VIPA System 300V

IM | Manual

HB130E_IM | Rev. 09/46

November 2009

VIPA art of automation

VIPA, Gesellschaft für Visualisierung und Prozessautomatisierung mbH
Ohmstraße 4, D-91074 Herzogenaurach,Germany
Tel.: +49 (91 32) 744 -0
Fax.: +49 9132 744 1864
EMail: info@vipa.de
http://www.vipa.de

Note

Every effort has been made to ensure that the information contained in this document was complete and accurate at the time of publishing. Nevertheless, the authors retain the right to modify the information. This customer document describes all the hardware units and functions known at the present time. Descriptions may be included for units which are not present at the customer site. The exact scope of delivery is described in the respective purchase contract.

CE Conformity

Hereby, VIPA GmbH declares that the products and systems are in compliance with the essential requirements and other relevant provisions of the following directives:

- 2004/108/EC Electromagnetic Compatibility Directive
- 2006/95/EC Low Voltage Directive

Conformity is indicated by the CE marking affixed to the product.

Conformity Information

For more information regarding CE marking and Declaration of Conformity (DoC), please contact your local VIPA customer service organization.

Trademarks

VIPA, SLIO, System 100V, System 200V, System 300V, System 300S, System 400V, System 500S and Commander Compact are registered trademarks of VIPA Gesellschaft für Visualisierung und Prozessautomatisierung mbH.

SPEED7 is a registered trademark of profichip GmbH.

SIMATIC, STEP, SINEC, S7-300 and S7-400 are registered trademarks of Siemens AG.

Microsoft und Windows are registered trademarks of Microsoft Inc., USA.

Portable Document Format (PDF) and Postscript are registered trademarks of Adobe Systems, Inc.

All other trademarks, logos and service or product marks specified herein are owned by their respective companies.

Information product support

Contact your local VIPA Customer Service Organization representative if you wish to report errors or questions regarding the contents of this document. If you are unable to locate a customer service center, contact VIPA as follows:

VIPA GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Telefax:+49 9132 744 1204
EMail: documentation@vipa.de

Technical support

Contact your local VIPA Customer Service Organization representative if you encounter problems with the product or have questions regarding the product. If you are unable to locate a customer service center, contact VIPA as follows:

VIPA GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Telephone: +49 9132 744 1150/1180 (Hotline)
EMail: support@vipa.de

Contents

About this Manual

This manual describes the operation of the System 300V and the according available interface modules (IM). A short overview over the range of products is followed by a detailed description of the single modules. You will get information for connecting and operating the System 300V and the additional IM modules.

Overview

Chapter 1: Basics

This introduction includes recommendations on the handling of the modules of the VIPA System 300V and introduces you to central res. decentral automation systems.

Chapter 2: Installation and assembly guide lines

All information that you need for installation and cabling of a PLC with components of the System 300V.

Chapter 3: Profibus-DP

This chapter contains a description of Profibus applications for the System 300V. The text describes the configuration of the VIPA Profibus slave modules as well as a number of different communication examples.

Chapter 4: CANopen

This chapter deals with the VIPA CANopen slave IM 353CAN from VIPA and its deployment at CAN bus.

Objective and contents

This manual describes the interface modules (IM) which can be used at the System 300. It contains a description of construction, project implementation and application of the products as well as the technical data.

Target audience

The manual is targeted at users who have a background in automation technology.

Structure of the manual

The manual consists of chapters. Every chapter provides a self-contained description of a specific topic.

Guide to the document

The following guides are available in the manual:

- 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:

- 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

Applications conforming with specifications

The interface modules are constructed and manufactured for:

- System 300 components from VIPA and Siemens
- communication and process control
- general control and automation applications
- 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

- 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 an overview about the System 300V from VIPA. We will outline the possibilities of the installation of central res. decentral systems.

This chapter also contains general information about the System 300V like measurements, hints for installation and the environmental conditions.

Content

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 electrostatically sensitive equipment.

It is possible that electrostatically 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, damaged in this way, are normally not immediately recognized. The according error may occur only after a while of operation.

Modules that have been damaged by electrostatic discharges can 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 modules

Modules must 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.

General description of the System 300V

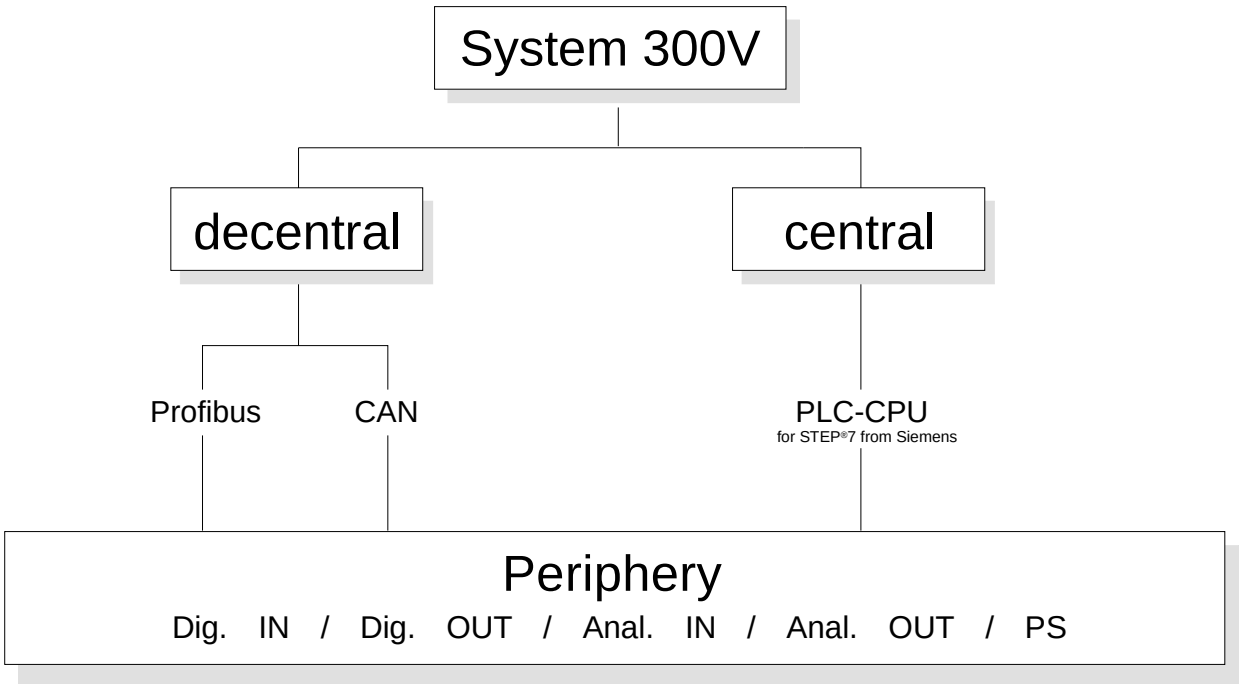
The System 300V

The System 300V is a modular automation system for middle and high performance needs, that you can use either distributed or non-distributed. The single modules are directly clipped to a 530 mm backplane and are connected together with the help of bus clips at the backside.

The single modules of the VIPA System 300V are design compatible to Siemens. Due to the compatible backplane bus it is no problem to mix the modules from VIPA and Siemens.

The CPUs of the System 300V are instruction set compatible to S7-300 from Siemens. The CPUs are programmed via the VIPA programming software WinPLC7 or the SIMATIC manager from Siemens or other available programming tools.

The following picture illustrates the performance range of the System 300V:



Components

Central system

The System 200V series consists of a number of PLC-CPUs. These are programmed in STEP®7 from Siemens. Herefore you may use WinPLC7 from VIPA or the SIMATIC manager from Siemens.

CPUs with integrated Ethernet interfaces or additional serial interfaces simplify the integration of the PLC into an existing network or the connection of additional peripheral equipment.

The application program is saved in Flash or an additional plug-in memory module.

Because of the automatic addressing, up to 32 peripheral modules can be called by the System 300V CPUs.

Decentral system

In combination with a Profibus DP master and slave the PLC-CPUs or the PC-CPU form the basis for a Profibus-DP network in accordance with DIN 19245-3.

The DP network can be configured with the hardware configurator from Siemens. Together with the hardware configuration you transfer your project into the CPU via MPI. Another component of the decentral system is the CAN-Slave. It allows the link-up to the fieldbus system CANopen.

Peripheral modules

A large number of peripheral modules are available from VIPA, for example digital as well as analog inputs/outputs.

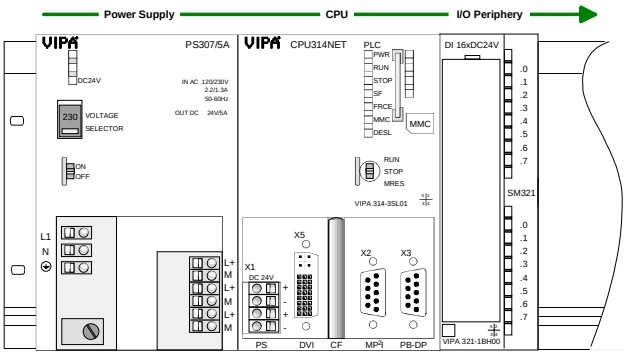
These peripheral modules can be deployed central as well as decentral.

Dimensions/ Weight

- Profile rail 530mm
- Peripheral modules with recessed labeling
- Dimensions of the basic enclosure:
  1tier width: (WxHxD) in mm: 40x125x120
  2tier width: (WxHxD) in mm: 80x125x120
  3tier width: (WxHxD) in mm: 120x125x120

Installation

Please regard that the power supply and header modules like CPUs and couplers may only plugged-in at the left side.



Reliability

- Wiring by means of spring pressure connections (CageClamps) at the front connector
- Core cross-section 0.08...2.5mm^2 or 1.5 mm^2
- Total isolation of the wiring at module change
- Potential separation of all modules to the backplane bus
- Burst/ESD acc. IEC 61000-4-2/IEC 61000-4-4 (up 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
- Ventilation by means of a fan is not required

Green Cable for project engineering

For project engineering of your DP slave you may transfer your projects from your PC to the CPU serial via MPI by using the "Green Cable". Please also regard the hints to the Green Cable in this chapter!

Integrated power supply

Every Profibus slave has an internal power supply. This power supply requires DC 24V. In addition to the electronics on the bus coupler, the supply voltage is also used to power any modules connected to the backplane bus. Please note that the maximum current that the integrated power supply can deliver to the backplane bus is 3.5A.

The power supply is protected against reverse polarity and overcurrent.

Compatibility

The digital in-/output modules of the System 300V from VIPA are pin and function compatible to Siemens.

The project engineering happens in the SIMATIC manager from Siemens.

Note!

For programming of a System 300V CPU from VIPA please use always the **CPU 315-2DP (6ES7 315-2AF03 V1.2)** from Siemens in the hardware catalog.

Please note the Profibus address 1 of the CPU 31x is system dependent reserved.

For the project engineering, a thorough knowledge of the Siemens SIMATIC manager and the hardware configurator is required!

Chapter 2 Assembly and installation guidelines

Outline

In this chapter you will find all information, required for the installation and the cabling of a process control with the components of the System 300V.

Content

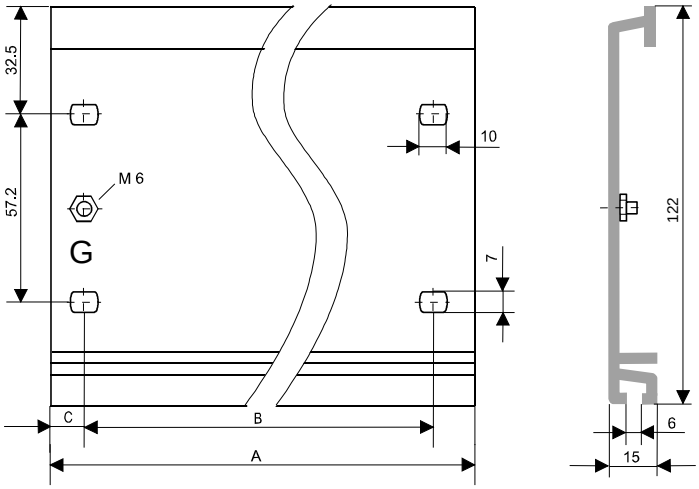
Overview

General

The single modules are directly installed on a profile rail and connected via the backplane bus coupler. Before installing the modules you have to clip the backplane bus coupler to the module from the backside.

The backplane bus coupler are included in the delivery of the peripheral modules.

Profile rail

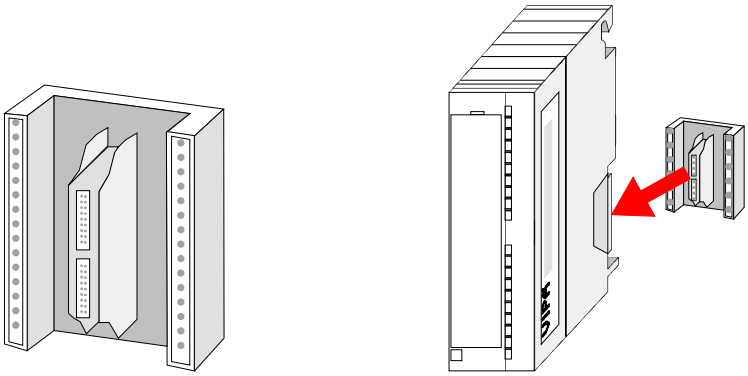


Order number	A	B	C
VIPA 390-1AB60	160mm	140mm	10mm
VIPA 390-1AE80	482mm	466mm	8.3mm
VIPA 390-1AF30	530mm	500mm	15mm
VIPA 390-1AJ30	830mm	800mm	15mm
VIPA 390-9BC00*	2000mm	no Drillings	15mm

* Unit pack: 10 pieces

Bus connector

For the communication between the modules the System 300V uses a backplane bus connector. The backplane bus connector are included in the delivering of the peripheral modules and are clipped at the module from behind before installing it to the profile rail.



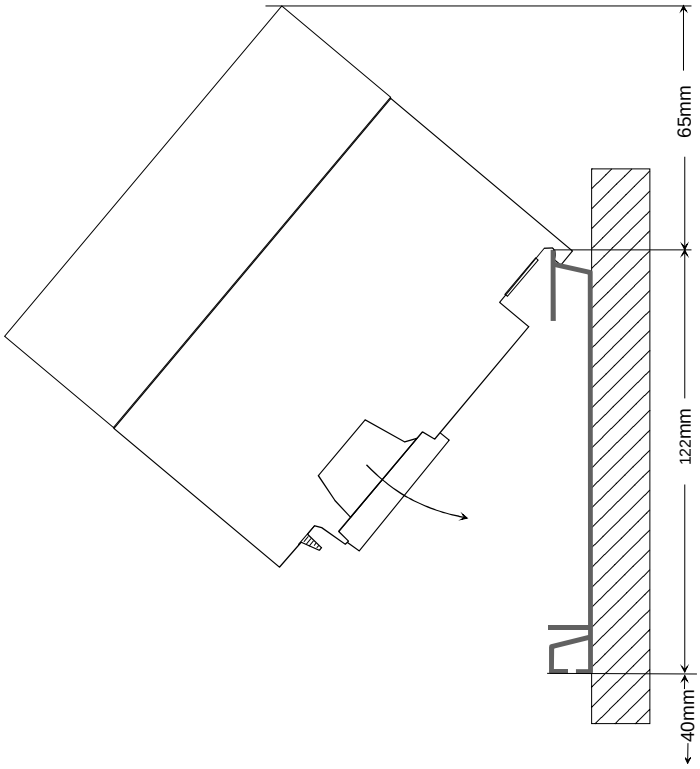
Installation dimensions

Overview Here follows all the important dimensions of the System 300V.

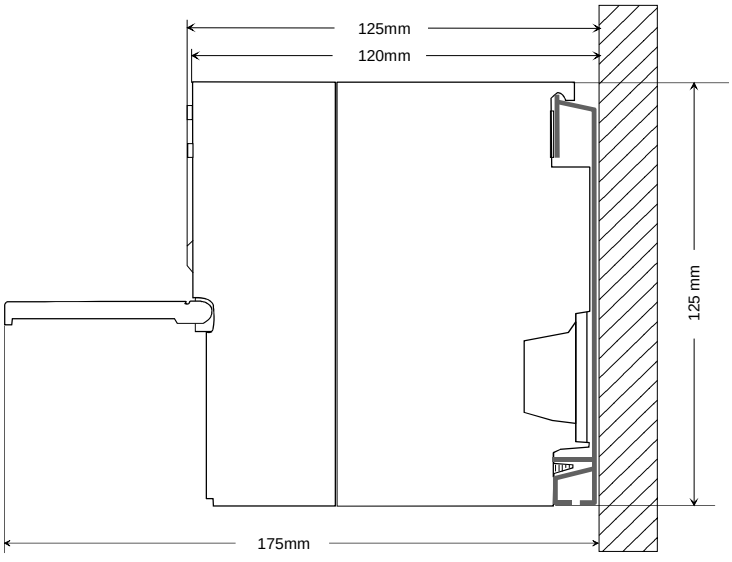
Dimensions Basic enclosure

1tier width (WxHxD) in mm: 40 x 125 x 120
2tier width (WxHxD) in mm: 80 x 125 x 120
3tier width (WxHxD) in mm: 120 x 125 x 120

Dimensions



Installation dimensions



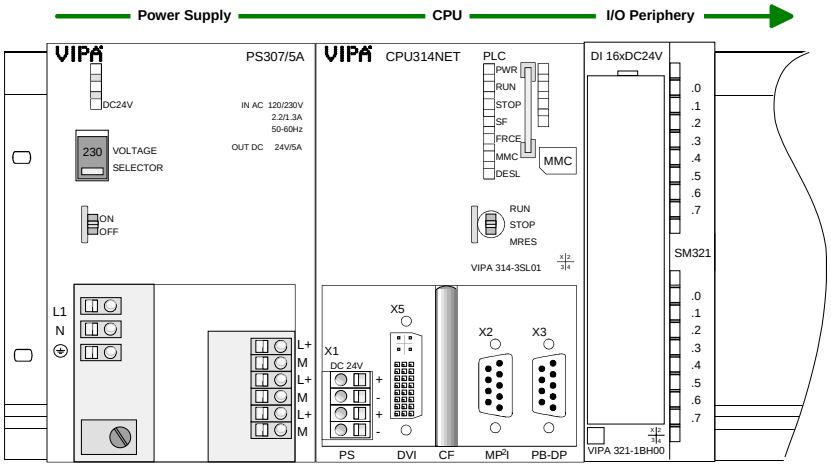
Installation at the profile rail

Structure:

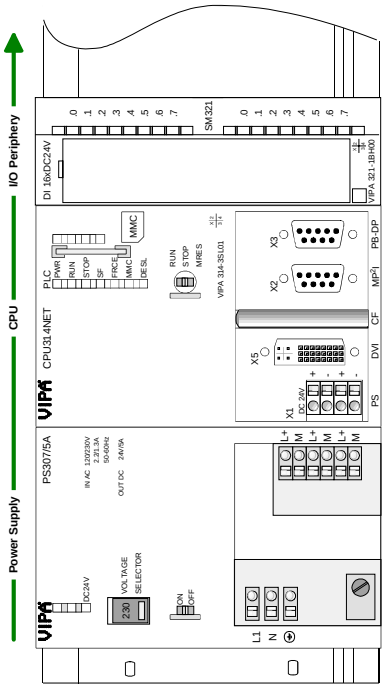
You may install the System 300V as well horizontal as vertical. Please regard the allowed environment temperatures:

- horizontal structure: from 0 to 60°
- vertical structure: from 0 to 40°

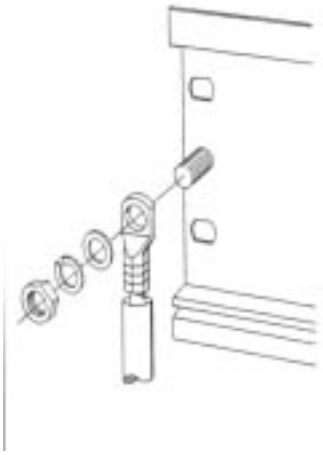
The horizontal structure always starts at the left side with the power supply and the CPU, then you plug-in the peripheral modules beside to the right.
You may plug-in maximum 32 peripheral modules to the CPU.



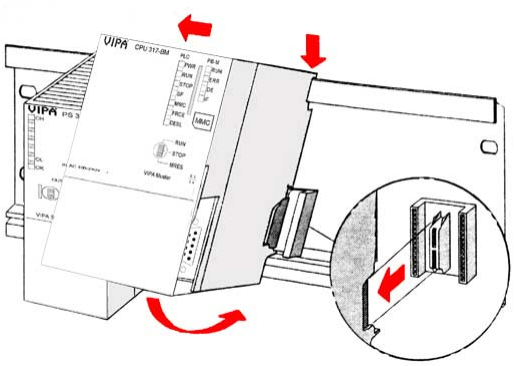
The vertical structure is turned for 90° against the clockwise direction.



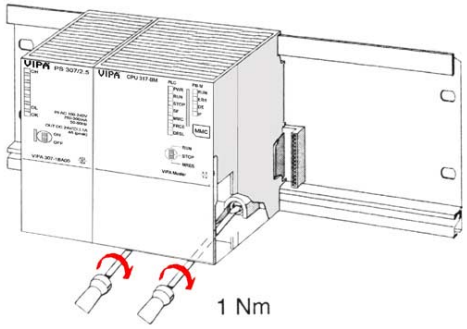
Approach



- Bolt the profile rail with the background (screw size: M6), so that you still have minimum 65mm space above and 40mm below the profile rail.
- If the background is a grounded metal or device plate, please look for a low-impedance connection between profile rail and background.
- Connect the profile rail with the protected earth conductor. For this purpose there is a bolt with M6-thread.
- The minimum cross-section of the cable to the protected earth conductor has to be 10mm^2.



- Stick the power supply to the profile rail and pull it to the left side to the grounding bolt of the profile rail.
- Fix the power supply by screwing.
- Take a bus coupler and click it at the CPU from behind like shown in the picture.
- Stick the CPU to the profile rail right from the power supply and pull it to the power supply.



- Click the CPU downwards and bolt it like shown.
- Repeat this procedure with the peripheral modules, by clicking a backplane bus coupler, stick the module right from the modules you've already fixed, click it downwards and connect it with the backplane bus coupler of the last module and bolt it.

Danger!

- Before installing or overhauling the System 300V, the power supplies must be disconnected from voltage (pull the plug or remove the fuse)!
- Installation and modifications only by properly trained personnel!

Cabling

Overview

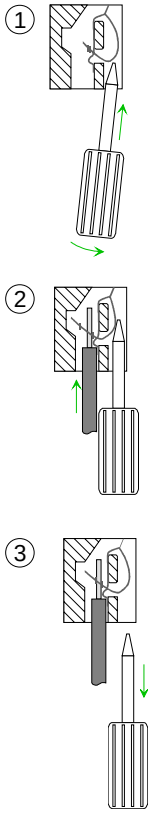
The power supplies and CPUs are exclusively delivered with CageClamp contacts. For the signal modules the front connectors are available from VIPA with screw contacts. In the following all connecting types of the power supplies, CPUs and input/output modules are described.

Danger!

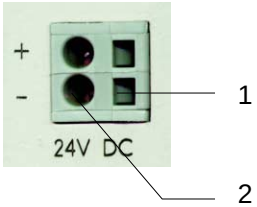
- Before installation or overhauling, the power supplies must be disconnected from voltage (pull the plug or remove the fuse)!
- Installation and modifications only by properly trained personnel!

CageClamp technology (gray)

For the cabling of power supplies, bus couplers and parts of the CPU, gray connectors with CageClamp technology are used.



You may connect wires with a cross-section of 0.08mm^2 to 2.5mm^2. You can use flexible wires without end case as well as stiff wires.



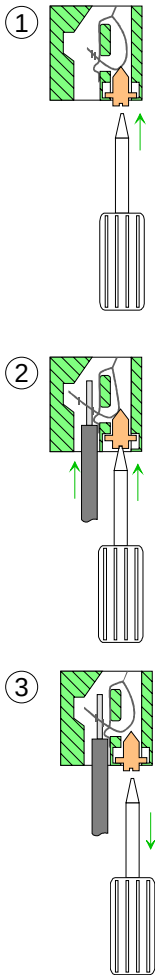
[1] Rectangular opening for screwdriver
[2] Round opening for wires

The picture on the left side shows the cabling step by step from top view.

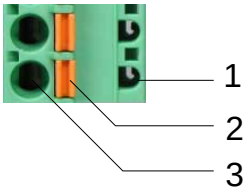
- To conduct a wire you plug a fitting screwdriver obliquely into the rectangular opening like shown in the picture.
- To open the contact spring you have to push the screwdriver in the opposite direction and hold it.
- Insert the insulation striped wire into the round opening. You may use wires with a cross-section from 0.08mm^2 to 2.5mm^2.
- By removing the screwdriver the wire is connected safely with the plug connector via a spring.

CageClamp technology (green)

For the cabling of e.g. the power supply of a CPU, green plugs with CageClamp technology are deployed.



Here also you may connect wires with a cross-section of 0.08mm^2 to 2.5mm^2. You can use flexible wires without end case as well as stiff wires.



[1] Test point for 2mm test tip
[2] Locking (orange) for screwdriver
[3] Round opening for wires

The picture on the left side shows the cabling step by step from top view.

- For cabling you push the locking vertical to the inside with a suiting screwdriver and hold the screwdriver in this position.
- Insert the insulation striped wire into the round opening. You may use wires with a cross-section from 0.08mm^2 to 2.5mm^2.
- By removing the screwdriver the wire is connected safely with the plug connector via a spring.

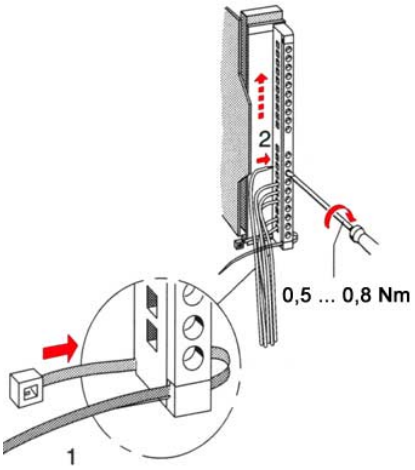
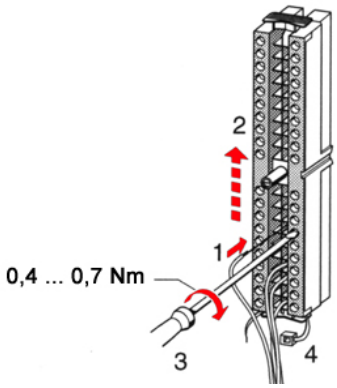
Note!

In opposite to the gray connection clamp from above, the green connection clamp is realized as plug that can be clipped off carefully even if it is still cabled.

Front connectors of the in-/output modules

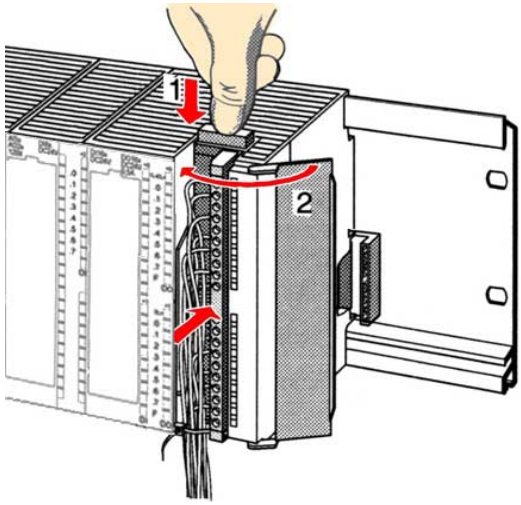
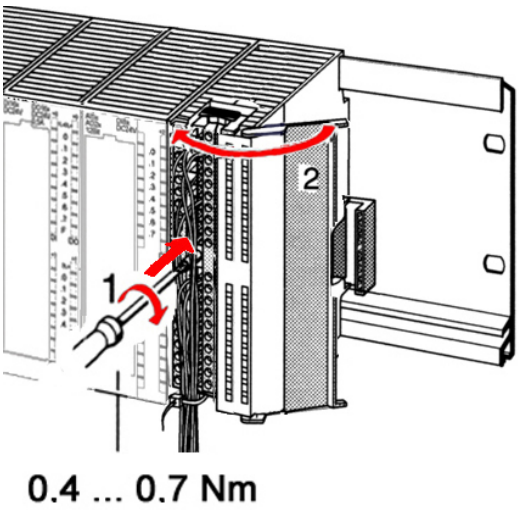
In the following the cabling of the three variants of the front-facing connector is shown:

For the I/O modules the following plugs are available at VIPA:

<table>
<tr><th>20pole screw connection
VIPA 392-1AJ00</th><th>40pole screw connection
VIPA 392-1AM00</th></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td colspan="2">Open the front flap of your I/O module.</td></tr>
<tr><td colspan="2">Bring the front connector in cabling position.
Herefore you plug the front connector on the module until it locks. In this position the front connector juts out of the module and has no contact yet.</td></tr>
<tr><td colspan="2">Deisolate your wires. If needed, use core end cases.</td></tr>
<tr><td>Thread the included cable binder into the front connector.</td><td></td></tr>
<tr><td colspan="2">If you want to lead out your cables from the bottom of the module, start with the cabling from bottom to top, res. from top to bottom, if the cables should be led out at the top.</td></tr>
<tr><td colspan="2">Bolt also the connection screws of not cabled screw clamps.</td></tr>
<tr><td> 
</td><td>Put the included cable binder around the cable bundle and the front connector.
 
</td></tr>
<tr><td colspan="2">Fix the cable binder for the cable bundle.</td></tr>
</table>

continued ...

... continue

<table>
<tr><th>20pole screw connection</th><th>40pole screw connection</th></tr>
<tr><td>Push the release key at the front connector on the upper side of the module and at the same time push the front connector into the module until it locks.
  </td><td>Bolt the fixing screw of the front connector.
  </td></tr>
<tr><td colspan="2">Now the front connector is electrically connected with your module.</td></tr>
<tr><td colspan="2">Close the front flap.</td></tr>
<tr><td colspan="2">Fill out the labeling strip to mark the single channels and push the strip into the front flap.</td></tr>
</table>

Installation Guidelines

General

The installation guidelines contain information about the interference free deployment of System 300V systems. There is the description of the ways, interference may occur in your control, how you can make sure the electromagnetic digestibility (EMC), and how you manage the isolation.

What means EMC?

Electromagnetic digestibility (EMC) means the ability of an electrical device, to function error free in an electromagnetic environment without being interferenced res. without interferencing the environment.

All System 300V components are developed for the deployment in hard industrial environments and fulfill high demands on the EMC. Nevertheless you should project an EMC planning before installing the components and take conceivable interference causes into account.

Possible interference causes

Electromagnetic interferences may interfere your control via different ways:

- Fields
- I/O signal conductors
- Bus system
- Current supply
- Protected earth conductor

Depending on the spreading medium (lead bound or lead free) and the distance to the interference cause, interferences to your control occur by means of different coupling mechanisms.

One differs:

- galvanic coupling
- capacitve coupling
- inductive coupling
- radiant coupling

Basic rules for EMC

In the most times it is enough to take care of some elementary rules to guarantee the EMC. Please regard the following basic rules when installing your PLC.

- Take care of a correct area-wide grounding of the inactive metal parts when installing your components.
  - Install a central connection between the ground and the protected earth conductor system.
  - Connect all inactive metal extensive and impedance-low.
  - Please try not to use aluminum parts. Aluminum is easily oxidizing and is therefore less suitable for grounding.
- When cabling, take care of the correct line routing.
  - Organize your cabling in line groups (high voltage, current supply, signal and data lines).
  - Always lay your high voltage lines and signal res. data lines in separate channels or bundles.
  - Route the signal and data lines as near as possible beside ground areas (e.g. suspension bars, metal rails, tin cabinet).
- Proof the correct fixing of the lead isolation.
  - Data lines must be laid isolated.
  - Analog lines must be laid isolated. When transmitting signals with small amplitudes the one sided laying of the isolation may be favorable.
  - Lay the line isolation extensively on a isolation/protected earth conductor rail directly after the cabinet entry and fix the isolation with cable clamps.
  - Make sure that the isolation/protected earth conductor rail is connected impedance-low with the cabinet.
  - Use metallic or metallized plug cases for isolated data lines.
- In special use cases you should appoint special EMC actions.
  - Wire all inductivities with erase links, that are not addressed by the System 300V modules.
  - For lightening cabinets you should prefer incandescent lamps and avoid luminescent lamps.
- Create an homogeneous reference potential and ground all electrical operating supplies when possible.
  - Please take care for the targeted employment of the grounding actions. The grounding of the PLC is a protection and functionality activity.
  - Connect installation parts and cabinets with the System 300V in star topology with the isolation/protected earth conductor system. So you avoid ground loops.
  - If potential differences between installation parts and cabinets occur, lay sufficiently dimensioned potential compensation lines.

Isolation of conductors

Electrical, magnetical and electromagnetical interference fields are weakened by means of an isolation, one talks of absorption.

Via the isolation rail, that is connected conductive with the rack, interference currents are shunt via cable isolation to the ground. Hereby you have to make sure, that the connection to the protected earth conductor is impedance-low, because otherwise the interference currents may appear as interference cause.

When isolating cables you have to regard the following:

- If possible, use only cables with isolation tangle.
- The hiding power of the isolation should be higher than 80%.
- Normally you should always lay the isolation of cables on both sides. Only by means of the both-sided connection of the isolation you achieve a high quality interference suppression in the higher frequency area.

  Only as exception you may also lay the isolation one-sided. Then you only achieve the absorption of the lower frequencies. A one-sided isolation connection may be convenient, if:
  - the conduction of a potential compensating line is not possible
  - analog signals (some mV res. µA) are transferred
  - foil isolations (static isolations) are used.
- With data lines always use metallic or metallized plugs for serial couplings. Fix the isolation of the data line at the plug rack. Do not lay the isolation on the PIN 1 of the plug bar!
- At stationary operation it is convenient to deisolate the isolated cable interruption free and lay it on the isolation/protected earth conductor line.
- To fix the isolation tangles use cable clamps out of metal. The clamps must clasp the isolation extensively and have well contact.
- Lay the isolation on an isolation rail directly after the entry of the cable in the cabinet. Lead the isolation further on to the System 300V module and **don't** lay it on there again!

Please regard at installation!

At potential differences between the grounding points, there may be a compensation current via the isolation connected at both sides.

Remedy: Potential compensation line

Chapter 3 Profibus DP

Outline

This chapter describes the usage of the Profibus DP slaves at the System 300.

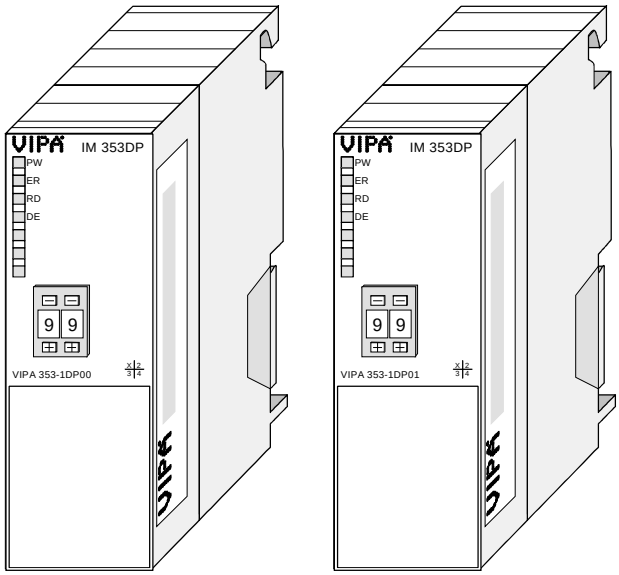
After a short introduction and a system overview you may find here all information about assembly, project engineering and diagnostic. The chapter closes the technical data.

Content

System overview

The following Profibus slaves are available for the System 300:

- Profibus DP slave IM 353DP with DP-V0
- Profibus DP slave IM 353DP with DP-V0 / DP-V1



Ordering data

Type	Order number	Description	Page
IM 353DP	VIPA 353-1DP00	Profibus DP-V0 slave	3-12
IM 353DP	VIPA 353-1DP01	Profibus DP-V0/V1 slave	3-22

Basics

General

Profibus is an international standard applicable to an open fieldbus for building, manufacturing and process automation. Profibus defines the technical and functional characteristics of a serial fieldbus system that can be used to create a low (sensor-/actuator level) or medium (process level) performance network of programmable logic controllers.

Together with other fieldbus systems, Profibus has been standardized in **IEC 61158** since 1999. *IEC 61158* bears the title "Digital data communication for measurement and control - Fieldbus for use in industrial control systems".

Profibus comprises an assortment of compatible versions. The following details refer to Profibus DP.

Profibus DP-V0

Profibus DP-V0 *(Decentralized Peripherals)* provides the basic functionality of DP, including cycle data exchange as well as station diagnostic, module diagnostic and channel-related diagnostic.

Profibus DP is a special protocol intended mainly for automation tasks in a manufacturing environment. DP is very fast, offers Plug'n'Play facilities and provides a cost-effective alternative to parallel cabling between PLC and remote I/O. Profibus DP was designed for high-speed data communication on the sensor-actuator level.

Profibus DP-V1

The original version, designed DP-V0, has been expanded to include version DP-V1, offering acyclic data exchange between master and slave.

DP-V1 contains enhancements geared towards process automation, in particular acyclic data communication for parameter assignment, operation, visualization and interrupt handling of intelligent field devices, parallel to cycle user data communication. This permits online access to station using engineering tools. In addition, DP-V1 defines interrupts. Examples for different types of interrupts are status interrupt, update interrupt and a manufacturer-specific interrupt.

If you'd like to use the DP-V1 functionality you have to make sure your DP master also supports DP-V1. More detailed information about this is to be found in the documentation of your DP master.

Master and slaves

Profibus distinguishes between active stations (master) and passive stations (slave).

Master devices

Master devices control the data traffic at the bus. It is also possible to operate with multiple masters on a Profibus. This is referred to as multi-master operation. The protocol on the bus establishes a logical token ring between intelligent devices connected to the bus. Only the master that has the token, can communicate with its slaves.

A master is able to issue unsolicited messages if it is in possession of the access key (token). The Profibus protocol also refers to masters as active participants.

Slave devices

A Profibus slave acquires data from peripheral equipment, sensors, actuators and transducers. The VIPA Profibus couplers are modular slave devices that transfer data between the System 300V periphery and the high-level master.

In accordance with the Profibus standards these devices have no bus-access rights. They are only allowed to acknowledge messages or return messages to a master when this has issued a request. Slaves are also referred to as passive participants.

Master class 1 MSAC_C1

The master of the class 1 is a central control that exchanges cyclically information with the decentral stations (slaves) in a defined message cycle. Typical MSAC_C1 devices are controls (PLC) or PCs. MSAC_C1 devices gain active bus access which allows them to read the measuring values (inputs) of the field devices and to write the set points (outputs) of the actuators at a fixed time.

Master class 2 MSAC_C2

MSAC_C2 are employed for service and diagnostic. Here connected devices may be configured, measuring values and parameters are evaluated and device states can be requested. MSAC_C2 devices don't need to be connected to the bus system permanently. These also have active bus access.

Typical MSAC_C2 devices are engineering, project engineering or operator devices.

Communication

The bus transfer protocol provides two alternatives for the access to the bus:

Master with master

Master communication is also referred to as token-passing procedure. The token-passing procedure guarantees the accessibility of the bus. The permission to access the bus is transferred between individual devices in the form of a "token". The token is a special message that is transferred via the bus.

When a master is in possession of the token it has the permission to access the bus and it can communicate with any active or passive device. The token retention time is defined when the system is configured. Once the token retention time has expired, the token is passed to the following master which now has permission to access the bus and may therefore communicate with any other device.

Master-slave procedure

Data communication between a master and the slaves assigned to it, is conducted automatically in a predefined and repetitive cycle by the master. You assign a slave to a specific master when you define the project. You can also define which DP slaves are included and which are excluded from the cyclic exchange of data.

Data communication between master and slave can be divided into a parameterization, a configuration and a data transfer phase. Before a DP slave is included in the data transfer phase the master checks whether the defined configuration corresponds with the actual configuration. This check is performed during the definition and configuration phase. The verification includes the device type, format and length information as well as the number of inputs and outputs. In this way a reliable protection from configuration errors is achieved.

The master handles the transfer of application related data independently and automatically. You can, however, also send new configuration settings to a bus coupler.

When the status of the master is DE "Data Exchange" it transmits a new series of output data to the slave and the reply from the slave contains the latest input data.

Data consistency

Consistent data is the term used for data that belongs together by virtue of its contents. This is the high and the low byte of an analog value (word consistency) as well as the control and status byte along with the respective parameter word for access to the registers.

The data consistency as applicable to the interaction between the periphery and the controller is only guaranteed for 1Byte. This means that input and output of the bits of a byte occurs together. This byte consistency suffices when digital signals are being processed.

Where the data length exceeds a byte, for example in analog values, the data consistency must be extended. Profibus guarantees that the consistency will cater for the required length.

Restrictions

- You can only install or remove peripheral modules when you have turned the power off!
- The max. distance for RS485 cables between two stations is 1200m (depending on the baud rate)
- The maximum baud rate is 12Mbaud

Diagnostic

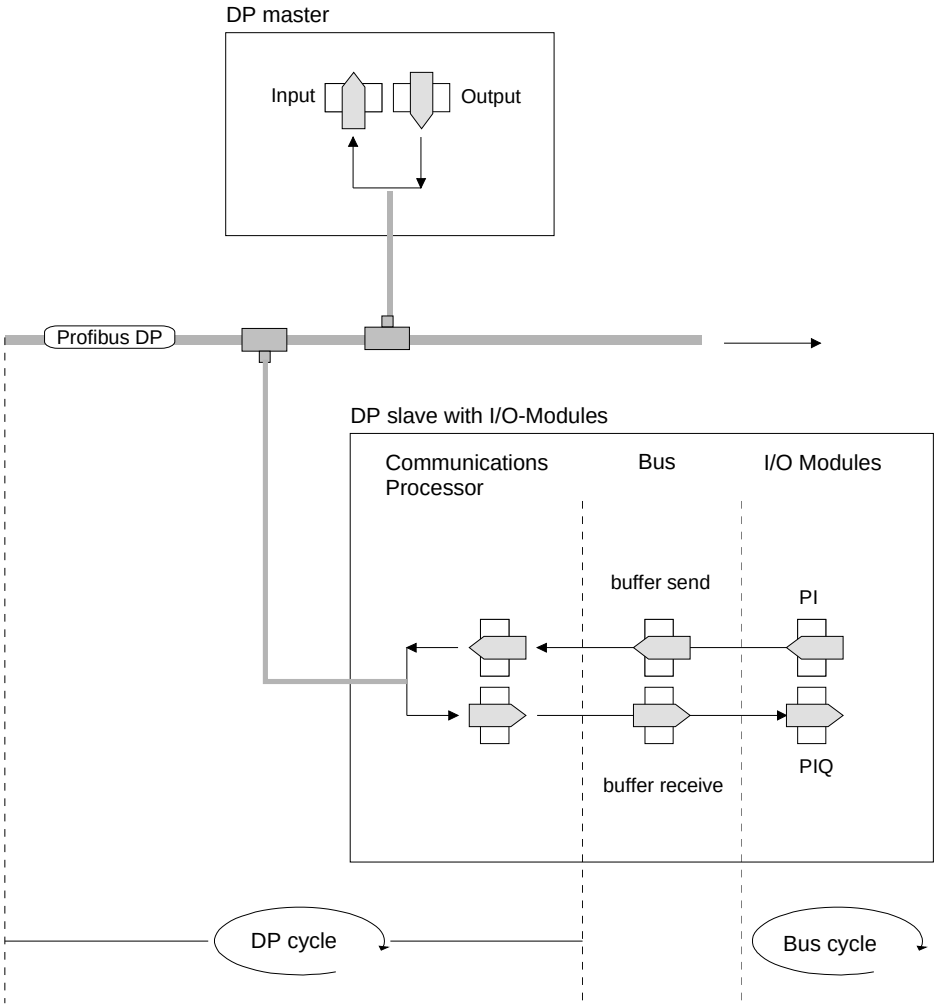
Profibus DP provides an extensive set of diagnostic functions for fast error localization. Diagnostic messages are transferred via the bus and collected by the master.

As a further function, the device-specific diagnostic of the DP-V1 have been enhanced and divided into the categories interrupts and status messages.

Function cyclic data communication (DP-V0)

DP-V0 provides the basic functionality of DP, including cycle data exchange as well as station diagnostic, module diagnostic and channel-related diagnostic.

Data is transferred cyclically between the DP master and the DP slave by means of transmit and receive buffers.



PII: process image of the inputs
PIQ: process image of the outputs

Bus cycle

A bus cycle saves all the input data from the modules in the PII and all the output data from the PIQ in the output modules. When the data has been saved the PII is transferred into the ”send buffer” and the contents of the ”receive buffer” is transferred into PIQ.

DP cycle

During a Profibus cycle the master addresses all its slaves according to the sequence defined in the data exchange. The data exchange reads and writes data from/into the memory areas assigned to the Profibus.

The contents of the Profibus input area is entered into the ”receive buffer” and the data in the ”send buffer” is transferred into the Profibus output area.

The exchange of data between DP master and DP slave is completed cyclically and it is independent from the bus cycle.

Bus cycle ≤ DP cycle

To ensure that the data transfer is synchronized the bus cycle time should always be less than or equal to the DP cycle time.

The parameter **min_slave_interval = 3ms** is located in the GSD-file.

In an average system it is guaranteed that the Profibus data on the bus is updated after a max. time of 3ms. You can therefore exchange data with the slave at intervals of 3ms.

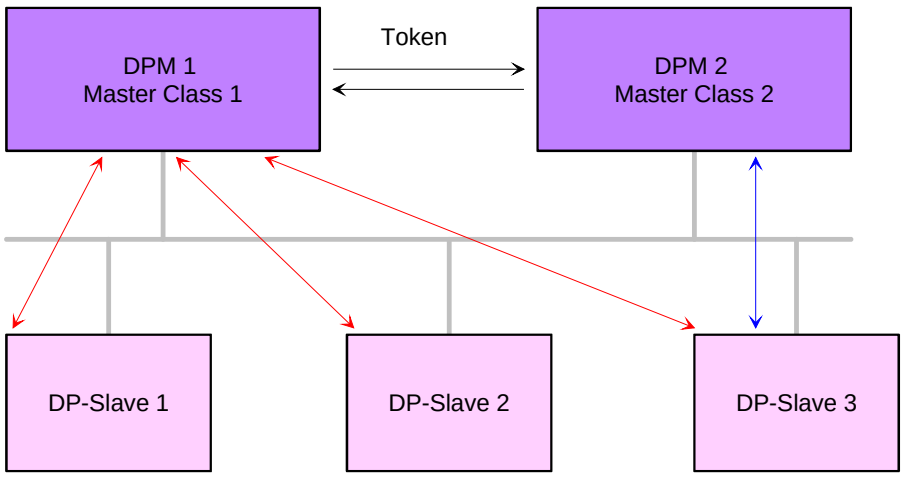
Note!

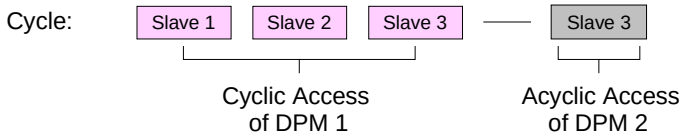
Starting with release version 6, the RUN-LED of a DP-V0 slave extinguishes as soon as the Bus cycle lasts longer than the DP cycle. This function is de-activated at the employment of a DP-V1 slave as DP-V0.

Function Acyclic data communication (DP-V1)

The key feature of version DP-V1 is the extended function for acyclic data communication. This forms the requirement for parameterization and calibration of the field devices over the bus during runtime and for the introduction of confirmed interrupt messages.

Transmission of acyclic data is executed parallel to cycle data communication, but with lower priority.





The DPM 1 (Master Class 1) has the token and is able to send messages to or retrieve them from slave 1, then slave 2, etc. in a fixed sequence until it reaches the last slave of the current list (MS0 channel); it then passes on the token to the DPM 2 (Master Class 2). This master can then use the remaining available time ("gap") of the programmed cycle to set up an acyclic connection to *any* slave (e.g. slave 3) to exchange records (MS2 channel); at the end of the current cycle time it returns the token to the DPM1.

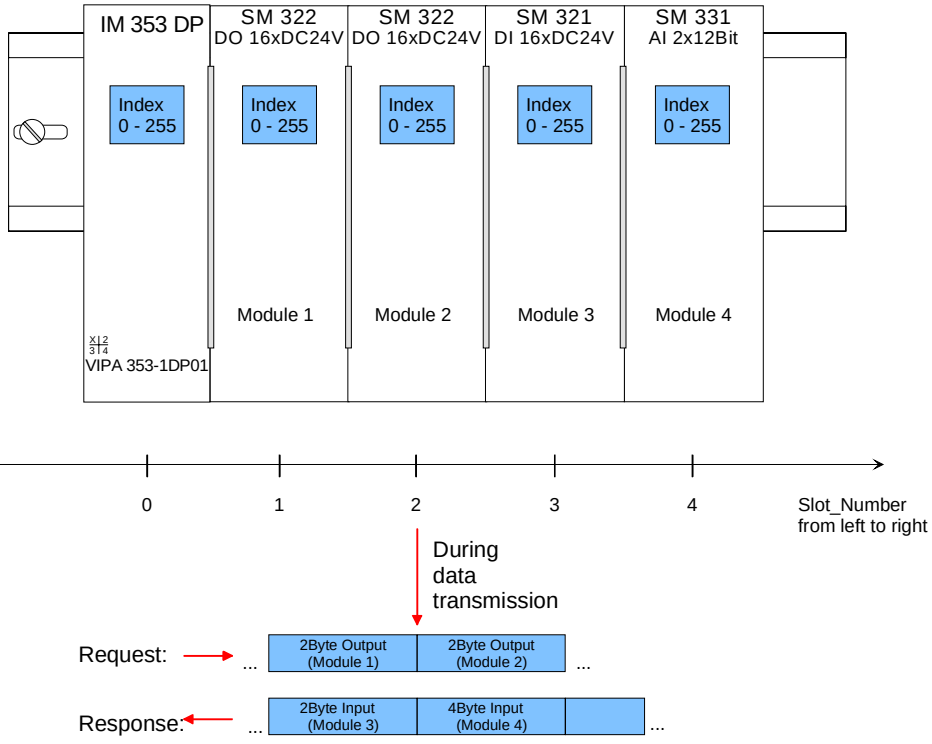
The acyclic exchange of records can last for several scan cycles on their "gaps"; at the end, the DPM 2 uses the gap to clear the connection. Similarly as well as the DPM 2, the DPM 1 can also execute acyclic data exchange with slaves (MS1 channel).

Addressing with Slot and Index

When addressing data, Profibus assumes that the physical structure of the slaves is *modular* or it can be structured internally in logical functional units, so-called *modules*. This model is also used in the basic DP functions for cyclic data communication where each module has a constant number of input-/output bytes that are transmitted in a fixed position in the user data telegram. The addressing procedure is based on identifiers, which characterize a module type as input, output or a combination of both. All identifiers combined produce the configuration of the slave, which is also checked by the DPM when the system starts up.

The acyclic data communication is also based on this model. All data blocks enabled for read/write access are also regarded as assigned to the modules and can be addressed using slot number and index.

The *Slot_Number* addresses the module and the *index* addresses the data blocks of a module. The Slot_Number = 0 addresses data of the Profibus coupler, Slot_Number > 0 addresses the data of the function modules.



Each data block can be up to 244Bytes. In the case of modular devices, the slot number is assigned to the modules. Compact devices are regarded as a unit of virtual modules. These can also be addressed whit slot number and index. Through the length specification in the read/write request, it is also possible to read/write parts of a data block.

Note!

For the addressing at the deployment of the Siemens SIMATIC manager the following conventions are valid:

DP slave coupler: Setting of the *diagnostic address* as ID

Modules of the DP slave coupler: Setting of the *module address* as ID. For an output module you have to set additionally Bit 15 of the module address (e.g. address 0004h becomes 8004h). With a combination module you have to set the lower one of the two addresses.

Services
Acyclic data communication

For the deployment of the DP-V1 services you have to take care that your CPU supports DP-V1 communication. More detailed information about this is to be found in the description of your CPU. The following system function blocks are available for this:

SFB 52 Read record set from a DP slave

SFB 53 Write record set to a DP slave

SFB 54 Receive interrupt from a DP slave

The following text shows the services for the acyclic data transfer that are using that function blocks.

More detailed information about the services and the DP-V0/V1 communication principles are to be found in the Profibus norm IEC 61158.

DPM 1
(Master class 1)

Services for Acyclic data transfer between DPM 1 and slaves	
Read	The master reads a data block from the slave.
Write	The master writes a data block to the slave.
Interrupt	An interrupt is transmitted from the slave to the master, which explicitly acknowledges receipt. The slave can only send a new interrupt message after it has received this acknowledgement; this prevents any interrupts being overwritten.
Interrupt_ Acknowledge	The master acknowledges receipt of an interrupt to the slave.
Status	A status message is transmitted from the slave to the master. There is no acknowledgment.
Data transmission is connection-oriented over a MS1 connection. This is set up by the DPM 1 and is closely linked to the connection for cyclic data communication. It can be used by the master that has parameterized and configured the respective slave.	

DPM 2
(Master class 2)

Services for Acyclic data transfer between DPM 2 and slaves	
Initiate Abort	Setup and termination of a connection for acyclic data communication between the DPM 2 and the slave
Read	The master reads a data block from the slave.
Write	The master writes a data block to the slave.
Data_ Transport	The master can write application-specific data (specified in profiles) acyclically to the slave and if required, read data from the slave in the same cycle.
Data transmission is connection-oriented over a MS2 connection. This is set up before the start of the acyclic data communication by the DPM 2 using the Initiate service. The connection is then available for Read, Write and Data_Transport services. The connection is terminated correspondingly. A slave can maintain several active MS2 connections simultaneously. A limitation is given by the resources of the slave.	

Data transfer medium as RS485 interface

Profibus employs screened twisted pair cable on the basis of the RS485 interface. The data transfer rate of the system is limited to a max. of 12MBaud.

The RS485 interface uses differential voltages. For this reason this kind of interface is less susceptible to interference than a plain voltage or current based interface. The network may be configured as linear or as tree structure. Your Profibus coupler carries a 9pin socket. This socket is used to connect the Profibus coupler to the Profibus network as a slave.

Due to the bus structure of RS485, any station may be connected or disconnected without interruptions and a system can be commissioned in different stages. Extensions to the system do not affect stations that have already been commissioned. Any failures of stations or new devices are detected automatically.

Addressing

Every device on the Profibus is identified by an address. This address must be an unique number in the bus system between 1 and 99. The address of the VIPA Profibus coupler is set by the addressing switch located on the front of the module.

GSD- file

Every VIPA Profibus slave is delivered together with a data medium. There you can find among others all GSD files of the VIPA Profibus modules.

The assignment of the GSD-file to your slave is shown in the following table:

Order number	GSD-file
VIPA 353-1DP00	VIPA056B.gsd
VIPA 353-1DP01(DP-V0)	VI0009AF.gsd *
VIPA 353-1DP01(DP-V1)	VI0109AF.gsd

*) This GSD-file is used for Profibus master that don't support DP-V1.

Please install the required files from your disc into your configuration tool. Details on the installation of the GSD and/or type files are available from the manual supplied with your configuration tool.

You may also download the GSD-file via the ftp-server

ftp: //ftp.vipa.de/support/profibus_gsd_files.

After the installation of the GSD-file you will find this entry e.g. in the hardware catalog from Siemens under:

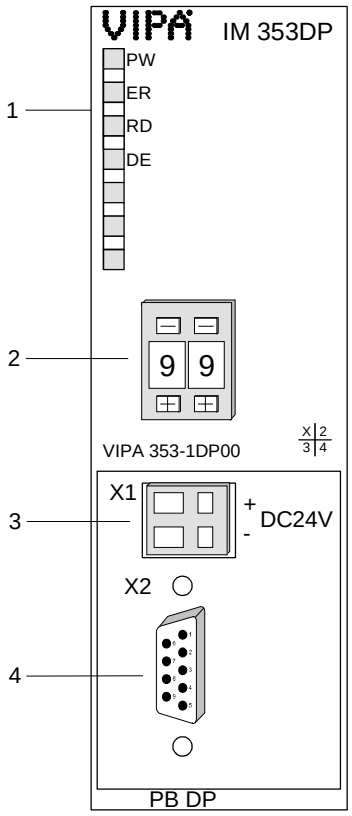
Profibus DP>Additional field devices>I/O>VIPA_System_300V>
VIPA 353-1DP00

IM 353-1DP00 - DP-V0 slave - Structure

Properties

- Profibus DP slave 9.6kBaud up to 12MBaud
- Profibus DP slave for max. 32 peripheral modules (max. 16 analog)
- Max. 152Byte input and 152Byte output data
- LED for bus diagnostic
- Internal diagnostic protocol with time stamp
- Integrated DC 24V power supply for providing the peripheral modules (max. 3.5A)

Front view 353-1DP00



[1] LED Status indicators
[2] Address selector

The following components are beneath a flap

[3] DC 24V voltage supply
[4] RS485 interface

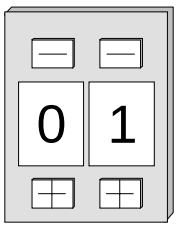
Components

LEDs

The module carries a number of LEDs that are available for diagnostic purposes on the bus and for displaying the local status. The following table explains the different colors of the diagnostic LEDs.

Label	Color	Description
PW	Yellow	Signalizes applying operation voltage (Power).
ER	Red	Short flash at reboot.
		On at internal error.
		Blinks at initialization error.
		Blinks alternately with RD at wrong configuration of the master (project engineering error).
		Blinks simultaneously with RD at wrong parameterization.
RD	Green	On at "Data exchange" if Bus cycle is faster than Profibus cycle.
		Off at "Data exchange" if Bus cycle is slower than Profibus cycle.
		Blinks at positive self test (READY) and successful initialization.
		Blinks alternately with ER at wrong configuration of the master (project engineering error).
		Blinks simultaneously with ER at wrong parameterization.
DE	Yellow	DE (Data exchange) indicates Profibus communication

Address selector



The address selector allows you to set the Profibus address of the Profibus slave. Permissible addresses are 1 to 99. Every address must be unique at the bus.

The slave address has to be set before turning on the bus coupler.

When you set the address 00 during operation, the diagnostic data is saved in the Flash-ROM one time. Please take care to reset the original Profibus address afterwards, so that the correct address is used at next start-up.

Note!

Please regard that the Profibus address assigned at project engineering and the address at the address selector have to be identical to assure an unambiguous identification of the Profibus slave.

Power supply

The Profibus slave has an internal power supply. This power supply requires DC 24V. In addition to the electronics on the bus coupler, the supply voltage is also used to power any modules connected to the backplane bus. Please note that the maximum current that the integrated power supply can deliver to the backplane bus is 3.5A.

The power supply is protected against reverse polarity.

Profibus and backplane bus are galvanically isolated from each other.

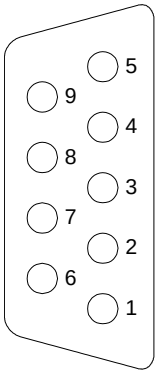
Attention!

Please ensure that the polarity is correct when connecting the power supply!

RS485 interface

Via a 9pin RS485 interface, you connect the Profibus slave with your Profibus network.

The pin assignment of this interface is as follows:



Pin	Assignment
1	n.c
2	n.c.
3	RxD/TxD-P (Line B)
4	RTS
5	M5V
6	P5V
7	n.c.
8	RxD/TxD-N (Line A)
9	n.c.

IM 353-1DP00 - DP-V0 slave - Project engineering

General

For project engineering a DP master engineering tool can be used like the Siemens SIMATIC manager. Here you assign the according Profibus DP slave modules to the DP master.

A direct assignment takes place via the Profibus address that you set at the DP slave address selector.

By installing the GSD file VIPA056B.gsd the IM 353-1DP00 is listed at the hardware catalog as "VIPA_DP300V". After the installation of the GSD-file, you'll find this at:

Profibus DP > Additional Field devices> I/O > VIPA_System_300V

GSD-file VIPA056B.gsd

VIPA supplies a data medium with every Profibus module. This medium contains among others all the GSD files of the VIPA Profibus modules.

Please install the file VIPA056B.gsd at your configuration tool. Details on the installation of a GSD file are available from the manual supplied with your configuration tool.

Project engineering

- Mount your Profibus system.
- Start your project engineering tool with a new project.
- Configure a master system and create a new Profibus subnet.
- For the project engineering of the IM 353-1DP00 take the "VIPA_DP300V" from the hardware catalog and drag it to the DP master subnet.
- Enter a Profibus address between 1 and 99 into the properties of the DP slave and set the same address at the address lever.
- Parameterize the DP slave (see parameters).
- Transfer your project to the PLC.

Parameter

Diagnostic interrupt

The Profibus slave has as parameter the value "Diagnostic interrupt" that you may activate via "YES" res. deactivate via "NO".

In delivery state, the diagnostic interrupt is activated. This causes the CPU to branch into the OB 82 in case of an error. Here you may react to the error and evaluate the diagnostic by using the SFC 13. Structure and evaluation of the diagnostic of the IM 353-1DP00 is described at the following pages.

Note

Every change in the arrangement of the modules must be followed by a re-calculation of the bus parameters!

IM 353-1DP00 - DP-V0 slave - Diagnostic functions

Overview

Profibus DP is provided with an extensive set of diagnostic functions that can be used to locate problems quickly and effectively. Diagnostic messages are transferred via the bus and collected by the master.

The DP slave transmits diagnostic data when requested by the master or when an error occurs. In case of an diagnostic interrupt, the CPU branches into OB 82. You may evaluate the diagnostic via SFC 13.

Additionally the DP slave stores the last 100 interrupt messages with a time stamp in a RAM res. in the Flash and may be evaluated with the help of software.

Please call the VIPA hotline for this!

Manual storage of diagnostic data

With the short setting of 00 at the address lever you may save the diagnostic data in the Flash-ROM during "DataExchange".

Internal diagnostic system messages

The system also stores diagnostic messages like the states "Ready" res. "DataExchange" that are not passed on to the master.

With every status change between "Ready" and "DataExchange" the Profibus slave stores the diagnostic-RAM content in a Flash-ROM and writes it back to the RAM at every reboot.

Diagnostic messages at voltage failure

At voltage failure res. decreasing voltage a time stamp is stored in the EEPROM. If enough voltage is still left, a diagnostic output to the master occurs.

At the next reboot an undervoltage/shut-down diagnostic message is generated from the time stamp of the EEPROMs and is stored in the Diagnostic-RAM.

Diagnostic data IM 353-1DP00

Diagnostic data consists of:

- Standard diagnostic data (Byte 0 ... 5)
- Identifier-related diagnostic data (Byte 6 ... 10)
- Module-related diagnostic data (Byte 11 ... 22)
- Channel-related diagnostic data (Byte 23 ... 25)

Standard diagnostic data

Byte 0	Station state 1
Byte 1	Station state 2
Byte 2	Station state 3
Byte 3	Master address
Byte 4	Ident number (low)
Byte 5	Ident number (high)

Identifier-related diagnostic data

Byte 6	Length and code identifier-related diagnostic
Byte 7 ... Byte 10	Identifier-related diagnostic messages

Module-related diagnostic data

Byte 11	Length and code module-related diagnostic
Byte 12	Module state
Byte 13 ... Byte 14	0 (fix)
Byte 15 ... Byte 22	Module-related diagnostic messages

Channel-related diagnostic data

Byte 23	Ident number of the module
Byte 24	Number of the channel
Byte 25	Error code

Standard diagnostic data IM 353-1DP00

The structure of the standard diagnostic data for slaves is as follows:

Standard diagnostic

Byte	Bit 7 ... Bit 0
0	Bit 0: 0 (fix) Bit 1: slave not ready for data exchange Bit 2: configuration data mismatch Bit 3: slave has external diagnostic data Bit 4: slave does not support the requested function Bit 5: 0 (fix) Bit 6: bad configuration Bit 7: 0 (fix)
1	Bit 0: slave requires re-configuration Bit 1: statistical diagnostic Bit 2: 1 (fix) Bit 3: Watchdog active Bit 4: Freeze-command was received Bit 5: Sync-command was received Bit 6: reserved Bit 7: 0 (fix)
2	Bit 6 ... 0: reserved Bit 7: diagnostic data overflow
3	Master address after configuration FFh: slave was not configured
4	Ident number high byte
5	Ident number low byte

Identifier-related diagnostic IM 353-1DP00

The identifier-related diagnostic tells whether modules of the System 300V DP slaves are defective or not.

The length of the identifier-related diagnostic data is fixed at 5 Byte.

Identifier-related diagnostic

Byte	Bit 7 ... Bit 0
6	Bit 5 ... 0: Length identifier-related diagnostic data 000101: Length 5 Byte (fix) Bit 7 ... 6: Code for identifier-related diagnostic 01: Code 01 (fix)
7	Bit 0: Module at plug-in location 1 Bit 1: Module at plug-in location 2 Bit 2: Module at plug-in location 3 Bit 3: Module at plug-in location 4 Bit 4: Module at plug-in location 5 Bit 5: Module at plug-in location 6 Bit 6: Module at plug-in location 7 Bit 7: Module at plug-in location 8
8	Bit 0: Module at plug-in location 9 Bit 1: Module at plug-in location 10 . . . Bit 7: Module at plug-in location 16
9	Bit 0: Module at plug-in location 17 Bit 1: Module at plug-in location 18 . . . Bit 7: Module at plug-in location 24
10	Bit 0: Module at plug-in location 25 Bit 1: Module at plug-in location 26 . . . Bit 7: Module at plug-in location 32

The Bits in Byte 7 and 10 are set when:

- the according module is dismantled.
- a not projected module is plugged-in.
- a plugged-in module denies access.
- the according module announces a diagnostic interrupt.

Module-related diagnostic data IM 353-1DP00

The module-related diagnostic shows the state of the projected modules and details the identifier-related diagnostic concerning the configuration.
The length of the module-related diagnostic data is fixed at 12Byte.

Module status

Byte	Bit 7 ... Bit 0
11	Bit 5 ... 0: Length module-related diagnostic data 001100: Length 12 Byte (fix) Bit 7 ... 6: Code for module-related diagnostic 00: Code 00 (fix)
12	Bit 6 ... 0: Status type 0000010: Module state Bit 7: 1 (fix)
13	permanent 00
14	permanent 00
15	Bit 1 ... 0: Module at plug-in location 1 Bit 3 ... 2: Module at plug-in location 2 Bit 5 ... 4: Module at plug-in location 3 Bit 7 ... 6: Module at plug-in location 4
16	Bit 1 ... 0: Module at plug-in location 5 . . . Bit 7 ... 6: Module at plug-in location 8
17	Bit 1 ... 0: Module at plug-in location 9 . . . Bit 7 ... 6: Module at plug-in location 12
18	Bit 1 ... 0: Module at plug-in location 13 . . . Bit 7 ... 6: Module at plug-in location 16
19	Bit 1 ... 0: Module at plug-in location 17 . . . Bit 7 ... 6: Module at plug-in location 20
20	Bit 1 ... 0: Module at plug-in location 21 . . . Bit 7 ... 6: Module at plug-in location 24
21	Bit 1 ... 0: Module at plug-in location 25 . . . Bit 7 ... 6: Module at plug-in location 28
22	Bit 1 ... 0: Module at plug-in location 29 . . . Bit 7 ... 6: Module at plug-in location 32

The Bits in Byte 15 ... 22 may have the following values:

00: Module OK; valid data
01: Modul error; invalid data (Module defective)
10: wrong module; invalid data
11: no module; invalid data

Channel-related diagnostic data IM 353-1DP00

The channel-related diagnostic shows channel errors of modules and details the identifier-related diagnostic.

The length of the channel-related diagnostic data is fixed at 3 Byte.

The channel-related diagnostic is only executed after the module-related diagnostic and only if one of the errors of Byte 25 occurs like Bus-init-error, Bus-QVZ and Bus-error.

Channel-related diagnostic

Byte	Bit 7 ... Bit 0
23	Bit 5 ... 0: Identifier number of the module with diagnostic 000000 ... 011111: Identifier number (Example: plug-in location 1 has the identifier number 0 etc.) Bit 7 ... 6: Code for channel-related diagnostic 10: Code 10 (fix)
24	Bit 5 ... 0: Number of the channel that announces diagnostic 000000 ... 111111: Channel number Bit 7 ... 6: In-/Output 01: Input 10: Output 11: In-/Output
25	Bit 4 ... 0: Error code 10101: *Bus-Init-Error*: Writing of parameters to the according module failed. 10110: *Bus-QVZ*: Read/Write error at the backplane bus occured 10111: *Bus-Error*: The number of the modules at the backplane bus is not equal to the projected modules. Bit 7 ... 5: Channel type 101: Word (fix)

IM 353-1DP01 - DP-V1 slave - Structure

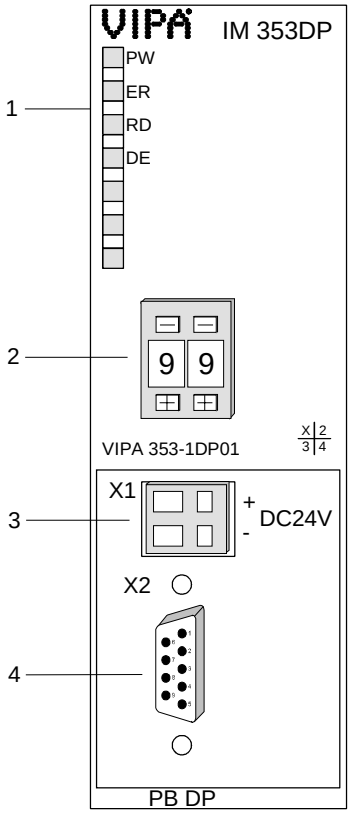
Properties

- Profibus (DP-V0, DP-V1)
- Profibus DP slave for max. 32 peripheral modules (max. 16 analog modules)
- Max. 244Byte input data and 244Byte output data
- Internal diagnostic protocol
- Integrated DC 24V power supply for the peripheral modules (3.5A max.)
- Supports all Profibus data transfer rates

Use as DP-V1 slave

- 1 MSAC_C1 connection (Read, Write) with 244Byte data (4Byte DP-V1-Header + 240Byte user data)
- 3 MSAC_C2 connections (Initiale, Read, Write, DataTransport, Initiate Abort) with each 244Byte data (4Byte DP-V1-Header + 240Byte user data)

Front view 353-1DP01



[1] LED Status indicators
[2] Address selector

The following components are beneath a flap

[3] DC 24V voltage supply
[4] RS485 interface

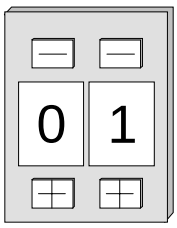
Components

LEDs

The module carries a number of LEDs that are available for diagnostic purposes on the bus and for displaying the local status. The following table explains the different colors of the diagnostic LEDs.

Label	Color	Description
PW	Green	Signalizes applying operation voltage (Power).
ER	Red	Short flash at reboot.
		On at internal error.
		Blinks at initialization error.
		Blinks alternately with RD at wrong configuration of the master (project engineering error).
		Blinks simultaneously with RD at wrong parameterization.
RD	Green	On at "Data exchange" if Bus cycle is faster than Profibus cycle.
		Off at "Data exchange" if Bus cycle is slower than Profibus cycle.
		Blinks at positive self test (READY) and successful initialization.
		Blinks alternately with ER at wrong configuration of the master (project engineering error).
		Blinks simultaneously with ER at wrong parameterization.
DE	Green	DE (Data exchange) indicates Profibus communication

Address selector



The address selector allows you to set the Profibus address of the Profibus slave. Permissible addresses are 1 to 99. Every address must be unique at the bus.

The slave address has to be set before turning on the bus coupler.

When you set the address 00 during operation, the diagnostic data is saved in the Flash-ROM one time. Please take care to reset the original Profibus address afterwards, so that the correct address is used at next start-up.

Note!

Please regard that the Profibus address assigned at project engineering and the address at the address selector have to be identical to assure an unambiguous identification of the Profibus slave.

Power supply

The Profibus slave has an internal power supply. This power supply requires DC 24V. In addition to the electronics on the bus coupler, the supply voltage is also used to power any modules connected to the backplane bus. Please note that the maximum current that the integrated power supply can deliver to the backplane bus is 3.5A.

The power supply is protected against reverse polarity.

Profibus and backplane bus are galvanically isolated from each other.

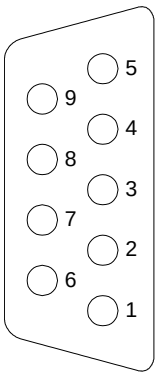
Attention!

Please ensure that the polarity is correct when connecting the power supply!

RS485 interface

Via a 9pin RS485 interface, you connect the Profibus slave with your Profibus network.

The pin assignment of this interface is as follows:



Pin	Assignment
1	n.c.
2	n.c.
3	RxD/TxD-P (Line B)
4	RTS
5	M5V
6	P5V
7	n.c.
8	RxD/TxD-N (Line A)
9	n.c.

IM 353-1DP01 - DP-V1 slave - Project engineering

General

For project engineering a DP master engineering tool can be used like the Siemens SIMATIC manager. Here you assign the according Profibus DP slave modules to the DP master.

A direct assignment takes place via the Profibus address that you set at the DP slave address selector.

By installing the corresponding GSD file the IM 353-1DP01 is listed at the hardware catalog as "VIPA_353-1DP01 (DP-V0 or DP-V1)".

You'll find this at:

Profibus DP > Additional Field devices> I/O > VIPA_System_300V

DP-V0/DP-V1 functionality by GSD file

VIPA supplies a data medium with every Profibus module. This medium contains among others all the GSD files of the VIPA Profibus modules.

Depending on the installed GSD file the following modules are listed at the hardware catalog.:

Module	GSD file
VIPA 353-1DP01 (DP-V0)	VI0009AF.gsd
VIPA 353-1DP01 (DP-V1)	VI0109AF.gsd

Install the appropriate GSD file in your engineering tool. For more information see the manual of you engineering tool.

Project engineering

- Mount your Profibus system.
- Start your project engineering tool with a new project.
- Configure a master system and create a new Profibus subnet.
- For the project engineering of the IM 353-1DP01 take the "VIPA 353-1DP01 (DPV0)" or "VIPA 353-1DP01 (DPV1)" for each functionality from the hardware catalog and drag it to the DP master subnet.
- Enter a Profibus address between 1 and 99 into the properties of the DP slave and set the same address at the address lever.
- Parameterize the DP slave (see parameters).
- Transfer your project to the PLC.

Note

Please note to place the following modules during hardware configuration to the first three slots:

Config for Slot1
Config for Slot2
Config for Slot3

These modules are automatically placed using the Siemens SIMATIC Manager:

Parameter data IM 353-1DP01 DP-V0

At usage of the IM 353-1DP01 (DP-V0) you have the following parameter data:

Byte	Bit 7 ... Bit 0	Default
0	Bit 2 ... 0: 0 (fix) Bit 3: 0 = WD-Timebase 10ms 1 = WD-Timebase 1ms Bit 4: 0 (fix) Bit 5: 0 = Publisher-Mode not available 1 = Publisher-Mode available	00h[1)]
1	00h (fix)	00h
2	08h (fix)	08h
3	0Ah (fix)	0Ah
4	81h (fix)	81h
5	00h (fix)	00h
6	00h (fix)	00h
7	Bit 0: 0 = Identifier-related diagnostic enable 1 = Identifier-related diagnostic disable Bit 1: 0 = Module status enable 1 = Module status disable Bit 2: 0 = Channel-related diagnostic enable 1 = Channel-related diagnostic disable Bit 3: 0 (fix) bit 4: 0 (fix) Bit 5: 0 = V0: Diagnostic interrupt not available 1 = V0: Diagnostic interrupt available Bit 6: 0 = V0: Hardware interrupt not available 1 = V0: Hardware interrupt available Bit 7: 0 (fix)	70h
8	Bit 7 ... 0: 0 (fix)	00h
9 ... 12	00h (fix)	00h

[1)] Using the Siemens SIMATIC Manager this value is automatically set and can not be changed.

Parameter data IM 353-1DP01 DP-V1

At usage of the IM 353-1DP01 (DP-V1) you have the following parameter data:

Byte	Bit 7 ... Bit 0	Default
0	Bit 2 ... 0: 0 (fix) Bit 3: 0 = WD-Timebase 10ms 1 = WD-Timebase 1ms Bit 4: 0 (fix) Bit 5: 0 = Publisher-Mode not available 1 = Publisher-Mode available Bit 6: 0 = Fail-Safe-Mode not available 1 = Fail-Safe-Mode available Bit 7: 0 = DP-V1 mode disable 1 = DP-V1 mode enable	C0h[1)]
1	Bit 0: Startup when expected/actual config. differ (must always be 0 else a parameterization error occures) Bit 3 ... 1: 0 (fix) Bit 4: 0 = V1: Vendor-specific interrupt not available 1 = V1: Vendor-specific interrupt available Bit 5: 0 = V1: Diagnostic interrupt not available 1 = V1: Diagnostic interrupt available Bit 6: 0 = V1: Hardware interrupt not available 1 = V1: Hardware interrupt available Bit 7: 0 (fix)	70h
2	08h (fix)	08h
3	0Ah (fix)	0Ah
4	81h (fix)	81h
5	00h (fix)	00h
6	00h (fix)	00h
7	Bit 0: 0 = Identifier-related diagnostic enable 1 = Identifier-related diagnostic disable Bit 1: 0 = Module status enable 1 = Module status disable Bit 2: 0 = Channel-related diagnostic enable 1 = Channel-related diagnostic disable Bit 7 ... 3: 0 (fix)	00h
8	Bit 7 ... 0: 0 (fix)	00h
9 ... 12	00h (fix)	00h

[1)] Using the Siemens SIMATIC Manager this value is automatically set and can not be changed.

IM 353-1DP01 - DP-V1 slave - DP-V1 services

Overview

For the deployment of the DP-V1 services you have to take care that your CPU supports DP-V1 communication. More detailed information about this is to be found in the description of your CPU. The following system function blocks are available for this:

SFB 52 Read record set from a DP slave

SFB 53 Write record set to a DP slave

SFB 54 Receive interrupt from a DP slave

Per default, one class-1 master and max 3 class-2 master connection with 244Byte data (4Byte DP-V1 header plus 240Byte user data) are supported. The class-1 master connection is established together with the cyclic connection and is activated via the parameterization. The class-2 master connection can be used by a C2 master that then communicates with the slave only acyclical and provides an own connection establishment.

Data from DP-V1 slave

To access the DP-V1 slave with the Siemens SIMATIC Manager the *diagnostic address*, which can be set by properties, is used as *ID*.

Using the following record set no. as *Index* you get access for reading (R) res. writing (W) to the listed DP slave elements:

Index	Access	Description
A0h	R	Device name as ASCII code (VIPA 353-1DP01)
A1h	R	Hardware Version as ASCII code (V1.00)
A2h	R	Software Version as ASCII code (V1.00)
A3h	R	Serial number of the device as ASCII code (e.g. 000347 = 30h, 30h, 30h, 33h, 34h, 37h)
A4h	R	Device configuration (see table next page with module identification assigned to module type)
D0h	R	Number of stored diagnostic
	W	Any write instruction deletes every diagnostic entries
D1h	R	Read diagnostic entries in sequence
	W	Any write instruction stores diagnostic entries permanently in the FLASH memory

Structure stored diagnostic entry

With every D1h call a stored diagnostic entry with max. 26Byte is displayed starting with the newest one.

Basically every stored diagnostic entry has the following structure:

Label	Type	Description
Length	Word	Length of the diagnostic data
Time stamp	Double word	Internal time stamp
Diagnostic (max. 20Byte)	Byte	Diagnostic entry (interrupt) that is stored internal

device configuration

Via the index A4h, the module configuration of DP slave can be monitored. The assignment *identification* to *module type* can be found at the following table:

Module type	Identification	Input byte	Output byte
DI 8	9FC1h	1	-
DI 8 - Alarm	1FC1h	1	-
DI 16	9FC2h	2	-
DI 14 / 2C	08C0h	6	6
DI 32	9FC3h	4	-
DO 8	AFC8h	-	1
DO 16	AFD0h	-	2
DO 32	AFD8h	-	4
DIO 8	BFC9h	1	1
DIO 16	BFD2h	2	2
AI2	15C3h	4	-
AI4	15C4h	8	-
AI4 - fast	11C4h	8	-
AI8	15C5h	16	-
AO2	25D8h	-	4
AO4	25E0h	-	8
AO8	25E8h	-	16
AI2 / AO2	45DBh	4	4
AI4 / AO2	45DCh	8	4

Data of the function modules

To access the function modules with the Siemens SIMATIC Manager the *module address*, which can be set by properties, is used as *ID*.

Using the following record set no. as *Index* you get access for reading (R) res. writing (W) to the listed function module elements:

Index	Access	Description
00h	R	Diagnostic – record set 0
	W	Write Module parameters
01h	R	Diagnostic – record set 1

IM 353-1DP01 - DP-V1 slave - Diagnostic functions

Overview

Profibus DP provides an extensive set of diagnostic functions for quick error localization. Diagnostic messages are transferred via the bus and collected by the master.

At the DP-V1 the device related diagnostic has been improved as further function and is subdivided into the categories interrupts and status messages.

Additionally in the DP-V1 slave from VIPA the last 100 interrupt messages are stored in a RAM res. in the flash with a time stamp and may be evaluated with a software.

For this, please call the VIPA hotline!

In addition you can access diagnostic data using the DP-V1 services.

Difference diagnostic DP-V0 and DP-V1

At DP-V0 and DP-V1 there are identical diagnostic structure and behavior. The only difference consists of the fact that with employment in a system 300 with a hardware interrupt at DP-V0 the OB 82 and with DP-V1 the OB 40 is called.

Internal diagnostic system messages

The system also stores diagnostic messages like the states "Ready" res. "DataExchange" that are not passed on to the master.

With every status change between "Ready" and "DataExchange" the Profibus slave stores the diagnostic-RAM content in a Flash-ROM and writes it back to the RAM at every reboot.

Manual storage of diagnostic data

With the short setting of 00 at the address lever you may save the diagnostic data in the Flash-ROM during "DataExchange".

Diagnostic messages at voltage failure

At voltage failure res. decreasing voltage a time stamp is stored in the EEPROM. If enough voltage is still left, a diagnostic output to the master occurs.

At the next reboot an undervoltage/shut-down diagnostic message is generated from the time stamp of the EEPROMs and is stored in the Diagnostic-RAM.

Structure of the 353-1DP01 diagnostic data

The diagnostic messages that are created by the Profibus slave have, depending on the parameterization, a length of 58Byte.

As soon as the Profibus slave sends a diagnostic to the master, the max. of 58Byte diagnostic data are prepended by 6Byte standard diagnostic data:

Byte 0 ... Byte 5	Standard diagnostic data	Is only prepended at transfer to the master via Profibus	
x ... x+4	Identifier-related diagnostic		Can be enabled or disabled via paramete-rization
x ... x+11	Module state		
max. 9·(x ... x+2)	Channel-related diagnostic		
x ... x+19	Interrupt	Internal stored diagnostic	

Standard diagnostic data

At the transfer of a diagnostic to the master the slave standard diagnostic data are prepended to the diagnostic bytes. More detailed information to the structure of the slave standard diagnostic data is to find in the standard papers of the Profibus User Organization.

The slave standard diagnostic data have the following structure:

Standard diagnostic

Byte	Bit 7 ... Bit 0
0	Bit 0: Bit is always at 0 Bit 1: slave is not yet ready for exchange data Bit 2: Configuration data does not correspond to actual configuration Bit 3: External slave diagnostic available Bit 4: Request function is not supported by slave Bit 5: 0 (fix) Bit 6: Wrong parameterization Bit 7: 0 (fix)
1	Bit 0: New parameters have to be assigned to slave Bit 1: Statistic Diagnostic Bit 2: 1 (fix) Bit 3: Response monitoring has been enabled Bit 4: "FREEZE" control command received Bit 5: "SYNC" control command received Bit 6: reserved Bit 7: 0 (fix)
2	Bit 6 ... 0: reserved Bit 7: Diagnostic data overflow
3	Master address after parameterizing FFh: Slave has not been parameterized
4	Ident number High Byte
5	Ident number Low Byte

Identifier-related diagnostic

Via the Identifier-related diagnostic you gain information at which plug-in location (module) an error has occurred.

More detailed information about the error is available via the *Module state* and the *channel-related diagnostic*.

The identifier-related diagnostic can be activated via the parameterization and has the following structure:

Identifier-related diagnostic

Byte	Bit 7 ... Bit 0
X	Bit 5 ... 0: 000101 (fix) Length of the Identifier-related diagnostic Bit 7 ... 6: 01 (fix) Code for Identifier-related diagnostic
X+1	The bit is set if one of the following occurs: - a module is removed - an unconfigured module is inserted - an inserted module cannot be accessed - a module reports a diagnostic interrupt Bit 0: Entry for module on slot 1 Bit 1: Entry for module on slot 2 Bit 2: Entry for module on slot 3 Bit 3: Entry for module on slot 4 Bit 4: Entry for module on slot 5 Bit 5: Entry for module on slot 6 Bit 6: Entry for module on slot 7 Bit 7: Entry for module on slot 8
X+2	Bit 0: Entry for module on slot 9 Bit 1: Entry for module on slot 10 Bit 2: Entry for module on slot 11 Bit 3: Entry for module on slot 12 Bit 4: Entry for module on slot 13 Bit 5: Entry for module on slot 14 Bit 6: Entry for module on slot 15 Bit 7: Entry for module on slot 16
X+3	Bit 0: Entry for module on slot 17 Bit 1: Entry for module on slot 18 Bit 2: Entry for module on slot 19 Bit 3: Entry for module on slot 20 Bit 4: Entry for module on slot 21 Bit 5: Entry for module on slot 22 Bit 6: Entry for module on slot 23 Bit 7: Entry for module on slot 24
X+4	Bit 0: Entry for module on slot 25 Bit 1: Entry for module on slot 26 Bit 2: Entry for module on slot 27 Bit 3: Entry for module on slot 28 Bit 4: Entry for module on slot 29 Bit 5: Entry for module on slot 30 Bit 6: Entry for module on slot 31 Bit 7: Entry for module on slot 32

Module status

The module status gives you detailed information about the error that occurred at a module.

The module status can be activated via the parameterization and has the following structure:

Module status

Byte	Bit 7 ... Bit 0
X	Bit 5 ... 0: 001100 (fix) Length of the Module status Bit 7 ... 6: 00 (fix) Code for Module status
X+1	82h (fix) Status type Module status
X+2	00h (fix)
X+3	00h (fix)
X+4	Follow bits indicates the status of the modules from slot 1 ... 32 00: Module ok - valid Data 01: Module error - invalid Data (Module defective) 10: Incorrect module - invalid Data 11: No Module - invalid Data Bit 1, 0: Module status module slot 1 Bit 3, 2: Module status module slot 2 Bit 5, 4: Module status module slot 3 Bit 7, 6: Module status module slot 4
X+5	Bit 1, 0: Module status module slot 5 Bit 3, 2: Module status module slot 6 Bit 5, 4: Module status module slot 7 Bit 7, 6: Module status module slot 8
X+6	Bit 1, 0: Module status module slot 9 Bit 3, 2: Module status module slot 10 Bit 5, 4: Module status module slot 11 Bit 7, 6: Module status module slot 12
X+7	Bit 1, 0: Module status module slot 13 Bit 3, 2: Module status module slot 14 Bit 5, 4: Module status module slot 15 Bit 7, 6: Module status module slot 16
X+8	Bit 1, 0: Module status module slot 17 Bit 3, 2: Module status module slot 18 Bit 5, 4: Module status module slot 19 Bit 7, 6: Module status module slot 20
X+9	Bit 1, 0: Module status module slot 21 Bit 3, 2: Module status module slot 22 Bit 5, 4: Module status module slot 23 Bit 7, 6: Module status module slot 24
X+10	Bit 1, 0: Module status module slot 25 Bit 3, 2: Module status module slot 26 Bit 5, 4: Module status module slot 27 Bit 7, 6: Module status module slot 28
X+11	Bit 1, 0: Module status module slot 29 Bit 3, 2: Module status module slot 30 Bit 5, 4: Module status module slot 31 Bit 7, 6: Module status module slot 32

Channel-related Diagnostic

With the channel-related diagnostic you gain detailed information about the channel error within a module. For the usage of the channel-related diagnostic you have to release the diagnostic interrupt for every module via the parameterization. The channel-related diagnostic can be activated via the parameterization and has the following structure:

Channel-related diagnostic

Byte	Bit 7 ... Bit 0
X	Bit 5 ... 0: ID number of the module that delivers the channel-specific diagnostic (000001 ... 011111) e.g.: Slot 1 has ID no. 0 Slot 32 has ID no. 31 Bit 7, 6: 10 (fix) Code for channel-related diagnostic
X+1	Bit 5 ... 0: Number of the channel or the channel group that delivers the diagnostic (00000 .... 11111) Bit 7 ... 6: 01=Input Module 10=Output Module 11=In-/Output Module
X+2	Bit 4 ... 0: *Error messages to Profibus standard* 00001: Short circuit 00010: Undervoltage (Supply voltage) 00011: Overvoltage (Supply voltage) 00100: Output Module is overloaded 00101: Temperature rise output Module 00110: Open circuit sensors or actors 00111: Upper limit violation 01000: Lower limit violation 01001: Error - Load voltage at the output - Sensor supply - Hardware error in the Module *Error messages - manufacturer-specific* 10000: Parameter assignment error 10001: Sensor or load voltage missing 10010: Fuse defect 10100: Ground fault 10101: Reference channel error 10110: Hardware interruptlost 11001: Safety-related shutdown 11010: External fault 11010: Indefinable error - not specified Bit 7 ... 5: Channel type 001: Bit 010: 2 Bit 011: 4 Bit 100: Byte 101: Word 110: 2 Words

The maximum number of channel-related diagnostic is limited by the total length of 58Byte for diagnostic. By de-activating of other diagnostic ranges you may release these areas for further channel-related diagnostic. For each channel always 3Byte are used.

Interrupts

The interrupt section of the slave diagnostic shows informations about interrupt type and cause. It consists of max. 20Byte. For every slave diagnostic max. 1 interrupt can be send. The interrupt section is always the last part of the diagnostic telegram if activated it in the parameterization.

Structure

Depending on the interrupt type, the interrupt section has the following structure:

Byte	Element	Description
x...x+3	Interrupt status	Contains information about the interrupt type
x+4...x+19	Diagnostic interrupt	The 16Byte correspond to the record set 1 of the CPU diagnostic
x+4...x+7	Hardware interrupt	The 4Byte are module specific and are described with the according module.

Interrupt status

If there is a diagnostic event for channel/group 0 of a module, there may be a module error as well as a channel error. The entry is made in this case even if you have not enabled the diagnostic for channel (/channel group) 0 of a module.

The interrupt section is structured as follows:

Interrupt status Byte x ... x+3

Byte	Bit 7 ... Bit 0
x	Bit 5 ... 0: 010100: Length of the interrupt section incl. Byte x Bit 7 ... 6: 00 (fix) Code for Module-Related diagnostic
x+1	Bit 6 ... 0: Type of interrupt 0000001: Diagnostic interrupt 0000010: Hardware interrupt Bit 7: Code for interrupt
x+2	Bit 7 ... 0: Slot of the module that is producing interrupt 1 ... 32
x+3	Bit 1, 0: 00: Hardware interrupt 01: Diagnostic interrupt $_{incoming}$ 10: Diagnostic interrupt $_{outgoing}$ 11: reserved Bit 2: 0 (fix) Bit 7 ... 3: interrupt sequence number 1 …32

Interrupt status at diagnostic interrupt Bytes x+4 to x+7
(corresponds CPU diagnostic record set 0)

Byte	Bit 7 ... Bit 0
x+4	Bit 0: Module malfunction, i.e. a problem has been detected Bit 1: Internal error in the module Bit 2: External error - module no longer addressable Bit 3: Channel error in the module Bit 4: Load power supply is missing Bit 5: Front connector is missing Bit 6: Module is not parameterized Bit 7: Parameter assignment error
x+5	Bit 3 ... 0: Module class 1111: Digital module 0101: Analog module 1000: FM 1100: CP Bit 4: Channel information available Bit 5: User information available Bit 6: 0 (fix) Bit 7: 0 (fix)
x+6	Bit 0: Memory or coding key analog module is missing Bit 1: Communication error Bit 2: Operating mode 0: RUN 1: STOP Bit 3: Cycle time monitoring addressed Bit 4: Module power supply failure Bit 5: Empty battery Bit 6: Complete backup failure Bit 7: 0 (fix)
x+7	Bit 0: reserved Bit 1: reserved Bit 2: reserved Bit 3: reserved Bit 4: reserved Bit 5: reserved Bit 6: Hardware interrupt lost Bit 7: reserved

continued …

… continue

Interrupt status at diagnostic interrupt Bytes x+8 to x+19 (corresponds to CPU diagnostic record set 1)

Byte	Bit 7 ... Bit 0
x+8	70h: Module with digital inputs 71h: Module with analog inputs 72h: Module with digital outputs 73h: Module with analog outputs 74h: Module with analog in-/-outputs 76h: Counter
x+9	Length of the channel-related diagnostic
x+10	Number of channels per module
x+11	Position (channel) with diagnostic event
x+12	Diagnostic event on the channel/channel group 0 Assignment see module description
x+13	Diagnostic event on the channel/channel group 1 Assignment see module description
. . .	. . .
x+19	Diagnostic event on the channel/channel group 7 Assignment see module description

Interrupt status at hardware interrupt Bytes x+4 to x+7

More detailed information to the diagnostic data is to find in the concerning module descriptions.

Installation guidelines

Profibus in general

- The VIPA Profibus DP network must have a linear structure.
- Profibus DP consists of minimum one segment with at least one master and one slave.
- A master is always used in conjunction with a CPU.
- Profibus supports a max. of 125 participants.
- A max. of 32 devices are permitted per segment.
- The maximum length of a segment depends on the transfer rate :

9.6 ... 187.5kBaud	→	1000m
500kBaud	→	400m
1.5MBaud	→	200m
3 ... 12MBaud	→	100m

- The network may have a maximum of 10 segments. Segments are connected by means of repeaters. Every repeater is also seen as participant on the network.
- All devices communicate at the same baud rate, slaves adapt automatically to the baud rate.
- The bus must be terminated at both ends.
- Masters and slaves may be installed in any combination.

Installation and integration with Profibus

- Assemble your Profibus system using the required modules.
- Transfer the supplied GSD-file into your system and configure the system as required. Choose a valid Profibus address.
- Transfer the configuration into your master.
- Connect the Profibus cable to the coupler and turn the power supply on.

Profibus using RS485

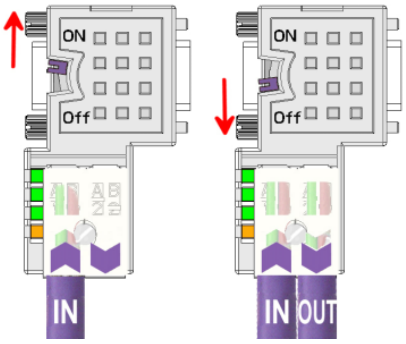
Profibus employs a screened twisted pair cable based on RS485 interface specifications as the data communication medium.

Note!

The Profibus line must be terminated with ripple resistor. Please ensure that the last participant the line is terminated by means of a terminating resistor.

Termination with "EasyConn"

The "EasyConn" bus connector is provided with a switch that is used to activate a terminating resistor.



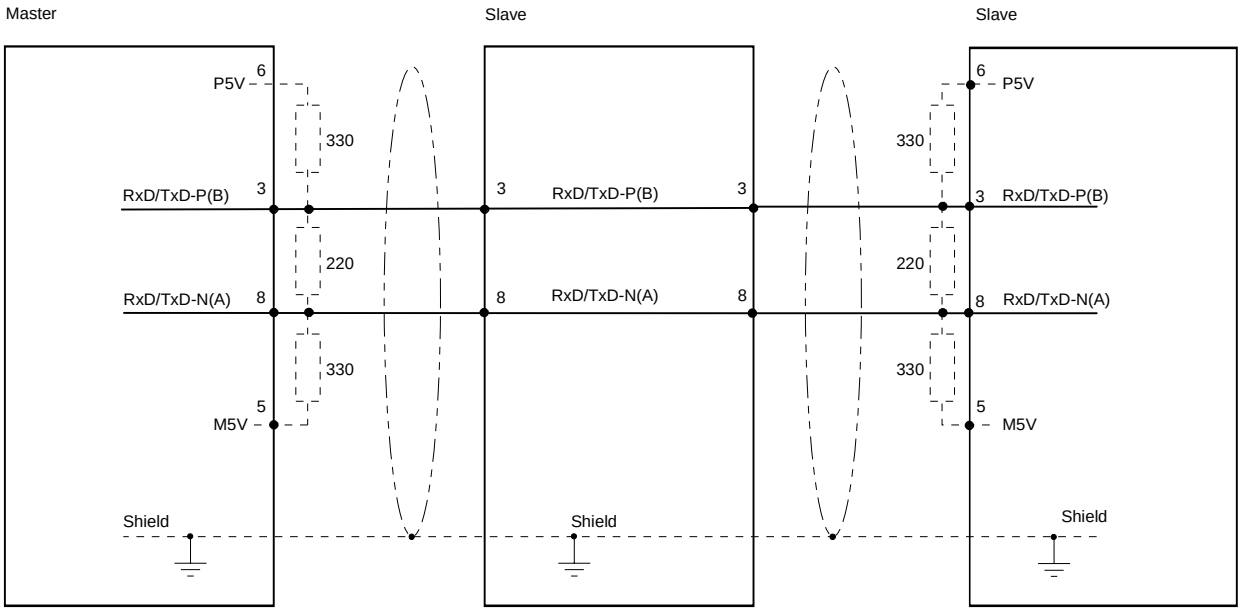
Attention!

The terminating resistor is only effective, if the connector is installed at a slave and the slave is connected to a power supply.

Note!

A complete description of installation and deployment of the terminating resistors is delivered with the connector.

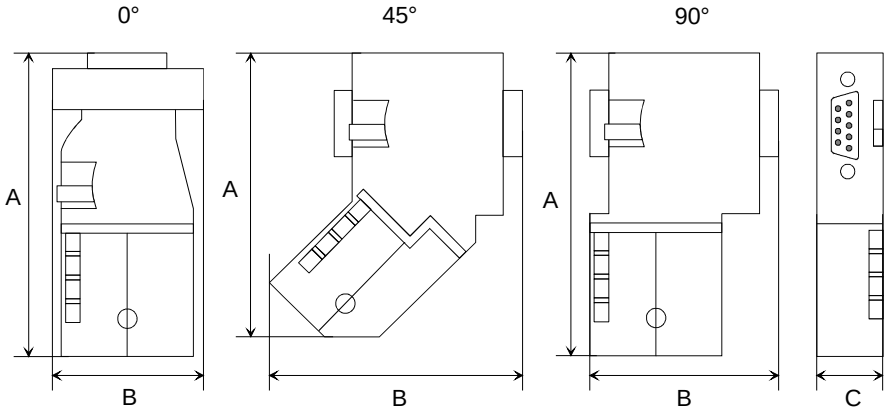
The following picture illustrates the terminating resistors of the respective start and end station.



"EasyConn" Bus connector

In systems with more than two stations all partners are wired in parallel. For that purpose, the bus cable must be feed-through uninterrupted.

Via the order number VIPA 972-0DP10 you may order the bus connector "EasyConn". This is a bus connector with switchable terminating resistor and integrated bus diagnostic.

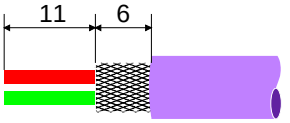


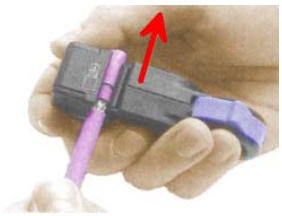
	0°	45°	90°
A	64	61	66
B	34	53	40
C	15.8	15.8	15.8

all in mm

Note!

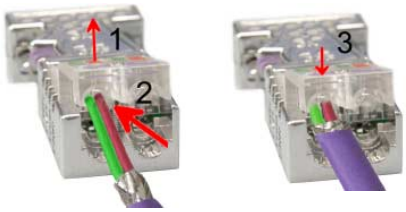
To connect this EasyConn plug, please use the standard Profibus cable type A (EN50170). Starting with release 5 also highly flexible bus cable may be used: Lapp Kabel order no.: 2170222, 2170822, 2170322.
Under the order no. 905-6AA00 VIPA offers the "EasyStrip" de-isolating tool that makes the connection of the EasyConn much easier.





all in mm

Assembly



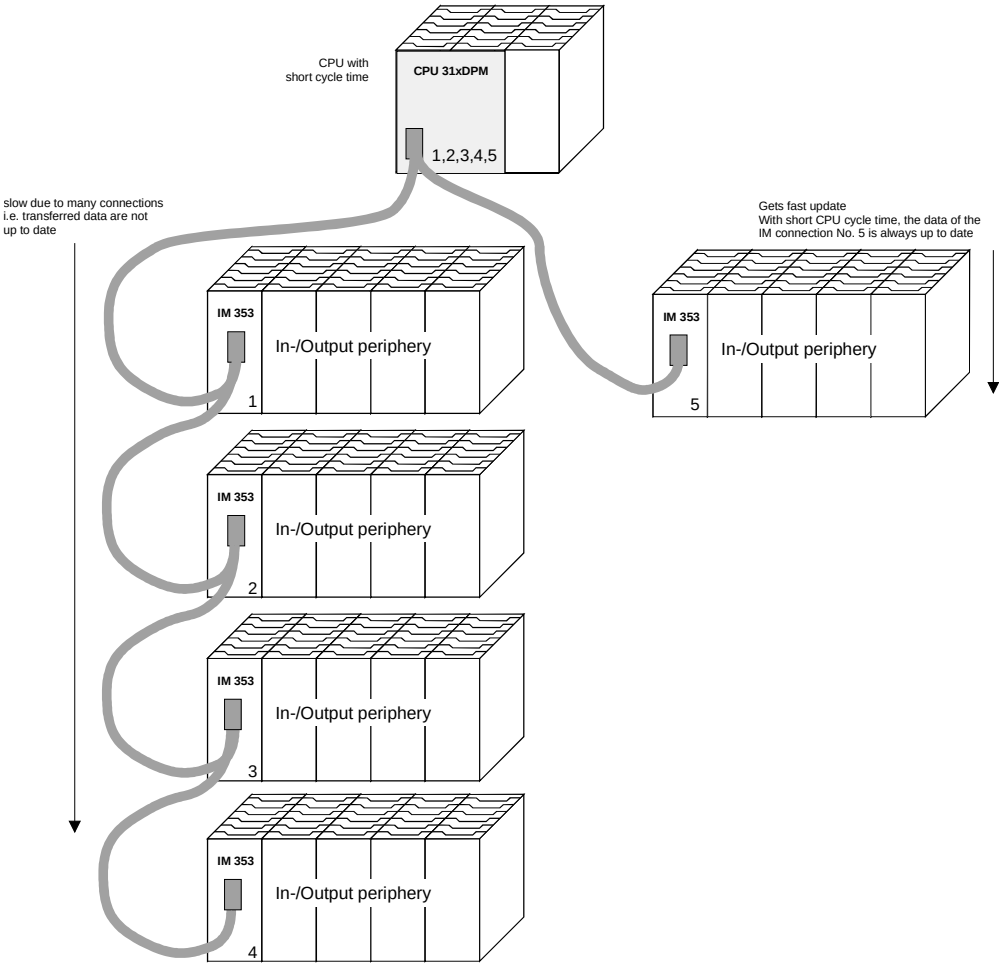
- Loosen the screw
- Lift contact-cover
- Insert both wires into the ducts provided (watch for the correct line colour as below!)
- Please take care not to cause a short circuit between screen and data lines!
- Close the contact cover
- Tighten screw
  (max. tightening torque 4Nm)

Please note: **The green line must be connected to A, the red line to B!**

Examples for Profibus network

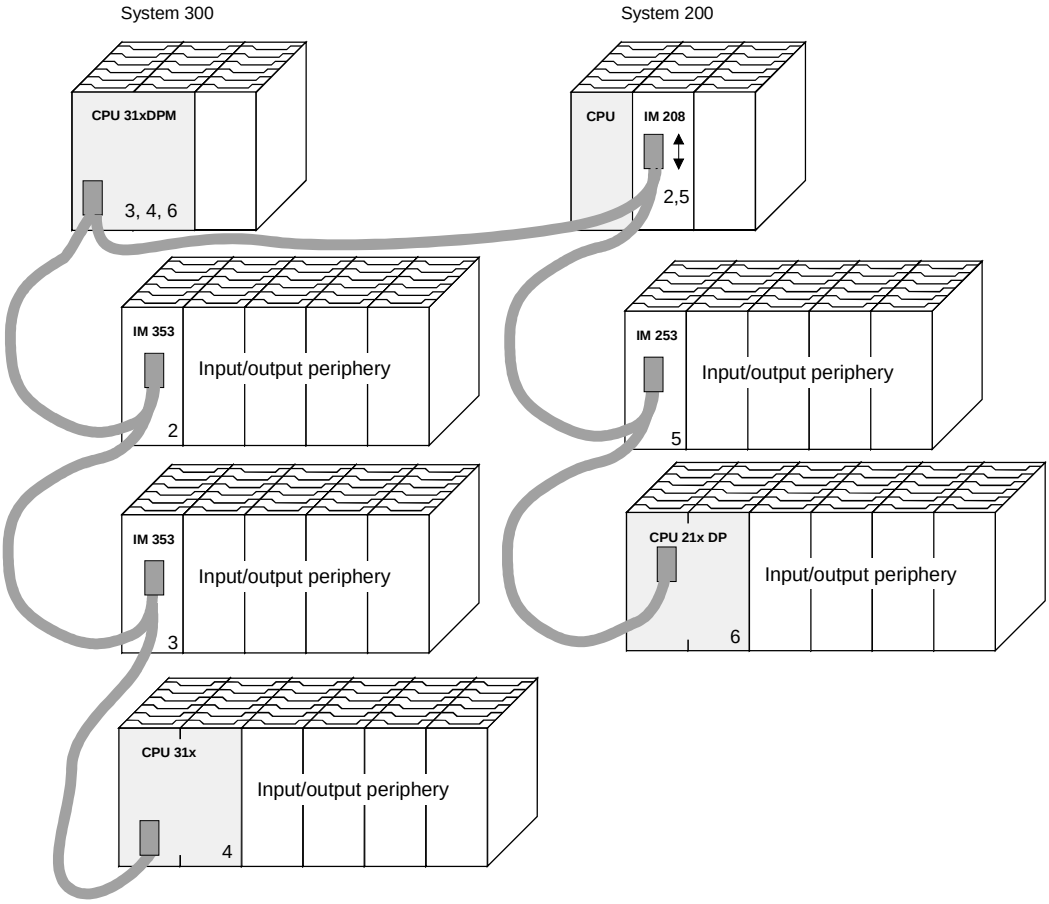
One CPU and multiple master connections

The CPU should have a short cycle time to ensure that the data from slave no. 5 (on the right) is always up to date. This type of structure is only suitable when the data from slaves on the slow trunk (on the left) is not critical. You should therefore not connect modules that are able to issue interrupts.



Multi Master System

Several master units are at the bus together with several slaves:



Commissioning

Overview

- Assemble your Profibus system.
- Configure your master system.
- Adjust a valid address of the Profibus.
- Transfer the configuration into your master.
- Connect the master and slave modules with the Profibus.
- Turn the power supply on.

Installation

Assemble your Profibus system with the wanted peripheral modules. Every Profibus slave coupler has an integrated power supply that has to be provided with DC 24V. Via the power supply not only the bus coupler is provided but also the modules connected via backplane bus. Please regard that the integrated power supply can provide the backplane bus with a max. of 3.5A.

Profibus and backplane bus are galvanically separated from each other.

Configuration in the master system

Configure your Profibus master in your master system. You can use the WinNCS of VIPA for this purpose ore the Siemens hardware configurator.

Addressing

At the Profibus slave modules, you set the Profibus address that you assigned at project engineering.

Transferring your project

Depending on the deployed master, there are different possibilities to transfer your project to the DP master.

Connecting a system by means of Profibus

In a system with more than one station all stations are wired in parallel. For this reason the bus cable must be feed-through uninterrupted.

You should always keep an eye on the correct polarity!

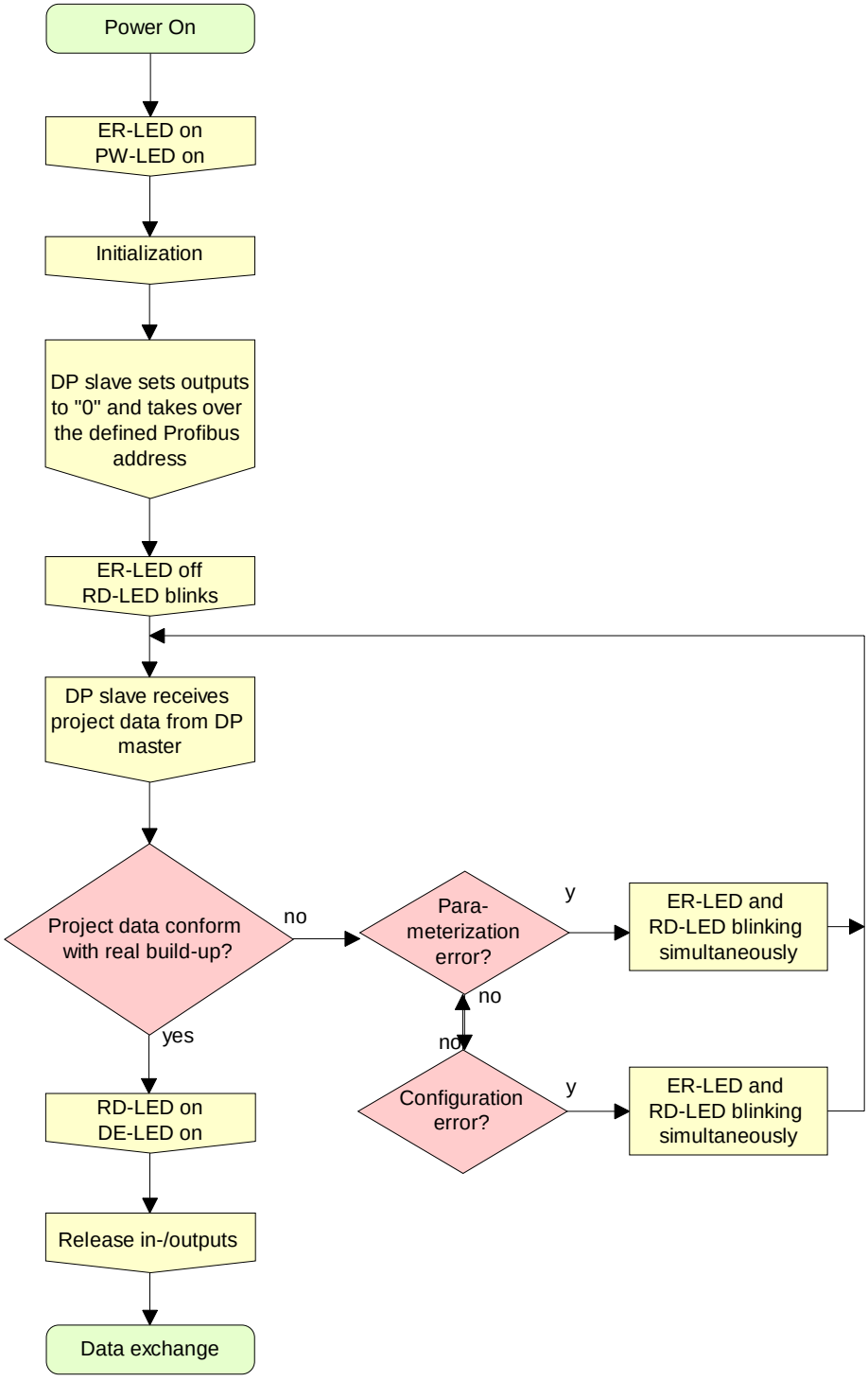
Note!

To prevent reflections and associated communication problems the bus cable has always to be terminated with its ripple resistor!

Start-up behavior IM 353DP slave

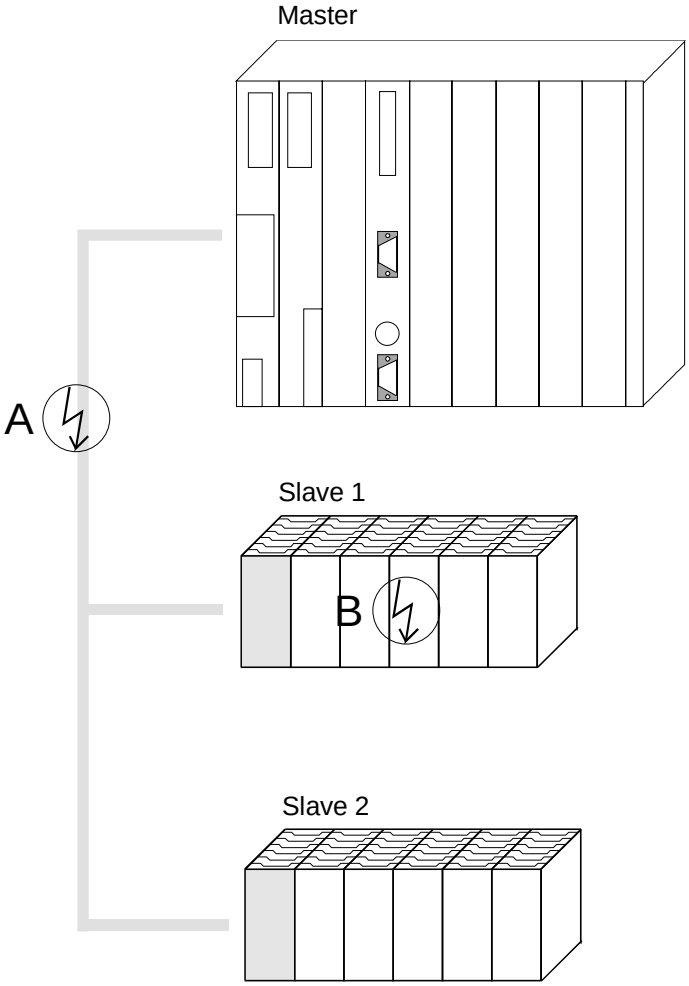
After power on, the DP slave executes a self test. It controls its internal functions and the communication via the backplane bus. After the error free start-up, the bus coupler switches into the state "ready". In this state, the DP slave gets its parameters from the DP master and, at valid parameters, switches into the state "DataExchange" DE (DE is permanently on).

At communication errors at the backplane bus, the Profibus slave switches into STOP and boots again after app. 2 seconds. As soon as the test has been completed positive, the RD-LED blinks.



Using the diagnostic LEDs

The following example shows the reaction of the LEDs for different types of network interruption.



Interruption at position A
The Profibus has been interrupted.

Interruption at position B
Communication via the backplane bus has been interrupted.

LED Slave 1	Position of interruption	
LED	A	B
RD	Blinks	Off
ER	Off	On
DE	Off	Off

LED Slave 2	Position of interruption	
LED	A	B
RD	blinks	On
ER	Off	Off
DE	Off	On

Technical Data

IM 353DP

Electrical data	VIPA 353-1DP00 (DP-V0)	VIPA 353-1DP01 (DP-V0/V1)
Voltage supply	DC 24V, via front from ext. power supply	
Current consumption	max. 1A	
Current supply int. bus	3.5A	
Potential separation	≥ AC 500V	
Status indication	Via LEDs at the front	
Connections/Interfaces	RS485: 9pin D-Type jack Profibus connection	
Profibus interface		
Connection	RS485: 9pin D-Type jack	
Network topology	Linear bus, active bus termination at both ends, tap lines possible.	
Medium	Screened and drilled Twisted Pair cable, screen may be left, depending on environment conditions	
Transfer rate	9.6kBaud up to 12MBaud (automatic adjustment)	
Total length	Without repeater 100m at 12MBaud; with Repeater up to 1000m	
max. number of participants	32 Stations at every segment without repeater. With repeater expandable to 126.	
Diagnostic functions		
Standard diagnostic	Storage of the last 100 diagnostic in the Flash-ROM.	
Extended diagnostic	no	possible
Combination with periphery		
max. number of modules	32 in one row	
max. digital	32	
max. analog	16	
max. inputs	152Byte	244Byte
max. outputs	152Byte	244Byte
Dimensions and Weight		
Dimensions (WxHxD) in mm	40x125x120	
Weight	170g	

Chapter 4 CANopen

Overview

This chapter contains the description of the VIPA IM 353CAN CANopen slave module. The introduction to the system is followed by the description of the module.

Another section of this chapter concerns the project engineering for "experts" and an explanation of the telegram structure and the function codes of CANopen.

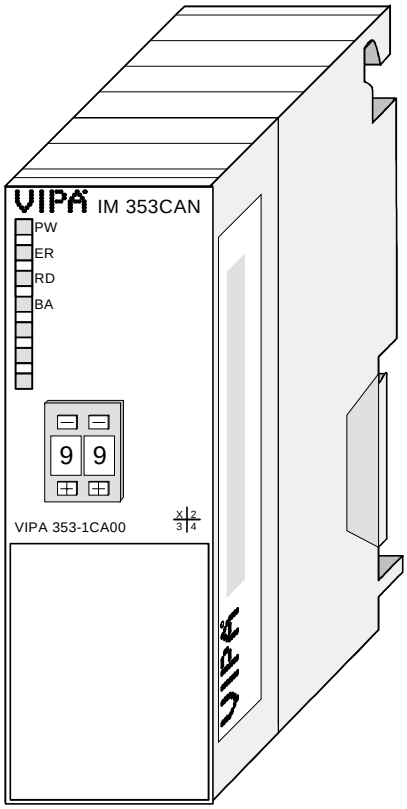
The chapter is finished by the description of the Emergency Object, NMT as well as the technical data.

Content

System overview

CANopen slave IM 353CAN

Currently the following CANopen bus couplers are available from VIPA:



Order data

Type	Order number	Description	Page
IM 353CAN	VIPA 353-1CA00	CAN-Bus CANopen slave	4-5

Basics

General

CANopen (**C**ontrol **A**rea **N**etwork) is an international standard for open fieldbus systems intended for building, manufacturing and process automation applications that was originally designed for automotive applications.

Due to its extensive error detection facilities, the CAN-Bus system is regarded as the most secure bus system. It has a residual error probability of less than $4.7x10^{-11}$. Bad messages are flagged and retransmitted automatically.

In contrast to Profibus and Interbus, CAN defines under the CAL-level-7-protocol (CAL=**C**AN **a**pplication **l**ayer) defines various level-7 user profiles for the CAN-Bus. One standard user profile defined by the CIA (**C**AN **i**n **A**utomation) e.V. is CANopen.

CANopen

CANopen is a user profile for industrial real-time systems, which is currently supported by a large number of manufacturers. CANopen was published under the heading of DS-301 by the CAN in Automation association (CIA). The communication specifications DS-301 define standards for CAN devices. These specifications mean that the equipment supplied by different manufacturers is interchangeable. The compatibility of the equipment is further enhanced by the equipment specification DS-401 that defines standards for the technical data and process data of the equipment. DS-401 contains the standards for digital and analog input/output modules.

CANopen comprises a communication profile that defines the objects that must be used for the transfer of certain data as well as the device profiles that specify the type of data that must be transferred by means of other objects.

The CANopen communication profile is based upon an object directory that is similar to the profile used by Profibus. The communication profile DS-301 defines two standard objects as well as a number of special objects:

- Process data objects (PDO)
  PDOs are used for real-time data transfers
- Service data objects (SDO)
  SDOs provide access to the object directory for read and write operations

Transfer medium

CAN is based on a linear bus topology. You can use router nodes to construct a network. The number of devices per network is only limited by the performance of the bus driver modules.

The maximum distance covered by the network is determined by the runtimes of the signals. This means that a data rate of 1Mbaud limits the network to 40m and 80kBaud limits the network to 1000m.

The CAN-Bus communication medium employs a screened three-core cable (optionally a five-core).

The CAN-Bus operates by means of differential voltages. For this reason it is less sensitive to external interference than a pure voltage or current based interface. The network must be configured as a serial bus, which is terminated by a 120Ω terminating resistor.

Your VIPA CAN-Bus coupler contains a 9pin socket. You must use this socket to connect the CAN-Bus coupler as a slave directly to your CAN-Bus network.

All devices on the network use the same baud rate.

Due to the bus structure of the network it is possible to connect or disconnect any station without interruption to the system. It is therefore also possible to commission a system in various stages. Extensions to the system do not affect the operational stations. Defective stations or new stations are recognized automatically.

Bus access method

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 and 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.

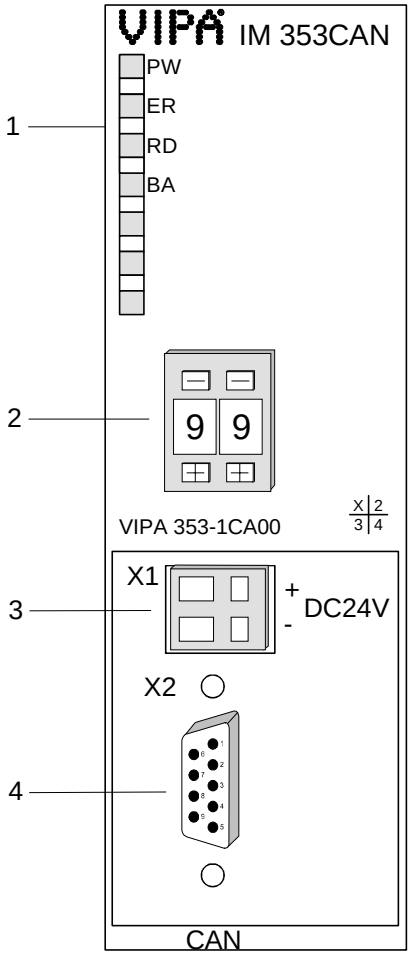
CAN-Bus access control is performed by means of a collision-free, bit-based arbitration algorithm. Collision-free means that the final winner of the arbitration process does not have to repeat his message. The station with the highest priority is selected automatically when more than one station accesses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

IM 353CAN - CANopen slave - Structure

Properties

- 10 Rx and 10 Tx PDOs
- 2 SDOs
- Support of all baud rates
- PDO linking
- PDO mapping

Front 353-1CA00



[1] LED status indicators
[2] Address or baud rate selector (Coding switch)
[3] Connector for an external 24V supply
[4] CAN-Bus socket

Components

LEDs

The module is equipped with four LEDs for diagnostic purposes. The following table shows how the diagnostic LEDs are used along with the respective colors.

Name	Color	Description
PW	yellow	Indicates that the supply voltage is available.
ER	red	On when an error was detected in the backplane bus communications.
RD	green	Blinks at 1Hz when the self-test was positive and initialization was OK.
		Is turned on when data is being communicated via the V-Bus.
BA	yellow	Off the self-test was positive and the initialization was OK.
		Blinks at 1Hz when the status is "Pre-operational".
		Is turned on when the status is "Operational".
		Blinks at 10Hz when the status is "Prepared".

Status indicator as a combination of LEDs

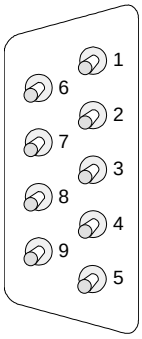
Various combinations of the LEDs indicate the different operating states:

PW on	Error during RAM or EEPROM initialization
ER on	
RD on	
BA on	
PW on	Baud rate setting activated
ER blinks 1Hz	
RD blinks 1Hz	
BA blinks 1Hz	
PW on	Error in the CAN baud rate setting
ER blinks 10Hz	
RD blinks 10Hz	
BA blinks 10Hz	
PW on	Module-ID setting activated
ER off	
RD blinks 1Hz	
BA off	

9pin D-type socket

The VIPA CAN-Bus coupler is connected to the CAN-Bus system by means of a 9pin socket.

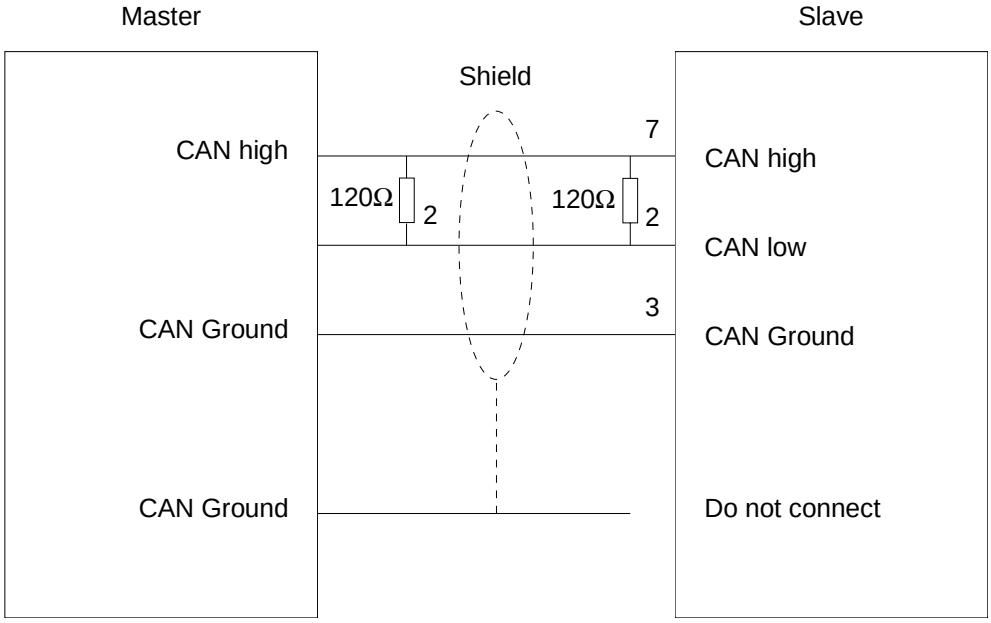
The following diagram shows the pin assignment for the interface.



Pin	Assignment
1	n.c.
2	CAN low
3	CAN ground
4	n.c.
5	n.c.
6	optional ground
7	CAN high
8	n.c.
9	optional pos. supply

CAN-Bus wiring

The CAN-Bus communication medium bus is a screened three-core cable.



Line termination

All stations on systems having more than two stations are wired in parallel. This means that the bus cable must be looped from station to station without interruptions.

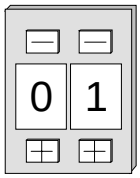
Note!

The end of the bus cable must be terminated with a 120Ω terminating resistor to prevent reflections and the associated communication errors!

Address selector for baud rate and module-ID

The address selector is used to specify the module-ID as well as the CAN baud rate. Each module ID must be unique on the bus.

For details please refer to "IM 353CAN - CANopen slave - Baud rate and module-ID" in this chapter.



Power supply

The CAN-bus coupler is equipped with an internal power supply. This power supply requires DC 24V. In addition to the internal circuitry of the bus coupler the supply voltage is also used to power any modules connected to the backplane bus. Please note that the maximum current that the integrated power supply can deliver to the backplane bus is 3.5A. The power supply is protected against reverse polarity.

CAN-Bus and backplane bus are isolated from each other.

Attention!

Please ensure that the polarity is correct when connecting the power supply!

IM 353CAN - CANopen slave - Fast introduction

Outline

This section is for experienced CANopen user that are already common with CAN. It will be shortly outlined, which messages are necessary for the deployment of the System 300V under CAN in the start configuration.

Note!

Please regard that this manual prints the hexadecimal numbers in the type for developers "0x".

e.g.: **0x**15AE = 15AE**h**

Adjusting baud rate and module-ID

Via the address selector you have to adjust a common baud rate at the bus couplers as well as different node-IDs.

After starting your power supply, you program the baud rate and the module-ID via 00 at the address selector within 10s.

For details please refer to the section under the heading "IM 353CAN - CANopen slave - Baud rate and module-ID" in this chapter.

CAN identifier

The CAN identifier for the in-/output data of the System 300V are generated from the node addresses (1...99):

Kind of data	Default CAN identifier	Kind of data	Default CAN identifier
digital inputs 1 ... 64Bit	0x180 + Node address	digital outputs 1 ... 64Bit	0x200 + Node address
analog inputs 1 ... 4 words	0x280 + Node address	analog outputs 1 ... 4 Words/Channels	0x300 + Node address
other digital or analog inputs	0x380 + Node address	other digital or analog outputs	0x400 + Node address
	0x480 + Node address		0x500 + Node address
	0x680 + Node address		0x780 + Node address
	0x1C0 + Node address		0x240 + Node address
	0x2C0 + Node address		0x340 + Node address
	0x3C0 + Node address		0x440 + Node address
	0x4C0 + Node address		0x540 + Node address
	0x6C0 + Node address		0x7C0 + Node address

Digital in-/outputs

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

Identifier 0x180+Node address + up to 8Byte user data

Identifier 11Bit	**DI 0** 8Bit	**DI 1** 8Bit	**DI 2** 8Bit	**...**	**DI 7** 8Bit

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

Identifier 0x200+Node address + up to 8Byte user data

Identifier 11Bit	**DO 0** 8Bit	**DO 1** 8Bit	**DO 3** 8Bit	**...**	**DO 7** Bit

Analog in-/outputs

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

Identifier 0x280+Node address + up to 4Words user data

Identifier 11Bit	**AI 0** 1Word	**AI 1** 1Word	**AI 2** 1Word	**AI 3** 1Word

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

Identifier 0x300+Node address + up to 4Words user data

Identifier 11Bit	**AI 0** 1Word	**AI 1** 1Word	**AI 2** 1Word	**AI 3** 1Word

Node Guarding

For the System 300V works per default in event-controlled mode (no cyclic DataExchange), a node failure is not always immediately detected. Remedy is the control of the nodes per cyclic state request (Node Guarding).

You request cyclically a state telegram via Remote-Transmit-Request (RTR): the telegram only consists of a 11Bit identifier:

Identifier 0x700+Node address

Identifier 11Bit

The System 300V node answers with a telegram that contains one state byte:

Identifier 0x700+Node address + State 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 send

To enable the bus coupler to recognize a network master failure (watchdog function), you still have to set the Guard-Time (Object 0x100C) and the Life-Time-Factor (Object 0x100D) to values≠0.

(reaction time at failure: Guard-Time x Life Time Factor).

Heartbeat

Besides the Node Guarding, the System 300V CANopen coupler also supports the Heartbeat Mode.

If there is a value set in the index 0x1017 (Heartbeat Producer Time), the device state (Operational, Pre-Operational, ...) is transferred when the Heartbeat-Timer run out by using the COB identifier (0x700+Module-Id):

Identifier 0x700+Node address + State byte

Identifier 11Bit	**Status** 8Bit

The Heartbeat Mode starts automatically as soon as there is a value in index 0x1017 higher 0.

Emergency Object

To send internal device failures to other participants at the CAN-Bus with a high priority, the VIPA CAN-Bus coupler supports the Emergency Object.

To activate the emergency telegram, you need the **COB-Identifier** that is fixed after boot-up in the object directory of the variable 0x1014in hexadecimal view: **0x80 + Module-ID.**

The emergency telegram has always a length of 8Byte. It consists of:

Identifier 0x80 + Node address + 8Byte user data

Identifier 11Bit	**EC0**	**EC1**	**Ereg**	**Inf0**	**Inf1**	**Inf2**	**Inf3**	**Inf4**

Error Code	Meaning	Info 0	Info 1	Info 2	Info 3	Info4
0x0000	Reset Emergency	0x00	0x00	0x00	0x00	0x00
0x1000	Module Configuration has changed and Index 0x1010 is equal to 'save'	0x06	0x00	0x00	0x00	0x00
0x1000	Module Configuration has changed	0x05	0x00	0x00	0x00	0x00
0x1000	Error during initialization of backplane modules	0x01	0x00	0x00	0x00	0x00
0x1000	Error during module configuration check	0x02	Module Number	0x00	0x00	0x00
0x1000	Error during read/write module	0x03	Module Number	0x00	0x00	0x00
0x1000	Module parameterization error	0x30	Module Number	0x00	0x00	0x00
0x1000	Diagnostic alarm from an analog module	0x40 + Module Number	diagnostic byte 1	diagnostic byte 2	diagnostic byte 3	diagnostic byte 4
0x1000	Process alarm from an analog module	0x80 + Module Number	diagnostic byte 1	diagnostic byte 2	diagnostic byte 3	diagnostic byte 4

continued ...

... continue Emergency object

Error Code	Meaning	Info 0	Info 1	Info 2	Info 3	Info4
0x1000	PDO Control	0xFF	0x10	PDO Number	LowByte Timer Value	HighByte Timer Value
0x5000	Module					
0x6300	SDO PDO-Mapping	LowByte MapIndex	HighByte MapIndex	No. Of Map Entries	0x00	0x00
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!

The now described telegrams enable you to start and stop the System 300V, read inputs, write outputs and control the modules.

In the following, the functions are described in detail.

IM 353CAN - CANopen slave - Baud rate and module-ID

Outline

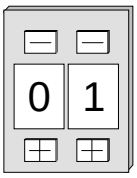
You have the option to specify the baud rate and the module-ID by setting the address selector to 00 within a period of 10s after you have turned the power on.

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

Specifying the baud rate by means of 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 by means of the address selector:



Address selector	CAN baud rate	max. guar. bus distance
"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 baud rate is saved in the EEPROM.

Module-ID selection

- LEDs ER and BA are turned off and the red RD-LED continues to blink.

  At this point you have 5s to enter the required 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-IDs are accepted when a period of 5s has expired after which the bus coupler returns to the normal operating mode (status: "Pre-Operational").

Baud rate selection by an SDO-write operation

You can also modify the CAN baud rate by means of an SDO-Write operation to the object "2001h". The entered value is used as the CAN baud rate when the bus coupler has been RESET. This method is a most convenient when you must change the CAN baud rate of all the bus couplers of a system from a central CAN terminal. The bus couplers use the programmed baud rate when the system has been RESET.

IM 353CAN - CANopen slave - Message structure

Identifier

All CANopen messages have the following structure according to CiA DS-301:

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

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 IM 353 CAN supports the following objects:

- 10 transmit PDOs (PDO Linking, PDO Mapping)
- 10 receive PDOs (PDO Linking, PDO Mapping)
- 2 standard SDOs
- 1 emergency object
- 1 network management object NMT
- Node Guarding
- Heartbeat

CANopen function codes

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

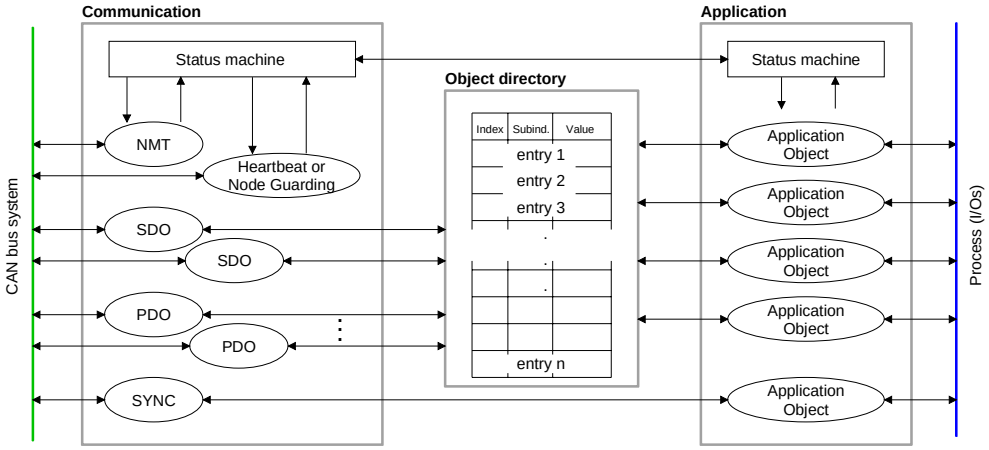
Object	Function code (4 bits)	Receiver	Definition	Function
NMT	0000	Broadcast	CiA DS-301	Network management
EMERGENCY	0001	Master	CiA DS-301	Error message
PDO1S2M	0011	Master, Slave (RTR)	CiA DS-301	Digital input data 1
PDO1M2S	0100	Slave	CiA DS-301	Digital output data 1
SDO1S2M	1011	Master	CiA DS-301	Configuration data
SDO1M2S	1100	Slave	CiA DS-301	Configuration data
Node Guarding	1110	Master, Slave (RTR)	CiA DS-301	Module monitoring
Heartbeat	1110	Master, Slave	Application spec.	Module monitoring

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 DS-401 Version 1.4".

Structure of the device model

A CANopen device can be structured as follows:



Communication

Serves the communication data objects and the concerning functionality for data transfer via the CANopen network.

Application

The application data objects contain e.g. in- and output data. In case of an error, an application status machine switches the outputs in a secure state.

The object directory is organized as 2 dimension table. The data is addressed via index and sub-index.

Object directory

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

IM 353CAN - CANopen slave - PDO

PDO

In many fieldbus systems the whole process image is transferred - mostly more or less cyclically. CANopen is not limited to this communication principle, for CAN supports more possibilities through multi master bus access coordination.

CANopen divides the process data into segments of max. 8Byte. These segments are called **p**rocess **d**ata **o**bjects (PDOs). Every PDO represents one CAN telegram and is identified and prioritized via its specific CAN identifier.

For the exchange of process data, the VIPA CAN-Bus coupler IM 353CAN supports 20 PDOs. Every PDO consists of a maximum of 8 data bytes. The transfer of PDOs is not verified by means of acknowledgments since the CAN protocol guarantees the transfer.

There are 10 Tx transmit PDOs for input data and 10 Rx receive PDOs for output data. The PDOs are named seen from the bus coupler:

Receive PDOs (RxPDOs) are received by the bus coupler and contain output data.

Transmit PDOs (TxPDOs) are send by the bus coupler and contain input data.

The assignment of the PDOs to input or output data occurs automatically.

Variable PDO mapping

CANopen predefines the first two PDOs in the device profile. The assignment of the PDOs is fixed in the mapping tables in the object directory. The mapping tables are the cross-reference between the application data in the object directory and the sequence in the PDOs.

The assignment of the PDOs, automatically created by the coupler, are commonly adequate. For special applications, the assignment may be changed. Herefore you have to configure the mapping tables accordingly.

First, you write a 0 to sub-index 0 (deactivates the current mapping configuration). Then you insert the wanted application objects into sub-index 1...8. Finally you parameterize the number of now valid entries in sub-index 0 and the coupler checks the entries for their consistency.

PDO identifier COB-ID

The most important communication parameter of a PDOs is the CAN identifier (also called "Communication Object Identifier", COB-ID). It serves the identification of the data and sets the priority of bus access.

For every CAN data telegram only one sending node may exist (producer). Due to the ability of CAN to send all messages per broadcast procedure, however, a telegram may be received by several bus participants at the same time (consumer). Therefore, one node may deliver its input information to different bus stations similarly - without needing the pass through a logical bus master.

The System 300V provides receive and transmit PDOs default identifier in dependence of the node address.

Below follows a list of the COB identifiers for the receive and the transmit PDO transfer that are pre-set after boot-up.

The transmission type in the object directory (indices 0x1400-0x1409 and 0x1800-0x1809, sub-index 0x02) is preset to asynchronous, event controlled (= 0xFF). The EVENT-timer (value * 1ms) can be used to transmit the PDOs cyclically.

Send:	0x180 + module-ID: PDO1S2M digital	(acc. DS-301)
	0x280 + module-ID: PDO2S2M analog	
	0x380 + module-ID: PDO3S2M digital or analog	
	0x480 + module-ID: PDO4S2M	
	0x680 + module-ID: PDO5S2M	
	0x1C0 + module-ID: PDO6S2M	
	0x2C0 + module-ID: PDO7S2M	
	0x3C0 + module-ID: PDO8S2M	
	0x4C0 + module-ID: PDO9S2M	
	0x6C0 + module-ID: PDO10S2M	
Receive:	0x200 + module-ID: PDO1M2S digital	(acc. DS-301)
	0x300 + module-ID: PDO2M2S analog	
	0x400 + module-ID: PDO3M2S digital or analog	
	0x500 + module-ID: PDO4M2S	
	0x780 + module-ID: PDO5M2S	
	0x240 + module-ID: PDO6M2S	
	0x340 + module-ID: PDO7M2S	
	0x440 + module-ID: PDO8M2S	
	0x540 + module-ID: PDO9M2S	
	0x7C0 + module-ID: PDO10M2S	

PDO linking

If the Consumer-Producer model of the CANopen PDOs shall be used for direct data transfer between nodes (without master), you have to adjust the identifier distribution accordingly, so that the TxPDO identifier of the producer is identical with the RxPDO identifier of the consumer:

This procedure is called PDO linking. this enables for example the simple installation of electronic gearing where several slave axis are listening to the actual value in TxPDO of the master axis.

PDO Communication types

CANopen supports the following possibilities for the process data transfer:

- Event triggered
- Polled
- Synchronized

Event triggered

The "event" is the alteration of an input value, the data is send immediately after value change. The event control makes the best use of the bus width for not the whole process image is send but only the changed values. At the same time, a short reaction time is achieved, because there is no need to wait for a master request.

Polled

PDOs may also be polled via data request telegrams (remote frames) to give you the opportunity to e.g. send the input process image of event triggered inputs to the bus without input change for example a monitoring or diagnosis device included during runtime.

The VIPA CANopen bus couplers support the query of PDOs via remote frames - for this can, due to the hardware, not be granted for all CANopen devices, this communication type is only partially recommended.

Synchronized

It is not only convenient for drive applications to synchronize the input information request and the output setting. For this purpose, CANopen provides the SYNC object, a CAN telegram with high priority and no user data which receipt is used by the synchronized nodes as trigger for reading of the inputs res. writing of the outputs.

PDO transmission type

The parameter "PDO transmission type" fixes how the sending of the PDOs is initialized and what to do with received ones:

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

Synchronous

The transmission type 0 is only wise for RxPDOs: the PDO is analyzed at receipt of the next SYNC telegram.

At transmission type 1-240, the PDO is send res. expected cyclically: after every "n^{th}" SYNC (n=1...240). For the transmission type may not only be combined within the network but also with a bus, you may thus e.g. adjust a fast cycle for digital inputs (n=1), while data of the analog inputs is transferred in a slower cycle (e.g. n=10). The cycle time (SYNC rate) may be monitored (Object 0x1006), at SYNC failure, the coupler sets its outputs in error state.

Asynchronous

The transmission types 254 + 255 are asynchronous or also event triggered. The transmission type 254 provides an event defined by the manufacturer, at 255 it is fixed by the device profile.

When choosing the event triggered PDO communication you should keep in mind that in certain circumstances there may occur a lot of events similarly. This may cause according delay times for sending PDOs with lower priority values.

You should also avoid to block the bus by assigning a high PDO priority to an often alternating input ("babbling idiot").

Inhibit time

Via the parameter "inhibit time" a "send filter" may be activated that does not lengthen the reaction time of the relatively first input alteration but that is active for the following changes.

The inhibit time (send delay time) describes the min. time span that has to pass between the sending of two identical telegrams.

When you use the inhibit time, you may ascertain the max. bus load and for this the latent time in the "worst case".

IM 353CAN - CANopen slave - SDO

SDO

The **S**ervice **D**ata **O**bject (SDO) serves the read or write access to the object directory. The CAL layer 7 protocol gives you the specification of the Multiplexed-Domain-Transfer-Protocol that is used by the SDOs. This protocol allows you to transfer data of any length because where appropriate, messages are distributed to several CAN messages with the same identifier (segment building).

The first CAN message of the SDO contain process information in 4 of the 8 bytes. For access to object directory entries with up to 4Byte length, one single CAN message is sufficient. The following segments of the SDO contain up to 7Byte user data. The last Byte contains an end sign. A SDO is delivered with acknowledgement, i.e. every reception of a message is receipted.

The COB identifiers for read and write access are:

- Receive-SDO1: 0x600 + Module-ID
- Transmit-SDO1: 0x580 + Module-ID

Note!

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

In the following only the error messages are described that are generated at wrong parameterization.

SDO error codes

Code	Error
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 an 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 directory dynamic generation fails or no object directory is present (e.g. object directory is generated from file and generation fails because of an file error)

IM 353CAN - CANopen slave - Object directory

Structure

The CANopen object directory contains all relevant CANopen objects for the bus coupler. Every entry in the object directory is marked by a 16Bit index.

If an object exists of several components (e.g. object type Array or Record), the components are marked via an 8Bit sub-index.

The object name describes its function. The data type attribute specifies the data type of the entry.

The access attribute defines, if the entry may only be read, only be written or read and written.

The object directory is divided into the following 3 parts:

Communication specific profile area (0x1000 – 0x1FFF)

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

0x1000 – 0x1029	General communication specific parameters (e.g. device name)
0x1400 – 0x1409	Communication parameters (e.g. identifier) of the receive PDOs
0x1600 – 0x1609	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 depending object.
0x1800 – 0x1809 0x1A00 – 0x1A09	Communication and mapping parameters of the transmit PDOs

Manufacturer specific profile area (0x2000 – 0x5FFF)

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

Standardized device profile area (0x6000 – 0x9FFF)

This area contains the objects for the device profile acc. DS-401.

Note!

For the CiA norms are exclusively available in English, we adapted the object tables. Some entries are described below the according tables.

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
0x1007		Synchronous windows length
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	Consumer heartbeat time
0x1017	X	Producer heartbeat time
0x1018		Identity object
0x1027		Module list
0x1029		Error behavior
0x1400 - 0x1409	X	Communication parameter RxPDOs (master to slave)
0x1600 - 0x1609	X	Mapping parameter RxPDOs
0x1800 - 0x1809	X	Communication parameter TxPDOs (slave to master)
0x1A00 - 0x1A09	X	Mapping parameter TxPDOs
0x2001		CAN baud rate
0x2100		Kill EEPROM
0x2101		SJA1000 message filter
0x2400	X	PDO control
0x3001 - 0x3010	X	Module parameterization
0x3401	X	Module parameterization
0x6000		Digital input 8 bit (see DS 401)
0x6002	X	Polarity digital input 8 bit (see DS 401)
0x6100		Digital input 16 bit (see DS 401)
0x6102		Polarity digital input 16 bit (v DS 401)
0x6120		Digital input 32 bit (see DS 401)
0x6122		Polarity digital input 32 bit (see DS 401)
0x6200		Digital output 8 bit (see DS 401)
0x6202	X	Polarity digital output 8 bit (see DS 401)
0x6206	X	Fault mode digital output 8 bit (see DS 401)
0x6207	X	Fault state digital output 8 bit (see DS 401)
0x6300		Digital output 16 bit (see DS 401)

continue ...

... continued object directory overview

Index		Content of Object
0x6302		Polarity digital output 16 bit (see DS 401)
0x6306		Fault mode digital output 16 bit (see DS 401)
0x6307		Fault state digital output 16 bit (see DS 401)
0x6320		Digital output 32 bit (see DS 401)
0x6322		Polarity digital output 32 bit (see DS 401)
0x6326		Fault mode digital output 32 bit (see DS 401)
0x6327		Fault state digital output 32 bit (see DS 401)
0x6401		Analog input (see DS 401)
0x6411		Analog output (see DS 401)
0x6421	X	Analog input interrupt trigger (see DS 401)
0x6422		Analog input interrupt source (see DS 401)
0x6423	X	Analog input interrupt enable (see DS 401)
0x6424	X	Analog input interrupt upper limit (see DS 401)
0x6425	X	Analog input interrupt lower limit (see DS 401)
0x6426	X	Analog input interrupt delta limit (see DS 401)
0x6443	X	Fault mode analog output (see DS 401)
0x6444	X	Fault state analog output (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	0x00050191	Statement of device type

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

MSB	LSB
Additional information device	**Profile number**
0000 0000 0000 wxyz (bit)	401dec=0x0191

The "additional information" contains data related to the signal types of the I/O device:

z=1 → digital inputs

y=1 → digital outputs

x=1 → analog inputs

w=1 → analog outputs

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.: Manufacturer specific error, specified in object 0x1003.

Comm.: Communication error (overrun CAN)

Generic: A not more precisely 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
	...	...	...	...	...	...	...
	254		Unsigned32	ro	N		A maximum of 254 error states

The "predefined error field" is divided into two 16Bit fields:

MSB	LSB
Additional information	**Error code**

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

New errors are always saved at sub-index 1, all the other sub-indices being appropriately incremented.

By writing a "0" to sub-index 0, the whole error memory is cleared. If there has not been an error since PowerOn, then object 0x1003 exists only of sub-index 0 with entry "0".

Via reset or PowerCycle, the error memory is cleared.

Number of PDOs

Index	Sub index	Name	Type	Attr.	Map.	Default value	Meaning
0x1004	0	Number of PDOs supported	Unsigned32	ro	N	0x000A000A	Number of PDOs supported
	1	Number of synchronous PDOs supported	Unsigned32	ro	N	0x000A000A	Number of synchronous PDOs supported
	2	Number of asynchronous PDOs supported	Unsigned32	ro	N	0x000A000A	Number of asynchronous PDOs supported

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

MSB	LSB
Number of receive (Rx)PDOs supported	**Number of send (Tx)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 11Bit of the 32Bit value contain the identifier (0x80=128dez), while the MSBit indicates whether the device receives the SYNC telegram (1) or not (0).

Attention: In contrast to the PDO identifiers, the MSB being set indicates 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 µs.

If a value other than zero is entered here, the coupler goes into error state if no SYNC telegram is received within the set time during 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 µs.

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 IM 353 1CA00 = VIPA CANopen slave IM 353-1CA00

Since the returned value is longer 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 IM 353 1CA00 = 1.00

Since the returned value is longer 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 IM 353 1CA00 = 3.xx

Since the returned value is longer 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. Is 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 a guarding telegram is not received within the life time, the node enters the error state. If the life time factor and/or guard time =0, the node does not carry out any life guarding.

Node 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 parameter

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 string "save" in ASCII code (hex code: 0x65766173) into sub-index 1, the current parameters are placed into non-volatile storage (byte sequence at the bus incl. SDO protocol: 0x23 0x10 0x10 0x01 0x73 0x61 0x76 0x65).

If successful, the storage process is confirmed by the corresponding TxSDO (0x60 in the first byte).

Note!

For the bus coupler is not able to send or receive CAN telegrams during the storage procedure, storage is only possible when the node is in pre-operational state.

It is recommended to set the complete net to the pre-operational state before storing data to avoid a buffer overrun.

Load parameter

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 string "load" in ASCII code (hex code: 0x64616F6C) into sub-index 1, all parameters are set back to default values (delivery state) **at next start-up (reset)** (byte sequence at the bus incl. SDO protocol: 0x23 0x11 0x10 0x01 0x6C 0x6F 0x61 0x64).

This activates the default identifiers for the PDOs.

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 unequal zero for the same node-ID, the node interrupts the SDO download and throws the error code 0604 0043hex.

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	0xAFFEAFFE	Vendor ID
	2	Product Code	Unsigned32	ro	N	*	Product Code
	3	Revision Number	Unsigned32	ro	N		Revision Number
	4	Serial Number	Unsigned32	ro	N		Serial Number

*) Default value Product Code: at 353-1CA00: 0x3531CA00

Module list

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1027	0	Number of connected modules	Unsigned8	ro	N		Contains general information about the device (number of entries)
	1	Module 1	Unsigned16	ro	N		Identification number of Module 1
	...	...	...	...	...	...	...
	N	Module N	Unsigned16	ro	N		Identification number of Module N

Module types

Module type	Identification (hex)	No. of Digital Input-Byte	No. of Digital Input-Byte
DI 8	9FC1h	1	-
DI 8 - Alarm	1FC1h	1	-
DI 16	9FC2h	2	-
DI 16 / 1C	08C0h	6	6
DI 32	9FC3h	4	-
DO 8	AFC8h	-	1
DO 16	AFD0h	-	2
DO 32	AFD8h	-	4
DIO 8	BFC9h	1	1
DIO 16	BFD2h	2	2
AI2	15C3h	4	-
AI4	15C4h	8	-
AI4 - fast	11C4h	8	-
AI8	15C5h	16	-
AO2	25D8h	-	4
AO4	25E0h	-	8
AO8	25E8h	-	16
AI2 / AO2	45DBh	4	4
AI4 / AO2	45DCh	8	4

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 failure is detected in "operational" state, the module should automatically change into the "pre-operational" state.

If e.g. an "Error behavior" is implemented, the module may be configured that its going into STOP at errors.

The following error classes may be monitored:

0 = pre-operational

1 = no state change

2 = stopped

3 = reset after 2 seconds

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	0xC0000200 + NODE_ID	COB-ID RxPDO1
	2	Transmis-sion type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

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

The sub-index 2 contains the transmission type.

Communication parameter RxPDO2

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1401	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	0xC0000300 + NODE_ID	COB-ID RxPDO2
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

Communication parameter RxPDO3

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1402	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	0xC0000400 + NODE_ID	COB-ID RxPDO3
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

Communication parameter RxPDO4

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1403	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	0xC0000500 + NODE_ID	COB-ID RxPDO4
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

Communication parameter RxPDO5

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1404	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	0xC0000780 + NODE_ID	COB-ID RxPDO5
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

Communication parameter RxPDO6

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1405	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	0xC0000240 + NODE_ID	COB-ID RxPDO6
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

Communication parameter RxPDO7

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1406	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	0xC0000340 + NODE_ID	COB-ID RxPDO7
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

Communication parameter RxPDO8

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1407	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	0xC0000440 + NODE_ID	COB-ID RxPDO8
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

Communication parameter RxPDO9

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1408	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	0xC0000540 + NODE_ID	COB-ID RxPDO9
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

Communication parameter RxPDO10

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1409	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	0xC00007C0 + NODE_ID	COB-ID RxPD10
	2	transm. type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

Mapping parameter RxPDO1

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1600	0	Number of Elements	Unsigned8	rw	N	0x01	Mapping parameter of the first receive PDO; sub-index 0: number of mapped objects
	1	1st mapped object	Unsigned32	rw	N	0x62000108	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2nd mapped object	Unsigned32	rw	N	0x62000208	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8th mapped	Unsigned32	rw	N	0x62000808	(2 byte index, 1 byte sub-index, 1 byte bit-width)

The first receive PDO (RxPDO1) is per default for the digital outputs. Depending on the number of the inserted outputs, the needed length of the PDO is calculated and mapped into the according objects.

For the digital outputs are organized in bytes, the length of the PDO can be directly seen in sub-index 0.

If the mapping is changed, the entry in sub-index 0 has to be adjusted accordingly.

Mapping parameter RxPDO2

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1601	0	Number of Elements	Unsigned8	rw	N	0x01	Mapping parameter of the second receive PDO; sub-index 0: number of mapped objects
	1	1st mapped object	Unsigned32	rw	N	0x64110110	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2nd mapped object	Unsigned32	rw	N	0x64110210	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8th mapped	Unsigned32	rw	N	0x00000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)

The 2nd receive PDO (RxPDO2) is per default for the analog outputs. Depending on the number of the inserted outputs, the needed length of the PDO is calculated and the according objects are mapped.

For the digital outputs are organized in words, the length of the PDO can be directly seen in sub-index 0.

If the mapping is changed, the entry in sub-index 0 has to be adjusted accordingly.

Mapping parameter RxPDO3-RxPDO10

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1602 - 0x1609	0	Number of Elements	Unsigned8	rw	N	0x01	Mapping parameter of the 3rd to 10th receive PDO; sub-index 0: number of mapped objects
	1	1st mapped object	Unsigned32	rw	N	0x00000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2 nd mapped object	Unsigned32	rw	N	0x00000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8th mapped	Unsigned32	rw	N	0x00000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)

The receive PDOs 3 to 10 (RxPDO3) get an automatic default mapping via the coupler depending from the connected terminals. The procedure is described under "PDO mapping".

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	0x80000180 + 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 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

Sub-index 1 (COB-ID): The lower 11Bit of the 32Bit value (Bits 0-10) contain the CAN identifier, the MSBit (Bit 31) shows if the PDO is active (1) or not (0), Bit 30 shows if a RTR access to this PDO is permitted (0) or not (1). The sub-index 2 contains the transmission type, sub-index 3 the repetition delay time between two equal PDOs. If an event timer exists with a value unequal 0, the PDO is transmitted when the timer exceeds.

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

Communication parameter TxPDO2

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1801	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter of the second transmit PDO, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x80000280 + NODE_ID	COB-ID TxPDO2
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

Communication parameter TxPDO3

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1802	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter for the 3rd transmit PDO.
	1	COB-ID	Unsigned32	rw	N	0x80000380 + NODE_ID	COB-ID TxPDO3
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

Communication parameter TxPDO4

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1803	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter for the 4th transmit PDO.
	1	COB-ID	Unsigned32	rw	N	0x80000480 + NODE_ID	COB-ID TxPDO4
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

Communication parameter TxPDO5

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1804	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter for the 5th transmit PDO.
	1	COB-ID	Unsigned32	rw	N	0x80000680 + NODE_ID	COB-ID TxPDO5
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

Communication parameter TxPDO6

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1805	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter for the 6th transmit PDO.
	1	COB-ID	Unsigned32	rw	N	0x800001C0 + NODE_ID	COB-ID TxPDO6
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

Communication parameter TxPDO7

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1806	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter for the 7th transmit PDO.
	1	COB-ID	Unsigned32	rw	N	0x800002C0 + NODE_ID	COB-ID TxPDO7
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

Communication parameter TxPDO8

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1807	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter for the 8th transmit PDO.
	1	COB-ID	Unsigned32	rw	N	0x800003C0 + NODE_ID	COB-ID TxPDO8
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

Communication parameter TxPDO9

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1808	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter for the 9th transmit PDO.
	1	COB-ID	Unsigned32	rw	N	0x800004C0 + NODE_ID	COB-ID TxPDO9
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

Communication parameter TxPDO10

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1809	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter for the 10th transmit PDO.
	1	COB-ID	Unsigned32	rw	N	0x800006C0 + NODE_ID	COB-ID TxPDO10
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

Mapping parameter TxPDO1

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1A00	0	Number of Elements	Unsigned8	rw	N	depending on the components fitted	Mapping parameter of the first transmit PDO; sub-index 0: number of mapped objects
	1	1st mapped object	Unsigned32	rw	N	0x60000108	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2nd mapped object	Unsigned32	rw	N	0x60000208	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x60000808	(2 byte index, 1 byte sub-index, 1 byte bit-width)

continue ...

... continue Mapping parameter TxPD01

The first send PDO (TxPDO1) is per default for digital inputs. Depending on the number of the inserted inputs, the needed length of the PDO is calculated and the according objects are mapped.

For the digital inputs are organized in bytes, the length of the PDO can be directly seen in sub-index 0.

If the mapping is changed, the entry in sub-index 0 has to be adjusted accordingly.

Mapping parameter TxPDO2

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1A01	0	Number of Elements	Unsigned8	rw	N	depending on the components fitted	Mapping parameter of the second transmit PDO; sub-index 0: number of mapped objects
	1	1st mapped object	Unsigned32	rw	N	0x64010110	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2nd mapped object	Unsigned32	rw	N	0x64010210	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x00000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)

The 2nd send PDO (RxPDO2) is per default for the analog inputs. Depending on the number of the inserted outputs, the needed length of the PDO is calculated and the according objects are mapped.

For the digital outputs are organized in words, the length of the PDO can be directly seen in sub-index 0.

If the mapping is changed, the entry in sub-index 0 has to be adjusted accordingly.

Mapping parameter TxPDO3-TxPDO10

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1A02 - 0x1A09	0	Number of Elements	Unsigned8	rw	N	depending on the components fitted	Mapping parameter of the 3rd to 10 th transmit PDO; sub-index 0: number of mapped objects
	1	1st mapped object	Unsigned32	rw	N	0x00000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2nd mapped object	Unsigned32	rw	N	0x00000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x000000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)

The send PDOs 3 to 10 (RxPDO3) get an automatic default mapping via the coupler depending from the connected terminals. The procedure is described under "PDO mapping".

CAN baud rate

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

This index entry writes a new baud rate into the EEPROM.

At the next start-up (reset) the CAN coupler starts with the new baud rate.

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

KILL EEPROM

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x2100	0	KILL EEPROM	Boolean	wo	N		KILL EEPROM

The KILL EEPROM is supported for reasons of compatibility.

Writing to index 0x2100 deletes all stored identifiers from the EEPROM.

The CANopen coupler start **at the next start-up (reset)** with the default configuration.

SJA1000 message filter

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x2101	0	Number of Elements	Unsigned8	ro	N	0x02	SJA1000 Message Filter
	1	Acceptance mask	Unsigned8	ro	N		Acceptance mask
	2	Acceptance code	Unsigned8	ro	N		Acceptance code

With the help of the acceptance filter, the CAN controller is able to allow passing of received messages to the RXFIFO only when the identifier bits of the received message are equal to the predefined ones within the acceptance filter. The acceptance filter is defined via the acceptance code register and the acceptance mask register.

These filters are updated after start-up and communication reset.

Acceptance mask: The acceptance mask register qualifies which of the corresponding bits of the acceptance code are relevant (AM.X = 0) and which ones are ‘don’t care’ (AM.X = 1) for acceptance filtering.

Acceptance code: The acceptance code bits (AC.7 to AC.0) and the eight most significant bits of the message identifier (ID.10 to ID.3) have to be in the same bit positions which are marked as relevant by the acceptance mask bits (AM.7 to AM.0). If the following condition is fulfilled, the messages are accepted:

$0(ID.10 \text{ to } ID.3) \equiv (AC.7 \text{ to } AC.0)] \vee (AM.7 \text{ to } AM.0) \equiv 11111111$

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]
	2	RxPDO2	Unsigned16	rw	N	0x0000	Timer value [ms]
	...	...	...	...	...	...	...
	10	RxPDO10	Unsigned16	rw	N	0x0000	Timer value [ms]

The control starts as soon as the timer is unequal 0. Every received RxPDO resets the timer. When the timer has been expired, the CAN coupler switches into the state "pre-operational" and sends an emergency telegram.

This function is exclusively suitable for master, where heartbeat or rather node guarding is not supported.

Module parameterization

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x3001 - 0x3010	0	Number of Elements	Unsigned8	ro	N	0x04 or 0x00	Number of entries 0x04 : module available 0x00 : no module available
	1	Prm 0 to 3	Unsigned32	rw	N	depending on the compo-nents fitted	Parameter bytes 0 to 3
	2	Prm 4 to 7	Unsigned32	rw	N	depending on the compo-nents fitted	Parameter bytes 4 to 7
	3	Prm 8 to 11	Unsigned32	rw	N	depending on the compo-nents fitted	Parameter bytes 8 to 11
	4	Prm 12 to 15	Unsigned32	rw	N	depending on the compo-nents fitted	Parameter bytes 12 to 15

Via the indices 0x3001 to 0x3010 you may parameterize the analog modules, counter and communication modules.

Default configuration

AI4	0x00, 0x00, 0x28, 0x28, 0x28, 0x28, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
AI8	0x00, 0x00, 0x26, 0x26, 0x26, 0x26, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
AO4	0x00, 0x00, 0x09, 0x09, 0x09, 0x09, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
AI/AO	0x00, 0x00, 0x09, 0x09, 0x09, 0x09, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
CP 240	0x00, 0x00, 0x00, 0x00, 0x00, 0x13, 0x06, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
FM 250	0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
FM 254	0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00

Example 1 **Set AI4 to mode 0x2C**

Read default configuration

Read SubIndex 0 M2S: 0x40 0x01 0x30 0x00 0x00 0x00 0x00 0x00
S2M: 0x4F 0x01 0x30 0x00 0x04 0x00 0x00 0x00
Read SubIndex 1 M2S: 0x40 0x01 0x30 0x01 0x00 0x00 0x00 0x00
S2M: 0x43 0x01 0x30 0x01 0x00 0x00 0x28 0x28
Read SubIndex 2 M2S: 0x40 0x01 0x30 0x02 0x00 0x00 0x00 0x00
S2M: 0x43 0x01 0x30 0x02 0x28 0x28 0x00 0x00
Read SubIndex 3 M2S: 0x40 0x01 0x30 0x03 0x00 0x00 0x00 0x00
S2M: 0x43 0x01 0x30 0x03 0x00 0x00 0x00 0x00
Read SubIndex 4 M2S: 0x40 0x01 0x30 0x04 0x00 0x00 0x00 0x00
S2M: 0x43 0x01 0x30 0x04 0x00 0x00 0x00 0x00

Write new configuration

Write SubIndex 1 M2S: 0x23 0x01 0x30 0x01 0x00 0x00 0x2C 0x2C
S2M: 0x60 0x01 0x30 0x01 0x00 0x00 0x00 0x00
Write SubIndex 2 M2S: 0x23 0x01 0x30 0x02 0x2C 0x2C 0x00 0x00
S2M: 0x60 0x01 0x30 0x02 0x00 0x00 0x00 0x00

Read new configuration

Read SubIndex 0 M2S: 0x40 0x01 0x30 0x00 0x00 0x00 0x00 0x00
S2M: 0x4F 0x01 0x30 0x00 0x04 0x00 0x00 0x00
Read SubIndex 1 M2S: 0x40 0x01 0x30 0x01 0x00 0x00 0x00 0x00
S2M: 0x43 0x01 0x30 0x01 0x00 0x00 0x2C 0x2C
Read SubIndex 2 M2S: 0x40 0x01 0x30 0x02 0x00 0x00 0x00 0x00
S2M: 0x43 0x01 0x30 0x02 0x2C 0x2C 0x00 0x00
Read SubIndex 3 M2S: 0x40 0x01 0x30 0x03 0x00 0x00 0x00 0x00
S2M: 0x43 0x01 0x30 0x03 0x00 0x00 0x00 0x00
Read SubIndex 4 M2S: 0x40 0x01 0x30 0x04 0x00 0x00 0x00 0x00
S2M: 0x43 0x01 0x30 0x04 0x00 0x00 0x00 0x00

Example 2 **Set FM250 to Counter Mode 0x08 and 0x0B**

Read default configuration

Read SubIndex 0	M2S: 0x40 0x02 0x30 0x00 0x00 0x00 0x00 0x00
	S2M: 0x4F 0x02 0x30 0x00 0x04 0x00 0x00 0x00
Read SubIndex 1	M2S: 0x40 0x02 0x30 0x01 0x00 0x00 0x00 0x00
	S2M: 0x43 0x02 0x30 0x01 0x00 0x00 0x00 0x00

Write new configuration

Write SubIndex 1	M2S: 0x23 0x02 0x30 0x01 0x08 0x0B 0x00 0x00
	S2M: 0x60 0x02 0x30 0x01 0x00 0x00 0x00 0x00

Read new configuration

Read SubIndex 0	M2S: 0x40 0x02 0x30 0x00 0x00 0x00 0x00 0x00
	S2M: 0x4F 0x02 0x30 0x00 0x04 0x00 0x00 0x00
Read SubIndex 1	M2S: 0x40 0x02 0x30 0x01 0x00 0x00 0x00 0x00
	S2M: 0x43 0x02 0x30 0x01 0x08 0x0B 0x00 0x00

Module parameterization

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x3401	0x00	Number of Elements	Unsigned8	ro	N	depending on the components fitted	Number of entries
	0x01	1st mapped object	Unsigned32	rw	N		
	...	...	...	...	...	...	
	0x40	8th mapped object	Unsigned32	rw	N		

The index 0x3401 is supported for reasons of compatibility.

Use index 3001 to 3010 for new projects. Alternative options to write/read analog parameters:

Sub-index 0…0x40 (256 bytes):

Sub-index 0: number of sub-indices

Sub-index 1: parameter byte 0 ... 3

...

Sub-index 0x20: parameter byte 124 ... 127

Every sub-index consists of 2 data words. Enter your parameter bytes here. Every analog input or output module has 16Byte parameter data, i.e. they occupy 4 sub-indices, e.g.:

1. analog module sub-indices 1 to 4,
2. analog module sub-indices 5 to 8,
3. analog module sub-indices 9 to 12.

Digital input 8 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6000	0x00	8bit digital input block	Unsigned8	ro	N	0x01	Number of available digital 8bit input blocks
	0x01	1st input block	Unsigned8	ro	Y		1st digital input block
	...	...	...	...	...	...	...
	0x48	72nd input block	Unsigned8	ro	Y		72nd digital input block

Polarity digital input 8 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6002	0x00	8bit digital input block	Unsigned8	ro	N	0x01	Number of available digital 8bit input blocks
	0x01	1st input block	Unsigned8	rw	N	0x00	1st polarity digital input block
	...	...	...	...	...	...	...
	0x48	72nd input block	Unsigned8	rw	N	0x00	72nd polarity digital input block

Individual inverting of input polarity:

1 = input inverted
0 = input not inverted

Digital input 16 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6100	0x00	16bit digital input block	Unsigned8	ro	N	depending on the fitted components	Number of available digital 16bit input blocks
	0x01	1st input block	Unsigned16	ro	N		1st digital input block
	...	...	...	...	...	...	...
	0x24	36th input block	Unsigned16	ro	N		36th digital input block

Polarity digital input 16 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6102	0x00	16bit digital input block	Unsigned8	ro	N	depending on the compo-nents fitted	Number of available digital 16bit input blocks
	0x01	1st input block	Unsigned16	rw	N	0x0000	1st polarity digital input block
	...	...	...	...	...	...	...
	0x24	36th input block	Unsigned16	rw	N	0x0000	36th polarity digital input block

Individual inverting of input polarity:

1 = input inverted
0 = input not inverted

Digital input 32 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6120	0x00	32bit digital input block	Unsigned8	ro	N	depending on the compo-nents fitted	Number of available digital 32bit input blocks
	0x01	1st input block	Unsigned32	ro	N		1st digital input block
	...	...	...	...	...	...	...
	0x12	18th input block	Unsigned32	ro	N		18th digital input block

Polarity digital input 32 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6122	0x00	8bit digital input block	Unsigned8	ro	N	depending on the components fitted	Number of available digital 32bit input blocks
	0x01	1st input block	Unsigned32	rw	N	0x00000000	1st polarity digital input block
	...	...	...	...	...	...	...
	0x12	18th input block	Unsigned32	rw	N	0x00000000	18th polarity digital input block

Individual inverting of input polarity:

1 = input inverted
0 = input not inverted

Digital output 8 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6200	0x00	8bit digital output block	Unsigned8	ro	N	0x01	Number of available digital 8bit output blocks
	0x01	1st output block	Unsigned8	rw	Y		1st digital output block
	...	...	...	...	...	...	...
	0x48	72nd output block	Unsigned8	rw	Y		72nd digital output block

Polarity digital output 8 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6202	0x00	8bit digital output block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available digital 8bit output blocks
	0x01	1st output block	Unsigned8	rw	N	0x00	1st polarity digital output block
	...	...	...	...	...	...	...
	0x48	72nd output block	Unsigned8	rw	N	0x00	72nd polarity digital output block

Individual inverting of input channels:

1 = input inverted
0 = input not inverted

Fault mode digital output 8 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6206	0x00	8bit digital output block	Unsigned8	ro	N	0x01	Number of available digital 8bit output blocks
	0x01	1st output block	Unsigned8	rw	N	0xFF	1st error mode digital output block
	...	...	...	...	...	...	...
	0x48	72nd output block	Unsigned8	rw	N	0xFF	72nd error mode digital output block

This object indicates whether an output is set to a pre-defined error value (set in object 0x6207) in case of an internal device failure.

1 = overtake the value from object 0x6207
0 = keep output value in case of error

Fault state digital output 8 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6207	0x00	8bit digital output block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available digital 8bit output blocks
	0x01	1st output block	Unsigned8	rw	N	0x00	1st error value digital output block
	...	...	...	...	...	...	...
	0x48	72nd output block	Unsigned8	rw	N	0x00	72nd error value digital output block

Presupposed that the error mode is active, device failures set the output to the value configured by this object.

Digital output 16 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6300	0x00	16bit digital input block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available digital 16bit output blocks
	0x01	1st output block	Unsigned16	rw	N		1st digital output block
	...	...	...	...	...	...	...
	0x24	36th output block	Unsigned16	rw	N		36th digital output block

Polarity digital output 16 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6302	0x00	16bit digital input block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available digital 16bit output blocks
	0x01	1st output block	Unsigned16	rw	N	0x0000	1st polarity digital output block
	...	...	...	...	...	...	...
	0x24	36th output block	Unsigned16	rw	N	0x0000	36th polarity output block

Individual inverting of output polarity:

1 = output inverted
0 = output not inverted

Fault mode digital output 16 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6306	0x00	16bit digital input block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available digital 16bit output blocks
	0x01	1st output block	Unsigned16	rw	N	0xFFFF	1st error mode digital output block
	...	...	...	...	...	...	...
	0x24	36th output block	Unsigned16	rw	N	0xFFFF	36th error mode digital output block

This object indicates whether an output is set to a pre-defined error value (set in object 0x6307) in case of an internal device failure.

1 = overtake the value from object 0x6307
0 = keep output value in case of error

Fault state digital output 16 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6307	0x00	16bit digital input block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available digital 16bit output blocks
	0x01	1st output block	Unsigned16	rw	N	0x0000	1st error value digital output block
	...	...	...	...	...	...	...
	0x24	36th output block	Unsigned16	rw	N	0x0000	36th error value digital output block

Presupposed that the error mode is active, device failures set the output to the value configured by this object.

Digital output 32 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6320	0x00	32bit digital input block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available digital 32bit output blocks
	0x01	1st output block	Unsigned32	rw	N		1st digital output block
	...	...	...	...	...	...	...
	0x12	18th output block	Unsigned32	rw	N		18th digital output block

Polarity digital output 32 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6322	0x00	32bit digital input block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available digital 32bit output blocks
	0x01	1st output block	Unsigned32	rw	N	0x00000000	1st polarity digital output block
	...	...	...	...	...	...	...
	0x12	18th output block	Unsigned32	rw	N	0x00000000	18th polarity output block

Individual inverting of output polarity:

1 = output inverted
0 = output not inverted

Fault mode digital output 32 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6326	0x00	32bit digital input block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available digital 32bit output blocks
	0x01	1st output block	Unsigned32	rw	N	0xFFFFFFFF	1st error mode digital output block
	...	...	...	...	...	...	...
	0x48	18th output block	Unsigned32	rw	N	0xFFFFFFFF	18th error mode digital output block

This object indicates whether an output is set to a pre-defined error value (set in object 0x6307) in case of an internal device failure.

1 = overtake the value from object 0x6307
0 = keep output value in case of error

Fault state digital output 32 bit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6237	0x00	32bit digital input block	Unsigned8	ro	N	depending on the compo-nents fitted	Number of available digital 32bit output blocks
	0x01	1st output block	Unsigned32	rw	N		1st error value digital output block
	...	...	...	...	...	...	...
	0x12	18th output block	Unsigned32	rw	N		18th error value digital output block

Presupposed that the error mode is active, device failures set the output to the value configured by this object.

Analog input

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6401	0x00	2byte input block	Unsigned8	ro	N	depending on the compo-nents fitted	Number of available analog inputs
	0x01	1st input channel	Unsigned16	ro	Y		1st analog input channel
	...	...	...	...	...	...	...
	0x24	24th input channel	Unsigned16	ro	Y		24th analog input channel

Analog output

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6411	0x00	2byte output block	Unsigned8	ro	N	depending on the compo-nents fitted	Number of available analog outputs
	0x01	1st output channel	Unsigned16	ro	Y		1st analog output channel
	...	...	...	...	...	...	...
	0x24	24th output channel	Unsigned16	ro	Y		24th analog output channel

Analog input interrupt trigger

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6421	0x00	Number of Inputs	Unsigned8	ro	N	depending on the compo-nents fitted	Number of available analog inputs
	0x01	Trigger 1st input channel	Unsigned8	rw	N	0x07	Input interrupt trigger for 1st analog input channel
	...	...	...	...	...	...	...
	0x24	Trigger 24th input channel	Unsigned8	rw	N	0x07	Input interrupt trigger for 24th analog input channel

This object determines which events shall cause an interrupt for a specific channel. Bits set in the list below refer to the interrupt trigger.

Bit no.	Interrupt trigger
0	Upper limit exceeded 6424
1	Input below lower limit 6425
2	Input changed by more than negative delta 6426
3 to 7	Reserved

Analog input interrupt source

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6422	0x00	Number of Interrupt	Unsigned8	ro	N	0x01	Number of interrupt source bank
	0x01	Interrupt source bank	Unsigned32	ro	N	0x00000000	Interrupt source bank 1

This object defines the channel that is responsible for the Interrupt. Bits set refer to the number of the channel that caused the Interrupt. The bits are automatically reset, after they have been read by a SDO or send by a PDO.

1 = Interrupt produced

0 = Interrupt not produced

Analog input interrupt enable

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x6423	0x00	Global interrupt enable	Boolean	rw	N	FALSE ("0")	Activates the event-driven transmission of PDOs with analog inputs

Although the analog inputs are -acc. to CANopen - per default set to the transmission type 255 (event triggered) in the TxPDO2, the "event" (the alteration of an input value) is suppressed by the event control in object 0x6423 in order to prevent the bus from being swamped with analog signals.

Before activation, it is convenient to parameterize the transmission behavior of the analog PDOs:

- inhibit time (object 0x1800ff, sub-index 3)
- limit value monitoring (objects 0x6424 + 0x6425)
- delta function (object 0x6426)

Analog input interrupt upper limit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6424	0x00	Number of Inputs	Unsigned8	ro	N	depending on the compo-nents fitted	Number of available analog inputs
	0x01	Upper limit 1st input channel	Unsigned32	rw	N	0x00000000	Upper limit value for 1st analog input channel
	...	...	...	...	...	...	...
	0x24	Upper limit 24th input channel	Unsigned32	rw	N	0x00000000	Upper limit value for 24th analog input channel

Values unequal to zero are activating the upper limit value for this channel. A PDO is then transmitted when the upper limit value is exceeded. In addition, the event trigger has to be active (object 0x6423). The data format corresponds to that of the analog inputs.

Analog input interrupt lower limit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6425	0x00	Number of Inputs	Unsigned8	ro	N	depending on the compo-nents fitted	Number of available analog inputs
	0x01	Lower limit 1st input channel	Unsigned32	rw	N	0x00000000	Lower limit value for 1st analog input channel
	...	...	...	...	...	...	...
	0x24	Lower limit 24th input channel	Unsigned32	rw	N	0x00000000	Lower limit value for 24th analog input channel

Values unequal to zero are activating the lower limit value for this channel. A PDO is then transmitted when the lower limit value is underrun. In addition, the event trigger has to be active (object 0x6423). The data format corresponds to that of the analog inputs.

Analog input interrupt delta limit

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6426	0x00	Number of Inputs	Unsigned8	ro	N	depending on the compo-nents fitted	Number of available analog inputs
	0x01	Delta value 1st input channel	Unsigned32	rw	N	0x00000002	Delta value for 1st analog input channel
	...	...	...	...	...	...	...
	0x24	Delta value 24th input channel	Unsigned32	rw	N	0x00000002	Delta value for 24th analog input channel

Values unequal to zero are activating the delta function for this channel. A PDO is then transmitted when the value has been changed for more than the delta value since the last transmission. In addition, the event trigger has to be active (object 0x6423). The data format corresponds to that of the analog inputs (The delta function accepts only positive values).

Fault mode analog output

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6443	0x00	Analog output block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available analog outputs
	0x01	1st analog output block	Unsigned8	rw	N	0xFF	1st error mode analog output block
	...	...	...	...	...	...	...
	0x24	36th analog output block	Unsigned8	rw	N	0xFF	36th error mode analog output block

This object indicates whether an output is set to a pre-defined error value (set in object 0x6444) in case of an internal device failure.

0 = current value

1 = set to error value 0x6444

Fault state analog output

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6444	0x00	16bit digital input block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available analog output blocks
	0x01	1st analog block	Unsigned16	rw	N	0x0000	1st analog output block
	...	...	...	...	...	...	...
	0x24	36th analog block	Unsigned16	rw	N	0x0000	36th analog output block

Presupposed that the corresponding error (0x6443) is active, device failures set the output to the value configured by this object.

SDO Abort 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 an 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)

IM 353CAN - CANopen slave - Emergency Object

Outline

The VIPA CAN-Bus coupler is provided with the emergency object to notify other devices connected to the CANopen bus about internal error events or CAN-Bus errors. It has a high priority and gives you important information about the states of device and network.

Note!

We strongly recommend to analyze the emergence object - it is an important information pool!

Telegram structure

The emergency telegram has always a length of 8Byte. It starts with 2Byte error code followed by the 1Byte error register and closes with 5Byte additional code.

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

IM 353CAN - CANopen slave - NMT - network management

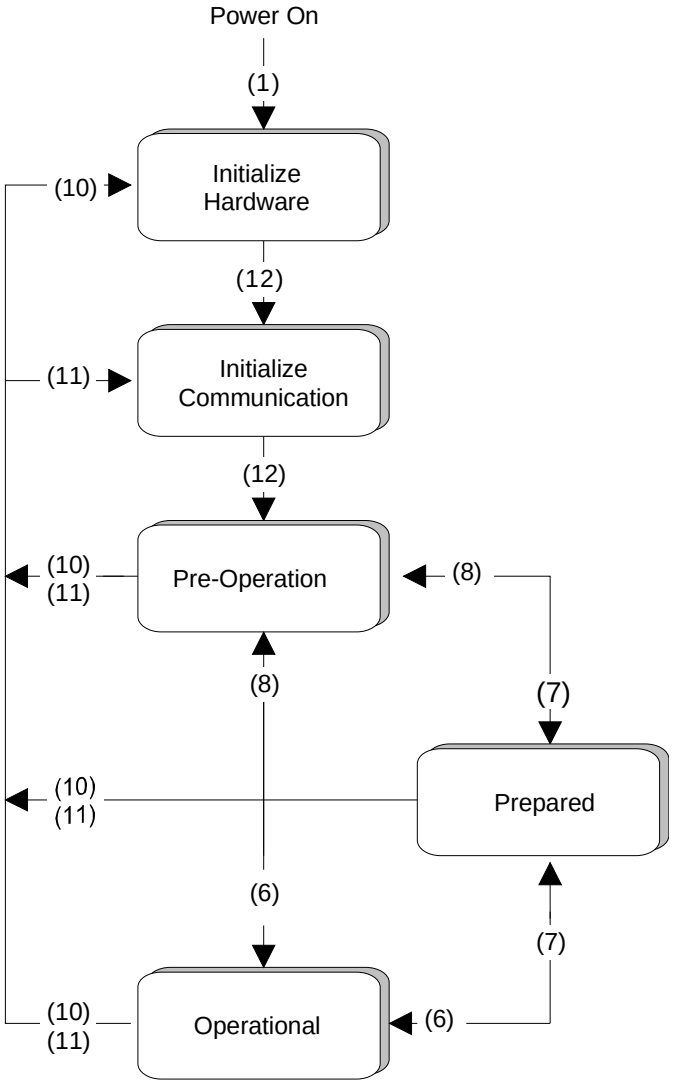
Network management (NMT) provides the global services specifications for network supervision and management. This includes the login and logout of the different network devices, the supervision of these devices as well as the processing of exceptions.

NMT service messages have the COB identifier 0000h. An additional module-ID is not required. The length is always 2 data bytes.

The 1st data byte contains the NMT command specifier: **CS**.

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

The following picture shows an overview over all CANopen status changes and the corresponding NMT command specifiers:



(1): The initialization state is reached automatically after start-up.

(6): "Start_Remote_Node" (CS: 0x01)
Starts the module, releases outputs and starts the PDO transmission.

(7): "Stop_Remote_Node" (CS: 0x02)
Outputs are switching into error state, SDO and PDO are switched off.

(8): "Enter_Pre-operational_State" (CS:0x80)
Stops PDO transmission, SDO still active.

(10): "Reset_Node" (CS:0x81)
Executes reset. All objects are set back to PowerOn defaults.

(11): "Reset_Communication" (CS:0x82)
Executes reset of the communication functions. Objects 0x1000 - 0x1FFF are set back to PowerOn defaults.

(12): After initialization the state "pre-operational is automatically reached - here the boot-up message is send.

Node Guarding

The bus coupler also supports the Node Guarding object as defined by CANopen to ensure that other devices on the bus are supervised properly.

Node Guarding operation is started when the first guard requests (RTR) is received from the master. The respective COB identifier is permanently set to 0x700 + module-ID at variable 0x100E in the object directory. If the coupler does not receive a guard request message from the master within the "guard time" (object 0x100C) when the node guarding mode is active the module assumes that the master is not operating properly. When the time determined by the product of "guard time" (0x100C) and "life-time factor" (0x100D) has expired, the module will automatically assume the status "pre-operational".

When either the "guard time" (object 0x100C) or the "life-time factor" (0x100D) has been set to zero by an SDO download from the master, the expiry of the guard time is not monitored and the module remains in its current operating mode.

Heartbeat

The VIPA CAN coupler also supports the Heartbeat Mode in addition to Node Guarding.

When a value is entered into index 0x1017 (Heartbeat Producer Time) then the device status (Operational, Pre-Operational,...) of the bus coupler is transferred by means of the COB identifier (0x700+module-ID) when the heartbeat timer expires.

The Heartbeat Mode starts automatically as soon as the index 1017h contains a value that is larger than 0.

Technical data

CANopen slave
IM 353CAN

Electrical data	VIPA 353-1CA00
Power supply	DC 24V (20.4 ... 28.8) via front from ext. power supply
Current consumption	max. 0.7A
Output current backplane bus	max. 3.5A
Isolation	≥ AC 500V
Status indicator	by means of LEDs located on the front
Connectors/interfaces	9pin D-type (socket) CAN-Bus connection
CAN-Bus interface	
Connection	9pin D-type plug
Network topology	Linear bus, active bus termination at one end, tap lines permitted.
Medium	Screened three-core cable, unscreened cable permitted - depending on environment.
Data transfer rate	10kBaud to 1MBaud
Max. overall length	1000m at 50kBaud without repeaters
Digital inputs/outputs	Any combination of max. of 32 I/O modules per coupler.
Combination with peripheral modules	
max. no. of modules	32 (depending on current consumption)
max. inputs/outputs	80Byte each (80Byte = 10 PDOs à 8Byte)
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x78
Weight	80g

Appendix

A Index

M.Stich