**DPS5005 Digital power communication protocol V1.2**

**I. Introduction to the agreement**

Using RS232, RS485 or Bluetooth serial port transmission interface, communication protocol for the MODBUS-RTU protocol, this product only supports the function code 0x03,0x06,0x10.

**II．Communication protocol introduction**

**The information transfer is asynchronous, and the Modbus-RTU mode is in 11-bit bytes**

|  |  |
| --- | --- |
| Word format (serial data) | 10-bit binary |
| Start bit | 1 bit |
| Data bit | 8 bits |
| Parity bit | none |
| Stop bit | 1 bit |

**Data frame structure:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data frame interval | Address code | Function code | Data area | CRC check |
| 3.5 bytes or more | 1 byte | 1 byte | N bytes | 2 bytes |

Data transmission before the data bus is no time to send data is greater than 3.5 (for example: baud rate of 9600 when the 5ms) message to send at least 3.5 bytes of time to start the pause interval, the entire message frame must be as a continuous Data transfer stream, if there is more than 3.5 bytes of pause before the frame is completed, the receiving device will refresh the incomplete message and assume that the next byte is the address field of a new message. Likewise, if a new message begins with a previous message in less than 3.5 characters, the receiving device will consider it a continuation of the previous message.

**1.1 address code**

The address code is the first byte (8 bits) of each communication message frame, from 1 to 255. This byte indicates that the slave set by the user will receive the information sent by the host. Each slave must have a unique address code, and only the slave code that matches the address code can respond to the loopback message. When the slave sends back the message, the echo data starts with the respective address code. The address code sent by the host indicates the slave address to be sent, and the address code returned by the slave indicates the slave address of the loopback. The corresponding address code indicates where the information came from.

**1.2 function code:**

The function code is the second byte transmitted for each communication message frame. The ModBus communication protocol defines a function code of 1 to 127. As a host request to send, through the function code to tell the slave should be what action. As a slave response, the function code returned by the slave is the same as the function code sent from the host and indicates that the slave has responded to the host and has performed the relevant operation. The unit only supports 0x03, 0x06, 0x10 function code.

|  |  |  |
| --- | --- | --- |
| function code | Definition | Operation (binary) |
| 0x03 | Read register data | Read data from one or more registers |
| 0x06 | Write a single register | Write a set of binary data to a single register |
| 0x10 | Write multiple registers | Write multiple sets of binary data to multiple registers |

**1.3 Data area**

The data area includes what information or what action is required to be returned from the slave, which can be data (eg, digital input / output, analog input / output, register, etc.), reference address, and so on. For example, if the host tells the slave to return the value of the register (including the start address of the register to be read and the length of the read register) via function code 03, the returned data contains the data length of the register and the contents of the data.

**0x03** **Read the function host format**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Address code | Function code | Register start address | Number of register addresses n (1 ~ 32) | CRC check code |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 2 bytes |

**0x03** **Read function Slave returns format**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| address code | function code | Returns the number of bytes 2 \* n | Register data | CRC check code |
| 1 byte | 1 byte | 1 byte | 2 \* n bytes | 2 bytes |

**0x06** **Write a single register function host format**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Address code | Function code | Register address | Register data | CRC check code |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 2 bytes |

**0x06** **Write a single register function from the machine to return the format**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Address code | Function code | Register address | Register data | CRC check code |
| 1 byte | 1 byte | 2 bytes | 2 bytes | 2 bytes |

**0x10** **Write function host format**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Address code | Function code | Register start address | Number of register addresses | Write the number of bytes 2 \* n | Register data | Register data |
| 1 byte | 1 byte | 2 bytes | N (1 ~ 32) | 1 byte | 2 \* n bytes | 2 \* n bytes |

**0x10** **Write function slave return format**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Address code | Function code | Register start address | Number of register addresses | Register data |
| 1 byte | 1 byte | 2 bytes | N (1 ~ 32) | 2 \* n bytes |

**Protocol register description (data in single register address is double-byte data)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Description | Number of bytes | Decimal point | Unit | Read and write | Register address |
| U-SET | Voltage setting | 2 | 2 | V | R/W | 0000H |
| I-SET | Current setting | 2 | 3 | A | R/W | 0001H |
| UOUT | Output voltage display value | 2 | 2 | V | R | 0002H |
| IOUT | Output current display value | 2 | 3 | A | R | 0003H |
| POWER | Output power display value | 2 | 2 | W | R | 0004H |
| UIN | Input voltage display value | 2 | 2 | V | R | 0005H |
| LOCK | Key lock | 2 | 0 | - | R/W | 0006H |
| PROTECT | Protection state | 2 | 0 | - | R | 0007H |
| CVCC | Constant voltage constant current state | 2 | 0 | - | R | 0008H |
| ONOFF | Switch output | 2 | 0 | - | R/W | 0009H |
| B\_LED | Backlight brightness level | 2 | 0 | - | R/W | 000AH |
| MODEL | Product number | 2 | 0 | - | R | 000BH |
| VERSON | Firmware version number | 2 | 0 | - | R | 000CH |
| EXTRACT\_M | Quickly bring up data sets | 2 | 0 | - | W | 0023H |
| U-SET | Voltage setting | 2 | 2 | V | R/W | 0050H |
| I-SET | Current setting | 2 | 3 | A | R/W | 0051H |
| S-OVP | Overvoltage protection | 2 | 2 | V | R/W | 0052H |
| S-OCP | Overcurrent protection value | 2 | 3 | A | R/W | 0053H |
| S-OPP | Over power protection value | 2 | 1、2 | W | R/W | 0054H |
| B-LED | Backlight brightness level | 2 | 0 | - | R/W | 0055H |
| M-PRE | The data is called out of the update output | 2 | 0 | - | R/W | 0056H |
| S-INI | Power on the output switch | 2 | 2 | - | R/W | 0057H |

**Note 1:** This product is designed with M0-M9 10 groups of data sets, each with serial number 10-17 8 data, which M0 data group for the product power default call data group, M1, M2 data group for the product M3-M9 is the ordinary storage array, the starting address of the data group is calculated as: 0050H + data group number \* 0010H, for example, the starting address of the M3 data group is: 0050H +3 \* 0010H = 0080H.

**Note 2:** key lock function read and write values of 0 and 1,0 for non-locking, 1 for the lock.

**Note 3:** The protection status read value is 0-3,0 for normal operation, 1 is OVP, 2 is OCP, 3 is OPP.

**Note 4:** constant voltage constant current state read value of 0 and 1, 0 for the CV state, 1 for the CC state.

**Note 5:** The switch output function reads and writes values of 0 and 1, 0 are off, and 1 is on.

**Note 6:** backlight brightness level read and write range of 0-5,0 level of the darkest, 5 the most bright.

**Note 7:** Quickly call out the data set function to write the value of 0-9, write will automatically call out the corresponding data set data.

**1.4 error check code (CRC check):**

The host or slave can use the check code to determine whether the received information is correct. Due to electronic noise or some other interference, the information in the transmission process sometimes fails. Error check code (CRC) can be checked by the host or slave if the communication data transmission process is wrong. The wrong data can give up (regardless is it sent or received), which increases the security and efficiency of the system. The CRC (redundant cyclic code) of the MODBUS communication protocol contains 2 bytes, that is, 16-bit binary numbers. The CRC code is calculated by the sending device (host) and placed at the end of the transmitted message frame. The device that receives the information (slave) recalculates the CRC of the received message, compares whether the calculated CRC matches the received one, and if the two do not match, it indicates an error. CRC check code sent when the low before the high after the post.

**CRC code calculation method:**

(1) Preset 1 16-bit register is hexadecimal FFFF (ie all 1); this register is called the CRC register;

(2) the first 8-bit binary data (the first byte of the communication information frame) is different from the lower 8 bits of the 16-bit CRC register, and the result is placed in the CRC register;

(3) Move the contents of the CRC register one bit to the right (to the lower) to fill the most significant bit with 0 and check the shifted bit after the right shift;

(4) If the shift bit is 0: repeat step 3 (shift right one again); if the shift bit is 1: The CRC register is XORed with the polynomial A001 (1010 0000 0000 0001)

(5) Repeat steps 3 and 4 until the right 8 times, so that the entire 8-bit data is processed;

(6) Repeat steps 2 through 5 to proceed to the next byte of the communication information frame;

(7) After all the bytes of the communication information frame are calculated as described above, the resulting high and low bytes of the 16-bit CRC register are exchanged;

(8) The final CRC register is the CRC code.

**III, communication examples**

**Example 1: The host reads the output voltage and the output current display value** Host sends the message format:

|  |  |  |  |
| --- | --- | --- | --- |
| Host sent | Number of bytes | Sent information | Remarks |
| Slave Address | 1 | 01 | To the slave at address 01 |
| function code | 1 | 03 | Read register |
| Register start address | 2 | 0002H | Register start address |
| Number of register addresses | 2 | 0002H | A total of 2 bytes |
| CRC code | 2 | 65CBH | The CRC code is calculated by the host |

For example, if the current display value is 05.00V, 5.000A, the slave response returns the message format:

|  |  |  |  |
| --- | --- | --- | --- |
| Slave response | Number of bytes | Returned information | Remarks |
| Slave Address | 1 | 01 | From slave 01 |
| function code | 1 | 03 | Read register |
| Read the number of bytes | 1 | 04 | A total of 1 byte |
| The address is 0002H register contents | 2 | 01F4H | Output voltage display value |
| The address is 0003H register contents | 2 | 1388H | Output current display value |
| CRC code | 2 | B76BH | The CRC code is calculated from the slave |

**Example 2: The host should set the voltage to 24.00V**

Host sends the message format:

|  |  |  |  |
| --- | --- | --- | --- |
| Host sent | Number of bytes | Sent information | Remarks |
| Slave Address | 1 | 01H | From slave 01 |
| function code | 1 | 06H | Write a single register |
| Register address | 2 | 0000H | Register address |
| The address is the contents of the 0000H register | 2 | 0960H | Set the output voltage value |
| CRC code | 2 | 8FB2H | The CRC code is calculated by the host |

After receiving the response from the machine to return the message format:

|  |  |  |  |
| --- | --- | --- | --- |
| Slave response | Number of bytes | Returned information | Remarks |
| Slave Address | 1 | 01H | To the slave at address 01 |
| function code | 1 | 06H | Write a single register |
| Register address | 2 | 0000H | Register start address |
| The address is the contents of the 0000H register | 2 | 0960H | Set the output voltage value |
| CRC code | 2 | 8FB2H | The CRC code is calculated from the slave |

**Example 3: the host to set the voltage of 24.00V, current 1.500A.**

Host sends the message format:

|  |  |  |  |
| --- | --- | --- | --- |
| 主机发送 | 字节数 | 发送的信息 | 备 注 |
| 从机地址 | 1 | 01H | 来自从机01 |
| 功能码 | 1 | 10H | 写寄存器 |
| 寄存器起始地址 | 2 | 0000H | 寄存器起始地址 |
| 寄存器地址数量 | 2 | 0002H | 共2个字节 |
| 写入字节数 | 1 | 04H | 共1个字节 |
| 地址为0000H寄存器的内容 | 2 | 0960H | 设定输出电压值 |
| 地址为0001H寄存器的内容 | 2 | 05DCH | 设定输出电流值 |
| CRC码 | 2 | F2E4H | 由主机计算得到CRC码 |

After receiving the response from the machine to return the message format:

|  |  |  |  |
| --- | --- | --- | --- |
| 从机响应 | 字节数 | 返回的信息 | 备 注 |
| 从机地址 | 1 | 01H | 发送至地址为01的从机 |
| 功能码 | 1 | 10H | 写寄存器 |
| 寄存器起始地址 | 2 | 0000H | 寄存器起始地址 |
| 寄存器地址数量 | 2 | 0002H | 共2个字节 |
| CRC码 | 2 | 41C8H | 由从机计算得到CRC码 |