EASY

Modbus Communication Instructions

Micro Automation

xLogic Micro PLC

Version:2.1

www.PLCshop.nl

Communication protocol between xLogic 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, xLogic 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.

Modbus ASCII also applied to Standard ELC-12& Upgraded ELC-18 series.

RTU mode

Address	Function code	Data Number	Data 1	…	Data n	CRC low-order byte	CRC high-order 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.

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 个 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 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.

The master devices 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:

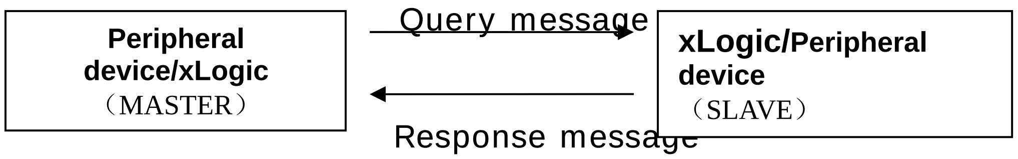
Without Parity Checking

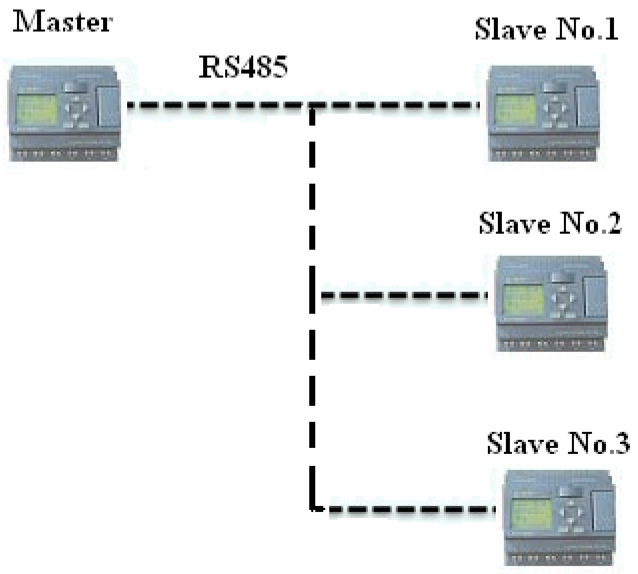
Start	1	2	3	4	5	6	7	8	Stop	Stop

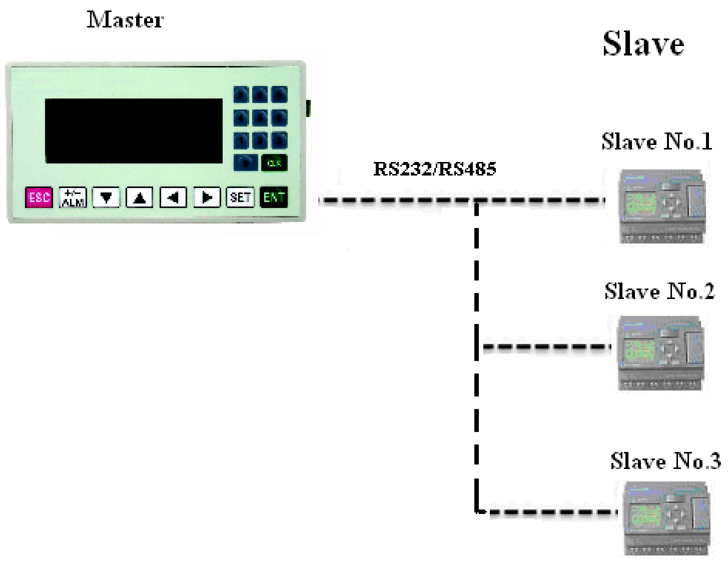
Bit Order（RTU）

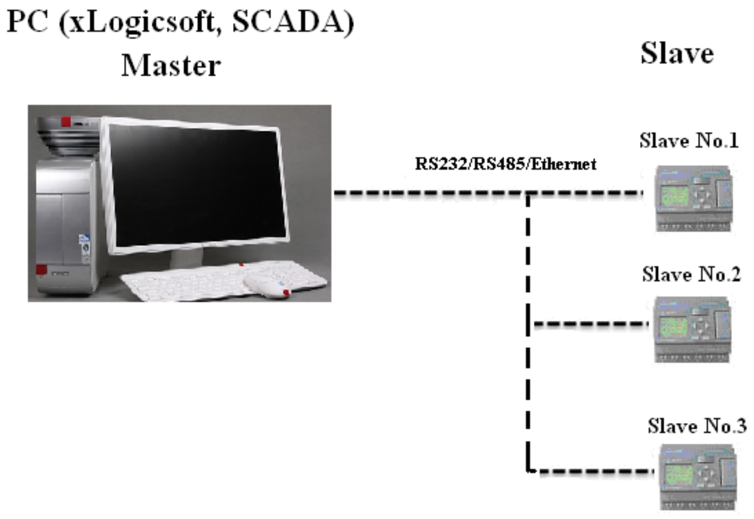
2 It is optional for xLogic to be as a slave or master in Modbus communication network.

As the following figure:


Peripheral device/xLogic
（MASTER）
Query message
Response message
xLogic/Peripheral device
（SLAVE）


Master
RS485
Slave No.1
Slave No.2
Slave No.3


Master
Slave
RS232/RS485
Slave No.1
Slave No.2
Slave No.3

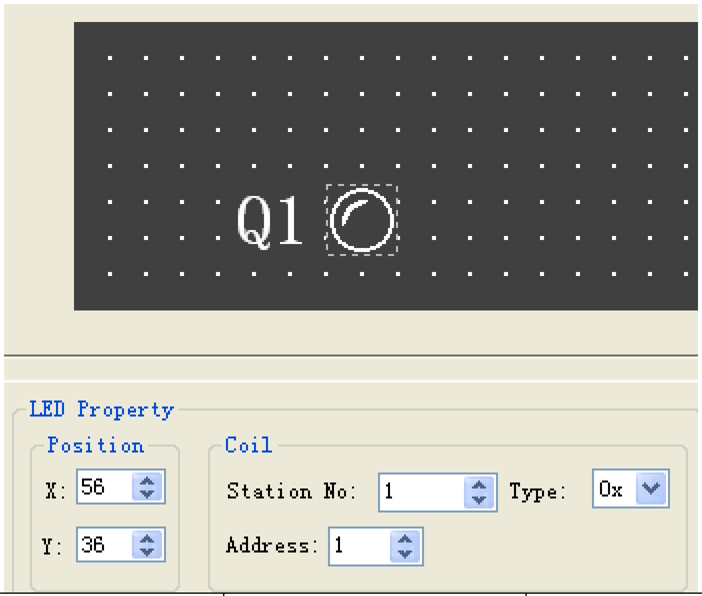


At present xLogic supports baud rate: 9600. The default is 9600. The default is non parity checking mode. MODBUS RTU is used as communication protocol of xLogic. The defaulted communication protocol is MODBUS RTU format .Defaulted address: 1, and legal address range: 1~247.

Notes: 1.The Max length of frame command/order which xLogic supports is 40 characters (Excluding STX and ETX).
2. The address and baud rate of xLogic can be modified via the unit’s keypad.
3. If xLogic serves as master, the blocks F, AF, Modbus Read, and Modbus Write would be used when programming.

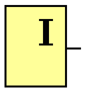
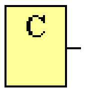
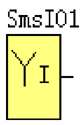
3 xLogic / x-Messenger MODBUS Protocol Memory Map:

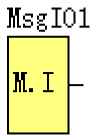
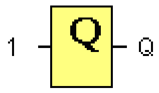
Note： All sorts of register's start address of xLogic is from 0 ,customers should plus 1 if start address is from 1 of third part device. For example in MD204L configuration software the Q1 address should be 0x 1.

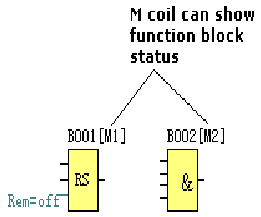
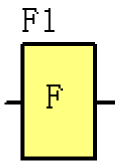


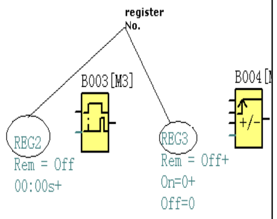
X: 56
Y: 36
Station No: 1
Type: 0x
Address: 1

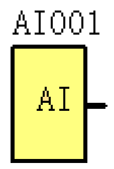
Name	Code	Set address method (DECIMAL)	Data format	Attribute

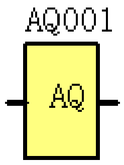
Digital quantity input switch Block in xlogicsoft:  Type: (1x)	ELC-6(CPU): ELC-18(CPU)： ELC-E-16(EXT1)： ELC-E-16(EXT2)： ELC-E-16(EXT3) . . . EXM-12/ ELC-12(CPU)： ELC12-E-8(EXT1) ELC12-E-8(EXT2) ELC12-E-8(EXT3) ELC-22/26(CPU) ELC-E-16(EXT1) ELC-E-16(EXT2) ELC-E-16(EXT3)	0~3 0～11 12～19 20～27 28～35 . . . 0～7 8～15 16～23 24～31 0~16 17~24 25~32 33~40	BIT	R
4 cursors （Cursor key）  (1x)	C	256~259	BIT	R
Sms Input  (1x)	ELC-SMS-D-R (SmsI1-SmsI6)	260~265	BIT	R

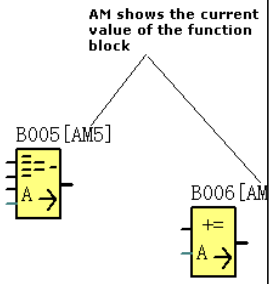
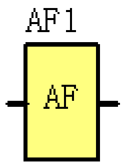
Sms Message Input  (1x)	ELC-SMS-D-R (MsgI1-MsgI10)	266~275	BIT	R
Coils outputs  (0x)	ELC-6(CPU)： ELC-18(CPU)： ELC-E-16(EXT1)： ELC-E-16(EXT2)： ELC-E-16(EXT3)： . . . EXM-12/ELC-12(CPU)： ELC12-E-8(EXT1) ELC12-E-8(EXT2) ELC12-E-8(EXT3) ELC-22/26(CPU) ELC-E-16(EXT1) ELC-E-16(EXT2) ELC-E-16(EXT3) SMS Output SMS Message Output	0~1 0～5 8～15 16～23 24～31 . . . 0～7 8～15 16～23 24～31 0~9 10~17 18~25 26~33 512～515 516～525	BIT	R/W

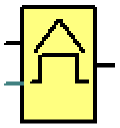
Middle coil (0x)  M coil can show function block status B001[M1] RS Rem=off B002[M2] & (0x)	M	ELC-6&EconomicELC-12 Series: 256~319 Standard EXM-12/ ELC-12 Series: 256~767 Standard/ economic ELC-18 Series: 256~511 Upgraded ELC-18 Series: 256~767 ELC-22/26 256~767	BIT	R
F outputs  (0x)	F	ELC-6&EconomicELC-12 Series: 1536~1567 EXM-12/Standard ELC-12 ： 1536~1599 ELC-18 Series: 768~799 Upgraded ELC-18 Series: 1536~1599 ELC-22/26 1536~1599	BIT	R/W

Holding register(timer、counter value) (4x)  register No. B003[M3] B004 REG2 Rem = Off 00:00s+ REG3 Rem = Off+ On=0+ Off=0 (4x)	REG	ELC-6&EconomicELC-12 Series: 0~63 EXM-12/ ELC-12 Series: 0~511 ELC-18 Series: 0～255 Upgraded ELC-18 Series: 0~511 ELC-22/26 0~511	LONG	R/W

Analog quantity input register  (4x)	AI	EXM-12/ ELC-12 Series: (1024~1279) CPU:1024~ 1031 EXT1:1032~ 1039 EXT2:1040~ 1047 ………… ELC-18 Series: (256~511) CPU:256~263 EXT1:264~ 271 EXT2:272~ 279 ….. Upgraded ELC-18 Series: CPU:1024~ 1031 EXT1:1032~ 1039 EXT2:1040~ 1047 ELC-22/26 (CPU) :1024~ 1031 EXT1:1032~ 1039 EXT2:1040~ 1047	Signed short	R

Analog quantity output buffer  (4x)	AQ	EXM-12/ ELC-12 Series: (1280~1535) CPU:1280~1281 EXT1:1282~1283 EXT2:1284~1285 ELC-18 Series: (512~531) CPU:512~513 EXT1:514~515 EXT2:516~517 ELC-22/26/Upgraded ELC-18 Series: CPU:1280~1281 EXT1:1282~1283 EXT2:1284~1285	Signed short	R/W

Analog quantity buffer  AM shows the current value of the function block B005 [AM5] B006 [AM (4x)	AM	ELC-6&EconomicELC-12 Series: 1536~1599 EXM-12/ ELC-12 Series: 1536~2074 ELC-18 Series: 768~1023 ELC-22/26/Upgraded ELC-18 Series: 1536~2074	Signed short	R
Analog quantity buffer  (4x)	AF	ELC-6&EconomicELC-12 Series: 3072~3103 ELC-12 Series: 3072~3135 ELC-18 Series: 1280~1311 Upgraded ELC-18 Series: 3072~3135	Signed short	R/W

The frequency value buffer of threshold trigger  (4x)	REG	EXM-12/ELC-12 Series: 2560~3071 ELC-18 Series: 1024~1279	Word	R
RTC	Year Month Day Hour Minute Second	All ELC series CPU 3328 3329 3330 3331 3332 3333	Signed short	R/W

On the upper table host address range and xLogic 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.

Note:

1. 10 milliseconds would be regarded as the unit of Time for writing to the HMI.
2. One second would be regarded as the unit of Time for reading from the HMI.
3. The default address of xLogic is 1.
4. The total number of address being accessed should less than the above table showing .

4 Explanation of communication order in detail

The following table contains some communication orders supported by xLogic.

Order code(Hex)	Function description	Length of message(one frame order can deal with)	Remarks
01	Read one group coil status （00000～0XXXX）	--	Read Coil Status (Output relay)
02	Fetch one group data of the status of switch input （10000～1XXXX）	--	Read input Status (input relay)
03	Read data of multi-holding register（40000～4XXXX）	--	Read Holding Registers (Output register)
05	Force the switch status of single coil（00000～0XXXX）	1	Force Single Coil
06	Pre-set the data of single register （40000～4XXXX）	80	Set single output register
15	Force multi-coils on/off data （00000～0XXXX）	many	
16	Write multi-holding registers data（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.