BECKHOFF New Automation Technology

Documentation | EN

CX1500-M510, CX1500-B510

CANopen - Bus interfaces for CX systems

2006/6/6 | Version: 1.0

Table of contents

1 Foreword

1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with the applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without prior announcement.
No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, EtherCAT®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC® and XTS® are registered trademarks of and licensed by Beckhoff Automation GmbH.
Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:
EP1590927, EP1789857, DE102004044764, DE102007017835
with corresponding applications or registrations in various other countries.

The TwinCAT Technology is covered, including but not limited to the following patent applications and patents:
EP0851348, US6167425 with corresponding applications or registrations in various other countries.

EtherCAT® is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

1.2 Safety instructions

Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

⚠ DANGER

Serious risk of injury!

Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.

⚠ WARNING

Risk of injury!

Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.

⚠ CAUTION

Personal injuries!

Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.

NOTE

Damage to the environment or devices

Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.

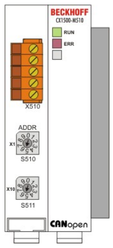
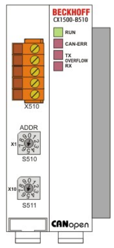
Tip or pointer

This symbol indicates information that contributes to better understanding.

1.3 Documentation Issue Status

Version	Changes
1.0	revised version
0.1	preliminary version

2 Product overview

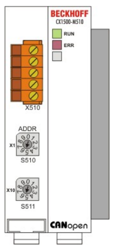
 

The connection to the Profibus for the CX-family is realized by field assembles modules. There are two versions of modules:

- Master connection [▶ 8]
- Slave connection [▶ 10]

The function is similar to the function of the Beckhoff PCI fieldbus cards. In opposite to them only one port is available in the modules for CX-Systems. The connection parameters will be set by using TwinCAT. The data transfer to the system is realized by a DPRAM via PC104 bus.

2.1 CX1500-M510 (Master)



Fieldbus connection enable the distributed collection of process data and signals, even for distant machines or equipment. The use of fieldbus master modules in a CX10x0 system enables the utilization of all Beckhoff fieldbus components (e. g. Bus Coupler, Bus Terminal Controller, drive technology) as distributed control components for the assembly of complex systems.

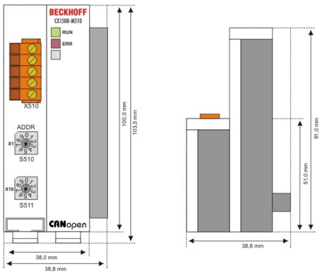
Parallel operation of several identical or different master connections is also possible, Mixed operation of master and slave connections is also not a problem. A CX system can thus also assume the functionality of an intelligent gateway between different fieldbuses, receiving data from a fieldbus, processing them via a program and then feeding them into another fieldbus.

The performance data of the CX fieldbus master modules are nearly identical to those of the Beckhoff PC fieldbus cards, except for the fact that the CX variants are always single-channel types. The number of slaves that can be connected is only limited by the respective bus system. The use of master or slave connections enables networking of several CX systems with each other via the fieldbus level. In contrast to networking via Ethernet, strictly deterministic data transmission can be achieved in this case.

CX fieldbus modules can be upgraded or exchanged in the field and can be connected to an existing CX system via the PC104 system bus. The power supply of the fieldbus connections is also ensured via the PC104 bus. Despite the ISA bus, no address switch is required for the address setting, since the modules are assigned an address via internal setup. (The address must be selected by ordering the module)

The software integration of the fieldbus connections into the TwinCAT automation software is done in the usual comfortable way: Scanning and detection of the modules, parameterization, configuration of the connected I/O components as well as online diagnosis of the process and fieldbus status are carried out in the familiar way through the TwinCAT System Manager.

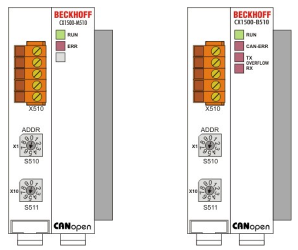
2.1.1 Technical data CX1500-M510



Technical data	CX1500-B310
Fieldbus	CANopen
Transmission rate	10, 20, 50, 100, 125, 250, 500, 800, 1000 kBaud
Bus connection	Open-Style-Connector, 5-polig
Bus nodes	max. 127 slaves
Max. I/O-size	1536 Bytes Input / 1536 Byte Output
Interface to the CPU	16 bit ISA (PC104-Standard) / 2 kbyte DPRAM
Max. power consumption	1,8 W
Dimensions	38 mm x 100 mm x 91 mm
Weight	190 g
Operating temperature	0 °C ... +55 °C
Storage temperature	-25 °C ... +85 °C
Relative humidity	95% no condensation
Vibration/shock resistance	confirms to EN 60068-2-6 / EN 60068-2-27/29
EMC resistance burst /ESD	confirms to EN 61000-6-2 / EN 61000-6-4
Protection class	IP 20

2.1.2 Connections CX1500-M510 / B510

The connection to the CANopen bus is realized by a 5 pin open style connector.



The pin of the connector have the following definition:

Pin	Meaning
1	reserved
2	CAN - High
3	Shield
4	CAN - Low
5	Ground

More details to the connection are described in the chapter bus description.

2.1.3 Adapter RAM Hardware address overview

available memory addresses for CX1000: D0000-EFFFF (hex)

Base Address (hex)	End Address (hex)	Size(Bytes)(hex)	Access Type	Description
D6000	D7FFF	2000	R/W	CX1500-M510 CANopen Master DPRAM
E6000	E7FFF	2000	R/W	CX1500-B510 CANopen Slave DPRAM

available memory addresses for CX1020: D0000-DFFFF (hex)

Base Address (hex)	End Address (hex)	Size(Bytes)(hex)	Access Type	Description
D6000	D7FFF	2000	R/W	CX1500-M510 CANopen Master DPRAM

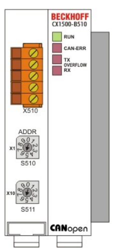
For CX1020 the memory space upper DFFFF is reserved for the BIOS and other functions.

For some fieldbus connections (all Slave modules) the base addresses are mapped in the memory region upper DFFFF(hex). So this modules must be ordered with other base addresses. The same situation takes place if more than two or more master modules of same type are used (for more see note below). The order numbers for the modules are:

Order number	Alternative ISA-Address
CX1500-M510-0001	D4000
CX1500-M510-0002	D6000
CX1500-M510-0003	D8000
CX1500-M510-0004	DA000
CX1500-M510-0005	DC000

Two connection modules (master or slave) can be used simultaneously. If more than two connections are needed call Beckhoff Automation GmbH for further information.

2.2 CX1500-B510 (Slave)



Fieldbus connection enable the distributed collection of process data and signals, even for distant machines or equipment. The use of fieldbus master modules in a CX1000 / CX1020 system enables the utilization of all Beckhoff fieldbus components (e. g. Bus Coupler, Bus Terminal Controller, drive technology) as distributed control components for the assembly of complex systems.

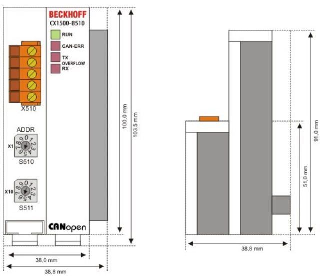
Parallel operation of several identical or different master connections is also possible, Mixed operation of master and slave connections is also not a problem. A CX system can thus also assume the functionality of an intelligent gateway between different fieldbuses, receiving data from a fieldbus, processing them via a program and then feeding them into another fieldbus.

The performance data of the CX fieldbus master modules are nearly identical to those of the Beckhoff PC fieldbus cards, except for the fact that the CX variants are always single-channel types. The number of slaves that can be connected is only limited by the respective bus system. The use of master or slave connections enables networking of several CX systems with each other via the fieldbus level. In contrast to networking via Ethernet, strictly deterministic data transmission can be achieved in this case.

CX fieldbus modules can be upgraded or exchanged in the field and can be connected to an existing CX system via the PC104 system bus. The power supply of the fieldbus connections is also ensured via the PC104 bus. Despite the ISA bus, no address switch is required for the address setting, since the modules are assigned an address via internal setup. (The address must be selected by ordering the module)

The software integration of the fieldbus connections into the TwinCAT automation software is done in the usual comfortable way: Scanning and detection of the modules, parameterization, configuration of the connected I/O components as well as online diagnosis of the process and fieldbus status are carried out in the familiar way through the TwinCAT System Manager.

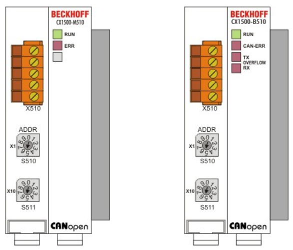
2.2.1 Technical data CX1500-B510



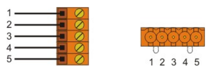
Technical data	CX1500-B310
Fieldbus	CANopen
Transmission rate	10, 20, 50, 100, 125, 250, 500, 800, 1000 kBaud
Bus connection	Open-Style-Connector, 5-polig
Bus nodes	max. 127 slaves
Max. I/O-size	1536 Bytes Input / 1536 Byte Output
Interface to the CPU	16 bit ISA (PC104-Standard) / 2 kbyte DPRAM
Max. power consumption	1,8 W
Dimensions	38 mm x 100 mm x 91 mm
Weight	190 g
Operating temperature	0 °C ... +55 °C
Storage temperature	-25 °C ... +85 °C
Relative humidity	95% no condensation
Vibration/shock resistance	confirms to EN 60068-2-6 / EN 60068-2-27/29
EMC resistance burst /ESD	confirms to EN 61000-6-2 / EN 61000-6-4
Protection class	IP 20

2.2.2 Connections CX1500-M510 / B510

The connection to the CANopen bus is realized by a 5 pin open style connector.



The pin of the connector have the following definition:



Pin	Meaning
1	reserved
2	CAN - High
3	Shield
4	CAN - Low
5	Ground

More details to the connection are described in the chapter bus description.

2.2.3 Adapter RAM Hardware Address overview

available memory addresses for CX1000: D0000-EFFFF (hex)

Base Address (hex)	End Address (hex)	Size (Bytes)(hex)	Access Type	Description
D6000	D7FFF	2000	R/W	CX1500-M510 CANopen DPRAM
E6000	E7FFF	2000	R/W	CX1500-B510 CANopen Slave DPRAM

available memory addresses for CX1020: D0000-DFFFF (hex)

Base Address (hex)	End Address (hex)	Size (Bytes)(hex)	Access Type	Description
D6000	D7FFF	2000	R/W	CX1500-B510 CANopen Master DPRAM

For some fieldbus connections (all Slave modules) the base addresses are mapped in the memory region upper DFFFF(hex). So this modules must be ordered with other base addresses. The same situation takes place if more than two or more master modules of same type are used (for more see note below). The order numbers for the modules are:

Order number	Alternative ISA-Address
CX1500-B510-0001	D4000
CX1500-B510-0002	D6000
CX1500-B510-0003	D8000
CX1500-B510-0004	DA000
CX1500-B510-0005	DC000

Two connection modules (master or slave) can be used simultaneously. If more than two connections are needed call Beckhoff Automation GmbH for further information.

2.3 CANopen

2.3.1 CANopen Introduction

Fig. 1: CANopenLogo

CANopen is a widely used CAN application layer, developed by the CAN in Automation association (CiA, http://www.can-cia.org), which has meanwhile been adopted for international standardization.

Device Model

CANopen consists of the protocol definitions (communication profile) and of the device profiles that standardize the data contents for the various device classes. Process data objects (PDO) [▶ 15] are used for fast communication of input and output data. The CANopen device parameters and process data are stored in a structured object directory. Any data in this object directory is accessed via service data objects (SDO). There are, additionally, a few special objects (such as telegram types) for network management (NMT), synchronization, error messages and so on.

Communication Types

CANopen defines a number of communication classes for the input and output data (process data objects):

- Event driven [▶ 15]: Telegrams are sent as soon as their contents have changed. This means that the process image as a whole is not continuously transmitted, only its changes.
- Cyclic synchronous [▶ 15]: A SYNC telegram causes the modules to accept the output data that was previously received, and to send new input data.
- Requested [▶ 15]: A CAN data request telegram causes the modules to send their input data.

The desired communication type is set by the Transmission Type [▶ 15] parameter.

Device Profile

The BECKHOFF CANopen devices support all types of I/O communication, and correspond to the device profile for digital and analog input/output modules (DS401 Version 1). For reasons of backwards compatibility, the default mapping was not adapted to the DS401 V2 profile version.

Transmission Rates

Transmission Rates [▶ 14]

Nine transmission rates from 10 kbaud up to 1 Mbaud are available for different bus lengths. The effective utilization of the bus bandwidth allows CANopen to achieve short system reaction times at relatively low data rates.

Topology

Topology

CAN is based on a linear topology. The number of devices participating in each network is logically limited by CANopen to 128, but physically the present generation of drivers allows up to 64 nodes in one network segment. The maximum possible size of the network for any particular data rate is limited by the signal transit time required on the bus medium. For 1 Mbaud, for instance, the network may extend 25 m, whereas at 50 kbaud the network may reach up to 1000 m. At low data rates the size of the network can be increased by repeaters, which also allow the construction of tree structures.

Bus access procedures

CAN utilizes the Carrier Sense Multiple Access (CSMA) procedure, i.e. all participating devices have the same right of access to the bus and may access it as soon as it is free (multi-master bus access). The exchange of messages is thus not device-oriented but message-oriented. This means that every message is unambiguously marked with a prioritized identifier. In order to avoid collisions on the bus when messages are sent by different devices, a bit-wise bus arbitration is carried out at the start of the data transmission. The bus arbitration assigns bus bandwidth to the messages in the sequence of their priority. At the end of the arbitration phase only one bus device occupies the bus, collisions are avoided and the bandwidth is optimally exploited.

Configuration and parameterization

The TwinCAT System Manager allows all the CANopen parameters to be set conveniently. An "EDS" file (an electronic data sheet) is available on the BECKHOFF website (http://www.beckhoff.com) for the parameterization of BECKHOFF CANopen devices using configuration tools from other manufacturers.

Certification

The BECKHOFF CANopen devices have a powerful implementation of the protocol, and are certified by the CAN in Automation Association (http://www.can-cia.org).

2.3.2 CANopen Baud Rate and Bit Timing

Bit Timing

The following baud rates and entries in the bit-timing register are supported by the CANopen devices:

Baud rate [kBaud]	BTR0	BTR1	Sampling Point
1000	0x00	0x14	75%
800	0x00	0x16	80%
500	0x00	0x1C	87%
250	0x01	0x1C	87%
125	0x03	0x1C	87%
100	0x04	0x1C	87%
50	0x09	0x1C	87%
20	0x18	0x1C	87%
10	0x31	0x1C	87%

The bit-timing register settings given (BTR0, BTR1) apply, for example, for the Philips 82C200, SJA1000, Intel 80C527, Siemens 80C167 and other CAN controllers. They are optimized for the maximum bus length.

2.3.3 Process Data Objects (PDO)

Introduction

In many fieldbus systems the entire process image is continuously transferred - usually in a more or less cyclic manner. CANopen is not limited to this communication principle, since the multi-master bus access protocol allows CAN to offer other methods. Under CANopen the process data is not transferred in a master/slave procedure, but follows instead the producer-consumer model. In this model, a bus node transmits its data, as a producer, on its own accord. This might, for example, be triggered by an event. All the other nodes listen, and use the identifier to decide whether they are interested in this telegram, and handle it accordingly. These are the consumers.

The process data in CANopen is divided into segments with a maximum of 8 bytes. These segments are known as process data objects (PDOs). The PDOs each correspond to a CAN telegram, whose specific CAN identifier is used to allocate them and to determine their priority. Receive PDOs (RxPDOs) and transmit PDOs (TxPDOs) are distinguished, the name being chosen from the point of view of the device: an input/output module sends its input data with TxPDOs and receives its output data in the RxPDOs. **This naming convention is retained in the TwinCAT System Manager.**

Communication parameters

The PDOs can be given different communication parameters according to the requirements of the application. Like all the CANopen parameters, these are also available in the device's object directory, and can be accessed by means of the service data objects. The parameters for the receive PDOs are at index 0x1400 (RxPDO1) onwards. There can be up to 512 RxPDOs (ranging up to index 0x15FF). In the same way, the entries for the transmit PDOs are located from index 0x1800 (TxPDO1) to 0x19FF (TxPDO512).

The BECKHOFF Bus Couplers or Fieldbus Coupler Box modules make 16 RxPDO and TxPDOs available for the exchange of process data (although the figure for Economy and LowCost BK5110 and LC5100 Couplers and the Fieldbus Boxes is 5 PDOs each, since these devices manage a lower quantity of process data). The FC510x CANopen master card supports up to 192 transmit and 192 receive PDOs for each channel - although this is restricted by the size of the DPRAM. Up to 32 TxPDOs and 32 RxPDOs can be handled in slave mode.

For each existing process data object there is an associated communication parameter object. The TwinCAT System Manager automatically assigns the set parameters to the relevant object directory entries. These entries and their significance for the communication of process data are explained below.

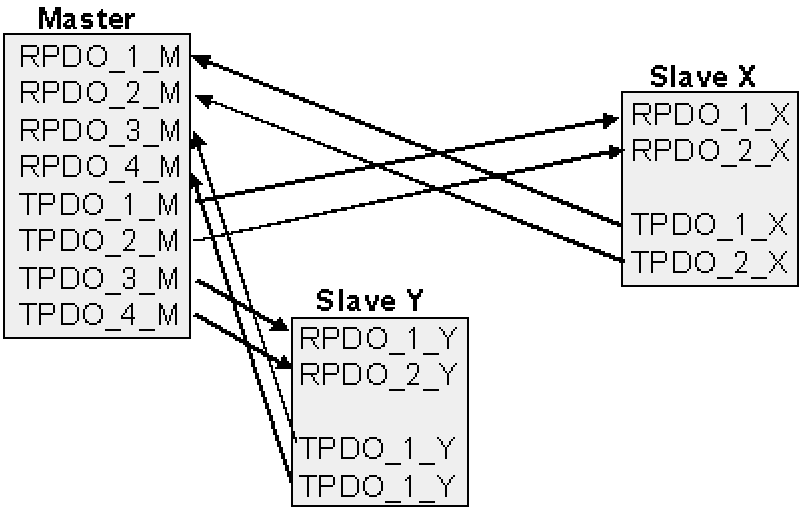
PDO Identifier

The most important communication parameter in a PDO is the CAN identifier (also know as the communication object identifier, or COB-ID). It is used to identify the data, and determines their priority for bus access. For each CAN data telegram there may only be one sender node (producer), although all messages sent in the CAN broadcast procedure can be received, as described, by any number of nodes (consumers). Thus a node can make its input information available to a number of bus devices at the same time - even without transferring them through a logical bus master. The identifier is located in sub-index 1 of the communication parameter set. It is coded as a 32-bit value in which the least significant 11 bits (bits 0...10) contain the identifier itself. The data width of the object of 32 bits also allows 29-bit identifiers in accordance with CAN 2.0B to be entered, although the default identifiers [▶ 110] always refer to the more usual 11-bit versions. Generally speaking, CANopen is economical it its use of the available identifiers, so that the use of the 29-bit versions remains limited to unusual applications. It is therefore also not supported by a Beckhoff's CANopen devices. The highest bit (bit 31) can be used to activate the process data object or to turn it off.

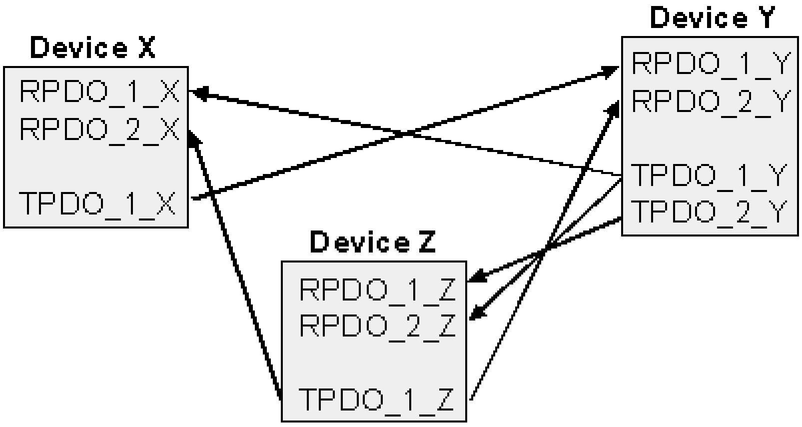
A complete identifier list [▶ 65] is provided in the appendix.

PDO linking

In the system of default identifiers, all the nodes (here: slaves) communicate with one central station (the master), since slave nodes do not listen by default to the transmit identifier of any other slave node.



Default identifier allocation: Master/Slave

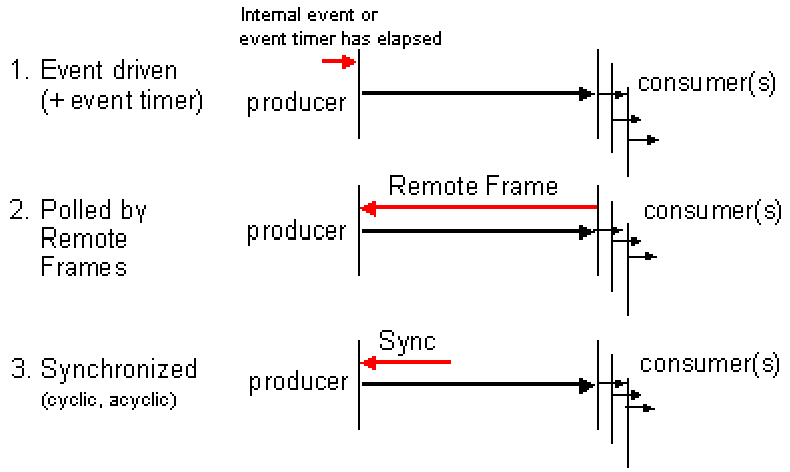


PDO linking: Peer to Peer

If the consumer-producer model of CANopen PDOs is to be used for direct data exchange between nodes (without a master), the identifier allocation must be appropriately adapted, so that the TxPDO identifier of the producer agrees with the RxPDO identifier of the consumer: This procedure is known as PDO linking. It permits, for example, easy construction of electronic drives in which several slave axes simultaneously listen to the actual value in the master axis TxPDO.

PDO Communication Types: Outline

CANopen offers a number of possible ways to transmit process data (see also: Notes on PDO Parameterization [▶ 104]).)



.

Event driven

The "event" is the alteration of an input value, the data being transmitted immediately after this change. The event-driven flow can make optimal use of the bus bandwidth, since instead of the whole process image it is only the changes in it that are transmitted. A short reaction time is achieved at the same time, since when an input value changes it is not necessary to wait for the next interrogation from a master.

As from CANopen Version 4 it is possible to combine the event driven type of communication with a cyclic update. Even if an event has not just occurred, event driven TxPDOs are sent after the event timer has elapsed. If an event does occur, the event timer is reset. For RxPDOs the event timer is used as a watchdog in order to monitor the arrival of event driven PDOs . If a PDO does not arrive within a set period of time, the bus node adopts the error state.

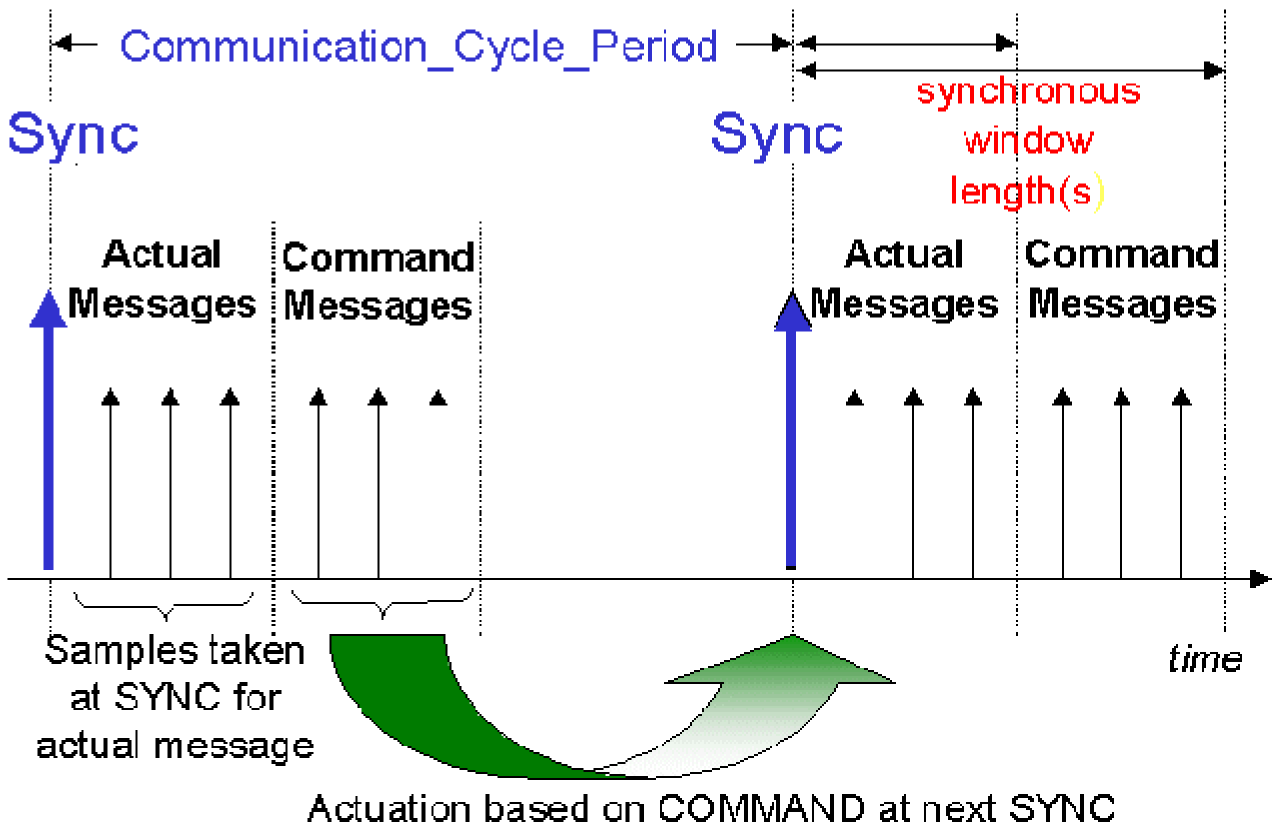
Polled

The PDOs can also be polled by data request telegrams (remote frames). In this way it is possible to get the input process image of event-driven inputs onto the bus, even when they do not change, for instance through a monitoring or diagnostic device brought into the network while it is running. The time behavior of remote frame and answer telegrams depends on what CAN controller is in use (Fig. 8). Components with full integrated message filtering ("FullCAN") usually answer a data request telegram immediately, transmitting data that is waiting in the appropriate transmit buffer - it is the responsibility of the application to see that the data there is continuously updated. CAN controllers with simple message filtering (BasicCAN) on the other hand pass the request on to the application which can now compose the telegram with the latest data. This does take longer, but does mean that the data is up-to-date. BECKHOFF use CAN controllers following the principle of Basic CAN.

Since this device behavior is usually not transparent to the user, and because there are CAN controllers still in use that do not support remote frames at all, polled communication can only with reservation be recommended for operative running.

Synchronized

It is not only for drive applications that it is worthwhile to synchronize the determination of the input information and the setting the outputs. For this purpose CANopen provides the SYNC object, a CAN telegram of high priority but containing no user data, whose reception is used by the synchronized nodes as a trigger for reading the inputs or for setting the outputs.



PDO transmission types: Parameterisation

The PDO transmission type parameter specifies how the transmission of the PDO is triggered, or how received PDOs are handled.

Transmission type	Cyclical	Acyclical	Synchronous	Asynchronous	Only RTR
0		X	X		
1-240	X		X		
241-251	- reserved -				
252			X		X
253				X	X
254, 255				X	

The type of transmission is parameterized for RxPDOs in the objects at 0x1400ff, sub-index 2, and for TxPDOs in the objects at 0x1800ff, sub-index 2.

Acyclic Synchronous

PDOs of transmission type 0 function synchronously, but not cyclically. An RxPDO is only evaluated after the next SYNC telegram has been received. In this way, for instance, axis groups can be given new target positions one after another, but these positions only become valid at the next SYNC - without the need to be constantly outputting reference points. A device whose TxPDO is configured for transmission type 0 acquires its input data when it receives the SYNC (synchronous process image) and then transmits it if the data correspond to an event (such as a change in input) having occurred. Transmission type 0 thus combines transmission for reasons that are event driven with a time for transmission (and, as far as possible, sampling) and processing given by the reception of "SYNC".

Cyclic Synchronous

In transmission types 1-240 the PDO is transmitted cyclically: after every "nth" SYNC (n = 1...240). Since transmission types can be combined on a device as well as in the network, it is possible, for example, for a fast cycle to be agreed for digital inputs (n = 1), whereas the data for analog inputs is transmitted in a slower cycle (e.g. n = 10). RxPDOs do not generally distinguish between transmission types 0...240: a PDO that has been received is set to valid when the next SYNC is received. The cycle time (SYNC rate) can be monitored (object 0x1006), so that if the SYNC fails the device reacts in accordance with the definition in the device profile, and switches, for example, its outputs into the fault state.

The FC510x card provides full support for the synchronous type of communication: transmitting the SYNC telegram is coupled to the linked task, so that new input data is available every time the task begins. The card will recognize the absence of a synchronous PDO, and will report it to the application.

Only RTR

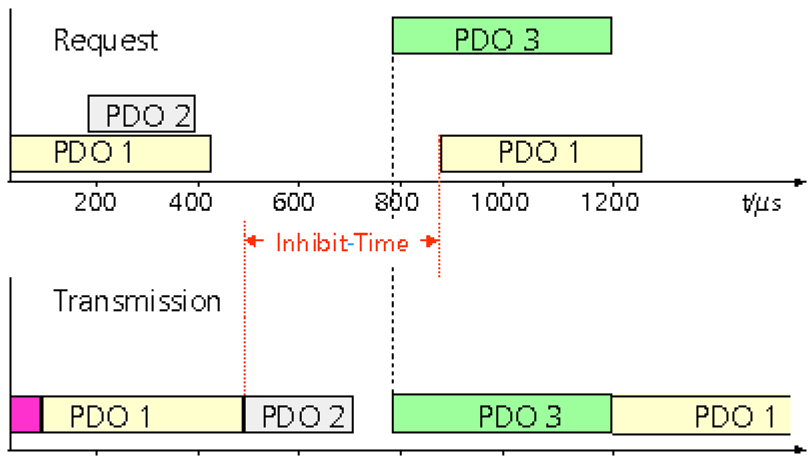
Transmission types 252 and 253 apply to process data objects that are transmitted exclusively on request by a remote frame. 252 is synchronous: when the SYNC is received the process data is acquired. It is only transmitted on request. 253 is asynchronous. The data here is acquired continuously, and transmitted on request. This type of transmission is not generally recommended, because fetching input data from some CAN controllers is only partially supported. Because, furthermore, the CAN controllers sometimes answer remote frames automatically (without first requesting up-to-date input data), there are circumstances in which it is questionable whether the polled data is up-to-date. Transmission types 252 and 253 are for this reason not supported by the BECKHOFF PC cards.

Asynchronous

The transmission types 254 + 255 are asynchronous, but may also be event-driven. In transmission type 254, the event is specific to the manufacturer, whereas for type 255 it is defined in the device profile. In the simplest case, the event is the change of an input value - this means that every change in the value is transmitted. The asynchronous transmission type can be coupled with the event timer, thus also providing input data when no event has just occurred.

Inhibit time

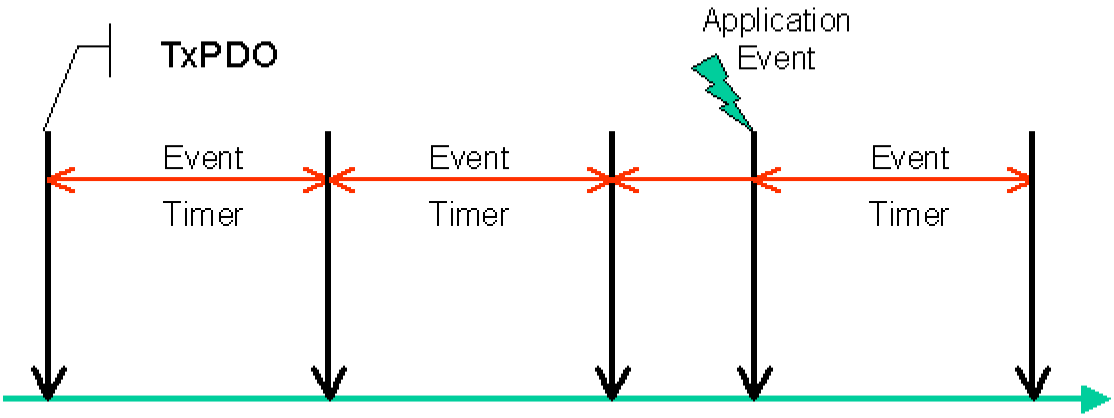
The "inhibit time" parameter can be used to implement a "transmit filter" that does not increase the reaction time for relatively new input alterations, but is active for changes that follow immediately afterwards. The inhibit time (transmit delay time) specifies the minimum length of time that must be allowed to elapse between the transmission of two of the same telegrams. If the inhibit time is used, the maximum bus loading can be determined, so that the worst case latency can then be found.



Although the BECKHOFF FC510x PC cards can parameterize the inhibit time on slave devices, they do not themselves support it. The transmitted PDOs become automatically spread out (transmit delay) as a result of the selected PLC cycle time - and there is little value in having the PLC run faster than the bus bandwidth permits. The bus loading, furthermore, can be significantly affected by the synchronous communication.

Event Timer

An event timer for transmit PDOs can be specified by sub-index 5 in the communication parameters. Expiry of this timer is treated as an additional event for the corresponding PDO, so that the PDO will then be transmitted. If the application event occurs during a timer period, it will also be transmitted, and the timer is reset.



In the case of receive PDOs, the timer is used to set a watchdog interval for the PDO: the application is informed if no corresponding PDO has been received within the set period. The FC510x can in this way monitor each individual PDO.

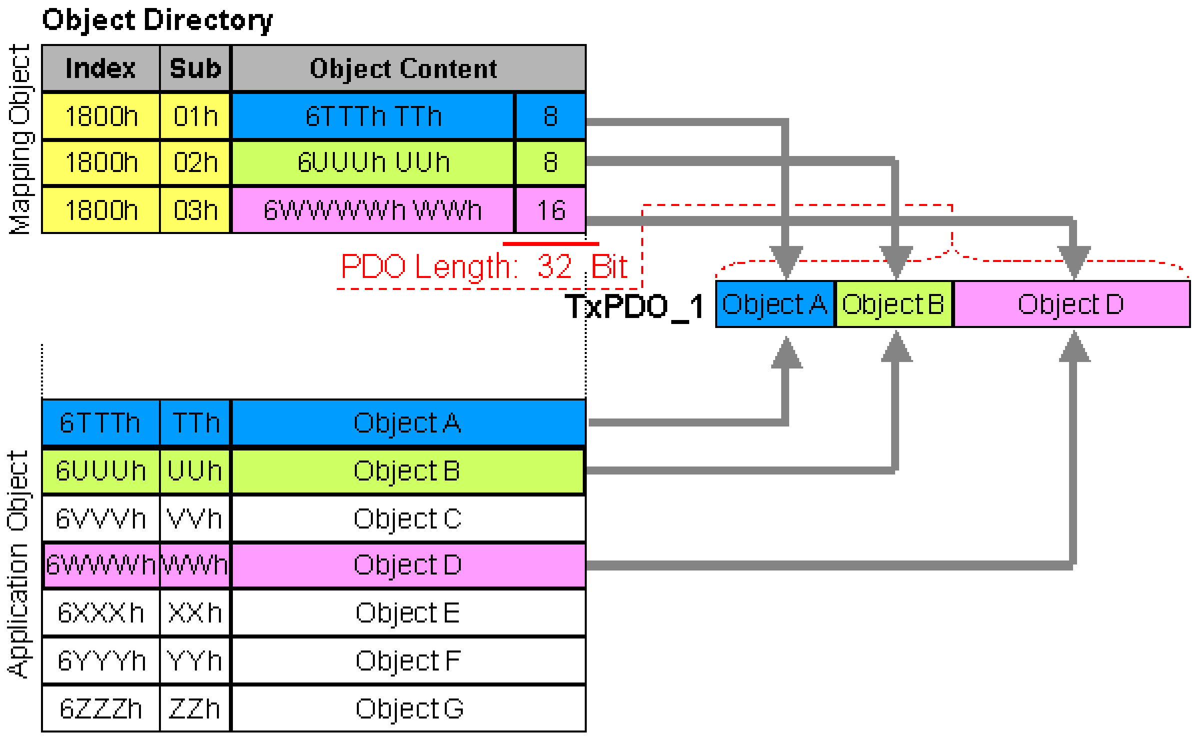
Notes on PDO Parameterization [▸ 104]

PDO Mapping

PDO mapping refers to mapping of the application objects (real time data) from the object directory to the process data objects. The CANopen device profile provide a default mapping for every device type, and this is appropriate for most applications. Thus the default mapping for digital I/O simply represents the inputs and outputs in their physical sequence in the transmit and receive process data objects.

The default PDOs for drives contain 2 bytes each of a control and status word and a set or actual value for the relevant axis.

The current mapping can be read by means of corresponding entries in the object directory. These are known as the mapping tables. The first location in the mapping table (sub-index 0) contains the number of mapped objects that are listed after it. The tables are located in the object directory at index 0x1600ff for the RxPDOs and at 0x1A00ff for the TxPDOs.



Digital and analog input/output modules: Read out the I/O number

The current number of digital and analog inputs and outputs can be determined or verified by reading out the corresponding application objects in the object directory:

Parameters	Object directory address
Number of digital input bytes	Index 0x6000, sub-index 0
Number of digital output bytes	Index 0x6200, sub-index 0
Number of analog inputs	Index 0x6401, sub-index 0
Number of analog outputs	Index 0x6411, sub-index 0

Variable mapping

As a rule, the default mapping of the process data objects already satisfies the requirements. For special types of application the mapping can nevertheless be altered: the Beckhoff CANopen Bus Couplers, for instance, thus support variable mapping, in which the application objects (input and output data) can be freely allocated to the PDOs. The mapping tables must be configured for this: as from Version 4 of CANopen, only the following procedure is permitted, and must be followed precisely:

1. First delete the PDO (set 0x1400ff, or 0x1800ff, sub-index 1, bit 31 to "1")
2. Set sub-index 0 in the mapping parameters (0x1600ff or 0x1A00ff) to "0"
3. Change mapping entries (0x1600ff or 0x1A00ff, SI 1..8)
4. Set sub-index 0 in the mapping parameters to the valid value. The device then checks the entries for consistency.
5. Create PDO by entering the identifier (0x1400ff or 0x1800ff, sub-index 1).

Dummy Mapping

A further feature of CANopen is the mapping of placeholders, or dummy entries. The data type entries stored in the object directory, which do not themselves have data, are used as placeholders. If such entries are contained in the mapping table, the corresponding data from the device is not evaluated. In this way, for instance, a number of drives can be supplied with new set values using a single CAN telegram, or outputs on a number of nodes can be set simultaneously, even in event-driven mode.

2.3.4 Objects and Data

Device type

Device type

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1000	0	Device type	Unsigned32	ro	N	0x00000000	Statement of device type

The 32 bit value is divided into two 16 bit fields:

MSB	LSB
Additional information	Device profile number
0000 0000 0000 wxyz	0x191 (401_{dez})

The *additional information* contains data related to the signal type of the I/O device:
z=1 signifies digital inputs,
y=1 signifies digital outputs,
x=1 signifies analog inputs,
w=1 signifies analog outputs.
A BK5120 with digital and analog inputs, but with no outputs, thus returns 0x00 05 01 91.

Special terminals (such as serial interfaces, PWM outputs, incremental encoder inputs) are not considered. A Coupler that, for example, only has KL6001 serial interface terminals plugged in, thus returns 0x00 00 01 91.

The device type supplies only a rough classification of the device. The terminal identifier register of the Bus Coupler can be read for detailed identification of the Bus Couplers and the attached terminals (for details see register communication index 0x4500).

Error register

Error register

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1001	0	Error register	Unsigned8	ro	N	0x00	Error register

The 8 bit value is coded as follows:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ManSpec.	reserved	reserved	Comm.	reserved	reserved	reserved	Generic

ManSpec. Manufacturer-specific error, specified more precisely in object 1003.

Comm. Communication error (CAN overrun)

Generic An error that is not more precisely specified has occurred (the flag is set at every error message)

Error store

Error store

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1003	0x00	Predefined error field (Error store)	Unsigned8	rw	N	0x00	Object 1003h contains a description of the error that has occurred in the device - sub-index 0 has the number of error states stored.
	1	Actual error	Unsigned32	ro	N	None	Last error state to have occurred
	...	...	...	--	...	...	...
	10	Standard error field	Unsigned32	ro	N	None	A maximum of 10 error states are stored.

The 32 bit value in the error store is divided into two 16 bit fields:

MSB	LSB
Additional code	Error Code

The additional code contains the error trigger (see emergency object [▶ 88]) and thereby a detailed error description.

New errors are always saved at sub-index 1, all the other sub-indices being appropriately incremented. The whole error store is cleared by writing a 0 to sub-index 0.

If there has not been an error since power up, then object 0x1003 only consists of sub-index 0 with a 0 entered into it. The error store is cleared by a reset or a power cycle.

As is usual in CANopen, the LSB is transferred first, followed by the MSB.

Sync Identifier

Sync Identifier

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1005	0	COB-ID Sync Message	Unsigned32	rw	N	0x80000080	Identifier of the SYNC message

The bottom 11 bits of the 32 bit value contain the identifier (0x80=128 dec). Bit 30 indicates whether the device sends the SYNC telegram (1) or not (0). The CANopen I/O devices receive the SYNC telegram, and accordingly bit 30=0. For reasons of backwards compatibility, bit 31 has no significance.

Sync Interval

Sync Interval

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1006	0	Communication cycle period	Unsigned32	rw	N	0x00000000	Length of the SYNC interval in µs.

If a value other than zero is entered here, the bus node will go into the fault state if, during synchronous PDO operation, no SYNC telegram is received within the watchdog time. The watchdog time corresponds here to 1.5 times the communication cycle period that has been set - the planned SYNC interval can therefore be entered.

The I/O update is carried out at the Beckhoff CANopen bus nodes immediately after reception of the SYNC telegram, provided the following conditions are satisfied:

- Firmware status C0 or above (CANopen Version 4.01 or higher).

- All PDOs that have data are set to synchronous communication (0..240).

- The sync interval has been entered in object 0x1006 and (sync interval x lowest PDO transmission type) is less than 90ms.

The modules are then synchronised throughout.

Device name

Device name

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1008	0	Manufacturer Device Name	Visible String	ro	N	BK51x0, LC5100, IPxxxx-B510 or ILxxxx-B510	Device name of the bus node

Since the returned value is longer than 4 bytes, the segmented SDO protocol is used for transmission.

Hardware version

Hardware version

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1009	0	Manufacturer hardware-version	Visible String	ro	N	-	Hardware version number of the bus node

Since the returned value is longer than 4 bytes, the segmented SDO protocol is used for transmission.

Software version

Software version

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x100A	0	Manufacturer software-version	Visible String	ro	N	-	Software version number of the bus node

Since the returned value is longer than 4 bytes, the segmented SDO protocol is used for transmission.

Node number

Node number

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x100B	0	Node-ID	Unsigned32	ro	N	none	Set node number

The node number is supported for reasons of compatibility.

Guard time

Guard time

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x100C	0	Guard time [ms]	Unsigned16	rw	N	0	Interval between two guard telegrams. Is set by the NMT master or configuration tool.

Life time factor

Life time factor

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x100D	0	Life time factor	Unsigned8	rw	N	0	Life time factor x guard time = life time (watchdog for life guarding)

If a guarding telegram is not received within the life time, the node enters the error state. If the life time factor and/or guard time = 0, the node does not carry out any life guarding, but can itself be monitored by the master (node guarding).

Guarding identifier

Guarding identifier

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x100 E	0	COB-ID guarding protocol	Unsigned32	ro	N	0x000007xy, xy = NodeID	Identifier of the guarding protocol

The guarding identifier is supported for reasons of compatibility. Changing the guarding identifier has no longer been permitted since version 4 of CANopen.

Save parameters

Save parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1010	0	Store Parameter	Unsigned8	ro	N	1	Number of store options
	1	store all parameters	Unsigned32	rw	N	1	Stores all (storable) parameters

By writing the string *save* in ASCII code (hexadecimal 0x65766173) to sub-index 1, the current parameters are placed into non-volatile storage. (The byte sequence on the bus including the SDO protocol: 0x23 0x10 0x10 0x01 0x73 0x61 0x76 0x65).

The storage process takes about 3 seconds, and is confirmed, if successful, by the corresponding TxSDO (0x60 in the first byte). Since the Bus Coupler is unable to send or receive any CAN telegrams during the storage process, saving is only possible when the node is in the pre-operational state. It is recommended that the entire network is placed into the pre-operational state before such storage. This avoids a buffer overflow.

Data saved includes:

- The terminals currently inserted (the number of each terminal category)
- All PDO parameters (identifier, transmission type, inhibit time, mapping).

NoteThe stored identifiers apply afterwards, not the default identifiers derived from the node addresses. Changes to the DIP switch setting no longer affects the PDOs!

- All SYNC parameters
- All guarding parameters
- Limit values, delta values and interrupt enables for analog inputs

Parameters directly stored in the terminals by way of register communication are immediately stored there in non-volatile form.

Load default values

Load default values

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1011	0	Restore Parameter	Unsigned8	ro	N	4	Number of reset options
	1	Restore all parameters	Unsigned32	rw	N	1	Resets all parameters to their default values
	4	Set manufacturer Defaults	Unsigned32	rw	N	1	Resets all coupler parameters to manufacturer's settings (including registers)

Writing the string *load* in ASCII code (hexadecimal 0x64616F6C) into sub-index 1 resets all parameters to default values (as initially supplied) **at the next boot (reset)**.

(The byte sequence on the bus including the SDO protocol: 0x23 0x11 0x10 0x01 0x6C 0x6F 0x61 0x64).

This makes the default identifiers for the PDOs active again.

Emergency identifier

Emergency identifier

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1014	0	COB-ID Emergency	Unsigned32	rw	N	0x00000080, + NodeID	Identifier of the emergency telegram

The bottom 11 bits of the 32 bit value contain the identifier (0x80=128 dec). The MSBit can be used to set whether the device sends (1) the emergency telegram or not (0).

Alternatively, the bus node's diagnostic function can also be switched off using the *Device diagnostics* bit in the K-Bus configuration (see object 0x4500).

Consumer heartbeat time

Consumer heartbeat time

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1016	0	Number of elements	Unsigned8	ro	N	2	The consumer heartbeat time describes the expected heartbeat cycle time and the node ID of the monitored node
	1	Consumer heartbeat time	Unsigned32	rw	N	0	Watchdog time in ms and node ID of the monitored node

The 32-bit value is used as follows:

MSB		LSB
Bit 31...24	Bit 23...16	Bit 15...0
Reserved (0)	Node ID (unsigned8)	Heartbeat time in ms (unsigned16)

The monitored identifier can be obtained from the node ID by means of the default identifier allocation: Guard-ID = 0x700 + Node-ID.

As is usual in CANopen, the LSB is transferred first, followed by the MSB.

Producer heartbeat time

Producer heartbeat time

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1017	0	Producer heartbeat time	Unsigned16	rw	N	0	Interval in ms between two transmitted heartbeat telegrams

Device identifier (identity object)

Device identifier (identity object)

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1018	0	Identity Object: Number of elements	Unsigned8	ro	N	4	The identity object contains general information about the type and version of the device.
	1	Vendor ID	Unsigned32	ro	N	0x00000002	Manufacturer identifier. Beckhoff has vendor ID 2
	2	Product Code	Unsigned32	ro	N	Depends on the product	Device identifier
	3	Revision Number	Unsigned32	ro	N	-	Version number
	4	Serial Number	Unsigned32	ro	N	-	Production date low word, high byte: calendar week (dec), low word, low byte: calendar year

Product	Product Code
BK5120	0x11400
BK5110	0x113F6
LC5100	0x113EC
IPwxyz-B510	0x2wxyz
IL2301-B510	0x2008FD

Server SDO parameters

Server SDO parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1200	0	Number of elements	Unsigned8	ro	N	2	Communication parameters of the server SDO. Sub-index 0: number of following parameters
	1	COB-ID Client ->Server	Unsigned32	ro	N	0x000006xy, xy=Node-ID	COB-ID RxSDO (Client -> Server)
	2	COB-ID Server ->Client	Unsigned32	ro	N	0x00000580 + Node-ID	COB-ID TxSDO (Client -> Server)

This is contained in the object directory for reasons of backwards compatibility.

Communication parametersfor the 1st RxPDO

for the 1st RxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1400	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the first receive PDO. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x000002xy, xy=Node-ID	COB-ID (Communication Object Identifier) RxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Present for reasons of backwards compatibility, but not used in the RxPDO.
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer. Watchdog time defined for monitoring reception of the PDO.

Sub-index 1 (COB-ID): The bottom 11 bits of the 32 bit value (bits 0-10) contain the CAN identifier. The MSB (bit 31) indicates whether the PDO exists currently (0) or not (1). Bit 30 indicates whether an RTR access to this PDO is permissible (0) or not (1). Changing the identifier (bits 0-10) is not allowed while the object exists (bit 31=0). Sub-index 2 contains the type of the transmission (see introduction to PDOs).

Communication parametersfor the 2nd RxPDO

for the 2nd RxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1401	0	Number of elements	Unsigned8	ro	N	5	Communication parameter for the second receive PDO.
	1	COB-ID	Unsigned32	rw	N	0x000003xy, xy=Node-ID	COB-ID (Communication Object Identifier) RxPDO2
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Present for reasons of backwards compatibility, but not used in the RxPDO.
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer. Watchdog time defined for monitoring reception of the PDO.

Communication parametersfor the 3rd RxPDO

for the 3rd RxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1402	0	Number of elements	Unsigned8	ro	N	5	Communication parameter for the third receive PDO.
	1	COB-ID	Unsigned32	rw	N	0x000004xy, xy=Node-ID	COB-ID (Communication Object Identifier) RxPDO3
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Present for reasons of backwards compatibility, but not used in the RxPDO.
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer. Watchdog time defined for monitoring reception of the PDO.

Communication parametersfor the 4th RxPDO

for the 4th RxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1403	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the fourth receive PDO.
	1	COB-ID	Unsigned32	rw	N	0x000005xy, xy=Node-ID	COB-ID (Communication Object Identifier) RxPDO4
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Present for reasons of backwards compatibility, but not used in the RxPDO.
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer. Watchdog time defined for monitoring reception of the PDO.

Communication parametersfor the 5th-16th RxPDOs

for the 5th-16th RxPDOsCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1404 - 0x140F (depending on the device type)	0	Number of elements	Unsigned8	ro	N	5	Communication parameter for the 5th to 16th receive PDOs.
	1dth="5%"> 1	COB-ID	Unsigned32	rw	N	0x8000000	COB-ID (Communication Object Identifier) RxPDO5...16
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Present for reasons of backwards compatibility, but not used in the RxPDO.
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer. Watchdog time defined for monitoring reception of the PDO.

The number of RxPDOs for each bus node type can be found in the technical data.

Mapping parametersfor the 1st RxPDO

for the 1st RxPDOMapping parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1600	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameter of the first receive PDO; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x62000108	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x62000208	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x62000808	8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The first receive PDO (RxPDO1) is provided by default for digital output data. Depending on the number of outputs inserted, the necessary length of the PDO is automatically determined, and the corresponding objects are mapped. Since the digital outputs are organised in bytes, the length of the PDO in bytes can be found directly at sub-index 0.

Changes to the mapping

The following sequence must be observed in order to change the mapping (specified as from CANopen, version 4):

1. Delete PDO (set bit 31 in the identifier entry (sub-index 1) of the communication parameters to 1)
2. Deactivate mapping (set sub-index 0 of the mapping entry to 0)
3. Change mapping entries (sub-indices 1...8)
4. Activate mapping (set sub-index 0 of the mapping entry to the correct number of mapped objects)
5. Create PDO (set bit 31 in the identifier entry (sub-index 1) of the communication parameters to 0)

Mapping parametersfor the 2nd RxPDO

for the 2nd RxPDOMapping parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1601	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameter of the second receive PDO; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x64110110	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x64110210	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x00000000	8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The second receive PDO (RxPDO2) is provided by default for analog outputs. Depending on the number of outputs inserted, the necessary length of the PDO is automatically determined, and the corresponding objects are mapped. Since the analog outputs are organised in words, the length of the PDO in bytes can be found directly at sub-index 0.

A specific sequence must be observed in order to change the mapping (see object index 0x1600).

Mapping parametersfor the 3rd-16th RxPDO

for the 3rd-16th RxPDOMapping parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1602-0x160F (depending on the device type)	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameters for the third to sixteenth receive PDOs; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x00000000 (see text)	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x00000000 (see text)	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x00000000 (see text)	8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The 3rd to 16th receive PDOs (RxPDO3ff) are automatically given a default mapping by the bus node depending on the attached terminals (or depending on the extension modules). The procedure is described in the section on PDO Mapping [▶ 63].

A specific sequence must be observed in order to change the mapping (see object index 0x1600).

NoteDS401 V2 specifies analog input and/or output data as the default mapping for PDOs 3+4. This corresponds to Beckhoff's default mapping when less than 65 digital inputs or outputs are present. In order to ensure backwards compatibility, the Beckhoff default mapping is retained - the mapping behaviour of the devices therefore corresponds to DS401 V1, where in all other respects they accord with DS401 V2.

Communication parametersfor the 1st TxPDO

for the 1st TxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1800	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the first transmit PDO. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x00000180 + Node-ID	COB-ID (Communication Object Identifier) TxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Repetition delay [value x 100 µs]
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer

Sub-index 1 (COB-ID): The bottom 11 bits of the 32 bit value (bits 0-10) contain the CAN identifier. The MSB (bit 31) indicates whether the PDO exists currently (0) or not (1). Bit 30 indicates whether an RTR access to this PDO is permissible (0) or not (1). Changing the identifier (bits 0-10) is not allowed while the object exists (bit 31=0). Sub-index 2 contains the type of transmission, sub-index 3 the repetition delay between two PDOs of the same type, while sub-index 5 contains the event timer. Sub-index 4 is retained for reasons of compatibility, but is not used. (See also the introduction to PDOs.)

Communication parametersfor the 2nd TxPDO

for the 2nd TxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1801	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the second transmit PDO. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x00000280 + Node-ID	COB-ID (Communication Object Identifier) TxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Repetition delay [value x 100 µs]
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer

The second transmit PDO is provided by default for analog inputs, and is configured for event-driven transmission (transmission type 255). Event-driven mode must first be activated (see object 0x6423), otherwise the inputs can only be interrogated (polled) by remote transmission request (RTR).

Communication parametersfor the 3rd TxPDO

for the 3rd TxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1802	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the third transmit PDO. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x00000380 + Node-ID	COB-ID (Communication Object Identifier) TxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Repetition delay [value x 100 µs]
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer

The third transmit PDO contains analog input data as a rule (see Mapping [▶ 63]). It is configured for event-driven transmission (transmission type 255). Event-driven mode must first be activated (see object 0x6423), otherwise the inputs can only be interrogated (polled) by remote transmission request (RTR).

Communication parametersfor the 4th TxPDO

for the 4th TxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1803	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the fourth transmit PDO. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x00000480 + Node-ID	COB-ID (Communication Object Identifier) TxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Repetition delay [value x 100 µs]
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer

The fourth transmit PDO contains analog input data as a rule (see Mapping [▶ 63]). It is configured for event-driven transmission (transmission type 255). Event-driven mode must first be activated (see object 0x6423), otherwise the inputs can only be interrogated (polled) by remote transmission request (RTR).

Communication parametersfor the 5th-16th TxPDOs

for the 5th-16th TxPDOsCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1804-0x180F (depending on the device type)	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the 5th to 16th transmit PDOs. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x0000000	COB-ID (Communication Object Identifier) TxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Repetition delay [value x 100 µs]
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer

Mapping 1st TxPDO

Mapping 1st TxPDO

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1A00	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameter of the first transmit PDO; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x60000108	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x60000208	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x60000808	8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The first transmit PDO (TxPDO1) is provided by default for digital input data. Depending on the number of inputs inserted, the necessary length of the PDO is automatically determined, and the corresponding objects are mapped. Since the digital inputs are organised in bytes, the length of the PDO in bytes can be found directly at sub-index 0.

A specific sequence must be observed in order to change the mapping (see object index 0x1600).

Mapping 2nd TxPDO

Mapping 2nd TxPDO

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1A01	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameter of the second transmit PDO; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x64010110	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x64010210	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N		8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The second transmit PDO (TxPDO2) is provided by default for analog input data. Depending on the number of inputs inserted, the necessary length of the PDO is automatically determined, and the corresponding objects are mapped. Since the analog inputs are organised in words, the length of the PDO in bytes can be found directly at sub-index 0.

A specific sequence must be observed in order to change the mapping (see object index 0x1600).

Mapping 3rd-16th TxPDO

Mapping 3rd-16th TxPDO

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1A02-0x1A0F (depending on the device type)	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameters for the third to sixteenth transmit PDOs; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x00000000 (see text)	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x00000000 (see text)	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x00000000 (see text)	8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The 3rd to 16th transmit PDOs (TxPDO3ff) are automatically given a default mapping by the bus node depending on the attached terminals (or depending on the extension modules). The procedure is described in the section on PDO Mapping [▶ 63].

A specific sequence must be observed in order to change the mapping (see object index 0x1600).

NoteDS401 V2 specifies analog input and/or output data as the default mapping for PDOs 3+4. This corresponds to Beckhoff's default mapping when less than 65 digital inputs or outputs are present. In order to ensure backwards compatibility, the Beckhoff default mapping is retained - the mapping behavior of the devices therefore corresponds to DS401 V1, where in all other respects they accord with DS401 V2.

For the sake of completeness, the following object entries are also contained in the object directory (and therefore also in the EDS files):

Index	Meaning
0x2000	Digital inputs (function identical to object 0x6000)
0x2100	Digital outputs (function identical to object 0x6100)
0x2200	1-byte special terminals, inputs (at present no terminals corresponding to this type are included in the product range)
0x2300	1-byte special terminals, outputs (at present no terminals corresponding to this type are included in the product range)
0x2400	2-byte special terminals, inputs (at present no terminals corresponding to this type are included in the product range)
0x2500	2-byte special terminals, outputs (at present no terminals corresponding to this type are included in the product range)
0x2E00	7-byte special terminals, inputs (at present no terminals corresponding to this type are included in the product range)
0x2F00	7-byte special terminals, outputs (at present no terminals corresponding to this type are included in the product range)

3-byte special terminals, input data

3-byte special terminals, input data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2600	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 3-byte special channels, inputs
	1	1st input block	Unsigned24	ro	Y	0x000000	1st input channel
	...	...	...	...	...	...	...
	0X80	128th input block	Unsigned24	ro	Y	0x000000	128th input channel

Example of special terminals with 3-byte input data (in the default setting): KL2502 (PWM outputs, 2 x 3 bytes)

3-byte special terminals, output data

3-byte special terminals, output data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2700	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 3-byte special channels, outputs
	1	1st output block	Unsigned24	rww	Y	0x000000	1st output channel
	...	...	...	...	...	...	...
	0X80	128th output block	Unsigned24	rww	Y	0x000000	128th output channel

Example of special terminals with 3-byte output data (in the default setting): KL2502 (PWM outputs, 2 x 3 bytes)

4-byte special terminals, input data

4-byte special terminals, input data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2800	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 4-byte special channels, inputs
	1	1st input block	Unsigned32	ro	Y	0x00000000	1st input channel
	...	...	...	...	...	...	...
	0X80	128th input block	Unsigned32	ro	Y	0x00000000	128th input channel

Examples of special terminals with 4-byte input data (in the default setting): KL5001, KL6001, KL6021, KL6051

4-byte special terminals, output data

4-byte special terminals, output data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2900	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 4-byte special channels, outputs
	1	1st output block	Unsigned32	rww	Y	0x00000000	1st output channel
	...	...	...	...	...	...	...
	0X80	128th output block	Unsigned32	rww	Y	0x00000000	128th output channel

Examples of special terminals with 4-byte output data (in the default setting): KL5001, KL6001, KL6021, KL6051

5-byte special terminals, input data

5-byte special terminals, input data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2A00	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 5-byte special channels, inputs
	1	1st input block	Unsigned40	ro	Y	0x0000000000	1st input channel
	...	...	...	...	...	...	...
	0X40	64th input block	Unsigned40	ro	Y	0x0000000000	64th input channel

Example of special terminals with 5-byte input data (in the default setting): KL1501

5-byte special terminals, output data

5-byte special terminals, output data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2B00	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 5-byte special channels, outputs
	1	1st output block	Unsigned40	rww	Y	0x0000000000	1st output channel
	...	...	...	...	...	...	...
	0X40	64th output block	Unsigned40	rww	Y	0x0000000000	64th output channel

Example of special terminals with 5-byte output data (in the default setting): KL1501

6-byte special terminals, input data

6-byte special terminals, input data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2C00	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 6-byte special channels, inputs
	1	1st input block	Unsigned48	ro	Y	0x0000000000	1st input channel
	...	...	...	...	...	...	...
	0X40	64th input block	Unsigned48	ro	Y	0x0000000000	64th input channel

Example of special terminals with 6-byte input data (in the default setting): KL5051, KL5101, KL5111

6-byte special terminals, output data

6-byte special terminals, output data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2D00	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 6-byte special channels, outputs
	1	1st output block	Unsigned48	rww	Y	0x0000000000	1st output channel
	...	...	...	...	...	...	...
	0X40	64th output block	Unsigned48	rww	Y	0x0000000000	64th output channel

Example of special terminals with 6-byte output data (in the default setting): KL5051, KL5101, KL5111

8-byte special terminals, input data

8-byte special terminals, input data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x3000	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 6-byte special channels, inputs
	1	1st input block	Unsigned64	ro	Y	0x0000000000	1st input channel
	...	...	...	...	...	...	...
	0x40	64th input block	Unsigned64	ro	Y	0x0000000000	64th input channel

Example for special terminals with 8-byte input data: KL5101 (with word alignment, not according to the default setting)

8-byte special terminals, output data

8-byte special terminals, output data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x3100	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 6-byte special channels, outputs
	1	1st output block	Unsigned64	rww	Y	0x0000000000	1st output channel
	...	...	...	...	...	...	...
	0X40	64th output block	Unsigned64	rww	Y	0x0000000000	64th output channel

Example for special terminals with 8-byte output data: KL5101 (with word alignment, not according to the default setting)

Bus node register communication

Bus node register communication

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x4500	0	Register Access	Unsigned32	rw	N	none	Access to internal bus node registers

The 32 bit value is composed as follows:

MSB			**LSB**
Access (bit 7) + table number (bits 6...0)	Register number	High byte register value	Low byte register value
[0..1] + [0...0x7F]	[0...0xFF]	[0...0xFF]	[0...0xFF]

As is usual in CANopen, the LSB is transferred first, followed by the MSB.

Accessing index 0x4500 allows any registers in the bus station to be written or read. The channel number and the register are addressed here with a 32 bit data word.

Reading the register value

The coupler must first be informed of which register is to be read. This requires an SDO write access to the appropriate index/sub-index combination, with:
- table number (access bit = 0) in byte 3
- register address in byte 2 of the 32-bit data value.

Bytes 1 and 0 are not evaluated if the access bit (MSB of byte 3) equals 0. The register value can then be read with the same combination of index and sub-index.

After the writing of the register address to be read, the coupler sets the access bit to 1 until the correct value is available. Thus an SDO read access must check that the table number lies in the range from 0...0x7F.

An access error during register communication is indicated by the corresponding return value in the SDO protocol (see the SDO section, Breakdown of parameter communication).

An example of reading register values

It is necessary to determine which baud rate index has been assigned to switch setting 1,1 (DIP 7,8). (See the section covering *Network addresses and baud rates*). To do this, the value in table 100, register 3, must be read. This means that the following SDO telegrams must be sent:

Write access (download request) to index 4500, sub-index 0, with the 32 bit data value 0x64 03 00 00.

Id=0x600+Node-ID DLC=8; Data=23 00 45 00 00 00 03 64

Then a read access (upload request) to the same index/sub-index. The data value sent here is irrelevant (00 is used here).

Id=0x600+Node-ID DLC=8; Data=40 00 45 00 00 00 00 00

The coupler responds with the upload response telegram:

Id=0x580+Node-ID DLC=8; Data=43 00 45 00 04 00 03 64

This tells us that the value contained in this register is 4, and this baud rate index corresponds to 125 kbit/s (the default value).

Writing register values

SDO write access to the corresponding combination of index and sub-index with:
- table number + 0x80 (access bit = 1) in byte 3
- register address in byte 2
- high byte register value in byte 1
- low byte register value in byte 0 of the 32-bit data value.

Remove coupler write protection

Before the registers of the Bus Coupler can be written, the write protection must first be removed. In order to do this, the following values must be written in the given sequence to the corresponding registers:

Step	Table	Register	Value	Corresponding SDO download value (0x4500/0)
1.	99	2	45054 (0xAFFE)	0xE3 02 AF FE (0xE3=0x63(=99)+ 0x80)
2.	99	1	1 (0x0001)	0xE3 01 00 01
3.	99	0	257 (0x0101)	0xE3 00 01 01

Remove coupler write protection (CAN representation)

In order to remove the coupler write protection, the following SDO telegrams (download requests) must thus be sent to the coupler:

Id=0x600+Node-ID DLC=8; Data=23 00 45 00 FE AF 02 E3

Id=0x600+Node-ID DLC=8; Data=23 00 45 00 01 00 01 E3

Id=0x600+Node-ID DLC=8; Data=23 00 45 00 01 01 00 E3

An example of writing register values

After the write protection has been removed, the baud rate index for DIP switch setting 1,1 is to be set to the value 7. This will assign a baud rate of 20 kbaud to this switch setting.

This requires the value 7 to be written into table 100, register 3. This is done with an SDO write access (download request) to index 0x4500, sub-index 0 with the 32 bit value E4 03 00 07 (0xE4 = 0x64+0x80):

Id=0x600+Node-ID DLC=8; Data=23 00 45 00 07 00 03 E4

Identify terminals

The identifier of the coupler (or of the bus station) and of the attached Bus Terminals can be read from the Bus Coupler's table 9. Register 0 then contains the identifier of the Bus Coupler itself, register 1 the identifier of the first terminal and register n the identification of the n^{th} terminal:

Table number	Register number	Description	Value range
9	0	Bus station identifier	0 - 65535
9	1-255	Identifier of the extension module/bus terminal	0 - 65535

The Bus Coupler description in register number 0 contains 5120 = 0x1400 for the BK5120, 5110 = 0x13F6 for the BK5110 and 5100 = 0x13EC for the LC5100. The Fieldbus Box modules contain the identifier 510 dec = 0x1FE in register 0.

In the case of analog and special terminals, the terminal identifier (dec) is contained in the extension module identifier or the terminal description.
Example: if a KL3042 is plugged in as the third terminal, then register 3 contains the value 3042_{dec} (0x0BE2).

The following bit identifier is used for digital terminals:

MSB								LSB							
1	s6	s5	s4	s3	s2	s1	s0	0	0	0	0	0	0	a	e

s6...s1: data width in bits; a=1: output terminal; e=1: input terminal

This identifier scheme results in the terminal descriptions listed below:

Extension module identifier	Meaning
0x8201	2 bit digital input terminal, e.g. KL1002, KL1052, Kl9110, KL9260
0x8202	2 bit digital output terminal, e.g. KL2034, KL2612, KL2702
0x8401	4 bit digital input terminal, e.g. KL1104, KL1124, KL1194
0x8402	4 bit digital output terminal, e.g. KL2124, KL2134, KL2184
0x8403	4 bit digital input/output terminal, e.g. KL2212

General coupler configuration (table 0)

Table 0 of the Bus Coupler contains the data for the general coupler configuration. It is not, as a general rule, necessary to change this; however, for special applications it is possible to change the settings using the KS2000 configuration software, or through direct access via register communication. The write protection must first be removed in order to do this (see above).

The relevant register entries are described below:

K-Bus configuration

Table 0, register 2, contains the K-Bus configuration, and is coded as follows (default value: 0x0006):

MSB								LSB							
0	0	0	0	0	0	0	0	0	0	0	0	0	D	G	A

A: Auto-reset

If there is a K-Bus error, attempts are made cyclically to start the K-Bus up again through a reset. If emergency telegrams and guarding are not evaluated, activation of auto-reset can lead to output and input information being lost without that loss being noticed.

0: No auto-reset (default)

1: Auto-reset active

G: Device diagnostics

Reporting (by means of emergency telegram), that, for example
- a current input is open circuit (with diagnostics)
- 10 V exceeded at a 1-10V input terminal

0: Device diagnostics switched off

1: Device diagnostics active (default)

D: Diagnostic data

from digital terminals is included in the process image (e.g. KL2212). This flag is only evaluated when device diagnostics is active (see above).

0: Do not display

1: Display (default)

Process image description

Table 0, register 3, contains the process image description, and is coded as follows (default value: 0x0903):

MSB								LSB							
0	0	0	0	k1	k0	f1	f0	0	0	a	0	d	k	1	1

k0...k1: Reaction to K-Bus errors

0,2: Inputs remain unchanged (default = 2);

1: Set inputs to 0 (TxPDO with zeros is sent)

f0...f1: Reaction to fieldbus error

0: Stop the K-Bus cycles, watchdog in the terminals triggers, fault output values become active. The old output values are initially set during a restart.

1: Set outputs to 0, then stop the K-Bus cycles (default). 2: Outputs remain unchanged.

a: Word alignment (of analog and special terminals)

0: No alignment (default)

1: Map data to word boundaries (process data always starts on an even address in the PDO)

d: Data format for complex terminals (analog and special terminals)

0: Intel format (default)

1: Motorola format

k: Evaluation of complex terminals (analog and special terminals)

0: User data only (default)

1: Complete evaluation (note: analog channels then, for example, need 3 input and 3 output bytes instead of, e.g., 2 input bytes; instead of 4 channels per PDO, 2 channels require a RxPDO and a TxPDO)

Bus Terminal / Extension Box register communication

Bus Terminal / Extension Box register communication

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x4501	0	Access Terminal Register	Unsigned8	ro	N	none	Index 0x4501 allows access to all the registers in the bus terminal or extension module. Sub-index 0 contains the number of attached bus terminals.
	1	Access Reg. Terminal 1	Unsigned32	rw	N	none	Access to bus terminal or extension module register 1
	...	...	...	...	...	...	...
	0XFE	Access Reg. Terminal 254	Unsigned32	rw	N	none	Access to bus terminal or extension module register 254

The 32 bit value is composed as follows:

MSB			LSB
Access (bit 7) + channel number (bits 6...0)	Register number	High byte register value	Low byte register value
[0..1] + [0...0x7F]	[0...0xFF]	[0...0xFF]	[0...0xFF]

As is usual in CANopen, the LSB is transferred first, followed by the MSB.

Accessing index 0x4501 allows the user registers in the bus terminal or extension module to be written or read. The modules have a set of registers for each input or output channel. The modules are addressed by means of the sub-index; the channel number and register are addressed in the 32-bit data value. Channel number 0 corresponds here to the first channel, 1 to the second channel, and so forth.

Reading the register value

The coupler must first be informed of which register is to be read. This requires an SDO write access to the appropriate index/sub-index combination, with:
- channel number (access bit = 0) in byte 3
- register address in byte 2 of the 32-bit data value.

Bytes 1 and 0 are not evaluated if the access bit (MSB of byte 3) equals 0. The register value can then be read with the same combination of index and sub-index.

After the writing of the register address to be read, the coupler sets the access bit to 1 until the correct value is available. Thus an SDO read access must check that the table number lies in the range from 0...0x7F.

An access error during register communication is indicated by the corresponding return value in the SDO protocol (see the SDO section, Breakdown of parameter communication).

An example of reading register values

The thermocouple type to which the second input channel of a KL3202 Thermocouple Input Terminal has been set is to be determined. This requires feature register 32 to be read. The terminal is located in the fifth slot, next to the Bus Coupler. This means that the following SDO telegrams must be sent:

Write access (download request) to index 4501, sub-index 5 with 32 bit data value 01 20 00 00 (0x01 = 2nd channel, 0x20 = register 32)
Id=0x600+Node-ID DLC=8; Data=23 01 45 05 00 00 20 01

Then a read access (upload request) to the same index/sub-index. The data value sent here is irrelevant (0x00 is used here).
Id=0x600+Node-ID DLC=8; Data=40 01 45 05 00 00 00 00

The coupler responds with the upload response telegram:
Id=0x580+Node-ID DLC=8; Data=43 01 45 05 06 31 20 01

This means that the feature register contains the value 31 06. The upper 4 bits indicate the thermocouple type. Their value here is 3, which means that PT500 is the type that has been set for this channel (see the KL3202 documentation).

Writing register values

SDO write access to the corresponding combination of index and sub-index with:
- channel number + 0x80 (access bit = 1) in byte 3
- register address in byte 2
- high byte register value in byte 1
- low byte register value in byte 0 of the 32-bit data value.

NOTE

WarningIf the write protection is not removed (as a result, for instance, of a faulty codeword), then although a write access to the terminal register will be confirmed (SDO download response), the value is not in fact entered into the register. It is therefore recommended that the value is read back after writing and compared.

Remove terminal write protection

Before the user registers in the Bus Terminal (register 32-xx, depending on terminal type or extension module) can be written to, it is first necessary for write protection to be removed. The following codeword is written for this purpose into register 31 of the channel concerned:

Write protection	Channel	Register	Value	Corresponding SDO download value (0x4500/0)
	1,2, 3 or 4	31 (0x1F)	4661 (0x1235)	8y 1F 12 35 (y = channel number)

Remove terminal write protection (CAN representation)

In order to remove the terminal's write protection, the following SDO telegram must thus be sent to the coupler:

Id=600 + Node-ID DLC=8; Data=23 01 45 xx 35 12 1F 8y

where xx is the terminal's slot, and y indicates the channel.

An example of removing write protection

Suppose that a KL3202 Thermocouple Input Terminal is inserted into slot 5 of a BK5120 that has node address 3, then the write protection for the first channel can be removed as follows:

Id=0x603 DLC=8; Data=23 01 45 05 35 12 1F 80

The following telegram is sent for the second channel:

Id=0x603 DLC=8; Data=23 01 45 05 35 12 1F 81

An example of writing register values

The type of thermocouple attached to the second channel of the KL3202 Terminal in slot 5 is now to be changed to PT1000. For this purpose, the value 2 must be written into the upper 4 bits (the upper nibble) of the feature register. It is assumed to that the default values are to be supplied for all the other bits in the feature register. Once the write protection has been removed, SDO write access (download request) is used to write the following 32 bit value into index 0x4501, sub-index 05: 81 20 21 06 (0x81=01+0x80; 0x20=32;0x2106 = register value).

The corresponding telegram on the bus looks like this:

Id=0x600+Node-ID DLC=8; Data=23 01 45 05 06 21 20 81

Activate PDOs

Activate PDOs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x5500	0	Activate PDO Defaults	Unsigned32	rw	N	0x00000000	sets PDO communication parameters for PDOs 2...11

CANopen defines default identifiers for 4 transmit (Tx) and 2 receive (Rx) PDOs, all other PDOs being initially deactivated after the nodes have started up. Index 0x5500 can activate all the PDOs that, in accordance with the terminals inserted, are filled with process data (manufacturer-specific default mapping). A manufacturer-specific default identifier allocation is carried out here for PDO5…11, while the transmission type and a uniform inhibit time is set for PDO2…11. PDOs that do not have process data (and which are thus superfluous in the present configuration) are not activated.

NoteThis object can only be written in the pre-operational state!

The 32-bit value is used as follows:

MSB			LSB
Transmission Type RxPDOs	Transmission Type TxPDOs	High byte inhibit time	Low byte inhibit time

As is usual in CANopen, the LSB is transferred first, followed by the MSB.

Example

Activate PDOs for bus node number 1, set inhibit time to 10ms (=100 x 100μs), set transmission type for TxPDOs to 255, and set transmission type for RxPDOs to 1. The following telegram must be sent:
Id=0x601 DLC=8; Data=23 00 55 00 64 00 FF 01

The node responds with the following telegram:
Id=0x601 DLC=8; Data=60 00 55 00 00 00 00 00

Identifiers used

The default identifier allocation for the additional PDOs leaves the pre-defined regions for guarding, SDOs etc. free, assumes a maximum of 64 nodes in the network with PDO6 as the next node, and proceeds according to the following scheme:

Object	Function code	Resulting COB ID (hex)	Resulting COB ID (dec)
TxPDO5	1101	0x681 - 0x6BF	1665 - 1727
RxPDO5	1111	0x781 - 0x7BF	1921- 1983
TxPDO6	00111	0x1C1 - 0x1FF	449 - 511
RxPDO6	01001	0x241 - 0x27F	577 - 639
TxDPO7	01011	0x2C1 - 0x2FF	705 - 767
RxPDO7	01101	0x341 - 0x37F	833 - 895
TxPDO8	01111	0x3C1- 0x3FF	961 - 1023
RxPDO8	10001	0x441 - 0x47F	1089 - 1151
TxPDO9	10011	0x4C1 - 0x4FF	1217 - 1279
RxPDO9	10101	0x541 - 0x57F	1345 - 1407
TxDPO10	10111	0x5C1 - 0x5FF	1473 - 1535
RxPDO10	11001	0x641 - 0x67F	1601- 1663
TxPDO11	11011	0x6C1 - 0x6FF	1729 - 1791
RxPDO11	11101	0x741 - 0x77F	1857 - 1919

NOTE

WarningEnsure that index 0x5500 is not used if Bus Couplers with more than 5 PDOs are present in networks with node addresses > 64, otherwise identification overlaps can occur. In that case, the PDO identifiers must be set individually.

For the sake of clarity, the default identifiers defined according to CANopen are also listed here:

Object	Function code	Resulting COB ID (hex)	Resulting COB ID (dec)
Emergency	0001	0x81 - 0xBF [0xFF]	129 - 191 [255]
TxPDO1	0011	0x181 - 0x1BF [0x1FF]	385 - 447 [511]
RxPDO1	0100	0x201 - 0x23F [0x27F]	513 - 575 [639]
TxPDO2	0101	0x281 - 0x2BF [0x2FF]	641 - 676 [767]
RxPDO2	0110	0x301 - 0x33F [0x37F]	769 - 831 [895]
TxDPO3	0111	0x381 - 0x3BF [0x3FF]	897 - 959 [1023]
RxPDO3	1000	0x401 - 0x43F [0x47F]	1025 - 1087 [1151]
TxPDO4	1001	0x481 - 0x4BF [0x4FF]	1153 - 1215 [1279]
RxPDO4	1010	0x501 - 0x53F [0x57F]	1281- 1343 [1407]
SDO (Tx)	1011	0x581 - 0x5BF [0x5FF]	1409 - 1471 [1535]
SDO (Rx)	1100	0x601 - 0x63F [0x67F]	1537 - 1599 [1663]
Guarding / Heartbeat/ Bootup	1110	0x701 - 0x73F [0x77F]	1793 - 1855 [1919]

The identifiers that result from the DIP switch settings on the coupler are given, as are the identifier regions for the node addresses 64...127 (not settable in Bus Couplers BK5110, BK5120 and LC5100) in square brackets. Addresses 1…99 can be set for the Fieldbus Box modules and the BK515x Bus Couplers.

The appendix [▶ 65] contains a tabular summary of all the identifiers.

Digital inputs

Digital inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6000	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available digital 8-bit input data blocks
	1	1st input block	Unsigned8	ro	Y	0x00	1st input channel
	...	...	...	...	...	...	...
	0XFE	254th input block	Unsigned8	ro	Y	0x00	254th input channel

Interrupt mask

Interrupt mask

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6126	0	Number of elements	Unsigned8	ro	N	Depending on type	The number of 32-bit interrupt masks = 2 x the number of TxDPOs
	1	IR-Mask0 TxPDO1	Unsigned32	rw	N	0xFFFFFFFF	IR-mask bytes 0...3 TxPDO1
	2	IR-Mask1 TxPDO1	Unsigned32	rw	N	0xFFFFFFFF	IR-mask bytes 4...7 TxPDO1
	3	IR-Mask0 TxPDO2	Unsigned32	rw	N	0xFFFFFFFF	IR-mask bytes 0...3 TxPDO2
	...	...	...	...	...	...	...
	0x20	IR-Mask1 TxPDO16	Unsigned32	rw	N	0xFFFFFFFF	IR-mask bytes 4...7 TxPDO16

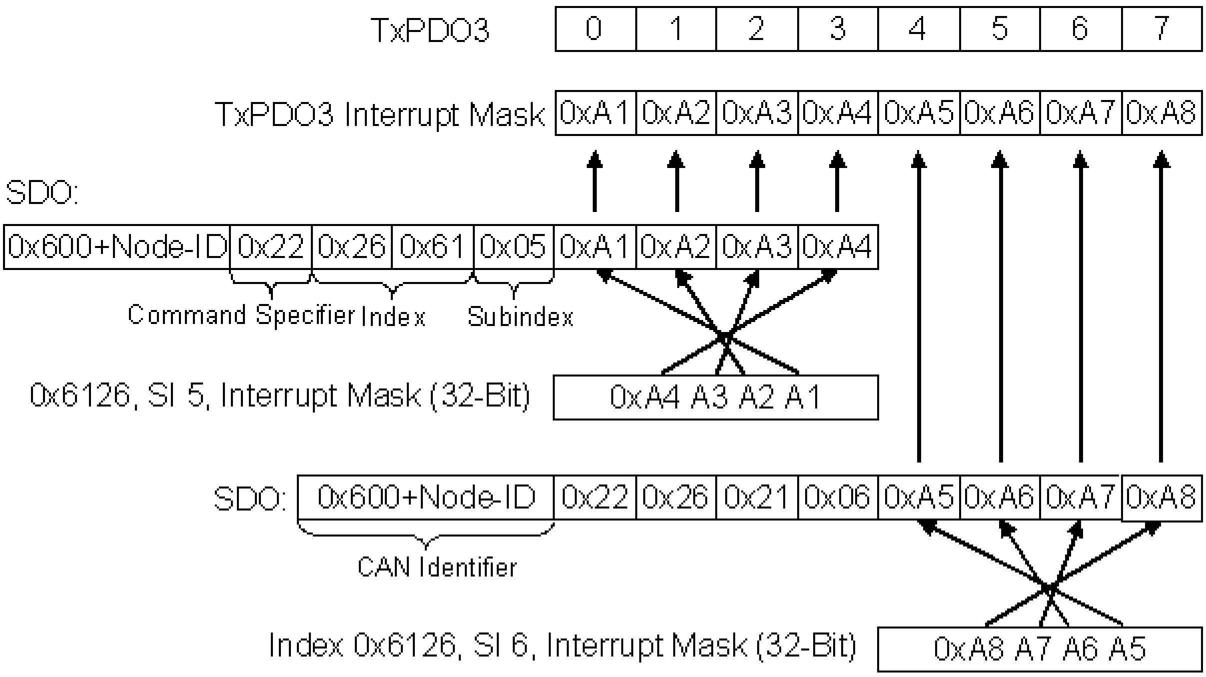
By default, every change in the value in an event-driven PDO causes a telegram to be sent. The interrupt mask makes it possible to determine which data changes are evaluated for this purpose. By clearing the appropriate ranges within the PDOs they are masked out for event-driving purposes (interrupt control). The interrupt mask does not just govern all the PDOs with digital inputs, but all the TxPDOs that are present. If the TxPDOs are shorter than 8 bytes, then the superfluous part of the IR mask is not evaluated.

The interrupt mask only has an effect on TxPDOs with transmission types 254 and 255. It is not stored in the device (not even through object 0x1010). Changes to the mask at runtime (when the status is operational) are possible, and are evaluated starting from the next change of input data.

The interrupt mask for TxPDOs with analog input data is not evaluated if either limit values (0x6424, 0x6425) or the delta function (0x6426) have been activated for the inputs.

This entry has been implemented in firmware C3 and above.

Example of data assignment



Application example

The value contained in a fast counter input is only to be transmitted when bits in the status word (the latch input, for instance) have changed. This requires the 32 bit counter value to be masked out (zeroed) in the interrupt mask. The status is located in byte 0, while the counter value is, by default, contained in bytes or 1..4 of the corresponding PDOs (TxPDO3 in this example, because < 65 digital and < 5 analog inputs are present).
This means that index 0x6126, sub-index5 must receive the value 0x0000 00FF and that sub-index6 must have 0xFFFF FF00 written into it.

The corresponding SDOs therefore appear as follows:

11 bit identifier	8 bytes of user data							
0x600+ node ID	0x22	0x26	0x61	0x05	0xFF	0x00	0x00	0x00

11 bit identifier	8 bytes of user data							
0x600+ node ID	0x22	0x26	0x61	0x06	0x00	0xFF	0xFF	0xFF

Digital outputs

Digital outputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6200	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available digital 8-bit output data blocks
	1	1st input block	Unsigned8	rw	Y	0x00	1st output channel
	...	...	...	...	...	...	...
	0XFE	254th input block	Unsigned8	rw	Y	0x00	254th output channel

Analog inputs

Analog inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6401	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of analog input channels available
	1	1st input	Unsigned16	ro	Y	0x0000	1st input channel
	...	...	...	...	...	...	...
	0XFE	254th input	Unsigned16	ro	Y	0x0000	254th input channel

The analog signals are displayed left aligned. The representation in the process image is therefore independent of the actual resolution. Detailed information on the data format can be found at the relevant signal type.

Analog outputs

Analog outputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6411	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of analog output channels available
	1	1st input block	Unsigned16	rw	Y	0x0000	1st output channel
	...	...	...	...	...	...	...
	0XFE	254th input block	Unsigned16	rw	Y	0x0000	254th output channel

The analog signals are displayed left aligned. The representation in the process image is therefore independent of the actual resolution. Detailed information on the data format can be found at the relevant signal type.

Event driven analog inputs

Event driven analog inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6423	0	Global Interrupt Enable	Boolean	rw	N	FALSE (0)	Activates the event-driven transmission of PDOs with analog inputs.

Although, in accordance with CANopen, the analog inputs in TxPDO2..4 are by default set to transmission type 255 (event driven), the event (the alteration of an input value) is suppressed by the event control in object 0x6423, in order to prevent the bus from being swamped with analog signals. It is recommended that the flow of data associated with the analog PDOs is controlled either through synchronous communication or through using the event timer. In event-driven operation, the transmission behavior of the analog PDOs can be parameterized before activation by setting the inhibit time (object 0x1800ff, sub-index 3) and/or limit value monitoring (objects 0x6424 + 0x6425) and/or delta function (object 0x6426).

Upper limit value analog inputs

Upper limit value analog inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6424	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of analog input channels available
	1	upper limit 1st input	Unsigned16	rw	Y	0x0000	Upper limit value for 1st input channel
	...	...	...	...	...	...	...
	0XFE	upper limit 254th input	Unsigned16	rw	Y	0x0000	Upper limit value for 254th input channel

Values different from 0 activate the upper limit value for this channel. A PDO is then transmitted if this limit value is exceeded. In addition, the event driven mode must be activated (object 0x6423). The data format corresponds to that of the analog inputs.

Lower limit value analog inputs

Lower limit value analog inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6425	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of analog input channels available
	1	lower limit 1st input	Unsigned16	rw	Y	0x0000	Lower limit value for 1st input channel
	...	...	...	...	...	...	...
	0XFE	lower limit 254th input	Unsigned16	rw	Y	0x0000	Lower limit value for 254th input channel

Values different from 0 activate the lower limit value for this channel. A PDO is then transmitted if the value falls below this limit value. In addition, the event driven mode must be activated (object 0x6423). The data format corresponds to that of the analog inputs.

Delta function for analog inputs

Delta function for analog inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6426	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of analog input channels available
	1	delta value 1st input	Unsigned16	rw	Y	0x0000	Delta value for the 1st input channel
	...	...	...	...	...	...	...
	0XFE	delta value 254th input	Unsigned16	rw	Y	0x0000	Delta value for the 254th input channel

Values different from 0 activate the delta function for this channel. A PDO is then transmitted if the value has changed by more than the delta value since the last transmission. In addition, the event driven mode must be activated (object 0x6423). The data format corresponds to that of the analog inputs (delta value: can only have positive values).

2.3.5 Automatic PDO Mapping

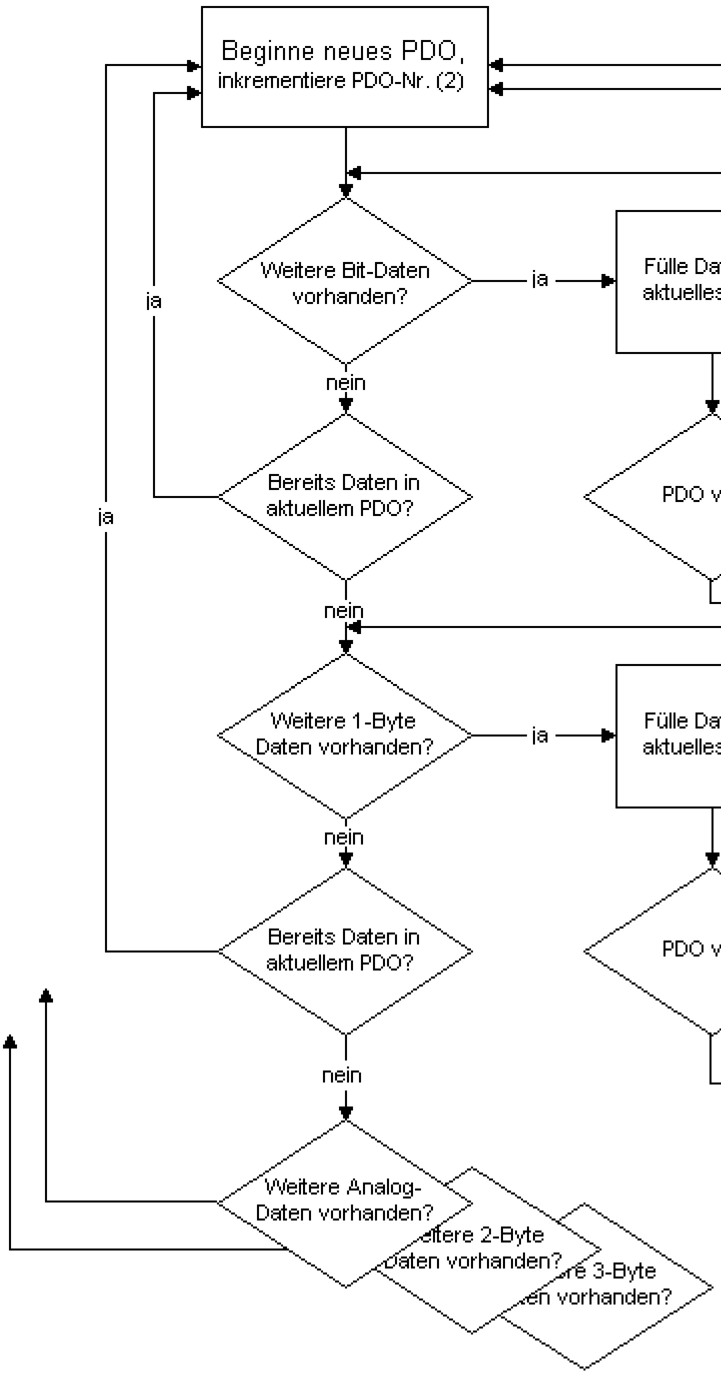
BK51x0, IL23x0-B510

PDO1 and PDO2 are occupied with digital and analog process data. For every other PDO, the CANopen node uses the procedure shown in flow chart below and occupies the PDOs with process data in the following order:

1. Digital I/Os (if more than than 64 are existent)
2. 1-byte terminals for special functions
3. Analog I/Os

4. 2-byte terminals for special functions
5. 3-byte terminals for special functions
6. ...10. 8-byte terminals for special functions

Data types are not mixed! For every new data type, a new PDO is filled (see example below).



Example

Example

A BK5120 (CANopen Coupler) has got

- 78 digital inputs und 48 digital outputs
- 6 analog inputs und 4 analog outputs
- one KL5001 (SSI-Sensor Interface: by default 4 byte inputs)
- one KL6001 (serial interface: by default 4 byte inputs and 4 byte outputs)

- one 1 KL5111 (Interface for incremental encoder - 6 byte inputs and 6 byte outputs)
- one KL6201 (AS-i master terminal) with default setting (22 byte process data interface).

PDO	data content (Mapping)	Object directory	PDO	data content (Mapping)	Object directory
RxPDO1	5 byte digital outputs 1...48	0x6200, SI 1..5	TxPDO1	8 byte digital inputs 1...64	0x6000, SI 1..8
RxPDO2	8 byte analog outputs 1...4	0x6411, SI 1..4	TxPDO2	4 byte analog inputs 1...4	0x6401, SI 1..4
RxPDO3	4 byte serial interface	0x2900, SI 1	TxPDO3	2 byte digital inputs 65...78	0x6000, SI 9..10
RxPDO4	6 byte encoder outputs	0x2D00, SI 1	TxPDO4	analog inputs 5 and 6	0x6401, SI 5..6
RxPDO5	8 byte AS-i master 1: parameter data block	0x3100, SI 1	TxPDO5	8 byte: 4 Bytes SSI and 4 Bytes serial interface	0x2800, SI 1..2
RxPDO6	8 byte AS-i master 1: process data block outputs AS-i slave 1...15	0x3100, SI 2	TxPDO6	6 Byte encoder inputs	0x2C00, SI 1
RxPDO7	8 byte AS-i master 1: process data block outputs AS-i slave 16...31	0x3100, SI 3	TxPDO7	8 byte AS-i master 1: parameter data block	0x3000, SI 1
			TxPDO8	8 byte AS-i master 1: process data block inputs AS-i Slave 1...15	0x3000, SI 2
			TxPDO9	8 byte AS-i master 1: process data block inputs AS-i Slave 16...31	0x3000, SI 3

2.3.6 CAN Identifier List

The list provided here should assist in identifying and assigning CANopen messages. All the identifiers allocated by the CANopen default identifier allocation are listed, as well as the manufacturer-specific default identifiers issued by BECKHOFF via object 0x5500 [▶ 22] (only to be used in networks with node addresses less than 64).

The following values can be used as search aids and "entry points" in the extensive identifier table in the *chm edition of the documentation:

Decimal: 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900

Hexadecimal: 0x181 0x1C1 0x201 0x301 0x401 0x501 0x601 0x701

Identifier allocation via object 0x5500 [▶ 22] follows this scheme:

Object	Resulting COB ID (hex)	Resulting COB ID (dec)
Emergency	0x81 - 0xBF [0xFF]	129 - 191 [255]
TxPDO1	0x181 - 0x1BF [0x1FF]	385 - 447 [511]
RxPDO1	0x201 - 0x23F [0x27F]	513 - 575 [639]
TxPDO2	0x281 - 0x2BF [0x2FF]	641 - 676 [767]
RxPDO2	0x301 - 0x33F [0x37F]	769 - 831 [895]
TxDPO3	0x381 - 0x3BF [0x3FF]	897 - 959 [1023]
RxPDO3	0x401 - 0x43F [0x47F]	1025 - 1087 [1151]
TxPDO4	0x481 - 0x4BF [0x4FF]	1153 - 1215 [1279]
RxPDO4	0x501 - 0x53F [0x57F]	1281- 1343 [1407]
TxPDO5	0x681 - 0x6BF	1665 - 1727
RxPDO5	0x781 - 0x7BF	1921- 1983
TxPDO6	0x1C1 - 0x1FF	449 - 511
RxPDO6	0x241 - 0x27F	577 - 639
TxDPO7	0x2C1 - 0x2FF	705 - 767
RxPDO7	0x341 - 0x37F	833 - 895
TxPDO8	0x3C1- 0x3FF	961 - 1023
RxPDO8	0x441 - 0x47F	1089 - 1151
TxPDO9	0x4C1 - 0x4FF	1217 - 1279
RxPDO9	0x541 - 0x57F	1345 - 1407
TxDPO10	0x5C1 - 0x5FF	1473 - 1535
RxPDO10	0x641 - 0x67F	1601- 1663
TxPDO11	0x6C1 - 0x6FF	1729 - 1791
RxPDO11	0x741 - 0x77F	1857 - 1919
SDO (Tx)	0x581 - 0x5BF [0x5FF]	1409 - 1471 [1535]
SDO (Rx)	0x601 - 0x63F [0x67F]	1537 - 1599 [1663]
Guarding / Heartbeat/ Bootup	0x701 - 0x73F [0x77F]	1793 - 1855 [1919]

Identifier List

Identifiers marked with * are given manufacturer-specific assignments on the Bus Couplers after writing index 0x5500

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
0	0	NMT	874	36A	RxPDO7*, Nd.42	1430	596	SDO Tx Nd.22
128	80	SYNC	875	36B	RxPDO7*, Nd.43	1431	597	SDO Tx Nd.23
129	81	EMCY Nd.1	876	36C	RxPDO7*, Nd.44	1432	598	SDO Tx Nd.24
130	82	EMCY Nd.2	877	36D	RxPDO7*, Nd.45	1433	599	SDO Tx Nd.25
131	83	EMCY Nd.3	878	36E	RxPDO7*, Nd.46	1434	59A	SDO Tx Nd.26
132	84	EMCY Nd.4	879	36F	RxPDO7*, Nd.47	1435	59B	SDO Tx Nd.27
133	85	EMCY Nd.5	880	370	RxPDO7*, Nd.48	1436	59C	SDO Tx Nd.28
134	86	EMCY Nd.6	881	371	RxPDO7*, Nd.49	1437	59D	SDO Tx Nd.29
135	87	EMCY Nd.7	882	372	RxPDO7*, Nd.50	1438	59E	SDO Tx Nd.30
136	88	EMCY Nd.8	883	373	RxPDO7*, Nd.51	1439	59F	SDO Tx Nd.31
137	89	EMCY Nd.9	884	374	RxPDO7*, Nd.52	1440	5A0	SDO Tx Nd.32
138	8A	EMCY Nd.10	885	375	RxPDO7*, Nd.53	1441	5A1	SDO Tx Nd.33
139	8B	EMCY Nd.11	886	376	RxPDO7*, Nd.54	1442	5A2	SDO Tx Nd.34
140	8C	EMCY Nd.12	887	377	RxPDO7*, Nd.55	1443	5A3	SDO Tx Nd.35
141	8D	EMCY Nd.13	888	378	RxPDO7*, Nd.56	1444	5A4	SDO Tx Nd.36
142	8E	EMCY Nd.14	889	379	RxPDO7*, Nd.57	1445	5A5	SDO Tx Nd.37
143	8F	EMCY Nd.15	890	37A	RxPDO7*, Nd.58	1446	5A6	SDO Tx Nd.38
144	90	EMCY Nd.16	891	37B	RxPDO7*, Nd.59	1447	5A7	SDO Tx Nd.39
145	91	EMCY Nd.17	892	37C	RxPDO7*, Nd.60	1448	5A8	SDO Tx Nd.40
146	92	EMCY Nd.18	893	37D	RxPDO7*, Nd.61	1449	5A9	SDO Tx Nd.41
147	93	EMCY Nd.19	894	37E	RxPDO7*, Nd.62	1450	5AA	SDO Tx Nd.42
148	94	EMCY Nd.20	895	37F	RxPDO7*, Nd.63	1451	5AB	SDO Tx Nd.43
149	95	EMCY Nd.21	897	381	TxPDO3*, Nd.1	1452	5AC	SDO Tx Nd.44
150	96	EMCY Nd.22	898	382	TxPDO3*, Nd.2	1453	5AD	SDO Tx Nd.45
151	97	EMCY Nd.23	899	383	TxPDO3*, Nd.3	1454	5AE	SDO Tx Nd.46
152	98	EMCY Nd.24	900	384	TxPDO3*, Nd.4	1455	5AF	SDO Tx Nd.47

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
153	99	EMCY Nd.25	901	385	TxPDO3*, Nd.5	1456	5B0	SDO Tx Nd.48
154	9A	EMCY Nd.26	902	386	TxPDO3*, Nd.6	1457	5B1	SDO Tx Nd.49
155	9B	EMCY Nd.27	903	387	TxPDO3*, Nd.7	1458	5B2	SDO Tx Nd.50
156	9C	EMCY Nd.28	904	388	TxPDO3*, Nd.8	1459	5B3	SDO Tx Nd.51
157	9D	EMCY Nd.29	905	389	TxPDO3*, Nd.9	1460	5B4	SDO Tx Nd.52
158	9E	EMCY Nd.30	906	38A	TxPDO3*, Nd.10	1461	5B5	SDO Tx Nd.53
159	9F	EMCY Nd.31	907	38B	TxPDO3*, Nd.11	1462	5B6	SDO Tx Nd.54
160	A0	EMCY Nd.32	908	38C	TxPDO3*, Nd.12	1463	5B7	SDO Tx Nd.55
161	A1	EMCY Nd.33	909	38D	TxPDO3*, Nd.13	1464	5B8	SDO Tx Nd.56
162	A2	EMCY Nd.34	910	38E	TxPDO3*, Nd.14	1465	5B9	SDO Tx Nd.57
163	A3	EMCY Nd.35	911	38F	TxPDO3*, Nd.15	1466	5BA	SDO Tx Nd.58
164	A4	EMCY Nd.36	912	390	TxPDO3*, Nd.16	1467	5BB	SDO Tx Nd.59
165	A5	EMCY Nd.37	913	391	TxPDO3*, Nd.17	1468	5BC	SDO Tx Nd.60
166	A6	EMCY Nd.38	914	392	TxPDO3*, Nd.18	1469	5BD	SDO Tx Nd.61
167	A7	EMCY Nd.39	915	393	TxPDO3*, Nd.19	1470	5BE	SDO Tx Nd.62
168	A8	EMCY Nd.40	916	394	TxPDO3*, Nd.20	1471	5BF	SDO Tx Nd.63
169	A9	EMCY Nd.41	917	395	TxPDO3*, Nd.21	1473	5C1	TxPDO10*, Nd.1
170	AA	EMCY Nd.42	918	396	TxPDO3*, Nd.22	1474	5C2	TxPDO10*, Nd.2
171	AB	EMCY Nd.43	919	397	TxPDO3*, Nd.23	1475	5C3	TxPDO10*, Nd.3
172	AC	EMCY Nd.44	920	398	TxPDO3*, Nd.24	1476	5C4	TxPDO10*, Nd.4
173	AD	EMCY Nd.45	921	399	TxPDO3*, Nd.25	1477	5C5	TxPDO10*, Nd.5
174	AE	EMCY Nd.46	922	39A	TxPDO3*, Nd.26	1478	5C6	TxPDO10*, Nd.6
175	AF	EMCY Nd.47	923	39B	TxPDO3*, Nd.27	1479	5C7	TxPDO10*, Nd.7
176	B0	EMCY Nd.48	924	39C	TxPDO3*, Nd.28	1480	5C8	TxPDO10*, Nd.8
177	B1	EMCY Nd.49	925	39D	TxPDO3*, Nd.29	1481	5C9	TxPDO10*, Nd.9
178	B2	EMCY Nd.50	926	39E	TxPDO3*, Nd.30	1482	5CA	TxPDO10*, Nd.10

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
179	B3	EMCY Nd.51	927	39F	TxPDO3*, Nd.31	1483	5CB	TxPDO10 *, Nd.11
180	B4	EMCY Nd.52	928	3A0	TxPDO3*, Nd.32	1484	5CC	TxPDO10 *, Nd.12
181	B5	EMCY Nd.53	929	3A1	TxPDO3*, Nd.33	1485	5CD	TxPDO10 *, Nd.13
182	B6	EMCY Nd.54	930	3A2	TxPDO3*, Nd.34	1486	5CE	TxPDO10 *, Nd.14
183	B7	EMCY Nd.55	931	3A3	TxPDO3*, Nd.35	1487	5CF	TxPDO10 *, Nd.15
184	B8	EMCY Nd.56	932	3A4	TxPDO3*, Nd.36	1488	5D0	TxPDO10 *, Nd.16
185	B9	EMCY Nd.57	933	3A5	TxPDO3*, Nd.37	1489	5D1	TxPDO10 *, Nd.17
186	BA	EMCY Nd.58	934	3A6	TxPDO3*, Nd.38	1490	5D2	TxPDO10 *, Nd.18
187	BB	EMCY Nd.59	935	3A7	TxPDO3*, Nd.39	1491	5D3	TxPDO10 *, Nd.19
188	BC	EMCY Nd.60	936	3A8	TxPDO3*, Nd.40	1492	5D4	TxPDO10 *, Nd.20
189	BD	EMCY Nd.61	937	3A9	TxPDO3*, Nd.41	1493	5D5	TxPDO10 *, Nd.21
190	BE	EMCY Nd.62	938	3AA	TxPDO3*, Nd.42	1494	5D6	TxPDO10 *, Nd.22
191	BF	EMCY Nd.63	939	3AB	TxPDO3*, Nd.43	1495	5D7	TxPDO10 *, Nd.23
385	181	TxPDO1, DI, Nd.1	940	3AC	TxPDO3*, Nd.44	1496	5D8	TxPDO10 *, Nd.24
386	182	TxPDO1, DI, Nd.2	941	3AD	TxPDO3*, Nd.45	1497	5D9	TxPDO10 *, Nd.25
387	183	TxPDO1, DI, Nd.3	942	3AE	TxPDO3*, Nd.46	1498	5DA	TxPDO10 *, Nd.26
388	184	TxPDO1, DI, Nd.4	943	3AF	TxPDO3*, Nd.47	1499	5DB	TxPDO10 *, Nd.27
389	185	TxPDO1, DI, Nd.5	944	3B0	TxPDO3*, Nd.48	1500	5DC	TxPDO10 *, Nd.28
390	186	TxPDO1, DI, Nd.6	945	3B1	TxPDO3*, Nd.49	1501	5DD	TxPDO10 *, Nd.29
391	187	TxPDO1, DI, Nd.7	946	3B2	TxPDO3*, Nd.50	1502	5DE	TxPDO10 *, Nd.30
392	188	TxPDO1, DI, Nd.8	947	3B3	TxPDO3*, Nd.51	1503	5DF	TxPDO10 *, Nd.31
393	189	TxPDO1, DI, Nd.9	948	3B4	TxPDO3*, Nd.52	1504	5E0	TxPDO10 *, Nd.32
394	18A	TxPDO1, DI, Nd.10	949	3B5	TxPDO3*, Nd.53	1505	5E1	TxPDO10 *, Nd.33
395	18B	TxPDO1, DI, Nd.11	950	3B6	TxPDO3*, Nd.54	1506	5E2	TxPDO10 *, Nd.34
396	18C	TxPDO1, DI, Nd.12	951	3B7	TxPDO3*, Nd.55	1507	5E3	TxPDO10 *, Nd.35
397	18D	TxPDO1, DI, Nd.13	952	3B8	TxPDO3*, Nd.56	1508	5E4	TxPDO10 *, Nd.36

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
398	18E	TxPDO1, DI, Nd.14	953	3B9	TxPDO3*, Nd.57	1509	5E5	TxPDO10 *, Nd.37
399	18F	TxPDO1, DI, Nd.15	954	3BA	TxPDO3*, Nd.58	1510	5E6	TxPDO10 *, Nd.38
400	190	TxPDO1, DI, Nd.16	955	3BB	TxPDO3*, Nd.59	1511	5E7	TxPDO10 *, Nd.39
401	191	TxPDO1, DI, Nd.17	956	3BC	TxPDO3*, Nd.60	1512	5E8	TxPDO10 *, Nd.40
402	192	TxPDO1, DI, Nd.18	957	3BD	TxPDO3*, Nd.61	1513	5E9	TxPDO10 *, Nd.41
403	193	TxPDO1, DI, Nd.19	958	3BE	TxPDO3*, Nd.62	1514	5EA	TxPDO10 *, Nd.42
404	194	TxPDO1, DI, Nd.20	959	3BF	TxPDO3*, Nd.63	1515	5EB	TxPDO10 *, Nd.43
405	195	TxPDO1, DI, Nd.21	961	3C1	TxPDO8*, Nd.1	1516	5EC	TxPDO10 *, Nd.44
406	196	TxPDO1, DI, Nd.22	962	3C2	TxPDO8*, Nd.2	1517	5ED	TxPDO10 *, Nd.45
407	197	TxPDO1, DI, Nd.23	963	3C3	TxPDO8*, Nd.3	1518	5EE	TxPDO10 *, Nd.46
408	198	TxPDO1, DI, Nd.24	964	3C4	TxPDO8*, Nd.4	1519	5EF	TxPDO10 *, Nd.47
409	199	TxPDO1, DI, Nd.25	965	3C5	TxPDO8*, Nd.5	1520	5F0	TxPDO10 *, Nd.48
410	19A	TxPDO1, DI, Nd.26	966	3C6	TxPDO8*, Nd.6	1521	5F1	TxPDO10 *, Nd.49
411	19B	TxPDO1, DI, Nd.27	967	3C7	TxPDO8*, Nd.7	1522	5F2	TxPDO10 *, Nd.50
412	19C	TxPDO1, DI, Nd.28	968	3C8	TxPDO8*, Nd.8	1523	5F3	TxPDO10 *, Nd.51
413	19D	TxPDO1, DI, Nd.29	969	3C9	TxPDO8*, Nd.9	1524	5F4	TxPDO10 *, Nd.52
414	19E	TxPDO1, DI, Nd.30	970	3CA	TxPDO8*, Nd.10	1525	5F5	TxPDO10 *, Nd.53
415	19F	TxPDO1, DI, Nd.31	971	3CB	TxPDO8*, Nd.11	1526	5F6	TxPDO10 *, Nd.54
416	1A0	TxPDO1, DI, Nd.32	972	3CC	TxPDO8*, Nd.12	1527	5F7	TxPDO10 *, Nd.55
417	1A1	TxPDO1, DI, Nd.33	973	3CD	TxPDO8*, Nd.13	1528	5F8	TxPDO10 *, Nd.56
418	1A2	TxPDO1, DI, Nd.34	974	3CE	TxPDO8*, Nd.14	1529	5F9	TxPDO10 *, Nd.57
419	1A3	TxPDO1, DI, Nd.35	975	3CF	TxPDO8*, Nd.15	1530	5FA	TxPDO10 *, Nd.58
420	1A4	TxPDO1, DI, Nd.36	976	3D0	TxPDO8*, Nd.16	1531	5FB	TxPDO10 *, Nd.59
421	1A5	TxPDO1, DI, Nd.37	977	3D1	TxPDO8*, Nd.17	1532	5FC	TxPDO10 *, Nd.60
422	1A6	TxPDO1, DI, Nd.38	978	3D2	TxPDO8*, Nd.18	1533	5FD	TxPDO10 *, Nd.61
423	1A7	TxPDO1, DI, Nd.39	979	3D3	TxPDO8*, Nd.19	1534	5FE	TxPDO10 *, Nd.62

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
424	1A8	TxPDO1, DI, Nd.40	980	3D4	TxPDO8*, Nd.20	1535	5FF	TxPDO10*, Nd.63
425	1A9	TxPDO1, DI, Nd.41	981	3D5	TxPDO8*, Nd.21	1537	601	SDO Rx Nd.1
426	1AA	TxPDO1, DI, Nd.42	982	3D6	TxPDO8*, Nd.22	1538	602	SDO Rx Nd.2
427	1AB	TxPDO1, DI, Nd.43	983	3D7	TxPDO8*, Nd.23	1539	603	SDO Rx Nd.3
428	1AC	TxPDO1, DI, Nd.44	984	3D8	TxPDO8*, Nd.24	1540	604	SDO Rx Nd.4
429	1AD	TxPDO1, DI, Nd.45	985	3D9	TxPDO8*, Nd.25	1541	605	SDO Rx Nd.5
430	1AE	TxPDO1, DI, Nd.46	986	3DA	TxPDO8*, Nd.26	1542	606	SDO Rx Nd.6
431	1AF	TxPDO1, DI, Nd.47	987	3DB	TxPDO8*, Nd.27	1543	607	SDO Rx Nd.7
432	1B0	TxPDO1, DI, Nd.48	988	3DC	TxPDO8*, Nd.28	1544	608	SDO Rx Nd.8
433	1B1	TxPDO1, DI, Nd.49	989	3DD	TxPDO8*, Nd.29	1545	609	SDO Rx Nd.9
434	1B2	TxPDO1, DI, Nd.50	990	3DE	TxPDO8*, Nd.30	1546	60A	SDO Rx Nd.10
435	1B3	TxPDO1, DI, Nd.51	991	3DF	TxPDO8*, Nd.31	1547	60B	SDO Rx Nd.11
436	1B4	TxPDO1, DI, Nd.52	992	3E0	TxPDO8*, Nd.32	1548	60C	SDO Rx Nd.12
437	1B5	TxPDO1, DI, Nd.53	993	3E1	TxPDO8*, Nd.33	1549	60D	SDO Rx Nd.13
438	1B6	TxPDO1, DI, Nd.54	994	3E2	TxPDO8*, Nd.34	1550	60E	SDO Rx Nd.14
439	1B7	TxPDO1, DI, Nd.55	995	3E3	TxPDO8*, Nd.35	1551	60F	SDO Rx Nd.15
440	1B8	TxPDO1, DI, Nd.56	996	3E4	TxPDO8*, Nd.36	1552	610	SDO Rx Nd.16
441	1B9	TxPDO1, DI, Nd.57	997	3E5	TxPDO8*, Nd.37	1553	611	SDO Rx Nd.17
442	1BA	TxPDO1, DI, Nd.58	998	3E6	TxPDO8*, Nd.38	1554	612	SDO Rx Nd.18
443	1BB	TxPDO1, DI, Nd.59	999	3E7	TxPDO8*, Nd.39	1555	613	SDO Rx Nd.19
444	1BC	TxPDO1, DI, Nd.60	1000	3E8	TxPDO8*, Nd.40	1556	614	SDO Rx Nd.20
445	1BD	TxPDO1, DI, Nd.61	1001	3E9	TxPDO8*, Nd.41	1557	615	SDO Rx Nd.21
446	1BE	TxPDO1, DI, Nd.62	1002	3EA	TxPDO8*, Nd.42	1558	616	SDO Rx Nd.22
447	1BF	TxPDO1, DI, Nd.63	1003	3EB	TxPDO8*, Nd.43	1559	617	SDO Rx Nd.23
449	1C1	TxPDO6*, Nd.1	1004	3EC	TxPDO8*, Nd.44	1560	618	SDO Rx Nd.24
450	1C2	TxPDO6*, Nd.2	1005	3ED	TxPDO8*, Nd.45	1561	619	SDO Rx Nd.25

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
451	1C3	TxPDO6*, Nd.3	1006	3EE	TxPDO8*, Nd.46	1562	61A	SDO Rx Nd.26
452	1C4	TxPDO6*, Nd.4	1007	3EF	TxPDO8*, Nd.47	1563	61B	SDO Rx Nd.27
453	1C5	TxPDO6*, Nd.5	1008	3F0	TxPDO8*, Nd.48	1564	61C	SDO Rx Nd.28
454	1C6	TxPDO6*, Nd.6	1009	3F1	TxPDO8*, Nd.49	1565	61D	SDO Rx Nd.29
455	1C7	TxPDO6*, Nd.7	1010	3F2	TxPDO8*, Nd.50	1566	61E	SDO Rx Nd.30
456	1C8	TxPDO6*, Nd.8	1011	3F3	TxPDO8*, Nd.51	1567	61F	SDO Rx Nd.31
457	1C9	TxPDO6*, Nd.9	1012	3F4	TxPDO8*, Nd.52	1568	620	SDO Rx Nd.32
458	1CA	TxPDO6*, Nd.10	1013	3F5	TxPDO8*, Nd.53	1569	621	SDO Rx Nd.33
459	1CB	TxPDO6*, Nd.11	1014	3F6	TxPDO8*, Nd.54	1570	622	SDO Rx Nd.34
460	1CC	TxPDO6*, Nd.12	1015	3F7	TxPDO8*, Nd.55	1571	623	SDO Rx Nd.35
461	1CD	TxPDO6*, Nd.13	1016	3F8	TxPDO8*, Nd.56	1572	624	SDO Rx Nd.36
462	1CE	TxPDO6*, Nd.14	1017	3F9	TxPDO8*, Nd.57	1573	625	SDO Rx Nd.37
463	1CF	TxPDO6*, Nd.15	1018	3FA	TxPDO8*, Nd.58	1574	626	SDO Rx Nd.38
464	1D0	TxPDO6*, Nd.16	1019	3FB	TxPDO8*, Nd.59	1575	627	SDO Rx Nd.39
465	1D1	TxPDO6*, Nd.17	1020	3FC	TxPDO8*, Nd.60	1576	628	SDO Rx Nd.40
466	1D2	TxPDO6*, Nd.18	1021	3FD	TxPDO8*, Nd.61	1577	629	SDO Rx Nd.41
467	1D3	TxPDO6*, Nd.19	1022	3FE	TxPDO8*, Nd.62	1578	62A	SDO Rx Nd.42
468	1D4	TxPDO6*, Nd.20	1023	3FF	TxPDO8*, Nd.63	1579	62B	SDO Rx Nd.43
469	1D5	TxPDO6*, Nd.21	1025	401	RxPDO3*, Nd.1	1580	62C	SDO Rx Nd.44
470	1D6	TxPDO6*, Nd.22	1026	402	RxPDO3*, Nd.2	1581	62D	SDO Rx Nd.45
471	1D7	TxPDO6*, Nd.23	1027	403	RxPDO3*, Nd.3	1582	62E	SDO Rx Nd.46
472	1D8	TxPDO6*, Nd.24	1028	404	RxPDO3*, Nd.4	1583	62F	SDO Rx Nd.47
473	1D9	TxPDO6*, Nd.25	1029	405	RxPDO3*, Nd.5	1584	630	SDO Rx Nd.48
474	1DA	TxPDO6*, Nd.26	1030	406	RxPDO3*, Nd.6	1585	631	SDO Rx Nd.49
475	1DB	TxPDO6*, Nd.27	1031	407	RxPDO3*, Nd.7	1586	632	SDO Rx Nd.50
476	1DC	TxPDO6*, Nd.28	1032	408	RxPDO3*, Nd.8	1587	633	SDO Rx Nd.51

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
477	1DD	TxPDO6*, Nd.29	1033	409	RxPDO3*, Nd.9	1588	634	SDO Rx Nd.52
478	1DE	TxPDO6*, Nd.30	1034	40A	RxPDO3*, Nd.10	1589	635	SDO Rx Nd.53
479	1DF	TxPDO6*, Nd.31	1035	40B	RxPDO3*, Nd.11	1590	636	SDO Rx Nd.54
480	1E0	TxPDO6*, Nd.32	1036	40C	RxPDO3*, Nd.12	1591	637	SDO Rx Nd.55
481	1E1	TxPDO6*, Nd.33	1037	40D	RxPDO3*, Nd.13	1592	638	SDO Rx Nd.56
482	1E2	TxPDO6*, Nd.34	1038	40E	RxPDO3*, Nd.14	1593	639	SDO Rx Nd.57
483	1E3	TxPDO6*, Nd.35	1039	40F	RxPDO3*, Nd.15	1594	63A	SDO Rx Nd.58
484	1E4	TxPDO6*, Nd.36	1040	410	RxPDO3*, Nd.16	1595	63B	SDO Rx Nd.59
485	1E5	TxPDO6*, Nd.37	1041	411	RxPDO3*, Nd.17	1596	63C	SDO Rx Nd.60
486	1E6	TxPDO6*, Nd.38	1042	412	RxPDO3*, Nd.18	1597	63D	SDO Rx Nd.61
487	1E7	TxPDO6*, Nd.39	1043	413	RxPDO3*, Nd.19	1598	63E	SDO Rx Nd.62
488	1E8	TxPDO6*, Nd.40	1044	414	RxPDO3*, Nd.20	1599	63F	SDO Rx Nd.63
489	1E9	TxPDO6*, Nd.41	1045	415	RxPDO3*, Nd.21	1601	641	RxPDO10*, Nd.1
490	1EA	TxPDO6*, Nd.42	1046	416	RxPDO3*, Nd.22	1602	642	RxPDO10*, Nd.2
491	1EB	TxPDO6*, Nd.43	1047	417	RxPDO3*, Nd.23	1603	643	RxPDO10*, Nd.3
492	1EC	TxPDO6*, Nd.44	1048	418	RxPDO3*, Nd.24	1604	644	RxPDO10*, Nd.4
493	1ED	TxPDO6*, Nd.45	1049	419	RxPDO3*, Nd.25	1605	645	RxPDO10*, Nd.5
494	1EE	TxPDO6*, Nd.46	1050	41A	RxPDO3*, Nd.26	1606	646	RxPDO10*, Nd.6
495	1EF	TxPDO6*, Nd.47	1051	41B	RxPDO3*, Nd.27	1607	647	RxPDO10*, Nd.7
496	1F0	TxPDO6*, Nd.48	1052	41C	RxPDO3*, Nd.28	1608	648	RxPDO10*, Nd.8
497	1F1	TxPDO6*, Nd.49	1053	41D	RxPDO3*, Nd.29	1609	649	RxPDO10*, Nd.9
498	1F2	TxPDO6*, Nd.50	1054	41E	RxPDO3*, Nd.30	1610	64A	RxPDO10*, Nd.10
499	1F3	TxPDO6*, Nd.51	1055	41F	RxPDO3*, Nd.31	1611	64B	RxPDO10*, Nd.11
500	1F4	TxPDO6*, Nd.52	1056	420	RxPDO3*, Nd.32	1612	64C	RxPDO10*, Nd.12
501	1F5	TxPDO6*, Nd.53	1057	421	RxPDO3*, Nd.33	1613	64D	RxPDO10*, Nd.13
502	1F6	TxPDO6*, Nd.54	1058	422	RxPDO3*, Nd.34	1614	64E	RxPDO10*, Nd.14

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
503	1F7	TxPDO6*, Nd.55	1059	423	RxPDO3*, Nd.35	1615	64F	RxPDO10 *, Nd.15
504	1F8	TxPDO6*, Nd.56	1060	424	RxPDO3*, Nd.36	1616	650	RxPDO10 *, Nd.16
505	1F9	TxPDO6*, Nd.57	1061	425	RxPDO3*, Nd.37	1617	651	RxPDO10 *, Nd.17
506	1FA	TxPDO6*, Nd.58	1062	426	RxPDO3*, Nd.38	1618	652	RxPDO10 *, Nd.18
507	1FB	TxPDO6*, Nd.59	1063	427	RxPDO3*, Nd.39	1619	653	RxPDO10 *, Nd.19
508	1FC	TxPDO6*, Nd.60	1064	428	RxPDO3*, Nd.40	1620	654	RxPDO10 *, Nd.20
509	1FD	TxPDO6*, Nd.61	1065	429	RxPDO3*, Nd.41	1621	655	RxPDO10 *, Nd.21
510	1FE	TxPDO6*, Nd.62	1066	42A	RxPDO3*, Nd.42	1622	656	RxPDO10 *, Nd.22
511	1FF	TxPDO6*, Nd.63	1067	42B	RxPDO3*, Nd.43	1623	657	RxPDO10 *, Nd.23
513	201	RxPDO1, DO, Nd.1	1068	42C	RxPDO3*, Nd.44	1624	658	RxPDO10 *, Nd.24
514	202	RxPDO1, DO, Nd.2	1069	42D	RxPDO3*, Nd.45	1625	659	RxPDO10 *, Nd.25
515	203	RxPDO1, DO, Nd.3	1070	42E	RxPDO3*, Nd.46	1626	65A	RxPDO10 *, Nd.26
516	204	RxPDO1, DO, Nd.4	1071	42F	RxPDO3*, Nd.47	1627	65B	RxPDO10 *, Nd.27
517	205	RxPDO1, DO, Nd.5	1072	430	RxPDO3*, Nd.48	1628	65C	RxPDO10 *, Nd.28
518	206	RxPDO1, DO, Nd.6	1073	431	RxPDO3*, Nd.49	1629	65D	RxPDO10 *, Nd.29
519	207	RxPDO1, DO, Nd.7	1074	432	RxPDO3*, Nd.50	1630	65E	RxPDO10 *, Nd.30
520	208	RxPDO1, DO, Nd.8	1075	433	RxPDO3*, Nd.51	1631	65F	RxPDO10 *, Nd.31
521	209	RxPDO1, DO, Nd.9	1076	434	RxPDO3*, Nd.52	1632	660	RxPDO10 *, Nd.32
522	20A	RxPDO1, DO, Nd.10	1077	435	RxPDO3*, Nd.53	1633	661	RxPDO10 *, Nd.33
523	20B	RxPDO1, DO, Nd.11	1078	436	RxPDO3*, Nd.54	1634	662	RxPDO10 *, Nd.34
524	20C	RxPDO1, DO, Nd.12	1079	437	RxPDO3*, Nd.55	1635	663	RxPDO10 *, Nd.35
525	20D	RxPDO1, DO, Nd.13	1080	438	RxPDO3*, Nd.56	1636	664	RxPDO10 *, Nd.36
526	20E	RxPDO1, DO, Nd.14	1081	439	RxPDO3*, Nd.57	1637	665	RxPDO10 *, Nd.37
527	20F	RxPDO1, DO, Nd.15	1082	43A	RxPDO3*, Nd.58	1638	666	RxPDO10 *, Nd.38
528	210	RxPDO1, DO, Nd.16	1083	43B	RxPDO3*, Nd.59	1639	667	RxPDO10 *, Nd.39
529	211	RxPDO1, DO, Nd.17	1084	43C	RxPDO3*, Nd.60	1640	668	RxPDO10 *, Nd.40

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
530	212	RxPDO1, DO, Nd.18	1085	43D	RxPDO3*, Nd.61	1641	669	RxPDO10*, Nd.41
531	213	RxPDO1, DO, Nd.19	1086	43E	RxPDO3*, Nd.62	1642	66A	RxPDO10*, Nd.42
532	214	RxPDO1, DO, Nd.20	1087	43F	RxPDO3*, Nd.63	1643	66B	RxPDO10*, Nd.43
533	215	RxPDO1, DO, Nd.21	1089	441	RxPDO8*, Nd.1	1644	66C	RxPDO10*, Nd.44
534	216	RxPDO1, DO, Nd.22	1090	442	RxPDO8*, Nd.2	1645	66D	RxPDO10*, Nd.45
535	217	RxPDO1, DO, Nd.23	1091	443	RxPDO8*, Nd.3	1646	66E	RxPDO10*, Nd.46
536	218	RxPDO1, DO, Nd.24	1092	444	RxPDO8*, Nd.4	1647	66F	RxPDO10*, Nd.47
537	219	RxPDO1, DO, Nd.25	1093	445	RxPDO8*, Nd.5	1648	670	RxPDO10*, Nd.48
538	21A	RxPDO1, DO, Nd.26	1094	446	RxPDO8*, Nd.6	1649	671	RxPDO10*, Nd.49
539	21B	RxPDO1, DO, Nd.27	1095	447	RxPDO8*, Nd.7	1650	672	RxPDO10*, Nd.50
540	21C	RxPDO1, DO, Nd.28	1096	448	RxPDO8*, Nd.8	1651	673	RxPDO10*, Nd.51
541	21D	RxPDO1, DO, Nd.29	1097	449	RxPDO8*, Nd.9	1652	674	RxPDO10*, Nd.52
542	21E	RxPDO1, DO, Nd.30	1098	44A	RxPDO8*, Nd.10	1653	675	RxPDO10*, Nd.53
543	21F	RxPDO1, DO, Nd.31	1099	44B	RxPDO8*, Nd.11	1654	676	RxPDO10*, Nd.54
544	220	RxPDO1, DO, Nd.32	1100	44C	RxPDO8*, Nd.12	1655	677	RxPDO10*, Nd.55
545	221	RxPDO1, DO, Nd.33	1101	44D	RxPDO8*, Nd.13	1656	678	RxPDO10*, Nd.56
546	222	RxPDO1, DO, Nd.34	1102	44E	RxPDO8*, Nd.14	1657	679	RxPDO10*, Nd.57
547	223	RxPDO1, DO, Nd.35	1103	44F	RxPDO8*, Nd.15	1658	67A	RxPDO10*, Nd.58
548	224	RxPDO1, DO, Nd.36	1104	450	RxPDO8*, Nd.16	1659	67B	RxPDO10*, Nd.59
549	225	RxPDO1, DO, Nd.37	1105	451	RxPDO8*, Nd.17	1660	67C	RxPDO10*, Nd.60
550	226	RxPDO1, DO, Nd.38	1106	452	RxPDO8*, Nd.18	1661	67D	RxPDO10*, Nd.61
551	227	RxPDO1, DO, Nd.39	1107	453	RxPDO8*, Nd.19	1662	67E	RxPDO10*, Nd.62
552	228	RxPDO1, DO, Nd.40	1108	454	RxPDO8*, Nd.20	1663	67F	RxPDO10*, Nd.63
553	229	RxPDO1, DO, Nd.41	1109	455	RxPDO8*, Nd.21	1665	681	TxPDO5*, Nd.1
554	22A	RxPDO1, DO, Nd.42	1110	456	RxPDO8*, Nd.22	1666	682	TxPDO5*, Nd.2
555	22B	RxPDO1, DO, Nd.43	1111	457	RxPDO8*, Nd.23	1667	683	TxPDO5*, Nd.3

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
556	22C	RxPDO1, DO, Nd.44	1112	458	RxPDO8*, Nd.24	1668	684	TxPDO5*, Nd.4
557	22D	RxPDO1, DO, Nd.45	1113	459	RxPDO8*, Nd.25	1669	685	TxPDO5*, Nd.5
558	22E	RxPDO1, DO, Nd.46	1114	45A	RxPDO8*, Nd.26	1670	686	TxPDO5*, Nd.6
559	22F	RxPDO1, DO, Nd.47	1115	45B	RxPDO8*, Nd.27	1671	687	TxPDO5*, Nd.7
560	230	RxPDO1, DO, Nd.48	1116	45C	RxPDO8*, Nd.28	1672	688	TxPDO5*, Nd.8
561	231	RxPDO1, DO, Nd.49	1117	45D	RxPDO8*, Nd.29	1673	689	TxPDO5*, Nd.9
562	232	RxPDO1, DO, Nd.50	1118	45E	RxPDO8*, Nd.30	1674	68A	TxPDO5*, Nd.10
563	233	RxPDO1, DO, Nd.51	1119	45F	RxPDO8*, Nd.31	1675	68B	TxPDO5*, Nd.11
564	234	RxPDO1, DO, Nd.52	1120	460	RxPDO8*, Nd.32	1676	68C	TxPDO5*, Nd.12
565	235	RxPDO1, DO, Nd.53	1121	461	RxPDO8*, Nd.33	1677	68D	TxPDO5*, Nd.13
566	236	RxPDO1, DO, Nd.54	1122	462	RxPDO8*, Nd.34	1678	68E	TxPDO5*, Nd.14
567	237	RxPDO1, DO, Nd.55	1123	463	RxPDO8*, Nd.35	1679	68F	TxPDO5*, Nd.15
568	238	RxPDO1, DO, Nd.56	1124	464	RxPDO8*, Nd.36	1680	690	TxPDO5*, Nd.16
569	239	RxPDO1, DO, Nd.57	1125	465	RxPDO8*, Nd.37	1681	691	TxPDO5*, Nd.17
570	23A	RxPDO1, DO, Nd.58	1126	466	RxPDO8*, Nd.38	1682	692	TxPDO5*, Nd.18
571	23B	RxPDO1, DO, Nd.59	1127	467	RxPDO8*, Nd.39	1683	693	TxPDO5*, Nd.19
572	23C	RxPDO1, DO, Nd.60	1128	468	RxPDO8*, Nd.40	1684	694	TxPDO5*, Nd.20
573	23D	RxPDO1, DO, Nd.61	1129	469	RxPDO8*, Nd.41	1685	695	TxPDO5*, Nd.21
574	23E	RxPDO1, DO, Nd.62	1130	46A	RxPDO8*, Nd.42	1686	696	TxPDO5*, Nd.22
575	23F	RxPDO1, DO, Nd.63	1131	46B	RxPDO8*, Nd.43	1687	697	TxPDO5*, Nd.23
577	241	RxPDO6*, Nd.1	1132	46C	RxPDO8*, Nd.44	1688	698	TxPDO5*, Nd.24
578	242	RxPDO6*, Nd.2	1133	46D	RxPDO8*, Nd.45	1689	699	TxPDO5*, Nd.25
579	243	RxPDO6*, Nd.3	1134	46E	RxPDO8*, Nd.46	1690	69A	TxPDO5*, Nd.26
580	244	RxPDO6*, Nd.4	1135	46F	RxPDO8*, Nd.47	1691	69B	TxPDO5*, Nd.27
581	245	RxPDO6*, Nd.5	1136	470	RxPDO8*, Nd.48	1692	69C	TxPDO5*, Nd.28
582	246	RxPDO6*, Nd.6	1137	471	RxPDO8*, Nd.49	1693	69D	TxPDO5*, Nd.29

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
583	247	RxPDO6*, Nd.7	1138	472	RxPDO8*, Nd.50	1694	69E	TxPDO5*, Nd.30
584	248	RxPDO6*, Nd.8	1139	473	RxPDO8*, Nd.51	1695	69F	TxPDO5*, Nd.31
585	249	RxPDO6*, Nd.9	1140	474	RxPDO8*, Nd.52	1696	6A0	TxPDO5*, Nd.32
586	24A	RxPDO6*, Nd.10	1141	475	RxPDO8*, Nd.53	1697	6A1	TxPDO5*, Nd.33
587	24B	RxPDO6*, Nd.11	1142	476	RxPDO8*, Nd.54	1698	6A2	TxPDO5*, Nd.34
588	24C	RxPDO6*, Nd.12	1143	477	RxPDO8*, Nd.55	1699	6A3	TxPDO5*, Nd.35
589	24D	RxPDO6*, Nd.13	1144	478	RxPDO8*, Nd.56	1700	6A4	TxPDO5*, Nd.36
590	24E	RxPDO6*, Nd.14	1145	479	RxPDO8*, Nd.57	1701	6A5	TxPDO5*, Nd.37
591	24F	RxPDO6*, Nd.15	1146	47A	RxPDO8*, Nd.58	1702	6A6	TxPDO5*, Nd.38
592	250	RxPDO6*, Nd.16	1147	47B	RxPDO8*, Nd.59	1703	6A7	TxPDO5*, Nd.39
593	251	RxPDO6*, Nd.17	1148	47C	RxPDO8*, Nd.60	1704	6A8	TxPDO5*, Nd.40
594	252	RxPDO6*, Nd.18	1149	47D	RxPDO8*, Nd.61	1705	6A9	TxPDO5*, Nd.41
595	253	RxPDO6*, Nd.19	1150	47E	RxPDO8*, Nd.62	1706	6AA	TxPDO5*, Nd.42
596	254	RxPDO6*, Nd.20	1151	47F	RxPDO8*, Nd.63	1707	6AB	TxPDO5*, Nd.43
597	255	RxPDO6*, Nd.21	1153	481	TxPDO4*, Nd.1	1708	6AC	TxPDO5*, Nd.44
598	256	RxPDO6*, Nd.22	1154	482	TxPDO4*, Nd.2	1709	6AD	TxPDO5*, Nd.45
599	257	RxPDO6*, Nd.23	1155	483	TxPDO4*, Nd.3	1710	6AE	TxPDO5*, Nd.46
600	258	RxPDO6*, Nd.24	1156	484	TxPDO4*, Nd.4	1711	6AF	TxPDO5*, Nd.47
601	259	RxPDO6*, Nd.25	1157	485	TxPDO4*, Nd.5	1712	6B0	TxPDO5*, Nd.48
602	25A	RxPDO6*, Nd.26	1158	486	TxPDO4*, Nd.6	1713	6B1	TxPDO5*, Nd.49
603	25B	RxPDO6*, Nd.27	1159	487	TxPDO4*, Nd.7	1714	6B2	TxPDO5*, Nd.50
604	25C	RxPDO6*, Nd.28	1160	488	TxPDO4*, Nd.8	1715	6B3	TxPDO5*, Nd.51
605	25D	RxPDO6*, Nd.29	1161	489	TxPDO4*, Nd.9	1716	6B4	TxPDO5*, Nd.52
606	25E	RxPDO6*, Nd.30	1162	48A	TxPDO4*, Nd.10	1717	6B5	TxPDO5*, Nd.53
607	25F	RxPDO6*, Nd.31	1163	48B	TxPDO4*, Nd.11	1718	6B6	TxPDO5*, Nd.54
608	260	RxPDO6*, Nd.32	1164	48C	TxPDO4*, Nd.12	1719	6B7	TxPDO5*, Nd.55

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
609	261	RxPDO6*, Nd.33	1165	48D	TxPDO4*, Nd.13	1720	6B8	TxPDO5*, Nd.56
610	262	RxPDO6*, Nd.34	1166	48E	TxPDO4*, Nd.14	1721	6B9	TxPDO5*, Nd.57
611	263	RxPDO6*, Nd.35	1167	48F	TxPDO4*, Nd.15	1722	6BA	TxPDO5*, Nd.58
612	264	RxPDO6*, Nd.36	1168	490	TxPDO4*, Nd.16	1723	6BB	TxPDO5*, Nd.59
613	265	RxPDO6*, Nd.37	1169	491	TxPDO4*, Nd.17	1724	6BC	TxPDO5*, Nd.60
614	266	RxPDO6*, Nd.38	1170	492	TxPDO4*, Nd.18	1725	6BD	TxPDO5*, Nd.61
615	267	RxPDO6*, Nd.39	1171	493	TxPDO4*, Nd.19	1726	6BE	TxPDO5*, Nd.62
616	268	RxPDO6*, Nd.40	1172	494	TxPDO4*, Nd.20	1727	6BF	TxPDO5*, Nd.63
617	269	RxPDO6*, Nd.41	1173	495	TxPDO4*, Nd.21	1729	6C1	TxPDO11 *, Nd.1
618	26A	RxPDO6*, Nd.42	1174	496	TxPDO4*, Nd.22	1730	6C2	TxPDO11 *, Nd.2
619	26B	RxPDO6*, Nd.43	1175	497	TxPDO4*, Nd.23	1731	6C3	TxPDO11 *, Nd.3
620	26C	RxPDO6*, Nd.44	1176	498	TxPDO4*, Nd.24	1732	6C4	TxPDO11 *, Nd.4
621	26D	RxPDO6*, Nd.45	1177	499	TxPDO4*, Nd.25	1733	6C5	TxPDO11 *, Nd.5
622	26E	RxPDO6*, Nd.46	1178	49A	TxPDO4*, Nd.26	1734	6C6	TxPDO11 *, Nd.6
623	26F	RxPDO6*, Nd.47	1179	49B	TxPDO4*, Nd.27	1735	6C7	TxPDO11 *, Nd.7
624	270	RxPDO6*, Nd.48	1180	49C	TxPDO4*, Nd.28	1736	6C8	TxPDO11 *, Nd.8
625	271	RxPDO6*, Nd.49	1181	49D	TxPDO4*, Nd.29	1737	6C9	TxPDO11 *, Nd.9
626	272	RxPDO6*, Nd.50	1182	49E	TxPDO4*, Nd.30	1738	6CA	TxPDO11 *, Nd.10
627	273	RxPDO6*, Nd.51	1183	49F	TxPDO4*, Nd.31	1739	6CB	TxPDO11 *, Nd.11
628	274	RxPDO6*, Nd.52	1184	4A0	TxPDO4*, Nd.32	1740	6CC	TxPDO11 *, Nd.12
629	275	RxPDO6*, Nd.53	1185	4A1	TxPDO4*, Nd.33	1741	6CD	TxPDO11 *, Nd.13
630	276	RxPDO6*, Nd.54	1186	4A2	TxPDO4*, Nd.34	1742	6CE	TxPDO11 *, Nd.14
631	277	RxPDO6*, Nd.55	1187	4A3	TxPDO4*, Nd.35	1743	6CF	TxPDO11 *, Nd.15
632	278	RxPDO6*, Nd.56	1188	4A4	TxPDO4*, Nd.36	1744	6D0	TxPDO11 *, Nd.16
633	279	RxPDO6*, Nd.57	1189	4A5	TxPDO4*, Nd.37	1745	6D1	TxPDO11 *, Nd.17
634	27A	RxPDO6*, Nd.58	1190	4A6	TxPDO4*, Nd.48	1746	6D2	TxPDO11 *, Nd.18

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
635	27B	RxPDO6*, Nd.59	1191	4A7	TxPDO4*, Nd.49	1747	6D3	TxPDO11 *, Nd.19
636	27C	RxPDO6*, Nd.60	1192	4A8	TxPDO4*, Nd.40	1748	6D4	TxPDO11 *, Nd.20
637	27D	RxPDO6*, Nd.61	1193	4A9	TxPDO4*, Nd.41	1749	6D5	TxPDO11 *, Nd.21
638	27E	RxPDO6*, Nd.62	1194	4AA	TxPDO4*, Nd.42	1750	6D6	TxPDO11 *, Nd.22
639	27F	RxPDO6*, Nd.63	1195	4AB	TxPDO4*, Nd.43	1751	6D7	TxPDO11 *, Nd.23
641	281	TxPDO2, AI, Nd.1	1196	4AC	TxPDO4*, Nd.44	1752	6D8	TxPDO11 *, Nd.24
642	282	TxPDO2, AI, Nd.2	1197	4AD	TxPDO4*, Nd.45	1753	6D9	TxPDO11 *, Nd.25
643	283	TxPDO2, AI, Nd.3	1198	4AE	TxPDO4*, Nd.46	1754	6DA	TxPDO11 *, Nd.26
644	284	TxPDO2, AI, Nd.4	1199	4AF	TxPDO4*, Nd.47	1755	6DB	TxPDO11 *, Nd.27
645	285	TxPDO2, AI, Nd.5	1200	4B0	TxPDO4*, Nd.48	1756	6DC	TxPDO11 *, Nd.28
646	286	TxPDO2, AI, Nd.6	1201	4B1	TxPDO4*, Nd.49	1757	6DD	TxPDO11 *, Nd.29
647	287	TxPDO2, AI, Nd.7	1202	4B2	TxPDO4*, Nd.50	1758	6DE	TxPDO11 *, Nd.30
648	288	TxPDO2, AI, Nd.8	1203	4B3	TxPDO4*, Nd.51	1759	6DF	TxPDO11 *, Nd.31
649	289	TxPDO2, AI, Nd.9	1204	4B4	TxPDO4*, Nd.52	1760	6E0	TxPDO11 *, Nd.32
650	28A	TxPDO2, AI, Nd.10	1205	4B5	TxPDO4*, Nd.53	1761	6E1	TxPDO11 *, Nd.33
651	28B	TxPDO2, AI, Nd.11	1206	4B6	TxPDO4*, Nd.54	1762	6E2	TxPDO11 *, Nd.34
652	28C	TxPDO2, AI, Nd.12	1207	4B7	TxPDO4*, Nd.55	1763	6E3	TxPDO11 *, Nd.35
653	28D	TxPDO2, AI, Nd.13	1208	4B8	TxPDO4*, Nd.56	1764	6E4	TxPDO11 *, Nd.36
654	28E	TxPDO2, AI, Nd.14	1209	4B9	TxPDO4*, Nd.57	1765	6E5	TxPDO11 *, Nd.37
655	28F	TxPDO2, AI, Nd.15	1210	4BA	TxPDO4*, Nd.58	1766	6E6	TxPDO11 *, Nd.38
656	290	TxPDO2, AI, Nd.16	1211	4BB	TxPDO4*, Nd.59	1767	6E7	TxPDO11 *, Nd.39
657	291	TxPDO2, AI, Nd.17	1212	4BC	TxPDO4*, Nd.60	1768	6E8	TxPDO11 *, Nd.40
658	292	TxPDO2, AI, Nd.18	1213	4BD	TxPDO4*, Nd.61	1769	6E9	TxPDO11 *, Nd.41
659	293	TxPDO2, AI, Nd.19	1214	4BE	TxPDO4*, Nd.62	1770	6EA	TxPDO11 *, Nd.42
660	294	TxPDO2, AI, Nd.20	1215	4BF	TxPDO4*, Nd.63	1771	6EB	TxPDO11 *, Nd.43
661	295	TxPDO2, AI, Nd.21	1217	4C1	TxPDO9*, Nd.1	1772	6EC	TxPDO11 *, Nd.44

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
662	296	TxPDO2, AI, Nd.22	1218	4C2	TxPDO9*, Nd.2	1773	6ED	TxPDO11 *, Nd.45
663	297	TxPDO2, AI, Nd.23	1219	4C3	TxPDO9*, Nd.3	1774	6EE	TxPDO11 *, Nd.46
664	298	TxPDO2, AI, Nd.24	1220	4C4	TxPDO9*, Nd.4	1775	6EF	TxPDO11 *, Nd.47
665	299	TxPDO2, AI, Nd.25	1221	4C5	TxPDO9*, Nd.5	1776	6F0	TxPDO11 *, Nd.48
666	29A	TxPDO2, AI, Nd.26	1222	4C6	TxPDO9*, Nd.6	1777	6F1	TxPDO11 *, Nd.49
667	29B	TxPDO2, AI, Nd.27	1223	4C7	TxPDO9*, Nd.7	1778	6F2	TxPDO11 *, Nd.50
668	29C	TxPDO2, AI, Nd.28	1224	4C8	TxPDO9*, Nd.8	1779	6F3	TxPDO11 *, Nd.51
669	29D	TxPDO2, AI, Nd.29	1225	4C9	TxPDO9*, Nd.9	1780	6F4	TxPDO11 *, Nd.52
670	29E	TxPDO2, AI, Nd.30	1226	4CA	TxPDO9*, Nd.10	1781	6F5	TxPDO11 *, Nd.53
671	29F	TxPDO2, AI, Nd.31	1227	4CB	TxPDO9*, Nd.11	1782	6F6	TxPDO11 *, Nd.54
672	2A0	TxPDO2, AI, Nd.32	1228	4CC	TxPDO9*, Nd.12	1783	6F7	TxPDO11 *, Nd.55
673	2A1	TxPDO2, AI, Nd.33	1229	4CD	TxPDO9*, Nd.13	1784	6F8	TxPDO11 *, Nd.56
674	2A2	TxPDO2, AI, Nd.34	1230	4CE	TxPDO9*, Nd.14	1785	6F9	TxPDO11 *, Nd.57
675	2A3	TxPDO2, AI, Nd.35	1231	4CF	TxPDO9*, Nd.15	1786	6FA	TxPDO11 *, Nd.58
676	2A4	TxPDO2, AI, Nd.36	1232	4D0	TxPDO9*, Nd.16	1787	6FB	TxPDO11 *, Nd.59
677	2A5	TxPDO2, AI, Nd.37	1233	4D1	TxPDO9*, Nd.17	1788	6FC	TxPDO11 *, Nd.60
678	2A6	TxPDO2, AI, Nd.38	1234	4D2	TxPDO9*, Nd.18	1789	6FD	TxPDO11 *, Nd.61
679	2A7	TxPDO2, AI, Nd.39	1235	4D3	TxPDO9*, Nd.19	1790	6FE	TxPDO11 *, Nd.62
680	2A8	TxPDO2, AI, Nd.40	1236	4D4	TxPDO9*, Nd.20	1791	6FF	TxPDO11 *, Nd.63
681	2A9	TxPDO2, AI, Nd.41	1237	4D5	TxPDO9*, Nd.21	1793	701	Guarding Nd.1
682	2AA	TxPDO2, AI, Nd.42	1238	4D6	TxPDO9*, Nd.22	1794	702	Guarding Nd.2
683	2AB	TxPDO2, AI, Nd.43	1239	4D7	TxPDO9*, Nd.23	1795	703	Guarding Nd.3
684	2AC	TxPDO2, AI, Nd.44	1240	4D8	TxPDO9*, Nd.24	1796	704	Guarding Nd.4
685	2AD	TxPDO2, AI, Nd.45	1241	4D9	TxPDO9*, Nd.25	1797	705	Guarding Nd.5
686	2AE	TxPDO2, AI, Nd.46	1242	4DA	TxPDO9*, Nd.26	1798	706	Guarding Nd.6
687	2AF	TxPDO2, AI, Nd.47	1243	4DB	TxPDO9*, Nd.27	1799	707	Guarding Nd.7

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
688	2B0	TxPDO2, AI, Nd.48	1244	4DC	TxPDO9*, Nd.28	1800	708	Guarding Nd.8
689	2B1	TxPDO2, AI, Nd.49	1245	4DD	TxPDO9*, Nd.29	1801	709	Guarding Nd.9
690	2B2	TxPDO2, AI, Nd.50	1246	4DE	TxPDO9*, Nd.30	1802	70A	Guarding Nd.10
691	2B3	TxPDO2, AI, Nd.51	1247	4DF	TxPDO9*, Nd.31	1803	70B	Guarding Nd.11
692	2B4	TxPDO2, AI, Nd.52	1248	4E0	TxPDO9*, Nd.32	1804	70C	Guarding Nd.12
693	2B5	TxPDO2, AI, Nd.53	1249	4E1	TxPDO9*, Nd.33	1805	70D	Guarding Nd.13
694	2B6	TxPDO2, AI, Nd.54	1250	4E2	TxPDO9*, Nd.34	1806	70E	Guarding Nd.14
695	2B7	TxPDO2, AI, Nd.55	1251	4E3	TxPDO9*, Nd.35	1807	70F	Guarding Nd.15
696	2B8	TxPDO2, AI, Nd.56	1252	4E4	TxPDO9*, Nd.36	1808	710	Guarding Nd.16
697	2B9	TxPDO2, AI, Nd.57	1253	4E5	TxPDO9*, Nd.37	1809	711	Guarding Nd.17
698	2BA	TxPDO2, AI, Nd.58	1254	4E6	TxPDO9*, Nd.38	1810	712	Guarding Nd.18
699	2BB	TxPDO2, AI, Nd.59	1255	4E7	TxPDO9*, Nd.39	1811	713	Guarding Nd.19
700	2BC	TxPDO2, AI, Nd.60	1256	4E8	TxPDO9*, Nd.40	1812	714	Guarding Nd.20
701	2BD	TxPDO2, AI, Nd.61	1257	4E9	TxPDO9*, Nd.41	1813	715	Guarding Nd.21
702	2BE	TxPDO2, AI, Nd.62	1258	4EA	TxPDO9*, Nd.42	1814	716	Guarding Nd.22
703	2BF	TxPDO2, AI, Nd.63	1259	4EB	TxPDO9*, Nd.43	1815	717	Guarding Nd.23
705	2C1	TxPDO7*, Nd.1	1260	4EC	TxPDO9*, Nd.44	1816	718	Guarding Nd.24
706	2C2	TxPDO7*, Nd.2	1261	4ED	TxPDO9*, Nd.45	1817	719	Guarding Nd.25
707	2C3	TxPDO7*, Nd.3	1262	4EE	TxPDO9*, Nd.46	1818	71A	Guarding Nd.26
708	2C4	TxPDO7*, Nd.4	1263	4EF	TxPDO9*, Nd.47	1819	71B	Guarding Nd.27
709	2C5	TxPDO7*, Nd.5	1264	4F0	TxPDO9*, Nd.48	1820	71C	Guarding Nd.28
710	2C6	TxPDO7*, Nd.6	1265	4F1	TxPDO9*, Nd.49	1821	71D	Guarding Nd.29
711	2C7	TxPDO7*, Nd.7	1266	4F2	TxPDO9*, Nd.50	1822	71E	Guarding Nd.30
712	2C8	TxPDO7*, Nd.8	1267	4F3	TxPDO9*, Nd.51	1823	71F	Guarding Nd.31
713	2C9	TxPDO7*, Nd.9	1268	4F4	TxPDO9*, Nd.52	1824	720	Guarding Nd.32
714	2CA	TxPDO7*, Nd.10	1269	4F5	TxPDO9*, Nd.53	1825	721	Guarding Nd.33

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
715	2CB	TxPDO7*, Nd.11	1270	4F6	TxPDO9*, Nd.54	1826	722	Guarding Nd.34
716	2CC	TxPDO7*, Nd.12	1271	4F7	TxPDO9*, Nd.55	1827	723	Guarding Nd.35
717	2CD	TxPDO7*, Nd.13	1272	4F8	TxPDO9*, Nd.56	1828	724	Guarding Nd.36
718	2CE	TxPDO7*, Nd.14	1273	4F9	TxPDO9*, Nd.57	1829	725	Guarding Nd.37
719	2CF	TxPDO7*, Nd.15	1274	4FA	TxPDO9*, Nd.58	1830	726	Guarding Nd.38
720	2D0	TxPDO7*, Nd.16	1275	4FB	TxPDO9*, Nd.59	1831	727	Guarding Nd.39
721	2D1	TxPDO7*, Nd.17	1276	4FC	TxPDO9*, Nd.60	1832	728	Guarding Nd.40
722	2D2	TxPDO7*, Nd.18	1277	4FD	TxPDO9*, Nd.61	1833	729	Guarding Nd.41
723	2D3	TxPDO7*, Nd.19	1278	4FE	TxPDO9*, Nd.62	1834	72A	Guarding Nd.42
724	2D4	TxPDO7*, Nd.20	1279	4FF	TxPDO9*, Nd.63	1835	72B	Guarding Nd.43
725	2D5	TxPDO7*, Nd.21	1281	501	RxPDO4*, Nd.1	1836	72C	Guarding Nd.44
726	2D6	TxPDO7*, Nd.22	1282	502	RxPDO4*, Nd.2	1837	72D	Guarding Nd.45
727	2D7	TxPDO7*, Nd.23	1283	503	RxPDO4*, Nd.3	1838	72E	Guarding Nd.46
728	2D8	TxPDO7*, Nd.24	1284	504	RxPDO4*, Nd.4	1839	72F	Guarding Nd.47
729	2D9	TxPDO7*, Nd.25	1285	505	RxPDO4*, Nd.5	1840	730	Guarding Nd.48
730	2DA	TxPDO7*, Nd.26	1286	506	RxPDO4*, Nd.6	1841	731	Guarding Nd.49
731	2DB	TxPDO7*, Nd.27	1287	507	RxPDO4*, Nd.7	1842	732	Guarding Nd.50
732	2DC	TxPDO7*, Nd.28	1288	508	RxPDO4*, Nd.8	1843	733	Guarding Nd.51
733	2DD	TxPDO7*, Nd.29	1289	509	RxPDO4*, Nd.9	1844	734	Guarding Nd.52
734	2DE	TxPDO7*, Nd.30	1290	50A	RxPDO4*, Nd.10	1845	735	Guarding Nd.53
735	2DF	TxPDO7*, Nd.31	1291	50B	RxPDO4*, Nd.11	1846	736	Guarding Nd.54
736	2E0	TxPDO7*, Nd.32	1292	50C	RxPDO4*, Nd.12	1847	737	Guarding Nd.55
737	2E1	TxPDO7*, Nd.33	1293	50D	RxPDO4*, Nd.13	1848	738	Guarding Nd.56
738	2E2	TxPDO7*, Nd.34	1294	50E	RxPDO4*, Nd.14	1849	739	Guarding Nd.57
739	2E3	TxPDO7*, Nd.35	1295	50F	RxPDO4*, Nd.15	1850	73A	Guarding Nd.58
740	2E4	TxPDO7*, Nd.36	1296	510	RxPDO4*, Nd.16	1851	73B	Guarding Nd.59

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
741	2E5	TxPDO7*, Nd.37	1297	511	RxPDO4*, Nd.17	1852	73C	Guarding Nd.60
742	2E6	TxPDO7*, Nd.38	1298	512	RxPDO4*, Nd.18	1853	73D	Guarding Nd.61
743	2E7	TxPDO7*, Nd.39	1299	513	RxPDO4*, Nd.19	1854	73E	Guarding Nd.62
744	2E8	TxPDO7*, Nd.40	1300	514	RxPDO4*, Nd.20	1855	73F	Guarding Nd.63
745	2E9	TxPDO7*, Nd.41	1301	515	RxPDO4*, Nd.21	1857	741	RxPDO11 *, Nd.1
746	2EA	TxPDO7*, Nd.42	1302	516	RxPDO4*, Nd.22	1858	742	RxPDO11 *, Nd.2
747	2EB	TxPDO7*, Nd.43	1303	517	RxPDO4*, Nd.23	1859	743	RxPDO11 *, Nd.3
748	2EC	TxPDO7*, Nd.44	1304	518	RxPDO4*, Nd.24	1860	744	RxPDO11 *, Nd.4
749	2ED	TxPDO7*, Nd.45	1305	519	RxPDO4*, Nd.25	1861	745	RxPDO11 *, Nd.5
750	2EE	TxPDO7*, Nd.46	1306	51A	RxPDO4*, Nd.26	1862	746	RxPDO11 *, Nd.6
751	2EF	TxPDO7*, Nd.47	1307	51B	RxPDO4*, Nd.27	1863	747	RxPDO11 *, Nd.7
752	2F0	TxPDO7*, Nd.48	1308	51C	RxPDO4*, Nd.28	1864	748	RxPDO11 *, Nd.8
753	2F1	TxPDO7*, Nd.49	1309	51D	RxPDO4*, Nd.29	1865	749	RxPDO11 *, Nd.9
754	2F2	TxPDO7*, Nd.50	1310	51E	RxPDO4*, Nd.30	1866	74A	RxPDO11 *, Nd.10
755	2F3	TxPDO7*, Nd.51	1311	51F	RxPDO4*, Nd.31	1867	74B	RxPDO11 *, Nd.11
756	2F4	TxPDO7*, Nd.52	1312	520	RxPDO4*, Nd.32	1868	74C	RxPDO11 *, Nd.12
757	2F5	TxPDO7*, Nd.53	1313	521	RxPDO4*, Nd.33	1869	74D	RxPDO11 *, Nd.13
758	2F6	TxPDO7*, Nd.54	1314	522	RxPDO4*, Nd.34	1870	74E	RxPDO11 *, Nd.14
759	2F7	TxPDO7*, Nd.55	1315	523	RxPDO4*, Nd.35	1871	74F	RxPDO11 *, Nd.15
760	2F8	TxPDO7*, Nd.56	1316	524	RxPDO4*, Nd.36	1872	750	RxPDO11 *, Nd.16
761	2F9	TxPDO7*, Nd.57	1317	525	RxPDO4*, Nd.37	1873	751	RxPDO11 *, Nd.17
762	2FA	TxPDO7*, Nd.58	1318	526	RxPDO4*, Nd.38	1874	752	RxPDO11 *, Nd.18
763	2FB	TxPDO7*, Nd.59	1319	527	RxPDO4*, Nd.39	1875	753	RxPDO11 *, Nd.19
764	2FC	TxPDO7*, Nd.60	1320	528	RxPDO4*, Nd.40	1876	754	RxPDO11 *, Nd.20
765	2FD	TxPDO7*, Nd.61	1321	529	RxPDO4*, Nd.41	1877	755	RxPDO11 *, Nd.21
766	2FE	TxPDO7*, Nd.62	1322	52A	RxPDO4*, Nd.42	1878	756	RxPDO11 *, Nd.22

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
767	2FF	TxPDO7*, Nd.63	1323	52B	RxPDO4*, Nd.43	1879	757	RxPDO11 *, Nd.23
769	301	RxPDO2, AO, Nd.1	1324	52C	RxPDO4*, Nd.44	1880	758	RxPDO11 *, Nd.24
770	302	RxPDO2, AO, Nd.2	1325	52D	RxPDO4*, Nd.45	1881	759	RxPDO11 *, Nd.25
771	303	RxPDO2, AO, Nd.3	1326	52E	RxPDO4*, Nd.46	1882	75A	RxPDO11 *, Nd.26
772	304	RxPDO2, AO, Nd.4	1327	52F	RxPDO4*, Nd.47	1883	75B	RxPDO11 *, Nd.27
773	305	RxPDO2, AO, Nd.5	1328	530	RxPDO4*, Nd.48	1884	75C	RxPDO11 *, Nd.28
774	306	RxPDO2, AO, Nd.6	1329	531	RxPDO4*, Nd.49	1885	75D	RxPDO11 *, Nd.29
775	307	RxPDO2, AO, Nd.7	1330	532	RxPDO4*, Nd.50	1886	75E	RxPDO11 *, Nd.30
776	308	RxPDO2, AO, Nd.8	1331	533	RxPDO4*, Nd.51	1887	75F	RxPDO11 *, Nd.31
777	309	RxPDO2, AO, Nd.9	1332	534	RxPDO4*, Nd.52	1888	760	RxPDO11 *, Nd.32
778	30A	RxPDO2, AO, Nd.10	1333	535	RxPDO4*, Nd.53	1889	761	RxPDO11 *, Nd.33
779	30B	RxPDO2, AO, Nd.11	1334	536	RxPDO4*, Nd.54	1890	762	RxPDO11 *, Nd.34
780	30C	RxPDO2, AO, Nd.12	1335	537	RxPDO4*, Nd.55	1891	763	RxPDO11 *, Nd.35
781	30D	RxPDO2, AO, Nd.13	1336	538	RxPDO4*, Nd.56	1892	764	RxPDO11 *, Nd.36
782	30E	RxPDO2, AO, Nd.14	1337	539	RxPDO4*, Nd.57	1893	765	RxPDO11 *, Nd.37
783	30F	RxPDO2, AO, Nd.15	1338	53A	RxPDO4*, Nd.58	1894	766	RxPDO11 *, Nd.38
784	310	RxPDO2, AO, Nd.16	1339	53B	RxPDO4*, Nd.59	1895	767	RxPDO11 *, Nd.39
785	311	RxPDO2, AO, Nd.17	1340	53C	RxPDO4*, Nd.60	1896	768	RxPDO11 *, Nd.40
786	312	RxPDO2, AO, Nd.18	1341	53D	RxPDO4*, Nd.61	1897	769	RxPDO11 *, Nd.41
787	313	RxPDO2, AO, Nd.19	1342	53E	RxPDO4*, Nd.62	1898	76A	RxPDO11 *, Nd.42
788	314	RxPDO2, AO, Nd.20	1343	53F	RxPDO4*, Nd.63	1899	76B	RxPDO11 *, Nd.43
789	315	RxPDO2, AO, Nd.21	1345	541	RxPDO9*, Nd.1	1900	76C	RxPDO11 *, Nd.44
790	316	RxPDO2, AO, Nd.22	1346	542	RxPDO9*, Nd.2	1901	76D	RxPDO11 *, Nd.45
791	317	RxPDO2, AO, Nd.23	1347	543	RxPDO9*, Nd.3	1902	76E	RxPDO11 *, Nd.46
792	318	RxPDO2, AO, Nd.24	1348	544	RxPDO9*, Nd.4	1903	76F	RxPDO11 *, Nd.47
793	319	RxPDO2, AO, Nd.25	1349	545	RxPDO9*, Nd.5	1904	770	RxPDO11 *, Nd.48

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
794	31A	RxPDO2, AO, Nd.26	1350	546	RxPDO9*, Nd.6	1905	771	RxPDO11 *, Nd.49
795	31B	RxPDO2, AO, Nd.27	1351	547	RxPDO9*, Nd.7	1906	772	RxPDO11 *, Nd.50
796	31C	RxPDO2, AO, Nd.28	1352	548	RxPDO9*, Nd.8	1907	773	RxPDO11 *, Nd.51
797	31D	RxPDO2, AO, Nd.29	1353	549	RxPDO9*, Nd.9	1908	774	RxPDO11 *, Nd.52
798	31E	RxPDO2, AO, Nd.30	1354	54A	RxPDO9*, Nd.10	1909	775	RxPDO11 *, Nd.53
799	31F	RxPDO2, AO, Nd.31	1355	54B	RxPDO9*, Nd.11	1910	776	RxPDO11 *, Nd.54
800	320	RxPDO2, AO, Nd.32	1356	54C	RxPDO9*, Nd.12	1911	777	RxPDO11 *, Nd.55
801	321	RxPDO2, AO, Nd.33	1357	54D	RxPDO9*, Nd.13	1912	778	RxPDO11 *, Nd.56
802	322	RxPDO2, AO, Nd.34	1358	54E	RxPDO9*, Nd.14	1913	779	RxPDO11 *, Nd.57
803	323	RxPDO2, AO, Nd.35	1359	54F	RxPDO9*, Nd.15	1914	77A	RxPDO11 *, Nd.58
804	324	RxPDO2, AO, Nd.36	1360	550	RxPDO9*, Nd.16	1915	77B	RxPDO11 *, Nd.59
805	325	RxPDO2, AO, Nd.37	1361	551	RxPDO9*, Nd.17	1916	77C	RxPDO11 *, Nd.60
806	326	RxPDO2, AO, Nd.38	1362	552	RxPDO9*, Nd.18	1917	77D	RxPDO11 *, Nd.61
807	327	RxPDO2, AO, Nd.39	1363	553	RxPDO9*, Nd.19	1918	77E	RxPDO11 *, Nd.62
808	328	RxPDO2, AO, Nd.40	1364	554	RxPDO9*, Nd.20	1919	77F	RxPDO11 *, Nd.63
809	329	RxPDO2, AO, Nd.41	1365	555	RxPDO9*, Nd.21	1921	781	RxPDO5*, Nd.1
810	32A	RxPDO2, AO, Nd.42	1366	556	RxPDO9*, Nd.22	1922	782	RxPDO5*, Nd.2
811	32B	RxPDO2, AO, Nd.43	1367	557	RxPDO9*, Nd.23	1923	783	RxPDO5*, Nd.3
812	32C	RxPDO2, AO, Nd.44	1368	558	RxPDO9*, Nd.24	1924	784	RxPDO5*, Nd.4
813	32D	RxPDO2, AO, Nd.45	1369	559	RxPDO9*, Nd.25	1925	785	RxPDO5*, Nd.5
814	32E	RxPDO2, AO, Nd.46	1370	55A	RxPDO9*, Nd.26	1926	786	RxPDO5*, Nd.6
815	32F	RxPDO2, AO, Nd.47	1371	55B	RxPDO9*, Nd.27	1927	787	RxPDO5*, Nd.7
816	330	RxPDO2, AO, Nd.48	1372	55C	RxPDO9*, Nd.28	1928	788	RxPDO5*, Nd.8
817	331	RxPDO2, AO, Nd.49	1373	55D	RxPDO9*, Nd.29	1929	789	RxPDO5*, Nd.9
818	332	RxPDO2, AO, Nd.50	1374	55E	RxPDO9*, Nd.30	1930	78A	RxPDO5*, Nd.10
819	333	RxPDO2, AO, Nd.51	1375	55F	RxPDO9*, Nd.31	1931	78B	RxPDO5*, Nd.11

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
820	334	RxPDO2, AO, Nd.52	1376	560	RxPDO9*, Nd.32	1932	78C	RxPDO5*, Nd.12
821	335	RxPDO2, AO, Nd.53	1377	561	RxPDO9*, Nd.33	1933	78D	RxPDO5*, Nd.13
822	336	RxPDO2, AO, Nd.54	1378	562	RxPDO9*, Nd.34	1934	78E	RxPDO5*, Nd.14
823	337	RxPDO2, AO, Nd.55	1379	563	RxPDO9*, Nd.35	1935	78F	RxPDO5*, Nd.15
824	338	RxPDO2, AO, Nd.56	1380	564	RxPDO9*, Nd.36	1936	790	RxPDO5*, Nd.16
825	339	RxPDO2, AO, Nd.57	1381	565	RxPDO9*, Nd.37	1937	791	RxPDO5*, Nd.17
826	33A	RxPDO2, AO, Nd.58	1382	566	RxPDO9*, Nd.38	1938	792	RxPDO5*, Nd.18
827	33B	RxPDO2, AO, Nd.59	1383	567	RxPDO9*, Nd.39	1939	793	RxPDO5*, Nd.19
828	33C	RxPDO2, AO, Nd.60	1384	568	RxPDO9*, Nd.40	1940	794	RxPDO5*, Nd.20
829	33D	RxPDO2, AO, Nd.61	1385	569	RxPDO9*, Nd.41	1941	795	RxPDO5*, Nd.21
830	33E	RxPDO2, AO, Nd.62	1386	56A	RxPDO9*, Nd.42	1942	796	RxPDO5*, Nd.22
831	33F	RxPDO2, AO, Nd.63	1387	56B	RxPDO9*, Nd.43	1943	797	RxPDO5*, Nd.23
833	341	RxPDO7*, Nd.1	1388	56C	RxPDO9*, Nd.44	1944	798	RxPDO5*, Nd.24
834	342	RxPDO7*, Nd.2	1389	56D	RxPDO9*, Nd.45	1945	799	RxPDO5*, Nd.25
835	343	RxPDO7*, Nd.3	1390	56E	RxPDO9*, Nd.46	1946	79A	RxPDO5*, Nd.26
836	344	RxPDO7*, Nd.4	1391	56F	RxPDO9*, Nd.47	1947	79B	RxPDO5*, Nd.27
837	345	RxPDO7*, Nd.5	1392	570	RxPDO9*, Nd.48	1948	79C	RxPDO5*, Nd.28
838	346	RxPDO7*, Nd.6	1393	571	RxPDO9*, Nd.49	1949	79D	RxPDO5*, Nd.29
839	347	RxPDO7*, Nd.7	1394	572	RxPDO9*, Nd.50	1950	79E	RxPDO5*, Nd.30
840	348	RxPDO7*, Nd.8	1395	573	RxPDO9*, Nd.51	1951	79F	RxPDO5*, Nd.31
841	349	RxPDO7*, Nd.9	1396	574	RxPDO9*, Nd.52	1952	7A0	RxPDO5*, Nd.32
842	34A	RxPDO7*, Nd.10	1397	575	RxPDO9*, Nd.53	1953	7A1	RxPDO5*, Nd.33
843	34B	RxPDO7*, Nd.11	1398	576	RxPDO9*, Nd.54	1954	7A2	RxPDO5*, Nd.34
844	34C	RxPDO7*, Nd.12	1399	577	RxPDO9*, Nd.55	1955	7A3	RxPDO5*, Nd.35
845	34D	RxPDO7*, Nd.13	1400	578	RxPDO9*, Nd.56	1956	7A4	RxPDO5*, Nd.36
846	34E	RxPDO7*, Nd.14	1401	579	RxPDO9*, Nd.57	1957	7A5	RxPDO5*, Nd.37

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
847	34F	RxPDO7*, Nd.15	1402	57A	RxPDO9*, Nd.58	1958	7A6	RxPDO5*, Nd.38
848	350	RxPDO7*, Nd.16	1403	57B	RxPDO9*, Nd.59	1959	7A7	RxPDO5*, Nd.39
849	351	RxPDO7*, Nd.17	1404	57C	RxPDO9*, Nd.60	1960	7A8	RxPDO5*, Nd.40
850	352	RxPDO7*, Nd.18	1405	57D	RxPDO9*, Nd.61	1961	7A9	RxPDO5*, Nd.41
851	353	RxPDO7*, Nd.19	1406	57E	RxPDO9*, Nd.62	1962	7AA	RxPDO5*, Nd.42
852	354	RxPDO7*, Nd.20	1407	57F	RxPDO9*, Nd.63	1963	7AB	RxPDO5*, Nd.43
853	355	RxPDO7*, Nd.21	1409	581	SDO Tx Nd.1	1964	7AC	RxPDO5*, Nd.44
854	356	RxPDO7*, Nd.22	1410	582	SDO Tx Nd.2	1965	7AD	RxPDO5*, Nd.45
855	357	RxPDO7*, Nd.23	1411	583	SDO Tx Nd.3	1966	7AE	RxPDO5*, Nd.46
856	358	RxPDO7*, Nd.24	1412	584	SDO Tx Nd.4	1967	7AF	RxPDO5*, Nd.47
857	359	RxPDO7*, Nd.25	1413	585	SDO Tx Nd.5	1968	7B0	RxPDO5*, Nd.48
858	35A	RxPDO7*, Nd.26	1414	586	SDO Tx Nd.6	1969	7B1	RxPDO5*, Nd.49
859	35B	RxPDO7*, Nd.27	1415	587	SDO Tx Nd.7	1970	7B2	RxPDO5*, Nd.50
860	35C	RxPDO7*, Nd.28	1416	588	SDO Tx Nd.8	1971	7B3	RxPDO5*, Nd.51
861	35D	RxPDO7*, Nd.29	1417	589	SDO Tx Nd.9	1972	7B4	RxPDO5*, Nd.52
862	35E	RxPDO7*, Nd.30	1418	58A	SDO Tx Nd.10	1973	7B5	RxPDO5*, Nd.53
863	35F	RxPDO7*, Nd.31	1419	58B	SDO Tx Nd.11	1974	7B6	RxPDO5*, Nd.54
864	360	RxPDO7*, Nd.32	1420	58C	SDO Tx Nd.12	1975	7B7	RxPDO5*, Nd.55
865	361	RxPDO7*, Nd.33	1421	58D	SDO Tx Nd.13	1976	7B8	RxPDO5*, Nd.56
866	362	RxPDO7*, Nd.34	1422	58E	SDO Tx Nd.14	1977	7B9	RxPDO5*, Nd.57
867	363	RxPDO7*, Nd.35	1423	58F	SDO Tx Nd.15	1978	7BA	RxPDO5*, Nd.58
868	364	RxPDO7*, Nd.36	1424	590	SDO Tx Nd.16	1979	7BB	RxPDO5*, Nd.59
869	365	RxPDO7*, Nd.37	1425	591	SDO Tx Nd.17	1980	7BC	RxPDO5*, Nd.60
870	366	RxPDO7*, Nd.38	1426	592	SDO Tx Nd.18	1981	7BD	RxPDO5*, Nd.61
871	367	RxPDO7*, Nd.39	1427	593	SDO Tx Nd.19	1982	7BE	RxPDO5*, Nd.62
872	368	RxPDO7*, Nd.40	1428	594	SDO Tx Nd.20	1983	7BF	RxPDO5*, Nd.63

dec	hex	Telegram type	dec	hex	Telegram type	dec	hex	Telegram type
873	369	RxPDO7*, Nd.41	1429	595	SDO Tx Nd.21			

2.3.7 Emergency Object

In order to be able to inform other participating devices on the CANopen bus about internal device errors or CAN bus errors, CANopen Bus Couplers can make use of the emergency object. It has a high priority, and provides valuable information about the state of the device and of the network.

NOTE

Warning It is strongly recommended that emergency objects are evaluated - they provide a valuable source of information.

Structure of the emergency message

The emergency object is always 8 bytes long; it contains first the 2-byte error code, then the 1-byte error register, and finally the additional code of 5 bytes. This is divided into a 2-byte bit field and a 3-byte parameter field:

11 bit identifier	8 bytes of user data							
0x80 (=128dec) + node-ID	EC0	EC1	EReg	Bit field 0: Comm	Bit field 1: DevErr	EMCY Trigger	Info 0	Info 1

Table 1: Key

Parameters	Explanation	
EC0	Error Code Low-Byte. Not used (always zero)	
EC1	Error Code High-Byte. 0x50 = device error, 0x81 = communication error, 0x00 = error reset	
EReg	Error register. 0x81 = device error, 0x91 = communication error	
Bit field 0: Comm	Bit field communication error:	
	0x01	Guarding delayed or failed
	0x02	Sync delayed or failed
	0x04	Incorrect PDO length parameterized
	0x08	Event timer timeout: RxPDO not received in time
	0x10	Receive queue overrun
	0x20	Transmit queue overrun
	0x40	CAN bus OFF
	0x80	CAN warning limit exceeded
Bit field 1: DevErr	Bit field device error:	
	0x01	Terminal error
	0x02	K-Bus error / IP-Link error
	0x03	-
	0x04	EEPROM error
	0x10	Unsupported terminal plugged in (BK5110, LC5100)
	0x80	Altered HW configuration.

Parameters	Explanation	
EMCY trigger	The *emergency trigger* byte contains the code for the particular error that has triggered the emergency telegram. If an error has been rectified, an emergency telegram with the error code 0x0000 is sent, and the emergency trigger contains the description of the error that has been corrected. Errors that are still current are signaled here in the bit fields. Once the Bus Coupler is free of errors, it sends an emergency telegram containing zeros everywhere other than in the emergency trigger.	
	0x01	CAN warning limit exceeded (too many error frames)
	0x02	CAN bus OFF state has been reached. Since the coupler can no longer send an emergency telegram, an emergency telegram with trigger 0x40 is sent when the bus leaves the "off" state (a new CAN controller initialization).
	0x03	Transmit queue overrun: CAN messages are being lost
	0x04	Transmit queue overrun: CAN messages are being lost
	0x06	Incorrect PDO length parameterized (check mapping). Info 0: parameterized (expected) PDO length in bytes Info 1: current PDO length (results from the added lengths of the mapped objects)
	0x07	Sync delayed (time-out after communication cycle period, index 0x1006) or failed
	0x08	Guarding or heartbeat delayed (timeout following guard time x lifetime factor, or following consumer heartbeat time) or failed.
	0x09	Altered HW configuration. The inserted terminals or the composition of the extension modules has been changed since the last save.
	0x0A	Event timer timeout: RxPDO not received in time
	0x0B	Logical Tx queue overrun: SYNC interval too short. The coupler could not deliver all the TxPDOs before the following SYNC telegram. The TxPDOs are then, for instance, delivered in every second SYNC interval. Remedy: Lengthen the SYNC interval or raise the transmission type. In some cases it may be appropriate to reduce the I/O count at this bus station (e.g. by moving I/Os to the neighboring station)
	0x0C	Unsupported terminal plugged in (BK5110 or LC5100) Info 1: terminal number (1...64)
	0x0E	EEPROM error; an error occurred when saving the configuration in the EEPROM
	0x0F	K-Bus error Info 0: Error type:

Parameters	Explanation
Info 0, Info 1	Contains additional error information; its meaning depends on the emergency trigger (see above)

Example of emergency behaviour

1. The CAN error counter in a Bus Coupler has exceeded the warning limit (too many error frames). It sends an emergency telegram with the identifier 0x80 + node address (default setting) with the following contents:
   00 81 91 80 00 01 00 00
   The first three bytes (0x00 81 91) identify a communication error, while the bit field 0 (0x80) indicates that the *CAN Warning Limit has been exceeded*. The EMCY trigger (0x01) shows that the emergency was triggered as a result of exceeding the warning limit.
2. Immediately afterwards a cable goes open circuit on the second channel of the 4-20 mA analog input terminal plugged into the tenth location. The Bus Coupler sends another emergency telegram with the following contents:
   00 50 91 80 01 10 0A 82
   The first two bytes (0x00 50) identify a hardware error. Bits 0 (generic error), 4 (communication) and 7 (manufacturer-specific) are set in the error register (0x91). Bit 7 is set in bit field 0 (0x80), showing that the CAN warning limit continues to be exceeded. Bit 0 is set in bit field 1 (0x01), indicating a terminal error. The EMCY trigger (0x10) indicates that it is this terminal error that has triggered the emergency telegram. Finally, Info 0 (0x0A) indicates the terminal number (10) while Info 1 (0x82) shows in bit 1 and bit 7 that channel 2 has an error.
3. If the error counter now falls below the warning limit again, the coupler sends the following emergency telegram:
   00 00 81 00 01 01 0A 82
   The error code (00 00) in the first two bytes shows that an error has been reset. The error register (0x81) continues to show the device error, because the cable is still broken. Bit field 0 (0x00) shows that the communication error is no longer present. According to bit field 1 (0x01) the terminal error continues to be present. The EMCY trigger (0x01) indicates that the reason for the transmission was the resetting of the CAN warning limit. Info 0 and Info 1 continue to show the terminal's diagnostics status code.
4. Once the broken cable has been repaired this error is also reset, and the coupler sends the following emergency telegram:
   00 00 00 00 00 00 00 00

2.3.8 Protocol description

2.3.8.1 Network Management

Simple Boot-Up

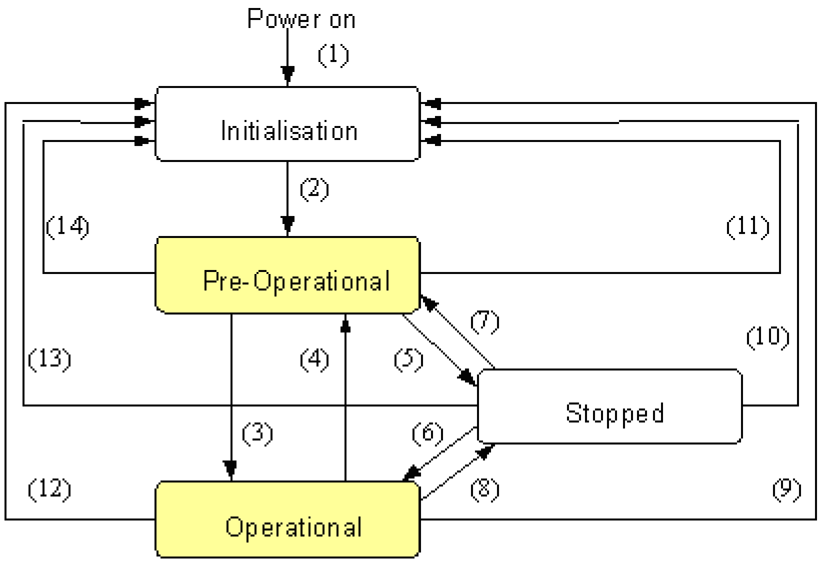
CANopen allows the distributed network to boot in a very simple way. After initialization, the modules are automatically in the *Pre-Operational* state. In this state it is already possible to access the object directory using service data objects (SDOs) with default identifiers, so that the modules can be configured. Since default settings exist for all the entries in the object directory, it is in most cases possible to omit any explicit configuration.

Only one CAN message is then required to start the module: Start_Remote_Node: Identifier *0*, two data bytes: 0x01, 0x00. It switches the node into the *Operational state*.

Network Status

Network Status

The states and the state transitions involved as CANopen boots up can be seen from the state diagram:



Pre-Operational

After initialization the Bus Coupler goes automatically (i.e. without the need for any external command) into the *Pre-Operational* state. In this state it can be configured, since the service data objects (SDOs) are already active. The process data objects, on the other hand, are still locked.

Operational

In the *Operational state* the process data objects are also active.

If external influences (such as a CAN error, or absence of output voltage) or internal influences (such as a K-Bus error) mean that it is no longer possible for the Bus Coupler to set outputs, to read inputs or to communicate, it attempts to send an appropriate emergency message, goes into the fault state, and thus returns to the *Pre-Operational* state. In this way the NMT status machine in the network master can also immediately detect fatal errors.

Stopped

In the *Stopped state* (formerly: *Prepared*) data communication with the Coupler is no longer possible - only NMT messages are received. The outputs go into the fault state.

State Transitions

State Transitions

The network management messages have a very simple structure: CAN identifier *0*, with two bytes of data content. The first data byte contains what is known as the command specifier (cs), and the second data byte contains the node address, the node address *0* applying to all nodes (broadcast).

11 bit identifier	2 bytes of user data							
0x00	cs	Node-ID						

The following table gives an overview of all the CANopen state transitions and the associated commands (command specifier in the NMT master telegram):

Status transition	Command Specifier cs	Explanation
(1)	-	The initialization state is reached automatically at power-up
(2)	-	After initialization the pre-operational state is reached automatically - this involves sending the boot-up message.
(3), (6)	cs = 1 = 0x01	Start_Remote_Node. Starts the module, enables outputs, starts transmission of PDOs.
(4), (7)	cs = 128 = 0x80	Enter_Pre-Operational. Stops PDO transmission, SDO still active.
(5), (8)	cs = 2 = 0x02	Stop_Remote_Node. Outputs go into the fault state, SDO and PDO switched off.
(9), (10), (11)	cs = 129 = 0x81	Reset_Node. Carries out a reset. All objects are reset to their power-on defaults.
(12), (13), (14)	cs = 130 = 0x82	Reset_Communication. Carries out a reset of the communication functions. Objects 0x1000 - 0x1FFF are reset to their power-on defaults.

Example 1

The following telegram puts all the modules in the network into the error state (outputs in a safe state):

11 bit identifier	2 bytes of user data							
0x00	0x02	0x00						

Example 2

The following telegram resets node 17:

11 bit identifier	2 bytes of user data							
0x00	0x81	0x11						

Boot-up message

Boot-up message

After the initialization phase and the self test, the Bus Coupler sends the boot-up message, a CAN message with no data bytes and with the identifier of the emergency message: CAN-ID = 0x700 + Node-ID. In this way temporary failure of a module during operation (e.g. due to a voltage interruption), or a module that is switched on at a later stage, can be reliably detected, even without Node Guarding. The sender can be determined from the message identifier (see default identifier allocation).

It is also possible, with the aid of the boot-up message, to recognize the nodes present in the network at start-up with a simple CAN monitor, without having to make write access to the bus (such as a scan of the network by reading out parameter 0x1000).

Finally, the boot-up message communicates the end of the initialization phase; the Bus Coupler signals that it can now be configured or started.

Firmware BA

Up to firmware status BA the emergency identifier was used for the boot up message.

Format of the Boot-up message

11 bit identifier	1 byte of user data							
0x700 (=1792) + Node-ID	0x00							

Node Monitoring

Node Monitoring

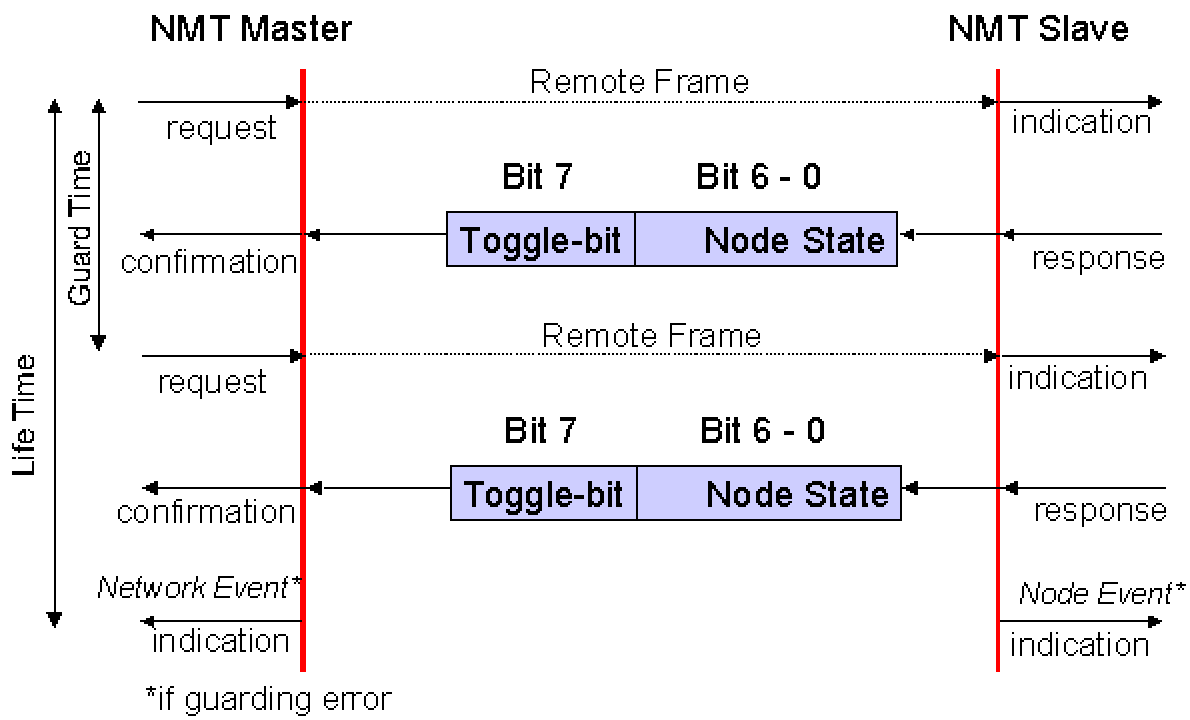
Heartbeat and guarding mechanisms are available to monitor failures in the CANopen network. These are of particular importance for CANopen, since modules do not regularly speak in the event-driven mode of operation. In the case of "guarding", the devices are cyclically interrogated about their status by means of a data request telegram (remote frame), whereas with "heartbeat" the nodes transmit their status on their own initiative.

Guarding: Node Guarding and Life Guarding

Guarding

Node Guarding is used to monitor the non-central peripheral modules, while they themselves can use Life Guarding to detect the failure of the guarding master. Guarding involves the master sending remote frames (remote transmit requests) to the guarding identifier of the slaves that are to be monitored. These reply with the guarding message. This contains the slave’s status code and a toggle bit that has to change after every message. If either the status or the toggle bit do not agree with that expected by the NMT master, or if there is no answer at all, the master assumes that there is a slave fault.

Guarding procedure



Protocol

Protocol

The toggle bit (t) transmitted in the first guarding telegram has the value *0*. After this, the bit must change (toggle) in every guarding telegram so that the loss of a telegram can be detected. The node uses the remaining seven bits to transmit its network status (s):

s	Status
4 = 0x04	Stopped (formerly: prepared)
5 = 0x05	Operational
127 = 0x7F	Pre-Operational

Example

The guarding message for node 27 (0x1B) must be requested by a remote frame having identifier 0x71B (1819_{dec}). If the node is *Operational*, the first data byte of the answer message alternates between 0x05 and 0x85, whereas in the *Pre-Operational* state it alternates between 0x7F and 0xFF.

Guard time and life time factor

If the master requests the guard messages in a strict cycle, the slave can detect the failure of the master. In this case, if the slave fails to receive a message request from the master within the set *Node Life Time* (a guarding error), it assumes that the master has failed (the watchdog function). It then puts its outputs into the error state, sends an emergency telegram, and returns to the pre-operational state. After a guarding time-out the procedure can be re-started by transmitting a guarding telegram again.

The node life time is calculated from the guard time (object 0x100C) and life time factor (object 0x100D) parameters:

Life time = guard time x life time factor

If either of these two parameters is "0" (the default setting), the master will not be monitored (no life guarding).

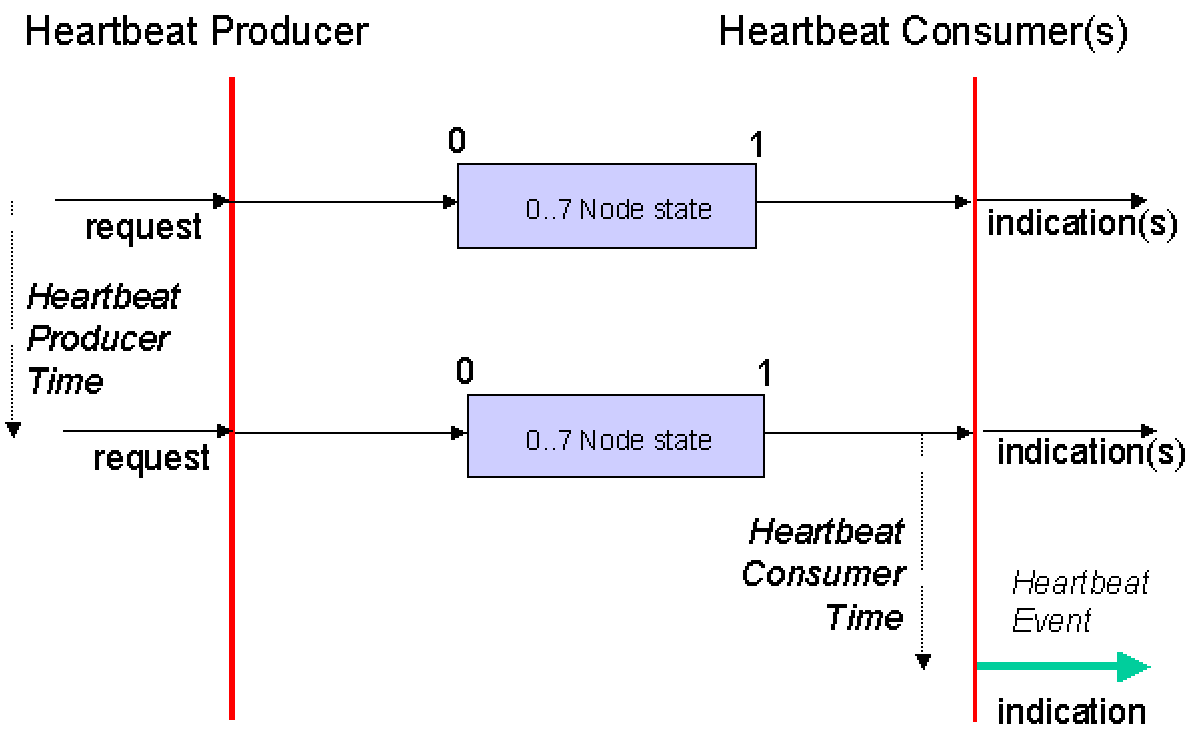
Heartbeat: Node Monitoring without Remote Frame

Heartbeat

In the heart beat procedure, each node transmits its status message cyclically on its own initiative. There is therefore no need to use remote frames, and the bus is less heavily loaded than under the guarding procedure.

The master also regularly transmits its heartbeat telegram, so that the slaves are also able to detect failure of the master.

Heartbeat procedure



Protocol

The toggle bit is not used in the heart beat procedure. The nodes send their status cyclically (s). See Guarding.

2.3.8.2 Process Data Objects (PDO)

Introduction

In many fieldbus systems the entire process image is continuously transferred - usually in a more or less cyclic manner. CANopen is not limited to this communication principle, since the multi-master bus access protocol allows CAN to offer other methods. Under CANopen the process data is not transferred in a master/slave procedure, but follows instead the producer-consumer model. In this model, a bus node transmits its data, as a producer, on its own accord. This might, for example, be triggered by an event. All the other nodes listen, and use the identifier to decide whether they are interested in this telegram, and handle it accordingly. These are the consumers.

The process data in CANopen is divided into segments with a maximum of 8 bytes. These segments are known as process data objects (PDOs). The PDOs each correspond to a CAN telegram, whose specific CAN identifier is used to allocate them and to determine their priority. Receive PDOs (RxPDOs) and transmit PDOs (TxPDOs) are distinguished, the name being chosen from the point of view of the device: an input/output module sends its input data with TxPDOs and receives its output data in the RxPDOs. **This naming convention is retained in the TwinCAT System Manager.**

Communication parameters

Communication parameters

The PDOs can be given different communication parameters according to the requirements of the application. Like all the CANopen parameters, these are also available in the device's object directory, and can be accessed by means of the service data objects. The parameters for the receive PDOs are at index 0x1400 (RxPDO1) onwards. There can be up to 512 RxPDOs (ranging up to index 0x15FF). In the same way, the entries for the transmit PDOs are located from index 0x1800 (TxPDO1) to 0x19FF (TxPDO512).

The BECKHOFF Bus Couplers or Fieldbus Coupler Box modules make 16 RxPDO and TxPDOs available for the exchange of process data (although the figure for Economy and LowCost BK5110 and LC5100 Couplers and the Fieldbus Boxes is 5 PDOs each, since these devices manage a lower quantity of process data). The FC510x CANopen master card supports up to 192 transmit and 192 receive PDOs for each channel - although this is restricted by the size of the DPRAM. Up to 32 TxPDOs and 32 RxPDOs can be handled in slave mode.

For each existing process data object there is an associated communication parameter object. The TwinCAT System Manager automatically assigns the set parameters to the relevant object directory entries. These entries and their significance for the communication of process data are explained below.

PDO Identifier

PDO Identifier

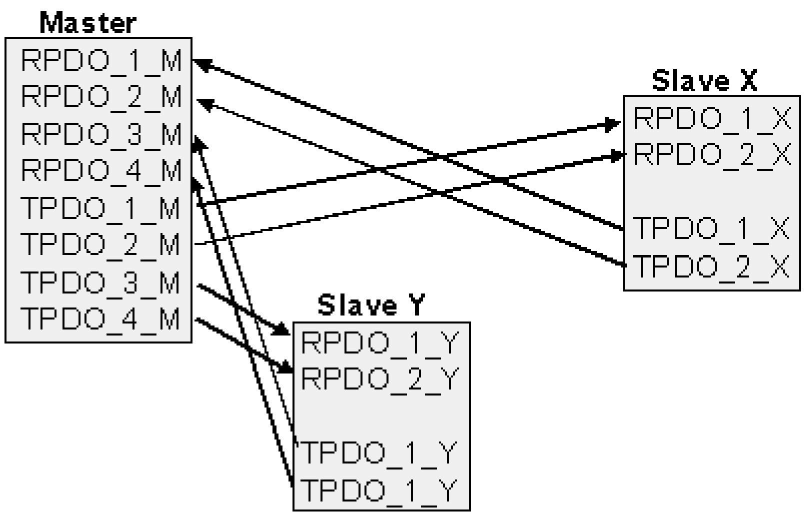
The most important communication parameter in a PDO is the CAN identifier (also know as the communication object identifier, or COB-ID). It is used to identify the data, and determines their priority for bus access. For each CAN data telegram there may only be one sender node (producer), although all messages sent in the CAN broadcast procedure can be received, as described, by any number of nodes (consumers). Thus a node can make its input information available to a number of bus devices at the same time - even without transferring them through a logical bus master. The identifier is located in sub-index 1 of the communication parameter set. It is coded as a 32-bit value in which the least significant 11 bits (bits 0...10) contain the identifier itself. The data width of the object of 32 bits also allows 29-bit identifiers in accordance with CAN 2.0B to be entered, although the default identifiers [▶ 110] always refer to the more usual 11-bit versions. Generally speaking, CANopen is economical it its use of the available identifiers, so that the use of the 29-bit versions remains limited to unusual applications. It is therefore also not supported by a Beckhoff's CANopen devices. The highest bit (bit 31) can be used to activate the process data object or to turn it off.

A complete identifier list [▶ 65] is provided in the appendix.

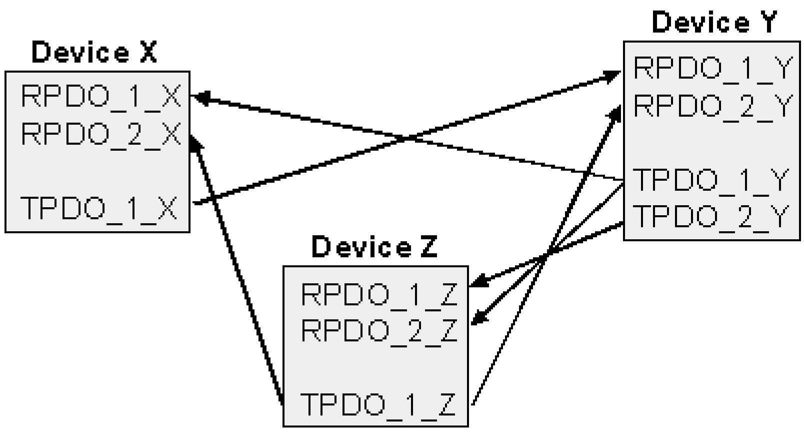
PDO linking

PDO linking

In the system of default identifiers, all the nodes (here: slaves) communicate with one central station (the master), since slave nodes do not listen by default to the transmit identifier of any other slave node.



Default identifier allocation: Master/Slave



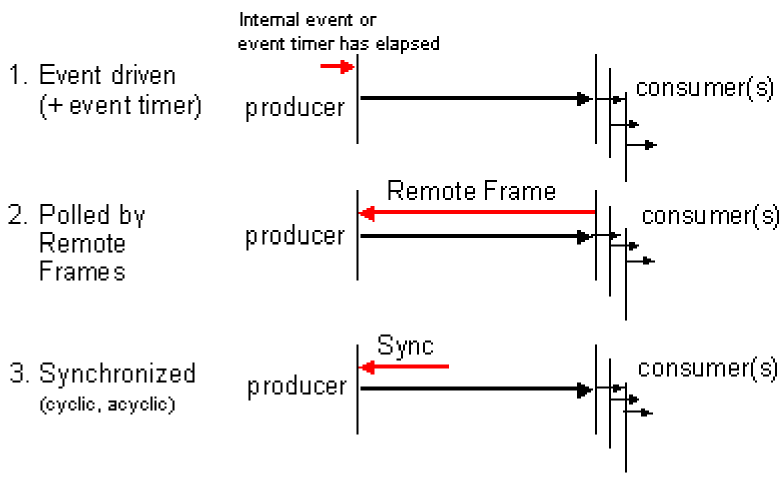
PDO linking: Peer to Peer

If the consumer-producer model of CANopen PDOs is to be used for direct data exchange between nodes (without a master), the identifier allocation must be appropriately adapted, so that the TxPDO identifier of the producer agrees with the RxPDO identifier of the consumer: This procedure is known as PDO linking. It permits, for example, easy construction of electronic drives in which several slave axes simultaneously listen to the actual value in the master axis TxPDO.

PDO Communication Types: Outline

PDO Communication Types: Outline

CANopen offers a number of possible ways to transmit process data (see also: Notes on PDO Parameterization [▶ 104]).)



.

Event driven

Event driven

The ”event" is the alteration of an input value, the data being transmitted immediately after this change. The event-driven flow can make optimal use of the bus bandwidth, since instead of the whole process image it is only the changes in it that are transmitted. A short reaction time is achieved at the same time, since when an input value changes it is not necessary to wait for the next interrogation from a master.

As from CANopen Version 4 it is possible to combine the event driven type of communication with a cyclic update. Even if an event has not just occurred, event driven TxPDOs are sent after the event timer has elapsed. If an event does occur, the event timer is reset. For RxPDOs the event timer is used as a watchdog in order to monitor the arrival of event driven PDOs . If a PDO does not arrive within a set period of time, the bus node adopts the error state.

Polled

Polled

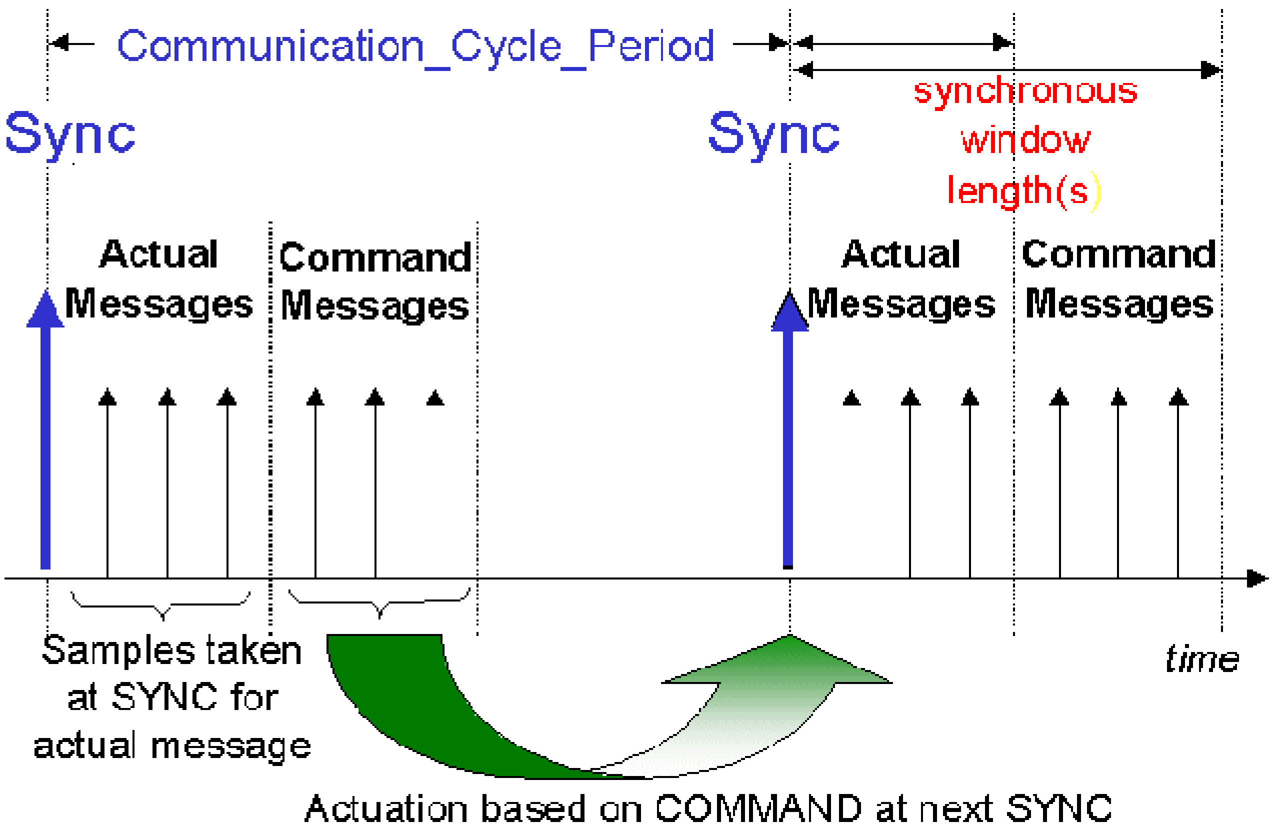
The PDOs can also be polled by data request telegrams (remote frames). In this way it is possible to get the input process image of event-driven inputs onto the bus, even when they do not change, for instance through a monitoring or diagnostic device brought into the network while it is running. The time behavior of remote frame and answer telegrams depends on what CAN controller is in use (Fig. 8). Components with full integrated message filtering ("FullCAN") usually answer a data request telegram immediately, transmitting data that is waiting in the appropriate transmit buffer - it is the responsibility of the application to see that the data there is continuously updated. CAN controllers with simple message filtering (BasicCAN) on the other hand pass the request on to the application which can now compose the telegram with the latest data. This does take longer, but does mean that the data is up-to-date. BECKHOFF use CAN controllers following the principle of Basic CAN.

Since this device behavior is usually not transparent to the user, and because there are CAN controllers still in use that do not support remote frames at all, polled communication can only with reservation be recommended for operative running.

Synchronized

Synchronized

It is not only for drive applications that it is worthwhile to synchronize the determination of the input information and the setting the outputs. For this purpose CANopen provides the SYNC object, a CAN telegram of high priority but containing no user data, whose reception is used by the synchronized nodes as a trigger for reading the inputs or for setting the outputs.



PDO transmission types: Parameterisation

PDO transmission types: Parameterisation

The PDO transmission type parameter specifies how the transmission of the PDO is triggered, or how received PDOs are handled.

Transmission type	Cyclical	Acyclical	Synchronous	Asynchronous	Only RTR
0		X	X		
1-240	X		X		
241-251	- reserved -				
252			X		X
253				X	X
254, 255				X	

The type of transmission is parameterized for RxPDOs in the objects at 0x1400ff, sub-index 2, and for TxPDOs in the objects at 0x1800ff, sub-index 2.

Acyclic Synchronous

PDOs of transmission type 0 function synchronously, but not cyclically. An RxPDO is only evaluated after the next SYNC telegram has been received. In this way, for instance, axis groups can be given new target positions one after another, but these positions only become valid at the next SYNC - without the need to be constantly outputting reference points. A device whose TxPDO is configured for transmission type 0 acquires its input data when it receives the SYNC (synchronous process image) and then transmits it if the data correspond to an event (such as a change in input) having occurred. Transmission type 0 thus combines transmission for reasons that are event driven with a time for transmission (and, as far as possible, sampling) and processing given by the reception of "SYNC".

Cyclic Synchronous

Cyclic Synchronous

In transmission types 1-240 the PDO is transmitted cyclically: after every ”nth" SYNC (n = 1...240). Since transmission types can be combined on a device as well as in the network, it is possible, for example, for a fast cycle to be agreed for digital inputs (n = 1), whereas the data for analog inputs is transmitted in a slower cycle (e.g. n = 10). RxPDOs do not generally distinguish between transmission types 0...240: a PDO that has been received is set to valid when the next SYNC is received. The cycle time (SYNC rate) can be monitored (object 0x1006), so that if the SYNC fails the device reacts in accordance with the definition in the device profile, and switches, for example, its outputs into the fault state.

The FC510x card provides full support for the synchronous type of communication: transmitting the SYNC telegram is coupled to the linked task, so that new input data is available every time the task begins. The card will recognize the absence of a synchronous PDO, and will report it to the application.

Only RTR

Transmission types 252 and 253 apply to process data objects that are transmitted exclusively on request by a remote frame. 252 is synchronous: when the SYNC is received the process data is acquired. It is only transmitted on request. 253 is asynchronous. The data here is acquired continuously, and transmitted on request. This type of transmission is not generally recommended, because fetching input data from some CAN controllers is only partially supported. Because, furthermore, the CAN controllers sometimes answer remote frames automatically (without first requesting up-to-date input data), there are circumstances in which it is questionable whether the polled data is up-to-date. Transmission types 252 and 253 are for this reason not supported by the BECKHOFF PC cards.

Asynchronous

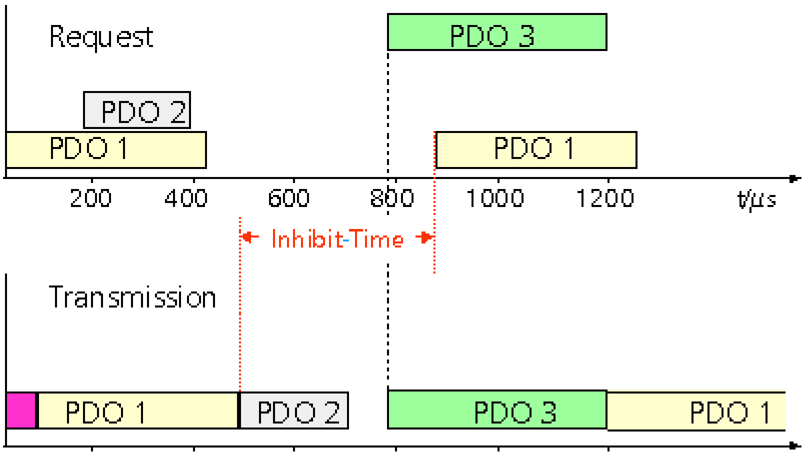
Asynchronous

The transmission types 254 + 255 are asynchronous, but may also be event-driven. In transmission type 254, the event is specific to the manufacturer, whereas for type 255 it is defined in the device profile. In the simplest case, the event is the change of an input value - this means that every change in the value is transmitted. The asynchronous transmission type can be coupled with the event timer, thus also providing input data when no event has just occurred.

Inhibit time

Inhibit time

The "inhibit time" parameter can be used to implement a "transmit filter" that does not increase the reaction time for relatively new input alterations, but is active for changes that follow immediately afterwards. The inhibit time (transmit delay time) specifies the minimum length of time that must be allowed to elapse between the transmission of two of the same telegrams. If the inhibit time is used, the maximum bus loading can be determined, so that the worst case latency can then be found.

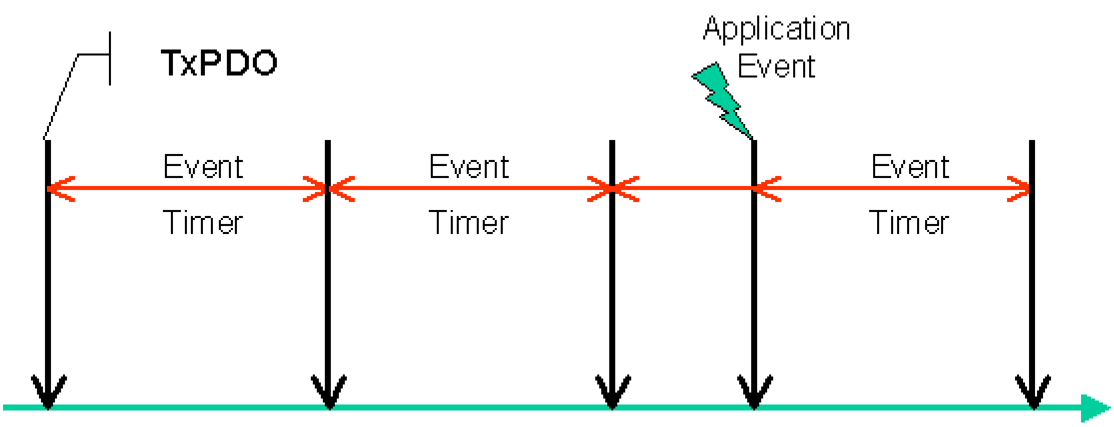


Although the BECKHOFF FC510x PC cards can parameterize the inhibit time on slave devices, they do not themselves support it. The transmitted PDOs become automatically spread out (transmit delay) as a result of the selected PLC cycle time - and there is little value in having the PLC run faster than the bus bandwidth permits. The bus loading, furthermore, can be significantly affected by the synchronous communication.

Event Timer

Event Timer

An event timer for transmit PDOs can be specified by sub-index 5 in the communication parameters. Expiry of this timer is treated as an additional event for the corresponding PDO, so that the PDO will then be transmitted. If the application event occurs during a timer period, it will also be transmitted, and the timer is reset.



In the case of receive PDOs, the timer is used to set a watchdog interval for the PDO: the application is informed if no corresponding PDO has been received within the set period. The FC510x can in this way monitor each individual PDO.

Notes on PDO Parameterization [▸ 104]

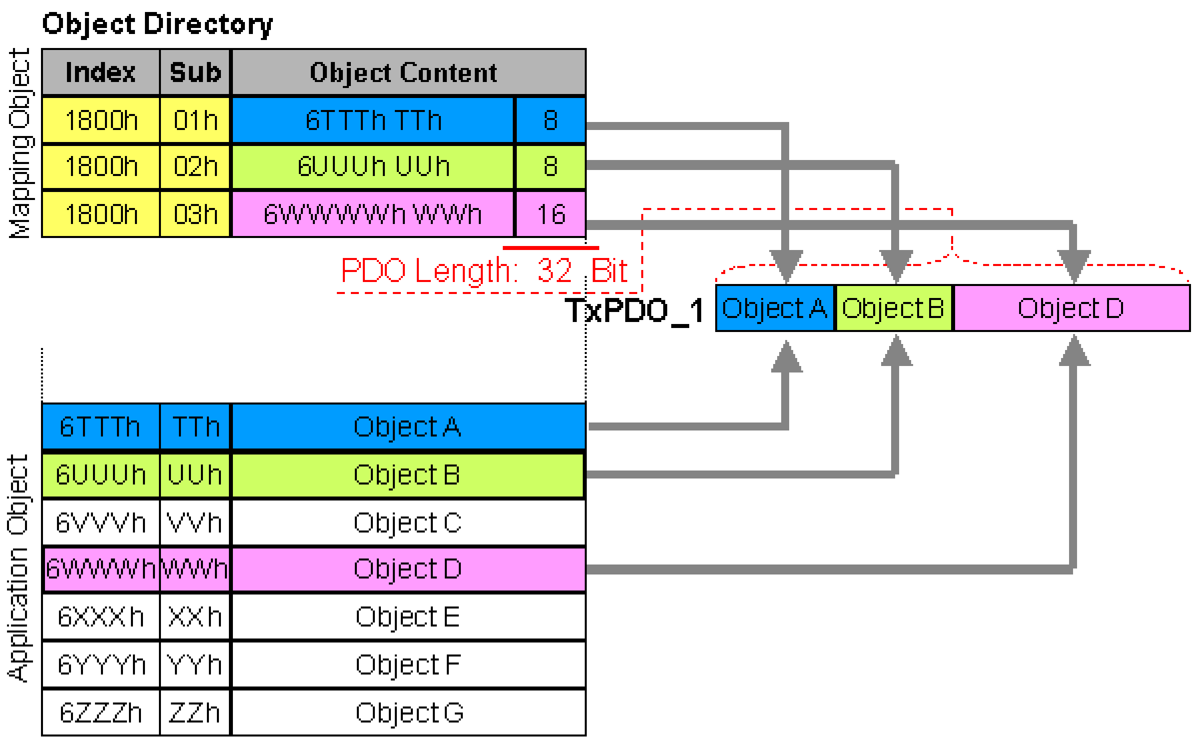
PDO Mapping

PDO Mapping

PDO mapping refers to mapping of the application objects (real time data) from the object directory to the process data objects. The CANopen device profile provide a default mapping for every device type, and this is appropriate for most applications. Thus the default mapping for digital I/O simply represents the inputs and outputs in their physical sequence in the transmit and receive process data objects.

The default PDOs for drives contain 2 bytes each of a control and status word and a set or actual value for the relevant axis.

The current mapping can be read by means of corresponding entries in the object directory. These are known as the mapping tables. The first location in the mapping table (sub-index 0) contains the number of mapped objects that are listed after it. The tables are located in the object directory at index 0x1600ff for the RxPDOs and at 0x1A00ff for the TxPDOs.



Digital and analog input/output modules: Read out the I/O number

The current number of digital and analog inputs and outputs can be determined or verified by reading out the corresponding application objects in the object directory:

Parameters	Object directory address
Number of digital input bytes	Index 0x6000, sub-index 0
Number of digital output bytes	Index 0x6200, sub-index 0
Number of analog inputs	Index 0x6401, sub-index 0
Number of analog outputs	Index 0x6411, sub-index 0

Variable mapping

As a rule, the default mapping of the process data objects already satisfies the requirements. For special types of application the mapping can nevertheless be altered: the Beckhoff CANopen Bus Couplers, for instance, thus support variable mapping, in which the application objects (input and output data) can be freely allocated to the PDOs. The mapping tables must be configured for this: as from Version 4 of CANopen, only the following procedure is permitted, and must be followed precisely:

1. First delete the PDO (set 0x1400ff, or 0x1800ff, sub-index 1, bit 31 to "1")
2. Set sub-index 0 in the mapping parameters (0x1600ff or 0x1A00ff) to "0"
3. Change mapping entries (0x1600ff or 0x1A00ff, SI 1..8)
4. Set sub-index 0 in the mapping parameters to the valid value. The device then checks the entries for consistency.
5. Create PDO by entering the identifier (0x1400ff or 0x1800ff, sub-index 1).

Dummy Mapping

A further feature of CANopen is the mapping of placeholders, or dummy entries. The data type entries stored in the object directory, which do not themselves have data, are used as placeholders. If such entries are contained in the mapping table, the corresponding data from the device is not evaluated. In this way, for instance, a number of drives can be supplied with new set values using a single CAN telegram, or outputs on a number of nodes can be set simultaneously, even in event-driven mode.

2.3.8.3 PDO Parameterization

Even though the majority of CANopen networks operate satisfactorily with the default settings, i.e. with the minimum of configuration effort, it is wise at least to check whether the existing bus loading is reasonable: 80% bus loading may be acceptable for a network operating purely in cyclic synchronous modes, but for a network with event-driven traffic this value would generally be too high, as there is hardly any bandwidth available for additional events.

Consider the Requirements of the Application

The communication of the process data must be optimized in the light of application requirements which are likely to be to some extent in conflict. These include

- Little work on parameterization - useable default values are optimal
- Guaranteed reaction time for specific events
- Cycle time for regulation processes over the bus
- Safety reserves for bus malfunctions (enough bandwidth for the repetition of messages)
- Maximum baud rate - depends on the maximum bus length
- Desired communication paths - who is speaking with whom

The determining factor often turns out to be the available bus bandwidth (bus load).

Baud Rate

Baud Rate

We generally begin by choosing the highest baud rate that the bus will permit. It should be borne in mind that serial bus systems are fundamentally more sensitive to interference as the baud rate is increased. The following rule therefore applies: just as fast as necessary. 1000 kbit/s are not usually necessary, and only to be unreservedly recommended on networks within a control cabinet where there is no electrical isolation between the bus nodes. Experience also tends to show that estimates of the length of bus cable laid are often over-optimistic - the length actually laid tends to be longer.

Determine the Communication Type

Once the baud rate has been chosen it is appropriate to specify the PDO communication type(s). These have different advantages and disadvantages:

- Cyclic synchronous communication provides an accurately predictable bus loading, and therefore a defined time behavior - you could say that the standard case is the worst case. It is easy to configure: The SYNC rate parameter sets the bus loading globally. The process images are synchronized: Inputs are read at the same time, output data is set valid simultaneously, although the quality of the synchronization depends on the implementation. The Beckhoff FC510x PC cards are capable of synchronizing the CANopen bus system with the cycles of the application program (PLC or NC). The guaranteed reaction time under cyclic synchronous communication is always at least as long as the cycle time, and the bus bandwidth is not exploited optimally, since old data, i.e. data that has not changed, is continuously transmitted. It is however possible to optimize the network through the selection of different SYNC multiples (transmission types 1...240), so that data that changes slowly is transmitted less often than, for instance, time-critical inputs. It must, however, be borne in mind that input states that last for a time that is shorter than the cycle time will not necessarily be communicated. If it is necessary for such conditions to be registered, the associated PDOs for asynchronous communication should be provided.
- Event-driven asynchronous communication is optimal from the point of view of reaction time and the exploitation of bus bandwidth - it can be described as "pure CAN". Your choice must, however, also take account of the fact that it is not impossible for a large number of events to occur simultaneously, leading to corresponding delays before a PDO with a relatively low priority can be sent. Proper network planning therefore necessitates a worst-case analysis. Through the use of, for instance, inhibit time [▶ 15], it is also necessary to prevent a constantly changing input with a high PDO priority from blocking the bus (technically known as a "babbling idiot"). It is for this reason that event driving is switched off by default in the device profile of analog inputs, and must be turned on specifically. Time windows for the transmit PDOs can be set using progress timers: the telegram is not sent again before the inhibit time [▶ 15] has elapsed, and not later than the time required for the progress timer to complete.
- The communication type is parameterized by means of the transmission type [▶ 15].

It is also possible to combine the two PDO principles. It can, for instance, be helpful to exchange the set and actual values of an axis controller synchronously, while limit switches, or motor temperatures with limit values are monitored with event-driven PDOs. This combines the advantages of the two principles: synchronicity for the axis communication and short reaction times for limit switches. In spite of being event-driven, the distributed limit value monitoring avoids a constant addition to the bus load from the analog temperature value.

In this example it can also be of value to deliberately manipulate the identifier allocation, in order to optimize bus access by means of priority allocation: the highest priority is given to the PDO with the limit switch data, and the lowest to that with the temperature values.

Optimization of bus access latency time through modification of the identifier allocation is not, however, normally required. On the other hand the identifiers must be altered if masterless communication is to be made possible (PDO linking [▶ 15]). In this example it would be possible for one RxPDO for each axis to be allocated the same identifier as the limit switch TxPDO, so that alterations of the input value can be received without delay.

Determining the Bus Loading

Determining the Bus Loading

It is always worth determining the bus loading. But what bus loading values are permitted, or indeed sensible? It is first necessary to distinguish a short burst of telegrams in which a number of CAN messages follow one another immediately - a temporary 100% bus loading. This is only a problem if the sequence of receive interrupts that it caused at the CAN nodes can not be handled. This would constitute a data overflow (or CAN queue overrun). This can occur at very high baud rates (> 500 kbit/s) at nodes with software telegram filtering and relatively slow or heavily loaded microcontrollers if, for instance, a series of remote frames (which do not contain data bytes, and are therefore very short) follow each other closely on the bus (at 1 Mbit/s this can generate an interrupt every 40 µs; for example, an NMT master might transmit all its guarding requests in an unbroken sequence). This can be avoided through skilled implementation, and the user should be able to assume that the device suppliers have taken the necessary trouble. A burst condition is entirely normal immediately after the SYNC telegram, for instance: triggered by the SYNC, all the nodes that are operating synchronously try to send their data at almost the same time. A large number of arbitration processes take place, and the telegrams are sorted in order of priority for transmission on the bus. This is not usually critical, since these telegrams do contain some data bytes, and the telegrams trigger a sequence of receive interrupts at the CAN nodes which is indeed rapid, but is nevertheless manageable.

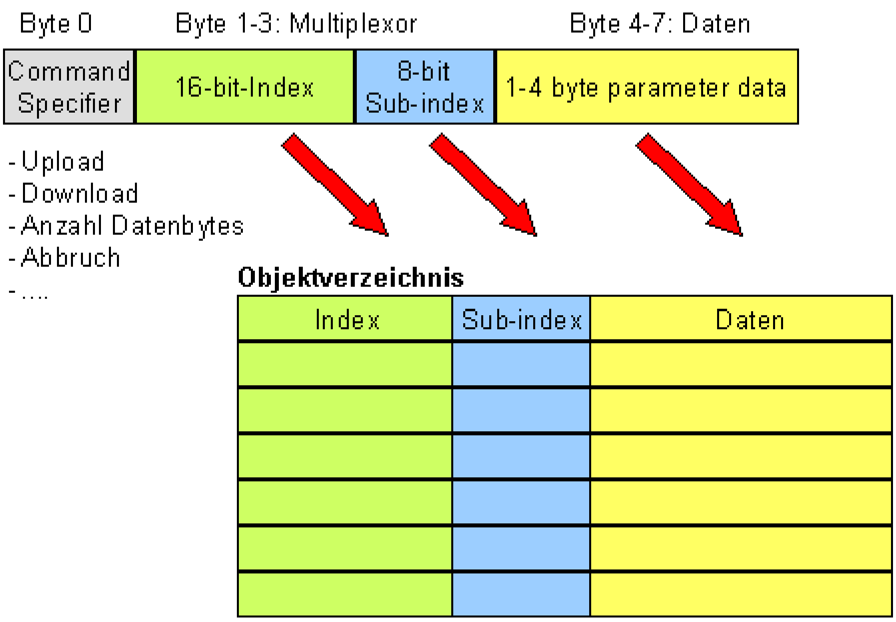
Bus loading most often refers to the value averaged over several primary cycles, that is the mean value over 100-500 ms. CAN, and therefore CANopen, is indeed capable of managing a bus loading of close to 100% over long periods, but this implies that no bandwidth is available for any repetitions that may be necessitated by interference, for asynchronous error messages, parameterization and so on. Clearly, the dominant type of communication will have a large influence on the appropriate level of bus loading: a network with entirely cyclic synchronous operation is always in any case near to the worst case state, and can therefore be operated with values in the 70-80% range. The figure is very hard to state for an entirely event-driven network: an estimate must be made of how many events additional to the current state of the system might occur, and of how long the resulting burst might last - in other words, for how long the lowest priority message will be delayed. If this value is acceptable to the application, then the current bus loading is acceptable. As a rule of thumb it can usually be assumed that an event-driven network running with a base loading of 30-40% has enough reserve for worst-case scenarios, but this assumption does not obviate the need for a careful analysis if delays could have critical results for the plant.

The BECKHOFF FC510x PC cards indicate the bus loading via the System Manager. This variable can also be processed in the PLC, or can be displayed in the visualization system.

The amount data in the process data objects is of course as relevant as the communication parameters: the PDO mapping [▶ 15].

2.3.8.4 Service Data Objects (SDO)

The parameters listed in the object directory are read and written by means of service data objects. These SDOs are *Multiplexed Domains*, i.e. data structures of any size that have a multiplexer (address). The multiplexer consists of a 16-bit index and an 8-bit sub-index that address the corresponding entries in the object directory.



SDO protocol: access to the object directory

The CANopen Bus Couplers are servers for the SDO, which means that at the request of a client (e.g. of the IPC or the PLC) they make data available (upload), or they receive data from the client (download). This involves a handshake between the client and the server.

When the size of the parameter to be transferred is not more than 4 bytes, a single handshake is sufficient (one telegram pair): For a download, the client sends the data together with its index and sub-index, and the server confirms reception. For an upload, the client requests the data by transmitting the index and sub-index of the desired parameter, and the server sends the parameter (including index and sub-index) in its answer telegram.

The same pair of identifiers is used for both upload and download. The telegrams, which are always 8 bytes long, encode the various services in the first data byte. All parameters with the exception of objects 1008h, 1009h and 100Ah (device name, hardware and software versions) are only at most 4 bytes long, so this description is restricted to transmission in expedited transfer.

Protocol

The structure of the SDO telegrams is described below.

Client -> Server, Upload Request

11 bit identifier	8 bytes of user data							
0x600 (=1536dez) + node ID	0x40	Index0	Index1	SubIdx	0x00	0x00	0x00	0x00

Parameters	Explanation
Index0	Index low byte (Unsigned16, LSB)
Index1	Index high byte (Unsigned16, MSB)
SubIdx	Sub-index (Unsigned8)

Client -> Server, Upload Response

11 bit identifier	8 bytes of user data							
0x580 (=1408dec) + node ID	0x4x	Index0	Index1	SubIdx	Data0	Data1	Data2	Data3

Parameters	Explanation
Index0	Index low byte (Unsigned16, LSB)
Index1	Index high byte (Unsigned16, MSB)
SubIdx	Sub-index (Unsigned8)
Data0	Data low low byte (LLSB)
Data3	Data high high byte (MMSB)

Parameters whose data type is Unsigned8 are transmitted in byte D0, parameters whose type is Unsigned16 use D0 and D1.

The number of valid data bytes is coded as follows in the first CAN data byte (0x4x):

Number of parameter bytes	1	2	3	4
First CAN data byte	0x4F	0x4B	0x47	0x43

Client -> Server, Download Request

11 bit identifier	8 bytes of user data							
0x600 (=1536dec) + node ID	0x22	Index0	Index1	SubIdx	Data0	Data1	Data2	Data3

Parameters	Explanation
Index0	Index low byte (Unsigned16, LSB)
Index1	Index high byte (Unsigned16, MSB)
SubIdx	Sub-index (Unsigned8)
Data0	Data low low byte (LLSB)
Data3	Data high high byte (MMSB)

It is optionally possible to give the number of valid parameter data bytes in the first CAN data byte

Number of parameter bytes	1	2	3	4
First CAN data byte	0x2F	0x2B	0x27	0x23

This is, however, not generally necessary, since only the less significant data bytes up to the length of the object directory entry that is to be written are evaluated. A download of data up to 4 bytes in length can therefore always be achieved in Beckhoff bus nodes with 22h in the first CAN data byte.

Client -> Server, Download Response

11 bit identifier	8 bytes of user data							
0x580 (=1408dec) + node ID	0x60	Index0	Index1	SubIdx	0x00	0x00	0x00	0x00

Parameters	Explanation
Index0	Index low byte (Unsigned16, LSB)
Index1	Index high byte (Unsigned16, MSB)
SubIdx	Sub-index (Unsigned8)

Breakdown of Parameter Communication

Parameter communication is interrupted if it is faulty. The client or server send an SDO telegram with the following structure for this purpose:

11 bit identifier	8 bytes of user data							
0x580 (client) or 0x600(server) + node ID	0x80	Index0	Index1	SubIdx	Error0	Error1	Error2	Error3

Parameters	Explanation
Index0	Index low byte (Unsigned16, LSB)
Index1	Index high byte (Unsigned16, MSB)
SubIdx	Sub-index (Unsigned8)
Error0	SDO error code low low byte (LLSB)
Error3	SDO error code high high byte (MMSB)

List of SDO error codes (reason for abortion of the SDO transfer):

SDO error code	Explanation
0x05 03 00 00	Toggle bit not changed
0x05 04 00 01	SDO command specifier invalid or unknown
0x06 01 00 00	Access to this object is not supported
0x06 01 00 02	Attempt to write to a Read_Only parameter
0x06 02 00 00	The object is not found in the object directory
0x06 04 00 41	The object can not be mapped into the PDO
0x06 04 00 42	The number and/or length of mapped objects would exceed the PDO length
0x06 04 00 43	General parameter incompatibility
0x06 04 00 47	General internal error in device
0x06 06 00 00	Access interrupted due to hardware error
0x06 07 00 10	Data type or parameter length do not agree or are unknown
0x06 07 00 12	Data type does not agree, parameter length too great
0x06 07 00 13	Data type does not agree, parameter length too short
0x06 09 00 11	Sub-index not present
0x06 09 00 30	General value range error
0x06 09 00 31	Value range error: parameter value too great
0x06 09 00 32	Value range error: parameter value too small
0x06 0A 00 23	Resource not available
0x08 00 00 21	Access not possible due to local application
0x08 00 00 22	Access not possible due to current device status

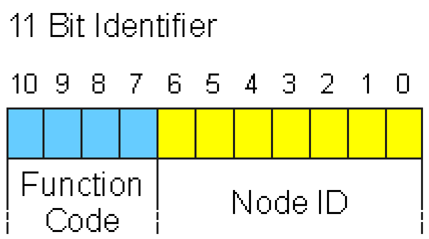
Further, manufacturer-specific error codes have been introduced for register communication (index 0x4500, 0x4501):

SDO error code	Explanation
0x06 02 00 11	Invalid table: Table or channel not present
0x06 02 00 10	Invalid register: table not present
0x06 01 00 22	Write protection still set
0x06 07 00 43	Incorrect number of function arguments
0x06 01 00 21	Function still active, try again later
0x05 04 00 40	General routing error
0x06 06 00 21	Error accessing BC table
0x06 09 00 10	General error communicating with terminal
0x05 04 00 47	Time-out communicating with terminal

2.3.8.5 Identifier Allocation

Default identifier

CANopen provides default identifiers for the most important communication objects, and these are derived from the 7-bit node address (the node ID) and a 4-bit function code in accordance with the following scheme:



For broadcast objects the node ID is set to *0*. This gives rise to the following default identifiers:

Broadcast objects

Object	Function	Function code	Resulting COB ID		Object for communication Parameter / mapping
			hex	dec	
NMT	Boot-Up	0	0x00	0	- / -
SYNC	Synchronization	1	0x80	128	0x1005 [▶ 22] + 0x1006 [▶ 22] / -

Peer-to-peer objects

Object	Function	Function code	Resulting COB ID		Object for communica-tion Parame-ter / mapping
			hex	dec	
Emergency	Status / error	1	0x81 - 0xFF	129 - 255	- / -
PDO1 (tx)	dig. inputs	11	0x181 - 0x1FF	385 - 511	0x1800 [▸ 22] / 0x1A00 [▸ 22]
PDO1 (rx)	digital outputs	100	0x201 - 0x27F	513 - 639	0x1400 [▸ 22] / 0x1600 [▸ 22]
PDO2 (tx)	analog inputs	101	0x281 - 0x2FF	641 - 767	0x1801 [▸ 22] / 0x1A01 [▸ 22]
PDO2 (rx)	analog outputs	110	0x301 - 0x37F	769 - 895	0x1401 [▸ 22] / 0x1601 [▸ 22]
PDO3 (tx)	analog inputs*	111	0x381 - 0x3FF	897 - 1023	0x1802 [▸ 22] / 0x1A02 [▸ 22]
PDO3 (rx)	analog outputs*	1000	0x401 - 0x47F	1025 - 1151	0x1402 [▸ 22] / 0x1602 [▸ 22]
PDO4 (tx)	analog inputs*	1001	0x481 - 0x4FF	1153 - 1279	0x1803 [▸ 22] / 0x1A03 [▸ 22]
PDO4 (rx)	analog outputs*	1010	0x501 - 0x57F	1281 - 1407	0x1403 [▸ 22] / 0x1603 [▸ 22]
SDO (tx)	Parameters	1011	0x581 - 0x5FF	1409 - 1535	- / -
SDO (rx)	Parameters	1100	0x601 - 0x67F	1537 - 1663	- / -
Guarding	Life/node guarding, Heartbeat, Boot-up message	1110	0x701 - 0x77F	1793 - 1919	(0x100C [▸ 22], 0x100D [▸ 22], 0x100E [▸ 22], 0x1016 [▸ 22], 0x1017 [▸ 22])

*) The Beckhoff Default Mapping [▸ 63] applies to PDO 3 + 4. In most configurations, PDOs 3 and 4 contain data related to analog inputs and outputs, but there can also be "excess" data from digital I/Os, or data from special terminals. Details may be found in the section covering PDO Mapping [▸ 15].

Up until version 3 of the CANopen specification, default identifiers were assigned to 2 PDOs at a time. The BECKHOFF Bus Couplers up to firmware status BA correspond to this issue of the specification. After firmware status C0 (CANopen version 4), default identifiers are provided for up to 4 PDOs.

Manufacture-Specific Default Identifiers for Additional PDOs

Default Identifiers for Additional PDOs

Identifiers are not assigned to the additional PDOs that are filled by the Beckhoff Bus Couplers in accordance with the standard scheme. The user must enter an identifier for these PDOs in the object directory. It is easier to activate the occupied PDOs by means of object 0x5500 [▸ 22].

This entry in the object directory extends the default identifier allocation up to 11 PDOs. This creates the following identifiers:

Object	Function code	Resulting COB ID (hex)	Resulting COB ID (dec)
PDO5 (tx)	1101	0x681 - 0x6BF	1665 - 1727
PDO5 (rx)	1111	0x781 - 0x7BF	1921- 1983
PDO6 (tx)	111	0x1C1 - 0x1FF	449 - 511
PDO6 (rx)	1001	0x241 - 0x27F	577 - 639
PDO7 (tx)	1011	0x2C1 - 0x2FF	705 - 767
PDO7 (rx)	1101	0x341 - 0x37F	833 - 895
PDO8 (tx)	1111	0x3C1- 0x3FF	961 - 1023
PDO8 (rx)	10001	0x441 - 0x47F	1089 - 1151
PDO9 (tx)	10011	0x4C1 - 0x4FF	1217 - 1279
PDO9 (rx)	10101	0x541 - 0x57F	1345 - 1407
PDO10 (tx)	10111	0x5C1 - 0x5FF	1473 - 1535
PDO10 (rx)	11001	0x641 - 0x67F	1601- 1663
PDO11 (tx)	11011	0x6C1 - 0x6FF	1729 - 1791
PDO11 (rx)	11101	0x741 - 0x77F	1857 - 1919

NOTE

WarningIndex 0x5500 must not be used if Bus Couplers with more than 5 PDOs are present in networks with node numbers greater than 64, otherwise identifier overlaps can occur.

2.3.9 Objekt directory

2.3.9.1 Object Directory - Structure

All the CANopen objects relevant for the Bus Coupler are entered into the CANopen object directory. The object directory is divided into three different regions:

1. communication-specific profile region (index 0x1000 – 0x1FFF).
   This contains the description of all the parameters specific to communication.
2. manufacturer-specific profile region (index 0x2000 – 0x5FFF).
   Contains the description of the manufacturer-specific entries.
3. standardized device profile region (0x6000 – 0x9FFF).
   Contains the objects for a device profile according to DS-401.

Every entry in the object directory is identified by a 16 bit index. If an object consists of several components (e.g. object type array or record), the components are identified by an 8-bit sub-index. The object name describes the function of an object, while the data type attribute specifies the data type of the entry. The access attribute specifies whether an entry may only be read, only written, or may be both read and written.

Communication-specific region

All the parameters and objects necessary for the CANopen Bus Coupler's communication are in this region of the object directory. The region from 0x1000 to 0x1018 contains various general communication-specific parameters (e.g. the device name).

The communication parameters (e.g. identifiers) for the receive PDOs are located in the region from 0x1400 to 0x140F (plus sub-index). The mapping parameters of the receive PDOs are in the region from 0x1600 to 0x160F (plus sub-index). The mapping parameters contain the cross-references to the application objects that are mapped into the PDOs and the data width of the corresponding object (see also the section dealing with PDO Mapping).

The communication and mapping parameters for the transmit PDOs are located in the regions from 0x1800 to 0x180F and from 0x1A00 to 0x1A0F.

Manufacturer-specific region

This region contains entries that are specific to BECKHOFF, e.g.:

- data objects for special terminals
- objects for register communication providing access to all the Bus Couplers' and Bus Terminals' internal registers
- objects for simplified configuration of the PDOs

Standardized device profile region

The standardized device profile region supports the device profile of CANopen DS-401, Version 1. Functions are available for analog inputs that can adapt communication in the event-driven operating mode to the requirements of the application and to minimize the loading of the bus:

- limit value monitoring
- Delta function
- activation/deactivation of event-driven mode

2.3.9.2 Objects and Data

Device type

Device type

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1000	0	Device type	Unsigned32	ro	N	0x00000000	Statement of device type

The 32 bit value is divided into two 16 bit fields:

MSB	LSB
Additional information	Device profile number
0000 0000 0000 wxyz	0x191 (401_{dez})

The *additional information* contains data related to the signal type of the I/O device:
z=1 signifies digital inputs,
y=1 signifies digital outputs,
x=1 signifies analog inputs,
w=1 signifies analog outputs.
A BK5120 with digital and analog inputs, but with no outputs, thus returns 0x00 05 01 91.

Special terminals (such as serial interfaces, PWM outputs, incremental encoder inputs) are not considered. A Coupler that, for example, only has KL6001 serial interface terminals plugged in, thus returns 0x00 00 01 91.

The device type supplies only a rough classification of the device. The terminal identifier register of the Bus Coupler can be read for detailed identification of the Bus Couplers and the attached terminals (for details see register communication index 0x4500).

Error register

Error register

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1001	0	Error register	Unsigned8	ro	N	0x00	Error register

The 8 bit value is coded as follows:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ManSpec.	reserved	reserved	Comm.	reserved	reserved	reserved	Generic

ManSpec. Manufacturer-specific error, specified more precisely in object 1003.

Comm. Communication error (CAN overrun)

Generic An error that is not more precisely specified has occurred (the flag is set at every error message)

Error store

Error store

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1003	0x00	Predefined error field (Error store)	Unsigned8	rw	N	0x00	Object 1003h contains a description of the error that has occurred in the device - sub-index 0 has the number of error states stored.
	1	Actual error	Unsigned32	ro	N	None	Last error state to have occurred
	...	...	...	--	...	...	...
	10	Standard error field	Unsigned32	ro	N	None	A maximum of 10 error states are stored.

The 32 bit value in the error store is divided into two 16 bit fields:

MSB	LSB
Additional code	Error Code

The additional code contains the error trigger (see emergency object [▶ 88]) and thereby a detailed error description.

New errors are always saved at sub-index 1, all the other sub-indices being appropriately incremented. The whole error store is cleared by writing a 0 to sub-index 0.

If there has not been an error since power up, then object 0x1003 only consists of sub-index 0 with a 0 entered into it. The error store is cleared by a reset or a power cycle.

As is usual in CANopen, the LSB is transferred first, followed by the MSB.

Sync Identifier

Sync Identifier

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1005	0	COB-ID Sync Message	Unsigned32	rw	N	0x80000080	Identifier of the SYNC message

The bottom 11 bits of the 32 bit value contain the identifier (0x80=128 dec). Bit 30 indicates whether the device sends the SYNC telegram (1) or not (0). The CANopen I/O devices receive the SYNC telegram, and accordingly bit 30=0. For reasons of backwards compatibility, bit 31 has no significance.

Sync Interval

Sync Interval

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1006	0	Communication cycle period	Unsigned32	rw	N	0x00000000	Length of the SYNC interval in µs.

If a value other than zero is entered here, the bus node will go into the fault state if, during synchronous PDO operation, no SYNC telegram is received within the watchdog time. The watchdog time corresponds here to 1.5 times the communication cycle period that has been set - the planned SYNC interval can therefore be entered.

The I/O update is carried out at the Beckhoff CANopen bus nodes immediately after reception of the SYNC telegram, provided the following conditions are satisfied:

- Firmware status C0 or above (CANopen Version 4.01 or higher).

- All PDOs that have data are set to synchronous communication (0..240).

- The sync interval has been entered in object 0x1006 and (sync interval x lowest PDO transmission type) is less than 90ms.

The modules are then synchronised throughout.

Device name

Device name

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1008	0	Manufacturer Device Name	Visible String	ro	N	BK51x0, LC5100, IPxxxx-B510 or ILxxxx-B510	Device name of the bus node

Since the returned value is longer than 4 bytes, the segmented SDO protocol is used for transmission.

Hardware version

Hardware version

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1009	0	Manufacturer hardware-version	Visible String	ro	N	-	Hardware version number of the bus node

Since the returned value is longer than 4 bytes, the segmented SDO protocol is used for transmission.

Software version

Software version

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x100A	0	Manufacturer software-version	Visible String	ro	N	-	Software version number of the bus node

Since the returned value is longer than 4 bytes, the segmented SDO protocol is used for transmission.

Node number

Node number

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x100B	0	Node-ID	Unsigned32	ro	N	none	Set node number

The node number is supported for reasons of compatibility.

Guard time

Guard time

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x100C	0	Guard time [ms]	Unsigned16	rw	N	0	Interval between two guard telegrams. Is set by the NMT master or configuration tool.

Life time factor

Life time factor

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x100D	0	Life time factor	Unsigned8	rw	N	0	Life time factor x guard time = life time (watchdog for life guarding)

If a guarding telegram is not received within the life time, the node enters the error state. If the life time factor and/or guard time = 0, the node does not carry out any life guarding, but can itself be monitored by the master (node guarding).

Guarding identifier

Guarding identifier

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x100 E	0	COB-ID guarding protocol	Unsigned32	ro	N	0x000007xy, xy = NodeID	Identifier of the guarding protocol

The guarding identifier is supported for reasons of compatibility. Changing the guarding identifier has no longer been permitted since version 4 of CANopen.

Save parameters

Save parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1010	0	Store Parameter	Unsigned8	ro	N	1	Number of store options
	1	store all parameters	Unsigned32	rw	N	1	Stores all (storable) parameters

By writing the string *save* in ASCII code (hexadecimal 0x65766173) to sub-index 1, the current parameters are placed into non-volatile storage. (The byte sequence on the bus including the SDO protocol: 0x23 0x10 0x10 0x01 0x73 0x61 0x76 0x65).

The storage process takes about 3 seconds, and is confirmed, if successful, by the corresponding TxSDO (0x60 in the first byte). Since the Bus Coupler is unable to send or receive any CAN telegrams during the storage process, saving is only possible when the node is in the pre-operational state. It is recommended that the entire network is placed into the pre-operational state before such storage. This avoids a buffer overflow.

Data saved includes:

- The terminals currently inserted (the number of each terminal category)
- All PDO parameters (identifier, transmission type, inhibit time, mapping).

> NoteThe stored identifiers apply afterwards, not the default identifiers derived from the node addresses. Changes to the DIP switch setting no longer affects the PDOs!

- All SYNC parameters

- All guarding parameters
- Limit values, delta values and interrupt enables for analog inputs

Parameters directly stored in the terminals by way of register communication are immediately stored there in non-volatile form.

Load default values

Load default values

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1011	0	Restore Parameter	Unsigned8	ro	N	4	Number of reset options
	1	Restore all parameters	Unsigned32	rw	N	1	Resets all parameters to their default values
	4	Set manufacturer Defaults	Unsigned32	rw	N	1	Resets all coupler parameters to manufacturer's settings (including registers)

Writing the string *load* in ASCII code (hexadecimal 0x64616F6C) into sub-index 1 resets all parameters to default values (as initially supplied) **at the next boot (reset)**.

(The byte sequence on the bus including the SDO protocol: 0x23 0x11 0x10 0x01 0x6C 0x6F 0x61 0x64).

This makes the default identifiers for the PDOs active again.

Emergency identifier

Emergency identifier

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1014	0	COB-ID Emergency	Unsigned32	rw	N	0x00000080, + NodeID	Identifier of the emergency telegram

The bottom 11 bits of the 32 bit value contain the identifier (0x80=128 dec). The MSBit can be used to set whether the device sends (1) the emergency telegram or not (0).

Alternatively, the bus node's diagnostic function can also be switched off using the *Device diagnostics* bit in the K-Bus configuration (see object 0x4500).

Consumer heartbeat time

Consumer heartbeat time

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1016	0	Number of elements	Unsigned8	ro	N	2	The consumer heartbeat time describes the expected heartbeat cycle time and the node ID of the monitored node
	1	Consumer heartbeat time	Unsigned32	rw	N	0	Watchdog time in ms and node ID of the monitored node

The 32-bit value is used as follows:

MSB		LSB
Bit 31...24	Bit 23...16	Bit 15...0
Reserved (0)	Node ID (unsigned8)	Heartbeat time in ms (unsigned16)

The monitored identifier can be obtained from the node ID by means of the default identifier allocation: Guard-ID = 0x700 + Node-ID.

As is usual in CANopen, the LSB is transferred first, followed by the MSB.

Producer heartbeat time

Producer heartbeat time

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1017	0	Producer heartbeat time	Unsigned16	rw	N	0	Interval in ms between two transmitted heartbeat telegrams

Device identifier (identity object)

Device identifier (identity object)

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1018	0	Identity Object: Number of elements	Unsigned8	ro	N	4	The identity object contains general information about the type and version of the device.
	1	Vendor ID	Unsigned32	ro	N	0x00000002	Manufacturer identifier. Beckhoff has vendor ID 2
	2	Product Code	Unsigned32	ro	N	Depends on the product	Device identifier
	3	Revision Number	Unsigned32	ro	N	-	Version number
	4	Serial Number	Unsigned32	ro	N	-	Production date low word, high byte: calendar week (dec), low word, low byte: calendar year

Product	Product Code
BK5120	0x11400
BK5110	0x113F6
LC5100	0x113EC
IPwxyz-B510	0x2wxyz
IL2301-B510	0x2008FD

Server SDO parameters

Server SDO parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1200	0	Number of elements	Unsigned8	ro	N	2	Communication parameters of the server SDO. Sub-index 0: number of following parameters
	1	COB-ID Client ->Server	Unsigned32	ro	N	0x000006xy, xy=Node-ID	COB-ID RxSDO (Client -> Server)
	2	COB-ID Server ->Client	Unsigned32	ro	N	0x00000580 + Node-ID	COB-ID TxSDO (Client -> Server)

This is contained in the object directory for reasons of backwards compatibility.

Communication parametersfor the 1st RxPDO

for the 1st RxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1400	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the first receive PDO. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x000002xy, xy=Node-ID	COB-ID (Communication Object Identifier) RxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Present for reasons of backwards compatibility, but not used in the RxPDO.
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer. Watchdog time defined for monitoring reception of the PDO.

Sub-index 1 (COB-ID): The bottom 11 bits of the 32 bit value (bits 0-10) contain the CAN identifier. The MSB (bit 31) indicates whether the PDO exists currently (0) or not (1). Bit 30 indicates whether an RTR access to this PDO is permissible (0) or not (1). Changing the identifier (bits 0-10) is not allowed while the object exists (bit 31=0). Sub-index 2 contains the type of the transmission (see introduction to PDOs).

Communication parametersfor the 2nd RxPDO

for the 2nd RxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1401	0	Number of elements	Unsigned8	ro	N	5	Communication parameter for the second receive PDO.
	1	COB-ID	Unsigned32	rw	N	0x000003xy, xy=Node-ID	COB-ID (Communication Object Identifier) RxPDO2
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Present for reasons of backwards compatibility, but not used in the RxPDO.
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer. Watchdog time defined for monitoring reception of the PDO.

Communication parametersfor the 3rd RxPDO

for the 3rd RxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1402	0	Number of elements	Unsigned8	ro	N	5	Communication parameter for the third receive PDO.
	1	COB-ID	Unsigned32	rw	N	0x000004xy, xy=Node-ID	COB-ID (Communication Object Identifier) RxPDO3
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Present for reasons of backwards compatibility, but not used in the RxPDO.
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer. Watchdog time defined for monitoring reception of the PDO.

Communication parametersfor the 4th RxPDO

for the 4th RxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1403	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the fourth receive PDO.
	1	COB-ID	Unsigned32	rw	N	0x000005xy, xy=Node-ID	COB-ID (Communication Object Identifier) RxPDO4
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Present for reasons of backwards compatibility, but not used in the RxPDO.
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer. Watchdog time defined for monitoring reception of the PDO.

Communication parametersfor the 5th-16th RxPDOs

for the 5th-16th RxPDOsCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1404 - 0x140F (depending on the device type)	0	Number of elements	Unsigned8	ro	N	5	Communication parameter for the 5^{th} to 16^{th} receive PDOs.
	1dth="5%"> 1	COB-ID	Unsigned32	rw	N	0x8000000	COB-ID (Communication Object Identifier) RxPDO5...16
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Present for reasons of backwards compatibility, but not used in the RxPDO.
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer. Watchdog time defined for monitoring reception of the PDO.

The number of RxPDOs for each bus node type can be found in the technical data.

Mapping parametersfor the 1st RxPDO

for the 1st RxPDOMapping parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1600	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameter of the first receive PDO; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x62000108	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x62000208	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x62000808	8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The first receive PDO (RxPDO1) is provided by default for digital output data. Depending on the number of outputs inserted, the necessary length of the PDO is automatically determined, and the corresponding objects are mapped. Since the digital outputs are organised in bytes, the length of the PDO in bytes can be found directly at sub-index 0.

Changes to the mapping

The following sequence must be observed in order to change the mapping (specified as from CANopen, version 4):

1. Delete PDO (set bit 31 in the identifier entry (sub-index 1) of the communication parameters to 1)
2. Deactivate mapping (set sub-index 0 of the mapping entry to 0)
3. Change mapping entries (sub-indices 1...8)
4. Activate mapping (set sub-index 0 of the mapping entry to the correct number of mapped objects)
5. Create PDO (set bit 31 in the identifier entry (sub-index 1) of the communication parameters to 0)

Mapping parametersfor the 2nd RxPDO

for the 2nd RxPDOMapping parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1601	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameter of the second receive PDO; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x64110110	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x64110210	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x00000000	8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The second receive PDO (RxPDO2) is provided by default for analog outputs. Depending on the number of outputs inserted, the necessary length of the PDO is automatically determined, and the corresponding objects are mapped. Since the analog outputs are organised in words, the length of the PDO in bytes can be found directly at sub-index 0.

A specific sequence must be observed in order to change the mapping (see object index 0x1600).

Mapping parametersfor the 3rd-16th RxPDO

for the 3rd-16th RxPDOMapping parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1602-0x160F (depending on the device type)	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameters for the third to sixteenth receive PDOs; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x00000000 (see text)	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x00000000 (see text)	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x00000000 (see text)	8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The 3rd to 16th receive PDOs (RxPDO3ff) are automatically given a default mapping by the bus node depending on the attached terminals (or depending on the extension modules). The procedure is described in the section on PDO Mapping [▶ 63].

A specific sequence must be observed in order to change the mapping (see object index 0x1600).

NoteDS401 V2 specifies analog input and/or output data as the default mapping for PDOs 3+4. This corresponds to Beckhoff's default mapping when less than 65 digital inputs or outputs are present. In order to ensure backwards compatibility, the Beckhoff default mapping is retained - the mapping behaviour of the devices therefore corresponds to DS401 V1, where in all other respects they accord with DS401 V2.

Communication parametersfor the 1st TxPDO

for the 1st TxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1800	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the first transmit PDO. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x00000180 + Node-ID	COB-ID (Communication Object Identifier) TxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Repetition delay [value x 100 µs]
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer

Sub-index 1 (COB-ID): The bottom 11 bits of the 32 bit value (bits 0-10) contain the CAN identifier. The MSB (bit 31) indicates whether the PDO exists currently (0) or not (1). Bit 30 indicates whether an RTR access to this PDO is permissible (0) or not (1). Changing the identifier (bits 0-10) is not allowed while the object exists (bit 31=0). Sub-index 2 contains the type of transmission, sub-index 3 the repetition delay between two PDOs of the same type, while sub-index 5 contains the event timer. Sub-index 4 is retained for reasons of compatibility, but is not used. (See also the introduction to PDOs.)

Communication parametersfor the 2nd TxPDO

for the 2nd TxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1801	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the second transmit PDO. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x00000280 + Node-ID	COB-ID (Communication Object Identifier) TxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Repetition delay [value x 100 µs]
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer

The second transmit PDO is provided by default for analog inputs, and is configured for event-driven transmission (transmission type 255). Event-driven mode must first be activated (see object 0x6423), otherwise the inputs can only be interrogated (polled) by remote transmission request (RTR).

Communication parametersfor the 3rd TxPDO

for the 3rd TxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1802	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the third transmit PDO. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x00000380 + Node-ID	COB-ID (Communication Object Identifier) TxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Repetition delay [value x 100 µs]
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer

The third transmit PDO contains analog input data as a rule (see Mapping [▶ 63]). It is configured for event-driven transmission (transmission type 255). Event-driven mode must first be activated (see object 0x6423), otherwise the inputs can only be interrogated (polled) by remote transmission request (RTR).

Communication parametersfor the 4th TxPDO

for the 4th TxPDOCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1803	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the fourth transmit PDO. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x00000480 + Node-ID	COB-ID (Communication Object Identifier) TxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Repetition delay [value x 100 µs]
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer

The fourth transmit PDO contains analog input data as a rule (see Mapping [▶ 63]). It is configured for event-driven transmission (transmission type 255). Event-driven mode must first be activated (see object 0x6423), otherwise the inputs can only be interrogated (polled) by remote transmission request (RTR).

Communication parametersfor the 5th-16th TxPDOs

for the 5th-16th TxPDOsCommunication parameters

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1804-0x180F (depending on the device type)	0	Number of elements	Unsigned8	ro	N	5	Communication parameters for the 5th to 16th transmit PDOs. Sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x0000000	COB-ID (Communication Object Identifier) TxPDO1
	2	Transmission Type	Unsigned8	rw	N	255	Transmission type of the PDO
	3	Inhibit Time	Unsigned16	rw	N	0	Repetition delay [value x 100 µs]
	4	CMS Priority Group	Unsigned8	rw	N	-	Present for reasons of backwards compatibility, but not used.
	5	Event Timer	Unsigned16	rw	N	0	Event-Timer

Mapping 1st TxPDO

Mapping 1st TxPDO

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1A00	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameter of the first transmit PDO; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x60000108	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x60000208	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x60000808	8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The first transmit PDO (TxPDO1) is provided by default for digital input data. Depending on the number of inputs inserted, the necessary length of the PDO is automatically determined, and the corresponding objects are mapped. Since the digital inputs are organised in bytes, the length of the PDO in bytes can be found directly at sub-index 0.

A specific sequence must be observed in order to change the mapping (see object index 0x1600).

Mapping 2nd TxPDO

Mapping 2nd TxPDO

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1A01	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameter of the second transmit PDO; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x64010110	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x64010210	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N		8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The second transmit PDO (TxPDO2) is provided by default for analog input data. Depending on the number of inputs inserted, the necessary length of the PDO is automatically determined, and the corresponding objects are mapped. Since the analog inputs are organised in words, the length of the PDO in bytes can be found directly at sub-index 0.

A specific sequence must be observed in order to change the mapping (see object index 0x1600).

Mapping 3rd-16th TxPDO

Mapping 3rd-16th TxPDO

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x1A02-0x1A0F (depending on the device type)	0	Number of elements	Unsigned8	rw	N	Depending on type and fittings	Mapping parameters for the third to sixteenth transmit PDOs; sub-index 0: number of mapped objects.
	1	1st mapped object	Unsigned32	rw	N	0x00000000 (see text)	1st mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	2	2nd mapped object	Unsigned32	rw	N	0x00000000 (see text)	2nd mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x00000000 (see text)	8th mapped application object (2 bytes index, 1 byte sub-index, 1 byte bit width)

The 3rd to 16th transmit PDOs (TxPDO3ff) are automatically given a default mapping by the bus node depending on the attached terminals (or depending on the extension modules). The procedure is described in the section on PDO Mapping [▸ 63].

A specific sequence must be observed in order to change the mapping (see object index 0x1600).

> NoteDS401 V2 specifies analog input and/or output data as the default mapping for PDOs 3+4. This corresponds to Beckhoff's default mapping when less than 65 digital inputs or outputs are present. In order to ensure backwards compatibility, the Beckhoff default mapping is retained - the mapping behavior of the devices therefore corresponds to DS401 V1, where in all other respects they accord with DS401 V2.

For the sake of completeness, the following object entries are also contained in the object directory (and therefore also in the EDS files):

Index	Meaning
0x2000	Digital inputs (function identical to object 0x6000)
0x2100	Digital outputs (function identical to object 0x6100)
0x2200	1-byte special terminals, inputs (at present no terminals corresponding to this type are included in the product range)
0x2300	1-byte special terminals, outputs (at present no terminals corresponding to this type are included in the product range)
0x2400	2-byte special terminals, inputs (at present no terminals corresponding to this type are included in the product range)
0x2500	2-byte special terminals, outputs (at present no terminals corresponding to this type are included in the product range)
0x2E00	7-byte special terminals, inputs (at present no terminals corresponding to this type are included in the product range)
0x2F00	7-byte special terminals, outputs (at present no terminals corresponding to this type are included in the product range)

3-byte special terminals, input data

3-byte special terminals, input data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2600	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 3-byte special channels, inputs
	1	1st input block	Unsigned24	ro	Y	0x000000	1st input channel
	...	...	...	...	...	...	...
	0X80	128th input block	Unsigned24	ro	Y	0x000000	128th input channel

Example of special terminals with 3-byte input data (in the default setting): KL2502 (PWM outputs, 2 x 3 bytes)

3-byte special terminals, output data

3-byte special terminals, output data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2700	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 3-byte special channels, outputs
	1	1st output block	Unsigned24	rww	Y	0x000000	1st output channel
	...	...	...	...	...	...	...
	0X80	128th output block	Unsigned24	rww	Y	0x000000	128th output channel

Example of special terminals with 3-byte output data (in the default setting): KL2502 (PWM outputs, 2 x 3 bytes)

4-byte special terminals, input data

4-byte special terminals, input data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2800	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 4-byte special channels, inputs
	1	1st input block	Unsigned32	ro	Y	0x00000000	1st input channel
	...	...	...	...	...	...	...
	0X80	128th input block	Unsigned32	ro	Y	0x00000000	128th input channel

Examples of special terminals with 4-byte input data (in the default setting): KL5001, KL6001, KL6021, KL6051

4-byte special terminals, output data

4-byte special terminals, output data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2900	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 4-byte special channels, outputs
	1	1st output block	Unsigned32	rww	Y	0x00000000	1st output channel
	...	...	...	...	...	...	...
	0X80	128th output block	Unsigned32	rww	Y	0x00000000	128th output channel

Examples of special terminals with 4-byte output data (in the default setting): KL5001, KL6001, KL6021, KL6051

5-byte special terminals, input data

5-byte special terminals, input data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2A00	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 5-byte special channels, inputs
	1	1st input block	Unsigned40	ro	Y	0x0000000000	1st input channel
	...	...	...	...	...	...	...
	0X40	64th input block	Unsigned40	ro	Y	0x0000000000	64th input channel

Example of special terminals with 5-byte input data (in the default setting): KL1501

5-byte special terminals, output data

5-byte special terminals, output data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2B00	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 5-byte special channels, outputs
	1	1st output block	Unsigned40	rww	Y	0x0000000000	1st output channel
	...	...	...	...	...	...	...
	0X40	64th output block	Unsigned40	rww	Y	0x0000000000	64th output channel

Example of special terminals with 5-byte output data (in the default setting): KL1501

6-byte special terminals, input data

6-byte special terminals, input data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2C00	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 6-byte special channels, inputs
	1	1st input block	Unsigned48	ro	Y	0x0000000000	1st input channel
	...	...	...	...	...	...	...
	0X40	64th input block	Unsigned48	ro	Y	0x0000000000	64th input channel

Example of special terminals with 6-byte input data (in the default setting): KL5051, KL5101, KL5111

6-byte special terminals, output data

6-byte special terminals, output data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x2D00	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 6-byte special channels, outputs
	1	1st output block	Unsigned48	rww	Y	0x0000000000	1st output channel
	...	...	...	...	...	...	...
	0X40	64th output block	Unsigned48	rww	Y	0x0000000000	64th output channel

Example of special terminals with 6-byte output data (in the default setting): KL5051, KL5101, KL5111

8-byte special terminals, input data

8-byte special terminals, input data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x3000	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 6-byte special channels, inputs
	1	1st input block	Unsigned64	ro	Y	0x0000000000	1st input channel
	...	...	...	...	...	...	...
	0x40	64th input block	Unsigned64	ro	Y	0x0000000000	64th input channel

Example for special terminals with 8-byte input data: KL5101 (with word alignment, not according to the default setting)

8-byte special terminals, output data

8-byte special terminals, output data

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x3100	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available 6-byte special channels, outputs
	1	1st output block	Unsigned64	rww	Y	0x0000000000	1st output channel
	...	...	...	...	...	...	...
	0X40	64th output block	Unsigned64	rww	Y	0x0000000000	64th output channel

Example for special terminals with 8-byte output data: KL5101 (with word alignment, not according to the default setting)

Bus node register communication

Bus node register communication

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x4500	0	Register Access	Unsigned32	rw	N	none	Access to internal bus node registers

The 32 bit value is composed as follows:

MSB			LSB
Access (bit 7) + table number (bits 6...0)	Register number	High byte register value	Low byte register value
[0..1] + [0...0x7F]	[0...0xFF]	[0...0xFF]	[0...0xFF]

As is usual in CANopen, the LSB is transferred first, followed by the MSB.

Accessing index 0x4500 allows any registers in the bus station to be written or read. The channel number and the register are addressed here with a 32 bit data word.

Reading the register value

The coupler must first be informed of which register is to be read. This requires an SDO write access to the appropriate index/sub-index combination, with:
- table number (access bit = 0) in byte 3
- register address in byte 2 of the 32-bit data value.

Bytes 1 and 0 are not evaluated if the access bit (MSB of byte 3) equals 0. The register value can then be read with the same combination of index and sub-index.

After the writing of the register address to be read, the coupler sets the access bit to 1 until the correct value is available. Thus an SDO read access must check that the table number lies in the range from 0...0x7F.

An access error during register communication is indicated by the corresponding return value in the SDO protocol (see the SDO section, Breakdown of parameter communication).

An example of reading register values

It is necessary to determine which baud rate index has been assigned to switch setting 1,1 (DIP 7,8). (See the section covering *Network addresses and baud rates*). To do this, the value in table 100, register 3, must be read. This means that the following SDO telegrams must be sent:

Write access (download request) to index 4500, sub-index 0, with the 32 bit data value 0x64 03 00 00.

Id=0x600+Node-ID DLC=8; Data=23 00 45 00 00 00 03 64

Then a read access (upload request) to the same index/sub-index. The data value sent here is irrelevant (00 is used here).

Id=0x600+Node-ID DLC=8; Data=40 00 45 00 00 00 00 00

The coupler responds with the upload response telegram:

Id=0x580+Node-ID DLC=8; Data=43 00 45 00 04 00 03 64

This tells us that the value contained in this register is 4, and this baud rate index corresponds to 125 kbit/s (the default value).

Writing register values

SDO write access to the corresponding combination of index and sub-index with:
- table number + 0x80 (access bit = 1) in byte 3
- register address in byte 2
- high byte register value in byte 1
- low byte register value in byte 0 of the 32-bit data value.

Remove coupler write protection

Before the registers of the Bus Coupler can be written, the write protection must first be removed. In order to do this, the following values must be written in the given sequence to the corresponding registers:

Step	Table	Register	Value	Corresponding SDO download value (0x4500/0)
1.	99	2	45054 (0xAFFE)	0xE3 02 AF FE (0xE3=0x63(=99)+ 0x80)
2.	99	1	1 (0x0001)	0xE3 01 00 01
3.	99	0	257 (0x0101)	0xE3 00 01 01

Remove coupler write protection (CAN representation)

In order to remove the coupler write protection, the following SDO telegrams (download requests) must thus be sent to the coupler:

Id=0x600+Node-ID DLC=8; Data=23 00 45 00 FE AF 02 E3

Id=0x600+Node-ID DLC=8; Data=23 00 45 00 01 00 01 E3

Id=0x600+Node-ID DLC=8; Data=23 00 45 00 01 01 00 E3

An example of writing register values

After the write protection has been removed, the baud rate index for DIP switch setting 1,1 is to be set to the value 7. This will assign a baud rate of 20 kbaud to this switch setting.

This requires the value 7 to be written into table 100, register 3. This is done with an SDO write access (download request) to index 0x4500, sub-index 0 with the 32 bit value E4 03 00 07 (0xE4 = 0x64+0x80):

Id=0x600+Node-ID DLC=8; Data=23 00 45 00 07 00 03 E4

Identify terminals

The identifier of the coupler (or of the bus station) and of the attached Bus Terminals can be read from the Bus Coupler's table 9. Register 0 then contains the identifier of the Bus Coupler itself, register 1 the identifier of the first terminal and register n the identification of the n^{th} terminal:

Table number	Register number	Description	Value range
9	0	Bus station identifier	0 - 65535
9	1-255	Identifier of the extension module/bus terminal	0 - 65535

The Bus Coupler description in register number 0 contains 5120 = 0x1400 for the BK5120, 5110 = 0x13F6 for the BK5110 and 5100 = 0x13EC for the LC5100. The Fieldbus Box modules contain the identifier 510 dec = 0x1FE in register 0.

In the case of analog and special terminals, the terminal identifier (dec) is contained in the extension module identifier or the terminal description.
Example: if a KL3042 is plugged in as the third terminal, then register 3 contains the value 3042_{dec} (0x0BE2).

The following bit identifier is used for digital terminals:

MSB								LSB							
1	s6	s5	s4	s3	s2	s1	s0	0	0	0	0	0	0	a	e

s6...s1: data width in bits; a=1: output terminal; e=1: input terminal

This identifier scheme results in the terminal descriptions listed below:

Extension module identifier	Meaning
0x8201	2 bit digital input terminal, e.g. KL1002, KL1052, Kl9110, KL9260
0x8202	2 bit digital output terminal, e.g. KL2034, KL2612, KL2702
0x8401	4 bit digital input terminal, e.g. KL1104, KL1124, KL1194
0x8402	4 bit digital output terminal, e.g. KL2124, KL2134, KL2184
0x8403	4 bit digital input/output terminal, e.g. KL2212

General coupler configuration (table 0)

Table 0 of the Bus Coupler contains the data for the general coupler configuration. It is not, as a general rule, necessary to change this; however, for special applications it is possible to change the settings using the KS2000 configuration software, or through direct access via register communication. The write protection must first be removed in order to do this (see above).

The relevant register entries are described below:

K-Bus configuration

Table 0, register 2, contains the K-Bus configuration, and is coded as follows (default value: 0x0006):

MSB								LSB							
0	0	0	0	0	0	0	0	0	0	0	0	0	D	G	A

A: Auto-reset

If there is a K-Bus error, attempts are made cyclically to start the K-Bus up again through a reset. If emergency telegrams and guarding are not evaluated, activation of auto-reset can lead to output and input information being lost without that loss being noticed.

0: No auto-reset (default)

1: Auto-reset active

G: Device diagnostics

Reporting (by means of emergency telegram), that, for example
- a current input is open circuit (with diagnostics)
- 10 V exceeded at a 1-10V input terminal

0: Device diagnostics switched off

1: Device diagnostics active (default)

D: Diagnostic data

from digital terminals is included in the process image (e.g. KL2212). This flag is only evaluated when device diagnostics is active (see above).

0: Do not display

1: Display (default)

Process image description

Table 0, register 3, contains the process image description, and is coded as follows (default value: 0x0903):

MSB								LSB							
0	0	0	0	k1	k0	f1	f0	0	0	a	0	d	k	1	1

k0...k1: Reaction to K-Bus errors

0,2: Inputs remain unchanged (default = 2);

1: Set inputs to 0 (TxPDO with zeros is sent)

f0...f1: Reaction to fieldbus error

0: Stop the K-Bus cycles, watchdog in the terminals triggers, fault output values become active. The old output values are initially set during a restart.

1: Set outputs to 0, then stop the K-Bus cycles (default). 2: Outputs remain unchanged.

a: Word alignment (of analog and special terminals)

0: No alignment (default)

1: Map data to word boundaries (process data always starts on an even address in the PDO)

d: Data format for complex terminals (analog and special terminals)

0: Intel format (default)

1: Motorola format

k: Evaluation of complex terminals (analog and special terminals)

0: User data only (default)

1: Complete evaluation (note: analog channels then, for example, need 3 input and 3 output bytes instead of, e.g., 2 input bytes; instead of 4 channels per PDO, 2 channels require a RxPDO and a TxPDO)

Bus Terminal / Extension Box register communication

Bus Terminal / Extension Box register communication

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x4501	0	Access Terminal Register	Unsigned8	ro	N	none	Index 0x4501 allows access to all the registers in the bus terminal or extension module. Sub-index 0 contains the number of attached bus terminals.
	1	Access Reg. Terminal 1	Unsigned32	rw	N	none	Access to bus terminal or extension module register 1
	...	...	...	...	...	...	...
	0XFE	Access Reg. Terminal 254	Unsigned32	rw	N	none	Access to bus terminal or extension module register 254

The 32 bit value is composed as follows:

MSB			LSB
Access (bit 7) + channel number (bits 6...0)	Register number	High byte register value	Low byte register value
[0..1] + [0...0x7F]	[0...0xFF]	[0...0xFF]	[0...0xFF]

As is usual in CANopen, the LSB is transferred first, followed by the MSB.

Accessing index 0x4501 allows the user registers in the bus terminal or extension module to be written or read. The modules have a set of registers for each input or output channel. The modules are addressed by means of the sub-index; the channel number and register are addressed in the 32-bit data value. Channel number 0 corresponds here to the first channel, 1 to the second channel, and so forth.

Reading the register value

The coupler must first be informed of which register is to be read. This requires an SDO write access to the appropriate index/sub-index combination, with:
- channel number (access bit = 0) in byte 3
- register address in byte 2 of the 32-bit data value.

Bytes 1 and 0 are not evaluated if the access bit (MSB of byte 3) equals 0. The register value can then be read with the same combination of index and sub-index.

After the writing of the register address to be read, the coupler sets the access bit to 1 until the correct value is available. Thus an SDO read access must check that the table number lies in the range from 0...0x7F.

An access error during register communication is indicated by the corresponding return value in the SDO protocol (see the SDO section, Breakdown of parameter communication).

An example of reading register values

The thermocouple type to which the second input channel of a KL3202 Thermocouple Input Terminal has been set is to be determined. This requires feature register 32 to be read. The terminal is located in the fifth slot, next to the Bus Coupler. This means that the following SDO telegrams must be sent:

Write access (download request) to index 4501, sub-index 5 with 32 bit data value 01 20 00 00 (0x01 = 2nd channel, 0x20 = register 32)
Id=0x600+Node-ID DLC=8; Data=23 01 45 05 00 00 20 01

Then a read access (upload request) to the same index/sub-index. The data value sent here is irrelevant (0x00 is used here).
Id=0x600+Node-ID DLC=8; Data=40 01 45 05 00 00 00 00

The coupler responds with the upload response telegram:
Id=0x580+Node-ID DLC=8; Data=43 01 45 05 06 31 20 01

This means that the feature register contains the value 31 06. The upper 4 bits indicate the thermocouple type. Their value here is 3, which means that PT500 is the type that has been set for this channel (see the KL3202 documentation).

Writing register values

SDO write access to the corresponding combination of index and sub-index with:
- channel number + 0x80 (access bit = 1) in byte 3
- register address in byte 2
- high byte register value in byte 1
- low byte register value in byte 0 of the 32-bit data value.

NOTE

WarningIf the write protection is not removed (as a result, for instance, of a faulty codeword), then although a write access to the terminal register will be confirmed (SDO download response), the value is not in fact entered into the register. It is therefore recommended that the value is read back after writing and compared.

Remove terminal write protection

Before the user registers in the Bus Terminal (register 32-xx, depending on terminal type or extension module) can be written to, it is first necessary for write protection to be removed. The following codeword is written for this purpose into register 31 of the channel concerned:

Write protection	Channel	Register	Value	Corresponding SDO download value (0x4500/0)
	1,2, 3 or 4	31 (0x1F)	4661 (0x1235)	8y 1F 12 35 (y = channel number)

Remove terminal write protection (CAN representation)

In order to remove the terminal's write protection, the following SDO telegram must thus be sent to the coupler:

Id=600 + Node-ID DLC=8; Data=23 01 45 xx 35 12 1F 8y

where xx is the terminal's slot, and y indicates the channel.

An example of removing write protection

Suppose that a KL3202 Thermocouple Input Terminal is inserted into slot 5 of a BK5120 that has node address 3, then the write protection for the first channel can be removed as follows:

Id=0x603 DLC=8; Data=23 01 45 05 35 12 1F 80

The following telegram is sent for the second channel:

Id=0x603 DLC=8; Data=23 01 45 05 35 12 1F 81

An example of writing register values

The type of thermocouple attached to the second channel of the KL3202 Terminal in slot 5 is now to be changed to PT1000. For this purpose, the value 2 must be written into the upper 4 bits (the upper nibble) of the feature register. It is assumed to that the default values are to be supplied for all the other bits in the feature register. Once the write protection has been removed, SDO write access (download request) is used to write the following 32 bit value into index 0x4501, sub-index 05: 81 20 21 06 (0x81=01+0x80; 0x20=32;0x2106 = register value).

The corresponding telegram on the bus looks like this:

Id=0x600+Node-ID DLC=8; Data=23 01 45 05 06 21 20 81

Activate PDOs

Activate PDOs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x5500	0	Activate PDO Defaults	Unsigned32	rw	N	0x00000000	sets PDO communication parameters for PDOs 2...11

CANopen defines default identifiers for 4 transmit (Tx) and 2 receive (Rx) PDOs, all other PDOs being initially deactivated after the nodes have started up. Index 0x5500 can activate all the PDOs that, in accordance with the terminals inserted, are filled with process data (manufacturer-specific default mapping). A manufacturer-specific default identifier allocation is carried out here for PDO5…11, while the transmission type and a uniform inhibit time is set for PDO2…11. PDOs that do not have process data (and which are thus superfluous in the present configuration) are not activated.

NoteThis object can only be written in the pre-operational state!

The 32-bit value is used as follows:

MSB			LSB
Transmission Type RxPDOs	Transmission Type TxPDOs	High byte inhibit time	Low byte inhibit time

As is usual in CANopen, the LSB is transferred first, followed by the MSB.

Example

Activate PDOs for bus node number 1, set inhibit time to 10ms (=100 x 100µs), set transmission type for TxPDOs to 255, and set transmission type for RxPDOs to 1. The following telegram must be sent:
Id=0x601 DLC=8; Data=23 00 55 00 64 00 FF 01

The node responds with the following telegram:
Id=0x601 DLC=8; Data=60 00 55 00 00 00 00 00

Identifiers used

The default identifier allocation for the additional PDOs leaves the pre-defined regions for guarding, SDOs etc. free, assumes a maximum of 64 nodes in the network with PDO6 as the next node, and proceeds according to the following scheme:

Object	Function code	Resulting COB ID (hex)	Resulting COB ID (dec)
TxPDO5	1101	0x681 - 0x6BF	1665 - 1727
RxPDO5	1111	0x781 - 0x7BF	1921- 1983
TxPDO6	00111	0x1C1 - 0x1FF	449 - 511
RxPDO6	01001	0x241 - 0x27F	577 - 639
TxDPO7	01011	0x2C1 - 0x2FF	705 - 767
RxPDO7	01101	0x341 - 0x37F	833 - 895
TxPDO8	01111	0x3C1- 0x3FF	961 - 1023
RxPDO8	10001	0x441 - 0x47F	1089 - 1151
TxPDO9	10011	0x4C1 - 0x4FF	1217 - 1279
RxPDO9	10101	0x541 - 0x57F	1345 - 1407
TxDPO10	10111	0x5C1 - 0x5FF	1473 - 1535
RxPDO10	11001	0x641 - 0x67F	1601- 1663
TxPDO11	11011	0x6C1 - 0x6FF	1729 - 1791
RxPDO11	11101	0x741 - 0x77F	1857 - 1919

NOTE

WarningEnsure that index 0x5500 is not used if Bus Couplers with more than 5 PDOs are present in networks with node addresses > 64, otherwise identification overlaps can occur. In that case, the PDO identifiers must be set individually.

For the sake of clarity, the default identifiers defined according to CANopen are also listed here:

Object	Function code	Resulting COB ID (hex)	Resulting COB ID (dec)
Emergency	0001	0x81 - 0xBF [0xFF]	129 - 191 [255]
TxPDO1	0011	0x181 - 0x1BF [0x1FF]	385 - 447 [511]
RxPDO1	0100	0x201 - 0x23F [0x27F]	513 - 575 [639]
TxPDO2	0101	0x281 - 0x2BF [0x2FF]	641 - 676 [767]
RxPDO2	0110	0x301 - 0x33F [0x37F]	769 - 831 [895]
TxDPO3	0111	0x381 - 0x3BF [0x3FF]	897 - 959 [1023]
RxPDO3	1000	0x401 - 0x43F [0x47F]	1025 - 1087 [1151]
TxPDO4	1001	0x481 - 0x4BF [0x4FF]	1153 - 1215 [1279]
RxPDO4	1010	0x501 - 0x53F [0x57F]	1281- 1343 [1407]
SDO (Tx)	1011	0x581 - 0x5BF [0x5FF]	1409 - 1471 [1535]
SDO (Rx)	1100	0x601 - 0x63F [0x67F]	1537 - 1599 [1663]
Guarding / Heartbeat/ Bootup	1110	0x701 - 0x73F [0x77F]	1793 - 1855 [1919]

The identifiers that result from the DIP switch settings on the coupler are given, as are the identifier regions for the node addresses 64...127 (not settable in Bus Couplers BK5110, BK5120 and LC5100) in square brackets. Addresses 1…99 can be set for the Fieldbus Box modules and the BK515x Bus Couplers.

The appendix [▸ 65] contains a tabular summary of all the identifiers.

Digital inputs

Digital inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6000	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available digital 8-bit input data blocks
	1	1st input block	Unsigned8	ro	Y	0x00	1st input channel
	...	...	...	...	...	...	...
	0XFE	254th input block	Unsigned8	ro	Y	0x00	254th input channel

Interrupt mask

Interrupt mask

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6126	0	Number of elements	Unsigned8	ro	N	Depending on type	The number of 32-bit interrupt masks = 2 x the number of TxDPOs
	1	IR-Mask0 TxPDO1	Unsigned32	rw	N	0xFFFFFFFF	IR-mask bytes 0...3 TxPDO1
	2	IR-Mask1 TxPDO1	Unsigned32	rw	N	0xFFFFFFFF	IR-mask bytes 4...7 TxPDO1
	3	IR-Mask0 TxPDO2	Unsigned32	rw	N	0xFFFFFFFF	IR-mask bytes 0...3 TxPDO2
	...	...	...	...	...	...	...
	0x20	IR-Mask1 TxPDO16	Unsigned32	rw	N	0xFFFFFFFF	IR-mask bytes 4...7 TxPDO16

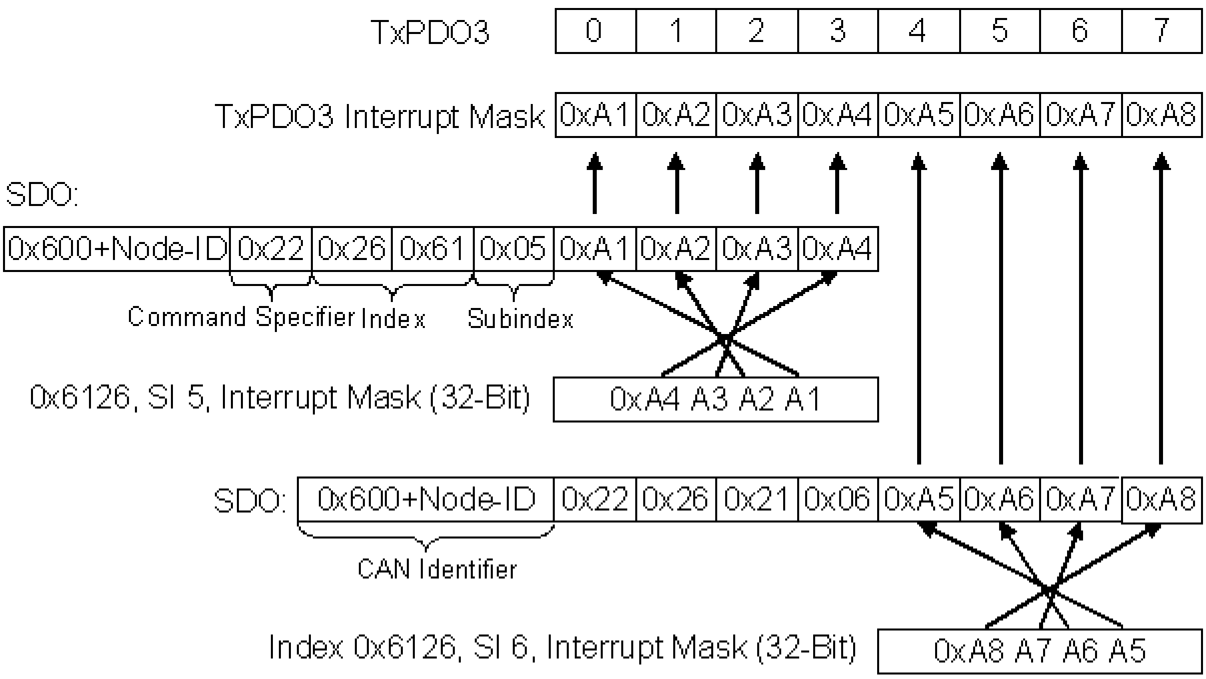
By default, every change in the value in an event-driven PDO causes a telegram to be sent. The interrupt mask makes it possible to determine which data changes are evaluated for this purpose. By clearing the appropriate ranges within the PDOs they are masked out for event-driving purposes (interrupt control). The interrupt mask does not just govern all the PDOs with digital inputs, but all the TxPDOs that are present. If the TxPDOs are shorter than 8 bytes, then the superfluous part of the IR mask is not evaluated.

The interrupt mask only has an effect on TxPDOs with transmission types 254 and 255. It is not stored in the device (not even through object 0x1010). Changes to the mask at runtime (when the status is operational) are possible, and are evaluated starting from the next change of input data.

The interrupt mask for TxPDOs with analog input data is not evaluated if either limit values (0x6424, 0x6425) or the delta function (0x6426) have been activated for the inputs.

This entry has been implemented in firmware C3 and above.

Example of data assignment



Application example

The value contained in a fast counter input is only to be transmitted when bits in the status word (the latch input, for instance) have changed. This requires the 32 bit counter value to be masked out (zeroed) in the interrupt mask. The status is located in byte 0, while the counter value is, by default, contained in bytes or 1..4 of the corresponding PDOs (TxPDO3 in this example, because < 65 digital and < 5 analog inputs are present).
This means that index 0x6126, sub-index5 must receive the value 0x0000 00FF and that sub-index6 must have 0xFFFF FF00 written into it.

The corresponding SDOs therefore appear as follows:

11 bit identifier	8 bytes of user data							
0x600+ node ID	0x22	0x26	0x61	0x05	0xFF	0x00	0x00	0x00

11 bit identifier	8 bytes of user data							
0x600+ node ID	0x22	0x26	0x61	0x06	0x00	0xFF	0xFF	0xFF

Digital outputs

Digital outputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6200	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of available digital 8-bit output data blocks
	1	1st input block	Unsigned8	rw	Y	0x00	1st output channel
	...	...	...	...	...	...	...
	0XFE	254th input block	Unsigned8	rw	Y	0x00	254th output channel

Analog inputs

Analog inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6401	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of analog input channels available
	1	1st input	Unsigned16	ro	Y	0x0000	1st input channel
	...	...	...	...	...	...	...
	0XFE	254th input	Unsigned16	ro	Y	0x0000	254th input channel

The analog signals are displayed left aligned. The representation in the process image is therefore independent of the actual resolution. Detailed information on the data format can be found at the relevant signal type.

Analog outputs

Analog outputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6411	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of analog output channels available
	1	1st input block	Unsigned16	rw	Y	0x0000	1st output channel
	...	...	...	...	...	...	...
	0XFE	254th input block	Unsigned16	rw	Y	0x0000	254th output channel

The analog signals are displayed left aligned. The representation in the process image is therefore independent of the actual resolution. Detailed information on the data format can be found at the relevant signal type.

Event driven analog inputs

Event driven analog inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6423	0	Global Interrupt Enable	Boolean	rw	N	FALSE (0)	Activates the event-driven transmission of PDOs with analog inputs.

Although, in accordance with CANopen, the analog inputs in TxPDO2..4 are by default set to transmission type 255 (event driven), the event (the alteration of an input value) is suppressed by the event control in object 0x6423, in order to prevent the bus from being swamped with analog signals. It is recommended that the flow of data associated with the analog PDOs is controlled either through synchronous communication or through using the event timer. In event-driven operation, the transmission behavior of the analog PDOs can be parameterized before activation by setting the inhibit time (object 0x1800ff, sub-index 3) and/or limit value monitoring (objects 0x6424 + 0x6425) and/or delta function (object 0x6426).

Upper limit value analog inputs

Upper limit value analog inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6424	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of analog input channels available
	1	upper limit 1st input	Unsigned16	rw	Y	0x0000	Upper limit value for 1st input channel
	...	...	...	...	...	...	...
	0XFE	upper limit 254th input	Unsigned16	rw	Y	0x0000	Upper limit value for 254th input channel

Values different from 0 activate the upper limit value for this channel. A PDO is then transmitted if this limit value is exceeded. In addition, the event driven mode must be activated (object 0x6423). The data format corresponds to that of the analog inputs.

Lower limit value analog inputs

Lower limit value analog inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6425	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of analog input channels available
	1	lower limit 1st input	Unsigned16	rw	Y	0x0000	Lower limit value for 1st input channel
	...	...	...	...	...	...	...
	0XFE	lower limit 254th input	Unsigned16	rw	Y	0x0000	Lower limit value for 254th input channel

Values different from 0 activate the lower limit value for this channel. A PDO is then transmitted if the value falls below this limit value. In addition, the event driven mode must be activated (object 0x6423). The data format corresponds to that of the analog inputs.

Delta function for analog inputs

Delta function for analog inputs

Index	Sub-index	Name	Type	Attribute	Mapping	Default value	Meaning
0x6426	0	Number of elements	Unsigned8	ro	N	Depending on type and fittings	Number of analog input channels available
	1	delta value 1st input	Unsigned16	rw	Y	0x0000	Delta value for the 1st input channel
	...	...	...	...	...	...	...
	0XFE	delta value 254th input	Unsigned16	rw	Y	0x0000	Delta value for the 254th input channel

Values different from 0 activate the delta function for this channel. A PDO is then transmitted if the value has changed by more than the delta value since the last transmission. In addition, the event driven mode must be activated (object 0x6423). The data format corresponds to that of the analog inputs (delta value: can only have positive values).

2.3.9.3 Objects and Data of the BX5100/BC5150

Access to located flags

Access to located flags

Index	sub index	Name	Type	Attrb.	Map.	Default Value	Meaning
0x2F00	0	Number of sub index	Unsigned8	ro	N	128	Number of sub index
	1	Flag	Unsigned32	rw	N	none	Flag %MB0..3
	2	Flag	Unsigned32	rw	N	none	Flag %MB4..7
	...	...	...	...	...	...	...
	128	Flag	Unsigned32	rw	N	none	Flag %MB508..511

Index	sub index	Name	Type	Attrb.	Map.	Default Value	Meaning
0x2F01-0x2F07	0	Number of sub index	Unsigned8	ro	N	128	Number of sub index
	1	Flag	Unsigned32	rw	N	none	Flag %MBx..x+4
	2	Flag	Unsigned32	rw	N	none	Flag %MBy..y+4
	...	...	...	...	...	...	...
	128	Flag	Unsigned32	rw	N	none	Flag %MBz...z +14

AMS NetId

NetId

Index	sub index	Name	Type	Attrb.	Map.	Default Value	Meaning
0x5FFE	0	NetId	String	rw	N	1.1.1.1.1.1	AMS Net Id

2.3.10 ADS-Communication

2.3.10.1 ADS Services

Local process image task 1 port 800 or port 801

Data can be read from and written to the local process image. If it is necessary for outputs to be written, it is important to ensure that they are not used by the local PLC, because the local controller will overwrite these values. The data is not associated with a watchdog, and therefore must not be used for outputs that would have to be switched off in the event of a fault.

Index group	Meaning	Index offset (value range)
0xF020	Inputs	0...2047
0xF030	Outputs	0...2047
0x4020	Flags	0...4095

ADS services

Table 2: AdsServerAdsState

Data type (read only)	Meaning
String	Start - the local PLC is running Stop - the local PLC is stopped

Table 3: AdsServerDeviceState

Data type (read only)	Meaning
INT	0: Start - the local PLC is running 1: Stop - the local PLC is stopped

Table 4: AdsServerType

Data type (read only)	Meaning
String	BX PLC Server

Register port 100

The ADS port number in the Bus Terminal Controller for register communication is fixed, being set at 100.

Index group	Index offset (value range)		Meaning
	Hi-Word	Low Word	
0 [READ ONLY]	0...127	0...255	Registers in the Bus Coupler High word, table number of the Bus Coupler Low word, register number of the table
1...255	0...3	1...255	Register of the Bus Terminal High word, channel number Low word, register number of the Bus Terminal

NoteNote when reading the register that the time out for the ADS block is set to a time longer than 1 second.

NoteNote when writing to the registers that the password is set (see the documentation for the particular Bus Terminal).

3 Transport

3.1 Unpacking, installation and transport

The specified storage conditions must be adhered to (see "Technical data").

Dimensions and weight of the individual modules:

Module type	CX1500-M510	CX1500-B510
Dimensions (B x H x T)	38 mm x 100 mm x 91 mm	38 mm x 100 mm x 91 mm
weight	190 g	190 g

Unpacking

Proceed as follows to unpack the unit:
1. Remove packaging.
2. Do not discard the original packaging. Keep it for future relocation.
3. Check the delivery for completeness by comparing it with your order.
4. Please keep the associated paperwork. It contains important information for handling the unit.
5. Check the contents for visible shipping damage.
6. If you notice any shipping damage or inconsistencies between the contents and your order, you should notify Beckhoff Service.

NOTE
Danger of damage to the unit!

During transport in cold conditions, or if the unit is subjected to extreme temperature swings, condensation on and inside the unit must be avoided.
Prior to operation, the unit must be allowed to slowly adjust to room temperature. Should condensation occur, a delay time of approximately 12 hours must be allowed before the unit is switched on.

Installation

The devices are designed for installation in control cabinets. You will find installation instructions in the chapter mechanical mounting.

Shipping and relocation

Despite the robust design of the unit, the components are sensitive to strong vibrations and impacts. During transport, your computer should therefore be protected from excessive mechanical stress. Therefore, please use the original packaging.

4 Assembly and connecting

4.1 Mechanical assembly

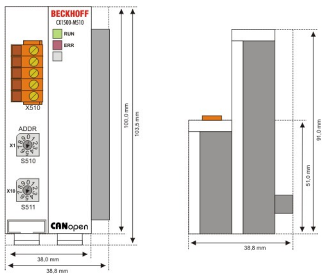
4.1.1 Dimensions

The CX10x0 product range is characterized by small overall installed size and high modularity. For project planning purposes, a CPU module, a power supply unit and the associated system interfaces and fieldbus interfaces have to be provided. The overall width of the application is made up of the individual modules. With a height of 100 mm, the module dimensions exactly match those of the Beckhoff Bus Terminals. Together with the lowered connector surfaces, this means that it can be used in a standard terminal box with a height of 120 mm.

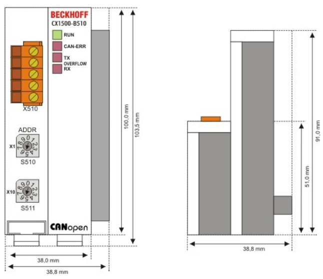
CX1500-M510 und CX1500-B510 Fieldbus connections

The modules for fieldbus connection have the dimension of 38 x 100 x 91 mm. Master- and Slave connections have the same dimensions.

CX1500-M510:



CX1500-B510:

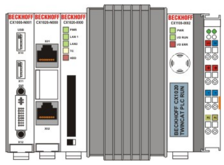


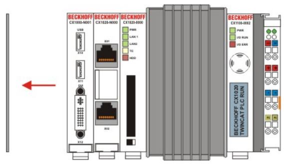
4.1.2 Mechanical installation of the fieldbus connection

Installation of a fieldbus connection involves several steps:

1. Removing the cover of the basic CX1020 module

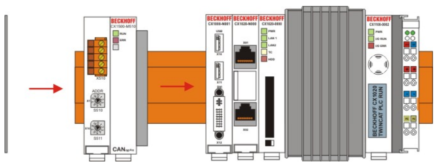
In order to be able to connect the fieldbus to the basic CX1020 module, the cover of the basic CX1000/ CX1020 module has to be removed first. This is achieved by applying slight pressure on the cover.





2. Assembly and connection to the CX1000/CX1020 configuration

Since the CX1000 configuration is already positioned on the top-hat rail, the assembly first has to be pushed onto the top-hat rail. To this end the latching mechanism has to be released by pulling the white straps downwards.
The assembly is connected to the existing CX1000/CX1020 configuration by simply plugging the two units together. Care must be taken that the plug of the PC104 interface is not damaged.
When correctly assembled, no significant gap can be seen between the attached housings. Finally, the white straps are returned to their original position, so that the locking mechanism engages.



3. Install cover

If the connection area does not have a closing cover on the left-hand side, the cover that was previously removed should be pressed over the connections until it audibly engages.

Note:

If the CX1000/CX1020 configuration is not positioned on the top-hat rail, it is possible to connect the assembly with the CX1000/CX1020 configuration first and then latch the whole module onto the top-hat rail. The installation is described in section Installation and wiring [▶ 178].

Note:

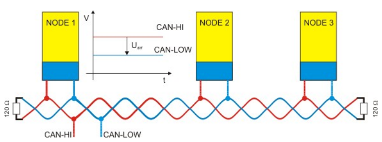
A locking mechanism prevents the individual housings from being pulled off again.

4.1.3 Connections

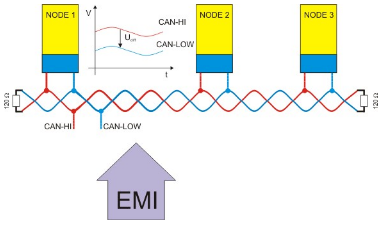
CAN topology

CAN topology

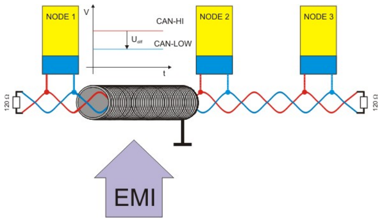
CAN is a 2-wire bus system, to which all participating devices are connected in parallel (i.e. using short drop lines). The bus must be terminated at each end with a 120 (or 121) Ohm terminating resistor to prevent reflections. This is also necessary even if the cable lengths are very short!



Since the CAN signals are represented on the bus as the difference between the two levels, the CAN leads are not very sensitive to incoming interference (EMI): Both leads are affected, so the interference has very little effect on the difference.



An additional shield surrounding the pair of signal wires can help to eliminate the disturbing influence of EMI.



Bus length

Bus length

The maximum length of a CAN bus is primarily limited by the signal transit time. The multi-master bus access procedure (arbitration) requires signals to reach all the nodes at effectively the same time (before the sampling within a bit period). Since the signal transit times in the CAN connecting equipment (transceivers, opto-couplers, CAN controllers) are almost constant, the line length must be chosen in accordance with the baud rate:

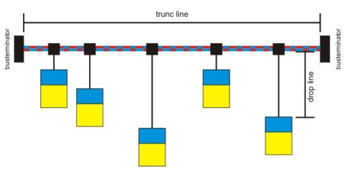
Baud Rate	Bus length
1 Mbit/s	< 20 m*
500 kbit/s	< 100 m
250 kbit/s	< 250 m
125 kbit/s	< 500 m
50 kbit/s	< 1000 m
20 kbit/s	< 2500 m
10 kbit/s	< 5000 m

*) A figure of 40m at 1 Mbit/s is often found in the CAN literature. This does not, however, apply to networks with optically isolated CAN controllers. The worst case calculation for opto-couplers yields a figure 5 m at 1 Mbit/s - in practice, however, 20 m can be reached without difficulty.

It may be necessary to use repeaters for bus lengths greater than 1000 m.

Drop lines

Drop lines



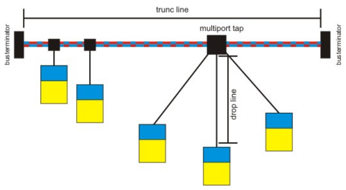
Drop lines must always be avoided as far as possible, since they inevitably cause reflections. The reflections caused by drop lines are not however usually critical, provided they have decayed fully before the sampling time. In the case of the bit timing settings selected in the Bus Couplers it can be assumed that this is the case, provided the following drop line lengths are not exceeded:

Baud Rate	Drop line length	Total length of all drop lines
1 Mbit/s	< 1m	< 5 m
500 kbit/s	< 5 m	< 25 m
250 kbit/s	< 10m	< 50 m
125 kbit/s	< 20m	< 100 m
50 kbit/s	< 50m	< 250 m

Drop lines must not have terminating resistors.

Star Hub (Multiport Tap)

Star Hub



Shorter drop line lengths must be maintained when passive distributors ("multiport taps"), such as the Beckhoff ZS5052-4500 Distributor Box. The following table indicates the maximum drop line lengths and the maximum length of the trunk line (without the drop lines):

Baud Rate	Drop line length with multiport topology	Trunk line length (without drop lines)
1 Mbit/s	< 0,3 m	< 25 m
500 kbit/s	< 1,2 m	< 66 m
250 kbit/s	< 2,4 m	< 120 m
125 kbit/s	< 4.8 m	< 310 m

CAN cable

CAN cable

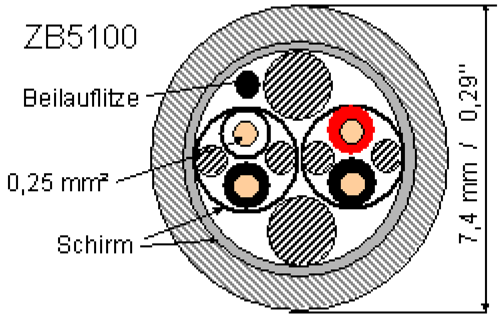
Screened twisted-pair cables (2x2) with a characteristic impedance of between 108 and 132 Ohm is recommended for the CAN wiring. If the CAN transceiver's reference potential (CAN ground) is not to be connected, the second pair of conductors can be omitted. (This is only recommended for networks of small physical size with a common power supply for all the participating devices).

ZB5100 CAN Cable

ZB5100

A high quality CAN cable with the following properties is included in Beckhoff's range:

- 2 x 2 x 0.25 mm² (AWG 24) twisted pairs, cable colors: red/black + white/black
- double screened
- braided screen with filler strand (can be attached directly to pin 3 of the 5-pin connection terminal),
- flexible (minimum bending radius 35 mm when bent once, 70 mm for repeated bending)
- characteristic impedance (60 kHz): 120 Ohm
- conductor resistance < 80 Ohm/km
- sheath: grey PVC, external diameter 7.3 +/- 0.4 mm
- Weight: 64 kg/km.
- printed with "BECKHOFF ZB5100 CAN-BUS 2x2x0.25" and meter marking (length data every 20 cm)

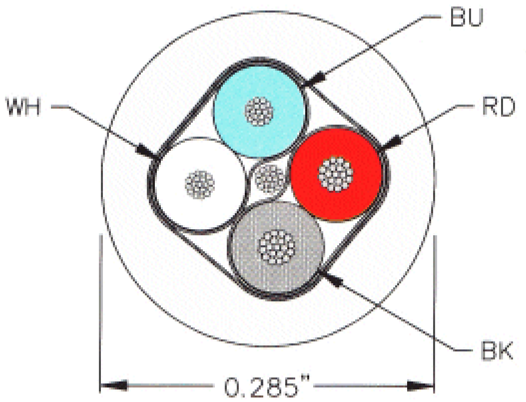


ZB5200 CAN/DeviceNet Cable

ZB5200

The ZB5200 cable material corresponds to the DeviceNet specification, and is also suitable for CANopen systems. The ready-made ZK1052-xxxx-xxxx bus cables for the Fieldbus Box modules are made from this cable material. It has the following specification:

- 2 x 2 x 0.34 mm² (AWG 22) twisted pairs
- double screened braided screen with filler strand
- characteristic impedance (1 MHz): 126 Ohm
- conductor resistance 54 Ohm/km
- sheath: grey PVC, external diameter 7.3 mm
- printed with "InterlinkBT DeviceNet Type 572" as well as UL and CSA ratings
- stranded wire colours correspond to the DeviceNet specification
- UL recognized AWM Type 2476 rating
- CSA AWM I/II A/B 80°C 300V FT1
- corresponds to the DeviceNet "Thin Cable" specification



Screening

Screening

The screen is to be connected over the entire length of the bus cable, and only galvanically grounded at one point, in order to avoid ground loops.
The design of the screening, in which HF interference is diverted through R/C elements to the mounting rail assumes that the rail is appropriately earthed and free from interference. If this is not the case, it is possible that HF interference will be transmitted from the mounting rail to the screen of the bus cable. In that case the screen should not be attached to the couplers - it should nevertheless still be fully connected through.

Notes related to checking the CAN wiring can be found in the Trouble Shooting section.

Cable colors

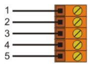
Cable colors

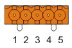
Suggested method of using the Beckhoff CAN cable on Bus Terminal and Fieldbus Box:

BK51x0 pin	Fieldbus Box pin	FC510x pin	Function	ZB5100 cable color	ZB5200 cable color
1	3	3	CAN Ground	**black**/ (red)	**black**
2	5	2	CAN Low	**black**	**blue**
3	1	5	Screen	Filler strand	Filler strand
4	4	7	CAN high	**white**	**white**
5	2	9	not used	**(red)**	**(red)**

5-pin Open Style Connector

5-pin Open Style Connector





The left figure shows the socket in the BK51x0 Bus Coupler. Pin 5 is the connection strip's top most pin. Pin 5 is not used. Pin 4 is the CAN high connection, pin 2 is the CAN low connection, and the screen is connected to pin 3 (which is connected to the mounting rail via an R/C network). CAN GND can optionally be connected to pin 1. If all the CAN ground pins are connected, this provides a common reference potential for the CAN transceivers in the network. It is recommended that the CAN GND be connected to earth at one location, so that the common CAN reference potential is close to the supply potential. Since the CANopen BK51X0 Bus Couplers provide full electrical isolation of the bus connection, it may in appropriate cases be possible to omit wiring up the CAN ground.

ZS1052-3000 Bus Interface Connector

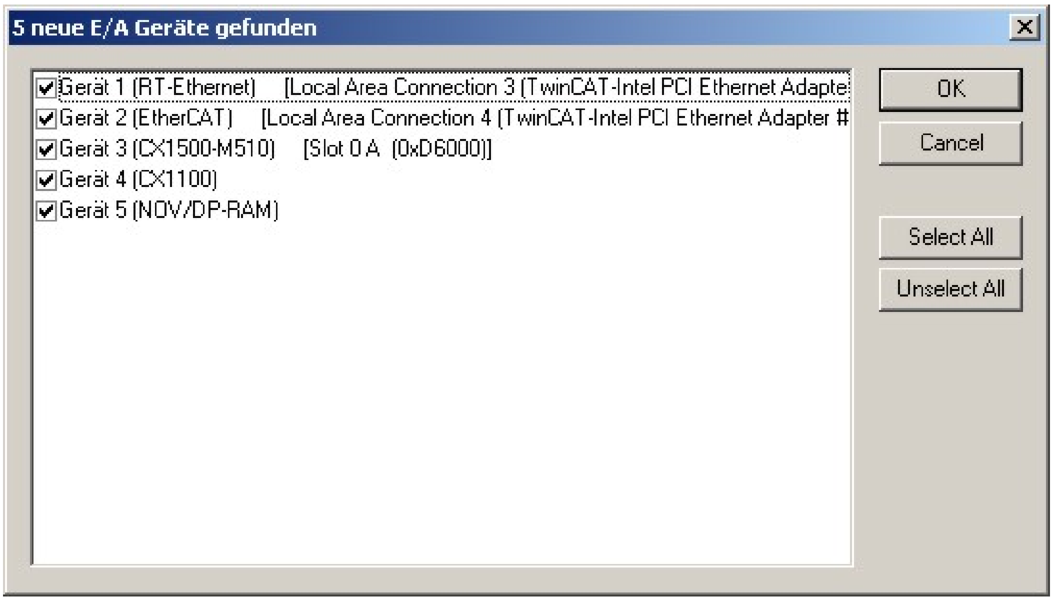
ZS1052-3000 Bus Interface Connector

The ZS1052-3000 CAN Interface Connector can be used as an alternative to the supplied connector. This makes the wiring significantly easier. There are separate terminals for incoming and outgoing leads and a large area of the screen is connected via the strain relief. The integrated terminating resistor can be switched externally. When it is switched on, the outgoing bus lead is electrically isolated - this allows rapid wiring fault location and guarantees that no more than two resistors are active in the network.

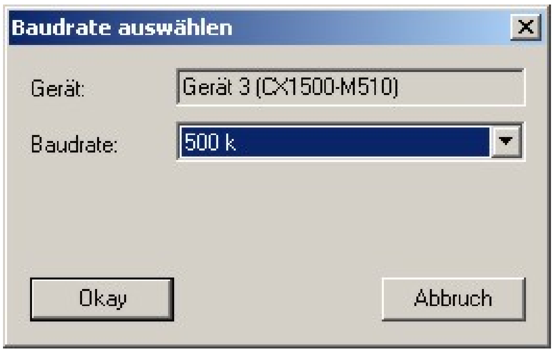
4.2 Software Setup

4.2.1 Setup of CX1500-M510 for CANopen

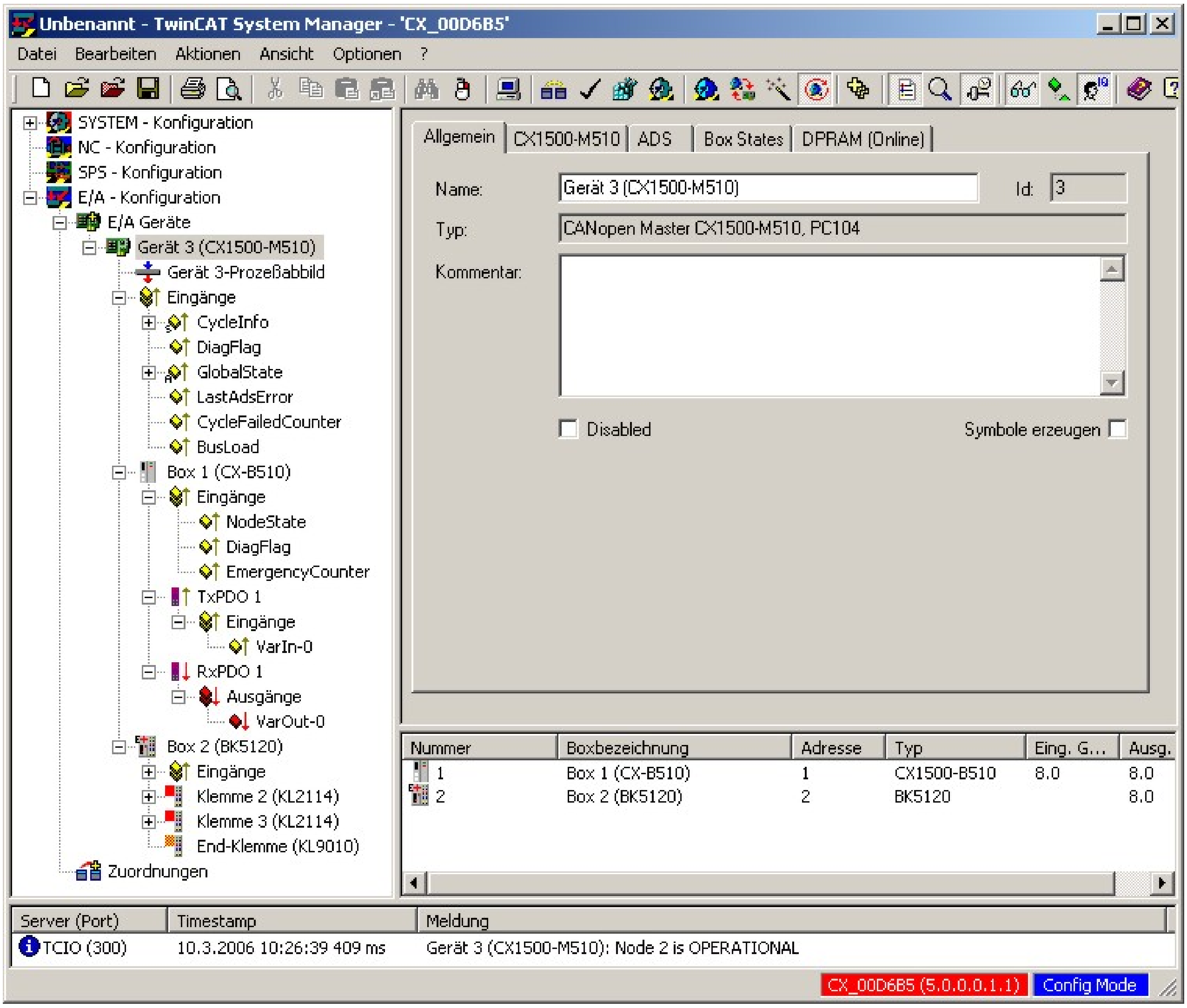
The module CX1500-M310 (master connection for Profibus) can, as all other modules of the CX-system, be accessed via the TwinCAT System Manager. The user selects the desired system and search for units in the System Manager (click with right button on I/O Units - this operation is only available in config-mode) After some seconds the connected units are listed. The user selects the devices he / she wants to make use of in the program.



Next the scan for boxes is requested. Answering "yes" starts the scan for boxes connected to the fieldbus. Before a bus scan can be initiated the transfer rate must be selected. The following dialog box opens the ask for the baud rate. The transfer speed depends on the net topology. Details are described in the chapter "wires and installation".



In some cases not all boxes can be found. Restart scan to find the missing boxes. If there are still boxes missing please check cables and operation mode of the boxes (the mode should be operational) If two or more devices on the bus have the same node-id none is found. Change the node-id and scan again. If the scan is complete the founded boxes are shown:

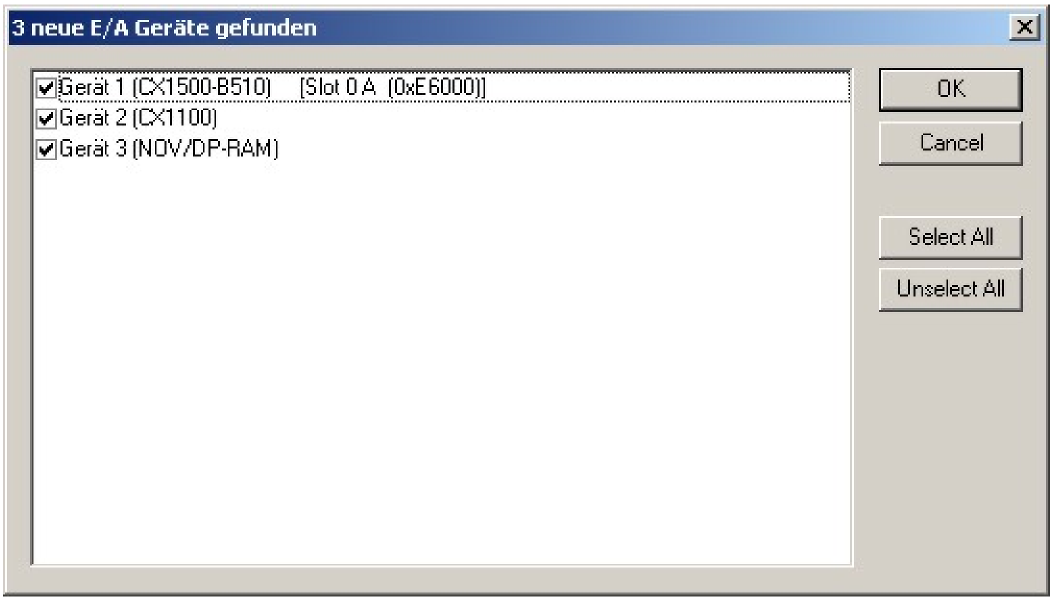


The status of the CANopen bus is shown in the upper region of CX1500-M310 entry. The programmer can connect these signals with variables in PLC to analyze them in the program (e.g. error codes for saving states or registers). Below "BOX1" is a status field (NodeState, DiagFlag und EmergencyCounter)) for transfer diagnosis. Next are the input / output signals. The tree shows the Process-Data-Objects (PDO) that are created in the process image of the salve module. The variables can be accessed by mapping them to the PLC program. The direction represents the view of the slave: inputs are read by TX-channel / outputs are written to the RX-channel. If the box is a bus coupler (see Box 2) the available bus terminals are displayed. The number of the box represents the bus address . In "Free-Run-Mode" of TwinCAT the bus function can be checked easily: Just force a signal by "write online" and see if the control LED ion the terminal lights up.

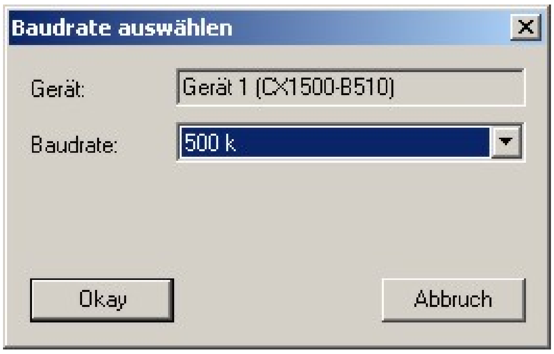
If the PLC project is loaded the signals can be connected with the variables by double click in signal. (For further details see documentation on TwinCAT). The connected signals are marked by a small arrow on the signal icon. If all signals and variables are connected the new configuration can be saved and loaded onto the CX-System. The PLC program is started via PLC control.

4.2.2 Startup of CX1500-B310 for CANopen

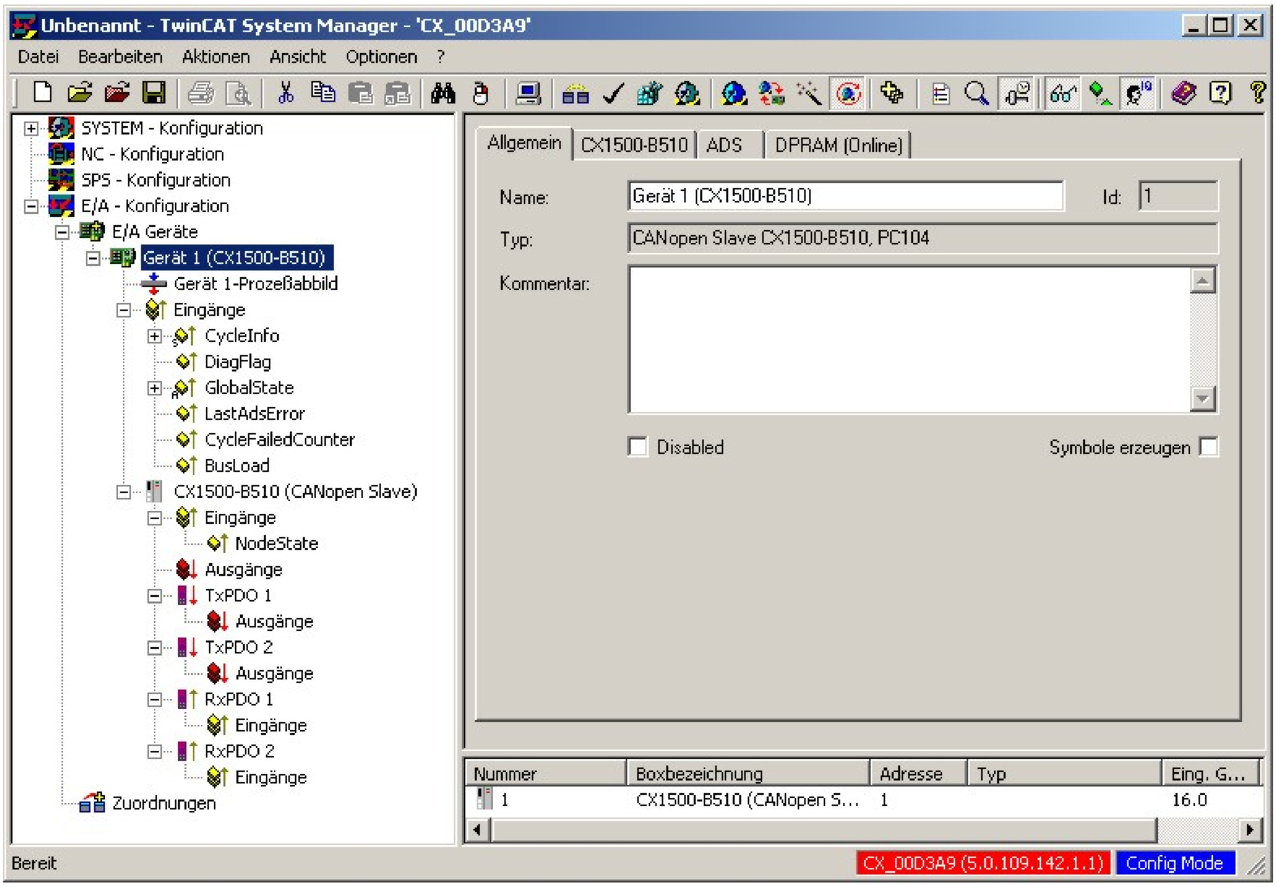
The module CX1500-B310 (salve connection for Profibus) can, as all other modules of the CX-system, be accessed via the TwinCAT System Manager. The user selects the desired system and search for units in the System Manager (click with right button on I/O Units - this operation is only available in config-mode) After some seconds the connected units are listed. The user selects the devices he / she wants to make use of in the program.



Next the scan for boxes is questioned. Answering with "yes" opens the following dialog. Before the CANopen bus can be scanned for other devices the baud rate must be selected. The transfer speed depends on the net topology. Details are described in the chapter "wires and installation".

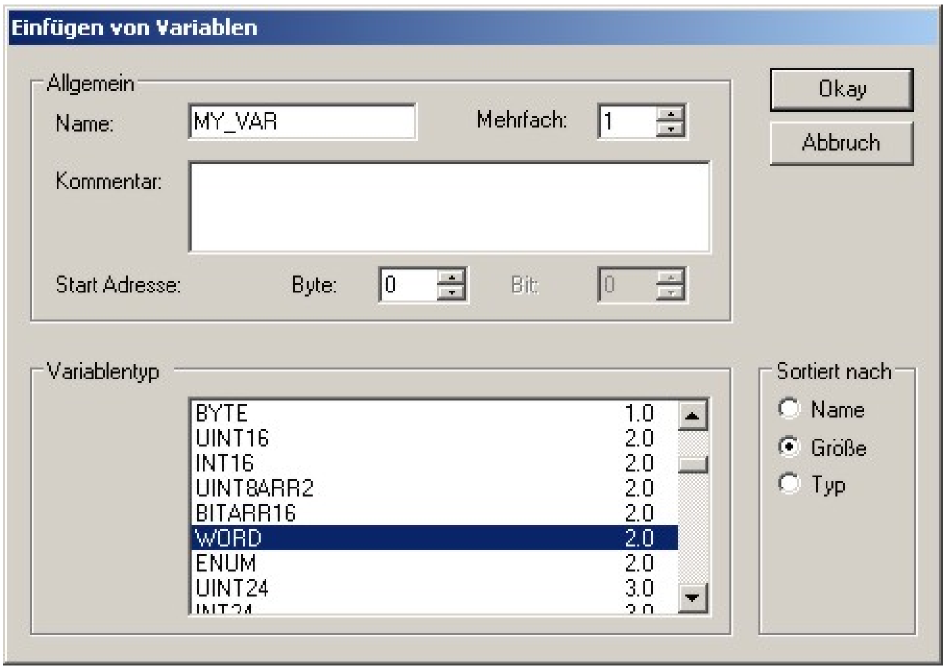


The hierarchical tree with the found devices is displayed after the bus scan.

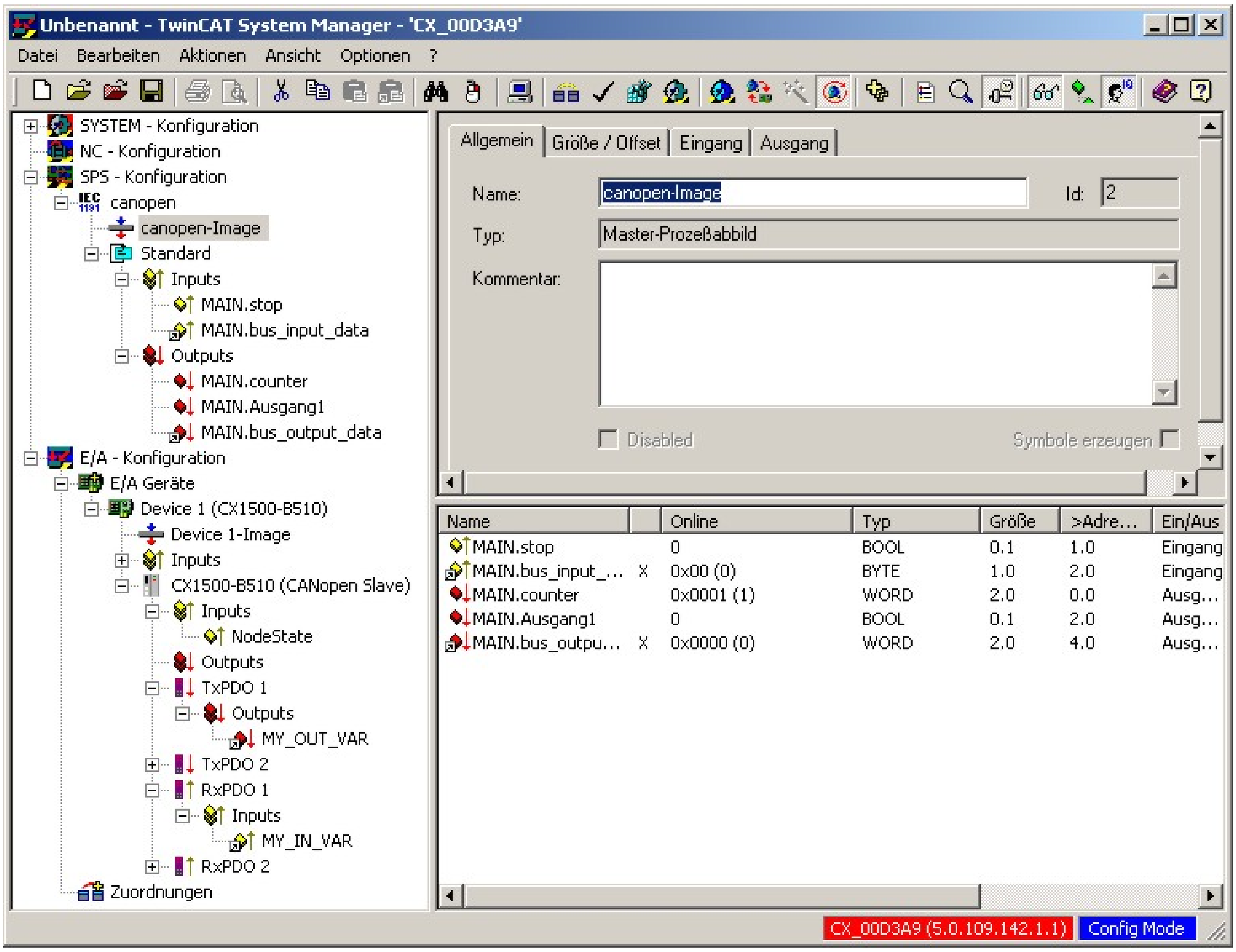


Below the box for the CANopen-Slave CX1500-B510 a status input shows the NodeState. Via this signal the PLC program can read the NodeState of the slave box. There are also four Process-Data-Objects (PDO) instanced: two for data output (TxPDO n) and two for data input (RxPDO n). Here the process variables can be inserted. To do so the user have to click on the input / output with the right mouse button. The following menu opens:

Select "Insert Variable". A dialog box opens to insert parameters for the variable.



The field "Name" defines the name for the variable. Start address can be used to define an address in the I/O area. In this way wanted overlays can be realized. The type of the variable will be set under "Variable Type". The sorting help make easier to find the desired variable. Click "OK" to accept settings.



If the process image of the PLC program is loaded the signals can be mapped to the variables (double click on the signal, see details in TwinCAT manual)

The mapped signals/variables are shown by a little arrow on the signal icon. If all signals and variables are mapped the configuration can be uploaded to the CX-System. The program will be loaded and started with the PCL control.

5 Error handling and diagnostics

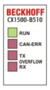
5.1 M510: LED diagnosis codes

Requirements

Display	LEDs		Meaning
	RUN	ERR	
	off	off	TwinCAT is in STOP mode
	on	off	all units connected to the bus are running without errors (Box State=0), TwinCAT Task or process is running.
	blinks at 2 Hz	off	Task connected to the Bus is not running. All units connected to the bus are up and running (Box State=0).
	an	blinks at 2 Hz	at least one Box State is not zero (e.g. unit not found, wrong configuration, unit in error state) TwinCAT Task is running.
	off	blinks at 2 Hz	at least one Box State is not zero (e.g. unit not found, wrong configuration, unit in error state), TwinCAT Task is not running.
	off	on	TwinCAT runs, CAN Controller is Bus OFF. Physical CAN bus problem. Probable errors : e.g. missing termination, wires are to long, wrong signal rate, node address is not definite, short circuit. Restart is necessary.
	blinks at 20 Hz	blinks at 20 Hz	configuration-upload is in progress
	off	blinks at 20 Hz	connector is in STOP mode

5.2 B510: LED diagnosis codes

Requirements

Display	LEDs		Meaning
	RUN	ERR	
	off	off	TwinCAT is in STOP mode
	on	off	all units connected to the bus are running without errors (Box State=0), TwinCAT Task or process is running.
	blinks at 2 Hz	off	Task connected to the Bus is not running. All units connected to the bus are up and running (Box State=0).
	an	blinks at 2 Hz	at least one Box State is not zero (e.g. unit not found, wrong configuration, unit in error state) TwinCAT Task is running.
	off	blinks at 2 Hz	at least one Box State is not zero (e.g. unit not found, wrong configuration, unit in error state), TwinCAT Task is not running.
	off	on	TwinCAT runs, CAN Controller is Bus OFF. Physical CAN bus problem. Probable errors : e.g. missing termination, wires are to long, wrong signal rate, node address is not definite, short circuit. Restart is necessary.
	blinks at 20 Hz	blinks at 20 Hz	configuration-upload is in progress
	off	blinks at 20 Hz	connector is in STOP mode

The TX / RX LEDs are reserved for further functions and has no meaning so far.

5.3 Diagnostics

CANopen state

In many cases it is important to know whether the communication with the higher-level master is still OK. To this end, link the "NodeState" variable with your PLC program. A TwinCAT configuration is necessary for this.

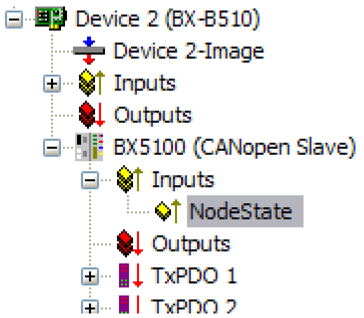


Fig. CANopen diagnostics byte in the System Manager

Requirements

Error number	Description	Remedy
0	No error	-
2	Guarding Error	Check your connection
20	Too less PDOs received (only TwinCAT Config.)	Check your configuration
22	Sync Error	Check your connection
129	Node is pre-operational	Start operational mode
130	Node is stopped	Start the node

Example

If CANopen is interrupted (e. g. because the cable is pulled of or the PLC is switched of), the BX5100 reports this with ??? in NodeState.

5.4 Trouble Shooting

Error Frames

One sign of errors in the CAN wiring, the address assignment or the setting of the baud rate is an increased number of error frames: the diagnostic LEDs then show *Warning Limit exceeded* or *Bus-off state entered*.

NOTE

Warning limit exceeded, passive error or bus-off state are indicated first of all at those nodes that have detected the most errors. These nodes are not necessarily the cause for the occurrence of error frames! If, for instance, one node contributes unusually heavily to the bus traffic (perhaps because it is the only one with analog inputs, the data for which triggers event-driven PDOs at a high rate), then the probability of its telegrams being damaged increases. Its error counter will, correspondingly, be the first to reach a critical level.

Node ID / Setting the Baud Rate

Care must be taken to ensure that node addresses are not assigned twice: there may only be one sender for each CAN data telegram.

Test 1

Check node addresses. If the CAN communication functions at least some of the time, and if all the devices support the boot up message, then the address assignment can also be examined by recording the boot up messages after the devices are switched on. This will not, however, recognize node addresses that have been swapped.

Test 2

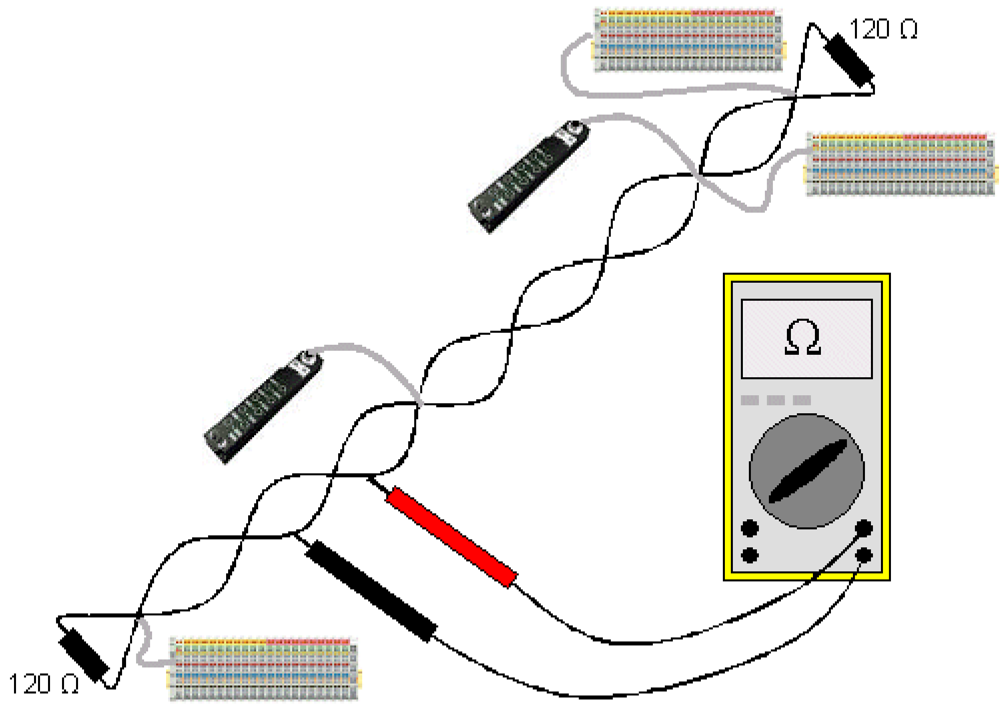
Check that the same baud rate has been set everywhere. For special devices, if the bit timing parameters are accessible, do they agree with the CANopen definitions (sampling time, SJW, oscillator).

Testing the CAN wiring

Do not carry out these tests when the network is active - communication should not take place during the tests. The following tests should be carried out in the stated sequence, because some of the tests assume that the previous test was successful. Not all the tests are generally necessary.

Network terminator and signal leads

The nodes should be switched off or the CAN cable unplugged for this test, because the results of the measurements can otherwise be distorted by the active CAN transceiver.



Test 3

Determine the resistance between CAN high and CAN low - at each device, if necessary.

If the measured value is greater than 65 Ohms, it indicates the absence of a terminating resistor or a break in a signal lead. If the measured value is less than 50 Ohms, look for a short circuit between the CAN lines, more than the correct number of terminating resistors, or faulty transceivers.

Test 4

Check for a short circuit between the CAN ground and the signal leads, or between the screen and signal leads.

Test 5

Remove the earth connection from the CAN ground and screen. Check for a short circuit between the CAN ground and screen.

Topology

The possible cable length in CAN networks depends heavily on the selected baud rate. CAN will tolerate short drop lines - although this again depends on the baud rate. The maximum permitted length of drop lines should not be exceeded. The length of cable that has been installed is often underestimated - estimates can even be a factor of 10 less than the actual length. The following test is therefore recommended:

Test 6

Measure the lengths of the drop lines and the total bus lengths (do not just make rough estimates!) and compare them with the topology rules for the relevant baud rate.

Screening and earthing

The power supply and the screen should be carefully earthed at the power supply unit, once only and with low resistance. At all connecting points, branches and so forth the screen of the CAN cable (and possibly the CAN GND) must also be connected, as well as the signal leads. In the Beckhoff IP20 Bus Couplers, the screen is grounded for high frequencies via an R/C element.

Test 7

Use a DC ammeter (16 amp max.) to measure the current between the power supply ground and the screen at the end of the network most remote from the power supply unit. An equalization current should be present. If there is no current, then either the screen is not connected all the way through, or the power supply unit is not properly earthed. If the power supply unit is somewhere in the middle of the network, the measurement should be performed at both ends. When appropriate, this test can also be carried out at the ends of the drop lines.

Test 8

Interrupt the screen at a number of locations and measure the connection current. If current is flowing, the screen is earthed at more than one place, creating a ground loop.

Potential differences

The screen must be connected all the way through for this test, and must not be carrying any current - this has previously been tested.

Test 9

Measure and record the voltage between the screen and the power supply ground at each node. The maximum potential difference between any two devices should be less than 5 volts.

Detect and localize faults

The "low-tech approach" usually works best: disconnect parts of the network, and observe when the fault disappears.

However, this does not work well for problems such as excessive potential differences, ground loops, EMC or signal distortion, since the reduction in the size of the network often solves the problem without the "missing" piece being the cause. The bus loading also changes as the network is reduced in size, which can mean that external interference "hits" CAN telegrams less often.

Diagnosis with an oscilloscope is not usually successful: even when they are in good condition, CAN signals can look really chaotic. It may be possible to trigger on error frames using a storage oscilloscope - this type of diagnosis, however, is only possible for expert technicians.

Protocol problems

In rare cases, protocol problems (such as faulty or incomplete CANopen implementation, unfavorable timing at boot up etc.) can be the cause of faults. In this case it is necessary to trace the bus traffic for evaluation by a CANopen experts - the Beckhoff support team can help here.

A free channel on a Beckhoff FC5102 CANopen PCI card is appropriate for such a trace - Beckhoff make the necessary trace software available on the internet. Alternatively, it is of course possible to use a normal commercial CAN analysis tool.

Protocol problems can be avoided if devices that have not been conformance tested are not used. The official CANopen Conformance Test (and the appropriate certificate) can be obtained from the CAN in Automation Association (http://www.can-cia.de).

6 Decomissioning

6.1 Removal and disposal

A CX10x0 hardware configuration is dismantled in 2 stages:

0. Switching off and disconnecting the power supply

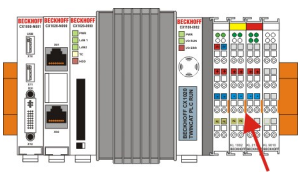
Before a CX10x0 system can be dismantled, the system should be switched off, and the power supply should be disconnected.

1. Removing from the top-hat rail:

Before the individual CX10x0 modules are disconnected, the whole CX1020 hardware block should be removed from the top-hat rail. Proceed as follows:

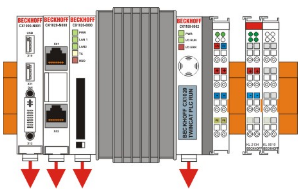
1.1. Release and remove the first Terminal next to the power supply unit on the top-hat rail.

First remove any wiring from power supply unit *and* then from the first terminal on the top-hat rail next to the power supply unit. If the wiring is to be reused for another system, it is advisable to make a note of the connections. Then pull the orange terminal release (see arrow) to release the terminal and pull it out.



1.2. Releasing the CX10x0 system

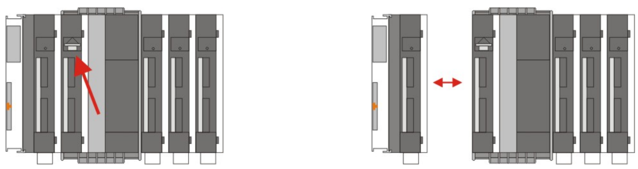
In order to release the CX10x0 block, pull the white straps at the bottom of the module in the direction of the arrows. They will lock in the extended position. After pulling the terminal release of the power supply unit, the block can be removed *carefully* from the top-hat rail.



2. Separating the individual modules

2.1. Separating the power supply unit, the CX10x0 CPU and other components

Place the CX10x0 block onto a suitable support with the front facing down. Then insert a flat screwdriver with dimensions 1.0 x 5.5 x 150 mm into the locking mechanism, and then operating the slider by turning it about 90 degrees. The locking mechanism on the rear affects an approx. 2-3 mm wide clearance of the module latching mechanism, pushing them apart. The plug connectors of the PC 104 interface can then be pulled apart carefully.



Only modules (CPU, fieldbus connections and UPS modules) that can be separated non-destructively feature a release device. Modules that cannot be separated only feature a marking point (with or without red paint seal). Applying force to these elements will destroy them.

NOTE
Forcibly opening the module housing (e.g. removing the cover) will destroy the housing.

Disposal

The device must be fully dismantled in order to dispose of it.

Electronic parts must be disposed of in accordance with national electronics scrap regulations.

7 Appendix

7.1 Mechanical assembly of the basic module

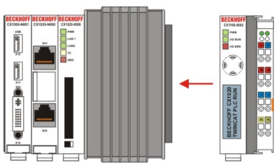
The installation of the modules takes place in three steps:

1. The sequence of the modules

The basic CPU module with system interfaces, which are factory-installed on the left side, is extended with the power supply unit on the right and with the fieldbus connection (master or slave) left side if available.

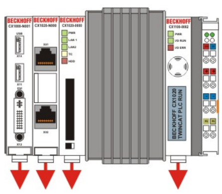
2. Assembly of the CPU and the power supply unit

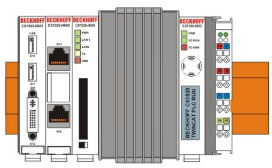
The individual modules are simply plugged together. The PC104 connector plugs should be handled carefully in order to avoid damage. When correctly assembled, no significant gap can be seen between the attached housings.



3. Engaging on the top-hat rail

On the bottom of the modules, there is a white tension strap, which is connected with a latching mechanism. These tension straps must be pulled down before attaching to the top-hat rail. This can be done using an ordinary screwdriver and a slight turn.





Then fix the CX1020 block on the top hat-rail using the latching straps. You should hear a soft click.

NOTE
Do not force the module or apply excessive pressure!

Only apply pressure at insensitive points of the housing (edges). Never apply pressure on the display, the buttons or movable parts of the CX10x0 system.

After successful latching on the top-hat rail the straps should be pushed back to their original position.

Note:

A locking mechanism prevents the individual housings from being pulled off again. Detailed information relating to disassembly of the CX1020 configuration from the top-hat rail can be found on page "Removal and disposal [▶ 176]".

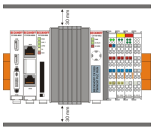
Installation position:

NOTE

The maximum ambient temperature for CPU modules mounted on a top-hat rail is 50°C. The orientation in which the device is fitted must be selected in such a way that cooling air can flow vertically through the ventilation holes. The images show the correct (Fig. 8) and two incorrect installation positions (Figs. 9 & 10).
Observe minimum clearance! Mounting must provide a clearance of 30 mm both above and below a CX1000 device combination to ensure adequate ventilation of the base CPU module and the power supply unit.

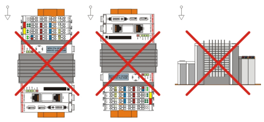
Correct installation position:

The high-performance CX1020 system generates a significant amount of heat, which is dissipated via a passive ventilation system. This system requires the unit to be mounted correctly. Ventilation openings are located at the top and bottom of the housing. The system therefore has to be installed horizontally. This ensures optimum air flow.



Incorrect installation positions:

The CX1020 system must not be operated vertically on the top-hat rail. A vertical position would lead to insufficient CPU ventilation, since the ventilation openings are located on the top and bottom of the housing. Installation of the system on its side would also lead to inadequate ventilation.



7.2 Certifications

All products of the Embedded PC family are CE, UL and GOST-R certified. Since the product family is continuously developed further, we are unable to provide a full listing here. The current list of certified products can be found on the Embedded PC certificates web page or at www.beckhoff.com under Embedded PC.

7.3 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on her internet pages:
http://www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

Phone:	+49(0)5246/963-0
Fax:	+49(0)5246/963-198
e-mail:	info@beckhoff.com

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline:	+49(0)5246/963-157
Fax:	+49(0)5246/963-9157
e-mail:	support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline:	+49(0)5246/963-460
Fax:	+49(0)5246/963-479
e-mail:	service@beckhoff.com

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com