.

During the communication process, the host computer is the master node and exchanges information through a question and answer method, various information and parameters in the slave nodes use the target register as the storage address, and the master node completes the read and write commands by accessing the registers. This protocol supports the networking of one master node and multiple slave nodes. The slave nodes are distinguished by addresses. The address setting range is 1-127. Different slave nodes correspond to different addresses. Slave nodes with the same address cannot hang on the same communication.

## 2 Terminology

Master node: background monitoring system.

Slave node: air conditioner controller.

RS485: A serial communication standard that supports half-duplex serial short-range communication.

Read command: sent from the master node to the slave node, so that the slave node returns the content of the corresponding register.

Write command: The master node packages related parameters and sends them to the slave node to complete the corresponding parameter settings.

Register address: Each signal and parameter of the slave node corresponds to a 2-byte address. The master node obtains related information or sets related parameters by accessing these registers. This address is called a register.

# 3 Physical interface

## 3.1 Serial communication interface electrical standard

The slave node communicates with the master node through the serial port in RS485 mode.

## 3.2 Information transmission method

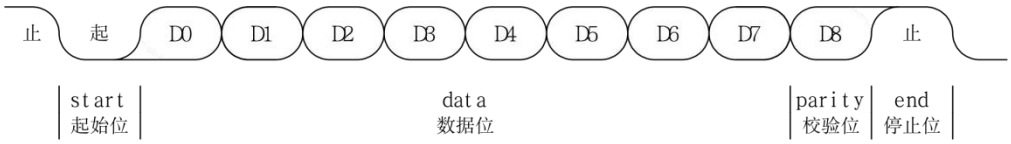
Communication transmission adopts asynchronous mode, and the unit is byte frame (data frame). Each data frame passed between the master node and the slave node is an 11-bit serial data stream.

Data frame format:

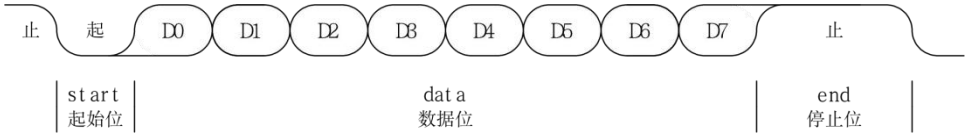
| | |
|---|---|
| Start bit | 1 bit |
| Data bit | 8 bits (Low position first, high position back) |
| Parity bit | None: This protocol does not use parity bits |
| Stop bit | 1 bit (I.e., the actual parity bits forced high) |

Reference

Timing diagram with parity bit:




Timing diagram without parity bit:




## 3.3 Data transfer rate

The default baud rate uses 19200bps

# 4 Physical layer communication

## 4.1 Basic process

After the slave node is powered on or reset, it can respond to the read and write commands of the master node after stable operation. After receiving the relevant command from the contact, the information required by the master node is returned under normal conditions, and the error code corresponding to the specific error type is returned under abnormal conditions.

# 5 Application layer command types and formats

When the communication command is sent to the instrument, the device that meets the corresponding address code receives the communication command, reads the information, and if there is no error, executes the corresponding task, and then returns the execution result to the sender. The returned information includes the address code, the function code to perform the action, the data after the action is performed, and the error check code (CRC). If an error occurs, no message is sent.

## 5.1 Information frame format

| START | ADDR | CMD | DATA | CRC | END |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| Delay (>=3.5 character time) | 1 byte 8 bit | 1 byte 8 bit | N byte N×8 bit | 2 byte 16 bit | Delay(>=3.5 character time) |

Note 1: The maximum frame length is not greater than 255 bytes

Note2: CRC check code low byte first, high byte last

## 5.2 CMD

The function code (CMD) is the second data frame in the information frame transmitted with each communication. The communication ModBus protocol defines the function code as1～127(01H～7FH). This protocol uses some of these function codes. It is sent as a master node request, and the slave node is informed by a function code what action to perform. As a slave node response, the function code sent by the slave node is the same as the function code sent by the master node, and indicates that the slave node has responded to the master node to perform the operation. If the most significant bit of the function code sent from the node is 1(CMD＞127), It means that the slave is not responding or has an error.

| Command encoding | Meaning | Mark |
|---|---|---|
| 0x03 | Read command | Supports single/multiple register continuous reads |
| 0x10 | Write multiple register commands | Support multiple registers to write continuously |
| 0x06 | Write single register command | Support single registers to write continuously |

## 5.3 Read command format

Note 3: MSB means high byte, LSB means low byte

Note 4: Each register stores two bytes. for register data of one byte, it is required to be stored in the low byte (LSB).

Master node sends frame format:

| No. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| Definition | ADDR | CMD | MSB | LSB | MSB | LSB | LSB | MSB |
| Explanation | Controller address | Command type | Register start address | | Number of registers n | | CRC verify | |

Normal response frame format from the slave node:

| No. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | ... | L+1 | L+2 | L+3 | L+4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Definition | ADDR | CMD | Length | MSB | LSB | MSB | LSB | ... | MSB | LSB | LSB | MSB |
| Explanation | Controller address | Command type | Number of sent bytes L = n * 2 | First register | | Second register | | ... | The value of the last register | | CRC verify | |

Slave node abnormal response format:

| No. | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| Definition | ADDR | CMD + 128 | ErrCode | LSB | MSB |
| Explanation | Controller address | Command type +128 | Error Code | CRC verify | |

## 5.4 Write single register command format

Master node sends frame format:

| No. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| Definition | ADDR | CMD | MSB | LSB | MSB | LSB | LSB | MSB |
| Explanation | Controller address | Command type | Register address | | DATA | | CRC verify | |

Normal response frame format from the slave node:

| No. | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| Definition | CMD | MSB | LSB | MSB | LSB | LSB | MSB |
| Explanation | Controller address | Command type | | Register address | | CRC verify | |

Slave node abnormal response format:

| No. | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| Definition | ADDR | CMD +128 | ErrCode | LSB | MSB |
| Explanation | Controller address | Command type +128 | Error Code | CRC verify | |

## 5.5 Write multiple register command format

Master node sends frame format:

| No. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | ... | L+5 | L+6 | L+7 | L+8 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Definition | ADDR | CMD | MSB | LSB | MSB | LSB | Length | MSB | LSB | MSB | LSB | ... | MSB | LSB | LSB | MSB |
| Explanation | Controller address | Command type | Start register address | | Number of registers n | | Number of sent bytes L = n * 2 | First register | | Second register | | ... | The value of the last register | | CRC verify | |

Normal response frame format from the slave node:

| No. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| Definition | ADDR | CMD | MSB | LSB | MSB | LSB | LSB | MSB |
| Explanation | Controller address | Command type | Start register address | | Number of registers | | CRC verify | |

Slave node abnormal response format:

| No. | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| Definition | ADDR | CMD +128 | ErrCode | LSB | MSB |
| Explanation | Controller address | Command type +128 | Error Code | CRC verify | |

Note: The CRC check range is the check of all bytes before the CRC field.

## 5.6 Error code definition

When the slave device sends a request to the master node device, the slave node

expects a normal response. One of the following four possible events occurs in the query from the master node:

1) If the slave device receives a request without communication errors and can process the query normally, the slave device will return a normal response.

2) If the slave node does not receive the request due to a communication error, the response cannot be returned. The master node program will eventually process the timeout status of the request.

3) If the slave receives the request, but detects a communication error (parity, LRC, CRC, ...), then the response cannot be returned. The master node program will eventually process the timeout status of the request.

4) If the slave receives a request without communication errors, but cannot process the request (for example, if the request reads a non-existent output or register), the slave will return an exception response to notify the user of the nature of the error.

The abnormal response message has two fields that are different from the normal response:

**Function code field:** In normal response, the slave node uses the response function code field to respond to the originally requested function code. The most significant bit (MSB) of all function codes is 0 (their values are all lower than 128). In the abnormal response in the slave node, the MSB of the function code is set to 1. This makes the function code value in the abnormal response higher than the function code value in the normal response by 128.

By setting the MSB of the function code, the application of the master node can identify the abnormal response and can detect the data field of the abnormal code.

**Data domain:** In the normal response, the slave node can return data or statistics table (any message required in the request) in the data domain. In the exception response, the slave node returns the exception code in the data domain. This defines the occurrence of

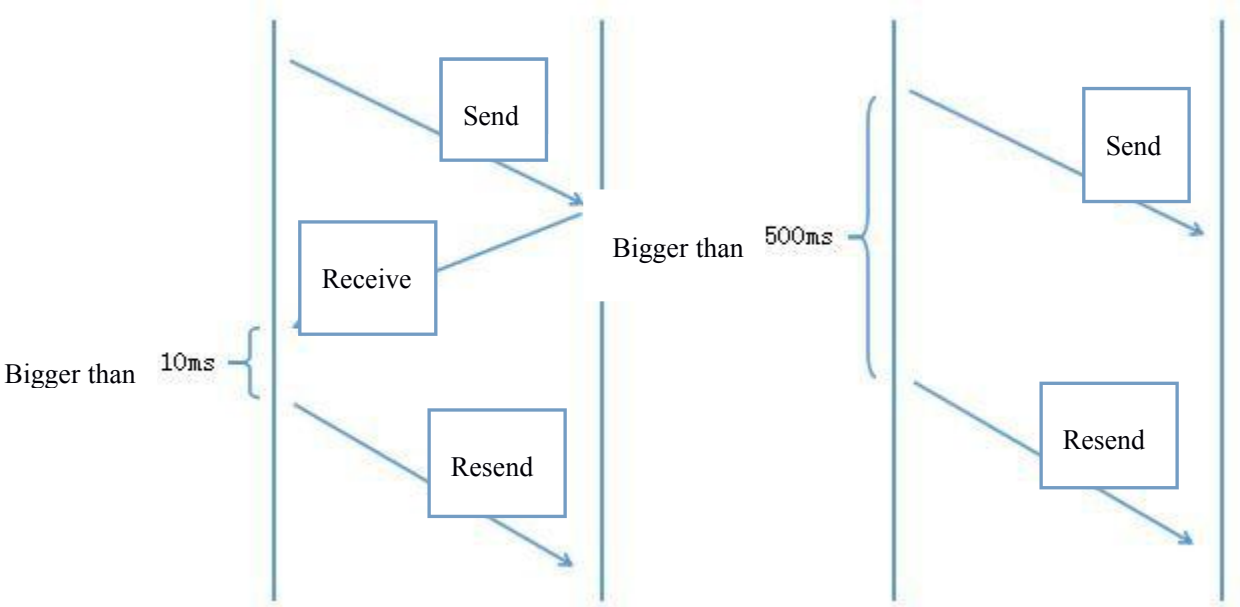
an exception Status of the slave node.

**MODBUS Error code:**

| Error code name | Description |
|---|---|
| 01 Illegal Function Code | For the slave node, the function code received in the query is an inadmissible operation. This may be because the function code is only applicable to new equipment and is not achievable in the selected unit. It also indicates that the slave node is in an error state. This kind of request is handled in, for example, because it is unconfigured and requires the return of register values. |
| 02 Illegal data address | For slave nodes, the data address received in the query is an unallowable address. In particular, the combination of the reference number and the transmission length is invalid. For a controller with 100 registers, it has an offset Requests with 96 and length 4 will succeed, and requests with offset 96 and length 5 will generate exception code 02. |
| 03 Illegal data value | For slave nodes, the value included in the query is an unallowable value. This value indicates a fault in the remaining structure of the combined request, for example: the implicit length is incorrect. It does not mean that because the MODBUS protocol does not know anything The significance of any special value of a special register.The data item in the register that is submitted for storage has a value that is not expected by the application. |

## 5.7 Interval time

After sending data and receiving data, the pins may not switch when sending the next frame, so the unified sending and receiving interval is greater than 10ms and the timeout is greater than 500ms.




# 6 CRC verify algorithm

## 6.1 CRC algorithm

```
unsigned short count_CRC (unsigned char *addr, int num)
{
    unsigned short CRC = 0xFFFF.
    int i.
    while (num--)
    {
        CRC ^= *addr++.
        for (i = 0. i < 8. i++)
        {
            CRC = (CRC & 0x0001) ? ((CRC >> 1)^0xa001) : (CRC >> 1).
        }
    }
    return CRC.
}
```

# 7 Register List:

**The protocol is normal modbus communication protocol**

**1.** Each register deposit two bytes.

2. Data transmission mode: high byte in the front, and the low byte.

| NO. | Item | unit | Register Address | Attribute （Read/Write） | Ratio | Remark |
|---|---|---|---|---|---|---|
| Version Information | | | | | | |
| 1 | Software version | | 0x0000 | Read Only | x1 | |
| Running Status（0:Stop,1:Running, 2:Invalid） | | | | | | |
| 1 | Unit running status | | 0x0100 | Read Only | x1 | |
| 2 | Internal fan status | | 0x0101 | Read Only | x1 | |
| 3 | External fan status | | 0x0102 | Read Only | x1 | |
| 4 | Compressor status | | 0x0103 | Read Only | x1 | |
| 5 | Heater status | | 0x0104 | Read Only | x1 | |
| 6 | Emergency fan status | | 0x0105 | Read Only | x1 | |
| Sensor Status（The invalid value of Temp. is 2000. The invalid value of humidity is 120. The invalid value of humidity is 32767.） | | | | | | |
| 1 | Evaporator Temp. | ℃ | 0x0500 | Read Only | x 10 | |
| 2 | Outdoor Temp. | ℃ | 0x0501 | Read Only | x 10 | |
| 3 | Condenser Temp. | ℃ | 0x0502 | Read Only | x 10 | |
| 4 | Indoor Temp. | ℃ | 0x0503 | Read Only | x 10 | |
| 5 | Humidity | % | 0x0504 | Read Only | x 1 | |
| 6 | Discharge Temp. | ℃ | 0x0505 | Read Only | x 10 | |
| 7 | AC running current | A | 0x0506 | Read Only | x 1000 | |
| 8 | AC input voltage | V | 0x0507 | Read Only | x 1 | |
| 9 | DC input voltage | V | 0x0508 | Read Only | x 10 | |
| Alarm Status（Normal:0, Fault:1） | | | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | High Temp. alarm | | 0x0600 | Read Only | x1 | Indoor temp is higher than the high temp setting point |
| 2 | Low Temp. alarm | | 0x0601 | Read Only | x1 | Indoor temp is lower than the low temp setting point |
| 3 | High humidity alarm | | 0x0602 | Read Only | x1 | NA |
| 4 | Low humidity alarm | | 0x0603 | Read Only | x1 | NA |
| 5 | Coil freeze protection | | 0x0604 | Read Only | x1 | The evaporator coil temp is lower than the coil temp setting point (the default setting is 0°C) |
| 6 | High exhaust Temp. alarm | | 0x0605 | Read Only | x1 | NA |
| 7 | Evaporator Temp. sensor failure | | 0x0606 | Read Only | x1 | Coil temp sensor failed |
| 8 | Outdoor Temp. sensor failure | | 0x0607 | Read Only | x1 | NA |
| 9 | Condenser Temp. sensor failure | | 0x0608 | Read Only | x1 | Condenser Temp. Sensor failed |
| 10 | Indoor Temp. sensor failure | | 0x0609 | Read Only | x1 | Indoor Temp. sensor failure |
| 11 | Exhaust Temp. sensor failure | | 0x060A | Read Only | x1 | NA |
| 12 | Humidity sensor failure | | 0x060B | Read Only | x1 | NA |
| 13 | Internal fan failure alarm | | 0x060C | Read Only | x1 | NA |
| 14 | External fan failure alarm | | 0x060D | Read Only | x1 | NA |
| 15 | Compressor failure alarm | | 0x060E | Read Only | x1 | NA |
| 16 | Heater failure alarm | | 0x060F | Read Only | x1 | NA |
| 17 | Emergency fan failure alarm | | 0x0610 | Read Only | x1 | NA |
| 18 | HP. alarm | | 0x0611 | Read Only | x1 | The refrigeration system pressure is exceeding the set pressure of the system |
| 19 | LP. alarm | | 0x0612 | Read Only | x1 | NA |
| 20 | Water alarm | | 0x0613 | Read Only | x1 | NA |

| | | | | | | |
|---|---|---|---|---|---|---|
| 21 | Fire alarm | | 0x0614 | Read Only | x1 | NA |
| 22 | Gating alarm | | 0x0615 | Read Only | x1 | NA |
| 23 | HP. lock | | 0x0616 | Read Only | x1 | System high-pressure alarms for more than 5 times will produce high-pressure lockout |
| 24 | LP. lock | | 0x0617 | Read Only | x1 | NA |
| 25 | High exhaust Temp. lock | | 0x0618 | Read Only | x1 | NA |
| 26 | AC over voltage alarm | | 0x0619 | Read Only | x1 | The current detection voltage is higher than the AC voltage point (internal parameters are generally set to 256V) |
| 27 | AC under voltage alarm | | 0x061A | Read Only | x1 | The current detection voltage is lower than the AC voltage point (internal parameters are generally set to 187V) |

| | | | | | | |
|---|---|---|---|---|---|---|
| 28 | AC power supply failure | | 0x061B | Read Only | x1 | The alarm generated after the unit is powered off |
| 29 | Lose phase alarm | | 0x061C | Read Only | x1 | NA |
| 30 | Freq.fault | | 0x061D | Read Only | x1 | NA |
| 31 | Anti phase alarm | | 0x061E | Read Only | x1 | NA |
| 32 | DC over voltage alarm | | 0x061F | Read Only | x1 | NA |
| 33 | DC under voltage alarm | | 0x0620 | Read Only | x1 | NA |
| Parameter Setting | | | | | | |
| 1 | Refrigeration stop point | ℃ | 0x0700 | Read/Write | x1 | 15~50℃ |
| 2 | Refrigeration band | ℃ | 0x0701 | Read/Write | x1 | 1~10℃ |
| 3 | Heating stop point | ℃ | 0x0702 | Read/Write | x1 | -15~15℃ |
| 4 | Heating band | ℃ | 0x0703 | Read/Write | x1 | 1~10℃ |
| 5 | Reserve | | 0x0704 | | | |
| 6 | Reserve | | 0x0705 | | | |
| 7 | High Temp.point | ℃ | 0x0706 | Read/Write | x1 | 25~80℃ |
| 8 | Low Temp. point | ℃ | 0x0707 | Read/Write | x1 | -20~15℃ |
| 9 | High humidity point | % | 0x0708 | Read/Write | x1 | 0~100% |
| 10 | Internal fan stop point | ℃ | 0x070A | Read/Write | x1 | -20~50℃ |
| Remote Control Parameter | | | | | | |
| 1 | Reserve | | 0x0800 | | | |
| 2 | Remote control | | 0x0801 | Read/Write | | 1：Open,<br>0：Close |
| 3 | Baudrate | | 0x0766 | Read/Write | | 0：9600，<br>1：14400，<br>2：19200 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 4 | ID | | 0x075C | Read/Write | | 1~254 |

For example：

①Read the value of a single register-Software version： Send command：01 03 00 00 00 01 84 0A

Return instruction：01 03 02 02 10 B8 E8 Parsing instructions：01 ADDR
03 CMD
02 Length
02 10 Data
B8 E8 CRC

②Read the value of multiple registers-Coil temperature-outdoor temperature-Condensation temperature：

Send command：01 03 05 01 00 03 54 C7

Return instruction：01 03 06 01 1F 01 16 01 12 D4 C3 Parsing instructions：
01 ADDR
03 CMD
06 Length
01 1F Data:Coil temperature,Decimal 287→28.7℃
01 16 Data:outdoor temperature,Decimal 278→27.8℃
01 12 Data:Condensation temperature,Decimal 274→27.4℃ D4 C3 CRC

③Write Single Register-Refrigeration point 30℃： Send command：01 06 07 00 01 2C 88 F3

Return instruction：01 06 07 00 01 2C 88 F3 Parsing instructions：
01 ADDR
06 CMD
07 00 Register address
01 2C Data：Decimal 300→30℃ 88 F3 CRC

④write the value of multiple registers-High Temp set point40℃-Low Temp set point15℃ Send command：01 10 07 06 00 02 04 01 90 00 96 D4 0A

Return instruction：01 10 07 06 00 02 A0 BD Parsing instructions：
01 ADDR

10 CMD
07 06 Start register address
00 02 Register
number A0 BD CRC