WIE485 and WIE232

Wiegand to RS485 or RS232 bidirectional interface converters for systems with contactless readers

WIE485 and WIE232

Datasheet

Created: 5.10.2010

Last revision: 27.03.2020 10:04

Number of pages: 28

Papouch s.r.o.

Address:

Strašnická 3164/1a
102 00 Praha 10

Phone:

+420 267 314 268

Website:

www.papouch.com

E-mail:

papouch@papouch.com

TABLE OF CONTENTS

DESCRIPTION

Wiegand is a standard communication protocol used in contactless card readers. The WIE232 and WIE485 converters differ only in the type of the serial port to access data from Wiegand or to send data to Wiegand. The converters can work with **Wiegand protocols of any type, ranging from 1 to 64 bits**.

The converters provide two modes of operation:

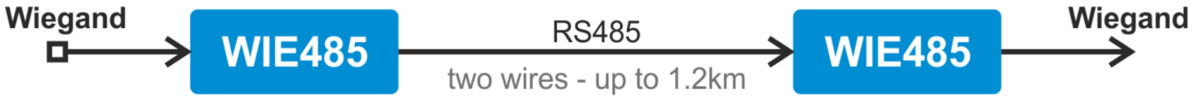
1) **Wiegand to serial line conversion**. This mode is typically used to connect standard contactless readers to the computer. The output of the converter is the number of the scanned card or directly a set of bits received by the Wiegand protocol. This data can be accessed in two ways:
   a) Method 1: The card number is sent in the form of raw data to the serial line. This method is described on page 6.
   b) Method 2: Card number or a set of bits received by the Wiegand protocol, obtained through the serial link by Spinel or MODBUS RTU protocols. More information about this method is provided on page 7.
2) **Generating the Wiegand protocol**, which is useful for simulating a contactless reader.

The direction of communication and some other parameters are selected by switches on the upper side.

The RS485 line of WIE485 is galvanically isolated from other parts of the device.

Application

- Contactless card readers
- Electronic access systems
- Security systems
- Modernization of existing access systems
- Contactless card reader simulator (serial line to Wiegand conversion)
- WIE232 and WIE485 **can be used to extend the Wiegand communication**. In case of WIE485 the length can be up to 1.2 km (0.7 miles).



- WIE232 only: RS232 convertor for Jablotron RFID reader JA-80H with integrated keyboard

Fig. 1 - Jablotron JA-80H

CONNECTION

Any change to the configuration parameters will apply after a power cycle.

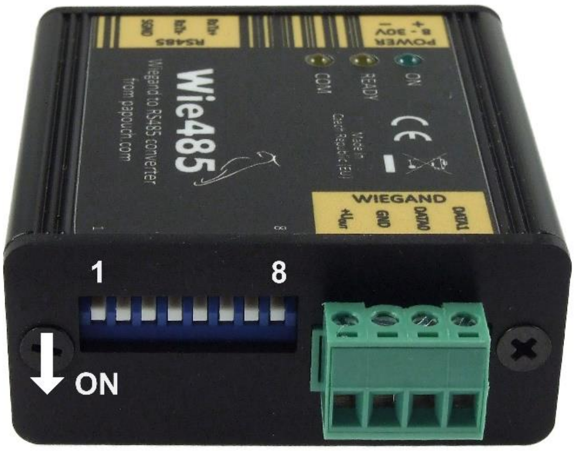


Fig. 2 - configuration switch position and ON position

1) Use switch **SW 7** on the side to set the operating mode of the converter:
   ON: The device operates as RSxxx → Wiegand converter (Wiegand "simulator").
   OFF: The device operates as Wiegand → RSxxx converter (card reader converter).

2) Set the type of Wiegand:[1]

   a. Turn off power supply.
   b. Set switch **SW 8** to 'ON'.
   c. Turn on power supply.
   d. The device now communicates (regardless of the current settings) at the rate of 9600 Bd, 8 data bits, no parity, 1 stop-bit and device ID is 1.
   e. Enter the changes. (Even after making changes, the device communicates using the aforementioned parameters.)[2]
   f. Turn off power supply.
   g. Set switch **SW 8** to 'OFF'.
   h. Turn on power supply. (The device now communicates using the new parameters.)

[1] If you use the device in the Wiegand → RSxxx, mode, you can set the desired Wiegand type using the switch on the upper side. This feature has been retained to ensure compatibility with previous versions of the firmware of the converter. The purpose of the switches is as follows:

SW1	SW2	SW3	SW4	SW5	Protocol Type
OFF	OFF	OFF	OFF	OFF	Wiegand 30
ON	OFF	OFF	OFF	OFF	Wiegand 26
OFF	ON	OFF	OFF	OFF	Wiegand 40
ON	ON	OFF	OFF	OFF	Wiegand 32
ON	OFF	ON	OFF	OFF	Wiegand 34
OFF	OFF	ON	OFF	OFF	Wiegand 42
ON	OFF	OFF	OFF	ON	Wiegand 26b (keyboard with JA-80H reader)

[2] Configuration can be done using e.g. Modbus Configurator, which can be downloaded from papouch.com. It is necessary to bear in mind that this SW does not expect Wie485 to communicate at the same speed in the configuration mode. Thus, it is first needed to set the speed to 9600 Bd and ID 1, and then enter new parameters.

3) WIE485: Connect the device to the host system using RxTx+ (A) and RxTx- (B) wires. Connect RxTx+ to the opposite RxTx+, and similarly RxTx- to RxTx-. GND wire can be used if the cable is shielded. In this case, remember to connect shielding on one side of the cable only!

   WIE232: Using a standard extension cable, connect the RS232 connector on WIE232 to a PC (or another device with a standard RS232 serial line).

4) Chose communication mode using **SW 6** (see pages 6 and 7).

   To extend Wiegand: Converter that receives Wiegand set using the following commands:
   a. Setting the automatic transmission instruction (pg. 17) set to value 03H.
   b. Automatically generated message instruction (pg. 18) set to value 01H.

5) Connect the device (typically a contactless reader) to the Wiegand connector.

6) Connect power supply to the green terminal block. Polarity is indicated on the label. (The device features integrated protection against damage caused by reversed polarity of the supply.)

METHOD 1: SENDING THE CARD NUMBER

*This method of communication is activated when the configuration switch **SW 6** is ON.*

Below is described how much data the converter sends after applying the card. The following abbreviations are used in the description:

CR ....carriage return (13 in decimal)

LF ....line feed (10 in decimal)

1B ....one byte representing number 0 to 255. It is sent as ASCII characters. Thus it can be sent as one character (0,1,2,…) or up to three characters (…,253,254,255).

2B ....two bytes representing number 0 to 65535. It is sent as ASCII characters. Thus it can be sent as one character (0,1,2,…) or up to five characters (…,65534,65535).

5Z ....five-character string. It is sent as five characters, completed with zeros from left.

7Z ....seven-character string. It is sent as seven characters, completed with zeros from left.

5b ....five bytes with card number.

HESCII(*4B*) ...four bytes converted to ASCII characters (each 4 bits are converted to character 0 to F). Highest bit is the first received bit.

Wiegand 26

When a card is applied, the converter sends the following data as an ASCII string:

[facility*(1B)*][card number*(2B)*][CR][LF]

Wiegand 30

When a card is applied, the converter sends the following data as an ASCII string:

[card number][CR][LF]

Wiegand 32

When a card is applied, the converter sends the following data as an ASCII string:

[#][ ][first number*(5Z)*][second number*(7Z)*][CR][LF]

Wiegand 34

When a card is applied, the converter sends the following data as an ASCII string:

[HESCII(*4B*)][CR][LF]

Wiegand 40

When a card is applied, the converter sends the following data as an ASCII string:

[card number*(2B)*][CR][LF]

Wiegand 42

When a card is applied, the converter sends the following binary data:

[card number*(5b)*]

Wiegand 26b from JA-80H reader

When a card is applied, the converter sends the following data as an ASCII string:

[facility*(1B)*][card number*(2B)*][CR][LF]

After pressing a button on the reader, the converter sends the following data as an ASCII string:

[character-s of button or buttons][CR][LF]

Jablotron JA-80H reader is to be connected according to following table:

JA80H **wire colour**	**Converter** **terminal**
red	+U_{OUT}
blue	GND
green	DATA0
brown	DATA1

tab. 1 – interconnection of the reader and converter

METHOD 2: COMMUNICATION VIA SPINEL OR MODBUS RTU PROTOCOLS

*This method of communication is activated when the configuration switch **SW 6** is OFF.*

Quick use guide

- Regarding RS232 (WIE232) and RS485 (WIE485), the WIE232 and WIE485 converters communicate using Spinel or Modbus unless configuration switch 6 is ON.
- To read **the protocol type** that is currently selected with the switch on the side of the converter, use the 'Reading the selected type' instruction in the Spinel protocol (see page 16).

- To read **the last card**, use the 'Reading the last received data' instruction in the Spinel protocol (see page 13).
- To **send a packed to Wiegand**, use the 'Sending a Wiegand packet' instruction in the Spinel protocol (see page 17).

MODBUS RTU

Wiegand → RSxxx

The code received from the contactless reader is stored in registers 13 to 16 (readable via the Input register). The first bit (MSb) in register 13 is the first bit (MSb) of the received Wiegand packet. Information about the number of bits of the received code is stored in register 12. Thus it is possible to find out how many bits in registers 13 to 16 are reserved for the received code. (Unused bits in these registers are ignored.)[3]

RSxxx → Wiegand

The MODBUS RTU protocol is used to load registers 12 to 16 simultaneously (by function 0x10 into the Holding Register). Register 12 shows the number of bits of Wiegand, while registers 13 to 16 contain the Wiegand bits themselves. After being loaded, Wiegand is transmitted. (No code may be loaded during transmission. End of loading may be viewed in register 11.)

Data are generated in Wiegand as follows: The pulse rate is 50 μs (±3 μs) long. There is always a delay of 2 ms between the beginnings of individual pulses.

Holding Register

Warning: It is only possible to read addresses 1 to 4 or 11 to 16 at a time. (It is not possible to read both ranges simultaneously.)

Address	*Access*	*Function*	*Description*
1	read write[4]	0x03 0x10	Positions for free use
2	read write	0x03 0x10	Device ID (number ranging from 1 to 247)

[3] For compatibility with the previous versions of the firmware, addresses 2 to 4 in the Input Register are set depending on the settings of Switches SW1 to SW3 on side of the converter – these addresses do not contain the whole Wiegand packet, but only the number of the card. (For detailed description see the next page.)

[4] Writing is only allowed in the configuration mode, i.e. if configuration switch 8 is ON while the device is being switched on.

Address	*Access*	*Function*	*Description*
3	read write	0x03 0x10	Communication speed. A code from the following list: 0 ....110 Bd 1 ....300 Bd 2 ....600 Bd 3 ....1 200 Bd 4 ....2 400 Bd 5 ....4 800 Bd 6 ....9 600 Bd (default value) 7 ....19 200 Bd 8 ....38 400 Bd 9 ....57 600 Bd 10 .....115 200 Bd
4	read write[4]	0x03 0x10	Serial line mode. A code from the following list: 0 .... 8 data bits, no parity, 1 stop bit 1 .... 8 data bits, even parity, 1 stop bit 2 .... 8 data bits, odd parity, 1 stop bit 3 .... 8 data bits, no parity, 2 stop bits 4 .... 8 data bits, even parity, 2 stop bits 5 .... 8 data bits, odd parity, 2 stop bits
5	read	0x03	Reserved
6	read write[4]	0x03 0x10	Communication protocol [4] Makes it possible to switch over to the Spinel protocol. After sending the response, the device switches to the selected protocol. (Each protocol contains an instruction for switching over to the other one.) Spinel protocol code: 0x0001 (default) Modbus RTU protocol code: 0x0002
11	read	0x03	Transmission mode. A code from the following list: 0 – transmission disabled 1 – data have been transmitted, the converter is ready to transmit new data
12	read write	0x03 0x10	Number of bits to be sent from addresses 13 to 16 (number ranging from 1 to 64).
13 - 16	read write	0x03 0x10	Received Wiegand bits or bits to be send. If the previous data have not been sent yet, function code 04 is returned (it was impossible to save the data).

Input Register

Warning: It is only possible to read addresses 1 to 4 or 11 to 16 at a time. (It is not possible to read both ranges simultaneously.)

The data in addresses 2 to 4 represent directly the number of the applied card. In contrast, addresses 13 to 16 contain all the bits of the received Wiegand packet.

Address	*Access*	*Function*	*Description*				
			for Wie 30	*for Wie 26*	*for Wie 40*	*for Wie 32/34*	*for Wie 42*
1	read	0x04	Code status 1 = no code read yet; 0 = registers contain a valid code If this position shows 0 and any of the following registers is read, the value is set to 1 here.				
2	read	0x04	N/A	N/A	N/A	N/A	8 bit LSB: the first byte of card number
3	read	0x04	16 bit The upper two bytes of the card number	8 bit LSB: the first byte of the card number	N/A	16 bit The upper two bytes of the card number	16 bit The 2nd and 3rd byte of the card number
4	read	0x04	16 bit The lowest two bytes of the card number	16 bit The lowest two bytes of the card number	16 bit The lowest two bytes of the card number	16 bit The lowest two bytes of the card number	16 bit The 4th and 5th byte of the card number
11	read	0x04	Code status 1 = no code read yet; 0 = registers contain a valid code If this position shows 0 and any of the following registers is read, the value is set to 1 here.				
12	read	0x04	Wiegand type according to the number of bits received.				
13 14 15 16	read	0x04	Memory space to store the complete received Wiegand packet. The highest bit in Address 13 represents the first incoming bit of the Wiegand code.				

SPINEL COMMUNICATION PROTOCOL

The devise comes with a standardized protocol called Spinel[5] of formal 97 (binary). Format 97 uses 8-bit binary characters to communicate (integers in the range from 0 to 255 in decimal). Spinel Terminal is a special program designed to enable easy communication debugging. The instructions are divided into Requests and Responses:

Structure

Request:

```
 PRE FRM NUM NUM ADR SIG INST DATA… SUMA CR
```

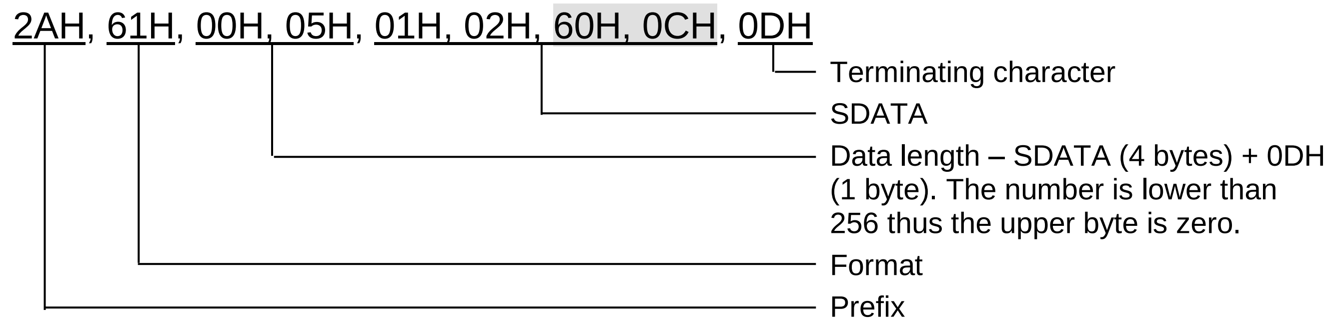
Response:

```
 PRE FRM NUM NUM ADR SIG ACK  DATA… SUMA CR
```

PRE	Prefix, 2AH (“*“ character).
FRM	Number of Format 97 (61H).
NUM	Number of instruction bytes from the following byte to the end of the frame.
ADR	Address of the module to which the Request is being sent or which is responding to it.
SIG	Message description – any number form 00H to FFH. The number that was sent in the Request is returned in the Response, which makes it easy to see which Request the Response belongs to.
INST[6]	Instruction code.
ACK	Acknowledgement of whether and how the Request was performed. ACK can be found within the range of 00H to 0FH.
DATA[6]	Data. Described in great detail in chapter Preview of Instructions for each instruction.
SUMA	Checksum.
CR	Terminating character (0DH).

Explanatory notes

Example


2AH, 61H, 00H, 05H, 01H, 02H, 60H, 0CH, 0DH
Terminating character
SDATA
Data length – SDATA (4 bytes) + 0DH (1 byte). The number is lower than 256 thus the upper byte is zero.
Format
Prefix

Data Length (NUM)

Sixteen-bit value indicating the number of bytes to the end of the instruction; number of all bytes after NUM up to CR (including). It takes values from 5 to 65535. If lower than 5, the instruction is considered faulty and it is answered (if intended for the respective device) with the ACK “Invalid Data” instruction.

Compilation of NUM:

Add up the number of bytes following both NUM bytes (i.e. the number of SDATA bytes + 1 CR byte). View the resulting sum as a sixteen-bit number. Divide it into the upper and lower byte. The first NUM byte is the upper byte of the amount, the second NUM byte is the lower byte of the amount. (If the amount of bytes is lower than 256, the first NUM byte is 00H.)

[5] Detailed information about the Spinel protocol can be found on the website papouch.com .
[6] For clarity, instructions and data in the examples on the following pages are highlighted like this.

Address (ADR)

The FFH address is reserved for broadcast. If the Request contains the FFH address, the device operates as if its own address has been entered. No Response is sent to Requests with this address.

The FEH address is a universal address. If the Request contains the FEH address, the device operates as if its own address has been entered. The device enters the real, currently set address into the Response. The universal address is used in cases where only one device is connected.

Request Acknowledgement (ACK)

ACK informs the superior device about how the received instruction has been processed. Acknowledgement codes:

00H ....EVERYTHING OK
The instruction has been properly received and completely executed.

01H ....UNSPECIFIED ERROR
Unspecified device error.

02H ....INVALID INSTRUCTION CODE
The received instruction code is unknown.

03H ....INVALID DATA
Data are of invalid length or contain an invalid value.

04H ....ENTRY NOT ALLOWED/ACCESS REFUSED
- The Request was not performed as some conditions had not been fulfilled.
- Attempt to enter data into inaccessible memory.
- Attempt to activate a function requiring a different configuration (e.g. higher communication speed).
- Attempt to change configuration without previous permission.
- Access to memory protected by a password.

05H ....DEVICE FAILURE
- Device failure requiring servicing.
- Device internal memory error or setup memory error.
- Device internal error (operation error or start-up error).
- Any other error affecting proper functioning of the device.

06H ....NO DATA AVAILABLE

0DH.......INSTRUCTION SENT AUTOMATICALLY – CHANGE IN THE STATUS OF THE DIGITAL INPUT

0EH.......INSTRUCTION SENT AUTOMATICALLY – CONTINUOUS MEASURING
- Periodical sending of measured values.

0FH ....INSTRUCTION SENT AUTOMATICALLY – EXCEEDED LIMITS OR RANGE

Checksum (SUMA)

The sum of all instruction bytes (all transmitted data except CR are included) subtracted from 255. Calculation: SUM = 255 – (PRE + FRM + NUM + ADR + SIG + ACK (INST) + DATA)

Incorrect checksums are not answered. (Device is waiting for CR even if the checksum is incorrect.)

Overview of all instructions

Reading the last received data

The last data received by Wiegand. The Response contains variable data according to the set protocol type.

Request:

Instruction code: A0H

Response:

Acknowledgement code: ACK 00H

Parameters: (status) (type) (data)

status	Status of the received data	length: 1 byte
0 = the data are valid 1 = the data are invalid or have already been read		

type	Type of Wiegand	length: 1 byte
00H … type 30 01H … type 26 02H … type 40 03H … type 32 05H … type 34 04H … type 42 80H … keyboard		

data	Received data	length: depends on the type
Type 30: 4 bytes: 32-bit number Type 26: 3 bytes: 1st indicates the FC code; 2nd and 3rd 16-bit code of the card Type 40: 2 bytes: 16-bit code of the card Type 32: 4 bytes: 1st and 2nd the first 16-bit number; 3rd and 4th the second 16-bit number Type 42: 5 bytes: Five bytes from the applied card		

Example:

Request:
`2AH,61H,00H,05H,FEH,02H,A0H,CFH,0DH`
Response:
`2AH,61H,00H,0AH,31H,02H,00H,00H,01H,F8H,39H,3DH,C8H,0DH`
Valid data, Wiegand 26, FC code: F8H, card code: 393DH.

Reading RAW data

The last data received by Wiegand. The response contains “raw data” as they were received according to the selected type of Wiegand.

Request:

Instruction code: A1H

Response:

Acknowledgement code*:* ACK 00H

Parameters: (status) (type) (data)

status	Status of the received data	length: 1 byte
0 = the data are valid 1 = the data are invalid or have already been read		

type	Type of Wiegand	length: 1 byte
The number of the received bits – maximum is 64.		

data	Received data	length: 8 byte
“Raw” data from Wiegand in the same order as they were received.		

Example:

Request:
`2AH,61H,00H,05H,31H,02H,A1H,9BH,0DH`
Response:
`2AH,61H,00H,0FH,31H,02H,00H,00H,1AH,FCH,1CH,9EH,80H,00H,00H,00H,00H,E2H,0DH`
Valid data, Wiegand 26, the first 26 bits represent the data, the rest of the bits is invalid.

Reading – automatically

The last data received by Wiegand. The response contains “raw data” as they were received regardless of the selected type of Wiegand.

Request:

Instruction code: A2H

Response:

Acknowledgement code: ACK 00H

Parameters: (status) (type) (data)

status	Status of the received data	length: 1 byte
0 = the data are valid 1 = the data are invalid or have already been read		

type	Type of Wiegand	length: 1 byte
The number of the received bits – maximum is 64.		

data	Received data	length: 8 byte
“Raw” data from Wiegand in the same order as they were received.		

Example:

Request:
`2AH,61H,00H,05H,FEH,02H,A2H,CDH,0DH`
Response:
`2AH,61H,00H,0FH,31H,02H,00H,00H,1AH,FCH,1CH,9EH,80H,00H,00H,00H,00H,E2H,0DH`
Valid data, Wiegand 26, the first 26 bits represent the data, the rest of the bits is invalid.

Reading the selected type

Reads the currently selected type of the Wiegand protocol.

Request:

Instruction code: A3H

Response:

Acknowledgement code: ACK 00H

Parameters: (type)

type	Type of Wiegand	length: 1 byte
00H … type 30 01H … type 26 02H … type 40 03H … type 32 05H … type 34 04H … type 42 80H … type 30 + keyboard 81H … type 26 + keyboard 82H … type 40 + keyboard 83H … type 32 + keyboard 85H … type 34 + keyboard 84H … type 42 + keyboard		

Example:

Request:
`2AH,61H,00H,05H,FEH,02H,A3H,CCH,0DH`
Response:
`2AH,61H,00H,06H,31H,02H,00H,01H,3AH,0DH`
Wiegand 26 without a keyboard.

Reading the settings of automatic transmission

Sends the status of automatic data transmission after receiving a packet from the reader.

Request:

Instruction code: A4H

Response:

Code: ACK 00H

Parameters: (set)

set	length: 1 byte
00H … automatic transmission is disabled 01H … automatic transmission of the card number according to the selected type of Wiegand 02H … automatic transmission of the received raw data according to the selected type of Wiegand 03H … automatic transmission of the received raw data (only the received bits of individual bytes regardless of the selected type of protocol)	

Example:

Request:
`2AH,61H,00H,05H,31H,02H,A4H,98H,0DH`
Response:
`2AH,61H,00H,06H,31H,02H,00H,01H,3AH,0DH`
Automatic transmission of the card number according to the selected type of Wiegand.

Setting the automatic transmission

Enables automatic transmission of data after receiving a packet from the reader.

Request:

Instruction code: B4H

Parameters: (set)

set	length: 1 byte
00H … automatic transmission is disabled 01H … automatic transmission of the card number according to the selected type of Wiegand 02H … automatic transmission of the received raw data according to the selected type of Wiegand 03H … automatic transmission of the received raw data (only the received bits of individual bytes regardless of the selected type of protocol)	

Response:

Acknowledgement code: ACK 00H

Example:

Request:
`2AH,61H,00H,06H,31H,02H,B4H,01H,86H,0DH`
Automatic transmission of the card number according to the selected type of Wiegand.
Response:
`2AH,61H,00H,05H,31H,02H,00H,3CH,0DH`

Sending a Wiegand packet

This instruction sends data from the converter to Wiegand. The type of the data depends on the selected type of protocol.

If the device has not been switched to the mode of generating Wiegand with switches on the side, or if the previous message has not yet been generated, the response is ACK 04H.

Request:

Instruction code: B5H

Parameters: (bits) (data)

bits	Number of bits	length: 1 byte
The number of Wiegand bits to be generated.		

data	Wiegand packet	length: 8 byte
Data for the Wiegand packet. To be filled from left to right (from MSb to LSb).		

Response:

Acknowledgement code: ACK 00H

Example:

Request:
`2AH,61H,00H,0EH,FEH,02H,B5H,40H,00H,FFH,00H,FFH,00H,FFH,00H,FFH,75H,0DH`
Generating of Wiegand 64.
Response:
`2AH,61H,00H,05H,31H,02H,00H,3CH,0DH`

Automatically generated message

After receiving a Wiegand packet, this automatic message is generated according to the settings.

Automatic message:

Acknowledgement code: 0CH

Parameters: (type) (wie) (bits) (data)

type	Message type	length: 1 byte
Automatic messages can have the following value: 01H … Decoded card number (Wiegand z) 02H … The raw data as they were received (Wiegand according to the settings on the switch) 03H … The raw data as they were received (Independently of the set pins)		

wie	Type of Wiegand	length: 1 bytes
This byte is sent only if the message type is 01H. 00H … type 30 01H … type 26 02H … type 40 03H … type 32 05H … type 34 04H … type 42 80H … type 30 + keyboard 81H … type 26 + keyboard 82H … type 40 + keyboard 83H … type 32 + keyboard 85H … type 34 + kayboard 84H … type 42 + keyboard		

bits	Number of bits	length: 1 byte
This byte is sent only if the message type is 02H or 03H. The number of the received bits – maximum is 64.		

data	Received data	length: 8 bytes
“Raw” data from Wiegand in the same order as they were received. To be filled from left to right (from MSb to LSb).		

Example:

Example 1:
`2AH,61H,00H,0AH,31H,00H,0CH,01H,01H,F8H,39H,3DH,BDH,0DH`
Message type 01H, Wiegand 26, FC code F8H, card code 393DH.
Příklad 2:
`2AH,61H,00H,0FH,31H,01H,0CH,02H,1AH,FCH,1CH,9EH,80H,00H,00H,00H,00H,D5H,0DH`
Message type 02H, Wiegand 26, followed by valid bits from left to right.

Setting the address for automatic messages

When sending an automated message, the sender's address is, by default, set to the address of the Wie converter. This instruction can be used to change this address to FFH, which is a broadcast address. (1) This settings (2) together with enabled automatic sending, and in the case of WieETH also (3) setting of proper network parameters, makes it possible to interconnect two Wie converters. One of them operates as a receiver and the second one as a transmitter of Wiegandu. Thus, the Wiegand protocol can be transmitted over greater distances.

Request:

Instruction code: B7H

Parameters: (mode)

mode	Number of bits	length: 1 byte
00H … the current address of the converter will be sent (standard) 01H … the FFH address will be sent (cooperation of two Wiegand converters)		

Response:

Acknowledgement code: ACK 00H

Example:

Request:
`2AH,61H,00H,06H,31H,02H,B7H,01H,83H,0DH`
The FFH address will be sent.
Response:
`2AH,61H,00H,05H,31H,02H,00H,3CH,0DH`
An example of an automated message with the FFH address:
`2AH,61H,00H,0FH,FFH,00H,0CH,03H,1AH,FCH,1CH,9EH,80H,00H,00H,00H,00H,07H,0DH`

Reading the settings of the address for automatic messages

Reading of the settings made by the previous instruction.

Request:

Instruction code: A7H

Response:

Acknowledgement code: ACK 00H

Parameters: (mode)

Example:

Request:
2AH,61H,00H,05H,FEH,02H,A7H,C8H,0DH
Response:
2AH,61H,00H,06H,31H,02H,00H,01H,3AH,0DH
The FFH address will be sent.

Permission of configuration

This instruction enables configuration to be carried out. It must immediately precede some instructions for the setting of communication parameters. After a subsequent instruction (even invalid), configuration is automatically disabled.

It is not possible to use the universal address with this instruction. The address of a particular device must always be given.

Request:

Instruction code: E4H

Response:

Acknowledgement code: ACK 00H

Examples:

Request:
2AH,61H,00H,05H,01H,02H,E4H,88H,0DH
Permission of configuration.
Response:
2AH,61H,00H,05H,01H,02H,00H,6CH,0DH
The receipt of the instruction has been acknowledged.

Setting the communication parameters

This command is used to set the address in the Spinel protocol and communication speed.

It is not possible to use the universal address with this instruction. If the address is not known and no additional device is connected to the line, the address can be detected using the “Reading the communication parameters” instruction. (Use the universal FEH address as the address of the device.) If this is not possible (there are other communication devises connected to the same communication line), you can assign an address to the device using the “Setting the address with serial number” instruction (see page 23).

The setting of communication parameters must be preceded by the “Permission of configuration” instruction (see page 20).

Request:

Instruction code: E0H

Parameters: (address) (speed)

address	New address of the device	length: 1 byte
New address of the device in the Spinel protocol. The address can be in the range of 00H to FDH. If protocol 66 is used for communication, it is necessary to use only the addresses that can be expressed as a displayable ASCII character. Default address: 31H		

speed	New communication speed		length: 1 byte
This parameter is used to set new communication speed of the device. The communication speed of WieETH is fixed and is set to 115 200 Bd. The default communication speed of Wie232 and Wie485 is 9 600 Bd. The codes of communication speeds can be found in the table on the right:	Speed [Bd]	Code for format 97	Code for format 66
	110	00H	0
	300	01H	1
	600	02H	2
	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
	230 400	0BH	B

Response:

Acknowledgement code: ACK 00H

The new address and communication speed will be set after the response has been sent.

Examples:

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

Reading the communication parameters

This command reads the address and communication speed of the device. This instruction is used to identify the set address when it is unknown. The request is sent to the universal FEH address. If the communication speed is not known as well, it is necessary to test all communication speeds of the device. When detecting the address of the device using the universal address, no other device can be connected to the line.

Request:

Instruction code: F0H

Response:

Acknowledgement code: ACK 00H

Parameters: (address) (speed)

address	Address of the device	length: 1 byte
The address of the device in the Spinel protocol.		

speed	Communication speed		length: 1 byte
Code of the communication speed. The communication speed of WieETH is fixed and is set to 115 200 Bd. The codes of communication speeds can be found in the table on the right:	Speed [Bd]	Code for format 97	Code for format 66
	110	00H	0
	300	01H	1
	600	02H	2
	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
	230 400	0BH	B

Examples:

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

Setting the address with serial number

This instruction makes it possible to set the address according to the unique serial number of the device. This instruction is useful in the event that the superior system or the operator loses the address of a device that is connected to the same communication line as other devices.

The serial number is indicated on the device as follows: *[product-number].[hardware-version].[software-version]/[serial-number]*, for example: *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 address of the device in the Spinel protocol.		

product_number	Product number	length: 2 bytes
Product number shown on the label on the device. For a device with number 0227.00.03/0001, the required number is 227.		

serial_number	Serial number of product	length: 2 bytes
Serial number shown on the label on the device. For a device with number 0227.00.03/0001, the required number is 1.		

Response:

Acknowledgement code: ACK 00H

Example:

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 using the new address.

Switching to another communication protocol

This command changes the type of the communication protocol. *It is possible to use, for example, Modbus Configurator, which can be downloaded from papouch.com.*

The instruction to change the protocol must be preceded by the “Permission of configuration” instruction (see page 20).

Request:

Instruction code: EDH

Parameters: (id)

id	Protocol ID	length: 1 byte
Protocol ID number: 01H – Spinel 02H – MODBUS RTU		

Response:

Acknowledgement code: ACK 00H

Examples:

Request:
`2AH,61H,00H,06H,31H,02H,EDH,FFH,4FH,0DH`
Response:
`2AH,61H,00H,05H,31H,02H,00H,3CH,0DH`
The receipt of the command has been confirmed.

INDICATION

The device contains three LEDs:

ON:

Indication of connected supply voltage.

READY:

If **SW 6** in ON: Flashing indicates the activity of the device.

If **SW 6** in OFF: Flashes when instructions from Spinel or Modbus protocol are being received.

Third LED:

Flashes when a card is applied.

TECHNICAL PARAMETERS

Communication parameters

Default communication protocol ....Spinel[7]

Communication speed.......................................adjustable from 300 Bd to 230 400 Bd

Default communication speed ....9600 Bd

Communication speed in configuration mode[8] ...always 9600 Bd

Number of data bits 8

Parity ....none

Number of stop-bits 1

RS485 bus (only WIE485):

Connector...removable terminal block with screws

Galvanic isolation ....yes, inductive

Resistors defining line status..............................22 kΩ[9]

Terminating resistor...120 Ω[10]

Serial port port RS232 (only WIE232)**:**

Connector...CAN 9 F

Pin	Name	Direction	Description
2	RXD	←	Receive Data
3	TXD	→	Transmit Data
4	DTR	→	Data Terminal Ready
5	GND	—	System Ground
6	DSR	←	Data Set Ready
7	RTS	→	Request to Send
8	CTS	←	Clear to Send

Table 1 – connection of RS232

[7] It is possible to switch between the protocols using, for example, *Modbus Configurator*, which can be downloaded from papouch.com.

[8] The converter is in configuration mode, if Configuration Switch 6 is ON while the converter is being switched on.

[9] These resistors are connected permanently.

[10] Not connected by the manufacturer. Can be connected using the S1 connector inside the device.

Wiegand:

Connector .... removable terminal block with screws

Interval between pulses .... 1.5 to 2.5 ms

Communication line.. Wiegand (1 to 64 bits)

Pin	Description
Data 1	Data 1 signal for the reader
Data 0	Data 0 signal for the reader
GND	Communication line grounding
$+U_{OUT}$	Supply voltage output for the reader [11]

Table 2 – connections for Wiegand

Power supply:

Supply voltage .... DC voltage of 8 to 30 V

Power consumption.......................... usually 20 mA at 15 V (without feeding the reader)

Other parameters

Operating temperature...................... -20 °C to +70 °C

Dimensions .... 70 × 55 × 24 mm

Housing material .... anodized aluminium

Degree of protection.......................... IP 30

Weight.. 90 g

[11] If the supply voltage is higher than 15 V, the voltage at this output is 12 V.
If the supply voltage is lower than 15 V, the voltage at this output is lower that the supply voltage.

Papouch s.r.o.

Data transmission in industry, line and protocol converters, RS232/485/422/USB/Ethernet/GPRS/WiFi, measurement modules, intelligent temperature sensors, I/O modules, customized electronic applications

Address:

Strašnická 3164/1a
102 00 Praha 10

Phone:

+420 267 314 267

Fax:

+420 267 314 269

Website:

www.papouch.com

E-mail:

papouch@papouch.com