Innofluid Co., Ltd.
Add: Building 10, No. 860, Xinyang Road, Lingang New Area, Pilot Free Trade Zone, Shanghai, China
Tel: 0086-312-5958380 Fax: 0086-312-6780636
Website: www.innofluid.com Email: info@innofluid.com

HPM Series MODBUS Communication Protocol

Note: The hexadecimal numbers are expressed by 'XXXXH' or 'XXH' in the below description.

1. MODBUS-RTU Standard Communication Format

This communication use MODBUS RTU mode, message frame as below:

Slave address	Function code	Data area	CRC Check (Cyclic Redundancy Check)	
1 Byte	1 Byte	0 or up to 252 bytes	2 Bytes	
			CRC low	CRC high

(1)**Slave address**: Host controls peristaltic pump address number. The pump address number should not be same when they are in the same 485 line. The address number range is 1~32, 0 means broadcast.

(2)**Function code**: The protocol use 2 common function codes which defined by MODBUS protocol.

03H: Read the contents of holding register

06H: Write a word to the holding register

10H: Write a long type to the holding register

Data zone: Specific information instructions that the peristaltic pump needs to execute, such as start/stop, direction, accelerate/decelerate etc.

(3)**CRC check**: CRC code is 2 bytes, 16 check codes. Use CRC-16 (which used in American binary synchronous system).

Polynomial: G(X)=X16+X15+X2+1.

CRC check C language code please refer to Appendix 1.

Frame format for transmission of each function code

03 function code:

Request:

Slave address	1 byte	Pump setting address 1-32 or 0
Function code	1 byte	0x03
Start address	2 bytes	0x0000-0xFFFF
Number of registers	2 bytes	1-125 (0x7D)
CRC check	2 bytes	

Respond

Slave address	1 byte	Pump setting address 1-32 or 0
Function code	1 byte	0x03
Number of bytes	1 byte	2* number of registers
Data	Number of registers *2 bytes	
CRC check	2 bytes	

06 function code

Request:

Slave address	1 byte	Pump setting address 1-32 or 0
Function code	1 byte	0x06
Register address	2 bytes	0x0000-0xFFFF
Register value	2 bytes	0x0000-0xFFFF
CRC check	2 bytes	

Respond:

Slave address	1 byte	Pump setting address 1-32 or 0
Function code	1 byte	0x06
Register address	2 bytes	0x0000-0xFFFF
Register value	2 bytes	0x0000-0xFFFF
CRC check	2 bytes	

0x10 function code

Request:

Slave address	1 byte	Pump setting address 1-32 or 0
Function code	1 byte	0x10
Start address	2 bytes	0x0000-0xFFFF
Number of registers	2 bytes	0x0001 到 0x0078
Number of bytes	1 byte	Number of registers*2
Register value	Number of register*2 bytes	
CRC check	2 bytes	

Respond:

Slave address	1 byte	Pump setting address 1-32 or 0
Function code	1 byte	0x10
Start address	2 bytes	0x0000-0xFFFF
Number of registers	2 bytes	0x0001 to 0x0078
CRC check	2 bytes	

2. Communication Setting

(1) Communication baud rate: 1200, 2400, 4800, 9600, 19200 for option.

(2) Check Digit: None parity, odd parity, even parity.

(3) Byte structure: 1 start bit + 8 data bits +1 check parity bit + 1 stop bit

(4) Bit sequence sending order: The least significant bit (**LSB**)….. The most significant bit (**MSB**)

Start	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**Check**	**Stop**

(5) Data transferring format:

Integer (2 bytes):

Data: (High bit) The second byte The first byte (Low bit)

Send: The second byte The first byte

For example: 1234H send 12H 34H

Floating-point type (4 bytes):

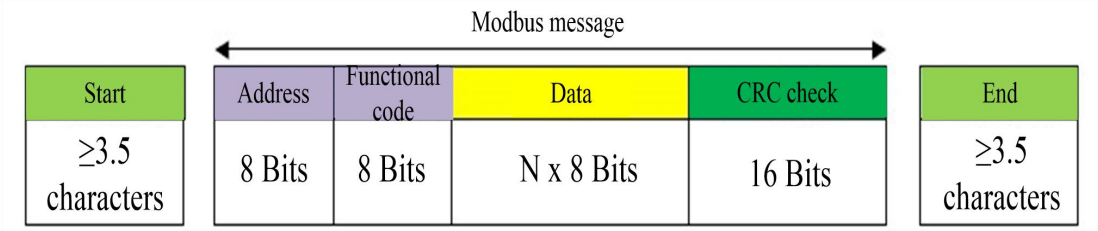
Data: (High bit) The fourth byte The third byte The second byte The first byte (Low bit)

Send: The fourth byte The third byte The second byte The first byte

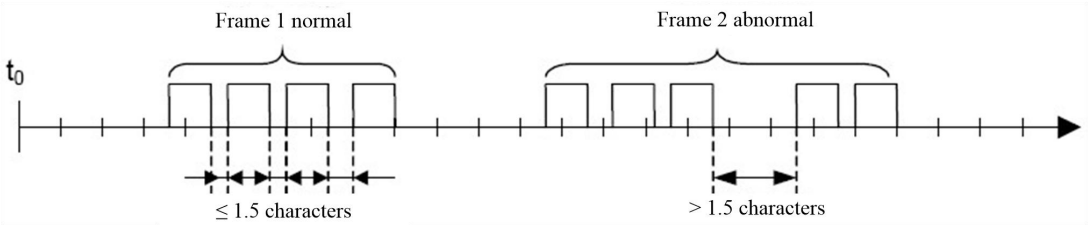
For example: 8.9 send 41H 0EH 66H 66H

3. MODBUS Message RTU Frame Format

In RTU mode, the message frames are distinguished by idle intervals with a duration of at least 3.5 characters. As the follow picture:



The entire message frame must be sent in a continuous stream of characters. If the idle space between two characters is greater than 1.5 character times, the message frame is considered as incomplete, should be discarded by receiving node. As below:



4. Abnormal Response

When host sends request data, the slave receives data abnormal, it should have abnormal reaction. If the address code sent from host is wrong, there is no this address code between slaves or the data received by slave is wrong when CRC check, no abnormal code return, the host should have super response mechanism.

Function code domain: Abnormal response function code is normal response function code +80H.

Data domain: Return to abnormal code, define as below:

Frame format returned by exception code

Slave address	1 byte	Pump setting address 1-32 or 0
Function code	1 byte	Host sends function code+0x80
Error code types	1 byte	As below chart
CRC check	2 bytes	

Chart 1: Abnormal code definition

Code	Name	Meaning
01H	Illegal function code	The function code received by peristaltic pump except 03H/06H/10H.
02H	Illegal data address	This abnormal code means the register address is not allowed data which received by peristaltic pump.
03H	Illegal data value	Written data does not meet the operating range.
04H	Parameter error	The calculated parameters of the operation are out of range.
05H	No permission obtained	When the third level permission is enabled and the current login user does not have communication permission
06H	Slave device (peristaltic pump) busy	The current state of the peristaltic pump conflict with the command received, unable to complete the command.

Peristaltic Pump only receive MODBUS command with the Main Interface, other interface do not receive message.

5. Holding Register Address and Contents

Basic Parameters Setting

Parameter type	Address (Decimal)	Name	Range	Function code	Data Type
The following register parameters are readable and writable.					
Basic parameters	1000	Pump head model	Corresponding parameters are shown in Table 1	06	unsigned short int (2 Bytes)
	1001	Tubing model		06	unsigned short int (2 Bytes)
Transmission parameter	1002	Motor speed	0.1-600 rpm	10	Float (4 Bytes)
	1004	Flow rate	0.1-99999 mL/min	10	Float (4 Bytes)
	1006 (only read)	Unit of flow rate	0: uL/min; 1: mL/min; 2: L/min;	03	unsigned short int (2 Bytes)
Controlling parameter	1007	Suck-back angle	0-360°	06	unsigned short int (2 Bytes)
	1008	Start/stop control	1: Start 0: Stop	06	unsigned short int (2 Bytes)
	1009	Direction control	1: Clockwise 0: Counterclockwise	06	unsigned short int (2 Bytes)

	1010	Full speed running	1: Start up full speed 0: Stop full speed	06	unsigned short int (2 Bytes)
	1011	Automatic restart	1: Turn on 0: Turn off	06	unsigned short int (2 Bytes)
	1012	Operation modes	0: Transmission 1: Dispensing	06	unsigned short int (2 Bytes)
	1013	Dispensing mode	0: Dispensing 1: Fixed volume 2: Speed dispensing	06	unsigned short int (2 Bytes)
Dispensing register parameters are readable and writable					
Dispensing parameter	1020	Set liquid volume	0.1-9999 (Accurate to 1 decimal place)	10	Float (4 Bytes)
	1022	Unit of liquid volume	0: uL; 1: mL; 2: L;	06	unsigned short int (2 Bytes)
	1023	Operation time	0.1-9999 (Accurate to 1 decimal place)	10	Float (4 Bytes)
	1025	Operation time unit	0: sec; 1: min; 2: hour;	06	unsigned short int (2 Bytes)
	1026	Pause time	0.1-9999 (The minimum is 0.5 seconds)	10	Float (4 Bytes)
	1028	Pause time unit	0: sec; 1: min; 2: hour;	06	unsigned short int (2 Bytes)
	1029	Repeat times	0-9999 (0 is unlimited)	06	unsigned short int (2 Bytes)
The quantitative metering register parameters are readable and writable.					
Fixed volume parameters	1030	Set liquid volume	0.1-9999 (Accurate to 1 decimal place)	10	Float (4 Bytes)
	1032	Liquid volume unit	0: uL; 1: mL; 2: L;	06	unsigned short int (2 Bytes)
	1033	Set flow rate	0.1-9999 (Accurate to 1 decimal place)	10	Float (4 Bytes)
	1035	Flow rate unit	0: uL/min; 1: mL/min; 2: L/min;	06	unsigned short int (2 Bytes)
	1036	Pause time	0.1-9999	10	Float (4 Bytes)
	1038	Pause time unit	0:sec; 1: min; 2: hour;	06	unsigned short int (2 Bytes)
	1039	Repeat times	0-9999 (0 is unlimited)	06	unsigned short int (2 Bytes)

The dispensing speed register parameters are readable and writable.					
Dispensing speed parameter	1040	Motor speed	0.1-600rpm	10	Float (4 Bytes)
	1043	Operation time	0.1-9999 (Accurate to 1 decimal place)	10	Float (4 Bytes)
	1045	Operation time unit	0: sec; 1: min; 2: hour;	06	unsigned short int (2 Bytes)
	1046	Pause time	0.1-9999	10	Float (4 Bytes)
	1048	Pause time unit	0: sec; 1: min; 2: hour;	06	unsigned short int (2 Bytes)
	1049	Repeat times	0-9999 (0 is unlimited)	06	unsigned short int (2 Bytes)
Calibration parameters can only be written, not read.					
Calibration parameter	2000	Reset	1: Reset	06	unsigned short int (2 Bytes)
	2003	Start up calibration	1: Start 0: Stop	06	unsigned short int (2 Bytes)
	2004	Actual filling volume	0.01-9999.99mL	10	Float (4 Bytes)
	2008	Fine tune to increase the liquid volume value	0.01-9999.99mL (Cannot exceed the set liquid volume value)	10	Float (4 Bytes)
	2010	Fine tune to decrease the liquid volume value	0.01-9999.99mL (Cannot exceed the set liquid volume value)	10	Float (4 Bytes)
The following register parameters can only be read					
	3000	Total amount of transmission	0-9999990mL	03	Float (4 Bytes)
	3001	Total transfer time	0-3599996400sec	03	Float (4 Bytes)

Note:

1) When the pump head and hose model are modified, all current parameters will be recalculated. In order to ensure the correct parameters, it is recommended to reset the function parameters to be used.

2) When external analog speed control is turned on, registers 1002 and 1004 cannot set parameters.

When you need to change from the transmission mode to the [Fixed volume] mode, you first need to set the operation mode (1012) value to 1, and then set the distribution mode (1013) value to 1 (the same applies to the dispensing and speed dispensing settings).

3) When modifying the [Dispensing], [Fixed volume], and [Speed dispensing] parameters, if the parameters sent exceed

the current pump head range, an error prompt will not be returned immediately. All parameters will be recalculated when starting. If there is an error, an exception code 04H will be returned.

4) Please set the register parameters individually according to the table. Do not accept a single instruction to set multiple registers continuously.

5) Some calibration parameters are only available in the stopped state.

Table 1 Pump head and tubing No.

Pump head and tubing chart:

Pump head	Pump head model	Tubing model	Tubing specification
EasyPumpI	0	13	13#
		14	14#
		19	19#
		16	16#
		25	25#
		17	17#
		18	18#
EasyPumpII	1	15	15#
		24	24#
		35	35#
		36	36#
EasyPumpIII	2	13	13#
		14	14#
		19	19#
		16	16#
		25	25#
		17	17#
		18	18#
EasyPumpIV	3	15	15#
		24	24#
		35	35#
		36	36#
EasyPumpV	4	13	13#
		14	14#
		19	19#
		16	16#
		25	25#
EasyPumpVI	5	13	13#
		14	14#
		19	19#

		16	16#
		25	25#
2*EasyPumpI	6	13	13#
		14	14#
		19	19#
		16	16#
		25	25#
		17	17#
		18	18#
2*EasyPumpII	7	15	15#
		24	24#
		35	35#
		36	36#
2*EasyPumpIII	8	13	13#
		14	14#
		19	19#
		16	16#
		25	25#
		17	17#
		18	18#
2*EasyPumpIV	9	15	15#
		24	24#
		35	35#
		36	36#
EasyPumpV-Y	10	115	0.5*1.6
		13	13#
		14	14#
		19	19#
		16	16#
EasyPumpVI_Y	11	115	0.5*1.6
		13	13#
		14	14#
		19	19#
		16	16#
DZ25-3L	16	15	15#
		24	24#
		35	35#
		36	36#

2*DZ25-3L	17	15	15#
		24	24#
		35	35#
		36	36#
AMC_(10)	25	101	1*1
		102	2*1
		103	2.4*0.86
		104	2.79*0.86
		105	3*1
		106	0.19*0.86
		107	0.25*0.86
		108	0.51*0.86
		109	0.89*0.86
		110	1.14*0.86
		111	1.42*0.86
		112	2.06*0.86
		114	0.13*0.86
AMC_(6)	26	101	1*1
		102	2*1
		103	2.4*0.86
		104	2.79*0.86
		105	3*1
		106	0.19*0.86
		107	0.25*0.86
		108	0.51*0.86
		109	0.89*0.86
		110	1.14*0.86
		111	1.42*0.86
		112	2.06*0.86
		114	0.13*0.86

6. Sending Data Format

Unsigned short int format

Peristaltic pump address	Function code	Register address		Data (unsigned short int)		CRC	
	06H	Address H	Address L	Data H	Data L	L	H

Float format

Pump address	Function code	Register address		The quantity of register		The quantity of byte	Data (Float)				CRC check	
	10H	H	L	00H	02H	04H	H2	H1	L2	L1	L	H

Read register format

Peristaltic pump address	Function code	Register start address		The number of registers		CRC Check	
	03H	Address H	Address L	Data H	Data L	L	H

(1) Set pump head model

The peristaltic pump address is 1, set the pump head to EasyPump, the number is 0000H.

Send: **01 06 03 E8 00 00 09 BA**

Back: **01 06 03 E8 00 00 09 BA**

(2) Set tubing model

The peristaltic pump address is 1, set the tubing model to 16#, the number is 0010H

Send: **01 06 03 E9 00 10 59 B6**

Back: **01 06 03 E9 00 10 59 B6**

(3) Set motor speed

The peristaltic pump address is 1, set the motor speed to 58.8rpm.

Send: **01 10 03 EA 00 02 04 42 6B 33 33 58 29**

Back: **01 10 03 EA 00 02 60 78**

(4) Set flow rate

The peristaltic pump address is 1, set the flow rate to 50ml/min

Send: **01 10 03 EC 00 02 04 42 48 00 00 7D 2C**

Back: **01 10 03 EC 00 02 80 79**

(5) Set back suction angle

The peristaltic pump address is 1, set back suction angle to 60°.

Send: **01 06 03 EF 00 3C B8 6A**

Back: **01 06 03 EF 00 3C B8 6A**

(6) Start/stop control

The peristaltic pump address is 1, controlling start is 0001H, controlling stop is 0000H

Send start: **01 06 03 F0 00 01 48 7D**

Back: **01 06 03 F0 00 01 48 7D**

Send stop: **01 06 03 F0 00 00 89 BD**

Back: **01 06 03 F0 00 00 89 BD**

(7) Direction control

The peristaltic pump address is 1, clockwise is 0001H, counterclockwise is 0000H.

Send: **01 06 03 F1 00 01 19 BD (clockwise)**

Back: **01 06 03 F1 00 01 19 BD**

(8) Set full speed

The peristaltic pump address is 1, full speed is 0001H

Send: **01 06 03 F2 00 01 E9 BD**

Back: **01 06 03 F2 00 01 E9 BD**

(9) Set operation mode

The peristaltic pump address is 1, set operation mode to dispensing mode

Send: **01 06 03 F4 00 01 09 BC**

Back: **01 06 03 F4 00 01 09 BC**

(10) Set allocation mode

The peristaltic pump address is 1, set allocation mode is dispensing mode

Send: **01 06 03 F5 00 00 99 BC**

Back: **01 06 03 F5 00 00 99 BC**

(11) Set the timed quantitative liquid volume

The peristaltic pump address is 1, set liquid volume to 10.00mL

Liquid volume value (register 1020) **Send: 01 10 03 FC 00 02 04 41 20 00 00 FD B8**

Back: 01 10 03 FC 00 02 81 BC

Liquid volume units (register 1022) **Send: 01 06 03 FE 00 01 29 BE**

Back: 01 06 03 FE 00 01 29 BE

(12) Set the timing and quantitative running time

The peristaltic pump address is 1, set operation time to 10.00s

Time value (register 1023) **Send: 01 10 03 FF 00 02 04 41 20 00 00 BD AD**

Back: 01 10 03 FF 00 02 71 BC

Time value unit (register 1025) Send: **01 06 04 01 00 00 D9 3A**

Back: **01 06 04 01 00 00 D9 3A**

(13) Set pause time of Dispensing

The peristaltic pump address is 1, set pause time to 1s

Pause time (register 1026) Send: **01 10 04 02 00 02 04 3F 80 00 00 4D 4A**

Back: **01 10 04 02 00 02 E1 38**

Pause time unit (register 1028) Send: **01 06 04 04 00 00 C9 3B**

Back: **01 06 04 04 00 00 C9 3B**

(14) Set the number of repetitions of the Dispensing (register 1029)

The peristaltic pump address is 1, set the number of repetitions to 100

Send: **01 06 04 05 00 64 99 10**

Back: **01 06 04 05 00 64 99 10**

7. Appendix 1——CRC Check C Language Code

CRC generation process:

(1) Put one 16 bits register into hexadecimal FFFF(all 1), we call it CRC register.

(2) Make the first 8 bytes with 16 CRC register low bytes XOR, the result put in CRC register.

(3) Detect LSB of CRC register

If LSB is 0, move CRC register 1 bit to right (towards to direction of LSB) , MSB zeroing.

If LSB is 1, move CRC register 1 bit to right (towards to direction of LSB), MSB zeroing, then XOR the polynomial

value of the CRC register 0xA001 (1010 0000 0000 0001).

(4) Repeat Step 3, until finish 8 shifts. After finish this operation, will finish the complete operation for 8 Bytes.

(5) Repeat Step 2 to Step 5 for the next Bytes in message. Continue this operation till all the message be deal with finished.

(6) The final content in CRC register is CRC value.

(7) When put CRC value in message, high and low Bytes must be exchanged, described as below:

C language code:

```
void CRCVerify(char *rec,char CRClen,char CRCdata[2])
{
    char i1,j;
    unsigned int crc_data=0xffff;
    for(i1=0; i1<CRClen; i1++)
    {
        crc_data=crc_data^rec[i1];
        for(j=0; j<8; j++)
        {
            if(crc_data&0x0001)
            {
                crc_data>>=1;
                crc_data^=0xA001;
            }
            else
            {
                crc_data>>=1;
            }
        }
    }
    CRCdata[0]=(char)(crc_data);
    CRCdata[1]=(char)(crc_data>>8);
}
```