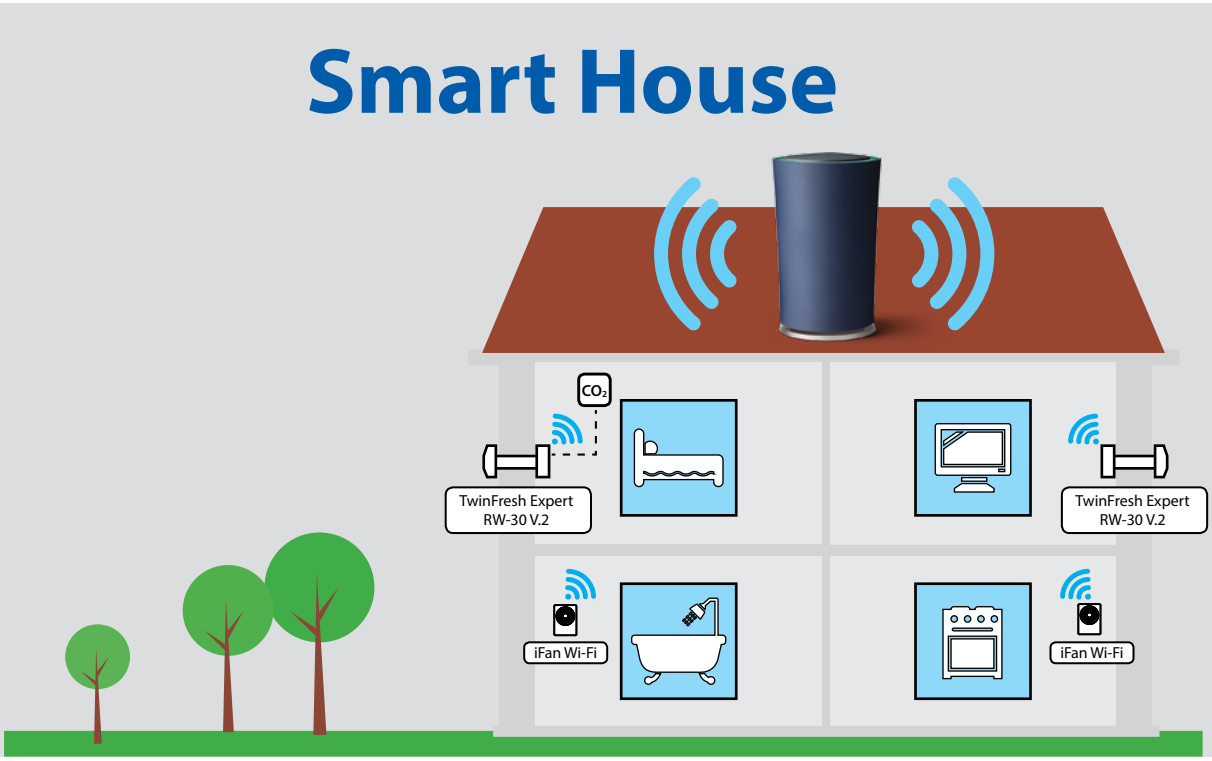
ANSCHLUSSANLEITUNG

IFAN WI-FI



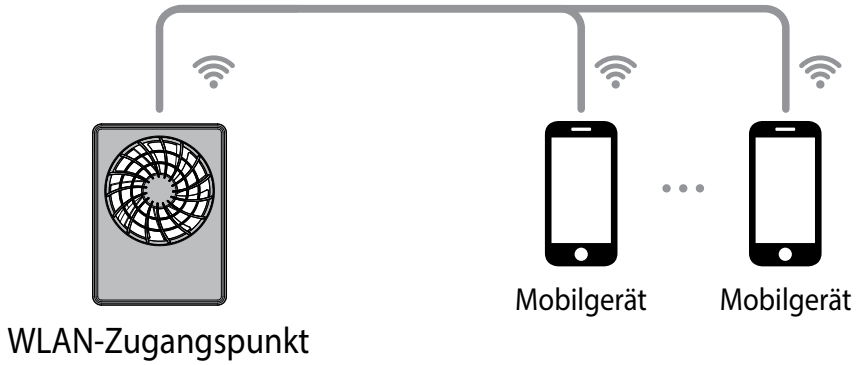
Anschluss an ein „Smart Home“ System

VENTS

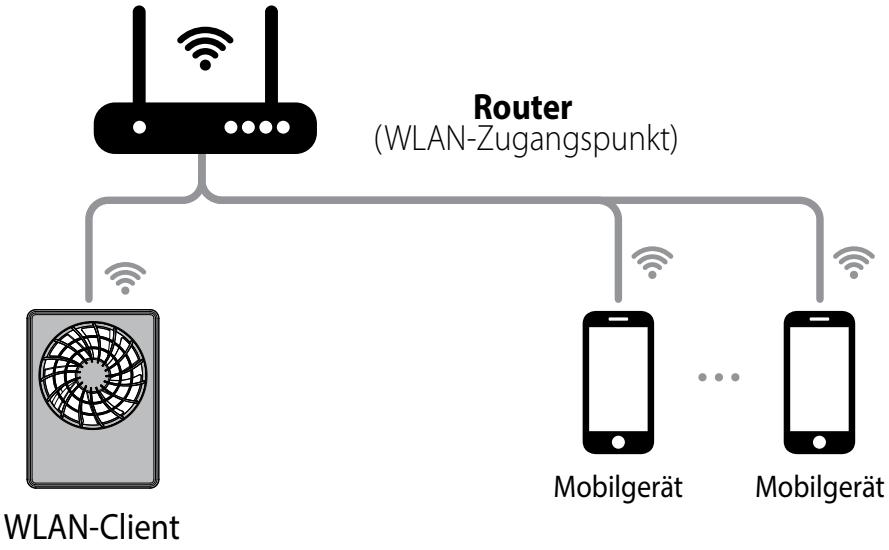
INHALT

NETZANSCHLUSS UND EINSTELLUNG

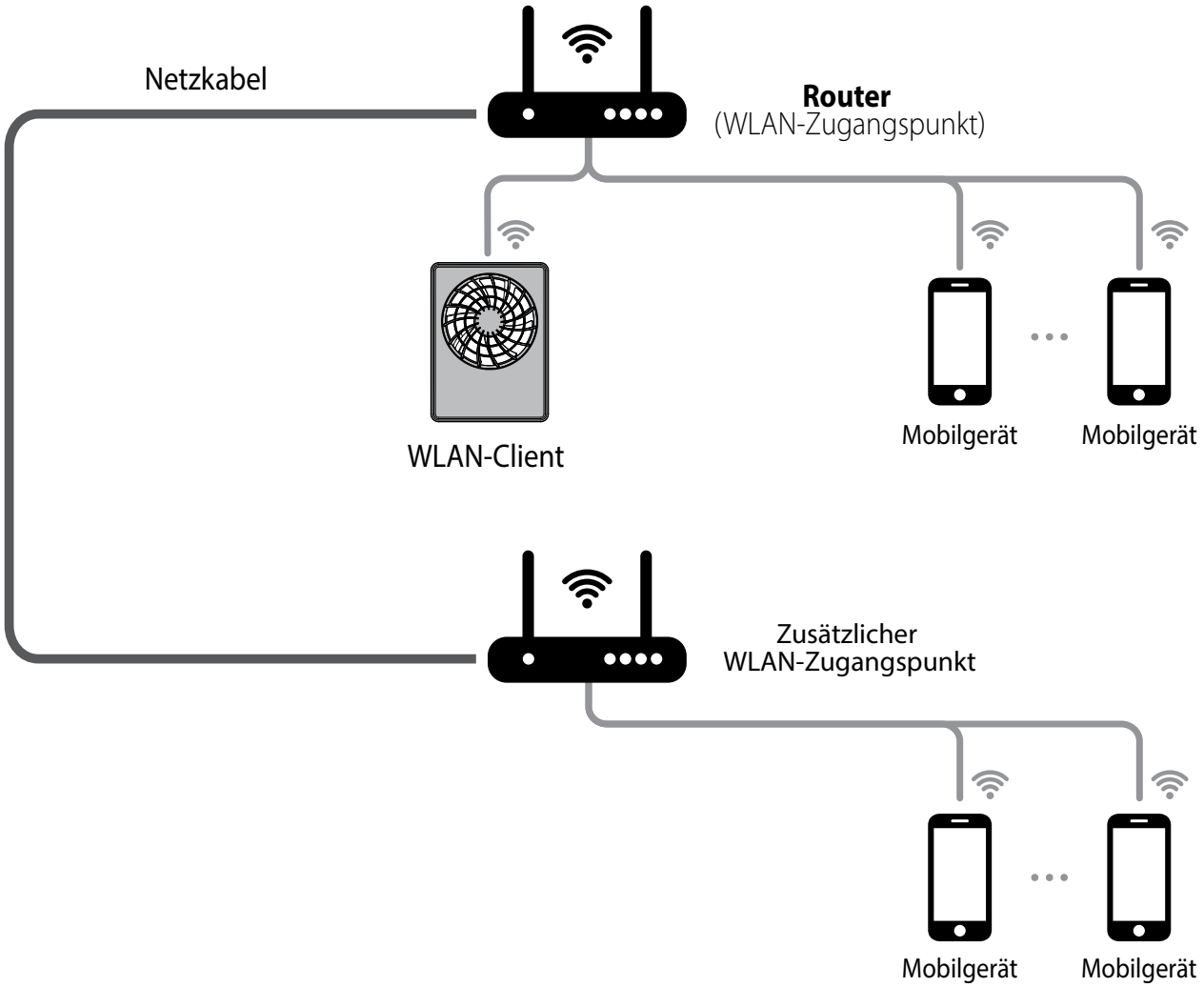
Beispiel 1: Schema des direkten Anschlusses eines Ventilators an das BMS-System „Smart Home“ ohne Verwendung eines Routers. Stellen Sie den Ventilator so ein, dass WLAN im Zugangspunktmodus betrieben wird (siehe Betriebsanleitung des Ventilators). Anmerkung: Es können bis zu acht Steuergeräte angeschlossen werden.



Beispiel 2: Anschlussschema mit einem Router mit einem einzelnen WLAN-Zugangspunkt.
Ein Ventilator, Mobilgeräte und ein BMS-System „Smart Home“ stellen eine Verbindung zum WLAN-Zugangspunkt des Netzwerk-Routers her.



Beispiel 3: Anschlussschema des BMS-Systems „Smart Home“ mit einem Router, an den mehrere WLAN-Zugangspunkte angeschlossen sind.



NETZPARAMETER

Der Datenaustausch erfolgt über das UDP-Protokoll (mit Broadcast-Unterstützung).

IP-Adresse der Master-Anlage:
- 192.168.4.1: Wenn die Master-Anlage ohne Router läuft (Anschlussschema Nr. 1).
- Wenn die Master-Anlage über einen Router verbunden ist (Anschlussschema Nr. 2), wird die IP-Adresse über die App eingestellt (siehe Betriebsanleitung des Geräts) und kann statisch oder dynamisch (DHCP) bestimmt werden.

Port der Master-Anlage: 4000

Maximale Paketgröße: 256 Bytes

PAKETSTRUKTUR

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

0xFD **0xFD** : Paketanfangszeichen (2 Bytes)

TYPE : Protokolltyp (1 Byte). Wert = 0x02

SIZE ID : **ID**-Blockgröße (1 Byte). Wert = 0x10

ID : ID-Nummer der Steuereinheit. Diese Nummer ist auf dem Etikett (16 Zeichen) auf der Steuerplatine oder dem Gerätegehäuse angegeben.

Sie können die ID-Nummer auch durch das Codewort "DEFAULT_DEVICEID" ersetzen.
Die ID-Nummer kann verwendet werden:

- Zur Steuerung, wenn die Master-Anlage ohne Router läuft (Anschlussschema Nr. 1).
- Um im Netzwerk nach Master-Anlagen zu suchen, wenn ein Router verwendet wird (Anschlussschema Nr. 2). In diesem Fall reagiert die Anlage nur auf zwei Parameter: 0x007C und 0x00B9 (siehe Parametertabelle).

SIZE PWD : **PWD**-Blockgröße (1 Byte). Mögliche Werte: von 0x00 bis 0x08.

PWD : Passwort der Anlage (zulässige Zeichen: "0… 9", "a… z" und "A… Z"). Das Standardpasswort lautet "1111".
Dieses Passwort kann über die App im Menü **Verbindung** -> **Lokal**-> **Einstellungen** geändert werden (siehe Betriebsanleitung des Geräts).

FUNC : Funktionsnummer (1 Byte). Es definiert die Aktion anhand der Daten und der **DATA**-Blockstruktur:
0x01: Parameterlesen
0x02: Parameterschreiben. Die Steuereinheit sendet keine Antwort bezüglich des Status der angegebenen Parameter. Schreiben der Parameter
0x03: Parameterschreiben mit anschließender Antwort der Steuereinheit bezüglich des Status der angegebenen Parameter.
0x04: Parameterinkrement mit anschließender Antwort der Steuereinheit bezüglich des Status der angegebenen Parameter.
0x05: Parameterdekrement mit anschließender Antwort der Steuereinheit bezüglich des Status der angegebenen Parameter.
0x06: Antwort der Steuereinheit auf die Anfrage (FUNC = 0x01, 0x03, 0x04, 0x05).

DATA : Datenblock. Er besteht aus Parameternummern und ihren Werten:

Wenn FUNC = 0x01 oder 0x04 oder 0x05:

P1	P2	Pn

Wenn FUNC = 0x02 oder 0x03 oder 0x06:

P1	Value 1	P2	Value 2	Pn	Value n

Parameternummern (siehe Parametertabelle) bestehen aus zwei Bytes (High-Byte ist virtuell).
Standardmäßig entspricht das High-Byte jeder Parameternummer in jedem neuen Paket 0x00.
Das High-Byte kann innerhalb eines einzelnen Pakets mit dem speziellen Befehl **0xFF** geändert werden (siehe unten).

P : Low-Byte der Parameternummer. Mögliche Werte: 0x00–0xFB. Die Werte 0xFC–0xFF sind Spezialbefehle:

0xFC : Funktionsnummer ändern (**FUNC**). Das folgende Byte muss die neue Funktionsnummer sein, die von 0x01 bis 0x05 reicht. Dieser Befehl wird verwendet, um mehrere Funktionen mit unterschiedlichen Aktionen in einem einzigen Paket zu organisieren.

0xFD : Parameter wird von der Steuereinheit nicht unterstützt. Das nachfolgende Byte ist das Low-Byte des nicht unterstützten Parameters. Dieser Befehl wird in der Antwort der Steuereinheit (**FUNC** = 0x06) auf eine nicht unterstützte Lese- oder Schreibanforderung von Parametern verwendet.

0xFE : Die Größe des Parameterwerts **Value** für einen der folgenden Parameter ändern. Das nachfolgende Byte muss die neue Parametergröße sein, gefolgt vom Low-Byte der Parameternummer und dann – vom Wert (**Value**) selbst.

0xFF : Das High-Byte für Parameternummern innerhalb eines einzelnen Pakets ändern. Das nachfolgende Byte muss das neue High-Byte sein.

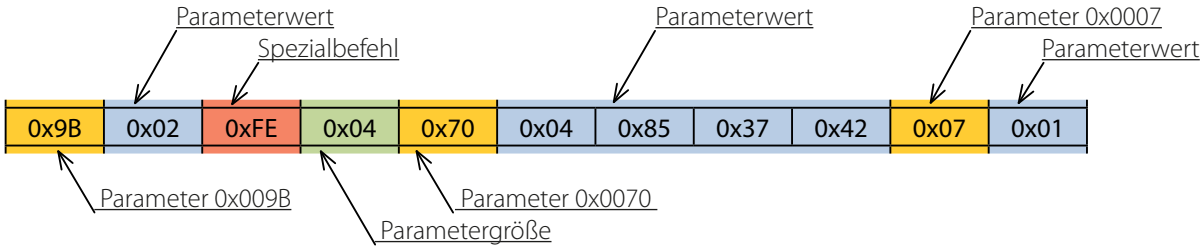
Value : Parameterwert (Standardgröße ist 1 Byte). Byteanordnung von Low-Byte zu High-Byte.

Chksum L **Chksum H** : Prüfsumme (2 Bytes). Dies wird als die Gesamtzahl von Bytes berechnet, die mit dem **TYPE**-Byte beginnen und mit dem letzten Byte des **DATA**-Blocks enden.

Chksum L: Low-Byte der Prüfsumme
Chksum H: High-Byte der Prüfsumme

ANWENDUNGSBEISPIELE DER SPEZIELLEN BEFEHLE IM DATENBLOCK

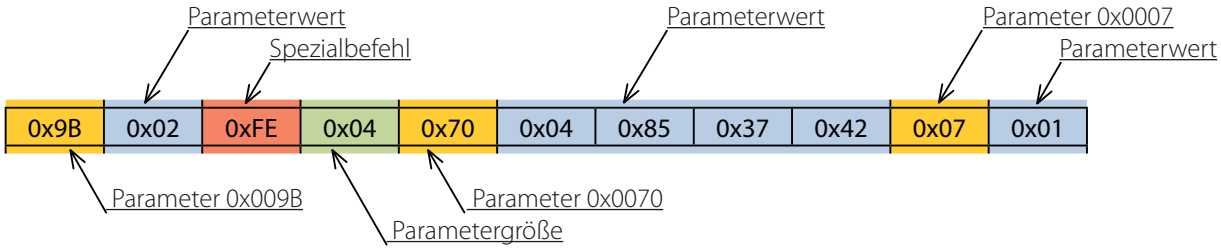
Schreibanforderung (FUNC = 0x03) für Parameter mit den Nummern 0x009B, 0x0070 und 0x0007



Details der Schreibanforderung:

- Dem Parameter 0x009B muss der Wert 0x02 zugewiesen werden.
- Dem Parameter 0x0070 muss der Wert 0x42378504 zugewiesen werden. Die Wertgröße beträgt 4 Bytes, wie durch den Spezialbefehl 0xFE+0x04 angegeben.
- Dem Parameter 0x0007 muss der Wert 0x01 zugewiesen werden.

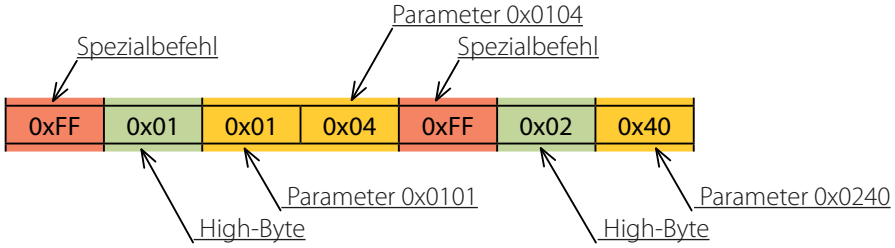
Antwort der Steuereinheit (FUNC = 0x06) auf die Schreibanforderung



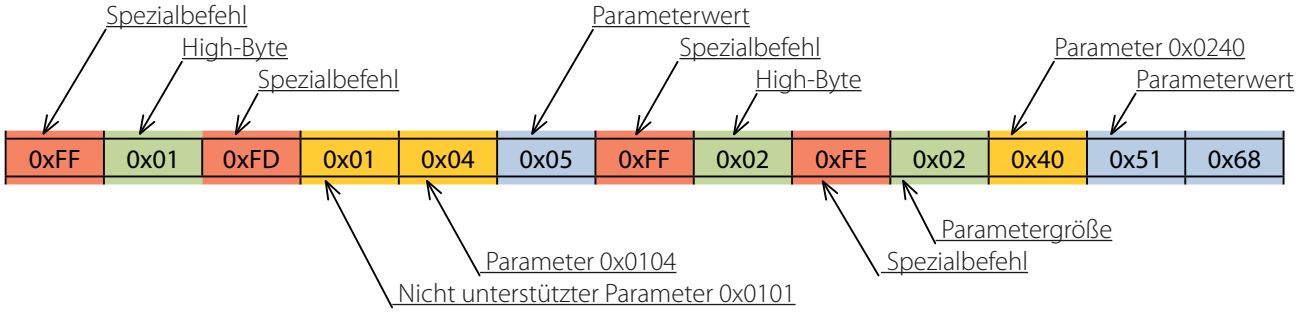
Antwort der Steuereinheit:

- Parameter 0x009B entspricht 0x02.
- Parameter 0x0070 entspricht 0x42378504. Die Wertgröße beträgt 4 Bytes, wie durch den Spezialbefehl 0xFE + 0x04 angegeben.
- Parameter 0x0007 entspricht 0x01.

Leseanforderung (FUNC = 0x01) für Parameter mit den Nummern 0x0101, 0x0104 und 0x0240



Antwort der Steuereinheit (FUNC = 0x06) zur Leseanforderung



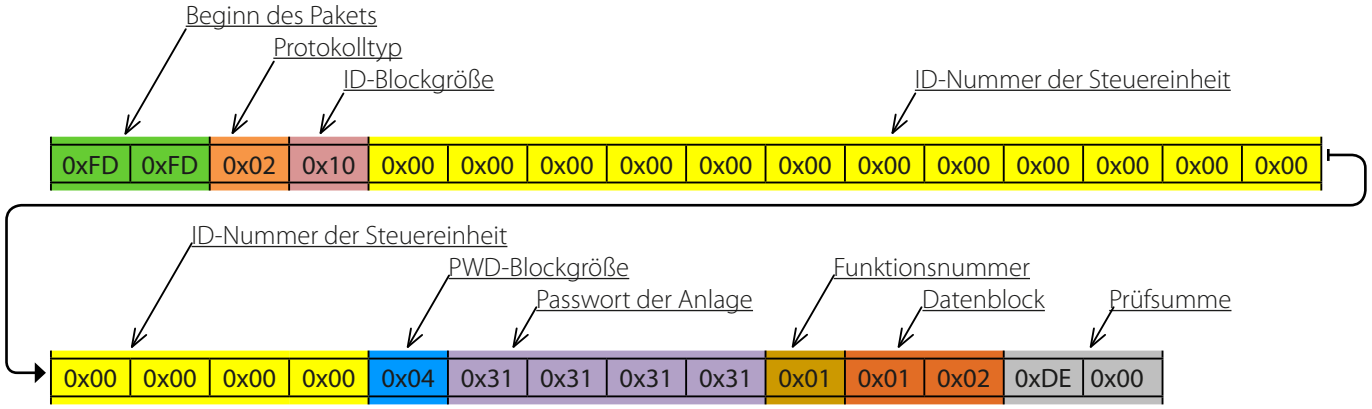
Antwort der Steuereinheit:

- Der Parameter 0x0101 wird von der Steuereinheit nicht unterstützt, wie durch den Spezialbefehl 0xFD angegeben.
- Parameter 0x0104 entspricht 0x05.
- Parameter 0x0240 entspricht 0x6851. Die Wertgröße beträgt 2 Bytes, wie durch den Spezialbefehl 0xFE + 0x02 angegeben.

BEISPIELE EINES KOMPLETTEN PAKETS

Senden des Pakets "Smart Home–> Steuereinheit"

Dieses Paket enthält eine Leseanforderung (FUNC = 0x01) für Parameter mit den Nummern: 0x0001, 0x0002.

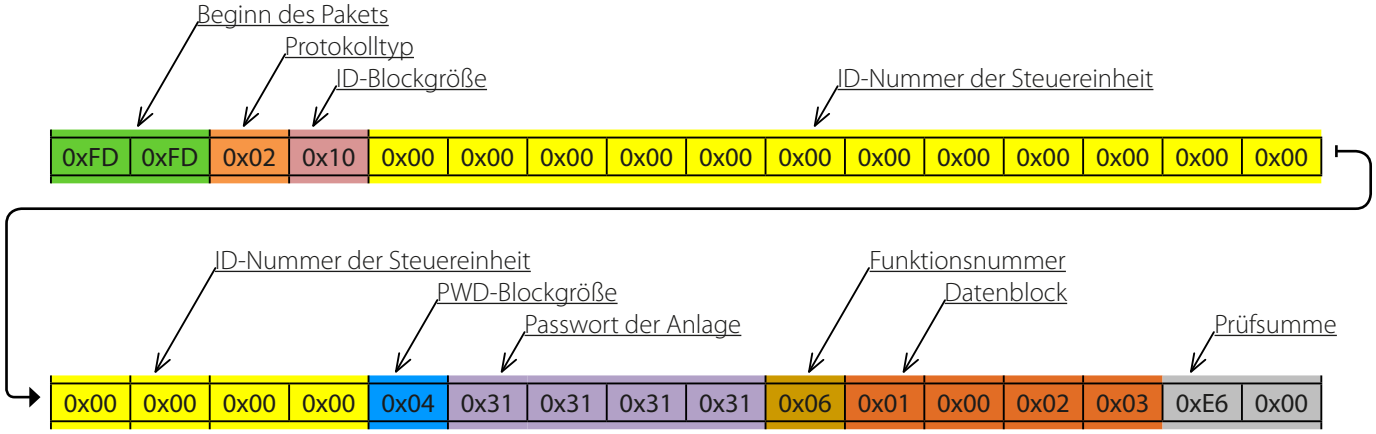


Anfragedetails:

- Prüfsumme: 0x00DE

Senden des Pakets "Steuereinheit –> Smart Home"

Dieses Paket enthält die Antwort der Steuereinheit (FUNC = 0x06) zur Schreibanforderung.



Antwort der Steuereinheit:

- Parameter 0x0001 entspricht 0x00.
- Parameter 0x0002 entspricht 0x03.
- Prüfsumme: 0x00E6

PARAMETERTABELLE

Funktionen:

R – 0x01 **INC** – 0x04 **RW** – 0x03

W – 0x02 **DEC** – 0x05

Parameter-Nummer, Dez./Hex.	Funktionen	Beschreibung	Mögliche Werte	Größe, Bytes
1/0x0001	R/W/RW	Ventilator Ein/Aus	0: Aus 1: Ein 2: invertieren	1
2/0x0002	R	Batteriestand	0: entladen (fehlt) 1: Ladung ist normal	1
3/0x0003	R/W/RW	Auswahl des 24 Stunden-Betriebs	0: Aus 1: Ein 2: invertieren	1
4/0x0004	R	Aktuelle Wert der Lüftungsstufe des Ventilators (Drehzahl)	0...6000 RPM	2
5/0x0005	R/W/RW	Boost Ein/Aus	0: Aus 1: Ein 2: invertieren	1
6/0x0006	R	Aktueller Wert der Countdown des BOOST-Timers in Sekunden	0...86400 Sekunden	3
7/0x0007	R	Aktueller Status des integrierten Timers	0: Aus 1: Ein	1
8/0x0008	R	Aktueller Status des Ventilatorbetriebs durch Feuchtigkeitssensor	0: Aus 1: Ein	1
10/0x000A	R	Aktueller Status des Ventilatorbetriebs durch Temperatursensor	0: Aus 1: Ein	1
11/0x000B	R	Aktueller Status des Ventilatorbetriebs durch Bewegungssensor	0: Aus 1: Ein	1
12/0x000C	R	Aktueller Status des Ventilatorbetriebs durch Signal von einem externen Schalter	0: Aus 1: Ein	1
13/0x000D	R	Aktueller Status des Ventilatorbetriebs im Intervalllüftungsmodus	0: Aus 1: Ein	1
14/0x000E	R	Aktueller Status des Ventilatorbetriebs im SILENT-Modus	0: Aus 1: Ein	1
15/0x000F	R/W/RW	Zulassung des Betriebs durch Feuchtigkeitssensor	0: Aus 1: im automatischen Betrieb 2: im manuellen Betrieb	1
17/0x0011	R/W/RW	Zulassung des Betriebs durch Temperatursensor	0: Aus 1: Ein 2: invertieren	1
18/0x0012	R/W/RW	Zulassung des Betriebs durch Bewegungssensor	0: Aus 1: Ein 2: invertieren	1
19/0x0013	R/W/RW	Zulassung des Betriebs durch Signal von einem externen Schalter	0: Aus 1: Ein 2: invertieren	1
24/0x0018	R/W/RW/INC/DEC	Sollwert der Lüftungsstufe Max	30...100 %	1
26/0x001A	R/W/RW/INC/DEC	Sollwert der Lüftungsstufe Silent	30...100 %	1

Parameter-Nummer, Dez./Hex.	Funktionen	Beschreibung	Mögliche Werte	Größe, Bytes
27/0x001B	R/W/RW/INC/DEC	Sollwert der Lüftungsstufe der Intervalllüftung	30...100 %	1
29/0x001D	R/W/RW	Aktivierung des Betriebs der Intervalllüftung	0: Aus 1: Ein 2: invertieren	1
30/0x001E	R/W/RW	Aktivierung des Betriebs Silent	0: Aus 1: Ein 2: invertieren	1
31/0x001F	R/W/RW	Startzeit des Betriebs Silent in Sekunden	0...86400 Sekunden	3
32/0x0020	R/W/RW	Endzeit des Betriebs Silent in Sekunden	0...86400 Sekunden	3
33/0x0021	R/W/RW	Aktuelle Zeit der internen Realzeituhr in Sekunden	0...86400 Sekunden	3
35/0x0023	R/W/RW/INC/DEC	Sollwert des Nachlaufschalters/ BOOST	0: Aus 2: 5 Minuten 3: 15 Minuten 4: 30 Minuten 6: 60 Minuten	1
36/0x0024	R/W/RW/INC/DEC	Sollwert der Einschaltverzögerung	0: Aus 1: 2 Minuten 2: 5 Minuten	1
37/0x0025	W	Zurücksetzen der Parameter auf Werkseinstellungen	Jedes Byte	1
124/0x007C	R	Suche der Anlagen im lokalen Netzwerk Ethernet	Text ("0…9", "A…F")	16
134/0x0086	R	Firmware-Version und Datum der Firmware-Version der Steuereinheit	Byte 1: Firmware-Version (major) Byte 2: Firmware-Version (minor) Byte 3: Tag Byte 4: Monat Byte 5 und 6: Jahr	6
148/0x0094	R/W/RW	WLAN-Betrieb	1: Client 2: Access Point	1
149/0x0095	R/W/RW	WLAN-Name im Client-Betrieb	Text	1 … 32
150/0x0096	R/W/RW	WLAN-Passwort	Text	8 … 64
153/0x0099	R/W/RW	WLAN-Verschlüsselungstechnologie	48: OPEN 50: WPA_PSK 51: WPA2_PSK 52: WPA_WPA2_PSK	1
154/0x009A	R/W/RW	WLAN-Kanalfrequenz	1...13	1
155/0x009B	R/W/RW	WLAN-Modul DHCP	0: STATIC 1: DHCP 2: invertieren	1
156/0x009C	R/W/RW	Zugewiesene IP-Adresse des WLAN-Moduls	Byte 1: 0…255 Byte 2: 0…255 Byte 3: 0…255 Byte 4: 0…255	4

Parameter-Nummer, Dez./Hex.	Funktionen	Beschreibung	Mögliche Werte	Größe, Bytes
157/0x009D	R/W/RW	Subnetzmaske des WLAN-Moduls	Byte 1: 0…255 Byte 2: 0…255 Byte 3: 0…255 Byte 4: 0…255	4
158/0x009E	R/W/RW	Haupt-Gateway des WLAN-Moduls	Byte 1: 0…255 Byte 2: 0…255 Byte 3: 0…255 Byte 4: 0…255	4
160/0x00A0	W	WLAN-Parameter übernehmen und Einstellmodus des WLAN-Moduls verlassen	Jedes Byte	1
163/0x00A3	R	Aktuelle IP-Adresse des WLAN-Moduls	0...255	4
185/0x00B9	R	Anlagentyp		2

BEISPIEL DER PAKETVERARBEITUNG, IN C GESCHRIEBEN

```
//================ Spezialbefehle =================//
#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"; // ID-Nummer der Steuereinheit

//********* Anfang der Prüfsumme und Beginn des Pakets ***********//
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) // Prüfsumme
    {
        if(receive_data[2] == 0x02) // Protokolltyp
        {
            if(memcmp(&receive_data[4], current_id, receive_data[3]) == 0) // ID-Nummer
            {
                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 im Array, wo FUNC-Block startet
                //******************** FUNC und DATA *********************//
                for(; r_pos < receive_data_size - 2; r_pos++)
                {
                    //=========== Spezialbefehle ===========//
                    param_size = 1;
                    //=== neue Funktionsnummer
                    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) // wenn die Funktionsnummer nicht unterstützt wird
                           break;
                        continue;
                    }
                    //=== neuer High-Byte-Wert für Parameternummer
                    else if(receive_data[r_pos] == BGCP_CMD_PAGE)
                    {
```

```
                    page = receive_data[++r_pos];
                    continue;
                }
                //=== neuer Wert der Parametergröße
                else if(receive_data[r_pos] == BGCP_CMD_SIZE)
                {
                    param_size = receive_data[++r_pos];
                    r_pos++;
                }
                //=== falls der Parameter nicht unterstützt wird
                else if(receive_data[r_pos] == BGCP_CMD_NOT_SUP)
                {
                    r_pos++;
                    //******* Verarbeitung der nicht unterstützten Parameter ******//
                    param = (uint16_t)(page << 8) | (uint16_t)(receive_data[r_pos]);
                    switch(param)
                    {
                       case 0x0001:
                          break;
                       case 0x0002:
                          break;
                        ...
                    }
                    //***********************//
                    continue;
                }
                jump_size = param_size;
                //===================================//

                //******* Verarbeitung der unterstützten Parameter ******//
                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;
            }
            //*********************************************************//
        }
    }
}
```

V168-1DE-01