

ID 40

3 842 530 344/2014-05

Replaces: 2014-01

EN

Edition **4.1**

Manual



The data specified only serve to describe the product. No statements concerning a certain condition or suitability for a certain application can be derived from our information. The information given does not release the user from the obligation of own judgment and verification. It must be remembered that our products are subject to a natural process of wear and aging.

This document, as well as the data, specifications and other information set forth in it, are the exclusive property of Bosch Rexroth AG. It may not be reproduced or given to third parties without the consent of Bosch Rexroth AG.

This manual was originally written in German.

Inhalt

1	About this manual	9
1.1	Scope of Application	9
1.2	Layout of this manual	9
1.3	Display	10
1.3.1	Numbers	10
1.3.2	Operating states	10
1.3.3	Information	10
1.4	Safety instructions	11
1.5	New in the 4.x versions of the ID 40/SLK software	11
2	Introduction	12
2.1	ID 40/MDT mobile data tag	12
2.2	ID 40/SLK read/write head	13
2.2.1	SLK operating states	14
2.2.2	Status display	18
2.3	Data transmission between SLK and MDT	20
2.3.1	Position of MDT and SLK during data transmission	20
2.3.2	Static and dynamic data transmission	21
2.3.3	Direct and parameterized data transmission	22
2.3.4	Securing data transmission	22
3	Installing MDT and SLK	23
3.1	Mounting the MDT on the workpiece pallet	23
3.2	Mounting the SLK on transfer section profiles	23
3.3	Antenna orientation	24
3.4	SLK electrical connection	25
3.4.1	Supply voltage	25
3.4.2	Connecting the fieldbus	26
3.4.3	Serial interface	27
3.4.4	Turning on the SLK	28
4	MDT memory structure	29
4.1	ID 40/MDT storage	29
4.2	Organization of MDT memory	30
4.2.1	MDT user data area	31
4.2.2	MDT system data area	31
4.2.3	MDT register area	32
5	SLK memory structure	34
5.1	Map of MDT user data area	35
5.2	Map of MDT register area	35
5.2.1	Map of MDT status register	35
5.2.2	Map of MDT pointer registers	35
5.2.3	Map of MDT ID code	35
5.2.4	MDT counter	35
5.2.5	MDT formatting	35
5.2.6	Map of MDT software version	37
5.3	SLK register area	37
5.3.1	Actual link state	37
5.3.2	Commanded link state	38

5.3.3	Auto mode	38
5.3.4	SLK operative flag	39
5.3.5	Look ahead function	39
5.3.6	SLK device information	39
6	Accessing data on the ID 40/MDT	40
6.1	Command-oriented data transmission	40
6.2	Event-oriented data transmission	40
6.3	Direct data exchange with MDT	40
6.4	Parameterized data exchange with MDT	41
6.4.1	Prefetch	41
6.4.2	Pretransmit	42
6.5	Default process for SLK-MDT communication	44
6.6	“Open transfer” and “Close transfer” functions	44
6.7	Auto reconnect function	45
6.8	Auto disconnect function	46
6.9	MDT lifeguarding	47
7	Profibus DP	49
7.1	Overview	49
7.2	Command-oriented data exchange	49
7.2.1	Profibus commands	52
7.3	Event-oriented data exchange	54
7.3.1	Status information in event-oriented data channel	54
7.3.2	Data array for event-oriented data channel	55
7.4	SLK address table	56
7.5	Profibus error codes	57
7.6	Profibus Diagnostic Service	57
7.7	Addressing data in the ID 40 system	58
7.8	Profibus GSD file	59
7.8.1	Distribution of SLK module configuration	59
7.8.2	Sample I/O module configuration	59
7.9	Sample applications	60
7.9.1	“Manual workstation” application	60
7.9.2	“Outfeed” application	62
7.9.3	“Workstation” application	63
7.9.4	“Resetting SLK” application	65
7.9.5	Handling “E00” errors	65
8	Interbus	66
8.1	Overview	66
8.2	Addressing data in the ID 40 system	66
8.3	Command-oriented data exchange	67
8.3.1	PCP communication	68
8.3.2	PCP channel objects	69
8.3.3	Example: writing bytes to MDT	71
8.4	Event-oriented data exchange	72
8.4.1	Status information in the process data channel	72
8.4.2	Data array for event-oriented data channel	73
8.5	Commanded link state via process data channel	74
8.5.1	Output map structure	74
8.5.2	Sample sequence	74

8.6	SLK address table	75
8.7	Interbus error codes	76
8.8	Sample applications	77
8.8.1	“Manual workstation” application	77
8.8.2	“Outfeed” application	79
8.8.3	“Workstation” application	80
8.8.4	“Resetting SLK” application	83
8.8.5	Handling “E00” errors	84
9	CANopen	85
9.1	Overview	85
9.2	Object directory	85
9.3	Command-oriented data exchange	86
9.4	Event-oriented data exchange	86
9.5	MDT data transmission with transfer buffers	87
9.5.1	Transfer buffer parameters (objects 2100–2103)	88
9.5.2	Object mapping in the transfer buffer (objects 2110–2113)	89
9.5.3	SLK address table	91
9.5.4	Transfer buffers (objects 2120–2123)	92
9.5.5	Direct data exchange	92
9.5.6	Prefetch data exchange	92
9.5.7	Pretransmit data exchange	93
9.6	ID 40 objects (manufacturer-specific profile area)	94
9.6.1	Saving objects (saving parameters)	97
9.6.2	MDT user memory (objects 2200–220E)	97
9.6.3	SLK register area (object 2600)	97
9.6.4	MDT register area (object 2800)	97
9.7	Standardized device profile	98
9.8	SDO communication	98
9.8.1	SDO 1	98
9.8.2	SDO 2	99
9.8.3	SDO timeout client (object 2020)	99
9.9	PDO communication	100
9.9.1	PDO mapping	101
9.9.2	PDO communication parameters	103
9.9.3	PDO transmission types	104
9.9.4	Transmit PDO status (objects 2030 – 2033)	104
9.9.5	Receive PDO status (objects 2040 – 2041)	104
9.9.6	SYNC synchronization protocol	105
9.10	CANopen error codes	106
9.11	Emergency object protocol (EMCY)	106
9.12	NMT protocols	106
9.13	Error control protocols	106
9.14	Node guarding protocol	106
9.15	Heartbeat protocol	107
9.16	Boot-up protocol	107
9.17	Communication profile area	107
9.17.1	Device type (object 1000)	107
9.17.2	Error register (object 1001)	107
9.17.3	Manufacturer status register (object 1002)	107

9.17.4	Pre-defined error field (object 1003)	108
9.18	Electronic data sheet (EDS)	108
9.19	Overview of ID 40 CANopen support	108
9.20	Sample applications	110
9.20.1	“Manual workstation” application	110
9.20.2	“Outfeed” application	112
9.20.3	“Workstation” application	113
9.20.4	“Resetting SLK” application	116
9.20.5	Handling “E00” errors	116
10	Web interface	117
10.1	Overview	117
10.2	Requirements	117
10.3	Setting up the network connection to the ID 40/SLK	118
10.3.1	Setting up the connection via serial cable	118
10.3.2	Setting up a network connection	122
10.3.3	Establishing “ID 40” network connection	127
10.3.4	Terminating “ID 40” network connection	129
10.3.5	Troubleshooting PPP connection to ID 40/SLK	129
10.4	HTTP connection to ID 40/SLK	132
10.4.1	Troubleshooting the ID 40/SLK network connection	132
10.5	The ID 40/SLK website	136
10.5.1	Navigating and using the ID 40/SLK website	136
10.5.2	ID 40/SLK homepage	139
10.5.3	MDT register page	140
10.5.4	SLK register page	141
10.5.5	MDT data page	142
10.5.6	Fieldbus settings page	143
10.5.7	Systems statistics page	143
10.5.8	System log and settings page	144
10.6	Tips and tricks	145
10.6.1	Bookmarks	145
10.6.2	Directly retrieving the syslog	145
10.7	Web access from application programs	146
10.7.1	Web interface data formats	146
10.7.2	Web interface write commands	147
10.7.3	Web interface read commands	148
11	Start-up and parameterization	152
11.1	Starting up Profibus DP	152
11.1.1	Configuring through the web interface	152
11.1.2	Configuring with Profibus master	153
11.1.3	Starting the Profibus master	153
11.2	Starting up Interbus	153
11.2.1	Outgoing remote bus configuration	153
11.2.2	Configuring the maximum PDU size of the PCP channel	154
11.2.3	Configuring with the IBS CMD SWT G4 software	155
11.2.4	Showing Interbus status on the display	155
11.3	Starting up CANopen	156
11.3.1	Configuring the bus parameters	156
11.3.2	ID 40/SLK-CAN boot-up behavior	157

11.3.3	Showing CANopen status on the display	157
11.3.4	Configuring the second SDO channel	158
11.4	Adjusting the baud rate of the serial interface	158
12	Diagnostics	159
12.1	Troubleshooting guide	159
12.1.1	Important information that can indicate errors	159
12.1.2	Fatal system errors	159
12.1.3	Diagram of possible causes of errors	161
12.2	Diagnostics using the web interface	162
12.3	SLK software upgrade via serial interface	162
13	Technical data	163
13.1	Cables, connector pin assignments	164
13.2	SLK nameplate	165
13.3	Data transmission times between MDT and controller	166
13.3.1	Transmission times with Profibus	166
13.4	Compatibility	167
13.4.1	Compatibility with older software versions	167
13.4.2	Compatibility with ID 80/E and MTS 2	167
14	Overview for ordering ID 40 modules	168
15	Service and support	170
15.1	Technical support	170
15.2	Internet	170
15.3	Site	170
16	Appendix	171
16.1	ID 40 system link model	171
16.2	Abbreviations and terms	173
16.3	References	175

1 About this manual

1.1 Scope of Application

This manual describes the following system components of the ID 40 identification and data memory system:

- ID 40/MDT2K mobile data tag
- ID 40/MDT8K mobile data tag
- ID 40/MDT32K mobile data tag
- ID 40/SLK-PDP read/write head
- ID 40/SLK-IBS read/write head
- ID 40/SLK-CAN read/write head

The functions described refer to the 4.x versions of the ID 40/SLK software. Refer to the following documents for more information on the ID 40 system:

Table 1: Required and supplementary documentation

Document	Edition	Order no.
ID 40 brochure	en	3 842 528 602
ID 40/SLK assembly instructions	de/en/fr/it/es/pt	3 842 527 942
ID 40/MDT assembly instructions	de/en/fr/it/es/pt	3 842 527 943
Programming manual		
ID 40 function blocks for Rexroth CL and PCL controllers	en	3 842 406 191 *)
Programming manual		
ID 40 function blocks for Siemens S7 controllers	en	3 842 406 190 *)
Operating instructions for ID 40 configuration and diagnostic program	en	3 842 406 119 *)

*) Manual is part of the software CD-ROM and is not available in print form.

1.2 Layout of this manual

This manual is structured so each chapter builds on the previous chapter. It is recommended that readers who are unfamiliar with identification systems read through this entire manual. Readers who have experience with the ID80/E identification system can skip Chapters 3 and 4.

- Chapter 2 contains an introduction to the ID 40 identification system and describes how the mobile data tag and read/write head function.
- Chapter 3 shows the critical steps for installing the ID 40 on transfer systems.
- Chapter 4 describes how the mobile data tag stores information.
- Chapter 5 describes how the read/write head stores information.
- Chapter 6 explains how the read/write head works as well as how data is exchanged via the fieldbuses.
- Chapter 7 describes how data is exchanged via Profibus DP.
- Chapter 8 describes how data is exchanged via Interbus-S.
- Chapter 9 describes how data is exchanged via CANopen.

- Chapter 10 describes the web interface for MDT data diagnostics and start-up support.
- Chapter 11 explains the steps for starting up the ID 40 on fieldbus systems.
- Chapter 12 shows control procedures using some typical applications. The corresponding ID 40 code sequences for the Profibus system are also listed.
- Chapter 13 can be used as a reference for running system diagnostics. It also contains a troubleshooting guide.
- Chapter 14 contains all technical data.
- Chapter 15 contains an overview for ordering available system components.
- The appendix (Chapter 16) contains a glossary of all technical terms and the link model for the ID 40 system.

1.3 Display

1.3.1 Numbers

- Decimal numbers are written without a suffix,
e.g., 123
- Hexadecimal numbers are written with the prefix "0x",
e.g., 0x0002001C
- Data blocks are written with their address and length in bytes as
<address>/<length>,
e.g., 0x00000815/25

1.3.2 Operating states

- The operating states of the read/write head are written in uppercase and in italics,
e.g., *CONNECTED*

1.3.3 Information

Symbol	Meaning
	Information marked with the "Caution" symbol <u>must</u> be strictly observed for your safety.
	Important information is marked with the "Info" symbol. It is intended to provide support and should be observed.
	Individual, independent action
1.	Numbered instruction:
2.	The numbers indicate that the actions must be carried out one after the other.
3.	

1.4 Safety instructions

Intended use	The MDT mobile data tag and SLK read/write head together form the ID 40 identification system and may only be used for this purpose in an industrial setting in accordance with Class A of the EMC Act. The manufacturer's declaration documents the ID 40 identification system's compliance with the requirements of the relevant standards. Country-specific features and regulations have to be additionally taken into account, if applicable.
Safety precautions for normal operation	<p>Only specially trained personnel may install and operate the system. The relevant safety instructions must be observed. In particular, measures must be taken to prevent danger to personnel and equipment in the case of a defect in the identification system. This includes maintaining the permissible ambient conditions and the use of an approved power supply. Details on this can be found in the Technical data chapter.</p> <p>The function of the identification system and all connected components must be checked at regular intervals. If there are any indications that the identification system is not working properly, it must be taken out of operation and secured against unauthorized use.</p>
Instructions for operators	Operating personnel should receive training from supervisors that includes safety precautions and operating the ID 40 in accordance with this manual.
Liability and warranty	The manufacturer does not accept any liability or warranty claims for damages arising from improper use or intervention that is unauthorized or not described in this manual.

1.5 New in the 4.x versions of the ID 40/SLK software

- TCP/IP connection via RS232 interface for supporting web-based system and MDT diagnostics, see Chapter 10 "Web interface".
- Access to MDT and SLK data via web browser, also in Chapter 10 "Web interface".
- Access to MDT and SLK data via user application program using the web interface, see Chapter 10.7 "Web access from application programs".
- The commands for opening and closing transfer are no longer needed, since the presence of the MDT is now monitored. It is no longer possible to switch the MDT unnoticed, see Chapter 6.9 "MDT lifeguarding".

2 Introduction

The ID 40 identification system was designed for automated assembly lines. It functions as an electronic tag and is used to track product- and order-related data with the workpiece on a workpiece pallet.

The ID 40 identification system consists of the following primary components:

- MDT mobile data tag
- SLK read/write head



Fig. 1: MDT mobile data tag and SLK read/write head

2.1 ID 40/MDT mobile data tag

The MDT mobile data tag contains re-writable memory where data can be read and written without contact. It is assigned to an individual workpiece by being mounted on the workpiece pallet and stores the production data for that workpiece.

By mounting the MDT on the workpiece pallet, the flow of information is continuously synchronized with the flow of materials, and the workpiece data can be accessed directly by each process station.

This data includes:

- Workpiece type
- Complete assembly plan
- Production status
- Next processing step

- Serial number
- Order number
- Workpiece pallet number
- Process station settings
- Test results
- Quality data

The MDT is supplied with inductive power by the read/write head and does not use any batteries. The MDT is passive outside of the range of a read/write head.

The latest FRAM memory technology is used in the ID 40/MDT, ensuring large memory capacity at low volume. The FRAM memory stores the data for a virtually unlimited amount of time without additional buffer batteries. Unlike other memory technologies, FRAMs can be re-written up to 10 billion times.

2.2 ID 40/SLK read/write head

Unlike the mobile data tag, the ID 40/SLK read/write head is placed in a single location on the assembly line. With the ID 40/SLK, data is exchanged with the MDT without contact. It is directly connected as part of a fieldbus system and allows a user controller to access the workpiece data. The read/write head comes in Profibus DP, Interbus-S and CANopen versions.

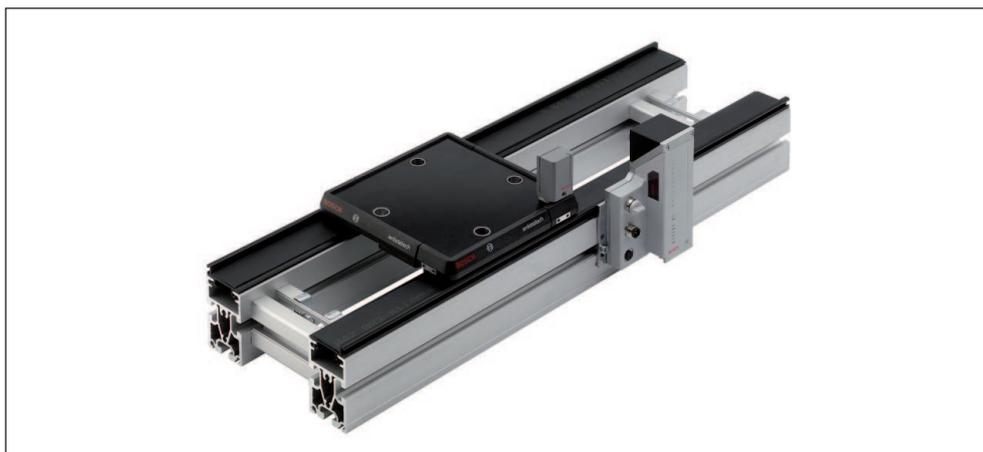


Fig. 2: ID 40 on TS 2plus transfer system

The SLK sends data from the fieldbus master to the MDT or, when given a read command, from the MDT to the fieldbus.

Parameters can also be set for data transfers, wherein write and read commands are saved to the SLK to be automatically executed when the next MDT arrives. Parameters are typically set for the SLK when the system is initialized, but can be set at any time as long as an MDT is not in the HF field of the SLK, or the HF field is off. The SLK generates a high-frequency alternating field (HF field) that is needed to provide the MDT with inductive power. No-contact data transmission with the MDT also occurs through this HF field, which is generated using a ferrite antenna under the dark, plastic cover.

The cover of the HF head lights up once data transmission with an MDT has initiated. The light deactivates once the control system completes the data transmission with the MDT.

The LED on the MDT lights up once it is in the HF field. The color of the LED indicates various operating states of the MDT.

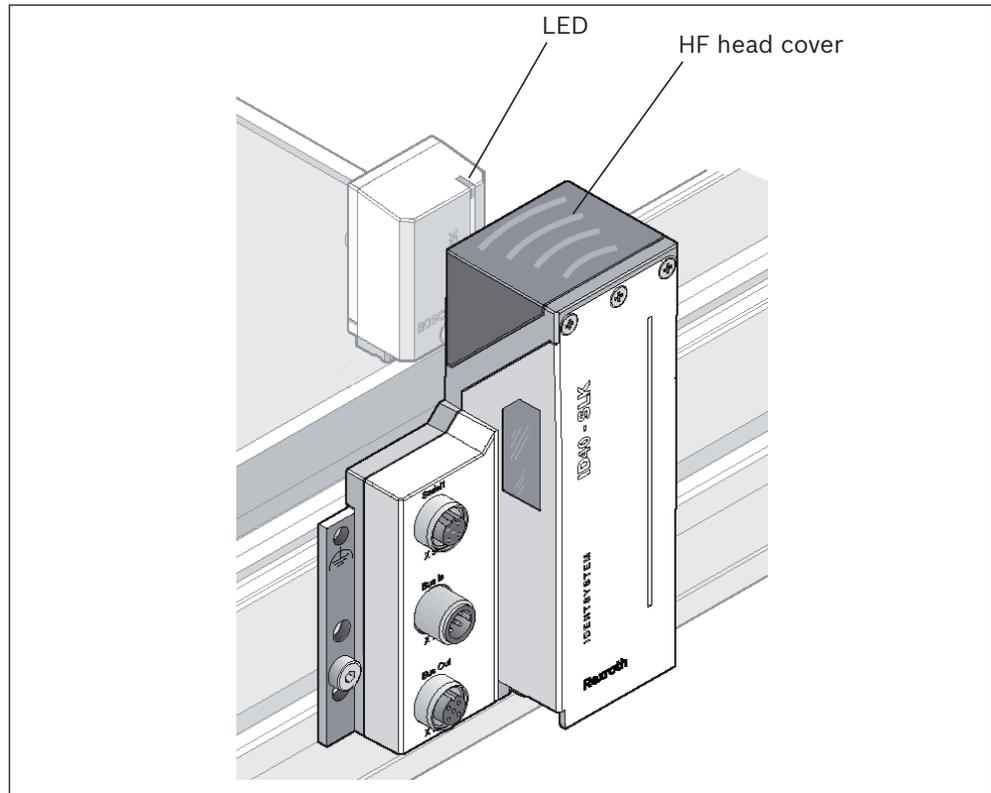


Fig. 3: Data transmission between SLK and MDT

2.2.1 SLK operating states

The SLK has multiple operating states, known as link states. The actual link state can be changed by a fieldbus command from the user program as well as by the ID 40 system itself.

The actual link state can be determined by the control system at any time using the fieldbus (for coding, see Chapter 5.3.1 "Actual link state"). An MDT entering the HF field, for example, is signaled by a change in the actual link state. The SLK also shows the actual link state on the status display.

The operating states of the SLK are described below. Appendix 16.1 "ID 40 system link model" contains a detailed overview of the link state.

2.2.1.1 Disconnected

DISCONNECTED is the base state once the SLK is activated and the self-test has been completed. The SLK's HF field is not active. No data can be transmitted between the SLK and an MDT. The SLK status display shows **off**. The SLK is ready to communicate with the control system via the fieldbus.

2.2.1.2 Connecting

The SLK only switches from the operating state *DISCONNECTED* to *CONNECTING* when commanded by the control system. The SLK's HF field is activated. The SLK is ready to receive. The SLK status display shows **wai**. The next MDT that enters the field automatically signs on to the SLK and is ready to communicate.

2.2.1.3 Preconnected

This state comes after the *CONNECTING* state and automatically runs once an MDT enters the SLK's field, i.e., has signed on to the SLK. The SLK executes data transfers parameterized beforehand through the fieldbus (prefetch and pretransmit). The status display shows **pre**. Once all pre-defined transfers are complete, the link state *PRECONNECTED* ends automatically. *PRECONNECTED* can end very quickly, so the **pre** signal on the display is not always visible. If the MDT in the *PRECONNECTED* state leaves the HF field early, the SLK switches to the *ERROR* state.

2.2.1.4 Connected

In the basic configuration, the SLK switches from *PRECONNECTED* to *CONNECTED*. The status display shows **con**. The control system can now directly access the MDT data. To do this, read/write commands are sent to the SLK through the fieldbus that are executed immediately. Data read from the MDT is directly transmitted to the control system.

The control system ends the *CONNECTED* state with a command to switch either to *DISCONNECTED* or *CONNECTING*. Only then can the MDT leave the HF field. If the MDT left the HF field early, the SLK switches to the *ERROR* state.

The SLK can also be configured so the *CONNECTED* state does not activate and automatically switches back to *CONNECTING* or *DISCONNECTED* after completing *PRECONNECTED*. This can be done by configuring “auto mode” (see Chapter 5.3.3 “Auto mode”).

2.2.1.5 Error

The read/write head switches to the “Error” state when it is unable to communicate with the MDT after several attempts, e.g., because the MDT has already left the HF field during a *DISCONNECT* request. The status display shows a flashing **E00**. This can occur, e.g., when the MDT moves past the SLK too quickly, or when too much data was read or written.



The *ERROR* state can be ended by switched to the *DISCONNECTED* or *CONNECTING* states.

The SLK does not switch to the *ERROR* state in the event of communication interruptions while exchanging data with the MDT. This kind of error information is sent by the system through the fieldbus.

2.2.1.6 Busy

The read/write head is temporarily processing link state switching requests. The status display may show **bsy**.

2.2.1.7 Transitions between link states

The link state switches either by a request from the control system or when an event defined in the following table occurs.

Operating state switches due to a **command** (commanded link state) from the control system (PLC):

Table 2: Relationship between commanded link state and link state

Commanded link state to SLK	Only permitted in link state (display)	Commanded link state results in link state (display)
CONNECT	<i>DISCONNECTED (OFF)</i> <i>ERROR (E00)</i>	<i>CONNECTING (WAI)</i>
DISCONNECT	<i>CONNECTED (CON)</i> <i>CONNECTING (WAI)</i>	<i>DISCONNECTED (OFF)</i>
RECONNECT	<i>CONNECTED (CON)</i>	<i>CONNECTING (WAI)</i>
ERROR	<i>DISCONNECTED (OFF)</i> <i>CONNECTING (WAI)</i> <i>CONNECTED (CON)</i>	<i>ERROR (E00)</i>



After every commanded link state, the system checks whether or not the actual link state corresponds to the expected state. Only then can the next step in the process be executed.

Operating state switches due to an **event****Table 3: Relationship between events and link states**

Event during communication	Event only possible in link state (display)	Results in link state (display)
MDT signs on to SLK	<i>CONNECTING (WAI)</i>	<i>PRECONNECTED (PRE)</i>
<i>PRECONNECTED</i> ends and auto reconnect is active (see Chapter 5.3.3 "Auto mode")	<i>PRECONNECTED (PRE)</i>	<i>CONNECTING (WAI)</i>
<i>PRECONNECTED</i> ends and auto disconnect is active (see Chapter 5.3.3 "Auto mode")	<i>PRECONNECTED (PRE)</i>	<i>DISCONNECTED (OFF)</i>
<i>PRECONNECTED</i> ends and auto mode is not active (see Chapter 5.3.3 "Auto mode")	<i>PRECONNECTED (PRE)</i>	<i>CONNECTED (CON)</i>
Error during MDT sign-on	<i>CONNECTING (WAI)</i>	<i>ERROR (E00)</i>
Error during direct data exchange with MDT	<i>CONNECTED (CON)</i> Note: The error is output through the bus in the form of an error message and does not alter the state	<i>CONNECTED (CON)</i>
Error during MDT sign-off	<i>PRECONNECTED (PRE)</i> <i>CONNECTED (CON)</i> Note: Can occur during commanded link states DISCONNECT, RECONNECT, ERROR or when <i>PRECONNECTED</i> ends and auto mode is active (see Chapter 5.3.3 "Auto mode")	<i>ERROR (E00)</i>
MDT leaves early	<i>CONNECTED (CON)</i> Note: Can occur when a workpiece pallet is released too early in the application.	<i>ERROR (E00)</i>

Appendix 16.1 "ID 40 system link model" describes the states and state transitions in detail.

2.2.2 Status display

The 4-character status display of the ID 40/SLK shows the operating states of the SLK, error messages and the status of the fieldbus interface. After the SLK is turned on, the currently configured fieldbus address (node no. depending on fieldbus system) appears on the display for a few seconds.

While running, the display is divided into two areas: Area 1 contains the first three segments on the left, and area 2 the first segment on the right.

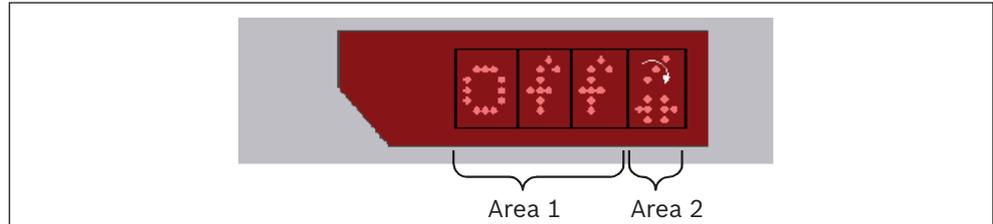


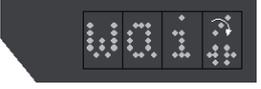
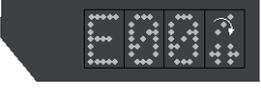
Fig. 4: ID 40/SLK read/write head status display

Area 1 shows the operating state of the SLK or an error code in the event of an error. Area 2 shows when the SLK is ready as well as the status of the fieldbus connection to the user controller.

In the **Interbus version**, area 2 is divided differently, see Chapter 11.2.4 “Interbus status display”.

The following table provides an overview:

Table 4: Overview of ID 40/SLK read/write head displays

Display range	Displaying	Remark	Example
Area 2	System activity	The SLK operating system is active when the bar in the fourth display segment rotates.	
Area 2	Fieldbus activity	The bottom half of the display segment contains symbols indicating the status of the fieldbus. Their appearance and meaning depend on the fieldbus. See Chapters 8 to 10 for more details.	
Area 1	Actual link state	See Chapter 2.2.1 "SLK operating states"	   
Area 1	Error state with error number	See Chapter 12.1 "Troubleshooting guide" for meaning	
Area 1 and area 2	Fieldbus node number	Only shown during boot up and consists of the fieldbus code letter and three-digit node number	
Area 1 and area 2 flashing	Fatal error, system stopped	See Chapter 12.1.2 "Fatal system errors" for meaning	

2.3 Data transmission between SLK and MDT

The data transmission between the SLK and the MDT is contactless and occurs via a high-frequency electromagnetic field (HF field). The MDT is also powered by this electromagnetic field. To begin exchanging data, the mobile data tag is moved into the field of the read/write head.

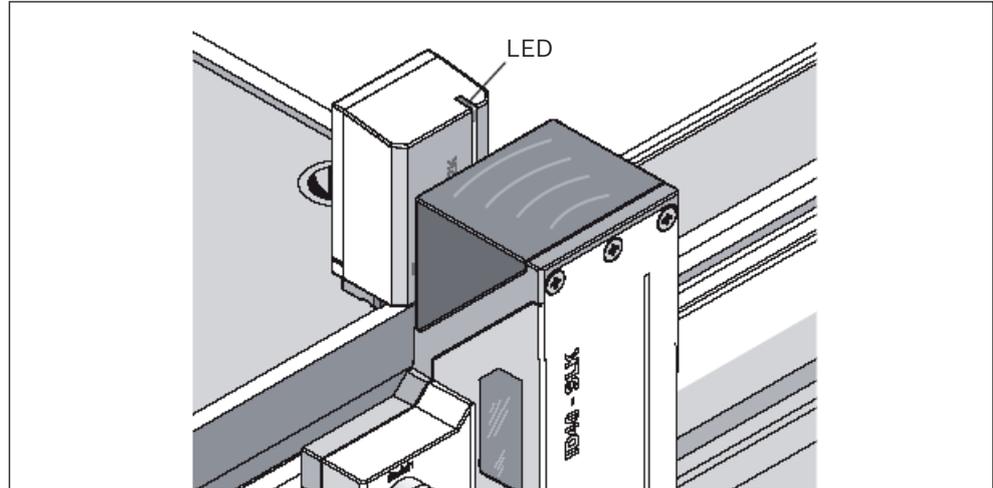


Fig. 5: Display indicating mobile data tag is ready for communication

The LED on the front edge of the housing lights up orange when the MDT enters the SLK's field. Once a secure data transmission is possible between the MDT and SLK, the color changes. When the LED shows green, the MDT is operating correctly. When the LED shows red, an error occurred during the last data exchange (see Chapter 4.2.3.1 "MDT status register").

When the data exchange between the SLK and the MDT ends, the LED color changes to orange, then turns off once the MDT leaves the HF field.

2.3.1 Position of MDT and SLK during data transmission

Proper exchanging of data always depends on the alignment of the MDT and SLK to one another, and on the read/write distance. MDT alignment consists of frontal and lateral alignment.

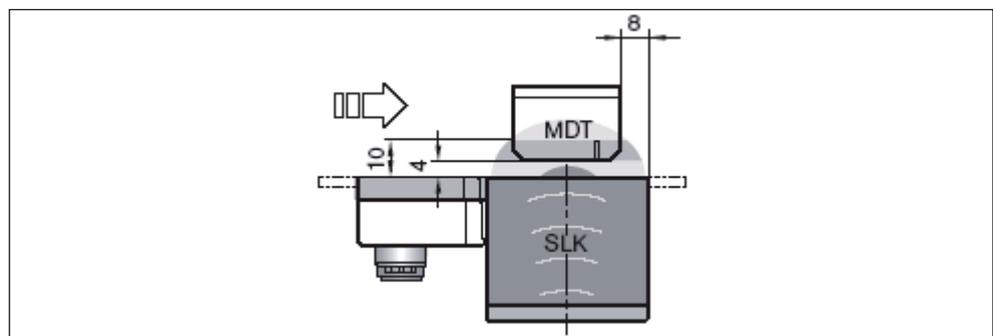


Fig. 6: Alignment of ID 40/MDT for frontal reading and writing

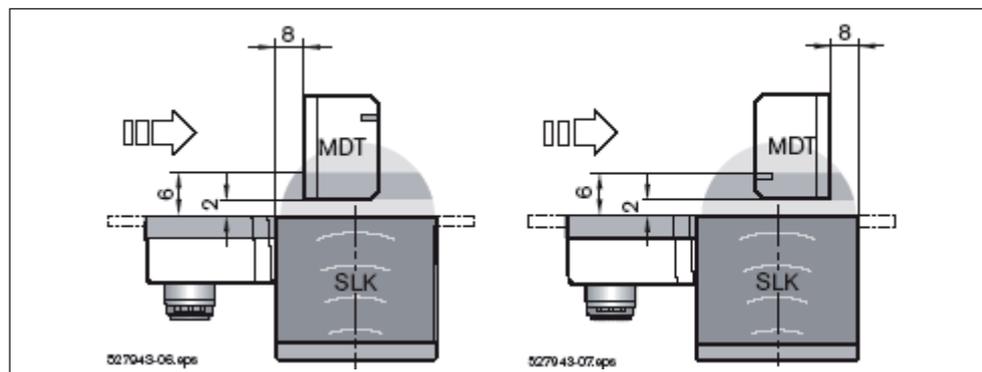


Fig. 7: Alignment of ID 40/MDT for lateral reading and writing

The read/write distances are listed in the following table in relation to the states and alignment, and are valid to the following limits:

- Max. height difference of ± 5 mm between MDT and SLK.
- Max. lateral difference of ± 10 mm between centerlines of MDT and SLK

Table 5: Distances between ID 40/MDT and ID 40/SLK for secure data transfer

Read/write distance	Frontal alignment		Lateral alignment	
	Min.	Max.	Min.	Max.
Static	4 mm	12 mm	2 mm	7 mm
Dynamic	4 mm	10 mm	2 mm	6 mm

With frontal alignment and a nominal distance of 4 mm, dynamic reading and writing is possible over a 40 mm section.

2.3.2 Static and dynamic data transmission

Data can be transmitted between the MDT and the SLK both statically and dynamically. In **static** mode, the MDT comes to a stop directly in front of the SLK. In this instance, the dwell time can be as long as desired, i.e., data can be exchanged for an unlimited amount of time. However, the MDT must remain in range until communication is complete.

In dynamic mode, the MDT moves past the SLK during communication. In this instance, the dwell time in the field is determined by the transport speed. The data quantity being transferred has to be adjusted to the dwell time by programming the user controller.

The MDT can approach and pass the SLK in different **directions**:

- **Horizontally**, as shown in Fig. 6. This is the optimal mode.
- **Vertically**. Here the dwell time in transit is limited.
ID 80/E-MDTs can experience sporadic communication errors, which is why this direction is only recommended for ID 40/MDTs.
- **Frontally**. The MDT approaches the SLK from the front and must be stopped within the specified distance range.

2.3.3 Direct and parameterized data transmission

The data on the mobile data tag can be accessed directly as long as the MDT is located within the HF field of the read/write head. To do this, the control system sends read and write commands to the read/write head through the fieldbus. The read/write head executes these commands immediately and reports the result to the control system.

For parameterized data transmissions, the control system sends the read and write commands to the read/write head when there is no mobile data tag in the HF field. These commands are buffered in the read/write head and automatically executed when an MDT enters the HF field. These commands are divided into those executed only once and those executed for each additional MDT.

Parameterized data transmissions are particularly quick, since the read and write commands are already stored in the SLK. This makes them best suited for dynamic reading and writing.

Direct and parameterized data transmissions can also be combined. When an MDT enters the HF field, all parameterized commands are executed first, followed by the direct data transmission.

2.3.4 Securing data transmission

The data between the SLK and MDT is sent and received wirelessly via an HF field. Checksums are used to monitor the transmission of data without errors. To do this, each data packet contains redundant parity bits, known as longitudinal and transverse parities. If the system detects a transmission error, the sent data packets are resent to ensure secure communication between the SLK and the MDT.

The MDT is also sent a special data package, known as the **sign-off sequence**, to indicate the end of the data transmission. If the MDT leaves the HF field of the SLK without signing off first, a **communication error** is entered in the MDT status register. The SLK itself registers the MDT's unexpected departure with a corresponding error state (see Chapter 6.9 "MDT lifeguarding").



If no data on the MDT was altered, an error is not registered even though communication is interrupted unexpectedly.

3 Installing MDT and SLK

3.1 Mounting the MDT on the workpiece pallet



Please observe ID 40/MDT assembly instructions 3 842 527 943.

Prior to mounting the MDT on a workpiece pallet, drill two M3 threaded holes into the carrying plate using the included drilling template. Now attach the appropriate mounting base onto the workpiece pallet. Mounting bases are included for all standard workpiece pallets for Bosch Rexroth transfer systems (TS 1, MTS 2, TS 2 plus and TS 4 plus) as well as for any workpiece pallet with a level carrying plate. Slide the MDT into the mounting base guide and fasten it down with the included screw. This ensures the MDT is mounted in stable fashion.



Do not use the MDT as a “handle” for the workpiece pallet. The weight of the workpiece pallet can damage the fastener or the MDT itself.

The MDT can be mounted onto the workpiece pallet plate in various positions. This flexibility arises from the fact that the MDT is readable from three sides (front, right and left). However, the vertical axes for both the SLK and the MDT have to run parallel to each other. The gap dimensions (or read/write distance) between the MDT and the SLK have to be observed for proper data transmission (see Chapter 13 “Technical data”).

The MDT can also be mounted on the underside of the workpiece pallet. When doing so, make sure that the MDT cannot come into contact with any section elements while the workpiece pallet is in motion. This can damage the MDT.

3.2 Mounting the SLK on transfer section profiles

Use the included mounting kit to mount the SLK on the Bosch section profiles ST 1, ST 2 and ST 4. This will ensure the specified distance between MDT and SLK. An illustrated assembly guide is also included.

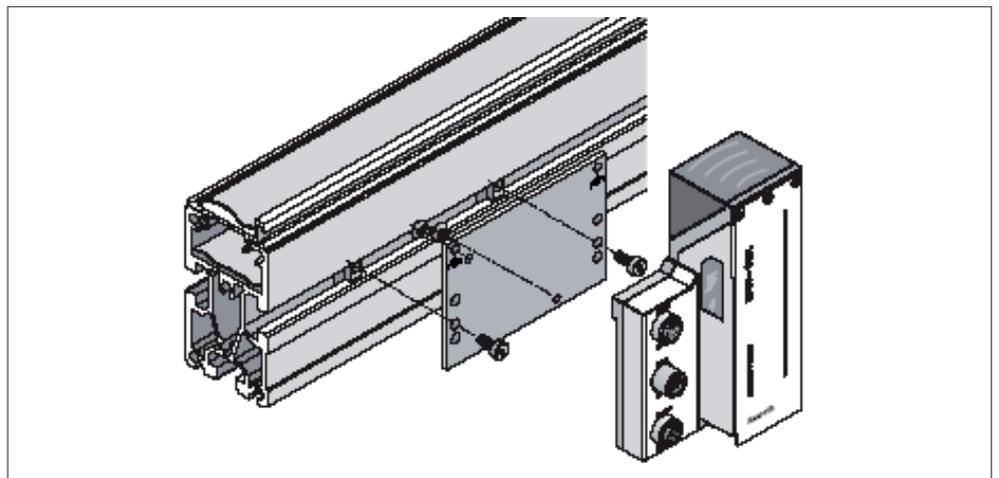


Fig. 8: Mounting the ID 40/SLK on an ST 2 section profile

When mounting the read/write head, make sure there are no metallic substances within 10 mm of the head. Metal in the vicinity of the SLK's antenna would greatly reduce the potential transmission distance between the SLK and the MDT or prevent the transmission of data altogether.

3.3 Antenna orientation

The read/write head contains a ferrite antenna under the dark, semi-transparent cap on the mounting frame of the SLK. The antenna generates the electromagnetic HF field for exchanging data with the MDT.

For data transmission without interruption, the antenna and the MDT must be within a certain distance from one another (see Fig. 9).

The antenna comes mounted such that the SLK can communicate with the MDT on the workpiece pallet given standard installation on the section profile of Bosch transfer systems.

To adjust to special mounting situations, the antenna can be carefully removed and replaced 90° vertically by removing the cap. This could be necessary if the MDT is mounted on the underside of a workpiece pallet and the SLK needs to be mounted in a space-saving vertical position.

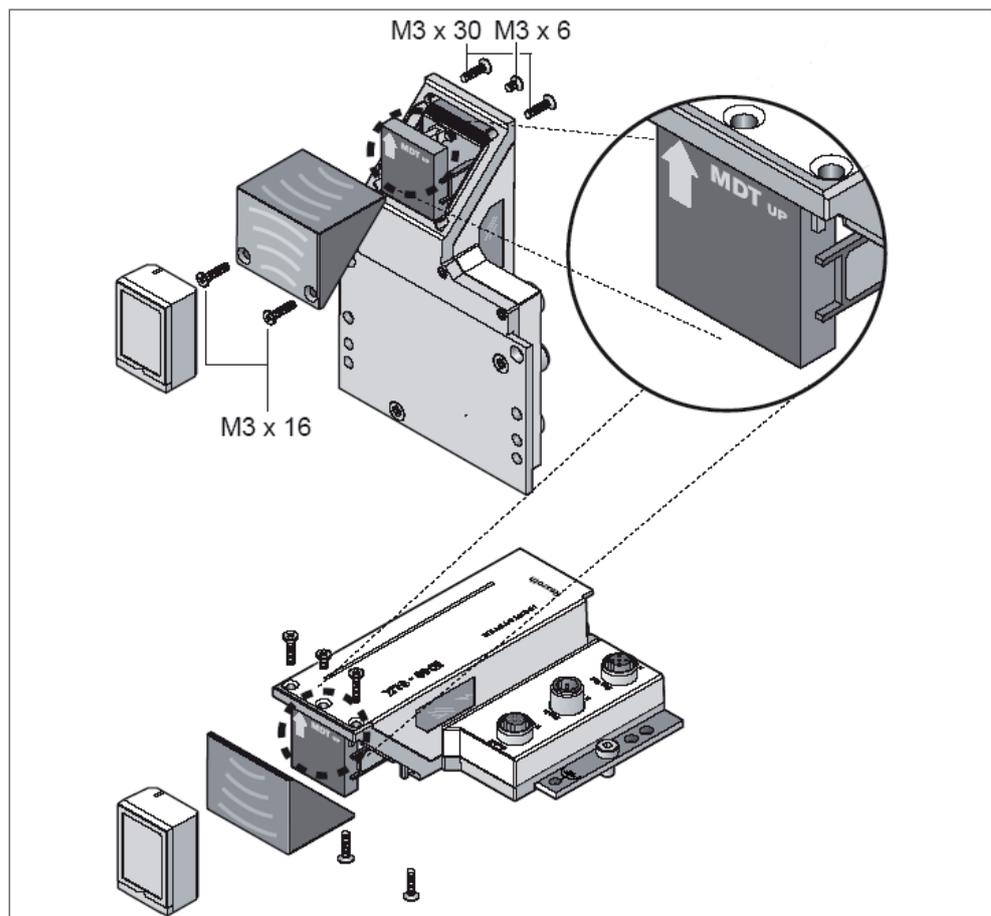


Fig. 9: Orienting the ID 40/SLK read/write head HF antenna

The HF antenna can not only be pivoted upward or outward, it can also be rotated. This makes various mounting situations possible.

The orientation of the HF antenna and the MDT must be the same, otherwise no communication is possible with the MDT. The arrow stamped on the HF antenna points toward the top of the MDT. The status LED is located on the upper edge of the housing.



Carefully remove the HF antenna without damaging the connection cable. Be sure to observe the orientation of the HF antenna as indicated by the arrow.

3.4 SLK electrical connection

3.4.1 Supply voltage

Insulation displacement is used to provide the 24 V supply voltage as per EN 61131-2. This is also used for AS-i bus systems, so a black cable has to be used for the connection as per AS-i specification.



Follow the ID 40/SLK assembly instructions.

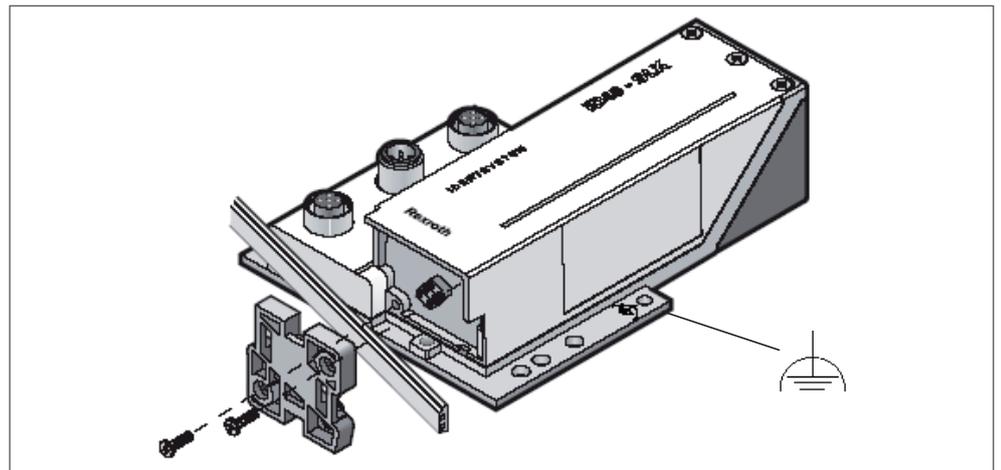


Fig. 10: ID 40/SLK read/write head power connection



Never connect the yellow AS-i signal cable on the SLK, as this can result in the ID 40 system and AS-i bus malfunctioning.

The functional grounding connection is optional and has to be connected for conveyor sections that do not ensure electrical contact when mounting the SLK. Functional grounding is typically unnecessary in Bosch Rexroth transfer systems, as the mounting kits that come with the SLK provide enough grounding to the section profiles.

3.4.2 Connecting the fieldbus

The fieldbus is connected through the corresponding **Bus In** and **Bus Out** M12 sockets. The mechanical coding of the plugs largely prevents any confusion.

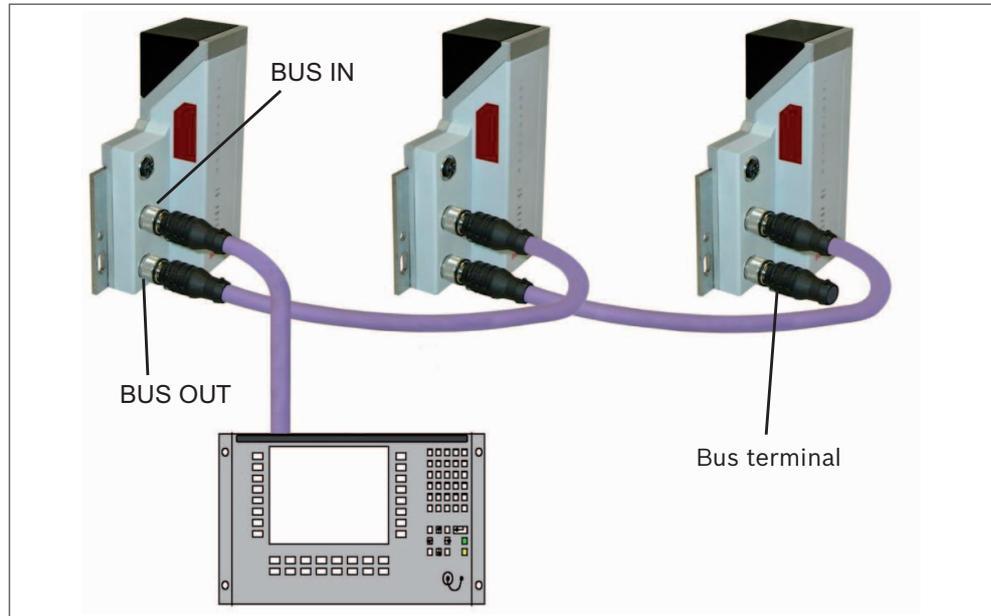


Fig. 11: Connecting the ID 40/SLK to the fieldbus

For nearly all fieldbus systems, a bus connection has to be inserted in **Bus Out** when the SLK is the last bus user.



Do not forget the bus connection on the last user, otherwise the entire bus will not function.

For Interbus, the bus connection can also be configured through a serial interface. In this instance, insert the included cap in the **Bus Out** socket to ensure the protection class.

3.4.3 Serial interface

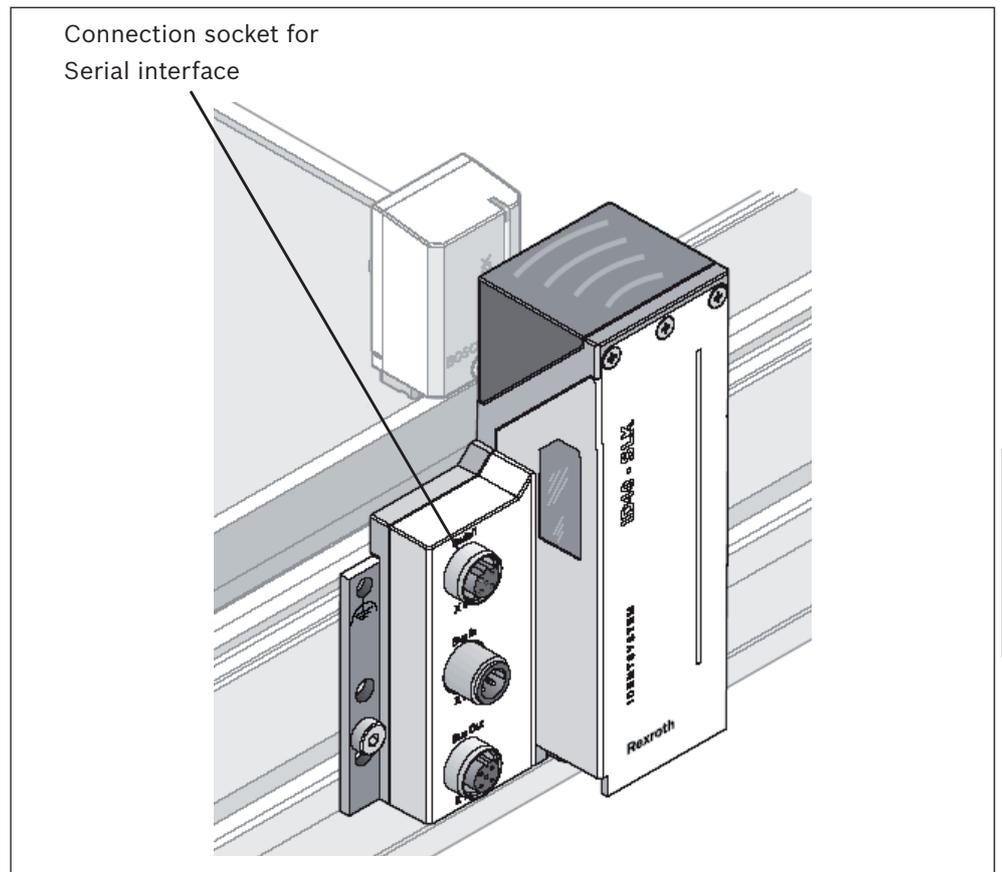


Fig. 12: ID 40/SLK serial interface

The RS232 socket on the SLK can be directly connected to the COM port of a PC or laptop with a diagnostic cable (accessory 3 842 406 117). Modern PC systems sometimes do not have an RS232-COM port. In this instance, an RS232 USB adapter can be used to connect the ID 40/SLK. These adapters do not come with the ID 40 system and can be purchased from a retailer.



Note the PC's protective disconnector when connecting the RS232 to the PC, otherwise current loops can occur through the grounding conductor (GND).

The ID 40 system offers a web server with SLK software versions 4.00 and higher (see Chapter 10 “Web interface”). The browser on a connected computer can be used to configure critical parameters for the fieldbus connection, such as node numbers or baud rates, and read and write **MDT data** - even while exchanging data through the fieldbus.

If the serial interface is not needed, screw the included **cap** onto the socket to ensure the protection class.

3.4.4 Turning on the SLK

Once the SLK has been mechanically and electrically installed, the **supply voltage** can be switched on. The SLK performs a self-test on start-up (approx. 15 sec.).

During this time, the cover on the HF head illuminates, though the status display remains inactive. After the self-test, the status display will first show the currently configured node number for a few seconds, followed by the “ready” signal. The light on the HF head then turns off. The initial state after power-on is always “off”, i.e., the HF field is off and no communication is possible with the MDT.



Fig. 13: Display showing current node number on a Profibus SLK



Fig. 14: Display showing ready

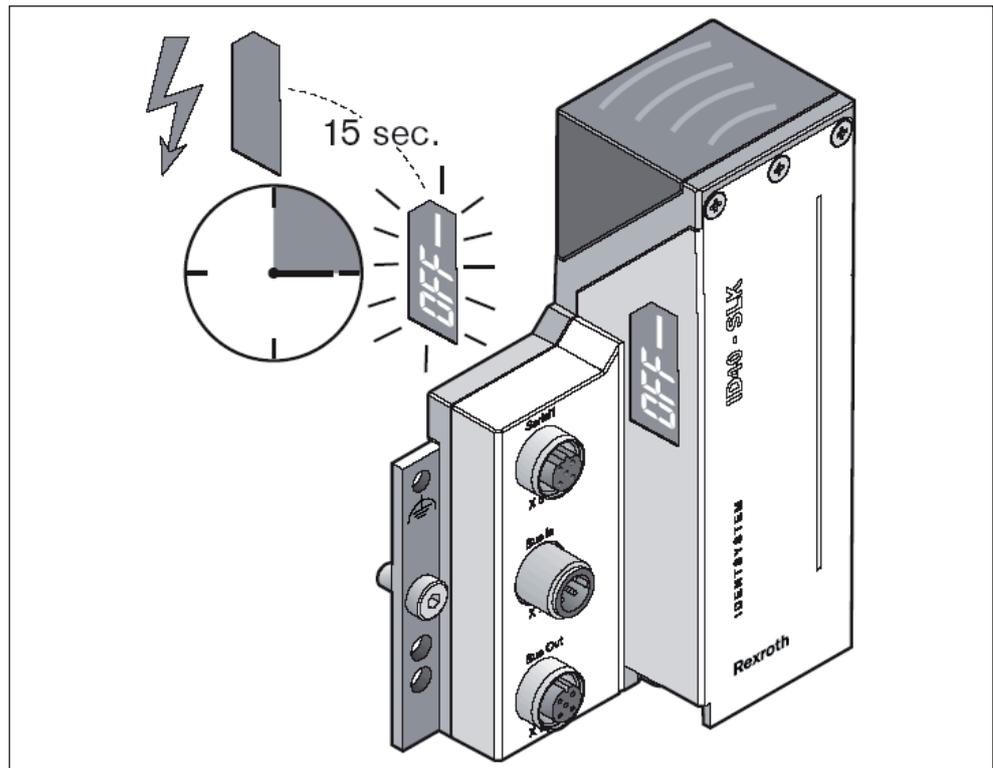


Fig. 15: Approx. 15 seconds until ready after turning on

4 MDT memory structure

4.1 ID 40/MDT storage

The ID 40/MDT is available in three variants:

- **ID 40/MDT2K** with a user data volume of 1,904 bytes
- **ID 40/MDT8K** with a user data volume of 7,664 bytes
- **ID 40/MDT32K** with a user data volume of 30,800 bytes

The functionality as well as electrical and mechanical properties are identical. Outwardly, the variants only differ in labeling and name plate.

4.2 Organization of MDT memory

The MDT memory is divided into three areas: user data, system data and registers. User data and the registers can be accessed directly.

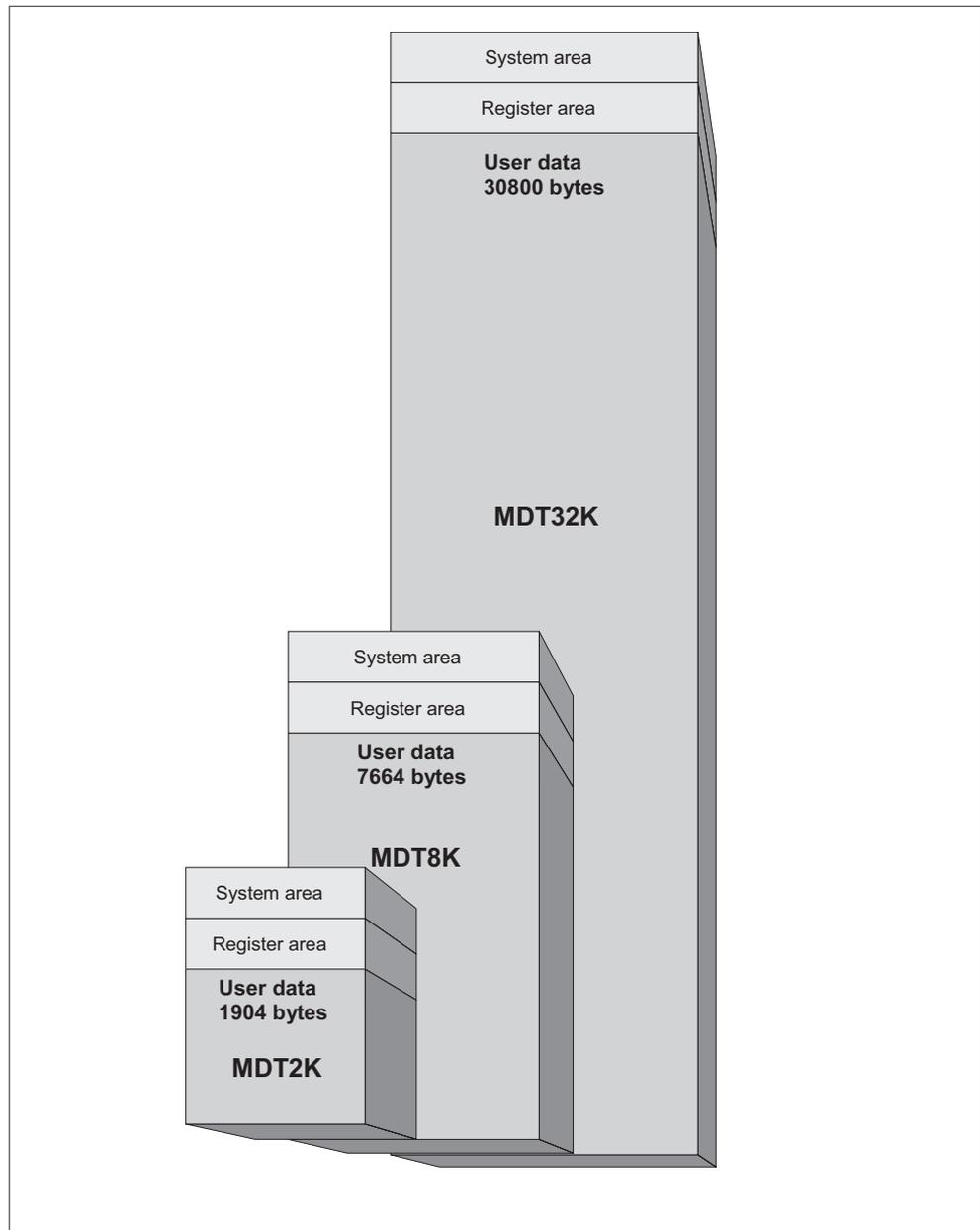


Fig. 16: Memory structure of the mobile data tag

4.2.1 MDT user data area

The user data area contains all production-related and product-related data.

The following applies for this area:

- The user has direct write and read access
- The data is organized in a byte array with 16-bit index, the **MDT address**. The MDT address is presented in this manual in hexadecimal format. The first byte in the user data area has the MDT address 0x0000.
- The smallest readable/writable data volume is 1 byte.
- The user data area in the **MDT2K** is **1,904 bytes**, in the **MDT8K** **7,664 bytes**, and in the **MDT32K** **30,800 bytes**. The remaining memory is reserved for registers and system data.
- The contents of the data are transparent, i.e., not interpreted by the SLK. This means the user can generate any data structure on the MDT.

4.2.2 MDT system data area

Checksums are stored in this area and the user does not have access.

All data stored on the MDT is secured with checksums.

The user data is secured in blocks, i.e., each data block is assigned a checksum. The block size is 16 bytes. The advantage of this is that the checksums do not take up much memory and memory errors can be localized precisely. Data not located in a block flagged with an error can continue to be used in the application.

The 16-byte blocks always begin on an address divisible by 16.

For example, the first three 16-byte blocks in the MDT memory:

Table 6: Segmentation of the user data area

MDT addresses	0x002F	User data	Checksum 3		
	0x0020		User data	User data	Checksum 2
	0x001F				Checksum 1
	0x0010				
	0x000F				
	0x0000				

The checksums are verified on a 16-byte block with each read or write access. If an error is detected, attempting to read this data results in an error sent to the SLK. The data itself flagged with the error is also returned during read access. This allows the user to assess the plausibility of the contents of the data.

An error bit is also entered in the MDT status register. This causes the status display of the MDT to show red.

4.2.3 MDT register area

The register area contains the MDT status register, the MDT pointers and the MDT ID code.

4.2.3.1 MDT status register

The MDT status register consists of two bytes.

The **first MDT status byte** contains the error flags and MDT type code according to the following table:

Table 7: Meaning of bits in MDT status byte

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Checksum error	MDT type code			Pointer error	Memory error	Communication error	reserved

Bit 0: reserved in the ID 40, value = 0. In the ID 80/E data carriers, bit 0 indicates battery errors when set.

Bit 1: set when the data transmission to the read/write head was unexpectedly interrupted. Cause: the MDT is no longer in the field of the SLK.

Bit 2: set for checksum errors in the user data area.

Bit 3: set for checksum errors in pointers 1 to 3.

Bits 4 to 6: coding for the **MDT type**:

000: 8 kB memory → ID 80/E - first generation MDT

001: 80 byte memory → ID 80/E - MDT80

011: 2 kB memory → ID 40/MDT2K

100: 8 kB memory → ID 40/MDT8K and ID 80/E - MDT8k

101: 32 kB memory → ID 40/MDT32K

Bit 7: set when at least one of bits 1 to 3 is set

The **second MDT status byte** is reserved in the ID 40 and has a value of 0xF0.

For ID 80/E data carriers, the voltage of the buffer battery is coded in this status byte.

The first MDT status byte is an error display and can be reset to any value through write access. Previously set error flags (bits 1 to 3 and bit 7) are deleted, the MDT type code remains unchanged.

If a memory error has been indicated (bit 2), the entire 16-byte block containing the error has to be formatted or written. This re-establishes the integrity of the memory block. Begin formatting on an address divisible by 16, e.g., 0x0130.



Resetting the MDT status only deletes the error flags and does not re-establish memory integrity.

4.2.3.2 MDT pointers

There are four **MDT pointers**. Each pointer contains 16 bits and has a value range of 0x0000 to 0xFFFF.

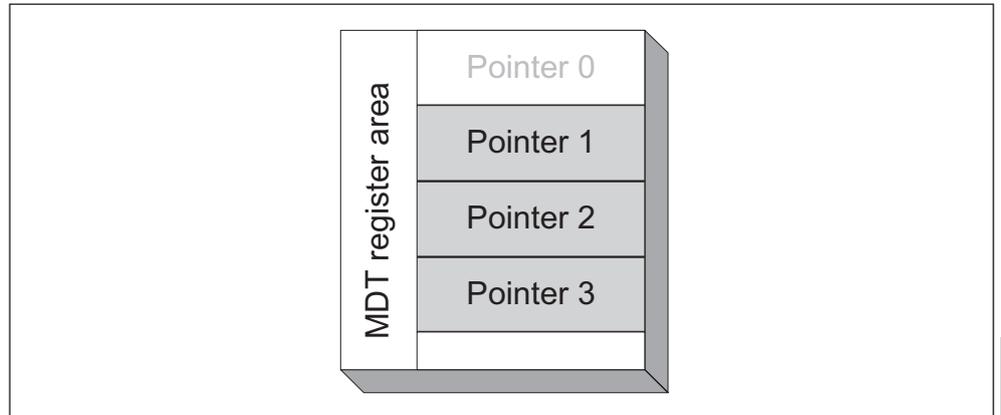


Fig. 17: Pointers in MDT register area

Pointers 1 to 3 can be accessed by the user and are used to save MDT addresses. Pointer 0 is used by the system.

This MDT address can be used in an application to determine the beginning of a data block on the MDT. The pointer first has to be read in order to read or write the MDT data block. The content of the pointer is then used to access the MDT data block.

4.2.3.3 MDT ID code

The **MDT ID code** contains a unique 32-bit number for each ID 40/MDT. The MDT ID code cannot be edited and can only be read.

The MDT ID code allows an MDT - and thereby a workpiece pallet - to be uniquely identified on a production line.

5 SLK memory structure

The SLK memory consists of the following areas:

- **Map** of the **MDT user data area**. The user data of the MDT currently in the HF field can be read and written through this area.
- **Map** of the **MDT register area**. Read/write access to the registers of the MDT currently in the HF field occurs through this area.
- **SLK register area**. This area contains all system variables used to control how the SLK functions.
- **Data buffer** for buffering commands and data for parameterized data exchange.

The individual areas are generally addressed with what is referred to as the SLK address and depends on the fieldbus system used. Addressing the memory areas is described in detail in Chapters 7 “Profibus DP”, 8 “Interbus” and 9 “CANopen”.

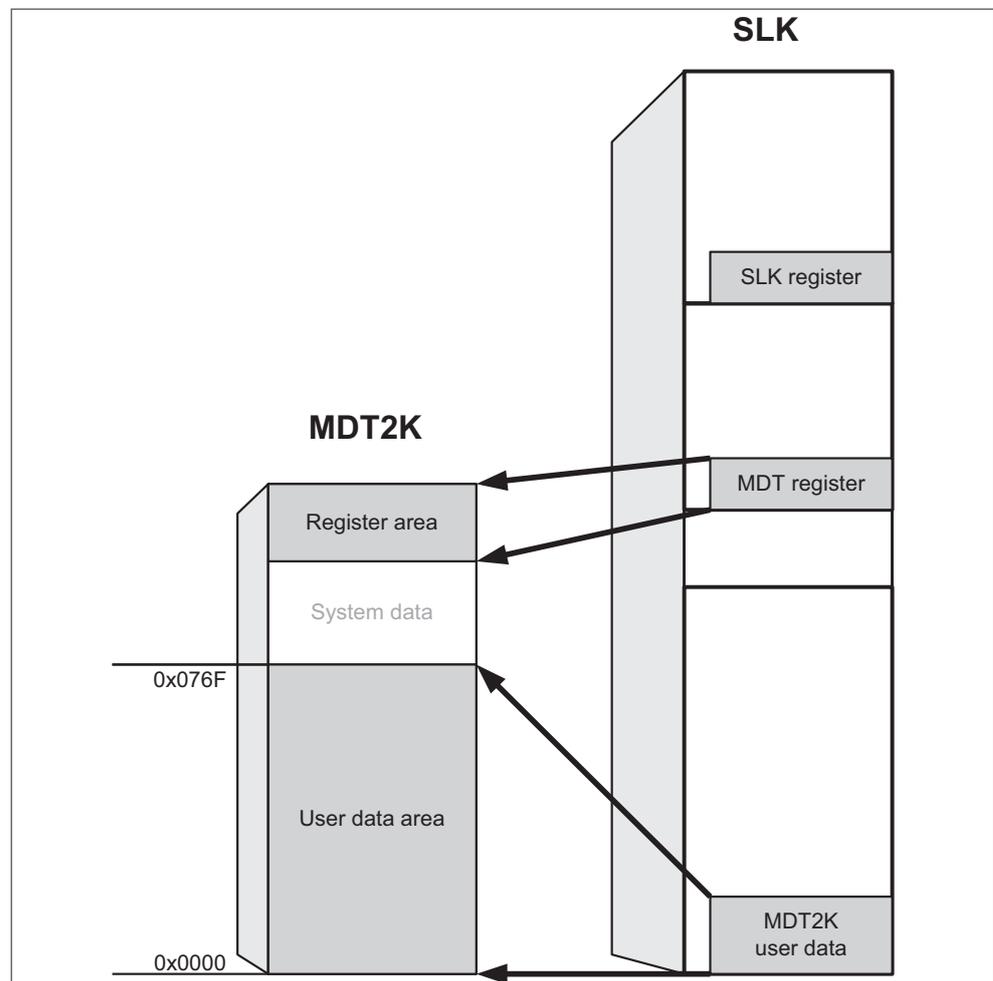


Fig. 18: Memory allocation between MDT and SLK

5.1 Map of MDT user data area

The MDT user data is accessed through its map in the SLK. Data read from this SLK memory area comes directly from the MDT. The SLK automatically creates a copy of the MDT user data the first time this area is accessed. Data written to this area is simultaneously written to the MDT.

5.2 Map of MDT register area

The MDT registers are accessed through their map in the SLK.

5.2.1 Map of MDT status register

Both MDT status bytes are read in accordance with Chapter 4.2.3.1 “MDT status register”.

The MDT status can be reset by entering any value to the map in the SLK.

5.2.2 Map of MDT pointer registers

The pointer values for the MDT can be read and modified using the map of the MDT pointer registers (Chapter 4.2.3.2 “MDT pointers”).

The following pointer functions are possible:

- Enter a 16-bit value in a pointer.
- The pointer values are shown in “Motorola” format, i.e., the high byte appears on the low SLK address.
- Read pointer contents through read access to map.
- All three pointers can be both read and written by accessing the pointer map once.

5.2.3 Map of MDT ID code

The MDT ID code (Chapter 4.2.3.3 “MDT ID code”) is read using the map. Write access to this data block returns an error message, since the MDT ID code cannot be edited.

5.2.4 MDT counter

The MDT counter can be used to count the workpiece pallets that pass by the SLK. The SLK increases the MDT counter by one every time an MDT signs on to the SLK. The 32-bit counter can be both read and written. Write access allows the counter to be deleted (i.e., reset to 0) or set to any starting value. The counting range spans from 0 to 4,294,967,295.

Reading or writing the value does not access the MDT. The counter is not stored on the MDT.

5.2.5 MDT formatting

The MDT formatting fills one data block in the MDT user memory with a constant value. This can be required in an application if, e.g., the content of the MDT should be deleted or set to a specific value at a start station.

The MDT is formatted by write accessing the byte with the formatting value (byte 5). The two other formatting parameters (start address and length of data block) have to be configured at this point. It is recommended to send all parameters in one command.

Table 8: Meaning of formatting parameters (5 bytes)

Byte	Parameter	Formatting parameter	Value ranges
High byte (1) Low byte (2)	Start address	MDT address from which the memory should be filled (2 bytes)	0x00–0x076F (MDT2K) 0x00–0x1DEF (MDT8K) 0x00–0x784F (MDT32K)
High byte (3) Low byte (4)	Length	Formatting range size (no. bytes to fill)	0x00–0x0770 (MDT2K) 0x00–0x1DF0 (MDT8K) 0x00–0x7850 (MDT32K)
Byte (5)	Data byte	Formatting value with which the memory range should be filled	0x00–0xFF

The formatting function only writes to the user memory of the MDT; the MDT register area is not altered.

If the selected memory range exceeds the memory capacity of the MDT, the “Communication interrupted” error is generated. In this instance, the contents of the MDT memory can be modified.

When reading the formatting parameters, the last parameter used is returned. The MDT is not accessed.

5.2.5.1 Special functions

If the entire memory range of the MDT should be formatted to the value in byte 5, the parameters byte 1 to byte 4 can be set to zero. This function does not depend on the type of memory capacity of the MDT.

If the length of the data block (parameter in byte 3 and byte 4) is set to zero, the memory area from the MDT address in byte 1 and byte 2 to the end the available memory is formatted to the value in byte 5, regardless of the memory capacity of the MDT.

5.2.5.2 Examples

Example 1 Use the following formatting parameters to restore the MDT to the factory settings:

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
0	0	0	0	255

The MDT status and the three MDT pointers are not affected by the formatting function and have to be reset separately by the application.

This function is recommended if, e.g., existing MDTs need to be used in a new system or if the MDTs are being put back into use after troubleshooting.

Example 2 The MDT memory area from address 0x0200 to address 0x0580 should be set to a value of zero.

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
0x02	0x00	0x03	0x80	0

5.2.6 Map of MDT software version

The version number of the MDT software can be used for diagnostic purposes. Write access to this parameter returns an error message, since the version number cannot be edited.

5.3 SLK register area

The SLK register area contains all variables used to control how the SLK functions. These variables can always be read or written, regardless of whether or not an MDT is in the HF field.

5.3.1 Actual link state

The actual link state (operating state) of the SLK is an important parameter. The link state indicates whether or not the SLK is waiting for an MDT (HF field is on), whether or not an MDT is ready for data exchange, or whether or not the system is in an error state.

Only the actual link state tells the application whether or not the command request (see below) was also executed on the SLK.



An application (e.g., a PLC) has to implement all link states listed here. If the application does not use prefetch, for example, the actual link state can also temporarily return the value *PRECONNECTED*.

The actual link state is available by default in the event-oriented data channel of the fieldbus and does not have to be read from a command. The application evaluates the states using the coding in the following table:

Table 9: Actual link state

State	Code	Display	Description
<i>DISCONNECTED</i>	1	OFF	The SLK field is off. MDTs cannot sign on to the SLK. Read access to the data from the previous MDT is possible through the fieldbus.
<i>CONNECTING</i>	2	WAI	The SLK is ready for MDT sign-on. The field is on and the next MDT will sign on to the SLK.
<i>PRECONNECTED</i>	3	PRE	An MDT has signed on to the SLK. Automatic data exchange is running if prefetches were parameterized.
<i>CONNECTED</i>	4	CON	The MDT is connected in the HF field, data can be exchanged with the MDT.
<i>ERROR</i>	5	E00	The SLK has gone into the ERROR state, the field is off. No other MDTs can sign on to the SLK. Active communication with the MDT was interrupted without the MDT signing off.
<i>PROGRAM</i>	6	PRG	Temporary state
<i>BUSY</i>	7	BSY	The SLK is processing the commanded link state.

The actual link state can change in one of two ways:

- Due to a commanded link state from the application (see Chapter 5.3.2 “Commanded link state”)
- Automatically by the SLK, e.g., when an MDT signs on

The actual link state can only be read. Write access to this parameter returns an error message, since the value of the actual link state can only be altered by the SLK itself.

5.3.2 Commanded link state

The application controls the SLK through write access to the commanded link state. This means, for example, that the SLK can be made ready for MDT sign-on or terminate currently active communication with an MDT.

Table 10: Commanded link state

Action	Code	Description
CONNECT	1	Makes the SLK ready for MDT sign-on. The HF field is turned on.
DISCONNECT	2	Closes the MDT, i.e., the current communication with the MDT is terminated, the MDT and the field are turned off. No other MDTs can sign on to the SLK.
RECONNECT	3	Closes the MDT, i.e., the current communication with the MDT is terminated, the MDT is turned off. However, the SLK remains ready to receive data, the field is not turned off. The next MDT will sign on to the SLK.
ERROR	4	The SLK has gone into the ERROR state, the field is off. No other MDTs can sign on to the SLK. The current communication with the MDT is first terminated, and the MDT signs off.

The term “request” means the actual link state does not always have to fulfill the request to switch immediately.

Example: After the CONNECT request, the SLK is only “connected” when an MDT enters the SLK field. Only then will the actual link state change to *CONNECTED*.



It is strongly recommended to check whether or not the system has implemented the expected state following a commanded link state to the actual link state. Only then can another commanded link state be made.

5.3.3 Auto mode

The auto mode function automatically terminates the MDT connection after executing the prefetch. See Chapter 6.7 “Auto reconnect function” and Chapter 6.8 “Auto disconnect function”. Auto mode is inactive by default, the value is zero in the auto mode register.

Entering a value of one immediately activates **auto reconnect**. All MDTs signing on to the SLK will now run through the *PRECONNECTED* state and are then completed. The connection switches to the *CONNECTING* state (display shows “wai”), i.e., the HF field is active and the SLK can communicate with other MDTs.

Entering a value of two immediately activates **auto disconnect**. All MDTs signing on to the SLK will now run through the *PRECONNECTED* state and are then completed.

The connection switches to the *DISCONNECTED* state (display shows “off”). To communicate with another MDT, the link state with the command-oriented data exchange has to be switched back to *CONNECTING*.

It is not possible to access an MDT through a command-based data exchange in either variant of auto mode.

5.3.4 SLK operative flag

The SLK operative information indicates a system restart by the SLK setting the SLK operative flag to one after boot up. This gives the application the opportunity to check, for example, whether or not the SLK performed a restart after a voltage interruption.

To do this, the application sets the SLK operative flag to zero during boot up. Should the SLK unexpectedly restart after an interruption in supply voltage, the application can place its own sequencers in their initial state after evaluating the data block and place the SLK back in the desired link state.

5.3.5 Look ahead function

The look ahead function supports the reading of MDT data from a data block addressed through one of the three MDT pointers.

When an MDT pointer is read, the SLK not only retrieves the corresponding pointer value from the MDT, but also data bytes addressed by the pointer value. The number of the data bytes can be adjusted with an entry in the **Look Ahead Control** SLK register.

The value zero in the Look Ahead Control register deactivates the look ahead function. In this case, only the pointer is read from the MDT.

The data from the user data area is only saved in the SLK and not returned through the event- or command-oriented data exchange.

Make sure that the entire data block falls within the MDT's user data area, i.e., the MDT address saved in the read pointer plus the current value of the Look Ahead Control register have to produce an MDT address within the user area. If this address area is exceeded, the prohibited data is flagged with an error in the SLK memory. When reading this data, the system returns the error “MDT communication interrupted”.

A look ahead is naturally executed during prefetch. In fact, the MDT data saved in the SLK can only be read when the MDT has left the HF field. This supports time-critical applications in conjunction with the auto reconnect function.

5.3.6 SLK device information

The SLK device information refers to the four-digit SLK software version and the SLK device name, for example: **ID 40/SLK-PDP**. This information is constant and cannot be modified.

6 Accessing data on the ID 40/MDT

This chapter describes how to access data on the ID 40 system. Data is exchanged both through the fieldbus interface of the SLK and through the web browser on a PC connected to the SLK's COM port (see Chapter 10 “Web interface”).

Using a fieldbus system requires a corresponding program running on a PLC, SoftPLC or other platform with a corresponding fieldbus master module. The ID 40 system itself does not support programming.

The ID 40 system has function blocks for **Siemens S7** controllers and **Rexroth CL and PCL** controllers to create the programs. See Chapter 14 “Overview for ordering ID 40 modules” for ordering codes.

Data can be transmitted through the fieldbus either by event or command. Different fieldbus channels are used that are typically mapped to various I/O address ranges in the controller.

Accessing the data on the mobile data tag is divided into direct and parameterized data exchanges. These access types can also be combined. When an MDT enters the HF field, all parameterized commands are executed first. Data is then either transmitted directly or the data exchanged with the MDT is automatically terminated.

6.1 Command-oriented data transmission

Command-oriented data transmissions are always initiated by the fieldbus master, e.g., a PLC. During this, the master sends a command to the SLK. The SLK executes the command and sends a response to the master. The response always contains the event code. For read commands, the SLK sends the read data to the master in the response. Once the response has been sent, the SLK is ready to execute additional commands.

6.2 Event-oriented data transmission

In event-oriented data transmissions, the data is transmitted by parameterized read commands from the read/write head to the control system. A new MDT entering the HF field is the event that triggers the updating of the data. The data sent to the control system in this manner has to be parameterized beforehand. This is done using command-oriented data transmissions (Chapter 6.4 “Parameterized data exchange with MDT”).

6.3 Direct data exchange with MDT

The data on the mobile data tag can be accessed directly through a command-oriented data transmission as long as the MDT is located within the HF field of the read/write head. To do this, the control system sends read and write commands to the read/write head through the fieldbus. The read/write head executes these commands immediately and sends the result to the control system.

Direct data exchange requires the MDT to be stationary in the HF field.



It is recommended to only exchange data directly with stopped workpiece pallets, since otherwise the dwell time of the MDT in the field is too brief to transfer the amount of data through the field. The runtime of the host system also has to be taken into consideration.

6.4 Parameterized data exchange with MDT

For parameterized data transmissions, the control system sends read and write commands to the read/write head in a special parameterization mode when there is no mobile data tag in the HF field. These commands are buffered in the read/write head and automatically executed when an MDT enters the HF field.

Parameterized data exchange is particularly quick, since the read and write commands are already stored in the SLK. This makes it best suited for dynamic reading and writing.

6.4.1 Prefetch

Prefetch refers to any parameterized read access to the mobile data tag. It is separated into the read data being transmitted immediately to the control system (unbuffered prefetch) or held in a special data buffer in the SLK (buffered prefetch). Setting the parameters of a prefetch means entering SLK addresses that should be read in the prefetch. Multiple, even overlapping data areas can be set up.



The prefetch function is only executed once the parameters have been set.

If an error occurs during prefetch (e.g., the MDT is no longer in the HF field), the invalid/unrefreshed data areas are flagged and the error information is returned in the status byte of the fieldbus channel.



The amount of data in the prefetch has to be readable within the dwell time of the MDT in the HF field.

6.4.1.1 Unbuffered prefetch

With unbuffered prefetch, the parameterized read commands are executed and the read data is sent to the control system in an event-oriented data transmission. The low protocol overhead is advantageous here. Since the number of bytes in the event-oriented data channel is limited, unbuffered prefetch is suitable primarily for reading small amounts of data.

6.4.1.2 Buffered prefetch

With buffered prefetch, the data read from the mobile data tag with the parameterized read commands are stored in a special buffer in the SLK. The data from multiple, consecutive read commands is entered in order of execution in the buffer without interruption. This means, for example, that unrelated data areas can be read from the mobile data tag and copied from the SLK in one related data block. The control system can read some or all of this data buffer at any time, even if the MDT has already left the HF field of the SLK. The data in the buffer remains valid until it is overwritten by a subsequent MDT.

The maximum buffer size corresponds to the current MDT size. This means the entire contents of an ID 40/MDT32K can be read with buffered prefetch.

The parameterized data blocks cannot exceed the address area of the MDT.

Otherwise, the wrong data bytes are flagged in the prefetch buffer, and an error code is returned when reading the buffer.

6.4.2 Pretransmit

Pretransmit refers to any parameterized write access to the mobile data tag. These write commands are divided into those executed only once and those executed for each additional MDT.

Setting the parameters of a pretransmit means entering SLK addresses together with the data that should be written starting with the addresses.

The pretransmit data block is entered in the SLK through a fieldbus command. Other commands add data blocks and the address ranges of these data blocks can also overlap.

Configuring the pretransmit means:

- Selecting whether the pretransmit should be executed as a **single transmit** or a **multiple transmit**
- Releasing the pretransmit for execution. As long as the pretransmit is locked, additional data blocks in the SLK can be entered without a present MDT with incomplete pretransmit data being written.
- Established pretransmits can also be deleted.

Pretransmit always occurs before prefetch in the *PRECONNECTED* state.



It is recommended to only execute a pretransmit with stopped workpiece pallets, since otherwise the dwell time of the MDT in the field is too brief to transfer the amount of data through the field.

If an error occurs during pretransmit (e.g., the amount of data parameterized for the MDT is too large), the bytes not transmitted are flagged in the SLK. This can be verified by subsequent read access to the pretransmit data block. Since there is no bus communication during pretransmit, errors cannot be reported through the fieldbus.

6.4.2.1 Single transmit

Write commands that are executed only once for the next MDT entering the HF field of the SLK are known as single transmit (ST).

6.4.2.2 Multiple transmit

Multiple transmit (MT) commands are repeated for all subsequent MDTs. This makes them particularly suitable for initializing data carriers, e.g., at the first process station of an assembly line.



Multiple transmit automatically overwrites every MDT passing the SLK without prompt on the fieldbus. There is the risk that accidentally either important data on the MDT is deleted or incorrect data is entered on the MDT.

6.5 Default process for SLK-MDT communication

The ID 40 system works in three phases:

- **Phase 1:** No data exchange between SLK and MDT.
The HF field is off
- **Phase 2:** Data exchange possible between SLK and MDT.
The HF field is on.
- **Phase 3:** Data transmission between PLC and MDT

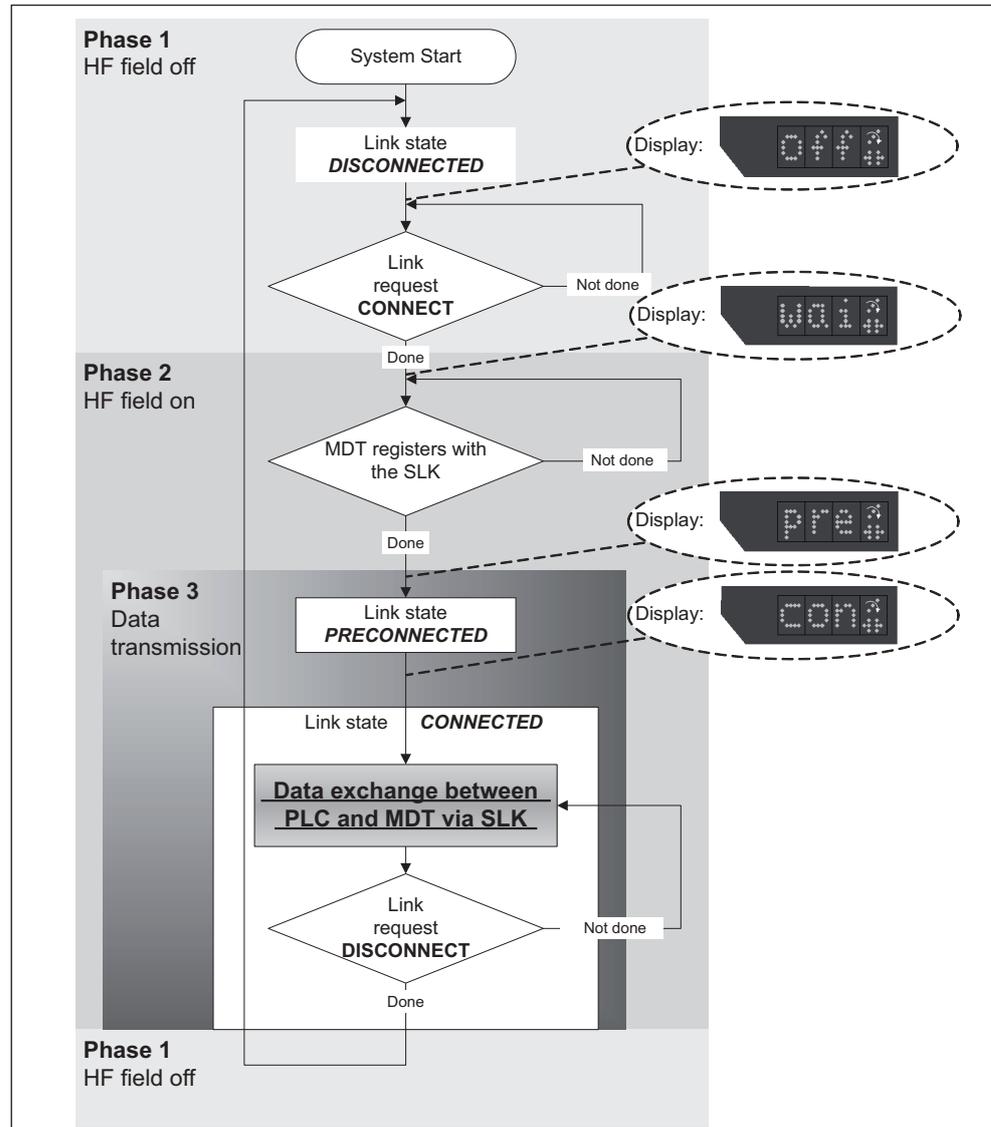


Fig. 19: Default process for SLK-MDT communication

6.6 “Open transfer” and “Close transfer” functions

Due to the new MDT lifeguarding function (see Chapter 6.9 “MDT lifeguarding”), the transfer channel functions are obsolete.

However, existing applications do not need to be adjusted, since an SLK running software version 3.11 or higher accepts the old transfer channel commands.

However, no function is executed in the SLK.

6.7 Auto reconnect function

Auto reconnect allows data to be exchanged with an MDT without stopping the workpiece pallet in front of the SLK. A typical application for this is a transfer station where it has to be decided whether the workpiece pallet continues forward or is fed out. Normally minimal information from the MDT is sufficient for this purpose. See also Chapter 7.9.2 "Transfer' application".

This information can be transmitted by the MDT to the SLK as it passes by. The desired data is read from the MDT with the prefetch function (in *PRECONNECTED* state), then communication with the MDT is automatically terminated. The field does not turn off once communication with the MDT stops, i.e., the SLK can immediately accept another MDT. No bus communication is necessary to control these processes. The data read in prefetch is transmitted to the application through an event-oriented data exchange (see Chapter 6.2 "Event-oriented data transmission").

Automatically terminating communication with the MDT is turned on and off through auto mode (see Chapter 5.3.3 "Auto mode").



It is not possible to access the MDT with the command-oriented data exchange when auto reconnect is active, since the link state *CONNECTED* is bypassed.

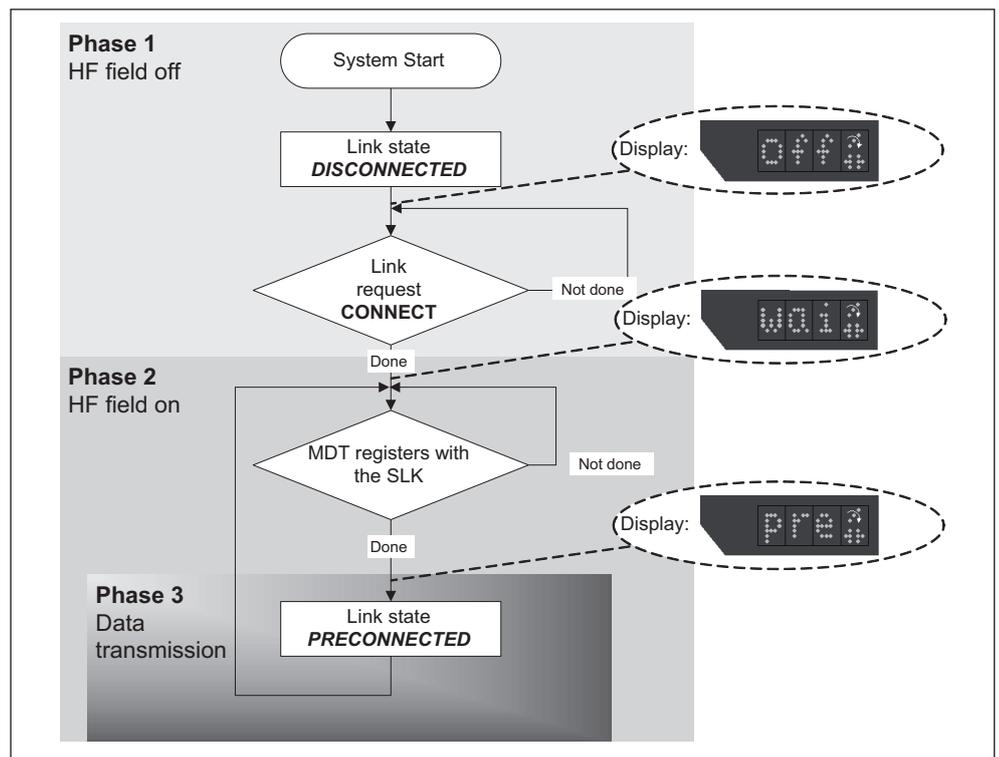


Fig. 20: Link states for auto reconnect

6.8 Auto disconnect function

Auto disconnect allows data to be exchanged with an MDT without stopping the workpiece pallet in front of the SLK, just like with auto reconnect. However, with auto disconnect, the SLK switches to the link state *DISCONNECTED* once *PRECONNECTED* is completed. Since the HF field turns off in *DISCONNECTED*, the link state has to go back to *CONNECTING* in order to communicate with a new MDT. A typical application for this is described in Chapter 7.9.3 “Workstation' application”.

As with auto reconnect, the data read in prefetch is transmitted to the application through an event-oriented data exchange (see Chapter 6.2 “Event-oriented data transmission”).

Automatically terminating communication with the MDT is turned on and off through auto mode (see Chapter 5.3.3 “Auto mode”).



It is not possible to access the MDT with the command-oriented data exchange when auto disconnect is active, since the link state *CONNECTED* is bypassed.

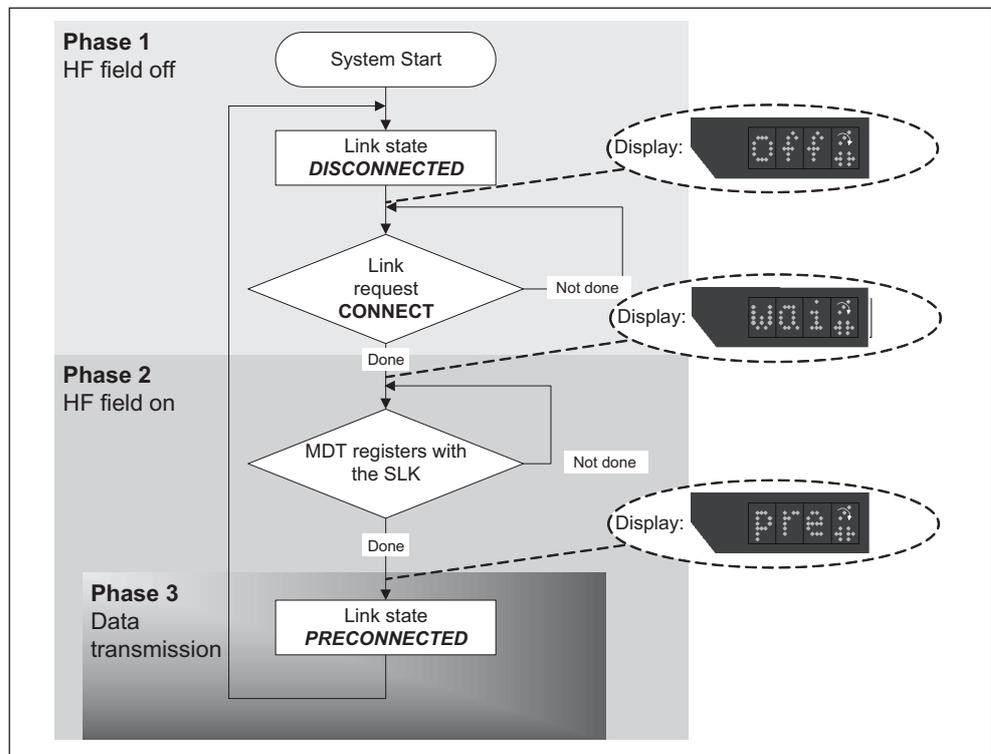


Fig. 21: Link states for auto disconnect

6.9 MDT lifeguarding

MDT lifeguarding monitors the **presence of an MDT in the *CONNECTED* state**. If the MDT leaves the HF field without signing off through DISCONNECT or RECONNECT, the SLK automatically switches to the *ERROR* state.

This results in the following properties:

- With an SLK in the link state *CONNECTED*, it is guaranteed that an MDT is present and read and write access within the memory capacity of the MDT is unrestricted.
- The “Open transfer” and “Close transfer” functions are no longer needed (described in Chapter 6.6 of the ID 40 system manual version 3.0). MDT lifeguarding ensures the consistency of the transmitted data packets.
- MDT lifeguarding also prevents an MDT from signing on again unexpectedly, since the HF field is off while in the *ERROR* state.

Tips for the application:

- Only access the MDT when the link state is *CONNECTED*. If *ERROR* appears between related data transfers, the MDT data set is inconsistent.
- No longer use commands to open and close the transfer.
- If MDT lifeguarding unexpectedly switches to the *ERROR* state, the sequence programs need to be checked. This can be caused by a stop gate that opens too early. Or the DISCONNECT or RECONNECT command was not sent to the SLK.
- Proximity switches to detect a workpiece pallet are no longer needed if no additional redundancy is desired.

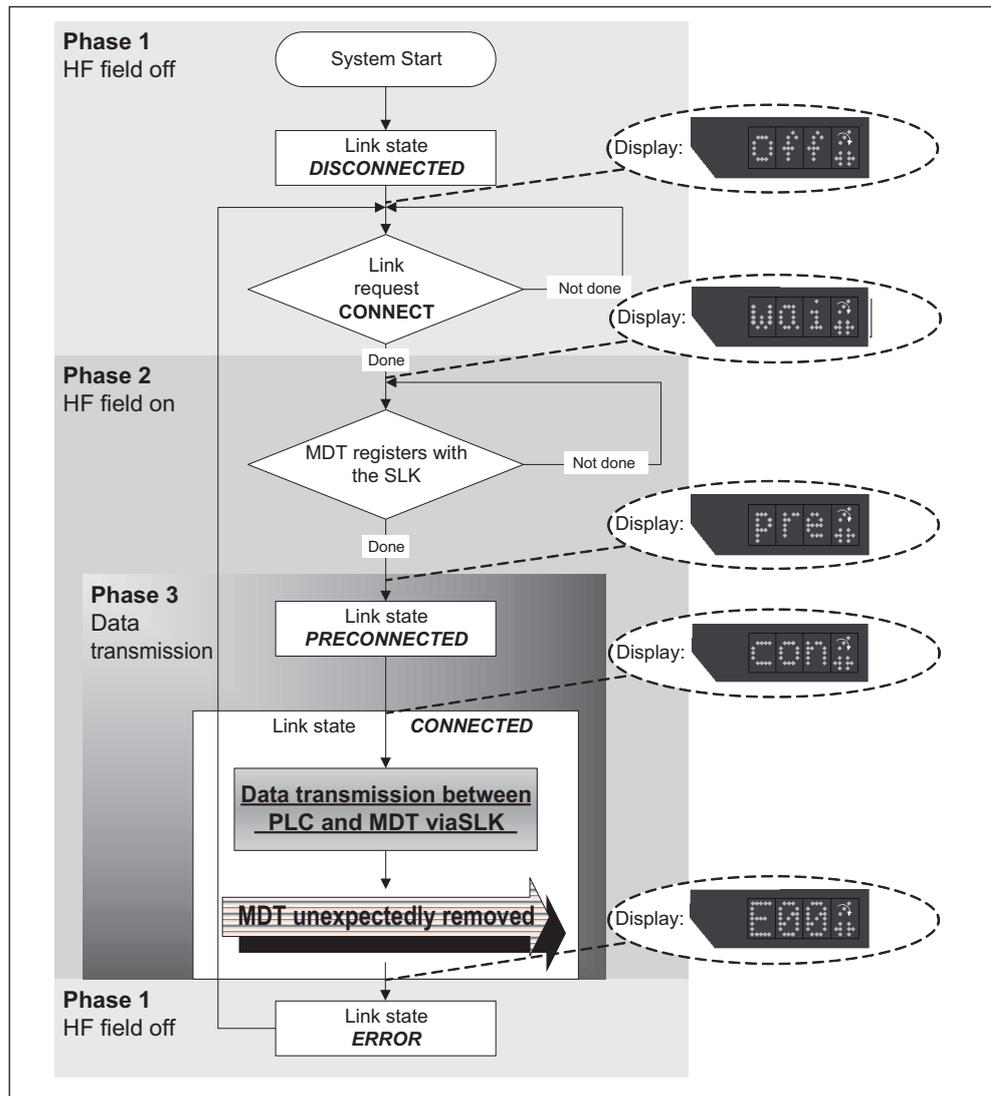


Fig. 22: Link states for MDT lifeguarding

7 Profibus DP

7.1 Overview

The ID 40/SLK-PDP supports the basic functions of the Profibus DP Communication Profile as per EN 50170.

The SLK does not support any extended DP functions such as noncyclical read and write functions, as specified for DP-V1.

Data is exchanged through the cyclical data communication of a Class 1 DP master (DPM1).

Based on this cyclical data communication, the SLK offers two types of data transmission:

- Command-oriented data exchange, bidirectional data transmission between bus master and SLK.
- Event-oriented data exchange, unidirectional data transmission from SLK to bus master

Command- and event-oriented data exchange is mapped to different I/O addresses in the controller.

7.2 Command-oriented data exchange

Command-oriented data exchange is bidirectional and allows parameters to be set for event-oriented data exchange, as well as the transmission of data from and to data carriers/the SLK.

The following functions are supported:

- Writing data (MDT data only in *CONNECTED* state).
- Reading data
- Configuring unbuffered prefetch
- Configuring buffered prefetch, reading the prefetch buffer
- Configuring pretransmit

A **command** consists of multiple bytes. The first byte (byte 0) always contains the command code. The entire command byte sequence is stored on the output map of the user controller. The SLK receives the command by inverting the **toggle bit** (bit 7 in byte 0). The SLK returns the same bit value once the command has been executed.

Accordingly, the PDP master controller handles the toggle bit as follows:

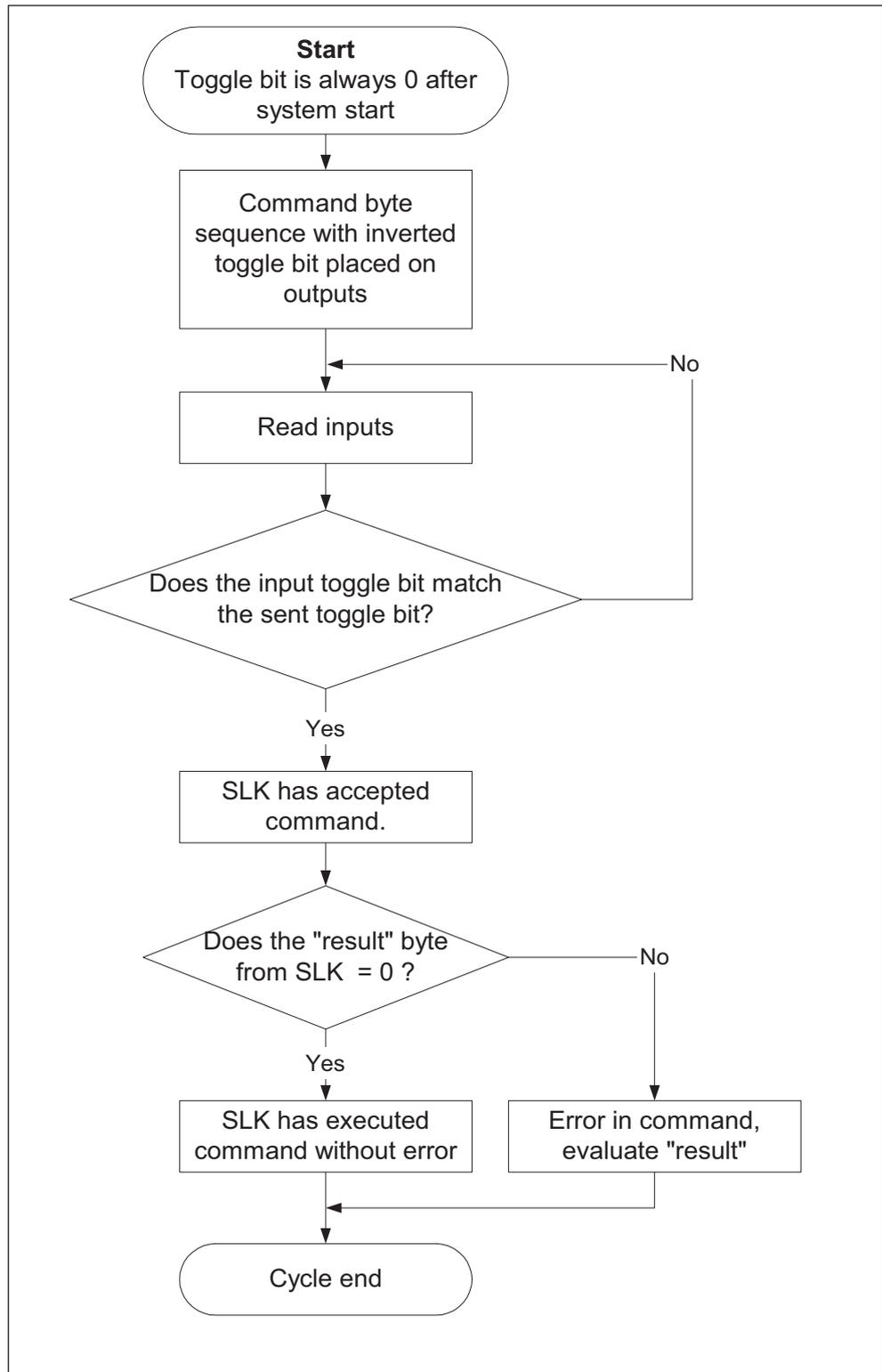


Fig. 23: Command-oriented data exchange sequence

Output assignments for a command from the PDP master to the SLK:

Table 11: Output assignments for command-oriented data exchange

Output address	Bit	Name	Description		
0	7	Tbit	Toggle bit, has to be inverted for each command		
	6 - 0	CMD	Command, see Chapter 7.2.1 "Profibus commands" for coding		
1		Data0	SLK address	SLK address	Parameter
2		Data1			
3		Data2			
4		Data3			
5		Data4	No. data bytes	No. data bytes	Configuration command
6		Data5			
7		Data6		Data byte 1	
8		Data7		Data byte 2, etc.	
			Read data	Write data	

ENGLISH

The data area is organized such that the high byte is in the low address (Motorola format), e.g., the "no. data bytes" is divided as follows:

Data4 (low output address) **Data5 (higher output address)**
 "No. data bytes" high byte "No. data bytes" low byte

Input assignments for SLK reply to PDP master

Table 12: Input assignments for command-oriented data exchange

Input address	Bit	Name	Description
0	7	Tbit	Toggle bit, has to have the same value as the corresponding command
	6 - 0	CMD	"Mirrored" command code
1		Result	Information on whether the command was correct (value = 0) or not (value > 0)
2		Data1	Data bytes (if command was read access)
...		...	

Unassigned bytes in event-oriented data exchange transmit 0.

7.2.1 Profibus commands

Table 13: Profibus commands table

CMD code	Command	Description	Query	Reply
0	Idle command	Command has no effect		
1	Set parameters for unbuffered prefetch	The addressed data area is added to the unbuffered prefetch. The data is included in the event-oriented data exchange in order of definition. This means the data defined first are added to the low addresses in the event-oriented data channel regardless of its address in the data carrier	Data0–Data3: SLK address Data4–Data5: No. bytes	Data0: Result Data1–Data2: Address in event-oriented data channel ([Data 1] high byte always zero). The data block begins at this address after the prefetch is executed
11	Configure unbuffered prefetch	Parameters for the unbuffered prefetch can set or all parameterized data areas can be deleted. The unbuffered prefetch is only executed once the parameters have been set	Data0 0: Begin setting parameters for unbuffered prefetch 1: Finish setting parameters for unbuffered prefetch 4: Delete unbuffered prefetch	Data0: Result
3	Set parameters for buffered prefetch	The addressed data area is added to the prefetch buffer. The data is included in the prefetch buffer in order of definition. This means the data defined first are added to the lowest address in the buffer regardless of its address in the data carrier	Data0–Data3: SLK address Data4–Data5: No. bytes	Data0: Result Data1–Data2: Address in command-oriented data exchange
10	Configure buffered prefetch	Parameters for the buffered prefetch can set or all parameterized data areas can be deleted. The buffered prefetch is only executed once the parameters have been set	Data0 0: Begin setting parameters for buffered prefetch 1: Finish setting parameters for buffered prefetch 4: Delete buffered prefetch	Data0: Result
4	Read prefetch buffer	The data from the prefetch buffer is read	Data0–Data1: Offset in command-oriented data exchange Data2–Data3: No. bytes	Data0: Result Data1.n: Read data
5	Read data	The indicated data area is read	Data0–Data3: SLK address Data4–Data5: No. bytes, Data4 always 0	Data0: Result Data1.n: Read data
6	Write data	The indicated data block is written	Data0–Data3: SLK address Data4–Data5: No. bytes, Data4 always 0 Data6.n: Data to be written	Data0: Result

CMD code	Command	Description	Query	Reply
7	Set parameters for pretransmit	The indicated data area is added to the pretransmit. This command can be selected multiple times, whereby the various areas can overlap.	Data0–Data3: SLK address Data4–Data5: No. bytes, Data4 always 0 Data6.n: Data in pretransmit to be written	Data0: Result
8	Configure pretransmit	Begin or finish setting pretransmit parameters. Start “single transmit” (ST) or “multiple transmit” (MT). Delete all parameterized data areas. Stop pretransmit	Data0: 0: Begin setting parameters 1: Finish setting parameters 2: Start ST 3: Start MT 4: Delete parameters 5: Stop pretransmit	Data0: Result

7.3 Event-oriented data exchange

With event-oriented data exchange, the data read by the unbuffered prefetch (Chapter 6.4.1.1 “Unbuffered prefetch”) is transmitted from the SLK to the fieldbus master. Write access to the SLK is not possible in event-oriented data exchange. All data in this channel is updated simultaneously once it has been read from the data carrier.

The **size** of the event-oriented data channel is determined by I/O modules in the Profibus GSD file, see Chapter 7.8 “Profibus GSD file”. These modules are designated by “... event-oriented data access”.

Example:

Module “32 | event-oriented data access” reserves 32 bytes of input for event-oriented data exchange in the cyclical data communication for Profibus. Determining the I/O module for the event-oriented data channel is based on the desired number of bytes that should be read by the system using the unbuffered prefetch. Two status bytes containing important operational information (see below) are added to this number.



Even if no unbuffered prefetch data is desired, **at least** two bytes have to be reserved for the event-oriented data channel, since the ID 40/SLK-PDP shows important system information here.

7.3.1 Status information in event-oriented data channel

The **first two bytes** in the event-oriented data exchange are status bytes with the following structure:

Table 14: Status byte 0 of event-oriented data exchange

Byte 0 (bits 0–7)							
7	6	5	4	3	2	1	0
Event counter				Result code			

Bits 0 to 3 comprise the result code. The result code is zero once a previously parameterized prefetch is executed without error. An error code greater than zero indicates when errors occurred during the prefetch (e.g., unexpected break in communication between SLK and MDT).

The coding for the result code is the same as for command-oriented data exchange (see Chapter 7.5 “Profibus error codes”).

Bits 4 to 7 comprise the **event counter**. The counter runs cyclically from 0 to 15 and is increased by 1 when the current *PRECONNECTED* link state is ended. This signals to the control system that the data from the prefetch parameterization is updated. At this point, the user program can import prefetch data. The counter also increases when no prefetch is parameterized.

The following functions have been completed when the counter value changes:

- Pretransmit data has been entered in the MDT (if parameterized)
- Prefetch data was read from the MDT (if parameterized) and is ready to be imported to a PLC DB
- If auto reconnect is active, the MDT was closed and the link state is *CONNECTING*
- If auto disconnect is active, the MDT was closed and the link state is *DISCONNECTED*
- If no auto mode is active, the link state is *CONNECTED*. Data can be exchanged with the MDT.

Table 15: Status byte 01 of event-oriented data exchange

Byte 1 (bits 0–7)							
7	6	5	4	3	2	1	0
Actual link state				Data transfer status	Prefetch param. open	Prefetch parameterized	Operative flag

Bit 0 shows the status of the SLK operative flag, see Chapter 5.3.4.

Bits 1 and 2 show the current status of prefetch parameterization. The flags apply both to the unbuffered prefetch function and the buffered prefetch.

Bit 1 Prefetch is parameterized:

‘0’ = no prefetch currently parameterized in system

‘1’ = **at least one** unbuffered prefetch **or one** buffered prefetch is parameterized

Bit 2 Prefetch parameterization is open:

“0” = prefetch parameterization is closed. If prefetches have already been parameterized, these are executed once the system completes the *PRECONNECTED* state.

‘1’ = prefetch parameterization is open. No prefetches are executed in this state.

Bit 3 Data transfer status (see Chapter 6.6 “‘Open transfer’ and ‘Close transfer’ functions”):

‘1’ = “Open transfer” command executed

‘0’ = “Close transfer” command executed

Bits 7 to 7 show the actual link state (see Chapter 5.3.1 “Actual link state”).

7.3.2 Data array for event-oriented data channel

The SLK provides the data from the unbuffered prefetch starting with the **third byte** of the event-oriented data exchange. If no prefetch was executed, the value of each byte is zero.

7.4 SLK address table

The address information consists of the start addresses for the data areas described in Chapter 5 “SLK memory structure” and is needed for data exchange via Profibus and program-assisted access via the web interface (Chapter 10.7). The ID 40 system supports the bytes data format (unsigned char or UNSIGNED8). The application can implement more complex data formats.

Table 16: SLK address table

Segment	Addresses	Data format	Access*	Meaning
0x0000	MDT user data area			
	0x00000000	Area length (byte number) depending on MDT type, see Chapter 4.1 “ID 40/MDT storage”	rd/wr	Area for user-specific data, not interpreted by system
0x0001	- Reserved -			
0x0002	MDT register area			
	0x00020000	High byte, low byte	rd/rst	MDT status
	0x00020002	High byte, low byte	rd/wr	MDT pointer 1
	0x00020004	High byte, low byte	rd/wr	MDT pointer 2
	0x00020006	High byte, low byte	rd/wr	MDT pointer 3
	0x00020008	Highest byte - lowest byte	rd only	MDT ID code, 4-digit
	0x0002000C	High byte, low byte start address	rd/wr	MDT formatting
	0x0002000E	High byte, low byte length (byte count)		The MDT memory is overwritten with the formatting value starting with the start address. See Chapter 5.2.5 “MDT formatting”
	0x00020010	Byte formatting value		
	0x00020011			- Reserved -
	0x00020012	High byte, low byte	rd only	Internal MDT work pointer (for diagnostic purposes only)
	0x00020014		rd only	MDT software version number
	0x00020015			- Reserved -
0x0003	SLK register area			
	0x00030000	Byte	rd only	Actual link state
	0x00030001	Byte	rd/wr	Commanded link state
	0x00030002	Byte	rd/wr	Auto mode configuration
	0x00030003	Byte	rd/wr	SLK operative flag
	0x00030004	Highest byte - lowest byte	rd/wr	MDT counter, 4-digit
	0x00030008	16 bytes ASCII	rd only	SLK device name
	0x00030018	Highest byte - lowest byte	rd only	SLK software version, 4-digit version ID
	0x0003001C	High byte, low byte	rd/wr	MDT pointer 1 Look Ahead Control register
	0x0003001E	High byte, low byte	rd/wr	MDT pointer 2 Look Ahead Control register
	0x00030020	High byte, low byte	rd/wr	MDT pointer 3 Look Ahead Control register

- * rd only Read only, no writing possible.
 rd/wr Read and write access.
 rst Register is reset by writing any value.
 none No access.

7.5 Profibus error codes

The following error codes occur both in the “result” byte of command-oriented data exchange (see Chapter 7.2.1 “Profibus commands”) and in bits 0–3 of the status byte in event-oriented data exchange (see Chapter 7.3.1 “Status information in event-oriented data channel”).

Table 17: Result byte Profibus coding

Result	Meaning
0	No error
1	Unexpected break in communication between SLK and MDT
2	MDT device error, e.g., memory area defective
3	Error in SLK addressing, e.g., start address and data length exceed limit of an SLK address segment Access to nonexistent SLK address segment
4	- Reserved -
5	- Reserved -
6	- Reserved -
7	Attempt to set parameters for prefetch although parameterization was not started. Prefetch buffer read although prefetch parameterization has started
8	Operand value outside permitted range, e.g., for command 15
15	Internal system error, e.g., memory shortage

7.6 Profibus Diagnostic Service

The PDP Diagnostic Service has two sections:

- **Normal diagnostics** and
- **User diagnostics**

The SLK provides diagnostic information either when prompted by the PDP master or when the SLK detects an error in the Profibus function area.

Information in normal diagnostics is Profibus standard. One component is the ID number, which for the ID 40 system is 0106.

Table 18: User diagnostic message coding

Byte	Description	Value	Format
1	No. bytes in user diagnostics incl. this byte count	8	1 byte
2	PDP node ID	1 - 126	INT8, info found in PDP module
3	SLK software version	Example: 2.10.04.12	4 bytes
7	Actual link state	See Chapter 5.3.1	1 byte
8	SLK operative flag	See Chapter 5.3.4	1 byte

7.7 Addressing data in the ID 40 system

The **SLK address** is needed to read and write data through the fieldbus both on the MDT and the SLK, and is shown as a 32-bit value in hexadecimal format.

Example: 0x000002C0

A **data block** consists of the SLK address and a byte count. The SLK address is the start address, and the byte count determines the area length from the start address. It is written as **address/byte count**.



The SLK address is always shown in hexadecimal format, whereas the byte count is always in decimal format.

Example:

Four bytes should be read starting from address 0x000002C0, so the data block is 0x000002C0/4

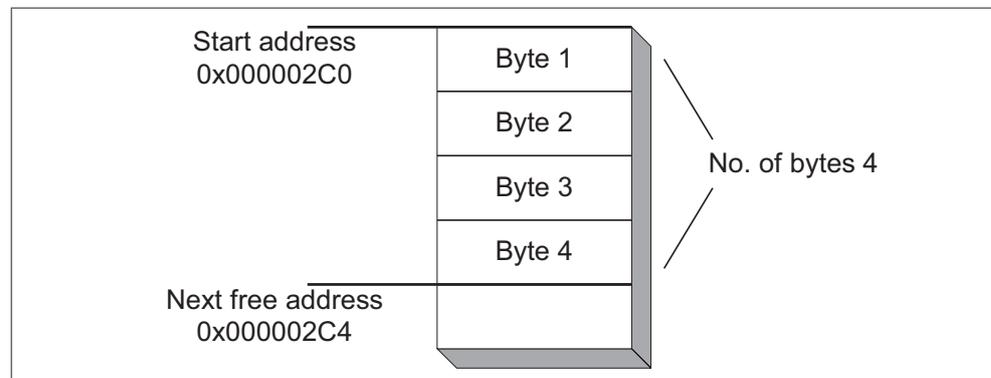


Fig. 24: Example of data block with length of 4



Do not confuse the SLK address with the fieldbus node number.

The **SLK address ranges** are listed in Chapter 7.4 “SLK address table”.

7.8 Profibus GSD file

The characteristic communication features of the ID 40/SLK is determined in the form of an electronic device data sheet (GSD file). GSD files for devices with the DP Communication Profile are outlined in Profibus Guideline No. 2.121.

The general settings in the GSD file include:

- Manufacturer: Bosch Rexroth AG
- Device name: ID 40/SLK-PDP
- Hardware and software release versions,
- supported baud rates [kbps]:
9,6 19,2 93,75 187,5 500 1500 3000 6000 12000

7.8.1 Distribution of SLK module configuration

The slave-specific settings in the GSD file contain information on the available I/O modules. The cyclical data communication of Profibus supports a payload transport of 216 bytes. This data area is available for both command-oriented and event-oriented data exchange. The I/O modules defined in the GSD file represent the allocation of these bytes to the data exchange channels.

The maximum volume of a command-oriented data exchange is 212 bytes, since at least 4 bytes always have to be allocated to the event-oriented data exchange. This is necessary for transmitting the system status, which is always present.

The maximum volume of an event-oriented data exchange is 208 bytes, since at least 8 bytes always have to be allocated to the command-oriented data exchange. Only then can a configuration command with at least one data byte be sent to the SLK.

7.8.2 Sample I/O module configuration

If there are 121 bytes of user data, the “128 I/O cmd-oriented data acc.” module has to be used for configuration. This means that a maximum of 121 bytes can be read from the MDT per command. If more data is desired, it has to be distributed across multiple read commands.

In the following **example**, 250 bytes should be read starting at SLK address 0x000001F0. The data block is retrieved through three consecutive read commands. Each command reads 121 bytes. The byte count of the previous read command is added to the SLK address of a read command.

Table 19: Distribution of read access to large amounts of data in individual commands

	SLK address	Byte count
First read command	0x000001F0	121
Second read command	0x00000269	121
Third read command	0x000002E2	8

Individual data packets can now be provided in the PLC program as a single data block.

7.9 Sample applications

The following examples show some typical applications of the ID 40 system using Profibus.

7.9.1 “Manual workstation” application

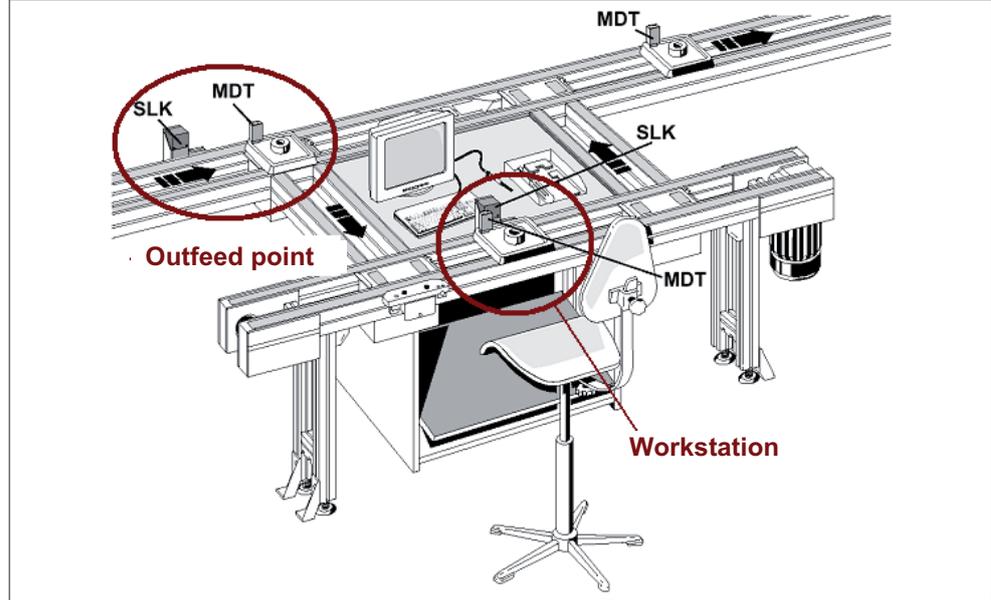


Fig. 25: Sample “manual workstation” application

The MDT should be read and written at a processing station. First, 32 bytes should be read from the MDT address 0x0110, then 60 bytes should be written from the MDT address 0x0100. The MDT status is also transmitted in order to allow any errors in the MDT to be addressed.

The actual link state is read in an event-oriented manner to determine whether or not a new MDT is in the link. The link state is contained in the event-oriented data channel by default and is, e.g., provided in the input map in PLC systems.

The remaining data is exchanged with the MDT directly in the *CONNECTED* state. The MDT then signs off with the commanded link state *RECONNECT*. The SLK switches to the *CONNECTING* state to wait for the next MDT.

SLK preparation

The SLK switches to the link state *CONNECTING* with the command-oriented data exchange.

No.	Command-oriented data exchange	CMD	Data0	Data1	Data2	Data3	Data4	Data5	Data6
1	Make SLK ready to receive (CONNECT)	0x06	0x00	0x03	0x00	0x01	0x00	0x01	0x01

SLK operation

Check the actual link state until the value *CONNECTED* appears (2). A new MDT has now arrived. The controller reads the MDT status (4). When the MDT status does not indicate an error, the user data can now be read starting with MDT address 0x0110. Once this data has been evaluated and the workpiece processed, the result of the processing is written to the MDT (6). Communication with the MDT is then terminated.

No.	Command-oriented data exchange	CMD	Data0	Data1	Data2	Data3	Data4	Data5	Data6
2	Wait until actual link state is <i>CONNECTED</i> . Read value through event-oriented data channel								
4	Read MDT status and check if error bits are present	0x05	0x00	0x02	0x00	0x00	0x00	0x01	
5	Read data from MDT, e.g., 32 bytes as of address 0x0110	0x05	0x00	0x00	0x01	0x10	0x00	0x20	
6	Write data to MDT, e.g., 60 bytes as of address 0x0100	0x06	0x00	0x00	0x01	0x00	0x00	0x3C	data...
8	Close MDT (RECONNECT)	0x06	0x00	0x03	0x00	0x01	0x00	0x01	0x03
9	Wait until actual link state is <i>CONNECTING</i> . Read value through event-oriented data channel								

ENGLISH

7.9.2 “Outfeed” application

Outfeed points are where it is decided whether or not the workpiece pallet is diverted. The workpiece pallets do not need to stop at the SLK before the junction, since the amount of data needed for the decision is small. In this example, 32 bytes should be read starting at MDT address 0x02F0. The MDT status is also transmitted in order to allow any errors in the MDT to be addressed.

The data is read from the MDT through an unbuffered prefetch, then the link immediately switches from *PRECONNECTED* to *CONNECTING*, since auto reconnect is active.

SLK preparation

No.	Command-oriented data exchange	CMD	Data0	Data1	Data2	Data3	Data4	Data5	Data6
1	Begin setting parameters for unbuffered prefetch	0x0B	0x00						
2	Delete unbuffered prefetch if any data blocks are entered	0x0B	0x04						
3	Set parameters for first data block in unbuffered prefetch: 1-byte MDT status	0x01	0x00	0x02	0x00	0x00	0x00	0x01	
4	Set parameters for second data block in unbuffered prefetch: 32 bytes starting from MDT address 0x02F0	0x01	0x00	0x00	0x02	0xF0	0x00	0x20	
5	Finish setting parameters for unbuffered prefetch. Prefetch is now active	0x0B	0x01						
6	Switch SLK to auto reconnect mode	0x06	0x00	0x03	0x00	0x02	0x00	0x01	0x01
7	Make SLK ready to receive (CONNECT)	0x06	0x00	0x03	0x00	0x01	0x00	0x01	0x01

SLK operation

Wait until event counter in event-oriented data channel (see Chapter 7.3.1 “Status information in event-oriented data channel”) to increase by 1.

At this point:

- A workpiece pallet has arrived
- Event-oriented data exchange with the MDT has occurred
- The link for the next MDT is back to *CONNECTING*
- The MDT status and the system data can be checked

The requested data bytes are sent to the fieldbus master through the event-oriented data exchange for every MDT. The data can be provided for additional evaluation using a controller function block.

7.9.3 “Workstation” application

This example is for a workstation with a lift positioning unit. Since a link cannot be established between the SLK and the MDT when the workpiece pallet is raised, data is exchanged in two stages.

Stage 1: Reading the workpiece-specific data to check whether or not this workpiece can be processed at this station. The data is read from the MDT in an event-oriented manner. The SLK then automatically switches to *DISCONNECTED*. Only when the lift positioning device lowers after processing does the SLK switch to *CONNECTING*. This eliminates the risk of an unexpected break in communication as the MDT approaches the SLK from above.

Stage 2: Entering measurement data obtained while the workpiece was being processed. Only the pretransmit function is used here.

SLK preparation

An unbuffered prefetch is configured that transmits both the MDT status (see Chapter 4.3.2.1 “MDT status register”) and the control data (e.g., 16 bytes as of MDT address 0x0024) in the event-oriented data channel. The SLK switches to auto disconnect mode and the link state switches to *CONNECTING*.

No.	Command-oriented data exchange	CMD	Data0	Data1	Data2	Data3	Data4	Data5	Data6
1	Begin setting parameters for unbuffered prefetch	0x0B	0x00						
2	Delete unbuffered prefetch if any data blocks are entered	0x0B	0x04						
3	Set parameters for first data block in unbuffered prefetch: 1-byte MDT status	0x01	0x00	0x02	0x00	0x00	0x00	0x01	
4	Set parameters for second data block in unbuffered prefetch: 16 bytes starting from MDT address 0x0024	0x01	0x00	0x00	0x00	0x24	0x00	0x10	
5	Finish setting parameters for unbuffered prefetch. Prefetch is now active	0x0B	0x01						
6	Switch SLK to auto disconnect mode	0x06	0x00	0x03	0x00	0x02	0x00	0x01	0x02
7	Make SLK ready to receive (CONNECT)	0x06	0x00	0x03	0x00	0x01	0x00	0x01	0x01
8	Wait until event counter in event-oriented data channel (see Chapter 7.3.1 “Status information in event-oriented data channel”) to increase by 1. At this point, a workpiece pallet has arrived, the event-oriented data is valid and the link is <i>DISCONNECTED</i> . The MDT status and control data can be checked and the workpiece pallet can be raised.								

Stage 2: Once processing is complete and the measurement data is available, the parameters for pretransmit are set (e.g., 80 bytes as of MDT address 0x01C0) and configured as a **single transmit** (see Chapter 6.4.2 “Pretransmit”). By this time, the workpiece pallet has lowered. The SLK switches to *CONNECTING* and the MDT immediately signs on and executes the pretransmit.

No.	Command-oriented data exchange	CMD	Data0	Data1	Data2	Data3	Data4	Data5	Data6
1	Delete pretransmit data from last MDT	0x08	0x04						
2	Begin setting pretransmit parameters	0x08	0x00						
3	Set parameters for 80-byte measurement data block as pretransmit	0x07	0x00	0x00	0x01	0xC0	0x00	0x50	Measurement data
4	Finish setting pretransmit parameters	0x08	0x01						
5	Activate “single transmit”	0x08	0x02						
6	Wait until lift positioning unit in lower position								
7	Make SLK ready to receive (<i>CONNECT</i>)	0x06	0x00	0x03	0x00	0x01	0x00	0x01	0x01
8	Wait until event counter in event-oriented data channel increases by 1. At this time, the measurement data is written to the MDT and the link is <i>DISCONNECTED</i> (auto reconnect is still active)								
9	Make the SLK ready to receive the next workpiece pallet only once the current workpiece pallet is gone	0x06	0x00	0x03	0x00	0x01	0x00	0x01	0x01

7.9.4 “Resetting SLK” application

This example allows the SLK to be reset in the *DISCONNECTED* link state. The actual link state here is not important. In addition, all preconnect settings and auto mode are deleted.

No.	Command-oriented data exchange	CMD	Data0	Data1	Data2	Data3	Data4	Data5	Data6
1	Switch SLK to <i>DISCONNECTED</i> state	0x06	0x00	0x03	0x00	0x01	0x00	0x01	0x02
6	Turn off SLK in auto mode	0x06	0x00	0x03	0x00	0x02	0x00	0x01	0x00
2	Stop pretransmit	0x08	0x05						
4	Finish setting pretransmit parameters	0x08	0x01						
3	Delete pretransmit data	0x08	0x04						
5	Finish setting parameters for unbuffered prefetch	0x0B	0x01						
4	Delete unbuffered prefetch	0x0B	0x04						
5	Finish setting parameters for buffered prefetch	0x0A	0x01						
5	Delete buffered prefetch	0x0A	0x04						
6	Wait until actual link state is <i>DISCONNECTED</i>								

ENGLISH

7.9.5 Handling “E00” errors

This example terminates direct data exchange with the MDT. The commanded link state *DISCONNECT* fails and the SLK switches to the *ERROR* state.

No.	Command-oriented data exchange	CMD	Data0	Data1	Data2	Data3	Data4	Data5	Data6
1	Switch SLK to <i>DISCONNECTED</i> state	0x06	0x00	0x03	0x00	0x01	0x00	0x01	0x02
2	Wait until actual link state is <i>DISCONNECTED</i> or <i>ERROR</i>								
3	DISCONNECT unsuccessful, link state is now <i>ERROR</i>								
4	Switch SLK to <i>DISCONNECTED</i> state	0x06	0x00	0x03	0x00	0x01	0x00	0x01	0x02
5	Wait until actual link state is <i>DISCONNECTED</i>								
6	The SLK is now in the initial <i>DISCONNECTED</i> state and the application can start over								

8 Interbus

8.1 Overview

Interbus was designed as an open fieldbus system when it was first released in 1987. At the beginning of 2000, IEC 61158 was adopted as a draft for an international fieldbus standard. Interbus was then set as the national standard (DIN 19258), then later adopted as the European standard (EN 50254). Literature on Interbus can be found under [I.] in Chapter 16.3 “References”.

- Bus topology: active ring, every remote bus user is a repeater
- Master-slave concept:
The interface module (master) is the central device for controlling the Interbus data ring. It exchanges the data serially transported in the data ring in fixed telegram lengths with the higher control or computer system and the subordinate Interbus users in both directions simultaneously and cyclically (full-duplex)
- Transmission rate: 500 kbps
- Max. 4,096 I/O points
- Bus length: 400 m (between two remote bus users), total length: 13 km
- Typical applications: general sensor/actuating systems, mechanical and systems engineering, process technology

The ID 40/SLK-IBS is a **remote bus** user (as opposed to local bus users). The term “remote bus” is also abbreviated as “RB”.

ID 40 uses the **PCP channel** and **process data channel** Interbus standard communication services to exchange data. The process data channel is used for event-oriented data exchange, the PCP channel for command-oriented data exchange.

The **I/O configuration** of the SLK is set in the device and cannot be modified. It indicates how many bytes are reserved for the process data area and the PCP channel segments.

The SLK display shows Interbus states, see Chapter 11.2.4 “Showing Interbus status on the display” for a description of the displays.

[I1] offers a general introduction to Interbus technology.

8.2 Addressing data in the ID 40 system

The SLK data areas are addressed in the Interbus SLK in the same way as with Profibus. See Chapter 7.7 “Addressing data in the ID 40 system”.

8.3 Command-oriented data exchange

Command-oriented data exchange uses the PCP channel **P**eripherals **C**ommunication **P**rotocol, see [12]). It is bidirectional and allows parameters to be set for event-oriented data exchange, as well as the transmission of data from and to data carriers/the SLK.

The following **functions** are supported:

- Writing data (MDT data only in *CONNECTED* state). This also includes, e.g., also switching the link state
- Reading data, e.g., also reading the actual link state
- Configuring unbuffered prefetch
- Configuring buffered prefetch

These functions are **communication objects** in the PCP channel and have various object indexes. An index is therefore the address of a communication object. Specific parameters of an object are represented by subindexes. The subindex 0 addresses the entire object.

Example: The object 5FF5 (**direct** reading and writing) has three subindexes. Subindex 1 contains the SLK address (4 bytes), subindex 2 the desired number of bytes (2 bytes) and subindex 3 the (octet) string of the bytes in the payload. If subindex 3 is read, the SLK first takes the data block indicated in subindexes 2 and 2 from the MDT and returns it as the octet string in subindex 3. If the Interbus master sends the payload in subindex 3 to the SLK, the data is entered in the data area indicated in subindexes 1 and 2.



Before the payload in subindex 3 is transferred, the desired data block in subindexes 1 and 2 has to be indicated.

Setting the parameters for the prefetch functions “unbuffered prefetch” and “buffered prefetch” allows multiple data block requests to be entered. Each command adds a new SLK data block specification to those already defined. When the prefetch is executed, the data blocks are read by the MDT in the specified order. All functions are listed in the table in Chapter 8.3.2 “PCP channel objects”.

8.3.1 PCP communication

Before data can be exchanged between two PCP Interbus users, a **logical connection** has to exist between them. These logical connections are referred to as **communication relationships**. Specific connection parameters for such a communication relationship are stored in what is known as the communication relationships list (**CRL**).

Each communication relationship contains a communication reference number (CR). This uniquely identifies the communication relationship. With the CR, the PCP function block of a PLC with an Interbus master module can access the individual devices via PCP communication.

Both the CRL and the CR are Interbus standard and are established when Interbus is started up (see Chapter 11.2 “Starting up Interbus”).

One CRL parameter - the maximum possible **PDU length** - can be modified on the SLK (see Chapter 11.2.2 “Configuring the maximum PDU size for the PCP channel”). The PDU (**P**rotocol **D**ata **U**nit) determines the maximum size of the SLK's transmit and receive buffer and with it the data packet that can be transmitted via PCP communication. This makes it possible to adjust the amount of MDT data (payload amount) transmitted in the application.

Table 20: Payload amount in every subindex 3 of every 5FF object depending on PDU size (npdu):

Object	Interbus data type	Payload amount in subindex 3 [bytes]
5FF4	0x0042	npdu - 7
5FF5	0x0041	npdu - 9
5FF6		

8.3.2 PCP channel objects

Object index	Communication object	Description	Subindexes	Format
5FF0	Set parameters for unbuffered prefetch	The addressed data area is added to the unbuffered prefetch. The data is included in the event-oriented data exchange in order of definition. This means the data defined first are added to the lowest address in the buffer regardless of its address in the data carrier	Subindex 1: SLK address	4 bytes
			Subindex 2: No. bytes, high byte always 0	2 bytes
5FF1	Configure unbuffered prefetch	Parameters for the unbuffered prefetch can set or all parameterized data areas can be deleted. The unbuffered prefetch is only executed once the parameters have been set	Subindex 0: 0: Begin setting parameters for unbuffered prefetch 1: Finish setting parameters for unbuffered prefetch 4: Delete unbuffered prefetch	2 bytes
5FF2	Set parameters for buffered prefetch	The addressed data area is added to the buffered prefetch. Once the prefetch is executed, the data is placed in the prefetch buffer in the order of the commands. This means the data defined first are added to the lowest address in the buffer regardless of its address in the data carrier	Subindex 1: SLK address	4 bytes
			Subindex 2: No. bytes, high byte always 0	2 bytes
5FF3	Configure buffered prefetch	Parameters for the buffered prefetch can set or all parameterized data areas can be deleted. The buffered prefetch is only executed once the parameters have been set	Subindex 0: 0: Begin setting parameters for buffered prefetch 1: Finish setting parameters for buffered prefetch 4: Delete buffered prefetch	2 bytes
5FF4	Read prefetch buffer	The data block indicated in subindexes 1 and 2 are read from the prefetch buffer. Subindex 1 is not an SLK address, but addresses the start of the data block in the prefetch buffer	Subindex 1: Byte address in prefetch buffer	2 bytes
			Subindex 2: No. bytes, high byte always 0	2 bytes
			Subindex 3: Payload	n bytes
5FF5	Read and write data directly	Reading subindex 3 returns the data in the data block indicated in subindexes 1 and 2. Writing to subindex 3 triggers a data entry in the data block indicated in subindexes 1 and 2.	Subindex 1: SLK address	4 bytes
			Subindex 2: No. bytes, high byte always 0	2 bytes
			Subindex 3: Payload	n bytes
5FF6	Set parameters for pretransmit	The indicated data area is added to the pretransmit. This command can be selected multiple times, whereby the various areas can overlap	Subindex 1: SLK address	4 bytes
			Subindex 2: No. bytes, high byte always 0	2 bytes
			Subindex 3: Data in pretransmit to be written	n bytes

Object index	Communication object	Description	Subindexes	Format
5FF7	Configure pretransmit	Begin or finish setting pretransmit parameters. Start “single transmit” (ST) or “multiple transmit” (MT). Delete all parameterized data areas. Stop pretransmit	Subindex 0: 0: Begin setting parameters 1: Finish setting parameters 2: Start ST 3: Start MT 4: Delete parameters 5: Stop pretransmit	2 bytes

Subindexes of objects that should transmit user data to the SLK have to be completely filled with data. The number of bytes that should be entered in the system can be lower and has to be entered in each “No. bytes” subindex. Unassigned bytes in the process data channel transfer the value 0.

8.3.3 Example: writing bytes to MDT

Six bytes with the values 1 through 6 are written to the MDT starting at the address 0x1234. The data is entered in the MDT once subindex 3 has been sent.



If the PDU size is set to 240 bytes, 231 bytes have to be assigned in subindex 3.

No.	Command-oriented data exchange	Index Subindex	Data1	Data2	Data3	Data4	Data5	...	Data55
1	Enter MDT address	5FF5-01	0x00	0x00	0x12	0x34			
2	Enter byte count	5FF5-02	0x06						
3	Enter byte values	5FF5-03	0x01	0x02	0x03	0x04	0x05	0x06	0x00

8.4 Event-oriented data exchange

Event-oriented data exchange transmits data provided by the unbuffered prefetch (Chapter 6.4.1.1 “Unbuffered prefetch”). The event-oriented data is purely input data for the application and transmitted through the Interbus **process data channel**.

The process data channel (PD channel) is mapped in the I/O field of the application (e.g., PLC) and is the default communication with general I/O bus users.

In the ID 40/SLK-IBS, the PD channel consists of six data words, or 96 bits. The assignment of the process data channel is described in the following sections.

Unassigned bytes in the PD channel transfer the value 0.

8.4.1 Status information in the process data channel

The **first two bytes** in the event-oriented data exchange are status bytes with the following structure:

Table 21: Status byte 0 of event-oriented data exchange

Byte 0 (bits 0–7)							
7	6	5	4	3	2	1	0
Event counter				Error code			

Bits 0 to 3 comprise the result code. The coding for the error code is the same as for command-oriented data exchange (see Chapter 8.7 “Interbus error codes”).

Bits 4 to 7 comprise the **event counter**. The counter runs cyclically from 0 to 15 and is increased by 1 when the current *PRECONNECTED* link state is ended. This signals to the control system that the data from the prefetch parameterization is updated. At this point, the user program can import prefetch data. The counter also increases when no prefetch is parameterized.

The following functions have been completed when the counter value changes:

- Pretransmit data has been entered in the MDT (if parameterized)
- Prefetch data was read from the MDT (if parameterized) and is ready to be imported to a PLC DB
- If auto reconnect is active, the MDT was closed and the link state is *CONNECTING*
- If auto disconnect is active, the MDT was closed and the link state is *DISCONNECTED*
- If no auto mode is active, the link state is *CONNECTED*. Data can be exchanged with the MDT.

Table 22: Status byte 1 of event-oriented data exchange

Byte 1 (bits 0–7)							
7	6	5	4	3	2	1	0
Actual link state				Data transfer status	Prefetch param. open	Prefetch parameterized	Operative flag

Bit 0 shows the status of the SLK operative flag, see Chapter 5.3.4. “SLK operative flag”)

Bits 1 and 2 show the current status of prefetch parameterization. The flags apply both to the unbuffered prefetch function and the buffered prefetch.

Bit 1 Prefetch is parameterized:

‘0’ = no prefetch currently parameterized in system

‘1’ = **at least one** unbuffered prefetch **or one** buffered prefetch is parameterized

Bit 2 Prefetch parameterization is open:

“0” = prefetch parameterization is closed. If prefetches have already been parameterized, these are executed once the system completes the *PRECONNECTED* state.

‘1’ = prefetch parameterization is open. No prefetches are executed in this state.

Bit 3 Data transfer status (see Chapter 6.6 “‘Open transfer’ and ‘Close transfer’ functions”):

‘1’ = “Open transfer” command executed

‘0’ = “Close transfer” command executed



The value of bit 3 does **not** return the accessibility of MDT data, since the transfer channel function is no longer active. The bit remains for the sake of compatibility with old applications. Bit 3 remains “0” when the “Open” and “Close” commands are not used.

Bits 4 to 7 show the actual link state (see Chapter 5.3.1 “Actual link state”); this value does not have to be entered separately in the prefetch.

8.4.2 Data array for event-oriented data channel

The SLK provides the data from the unbuffered prefetch starting with the **third byte** of the event-oriented data exchange. If no prefetch was executed, the value of each byte is zero.

8.5 Commanded link state via process data channel

Data is entered in the SLK using PCP communication (see Chapter 8.3 “Command-oriented data exchange”). Since PCP transfer takes a lot of time to transfer a byte, the link request (see Chapter 5.3.2 “Commanded link state”) can also be sent to the SLK through the process data channel. The outputs in the PLC are used to do this (the inputs are used for event-oriented data exchange, see Chapter 8.4 “Event-oriented data exchange”).

A trigger bit is used to enter the commanded link state in the SLK since the I/Os are permanently exchanged through the PD channel. The trigger bit is not returned in the input map (event-oriented data channel). The actual link state (see Chapter 5.3.1 “Actual link state”) indicates the result of the commanded link state. A signal handshake via the trigger bit is therefore unnecessary.

8.5.1 Output map structure

Table 23: Output byte 0 of output map

Byte 0 (bits 0–7)							
7	6	5	4	3	2	1	0
Trigger bit = 7, bits 4–6 reserved				Commanded link state			

When switching **bit 7** (from 0 to 1 or 1 to 0), the value of bits 0 – 3 is entered in the SLK commanded link state and executed.

SLK ignores bits 4 – 6.

The SLK also ignores the **remaining 11 output bytes** of the PD channel.

The trigger bit and the commanded link state value can be entered in the same PLC cycle (simultaneously). Data consistency is ensured within a byte.

8.5.2 Sample sequence

In this example, the commanded link state CONNECT is first sent to the SLK, followed by DISCONNECT.

1. Enter the value for CONNECT in the lower half-byte (bits 0 – 3) and switch the trigger bit.

This puts the commanded link state CONNECT into effect in the SLK.

...

2. Enter the value for DISCONNECT in the lower half-byte and switch the trigger bit again.

The new commanded link state DISCONNECT is implemented.

8.6 SLK address table

The address information is start addresses for the data areas described in Chapter 5 “SLK storage” and are needed for data exchange via Interbus. The ID 40 system supports the bytes data format (unsigned char or UNSIGNED8). The application can implement more complex data formats.

Table 24: SLK address table

Segment	Addresses	Data format	Access*	Meaning
0x0000	MDT user data area			
	0x00000000	Area length (byte number) depending on MDT type, see Chapter 4.1 “ID 40/MDT storage”	rd/wr	Area for user-specific data, not interpreted by system
0x0001	- Reserved -			
0x0002	MDT register area			
	0x00020000	High byte, low byte	rd/rst	MDT status
	0x00020002	High byte, low byte	rd/wr	MDT pointer 1
	0x00020004	High byte, low byte	rd/wr	MDT pointer 2
	0x00020006	High byte, low byte	rd/wr	MDT pointer 3
	0x00020008	Highest byte - lowest byte	rd only	MDT ID code, 4-digit
	0x0002000C	High byte, low byte start address	rd/wr	MDT formatting
	0x0002000E	High byte, low byte length (byte count)		The MDT memory is overwritten with the formatting value starting with the start address. See Chapter 5.2.5 “MDT formatting”
	0x00020010	Byte formatting value		
	0x00020011			- Reserved -
	0x00020012	High byte, low byte	rd only	Internal MDT work pointer (for diagnostic purposes only)
	0x00020014		rd only	MDT software version number
	0x00020015			- Reserved -
0x0003	SLK register area			
	0x00030000	Byte	rd only	Actual link state
	0x00030001	Byte	rd/wr	Commanded link state
	0x00030002	Byte	rd/wr	Auto mode configuration
	0x00030003	Byte	rd/wr	SLK operative flag
	0x00030004	Highest byte - lowest byte	rd/wr	MDT counter, 4-digit
	0x00030008	16 bytes ASCII	rd only	SLK device name
	0x00030018	Highest byte - lowest byte	rd only	SLK software version, 4-digit version ID
	0x0003001C	High byte, low byte	rd/wr	MDT pointer 1 Look Ahead Control register
	0x0003001E	High byte, low byte	rd/wr	MDT pointer 2 Look Ahead Control register
	0x00030020	High byte, low byte	rd/wr	MDT pointer 3 Look Ahead Control register

- * rd only Read only, no writing possible.
 rd/wr Read and write access.
 rst Register is reset by writing any value.
 none No access.

8.7 Interbus error codes

The Interbus error codes for **PCP** communication are divided into **error class**, **error number** and a device-specific error code called **additional error code**. The meaning of Error_Class and Error_Code are described in [I2].

The following table lists the additional error codes specific to the ID 40.

Table 25: Interbus device-specific additional error codes

Result	Meaning
0	No error
1	Unexpected break in communication between SLK and MDT
2	MDT device error, e.g., memory area defective
3	Error in SLK addressing, e.g., start address and data length exceed limit of an SLK address segment Access to nonexistent SLK address segment
4	Data length in index/subindex too long
5	Data length in index/subindex too short
6	- Reserved -
7	Attempt to set parameters for prefetch although parameterization was not started. Attempt to set parameters for pretransmit although parameterization was not started. Prefetch buffer read although buffered prefetch parameterization has started
8	Command operand value outside permitted range, e.g., for command 15
9	Wrong index/subindex
15	Internal system error, e.g., memory shortage

These error codes also characterize errors that can occur in event-oriented data exchange (see Chapter 8.4.1 “Status information in process data channel”).

Example: An incorrectly parameterized prefetch resulted in a break in communication during execution.

8.8 Sample applications

The following examples show some typical applications of the ID 40 system using Interbus.

8.8.1 “Manual workstation” application

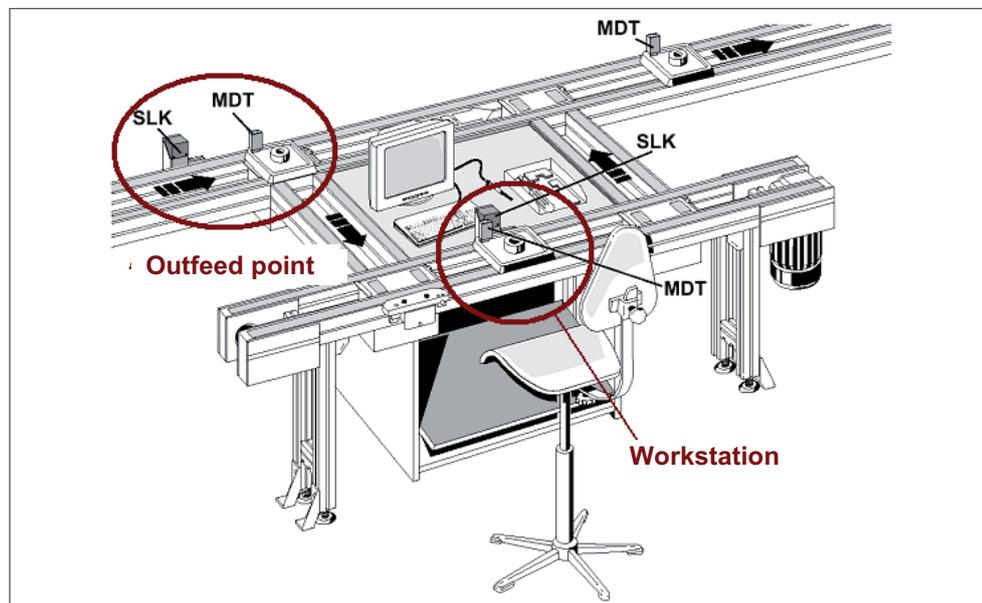


Fig. 26: Sample “manual workstation” application

The MDT should be read and written at a processing station. First, 32 bytes should be read from the MDT address 0x0110, then 50 bytes should be written from the MDT address 0x0100. The MDT status is also transmitted in order to allow any errors in the MDT to be addressed.

The actual link state is read in an event-oriented manner to determine whether or not a new MDT is in the link. Since the link state is contained in the event-oriented data channel by default (see Chapter 8.4 “Event-oriented data exchange”), the SLK does not need an additional configuration command.

The remaining data is exchanged with the MDT by command in the *CONNECTED* state. The MDT then signs off with the commanded link state *RECONNECT*. At the same time, the SLK switches to the *CONNECTING* state to wait for the next MDT. Command-oriented data exchange occurs in Interbus through PCP communication.

SLK preparation

The link *CONNECTING* is configured through the process data channel. See Chapter 8.5 “Commanded link state via process data channel”.

SLK operation

Check the actual link state until the value *CONNECTED* appears (2). A new MDT has now arrived. The controller reads the MDT status (4). When the MDT status does not indicate an error, the user data can now be read starting with MDT address 0x0110. Once this data has been evaluated and the workpiece processed, the result of the processing is written to the MDT (6). Communication with the MDT is then terminated (8).

No.	PCP communication	Index	Subindex 0	Subindex 1	Subindex 2	Subindex 3
2	Wait until actual link state in process data channel is <i>CONNECTED</i>					
4	Read MDT status and check if error bits are present	0x5FF5		0x00020000	0x0001	Read data
5	Read data from MDT, e.g., 32 bytes as of address 0x0110	0x5FF5		0x00000110	0x0020	Read data
6	Write data to MDT, e.g., 50 bytes as of address 0x0100	0x5FF5		0x00000100	0x0032	Byte1 Byte2 ... Write data
8	Close MDT (RECONNECT)	Via process data channel: Byte 0: 0x03 (switch trigger bit: now 0, used to be 1)				
9	Wait until actual link state in process data channel is <i>CONNECTED</i>					

8.8.2 “Outfeed” application

Outfeed points are where it is decided whether or not the workpiece pallet is diverted. The workpiece pallets do not need to stop at the SLK before the junction, since the amount of data needed for the decision is small. In this example, 8 bytes should be read starting at MDT address 0x02F0. The MDT status is also transmitted in order to allow any errors in the MDT to be addressed.

The data is read from the MDT through an **unbuffered prefetch**, then the link immediately switches from *PRECONNECTED* to *CONNECTING*, since auto reconnect is active.

SLK preparation

No.	PCP communication	Index	Subindex 0	Subindex 1	Subindex 2	Subindex 3
1	Deactivate unbuffered prefetch so no parameters are set	0x5FF1	0x0000			
2	Delete old prefetch entries	0x5FF1	0x0004			
3	Set parameters for first data block in unbuffered prefetch: 1-byte MDT status	0x5FF0		0x00020000	0x0001	
4	Set parameters for second data block in unbuffered prefetch: 8 bytes starting from MDT address 0x02F0	0x5FF0		0x000002F0	0x0008	
5	Finished setting parameters, activate unbuffered prefetch	0x5FF1	0x0001			
6	Switch SLK to auto reconnect mode	0x5FF5		0x00030002	0x0001	0x01 Write data
7	Make SLK ready to receive (CONNECT)	Via process data channel: Byte 0: 0x81 (switch trigger bit: now 1, used to be 0)				
8	SLK operation Wait until event counter in event-oriented data channel (see Chapter 7.3.1 “Status information in event-oriented data channel”) to increase by 1. At this time, a workpiece pallet has arrived, event-oriented data exchange has occurred with the MDT and the link for the next MDT is back to <i>CONNECTING</i> . The MDT status and the application data can be checked					

The requested data bytes are sent to the fieldbus master through the event-oriented data exchange for every MDT. The data can be provided for additional evaluation using a controller function block.

8.8.3 “Workstation” application

This example is for a workstation with a lift positioning unit. Since a link cannot be established between the SLK and the MDT when the workpiece pallet is raised, data is exchanged in two stages.

Stage 1: Reading the workpiece-specific data to check whether or not this workpiece can be processed at this station. The data is read from the MDT in an event-oriented manner. The SLK then automatically switches to *DISCONNECTED*. Only when the lift positioning device lowers after processing does the SLK switch to *CONNECTING*. This eliminates the risk of an unexpected break in communication as the MDT approaches the SLK from above.

Stage 2: Entering measurement data obtained while the workpiece was being processed. Only the pretransmit function is used here.

SLK preparation

An unbuffered prefetch is configured that transmits both the MDT status (see Chapter 4.3.2.1 “MDT status register”) and the control data (e.g., 9 bytes as of MDT address 0x0024) in the event-oriented data channel. The SLK switches to auto disconnect mode and the link state switches to *CONNECTING*.

No.	PCP communication	Index	Subindex 0	Subindex 1	Subindex 2	Subindex 3
1	Deactivate unbuffered prefetch so no parameters are set	0x5FF1	0x0000			
2	Delete old prefetch entries	0x5FF1	0x0004			
3	Set parameters for first data block in unbuffered prefetch: 1-byte MDT status	0x5FF0		0x00020000	0x0001	
4	Set parameters for second data block in unbuffered prefetch: 8 bytes starting from MDT address 0x0024	0x5FF0		0x00000024	0x0009	
5	Finished setting parameters, activate unbuffered prefetch	0x5FF1	0x0001			
6	Switch SLK to auto disconnect mode	0x5FF5		0x00030002	0x0001	0x02 Write data
7	Make SLK ready to receive (CONNECT)	Via process data channel: Byte 0: 0x01 (switch trigger bit: now 0, used to be 1)				
8	SLK operation Wait until event counter in event-oriented data channel (see Chapter 7.3.1 “Status information in event-oriented data channel”) to increase by 1. At this point, a workpiece pallet has arrived, the event-oriented data is valid and the link is <i>DISCONNECTED</i> . The MDT status and control data can be checked and the workpiece pallet can be raised.					

Stage 2: If measurement data is available during processing, the parameters for pretransmit are set (e.g., 80 bytes as of MDT address 0x01C0) and configured as a **single transmit** (see Chapter 6.4.2 “Pretransmit”). By this time, the workpiece pallet has lowered. The SLK switches to *CONNECTING* and the MDT immediately signs on and executes the pretransmit.

No.	PCP communication	Index	Subindex 0	Subindex 1	Subindex 2	Subindex 3
1	Begin setting pretransmit parameters	0x5FF1	0x0000			
2	Delete pretransmit data from last MDT	0x5FF1	0x0004			
3	Set parameters for 80-byte measurement data block as pretransmit	0x5FF0		0x00020000	0x0001	
1	Finish setting pretransmit parameters	0x5FF1	0x0001			
1	Activate single transmit	0x5FF1	0x0002			
4	Wait until lift positioning unit in lower position					
5	Make SLK ready to receive (<i>CONNECT</i>)	Via process data channel: Byte 0: 0x81 (switch trigger bit: now 1, used to be 0)				
6	Wait until event counter in event-oriented data channel increases by 1. At this time, the measurement data is written to the MDT and the link is <i>DISCONNECTED</i> (auto reconnect is still active)					
7	Make the SLK ready to receive the next workpiece pallet only once the current workpiece pallet is gone (<i>CONNECT</i>)	Via process data channel: Byte 0: 0x01 (switch trigger bit: now 0, used to be 1)				

8.8.4 “Resetting SLK” application

This example deletes all settings and switches the SLK to the *DISCONNECTED* link state. The actual link state here is not important.

No.	PCP communication	Index	Subindex 0	Subindex 1	Subindex 2	Subindex 3
2	Finish setting pretransmit parameters	0x5FF1	0x0001			
2	Stop pretransmit	0x5FF1	0x0005			
2	Delete pretransmit data	0x5FF1	0x0004			
2	Finish setting parameters for buffered prefetch	0x5FF2	0x0001			
2	Delete buffered prefetch	0x5FF2	0x0004			
2	Finish setting parameters for unbuffered prefetch	0x5FF1	0x0001			
2	Delete unbuffered prefetch	0x5FF1	0x0004			
6	Turn off auto reconnect mode	0x5FF5		0x00030002	0x0001	0x00
3	Send command- ed link state <i>DISCONNECT</i>	Via process data channel: Byte 0: 0x02 (switch trigger bit: now 0, used to be 1)				
4	Wait until actual link state is DISCONNECTED					

8.8.5 Handling “E00” errors

This example shows how an unexpected E00 error is handled when completing direct data exchange with the MDT. The commanded link state *DISCONNECT* fails and the SLK switches to the *ERROR* state.

No.	PCP communication	Index	Subindex 0	Subindex 1	Subindex 2	Subindex 3
1	Send commanded link state <i>DISCONNECT</i>		Via process data channel: Byte 0: 0x82 (switch trigger bit: now 1, used to be 0)			
2	Wait until actual link state is <i>DISCONNECTED</i> or <i>ERROR</i>					
3	DISCONNECT unsuccessful, link state is now <i>ERROR</i>					
4	Send commanded link state <i>DISCONNECT</i>		Via process data channel: Byte 0: 0x02 (switch trigger bit: now 0, used to be 1)			
5	Wait until actual link state is <i>DISCONNECTED</i>					
6	The SLK is now in the initial <i>DISCONNECTED</i> state and the application can start over					

9 CANopen

9.1 Overview

The ID 40/SLK-CAN is a CANopen slave. The SLK comes with nearly every data transmission mechanism specified for CANopen and can be activated by any bus master in compliance with CANopen.

All specifications and guidelines for CANopen can be found in the CiA (CAN in Automation e.V.) specifications. Literature on CAN can be found under [C.] in Chapter 16.3 "References".

Data is exchanged between CANopen users exclusively through **objects** addressed by a 16-bit **index** and an 8-bit **subindex**. These objects are compiled into an **object directory**. For example, the data areas described in Chapter 5 "SLK storage" are also mapped by objects.

CANopen provides **communication services** (also known as communication objects) for data exchange. Command-oriented SLK data exchange occurs via **SDO** services (**S**ervice **D**ata **O**bject), and event-oriented data exchange via **PDO** services (**P**rocess **D**ata **O**bject).

9.2 Object directory

The object directory contains not only user data, but also CANopen communication objects, device profiles and even data types. Default values and access rights to the objects are also set there. See [C17] for reference.

[C17] determines objects that have to be in a CANopen slave and those that are optional.

Table 26: Distribution of CANopen object directory index ranges as per [C17]

Index	CANopen objects	ID 40/SLK-CAN
0000	Not used	Not used
0001-001F	Static data types	All used
0020-003F	Complex data types	0020 - 0023
0040-005F	Manufacturer-specific complex data types	0040 - 0043
0060-007F	Device profile-specific static data types	Not used
0080-009F	Device profile-specific complex data types	Not used
00A0-0FFF	Reserved	Not used
1000-1FFF	Communication profile area	Used in part
2000-5FFF	Manufacturer-specific profile area	Used in part
6000-9FFF	Standardized device profile area	Not used
A000-FFFF	Reserved	Not used

All object index and subindexes are written in this manual in **hexadecimal** format without prefix or suffix.

The following example shows the object name of the actual link state **2600/01**. The index is 2600 [hex] and the subindex is 01 [hex].

The **EDS file** (see Chapter 9.18 “Electronic data sheet [EDS]”) defines the object directory for the ID 40 system. The application obtains information from the EDS file that provides objects in the ID 40 system.

9.3 Command-oriented data exchange

Command-oriented data exchange occurs via SDO. An SDO is a confirmed data transmission, i.e., the execution of an SDO is reported. If, e.g., an SDO reports “OK” after entering data in the MDT, it is ensured that the data was entered in the MDT memory.

The SDO service allows unrestricted access to the ID 40 system and supports the following functions:

- Writing data, e.g., MDT data in *CONNECTED* state or the link state switch request
- Reading data, e.g., MDT data in *CONNECTED* state or reading the actual link state
- Configuring prefetch and pretransmit (see Chapter 9.5 “MDT data transmission with transfer buffers”)
- Configuring event-oriented data exchange
- Configuring and running diagnostics for the SLK on CAN, e.g., communication parameters

To use the SDO service, a PLC with a CANopen master module provides SDO function blocks. This allows the data to be read and written by command. The application and parameters of such function blocks depend on the PLC or module and are documented in the relevant manuals.

One parameter needed for SDO communication is the **object index**. Other parameters are **data source** or **data sink**. The SDO takes the bytes transmitted to the SLK from the data source, e.g., a data block. Data read from the SLK via SDO is entered by the function block in the data sink, usually also a data block in the PLC.

9.4 Event-oriented data exchange

During event-oriented data exchange, the ID 40 system sends objects whose value has currently changed.

Example: MDT data read with the prefetch function. This allows the application to be automatically notified of an MDT entering the SLK field.

PDO services are available for event-oriented data exchange (see Chapter 9.9 “PDO communication”). Contrary to SDOs, PDOs are messages that are not confirmed by the recipient.

A **transmit PDO** (TPDO) transmits objects from the SLK to the PLC, e.g., read MDT data or the actual link state. The SLK sends a **TPDO** when the content of an object assigned to the TPDO changes. This assignment is known as **PDO mapping**. If the content of an object mapped in TPDO1 changes, e.g., data read from the MDT, the SLK sends the TPDO1 with the MDT data to the master.

A **receive PDO** (PDO) transmits objects from the PLC to the SLK, e.g., setting the commanded link state. This means CANopen masters that have no implemented SDO communication can also work with the ID 40 system. However, the aspects described in Chapter 9.9 “PDO communication” have to be observed.

The default mapping for all PDOs is described in Chapter 9.9.1 “PDO mapping”. PDO mapping can be changed by command-oriented data exchange.

Detailed PDO transmission conditions are explained in Chapters 9.9.2 “PDO communication parameters” and 9.9.3 “PDO transmission types”.

9.5 MDT data transmission with transfer buffers

The ID 40/SLK-CAN provides **four transfer buffers** for flexible data exchange. The payload areas in the transfer buffers can be used both for direct data exchange with the MDT as well as for parameterized data exchange. The content of an entire MDT can be saved in each buffer.

- Multiple payload blocks of any size can be transferred directly from and to the MDT through a command.
- Parameterized data exchange (see Chapter 6.4 “Parameterized data exchange with MDT”) is supported by all four buffers independently of one another. Buffered prefetch and pretransmit are available.
- The transfer buffers can be activated and deactivated at will to select specific MDTs. These have to be selected before the MDT arrives.
- The contents of up to four MDTs can be buffered.
- Status information on all transfer buffers can be transmitted to the bus master through event-oriented data exchange.

The functionality of each transfer buffer forms four objects, the **buffer status** object, the **buffer mode** object, the **buffer mapping** object and the **transfer buffer** itself.

The following diagram shows the relationships between the buffer objects.

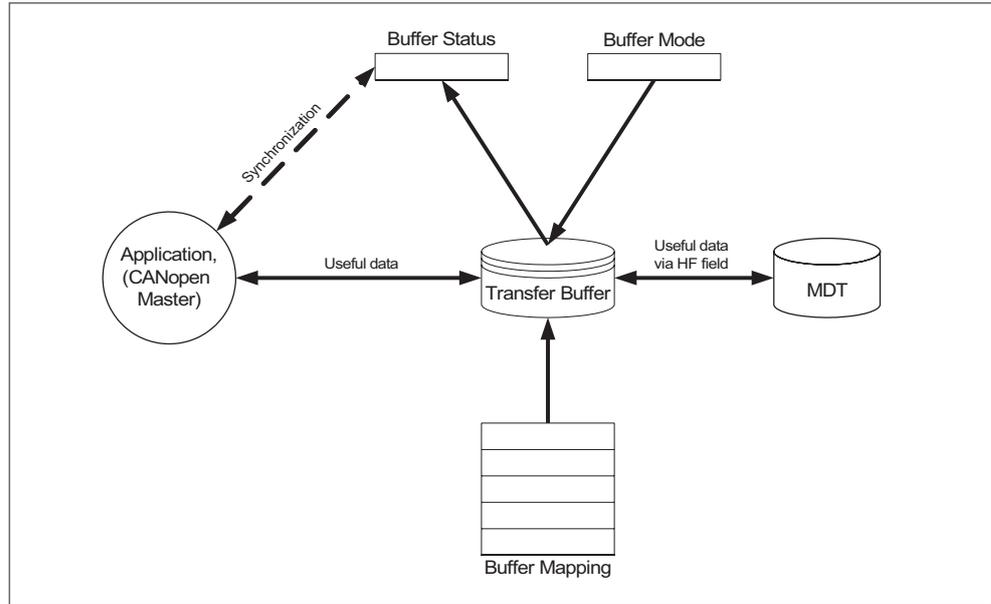


Fig. 27: Interdependencies between transfer buffer objects

9.5.1 Transfer buffer parameters (objects 2100–2103)

These parameters are available in the 210x objects, whereby x + 1 stands for the number of each transfer buffer.

The modes the transfer buffers should use and whether or not each buffer is active can be configured in the **buffer mode register** (subindex 01) for these objects. The data type of the buffer mode register is UNSIGNED8.

The **buffer status register** (subindex 02) provides information on, e.g., whether data transmission is in progress or has completed. This register can be included in TPDO mapping in order to transmit the information by event. The data type of the buffer status register is UNSIGNED8.

Table 27: Buffer mode register 210x/01

Bit	7	6	5	4	3	2	1	0
		X				M		V

V: Buffer valid

- 0 =** Transfer buffer locked, no SLK or MDT data exchange. In this state, the buffer mapping can be edited.
- 1 =** Transfer buffer active. The buffer is working according to the configured buffer mode. The buffer mapping cannot be edited.

M: Buffer mode

- 0 =** Direct MDT data exchange
- 1 =** Prefetch, see Chapter 6.4.1.2 “Buffered prefetch”
- 2 =** Pretransmit: single transmit, see Chapter 6.4.2.1 “Single transmit”
- 3 =** Pretransmit: multiple transmit, see Chapter 6.4.2.2 “Multiple transmit”
- 4–7 =** Not used, ignored.

X: Reserved, set to 0

Table 28: Buffer status register 210x/02

Bit	7	6	5	4	3	2	1	0
			X				S	

S: Buffer status**0** = Data not yet transmitted**1** = Data exchange in progress**2** = Data exchange complete**3** = Data exchange interrupted due to error More information can be found in Chapter 9.17.4 “Pre-defined error field (object 1003)”**4–7** = Not used, ignored.**X:** Reserved, set to 0**9.5.2 Object mapping in the transfer buffer (objects 2110–2113)**

The **buffer mappings** are available in the 211x objects, whereby x+1 stands for the number of each transfer buffer.

The buffer mapping determines the **data blocks** where the payload of the transfer buffer is entered or read. See Chapter 5 “SLK storage” for the meaning of data blocks.

Up to 254 data blocks can be assigned to one transfer buffer. A data block is determined by the following:

Table 29: Buffer mapping 211x/01 to 211x/FE

Bit	31	...	14	13	...	0
	Address			Byte count		

The data blocks are specified in accordance with Chapter 7.7 “Addressing data in the ID 40 system”. However, since the address/byte count are coded together with a 32-bit value, the SLK data is reduced to 18 bits (bit 14 to bit 31), and the byte count to 14 (bit 0–13). Despite this restriction to the value ranges, all areas described in Chapter 5 “SLK storage” can be accessed.

Address and byte count are combined and entered into the 32-bit value of the mapping object by multiplying the address value by 0x4000 and adding the byte count to the result.

The address ranges of the SLK can be found in Chapter 9.5.3 “SLK address table”. The number of specified data blocks are found in subindex 00 of the buffer mapping object. When the number is 0, the transfer buffer is deactivated since there are no mappings.

The following figure shows the relationships between transfer buffer, buffer mapping and MDT memory. The data blocks distributed throughout the MDT are seamlessly strung together in the buffer object. The data blocks can only be reassigned using the mapping information.

The SLK verifies the total length of the transfer buffer using the sum of the byte counts in the mapping object. Any deviations trigger a corresponding error message.

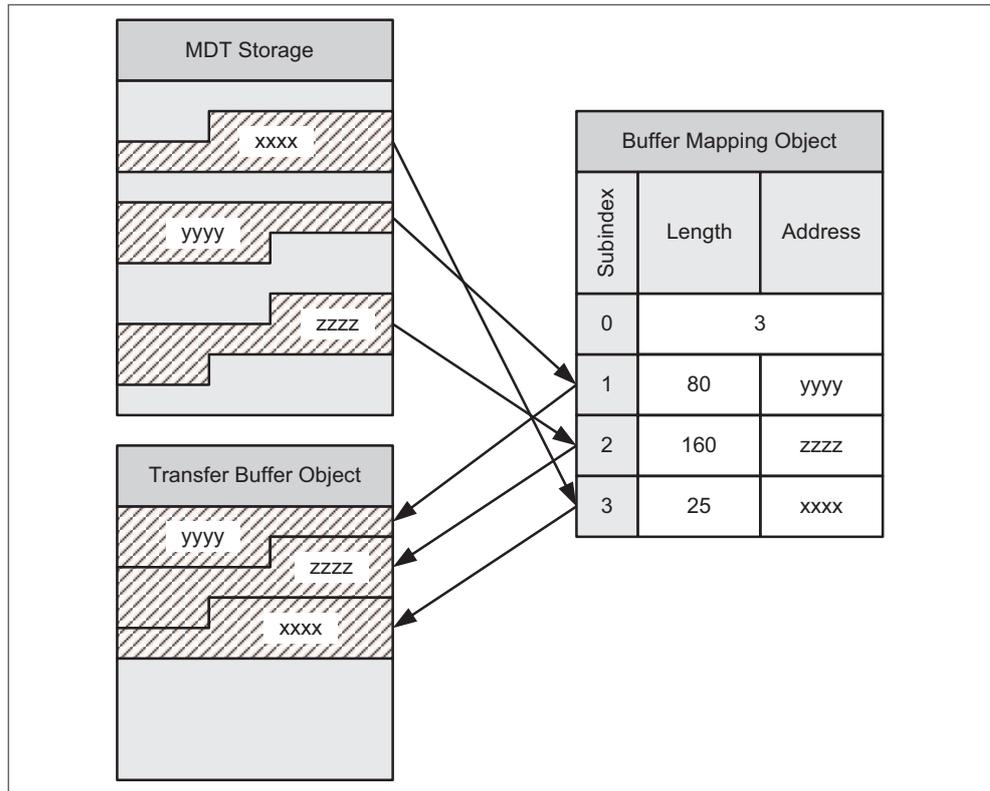


Fig. 28: Mapping MDT data in the transfer buffer

9.5.3 SLK address table

The address information is start addresses for the data areas described in Chapter 5 “SLK storage” and are needed for mapping the transfer buffer.

The ID 40 system supports the bytes data format (unsigned char or UNSIGNED8). The application can implement more complex data formats.

Table 30: SLK address table

Segment	Addresses	Data format	Access*	Meaning
0x0000	MDT user data area			
	0x00000000	Area length (byte number) depending on MDT type, see Chapter 4.1 “ID 40/MDT storage”	rd/wr	Area for user-specific data, not interpreted by system
0x0001	- Reserved -			
0x0002	MDT register area			
	0x00020000	High byte, low byte	rd/rst	MDT status
	0x00020002	High byte, low byte	rd/wr	MDT pointer 1
	0x00020004	High byte, low byte	rd/wr	MDT pointer 2
	0x00020006	High byte, low byte	rd/wr	MDT pointer 3
	0x00020008	Highest byte - lowest byte	rd only	MDT ID code, 4-digit
	0x0002000C	High byte, low byte start address	rd/wr	MDT formatting
	0x0002000E	High byte, low byte length (byte count)		The MDT memory is overwritten with the formatting value starting with the start address. See Chapter 5.2.5 “MDT formatting”
	0x00020010	Byte formatting value		
	0x00020011			- Reserved -
	0x00020012	High byte, low byte	rd only	Internal MDT work pointer (for diagnostic purposes only)
	0x00020014		rd only	MDT software version number
	0x00020015			- Reserved -
0x0003	SLK register area			
	0x00030000	Byte	rd only	Actual link state
	0x00030001	Byte	rd/wr	Commanded link state
	0x00030002	Byte	rd/wr	Auto mode configuration
	0x00030003	Byte	rd/wr	SLK operative flag
	0x00030004	Highest byte - lowest byte	rd/wr	MDT counter, 4-digit
	0x00030008	16 bytes ASCII	rd only	SLK device name
	0x00030018	Highest byte - lowest byte	rd only	SLK software version, 4-digit version ID
	0x0003001C	High byte, low byte	rd/wr	MDT pointer 1 Look Ahead Control register
	0x0003001E	High byte, low byte	rd/wr	MDT pointer 2 Look Ahead Control register
	0x00030020	High byte, low byte	rd/wr	MDT pointer 3 Look Ahead Control register

- * rd only Read only, no writing possible.
 rd/wr Read and write access.
 rst Register is reset by writing any value.
 none No access.

9.5.4 Transfer buffers (objects 2120–2123)

212x objects are available to transfer buffers for transporting payloads, whereby $x + 1$ stands for the number of each transfer buffer. The data type of the objects is DOMAIN and they can only be read or written by command.

9.5.5 Direct data exchange

It is possible to directly exchange data with an MDT when it is in the *CONNECTED* link state (see Chapter 2.2.1 “SLK operating states”).

To do this, the buffer mode (subindex 01) in the buffer parameter object of a selected transfer buffer has to be set to 0x00 and the desired data blocks entered in the corresponding buffer mapping object.

Reading MDT data: If the CANopen master initiates an SDO upload of the selected transfer buffer, the SLK retrieves the data blocks indicated in the buffer mapping from the MDT to execute the SDO request.

Writing MDT data: If the CANopen master initiates an SDO download of the selected transfer buffer, the SLK enters the transmitted data in the MDT according to the data areas indicated in the buffer mapping.

During the data exchange, the buffer status register shows the value 0x01, then 0x02 once the exchange is complete. If the buffer status register is mapped in a TPDO channel, TPDOs are also sent during direct data exchange when the buffer status changes.

9.5.6 Prefetch data exchange

Prefetch automatically runs in the *PRECONNECTED* link state, see Chapter 2.2.1 “SLK operating states”.

To do this, the buffer mode (subindex 01) in the buffer parameter object of a selected transfer buffer has to be set to 0x01 and the desired MDT data blocks entered in the corresponding buffer mapping object.

Once an MDT signs on, the mapped payload is automatically read from the MDT in the *PRECONNECTED* link state. Once executed, the data is in the transfer buffer and the buffer status switches from 0x01 to 0x02. At this point, the SLK sends a TPDO to the bus master (if the buffer status is mapped) and the application can then retrieve the transfer buffer by command (via SDO upload).

9.5.7 Pretransmit data exchange

Pretransmit automatically runs before any configured prefetch in the *PRECONNECTED* link state, see Chapter 2.2.1 “SLK operating states”.

The buffer mode of a selected transfer buffer indicates whether a single transmit (buffer mode 2) or a multiple transmit (buffer mode 3) should be executed. The data blocks designated for the MDT are entered consecutively in the corresponding transfer buffer. The MDT addresses to which and length with which the individual data blocks should be transferred is indicated in the buffer mapping.

It is also possible to only transmit one MDT data block with a transfer buffer for better clarity. Up to four data blocks can be written in this manner.

If the buffer status value is 0x02, the data was entered in the MDT.

9.6 ID 40 objects (manufacturer-specific profile area)

The manufacturer-specific profile area is reserved in CANopen for device-specific objects in the index range 2000–5FFF.

This area also contains all **ID 40-specific objects**. The following table lists all of the objects available in the ID 40 system. Detailed object descriptions can be found in the appropriate chapters.

Table 31: ID 40-specific objects

Object index	Subindex	Object code ¹⁾	Name	Data type	Acc ²⁾	S ³⁾
Manufacturer-specific profile area						
2020		ARRAY	SDO timeout client, see Chapter 9.8.3 “SDO timeout client (object 2020)”	Unsigned32	RW	Y
2030		RECORD	Transmit PDO 1 status, see Chapter 9.9.4 “Transmit PDO status (objects 2030–2033)”	PDO status	RW	N
2031		RECORD	Transmit PDO 2 status	PDO status	RW	N
2032		RECORD	Transmit PDO 3 status	PDO status	RW	N
2033		RECORD	Transmit PDO 4 status	PDO status	RW	N
2034-203F			reserved			
2040		RECORD	Receive PDO 1 status, see Chapter 9.9.5 “Receive PDO status (objects 2040–2041)”	PDO status	RW	N
2041		RECORD	Receive PDO 2 status	PDO status	RW	N
2042-204F			reserved			
2060			Dummy with no function, returns 0, see Chapter 9.9.1.2 “Object 1A00-1A03: transmit PDO mapping parameter”	Unsigned8	RO	N
2110			Transfer buffer mapping		RW	N
2110			Transfer buffer status		RW	N
2110			Transfer buffer		RW	N
MDT user memory (SLK addresses 0x00000000–0x0000FFFF), see Chapter 9.6.2 “MDT user memory (objects 2200–220E)”						
2200		ARRAY	DATA segment 0: Byte 0–199	Unsigned8	RW	N
2201		ARRAY	DATA segment 1: Byte 200–399	Unsigned8	RW	N
2202		ARRAY	DATA segment 2: Byte 400–599	Unsigned8	RW	N
2203		ARRAY	DATA segment 3: Byte 600–799	Unsigned8	RW	N
2204		ARRAY	DATA segment 4: Byte 800–999	Unsigned8	RW	N
2205		ARRAY	DATA segment 5: Byte 1000–1199	Unsigned8	RW	N
2206		ARRAY	DATA segment 6: Byte 1200–1399	Unsigned8	RW	N
2207		ARRAY	DATA segment 7: Byte 1400–1599	Unsigned8	RW	N
2208		ARRAY	DATA segment 8: Byte 1600–1799	Unsigned8	RW	N
2209		ARRAY	DATA segment 9: Byte 1800–1999	Unsigned8	RW	N
220A		ARRAY	DATA segment 10: Byte 2000–2199	Unsigned8	RW	N
220B		ARRAY	DATA segment 11: Byte 2200–2399	Unsigned8	RW	N
220C		ARRAY	DATA segment 12: Byte 2400–2599	Unsigned8	RW	N
220D		ARRAY	DATA segment 13: Byte 2600–2799	Unsigned8	RW	N
220E		ARRAY	DATA segment 14: Byte 2800–2999	Unsigned8	RW	N
220F		ARRAY	DATA segment 15: Byte 3000–3199	Unsigned8	RW	N
2210		ARRAY	DATA segment 16: Byte 3200–3399	Unsigned8	RW	N
2211		ARRAY	DATA segment 17: Byte 3400–3599	Unsigned8	RW	N
2212		ARRAY	DATA segment 18: Byte 3600–3799	Unsigned8	RW	N

Object index	Subindex	Object code ¹⁾	Name	Data type	Acc ²⁾	S ³⁾
2213		ARRAY	DATA segment 19: Byte 3800–3999	Unsigned8	RW	N
2214		ARRAY	DATA segment 20: Byte 4000–4199	Unsigned8	RW	N
2215		ARRAY	DATA segment 21: Byte 4200–4399	Unsigned8	RW	N
2216		ARRAY	DATA segment 22: Byte 4400–4599	Unsigned8	RW	N
2217		ARRAY	DATA segment 23: Byte 4600–4799	Unsigned8	RW	N
2218		ARRAY	DATA segment 24: Byte 4800–4999	Unsigned8	RW	N
2219		ARRAY	DATA segment 25: Byte 5000–5199	Unsigned8	RW	N
221A		ARRAY	DATA segment 26: Byte 5200–5399	Unsigned8	RW	N
221B		ARRAY	DATA segment 27: Byte 5400–5599	Unsigned8	RW	N
221C		ARRAY	DATA segment 28: Byte 5600–5799	Unsigned8	RW	N
221D		ARRAY	DATA segment 29: Byte 5800–5999	Unsigned8	RW	N
221E		ARRAY	DATA segment 30: Byte 6000–6199	Unsigned8	RW	N
221F		ARRAY	DATA segment 31: Byte 6200–6399	Unsigned8	RW	N
2220		ARRAY	DATA segment 32: Byte 6400–6599	Unsigned8	RW	N
2221		ARRAY	DATA segment 33: Byte 6600–6799	Unsigned8	RW	N
2222		ARRAY	DATA segment 34: Byte 6800–6999	Unsigned8	RW	N
2223		ARRAY	DATA segment 35: Byte 7000–7199	Unsigned8	RW	N
2224		ARRAY	DATA segment 36: Byte 7200–7399	Unsigned8	RW	N
2225		ARRAY	DATA segment 37: Byte 7400–7599	Unsigned8	RW	N
2226		ARRAY	DATA segment 38: Byte 7600–7799	Unsigned8	RW	N
2299		ARRAY	DATA segment 153: Byte 30600–30799	Unsigned8	RW	N
			Reserved area (SLK address 0x00010000 - 0x0001FFFF)			
2400-25FF			reserved			
			SLK register area (SLK address 0x00030000–0x0003FFFF) See Chapter 9.6.3 “SLK register area (object 2600)”			
2600		ARRAY	SREG segment 0: Byte 0–199	Unsigned8		N
	00		No. elements in array	Unsigned8	RW	N
	01		Actual link state	Unsigned8	RO	N
	02		Commanded link state	Unsigned8	RW	N
	03		Auto mode configuration	Unsigned8	RW	N
	04		SLK operative flag	Unsigned8	RW	N
	05		MDT counter - highest byte (3)	Unsigned8	RW	N
	06		MDT counter - byte (2)	Unsigned8	RW	N
	07		MDT counter - byte (1)	Unsigned8	RW	N
	08		MDT counter - lowest byte (0)	Unsigned8	RW	N
	09-1C		reserved	Unsigned8	RW	N
	1D		Look Ahead Control register MDT pointer 1, high byte	Unsigned8	RW	N
	1E		Look Ahead Control register MDT pointer 1, low byte	Unsigned8	RW	N
	1F		Look Ahead Control register MDT pointer 2, high byte	Unsigned8	RW	N
	20		Look Ahead Control register MDT pointer 2, low byte	Unsigned8	RW	N

Object index	Subindex	Object code ¹⁾	Name	Data type	Acc ²⁾	S ³⁾
	21		Look Ahead Control register MDT pointer 3, high byte	Unsigned8	RW	N
	22		Look Ahead Control register MDT pointer 3, low byte	Unsigned8	RW	N
2601-27FF			reserved			
			MDT register area (SLK address 0x00020000–0x0002FFFF) See Chapter 9.6.4 “MDT register area (object 2800)”			
2800		ARRAY	HREG segment 0: Byte 0–199	Unsigned8		N
	00		No. elements in array	Unsigned8	RW	N
	01		MDT status - high byte	Unsigned8	RW	N
	02		MDT status - low byte	Unsigned8	RW	N
	03		MDT pointer 1 - high byte	Unsigned8	RW	N
	04		MDT pointer 1 - low byte	Unsigned8	RW	N
	05		MDT pointer 2 - high byte	Unsigned8	RW	N
	06		MDT pointer 2 - low byte	Unsigned8	RW	N
	07		MDT pointer 3 - high byte	Unsigned8	RW	N
	08		MDT pointer 3 - low byte	Unsigned8	RW	N
	09		MDT ID code - highest byte (3)	Unsigned8	RO	N
	0A		MDT ID code - byte (2)	Unsigned8	RO	N
	0B		MDT ID code - byte (1)	Unsigned8	RO	N
	0C		MDT ID code - lowest byte (0)	Unsigned8	RO	N
	0D		MDT formatting, start address - high byte	Unsigned8	RW	N
	0E		MDT formatting, start address - low byte	Unsigned8	RW	N
	0F		MDT formatting, area length (byte count) - high byte	Unsigned8	RW	N
	10		MDT formatting, area length (byte count) - low byte	Unsigned8	RW	N
	11		MDT formatting, formatting value	Unsigned8	RW	N
	12		reserved	Unsigned8	RW	N
	13		MDT work pointer (pointer 0) - high byte	Unsigned8	RO	N
	14		MDT work pointer (pointer 0) - low byte	Unsigned8	RO	N
	15		MDT software version number	Unsigned8	RO	N
	16		reserved	Unsigned8		N
2801-2827			reserved			
			Standardized device profile area			
6000-9FFF			Not supported			
A000-FFFF			reserved			

1) Object type

2) Acc = access RW = read and write
RO = read only3) S - save Y = apply change
N = reload default value on restart

9.6.1 Saving objects (saving parameters)

Objects with “Y” in the “S” column are applied to the SLK. Parameters saved in this manner are loaded every time the SLK restarts.

The currently configured values of the parameters marked “N” are lost after voltage recovery. The default values are reloaded in the objects.



Any SDO response telegram may be delayed depending on how long the initiated write or read process on the nonvolatile memory takes. Ensure the master has sufficient timeout specifically for these kinds of access.

9.6.2 MDT user memory (objects 2200–220E)

For access via CANopen, the memory of the data carrier is divided into 200-byte segments.

When accessing a specific MDT byte, both the segment number and the resulting position of the desired data byte within the segment have to be determined. The segment number is represented here by the index, and the relative position of the data byte in the segment by the subindex.

Example: The data byte with the MDT address 0x183C is in segment 31 and is mapped in the object with index 221F/05.

If a data object outside the addressable MDT area is accessed, the SLK reports a corresponding error (see Chapter 9.10 “CANopen error codes”).

Table 32: Maximum possible addressing of MDT types

MDT type	MDT start address	Object index start address	MDT end address	Object index end address
ID 40/MDT2K	0x0000	2200/01	0x076F	2209/68
ID 40/MDT8K, ID 80/E-MDT8K	0x0000	2200/01	0x1DEF	2226/40
ID 40/MDT32K	0x0000	2200/01	0x784F	2299/C8

9.6.3 SLK register area (object 2600)

The ID 40 system is checked with this object as described in Chapter 5.3 “SLK register area”.

9.6.4 MDT register area (object 2800)

The MDT functions described in Chapter 5.2 “Map of MDT register area” are found here.

9.7 Standardized device profile

The ID 40 system cannot be assigned to any of the CANopen device profiles currently available. For this reason, the standardized device profile specified in [C17] is not supported by the ID 40/SLK-CAN.

9.8 SDO communication

The SLK supports two independent SDO channels that can transmit and receive. The first SDO channel is used to transmit data in process, the second SDO channel is intended for diagnostic purposes. The two SDO channels are distinguished by corresponding **COB IDs** (**C**ommunication **O**bject **I**dentifiers).

The SLK does not support client SDO channels, only server SDOs.

The following SDO protocols defined in [C17] are supported:

- SDO download
- SDO upload
- Block SDO download
- Block SDO upload
- Abort SDO transfer for selected protocol

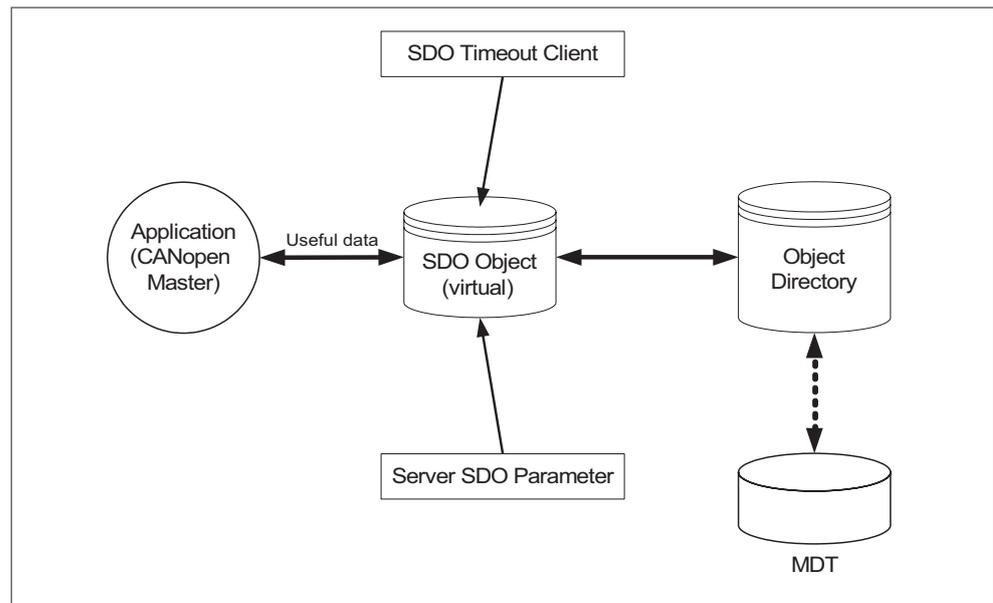


Fig. 29: SDO communication

SDO parameters are set in the objects **server SDO parameters** and **SDO timeout client** (see Chapter 9.8.3 “SDO timeout client [object 2020]”).

9.8.1 SDO 1

SDO 1 is the default SDO channel.

The COB IDs are fixed in [C17] in the server SDO parameters object (index 1200) and cannot be edited.

9.8.2 SDO 2

This SDO channel is reserved for online diagnostics with corresponding diagnostic tools. It remains inactive by default and only runs once configured (see Chapter 11.3.4 “Configuring the second SDO channel”). It functions in the same manner as SDO 1.

Both the CANopen master and the diagnostic tools can edit the COB IDs of SOD 2 using corresponding object access if necessary.

9.8.3 SDO timeout client (object 2020)

This object defines the maximum time the SLK has to wait for a follow-up telegram from the master (client). If this time is exceeded, the SLK terminates communication with **abort transfer** (see [C17] for abort code).

By default, the SLK waits one second. IF the bus master needs more time to process more data from the SDO upload, this parameter can be adjusted. The time counts down until the SLK receives packet confirmation.

Table 33: Object description

INDEX	2020
Name	SDO timeout client
Object code	ARRAY
Data type	Unsigned32
Category	Optional

Table 34: Description of entries

Subindex	00
Description	Number of implemented SDO channels
Entry category	Mandatory
Access	Read only
PDO mapping	No
Value range	2
Default value	2
Subindex	01
Description	Timeout for SDO channel 1 (client side)
Entry category	Mandatory
Access	Read write
PDO mapping	No
Value range	0 = disable timeout, > 0 = timeout value in ms
Default value	1,000 ms
Subindex	02
Description	Timeout for SDO channel 2 (client side)
Entry category	Mandatory
Access	Read write
PDO mapping	No
Value range	0 = disable timeout, > 0 = timeout value in ms
Default value	1,000 ms

9.9 PDO communication

PDO services support event-oriented data exchange with the SLK.

PDOs transmit the contents of **mapped** objects.

The SLK provides four PDOs for SLK → master (TPDO1 to TPDO4) and two PDOs for master → SLK (RPDO1 and RPDO2). Each PDO can transmit 8 bytes of payload.

The primary application is TPDOs for event-oriented transmission of MDT data and status messages. Note that all MDT objects in a TPDO together only activate one trigger condition. If, e.g., 8 MDT data bytes are assigned to a TPDO, they are first all read from the MDT and then entered together in the TPDO. Therefore only one TPDO is sent.

By using RPDOs, it is also possible to run the ID 40 system with CANopen masters that do not support command-oriented data exchange via SDOs.

However, be sure to observe the following restrictions:

1. The PDO default mappings in the SLK only provide access to the first six MDT data bytes, the MDT status register and the “actual link state” and “commanded link state” SLK register (see “PDO default mapping” in Chapter 9.9.1 “PDO mapping”).
2. The recipient will not respond to PDO telegrams. This makes a PDO an **unconfirmed** CANopen service. The SLK saves the successful execution of a data entry by an RPDO in the “error register” and “pre-defined error field” objects (see 9.17.4 “Pre-defined error field [object 1003]”). These objects can only be checked through additional access.

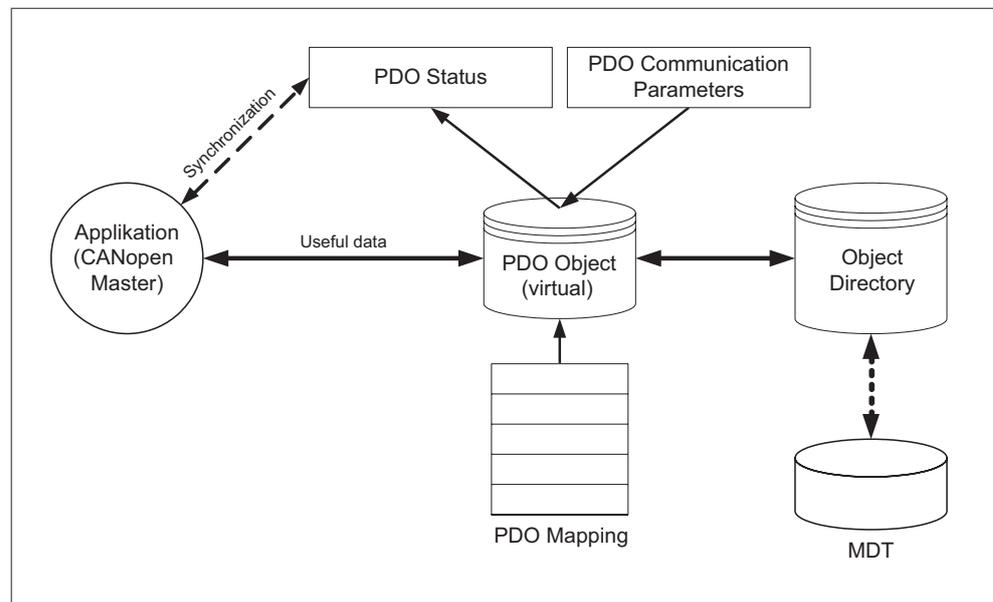


Fig. 30: PDO communication

9.9.1 PDO mapping

Eight bytes can be transmitted in each PDO. Which bytes are sent or received is determined by the **PDO mapping**.

PDO contents can be dynamically mapped both in OPERATIONAL and PRE-OPERATIONAL status.

The data bytes in RPDOs are then only applied to the mapped objects by the SLK when the value of the **RPDO status object** (see Chapter 9.9.5 “Receive PDO status [objects 2040–2041]”) is 0x00. Applying the data of an executed RPDO to the SLK can be prevented with the **PDO invalid** value of 0x80.

The **TPDO status object** (see Chapter 9.9.4 “Transmit PDO status [objects 2030–2033]”) indicates the **validity** of the transmitted contents.

Both the TPDO and RPDO mappings are generally independent of one another, however the following restrictions should be observed.

1. Not all available objects are permitted for PDO mapping. This property is set for each object in the EDS file (see Chapter 9.18 “Electronic data sheet [EDS]”).
2. If both objects from the SLK register area (e.g., actual link state) and objects from the MDT user data area are mapped in a TPDO, changing the link state executes the TPDO.

However, if no MDT is linked at this time, the MDT data cannot be read. This TPDO is marked as **invalid** in the PDO status.



It is recommended to not map MDT data or registers to a PDO with SLK registers.

3. In each RPDO, the data sent from the application is entered in every mapped object – even when only one of the values should be written.



An RPDO overwrites **all** mapped objects, so pay attention to sent values.

4. Objects not allowing write access have the RO (read only) attribute. They cannot be mapped to RPDOs.



Read-only objects **cannot** be mapped to RPDOs.

The values contained in the following tables (object indexes) are **default values** for the ID 40/SLK. They can be adapted to the needs of the application through command-oriented data exchange.

The default mappings are arranged symmetrically, i.e., objects that can be read and written (R/W objects) are mapped to the same subindexes in TPDO and RPDO.

9.9.1.1 Object 1600-1601: Receive PDO mapping parameters

These parameters are available in the 160x objects, whereby x + 1 stands for the number of each RPDO channel.

Table 35: RPDO mapping 1

Index	Subindex	Description	Value (index)
1600	00	No. mapped objects	8
	01	PDO valid	2040/01
	02	Commanded link state	2600/02
	03	Auto mode configuration	2600/03
	04	SLK operative flag	2600/04
	05	MDT counter - high byte (byte 3)	2600/05
	06	MDT counter - byte 2	2600/06
	07	MDT counter - byte 1	2600/07
	08	MDT counter - low byte (byte 0)	2600/08

Table 36: RPDO mapping 2

Index	Subindex	Description	Value (index)
1601	00	No. mapped objects	8
	01	PDO valid	2041/01
	02	MDT status	2800/01
	03	Byte at MDT address 0x0000	2200/01
	04	Byte at MDT address 0x0001	2200/02
	05	Byte at MDT address 0x0002	2200/03
	06	Byte at MDT address 0x0003	2200/04
	07	Byte at MDT address 0x0004	2200/05
	08	Byte at MDT address 0x0005	2200/06

9.9.1.2 Object 1A00-1A03: Transmit PDO mapping parameters

These parameters are available in the 1A0x objects.

Table 37: TPDO mapping 1

Index	Subindex	Description	Value (index)
1A00	00	No. mapped objects	8
	01	PDO valid	2030/01
	02	Actual link state	2600/01
	03	Auto mode configuration	2600/03
	04	SLK operative flag	2600/04
	05	MDT counter - high byte (byte 3)	2600/05
	06	MDT counter - byte 2	2600/06
	07	MDT counter - byte 1	2600/07
	08	MDT counter - low byte (byte 0)	2600/08

Table 38: TPDO mapping 2

Index	Subindex	Description	Value (index)
1A01	00	No. mapped objects	8
	01	PDO valid	2031/01
	02	MDT status	2800/01
	03	Data byte at MDT address 0x0000	2200/01
	04	Data byte at MDT address 0x0001	2200/02
	05	Data byte at MDT address 0x0002	2200/03
	06	Data byte at MDT address 0x0003	2200/04
	07	Data byte at MDT address 0x0004	2200/05
	08	Data byte at MDT address 0x0005	2200/06

Table 39: TPDO mapping 3

Index	Subindex	Description	Value (index)
1A02	00	No. mapped objects	8
	01	MDT ID code - high byte (3)	2032/01
	02	MDT ID code - byte (2)	2800/09
	03	MDT ID code - byte (1)	2800/0A
	04	MDT ID code - low byte (0)	2800/0B
	05	MDT software version number	2800/0C
	06	MDT work pointer - high byte	2800/13
	07	MDT work pointer - low byte	2800/14
	08	Dummy, returns 0	2060/00

Table 40: TPDO mapping 4

Index	Subindex	Description	Value (index)
1A03	00	No. mapped objects	0

9.9.2 PDO communication parameters

The following COB IDs are set by the factory as the default identifiers. These settings correspond to CANopen specifications and can be changed by the CANopen master by accessing the appropriate objects, if necessary.

Table 41: PDO communication parameters

Object	Index/ subindex	COB ID	Direction
RPDO1	1400/01	0x0200 + node ID	Master → ID 40/SLK
RPDO2	1401/01	0x0300 + node ID	Master → ID 40/SLK
TPDO1	1800/01	0x0180 + node ID	ID 40/SLK → master
TPDO2	1801/01	0x0280 + node ID	ID 40/SLK → master
TPDO3	1802/01	0x0380 + node ID	ID 40/SLK → master
TPDO4	1803/01	0x0480 + node ID	ID 40/SLK → master

9.9.3 PDO transmission types

The SLK supports the noncyclical/asynchronous and asynchronous transmission types defined in [C17]. This applies both for RPDO and TPDO. Remote transmission requests (RTR) are not supported.

Asynchronous PDO communication is recommended (transmission types 254, 255). This transmission type is suitable for very quick system reaction to an event.

9.9.4 Transmit PDO status (objects 2030 – 2033)

This object contains status information for each TPDO channel.

Table 42: Object description

INDEX	2030 - 2033
Name	Transmit PDO status for TPDO 1–4
Object code	RECORD
Data type	PDO status
Category	Optional

Table 43: Description of entries

Subindex	00
Description	Largest subindex supported
Entry category	Mandatory
Access	Read only
PDO mapping	Nein
Value range	1
Default value	See table
Subindex	01
Description	PDO valid
Entry category	Mandatory
Access	Read only
PDO mapping	Ja
Value range	0 = content of all PDO entries is valid 0x80 = content of all PDO entries is invalid
Default value	None

9.9.5 Receive PDO status (objects 2040 – 2041)

This object contains status information for each RPDO channel.

Table 44: Object description

INDEX	2040 - 2041
Name	Receive PDO status for RPDO1–4
Object code	RECORD
Data type	PDO status
Category	Optional

Table 45: Description of entries

Subindex	00
Description	Largest subindex supported
Entry category	Mandatory
Access	Read only
PDO mapping	Nein
Value range	1
Default value	See table
Subindex	01
Description	PDO valid
Entry category	Mandatory
Access	Read only
PDO mapping	Ja
Value range	0 = content of all PDO entries is valid 0x80 = content of all PDO entries is invalid
Default value	None

9.9.6 SYNC synchronization protocol

The SLK can process bus master SYNC telegrams – it is a consumer, but it cannot generate SYNC telegrams (producer).

The COB ID is set by the factory as the default identifier. These default settings correspond to CANopen specifications under [C17] and can be changed by the bus master (by accessing the appropriate objects), if necessary.

9.10 CANopen error codes

In error situations, the SLK signals the **cause** of the error, if possible.

The SLK responds to **command-oriented data access** to an invalid MDT address with an **abort code** in the SDO telegram. Abort codes are described in the Chapter “Abort SDO transfer protocol” in [C17].

If errors occur during **RPDO access** or when executing a **parameterized data exchange**, the SLK enters **emergency error codes** in the “pre-defined error field (object 1003)” described in Chapter 9.17.4.

Table 46: Error codes for ID 40-specific errors

Abort code	Emergency error code	Error
0x08000022	0x6300	Unexpected break in communication between SLK and MDT MDT device error, e.g., memory area defective
0x08000021	-	- Reserved -

Example: Emergency error code 0x6300 “device software data set” is set for an unexpected break in MDT communication while executing a prefetch.

9.11 Emergency object protocol (EMCY)

The SLK sends EMCYs (producer) when an emergency error code is entered in the internal error list (pre-defined error field, index 1003), see Chapter “CANopen error codes”. The error codes and EMCY are described in the Chapter “Emergency object” in [C17].

The SLK ignores EMCY objects send by the bus master, i.e., they are not entered in the pre-defined error field and do not trigger any additional EMCY.

The COB ID is set by the factory as the default identifier. These default settings correspond to [C17] specifications and can be changed by the bus master (by accessing the appropriate objects), if necessary.

9.12 NMT protocols

The SLK supports all **NMT** protocols set in [C17]. Both COB ID and data content are set and can therefore not be altered by the bus master.

9.13 Error control protocols

The SLK supports both possible **error control** protocols. As described in [C17], only one of the two protocols can be active at a time.

9.14 Node guarding protocol

For device monitoring, the SLK also supports the **node guarding protocol** for connecting to older bus masters. If possible, the more modern heartbeat protocol is preferable to the node guarding protocol.

9.15 Heartbeat protocol

Protocol for device monitoring, similar to the node guarding protocol. A significant advantage of the heartbeat protocol is that the bus master only has to define the SLK's behavior once. Afterward, the SLK reports its current device status automatically and cyclically as the **heartbeat producer**.

9.16 Boot-up protocol

The boot-up message indicates to the bus master that the SLK is in PRE-OPERATIONAL mode and now no longer releases heartbeat telegrams.

9.17 Communication profile area

The objects in the communication profile area have the index range of 1000–1FFF and are defined in [C17].

This chapter provides only an excerpt of the objects.

9.17.1 Device type (object 1000)

The content of the device type object is 0, see Chapter 9.7 “Standardized device profile”.

9.17.2 Error register (object 1001)

If the ID 40 system detects an error, bit 0 (generic error) is set. The other bits in the error register are not supported.

The error register can be mapped to a TPDO (see Chapter 9.9.1 “PDO mapping”). When an error occurs, the bus master is notified by event.

9.17.3 Manufacturer status register (object 1002)

The **bus states** for the SLK are shown in the MSR:

- INITIALIZING
- PRE-OPERATIONAL
- OPERATIONAL
- STOPPED

Bit 7 (generic error) always has the same value as bit 0 in the error register.

Table 47: Object description

INDEX	1002
Name	Manufacturer status register (MSR)
Object code	VAR
Data type	Unsigned32
Category	Optional

Table 48: Description of entries

Access	Read only
PDO mapping	Ja
Value range	Bits 0-2: 0 = INITIALIZING 1 = undefined 2 = undefined 3 = undefined 4 = undefined 5 = PRE-OPERATIONAL 6 = OPERATIONAL 7 = STOPPED Bits 3-6: not used (content 0) Bit 7: generic error Bits 8-31: not used (content 0)
Default value	0

9.17.4 Pre-defined error field (object 1003)

The pre-defined error field is designated in [C17] for internal error entries. The errors detected by the ID 40 system are entered in the pre-defined error field as emergency error codes (see Chapter 9.11 “Emergency object protocol [EMCY]” and Chapter 9.10 “CANopen error codes”).

Subindex 0 contains the number of entries. If this subindex is set to 0, the list is deleted.

9.18 Electronic data sheet (EDS)

The EDS file describes the objects in a CANopen device.

The EDS file can be imported to controllers with CANopen bus master, as well as to various CANopen configuration tools (e.g., node master, Vector configuration tool, etc.). This gives the user a convenient project planning solution.

9.19 Overview of ID 40 CANopen support

Function	Availability	Comments
CAN COB ID		
11-bit (standard)	yes	
29-bit (extended)	no	
Baud rates	10 kbps 20 kbps 50 kbps 100 kbps 125 kbps 250 kbps 500 kbps 800 kbps 1 Mbps	Configurable and readable via RS232. Factory setting = 500 kbps
Node address	1 – 127, other values not permitted	Configurable and readable via RS232. Factory setting = 63
DEFTYPE length support	no	Read access to the index of a standard data type returns the length of the data type in bytes
Subindex FF support	no	

Function	Availability	Comments
Permanently save parameters	yes	Selected values remain unchanged after voltage recovery
Triggering modes		
Event-driven	yes	
Timer-driven	no	
Remotely requested	no	
SDO (client)		
SDO (server)		
No. SDO channels	2	1x master, 1x diagnostics
Upload/download		The second SDO channel can be deactivated.
Expedited transfer	yes	
Segmented transfer	yes	
Block transfer	Yes (with CRC support)	
PDO		
No. PDO channels	4 TPDO, 2 RPDO	The indicated transmission modes apply both for TPDOs and RPDOs.
Transmission mode		
Noncyclical/synchronous	yes	
Cyclical/synchronous	no	
Synchronous/RTR	no	
Asynchronous/RTR	no	
Asynchronous	yes	
Dynamic mapping in		
PRE-OPERATIONAL	yes	
OPERATIONAL	yes	
Synchronization protocol (SYNC)		
No high resolution synchronization protocol required		
Support as		
Consumer	yes	
Producer	no	
Time protocol		
Support as		
Consumer	no	
Producer	no	
Emergency object protocol		
Support as		
Consumer	no	
Producer	yes	
NMT protocols		
Start remote node	yes	
Stop remote node	yes	
Enter PRE-OPERATIONAL	yes	
Reset node	yes	
Reset communication	yes	
Error control protocol		
Node guarding protocol	yes	
Heartbeat protocol		
Consumer	no	
Producer	yes	
Boot-up protocol		
	yes	

9.20 Sample applications

The following examples show some typical applications of the ID 40 system using **CANopen**.

9.20.1 “Manual workstation” application

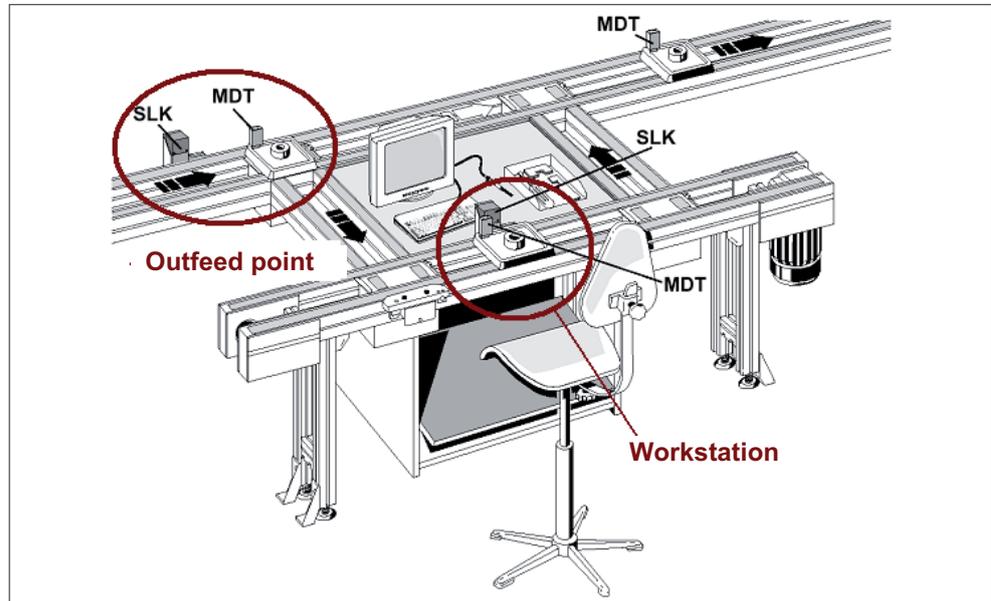


Fig. 31: Sample “manual workstation” application

The MDT should be read and written at a processing station. First, 32 bytes should be read from the MDT address 0x0110, then 60 bytes should be written from the MDT address 0x0100. The MDT status (see Chapter 4.2.3.1 “MDT status register”) is also transmitted in order to allow any errors in the MDT to be addressed.

This example is oriented on CANopen masters that do not support PDO communication. For this reason, the event-oriented data channel is not used. Even the actual link state is read by command using SDO access.

The application data is exchanged with the MDT directly in the *CONNECTED* state.

The MDT then signs off with the commanded link state *RECONNECT*. The SLK switches to the *CONNECTING* state to wait for the next MDT.

SLK preparation

The application data is read/written using the transfer buffers. The buffer mapping has to be configured once for each buffer. Afterward, the SLK switches to the link state *CONNECTING* with the command-oriented data exchange.

No.	Action	COB	rd/wr	Index/ subindex	Data
1	Lock both transfer buffers, 0 and 1, so buffer mapping can be configured	SDO	wr	2100/01	0x00
		SDO	wr	2101/01	0x00
2	Delete mappings for buffers 0 and 1	SDO	wr	2110/00	0x00000000
		SDO	wr	2111/00	0x00000000
3	Prepare first mapping for transfer buffer 0 to read MDT status	SDO	wr	2110/01	0x80000001
4	Prepare second mapping for transfer buffer 0 to read 32 bytes starting at MDT address 0x0110	SDO	wr	2110/02	0x00440020
5	Configure both mappings for buffer 0 by entering the number of mappings in subindex 0.	SDO	wr	2110/00	0x00000002
6	Prepare transfer buffer 1 mapping to write 60 bytes starting at MDT address 0x0100	SDO	wr	2111/02	0x0040003C
7	Configure mapping for buffer 1 by entering the number of mappings in subindex 0, in this case 1	SDO	wr	2111/00	0x00000001
8	Activate transfer buffer 1 for direct data exchange	SDO	wr	2101/01	0x01
9	Activate transfer buffer 0 as prefetch buffer. The mapped data blocks will be read automatically from the next MDT	SDO	wr	2100/01	0x03
10	Make SLK ready to receive (CONNECTING)	SDO	wr	2600/02	0x01

SLK operation

Check the actual link state until the value *CONNECTED* appears (2). At this point, the SLK has already read the MDT status and the user data starting at MDT address 0x0110 from the new MDT. The data is in transfer buffer 0 and can be retrieved using SDO.

Once this data has been evaluated and the workpiece processed, the result of the processing is written to the MDT. The data from the application is entered in transfer buffer 1 using SDO. Communication with the MDT is then terminated.

No.	Action	COB	rd/wr	Index/ subindex	Data
2	Read the actual link state until the value <i>CONNECTED</i> appears	SDO	rd	2600/01	
	Use the status register of transfer buffer 0 to check whether or not the data was retrieved from the MDT (see Chapter 9.5.1 “Transfer buffer parameters [objects 2100–2103]”)	SDO	rd		
	Read transfer buffer 0 once MDT data is copied	SDO	rd	2120/00	
	Enter user data in transfer buffer 1 so the bytes are immediately written to the MDT	SDO	wr	2121/00	<data bytes>
	Close MDT with RECONNECT. The next MDT can automatically sign on without another command	SDO	wr	2600/02	0x03
2	Read the actual link state until the value <i>CONNECTING</i> appears	SDO	rd	2600/01	

9.20.2 “Outfeed” application

Outfeed points are where it is decided whether or not the workpiece pallet is diverted. The workpiece pallets do not need to stop at the SLK before the junction, since the amount of data needed for the decision is small. In this example, 6 bytes should be read starting at MDT address 0x0000. The MDT status is also transmitted in order to allow any errors in the MDT to be addressed.

The data is read from the MDT using an unbuffered prefetch and sent to the CANopen master using TPDO. The default mapping for TPDO1 and TPDO2 can be used (see Chapter 9.9.1.2 “Object 1A00–1A03: transmit PDO mapping parameters”). The link then switches immediately from the *PRECONNECTED* to the *CONNECTING* state, since auto reconnect is active.

SLK preparation

Auto reconnect mode is activated (see Chapter 5.3.3 “Auto mode”) and the SLK switches to the *CONNECTING* link state.

No.	Action	COB	rd/wr	Index/ subindex	Data
1	Switch auto mode register to auto reconnect	SDO	wr	2600/03	0x01
10	Make SLK ready to receive (CONNECTING)	SDO	wr	2600/02	0x01

SLK operation

The CANopen master of the PLC receives two PDOs for every MDT that passes the outfeed SLK: TPDO1 and TPDO2. The MDT status and the six payload bytes are sent in TPDO2 according to the mapping. The outfeed decision can now be made with this information.

9.20.3 “Workstation” application

This example is for a workstation with a lift positioning unit. Since a link cannot be established between the SLK and the MDT when the workpiece pallet is raised, data is exchanged in two stages.

Stage 1: Reading the workpiece-specific data to check whether or not this workpiece can be processed at this station. This data is read from the MDT in an unbuffered prefetch. The SLK then automatically switches to *DISCONNECTED*. Only when the lift positioning device lowers after processing does the SLK switch to *CONNECTING*. This eliminates the risk of an unexpected break in communication as the MDT approaches the SLK from above.

Stage 2: Entering measurement data obtained while the workpiece was being processed. Only the pretransmit function is used via transfer buffer 1 here.

SLK preparation

A buffered prefetch is configured to transmit the control data (e.g., 16 bytes starting at MDT address 0x0024) in transfer buffer 0. Transfer buffer 1 is prepared for entering the measurement data (e.g., 800 bytes starting at MDT address 0x03C0). The SLK switches to auto disconnect mode and the link state switches to *CONNECTING*.

The actual link state and the MDT status are transmitted by event with TPDO1 and TPDO2. The status information for transfer buffers 0 and 1 are mapped to TPDO4.

No.	Action	COB	rd/wr	Index/ subindex	Data
1	Lock both transfer buffers, 0 and 1, so buffer mapping can be configured	SDO	wr	2100/01	0x00
		SDO	wr	2101/01	0x00
2	Delete mappings for buffers 0 and 1	SDO	wr	2110/00	0x00000000
		SDO	wr	2111/00	0x00000000
4	Prepare second mapping for transfer buffer 0 to read 16 bytes starting at MDT address 0x0024	SDO	wr	2110/02	0x00090010
5	Configure mapping for buffer 0 by entering the number of mappings in subindex 0, in this case 1	SDO	wr	2110/00	0x00000001
6	Prepare transfer buffer 1 mapping to write 800 bytes starting at MDT address 0x03C0	SDO	wr	2111/02	0x00F00320
7	Configure mapping for buffer 1 by entering the number of mappings in subindex 0, in this case 1	SDO	wr	2111/00	0x00000001
8	Activate transfer buffer 0 as prefetch buffer. The mapped data blocks will be read automatically from the next MDT	SDO	wr	2100/01	0x03
9	Transfer buffer 1 is only activated in Stage 2				
10	Enter the status registers of transfer buffers 0 and 1 in the mapping for TPDO4, set the number of mapped objects to 2	SDO	wr	1A03/01	0x21000208
		SDO	wr	1A03/02	0x21010208
		SDO	wr	1A03/00	0x02
11	Switch auto mode register to auto disconnect	SDO	wr	2600/03	0x02
12	Make SLK ready to receive (<i>CONNECTING</i>)	SDO	wr	2600/02	0x01

SLK operation

Stage 1: Once an MDT enters the HF field, the MDT status (triggers TPDO2) and the data in transfer buffer 0 mapping are read. If the data in the transfer buffer is available, the SLK sends a TPDO4 with the appropriate status information.

Stage 2: The lift unit moves the workpiece pallet into processing position. The measurement data obtained in the application is entered in transfer buffer 1, the buffer is switched to pretransmit mode and auto mode is set to reconnect. Once the lift unit returns the MDT to the position in front of the SLK, the system switches to *CONNECTING*. The data in transfer buffer 1 is automatically entered after the MDT signs on.

No.	Action	COB	rd/wr	Index/ subindex	Data
1	Wait until transfer buffer 0 status is "Data exchange complete".	TPDO4			
2	MDT communication is terminated by auto disconnect and the HF field is off. The lift unit can move the workpiece pallet into processing position				
3	Transfer buffer 0 can be read at the same time	SDO	rd	2120/00	
4	Once the measurement data from processing is available, enter it into transfer buffer 1.	SDO	wr	2121/00	<data bytes>
5	Switch transfer buffer 1 to "single transmit" pretransmit mode and activate	SDO	wr	2101/01	0x05
6	Switch auto mode register to auto reconnect	SDO	wr	2600/03	0x01
7	Wait until lift unit is in lower position				
8	Make SLK ready to receive (<i>CONNECTING</i>)	SDO	wr	2600/02	0x01
9	Wait until transfer buffer 1 status is "Data exchange complete".	TPDO4			
10	Switch the auto mode register to auto disconnect and allow the workpiece pallet to continue	SDO	wr	2600/03	0x02

9.20.4 “Resetting SLK” application

This example allows the SLK to be reset in the *DISCONNECTED* link state. The actual link state here is not important.

No.	Action	COB	rd/wr	Index/ subindex	Data
1	Switch SLK to ERROR state	SDO	wr	2600/02	0x04
2	Wait until actual link state is ERROR				
3	Switch SLK to DISCONNECTED state	SDO	wr	2600/02	0x02
4	Wait until actual link state is <i>DISCONNECTED</i>				

9.20.5 Handling “E00” errors

This example terminates direct data exchange with the MDT. The commanded link state *DISCONNECT* fails and the SLK switches to the *ERROR* state.

No.	Action	COB	rd/wr	Index/ subindex	Data
1	Switch SLK to <i>DISCONNECTED</i> state	SDO	wr	2600/02	0x02
2	Wait until actual link state is <i>DISCONNECTED</i> or <i>ERROR</i>				
3	<i>DISCONNECT</i> unsuccessful, link state is now <i>ERROR</i>				
4	Switch SLK to <i>DISCONNECTED</i> state	SDO	wr	2600/02	0x02
5	Wait until actual link state is <i>DISCONNECTED</i>				
6	The SLK is now in the initial <i>DISCONNECTED</i> state and the application can start over				

10 Web interface

10.1 Overview

The ID 40/SLK supports data exchange using the web browser of a connected computer. The homepage of the ID 40/SLK provides both interactive access to all **MDT** and **SLK** data (see Chapter 5 “SLK storage”) and support for system configuration (see Chapter 11 “Start-up and parameterization”) and system diagnostics (see Chapter 12 “Diagnostics”).

Data can be accessed using the SLK web interface at the same time fieldbus communication is active. This allows diagnostics to be run on the MDT data easily and efficiently while the application is running.



You can use the web interface to modify both MDT data and the link interface while the fieldbus application is running. This can even result in the product being damaged, depending on the application. This is why caution must be used when making these kinds of modifications during production.

All fieldbus variants of the ID 40/SLK with software version 4.0 or higher support the web interface.

The SLK is connected to the computer using the diagnostics cable (see Chapter 3.4.3 “Serial interface”).

The web interface replaces the ID 40/KONF diagnostics and configuration program. This program can still be used, since the SLK command shell is still supported.

10.2 Requirements

Computer (PC, notebook or similar) with the following requirements:

- RS232 port
 - COM1 (for example)
 - RS232 USB adapter if computer has no COM port
- Microsoft operating system
 - Windows 2000
 - Windows XP Professional
 - Windows 7 Professional
 - Other operating system that supports a PPP connection with adjustable dial-up chat via the RS232 serial interface
- Web browser:
 - Microsoft Internet Explorer version 6 or higher
 - Firefox version 1.5 or higher
 - Apple Safari version 5 or higher
 - Google Chrome version 25 or higher
 Other browsers may not display the websites correctly.
- Account with **administrator rights** for installing SLK access. An account with user rights is sufficient for later use.

The ID 40/SLK interacts with the computer as the **host** of a **network connection between two PCs with the serial communication cable**. This connection is used to run the standardized PPP (point-to-point protocol) known from modem connections, which supports the TCP/IP network protocol. This is basically a network connection to the ID 40/SLK.

The requirements for this kind of connection are met by the aforementioned Windows versions, so no additional driver software is required.

The following chapters describe how to set up and configure the network connection to the ID 40/SLK. This only has to be done once on the computer, similar to installing a modem.

10.3 Setting up the network connection to the ID 40/SLK

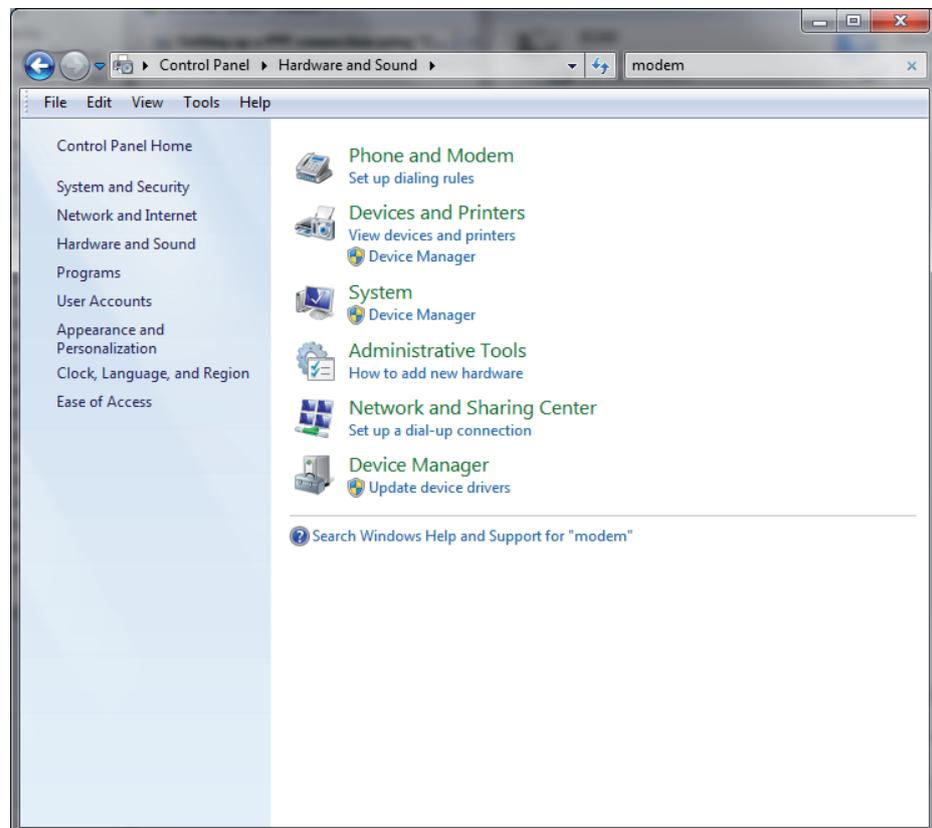
To set up the connection to the SLK, first use the Phone and Modem options, then the Network and Sharing Center in the Windows 7 Control Panel.

If you are using Windows XP, use the New Connection Wizard. If you are using Windows 2000, use the Network Connection Wizard. Both tools have similar functions and only differ somewhat in appearance.

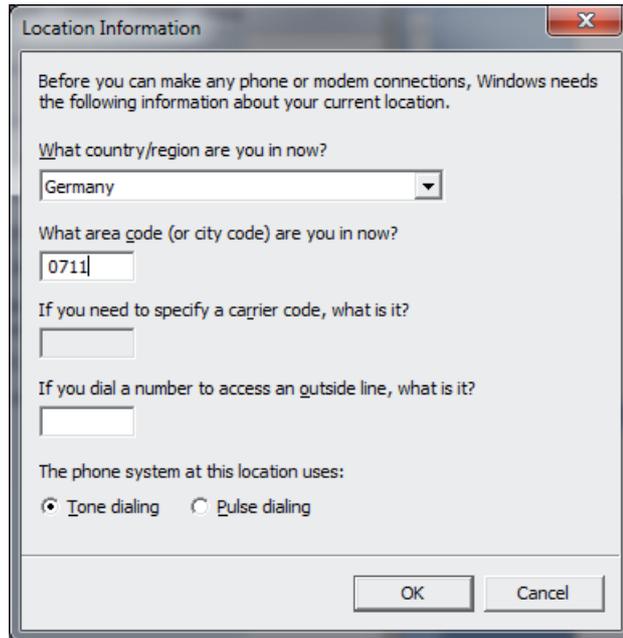
The following instructions describe setting up a computer running **Windows 7**. The installation information is only available in English.

10.3.1 Setting up the connection via serial cable

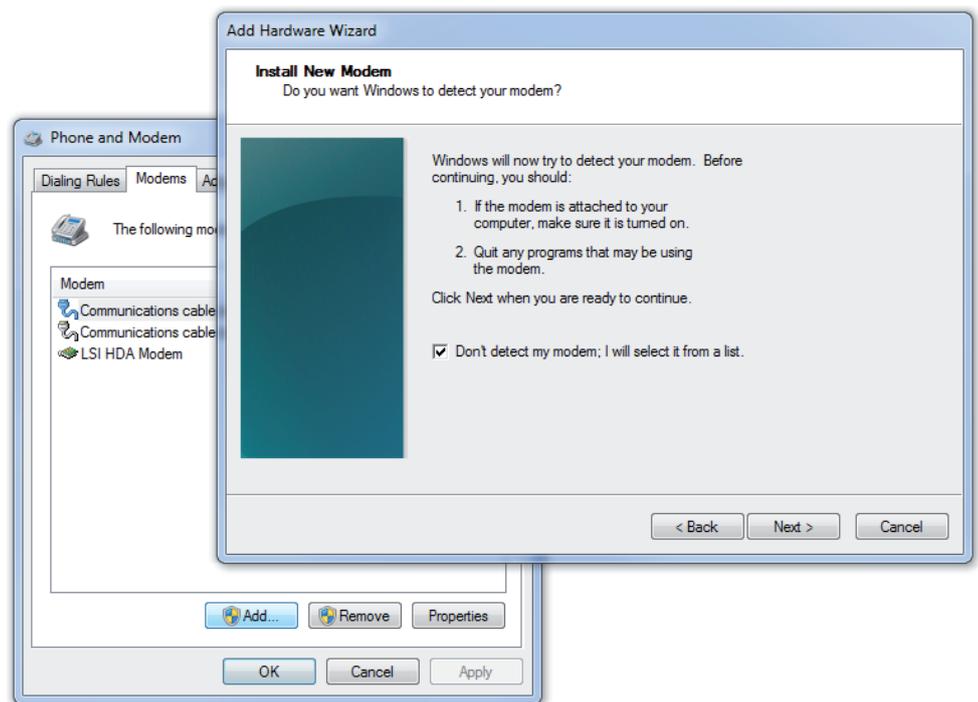
In Windows 7 it is used to install a **Modem** that uses a serial COM port, and not the real modem possibly built in your computer. To install it, open the Control Panel and select the **Phone and Modem** entry



- ▶ If requested by Windows 7, enter any **Location Information** – its only placeholder

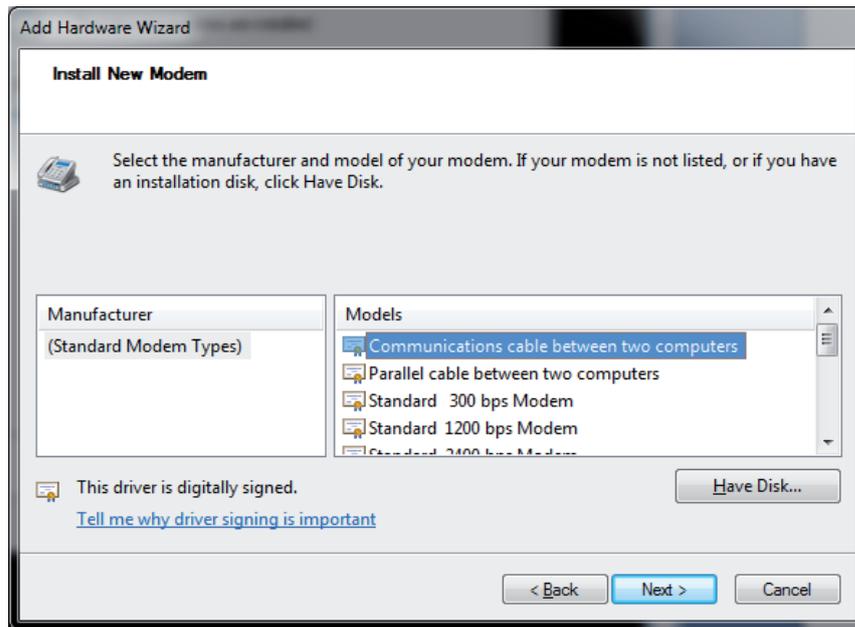


- ▶ **Add** a modem, but don't **detect** it, **select** it from a list

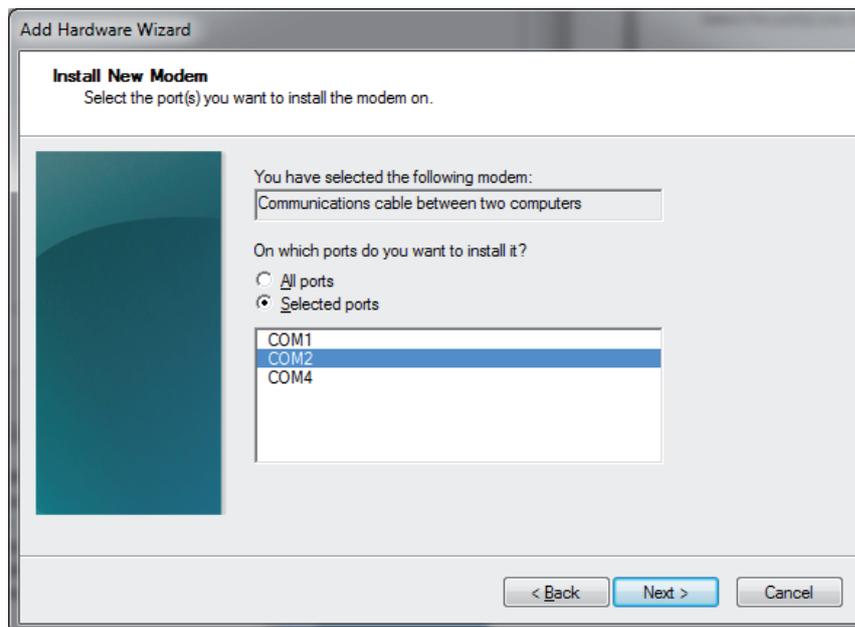


ENGLISH

- ▶ Install a **Communications cable between two computers**

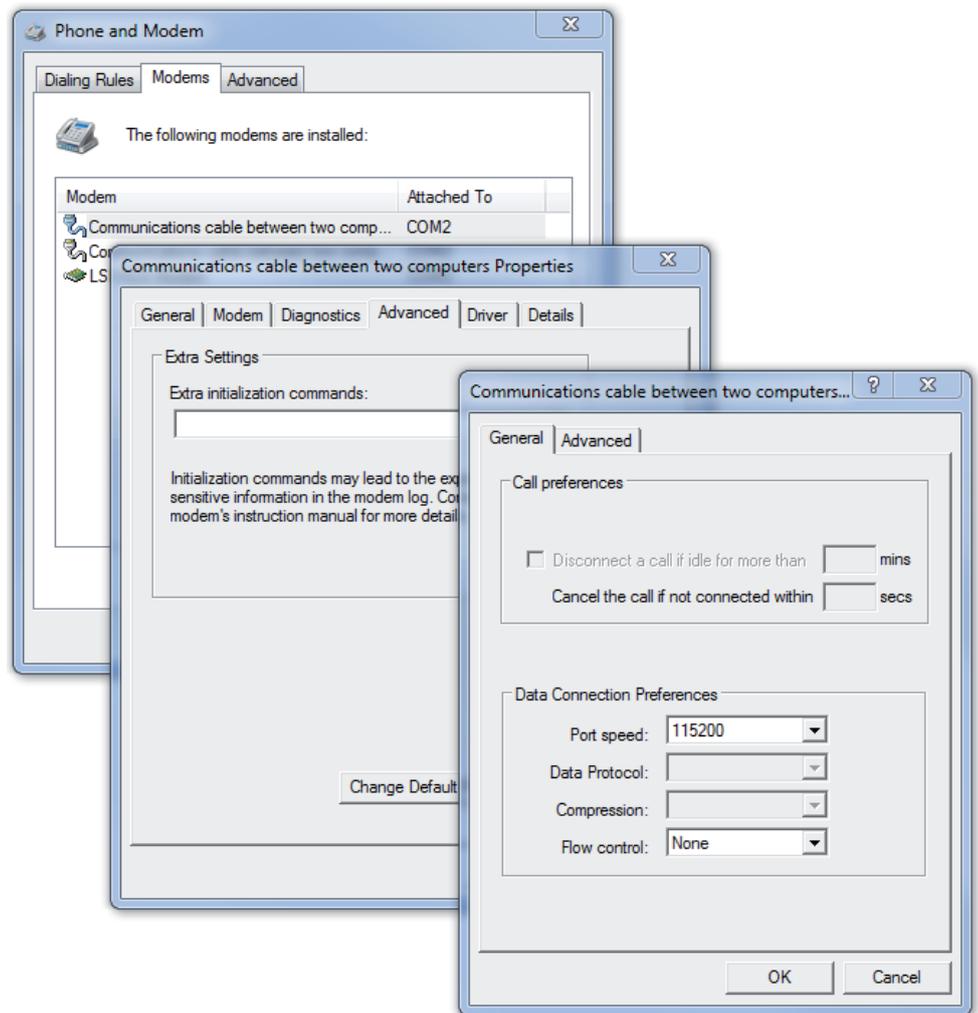


- ▶ Select one of the **COM ports** displayed; the ID 40/SLK must have connected to this COM port.



- ▶ Set the default baud rate of the COM port to the ID 40/SLK baud rate, otherwise the baud rate set in the network adapter properties was not saved.

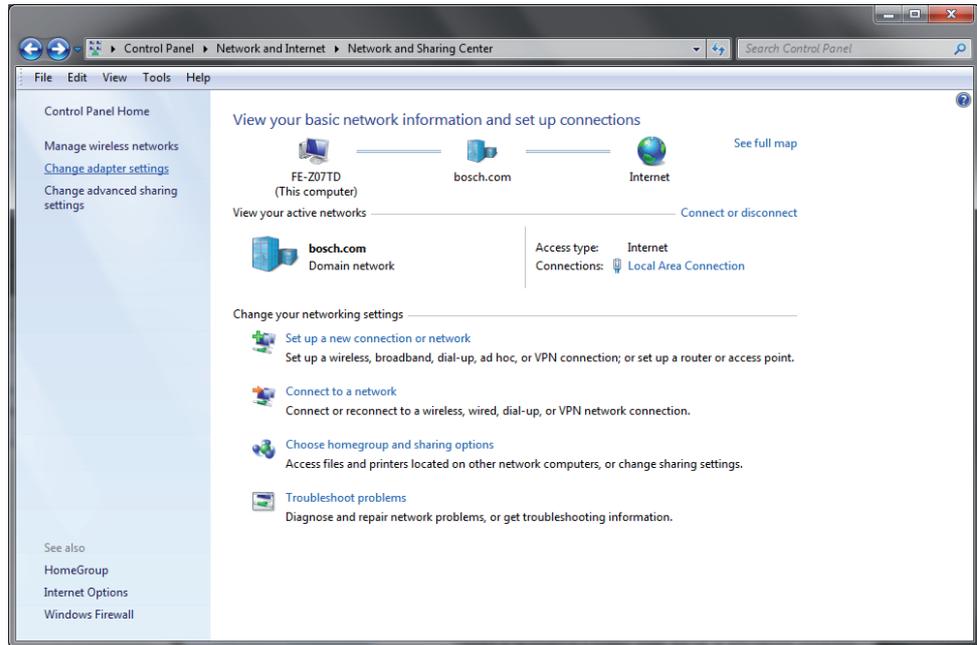
- ▶ Click **Change settings** in the **Properties** of the selected Communications Cable.



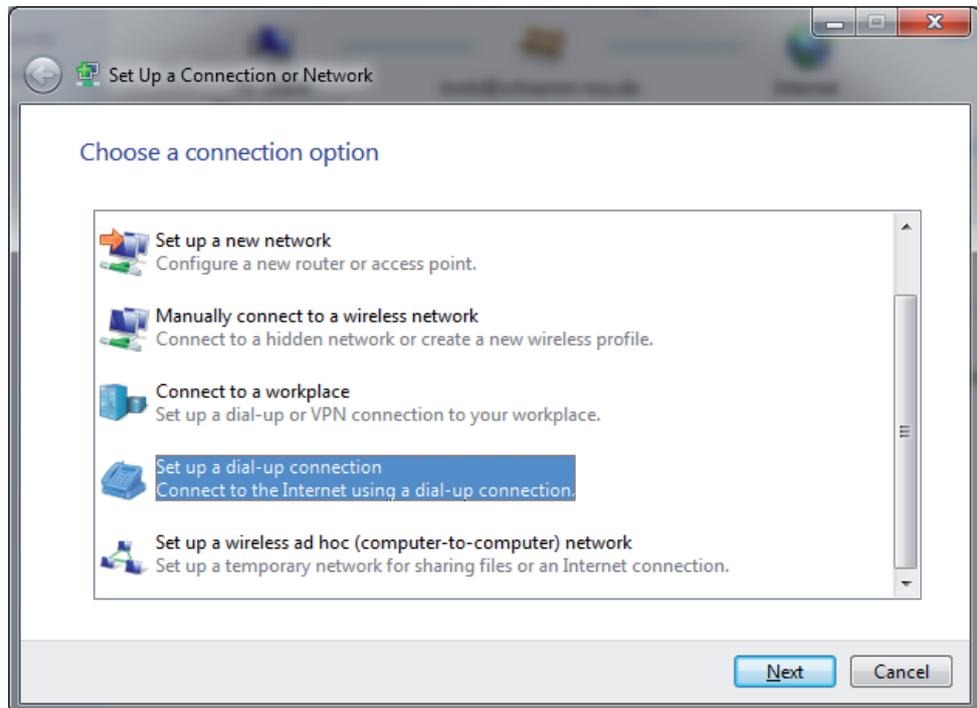
- ▶ Change **Default Preferences** in the **Advanced** tab and set **Port speed** to **115200** and **Flow control** to **None**
- ▶ Confirm all the dialogs with **OK**
- ▶ **Reboot** the PC to keep these settings.

10.3.2 Setting up a network connection

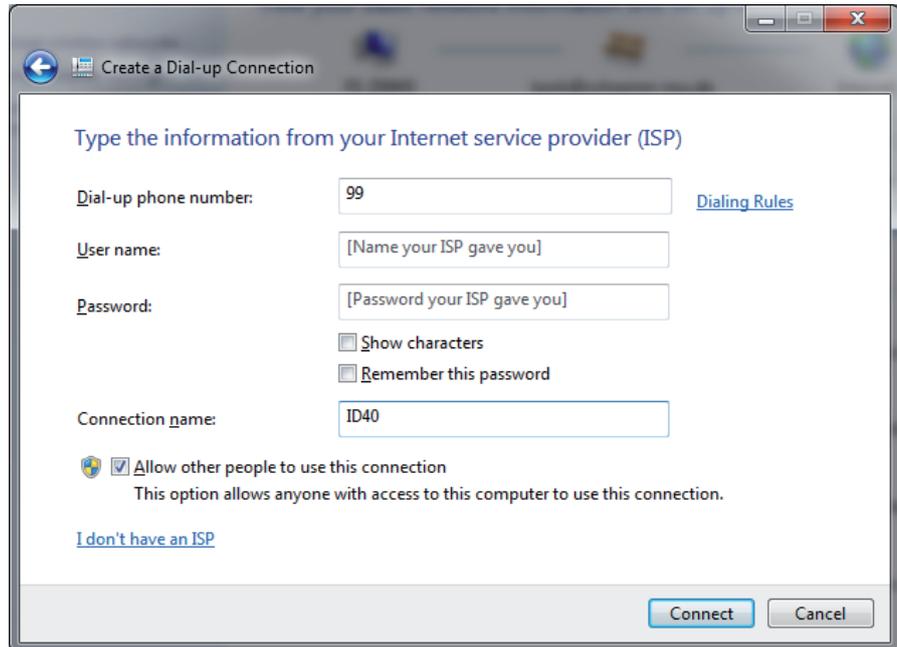
► **Set up a new connection or network** in the **Network and Sharing Center**



► And choose **Set up a dial-up connection**



- ▶ Then select the **Communications cable modem**, enter a **phone number** (only a placeholder) and enter a **Connection name**, e.g. **ID40**



Create a Dial-up Connection

Type the information from your Internet service provider (ISP)

Dial-up phone number: 99 [Dialing Rules](#)

User name: [Name your ISP gave you]

Password: [Password your ISP gave you]

Show characters

Remember this password

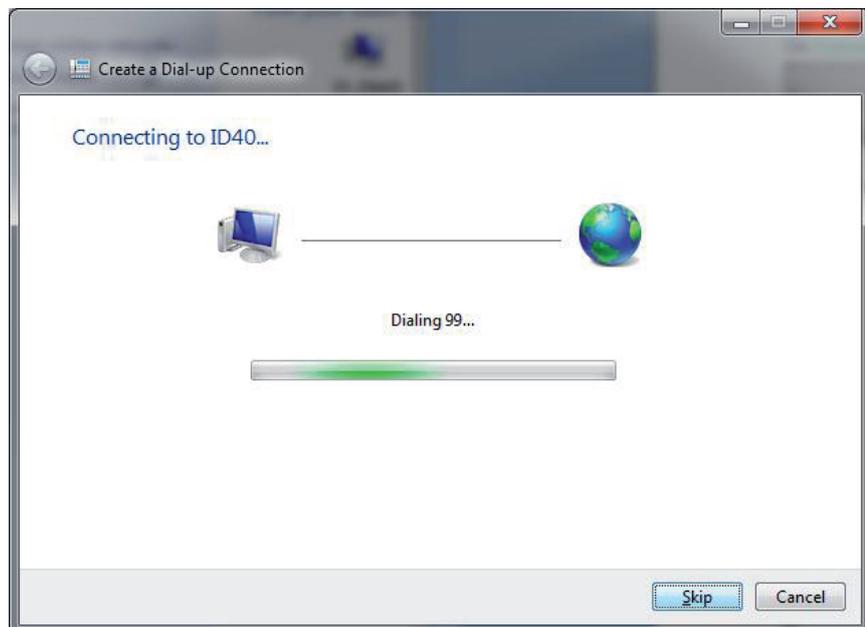
Connection name: ID40

Allow other people to use this connection
This option allows anyone with access to this computer to use this connection.

[I don't have an ISP](#)

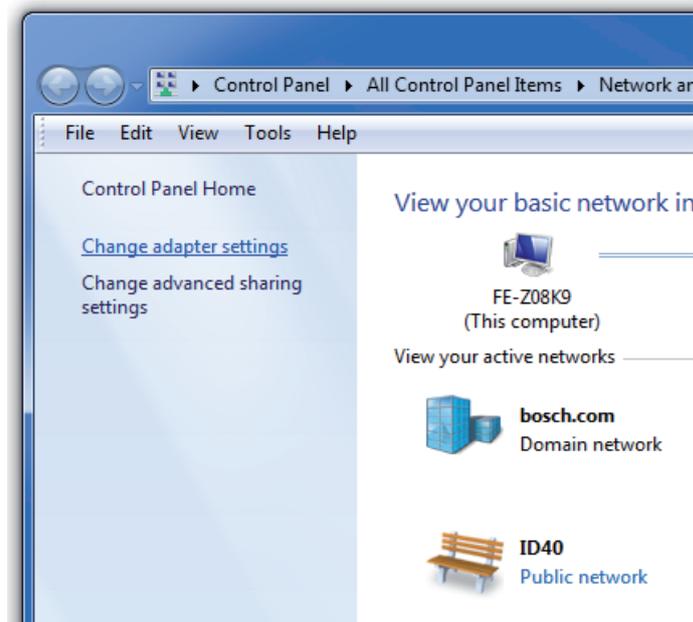
Connect Cancel

- ▶ On **Connect**, it tries to dial the phone number, so **Skip** this - it's not supported by the ID 40/SLK

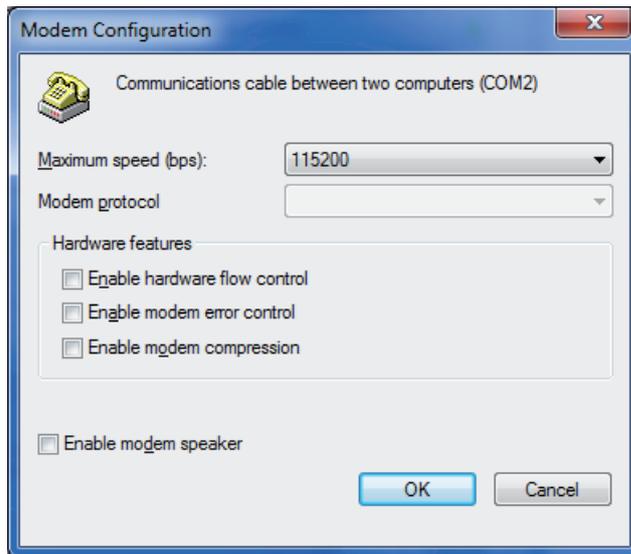


- ▶ Close the Dial-up dialog.

- ▶ In the **Network and Sharing Center** select **Change adapter settings**

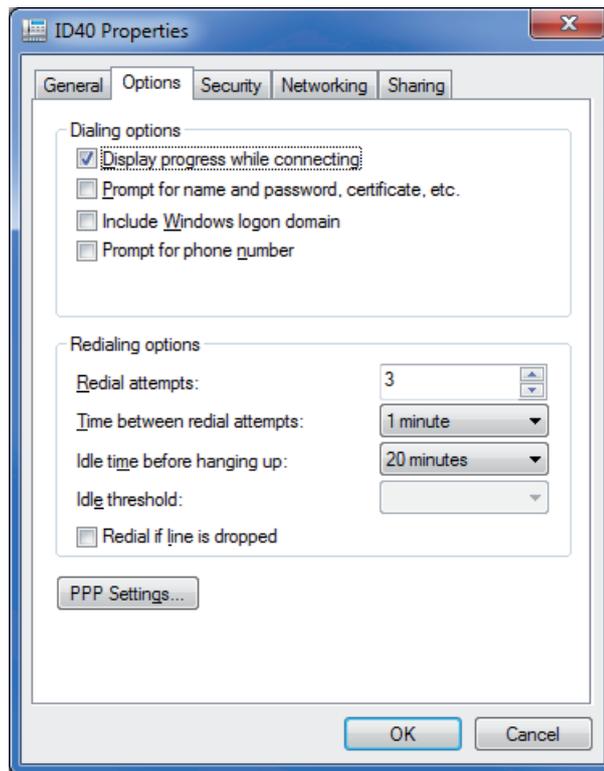


- ▶ Right-click the adapter entry **ID 40**, choose **Properties** and **Configure the Communications cable...**

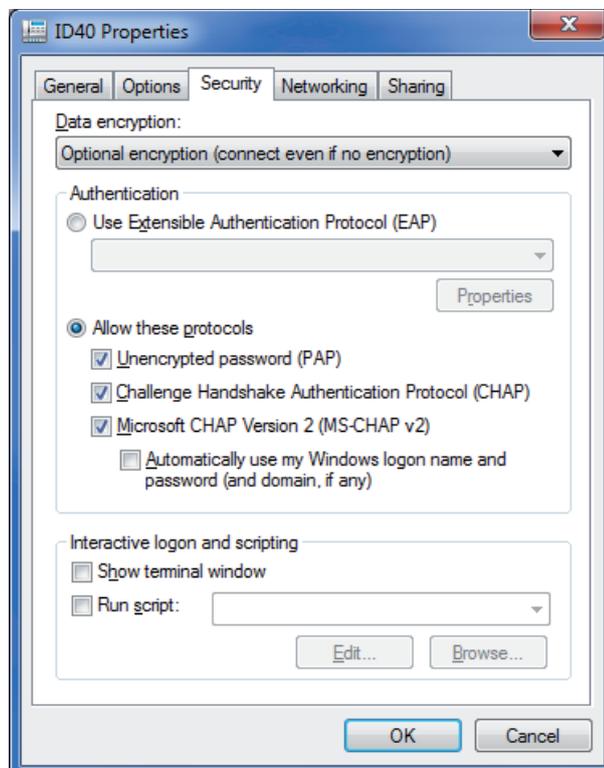


- ▶ Uncheck / disable the **Enable hardware flow control**, verify the **Maximum speed (bps)** is set to **115200**
- ▶ Back to ID 40 Properties, remove the **Phone number**

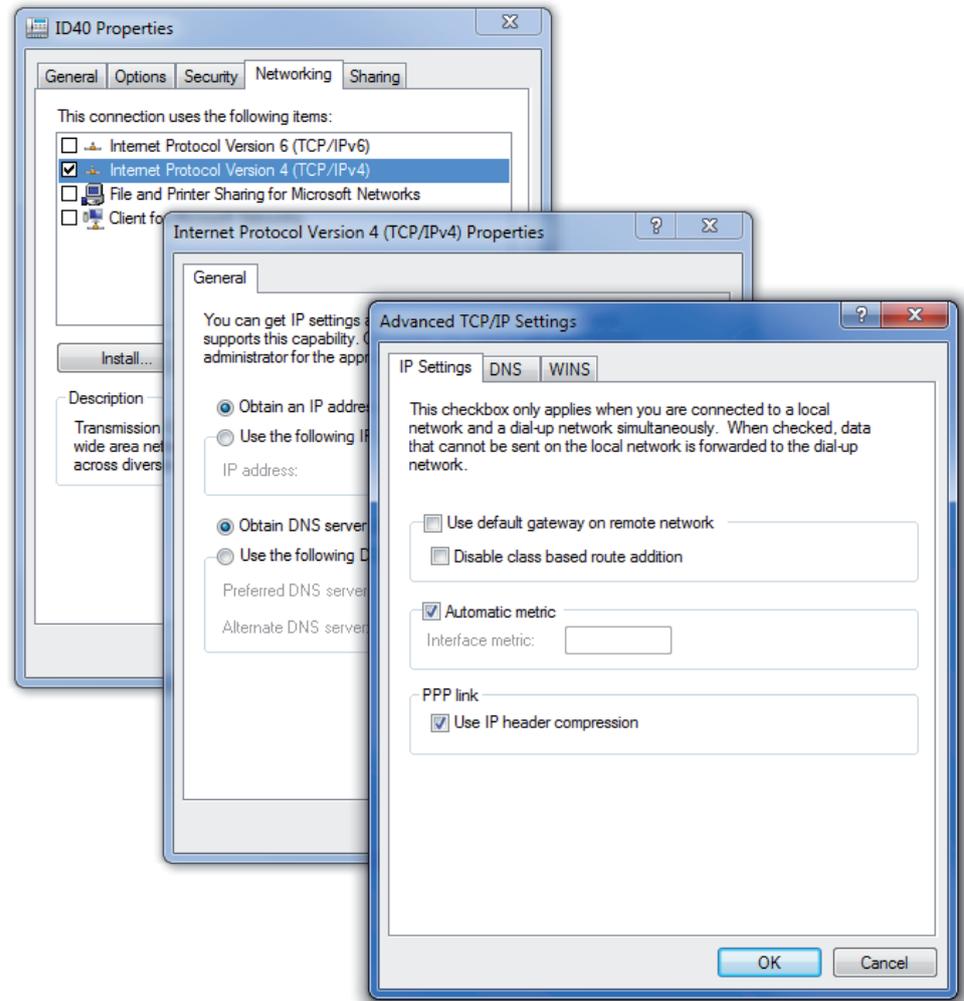
- In the Options tab, uncheck both **Prompt for phone number** and **Prompt for name and password...**



- Verify the Security tab, keep the **Optional encryption...**



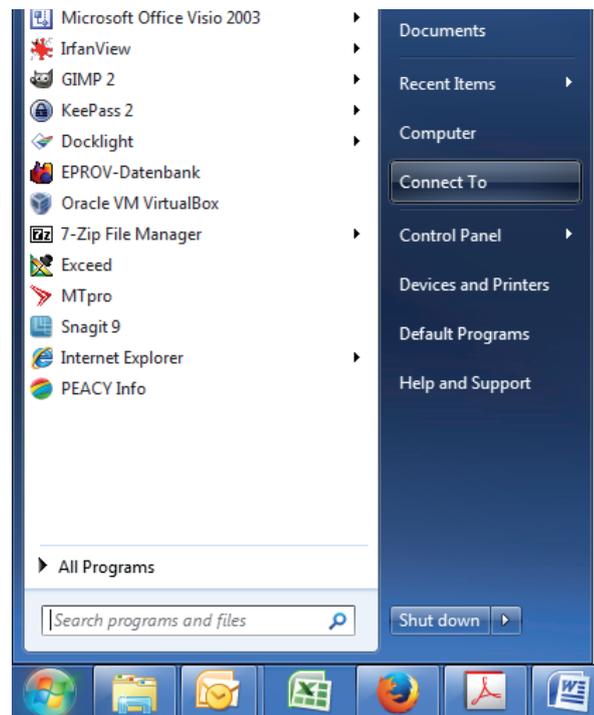
- ▶ In the Networking tab, check only the Internet Protocol Version 4 (TCP/IPv4)
- ▶ In the Internet Protocol Version 4 (TCP/IPv4) Properties, choose Advanced



- ▶ Uncheck **Use default gateway on remote network**, so the IP traffic with other destinations than ID 40 will still go on.
- ▶ Confirm the dialogs.

10.3.3 Establishing “ID 40” network connection

The start menu item **Connect To ...**

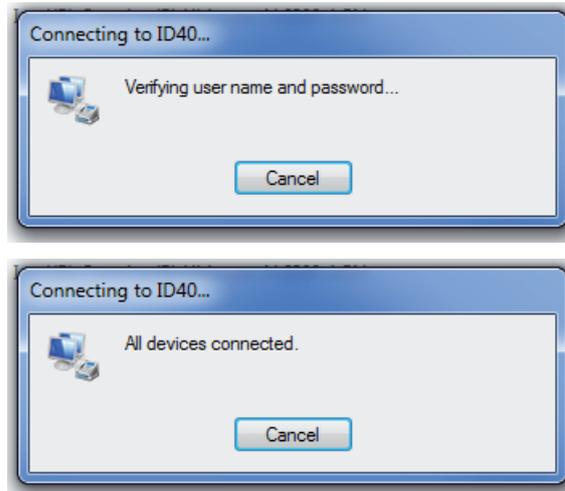


... opens a connection dialog box ...



... in which the ID 40 network connection can be started by selecting **Connect**.

After a few rapidly changing status messages, the connection is established.



The notification area of the Windows task bar shows the active network connections when the mouse pointer is moved over the "Network" icon.

"ID 40" should appear here once the connection is established.



10.3.4 Terminating “ID 40” network connection

If desired, you can terminate the connection by selecting **Connect To** from the Start menu and opening the connection dialog box, then clicking on **Disconnect** in the ID 40 connection.



The connection can be re-established at any time.



Terminate the network connection before removing the ID 40 diagnostics cable.

10.3.5 Troubleshooting PPP connection to ID 40/SLK

10.3.5.1 Error message appears when establishing connection

Possible causes of failed connection attempt:

- The ID 40/SLK is not connected - or connected to a COM port without a communication cable
- The assigned COM port is already being used by another program or connection
- The baud rate is not configured properly
- Hardware flow control is active

10.3.5.2 Behavior during fatal system errors

In exceptional instances, the SLK can experience a fatal system error. This is indicated by all display segments flashing and all functions ceasing, see Chapter 12.1.2 “Fatal system errors”.

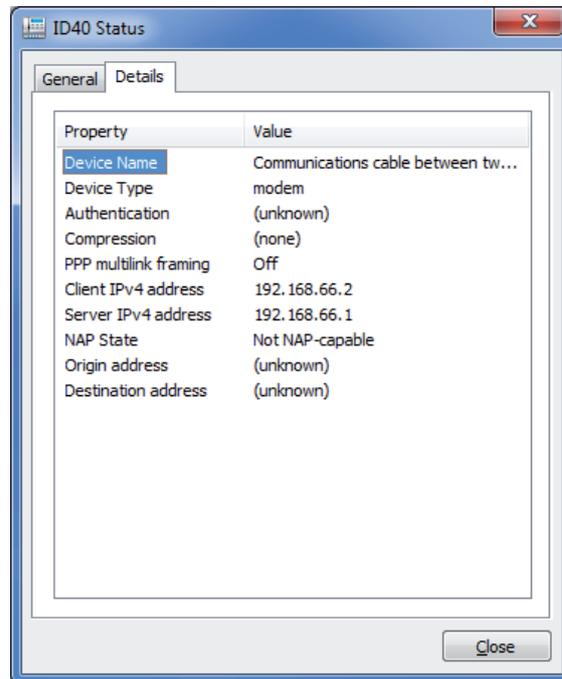
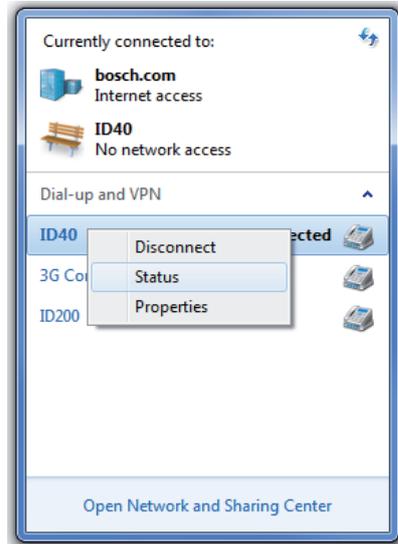
In addition, the SLK serial interface switches from PPP mode to emergency shell mode.

In this instance, access through a web browser is no longer possible. If this occurs, terminate the network connection (see Chapter 10.3.4 “Terminating ‘ID 40’ network connection”) and start the ID 40/KONF program to diagnose the fatal system error; see Chapter 12.1.2 “Fatal system errors” on how to proceed.

10.3.5.3 Changing the ID 40/SLK network address

In addition to the SLK, the network connection entry on the computer also has an IP address. Both IP addresses are specified by the SLK by default and are **192.168.66.1** (server = ID 40/SLK) and **192.168.66.2** (client = Windows 7 PC).

You can check these IP addresses in the Network Information window by opening the context menu (right click) for the ID 40 connection.



The **Details** tab in ID 40 Status contains the **Server IPv4 address**, which refers to the ID 40/SLK, and the **Client IPv4 address**, which refers to the computer. If your computer already has network connections in the 192.168.66.x address range, you have to use other IP addresses for connecting to the SLK to avoid conflicts. You can configure other IP addresses both on the computer in the TCP/IP network properties of the “ID 40” network connection and on the SLK using the ID 40/KONF program.



Basic knowledge of network technology and familiarity with the network environment of your computer are required to modify the network configuration. Consult your network administrator before changing the settings.

The network connection to the SLK has to be terminated before the IP addresses specified in the SLK can be changed.

Enter the command `pppoption` when prompted in the terminal window of the ID 40/KONF program, followed by a space and the IP address of the SLK, then a colon followed by the IP address of the computer.

Example:

```
ID 40/SLK > pppoption 10.66.82.8:10.66.32.12
```

Once the network connection is established, check the IP addresses as described above: the SLK (server) is now at 10.66.82.8, and the computer (client) is now at 10.66.32.12.

Enter an IP address instead of **Automatically find IP address** in the TCP/IP network properties of the “ID 40” network connection to replace the SLK default for the computer (10.66.32.12 in the above example) with the one entered.

10.3.5.4 Multiple SLKs on one computer

Every SLK being connected requires a COM port on the computer. Add a new direct network connection for every SLK as described above. A different network should be configured for each SLK to avoid network address conflicts on your computer. See also Chapter 10.3.5.3 “Changing the ID 40/SLK network address”.

Table 49: Example of connecting four SLKs to one computer

SLK no.	SLK IP address	IP address of network connection “directly” on computer	Entry in ID 40/KONF:
1	192.168.66.1	192.168.66.2	ID 40/SLK > pppoption... Default setting, no change needed
2	192.168.67.1	192.168.67.2	...192.168.67.1:192.168.67.2
3	192.168.68.1	192.168.68.2	...192.168.68.1:192.168.68.2
4	192.168.69.1	192.168.69.2	...192.168.69.1:192.168.69.2

10.4 HTTP connection to ID 40/SLK

Once the “ID 40” connection is set up and established on the computer (see Chapter 10.3 “Setting up the network connection to the ID 40/SLK”), the SLK can be operated using the web browser on the computer or automated using application programs (see Chapter 10.7 “Web access from application programs”).

As with a website, enter the web address (URL) of the SLK in the address bar of your browser, press **ENTER**, and after a brief loading period, the homepage of the ID 40/SLK will appear.

The network address or IP address is part of the web address. The default IP address of the SLK is **192.168.66.1**. Chapter 10.3.5.3 “Changing the ID 40/SLK network address” describes how this address can be changed if necessary.

The web address should be as follows:

http://192.168.66.1/index.html

The ID 40/SLK homepage loads within around 10 seconds.

If the page does not appear, refer to Chapter 10.4.1 “Troubleshooting the ID 40/SLK network connection” below.

The ID 40 website is described in Chapter 10.5 “The ID 40/SLK website”.

10.4.1 Troubleshooting the ID 40/SLK network connection

If your computer is connected to a local network (LAN, Intranet) or there are other network connections (e.g., DSL, modem), it may not be possible to access the ID 40 website or, if the SLK connection is active, the network resources due to your computer's network settings.

The following chapters recommend steps that should allow simultaneous access to the ID 40 website and any LAN.



If you are not familiar with network configurations, consult your network administrator. Do not make any ill-advised changes to the network settings.

10.4.1.1 Deactivating the default gateway

If your computer needs to access the Internet, an intranet, a file server, etc. in addition to the ID 40/SLK connection, the default gateway **cannot** be changed to the ID 40/SLK connection. Otherwise you will no longer be able to access your network resources once the connection to the SLK is established.

Deactivating the default gateway is described in Chapter 10.3.2 “Setting up a network connection” under the phrase **Use default gateway on remote network**.

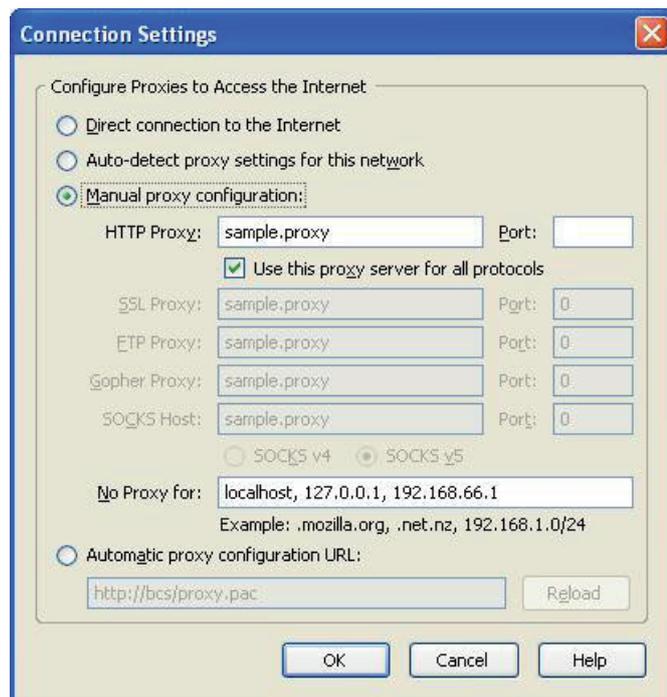
Terminate the connection (if active) and re-establish to apply the settings.

10.4.1.2 LAN connections via proxy server

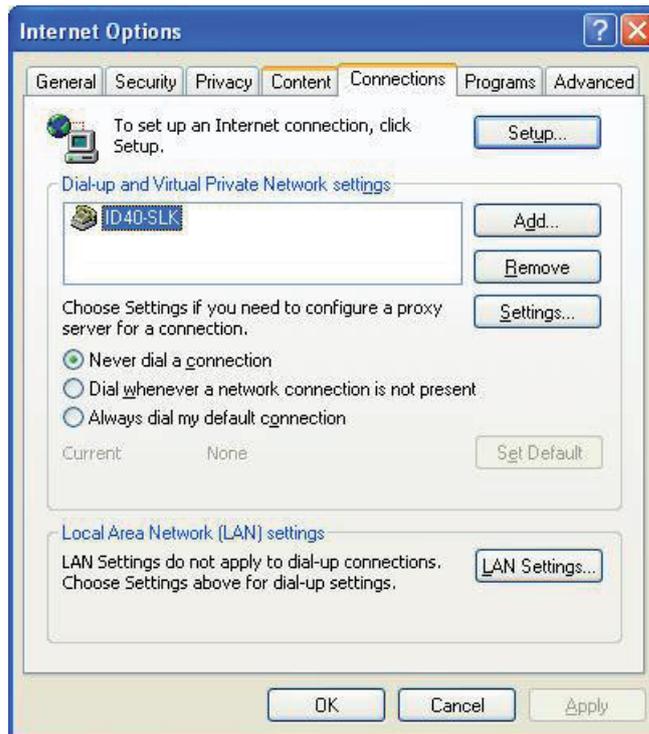
Addresses of proxy servers may be entered in your browser settings. Proxy servers store, e.g., web content to speed up access and they can sometimes prevent access to local web addresses, such as the ID 40/SLK. In this instance, the goal is to use a suitable connection configuration in the browser to allow access to the local ID 40 website.

There are many proxy configurations whose description would exceed the scope of this manual. Consult your network administrator if you are unsure how to proceed. If you are directly connected to the Internet, the following recommended settings do not apply.

Example using Firefox:



Enter the IP address of the ID 40/SLK in the **No Proxy for:**, possibly separated by a comma, as shown in the example.

Example using Internet Explorer:

1. Enter the same proxy configuration used for the LAN settings in the **settings** for the **ID 40-SLK (default)** connection.
2. If a **proxy server** is entered in the **settings** for the **ID 40-SLK (default)** connection, select **Bypass proxy server for local addresses**.
3. If **Use automatic configuration script** is selected in the settings for the **ID 40-SLK (default)** connection, do not make any changes. If the ID 40 website cannot be reached with this configuration, assign the name of the network address (see Chapter 10.4.1.3 “Assigning a name to the network address”) and try to access the website again.

10.4.1.3 Assigning a name to the network address

The IP address of the SLK can be assigned a name for the following reasons:

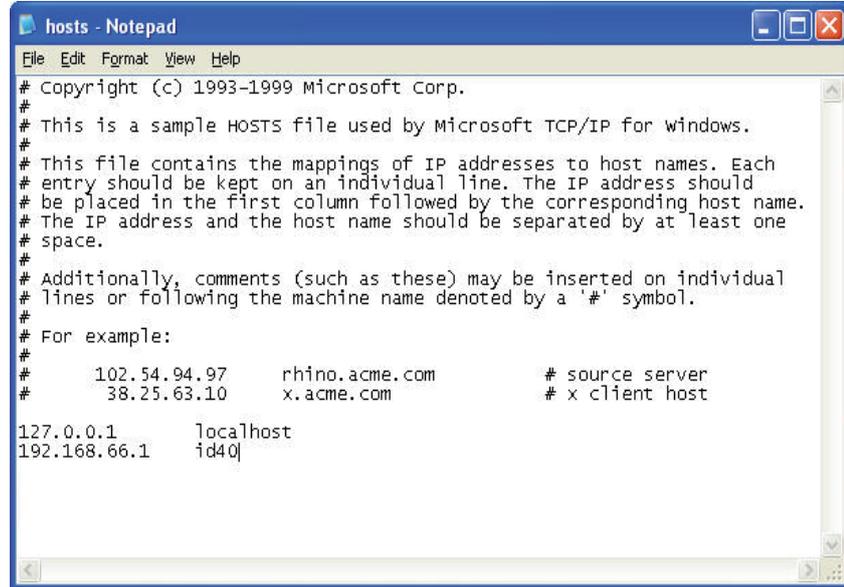
- It can allow access to the SLK homepage if your browser will not load the SLK website because the proxy settings are configured automatically (see Chapter 10.4.1.2 “LAN connections via proxy server”)
- It makes entering the web address easier
- It improves the readability of the web address

The name of the IP address is assigned in Windows XP and Windows 7 in the text file
 C:\Windows\system32\drivers\etc\hosts

- ▶ Open this file in an editor (e.g., Notepad) and add a new line with the IP address and name at the end of the file.

Example:

192.168.66.1 id40



```
hosts - Notepad
File Edit Format View Help
# Copyright (c) 1993-1999 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
# 102.54.94.97 rhino.acme.com # source server
# 38.25.63.10 x.acme.com # x client host
127.0.0.1 localhost
192.168.66.1 id40
```

- ▶ Save the file and close the editor.
- ▶ Now enter the following web address in the browser according to our example:
<http://id40/index.html>

10.5 The ID 40/SLK website

The ID 40/SLK website provides all data and information on the ID 40 system. If an MDT is located in front of the SLK at the time the website is accessed, all MDT data can be displayed and even edited.

The layout of the pages is self-explanatory and the minimal text is in English for international use.

The ID 40/SLK website can be accessed using your browser by entering the web address <http://192.168.66.1/index.html> or <http://id40/index.html> if “id40” was assigned to the IP address, see Chapter 10.4.1.3 “Assigning a name to the network address”.

10.5.1 Navigating and using the ID 40/SLK website

The ID 40/SLK website is divided into the “**ID 40/SLK Homepage**”, which contains links to the following pages:

- **MDT register:** contains the MDT status, MDT ID code and additional MDT information as described in Chapter 4.2.3 “MDT register area”
- **SLK register:** contains the actual link state and other information described in Chapter 5.3 “SLK register area”
- **MDT data:** allows MDT user data areas to be read and written, see Chapter 4.2.1 “MDT user data area”
- **Fieldbus-specific settings and information**, such as the Profibus node number
- **System statistics** and system information
- **System logging and shell**, supports system diagnostics as described in Chapter 12 “Diagnostics”

10.5.1.1 Switching the number format of displayed values

The default format of the SLK registers and fieldbus parameters is decimal, while MDT data and MDT registers are displayed in hexadecimal form by default.

The data format for the SLK registers, MDT registers and MDT data can be switched to decimal or hexadecimal. You can configure the MDT addresses “offset” and the displayed MDT data independently on the MDT data page. Just use the “Number Format” selection box.

Entering the values as described in the next section is completely independent of the configured display format.

10.5.1.2 Syntax when entering values

Numerical values can be **entered** using the following number systems:

Number system	Format	Value range
Hexadecimal	0xnn	0x00 – 0xFF
Decimal	nnn	0 - 255
Octal	0nnn	0 - 0377

Entering **address ranges** consists of a string of address ranges, each separated by a comma or a space. An **address range** consists of a value for the **start address**, two periods, then the **end address**.

Example of an address range:

0.34, 0x0200.1280, 5000.0x13bc



The end address always has to be greater than or equal to the start address. If both addresses are the same, the one byte at this address is returned.

10.5.1.3 Auto reload function

Auto reload automatically sends requests to the web server. This periodically updates the page content without having to select **Send request**. Select auto reload by clicking on the check box. You can configure the period between individual requests in the selection box.

Be sure to consider the loading time of the data, especially with larger data loads, and do not set an auto reload time that is too short.

Available in the following pages:

- 10.5.4 “SLK register page”
- 10.5.3 “MDT register page”
- 10.5.5 “MDT data page”

10.5.1.4 Prefetch function

As seen in the fieldbus systems, the web server also offers a prefetch function (see also Chapter 6.4.1 “Prefetch”). When **prefetch** is active, the SLK reads the data areas indicated in the address range from the MDT during the PRECONNECTED phase, see also Chapter 2.2.1.3 “Preconnected”.

- The prefetch function is available on the “MDT data page”, Chapter 10.5.5.
- The desired data is displayed as expected once the prefetch has completed.
- The web server only provides the prefetch data through the **Request** button or auto reload - not automatically like with the fieldbus (see Chapter 6.2 “Event-oriented data transmission”).
- The prefetches configured in the web server are independent of those in the fieldbus. They are **also** executed in the PRECONNECTED phase.

To use the prefetch, enter the desired address ranges in the “Address Range” input field, click the check box next to the prefetch and press **Send Request**. The web server applies the address ranges to the internal prefetch configuration and returns the data that is currently available. Red question marks (??) may appear instead of bytes if the current data is not yet valid. As always with prefetch, the data is only read from the **next** MDT sign-on.

Press **Send Request** again once the PRECONNECTED phase is complete. The desired MDT data will now appear.



The web server does **not automatically** start communication with the MDT. This occurs either through a commanded link state (commanded link state) on the SLK register page or by the application to which the SLK is connected.

If you want to retrieve **MDT registers** or SLK registers via prefetch, add the corresponding SLK addresses to the address ranges. See Chapter 7.4 “SLK address table” for the address assignments.

If you want to monitor an SLK in a fieldbus application with prefetches, the prefetches configured through the web interface work independently of those configured via the fieldbus.

The web server's prefetch access is deactivated when you deactivate the prefetch on the MDT data page. Fieldbus prefetches remain active and cannot be manipulated using the web interface.

10.5.1.5 Error on the MDT data page

As with the fieldbus, the MDT has to be in the *CONNECTED* state to receive valid MDT data when accessing the MDT through the web interface.

If no MDT is present during read access or invalid addresses are accessed, the web server marks the requested address ranges with red question marks.

The screenshot shows the 'MDT Data' page of the ID 40/SLK web interface. The status bar indicates a successful write operation: 'bal-write: 11 bytes successfully written'. The 'Address Ranges' field is set to '0x1de0..0x1e20' and 'Bytes per Line' is set to '16'. The 'Read Data' button is active. The data table below shows the following values:

Address	174	145	21	238	100	115	165	219	2	163	48	43	136	246	199	245
0x1de0	185	244	99	163	110	220	214	11	47	232	1	220	4	209	72	60
0x1e00	??	??	??	??	??	??	??	??	??	??	??	??	??	??	??	??
0x1e10	??	??	??	??	??	??	??	??	??	??	??	??	??	??	??	??
0x1e20	??															

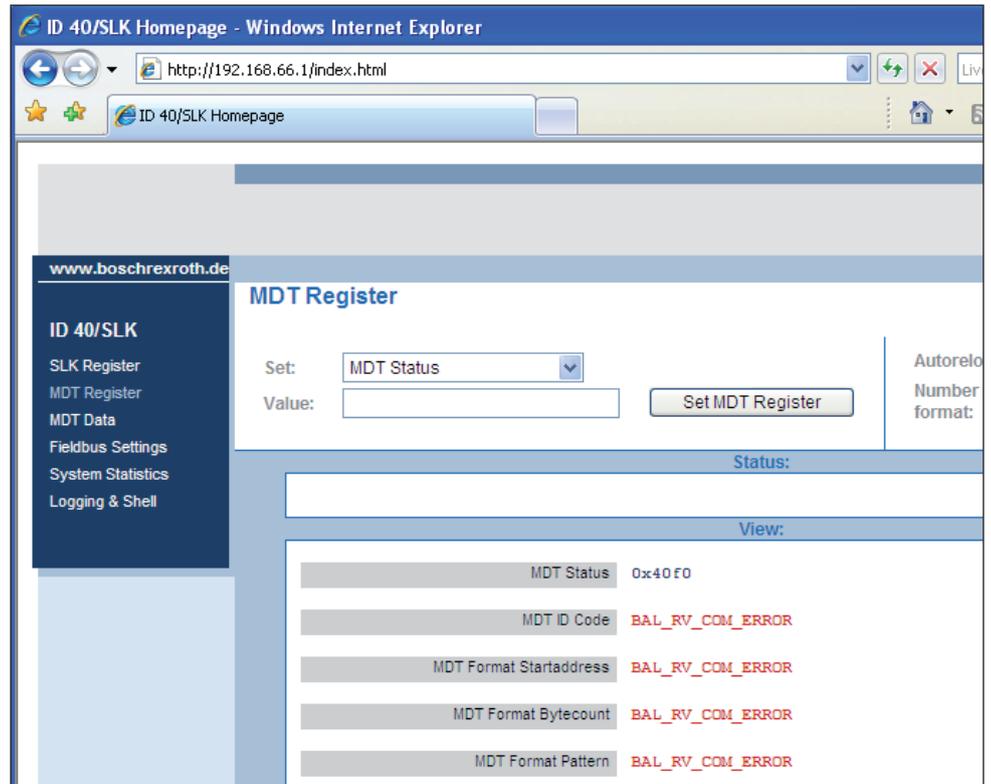
If the MDT data is sound, the values are displayed in dark blue.

If the read bytes come from an MDT segment with memory errors, the affected bytes appear in red.

If the system encounters a temporary access conflict, error codes appear in the data field in red.

10.5.1.6 Error on the MDT register page

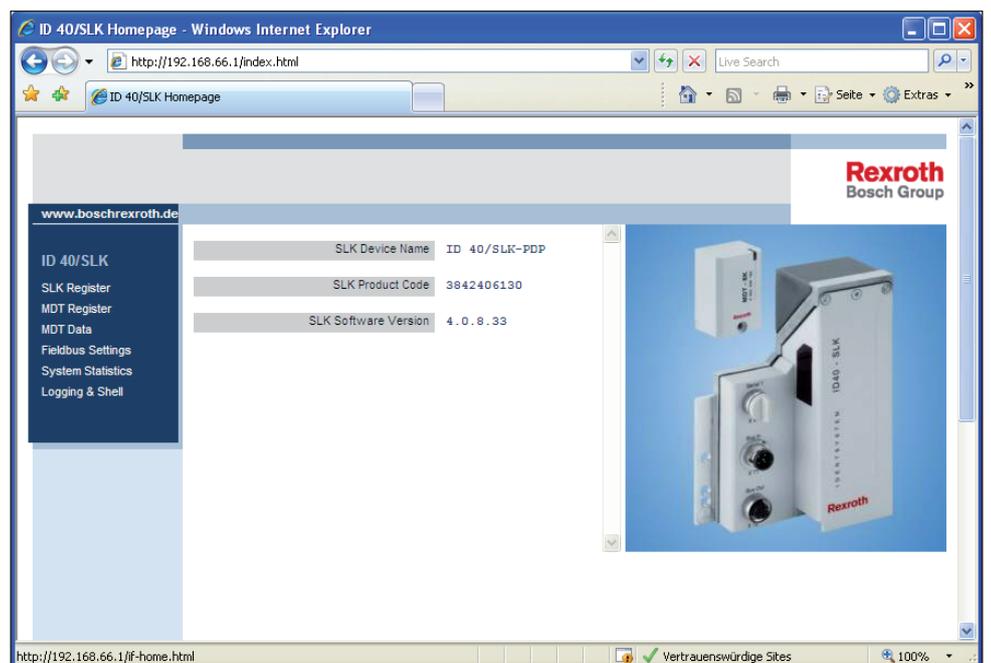
If the web interface was unable to read any values from the MDT when accessing MDT registers, an error code appears in red instead of the content of the registers:



ENGLISH

10.5.2 ID 40/SLK homepage

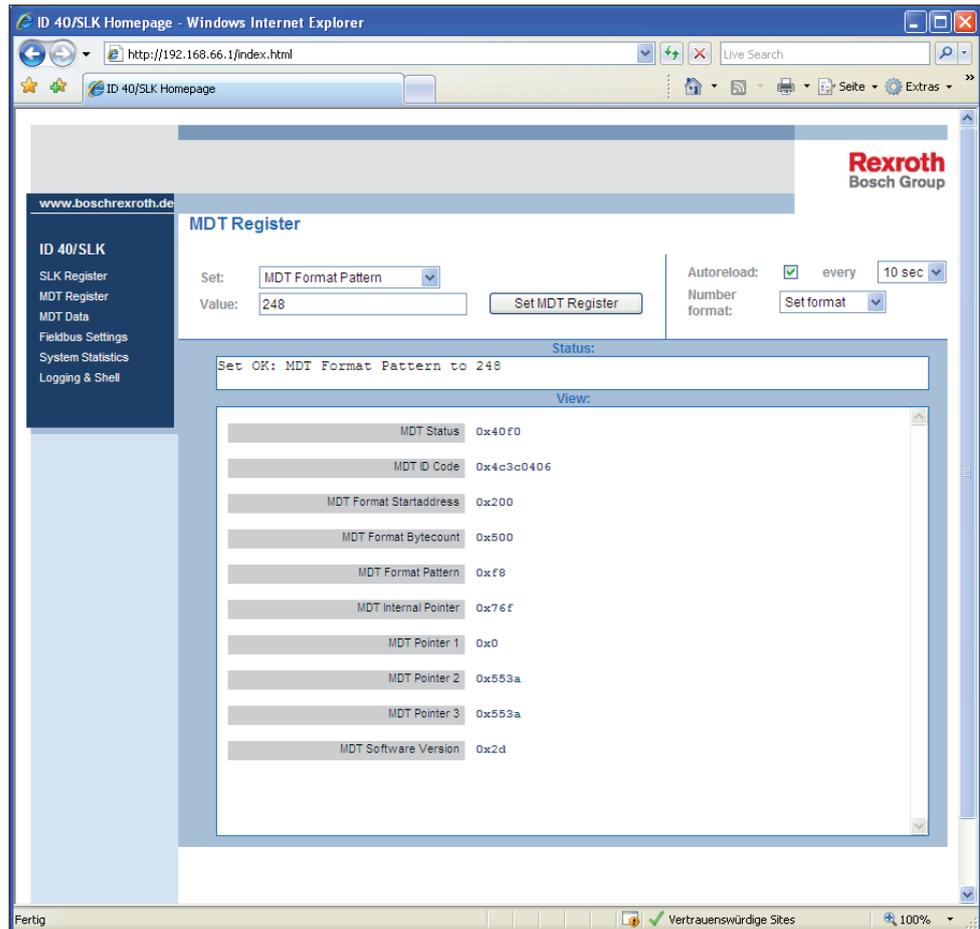
Product code, device name, firmware version and other information appear here.



Select the pages described below from the left navigation menu.

10.5.3 MDT register page

The **View** area of the MDT register data field contains the current values in hexadecimal format to the right of the register names in gray.



Modify individual registers by selecting the desired MDT register in the **Set** selection field, entering the corresponding value in the **Value** field and entering it in the MDT by selecting **Set MDT Register**.

The following MDT registers can be modified:

- **MDT status**: entering any value resets the MDT
- **MDT counter**: the MDT counter is set to the entered value
- **SLK operative flag**
- **Auto mode**
- **All Look Ahead Control registers**

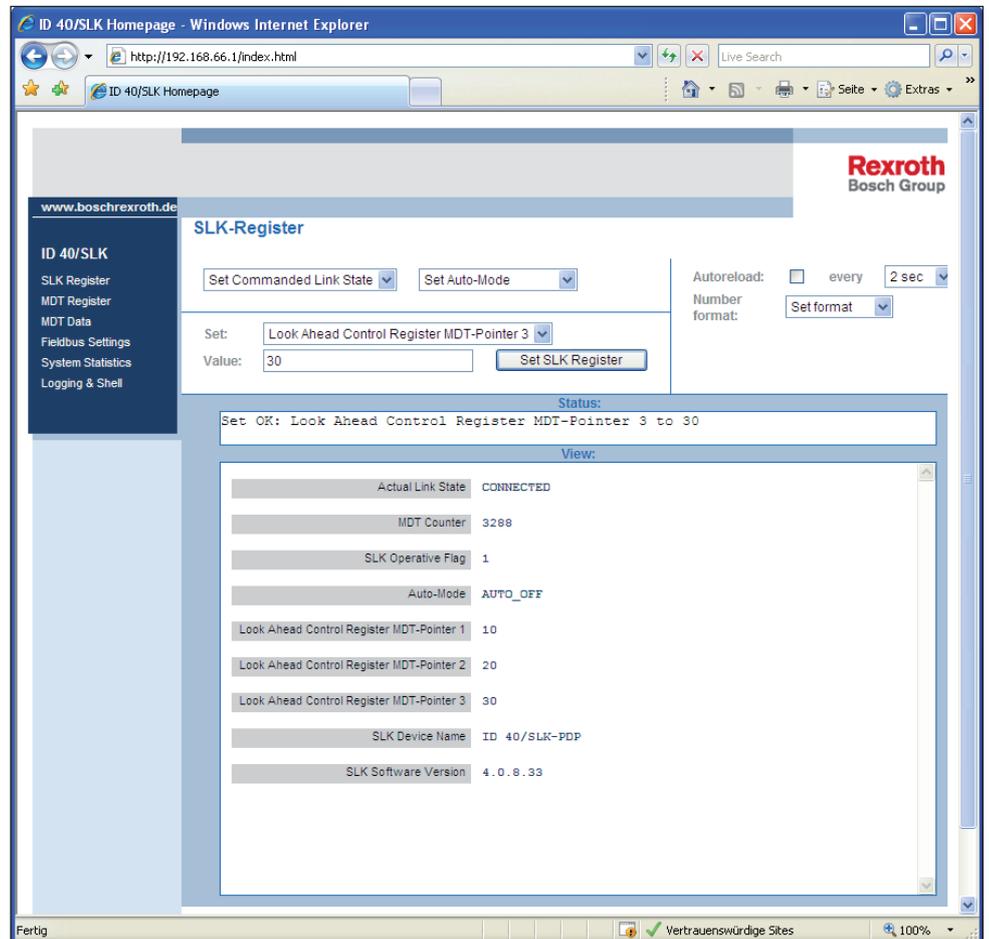
The value ranges and how the SLK registers function are described in Chapter 4.2.3 “MDT register area”.

The MDT register values are automatically updated after the values are changed.

The “Status” field shows the result of the write access. If no MDT is in the **CONNECTED** link state, the error message **BAL_RV_COM_ERR** appears.

10.5.4 SLK register page

The current values appear in the **View** area in decimal format. Since the website is in English, the most critical system state, the **actual link state**, appears under **Actual Link State**.



The following SLK registers can be modified:

- **Commanded link state**
- **MDT counter**
- **SLK operative flag**
- **Auto mode**
- **All Look Ahead Control registers**

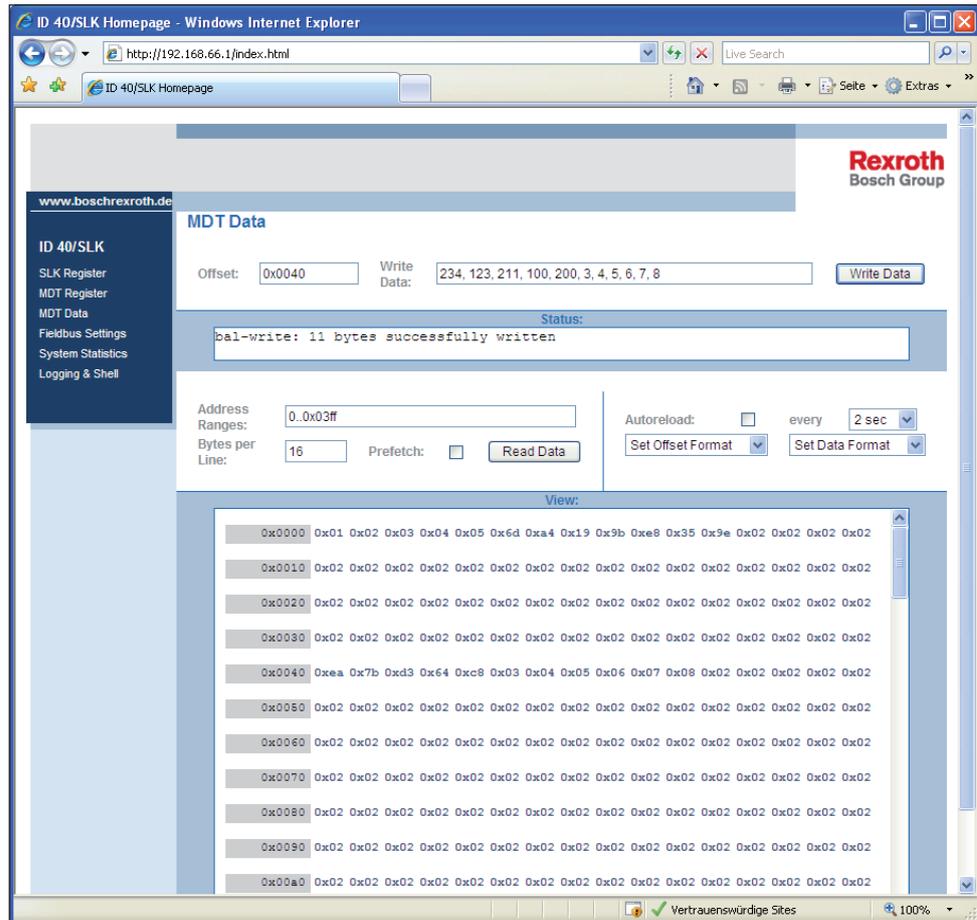
The register values are automatically updated after being configured.

The value ranges and how the SLK registers function are described in Chapter 5.3 “SLK register area”.

As with the fieldbus, note the processes during “data access to the ID 40/MDT” described in Chapter 6 when switching the commanded link state.

10.5.5 MDT data page

This page allows full write and read access to the entire MDT user data area.



To **read the MDT data**, enter MDT address ranges in the **Address Ranges** input field. Chapter 10.5.12 “Syntax when entering values” describes how to make the entries. Enter the number of bytes per line in **Bytes per Line**. When the **Offset** is displayed in hexadecimal format, 8, 16 or 32 is recommended; when displayed in decimal format, 10 is recommended.

The **Read Data** button reads the data from the MDT and displays it in the bottom window of the MDT data section.

Note that either the MDT has to be located in front of the SLK in the **CONNECTED** state or the data areas matching the request are from the previous MDT. If this is not the case, the web interface designates the bytes that cannot be read from the MDT with two red question marks, see Chapter 10.5.1.5 “Error on MDT data page”.

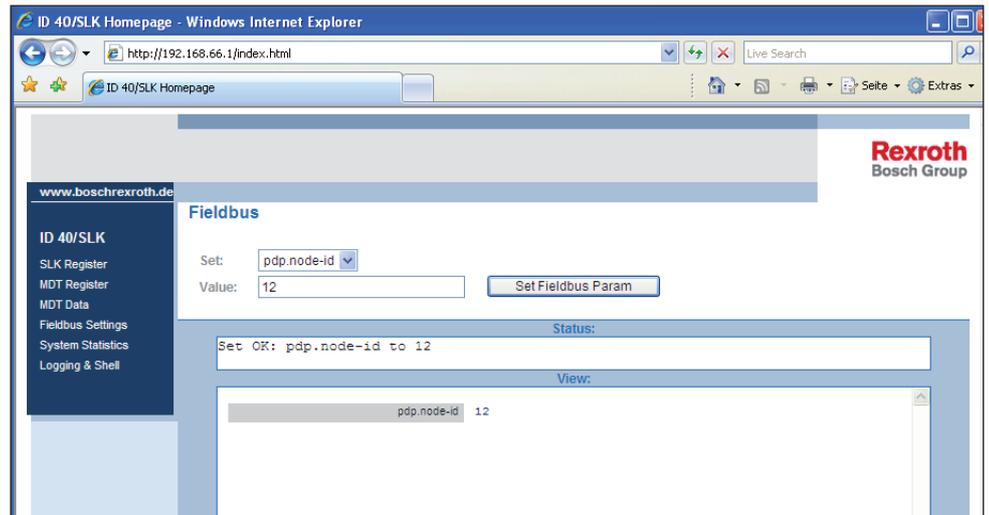
Transmitting the MDT data can take some time, depending on the amount of data requested.

If you want to **write MDT data**, enter the start address of the MDT data area in the input field next to **Offset**. Edit the data itself in the field next to **Write Data**. The format described in Chapter 10.5.1.2 “Syntax when entering values” also applies here. You can enter up to 92 bytes separated by commas or spaces.

Pressing the **Write Data** button sends the data to the MDT. The result of the writing is shown in the **Status** field. If an MDT is in the **CONNECTED** link state, **bal_write: x bytes successfully written** appears, whereby **x** is the number of bytes entered. The error message **BAL_RV_COM_ERR** appears when the link state is **not CONNECTED**. In this instance, data cannot be written.

10.5.6 Fieldbus settings page

Configure the fieldbus parameters, such as the Profibus node number, here.



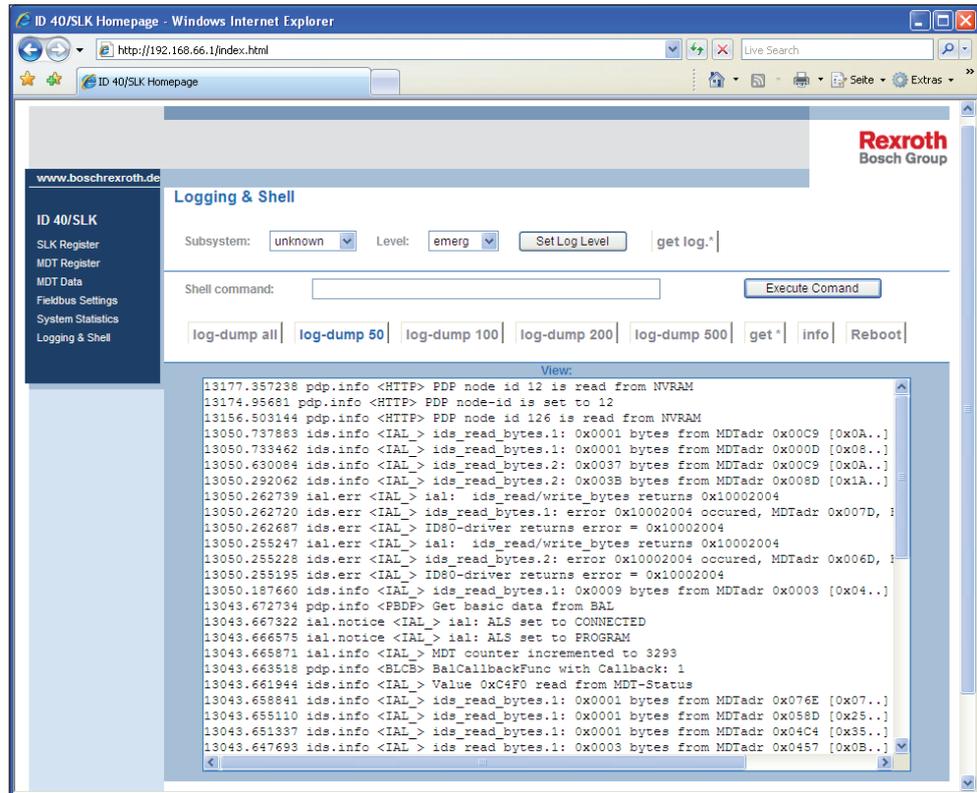
Enter the desired value in the input field, then submit with **Set Fieldbus Param**. The fieldbus-dependent parameters themselves are described in Chapter 11 “Start-up and parameterization”.

10.5.7 Systems statistics page

The statistics data is used to evaluate the quality of HF communication between the SLK and the MDT.

10.5.8 System log and settings page

This page supports the ID 40 system diagnostics.



In the event of unexpected system behavior, you can use the **log-dump 50**, **log-dump 100**, etc. buttons to download the system log, or “syslog”. The number indicates the number of syslog lines. Right clicking in the log text and selecting the context menu item **Save target as...** (context menu depends on browser) allows you to save the syslog to your computer. The download can take some time.

10.6 Tips and tricks

10.6.1 Bookmarks

Bookmarks may already be familiar to you from surfing the Internet. Bookmarks also allow quick access to the homepage of the ID 40/SLK and even individual functions on the ID 40 website.

Once the ID 40/SLK homepage has loaded, save the web address as a bookmark on your browser.

10.6.2 Directly retrieving the syslog

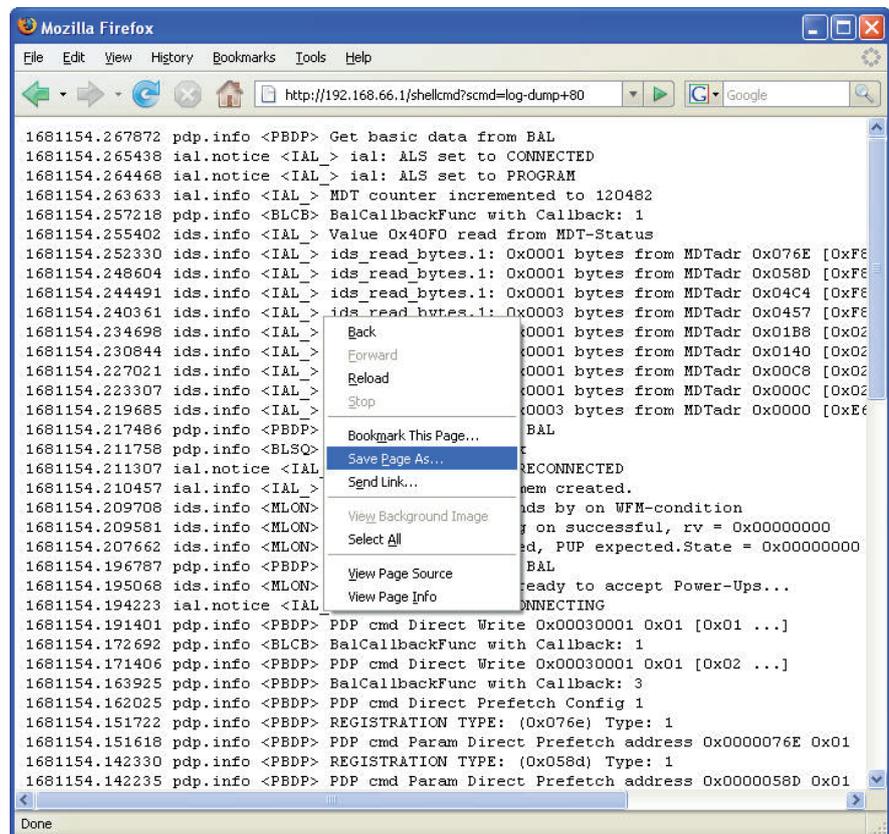
If you want to export the syslog multiple times during troubleshooting and save it to a file to analyze later on, it can be beneficial to directly retrieve the syslog using a web address instead of in the frame of the website.

Background: Versions of Microsoft Internet Explorer do not support saving frame contents.

The following web address can be used to display the last 80 lines of the syslog as text in your browser (the current entry is at the top):

<http://id40/shellcmd?scmd=log-dump+80>

Select **Save as** from the context menu (right click in the text field) to save the log dump as a file:



10.7 Web access from application programs

The ID 40 system can be controlled and data exchanged with MDTs from a PC-based application program by using the **web interface**. The requirements are the same as those for using a browser (see Chapter 10.2 “Requirements”).

- To exchange data, the application program sends command strings in the form of a web address (URL) as shown in the “Directly retrieving the syslog” example (see Chapter 10.6.2 “Directly retrieving the syslog”). In the following chapters, the IP address of the ID 40/SLK is assigned the name **id40**, as described in Chapter 10.4.1.3 “Assigning a name to the network address”.
- In addition to the IP address of the ID 40/SLK, the URL string contains the identifier of the read or write command as well as the SLK address ranges from Chapter 7.7 “Addressing data in the ID 40 system” and the data being written.
- Data read by the ID 40 is returned in XML format in a structured manner.
- The ID 40 web interface only returns data after a request. It is not possible to transmit data by event.
- The pretransmit function is not supported by the web interface.

10.7.1 Web interface data formats

10.7.1.1 Numerical values

The numerical values in the web address and XML structure are bytes ranging from 0–255, defined as `unsigned char`.

Other number formats (e.g., larger value ranges) can be used in the application, but have to be implemented into an array of bytes when generating the URL for a write command.

See Chapter 10.5.1.2 “Syntax when entering values” for supported number systems.

10.7.1.2 Web interface URL syntax

Parameter	Meaning	Value range
http://id40	Transfer protocol and IP address for the ID 40/SLK-PDP	See Chapter 10.4 "HTTP connection to ID 40/SLK"
xid	Constant identifier for the ID 40 data area	
bal-mem	Designates read access to the ID 40 system	
bal-write	Designates write access to the ID 40 system	
offset, adrset	SLK address ranges in which the data is entered with <code>bal-write (offset)</code> or read with <code>bal-mem (adrset)</code> are assigned to this parameter	See Chapter 7.7 "Addressing data in the ID 40 system"
prefetch	A value of 1 means the <code>adrset</code> is read from the ID 40 system and then saved as a prefetch. The value 0 deletes all marked prefetches, see Chapter 10.7.3.3 "Web interface prefetch"	'1', '0'
fmt-offset, fmt-data:	Output format of the address and data bytes in the XML string	"hex", "dec"
data	Byte array being written, bytes separated by "+" or ","	0 ... 255

10.7.2 Web interface write commands

These commands write data to the MDT and also switch the link state. A completed write access of <n> bytes is confirmed with the string:

```
bal-write: <n> bytes successfully written
```

Other confirmations refer to invalid URL syntax or access during an invalid link state. A write command contains the start address <start-adr> of the write process as well as the data being written in bytes <data-n> and the number of bytes n.

```
http://id40/xid/bal-write?offset=<start-adr>&data=<data-1>+<data-2>+...+<data-n>
```

The start address defines the ID 40 area (MDT data, MDT register, SLK register, see Chapter 7.4 "SLK address table") in which the data should be entered.

The **length of the URL** is limited, so only 220 string characters can be used for the data being written.

A maximum of 72 bytes can be written in two-digit decimal format, with a maximum of 44 bytes in hexadecimal format.

10.7.2.1 Write command examples

Switching the link state using a commanded link state (see Chapter 5.3.2). The URL contains the SLK address of the commanded link state 0x30001 in the `offset` parameter and data is assigned to the code from the table in Chapter 5.3.2.

CONNECT

```
http://id40/xid/bal-write?offset=0x30001&data=1
```

DISCONNECT

```
http://id40/xid/bal-write?offset=0x30001&data=2
```

RECONNECT

```
http://id40/xid/bal-write?offset=0x30001&data=3
```

Writing data to the MDT starting at MDT address 0x0020, the bytes are 10, 11, 12, 13, 14, 15, 16, 17.

```
http://id40/xid/bal-write?offset=0x20&data=10+11+12+13+14+15+16+17
```

The SLK address 0x00000020 assigned to the `offset` parameter (corresponds to MDT address 0x0020 according to the table in Chapter 7.4) should be entered with no leading zeros in order to not extend the URL string needlessly.

The web interface confirms the write access with the string:

```
bal-write: 8 bytes successfully written
```

Entering the Profibus node number

Number 54 in this example

```
http://id40-slk/xid/set?path=xid.univar.pdp.node-id&value=54
```

Successful entry is confirmed with the string:

```
Set OK: pdp.node-id to 54
```



Attention: The value entered for the Profibus node number is not checked. It is recommended to read back the Profibus node number after entry in order to confirm it is correct. Another way is to check the display after restarting the ID 40/SLK, see Chapter 11.1 “Starting up Profibus DP”.

10.7.3 Web interface read commands

These commands are used to read data from the MDT as well as data from SLK registers, such as the actual link state. A read command consists of at least one parameter, `adrset`, which is assigned at least one (or more) SLK address ranges. An SLK address range is specified by the start and end addresses. These addresses are separated by two periods.

```
http://id40/xid/bal-mem?adrset=<start-adr>.<end-adr>
```

Multiple data areas can be read with a single command by stringing the address ranges together with a “+”.

```
http://id40/xid/bal-mem?adrset=<start-adr-1>.<end-adr-1>+...+<start-adr-n>.<end-adr-n>
```

A maximum of 64 kB can be requested with a read command, i.e., an entire ID 40/MDT32K can be read in one pass.

10.7.3.1 Web interface data output

The ID 40 web interface responds to every valid read command with an XML string.

This string contains the following XML elements:

- XML header, Cascading Style Sheets (CSS) for displaying data in browsers, <xid> framing element. These elements are not relevant to gathering the requested data.
- <bal-mem>
All requested data areas are shown consecutively here
- <bal-range>
One of the requested data areas
- <offset>
The start address of this data area
- The data bytes themselves.
Every data byte is designated by its own error flag and executed as an XML tag with the byte
 - byte: no error
 - <ec> error connection: error transmitting between MDT and SLK;
these locations are displayed as follows: <ec>??</ec>
 - <ed> error device: byte read from flagged MDT data area

Example of an XML string

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet href="/id40xml.css" type="text/css"?
<xid>
  <bal-mem>
    <bal-range>
      <offset>0x0000</offset>
      <data>
        <b>0x37</b>
        <b>0x36</b>
        <b>0x35</b>
        <b>0x34</b>
        <b>0x33</b>
      <data>
    </bal-range>
  </bal-mem>
</xid>
```

10.7.3.2 Examples of read commands

Read actual link state

This read command returns the byte with the link state coding. The coding is described in Chapter 5.3.1 “Actual link state”.

`http://id40/xid/bal-mem?adrset=0x30000`

XML output, whereby the link state value read here = 4:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<?xml-stylesheet href="/id40xml.css" type="text/css"?>
<xid><bal-mem><bal-range><offset>0x30000</offset><data>
<b>0x04</b></data></bal-range>
</bal-mem></xid>
```

Read bytes from MDT and SLK register

First address range 0 – 99: 100 bytes of MDT data

Second address range 0x1200 –4700: 93 bytes of MDT data

Third address 0x00030000: 1 byte for link state

Hexadecimal output formats for address and data:

`http://id40/xid/bal-mem?adrset=0.99+0x1200.4700+0x30000&fmt-
offset=hex&fmt-data=hex`

Reading the Profibus node number

The Profibus node number is read using a special system command.

The URL

`http://id40-slk/xid/univar/pdp`

returns the following XML string:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<?xml-stylesheet href="/id40xml.css" type="text/css"?>
<xid><pdp><node-id><var><desc>pdp.node-id
</desc><value>100</value></var></node-id>
</pdp>
</xid>
```

The node number is in the range `<value ... </value>`, in this example = 100

10.7.3.3 Web interface prefetch

With the URL parameter `&prefetch=1`, the SLK address ranges specified in `adrset` are read from the current MDT and also automatically copied from each subsequent MDT to the SLK, however they are **not** transmitted from the web interface by event. The MDT data buffered in the SLK can then be retrieved using the same read command once the MDT has left the HF link.

Attention: The next MDT overwrites the buffered data, so the read access has to occur beforehand.

The prefetch function is described in Chapter 6.4.1. The web interface prefetch is similar to the buffered prefetch in Chapter 6.4.2.1 “Single transmit”.

Example:

The high byte of the MDT status register and the MDT data area from addresses 200–399 are read and also applied to the prefetch configuration. This `adrset` is automatically read starting with the next MDT.

```
http://id40-slk/xid/bal-mem?adrset=200.399&prefetch=1
```

- The prefetch read command can only be sent in the *CONNECTED* state, otherwise the prefetch `adrset` is not entered. The data from the current MDT is returned.
- The prefetch is only activated for the `adrset` specified in the command string. Any deviating prefetch settings from a previous command are automatically deleted.
- If the `&prefetch` parameter is omitted from the read command, the data is returned, but all prefetch settings are deleted. This also applies to the `&prefetch=0` parameter.

11 Start-up and parameterization

11.1 Starting up Profibus DP

Every SLK comes from the factory with the node number 126 preset.

To operate the SLK on a production line, only this node number has to be changed according to the existing Profibus topology.

There are two ways to do this:

- Configuration and diagnostics software
- With the Profibus DP master

One easy way to label the SLK is to write the configured node number on the provided area of the SLK nameplate with a dry erase marker.



If the node number is later changed, the label also has to be changed to avoid misinterpretation.

The PDP node number is briefly displayed after turning on the SLK.

Example: A node number of 4 is configured:

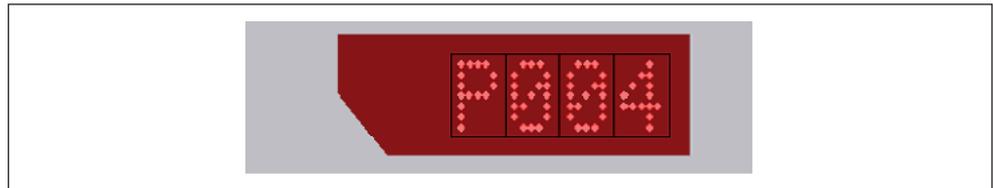


Fig. 32: Display of configured node number

11.1.1 Configuring through the web interface

The “Fieldbus Settings” link on the ID 40/SLK homepage allows the PDP node number to be read and configured, see Chapter 10.5.6 “Fieldbus settings page”. The new node number is only applied by the system when the SLK is temporarily offline on the fieldbus. The fieldbus indicators cannot appear on the SLK display (see Chapter 2.2.2 “Status display”).

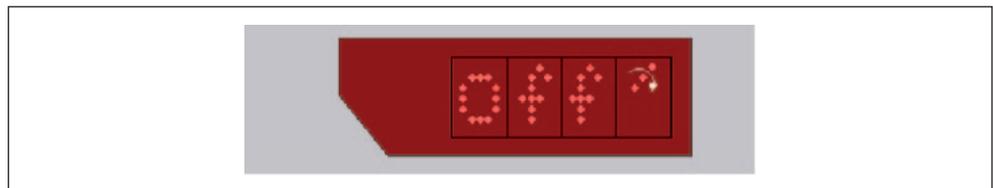


Fig. 33: The fieldbus activity indicator on the display is off

11.1.2 Configuring with Profibus master

A connected PDP master can replace the existing node number with a new one. To do this, no other slave can be connected to Profibus.

11.1.3 Starting the Profibus master

The left fieldbus activity indicator lights up once the Profibus master has activated the SLK on the bus. The right fieldbus activity indicator flashes when commands are being transmitted to the Profibus.



Fig. 34: The left fieldbus activity indicator lights up on the display

11.2 Starting up Interbus

The following steps are necessary to start up the ID 40 system on Interbus:

- Connect the SLK to Interbus, connect the 24 V supply voltage.
- Configure the outgoing remote bus signals depending on the position of the SLK on Interbus, see Chapter 11.2.1 “Outgoing remote bus configuration”.
Alternatively, the Interbus terminating bridge can be used as described in that same chapter.
- Configure the desired maximum possible PDU size if the default value of 240 is not desired.
- Start up the Interbus with the IBS CMD SWT G4 software by Phoenix Contact. See Phoenix Contact for information on this program.

11.2.1 Outgoing remote bus configuration

If there are no other bus users on an ID 40/SLK-IBS, the outgoing remote bus cannot be routed. In common Interbus devices, this is detected at the output of the Interbus bus connection using a jumper.

Since the ID 40/SLK uses a 5-pin M12 plug to save space, there is no pin available for the standard RBST signal to turn off the outgoing remote bus. This is why the outgoing remote bus is configured via the RS232 interface using the ID 40/KONF program.

11.2.1.1 Outgoing remote bus configuration using browser

The ID 40/SLK homepage supports configuring the outgoing remote bus (see also Chapter 10.5.6 “Fieldbus settings page”).

1. **ibs.outgoing** is preselected by choosing the selection box under **Set**.
2. Enter the value **0** or **1** in the “Value” field.

- **ibs.outgoing 1:** outgoing remote bus is routed (ORB indicator lights up on the display)
- **ibs.outgoing 0:** outgoing remote bus is terminated (ORB indicator turns off on the display)

After setting the value with **Set Fieldbus Param**, the current settings appear under **View**.

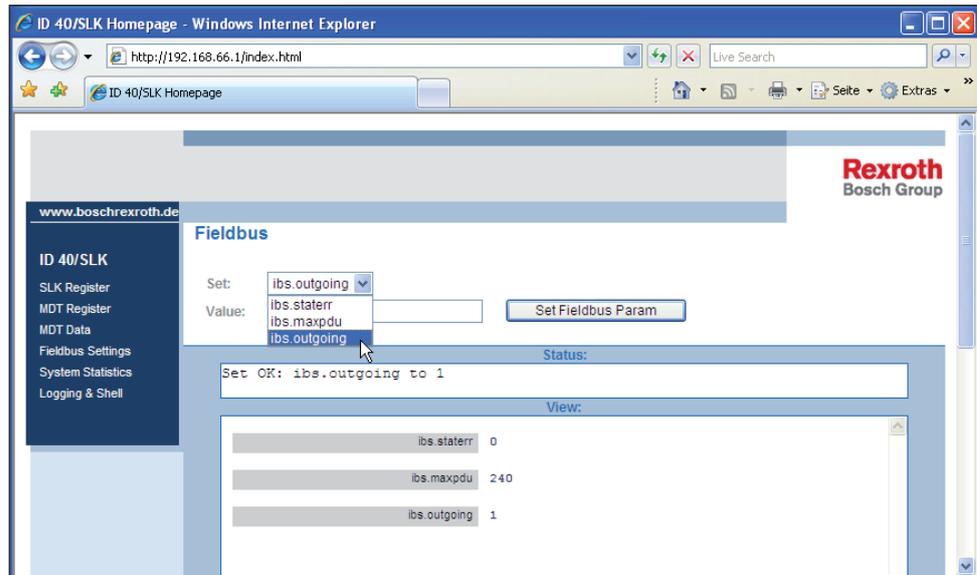


Fig. 35: SLK configuration for outgoing remote bus

11.2.1.2 Returning outgoing remote bus configuration using terminating bridge

Alternatively, the ID 40-specific Interbus terminating bridge can be used to “terminate” the IBS at the last SLK. See Chapter 14 “Overview for ordering ID 40 modules” for terminating bridge ordering information.

The bridge is in the form of an M12 plug and is screwed onto the “Bus Out” contact (X72). This bridges the outgoing remote bus to the incoming remote bus to close the IBS loop. This renders configuring via RS232 ineffective and unnecessary.

11.2.2 Configuring the maximum PDU size of the PCP channel

The PDU size determines the number of bytes that can be provided in the PCP channel. The size configured here is applied with the CMD software for PCP communication while configuring the Interbus. See also Chapter 8.3.1 “PCP communication”.

The value can be configured from a range of 51 to 242 on the **“Fieldbus Settings” page** (see Chapter 11.2.1.1 “Outgoing remote bus configuration using browser”). Enter the value in the **Value** field. The default value is 240 bytes.

The SLK has to be restarted and the Interbus reconfigured with the CMD tool in order to apply the new PDU size.

The new settings are retained even after SLK voltage recovery.

11.2.3 Configuring with the IBS CMD SWT G4 software

The IBS CMD SWT G4 parameterization tool is used to plan the overall configuration of the Interbus system.

See www.Interbusclubs.com for more information.

11.2.4 Showing Interbus status on the display

The following fieldbus states are active when the corresponding symbol appears on the display:

- **BA bus active:** the Interbus master is in the “running” state
- **RD remote bus disable:** the Interbus master has not yet initialized the bus.
Transition state after turning on voltage
- **CC cable check:** the bus cable is connected
- **TR transmit/receive:** data is being exchanged via PCP communication. The duration of this symbol depends on the data transmission. With sporadic PCP communication, the symbol only flashes briefly and may not always be visible
- **ORB outgoing remote bus active:** the outgoing remote bus is routed. This state is configured using the ID 40/KONF diagnostics and parameterization program (see Chapter 11.2.1 “Outgoing remote bus configuration”)

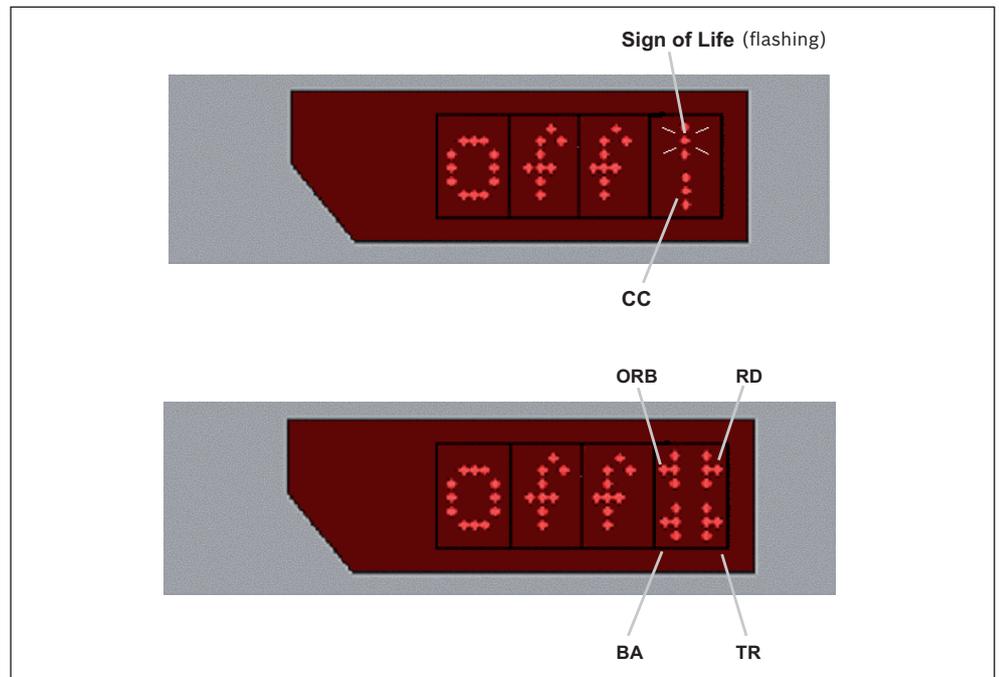


Fig. 36: Interbus fieldbus indicators

11.3 Starting up CANopen

The following steps are necessary to start up the ID 40 system on CAN:

- Configure the node number (= CAN node ID), see Chapter 11.3.1 “Configuring the bus parameters”.

The value range for the node number is 1 to 127



A node number of 0 is not allowed.

- Configure the baud rate
Default: 500 kbps
Possible values [kbps]: 10, 20, 50, 100, 125, 250, 500, 800, 1000
- Configure the CANopen master using the EDS file, see Chapter 9.18 “Electronic data sheet (EDS)”
- SDO and/or PDO communication services are used depending on the planned application. These services also have to be configured.

Example: Assigning the PDO channels to the PLC I/O modules

11.3.1 Configuring the bus parameters

The CAN node ID and the CAN baud rate are configured on the **Fieldbus page** (see Chapter 10 “Web interface”).

The values are entered in decimal format and are applied after the SLK is restarted.

The new settings are retained even after SLK voltage recovery.

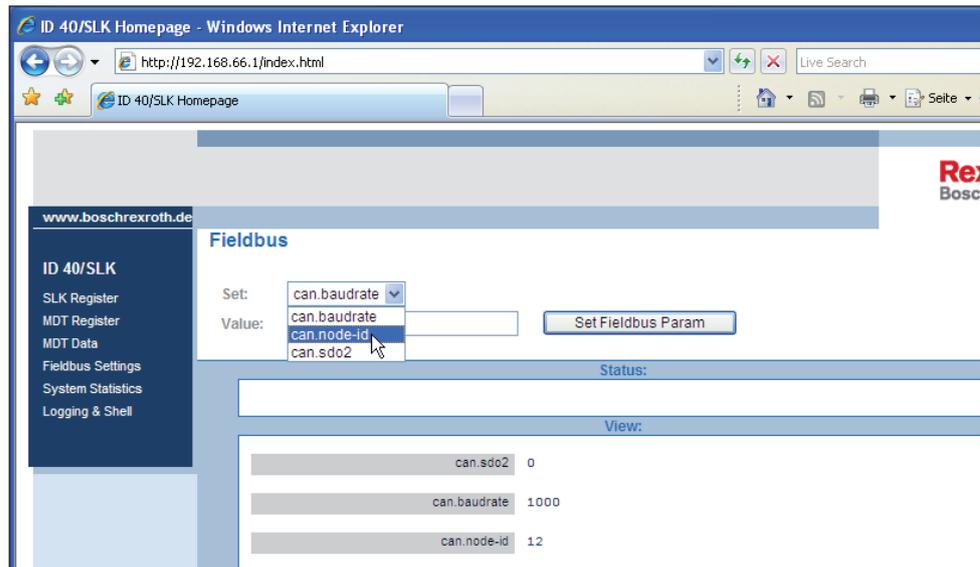


Fig. 37: Fieldbus page

11.3.2 ID 40/SLK-CAN boot-up behavior

The system boots up after the SLK is turned on (24 V power supplied).

Once the system has booted up, the CAN controller is initialized according to the configured parameters. Node number and baud rate appear briefly on the display.

Once initialization is complete, the module is in **PRE-OPERATIONAL** mode. No fieldbus indicators are visible in area 2 of the display (see Chapter 11.3.3 “Showing CANopen status on the display”).

In PRE-OPERATIONAL mode, PDO settings can be made by command if necessary.

The SLK is then put into **OPERATIONAL** mode by an “NMT START” telegram from the CANopen master.

Once the module is in OPERATIONAL mode, event-oriented data exchange via PDOs is possible.



Do not disconnect the bus cable during operation, as this will result in BUS OFF. The system then has to be restarted.

11.3.3 Showing CANopen status on the display

The configured CAN node ID (node number) and the transfer rate are briefly displayed on system start.

Example 1: Node ID 32 and baud rate 1,000 kbps (= 1.0 Mbps):

C032--1.0M

Example 2: Node ID 4 and baud rate 500 kbps

C004--500k

The display shows the character as a “ticker”.

During PRE-OPERATIONAL mode, the display does **not** show any fieldbus indicators in area 2 of the display (see Chapter 2.2.2 “Status display”). The usual link state is shown in area 1.

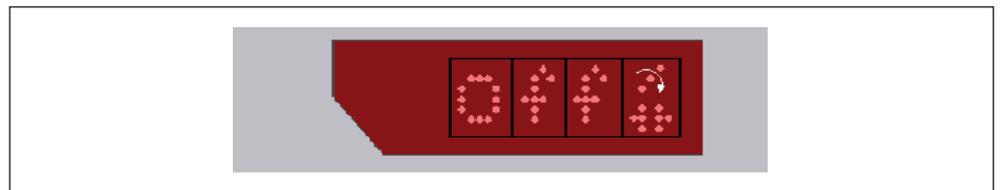


Fig. 38: CANopen fieldbus indicators

After switching to OPERATIONAL, area 2 of the display shows a solid left fieldbus indicator and a flashing right indicator when PDO/SDO communication is running. If the CAN bus is electrically interrupted, the CAN controller in the SLK switches to the BUS OFF status. The display then shows “BUS”. The SLK has to be restarted to continue operating once the error is corrected in the bus.

11.3.4 Configuring the second SDO channel

The second SDO channel in the SLK can be helpful for diagnostics and troubleshooting. The SDO2 is not needed for standard operation of the SLK in production.

The SDO2 has to be activated before it can be used.

This is done in two steps:

1. Activate the SDO2 function on the **Fieldbus page** by entering a "1" in the appropriate input field and sending it to the SLK.
This state is retained even after voltage recovery.
2. Enter the **COB ID pairs** in object 1201. Data format: UNSIGNED32

Index	Subindex	Description	Default values
1201	00	No. entries	2
	01	COB ID client > server (rx)	Not valid 0x0680 + node ID + 0x40
	02	COB ID server > client (tx)	Not valid 0x0680 + node ID

The structure of the COB ID value is described in [C17].

When using the SDO2, only node numbers 1–63 are permitted.

The second SDO channel can be turned on and off on the **Fieldbus page** of the web interface.

11.4 Adjusting the baud rate of the serial interface

The baud rate of the serial interface is converted from 9600 bps to 115200 bps in SLK software version 4.0 and higher in order to accelerate page loading times on browsers (see Chapter 10 "Web interface").

If you use the ID 40/KONF instead of the web interface or you want to modify the network addresses for the web connection, the baud rate has to be configured to 115200. The ID 40/KONF manual describes how to do this.

If the ID 40/KONF tool is active during SLK restart, you will first see illegible characters in the terminal window, then the input prompt will appear after a few seconds.

```
ID 40/SLK >
```

However, the baud rate has to be returned to 9600 in order to load a new software version to the SLK using the **upload** function in ID 40/KONF. The ID 40/KONF manual describes how to upload software.

12 Diagnostics

This chapter is intended to assist you in determining the cause of problems or errors.

12.1 Troubleshooting guide

Consider the following when troubleshooting:

What was the **situation in the system** when the error(s) occurred?

1. After starting up the system/a workstation
2. After modifying/expanding the system or PLC program
3. While the system was running normally

The first and second instances can be attributed to an installation or start-up error, or errors following a modification or reprogramming.

The third instance could be caused by operator error (e.g., manual workstation, trainee) or defective equipment.

12.1.1 Important information that can indicate errors

Error indications in the application program (PLC)

- Error codes in response to command-oriented data exchange
- Actual SLK link state
- Error messages from the PDP master
- Error bits in the MDT status

Error indications on the SLK display

- The SLK display flashes “E00” for a fault during MDT sign-on or sign-off after the DISCONNECT or RECONNECT commanded link state
- No fieldbus indicators, SLK is not online on the bus
- The entire display flashes, regardless of content. See Chapter 12.1.2 “Fatal system errors” for what this means
- Sign of life stops rotating, system error

Error indications on MDT and SLK

- HF antenna cover does not light up when the MDT enters the SLK area
- MDT LED remains off when the MDT enters the SLK area
- MDT LED shows red

12.1.2 Fatal system errors

When a fatal system error occurs, the ID 40 system can no longer work in a controlled fashion. This is why the SLK stops all functions. This state is comparable to the “blue screen” seen in Microsoft Windows operating systems

System status after fatal system error

- The entire display flashes continuously, including the indicators in display area 2 (see Chapter 2.2.2 “Status display”). The content of the display shows the last state before the fatal system error occurred. Note this for diagnostic purposes
- The HF field to the MDT is turned off for safety reasons. Here it is possible that a linked MDT has encountered a communication error

- Details on the system error can be read via the RS232 interface. Start the ID 40/KONF program after connecting the diagnostics cable to the COM port of a computer (see also ID 40/KONF manual).
Pressing the ENTER key displays information on the fatal system error as well as the ID 40/SLK emergency shell:

```
Assertion failed: ..

ID 40/SLK Emergency Shell
Commands: m - dump RAM pagewise, M - dump all RAM, l - logdump
B - burn log to flash, E - erase logs in flash, R - reboot slk
```

Pressing the “L” key outputs the system's syslog buffer. This makes it possible for Service to trace the source of the error.

Save the log dump initiated by pressing the “L” key as a file and send it to Service along with the text starting with “Assertion ..” or “Exception ..”.

The system can be restarted by pressing “R” (reboot). This corresponds to a cold start, so the PLC application may also have to be reinitialized and restarted.



It is no longer possible to operate the ID 40 website after a fatal system error.

Possible causes of errors:

1. Defective equipment
2. Pretransmit function being on constantly without being deactivated occasionally.
This can result in memory overflow in the SLK
3. Access to invalid SLK address ranges with older SLK software versions (3.00 or lower), see Chapter 5 “SLK storage”

What to do when a fatal system error occurs

1. Save at least one line of the running text from the ID 40/KONF using a log file, a screen shot (Alt+Print Screen) or writing it down. Send it to Bosch Rexroth Service so they can analyze the error
2. Reset the SLK by turning it off, then on again. If the error persists and causes 2 and 3 have been ruled out, contact Bosch Rexroth Service

System behavior with older SLK software versions (3.00 or lower)

- A ticker is visible on the display showing information on the error
- The syslog may still be readable with ID 40/KONF. If possible, send the content to Bosch Rexroth Service

12.1.3 Diagram of possible causes of errors

The following **chart** of error indications to causes assumes that an error occurring in conjunction with the ID 40 system is first detected in the **application program** of the PLC. The displays on the MDT and SLK are visually checked afterward, if necessary.

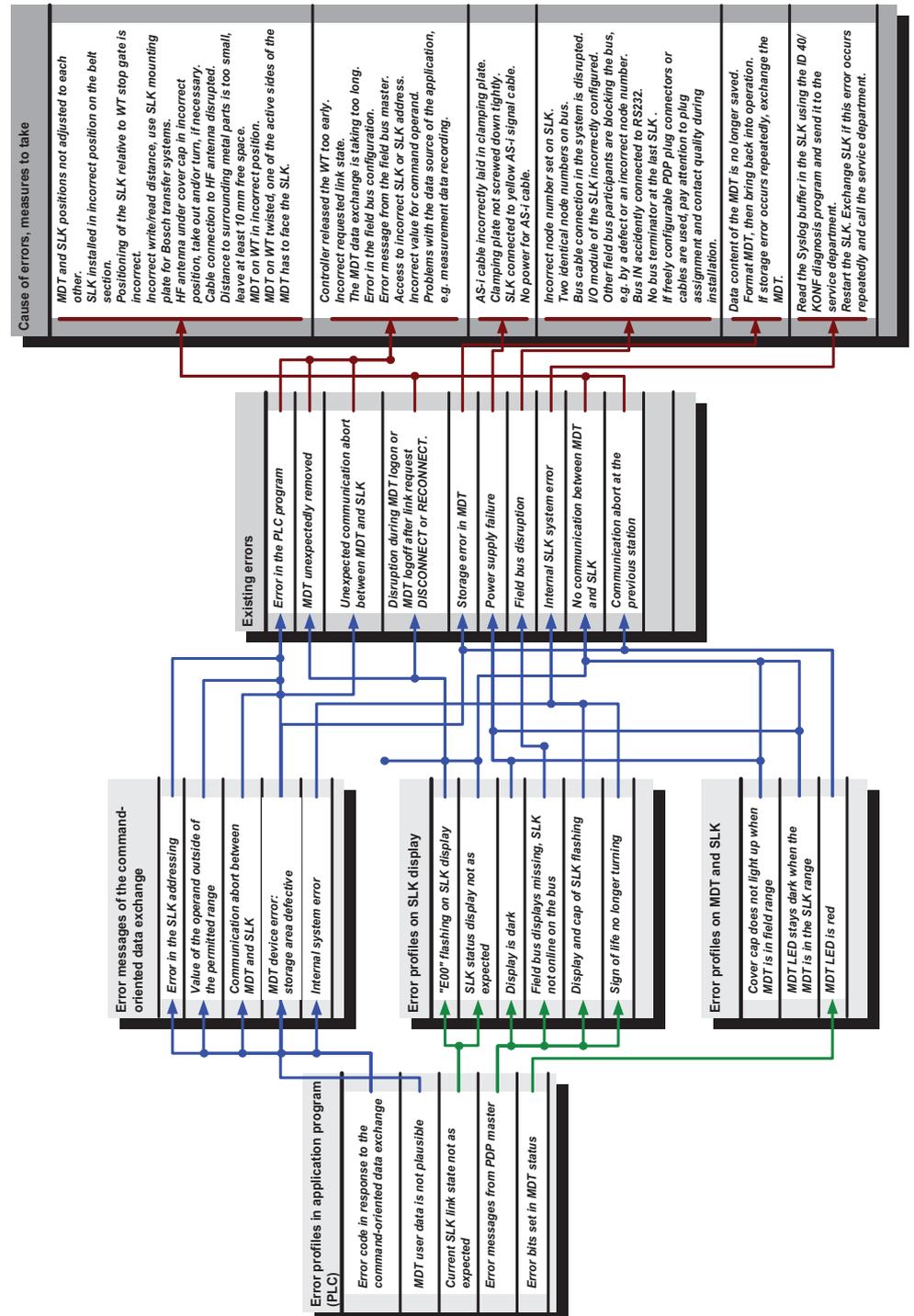


Fig. 39: Diagram of possible causes of errors

ENGLISH

12.2 Diagnostics using the web interface

The SLK logs events and errors in what is known as a syslog buffer. This data buffer can be read by a browser from **System Diagnostics** on the ID 40/SLK homepage and saved to a syslog file. See also Chapter 10.5.8 “System log and settings page” and 10.6.2 “Directly retrieving the syslog”.

Export the syslog when the ID 40 experiences a system error and the actual cause for a specific error indication is unclear.

The syslog file has to be sent to Service for further diagnostics and cannot be interpreted by the user.

12.3 SLK software upgrade via serial interface

ID 40/KONF can be used to upgrade the SLK software by transferring a file with the new software version to the SLK through the serial interface. The new operating system becomes active after a restart.

The ID 40/KONF online manual describes how to do this.

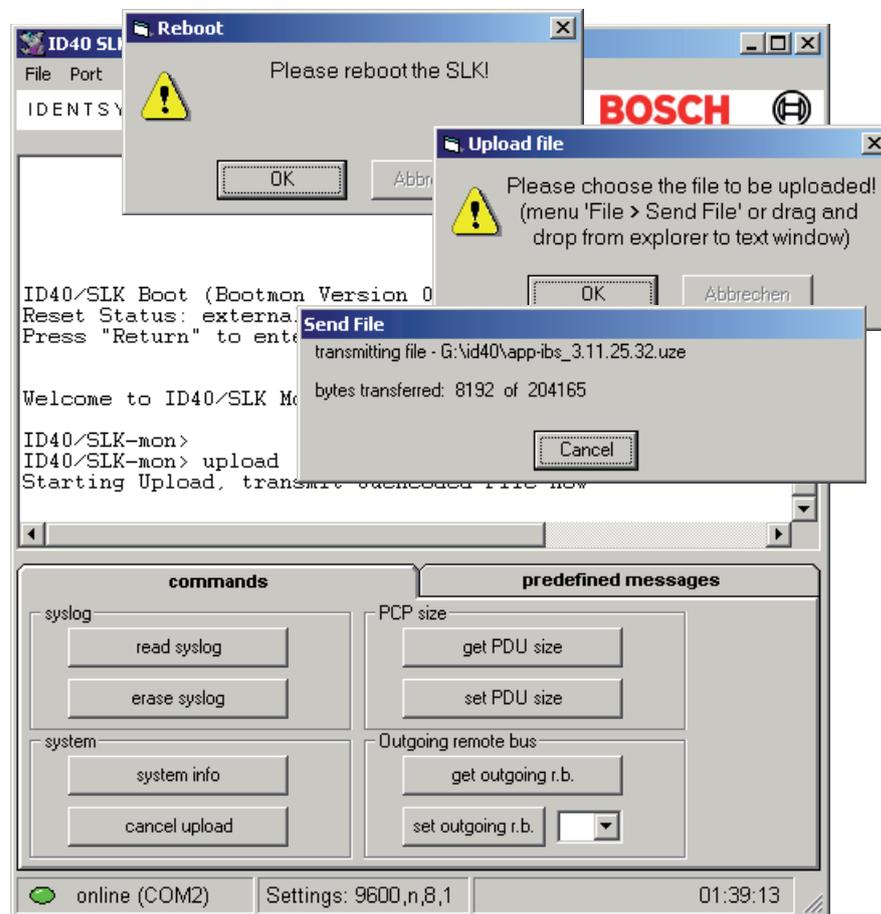


Fig. 40: SLK software upgrade with ID 40

13 Technical data

ID 40	
Type	High performance no-contact identification system
Static read/write distance	Frontal MDT alignment: 2 – 12 mm Lateral MDT alignment: 1 – 7 mm
Dynamic read/write distance	Frontal MDT alignment: 2 – 10 mm Lateral MDT alignment: 1 – 6 mm
Vmax for dynamically reading 64 bytes	30 m/min
Vmax for dynamically writing 64 bytes	30 m/min
SLK-MDT permissible height difference	± 5 mm
Certifications	ETS 300 683, EN 300 330 CE 0678 !

ID 40/SLK read/write head	
Housing dimensions	Overall dimensions: 130 x 79 x 40 mm (H x W x D)
Installation in metal	10 mm clearance on all sides required
Operating temperature	+5 °C to +55 °C
Storage temperature	-20 °C to +85 °C
Protection class	IP 65
Resistance to media	On request
Status display	Alphanumeric, 4-digit, red
Communication LED	1 color
Antenna position	90° forward, 0° upward
Connection type	Insertable M12 connections for fieldbus, IDC connector for 24 V power supply (black AS-i flat cable)
Connectable to	Profibus DP, CANopen, Interbus-S, RS232 (parameterization, diagnostics)
Power supply	24 V DC as per EN 61131-2
Current consumption	Max. 350 mA
Weight of SLK	Approx. 250 g
Mounting plate	Approx. 150 g

ID 40/MDT mobile data tag	
Type	Readable/writable memory
Storage capacity	Approx. 2 kB/8 kB/32 kB
Data organization	Byte array, address-oriented
Reading/writing cycles	Up to 10 trillion
Guaranteed data retention time	10 years (20 to 40 °C)
Housing dimensions	42 x 28 x 20 mm (H x W x D)
Installation in metal	Flush
Storage temperature	-25 °C to +85 °C
Operating temperature	-25 °C to +70 °C
Protection class	IP 68
Resistance to media	Water, mineral oil, lubricating oil, brake fluid, drilling water; more on request
Readability	Readable from 3 sides
Status LED	2 colors (red, green)
Weight	Approx. 50 g

13.1 Cables, connector pin assignments

Table 50: ID 40 cable variants

Fieldbus	Profibus DP	CANopen	Interbus	Diagnostic cable
Color	Purple	Teal or gray	Green	Gray
Certification	PNO (Profibus user organization) UL CSA	CIA (CAN in Automation user organization) UL	Interbus Club (user organization) UL	RS232 Standard
Plug	M12 (B-coded)	M12 (A-coded)	M12 (B-coded)	M12 (B-coded) 9-pin D-sub
No. pins	5	5	5	M12: 5 D-sub: 9
Wires	Data lines A (green): pin 2 B (red): pin 4 Shielding Drain wire: pin 5 Mesh: Housing	Data lines CAN-H: pin 4 CAN-L: pin 5 Voltage 24 V: pin 2 0 V: pin 3 Shielding Drain wire: pin 1 Mesh: Housing	Data lines DO: pin 1 /DO: pin 2 DI: pin 3 /DI: pin 4 GND: pin 5 Shielding Mesh: Housing	Data lines RxD: pin 1 TxD: pin 5 GND: pin 3 Shielding Housing
Material	PUR	PUR	PUR	PUR
Gauge	0.34 mm	0.32 mm	0.34 mm	0.34 mm
Max. current	6.4 A	6.4 A	6.4 A	6.4 A

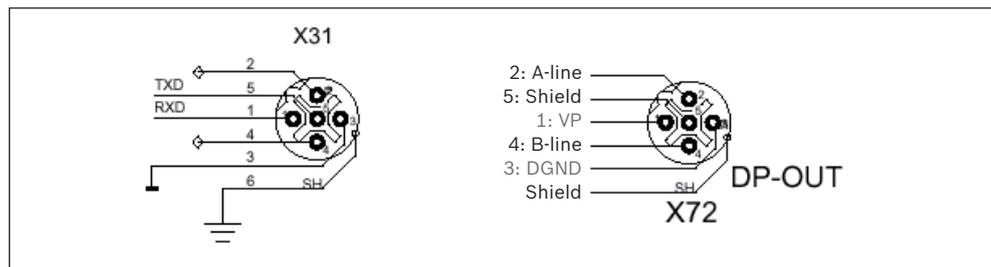


Fig. 41: RS232, M12, B-coded

PDP, M12, B-coded

13.2 SLK nameplate



Fig. 42: Nameplate for an ID 40/SLK-IBS

The most critical information is:

- CE certification mark: CE 0678 (!) for the entire ID 40/SLK family
- Device name with order number, in above example “ID 40/SLK-PDP”, “3 842 406 130”
- “Nodes”: Field for the node numbers of the SLK in the application, important for ID 40/SLK-PDP and ID 40/SLK-CAN.
- “Serial no.”: Serial number for the device.
- “Software vers.”: SLK software version

The version indicated refers to the version when the device was shipped. Since the system supports downloading operating system releases, the version printed does not necessarily reflect the current version of the operating system. However, it is ensured that it is the version when shipped from the factory.



The current software version of the device can be found on the ID 40/SLK homepage.

- Electrical values for power supply
- “Hardware vers.”: Hardware version of the internal electronics.
- “PD”: Coded production date, important for Service topics.

13.3 Data transmission times between MDT and controller

The application program reads data from the MDT via the SLK and writes data via the SLK to the MDT. Aside from the amount of data, the times required for transmitting data from the controller via the read/write head to the mobile data tag depend on the following, fieldbus-specific factors:

- **SLK function used.** If data is read using event-oriented data exchange, the controller does not need to send a read command via command-oriented data exchange
- **PLC cycle time,** which executes the read and write commands and transfers the MDT data from the fieldbus master to the application
- **Fieldbus system bitrate.**
- If the **positioning of the MDT and SLK fall within the tolerance limits,** the effective data rate between SLK and MDT can drop due to frequent repetitions
- If the system is used in **extremely low EMC environments,** the data rate between SLK and MDT can drop. Additional shielding steps may have to be taken for the SLK in such applications (e.g., functional grounding as described in the assembly instructions).

13.3.1 Transmission times with Profibus

Additional factors affecting the data transmission time with Profibus:

- **I/O module configuration from the GSD file.** A 128-byte I/O module for command-oriented data exchange has a higher data rate than, e.g., a 16-byte module for large amounts of data. In the latter case, significantly more bus cycles are required
- **Number of users on the bus.** The lower the number, the higher the data throughput of each.
- Type of additional users on the bus. Users with greater latency time also delay data access to the SLK

The **measured values** in the following table were obtained by a **Bosch PCL controller** with a PLC cycle of **10 ms**, a 128-byte I/O module for command-oriented transmission and a **12 Mbps** PDP transfer rate.

SLK and MDT communicated under standard conditions:
Frontal position, 6 mm gap, no sources of interference.

Table 51: PDP data transfer times, 10 ms PLC cycle

Byte count	8	16	32	64	256	1024	7664
Read [ms]	16	19	40	78	160	640	4740
Write [ms]	17	20	45	86	220	860	6280

The **measured values** in the following table were obtained by a Bosch PCL controller with a PLC cycle of **2 ms**. The other test parameters are the same as above.

Table 52: PDP data transfer times, 2 ms PLC cycle

Byte count	8	16	32	64	256	1024	7664
Read [ms]	14	16	24	36	145	546	3800
Write [ms]	16	20	28	44	176	690	5006

13.4 Compatibility

13.4.1 Compatibility with older software versions

Future ID 40 software versions will remain backward compatible as far as possible. However, it cannot be ruled out that modifications to the SLK/user interface will become necessary in order to enhance functionality.

13.4.2 Compatibility with ID 80/E and MTS 2

The previous Bosch ID 80/E identification system has been replaced with the ID 40. The system components are compatible with one another to ensure a seamless transition in the applications, i.e., both ID 40/MDTs and ID 80/E-MDTs can be run with the ID 40/SLK. Even the ID 80/E-SLS can communicate with all MDTs.

Nevertheless, replacing an ID 80/E-SLS with an ID 40/SLK requires the following:

- Availability of a fieldbus system supported by the ID 40. The ID 80/E-SLS cannot be connected to a fieldbus
- Existing SLS Programs have to be implemented in the control application that exchanges data with the ID 40/SLK through the fieldbus

ID 80/E-MDTs can be replaced by ID 40/MDT8Ks in **MTS 2** systems without restriction.



It is not possible to use ID 40/MDT2Ks on MTS 2 systems due to the lack of memory capacity.

14 Overview for ordering ID 40 modules

Mobile data tag		
2 kB memory capacity	ID 40/MDT2K	3 842 406 150
8 kB memory capacity → Only available from Service, replaced by the ID 40/ MDT32K	ID 40/MDT8K	3 842 406 160
32 kB memory capacity	ID 40/MDT32K	3 842 406 170
Read/write head		
CANopen version	ID 40/SLK-CAN	3 842 406 120
Profibus DP version	ID 40/SLK-PDP	3 842 406 130
Interbus version	ID 40/SLK-IBS	3 842 406 140
Fieldbus connection cable for SLK CANopen version		
M12 plug, straight - open end, l = 3 m		3 842 406 121
M12 socket, straight - open end, l = 3 m		3 842 406 122
M12 plug, angled - open end, l = 3 m		3 842 406 123
M12 socket, angled - open end, l = 3 m		3 842 406 125
M12 plug, angled - M12 socket, angled, l = 3 m		3 842 406 152
Fieldbus connection cable for SLK Profibus DP version		
M12 plug, straight - open end, l = 3 m		3 842 406 131
M12 socket, straight - open end, l = 3 m		3 842 406 132
M12 plug, angled - open end, l = 3 m		3 842 406 133
M12 socket, angled - open end, l = 3 m		3 842 406 135
M12 plug, angled - M12 socket, angled, l = 3 m		3 842 406 153
Fieldbus connection cable for SLK Interbus version		
M12 plug, straight - open end, l = 3 m		3 842 406 141
M12 socket, straight - open end, l = 3 m		3 842 406 142
M12 plug, angled - open end, l = 3 m		3 842 406 143
M12 socket, angled - open end, l = 3 m		3 842 406 145
M12 plug, angled - M12 socket, angled, l = 3 m		3 842 406 154
Fieldbus terminating resistor		
CANopen		3 842 406 155
Profibus DP		3 842 406 156
Interbus terminating bridge		3 842 409 904
Accessories		
RS232 diagnostics cable (9-pin sub-D M12 round plug)		3 842 406 117
Configuration and diagnostics software pack	ID 40/KONF	3 842 406 119
Function blocks for Siemens S7 controllers		3 842 406 190
Function blocks for Bosch CL and PCL controllers		3 842 406 191

Manuals

System manual	3 842 406 124
Manual for function blocks for Siemens S7 controllers, included in 3842406190	
Manual for function blocks for Bosch CL and PCL controllers, included in 3842406191	
ID 40/KONF manual, included in 3842406119	

Spare parts

Protective cap for HF antenna incl. screws	0 842 903 604
HF antenna	0 842 903 606
Clamping plate for power supply incl. screws	0 842 903 605
M12 cap for RS232 interface	0 842 903 607
ID 40/SLK mounting kit	3 842 527 634
ID 40/MDT mounting kit	3 842 529 237

15 Service and support

15.1 Technical support

Technical support can be reached under

- Phone: +49 (0) 711 811 - 78 62
- Fax: +49 (0) 711 811 - 78 51
- E-mail: brs.svl@boschrexroth.de

15.2 Internet

You can find general [informational material](#) on Bosch Rexroth Linear Motion and Assembly Technologies as well as details on the [ID 40 system](#) on the [Internet](#).

15.3 Site



Stuttgart Administration and Factory

Administration for Assembly Technology
Production of Basic Mechanical Elements, Manual Work Systems,
Material and Information Flow Technology

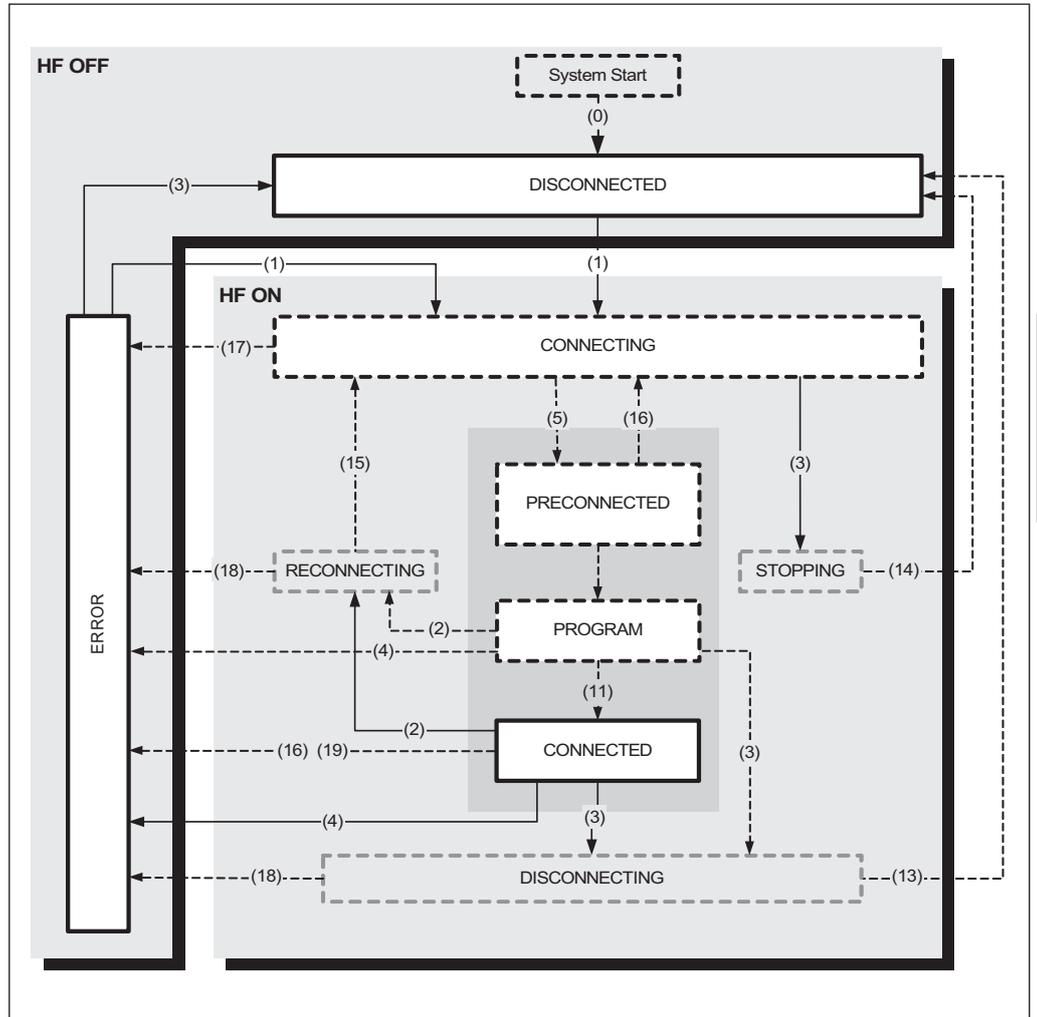
Bosch Rexroth AG

Administration and Plant
Linear Motion and Assembly Technologies
Löwentorstraße 68-70
70376 Stuttgart
Phone: +49 (0)7 11 8 11-3 06 98

[Directions to the Löwentorstraße site](#)

16 Appendix

16.1 ID 40 system link model



ENGLISH

Notes:

	Link state	Transition conditions
Solid line	Stable state only changed upon commanded link state	Command for commanded link state
Dashed line	SLK can remain in this state for an unspecified time	SLK automatically switches to another state
Dashed gray box	This state is system internal only. The actual link state shows BUSY	

Description of transition conditions

The link state is changed either automatically (auto) or by a “CLS” commanded link state. The number of the transition condition (“TC”) is shown in parentheses.

The PROGRAM state only executes the auto mode function in 3.x versions.

TC	Trigger	Initial state	Target state	Comments
(0)	Power on	SLK off	DISCONNECTED	SLK turned on
(1)	CLS	DISCONNECTED ERROR	CONNECTING	HF field turned on, system standing by for MDTs
(2)	CLS	CONNECTED PROGRAM	RECONNECTING	RECONNECTING command can come from the fieldbus or auto mode
(3)	CLS	CONNECTING CONNECTED PROGRAM ERROR	STOPPING DISCONNECTING	Switching cancels standby state both in CONNECTING and when CONNECTED ends. The final state is DISCONNECTED in both instances
(4)	CLS	CONNECTED PROGRAM	ERROR	Commanded link state ERROR executed
(5)	Auto	CONNECTING	PRECONNECTED	MDT signed on
(6)	Auto	PRECONNECTED	PRECONNECTED Pretransmit Prefetch	Pretransmits are always executed before prefetches. If neither are parameterized, PRECONNECTED contains no action
(7)	Auto	PRECONNECTED	PROGRAM	PRECONNECTED phase completed without error
(8)				
(9)				
(10)				
(11)	Auto	PROGRAM	CONNECTED	CONNECTED active by default when no user program loaded or auto mode off
(12)				
(13)	Auto	DISCONNECTING	DISCONNECTED	Communication channel to MDT closed, HF field turned off
(14)	Auto	STOPPING	DISCONNECTED	No longer standing by for MDTs, HF field turned off
(15)	Auto	RECONNECTING	CONNECTING	Communication channel to MDT closed, SLK standing by for new MDT sign-on
(16)	Auto	PRECONNECTED CONNECTED	CONNECTING	Unexpected power-ups received
(17)	Auto	CONNECTING	ERROR	Error during MDT sign-on
(18)	Auto	RECONNECTING DISCONNECTING	ERROR	Error during MDT sign-off
(19)	Auto	CONNECTED	ERROR	MDT removed unexpectedly, MDT lifeguarding automatically switches to ERROR state

16.2 Abbreviations and terms

Technical term	Description
Sign-off	Terminates MDT communication either with the commands DISCONNECT/RECONNECT or the appropriate auto mode.
Actual link state	Shows the current link state between SLK and MDT.
Commanded link state	Command to the SLK to change the system link state, e.g., from the <i>CONNECTED</i> state phase to close the MDT.
Auto mode	Automatically terminates the data transmission phase with >auto reconnect or >auto disconnect.
Auto disconnect	Automatically terminates the data transmission phase and turns off the >HF field.
Auto reconnect	Automatically terminates the data transmission phase, but the HF field remains on.
Buffered prefetch	Data is read and stored in internal buffers.
Byte array	Data array with a width of 8 bits and variable size.
CAN	Abbreviation for the CANopen fieldbus system that offers standardized payload transmission services.
COB ID	Communication Object Identifier. Unique ID of a data telegram in CANopen.
<i>CONNECTED</i> state	Direct MDT access via the fieldbus. The <i>CONNECTED</i> state has to be terminated by a bus command.
Data block	A data block consists of the SLK address and a byte count. The SLK address is the start address, and the byte count determines the area length from the start address. It is written as address/byte count.
Unbuffered prefetch	Data is read and transmitted without buffers via >event-oriented data exchange.
Event-oriented data exchange	Cyclical data transmission from the SLK to the PLC.
Emergency shell	Active shell after fatal system error.
Fieldbus master	User on a bus system with command authority over subordinate users (>slaves).
Fieldbus slave	User on a bus system without command authority.
Formatting value	Entered in the MDT memory during MDT formatting.
HF field	Electromagnetic high frequency field. Allows communication between MDT and SLK without contact.
High byte	High byte of a 16-bit value.
IBS	Abbreviation for the Interbus open fieldbus system according to standard EN 50254.
ID 40/SLK-CAN	Product designation for the SLK of the ID 40 system for connecting to CANopen.
ID 40/SLK-IBS	Product designation for the SLK of the ID 40 system for connecting to Interbus.
ID 40/SLK-PDP	Product designation for the SLK of the ID 40 system for connecting to Profibus.
ID 80/E	Product designation for the forerunner system of the ID 40. The MDTs in both identification systems are compatible.
Node number	The SLK can be reached in the fieldbus system under this number. The node number is configured via the serial interface. The IBS does not require node numbers. Do not confuse with the SLK address!
Command-oriented data exchange	Data transmission option independent of the cyclical functioning of fieldbus systems.
Console	Shell
Low byte	Low byte of a 16-bit value.

Technical term	Description
Master	See >Fieldbus master
MDT Mobile Data Tag	Electronic memory. The MDT is fixed to the workpiece pallet and contains the current information on the workpieces on the workpiece pallet.
MDT address	Located in the first segment of the SLK address area.
MDT formatting	Fills an MDT memory area with a constant formatting value
MDT ID code	32-bit number that has a different value in each MDT and cannot be edited. This allows every MDT to be uniquely identified.
MDT status	Information on MDT type and current MDT state.
MDT type code	3-bit coding of the MDT type contained in the MDT status. The MDT type code provides information on the MDT memory capacity.
Multiple transmit	Pretransmit executed for all MDTs.
Network address	IP address, part of the web address.
Emergency console	Emergency shell
PDP	Abbreviation for the Profibus DP fieldbus system that supports cyclical data communication in accordance with EN 50170.
Prefetch	Automatic read access to the MDT. Prefetch has to be configured in advance. Prefetch always occurs before the <i>CONNECTED</i> state.
Pretransmit	Automatic write access to the MDT. Pretransmit data has to be entered in the SLK in advance. Pretransmit occurs as soon as the MDT is linked and always before prefetch.
RS232 interface	Serial interface for the SLK for parameterization and extended diagnostics as per RS232 specification.
RPDO	R eceive P rocess D ata O bject: event-oriented CANopen communication service that transmits data from the bus master to the SLK.
Segment	Division of the SLK memory area into individual segments that are 64 kB in size.
Shell	Command line-based input option via the serial interface using a terminal program or ID 40/KONF.
Single transmit	Pretransmit only executed on the next MDT.
Slave	See >Fieldbus slave
SLK Schreib-Lese-Kopf	Fieldbus slave with HF communication unit for non-stationary, no-contact communication between user controller and MDT.
SLK address	Required to read and write data via the fieldbus.
Syslog buffer	Buffer for recording system internal events.
Toggle bit	Bit for controlling command-oriented data exchange.
TPDO	T ransmit P rocess D ata O bject: event-oriented CANopen communication service that transmits data from the SLK to the bus master.
Dwell time	Time in which the MDT is in the SLK transmission area.
Web browser	Internet browser or simply "browser", interactive program for displaying websites.
Web server	Functional component in the ID 40/SLK, contains the ID 40/SLK website.
Website	The entire online presence of the SLK.
Web address	URL, Uniform Resource Locator. Addresses the SLK website in the browser. Contains the network address for the SLK.

16.3 References

- [P1] Profibus Standard EN 50170
- [P2] [Profibus User Organization e.V.](#)
Profibus Technical Description, Order No. 4.001, September 1999
- [I1] [Phoenix Contact](#)
Interbus user manual, General Introduction to the Interbus System
Designation: IBS SYS INTRO G4 UM
Revision: A
Order No.: 27 45 10 1
- [2] Phoenix Contact
Interbus user manual for the Peripherals Communication Protocol (PCP)
Description: IBS SYS PCP G4 UM
Revision: B
Order No.: 27 45 11 4
- [C1] CiA/DS 102, CAN Physical Layer for Industrial Applications
- [C2] Robert Bosch GmbH, CAN Specification 2.0 Part B, September 1991
- [C3] ISO 11898, November 1993, Road Vehicles, Interchange of Digital Information – Controller Area Network (CAN) for High-speed Communication.
- [C4] CiA/DS 201, CAN Reference Model, February 1996
- [C5] CiA/DS 202-1, CMS Service Specification, February 1996
- [C6] CiA/DS 202-2, CMS Protocol Specification, February 1996
- [C7] CiA/DS 202-3, CMS Encoding Rules, February 1996
- [C8] CiA/DS 203-1, NMT Service Specification, February 1996
- [C9] CiA/DS 203-2, NMT Protocol Specification, February 1996
- [C10] CiA/DS 204-1, DBT Service Specification, February 1996
- [C11] CiA/DS 204-2, DBT Protocol Specification, February 1996
- [C12] CiA/DS 205-1, LMT Service Specification, February 1996
- [C13] CiA/DS 205-2, LMT Protocol Specification, February 1996
- [C14] CiA/DS 206, Application Specific Data Types, February 1996
- [C15] CiA/DS 207, Application Layer Naming Specification, Feb. 1996
- [C16] CiA/DS 301, CAL-based Communication Profile, Oct. 1996
- [C17] CiA/DS 301 V4.0, Application Layer and Communication Profile, June 1999

Bosch Rexroth AG

Postfach 30 02 07

D-70442 Stuttgart

Germany

Fax +49 (0) 711 811-7777

info@boschrexroth.de

www.boschrexroth.com