EF900 Explosion Proof Pump

MODBUS Communication Instruction

Content

1 Introduction

EF900 supports RS485 MODBUS RTU protocol to communicate with external controlling devices such as HMIs, PCs, or PLCs. The program is designed with a modular approach, ensuring stability and reliability. The Modbus communication stack comprises two layers: the Modbus application layer and the network layer.

Table 1. Currently Supported Commands

Function code	Command text	Description
0x01	Read Coils	Read the status of coil (on/off)
0x02	Read Discrete Inputs	Read Input Status (on/off)
0x03	Read Holding Registers	Read the contents of read/write location
0x04	Read Input Registers	Read the contents of read-only location
0x05	Write Single Coil	Force single coil
0x06	Write Single Register	Preset single Register
0x0F	Write Multiple Coils	Force multiple coils
0x10	Write Multiple Registers	Preset multiple registers
0x11	Gateway Target Device Failed to Respond	Specialized use in conjunction with gateways, indicates no response was received from the target device.
0x17	Read/Write Multiple Registers	Read/Write multiple registers at the same time

2 dbus Protocol

2.1 Modbus Protocol Model

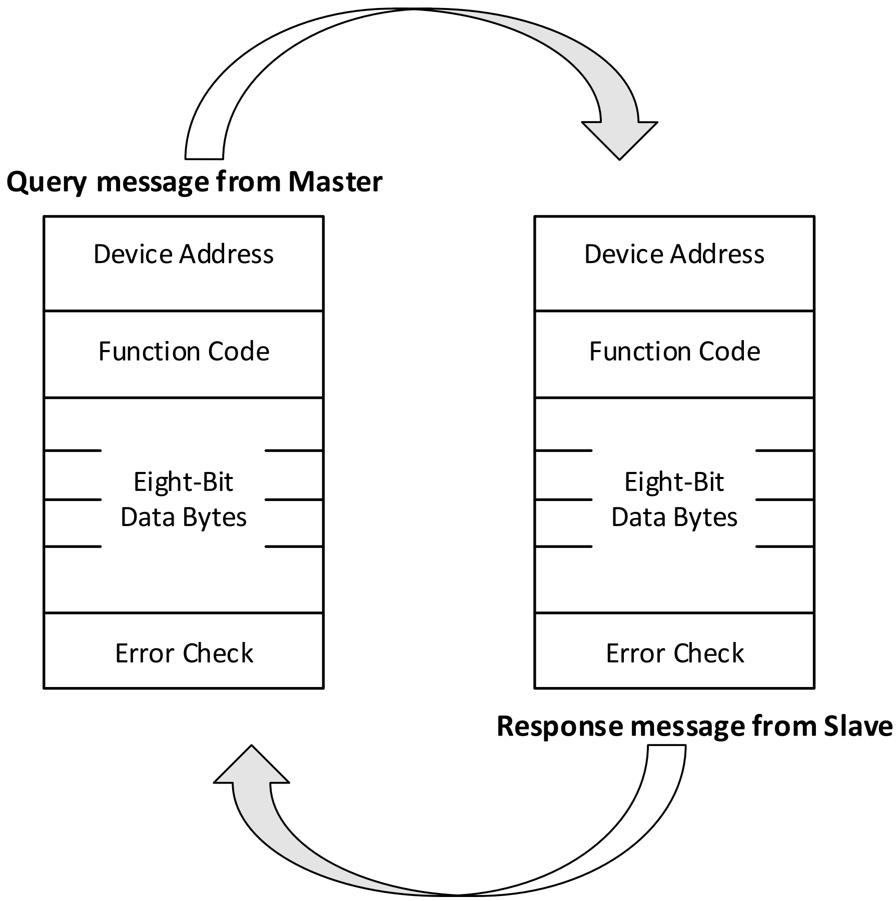


Figure 1. Master–Slave Query–Response Cycle

2.2 Byte Format

- **Coding System**
  8-bit binary, hexadecimal 0-9, A-F. Two hexadecimal characters contained in each 8-bit field of the message.
- **Bits per Byte:**
  1 start bit
  8 data bits, transmitted with the least significant bit first
  1 parity bit for even/odd parity; no parity bit for no parity
  1 stop bit if parity is used; 2 stop bits if no parity
- **Error Check Field**
  Cyclical Redundancy Check (CRC)

2.3 MODBUS Message Timing

In RTU mode, messages start with a silent interval of at least 3.5 character times. This is most easily implemented as a multiple of character times at the baud rate that is being used on the network (shown as T1-T2-T3-T4 in Table 2). The first field then transmitted is the device address. The allowable characters transmitted for all fields are hexadecimal 0-9, A-F. Devices monitor the network bus continuously, including during the “silent” intervals. When the first field (the address field) is received, each device decodes it to find out if it is the addressed device. If the device determines that it is the one being addressed it decodes the whole message and acts accordingly, if it is not being addressed it continues monitoring for the next message. Following the last transmitted character, a silent interval of at least 3.5 character times marks the end of the message. A new message can begin after this interval. The entire message frame must be transmitted as a continuous stream. If a silent interval of more than 1.5 character times occurs before the completion of the frame, the receiving device flushes the incomplete message and assumes that the next byte will be the address field of a new message. Similarly, if a new message begins earlier than 3.5 character times following a previous message, the receiving device will consider it a continuation of the previous message. This will result in an error, as the value in the final CRC field will not be valid for the combined messages. A typical message frame is shown below.

Table 2. RTU Message Frame

START	ADDRESS	FUNCTION	DATA	CRC CHECK	END
T1-T2-T3-T4	8 BTIS	8 BTIS	n x 8 BTIS	16 BITS	T1-T2-T3-T4

2.4 Address Field

The address field of a message frame contains eight bits (RTU). Valid slave device addresses are in the range of 0-247 decimal. The individual slave devices are assigned addresses in the range of 1-247. A master addresses a slave by placing the slave address in the address field of the message. When the slave sends its response, it places its own address in this address field of the response to let the master know which slave is responding. Address 0 is used for the broadcast address, which all slave devices recognize. When Modbus protocol is used on higher level networks, broadcasts may not be allowed or may be replaced by other methods. For example, Modbus Plus uses a shared global database that can be updated with each token rotation.

2.5 Function Field

The function code field of a message frame contains eight bits (RTU). Valid codes are in the range of 1-255 decimal. Of these, some codes are applicable to all xLogic, while some codes apply only to certain models, and others are reserved for future use. When a message is sent from a master to a slave device the function code field tells the slave what kind of action to perform. Examples are to read the ON/OFF states of a group of discrete coils or inputs; to read the data contents of a group of registers; to read the diagnostic status of the slave; to write to designated coils or registers; or to allow loading, recording, or verifying the program within the slave.

When the slave responds to the master, it uses the function code field to indicate either a normal (error–free) response or that some kind of error occurred (called an exception response). For a normal response, the slave simply echoes the original function code. For an exception response, the slave returns a code that is equivalent to the original function code with its most–significant bit set to logic 1. For example, the Read Holding Registers command has the function code 0000 0011 (03H).

If the slave device takes the requested action without error, it returns the same code in its response. However, if an exception occurs, it returns 1000 0011 (83H) in the function code field and appends a unique code in the data field of the response message that tells the master device what kind of error occurred, or the reason for the exception.

The master's application program must handle the exception response. It may choose to post subsequent retries of the original message, it may try sending a diagnostic query, or it may simply notify the operator of the exception error.

2.6 Data Field

The data field is constructed using sets of two hexadecimal digits, in the range of 00 to FF hexadecimal. These can be made from one RTU character, according to the network's serial transmission mode. The data field of messages sent from a master to slave devices contains additional information which the slave must use to take the action defined by the function code. This can include items like discrete and register addresses, the quantity of items to be handled, and the count of actual data bytes in the field. For example, when the master request the slave to read the holding registers (function code 03H), the data includes register address, the quantity of the resisters.

If no error occurs, the data field of a response from a slave to a master contains the data requested. If an error occurs, the field contains an exception code that the master application can use to determine the next action to be taken. The data field can be nonexistent (of zero length) in certain kinds of messages. For example, in a request from a master device for a slave to respond with its communications event log (function code 0B hexadecimal), the slave does not require any additional information.

2.7 CRC Error Checking

Two kinds of error–checking methods are used for standard Modbus networks. The error checking field contents depend upon the method that is being used. When RTU mode is used for character framing, the error checking field contains a 16–bit value implemented as two 8–bit bytes. The error check value is the result of a Cyclical Redundancy Check calculation performed on the message contents.

The CRC field is appended to the message as the last field in the message. When this is done, the low–order byte of the field is appended first, followed by the high–order byte. The CRC high–order byte is the last byte to be sent in the message.

3 Transmit Characters Serially

When messages are transmitted on Modbus serial networks, each character or byte is sent in the order of Least Significant Bit (LSB) to Most Significant Bit (MSB) as outlined below (left to right).

With Parity Checking

Start	1	2	3	4	5	6	7	8	Par	Stop

Without Parity Checking

Start	1	2	3	4	5	6	7	8	Stop	Stop

Figure 2. Bit Order (RTU)

4 Instructions for Use

- During communication debugging, start by testing a single-word input register, such as temperature, and observe if the readout matches the display to confirm normal communication. If the data consistently appears incorrect, try swapping the A and B wiring connections.
- The communication status displayed on the LCD is determined by the transmission and reception of communication data, not by physical connections.
- In this technical standard, unless under special circumstances, it is recommended to use the default byte order CDAB for long integers and floating-point numbers (original communication mode: computer mode). Under byte order ABCD (original communication mode: PLC mode), to ensure data correctness, each time only one instruction is allowed to read or write a long integer or single-precision floating-point variable. In this mode, attempting to read or write two or more consecutive single/double-word registers will result in erroneous data.
- When devices communicate, there must be a delay of more than 50ms between each instruction frame to ensure the integrity of communication data. Otherwise, it may lead to data errors and communication abnormalities. Especially in PLC programming and computer programming, each instruction frame must be manually added.
- When writing to registers for flow rate and liquid volume, ensure to write the unit first and then the value. Otherwise, it may lead to abnormal units and prevent startup.
- Changing the pump number, communication rate, or parity bit requires restarting the device for the changes to take effect.

5 Parameters and Addresses

Mode: RTU
Addresses: 1-247 (User Defined)
Baud Rate: 9600
Data Bits: 8
CRC: Even Parity
Stop Bit: 1

Table 3. Input Registers (Read Only)

Address	Data Type	Function	Mark
5000	Dword	Current rotational speed	unit: RPM
5001			
5002	Dword	Analog conversion speed	unit: RPM
5003			
5004	Dword	The number of steps currently running	
5005			
5006	Dword	The number of steps required to run	
5007			
5008	Dword	Current running time	unit: second
5009			
5010	Dword	Required run time	unit: second
5011			

5012	float	Current dispensing volume	unit: uL
5013			
5014	float	The amount of dispensing fluid required	unit: uL
5015			
5016	word	Working status	
5017	word	Malfunction status	
5018	word	Current motor temperature	unit: °C
5019	word	Motor status	
5020	word	Modbus connection status	
5021	word	WiFi connection status	
5022	word	ADC value	
5023	word	reserved	
5024	Dword	Assign 10ms timing currently	
5025			
5026	Dword	Assign a 10ms timing setting	
5027			
5028	float	Current pump head hose maximum flow rate	unit: ml
5029			
5030	float	Current pump head hose minimum flow rate	unit: ml
5031			
5032	float	Cumulative volume	
5033			
5034	word	Cumulative fluid volume unit	
5035	word	The highest allowed rpm speed for current pump head	
5036	word		
5037	word		
5038	word		
5039	word		
5040	Dword	Cumulative on time of the pump	unit: Sec
5041			
5042	Dword	Cumulative run time of the pump	unit: Sec
5043			
5044	Dword	Accumulate the boot cycles of the pump	
5045			
5046	Dword	Accumulate motor steps	Unit: ustep
5047			
5111			
5112	char*16	Company Information	
5120	char*16	Product Information	

Table 4. Holding Register

Address	**Type**	**Function**	**Range**	**Mark**
System settings				
6000	word	Key tone	off: 0 (factory default); on: 1	

6001	word	Screen lock	unlocked: 0 (default); locked: 1	
6002	word	System language	Chinese Simplified: 0 (default); English: 1	
6003	word	LCD backlight	0-99 (default: 30)	
6004	word	Abnormal motor alarm	off: 0; on: 1 (default)	
6005	word	Dispensing Complete Alarm	0: No alarm; 1-100: When the set percentage is completed, alarm will be on (default 100)	
6006	word	Pump head ID		
6007	word	Tubing ID		
6008	word	Is first use or not	55: Unused; 170: Used	
6009	word	Pump number	1-149 (default: 1)	
6010	word	RS485 bit rate	0: 4800bps; 1: 9600bps (default); 2: 19200bps; 3: 238400bps	
6011	word	RS485 check digit	0: No checksum; 1: odd checksum; 2: even checksum (default)	
6012	word	RS232 bit rate	4800bps 09600bps 1 (production default) 19200bps 238400bps 3	
6013	word	RS232 check digit	No checksum, 0 odd check, 1 even checksum, 2 (production default)	
6014	word	reserved		
6015	word	reserved		
6016	word	Ethernet IP mode	0: Static IP; 1: Dynamic IP. DHCP is enabled, the device must be connected to a DHCP-capable network (for example, a network composed of routers). Restarting the device to apply settings.	
6017	word	Ethernet Subnet mask length	1-24. (Generally set to 24)	
6018	word	Ethernet IP Range 1	IP address 24-31 bits. When a static IP is used, this register is used to set the IP address. When DHCP is used, this register is used to display dynamically assigned IP addresses.	
6019	word	Ethernet IP Range 2	IP address 16-23 bits. The function is the same as address 6018.	
6020	word	Ethernet IP Range 3	IP address 8-15 bits. The function is the same as address 6018.	
6021	word	Ethernet IP Range 4	IP address 0-7 bits. The function is the same as address 6018.	
6022	word	Ethernet gateway Range 1	Set the 24-31 bits of the Ethernet gateway address	
6023	word	Ethernet gateway	Set the Ethernet gateway address to 16-23	

		Range 2	bits	
6024	word	Ethernet gateway Range 3	Set the 8-15 bits of the Ethernet gateway address	
6025	word	Ethernet gateway Range 4	Set the Ethernet gateway address to digits 0-7	
6026	word	reserved		
6027	word	reserved		
6028	word	Administrator password	0-9999 (default: 8888) 0000 indicates no password	
6029	word	Technician password	0-9999 (default 1234) 0000 indicates no password	
6030	word	Operator password	0-9999 (default 0000) 0000 indicates no password	
6031	word	Older versions of Modbus virtual registers	Ignore this setting.	
Task parameter settings				
6032	word	Working mode	0: Flow mode; 1: Volume dispense mode; 2: Time dispense mode; 3: Program mode; 4: External control mode	
6033	word	External control mode signal	0: 0-5V(default); 1: 0-10V; 3: 4-20mA	
6034	word	Start-stop control and status	0: stop; 1: run In Flow mode, to control start / stop. In dispense modes, to show the status of motor, stopped or running.	
6035	word	Direction control and status	0: clockwise; 1: counter-clockwise.	Set the direction of rotation
6036	word	Priming control and status	1: priming; 2: normal speed.	
6037	word	Dispense control and status	0: stopped; 1: dispensing.	
6038	word	Externally start-stop input signal trigger mode	1: Level signal (default); 0: pulse edge.	
6039	word	External direction control input signal trigger mode	1: Level signal (default); 0: pulse edge.	
6040	word	Start-stop input signal level	0: Low level (Falling Edge); 1: High level (Rising Edge, default)	
6041	word	Direction input signal level	0: Low level (Falling Edge, default); 1: High level (Rising Edge)	
6042	word	External control	0: Start and stop state (default)	

		output 1 function setting	1: Direction state 2: Working state 3: Abnormal state	
6043	word	External control output 2 function settings	0: Start-stop state 1: Direction status (default) 2: Working state 3: Abnormal state	
6044	word	External control output 3 function settings	0: Start-stop state 1: Direction state 2: Working state (default) 3: Abnormal state	
6045	word	Reserved		
6046	word	Analog 0V corresponds to rotational speed	Actual value = set value / 10 units rpm	
6047	word	Analog 5V (10V) corresponding to rotational speed	Actual value = set value / 10 units rpm	
6048	word	4mA analog input corresponding to rotation speed	Actual value = set value / 10 units rpm	
6049	word	20mA analog input corresponding to rotational speed	Actual value = set value / 10 units rpm	
6050	float	Calibration coefficient		
6051				
6052	word	Reset calibration coefficient	Set one to automatically reset the calibration coefficient, and the system will automatically reset to zero	
6053	word	Reserved		
6054	word	Suck-back angle	0-720	Unit°
6055	word	Suck-back speed		
6056	word	Assign steps	0: Idle 1: Running 2: Delay	
6057	word	Completed cycle count		
6058	word	Current group in volume mode	0-4	
6059	word	Current group in time mode	0-4	
6060	word	Current step in program mode	1~30	
6061	word	Total steps in	1~30	

		program mode		
6062	word	The programming mode is currently looping	0-999	
6063	word	Total loop count in program mode	0~999	
6064	word	Reserved		
6079	word	Reserved		
6080 6081	float	Flow mode, flow settings or displays	0.001-999.9	
6082	word	Flow mode, flow unit	0: μL/min 1: mL/min 2: L/min	
6083	word	Reserved		
6084	word	Flow mode timed start and stop function	0: Off 1: On	
6085	word	Delayed start time-hours	0–999 hours Set the number of hours for delayed start in flow mode	Set the hour value for the scheduled start in the flow mode
6086	word	Delayed start time-minutes	0–59 minutes Set the number of minutes for delayed start in flow mode	Set the value of the minute for the scheduled start in the flow mode
6087	word	Delayed start time-seconds	0–59 seconds Set the number of seconds for delayed start in flow mode	Set the value of seconds for the scheduled start in traffic mode
6088	word	Timed run in minutes	0: Off 1: On	
6089	word	Timed Run Duration in hours	0–999 hours Set the number of hours for timed run in flow mode	Set the hour value for the scheduled start in the flow mode
6090	word	Timed Run Duration in Minutes	0–59 minutes Set the number of minutes for timed run in flow mode	Set the value of the minute for the

				scheduled start in the flow mode
6091	word	Timed Run Duration in Seconds	0–59 seconds Set the number of seconds for timed run in flow mode	Set the value of seconds for the scheduled start in traffic mode
6092	word	Reserved		
6093	word	Reserved		
6094	word	Reserved		
6095	word	Reserved		
Volume dispense setting group 1				
6096	float	flow rate		
6097				
6098	float	Fluid volume		
6099				
6100	word	Dispensing time	Actual value = set value/10	
6101	word	Interval time	Actual value - set value/10	
6102	word	Flow Unit	0: µL/min 1: mL/min 2: L/min	
6103	word	Volume unit	1: µL 2: mL 3: L	
6104	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6105	word	Interval time unit	0: Seconds 1: Minutes 2: Hours	
6106	word	Number of cycles		
6107	word	Mode	0: Flow–Volume Mode 1: Time–Volume Mode	
6108	word	Reserved		
6109	word	Reserved		
Volume dispense setting group 2				
6110	float	Flow rate		
6111				
6112	float	Fluid volume		
6113				
6114	word	Dispensing time	Actual value = set value/10	
6115	word	Interval time	Actual value - set value/10	

6116	word	Flow unit	0: μL/min 1: mL/min 2: L/min	
6117	word	Volume unit	0: μL 1: mL 2: L	
6118	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6119	word	Interval time unit	0: Seconds 1: Minutes 2: Hours	
6120	word	Number of cycles		
6121	word	Mode	0: Flow–Volume Mode 1: Time–Volume Mode	
6122	word	Reserved		
6123	word	Reserved		
Volume dispense setting group 3				
6124	float	Flow rate		
6125				
6126	float	Fluid volume		
6127				
6128	word	Dispensing time	Actual value = set value/10	
6129	word	Interval time	Actual value - set value/10	
6130	word	Flow unit	0: μL/min 1: mL/min 2: L/min	
6131	word	Volume unit	0: μL 1: mL 2: L	
6132	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6133	word	Interval time unit	0: Seconds 1: Minutes 2: Hours	
6134	word	Number of cycles		
6135	word	Mode	0: Flow–Volume Mode 1: Time–Volume Mode	
6136	word	Reserved		
6137	word	Reserved		
Volume dispense setting group 4				
6138	float	Flow rate		
6139				

6140	float	Fluid volume		
6141				
6142	word	Dispensing time	Actual value = set value/10	
6143	word	Interval time	Actual value - set value/10	
6144	word	Flow unit	0: µL/min 1: mL/min 2: L/min	
6145	word	Volume unit	0: µL 1: mL 2: L	
6146	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6147	word	Interval time unit	0: Seconds 1: Minutes 2: Hours	
6148	word	Number of cycles		
6149	word	Mode	0: Flow–Volume Mode 1: Time–Volume Mode	
6150	word	Reserved		
6151	word	Reserved		
Volume dispense setting group 5				
6152	float	Flow rate		
6153				
6154	float	Fluid volume		
6155				
6156	word	Dispensing time	Actual value = set value/10	
6157	word	Interval time	Actual value - set value/10	
6158	word	Flow unit	0: µL/min 1: mL/min 2: L/min	
6159	word	Volume unit	0: µL 1: mL 2: L	
6160	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6161	word	Interval time unit	0: Seconds 1: Minutes 2: Hours	
6162	word	Number of cycles		
6163	word	Mode	0: Flow–Volume Mode 1: Time–Volume Mode	
6164	word	Reserved		

6165	word	Reserved		
Time dispense setting group 1				
6166	float	Flow rate		
6167				
6168	word	Dispensing time	Actual value = set value/10	
6169	word	Interval time	Actual value - set value/10	
6170	word	Flow Unit	0: µL/min 1: mL/min 2: L/min	
6171	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6172	word	Interval time unit	0: Seconds 1: Minutes 2: Hours	
6173	word	Number of cycles		
6174	word	Reserved		
6175	word	Reserved		
Time dispense setting group 2				
6176	float	Flow rate		
6177				
6178	word	Dispensing time	Actual value = set value/10	
6179	word	Interval time	Actual value - set value/10	
6180	word	Flow Unit	0: µL/min 1: mL/min 2: L/min	
6181	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6182	word	Interval time unit	0: Seconds 1: Minutes 2: Hours	
6183	word	Number of cycles		
6184	word	Reserved		
6185	word	Reserved		
Time dispense setting group 3				
6186	float	Flow rate		
6187				
6188	word	Dispensing time	Actual value = set value/10	
6189	word	Interval time	Actual value - set value/10	
6190	word	Flow Unit	0: µL/min 1: mL/min 2: L/min	

6191	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6192	word	Interval time unit	0: Seconds 1: Minutes 2: Hours	
6193	word	Number of cycles		
6194	word	Reserved		
6195	word	Reserved		
Time dispense setting group 4				
6196	float	Flow rate		
6197				
6198	word	Dispensing time	Actual value = set value/10	
6199	word	Interval time	Actual value - set value/10	
6200	word	Flow Unit	0: μL/min 1: mL/min 2: L/min	
6201	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6202	word	Interval time unit	0: Seconds 1: Minutes 2: Hours	
6203	word	Number of cycles		
6204	word	Reserved		
6205	word	Reserved		
Time dispense setting group 5				
6206	float	Flow rate		
6207				
6208	word	Dispensing time	Actual value = set value/10	
6209	word	Interval time	Actual value - set value/10	
6210	word	Flow Unit	0: μL/min 1: mL/min 2: L/min	
6211	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6212	word	Interval time unit	0: Seconds 1: Minutes 2: Hours	
6213	word	Number of cycles		
6214	word	Reserved		
6215	word	Reserved		
Program mode setting group 1				

6216	float	Flow rate		
6217				
6218	float	Fluid volume		
6219				
6220	word	Dispensing time	Actual value = set value/10	
6221	word	Interval time	Actual value - set value/10	
6222	word	Flow Unit	0: μL/min 1: mL/min 2: L/min	
6223	word	Volume unit	0: μL 1: mL 2: L	
6224	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6225	word	Interval time unit	0: Seconds	
6226	word	Number of cycles		
6227	word	Rotation direction	0: Clockwise 1: Counterclockwise	
6228	word	Mode	0: Flow–Volume Mode 1: Time–Volume Mode	
6229	word	Disable	0: Not Disabled 1: Disabled	
6230	word	Reserved		
6231	word	Reserved		
Sets 2-29 are arranged sequentially				
Program mode setting group 30				
6680	float	Flow rate		
6681				
6682	float	Fluid volume		
6683				
6684	word	Dispensing time	Actual value = set value/10	
6685	word	Interval time	Actual value - set value/10	
6686	word	Flow Unit	0: μL/min 1: mL/min 2: L/min	
6687	word	Volume unit	0: μL 1: mL 2: L	
6688	word	Dispensing time unit	0: Seconds 1: Minutes 2: Hours	
6689	word	Interval time unit	0: Seconds 1: Minutes	

			2: Hours	
6690	word	Number of cycles		
6691	word	Rotation direction	0: Clockwise 1: Counterclockwise	
6692	word	Mode	0: Flow–Volume Mode 1: Time–Volume Mode	
6693	word	Disable	0: Not Disabled 1: Disabled	
6694	word	Reserved		
6695	word	Reserved		

6 Instruction on data transmission format

6.1 Read Input Register

Temperature: Address 1000, L or F series, 1 word length, integer variable
000000-Tx: 01 04 03 E8 00 01 B1 BA
000001-Rx: 01 04 02 00 22 39 29
Read

	HEX
Slave Address	01
function code	04
Register high	03
Register low	E8
Register number high	00
Register number low	01
CRC high bit	B1
CRC low bit	BA

Description: Address 1000 (03E8)
response

	HEX
Slave Address	01
function code	04
Number of bytes	02
Data high	00
Data low	22
CRC high bit	39
CRC low bit	29

Description: Actual return value 34 (00 22)

6.2 Read holding register

Flow: Address 4015, L or F series, 2 bytes, single precision floating point variable, byte order CDAB (original

computer mode)
000032-Tx:01 03 0F AF 00 02 F7 3E
000033-Rx:01 03 04 EF CA 42 65 1E 52
read

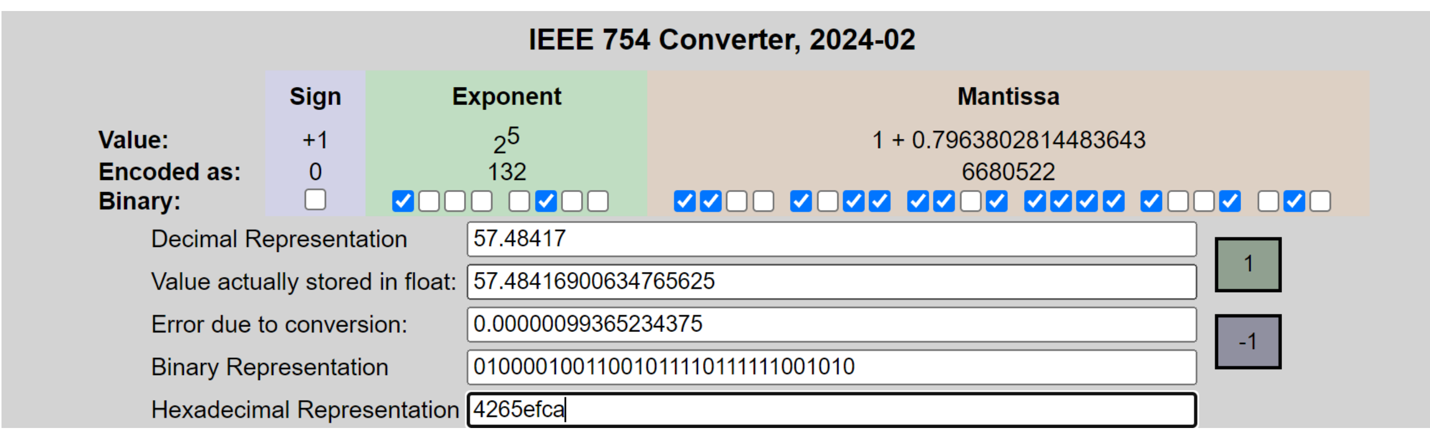
	HEX
slave address	01
function code	03
Register high bit	0F
Register low bit	AF
Register number high bit	00
Register number low bit	02
CRC high bit	F7
CRC low bit	3E

Description: Address 4015 (0F AF)

response

	HEX
slave address	01
function code	03
Number of bytes	04
Data 1 high bit (4015)	EF
Data 1 low bit (4015)	CA
Data 2 high bit (4016)	42
Data 2 low bit (4016)	65
CRC high bit	1E
CRC low bit	52

Note: The actual return value is 57.484169 (42 65 EF CA). The byte order used here is: CDAB (original communication mode: computer mode), and the data format is data 2 + data 1 to form a four-byte hexadecimal number, which needs to be converted to a floating point number.



Website: https://www.h-schmidt.net/FloatConverter/IEEE754.html

6.3 Writing Holding Registers

Start and stop : Address 4126, L or F series, 1 word length, integer variable
Tx:088-01 06 10 1E 00 01 2C CC

Rx:089-01 06 10 1E 00 01 2C CC

Write

	HEX
Slave Address	01
function code	06
Register high	10
Register low	1E
Register number high	00
Register number low	01
CRC high bit	2C
CRC low bit	CC

Description: Address 4126 (10 1E)

response

	HEX
Slave Address	01
function code	06
Register high	10
Register low	1E
Register number high	00
Register number low	01
CRC high bit	2C
CRC low bit	CC

6.4 Writing Holding Registers

Traffic: Address 4015, L or F series, 2 word length, single precision floating point variable, byte order CDAB (original computer mode)

Tx:000082-01 10 0F AF 00 02 04 00 00 41 20 C9 EF

Rx:000083-01 10 0F AF 00 02 72 FD

write

	HEX
slave address	01
function code	10
Register high bit	0F
Register low bit	AF
Register number high bit	00
Register number low bit	02
Number of bytes	04
Data 1 high bit (4015)	00
Data 1 low bit (4015)	00
Data 2 high bit (4016)	41
Data 2 low bit (4016)	20
CRC high bit	C9

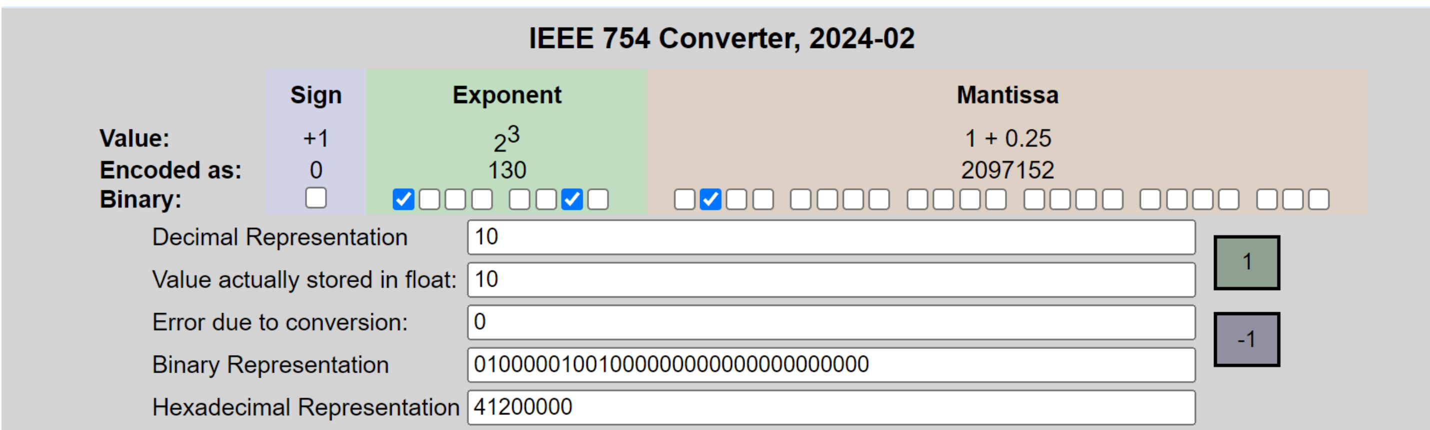
CRC low bit	EF

Description: Address 4015 (0F AF); data 10.0 (41 20 00 00)

response

	HEX
slave address	01
function code	10
Register high bit	0F
Register low bit	AF
Register number high bit	00
Register number low bit	02
CRC high bit	72
CRC low bit	FD

Note: The byte order selected here: CDAB (original communication mode: computer mode), the data format is data 2 + data 1 to form a four-byte hexadecimal number, which needs to be converted to a floating point number



6.5 Read Holding Registers

Flow: Address 4015, L or F series, 2 word length, single precision floating point variable, byte order ABCD (original PLC mode)

000032-Tx:01 03 0F AF 00 02 F7 3E

000033-Rx:01 03 04 42 65 EF CA 32 33

Read

	HEX
Slave Address	01
function code	03
Register high	0F
Register low	AF
Register number high	00
Register number low	02
CRC high bit	F7
CRC low bit	3E

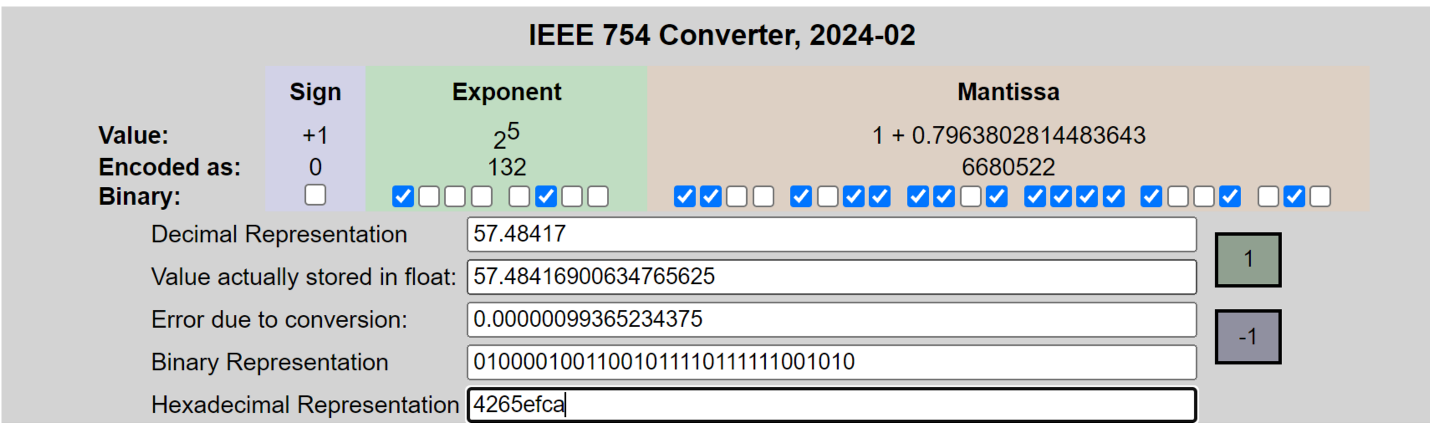
Description: Address 4015 (0F AF)

response

	HEX
Slave Address	01
function code	03
Number of bytes	04
Data 1 high digit (4015)	42
Data 1 low bit (4015)	65
Data 2 high digit (4016)	EF
Data 2 low bit (4016)	CA
CRC high bit	32
CRC low bit	33

Note: The actual return value is 57.484169 (42 65 EF CA).

The byte order used here is: ABCD (original communication mode: PLC mode). Its data sorting order is different from that of the computer mode. The data format is data 1 + data 2, which forms a four-byte hexadecimal number and needs to be converted to a floating point number.



Appendix 1: Solution for converting hexadecimal to single-precision floating point numbers

```
Union
{
float f;
char buf[4];
}data;

void write_Df(u16 ndata, float df)
{
u16 d0,d1;
data.f = df;

d0 = (data.buf[1] << 8) + data.buf[0];
d1 = (data.buf[3] << 8) + data.buf[2];
}
```

Appendix 2: CRC cyclic redundancy check

unsigned char *puchMsg; //To generate CRC value, point the pointer to the buffer containing binary data

```
unsigned short usDataLen; //The number of bytes in the buffer.
unsigned short CRC16(unsigned char *puchMsg, unsigned short usDataLen)
{
      unsigned char uchCRCHi = 0xFF; /* Initialize high byte*/
      unsigned char uchCRCLo = 0xFF; /*Initialize low byte*/
      unsigned uIndex ; /*Put CRC table*/
      while (usDataLen––) /*Pass data buffer*/
         {
         uIndex = uchCRCHi ^ *puchMsg++; /*Calculate CRC */
         uchCRCHi = uchCRCLo ^ auchCRCHi[uIndex];
         uchCRCLo = auchCRCLo[uIndex];
         }
return (uchCRCHi << 8 | uchCRCLo) ;
}
//High digit
/* Table of CRC values for high–order byte */
static unsigned char auchCRCHi[] = {
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40
} ;
//Low Digit
/* Table of CRC values for low–order byte */
static char auchCRCLo[] = {
0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7, 0x05, 0xC5, 0xC4,
0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09,
0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE, 0xDF, 0x1F, 0xDD,
0x1D, 0x1C, 0xDC, 0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32, 0x36, 0xF6, 0xF7,
0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A,
```

```
0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B, 0x2A, 0xEA, 0xEE,
0x2E, 0x2F, 0xEF, 0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1, 0x63, 0xA3, 0xA2,
0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F,
0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8, 0xB9, 0x79, 0xBB,
0x7B, 0x7A, 0xBA, 0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0, 0x50, 0x90, 0x91,
0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,
0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59, 0x58, 0x98, 0x88,
0x48, 0x49, 0x89, 0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83, 0x41, 0x81, 0x80,
0x40
} ;
```