CONNECTION TO A „SMART HOME" SYSTEM

VENTO Expert A30 W V.2
VENTO Expert A50-1 W V.2
VENTO Expert A85-1 W V.2
VENTO Expert A100-1 W V.2
VENTO Expert Duo A30-1 W V.2
VENTO Expert A50-1 W V.3

EN CONNECTION GUIDE

BLAUBERG
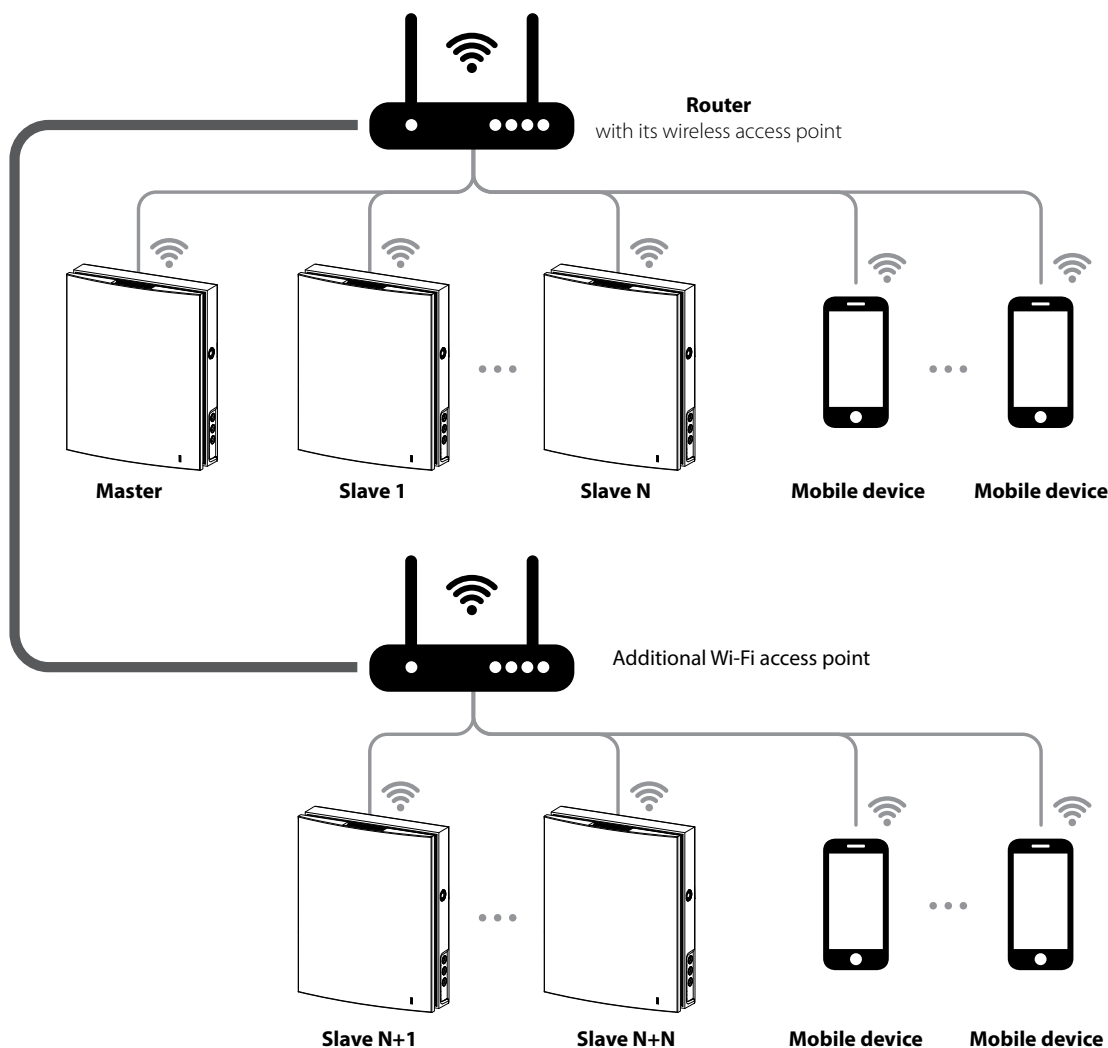*Ventilatoren*

CONTENTS

PURPOSE

This guide covers the connection of VENTO Expert (Duo) A30/50/85/100 W V.2/V.3 series of air handling units to a „Smart Home" system. Control is provided only by Master devices. Slave devices, mobile devices, and „Smart Home" system components are connected to master devices over Wi-Fi. Slave devices are controlled by means of commands received from master devices.

There are two wireless connection patterns available:
1. A master device with a Wi-Fi access point, which can accept up to eight slave connections. If all the available eight connections of the master device are used, a mobile devices or a „Smart Home" system will not be able to connect to the master device.

Master
with a Wi-Fi access point

Slave 1   **Slave 2**   **Slave N**   **Mobile device**

BLAUBERG
Ventilatoren
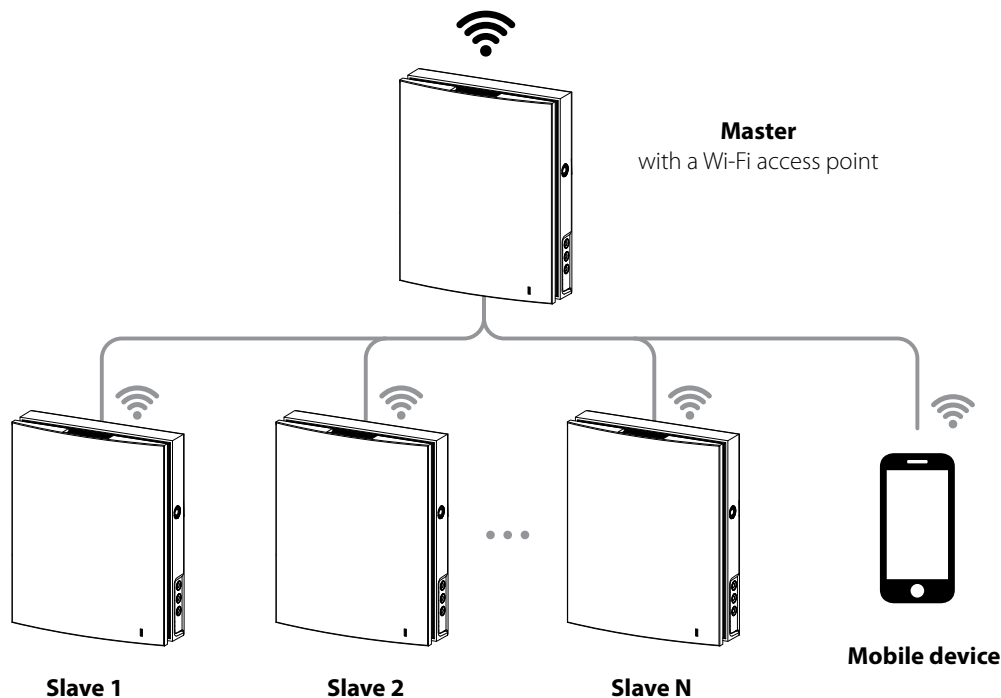
2. Master devices, slave devices, mobile devices and a „Smart Home" system connect to the Wi-Fi access point of the network router. In this case the maximum number of available Wi-Fi connections is limited by the router functionality. If the number of necessary ventilation units is more than the router can accept, you may use an additional Wi-Fi access point to connect the remaining ventilation units. The router network may include several master devices to provide control by zones.

Router
with its wireless access point

Master   **Slave 1**   **Slave N**   **Mobile device**   **Mobile device**

Additional Wi-Fi access point

Slave N+1   **Slave N+N**   **Mobile device**   **Mobile device**

The connection is set up via the mobile application from the Connection – > Wi-Fi setup menu (see the unit data sheet).

NETWORK PARAMETERS

Data is exchanged via the UDP protocol (with broadcast support).

Master device IP address:
- 192.168.4.1 – if the master device runs without a router (connection pattern 1).
- If the master device is connected via a router (connection pattern 2), the IP address is set up via the mobile application (see unit data sheet) and can be defined as static or dynamic (DHCP).

Master device port: 4000.

Maximum packet size: 256 bytes.

BLAUBERG
Ventilatoren

PACKET STRUCTURE

0xFD	0xFD	TYPE	SIZE ID	ID	SIZE PWD	PWD	FUNC	DATA	Chksum L	Chksum H

0xFD	0xFD	: packet beginning character (2 bytes).

TYPE : protocol type (1 byte). Value = 0x02.

SIZE ID : ID block size (1 byte). Value = 0x10.

ID : controller ID. This number is printed on the label (16 characters) applied to the control circuit board or the unit casing.

You can also substitute the ID with „DEFAULT_DEVICEID" code word. The ID can be used:
- To control the master device if it runs without a router (connection pattern 1).
- To search for master devices on the network if a router is used (connection pattern 2). In this case the device will respond to two parameters only: 0x007C and 0x00B9 (see parameter table).

SIZE PWD : PWD block size (1 byte). Possible values: from 0x00 to 0x08.

PWD : device password (permissible characters: „0…9", „a…z", and „A…Z"). The default password is „1111".

This password can be changed via the mobile application from the **Connection –> At home –> Settings menu** (see the unit data sheet).

FUNC : the function number (1 byte). It defines the action with the data and the **DATA** block structure:

0x01: parameter read.
0x02: parameter write. The controller does not send any response regarding the status of the given parameters.
0x03: parameter write with subsequent controller response regarding the status of the given parameters.
0x04: parameter increment with subsequent controller response regarding the status of the given parameters.
0x05: parameter decrement with subsequent controller response regarding the status of the given parameters.
0x06: controller response to the request (FUNC = 0x01, 0x03, 0x04, 0x05).

DATA : the data block. It consists of parameter numbers and their values:

If FUNC = 0x01 or 0x04 or 0x05:

P1	P2	Pn

If FUNC = 0x02 or 0x03 or 0x06:

P1	Value 1	P2	Value 2	Pn	Value n

Parameter numbers(see parameter table) consists of two bytes (the high byte is virtual). By default the high byte of each parameter number in each new packet equals 0x00. The high byte can be changed within a single packet using the special 0xFF command (see below).

P is the low byte of the parameter number. Possible values: 0x00 – 0xFB. The 0xFC – 0xFF values are special commands:

0xFC : change function (**FUNC**) number. The following byte must be the new function number ranging from 0x01 to 0x05. This command is used to organise several functions with different actions into a single packet.

0xFD : parameter not supported by the controller. The following byte is the low byte of the non-supported parameter. This command is used in controller response (**FUNC** = 0x06) to a non-supported parameter read or write request.

0xFE : change a size of the Value for one parameter which follows. The following byte must be the new parameter size followed by the low byte of the parameter number, and then – by the **Value** itself.

0xFF : change the high byte for parameter numbers within a single packet. The following byte must be the new high byte.

Value : parameter value (the default size of value is 1 byte). Byte ordering from least significant byte to most significant byte.
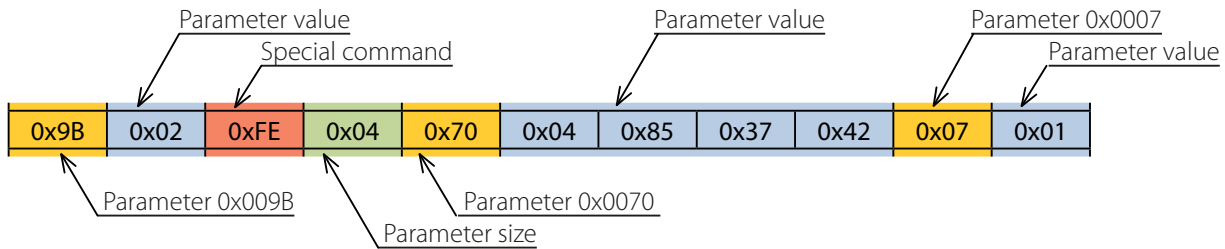
Chksum L **Chksum H** : checksum (2 bytes). This is calculated as the total of bytes beginning with the TYPE byte and ending with the final byte of the **DATA** block.
Chksum L: checksum low byte.
Chksum H: checksum high byte.

BLAUBERG Ventilatoren

EXAMPLES OF USING SPECIAL COMMANDS IN THE DATA BLOCK

Write request (FUNC = 0x03) for parameters number 0x009B, 0x0070, and 0x0007

0x9B	0x02	0xFE	0x04	0x70	0x04	0x85	0x37	0x42	0x07	0x01

- Parameter 0x009B (0x9B), Parameter value (0x02), Special command (0xFE), Parameter size (0x04), Parameter 0x0070 (0x70), Parameter value (0x04 0x85 0x37 0x42), Parameter 0x0007 (0x07), Parameter value (0x01)

Write request details:
- Parameter 0x009B to be assigned the value of 0x02.
- Parameter 0x0070 to be assigned the value of 0x42378504. The value size is 4 bytes as indicated by the special command 0xFE + 0x04.
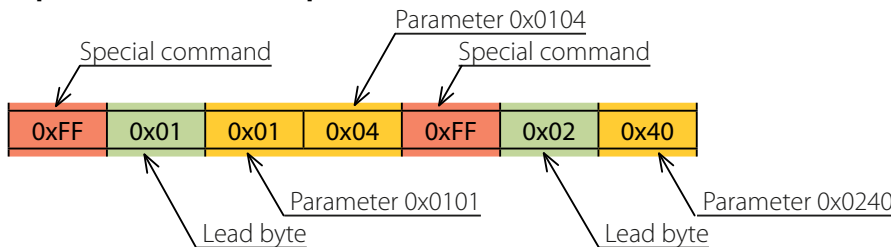- Parameter 0x0007 to be assigned the value of 0x01.

Controller response (FUNC = 0x06) to write request

0x9B	0x02	0xFE	0x04	0x70	0x04	0x85	0x37	0x42	0x07	0x01

- Parameter 0x009B (0x9B), Parameter value (0x02), Special command (0xFE), Parameter size (0x04), Parameter 0x0070 (0x70), Parameter value (0x04 0x85 0x37 0x42), Parameter 0x0007 (0x07), Parameter value (0x01)
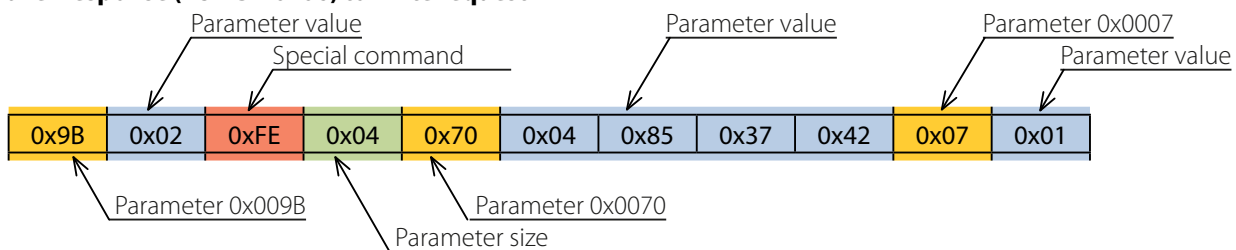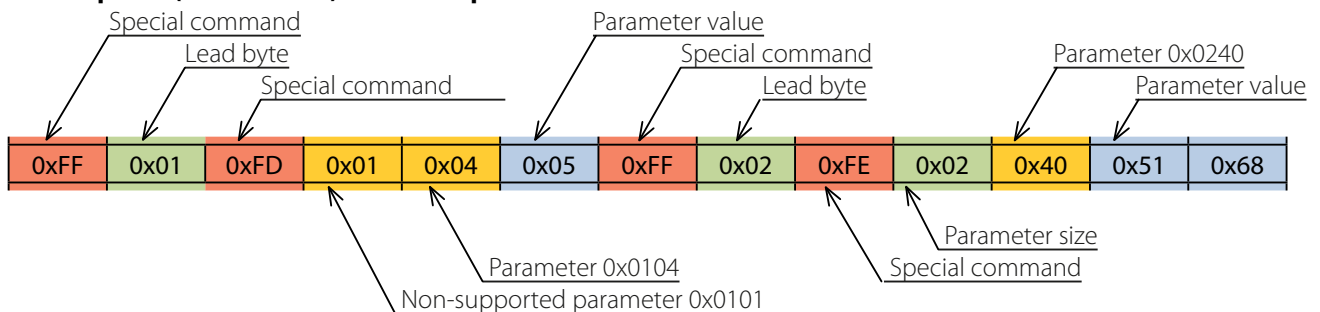
Controller response details:
- Parameter 0x009B equals 0x02.
- Parameter 0x0070 equals 0x42378504. The value size is 4 bytes as indicated by the special command 0xFE + 0x04.
- Parameter 0x0007 equals 0x01.

Read request (FUNC = 0x01) for parameters number 0x0101, 0x0104, and 0x0240

0xFF	0x01	0x01	0x04	0xFF	0x02	0x40

- Special command (0xFF), Lead byte (0x01), Parameter 0x0101 (0x01), Parameter 0x0104 (0x04), Special command (0xFF), Lead byte (0x02), Parameter 0x0240 (0x40)

Controller response (FUNC = 0x06) to write request

0xFF	0x01	0xFD	0x01	0x04	0x05	0xFF	0x02	0xFE	0x02	0x40	0x51	0x68

- Special command (0xFF), Lead byte (0x01), Special command (0xFD), Non-supported parameter 0x0101 (0x01), Parameter 0x0104 (0x04), Parameter value (0x05), Special command (0xFF), Lead byte (0x02), Special command (0xFE), Parameter size (0x02), Parameter 0x0240 (0x40), Parameter value (0x51 0x68)
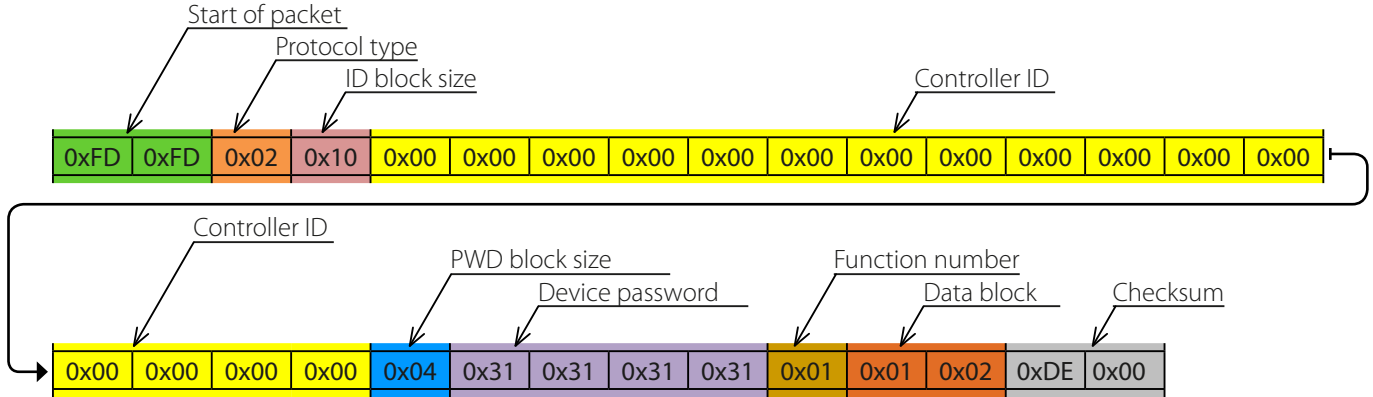
Controller response details:
- Parameter 0x0101 is not supported by the controller as indicated by the special command 0xFD.
- Parameter 0x0104 equals 0x05.
- Parameter 0x0240 equals 0x6851. The value size is 2 bytes as indicated by the special command 0xFE + 0x02.

COMPLETE PACKET EXAMPLES
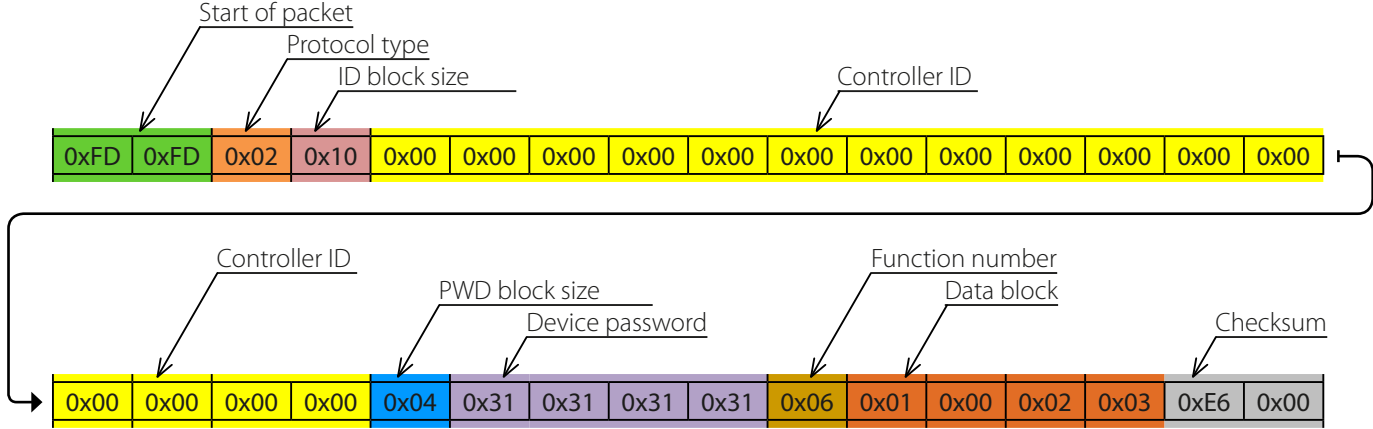
Sending „Smart Home –> Controller" packet

This packet contains a read request (FUNC = 0x01) for parameters number: 0x0001, 0x0002.

Start of packet		Protocol type	ID block size						Controller ID						
0xFD	0xFD	0x02	0x10	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Controller ID				PWD block size	Device password				Function number	Data block		Checksum	
0x00	0x00	0x00	0x00	0x04	0x31	0x31	0x31	0x31	0x01	0x01	0x02	0xDE	0x00

Request details:

- Checksum: 0x00DE.

Sending „Controller –> Smart Home" packet

This packet contains the controller response (FUNC = 0x06) to the write request.

Start of packet		Protocol type	ID block size						Controller ID						
0xFD	0xFD	0x02	0x10	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Controller ID				PWD block size	Device password				Function number	Data block			Checksum	
0x00	0x00	0x00	0x00	0x04	0x31	0x31	0x31	0x31	0x06	0x01	0x00	0x02	0x03	0xE6	0x00

Controller response details:

- Parameter 0x0001 equals 0x00.
- Parameter 0x0002 equals 0x03.
- Checksum: 0x00E6.

BLAUBERG *Ventilatoren*

PARAMETER TABLE

Functions:	R – 0x01	INC – 0x04	RW – 0x03	W – 0x02	DEC – 0x05

Parameter number [Dec./Hex.]	Functions	Description	Possible values	Size [bytes]
1/0x0001	R/W/RW	Unit On/Off	0 — Off 1 — On 2 – Invert	1
2/0x0002	R/W/RW/INC/DEC	Speed number	1 – Speed 1 2 – Speed 2 3 – Speed 3 255 – manual speed setting mode (see parameter 68)	1
6/0x0006	R	Boost mode status	0 – Off 1 – On	1
7/0x0007	R/W/RW/INC/DEC	„Timer" mode (see parameters 770 and 771)	0 – Off 1 – Night mode 2 – „Party" mode	1
11/0x000B	R	Current countdown of the „Timer" mode	Byte 1 – seconds (0…59) Byte 2 – minutes (0…59) Byte 3 – hours (0…23)	3
15/0x000F	R/W/RW	Humidity sensor activation	0 – Off 1 – On 2 – Invert	1
20/0x0014	R/W/RW	Relay sensor activation	0 – Off 1 – On 2 – Invert	1
22/0x0016	R/W/RW	0–10 V* sensor activation	0 – Off 1 – On 2 – Invert	1
25/0x0019	R/W/RW/INC/DEC	Humidity threshold setpoint	40…80 RH%	1
36/0x0024	R	Current RTC battery voltage	0…5000 mV	2
37/0x0025	R	Current humidity	0…100 RH%	1
45/0x002D	R	Current 0–10 V* sensor signal value	0…100 %	1
50/0x0032	R	Current relay sensor state	0 – Off 1 – On	1
58/0x003A	R/W/RW/INC/DEC	Supply fan speed in 1st speed mode**	10…255	1
59/0x003B	R/W/RW/INC/DEC	Exhaust fan speed in 1st speed mode**	10…255	1
60/0x003C	R/W/RW/INC/DEC	Supply fan speed in 2nd speed mode**	10…255	1
61/0x003D	R/W/RW/INC/DEC	Exhaust fan speed in 2nd speed mode**	10…255	1
62/0x003E	R/W/RW/INC/DEC	Supply fan speed in 3rd speed mode**	10…255	1
63/0x003F	R/W/RW/INC/DEC	Exhaust fan speed in 3rd speed mode**	10…255	1
68/0x0044	R/W/RW/INC/DEC	Speed of the fans in the manual speed setting mode	0…255	1
74/0x004A	R	Fan 1 speed	0…5000 rpm	2
75/0x004B	R	Fan 2 speed	0…5000 rpm	2
99/0x0063	R/W/RW/INC/DEC	Filter replacement timer setup**	70…365 days	2
100/0x0064	R	Timer countdown to filter replacement	Byte 1 – minutes (0…59) Byte 2 – hours (0…23) Byte 3 – days (0…181)	3

BLAUBERG
Ventilatoren

Parameter number [Dec./Hex.]	Functions	Description	Possible values	Size [bytes]
101/0x0065	W	Reset timer countdown to filter replacement	Any byte	1
102/0x0066	R/W/RW/INC/DEC	Setpoint of the Boost mode deactivation delay	0…60 minutes	1
111/0x006F	R/W/RW	RTC time	Byte 1 – RTC seconds (0…59) Byte 2 – RTC minutes (0…59) Byte 3 – RTC hours (0…23)	3
112/0x0070	R/W/RW	RTC calendar	Byte 1 – RTC number (1…31) Byte 2 – RTC day of the week (1…7) Byte 3 – RTC month (1…12) Byte 4 – RTC year (0…99)	4
114/0x0072	R/W/RW	"Weekly schedule" mode	0 — Off 1 — On 2 – Invert	1
119/0x0077	R/W/RW	Schedule setup In the read request you should use special command 0xFE and specify the size of the 0x02 parameter value required for selecting the necessary day of the week and time period number. The write request and the controller response use all the 6 bytes. The first time period always starts at 00:00 while the start of each following period begins at the end of the previous one. The final time period always ends at 24:00	**Byte 1** – day of the week: 0 – all days (write only) 1 – Monday 2 – Tuesday 3 – Wednesday 4 – Thursday 5 – Friday 6 – Saturday 7 – Sunday 8 – Mon…Fri (write only) 9 – Sat…Sun (write only) **Byte 2** – period number: 1…4 **Byte 3** – speed number: 0 – standby 1…3 **Byte 4** – reserved: any byte **Byte 5** – minutes to end of period: 0…59 **Byte 6** – hours to end of period: 0…23	6
124/0x007C	R	Device search on the local network, ID	Text („0…9",„A…F")	16
125/0x007D	R/W/RW	Device password	Text („0…9",„a…z",„A…Z")	0–8
126/0x007E	R	Machine hours	Byte 1 – minutes (0…59) Byte 2 – hours (0…23) Byte 3 and Byte 4 – days (0…65535)	4
128/0x0080	W	Reset alarms	Any byte	1
131/0x0083	R	Alarm/warning indicator	0 – No 1 – alarm (highest priority) 2 – warning	1
133/0x0085	R/W/RW	Cloud server operation permission	0 – Off 1 – On 2 – Invert	1
134/0x0086	R	Controller base firmware version and date	Byte 1 – firmware version (major) Byte 2 – firmware version (minor) Byte 3 – day Byte 4 – month Byte 5 and Byte 6 – year	6

BLAUBERG *Ventilatoren*

Parameter number [Dec./Hex.]	Functions	Description	Possible values	Size [bytes]
135/0x0087	W	Restore factory settings	Any byte	1
136/0x0088	R	Filter replacement indicator	0 – filter replacement not required 1 – replace filter	1
148/0x0094	R/W/RW/INC/DEC	Wi-Fi operation mode	1 – Client 2 – Access Point	1
149/0x0095	R/W/RW	Wi-Fi name in Client mode	Text	1…32
150/0x0096	R/W/RW	Wi-Fi password	Text	8…64
153/0x0099	R/W/RW	Wi-Fi data encryption type	48 – OPEN 50 – WPA_PSK 51 – WPA2_PSK 52 – WPA_WPA2_PSK	1
154/0x009A	R/W/RW/INC/DEC	Wi-Fi frequency channel	1…13	1
155/0x009B	R/W/RW	Wi-Fi module DHCP	0 – STATIC 1 – DHCP 2 – Invert	1
156/0x009C	R/W/RW	IP address assigned to Wi-Fi module	**Byte 1** – 0…255, **Byte 2** – 0…255, **Byte 3** – 0…255, **Byte 4** – 0…255	4
157/0x009D	R/W/RW	Wi-Fi module subnet mask	**Byte 1** – 0…255, **Byte 2** – 0…255, **Byte 3** – 0…255, **Byte 4** – 0…255	4
158/0x009E	R/W/RW	Wi-Fi module main gateway	**Byte 1** – 0…255, **Byte 2** – 0…255, **Byte 3** – 0…255, **Byte 4** – 0…255	4
160/0x00A0	W	Apply new Wi-Fi parameters and quit Setup Mode	Any byte	1
162/0x00A2	W	Discard new Wi-Fi parameters and quit Setup Mode	Any byte	1
163/0x00A3	R	Current Wi-Fi module IP address	**Byte 1** – 0…255, **Byte 2** – 0…255, **Byte 3** – 0…255, **Byte 4** – 0…255	4
183/0x00B7	R/W/RW/INC/DEC	Ventilator operation mode	0 – ventilation 1 – heat recovery 2 – supply	1
184/0x00B8	R/W/RW/INC/DEC	0–10 V* sensor threshold setpoint	5…100 %	1

185/0x00B9	R	Unit type	3 – VENTO Expert A50-1 W V.2
			    VENTO Expert A85-1 W V.2
			    VENTO Expert A100-1 W V.2
			4 – VENTO Expert Duo A30-1 W V.2
			5 – VENTO Expert A30 W V.2
				2

252/0x00FC 253/0x00FD 254/0x00FE 255/0x00FF	**Special commands**			
770/0x0302	R/W/RW	Night mode timer setpoint	Byte 1 – minutes (0…59) Byte 2 – hours (0…23)	2
771/0x0303	R/W/RW	„Party" mode timer setpoint	Byte 1 – minutes (0…59) Byte 2 – hours (0…23)	2
772/0x0304	R	Humidity sensor status	0 – below setpoint 1 – over setpoint	1
773/0x0305	R	0–10 V* sensor status	0 – below setpoint 1 – over setpoint	1

*Available for all models except VENTO Expert A30 W V.2
**Available for VENTO Expert A50-1 W V.3.

EXAMPLE OF PROCESSING PACKETS WRITTEN IN C

```c
//=============== Special commands ================//
#define BGCP_CMD_PAGE                         0xFF
#define BGCP_CMD_FUNC                         0xFC
#define BGCP_CMD_SIZE                         0xFE
#define BGCP_CMD_NOT_SUP                      0xFD
//======================================================//

#define BGCP_FUNC_RESP                        0x06

uint8_t receive_data[256];
uint16_t  receive_data_size;
uint8_t State_Power;
uint8_t State_Speed_mode;
char current_id[17] = "002D6E1B34565815"; // Controller ID

//********* Checksum and start of packet check **********//
uint8_t check_protocol(uint8_t *data, uint16_t size)
{
    uint16_t i, chksum1 = 0, chksum2 = 0;
    if((data[0] == 0xFD) && (data[1] == 0xFD))
    {
        for(i = 2; i <= size-3; i++)
            chksum1 += data[i];
        chksum2 = (uint16_t)(data[size-1] << 8) | (uint16_t)(data[size-2]);
        if(chksum1 == chksum2)
            return 1;
        else
            return 0;
    }
    else
        return 0;
}
//**************************************************************//


int main(void)
{
    ...

    if(check_protocol(receive_data, receive_data_size) == 1) // Checksum
    {
        if(receive_data[2] == 0x02) // Protocol type
        {
            if(memcmp(&receive_data[4], current_id, receive_data[3]) == 0) // ID
            {
                uint16_t jump_size = 0, page = 0, param, param_size, r_pos;
                uint8_t flag_check_func = 1, BGCP_func;

                r_pos = 4 + receive_data[3];
                r_pos += 1 + receive_data[r_pos]; // Position in array where FUNC block begins
                //******************** FUNC and DATA ********************//
                for(; r_pos < receive_data_size - 2; r_pos++)
                {
                    //=========== Special commands ===========//
                    param_size = 1;
                    //=== New function number
                    if((flag_check_func == 1) || (receive_data[r_pos] == BGCP_CMD_FUNC))
                    {
                        if(receive_data[r_pos] == BGCP_CMD_FUNC)
                            r_pos++;
                        flag_check_func = 0;
                        BGCP_func = receive_data[r_pos];
                        if(BGCP_func != BGCP_FUNC_RESP) // If the function number is not supported
                            break;
                        continue;
                    }
                    //=== New lead byte value for parameter numbers
                    else if(receive_data[r_pos] == BGCP_CMD_PAGE)
                    {
```

BLAUBERG
Ventilatoren

```
            page = receive_data[++r_pos];
            continue;
        }
        //=== New parameter size value
        else if(receive_data[r_pos] == BGCP_CMD_SIZE)
        {
            param_size = receive_data[++r_pos];
            r_pos++;
        }
        //=== If the parameter is not supported
        else if(receive_data[r_pos] == BGCP_CMD_NOT_SUP)
        {
            r_pos++;
            //******* Processing of non-supported parameters ******//
            param = (uint16_t)(page << 8) | (uint16_t)(receive_data[r_pos]);
            switch(param)
            {
                case 0x0001:
                    break;
                case 0x0002:
                    break;
                 ...
            }
            //*********************//
            continue;
        }
        jump_size = param_size;
        //==================================//

        //******* Processing of supported parameters ******//
        param = (uint16_t)(page << 8) | (uint16_t)(receive_data[r_pos]);
        switch(param)
        {
            case 0x0001:
                State_Power = receive_data[r_pos+1];
                break;
            case 0x0002:
                State_Speed_mode = receive_data[r_pos+1];
                break;
             ...
        }
        //*********************//
        r_pos += jump_size;
    }
    //********************************************************//
        }
      }
    }
}
```

BLAUBERG
Ventilatoren