Ingenieurbüro für Echtzeitprogrammierung

PK-DP

Document Rev. 2.7 from 07.02.2006

Hardware Rev. 1.4 from 26.04.2004

Table of contents

Revision list:

Rev.	Date	Na.	Change
1.0	20.09.2001	Ko	Production
1.1	12.04.2002	Ko	div. Additions
1.2	03.07.2003	Ko	Error corrects
1.3	05.09.2003	Ko	LED's supplements (chapter 4.4), Description of the ASCII-drivers (chapter 5)
1.4	19.09.2003	Ko	Supplements communication references (chapter 6.4), Firmware update (chapter 7)
1.5	29.09.2003	Ko	Div. Additions
1.6	10.10.2003	Ko	Adjustment on firmware 2.0 (support COM1/2)
1.7	14.10.2003	Ko	Parameter data length on 32 shortened, S5 cannot more!
1.8	15.10.2003	Ko	Update is always made by COM 2
1.9	31.03.2004	Ko	Problem treatment extends
2.0	15.04.2004	Ko	Description for the receiving of data improves
2.1	12.05.2004	Ko	S-command at the update and History supplements
2.2	25.05.2004	Ko	FAQ extends
2.3	28.05.2004	Ko	Description of parity was unclear (even and odd)
2.4	23.07.2004	Ko	Adjustment for hardware Rev. 1.4
2.5	30.09.2004	Ko	PK-DP transmits only after "speech" data
2.6	01.06.2005	Ko	Typing error corrects
2.7	30.08.2005	Ko	English version

1 General references

1.1 Handling

1. First please read carefully this documentation before you unpack the hardware and switch on. You save time and avoid problems.

2. Please consider the precautionary measures during the handling of electrostatically endangered hardware.

3. If the hardware contains batteries, do not put it on electrically conductive underlays. The battery could be shortet and caused damage.

4. Please make sure that the specified temperature range will not leave.

1.2 Installation

1. Examine whether all Jumpers are set according to your application.

2. Switch the voltage supply off, before you make a connection.

3. If you are safe that all connections are correctly installed switch on the supply voltage.

1.3 Explanation

We reserve ourselves the right to make changes, which serve an improvement of the circuit or the product, without special references. Despite careful control no adhesion can be taken over for the correctness of the data, connection diagrams, programs given here and descriptions. The suitability of the product for a certain targeted application is not assured.

1.4 Repairs

If the product should be defective, then send it back freely in suitable packing with the following description to us:

• Error description

• Did the error arise only under certain conditions?

• What was attached?

• How did the attached signals look?

• Warranty repair or not?

2 Technical data

2.1 Ambient conditions

Ambient temperature (normal operation)	0-50 °C
Ambient temperature (storage)	-20-85 °C
rel. humidity	max. 95%, non-condensing
Height	-300m to +3000m

2.2 Connections

Supply voltage:	12-36 V DC, max. 0.2 A, typ.: 24 Volt/0,1 A
Profibus DP Slave:	all baud rates up to 12 MBaud, automatic recognition
	Variable I/O range from 2-80 data items
	Over GSD file freely configurable ASCII-driver or programming of own drivers for special protocols possible
	Configuration by the DP master, no extra configuration program necessary
serial connections:	1 serial RS-232 interface 5-wire, up to 57600 KBaud
	1 serial RS-232 interface 5-wire alternatively also as RS-485 or RS-422, up to 76800 KBaud
Housing:	112x112x50 mm, for DIN rail assembly

3 Starting apparatus

The PK-DP interface converter makes communication of a PROFIBUS-DP-Master with one or two field devices with serial interface possible. The data between DP master and PK-DP exchanged in the form of telegrams resp. telegram fragments over a data channel. The size of the data channel can be varied by configuration and be adapted to the telegram length and the available I/O range of the DP master. The data channel consists of a sending and a receiving channel.

The PK-DP can be easily installed, configured and parameterized. An additional configuration program is not necessary. The configuration is exclusively made by the PROFIBUS-DP master.

Port COM 1 for seriel device

Port COM 2 for seriel device

Powersupply indication LED

Profibus connector

Powersupply 24 Volt DC

PE GND +24V

Profibus Address hexadecimal

10er 1er digit

Operatingstate indication
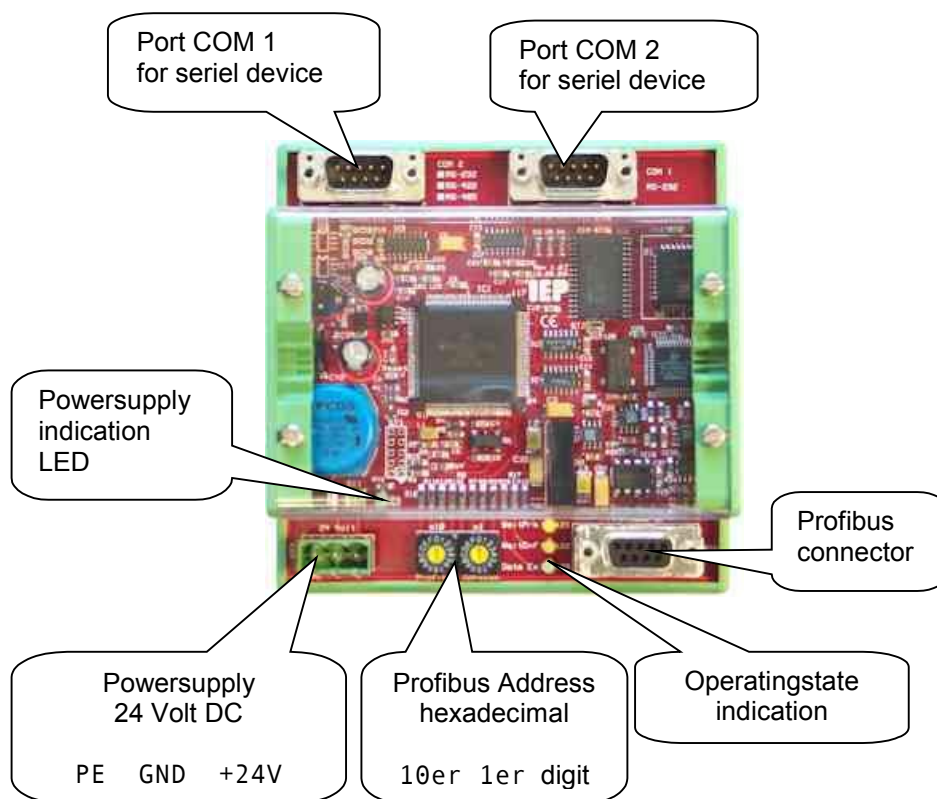
3.1 Housing

The PK-DP has a 112 x 112 x 50 mm large housing. The housing can be snapped on commercial DIN rails.

3.2 Installation

The PK-DP is intended for the installation in switchgear cabinets or similar EMV close housings. The wiring is to be accomplished EMC compatible with shielded cables. Use as bus cables only special bifilar, twisted and shielded PROFIBUS cable. Only then the high transmission rates are guaranteed. Data to cable lengths with the PROFIBUS you find in

your DP master manual. Use terminal resistances, if the PK-DP at the beginning or at the end of the PROFIBUS line is. If they are missing, you can have disturbances in the data communication as well as damage on other bus participants. Use only PROFIBUS plugs, where the terminating resistor is already integrated.

3.3 Voltage supply

Attach the cables for voltage supply, mass and ground wire to the characterized connecting terminals. Consider the polarity of voltage supply. The PK-DP is protected against reverse polarity, but must be attached however with correctly polarized for a troublefree oparation.

The permissible voltage range amounts from 12-36 V DC, max. 0.2 A. At 24 V the PK-DP needs circa 0.1 A. Pay attention to a good grounding of the PK-DP.

3.4 Setting up the Profibus address

The address of the PK-DP can be adjusted over the two HEX switches. The right switch sets up the ones column, the left the tens column. **The address switch works hexadecimal.**

The attitude of e.g. left=4 and right=2 results in the hexadecimal address 0x42, converted 66 in decimal notation.

The permissible address range amounts from 1 to 126 (0x7E). From practical considerations only addresses between 1 to 123 (0x7B) should be used. The address 0 is used internally for the software update.

Please note that the adjusted address is read in only when the PK-DP is starting. If you have change the address, you must separate the PK-DP from voltage supply, in order to read the address in.

3.5 Connectors

3.5.1 Profibus connection

The profibus connection is according to standard 9-Pin SubD plug. The profibus interface is galvanically separate from the residue of the PK-DP. Use as bus cables only special bifilar, twisted and shielded PROFIBUS cable as well as the pertinent plugs.

3.5.2 Serial interfaces

The serial interfaces are 9-Pin SubD plugs. The assignment of the RS-232 port is PC compatible. Use high-quality shielded cables to avoid EMC problems.

4 Hardware description

4.1 Connector assignment for voltage supply

The voltage supply connector is a 3-Pin plug connector e.g. from Phoenix Contact version FRONT MSTB 2,5/3-ST-5,08. Assignment:
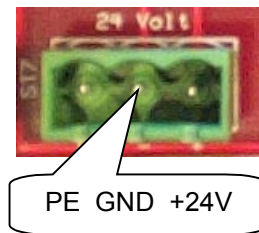
4.2 Connector assignment for Profibus connection

The Profibus connection is a 9-Pin SubD plug with the following assignment (according to IEC 61158):
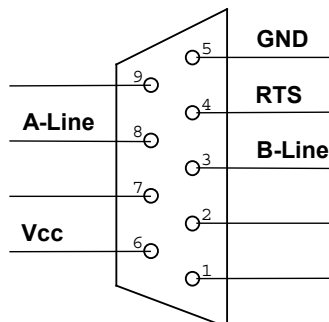
The RTS line is routed over a 330 Ω resistor. The Vcc line may loaded with max. 10 mA. Both lines may not be shortet and/or be loaded beyond their limits. Malfunctions as well as permanent damage are the result.

4.3 Connector assignment of the serial interfaces

The interface COM 1 is varied depending upon assembly. Assignment RS-232:
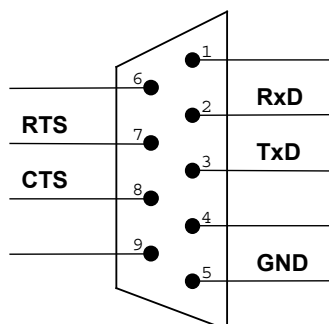
Assignment RS-422:
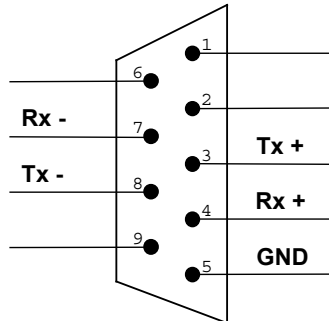
The Rx-line is terminated with 120 Ω. The Tx-line is terminated with 120 Ω and each with 100 KΩ at quiescent level put.

Assignment RS-485:
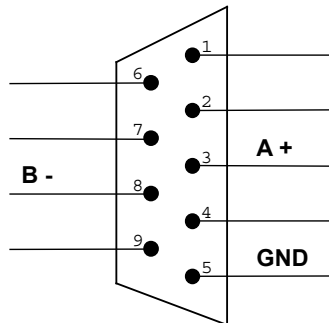
The line is terminated with 120 Ω and each with 100 KΩ at quiescent levels put.

The interface COM 2 is like the interface COM 1 in the version RS-232.

4.4 Meaning of the LED's

One LED indicates the supply voltage, the operating conditions are indicated with 3 further light emitting diodes.

4.4.1 Supply voltage LED

After appling the supply voltage `LE4` - green, 5V - must shine. Thus is the indication that the PK-DP has supply voltage. If `LE4` does not shine, then examine whether the PK-DP is correctly attached to the supply voltage. On the PK-DP is a itself-resetting safety device, which does not have to be changed. If `LE4` does not shine or if it flickers, please return the PK-DP to the repair.

4.4.2 Operating condition LED's

The LED's `LE1`, `LE2` and `LE3` informs about the condition of the Profibus State Machine. After switching on `LE1` and `LE2` - `WaitPrm` and `WaitCnf` - are blinking, up to the first time a valid Profibus DP Baud rate is found.

Afterwards `LE1` - `WaitPrm` - should shine, i.e. the PK-DP is waiting for the parameter telegram of the DP master. The condition `WaitCnf` - `LE2` shine - is normally not to observed,

because immediately a chance to `DATA Exchange` or back to `WaitPrm` take place. Even if the configuration telegram has the right content, the PK-DP changes in the condition `DATA Exchange` and is thereby ready for data exchange.

If a fatal error arises, which the software cannot repair, then the PK-DP changes into the condition error, which is indicated by simultaneous flashing of `LE1` and `LE2`. The PK-DP can be resetet only by separation from the supply voltage.

With Profibus address `00`, an update of the firmware can be accomplished . After switching on `LE2` - `WaitCnf` - shines without interruption.

5 Description of the ASCII drivers

In this chapter the fundamental function of the ASCII drivers is described.

It is common to all drivers that max. 255 characters can transferred in a serial telegram. The data length on the serial interface is independently of the length of the Profibus DP telegrams, if necessary several DP telegrams for transmission are used. On the serial interface a telegram is emitted not until it is transfer completely over the profibus. Likewise it is only passed to the DP master if an end condition was fulfilled, i.e. the telegram was received completely over the serial interface.

The input data of the serial interface are always received, i.e. even if the DP master does not fetch straight data from the PK-DP. The data are analysed according to the adjusted parameters (see below) and temporary buffered, so that no data are lost. Maximally 20 data telegrams of the serial interface can be buffered. The receive data will transfer in the order of their arrival to the DP master.

Please note that if the telegrams over the serial interface are shorter than the telegram length configured on the profibus you may analyze only up to the length of the data! Data behind are invalid!

5.1 Parameter

In the following the individual parameters are described. Not all parameters are evaluated by all drivers.

5.1.1 Handshake

If you select X_{ON}/X_{OFF} only 7 bits data can be transferred. The highest bit is set to 0, i.e. the character set from $0x01$ to $0x7F$ is available. In the RTS/CTS mode all possible 255 characters will be transferd. In the mode "no handshake" the RTS is always active and the CTS is off, so all possible 255 characters can received.

5.1.2 Reading from input data

If input data are requested "without wait", then the PK-DP answers on a request of the master with the data length 0, if no data are present. In the mode "with wait" only an answer will sent if valid input data are present.

5.2 Characteristics of the ASCII drivers

4 different ASCII drivers are available, with which most applications can be work. If you should have an application, which cannot be settled with one of the 4 drivers, let it us know, we gladly provide a special driver for your application.

5.2.1 ASCII driver with Idle time end recognition

This driver rates the time between 2 bytes, which will receive over the serial interface, for end recognition. If the adjusted Idle time is exceeded, then the telegram ended for the PK-DP and is transferred over the profibus to the profibus master. The Idle time is adjustable

in 1 ms steps between `0` and `65535` ms. If the Idle time is set to `0`, then no examination takes place. This attitude is meant for the other ASCII drivers, where no end recognition take place over the Idle time.

5.2.2 ASCII driver constant length

After the number of characters set by the telegram length is received the data is transferred to the DP master. If no additional Idle time recognition is desired, the Idle time must be set to `0`.

5.2.3 ASCII driver stop character

Here characters are received until the stop character is recognized. Afterwards a new telegram begins, until the next stop character will receive. Also here the Idle time monitoring must be turned off if necessary.

5.2.4 ASCII driver starting/stop character

Additionally to the recognition of the stop character this driver also still waits for a start character. Only if the start character is recognized, the telegram begins, it ends then with the stop character. As already described above, if necessary the Idle time monitoring must be deactivated.

6 Start-up software

The PK-DP must be configured and parameterised. Further you must program your DP master accordingly.

If the DP master has a Profibus configurator make start-up thereby. You must copy the GSD file `IEP_069C.GSE` into the appropriate directory of your configurator. The file `PKDP.DIB` you have to place in the bit-map directory of your configurator. You find the PK-DP then under Gateways/IEP. Select the parameters according to your application.

If you do not have a configurator available, you must itself a configuring- and parametrie-sing telegram (see Configuration/Parameteron page 27).

6.1 PK-DP configuration

Select one of the modules indicated in the following table. Your choice should align itself at the length of transferring data. If possible the length should be selected in such a way that the complete data fit into a telegram, otherwise you must divide the serial data on several DP telegrams. Note please that for the transmission 2 bytes Overhead are needed.

Module name	Data length	protocol
ASCII_4B	4 byte	free ASCII driver
ASCII_8B	8 byte	free ASCII driver
ASCII_16B	16 byte	free ASCII driver
ASCII_32B	32 byte	free ASCII driver
ASCII_64B	64 byte	free ASCII driver
ASCII_128B	128 byte	free ASCII driver
ASCII_160B	160 byte	free ASCII driver

With the ASCII_8B e.g. thus 6 byte user data can be transferred in a telegram. Please note that the DP master must take over the data with consistency over the entire data length. After the configuration the PK-DP must be parameterised.

6.2 PK-DP parameterisation

Select the following parameters according to your application:
- ➢ Driver
- ➢ Baud rate
- ➢ Handshake mode
- ➢ Number of bits per character
- ➢ Number of stop bit
- ➢ Parity
- ➢ ASCII-Code start char

➢ ASCII-Code stop char
➢ Telegram length
➢ Idle time
➢ Mode

6.3 Communication DP master <-> PK-DP

The data between a DP master and the PK-DP is exchanged in the form of telegrams. The telegram length is specified with the configuration and adapted in such a way to the data which can be transmitted and the I/O range of the Masters. If possible avoid the use of fragmented telegrams! They are needed only if the number of bytes which can be transferred + 2 byte (for control byte and length) is more largely than the data channel configured.

Example of the module selection: You want 10 byte user data sending/receiving. Select the module ASCII_16B to transfer 10 byte user data + 2 byte = 12 byte. If you select a module with shorter data length, then the user data must be transmitted fragmented.

6.3.1 Structure of the data telegram

A data telegram has the following structure:

Byte 0	Byte 1	Byte 2	Byte 3	...	Byte n
Control byte	Length if necessary	user data			
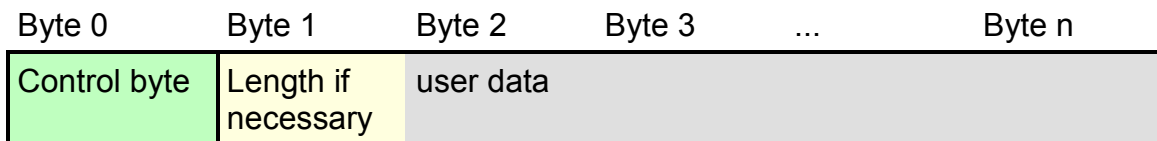
The control byte is always the first byte of a telegram. The length is always located in the first telegram of a new data record, i.e. only with fragmented telegrams the length in the further telegrams is void. Afterwards the user data, either – with no fragmented telegrams – up to the indicated length, or – with fragmented telegrams – up to the end of the data telegram. With fragmented telegrams the last telegram contains if so fewer user data.

6.3.2 Structure of the control byte

With the help of the control byte data flow is controlled and guaranteed that the telegrams were understood correctly from master and slave.

The control byte has the following structure:

OUT (from master to slave)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Choice of the serial interface	RESET	Command Send DATA	Command Send DATA Fragment	Command Receive DATA	Command Receive Fragment	Software Revision query	reserved, send as 0

IN (from slave to master)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
serial interface	RESET	Status Send DATA	Status Send DATA Fragment	Status Receive DATA	Status Receive Fragment	Revision Software	reserved always 1

The master gives the command bits, if the Slave sent the appropriate status bit back, sees the master that the command was understood. The command bits are toggle for each new command. Thus a definite recognition of a new command is ensured. The Toggle bits for fragmenting begin with each new command with 0. The Toggle bits must be administered for the two serial interfaces COM 1 and COM 2 separately.

The bits have the following meaning:

Bit 7	**0** selects the interface COM 1, **1** selects the interface COM 2
Bit 6	with **1** a RESET of the Slave is released, i.e. all hanging orders are broken off
Bit 5	Toggle bit to transfer data from the master to the slave
Bit 4	Toggle bit to transfer fragmentet data from the master to the slave
Bit 3	Toggle bit to transfer data from the slave to the master
Bit 2	Toggle bit to transfer fragmentet data from the slave to the master
Bit 1	The software revision is announce
Bit 0	Reserved for future extensions, at present always **1**

For output data the confirmation is first sent away if the data were actually send out.

The PK-DP transmits only data to the master if she were addressed by the master, i.e. after the PK-DP is changed into the condition DATA Exchange, the first byte in the input words of the Masters with set bit 0 does not appear automatic!

6.3.3 Examples of the data communication

In the following examples of all transmissions the command bits are set to 0 in the starting situation. As module is ASCII_4B selected, so 2 byte for the user data are available.

6.3.3.1 Send DATA without fragmenting

The master would like to send 2 byte data over the serial interface of the PK-DP. The I/O field is largely enough (user data length+2), so that no fragmenting of the user data is nessecary.

Step 1 (starting situation)

OUT	Byte 0							Byte 1	Byte 2	Byte 3	Request (Master)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
0		0									

COM 1

IN	Byte 0							Byte 1	Byte 2	Byte 3	Response (PK-DP)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
0		0					1				

Step 2 (masters sends telegram to COM 1)

OUT	Byte 0							Byte 1	Byte 2	Byte 3	Request (Master)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x02	0x41	0x42	
0		1									

COM 1   Data 1   user data length   user data (A/B)

IN	Byte 0							Byte 1	Byte 2	Byte 3	Response (PK-DP)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
0		1					1				

acknowledge 1
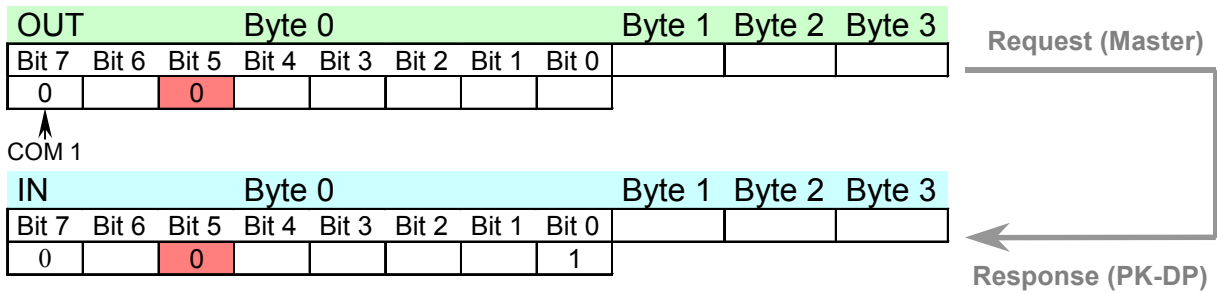
Step 3 (master sends further telegram to COM 1)

OUT	Byte 0							Byte 1	Byte 2	Byte 3	Request (Master)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x01	0x51		
0		0									

COM 1   data 2   user data length   user data (P)

IN	Byte 0							Byte 1	Byte 2	Byte 3	Response (PK-DP)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
0		0					1				

acknowledge 2
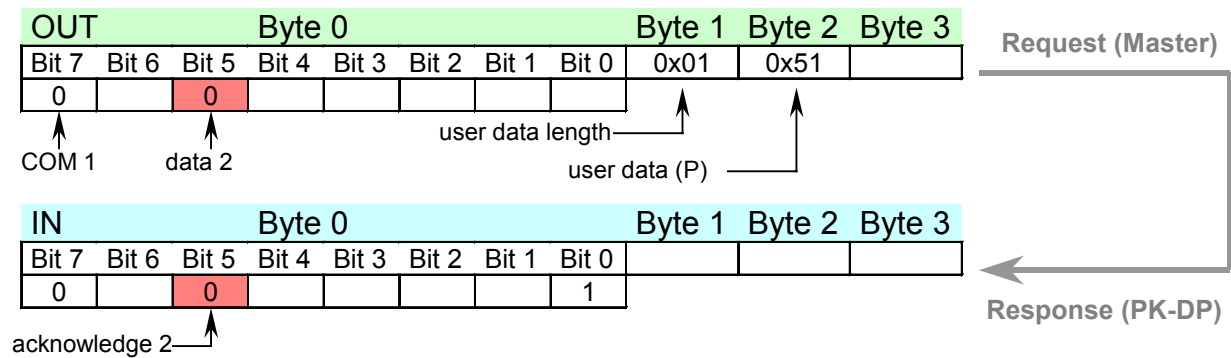
For each new telegram, which the master would like to send, bit 5 has to toggle. If the toggle bit is sent back from the PK-DP, the master can be safe that the data were sent away. Primary then the master may send a new telegram.

6.3.3.2 Receive DATA without fragmenting

The master would like to receive data over the serial interface COM 1 of the PK-DP.

Step 1 (starting situation)

OUT	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
0				0						

↑ COM 1 — Request (Master)

IN	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
0				0			1			

Response (PK-DP)

Step 2 (masters demands receive data of the Slave)

OUT	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
0				1						

↑ receive data 1 — Request (Master)

IN	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x02	0x41	0x42
0				1			1			

↑ COM 1 acknowledge 1 — Response (PK-DP)
user data length — 0x02
user data (A/B) — 0x41, 0x42

Step 3 (master demands further receive data from the Slave over COM 2)

OUT	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
1				0						

↑ COM 2 receive data 2 — Request (Master)

IN	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x02	0x51	0x52
1				0			1			

↑ COM 2 acknowledge 2 — Response (PK-DP)
user data length — 0x02
user data (P/Q) — 0x51, 0x52

For each new telegram, which the master would like to receive, bit 3 has to toggle. If the bit is sent back to the PK-DP, the master can be sure that the telegram contains valid data. In bit 7 the master specifies, from which interface he would like to receive data.

6.3.3.3 Send DATA with fragmenting

The master would like to send 7 byte data over the serial interface COM 1 of the PK-DP. The I/O field is too small, so that a fragmenting of the user data is nessecary.

Step 1 (starting situation)

OUT			Byte 0					Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				**Request (Master)**
0		0	0								

↑ COM 1

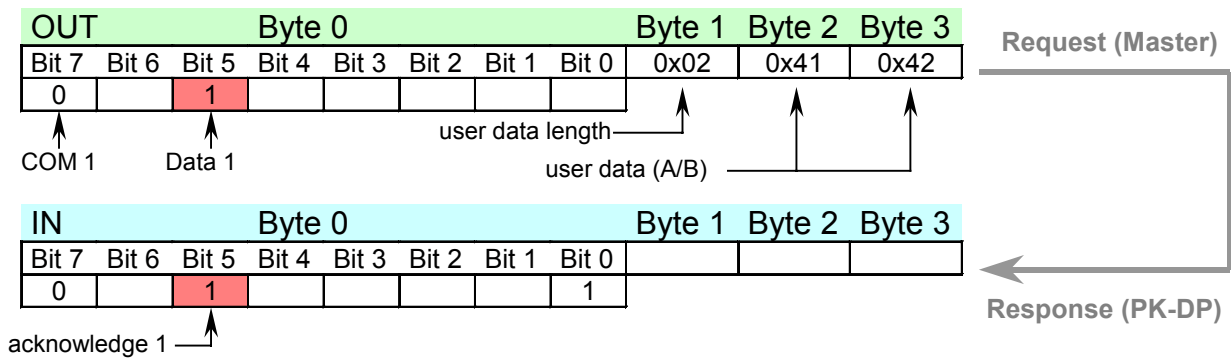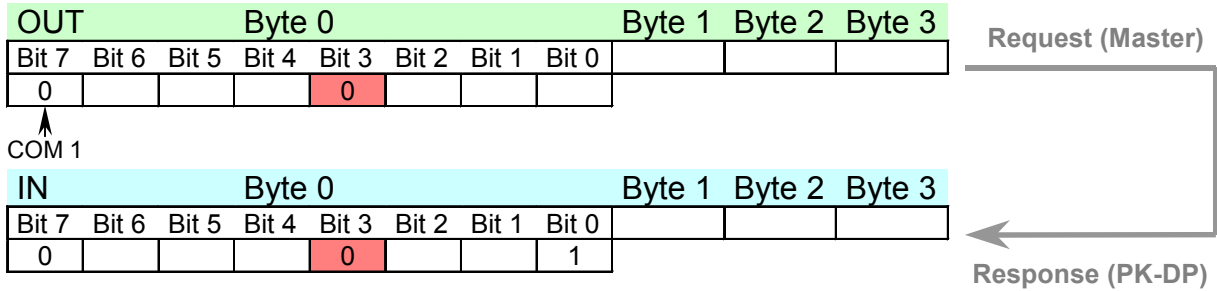| IN			Byte 0					Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
0		0	0				1				**Response (PK-DP)**

Step 2 (masters sends telegram to COM 1)

OUT			Byte 0					Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x07	0x41	0x42	**Request (Master)**
0		1	0								

↑ COM 1   ↑ data 1

user data length ─── 0x07

user data (A/B) ─── 0x41 0x42

IN			Byte 0					Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
0		1	0				1				**Response (PK-DP)**

acknowledge 1 ───↑
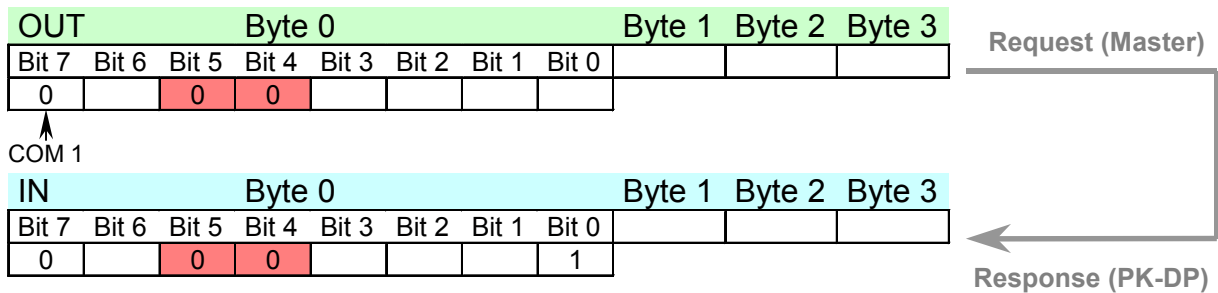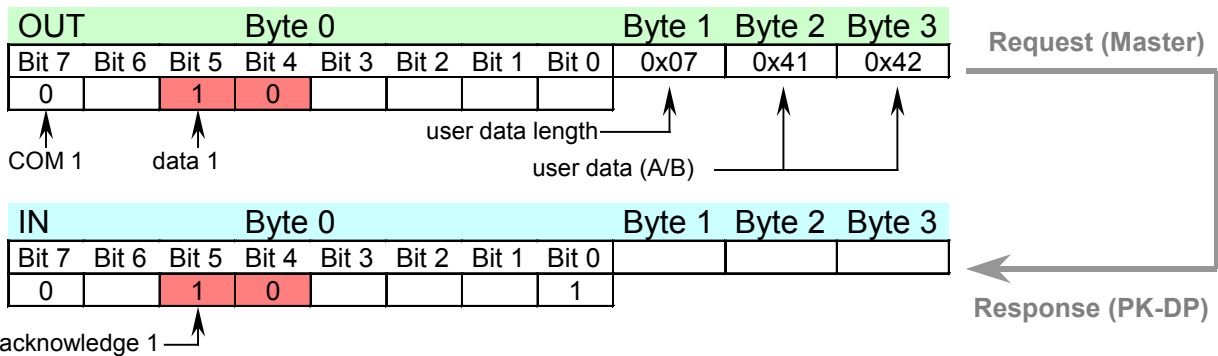
Step 3 (master sends 2nd fragment of the telegram to COM 1)

OUT			Byte 0					Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x43	0x44	0x45	**Request (Master)**
0		1	1								

↑ COM 1   ↑ data 1   ↑ toggle bit fragment

user data (C/D/E) ─── 0x43 0x44 0x45

IN			Byte 0					Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
0		1	1				1				**Response (PK-DP)**

acknowledge 1 ───↑
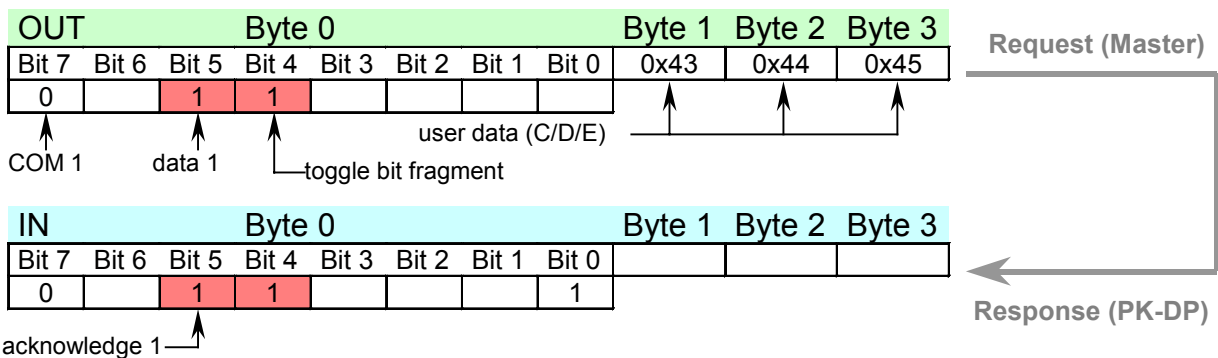
Step 4 (masters sends 3rd fragment of the telegram to COM 1)

OUT	Byte 0							Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x46	0x47		Request (Master)
0		1	0								

COM 1 (Bit 7) · data 1 (Bit 5) · toggle bit fragment (Bit 4) · user data (F/G) → 0x46 / 0x47

IN	Byte 0							Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				Response (PK-DP)
0		1	0				1				

acknowledge 1 (Bit 5)
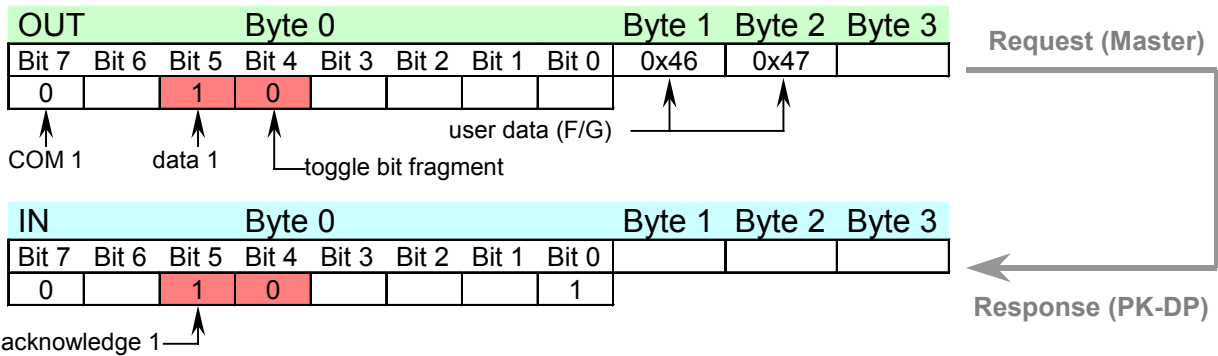
Step 5 (masters sends next telegram to COM 1)

OUT	Byte 0							Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x09	0x51	0x52	Request (Master)
0		0	0								

COM 1 (Bit 7) · data 2 (Bit 5) · user data length → 0x09 · user data (Q/P) → 0x51 / 0x52

IN	Byte 0							Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				Response (PK-DP)
0		0	0				1				

acknowledge 2 (Bit 5)
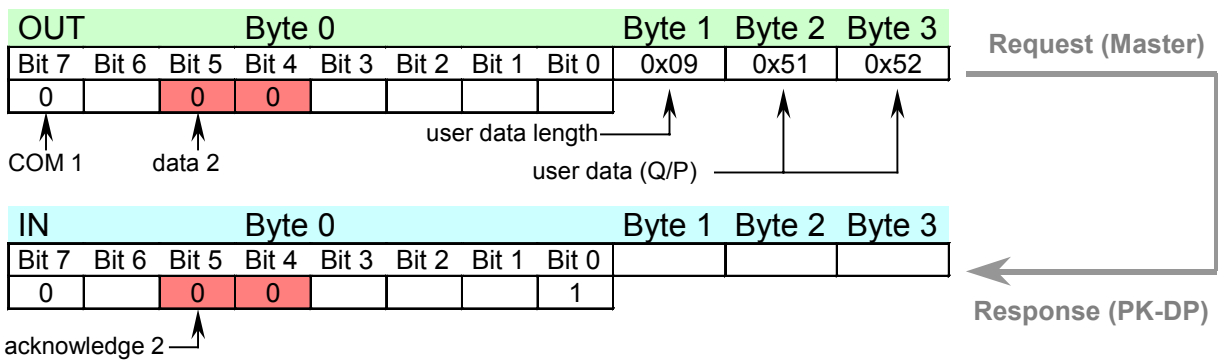
For each new telegram, which the master would like to send, bit 5 has to toggle. Within a fragmented telegram only bit 4 has to toggle. If the toggle bit is sent back to the PK-DP, the master can be sure that the data is sent away. Primary then the master may send a new telegram.
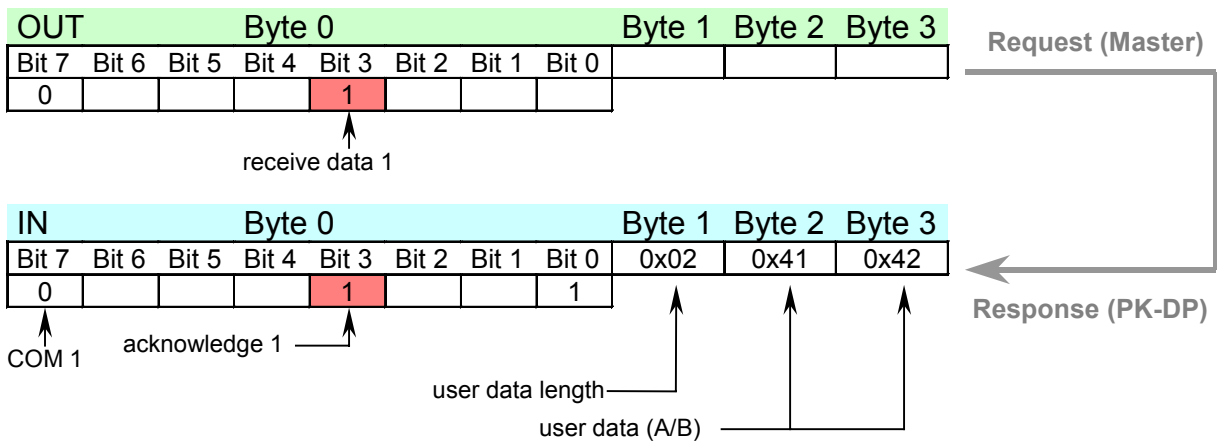
6.3.3.4 Receive DATA with fragmenting

The master would like to receive data over the serial interface COM 1 of the PK-DP.
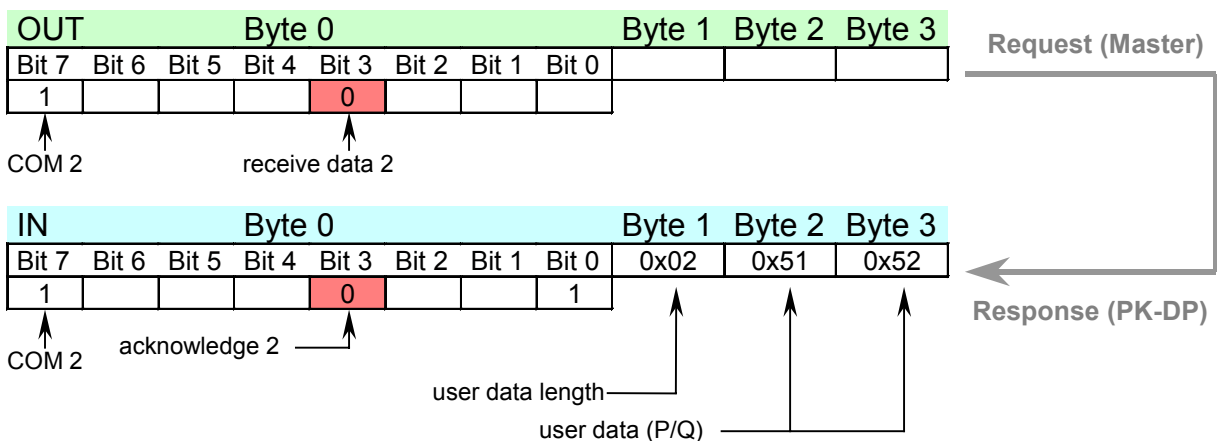
Step 1 (starting situation)

OUT	Byte 0							Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				Request (Master)
0				0	0						

COM 1 (Bit 7)

IN	Byte 0							Byte 1	Byte 2	Byte 3	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				Response (PK-DP)
0				0	0		1				

Step 2 (masters demands receive data from the Slave)

OUT	Byte 0								Byte 1	Byte 2	Byte 3	Request (Master)
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
	0				1	0						

receive data 1

IN	Byte 0								Byte 1	Byte 2	Byte 3	Response (PK-DP)
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x07	0x41	0x42	
	0				1	0		1				

COM 1
acknowledge 1
user data length
user data (A/B)
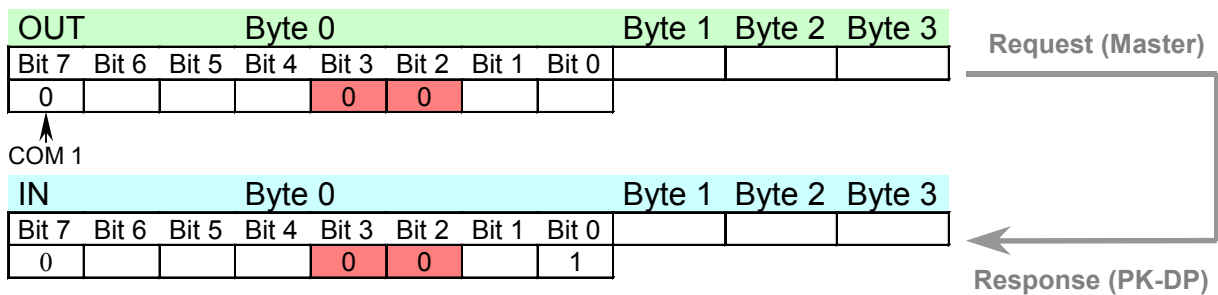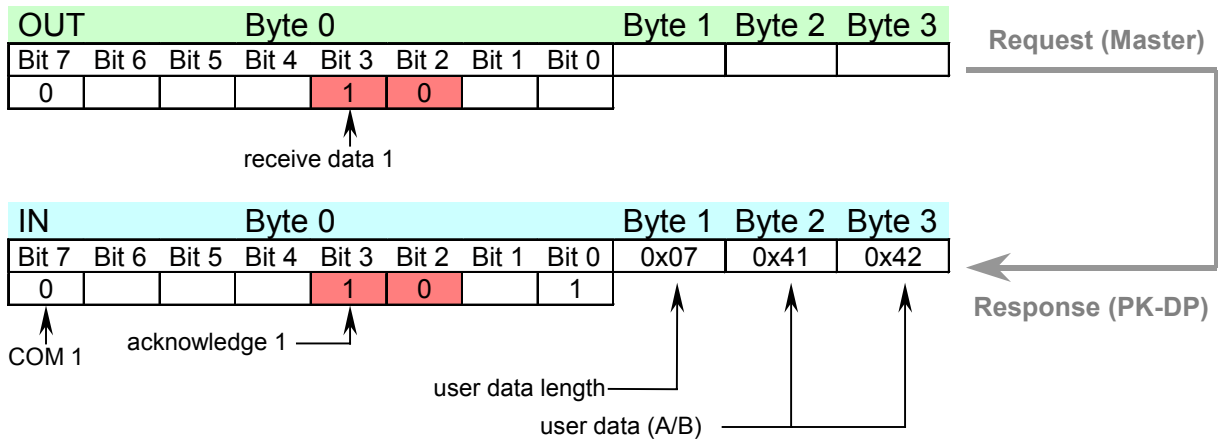
Step 3 (master demands 2nd fragment from the Slave)

OUT	Byte 0								Byte 1	Byte 2	Byte 3	Request (Master)
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
	0				1	1						

receive data 1

IN	Byte 0								Byte 1	Byte 2	Byte 3	Response (PK-DP)
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x43	0x44	0x45	
	0				1	1		1				

COM 1
acknowledge 1
toggle bit fragment
user data (C/D/E)
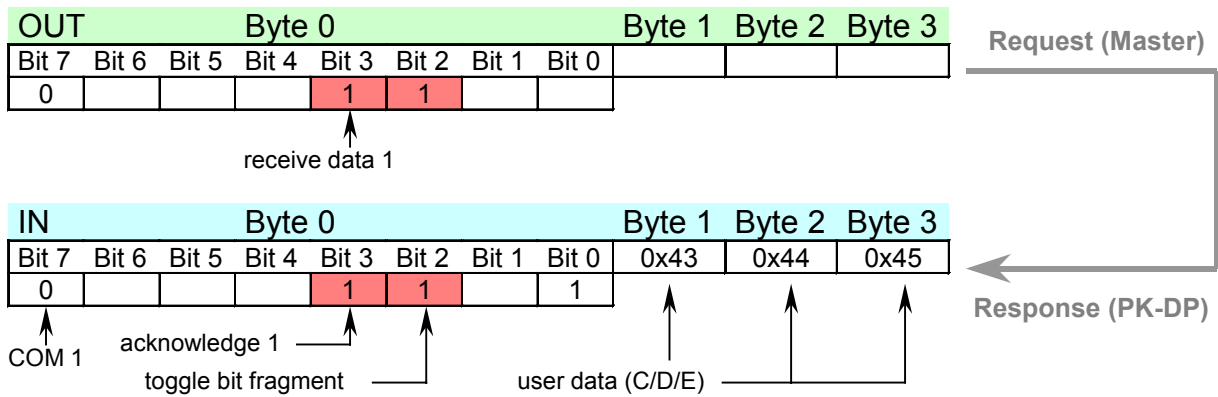
Step 4 (masters demands 3rd fragment from the Slave)

OUT	Byte 0								Byte 1	Byte 2	Byte 3	Request (Master)
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
	0				1	0						

receive data 1

IN	Byte 0								Byte 1	Byte 2	Byte 3	Response (PK-DP)
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x43	0x46	0x47	
	0				1	0		1				

COM 1
acknowledge 1
toggle bit fragment
user data (F/G)
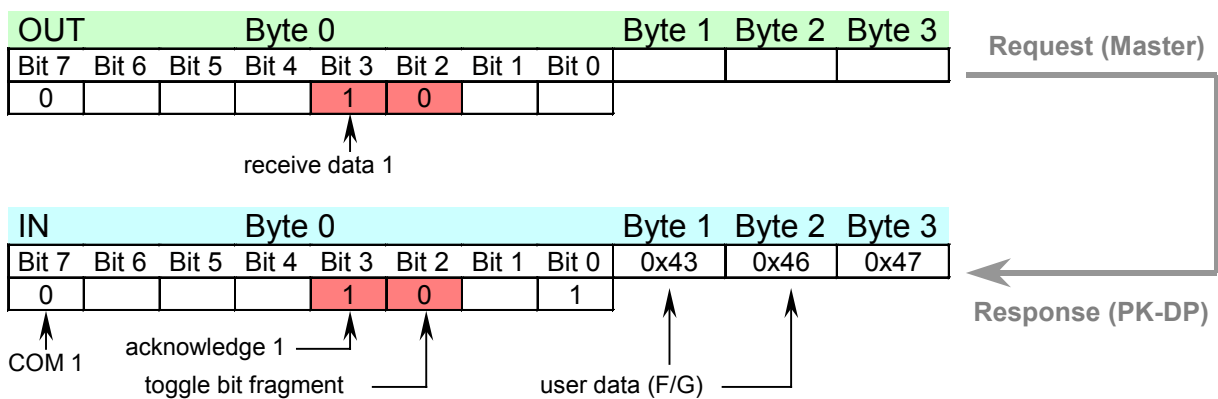
Step 5 (masters demands further telegram from the Slave from COM2)

OUT	Byte 0							Byte 1	Byte 2	Byte 3	Request (Master)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
1				0	0						

receive data 1

IN	Byte 0							Byte 1	Byte 2	Byte 3	Response (PK-DP)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x09	0x51	0x52	
1				0	0		1				

COM 2   acknowledge 1
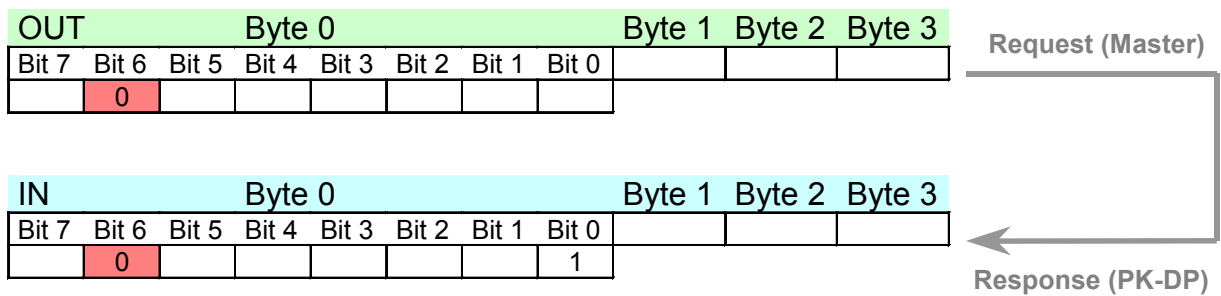
user data length

user data (P/Q)
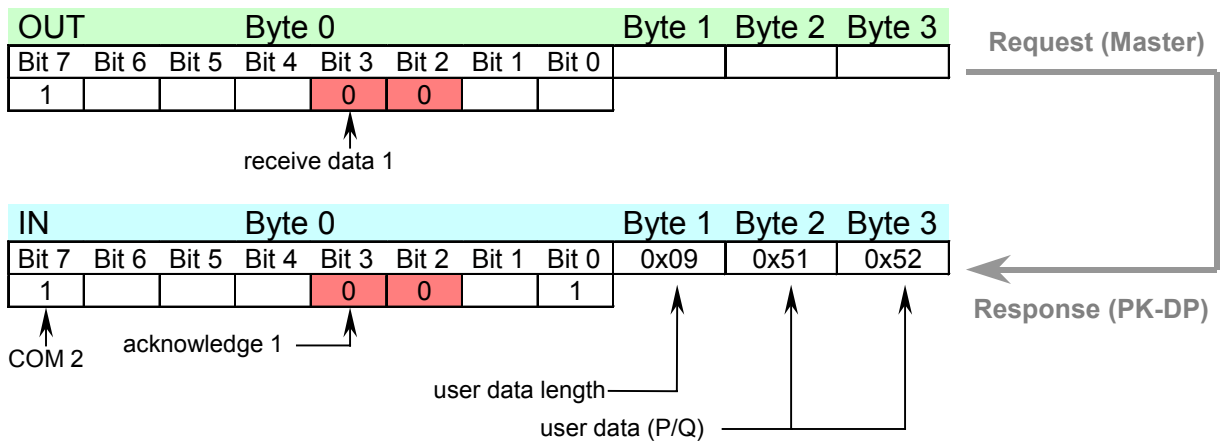
For each new telegram, which the master would like to receive, bit 3 has to toggle. Within a telegram only bit 2 has to toggle. If the toggle bit is sent back from the PK-DP, the master can be sure that the telegram contains valid data. If both interfaces are used, the master can recognize in bit 7, from which interface the data has come.
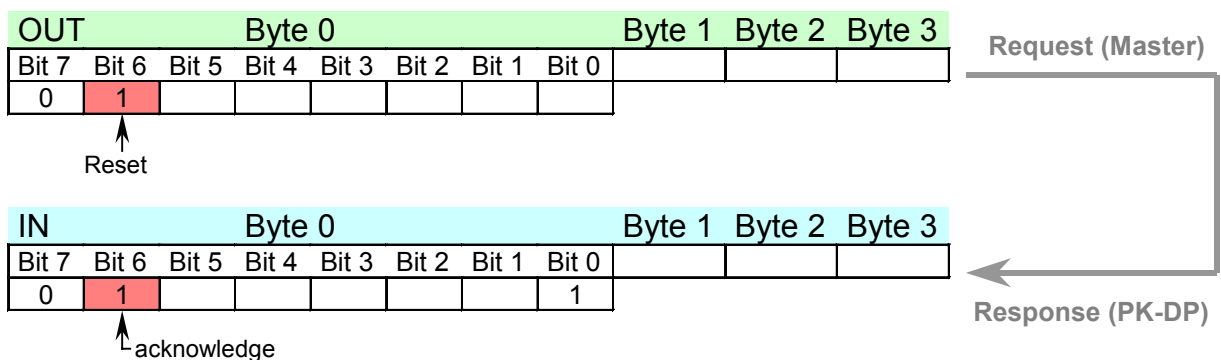
6.3.3.5 RESET of the PK-DP

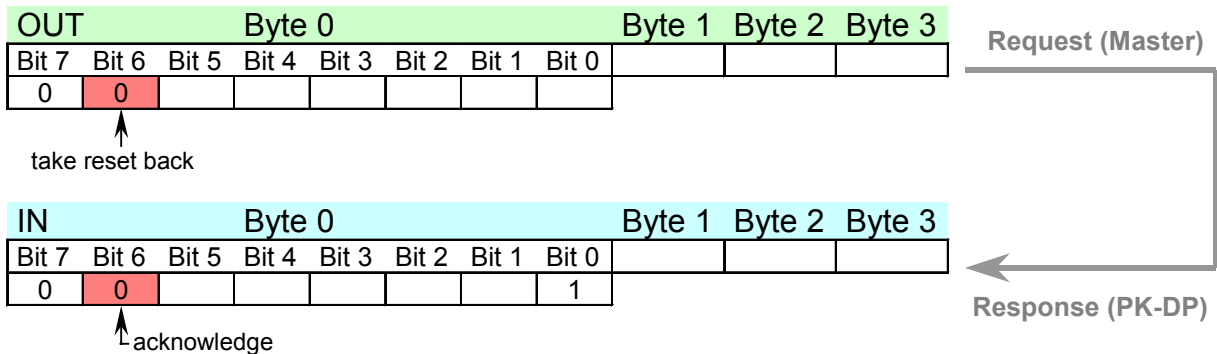Over the RESET bit all still running requests can be deleted on the PK-DP. Further all toggle bits are set again to 0.
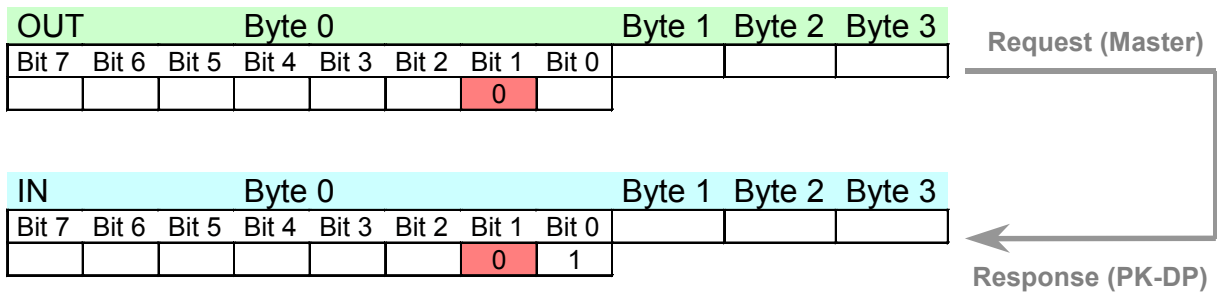
Step 1 (starting situation)

OUT	Byte 0							Byte 1	Byte 2	Byte 3	Request (Master)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
	0										

IN	Byte 0							Byte 1	Byte 2	Byte 3	Response (PK-DP)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
	0						1				

Step 2 (RESET of the PK-DP)

OUT	Byte 0							Byte 1	Byte 2	Byte 3	Request (Master)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
0	1										

Reset

IN	Byte 0							Byte 1	Byte 2	Byte 3	Response (PK-DP)
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
0	1						1				

acknowledge

Step 3 (RESET is accomplished)

OUT	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
0	0									

↑ take reset back

Request (Master)

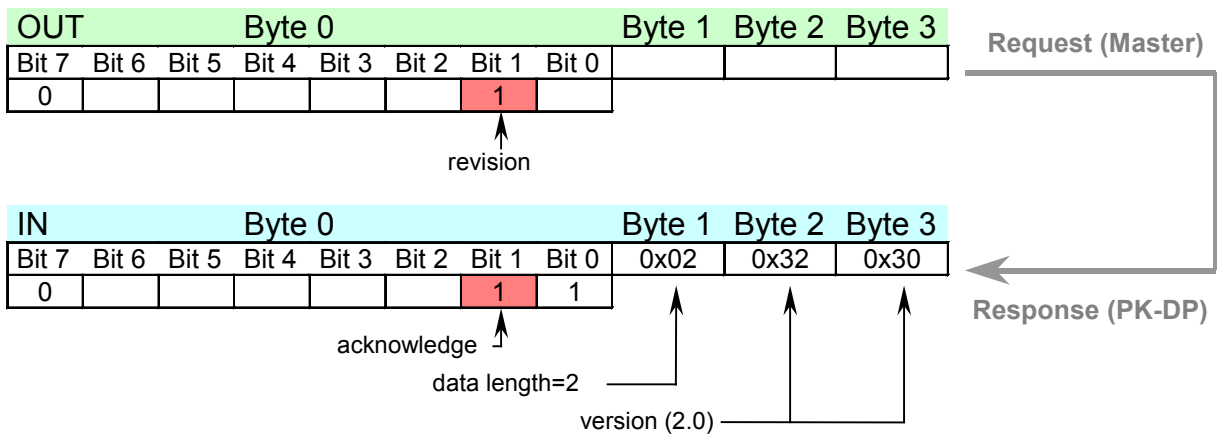| IN	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
0	0						1			

↑ acknowledge

Response (PK-DP)

6.3.3.6 Software revision number query

With this bit the software revision number of the PK-DP is queried. The data will transfer in the same format as receive data:

Step 1 (starting situation)

OUT	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
						0				

Request (Master)

IN	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
						0	1			

Response (PK-DP)

Step 2 (revision number of the PK-DP)

OUT	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
0						1				

↑ revision

Request (Master)

IN	Byte 0							Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0x02	0x32	0x30
0						1	1			

acknowledge ↑

data length=2

version (2.0)

Response (PK-DP)

6.4 References to communication

You have to read the following lines accurately, otherwise you can get unexplainable data loss!

You got no problems, if communication between DP master and PK-DP is as half duplex implemented. Your application sends or receives data, but never both at the same time. Only if a communication procedure is final, the next one starts.

Problems can arise, if you want to receive and send data at the same time. Imagine the following case: You are straight thereby to receive fragmented data on the DP master. Now also data are sent away at the same time, i.e. not only the receive fragment toggle bit but also the send toggle bit change. In the telegram also transmit data will transfer. On the PK-DP 2 actions are released at the same time: On the one hand an answer telegram is made finished, in that the next received data stands. This telegram is transferred immediately to the DP master. At the same time the data which can be send are send over the serial interface. If the data are sended, the confirmation telegram is sent to the DP master and overwrites there the briefly before received data telegram. If your master did not read this data telegram with the receive data yet, the data are lost!

Differently verbalised: **You may never activate more than one action on the PK-DP, which returns data!**

7 Firmware update

Please follow exact these instructions. Do not delete other sectors in the Flash than indi-
cated here!

1) Connect the interface COM 2 with a PC. You can use a commercial zero-modem cable.

2) Start on your PC a terminal program, e.g. our RTW driven out as Freeware.

   To find under: http://www.iep.de/Downloads/programme/rtw.zip. RTW must not be in-
   stalled , unpack the archives into any directory and start the program Rterm32.exe.

3) Set the transfer parameters to 57600 Baud, 8 bits, 1 stop bit and no parities.

4) Adjusting the profibus address to 0.

5) Switch the PK-DP on.

6) In order to be able to enter a command on the PK-DP, you must press always first the
   keys Ctrl/Strg and A together. As reaction you receive a "*". Now you can enter a
   command.

7) Test whether connection to the PK-DP exists, by entering a S after Ctrl/Strg and
   A. You receive a storage allocation list.

8) Enter the instruction "rtflash -b -x -s 3 -d" – after you have pressed Ctrl/Strg
   A. You must receive the following reaction:

   ```
   'flash_mode' set to BYTE
   Program and Delete Flash with Interrupts-OFF
   'start_sector' set to 3
   Sector 3: $00D30000 $00D3ffff -> clean
   >> RTFLASH/0000: (TERMI).
   ```

   You have deleted the profibus firmware.

9) Enter the instruction "rtflash -b -x -t D30000 -r /tyb" – after you have
   Ctrl/Strg A pressed. You must receive the following reaction:

   ```
   'flash_mode' set to BYTE
   Program and delete Flash with Interrupts-OFF
   'start_addr' set toD30000
   Reading S-Record '/tyb'
   ```

10) Now the new firmware must be transferred. If you use the RTW call the menu "`Datei/Senden`". In the selection of the files box you select the firmware "`pkdp21.sr`" . The number can look differently according to the revision.

Importantly: The check box "`Ohne RTOS-Befehle senden`" must be selected. Press on "`Senden`".
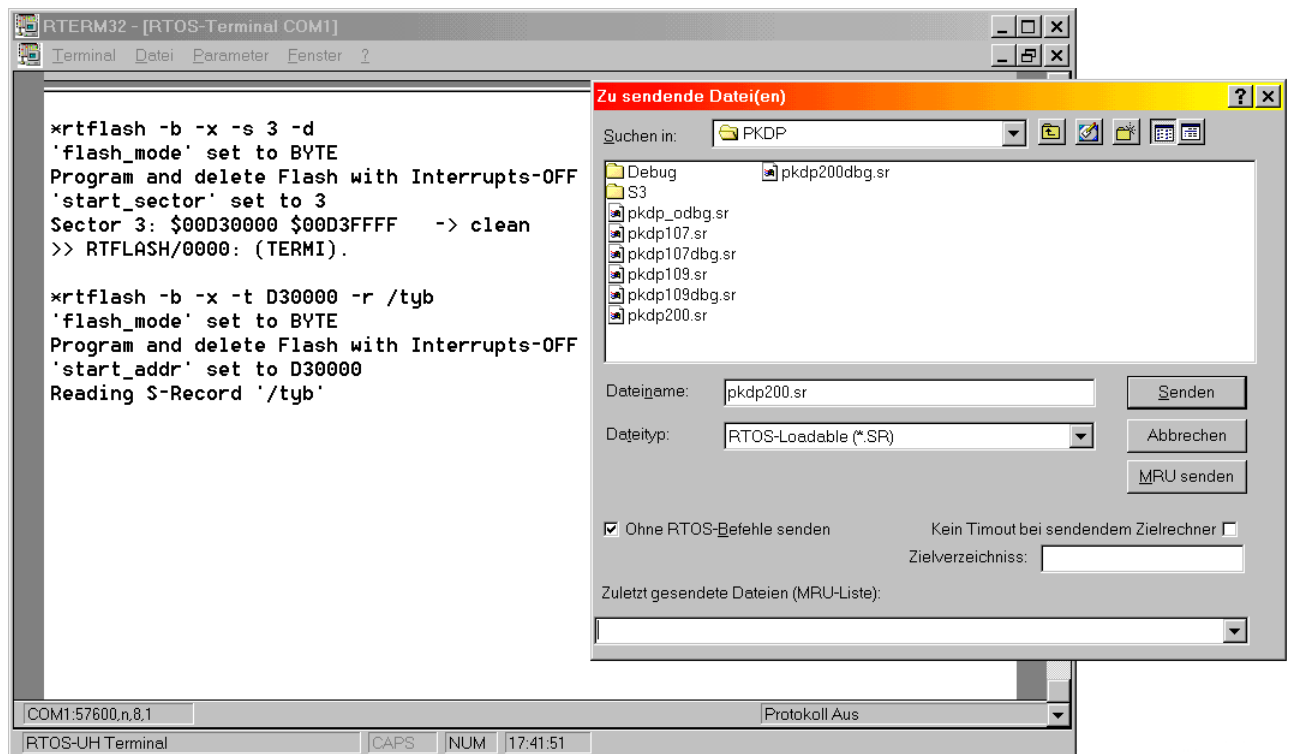
While the file will transfer, see in the status line of the RTW a progress bar. If the file is completely transferred, then you must enter `Strg/Ctrl D` - to press again at the same time. You receive the following reaction:

```
New 'start_addr' set to D3A7A0
>> RTFLASH/0001: (TERMI).
```

The address can vary naturally.

11) Now you are already finished, reset the profibus address and interrupt the voltage supply. Now the update is succedd. If your effort was without success, repeat all steps carefully.

7.1 Debug version of the firmware

On the provided CD you find a Debug version of the firmware. You can install this version on the PK-DP, if you have problems with start-up. The installation effected as in the chapter firmware update described. You recognize the Debug version of the firmware to `dbg` in the name.

The Debug firmware spends information about the internal condition of the PK-DP about the COM2. You can attach a PC with terminal program to this interface (57600 Baud, 8 bits, 1 stop bit and no parities) and received div. Expenditures.

The most important expenditures are the following:

During the initialization the PK-DP spends the data parameters and config telegram. You can examine whether the desired attitudes arrived on the PK-DP.

If the PK-DP gets new (i.e. the control byte has changed) data from the master, it always spends "`Daten vom Master, Steuerbyte=`*XX*". *XX* is the new control byte. If new data from the PK-DP are handed over to the master, the expenditure "`Daten zum Master: `*XXXXXXXX*". *XXXXXXXX* are the first 4 byte, which will transfer to the master. Please consider, which the control byte and the data are shown in hexadecimals.

Data, which are send over the serial interface, are at the expenditure "`Send COM1/2: `*YY* `daten verschickt`" to recognize. When receiving "`COM1 `*YY* `Zeichen empfangen`" is shown. *YY* is the number of characters. The expenditure "`Receive COM1: Puffer `*Z* ` in Arbeit`" shows that for the PK-DP a receive buffer is finished, which can be handed over to the master.

The further expenditures are internal conditions.

H I N T:

Internal speed of operation is strongly limited by the debug expenditures. The cycle times can rise from <10 ms to several 100 ms. You may set also the Profibus watchdog not to small values, otherwise the Watchdog can react.

After successful start-up you should bring the original firmware in again!

8 Configuration/Parameter

If you have no Profibus configurator, you must provide the configuration and parameter telegram by hand. In the following chapters you find the necessary information for it.

8.1 Configuration telegram

The configuration telegram consists depending upon number and size of the entrances and exits of module identifications. With the module identification you can select the mode of operation of the PK-DP. Select from the following table your configuration:

Module name	Data length	Parameter
ASCII_4B	4 byte	C1, 83, 83, 01
ASCII_8B	8 byte	C1, 87, 87, 01
ASCII_16B	16 byte	C1, 8F, 8F, 01
ASCII_32B	32 byte	C1, 9F, 9F, 01
ASCII_64B	64 byte	C1, BF, BF, 01
ASCII_128B	128 byte	C1, FF, FF, 01
ASCII_160B	160 byte	C0, FF, FF, C1, CF, CF, 01

8.2 Parameter telegram

The parameter telegram consists of bus scope parameters and of PK-DP specific parameters.

8.2.1 Bus scope parameter

The first 7 bytes (byte 1 to byte 7) of the parameter telegram contain bus scope parameters, which are standardized after EN 50170, whereby byte 5 and 6 contain the Vendor ID (Hexcode 0x069C). The attitudes of the remaining bytes depend on your net configuration (see to EN 50170).

Byte	Parameter	Value	Hexcode	Default value
1	Watchdog etc..			
2	WD_Fact_1			
3	WD_Fact_2 (TWD=10ms*WD_Fact_1*WD_Fact_2)			
4	TSDR			
5	ID number High		0x06	0x06
6	ID number Low		0x9C	0x9C
7	Group Ident			

8.2.2 Specific parameters of the PK-DP

The following 25 bytes (byte 8 to byte 32) of the parameter telegram contains PK-DP specific parameters.

Byte	Parameter	Value	Hexcode	Default
8	-	0x00 because of SPC3	0x00	0x00
9	COM1: Move mode	ASCII driver with Idle time end recognition	0x00	0x00
		ASCII driver fixed length	0x01	
		ASCII driver stop character	0x02	
		ASCII driver starting/stop character	0x03	
10	COM1: Baud rate	75 Baud	0x00	0x09
		110 Baud	0x01	
		300 Baud	0x02	
		600 Baud	0x03	
		1200 Baud	0x04	
		2400 Baud	0x05	
		3600 Baud	0x06	
		4800 Baud	0x07	
		7200 Baud	0x08	
		9600 Baud	0x09	
		19200 Baud	0x0A	
		38400 Baud	0x0B	
		57600 Baud	0x0C	
		76800 Baud	0x0D	
11	COM1: Handshake	XON/XOFF	0x00	0x02
		RTS/CTS	0x01	
		no handshake	0x02	
12	COM1: Number of bits	5 bits	0x05	0x08
		6 bits	0x06	
		7 bits	0x07	
		8 bits	0x08	
13	COM1: Amount of stop bit	1 stop bit	0x01	0x01
		2 stop bit	0x02	
14	COM1: Parity	no parity (NO)	0x00	0x00
		even parity (even)	0x01	
		odd parity (odd)	0x02	
15	COM1: Start indication	ASCII-Code of the start character 0...255	0xXX	0x02
16	COM1: Stop indication	ASCII-Code of the stop character 0...255	0xXX	0x03
17	COM1: Telegram length	1...255 bytes	0xXX	0x01
18	COM1: Reading from input data	without wait	0x00	0x01
		with wait	0x01	
19+20	COM1: Idle time (*1ms)	0...65535 * 1 ms	0xXXXX	0x0064

Byte	Parameter	Value	Hexcode	Default
21	COM2: Move mode	ASCII driver with Idle time end recognition	0x00	0x00
		ASCII driver fixed length	0x01	
		ASCII driver stop character	0x02	
		ASCII driver starting/stop character	0x03	
22	COM2: Baud rate	75 Baud	0x00	0x09
		110 Baud	0x01	
		300 Baud	0x02	
		600 Baud	0x03	
		1200 Baud	0x04	
		2400 Baud	0x05	
		3600 Baud	0x06	
		4800 Baud	0x07	
		7200 Baud	0x08	
		9600 Baud	0x09	
		19200 Baud	0x0A	
		38400 Baud	0x0B	
		57600 Baud	0x0C	
		76800 Baud	0x0D	
23	COM2: Handshake	XON/XOFF	0x00	0x02
		RTS/CTS	0x01	
		no handshake	0x02	
24	COM2: Number of bits	5 bits	0x05	0x08
		6 bits	0x06	
		7 bits	0x07	
		8 bits	0x08	
25	COM2: Amount of stop bit	1 stop bit	0x01	0x01
		2 stop bit	0x02	
26	COM2: Parity	no parity (NO)	0x00	0x00
		even parity (even)	0x01	
		odd parity (odd)	0x02	
27	COM2: Start indication	ASCII-Code of the start character 0...255	0xXX	0x02
28	COM2: Stop indication	ASCII-Code of the stop character 0...255	0xXX	0x03
29	COM2: Telegram length	1...255 bytes	0xXX	0x01
30	COM2: Reading from input data	without wait	0x00	0x01
		with wait	0x01	
31+32	COM2: Idle time (*1ms)	0...65535 * 1 ms	0xXXXX	0x0064

References to the individual bytes:

Byte 9/21: The driver with Idle time end recognition recognizes after the adjusted Idle time (Byte19/20 and/or. 31/32) the end of a data telegram. This end recognition takes place also at the other drivers. If the end recognition over the time is not to take place, then the Idle time is to set to 0.

Byte 11/23: One selects XON/XOFF, only 7 bits data can be transferred. The highest bit is set to `0`, i.e. the character set from `0x01` to `0x7F` is available. In the RTS/CTS mode all possible 255 characters will transfer, which flow control realized over RTS/CTS. If you select "no handshake", RTS is set to 1, i.e. the opposite side may send and CTS is switched off. All 255 possible characters can be transmitted .

Byte 19/20 and/or. 31/32: You must place the Hexcode for the Idle time in the Motorola format (High byte first). If the time is set to `0`, no monitoring takes place.

Byte 15/27: The start character finds only consideration with choice "ASCII driver starting/stop character".

Byte 16/28: The stop character finds only consideration with choice "ASCII drivers stop character" and " ASCII drivers starting/stop character".

Byte 17/29: The telegram length finds only consideration with choice "ASCII drivers fixed length".

Byte 18/30: If input data are requested "without wait", then the Slave answers on a request of the master with the data length `0`, if no data are present. In the mode of operation "with wait" only an answer will sent away if valid input data are present.

9 Trouble shooting

1) The PK-DP cannot be addressed by the master, the WaitPrm and WaitCnf LED flashes alternating.

   a possible cause: You did not consider that the address is setting up in hexadecimals to the PK-DP.

2) The PK-DP rejects the parameter data, i.e. she remains in the condition WaitPrm.

   1. There are configurators, which pre-occupy the parameter bytes with 0 instead of the default values from the GSD file. Then it can occur e.g. that the 2nd interface is not parameterised and the PK-DP rejects the parameter data.

   2. You used a universal module. Only the modules from the GSD file may be used, since the PK-DP examines the parameter data.