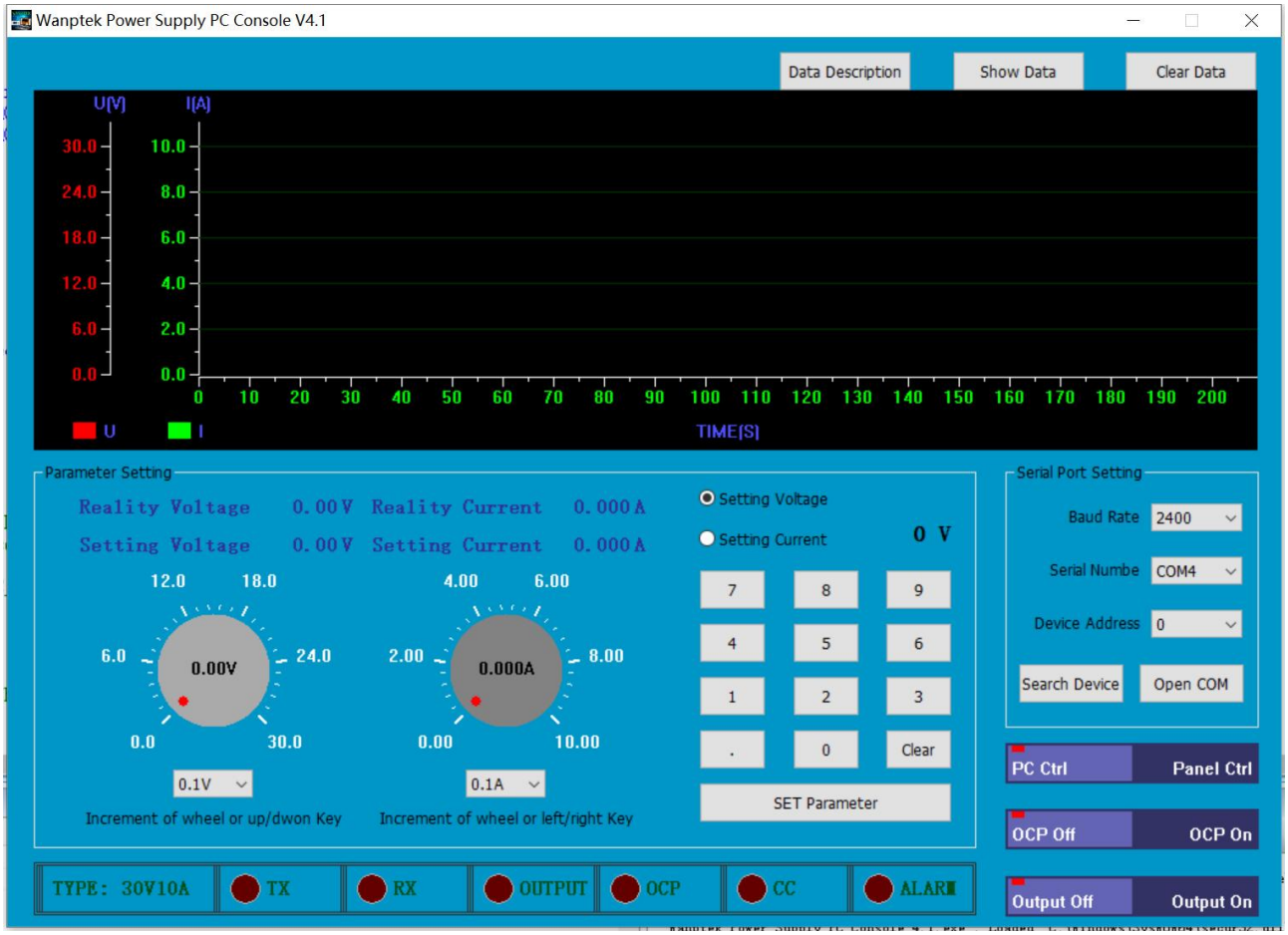
Table of contents

INTRODUCTION

Our company produces various models of "switching type DC regulated power supplies" (hereinafter referred to as "power supplies"). The upper computer (PC or PLC) communicates data with the power supply through RS232 or RS485 serial ports to control the on and off of the power supply, adjust the size of voltage and current, and can also read the operating parameters of the power supply, such as Set the magnitudes of voltage and current, actual voltage, actual current magnitudes, etc.

On our company's website, you can download the upper computer test software "Wanptek Power Supply PC Console.exe (4.1)". This is a green software that does not require installation before use; you can run it directly. The software interface is as follows:



Through this software, parameters such as set voltage, set current, actual voltage, and actual current can be viewed. It also allows control of the power supply's on/off status, OCP, and keyboard lock functions. This is merely a functional demonstration software. Users can, based on their actual needs and the communication protocol provided later in this document, develop their own upper-level control software for their products or control the power supply's operation through devices like PLCs.

In communication, the serial port should be set to 8N1, that is: 8 data bits, no parity bit, and 1 stop bit. Communication is carried out using the MODBUS-RTU protocol commonly used in industrial control, supporting four instructions: reading registers (0x03), writing registers (0x10), reading coils (0x01), and writing coils (0x05).

Chapter 1 Registers and coils

Registers and coils are terms used in PLC control technology. In a PLC, one register occupies 2 bytes (16 bits) and is used as an unsigned integer. The data range is from 0 to 65535, which is used to represent the magnitude of voltage or current values. Depending on the decimal point positions of voltage and current (the decimal point of the voltage value has 1 or 2 valid digits, while the decimal point of the current value has 2 or 3 valid digits). When used for voltage display, it can represent 0.00V - 655.35V or 0.0V - 6553.5V. When used for current display, it can represent 0.000A - 65.535A or 0.00A - 655.35A.

In the original PLC control technology, the term "coil" actually referred to a relay. Nowadays, in PLCs or single-chip microcontrollers, it is represented by a single data bit (bit). Generally, one coil (or one bit) represents a function. When reading the value of this coil, 0 or 1 respectively indicate whether the function is enabled or not. When writing to this coil, 0 and 1 respectively specify disabling or enabling the function.

1.1 The data sequence during transmission

In a PLC, a register consists of two bytes. If two devices need to exchange the data of a register, then there is a problem of which byte is sent first. According to the MODBUS protocol, the address of the register and the field for transmitting data length are transmitted in the order of "first high then low" (as shown in the "Starting Address" and "Data Length" fields in the following table).

	SlaveAddr.	FuncCode	StartAddr.		Data Length		Byte number	Data	CRC	
Byte	0	1	2	3	4	5	6	7-12	13	14
Data	0x00-0x1F	0x10	AddrH	AddrL	LenghH	LenghL	0x06	……	CRCL	CRCH

The sequence of the data segment is determined by the user. The power supply produced by our company is set to the little-endian structure by default when it is manufactured. That is, the low bytes are transmitted first, followed by the high bytes. If you need to change it, please refer to the "Serial Port Data Big/Little Endian Setting" section in Chapter 3 "Power Supply Parameter Settings". For example, the data "1600" is represented in hexadecimal as 0x0640. If you choose "Little-Endian Structure (Parameter Setting 0)", then 0x40 is transmitted first, followed by 0x06. If you choose "Big-Endian Structure (Parameter Setting 1)", then 0x06 is transmitted first, followed by 0x40.

1.2 Register structure

In the standard MODBUS protocol, each register has a specific address. When reading, the data is retrieved from this address, and when writing, it is also written to the same address. However, in this power supply, in order to be able to continuously read or write multiple registers, the register addresses during reading and writing are different. When designing your own communication program,

please pay attention to the distinction. The following table lists the register address structures for reading and writing respectively:

Register structure during instruction reading		Register structure during instruction writing	
Register address	Description	Register address	Description
0x0000	status register 0	0x0000	Control register
0x0001	status register 1	0x0001	Set voltage value
0x0002	Current voltage value	0x0002	Set current value
0x0003	Current current value		
0x0004	Set voltage value		
0x0005	Set current value		
0x0006	Output voltage value(Max)		
0x0007	Output current value(Max)		

During reading, the first two registers (the status register 0 and the status register 1) store certain settings of the power supply. A certain bit or several bits in them represent a certain setting of the power supply. For ease of operation, the status register 0 and the status register 1 are transmitted in a fixed order, unaffected by the parameters in the "serial data endianness setting", always transmitting the high byte (the high 8 bits, namely 15-8 bits) first, and then the low byte (the low 8 bits, namely 7-0 bits).

These setting bits are organized together in the form of registers and are read in the form of reading the registers. Among them, some commonly used status bits can also be read separately in the form of reading coils (MODBUS function code 0x01), such as the on/off status of the power supply.

During writing, the first register is called the control register. Some of the bit variables within it can control the operating state of the power supply, such as whether to start the output, whether to lock the panel buttons, whether to enable the OCP function, etc.

These control bits are also organized together in the form of registers and are written in the form of writing to the register. Moreover, these bits can also be written separately in the form of writing a coil (MODBUS function code 0x05).

1.3 Register structure during reading

1: Status register 0

The 16 bits in the Status Register 0 have their own meanings. Below are the explanations for them.

bit	15	14	13	12	11	10	9	8
Description	-	-	AS	WS	BL	LS	OS	PS
bit	7	6	5	4	3	2	1	0
Description	VID			VPOD	VID			

Bit 13 - AS(Status of alarm):

0: The power supply is not currently in an alarm state.

1: The power supply is currently in an alarm state.

Bit 12 - WS(Status of Work):

0: The power supply is currently in a constant voltage mode (or has not started outputting).

1: The power supply is currently in constant current mode.

Bit 11 - BL(Big-endian Or Little-endian):

0: The voltage and current data are transmitted in little-endian format. The low bytes are transmitted first, followed by the high bytes.

1: The voltage and current data are transmitted using big-endian format. The high byte is transmitted first, followed by the low byte.

Note: The big-endian or little-endian structure of the data is set in the "Power Parameter Settings" (refer to Chapter 3), and this setting only applies to the data registers such as voltage and current, while the transmission sequence of the status register 0 and status register 1 is not affected by this setting and always transmits the high bytes first and then the low bytes.

Bit 10 - LS(Status of lock):

0: The buttons and knobs on the power panel are not locked and can be operated.

1: The buttons and knobs on the power panel are locked and cannot be operated.

Bit 9 - OS(Status of OCP):

0: The output current protection function is in a deactivated state.

1: The output current protection function is activated. When the load current exceeds the set current, the power supply will stop outputting and enter an alarm state.

Bit 8 - PS(Status of power out):

0: The output current protection function is currently turned off.

1: The power output is in the startup state.

Bit 7-5 + Bit 3-0 - VID(Position of the decimal point in the voltage data):

These status bits indicate the voltage series to which the power supply belongs. The data meanings are as follows:

Data	Description	Data	Description
0x00	15V	0x07	200V
0x01	30V	0x08	300V
0x02	60V	0x09	400V
0x03	100V	0x0A	500V
0x04	120V	0x0B	600V
0x05	150V	0x0C	800V
0x06	160V	0x0D	1000V

Bit 4 - VPOD(Position of the decimal point in the voltage data):

0: The voltage data has 2 decimal places of significant figures (the number 1000 represents 10.00V).

1: The voltage data has one decimal place of accuracy (the number 1000 represents 100.0V).

2: Status register 1

The 16 bits in the Status Register 1 have their own meanings. Below are the explanations for them.

bit	15	14	13	12	11	10	9	8
Description	IID			IPOD	IID			
bit	7	6	5	4	3	2	1	0
Description	-	-	-	-	-	-	-	-

Bit 15-13 + Bit 11-8 - IID(Position of the decimal point in the current data):

These status bits indicate the current series to which the power supply belongs. The data meanings are as follows:

Data	Description	Data	Description	Data	Description
0x00	1A	0x07	30A	0x0E	200A
0x01	2A	0x08	40A	0x0F	400A
0x02	3A	0x09	50A	0x20	15A
0x03	5A	0x0A	60A		
0x04	6A	0x0B	80A		
0x05	10A	0x0C	100A		
0x06	20A	0x0D	150A		

Bit 4 - IPOD(Position of the decimal point in the voltage data):

0: The voltage data has 3 significant digits after the decimal point (the number 1000 represents 1.000A).

1: The voltage data has two decimal places after the decimal point (the effective value number 1000 represents 10.00A).

3: Other register

Other registers include actual voltage, actual current, set voltage, set current, and the maximum voltage and current supported by the power supply, totaling six registers. They are double-byte unsigned integers with a data range of 0-65535. After the data is read, based on the VID and IID identifiers in the status register, the decimal point position is determined, and the actual value is finally obtained. For example, the read data is 1234:

If it is a voltage value, according to the VID identifier, if it is 0, it represents 12.34V, and if it is 1, it represents 123.4V.

If it is the current value, according to the IID identifier, if it is 0, it represents 1.234A, and if it is 1, it represents 12.34A.

1.4The register structure during writing

1: Control register

The 16 bits in the control register each have their own meaning. By writing 0 or 1, you can turn

off or enable a certain function.

bit	15	14	13	12	11	10	9	8
Description	-	-	-	-	-	LOCK	OCP	OUTCTRL
bit	7	6	5	4	3	2	1	0
Description	-	-	-	-	-	-	-	-

Bit 10 - LOCK(Lock keyboard):

0: Unlock the power panel and enable operation.

1: Lock the power panel for 2 seconds. During this period, no operation is allowed. If any reading or writing action occurs within 2 seconds, the time will be extended by 2 seconds. If there is no reading or writing action within 2 seconds, the panel will be automatically unlocked, allowing the buttons or knobs on the panel to control the power.

Bit 9 - OCP(Output current protect):

0: Disable the OCP function. When the load current reaches the current limit, it will enter the constant current mode.

1: Enable the OCP function. When the load current reaches the current limit, it will enter the power-off and alarm mode.

Bit 8 - OUTCTRL(Output control):

0: Prohibit power output.

1: Start the power output.

2: Other registers

Two registers for setting voltage and setting current are provided, using double-byte unsigned numbers. When writing, please select the appropriate byte order (please refer to Chapter 3 "Power Parameter Settings" "Serial Port Data Big/Little Endian Setting"). Based on the VID and IID identifiers in the status register, confirm the data to be written. If the power voltage needs to be set to 12V, for products with 2 decimal places, write 1200 (0x04B0) in the register at address 0x0001, and for products with 1 decimal place, write 120 (0x0078).

1.5 Coil Description

The coil represents a certain function, and its attributes are divided into read-write and read-only types. The coil with read-write attributes indicates that it can not only read its state to obtain whether the function is enabled or not, but also enable or disable the function by writing 1 or 0. The coil with read-only attributes cannot be written.

When reading or writing to the coils, only one coil's value can be read or written at a time. It is important to note that the address must not exceed the specified range. When reading, the address space is from 0 to 5, and when writing, the address space is from 0 to 2.

The coils supported by this power supply are as shown in the following table:

Coil Addr.	Description	Attribute
0x0000	Power output	read-write
0x0001	OCP function	read-write
0x0002	Power panel keyboard lock	read-write
0x0003	Serial port data endianness setting	read only
0x0004	Constant pressure and constant current state	read only
0x0005	Alarm status	read only

1: Power Output

Reading:

0: The power supply is currently in a shutdown output state.

1: The power supply is currently in the startup output state.

Writing:

0: Specify the power supply to be turned off and the output to be disabled.

1: Specify the power supply startup output.

2: OCP function

Reading:

0: The OCP function is currently turned off.

1: The OCP function is enabled.

Writing:

0: Disable the OCP function.

1: Enable the OCP function.

3: Power panel keyboard lock

Reading:

0: The keyboard on the power panel is in an unlocked state, allowing for free operation.

1: The keyboard on the power panel is in a locked state and cannot be operated.

Writing:

0: he keyboard on the power panel is in an unlocked state and can be operated freely.

1: Enable the keyboard lock function.

Note:

When the keyboard lock is set using the host computer, it is not permanently locked but only locked for 2 seconds. If communication with the power supply is re-established within 2 seconds (including reading, writing to registers or coils), the lock duration will be extended by 2 seconds. If there is no re-communication with the power supply within 2 seconds, the keyboard lock state will be automatically released by the power supply.

4: Serial port data endianness setting

Reading:

0: The voltage and current data are in little-endian format.

1: The voltage and current data are in big-endian format.

5: Constant pressure and constant current state

Reading:

0: The power supply is currently in a constant voltage state.

1: The power supply is currently in a constant current mode.

6: Alarm status

Reading:

0: The power supply is not currently in an alarm state.

1: The power supply is currently in an alarm state.

Note:

The alarm status, also known as the short-circuit protection status. After enabling the OCP function, if the load current in the circuit reaches the current limit point (i.e., the current setting value), the power supply will cut off the output to prevent the expansion of the fault and enter the alarm status. The "alarm status" can be released by writing 0 to the "power output" coil.

Chapter 2 MODBUS instructions applicable to this power supply

This power supply supports the read register (0x03), write register (0x10), read coil (0x01), and write coil (0x05) instructions in the standard MODBUS protocol. Please note that the addresses and quantities for reading and writing cannot exceed the range supported by the power supply. Moreover, the read coil and write coil instructions can only operate one coil at a time and cannot read or write multiple coils simultaneously. If you want to read or write multiple coils at the same time, please use the read and write register instructions for operation.

2.1 Read one or more registers(0x03)

1: Read register instruction

	Slave Addr.	FuncCode	Start Addr.		Data Length		CRC	
Byte	0	1	2	3	4	5	6	7
Data	0x00-0x1F	0x03	AddrH	AddrL	LenghH	LenghL	CRCL	CRCH

Description:

Slave Addr.: The data range is from 0 to 31, and it must be the same as the slave station address set by the power supply.

FuncCode: 0x03, Indicates the execution of an instruction.

Start Addr.: 0x0000-0x0007, Addresses that are out of the specified range are invalid.

Data Length: The number of read registers is 0x0001 to 0x0008. If the starting address plus the data length exceeds 8, it is invalid.

CRC Check Code: The CRC check code generated based on the polynomial 8005.

2: Read register instruction response

	Slave Addr.	FuncCode	Byte Num.	Data	CRC	
Byte	0	1	2	3-N	N+1	N+2
Data	0x00-0x1F	0x03	N-3	(N-3)/2 registers	CRCL	CRCH

Description:

Slave Addr.: The slave station address of the power supply that responds.

FuncCode: 0x03，表示读指令的响应。

Byte Num.: The length of the returned data is indicated by the byte length. Depending on the number of registers to be read (1-8), the byte count ranges from 2 to 16.

Data: The 2-16 byte data read from the power supply.

CRC Check Code: The CRC check code generated based on the polynomial 8005.

2.2 Write one or more registers(0x10)

1: Write register instruction

	Slave Addr.	FuncCode	Start Addr.		Data Length		Byte count	Data	CRC	
Byte	0	1	2	3	4	5	6	7-N	N+1	N+2
Data	0x00-0x1F	0x10	AddrH	AddrL	LenghH	LenghL	N-6	(N-6)/2| registers	CRCL	CRCH

Description:

Slave Addr.: The slave station address of the power supply that responds.

FuncCode: 0x10, Indicates the issuance of a write instruction.

Start Addr.: 0x0000-0x0002, Addresses outside the specified range are invalid.

Data Length: The number of written registers must be specified as 0x0001 - 0x0003. If the starting address plus the data length exceeds 3, it will be invalid.

Byte count: The total number of bytes written into the register is twice the number of registers.

Data: Expect to write the data into the power register.

CRC Check Code: The CRC check code generated based on the polynomial 8005.

2: Write register instruction response, power supply activated.

	Slave Addr.	FuncCode	Start Addr.		Data Length		CRC	
Byte	0	1	2	3	4	5	6	7
Data	0x00-0x1F	0x10	AddrH	AddrL	LenghH	LenghL	CRCL	CRCH

Description:

Slave Addr.: The slave station address of the power supply that responds.

FuncCode: 0x10, Indicates the issuance of a write instruction.

Start Addr.: 0x0000-0x0002, Addresses outside the specified range are invalid.

Data Length: The same as writing instructions.

CRC Check Code: The CRC check code generated based on the polynomial 8005.

2.3 Read coil(0x01)

1: Read coil command

	Slave Addr.	FuncCode	Start Addr.		Data Length		CRC	
Byte	0	1	2	3	4	5	7	8
Data	0x00-0x1F	0x01	AddrH	AddrL	LenghH	LenghL	CRCL	CRCH

Description:

Slave Addr.: The slave station address of the power supply that responds.

FuncCode: 0x01, Indicates reading of the coil command.

Start Addr.: 0x0000-0x0005, Addresses outside the specified range are invalid.

Data Length: The number of written registers must be specified as 0x0001.

CRC Check Code: The CRC check code generated based on the polynomial 8005.

Note:

This power supply only supports reading the status of one coil with one instruction.

2: Read coil command response

	Slave Addr.	FuncCode	Data Length	Data	CRC	
Byte	0	1	2	3	4	5
Data	0x00-0x1F	0x01	0x01	0 或 1	CRCL	CRCH

Description:

Slave Addr.: The slave station address of the power supply that responds.

FuncCode: 0x01, Indicates the response to the read coil command.

Start Addr.: The same as writing instructions.

Data Length: The same as writing instructions.

CRC Check Code: The CRC check code generated based on the polynomial 8005.

2.4 Winding the coil(0x05)

1: Write coil command

	Slave Addr.	FuncCode	Start Addr.		Data		CRC	
Byte	0	1	2	3	4	5	6	7
Data	0x00-0x1F	0x05	AddrH	AddrL	DataH	DataL	CRCL	CRCH

Description:

Slave Addr.: The data range is from 0 to 31 and must be the same as the slave address set by the power supply.

FuncCode: 0x05，表示写线圈指令。

Start Addr.: 0x0000-0x0002，超出范围的地址无效。

Data: The data expected to be written into the coil: Writing 0 will disable the corresponding function, while a non-zero value will enable it.

Note: This power supply only supports writing one coil per instruction.

2: Write the instruction response.

	Slave Addr.	FuncCode	Start Addr.		Data		CRC	
Byte	0	1	2	3	4	5	6	7
Data	0x00-0x1F	0x05	AddrH	AddrL	DataH	DataL	CRCL	CRCH

Description:

When the write operation is successful, the returned response is exactly the same as the write coil instruction.

Chapter 3 Power parameter settings

On the DC power supply panel, long press the "OUTPUT" button for 5 seconds. This will enter the system parameter setting mode. The system parameters that can be set include:

1. This machine's ID.
2. The output status of the power supply at startup (ON or OFF).
3. Screen luminance.
4. Buzzer mute。
5. Serial port communication baud rate.
6. Serial port data endianness setting.

In the system settings mode, rotate the voltage encoding switch to change the parameters; press the voltage encoding switch to switch to the next item and simultaneously save the parameters that were just selected. When you are at the 6th item "Serial Port Data Big/Little Endian Setting", press the voltage encoding switch again to save this parameter and then exit the system parameter setting mode.

The specific setting items and parameter meanings are as shown in the following table:

	Item	Parameter	Description	Default
1	This machine's ID	0 - 31	Specify the number of this machine in the network	0
2	The output status of the power supply at startup	0	Power is turned off upon startup.	0
		1	Power is turned on upon startup.	
3	Screen luminance	0	Low-light level.	0
		1	High-light level.	
4	Buzzer mute	0	No tone notification	1
		1	There is a tone indication	
5	Serial port communication baud rate	1	2400	1
		2	4800	
		3	9600	
		4	19200	
6	Serial port data endianness setting	0	Little-endian structure: Byte transmission is from low to high.	0
		1	Big-endian structure, byte transmission is from high to low.	

Chapter 4 The necessary settings for normal communication

If other devices wish to communicate normally with this power supply, they need to set the parameters of the serial port and the protocol data correctly. If the communication between the upper computer and the power supply fails, these parameters should be checked, including:

1: Address of power supply equipment (slave station)

In the first item of the power system parameter settings, you can specify a device address for the power supply. In the read/write commands sent by the host, the device address (slave address) of the power supply is included. In the network, only the power supply with the matching device address value will respond to the command; otherwise, the communication will not succeed.

2: Serial port selection

Computers and PLCs usually have multiple serial ports. The upper-level software can only successfully communicate if it selects the correct serial port (the one connected to the power supply).

3: Baud rate

In the fifth item of the power system parameter settings, you can specify the communication rate of the power supply. There are four selectable rates: 2400, 4800, 9600 and 19200. The computer (or PLC) must also use the same baud rate for successful communication; otherwise, the communication will not be successful.

4: Data endianness

The big-endian or little-endian format of multi-byte data in communication can be specified. Both parties involved in the communication must use data (set data and actual data) that are formatted consistently in terms of big-endian or little-endian structure. Otherwise, the communication can succeed, but the displayed data and set data of both parties may be inconsistent. In the 6th item of the power system parameter settings, the data big-endian or little-endian format can be set.

5: Communication byte format

When the power supply communicates with other devices, it uses the fixed "8N1" format for communication, which is a data format consisting of 8 data bits, no parity check bit, and 1 stop bit.

6: Register address and data length

In the read register and write register instructions, the starting address cannot exceed the prescribed range. Even after adding the read quantity to the starting address, it must not exceed the prescribed range either; otherwise, there will be no response.

When reading or writing to the coils, only one coil's value can be read or written at a time, and it is necessary to ensure that the address does not exceed the specified range.

Chapter 5 CRC demo code

CRC verification has direct calculation method and lookup table method. The direct calculation method occupies less program space but takes a long time to use the CPU. Therefore, it is rarely used. In the following example, the lookup table method is used to implement the CRC calculation function.

5.1 An example program for implementing CRC calculation

```
const unsigned char CRCHT[256]=
{
    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 CRCLT[256] =
{
    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 Calculation_CRC(unsigned char *Buf, unsigned char Length)
{
    unsigned char Char = 0;
    unsigned char CRCH = 0xFF;
    unsigned char CRCL = 0xFF;
    unsigned int Index;

    while (Char < Length)
    {
        Index = CRCH ^ Buf[Char++];
        CRCH = CRCL ^ CRCHT[Index];
        CRCL = CRCLT[Index] ;
    }

    return (((unsigned int)CRCH << 8) + CRCL);
}
```

There are two scenarios where this function needs to be invoked. One is when receiving data packets sent by other devices, and it is necessary to verify whether there are any errors in the data packet. The other is adding two bytes of CRC check data at the end of the data packet to be sent, and then sending it out together. The following will introduce them respectively.

5.2 when receiving data packets sent by other devices

In the following program, it is assumed that the received data is stored in the array RS232RxBuffer, and the length of the received data (including the two-byte CRC check data at the end) is saved in the variable RS232RxCounter.

```
if (RS232RxBuffer[0] == DeviceAddress &&
    Calculation_CRC(RS232RxBuffer, RS232RxCounter) == 0)
{
    //Process the received data
}
```

5.3 When sending data to other devices, add the CRC data at the end of the data packet.

In the following program, it is assumed that the data to be sent is stored in the array RS232TxBuffer, and the length of the data to be sent (excluding the two-byte CRC check data at the end) is 8.

```
unsigned int DataLength;
//Fill the RS232TxBuffer with 8 bytes of data that is ready for transmission
RS232TxBuffer[0] = 0xXX;
…
RS232TxBuffer[7] = 0xXX;

DataLength = Calculation_CRC(RS232TxBuffer, 8);
RS232TxBuffer[8] = DataLength >> 8;
RS232TxBuffer[9] = DataLength;

//Start the serial port transmission program. The data length is 10 bytes.
…
```