



OEL85 Series
Modbus Communication Protocol
Manual

For product support, visit: www.owon.com.hk/download

Jun. 2025 edition V1.0.0

Copyright © LILLIPUT Company. All rights reserved.

The LILLIPUT's products are under the protection of the patent rights, including ones which have already obtained the patent rights and those which are applying for. The information in this manual will replace all that in the materials published originally.

The information in this manual was correct at the time of printing. However, LILLIPUT will continue to improve products and reserves the rights to change specification at any time without notice.

owon[®] is the registered trademark of the LILLIPUT Company.

Fujian LILLIPUT Optoelectronics Technology Co., Ltd.

No. 19, Heming Road

Lantian Industrial Zone, Zhangzhou 363005 P.R. China

Tel: +86-596-2130430

Fax: +86-596-2109272

Web: www.owon.com.cn

E-mail: info@owon.com.cn

Table of Contents

1. Overview	2
1.1 Configure communication baud rate	2
1.2 Select communication protocol	3
1.3 Set communication address	3
2. Modbus Introduction	5
2.1 Frame format	5
2.1.1 Modbus RTU	5
2.2 Function Code	5
2.2.1 03 (0x03) Read Register	6
2.2.2 06(0x06) Write to a single register	7
2.2.3 16(0x10) Write Register	8
2.2.4 04(0x04) Read Input Register	9
3. Address List	11
3.1 Register Address	11
4. Programming Reference	15
4.1 Communication Model	15
4.2 Number Conversion	15
4.3 Touchscreen Communication Configuration	16

Statement

Copyright of this manual belongs to our company. The information contained in this manual is intended for customers' secondary development or building test systems.

The company shall not be held liable for any errors contained in this manual or damages resulting from the provision, performance, and use of this manual.

Version Revision History

Date	Version	Revised Chapters
2025.06	1.0.0	Complete this manual

1. Overview

To meet users' needs for developing upper computers using configuration software or other software packages, our company has developed standard Modbus communication protocol based on the original design. Therefore, OEL8500 supports both SCPI and Modbus communication protocols. Users should select the appropriate protocol on the display panel before use.

This manual describes the Modbus communication protocol for OEL8500.

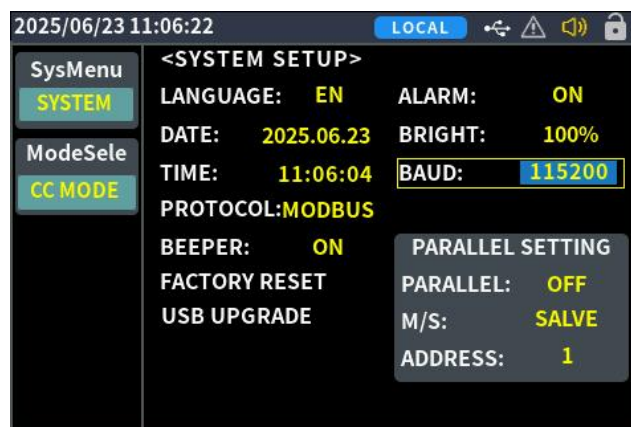
1.1 Configure communication baud rate

The OEL8500 series load supports communication interfaces such as USB (VCP) and RS232.

Interface	Description
USB	USB to Virtual COM Port (VCP)
RS232	Standard Serial Port

The configuration method for communication parameters is as follows:

1. On the front panel, press **config** key, use the knob to select "System Settings", and press the **Enter** key to access the system parameter settings interface.



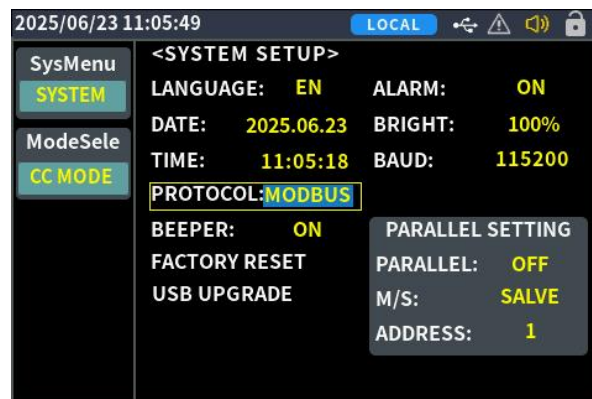
2. Rotate the knob to move the cursor to the "Baud Rate" option, and press the **Enter** key to enter edit mode.
3. Rotate the knob to select relevant parameters.

4. Press the **Enter** key to activate the communication parameters.

1.2 Select communication protocol

The OEL8500 series load supports both SCPI and Modbus communication protocols. The system defaults to SCPI protocol. If you need to switch to Modbus protocol, please follow the steps below:

1. On the front panel, press **config** key, use the knob to select "System Settings", and press the **Enter** key to access the system parameter settings interface.

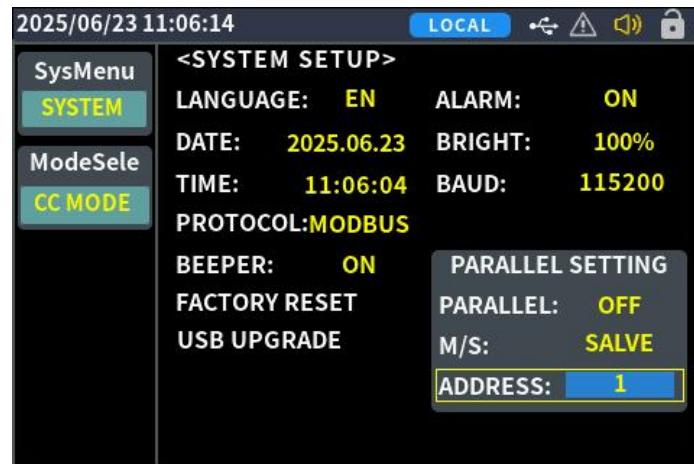


2. Rotate the knob to move the cursor to "Communication Protocol" option, and press the **Enter** key to enter edit mode.
3. Rotate the knob or press the direction keys to select "Modbus".
4. Press the **Enter** key to activate the communication protocol.

1.3 Set communication address

When the user selects Modbus as the communication protocol, further settings for the Modbus communication address are required. Please follow the steps below:

1. On the front panel, press **config** key, use the knob to select "System Settings", and press the **Enter** key to access the system parameter settings interface.



2. Rotate the knob to move the cursor to "Device Address", and press **Enter** to enter edit mode.
3. Rotate the knob or press the number keys to enter a new address (address range: 1~9).
4. Press the **Enter** key to activate the device address.

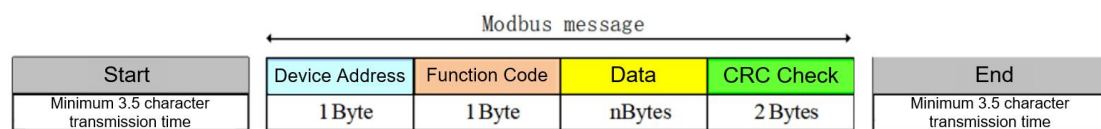
2. Modbus Introduction

The OEL8500 series load supports standard Modbus RTU protocol.

2.1 Frame format

2.1.1 Modbus RTU

For serial communication (e.g., RS232), when using RTU mode to transmit Modbus messages, the format is shown in the following figure.



- **Device Address:** Modbus Communication Address, Range 1~9.
- **Function Code:** Requests a specific operation to be performed, such as reading registers, writing registers, reading block data, etc.
- **Data:** Describes specific operation information, such as the starting address of reading registers and the number of registers to read.
- **CRC Checksum:** Calculate the CRC checksum result for "Device Address", "Function Code", and "Data".

2.2 Function Code

The Modbus protocol of OEL8500 series only uses partial function codes from the 《Standard Modbus Protocol》, as detailed in the following table.

Function Code	Description	Note
0x03	Read Retention Register	Read in 16-bit bytes
0x04	Read Input Register	Read in 16-bit bytes
0x06	Write a holding register	Write in 16-bit bytes
0x10	Write multiple holding registers	Write in 16-bit bytes

2.2.1 03 (0x03) Read Register

This function code is used to read the contents of multiple consecutive registers.

Request

Function Code	1 Byte	0x03
Initial Address	2 Bytes	0x0000~0xFFFF
Register Count	2 Bytes	

Response

Function Code	1 Byte	0x03
Number of bytes returned	1 Byte	2xN
Register Value	N x 2 Bytes	value

Error

Error Code	1 Byte	0x83
Exception Code	1 Byte	01/02/03/04

Exception Code Description

Error Code	0x01	0x02	0x03	0x04
Description	Unsupported Function Code	Register address out of range.	The number of registers exceeds the range.	Read failure

For example

Request: 01 03 03 E8 00 02 44 7B

Response: 01 03 04 00 01 00 00 AB F3

Request		Response	
Filed	Hex	Field	Hex
Device Address	01	Device Address	01
Function Code	03	Function Code	03
Register Address(High Byte)	03	Byte count	04
Register Address (Low Byte)	E8	Register Value (High Byte)	00

Number of Registers(High Byte)	00	Register Value	01
Number of Registers (Low Byte)	02	Register Value	00
CRC Checksum (Low Byte)	44	Register Value (Low Byte)	00
CRC Checksum (High Byte)	7B	CRC Checksum (Low Byte)	AB
		CRC Checksum (High Byte)	F3

2.2.2 06(0x06) Write to a single register

This function code is used to write a single register.

Request

Function Code	1 Byte	0x06
Initial Address	2 Bytes	0x0000~0xFFFF
Register Value	2 Bytes	value

Response

Function Code	1 Byte	0x10
Register Address	2 Bytes	0x0000~0xFFFF
Register Value	2 Bytes	value

Error

Error Code	1 Byte	0x86
Exception Code	1 Byte	01/02/03/04

Exception Code Description

Error Code	0x01	0x02	0x03	0x04
Description	Unsupported Function Code	Register Address	The number of registers exceeds the range, or the number of bytes is incorrect, or the register value is incorrect.	Write failure

For example

Request: 01 06 03 E8 00 01 C8 7A

Response: 01 06 03 E8 00 01 C8 7A

Request		Response	
Filed	Hex	Field	Hex
Device Address	01	Device Address	01
Function Code	06	Function Code	06
Register Address(High Byte)	03	Register Address (High Byte)	03
Register Address(Low Byte)	E8	Register Address (Low Byte)	E8
Number of Registers (High Byte)	00	Number of Registers (High Byte)	00
Number of Registers (Low Byte)	01	Number of Registers (Low Byte)	01
CRC Checksum (Low Byte)	C8	CRC Checksum (Low Byte)	CB
CRC Checksum (High Byte)	7A	CRC Checksum (High Byte)	7A

2.2.3 16(0x10) Write Register

This function code is used to write multiple consecutive registers.

Request

Function Code	1 Byte	0x10
Initial Address	2 Bytes	0x0000~0xFFFF
Register Count	2 Bytes	2
Byte count	1 Byte	4
Register Value	4 Bytes	value

Response

Function Code	1 Byte	0x10
Initial Address	2 Bytes	0x0000~0xFFFF
Register Value	2 Bytes	2

Error

Error Code	1 Byte	0x90
Exception Code	1 Byte	01/02/03/04

Exception Code Description

Error Code	0x01	0x02	0x03	0x04
Description	Unsupported Function Code	Register Address	The number of registers exceeds the range, or the number of bytes is incorrect, or the register value is incorrect.	Write failure

For example

Request: 01 10 03 F4 00 02 04 00 00 44 7A 5B 0B

Response: 01 10 03 F4 00 02 00 7E

Request		Response	
Filed	Hex	Field	Hex
Device Address	01	Device Address	01
Function Code	10	Function Code	10
Register Address(High Byte)	03	Register Address (High Byte)	03
Register Address (Low Byte)	F4	Register Address (Low Byte)	F4
Number of Registers (High Byte)	00	Number of Registers (High Byte)	00
Number of Registers (Low Byte)	02	Number of Registers (Low Byte)	02
Byte count	04	CRC Checksum (Low Byte)	00
Register Value (High Byte)	00	CRC Checksum (High Byte)	7E
Register Value	00		
Register Value	44		
Register Value (Low Byte)	7A		
CRC Checksum (Low Byte)	5B		
CRC Checksum (High Byte)	0B		

2.2.4 04(0x04) Read Input Register

This function code is used to read the contents of multiple consecutive holding registers.

Request

Function Code	1 Byte	0x04
Initial Address	2 Bytes	0x0000~0xFFFF
Register Count	2 Bytes	2

Response

Function Code	1 Byte	0x04
Initial Address	1 Byte	2xN
Register Value	N x 2 Bytes	value

Error

Error Code	1 Byte	0x83
Exception Code	1 Byte	01/02/03/04

Exception Code Description

Error Code	0x01	0x02	0x03	0x04
Description	Unsupported Function Code	Register Address	The number of registers exceeds the range.	Read failure

For example

Request: 01 04 00 96 00 02 91 E7

Response: 01 04 04 AA C6 3D 1B 6A FA

Request		Response	
Field	Hex	Field	Hex
Device Address	01	Device Address	01
Function Code	04	Function Code	04
Register Address (High Byte)	00	Byte count	04
Register Address (Low Byte)	96	Register Value (High Byte)	AA
Number of Registers(High Byte)	00	Register Value	C6
Number of Registers (Low Byte)	02	Register Value	3D
CRC Checksum (Low Byte)	91	Register Value (Low Byte)	1B
CRC Checksum (High Byte)	E7	CRC Checksum (Low Byte)	6A
		CRC Checksum (High Byte)	FA

3. Address List

3.1 Register Address

The holding registers and input registers are shown in Tables 3-1 and 3-2 respectively.

Table 3-1 Retention Register

Address	Function	Data Type	Address Increment
1000	Output control (ON/OFF)	uint16_t	1
1001	Reserve	uint16_t	1
1002	Output voltage setting	float	2
1004	Output current setting	float	2
1006	Current rise slope	float	2
1008	Current fall slope	float	2
1010	Output power setting	float	2
1012	Output resistance setting	float	2
1014	Voltage range selection	uint16_t	1
1015	Current range selection	uint16_t	1
1016	Voltage rise slope	float	2
1018	Voltage fall slope	float	2
1020	Output mode	int16_t	1
1021	Short circuit mode	uint16_t	1
1022	Load voltage	float	2
1024	Current limit	float	2
1026	Dynamic way	uint16_t	1
1027	CC dynamic low current value	float	2
1029	CC dynamic low current time value	float	2
1031	CC dynamic high current value	float	2

1033	CC dynamic high current time value	float	2
1035	CC dynamic current rise slope	float	2
1037	CC dynamic current drop slope	float	2
1039	Remote compensation settings	uint16_t	1
1040	RS485 communication address setting	uint16_t	1
1041	Trigger signal	uint16_t	1
1042	Battery Test - Cut-off discharge voltage	float	2
1044	Battery Test - discharge capacity	float	2
1046	Battery Test - discharge time	float	2
1048	Battery Test - test condition mask	uint16_t	1
1049	Battery Test - Start-Stop control test	uint16_t	1
1050	Serial/Parallel mode selection	uint16_t	1
1051	Number of series/parallel units	uint16_t	1
1052	Master-Slave settings	uint16_t	1
1053	Local address	uint16_t	1
1054	Parallel control	uint16_t	1
1055	Retain	uint16_t	1
1100	OCP starting voltage	float	2
1102	OCP initial current	float	2
1104	OCP step Current	float	2
1106	OCP step delay	float	2
1108	OCP protection voltage	float	2
1110	OCP test maximum current	float	2
1112	Allocate	float	2
1114	Allocate	float	2
1116	OPP starting voltage	float	2

1118	OPP initial power	float	2
1120	OPP step power	float	2
1122	OPP step delay	float	2
1124	OPP protection voltage	float	2
1126	OPP maximum power	float	2
1128	OPP minimum power	float	2
1130	Allocate	float	2
1200	Voltage limit	float	2
1202	Current limit	float	2
1204	OVP	float	2
1206	OCP	float	2
1208	OPP	float	2
1210	Trigger Mode Selection	uint16_t	1

Table 3-2 Input Registers

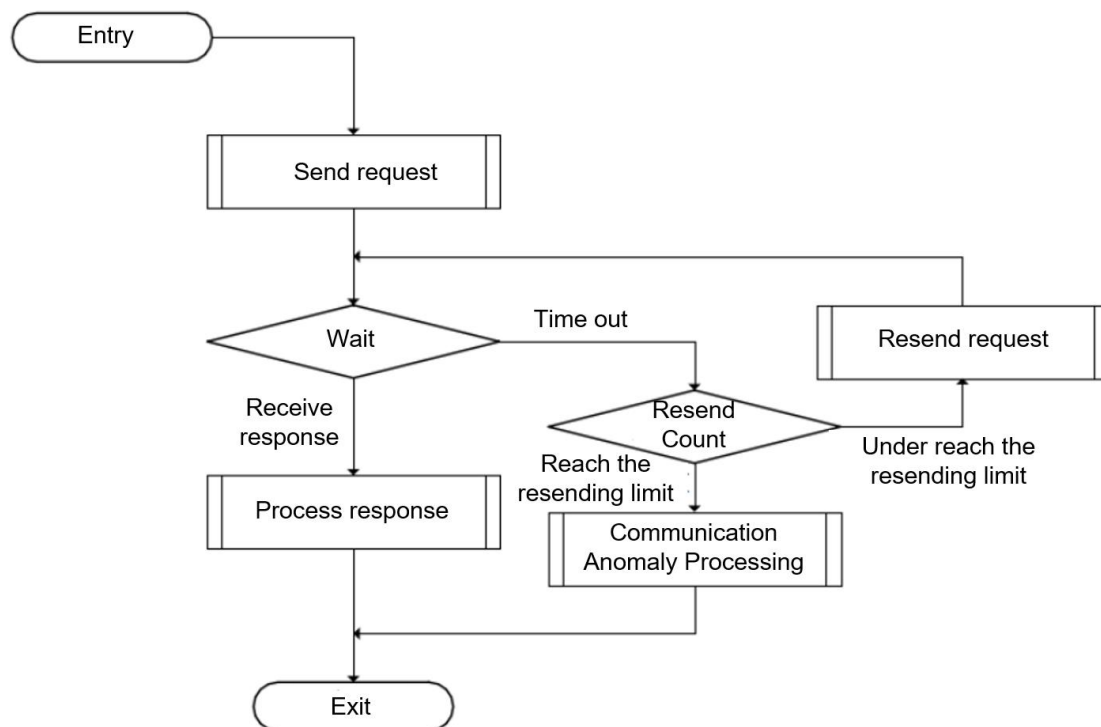
Address	Function	Data Type	Address Increment
150	Output voltage (Local Unit)	float	2
152	Output current (Local Unit)	float	2
154	Output power (Local Unit)	float	2
156	Discharge capacity (Ah)	float	2
158	Electricity discharge(Wh)	float	2
160	Discharge time	float	2
162	Test status	uint16_t	1
163	Allocate	uint16_t	1
164	OCP test value	float	2
166	OCP test status	uint16_t	1
167	Allocate	uint16_t	1
168	OPP test value	float	2

170	OPP test status	uint16_t	1
171	Allocate	uint16_t	1
250	Operational status	uint16_t	1
251	System operation flag	uint16_t	1
252	System fault status word	uint16_t	1
253	Mains input fault	uint16_t	1
254	Internal fault word 1	uint16_t	1
255	Internal fault word 2	uint16_t	1
256	Output error message	uint16_t	1
257	Synchronizing unit error message	uint16_t	1
258	Alarm word	uint16_t	1
259	Allocate	uint16_t	1
260	Allocate	uint16_t	1
261	System operation flag (Latched)	uint16_t	1
262	Allocate	uint16_t	1
263	Mains power failure (Latched)	uint16_t	1
264	Internal fault word 1 (Latched)	uint16_t	1
265	Internal fault word 2 (Latched)	uint16_t	1
266	Output fault word 2 (Latched)	uint16_t	1
267	Standby fault word (Latched)	uint16_t	1
268	Alarm word	uint16_t	1

4. Programming Reference

4.1 Communication Model

The timeout period is recommended to be set between 50 ~ 500ms, depending on the length of the Response frame. The longer the Response frame, the longer the timeout period should be. The number of retries is recommended to be set between 3~5 times.



4.2 Number Conversion

Modbus data transmission uses big-endian format. If the control device (PC, PLC, configuration software, user APP, etc.) uses little-endian format, then the data needs to be converted between big-endian and little-endian formats before use.

typedef union

{

int reg_int; //32-bit signed integer

```

float    reg_float;           //Single-precision floating-point
UCHAR    reg_bytes[4];       //byte sequence
}unReg32Type;
UCHAR buff_modbus[256];      //Modbus Data Buffer
unReg32Type data_reg;        //Composite Register Data

```

On little-endian devices

```

data_reg.reg_bytes[3] = buff_modbus[0];
data_reg.reg_bytes[2] = buff_modbus[1];
data_reg.reg_bytes[1] = buff_modbus[2];
data_reg.reg_bytes[0] = buff_modbus[3];

```

On big-endian devices

```

data_reg.reg_bytes[0] = buff_modbus[0];
data_reg.reg_bytes[1] = buff_modbus[1];
data_reg.reg_bytes[2] = buff_modbus[2];
data_reg.reg_bytes[3] = buff_modbus[3];

```

The above code is for reference only. Users should properly handle data according to the actual situation of the controlled device in communication applications.

4.3 Touchscreen Communication Configuration

Device Property Name	Device attribute value
Collection Optimization	1 ~ Optimization
Device name	Device 0
Device notes	ModbusRTU_serial port
Initial working status	1 ~ start
Minimum acquisition period(ms)	100
Device address	1
Response time	200
16-bit integer byte order	0 ~ 22
32-bit integer byte order	2 ~ 3412
32-bit floating point byte order	2 ~ 3412
Byte Order	0 ~ 21
Character Encoding Format	0 ~ ASCII

Verify Data Byte Order	0 ~ LH(Low byte,High byte)
Function Code Verification	0 ~ Check Function Code
Block Acquisition Method	0 ~ Chunk by maximum length
Write Function Code to Zone 0	2 ~ default value
Write Function Code to Zone 4	2 ~ default value
Maximum block length in Zone 1	120
Maximum block length in Zone 0	120