

ICS 25.040
CCS N 10



**NATIONAL STANDARD OF THE
PEOPLE'S REPUBLIC OF CHINA**

GB/T 19582.1-2008

**Modbus industrial automation network specification -- Part 1:
Modbus application protocol
(English Translation)**

Issued on: February 27, 2008

Implemented on: September 1, 2008

**Issued by: General Administration of Quality Supervision, Inspection and
Quarantine of PRC**

Table of Contents

Foreword...	3
Introduction...	5
1 Scope...	6
2 Normative references...	7
3 Abbreviations...	7
4 Background overview...	8
5 General description...	9
5.1 Protocol description...	9
5.2 Data encoding...	11
5.3 Modbus data model...	12
5.4 Modbus addressing model...	13
5.5 Definition of Modbus transaction processing...	14
6 Function code classification...	16
6.1 Definition of common function code...	17
7 Function code description...	18
7.1 01 (0x01) Read coil...	18
7.2 02 (0x02) Read discrete inputs...	21
7.3 03 (0x03) read holding registers...	23
7.4 04 (0x04) Read input registers...	25
7.5 05 (0x05) write single coil...	27
7.6 06 (0x06) Write single register...	28
7.7 07 (0x07) Read exception status (serial link only)...	30
7.8 08 (0x08) Diagnostics (serial link only)...	32
7.9 11 (0x0B) Obtain communication event counter (serial link only)...	38
7.10 12 (0x0C) Obtain communication event log (serial link only)...	39
7.11 15 (0x0F) Write multiple coils...	43
7.12 16 (0x10) Write multiple registers...	45
7.13 17 (0x11) Report slave ID (serial link only)...	47
7.14 20 (0x14) Read file records...	48
7.15 21 (0x15) Write file record...	51
7.16 22 (0x16) Mask write register...	54
7.17 23 (0x17) Read/write multiple registers...	56

7.18 24 (0X18) Read FIFO queue...	58
7.19 43 (0x2B) encapsulate interface transmission...	60
7.20 43/13 (0X2B/0X0D) CANopen Generic reference request and response PDU...	62
7.21 43/14 (0X2B/0X0E) read device identifier...	62
8 Modbus abnormal response...	67
Appendix A (Informative) Modbus reserved function codes, subcodes, MEI types	70
Appendix B (Informative) CANopen Generic reference commands...	71
References...	72

1 Scope

Modbus is an application layer message transmission protocol at Layer 7 of the OSI model. It provides client/server communication between devices connected to different types of buses or networks, as shown in Figure 1.

Since 1979, Modbus has been the de facto standard for industrial serial links, enabling communication between thousands of automated devices. Currently, support for the simple and sophisticated Modbus architecture continues to grow. Internet users can access Modbus using reserved system port 502 on the TCP/IP stack.

Modbus is a request/response protocol, that provides services defined by function codes.

Modbus function codes are elements of a Modbus request/response PDU. This Part describes the function codes used within the Modbus transaction processing framework.

The Modbus application layer message transport protocol is used for client/server communication between devices connected via different types of buses or networks.

Currently, Modbus communication is implemented in the following ways.

- TCP/IP over Ethernet, see GB/T 19582.3.
- Asynchronous serial transmission over various media (wired. EIA/TIA-232-E, EIA-422, EIA/TIA-485-A; fiber optic, wireless, etc.).
- Modbus+, a high-speed token passing network.

Figure 1 -- Modbus communication stack

2 Normative references

The following documents, through reference in this part of GB/T 19582, become provisions of this Part. For dated references, all subsequent amendments (excluding errata) or revisions do not apply to this Part. However, parties agreeing to this Part are encouraged to

investigate the possibility of using the most recent versions of these documents. For undated references, the latest version applies to this Part.

GB/T 15969 Programmable logic controller

RFC 791 Internet Protocol, Sep81 DARPA

3 Abbreviations

ADU (Application Data Unit)

HDLC (High Level Data Link Control)

HMI (Human Machine Interface)

IETF (Internet Engineering Task Force)

I/O (Input/Output)

IP (Internet Protocol)

LSB (Least Significant Bit)

MAC (Medium Access Control)

MB (Modbus Protocol)

MBAP (Modbus Application Protocol)

MEI (Modbus Encapsulated Interface)

MSB (Most Significant Bit)

NAK (Negative Acknowledgment)

PDU (Protocol Data Unit)

PLC (Programmable Logic Controller)

RFC (Request For Comment)

TCP (Transport Control Protocol)

4 Background overview

The Modbus protocol facilitates communication across various network architectures, as shown in Figure 2.

Figure 2 -- Example of a Modbus network architecture

Each device (PLC, HMI, control panel, drive, motion control, I/O device, etc.) can use the Modbus protocol to initiate remote operation.

The same communication is possible over serial links and TCP/IP Ethernet. Gateways enable communication between various buses or networks using the Modbus protocol.

5.1 Protocol description

The Modbus protocol defines a simple protocol data unit (PDU) independent of the underlying communication layer. Modbus protocol mappings on a specific bus or network can introduce additional fields into the Application Data Unit (ADU), as shown in Figure 3.

Figure 3 -- General Modbus frame

Modbus application data units are created by the client initiating Modbus transaction processing. The function code indicates to the server which operation to perform. The Modbus application protocol establishes the request format for client initiation.

The function code field of the Modbus data unit is encoded in one byte. The valid code range is decimal 1 ~ 255 (128 ~ 255 are reserved for exception responses). When a message is sent from the client to the server device, the function code field informs the server which operation to perform. A function code "0" is invalid.

Sub-function codes are added to some function codes to define multiple operations.

The message data field sent from the client to the server device includes additional information that the server uses to perform the operation defined by the function code.

This field also includes discrete values and register addresses, the number of items to be processed, the actual number of data bytes in the field.

In some requests, the data field may be absent (0-length), in which case the server does not need any additional information. The function code only indicates the operation.

If a correctly received Modbus ADU contains no errors related to the requested Modbus function, the server's response data field to the client includes the requested data. If an error related to the requested Modbus function occurs, this field includes an exception code, which the server application can use to determine the next action to take.

For example, the client might be able to read the on/off states of a set of discrete outputs

TCP Modbus ADU = 253 bytes + MBAP (7 bytes) = 260 bytes.

The Modbus protocol defines three types of PDUs. These are.

- Modbus request PDU, mb_req_pdu;
- Modbus response PDU, mb_rsp_pdu;
- Modbus exception response PDU, mb_excep_rsp_pdu.

Define mb_req_pdu as.