37389

DTSC-200 Series Interfaces

Interface Description

Software Version 1.0xxx

Manual 37389

WARNING

Read this entire manual and all other publications pertaining to the work to be performed before installing, operating, or servicing this equipment. Practice all plant and safety instructions and precautions. Failure to follow instructions can cause personal injury and/or property damage.

The engine, turbine, or other type of prime mover should be equipped with an overspeed (overtemperature, or overpressure, where applicable) shutdown device(s), that operates totally independently of the prime mover control device(s) to protect against runaway or damage to the engine, turbine, or other type of prime mover with possible personal injury or loss of life should the mechanical-hydraulic governor(s) or electric control(s), the actuator(s), fuel control(s), the driving mechanism(s), the linkage(s), or the controlled device(s) fail.

Any unauthorized modifications to or use of this equipment outside its specified mechanical, electrical, or other operating limits may cause personal injury and/or property damage, including damage to the equipment. Any such unauthorized modifications: (i) constitute "misuse" and/or "negligence" within the meaning of the product warranty thereby excluding warranty coverage for any resulting damage, and (ii) invalidate product certifications or listings.

CAUTION

To prevent damage to a control system that uses an alternator or battery-charging device, make sure the charging device is turned off before disconnecting the battery from the system.

Electronic controls contain static-sensitive parts. Observe the following precautions to prevent damage to these parts.

- **Discharge body static before handling the control (with power to the control turned off, contact a grounded surface and maintain contact while handling the control).**
- **Avoid all plastic, vinyl, and Styrofoam (except antistatic versions) around printed circuit boards.**
- **Do not touch the components or conductors on a printed circuit board with your hands or with conductive devices.**

OUT-OF-DATE PUBLICATION

This publication may have been revised or updated since this copy was produced. To verify that you have the latest revision, be sure to check the Woodward website:

http://www.woodward.com/pubs/current.pdf

The revision level is shown at the bottom of the front cover after the publication number. The latest version of most publications is available at:

http://www.woodward.com/publications

If your publication is not there, please contact your customer service representative to get the latest copy.

Important definitions

WARNING

Indicates a potentially hazardous situation that, if not avoided, could result in death or serious injury.

CAUTION

Indicates a potentially hazardous situation that, if not avoided, could result in damage to equipment.

NOTE

Provides other helpful information that does not fall under the warning or caution categories.

Revision History

Rev.	Date	Editor	Changes
NEW	07-12-12	TP	Release

Contents

CHAPTER 1. GENERAL INFORMATION 6
Related Documents 6
Interface Overview 7
Modbus Half/Full Duplex Application 8
CAN Bus 9

CHAPTER 2. DATA TELEGRAMS 10
Interface Monitoring 10
Transmit Telegram 10
Modbus 10
CAN (CAL) 10
CANopen 10
Receive Telegram 11
Modbus 11
CAN (CAL) 12
CANopen 12

CHAPTER 3. SERIAL INTERFACE 13
Overview 13
Modbus RTU Slave 14
General Information 14
Configuration 14
Modbus Addressing and Data Model 15
Visualization 16
Configuration 17
Exception Responses 20

CHAPTER 4. CAN (CAL) 21

CHAPTER 5. CANOPEN 22
Introduction 22
Server Data Objects (SDO) - Communication 23
Process Data Objects (PDO) 25
Setting the Transmit PDO (Examples) 26
SYNC Message 27
Using a CANopen Configuration Program 27
Settings for Connection with External Devices 28
Expansion with One IKD 1 (8 Additional External DI/DO) 29
Expansion with Two IKD 1 (16 Additional External DI/DO) 30
Expansion with the Phoenix terminal IL CAN BK / ILB CO 24 16DI 16DO (16 DI/DO) 32
Description of the DTSC Parameters 34
Interfaces: General 34
General CANopen Parameters 35
CANopen Receive PDO (RPDO) {x} ({x} = 1/2) 37
Combine Functions with Each Other 37
CANopen Transmit PDO (TPDO) {x} ({x} = 1 to 4) 38
FAQ CAN Bus 40
Recommendations of Woodward 40
Device Combinations and Bus Load 40

APPENDIX A. TELEGRAMS 43
Transmission Telegram 43
Data Protocol 4700 43
Data Protocol 4800 (Source 1 Data) 52
Data Protocol 4801 (Source 2 Data) 54
Remote Control Telegram 57

APPENDIX B. CANOPEN 58
Description of the Common Data Types 58
Structure of the PDO-COB-ID Entry (UNSIGNED32) 58
Description of the Object Parameter 59
Data Format of Different Functions 66
Receiving Messages 66
Definition of Protocol Descriptions 67
Unsigned Integer 67
Signed Integer 68
Transmission Telegram 69
CANopen: Mapping Parameter 71

APPENDIX C. APPLICATION EXAMPLES 76
Remote Control 76
Configuration of the *LogicsManager* Functions 76
Remote Control Telegram 77
Remote Control via CAN 77
Remote Acknowledgement 77
Remote Control via Modbus 80
Sending a Data Protocol via TPDO 82
Cyclically Sending of Data 82
Sending of Data on Request 82

Illustrations And Tables

Illustrations

Figure 1-1: Interface overview 7
Figure 1-2: Interface overview - serial interface Modbus full-duplex 8
Figure 1-3: Interface overview - serial interface Modbus half-duplex 8
Figure 1-4: CAN bus topology 9
Figure 1-5: Interface - The CAN bus loop 9
Figure 2-1: Data telegrams - remote control via CAN 12
Figure3-1: Serial interface - overview 13
Figure 3-2: Modbus - visualization configurations 16
Figure 3-3: Modbus - configuration example 1 18
Figure 3-4: Modbus - configuration example 2 18
Figure 3-5: Modbus - configuration example 3 19
Figure 4-1: CAN (CAL) interface - overview 21
Figure 5-1: CANopen interface - overview 24
Figure 5-2: CANopen interface - CANopen configuration software 27
Figure 5-3: CANopen interface - external devices 28
Figure 5-4: CANopen Schnittstelle - Einstellungen für externe Geräte 29
Figure 5-5: CANopen interface - expansion with two IKD 1 30
Figure 5-6: CANopen interface - expansion with Phoenix terminal 32
Figure 5-7: Display screen - Ext. acknowledge 76
Figure 5-8: Display screen - configure CAN interface 77
Figure 5-9: CANopen request data for Node ID 1 78
Figure 5-10: Display screen - configure device number 78
Figure 5-11: CANopen request data for Node ID 2 78
Figure 5-12: Display screen - configure Server SDOs 79
Figure 5-13: CANopen request data for additional Server SDO 80
Figure 5-14: Modbus - remote control parameter 503 81
Figure 5-15: Modbus - write register 81
Figure 5-16: Cyclical sending of data - Sync Message request 82
Figure 5-17: Cyclical sending of data - reply 82

Tables

Table 1-1: Manual - overview 6
Table 3-1: Modbus - address range block read 16
Table 3-2: Modbus - address calculation 17
Table 3-3: Modbus - data types 17
Table 3-4: Modbus - exception responses 20

Chapter 1.
General Information

Related Documents

Type		English	German
DTSC-200 Series			
DTSC-200 - Installation		37385	-
DTSC-200 - Configuration		37386	-
DTSC-200 - Operation		37387	-
DTSC-200 - Application		37388	-
DTSC-200 - Interfaces	this manual ⇨	37389	-
Additional Manuals			
IKD 1 - Manual		37135	GR37135
Discrete expansion board with 8 discrete inputs and 8 relay outputs that can be coupled via the CAN bus to the control unit. Evaluation of the discrete inputs as well as control of the relay outputs is done via the control unit.			
LeoPC1 - User Manual		37146	GR37146
PC program for visualization, configuration, remote control, data logging, language upload, alarm and user management, and management of the event recorder. This manual describes the set up of the program and interfacing with the control unit.			
LeoPC1 - Engineering Manual		37164	GR37164
PC program for visualization, configuration, remote control, data logging, language upload, alarm and user management, and management of the event recorder. This manual describes the configuration and customization of the program.			

Table 1-1: Manual - overview

Intended Use The unit must only be operated in the manner described by this manual. The prerequisite for a proper and safe operation of the product is correct transportation, storage, and installation as well as careful operation and maintenance.

NOTE

This manual has been developed for a unit fitted with all available options. Inputs/outputs, functions, configuration screens, and other details described, which do not exist on your unit, may be ignored.

The present manual has been prepared to enable the installation and commissioning of the unit. Due to the large variety of parameter settings, it is not possible to cover every combination. The manual is therefore only a guide. In case of incorrect entries or a total loss of functions, the default settings may be taken from the list of parameters enclosed in the configuration manual 37386.

Interface Overview

The DTSC-200 provides the following communication interfaces:

- **Serial interface 1 (DPC)**
  LeoPC1
- **Serial interface 2 (RS-485)**
  Modbus
- **CAN interface**
  CANopen or CAN CAL (dependent on application)

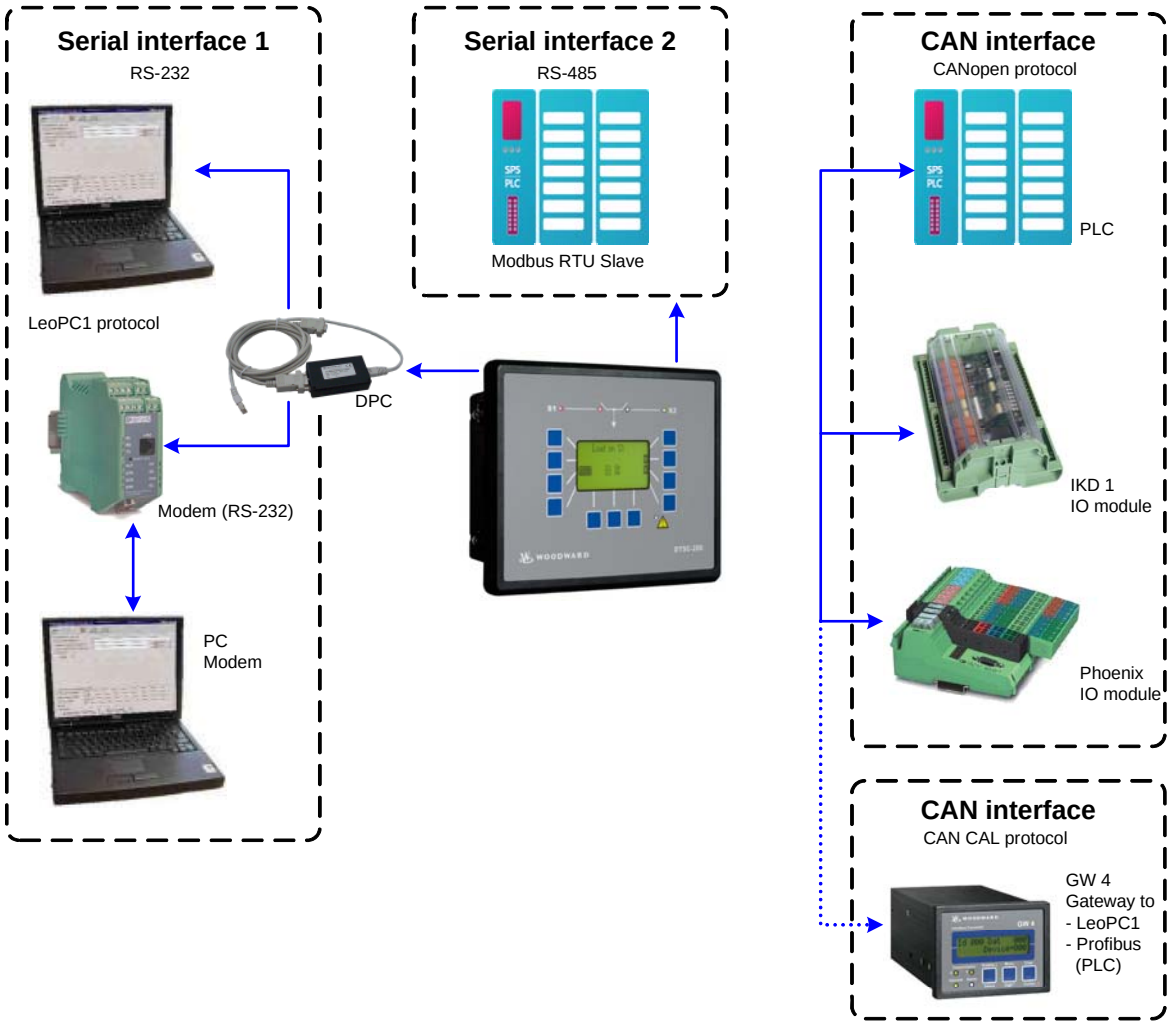


Figure 1-1: Interface overview

WARNING

When connecting the direct configuration interface, the Woodward DPC with RJ45 connector must be used. Failure to do so may destroy the unit.

Modbus Half/Full Duplex Application

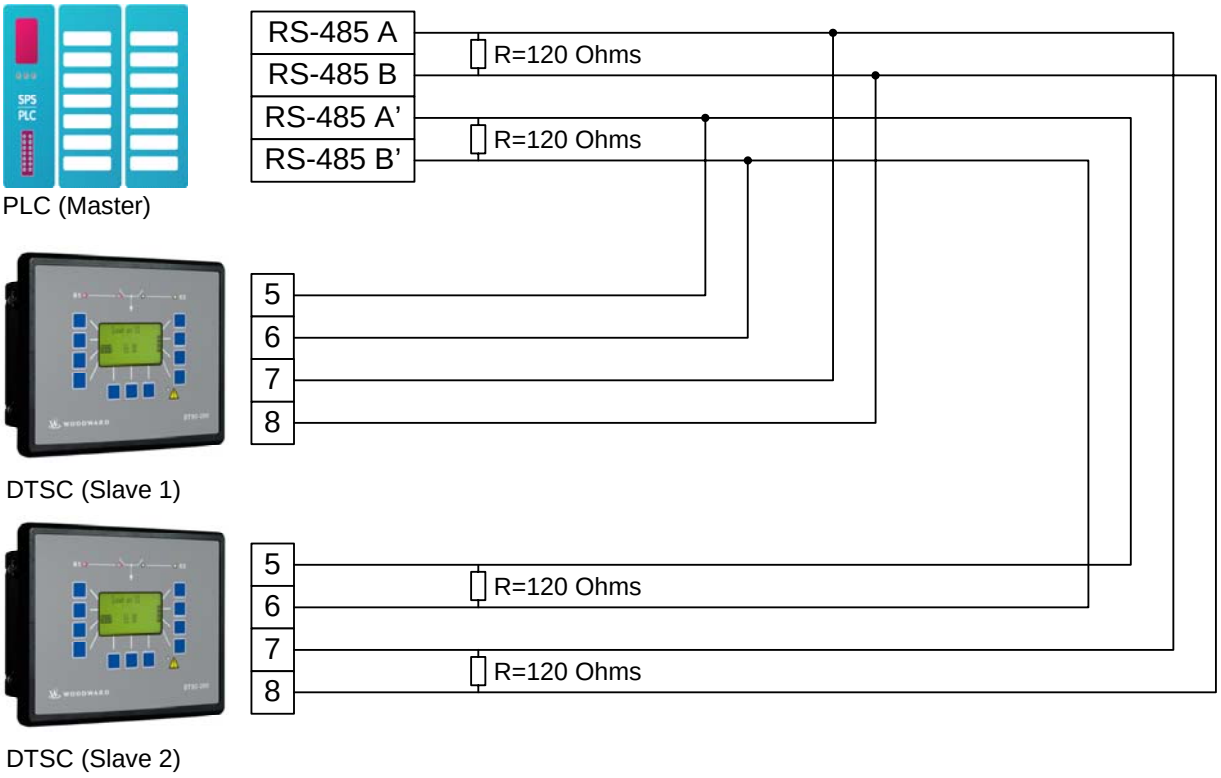


Figure 1-2: Interface overview - serial interface Modbus full-duplex

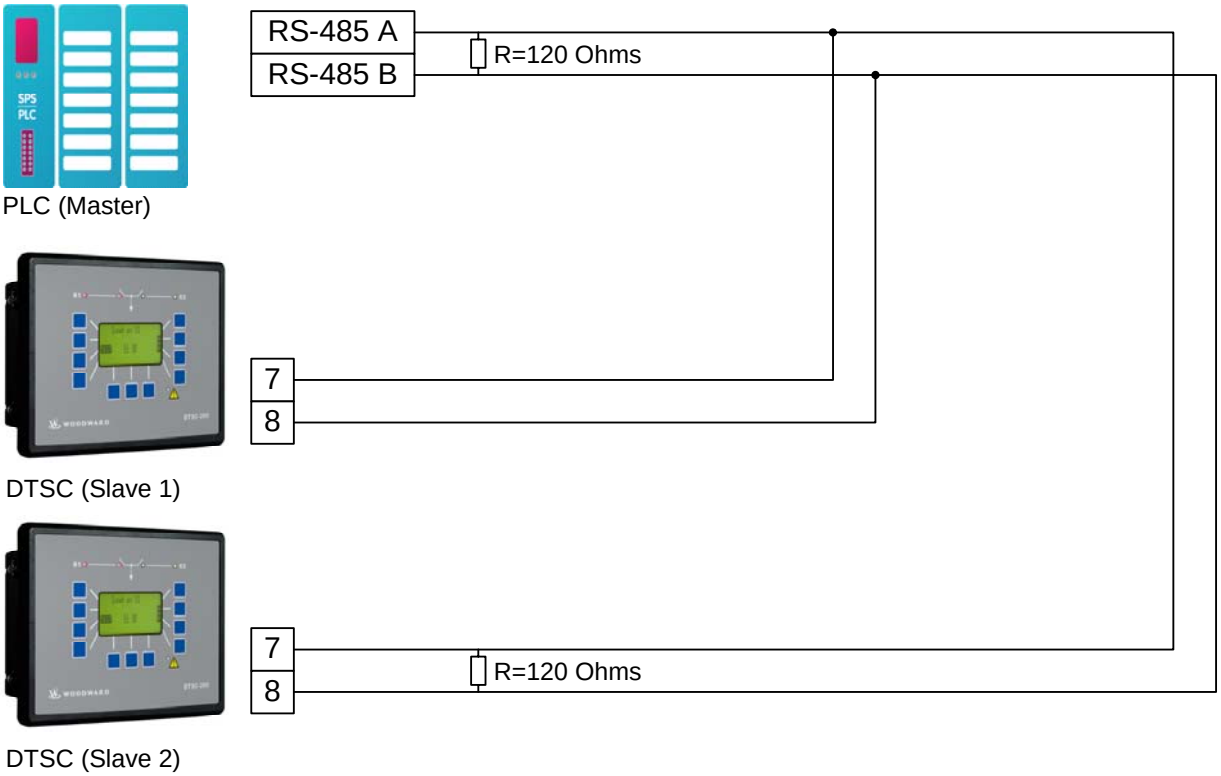


Figure 1-3: Interface overview - serial interface Modbus half-duplex

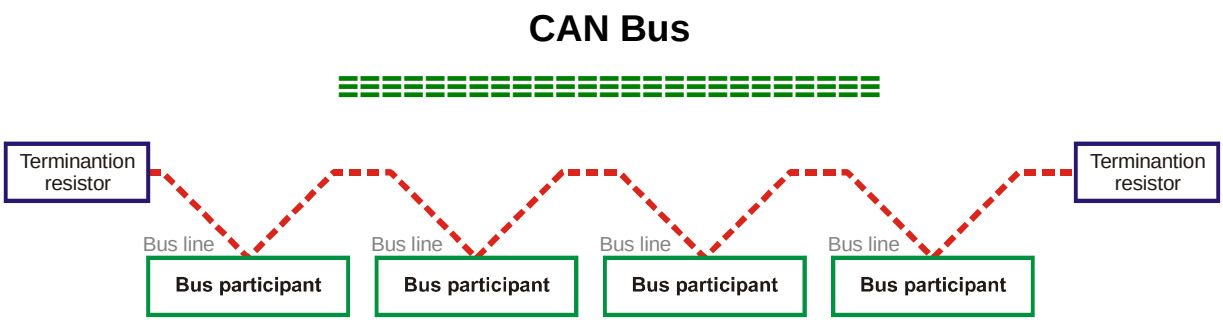


Figure 1-4: CAN bus topology

Characteristics of the CAN interface used by Woodward:

- Standard: Compatible with ISO 11898
- Electrically isolated: Isolation voltage 1,500 V_{DC}

NOTE

Please note that the CAN bus must be terminated with an impedance which corresponds to the wave impedance of the cable (e.g. 120 Ohm, ¼ W). The CAN bus is terminated between CAN-H and CAN-L.

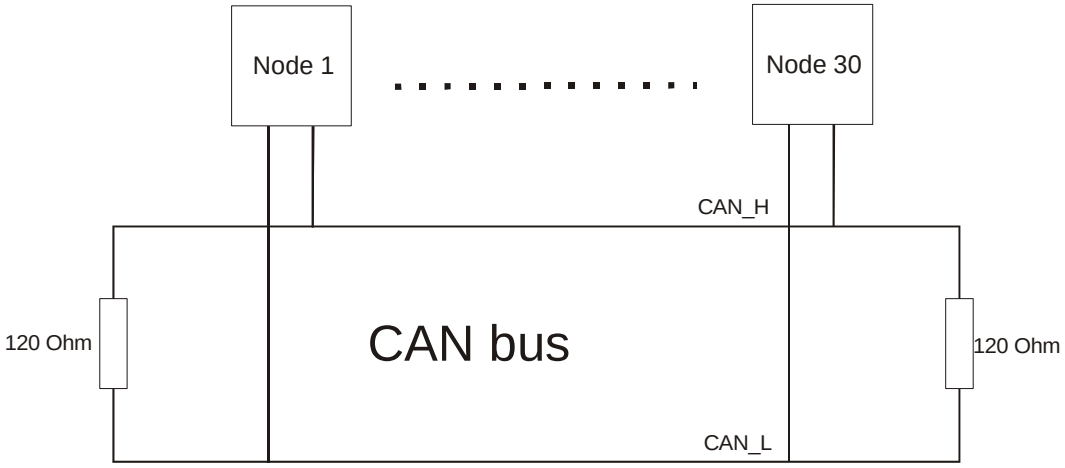


Figure 1-5: Interface - The CAN bus loop

Chapter 2.
Data Telegrams

Interface Monitoring

It is possible to monitor the CAN interface for received data of an external I/O board. Refer to the configuration manual for more information about this monitoring function.

Transmit Telegram

The transmit telegram provides all measuring and status data of the DTSC. The data have different addresses and will be transmitted in the respective format depending on the selected interface.

Modbus

Data transmission in Modbus format is performed in the order of the transmit telegram (refer to Appendix A: Transmission Telegram on page 43). The data addresses may be taken from the respective column of the transmit telegram.

CAN (CAL)

The DTSC sends its data via cyclic CAN messages. If a GW 4 is used, the baud rate must be configured to 125 kBaud.

NOTE

Instead of using a GW 4, a CAN to USB (or RS-232) converter may be used.

CANopen

Using the mapped objects, which are described in detail starting on page 25, enables you to send data by setting the object ID 2C76h on the basis of the CANopen protocol.
This document contains tables of further mapped objects, which may be configured. Refer to Appendix A: Transmission Telegram on page 43.

NOTE

When using the mapped objects listed in the appendix instead of the complete transmit telegram, the refresh rate of the messages may be reduced.

Receive Telegram

The receive telegram enables to acknowledge alarm messages, which are no longer active, via remote control

In order to execute the desired command, a rise of the pulse of the respective signal from Low to High is required.

An acknowledgement command must be sent twice. The first rise of the pulse resets the horn. The second rise of the pulse acknowledges the unit, if the fault is not present anymore.

NOTE

Please note that the respective remote control parameters must be configured in the *LogicsManager* of the unit. Refer to the application manual 37388 for more detailed information about this.

Modbus

It is possible to remote control the DTSC using the bits 2 to 4 of control word 1 on address 503. The Remote Control Telegram in Appendix A on page 57 is valid for both, CANopen as well as Modbus, and indicates the arrangement of the remote control bits.

CAN (CAL)

The Woodward LeoPC1 software may be used to remote control the DTSC via a connected PC. After selecting the desired remote control command, the remote control command must be confirmed by selecting the "Set" button.

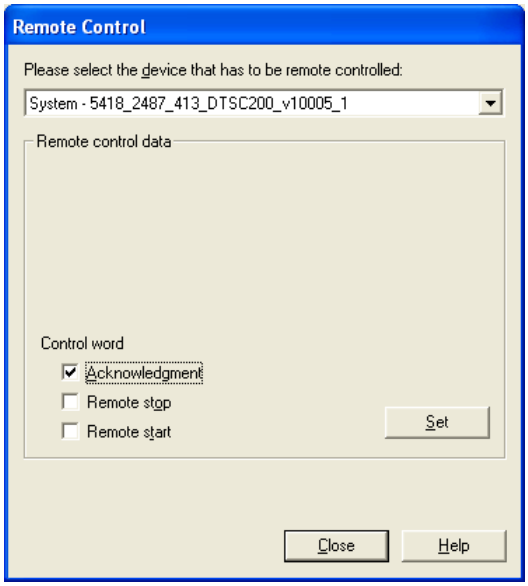


Figure 2-1: Data telegrams - remote control via CAN

NOTE

The control words "Remote stop" and "Remote start" have no effect on the DTSC-200.

CANopen

It is possible to remote control the DTSC using the bits 2 to 4 of control word 1 on address 503. The Remote Control Telegram in Appendix A on page 57 is valid for both, CANopen as well as Modbus, and indicates the arrangement of the remote control bits.

Chapter 3.
Serial Interface

Overview

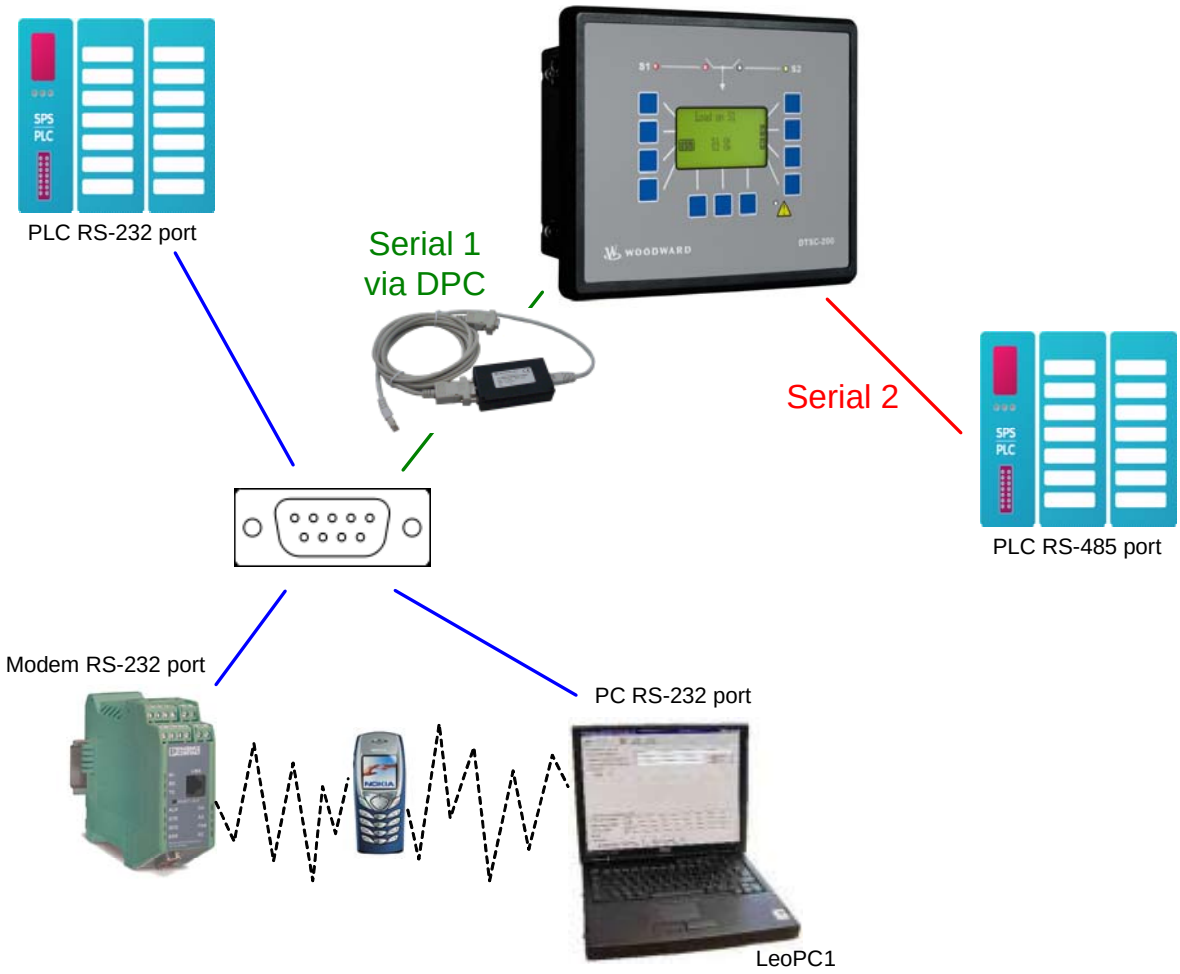


Figure3-1: Serial interface - overview

Modbus RTU Slave

General Information

Modbus is a serial communications protocol published by Modicon in 1979 for use with its programmable logic controllers (PLCs). It has become a de facto standard communications protocol in industry, and is now the most commonly available means of connecting industrial electronic devices. The DTSC supports a Modbus RTU Slave module. This means that a Master node needs to poll the DTSC slave node. Modbus RTU can also be multi-dropped, or in other words, multiple Slave devices can exist on one Modbus RTU network, assuming that the serial interface is a RS-485. Detailed Information about the Modbus protocol are available on the following website:
http://www.modbus.org/specs.php

There are also various tools available on the internet. We recommend to use ModScan32 which is a Windows application designed to operate as a Modbus Master device for accessing data points in a connected Modbus Slave device. It is designed primarily as a testing device for verification of correct protocol operation in new or existing systems. It is possible to download a trial version from the following website:
http://www.win-tech.com/html/modscan32.htm

Configuration

EN Baudrate
DE Baudrate
CL2
3170

Serial interface 2: Baud rate **2.4 / 4.8 / 9.6 / 14.4 / 19.2 / 38.4 / 56 / 115 kBaud**

This parameter defines the baud rate for communications. Please note, that all participants on the service interface must use the same baud rate.

EN Parity
DE Parity
CL2
3171

Serial interface 2: Parity **no / even / odd**

The used parity of the service interface is set here.

EN Stop bits
DE Stop Bits
CL2
3172

Serial interface 2: Stop bits **one / two**

The number of stop bits is set here.

EN Full-, halfduplex mode
DE Voll-, Halbduplex Modus
CL2
3173

Serial interface 2: Full-/halfduplex mode **Fullduplex / Halfduplex**

Fullduplex ... Fullduplex mode is enabled.
Halfduplex... Halfduplex mode is enabled.

EN ModBus Slave ID
DE ModBus Slave ID
CL2
3185

Serial interface: Modbus Slave ID **0 to 255**

The Modbus device address is entered here, which is used to identify the device via Modbus. If 0 is entered here, the Modbus Slave module is disabled.

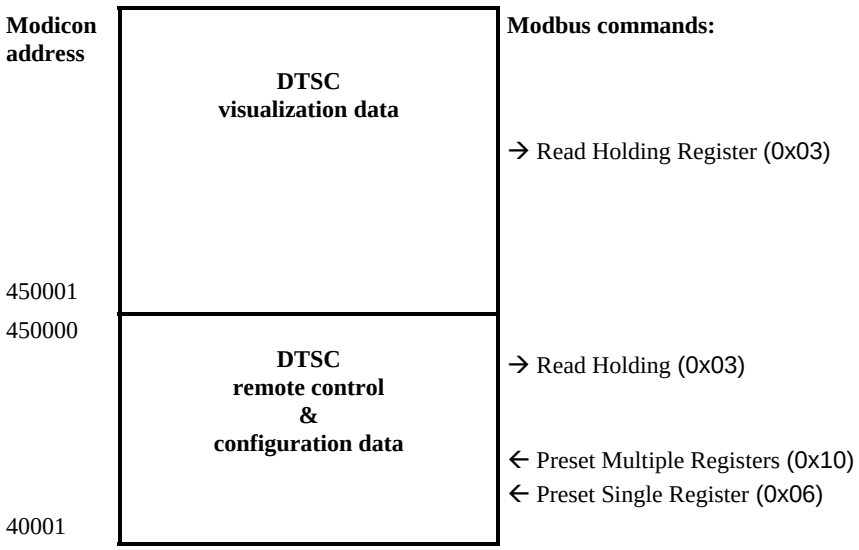
EN Modbus Reply delay time
DE Modbus Zeitverzöger. der Antwort
CL2
3186

Serial interface: Reply delay time **0,00 to 1,00 s**

This is the minimum delay time between a request from the Modbus master and the sent response of the slave. This time is also required if an external interface converter to RS-485 is used for example. Please note that you also need the DPC in this case.

Modbus Addressing and Data Model

The DTSC Modbus slave module distinguishes between visualization data and configuration & remote control data. The different data is accessible over a split address range and may be read via the "Read Holding Register" function. Furthermore, DTSC parameters and remote control data can be written with the "Preset Single Registers" function or "Preset Multiple Registers" (refer to figure below).



NOTE

All addresses in this document comply with the Modicon address convention. Some PLCs or PC programs use different address conventions depending on their implementation. Then the address must be increased and the leading 4 may be omitted.

Please refer to your PLC or program manual for more information. This determines the address sent over the bus in the Modbus telegram. The Modbus starting address 450001 of the visualization data may become bus address 50000 for example.

Visualization

The visualization over Modbus is provided in a very fast data protocol where important system data like alarm states, AC measurement data, switch states and various other information may be polled. According to the DTSC Modbus addressing range, the visualization protocol can be reached on addresses starting at 450001. On this address range it is possible to do block reads from 1 up to 128 Modbus registers at a time.

Modbus Read Addresses	Description	Multiplier	Units
450001	Protocol-ID		--
450002	Source 2: Voltage V_{L12}	0.1	V
.....	.....	.....	.....
.....	.....	.....	.....
.....	.....	.....	.....
.....	.....	.....	.....
450088	Timer state feedback signals	-	-

Table 3-1: Modbus - address range block read

NOTE

Table 3-1 is only an excerpt of the data protocol. It conforms to the data protocol, that is also used by CAN bus. Refer to Appendix A: Transmission Telegram on page 43 for the complete protocol.

The following exemplary ModScan32 screenshot shows the configurations made to read the visualization protocol with a block read of 128 registers.

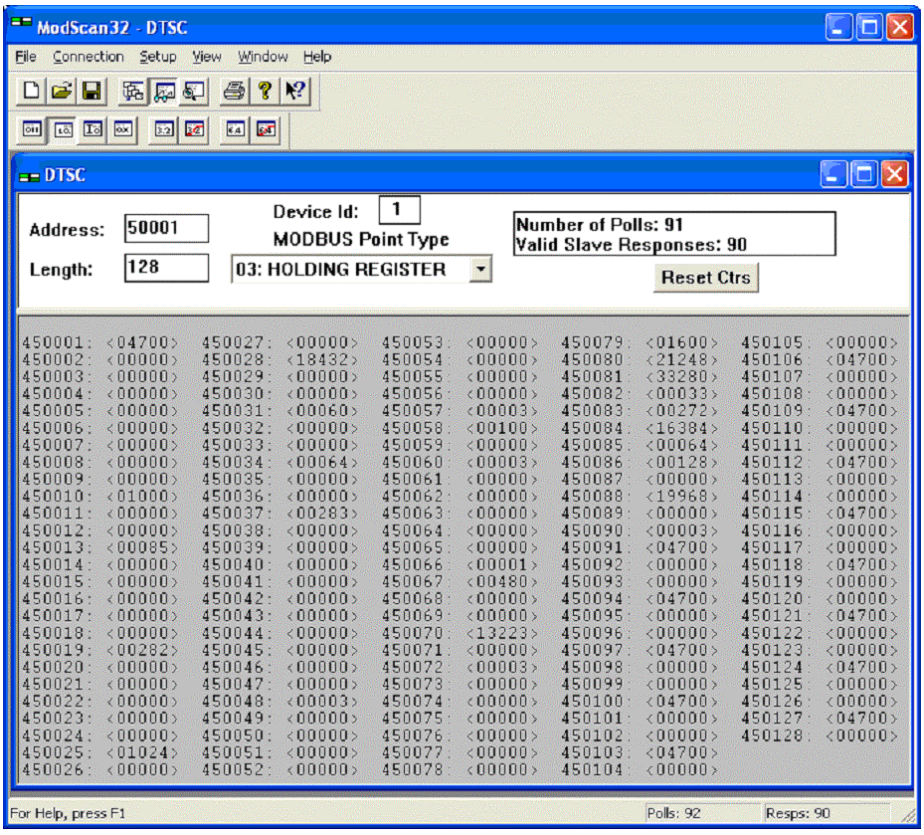


Figure 3-2: Modbus - visualization configurations

Configuration

The Modbus interface can be used to read/write parameters of the DTSC. According to the DTSC Modbus addressing range for the configuration addresses, the range starts at 40001 and ends at 450000. You can always access only one parameter of the system in this address range. The Modbus address can be calculated depending on the parameter ID as illustrated below:

	Parameter ID < 10000	Parameter ID >= 10000
Modbus address =	40000 + (Par. ID+1)	400000 + (Par. ID+1)

Table 3-2: Modbus - address calculation

Block reads in this address range depend on the data type of the parameter. This makes it important to set the correct length in Modbus registers which depends on the data type (UNSIGNED 8, INTEGER 16, etc.). Refer to Table 3-3 for more information.

DTSC types	Modbus registers
UNSIGNED 8	1
UNSIGNED 16	1
INTEGER 16	1
UNSIGNED 32	2
INTEGER 32	2
LOGMAN	7
TEXT/X	X/2

Table 3-3: Modbus - data types

NOTE

The parameters of the following examples are an excerpt of the parameter list in the appendix of the Configuration Manual 37386. Please refer to this manual for the complete parameter list.

NOTE

Be sure to enter the password for code level 2 or higher for the corresponding interface to get access for changing parameter settings.

NOTE

The new entered value must comply with the parameter setting range when changing the parameter setting.

Example 1: Addressing the password for the CAN interface:

Par. ID.	Parameter	Setting range	Data type
10402	Password for CAN interface1	0000 to 9999	UNSIGNED 16

Modbus address = 400000 + (Par. ID +1) = 410403
Modbus length = 1 (UNSIGNED 16)

The following Modscan32 screenshot shows the configurations made to address parameter 10402.

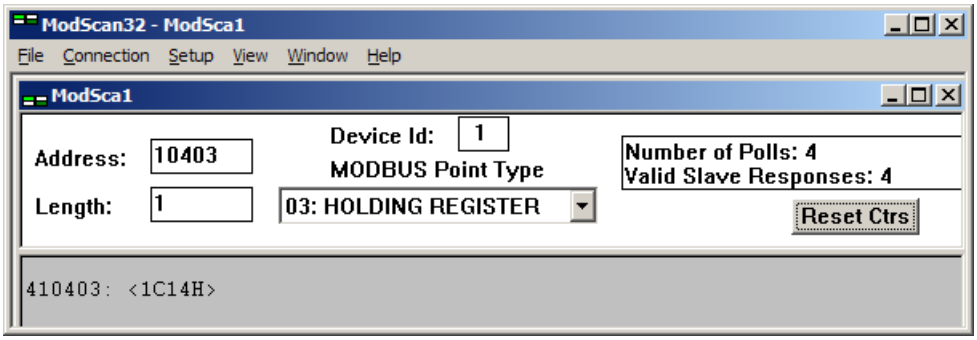


Figure 3-3: Modbus - configuration example 1

Example 2: Addressing the rated voltage of source 1:

Par. ID.	Parameter	Setting range	Data type
1774	Rated voltage S1	50 to 650000 V	UNSIGNED 32

Modbus address = 40000 + (Par. ID +1) = 41775
Modbus length = 2 (UNSIGNED 32)

The following Modscan32 screenshot shows the configurations made to address parameter 1774.

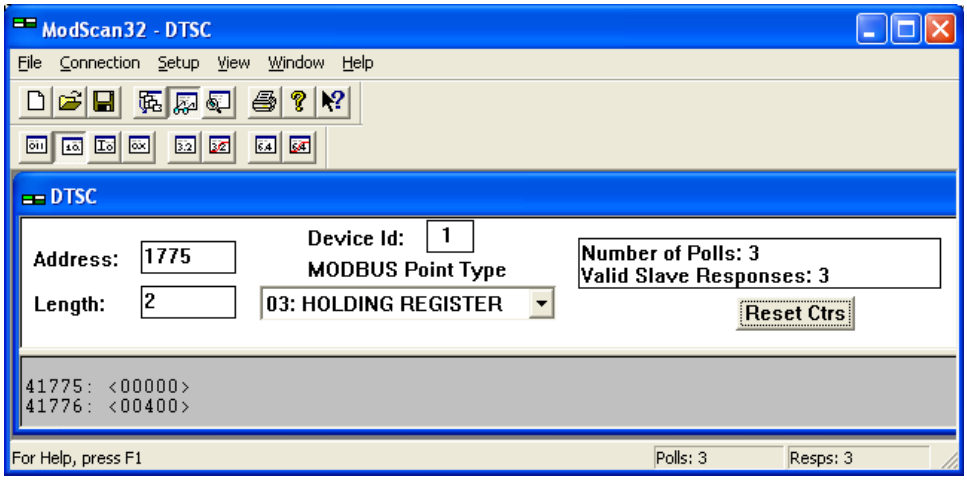


Figure 3-4: Modbus - configuration example 2

Example 3: Addressing source 2 voltage measuring:

Par. ID.	Parameter	Setting range	Data type
1861	S2 voltage measuring	3Ph 4W {0} 3Ph 3W {1} 1Ph 2W {2} 1Ph 3W {3}	UNSIGNED 16

Modbus address = 40000 + (Par. ID +1) = 41862
Modbus length = 1 (UNSIGNED 16)

NOTE

If the setting range contains a list of parameter settings like in this example, the parameter settings are numbered and start with 0 for the first parameter setting. The number corresponding with the respective parameter setting must be configured.

The following Modscan32 screenshot shows the configurations made to address parameter 1861, which is configured to "3Ph 4W".

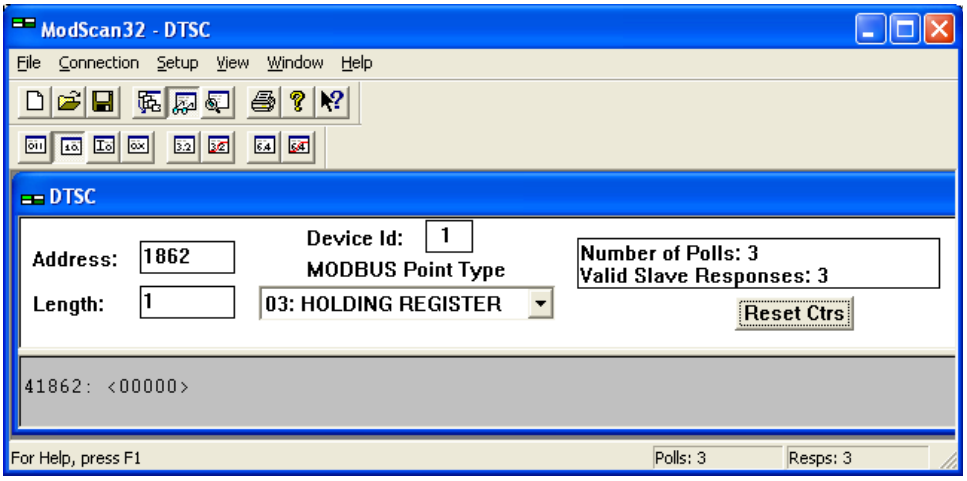


Figure 3-5: Modbus - configuration example 3

Exception Responses

The DTSC Modbus interface has multiple exception responses to show that a request could not be executed. Exception responses can be recognized if the response telegram contains the request function code with an offset of 128 (0x80 hex).

Table 3-4 explains possible reasons for an exception response that occurred.

DTSC Modbus Exception Responses		
Code	**Name**	**Reason**
01	ILLEGAL FUNCTION	• The sent request function code is not supported by the DTSC Modbus interface.
02	ILLEGAL ADDRESS	• Permission to read/write the parameter is denied. • The amount of requested registers is wrong to read/write this registers.
03	ILLEGAL DATA VALUE	• The data value exceeds the min. and max. limitations of the parameter upon a write request. • There is no parameter on the requested address.

Table 3-4: Modbus - exception responses

Chapter 4.
CAN (CAL)

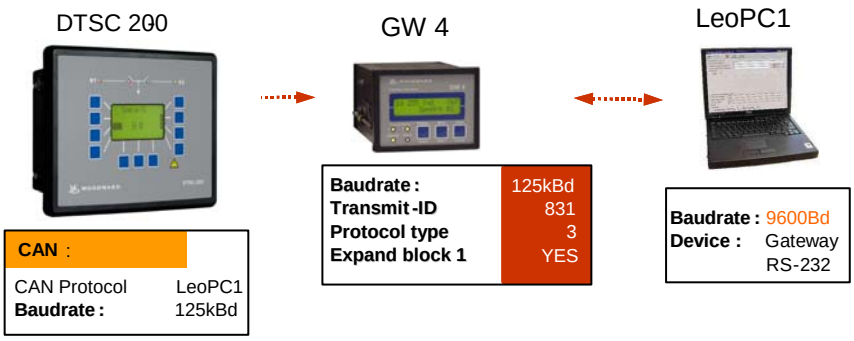


Figure 4-1: CAN (CAL) interface - overview

NOTE

The transmission rate is configurable (default: 125 kBaud). If a GW 4 is used for data transfer, a transmission rate of 125 kBaud must be configured.

The CAN ID, on which the DTSC is transmitting is calculated as follows:

CAN-ID = d‘800 + Item number (or H‘320 + item number)

(The item number is an adjustable parameter in the DTSC, which directly influences the CAN ID that the unit sends the visualization message).

A visualization message which is send out of an DTSC has got 8 Byte and is built as follows:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
H‘DD	MUX number	Data word 1 High-Byte	Data word 1 Low Byte	Data word 2 High-Byte	Data word 2 Low Byte	Data word 3 High-Byte	Data word 3 Low Byte

The byte 0 is always used to show the hexadecimal value H'DD in a visualization message. This defines the message as a visualization message. As the complete transmission telegram of the DTSC includes more than three words byte 1 sends additionally a MUX number starting with 0. Therefore it is theoretically possible to send (256 × 3 = 768) words via the CAN ID. The whole telegram is built up as follows:

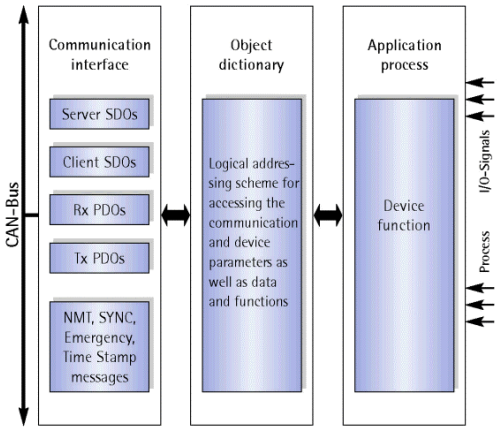
line 1: MUX number 0, word 1
line 2: MUX number 0, word 2
line 3: MUX number 0, word 3
line 4: MUX number 1, word 1
line 5: MUX number 1, word 2
.
line (n): MUX number (n-1/3), word 1
line (n+1): MUX number (n-1/2), word 2
line (n+2): MUX number (n-1/1), word 3

(n) depends on the total length of the unit special telegram and can not be larger than H’FF.
Refer to Appendix A for the interface telegram.

Chapter 5.
CANopen

Introduction

Extract from: Controller Area Network; Basics, Protocols, Chips and Applications; By Prof. Dr.-Ing. K. Etschberger; ISBN: 3-00-007376-0; also see IXXAT GmbH (http://www.ixxat.de)



The CANopen family profile defines a standardized application for distributed industrial automation systems based on CAN as well as the communication standard CAN CAL. CANopen is a standard of CAN-in-Automation (CiA) that after its release, found a broad acceptance, especially in Europe. CANopen can be considered the leading standard for CAN based industrial and embedded system solutions.

The CANopen family profile is based on a "Communication Profile", which specifies the basic communication mechanisms and their description.

The most important device types such as digital and analog I/O modules, drives, operating devices, controllers, programmable controls or encoders, are described by "Device Profiles". The device profiles define the functionality, parameters, and access to process data corresponding to the types of standard devices. These standardized profiles permit devices from different manufacturers to be accessed via the bus in exactly the same manner.

The fundamental element of the CANopen standard is the description of the device functionality through an object dictionary (OD). The object dictionary is divided into two sections. The first section contains general device information like device identification, manufacturer name, etc., as well as communication parameters. The second section describes the specific device functionality.

A 16-Bit index and an 8-Bit sub-index identify the entry ("object") in the object dictionary. Each entry in the object dictionary provide a basis for a standardized network access to the "Application Objects" of a device, such as input and output signals, device parameters, device functions or network variables.

The functionality and characteristics of a CANopen device can be described by means of an "Electronic Data Sheet" (EDS) using the ASCII-format. The EDS acts as a kind of template that describes the data and features, which are accessible via the network. The "Device Configuration File" (DCF) describes the actual device settings. EDS and DCF can be provided in the form of a data carrier, which can be downloaded from the Internet or stored inside the device.

Similar to other well-known field bus systems CANopen also distinguishes two basic data transfer mechanisms: The high-speed exchange of small process data portions through "Process Data Objects (PDO)" as well as the access to entries in the object dictionary through "Service Data Objects (SDO)". The latter ones are primarily used for the transmission of parameters during the device configuration as well as in general for the transmission of larger data portions. Process data object transmissions are generally event triggered, cyclic or requested as broadcast objects without the additional protocol overhead. A PDO can be used for the transmission of a maximum of 8 data bytes. In connection with a synchronization message, the transmission as well as the acceptance of PDOs can be synchronized through the entire network ("Synchronous PDOs"). The assignment of application objects to a PDO (Transmission Object) is adjustable through a structure description ("PDO Mapping") which is stored in the object dictionary, thus allowing the adjustment of a device to the corresponding application requirements.

The transmission of SDOs is performed as a confirmed data transfer with two CAN objects in form of a peer-to-peer connection between two network nodes. The addressing of the corresponding object dictionary entries is accomplished by specifying the index and the sub-index of the entry. Transmitted messages can be unlimited in length. The transmission of SDO messages involves an additional protocol overhead.

Standardized event-triggered "Emergency Messages" of high priority are reserved to report device malfunctions. A common system time can be provided through a central timing message (not included yet).

Management functionality like controlling and monitoring the communication status of the nodes is accomplished by a network management protocol (NMT) organized according to a logical master-slave relationship. Two alternative mechanisms ("Node-Guarding" and "Heartbeat-messages") are available to implement node-monitoring functionality.

The assignment of CAN message identifiers to PDOs and SDOs is possible by direct modifications of entries inside the data structure of the object dictionary or, for simple system structures, through the use of pre-defined identifiers.

Server Data Objects (SDO) - Communication

As already mentioned in the introduction, each CANopen device has an object directory.
All parameters, status variables, measurement values, and input values of the device are stored in this object directory. These parameters are called objects in the CANopen protocol description.
The single objects may contain up to 254 values. If an object has more than one value, these contain a sub-index.

Example: Object 1017h with One Value

Name of the object: Producer Heartbeat Time
Contains a value, which may be read and written.

Example: Object 1200h with Several Values

Name of the object: Server SDO parameter
Sub-index 0 contains the number of sub-indices.
Sub-index 1 contains the COB-ID Client -> Server (rx)
Sub-index 2 contains the COB-ID Server -> Client (tx)

Reading out and changing these objects is performed using an SDO.
This data exchange will be implemented using at least two CAN telegrams, where each on is using an own CAN identifier.

The CAN identifiers of the default service data object are fixed in the object 1200h and are changed using the Node ID.
The values are:
CAN identifier for the reception (Client -> Server): Node ID + 1536 (600h)
CAN identifier for the reply (Server -> Client): Node ID + 1408 (580h)

Some applications require that several SDO clients access one SDO server. To ensure a proper communication, the SDO server must provide several service data objects.
These are described in the objects 1201h to 127Fh.

The DTSC provides five additional service data objects.

These may be configured under the point "Additional S-SDO".
2 to 5 Client->Server COP-ID (tx)
CAN-IDs, on which SDO requests are received.

2 to 5 Server->Client COP-ID (rx)
CAN-IDs, on which SDO replies are sent.

If a unit is not only intended to work as a server, but also as a client, it requires client service data objects.

These may be configured under the point "Additional C-SDO (client SDO)".
1. Client->Server COP-ID (rx)
CAN-IDs, on which SDO requests are sent.

1. Server -> Client COP-ID (tx)
CAN-IDs, on which SDO replies are received.

By entering 80000000h (2147483648 dec) for the CAN ID, the CAN identifiers can be disabled if they are not necessary.

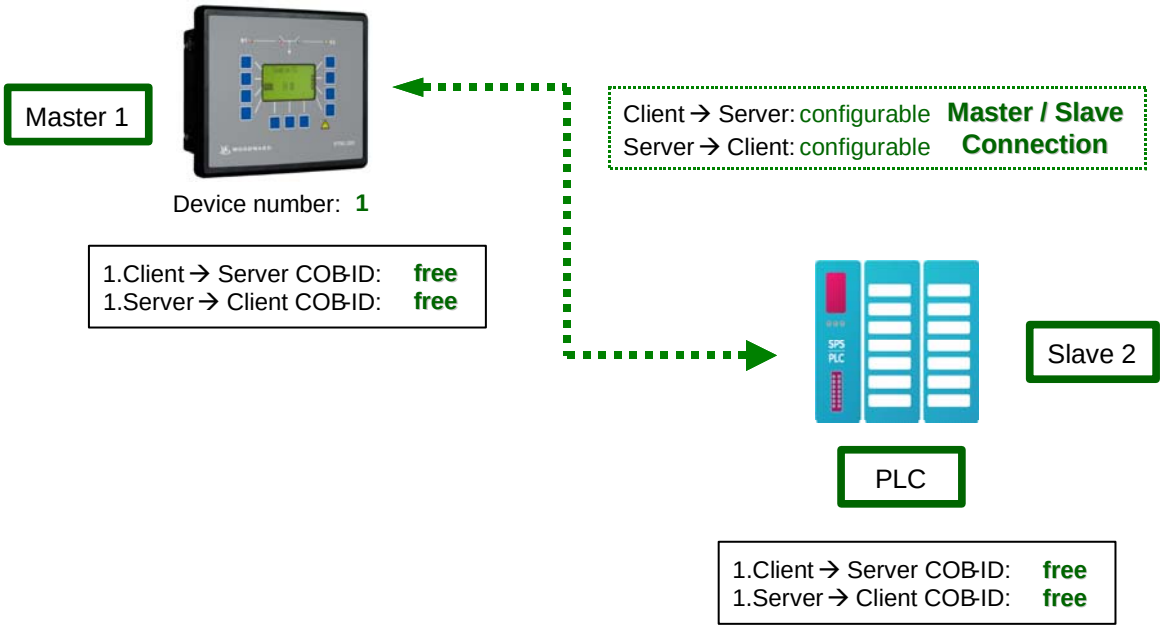


Figure 5-1: CANopen interface - overview

NOTE

If the DTSC-200 is configured to CAN-Open Master = "Yes" and one external terminal, it sends configuration messages to the default service data objects to the connected terminal as SDO client.

Process Data Objects (PDO)

Process data objects are used to transmit real-time data. No, one, or several recipients are possible with this. Process data objects may be sent cyclically or continuously (other transmission types are not supported by the DTSC), this is configured using the parameter "Transmission Type".
The values 254 and 255 define an asynchronous transmission.
In case of the asynchronous transmission, the PDOs are sent after a certain time. This will be configured using the event timer.

The values 1 to 240 are used for a synchronous transmission. The PDO will be sent as a response to a received SYNC message here. If the value is configured to 1, the PDO will be sent for every received SYNC message, if the value is configured to 2, the PDO will only be sent for every 2nd SYNC message, and so on.

No PDOs will be sent for the remaining values.

Data in the PDO

The data, which is transmitted with the PDO, is to be configured at the unit. The parameters "Mapped Object" are provided for this.

The parameter "Number of Mapped Objects" is used to configure the number of mapped objects.

Then, up to four objects may be entered, whose data is to be transmitted. The identifiers of the objects may be found in the operating instructions.

Setting the Transmit PDO (Examples)

With the TPDOs up to 8 data bytes can be send.

Configuration of a data protocol

Parameter	Value
Number of mapped objects	Parameter no. 1 to 4
1. Mapped Object	for example parameter no. 3191
2. Mapped Object	Parameter no. 0
3. Mapped Object	Parameter no. 0
4. Mapped Object	Parameter no. 0

Configuration of a TPDO message

A TPDO can contain one or more mapped objects with a maximum of 4 data bytes each. The TDPO message has a maximum combined total of 8 bytes.

Example 1

Parameter	Value	Number of bytes
Number of mapped objects	Parameter no. 2	
1. Mapped Object	Parameter no. 108	unsigned32 -> 4byte
2. Mapped Object	Parameter no. 160	unsigned16 -> 2byte – total 6 bytes
3. Mapped Object	Parameter no. 0	
4. Mapped Object	Parameter no. 0	

The TPDO has a length of 6 bytes.

Example 2:

Parameter	Value	Number of bytes
Number of mapped objects	Parameter no. 2	
1. Mapped Object	Parameter no. 108	unsigned32 -> 4Byte
2. Mapped Object	Parameter no. 109	unsigned32 -> 4Byte – total 8 bytes
3. Mapped Object	Parameter no. 0	
4. Mapped Object	Parameter no. 0	

The TPDO has a length of 8 bytes.

Example 3:

Parameter	Value	Number of bytes
Number of mapped objects	Parameter no. 3	
1. Mapped Object	Parameter no. 108	unsigned32 -> 4byte
2. Mapped Object	Parameter no. 109	unsigned32 -> 4byte – total 8 bytes
3. Mapped Object	Parameter no. 110	unsigned32 -> 4byte – total 12 bytes !FAULT!
4. Mapped Object	Parameter no. 0	

The TPDO has a length of 12 bytes, as only 8 bytes are admissible, an idle TPDO is sent.

Configuration of a SYNC message

Parameter	Value	Number of bytes
Number of mapped objects	Parameter no. 0	
1. Mapped Object	Parameter no. 0	
2. Mapped Object	Parameter no. 0	
3. Mapped Object	Parameter no. 0	
4. Mapped Object	Parameter no. 0	

The TPDO has a length of 0 bytes. If the COP ID is configured accordingly for example 80h = 128dez, it is working like a SYNC message. Thereby the DTSC has the possibility to send a SYNC message to the attached devices to arrange a reaction with a PDO, however the time of the transmission is not appraised.

SYNC Message

The SYNC message is a CAN message without data. The CAN ID on which the DTSC sends appropriately configured PDOs, is configured with the parameter "COB-ID SYNC Message".

Using a CANopen Configuration Program

If the DTSC is used as a single unit, the default settings provide useful operation possibilities already. If the DTSC is used together with other CANopen devices, a detailed configuration will be necessary.

An *.eds file is enclosed with the unit for this purpose. An example of this file being used with the CANopen Configuration Studio of IXXAT is shown in the following.

Please refer to IXXAT for a more detailed explanation about this tool.

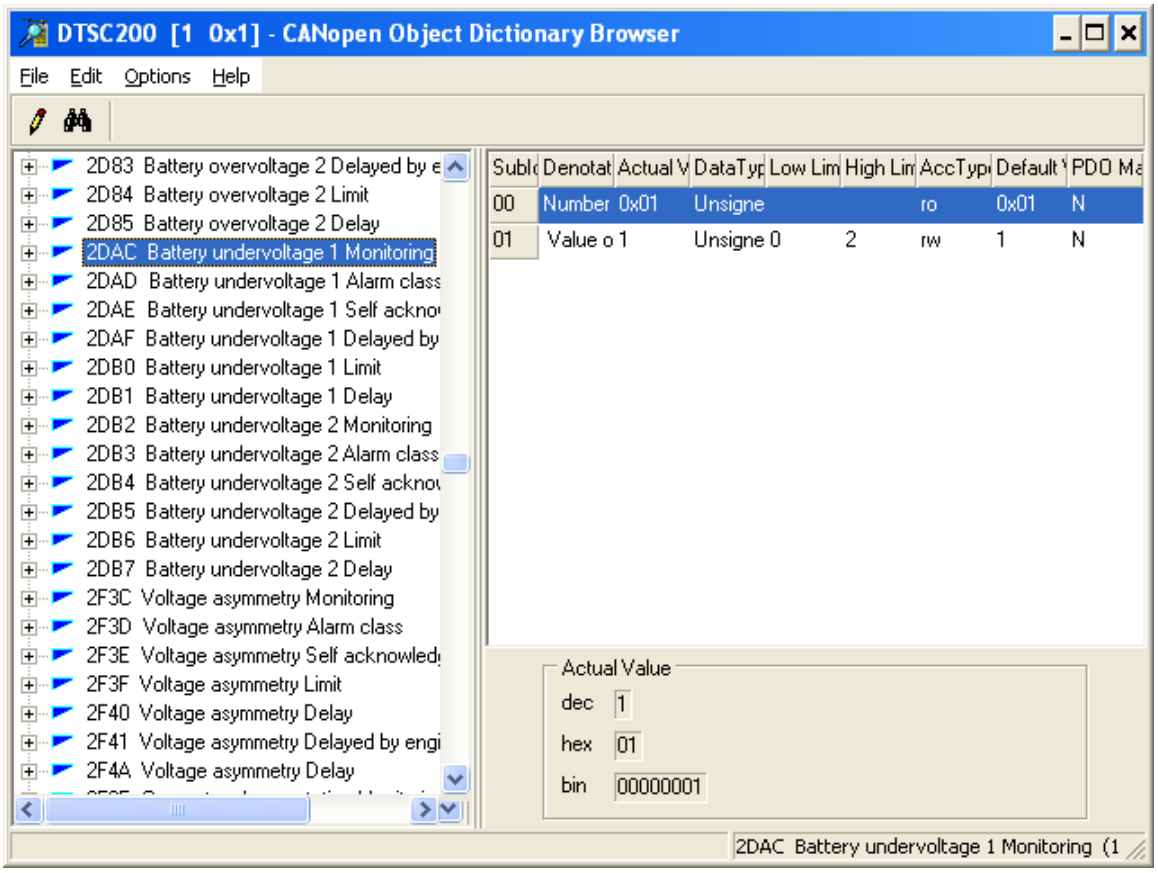


Figure 5-2: CANopen interface - CANopen configuration software

The DTSC parameters may be changed after loading the *.eds file. The values are only overwritten by the DTSC if the correct password has been entered prior to attempting to make any changes; otherwise, a fault message will be issued, which states that the parameter may not be overwritten.

The configuration of the mapped objects of a send PDO is very clear and easy with this program.

Configuration of the transmission type:
The following transmission types are supported:

- "asynchronous (Profile Event)" and "asynchronous (Manuf. Event)" – both send a message after the event timer has expired
- "synchronous cyclic" with the according transmission rate

Settings for Connection with External Devices

Name	Description
Device number	Determines the node ID for CANopen
Protocol	Determines the protocol – select this for CANopen
Baudrate	Determines the baud rate

NOTE

The standard values of the DTSC enable to connect devices on the basis of the CANopen protocol quickly and easily.

Figure 5-3 shows an overview of the different device combinations, which are possible:

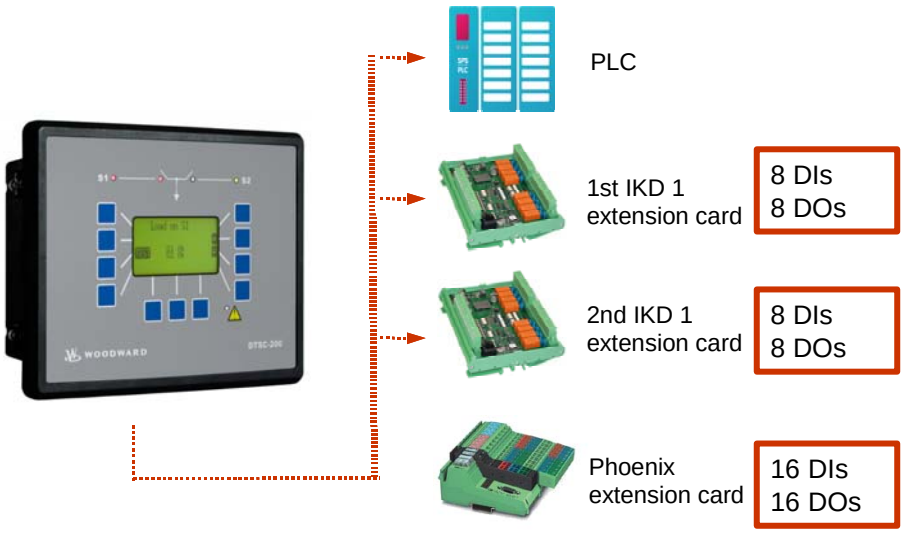


Figure 5-3: CANopen interface - external devices

PLC: PLC of the plant
IKD 1: 2 extension cards, each for 8 additional external inputs and outputs
Phoenix extension card: Extension card for 16 additional external inputs and outputs

NOTE

The parameters, which are highlighted red in the following figures, must be observed particularly, because these are essential for a communication with the respective device and may differ the default values.

CAUTION

The ID settings are entered in hexadecimal format in the DTSC and are therefore listed in decimal and hexadecimal format in the following tables.

Expansion with One IKD 1 (8 Additional External DI/DO)

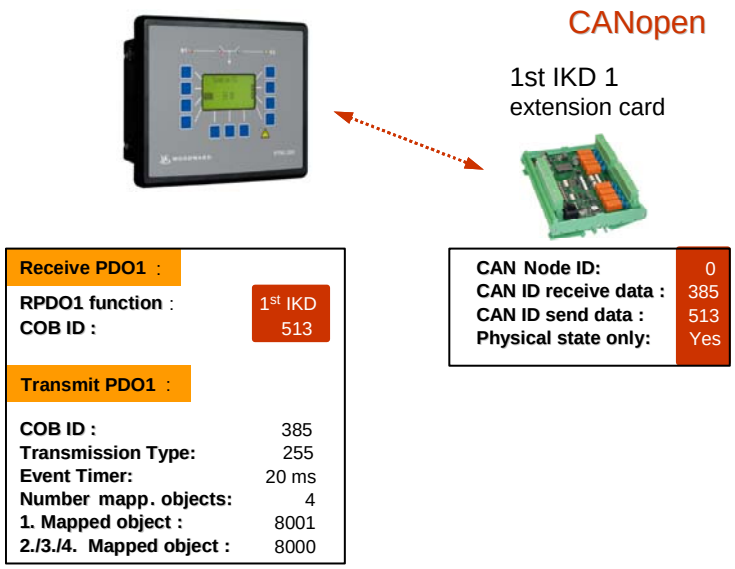


Figure 5-4: CANopen Schnittstelle - Einstellungen für externe Geräte

Configuration of the receive PDO 1

Parameter	Value	Comment
COB-ID	201h = 513 Dec	CAN-ID on which the data are received
Function	1. IKD	The data received on the COB-ID were assigned to the external DI 1 to DI 8
Node-ID of the device	2	The IKD is not configured by the DTSC; the suggested value is therefore a default value.
RPDO-COB-ID ext. device 1	282h = 642 Dec	The IKD is not configured by the DTSC; the suggested value is therefore a default value.

Configuration of transmit PDO (e.g. PDO1)

Parameter	Value	Comments
COB-ID	181h = 385 Dec	CAN-ID on which the data was sent
Transmission type	FFh = 255 Dec	The PDO is sent circular
Event-timer	20	The PDO is sent every 20 ms
Number of mapped objects	4	
1. Mapped Object	Parameter no. 8001	DI 1 to 8 is issued
2. Mapped Object	Parameter no. 8000	
3. Mapped Object	Parameter no. 8000	
4. Mapped Object	Parameter no. 8000	

Settings at the IKD

Parameter	Value	Comments
Node-ID	0	So that the entries of the CAN IDs are taken over
CAN-ID transmitting data	201h = 513 Dec	The DTSC receives on this ID.

Settings for DIs on IKD

Parameter	Value	Comments
Physical state	YES	Only the physical state of the inputs is transmitted. (The settings under idle current, tripping delay, revert delay, enabling, self-resetting and acknowledge input are without effect). These settings have to be selected for devices, which include these parameters (e.g. the DTSC-200).

Check of the settings

Actuate an external DO via the *LogicsManager* and check whether the respective relay at the IKD operates. Scroll the display screens to view the ext. discrete inputs 1 to 8. A set of discrete inputs will be shown that correspond to the IKD. Use the "FAQ CAN Bus" chapter on page 40 to troubleshoot any CAN bus faults.

Expansion with Two IKD 1 (16 Additional External DI/DO)

The first IKD will be adjusted like described above. For the second IKD the following settings must be configured.

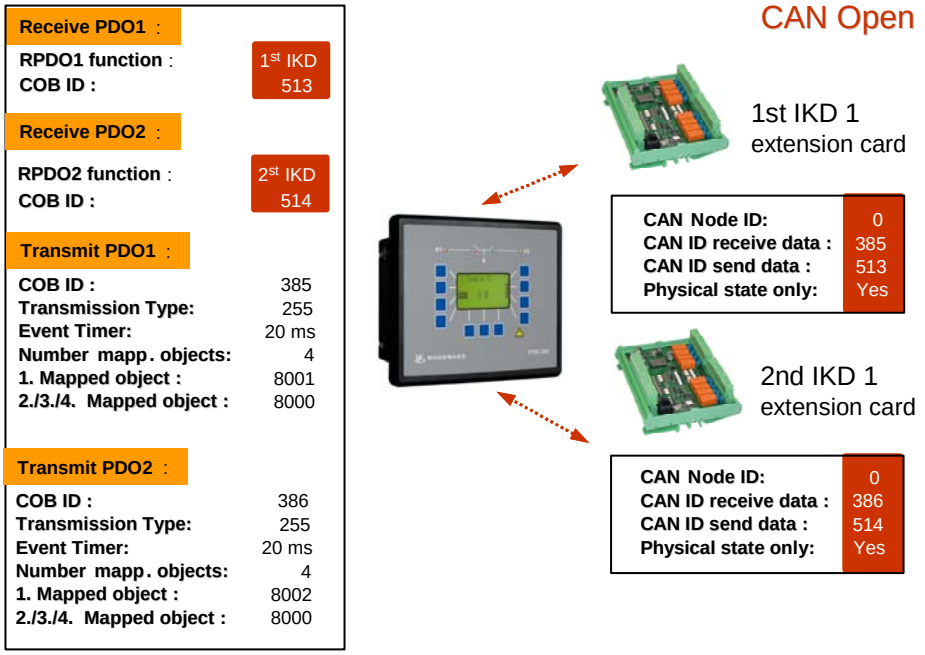


Figure 5-5: CANopen interface - expansion with two IKD 1

Setting of the receive PDO 2

Parameter	Value	Comments
COB-ID	202h = 514 Dec	CAN-ID on which the data are received
Function	2. IKD	The data received on the COB-ID were assigned to the external DI 9 to DI 16
Node-ID of the device	3	The IKD is not configured by the DTSC; the suggested value is therefore a default value.
RPDO-COB-ID ext. device 1	283h = 643 Dec	The IKD is not configured by the DTSC; the suggested value is therefore a default value.

Settings of transmit PDO (e.g. PDO 2)

Parameter	Value	Comments
COB-ID	182h = 386 Dec	CAN-ID on which the data was sent
Transmission type	FFh = 255 Dec	The PDO is sent circular
Event-timer	20	The PDO is sent every 20 ms
Number of mapped objects	4	
1. Mapped Object	Parameter no. 8002	DI 9 to 16 is issued
2. Mapped Object	Parameter no. 8000	
3. Mapped Object	Parameter no. 8000	
4. Mapped Object	Parameter no. 8000	

Settings of DIs on IKD 1 #2

Parameter	Value	Comments
Node-ID	0	That the entries of CAN-IDs are accepted
CAN-ID receiving data	182h = 386 Dec	DTSC receives on this ID
Relay 1 as ready for operation	NO	Otherwise the DTSC cannot be controlled correctly.

Settings on IKD 1 #2

Parameter	Value	Comments
Node-ID	0	So that the entries of the CAN IDs are taken over
CAN-ID transmit-ting data	202h = 514 Dec	The DTSC receives on this ID.

Settings for DIs on IKD 1 #2

Parameter	Value	Comments
Physical state	YES	Only the physical state of the inputs is transmitted. (The settings under idle current, tripping delay, revert delay, enabling, self-resetting and acknowl-edgeinput are without effect). These settings have to be selected for devices, which include these parameters (e.g. the DTSC-200).

Check of the settings

Actuate an external DO via the *LogicsManager* and check whether the respective relay at the IKD operates. Scroll the display screens to view the ext. discrete inputs 9 to 16. A set of discrete inputs will be shown that correspond to the IKD. Use the "FAQ CAN Bus" chapter on page 40 to troubleshoot any CAN bus faults.

Expansion with the Phoenix terminal IL CAN BK / ILB CO 24 16DI 16DO (16 DI/DO)

The specified settings are valid for a Phoenix terminal with Node ID 2.

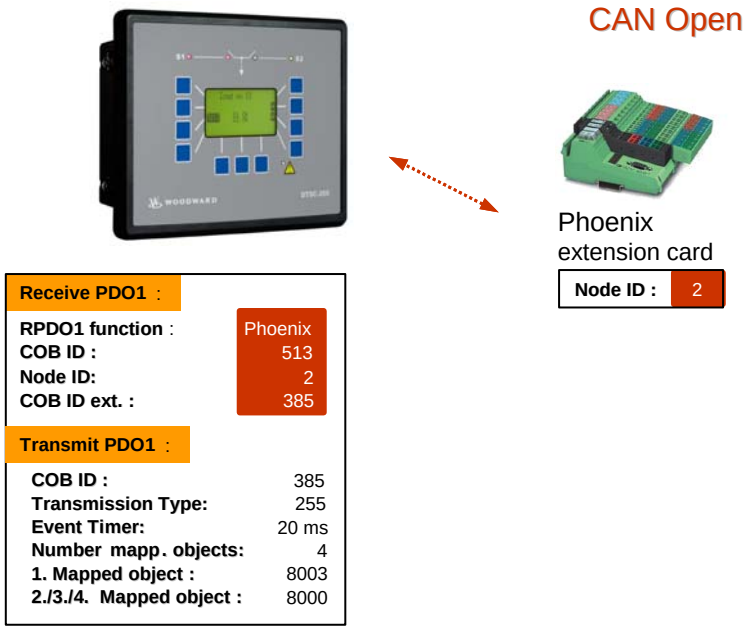


Figure 5-6: CANopen interface - expansion with Phoenix terminal

Parameter	Value	Comments
CAN-open Master	YES	
Max time for reply ext. devices	1.0	
Time for re-init ext. devices	100	If this time is set 0, the attached Phoenix terminal may not be configured correctly.

Setting of the receiving PDO 1

Parameter	Value	Note
COB-ID	201h = 513 Dec	CAN-ID to receive data
Function	BK16DIDO	The received data (via the COB-ID) are copied to the ext. DI 1 to 16
Node-ID of the device	2	According to the setting of the terminals
RPDO-COB-ID ext. device 1	181h = 385 Dec	The Phoenix terminal must be configured in that way that it can receive a PDO on that COB-ID

CAUTION

The 2nd PDO this function must be configured to OFF.

NOTE

The DTSC is the CANopen master.

Settings of the transmitting PDO (i.e. PDO3)

Parameter	Value	Note
COB-ID	381h = 385 Dec	CAN-ID which is used to send data Has to be the same as parameter RPDO-COB-ID of the ext. device 1
Transmission type	FFh = 255 Dec	The PDO is cyclically sent
Event-timer	20	The PDO is sent every 20 ms
Number of mapped objects	1	
1. Mapped Object	Parameter no. 8003	The status of DI 1 to 16 is issued
2. Mapped Object	Parameter no. 0	
3. Mapped Object	Parameter no. 0	
4. Mapped Object	Parameter no. 0	

Check of the settings

Actuate an external DO via the *LogicsManager* and check whether the respective relay at the Phoenix terminal operates.
Scroll the display screens to view the ext. discrete inputs 1 to 8 and ext. discrete inputs 9 to 16. A set of discrete inputs will be shown that correspond to the Phoenix terminal. Use the "FAQ CAN Bus" chapter on page 40 to troubleshoot any CAN bus faults.

Description of the DTSC Parameters

Interfaces: General

EN Device number
DE Gerätenummer
CL2
1702

CAN bus: Device number **1 to 127**

So that this control unit may be positively identified on the CAN bus, the unit address must be set in this parameter. The address may only be represented once on the CAN bus. All other addresses on the CAN bus are calculated on the basis of the address entered in this parameter.

NOTE

If the protocol is CANopen, the Node ID is defined with the device number.

NOTE

The CAN bus is a field bus and subject to various disturbances. Therefore, it cannot be guaranteed that every request will be answered. We recommend to repeat a request, which is not answered within reasonable time.

EN Protocol
DE Protocol
CL2
3155

CAN bus: Protocol **OFF / CANopen / LeoPC**

The CAN bus of this unit may be operated with different protocols and Baud rates. This parameter defines the protocol to be utilized. Please note, that all participants on the CAN bus must use the same protocol.

OFF .....The CAN bus is disconnected. Values are not sent or received.
CANopen .....The CANopen protocol is used.
LeoPC .....The CAN CAL protocol is used.

EN Baudrate
DE Baudrate
CL2
3156

CAN bus: Baudrate **20 / 50 / 100 / 125 / 250 / 500 / 800 / 1,000 kBaud**

The CAN bus of this unit may be operated with different protocols and Baud rates. This parameter defines the used Baud rate. Please note, that all participants on the CAN bus must use the same Baud rate.

General CANopen Parameters

EN CAN-Open Master
DE CAN-open Master
CL2
9000

CANopen Master **YES / NO**

YES..... The DTSC-200 is the CANopen Master.
The unit automatically changes into operational mode and sends broadcast messages (Start_Remote_Node), which cause all other units to change into operational mode as well.
Attached external devices were configured from the unit with SDO messages. The unit sends a SYNC message all 20ms on COB ID 80 Hex.
NO..... The DTSC-200 is a CANopen Slave.

EN Producer heartbeat time
DE Producer heartbeat time
CL2
9120

CAN bus: Producer heartbeat time **20 to 65,530 ms**

Independent from the CANopen Master configuration, the unit transmits a heartbeat message with this configured heartbeat cycle time. If the producer heartbeat time is equal 0, the heartbeat will only be sent as response to a remote frame request. The time configured here will be rounded up to the next 20 ms step.

EN COB-ID SYNC Message
DE COB-ID SYNC Message
CL2
9100

COB-ID SYNC Message **1 to FFFFFFFF**

This parameter defines whether the unit generates the SYNC message or not. Complies to object 1005h (see "Object 1005h: COB-ID SYNC Message" on page 60).

EN Max. answer time ext. devices
DE Max. Antwortzeit ext. Geräte
CL2
9010

Max response time ext. devices **0.1 to 9.9 s**

The maximum time that an attached external device has to answer an SDO message. If the external device fails to answer before this time expires, an abort message is sent and the SDO message will be sent again. This is only effective, if DTSC-200 CAN open master is enabled.

EN Time re-init. ext. devices
DE Zeit Re-init. ext- Geräte
CL2
9009

Time re-init (re-initialization) ext. devices **0 to 9,999 s**

An external device will be configured again with SDO messages after the time set for this parameter.
If 0 is input in this parameter, the external device will not be configured again with SDO messages
This only functions if DTSC-200 CAN open master is enabled.

EN 2nd Client->Server COB-ID (rx)
DE 2. Client->Server COB-ID (rx)
CL2
9020

CAN bus: Client->Server COB-ID (rx) **1 to FFFFFFFF**

In a multi-master application, each Master needs its own identifier (Node ID) from the unit. in order to send remote signals (i.e. acknowledge) to the unit. The additional SDO channel will be made available by configuring this Node ID to a value different than zero. This is the additional CAN ID for the PLC.

EN 2nd Server->Client COB-ID (tx)
DE 2. Server->Client COB-ID (tx)
CL2
9022

CAN bus: Server-> Client COB-ID (tx) **1 to FFFFFFFF**

In a multi-master application, each Master needs its own identifier (Node ID) from the unit. in order to receive remote signals (i.e. acknowledge). The additional SDO channel will be made available by configuring this Node ID to a value different than zero. This is the additional CAN ID for the unit.

EN 3rd Client->Server COB-ID (rx)
DE 3. Client->Server COB-ID (rx)
CL2
9024

CAN bus: Client->Server COB-ID (rx) **1 to FFFFFFFF**

In a multi-master application, each Master needs its own identifier (Node ID) from the unit. in order to send remote signals (i.e. acknowledge) to the unit. The additional SDO channel will be made available by configuring this Node ID to a value different than zero. This is the additional CAN ID for the PLC.

EN 3rd Server->Client COB-ID (tx)
DE 3. Server->Client COB-ID (tx)
CL2
9026

CAN bus: Server-> Client COB-ID (tx) **1 to FFFFFFFF**

In a multi-master application, each Master needs its own identifier (Node ID) from the unit. in order to receive remote signals (i.e. acknowledge). The additional SDO channel will be made available by configuring this Node ID to a value different than zero. This is the additional CAN ID for the unit.

EN 4th Client->Server COB-ID (rx)
DE 4. Client->Server COB-ID (rx)
CL2
9028

CAN bus: Client->Server COB-ID (rx) **1 to FFFFFFFF**

In a multi-master application, each Master needs its own identifier (Node ID) from the unit. in order to send remote signals (i.e. acknowledge) to the unit. The additional SDO channel will be made available by configuring this Node ID to a value different than zero. This is the additional CAN ID for the PLC.

EN 4th Server->Client COB-ID (tx)
DE 4. Server->Client COB-ID (tx)
CL2
9030

CAN bus: Server-> Client COB-ID (tx) **1 to FFFFFFFF**

In a multi-master application, each Master needs its own identifier (Node ID) from the unit. in order to receive remote signals (i.e. acknowledge). The additional SDO channel will be made available by configuring this Node ID to a value different than zero. This is the additional CAN ID for the unit.

EN 5th Client->Server COB-ID (rx)
DE 5. Client->Server COB-ID (rx)
CL2
9032

CAN bus: Client->Server COB-ID (rx) **1 to FFFFFFFF**

In a multi-master application, each Master needs its own identifier (Node ID) from the unit. in order to send remote signals (i.e. acknowledge) to the unit. The additional SDO channel will be made available by configuring this Node ID to a value different than zero. This is the additional CAN ID for the PLC.

EN 5th Server->Client COB-ID (tx)
DE 5. Server->Client COB-ID (tx)
CL2
9034

CAN bus: Server-> Client COB-ID (tx) **1 to FFFFFFFF**

In a multi-master application, each Master needs its own identifier (Node ID) from the unit. in order to receive remote signals (i.e. acknowledge). The additional SDO channel will be made available by configuring this Node ID to a value different than zero. This is the additional CAN ID for the unit.

NOTE

The COB IDs must be entered in decimal numbers in LeoPC1!

CANopen Receive PDO (RPDO) {x} ({x} = 1/2)

Two RPDOs are available.

EN COB-ID
DE COB-ID
CL2
9300
9310

Receive PDO 1/2 - COB-ID **1 to FFFFFFFF**

This parameter contains the communication parameters for the PDOs, the device is able to receive. This corresponds to object 1400h sub index 1h (see "Object 1400h – 141Fh: Receive PDO Communication Parameter" on page 62).

CAUTION

The COB-IDs have to be configured different, even if one RPDO is configured to "no func."2.

EN Function
DE Funktion
CL2
9050
9051

Function for RPDO 1/2 **no func. / 1st IKD /2nd IKD / Bk 16DIDO / Co 16DIDO**

The unit provides pre-configured CAN bus settings for the connection of different units. The unit to be connected must be selected here.

no func...... No external unit is selected for connection. The CAN bus is disabled. Values are not sent or received.
1st IKD ..... The unit is pre-configured for the connection of a Woodward IKD 1 expansion board.
2nd IKD..... The unit is pre-configured for the connection of a second Woodward IKD 1 expansion board.
BK 16 DIDO The unit is pre-configured for the connection of a Phoenix Contact BK 16 DIDO expansion board.
Co 16 DIDO The unit is pre-configured for the connection of a Phoenix Contact Co 16 DIDO expansion board.

Combine Functions with Each Other

PDO1	PDO2		
	1. IKD	2. IKD	OFF
1. IKD	NO	YES	YES
2. IKD	YES	NO	YES
Bk 16DIDO	NO	NO	YES
Co 16DIDO	NO	NO	YES
no func.	YES	YES	YES

Read: If PDO1 is configured as 1. IKD, then PDO2 can only be configured as either 2. IKD or "no func.".

EN Node-ID of the device
DE Node-ID des Gerätes
CL2
9060
9061

Node-ID of the device **1 to 127**

Node-ID of the attached device. The SDO messages were sent on the standard SDO-IDs or the answers were expected.

EN RPDO-COP-ID ext. device {x}
DE RPDO-COP-ID ext. Gerät {x}
CL2
9070
9072

RPDO-COB-ID ext. device 1 **1 to FFFFFFFF**

Value to be written in the object 1800h sub index 1h of the external device.

CAUTION

COB-IDs, which are already used, should not be used.

COB-IDs in a CANopen device after loading the standard values:
280h + Node-ID = 640 + Node-ID Object 1801h Subindex 1
380h + Node-ID = 896 + Node-ID Object 1802h Subindex 1
480h + Node-ID = 1152 + Node-ID Object 1803h Subindex 1

The receiving COB-IDs are preallocated:
300h + Node-ID = 768 + Node-ID Object 1401h Subindex 1
400h + Node-ID = 1024 + Node-ID Object 1402h Subindex 1
500h + Node-ID = 1280 + Node-ID Object 1403h Subindex 1.

Problems may be encountered if a COB-ID is assigned multiple times.

CANopen Transmit PDO (TPDO) {x} ({x} = 1 to 4)

4 TPDOs are available.

EN **COB-ID**
DE **COB-ID**
CL2
9600
9610
9620
9630

CAN bus 1: Transmit PDO 1 - COB ID **1 to FFFFFFFF**

This parameter contains the communication parameters for the PDOs the unit is able to transmit. The unit transmits data (i.e. visualization data) on the CAN ID configured here.

Complies with CANopen specification: object 1800 for (TPDO 1, 1801 for TPDO 2, 1802 for TPDO 3, and 1803 for TPDO 4), subindex 1.

EN **Transmission type**
DE **Transmission type**
CL2
9602
9612
9622
9632

CAN bus 1: Transmit PDO 1 - Transmission type **0 to 255**

This parameter contains the communication parameters for the PDOs the unit is able to transmit. It defines whether the unit broadcasts all data automatically (value 254 or 255) or only upon request with the configured address of the COB ID SYNC message (parameter 9100).

Complies with CANopen specification: object 1800 (for TPDO 1, 1801 for TPDO 2, 1802 for TPDO 3, and 1803 for TPDO 4), subindex 2.

EN **Event-timer**
DE **Event-timer**
CL2
9604
9614
9624
9634

CAN bus 1: Transmit PDO 1 – Event timer **0 to 65000 ms**

This parameter contains the communication parameters for the PDOs the unit is able to transmit. The broadcast cycle for the transmitted data is configured here. The time configured here will be rounded up to the next 5 ms step.

Complies with CANopen specification: object 1800 (for TPDO 1, 1801 for TPDO 2, 1802 for TPDO 3, and 1803 for TPDO 4), subindex 5

EN **Number of Mapped Objects**
DE **Anzahl der Mapped Objekte**
CL2
9609
9619
9629
9639

CAN bus 1: Transmit PDO 1 - Number of mapped objects **0 to 4**

This parameter contains the mapping for the PDOs the unit is able to transmit. This number is also the number of the application variables, which shall be transmitted with the corresponding PDO.

Complies with CANopen specification: object 1A00 (for TPDO 1, 1A01 for TPDO 2, 1A02 for TPDO 3, and 1A03 for TPDO 4), subindex 0

EN 1. Mapped Object
DE 1. Mapped Objekt

CL2
9605
9615
9625
9635

CAN bus 1: Transmit PDO 1 - 1. mapped object **0 to 65535**

This parameter contains the information about the mapped application variables. These entries describe the PDO contents by their index. The sub-index is always 1. The length is determined automatically.

Complies with CANopen specification: object 1A00 (for TPDO 1, 1A01 for TPDO 2, 1A02 for TPDO 3, and 1A03 for TPDO 4), subindex 1

EN 2. Mapped Object
DE 2. Mapped Objekt

CL2
9606
9616
9626
9636

CAN bus 1: Transmit PDO 1 - 2. mapped object **0 to 65535**

This parameter contains the information about the mapped application variables. These entries describe the PDO contents by their index. The sub-index is always 1. The length is determined automatically.

Complies with CANopen specification: object 1A00 (for TPDO 1, 1A01 for TPDO 2, 1A02 for TPDO 3, and 1A03 for TPDO 4), subindex 2

EN 3. Mapped Object
DE 3. Mapped Objekt

CL2
9607
9617
9627
9637

CAN bus 1: Transmit PDO 1 - 3. mapped object **0 to 65535**

This parameter contains the information about the mapped application variables. These entries describe the PDO contents by their index. The sub-index is always 1. The length is determined automatically.

Complies with CANopen specification: object 1A00 (for TPDO 1, 1A01 for TPDO 2, 1A02 for TPDO 3, and 1A03 for TPDO 4), subindex 3

EN 4. Mapped Object
DE 4. Mapped Objekt

CL2
9608
9618
9628
9638

CAN bus 1: Transmit PDO 1 - 4. mapped object **0 to 65535**

This parameter contains the information about the mapped application variables. These entries describe the PDO contents by their index. The sub-index is always 1. The length is determined automatically.

Complies with CANopen specification: object 1A00 (for TPDO 1, 1A01 for TPDO 2, 1A02 for TPDO 3, and 1A03 for TPDO 4), subindex 4

NOTE

CANopen allows to send 8 byte of data with each Transmit PDO. These may be defined separately if no pre-defined data protocol is used.

All data protocol parameters with a parameter ID may be sent as an object with a CANopen Transmit PDO.

In this case, the data length will be taken from the data byte column (refer to the Data Protocols section in the Interface Manual 37389):

- **1,2 UNSIGNED16 or SIGNED16**
- **3,4 UNSIGNED16 or SIGNED16**
- **5,6 UNSIGNED16 or SIGNED16**
- **1,2,3,4 UNSIGNED32 or SIGNED32**
- **3,4,5,6 UNSIGNED32 or SIGNED32**
- **etc.**

The object ID is identical with the parameter ID when configuring via front panel or LeoPC 1.

NOTE

Configuration examples may be found on page 26 "Setting the Transmit PDO (Examples)".

FAQ CAN Bus

The following are reason that no data is transmitted:

- T structure bus is utilized
- CAN-L and CAN-H are interchanged
- Not all devices on the bus are using identical Baud rates
- Terminating resistor are missing
- Baud rate to high for wiring length

Recommendations of Woodward

The maximum length of the communication bus wiring is dependent on the configured Baud rate.

Baud rate	Max. length
1000 kbit/s	25 m
800 kbit/s	50 m
500 kbit/s	100 m
125 kbit/s	250 m
50 kbits/s	1000 m
20 kbit/s	2500 m

Source: CANopen; Holger Zeltwanger (Hrsg.); 2001 VDE VERLAG GMBH, Berlin und Offenbach; ISBN 3-8007-2448-0

The maximum specified length for the communication bus wiring might not be achieved if wire of poor quality is utilized, there is high contact resistance, or other conditions exist. Reducing the baud rate may overcome these issues.

Device Combinations and Bus Load

The baud rate has a direct effect on the number of messages, which may be exchanged via the bus per time unit. A bus load should not exceed approx. 40% capacity to prevent long waiting times or loss of messages.

The following information provides clues for reasonable device configurations at certain baud rates. The exact configuration is to be taken from the respective operation manuals.

20 kBaud

DTSC	PLC	IKD (8DIDO)
1 PDO every 50ms	only receiver	--
2 PDOs every 100 ms	only receiver	--
2 PDOs every 150 ms	1 PDO every 150 ms	--
2 PDOs every 150 ms	only receiver	1 PDO every 160 ms

If the IKD sends only every 160ms, the respective discrete inputs have a jitter of 160ms, it is recommended to receive two messages, therefore, the delay of the ext. discrete inputs should also be configured greater than 160ms.

50 kBaud

DTSC	PLC	BK 16DIDO	IKD (8DIDO)
1 PDO every 20ms (for BK 16DIDO) 1 PDO every 200ms for PLC	only receiver	1 PDO every 20ms	--
1 PDO every 20ms for PLC (e.g. DOs) 1 PDO every 150ms for PLC (e.g. visu data)	1 PDO every 20 ms	Not existing, if the DTSC is the NMT master, set "Time re-init ext. devices" to 0 (off).	--
1 PDO every 20ms (for IKD) 1 PDO every 200ms for PLC	only receiver	---	1 PDO every 20ms
2 PDO every 40ms (for IKD/PLC) 1 PDO every 200ms for PLC	1 PDO every 40ms (may also be the 2.IKD)	---	1 PDO every 40ms

Sometimes the Phoenix CO 16DIDO fails with this baud rate.

100 kBaud

DTSC	PLC	IKD (8DIDO)
1 PDO every 20ms for PLC (e.g. DOs) 1 PDO every 20ms for PLC (e.g. visu data)	1 PDO every 20 ms	
2 PDO every 20ms for PLC (e.g. DOs) 1 PDO every 40ms for PLC (e.g. visu data)	1 PDO every 20ms (may also be the 2.IKD)	1 PDO every 20ms

The Phoenix terminals do not support this baud rate.

125 kBaud

DTSC	PLC / Phoenix BK 16 DIDO	IKD (8DIDO)
4 PDO every 20ms for DO, visualization	1 PDO every 20 ms	
4 PDO every 20ms for DO, visualization	PLC with 1 PDO every 20ms	1 PDO every 20ms

Sometimes the Phoenix CO 16DIDO fails with this baud rate.

250kBaud and above

The maximum load of the CAN bus cannot be reached with combinations of DTSC and external terminals.

A maximum baud rate of 500kBaud may be configured at the IKD.

Appendix A. Telegrams

Transmission Telegram

Data Protocol 4700

Modbus Modicon start addr.	Modbus Start addr. (*1)	CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
450001	450000	0	1,2	3190	Protocol ID, always 4700	1	-	unsigned16
450002	450001	0	3,4,5,6	108	Source 2: Voltage 12	0.1	V	signed32
450004	450003	1	1,2	144	Source 2: Frequency	0.01	Hz	signed16
450005	450004	1	3,4,5,6	114	Source 2: Voltage 1-N	0.1	V	signed32
450007	450006	2	1,2	147	Source 1: Frequency	0.01	Hz	signed16
450008	450007	2	3,4,5,6	109	Source 2: Voltage 23	0.1	V	signed32
450010	450009	3	1,2	160	Source 2: power factor	0.01	-	signed16
450011	450010	3	3,4,5,6	115	Source 2: Voltage 2-N	0.1	V	signed32
450013	450012	4	1,2	10166	Actual Alarm: S1 open failure	Mask : 8000h	Bit	unsigned16
					Actual Alarm: S2 open failure	Mask : 4000h	Bit	
					Actual Alarm: S1 close failure	Mask : 2000h	Bit	
					Actual Alarm: S2 close failure	Mask : 1000h	Bit	
					Actual Alarm: Transfer switch mechanical failure	Mask : 0800h	Bit	
					internal	Mask : 0400h	Bit	
					internal	Mask : 0200h	Bit	
					internal	Mask : 0100h	Bit	
					Actual Alarm: S1 Overvoltage	Mask : 0080h	Bit	
					Actual Alarm: S1 Undervoltage	Mask : 0040h	Bit	
					Actual Alarm: S1 Overfrequency	Mask : 0020h	Bit	
					Actual Alarm: S1 Underfrequency	Mask : 0010h	Bit	
					Actual Alarm: S2 Overvoltage	Mask : 0008h	Bit	
					Actual Alarm: S2 Undervoltage	Mask : 0004h	Bit	
					Actual Alarm: S2 Overfrequency	Mask : 0002h	Bit	
					Actual Alarm: S2 Underfrequency	Mask : 0001h	Bit	
450014	450013	4	3,4,5,6	110	Source 2: Voltage 31	0.1	V	signed32
450016	450015	5	1,2	10167	Latched Alarm: S1 open failure	Mask : 8000h	Bit	unsigned16
					Latched Alarm: S2 open failure	Mask : 4000h	Bit	
					Latched Alarm: S1 close failure	Mask : 2000h	Bit	
					Latched Alarm: S2 close failure	Mask : 1000h	Bit	
					Latched Alarm: Transfer switch mechanical failure	Mask : 0800h	Bit	
					internal	Mask : 0400h	Bit	
					internal	Mask : 0200h	Bit	
					internal	Mask : 0100h	Bit	
					Latched Alarm: S1 Overvoltage	Mask : 0080h	Bit	
					Latched Alarm: S1 Undervoltage	Mask : 0040h	Bit	
					Latched Alarm: S1 Overfrequency	Mask : 0020h	Bit	
					Latched Alarm: S1 Underfrequency	Mask : 0010h	Bit	
					Latched Alarm: S2 Overvoltage	Mask : 0008h	Bit	
					Latched Alarm: S2 Undervoltage	Mask : 0004h	Bit	
					Latched Alarm: S2 Overfrequency	Mask : 0002h	Bit	
					Latched Alarm: S2 Underfrequency	Mask : 0001h	Bit	
450017	450016	5	3,4,5,6	116	Source 2: Voltage 3-N	0.1	V	signed32
450019	450018	6	1,2	10110	Battery voltage	0.1	V	signed16
450020	450019	6	3,4,5,6	118	Source 1: Voltage 12	0.1	V	signed32
450022	450021	7	1,2	10168	Actual Alarm: S1 voltage imbalance	Mask : 8000h	Bit	unsigned16

Modbus Modicon start addr.	Modbus Start addr. (*1)	CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
					Actual Alarm: S2 voltage imbalance	Mask : 4000h	Bit	
					Actual Alarm: S1 Phase rotation mismatch	Mask : 2000h	Bit	
					Actual Alarm: S2 Phase rotation mismatch	Mask : 1000h	Bit	
					Actual Alarm: Inphase-Check timeout	Mask : 0800h	Bit	
					Actual Alarm: Startfailure S2	Mask : 0400h	Bit	
					Actual Alarm: Unintended Stop S2	Mask : 0200h	Bit	
					Actual Alarm: Startfailure S1	Mask : 0100h	Bit	
					Actual Alarm: Unintended Stop S1	Mask : 0080h	Bit	
					Actual Alarm: Overlap time exceeded	Mask : 0040h	Bit	
					internal	Mask : 0020h	Bit	
					internal	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	
450023	450022	7	3,4,5,6	121	Source 1: Voltage 1-N	0.1	V	signed32
450025	450024	8	1,2	10169	Latched Alarm: S1 voltage imbalance	Mask : 8000h	Bit	unsigned16
					Latched Alarm: S2 voltage imbalance	Mask : 4000h	Bit	
					Latched Alarm: S1 Phase rotation mismatch	Mask : 2000h	Bit	
					Latched Alarm: S2 Phase rotation mismatch	Mask : 1000h	Bit	
					Latched Alarm: Inphase-Check timeout	Mask : 0800h	Bit	
					Latched Alarm: Startfailure S2	Mask : 0400h	Bit	
					Latched Alarm: Unintended Stop S2	Mask : 0200h	Bit	
					Latched Alarm: Startfailure S1	Mask : 0100h	Bit	
					Latched Alarm: Unintended Stop S1	Mask : 0080h	Bit	
					Latched Alarm: Overlap time exceeded	Mask : 0040h	Bit	
					internal	Mask : 0020h	Bit	
					internal	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	
450026	450025	8	3,4,5,6	119	Source 1: Voltage 23	0.1	V	signed32
450028	450027	9	1,2	10106	Digital input 1	Mask : 8000h	Bit	unsigned16
					Digital input 2	Mask : 4000h	Bit	
					Digital input 3	Mask : 2000h	Bit	
					Digital input 4	Mask : 1000h	Bit	
					Digital input 5	Mask : 0800h	Bit	
					Digital input 6	Mask : 0400h	Bit	
					Digital input 7	Mask : 0200h	Bit	
					Digital input 8	Mask : 0100h	Bit	
					Digital input 9	Mask : 0080h	Bit	
					Digital input 10	Mask : 0040h	Bit	
					Digital input 11	Mask : 0020h	Bit	
					Digital input 12	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	
450029	450028	9	3,4,5,6	122	Source 1: Voltage 2-N	0.1	V	signed32

Modbus Modicon start addr.	Modbus Start addr. (*1)	CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
450031	450030	10	1,2	10107	Relay-Output 1	Mask : 8000h	Bit	unsigned16
					Relay-Output 2	Mask : 4000h	Bit	
					Relay-Output 3	Mask : 2000h	Bit	
					Relay-Output 4	Mask : 1000h	Bit	
					Relay-Output 5	Mask : 0800h	Bit	
					Relay-Output 6	Mask : 0400h	Bit	
					Relay-Output 7	Mask : 0200h	Bit	
					Relay-Output 8	Mask : 0100h	Bit	
					Relay-Output 9	Mask : 0080h	Bit	
					internal	Mask : 0040h	Bit	
					internal	Mask : 0020h	Bit	
					internal	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	
450032	450031	10	3,4,5,6	120	Source 1: Voltage 31	0.1	V	signed32
450034	450033	11	1,2		internal			
450035	450034	11	3,4,5,6	123	Source 1: Voltage 3-N	0.1	V	signed32
450037	450036	12	1,2		internal	0.1	V	signed16
450038	450037	12	3,4,5,6	111	Load Current Phase A	0.001	A	signed32
450040	450039	13	1,2		internal			unsigned16
450041	450040	13	3,4,5,6	112	Load Current Phase B	0.001	A	signed32
450043	450042	14	1,2	10133	internal	Mask : 8000h	Bit	unsigned16
					internal	Mask : 4000h	Bit	
					internal	Mask : 2000h	Bit	
					internal	Mask : 1000h	Bit	
					internal	Mask : 0800h	Bit	
					internal	Mask : 0400h	Bit	
					internal	Mask : 0200h	Bit	
					internal	Mask : 0100h	Bit	
					internal	Mask : 0080h	Bit	
					internal	Mask : 0040h	Bit	
					internal	Mask : 0020h	Bit	
					internal	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					CAN interface Error	Mask : 0001h	Bit	
450044	450043	14	3,4,5,6	113	Load Current Phase C	0.001	A	signed32
450046	450045	15	1,2	10134	internal	Mask : 8000h	Bit	unsigned16
					internal	Mask : 4000h	Bit	
					internal	Mask : 2000h	Bit	
					internal	Mask : 1000h	Bit	
					internal	Mask : 0800h	Bit	
					internal	Mask : 0400h	Bit	
					internal	Mask : 0200h	Bit	
					internal	Mask : 0100h	Bit	
					Load Overcurrent Limit 1	Mask : 0080h	Bit	
					Load Overcurrent Limit 2	Mask : 0040h	Bit	
					Load Overcurrent Limit 3	Mask : 0020h	Bit	
					internal	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					Load Overload Limit 1	Mask : 0004h	Bit	
					Load Overload Limit 2	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	
450047	450046	15	3,4		Internal			
450048	450047	15	5,6		Internal			
450049	450048	16	1,2		internal			

Modbus Modicon start addr.	Modbus Start addr. (*1)	CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
450050	450049	16	3,4,5,6	136	Load Reactive Power (If Load is powered by Source 2)	1	var	signed32
450052	450051	17	1,2		internal			
450053	450052	17	3,4,5,6	135	Load Real Power (If Load is powered by Source 2)	1	W	signed32
450055	450054	18	1,2	10141	internal	Mask : 8000h	Bit	unsigned16
					internal	Mask : 4000h	Bit	
					internal	Mask : 2000h	Bit	
					internal	Mask : 1000h	Bit	
					internal	Mask : 0800h	Bit	
					internal	Mask : 0400h	Bit	
					internal	Mask : 0200h	Bit	
					internal	Mask : 0100h	Bit	
					internal	Mask : 0080h	Bit	
					internal	Mask : 0040h	Bit	
					internal	Mask : 0020h	Bit	
					internal	Mask : 0010h	Bit	
					Battery overvoltage Limit 2	Mask : 0008h	Bit	
					Battery undervoltage Limit 2	Mask : 0004h	Bit	
					Battery overvoltage Limit 1	Mask : 0002h	Bit	
					Battery undervoltage Limit 1	Mask : 0001h	Bit	
450056	450055	18	3,4		internal			
450057	450056	18	5,6		internal			
450058	450057	19	1,2	10306	Load Power Factor (If Load is powered by Source 2)			signed16
450059	450058	19	3,4		internal			
450060	450059	19	5,6		internal			
450061	450060	20	1,2	10302	Source 2 real power	0.1	kW	signed16
450062	450061	20	3,4,5,6		Source 2 reactive power	0.1	kvar	signed16
450064	450063	21	1,2		internal			
450065	450064	21	3,4,5,6	2520	Real energy	0.01	MWh	unsigned32
450067	450066	22	1,2	10140	Logicsmanager Flag 1 is TRUE	Mask : 8000h	Bit	unsigned16
					Logicsmanager Flag 2 is TRUE	Mask : 4000h	Bit	
					Logicsmanager Flag 3 is TRUE	Mask : 2000h	Bit	
					Logicsmanager Flag 4 is TRUE	Mask : 1000h	Bit	
					Logicsmanager Flag 5 is TRUE	Mask : 0800h	Bit	
					Logicsmanager Flag 6 is TRUE	Mask : 0400h	Bit	
					Logicsmanager Flag 7 is TRUE	Mask : 0200h	Bit	
					Logicsmanager Flag 8 is TRUE	Mask : 0100h	Bit	
					internal	Mask : 0080h	Bit	
					internal	Mask : 0040h	Bit	
					internal	Mask : 0020h	Bit	
					internal	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	
450068	450067	22	3,4,5,6	2522	Reactive Energy	0.01	Mvarh	unsigned32
450070	450069	23	1,2		internal			unsigned16
450071	450070	23	3,4		internal			
450072	450071	23	5,6		internal			
450073	450072	24	1,2		internal			
450074	450073	24	3,4,5,6	10308	internal			

Modbus Modicon start addr.	Modbus Start addr. (*1)	CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
450076	450075	25	1,2	8003	External discrete output 16 [Rex16]	Mask : 8000h	Bit	unsigned16
					External discrete output 15 [Rex15]	Mask : 4000h	Bit	
					External discrete output 14 [Rex14]	Mask : 2000h	Bit	
					External discrete output 13 [Rex13]	Mask : 1000h	Bit	
					External discrete output 12 [Rex12]	Mask : 0800h	Bit	
					External discrete output 11 [Rex11]	Mask : 0400h	Bit	
					External discrete output 10 [Rex10]	Mask : 0200h	Bit	
					External discrete output 9 [Rex9]	Mask : 0100h	Bit	
					External discrete output 8 [Rex8]	Mask : 0080h	Bit	
					External discrete output 7 [Rex7]	Mask : 0040h	Bit	
					External discrete output 6 [Rex6]	Mask : 0020h	Bit	
					External discrete output 5 [Rex5]	Mask : 0010h	Bit	
					External discrete output 4 [Rex4]	Mask : 0008h	Bit	
					External discrete output 3 [Rex3]	Mask : 0004h	Bit	
					External discrete output 2 [Rex2]	Mask : 0002h	Bit	
					External discrete output 1 [Rex1]	Mask : 0001h	Bit	
450077	450076	25	3,4	8013	External discrete input 16 [DIex16]	Mask : 8000h	Bit	unsigned16
					External discrete input 15 [DIex15]	Mask : 4000h	Bit	
					External discrete input 14 [DIex14]	Mask : 2000h	Bit	
					External discrete input 13 [DIex13]	Mask : 1000h	Bit	
					External discrete input 12 [DIex12]	Mask : 0800h	Bit	
					External discrete input 11 [DIex11]	Mask : 0400h	Bit	
					External discrete input 10 [DIex10]	Mask : 0200h	Bit	
					External discrete input 9 [DIex9]	Mask : 0100h	Bit	
					External discrete input 8 [DIex8]	Mask : 0080h	Bit	
					External discrete input 7 [DIex7]	Mask : 0040h	Bit	
					External discrete input 6 [DIex6]	Mask : 0020h	Bit	
					External discrete input 5 [DIex5]	Mask : 0010h	Bit	
					External discrete input 4 [DIex4]	Mask : 0008h	Bit	
					External discrete input 3 [DIex3]	Mask : 0004h	Bit	
					External discrete input 2 [DIex2]	Mask : 0002h	Bit	
					External discrete input 1 [DIex1]	Mask : 0001h	Bit	
450078	450077	25	5,6		internal			unsigned16
450079	450078	26	1,2	10328	Source 1 is Available and Stable	Mask : 8000h	Bit	unsigned16
					Source 2 is Available and Stable	Mask : 4000h	Bit	
					Source 1 is available	Mask : 2000h	Bit	
					Source 2 is available	Mask : 1000h	Bit	
					internal	Mask : 0800h	Bit	
					internal	Mask : 0400h	Bit	
					Source priority is S1	Mask : 0200h	Bit	
					Source priority is S2	Mask : 0100h	Bit	
					internal	Mask : 0080h	Bit	
					internal	Mask : 0040h	Bit	
					internal	Mask : 0020h	Bit	
					internal	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	

Modbus Modicon start addr.	Modbus Start addr. (*1)	CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
450080	450079	26	2,3	10329	internal	Mask : 8000h	Bit	unsigned16
					internal	Mask : 4000h	Bit	
					S1 Start delay timer is timing or expired	Mask : 2000h	Bit	
					S2 Start delay timer is timing or expired	Mask : 1000h	Bit	
					S1 Stable timer is timing or expired	Mask : 0800h	Bit	
					S2 Stable timer is timing or expired	Mask : 0400h	Bit	
					S1 Outage timer is timing or expired	Mask : 0200h	Bit	
					S2 Outage timer is timing or expired	Mask : 0100h	Bit	
					internal	Mask : 0080h	Bit	
					internal	Mask : 0040h	Bit	
					Load is powered by S1	Mask : 0020h	Bit	
					Load is powered by S2	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					A transfer failure occured [OPEN/CLOSE failure]	Mask : 0001h	Bit	
450081	450080	26	4,5	10330	internal	Mask : 8000h	Bit	unsigned16
					internal	Mask : 4000h	Bit	
					internal	Mask : 2000h	Bit	
					Gen-2-Gan application mode is active	Mask : 1000h	Bit	
					Motor Load Disconnect direction is: S1->S2	Mask : 0800h	Bit	
					Motor Load Disconnect direction is: S2->S1	Mask : 0400h	Bit	
					Motor Load Disconnect direction is: BOTH	Mask : 0200h	Bit	
					Synchronicity has been established	Mask : 0100h	Bit	
					Inphase check in progress for transfer direction S1->S2	Mask : 0080h	Bit	
					Inphase check in progress for transfer direction S2->S1	Mask : 0040h	Bit	
					S1 start fail delay counter timing or expired	Mask : 0020h	Bit	
					S2 start fail delay counter timing or expired	Mask : 0010h	Bit	
					Sources OK for inphase-transfer (Both Sources are available and stable)	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	
450082	450081	27	1,2	10331	internal	Mask : 8000h	Bit	unsigned16
					internal	Mask : 4000h	Bit	
					internal	Mask : 2000h	Bit	
					internal	Mask : 1000h	Bit	
					Transfer to S1 is inhibited [for display system]	Mask : 0800h	Bit	
					Transfer to S2 is inhibited [for display system]	Mask : 0400h	Bit	
					internal	Mask : 0200h	Bit	
					internal	Mask : 0100h	Bit	
					internal	Mask : 0080h	Bit	
					internal	Mask : 0040h	Bit	
					S1 cooldown timer is timing or expired	Mask : 0020h	Bit	
					S2 cooldown timer is timing or expired	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	

Modbus Modicon start addr.	Modbus Start addr. (*1)	CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
450083	450082	27	3,4	10332	Neutral timer S1->S2 is timing or expired	Mask : 8000h	Bit	unsigned16
					Neutral timer S2->S1 is timing or expired	Mask : 4000h	Bit	
					Switch reply timer S1->S2 is timing or ex-pired	Mask : 2000h	Bit	
					Switch reply timer S2->S1 is timing or ex-pired	Mask : 1000h	Bit	
					Transfer pause timer S1->S2 is timing or ex-pired	Mask : 0800h	Bit	
					Transfer pause timer S2->S1 is timing or ex-pired	Mask : 0400h	Bit	
					Standard transition mode is selected.	Mask : 0200h	Bit	
					Delayed transition mode is selected	Mask : 0100h	Bit	
					Closed transition mode is selected	Mask : 0080h	Bit	
					internal	Mask : 0040h	Bit	
					internal	Mask : 0020h	Bit	
					Switch is in S1 position	Mask : 0010h	Bit	
					Switch is in S2 position	Mask : 0008h	Bit	
					Switch is in NEUTRAL position	Mask : 0004h	Bit	
					Switch is in OVERLAP position	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	
450084	450083	27	5,6	10333	internal	Mask : 8000h	Bit	unsigned16
					internal	Mask : 4000h	Bit	
					internal	Mask : 2000h	Bit	
					internal	Mask : 1000h	Bit	
					internal	Mask : 0800h	Bit	
					internal	Mask : 0400h	Bit	
					Load shed Signal is active	Mask : 0200h	Bit	
					Load shed Situation is present	Mask : 0100h	Bit	
					internal	Mask : 0080h	Bit	
					internal	Mask : 0040h	Bit	
					internal	Mask : 0020h	Bit	
					internal	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					A Engine Test is requested by HMI	Mask : 0004h	Bit	
					A Load Test is requested by HMI	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	
450085	450084	28	1,2	10334	A Engine Test is active	Mask : 8000h	Bit	unsigned16
					Shunt trip enable Signal is active	Mask : 4000h	Bit	
					Elevator Pre-Signal is active	Mask : 2000h	Bit	
					Motor Load Disconnect Signal is active	Mask : 1000h	Bit	
					Command: Close Switch to S1	Mask : 0800h	Bit	
					Command: Open switch from S1	Mask : 0400h	Bit	
					Command. Close Switch to S2	Mask : 0200h	Bit	
					Command: Open Switch from S2	Mask : 0100h	Bit	
					Engine 1 Start Signal is active	Mask : 0080h	Bit	
					Engine 2 Start Signal is active	Mask : 0040h	Bit	
					A Load Test is active	Mask : 0020h	Bit	
					internal	Mask : 0010h	Bit	
					internal	Mask : 0008h	Bit	
					internal	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	

Modbus Modicon start addr.	Modbus Start addr. (*1)	CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
450086	450085	28	3,4	10165	Logicsmanager Output Flag : ATS Controller is in Inhibit Mode	Mask : 8000h	Bit	unsigned16
					Logicsmanager Output Flag : Remote Peak Shave mode is requested	Mask : 4000h	Bit	
					Logicsmanager Output Flag : Inhibit Transfer to S1 is requested	Mask : 2000h	Bit	
					Logicsmanager Output Flag : Inhibit Transfer to S2 is requested	Mask : 1000h	Bit	
					Logicsmanager Output Flag : Interruptable power rate provisions are re-quested	Mask : 0800h	Bit	
					Logicsmanager Output Flag : Delayed transition mode is forced	Mask : 0400h	Bit	
					Logicsmanager Output Flag : Extended parallel time is requested	Mask : 0200h	Bit	
					Logicsmanager Output Flag : Load shed is requested	Mask : 0100h	Bit	
					Logicsmanager Output Flag : S1 priority is requested	Mask : 0080h	Bit	
					Logicsmanager Output Flag : S2 priority is requested	Mask : 0040h	Bit	
					Logicsmanager Output Flag : External timer Bypass is requested	Mask : 0020h	Bit	
					Logicsmanager Output Flag : No Load Test is requested	Mask : 0010h	Bit	
					Logicsmanager Output Flag : Load Test is requested	Mask : 0008h	Bit	
					Logicsmanager Output Flag : Gen-2-Gen mode is requested	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	
450087	450086	28	5,6	10336	Start Delay timer S1 is timing at the moment	Mask : 8000h	Bit	unsigned16
					Start Delay timer S2 is timing at the moment	Mask : 4000h	Bit	
					Stable timer S1 is timing at the moment	Mask : 2000h	Bit	
					Stable timer S2 is timing at the moment	Mask : 1000h	Bit	
					Outage timer S1 is timing at the moment	Mask : 0800h	Bit	
					Outage timer S2 is timing at the moment	Mask : 0400h	Bit	
					Cooldown timer S1 is timing at the moment	Mask : 0200h	Bit	
					Cooldown timer S2 is timing at the moment	Mask : 0100h	Bit	
					Neutral timer S1 is timing at the moment	Mask : 0080h	Bit	
					Neutral timer S2 is timing at the moment	Mask : 0040h	Bit	
					Switch reply timer S1 is timing at the mo-ment	Mask : 0020h	Bit	
					Switch reply timer S2 is timing at the mo-ment	Mask : 0010h	Bit	
					Transfer pause timer S1 is timing at the mo-ment	Mask : 0008h	Bit	
					Transfer pause timer S2 is timing at the mo-ment	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	

Modbus Modicon start addr.	Modbus Start addr. (*1)	CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
450088	450087	29	1,2	10337	Start Delay timer S1 is expired	Mask : 8000h	Bit	unsigned16
					Start Delay timer S2 is expired	Mask : 4000h	Bit	
					Stable timer S1 is expired	Mask : 2000h	Bit	
					Stable timer S2 is expired	Mask : 1000h	Bit	
					Outage timer S1 is expired	Mask : 0800h	Bit	
					Outage timer S2 is expired	Mask : 0400h	Bit	
					Cooldown timer S1 is expired	Mask : 0200h	Bit	
					Cooldown timer S2 is expired	Mask : 0100h	Bit	
					Neutral timer S1 is expired	Mask : 0080h	Bit	
					Neutral timer S2 is expired	Mask : 0040h	Bit	
					Switch reply timer S1 is expired	Mask : 0020h	Bit	
					Switch reply timer S2 is expired	Mask : 0010h	Bit	
					Transfer pause timer S1 is expired	Mask : 0008h	Bit	
					Transfer pause timer S2 is expired	Mask : 0004h	Bit	
					internal	Mask : 0002h	Bit	
					internal	Mask : 0001h	Bit	

Data Protocol 4800 (Source 1 Data)

CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
0	1,2	15603	Protocol ID, always 4800	1		unsigned16
0	3,4		internal			
0	5,6		internal			
1	1,2,3,4	118	Source 1: Voltage 12	0.1	V	signed32
1	5,6	147	Source 1: Frequency	0.01	Hz	signed16
2	1,2,3,4	119	Source 1: Voltage 23	0.1	V	signed32
2	5,6	10166	Actual Alarm: S1 open failure	Mask : 8000h	Bit	unsigned16
			Actual Alarm: S2 open failure	Mask : 4000h	Bit	
			Actual Alarm: S1 close failure	Mask : 2000h	Bit	
			Actual Alarm: S2 close failure	Mask : 1000h	Bit	
			Actual Alarm: Transfer switch mechanical failure	Mask : 0800h	Bit	
			Actual Alarm: S1 Overvoltage	Mask : 0080h	Bit	
			Actual Alarm: S1 Undervoltage	Mask : 0040h	Bit	
			Actual Alarm: S1 Overfrequency	Mask : 0020h	Bit	
			Actual Alarm: S1 Underfrequency	Mask : 0010h	Bit	
			Actual Alarm: S2 Overvoltage	Mask : 0008h	Bit	
			Actual Alarm: S2 Undervoltage	Mask : 0004h	Bit	
			Actual Alarm: S2 Overfrequency	Mask : 0002h	Bit	
			Actual Alarm: S2 Underfrequency	Mask : 0001h	Bit	
3	1,2,3,4	120	Source 1: Voltage 31	0.1	V	signed32
3	5,6	10167	Latched Alarm: S1 open failure	Mask : 8000h	Bit	unsigned16
			Latched Alarm: S2 open failure	Mask : 4000h	Bit	
			Latched Alarm: S1 close failure	Mask : 2000h	Bit	
			Latched Alarm: S2 close failure	Mask : 1000h	Bit	
			Latched Alarm: Transfer switch mechanical failure	Mask : 0800h	Bit	
			Latched Alarm: S1 Overvoltage	Mask : 0080h	Bit	
			Latched Alarm: S1 Undervoltage	Mask : 0040h	Bit	
			Latched Alarm: S1 Overfrequency	Mask : 0020h	Bit	
			Latched Alarm: S1 Underfrequency	Mask : 0010h	Bit	
			Latched Alarm: S2 Overvoltage	Mask : 0008h	Bit	
			Latched Alarm: S2 Undervoltage	Mask : 0004h	Bit	
			Latched Alarm: S2 Overfrequency	Mask : 0002h	Bit	
			Latched Alarm: S2 Underfrequency	Mask : 0001h	Bit	
4	1,2,3,4	121	Source 1: Voltage 1-N	0.1	V	signed32
4	5,6	10110	Battery voltage	0.1	V	signed16
5	1,2,3,4	122	Source 1: Voltage 2-N	0.1	V	signed32
5	5,6	10168	Actual Alarm: S1 voltage imbalance	Mask : 8000h	Bit	unsigned16
			Actual Alarm: S2 voltage imbalance	Mask : 4000h	Bit	
			Actual Alarm: S1 Phase rotation mismatch	Mask : 2000h	Bit	
			Actual Alarm: S2 Phase rotation mismatch	Mask : 1000h	Bit	
			Actual Alarm: Inphase-Check timeout	Mask : 0800h	Bit	
			Actual Alarm: Startfailure S2	Mask : 0400h	Bit	
			Actual Alarm: Unintended Stop S2	Mask : 0200h	Bit	
			Actual Alarm: Startfailure S1	Mask : 0100h	Bit	
			Actual Alarm: Unintended Stop S1	Mask : 0080h	Bit	
			Actual Alarm: Overlap time exceeded	Mask : 0040h	Bit	
			internal	Mask : 0020h	Bit	
			internal	Mask : 0010h	Bit	
			internal	Mask : 0008h	Bit	
			internal	Mask : 0004h	Bit	
			internal	Mask : 0002h	Bit	
			internal	Mask : 0001h	Bit	
6	1,2,3,4	123	Source 1: Voltage 3-N	0.1	V	signed32
6	5,6	10169	Latched Alarm: S1 voltage imbalance	Mask : 8000h	Bit	unsigned16
			Latched Alarm: S2 voltage imbalance	Mask : 4000h	Bit	
			Latched Alarm: S1 Phase rotation mismatch	Mask : 2000h	Bit	
			Latched Alarm: S2 Phase rotation mismatch	Mask : 1000h	Bit	
			Latched Alarm: Inphase-Check timeout	Mask : 0800h	Bit	
			Latched Alarm:Startfailure S2	Mask : 0400h	Bit	
			Latched Alarm: Unintended Stop S2	Mask : 0200h	Bit	

CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
			Latched Alarm: Startfailure S1	Mask : 0100h	Bit	
			Latched Alarm: Unintended Stop S1	Mask : 0080h	Bit	
			Latched Alarm: Overlap time exceeded	Mask : 0040h	Bit	
			internal	Mask : 0020h	Bit	
			internal	Mask : 0010h	Bit	
			internal	Mask : 0008h	Bit	
			internal	Mask : 0004h	Bit	
			internal	Mask : 0002h	Bit	
			internal	Mask : 0001h	Bit	
7	1,2,3,4	123	Source 1: Voltage 3-N	0.1	V	signed32
7	5,6		internal			
8	1,2,3,4	2520	Real energy	0.01	MWh	unsigned32
8	5,6		internal			
9	1,2,3,4	2522	Reactive Energy	0.01	Mvarh	unsigned32
9	5,6		internal			

Data Protocol 4801 (Source 2 Data)

CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
0	1,2	15603	Protocol ID, always 4801	1		unsigned16
0	3,4		internal			
0	5,6		internal			
1	1,2,3,4	108	Source 2: Voltage 12	0.1	V	signed32
1	5,6	144	Source 2: Frequency	0.01	Hz	signed16
2	1,2,3,4	109	Source 2: Voltage 23	0.1	V	signed32
2	5,6	160	Source 2: Power factor	0.01	-	signed16
3	1,2,3,4	110	Source 2: Voltage 31	0.1	V	signed32
3	5,6	10134	internal	Mask : 8000h	Bit	unsigned16
			internal	Mask : 4000h	Bit	
			internal	Mask : 2000h	Bit	
			internal	Mask : 1000h	Bit	
			internal	Mask : 0800h	Bit	
			internal	Mask : 0400h	Bit	
			internal	Mask : 0200h	Bit	
			internal	Mask : 0100h	Bit	
			Load Overcurrent Limit 1	Mask : 0080h	Bit	
			Load Overcurrent Limit 2	Mask : 0040h	Bit	
			Load Overcurrent Limit 3	Mask : 0020h	Bit	
			internal	Mask : 0010h	Bit	
			internal	Mask : 0008h	Bit	
			Load Overload Limit 1	Mask : 0004h	Bit	
			Load Overload Limit 2	Mask : 0002h	Bit	
			internal	Mask : 0001h	Bit	
4	1,2,3,4	111	Load Current Phase A	0.001	A	signed32
4	5,6	10166	Actual Alarm: S1 open failure	Mask : 8000h	Bit	unsigned16
			Actual Alarm: S2 open failure	Mask : 4000h	Bit	
			Actual Alarm: S1 close failure	Mask : 2000h	Bit	
			Actual Alarm: S2 close failure	Mask : 1000h	Bit	
			Actual Alarm: Transfer switch mechanical failure	Mask : 0800h	Bit	
			internal	Mask : 0400h	Bit	
			internal	Mask : 0200h	Bit	
			internal	Mask : 0100h	Bit	
			Actual Alarm: S1 Overvoltage	Mask : 0080h	Bit	
			Actual Alarm: S1 Undervoltage	Mask : 0040h	Bit	
			Actual Alarm: S1 Overfrequency	Mask : 0020h	Bit	
			Actual Alarm: S1 Underfrequency	Mask : 0010h	Bit	
			Actual Alarm: S2 Overvoltage	Mask : 0008h	Bit	
			Actual Alarm: S2 Undervoltage	Mask : 0004h	Bit	
			Actual Alarm: S2 Overfrequency	Mask : 0002h	Bit	
			Actual Alarm: S2 Underfrequency	Mask : 0001h	Bit	
5	1,2,3,4	112	Load Current Phase B	0.001	A	signed32
5	5,6	10167	Latched Alarm: S1 open failure	Mask : 8000h	Bit	unsigned16
			Latched Alarm: S2 open failure	Mask : 4000h	Bit	
			Latched Alarm: S1 close failure	Mask : 2000h	Bit	
			Latched Alarm:S2 close failure	Mask : 1000h	Bit	
			Latched Alarm: Transfer switch mechanical failure	Mask : 0800h	Bit	
			internal	Mask : 0400h	Bit	
			internal	Mask : 0200h	Bit	
			internal	Mask : 0100h	Bit	
			Latched Alarm: S1 Overvoltage	Mask : 0080h	Bit	
			Latched Alarm: S1 Undervoltage	Mask : 0040h	Bit	
			Latched Alarm: S1 Overfrequency	Mask : 0020h	Bit	
			Latched Alarm: S1 Underfrequency	Mask : 0010h	Bit	
			Latched Alarm: S2 Overvoltage	Mask : 0008h	Bit	
			Latched Alarm: S2 Undervoltage	Mask : 0004h	Bit	
			Latched Alarm: S2 Overfrequency	Mask : 0002h	Bit	
			Latched Alarm: S2 Underfrequency	Mask : 0001h	Bit	
6	1,2,3,4	113	Load Current Phase C	0.001	A	signed32
6	5,6	10132	Digital input 1 is set	Mask : 8000h	Bit	unsigned16

CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
			Digital input 2 is set	Mask : 4000h	Bit	
			Digital input 3 is set	Mask : 2000h	Bit	
			Digital input 4 is set	Mask : 1000h	Bit	
			Digital input 5 is set	Mask : 0800h	Bit	
			Digital input 6 is set	Mask : 0400h	Bit	
			Digital input 7 is set	Mask : 0200h	Bit	
			Digital input 8 is set	Mask : 0100h	Bit	
			Digital input 9 is set	Mask : 0080h	Bit	
			Digital input 10 is set	Mask : 0040h	Bit	
			Digital input 11 is set	Mask : 0020h	Bit	
			Digital input 12 is set	Mask : 0010h	Bit	
			internal	Mask : 0008h	Bit	
			internal	Mask : 0004h	Bit	
			internal	Mask : 0002h	Bit	
			internal	Mask : 0001h	Bit	
7	1,2,3,4	114	Source 2: Voltage 1-N	0.1	V	signed32
7	5,6	10133	internal	Mask : 8000h	Bit	unsigned16
			internal	Mask : 4000h	Bit	
			internal	Mask : 2000h	Bit	
			internal	Mask : 1000h	Bit	
			internal	Mask : 0800h	Bit	
			internal	Mask : 0400h	Bit	
			internal	Mask : 0200h	Bit	
			internal	Mask : 0100h	Bit	
			internal	Mask : 0080h	Bit	
			internal	Mask : 0040h	Bit	
			internal	Mask : 0020h	Bit	
			internal	Mask : 0010h	Bit	
			internal	Mask : 0008h	Bit	
			internal	Mask : 0004h	Bit	
			internal	Mask : 0002h	Bit	
			CAN interface Error	Mask : 0001h	Bit	
8	1,2,3,4	115	Source 2: Voltage 2-N	0.1	V	signed32
8	5,6	10141	internal	Mask : 8000h	Bit	unsigned16
			internal	Mask : 4000h	Bit	
			internal	Mask : 2000h	Bit	
			internal	Mask : 1000h	Bit	
			internal	Mask : 0800h	Bit	
			internal	Mask : 0400h	Bit	
			internal	Mask : 0200h	Bit	
			internal	Mask : 0100h	Bit	
			internal	Mask : 0080h	Bit	
			internal	Mask : 0040h	Bit	
			internal	Mask : 0020h	Bit	
			internal	Mask : 0010h	Bit	
			Battery overvoltage Limit 2	Mask : 0008h	Bit	
			Battery undervoltage Limit 2	Mask : 0004h	Bit	
			Battery overvoltage Limit 1	Mask : 0002h	Bit	
			Battery undervoltage Limit 1	Mask : 0001h	Bit	
9	1,2,3,4	116	Source 2: Voltage 3-N	0.1	V	signed32
9	5,6	10168	Actual Alarm: S1 voltage imbalance	Mask : 8000h	Bit	unsigned16
			Actual Alarm: S2 voltage imbalance	Mask : 4000h	Bit	
			Actual Alarm: S1 Phase rotation mismatch	Mask : 2000h	Bit	
			Actual Alarm: S2 Phase rotation mismatch	Mask : 1000h	Bit	
			Actual Alarm: Inphase-Check timeout	Mask : 0800h	Bit	
			Actual Alarm: Startfailure S2	Mask : 0400h	Bit	
			Actual Alarm: Unintended Stop S2	Mask : 0200h	Bit	
			Actual Alarm: Startfailure S1	Mask : 0100h	Bit	
			Actual Alarm: Unintended Stop S1	Mask : 0080h	Bit	
			Actual Alarm: Overlap time exceeded	Mask : 0040h	Bit	
			internal	Mask : 0020h	Bit	
			internal	Mask : 0010h	Bit	
			internal	Mask : 0008h	Bit	
			internal	Mask : 0004h	Bit	

CAN Data Byte 0 (Mux)	Data Byte	Para-meter ID	Description	Multiplier	Units	Data Type
			internal	Mask : 0002h	Bit	
			internal	Mask : 0001h	Bit	
10	1,2,3,4	135	Load Real Power (if load is powered by Source 2)	1	W	signed32
10	5,6	10169	Latched Alarm: S1 voltage imbalance	Mask : 8000h	Bit	unsigned16
			Latched Alarm: S2 voltage imbalance	Mask : 4000h	Bit	
			Latched Alarm: S1 Phase rotation mismatch	Mask : 2000h	Bit	
			Latched Alarm: S2 Phase rotation mismatch	Mask : 1000h	Bit	
			Latched Alarm: Inphase-Check timeout	Mask : 0800h	Bit	
			Latched Alarm: Startfailure S2	Mask : 0400h	Bit	
			Latched Alarm: Unintended Stop S2	Mask : 0200h	Bit	
			Latched Alarm: Startfailure S1	Mask : 0100h	Bit	
			Latched Alarm: Unintended Stop S1	Mask : 0080h	Bit	
			Latched Alarm: Overlap time exceeded	Mask : 0040h	Bit	
			internal	Mask : 0020h	Bit	
			internal	Mask : 0010h	Bit	
			internal	Mask : 0008h	Bit	
			internal	Mask : 0004h	Bit	
			internal	Mask : 0002h	Bit	
			internal	Mask : 0001h	Bit	
11	1,2,3,4	136	Load Reactive Power (if load is powered by Source 2)	1	var	signed32
11	5,6	10306	Load Power Factor (if load is powered by Source 2)			signed16

Remote Control Telegram

Parameter No.	Object ID	Name		Unit	Data type	Note
503	21F7h	Control word 1		Bit field	Unsigned16	
		Bit 15	Not used			
		Bit 14	Not used			
		Bit 13	Not used			
		Bit 12	Not used			
		Bit 11	Not used			
		Bit 10	Not used			
		Bit 9	Not used			
		Bit 8	Not used			
		Bit 7	Not used			
		Bit 6	Not used			
		Bit 5	Not used			
		Bit 4	Remote acknwledgement : reset alarm messages (rise of the pulse)			Transmit first a 0, then a 1 to acknowledge
		Bit 3	Must always be configured to 0			
		Bit 2	Must always be configured to 0			
		Bit 1	Not used			
		Bit 0	Not used			

Bit 4 "Remote acknowledgement: reset alarm messages"
This bit controls the logical command variable 04.14.
This command must be executed twice.
The first rise of the pulse resets the horn and the second rise of the pulse acknowledges a fault, which is not present anymore.

Appendix B.
CANopen

Description of the Common Data Types

Structure of the PDO-COB-ID Entry (UNSIGNED32)

MSB					LSB
Bits	31	30	29	28-11	10 – 0
11 bit ID	0/1	0	0	all 0	11 bit identifier
29 bit ID	0/1	0	1	29 bit identifier	

Description of the PDO-COB-ID entry

Bit number	Value	Description
31 (MSB)	0	PDO exists / is valid
	1	PDO does not exist / is invalid
30	0	Device does not generate SYNC message
	1	Device generates SYNC message
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28 – 11	0 X	If bit 29=0 and if bit 29=1: bits 28-11 of 29-bit-SYNC-COB-ID
10-0 (LSB)	X	Bits 10-0 of SYNC-COB-ID

Transmission Types (PDO Transmission)

	cyclically	continuously	synchronous	asynchronous	RTR only
0 *	--	X	X	--	--
1-240	X	--	X	--	--
241-251	----------	----------	reserved	----------	----------
252 *	--	--	X		X
253 *	--	--	--	X	X
254	--	--	--	X	--
255	--	--	--	X	--

* not supported

Description of the Object Parameter

Object 1000h: Device Type

This contains information about the type of the participant.

Object description

Index 1000h
Name ..... Device Type
Object code ..... VAR
Data type UNSIGNED32
Category ..... obligatory

Entry description

Access ..... Read Only
PDO figure ..... no
Value range UNSIGNED32
Default value ..... 0 h no standard profile

Object 1001h: Error Register

This object is an error register for the participant.

Object description

Index 1001h
Name ..... Error Register
Object code ..... VAR
Data type UNSIGNED8
Category ..... obligatory

Entry description

Access ..... Read Only
PDO figure ..... no
Value range UNSIGNED8
Default value ..... no

Note

This object is always value 0.

Object 1005h: COB-ID SYNC Message

The index 1005h defines the COB-ID of the synchronization object (SYNC).

Description of the SYNC-COB-ID entry (UNSIGNED32)

MSB					**LSB**
Bits	31	30	29	28-11	10 – 0
11 Bit-ID	X	0/1	0	all 0	11-bit Identifier
29 Bit-ID	X	0/1	1	29-bit Identifier	

Description of the SYNC-COB-ID entry

Bit number	**Value**	**Description**
31 (MSB)	0/1	0 = valid / 1 = invalid
30	0	Device does not generate SYNC message
	1	Device generates SYNC message
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2,0B)
28 – 11	0 X	If bit 29=0 and if bit 29=1: bits 28-11 of 29-bit-SYNC-COB-ID
10-0 (LSB)	X	Bits 10-0 of SYNC-COB-ID

Object description

Index.....1005h
Name.....COB-ID SYNC
Object code.....VAR
Data type.....UNSIGNED32

Entry description

Access.....Read/Write
PDO figure.....no
Value range.....UNSIGNED32
Default value .....80 hex

Note

Bit 31-29 are ignored. Writing these bits does not cause faults. The bit 28-11 should be configured to 0. This parameter can be configured using the parameter COB-ID SYNC Message. If a SYNC message is to be sent the PDO can be configured in that way that it contains no values.

Object 1017h: Producer Heartbeat Time

The object Producer Heartbeat Time defines the heartbeat cycle time in ms. If no Producer Heartbeat (NMT Error Control) is to be sent, this is to be configured to 0.

Object description

Index 1017h
Name ..... Producer Heartbeat Time
Object code ..... VAR
Data type UNSIGNED16

Entry description

Access ..... Read/Write
PDO figure ..... no
Value range UNSIGNED16
Default value 240

Note

The time is extended to the next full 20 ms. If the time is 0, the (NMT Error Control) will be sent as response to a remote frame.

Object 1018h: Identity Object

The object contains common information of one participant.

Object description

Index 1018h
Name ..... Identity Object
Object code ..... RECORD
Data type ..... Identity
Category ..... obligatory

Entry description

Sub index 0h

Description ..... Number of entries
Entry category ..... obligatory
Access ..... Read Only
PDO figure ..... no
Value range 1
Default value 1

Sub index 1h

Description ..... Vendor ID
Entry category ..... obligatory
Access ..... Read Only
PDO figure ..... no
Value range UNSIGNED32
Default value 0

Object 1200h – 1201h: Server SDO Parameter

Objects are not supported.

The receive SDO is: 600h+Node-ID
The transmit SDO for answers is 580h+Node-ID
The Node ID can be entered using the parameter "Unit number".

Object 1400h – 141Fh: Receive PDO Communication Parameter

This object contains the communication parameter for the PDOs that can be received from the participant. The sub index 0h contains the number of valid entries within the communication recording. The sub index 1h contains the COB ID of the PDO. The interpretation of the entry occurs according to the tables "Structure of the PDO-COB-ID entry" and the "Description of the POD-COB-ID entry".

Object description

Index.....1400h — 141Fh
Name.....Receive PDO parameter
Object code.....RECORD
Data type.....PDO CommPar
Category .....conditioned; obligatory for every supported PDO

Entry description

Sub index.....0h
Description .....Largest Sub index supported
Entry category .....obligatory
Access.....Read Only
PDO figure.....no
Value range.....2

Sub index 1h

Description .....COB-ID used by PDO
Entry category .....obligatory
Access.....Read Only; Read/Write if variable COB-ID is supported
PDO figure.....no
Value range.....UNSIGNED32 (Table 54)
Default value .....Index 1400h: 200h + Node-ID,
Index 1401h: 300h + Node-ID,
Index 1402h: 400h + Node-ID,
Index 1403h: 500h + Node-ID,
Index 1404h - 15FFh: disabled

Sub index 2h

Description .....Transmission type
Entry category .....obligatory
Access.....Read Only
PDO figure.....no
Value range.....UNSIGNED8 (Table 55)
Default value .....(Device Profile dependent)

Note

The device possesses only two RPDOs. Therefore the objects 1402h-141Fh are not available.

Sub index 1h

The bits 30-29 were ignored. Writing these bits do not cause faults. The bits 28-11 should be configured to 0. This value can be set in the display mask "COB-ID" in sub menu CAN-OPEN RPDO 1 / 2.

Sub index 2h

This value is always set 0xFF.

Object 1600h – 161Fh: Receive PDO Mapping Parameter

Is not used. The receive PDOs can be assigned to defined functions. The corresponding parameter can be set in the display screen "Function" in sub menu CAN-OPEN RPDO 1 / 2.

Object 1800h – 181Fh: Transmit PDO Communication Parameter

Includes the communication parameter for the PDOs that can be sent from the participant.

Object description

Index ..... 1800h — 181Fh
Name ..... Transmit PDO parameter
Object code ..... RECORD
Data type ..... PDO CommPar
Category ..... conditioned; obligatory for every supported PDO

Entry description

Sub index 0h

Description ..... Largest Sub index supported
Entry category ..... obligatory
Access ..... Read Only
PDO figure ..... no
Value range 5

Sub index 1h

Description ..... COB-ID used by PDO
Entry category ..... obligatory
Access ..... Read Only; Read/Write if COB-ID can be configured
PDO figure ..... no
Value range ..... UNSIGNED32 (Figure 65)
Default value: ..... Index 1800h: 181h,
Index 1801h: 281h,
Index 1802h: 381h,
Index 1803h: 481h, because Default value for Node-ID is 1.

Sub index 2h

Description ..... Transmission type
Entry category ..... obligatory
Access ..... Read Only; Read/Write if transmission type can be changed
PDO figure ..... no
Value range ..... UNSIGNED8 (Table 54)
Default value 0

Sub index 5h

Description ..... Event timer
Entry category ..... optional
Access ..... Read/Write
PDO figure ..... no
Value range ..... 0 = not used UNSIGNED16
Default value 20

Note

Sub index 1h

The bits 31-29 were ignored. Writing these bits does not cause faults. The bits 28-11 should be configured to 0. This sub index can be set in the display screens "COB-ID" in sub menu CAN-OPEN TPDO 1 / 2 / 3 / 4.

Sub index 2h

Value	Function
0	A PDO will not be sent
1-240	A PDO will be sent as answer to a SYNC message
241-251	A PDO will not be sent
252-253	A PDO will not be sent
254-255	A PDO will be sent cyclically

This sub index does not change the PDO communication parameter screen. This sub index can be set in the display screen "Transmission type" in sub menu CAN-OPEN TPDO 1 / 2 / 3 / 4.

Sub index 5h

The time is rounded up to the next full 5 ms. The sub index can be set in the display screen "Event-timer" in sub menu CAN-OPEN TPDO 1 / 2 / 3 / 4.

Object 1A00h – 1A1Fh: Transmit PDO Mapping Parameter

The mapping for the PDOs, which the participant can send, is located here. An exact description of the entries can be found in the chapter "Parameter description".

CAUTION

The parameter can be configured only if the respective PDO is valid (Object 1800 Sub index 1 Bit 31 is set).

Object description

Index.....1A00h — 1A1Fh
Name.....Transmit PDO mapping
Object code.....RECORD
Data type.....PDO figure
Category .....conditioned; obligatory for every supported PDO

Entry description

Sub index 0h

Description ..... number of mapped application objects in PDO
Entry category ..... obligatory
Access ..... Read Only; Read/Write if dynamic mapping is supported
PDO figure ..... no
Value range 4
Default value 4

Sub index 1h - 4h

Description ..... PDO mapping for the n^{th} application object to be mapped
Entry category ..... conditioned, dependent on the number and size of the objects
Access ..... Read/Write
PDO figure ..... no
Value range UNSIGNED32
Default value ..... (Device profile dependent)

Note

Sub index 0h

The sub index 0 cannot be changed. Writing does not cause fault messages however the value will not be saved. For configuration of the other sub indexes the sub index 0h has to be set **not** 0.

Sub index 1h-4h

You have to enter the object numbers from the EDS file into the sub indexes 1h-4h. The sub indexes 1h-4h can be set in the display masks "1-4 Mapped Object" in sub menu CAN-OPEN TPDO 1 / 2 / 3 / 4.

CAUTION

With configuration over CAN open the object ID is to be used (see EDS file).

With configuration over display/LeoPC1 the parameter number is to be used (see "CANopen: Mapping Parameter" after page 71.)

Data Format of Different Functions

Depending on the selected RPDO function a different data format will be expected.

Receiving Messages

1.IKD / 2.IKD

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
01	Bit 0 DI1 Bit 1 DI 2 + + + Bit 7 DI 8	not analyzed	not analyzed	not analyzed	not analyzed	not analyzed	not analyzed

Phoenix16

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Bit 0 DI 1 Bit 1 DI 2 + + + Bit 7 DI 8	Bit 0 DI 9 Bit 1 DI 10 + + + Bit 7 DI 16	not analyzed	not analyzed	not analyzed	not analyzed	not analyzed	not analyzed

CAUTION

Please note for combination of the different functions.

CAUTION

Configuration of the Phoenix terminal, if the DTSC is not CAN open master.

If the discrete inputs of the Phoenix terminal shall be evaluated by the DTSC, it must be configured this way that the corresponding discrete inputs in byte 1 and byte 2 are available for the received PDO. This PDO must be sent independently from the terminal. The DTSC does not pick up PDOs with remote frames.

The receiving PDO of the Phoenix terminal and the corresponding transmitting PDO of the DTSC must be adjusted on both units.

Definition of Protocol Descriptions

If in a PDO a protocol number is entered as 1. Mapped object, a data array with 8x unsigned8 is sent.

The denotation is:

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
MUX	Data byte	Data byte	Data byte	Data byte	Data byte	Data byte	internal

The MUX byte is counted up, the meaning of the data byte changes according to the value of the MUX byte.
In the protocol tables is listed which parameter at which MUX on which position is transmitted.
The meaning of the parameter can be taken by means of the number of the parameter description ("CANopen Mapping parameter").

Example:

MUX	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
1	118				147		internal

In MUX 1 (byte 1 has got value 1) the value of parameter 118 is included in the byte 2 up to byte 5 (mains voltage 1-2).
In byte 6 up to byte 7 the value of parameter 147 is included (mains frequency).
Byte 8 includes internal definitions and can be ignored.
The data format is low Byte/high Byte (compare with CiA draft standard 01 on page 26).

Unsigned Integer

UNSIGNED type data has positive integers as values. The range is between 0 and 2n-1. The data is shown by the bit sequence of length n.

Bit sequence $b = b_0$ to b_{n-1}

shows the value $UNSIGNEDn(b) = b_{n-1}*2^{n-1}+....+b_1*2^1+b_0*2^0$

NOTE

Please note that the bit sequence starts on the left with the least significant byte.

Example: Value 266 = 10Ah of type UNSIGNED16 is transmitted on the bus in two octets, first 0Ah and then 01h.

The following UNSIGNED data types are transmitted as follows:

Octet Number	1.	2.	3.	4.	5.	6.	7.	8.
UNSIGNED8	b_7 to b_0							
UNSIGNED16	b_7 to b_0	b_{15} to b_8						
UNSIGNED24	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}					
UNSIGNED32	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}	b_{31} to b_{24}				
UNSIGNED40	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}	b_{31} to b_{24}	b_{39} to b_{32}			
UNSIGNED48	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}	b_{31} to b_{24}	b_{39} to b_{32}	b_{47} to b_{40}		
UNSIGNED56	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}	b_{31} to b_{24}	b_{39} to b_{32}	b_{47} to b_{40}	b_{55} to b_{48}	
UNSIGNED64	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}	b_{31} to b_{24}	b_{39} to b_{32}	b_{47} to b_{40}	b_{55} to b_{48}	b_{63} to b_{56}

Signed Integer

SIGNED type data has integers as values. The range is between 0 and 2^n-1. The data is shown by the bit sequence of length n.

Bit sequence $b = b_0$ to b_{n-1}

shows the value $SIGNEDn(b) = b_{n-2}*2^{n-2}+...+b_1*2^1+b_0*2^0$ if $b_{n-1} = 0$

and with two's complement $SIGNEDn(b) = SIGNEDn(\hat{}b)-1$ if $b_{n-1} = 1$

NOTE

Please note that the bit sequence starts on the left with the least significant byte.

Example: The value -266 = FEF6h of type SIGNED16 is transmitted in two octets, first F6h and then FEh.

The following SIGNED data types are transmitted as follows:

Octet Number	1.	2.	3.	4.	5.	6.	7.	8.
SIGNED8	b_7 to b_0							
SIGNED16	b_7 to b_0	b_{15} to b_8						
SIGNED24	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}					
SIGNED32	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}	b_{31} to b_{24}				
SIGNED40	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}	b_{31} to b_{24}	b_{39} to b_{32}			
SIGNED48	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}	b_{31} to b_{24}	b_{39} to b_{32}	b_{47} to b_{40}		
SIGNED56	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}	b_{31} to b_{24}	b_{39} to b_{32}	b_{47} to b_{40}	b_{55} to b_{48}	
SIGNED64	b_7 to b_0	b_{15} to b_8	b_{23} to b_{16}	b_{31} to b_{24}	b_{39} to b_{32}	b_{47} to b_{40}	b_{55} to b_{48}	b_{63} to b_{56}

Transmission Telegram

NOTE

When using the listed Mapped Objects instead of the complete transmission telegram, the refresh rate of the individual messages may be reduced.

Data Protocol Parameter No.3190/Object 2C76h

In this protocol the LeoPC display messages were sent:

Parameter 3190, Object 2C76h							
MUX	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**	**Byte 8**
0	Parameter No. 3190		Parameter No. 108				-Internal-
1	Parameter No. 144		Parameter No. 114				-Internal-
2	Parameter No. 147		Parameter No. 109				-Internal-
3	Parameter No. 160		Parameter No. 115				-Internal-
4	Parameter No. 10166		Parameter No. 110				-Internal-
5	Parameter No. 10167		Parameter No. 116				-Internal-
6	Parameter No. 10110		Parameter No. 118				-Internal-
7	Parameter No. 10168		Parameter No. 121				-Internal-
8	Parameter No. 10169		Parameter No. 119				-Internal-
9	Parameter No. 10106		Parameter No. 122				-Internal-
10	Parameter No. 10107		Parameter No. 120				-Internal-
11	Parameter No. 10201		Parameter No. 123				-Internal-
12	---		Parameter No. 111				-Internal-
13	---		Parameter No. 112				-Internal-
14	Parameter No. 10133		Parameter No. 113				-Internal-
15	Parameter No. 10134		---				-Internal-
16	Parameter No. 10135		Parameter No. 136				-Internal-
17	---		Parameter No. 135				-Internal-
18	Parameter No. 10141		---				-Internal-
19	Parameter No. 10306		---				-Internal-
20	Parameter No. 10302		Parameter No. 10303				-Internal-
21	Parameter No. 10138		Parameter No. 2520				-Internal-
22	Parameter No. 10140		Parameter No. 2522				-Internal-
23	Parameter No. 10202		---				-Internal-
24	Parameter No. 10307		Parameter No. 10308				-Internal-
25	Parameter No. 8003		Parameter No. 8013		Parameter No. 8003		-Internal-
26	Parameter No. 10328		Parameter No. 10329		Parameter No. 10330		-Internal-
27	Parameter No. 10331		Parameter No. 10332		Parameter No. 10333		-Internal-
28	Parameter No. 10334		Parameter No. 10165		Parameter No. 10336		-Internal-
29	Parameter No. 10337		---		---		-Internal-

Data Protocol Parameter No. 15603/Object 5CF3h – Source 1 Values

If the object 5CF3h is read out, the protocol known value is replaced

Parameter No.15603,Object 5CF3h							
MUX	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**	**Byte 8**
0	Parameter No. 15603		---		---		-Internal-
1	Parameter No. 118				Parameter No. 147		-Internal-
2	Parameter No. 119				Parameter No. 10166		-Internal-
3	Parameter No. 120				Parameter No. 10167		-Internal-
4	Parameter No. 121				Parameter No. 10110		-Internal-
5	Parameter No. 122				Parameter No. 10168		-Internal-
6	Parameter No. 123				Parameter No. 10169		-Internal-
7	Parameter No. 123				Parameter No. 2862		-Internal-
8	Parameter No. 2510				---		-Internal-
9	Parameter No. 2522				---		-Internal-

Data Protocol Parameter No. 15604/Object 5CF4h – Source 2 Values

If the object 5CF4h is read out, the protocol known value is replaced

Parameter No.15604,Object 5CF4h							
MUX	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**	**Byte 8**
0	Parameter No. 15604		---		---		-Internal-
1	Parameter No. 108				Parameter No. 144		-Internal-
2	Parameter No. 109				Parameter No. 160		-Internal-
3	Parameter No. 110				Parameter No. 10134		-Internal-
4	Parameter No. 111				Parameter No. 10166		-Internal-
5	Parameter No. 112				Parameter No. 10167		-Internal-
6	Parameter No. 113				Parameter No. 10132		-Internal-
7	Parameter No. 114				Parameter No. 10133		-Internal-
8	Parameter No. 115				Parameter No. 10141		-Internal-
9	Parameter No. 116				Parameter No. 10168		-Internal-
10	Parameter No. 135				Parameter No. 10169		-Internal-
11	Parameter No. 136				Parameter No. 10306		-Internal-

CANopen: Mapping Parameter

Parameter no.	Object-ID	Name	Unit	Data type	Note
108	206Ch	Source 2: Voltage V_{L12}	1/10 V	signed32	
109	206Dh	Source 2: Voltage V_{L23}	1/10 V	signed32	
110	206Eh	Source 2: Voltage V_{L31}	1/10 V	signed32	
111	206Fh	Source 2: Current I_{L1}	mA	signed32	
112	2070h	Source 2: Current I_{L2}	mA	signed32	
113	2071h	Source 2: Current I_{L3}	mA	signed32	
114	2072h	Source 2: Voltage V_{L1N}	1/10 V	signed32	
115	2073h	Source 2: Voltage V_{L2N}	1/10 V	signed32	
116	2074h	Source 2: Voltage V_{L3N}	1/10 V	signed32	
118	2076h	Source 1: Voltage V_{L12}	1/10 V	signed32	
119	2077h	Source 1: Voltage V_{L23}	1/10 V	signed32	
120	2078h	Source 1: Voltage V_{L31}	1/10 V	signed32	
121	2079h	Source 1: Voltage V_{L1N}	1/10 V	signed32	
122	207Ah	Source 1: Voltage V_{L2N}	1/10 V	signed32	
123	207Bh	Source 1: Voltage V_{L3N}	1/10 V	signed32	
135	2087h	Source 2: Real power P	W	signed32	
136	2088h	Source 2: Reactive power Q	var	signed32	
144	2090h	Source 2: Frequency	1/100 Hz	signed16	
147	2093h	Source 1: Frequency f_{123}	1/100 Hz	signed16	
160	20A0h	Source 2: Power factor $\cos\varphi_{L1}$	1/1000, dimls.	signed16	
2520	29D8h	Source 2: Real energy	1/100 MWh	unsigned32	
2522	29DAh	Source 2: Reactive energy	1/100 Mvarh	unsigned32	
8000	3F40h	always 0		unsigned16	
8001	3F41h	Output of the 1st IKD1	Bit field	unsigned16	
		Bit 15 Relay output [REx08]			
		Bit 14 Relay output [REx07]			
		Bit 13 Relay output [REx06]			
		Bit 12 Relay output [REx05]			
		Bit 11 Relay output [REx04]			
		Bit 10 Relay output [REx03]			
		Bit 9 Relay output [REx02]			
		Bit 8 Relay output [REx01]			
		Bit 7 always 0			
		Bit 6 always 0			
		Bit 5 always 0			
		Bit 4 always 0			
		Bit 3 always 0			
		Bit 2 always 0			
		Bit 1 always 0			
		Bit 0 always 1			

Parameter no.	Object-ID	Name		Unit	Data type	Note
8002	3F42h	Outputs of the 2nd IKD1		Bit field	unsigned16	
		Bit 15	Relay output [REx16]			
		Bit 14	Relay output [REx15]			
		Bit 13	Relay output [REx14]			
		Bit 12	Relay output [REx13]			
		Bit 11	Relay output [REx12]			
		Bit 10	Relay output [REx11]			
		Bit 9	Relay output [REx10]			
		Bit 8	Relay output [REx09]			
		Bit 7	always 0			
		Bit 6	always 0			
		Bit 5	always 0			
		Bit 4	always 0			
		Bit 3	always 0			
		Bit 2	always 0			
		Bit 1	always 0			
		Bit 0	always 1			
8003	3F43h	External relay outputs, status		Bit field	unsigned16	
		Bit 15	Relay output [REx16]			
		Bit 14	Relay output [REx15]			
		Bit 13	Relay output [REx14]			
		Bit 12	Relay output [REx13]			
		Bit 11	Relay output [REx12]			
		Bit 10	Relay output [REx11]			
		Bit 9	Relay output [REx10]			
		Bit 8	Relay output [REx09]			
		Bit 7	Relay output [REx08]			
		Bit 6	Relay output [REx07]			
		Bit 5	Relay output [REx06]			
		Bit 4	Relay output [REx05]			
		Bit 3	Relay output [REx04]			
		Bit 2	Relay output [REx03]			
		Bit 1	Relay output [REx02]			
		Bit 0	Relay output [REx01]			
8013	3F43h	External discrete inputs, status		Bit field	unsigned16	
		Bit 15	Discrete input [DEx16]			
		Bit 14	Discrete input [DEx15]			
		Bit 13	Discrete input [DEx14]			
		Bit 12	Discrete input [DEx13]			
		Bit 11	Discrete input [DEx12]			
		Bit 10	Discrete input [DEx11]			
		Bit 9	Discrete input [DEx10]			
		Bit 8	Discrete input [DEx09]			
		Bit 7	Discrete input [DEx08]			
		Bit 6	Discrete input [DEx07]			
		Bit 5	Discrete input [DEx06]			
		Bit 4	Discrete input [DEx05]			
		Bit 3	Discrete input [DEx04]			
		Bit 2	Discrete input [DEx03]			
		Bit 1	Discrete input [DEx02]			
		Bit 0	Discrete input [DEx01]			

Parameter no.	Object-ID	Name		Unit	Data type	Note
10106	---	Discrete inputs, status		Bit field	unsigned16	
		Bit 15	Discrete input [D1]			
		Bit 14	Discrete input [D2]			
		Bit 13	Discrete input [D3]			
		Bit 12	Discrete input [D4]			
		Bit 11	Discrete input [D5]			
		Bit 10	Discrete input [D6]			
		Bit 9	Discrete input [D7]			
		Bit 8	Discrete input [D8]			
		Bit 7	Discrete input [D9]			
		Bit 6	Discrete input [D10]			
		Bit 5	Discrete input [D11]			
		Bit 4	Discrete input [D12]			
		Bit 3	-Internal-			
		Bit 2	-Internal-			
		Bit 1	-Internal-			
		Bit 0	-Internal-			
10107	---	Relay outputs, status		Bit field	unsigned16	
		Bit 15	Relay output [R01]			
		Bit 14	Relay output [R02]			
		Bit 13	Relay output [R03]			
		Bit 12	Relay output [R04]			
		Bit 11	Relay output [R05]			
		Bit 10	Relay output [R06]			
		Bit 9	Relay output [R07]			
		Bit 8	Relay output [R08]			
		Bit 7	Relay output [R09]			
		Bit 6	-Internal-			
		Bit 5	-Internal-			
		Bit 4	-Internal-			
		Bit 3	-Internal-			
		Bit 2	-Internal-			
		Bit 1	-Internal-			
		Bit 0	-Internal-			
10110	477Eh	Battery voltage		1/10 V	unsigned16	
10134	4796h	Generator, watchdog 1		Bit field	unsigned16	
		Bit 15	-Internal-			
		Bit 14	-Internal-			
		Bit 13	-Internal-			
		Bit 12	-Internal-			
		Bit 11	-Internal-			
		Bit 10	-Internal-			
		Bit 9	-Internal-			
		Bit 8	-Internal-			
		Bit 7	Load, overcurrent, limit 1			Time-overcurrent
		Bit 6	Load, overcurrent, limit 2			Time-overcurrent
		Bit 5	Load, overcurrent, limit 3			Time-overcurrent
		Bit 4	-Internal-			Rev/red load
		Bit 3	-Internal-			Rev/red load
		Bit 2	Load, overload, limit 1			
		Bit 1	Load, overload, limit 2			
		Bit 0	-Internal-			

Parameter no.	Object-ID	Name		Unit	Data type	Note
10136	4798h	Latched alarm bits analog input		Bit field	unsigned16	
		Bit 15	-Internal-			
		Bit 14	-Internal-			
		Bit 13	-Internal-			
		Bit 12	-Internal-			
		Bit 11	-Internal-			
		Bit 10	-Internal-			
		Bit 9	-Internal-			
		Bit 8	-Internal-			
		Bit 7	-Internal-			
		Bit 6	-Internal-			
		Bit 5	-Internal-			
		Bit 4	-Internal-			
		Bit 3	Alarm bit monitoring battery voltage overvoltage threshold 2			
		Bit 2	Alarm bit monitoring battery voltage undervoltage threshold 2			
		Bit 1	Alarm bit monitoring battery voltage overvoltage threshold 1			
		Bit 0	Alarm bit monitoring battery voltage undervoltage threshold 1			
10140	---	Flag of the *LogicsManager*		Bit field	unsigned16	
		Bit 15	Flag 1 is TRUE			
		Bit 14	Flag 2 is TRUE			
		Bit 13	Flag 3 is TRUE			
		Bit 12	Flag 4 is TRUE			
		Bit 11	Flag 5 is TRUE			
		Bit 10	Flag 6 is TRUE			
		Bit 9	Flag 7 is TRUE			
		Bit 8	Flag 8 is TRUE			
		Bit 7	-Internal-			
		Bit 6	-Internal-			
		Bit 5	-Internal-			
		Bit 4	-Internal-			
		Bit 3	-Internal-			
		Bit 2	-Internal-			
		Bit 1	-Internal-			
		Bit 0	-Internal-			

Parameter no.	Object-ID	Name		Unit	Data type	Note
10146	47A2h	Internal flags of the *LogicsManager*		Bit field	unsigned16	
		Bit 15	-Internal-			
		Bit 14	-Internal-			
		Bit 13	Horn output			
		Bit 12	-Internal-			
		Bit 11	-Internal-			
		Bit 10	-Internal-			
		Bit 9	Daily time set point 1 exceeded			
		Bit 8	Daily time set point 2 exceeded			
		Bit 7	Actual weekday is in group of active weekdays			
		Bit 6	Actual day is active day			
		Bit 5	Actual hour is active hour			
		Bit 4	Actual minute is active minute			
		Bit 3	Actual second is active second			
		Bit 2	-Internal-			
		Bit 1	-Internal-			
		Bit 0	-Internal-			
10302	---	Source 2: real power P		1/10 kW	unsigned16	These variables are necessary to ensure downward compatibility with LeoPC1 V2.1.xxx.
10303	---	Source 2: reactive power Q		1/10 kvar	unsigned16	
10306	---	Source 2: power factor cosphi		cos1=100	unsigned16	
10307	---	External discrete inputs with alarm class		Bit filed	unsigned16	
		Bit 15	Discrete input [DEx16]			
		Bit 14	Discrete input [DEx15]			
		Bit 13	Discrete input [DEx14]			
		Bit 12	Discrete input [DEx13]			
		Bit 11	Discrete input [DEx12]			
		Bit 10	Discrete input [DEx11]			
		Bit 9	Discrete input [DEx10]			
		Bit 8	Discrete input [DEx09]			
		Bit 7	Discrete input [DEx08]			
		Bit 6	Discrete input [DEx07]			
		Bit 5	Discrete input [DEx06]			
		Bit 4	Discrete input [DEx05]			
		Bit 3	Discrete input [DEx04]			
		Bit 2	Discrete input [DEx03]			
		Bit 1	Discrete input [DEx02]			
		Bit 0	Discrete input [DEx01]			
15603	5CF3	Source 1 values		---	unsigned 64	Data Protocol
15604	5CF4	Source 2 values		---	unsigned 64	Data Protocol

Appendix C.
Application Examples

Remote Control

The DTSC-200 controller may be configured to perform acknowledgement functions remotely through the CAN bus. The required procedure is detailed in the following steps.

NOTE

Refer to the operation manual 37387 for a detailed description of the navigation through the various display screens. A detailed description of the individual parameters may be found in the configuration manual 37386.

Be sure to enter the password for code level 2 or higher to be able to access the required configuration screens.

The DTSC may be acknowledged with CAN/Modbus. Therefore, a logical command variable has to be configured with the *LogicsManager*:

04.14 Remote acknowledge

Configuration of the *LogicsManager* Functions

Open the main menu by pressing the softkey and navigate to "Configure monitoring" screen by using the softkey. Open the "Configure monitoring" menu by using the softkey. Navigate to "External acknowledge" by using the softkey and enter the "External acknowledge" *LogicsManager* screen by pressing the softkey.

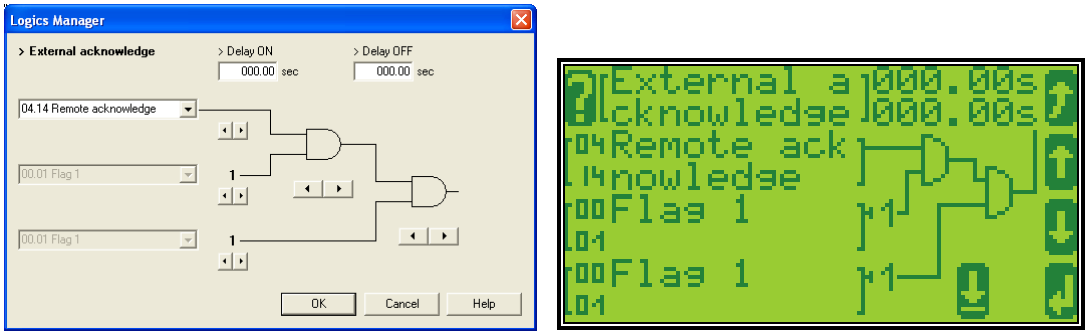


Figure 5-7: Display screen - Ext. acknowledge

Configure the respective values for the "External acknowledge" *LogicsManager* function using the and as well as the softkey and Confirm the change by pressing the softkey:

With this setting, the "External acknowledge" *LogicsManager* output becomes TRUE as soon as the remote acknowledge signal is enabled.

NOTE

The *LogicsManager* commands 2 and 3 may be used to configure additional conditions like discrete inputs, which must be energized to be able to issue the remote command.

Remote Control Telegram

The internal parameter 503 of the DTSC must be set to react on the remote control instructions. This is performed by sending rising signals for the respective bits.

Refer to the Remote Control Telegram section on page 57 for detailed information about the telegram structure and the control bits.

Ext. Acknowledge: The command variable "04.14 Remote acknowledge" is the reflection of the control bit (bit 4). The DTSC deactivates the horn with the first change from "0" to "1" of the logical output "External acknowledge", and acknowledges inactive alarm messages with the second change from "0" to "1".

Remote Control via CAN

It is possible to perform a remote acknowledgement via a default SDO communication channel.

Remote Acknowledgement

Configuration of CAN Interface

Be sure to enable CAN-Open Master if there is no PLC taking over the master function.

Open the main menu by pressing the softkey and navigate to "Set up Comm interfaces" by using the softkey. Open the "Set up Comm interfaces" menu by using the softkey and navigate to "Set up CAN interfaces" by using the softkey. Open the "Set up CAN interfaces" menu by using the softkey and navigate to "CAN Open interface" by using the softkey. Open the "CAN Open interface" menu by using the softkey, navigate to "CAN-Open Master" by using the softkey and enter the "CAN-Open Master" screen by pressing the softkey.

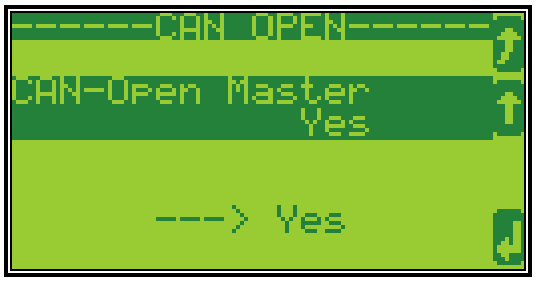


Figure 5-8: Display screen - configure CAN interface

Select "Yes" by using the softkey and confirm your selection by pressing the softkey.

General Information

The device listens to the CAN ID 600 (hex) + Node ID internally to perform the desired control, the reply is on CAN ID 580 (hex) + Node ID.

The following examples show the request format on CANopen with different Node IDs.

The request on the bus is sent via the control parameter 503 of the device.
The hexadecimal value 2000 is calculated internally.
503(decimal) -- 1F7 (hexadecimal)
1F7+2000 (hexadecimal) = 21F7

Please note that high and low byte are exchanged in the sent address.
The data (hex) shows the state of parameter 503 to achieve the required control.

Node ID 1 standard

Figure 5-9 shows exemplary request data for the device on the CANopen bus.

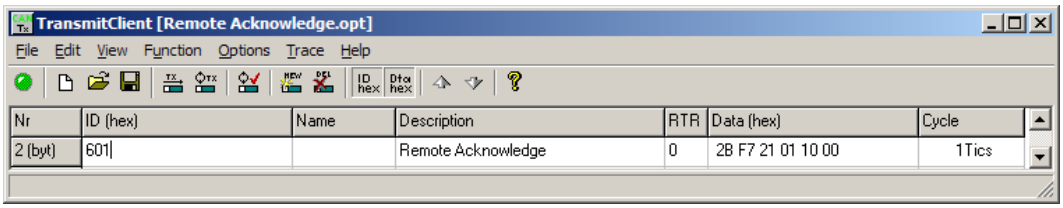


Figure 5-9: CANopen request data for Node ID 1

Node ID (not standard value)

If the Node ID of the device is intended to be different from the standard value, the "Device number" parameter must be configured accordingly. Node ID 2 is used in the following example.

Press ↩ until you return to the start screen.

Open the main menu by pressing the ↑ softkey and navigate to "Set up Comm interfaces" by using the ↓ softkey. Open the "Set up Comm interfaces" menu by using the ↵ softkey and navigate to "Device number" by using the ↓ softkey and enter the "Device number" screen by pressing the ↵ softkey.

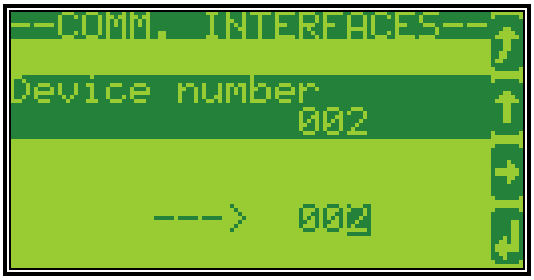


Figure 5-10: Display screen - configure device number

Configure "002" by using the → and ↑ softkeys and confirm your selection by pressing the ↵ softkey.

With this setting, the Node ID of the CAN interface is set to 002.

The request on the bus is sent via the control parameter 503 of the device.
The hexadecimal value 2000 is calculated internally.
503(decimal) -- 1F7 (hexadecimal)
1F7+2000 (hexadecimal) = 21F7
Please note that high and low byte are exchanged in the sent address.
The data (hex) shows the state of parameter 503 to achieve the required control.

Figure 5-11 shows exemplary request data for the device on the CANopen bus.

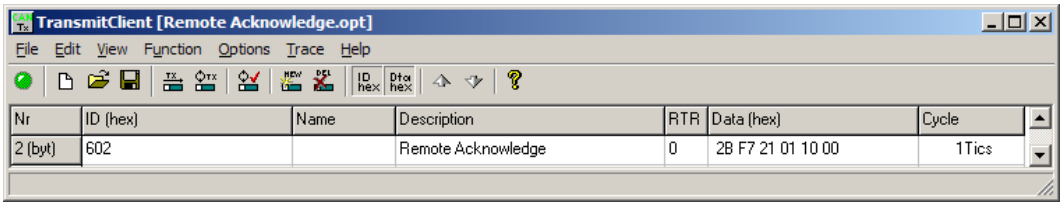


Figure 5-11: CANopen request data for Node ID 2

Additional SDO Communication Channels

It is also possible to allow several PLCs to acknowledge the unit in addition to the default SDO communication channel. Four additional SDO communication channels are provided for this. The additional SDO 127 (decimal) or 7F (hex) is used in the following example.

Press until you return to the start screen.

Open the main menu by pressing the softkey and navigate to "Set up Comm interfaces" by using the softkey. Open the "Set up Comm interfaces" menu by using the softkey and navigate to "Set up CAN interfaces" by using the softkey. Open the "Set up CAN interfaces" menu by using the softkey and navigate to "CAN Open interface" by using the softkey. Open the "CAN Open interfaces" menu by using the softkey and navigate to "Additional Server SDOs" by using the softkey. Enter the "Additional S-SDO" screen by pressing the softkey.

Navigate to "2nd Client->Server COB-ID (rx)" by using the softkey and press the softkey to edit this parameter. Configure "0000067F" by using the and softkeys and confirm your entry by pressing the softkey.

Navigate to "2nd Server->Client COB-ID (tx)" by using the softkey and press the softkey to edit this parameter. Configure "000005FF" by using the and softkeys and confirm your entry by pressing the softkey.

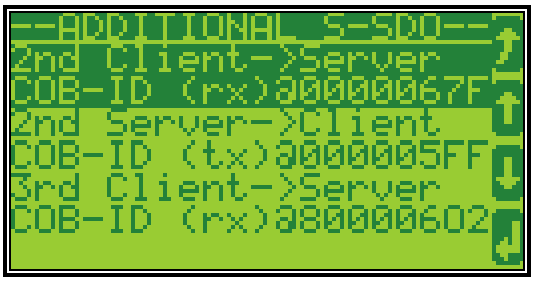


Figure 5-12: Display screen - configure Server SDOs

NOTE

Be sure to remove the leading 8 from the COB-IDs to enable them. For example, change the standard value of "2nd Client-Server COB-ID (rx)", which is "80000601", to "0000067F".

In this example, an additional SDO communication channel is configured to 127 (decimal) or 7F (hex).

The control request is equal to the request via default SDO communication channel, but the device will listen to messages including the configured address as well.

The device listens to the CAN ID 600 (hex) + Node ID internally to perform the desired control, the reply from the DTSC is sent on CAN ID 580 (hex) + Node ID.
Receive CAN ID 67F (hex) (600 (hex) + 7F (hex))
Transmit CAN ID 5FF (hex) (580 (hex) + 7F (hex))

The same is valid for the additional SDO communication channels 3, 4, and 5.
Figure 5-13 shows exemplary request data for the device on the CANopen bus.

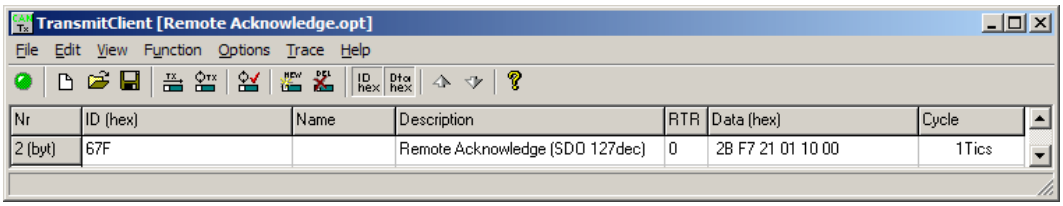


Figure 5-13: CANopen request data for additional Server SDO

NOTE

If parameters are written or read via two or more SDO communication channels at the same time (before the first has answered), the second one will be refused.

Remote Control via Modbus

The DTSC controller may be configured to perform acknowledgement functions remotely through the Modbus. The required procedure is detailed in the following steps.

NOTE

The following descriptions refer to the remote control parameter 503 as described under Remote Control Telegram on page 77.

It may be necessary to shift the address by 1 depending on the used PC software. In this case, the address would be 504 for example.

Be sure to check both possibilities in case of remote control problems.

Par. ID.	Parameter	Setting range	Data type
503	Remote control word	0 to 65535	UNSIGNED 16

Modbus address = 40000 + (Par. ID +1) = 504
Modbus length = 1 (UNSIGNED 16)

The following Modscan32 screenshot shows the configurations made to remote control parameter 503. It is possible to set the format to binary to view single the bits using the "display options".

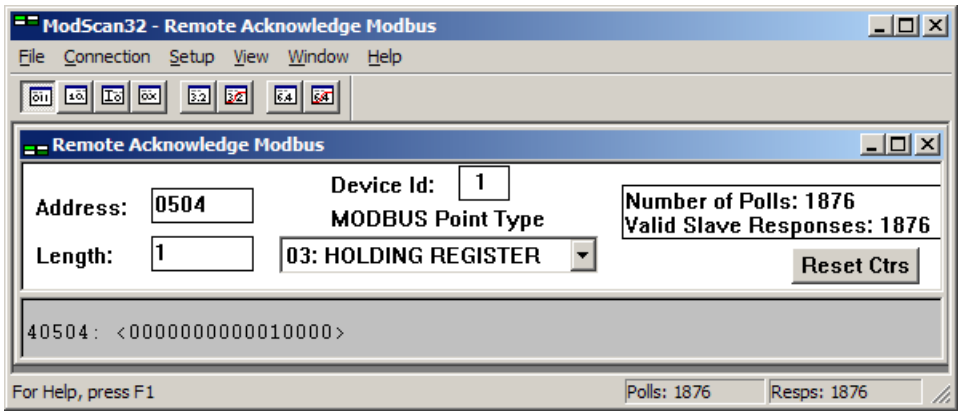


Figure 5-14: Modbus - remote control parameter 503

By double-clicking the address, a Write Register command may be issued. Figure 5-15 shows how bit 4 is set using the ModScan32 Software.

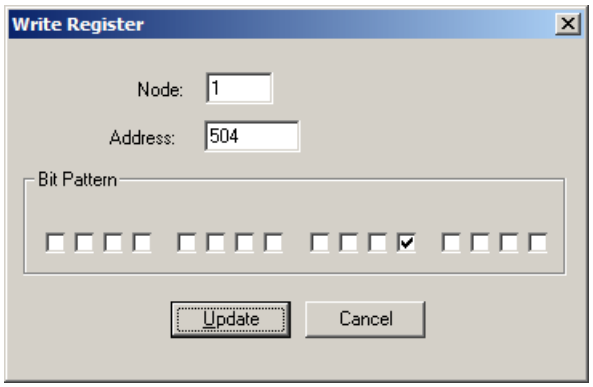


Figure 5-15: Modbus - write register

NOTE

Be sure to enter the password for code level 2 or higher for the corresponding interface to get access for changing parameter settings.

Sending a Data Protocol via TPDO

Cyclically Sending of Data

This is a configuration example for sending an object with the index 3190 (data protocol 4007) on CAN ID 2AEh every 20 ms on TPDO1. For this, TPDO1 must be configured as follows:

COB-ID	2AE (hex)	
Transmission type	255	
Event-timer	20 ms	
Number of Mapped Objects	1	(there is only one object to be transmitted)
1. Mapped Object	3190	(display value, the object with the index 3190)
2. Mapped Object	0	(will not be used)
3. Mapped Object	0	(will not be used)
4. Mapped Object	0	(will not be used)

Sending of Data on Request

The data to be sent (Mapped Objects) may be provided on request by configuring the Sync Message and the Transmission Type of a TPDO.
The unit is requested to send its data by sending a Sync Message.

The number of required Sync Messages is determined by the setting of the Transmission Type.

If the data is to be sent on request, Bit 31 of the Sync Message must be configured to "1" and the CANopen Master function must be configured to "Off".

The Transmission Type of TPDO 1 is configured to "2" in the following example.
This means that a message of the configured TPDO is sent by the unit after two Sync Messages have been sent to the unit.

The recorded data shows that the data of the Mapped Object (in this example Mux 5) is sent (refer to Figure 5-17) after sending the Sync Message twice (refer to Figure 5-16).

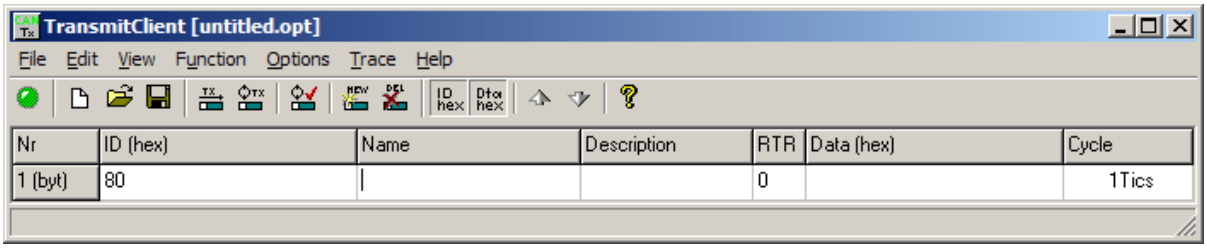


Figure 5-16: Cyclical sending of data - Sync Message request

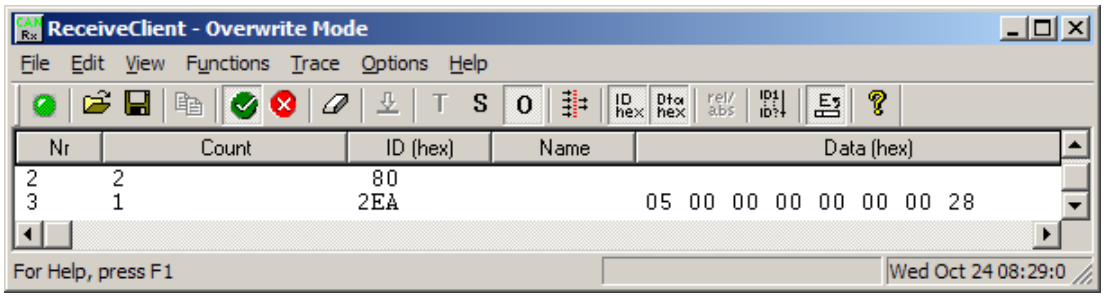


Figure 5-17: Cyclical sending of data - reply

We appreciate your comments about the content of our publications.

Please send comments to: stgt-documentation@woodward.com

Please include the manual number from the front cover of this publication.

Woodward GmbH
Handwerkstrasse 29 - 70565 Stuttgart - Germany
Phone +49 (0) 711-789 54-0 • Fax +49 (0) 711-789 54-100
stgt-info@woodward.com

Homepage

http://www.woodward.com/power

Woodward has company-owned plants, subsidiaries, and branches, as well as authorized distributors and other authorized service and sales facilities throughout the world.

Complete address/phone/fax/e-mail information for all locations is available on our website (www.woodward.com).

2007/12/Stuttgart