programmable power supply series **Modbus** protocol communication protocol

Port parameter configuration

版本：VER 04

Baud rate：9600

Data bits：8

Stop bit： 1

Parity check：nothing

Register Definition：

Register Address	name	03 Function code	06 Function code	10 Function code	explain
0x0000	Remote Mode	Y	Y	N	Remote control status 0: Local mode 1: Remote mode
0x0001	V_SET	Y	N	Y	Voltage setting register float type
0x0003	A_SET	Y	N	Y	Current setting register float type
0x001B	OUTPUT	Y	Y	N	Output status setting register 0: Close output 1:Open output
0x001D	V_OUT	Y	N	N	Output actual voltage register float type
0x001F	A_OUT	Y	N	N	Output actual current register float type
0x0021	CV/CC	Y	N	N	Output working status register 0: CV type 1: CC type

Note: A register is a 16 bit (2 bytes) data unit, and all float type parameters occupy two register addresses

The function code table and interval time are as follows：

Function code	Corresponding functions	The interval between two operations
0x03	Command to read one or more registers consecutively	N * 5ms
0x06	Write a single register command	10ms

0x10	Command to write one or more registers consecutively	N * 5ms

Note: N represents the number of registers

The communication protocol format is as follows：

Function code 0x03 (read hold register)

PC sending: 8 Byte

Address	Function code	Starting address High Byte	Starting address Low Byte	Number of registers High Byte	Number of registers Low Byte	CRC16 Verification code Low Byte	CRC16 Verification code High Byte
0x01	0x03						

Power return: 5+N*2 Byte

Address	Function code	Data length bytes	Return data High Byte	Return data Low Byte	Return data N+1 High Byte	Return data N+1 Low Byte	CRC16 Verification code Low Byte	CRC16 Verification code High Byte
0x01	0x03							

Example 1: Reading a single register (output voltage value 0x001D)

PC sending：01 03 00 1D 00 02 54 0D (Starting address 0x001D，Length of 2 registers (float type data)，The CRC16 verification result 0x0D54)

Power return：01 03 04 40 A0 00 00 EF D1 (0x04 bit data length, 0x40A00000 is 5.000V, 0xD1EF is the CRC16 verification result)

Example 2: Reading multiple registers（ Output voltage and current value 0x001D~0x001F)

PC sending：01 03 00 1D 00 04 D4 0F (Starting address 0x001D ，Length is 4 registers, CRC16 checksum result 0x0FD4)

Power return：01 03 08 40 A0 00 00 40 00 00 00 24 2D (0x08 bit data length, 0x40A00000 represents 5.00V, 0x40000000 represents 2.0000A, and 0x2D24 is the CRC16 verification result)

Function code 0x06(Write Holding Register)

PC sending: 8 Byte

Address	Function code	Write address High Byte	Write address Low Byte	Write data High Byte	Write data Low Byte	CRC16 Verification code Low Byte	CRC16 Verification code High Byte
0x01	0x06						

Power return： 8Byte

Address	Function code	Write address High Byte	Write address Low Byte	Write data High Byte	Write data Low Byte	CRC16 Verification code Low Byte	CRC16 Verification code High Byte
0x01	0x06						

Note: All float type parameter registers do not support code 06, which only supports single register operations

Example 1: Remote control status (0x0000)

PC sending：01 06 00 00 00 01 48 0A (Write address 0x0000，Write data 0x0001，CRC16 Verification results 0x480A)

Power return：01 06 00 00 00 01 48 0A (Write address 0x0000，Write data 0x0001，CRC16 Verification results 0x480A)

Example 2: Turn on the power output (0x001B)

PC sending：01 06 00 1B 00 01 38 0D (Write address 0x001B，Write data 0x0001，CRC16 Verification results 0x380D)

Power return：01 06 00 1B 00 01 38 0D (Write address 0x001B，Write data 0x0001，CRC16 Verification results 0x380D)

Function code 0x10(Write multiple holding registers)

PC sending： 9+N*2 Byte

Address	Function code	Starting address High Byte	Starting address Low Byte	Number of registers High Byte	Number of registers Low Byte	Data length byte	Write data High Byte	Write data Low Byte	Write data N+1 High Byte	Write data N+1 Low Byte	CRC16 Verification code Low Byte	CRC16 Verification code High Byte
0x01	0x10											

Power return: 8 Byte

Address	Function code	Starting address High Byte	Starting address Low Byte	Data length High Byte	Data length Low Byte	CRC16 Verification code Low Byte	CRC16 Verification code High Byte

0x01	0x10						

Example 1: Simultaneously setting voltage and current values (0x0001~0x0003)

PC sending：01 10 00 01 00 04 08 40 A0 00 00 40 00 00 00 FA 43 (Starting address 0x0001, Length of 4 registers (8 bytes), 0x40A00000 为 5.00V/40 00 00 00 为 1.0000A, CRC16 Verification results 0x43FA)

Power return：01 10 00 01 00 04 90 0A (Starting address 0x0001, 4 registers,CRC16 Verification results 0x0A90)

CRCR16 Calculate code：

```
const unsigned char  CRCHTalbe[] =

{

    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,

    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,

    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,

    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,

    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,

    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,

    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,

    0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
```

```
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40
}
const unsigned char    CRCLTalbe[] =
{
```

0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7,

0x05, 0xC5, 0xC4, 0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E,

0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09, 0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9,

0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC,

0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,

0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32,

0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D,

0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A, 0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38,

0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF,

0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,

0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1,

0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4,

0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F, 0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB,

0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA,

0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,

0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0,

0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97,

```
    0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C, 0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E,
    0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89,
    0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
    0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83,
    0x41, 0x81, 0x80, 0x40
};
unsigned int crc16(unsigned char *DData,unsigned char len)
{
    unsigned char CRCHi = 0XFF;
    unsigned char CRCLo = 0XFF;
    unsigned int wIndex;
    unsigned int CRC_DData;
    while(len--)
     {
          wIndex = CRCLo ^ *DData++;
          CRCLo = CRCHi ^ CRCHTalbe[wIndex];
          CRCHi = CRCLTalbe[wIndex];
```

```
    }
    CRC_DData = CRCHi;
    CRC_DData<<= 8;
    CRC_DData |=CRCLo;
  return CRC_DData;
}
```

Testing process

Note: After the device is powered on, it defaults to local mode. Before testing MODBUS communication with other commands, it must be configured to remote mode

Number		Host sending	Power recovery	Note
1	Configure to remote mode	01 06 00 00 00 01 48 0A	01 06 00 00 00 01 48 0A	Connecting devices
2	Single write voltage	01 10 00 01 00 02 04 41 40 00 00 27 8B	01 10 00 01 00 02 10 08	
3	Single write current	01 10 00 03 00 02 04 41 00 00 00 A7 86	01 10 00 03 00 02 B1 CB	
4	Continuous writing of voltage and current	01 10 00 01 00 04 08 41 70 00 00 42 0C 00 00 2B F9	01 10 00 01 00 04 90 0A	
5	Read 5 registers continuously (including current, voltage, CC/CV status)	01 03 00 1D 00 05 15 CF	01 03 0A 41 70 00 00 3F FA C0 83 00 00 02 96	New version
6	Read 6 registers continuously (including current, voltage, CC/CV status)	01 03 00 1D 00 06 55 CE	01 03 0C 41 20 00 00 40 3E B8 52 00 00 00 00 BD 0B	Old version
7	Turn on the power	PC sending：01 06 00 1B 00 01 38 0D	01 06 00 1B 00 01 38 0D	

8	Turn off power	PC sending：01 06 00 1B 00 00 F9 CD	01 06 00 1B 00 00 F9 CD	
9	Configure to local mode	PC sending：01 06 00 00 00 00 89 CA	01 06 00 00 00 00 89 CA	Disconnect device