Communication interface RS-485 / MODBUS RTU

Electromagnetic flowmeters
FLONET FH30xx and **FLONEX FXx11x**

Content

1. INTRODUCTION

Electromagnetic flowmeters of the type series FLONET FH30xx (for normal operating environments) and FLONEX FXx11x (for explosive atmospheres) are meters intended for bi-directional measurement of volume flow rates of conductive liquids in a fully flooded piping. The flow velocity range is 0.025 – 10m/s and the minimum conductivity of the measured fluid is 10 μS/cm; for de-mineralised water 20 μS/cm.

To facilitate communication with master/plant control systems, the meters are provided with digital interfaces

- RS-485 MODBUS RTU
- HART®

This manual contains information and instructions related to interconnections between flowmeters of the type series FLONET FH30xx / FLONEX FXx11x and the master control system using the serial communication line RS-485 MODBUS RTU.

2. ABBREVIATIONS, TERMS AND DEFINITIONS

MSB	Most Significant Bit
LSB	Least Significant Bit
Master	Controlling stations
Slave	Controlled stations
Hi byte	Upper part of the xx_H register
Lo byte	Lower part of the xx_H register
xx_H	Number in hexadecimal format
xx_D	Number in decimal format
xx_B	Number in binary format

In reference to PDU, discrete inputs, coils, input and holding registers are addressed starting with 0.
For example: **c**oils designated 1 to 5 are addressed $0 – 4_D$ … $00_H – 04_H$.

MODBUS-specific data transfer method

- **Byte order in register 1** (16 bits):
  The register data transfer is facilitated using the so-called **Big Endian** (the Hi byte comes first, followed by Lo byte).
- **Register order**:
  Longer data sequences (32 and 64 bits) are transferred as a series of 16-bit registers.
  The registers in the package to be transferred (message) are arranged in the **Little Endian** format (the lowest register comes first, followed by progressively higher registers).

3. MODBUS

3.1. General description

MODBUS is a communication protocol on the level of the application layer of the reference ISO/OSI model (Open Systems Interconnection Reference Model) facilitating a master-slave kind of communication among equipment installed within different types of network and bars.

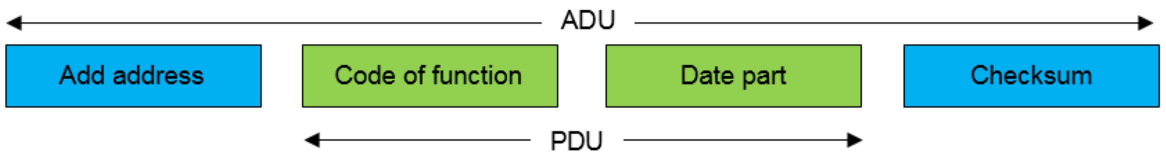
Presently support is available for various implementations including serial lines RS-232, RS-422, RS-485 and other networks using the TCP/IP protocol.

Detailed information on the MODBUS communication can be found at **http://www.modbus.org**.

A MODBUS message on the level of **PDU** (Protocol Data Unit) of the ISO/OSI model is independent of the type of the communication layer and includes:

- Function code
- Data

Depending on the type of communication network where the protocol is to be used, the PDU message is extended by additional data packages to form a message on the **ADU** (Application Data Unit) level.



3.2. MODBUS PDU message

Function code identifies the type of operation the slave station is to perform.

Code range: 1–255_D … 1–FF_H
Valid codes: 1–127_D … 1–$7F_H$

Codes 129–255 are reserved for error notification purposes. Function codes 0 and 128 are not valid codes and their use in not permitted.

The data contained in the data part of the message sent by the master station specify the details of the operation specified in turn by the function code to be performed by the slave station. The data part content depends on the required function.

The MODBUS protocol recognises three basic PDU message types:

1 PDU request

1 byte Function code
n bytes The request data part (address, variables, number of variables and others)

2 PDU response

1 byte Function code (the same as in the associated request)
m bytes The response data part (inputs read, equipment status data and others)

3 PDU exception response

1 byte Function code + 80_H (error indication)
1 byte Error code (error definition)

3.3. MODBUS ADU message

MODBUS ADU message is generated and sent by the master station as the first step in the MODBUS communication. The slave station addressed will respond to this message. Upon receipt of the response the MODBUS transaction is completed and finished.

Slave address
The address field data specify which slave station connected to the bar is addressed by the master station.

Address field:	**1 byte**	
Address:	0	**Broadcast message** addressed to all connected equipment; no response is expected
Address:	1 – 247	**Unicast message** addressed to a particular slave station; upon the request receipt and processing, the slave station will send back a response
Address:	248 – 255	Reserve

MODBUS transaction

Upon receiving a message, the slave station will process the message data and respond. Two types of response are recognised:

- **Standard response** in cases of successful message data processing and performance of the expected action
  The function code in the response shall be the same as that in the request
- **Exception response** in cases where:
  - The message content deviates from the MODBUS specifications
  - An error occurs during the response message preparation
    The response function code shall be the same as that of the associated request + 80_H

The response data field will include the error code (Exception Code) identifying the cause of the transaction failure.

Should the slave station receive an invalid or incomplete request due to a communication error, incorrect CRC or parity, it will not respond in any way. In that case the master station, upon expiry of the time limit for the response receipt, will identify the error concerned.

Maximum PDU size: **253 bytes** (256 bytes less 1 byte for slave address less 2 bytes for CRC)
Maximum ADU size: **256 bytes** (PDU_{max} + 1 byte for slave address + 2 bytes for CRC)

Checksum
The checksum (CRC) value is sent from the master station with the other message data. Upon the message receipt, the CRC value is computed using the same procedure at the slave station. Should the checksums so derived by different, the data transfer will be evaluated as erroneous. On the other hand, if both CRCs are identical, it can be assumed that the data transfer has been error-free. The probability of such conclusion depends on the type of algorithm used in the CRC value calculation. The standard method used is CRC16.

3.4. Data model

The MODBUS data model is based on four basic tables:

Table	Item type	Access	Description	
Discrete inputs	1 bit	Read only	Data received from I/O system	Example: binary input
Coils	1 bit	Read / Write	Data modifiable by application software	Ex: relay coil; both control and status indication possible
Input Registers	16-bit word	Read only	Data received from I/O system	Example: analog input
Holding Registers	16-bit word	Read / Write	Data modifiable by application software	Ex: counter; both setting and status reading possible

3.5. Addressing rules

The MODBUS protocol includes precise rules for the PDU message addressing:

- Data packages within the MODBUS messages are addressed 0 – 65535_D (0 – $FFFF_H$).
- The data block items are numbered 1 to n.

The procedure for table mapping into the slave station address space is defined by the equipment manufacturer.

3.6. Function codes

The MODBUS function codes fall within three categories:

- General functions codes:
  - Unambiguously defined;
  - Documented in public domain;
  - With guaranteed function uniqueness;
  - Approved by the company MODBUS-IDA.org.
- User-defined function codes:
  - Two ranges: 65 – 72 and 100 – 110
  - Intended for implementation of project-unspecified functions;
  - With no guaranty regarding function uniqueness.
- Reserved function codes:
  Function codes reserved for use by certain companies (inaccessible to the public).

Function codes used with flowmeters FLONET FH30xx and FLONEX FXx11x

Function code		Modbus functions	Description
1	01_H	Read Coils	Reading of the coil statuses
2	02_H	Read Discrete Inputs	Reading of one or more input statuses
3	03_H	Read Holding Registers	Reading of the holding registers
4	04_H	Read Input Registers	Reading of the input registers
5	05_H	Write Single Coil	Writing of a single coil status
6	06_H	Write Single Register	Writing of a single register status
15	$0F_H$	Write Multiple Coils	Writing of more than one coil statuses
16	10_H	Write Multiple Registers	Writing of more than one register statuses

The complete set of the encoded functions can be found on the Internet at **www.modbus.org.**

3.7. Description of code functions

3.7.1. Function 01: Read Coils

Function 01 reads the status (ON/OFF) of 1 to 2,000 coils associated with a slave station.
The request shall specify the address of the first coil and the number of coils. In the response, each byte will include the status information on 8 coils. The lowest bit of the first byte shows the status of the first coil.

Request

Address:	1 byte
Function code:	1 byte 01_H
Address of the first coil:	2 bytes 0–65535_D … 0000–$FFFF_H$
Number of coils:	2 bytes 1–2000_D … 0000–$7D0_H$
Checksum:	2 bytes CRC16

Response

Address:	1 byte
Function code:	1 byte 01_H
Number of bytes:	1 byte N
Coil statuses:	N bytes (each byte carrying status information on eight coils)
Checksum:	2 bytes CRC16
Number of bytes (N):	Number of coils : 8; unless the remainder after division is zero, N = N+1

If the request cannot be met, the slave station response message will include an error code.

Function code (1 byte)	81_H
Error code (1 byte)	01, 02, 03 or 04_H

Example

Read the coil statuses at addresses 20 – 56.

Address:	17_D … 11_H
Number of coils:	(56 − 20 + 1) = 37_D … 25_H
Initial coil address:	20_D … 14_H
Number of bytes (N):	5_D … 05_H (37 coils : 8 = 4; remainder = 5; →N = 4+1 =5)

Request

Address	Function	Data				Checksum CRC16	
		Start address		Number of coils			
11_H	01_H	Hi byte 00_H 0_D	Lo byte 14_H 20_D	Hi byte 00_H 00_D	Lo byte 25_H 37_D	Hi byte xx_H	Lo byte xx_H

Response

Address	Function	Data						Checksum CRC16	
		Number of bytes	Coils						
11_H	01_H	05_H 5_D	CD_H 27–20	$6B_H$ 35–28	$B2_H$ 43–36	02_H 51–44	22_H 59–52	Hi byte xx_H	Lo byte xx_H

In the response, each coil status is expressed in the form of a data bit field:
ON = 1
OFF = 0

In the response, LSB of the first data byte will show the status of the coil at the initial address in the request message. The statuses of the coils at the following addresses are recorded in the higher byte positions. The same principles of the coil status writing will be applied to the following data bytes.

Unless the number of coils in the response is a multiple of eight, the remaining bits in the last byte will be set at zeros. The number of bytes is the number of complete data bytes.

Assume that the statuses of coils 27 – 20 are expressed as the value of CD_H … $1100\ 1101_B$

Data	CD_H … $1100\ 1101_B$							
Coil ordinal number	27	26	25	24	23	22	21	20
Coil status	1 (MSB)	1	0	0	1	1	0	1 (LSB)

MSB of this byte is the status of the coil at the address of 27, LSB that of the coil at the address of 20.

Assume that the statuses of coils 28–35 are expressed as the value of $6B_H$ … $0110\ 1011_B$

Data	$6B_H$ … $0110\ 1011_B$							
Coil ordinal number	35	34	33	32	31	30	29	28
Coil status	0	1	1	0	1	0	1	1

The following bytes contain information on the statuses of coils 36 – 43 and 44 – 51 (see Response).

Assume that the statuses of the last coils (52 – 59) are expressed as the value of 22_H … $0010\ 0010_B$
In the associated request the last coil to be checked is coil 56.
The remaining three bits (of the ordinal numbers / addresses 57, 58 and 59) are not subject of the request and therefore are set at zeros.

Data	22_H … $0010\ 0010_B$							
Coil ordinal number	59	58	57	56	55	54	53	52
Coil status	0	0	0	0	0	0	1	0

3.7.2. Function 02: Read Discrete Inputs

Function 02 reads the status (ON/OFF) of 1 to 2,000 discrete inputs associated with a slave station.
The request shall specify the address of the first input and the number of inputs. In the response, each byte will include the status information on 8 inputs. The lowest bit of the first byte shows the status of the input at the initial address.

Request

Address:	1 byte
Function code:	1 byte 02_H
Initial input address:	2 bytes 0–65 535_D … 0000–$FFFF_H$
Number of inputs:	2 bytes 1–2 000_D … 00000–$7D0_H$
Checksum:	2 bytes CRC16

Response

Address:	1 byte
Function code:	1 byte 02_H
Number of data bytes:	1 byte N
Input statuses:	N bytes (each byte carrying information on eight discrete inputs)
Checksum:	2 bytes CRC16
Number of bytes (N):	Number of inputs : 8; unless the remainder after division is zero, N = N+1

If the request cannot be met, the slave station response message will include an error code.

Function code	1 byte	82_H
Error code	1 byte	01, 02, 03 or 04

Example

Request: read the statuses of discrete inputs at addresses 197 – 208
Address: 17_D … 11_H
Number of inputs: (208 − 197 + 1) = 12_D … $0C_H$
Initial input address: 197_D … $C5_H$
Number of bytes (N): 2_D … 2_H (12 inputs : 8 = 1; remainder = 4; →N = 1+1 =2)

Request

Address	Function	Data				Checksum CRC16	
		Start address		Number of inputs			
11_H	02_H	Hi byte 00_H	Lo byte $C5_H$	Hi byte 00_H	Lo byte $0C_H$	Hi byte xx_H	Lo byte xx_H

Response

Address	Function	Data			Checksum CRC16	
		Number of bytes	Discrete inputs			
11_H	02_H	02_H	204–197 xx_H	212–205 xx_H	Hi byte xx_H	Lo byte xx_H

In the response, the statuses of the discrete inputs are shown in the form of a data bit field, e.g.

Input status	0	1	1	1	0	1	1	1	0	0	0	0	1	1	1	1
Input address	204_D							197_D (LSB)	212_D				208_D			205_D

In the response, LSB of the first data byte will show the status of the input at the initial (start) address (197_D). Unless the number of discrete inputs in the response is a multiple of eight, the remaining bits in the last data byte will be set at zeros. The number of bytes is the number of complete data bytes.

3.7.3. Function 03: Read Holding Registers

Function 03 reads the binary content of a coherent block of 1 to 125 holding registers associated with a slave station.
Specified in the respective request are the address of the first register and the number of registers requested to be read. In the slave station response, each register will occupy the space of two bytes.

The maximum number of registers per request is 125, representing:

- 125 numbers of the word type, each of two bytes.
-

Request

Address: 1 byte
Function code: 1 byte 03_H
Initial address: 2 bytes 0–65535_D … 0000–$FFFF_H$
Number of registers (N): 2 bytes N = 1–125_D … 0000–$7D_H$
Checksum: 2 bytes CRC16

Response

Address: 1 byte
Function code: 1 byte 03_H
Number of data bytes: 1 byte 2*N
Register readings: 2*N bytes
Checksum: 2 bytes CRC16

If the request cannot be met, the slave station response message will include an error code.

Function code	1 byte	83_H
Error code	1 byte	01, 02, 03 or 04

Example

Read the register contents at the addresses 109–111 at slave station 17.
Address: 17_D … 11_H
Number of registers: (111 − 109 + 1) = 3_D … 03_H
Initial register address: 109_D … $6D_H$

Request

Address	Function	Data				Checksum CRC16	
		Start address		Number of registers			
11_H	03_H	Hi byte 00_H	Lo byte $6D_H$	Hi byte 00_H	Lo byte 03_H	Hi byte xx_H	Lo byte xx_H

Response

Address	Function	Data							Checksum CRC16	
		Number of bytes	Register							
11_H	03_H	06_H	109 Hi byte 01_H	109 Lo byte $5A_H$	110 Hi byte 00_H	110 Lo byte $1F_H$	111 Hi byte 00_H	111 Lo byte 64_H	Hi byte xx_H	Lo byte xx_H

In the response, the data in each register occupy the space of two bytes. The first byte in each register contains MSB, the other byte LSB.

Example: Register 109 reads $015A_H$ (346_D … $00000001\ 01011010_B$),
Register 110 reads $001F_H$ (542_D … $00000010\ 00011110_B$),
Register 111 reads 0064_H (100_D … $00000000\ 01100100_B$).

3.7.4. Function 04: Read Input Registers

Function 04 reads the binary content of a coherent block of 1 to 125 input registers associated with a slave station.
Specified in the respective request are the address of the first register and the number of registers requested to be read. In the slave station response, each register will occupy the space of two bytes.

The maximum number of registers per request is 125, representing:

- 125 numbers of the word type, each of two bytes.

Request

Address: 1 byte
Function code: 1 byte 04_H
Initial address: 2 bytes $0–65535_D$ … $0000–FFFF_H$
Number of registers (N): 2 bytes $1–125_D$ … $7D_H$
Checksum: 2 bytes CRC16

Response

Address: 1 byte
Function code: 1 byte 04_H
Number of bytes: 1 byte 2*N
Register readings: 2*N bytes
Checksum: 2 bytes CRC16

N … Number of input registers

If the request cannot be met, the slave station response message will include an error code.

Function code	1 byte	84_H
Error code	1 byte	01, 02, 03 or 04

Example
Read the register contents at the addresses 30009–30011 at slave station 17:
Address: 17_D … 11_H
Number of registers: (30011 − 30009 + 1) = 3_D … 03_H
Initial register address: 30009_D … 7539_H

Request

Address	Function	Data				Checksum CRC16	
		Start address		Number of registers			
11_H	04_H	Hi byte 75_H	Lo byte 39_H	Hi byte 00_H	Lo byte 03_H	Hi byte xx_H	Lo byte xx_H

Response

Address	Function	Data							Checksum CRC16	
		Number of bytes	Register							
11_H	04_H	06_H	30009 Hi byte 02_H	30009 Lo byte $2B_H$	30010 Hi byte 00_H	30010 Lo byte 00_H	30011 Hi byte 00_H	30011 Lo byte 64_H	Hi byte xx_H	Lo byte xx_H

In the response, the data in three registers occupy the space of six bytes. The first byte in each register contains the higher-order bits (Hi byte), the other byte the lower-order bits (Lo byte).

Example: Register 30009 reads $022B_H$ (555_D … $00000010\ 00101011_B$),
Register 30010 reads 0000_H (0_D … $00000000\ 00000000_B$),
Register 30011 reads 0064_H (100_D … $00000000\ 01100100_B$).

3.7.5. Function 5: Write Single Coil

Function 5 sets a selected coil in the status ON or OFF. Specified in the respective request shall be the coil address and the requested status.
A standard response is a plain copy of the request.

Request

Address: 1 byte
Function code: 1 byte 05_H
Coil address: 2 bytes 0–65535_D … 0000–$FFFF_H$
Coil status: 1 byte 00_H … OFF, FF_H … ON
Checksum: 2 bytes CRC16

Response

Address: 1 byte
Function code: 1 byte 05_H

Coil address: 2 bytes 0–65535$_D$ … 0000–FFFF$_H$
Coil status: 1 byte 00$_H$ … OFF, FF$_H$ … ON
Checksum: 2 bytes CRC16

If the request cannot be met, the slave station response message will include an error code.

Function code	1 byte	85$_H$
Error code	1 byte	01, 02, 03 or 04

Example
Coil at the address of 20 at slave station 17 is to be set to ON … FF$_H$
Address: 17$_D$ … 11$_H$
Coil address: 20$_D$ … 14$_H$

Request

Address	Function	Data			Checksum CRC16	
		Coil address		Requested data		
11$_H$	05$_H$	Hi byte 00$_H$	Lo byte 14$_H$	FF	Hi byte xx$_H$	Lo byte xx$_H$

Response

Address	Function	Data			Checksum CRC16	
		Coil address		Requested data		
11$_H$	05$_H$	Hi byte 00$_H$	Lo byte 14$_H$	FF	Hi byte xx$_H$	Lo byte xx$_H$

3.7.6. Function 6: Write Single Register

Function 6 writes data into one holding register associated with a slave station.
1 register = 2 bytes
Specified in the respective request shall be the register address and data to be written into this register.
A standard response is a plain copy of the request.

Request
Address: 1 byte
Function code: 1 byte 06$_H$
Register address: 2 bytes 0–65 535$_D$ … 0000–FFFF$_H$
Register data: 2 bytes 0–65 535$_D$ … 0000–FFFF$_H$
Checksum: 2 bytes CRC16

Response
Address: 1 byte
Function code: 1 byte 06$_H$
Register address: 2 bytes 0–65 535$_D$ … 0000–FFFF$_H$
Register data: 2 bytes 0–65 535$_D$ … 0000–FFFF$_H$
Checksum: 2 bytes CRC16

If the request cannot be met, the slave station response message will include an error code.

Function code	1 byte	86$_H$
Error code	1 byte	01, 02, 03 or 04

Example
Write the value of 0102_H … 1 00000010_B into the register at the address of 40843 at slave station 17.
Register address: 40843_D … $9F8B_H$

Request

Address	Function	Data				Checksum CRC16	
		Register address		Required data			
11_H	06_H	Hi byte $9F_H$	Lo byte $8B_H$	Hi byte 01_H	Lo byte 02_H	Hi byte xx_H	Lo byte xx_H

Response

Address	Function	Data				Checksum CRC16	
		Register address		Required data			
11_H	06_H	Hi byte $9F_H$	Lo byte $8B_H$	Hi byte 01_H	Lo byte 02_H	Hi byte xx_H	Lo byte xx_H

3.7.7. Function 15: Write Multiple Coils

Function 15 sets the status (ON or OFF) of each of 1 to 1968 coils. Specified in the respective request shall be the address of the first coil, number of coils and the required status of each coil.
A standard response to such request includes information on the initial coil address and the number of coils set at the required status.

Request

Address:	1 byte
Function code:	1 byte $0F_H$
Initial address:	2 bytes 0–65 535_D … 0000–$FFFF_H$
Number of coils:	2 bytes 1–1 968_D … $7B0_H$
Number of bytes (N):	1 byte N
Coil status information:	N bytes
Checksum:	2 bytes CRC16
Number of bytes (N):	Number of coils : 8; unless the remainder after division is zero, N = N+1

Response

Slave station address:	1 byte
Function code:	1 byte $0F_H$
Initial address:	2 bytes 0–65 535_D … 0000–$FFFF_H$
Coil status information:	2 bytes 1–1 968_D … $7B0_H$
Checksum:	2 bytes CRC16

If the request cannot be met, the slave station response message will include an error code.

Function code	1 byte	$8F_H$
Error code	1 byte	01, 02, 03 or 04

Example
At slave station 17, set the statuses of 10 coils starting with the coil at address 20 as follows:
CD 03_H … 11001101 00000001_B
Initial (start) address: 20_D … 014_H
Number of coils: 10_D … $000A_H$
Number of bytes (N): 2_D … 2_H (10 coils : 8 = 1; the remainder = 2; →N = 1+1 =2)

CD03$_H$	CD$_H$								01$_H$							
	1	1	0	0	1	1	0	1	0	0	0	0	0	0	0	1
Coils	27	26	25	24	23	22	21	20	x	x	x	x	x	x	29	28
	MSB							LSB	MSB							LSB

X means that the respective coils statuses are irrelevant where the remaining part of byte 01$_H$ (the bit positions above coil 29) are set to zeros.
The least significant bit (LSB) in byte CD$_H$ carries the required status information for coil at the address of 20.
The least significant bit (LSB) in byte 01$_H$ carries the required status information for coil at the address of 28.

Request

Address	Function	Data							Checksum CRC16	
		Start address		Number of coils		Number of bytes (N)	Required data			
11$_H$	0F$_H$	Hi byte 00$_H$	Lo byte 14$_H$	Hi byte 00$_H$	Lo byte 0A$_H$	02$_H$	Hi byte CD$_H$	Lo byte 01$_H$	Hi byte xx$_H$	Lo byte xx$_H$

Response

Address	Function	Data				Checksum CRC16	
		Start address		Number of coils			
11$_H$	0F$_H$	Hi byte 00$_H$	Lo byte 14$_H$	Hi byte 00$_H$	Lo byte 0A$_H$	Hi byte xx$_H$	Lo byte xx$_H$

3.7.8. Function 16: Write Multiple Registers

Function 16 writes a coherent block of data in up to 120 registers associated with a slave station. Specified in the respective request shall be the address of the initial register, number of registers and the data to be written into the registers.
A standard response to such request includes information on the initial register address and the number of registers filled with the required data.

The maximum number of registers per request is 120, representing:

- 120 numbers of the word type, each of two bytes

Request

Address:	1 byte
Function code:	1 byte 10$_H$
Initial (start) address:	2 bytes 0–65535$_D$ … 0000–FFFF$_H$
Number of registers (N):	2 bytes 1–120$_D$ … 78$_H$
Number of bytes:	1 byte 2*N
Required register data:	2*N bytes
Checksum:	2 bytes CRC16

Response

Address:	1 byte
Function code:	1 byte 10$_H$
Initial (start) address:	2 bytes 0–65535$_D$ … 0000–FFFF$_H$
Number of registers:	2 bytes 1–120$_D$ … 78$_H$
Checksum:	2 bytes CRC16

If the request cannot be met, the slave station response message includes an error code.

Function code	1 byte	90_H
Error code	1 byte	01, 02, 03 or 04

Example

Write data into two registers starting with the initial register address of 21 at slave station 17.

Initial address: 21_D … 15_H
Number of registers: 2_D … 0002_H
Data for register 1: 10_D … $000A_H$ … 1010_B
Data for register 2: 258_D … 0102_H … $1\ 00000010_B$

Request

Address	Function	Data										Checksum CRC16	
		Start address		Number of registers (N)		Number of bytes		Required data					
11_H	10_H	Hi byte 00_H	Lo byte 15_H	Hi byte 00_H	Lo byte 02_H	Hi byte 00_H	Lo byte 04_H	Hi byte 00_H	Lo byte $0A_H$	Hi byte 01_H	Lo byte 02_H	Hi byte xx_H	Lo byte xx_H

Response

Address	Function	Data				Checksum CRC16	
		Start address		Number of registers			
11_H	10_H	Hi byte 00_H	Lo byte 015_H	Hi byte 00_H	Lo byte 02_H	Hi byte xx_H	Lo byte xx_H

3.7.9. Exception Responses

When the master station sends a request to a slave station, it expects a response.

When a request has been sent, one of the following events may occur:

1. The slave station receives the request free of any communication error, processes the request and sends a standard response to the master station.
2. Should the slave station fail to receive the request due to a communication error, it will not respond. The master station will then apply the criterion of exceeding the time limit for request.
3. If the slave station receives a request but identifies a communication error (parity or CNC), it will send no response. The master station will then apply the criterion of exceeding the time limit for request.
4. If the slave station receives a request free of any communication error but it cannot carry out the requested action, it will send a negative (exception) response and information on the nature of the error to the master station.

Exception response structure

Slave station address	Function code	Data	Checksum
	Code + 80_H	Error code	CRC 16

Exception response contains two fields not found in the standard response:

- Error function code
- Data – Error code

3.7.9.1. Error function code

In the standard response, the function code is the same as that in the associated request. All function codes have MSB set at zero (0).
In the exception response, the most significant bit of the function code is set at 1 (added to the function code sent by the master station in its request is 80_H).
Master station will identify any exception response by that bit.

3.7.9.2. Error code

In the exception response, the slave station will identify the error by the respective code written in the data field.

Error codes used in the MODBUS protocol

Error codes		
Code	Title	Explanation
01_H	Illegal function	Required function is not supported by slave station
02_H	Illegal address	Given address is not within the range supported by slave
03_H	Illegal data	Transferred data are not valid
04_H	Equipment failure	Failure that cannot be rectified within meeting the request

4. MODBUS ON A SERIAL LINE

On the physical level 1 of the ISO/OSI reference model, different types of serial interfaces can be used, such as RS-232, RS-485 or various versions of the same.

Serial interface protocol
The MODBUS Serial Line protocol is of the Master – Slave type.
At any moment, the system bus may accommodate only one master station and 1 to 247 slave units. The communication is always started by the master station. The slave station may not send data unless authorised to do so by the master station.

Master station may send requests to slave units in one of the following operational modes:

- **Unicast** – master addresses one specific slave unit and this shall respond to the master station
- **Broadcast** – master sends a request to all slave units none of which shall respond

Master station has no address; addresses are associated with the slave stations only.

The MODBUS protocol defines two formats for the serial data communication:

- Modbus RTU
- Modbus ASCII

All units/stations connected to a particular bar shall operate in the same communication mode.

Electromagnetic flowmeters of the type series FLONET FH30xx and FLONET FXx11x use the MODBUS RTU communication mode.

4.1. MODBUS RTU

In the RTU mode, each message byte contains two four-bit hexadecimal numbers.

Every 8-bit byte (complete with start, stop and possibly parity bits) is sent as a single character.

Character format:

- 1 start bit
- 8 data bits
- 1 parity bit (optional)
- 1 stop bit

A character consists of a sequence of 10 or 11 bits.
Each slave unit shall support an odd, even or no parity.

Start	1	2	3	4	5	6	7	8	Stop

or

Start	1	2	3	4	5	6	7	8	Par	Stop

RTU frame format

Start	Slave address	MODBUS function	Data	CRC	Stop
> 3.5 characters	1 byte	1 byte	n * 1 bytes	2 bytes	> 3.5 characters

- Each message shall be sent as a continuous sequence where gaps between characters shall not be longer than 1.5 characters.
- The message beginning and end are identified as gaps/pauses on the bar longer than 3.5 characters.
- Should a pause (idle interval) longer than 3.5 characters occur prior to the transaction completion, the receiving station will cancel the preceding message and take the following byte for the address of a new message.
- The master station works with a pre-defined time interval before it cancels a transaction.

Data field

The data field contains the information needed by the master and slave stations to implement the required MODBUS function.

Checksum

The message integrity is ensured and verified by means of a checksum of the CRC type (and the parity bit, if used).
In the flowmeters of the type series FLONET FH30xx and FLONET FXx11x, this control feature consists of CRC16 (a polynomial of the 16th degree). For more detailed information and recommendations regarding the MODBUS communication, see:
http//: www.modbus.org.

The CRC16 polynomial makes it possible to identify errors in data packages of a higher-bit format (e.g. in 16-bit records) with probability in excess of 99.99%.
With the RTU mode, the 16-bit CRC value is expressed by two bytes. Byte CRC_Hi is transferred first, followed by byte CRC_Lo.
The CRC field is attached to the message as the last field.

The character and message checks/verifications are carried out in both master and slave stations prior to the message transfer.
The slave station checks each character and the whole message during the receiving operation.

Data transfer in accordance with the MODBUS specifications

- **Byte order within a register** (16 bit):
  The data contained in a register are transferred in the format of **Big Endian** (the Hi byte comes first, followed by the Lo byte).
- **Register order:**
  Longer data messages (32 and 64 bits) are transferred as a series of 16-bit registers.

The registers within a message to be transferred are arranged as **Little Endian** (starting with the lowest register).

Transfer of data contained in 32- and 64-bit registers

32 bit

	16 bit register		16 bit register	
	reg_Lo (bit 15...0)		reg_Hi (bit 31...16)	
	Hi Byte	Lo Byte	Hi Byte	Lo Byte
Transfer order	1	2	3	4

64 bit

	16 bit register		16 bit register		16 bit register		16 bit register	
	reg_Lo (bit 15...0)		reg_Hi (bit 31...16)		reg_Lo (bit 47...32)		reg_Hi (bit 63...48)	
	Hi Byte	Lo Byte	Hi Byte	Lo Byte	Hi Byte	Lo Byte	Hi Byte	Lo Byte
Transfer order	1	2	3	4	5	6	7	8

5. RS-485 interface

The RS-485 interface is intended to provide serial communication among multiple units in a network of a bus-type topology. The interface parameters are specified in the documents TIA/EIA-485.
The communication line is of a half-duplex type where the equipment connected to the line transmit messages in turns (at any time, only one communication unit may send a message).
The simplest version of the system consists of the permanently activated (master) station and up to 31 receiving (slave) stations.

5.1. Logic levels

The logic levels are defined in reference to the recommendations by MODBUS as a voltage difference between two conductors, conventionally designated A and B.
At the standstill condition, the voltage on conductor A is lower than that on conductor B. The logic status detection based on the voltage difference between the two conductors is preferable as it makes possible elimination of the effects of spurious signals that are usually induced in both conductors in the same magnitudes.

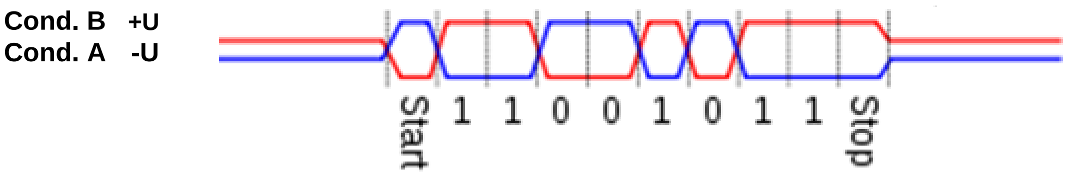
Decisive levels

Logic 1: $U_B - U_A > 200$ mV
Logic 0: $U_B - U_A < -200$ mV

Maximum bar length

The maximum length of the RS-485 line depends on the communication speed.
With communication speed up to 19,200Bd it is recommended to observe the limit on the line length of 1,200m. For higher communication speeds the maximum line length becomes progressively shorter.

Example: transmission of number 203_D (CB_H … 11001011_B), no parity.

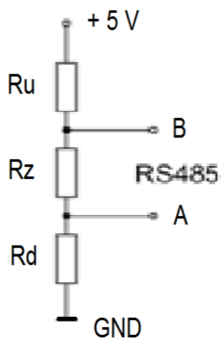


At the line standstill condition, when no station is transmitting, the circuits are disconnected and at high-impedance status (logic 1).
The message commences with the start bit (log 0), followed by eight data bits and the stop bit (log 1). The total number of bits transmitted is 10.

Standstill	1
Start bit	0
Stop bit	1

5.2. Line termination

Standard EIA/TIA RS-485 requires that the ends of communication line RS-485 be impedance matched to avoid signal reflections on the bar. It is therefore necessary to install in the end units termination resistors of magnitudes close to the characteristic cable impedance (typically 120 Ω).
As long as no equipment connected to the communication line is in the active operational mode, all RS-485 circuits are at high-impedance status.
Interference signals, even low-level ones, may generate at some parts of the bar induced voltage between the data conductors of magnitudes exceeding the decisive levels, whereby some stations may commence erroneous data reception.
To ensure safe potential conditions on conductors A and B at standstill, it is recommended to install on the bar the so-called active termination element consisting of termination resistor Rz, pull-up resistor Ru and pull-down resistor Rd. The active termination element is often incorporated in the RS-485/USB convertor.
Do not use more than one active termination element on the communication line.

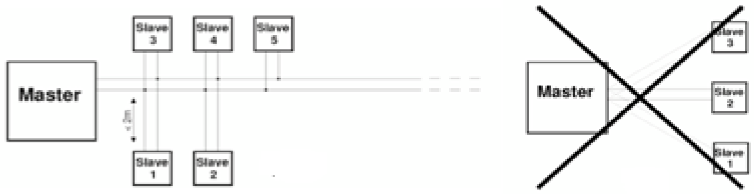


Ru, Rd	… approx.	390–820Ω
Rz	… approx.	120Ω

5.3. Topology

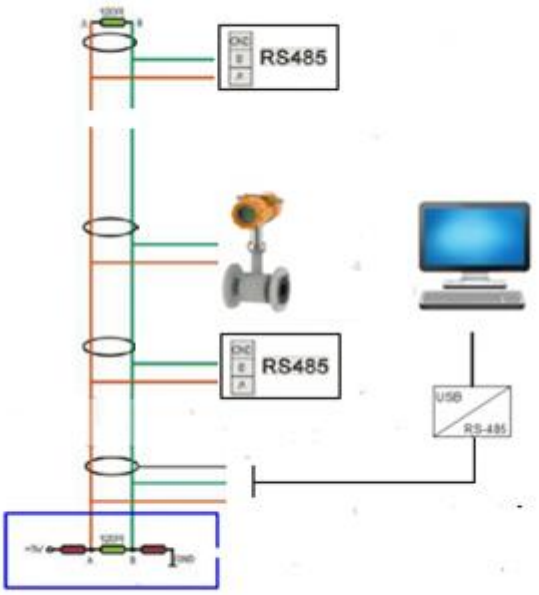
The RS-485 communication bar shall be a line structure with two end units.
The master and slave stations shall be connected to this line structure directly or, if necessary, by means of branch lines of which none shall be longer than 3m. Each branch line shall lead to no more than one station. Stations connected by means of branch lines are not considered end units.
The master station can be connected to any part of the bar – at the end or anywhere between the line ends.

Should the necessity to use a longer branch line arise, use a double cable leading from the branching point to the master or slave station and back to the same branching point. In this arrangement, the line topology of the bar will be maintained.



Correct Wrong

Connections of master/slave stations to the RS-485 line



5.4. Potential equalising

In practical situations, the ground potentials at the points where individual master and slave stations are installed may vary and exceed the permitted parameter limits associated with the electronic communication equipment connected to the RS-485 communication line.
To ensure correct functioning of the stations on the RS-485 bar, it is necessary to provide for a reliable conductive interconnection among the grounds of all master and slave stations. Such interconnection can be carried out by means of the shielding on the communication line or by a separate conductor. In any case, the RS-485 bar should be a shielded cable. The cable shielding shall be connected to the ground potential at one point only (at the master station).
Recommended cable: a shielded twisted pair of conductors of cross-section 0.35 to 0.8mm^2 and impedance about 120Ω. Best are cables designed and manufactured specifically for use as RS-485 lines.

Possible causes of data transmission errors:

- Termination resistor missing
- Active termination element missing
- Reversed polarity of conductors A, B of the communication bar
- Incorrect setting of the data transfer speed
- Incorrect stop bit setting
- Master and slave units have not been set at the same operational mode (RTU)
- Incorrect slave station address

6. COMMUNICATION INTERFACE RS-485 / MODBUS RTU

Flowmeters of the type series FLONET FH30xx and FLONEX FXx11x are provided with the RS-485 / MODBUS RTU communication interface.
Using this interface, the flowmeter can be connected to a personal computer or other similar control equipment.

Communication interface: technical specifications

MODBUS equipment type:	Slave
Standard:	Modbus v 1.02, level 7 of the reference ISO/OSI model
Protocol:	RS-485 MODBUS RTU according to ČSN EN/IEC 61158
Supported functions: MODBUS	01_H Read Coils 02_H Read Discrete Inputs 03_H Read Holding Registers 04_H Read Input Registers 05_H Write Single Coil 06_H Write Single Register 15_H Write Multiple Coils 16_H Write Multiple Registers
Slave address range:	1–247, initial setting 4
Stations on the line:	max. 32
Physical interface:	RS-485 according to EIA/TIA-485, electrically insulated
Character format:	Start bit 1 bit Data 8 bits No parity Stop bit 1 bit
Transfer speed:	1,200, 2,400, 4,800, 9,600, 19,200, 38,400, 57,600Bd Initial setting: 9,600Bd
Maximum bar length:	1,200m for data transfer speed 19,200Bd
Communication mode Master/slave	Unicast
Software master:	Computer with Windows 7 or higher upgrade (Linux, iOS), JAVA 8u40 or higher upgrade, and implemented MODBUS RTU protocol as of this manual Communication software program FLOSET 4.0 Configuration file *.flo,
Connection:	2 conductors + GND

FH30xx/FXx11x	Bar conductor
1	A
2	B
PE	Shielding

Bar polarity:	A -, B +
Cable:	Type A according to standard ČSN IEC 61158 – 2 (a twisted pair of conductors, shielding with 90% covering)

7. VARIABLE MAPPING INTO ADDRESS SPACE

The procedures of mapping variables into the MODBUS address space for electromagnetic flowmeters FLONET FH30xx and FLONEX FXx11x are described is document Es 90694K

8. DOCUMENTATION AND STANDARDS

Standard
EN/IEC 61158-2

Product manuals

Es 90678K	Electromagnetic flowmeter FLONET FH30xx
Es 90686K	Electromagnetic flowmeters FLONEX FXx11x a FLONET FH30xx; meter control instructions
Es 90694K	Electromagnetic flowmeters – general FLONET FH30xx and FLONEX FXx11x; variable mapping into the MOBUS address space

Ronešová, A MODBUS protocol review

http://www.modbus.org.

Manufacturer's address:

ELIS PLZEŇ a. s.
Luční 425/15,
301 00 Plzeň
Czech Republic
Tel.: +420/377 517 711
Fax: +420/377 517 722
E-mail: sales@elis.cz
http://www.elis.cz

Issue No 1