



OEM-DES-R831-USB-18/3

OEM-DES-M900-18/3

**13.56 MHz OEM RFID Device
Communication Protocol ISO18000-3**

iDTRONIC GmbH
Ludwig-Reichling-Straße 4
67059 Ludwigshafen
Germany/Deutschland

Issue 0.5
– 15. April 2019 –

Phone: +49 621 6690094-0
Fax: +49 621 6690094-9
E-Mail: info@idtronic.de
Web: idtronic.de

Subject to alteration without prior notice.
© Copyright iDTRONIC GmbH 2019
Printed in Germany

Contents

1	Description	4
1.1	Reference Documents	4
1.2	Telegram Frame	4
1.3	General dialog structure	4
1.3.1	Successful command	4
1.3.2	Unsuccessful command	5
1.3.3	Answer with an acknowledge: (power_off, idle_mode, power_down_mode)	5
1.4	Important Note	5
2	UART Interface	5
2.1	General description	5
3	Command Set	5
4	Error Code List	6
5	Commands	6
5.1	General Definitions	6
5.2	ISO18000P3M3_INVENTORY (0xB1)	6
5.3	ISO18000P3M3_ACK (0xB2)	7
5.4	ISO18000P3M3_REQRN (0xB3)	7
5.5	ISO18000P3M3_READ(0xB4)	8
5.6	ISO18000P3M3_WRITE(0xB5)	9
5.7	ISO18000P3M3_KILL(0xB6)	10
5.8	ISO18000P3M3_LOCK(0xB7)	10
5.9	ISO18000P3M3_ACCESS (0xB8)	11
5.10	ISO18000P3M3_BLOCKWRITE (0xB9)	11
5.11	ISO18000P3M3_BLOCKERASE (0xBA)	12
5.12	ISO18000P3M3_BLOCKPERMALOCK (0xBB)	12
5.13	ISO18000P3M3_SETHANDLE (0xBC)	13
6	Revision History	14

1 Description

1.1 Reference Documents

This device uses the following commands on top of the DESFire communication protocol. So, in order to gain full access to the device, please consult this communication protocol description:

Command Protocol and API Description: OEM-DES devices Communication Protocol_x.yy_EN.pdf

For test operation:

Manual of Test/Demo Software: OEM-DES devices Test Software Manual_x.y_EN.pdf

1.2 Telegram Frame

The communication between the host controller and the reader obeys to a protocol named PARA. This protocol encapsulates the useful data of a message in an invariant frame structure and defines a dialog structure of messages exchanges.

Frame structure

Data is exchanged between the host controller and the reader in blocks, each made up of binary characters on one byte:

4 bytes	0 to 506 bytes	1 byte
Header characters	Data	XOR
	Information field	Checksum

4 bytes header byte includes:*

1 st byte	2 nd byte	3 rd byte	4 th byte
A 1 A 1 0 0 0 0			
	Data length to be transmitted excluding header and XOR	Command byte	

Remark: A = 0, ACKnowledge of the frame (1st byte = 50)

A = 1, NACK of the frame (message with a status error, 1st byte = F0)

XOR byte: XOR sum over all bytes including the 0x50 Start of Telegram

1.3 General dialog structure

The host controller is the master for the transmission; each command from the master is followed by an answer from the reader including the same command byte as the input command.

However, in some cases (card insertion or extraction, time out detection on Rx line or an automatic emergency deactivation of the card) the reader is able to initiate an exchange.

1.3.1 Successful command

System to Reader

50	XX XX	YY	Nnnnnnnnnnnnnnnnnnnnn	ZZ
ACK	Length	CMD	Data	XOR

Reader to System:

50	UU UU	YY	mmmmmmmmmmmmmmmmmmmm	ZZ
ACK	Length	CMD	Data	XOR

The same command byte YY is returned in the answer from the reader.

1.3.2 Unsuccessful command

System to Reader

50	XX XX	YY	nnnnnnnnnnnnnnnnnnnnnn	ZZ
ACK	Length	CMD	Data	XOR

Reader to System

F0	UU UU	YY	SS	TT
ACK	Length	CMD	Status	XOR

In that case, the status contains the error code information (see error list).

1.3.3 Answer with an acknowledge: (power_off, idle_mode, power_down_mode)

System to Reader (example: PiccHalt)

50	00 00	14	44
ACK	Length	CMD	XOR

Reader to System:

50	00 00	14	44
ACK	Length	CMD	XOR

In the case where the answer is an acknowledge of the command, the reader sends back a frame with the same content of the command.

1.4 Important Note

Depending on the type of the RFID tags, some of these commands are not supported.

2 UART Interface

2.1 General description

The serial interface between the Reader and the host controller is a full duplex interface using the two lines RX and TX.

RX is used to receive data from the host controller;

TX is used to send data to the host controller.

No flow control or supplementary line is used (no hand check).

The serial data format used is:

1	Start bit
8	Data bit
1	Stop bit, no parity
Default baudrate	152000 bps

3 Command Set

The following command bytes are available (listed in numerical order):

Command	Code
ISO18000P3M3_INVENTORY	0xB1
ISO18000P3M3_ACK	0xB2
ISO18000P3M3_REQRN	0xB3

ISO18000P3M3_READ	0xB4
ISO18000P3M3_WRITE	0xB5
ISO18000P3M3_KILL	0xB6
ISO18000P3M3_LOCK	0xB7
ISO18000P3M3_ACCESS	0xB8
ISO18000P3M3_BLOCKWRITE	0xB9
ISO18000P3M3_BLOCKERASE	0xBA
ISO18000P3M3_BLOCKPERMALOCK	0xBB
ISO18000P3M3_SETHANDLE	0xBC

4 Error Code List

Status Code	Code
I18000P3M3_ERR_OTHER	0x80
I18000P3M3_ERR_MEMORY_OVERRUN	0x81
I18000P3M3_ERR_MEMORY_LOCKED	0x82
I18000P3M3_ERR_INSUFFICIENT_POWER	0x83
I18000P3M3_ERR_NON_SPECIFIC	0x84
Verification of input Package checksum has failed.	0xF1

5 Commands

5.1 General Definitions

Memory Banks

```
#define PHAL_I18000P3M3_MEMBANK_RESERVED 0x00U /** < Reserved Memory Bank. */
#define PHAL_I18000P3M3_MEMBANK_UII 0x01U /** < UII Memory Bank. */
#define PHAL_I18000P3M3_MEMBANK_TID 0x02U /** < TID Memory Bank. */
#define PHAL_I18000P3M3_MEMBANK_USER 0x03U /** < User Memory Bank. */
```

5.2 ISO18000P3M3_INVENTORY (0xB1)

Notes

```
Status_t ISO18000P3M3_Activate(
    uint8_t *rebInforLen,
    uint8_t *rebInfor,
    uint8_t *pbM,uint8_t *pbDr
);
```

Telegram Example

Command from PC/PLC to RFID: 50 00 00 B1 E1

Reply form RFID to PC/PLC: 50 00 0E B1 03 01 00 00 00 00 00 00 48 04 27 C4 5D 5B 44

The Bytes in Detail:

```
50          = Start of telegram
00 0E       = 14 Bytes of payload between command code and CRC
B1          = Command code
03          = Modulation
01          = Link frequency
00 00 00 00
```

```

00 00 48 04
27 C4 5D 5B = 12 Bytes of EPC
44          = CRC

```

5.3 ISO18000P3M3_ACK (0xB2)

Acknowledge a single tag.

Notes

```

Status_t ISO18000p3m3_Ack(
    uint8_t * pRxBuffer,    /**< [Out] Pointer to Tag data and, if required, PacketCRC. */
    uint16_t * pRxLength    /**< [Out] Tag response length in bits. */
);

```

Telegram Example

Command from PC/PLC to RFID: 50 00 00 B2 E2

Reply form RFID to PC/PLC: 50 00 0E B2 30 00 00 00 00 00 00 00 48 04 27 C4 5D 5B 75

The Bytes in Detail:

```

50          = Start of telegram
00 0E0E     = 14 Bytes of payload between command code and CRC
B2          = Command code
30 00       = Protocol Control (PC)
00 00 00 00
00 00 48 04
27 C4 5D 5B = 12 Bytes of EPC
75          = CRC

```

5.4 ISO18000P3M3_REQRN (0xB3)

Instruct a tag to loadmodulate a new RN16 or Handle.

Notes

```

Status_t ISO18000p3m3_ReqRn(
    uint8_t ** pRxBuffer    /**< [Out] New RN16 or handle. */
);

```

Telegram Example

Command from PC/PLC to RFID: 50 00 00 B3 E3

Reply form RFID to PC/PLC: 50 00 02 B3 00 00 44

The Bytes in Detail:

```

50          = Start of telegram
00 02       = 2 Bytes of payload between command code and CRC
B3          = Command code
00 00       = New RN16 or handle
44          = CRC

```

5.5 ISO18000P3M3_READ(0xB4)

Description

Read part or all of a tag Reserved, UII, TID, or User memory.

bWordPtrLength depends on the TAG memory size. For TAGs with 8 bits memory, bWordPtrLength should be always '0'. If we make 'bWordPtrLength' =1 (16bits) or higher for 8 bits memory TAGs then this function returns MEMORY_OVERRUN error.

Notes

```
Status_t ISO18000p3m3_Read(
    uint8_t bMemBank,          /**< [In] Memory bank where the read shall be performed. */
    uint8_t * pWordPtr,        /**< [In] Starting read address. */
    uint8_t bWordPtrLength,    /**< [In] Length of the pointer in bytes;
                                0 -> 1byte, 1 -> 2bytes, 2 -> 3bytes or 3 -> 4bytes. */
    uint8_t bWordCount,        /**< [In] Number of bytes to read. */
    uint8_t * pRxBuffer,        /**< [Out] Header and requested memory words. */
    uint16_t * pRxLength        /**< [Out] Number of received bits. */
);
```

Telegram Example 1

Command from PC/PLC to RFID: 50 00 05 B4 03 00 00 00 02 E0 //USER(03) 2*2 bytes

The Bytes in Detail:

50	= Start of telegram
00 05	= 5 Bytes of payload between command code an CRC
B4	= Command code
03	= Memory bank
00 00	= Start address in blocks, LSB first!, only even numbers!
00 02	= Number of blocks to read, 1 block = 2 Bytes
E0	= CRC

Reply form RFID to PC/PLC: 50 00 04 B4 00 00 00 00 E0

The Bytes in Detail:

50	= Start of telegram
00 04	= 4 Bytes of payload between command code and CRC
B4	= Command code
00 00 00 00	= 4 Bytes of data
E0	= CRC

Telegram Example 2

Command from PC/PLC to RFID: 50 00 05 B4 02 00 00 00 04 E7 //TID(02) 4*2 bytes

Reply form RFID to PC/PLC: 50 00 08 B4 E2 00 68 03 00 00 48 04 29

Telegram Example 3

Command from PC/PLC to RFID: 50 00 05 B4 01 00 00 00 08 E8 //EPC(01) 0x08*2 bytes = 16bytes

Reply form RFID to PC/PLC: 50 00 10 B4 EC D0 30 00 00 00 00 00 00 00 48 04 27 C4 5D 5B 51

Telegram Example 4

Command from PC/PLC to RFID: 50 00 05 B4 02 02 00 00 04 E5 Read 4 blocks, start at block 2

Reply form RFID to PC/PLC: 50 00 08 B4 00 00 48 01 38 F2 55 2D 17

Telegram Example 5

Command from PC/PLC to RFID: 50 00 05 B4 02 04 00 00 02 E5

Read 2 blocks, start at block 4

Reply form RFID to PC/PLC: 50 00 04 B4 38 F2 55 2D 52

5.6 ISO18000P3M3_WRITE(0xB5)

Description

Write a word in a tag Reserved, UII, TID, or User memory.

bWordPtrLength depends on the TAG memory size. For TAGs with 8 bits memory, bWordPtrLength should be always '0'. If we make 'bWordPtrLength' =1 (16bits) or higher for 8 bits memory TAGs then this function returns MEMORY_OVERRUN error. This is an expected behaviour.

bOption can be one of: #PHAL_I18000P3M3_AC_NO_COVER_CODING
#PHAL_I18000P3M3_AC_USE_COVER_CODING

Notes

```
Status_t ISO18000p3m3_Write(
    uint8_t bOption,          /**< [In] Option parameter. */
    uint8_t bMemBank,         /**< [In] Memory bank where the write shall be performed. */
    uint8_t * pWordPtr,       /**< [In] Starting write address. */
    uint8_t bWordPtrLength,   /**< [In] Length of the pointer in bytes;
                                0 -> 1byte, 1 -> 2bytes, 2 -> 3bytes or 3 -> 4bytes. */
    uint8_t * pData           /**< [In] Word to write; uint8_t[2]. */
);
```

Telegram Example

Command from PC/PLC to RFID: 50 00 07 B5 01 03 00 00 00 11 22 D3

//USER(03) 1*2 bytes

The Bytes in Detail:

50	= Start of Telegram
00 07	= 7 Bytes of payload between command code and CRC
B5	= Command code
01	= Option Byte
03	= Memory bank
00 00	= Start address in blocks, LSB first!
00	= Number of blocks to write, 0 = 1, 1 = 2 (not supported by I-Code ILT-M)
11 22	= Data
D3	= CRC

Reply form RFID to PC/PLC: 50 00 00 B5 E5

Notes

The I-Code ILT-M supports to write only 1 block at once using this command.

```
#define PHAL_I18000P3M3_AC_NO_COVER_CODING 0x00U /**< Use cover coding
                                                    to diversify passwords. */
#define PHAL_I18000P3M3_AC_USE_COVER_CODING 0x01U /**< Do not use cover coding,
                                                    send plain passwords. */
```

5.7 ISO18000P3M3_KILL(0xB6)

Description

Render a tag killed or recommissioned as appropriate.

bOption can be one of: #PHAL_I18000P3M3_AC_NO_COVER_CODING
 #PHAL_I18000P3M3_AC_USE_COVER_CODING

Notes

```
Status_t ISO18000p3m3_Kill(
    uint8_t bOption,      /**< [In] Option parameter. */
    uint8_t * pPassword,  /**< [In] Full kill password; uint8_t[4] */
    uint8_t bRecom        /**< [In] Recommissioning bits. */
);
```

Telegram Example

Command from PC/PLC to RFID: 50 00 04 B6 01 00 00 00 00 00 E2 //send plain passwords(01) 4 bytes pPassword

The Bytes in Detail:

50	= Start of telegram
00 04	= 4 Bytes of payload between command code and CRC
B6	= Command code
01	= Option parameter
00 00 00 00	= Kill password
00	= Recommissioning bits
E2	= CRC

Reply form RFID to PC/PLC: 50 00 00 B6 E6

Notes

```
#define PHAL_I18000P3M3_AC_NO_COVER_CODING 0x00U /**< Use cover coding
                                                to diversify passwords. */
#define PHAL_I18000P3M3_AC_USE_COVER_CODING 0x01U /**< Do not use cover coding,
                                                send plain passwords. */
```

5.8 ISO18000P3M3_LOCK(0xB7)

Description

Lock or Permalock individual passwords and memory banks.

Notes

```
Status_t ISO18000p3m3_Lock (
    uint8_t * pMask,      /**< [In] 10bit Action Field Mask; uint8_t[2]. */
    uint8_t * pAction     /**< [In] 10bit Action Field; uint8_t[2]. */
);
```

Telegram Example

Command from PC/PLC to RFID: 50 00 04 B7 00 00 00 00 00 E3

The Bytes in Detail:

50	= Start of telegram
----	---------------------

00 04 = 4 Bytes of payload between command code and CRC
 B7 = Command code
 00 00 = pMask
 00 00 = pAction
 E3 = CRC

Reply form RFID to PC/PLC: 50 00 00 B7 E7

5.9 ISO18000P3M3_ACCESS (0xB8)

Cause a tag with a non-zero-valued access password to transition from the open to the secured state.

bOption can be one of: #PHAL_I18000P3M3_AC_NO_COVER_CODING
 #PHAL_I18000P3M3_AC_USE_COVER_CODING

Notes

```

Status_t ISO18000p3m3_Access(
    uint8_t bOption,          /**< [In] Option parameter. */
    uint8_t * pPassword      /**< [In] Full access password; uint8_t[4] */
);
  
```

Telegram Example

Command from PC/PLC to RFID: 50 00 05 B8 01 00 00 00 00 EC

The Bytes in Detail:

50 = Start of telegram
 00 05 = 5 Bytes of payload between command code and CRC
 B8 = Command code
 01 = Option parameter
 00 00 00 00 = Password
 EC = CRC

Reply form RFID to PC/PLC: 50 00 00 B8 E8

5.10 ISO18000P3M3_BLOCKWRITE (0xB9)

Write multiple words in a tag Reserved, UII, TID, or User memory.

Return value: Status code #PH_ERR_SUCCESS = Operation successful.
 Other Depending on implementation and underlying component.

Notes

The I-Code ILT-M supports to write only 2 block at once using this command.

```

Status_t ISO18000p3m3_BlockWrite(
    uint8_t bMemBank,          /**< [In] Memory bank where the write shall be performed. */
    uint8_t * pWordPtr,        /**< [In] Starting write address. */
    uint8_t bWordPtrLength,    /**< [In] Length of the pointer in bytes; 0,1,2,3 */
    uint8_t bWordCount,        /**< [In] Number of blocks to write. */
    uint8_t * pData            /**< [In] Words to write; uint8_t[2U * \c bWordCount]. */
);
  
```

Telegram Example

Command from PC/PLC to RFID: 50 00 09 B9 03 00 00 00 02 11 22 33 44 A5 //USER(03) 2*2 bytes

The Bytes in Detail:

50 = Start of telegram
 00 09 = 9 Bytes of payload between command code and CRC
 B9 = Command code
 03 = Memory bank, 0x03 = USER
 00 00 = Start address
 00 = Length of pointer
 02 = Number of blocks
 11 22 33 44 = Data to be written, 4 Bytes = 2 blocks
 A5 = CRC

Reply form RFID to PC/PLC: 50 00 00 B9 E9

5.11 ISO18000P3M3_BLOCKERASE (0xBA)

Erase multiple words in a tag Reserved, Ull, TID, or User memory.

Notes

```
Status_t ISO18000p3m3_BlockErase(
    uint8_t bMemBank,          /**< [In] Memory bank where the erase shall be performed. */
    uint8_t * pWordPtr,        /**< [In] Starting erase address. */
    uint8_t bWordPtrLength,    /**< [In] Length of the pointer in bytes; 0,1,2,3. */
    uint8_t bWordCount         /**< [In] Number of blocks to erase. */
);
```

Telegram Example

Command from PC/PLC to RFID: 50 00 05 BA 03 00 00 00 04 E8 //USER(03)

The Bytes in Detail:

50 = Start of telegram
 00 05 = 5 Bytes of payload between command code and CRC
 BA = Command code
 03 = Memory bank, 0x03 = USER
 00 00 = Start address
 00 = Length of the pointer
 04 = Number of blocks to erase
 E8 = CRC

Reply form RFID to PC/PLC: 50 00 00 BA EA

5.12 ISO18000P3M3_BLOCKPERMALOCK (0xBB)

Erase multiple words in a tag Reserved, Ull, TID, or User memory.

Notes

```
Status_t ISO18000p3m3_BlockPermaLock(
    uint8_t bRFU,              /**< [In] RFU, shall be set to \c 0. */
```

```

uint8_t bReadLock,          /**< [In] Whether the permalock states shall be retrieved
                             (\c 0) or the blocks shall be permalocked (\c 1). */
uint8_t bMemBank,          /**< [In] Memory bank where the erase shall be performed. */
uint8_t * pBlockPtr,       /**< [In] Starting erase adress. */
uint8_t bBlockPtrLength,   /**< [In] Length of the pointer in bytes; 0,1,2,3. */
uint8_t bBlockRange,       /**< [In] Mask range, specified in units of 16 blocks. */
uint8_t * pMask,           /**< [In] Specifies which memory blocks a tag permalocks;
                             uint8_t[2U * \c bBlockRange]
                             Ignored if \c bReadLock is \c 0 */
uint8_t *pRxBuffer,        /**< [Out] Header and Permalock bits
                             if \c bReadLock is \c 0 or NULL otherwise. */
uint16_t * pRxLength       /**< [Out] Number of received bytes
                             if \c bReadLock is \c 0. */
);

```

Telegram Example

Command from PC/PLC to RFID: 50 00 09 BB 00 00 00 00 00 02 00 00 EE

The Bytes in Detail:

50	= Start of telegram
00 09	= 9 Bytes of payload between command code and CRC
B8	= Command code
01	= Option parameter
00 00 00 00	= Password
EC	= CRC

Reply form RFID to PC/PLC: 50 00 0x BB yy EB

5.13 ISO18000P3M3_SETHANDLE (0xBC)

Set the Handle into the internal data structure.

Notes

```

Status_t ISO18000p3m3_SetHandle(
    uint8_t* pHandle          /**< [In] Handle to the Card. */
);

```

Telegram Example

Command from PC/PLC to RFID: 50 00 02 BC 00 00 EE

The Bytes in Detail:

50	= Start of telegram
00 02	= 2 Bytes of payload between command code and CRC
BC	= Command code
00 00	= Handle word
EC	= CRC

Reply form RFID to PC/PLC: 50 00 00 BC 44

6 Revision History

2019-03-07	0.1	Initial release.
2019-03-12	0.2	Minor changes, reference documents added
2019-03-14	0.3	Commands added
2019-03-15	0.4	Redundant information removed
2019-04-15	0.5	Read/write command writes blocks instead of Bytes. LSB first notation of start address added. Information on limitations of the I-Code ILT-M added.