

DS867 Communication Protocol Specification



Version History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version/Status | Author | Review | Start and End Date | Remarks |
| V1.0 | Lin Hui |  | 2021-03-20 |  |
| V1.01 | Lin Hui  Li Gang |  |  | Modify some modbus reply commands |
| V1.02 | Lin Hui |  | 2021-06-02 | Modify some modbus commands and delete 04 commands  Add card swiping serial number register 00 2F |
| V1.1 | Lin Hui |  | 2021-08-01 | 1. Added fingerprint data in 867 protocol  2. Separated 867 protocol from overall protocol |
| V1.2 | Full Text |  | 2021-08-20 | Improve some modbus commands |
| V1.3 | Lin Hui |  | 2021-09-29 | Added encryption and decryption instructions |
| V1.3 | Full Text |  | 2021-10-25 | Added clear all users command |

Contents

1. MODBUS protocol description 4

2. Basic frame format 4

1. Function code 4
2. Abnormal function code 5
3. Communication format description 5

3. Smart lock register 6

1. Control register 6
2. Card swipe register 8
3. Fingerprint register 9
4. Password register 10
5. Unlock record register 11

4. Example 12

1. Control register 12
   1. Get handle status 12
   2. Unlock 12
2. Lock 12
3. Modify station number 12
4. Get station number 12

6) Restore initial station number 13

1. Hardware version number register 13
2. Software version number register 13
3. Lock function register 13
4. User number setting register 13
5. User number query start register 13
6. Time register 14
7. Card swipe register 14
8. Write card number to device 14
9. Read card number in device 14
10. Fingerprint register 14
11. Host sends data to device 14
12. The host obtains data from the device 14
13. Password register 15
    1. Write the password to the device 15
    2. Read the device password 15
14. Unlock record register 15
    1. The host sends a request to the device for the latest unlock record data 15
    2. The host sends a request to the device for all unlock record data 15

V. Encryption and Decryption **Error! Bookmark not defined.**

**1. MODBUS Protocol Description**

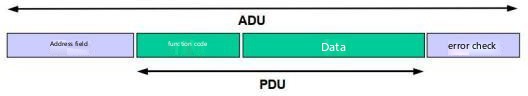
This product uses the standard Modbus-RTU communication protocol. If you need a custom protocol, please contact the product supplier. The following is a brief explanation and description of the Modbus protocol for the use of this product. The master station in the following content refers to the user's computer, single-chip microcomputer or PLC, etc., which can actively send data. The slave station in the following content refers to this product.

Figure 1 General MODBUS frame

Error check uses CRC-16 check mode

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mode | Polynomial | Initial value | Data sequence | Result processing |
| CRC16-MODBUS | X16+X15+X5+1(0x8005) | 0xFFFF | High bit first, low bit last | XOR with 0x0000 |

**2. Basic Frame Format**

Master station request format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Data length | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 2 bytes |

Slave station response format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Station number | Function code | Data length | Data content | CRC check |
| 1 byte | 1 byte | 1 byte | N bytes | 2 bytes |



|  |  |
| --- | --- |
| Function | Definition |
| 0x03 | Read register |
| 0x06 | Write single register |



The abnormal function code is based on the function code plus 0x80. For example, the abnormal code for the function code 0x03 is 0x83 abnormal function code, and the abnormal function code for the function code 0x06 is 0x86. The abnormal code here is unified as 0x01. That is, when the lock receives an abnormal instruction, the reply format is as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Station number | Abnormal function code | Abnormal code | CRC check |
| 1 byte | Function code + 0x80 | 0x01 | 2 bytes |



1) Read register

Master station request format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Number of registers | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 2 bytes |
| 0x01-0xFE | 0x03 | 0x0000-0xFFFF | 1-123 | CRC16 |

Slave station response format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Station number | Function code | Number of bytes | Data in register | CRC check |
| 1 byte | 1 byte | 1 byte | N\*2 bytes | 2 bytes |
| 0x01-0xFE | 0x03 | 2\*N |  | CRC16 |

Where N is the number of registers

2) Write a single register

Master station request format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Write data | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 2 bytes |
| 0x01-0xFE | 0x06 | 0x0000-0xFFFF | 0x0000-0xFFFF | CRC16 |

Slave station response format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Write data | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 2 bytes |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0x01-0xFE | 0x06 | 0x0000-0xFFFF | 0x0000-0xFFFF | CRC16 |

4) Write multiple registers

Master station request format:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Number of registers | Number of data | Data | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 1 byte | N\*2 bytes | 2 bytes |
| 0x01-0xFE | 0x10 | 0x0000-0xFFFF | 1-123 | 2\*N |  | CRC16 |

Where N is the number of registers

Slave station response format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Number of registers | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 2 bytes |
| 0x01-0xFE | 0x10 | 0x0000-0xFFFF | 1-123 | CRC16 |

**III. Intelligent Lock Register**



|  |  |  |  |
| --- | --- | --- | --- |
| Address | Register name | Parameter description | Read and write |
| 00 00 | Lock storage control register | Clear all unlock records: 00 01  Clear all card numbers in the lock: 00 02  Clear all passwords in the lock: 00 03  Clear all fingerprints in the lock: 00 04  Clear all users in the lock: 00 06 | W |
| Restore the initial station number in the lock: 00 05 (This instruction needs to use the broadcast address to operate) |
| 00 01 | Handle/door status register | Door open: 01 XX Door closed: 00 XX  Handle open: XX 01 Handle closed: XX 00 | R |
| 00 02 | Unlock register | Unlock: 00 01 | W |
| 00 03 | Station number register | Module station number: 1 -254, default: 00 01 | R/W |

|  |  |  |  |
| --- | --- | --- | --- |
| 00 82 | Hardware version number register | High byte as integer version number, low byte as decimal version number | R |
| 00 83 | Software version number register | High byte as integer version number, low byte as decimal version number | R |
| 00 84 | Lock function register | High byte reserved  Low byte: 0 bit: 485 function   1. Bit: card swipe function 2. Bit: Bluetooth function 3. Bit: switch function 4. Bit: fingerprint function 5. Bit: password function   Parameter bit 0 --- not available 1 Available | R |
| 00 10 | User number setting register | Device issued:  High byte: 01 - New user 02 - Delete user Low byte: User number (1 -100)  Device reply:  01- Success 02- User already exists 03 - Failure | W |
| 00 11 | Empty user number query start register | Reading this register can return an empty user number  name | R |
| 00 12 | Time register | This register is used to modify the lock's internal time device reply and write format according to: YY MM DD HH mm SS, BCD format | R/W |

Note:

 255 (0xFF) is the broadcast address. The station number register can be used to obtain the current station number and restore the initial station number of the lock. This command is only suitable for operating one lock;

 In the case of 485 bus, it is not allowed to configure it as active upload by swiping the card, otherwise it may cause errors;

|  |  |  |  |
| --- | --- | --- | --- |
| 00 05 | Card number data storage starting register |  | R/W |

1) Write the card number to the device

The host sends a normal card number data packet to the device in the following format:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Station number | Function  Code | Start  Address | Register  Quantity | Data  Number | Data | | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 1 byte | 1 byte | 1 byte | 4 bytes | 2 bytes |
| 0x01 | 0x10 | 0x00  0x05 | 0x00  0x03 | 0x06 | User number | Packet sequence number | IC card number | CRC16 |

Since only one packet of data is sent each time, the packet number is always 0x01. The data packet format of the device responding to the host is:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Number of registers | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | | 2 bytes |
| 0x01 | 0x10 | 0x00 0x05 | User number | Packet sequence number | CRC16 |

After the device succeeds, the user number is the user number sent by the host, and the packet number is 0x01

2) Read the card number in the device

The host reads the card number stored in the device

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Number of registers | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | | 2 bytes |
| 0x01 | 0x03 | 0x00 0x05 | User number | Packet sequence number | CRC16 |

The packet number is 0x01

If the host obtains the correct data packet frame format, the device replies to the host in the following format:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Station number | Function code | Number of bytes | Data in register | | | CRC check |
| 1 byte | 1 byte | 1 byte | N\*2 bytes | | | 2 bytes |
| 1 byte | 1 byte | 4 bytes |
| 0x01 | 0x03 | 0x06 | User number | Packet sequence number | IC card number | CRC16 |

|  |  |  |  |
| --- | --- | --- | --- |
| 00 90 | Fingerprint data storage start register | The register value does not need to be changed during packet processing | R/W |

Single fingerprint data is 2048 Bytes, so it needs to be packaged. Considering the data length of 485 transmission,

Here, 128 Bytes is 1 package, and there are 16 packages in total. Here is an explanation of the data domain in the protocol:

1. The host sends data to the device (write register)

The normal fingerprint data packet format sent by the host to the device is:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Station number | Function  Code | Start  Address | Register  Quantity | Data  Number | Data | | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 1 byte | 1 byte | 1 byte | 128 bytes | 2 bytes |
| 0x01 | 0x10 | 0x00  0x90 | 0x00  0x41 | 0x82 | User number | Packet sequence number | Fingerprint packet data | CRC16 |

The data packet format of the device responding to the host is:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Number of registers | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | | 2 bytes |
| 0x01 | 0x10 | 0x00 0x90 | User number | Packet sequence number | CRC16 |

1. The host obtains data from the device

The frame format of the fingerprint data packet sent by the host to the device

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Number of registers | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | | 2 bytes |
| 0x01 | 0x03 | 0x00 0x90 | User number | Packet sequence number | CRC16 |

If the host obtains the correct data packet frame format, the device replies to the host with the following frame format:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Station number | Function code | Bytes | Data in register | | | CRC check |
| 1 byte | 1 byte | 1 byte | N\*2 bytes | | | 2 bytes |
| 1 byte | 1 byte | 128 bytes |
| 0x01 | 0x03 | 0x82（130） | User number | Packet sequence number | Fingerprint packet data | CRC16 |

|  |  |  |  |
| --- | --- | --- | --- |
| 00 70 | Password storage starting register |  | R/W |

The maximum password is 6 digits, a total of 6 bytes, and the maximum value of each byte is 0x09. If it is greater than the password value, the default value is the maximum value 999999.

1) Write password to device

Host sends normal card number data packet to device in the format of:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Station number | Function  Code | Start  Address | Register  Quantity | Data  Number | Data | | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 1 byte | 1 byte | 1 byte | 6 bytes | 2 bytes |
| 0x01 | 0x10 | 0x00  0x70 | 0x00  0x04 | 0x08 | User number | Packet sequence number | Password | CRC16 |

Since only one packet of data is sent each time, the packet number is always 0x01. The data packet format of the device responding to the host is:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Number of registers | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | | 2 bytes |
| 0x01 | 0x10 | 0x00 0x70 | User number | Packet sequence number | CRC16 |

After the device succeeds, the user number is the user number sent by the host, and the packet number is 0x01

2) Read the password in the device

Host reads password stored in device

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Station number | Function code | Start address | Number of registers | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | | 2 bytes |
| 0x01 | 0x03 | 0x00 0x70 | User number | Packet sequence number | CRC16 |

The packet number is 0x01

If host obtains data packet frame format correctly, device replies to host in the format of:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Station number | Function code | Number of bytes | Data in register | | | CRC check |
| 1 byte | 1 byte | 1 byte | N\*2 bytes | | | 2 bytes |
| 1 byte | 1 byte | 6 bytes |
| 0x01 | 0x03 | 0x08 | User number | Packet sequence number | Password | CRC16 |

After the password is uploaded here, the host computer considers doing \* processing



|  |  |  |  |
| --- | --- | --- | --- |
| 00 60 | Unlock record storage starting register |  | R |
| 00 61 | The latest unlock record register |  | R |

 Can store up to 20 unlock records

1. The host sends a request to the device for the latest unlock record data packet frame format

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Station number | Function code | Starting address | Number of registers | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | | 2 bytes |
| 0x01 | 0x03 | 0x00 0x61 | User number | Packet sequence number | CRC16 |

The user number here is filled with 0x00, and the package number is 0x01

If the host obtains the data packet frame format correctly, the device replies to the host with the packet frame format:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Station number | Function code | Number of bytes | Data in register | | | CRC check |
| 1 byte | 1 byte | 1 byte | N\*2 bytes | | | 2 bytes |
| 1 byte | 1 byte | 8 bytes |
| 0x01 | 0x03 | 0x0A（10） | User number | Packet sequence number | Unlock record | CRC16 |

The user number in the reply data here is 0x00, and the package number is 0x01. The most recent unlocking record returned here

Unlock record data is: unlock user number (1 byte) + unlock time (6 bytes, YY MM DD HH mm SS) + unlock method (1 byte, 01-remote 485, 02-card, 03-fingerprint, 04-password, 05-Bluetooth, 06-switch, 07-abnormal)

1. The host sends a request to the device for all unlock records data packet frame format

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Station number | Function code | Starting address | Number of registers | | CRC check |
| 1 byte | 1 byte | 2 bytes | 2 bytes | | 2 bytes |
| 0x01 | 0x03 | 0x00 0x60 | User number | Packet sequence number | CRC16 |

The user number here is filled with 0x00, and the package number is 0x01

If the host obtains the data packet frame format correctly, the device replies to the host with the packet frame format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Station number | Function code | Number of bytes | Data in register | CRC check |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 byte | 1 byte | 1 byte | N\*2 bytes | | | 2 bytes |
| 1 byte | 1 byte | 160 bytes |
| 0x01 | 0x03 | 0xA2（162） | User number | Packet sequence number | Unlock record | CRC16 |

The user number in the reply data here is 0x00, and the package number is 0x01

**IV. Example**



1. Get handle/door status

Host request 01 03 00 01 00 01 D5 CA

Slave reply 01 03 02 01 01 78 14 Handle/door status open

01 03 02 01 00 B9 D4 Door status open/handle closed

01 03 02 00 01 79 84 Door status closed/handle open

01 03 02 00 00 B8 44 Handle/door status closed

1. Unlock

Host request 01 06 00 02 00 01 E9 CA

Slave reply 01 06 00 02 00 01 E9 CA

1. Lock

Host request 01 06 00 02 00 00 28 0A

Slave reply 01 06 00 02 00 00 28 0A

1. Modify station number

Host request 01 06 00 03 00 10 78 06

Slave reply 01 06 00 03 00 10 78 06

Modify station number from 01 to 10 After modifying the station number, communicate according to the new station number. 5) Get station number

Host request FF 03 00 03 00 01 61 D4

Slave reply FF 03 02 00 10 90 5C

6) Restore initial station number

Master request FF 06 00 00 00 05 5C 17

Slave reply FF 06 00 00 00 05 5C 17

After recovery, the initial station number is 00 01

7) Hardware version number register

Host request 01 03 00 82 00 01 24 22

Slave reply 01 03 02 01 01 78 14

1. Software version number register

Master request 01 03 00 83 00 01 75 E2

Slave reply 01 03 02 01 01 78 14

1. Lock function register

Master request 01 03 00 84 00 01 C4 23

Slave reply 01 03 02 00 1F F9 8C

1. User number setting register

Host request  
Slave reply

Host request  
Slave reply

01 06 00 10 01 01 48 5F New user

01 06 00 10 01 01 48 5F New success

01 06 00 10 01 02 08 5E User saved

01 06 00 10 01 03 C9 9E Failure

01 06 00 10 02 01 48 AF Delete user

01 06 00 10 02 01 48 AF Delete success

01 06 00 10 02 03 C9 6E Failure

1. User number query start register

Master request 01 03 00 11 00 32 95 C8

Slave reply 01 03 64 xx xx xx xx ………xx xx xx xx CRC16

The host sends data starting from register 0x0011 and reads 50 registers, that is, 100 user numbers. In the slave reply data, each byte is a user, 100 user numbers, a total of 100 bytes,

Data ff means user does not exist, 01-100 Indicates the user number exists.

Host request 01 03 00 12 00 03 95 C8 Read time

Slave reply 01 03 06 YY MM DD HH mm SS CRC16

Host request 01 10 00 12 00 03 06 YY MM DD HH mm SS CRC16 Write time

Slave reply 01 10 00 12 00 03 20 0D



1. Write card number to device

Host request 01 10 00 05 00 03 06 01 01 ID1 ID2 ID3 ID4 CRC16

Slave reply 01 10 00 05 01 01 CRC16

If the user number written by the slave does not exist, an error frame is returned.

1. Read the card number in the device

Host request 01 03 00 05 01 01 CRC16

Slave reply 01 03 06 01 01 ID1 ID2 ID3 ID4 CRC16

If the user number read from the slave does not exist, an error frame is returned.

If the slave machine replies with FF FF FF FF FF FF, it means no password is set.



1. The host sends data to the device

Host request 01 10 00 90 00 41 82 01 01 xx xx xx xx ………xx xx xx xx CRC16

Slave reply 01 10 00 90 01 01 CRC16

If the user number written by the slave does not exist, an error frame is returned.

The host sends data starting from register 0x0090 and writes 65 registers, a total of 130 bytes of data, the data user number remains unchanged, the packet number increases, and the slave replies with the user number and packet number.

1. The host obtains data from the device

Host request 01 03 00 90 01 01 85 B7

Slave reply 01 03 82 01 01 xx xx xx xx ………xx xx xx xx CRC16

Data user number remains unchanged, packet number increases, slave replies 130 bytes of data, user number, packet number, fingerprint data.



61) Write password to device

Host request 01 10 00 70 00 04 08 01 01 \*\* \*\* \*\* \*\* \*\* \*\* CRC16

Slave reply 01 10 00 70 01 01 01 82

If the user number written by the slave does not exist, return an error frame.

The host sends data, the data user number, the packet number is always 0x01, the password is 6 bytes, and the maximum value of each byte is 0x09.

2) Read device password

Host request 01 03 00 70 01 01 84 41

Slave reply 01 03 08 01 01 \*\* \*\* \*\* \*\* \*\* \*\* CRC16

The slave replies data, the data user number, the packet number is always 0x01, the password is 6 bytes, and the maximum value of each byte is 0x09. If it is FF FF FF FF FF FF, the password is not set.



1. The Host Sends a Request to the Device for the Latest Unlock Record Data

Host request 01 03 00 61 00 01 D5 D4

Slave reply 01 03 0A 00 01 XX YY MM DD HH mm SS XX CRC16

The slave machine replies with unlock record data: unlock user number (1 byte) + unlock time (6 bytes, YY MM DD HH mm SS)

+ unlock mode (1 byte, including 01-remote 485, 02-swipe card, 03-Bluetooth, 04-switch, 05-fingerprint, 06-password, 07-abnormal), a total of 8 bytes.

1. The Host Sends a Request to the Device for All Unlock Record Data

Host request 01 03 00 60 00 01 84 14

Slave reply 01 03 A2 00 01 xx xx xx xx ………xx xx xx xx CRC16

The slave replies 20 unlock record data, a total of 160 bytes. If there are less than 20, the remaining reply data is 00.