AMK

AMKASYN

VARIABLE SPEED DRIVES

AMKASYN

Digital drive systems

Option card AE-PSC

- **Programmable control PS**
- **CAN Interface (CAN-S)**

Option card for AZ/AW systems in the version AZ-CNS
Option card for KU systems in the version KU-PSC

CAN Network configuration

Important notes

Due to possible destruction of components by static discharge, touching the electrical connections on the option card should be avoided.

Please attach option card directly from the packaging in the option slot of the KU or AZ module without exerting force and secure with the screw on the front panel.

AMK Arnold Müller, Antriebs- und Regeltechnik GmbH & Co.KG, D-73221 Kirchheim/Teck,
Tel.: 07021/50 05-0, Telefax: 07021/50 05-176

Document overview for option card AE-PSC

The following documentation is available:

1. Product information (AMK part No.:28681)

- What is the AE-PSC?
- By what is the CAN interface of AMK characterized?
- What advantages does the CAN bus have?
- How does CAN work?
- What does integrated PS functionality mean?
- Properties and application examples of the AE-PSC
- How is the CAN network configured?
- Principles for CANopen

2. Hardware description (AMK part No.:28621)

- Short description
- Installation instructions of the option card
- Important notes on handling
- Front panel and board structure
- Interfaces and pin assignment

3. CAN network configuration (AMK part No.:26684)

- Parameter settings in the AMKASYN system
- CANopen principles
- Network configuration with CANconv program
- Example of a configuration
- AMK Tool CANconv
- AMK Tool DVLader

Contents

List of Figures

1 Abbreviations and explanations

AE-PSC	**A**MKASYN **E**xtension **PS C**AN
APROS	AMK PS programming software
Arbitration	Bus access method; method with which access to the bus is regulated. Solution of the conflict if several stations want to send a message at the same time
AZ/AW system	AMKASYN modular drive system, consisting of central module and inverter modules
AZ-CNS	AMKASYN option card for AZ modules
Broadcasting	describes the possibility of addressing all subscribers to the network simultaneously
CAN	**C**ontroller **A**rea **N**etwork
CANconv	AMK Can converter auxiliary program for transferring the CAN network configuration to the master
ccb	**C**AN **c**onfiguration **b**inary file type *.ccb
ccf	**C**AN **c**onfiguration **f**ile *.ccf
CiA	**C**AN **i**n **A**utomation, international users and manufacturers group e.V.
DVLader	AMK auxiliary tool for flash database access
Emergency Service	Bus fault characteristic on failure of one or several subscribers.
Telegram header	Header information of a message (e.g. priority...)
Ident number	(ID No.) Parameter for parameterizing the AMKASYN system
NMT service	Network management service (network initialization, bus error monitoring, status monitoring of the individual devices)
Node Guarding	Network node monitoring, is performed by the NMT master
Parameter	(ID No.) by which the AMKASYN systems are parameterized
KU	AMKASYN digital compact converter
KU-PSC	AMKASYN option card for KU system
Life Guarding	NMT slave monitors whether the network node monitoring of the NMT master is performed.
PDO	**P**rocess **D**ata **O**bject
PS	**P**rogrammable control
R-PDO	**R**eceive **PDO**
SDO	**S**ervice **D**ata **O**bject
T-PDO	**T**ransmit **PDO**

2 Parameter settings

Overview:

System / module	Cycle time	Transmission rate	Addressing	Selected function (CAN or PS)
KU	ID2	ID34024	Switch S1, S2 or ID34023	ID32799
AZ	ID2	-	-	-
AZ-CNS	ccf → ID2	ccf → ID34024	Switch S1, S2	On AZ-CNS only CAN master in association with AZ-PS5 implemented
	↑ equal values	↑ equal values	↑ unequal values	

For the synchronization of a CAN master with the slaves, all drives must comply with the same time conditions. This is guaranteed if in all subscribers the Ident number ID2 SERCOS cycle time and ID34024 bus transmission rate are occupied with the same values. This is agreed in the AZ-CNS in the "CAN Configuration File" (ccf file).

Parameters which are set application and system dependently are located in the **Parameter KU/AZ** documentation.

2.1 Parameters in the KU

The following parameters must be set correspondingly for operating the KU-PSC.

2.1.1 ID 32799 "Configuration standard periphery"

This parameter defines:

- Square pulse encoder input signals X34
- Activating / deactivating PS function
- Activating / deactivating field bus function

ID 32799=00**ab**00**cc hex**

cc – Setting code for square pulse encoder input (X34)

Entry	Meaning
0	2 square pulses offset by 90 degrees
1	Counting pulses track 1, direction signal track 2
2	Forwards pulses track 1, backwards pulses track 2
3...FF	reserved

b - PS function

Entry	Meaning
0	Function deactive (default) With plugged in KU-PSC error message **1376** is output, indication for user to activate / deactivate the required function
1	PS active
2...E	reserved
F	Function deactive, no error message

a – Field bus function

Entry	Meaning
0	Field bus deactive (default) With plugged in KU-PSC error message **1376** is output, indication for user to activate / deactivate the required function
1	Field bus active (e.g. CAN, ...)
2...E	reserved
F	Field bus deactive, no error message

Example:

ID32799 = 0x 00 11 00 00

- 2 square pulses offset by 90 degrees
- Field bus active
- PS active

2.1.2 Communication parameters

ID 34023 "BUS subscriber address"

The slave address is entered in this Ident number. An entered subscriber address is valid only if the hexadecimal switches S1 and S2 on the KU-PSC have the code 0. If the code is not equal to zero, then the Ident number ID34023 is written with the address set at S1 and S2. Allowed subscriber addresses are the values (01h to 7Fh) 1 to 127. The subscriber address 1 is reserved for the master when an AZ-CNS card is used in the AZ/AW system.

ID 34024 "BUS transmission rate"

The baud rate must be set in the range from 10 KBaud to 1 MBaud in this parameter. Note here that **the same baud rate** must be set for **all subscribers to the bus**.

Permissible values:

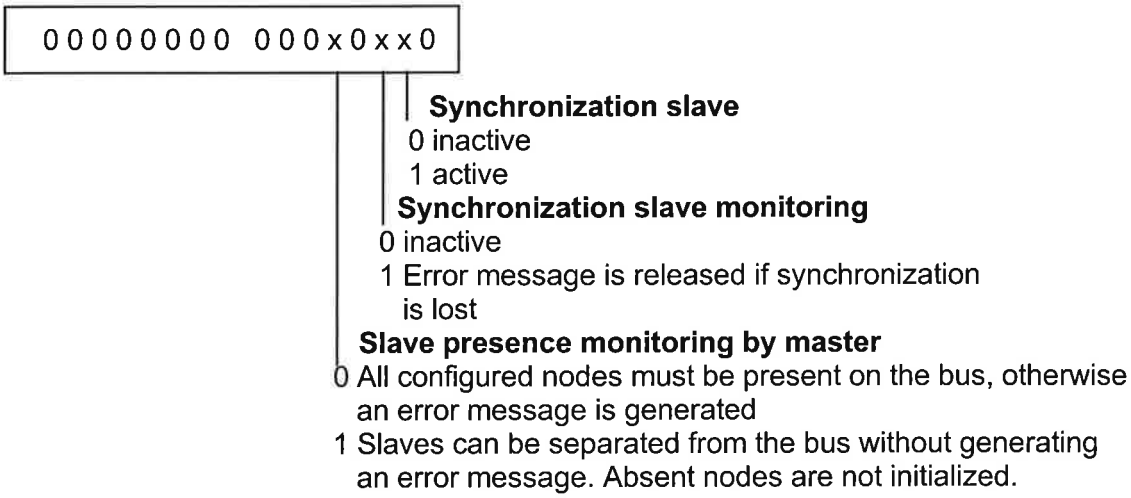
1000.00	1Mbaud;
800.00	800kbaud;
500.00	500kbaud;
250.00	250kbaud;
125.00	125kbaud;
50.00	50kbaud;
20.00	20kbaud;
10.00	10kbaud;

With invalid value entry, the default value of 20kbaud is entered automatically.

ID34026 "Bus mode attribute"

The parameter defines the differentiated features of the CAN bus.

ID34026 "Bus mode attribute"



Reserved bits are preassigned with 0

Example:

ID34026 = 0006 hex, hardware synchronization and message on error

2.2 Parameters in the AZ

All settings in the AZ module for the use of the AZ-CNS option card must be made through the "CAN Configuration File" (*.ccf file). The CAN network configuration is described in the following sections.

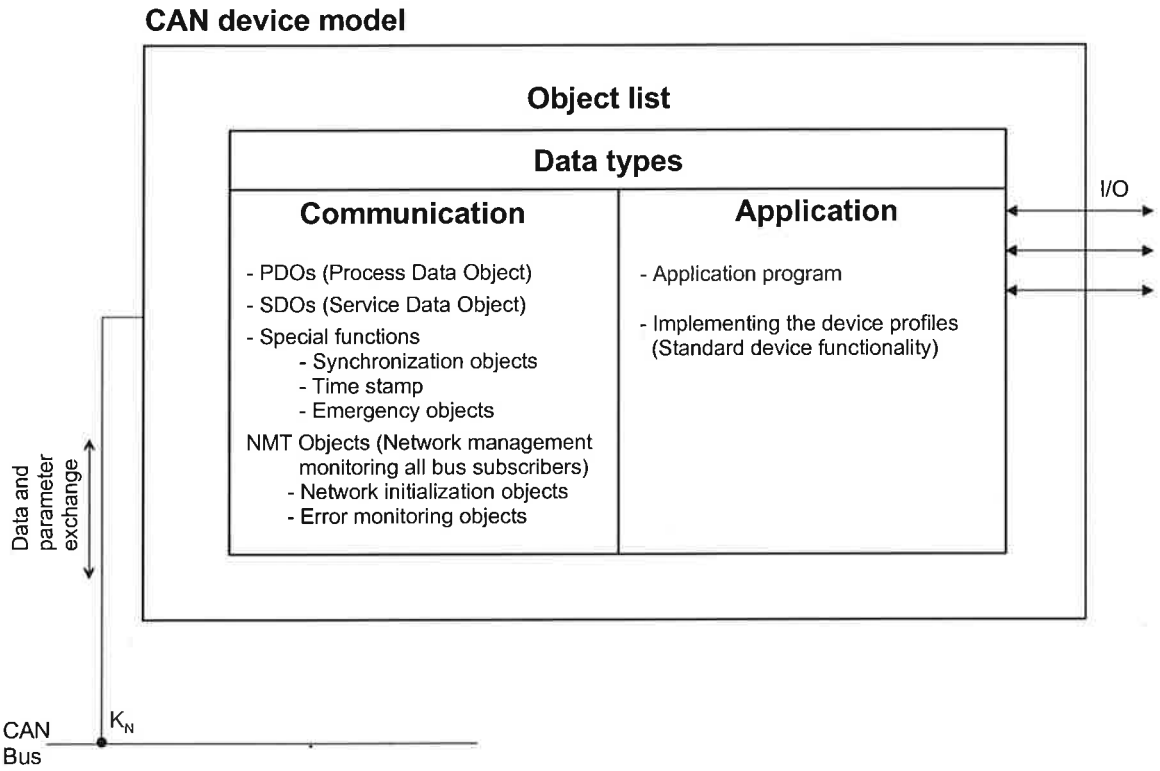
3 Principles of CANopen

3.1 Object list

Each CANopen device has a CANopen object list. The object list is divided into different areas. There are areas for the description of the data types, of the communication and of the application. All data which can be exchanged through the CAN network are represented by corresponding objects in the object list. Access to entries of the object list is made through a 16-bit index and an 8-bit subindex.

The object list is the data and parameter interface of the node to the CAN network. It describes the device with regard to its application and communication properties.

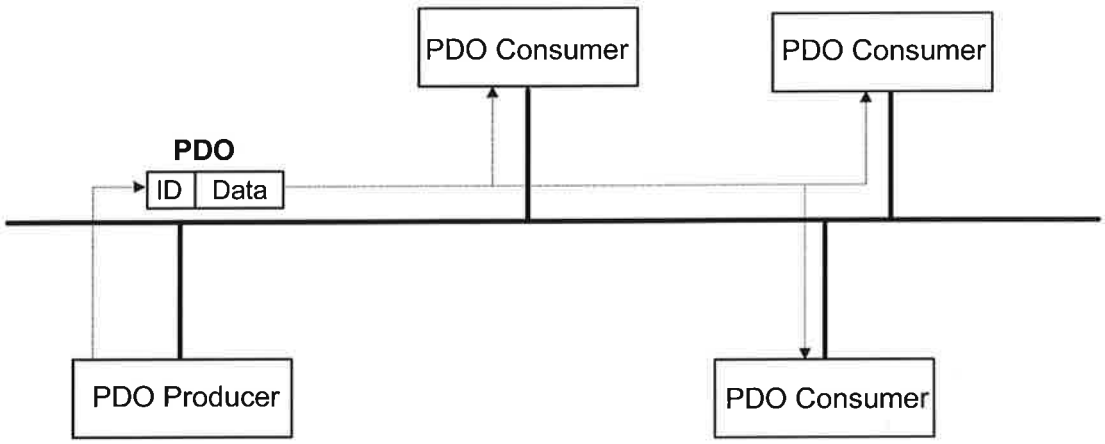
Figure 3-1 CANopen communication model



3.2 Real time communication

So-called PDOs (Process Data Objects) are used for exchanging real time data. PDO communication can be described with the producer / consumer model. The process data are transmitted by a node (producer) and received by one or several nodes (consumers). PDOs are not confirmed by the receiver. In PDOs all maximum 8 data bytes of a CAN frame are available for the data exchange.

Figure 3-2 Process Data Object PDO



3.3 Communication profile

The communication profile is the part of the object list which determines the communication properties of a node (see object list figure). Therefore the communication profile of the object list contains entries which describe the properties of PDOs.

3.4 PDOs in the communication profile

A PDO to be transmitted (T-PDO) is described in the communication profile of the object list with the "Communication parameters" and the "Mapping parameters".

3.4.1 Communication parameters

COB-ID

- Communication object identifier
- 11-bit identifier at the beginning of the message
- Lower value of the identifier = high priority of the PDO
- A data transmission between 2 nodes takes place if the COB-ID of a T-PDO of a node agrees with the COB-ID of a R-PDO of another node.

Transmission type
- States when data are updated and transmitted or received
- Examples:

Type 0 Acyclic synchronous
Type 1-240 Cyclic synchronous (transmitting and updating with every SYNC message)
Type 253 Asynchronous RTR only
Type 254 AMK asynchronous device specific (transmitting and updating on data change in the mapped objects)

3.4.2 PDO Mapping Parameters

- List of objects which are contents of a PDO
- The object is stated with index / subindex / length
- T-PDO: which objects (index/subindex) are copied in a PDO before transmitting
- R-PDO: into which objects (index/subindex) are the PDO contents copied after reception

The index for the PDOs results from the table "Communication Profile Area" in the appendix of this document.

Transmit PDO Communication Parameter	Index 0x1800 ff.
Transmit PDO Mapping Parameter	Index 0x1A00 ff.

A R-PDO is described with the following parameters in the communication profile of the object list:

Receive PDO Communication Parameter	Index 0x1400 ff.
Receive PDO Mapping Parameter	Index 0x1600 ff.

4 Network configuration with CANconv

4.1 Configuration of a CAN network

The configuration of a CAN network consists now in assigning suitable values for the application to the entries of the object list. An important part of this configuration is the assignment of parameters for T-PDOs and R-PDOs.

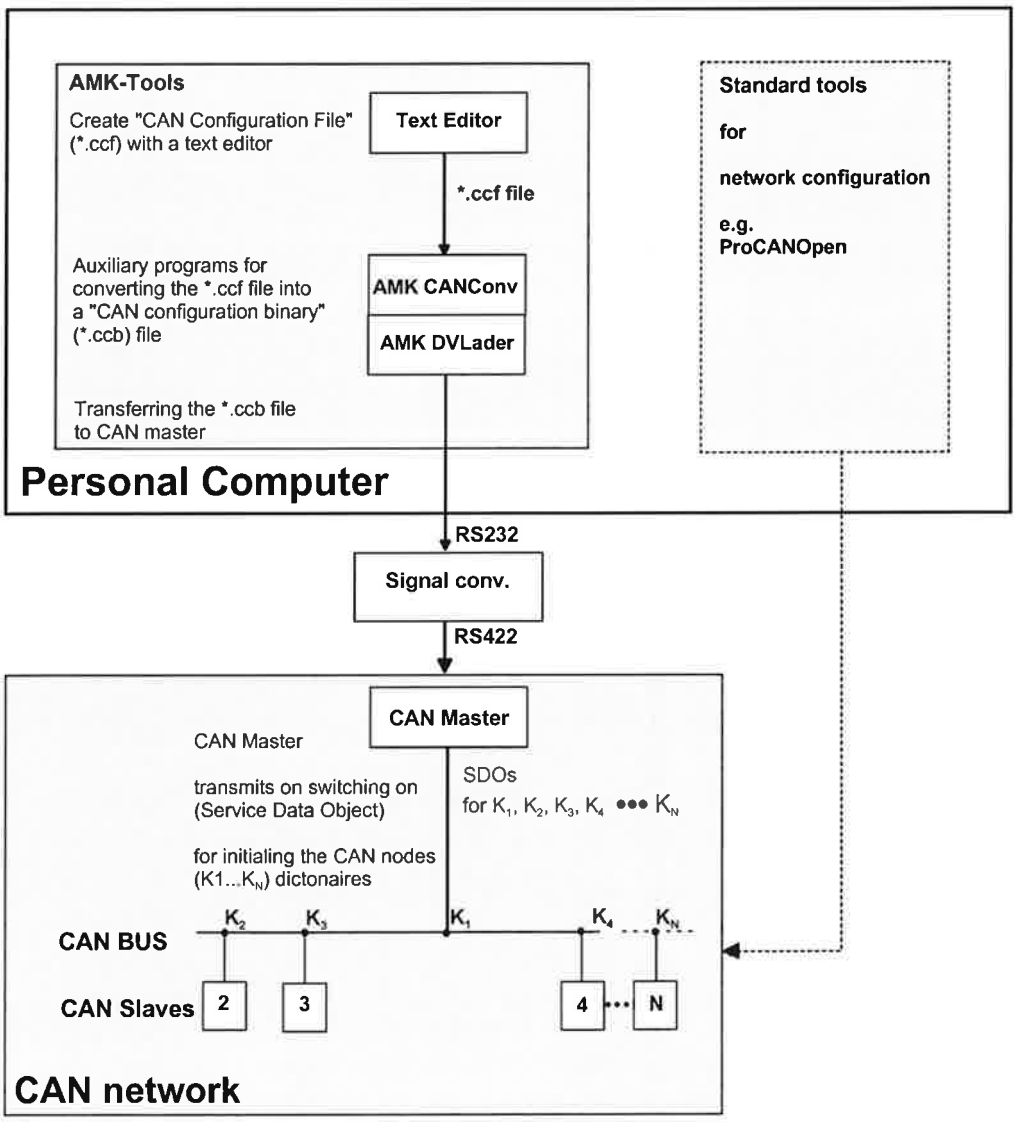
The following questions must be clarified for the configuration of a CAN network:
- **Which nodes should be subscribers in the network?**
- **Which data should be exchanged between the nodes?**
- **When and how frequently must which data be exchanged?**
- **Which priorities do the data to be exchanged have?**

The answers to these questions result in the values for the parameters of PDOs.

Parameters for T-PDOs and R-PDOs are assigned by means of a CAN configuration file (*.ccf). The contents of the CAN configuration file is a list of value assignments for entries in the object lists of the nodes. This file can be created with a standard Windows text editor.

The "CAN Configuration File" is transmitted through the serial interface to the master AE-PSC card after completion with the AMK auxiliary tools CANconv and DVLader. With these data the CAN master firstly initializes its own dictionary and then the dictionaries of the other network nodes through SDOs (Service Data Object).
The following overview illustrates the procedure.

Figure 4-3 Overview of CAN network configuration



The details on communication through CANopen are in:

CiA Draft Standard 301 CANopen Application Layer and Communication Profile

4.2 Requirements on the master configuration

PDOs and SDOs are created only by the configuration, i.e. no PDOs and no SDOs exist without configuration. The following dictionary entries must always be made:

Important:
If a dictionary entry contains subindices, then an entry must be made for each subindex.

1. **PDO indices**
- 0x14XX Receive PDO Communication parameter
- 0x16XX Receive PDO Mapping parameter
- 0x18XX Transmit PDO Communication parameter
- 0x1AXX Transmit PDO Mapping parameter
- 0x1004 Number of PDOs supported (depending upon number of PDOs)

2. **SDO indices**
- 0x1280ff. Client SDO parameter, for each slave to be configured
- 0x1201ff. Server SDO parameter, server SDO 1 (0x1200) already exists
- 0x100F Number of SDOs (depending upon number of ClientSDOs)

3. **AMK master-specific data**
These data correspond to the communication parameters of the KU device. These are currently not accessible for the master in the basic device (AZ/AW).

• 0x2102 sub 0	SERCOS cycle time*	AMK ID00002
• 0x2102 sub 1	BUS subscriber address*	AMK ID34023
• 0x2102 sub 2	BUS transmission rate	AMK ID34024

Values for transmission rate

0	1000 kBaud
1	800 kBaud
2	500 kBaud
3	250 kBaud
4	125 kBaud
5	50 kBaud
6	20 kBaud
7	10 kBaud

• 0x2102 sub 3	BUS mode*	AMK ID34025
• 0x2102 sub 4	BUS mode attribute	AMK ID34026

The bits in the BUS mode attribute have the following meaning

Bit 4	Active Nodes Mode	0	All configured nodes must be present on the bus, otherwise error message
		1	Absent nodes are not initialized

• 0x2102 sub 5	BUS failure behaviour*	AMK ID34027
• 0x2102 sub 6	BUS output rate*	AMK ID34028
• 0x2102 sub 7	BUS parameter	

The bits in BUS parameter have the following meaning:

Bit 0	Node Guard Active	0	No node guard
		1	Node guard is executed

*Parameters are currently used only in the KU. In the CAN configuration file entries in these subindices are indeed not edited but must be entered.

4.3 Requirements on the slave configuration

Indices and subindices can be described in arbitrary order.

The preset values of the following indices can be changed to achieve the required network behaviour:

1. PDO indices
- 0x14XX Receive PDO Communication parameter
- 0x16XX Receive PDO Mapping parameter
- 0x18XX Transmit PDO Communication parameter
- 0x1AXX Transmit PDO Mapping parameter
- 0x1004 Number of PDOs (depending upon number of PDOs)

2. Node Guard
- 0x100C Guard Time (default 0 -Life Guard off)
- 0x100D Life Time (default 0 -Life Guard off)

4.4 Configuration with symbols and predefined files

alias and **rule** are used for repeated settings and the use of symbols for parameters. Use of these keywords can highly simplify the configuration with many nodes. Special care is necessary in use to avoid unwanted overwriting.

The folder ..\user\examples contains 2 predefined files **predefined.ccf** and **confCommon.ccf** which frequently contain the usable definitions and processes.

predefined.ccf contains:
- Definitions of communication indices
- Definitions of COB-IDs from the predefined connection set
- Definitions for use of PS terms for mapping entries
- Definitions of transmission types
- Definitions of frequently used indices
- Definitions of AMK specific indices
- Definitions of complex configuration commands

This file is read at the start of a CAN configuration file with **readFile**.

confCommon.ccf contains:
- Master: automatic setting of index 0x1004 "Number PDOs"
- Master: automatic generation of the necessary client SDO
- Master: setting of index 0x2102 "AMK specific data"
- All slaves: setting of index 0x100c "GuardTime" and index 0x100D "LifeTimeFaktor"

Different symbols must be assigned values for executing **confCommon.ccf.** This file is read at the and of a CAN configuration file with **readFile**.

It is naturally possible to create own definitions and processes and to file them in own *.ccf files. (See AMK additional documentation CANconv)

Example: exam4.ccf. Version with stronger formalization (see printout at the end of this chapter)

5 Configuration by reference to an example

CAN configuration files according to the following example are delivered with the AMK CAN network configuration program CANconv. All examples represent a CAN configuration file for this example. Example 4 is the one with the strongest formalization and most symbolism in the syntax. This example is based on the "predefined.ccf" file delivered with AMK CANconv. This file is predefined so that it has best possible agreement with the PS program parameters, so that creating the CAN configuration file is very illustrative for the PS programmer.

5.1 Nodes in the network

• 1 x AMKASYN AZ-PS5 CAN Master AZ-CNS	AZ-PS5	Node No. 1
• 3 x KU-PSC CAN Slave	KU-PSC	Node No. 2,3,4
• 1 x IO module with 8 outputs and 8 inputs	IO module	Node No. 5

5.2 Data exchange between the nodes

5.2.1 Data exchange from the AZ-PS5 to the KU-PSC

The PS5 sends the following data to all KU-PSC (Broadcast):

Variable	Variable type PS5	Variable type KU-PSC
Control word	Output word	Input word
Control byte 1	Output byte	Input byte
Control byte 2	Output byte	Input byte
Position feedback value	Output double word fast function	Input double word fast function

5.2.2 Data exchange from the KU-PSC to the AZ-PS5

Each KU-PSC sends the following data to the AZ-PS5:

Variable	Variable type PS5	Variable type KU-PSC
Status word	Input word	Output word
Status byte 1	Input byte	Output byte
Status byte 2	Input byte	Output byte
Position feedback value	Input double word fast function	Double word fast function

5.2.3 Data exchange from the AZ-PS5 to the IO module

The AZ-PS5 sends the following data to the output bits of the IO module:

Variable	Variable type PS5	Variable type KU-PSC
Output 1	Output byte	Outputs 8 bits

5.2.4 Data exchange from the IO module to the AZ-PS5

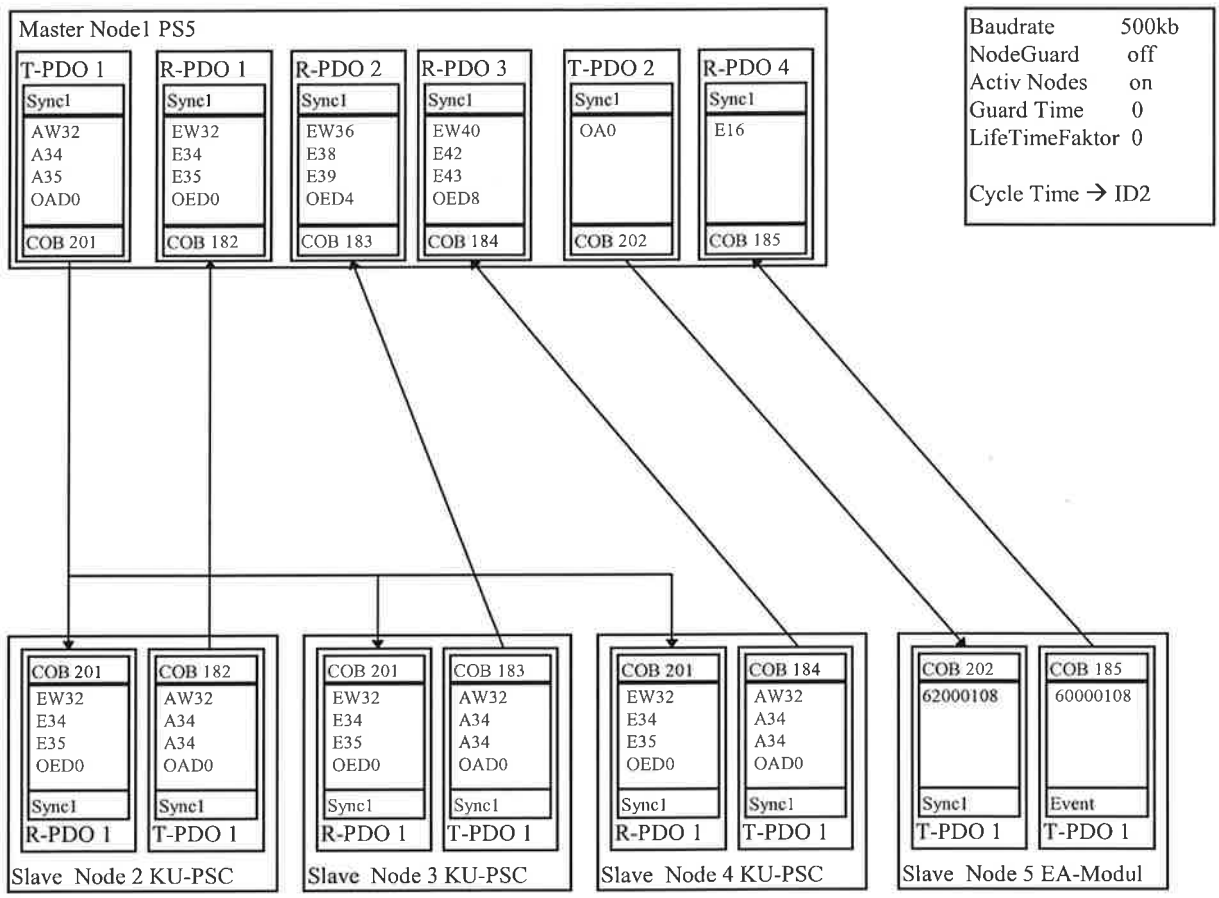
The IO module sends the following data from its input bits to the AZ-PS5

Variable	Variable type PS5	Variable type KU-PSC
Input 1	Input byte	Inputs 8 bits

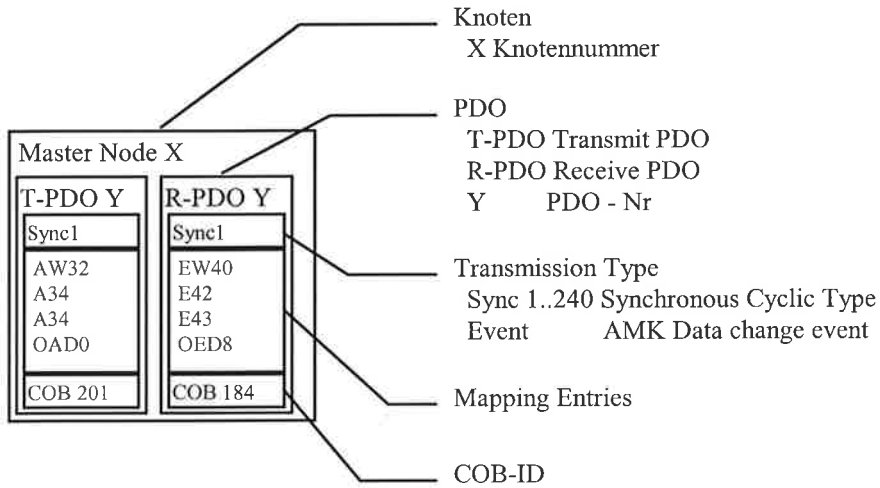
5.3 Priority and frequency of the data exchange

Data exchange	Data exchange frequency	Priority
KU-PSC → AZ-PS5	in each communication cycle	1
IO module → AZ-PS5	in each communication cycle	2
AZ-PS5 → KU-PSC	in each communication cycle	3
AZ-PS5 → IO module	on data change	4

5.4 Configuration overview of the example application



Key



5.5 Mapping entries

The view of the PS programmer on the CAN bus is described in the document Extensions to PS command inventory V02.14.

The documentation of the PS functions explains all input and output variables to be transferred which have to be defined for the required PS function in the CAN network.

5.5.1 PS designator names

To be able to work with terms of the PS when stating the PDO mapping, the following names are agreed:

Ex	Input byte
EWx	Input word
EDx	Input double word
OEx	Input byte optional module for fast function
OEWx	Input word optional module for fast function
OEDx	Input double word optional module for fast function

Ax	Output byte
AWx	Output word
ADx	Output double word
OAx	Output byte optional module for fast function
OAWx	Output word optional module for fast function
OADx	Output double word optional module for fast function

The relation between PS designators and CAN index/subindex is defined predefined in predefined.ccf with "CANconv". Thus the use of the PS designator is possible for the user of "CANconv AMK CAN network configuration tool".

5.5.2 Relation of PS designator to the CAN index/subindex

	OUT asyn from PS	OUT sync from PS	IN asyn to PS	IN sync to PS
Index DWord	200C sub 1-64	200F sub 1-64	2000 sub1-64	2003 sub 1-64
PS DW	AD0,4,8...252	OAD0,4,8...252	ED0,4,8...252	OED0,4,8...252
Index Word	200D sub 1-128	2010 sub 1-128	2001 sub 1-128	2004 sub 1-128
PS Word	AW0,2,4...254	OAW0,2,4...254	EW0,2,4...254	OEW0,2,4...254
Index Byte	200E sub 1-255	2011 sub 1-255	2002 sub 1-255	2005 sub 1-255
PS Byte	A0,1,2,3...255	---	E0,1,2,3...255	---

5.5.3 Mapped objects of the IO module

The objects used here are agreed in CiA-DS 401 Device Profile for I/O modules.

62000108 Index 6200 Subindex 01 size 08 (8-bit)
8 bit digital output

60000108 Index 6000 Subindex 01 size 08 (8-bit)
8 bit digital input

5.6 Example (DocExam4.ccf)

Example of the subject of configuration with symbols and predefined files (version with stronger formalization)

```
// ================================================================================
//  Filename: DocExam4.ccf
//
//  Date: 10.08.00
// ================================================================================
    readFile predefined.ccf
//------------------------
//  Common parameters
//------------------------
    nodelist 1  2 3 4 5
    nodegroup slaves 2 3 4 5

    alias BaudrateVal       500kb
    alias NodeGuardVal        0     //0 off   1 on
    alias ActivNodesVal      0x10     //0 off 0x10 on
    alias GuardTimeVal      0x0000
    alias LifeTimeFaktorVal   0x00

//-------------------------
//  Master Configuration
//-------------------------
//  1.Transmit PDO
    alias PDOno       1
    alias master      1
    alias Map         AW32 A34 A35 OAD0
    alias TransTyp    CycSync1
    alias COBBrodcast  0x00000201
    alias COBpdo      COBBrodcast
    aliasend

    confMasterTransmitPDO

//  1. Receive PDO  from Slave 2
    alias PDOno       1
    alias slave       2
    alias Map         EW32 E34 E35 OED0
    alias TransTyp    CycSync1
    alias COBpdo      COBtx1
    aliasend

    confMasterReceivePDO

//  2. Receive  PDO from Slave 3
    alias PDOno       2
    alias slave       3
    alias Map         EW36 E38 E39 OED4
    aliasend

    confMasterReceivePDO
```

```
//  3. Receive  PDO from Slave 4
      alias PDOno       3
      alias slave       4
      alias Map         EW40 E42 E43 OED8
      aliasend

      confMasterReceivePDO

//I/O-Modul
//  2.Transmit PDO
      alias PDOno       2
      alias master      1
      alias Map         OA0
      alias TransTyp    CycSync1
      alias COBterminal  0x00000202
      alias COBpdo      COBterminal
      aliasend

      confMasterTransmitPDO

//  4. Receive PDO  from Slave 5
      alias PDOno       4
      alias slave       5
      alias Map         E16
      alias TransTyp    CycSync1
      alias COBpdo      COBtx1
      aliasend

      confMasterReceivePDO


//------------------------
//  Slave (KU) Configuration
//------------------------
//  1. Transmit PDO
      alias PDOno       1
      alias TransTyp    CycSync1
      alias Map         AW32 A34 A35 OAD0
      alias COBpdo      COBtxNo0
      aliasend

      alias slave       2 aliasend
      confSlaveTransmitPDO

      alias slave       3 aliasend
      confSlaveTransmitPDO

      alias slave       4 aliasend
      confSlaveTransmitPDO


//  2. Transmit PDO  switch off
      alias PDOno       2
      alias COBpdo      COBtxNo
      aliasend

      alias slave       2 aliasend
      confSlaveTransmitPDOoff

      alias slave       3 aliasend
      confSlaveTransmitPDOoff

      alias slave       4 aliasend
      confSlaveTransmitPDOoff

//  1. Receive PDO
      alias PDOno       1
      alias Map         EW32 E34 E35 OED0
      alias TransTyp    CycSync1
      alias COBpdo      COBBrodcast
      aliasend

      alias slave       2 aliasend
      confSlaveReceivePDO

      alias slave       3 aliasend
```

```
      confSlaveReceivePDO

      alias slave       4 aliasend
      confSlaveReceivePDO

//------------------------
//  I/O-Modul
//------------------------
//  1. Transmit PDO
      alias slave       5
      alias PDOno       1
      alias TransTyp    DeviceEvent
      alias COBpdo      COBtxNo0
      alias Map         0x60000108
      aliasend

      confSlaveTransmitPDOnew

//  2. Transmit PDO
      alias PDOno       2
      alias COBpdo      COBnotValid
      alias Map         0x64010110
      aliasend

      confSlaveTransmitPDOnew

//  1. Receive PDO
      alias PDOno       1
      alias COBpdo      COBterminal
      alias Map         0x62000108
      aliasend

      confSlaveReceivePDOnew


readFile confCommon.ccf

/*================================= EOF ====================================*/
```

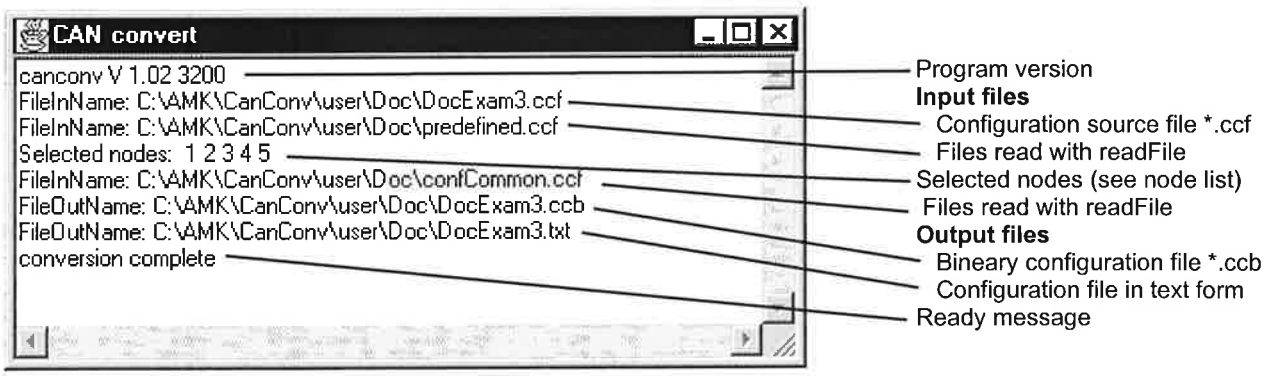
6 Converting with AMK tool CANConv

The following operations are performed in the conversion of a configuration source file (*.ccf) into a configuration binary file (*.ccb):

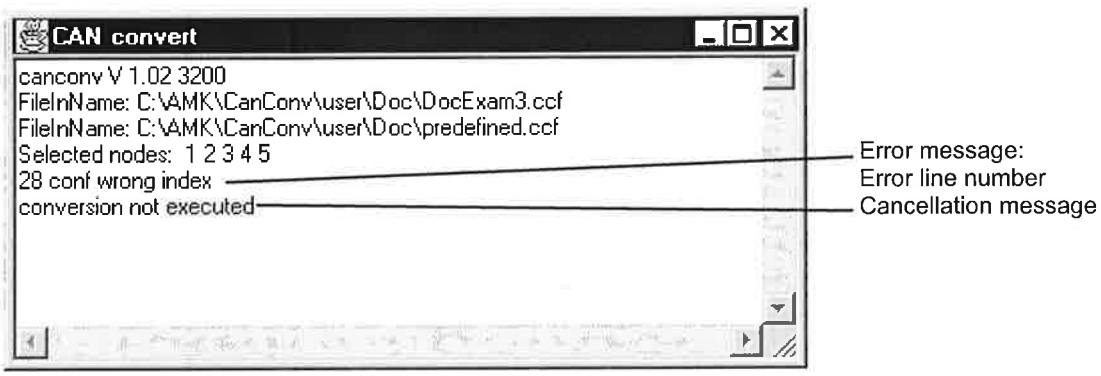
- Creation of the entries according to regulations in the configuration source file
- However no configuration data are contained for nodes which are listed in the **nodelist**, stands in the configuration binary file for number of supported entries: 0
- Configuration data for nodes which stand in **nodelist** are removed
- Sorting the entries according to node number
- Sorting the entries in node 1 = Master according to index and subindex
- Testing for double entries for index/subindex of a node
  Note: Double entries generated by **PDO** for **[COBID]** and the number of the mapping entries are ignored
- Output of the result in the configuration binary file (*.ccb)
- Transformation of the configuration binary file (*.ccb) into a text file *.txt
- If errors are determined, then the conversion is aborted and there is an error output

6.1 Results message

Screen display for errorfree conversion (example)



Screen display on faulty configuration source file (example):



6.2 Installation and selection of CANConv

The installation of all required files including examples is performed by selecting **setupCanConvert.exe**.

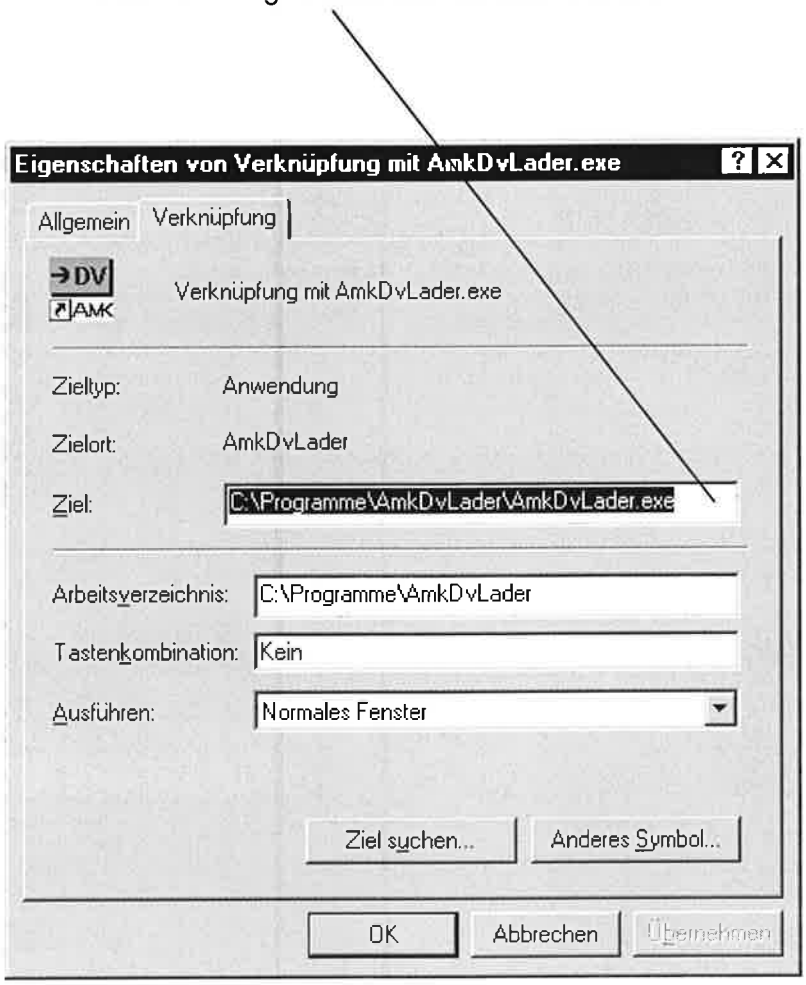
Standalone mode
Selecting **Canconv.bat** starts the application and initially shows a file selection dialog. After a file of the type *.ccf has been selected, the conversion starts automatically. After completed conversion (see results message)
the window can be closed again. On frequent use you are recommended to create a link with **Canconv.bat** on the desktop. To avoid visibility of the DOS window, the properties "Execute as symbol" and "Close on exiting" should be assigned to it (only Win95/Win98).

Linking with AmkDvLader
The configuration binary files *.ccb created in the conversion must be loaded on the CAN master for your application with the aid of the AmkDvLader program. Canconv.bat can also be selected by AmkDvLader. For this purpose the selection of AmkDvLader must be supplemented by the following parameters (see also AmkDvLader description).

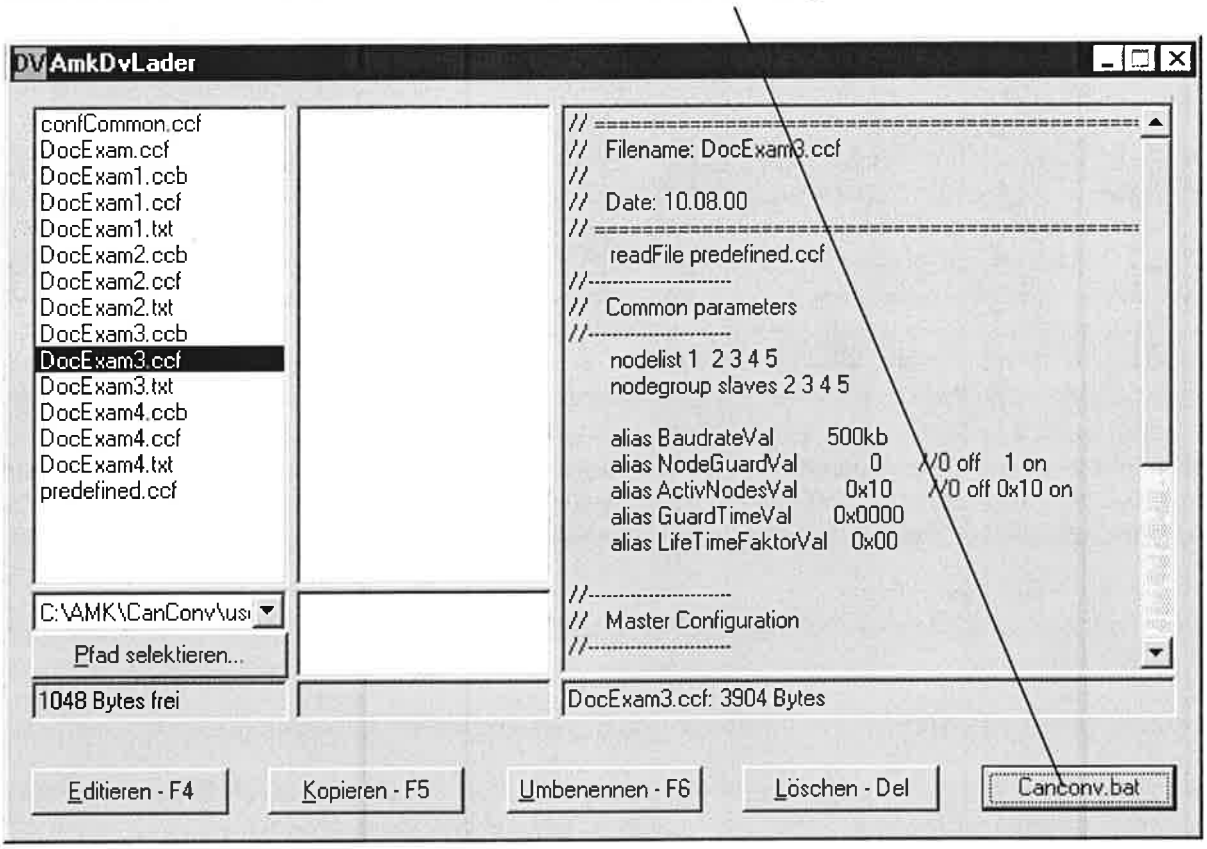
-tccf -cC:*YOUR_PATH*\CanConv\Canconv.bat -eccf

e.g.

-tccf -cC:\Programme\CanConv\Canconv.bat -eccf



AmkDvLader now receives an additional button "Canconv.bat".



The button is activated if a file of the type *.ccf is selected. The conversion of the selected *.ccf file is started by activating the "Canconv.bat" button. After successful conversion the created *.ccb file can be loaded immediately to the CAN master by activating "Copy - F5".

7 AMK tool DVLader

7.1 Introduction

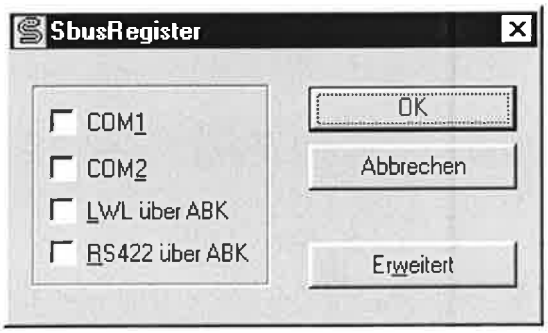
The program described here is a Windows program for access to file management systems (DV) on MC and PS cards which can be reached through Sbus. It has the following properties:

- Simultaneous display of all files of a selected PC path and a selected DV (2-window display)
- Fast view of the contents of a selected file (PC or DV) as binary file or for known file types as text file
- Functions for copying between PC and DV (both directions) as well as for deleting and renaming files in the PC or in the DV
- Display of the current Sbus topology (all active interfaces as well as the DVs connected to them)
- The program works under Window 95®, Windows 98® and Windows NT®. The Sbus can always be used through Com interface; the AB-K02 card does **not** function under **Windows NT**!
- There is a freely assignable special function for selecting an arbitrary program.

The general principles of Windows operation are presupposed in this description.

7.2 Configuring Sbus

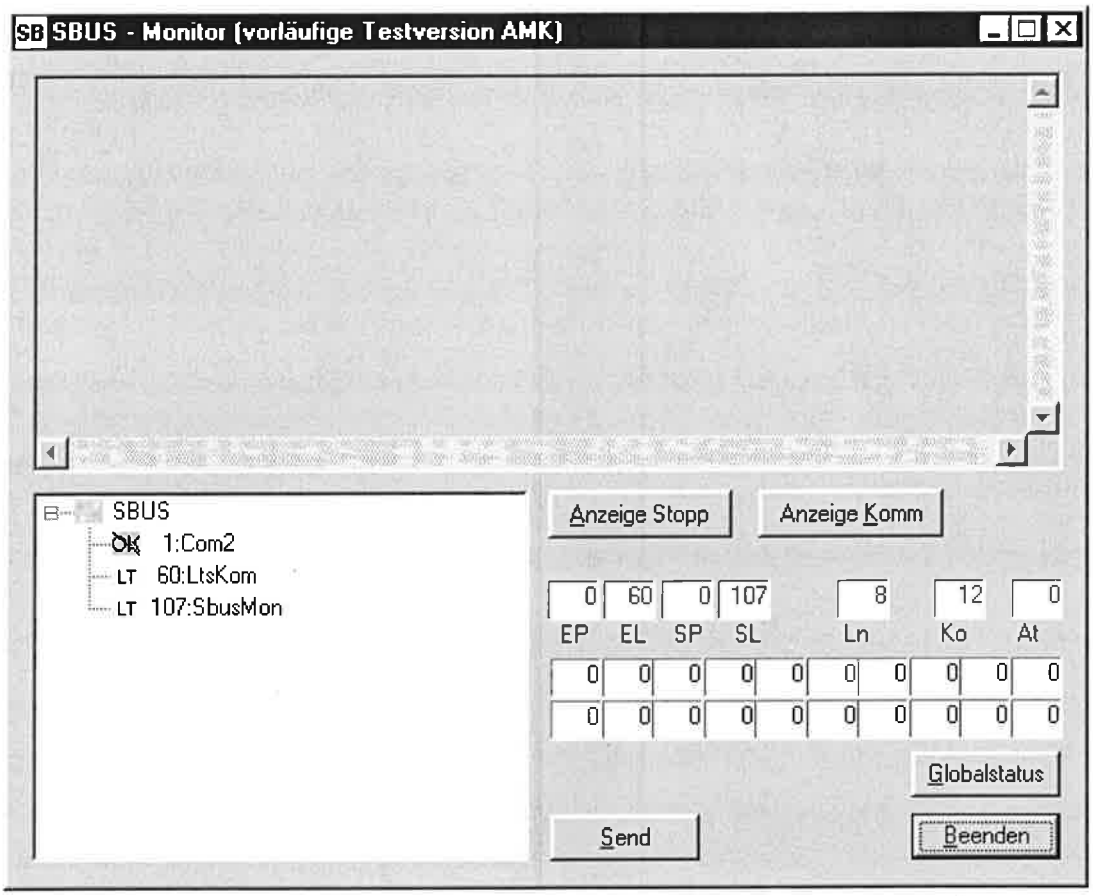
So that it is possible to work with the Sbus, its physical interfaces (physical ports) must be defined and configured. This is done using the "SbusRegister" program which is enclosed with the installation.



In the normal case only the required COM port 1 or 2 must be selected and confirmed by OK button. The two other options concern bus interfaces through additional special hardware. Other Com interfaces can be set using the "Advanced" button if available in the current PC.

The settings which are made here are valid globally for all programs running on this PC. They must be made generally before selecting the DvLader.

The "SbusDialog" program also enclosed is used for testing the Sbus connection.



It is possible to test with this whether a connection between PC and the opposite number can be made with the selected settings. The connection changes the icon of the Com port into the OK state.

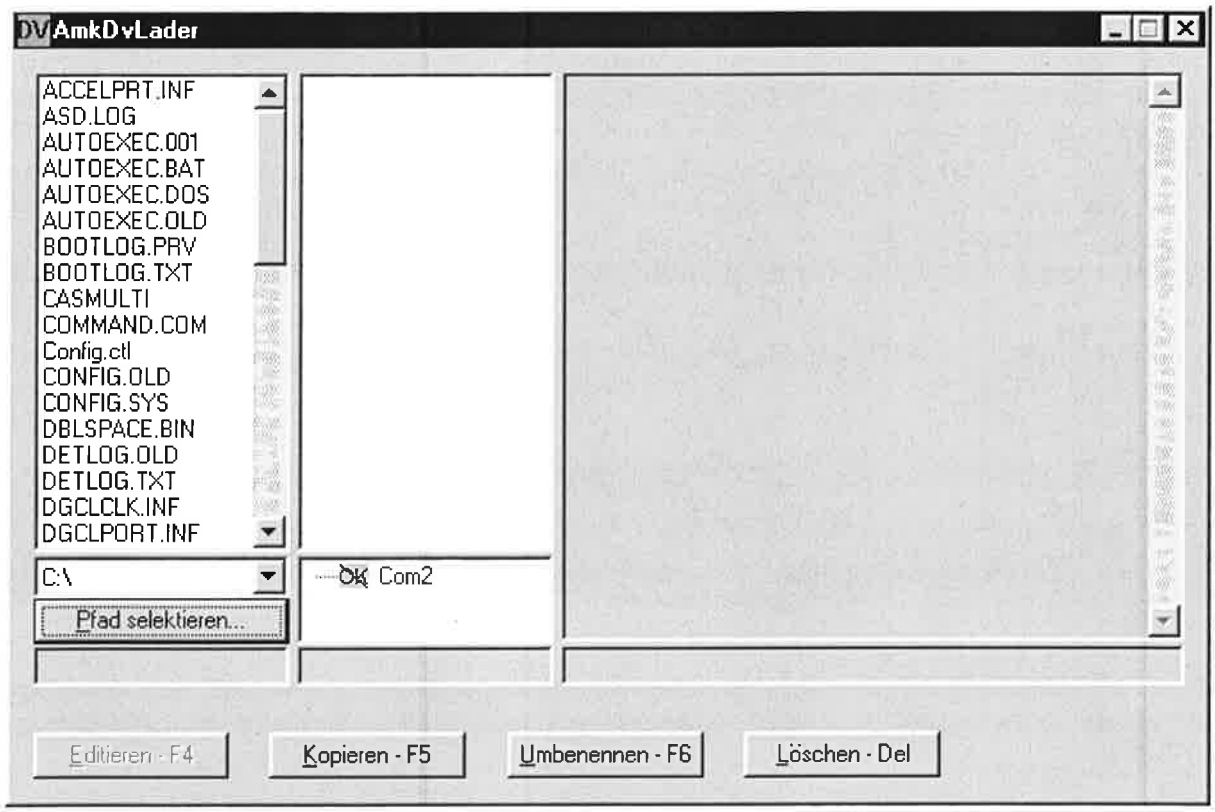
Note!

This test program cannot run jointly with another Sbus application in the normal case. Therefore close the monitor before starting the DvLader!

7.3 Operation

7.3.1 Starting the program

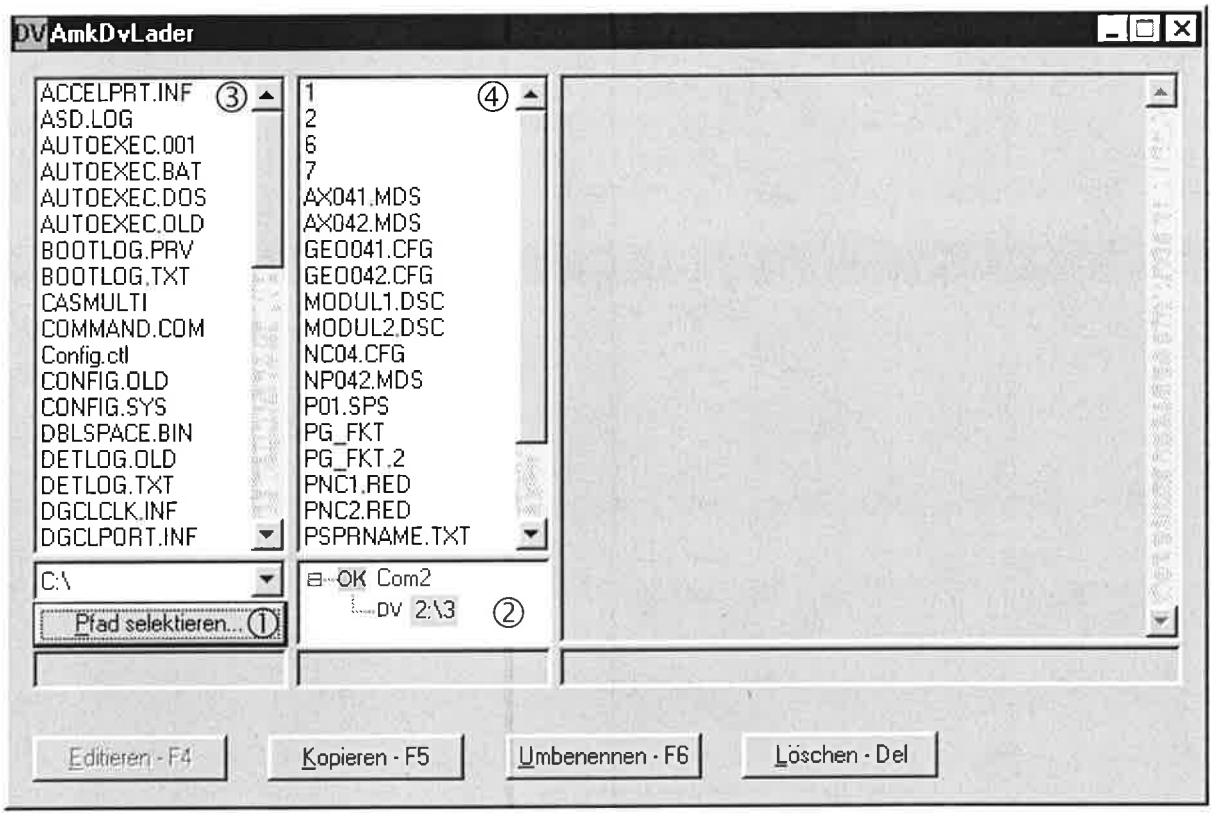
The following main window appears after selecting the program:



The workspace is divided into three vertically divided parts:

- The left column shows the selected PC path and the files contained in it
- The central column is used for displaying in each case a selected DV and the files contained in it
- The right, largest part serves for displaying the contents of a file marked in one of the two left column (fast view)

After a few seconds the Sbus goes into the OK state and the DV and the files contained in it are displayed in the central column:



7.3.2 Selection of the working path in the PC

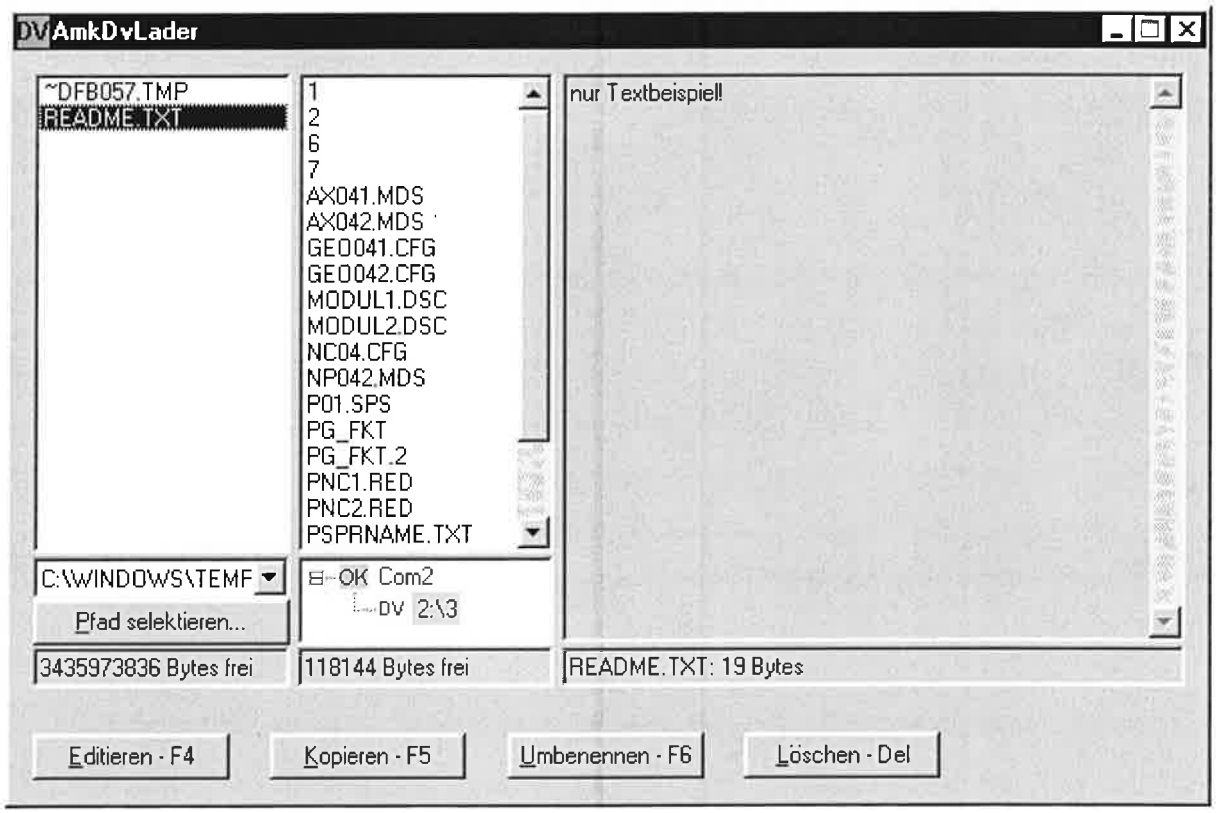
A selection dialog with which the PC working path can be selected appears after activating the "Select path..." button ①.

7.3.3 Selection of the DV

In the normal case only one DV is available on the Sbus. This is then selected automatically. If several DVs are active, the required DV must be selected by the operator by clicking in the tree diagram ②.

7.3.4 Selection of the file to be edited

To be able to work with the actual DV function, a file must be selected in the PC ③ or in the DV ④. Only one of the files is always ready for editing either in the PC or in the DV. The contents of the selected file are displayed in the window of the fast view. The display is made in text form if the file is recognized as text file, or binary for all other file types. Only the first 1000 bytes of the file are displayed in each case and the remainder is cut off.



7.3.5 Editing and copying a file

The buttons at the bottom of the window refer generally to the file selected in the PC or in the DV:

- Edit: The selected file is transferred to the Windows system editor for editing. This function works only with the files available on the PC.
- Copy: The selected file is copied from the PC to the DV or vice versa.
- Rename: The selected file can be remained.
- Delete: The selected file is deleted (after a prompt).

7.4 Configuration possibilities through command line parameters

7.4.1 Extension of the list of the text file formats

The following file types are always interpreted as text files:

- .red
- .ini
- .txt

This list can be extended as desired using the command line parameter "-t". The file types .doc and .bat are displayed in addition as text files by selecting the program with the parameters

DvLader -tdoc -tbat

Caution: There must be no space between the command and the file type!

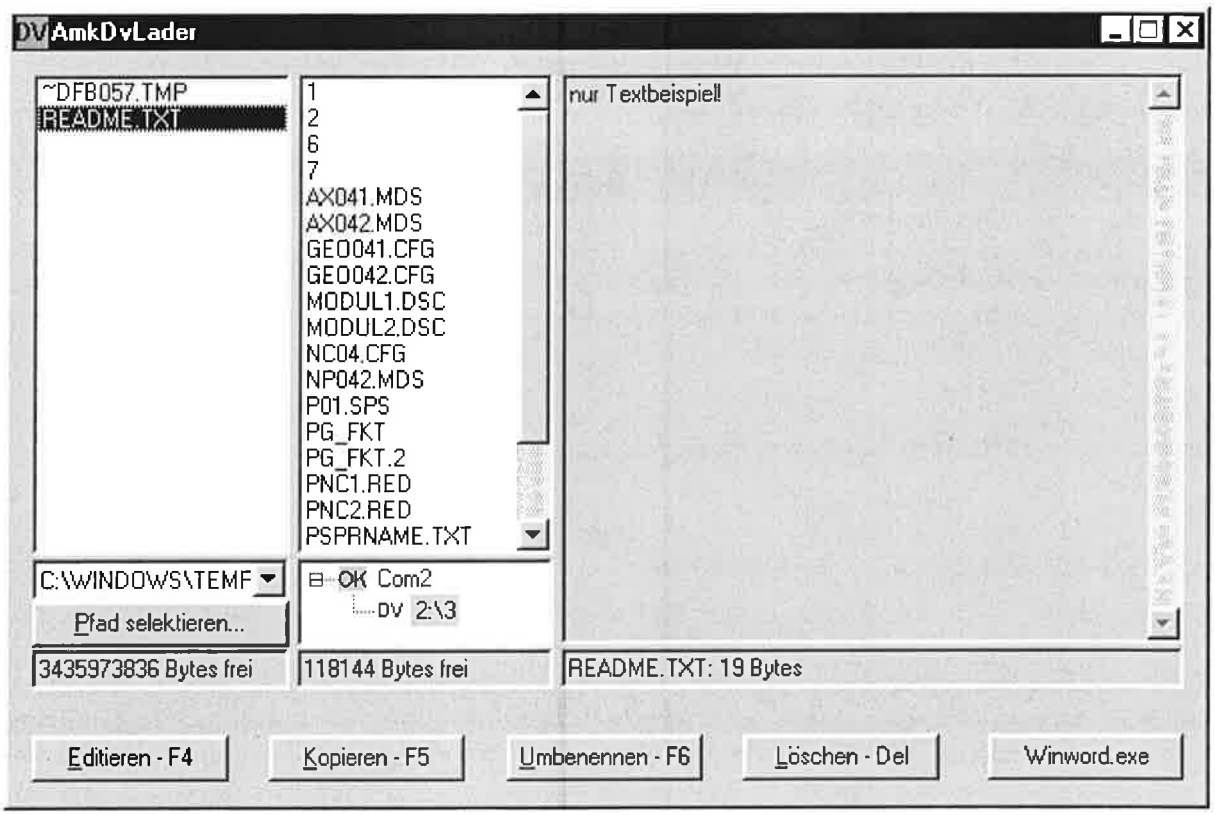
7.4.2 Definition of a special function

The DvLader program can be extended by the selection of a special function. An executable file which implements the function is determined by the command "-c". The file types which can edit the special function are defined with the command "-e". The special function refers like all other functions always to the selected file. However, it works only with files available on the PC.

In the following case

DvLader -cc:\programme\msoffice\winword\Winword.exe -edoc -etxt

the special function becomes active if the selected file has the type .doc or .txt.



8 Appendix

8.1 Excerpt from CANopen

8.1.1 Manufacturer-specific data types

All definitions and abbreviations in the following tables are in conformity with the agreements according to CiA Draft Standard 301 – "CAL-based Communication Profile for industrial Systems". You will find further information in CiA Draft Standard 301!

KU-PSC / AZ-CNS

Index (hex)	Object	Name	Type	Attr.	M/O	Default Value
40	DEFSTRUCT	Special overlapped PDO Mapping - Byte	Unsigned8	ro	O	
41	DEFSTRUCT	Special overlapped PDO Mapping - Word	Unsigned8	ro	O	
42	DEFSTRUCT	Special overlapped PDO Mapping – Double Word	Unsigned8	ro	O	

These objects serve for implementing overlapping data from different objects in the DPRAM input/output area which is used by the PLC.

8.1.2 Manufacturer-specific "Profile Area Objects"

KU-PSC / AZ-CNS
"Manufacturer-specific Profile Area Objects" are summarized in the following table:

Index (hex)	Object	Name	Type	Attr.	M/O	Default Value
2000	RECORD	Asynchronous Input Data Area - Double Word	42h	ro	O	
2001	RECORD	Asynchronous Input Data Area - Word	41h	ro	O	
2002	RECORD	Asynchronous Input Data Area – Byte	40h	ro	O	
2003	RECORD	Synchronous Input Data Area - Double Word	42h	ro	O	
2004	RECORD	Synchronous Input Data Area – Word	41h	ro	O	
2005	RECORD	Synchronous Input Data Area – Byte	40h	ro	O	
200C	RECORD	Asynchronous Output Data Area – Double Word	42h	ro	O	
200D	RECORD	Asynchronous Output Data Area – Word	41h	ro	O	

Index (hex)	Object	Name	Type	Attr.	M/O	Default Value
200E	RECORD	Asynchronous Output Data Area – Byte	40h	ro	O	
200F	RECORD	Synchronous Output Data Area - Double Word	42h	ro	O	
2010	RECORD	Synchronous Output Data Area – Word	41h	ro	O	
2011	RECORD	Synchronous Output Data Area – Byte	40h	ro	O	
2100	VAR	Concise DCF name	Visib. String	ro	O	Defined by DCF
2101	VAR	Concise DCF version	Unsigned8	ro	O	Defined by DCF
2102	ARRAY	AMK specific data (only for AZ-CNS)			M	
2102sub0	VAR	SERCOS cycle (ID00002)	Unsigned16	ro		Defined by DCF
2102sub1	VAR	Node-ID (ID34023) Not used!	Unsigned16	ro		Defined by DCF
2102sub2	VAR	Kbauds Selection (ID34024)	Unsigned16	ro		Defined by DCF
2102sub3	VAR	BUS Mode (ID34025) Not used!	Unsigned16	ro		Defined by DCF
2102sub4	VAR	BUS Mode Attribute (ID34026)	Unsigned16	ro		Defined by DCF
2102sub5	VAR	Busfailure Behaviour (ID34027) Not used!	Unsigned16	ro		Defined by DCF
2102sub6	VAR	BUS Rate (ID34028) Not used!	Unsigned16	ro		Defined by DCF
2102sub7	VAR	AMK Parameters	Unsigned16	ro		Defined by DCF

The index 2102h is used only by the AZ-CNS. Subindices contain data corresponding to the AZ parameter (Ident numbers or in short IDs). Currently not all subindices are used. Unused subindices are reserved for future use.

8.1.3 Communication profile "Communication Profile Area"

8.1.3.1 KU-PSC communication profile

Index (hex)	Object	Name	Type	Attr.	M/O	Default Value
1000	VAR	DeviceType	Unsigned32	ro	M	0001012Eh
1001	VAR	ErrorRegister	Unsigned8	ro	M	00h
1003	RECORD	Pre-definedErrorField	21h		O	
1003sub0	VAR	NumberOfEntries	Unsigned8	rw		00h
1003sub1	VAR	StandardErrorField 1	Unsigned32	ro		00000000h

Index (hex)	Object	Name	Type	Attr.	M/O	Default Value
1003sub1	VAR	StandardErrorField 2	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 3	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 4	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 5	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 6	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 7	Unsigned32	ro		00000000h
1003sub8	VAR	StandardErrorField 8	Unsigned32	ro		00000000h
1004	ARRAY	NumberOfPDOsSupported			O	
1004sub0	VAR	NumberOfPDOsSupported	Unsigned32	ro		00020002h
1004sub1	VAR	NumberOfSyncPDOsSupp.	Unsigned32	ro		00000002h
1004sub2	VAR	NumberOfAsyncPDOsSupp	Unsigned32	ro		00020000h
1005	VAR	Sync COB-ID	Unsigned32	rw	O	80000080h
1008	VAR	DeviceName	Visib. String	ro	O	AE-PSC
1009	VAR	HardwareVersion	Visib. String	ro	O	V1.0
100A	VAR	SoftwareVersion	Visib. String	ro	O	V1.0
100B	VAR	Node-ID	Unsigned32	ro	O	Node-ID
100C	VAR	GuardTime	Unsigned16	rw	O	0
100D	VAR	LifeTimeFactor	Unsigned8	rw	O	0
100E	VAR	Guard COB-ID	Unsigned32	rw	O	1792 + Node-ID
100F	VAR	NumberOfSDOsSupported	Unsigned32	ro	O	00000001h
1014	VAR	Emergency COB-ID	Unsigned32	rw	O	128 + Node-ID
1200	RECORD	Server SDO1 Parameter	22h	ro	O	
1200sub0	VAR	NumberOfEntries	Unsigned8	ro		3
1200sub1	VAR	COB-ID Client-Server	Unsigned32	ro		1536 + Node-ID
1200sub2	VAR	COB-ID Server-Client	Unsigned32	ro		1408 + Node-ID
1200sub3	VAR	Node-ID	Unsigned8	ro		Node-ID
1400	RECORD	ReceivePDO1 Parameter	20h	rw	O	
1400sub0	VAR	NumberOfEntries	Unsigned8	rw		4
1400sub1	VAR	COB-ID	Unsigned32	rw		512 + Node-ID
1400sub2	VAR	TransmissionType	Unsigned8	rw		254
1400sub3	VAR	InhibitTime	Unsigned16	rw		0
1400sub4	VAR	CMSPriorityGroup	Unsigned8	rw		2
1401	RECORD	ReceivePDO1 Parameter	20h	rw	O	
1401sub0	VAR	NumberOfEntries	Unsigned8	rw		4
1401sub1	VAR	COB-ID	Unsigned32	rw		768 + Node-ID
1401sub2	VAR	TransmissionType	Unsigned8	rw		254
1401sub3	VAR	InhibitTime	Unsigned16	rw		0
1401sub4	VAR	CMSPriorityGroup	Unsigned8	rw		2
1600	RECORD	ReceivePDO1 Mapping	21h	rw	O	
1600sub0	VAR	NumberOfMappedObjects	Unsigned8	rw		8
1600sub1	VAR	MappedObject 1	Unsigned32	rw		20022108h
1600sub2	VAR	MappedObject 2	Unsigned32	rw		20022208h
1600sub3	VAR	MappedObject 3	Unsigned32	rw		20022308h
1600sub4	VAR	MappedObject 4	Unsigned32	rw		20022408h
1600sub5	VAR	MappedObject 5	Unsigned32	rw		20022508h

Index (hex)	Object	Name	Type	Attr.	M/O	Default Value
1600sub6	VAR	MappedObject 6	Unsigned32	rw		20022608h
1600sub7	VAR	MappedObject 7	Unsigned32	rw		20022708h
1600sub8	VAR	MappedObject 8	Unsigned32	rw		20022808h
1601	RECORD	ReceivePDO1 Mapping	21h	rw	O	
1601sub0	VAR	NumberOfMappedObjects	Unsigned8	rw		8
1601sub1	VAR	MappedObject 1	Unsigned32	rw		62000108h
1601sub2	VAR	MappedObject 2	Unsigned32	rw		62000208h
1601sub3	VAR	MappedObject 3	Unsigned32	rw		62000308h
1601sub4	VAR	MappedObject 4	Unsigned32	rw		62000408h
1601sub5	VAR	MappedObject 5	Unsigned32	rw		62000508h
1601sub6	VAR	MappedObject 6	Unsigned32	rw		62000608h
1601sub7	VAR	MappedObject 7	Unsigned32	rw		62000708h
1601sub8	VAR	MappedObject 8	Unsigned32	rw		62000808h
1800	RECORD	TransmitPDO1 Parameter	20h	rw	O	
1800sub0	VAR	NumberOfEntries	Unsigned8	rw		4
1800sub1	VAR	COB-ID	Unsigned32	rw		384 + Node-ID
1800sub2	VAR	TransmissionType	Unsigned8	rw		1
1800sub3	VAR	InhibitTime	Unsigned16	rw		0
1800sub4	VAR	CMSPriorityGroup	Unsigned8	rw		1
1801	RECORD	TransmitPDO1 Parameter	20h	rw	O	
1801sub0	VAR	NumberOfEntries	Unsigned8	rw		4
1801sub1	VAR	COB-ID	Unsigned32	rw		640 + Node-ID
1801sub2	VAR	TransmissionType	Unsigned8	rw		1
1801sub3	VAR	InhibitTime	Unsigned16	rw		0
1801sub4	VAR	CMSPriorityGroup	Unsigned8	rw		1
1A00	RECORD	TransmitPDO1 Mapping	21h	rw	O	
1A00sub0	VAR	NumberOfMappedObjects	Unsigned8	rw		8
1A00sub1	VAR	MappedObject 1	Unsigned32	rw		200E2108h
1A00sub2	VAR	MappedObject 2	Unsigned32	rw		200E2208h
1A00sub3	VAR	MappedObject 3	Unsigned32	rw		200E2308h
1A00sub4	VAR	MappedObject 4	Unsigned32	rw		200E2408h
1A00sub5	VAR	MappedObject 5	Unsigned32	rw		200E2508h
1A00sub6	VAR	MappedObject 6	Unsigned32	rw		200E2608h
1A00sub7	VAR	MappedObject 7	Unsigned32	rw		200E2708h
1A00sub8	VAR	MappedObject 8	Unsigned32	rw		200E2808h
1A01	RECORD	TransmitPDO1 Mapping	21h	rw	O	
1A01sub0	VAR	NumberOfMappedObjects	Unsigned8	rw		8
1A01sub1	VAR	MappedObject 1	Unsigned32	rw		200E2108h
1A01sub2	VAR	MappedObject 2	Unsigned32	rw		200E2208h
1A01sub3	VAR	MappedObject 3	Unsigned32	rw		200E2308h
1A01sub4	VAR	MappedObject 4	Unsigned32	rw		200E2408h
1A01sub5	VAR	MappedObject 5	Unsigned32	rw		200E2508h
1A01sub6	VAR	MappedObject 6	Unsigned32	rw		200E2608h
1A01sub7	VAR	MappedObject 7	Unsigned32	rw		200E2708h
1A01sub8	VAR	MappedObject 8	Unsigned32	rw		200E2808h

8.1.3.2 AZ-CNS communication profile

Index (hex)	Object	Name	Type	Attr.	M/O	Default Value
1000	VAR	DeviceType	Unsigned32	ro	M	0000012Eh
1001	VAR	ErrorRegister	Unsigned8	ro	M	00h
1003	RECORD	Pre-definedErrorField	21h		O	
1003sub0	VAR	NumberOfEntries	Unsigned8	rw		00h
1003sub1	VAR	StandardErrorField 1	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 2	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 3	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 4	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 5	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 6	Unsigned32	ro		00000000h
1003sub1	VAR	StandardErrorField 7	Unsigned32	ro		00000000h
1003sub8	VAR	StandardErrorField 8	Unsigned32	ro		00000000h
1004	ARRAY	NumberOfPDOsSupported			O	
1004sub0	VAR	NumberOfPDOsSupported	Unsigned32	ro		00000000h
1004sub1	VAR	NumberOfSyncPDOsSupp.	Unsigned32	ro		00000000h
1004sub2	VAR	NumberOfAsyncPDOsSupp	Unsigned32	ro		00000000h
1005	VAR	Sync COB-ID	Unsigned32	rw	O	C000080h
1008	VAR	DeviceName	Visib. String	ro	O	AE-PSC
1009	VAR	HardwareVersion	Visib. String	ro	O	V1.0
100A	VAR	SoftwareVersion	Visib. String	ro	O	V1.0
100B	VAR	Node-ID	Unsigned32	ro	O	Node-ID
100C	VAR	GuardTime	Unsigned16	rw	O	0
100D	VAR	LifeTimeFactor	Unsigned8	rw	O	0
100E	VAR	Guard COB-ID	Unsigned32	rw	O	1792 + Node-ID
100F	VAR	NumberOfSDOsSupported	Unsigned32	ro	O	00000001h
1014	VAR	Emergency COB-ID	Unsigned32	rw	O	128 + Node-ID
1200	RECORD	Server SDO1 Parameter	22h	ro	O	
1200sub0	VAR	NumberOfEntries	Unsigned8	ro		3
1200sub1	VAR	COB-ID Client-Server	Unsigned32	ro		1536 + Node-ID
1200sub2	VAR	COB-ID Server-Client	Unsigned32	ro		1408 + Node-ID
1200sub3	VAR	Node-ID	Unsigned8	ro		Node-ID

8.1.4 Standard device profile "Standard Device Profile Area"

KU-PSC / AZ-CNS

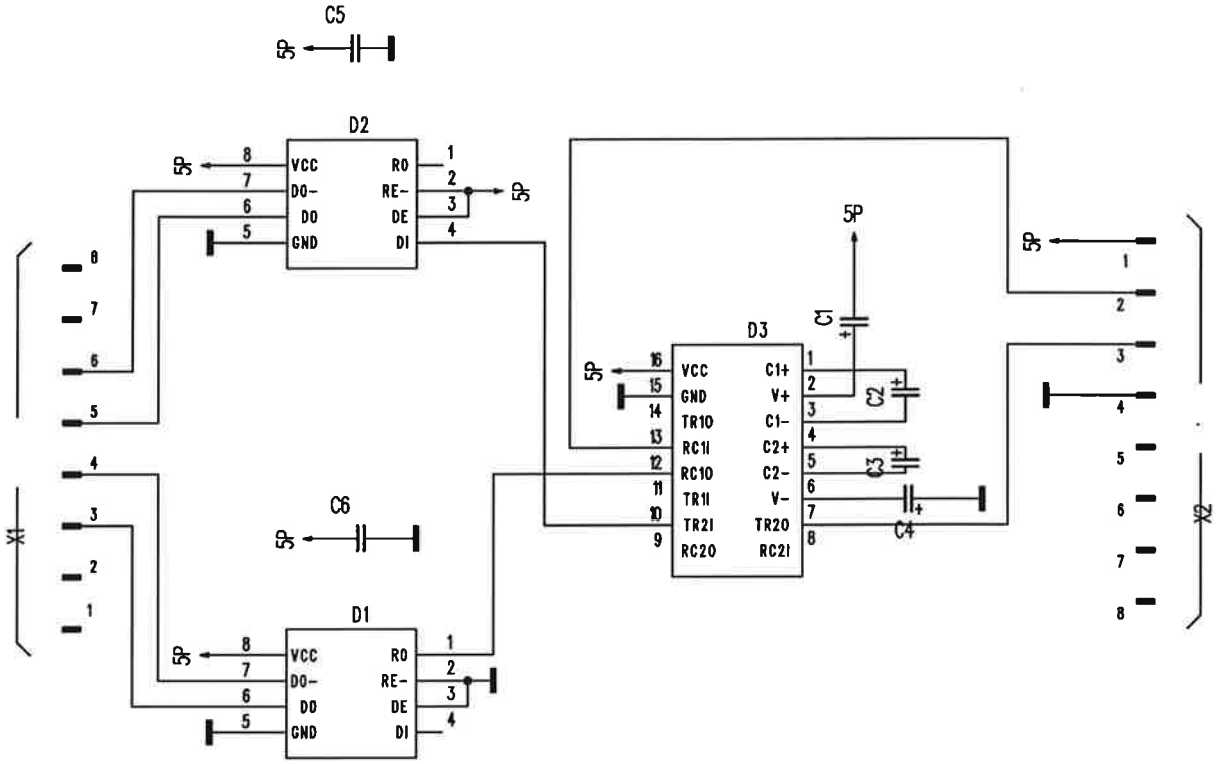
Index (hex)	Object	Name	Type	Attr.	M/O	Default Value
6000	ARRAY	ReadState8InputLines	Unsigned8	ro	O	
6000sub0	VAR	NumberOfElements	Unsigned8	ro		8
6000sub1	VAR	ReadStateByte1	Unsigned8	ro		0
6000sub2	VAR	ReadStateByte2	Unsigned8	ro		0
6000sub3	VAR	ReadStateByte3	Unsigned8	ro		0
6000sub4	VAR	ReadStateByte4	Unsigned8	ro		0
6000sub5	VAR	ReadStateByte5	Unsigned8	ro		0
6000sub6	VAR	ReadStateByte6	Unsigned8	ro		0
6000sub7	VAR	ReadStateByte7	Unsigned8	ro		0
6000sub8	VAR	ReadStateByte8	Unsigned8	ro		0
6200	ARRAY	WriteState8OutputLines	Unsigned8	wr	O	
6200sub0	VAR	NumberOfElements	Unsigned8	wr		8
6200sub1	VAR	WriteStateByte1	Unsigned8	wr		0
6200sub2	VAR	WriteStateByte2	Unsigned8	wr		0
6200sub3	VAR	WriteStateByte3	Unsigned8	wr		0
6200sub4	VAR	WriteStateByte4	Unsigned8	wr		0
6200sub5	VAR	WriteStateByte5	Unsigned8	wr		0
6200sub6	VAR	WriteStateByte6	Unsigned8	wr		0
6200sub7	VAR	WriteStateByte7	Unsigned8	wr		0
6200sub8	VAR	WriteStateByte8	Unsigned8	wr		0

Currently these objects are not used for the transmission between the "Object Dictionaries" and the devices. They are reserved for future extensions. Nevertheless, these objects can be mapped in PDOs and changed using SDOs.

9 RS232/RS422 interface converter

The following converter is required for the connection from the PC with a RS232 interface to the RS422 on the option card AE-PSC:

Figure 9-1 RS232 / RS422 interface converter



Parts list for RS232 / RS422 converter

ID.	Component type
C1	1uF / 25V
C2	1uF / 25V
C3	1uF / 25V
C4	1uF / 25V
C5	100nF / 63V
C6	100nF / 63V
D1	ADM1485JN
D2	ADM1485JN
D3	LT 1181ACN
X1	WAGO733-168
X2	WAGO733-168