



Handbücher/Manuals



VIPA
Gesellschaft für Visualisierung
und Prozessautomatisierung mbH

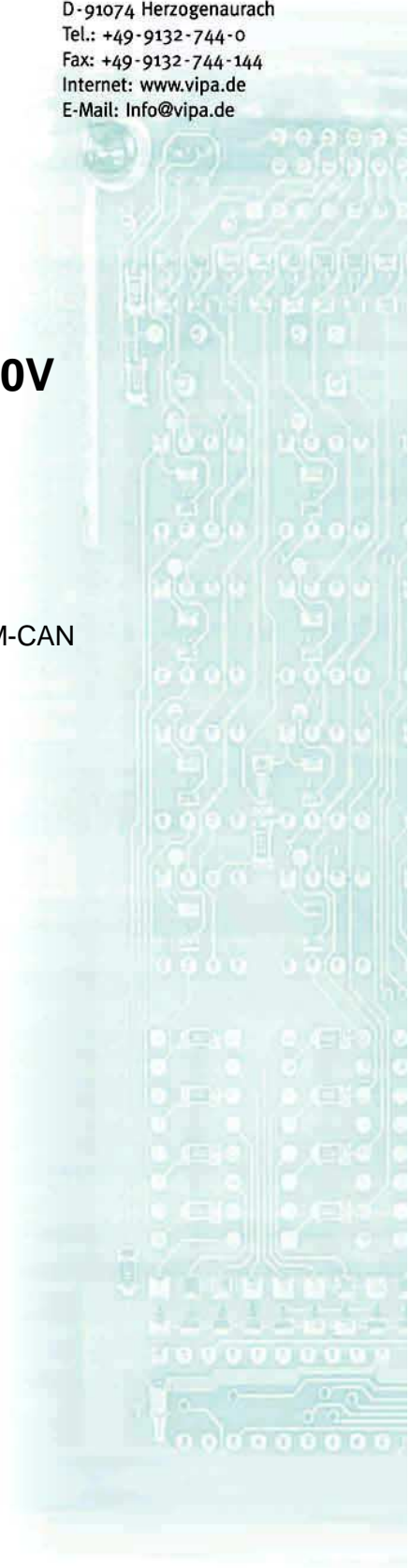
Ohmstraße 4
D-91074 Herzogenaurach
Tel.: +49-9132-744-0
Fax: +49-9132-744-144
Internet: www.vipa.de
E-Mail: Info@vipa.de

Manual

VIPA System 100V

SM-CAN

Order No.: VIPA HB100E_SM-CAN
Rev. 06/14



The information in this manual is supplied without warranties. Information is subject to change without notice.

© Copyright 2006 VIPA, Gesellschaft für Visualisierung und Prozessautomatisierung mbH
Ohmstraße 4, D-91074 Herzogenaurach,
Tel.: +49 (91 32) 744 -0
Fax.: +49 (91 32) 744-144
Email: info@vipa.de
<http://www.vipa.de>

Hotline: +49 (91 32) 744-114

All rights reserved.

Disclaimer of liability

The content of this manual was carefully examined to ensure that it conforms with the described hardware and software. However, discrepancies can not be avoided. The specifications in this manual are examined regularly and corrections will be included in subsequent editions. We gratefully accept suggestions for improvement.

Trademarks

VIPA, System 100V, System 200V, System 300V and System 500V are registered trademarks of VIPA Gesellschaft für Visualisierung und Prozessautomatisierung mbH.

SIMATIC, STEP and S7-300 are registered trademarks of Siemens AG.

Any other trademarks referred to in the text are the trademarks of the respective owner and we acknowledge their registration.

About this Manual

This manual describes the available System 100V decentral block periphery from VIPA. Besides of a product overview you will find the detailed description of the single components.

You'll get information about installing and operating decentral block periphery in a CAN system.

Outline

Chapter 1: Basics

This introduction includes recommendations on the handling of the modules of the VIPA System 100V as central resp. decentral automation system.

Besides a system overview you will find general information to the System 100V like dimensions, installation and operating conditions.

Chapter 2: Decentral block periphery CANopen

This chapter describes the decentral block periphery with integrated CAN bus coupler. It contains a fast introduction for the deployment under CANopen.

Contents

User considerations	1
Safety information	2
Chapter 1 Basics	1-1
Safety information for Users	1-2
Overview System 100V	1-3
General Description of the System 100V	1-4
Assembly dimensions	1-5
Chapter 2 Decentral block periphery CAN-Bus CANopen	2-1
Principles of CANopen	2-2
System overview	2-5
Structure	2-6
Installation and Cabling	2-12
Circuit diagrams	2-13
Deployment with CANopen	2-14
Fast introduction	2-15
Baudrate and module-ID settings	2-18
Message structure	2-19
PDO - Process Data Object	2-21
SDO - Service Data Object	2-24
Object directory	2-26
Emergency object	2-40
NMT - Network Management	2-41
Technical data	2-43
Appendix	A-1
Index	A-1

User considerations

Objective and contents	This manual describes the installation, project engineering and usage of the System 100V.
Target audience	The manual is targeted at users who have a background in automation technology and PLC programming.
Structure of the manual	This manual consists of chapters. Every chapter provides the description of one specific topic.
Guide to the document	This manual provides the following guides: <ul style="list-style-type: none">• An overall table of contents at the beginning of the manual• An overview of the topics for every chapter• An index at the end of the manual.
Availability	The manual is available in: <ul style="list-style-type: none">• printed form, on paper• in electronic form as PDF-file (Adobe Acrobat Reader)

Icons Headings

Important passages in the text are highlighted by following icons and headings:



Danger!

Immediate or likely danger.
Personal injury is possible.



Attention!

Damages to property is likely if these warnings are not heeded.



Note!

Supplementary information and useful tips.

Safety information

Application specifications

The System 100V is constructed and manufactured for

- communication and process control
- general control and automation tasks
- industrial applications
- operation within the environmental conditions specified in the technical data
- installation into a cubicle



Danger!

This device is not certified for applications in

- explosive environments (EX-zone)

Documentation

The manual must be available to all personnel in the

- project design department
- installation department
- commissioning
- operation



The following conditions must be met before using or commissioning the components described in this manual:

- Modification to the process control system should only be carried out when the system has been disconnected from power!
- Installation and modifications only by properly trained personnel
- The national rules and regulations of the respective country must be satisfied (installation, safety, EMC ...)

Disposal

National rules and regulations apply to the disposal of the unit!

Chapter 1 **Basics**

Outline Main theme of this chapter is to give you information and hints about deployment areas and usage of the System 100V.

Below follows a description of:

- Safety information for the user
- System overview
- Installation and environmental conditions

Content	Topic	Page
	Chapter 1 Basics	1-1
	Safety information for Users.....	1-2
	Overview System 100V	1-3
	General Description of the System 100V.....	1-4
	Assembly dimensions.....	1-5

Safety information for Users

Handling of electrostatically sensitive modules

VIPA modules make use of highly integrated components in MOS-technology. These components are extremely sensitive to over-voltages that can occur during electrostatic discharges.

The following symbol is attached to modules that can be destroyed by electrostatic discharges:



The symbol is located on the module, the module rack or on packing material and it indicates the presence of electrostatic sensitive equipment.

It is possible that electrostatic sensitive equipment is destroyed by energies and voltages that are far less than the human threshold of perception. These voltages can occur where persons do not discharge themselves before handling electrostatically sensitive modules and they can damage components thereby, causing the module to become inoperable or unusable. Modules that have been damaged by electrostatic discharges may fail after a temperature change, mechanical shock or changes in the electrical load.

Only the consequent implementation of protection devices and meticulous attention to the applicable rules and regulations for handling the respective equipment can prevent failures of electrostatically sensitive modules.

Shipping of electrostatically sensitive modules

Modules have to be shipped in the original packing material.

Measurements and alterations on electrostatically sensitive modules

When you are conducting measurements on electrostatically sensitive modules you should take the following precautions:

- Floating instruments must be discharged before use.
- Instruments must be grounded.

Modifying electrostatically sensitive modules you should only use soldering irons with grounded tips.



Attention!

Personnel and instruments should be grounded when working on electrostatically sensitive modules.

Overview System 100V

General

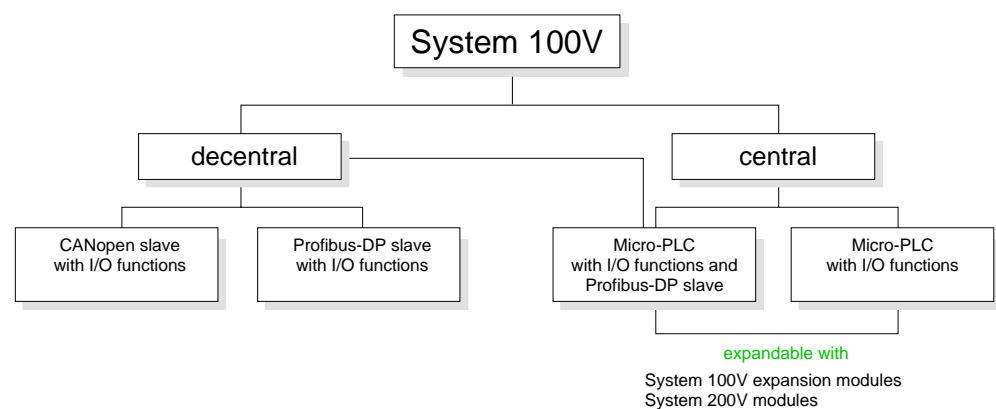
The System 100V from VIPA is a compact central and decentral usable automation system from VIPA. The system is recommended for lower and middle performance needs.

At a System 100V module, CPU res. bus coupler are integrated together with in-/output functions in one case.

System 100V modules are installed directly to a 35mm norm profile rail.

You may expand the number of I/Os of the Micro-PLC by means of up to 4 expansion modules res. connect System 200V modules via bus couplers.

The following picture shows the performance range of the System 100V:



Central system

The central system is built of one CPU and integrated I/O-functions. The CPU is instruction compatible to the S7-300 from Siemens and may be programmed and projected by means of S7 programming tools from Siemens and VIPA via MPI.

By means of bus couplers you may connect up to 4 modules of the System 200V family res. enlarge the number of I/Os by installing up to 4 System 100V expansion modules.

The CPUs are available in different variants.

Central system with DP slave

At the central system besides the CPU and I/O functions, a Profibus-DP slave is included that acknowledges itself within the address range of the CPU.

Decentral system

This system contains a Profibus-DP res. CANopen slave with I/O functions instead of the CPU. The system is not expandable.

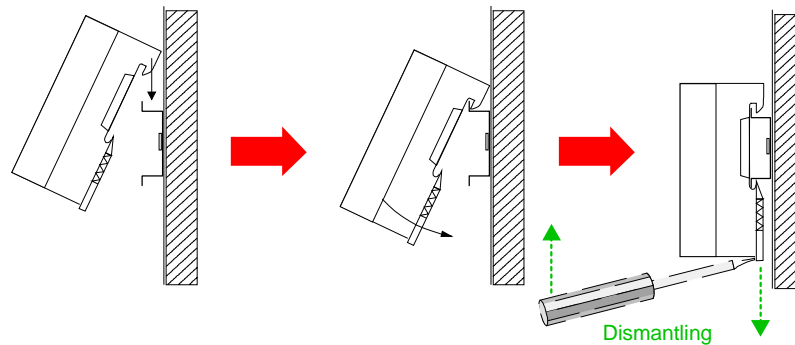
General Description of the System 100V

Structure and dimensions

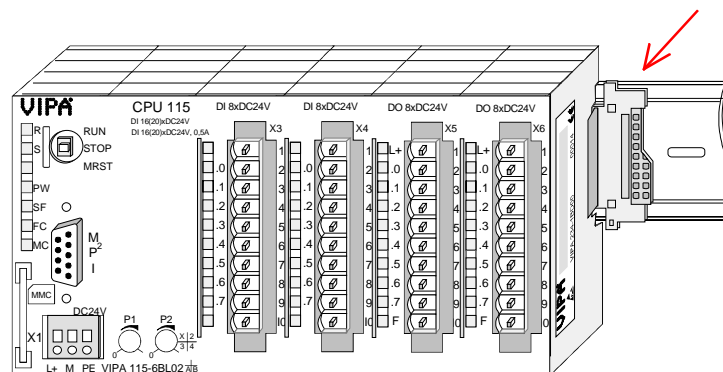
- Norm profile head rail 35mm
- Dimensions basic module:
4tier width: (WxHxD) in mm: 101.6x76x48 / in inches: 4x3x1.9
6tier width: (WxHxD) in mm: 152.4x76x48 / in Inches: 6x3x1.9

Installation

The installation of a System 100V module works via snapping on a norm profile head rail.



When using expansion modules, you have to clip the included 1tier bus connector at the right side to the module from behind before the installation.



Operation security

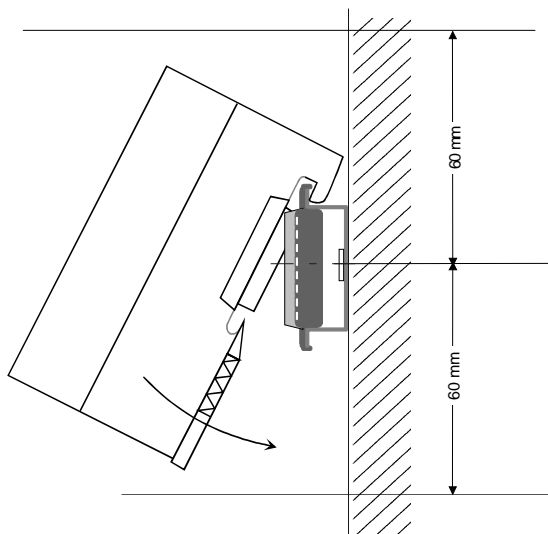
- Plug in via CageClamps, core cross-section 0.08...2.5mm²
- Total isolation of the cables during module changes
- EMV resistance ESD/Burst acc. IEC 61000-4-2 / IEC 61000-4-4 (to level 3)
- Shock resistance acc. IEC 60068-2-6 / IEC 60068-2-27 (1G/12G)

Environmental conditions

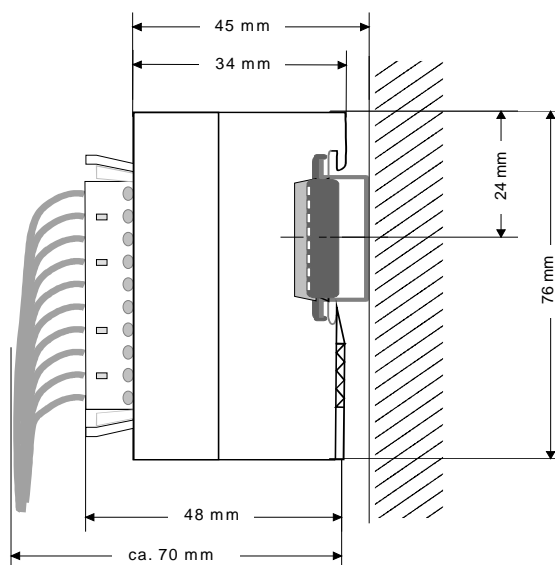
- Operating temperature: 0... + 60°C
- Storage temperature: -25... + 70°C
- Relative humidity: 5 ... 95% without condensation
- fan-less operation

Assembly dimensions

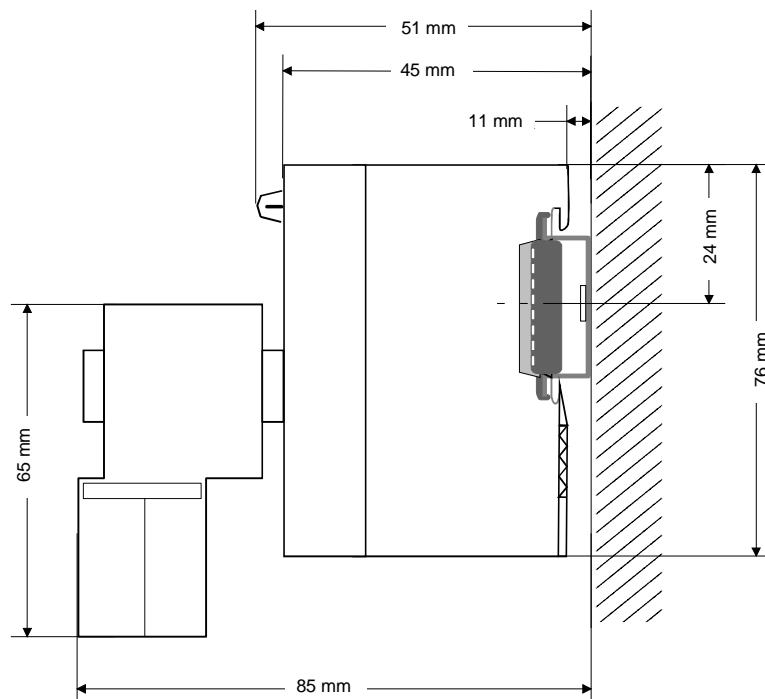
Installation dimensions



Installed and wired dimensions



**CPU with
EasyConn from
VIPA**



Chapter 2 Decentral block periphery CAN-Bus CANopen

Overview

The decentral block periphery consists of one CANopen slave and in-/output components. The CANopen block periphery is available in different variants described in the following chapter.

This chapter summarizes all information you need for assembly, project engineering and operating of this System 100V components.

Besides a fast step-in in the project engineering for "experts", this chapter contains an introduction of the telegram structure and the function codes of CANopen. Afterwards the principles of process and service data exchange are described, followed by the object directory, the parameter data and information about error control and maintenance.

The following chapter describes:

- General information about CAN-Bus
- Structure and assembly
- Circuit diagrams
- Object directory and description
- Parameter data and error control

Content

Topic	Page
Chapter 2 Decentral block periphery CAN-Bus CANopen	2-1
Principles of CANopen	2-2
System overview	2-5
Structure	2-6
Installation and Cabling	2-12
Circuit diagrams	2-13
Deployment with CANopen	2-14
Fast introduction	2-15
Baudrate and module-ID settings	2-18
Message structure	2-19
PDO - Process Data Object	2-21
SDO - Service Data Object	2-24
Object directory	2-26
Emergency object	2-40
NMT - Network Management	2-41
Technical data	2-43

Principles of CANopen

UL Certification

All the modules in this chapter have got the UL Certification:



UL-Recognition-Mark
Underwriters Laboratories (UL)
Standard UL 508, File Nr.: E234291

General

The CAN-Bus (**C**ontrol **A**rea **N**etwork) is an international open fieldbus standard for building, manufacturing and process automation and has been developed for automotive production.

Due to the wide range of error localization possibilities, the CAN-Bus is seen as the bus system with the highest security with a rest error probability of less than 4.7×10^{-11} . Defective messages are signaled and automatically transferred again.

In contrast to Profibus and Interbus, CAN-Bus also defines several 7-layer user profiles under the CAL-Layer-7- protocol (CAL=CAN application layer).

One of those user profiles is CANopen, which standardization is controlled by the CiA CAN in Automation e.V.

CANopen

CANopen is the user profile for industrial real-time systems and is at this time on its way through the manufacturing sites. CANopen has been published as Profile DS-301 from the CAN-User Organization (C.i.A). The communication profile DS-301 serves the standardization of the devices to ensure the compatibility of the products from different manufacturers. Another step is the device profile DS-401 that standardizes the device specific and the process data of the digital and analog in-/output modules.

CANopen consists of the communication profile that defines the objects used for the transfer of certain data, and the device profiles that define the kind of data transferred with the objects.

PDO and SDO

The CANopen communication profile bases on an object directory comparable to the Profibus directory. The communication profile DS-301 defines two kinds of objects and some special objects:

- Process data objects (PDO)
PDOs provide the transfer of real-time data
- Service data objects (SDO)
SDOs allow the read and write access to the object directory

Transfer medium

CAN is based on a line topology. With the help of a router node, it is possible to build-up a network structure. The number of participants per network is only limited by the efficiency of the used bus driver module.

The maximum net extension is limited by the signal run times. At 1Mbaud, for example, a net extension of 25m and at 50kbaud of 1000m is possible.

As transfer medium, CAN-Bus uses a screened 3core cable (5core optional).

The CAN-Bus is working with voltage differences. Thus it is less sensitive to interference than a voltage or current interface. The net should be configured as line with a 120Ω terminating resistor at the end.

Your VIPA CAN-Bus coupler provides a 9pin plug. This plug allows you to link-up the CAN-Bus coupler directly to the CAN-Bus net as slave.

All participants in the net are communicating with the same baudrate.

The bus structure allows the free installation or dismantling of station or the start-up step by step. Later expansions don't have any influence on stations that are already integrated. The system realizes automatically if one partner had a fail down or is new at the network.

Bus access procedure

Bus access methods are commonly divided into controlled (deterministic) and uncontrolled (random) bus access systems.

CAN employs a Carrier-Sense Multiple Access (CSMA) method, i.e. all stations have the same right to access the bus as long as the bus is not in use (random bus access).

Data communications is message related, not station related. Every message contains a unique identifier, which also defines the priority of the message. At any instance only one station can occupy the bus for a message.

CAN-Bus access control is performed by means of a collision-free, bit-based arbitration algorithm. Collision-free means that the final winner of the arbitration process does not have to repeat his message. The station with the highest priority is selected automatically when more than one station accesses the bus simultaneously. Any station that has information to send will delay the transmission if it detects that the bus is occupied.

Parameterization and configuration

For parameterization and configuration of the CANopen-Bus, the manufacturers of the CANopen master deliver a software configuration tool. This tool is able to access the object directory of your CAN slave and thus parameterize the coupler.

The system specific properties of your CANopen couplers are delivered in form of an EDS-file (electronic data sheet).

- Restrictions**
- max. 128 participants per net - max. 64 slaves per segment
 - max. net extension at 1MBaud 25m, at 50kBAud 1000m (depending on baudrate)
 - max. baudrate 1MBaud

Bus length

The maximum bus length with CAN is predominantly limited by the signal run time. The multi-master bus access procedure (arbitration) requires that the signals are applied at all nodes quasi simultaneously (before the scanning within one Bit time). For the signal run times of the CAN modules (transceiver, opto coupler, CAN controller) are nearly constant, the line length has to be adjusted to the baudrate:

Baudrate	Length
1MBit/s	25m
800MBit/s	50m
500kBit/s	100m
250kBit/s	250m
125kBit/s	500m
(50kBit/s)	1000m
(20kBit/s)	2500m

Diagnosis via Emergency object

The "Emergency" object serves the diagnosis. It is provided with a high priority and delivers information about the device and the net.

Addressing

Every bus participant identifies itself with an address. This address must be unique in this bus system and may range between 01... 99. The address is adjusted at the CAN-Bus coupler via the address adjuster.

EDS-file

To configure a slave connection in your projecting tool, you get the performance data of the components in the consignment in form of an EDS-file (electronic data sheet).

The EDS-files are:

VIPA_153_4CF00.EDS
VIPA_153_4CH00.EDS
VIPA_153_6CH00.EDS
VIPA_153_6CL10.EDS


Install your EDS-files in your projecting tool! More detailed information about the installation of the EDS-files is in the manual to your tool.

System overview

Decentral block periphery

The System 100V is an universal link between a fieldbus and the sensor/actuator level.

One System 100V unit consists of a CANopen-Bus coupler and a combination of in-/output channels and extension terminals.

Old order no.	 New order no.	Module width	Number of input DC 24V	Number of output DC 24V, 1A	Number of relay output DC 30V/AC 230V, 5A	Input data	Output data	Number of clamps	Current consumption module
	Digital in-/output								
123-4CF00	153-4CF00	4tier	tot. 8		-	1Byte	1Byte	2x11	55mA
123-4CH00	153-4CH00	4tier	8...12	4...8	-	2Byte	1Byte	-	55mA
123-6CH00	153-6CH00	6tier	8...12	4...8	-	2Byte	1Byte	4x11	55mA
123-6CL10	153-6CL10	6tier	24	8	-	3Byte	1Byte	-	55mA

Link-up to CANopen

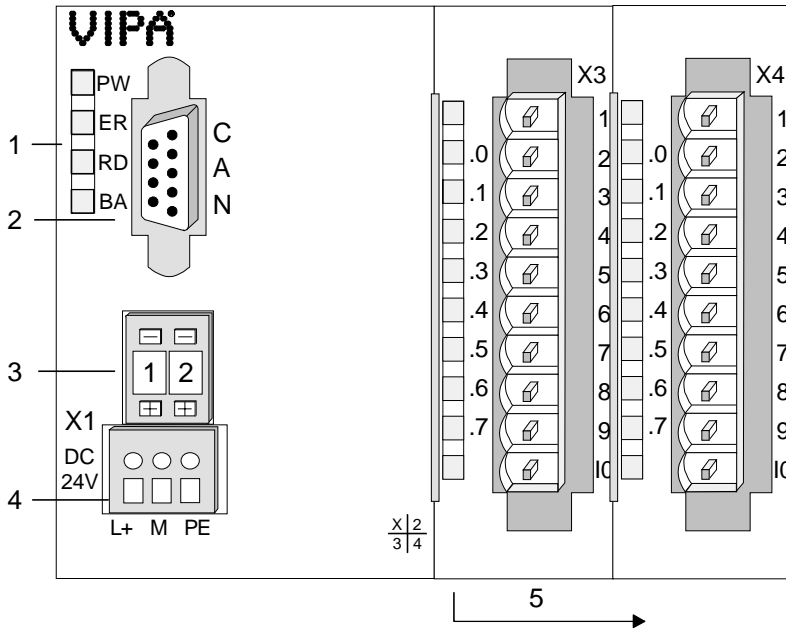
Your System 100V gets the link-up to CAN via the integrated CAN-Bus coupler.

CANopen system properties

- each 1 PDO for send and receive (PDO Linking)
- 2 SDOs as server
- Emergency object
- NMT
- Node Guarding
- Heartbeat

Structure

Front view



- [1] LED status monitoring
- [2] CANopen interface
- [3] Address selector
- [4] Power supply DC24V
- [5] In-/output periphery

Components

LEDs

The module provides four LEDs for status monitoring. The usage and the according colors of the diagnostic LEDs are shown in the following tables:

Label	Color	Description
PW	yellow	Signalizes an applying operating voltage
ER	red	On at error
RD	green	Blinks with 1Hz at positive self test and successful initialization
BA	yellow	Off at positive self test and successful initialization 1Hz blinking in state "Pre-Operational". On at state "Operational". 10Hz blinking in state "Prepared".

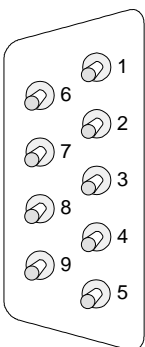
Status monitoring via LED combination

By combining the LEDs, several states may be shown:

- PW on
 - ER on
 - RD on
 - BA on
- Error in RAM- or EEPROM initialization
-
- PW on
 - ER blinks 1Hz
 - RD blinks 1Hz
 - BA blinks 1Hz
- Baudrate adjusting activated
-
- PW on
 - ER blinks 10Hz
 - RD blinks 10Hz
 - BA blinks 10Hz
- Error in CAN baudrate adjusting
-
- PW on
 - ER off
 - RD blinks 1Hz
 - BA off
- Module-ID adjusting activated

9pin D-Sub plug

The CAN-Bus coupler is linked-up to the CAN-Bus system via a 9pin plug. The following illustration shows the pin assignment of this interface:



Pin	Assignment
1	not used
2	CAN low
3	CAN Ground
4	not used
5	not used
6	optional Ground
7	CAN high
8	not used
9	not used

Address selector for baudrate and module-ID

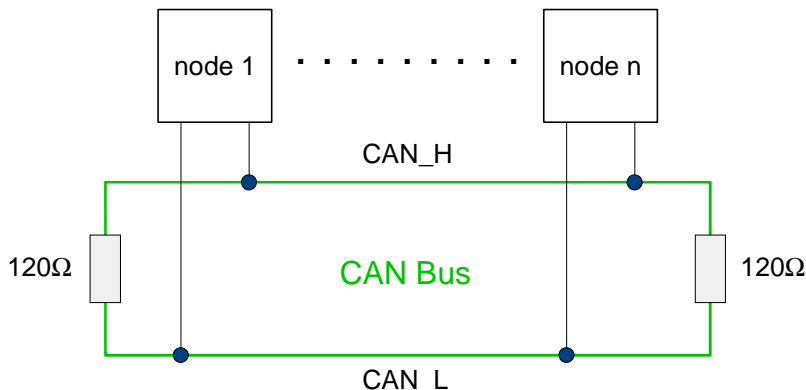
Via the address selector, you select the CAN baudrate as well as the module-ID.
 More detailed information is to find under "Baudrate and module-ID settings" in this chapter.

Power supply

The CAN-Bus coupler has an integrated mains power supply. It has to be provided with DC 24V.
 The mains power supply is protected against inverse polarity and over-current.

Cabling under CAN-Bus

CAN is a 2wire bus system where all participants are connected in parallel. The bus has to be terminated at both ends with a terminating resistor of 120 (res. 121) Ω to avoid reflections. This is also necessary for short cable lines.



For all CAN signals are monitored at the bus as difference levels, the CAN line is comparable less sensitive to interference (EMC). Due to the fact that always both lines have the same environment, the interference doesn't affect the difference level.



Note!

The bus cable has always to be terminated with a terminating resistor of 120 Ω to avoid reflections and therefore transfer problems!

Cable

For the CAN cabling, you use a screened twisted-pair cable (2x2) with a ripple resistor of 108...132 Ω .
 When the bus potential of the CAN receivers (CAN ground) is not connected, you can leave the second pair of cores. This is only convenient with small net extensions and common supply of all participants.

Input section

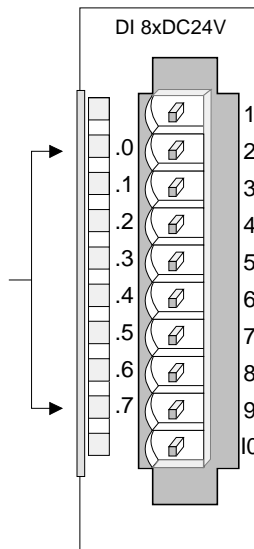
The digital input section a System 100V module collects the binary control signals of the process level and transmits them to the leading CANopen slave.

Every input channel shows its status via a green LED with a time delay of max. 3ms. The nominal input voltage is DC 24V. Hereby 0...5V are recognized as signal state "0" and 15...28.8V as signal state "1". More information about the installation of the input section is to find under "Circuit diagrams".

Status monitor pin assignment

LED Description

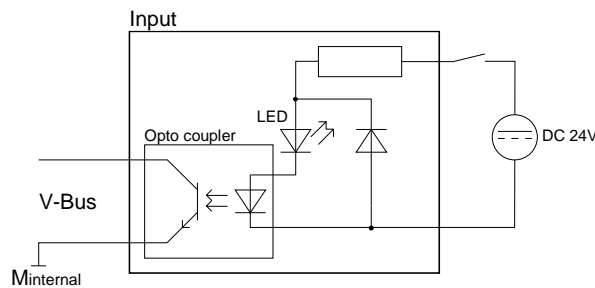
.0... .7 LEDs (green)
I+0.0 to I+0.7
starting with ca. 15V the signal "1" is recognized and the according LED is addressed



Pin Assignment

Pin	Assignment
1	not used
2	Input I+0.0
3	Input I+0.1
4	Input I+0.2
5	Input I+0.3
6	Input I+0.4
7	Input I+0.5
8	Input I+0.6
9	Input I+0.7
10	Ground

Schematic diagram input section



Output section

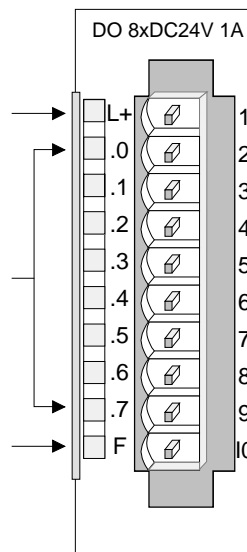
The digital output section collects the binary control signals of the leading CANopen slave and transmits them via the outputs to the process level. The output section has to be provided with additional external DC 24V via the front-facing connector. The applying supply voltage is monitored via the yellow LED (L+).

Every digital output channel shows its status via a green LED. At activated output, the concerning LED is on.

At overload, overheat or short circuit, the red error-LED labeled with "F" is on.

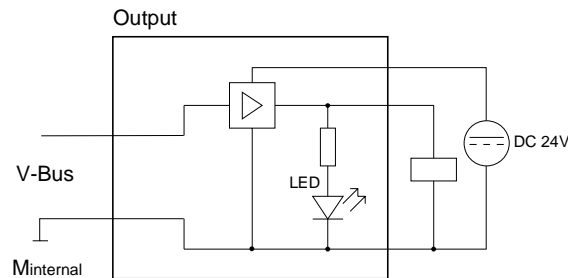
Status monitor pin assignment

LED	Description
L+	LED (yellow) Supply voltage is applied
.0... .7	LEDs (green) Q+0.0 to Q+0.7 as soon as an output is active, the according LED is addressed
F	LED (red) Error at overload, overheat or short circuit



Pin	Assignment
1	Supply voltage DC 24V
2	Output Q+0.0
3	Output Q+0.1
4	Output Q+0.2
5	Output Q+0.3
6	Output Q+0.4
7	Output Q+0.5
8	Output Q+0.6
9	Output Q+0.7
10	Supply voltage ground

Schematic diagram output section



In-/Output section

The In-/Output section has 4 I/O channels that may be used as input or as output channels and 4 normal outputs. Every I/O channel is provided with a diagnostic function, i.e. when an output is active the respective input is set to "1". When a short circuit occurs at the load, the input is held at "0" and the error is detectable by analyzing the input.

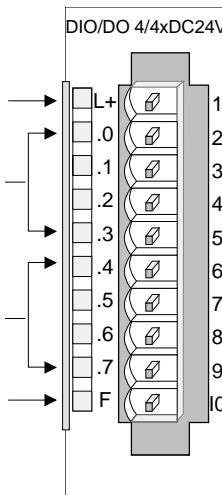
The In-/output section has to be additionally provided with DC 24V via the front-facing connector (see also schematic diagrams). The available supply voltage is shown via the yellow LED (L+).

Every digital in-/output channel shows its status via a green LED. At activated in-/output, the concerning LED is on.

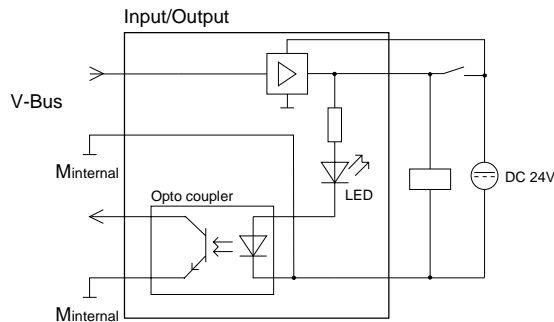
If an overload, overheat or short circuit occurs, the error-LED, marked with "F", is blinking red.

Status monitor pin assignment

LED	Description	Pin	Assignment
L+	LED (yellow) Supply voltage is available	1	Supply voltage DC 24V
.0... .3	LEDs (green) I/Q+0.0 to I/Q+0.3 as soon as an I/O=1 the according LED is addressed	2	In-/Output I/Q+0.0
		3	In-/Output I/Q+0.1
		4	In-/Output I/Q+0.2
		5	In-/Output I/Q+0.3
.4... .7	LEDs (green) Q+0.4 to Q+0.7 as soon as an output is active, the according LED is addressed	6	Output Q+0.4
		7	Output Q+0.5
		8	Output Q+0.6
		9	Output Q+0.7
F	LED (red) Error at overload, overheat or short circuits.	10	Supply voltage ground



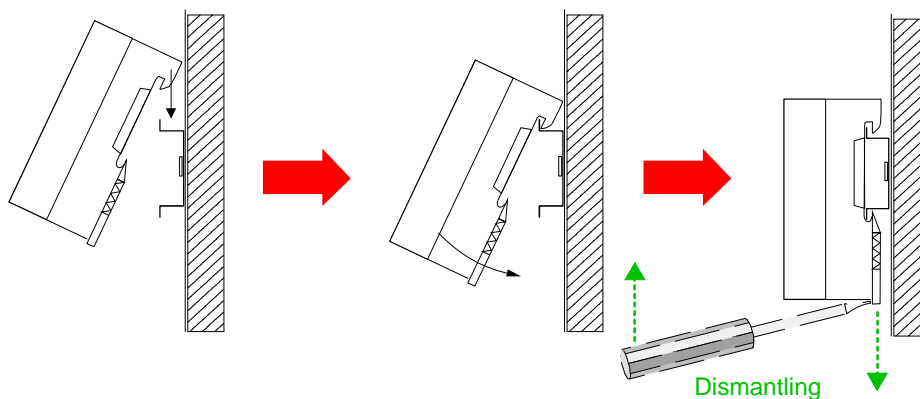
Schematic diagram output section



Installation and Cabling

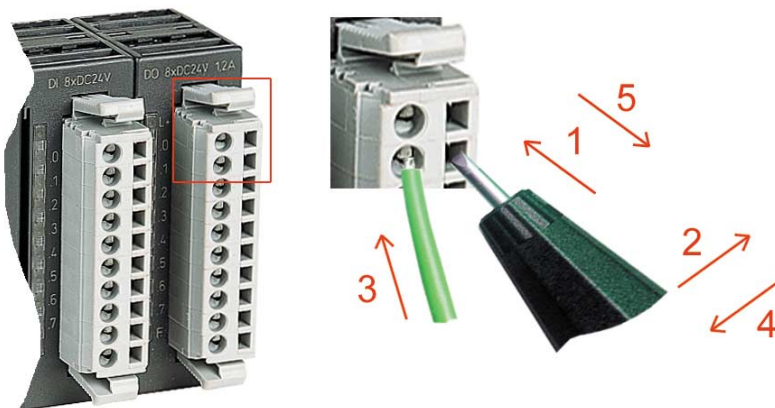
Assembly/ Dismantling

System 100V modules are clipped at a 35mm standard norm profile rail. For dismantling the modules, you have to pull down the locker with a screwdriver and lift the module from the profile rail.



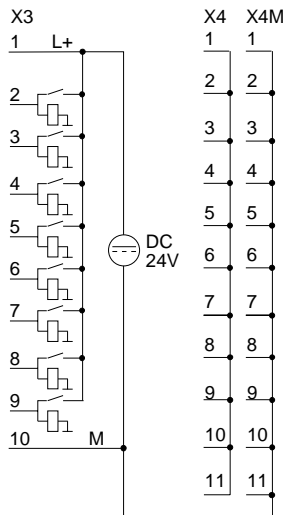
Cabling

Take a fitting screwdriver and push the cage clamp in the rectangular opening to the back, then insert the cable into the round opening. The cage clamp locks securely by removing the screwdriver.

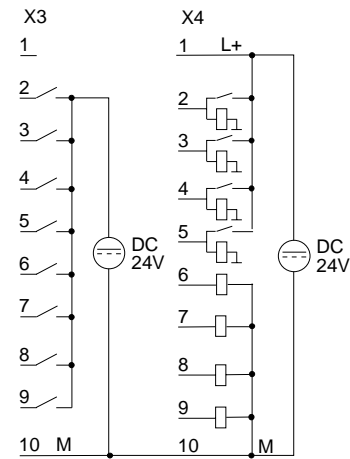


Circuit diagrams

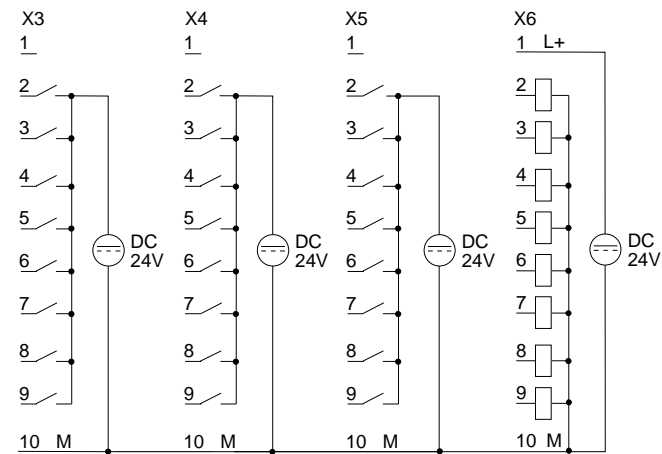
VIPA 153-4CF00



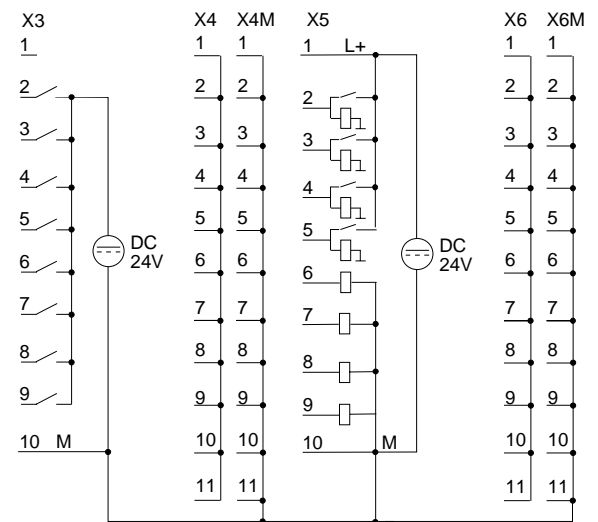
VIPA 153-4CH00



VIPA 153-6CL10



VIPA 153-6CH00



Deployment with CANopen

Overview

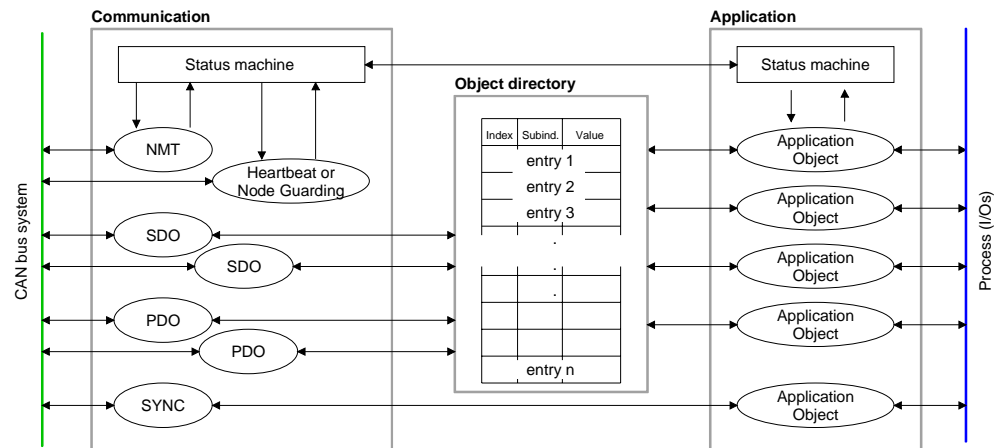
The data transfer between CANopen devices happens via data objects. The CANopen communication profile defines two standard object types (PDO and SDO) and some special objects (for network management purposes etc.).

The System 100V for CAN supports the following objects:

- 1 send-PDO, 1 receive-PDO, PDO Linking
- 2 standard-SDO (Server)
- 1 emergency object
- 1 synchronization object (SYNC without time stamp)
- Node Guarding / Heartbeat
- NMT

Structure of the device model

A CANopen device may be structured as follows:



Communication

Supports the communication data objects and the according functionality for the data transfer via the CANopen network.

Application

The application data objects contain e.g. the input and output data. An application status machine leads the outputs into a secure state in case of an error.

The object directory is organized as a 2 dimensional table. Data is addressed via index and subindex.

Object directory

This directory contains all data objects (application data + parameters) that are accessible and that influence communication, application and status machine behavior.

Fast introduction

Overview

This part of the manual points to experienced CANopen user that already know CAN. Here follows a quick introduction to the messages that are necessary for deploying the System 100V under CAN, using the default configuration.



Note!

Please regard, that in this manual the hexadecimal numbers are written in the "0x"-style usual for developers,
e.g.: **0x15AE = 15AEh**

Adjusting baudrate and module-ID

Via this address selector you fix a common CAN baudrate for the bus couplers and different node-IDs.

After turning on the power supply, you may specify the baudrate and the module-ID via 00 at the address selector within a period of 10s.

Further information is available under "Baudrate and module-ID settings" below.

CAN identifier

The CAN identifier for the I/O data of the System 100V are evaluated from the node addresses (1...63):

Data type	Default CAN identifier
Digital inputs 1 ... 24 Bit	0x180 (=384) + node address
Digital outputs 1 ... 8 Bit	0x200 (=512) + node address

Digital in-/outputs

The CAN messages with digital input data are represented as follows:

Identifier 0x180 + node address + up to 3 Byte user data (depending on the module)

Identifier 11Bit	DI 0 8Bit	DI 1 8Bit	DI 2 8Bit
------------------	-----------	-----------	-----------

The CAN messages with digital output data are represented as follows:

Identifier 0x200 + node address + up to 1Byte user data

Identifier 11Bit	DO 0 8Bit
------------------	-----------

Node Guarding

For the System 100V works per default in the event triggered mode (no cyclic data exchange), the break down of one node is not necessarily recognized. Remedy provides the supervision of the nodes by means of a cyclic status request (Node Guarding).

Herefore a status telegram is requested cyclically via Remote-Transmit-Request (RTR): The telegram only consists of one 11Bit identifier:

Identifier 0x700 + node address

Identifier 11Bit

The System 100V node answers with a telegram that contains a status byte:

Identifier 0x700 + node address + status byte

Identifier 11Bit	Status 8Bit
-------------------------	--------------------

Bit 0 ... 6: node state

0x7F: pre-operational

0x05: operational

0x04: stopped res. prepared

Bit 7: Toggle-Bit, toggles after every sending

If you want the bus coupler to recognize a network master failure (Watchdog function), you have to set the guard time (object 0x100C) and the life time factor (object 0x100D) to values $\neq 0$.

(Reaction time at break-down: Guard-Time x Life Time Factor).

Heartbeat

The System 100V CANopen coupler also supports the Heartbeat Mode in addition to Node Guarding.

When a value is entered into index 0x1017 (Heartbeat Producer Time), the device status (operational, pre-operational,...) of the bus coupler is transferred by means of the COB identifier (0x700+module-Id):

Identifier 0x700+ node address + status byte

Identifier 11Bit	Status 8Bit
-------------------------	--------------------

The Heartbeat Mode is started automatically as soon as the index 0x1017 holds a value higher 0.

Emergency Object To notify internal device errors to other participants at the CANopen bus with high priority, the CAN-Bus coupler is provided with the emergency object.

The emergency message employs the **COB-Identifier** that is pre-set at boot-up in the variable 1014h of the object directory in hexadecimal representation: **0x80 + Module-ID**.

The emergency message length is always 8Byte. It contains:

Identifier 80h + node address + 8 user data byte

Identifier 11 Bit	EC0	EC1	EReg	Inf0	Inf1	Inf2	Inf3	Inf4
-------------------	-----	-----	------	------	------	------	------	------

Error Code	Meaning	Info 0	Info 1	Info 2	Info 3	Info 4
0x0000	Reset Emergency					
0x1000	PDO Control	0xFF	0x10	PDO Number	LowByte Timer Value	HighByte Timer Value
0x8100	Heartbeat Consumer	Node ID	LowByte Timer Value	HighByte Timer Value	0x00	0x00
0x8100	SDO Block Transfer	0xF1	LowByte Index	HighByte Index	SubIndex	0x00
0x8130	Node Guarding Error	LowByte GuardTime	HighByte GuardTime	LifeTime	0x00	0x00
0x8210	PDO not processed due to length error	PDO Number	Wrong length	PDO length	0x00	0x00
0x8220	PDO length exceeded	PDO Number	Wrong length	PDO length	0x00	0x00



Note!

With the messages described here, you are now able to start and to stop your System 100V, read inputs, write outputs and control the modules.

In the following, all functions are once more described in detail.

Baudrate and module-ID settings

Overview

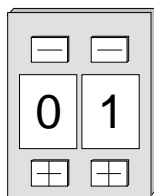
You have the option to specify the baudrate and the module-ID by setting the address selector to 00 within a period of 10s after you have turned on the power supply.

The selected settings are saved permanently in an EEPROM and can be changed at any time by means of the procedure shown above.

Adjusting the baudrate at the address selector

- Set the address selector to 00.
- Turn on the power to the CAN-Bus coupler

The LEDs ER, RD, and BA will blink at a frequency of 1Hz. For a period of 5s you can now enter the CAN-Baud rate means of the address selector:



Address selector	CAN baudrate	guaranteed max. bus length
"00"	1Mbaud	25m
"01"	500kBaud	100m
"02"	250kBaud	250m
"03"	125kBaud	500m
"04"	100kBaud	600m
"05"	50kBaud	1000m
"06"	20kBaud	2500m
"07"	10kBaud	5000m
"08"	800kBaud	50m

After 5 seconds the selected CAN baudrate is saved in the EEPROM.

The LEDs ER and BA are extinguished and the green RD-LED still blinks.

Now you have another 5s for adjusting the module-ID.

Selecting module-ID

- Define the module-ID in a range between 01...99 by means of the address selection switch. Every module-ID may only exist once on the bus. The module-ID must be defined before the bus coupler is turned on.

The entered module-ID's are accepted when a period of 5s has expired after which the bus coupler returns to the normal operating mode (status: "Pre-Operational").

Adjusting the baudrate via SDO-Write

You can also modify the CAN baudrate by means of an SDO write operation to the object "0x2001". The entered value is used as the CAN baudrate when the bus coupler has been RESET. This method is most convenient when you must change the CAN baudrate of all the bus couplers of a system from a central CAN terminal. The bus couplers use the programmed baudrate when the system has been RESET.

Message structure

All CANopen messages have got the following structure:

Identifier

Byte	Bit 7 ... Bit 0
1	Bit 3 ... Bit 0: most significant 4 bits of the module-ID Bit 7 ... Bit 4: CANopen function code
2	Bit 3 ... Bit 0: data length code (DLC) Bit 4: RTR-Bit: 0: no data (request code) 1: data available Bit 7 ... Bit 5: Least significant 3 bits of the module-ID

Data

Byte	Bit 7 ... Bit 0
3 ... 10	Data

An additional division of the 2Byte identifier into function portion and a module-ID gives the difference between this and a level 2 message. The function determines the type of message (object) and the module-ID addresses the receiver.

CANopen devices exchange data in the form of objects. The CANopen communication profile defines two different object types as well as a number of special objects.

The VIPA CAN bus coupler supports the following objects:

- 1 Transmit-PDO
- 1 Receive-PDO
- 2 SDOs
- 1 Emergency Object
- 1 Network management object NMT
- Node Guarding
- Heartbeat

Every object is associated with a function code. You can obtain the required function code from the following table!

**CANopen
function codes**

The following table lists the defined CANopen objects and the function codes that are supported by the VIPA CAN bus coupler:

Object	Function code (4 bits)	Receiver	Definition	Function
NMT	0000	Broadcast	CiA DS-301	Network managem.
EMERGENCY	0001	Master	CiA DS-301	Error message
PDO1S2M	0011	Master, Slave (RTR)	CiA DS-301	Digital input data 1
PDO1M2S	0100	Slave	CiA DS-301	Digital output data 1
SDO1S2M	1011	Master	CiA DS-301	Configuration data
SDO1M2S	1011	Slave	CiA DS-301	Configuration data
SDO2S2M	*	Master	Application spec.	Configuration data
SDO2M2S	*	Slave	Application spec.	Configuration data
Node Guarding	1110	Master, Slave (RTR)	CiA DS-301	Module monitoring
Heartbeat	1110	Master, Slave	Application spec.	Module monitoring

*) The function code for SDO2 is to find in the object directory 0x1201.

**Note!**

A detailed description of the structure and the contents of these objects is available in "CiA Communication Profile DS-301 Version 3.0" and "CiA Device Profile for I/O-Modules DPS 401 Version 1.4".

PDO - Process Data Object

PDO

For the process data exchange there are 2 **process data objects** (PDO) available. Every PDO exists of max. 8 data bytes. PDOs are transferred unacknowledged since the CAN protocol guarantees the transfer.

For input data 1 transmit PDO is at disposal and for output data 1 receive PDO.

In many fieldbus systems every time the whole process image is transferred - more or less cyclically. CANopen is not reduced to this communication principle, for CAN offers other possibilities due to the multi master bus access control.

With CANopen, the process data is divided in segments of max. 8 byte. These segments are called process data objects (PDOs). The PDOs correspond each to one CAN message and are assigned via those specific CAN identifier and defined in their priority.

The PDOs are named from the view of the bus coupler: Receive-PDOs (RxPDOs) are received by the coupler and contain output data.

Transmit-PDOs (TxPDOs) are sent by the coupler and contain input data. For the process data exchange there is each 1 RxPDO and 1 TxPDO available. The occupancy of these PDOs with input res. output data occurs automatically.

PDO Mapping

The PDO-Mapping is fixed and must not be altered.

PDO Identifier COB-ID

The most important communication parameter of a PDO is the CAN identifier (also called Communication Object Identifier, COB-ID). It identifies the data and specifies their priority at the bus access.

For every CAN data telegram there may be only one sending node (producer), but due to CAN, where the messages are sent in a broadcasting procedure, a telegram may be received from that many nodes you want (consumer). Thus one node may place its input information at the disposal of several bus participants at the same time - even without the passing on by a logical bus master.

In the System 100V for the send and receive PDOs default identifiers in dependence on the node address are provided.

The COB identifiers for the receive and send PDO transfers, fixed after the boot-up, are listed below.

The transmission type is set in the object directory (Indices 0x1400-0x1404 and 0x1800-0x1804, Subindex 0x02) fix to asynchronous and event triggered (=0xFF). Via the EVENT-Timer (Value * 1ms) the PDOs may be transferred cyclically.

Send: 0x180 + module-ID: PDO1S2M Digital (acc. DS-301)

Receive: 0x200 + Module-ID: PDO1M2S Digital (acc. DS-301)

PDO Linking

If the consumer–producer model of the CANopen PDOs shall be used for the direct data transfer between nodes (without master), you have to adjust the identifier allocation, so that the TxPDO identifier of the producer is fitting with the RxPDO identifier of the consumer:

This procedure is called PDO linking. It enables the easy structure of electronic gearing, where several slave axis follow the actual value in TxPDO of the master axis at the same time.

PDO communication types

CANopen supports the following possibilities for process data exchange:

- event triggered
- polled
- synchronized

Event triggered

The "event" is the change of an input value, the data is sent directly after an alteration. Using the event triggering, the bus spread is utilized best, because you do not transfer the whole process image all the time but only the alterations of it. At the same time, a short reaction time is received, because you don't have to wait for a request of another master after the change of an input value.

Polled

The PDOs may also be polled by data request telegrams (Remote Frames). Thus allows that e.g. the input process image at event triggered inputs may be brought on the bus also without input changes, for example a monitor or diagnostic device linked up to the network during runtime.

The VIPA CANopen bus couplers support the request of PDOs via remote frames - for this is not presupposeable due to the hardware, this communication type is only recommended partially.

Synchronized

It is convenient not only for drive applications to synchronize the evaluation of the input information as well as the setting of the outputs. For this, CANopen supports the SYNC object, a CAN telegram of high priority without user data, which reception is used as trigger for the reading of the inputs res. the setting of the outputs by the synchronized nodes.

PDO transmission type

The parameter "PDO transmission type" fixes, how the sending of the PDOs is initialized res. how received PDOs are handled:

Transmission Type	Cyclical	Acyclical	Synchronous	Asynchronous
0		x	x	
1-240	x		x	
254, 255				x

Synchronous

The transmission type 0 is only convenient for RxPDOs: The PDO is only evaluated after the receipt of the next SYNC telegram.

At the transmission types 1-240, the PDO is sent res. expected cyclically: after every n^{th} SYNC ($n=1\dots240$). For the transmission type may not only be combined in the network but also at one coupler, you may thus define e.g. a fast cycle for the digital inputs ($n=1$), while the data of the analog inputs are transmitted in a slower cycle (e.g. $n=10$). The cycle time (SYNC-Rate) may be watched (object 0x1006), the coupler then switches its outputs to error state at SYNC break-down.

Asynchronous

The transmission types 254 + 255 are asynchronous or also event triggered. At the transmission type 254 the event is proprietary, at 255 it is defined in the device profile.

When choosing the event triggered PDO communication, you have to regard, that there may be a lot of events at the same time and therefore according delay times may occur until a low priority PDO can be sent.

You have also to avoid, that a often changing input with high PDO priority is blocking the bus ("babbling idiot").

Inhibit time

Via the parameter "Inhibit time" you may activate a "Send filter", that doesn't lengthen the reaction time of the relatively first input alteration, but is active at the directly following changes.

The inhibit time (send delay time) describes the time span you have minimum to wait between the sending of two identical telegrams.

If you use the inhibit time, you may evaluate the maximum bus load and therefore the latent time in a "worst case" scenario.

SDO - Service Data Object

For accesses on the object directory the **service data object (SDO)** is used. With the help of the SDO you may access the object directory for reading or writing. In the CAL-Layer-7-Protocol you find the specification of the Multiplexed-Domain-Transfer-Protocol, that is used by the SDOs. With this protocol you may transfer messages of undefined length. Hereby messages may be divided in several CAN messages with identical identifier (segmentation) if necessary.

In the first CAN message of the SDO 4 of the 8 bytes are occupied by protocol information. For access on object directory entries with a length up to 4 bytes, a single CAN message is sufficient. For data lengths more than 4 bytes a segmented transmission takes place. The following segments of the SDO contain up to 7 bytes user data. The last byte contains an end identifier. A SDO is transmitted confirmed, i.e. every receipt of a message is acknowledged.

The COB identifier for read and write access are:

- Receive-SDO1: 0x600 + Module-ID
- Transmit-SDO1: 0x580 + Module-ID
- Receive-SDO2: 0x640 + Module-ID
- Transmit-SDO2: 0x5C0 + Module-ID



Note!

A more detailed description of the SDO telegrams is to find in the DS-301 norm from CiA.

In the following we will describe only those error messages that are thrown at a wrong parameter communication.

SDO Error-Codes

0x05030000	Toggle bit not alternated
0x05040000	SDO protocol timed out
0x05040001	Client/server command specifier not valid or unknown
0x05040002	Invalid block size (block mode only)
0x05040003	Invalid sequence number (block mode only)
0x05040004	CRC error (block mode only)
0x05040005	Out of memory
0x06010000	Unsupported access to an object
0x06010001	Attempt to read a write only object
0x06010002	Attempt to write a read only object
0x06020000	Object does not exist in the object dictionary
0x06040041	Object cannot be mapped to the PDO
0x06040042	The number and length of the objects to be mapped would exceed PDO length
0x06040043	General parameter incompatibility reason
0x06040047	General internal incompatibility in the device
0x06060000	Access failed due to a hardware error
0x06070010	Data type does not match, length of service parameter does not match
0x06070012	Data type does not match, length of service parameter too high
0x06070013	Data type does not match, length of service parameter too low
0x06090011	Sub-index does not exist
0x06090030	Value range of parameter exceeded (only for write access)
0x06090031	Value of parameter written too high
0x06090032	Value of parameter written too low
0x06090036	Maximum value is less than minimum value
0x08000000	general error
0x08000020	Data cannot be transferred or stored to the application
0x08000021	Data cannot be transferred or stored to the application because of local control
0x08000022	Data cannot be transferred or stored to the application because of the present device state
0x08000023	Object dictionary dynamic generation fails or no object dictionary is present (e.g. object dictionary is generated from file and generation fails because of an file error)

Object directory

Structure

In the CANopen object directory all CANopen objects relevant for the bus coupler are listed.

Every entry in the object directory is marked by a 16-Bit-Index.

If an object exists of several components (e.g. object type Array or Record), the components are marked by a 8-Bit-Subindex.

The object name describes the functionality of the object.

The data type attribute specifies the data type of the entry.

Via the access attribute it is specified, if an entry may only be read, only be written or read and written.

The object directory is divided into the following 3 areas:

Communication specific profile range (0x1000 – 0x1FFF)

This area contains the description of all specific parameters for the communication.

0x1000 – 0x1011 General communication specific parameters (e.g. the device name).

0x1400 – 0x140F Communication parameters of the Receive-PDOs (e.g. identifier)

0x1600 – 0x160F The mapping parameters of the Receive-PDOs. The mapping parameters contain the cross-references to the application objects, that are mapped into the PDOs, and the data width of the according object.

0x1800 – 0x180F
0x1A00 – 0x1A0F Communication and mapping parameters of the Transmit-PDOs

Manufacturer specific profile range (0x2000 – 0x5FFF)

Here you find the manufacturer specific entries like e.g. PDO Control, CAN baudrate (Baudrate after RESET) etc.

Standardized device profile range (0x6000 – 0x9FFF)

This range contains the objects for the device profile according to DS-401.



Note!

For the CiA Norms are only available in English, the table entries of the objects are in English too, to avoid misunderstandings.

A more detailed description of the table entries is to find below each table.

Object directory overview

Index		Content of Object
0x1000		Device type
0x1001		Error register
0x1003		Error store
0x1004		Number of PDOs
0x1005		SYNC identifier
0x1006		SYNC interval
0x1008		Device name
0x1009		Hardware version
0x100A		Software version
0x100B		Node number
0x100C		Guard time
0x100D		Life time factor
0x100E		Node Guarding Identifier
0x1010	X	Save parameter
0x1011	X	Load parameter
0x1014		Emergency COB-ID
0x1016	X	Heartbeat consumer time
0x1017	X	Heartbeat producer time
0x1018		Device identification
0x1029		Error behavior
0x1400	X	Communication parameter for Receive-PDOs (RxPDO, Master to Slave)
0x1600		Mapping parameter for Receive-PDOs (RxPDO)
0x1800	X	Communication parameter for Transmit-PDOs (TxPDO, Slave to Master)
0x1A00		Mapping parameter for Transmit-PDOs (TxPDO)
0x2001		CAN-Baudrate
0x2100		Kill EEPROM
0x2400	X	PDO Control
0x6000		Digital-Input-8-Bit Array (see DS 401)
0x6002	X	Polarity Digital-Input-8-Bit Array (see DS 401)
0x6100		Digital-Input-16-Bit Array (see DS 401)
0x6102		Polarity Digital-Input-16-Bit Array (v DS 401)
0x6120		Digital-Input-32Bit Array (see DS 401)
0x6122	X	Polarity Digital-Input-32-Bit Array (see DS 401)
0x6200		Digital-Output-8-Bit Array (see DS 401)
0x6202	X	Polarity Digital-Output-8-Bit Array (see DS 401)
0x6206	X	Fault Mode Digital-Output-8-Bit Array (see DS 401)
0x6207	X	Fault State Digital-Output-8-Bit Array (see DS 401)

X = save into EEPROM

Device Type

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1000	0	Device Type	Unsigned32	ro	N	0x00030191	Statement of device type

The 32Bit value is divided in two 16Bit fields:

MSB	LSB
Additional information Device	profile number
0000 0000 0000 wxyz (bit)	401dec=0x0191

The "Additional Information" contains information about the signal types of the I/O device:

z=1 means digital inputs,
y=1 means digital outputs,
x=1 means analog inputs,
w=1 means analog outputs.

VIPA 123-xxxxx = 0x0003 0191

Error register

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1001	0	Error Register	Unsigned8	ro	Y	0x00	Error register

Bit7							Bit0
ManSpec	reserved	reserved	Comm.	reserved	reserved	reserved	Generic

ManSpec.: Proprietary error, nearer specified in object 0x1003.

Comm.: Communication error (Overrun CAN)

Generic: A not nearer specified error occurred (Flag is set at every error message)

Error store

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1003	0	Predefined error field (error store)	Unsigned8	ro	N	0x00	Object 0x1003 contains a description of the error that has occurred in the device - sub-index 0 has the number of error states stored
	1	Actual error	Unsigned32	ro	N		Last error state to have occurred

	10		Unsigned32	ro	N		A maximum of 10 error states

The Predefined Error Field is divided in two 16Bit fields:

MSB	LSB
Additional information	Error code

The Additional Code shows the error trigger (see also Emergency Object) and therefore a detailed error description.

New errors are always saved at subindex 1, all other subindices are incremented accordingly.

By means of writing a "0" to subindex 0, the complete error memory is deleted. If there were no error since power-on, the object 0x1003 only exists of a subindex 0 with entered "0".

By means of a reset or power cycle the error memory is deleted.

Number of PDOs

Index	Sub index	Name	Type	Attr.	Map.	Default value	Meaning
0x1004	0	Number of PDOs supported	Unsigned32	ro	N	0x00010001	Number of PDOs supported
	1	Number of synchronous PDOs supported	Unsigned32	ro	N	0x00010001	Number of synchronous PDOs supported
	2	Number of asynchronous PDOs supported	Unsigned32	ro	N	0x00010001	Number of asynchronous PDOs supported

The 32Bit value is divided in two 16Bit fields:

MSB	LSB
Number of receive (Rx)PDOs supported	Number of send (Rx)PDOs supported

SYNC identifier

Index	Sub index	Name	Type	Attr.	Map.	Default value	Meaning
0x1005	0	COB-Id sync message	Unsigned32	ro	N	0x80000080	Identifier of the SYNC message

The lower 11 bit of the 32-Bit value contain the identifier (0x80=128dez), the MSBit shows if the device receives the SYNC telegram (1) or not (0).

Attention: In opposite to the PDO identifiers the set MSB signalizes that this identifier is relevant for the node.

SYNC interval

Index	Sub index	Name	Type	Attr.	Map.	Default value	Meaning
0x1006	0	Communication cycle period	Unsigned32	rw	N	0x00000000	Maximum length of the SYNC interval in μ s.

If a value different from Zero is entered here, the coupler switches into error state, when it doesn't receive a SYNC telegram during the "Watchdog time" at synchronous PDO operation.

Synchronous Window Length

Index	Sub index	Name	Type	Attr.	Map.	Default value	Meaning
0x1007	0	Synchronous window length	Unsigned32	rw	N	0x00000000	Contains the length of time window for synchronous PDOs in μ s.

Device name

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1008	0	Manufacturer device name	Visible string	ro	N		Device name of the bus coupler

VIPA 153-xxxxx = VIPA CANopen slave 153-4CF00
 VIPA CANopen slave 153-4CH00
 VIPA CANopen slave 153-6CH00
 VIPA CANopen slave 153-6CL10

For the returned value is larger than 4Byte, the segmented SDO protocol is used for transmission.

Hardware version

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1009	0	Manufacturer Hardware version	Visible string	ro	N		Hardware version number of bus coupler

VIPA 153-xxxxx = 1.00

For the returned value is larger than 4Byte, the segmented SDO protocol is used for transmission.

Software version

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x100A	0	Manufacturer Software version	Visible string	ro	N		Software version number CANopen software

VIPA 153-xxxxx = 1.00

For the returned value is larger than 4Byte, the segmented SDO protocol is used for transmission.

Node number

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x100B	0	Node ID	Unsigned32	ro	N	0x00000000	Node number

The node number is supported for reasons of compatibility.

Guard time

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x100C	0	Guard time [ms]	Unsigned16	rw	N	0x0000	Interval between two guard telegrams set by the NMT master or configuration tool.

Life time factor

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x100D	0	Life time factor	Unsigned8	rw	N	0x00	Life time factor x guard time = life time (watchdog for life guarding)

If the node doesn't receive a guarding telegram during the life time, the node switches to error state. If life time factor and/or guard time are = 0, the node doesn't handle lifeguarding, but may however be monitored by the master (Node Guarding).

Guarding identifier

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x100E	0	COB-ID Guarding Protocol	Unsigned32	ro	N	0x000007xy, xy = node ID	Identifier of the guarding protocol

Save parameters

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1010	0	Store Parameter	Unsigned8	ro	N	0x01	Number of store Options
	1	Store all parameters	Unsigned32	ro	rw	0x01	Stores all (storable) Parameters

By writing the signature "save" in the ASCII-Code (hex-Code: 0x65766173) at Subindex 1, the current parameters are saved nonvolatile. (Byte sequence at the bus incl. SDO protocol: 0x23 0x10 0x10 0x01 0x73 0x61 0x76 0x65).

A successful storing procedure is acknowledged by the according TxSDO (0x60 in the first byte).



Note!

For the bus coupler is not able to send or receive CAN telegrams during the storing procedure, data may only be stored if the node is in the state pre-operational.

It is recommended to set the whole network in the state pre-operational before saving to avoid an buffer overflow.

Load default values

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1011	0	Restore parameters	Unsigned8	ro	N	0x01	Number of reset options
	1	Restore all parameters	Unsigned32	rw	N	0x01	Resets all parameters to their default values

By writing the signature "load" in the ASCII-Code (hex-Code: 0x64616663) at Subindex 1, all parameters are set back to default values (delivery state) **at the next start-up (Reset)** (Byte sequence at the bus incl. SDO protocol: 0x23 0x11 0x10 0x01 0x6C 0x6F 0x61 0x64).

By this means the default identifier for the PDOs are set active again.

Emergency COB-ID

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1014	0	COB-ID Emergency	Unsigned32	ro	N	0x00000080 + Node_ID	Identifier of the emergency telegram

Consumer Heartbeat Time

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1016	0	Consumer heartbeat time	Unsigned8	ro	N	0x05	Number of entries
	1		Unsigned32	rw	N	0x00000000	Consumer heartbeat time

Structure of the Consumer Heartbeat Time entry:

Bits	31-24	23-16	15-0
Value	Reserved	Node-ID	Heartbeat time
Encoded as	Unsigned8	Unsigned8	Unsigned16

As soon as you try to configure a consumer heartbeat time different from 0 for that node-ID, the node stops the SDO download and throws the error code 06040043h.

Producer Heartbeat Time

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1017	0	Producer heartbeat time	Unsigned16	rw	N	0x0000	Defines the cycle time of heartbeat in ms

Identity Object

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1018	0	Identity Object	Unsigned8	ro	N	0x04	Contains general Information about the device (number of entries)
	1	Vendor ID	Unsigned32	ro	N	0xAFFEAF00	Vendor ID
	2	Product Code	Unsigned32	ro	N	VIPA 153-4CF00 = 0x1534CF00 VIPA 153-4CH00 = 0x1534CA00 VIPA 153-6CL10 = 0x1536CB10 VIPA 153-6CH00 = 0x1536CA00	Product Code
	3	Revision Number	Unsigned32	ro	N		Revision Number
	4	Serial Number	Unsigned32	ro	N		Serial Number

Error Behavior

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1029	0	Error behavior	Unsigned8	ro	N	0x02	Number of error classes
	1	Communication Error	Unsigned8	ro	N	0x00	Communication error
	2	Manufacturer specific error	Unsigned8	ro	N	0x00	Manufacturer specific error

As soon as a device error is detected in the "operational" state, the module should change automatically to "pre-operational" state. Otherwise, if you for e.g. implemented Error behavior, the module may be configured to switch to "stopped" state at an error event.

The following error classes may be shown:

- 0 = pre-operational
- 1 = no state change
- 2 = stopped

Communication parameter RxPDO1

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1400	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x40000200 + NODE_ID	COB-ID RxPDO1
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

Subindex 1 (COB-ID): The lower 11 bit of the 32-bit value (Bits 0-10) contain the CAN-Identifier, the MSBit (Bit 31) shows, if the PDO is active (1) or not (0), Bit 30 monitors, if a RTR access to this PDO is allowed (0) or not (1).

The Subindex 2 contains the transmission type.

Mapping RxPDO1

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1600	0	Number of Elements	Unsigned8	ro	N	0x01	Mapping parameter of the first receive PDO; sub-index 0: number of mapped objects
	1	1st mapped object	Unsigned32	ro	N	0x62000108	(2 byte index, 1 byte sub-index, 1 byte bit-width)

The first receive-PDO (RxPDO1) is per default planned for digital outputs. Depending on the number of equipped outputs, the needed length of the PDO is evaluated automatically and the according objects are mapped.

For the digital outputs are organized byte by byte, the length of the PDO in bytes may be read directly in the subindex 0.

If the mapping is changed, you have to adjust the entry in the subindex 0 accordingly.

Communication parameter TxPDO1

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1800	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter of the first transmit PDO, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x00000180 + NODE_ID	COB-ID TxPDO1
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1ms]

Subindex 1 (COB-ID): The lower 11 bit of the 32-bit value (Bits 0-10) contain the CAN-Identifier, the MSBit (Bit 31) shows, if the PDO is active (1) or not (0), Bit 30 monitors, if a RTR access to this PDO is allowed (0) or not (1). The Subindex 2 contains the transmission type, Subindex 3 the repetition delay between two identical PDOs. If an event timer exists with an value different 0, the PDO is transmitted after expiration of this timer.

If an inhibit timer exists, the event is delayed for this time.

Mapping TxPDO1

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1A00	0	Number of Elements	Unsigned8	ro	N	VIPA 153-4CF00=0x01 VIPA 153-4CH00=0x02 VIPA 153-6CL10=0x03 VIPA 153-6CH00=0x02	Mapping parameter of the first transmit PDO; sub-index 0: number of mapped objects
	1	1st mapped object	Unsigned32	ro	N	0x60000108	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2nd mapped object	Unsigned32	ro	N	0x60000208	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	3	3rd mapped object	Unsigned32	ro	N	0x60000308	(2 byte index, 1 byte sub-index, 1 byte bit-width)

The first send-PDO (RxPDO1) is per default planned for digital inputs. Depending on the number of equipped inputs, the needed length of the PDO is evaluated automatically and the according objects are mapped. For the digital outputs are organized byte by byte, the length of the PDO in bytes may be read directly in the subindex 0.

If the mapping is changed, you have to adjust the entry in the subindex 0 accordingly.

CAN baudrate

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x2001	0	CAN-Baudrate	Unsigned8	rw	N	0x01	Setting CAN-Baudrate

This index entry writes a new baudrate into the EEPROM.

At the next start-up (Reset) the CAN coupler starts with the new baudrate.

Value	CAN baudrate
"00"	1MBaud
"01"	500kBaud
"02"	250kBaud
"03"	125kBaud
"04"	100kBaud
"05"	50kBaud
"06"	20kBaud
"07"	10kBaud
"08"	800kBaud

PDO-Control

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x2400	0	Number of Elements	Unsigned8	ro	N	0x0A	Time control for RxPDOs
	1	RxPDO1	Unsigned16	rw	N	0x0000	Timer value [ms]

As soon as the timer value differs from 0, the control starts. With every received RxPDO the timer is set back again.

As soon as the timer runs out, the CAN coupler switches to the state "pre-operational" and sends an emergency telegram.

8Bit digital inputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6000	0x00	8-bit digital input block	Unsigned8	ro	N	VIPA 153-4CF00=0x01 VIPA 153-4CH00=0x02 VIPA 153-6CL10=0x03 VIPA 153-6CH00=0x02	Number of available digital 8-bit input blocks
	0x01	1st input block	Unsigned8	ro	Y		1st digital input block
	0x02	2nd input block	Unsigned8	ro	Y		2nd digital input block
	0x03	3rd input block	Unsigned8	ro	Y		3rd digital input block

8Bit polarity digital inputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6002	0x00	8-bit digital input block	Unsigned8	ro	N	VIPA 153-4CF00=0x01 VIPA 153-4CH00=0x02 VIPA 153-6CL10=0x03 VIPA 153-6CH00=0x02	Number of available digital 8-bit input blocks
	0x01	1st input block	Unsigned8	rw	N	0x00	1st polarity digital input block
	0x02	2nd input block	Unsigned8	rw	N	0x00	2nd polarity digital input block
	0x03	3rd input block	Unsigned8	rw	N	0x00	3rd polarity digital input block

Individual negation of the input channels

1 = input inverted
0 = input not inverted

8Bit digital outputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6200	0x00	8-bit digital output block	Unsigned8	ro	N	0x01	Number of available digital 8-bit output blocks
	0x01	1st output block	Unsigned8	rw	Y		1st digital output block

8Bit change polarity digital outputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6202	0x00	8-bit digital output block	Unsigned8	ro	N	0x01	Number of available digital 8-bit output blocks
	0x01	1st output block	Unsigned8	rw	N	0x00	1st polarity digital output block

Individual negation of the output channels

1 = output inverted
0 = output not inverted

8Bit error mode digital outputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6206	0x00	8-bit digital output block	Unsigned8	ro	N	0x01	Number of available digital 8-bit output blocks
	0x01	1st output block	Unsigned8	rw	N	0xFF	1st error mode digital output block

With the help of this object you may define, if an output channel changes at an error event to a specified value, predefined in the object 0x6207.

1 = take the value in object 0x6207
0 = fix output value at error event

8Bit error value digital outputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6207	0x00	8-bit digital output block	Unsigned8	ro	N	0x01	Number of available digital 8-bit output blocks
	0x01	1st output block	Unsigned8	rw	N	0x00	1st error value digital output block

Presupposed that the error mode is activated, at an error event the here predefined value is taken.

1 = At error event output value to 0 as soon as object 0x6206 is activated.
0 = At error event output value to 1 as soon as object 0x6206 is activated.

Emergency object

Overview

To inform the other participants at the CANopen bus of internal device errors or CAN-Bus errors, the CANopen bus coupler supports the emergency object. It is provided with a high priority and delivers important information about the state of the device and the network.



Note!

It is highly recommended to analyze the emergency objects - they are an important source of information!

Telegram structure

The emergency telegram has always a length of 8Byte. It contains firstly the 2byte error code, then the 1byte error register and finally the additional code of 5byte length.

Error code low byte	Error code high byte	ErrorRegister Index 0x1001	Info 0	Info 1	Info 2	Info 3	Info 4
---------------------	----------------------	----------------------------	--------	--------	--------	--------	--------

Error messages

Error Code	Meaning	Info 0	Info 1	Info 2	Info 3	Info4
0x0000	Reset Emergency					
0x1000	PDO Control	0xFF	0x10	PDO Number	LowByte Timer Value	HighByte Timer Value
0x8100	Heartbeat Consumer	Node ID	LowByte Timer Value	HighByte Timer Value	0x00	0x00
0x8100	SDO Block Transfer	0xF1	LowByte Index	HighByte Index	SubIndex	0x00
0x8130	Node Guarding Error	LowByte GuardTime	HighByte GuardTime	LifeTime	0x00	0x00
0x8210	PDO not processed due to length error	PDO Number	Wrong length	PDO length	0x00	0x00
0x8220	PDO length exceeded	PDO Number	Wrong length	PDO length	0x00	0x00

NMT - Network Management

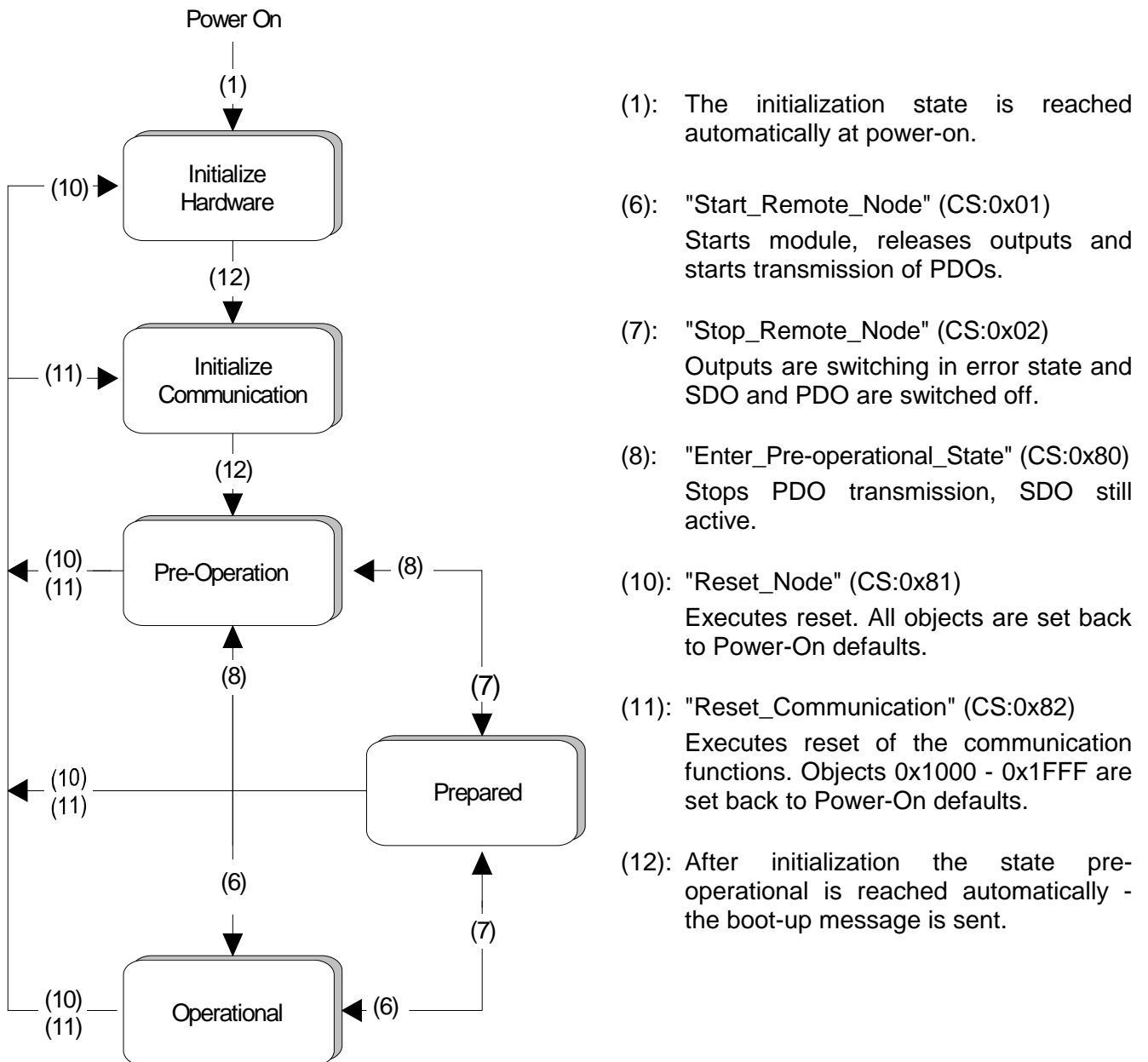
The network management (NMT) specifies global services for network monitoring and management. Together with the login and logout of single participants this also includes the control of the participants during operation and the handling of exceptions.

NMT-Service-Telegrams have the COB identifier 0x0000. An additive module-ID is not necessary. The length is always 2 data bytes.

The 1. data byte contains the NMT-Command Specifier: **CS**

The 2. data byte contains the module-ID (0x00 for a broadcast command).

The following illustration gives an overview over all CANopen status transitions and the concerning NMT command specifier "CS":



Node Guarding

The bus coupler supports the Node Guarding defined by CANopen, to guarantee the control of the bus participants.

The guarding operation of the module starts with the first guarding request telegram (RTR) received by the master. The assigned COB identifier is fixed in the object directory in the variable 0x100E at 0x700 + module-ID. If during the guarding operation the master doesn't receive further guarding request telegrams within the "guard time" (object 0x100C), the module assumes that the master is not working proper anymore. After the time, fixed by the multiplication of "guard time" (0x100C) and "Life-Time-Factor" (0x100D), the module automatically switches to the state "pre-operational".

If either the "guard time" (object 0x100C) or the and "Life-Time-Factor" (0x100D) is set to 0 by the master by means of a SDO download, the expiration of the guarding time is not proofed and the module stays in the current state.

Heartbeat

Besides of the node guarding the VIPA CAN coupler also supports the heartbeat mode.

If you are typing a value into the index 0x1017 (Heartbeat Producer Time), the device state of the bus coupler (operational, pre-operational, ...) is transmitted via COB identifiers(0x700+module-ID) as soon as the heartbeat timer expires.

The heartbeat mode starts automatically as soon as a value different from 0 is typed in.

Technical data

Decentral block periphery CAN-Bus

Electrical Data				
Voltage supply	DC 24V, via front with external mains power supply			
Current consumption	max. 55mA			
Status monitor	via LEDs at the front side			
Interfaces	9pin D-Sub (plug) CAN-Bus coupler			
CAN-Bus interface				
Interface	9pin D-Sub plug			
Network topology	Linear bus, active bus termination at one end, tap lines are possible.			
Medium	Screened 3core cable; depending on environmental conditions, the screening may be left.			
Transfer rate	10kBaud up to 1MBaud			
max. total length	without repeater 1000m at 50kBaud			
max. no. of participants	127 stations (depending on master)			
Digital in-/output	153-4CF00	153-4CH00	153-6CL10	153-6CH00
Number of inputs	0...8	8...12	24	8...12
Number of outputs	0...8	4...8	8	4...8
Input data	1Byte	2Byte	3Byte	2Byte
Output data	1Byte	1Byte	1Byte	1Byte
Input voltage at "1"	DC15...28.8V	DC15...28.8V	DC15...28.8V	DC15...28.8V
Input voltage at "0"	DC 0...5V	DC 0...5V	DC 0...5V	DC 0...5V
Delay time	3ms	3ms	3ms	3ms
Output current per channel	1A	1A	1A	1A
Module width	4	4	6	6
Number of terminals	2x11	-	-	4x11
Interface	RS 485; 9pin D-Sub plug			
Transfer rate	10kBaud up to 1MBaud			

Appendix

A Index

A	
Address selector	2-18
Assembly	2-12
B	
Basics	1-1
CAN-Bus.....	2-2
System 100V	1-3
Baudrate	2-18
Bus access	2-3
Bus length.....	2-4
C	
Cabling.....	2-8, 2-12
CANopen	2-2
Circuit diagrams.....	2-13
Components	2-6
Core cross-section.....	1-4
D	
Data objects.....	2-14
Deployment.....	2-14
Device model.....	2-14
Digital input	2-9
Digital output.....	2-10
D-Sub plug.....	2-7
E	
EDS	2-4
EDS-file.....	2-4
Emergency object.....	2-40
Emergency Object	2-17
Environmental conditions.....	1-4
Error codes	2-17
F	
Fast introduction	2-15
Function codes	2-20
H	
Hardware description.....	2-6
Heartbeat.....	2-16, 2-42
I	
Identifier.....	2-15
In-/output section.....	2-11
Inhibit time	2-23
Input section	2-9
Installation	2-12
L	
LEDs.....	2-6
M	
Message structure.....	2-19
Module-ID	2-18
N	
NMT.....	2-41
Node Guarding	2-42
O	
Object directory	2-26
Output section	2-10
P	
parameterization.....	2-3
PDO.....	2-2, 2-21
Power supply	2-8
R	
Restrictions.....	2-4
S	
Safety Information	1-2
SDO.....	2-2, 2-24
Status monitoring.....	2-7
Structure	2-6
System overview	2-5
System overview	1-3, 2-5
General description	1-4
T	
Technical data	2-43
W	
Wiring	2-12

