**LIEBHERR Supermarket Communication**

**Modbus-RTU Communication Protocol**

**Rev. 1.3 Platform - 02/2021**

1. **General Information**
   1. **Serial configuration**

Physical Layer: RS485

Baud Rate: 19200 bps (optional 9600, 38400, 56000 bps)

Data Length: 8 bit

Parity / Stopbits: Even +1 Sb (optional: odd +1 Sb, none + 2 Sb, none + 1 Sb)

Inter Frame Space: silent for 3.5 characters

Timeout Time: 250 ms

* 1. **Addressing**

The slaves with *LiebherrOpenBus* can have an address in the range from 1 to 247. Address 0 is used for broadcast messages. It is not a valid controller address.

Due to the mostly heterogeneous bus users each device should be addressed individually. That means the light on or off command is sent to each address individually. Broadcast messages are ignored by the device.

* 1. **Physical Layer / Wiring**

Modbus is defined as a 3-wire bus. The lines B+ and A- are the differential signal lines in according with EIA/TIA-485 standard. A common line must also interconnect all the devices of the bus (see “MODBUS over serial line specification and implementation guide V1.02” from Modbus-IDA.org).

Use standard CAT5 (or better) patch cables to connect the cabinets.

The pin-out of the connector is as follows.

1: B+

1

8

2: A-

3: GND, common

4: B+

5: B+

RJ45 female connector 6: GND, common

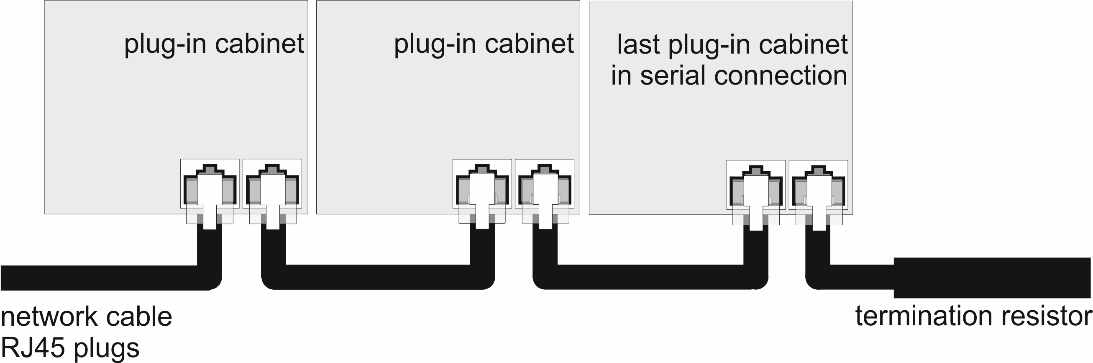
(front view) 7: A-

8: A-

Attention: Due to compatibility with other devices in commercial refrigeration the pin-out is different from the standard pin-out.

The MODBUS has to be one line (without trunks) and it has to be terminated with a resistor 120 Ohms at the end of the line. A termination resistor with RJ45-plug is available. The resistor is between pin 2 and pin 4 (B+ / A-).

The MODBUS master must provide a full termination with Pull Up, Pull Down and termination resistors.



Cable connection between cabinets with termination resistor

1. **Commands**

For the communication between master and slave devices commands are defined to obtain the data access. In Modbus specification these commands are called *function codes*.

Accessing data needs addressing the related register.

In some Modbus parameter descriptions the register numbers are given. Normally register numbers start with 1 while the corresponding register addresses start with 0. In this documentation the register address is given for each parameter which has to be set in the message without offset.

The register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte. For each register, the first byte contains the high order bits and the second contains the low order bits.

For more detailed information please refer to documentation of http://www.modbus.org.

* 1. **Read Coils (0x01)**

This function code is used to read the contents of bit registers in a slave. The request message specifies the starting register address and the number of coils. If the requested number of coils is greater than 1, the requested coils must have continuous addresses.

Request from master (8 bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x01 |
| starting register address MSB | 0x0000 – 0xFFFF |
| starting register address LSB |
| quantity of coils MSB | N (1 – 2000) |
| quantity of coils LSB |
| CRC LSB |  |
| CRC MSB |

Response from slave (5 + N bytes):

|  |  |
| --- | --- |
| slave address | 1 – 247 |
| function code | 0x01 |
| byte count registers | N |
| coils 1-8 | 1 = LSB, 8= MSB |
| following coils | (if N > 8) |
| CRC LSB |  |
| CRC MSB |

Note: Function code 0x02 (Read discrete inputs) is also supported.

* 1. **Read Holding Registers (0x03)**

This function code is used to read the contents of holding registers in a slave. The request message specifies the starting register address and the number of registers. If the requested number of registers is greater than 1, the requested registers must have continuous adresses.

Request from master (8 bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x03 |
| starting register address MSB | 0x0000 – 0xFFFF |
| starting register address LSB |
| quantity of registers MSB | N (1 – 125) |
| quantity of registers LSB |
| CRC LSB |  |
| CRC MSB |

Response from slave (5 + 2\*N bytes):

|  |  |
| --- | --- |
| slave address | 1 – 247 |
| function code | 0x03 |
| byte count registers | 2 x N |
| 1 register value MSB |  |
| 1 register value LSB |
| N register value MSB | (If N > 1) |
| N register value LSB |
| CRC LSB |  |
| CRC MSB |

Note: Function code 0x04 (Read Input registers) is also supported.

* 1. **Write Single Coil (0x05)**

This function code is used to write a single bit as a command to either ON or OFF in a slave. A value of 0xFF00 requests the register to be ON. A value of 0x0000 requests it to be OFF. All other values are illegal and will not affect the register. The request message specifies the address of the register to be forced.

The normal response is an echo of the request, returned after the register state has been written.

Request from master (8 bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x05 |
| register address MSB | 0x0000 – 0xFFFF |
| register address LSB |
| output value MSB | 0xFF00 or 0x0000 |
| output value LSB |
| CRC LSB |  |
| CRC MSB |

Response from slave (8 bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x05 |
| register address MSB | 0x0000 – 0xFFFF |
| register address LSB |
| output value MSB | 0xFF00 or 0x0000 |
| output value LSB |
| CRC LSB |  |
| CRC MSB |

* 1. **Write Single Register (0x06)**

This function code is used to write the contents of holding registers in a slave. The request message specifies the register address, the number of registers to be written, the quantity of bytes and the register values.

Request from master (8 bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x06 |
| register address MSB | 0x0000 – 0xFFFF |
| register address LSB |
| register value MSB |  |
| register value LSB |
| CRC LSB |  |
| CRC MSB |  |

Response from slave (8 bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x06 |
| register address MSB | 0x0000 – 0xFFFF |
| register address LSB |
| register value MSB |  |
| register value LSB |
| CRC LSB |  |
| CRC MSB |

* 1. **Write multiple Coils (0x0F)**

This function code is used to write the contents of multiple bits in a slave. The request message specifies the starting register address, the number of registers to be written, the quantity of bytes and the register values. If the number of coils to be written is greater than 1, the coils must have continous adresses.

Request from master (9 + N bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x10 |
| starting register address MSB | 0x0000 – 0xFFFF |
| starting register address LSB |
| quantity of coils MSB | N (1-1968) |
| quantity of coils LSB |
| byte count registers | N |
| values 1, coils 1-8 | 1 = LSB, 8 = MSB |
| values N, coils 9 following | (If N > 8) |
| CRC LSB |  |
| CRC MSB |

Response from slave (8 bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x0F |
| starting register address MSB | 0x0000 – 0xFFFF |
| starting register address LSB |
| quantity of coils MSB | (1 – 1968) |
| quantity of coils LSB |
| CRC LSB |  |
| CRC MSB |

* 1. **Write multiple Registers (0x10)**

This function code is used to write the contents of multiple holding registers in a slave. The request message specifies the starting register address, the number of registers to be written, the quantity of bytes and the register values.

Request from master (8 + 2 x N bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x10 |
| starting register address MSB | 0x0000 – 0xFFFF |
| starting register address LSB |
| quantity of registers MSB | N (1-123) |
| quantity of registers LSB |
| byte count registers | 2 x N |
| 1 register value MSB |  |
| 1 register value LSB |
| N register value MSB | (If N > 1) |
| N register value LSB |
| CRC LSB |  |
| CRC MSB |

Response from slave (8 bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x10 |
| starting register address MSB | 0x0000 – 0xFFFF |
| starting register address LSB |
| quantity of registers MSB | (1 – 123) |
| quantity of registers LSB |
| CRC LSB |  |
| CRC MSB |

* 1. **Diagnostic (0x08), Subcode 0 (Echo)**

This function code is used for diagnostic purposes. Liebherr Open Bus supports Sub-function code 0.

Request from master (6 + N bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x08 |
| Subfunction MSB | 0x0000 |
| Subfunction LSB |
| Data | N Bytes, N <= 250 |
| CRC LSB |  |
| CRC MSB |

Response from slave (6 + N bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x08 |
| Subfunction MSB | 0x0000 |
| Subfunction LSB |
| Data received from master | N Bytes |
| CRC LSB |  |
| CRC MSB |

1. **CRC Checksum**

Each transmitted message contains a CRC checksum which is calculated by the transmitting device. The receiver compares this value in the CRC field with a calculated value on receiving the message. An error occurs if the calculated CRC value is not equal to the received one.

The low-order byte (LSB) of the CRC checksum is transmitted first, followed by the high-order byte (MSB).

The following code (in C) is for calculating the CRC checksum:

#define MODBUS\_POLYNOM 0xA001

unsigned int crc16;

void modbusCRC16 (unsigned char\* msg, unsigned char lenMsg) {

unsigned char byteIndex;

unsigned char n;

unsigned char bitValue;

crc16 = 0xFFFF;

for(byteIndex=0;byteIndex<lenMsg;byteIndex++) {

crc16 ^= msg[byteIndex];

for(n=0;n<8;n++) {

bitValue = crc16 & 0x0001;

crc16 = crc16 >> 1;

if(bitValue == 1) crc16 ^= MODBUS\_POLYNOM;

}

}

}

1. **Exception Codes**

When a master device sends a request to a slave device it expects a normal response. The normal response contains the requested data.

Each device with *LiebherrOpenBus* answers with an exception code when it has received a command from a master but it is not possible to execute the command. The exception codes echoes the received function code with bit 7 set.

The exception response has the following format (5 bytes):

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | value + 0x80 |
| exception code | 01, 02, 03, 04, 06 |
| CRC LSB |  |
| CRC MSB |

Possible exception codes:

|  |  |  |
| --- | --- | --- |
| Code | Name | Meaning |
| 01 | Illegal function | The received function code is not supported by the slave. The function codes 0x03 and 0x10 are allowed only. |
| 02 | Illegal data address | The data address received in the request from a master is not valid in the slave. Only the parameters listed in chapter 5 have valid addresses. |
| 03 | Illegal data value | The value contained in the data field of a slave message is not valid because it is out of the allowable range. |
| 04 | Slave device error | An unrecoverable error occurred while the slave was performing the requested action. |
| 06 | Slave device busy | The slave is not able to execute the request because of internal operations. The master should repeat the request later when the salve is free. |

1. **Register Addresses**
   1. **Device Identification**

LiebherrOpenBus provides two different ways to identify the connected device.

1. **Reading device identification with function code 0x03 (read holding register)**

|  |  |  |  |
| --- | --- | --- | --- |
| Register address | Description | Size (word) | Notes |
| 0 | Family Code | 1 | MSByte: “L” (0x4C)  LSByte: “i” (0x69) |
| 1 | Device Code MSWord | 1 | MSByte: “L”  LSByte: “F” |
| 2 | Device Code LSWord | 1 | MSByte: “T”  LSByte: "1" (0x31) |
| 3 | Firmwareversion | 1 | MSByte, LSByte: e.g. 110 = V1.10 |

1. **Reading device identification with function code 0x2B (read device identification, Modbus Application Protocol Specification V1.1b)**

|  |  |  |  |
| --- | --- | --- | --- |
| Object ID | Description | Type | Notes |
| 0x00 | Vendor Name | ASCII String |  |
| 0x01 | Product Code | ASCII String |  |
| 0x02 | Revision | ASCII String |  |

Request as one stream (device ID code = 01)

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x2B |
| MEI Type | 0x0E |
| Read Device ID code | 01 Basic Identification |
| Object ID | 0x00 |
| CRC LSB |  |
| CRC MSB |

Response stream

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x2B |
| MEI Type | 0x0E |
| Read Device ID code | 01 Basic Identification |
| Conformity Level | 0x81 |
| More Follows | 00 |
| Next Object ID | 00 |
| Number of objects | 03 |
| Object ID | 0x00 |
| Object Length | 0x08 |
| Object Value | “LIEBHERR” |
| Object ID | 0x01 |
| Object Length | 0x06 |
| Object Value | “LFT001” |
| Object ID | 0x02 |
| Object Length | 0x06 |
| Object Value | “V1.10” |
| CRC LSB |  |
| CRC MSB |

Request individual (device ID code = 04), Example for Vendor Name

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x2B |
| MEI Type | 0x0E |
| Read Device ID code | 04 Individual Identification |
| Object ID | 0x00 Vendor Name |
| CRC LSB |  |
| CRC MSB |

Response individual (device ID code = 04)

|  |  |
| --- | --- |
| slave address | 1 - 247 |
| function code | 0x2B |
| MEI Type | 0x0E |
| Read Device ID code | 04 Individual Identification |
| Conformity Level | 0x81 |
| More Follows | 00 |
| Next Object ID | 00 |
| Number of objects | 01 |
| Object ID | 0x00 |
| Object Length | 0x08 |
| Object Value | “LIEBHERR” |
| CRC LSB |  |
| CRC MSB |

* 1. **Examples for communication**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Slave Adress** | **Modbus Function Code** | **Register MSB** | **Register LSB** | **Value MSB** | **Value LSB** | **ModBus CRC LSB** | **ModBus CRC MSB** | **Remark** |
| 0x2A | 0x05 | 0x03 | 0xE9 | 0xFF | 0x00 | 0x5B | 0x91 | Switch light on adress 42 |
| 0x2A | 0x05 | 0x03 | 0xE9 | 0x00 | 0x00 | 0x1A | 0x61 | Switch light off adress 42 |
| 0x2B | 0x01 | 0x03 | 0xE9 | 0x00 | 0x01 | 0x2B | 0xB0 | Get status light adress 43 |
| 0x2B | 0x01 | 0x00 | 0xE8 | 0x00 = MSB num. of coils | 0x20 = LSB num. of coils | 0xBA | 0x2C | Read error status (-> 5.2.3), 32 coils adress 43 |
| 0x2B | 0x01 | 0x04 = num. of Bytes | 0xXX: Coils 1-8 (MSB, LSB)  0xXX: Coils 9-16 (MSB, LSB)  0xXX: Coils 17-24 (MSB, LSB)  0xXX: Coils 25-32 (MSB, LSB) | | | depends on error status | depends on error status | Response from device error status adress 43 |
| 0x2B | 0x03 | 0x00 | 0x65 | 0x00 = MSB num. of registers | 0x01 = LSB num. of registers | 0x93 | 0xDF | Get actual control value adress 43 |
| 0x2B | 0x03 | 0x02 = num. of bytes | 0xXX: MSB of Value 0xXX: LSB of Value | | | depends on value | depends on value | Response from device, get actual value |

Example: Set data / time of device adresse 42 to 14:30h (2:30 pm), 2018/12/20/

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Slave Adress** | **Modbus Function Code** | **register start adress (MSB)** | **register start adress (LSB)** | **Qty. of registers (MSB)** | **Qty. of registers (LSB)** | **Num. of Bytes** |  |
| 0x2A | 0x10 | 0x07 | 0xD0 | 0x00 | 0x03 | 0x06 | continued next line, send in one command |
| **register 1 value (hours)** | **register 1 value (minutes)** | **register 2 value (month)** | **register 2 value (day)** | **register 3 value (year, MSB)** | **register 3 value (year, LSB)** | **ModBus CRC LSB (depends on values)** | **ModBus CRC MSB (depends on values)** |
| 0x0E | 0x1E | 0x0C | 0x14 | 0x07 | 0xE2 | F5 | B7 |

All notations in hexadecimal

01 03 00 00 00 01 84 0A -> Abfrage 1 Register Vendor Code

01 2B 0E 01 00 70 77 -> Read Device Info Code 43 Basic Ident Stream

01 2B 0E 04 00 73 27 -> Read Device Info Code 43 Basic Ident HM Name

01 2B 0E 04 01 B2 E7 -> Read Device Info Code 43 Basic Ident Serial Nr

01 2B 0E 04 02 F2 E6 -> Read Device Info Code 43 Basic Ident Version

01 2B 0E 03 80 70 B7 -> Read Device Info Code 43 Extended Ident Stream

01 03 00 00 00 04 44 09 -> Read DeviceInfo Code 03 (4 Register)

01 03 00 6E 00 05 E4 14 -> Fühlewerte F1-F4 + onboard abfragen

01 03 13 BB 00 01 F0 AB -> Abfrage Türkontakt 2 installiert

01 06 13 BB 00 01 3C AB -> Türkontakt 2 = installiert setzen

01 03 13 88 00 02 40 A5 -> Abfrage Türöffnung 1 Lowword / Highword

01 03 00 64 00 01 C5 D5 -> Abfrage Sollwert

01 06 00 64 FF 42 09 D4 -> Sollwert auf -19,0°C

01 06 00 64 FF 2E 09 F9 -> Sollwert auf -21,0°C

* 1. **Variables Representation Freezer Top**

AL = Access Level

* + 1. **Main Function values**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| setpoint | RW | 3/6/16 | 0x0064 | Int16 | 10 | °C |
| actual control value | R | 3 | 0x0065 | Int16 | 10 | °C |
| actual value probe 1 | R | 3 | 0x006E | Int16 | 10 | °C |
| actual value probe 2  (not installed, reads -50°C) | R | 3 | 0x006F | Int16 | 10 | °C |
| actual value probe 3 | R | 3 | 0x0070 | Int16 | 10 | °C |
| actual value probe 4 | R | 3 | 0x0071 | Int16 | 10 | °C |
| actual value mainboard | R | 3 | 0x0072 | Int16 | 10 | °C |

Note: The setpoint has to be one of these values: -23°C (0xFF1A), -22°C (0xFF24), -21°C (0xFF2E), -20°C (0xFF38), -19°C (0xFF42), -18°C (0xFF4C). Other values will not be accepted.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| Allowed setpoints | R | 3 | 0x00C7 | Uint16 | 1 | bit encoded  HB, LB |

Allowed Setpoints:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Lowbyte (Bit 0 = LSB)** | | | | **Highbyte (Bit 8 = LSB)** | | | |
| **Bit 0** | -23°C | **Bit 4** | -19°C | **Bit 8** | -4°C | **Bit 12** | 4°C |
| **Bit 1** | -22°C | **Bit 5** | -18°C | **Bit 9** | 1°C | **Bit 13** | 5°C |
| **Bit 2** | -21°C | **Bit 6** | -6°C | **Bit 10** | 2°C | **Bit 14** | 6°C |
| **Bit 3** | -20°C | **Bit 7** | -5°C | **Bit 11** | 3°C | **Bit 15** | 7°C |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| device operation | R | 1 | 0x03E8 | Bit | - | ON/OFF |
| lighting | RW | 1/5/15 | 0x03E9 | Bit | - | ON/OFF |
| start manual defrost | RW | 1/5/15 | 0x03EA | Bit | - | ON |

* + 1. **Main Function status (1=On)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| device operation | R | 1 | 0x00C8 | Bit | - | 0, 1 |
| general alarm | R | 1 | 0x00C9 | Bit | - | 0, 1 |
| cooling | R | 1 | 0x00CA | Bit | - | 0, 1 |
| defrost | R | 1 | 0x00CB | Bit | - | 0, 1 |
| hotgas valve | R | 1 | 0x00CC | Bit | - | 0, 1 |
| lighting | R | 1 | 0x00CD | Bit | - | 0, 1 |
| fan engine 1 | R | 1 | 0x00CE | Bit | - | 0, 1 |
| fan engine 2 | R | 1 | 0x00CF | Bit | - | 0, 1 |
| railheating | R | 1 | 0x00D0 | Bit | - | 0, 1 |
| door 1 (1 = Open) | R | 1 | 0x00D1 | Bit | - | 0, 1 |
| door 2 (1 = Open) | R | 1 | 0x00D2 | Bit | - | 0, 1 |
| door 3 (1 = Open) | R | 1 | 0x00D3 | Bit | - | 0, 1 |
| door 4 (1 = Open) | R | 1 | 0x00D4 | Bit | - | 0, 1 |
| dummy | R | 1 | 0x00D5 | Bit | - | 0 |
| dummy | R | 1 | 0x00D6 | Bit | - | 0 |
| dummy | R | 1 | 0x00D7 | Bit | - | 0 |

* + 1. **Error status (1=error)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| error probe 1 | R | 1 | 0x00E8 | Bit | - | 0, 1 |
| error probe 2 | R | 1 | 0x00E9 | Bit | - | 0, 1 |
| error probe 3 | R | 1 | 0x00EA | Bit | - | 0, 1 |
| error probe 4 | R | 1 | 0x00EB | Bit | - | 0, 1 |
| low goods temperature alarm | R | 1 | 0x00EC | Bit | - | 0, 1 |
| high goods temperature alarm | R | 1 | 0x00ED | Bit | - | 0, 1 |
| malfunction hotgas valve | R | 1 | 0x00EE | Bit | - | 0, 1 |
| dummy | R | 1 | 0x00EF | Bit | - | 0 |
| memory error | R | 1 | 0x00F0 | Bit | - | 0, 1 |
| dummy | R | 1 | 0x00F1 | Bit | - | 0 |
| dummy | R | 1 | 0x00F2 | Bit | - | 0 |
| rtc error | R | 1 | 0x00F3 | Bit | - | 0, 1 |
| display error | R | 1 | 0x00F4 | Bit | - | 0, 1 |
| high temperature engine bay | R | 1 | 0x00F5 | Bit | - | 0, 1 |
| high goods temperature after power-on | R | 1 | 0x00F6 | Bit | - | 0, 1 |
| error probe mainboard | R | 1 | 0x00F7 | Bit | - | 0, 1 |
| dummy | R | 1 | 0x00F8 | Bit | - | 0 |
| dummy | R | 1 | 0x00F9 | Bit | - | 0 |
| wrong display type | R | 1 | 0x00FA | Bit | - | 0, 1 |
| malfunction goods fan 1 | R | 1 | 0x00FB | Bit | - | 0, 1 |
| malfunctio goods fan 2 | R | 1 | 0x00FC | Bit | - | 0, 1 |
| malfunction goods fan 3 | R | 1 | 0x00FD | Bit | - | 0, 1 |
| malfunction goods fan 4 | R | 1 | 0x00FE | Bit | - | 0, 1 |
| malfunction dryer fan | R | 1 | 0x00FF | Bit | - | 0, 1 |
| malfunction door 1 switch | R | 1 | 0x0100 | Bit | - | 0, 1 |
| malfunction door 2 switch | R | 1 | 0x0101 | Bit | - | 0, 1 |
| malfunction door 3 switch | R | 1 | 0x0102 | Bit | - | 0, 1 |
| malfunction door 4 switch | R | 1 | 0x0103 | Bit | - | 0, 1 |
| malfunction multiple goods fan | R | 1 | 0x0104 | Bit | - | 0, 1 |
| dummy | R | 1 | 0x0105 | Bit | - | 0 |
| dummy | R | 1 | 0x0106 | Bit | - | 0 |
| dummy | R | 1 | 0x0107 | Bit | - | 0 |

* + 1. **Date / Time**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| time hh:mm | RW | 3/6/16 | 0x07D0 | uint16 | - | MSB: Hours LSB: Minutes |
| date mm:dd | RW | 3/6/16 | 0x07D1 | uint16 | - | MSB: month LSB: day |
| year yyyy | RW | 3/6/16 | 0x07D2 | uint16 | - | 2000-2099 |

* + 1. **Compressor**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| actual speed | R | 3 | 0x0802 | uint16 | 1 | % |

* + 1. **Rail heating**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| Max. temperature limit | RW | 3/6/16 | 0x08FC | int16 | 10 | 0-15 °C |
| day of operation | RW | 3/6/16 | 0x08FD | uint16 | 1 | 0-7 |
| time of operation | RW | 3/6/16 | 0x08FE | uint16 | 1 | hh:mm |
| duration | RW | 3/6/16 | 0x08FF | uint16 | 10 | 0.1 – 24.0 h |

The start time is encoded as hh:mm. MSB represents hours, LSB represents minutes. E.g. 0x173B means 23:59, 0x0000 means 00:00, 0xFFFF deactivates the rail heating.

The days of defrost are encoded as value 0-7. 0: daily, 1 = Mondays, 2 = Tuesdays, …, 7 = Sundays.

* + 1. **Defrost**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| start manual defrost | RW | 5 | 0x03EA | Bit | - | ON |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| automatic defrost enabled | RW | 3/6/16 | 0x0A8C | uint16 | 1 | 0, 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| defrost time 1 | RW | 3/6/16 | 0x0AA0 | uint16 | 1 | hh:mm |
| day of defrost 1 | RW | 3/6/16 | 0x0AA1 | uint16 | 1 | 0-7 |
| defrost time 2 | RW | 3/6/16 | 0x0AA2 | uint16 | 1 | hh:mm |
| day of defrost 2 | RW | 3/6/16 | 0x0AA3 | uint16 | 1 | 0-7 |
| defrost time 3 | RW | 3/6/16 | 0x0AA4 | uint16 | 1 | hh:mm |
| day of defrost 3 | RW | 3/6/16 | 0x0AA5 | uint16 | 1 | 0-7 |
| defrost time 4 | RW | 3/6/16 | 0x0AA6 | uint16 | 1 | hh:mm |
| day of defrost 4 | RW | 3/6/16 | 0x0AA7 | uint16 | 1 | 0-7 |
| defrost time 5 | RW | 3/6/16 | 0x0AA8 | uint16 | 1 | hh:mm |
| day of defrost 5 | RW | 3/6/16 | 0x0AA9 | uint16 | 1 | 0-7 |
| defrost time 6 | RW | 3/6/16 | 0x0AAA | uint16 | 1 | hh:mm |
| day of defrost 6 | RW | 3/6/16 | 0x0AAB | uint16 | 1 | 0-7 |

The defrost time is encoded as hh:mm. MSB represents hours, LSB represents minutes. E.g. 0x173B means 23:59, 0x0000 means 00:00, 0xFFFF deactivates the referenced time.

The days of defrost are encoded as value 0-7. 0: daily, 1 = Mondays, 2 = Tuesdays, …, 7 = Sundays.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| defrost limit temperature | R | 3 | 0x0A96 | int16 | 10 | 0-30 °C |
| defrost limit time | R | 3 | 0x0A97 | uint16 | 1 | 10- 300 min |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| defrosting cycle time | R | 3 | 0x0A9F | uint16 | 1 | 0- 200 min |

0: defosting cycle deaktivated. Defrosting will only be started by defrost time 1-6 (default).

* + 1. **Alarm**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| high temperature difference | RW | 3/6/16 | 0x09C4 | uint16 | 10 | 0–50 K |
| low temperature difference | RW | 3/6/16 | 0x09C5 | uint16 | 10 | 0–50 K |
| alarm delay temperature standard | RW | 3/6/16 | 0x09C6 | uint16 | 1 | 0-300 min |
| alarm delay temperature after defrost / power on | RW | 3/6/16 | 0x09C7 | uint16 | 1 | 0-300 min |

* + 1. **Operating element / parameter access**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| alarm buzzer | RW | 3/6/16 | 0x0A28 | uint16 | 1 | 0, 1 |
| acoustic feedback | RW | 3/6/16 | 0x0A29 | uint16 | 1 | 0, 1 |

* + 1. **fan information**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| fan 1 goods actual setpoint | R | 3 | 0x0FA0 | uint16 | 1 | rpm |
| fan 2 goods actual setpoint | R | 3 | 0x0FA1 | uint16 | 1 | rpm |
| fan 3 goods actual setpoint | R | 3 | 0x0FA2 | uint16 | 1 | rpm |
| fan 4 goods actual setpoint | R | 3 | 0x0FA3 | uint16 | 1 | rpm |
| fan dryer actual setpoint | R | 3 | 0x0FA4 | uint16 | 1 | rpm |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| fan 1 goods actual speed | R | 3 | 0x0FAA | uint16 | 1 | rpm |
| fan 2 goods actual speed | R | 3 | 0x0FAB | uint16 | 1 | rpm |
| fan 3 goods actual speed | R | 3 | 0x0FAC | uint16 | 1 | rpm |
| fan 4 goods actual speed | R | 3 | 0x0FAD | uint16 | 1 | rpm |
| fan dryer actual speed | R | 3 | 0x0FAE | uint16 | 1 | rpm |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| fan 1 PWM duty cycle | R | 3 | 0x0FB4 | uint16 | 10 | % |
| fan 2 PWM duty cycle | R | 3 | 0x0FB5 | uint16 | 10 | % |
| fan 3 PWM duty cycle | R | 3 | 0x0FB6 | uint16 | 10 | % |
| fan 4 PWM duty cycle | R | 3 | 0x0FB7 | uint16 | 10 | % |
| fan dryer PWM duty cycle | R | 3 | 0x0FB8 | uint16 | 10 | % |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit / Value** |
| fan 1 mode of operation | R | 3 | 0x0FBE | uint16 | - | 0, 1, 2 |
| fan 2 mode of operation | R | 3 | 0x0FBF | uint16 | - | 0, 1, 2 |
| fan 3 mode of operation | R | 3 | 0x0FC0 | uint16 | - | 0, 1, 2 |
| fan 4 mode of operation | R | 3 | 0x0FC1 | uint16 | - | 0, 1, 2 |
| fan dryer mode of operation | R | 3 | 0x0FC2 | uint16 | - | 0, 1, 2 |

0: Not installed, 1: fan with speed signal, alarting active, 2: fan without speed signal, no alart

* + 1. **door information**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| status door 1 | R | 1 | 0x00D1 | Bit | - | 0, 1 |
| status door 2 | R | 1 | 0x00D2 | Bit | - | 0, 1 |
| status door 3 | R | 1 | 0x00D3 | Bit | - | 0, 1 |
| status door 4 | R | 1 | 0x00D4 | Bit | - | 0, 1 |

0: closed, 1: open

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| total number of door 1 openings | R | 3 | 0x1388 | uint32 lowword | 1 |  |
| total number of door 1 openings | R | 3 | 0x1389 | uint32 highword | 1 |  |
| total number of door 2 openings | R | 3 | 0x138A | uint32 lowword | 1 |  |
| total number of door 2 openings | R | 3 | 0x138B | uint32 highword | 1 |  |
| total number of door 3 openings | R | 3 | 0x138C | uint32 lowword | 1 |  |
| total number of door 3 openings | R | 3 | 0x138D | uint32 highword | 1 |  |
| total number of door 4 openings | R | 3 | 0x138E | uint32 lowword | 1 |  |
| total number of door 4 openings | R | 3 | 0x138F | uint32 highword | 1 |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| number of door 1 openings last 5min | R | 3 | 0x1392 | uint16 | 1 |  |
| number of door 2 openings last 5min | R | 3 | 0x1393 | uint16 | 1 |  |
| number of door 3 openings last 5min | R | 3 | 0x1394 | uint16 | 1 |  |
| number of door 4 openings last 5min | R | 3 | 0x1395 | uint16 | 1 |  |
| dummy read | R | 3 | 0x1396 | uint16 | 1 | 0 |
| dummy read | R | 3 | 0x1397 | uint16 | 1 | 0 |
| door 1 opening duration last 5 min | R | 3 | 0x1398 | uint16 | 2 | s |
| door 1 opening duration last 5 min | R | 3 | 0x1399 | uint16 | 2 | s |
| door 3 opening duration last 5 min | R | 3 | 0x139A | uint16 | 2 | s |
| door 4 opening duration last 5 min | R | 3 | 0x139B | uint16 | 2 | s |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit / value** |
| door 1 switch installed | R | 3 | 0x13BA | uint16 |  | 0 ,1 |
| door 2 switch installed | R | 3 | 0x13BB | uint16 |  | 0 ,1 |
| door 3 switch installed | R | 3 | 0x13BC | uint16 |  | 0 ,1 |
| door 4 switch installed | R | 3 | 0x13BD | uint16 |  | 0 ,1 |

* 1. **Variables Representation SMT (LBH002)**

AL = Access Level

* + 1. **Main Function values**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| setpoint | RW | 3/6/16 | 0x0064 | Int16 | 10 | °C |
| actual control value | R | 3 | 0x0065 | Int16 | 10 | °C |
| actual value probe 1 | R | 3 | 0x006E | Int16 | 10 | °C |
| actual value probe 2 | R | 3 | 0x006F | Int16 | 10 | °C |
| actual value probe 3 | R | 3 | 0x0070 | Int16 | 10 | °C |
| actual value probe 4 | R | 3 | 0x0071 | Int16 | 10 | °C |
| actual value mainboard | R | 3 | 0x0072 | Int16 | 10 | °C |

Note: The setpoint has to be one of these values: -23°C (0xFF1A), -22°C (0xFF24), -21°C (0xFF2E), -20°C (0xFF38), -19°C (0xFF42), -18°C (0xFF4C). Other values will not be accepted.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| Allowed setpoints | R | 3 | 0x00C7 | Uint16 | 1 | bit encoded  HB, LB |

Allowed Setpoints:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Lowbyte (Bit 0 = LSB)** | | | | **Highbyte (Bit 8 = LSB)** | | | |
| **Bit 0** | -23°C | **Bit 4** | -19°C | **Bit 8** | -4°C | **Bit 12** | 4°C |
| **Bit 1** | -22°C | **Bit 5** | -18°C | **Bit 9** | 1°C | **Bit 13** | 5°C |
| **Bit 2** | -21°C | **Bit 6** | -6°C | **Bit 10** | 2°C | **Bit 14** | 6°C |
| **Bit 3** | -20°C | **Bit 7** | -5°C | **Bit 11** | 3°C | **Bit 15** | 7°C |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| device operation | R | 1/ | 0x03E8 | Bit | - | ON/OFF |
| lighting | RW | 1/5/15 | 0x03E9 | Bit | - | ON/OFF |
| start manual defrost | RW | 1/5/15 | 0x03EA | Bit | - | ON |

* + 1. **Main Function status (1=On)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| device operation | R | 1 | 0x00C8 | Bit | - | 0, 1 |
| general alarm | R | 1 | 0x00C9 | Bit | - | 0, 1 |
| cooling | R | 1 | 0x00CA | Bit | - | 0, 1 |
| defrost | R | 1 | 0x00CB | Bit | - | 0, 1 |
| hotgas valve | R | 1 | 0x00CC | Bit | - | 0, 1 |
| lighting | R | 1 | 0x00CD | Bit | - | 0, 1 |
| fan engine | R | 1 | 0x00CE | Bit | - | 0, 1 |
| dummy | R | 1 | 0x00CF | Bit | - | 0, 1 |
| dummy | R | 1 | 0x00D0 | Bit | - | 0 |
| dummy | R | 1 | 0x00D1 | Bit | - | 0 |
| dummy | R | 1 | 0x00D2 | Bit | - | 0 |
| dummy | R | 1 | 0x00D3 | Bit | - | 0 |
| dummy | R | 1 | 0x00D4 | Bit | - | 0 |
| dummy | R | 1 | 0x00D5 | Bit | - | 0 |
| dummy | R | 1 | 0x00D6 | Bit | - | 0 |
| dummy | R | 1 | 0x00D7 | Bit | - | 0 |

* + 1. **Error status (1=error)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| error probe 1 | R | 1 | 0x00E8 | Bit | - | 0, 1 |
| error probe 2 | R | 1 | 0x00E9 | Bit | - | 0, 1 |
| error probe 3 | R | 1 | 0x00EA | Bit | - | 0, 1 |
| error probe 4 | R | 1 | 0x00EB | Bit | - | 0, 1 |
| low goods temperature alarm | R | 1 | 0x00EC | Bit | - | 0, 1 |
| high goods temperature alarm | R | 1 | 0x00ED | Bit | - | 0, 1 |
| malfunction hotgas valve | R | 1 | 0x00EE | Bit | - | 0, 1 |
| dummy | R | 1 | 0x00EF | Bit | - | 0 |
| memory error | R | 1 | 0x00F0 | Bit | - | 0, 1 |
| dummy | R | 1 | 0x00F1 | Bit | - | 0 |
| dummy | R | 1 | 0x00F2 | Bit | - | 0 |
| rtc error | R | 1 | 0x00F3 | Bit | - | 0, 1 |
| display error | R | 1 | 0x00F4 | Bit | - | 0, 1 |
| high temperature engine bay | R | 1 | 0x00F5 | Bit | - | 0, 1 |
| high goods temperature after power-on | R | 1 | 0x00F6 | Bit | - | 0, 1 |
| error probe mainboard | R | 1 | 0x00F7 | Bit | - | 0, 1 |
| dummy | R | 1 | 0x00F8 | Bit | - | 0 |
| dummy | R | 1 | 0x00F9 | Bit | - | 0 |
| wrong display type | R | 1 | 0x00FA | Bit | - | 0, 1 |
| dummy | R | 1 | 0x00FB | Bit | - | 0 |
| dummy | R | 1 | 0x00FC | Bit | - | 0 |
| dummy | R | 1 | 0x00FD | Bit | - | 0 |
| dummy | R | 1 | 0x00FE | Bit | - | 0 |
| dummy | R | 1 | 0x00FF | Bit | - | 0 |
| dummy | R | 1 | 0x0100 | Bit | - | 0 |
| dummy | R | 1 | 0x0101 | Bit | - | 0 |
| dummy | R | 1 | 0x0102 | Bit | - | 0 |
| dummy | R | 1 | 0x0103 | Bit | - | 0 |
| dummy | R | 1 | 0x0104 | Bit | - | 0 |
| dummy | R | 1 | 0x0105 | Bit | - | 0 |
| dummy | R | 1 | 0x0106 | Bit | - | 0 |
| dummy | R | 1 | 0x0107 | Bit | - | 0 |

* + 1. **Date / Time**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| time hh:mm | RW | 3/6/16 | 0x07D0 | uint16 | - | MSB: Hours LSB: Minutes |
| date mm:dd | RW | 3/6/16 | 0x07D1 | uint16 | - | MSB: month LSB: day |
| year yyyy | RW | 3/6/16 | 0x07D2 | uint16 | - | 2000-2099 |

* + 1. **Compressor**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| actual speed | R | 3 | 0x0802 | uint16 | 1 | % |

* + 1. **Defrost**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| start manual defrost | RW | 5 | 0x03EA | Bit | - | ON |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| automatic defrost enabled | RW | 3/6/16 | 0x0A8C | uint16 | 1 | 0, 1 |

* + 1. **Defrost freezing mode**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| defrost time 1 | RW | 3/6/16 | 0x0AA0 | uint16 | 1 | hh:mm |
| day of defrost 1 | RW | 3/6/16 | 0x0AA1 | uint16 | 1 | 0-7 |
| defrost time 2 | RW | 3/6/16 | 0x0AA2 | uint16 | 1 | hh:mm |
| day of defrost 2 | RW | 3/6/16 | 0x0AA3 | uint16 | 1 | 0-7 |
| defrost time 3 | RW | 3/6/16 | 0x0AA4 | uint16 | 1 | hh:mm |
| day of defrost 3 | RW | 3/6/16 | 0x0AA5 | uint16 | 1 | 0-7 |
| defrost time 4 | RW | 3/6/16 | 0x0AA6 | uint16 | 1 | hh:mm |
| day of defrost 4 | RW | 3/6/16 | 0x0AA7 | uint16 | 1 | 0-7 |
| defrost time 5 | RW | 3/6/16 | 0x0AA8 | uint16 | 1 | hh:mm |
| day of defrost 5 | RW | 3/6/16 | 0x0AA9 | uint16 | 1 | 0-7 |
| defrost time 6 | RW | 3/6/16 | 0x0AAA | uint16 | 1 | hh:mm |
| day of defrost 6 | RW | 3/6/16 | 0x0AAB | uint16 | 1 | 0-7 |

The defrost time is encoded as hh:mm. MSB represents hours, LSB represents minutes. E.g. 0x173B means 23:59, 0x0000 means 00:00, 0xFFFF deactivates the referenced time.

The days of defrost are encoded as value 0-7. 0: daily, 1 = Mondays, 2 = Tuesdays, …, 7 = Sundays.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| defrost limit temperature | R | 3 | 0x0A96 | uint16 | 10 | 0-30 °C |
| defrost limit time | R | 3 | 0x0A97 | uint16 | 1 | 10- 300 min |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| defrosting cycle time | R | 3 | 0x0A9F | uint16 | 1 | 0- 200 min |

0: defosting cycle deaktivated. Defrosting will only be started by defrost time 1-6 (default).

* + 1. **Defrost cooling mode**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| defrost time 1 | RW | 3/6/16 | 0x0ABE | uint16 | 1 | hh:mm |
| day of defrost 1 | RW | 3/6/16 | 0x0ABF | uint16 | 1 | 0-7 |
| defrost time 2 | RW | 3/6/16 | 0x0AC0 | uint16 | 1 | hh:mm |
| day of defrost 2 | RW | 3/6/16 | 0x0AC1 | uint16 | 1 | 0-7 |
| defrost time 3 | RW | 3/6/16 | 0x0AC2 | uint16 | 1 | hh:mm |
| day of defrost 3 | RW | 3/6/16 | 0x0AC3 | uint16 | 1 | 0-7 |
| defrost time 4 | RW | 3/6/16 | 0x0AC4 | uint16 | 1 | hh:mm |
| day of defrost 4 | RW | 3/6/16 | 0x0AC5 | uint16 | 1 | 0-7 |
| defrost time 5 | RW | 3/6/16 | 0x0AC6 | uint16 | 1 | hh:mm |
| day of defrost 5 | RW | 3/6/16 | 0x0AC7 | uint16 | 1 | 0-7 |
| defrost time 6 | RW | 3/6/16 | 0x0AC8 | uint16 | 1 | hh:mm |
| day of defrost 6 | RW | 3/6/16 | 0x0AC9 | uint16 | 1 | 0-7 |

The defrost time is encoded as hh:mm. MSB represents hours, LSB represents minutes. E.g. 0x173B means 23:59, 0x0000 means 00:00, 0xFFFF deactivates the referenced time.

The days of defrost are encoded as value 0-7. 0: daily, 1 = Mondays, 2 = Tuesdays, …, 7 = Sundays.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| defrost limit temperature | R | 3 | 0x0AB4 | uint16 | 10 | 0-30 °C |
| defrost limit time | R | 3 | 0x0AB5 | uint16 | 1 | 10- 300 min |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| defrosting cycle time | R | 3 | 0x0ABD | uint16 | 1 | 0- 200 min |

0: defosting cycle deaktivated. Defrosting will only be started by defrost time 1-6 (default).

* + 1. **Alarm**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| high temperature difference | RW | 3/6/16 | 0x09C4 | uint16 | 10 | 0–50 K |
| low temperature difference | RW | 3/6/16 | 0x09C5 | uint16 | 10 | 0–50 K |
| alarm delay temperature standard | RW | 3/6/16 | 0x09C6 | uint16 | 1 | 0-300 min |
| alarm delay temperature after defrost / power on | RW | 3/6/16 | 0x09C7 | uint16 | 1 | 0-300 min |

* + 1. **Operating element / parameter access**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Label** | **R/W** | **Function code** | **Address HEX** | **Data Type** | **Gain** | **Unit** |
| alarm buzzer | RW | 3/6/16 | 0x0A28 | uint16 | 1 | 0, 1 |
| acoustic feedback | RW | 3/6/16 | 0x0A29 | uint16 | 1 | 0, 1 |