

RS485 TO ETH (B)

RS485 TO POE ETH (B)

RS232/485/422 TO POE ETH (B)

RS232/485/422 TO ETH (B)

Serial port modification parameters and hardware TCPIP co-stack

Embedded Device Networking Solutions

Directory

[1. overview 5](#_bookmark1)

[1 .1. hardware protocol stack and software protocol stack 5](#_bookmark2)

[1. 2. hardware protocol stack principle 5](#_bookmark3)

[2. command mode 6](#_bookmark4)

[2.1. Command Format 6](#_bookmark5)

[2 .2. Second Generation Command Format 7](#_bookmark6)

[2.3. parameter format 1 0](#_bookmark7)

[2 .4. Read Parameters Steps 14](#_bookmark8)

[2 .5. Write Parameters Step 16](#_bookmark9)

[2.6. Internal Register Area 16](#_bookmark10)

[2.7. considerations 1 7](#_bookmark11)

[3. Typical Application 17](#_bookmark12)

[3.1. Read Connection Status 17](#_bookmark13)

[3 .2. Read Multiple Connection Status 18](#_bookmark14)

[3.3. Control Networking Product Connection 19](#_bookmark15)

[3.4. Reading Local IP Address 1 9](#_bookmark16)

[3.5. modify dns server 1 9](#_bookmark17)

[3.6. to see if the system is initialized 20](#_bookmark18)

[3.7. one-time setting method 20](#_bookmark19).

[3.8. serial port restart device 21](#_bookmark20)

[3 .9. Specify DNS Server 21](#_bookmark21)

[3 .10. Read Device Name 22](#_bookmark22)

[3 .11. Write Device Name 22](#_bookmark23)

[3.12. reading device ID 22](#_bookmark24)

[3 .13. Reading Device Password 22](#_bookmark25)

[3.14. Modify Destination IP 22](#_bookmark26)

[3.15. Modify Destination Port 22](#_bookmark27) UDP Mode

[3. 16. reading device version number 23](#_bookmark28)

[3.17. 2003 Low Power Mode 2 3](#_bookmark29)

[4. variable parameter area 2 3](#_bookmark30)

[4. 1. for use of the variable parameter zone 23](#_bookmark31)

[4. 2. WIFI Parameter Zone Use 27](#_bookmark32)

[4. 3. user parameter space using 30](#_bookmark33)

[4 .4. Multi-Target Setup MDIP Devices 30](#_bookmark34)

[4.5. Custom Parameters with Interface 31](#_bookmark35)

[4.6. Serial port registration packet and heartbeat packet write 3 1](#_bookmark36)

[5. after-sales service and technical support 32](#_bookmark37)

1. Overview

1.1. Hardware protocol stack and software protocol stack

The TCP/IP hardware stack is relative to the TCP/IP software stack. TCP/IP software generally provides socket interface, by calling library functions to connect, monitor, send, receive and other operations, such as Windows socket API functions have connect, listen, send, recv, etc. Hardware protocol stack is a 1 relatively new concept. Hardware protocol stack exists in RS485 TO ETH (B) networking module. User MCU sends commands to RS485 TO ETH (B) through serial port and controls RS485 TO ETH (B) to run the TCP/IP protocol stack to achieve network functions such as connection, monitoring, sending and receiving. Its function is similar to directly calling software TCP/IP protocol stack.

Hardware embedded systems using TCP/IP protocol stack advantages:

1. With the use of software TCP/IP protocol stack compared to the hardware protocol stack does not occupy the user's CPU, no RAM, reducing the user's burden on the MCU, and the hardware protocol stack is a mature product, with strong stability.

2. With TCP/IP protocol stack without hardware networking module, hardware TCP/IP MCU makes the user more flexible, networking module can be secondary development, basically can achieve all the control functions of the software protocol stack.

1.2. Principle of hardware protocol stack

RS485 TO ETH (B), the hardware protocol stack is actually achieved by modifying the RS485 TO ETH (B) internal parameters, for example, the user modified the destination domain name or the IP RS485 TO ETH (B) automatically to the new destination to initiate a connection, in order to connect() function.

Therefore, the function of the hardware protocol stack actually also provides a method for modifying a module parameter on the user equipment side. This provides a method for the user to modify the IP, mode and other parameters of the device through the keyboard, touch screen and other methods.

RS485 TO ETH (B) provides two modes to modify device parameters: command mode and hardware mode.

Command mode: the user through the serial port to the RS485 TO ETH (B) to send a fixed command identification stream, let the RS485 TO ETH (B) parameter modification.

Hardware mode: the user's MCU IO control pin and the RS485 TO ETH (B) SPR, SPA connection, through the hardware pin between the timing control to achieve parameter modification purposes.

Comparison of the two models:

No. 1. Command mode does not need to increase the hardware connection, the hardware impact is small, especially through the DB9 serial port line and equipment connection, increase the hardware connection is more difficult.

2. The control sequence of the hardware mode is more complicated, and the user development takes a long time. The command mode operation is relatively simple, and the command word can be simulated and sent through the computer serial port debugging tool.

3. The only advantage of the hardware mode is that there is no concern of command recognition stream and data stream interference, I .e. the presence of command recognition stream in the data stream transmitted by the user. But... RS485 TO ETH (B) the command recognition stream has10 bytes, total1.2 × 1024 possibly, I .e.1.2 billions of billions of possibilities, if the highest115200bps of baud rate constantly sending data, then on average, the need it only appears once in 36604 billion years. This possibility can be ignored.

Suggestion: Advise users to use command mode.

Command mode

Command Format 2.1.

The steps for reading and writing parameters in command mode are as follows: at any time, write to the RS485 TO ETH (B) serial port

1 command to complete parameter reading and writing.



1 Command format

The command identification stream is 10 bytes, which indicates the start of the command and must be the following data: ed f2 a3 56 ca db 91 84 b0 d7

The command type is 1 bytes, indicating the type of the command. The 0 bit is a 1 table for the write parameter, and a 0 table for the read parameter.

Table 1 Common command types

|  |  |  |
| --- | --- | --- |
| CommandWord | often makeUse | Function |
| 0x00 | Yes | Read parameters.  |
| 0x01 | Yes | Write parameters. Power off does not save. But IP, subnet mask, gateway, DHCP mode, DNSAfter the IP address is modified, the server restarts and saves the parameters.  |

2 Special command classes

|  |  |  |
| --- | --- | --- |
| CommandWord | often makeUse | Function |

|  |  |  |
| --- | --- | --- |
| 0x02 | No | For MDIP models: Initiate a second TCP connection without closing the previous TCP connection.  |
| 0x03 | No | Write parameters. And save the parameters, write to the RS485 TO ETH (B) Flash memory to save.  |
| 0x04 | No | The dongle command.  |
| 0x05 | No | Write parameters. The parameters must be saved and the module must be restarted without closing the previous TCP connection.  |
| 0x06 | No | Read web pages (ZLSN2030EX only).  |
| 0x07 | No | Same as the 0x 03 command, but must restart the module.  |
| 0x08 | No | Write web pages (ZLSN2030EX only).  |
| 0x09 | No | For parameter-based communication, refer to the function of sending module parameters to the central server. pdf. |
| 0x0a | No | control Uniform control triggers (ZLSN2030EX -specific).  |
| 0x0b | No | For parameter-based communication for saving parameters, refer to the function of sending module parameters to the central server. pdf. |
| 0x0c | No | control TXET controls submit data (the ZLSN2030EX is private).  |
| 0x0d | No | ZLAN7142 specific configuration instructions.  |
| 0x12 | No | Similar to the 0x 00 read instruction, but this read place is not the parameter area, but the internal registerarea. The length of each register is not fixed, but the number of registers is only 256.  |
| 0x13 | No | Similar to the 0x 01 write instruction, but this write place is not the parameter area, but the internal registerarea. The length of each register is not fixed, but the number of registers is only 256.  |

The read/write position is 1 bytes, which indicates the byte from which the parameter is read. Refer to Figure 4 for the parameter format.

The number of bytes is 1 byte, indicating the need to read or write data.

Content, in the write operation, the internal part of the content to be written, its length should be consistent with the number of bytes field definition.

2.2. The second-generation command format.

In order to ensure the correctness of the serial port command sent, the second-generation command format can now be supported as follows:



2 Second-generation command format

As shown in the figure: the difference between the second generation command and the first generation command is that

1. Change the command type to a second-generation command. The second generation command is the first generation command plus 0x 80. For example, the original 0x 01 for the write command is now 0 x 81.

If it is a second-generation command, it must be at the end of the increase of CRC16 2 bytes of the check. The is verified without command identification stream (excluding the CRC itself), as shown in the figure. If the a CRC error then the command will not be executed. This ensures that accidental writes of data will not be performed.

3. The second-generation command will have feedback. If the execution is successful, it will return the correct instruction, and if it is wrong, it will return the wrong instruction.



3 Return instruction format

It can be seen that the format of the feedback command is the same as that of the written command, which can facilitate the user to calculate the CRC and identify the command. There are 4 kinds of "return commands":( 1) 0x0e indicates instruction format error, including CRC error. At this time, the read and write position is 0, the number of bytes is 0, the content is empty, and CRC is 10 03. (2) 0x 10 indicates that the instruction format and CRC are correct, but the numerical range may be out of bounds or the written MAC address does not match the original MAC address (MAC cannot be modified). Note that this return error only exists in 0x 03 and other instructions that need to write flash. 0x 01 instructions will not return this error. (3) 0x0f indicates that the instruction is correctly executed. At this time, the read and write position is 0, the number of bytes is 0, the content is empty, and CRC is 41c3. (4) 0 x 80 represents the return value data of the second generation read instruction. At this time, the read and write position and byte number are the same as the read and write position and byte number of the sent instruction. The content is the content of the read data. CRC is the checksum of the illustrated instruction part.

Any module that supports second-generation commands also supports first-generation commands. It is recommended to support the adoption of the second generation of life.

Order.

The 1.558(2003) version of the firmware was tested as follows:

For example, the original command to read the connection status is: ed f2 a3 56 ca db 91 84 b0 d7 00 3d 01, and its return value is only 00 or 01, which is easy to confuse with communication data. When changing to a second-generation command, the first 00 to change for80, then add "80 3d 01 " CRC the check is a1 78. So the whole command is ed f2 a3 56 ca db 91 84 b0 d7 80 3d 01 a1 78. The return command received after sending is: ed f2 a3 56 ca db 91 84 b0 d7 80 3d 01 00 b9 b8.. Here ed f2 a3 56 ca db 91 84 b0 d7 is the command identification stream, 80 is

The read instruction returns the instruction, 3d is the sending position, 01 indicates that the following data content is 1 byte, followed by the 00 of one byte of content, followed by the content CRC check b9 b8 of 80 3d 01 00. If the indication sent makes a CRC error, it will receive ed f2 a3 56 ca db 91 84 b0 d7 0e 00 00 10 03.

For example, the original command to modify the mode to the client is ed f2 a3 56 ca db 91 84 b0 d7 01 14 01 01. The second generation command is ed f2 a3 56 ca db 91 84 b0 d7 81 14 01 01 a8 4c. If not returned, the baud rate is wrong and the module is not connected. If returned ed f2 a3 56 ca db 91 84 b0 d7 0f00 00 41 c3 indicates that the command was executed correctly; if ed f2 a3 56 ca db 91 84 b0 d7 0e00 00 10 03 indicates a command parsing error (including CRC error).

There is also a 1 possibility that the written value range is out of bounds or an attempt is made to modify the MAC address, for example, ed f2 a3 56 ca db 91 84 b0 d7 83 14 01 04 69 f7 is sent. Note here that 0x 83 is the write flash instruction instead of the 0x 81 memory write instruction. In addition, the working mode is set to 4. In fact, the range of the mode is 0 to 3,4, which is illegal. The return instruction for this is ed f2 a3 56 ca db 91 84 b0 d7 10 00 00 70 05. 0x 10 here indicates that the data content is out of bounds, causing the flash to fail to write. In addition, note that if the 0x 81 instruction was used to write data to memory and the 0x 83 instruction was used to write data to flash, as long as any previous instruction is out of bounds, the 0x 83 instruction will return a 10 error code.

In addition, for example, the instruction ed f2 a3 56 ca db 91 84 b0 d7 83 1f 01 ff59 b6, where 1f starts at the mac address and cannot be modified, where the user tries to write flash to modify it. It will also return the 10 instruction content can not be written error.

Instruction return value has two conditions: (1) There must be a write -flash action. If the 0x 83 instruction is not used, the 0x 81 instruction, the actual out-of-bounds data will be written to memory, but because it is not written to flash, it will not be used it is saved by the system, so it is also invalid. 2 () The written content cannot be correctly written to the flash memory.

Note that whether it is a write or read instruction, it is recommended to have a 100ms interval between adjacent instructions.

Parameter Format 2.3.



4 Parameter format

The module parameter format is shown in FIG. 4. When sending and receiving parameters, the first byte is sent first. Above

The bytes in are in big-endian mode, for example, the high byte of Local IP Addr is on the left. For the meaning of each parameter, refer to the RS485 TO ETH (B) Data Book.

1. Local IP Addr (Local IP Address): 4 bytes.

2. Net Mask: 4 bytes.

3. GateWay (Gateway): 3 bytes.

4. Dest IP (Destination IP ): 4 bytes. For networking products with DNS function, the role of this field is replaced by DestString field, which is invalid; For networking products without DNS function, this field makes the 4- byte destination IP address used. Whether your product has the DNS function, please consult Zhuo Lan company.

5. Local IP Port (Local Port): 2 bytes.

6. Dest Port (destination port): 2 bytes.

7. Work mode (work mode): byte 1. The values of 0, 1, 2, and 3 correspond to the following values: server mode (TCP Server ), client mode (TCP Client ), UDP mode, UDP multicast mode.

8. Pad (padding area): 10 bytes, should be all 0.

9. Pad2 (padding area 2, also known as DevID ): 6 bytes. When the write parameter contains this field, it should fill in the correct ID of the device here, otherwise if the write ID is incorrect, the device will discard all other write parameters.

10. Baundrate (baud rate): byte 1. From 0 to 13 (13 means 460800) correspond to:, 1200,

2400, 4800, 7200, 9600, 14400, 19200, 28800, 38400, 57600, 76800,

115200, 230400, 460800.

11. Device Name: Bytes 10. Must be a 0-terminated visible string.

12. Parity: 1 byte. 4 0 to 4 respectively: None, Odd, Even, Mark, Space five ways.

13. Gap Time (interval time): 1 byte.

14. Packing length (packet length): 2 bytes, numerical range ~ 1 1400.

15. F\_end en (end-of-frame character valid bit): byte 1, 0 and 1 respectively indicate that the end-of-frame rule does not work and works. Version V1.472 starts this byte is no longer valid.

16. F\_end byte (half\_485\_wait): 1 byte. 485 the half-duplex time interval.

17. F\_start en ( half\_485\_max\_wait ):1 byte, 485 half-duplex maximum waiting time, general for3ms. 0 cancels the 485 half-duplex.

18. Byte F\_start (frame first character): 1 byte. Version V1.472 starts this byte is no longer valid.

19. DHCP en (DHCP valid bits): 1 byte. 0 and 1, respectively, indicate the use of static IP and DHCP, respectively

Get the IP.

20. Flow Control (flow control method): 1 means CTS and RTS flow control is adopted; 0 means no flow control.

21. Dest\_Mode (destination mode):1 byte. 0 and 1 represent static mode and dynamic mode, respectively.

22. DataSize number (serial port number): 8 ~ 5bit, corresponding to 0, 1, 2, 3.

AppPro (conversion protocol): 0 indicates transparent transmission, 1 indicates Modbus TCP and Modbus RTU conversion, 2 indicates RealCom.

24. Status (modification mode): When reading this parameter, bit0 = 1 of Status indicates that the current TCP connection has been established or is in UDP state, otherwise bit0 = 0. This field provides a way to read the current state of the networking module.

25. DnsServerIP (DNS Server IP ): Set to DNS Server IP ,4 bytes.

26. Dest string (destination address): has DNS of the networking product, this field indicates the purpose IP string, such as writing"192.168.0.3 "This string is hexadecimal: 0x 31, 0x 39, 0x 32, 0x2e, 0x 31, 0x 36, 0x 38, 0x2e, 0x 30, 0x2e, 0x 33, 0x 00. Note that a 0 is added at the end of the string. Networked products without the DNS function This field is invalid.

27. Recon\_Time time (disconnection and reconnection time): 1 byte, range from 0 to 255.

28. Keep\_Alive time (keep-alive timing): 1 byte, range from 0 to 255.

29. Web\_port (web page access port):2 bytes, the port for accessing web pages through the browser.

30. UDPF\_pos, UDPF\_code and UDP\_mask (UDP application layer filtering parameters): a total of 3 bytes, which are generally set to 0. The parameter is currently invalid.

31. ver (module version): 1 byte, cannot be modified. ver = 0 means that the version number is 1.383. The actual version number is 383 + ver, for example, ver = 117, which means the version is 1.500.

32. func\_sel (module supported function): 1 byte, cannot be modified. The specific meaning of each bit is as follows: bit0 = 1 indicates support for web page download; bit1 = 1 indicates support for DNS domain name system; bit2 = 1 indicates support for REAL\_COM protocol; bit3 = 1 indicates support for Modbus TCP to RTU;bit4 = 1 indicates support for serial port modification parameters; bit5 = 1 indicates that it supports automatic acquisition of IP (DHCP); bit6 = 1 indicates that it supports storage and extension of EX functions; bit7 = 1 indicates that it supports multi -TCP connections.

33. Group\_IP (UDP Multicast Address): 4 bytes, range from 224.0.0.0 to 239.255.255.255. The position in the whole package is as follows.

zl\_u8; zl\_s8; zl\_u16; zl\_s16; zl\_u32;

bit zl\_bool;

signed long zl\_s32;

zl\_u32 IP\_ADDR;

(

34. io\_set (IO port settings): IO port control word, please refer to IO control related documents.

35. func\_en (function selection): function selection to enable/control word. bit0 = 1 turns on the "data restart function", bit1 = 1 turns on the "function of sending module parameters to the central server". bit2 = 1 to open the "modify parameters need password function. bit3 = 1 to turn on the "UDP disable receiving broadcast packet function". Please refer to the relevant documentation. bit4 = 1 turns on the "P2P function".

36. sm\_param\_t (sm parameter time): to the central server to send module parameter function sending interval, the unit is minutes.

37. func\_sel2 (module supported function 2 ): The module supports advanced functions. bit0 = 1 is used to support the IO configuration function. bit1 = 1 UDP multicast function is supported. bit2 = 1 is used to support the multi- target IP function.

38. Reserved 1 byte (QueryOneAckOneMaxWait ): multi-host. 0 is to cancel the multi-host function, otherwise it is the most common wait timeout divided by the 32.

39. 1 byte reserved

40.52 bytes are reserved. The 54 reserved bytes are reserved for future upgrades. The total number of bytes of the parameter is 167 bytes (for UDP management port protocol, the header is marked with 2 command type, with a total length of 170 bytes).

The C description of the parameter format i

#define DNS\_NAME\_MAX\_LEN 30

struct SSServerParam

{

IP\_ADDR param\_local\_ip;

IP\_ADDR param\_net\_mask;

IP\_ADDR param\_gate\_way;

IP\_ADDR param\_dest\_ip; zl\_u16 param\_local\_port; zl\_u16 param\_dest\_port; zl\_u8 param\_work\_mode; zl\_s8 param\_key[MAX\_KEY\_LEN]; zl\_u8 ether\_addr[ETHER\_ADDR\_LEN]; zl\_u8 baundrate\_index; zl\_s8 dev\_name[MAX\_DEV\_NAME\_LEN]; zl\_u8 param\_parity; zl\_u8 param\_max\_no\_s\_data\_interval; zl\_u8 param\_fram\_end\_en; zl\_lu8 param\_ip\_mode; zl\_lu8 param\_frame\_start\_byte; z\_param\_flow\_control; z\_lu\_lu8; zl\_lu8; zl\_lu\_lu8; zl\_lu;

zl\_u8 status;

IP\_ADDR dns\_server\_ip; zl\_s8 dns\_name[DNS\_NAME\_MAX\_LEN]; zl\_u8 keep\_alive\_time; zl\_u8 reconnect\_time;

zl\_u16 web\_port;

zl\_u8 udp\_filter\_pos, udp\_filter\_code, udp\_filter\_mask; zl\_u8 ver;

zl\_u8 func\_sel;

IP\_ADDR udp\_group\_ip;

zl\_u8 io\_set;

zl\_u8 func\_en;

zl\_u8 server\_mode\_param\_t;

zl\_u8 func\_sel2;

zl\_u8 var1[2];

zl\_u8 var2[52];

};

Read parameter steps 2.4.

The commands shown in FIG. 1 are sent to the RS485 TO ETH (B) most quickly and continuously. That is, the pause between each byte should not be longer than the Gap Time. Please refer to the Gap Time in the RS485 TO ETH (B) manual.

for 115200bps, Gap Time is 3 ms by default. The baud rate sent must be the current model rate of the module. After sending the command, the parameters of the specified content can be received at the serial port.

Taking reading the destination domain name or IP string as an example, the range of Dest String is 42H to 60H in Figure 4, and the number of bytes is 1EH. Then the command should be: ed f2 a3 56 ca db 91 84 b0 d7 00 42 1E. Where 00 indicates the read command, does not write to Flash ,42 is the parameter start position, and 1E is the read byte number.

The result of sending the above command in 16 ZLComDebug is shown in fig. 5. it can be seen that the string received the destination IP string.



5 Read parameter demonstration

Tips: When reading parameters in communication, the returned data may be in communication data conflict. For example, the query connection status is: ed f2 a3 56 ca db 91 84 b0 d7 00 3d 01, and the return of 00 indicates that 01 is not connected, indicating that it is connected. But this may conflict with the communication data. For this reason, the query can be changed to: ed f2 a3 56 ca db 91 84 b0 d7 00 3d 05, thus the return is the 00 08 08 04 04, as shown in fig. 8 4. the following 08 08

04 04 is DNS, which is generally a fixed DNS 08080404, so this is equivalent to an iconic suffix, which provides a distinction between communication data and query data.

Write parameter steps 2.5.

The steps for writing parameters are basically the same as the steps for reading parameters, as long as the command type is changed from 00 to 01 or

03 Flash (need to write when 03) can be.

For example, the target domain name is changed [to www. waveshare. com](http://www.zlmcu.com) the command not to save to Flash is: ed f2 a3 56 ca db 91 84 b0 d7 01 42 1e 77 77 77 2e 7a 6c 6d 63 75 2e 63 6f 6d 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00

01 is the command type, which means to write the parameter, but does not need to be saved to Flash ,42 is the start position of the Dest String parameter, 1e is the size of the Dest String parameter, followed by the hexadecimal table of the target domain name, and the length is 1e.

Write operation will first close the current network connection, and then establish the network connection after reading and writing. If the IP address, subnet mask, gateway, IP mode, and DNS server are modified, the system will automatically restart after these parameters are modified.

Note: During the initialization process of the system, especially when DHCP = 1 mode, the startup time will be compared

Slow, the parameters written at this time cannot be received by the module. At this time, ed f2 a3 56 ca db 91 84 b0 d7 00 3d 01 needs to be sent first. if there is a return 00 or 01, the parameters can be written. If no data is returned, please continue to wait before querying whether the initialization is complete.

Internal register area 2.6.

The above reading and writing are all based on the internal parameter area. Now, a 1 reading and writing function of the internal register area is defined. Different positions of the internal register may represent different meanings.

Table 3 Internal deposit

|  |  |  |  |
| --- | --- | --- | --- |
| register address | Length | Meaning | Examples |
| 0 | / | Retention |  |
| 1 | 1 | 0x 01: Low Power On0x 00: Turn off low power consumption | ed f2 a3 56 ca db 91 84 b0 d7 93 01 01 01 low power on bc f0 off low power ed f2 a3 56 ca db 91 84 b0 |

(

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | d7 93 01 01 00 7d 30 |
| 2 | 10 | Read the NTP network time.  | ed f2 a3 56 ca db 91 84 b0 d7 92 02 0a 514a |

2.7. Precautions

No. 1. The user ZLVircom the program to view the parameters to check whether the parameters have been successfully modified.

2. There are reasons why waiting to read parameters takes 2- 3 seconds. This is because you put the module as the TCP client, in TCP client mode, if the connection is not established (the LINK light is not on), then sometimes there will be a delay in reading and writing parameters. The reason is that the external command response is suspended while the TCP client is connecting. This problem does not occur if you are in TCP server or UDP mode, or if the connection is established.

3. Intermittant condition when reading parameters. When using the parameter read command to read parameters, the module will continuously output the parameter content, but the entire output content may be interrupted for a few milliseconds in the middle. This is because the output of parameters and the output of network port to serial port data use different mechanisms, and the latter will not be intermittent.

4 . For a new version module, if the parameter written once contains the device ID, then must the same as the device must be written ID, otherwise all written parameters will be discarded. If you do not know to read the ID first, fill in the ID area.

Typical applications

Read the connection status. 3.1.

Send the following command to the serial port of the module

ed f2 a3 56 ca db 91 84 b0 d7 00 3d 01

The module will return a one-byte status word. If the status word 0x 00 indicates that it is not connected (please use hex mode to display this data in the serial debugging assistant). If 0 x 01, it is connected. The so-called connection means that the TCP connection has been established or is in UDP mode, and the connection is equivalent to the LINK (generally green) light (LINK pin level is 0 ). Note that all UDP mode queries are 1.

If the module is in a busy working state, it may not respond to the above command for a while

Simply ignored. So the user's receiving program needs to have a wait timeout mechanism. busy working state

Includes:

1. The module is moving to the destination IP during the connection, if tcp connection, this connection lasts the longest possible time is five seconds.

2. The module is initializing the boot, according to whether to use the DHCP IP mode, boot time varies.

The serial port commands sent above will not be received by the other end (e. g. PC) connected to the Zhuo Lan module, which ensures that sending the above commands will not affect the communication protocol.

The average time interval from sending the above command to receiving the status word is 7ms or so, so waiting for a timeout of 10ms should be possible.

For the networking module located at the TCP client, the user can send the above command every 100ms (selected according to the user's needs and as large as possible if allowed) to detect whether the connection is established. The user protocol data may be sent once the connection is established.

The query can be changed to: ed f2 a3 56 ca db 91 84 b0 d7 00 3d 05 , so return is00 08 08 04 04, by Fig.4 know the back.08 08 04 04 yes DNS, this DNS is generally fixed of08080404, this is equivalent to a symbolic suffix, which provides users with a distinction between communication data and query data.

Read multi-connection status. 3.2.

Assume that four destination IP addresses and ports are set as follows:

Main interface:



More advanced options



They are referred to as connection 1, connection 2, connection 3, and connection 4, respectively, in that order. Now you need to query these connections

The state.

Method 1: No CRC check method, send hexadecimal ed f2 a3 56 ca db 91 84 b0 d7 00 3d

X, returns a byte, 01. If bit1 (rightmost bit of X) indicates the state of connection 1, 1 indicates that it is connected, and 0 indicates that it is not connected. Bit2 indicates the second connection state; And so on. For example return0x0b then indicates a connection1 connections2 connections4 has been established.

Method 2: There is a CRC check method, sending ed f2 a3 56 ca db 91 84 b0 d7 80 3d 01 a1 78 in hexadecimal. The last 2 bits are CRC16, and ed f2 a3 56 ca db 91 84 b0 d7 80 3d 01 0b f8 7f is returned, where ed f2 a3 56 ca db 91 84 b0 d7 is the identification code, 0b is the required connection state, and f8 7f is the CRC check of 80 3d 01 0b. This method is more complicated than the 1 parsing method, but there is a CRC check to ensure data correctness.

3.3. Controlling Networked Product Connections

Q: Is it possible to send control commands to the MCU to enable networking products to initiate client connections? Send a command to disconnect the networked product?

Answer: It is possible. The specific approach is to put networking products in TCP server mode, and send commands when connecting to TCP client mode, after which networking products can automatically connect.

The command to initiate the connection is as follows:

ed f2 a3 56 ca db 91 84 b0 d7 01 14 01 01

The command to disconnect is:

ed f2 a3 56 ca db 91 84 b0 d7 01 14 01 00

3.4. IP Address Reading Local

Send command:

ed f2 a3 56 ca db 91 84 b0 d7 00 00 04

The return data is ca a8 01c8 this hexadecimal IP address.

3.5. Modify dns Server

First, change the IP mode to static, otherwise you cannot manually set the IP of the dns server. ed f2 a3 56 ca db 91 84 b0 d7 01 38 01 00

After that, set the new dns server IP. Note that there should be a 1 second interval between the two commands. ed f2 a3 56 ca db 91 84 b0 d7 01 3e 04 08 08 08 08

Check whether the system is initialized. 3.6

When the DHCP is enabled, the initialization time of the system is indefinite. To confirm whether the system has been initialized. Can send a serial port command, if there is a response to the system initialization. Send command: ed f2 a3 56 ca db 91 84 b0 d7 00 03 01 for example. If the IP end byte of one byte is returned, the system is initialized. However, users should not send commands too frequently. Sending commands every 1 second is a recommended value, because sending commands too frequently will slow down the system's startup time.

3.7. One -time setting method.

In fact, when the system starts, the user can choose to send a one-time parameter setting command through the serial port to set all the parameters required by the user, for example, the following command is sent from the serial port:

ed f2 a3 56 ca db 91 84 b0 d7 01 00 68 c0 a8 01 c8 ff ff ff 00 c0 a8 01 01 c0 a8 01 03 10 64 00 50 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0b 5a 4c 44 45 56 30 30 30 31 00 00 03 05 14 00 00 00 00 00 00 01 00 00 00 ca 60 d 1 85 77 77 77 2e 62 61 69 64 75 2e 63 6f 6d 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 0c 3c 00 50 00 00 00 12

The command is interpreted as follows: ed f2 a3 56 ca db 91 84 b0 d7 (identify stream)01(write parameters and do not save parameters)00(from00 position Start Write)68(write together0x68 bytes) c0 a8 01 c8( IP address192.168.1.200) ff ff ff 00(subnet mask255.255.255.0) c0 a8 01 01(Gateway192.168.1.1) c0 a8 01 03(Undefined)10 64(Port4196)00 50(destination port80)01(Working mode TCP client)00 00 00 00 00 00 00 00 00 00 00 5a 4d 01 02 03 04(Equipment ID, must be the correct corresponding IP if you don't know read first ID, fill in this area)0b(the baud rate is115200)5a 4c 44 45 56 30 30 30 31 00(Equipment Name)00(check digit is none)03(the interval is3ms)05 14(packet length)00 00 00 00(Frame Start and End Characters)00(static or dynamic IP)00(Flow control mode)01(Purpose

mode)00(Serial Number of Bits)00(Conversion Agreement)00(Undefined) ca 60 d1 85( DNS)77 77 77 2e 62 61 69 64 75 2e 63 6f 6d 00(destination domain name [www . baidu . com](http://www.baidu.com) of sixteen.

Bary) 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0c (reconnection time) 3c (keep alive timing time) 00 50 (web port) 00 00 00 (UDP filtering) 12 (version)

The latest version of the parameter is 169 bytes, that is, ed f2 a3 56 ca db 91 84 b0 d7 03 00

The a9...

3.8. The serial port restarts the device.

Method 1

For firmware version V506 and above, use the 07 command to restart the module (refer to Table 2. Special Command Type). The 07 command is similar to the 03 command, except that the 07 command restarts the module. For example, send ed f2 a3 56 ca db 91 84 b0 d7 07 1f 01 00. This command attempts to rewrite the ID of the first device to 00, but since the ID cannot be modified, the actual effect of this command is to restart the module.

Method 2

For the firmware version before V506, the following method is used to restart the module. ed f2 a3 56 ca db 91 84 b0 d7 01 3e 04 08 08 08 08 or ed f2 a3 56 ca db 91 84 b0 d7 01 3e 04 08 08 04. Note: If the first command was sent last time, then this time let the module restart and send the second command, otherwise send the first command. The module will restart after each transmission.

3.9. Specify DNS Server

Describes how to manually specify a DNS server in a dynamic IP address acquisition mode. In the dynamic acquisition IP (IP mode is dynamic or DHCP) mode, the DNS server is also automatically acquired. If you want to obtain IP dynamically and set up DNS server manually, the idea is to start the module with dynamic IP mode and then modify DNS server. In this way, the module will automatically retain the automatically acquired IP, while at the same time

Allow users to manually set the IP.

The module is set to dynamic mode IP. Send: ed f2 a3 56 ca db 91 84 b0 d7 01 38 01 01

2. Check that the system has been started up: that is, it is sent every second: ed f2 a3 56 ca db 91 84 b0 d7 00 03 01. If a one-byte response is received, the startup is complete.

3. Change to static IP mode. Command Sent ed f2 a3 56 ca db 91 84 b0 d7 01 38 01 00

4. Also use the second step method to check that the system is started.

5. Send command ed f2 a3 56 ca db 91 84 b0 d7 01 3e 04 08 08 08 08. Set the DNS server IP to 8.8.8.8, and note that the 4 bytes are the hexadecimal representation of the DNS server IP.

Read device name 3.10.

ed f2 a3 56 ca db 91 84 b0 d7 00 26 0a

Write device name 3.11.

ed f2 a3 56 ca db 91 84 b0 d7 03 26 0a 61 62 63 64 65 66 67 68 69 00

3.12. ID reading device

Versions after V1.512 can support reading of the ID, and the read ID is 6 bytes. The read instruction is: ed f2 a3 56 ca db 91 84 b0 d7 00 1f 06

Read device password 3.13.

ed f2 a3 56 ca db 91 84 b0 d7 00 15 0a. Sending this command can read the 10-byte password of the device through the serial port.

3.14. IP of modification destination

For modules that support DNS, modify the Dest string parameter. For modules that do not support DNS, modify the Dest IP field. For modules that do not support DNS, modify the Dest IP field.

For example, a device supporting DNS modifies the destination IP to 192. 168. 1. 188 by sending: ed f2 a3 56 ca db 91 84 b0 d7 01 42 1e 31 39 32 2e 31 36 38 2e 31 2e 31 38 38 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00

Method of reading: ed f2 a3 56 ca db 91 84 b0 d7 00 42 1e

Note the baud rate to be consistent.

For devices that do not support domain names, the modification destination IP is 192. 168. 1. 188. The method is: ed f2 a3 56 ca db 91 84 b0 d7 01 0c 04 c0 a8 01 b5

For products with MDIP, use the 02 command: ed f2 a3 56 ca db 91 84 b0 d7 0c 04 c0 a8 01b5 for 02

3.15. Modify the destination port of UDP mode.

The last 2 bytes are the hexadecimal representation of the new destination port. After the modification, wait for 5ms or more before sending the data.

ed f2 a3 56 ca db 91 84 b0 d7 01 12 02 04 01

Read device version number 3.16.

Sending to serial port: ed f2 a3 56 ca db 91 84 b0 d7 00 67 01 can return a byte of firmware version number, and adding this value to 383 is the current firmware version number.

For example, if the return value is 0x8a, then the conversion to decimal is 138, plus the 383 is 521, so this firmware version is 521.

3.17.2003 -Low Power Mode

low power on ed f2 a3 56 ca db 91 84 b0 d7 93 01 01 01 bc f0

off low power ed f2 a3 56 ca db 91 84 b0 d7 93 01 01 00 7d 30

The network will be turned on and off within 1.5S.

Variable parameter area

4.1. Use of variable parameter areas.

As can be seen from fig. 4, the last part of the parameter is User Param. this part of the space can be used by the user for a total of 52 bytes from position 115. At the same time is also wifi parameters, multi-purpose IP and other parameters stored in the place.

Since the parameter of the variable parameter area is not fixed in position, the existence of this parameter is also changed according to needs, and the length is not necessarily the same each time, which is different from the previous usage, so it is necessary to explain its structure here.



Fig.6 Structure of variable parameter area

As shown in FIG. 6, the variable parameter area starts from the 115, param1, param2, param3, etc. are arranged, the first byte of each param is a type, and if the type is 0, it means the end. Otherwise

The following table shows:

Table 4 Variable parameter types

|  |  |  |
| --- | --- | --- |
| Name | Value | Meaning |
| PARAM VAR2 END\_\_ | 0 | It means that there are no parameters behind it. |
| PARAM VAR2 FUN SEL1\_ \_ \_ | 1 | Function selection 1 |
| PARAM VAR2 SSID\_\_ | 2 | WIFI parameters, SSID string |
| PARAM VAR2 CHANNEL\_\_ | 3 | WIFI parameters, signal number |
| PARAM VAR2 KEY\_\_ | 4 | WIFI parameters, password |
| PARAM VAR2 KEY TYPE\_ \_ \_ | 5 | WIFI parameters, encryption method |
| PARAM VAR2 VLAN TCI\_ \_ \_ | 6 | VLAN Parameters |
| PARAM VAR2 MULTI DEST IP\_ \_ \_ \_ | 7 | Multi-purpose IP parameters (IP form) |
| PARAM VAR2 SOCKET5\_\_ | 8 | SOCKET5 Agent Parameters |
| PARAM VAR2 TXD CNT\_ \_ \_ | 9 | Bytes sent parameter |
| PARAM VAR2 RXD CNT\_ \_ \_ | 10 | Bytes Received Parameter |
| PARAM VAR2 DEST DNS STRING\_ \_ \_ \_ | 11 | Multi-purpose IP parameters in the form of domain names |
| PARAM VAR2 USER DEF\_ \_ \_ | 255 | User-defined parameters |

len1 indicates the length of the param1 content 1 parameter of type1 type. Read the program after content 1. Check to see if the following byte is0, if0 representation PARAM\_VAR2\_END, there is no need to continue. Otherwise, continue to read according to the parameter format.

If you write parameters note that you need to put the PARAM\_VAR2\_END at the end to prevent out-of-bounds.

The following two functions are FindVar2Param and InsertVar2Param, the first function is used to get the 1 type of parameter content, the second parameter is used to insert a type of parameter or delete the 1 type

The parameters of the type.

#define PARAM\_VAR2\_SIZE 52

#define PARAM\_VAR2\_END 0

/\* find param of ParamType in ParamString

ParamString: input, the var2 string

RetString: output, find the string. max is PARAM\_VAR2\_SIZE=52 return: 0 for not find, else the size of RetString(content of ParamType) \*/

zl\_u8 FindVar2Param(int ParamType, zl\_u8 \*ParamString, zl\_u8 \*RetString) {

int I;

for( I = 0; I <PARAM\_VAR2\_SIZE;)

{

if(ParamString[ I] == PARAM\_VAR2\_END)

{

return 0;

}

else

{

if(ParamString[ I] == ParamType)

{

memcpy(RetString, ParamString+ I +2, ParamString[ I +1]); RetString[ParamString[ I +1]] = 0;

return ParamString[ I +1];

}

else

{

I += ParamString[ I +1]+2;

}

}

}

return 0;

}

/\* insert InsertString with type of ParamType into the string of ParamString

if the ParamType exist delete it and replace

if InsertString is NULL, delete the param frome ParamString

return:

\*/

bool InsertVar2Param(int ParamType, zl\_u8 \*ParamString, const zl\_u8 \*InsertString, zl\_u8 InsertLen)

{

int I, len;

for( I = 0; I <PARAM\_VAR2\_SIZE;)

{

if(ParamString[ I] == PARAM\_VAR2\_END)

{

break;

}

else

{

if(ParamString[ I] == ParamType)

{

/\* copy the remain data to front \*/

len = ParamString[ I +1] +2;

memcpy(ParamString+ I

PARAM\_VAR2\_SIZE-i-len);

ParamString+ I +len

/\* pad remain buffer with 0 \*/

memset(ParamString + PARAM\_VAR2\_SIZE - len, 0, len);

}

else

{

I += ParamString[ I +1]+2;

}

}

}

if(InsertString)

{

/\* insert the string \*/

if( I +InsertLen+2 <= PARAM\_VAR2\_SIZE)

{

ParamString[ I] = (zl\_u8)ParamType;

ParamString[ I +1] = InsertLen;

memcpy(ParamString+ I +2, InsertString, InsertLen);

memset(ParamString+ I +2+InsertLen, 0, PARAM\_VAR2\_SIZE-

(I +2+InsertLen));

return TRUE;

}

else

{

return FALSE;

}

}

else

{

return TRUE;

}

}

4.2. Use of parameter area of WIFI

WIFI parameters are stored using the variable parameter area, refer to the 4.1 variable parameter area of the use of the chapter section.

WIFI parameters are not necessarily in a fixed position when reading, so the method of reading is:

1. Read 52- byte data in the parameter area at the beginning of the 115, I .e. obtain all the data of var2. The serial port sends the read instruction ed f2 a3 56 ca db 91 84 b0 d7 00 73 34. Assume that the returned data is stored in var2.

2. Through the variable parameter area of 4.1, the use of the 1 section of the FindVar2Param() function to obtain the parameter content, for example, to obtain the content of the SSID, first through table 4 to find the type of SSID 2, then through

FindVar2Param(2, var2, ssid), where ssid is the output ssid string.

Here again WIFI parameter type format to do a detailed introduction:

Table 5 WIFI parameter format

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Value | Meaning | Format |
| PARAM VAR2 SSID\_\_ | 2 | SSID character WIFI parameter,String | 0-terminated string |
| PARAM VAR2 CHANNEL\_\_ | 3 | WIFI parameters, signal number | The highest bit. (0x80)1 indicates that the network port is open wifi intercommunication, generally do not need to call open; secondary high (0x40) for0 indicates open DHCP Server function. The remaining 6bits (0x3F) have values of 1 to 13, respectively representing channels of 1 to 13, and the default is 4. There is no channel setting in STA mode.It is generally set to 4.Can.  |
| PARAM VAR2 KEY\_\_ | 4 | WIFI parameters, password | 0-terminated string |
| PARAM VAR2 KEY TYPE\_ \_ \_ | 5 | WIFI parameters, encryption method | The highest bit(0x 80) is 1, which means STA mode; The second highest bit (0 x 40) is 1, which means AP mode; The remaining low 6bit (0x3F) indicates encryption, 0 indicates no encryption, 1 is WEP64, 2 is TKIP ,3 is meaningless, 4 is AES, 5 is WEP126,6 is AUTO. In general, AUTO can be |

(

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | to cover all types of encryption with passwords, so generallyChoose to 6.  |

If there are two ways to write WIFI parameters, the 1 is to use the InsertVar2Param () in the use 1 section of the 4. 1 variable parameter area to generate string write data. The 2 is to write data directly.

InsertVar2Param method:

1 . Read 52-byte data at the beginning of the 115 parameter area, that is, obtain all data of var2, assuming var2.

2. With SSID for example, first pass the table4 check it out SSID is of type2, InsertVar2Param (2, var2, "MY\_SSID") will be new SSID insert in var2.

3. The serial port sends a call to ed f2 a3 56 ca db 91 84 b0 d7 07 73 34 (var2 content), where the var2 content indicates the content of the var2 data above, with a length of 52 bytes. You must use the that needs to be restarted here.

07 Directive.

The InsertVar2Param method can retain other variable areas of the var2 parameter area, and the direct write method is the direct construction. build one var2 the disadvantage of this method is that it is possible to cover other non- WIFI variable parameters. But this method is more direct:

Assuming that the SSID to be written is MECURY\_9898, the param1 control word is 02 0c 4d 45 52 43 55 52 59 5f 39 38 39 38. Channel 4 is adopted to turn off the DHCP server function, so the control word is 03 01 44;STA mode, encryption is automatic, control word is 05 01 86; The password is 11111111 and the control word is 04 08

31 31 31 31 31 31 31 31.

param1 and param2, etc. are combined 02 0c 4d 45 52 43 55 52 59 5f 39 38 39 38 03 01 44 05 01 86 04 08 31 31 31 31 31 31 31 00. The end of the PARAM\_VAR2\_END that is 00 has been added.

The 07 command is used to write the above parameters into the 115 starting area:

ed f2 a3 56 ca db 91 84 b0 d7 07 73 1F 02 0c 4d 45 52 43 55 52 59 5f 39 38 39 39 03 01 44 05 01 86 04 08 31 31 31 31 31 31 31 31 00. where, 0x1F is the subsequent data length, which may need to be calculated based on the ssid and the key length.

Serial port to send this command after the module will restart and in accordance with the new wifi parameters to connect.

4.3. The user parameter space is used.

As you can see from Figure 4, the last part is User Param. This part of the space is available to the user for a total of 52 bytes starting from position 115. For example, the user temporarily stores data, which is used to store the data in the parameter space, and then transmits the data to the server through "regularly sending data to the server" and "parameter acquisition.

However, note that the User Param part of the module with the following functions has been used, and users should not use this part of the area. These functions include: wireless function, VLAN function, multi-target IP MDIP device, proxy server function, and registration packet heartbeat packet.

The method of writing data is to send the following command to the serial port of the module:

ed f2 a3 56 ca db 91 84 b0 d7 02 73 (size) ff (size-3) (content) 00

Where size is a size that needs to be less than 52, indicating how much data is left behind the size. The content is the written data. For example, when size is 11, the instruction is:

ed f2 a3 56 ca db 91 84 b0 d7 02 73 0b ff08 d 1 d2 d3 d4 d5 d6 d7 d8 00

where d1 to d8 are any data written by the user. The User Param data is ff08 d 1 d2 d3 d4 d5 d6 d7 d8 00 actually written. The beginning ff and the end 00 are for subsequent format compatibility with additional data.

The above data written to User Param is not necessarily saved to the module Flash (power may be lost), but temporarily stored in memory. However, it is also possible to save parameters manually when the operation is saved. The above instructions do not affect (e. g. close) the current module TCP connection.

The command type is above 02. 02 This command is very useful for users to set the multi-purpose IP of products with MDIP type numbers. However, in general, command 03 (write instruction to save parameters) can be used. However, if the user wants to save the parameters and then restart immediately, replace the 07 command with the 03 command.

4.4. Multi-target settings for MDIP devices.

The MDIP device can set up to a target IP of 8. The setting method of the first target IP has been described in "Step of Writing Parameters. However, note that MDIP also needs to use the 02 or 07 command to modify the first target IP (specifically described below), instead of the ordinary 01 write command.

This article focuses on how to set and edit the remaining 7 IP addresses and ports.

When editing the MDIP IP IP and port, the command control word in the command format in Figure 1 uses the 02 command, as shown in Table 2. Because the 02 command is specially designed for MDIP, it can detect 7 IP changes to the target and reconnect the newly added IP + port without closing the old TCP connection. The specific write command is as follows:

ed f2 a3 56 ca db 91 84 b0 d7 02 73 size 01 01 00 07 (size-6) (4byte IP) (2byte port) (4byte IP) (2byte port) &hellip;&hellip; 00

In this instruction, ed f2 a3 56 ca db 91 84 b0 d7 is the identification stream. 02 is a command word specific to MDIP products. size is the total length of the subsequent data. 01 01 00 07 is fixed data. (size-6) is the total length minus 6 bytes. Next is 4 byte IP +2 byte port, all in big endian format, with up to 7 combinations, I .e. 7 target IP. It may have an arbitrary length of 1 to 7 pieces. There is a last 00 character.

For example the command:

ed f2 a3 56 ca db 91 84 b0 d7 02 73 12 01 01 00 07 0c c0 a8 01 58 04 01 c0 a8

01 58 04 02 00

Here size is 18 bytes. And set the destination IP and port of the destination to 2 IP: 192. 168. 1.88:1025 and 192. 168. 1.88:1026.

After sending this command, the module immediately initiates a connection to the 1026 port of the newly added 192. 168. 1.88. Note:

1. Here's02 the command itself does not save the purpose IP that is, the purpose of serial port writing after power failure the IP port is lost.

2. The 07 command can be used to save the written 02 IP address. However, it also closes its TCP connection that is communicating. Only 02 TCP commands can be used without closing the current old connection.

Another feature is that the 02 TCP command can only add new ones and cannot delete (close) old ones. If shutdown is required then use the 07 command.

4.5. Custom parameters with interface.

Unlike "user parameter space usage", custom parameters here can be viewed and edited zlvircom. The parameter used is a multi-purpose IP area similar to "multi-purpose setting of MDIP devices". However, since the device is not a multi-purpose MDIP device, this area can be set by the user but will not be considered as a target IP.

For example

ed f2 a3 56 ca db 91 84 b0 d7 01 73 0c 01 01 00 07 06 00 00 00 09 00 00 00

Write IP: 0.0.0.9, Port: 0 to the device. It can then be viewed and edited with the first multipurpose of the zlircom's advanced parameters.

4.6. The serial port of the registration package and the heartbeat package is written.

If the registration package of a device is "123456" and the heartbeat package is "abcdef", select

The settings in the item are as follows, refer to "Instructions for Use of Zhuolan Custom Heartbeat Package"



Figure 7 Registration Package Heartbeat Package Example

This parameter is set to the user parameter space.

The ed f2 a3 56 ca db 91 84 b0 d7 00 73 25 command is sent to the serial port of the module to obtain the parameter space content with the above parameters set: "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 31 32 33 34 35 36 00 00 00 61 62 63 64 65 66 00 00 00 00 00 00 00 00 00 00 00 ". It can be seen that position 0x 82 starts with the string "31 32 33 34 35 36" of the registration package and ends with 3 00; Then there is

The heartbeat packet string "61 62 63 64 65 66 " also ends with a 3 00;

Note that please set the space from 0x 73 to 0x 82 to all 00. For example, to only set the heartbeat packet to "hello", you need to write "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 68 65 6c 6c 6c 6f 00 00 68 65 6c 6c 6f 00 00 00 " to the position where 0x 73 starts in the parameter area, this data length is 0x1F. Here's "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 "Yes 0x73 to 0x82 Required to set part of 00; here "68 65 6c 6c 6f 00 00 00"Yes hello(note that when there is only a heartbeat package, the registration package should also be set to be the same as the heartbeat package; 68 65 6c 6c 6f 00 00 00 "is a heartbeat pack. Note that only when registering the package, some parameters of the following heartbeat package do not need to be set.

The serial port command actually sent is:

ed f2 a3 56 ca db 91 84 b0 d7 03 73 1f 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 68 65 6c 6c 6f 00 00 00 68 65 6c 6c 6f 00 00 00

Pay attention to the position of the length field 1f, 1f is followed by data, and the part before 1f can be copied as well. The module will not restart after the above parameters are written, so the new registration report heartbeat package will not take effect immediately (the new parameter package content will be loaded into memory only when the system is started). So you can change the 03 command to 07.

The rest remains unchanged, so that the device can be restarted immediately after writing:

ed f2 a3 56 ca db 91 84 b0 d7 07 73 1f 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 68 65 6c 6c 6f 00 00 00 68 65 6c 6c 6f 00 00 00