ZL-U09D2-SL

AC air conditioner communication protocol

CONTENTS

1.INTRODUCTION ……....3
2.SCOPE OF APPLICATION…….…..3
3.REFERENCES……..3
4.PHYSICAL INTERFACE ……..……3
5.MODBUS FUNCTION CODE……..……3
5.1 EFFECTIVE FUNCTION CODE…….……..3
5.2 EFFECTIVE ERROR CODE……..……3
6.MESSAGE FORMAT……..……..4
6.1 READ COIL (0X01)……4
6.2 READ DISCRETE INPUTS (0X02)…….……..5
6.3 READ HOLDING REGISTERS (0X03)…….5
6.4 READ INPUT REGISTERS (0X04)……6
6.5 WRITE SINGLE COIL (0X05)……6
6.6 WRITE SINGLE REGISTER (0X06)……..7
7.COIL DATA POINTS LIST…….……8
8.DISCRETE INPUT DATA POINTS LIST ……9
9.CRC16 CHECK…….……..11

1. Introduction

This article describes the MODBUS communication protocol standard between the air-conditioning controller and the CSU monitoring unit.

2. Scope of application

This agreement applies to the MODBUS agreement of the air-conditioning cooling system.

3. References

Modicon Modbus Protocol Reference Guide PI-MBUS-300 Rev.J

4.Physical interface

Serial port: RS485

Message frame transmission mode: RTU mode

Frame format: 1 start bit, 8 data bits, no check bit, 1 stop bit, CRC check

Baud rate: 9600, 19200 can be set.

5.MODBUS function code

5.1Effective function code

Sheet 1 ModBus Effective function code

Function Code	Description	Definition
0x01	Read Coils	This function code reads status from 1 to 2000 contiguous coils in a remote device.
0x02	Read Discrete Inputs	This function code reads status from 1 to 2000 contiguous discrete inputs in a remote device.
0x03	Read Holding Registers	This function code reads the contents of a contiguous block of holding registers in a remote device.
0x04	Read Input Registers	This function code reads from 1 to approx.125 contiguous input registers in a remote device.
0x05	Write Single Coil	This function code writes a single output to either ON or OFF in a remote device.
0x06	Write Single Register	This function code writes a single holding register in a remote device.
0x10	Write Multiple Registers	This function code writes a block of contiguous register(1 to approx. 120 registers) in a remote device.

5.2 Effective error code

Sheet 2 ModBus Effective error code

Error Code	Description	Definition
0x01	Illegal instruction	The received command is outside the defined valid command.
0x02	Illegal data address	The received data address is out of the defined range.
0x03	Addressing illegal data values	Data address is out of specification.
0x04	Slave device failure	Unsuccessful to modify the data of the register
0x05	Controller is busy, suspend receiving data	The controller cannot execute the command correctly due to special reasons.
0x06	Information frame overlength error	The amount of data received by the controller is too long(255Byte)
0x0C	CRC check error	

5.3 This agreement opens public function codes

Sheet 3This agreement opens public function codes

Code	Description	Definition
0x01	Read coil	
0x03	Read multiple registers	
0x05	Write a single line graph	
0x10	Write multiple registers	

6.Message format

6.1 READ COIL (0x01)

0x01 Command format:

Definition	Address	Function code	Register	Number of registers	CRC Checking
Data	ADDR	01H	Start REG	N REG	CRC16
Number of bytes	1	1	2	2	2

0x01 Response format:

Definition	Address	Function code	Return data bytes	Return data	CRC Checking
Data	ADDR	01H	X	DATA	CRC16
Number of bytes	1	1	1	X	2

Example:

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
Function	01	Function	01
Starting Address Hi	00	Byte Count	**01**
Starting Address Lo	13	Outputs status 27-20	CD
Quantity of Outputs Hi	00	Outputs status 35-28	6B
Quantity of Outputs Lo	13	Outputs status 38-36	05

6.2 READ DISCRETE INPUTS (0x02)

0x02 Command format:

Definition	Address	Function code	Register	Number of registers	CRC Checking
Data	ADDR	02H	Start REG	N REG	CRC16
Number of bytes	1	1	2	2	2

0x02 Response format:

Definition	Address	Function code	Return data bytes	Return data	CRC Checking
Data	ADDR	02H	X	DATA	CRC16
Number of bytes	1	1	1	X	2

Example:

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
Function	02	Function	02
Starting Address Hi	00	Byte Count	03
Starting Address Lo	C4	Inputs status 204-197	AC
Quantity of Inputs Hi	00	Inputs status 212-205	DB
Quantity of Inputs Lo	16	Inputs status 218-213	35

6.3 READ HOLDING REGISTERS (0x03)

0x03 Command format:

Definition	Address	Function code	Register	Number of registers	CRC Checking
Data	ADDR	03H	Start REG	N REG	CRC16
Number of bytes	1	1	2	2	2

0x03 Response format:

Definition	Address	Function code	Return data bytes	Return data	CRC Checking
Data	ADDR	03H	X	DATA	CRC16
Number of bytes	1	1	1	X	2

Example:

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
Function	03	Function	03
Starting Address Hi	00	Byte Count	06
Starting Address Lo	6B	Register value Hi(108)	02
No.of Registers Hi	00	Register value Lo(108)	2B
No.of Registers Lo	03	Register value Hi(109)	00
		Register value Lo(109)	00
		Register value Hi(110)	00
		Register value Lo(110)	64

6.4 READ INPUT REGISTERS (0x04)

0x04 Command format:

Definition	Address	Function code	Register	Number of registers	CRC Checking
Data	ADDR	04H	Start REG	N REG	CRC16
Number of bytes	1	1	2	2	2

0x04 Response format:

Definition	Address	Function code	Return data bytes	Return data	CRC Checking
Data	ADDR	04H	X	DATA	CRC16
Number of bytes	1	1	1	X	2

Example:

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
Function	04	Function	04
Starting Address Hi	00	Byte Count	02
Starting Address Lo	08	Input Reg.9 Hi	00
Quantity of Input Reg.Hi	00	Input Reg.9 Lo	0A
Quantity of Input Rge.Lo	01		

6.5 WRITE SINGLE COIL (0x05)

0x05 Command format:

Definition	Address	Function code	Register	Register value	CRC Checking
Data	ADDR	05H	Start REG	N REG	CRC16
Number of bytes	1	1	2	2	2

0x05 Response format:

Definition	Address	Function code	Register	Register value	CRC Checking
Data	ADDR	05H	Start REG	N REG	CRC16
Number of bytes	1	1	2	2	2

Example:

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
Function	05	Function	05
Output Address Hi	00	Output Address Hi	00
Output Address Lo	AC	Output Address Lo	AC
Output Value Hi	FF	Output Value Hi	FF
Output Value Lo	00	Output Value Lo	00

6.6 WRITE SINGLE REGISTER (0x06)

0x06 Command format:

Definition	Address	Function code	Register	Register value	CRC Checking
Data	ADDR	06	Start REG	N REG	CRC16
Number of bytes	1	1	2	2	2

0x06 Response format:

Definition	Address	Function code	Register	Register value	CRC Checking
Data	ADDR	06	Start REG	N REG	CRC16
Number of bytes	1	1	2	2	2

Example:

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
Function	06	Function	06
Register Address Hi	00	Register Address Hi	00
Register Address Lo	01	Register Address Lo	01
Register Value Hi	00	Register Value Hi	00
Register Value Lo	03	Register Value Lo	03

6.7 Write Multiple registers (0x10)

0x10 Command format:

Definition	Address	Function code	Register	Number of registers	Number of bytes	Register value	CRC Checking
Data	ADDR	16	Start REG	Quantity	Byte Cnt	N REG	CRC16
Number of bytes	1	1	2	2	1	X	2

0x10 Response format:

Definition	Address	Function code	Register	Number of registers	CRC Checking
Data	ADDR	16	Start REG	Quantity	CRC16
Number of bytes	1	1	2	2	2

Example:

Request		**Response**	
Field Name	***(Hex)***	***Field Name***	***(Hex)***
Function	**10**	**Function**	**10**
Starting Address Hi	**00**	**Starting Address Hi**	**00**
Starting Address Lo	**01**	**Starting Address Lo**	**01**
Quantity of Registers Hi	**00**	**Quantity of Registers Hi**	**00**
Quantity of Registers Lo	**02**	**Quantity of Registers Lo**	**02**
Byte Count	**04**		
Registers Value Hi	**00**		
Registers Value Lo	**0A**		
Registers Value Hi	**01**		
Registers Value Lo	**02**		

Notes:

- Address: the address of the air-conditioning equipment;
- Data length: The command packet is the number of registers, and the response packet is the number of bytes of the returned data
- Set command: After the setting is successful, the data message will return as it is
- CRC16 checking: address, function code, starting register or the number of returned data bytes, the number of registers or the number of returned data bytes are CRC16 checked, the CRC16 sending order is low byte first, high byte later

7.COIL DATA POINTS LIST

Function code:0x01

MODBUS COIL ADDR				
Address	Types	Description	Read/write	Remark
1	bit	RT1 Sensor failure	Read	1: Alarm; 0: Stop
2	bit	RT2 Sensor failure	Read	1: Alarm; 0: Stop
3	bit	Internal reservation	Read	
4	bit	Internal reservation	Read	
5	bit	Internal reservation	Read	
6	bit	Humidity sensor failure	Read	1: Alarm; 0: Stop
7	bit	EEPROM Setting data error	Read	1: Alarm; 0: Stop
8	bit	Slave power on, power off	Read	0: OFF, 1:ON
9	bit	High temperature alarm	Read	1: Alarm; 0: Stop
10	bit	Low temperature alarm	Read	1: Alarm; 0: Stop

11	bit	Internal reservation	Read	
12	bit	Internal reservation	Read	
13	bit	Compressor pressure alarm	Read	1: Alarm; 0: Stop
14	bit	Freeze alarm	Read	1: Alarm; 0: Stop
15	bit	Internal reservation	Read	
16	bit	Internal reservation	Read	
17	bit	Compressor working status bit	Read	1: ON, 0: OFF
18	bit	Electric heating working status bit	Read	1: ON, 0: OFF
19	bit	Internal fan station status bit	Read	1: ON, 0: OFF
20	bit	External fan working status bit	Read	1: ON, 0: OFF
21	bit	Working state of hydrogen exhaust machine	Read	1: ON, 0: OFF
22	bit	Internal reservation	Read	
23	bit	Internal reservation	Read	
24	bit	Internal reservation	Read	
25	bit	Internal reservation	Read	
26	bit	Internal reservation	Read	
27	bit	Internal reservation	Read	
28	bit	Internal reservation	Read	
29	bit	Internal reservation	Read	
30	bit	Internal reservation	Read	
31	bit	Internal reservation	Read	
32	bit	Internal reservation	Read	

8.DISCRETE INPUT DATA POINTS LIST

Function code:0x10 write Function code:0x03 read

Function code:0x05 write

Register address	Item	Name	Read/write	Range	Remark
001	int	Indoor temperature 1 temperature	Read	-40 ～ 130	Actual value*10
002	int	Outdoor temperature 1 temperature	Read	-40 ～ 130	Actual value*10
003	int	Internal reservation	Read		
004	int	Internal reservation	Read		
005	int	Internal reservation	Read		
006	int	Humidity	Read	0~99.9	Actual value*10
007	int	Internal reservation	Read		
008	int	Internal reservation	Read		
009	int	Refrigeration start temperature point	Read/write	-9～60	Actual value*10
010	int	Refrigeration stop temperature point	Read/write		Actual value*10
011	int	Heating start temperature point	Read/write		Actual value*10
012	int	Heating stop temperature point	Read/write		Actual value*10

013	int	Heat pipe start temperature point	Read/write	20～35	Actual value*10
014	int	Heat pipe stop temperature point	Read/write	20～35	Actual value*10
015	int	High temperature alarm temperature point	Read/write	25～70	Actual value*10
016	int	Low temperature alarm temperature point	Read/write	-9～19	Actual value*10
017	int	Dehumidification start humidity point	Read/write	30～90	Actual value*10
018	int	Dehumidification stop humidity point	Read/write	30～90	Actual value*10
019	int	Temperature sensor 1 calibration temperature	Read/write	-9 ～ +9	Actual value*10
020	int	Temperature sensor 2 calibration temperature	Read/write	-9 ～ +9	Actual value*10
021	int	Pressure alarm normally open, normally closed setting	Read/write	0 ～ 2	0: Prohibited, 1: Normally open, 2 Normally closed
022	int	Temperature sensor 1 enable setting	Read/write	0 ～ 1	0: disable, 1: enable
023	int	Temperature sensor 2 enable setting	Read/write	0 ～ 1	0: disable, 1: enable
024	int	Humidity sensor enable setting	Read/write	0 ～ 1	0: disable, 1: enable
025	int	Compressor mode setting	Read/write	0 ～ 2	0: Normal control, 1: Forced to open, 2: Forced to close.
026	int	Electric heating mode setting	Read/write	0 ～ 2	0: Normal control, 1: Forced to open, 2: Forced to close.
027	int	Internal fan mode setting	Read/write	0 ～ 2	0: Normal control, 1: Forced to open, 2: Forced to close.
028	int	External fan mode setting	Read/write	0 ～ 2	0: Normal control, 1: Forced to open, 2: Forced to close.
029	int	Temperature sensor 1 invalid setting	Read/write	0 ～ 1	0: Disable, do not detect temperature sense 1 failure alarm
030	int	Temperature sensor 2 invalid setting	Read/write	0 ～ 1	0: Disable, do not detect temperature sense 2 failure alarm
031	int	Humidity failure setting	Read/write	0 ～ 1	0: Disable, do not detect dampness failure alarm
032	int	High temperature alarm failure setting	Read/write	0 ～ 1	0: Disable, do not detect high temperature failure alarm
033	int	Low temperature alarm failure setting	Read/write	0 ～ 1	0: Disable, do not detect low temperature failure alarm
034	int	Pressure alarm failure setting	Read/write	0 ～ 1	0: Disable, do not detect pressure failure alarm

035	int	Freeze alarm invalidation setting	Read/write	0 ～ 1	0: Disabled, no freeze failure alarm is detected
036	int	System (controller) switch	Read/write	0 ～ 0xFF00	Turn on:0xFF00, close: 0x0000

Temperature representation: 27.5℃= 27.5 x 10 = 275 = 0x0113
-9.9℃= -9.9 x 10 = -99 = 0xFF9D

Humidity: 80%RH = 80 x 10 = 800 = 0x0320

9.CRC16 Check

CRCWhen adding to the message, the low byte is added first, then the high byte.

```
unsigned int CRC_Check(unsigned char *puchMsg, unsigned int usDataLen)
{
    unsigned char uchCRCHi = 0xFF;   // high byte of CRC initialized
    unsigned char uchCRCLo = 0xFF;   // low byte of CRC initialized
    unsigned char uIndex;            // will index into CRC lookup table

    while (usDataLen--)              // pass through message buffer
    {
        uIndex = uchCRCLo ^ *puchMsg++; // calculate the CRC
        uchCRCLo = uchCRCHi ^ auchCRCHi[uIndex];
        uchCRCHi = auchCRCLo[uIndex];
    }
    return (uchCRCHi << 8 | uchCRCLo);
}

const unsigned char auchCRCHi[] =
{
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
};

// Table of CRC values for low□order byte
const unsigned char auchCRCLo[] =
{
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
};
```