

SG series-modbus use detailed instructions

Serial port settings

Baud Rate: 9600, Parity check: none, Data bits: 8, Stop Bit: 1

Function code	Data Types	Attributes
101	uint16_t	Write

Host sends:

Slave address
Function code
Register address high byte
Register address low byte
Register number high byte
Register number low byte
CRC high byte
CRC low byte

The slave responds:

Slave address
Function code
Number of bytes
Data 1 high byte
Data 1 low byte
CRC high byte
CRC low byte

Address	Type of data	Attributes	illustrate
40001	uint16_t	Read	Firmware version

For example, when reading the firmware version, the default value of the slave is 01, which can be modified in the instrument settings - communication.

Note : that this is in hexadecimal.

The command sent is: 01 64 9C 41 00 01 CF 86

01 is the slave address

64 is the function code //function code 100 becomes hexadecimal = 64

9C is the register address high byte

41 is the low byte of the register address

00 is the high byte of the number of registers

01 is the low byte of the number of registers

CF and 86 are CRC

Received: 01 64 02 00 24 XX XX

01 is the slave address

64 is the function code //function code 100 becomes hexadecimal = 64

02 is the number of bytes

00 is data 1 high byte //here 0X0024 is the version number

24 is the low byte of data 1

Function code	Data Types	Attributes
101	uint16_t	Write

Host sends:

Slave address
 Function code
 Register address high byte
 Register address low byte
 Register data high byte
 Register data low byte
 CRC high byte
 CRC low byte

The slave responds:

Slave address
 Function code
 Register address high byte
 Register address low byte
 Register data high byte
 Register data low byte
 CRC high byte
 CRC low byte

Address	Data Types	Attributes	illustrate
40002	uint16_t	read and write	input signal

Input and output signal description

Input						
High byte:	Signal type	1 is the electric current	2 is the voltage	3 is the frequency	4 is the millivolt	5 is the resistor
output						
1 is the electric current	2 is the voltage	3 is the passive current	4 is the frequency	5 is the millivolt	6 is 24V	7 is the resistor
Low byte	Signal mode	Lower four digits	Mode	High four digits	High four digits	

Such as **write input millivolt signal, thermocouple mode, J thermocouple type.**

The command sent was: 01 65 9C 42 04 62 40 AF

01 is the slave address

65 is the function code //Function code 101 becomes hexadecimal=65

9C is the register address high byte
 42 is the low byte of the register address //register address 40002 becomes hexadecimal=9C42
 04 is the high byte of register data //01 is the electric current, 02 is voltage, 03 is frequency, 04 is millivolt, 05 is resistance
 62 is the low byte of register data /*Lower four-bit mode: 1 is millivolt, 2 is thermocouple, 3 is WR thermocouple) */
 /* (The upper four bits are the type: 1 is S, 2 is B, 3 is E, 4 is K, 5 is R, 6 is J, 7 is T, 8 is N) */
 40 and AF are CRC
 Received: 01 65 9C 42 04 62 40 AF
 01 is the slave address
 65 is the function code //function code 101 becomes hexadecimal=65
 9C is the register address high byte
 42 is the low byte of the register address //Register address 40002 becomes hexadecimal = 9C42
 04 is the high byte of register data //1 is the electric current 2 is the voltage 3 is the frequency 4 is the millivolt 5 is the resistor
 62 is the low byte of register data/*Lower four-bit mode: 1 is millivolt, 2 is thermocouple, 3 is WR thermocouple) */
 /* (The upper four bits are the type: 1 is S, 2 is B, 3 is E, 4 is K, 5 is R, 6 is J, 7 is T, 8 is N) */

Input and output signal description						
Input						
High byte:	signal type	1 is the electric current	2 is the voltage	3 is the frequency	4 is the millivolt	5 is the resistor
output						
1 is the electric current	2 is the voltage	3 is the passive current	4 is the frequency	5 is the millivolt	6 is 24V	7 is the resistor

Here the high byte should be easier to understand, so I won't talk about it more, the low byte is more troublesome.

Low byte	Signal mode	Lower four digits	Mode	High four digits	Type
----------	-------------	-------------------	------	------------------	------

The low-order four-bit mode of the low byte is:

The electric current, voltage, passive current, 24V: 1 is the engineering quantity, 2 is the actual value.

Frequency: 1 is frequency, 2 is PWM, 3 is speed, 4 is quantitative.

millivolts: 1 is millivolts, 2 is a thermocouple and ,3 is a WR thermocouple.

Resistance: 1 is Pt100, 2 is Cu50, 3 is resistance.

The high four-bit types of the low byte are:

The electric current,, Voltage, Passive Current, 24V: None.

Frequency: None.

Millivolt: Millivolt is nothing,

Thermocouple is 1 is S, 2 is B, 3 is E, 4 is K, 5 is R, 6 is J, 7 is T, 8 is N,

WR thermocouple is 5 is WRE25 and 6 is WRE26.

Resistor: None.

Address	Data Types	Attributes	Attributes
40008	uint16_t	Read and write	Software output mode and output switch

Software output mode and output switch description:	High byte:	Output mode	Low byte:	Output switch
--	---------------	----------------	--------------	------------------

High byte output mode:			
1 is normal	2 is programmed to automatically lift	3 is the signal linear conversion	4 is preset
Low byte output switch:			
1 is off/off	2 is on/output		

Function code	Data Types	Attributes
102	Float	Read

Host sends:

Slave address

Register address high byte

register address low byte

Register number high byte

Register number low byte //The number of registers here can only
be written to 2

CRC high byte

CRC low byte

The slave responds:

Slave address

Function code

Number of bytes

Data 1 high and high byte

Data 1 high and low bytes

Data 1 low and high byte

Data 1 low-low byte

CRC high byte

CRC low byte

Function code	type of data	Attributes
---------------	--------------	------------

103	float	Write
-----	-------	-------

Host sends:

Slave address

Function code

Register start address high byte

Register start address low byte

Register number high byte

Register number low byte //The number of registers here can only be written to 2

Number of bytes //Only 4 bytes can be written here

Register 1 data high byte

Register 1 data low byte

Register 2 data high byte

Register 2 data low byte

CRC high byte

CRC low byte

The slave responds:

Slave address

Function code

Register address high byte

Register address low byte

Register number high byte

Register number low byte

Number of bytes

CRC high byte

CRC low byte