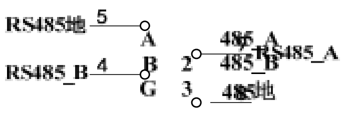
Modbus, Communication function description V3.3

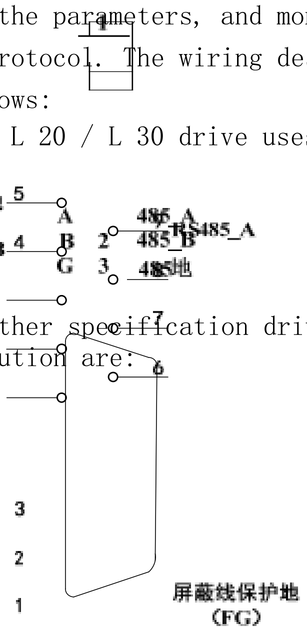
1 Function overview and wiring diagram

The servo drive has the RS-485 serial communication interface, which can control the servo system, change the parameters, and monitor the status of the servo system through the MODBUS protocol. The wiring description of the communication port CN3 is as follows:

The communication port of the L 20 / L 30 drive uses three terminals with shape and pin distribution:



The communication ports of other specification drives use double row DB 9 socket, shape and pin distribution are:



Special note: RS 485 communication end, the mouth can also be connected through 40 / 41 / 42 of CN1, see the product specification.

2 Communication parameters

P-181	Drive communication ID number	scope	default value	unit
		- 1~32	- 1	

When using RS-485 communication, the upper controller is the host, and the servo drive is the slave. The communication address of the servo drive should be set to different communication station numbers. The setting range of the station number address is-1-32. The default value of-1 indicates the closed communication function, and the set value greater than 0 indicates the open communication function. Before using the communication function, this parameter must be set to the required station number. This station number represents the absolute address of the drive in the communication network. A set of servo drives can only set the station number. If the station number is repeated, it will lead to no normal communication.

P-182	MODBUS Communication Baud rate	scope	default value	unit
		0～3	2	

Through this parameter selects the wave rate of RS-485 communication.

Different values correspond to different port rates. The selected communication wave rate should be consistent with the communication wave rate of the upper controller. The specific set value is as follows:

Parameter significance:

0: Porter rate is 4800bps	1: Porter rate is 9600bps
2: The Porter rate is 19,200 bps	3: The Porter rate is 38,400 bps
4: The Porter rate is 57,600 bps	5: The Baud rate is 115,200 bps

P-183	MODBUS Communication data mode selection (temporarily only support the RTU data format)	scope	default value	unit
		0～5	1	

Select the data mode of RS-485 communication through this parameter. The selected data mode shall be consistent with the communication protocol of the upper controller. The specific parameter values is as follows:

0: Data bit-8 bit check bit-no stop bit-1 bit

1: Data bit-8-bit check bit-even check (Even) stop bit-1 bit

2: Data bit-8-bit check bit-odd check (Odd) stop bit-1 bit
3: Data bit-8-bit check bit-no-stop bit-2-bit
4: Data bit-8 bit check bit-even check (Even) stop bit-2 bits
5: Data bit-8-bit check bit-odd check (Odd) stop bit-2 bits

3 MODBUS communication protocol

When using RS-485 serial communication, each servo driver must set its communication station number on the parameter P181 in advance, and the computer or the upper controller shall control the corresponding servo driver according to the station number; the port rate must set the parameter P182 according to the communication mode of the upper controller; the MODBUS communication protocol temporarily supports the RTU (Remote Terminal Unit) mode, and the user may set the required communication data mode on the parameter P183 according to the requirements of the upper controller. After the above parameters are set, the parameter saving operation must be performed and the drive can be restarted. The following describes the MODBUS communication.

Communication data structure:

ORTU pattern:

MODBUS The application layer protocol defines a simple protocol data unit (PDU), as shown in the figure below, which does not depend on the underlying communication layer.

Address field	FC	Data field	check code

MODBUS Starting the address field as the frame, the content of the address field is the valid address value (0~247), the host places the slave address value in the address field of the request information to determine the receiver of the request information, after the qualified slave receiving the information and completing the corresponding processing, places the own address value in the address field of the response information to make the host know what the response is sent by the machine.

The address field is followed by a function code, which indicates what the slave will do. The function code is followed by the data field, which contains the parameters of the request and the response. According to the function code, the format, length and meaning of the data field are also different.

The verification code is used to verify the validity of information and ensure the reliability of information transmission. In RTU mode, the 16-bit CRC (Cyclical Redundancy Check) check is used.

In RTU mode, each 8-bit one byte of data in the information frame is sent by 24-bit 16 decimal codes, such as 1 byte data 64H.

The RTU mode is a bit-oriented transmission mode, starting with an idle time of not less than 3.5 characters, and then sending the address field. The online device monitors the communication bus and receives the address field information after continuously monitoring an idle time of 3.5 characters. When judging that the received address field information is valid, it continues to receive the subsequent information, and then operates according to the function code and additional information, if the response information is required and sent to the host machine. The last sent byte indicates the end of the information frame after about 3 characters of idle time, and a new information frame can be sent..5 RTU mode maintains frames synchronization by simulated synchronization information, and the entire information frame must be transmitted as a continuous data stream. If a continuous data stream is transmitted and the receiving device detects more than 1.5 character intervals, one frame of data is considered received and the next receiving character is taken as the beginning of the next frame. Under normal circumstances, the interval between information frames is at least 3.5 characters, that is, one frame of data is sent, and at least 3.5 characters of idle time is needed to send the next frame of data.

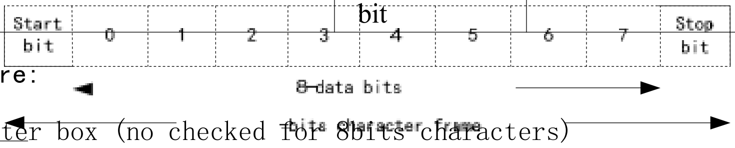
The information frame format in the RTU mode is:

START	Address Field (ADR)	**Function code (CMD)**	Data field **(DATA)**	check code **(CRC)**	**E ND**
T 1-T 2-T3-T4	8 bits	8 bits	n *8 bits	16 bits	T 1-T 2-T3-T4

Note: T1-T2-T3-T4 means that the minimum time interval from the previous frame is 3.5 characters.

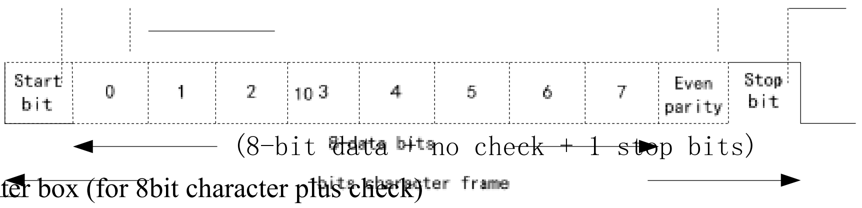
In RTU mode, the format for sending each byte is:

start bit	Data level (low in front)	parity check bit	stop bit
1 bit	8 bits	1 bit or no bit	1 bit or 2 bits



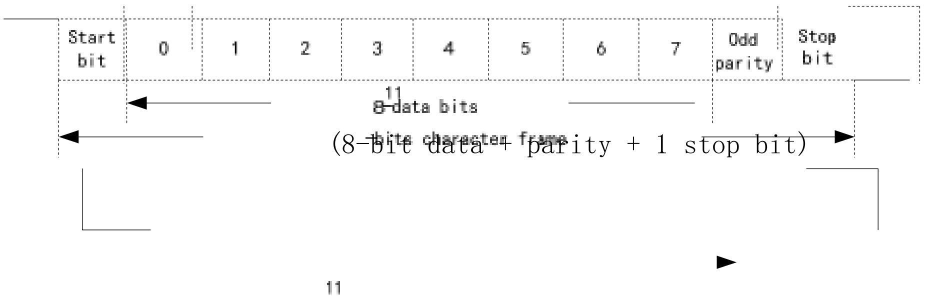
character structure:

10bits character box (no checked for 8bits characters)



(8-bit data + no check + 1 stop bits)

11bits character box (for 8bit character plus check)



(8-bit data + parity + 1 stop bit)

(8-bit data + odd check + 1 stop bits)

Communication Information Frame Format:

The items in the communication information frame format box are described as follows:

p START (Communication Start)

RTU mode: The minimum time interval from the previous frame is 3.5 characters.

PADR (mailing address)

Legal mailing address range between 1 and 32 as following: Communication with the servo drive with station number 16 (hex 10H):

RTU pattern:ADR=10H

The pCMD (function code) and the DAT A (data field)

The format of the data characters depends on the function code. The common functional codes are described as follows:

Command code 03H: Read the parameter value of the drive

Command code 04H: Read the running state of the drive (such as motor speed, position, current, torque, etc.) Command code 06H: parameters written to the drive (single write)

Command code 41H: Write the parameters of the drive to the EEPROM (parameter saving) p CRC (RTU mode) Frame

calibration calculation:

RTU mode:

RTU mode adopts CRC (Cyclical Redundancy Check) frame check, CRC frame check calculation with the following steps:

Step 1: Initialize a 16b its register with a FFFFH content, called the CRC register.

Step 2: compare the first byte of the command information with the low byte of the 16-bitsCRC register, and save the result back in the CRC register.

Step 3: Check the lowest bit (LSB) of the CRC register. If this bit is 0, move one right; if this bit is 1, the CRC register value moves one right and then varies with A001H.

Step 4: Back to step 3, until step 3 has been executed 8 times, and then into step 5.

Step 5: Repeat steps 2 to step 4 for the next byte of the command information until all the bytes complete the above processing, when the content of the CRC register is the CRC frame check.

Note: After calculating the CRC frame check, first fill in the low level of CRC in the command information, and then fill in the CRC high level. Please refer to the following examples.

For example, read the parameter 0 segment 05 of the servo drive with station number 01H. The last content of the CRC register calculated from the ADR to the last byte of the data is 3794H, then the command information is as follows, noting that the byte 94H shall be transmitted before the byte 37H.

A DR	01H
CM D	03H
Startin g data locat ion	00H (high bytes)
	05H (Low bytes)
Number of data	00H (high bytes)
	02H (Low bytes)
CRC Low	94H (high bytes)
CRC High	37H (Low bytes)

p END (End of communication):

RTU mode: The minimum time interval from the next frame is 3.5 characters.

Function Code (CMD):

The function code is 1 byte (8bi ts), ranging from 1 to 255.

Function code (HEX)	Specific meaning	Modbus The meaning in the protocol specification
03H	Read the parameter value of	Read Holding Register

	the drive (can be read simultaneously, take multiple parameters)	
04H	Read the operating state of the drive (such as motor, speed, position, current, torque and other variable values, can read multiple states simultaneously)	Read Input Register
06H	Parameters for the write drive (single write)	Write Si ngle Register
10H	Parameters for the write drive (multiple writes)	Write Mul tiple Register
08h	deagnostic function	Diag nostic
41H	Write the parameters of the drive to the EEPROM (parameter save)	

Various exceptions and errors may occur during communication, defined in order to be identified by the host

The corresponding exception codes are provided, as shown in the table below. When the information from the host receives an error or fails to complete the normal response, the slave will send the error response frame to the host, that is, the highest position of the function code byte 1, and the data field places the corresponding exception code (Except ion Code)。

Exception Code (HEX)	Mea n
01h	ILLEGAL FUNCTIO N
02h	ILLEGAL DATA ADDRES S
03h	ILLEGAL DATA VALUE
04h	SLAVE DEVICE FAILUR E
05h	ACKN OWLEDGE
06h	S L AVE DEVICE BUSY

4 Write and readout of the parameters

For all parameter details of servo drive, refer to the parameter section of debugging manual. The parameters are divided by parameter segments. Each parameter, the number is expressed by the data of 16bits, the correspondence address of each parameter is determined by the parameter serial number (range 0~249), for example: the mailing address of parameter P-005 is 005 (decimal), the correspondence address of parameter P108 is 108 (decimal), parameter P-204, the mailing address is 204 (decimal), other parameters and so on.

All the parameters described in the parameter section can be read by communication, but only part of the parameters are open to allow the user to modify the write, the parameters listed in the debugging manual are allowed to modify the write, and the other parameters not explained are the servo drive reserved parameters (The user cannot write to the reserved parameters, otherwise it may cause the servo drive to run abnormally). If the user needs to modify other parameters, please contact our company technician.

5 Description of the monitored status quantity reads

The state amount inside the servo drive can be read out through the RS-485 communication port, and the write operation is invalid. The state quantity is stored in 16bit data, providing 100 state variables from 0 to 99. The state quantity read can be realized through the following two functional codes:
(1) Function code (04H), address range 0x 0000~0x 0063
(2) Function code (03H), address range 0x 1000~0x 1063

The function code (04H) the address of read status (hex) is as follows:

0x 0000: Motor speed in "r/min";

0x0001: Original position command (input pulse) is 16 bits lower;

0x0002: Original position command (input pulse) is 16 bits high;

0x 0003: Position command (pulse) is low by 16 bits;

0x 0004: Position command (pulse) high of 16 bits;

0x 0005: Current motor position (pulse) low: 16 bits;

0x 0006: current motor position (pulse) 16 bits;

0x 0007: Position deviation (pulse) is low by 16 bits;

0x 0008: Position deviation (pulse) is 16 bits higher;

0x 0009: motor torque, unit "%" (percentage of the rated torque), 12 is 12% of the rated torque;

0x 000A: peak torque, unit "%" (peak torque within 1s); <10>

0x000B: instantaneous current of the motor in unit "0. 1A" (12: means the real-time current of the motor is 1.2A);

0x 000C: Peak current of the motor in "0. 1A" (152: the maximum current in 1s is 15.2A);

0x000D: Position command pulse frequency, in "0. 1kHz" (3000: for 300 kHz);

0x 000E: Speed instruction, in unit "r / min";

0x000F: Torque instruction, unit "%"; <15>

0x 0010: Speed analog command voltage, per unit "mV";

0x 0011: Torque analog command voltage, per unit "mV";

0x 0012: Input terminal DI status, [Note 1];

0x 0013: the DO state of the output terminal, [Note 2];

0x 0014: Motor encoder single coil absolute position (pulse) low 16 bits; <20>

0x 0015: Motor encoder single coil absolute position (pulse) height of 16 bits;

0x 0016: the multi-loop position of the motor encoder (read out the 0 value when there is no multi-loop information);

0x 0017: Regenerative brake load rate, per unit of "%";

0x 0018: Average load rate in "%";

0x0019: Output voltage in "%"; <25>

0x001A: alarm code;

0x 001B: Motor speed in "0. 1r / min";

0x 001C: the second encoder position (pulse) is low by 16 bits;

0x 001D: the second encoder position (pulse) is 16 bits high;

0x 001E: [illegible] "0.01A" (1[illegible]

0x 001F: Absolute position of the motor is 16 bits low (32 bits of multiple circles + absolute position of single circle data splicing) <31>

0x 0020: The absolute position of the motor is 16 bits high (32 bits + absolute position of single circle data splicing) <32>

0x 0021: Keep <33>

0x 0022:16 lower for motor feedback; (relative encoder) <34>

0x 0023:16 bits higher in relative position of motor feedback; (relative encoder) <35>

0x0024: motor feedback 16 bits (relative encoder) <36>

0x0025:16 bits higher (relative encoder) <37>

0x 0026: Drive radiator temperature <38>

0x 0027: module internal temperature <39>

0x 0028: DC bus voltage <40>

0x 0029: Hold <41>

0x 002A: Hold <42>

0x 002B: Hold <43>

0x 002C: Hold <44>

0x 002D: Keep <45>

The function code (03H) the address of read status (hex) is as follows:

0x 1000: Motor speed in "r/min";

0x 1001: Original position command (input pulse) low 16bit;

0x 1002: Original position command (input pulse) is 16bit high;

0x 1003: Position command (pulse) low 16bit;

0x 1004: Position command (pulse) high 16bit;

0x 1005: Current motor position (pulse) low 16bit;

0x 1006: Current motor position (pulse) 16bit;

0x 1007: Position deviation (pulse) low 16bit;

0x 1008: Position deviation (pulse) height of 16bit;

0x 1009: Motor torque, unit "%" (percentage of rated torque);

0x 100A: peak torque, unit "%" (peak torque within 1s); <10>

0x100B: instantaneous current of the motor in "0. 1A" (12: real-time current of the motor 1.2A);

0x 100C: Peak current of the motor in "0. 1A" (152: the maximum current in 1s is 15.2A);

0x100D: Position command pulse frequency, in "0. 1kHz" (3000: for 300 kHz);

0x 100E: Speed instruction, unit "r / min";

0x100F: Torque instruction, unit "%"; <15>

0x 1010: Speed analog command voltage, per unit "mV";

0x 1011: Torque analog command voltage, per unit "mV";

0x 1012: Input terminal DI status, [Note 1];

0x 1013: the DO state of the output terminal, [Note 2];

0x 1014: Motor encoder single coil absolute position (pulse) low 16bit; <20>

0x1015: Motor encoder single coil absolute position (pulse) height 16bit;

0x 1016: the multi-loop position of the motor encoder (read out the 0 value when there is no multi-loop information);

0x 1017: Regenerative brake load rate, per unit of "%";

0x 1018: Average load rate in "%";

0x1019: Output voltage in "%"; <25>

0x101A: alarm code;

0x 101B: Motor speed in "0. 1r / min";

0x 101C: the second encoder position (pulse) is low by 16bit;

0x 101D: second encoder position (pulse) 16bit;

[illegible]

0x 101F: Motor absolute position 16 bits low (32 bits + absolute position of single circle data splicing)

0x 1020: The absolute position of the motor is 16 bits high (multiple circles + 32 bit absolute position of single circle data splicing)

0x 1021: Hold on

0x 1022: Hold on

0x 1023: Keep <35>

0x 1024: Hold on

0x 1025: Hold on

0x 1026: Hold on

0x 1027: Hold <39>

[Note 1]: The data read by this address is 16bit, where bit 7-bit 0 corresponds to the input state of DI 8 to DI 1, "1" means the input high level, "0" means the input low level; bit 15 to bit 8 is the reserved bit.

[Note 2]: The data bit 16bit, where bit 5-bit 0 is the output state of DO 6 to DO 1; "1" is the output high level and "0" is the output low level; bit 15 to bit 6 is the reserved bit.

6 Communication instances

.16 Read status (CMD = 04 H)

The read status function code (04H) can be read in the servo drive. The communication frame format is described as follows:

Note: The number of states per read is up to 8.

The request frame format is as follows:

Address field 1 byte	FC 1 byte	Start address value 2 bytes, high byte in front	Number of read states 2 bytes, high byte in front	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 04	0x00， 0x00	0x00， 0x01 (N)	CRC_L o , CRC _Hi

normal response:

Address field 1 byte	FC 1 byte	Status-occupied number of bytes 1byte	Status value, 2N bytes, high bytes before	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 04	0x 02 (2N)	0xXX， 0xXX， ……	CRC_L o , CRC _Hi

exception response:

Address field 1 byte	FC 1 byte	exception code 1byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 84	0x 01 or 02 or 03 or 04	CRC_L o , CRC _Hi

Examples are as follows:

Host to send a frame:

A DDR	COD E	S TADDR _H	S TADDR _L	R NUM _H	R NUM _L	CRC _L	CRC _H
0x 01	0x 04	0x 00	0x 01	**0x 00**	**0x 02**	0xX X	0xX X

Deliver response frame:

A DDR	CO DE	BYT E _NUM	DATA 1 _H	DATA 1 _L	DATA 2 _H	DATA 2 _L	CRC _L	CRC _H
0x 01	0x 04	**0x 04**	0xX X	0xX X	0xX X	0xX X	0xX X	0xX X

STADDR: Read the state start address

RNUM: Number of states read BYTE _ NUM: Number of bytes occupied by the state read (for example RNUM 2,216-bit data, BYTE _ NUM 2 * 2 = 4 bytes)

DATAX: The state value read back, X represents the serial number, DATA1 is the first data, DATA2 is the second data in the sample sending frame STADDR = 0x0001, RNUM = 0 x 02, indicating the reading of two state data starting from the starting address 0x 0001 (i. e., read the two state values of 0x 0001 and 0x 0002, see the specific meaning in the previous section). BYTE _ NUM = 0 x 04 in the response frame, where the two state values of the read occupy 4 bytes, followed by DATA 1, DATA2 is the address 0x0001 and 0x0002.

6.2 Read Parameters (CMD = 03 H)

All parameters in the drive can be read by reading parameter function code (03H) and the communication frame format is described as follows:

Note: The number of parameters per read is up to 8.

The request frame format is as follows:

Address field 1 byte	FC 1 byte	Start address value 2 bytes, high byte in front	Number of read parameters 2 bytes, high byte in front	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 03	0x00 , 0x00	0x00 , 0x01 (N)	CRC_L o , CRC _Hi

normal response:

Address field 1 byte	FC 1 byte	Parameter occupied by bytes 1 byte	Parameter value, 2N bytes, high bytes first	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 03	0x 02 (2N)	0xXX , 0xXX , ……	CRC_L o , CRC _Hi

exception response:

Address field 1 byte	FC 1 byte	exception code 1byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 83	0x 01 or 02 or 03 or 04	CRC_L o , CRC _Hi

Examples are as follows:

Host to send a frame:

A DDR	COD E	S TADDR _H	S TADDR _L	R NUM _H	R NUM _L	CRC _L	CRC _H
0x 01	0x 03	0x 00	0x 01	**0x 00**	**0x 02**	0xX X	0xX X

Deliver response frame:

A DDR	CO DE	BYT E _NUM	DATA 1 _H	DATA 1 _L	DATA 2 _H	DATA 2 _L	CRC _L	CRC _H
0x 01	0x 03	**0x 04**	0xX X	0xX X	0xX X	0xX X	0xX X	0xX X

STADDR: Read the parameter starting address

RNUM: Number of parameters read BYTE _ NUM: number of bytes of the read parameters (for example RNUM 2,216-bit data, BYTE _ NUM 2 * 2 = 4)

DATAX: The parameter value read back, X represents the serial number, DATA1 is the first parameter, DATA2 is the second parameter, STADDR = 0x01, RNUM = 0 x 02, means to read 2 parameters from the starting address 1 (I read two parameters P-001 and P-002 with addresses 0x01 and 0x02). BYTE _ NUM = 0 x 04 in the response frame, where the two parameter values read occupy 4 bytes, followed by DATA 1, DATA2 is the parameter values at 0x01 and 0x02.

6.3 Write a single parameter (CMD = 06 H)

The parameters listed in the debugging manual can be written by writing the parameter function code (06H). The communication frame format is described as follows:

pay attention to: Function code 06H can only write one parameter at a time. Before writing the parameter, please refer to the parameter instructions to confirm the specific meaning of the parameter. Incorrectly writing the parameter value may cause abnormal operation of the servo drive!

The request frame format is as follows:

Address field 1 byte	FC 1 byte	Parameter number 2 bytes, high byte in front	Parameter values written 2 bytes, high byte in front	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 06	0x00， 0x00	0x 00， 0x64	CRC_L o , CRC _Hi

normal response:

Address field 1 byte	FC 1 byte	Parameter number 2 bytes, high byte in front	Parameter values written 2 bytes, high byte in front	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 06	0x00， 0x00	0x 00， 0x64	CRC_L o , CRC _Hi

exception response:

Address field 1 byte	FC 1 byte	exception code 1byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 86	0x 01 or 02 or 03 or 04	CRC_L o , CRC _Hi

Examples are as follows:

Host to send a frame:

ADDR	CO DE	W ADDR _H	W ADDR _L	VALUE _H	VALUE _L	CRC _L	CRC _H
0x 01	0x 06	0x 00	0x 01	0x 01	0x 02	0xX X	0xX X

Deliver response frame:

ADDR	CO DE	W ADDR _H	W ADDR _L	VALUE _H	VALUE _L	CRC _L	CRC _H
0x 01	0x 06	0x 00	0x 01	0x 01	0x 02	0xX X	0xX X

WADDR: Write the parameter address

VALUE: Value to be written in this example send frame ADDR = 0x01, VALUE = 0x0102, where the value 0x0102 is written to the parameter with address 0 x 01. The response frame and the same send frame.

6.4 Write Multiple Parameters (CMD = 10 H)

The parameter function code (10H) can be written to the debugging manual. the communication frame format is described as follows:

pay attention to: The function code 10H can write multiple parameters at a time (the maximum number of parameters for a single write is 10). Before writing the parameter, please refer to the parameter instructions to confirm the specific meaning of the parameter. Incorrectly writing the parameter value may cause abnormal operation of the servo drive!

The request frame format is as follows:

Address field 1 byte	Function code 1 byte	Starting address value, 2 bytes, High byte in the front	Number of written parameters, 2 bytes, high bytes first (1~122)	Number of bytes occupied by the parameter, 1 byte (0~255)	Parameter values, 2N bytes, High byte in the front	CRC, check code 2 bytes, Low byte in front
0x 01	0x 10	0x00, 0x00	0x00, 0x02 (N)	0x 04 (2N)	0xXX, 0xXX, ……	CRC_Lo, CRC _Hi

normal response:

Address field 1 byte	FC 1 byte	Starting address value, 2 bytes, High byte in the front	Number of written parameters, 2 bytes, high bytes first	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 10	0x00, 0x00	0x00, 0x02 (N)	CRC_L o , CRC _Hi

exception response:

Address field 1 byte	FC 1 byte	exception code 1byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 90	0x 01 or 02 or 03 or 04	CRC_L o , CRC _Hi

Examples are as follows:

Host to send a frame:

A DR	CO DE	SADRH	SADRL	WNU MH	WNUM L	WNU MB	D 1_H	D 1_L	D 2_H	D 2_L	CR CL	C RCH
0x01	0x10	0x00	0x01	**0x00**	**0x02**	**0x04**	0xXX	0xXX	0xXX	0xXX	0xXX	0xXX

Deliver response frame:

A DR	CO DE	SADRH	SADRL	WNU MH	WNUM L	CR CL	CR CH
0x 01	0x 10	0x00	0x01	**0x00**	**0x02**	0xX X	0xX X

SADR: SADRH (high bytes) SADRL (Low bytes)

WNUM: Number of written parameters WNUMH (high bytes) WNUML (low bytes)

WNUMB: Number of bytes of the written parameters (for example, WNUM is 2, That is, 216-bit data, Then WNUMB is 2 ∗ 2 = 4) D 1_H: the high byte value of the first parameter written D 1_L: the low byte value of the first parameter written D 2_H: the high byte value of the second parameter written D 2_L: the low byte value of the second parameter written SADR = 0 x 01 in this sample sending frame, WNUM = 0x02, Represents starting at starting address 1, Write 2 parameters (i. e. two parameters P-001 and P-002 with addresses 0x01 and 0x02).

6. 5 Diagnostic function (CMD = 08 H)

After the host sends the diagnostic frame, the slave (servo drive) responds to the same data.
Note: The diagnostic function can be used to determine if the MODBUS communication is normal.
Host to send a frame:

ADDR	CO DE	DATA _H	DATA _L	CRC _L	CRC _H
0x 01	0x 08	0x 12	0x 34	0xX X	0xX X

Deliver response frame:

ADDR	CO DE	DATA _H	DATA _L	CRC _L	CRC _H
0x 01	0x 08	0x 12	0x 34	0xX X	0xX X

The slave response frame data of the diagnostic command should be exactly the same as the host sending frame data, so the diagnostic command can detect whether the communication is normal. DATA can be arbitrary custom data, for example, if the host sends 0xABCD, the slave response is also 0 xABCD

.66 Save parameters (CMD = 41 H)

The parameters of the drive can be stored in the EEPROM by using the saving parameter function code (41H), and the communication frame format is described as follows:
Note: After sending this command, the servo drive takes some time to save the parameters to the EEPROM
It is suggested that after sending this command, wait for above 5S before performing other operations.

The request frame format is as follows:

Address field 1 byte	FC 1 byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 41	CRC_L o , CRC _Hi

normal response:

Address field 1 byte	FC 1 byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 41	CRC_L o , CRC _Hi

exception response:

Address field 1 byte	FC 1 byte	exception code 1byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0xC1	0x 01 or 02 or 03 or 04	CRC_L o , CRC _Hi

The second way of saving the parameters:

Use the function code (06H) for writing a single parameter, write the value 0x1234 to the address unit 0x1001, and the drive performs the parameter saving operation.

6. 7 Servo enable / turn-off (CMD = 42 H)

Request / normal response:

Address field 1 byte	FC 1 byte	Enable to turn off the sign 1byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 42	0x 55 (enabled) or 0xAA (cut-off)	CRC_L o , CRC _Hi

exception response:

Address field 1 byte	FC 1 byte	exception code 1byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0x C 2	0x 01 or 02 or 03 or 04	CRC_L o , CRC _Hi

The second method of servo-enable / shut-off:

Use the function code (06H) of a single parameter, write to address unit 0x62, write value 0x01 (i. e., parameter P-098 is set to 1), drive enable open; to address unit 0x62, write value 0x00 (i. e., parameter P-098 is set to 0), drive disconnected enable.

.86 Alarm clear (CMD = 43 H)

Request / normal response:

Addres s field 1 byte	FC 1 byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0x 43	CRC_L o , CRC _Hi

exception response:

Address field 1 byte	FC 1 byte	exception code 1byte	CRC, check code 2 bytes, with low bytes in front
0x 01	0x C 3	0x 01 or 02 or 03 or 04	CRC_L o , CRC _Hi

The second alarm clearance method:

Use the function code (06H) for writing a single parameter, write the value 0x1010 to the address unit 0x1004, and the driver performs the alarm clear operation. (Note: Some alarms related to hardware failure, this operation is invalid)

6.9 Set the current position to the absolute zero

Use the function code (06H) writing a single parameter, write the value 0x1111 to the address unit 0x1000, and the driver sets the absolute position of the current motor encoder to zero.

(Note: This feature is only applicable for multi-loop absolute encoder with battery case)

6.10 Communication method to realize point movement operation

Write the function code (06H), write the corresponding value to the address unit 0x 1010 to realize the point operation:

To the address unit 0x1010, write the value 0x1234, drive enables and maintain zero speed state; to the address unit 0x 1010, write the value 0x2222, servo drive motor point forward operation; to the address unit 0x1010, write the value 0x1111, servo drive motor point reverse operation;

To the address unit 0x 1010, write the value 0x0, exit the dot state and break the enable; the dot speed is determined by the parameter P-076 (unit: r / min).

6.11 Communication control method for input of DI

The function code (06H) can be used to write the corresponding value to the address unit of the following parameters.

Number of the parameter P120	myriabit	kilobit	hundreds place	decade	the unit
Corresponding function	continue to have	T CW	TCCW	ACLR	S O N

If the corresponding function bit is set to 1, the function forces ON (valid)

Number of the parameter P121	myriabit	kilobit	hundreds place	decade	the unit
Corresponding function	ZCLA MP	S PD 3	S PD 2	S PD 1	CM ODE

If the corresponding function bit is set to 1, the function forces ON (valid)

Number of the parameter P122	myriabit	kilobit	hundreds place	decade	the unit
Corresponding function	T RQ 2	T RQ 1	CI NV	CZ ERO	CC W

If the corresponding function bit is set to 1, the function forces ON (valid)

Number of the parameter P123	myriabit	kilobit	hundreds place	decade	the unit
Corresponding function	G EAR 2	G EAR 1	EM G	PC	CW

If the corresponding function bit is set to 1, the function forces ON (valid)

Number of the parameter P124	myriabit	kilobit	hundreds place	decade	the unit
Corresponding function	GA IN	CWL	CC WL	PE CLR	I NH

If the corresponding function bit is set to 1, the function forces ON (valid)