

# Communication protocol between ELC and HMI

This communication protocol adopts MODBUS protocol. Any operation on PLC data, such as acquisition data from PLC or write data to PLC, and control etc must be in accordance with this communication protocol format, besides connecting hardware and communication parameters setting shall match each other between PLC and HMI, otherwise, PLC cannot normally respond.

## 1.Communication Mode

At present, ELC Controllers can only be setup to communicate on standard Modbus networks using the transmission mode: RTU. Users select this mode, along with the serial port communication parameters (baud rate, parity mode, etc), during configuration of each controller. The mode and serial parameters must be the same for all devices on a Modbus network.

### RTU mode

|         |                  |                |        |     |        |                       |                        |
|---------|------------------|----------------|--------|-----|--------|-----------------------|------------------------|
| Address | Function<br>code | Data<br>Number | Data 1 | ... | Data n | CRC low-order<br>byte | CRC high-order<br>byte |
|---------|------------------|----------------|--------|-----|--------|-----------------------|------------------------|

**PLC mode selection:** MODBUS RTU

**Communication parameter set:**

**Baud rates:** 9600

**Data bit:** 8

**Stop bit:** 1


**Checkout mode:** Non parity checking

The selection of RTU mode pertains only to standard Modbus networks. It defines the bit contents of message fields transmitted serially on those networks. It determines how information will be packed into the message fields and decoded.

On other networks like MAP and Modbus Plus, Modbus messages are placed into frames that are not related to serial transmission.

ELC-RS232 Pin Definition:

| <b>Pin</b> | <b>Contact</b> | <b>Definition</b> |
|------------|----------------|-------------------|
| 1          | N.C.           |                   |
| 2          | <b>TXD</b>     | Send char         |
| 3          | <b>RXD</b>     | Take char         |
| 4          | N.C.           |                   |
| 5          | <b>GND</b>     | <b>GND</b>        |
| 6          | N.C.           |                   |
| 7          | N.C.           |                   |
| 8          | N.C.           |                   |
| 9          | N.C.           |                   |



Notes: Both Pin 4 & Pin 7 cannot be used , otherwise, it would cause communication failure.

## RTU Framing

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 the figure below).The first field then transmitted is the device address.

The allowable characters transmitted for all fields are hexadecimal 0–9, A–F. Networked 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. Following the last transmitted character, a similar 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 3.5 character times occurs before 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 set 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.

|       |         |          |      |           |     |
|-------|---------|----------|------|-----------|-----|
| START | ADDRESS | FUNCTION | DATA | CRC CHECK | END |
|-------|---------|----------|------|-----------|-----|

|             |      |      |          |       |             |
|-------------|------|------|----------|-------|-------------|
| T1-T2-T3-T4 | 8Bit | 8Bit | n ↑ 8Bit | 16Bit | T1-T2-T3-T4 |
|-------------|------|------|----------|-------|-------------|

## How the Address Field is Handled

The address field of a message frame contains two characters (ASCII) or 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.

## How the Function Field is Handled

The function code field of a message frame contains two characters (ASCII) or eight bits (RTU). Valid codes are in the range of 1 – 255 decimal. Of these, some codes are applicable to all ELC controllers, 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 a logic 1.

The master device's application program has the responsibility of handling exception responses. Typical processes are to post subsequent retries of the message, to try diagnostic messages to the slave, and to notify operators.

## 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 a pair of ASCII characters, or 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.

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.

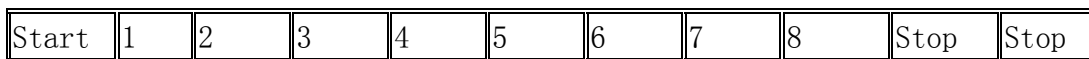
## How Characters are Transmitted Serially

When messages are transmitted on standard Modbus serial networks, each character or byte is sent in this order (left to right):

Least Significant Bit (LSB) . . . Most Significant Bit (MSB)

**With RTU character framing, the bit sequence is:**

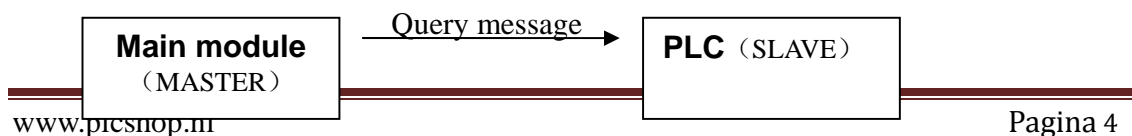
Without Parity Checking



Bit Order (RTU)

## 2 Is PLC master or slave?

In the configuration of our PLC communication, the PLC serves as a slave using Modbus RTU Protocol, and can communicate with a Modbus RTU master directly. That's to say any device communicate with PLC sends command to PLC, the response will be out only when the PLC received the command. As the following figure:



←  
Response message

At present ELC series controller supports baud rate:9600. The default is 9600. The default is non parity checking mode. MODBUS RTU is used as communication protocol of ELC series controller. The defaulted communication protocol is MODBUS RTU format .Defaulted address :1, and legal address range : 1~247.

If users want to modify setting, the controller must serve as a single slave device, and further send corresponding command to modify correlative parameters.

Notes: The Max length of frame command/order which ELC series controller supports is 40 characters (Excluding STX and ETX).

### 3 How is ELC inner register address reflecting in MODBUS protocol:

| Name                          | Code  | Set address method (hex)  | Data format | Attribute |
|-------------------------------|---|---|-------------|-----------|
| Digital quantity input switch | ELC-18(main module):<br>ELC-E-16(extend1):<br>ELC-E-16(extend2):<br>.<br>.<br>.<br>ELC-12(main module): | 10000~1000b<br>1000C~10013<br>10014~1001B<br>.<br>.<br>.<br>10000~10008 | BIT         | R         |
| 4 cursors (Cursor key)        | C   | 10100~10103   | BIT         | R         |
|                               |   |   |             |           |

|   |   |   |              |     |
|---|---|---|--------------|-----|
| Coils outputs                           | ELC-18(main module):<br>ELC-E-16(extend1):<br>ELC-E-16(extend2):<br>.<br>.<br>.<br>ELC-12(main module): | 00000~00005<br>00008~0000F<br>00010~0001F<br>.<br>.<br>.<br>00000~00003 | BIT          | R/W |
| Middle coil                             | M   | 00100~001FF   | BIT          | R   |
| Holding register(timer , counter value) | REG   | 40000~400FF   | LONG         | R   |
|   |   |   |              |     |
| Analog quantity input register          | AI  | 40100~401FF   | Signed short | R   |
| Analog quantity output buffer           | AQ  | 40200~402FF   | Signed short | R   |
| Analog quantity buffer                  | AM  | 40300~403FF   | Signed short | R   |

On the upper table host address range and ELC Max address range are the same, and also different series plc has different address range, hence user shall voluntarily pay more attention to host address range of the PLC being used. In case host address of

communication order/command exceeds the address range of PLC being used, then such PLC would respond to ERROR 4 (illegal address), and simultaneously such command/order would not be executed by PLC being used.

#### 4 Explanation of communication order in detail

The following table contains some communication orders supported by ELC controller.

| Order code | Function description   | Length of message(one frame order can deal with) | Remarks                                  |
|------------|--|--|--|
| 0x01       | Read one group coil status (00000~0XXXX)                         | --   | Read Coil Status (Output relay)          |
| 0x02       | Fetch one group data of the status of switch input (10000~1XXXX) | --   | Read input Status (input relay)          |
| 0x03       | Read data of multi-holding register (40000~4XXXX)                | --   | Read Holding Registers (Output register) |
| 0x05       | Force the switch status of single coil                           | 1  | Force Single Coil                        |

|       |  |      |                            |
|-------|--|------|----------------------------|
|       | (00000~0XXXX)  |      |                            |
| 0x06  | Pre-set the data of single register<br>(40000~4XXXX) | 80   | Set single output register |
| 0x15  | Force multi-coils on/off data<br>(00000~0XXXX)       | many |                            |
| 0x16  | Write multi-holding registers data<br>(40000~4XXXX)  |      |                            |
|       |  |      |                            |
|       |  |      |                            |
| 19~4F | Reserve  |      |                            |

#### RTU Format

Note1: In data field, one byte stands for BIT(1 means ON,0 means OFF). One byte(00 ~FF) would be used to represent "char" type register parameters. The "int" type register parameter can be expressed with two bytes(0000~FFFF). 4 bytes(00 00 00 00 ~ FF FF FF FF) can stand for "long" type register parameter . The high-order byte is appended first, followed by the low-order byte.

Note2:The Max length of command/order message sent to PLC by host can not exceed 80 bytes, otherwise PLC will not execute such order, also without responding to such command/order message, furthermore , host cannot allow the Max length of responding message from PLC to exceed 80 bytes, otherwise, PLC would return to ERROR 3 (command/order cannot be executed.)status.