ModBus Communication protocol

ModBus Communication Standards

The ModBus protocol is an industrial communications and distributed control system to integrate PLCs, computers, terminals, and other monitoring, sensing and control devices.

ModBus is a Master-Slave communications protocol. The Master controls all serial activity by selectively polling one or more slave devices. The protocol provides for one master device and up to 247 slave devices on a common line. Each device is assigned an address to distinguish it from all other connected devices.

The ModBus protocol uses the master-slave technique, in which only one device (the master) can initiate a transaction. The other devices (the slaves) respond by supplying the requested data to the master, or by taking the action requested in the query. The master can address individual slaves or initiate a broadcast message to all slaves. Slaves return a message (“response”) to queries that are addressed to them individually. Responses are not returned to broadcast queries from the master. A transaction comprises a single query and single response frame or a single broadcast frame. The transaction frames are defined below.

Valid slave device addresses are in the range of 0–247 decimal. The individual slave devices are assigned addresses in the range of 1–247. A master addresses a slave by placing the slave address in the address field of the message. When the slave sends its response, it places its own address in this address field of the response to let the master know which slave is responding.The Instruction code field of a message frame contains eight bits (RTU). Valid codes are in the range of 1–255 decimal. When a message is sent from a master to a slave device, the Instruction code field tells the slave what kind of action to perform.Examples are to read the ON/OFF states of a group of discrete coils or inputs; to read the data contents of a group of registers; to read the diagnostic status of the slave; to write to designated coils or registers; or to allow loading, recording or verifying the program within the slave. When the slave

responds to the master, it uses the function code field to indicate either a normal (error-free) response or that some kind of error occurred (called an exception response). For a normal response, the slave simply echo to the original instruction code. For an exception response, the slave returns a code that is equivalent to the original function code with its most significant bit set to a logic state of 1.The data field is constructed using sets of two digits, in the range of 00 to FF hexadecimal. These can be made from one RTU character, according to the network's serial transmission mode.

The data field of messages sent from a master to slave devices contains additional information that the slave must use to take the action defined by the function code. This can include items like discrete and register addresses, the quantity of items to be handled, and the count of actual data bytes in the field. If no error occurs, the data field of a response from a slave to a master contains the data requested. If an error occurs, the field contains an exception code that the master application can use to determine the next action to be taken. Two kinds of checksum are used for standard ModBus networks. The error checking field contents depend upon the transmission method that is being used.

1.2 Instruction code and data

1.2.1 Instruction code: 03H, read N(1~16) words

Example: To read 2 words from register address of 0x0004 from the salve with address of 01H, the base structure of the frames should be:

RTU Master sending command frame:

START	T1-T2-T3-T4
Slave Address	01H
Function Code	03H
Start address High	00H
Start address Low	04H
Number of Data(Words)High	00H
Number of Data(Words)Low	02H

CRC High	85H
CRC Low	CAH
END	T1-T2-T3-T4

RTU Slave Response Frame

START	T1-T2-T3-T4
Slave Address	01H
Function Code	03H
Number of Data(Words)	02H
Data high at address 0004H	13H
Data low at address 0004H	88H
Data high at address 0005	02H
Data Low at address 0005	85H
CRC High	73H
CRC Low	CBH
END	T1-T2-T3-T4

1.2.2 Instruction Code：06H（0000 0110） To write a word

Example o write 5000（1388H）to register address of 0x002B to the salve with address of 02H, the base structure of the frames should be:

RTU Master sending command frame:

START	T1-T2-T3-T4
Slave Address	02H
Function Code	06H
Start Address High	00H
Start Address Low	2BH
Data High	13H
Data Low	88H
CRC High	F4H
CRC Low	A7H
END	T1-T2-T3-T4

RTU Slave response command frame

START	T1-T2-T3-T4
Slave Address	02H
Function Code	06H
Start Address High	00H
Start Address Low	2BH

Data High	13H
Data Low	88H
CRC High	F4H
CRC Low	A7H
END	T1-T2-T3-T4

1.3 Communication frame Checksum

2 kind of ModBus standards checksum are support. i.e. Parity Check and Frame Check Sequence (CRC or LRC)

1.3.1 CRC CRC(Cyclical Redundancy Check) Checksum：

To use RTU frame, 2 Bytes of standards CRC checksum are included.

A standards RTU CRC algorithm are use in the protocol, here below a CRC function implementation in C language is provided for reference.

```
//=================================================================
unsigned int crc_cal_value(unsigned char *data_value,unsigned char data_length)
{
    int i;
    unsigned int crc_value=0xffff;
    while(data_length--)
        {
        crc_value^=*data_value++;
          for(i=0;i<8;i++)
        {
            if(crc_value&0x0001)
            crc_value=(crc_value>>1)^0xa001;
            else
             crc_value=crc_value>>1;
           }
        }
    return(crc_value);
}
//=================================================================
```


1.4 Definition of Register address

(1) Function code access address

In SVD-P product function code access address is a double word , the high word of the address is simply mapped with the function code group index:

Function code Group	Access address High word	RAM running Access address

P0-PE	0xF0-PE	0x00-0x0E
A0-AC	0xA0~0xAC	0x40-0x4C
U0	0x07	

And the low word of the address is simple hexadecimal of the function code sub-index

For example:

Function code of Index P0.12 access address is 0xF0 0C,

Function code of Index PC.19 access address is 0xFC 13,

Function code of Index A2.05 access address is 0xA2 05,

Frequently writing EEPROM will affect the lifetime of the EEPROM, parts of function code do not need to be save into EEPROTM but just change parameter in corresponding RAM address could implement function code effect. To use the function code in RAM, just need to change the first digit of address of function code form F-->0, A-->4 . e.g. To change and apply Function code P0.07 in RAM, but do not save to EEPROM, user can use address 0x0007, but the address is not readable. Otherwise, it is regards as illegal address.

（2）Instruction register address & description

Here below is address and description of registers which is used to control the running of the VFD and access to VFD running status

Instruction	Register Address	Function description	R/W
Running frequency setting	1000H	Running frequency setting range (-10000～10000 Decimal)	W/R
Running control instruction	2000H	0001H：Start forward run	W
		0002H：Start reverse run	
		0003H：Forward JOG	
		0004H：Reverse JOG	
		0005H：Coast to stop	
		0006H：Decelerate to stop	
		0007H：Fault reset	
VFD Status	3000H	0001H：Forward Running	R
		0002H：Reverse Running	

Instruction	Register Address	Function description	R/W
		0003H：VFD standby	
Running / Status address	1001H	Setting frequency	R
	1002H	DC BUS voltage	R
	1003H	Output Voltage	R
	1004H	Output Current	R
	1005H	Output Power	R
	1006H	Output Torque	R
	1007H	Running frequency	R
	1008H	DI input status	R
	1009H	DO output status	R
	100AH	AI1 voltage	R
	100BH	AI2 voltage	R
	100CH	AI3 voltage	R
	100DH	Count value	R
	100EH	Length value	R
	100FH	Load speed	R
	1014H	Running frequency2	R
	1015H	Remaining running time	R
	1016H	AI1 voltage before correction	R
	1017H	AI2 voltage before correction	R
	1018H	AI3 voltage before correction	R
	1019H	Linear speed	R
	101AH	Current power-on time (Hour)	R
	101BH	Current running time (Minute)	R
	101CH	Pulse setting frequency (Hz)	R
	101DH	Communication setting value	R
	101EH	Encoder feedback speed (Hz)	R
	101FH	Main frequency X display (Hz)	R
	1020H	Auxiliary frequency Y display (Hz	R
Parameter password validation address	1F00H	****	R
Fault code address	8000H	The fault code shown in 8000H is in accordance with fault code response to function code menu, the only difference it the code response to function code menu is hexadecimal.	R

DO control :

Register Address	Function description
2001H	BIT0: DO1 BIT1: DO2 BIT2: Relay1 BIT3:Relay2 BIT4: FMR BIT5: VDO1 BIT6: VDO2 BIT7: VDO3 BIT8: VDO4 BIT9: VDO5

AO1 control (Write Only):

Register Address	Function description
2002H	0~7FFF Represent 0%~100%

AO2 control (Write Only):

Register Address	Function description
2003H	0~7FFF Represent 0%~100%

Pulse output control (Write Only):

Register Address	Function description
2002H	0~7FFF Represent 0%~100%

Fault code from address 8000H definition

Fault code	Fault type
0x00	No fault
0x01	Reserved
0x02	Overcurrent during acceleration
0x03	Overcurrent during deceleration
0x04	Overcurrent at constant speed
0x05	Overvoltage during acceleration
0x06	Overvoltage during deceleration
0x07	Overvoltage at constant speed
0x08	Buffer resistance overload
0x09	Undervoltage
0x0A	AC drive overload
0x0B	Motor overload
0x0C	Power input phase loss
0x0D	Power output phase loss
0x0E	IBGT Module overheat

Fault code	Fault type
0x0F	External equipment fault
0x10	Communication fault
0x11	Contactor fault
0x12	Current detection fault
0x13	Motor auto-tuning fault
0x14	Encoder/PG card fault
0x15	EEPROM read-write fault
0x16	AC drive hardware fault
0x17	Short circuit to ground
0x18	Reserved
0x19	Reserved
0x1A	Accumulative running time reached
0x1B	User-defined fault 1
0x1C	User-defined fault 2
0x1D	Accumulative power-on time reached
0x1E	Load Loss
0x1F	PID feedback lost during running
0x28	With-wave current limit fault
0x29	Motor switch-over fault during running
0x2A	Too large speed deviation
0x2B	Motor over-speed
0x2D	Motor overheat
0x5A	Encoder setting fault
0x5B	Encoder disconnected fault
0x5C	Initial position fault
0x5E	Encoder speed feedback fault