Communication protocols in the TE485 transmitter

Spinel and Modbus RTU

TE485 - communication

Datasheet

Created: 13.5.2013

Latest update: 11.11.2024 13:10

Number of pages: 32

Papouch s.r.o.

Address:

Strašnická 3164/1a
102 00 Prague 10

Phone:

+420 267 314 268

Web:

papouch.com

Mail:

papouch@papouch.com

TABLE OF CONTENTS

History of changes to this document

7/2023 - firmware version 11

- Added 3 mV/V sensitivity - see complete list of sensitivities in Tab. 1 on page 17.
- New option to set the measurement speed (Spinel: page 18, Modbus: Holding register 22).

DESCRIPTION

This document describes the spinel and Modbus RTU communication protocols in the TE485. Documentation of the converter hardware and function descriptions are available at papouch.com (detailed documentation is available for download in PDF format).

Communication parameters

Communication speed ..... adjustable from 300 Bd to 230400 Bd (default: 9600 Bd)

Number of data bits 8

Parity ..... none

Number of stop bits 1

How to find out the current communication parameters?

If you have older hardware that has a connector from Fig. 1, shorting the two highlighted pins switches the TE485's current communication parameter settings to the serial link. This information is always sent in the Spinel protocol, at 9600 Bd.

First a packet is sent with the response to the instruction *Reading the name and version* and then a packet containing the address, speed and protocol in the data in ASCII format. Example:

**a? "4N?Address:34 Speed:6 Protocol:1ü?*

The address is hexadecimal, the rate is the code according to the instruction *Setting communication parameters* and protocol is the protocol number according to the instruction *Switch*.

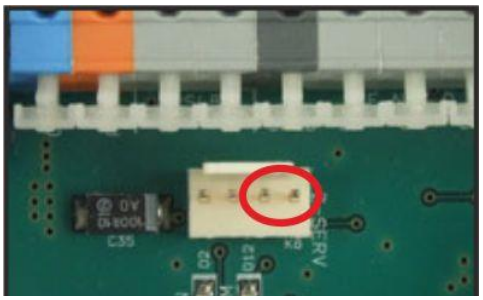


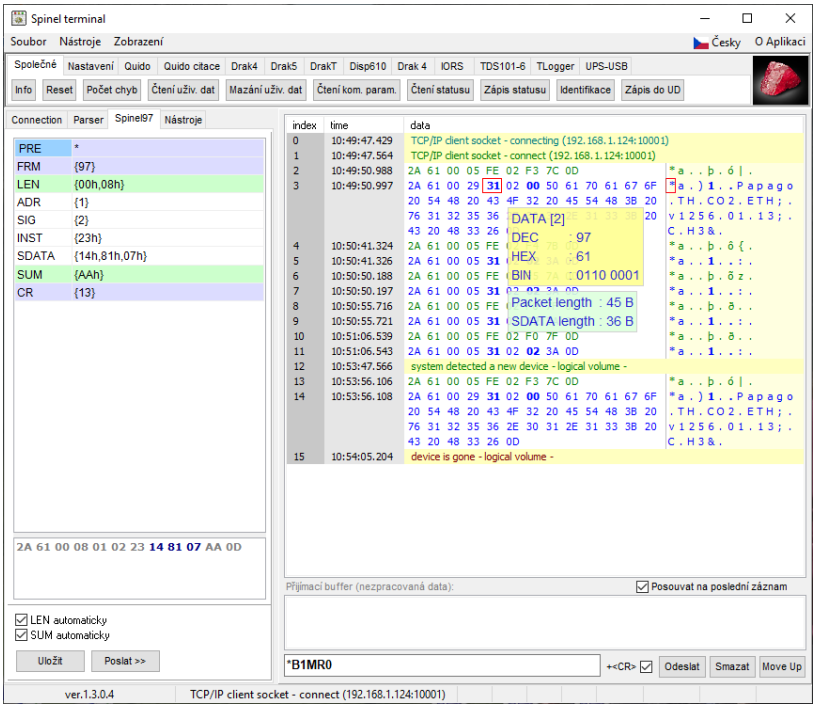
Fig. 1 - Z-SERV connector

Spinel

Spinel terminal: For easier debugging of Spinel devices, the Spinel Terminal program is available for free download at papouch.com/spinel. It allows is to communicate over serial ports and Ethernet, using the binary protocol Spinel (Format 97).

Online parser: Here on the web we have an online parser for Spinel.

Spinel.NET: Spinel.NET is a .NET API for working with Spinel. It is available here on GitHub.



FULL DESCRIPTION OF THE COMMUNICATION PROTOCOL SPINEL

Format 97

The device communicates using the Spinel binary protocol (format 97). It uses binary 8-bit characters in communication (decadic in the range 0 to 255, hexadecimal 0x00 to 0xFF).

For developers, there is a convenient program Spinel Terminal (for Windows) and also an online parser and validator for Spinel.

The following are two typical examples of a request and response structure. The first line contains the names of the individual bytes or groups and the second line contains a specific example of the request or response.

→ Request:

PRE	FRM	NUM	NUM	ADR	SIG	INST	[DATA]	SUM	CR
2A	61	00	07	31	02	61	38, E6	BB	0D

← Response:

PRE	FRM	NUM	NUM	ADR	SIG	ACK	[DATA]	SUM	CR
2A	61	00	05	31	02	00		3C	0D

field	*Length in bytes*	***Description***
PRE	1	**Prefix. Always 0x2A, character *, decadic 42.**
FRM	1	**Format number 97. Always 0x61, character a, decadic 97.**
NUM	2	**The number of bytes in the instruction from the next byte to the end of the message (i.e. ADR to CR).**
ADR	1	**The address of the device to which the request is sent or which sends the response.**
SIG	1	Message **signature**. The SIG sent in the request will be returned in the reply. It is thus possible to know to which request the response belongs.
INST	1	**Instruction code** (0x10 to 0xFF).
ACK	1	**Request acknowledgement** (0x00 to 0x0F) informs whether the request was accepted, an error occurred, it is an automatic message, etc. The list of standard ACKs is given below.
DATA	x	**Message Data**. The length of the field and the content varies according to the specific instruction.[1]
SUM	1	**Checksum.** It is calculated as follows: `SUM = 0xFF – ((PRE + FRM + ((NUM & 0xFF00) >> 8) + (NUM & 0xFF) + ADR + SIG + ACK_INST + DATA) & 0xFF)`
CR	1	**Terminating character** (Carriage Return, \r). Always 0x0D, decadic 13.

- **NUM:** Number of bytes from ADR (inclusive) up to CR (inclusive). This is two bytes, so the NUM can be up to 65535. The minimum is 5, which corresponds to an instruction that contains no data. If the NUM is less than 5, the packet is not valid. The upper byte is the MSB, the lower byte is the LSB. If the number of bytes is less than 256, the upper byte is null.

[1] If one or more parameters in the data are enclosed in square brackets [ ], the enclosed group of data may be repeated.

- **ADR:** The device address can be in the range 0x00 to 0xFD (253). The following addresses are reserved for special use:
  - 0xFF (255) is broadcast. This means that if the device receives a message with this address, the device behaves as if it were its own address, executes the instruction, but sends no response. No configuration can be performed with this address.
  - 0xFE (254) is a universal address. If a device receives a message with this address, the device acts as if it were its address, executes the instruction, and sends a response. The universal address can only be used if there is only one device on the communication link. Configuration cannot be done with this address.
- **ACK** (acknowledge) is in the same place in the response as it is in the INST request. It is in the range 0x00 to 0x0F. This byte informs the device about the status of the last received instruction. The reserved ACKs are as follows:
  - 0x00 All OK: The instruction has been received and executed.
  - 0x01 General error: unspecified error.
  - 0x02 Unknown INST code: the device does not know the instruction code.
  - 0x03 Data error: the DATA has an unexpected length or contains an unexpected value.
  - 0x04 Not allowed for any of the following reasons:
    - Attempt to change the settings without first *enabling the configuration*.
    - Attempting to write to inaccessible memory.
    - The required function of the device cannot be performed because the conditions for this are not met. For example, a higher communication speed is required.
    - Access to password-protected memory.
  - 0x05 Malfunction:
    - The device requires service intervention.
    - Device internal memory or settings memory error.
    - An error in an internal peripheral of the device.
    - Any other error affecting the proper functioning of the device.
  - 0x06 No data available: for example, shortly after switching on the device, readings from external sensors etc. may not yet be available.
  - 0x0A to 0x0F are messages that the device has sent automatically without any request from the higher-level system. For example, change notifications on input, periodic measurements, logs, etc.
- **SUM** is the checksum. A message with an incorrect checksum is not responded to. A CR is pending even if an incorrect checksum is received.

Examples

- The examples below are in hexadecimal format. [2]

[2] (Unless otherwise stated.) Hexadecimal means that the decade number 142 is given as 8E, number 11 as 0B. For more information on hexadecimal representation of numbers, see, for example, the Wikipedia article Hexadecimal system.

Format 66

Format 66 uses only decadic variables or characters that can be typed on a regular keyboard. This format is therefore useful when debugging applications with Spinel. There must be no more than a 5 sec delay between characters. Instructions are divided into request response:

Structure

Request:

```
  FOR FRM ADR INSTDATA   ...
CR
```

Response:

```
  FOR FRM ADR ACKDATA    ...
CR
```

PREPrefix , 2AH (sign "*").

FRMNumber of format 66 (character "B").

ADRAdress of the module to which the request is sent or which sends the response.

INST The instruction code of the device. These are the ASCII codes for the letters "A" to "Z" and "a" to "z" and the digits "0" to "9". The module instructions are described in detail in chapter **Chyba! Nenalezen zdroj odkazů.** on page **Chyba! Záložka není definována.**.

ACKAcknowledge whether and how the request was executed. ACKs are from the interval 00H to 0FH.

DATADATA . ASCII expression of the transferred variables. It is recommended to transfer the data in common format and units. It must not contain a prefix or CR. **Chyba! Nenalezen zdroj odkazů.** (page **Chyba! Záložka není definována.**) for each instruction.

CREnding character (0DH).

Explanatory notes

Example – one-time measurement

Address (ADR)

An address is a single character that uniquely identifies a particular device among others on a single communication link. A device always uses this number to identify itself in responses to queries from the parent system. The address may be the following ASCII characters: digits '0' to '9', lower case 'a' to 'z' and upper case 'A' to 'Z'. The address shall not be the same as a prefix or CR.

The address "%" is reserved for "broadcast". If the address "%" is in the request, the device behaves as if its address is specified. No response is returned to requests with this address.

The "$" address is a universal address. If the address "$" is requested, the device behaves as if its address is specified. In response, the device will provide the actual address just set. The universal address is only used when only one device is connected to the link.

[3] **For the example** instructions in the chapter **Chyba! Nenalezen zdroj odkazů. the terminating character <CR> is not printed!** (It is replaced by↵ .)

Instruction code (INST)

The instruction code of the respective device.

If a valid instruction is received (ADR accepted) and the received message flag is set, the device must already respond to the instruction.

Acknowledgement of request (ACK)

The ACK informs the parent device how the received instruction was processed. Acknowledgement codes:

0 ..... EVERYTHING IN ORDER
The instruction was received in order and completely executed.

1 ..... OTHER ERROR
Unspecified device error.

2 ..... INVALID INSTRUCTION CODE
The received instruction code is unknown.

3 ..... INVALID DATA
The data does not have a valid length or contains an invalid value.

4 ..... WRITE/ACCESS DENIED
- The request was not executed because certain conditions were not met.
- Attempting to write data to inaccessible memory -
Attempting to activate a device function that requires a different setting (e.g. higher communication speed) -
Attempting to change the configuration without first enabling the setting -
Accessing password-protected memory.

5 ..... EQUIPMENT FAILURE
- Equipment failure requiring service intervention.
- Error in the device's internal memory or settings memory -
Error in one of the device's internal peripherals (runtime error or initialization error) -
Any other error affecting the correct functioning of the device.

6 ..... NO DATA AVAILABLE

D ..... AUTOMATICALLY SENT INSTRUCTION - CHANGE OF DIGITAL INPUT STATE

E ..... AUTOMATICALLY SENT INSTRUCTIONS - CONTINUOUS MEASUREMENT
- Periodic sending of measured values.

F ..... AUTOMATICALLY SENT INSTRUCTION - EXCEEDING THE LIMITS OR RANGE

Data (DATA)

Instruction data.

COMPLETE LIST OF INSTRUCTIONS

Measurements

Recalculated value

This instruction reads the last measured value, recalculated according to the specified *Zero calibration* a *Calibration of the upper measurement limit*. [4]

Request:

Instruction code: 51H

Response:

Confirmation code: ACK 00H

Parameters: (ch)(status)(value)

chn	Channel number	length: 1 byte
Here always 01H.		

status	Measured value status	length: 1 byte

[4] Until it is set *Zero calibration* a *Calibration of the upper measurement limit*, the transmitter sends the same reading in this instruction as in the *Normalized RAW* value!

Status of measured values.	
bit 3.2	00 = the measured value is within the measuring range
	01 = measured value is less than the lower limit of the measuring range (underflow)
	10 = exceeding the upper limit of the measuring range (overflow)
bit 7 (MSb)	0 = the measured value is invalid
	1 = the measured value is valid

value	Measured value	Length: 2 bytes
Measured value as 16 bit signed integer. The bytes are in the order MSB:LSB.		

Examples:

Request:
2AH,61H,00H,05H,31H,02H,51H,EBH,0DH
Examples of responses:
2AH,61H,00H,09H,31H,02H,00H,01H,80H,62H,D3H,82H,0DH
The measured value 62D3H is valid and represents the number 25299 (decadic).
2AH,61H,00H,09H,31H,02H,00H,01H,80H,9DH,5EH,BCH,0DH
The measured value 9D5EH is valid and represents the number -25250 (decadic).
2AH,61H,00H,09H,31H,02H,00H,01H,04H,80H,00H,B3H,0DH
The measured value is not valid and indicates an underflow of the range.
2AH,61H,00H,09H,31H,02H,00H,01H,08H,7FH,FFH,B1H,0DH
The measured value is not valid and indicates a range overflow.

In 66 format:

Request: "MR0" *(Measure Read)*

Response: (ACK "0") (ch) (stat) (val)

Legend: (ch) 2 characters; space character and channel digit

(stat) 3 characters; a space character and two status characters for the measured value:

80 ... the measured value is valid

88 ... exceeding the upper limit of the measured range

(val) 2 to 6 characters; character space and measured value

Example: *Request:*

**B1MR0↵*

Response:

**B10 1 80 -25248↵*

Normalized RAW value

Here the upper 16 bits of the A/D converter can be read, with only the sensitivity constant applied.

Request:

Instruction code: 5FH

Response:

Confirmation code: ACK 00H

Parameters: (ch)(status)(value)

chn	Channel number	length: 1 byte
Here always 01H.		

status	Measured value status	length: 1 byte
Status of measured values.		
bit 3.2	00 = the measured value is within the measuring range	
	01 = measured value is less than the lower limit of the range (underflow)	
	10 = overflow of the upper limit of the range	
bit 7 (MSb)	0 = the measured value is invalid	
	1 = the measured value is valid	

value	Measured value	Length: 2 bytes
Measured value without conversion as 16 bit signed integer. The bytes are in the order MSB:LSB.		

Examples:

Request:
2AH,61H,00H,05H,31H,02H,5FH,DDH,0DH
Examples of responses:
2AH,61H,00H,09H,31H,02H,00H,01H,80H,62H,D3H,82H,0DH
The measured value 62D3H is valid and represents the number 25299 (decadic).
2AH,61H,00H,09H,31H,02H,00H,01H,80H,9DH,5EH,BCH,0DH
The measured value 9D5EH is valid and represents the number -25250 (decadic).
2AH,61H,00H,09H,31H,02H,00H,01H,04H,36H,30H,CDH,0DH
The measured value is not valid and indicates an underflow of the range.
2AH,61H,00H,09H,31H,02H,00H,01H,08H,C9H,F8H,6EH,0DH
The measured value is not valid and indicates a range overflow.

In 66 format:

Request: "RR0" *(Measure Read)*

Response: (ACK "0") (ch) (stat) (val)

Legend: (ch) 2 characters; space character and channel digit

(stat) 3 characters; a space character and two status characters for the measured value:

80 ... the measured value is valid

08 ... exceeding the upper limit of the measured range

04 ... drop below the lower limit of the measured range

(val) 2 to 6 characters; character space and measured value

Example: *Request:*

**B1RR0↵*

Response:

**B10 1 80 -25248↵*

Communication line configuration and address settings

Enabling configuration

This instruction enables the configuration. It must immediately precede some instructions for setting communication parameters. After the following instruction (even an invalid one) the configuration is automatically disabled.

It is not possible to use a universal address for this instruction. The address of the specific device must always be specified.

Request:

Instruction code: E4H

Response:

Confirmation code: ACK 00H

Examples:

Request:
2AH,61H,00H,05H,01H,02H,E4H,88H,0DH
Enable configuration.
Response:
2AH,61H,00H,05H,01H,02H,00H,6CH,0DH
Command acknowledged.

In 66 format:

Request: "E" *(Enable)*

Response: (ACK "0")

Example: *Request*

**B1E↵*

Response

**B10↵*

Setting communication parameters

This command sets the Spinel address and communication rate.

It is not possible to use a universal address for this instruction. In case the address is unknown and no other device is connected to the link, the address can be found out with the "Reading communication parameters". (Use the universal address FEH as the device address.) If this is not possible (there are other devices on the same communication link), you can assign an address to the device using the "Setting the address by serial number" (page 15).

Before setting the configuration parameters, the instruction Enabling configuration must precede (page 12).

Request:

Instruction code: E0H

Parameters: (address) (speed)

address	New device address	length: 1 byte
New device address in the Spinel protocol. The address can be from the interval 00H to FDH. If protocol 66 is also used for communication, it is necessary to use only addresses that can also be expressed as a displayable ASCII character (see paragraph Address on page 7). Default address: 31H		

speed	New communication speed		length: 1 byte
This parameter sets the new communication speed of the device. The default communication speed is 9 600 Bd. The communication speed codes are in the table on the right:	Speed [Bd]	Format 97	Format 66
	1 200	03H	3
	2 400	04H	4
	4 800	05H	5
	9 600	06H	6
	19 200	07H	7
	38 400	08H	8
	57 600	09H	9
	115 200	0AH	A

Response:

Confirmation code: ACK 00H

The new address and communication speed will be set after the reply is sent.

Examples:

Request:
2AH,61H,00H,07H,01H,02H,E0H,02H,0AH,7EH,0D
Setting address 02H and communication speed 115200 Bd.
Response:
2AH,61H,00H,05H,01H,02H,00H,6CH,0DH
The new address and communication speed will be set after the reply is sent.

In 66 format:

Request: "AS"(address) [5] (Address Set)

Response: (ACK "0")

Legend: (address) See Address on page 7.

[5] The address and communication rate must be set by two different instructions in protocol 66. (For protocol 97, it is only one instruction.)

Example: *Request: Address 4*

**B1AS4↵*

Response

**B10↵*

Request: "SS"(code)[5] (Speed Set)

Response: (ACK "0")

Legend: (code) Communication speed code according to the table for the outgoing speed parameter

Example: *Request: speed 19200Bd (code 7)*

**B1SS7↵*

Response

**B10↵*

Reading communication parameters

This command reads the address and communication speed of the device. Use this instruction to find the set address when it is not known. The request is sent to the universal address FEH. If the communication speed is also unknown, all communication speeds of the device must be tried. No other devices must be connected to the link when the device address is determined using the universal address.

Request:

Instruction code: F0H

Response:

Confirmation code: ACK 00H

Parameters: (address) (speed)

address	Device address	length: 1 byte
Address of the device in the Spinel log.		

speed	Communication speed		length: 1 byte
Communication rate code.	Speed [Bd]	Code for format 97	Code for format 66
The communication speed codes are in the table on the right:	1 200	03H	3
	2 400	04H	4
	4 800	05H	5
	9 600	06H	6
	19 200	07H	7
	38 400	08H	8
	57 600	09H	9
	115 200	0AH	A

Examples:

Request:
2AH,61H,00H,05H,FEH,02H,F0H,7FH,0DH
Reading communication parameters with universal address FEH.
Response:
2AH,61H,00H,07H,04H,02H,00H,04H,06H,5DH,0DH
Address 04H, communication speed 9600 Bd.

In 66 format:

Request: "CP" (Comm Parameter)

Response: (ACK "0")(address)(speed)

Legend: (address) See Address on page 7.

(speed) Communication speed code according to the table for the speed parameter.

Example: *Request with universal address:* `*$1CP↵`

Response - Address B, speed 9600Bd (code 6): `*B10B6↵`

Setting the address by serial number

The instruction allows you to set the address according to the unique serial number of the device. This instruction is handy in case the parent system or operator loses the address of a device that is on the same communication link with other devices.

The serial number is indicated on the device in the form *[product-number].[hardware-version].[software-version]/[serial-number]* for example as follows: *0227.00.03/0001*

Request:

Instruction code: EBH

Parameters: (new_address)(product_number)(serial_number)

new_address	New address of the device	length: 1 byte
New device address in the Spinel protocol.		

product_number	Product number	Length: 2 bytes
The product number listed on the label on the device. For device number 0227.00.03/0001, this is number 227.		

serial_number	Product serial number	Length: 2 bytes
The serial number of the product as shown on the label on the device. For device number 0227.00.03/0001, this is number 1. This number can also be found by the instruction "Reading production data" (see page 21).		

Response:

Confirmation code: ACK 00H

Examples:

Request:
2AH,61H,00H,0AH,FEH,02H,EBH,32H,00H,C7H,00H,65H,21H,0DH
New address 32H, product number 199 (= 00C7H), product serial number 101 (= 0065H).
Response:
2AH,61H,00H,05H,32H,02H,00H,3BH,0DH
The address has been changed - the device responds with the new address.

Calibration

How to proceed with calibration?

1) Adjust the sensitivity of the strain gauge.
2) Set to zero.
3) Calibrate the upper limit of the range.

Reading calibration constants

The instruction reads the calibration constants including how many ticks correspond to the minimum and maximum range.[6]

Request:

Instruction code: 13H

Response:

Confirmation code: ACK 00H

Parameters:(sensitivity)(zero)(r-hour)(r-parts)

Sensitivity	Sensitivity	Length: 2 bytes
The sensitivity code for which the calibration is performed. The available options are in Tab. 1 on page 17.		

zero	Zero calibration	Length: 2 bytes
Zero offset from the beginning of the range. The default value is 8000H.[7]		

r-hod	RAW under load	Length: 2 bytes
RAW value under calibration load. The default value is FFFFH.[7]		

r-dilky	RAW in parts	Length: 2 bytes
A value in parts that indicates the calibration load. The default value is FFFFH.[7]		

Examples:

Request:
2AH,61H,00H,05H,31H,02H,13H,29H,0DH
Response:
2AH,61H,00H,0DH,31H,02H,00H,00H,00H,80H,00H,FFH,FFH,FFH,FFH,B8H,0DH
Sensitivity 2 mV/V, other parameters are in default state.

[6] Calibration is tied to a specific sensitivity setting. So first you need to set the sensitivity and then the calibration. If the sensitivity is changed, the calibration setting is cancelled and a new calibration must be performed for the new sensitivity setting. As long as it is not set Zero calibration a Calibration of the upper measurement limit, the transmitter sends the following instruction Recalculated value the same value as in the instruction Normalized RAW value.

[7] If one of the three parameters is in the default value, the recalculated values always the same as RAW.

Sensitivity setting

The sensitivity of the strain gauge can be set to one of the options according to Tab. 1. [6]

Sensitivity [mV/V]	**Sensitivity code** (hex)
2 *(default)*	00H
3	03H
5	01H
10	02H

Tab. 1 - Strain gauge sensitivity codes

Request:

Instruction code: 14H

Parameters: (sensitivity)

Sensitivity	Strain gauge sensitivity	length: 1 byte
The sensitivity code for which the calibration is performed. Available options are in Tab. 1 on page 17.		

Response:

Confirmation code: ACK 00H

Examples:

Request:
2AH,61H,00H,06H,31H,02H,14H,01H,26H,0DH
Setting the strain gauge sensitivity to 5 mV/V.
Response:
2AH,61H,00H,05H,31H,02H,00H,3CH,0DH
The setup went fine.

Reading the set sensitivity

Read the strain gauge sensitivity setting. [6]

Request:

Instruction code: 15H

Response:

Confirmation code: ACK 00H

Parameters: (sensitivity)

Examples:

Request:
2AH,61H,00H,05H,31H,02H,15H,27H,0DH
Reading the set range sensitivity.
Response:
2AH,61H,00H,06H,31H,02H,00H,01H,3AH,0DH
The range is set to 5 mV/V. (The ranges are described in the previous instruction.)

Setting measurement speed

Request:

Instruction code: 16H

Parameters: (speed)

speed	Measurement speed	length: 1 byte
Measurement speed code - 00H for a default speed of 6.25 SPS or 01H for a speed of 50 SPS.		

Response:

Confirmation code: ACK 00H

Examples:

Request:
2AH,61H,00H,06H,31H,02H,16H,01H,24H,0DH,
Speed setting 50 SPS.
Response:
2AH,61H,00H,05H,31H,02H,00H,3CH,0DH
The setup went fine.

Reading the set measurement speed

Request:

Instruction code: 17H

Response:

Confirmation code: ACK 00H

Parameters: (speed)

Examples:

Request:
2AH,61H,00H,05H,31H,02H,17H,25H,0DH
Reading the set measurement speed.
Response:
2AH,61H,00H,06H,31H,02H,00H,01H,3AH,0DH
The speed is set to 50 SPS.

Zero calibration

Sets a zero value. This instruction can be used in one of the following two situations:

- The strain gauge is connected and is just in the position to be considered as zero.
- A constant for zero has been found by RAW measurement or calculation and needs to be entered.

Request:

Instruction code: 11H

Parameters: [const]

const	Constant (optional parameter)[8]	Length: 2 bytes
If this constant is not specified[8], the current strain gauge load is stored as zero. Otherwise, it is the number of converter parts that corresponds to the zero strain gauge load.		

Response:

Confirmation code: ACK 00H

Examples:

Request - without entering a value:
2AH,61H,00H,05H,31H,02H,11H,2BH,0DH
Request - with manual value entry:
2AH,61H,00H,07H,31H,02H,11H,15H,90H,84H,0DH
Response:
2AH,61H,00H,05H,31H,02H,00H,3CH,0DH

Calibration of the upper measurement limit

Sets the upper limit of the measurement. This instruction can be used in one of the following two situations:

- The strain gauge is connected and is in a position that corresponds to at least 80% of the measuring range.
- A load constant above 80% of the maximum has been found by RAW measurement or calculation and needs to be entered.

Request:

Instruction code: 12H

Response:

Confirmation code: ACK 00H

Parameters: (load)[(raw)]

load	Calibration load	Length: 2 bytes
The actual load or RAW value shown below is to match the value entered here. Thus: *If RAW is not entered:* indicate here which load value corresponds to the current strain gauge load. *If RAW is specified:* indicate here which load value corresponds to the specified value of the RAW parameter.		

[8] This means that the parameter does not need to be sent at all (it can be omitted).

raw	RAW load (optional parameter) [8]	Length: 2 bytes
The number of parts in the converter that corresponds to the specified strain gauge load. We recommend working with values that are greater than 80% of the range in this instruction.		

Examples:

Request - with load value only:
2AH,61H,00H,07H,31H,02H,12H,27H,10H,F1H,0DH
Request - with load value and RAW value:
2AH,61H,00H,09H,31H,02H,12H,27H,10H,4EH,20H,81H,0DH
Response (confirmation):
2AH,61H,00H,05H,31H,02H,00H,3CH,0DH

Supplementary

Reading the name and version

It reads the device name, the internal software version and a list of possible communication formats. Set at the factory.

Request:

Instruction code: F3H

Response:

Confirmation code: ACK 00H

Parameters: (string)

String	Name and version	length: 1 byte
A string in the format of this example: *TE485; v0672.01.011; f66 97*		

Examples:

Request:
2AH,61H,00H,05H,FEH,02H,F3H,7CH,0DH
Command to read name and version.
Response:
2AH,61H,00H,21H,31H,02H,00H,54H,45H,34H,38H,35H,3BH,76H,30H,36H,37H,32H,2EH,30H,31H,2EH,31H,31H,3BH,20H,69H,42H,69H,70H,6FH,6CH,61H,72H,3BH,7FH,0DH
Example response: TE485;v0672.01.11; iBipolar;

In 66 format:

Request: "?"

Response: (ACK "0")

Example: *Request*

**B1?↵*

Response - example of a converter response:

**B10 TE485; V0672.01.11; F66 97↵*

Reading production data

The instructions read the manufacturing data from the device.

Request:

Instruction code: FAH

Response:

Confirmation code: ACK 00H

Parameters: (product_number)(serial_number)(other)

product_number	Length: 2 bytes
Product number. In the case of equipment number 0227.00.03/0001, this is number 227.	

serial_number	Length: 2 bytes
Serial number of the product. For the device number 0227.00.03/0001, this is number 1.	

other	Length: 4 bytes
Further production information.	

Examples:

Request:
2AH,61H,00H,05H,FEH,02H,FAH,75H,0DH
Response:
2AH,61H,00H,0DH,35H,02H,00H,00H,C7H,00H,65H,20H,05H,09H,23H,B3H,0DH
The product number is 199 (= 00C7H) and the serial number is 101 (= 0065H).

Saving user data

The instruction saves the user data. The user data space is a memory in which the user can store any data that the device will remember even after power off or reset. This space is useful, for example, for naming the location of the device, etc.

Request:

Instruction code: E2H

Parameters: (position)(data)

Positions	length: 1 byte
The address of the memory location where the entered data will be stored. It is possible to enter a number from the range 00H to 0FH.	

data	Length: 1 to 16 bytes
Any user data. The memory has a capacity of 16 bytes if written from the first position. If a longer string than this is written, the device will return an error and no write will occur. (If writing to a memory address such as 0CH, a maximum of 4 bytes can be written.)	

Response:

Confirmation code: ACK 00H

Examples:

Request:
2AH,61H,00H,0FH,31H,02H,E2H,00H,53H,74H,6FH,72H,61H,67H,65H,20H,41H,1AH,0DH
Storage *A string* storage (53H,74H,6FH,72H,61H,67H,65H,20H,41H).
Response:
2AH,61H,00H,05H,31H,02H,00H,3CH,0DH
The string has been saved.

In 66 format:

Request: "DW"(position)(data) *(Data Write)*

Response: (ACK "0")

Legend: (position) Address of the memory position to be written to. From interval 0-9 or A-F.

(data) 1 to 16 bytes; Any user data. From interval 0-9 or A-F.

Example: *Request*

**B1DW0BOILERROOM 1↵*

Response

**B10↵*

Reading stored user data

The instruction reads the stored user data. The user data space is a memory in which the user can store any data that the device will remember even after power off or reset. This space is useful, for example, for naming a measurement location.

Request:

Instruction code: F2H

Response:

Confirmation code: ACK 00H

Parameters: (data)

data	Length: 16 bytes
User data.	

Examples:

Request:
2AH,61H,00H,05H,31H,02H,F2H,4AH,0DH
Response:
2AH,61H,00H,15H,31H,02H,00H,53H,74H,6FH,72H,61H,67H,65H,20H,41H,20H,20H,20H,20H,20H,20H,16H,0DH
The string "Storage A " is stored in the user data.

In 66 format:

Request: "DR" *(Data Read)*

Response: (ACK "0")(data)

Legend: (data) 1 to 16 bytes; User data.

Example: *Request*

**B1DR↵*

Response

**B10BOILERROOM 1↵*

Status settings

Sets the status of the device. A user-defined byte that can be used to determine the status of the instrument. This byte is freely user-writable. It serves as a memory location suitable, for example, for user indication of the device status. (It is reset after reset or power-on.)

Request:

Instruction code: E1H

Parameters: (status)

status	length: 1 byte
Device status. When the instrument is switched on, or after a reset (even a software reset), the status is automatically set to 00H. If the Status Setting instruction is changed to a different value, it can be easily identified later what state the instrument is in.	

Response:

Confirmation code: ACK 00H

Examples:

Request:
2AH,61H,00H,06H,01H,02H,E1H,12H,78H,0DH
12H status setting.
Response:
2AH,61H,00H,05H,01H,02H,00H,6CH,0DH
Confirmation.

In 66 format:

Request: "SW"(status) (Status Write)

Response: (ACK "0")

Legend: (status) character from the interval "space" to "~" (32 - 126)

Example: Request - character A

**B1SWA↵*

Response

**B10*

Status reading

It reads the status of the device. This is a user-defined byte that can be used to determine the status of the instrument.

Request:

Instruction code: F1H

Response:

Confirmation code: ACK 00H

Parameters: (status)

status	length: 1 byte
Device status. After switching on the device or after a reset (even software reset) the status is automatically set to 00H.	

Examples:

Request:
2AH,61H,00H,05H,01H,02H,F1H,7BH,0DH
Response:
2AH,61H,00H,06H,01H,02H,00H,12H,59H,0DH
The device status is set to 12H.

In 66 format:

Request: "SR" (Status Read)

Response: (ACK "0")(character)

Legend: (character) character from the interval "space" to "~" (32 - 126)

Example: Request

**B1SR↵*

Response

**B10A↵/*

Reading communication errors

The instruction returns the number of communication errors that have occurred since the device was turned on, or since the last communication error reading.

Request:

Instruction code: F4H

Response:

Confirmation code: ACK 00H

Parameters: (errors)

errors	length: 1 byte
The number of communication errors that have occurred since the device was powered on, or since the last reading. The following events are considered communication errors: • A prefix is expected and another byte will come. • SUM checksum does not match. • The report is incomplete.	

Examples:

Request:
2AH,61H,00H,05H,01H,02H,F4H,78H,0DH
Response:
2AH,61H,00H,06H,01H,02H,00H,05H,66H,0DH
There have been 5 communication errors since the power was turned on.

Enabling checksum

Allows you to cancel checksum validation. This instruction is handy for debugging applications. When manually entering instructions via the terminal, it is not necessary to enter the checksum (second to last byte) correctly.

It is not recommended to switch off the control in cases other than test operation of the device. The checksum is a protection against data corruption during transmission over the communication link. The check is factory enabled.

Request:

Instruction code: EEH

Parameters: (status)

Status	length: 1 byte
00H to disable checksum control. 01H to enable checksum control.	

Response:

Confirmation code: ACK 00H

Examples:

Request:
2AH,61H,00H,06H,01H,02H,EEH,01H,7CH,0DH
Turning on the checksum control.

Response:
2AH,61H,00H,05H,01H,02H,00H,6CH,0DH
Confirmation of the order.

Checksum - reading settings

Reads the current checksum control settings. (See the description of the previous instruction "Enabling checksum".)

Request:

Instruction code: FEH

Response:

Confirmation code: ACK 00H

Parameters: (status)

Status	length: 1 byte
00H checksum check disabled. 01H checksum control enabled.	

Examples:

Request:
2AH,61H,00H,05H,01H,02H,FEH,6EH,0DH
Response:
2AH,61H,00H,06H,01H,02H,00H,01H,6AH,0DH
Checksum control is on.

Reset

Resets the device. The module will return to the same state as when the power was turned on.

Request:

Instruction code: E3H

Response:

Confirmation code: ACK 00H

Examples:

Request:
2AH,61H,00H,05H,01H,02H,E3H,89H,0DH
Response:
2AH,61H,00H,05H,01H,02H,00H,6CH,0DH
The reset will be performed after this reply has been sent.

In 66 format:

Request: "RE" *(REset)*

Response: (ACK "0")

Example: *Request:* **B1RE↵*

Response: **B10↵*

Note: The reset is performed after the reply has been sent.

MODBUS RTU COMMUNICATION PROTOCOL

For the initial configuration of the address, etc., we recommend using, for example, the ModbusConfigurator program, which can be downloaded here:

https://papouch.com/modbus-configurator-prepinac-typu-protokolu-p4024/

List of instructions

Depending on the type of register, the device allows you to access its memory with the following instructions:

- 0x03 .....reading holding registers
- 0x04 .....reading input registers
- 0x06.....setting one holding register
- 0x10.....writing to several holding registers
- 0x11.....identification

Device identification

Reading the device identification string (Report slave ID).

Functional codes:

0x11 - Report slave ID

Parameters:

Number of bytes	1 Byte	by string
ID	1 Byte	The ID is the same as the device address
RI	1 Byte	Run Indicator - always 0xFF (on)
Data	N Byte	The string is the same as in the Spinel protocol. For example: *TE485; v0672.01.11; f66 97*

Holding Register

Device configuration, operation of pulse counters and analogue outputs.

Address	*Access*	*Functions*	*Name*
0 [9]	reading	0x06	**Enabling configuration** Writing the value 0x00FF to this memory location must precede all instructions that write to the holding register at addresses 0 to 15. It serves to protect against unwanted configuration changes. Not allowed to write Enable configuration using Multiply write at the same time as other parameters.
1	reading, writing	0x03, 0x06, 0x10	**Address (ID)** [10] Unique address of the device in the Modbus protocol. A number in the range 1 to 247 is expected. The address is unique to the Modbus protocol. *The default address is 0x0031.*

[9] It is also possible to encounter the numbering of registers starting from one, because this first register has address 0.

[10] Writing to this memory location must be preceded by writing the value 0x00FF to address 0 in the Configuration Enable position. This is to protect against unwanted configuration changes. It is not allowed to write the Configuration Enable using Multiply write at the same time as other parameters.

<table>
<tr><th>Address</th><th>Access</th><th>Functions</th><th>Name</th></tr>
<tr><td>2</td><td>reading,
writing</td><td>0x03, 0x06, 0x10</td><td>Communication speed [10]
Speeds and their corresponding codes:
1 200 Bd 0x0003
2 400 Bd 0x0004
4 800 Bd 0x0005
9 600 Bd ..... 0x0006 (default setting)
19 200 Bd 0x0007
38 400 Bd 0x0008
57 600 Bd 0x0009
115 200 Bd ..... 0x000A</td></tr>
<tr><td>3</td><td>reading,
writing</td><td>0x03, 0x06, 0x10</td><td>Data word [10]
The data word is always 8-bit.
<table>
<tr><th>Value</th><th>Parity</th><th>Number of stopbits</th></tr>
<tr><td>0x0000 (default)</td><td>none (N)</td><td>1</td></tr>
<tr><td>0x0001</td><td>even (E)</td><td>1</td></tr>
<tr><td>0x0002</td><td>odd (O)</td><td>1</td></tr>
<tr><td>0x0003</td><td>not (N)</td><td>2</td></tr>
<tr><td>0x0004</td><td>even (E)</td><td>2</td></tr>
<tr><td>0x0005</td><td>odd (O)</td><td>2</td></tr>
<tr><td>0x0006 to 0x00FF</td><td>none (N)</td><td>1</td></tr>
</table></td></tr>
<tr><td>4</td><td>reading,
writing</td><td>0x03, 0x06, 0x10</td><td>End of packet resolution [10]
Configures how much delay between bytes will be considered the end of the packet. The delay is specified in number of bytes. You can specify a value from 4 to 100. The default value is 10.</td></tr>
<tr><td>5</td><td>reading,
writing</td><td>0x03, 0x06, 0x10</td><td>Communication protocol [10]
Allows you to switch the device to Spinel protocol communication. After sending a response, the device switches to the selected protocol and continues to communicate only with it. (There are instructions in each protocol for switching protocols.)
Code for Spinel protocol: 0x0001 (default)
Code for Modbus RTU protocol: 0x0002</td></tr>
<tr><td>7 - 9</td><td>writing</td><td>0x06, 0x10</td><td>Setting the address by serial number [10]
Addr. 7 - new address
Addr. 8 - product number
Addr. 9 - serial number
The product number and serial number of the device is indicated on the label on the device as 0516/0001, where 0516 is the manufacture number and 0001 is the serial number.</td></tr>
<tr><td>16</td><td>reading,
writing</td><td>0x03, 0x06, 0x10</td><td>Tensiometer sensitivity [10]
The sensitivity code for which the calibration is performed. The available options are in Tab. 1 on page 17.</td></tr>
</table>

Address	*Access*	*Functions*	*Name*
17	reading	0x03	**Calibrated sensitivity** [10] This indicates for which sensitivity the strain gauge was calibrated. The available values are the same as in the previous register.
18	reading, writing	0x03, 0x06, 0x10	**Zero Calibration - RAW** [10] RAW value that corresponds to the zero position of the strain gauge. The default value is: 8000H Tip: This register can also be filled in automatically according to the information in the register 21.
19	reading, writing	0x03, 0x06, 0x10	**Calibration of the upper measurement limit - RAW** [10] RAW value that corresponds to the strain gauge load to the level specified in the following register (No.20). The default value is: FFFFH Tip: This register can also be filled in automatically according to the information in the register 21.
20	reading, writing	0x03, 0x06, 0x10	**Calibration of the upper measurement limit - load** [10] The load corresponding to the RAW value entered in the previous register or that will be valid when the value 0100H is written to the next register. We recommend working here with a load that is higher than 80% of the range.
21	writing	0x06, 0x10	**Semi-automatic calibration** [10] Zero calibration: Writing a value of 0000H represents the determination of when the strain gauge is in the zero position. (The currently detected RAW value is automatically written to register 18.) Calibration of the upper limit of the range: Writing the value 0100H represents the determination of when the strain gauge is loaded with the load specified in the previous register.
22	writing	0x06, 0x10	**Measurement speed** • 00H for default speed 6.25 SPS • 01H for speed 50 SPS

Input Register

Reading the measured value.

Address	*Access*	*Functions*	*Name*
0 [11]	reading	0x04	**Measurement status** Status of measured values, accessible in the following registers. The low byte (LSB) of the 2-byte register value is bit oriented and contains the following information: bit 3.2: 00 = the measured value is within the measuring range bit 3.2: 01 = measured value is less than the lower limit of the measuring range (underflow) bit 3.2: 10 = exceeding the upper limit of the measuring range (overflow) bit 7 (MSb): 0 = the measured value is invalid bit 7 (MSb): 1 = the measured value is valid
1	reading	0x04	**Converted value** Last measured value, recalculated according to the calibration specified in the Holding register. As long as Zero Calibration and Upper Measurement Limit Calibration are not set, the value available here is the same as the value in the following RAW value register!
2	reading	0x04	**RAW value** The last measured value is shown here as a number without recalculation.

[11] It is also possible to encounter the numbering of registers starting from one, because this first register has address 0.

SWITCH PROTOCOLS

The default protocol is Spinel. The following instructions from the Spinel protocol are used to switch to the MODBUS protocol. The device can be conveniently switched to the Modbus protocol (or back) using the Modbus configurator (there are also examples for protocol switching in Python on the page).

Spinel → MODBUS RTU

Enabling configuration

Enables the execution of a service instruction. It must immediately precede the Switch instruction. The instruction cannot be used with a universal address or a "broadcast" address.

Minutes

Structure:	→ **0xE4**
Example:	→ 2A 61 00 05 31 02 **E4** 58 0D ← 2A 61 00 05 31 02 **00** 3C 0D

Switch

Protocol switching is done by a special Spinel protocol instruction, format 97. The address must be the address of the specific module (neither the "broadcast" nor the universal address can be used). The instruction must be immediately preceded by the "Enable Configuration" instruction.

Parameters

protocol	1 byte	Protocol identification number: • 0x01: Spinel • 0x02: Modbus RTU

Minutes

Structure:	→ **0x2A**, protocol
Example:	→ 2A 61 00 06 31 02 **ED 02** 4C 0D • 0x02: Switching to Modbus RTU protocol. ← 2A 61 00 05 31 02 **00** 3C 0D

MODBUS RTU→ Spinel

It is switched by writing to the Holding register.

Papouch s.r.o.

Data transmission in industry, line and protocol conversions, RS232/485/422/USB/Ethernet/GPRS/ WiFi, measurement modules, intelligent temperature sensors, I/O modules, and custom-made electronic applications.

Address:

Strašnická 3164/1a
102 00 Prague 10

Phone:

+420 267 314 268

Web:

papouch.com

Mail:

papouch@papouch.com