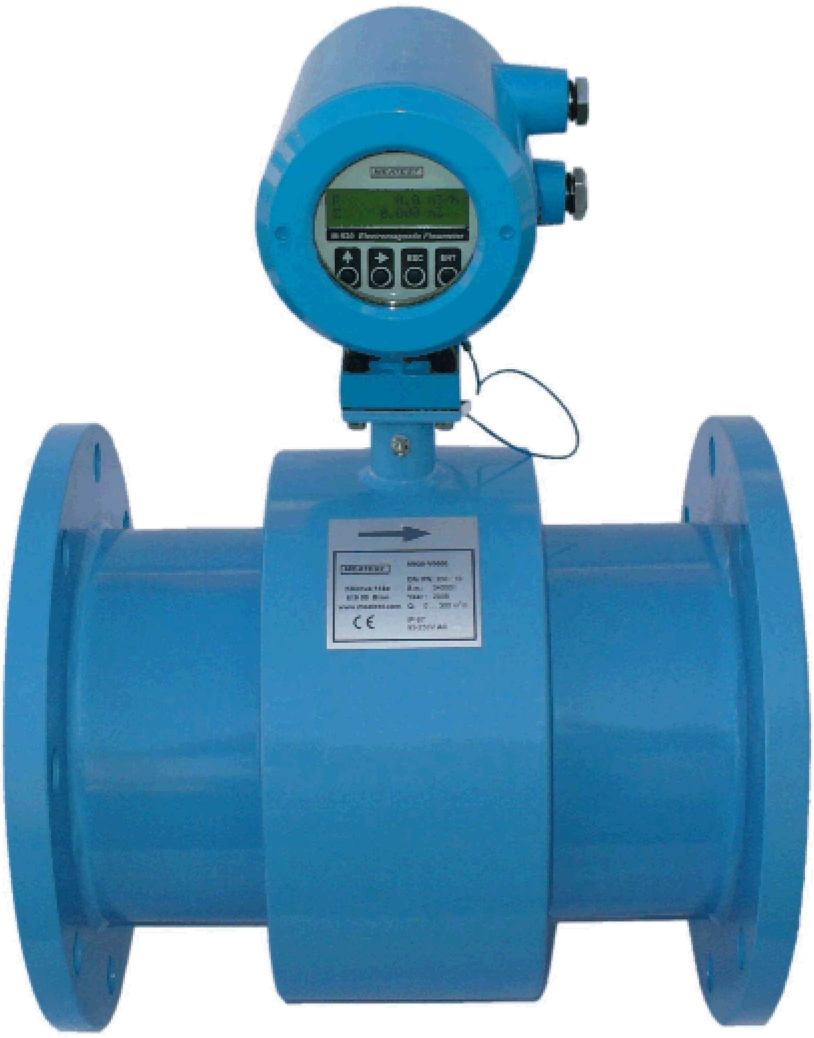
# MODBUS

## *M920 protocol description*

**User's manual**



**CONTENT**

# 1. Basic information

## 1.1. Introduction to Modbus

This document specifies the MODBUS communications protocol as implemented on the magnetic flowmeter M920.

This manual does not try to be a complete guide to the MODBUS protocol, but will show how to structure a message that the instruments will recognize.

For Modbus communication is used RS-485 interface. Instruments communicate using a master-slave technique, in which only one device is the master and theslave devices supply the requested data when addressed. Typical master devices can be a host computer.

Only the master can initiate transactions (requests), and only the addressed device responds.

The Modbus request consist of:

- an address,
- a function code defining the requested action,
- data (if necessary for the requested function), and
- error check for testing the integrity of the message.

The slave’s response contains:

- the slave address,
- data conform the request type, and
- error check.

If the data integrity test fails, no response is sent back.
If a request cannot be processed an exception message is returned.

## 1.2. Transmission format

There are two serial transmission modes for the MODBUS protocol, ASCII or RTU (Remote Transmission Unit) framing. The user has to select the desired protocol along with the serial communication parameters (baud rate, paritytype). Note that all these parameters must be the same for all instruments in the network.

### 1.2.1. ASCII-mode

When device communicate on a MODBUS serial line using ASCII mode, each 8–bit byte in a message is sent as two ASCII characters. This mode is used when the physical communication link or the capabilities of the device does not allow the conformance with RTU mode requirements regarding timers management.

Remark : this mode is less efficient than RTU since each byte needs two characters.

The format ( 10 bits ) for each byte in ASCII mode is :

Coding System: Hexadecimal, ASCII characters 0–9, A–F. One hexadecimal character contains 4-bits of data within each ASCII character of the message

Bits per Byte: 1 start bit

7 data bits (least significant bit sent first)

1 bit for parity completion

1 stop bit

The default parity mode is Even parity.

Remark : the use of no parity requires 2 stop bits.

ASCII Message framing:

| Start | Address | Function | Data | LRC | End |
|---|---|---|---|---|---|
| 1 char<br>: | 2 chars | 2 chars | 0 up to 2x252 char(s) | 2 chars | 2 chars<br>CR, LF |

In ASCII mode, a message is delimited by specific characters as Start-of-frames and End-of-frames. A message must start with a 'colon' ( : ) character (ASCII 3A hex), and end with a 'carriage return – line feed' (CRLF) pair (ASCII 0D and 0A hex).

In ASCII mode, messages include an error–checking field that is based on a Longitudinal Redundancy Checking (LRC) calculation that is performed on the message contents, exclusive of the beginning 'colon' and terminating CRLF pair characters.

### *1.2.2. RTU-mode*

When devices communicate on a MODBUS serial line using the RTU (Remote Terminal Unit) mode, each 8–bit byte in a message contains two 4–bit hexadecimal characters. The main advantage of this mode is that its greater character density allows better data throughput than ASCII mode for the same baud rate. Each message must be transmitted in a continuous stream of characters.

The format ( 11 bits ) for each byte in RTU mode is :

Coding System: 8–bit binary

Bits per Byte: 1 start bit

8 data bits (least significant bit sent first)

1 bit for parity completion

1 stop bit

The default parity mode is even parity.

Remark : the use of no parity requires 2 stop bits.

RTU Message framing:

| Start | Address | Function | Data | CRC | End |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| >= 3.5 char. | 1 byt | 1 byt | 0 up to 252 byte(s) | 2 bytes | >= 3.5 char. |

In RTU mode, message frames are separated by a silent interval of at least 3.5 character times.

The RTU mode includes an error–checking field that is based on a Cyclical Redundancy Checking (CRC) method performed on the message contents.

# 2. Supported functions

## 2.1. Read coil status (Function 01)

Function reads the ON/OFF status of discrete inputs or discrete (bit) variables in the instrument. Query contains the starting coil address and the quantity of coils to be read.

## 2.2. Read input status (Function 02)

Function 1 and 2 perform the same action – see description of Function 01.

## 2.3. Read multiple holding registers (Function 03)

Function reads the binary contents of holding registers in the instrument. Query contains the starting registr address and the quantity of registers to be read. The maximum number of registers at each request is limited to 44 (RTU) or 22 (ASCII). Exception is double which can be read only one variable using this function.

## 2.4. Read input registers (Function 04)

Function 3 and 4 perform the same action – see description of Function 03.

## 2.5. Write single coil (Function 05)

Function writes to a single coil value ON or OFF. ON value is presented as 0xff00, OFF value is presented as 0x0000. Command contains the coil address and requested value. The normal response is an echo of the command, returned after the coil state has been changed.

## 2.6. Write multiple holding registers (Function 16)

Function writes new values into a sequnce of holding registers. Command contains the register starting address, number of affected registers and requested values. The normal response contains number of changed registers. In this function can be write just one variable (integer, float, double ...).

# 3. Commands table

Tables in this chapter contain following columns (description of variables):

1) Address
2) Name
3) Type
4) Access – Read / Write
5) RS232 command – see description of this command in the user's manual

## 3.1. Table 1 Bit variables (single coils)

| Address | Name | Type | Access | RS232 command |
|---|---|---|---|---|
| 0x1000 | Negative flow direction | bit | R/W | FFD |
| 0x1001 | Selftest | bit | R/W | FIS |
| 0x1002 | Current loop test | bit | R/W | FCE |
| 0x1003 | Current output status | bit | R | RCE |
| 0x1004 | System error status | bit | R | RES |
| 0x1005 | Dosing activity status | bit | R | RDA |
| 0x1006 | Auxiliary volume clear | bit | R/W | CLRAV |
| 0x1007 | Min. / max. flowrate clear | bit | R/W | CLRMM |
| 0x1008 | Main volume clear | bit | R/W | CLRVO |
| 0x1009 | Datalogger clear | bit | R/W | DCLR |
| 0x100a | Dosing volume reset | bit | R/W | CLRDO |
| 0x100b | Timed volume counter reset | bit | R/W | CLRTV |
| 0x100c | Magnetic pointer control | bit | R/W | FME |
| 0x100d | Dosing volume reset with restart | bit | R/W | CLRDR |
| 0x100e | Empty pipe control | bit | R/W | FEP |

Bit variables data format:

```
ON  is expressed as 0xff00
OFF is expressed as 0x0000
```

### 3.2. Table 2 Integer variables (holding registers)

| Address | Name | Type | Access | RS232 command |
|---|---|---|---|---|
| 0x3000 | Nominal diameter | integer | R/W | RDN |
| 0x3001 | Datalogger number of samples | integer | R | DNR |
| 0x3002 | Datalogger filling (bytes) | integer | R | DBT |

Integer variables data format:

```
MSB                             LSB
(most sign.bit)     (least sign.bit)
-----------------  -----------------
|X|X|X|X|X|X|X|X|  |X|X|X|X|X|X|X|X|
-----------------  -----------------
      byte 1            byte 0
```

### 3.3. Table 3 Long variables (holding registers)

| Address | Name | Type | Access | RS232 command |
|---|---|---|---|---|
| 0x5000 | Calibration password access setting | long | R/W | FPC |
| 0x5002 | Password setting | long | W | PSW |
| 0x5004 | Basic password access setting | long | R/W | FPB |
| 0x5006 | Errors reading | long | R | IER |
| 0x5008 | Mask for State Output error message | long | R/W | SEM |

Long variables data format:

```
MSB                                                                     LSB
(most sign.bit)                                             (least sign.bit)
-----------------  -----------------  -----------------  -----------------
|X|X|X|X|X|X|X|X|  |X|X|X|X|X|X|X|X|  |X|X|X|X|X|X|X|X|  |X|X|X|X|X|X|X|X|
-----------------  -----------------  -----------------  -----------------
      byte 3             byte 2             byte 1             byte 0
```

## 3.4. Table 4 Time (long) variables (holding registers)

| Address | Name | Type | Access | RS232 command |
|---|---|---|---|---|
| 0x5800 | Time & date of the begin last finished counting period | time (long) | R | RTB |
| 0x5802 | Time & date of the end last finished counting period | time (long) | R | RTE |
| 0x5804 | Date and time of minimum flowrate | time (long) | R | RND |
| 0x5806 | Date and time of maximum flowrate | time (long) | R | RXD |

Time long variables data format:

```
MSB                                                               LSB
(most sign.bit)                                       (least sign.bit)
----------------- ----------------- ----------------- -----------------
|Y|Y|Y|Y|Y|Y|L|L| |L|L|D|D|D|D|D|H| |H|H|H|H|M|M|M|M| |M|M|S|S|S|S|S|S|
----------------- ----------------- ----------------- -----------------
      byte 3            byte 2            byte 1            byte 0
Where:
Y: 0-63 year (2000- 2063)
L: 0-11 month (1-12)
D: 0-30 day (1-31)
H: 0-23 hour (0-23)
M: 0-59 minute (0-59)
S: 0-59 second (0-59)
```

## 3.5. Table 5 Char variables (holding registers)

| Address | Name | Type | Access | RS232 command |
|---|---|---|---|---|
| 0x6000 | Current output mode setting | char | R/W | SCM |
| 0x6001 | Frequency output mode setting | char | R/W | SFM |
| 0x6002 | Impulse output mode setting | char | R/W | SPM |
| 0x6003 | State output mode setting | char | R/W | SSM |
| 0x6004 | Digital input mode setting | char | R/W | SIM |
| 0x6005 | Impulse width setting | char | R/W | SPT |
| 0x6006 | Flowrate unit | char | R/W | FFS |
| 0x6007 | Volume unit | char | R/W | FVS |
| 0x6008 | Flowrate resolution | char | R/W | FFR |
| 0x6009 | Volume resolution | char | R/W | FVR |
| 0x600a | Time constant | char | R/W | FTC |
| 0x600b | Datalogger step | char | R/W | DST |
| 0x600c | Datalogger filling (percentage) | char | R | DPC |

| | | | | |
|---|---|---|---|---|
| 0x600d | Language setting | char | R/W | FLG |
| 0x600e | Number of calibration points | char | R/W | CPN |
| 0x600f | Backlight mode setting | char | R/W | FDB |
| 0x6010 | Contrast setting | char | R/W | FDC |
| 0x6011 | Message time setting | char | R/W | FDM |
| 0x6012 | Timed volume counting period | char | R/W | FTI |
| 0x6013 | Week begin | char | R/W | FTW |
| 0x6014 | Date format setting | char | R/W | FDF |
| 0x6015 | Actual access level | char | R/W | PAL |
| 0x6016 | Power supply type | char | R/W | PPW |

Char variables data format:

```
MSB              LSB
(most s.bit)   (least s.bit)
-----------------
|X|X|X|X|X|X|X|X|
-----------------
      byte 0
```

## 3.6. Table 6 Float variables (holding registers)

| Address | Name | Type | Access | RS232 command |
|---|---|---|---|---|
| 0x7000 | Current output constant QI setting | float | R/W | SCO |
| 0x7002 | Frequency output constant QF setting | float | R/W | SFO |
| 0x7004 | Impulse output constant QP setting | float | R/W | SPO |
| 0x7006 | Dosing constant QD setting | float | R/W | SIO |
| 0x7008 | Fixed current setting | float | R/W | SFC |
| 0x700a | Fixed frequency setting | float | R/W | SFF |
| 0x700c | Low limit value PF1 setting | float | R/W | SF1 |
| 0x700e | High limit value PF2 setting | float | R/W | SF2 |
| 0x7010 | Hysteresis setting | float | R/W | SHY |
| 0x7012 | Conversion constant for flowrate user unit | float | R/W | FFC |
| 0x7014 | Conversion constant for volume user unit | float | R/W | FVC |
| 0x7016 | Low flow cutoff | float | R/W | FLF |
| 0x7018 | Actual flowrate | float | R | RFL |
| 0x701a | Maximum flowrate | float | R | RMX |
| 0x701c | Minimum flowrate | float | R | RMN |
| 0x701e | Nominal flowrate | float | R/W | RQN |
| 0x7020 | Electroni unit (display) temperature | float | R | IT |
| 0x7022 | Nominal value of calibration point 1 | float | R/W | CX1 |
| 0x7024 | Calibration constant for calibration point 1 | float | R/W | CY1 |
| 0x7026 | Nominal value of calibration point 2 | float | R/W | CX2 |
| 0x7028 | Calibration constant for calibration point 2 | float | R/W | CY2 |

| | | | | |
|---|---|---|---|---|
| 0x702a | Nominal value of calibration point 3 | float | R/W | CX3 |
| 0x702c | Calibration constant for calibration point 3 | float | R/W | CY3 |
| 0x702e | Nominal value of calibration point 4 | float | R/W | CX4 |
| 0x7030 | Calibration constant for calibration point 4 | float | R/W | CY4 |
| 0x7032 | +5V power supply | float | R | IU1 |
| 0x7034 | +15V power supply | float | R | IU2 |
| 0x7036 | -15V power supply | float | R | IU3 |
| 0x7038 | Excitation coils resistance | float | R | ICO |
| 0x703a | Excitation coils temperature | float | R | ICT |
| 0x703c | Actual dosing volume | float | R | RDO |
| 0x703e | Minimum allowed coil temperature | float | R/W | FTL |
| 0x7040 | Maximum allowed coil temperature | float | R/W | FTH |

Float variables data format:

```
MSB                                                                   LSB
(most sign.bit)                                           (least sign.bit)
----------------- ----------------- ----------------- -----------------
|S|E|E|E|E|E|E|E|  |E|M|M|M|M|M|M|M|  |M|M|M|M|M|M|M|M|  |M|M|M|M|M|M|M|M|
----------------- ----------------- ----------------- -----------------
      byte 3            byte 2            byte 1            byte 0
```

**Where:**
**S:** sign bit where 1 is negative and 0 is positive
**E:** exponent with an offset of 127
**M:** 24-bit mantissa (stored in 23 bits)
The mantissa is a 24-bit value whose most significant bit (MSB) is always 1 and is, therefore, not stored.

### 3.7. Table 7 String variables (holding registers)

| Address | Name | Type | Access | RS232 command |
|---|---|---|---|---|
| 0x8000 | Device identification | string[10] | R | IDN |
| 0x8005 | Flowrate unit user text | string[4] | R/W | FFU |
| 0x8007 | Volume unit user text | string[4] | R/W | FVU |
| 0x8009 | Time setting | string[8] | R/W | FTM |
| 0x800d | Date setting | string[10] | R/W | FDT |

String variables data format:

String variables have defined length (see table above) and consists of ASCII characters. One byte represents one character.

## 3.8. Table 8 Double variables (holding registers)

| Address | Name | Type | Access | RS232 command |
|---|---|---|---|---|
| 0x9000 | Volume | double | R | RVO |
| 0x9004 | Volume positive | double | R | RVP |
| 0x9008 | Volume negative | double | R | RVN |
| 0x900c | Volume auxiliary | double | R | RVA |
| 0x9010 | Timed volume finished | double | R | RVT |
| 0x9014 | Timed volume actual | double | R | RTA |

Double variables data format (decimal/64 data format)

```
MSB
(most sign.bit)
----------------- ----------------- ----------------- -----------------
|S|C|C|C|C|C|E|E| |E|E|E|E|E|E|M|M| |M|M|M|M|M|M|M|M| |M|M|M|M|M|M|M|M|
----------------- ----------------- ----------------- -----------------
      byte 7            byte 6            byte 5            byte 4
                                                                  LSB
                                                      (least sign.bit)
----------------- ----------------- ----------------- -----------------
|M|M|M|M|M|M|M|M| |M|M|M|M|M|M|M|M| |M|M|M|M|M|M|M|M| |M|M|M|M|M|M|M|M|
----------------- ----------------- ----------------- -----------------
      byte 3            byte 2            byte 1            byte 0
```

**Where:**
**S:** sign bit where 1 is negative and 0 is positive
**C:** combination field
**E:** 8-bit exponent continuation with an offset of 398
**M:** 50-bit mantissa continuation

Combination field:

| **Combination field** (5 bits) | **Type** | **Exponent MSBs** (2 bits) | **Coefficient MSD** (4 bits) |
|---|---|---|---|
| a b c d e | Finite | a b | 0 c d e |
| 1 1 c d e | Finite | c d | 1 0 0 e |
| 1 1 1 1 0 | Infinity | - - | - - - - |
| 1 1 1 1 1 | NaN | - - | - - - - |

Example:

In this format, the finite number -7.50 would be encoded as follows:

- The sign is 1 indicating that the number is negative.
- The coefficient will be 750, with 13 leading zeros. This is encoded with the first digit (0) in the combination field, and the remaining 15 digits in the coefficient continuation field (four 10-bit groups of all zero bits and the final group being the encoding of 750, which is the ten bits 11 1101 0000).
- The exponent will be -2, so the encoded exponent is this plus the bias, or 396. This is 01 1000 1100 in binary, with the first two bits being embedded in the combination field and the remainder being placed in the exponent continuation field.

The bits of the combination field are therefore 01000 (the last three bits are 0 because the most significant digit of the coefficient is 0). The full encoding is therefore (in hexadecimal, shown in network byte order):

A2 30 00 00 00 00 03 D0