Serial port

Modify parameters and hardware TCPIP protocol stack

Embedded device
networking solutions

ZL DUI 20090825.1.0

reserved

Version information

The following changes have been made to the document:

Modification record

Date	Version	Document number	Modified content
2009-8-25	Rev.1	ZL DUI 20090825.1.0	Release version
2010-4-26	Rev.2	ZL DUI 20090825.2.0	Added the description of command mode
2013-2-25	Rev.3	ZL DUI 20090825.3.0	Remove the "hardware mode" that is no longer used and the "hardware protocol stack library functions that are not commonly used.
2018-3-13	Rev.4	ZL DUI 20090825.3.0	Added the second generation of serial port instructions

Ownership information

CATALOGUE

1. Overview

1.1. Hardware protocol stack and software protocol stack

The hardware TCP/IP stack is relative to the software TCP/IP stack. Software TCP/IP generally provides socket interfaces to connect, listen, send, and receive by calling library functions. For example, Windows socket API functions include connect, listen, send, recv, and so on. The hardware protocol stack is a relatively new concept. The hardware protocol stack exists in the ZLSN2000 networking module. The user MCU sends commands to ZLSN2000 through serial port to control ZLSN2000 to run the TCP/IP protocol stack to achieve network functions such as connection, monitoring, sending and receiving. Its function is similar to calling the software TCP/IP stack directly.

Advantages of using hardware TCP/IP stack for embedded systems:

1. one Compared with the use of software TCP/IP protocol stack, the hardware protocol stack does not occupy the user's CPU, no RAM, reduce the burden of the user MCU, and the hardware protocol stack is a mature product, with strong stability.
2. Compared with the networking module that does not have the hardware TCP/IP protocol stack, the hardware TCP/IP makes the user MCU more flexible, and the networking module can be developed twice, basically realizing all the control functions of the software protocol stack.

1.2. Principle of hardware protocol stack

The implementation of the hardware protocol stack of ZLSN2000 is actually achieved by modifying the internal parameters of ZLSN2000. For example, when the user changes the destination domain name or IP, ZLSN2000 automatically initiates a connection to the new destination, thus realizing the connect() function.

Therefore, the function of the hardware protocol stack actually provides a way for the user to modify module parameters on the device side. This provides a method for users to modify device IP, mode and other parameters through keyboard, touch

screen and other methods on the device side.

ZLSN2000 provides two modes to modify equipment parameters: command mode and hardware mode.

Command mode: The user sends a fixed command identification stream to ZLSN2000 through the serial port to modify ZLSN2000 parameters.

Hardware mode: The IO control pin of the user's MCU is connected to the SPR and SPA of ZLSN2000, and the purpose of parameter modification is achieved through the timing control between the hardware pins.

Comparison of the two models:

1. one The command mode does not require hardware connections and has little impact on hardware. In particular, it is difficult to add hardware connections when the DB9 serial cable is used to connect the device.
2. The control sequence of hardware mode is more complicated, and the user development needs a long time. The command mode operation is relatively simple, and the command word can be simulated by the computer serial debugging tool.
3. The only advantage of the hardware mode is that there is no concern about command recognition flow and data flow interference, that is, the command recognition flow appears in the data stream sent by the user. But ZLSN2000 command recognition stream has 10 bytes, a total of 1.2×1024 possibilities, that is, 120 billion billion billion possibilities, if the highest baud rate of 115,200 BPS constantly sent data, then on average, it takes 36,604 billion years to appear once. Then the possibility is negligible.

   Suggestion: You are advised to use the command mode.

2. Command mode

2.1. Command format

The steps of reading and writing parameters in command mode are as follows: at any time, write commands as shown in Figure 1 to the serial port of ZLSN2000 to complete the reading and writing of parameters.

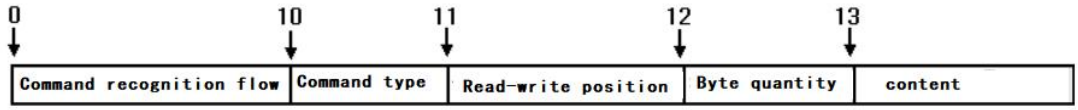


Figure 1 Command format

The command identification flow is 10 bytes, indicating the start of the command, and must contain the following data: ed f2 a3 56 ca db 91 84 b0 d7

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

Table 1. Common command types

Command word	Frequent use	Feature
0x00	Yes	Read parameters.
0x01	Yes	Write parameters. Power off does not save. However, after the local IP address, subnet mask, gateway, DHCP mode, and DNS server IP address are changed, the system restarts and saves the parameters.

Table 2. Types of specialized commands

Command word	Frequent use	Feature
0x02	no	For MDIP products: Initiate a second TCP connection without closing the previous TCP connection.
0x03	no	Write parameters. The parameters are saved and written into the Flash memory of ZLSN2000.
0x04	no	Dongle command.
0x05	no	Write parameters. The parameters must be saved, the module must be restarted, and the previous TCP connection must not be closed.
0x06	no	Read the web page (ZLSN2030EX only).
0x07	no	Same as 0x03, but you must restart the module.
0x08	no	Write web pages (ZLSN2030EX only).
0x09	no	For parametr-based communication, refer to "Sending Module Parameter

		functions.pdf"
0x0a	no	Uniform control controls triggered (ZLSN2030EX specific).
0x0b	no	For parameter-based communication to save parameters, refer to "Sending Module parameter functions.pdf"
0x0c	no	TXET control submits data (ZLSN2030EX specific).
0x0d	no	ZLAN7142 specific configuration instructions.

The read/write position is 1 byte, indicating that the read starts from the first byte of the parameter. Refer to Figure 4 for parameter format.

The number of bytes is 1 byte, indicating the amount of data to be read or written this time.

The inner part is the content that needs to be written during the write operation, and its length should be the same as the byte number field definition.

2.2. Format of the second-generation command

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

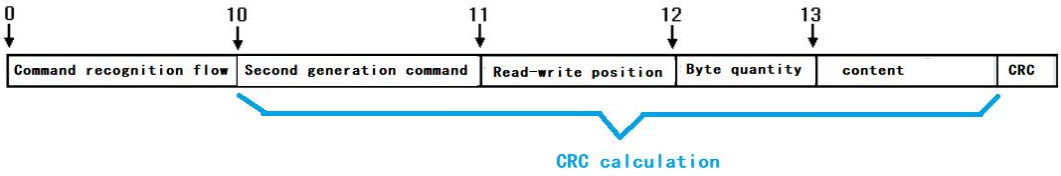


Figure 2 Format of the second-generation command

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

1. Change the command type to the second-generation command. The second generation command is obtained by adding 0x80 to the first generation command. For example, the original write command was 0x01, and now it is 0x81.
2. If it is the second-generation command, you must add the CRC16 two-byte check at the end of the command. The content of the check is without the command identification stream (without the CRC itself), as shown in the figure. If the CRC error occurs, the command will not be executed. This ensures that unexpected data writes will not be executed.

3. The second-generation command will have feedback, if the execution is successful, the correct command is returned, if the error is returned.

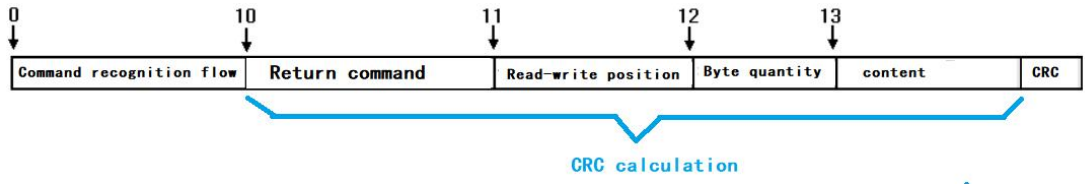


Figure 3 returns the instruction format

You can see that the format of the feedback command is the same as that of the written command, which facilitates the calculation of CRC and command identification. There are four types of "return command" : (1) 0x0e indicates an instruction format error, including a CRC error, where the read/write position is 0, the number of bytes is 0, the content is empty, and the CRC is 10 03. (2) 0x10 indicates that the instruction format and CRC are correct, but the value range may be out of bounds or the written MAC address does not match the original MAC address (the MAC cannot be modified). Note that this error only exists for instructions such as 0x03 that need to write flash. 0x01 does not return this error. (3) 0x0f indicates that the instruction is executed correctly, the read/write position is 0, the number of bytes is 0, the content is empty, and the CRC is 41 c3. (4) 0x80 indicates the returned value data of the second-generation read instruction. At this time, the read and write position and number of bytes are the same as the sent instruction, and 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 that the second-generation command be used.

Firmware tested with version 1.558(2003) is 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 the communication data. When changing to the second-generation command, first change 00 to 80 and then add the CRC check of "80 3d 01" to a1 78. So the entire command is ed f2 a3 56 ca db 91 84 b0 d7 80 3d 01 a1 78. The command output 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 return instruction, 3d is the sending position, 01 indicates that the following data content is 1 byte, followed by the content of 1 byte 00, and then the content of 80 3d 01 00 CRC check b9 b8. If the sent instruction CRC error will receive ed f2 a3 56 ca db 91 84 b0 d7 0e 00 00 10 03.

For example, the original command to change the mode to the client is ed f2 a3 56 ca db 91 84 b0 d7 01 14 01 01. Change the second generation command to ed f2 a3 56 ca db 91 84 b0 d7 81 14 01 01 a8 4c. If there is no return, the baud rate is incorrect or the module is not connected. If ed f2 a3 56 ca db 91 84 b0 d7 0f 00 00 41 c3 is displayed, the command is executed correctly. If ed f2 a3 56 ca db 91 84 b0 d7 0e 00 00 10 03 is returned, the command resolution error (including CRC error) is displayed.

It is also possible to write values that are out of bounds or attempt to modify the MAC address, such as ed f2 a3 56 ca db 91 84 b0 d7 83 14 01 04 69 f7. Note that 0x83 is to write the flash instruction instead of 0x81 to write the memory instruction, and set the working mode to 4, in fact, the mode range is 0 to 3,4 is not legal. The command returned is ed f2 a3 56 ca db 91 84 b0 d7 10 00 00 70 05. 0x10 indicates that the data content is out of bounds, causing the flash to fail to write. Also note that if the 0x81 instruction was used to write data to memory and the 0x83 instruction was used to write data to flash, the 0x83 instruction will return 10 error code as long as the contents of any of the previous instructions are out of bounds.

In addition, for example, the instruction ed f2 a3 56 ca db 91 84 b0 d7 83 1f 01 ff 59 b6, where 1f starts from the mac address, cannot be modified, here the user tries to write flash modification. An error indicating that the contents of the 10 instruction cannot be written is also returned.

The return value of the 10 instruction has two conditions: (1) There must be an action to write the flash. If the 0x81 instruction is used instead of 0x83, the actual out-of-line data will be written to memory, but because it is not written to flash and is not saved and used by the system, it is also invalid. (2) The content written to the flash cannot be correctly written.

Notice Whether it is a write or read command, it is recommended that the interval

between adjacent commands be 100ms.

2.3. Parameter format

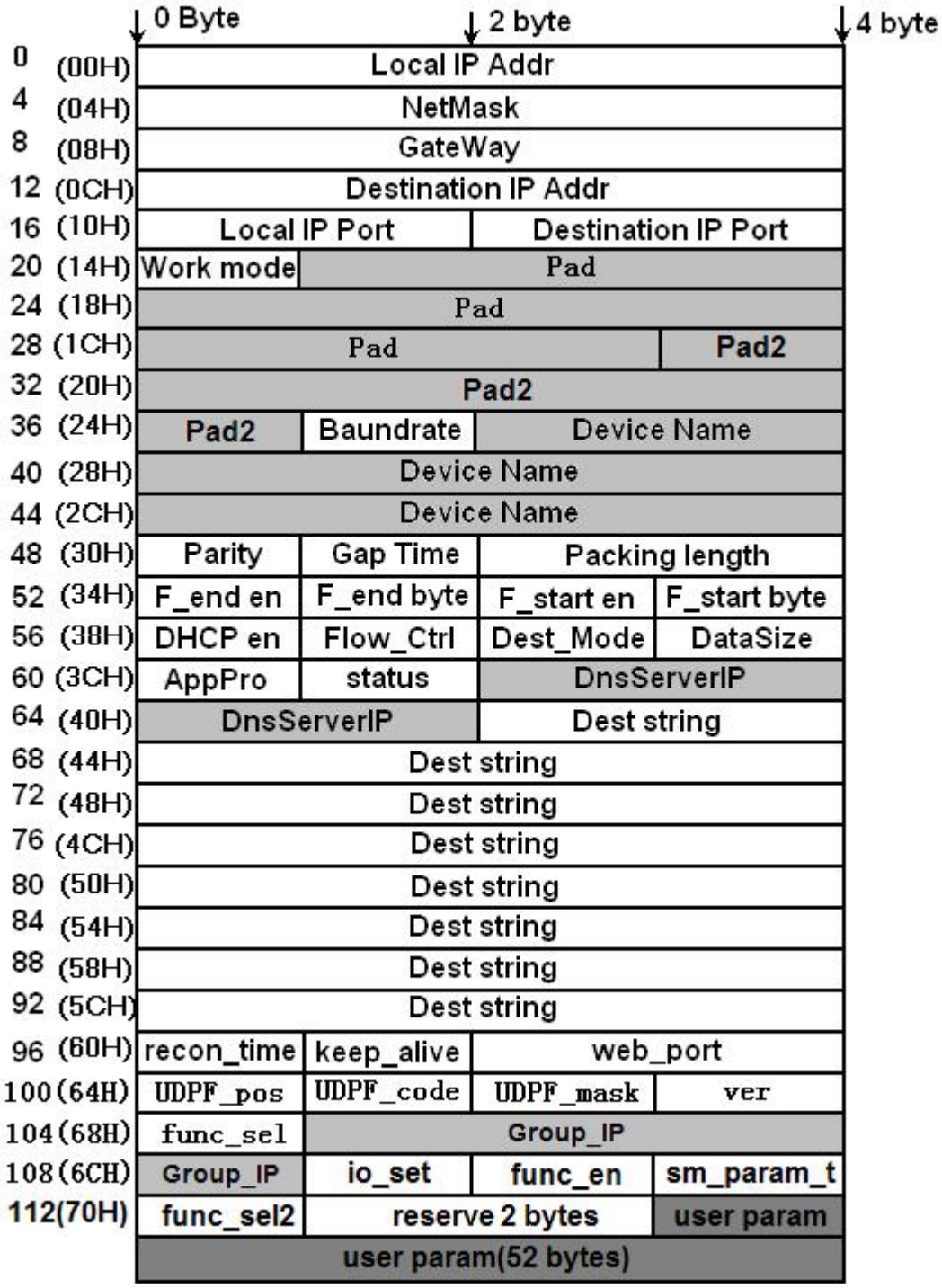


Figure 4 Parameter format

Figure 4 shows the module parameter format, when sending and receiving parameters, the first byte is sent first. The bytes in the preceding figure are in

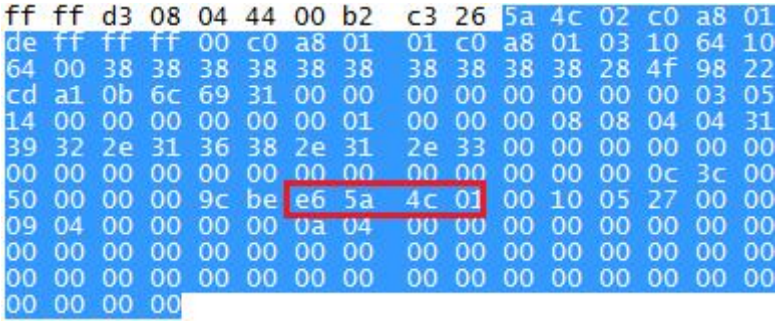
big-endian mode. For example, the high byte of Local IP Addr is on the left. Please refer to ZLSN2000 Data Manual for the meaning of each parameter.

1. Local IP Addr (local IP address) : 4 bytes.
2. Net Mask(subnet mask) : 4 bytes.
3. GateWay: 4 bytes.
4. Dest IP (destination IP address) : 4 bytes. For networked products that provide the DNS function, the DestString field takes the place of the deststring field. For networked products that do not have the DNS function, the 4-byte destination IP address is used in this field. If your product has DNS function, please consult ZLAN Company.
5. Local IP Port: 2 bytes.
6. Dest Port (destination port) : 2 bytes.
7. Work mode: 1 byte. Values 0, 1, 2, and 3 correspond to TCP Server mode, TCP Client mode, UDP mode, and UDP multicast mode, respectively.
8. Pad (padding area) : 10 bytes, should all be 0.
9. Pad2 (Fill area 2, or DevID) : 6 bytes. When the write parameter contains this field, the correct ID of the device should be filled in here, otherwise if the write ID is incorrect, the device will discard all other write parameters.
10. Baundrate: 1 byte. The values from 0 to 13 (13 indicates 460800) correspond to 1200, 2400, 4800, 7200, 9600, 14400, 19200, 28800, 38400, 57600, 76800, 115200, 230400, and 460800.
11. Device Name: 10 bytes. Must be a visible string ending in 0.
12. Parity: 1 byte. 0 to 4 correspond to None, Odd, Even, Mark, and Space.
13. Gap Time: 1 byte.
14. Packing length: 2 bytes. The value ranges from 1 to 1400.
15. F_end en (frame end character significant bit) : 1 byte, 0, 1 indicates that the frame end rule is invalid, effective. Since version V1.472 this byte is no longer valid.
16. F_end byte (half_485_wait) : 1 byte. 485 Half-duplex interval.
17. F_start en (half_485_max_wait) : 1 byte, the maximum wait time of 485 half-duplex, usually 3ms. If the value is 0, 485 half-duplex is canceled.

18. F_start byte (the first character of the frame) : 1 byte. Since version V1.472 this byte is no longer valid.
19. DHCP en (DHCP significant bit) : 1 byte. 0 and 1 indicate that static IP addresses are used and DHCP is used to obtain IP addresses.
20. Flow Control (flow control mode) : 1 indicates the use of CTS, RTS flow control; 0 indicates that flow control is not used.
21. Dest_Mode(destination mode) : 1 byte. 0 and 1 indicate static mode and dynamic mode respectively.
22. DataSize (serial port bits) : 8 to 5 bits, corresponding to 0, 1, 2, and 3 respectively.
23. AppPro (Conversion protocol) : 0 represents transparent transmission, 1 represents conversion between Modbus TCP and Modbus RTU, and 2 represents RealCom.
24. Status (modified mode) : When reading this parameter, bit0=1 of Status indicates that the current TCP connection has been established or is in the 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 it to the IP address of the DNS server, 4 bytes.
26. Dest string (destination address) : specifies the destination IP address of a network product with DNS. For example, write "192.168.0.3" to the hexadecimal string: 0x31, 0x39, 0x32, 0x2e, 0x31, 0x36, 0x38, 0x2e, 0x30, 0x2e, 0x33, 0x00. Note that a string ending 0 is added at the end. For networked products that do not have the DNS function, this field is invalid.
27. Recon_Time: 1 byte the value ranges from 0 to 255.
28. Keep_Alive (keepalive time) : 1 byte. The value ranges from 0 to 255.
29. Web_port(Web access port) : 2 bytes, the port for accessing a web page through a browser.
30. UDPF_pos, UDPF_code, UDP_mask (UDP application layer filtering parameters) : A total of three bytes, generally set to 0, you can. This parameter is currently invalid.
31. ver (module version) : 1 byte and cannot be modified. ver=0 indicates that the

version number is 1.383. The actual version number is 383+ver. For example, ver=117, the version is 1.500.

32. func_sel (function supported by the module) : 1 byte and cannot be modified. The specific meaning of each is as follows: bit0=1 indicates that web page download is supported; bit1=1 indicates that the DNS domain name system is supported. bit2=1 indicates that REAL_COM is supported. bit3=1 indicates that Modbus TCP to RTU is supported. bit4=1 indicates that serial port modification parameters are supported. bit5=1 indicates that automatic IP (DHCP) acquisition is supported. bit6=1 indicates that the storage expansion EX function is supported. bit7=1 indicates that multiple TCP connections are supported.
33. Group_IP (UDP multicast address) : 4 bytes, ranging from 224.0.0.0 to 239.255.255.255. The locations throughout the package are as follows.



34.io_set (I/O Port Settings) : indicates the I/O port control word. For details, see the I/O control documents.

35.func_en: Function selection/enable control word. bit0=1 Enables "Data restart function" and bit1=1 enables "Send module parameter function to central server". bit2=1 Enables "Password required for parameter modification." bit3=1 Enable UDP Disable Broadcast packets. Please refer to related documents. bit4=1 Enable P2P functionality.

36.sm_param_t (sm parameter time) : indicates the interval for sending module parameters to the central server. The unit is minute.

37.func_sel2 (Function supported by the module 2) : The advanced function supported by the module. bit0=1 indicates that I/O configuration is supported. bit1=1 indicates that UDP multicast is supported. bit2=1 indicates that the multi-destination IP function is supported.

38. Reserve 1 byte (QueryOneAckOneMaxWait) : multiple hosts. 0 is to cancel the multi-host function, otherwise it is the most common waiting timeout time divided by 32.

39. Reserve 1 byte

40. Reserve 52 bytes. The reserved 54 bytes are reserved for future upgrades. The total length of the parameter is 167 bytes. (For UDP management port protocol, you need to add the first two identifiers and one command type, and the total length is 170 bytes.)

Parameter format C is described as:

```
typedef unsigned char    zl_u8;
typedef char             zl_s8;
typedef unsigned short   zl_u16;
typedef short            zl_s16;
typedef unsigned long    zl_u32;
typedef bit              zl_bool;
typedef signed long      zl_s32;

typedef zl_u32 IP_ADDR;

#define MAX_KEY_LEN                 10
#define MAX_DEV_NAME_LEN            10
#define ETHER_ADDR_LEN              6
#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_u16 param_max_data_len;
    zl_u8 param_fram_end_en;
    zl_u8 param_frame_end_byte;
    zl_u8 param_fram_start_en;
    zl_u8 param_frame_start_byte;
```

```
    zl_u8 param_ip_mode;
    zl_u8 param_flow_control;
    zl_u8 param_dest_dynamic;
    zl_u8 param_data_bits;
    zl_u8 app_protocol;
    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];
};
```

2.4. Read parameters

The commands shown in Figure 1 are sent to ZLSN2000 as quickly and continuously as possible. That is, the pause between each byte should not be longer than the Gap Time (interval Time), please refer to the ZLSN2000 manual on the Gap Time description, for 115200bps, the default Gap Time is 3ms. The baud rate sent must be the current model rate of the module. After the command is sent, the serial port can receive the specified parameters.

Take reading the destination domain name or IP String as an example. Figure 4 shows that the Dest String ranges from 42H to 60H and contains 1EH bytes. The command should be: ed f2 a3 56 ca db 91 84 b0 d7 00 42 1E. 00 indicates the read command and does not write to the Flash, 42 indicates the start position of the parameter, and 1E indicates the number of bytes to be read.

The result of sending the above command with ZLComDebug in hexadecimal is shown in Figure 5. It can be seen that the destination IP string is received in the serial port at this time.

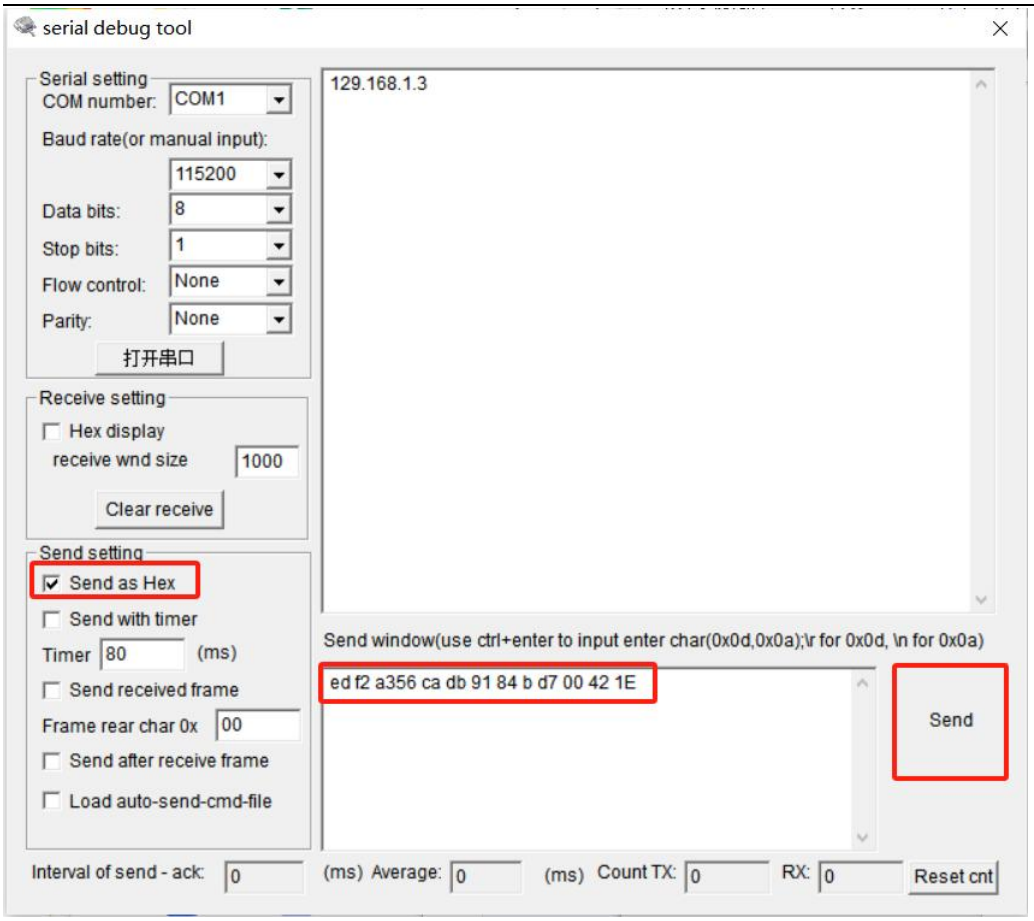


Figure 5 Reading parameter demonstration

Usage Tip: When reading parameters in communication, the data that may be returned may conflict with communication data. For example, if the connection status is ed f2 a3 56 ca db 91 84 b0 d7 00 3d 01, 00 indicates that the connection is not connected. 01 indicates that the connection is connected. But this may conflict with the communication data. To this end, the query can be changed to: ed f2 a3 56 ca db 91 84 b0 d7 00 3d 05, so the return is 00 08080404 04, as shown in Figure 4, the following 08080404 is DNS, and this DNS is generally fixed 08080404. This is equivalent to a signature suffix that provides a distinction between communication data and query data for users.

2.5. Write parameters

写 The procedure for writing parameters is basically the same as that for reading parameters, as long as the command type is changed from 00 to 01 or 03 (or 03 if the

command type needs to be written to Flash).

For example, the command to change the target domain name to www.zlmcu.com without saving it to Flash is as follows: 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

Where 01 is a command type and indicates a write parameter but does not need to be saved to the Flash, 42 is the start position of the Dest String parameter, 1e is the Dest String parameter size, followed by the hexadecimal value of the destination domain name, and the length is 1e.

The write operation closes the current network connection first, and then establishes the network connection after the read/write operation is complete. If the IP address, subnet mask, gateway, IP mode, and DNS server are changed, the system automatically restarts after these parameters are changed.

Note: During system initialization, especially in DHCP=1 mode, the startup time is slow, and the parameters written in this case cannot be received by the module. In this case, you need to send ed f2 a3 56 ca db 91 84 b0 d7 00 3d 01. If 00 or 01 is returned, you can write parameters. If no data is returned, wait and then check to see if the initialization is complete.

2.6. Precautions

1. one Users can view the parameters through the ZLVircom program to check whether the parameters have been successfully modified.
2. 2. Sometimes it takes 2-3 seconds to read the parameters. This is because you put the module on the TCP client, in TCP client mode, if the connection is not established (LINK light is not on), then sometimes there is a delay in reading and writing parameters. The reason is that the external command response is suspended when the TCP client is conducting connection operations. This problem does not occur if you are in TCP server or UDP mode, or if the connection is established.
3. 3. Intermittent condition when reading parameters. When a parameter is read using the parameter read command, the module continuously outputs the

parameter content, but the entire output content may be interrupted for a few milliseconds in between. This is because the output of parameters and the output of network port to serial port data use different mechanisms, the latter will not occur intermittent situation.

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

3. Typical applications

3.1. Read the connection status

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 returns a status word of one byte, if the status word 0x00 indicates that it is not connected (display this data in hex mode in serial debugging Assistant). A value of 0x01 indicates a connection. The so-called connection is that the TCP connection has been established or is in UDP mode, and the connection is equivalent to the LINK (usually green) light on (LINK pin level is 0). Note that all UDP queries are 1.

If the module is in busy working state, it may not respond to the above command temporarily, and the above command is simply ignored. Therefore, the user's receiver needs to have a waiting timeout mechanism. Busy working conditions include:

1. The module is in the process of connecting to the destination IP address. If a tcp connection cannot be established, the maximum duration of the connection is 5 seconds.

2. The module is being initialized. The startup time varies depending on whether the DHCP IP mode is used.

The above sent serial port command will not be received by the other end (such as the PC) connected to the ZLAN module, which ensures that the sending of the above command will not affect the communication protocol.

The average interval between sending the preceding command and receiving the status word is about 7ms. Therefore, a timeout period of 10ms is acceptable.

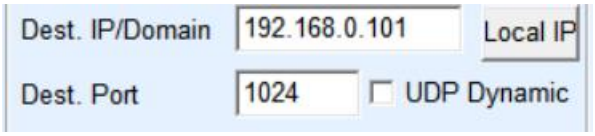
For the networking module in the TCP client, the user can send the above command every 100ms (depending on the user's needs, and as large as possible) to detect whether the connection is established. User protocol data can be sent once the connection is established.

The query can be changed to: ed f2 a3 56 ca db 91 84 b0 d7 00 3d 05, so the return is 00 08080404 04, as shown in Figure 4, the following 08080404 is DNS, and this DNS is generally fixed 08080404. This is equivalent to a signature suffix that provides a distinction between communication data and query data for users.

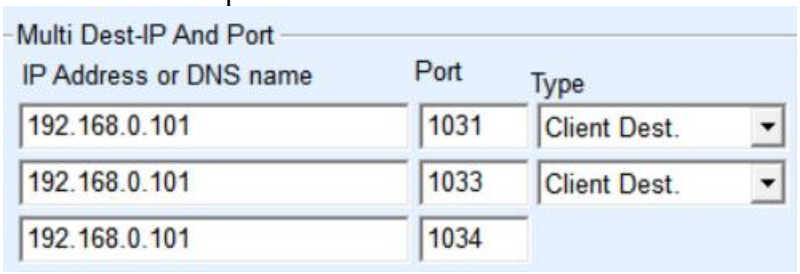
3.2. Read the status of multiple connections

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

Main interface:



More advanced options



They are called connection 1, connection 2, connection 3, and connection 4, respectively. Now you need to query the status of these connections.

Method 1: Without CRC check method, send hexadecimal ed f2 a3 56 ca db 91 84 b0 d7 00 3d 01 and return a byte X. If bit1 (the rightmost bit) of X represents the status of connection 1, 1 means connected and 0 means not connected; bit2 represents the second connection state; And so on. For example, if the value 0x0b is returned, connection 1, connection 2, and connection 4 are established.

Method 2: With the CRC check method, send hexadecimal ed f2 a3 56 ca db 91 84 b0 d7 80 3d 01 a1 78. The last two bits are CRC16 and return ed f2 a3 56 ca db 91

84 b0 d7 80 3d 01 0b f8 7f, where ed f2 a3 56 ca db 91 84 b0 d7 is the identification code and 0b is the required connection status. f8 7f is CRC check of 80 3d 01 0b. This method is more complicated than method 1 analysis, but there is CRC check, which can ensure the correctness of the data.

3.3. Control the connection of networked products

Q: Is it possible to send a control command through the MCU to have a networked product initiate a client connection? Send a command to disconnect connected products?

A: Yes. The specific approach is to make the networking product usually placed in TCP server mode, and send commands to make the networking product placed in TCP client mode when the connection needs to be initiated, and then the networking product can be automatically connected.

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. Read the local IP address

Send instructions:

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

The returned data is ca a8 01 c8 hexadecimal IP address.

3.5. Modify the dns server

Change the IP mode to static; otherwise, you cannot manually set the IP address of the dns server.

ed f2 a3 56 ca db 91 84 b0 d7 01 38 01 00

Then set the IP address of the new dns server. Note that there should be an interval of 1 second between the two commands.

ed f2 a3 56 ca db 91 84 b0 d7 **01 3e 04** 08 08 08 08

3.6. Check whether the system is initialized

When DHCP is enabled, the initialization time of the system is variable. To check whether the system has been initialized. You can send a serial port command, and if there is a response, the system initialization is complete. For example, run the ed f2 a3 56 ca db 91 84 b0 d7 00 03 01 command. If the last byte of the IP address is returned, the system initialization is complete. However, the user should not send the command too frequently. Sending the command every 1 second is recommended, because sending the command too frequently will slow down the startup time of the system.

3.7. One-time setting method

When the system starts, you can send a one-time parameter setting command through the serial port to set all the parameters required by the user. For example, send the following command through 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 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 d1 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 00 00 0c 3c 00 50 00 00 00 12

The command is explained as follows: ed f2 a3 56 ca db 91 84 b0 d7 (Identify stream) 01 (write parameter, The parameter is not saved.) 00 (write from 00 position) 68 (write 0x68 bytes in total) c0 a8 01 c8 (IP address 192.168.1.200) ff ff ff 00 (subnet mask 255.255.255.0) c0 a8 01 01 (Gateway 192.168.1.1) c0 a8 01 03 (undefined) 10 64 (Port 4196) 00 50 (Destination port 80) 01 (Working mode TCP client) 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 5a 4d 01 02 03 04 (Device ID, must be the correct corresponding IP address, if you do not know the ID first read, Enter in this field) 0b (baud rate: 115200) 5a 4c 44 45 56 30 30 30 31 00 (Device name) 00 (Check bit: none) 03 (interval: 3ms) 05 14 (packet length) 00 00 00 00 (frame start and end characters) 00 (Static or dynamic IP address) 00 (Flow control mode) 01 (destination mode) 00 (serial port bits) 00 (Conversion protocol) 00 (undefined) ca 60 d1 85 (DNS) 77 77 2e 62 61 69 64

75 2e 63 6f 6d 00 (hexadecimal of the destination domain name www.baidu.com) 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0c (Reconnection time) 3c (Keepalive time) 00 50 (web port) 00 00 00 (UDP filtering) 12 (version)

The latest version of the parameter is 169 bytes, which is changed to ： ed f2 a3 56 ca db 91 84 b0 d7 03 00 a9 …

3.8. Restart the device through the serial port

Method 1

Firmware version V506 or later, run 07 to restart the module (see Table 2. Special command type). The 07 command is similar to the 03 command, except that 07 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 first part of the device ID directly to 00, but since the ID cannot be changed, the actual effect of this command is to restart the module.

Method 2

If the firmware version is earlier than V506, perform the following steps to restart the module. Alternately send ed f2 a3 56 ca db 91 84 b0 d7 01 3e 04 08 08 08 08 08 08 or ed f2 a3 56 ca db 91 84 b0 d7 01 3e 04 08 08 04 04. Note: If the last restart sent the first command, then this time let the module restart to send the second command, and vice versa to send the first command. The module restarts after each transmission.

3.9. Specify a DNS server

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

1. Set the module to dynamic IP address mode. Send: ed f2 a3 56 ca db 91 84 b0 d7 01 38 01 01
2. Check that the system is successfully started. That is, the system sends the following message every second: ed f2 a3 56 ca db 91 84 b0 d7 00 03 01. If a one-byte reply is received, the startup is complete.
3. Change the IP address mode to static. Send command ed f2 a3 56 ca db 91 84 b0 d7 01 38 01 00
4. Repeat Step 2 to check that the system is started.
5. Send the command ed f2 a3 56 ca db 91 84 b0 d7 01 3e 04 08 08 08 08 08. Set the DNS server IP address to 8.8.8.8. Note that the last four bytes are the hexadecimal representation of the DNS server IP address.

3.10.Read the device name

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

3.11.Write the device name

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

ed f2 a3 56 ca db 91 84 b0 d7 00 03 01View initialization

3.12.Read the device ID

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

3.13.Changing the destination IP address

Change Dest string (destination IP address) for a module that supports DNS. Change Dest IP (destination IP address) for a module that does not support DNS.

For example, to change the destination IP address of a device that supports DNS to 192.168.1.188, perform the following operations: 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

The reading method is ed f2 a3 56 ca db 91 84 b0 d7 00 42 1e

Make sure the baud rate is consistent.

To change the destination IP address of a device that does not support a domain name to 192.168.1.188, run the following command: ed f2 a3 56 ca db 91 84 b0 d7 01 0c 04 c0 a8 01 b5

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

3.14.Modify the destination port in UDP mode

The last two bytes are the hexadecimal representation of the new destination port. After the modification is complete, wait more than 5ms before sending data.

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

3.15.Reading the device version number

Send to the serial port: ed f2 a3 56 ca db 91 84 b0 d7 00 67 01 to return the firmware version number in one byte, adding 383 to this value is the current firmware version number.

For example, if the value 0x8a is returned, then the decimal value is 138, adding 383 is 521, so the firmware version is 521.

3.16.Multi-objective setup of MDIP devices

MDIP devices can be configured with up to eight target IP addresses. The setting method of the first target IP address is described in the "Write Parameter Steps". However, note that MDIP also needs to use the 02 or 07 command to modify the first target IP address (described in the following section), and cannot use the common 01 write command.

This section describes how to set up and edit the remaining 7 target IP addresses and ports through the serial port.

When editing the IP and port of the MDIP, 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 specifically designed for MDIP, it can detect changes in the seven destination IP addresses and reconnect the newly added IP+ port without closing the old TCP connection. The 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) …… 00

In this instruction ed f2 a3 56 ca db 91 84 b0 d7 is the identification stream. 02 is a command word specifically for MDIP products. size is the total length of the subsequent data. 01 01 00 07 is a fixed value. (size-6) is the total length minus 6 bytes. This is followed by 4-byte IP+ 2-byte ports, both in big-endian format, with up to 7 combinations, i.e., 7 destination IP addresses. It can also be any length from 1 to 7. There is a 00 character at the end.

For example, 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. Then set two destination IP addresses and ports. The destination IP addresses are 192.168.1.88:1025 and 192.168.1.88:1026.

After sending this command, the module immediately initiates a connection to port 1026 of the newly added 192.168.1.88.

Attention:

1. The 02 command does not save the destination IP address. That is, the destination IP address written to the serial port is lost after a power failure.
2. Run 07 instead of 02 to save the written IP address. However, it also closes other TCP connections that are communicating. Only the 02 command can not close the existing TCP connection.
3. The 02 command can only add new TCP connections, but cannot delete or close old TCP connections. To disable it, run the 07 command.

3.17.User parameter space usage

As you can see from Figure 4, the last part is User Param, a total of 52 bytes of space that can be used by the user starting at position 115. For example, users temporarily store data, store data in the parameter space, and then transmit data to the server through Periodic Send data to the server and Parameter Obtain.

However, the User Param part of the module with the following functions has been used, and users should not use this part of the module, such as wireless function,

VLAN function, MDIP device with multi-destination IP address, proxy server function, and registered 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

size is a size, which must be less than 52, indicating how much data is left after size. content is the data written. For example, when size is 11, the instruction is:

ed f2 a3 56 ca db 91 84 b0 d7 02 73 0b ff 08 d1 d2 d3 d4 d5 d6 d7 d8 00

d1 to d8 indicates any data written by the user. The actual data written to User Param is ff 08 d1 d2 d3 d4 d5 d6 d7 d8 00. ff at the beginning and 00 at the end are additional data for subsequent format compatibility.

The above data written to the User Param is not necessarily saved to the module Flash (which can be lost in a power failure), but is temporarily stored in memory. However, it is also possible to save the parameters manually when the operation is done. The above commands do not affect (for example, close) the TCP connection of the current module.

The preceding command type is 02. 02 This command is useful for users to set the multi-purpose IP of MDIP models. But in general, you can use the 03 command (save the parameter write instruction). However, if the user wants to save the parameters and restart immediately, use the 07 command instead of the 03 command.

3.18.User-defined parameters with interface

Unlike the "user parameter space usage", the custom parameters here can be viewed and edited with zlvircom. The parameter used is a multi-purpose IP zone similar to "Multi-purpose Settings for MDIP devices", except that since the device is not a multi-purpose MDIP device, this zone can be set by the user but is not considered a destination IP.

Such as

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 address 0.0.0.9 and port 0 to the device. It can then be viewed and edited with the first multi-purpose of zlircom's advanced parameters.

3.19.Serial port write of registered packets and heartbeat packets

If the registered packet of a device is 123456 and the heartbeat packet is abcdef, set the following parameters in more advanced options. For details, see the User Guide for Customized Heartbeat Packets.

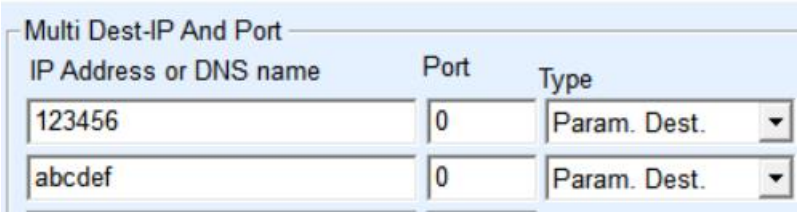


Figure 6 Example of a registered packet heartbeat packet

This parameter is set to the user parameter space.

Send the command ed f2 a3 56 ca db 91 84 b0 d7 00 73 25 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 00 00 31 32 33 34 35 36 00 00 00 00 61 62 63 64 65 66 00 00 00 00 00 00 00 00 00 00 00 00 00 00". It can be seen that the position 0x82 starts with the string "31 32 33 34 35 36" of the registration package, followed by 3 00 at the end; Next is the heartbeat packet string "61 62 63 64 65 66" with 3 00 at the end;

Note Please set the space from 0x73 to 0x82 to all 00. For example, to set only the heartbeat packet "hello", "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 68 65 6c 6c 6f 00 00 00 00 00 68 65 6c 6c 6f 00 00 00" needs to be written to the beginning of the parameter field 0x73, The data length is 0x1F. Here "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00" is the part of 0x73 to 0x82 that needs to be set to 00; 68 65 6c 6c 6f 00 00 00 is the registration package of hello. (Note that if there is only a heartbeat packet, the registration package must be set to the same as the heartbeat packet. The "68 65 6c 6c 6f 00 00 00" is the heartbeat packet. Note Only when the packet is registered, the following heartbeat packet parameters do not need to be set.

The actual serial port instructions sent are:

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 00 00 00 68 65 6c 6c 6f 00 00 00 68 65 6c 6c 6f 00 00 00**

Note the position of the length field 1f, 1f is followed by data, and the part before

1f can be copied as well. The module does not restart after the preceding parameters are written. Therefore, the new registered heartbeat packet does not take effect immediately (the new parameter packet is loaded into the memory only after the system is started). Therefore, you can change the 03 command to the 07 command, the other remains unchanged, so that after writing, you can restart the device immediately:

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**

4. After-sales service and technical support

Shanghai Zlan Information Technology Co., LTD

Address: Room 2001, Jinyuan Center, No.28 Yuanwen Road, Xinzhuang Town, Minhang District, Shanghai

Tel: 021-64325189

Fax: 021-64325200

website：http://www.zlmcu.com

mailbox：support@zlmcu.com